CC1 - LIF6 - mardi 16/11/15 - noté sur 20 - durée : 1h20

Aucun document autorisé ; calculatrices, téléphones et ordinateurs interdits. Le barème pourra être modifié.
Le sujet n'est pas à rendre : notez proprement vos réponses sur votre copie.

Exercice 1 (Codage des programmes en machine — 6 pts)**.** Un processeur est doté d'une mémoire de 256 o : à chaque adresse correspond une case de 1 octet. Il dispose de 4 registres généraux, indicés de 0 à 3 et notés `R0...R3`. Les instructions sont codées sur 1 ou 2 octets :

		1er octet								2d octet
assembleur	action	7	6	5	4	3	2	1	0	7...0
`HALT`	Met fin à l'exécution	0	0	0	0	0	0	0	0	—
`LHS RD,RS`	`RD <- RS * 2`	0	0	0	1	`RD`		`RS`		—
`ADD RD,RS`	`RD <- RD + RS`	0	0	1	0	`RD`		`RS`		—
`ADDI RD,imm2`	`RD <- RD + imm2`	0	0	1	1	`RD`		`imm2`		—
`SUB RD,RS`	`RD <- RD - RS`	0	1	0	0	`RD`		`RS`		—
`SUBI RD,imm2`	`RD <- RD - imm2`	0	1	0	1	`RD`		`imm2`		—
`LD RD,label`	`RD <- Mem[adr]`	0	1	1	0	`RD`		0	0	`adr`
`ST RS,label`	`Mem[adr] <- RS`	0	1	1	1	`RS`		0	0	`adr`
`BR label`	`PC <- adI+dec5`	1	1	0	`dec5`					—
`BRZ label`	si `cond` alors `PC <- adI+dec5`	1	1	1	`dec5`					—

Dans ce tableau :

- `RD` désigne un registre de destination et `RS` un registre source (`R0...R3`).
- `imm2` est un entier naturel codé sur 2 bits.
- `adr` est une adresse mémoire, et `Mem[adr]` la case mémoire correspondante.
- `dec5` est un décalage en complément à deux sur 5 bits.
- `adI` est l'adresse de l'instruction courante, et `adI+dec5` est déterminé d'après le `label`.
- pour `BRZ`, la condition de branchement `cond` est que la dernière valeur rangée dans l'un des registres `R0...R3` était nulle.

On considère le programme suivant :

```
        LD R0,zero   // R0 <- Mem[zero] = 0
        LD R1,n      // R1 <- Mem[n]
        LD R2,vingt  // R2 <- Mem[vingt] = 20
loop:   BRZ endloop  // si R2 = 0 aller à endloop
        ADD R0,R1    // R0 <- R0+R1
        SUBI R2,1    // R2 <- R2-1
        BR loop      // retour au début de boucle
endloop: HALT
n:      N            // valeur décimale
zero:   0            // constante 0
vingt:  20           // constante décimale 20
```

Remarquez que l'on a noté `N` la valeur se trouvant à l'adresse désignée par l'étiquette `n`. A la fin de l'exécution du programme (au niveau de l'instruction `HALT`), `R0` a la valeur $20 \times$ `N`.

Q. 1 (1 pt)**.** Combien d'instructions (en tout) vont être exécutées par ce programme ? Justifiez votre résultat en donnant votre calcul.

Q. 2 (1 pt)**.** En supposant que le premier octet du programme est stocké à l'adresse 35_H, donnez l'adresse en hexadécimal de toutes les instructions du programme.

Q. 3 (2 pts)**.** Calculez `dec5` pour les instructions `BRZ endloop` et `BR loop`, et donnez leurs codages en hexadécimal. Justifiez vos résultats en donnant vos calculs.

Q. 4 (2 pts)**.** En remarquant que $20 \times \texttt{N} = 2^2 \times \texttt{N} + 2^4 \times \texttt{N}$, donnez un morceau de programme en langage d'assemblage pour multiplier par 20 le contenu de `R0`, en n'utilisant que des instructions `LHS` et `ADD`.

Exercice 2 (Nombres à virgule flottante — 6 pts)**.** On travaille sur une machine où les flottants binaires sont codés sur 12 bits. On considère le code :

$c =$

11	10	9	8	7	6	5	4	3	2	1	0
s	b				m						

Le nombre représenté est $f(c) = (-1)^s \times (1, m)_2 \times 2^e$, avec $e = (b)_2 - 2^3$ si $1 \leq (b)_2 \leq 14$, (on ne se préoccupe pas du codage des sous-normaux ni des valeurs exceptionnelles ici).

Q. 5 (2 pts)**.** Quel nombre x_1 est représenté par le code hexadécimal $c = 637_H$ si on le regarde comme le codage d'un flottant ? Donnez votre résultat sous la forme d'une représentation positionnelle à virgule en décimal.

