ENSICAEN – 1ère année Electronique et Informatique
Année 2018-2019 – Semestre 1

Circuits et architectures logiques
Electronique numérique

Vendredi 18 janvier 2019 – 8h30-10h – durée 1h30 – Matthieu Denoual
Aucun document autorisé

Nom : **N° de place :**

Prénom : CORRIGÉ

Consignes
Les documents, calculatrices et téléphones portables ne sont pas autorisés.
Les réponses seront données sur ces feuilles à l'intérieur des espaces prévus à cet usage.

Partie 1 : Numération et codage [/7]

Exercice 1.1 [/4] Complétez le tableau ci-dessous

Base 2 (12 bits)*	Base 10 **	Erreur de représentation
0000 1100.0110	12,35	$\|10,375 - 0,35\| = 0,025 < \frac{1}{2} = 0,03125$
0000 0101 . 1010 1111 1010 . 0101 +1 1111 1010. 0110	-5,60	$\|10,6 - 9,625\| = 0,025 < \frac{1}{2}$
0000 0001 0101	1,31	0,0025
1111 0111 1010	$-16+4+2+1+0,5+0,125$ $-8,37$	0,005

* les nombres binaires seront représentés en complément à deux sur 12 bits et les valeurs non entières en virgule fixe $Q_{8,4}$. (Rappel représentation $Q_{m,k}$ sur N bits: $b_{m+k-1}b_{m+k-2}.....b_k,b_{k-1}.....b_2b_1b_0$; N=m+k)

** les nombres non entiers en base 10 seront représentés avec 2 chiffres significatifs derrière la virgule.

Exercice 1.2 [/3]

Codez les valeurs suivantes sur 32 bits virgule flottante selon la norme IEEE 754 rappelée en bas de page.

A = -0,75 $= -1,5 \cdot 2^{-1}$ $S=1$ $E-127=-1 \Rightarrow E=126$

0,5 × 2 = 1,0

S e_7 e_0 b_{22} b_{21} b_0
1 0111 1110 1 0000 0000 0000 0000 0000 000

B = 192 $= 1,5 \cdot 2^7$ $S=0$ $E-127=7 \Rightarrow E=134$

S e_7 e_0 b_{22}
0 1000 0110 1000 0000 0000 0000 0000 000

Rappel : représentation selon la norme IEEE 754. La valeur X est représentée suivant la forme : $X = (-1)^S \cdot 2^{E-127} \cdot 1,F$

X s'écrit alors en binaire virgule flottante : $\underbrace{S}_{\text{signe}} \underbrace{e_7e_6\ldots e_1e_0}_{E} \underbrace{f_{22}f_{21}\ldots f_2f_1f_0}_{F}$; E et F sont codés en binaire non signé.

Partie 2 : Convertisseur code binaire → code Gray [/7]

Exercice 2 [/7]

L'objectif de cet exercice est la synthèse d'un transcodeur du code binaire classique vers le code Gray. Le code Gray est rappelé dans la table ci-dessous.

	Code binaire $b_3\ b_2\ b_1\ b_0$	Code Gray $g_3\ g_2\ g_1\ g_0$
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 0	0 0 1 1
3	0 0 1 1	0 0 1 0
4	0 1 0 0	0 1 1 0
5	0 1 0 1	0 1 1 1
6	0 1 1 0	0 1 0 1
7	0 1 1 1	0 1 0 0
8	1 0 0 0	1 1 0 0
9	1 0 0 1	1 1 0 1
10	1 0 1 0	1 1 1 1
11	1 0 1 1	1 1 1 0
12	1 1 0 0	1 0 1 0
13	1 1 0 1	1 0 1 1
14	1 1 1 0	1 0 0 1
15	1 1 1 1	1 0 0 0

Question 2.1 [/0.5]

Le code Gray est-il pondéré ? non

Question 2.2 [/0.5]

Le code Gray est-il cyclique ? oui

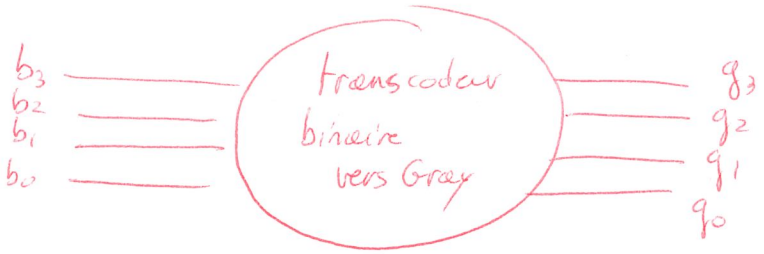
Question 2.3 [/0.5]

