CCF - LIF6 Printemps 2016
Durée prévue : 1h20

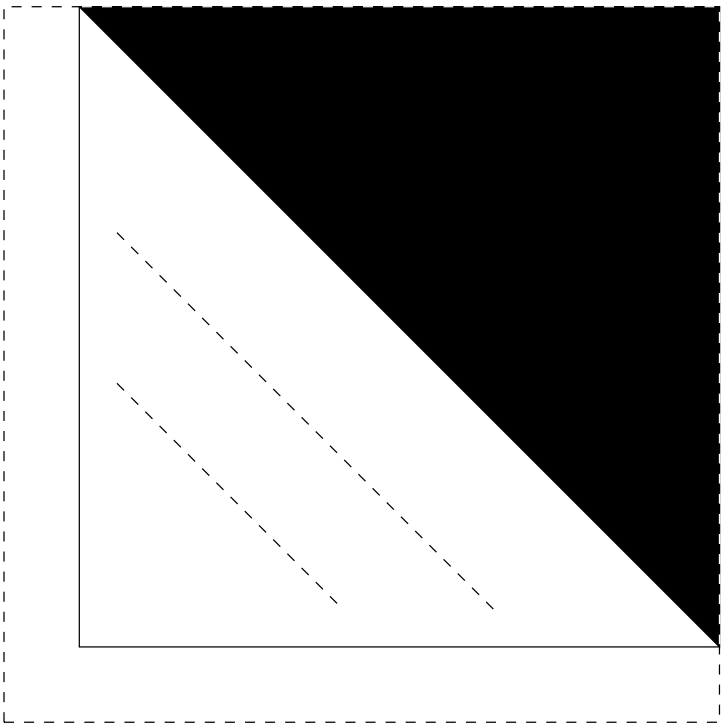
Aucun document autorisé. Calculatrices, téléphones et ordinateurs portables interdits.

N'oubliez pas de noter votre nom, votre prénom et votre numéro d'étudiant sous le cache rabattable. Débrouillez vous pour que le cache soit rabattu et collé !

Justifiez précisément vos réponses dans les emplacements prévus.

Le barème est donné à titre indicatif.

Ne rien écrire dans ce cadre.
Partie 1 :
Partie 2 :
Partie 3 :
Partie 4 :



1 Mémoire cache et petits calculs (4 pts)

Une machine est dotée d'un processeur avec un niveau de cache et une mémoire RAM : le processeur et son cache tournent à 800 MHz, le bus entre le cache et la mémoire à 200 MHz. On considère un programme dont 25% des instructions sont des accès à la mémoire, et le taux de cache-misses pour ces accès est de 5%. Le coût d'un accès en cas de cache-miss est de 5 cycles du bus, mais il n'y a pas de pénalité en cas de cache-hit. Si le programme était exécuté sur une machine parfaite à 800 MHz sans aucun cache-miss, alors l'exécution durerait 400 ms.

1. (2 points) Donner le temps d'exécution du programme sur la machine (réelle) considérée ?

   .....

   .....

   .....

2. (1 point) La fréquence du processeur passe à 1600 MHz, mais celle du bus reste à 200 MHz : que devient le temps d'exécution total ?

   .....

   .....

   .....

3. (1 point) Quel est (en %) le gain en terme de temps de calcul provoqué par le doublement de la fréquence du processeur ? Que remarquez-vous ? Comment le programmeur peut-il améliorer encore la vitesse d'exécution ?

   .....

