**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 :** *02/03/2007*

| | |
|---|---|
| I ..................... | Conversion et Arithmétique en Base n (30 pts). |
| II .................... | Représentation des Données Entières (15 pts). |
| III ................... | Représentation Flottante (10 pts). |
| III ................... | Circuits Logiques (21 pts). |
| IV ................... | Circuits Séquentiels (30 pts). |
| Total ................ | 106 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 $555_{10}$ 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 $0.48_{10}$ 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 ou 10 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 $555.48_{10}$ en base 2, base 8 et base 16.
   <**5 pts**>
4. Convertir le nombre $0.1\overline{0100}_2$ (i.e., $0.1\,0100\,0100\,0100\ldots$ avec le motif 0100 se répétant à l'infinie) en une fraction décimale.
   <**5 pts**>
5. Quel serait l'algorithme (donner par exemple le pseudo code) qui permettrait, à partir d'une suite de caractères ASCII représentant du texte, de vérifier que la première lettre du premier mot de chaque phrase (donnée dans ce texte) est en majuscule et si ce n'est pas le cas de transformer cette première lettre en majuscule (lorsque cela est nécessaire).
   $N$ota : On considérera le cas simple dans lequel il n'y a pas de nombre, ni de “ !” , “ ?” , “ :” et “ ;” dans le texte. On rappelle que le caractère . (i.e., “point”) est représenté en codage ASCII par la représentation hexadécimal $2E_{16}$.
   <**5 pts**>
6. Représenter $0.1\overline{6}_8$ en une fraction décimale [1].
   <**5 pts**>
   Nota : Si vous n'y arrivez pas, donner une approximation de cette valeur en base 10 avec quatre chiffres significatifs après la virgule (pour la moitié des points de la question).

---

**Réponse**

$\boxed{\textbf{1.}}$

$555_{10} = 10\,0010\,1011_2 = 1053_8 = 22B_{16}$.  <**5pts**>

$\boxed{\textbf{2.}}$

$0.48_{10} = 0.0111\ 1010\ 1110_2 = 0.3656_8 = 0.7AE_{16}$ ($0.\overline{7AE14}_{16}$).  <**5 pts**>

$\boxed{\textbf{3.}}$

$555.48_{10} = 10\,0010\,1011.0111\,1010\ 1110_2 = 1053.3656_8 = 22B.7AE$ <**5 pts**>

$\boxed{\textbf{4.}}$

$\triangleright$ 1−**ère méthode** : En utilisant les suites géométriques :

$0.1\,\overline{0100}_2$ se traduit en base 10 par la valeur :

$$2^{-1} + (2^{-3})\big(1 + 2^{-4} + 2^{-8}\ldots\big) = 2^{-1} + (2^{-3})\,\frac{1-(2^{-4})^{\infty}}{1-2^{-4}} = \frac{1}{2} + \frac{1}{8}\,\frac{16}{15} = \frac{19}{30}$$

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

1. Soit $p = 0.1\,\overline{010}_2$. On a

$$\begin{array}{rrr} p = & 0. & 1\,\overline{0100}_2 \\ 2\,p = & 1. & \overline{0100}_2 \\ 32\,p = & 10100. & \overline{0100}_2 \\ \hline \end{array}$$

$$32\,p - 2\,p = \quad 30\,p = \quad 10100_2 - 1_2 = 19_{10}$$

$$p = \tfrac{19}{30}$$

**<5 pts>**

$\boxed{\textbf{5.}}$

Pour le premier caractère du texte et pour tout ceux qui suivent le caractère hexadécimal “2E”, on vérifie que le caractère hexadécimal est compris entre $41_{16}$ et $5A_{16}$ sinon on retranche 20 au caractère pour le passer en majuscule. **<5 pts>**

$\boxed{\textbf{6.}}$

Pour avoir la valeur exacte sous la forme d’une fraction décimale de $0.1\overline{6}_8$, on peux utiliser les deux méthodes mentionnées à l’exercice 4. La plus simple reste encore celle utilisant les décalage de digit.

$$\begin{array}{rccrr} & x & = & 0. & 1\,\overline{6}_8 \\ 8_{10} & x & = & 1. & \overline{6}_8 \\ 64_{10} & x & = & 16. & \overline{6}_8 \\ \hline \end{array}$$

$$64\,x - 8\,x = \quad 56\,x = \quad 16_8 - 1_8 = 15_8 = 13_{10}$$

$$x = \tfrac{13}{56} \approx 0.2321166993.$$

**<5 pts>**

Sinon, on a besoin de quatre chiffres après la virgule en base 8 pour arriver à la précision demandée en base 10. Dans ce cas, on a :

$$0.1\overline{6}_8 = \frac{1}{8} + \frac{6}{8^2} + \frac{6}{8^3} + \frac{6}{8^4} + \frac{6}{8^5} + \frac{6}{8^6} \approx 0.2321_{395875}.$$

