Examen final – Jeudi 15 janvier 2015
Architecture des ordinateurs – L1

Durée de l'épreuve : 2h00
Aucun document autorisé.

Il est fait appel, pour les questions de cet examen, à votre esprit de synthèse : il ne s'agit pas de répondre par une unique phrase mais bien d'articuler vos connaissances et votre compréhension du cours dans le contexte de la question afin d'être le plus précis possible dans votre propos. **Enfin**, *pensez à soigner l'écriture*, le correcteur est, comme vous, limité en capacités de déchiffrage !

I Circuit combinatoire – [5 pts]

Soit la fonction définie par la table de vérité suivante :

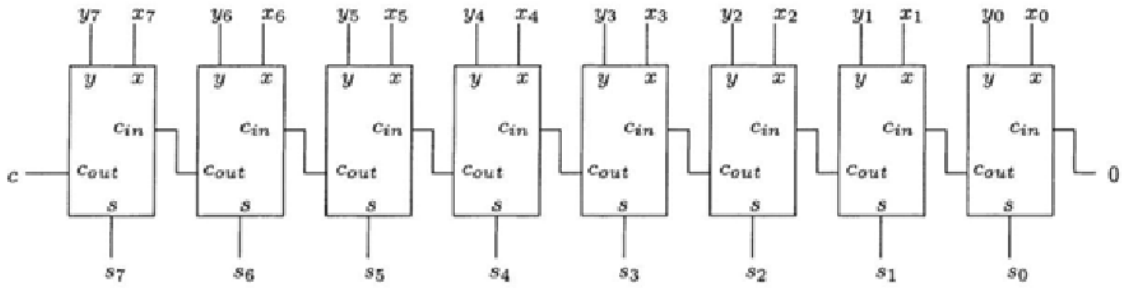
x	y	z	a	b
0	0	0	0	0
0	0	1	1	1
0	1	0	1	0
0	1	1	0	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	0

Donnez une implantation sous une forme graphique de cette fonction en utilisant uniquement des portes NAND et NOT.

II Additionneur spéculatif – [5 pts]

Rappelez la table de vérité de l'additionneur 1-bit qui utilise en entrée deux opérandes *x* et *y* et une retenue *cin* et calcule une somme *s* et la nouvelle retenue *cout*.

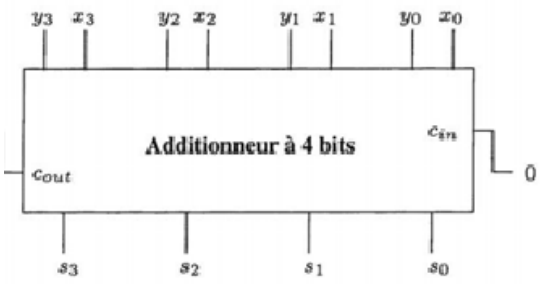
Afin de produire un additionneur 8 bits, on utilise généralement 8 additionneurs 1 bit en cascade selon le schéma suivant :



Cette mise en œuvre n'est pas efficace, en effet, la sortie et la retenue du dernier additionneur (en position 7) est déterminée en partie par les entrées du premier

additionneur (en position 0). Le signal, pour être disponible, doit donc traverser l'ensemble des additionneurs, accumulant les retards.

Il existe plusieurs solutions quand on souhaite construire de gros additionneurs (32 ou 64 bits par exemple). Une de ces solutions que l'on nomme *additionneur spéculatif* est d'utiliser des additionneurs 4 bits ad-hoc et, plutôt que d'attendre la propagation de la retenue, de calculer la somme des 4 bits de poids fort (bits s4-s7) deux fois, une fois avec une retenue à 0 et une fois avec une retenue à 1. Il ne reste plus alors qu'à choisir, quand la retenue est disponible sur les 4 bits de poids faible (s0-s3), quelle addition des 4 bits de poids fort est correcte. Cette méthode nécessite donc 3 additionneurs 4 bits et un peu de logique combinatoire pour sélectionner les sorties correctes.



En supposant que vous disposez d'un bloc additionneur 4 bits comme celui représenté ci-dessus, dessinez le circuit qui permet de mettre en œuvre l'additionneur 8 bits spéculatif.

III Soustraction binaire - [5 pts]

On souhaite, à partir de l'additionneur composé de 8 fois un additionneur 1 bit en cascade, réaliser un circuit pour effectuer la soustraction. Pour cela, on remarque que la différence $x - y$ est identique à $x + (-y)$, c'est-à-dire, dans une représentation en complément à deux, $x + not(y) + 1$. En utilisant cette propriété, dessinez un circuit – le plus simple possible - permettant de réaliser la soustraction sur 8 bits.

III Assembleur MIPS – [5 pts]

Soit le code suivant :

```
            .data
result:     .asciiz "toto"

            .text
            .globl main

main:       addi $sp, $sp, -4
            sw   $ra, 0($sp)
            li   $a0, 5
            jal  gg
            la   $t0, result
            sw   $v0, 0($t0)
            lw   $ra, 0($sp)
            addi $sp, $sp, 4
            jr $ra

gg:         addi $sp, $sp, -4
            sw   $ra, 0($sp)
            bne  $a0, $zero, znz
            addi $v0, $zero, 1
            lw   $ra, 0($sp)
            addi $sp, $sp,4
            jr   $ra

znz:        addi $sp, $sp,-4
            sw   $a0, 0($sp)
            addi $a0, $a0, -1
            jal  gg
            lw   $t0, 0($sp)
            addi $sp, $sp, 4
            mul  $v0, $v0, $t0
            lw   $ra, 0($sp)
            addi $sp, $sp, 4
            jr   $ra
```

Répondez aux questions suivantes :

1. Expliquez ce que font les fonctions `main`, `gg` et `znz` (si nécessaire, commentez chaque ligne du programme).
2. À quoi sert le registre `$sp` ?
3. Écrivez une fonction équivalente au code proposé en Ada (ou tout autre langage de votre choix).
4. Quelle est la valeur stockée en `result` à l'issue de l'exécution de ce code ?