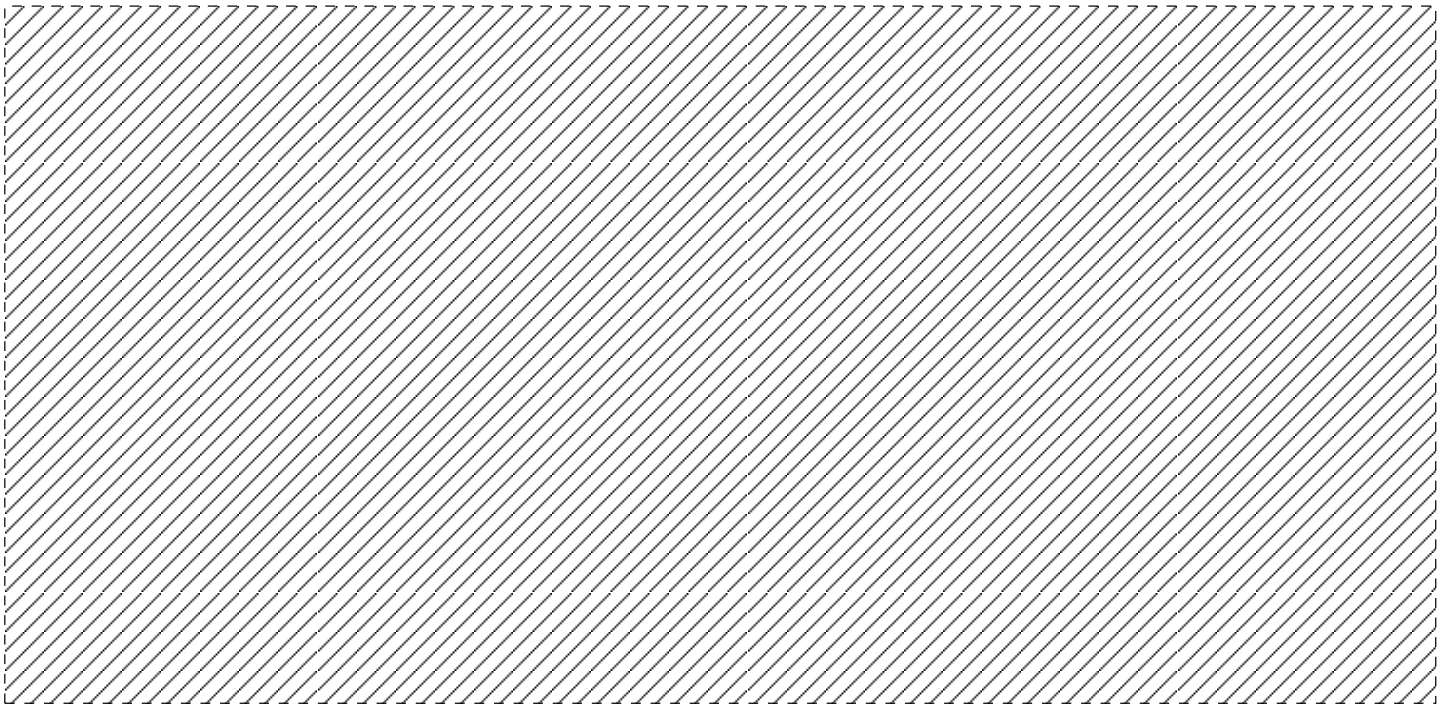
   .....

   .....

2 Pipeline (5 points)

On considère un pipeline à 4 étages : FE (*fetch*, chargement), DE (*decode*, décodage), EX (*execute*, exécution) et SR (*store register*, stockage du résultat en registre). Le processeur dispose de 16 registres, `R0`, ..., `R15`. On ne travaille ici qu'avec une seule instruction, `ADD RD, RS1, RS2`, qui effectue `RD` $\leftarrow$ `RS1` + `RS2`.

On rappelle qu'une instruction i_2 présente une dépendance de donnée vis-à-vis d'une instruction i_1 si le résultat produit par i_1 est un opérande source de i_2 : on dit que i_2 dépend de i_1. On suppose dans un premier temps qu'une instruction reste bloquée à l'étage DE tant qu'une instruction dont elle dépend n'a pas passé sa phase SR.



1. (1 point) Représentez le traitement des instructions dans le pipeline en remplissant avec FE, DE, EX ou SR dans les cases du tableau suivant.

Instruction	cycle 1	cycle 2	cycle 3	cycle 4	cycle 5	cycle 6	cycle 7	cycle 8	cycle 9	cycle 10
`ADD R1,R2,R3`										
`ADD R4,R3,R5`										
`ADD R7,R6,R2`										
`ADD R8,R1,R2`										

2. (1 point) Même question avec les instructions du tableau suivant.

Instruction	cycle 1	cycle 2	cycle 3	cycle 4	cycle 5	cycle 6	cycle 7	cycle 8	cycle 9	cycle 10
`ADD R1,R2,R3`										
`ADD R5,R1,R4`										
`ADD R6,R2,R5`										
`ADD R7,R1,R6`										

3. (1 point) Si toutes les instructions peuvent se terminer sans jamais rester bloquées à l'étage DE, combien de cycles va nécessiter l'exécution d'un bloc de k instructions dans le pipeline décrit ?

   .....

   .....

   .....

4. (1 point) Si, dans un bloc de k instructions, chaque instruction dépend de l'instruction qui la précède dans le programme, en combien de cycle va s'exécuter ce bloc ?

   .....

   .....

   .....

5. (1 point) On suppose ici qu'une instruction reste bloquée à l'étage DE tant qu'une instruction dont elle dépend n'a pas passé sa phase EX. Représentez à nouveau le traitement des instructions suivantes dans le pipeline :

Instruction	cycle 1	cycle 2	cycle 3	cycle 4	cycle 5	cycle 6	cycle 7	cycle 8	cycle 9	cycle 10
`ADD R1,R2,R3`										
`ADD R5,R1,R4`										
`ADD R6,R2,R5`										

   Quel est, avec cette hypothèse, le temps d'exécution d'une suite de k instructions dans laquelle chaque instruction dépend de la précédente ?

   .....

   .....

