DIRO
IFT 1215

EXAMEN INTRA

Max Mignotte

DIRO, Département d'Informatique et de Recherche Opérationnelle, local 2377
Http : //www.iro.umontreal.ca/∼mignotte/ift1215/
E-mail : mignotte@iro.umontreal.ca

Date : *26/02/2009*

I	Conversion et Arithmétique en Base n (30 pts).
II	Représentation des Données Entières (20 pts).
III	Représentation Flottante (14 pts).
IV	Circuits Logiques (17 pts).
V	Circuits Séquentiels (30 pts).
Total	111 points.

Directives

- Toute documentation est permise
- Toute calculatrice est interdite

- Les réponses devront être clairement présentées et justifiées (elles peuvent être concises mais devront néanmoins contenir les résultats intermédiaires nécessaires permettant de montrer sans ambiguïté que vous êtes arrivés au résultat demandé).

- Si vous ne comprenez pas une question, faites en une interprétation, et proposez une réponse.

I. Conversion et Arithmétique en Base n (30 pts)

1. Convertir 333_{10} en binaire (base 2). Utiliser le résultat de sa conversion en binaire pour trouver son équivalent octal (base 8) et hexadécimal (base 16).
   <5 pts>
2. Convertir 0.33_{10} en base 2, base 8 et base 16 en utilisant comme notation une barre au dessus d'un éventuel motif de plusieurs bits qui se répéterait à l'infini ou 12 bits après la virgule si ce n'est pas le cas et au moins trois digits après la virgule en base 8 et 16.
   <5 pts>
3. En déduire la représentation de 333.33_{10} en base 2, base 8 et base 16.
   <5 pts>
4. Convertir le nombre $1.00\,\overline{001}_2$ (i.e., $0.00\,001\,001\,001\ldots$ avec le motif '001' se répétant à l'infinie) en une fraction décimale.
   <5 pts>
5. Quel serait l'algorithme qui permettrait, à partir d'une suite de caractères alphanumériques codé en Unicode de les convertir en une suite de caractères ASCII (quand cela est possible sinon on transforme ce caractère en caractère ASCII NUL) avec un code correcteur de parité paire ?
   <5 pts>
6. Représenter $0.1\overline{7}_8$ en une fraction décimale [1].
   <5 pts>
   NOTA : Si vous n'y arrivez pas, donner une approximation de cette valeur en base 10 avec deux chiffres significatifs après la virgule (pour la moitié des points de la question et en laissant une somme de fractions).

Réponse

1.

$333_{10} = 1\,0100\,1101_2 = 515_8 = 14D_{16}$. **<5pts>**

2.

$0.33_{10} = 0.0101\ 0100\ 0111_2 = 0.2507_8 = 0.547_{16}$. **<5 pts>**

3.

$333.33_{10} = 1\ 0100\ 1101\ .0101\ 0100\ 0111_2 = 515.2507_8 = 14D.547_{16}$ **<5 pts>**

4.

▷ 1−**ère méthode** : En utilisant les suites géométriques :

$1.00\,\overline{001}_2$ se traduit en base 10 par la valeur :

$$1 + (2^{-5})\big(1 + 2^{-3} + 2^{-6}\ldots\big) = 1 + (2^{-5})\big(\tfrac{1-(2^{-3})^{\infty}}{1-2^{-3}}\big) = 1 + \tfrac{1}{28} = \tfrac{29}{28}$$

▷ 2−**ième méthode** : En utilisant la méthode expliquée aussi en cours et utilisant les décalage de bits :

1. Soit $p = 1.00\,\overline{001}_2$. On a

$p =$	$1.$	$00\,\overline{001}_2$
$4\,p =$	$100.$	$\overline{001}_2$
$32\,p =$	$100001.$	$\overline{001}_2$

$$32\,p - 4\,p = \quad 28\,p = \quad 100001_2 - 100_2 = 33_{10} - 4_{10} = 29_{10}$$
$$p = \tfrac{29}{28}$$

<5 pts>

5.

Dans le cas où les 8 premiers bits de poids fort sont différents de 00000000 alors on stocke 00000000 (caractère NUL).

Dans le cas contraire, il suffit de supprimer les 8 premiers bits de poids fort (l'octet de poids fort), puis de compter le nombre de bits à "1" dans les 7 derniers bits de poids faibles (de l'octet de poids faible restant). Si ce nombre est paire, alors on change le 8 ième bit (celui de poids fort) par "0" sinon on le change à "1".

<5 pts>

6.

