Université d'Aix-Marseille III

Licence Math-Info 1ère année - I2 - Examen de septembre 2006
Durée : 1h30 - Aucun document autorisé - Calculatrice interdite

Exercice 1 : Circuits logiques (7 points)

Nous allons construire des circuits déterminant des propriétés concernant une suite de 4 bits x_0, x_1, x_2 et x_3, fournis en entrée du circuit. Ces circuits n'ont qu'une sortie s qui est à 1 si la propriété à déterminer est vraie et 0 si elle est fausse. Etant donné que ces 4 bits d'entrée forme une suite, on pourra simplifier leur expression en ne spécifiant que leur valeur dans l'ordre de la suite. Ainsi, si on dit qu'on fournit 0010 en entrée du circuit, cela signifie que $x_0 = 0$, $x_1 = 0$, $x_2 = 1$ et $x_3 = 0$.

1.1 Dans cette question, nous allons réaliser le circuit qui indique si la suite donnée en entrée contient un nombre de 1 supérieur ou égal à son sombre de 0. Cette propriété est donc vrai par exemple pour les entrées 1101, 1100, 1001 mais fausse par exemple pour les entrées 0000 et 0100.

Ecrire la table de vérité de la fonction logique qui indique la valeur de sortie s en fonction des valeurs des entrées x_0, x_1, x_2 et x_3.

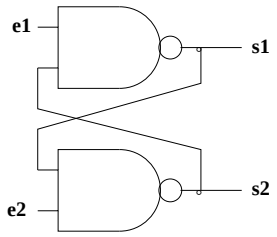
En utilisant la méthode des tables de Karnaugh, écrire la fonction booléenne correspondant à la valeur de sortie s.

Dessiner le circuit permettant d'obtenir s à partir de x_0, x_1, x_2 et x_3. Pour ceci, on n'a le droit d'utiliser que la porte logique unaire NON et les portes logiques OU et ET.

1.2 Dans cette question, nous allons réaliser le circuit qui indique si la suite donnée en entrée contient au moins une fois deux 1 consécutifs (la sous-suite 11). Cette propriété est donc vrai par exemple pour les entrées 1101, 0110, 1111 mais fausse par exemple pour les entrées 0100, 0101 et 1001.

Définir ce circuit en répondant aux mêmes questions que précédemment : écrire la table de vérité, la table de Karnaugh et en déduire la fonction booléenne correspondante. (ne pas dessiner le circuit cette fois-ci)

1.3 Cette question est indépendante des questions 1.1 et 1.2.
Considérons le circuit ci-dessous.



Ecrire la table de vérité des sorties s_1 et s_2 en fonction des valeurs d'entrées e_1 et e_2. Expliquer pourquoi certaines valeurs de sorties sont impossibles à déterminer uniquement en fonction des entrées e_1 et e_2.

Pour quels changements simultanés des deux valeurs d'entrées e_1 et e_2 peut-on avoir une indécision quant aux valeurs de sorties s_1 et s_2 ? Expliquer pourquoi.

Exercice 2 : Représentation des nombres et codage en mémoire (3 points)

2.1 Donnez la représentation décimale et la représentation hexadécimale du nombre 1010111 (représenté en notation binaire).

2.2 Donnez la valeur en représentation décimale de l'entier relatif codé sur un octet dont la représentation binaire est 10101010.

2.3 Combien de bits faut-il pour coder la date, sachant que le jour varie entre 1 et 31, le mois entre 1 et 12 et l'année entre 00 et 99? Combien d'octets?

Exercice 3 : Langage machine (4 points)

Dans cet exercice, on se place dans le cadre du microprocesseur MP0 et du langage machine LM0 décrit en cours. On rappelle que MP0 est un microprocesseur possédant deux registres d'adresses 8 bits A0 et A1 et deux registres de données 8 bits D0 et D1. LM0 possède notamment les instructions de transfert d'octet MOVE, d'addition ADD, de soustraction SUB. Ces instructions ont deux opérandes. Le premier est l'opérande source et le deuxième est l'opérande destination, qui sera modifié grâce à l'opérande source. LM0 a 3 types d'adressage : immédiat (on préfixe l'opérande avec #), direct (l'opérande est laissé tel quel) et indirect (l'opérande, qui est toujours un registre d'adresse, est mis entre parenthèses).

