3TC31 (ex. INF107)

Examen intermédiaire (première partie)
Éléments de correction

2025–2026

Questions de cours (4 points / 5 minutes)

Question 1 (2 points)

La valeur, en décimal, du nombre représenté en complément à 2 sur 4 bits par la valeur 1111 est -1.

En effet, on se souvient que la valeur d'un nombre représenté en complément à 2 sur n bits $(a_{n-1}a_{n-2}\dots a_1a_0)$ est $-a_{n-1}2^{n-1} + \sum_{i=0}^{n-2} a_i 2^i$ soit ici $-1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = -1$.

Question 2 (2 points)

À chaque front montant sur le signal d'horloge (`clk`), la bascule échantillonne la valeur de l'entrée `D` et recopie cette valeur sur la sortie `Q`. Le reste du temps, la sortie `Q` garde sa valeur quelque soit l'évolution de l'entrée `D`.

Exercice 1 : Logique combinatoire (6 points / 10 minutes)

Question 3 (3 points)

$E_{A>B}$	$E_{A<B}$	A_i	B_i	S_i	$S_{A>B}$	$S_{A<B}$
0	0	0	0	0	0	0
0	0	0	1	1	0	1
0	0	1	0	1	1	0
0	0	1	1	1	0	0
0	1	0	0	0	0	1
0	1	0	1	1	0	1
0	1	1	0	0	0	1
0	1	1	1	1	0	1
1	0	0	0	0	1	0
1	0	0	1	0	1	0
1	0	1	0	1	1	0
1	0	1	1	1	1	0

Explications :

- Les lignes 5 à 8 du tableau (cas $E_{A<B} = 1$) représentent le cas où la comparaison des bits de poids forts a déjà permis de conclure que $B > A$. Il faut donc faire sortir sur S_i la valeur de B_i et continuer à indiquer aux opérateurs travaillant sur les bits de poids faibles que $B > A$, d'où $S_{A>B} = 0$ et $S_{A<B} = 1$
- Les lignes 9 à 12 du tableau (cas $E_{A>B} = 1$) représentent le cas où la comparaison des bits de poids forts a déjà permis de conclure que $A > B$. Il faut donc faire sortir sur S_i la valeur de A_i et continuer à indiquer aux opérateurs travaillant sur les bits de poids faibles que $A > B$, d'où $S_{A>B} = 1$ et $S_{A<B} = 0$
- Les lignes 1 à 4 du tableau (cas $E_{A>B} = 0$ et $E_{A<B} = 0$) représentent le cas où la comparaison des bits de poids forts n'a pas déjà permis de conclure que A est plus grand que B ou l'inverse.
  - Si $A_i = B_i$ (lignes 1 et 4), il n'est toujours pas possible de savoir lequel de A et B est le plus grand, donc $S_{A>B} = 0$ et $S_{A<B} = 0$. On sait qu'en sortie on doit avoir le nombre le plus grand, donc S_i doit valoir soit A_i soit B_i. Comme $A_i = B_i$, on a simplement $S_i = A_i = B_i$.
  - Si $A_i \neq B_i$ (lignes 2 et 3), il devient possible de connaître lequel de A et B est le plus grand : A si $A_i = 1$, B si $B_i = 1$. On l'indique donc en sortie pour les opérateurs suivants ($S_{A>B} = 0$ et $S_{A<B} = 1$

si $B > A$ ou $S_{A>B} = 1$ et $S_{A<B} = 0$ si $A > B$), et on sort sur S_i soit A_i si $A > B$ soit B_i si $B > A$, et donc dans les deux cas $S_i = 1$.

Question 4 (3 points)

$S_i = E_{A>B} \cdot A_i + E_{A<B} \cdot B_i + \overline{E_{A>B}} \cdot \overline{E_{A<B}} \cdot (A_i + B_i)$

$S_{A>B} = E_{A>B} + \overline{E_{A>B}} \cdot \overline{E_{A<B}} \cdot A_i \cdot \overline{B_i}$

$S_{A<B} = E_{A<B} + \overline{E_{A>B}} \cdot \overline{E_{A<B}} \cdot \overline{A_i} \cdot B_i$

Exercice 2 : Processeur RISC-V (10 points / 15 minutes)

Question 5 (8 points)

pc	instr	rs1	rs2	rd	op	ALUsrc	imm	op1	op2	res / Addr	WData	RData	write	load	store	wrdata
0	0x00210193	2	x	3	+	0	2	8	2	10	x	x	1	0	0	10
4	0x0041A123	3	4	x	+	0	2	10	2	12	16	x	0	0	1	x

Question 6 (2 points)

```
addi x1, x0, 20 ; x1 = x0 + 20 = 0 + 20 = 20
lw   x2, 0(x1)  ; x2 = RAM[x1] = RAM[20] = i
addi x2, x2, 1  ; x2 = x2 + 1 = i + 1
sw   x2, 0(x1)  ; RAM[x1] = RAM[20] = i = x2 = i + 1
```

Cet exercice est une version simplifiée des premières questions du devoir maison sur l'assembleur.

La première instruction permet de charger l'adresse de la variable `i`, c'est-à-dire la valeur 20, dans le registre `x1`, en faisant une addition du contenu du registre `x0` (toujours 0) avec l'immédiat 20. Dans le devoir maison, vous avez vu une construction plus générale pour traiter le cas où la valeur à charger n'est pas représentable directement dans un immédiat.

La deuxième instruction charge la valeur stockée en mémoire à l'adresse contenue dans le registre `x1` (20) vers le registre `x2`. On charge donc ici la valeur actuelle de la variable `i` vers le registre `x2`.

La troisième instruction incrémente le contenu du registre `x2` (donc la valeur de `i`).

La dernière instruction écrit la valeur du registre `x2` (la valeur de `i` incrémentée de 1) en mémoire à l'adresse contenue dans `x1` (c'est-à-dire 20, soit l'adresse de la variable `i`).