Pour avoir la valeur exacte sous la forme d'une fraction décimale de $q = 0.1\overline{7}_8$, on peux utiliser les deux méthodes mentionnées à l'exercice 4. La plus simple reste encore celle utilisant les décalage de digit.

$$\begin{array}{rrcll} & \mathrm{q} & = & 0. & 1\,\overline{7}_8 \\ 8_{10} \times & \mathrm{q} & = & 1. & \overline{7}_8 \\ 64_{10} \times & \mathrm{q} & = & 17. & \overline{7}_8 \\ \hline \end{array}$$

$$64\,x - 8\,x = \quad 56\,x = \quad 16_8 = 14_{10}$$
$$x = \tfrac{14}{56} = \tfrac{1}{4}$$

<5 pts>

Sinon, on a besoin de deux chiffres après la virgule en base 8 pour arriver à la précision demandée en base 10. On doit donc negliger dans la somme, les termes plus petits qu'un centième cad le terme $\frac{7}{8^4}$.

Dans ce cas, on a :

$$0.1\overline{7}_8 = \frac{1}{8} + \frac{7}{8^2} + \frac{7}{8^3} + \frac{7}{8^4} (\approx 0.2497558)$$

II. Représentation des Entiers Signés (20 pts)

1. Représenter puis calculer $-51 - 32$ sur 8 bits en complément à un (complément logique ou restreint) puis en complément à deux (complément arithmétique ou vrai) en indiquant à chaque fois lorsqu' apparaît un débordement (dépassement de capacité) ou une retenue ainsi que le résultat en base 10 obtenu.
   <10 pts>
2. Quel est le plus petit nombre entier A pour lequel l'opération $-A - 32$ sur 8 bits en complément à un puis en complément à deux provoquerait un dépassement (ou débordement) de capacité. Justifier votre réponse.
   <5 pts>
3. À quoi correspond l'expression (i.e., le calcul) suivante en base 10 ?

   $$1101\,1101 - 1110\,0001$$

   exprimées en complément à deux et faites cette opération (calcul sur 8 bits) en indiquant lorsqu'apparaît un débordement ou une retenue.
   <5 pts>

Réponse

1.

Sur 8 bits, $-51_{10} = -0011\,0011_2$ et $-32_{10} = -0010\,0000_2$

En complément à un (sur 8 bits) on a $-51_{10} = 1100\,1100_2$ et $-32_{10} = 1101\,1111_2$ et l'opération suivante

$$1100\,1100 + 1101\,1111_2 = 1\,1010\,1011 = 1010\,1011 + 1 = 1010\,1100 = -0101\,0011 = -83_{10}.$$

<3 pts>

On a donc eu une retenue (que l'on a ajouté au résultat) et on obtient en final un chiffre négatif (premier des 8 bits a "1") et finalement une valeur égale à -83_{10}.

<2 pt>

En complément à deux (sur 8 bits) on a $-51_{10} = 1100\,1101_2$ et $-32_{10} = 1110\,0000$ et l'opération suivante

$$1100\,1101_2 + 1110\,0000_2 = 1\,1010\,1101 = 1010\,1101 = -0101\,0011 = -83_{10}$$

<3 pts>

On a donc eu une retenue (que l'on a ignoré) et on obtient en final un chiffre négatif (premier des 8 bits a "1") et finalement une valeur égale à -83_{10}.

<2 pt>

2.

En complément à un, il n'existe pas de -128 (sur 8 bits), donc le plus petit entier A pour lequel l'opération $-A - 32$ sur 8 bits provoquerait un dépassement de capacité est $A = 128 - 32 = 96$ (i.e., l'opération $-96 - 32$ provoquerait un dépassement de capacité).

<2,5 pts>

En complément à deux, il n'existe pas de -129 sur 8 bits donc $A = 129 - 32 = 97$ (i.e., l'opération $-97 - 32$ provoquerait un dépassement de capacité).

<2,5 pts>

3.