Le code Gray est-il continu ? oui

Question 2.4 [/5.5]

L'objectif est de synthétiser un transcodeur code binaire vers code Gray. La synthèse conduira aux expressions des équations logiques pour une implémentation avec uniquement des portes NAND. Il n'est pas nécessaire de dessiner des schémas avec des portes logiques.

Schématiser le transcodeur en identifiant ses entrées et ses sorties :



Utilisez les tables de Karnaugh ci-après pour réduire les expressions logiques donnant les sorties en fonction des entrées. Ecrivez sous chaque table l'expression logique réduite ainsi que l'expression mise en forme pour une implémentation uniquement avec des portes NAND.

g_3

b_3b_2 \ b_1b_0	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	1	1	1	1

$g_3 = b_3$

$\overline{g_3} = \overline{b_3}$

$b_3 - \boxed{\quad} - \boxed{\quad} - g_3$

g_2

b_3b_2 \ b_1b_0	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	0	0	0	0
10	1	1	1	1

$g_2 = \overline{b_3}\, b_2 + b_3 \overline{b_2}$

$\overline{g_2} = \overline{\overline{b_3} \cdot b_2} \cdot \overline{b_3 \overline{b_2}}$

g_1

b_3b_2 \ b_1b_0	00	01	11	10
00	0	0	1	1
01	1	1	0	0
11	1	1	0	0
10	0	0	1	1

$g_1 = b_2 \overline{b_1} + \overline{b_2} b_1$

$\overline{g_1} = \overline{b_2 \overline{b_1}} \cdot \overline{\overline{b_2} \cdot b_1}$

g_0

b_3b_2 \ b_1b_0	00	01	11	10
00	0	1	0	1
01	0	1	0	1
11	0	1	0	1
10	0	1	0	1

$g_0 = \overline{b_1} b_0 + b_1 \overline{b_0}$

$\overline{g_0} = \overline{\overline{b_1} b_0} \cdot \overline{b_1 \overline{b_0}}$

Partie 3 : Synthèse logique séquentielle

Exercice 3 [/6]

La méthodologie de conception de machine à états finis, *Finite State Machine* en anglais, s'applique aux systèmes séquentiels électroniques mais également à la programmation informatique dans le cas de la programmation objet par exemple.

L'objectif de cet exercice est la synthèse d'une machine à états finis de type Moore implémentant le diagramme d'état décrit dans la figure 1.

Remarque : veillez à expliquer votre démarche et à présenter vos résultats intermédiaires.

