EXAMEN D'ELECTRONIQUE NUMERIQUE INTEGREE - RATTRAPAGE– Février 2012 - *CORRIGE*

1. Analyse de système logique

1. Pour quelle combinaison de CBA s'ouvre la serrure ? CBA = $\boxed{110}$ $a_2a_1a_0 = 110$ (e_6 actif); $O_S = 1$ si $CBA = 110$, $O_S = 0$ sinon.

2. Donner les équations de J_C, K_C, J_B, K_B, J_A et K_A en fonction de C, B, A et E :

$J_C =$ $\boxed{B}$

$K_C =$ $\boxed{\overline{B} + A \cdot E}$

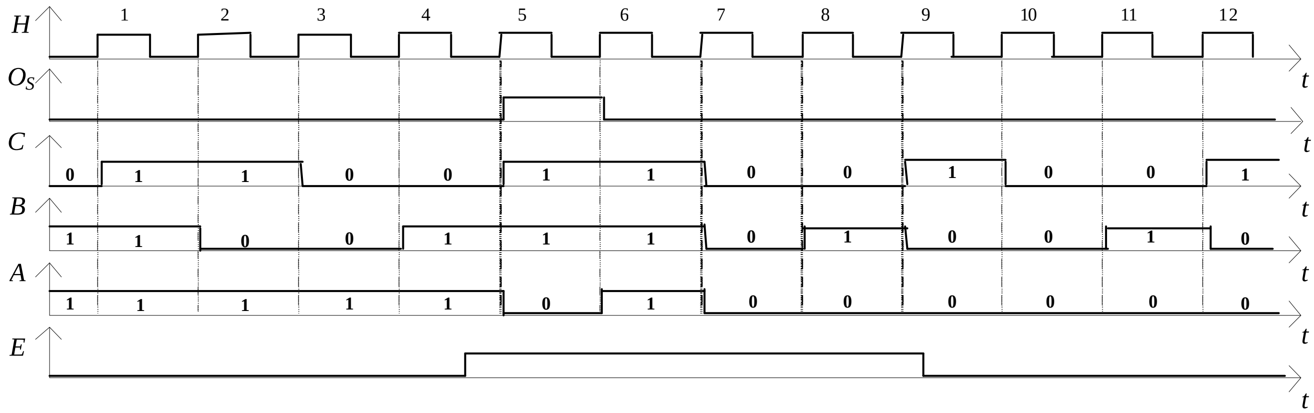
$J_B =$ $\boxed{\overline{C} \cdot \left(\overline{E} + \overline{A}\right)}$

$K_B =$ $\boxed{\overline{A \oplus C}}$

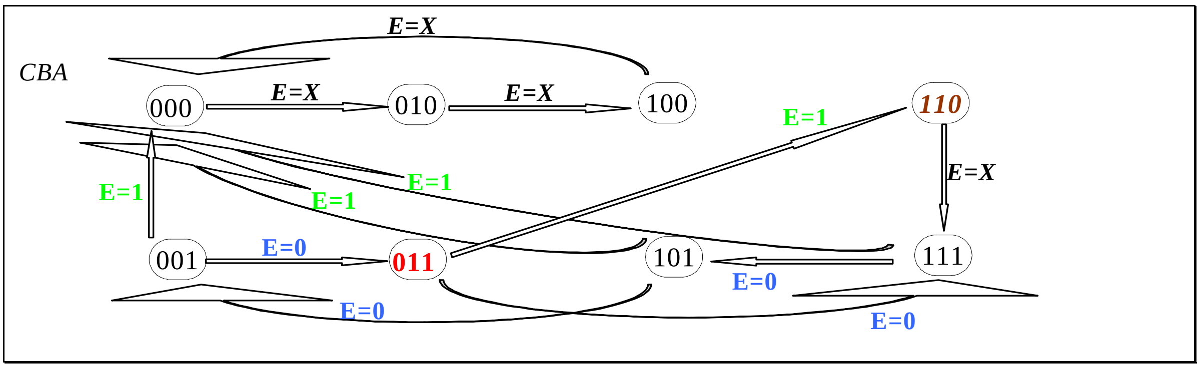
$J_A =$ $\boxed{B \cdot C}$

$K_A =$ $\boxed{E}$

3. Compléter le chronogramme avec l'état initial CBA = 011.



4. Compléter le diagramme des états-transitions du compteur CBA en tenant compte de **toutes les transitions possibles**. On précisera la valeur de E (0, 1 ou X si indifférent) sur chaque arc de transition :