On soustrait un nombre négatif $(-(\,0010\,0010_2 + 1_2 = -(0010\,0011_2) = -35_{10})$ avec un nombre négatif $(-(\,0001\,1110_2 + 1_2 = -(0001\,1111_2) = -31_{10})$; c'est à dire l'opération en base 10 suivante ▷

$$-35_{10} - (-31_{10}) = -35_{10} + 31_{10}$$

<2 pts>

et on a en base 2, et complément à deux (sur 8 bit), on calcul donc l'opération suivante,

$$1101\,1101_2 + 0001\,1111_2 = 1111\,1100_2 = -(0000\,0011_2 + 1_2) = -(0100_2) = -4_{10}.$$

<3 pts>

III. Représentation Flottante (14 pts)

1. Avec une représentation flottante binaire avec un format du type $\pm 0.xxxx\ldots$ (avec 23 x, i.e , 23 bits pour la mantisse), un bit de signe ("0" pour le signe "+") et un exposant sur 8 bits avec un biais de 127 permettant d'exprimer les valeurs de l'exposant de -126 à 127 (l'exposant égale à -127 et 128 permettant d'exprimer des valeurs spéciales) et biais de 127.
   Donner en binaire (en utilisant la disposition suivante [bit de signe | 23 bits mantisse | 8 bits de l'exposant]) la représentation de la plus petite valeur valeur supérieur à 1.0. Convertir cette valeur binaire en valeur décimale (la mettre sous la forme 1 + une fraction décimale).
   <7 pts>
2. Souvenez vous du problème de la défaillance du missile US patriote présenté en cours (Chapitre Système de nombre, slide 15). Expliquer quelle aurait été l'erreur maximale de l'horloge interne du missile US si l'informaticien en charge de ce projet avait décidé d'utiliser un registre flottant double précision (pour lequel on a 52 bits pour la mantisse).
   NOTA : pour obtenir cette erreure maximale, on considerera le pire des cas, c'est a dire la cas ou les bits qu'on a laissé tombé (après le 52ième bit étaient tous à un [on obtient ainsi notre borne supérieure de l'erreur]). De plus on laissera le résultat sous forme d'une multiplication.
   <7 pts>

Réponse

1.

La plus petite valeur flottante, plus grande que 1.0, a dans cette numérotation flottante la représentation suivante

$$0.\underbrace{10000000000000000000001}_{23 \text{ bits}} \times 2^1 \qquad \text{soit} \qquad [0|10000000000000000000001|10000000]$$

et une valeur décimale égale à $(2^{-1} + 2^{-23}) \times 2^1 = 1 + 2^{-22}$.

<7 pts>

2.

On rappelle que $0.1_2 = 0\,\overline{0011}_2$.

2 réponses possibles :

• si on considère une représentation flottante avec $\pm 0.1xxx\ldots$, on a $0.1_2 = 0\,\overline{1100}_2 \times 2^{-4}$ et dans le cas où on laisse tomber tous les bits de la mantisses après le 52ième, l'erreur est donc, (en considérant au pire que des 1 qu'on laisse tomber) ; $(2^{-53} + 2^{-54} + 2^{-55} + \ldots) \times 2^{-4} = 2^{-52} \times 2^{-4} = 2^{-56}$. Soit une erreur finale de 3600000×2^{-56}.

• si on considère une représentation flottante avec $\pm 0.xxxx\ldots$ sans "1" pour le premier x, on a une erreur finale de 3600000×2^{-52}.

<7 pts>

Nota : Dans les deux cas, on obtient une errreur négligeable de repectivement 8×10^{-10} (seconde).

IV. Circuits Logiques (17 pts)

On désire construire un circuit logique qui a pour entré, trois entrées A_1, A_2 et A_3 (respectivement l'entrée numéro UN, l'entrée numéro DEUX et TROIS) et deux sorties F_0 et F_1.
On veut que ces deux sorties codent **en code gray** le numéro de la plus grande entrée mise à un (i.e., un encodeur prioritaire en code gray avec la règle de priorité "A_{i+1} À UNE PLUS HAUTE PRIORITÉ QUE A_i"). F_0 correspondra au LSB (Less significant bit) de ce code. Lorsqu'aucune de ces entrées ne sera active (i.e., mise à un), $F_1\,F_0$ doit nous renvoyer le code binaire 0 0, c'est à dire 0 en code gray.

Rappel : on rappel que $00_{\text{gray}} = 0_{10}$, $01_{\text{gray}} = 1_{10}$, $11_{\text{gray}} = 2_{10}$ et $10_{\text{gray}} = 3_{10}$.
Exemple : si $A_1 = 0$ $A_2 = 1$ $A_3 = 1$, F_1 F_0 doit nous renvoyer le code binaire 1 0.