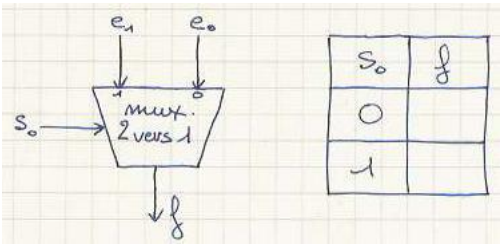
Q. 6 (1 pt)**.** Quel est le nombre flottant x_2 immédiatement supérieur à x_1 ?
- Donnez votre résultat sous la forme $x_2 = (-1)^s \times (1, m)_2 \times 2^e$,
- puis donnez le codage de x_2 en hexadécimal.

Q. 7 (1 pt)**.** Comment est représenté le nombre $(13/3)_{10}$ en représentation positionnelle à virgule en binaire ? Détaillez un peu vos calculs.

Q. 8 (2 pts)**.** Comment peut-on représenter le nombre $(13/3)_{10}$ dans le format flottant binaire décrit ci-dessus ? Vous effectuerez un arrondi au plus proche, et donnerez votre résultat en hexadécimal.

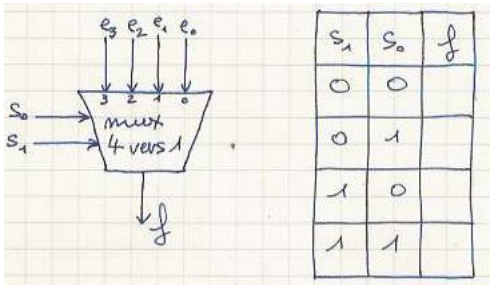
Exercice 3 (Circuits combinatoires — 7 pts)**.**

Q. 9 (1 pts)**.** On considère le multiplexeur 2 vers 1 ci dessous. Complétez le tableau pour exprimer f (qui peut prendre pour valeur e_0 ou e_1) en fonction de s_0. Exprimez f par une forme normale disjonctive en fonction de s_0, e_0 et e_1. Déduire un circuit pour ce multiplexeur, en utilisant 2 portes AND, 1 porte OR et 2 portes NOT.



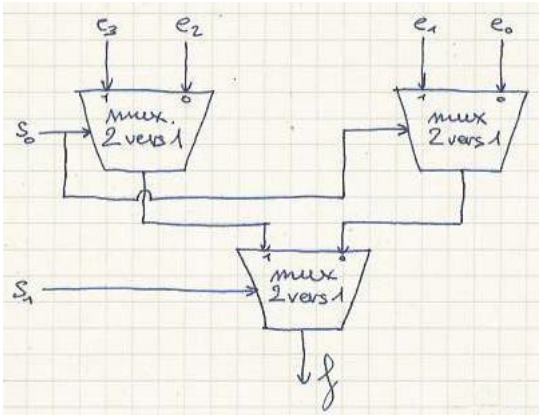
s_0	f
0	
1	

Q. 10 (1,5 pts)**.** On considère le multiplexeur 4 vers 1 ci dessous. Complétez le tableau pour exprimer f (qui peut prendre pour valeur e_0, e_1, e_2 ou e_3) en fonction de s_1 et s_0. Exprimez f par une forme normale disjonctive, et déduire directement un circuit pour ce multiplexeur. Combien de portes AND, OR et NOT utilise votre circuit ?



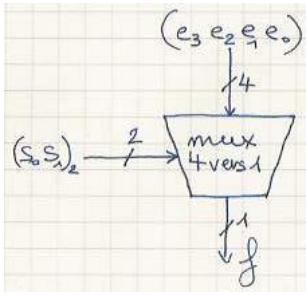
s_1	s_0	f
0	0	
0	1	
1	0	
1	1	

Q. 11 (1,5 pts)**.** On considère maintenant le circuit ci-dessous, et on note x_1 la sortie du multiplixeur 2 vers 1 qui sélectionne soit e_3 soit e_2, et x_0 celle de celui qui sélectionne soit e_1 soit e_0. Complétez le tableau (colonnes x_1, x_0 et f), puis justifiez le fait qu'il s'agisse bien d'un multiplexeur 4 vers 1. Combien de portes AND, OR et NOT utilise ce circuit (détaillez un peu les décomptes) ?



S_1	S_0	x_1	x_0	f
0	0			
0	1			
1	0			
1	1			

Q. 12 (2 pts)**.** On utilise le schéma suivant pour un multiplexeur 4 vers 1.



Donnez le schéma d'un multiplexeur 8 vers 1 en utilisant :

- 2 multiplexeurs 4 vers 1,
- 1 multiplexeur 2 vers 1.

Combien de portes AND, OR et NOT utilise ce circuit (détaillez un peu les décomptes) ?

Q. 13 (1 pt + 1 pt bonus)**.** Dressez une table de vérité, similaire à celles des questions 9 et 10 pour une multiplexeur 8 vers 1. Si vous utilisiez cette table pour obtenir une forme normale conjonctive pour le multiplexeur, puis que vous en déduisiez un circuit combinatoire, combien de portes AND, OR et NOT utiliseriez vous ?