3 Programmation du LC3 (8 points)

Programme mystère : On considère le programme ci-dessous, qui s'assemble et s'exécute sans problème sur le LC3 : il affiche un certain nombre d'étoiles puis retourne à la ligne... Vous devez en expliquer le fonctionnement.

```
1           .ORIG x3000
2   ; Programme principal
3           AND R1, R1, 0   ;
4   loop:   ADD R2, R1, 10  ;
5           BRz endloop     ;
6           LD R2, loop     ;
7           ADD R2, R2, -1  ;
8           ST R2, loop     ;
9           LD R0, etoile   ; charge le code ASCII de '*' dans R0
10          OUT             ; affiche le caractère '*'
11          BR loop         ;
12  endloop: LD R0, etoile  ; charge le code ASCII de '*' dans R0
13          OUT             ; affiche le caractère '*'
14          HALT            ; rend la main à l'OS
15  ; Partie dédiée aux données
16  etoile:  .FILL 42       ; code ASCII du caractère '*'
17  carret:  .FILL 13       ; code ASCII du retour-chariot
18          .END
```

1. (1 point) Donnez le codage binaire de l'instruction `ADD R2, R1, 10` en découpant bien chacun des champs.

   .....

   .....

2. (1,5 points) Quel est l'effet des instructions aux lignes 6 à 8 sur l'instruction à la ligne 4 ?

   .....

   .....

   .....

   .....

3. (1 point) Une boucle est mise en place entre les lignes 4 et 11 du programme : à chaque itération, une étoile est affichée. Exprimez précisément la condition de sortie de la boucle.

   .....

   .....

   .....

4. (1 point) Combien d'étoiles sont affichées par le programme ci-dessous à l'exécution ? Donnez le calcul.

   .....

   .....

Débogage : On considère le programme ci-dessous, qui s'assemble sans message d'erreur, est s'exécute sur le LC3 ; par contre, l'exécution n'aboutit jamais au `HALT` : le programme boucle indéfiniment...

```
1           .ORIG x3000
2   ; Programme principal
3           LD R0, n        ; R0 <- mem[n]
4           JSR fois20      ; appel à la routine fois20 : R1 <- 20 * R0
5           ST R1, r        ; mem[r] <- R1
6           HALT            ; rend la main à l'OS
7   n:      .FILL 7         ; Donnée, entier 7
8   r:      .BLKW 1         ; Résultat
9
10  ; Routine pour effectuer R1 <- 20 * R0
11  ; Attention, R2 est écrasé par l'appel.
12  fois20: JSR fois10      ; R1 <- 10 * R0
13          ADD R1, R1, R1  ; R1 <-  2 * R1 = 20 * R0
14          RET             ; retour à l'appelant
15
```

```
16  ; Routine pour effectuer R1 <- 10 * R0
17  ; Attention, R2 est écrasé par l'appel.
18  fois10: ADD R1, R0, R0  ; ........................................
19          ADD R2, R1, R1  ; ........................................
20          ADD R2, R2, R2  ; ........................................
21          ADD R1, R1, R2  ; ........................................
22          RET             ; retour à l'appelant
23          .END
```

1. (2 points) Identifiez la raison pour laquelle le programme entre dans une boucle infinie : expliquez clairement les raisons du problème.

   .....

   .....

   .....

   .....

2. (1,5 points) Quelle solution classique permet de remédier à ce problème ?

   .....

   .....

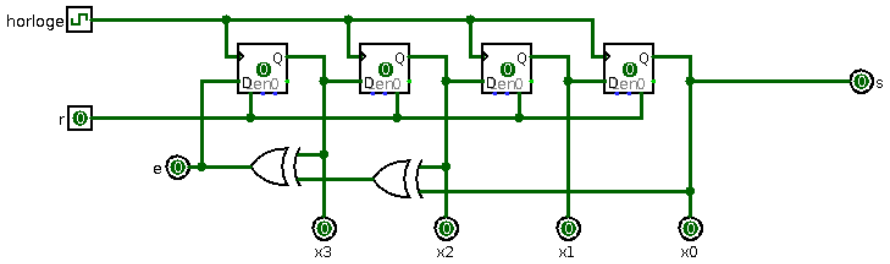
   .....

   .....

3. (1 point) `fois10` effectue `R1 <- 10 * R0` : ajouter des commentaires dans le programme pour justifier cela.

4 Analyse d'un circuit séquentiel (3 pts)

Le circuit séquentiel ci-dessous comporte quatre flip-flops régies par un signal d'horloge commun, et deux portes XOR. On note $(x_3x_2x_1x_0)$ l'état courant du circuit (les bits `x3`, `x2`, `x1`, `x0` du circuit). L'entrée r permet, lorsqu'elle passe à 1, de remettre le circuit dans son état initial (1000). On s'intéresse à la sortie s du circuit.



1. (2 points) On suppose que le circuit est dans son état initial au cycle 0, et que l'entrée r reste à 0. Complétez le tableau ci-dessous en donnant à chaque cycle l'état courant du circuit, et la valeur des sorties e et s. Quelle est la période (en nombre de cycles) du signal s ? Justifiez votre réponse.

cycle	e	x_3	x_2	x_1	x_0	s
0		1	0	0	0	
1						
2						
3						
4						
5						
6						
7						
8						
9						

   .....

   .....

   .....

   .....

   .....

   .....

2. (1 point) Si on avait implanté ce circuit séquentiel sous la forme d'un automate de Moore, combien de flip-flops aurait-on dû utiliser ? Justifiez votre réponse.

   .....

   .....