1. Construire la table de vérité de F_0 et F_1 en fonction des trois variables Booléennes A_1, A_2, et A_3. **<5 pts>**
2. Donner les équations logiques (simplifiées au maximum) des signaux de sortie de ce circuit. **<6 pts>**.
3. Implémenter les deux expressions non simplifiées en utilisant pour chacune d'entre elle un $\text{MUX}_{4\times1}$ et les variables A_3 et A_2 comme variables de contrôle des $\text{MUX}_{4\times1}$. **<6 pts>**

   Nota : Si vous n'y arrivez pas, alors implémenter l'expression de F_0 et F_1 avec un $\text{MUX}_{8\times1}$ pour la moitié des points de cette question.

Réponse

1.

A_3	A_2	A_1	F_1	F_0
0	0	0	0	0
0	0	1	0	1
0	1	0	1	1
0	1	1	1	1
1	0	0	1	0
1	0	1	1	0
1	1	0	1	0
1	1	1	1	0

<5 pts>

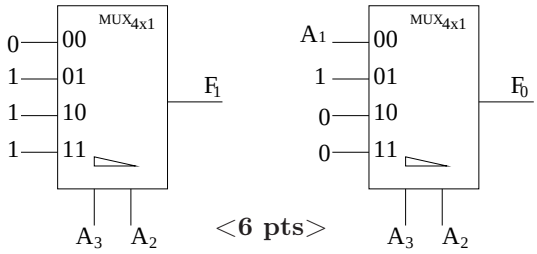
2.

$$F_0 = A_2\overline{A}_3 + A_1\overline{A}_3$$
$$F_1 = A_2 + A_3$$

<6 pts>

Nota : Eqution non simplifiées : 4/6 pts

3.



<6 pts>

Nota : RJCF : 4/6 pts

V. Circuits Séquentiels (30 pts)

On va considérer dans cet exercice, la conception d'un système (simplifié) de feux de circulation à un carrefour où deux voies (nord-sud et est-ouest) se croisent avec deux passages piétons chacun possédant

un bouton d'appel permettant d'arrêter la circulation dans les deux sens (et donc permettant au piéton de traverser).

Dans ce système séquentiel, les sorties seront donc les deux feux que l'on notera NS et EO (pour feux nord-sud et feux est-ouest) qui prendront donc les valeurs binaires 1 (pour rouge) et 0 pour vert (Nota : il n'y a donc pas de feux oranges ou de feux clignotants dans ce système). L'entrée unique sera donc le bouton d'appel que l'on notera B qui prendra la valeur 1 qui enregistrera un "1" lorsqu'on appuyera dessus, (enregistrant ainsi le désir du piéton de traverser et "0" dans le cas contraire). La période d'horloge qui synchronisera le système sera d'une minute.

Dans ce système, si personne n'appuie sur le bouton d'appel, il y a séquentiellement (i.e., après chaque période d'horloge), le feux vert dans la direction nord-sud et le feux rouge dans la direction est-ouest puis après une période d'horloge (i.e., une minute), l'inverse, c'est à dire le feux vert dans la direction est-ouest et le feux rouge dans la direction nord-sud.

Si un piéton appuie sur le bouton B, le système passe, durant une période d'horloge (i.e., une minute) dans un état ou les deux feux (est-ouest et nord-sud) sont rouges dans les deux sens. Après cette période d'horloge, le système rétablira la circulation dans le sens contraire à celle qui précédait l'appuie du bonton B **[que l'on appuie ou non une nouvelle fois sur** B**]** ; Le système mettra le feux vert dans la direction nord-sud et le feux rouge dans la direction est-ouest si avant d'appuyer sur le bouton B on avait le feux rouge dans la direction nord-sud et le feux vert dans la direction est-ouest et inversement.
On ne peut donc pas en appuyant continuellement sur B mettre les feux rouges continuellement.

1. Dessiner le diagramme de transition d'états de ce système.
   <10 pts>
   Nota : On utilisera sur les flèches de transition d'états, la convention suivante Entrée/Sortie1 Sortie2, i.e., $B/NS\,EO$ du style 0/01 indiquant que l'on a pas appuyer sur B (0 en entré) et qu'en sortie le feux nord-sud est vert (0 pour la 1ère sortie) et que le feux est-ouest est rouge (1 pour la 2ème sortie).
2. Donner ensuite la table d'état de ce système.
   <3 pts>
3. Établir la table de vérité de ce système.
   <5 pts>
4. Donner les équations logiques (**simplifiées au maximum**) des signaux de sortie et des états futurs de ce système.
   <6 pts>.
5. Donner le logigramme de ce système en utilisant des bascules D.
   <6 pts>

Réponse

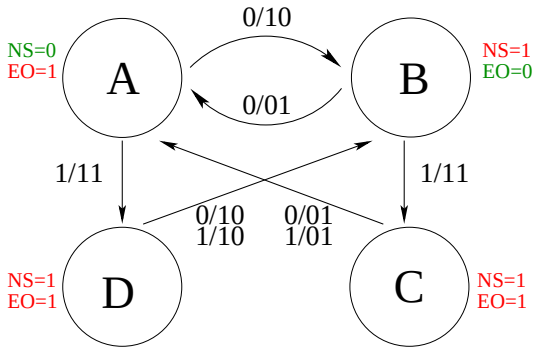
1.