---

## II. Représentation des Entiers Signés (15 pts)

1. Représenter puis calculer $76 - 108$ 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.
   **<8 pts>**
2. Quel est le nombre maximal de bits qui aurait conduit au débordement du calcul précédent (en complément à deux). Justifiez bien votre réponse.
   **<2 pts>**
3. À quoi correspond l’expression (i.e., le calcul) suivante en base 10 ?
   **<2 pts>**

   $$0111\,0110 - 1001\,0010$$

   exprimées en complément à deux et faites cette opération (calcul sur 8 bits en complément à deux) en indiquant lors-qu’apparaît un débordement ou une retenue.
   **<3 pts>**

**Réponse**

$\boxed{\textbf{1.}}$

Sur 8 bits, $76_{10} = 0100\,1100_2$ et $-108_{10} = -0110\,1100_2$

En complément à un (sur 8 bits) on a $76_{10} = 0100\,1100_2$ et $-108_{10} = 1001\,0011_2$ et l'opération suivante,

$$0100\,1100 + 1001\,0011_2 = 1101\,1111 = -0010\,0000_2. \quad < \textbf{3 pts} >$$

On obtient donc un chiffre négatif (premier des 8 bits a "1") et finalement une valeur égale à $-32_{10}$.
<**1 pt**>

En complément à deux (sur 8 bits) on a $76_{10} = 0100\,1100_2$ et $-108_{10} = -0110\,1010_2 = 1001\,0100$ et l'opération suivante,

$$0100\,1100_2 + 1001\,0100_2 = 1110\,0000 = -(0001\,1111 + 1_2) = -(0010\,0000_2) \quad < \textbf{3 pts} >$$

On obtient donc un chiffre négatif (premier des 8 bits a "1") et finalement une valeur égale à $-32_{10}$.
<**1 pt**>

$\boxed{\textbf{2.}}$

Il faut pouvoir représenter en complément à deux sans problème $-108$ ce qui aurait posé problème pour 7 bits (nombre maximal de bits posant donc problème). <**2 pts**>

$\boxed{\textbf{3.}}$

Le premier chiffre est positif sans ambiguïté et est égale à $118_{10}$ le deuxième est négatif, car il commence par 1. En complément à deux, sa magnitude est $0110\,1101_2 + 1_2 = 0110\,1110_2 = 110_{10}$. Donc on a l'opération $118_{10} + 110_{10} = 228_{10}$. <**2 pts**>

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

$$0111\,0110_2 + 0110\,1110_2 = 1110\,0100_2$$

Il y a débordement de capacité car le premier et deuxième chiffre sont positifs et le résultats est négatif.
<**3 pts**>

---

## III. Représentation Flottante (10 pts)

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 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 et une notation flottante utilisant les valeurs arrondies et la disposition suivante [bit de signe | 23 bits mantisse | 8 bits de l'exposant] ;

1. En utilisant la convention qui permet d'assurer l'unicité de la représentation, donner la représentation binaire flottante de 3.75. As t-on une représentation exacte ?
   <**4 pts**>
2. En utilisant la convention qui permet d'assurer l'unicité de la représentation, donner la représentation de 3.7500001. As t-on une représentation exacte ? Dans le cas contraire donner en base 10, l'erreur d'affectation commise.
   $N$ota : $2^{-21} \approx 0.000000476$
   <**6 pts**>

**Réponse**

$\boxed{\textbf{1.}}$

On a $3.75_{10} = .1111_2 \times 2^2$ ou avec pour l'exposant biaisé $127 + 2 = 129$ <**1 pt**> exprimé en binaire
[0|.11110000000000000000000|10000001]
qui s'exprime parfaitement sans erreur d'affectation.
<**3 pts**>

$\boxed{\textbf{2.}}$

La valeur que l'on cherche n'est pas trop loin de la valeur précédente ; essayons de voir si on met le dernier bit (LSB) disponible de la mantisse a "un" (i.e., une mantisse à 11110000000000000000001) quel valeur va t-on obtenir. On aura un incrément de $2^{-23} \times 2^2 = 2^{-21} \approx 0.000000476$ ce qui nous fera en tout 3.750000476. Parmi [3.75et3.750000476] l'arrondissement préférera la valeur 3.75 et l'ordinateur fera une erreur de $10^{-7}$. On aura donc pas de représentation exacte.