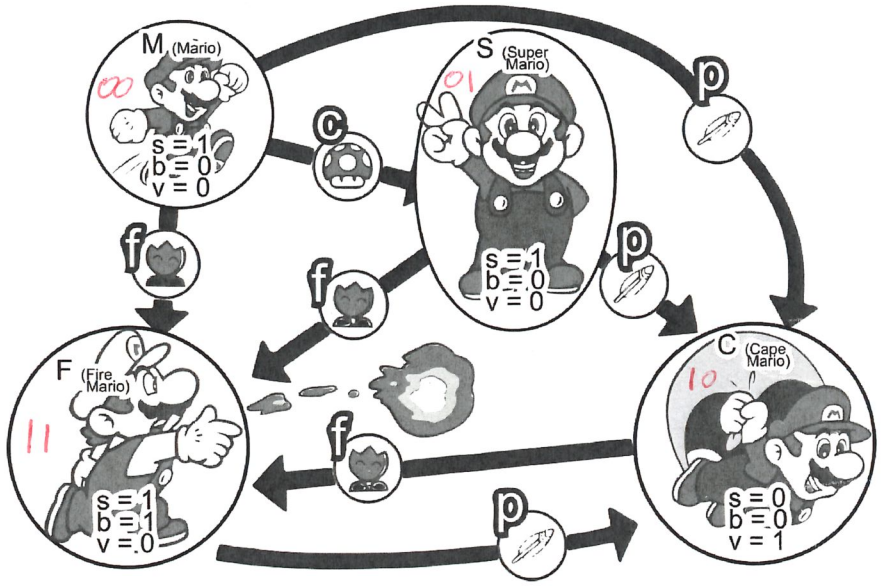


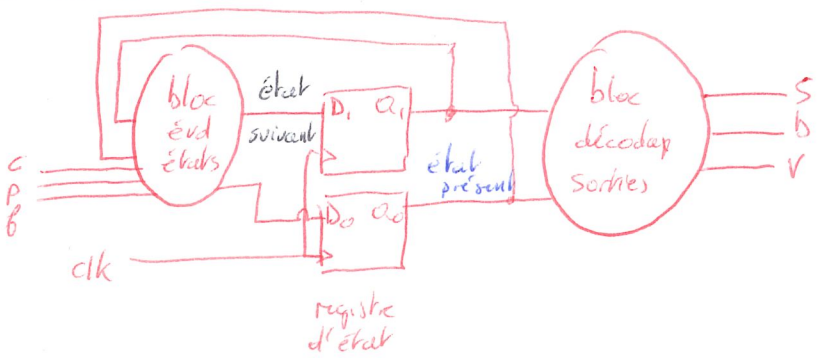
Figure 1: diagramme d'états de l'évolution de Mario
(c : champignon, p : plume, f : fleur de feu / s :saut, b : boule de feu, v : vole)

Question 3.1 : A partir du diagramme d'états, identifiez les nombres d'états, d'entrées et de sorties.

4 états → registre d'état à 2 bascules
3 entrées : c, f et p
3 sorties : s, b et v

état	Q_1 Q_0
M	0 0
S	0 1
C	1 0
F	1 1

Super
Cape
Fire



Question 3.2 : Synthétisez la machine d'états correspondant à ce diagramme d'états. La synthèse ira jusqu'aux expressions logiques, il n'est pas nécessaire de dessiner les schémas à base de portes logiques. Réponses sur la page suivante, page 5.

On ne peut pas avoir plusieurs entrées actives en même temps c, p, f. Dans le table on reste à l'état présent pour ces combinaisons.

Table d'évolution des états

entrées c p f	état présent Q_1 Q_0			sorties état suivant D_1 D_0		
1 0 0	0	0	M	S	0	1
0 1 0	0	0	M	C	1	0
0 0 1	0	0	M	F	1	1
autres cas	0	0	M	M	0	0
0 1 0	0	1	S	C	1	0
0 0 1	0	1	S	F	1	1
autres cas	0	1	S	S	0	1
0 0 1	1	0	C	F	1	1
autres cas	1	0	C	C	1	0
0 1 0	1	1	F	C	1	0
autres cas.	1	1	F	F	1	1

Table de décodage des sorties

état présent	entrées Q_1 Q_0	sorties s b v
M	0 0	1 0 0
S	0 1	1 0 0
C	1 0	0 0 1
F	1 1	1 1 0