Graphe de Transition d'États :
Dans ce diagramme d'états

- A est l'état ou le feux nord-sud est vert ($NS = 0$) et le feux est-ouest est rouge ($EO = 1$).
- B est l'état ou le feux nord-sud est rouge ($NS = 1$) et le feux est-ouest est vert ($EO = 0$).
- C est l'état où les deux feux sont rouges ($NS = EO = 1$) après B.
- D est l'état où les deux feux sont rouges ($NS = EO = 1$) après A.

Nota : -2 pour chaque flêche erronée, -1 pour chaque annotation erronée.

<10 pts>



2.

Table d'États :

Etat present \ Input	B = 0	B = 1
A : 00	B:01 / 10	D:11/ 11
B : 01	A:00/ 01	C:10/ 11
C : 10	A:00/ 01	A:00/ 01
D : 11	B:01/ 10	B:01/ 10

<3 pts>

RJCF : 2/3 pts

Table de Vérité :

$q_1(t)$	$q_0(t)$	B	$q_1(t+1)$	$q_0(t+1)$	NS	EO
0	0	0	0	1	1	0
0	0	1	1	1	1	1
0	1	0	0	0	0	1
0	1	1	1	0	1	1
1	0	0	0	0	0	1
1	0	1	0	0	0	1
1	1	0	0	1	1	0
1	1	1	0	1	1	0

<5 pts>

RJCF : 4/5 pts

4.

Équations Logiques :

On trouve, à partir de la table de vérité, et se servant (ou non) des tableaux de Karnaugh, les équations logiques suivantes, simplifiées au maximum,

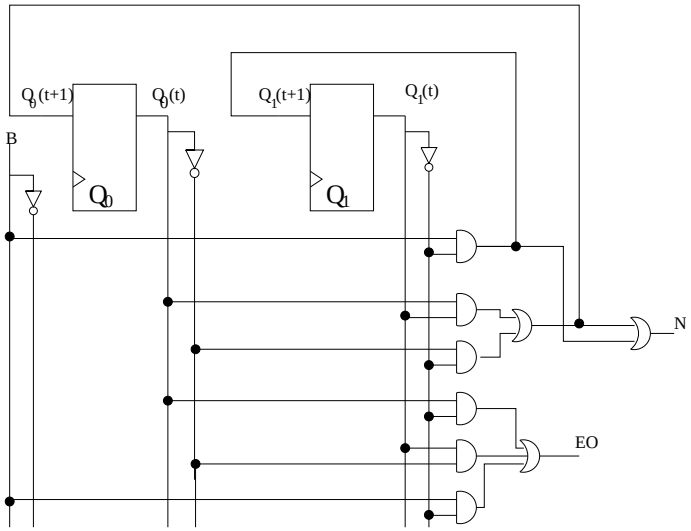
$$
\begin{aligned}
Q_1(t+1) &= B\,\overline{Q}_1(t) \\
Q_0(t+1) &= Q_0(t)Q_1(t) + \overline{Q}_0(t)\overline{Q}_1(t) \\
NS &= Q_0(t)\,Q_1(t) + \overline{Q}_0(t)\,\overline{Q}_1(t) + B\,Q_0(t) \quad \text{ou} \quad NS = Q_0(t)Q_1(t) + \overline{Q}_0(t)\overline{Q}_1(t) + B\,\overline{Q}_1(t) \\
E0 &= \overline{Q}_0(t)Q_1(t) + Q_0(t)\overline{Q}_1(t) + B\,\overline{Q}_1(t) \quad \text{ou} \quad E0 = \overline{Q}_0(t)Q_1(t) + Q_0(t)\overline{Q}_1(t) + B\,\overline{Q}_0(t)
\end{aligned}
$$

<6 pts>

RJCF : 4/5 pts

5.

On en déduit le schéma de notre circuit séquentiel :



<6 pts>

RJCF : 4/6 pts