<**6 pts**>

---

## IV. Circuits Logiques (21 pts)

On désire construire un circuit logique qui a pour entré, quatre variables binaires $A$, $B$, $C$, $D$ et une sortie $S$. On veut que $S = 1$ si le code binaire donné en entrée par AB (avec $A$ le MSB et $B$, le LSB) soit supérieur strictement au code binaire donné en entrée par CD (avec $C$ le MSB et $D$, le LSB).
Exemple -1- : Si $AB = 01$ et $CD = 11$, alors $S = 0$ car $1 \not> 3$.
Exemple -2- Si $AB = 01$ et $CD = 01$, alors $S = 0$ car $1 \not> 1$.
Exemple -3- Si $AB = 11$ et $CD = 01$, alors $S = 1$ car $3 > 1$.

1. Construire la table de vérité de $S$ en fonction des quatre variables Booléennes $A$, $B$, $C$, $D$.
   <**7 pts**>
2. En utilisant la table de Karnaugh simplifier le plus possible la fonction $S$ sous la forme d'une **SOP**.
   <**7 pts**>
3. Implémenter l'expression brute (i.e., non simplifiée) de cette fonction $S$ avec un $\text{MUX}_{4\times1}$ en utilisant quelques portes logiques (des NOR et des NON) et les variables $A$ et $B$ comme variables de contrôle du $\text{MUX}_{4\times1}$ (avec $A$ le MSB et $B$, le LSB).
   <**7 pts**>

---

**Réponse**

$\boxed{\textbf{1.}}$

$\boxed{\textbf{2.}}$

| A | B | C | D | S |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 |

<**7 pts**>

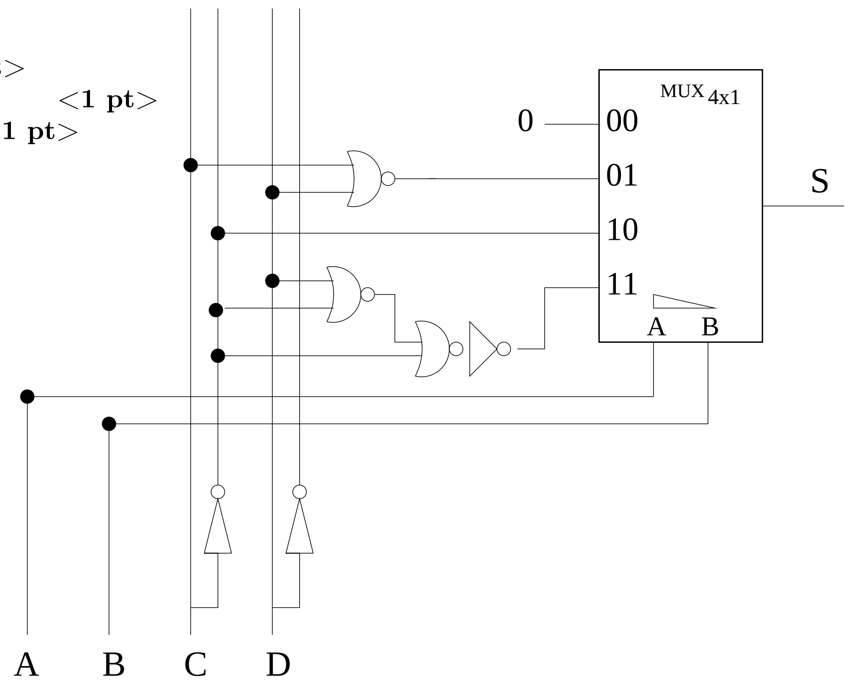
| CD \ AB | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | | 1 | 1 | 1 |
| 01 | | | 1 | 1 |
| 11 | | | | |
| 10 | | | 1 | |

$$S \quad = \quad A\,\overline{C} + B\,\overline{C}\,\overline{D} + A\;B\;\overline{D}$$

<**7 pts**>

$\boxed{\textbf{3.}}$

AB=00 ▷ $0$ <**1 pt**>
AB=01 ▷ $\overline{CD}$ <**1 pt**>
AB=10 ▷ $\overline{CD} + \overline{C}D = \overline{C}$ <**1 pt**>
AB=11 ▷ $\overline{C} + C\overline{D}$ <**1 pt**>




<**3 pts**>

---

## V. Circuits Séquentiels (30 pts)

On veut réaliser le dispositif logique séquentiel qui permette de détecter trois "1" reçus en entré, qu'importe le nombre de "0" entremêlé entre eux. De telle façon que pour une entrée du type (donnée ci dessous), on génère la sortie (donnée ci dessous) suivante

| | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Entré | 0 | 1 | 0 | 1 | $\underline{1}$ | 0 | 1 | 0 | 0 | 1 | 0 | $\underline{1}$ | 1 | 1 | 0 | $\underline{1}$ | 1 | 0 | 0 | 0 | 1 | 0 | $\underline{1}$ |
| Sortie | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