3.1 *Pour chaque instruction suivante, indiquez le registre de donnée, le registre d'adresse ou la case mémoire qui sera modifiée ainsi que sa nouvelle valeur. Si l'instruction est incorrecte, indiquez-le et expliquez pourquoi.* On rappelle qu'une instruction est toujours mémorisée sur deux octets. Chaque instruction doit être considérée indépendamment des autres et les éventuelles modifications de la mémoire ou des registres des instructions précédentes ne seront pas prises en compte. Avant chaque exécution d'instruction, le registre D0 contient la valeur 5, le registre A0 contient la valeur 100 et la case mémoire d'adresse 100 contient la valeur 13.

```
a) MOVE #100, D0
b) MOVE A0, #100
c) ADD 100, D0
d) MOVE D0, 100
e) MOVE 100, 101
f) MOVE (A0), 101
g) MOVE (A0), (A0)
h) ADD (A0), (A0)
```

3.2 On considère que l'adresse 100 contient la valeur 6. On exécute la série d'instructions suivante stockée de l'adresse 200 à l'adresse 227:

```
200 : MOVE 100, D0
202 : SUB #1, D0
204 : MOVE 100, D1
206 : DIV D0, D1
208 : MUL D0, D1
210 : CMP 100, D1
212 : JEQ #222
214 : SUB #1, D0
216 : CMP #1, D0
218 : JEQ #226
```

```
220 : JMP #204
222 : MOVE #0, 101
224 : JMP #228
226 : MOVE #1, 101
228 : ...
```

Une fois arrivé à l'adresse 228, quelle valeur sera à l'adresse 101 ? Même question s'il y avait eu la valeur 7 (et non pas 6) à l'adresse 100 ? D'une façon générale, indiquez la valeur qu'on a à la fin à l'adresse 101 en fonction du type de valeur qu'on a à l'adresse 100. On rappelle que l'instruction JEQ est le saut conditionnel qui effectue un saut à l'adresse indiquée si les deux opérandes de l'instruction CMP la précédant ont la même valeur. On rappelle aussi que MUL est l'instruction de multiplication et que DIV est l'instruction de division *entière*.

Exercice 4 : Utilisation d'Unix (6 points)

4.1 On se trouve dans un répertoire ne contenant que les fichiers `deserteur.txt` et `vian.txt`. Le fichier `deserteur.txt` contient les 2 lignes de texte suivantes :

```
Monsieur le President
Je vous fais une lettre
```

et le fichier `vian.txt` contient les 2 lignes de texte suivantes :

```
Que vous lirez peut-etre
Si vous avez le temps.
```

Indiquez les effets du lancement des commandes Unix suivantes à partir d'un shell. Signalez les erreurs éventuelles. Chaque question est indépendante : on fera comme si la commande précédente n'avait pas été exécutée.

a) `$ cat deserteur.txt`
b) `$ cat < vian.txt`
c) `$ cat boris.txt`
d) `$ cat deserteur.txt vian.txt`
e) `$ echo 'Je viens de recevoir' > deserteur.txt`
f) `$ echo 'Mes papiers militaires' vian.txt >> deserteur.txt`
g) `$ cat vian.txt deserteur.txt | grep s`
h) `$ echo 'Pour partir a la guerre' | cat`
i) `$ cat | cat > deserteur.txt`
j) `$ ls *.txt > deserteur.txt`
k) `$ cat deserteur.txt >> vian.txt`
l) `$ ls boris.txt > boris.txt`

4.2 *Expliquer ce qui se passe lorsqu'on exécute les commandes suivantes* :

1) `$ cat > machin`
2) `$ cat`
3) `$ head -2`
4) `$ head -2 > truc`
5) `$ cat chose >> chose`
6) `$ tail -5 | grep e`