$s = \overline{Q_1} + Q_0$

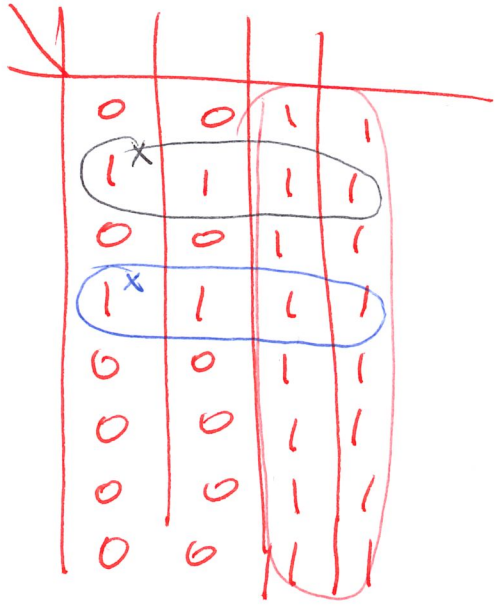
$b = Q_1 \cdot Q_0$

$v = Q_1 \cdot \overline{Q_0}$

$D_1\ D_0$

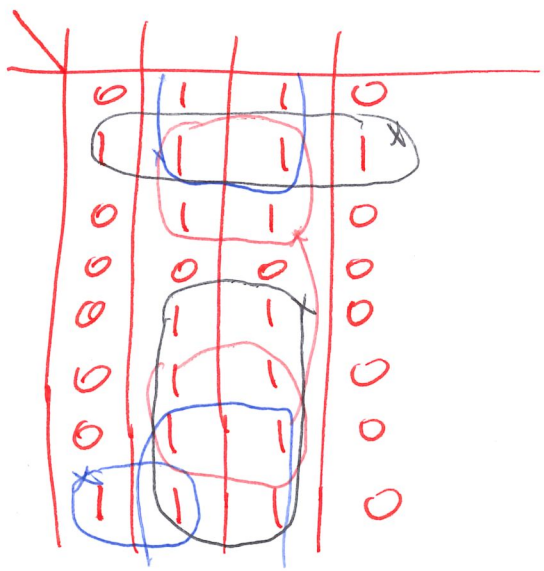
cpf \ Q_1Q_0	00	01	11	10
000	00	01	11	10
001	11	11	11	11
011	00	01	11	10
010	10	10	10	10
110	00	01	11	10
111	00	01	11	10
101	00	01	11	10
100	01	01	11	10

D_1



$$D_1 = Q_1 + \bar{c}pf + c\bar{p}f$$

D_0



$$D_0 = \bar{c}\bar{p}f + cp\bar{f}\bar{Q}_1 + cQ_0 + \bar{p}Q_0 + fQ_0$$

Partie 4 : Optionnel, partie bonus [+5]

Exercice 4 [/5] Optionnel, point de bonus [+5]

L'objectif est la synthèse d'un transcodeur code binaire vers code Gray en suivant une approche modulaire.

Le passage d'une représentation binaire B à une représentation en code Gray G peut se faire suivant : $G = (B \oplus 2 \cdot B)/2$.

Avec « ⊕ » une addition bit à bit modulo 2. « 2· » représente un décalage à gauche et « /2 » un décalage à droite.

Par exemple pour la valeur décimale 14 représentée par 1110 en binaire :

14 en binaire → 1 1 1 0
décalage à gauche → 1 1 1 0
addition bit à bit modulo 2 → 1 0 0 1 0
décalage à droite → 1 0 0 1
14 en code Gray

Question 4.1 [/1] [+1]

Vérifiez le principe de passage de la représentation binaire au code Gray avec les valeurs décimales 2 et 11.

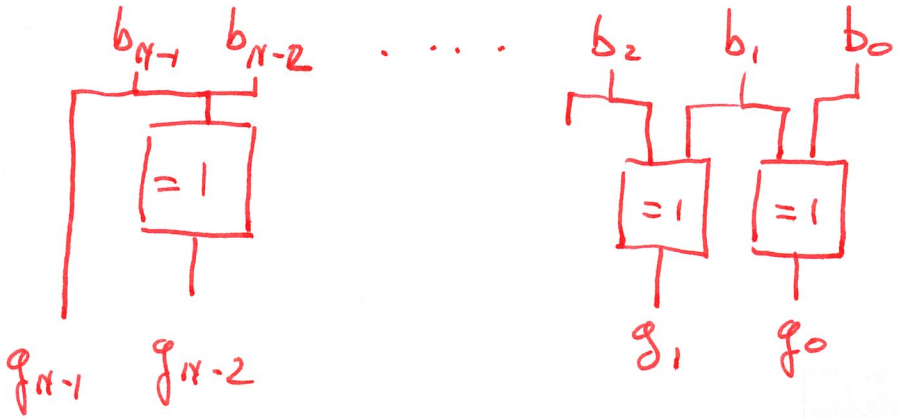
0 0 1 0
0 0 1 0
0 0 1 1 0
→ 0 0 1 1 code 2 en Gray

1 0 1 1
1 0 1 1
1 1 1 0 1
→ 1 1 1 0 code de 11 en Gray

Question 4.2 [/4] [+4]

Proposez une structure pour un transcodeur modulaire délivrant le code Gray sur N bits d'un nombre représenté en binaire sur N bits.

⊕ addition modulo 2 correspond au ou-exclusif. =1



pour un code sur N bits on fait le ou-exclusif bit à bit du code binaire et du code binaire décalé d'un bit à gauche.

→ le bit de poids fort est conservé tel quel.

Remarque : veillez à expliquer votre démarche.