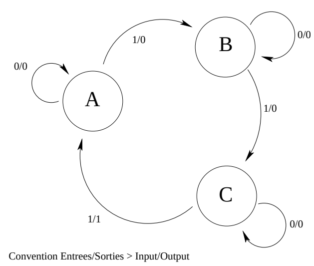
1. Dessiner le diagramme de transition d'états de ce système[1].
   **<10 pts>**
2. Donner ensuite la table d'état de ce système.
   **<3 pts>**
3. Établir la table de vérité de ce système.
   **<5 pts>**
4. Donner les équations logiques (**simplifiées au maximum**) des signaux de sortie et des états futurs de ce système.
   **<6 pts>**.
5. Donner le logigramme de ce système en utilisant des bascules D.
   **<6 pts>**

---

**Réponse**

$\boxed{\textbf{1.}}$

**Graphe de Transition d'États :**




**<10 pts>**

État A : on a recu zéro "1". État B : on a recu un "1". État C : on a recu deux "1".

$\boxed{\textbf{2.}}$

**Table d'États :**

| Etat present \ Input | 0 | 1 |
|---|---|---|
| A : 00 | A: 00 / 0 | B : 01/ 0 |
| B : 01 | B : 01/ 0 | C : 10/ 0 |
| C : 10 | C : 10/ 0 | A : 00/ 1 |

**<3 pts>**

$\boxed{\textbf{3.}}$

**Table de Vérité :**

1. On utilisera sur les flèches de transition d'états, la convention suivante Entrée/Sortie, i.e., *Input/Output* du style 1/0 indiquant que l'on a un 1 en entré et un 0 en sortie.

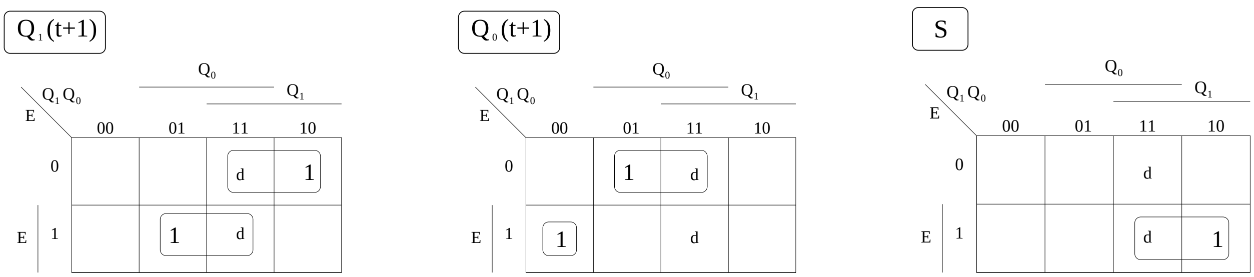
| $q_1(t)$ | $q_0(t)$ | E | $q_1(t+1)$ | $q_0(t+1)$ | S |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | d | d | d |
| 1 | 1 | 1 | d | d | d |

**<5 pts>**

$\boxed{\textbf{4.}}$

**Équations Logiques :**

On trouve, à partir de la table de vérité, et se servant des tableaux de Karnaugh et en considérant les états indifférents pour les combinaisons des bascules $q_1(t)q_0(t) = 11$, les équations logiques suivantes, simplifiées au maximum,

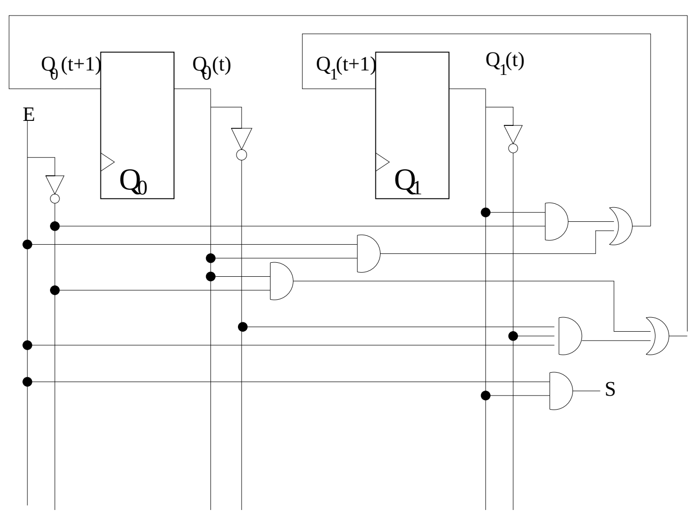



$$
\begin{aligned}
S &= E\,Q_1(t)\\
Q_0(t+1) &= Q_0(t)\,\overline{E} + \overline{Q}_1(t)\,\overline{Q}_0(t)\,E\\
Q_1(t+1) &= Q_1(t)\,\overline{E} + Q_0(t)\,E
\end{aligned}
$$

**<6 pts>**

$\boxed{\textbf{5.}}$

On en déduit le schéma de notre circuit séquentiel :




**<6 pts>**