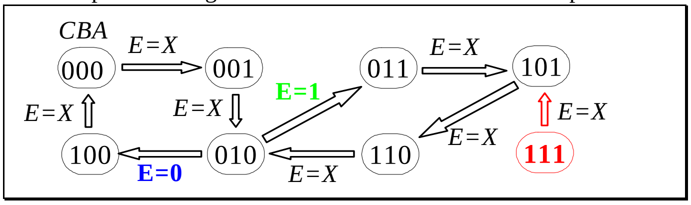
5. Quand faut-il actionner E pour pouvoir ouvrir la serrure ? CBA= $\boxed{011}$

6. Que se passe-t-il quand E est actionné sur les autres états ? $\boxed{\text{serrure bloquée (boucle)}}$

Si on essaye d'ouvrir la serrure au mauvais moment, le circuit rentre dans la boucle infinie 000→010→100→000 => ***La serrure reste bloquée***

2. Synthèse de circuit logique

1. Compléter le diagramme d'états suivant en tenant compte des 2 valeurs possibles de E.



2. Concevoir un circuit synchrone, de réalisation **la plus simple**, cadencé par une horloge *CLK* et constitué de 3 bascules *JK positive edge triggered*, de sorties Q_2, Q_1, Q_0 et d'entrées respectives J_2K_2, J_1K_1, et J_0K_0 permettant de produire la séquence d'entrée ***CBA=Q₂Q₁Q₀*** décrite ci-dessus :

J_2 (Q_2Q_1 \ Q_0E)	00	01	11	10
00	0	0	0	0
01	1	0	1	1
11	X	X	X	X
10	X	X	X	X

J_1 (Q_2Q_1 \ Q_0E)	00	01	11	10
00	0	0	1	1
01	X	X	X	X
11	X	X	X	X
10	0	0	1	1

J_0 (Q_2Q_1 \ Q_0E)	00	01	11	10
00	1	1	X	X
01	0	1	X	X
11	0	0	X	X
10	0	0	X	X

K_2 (Q_2Q_1 \ Q_0E)	00	01	11	10
00	X	X	X	X
01	X	X	X	X
11	1	1	X	X
10	1	1	0	0

K_1 (Q_2Q_1 \ Q_0E)	00	01	11	10
00	X	X	X	X
01	1	0	1	1
11	0	0	X	X
10	X	X	X	X

K_0 (Q_2Q_1 \ Q_0E)	00	01	11	10
00	X	X	1	1
01	X	X	0	0
11	X	X	X	X
10	X	X	1	1

$J_2 = \boxed{Q_0 \cdot Q_1 + Q_1 \cdot \overline{E}}$ $J_1 = \boxed{Q_0}$ $J_0 = \boxed{\overline{Q}_1 \cdot \overline{Q}_2 + \overline{Q}_2 \cdot E}$

$K_2 = \boxed{\overline{Q}_0}$ $K_1 = \boxed{Q_0 + \overline{Q}_2 \cdot \overline{E}}$ $K_0 = \boxed{\overline{Q}_1}$

3. Le circuit obtenu est-il auto-correcteur ? $\boxed{OUI}$

Justification :

L'état futur $Q_{2_{n+1}} Q_{1_{n+1}} Q_{0_{n+1}}$ de $Q_{2_n} Q_{1_n} Q_{0_n}$ = **111** : E = X :

$$\begin{cases} \left| \begin{array}{l} J_2 = 1 \\ K_2 = 0 \end{array} \right. & Q_{2_{n+1}} = 1 \\ \left| \begin{array}{l} J_1 = 1 \\ K_1 = 1 \end{array} \right. & Q_{1_{n+1}} = 0 \\ \left| \begin{array}{l} J_0 = 0 \\ K_0 = 0 \end{array} \right. & Q_{0_{n+1}} = 1 \end{cases}$$

$$\rightarrow \quad Q_{2_{n+1}} Q_{1_{n+1}} Q_{0_{n+1}} = 101$$

D'où : CBA = 111 $\xrightarrow{E=X}$ 101

3. Représentation des nombres en machine : somme et produit en flottant

1. Donner (*en Hexadécimal*) le codage IEEE754 simple précision des nombres flottants X = 14_{10} et Y = -12_{10}.

1. X $\boxed{14 = 41\ 60\ 00\ 00_{\ (HEXA)}}$ Y $\boxed{-12 = C1\ 40\ 00\ 00_{\ (HEXA)}}$

X = 14 = $2^k \cdot (1,\cdots)$ = 8 x 1,75 = 2^3 x (1 + 0,75) = $2^{130-127}$ x (1 + 0,75)
s (1) = **0**
e (8) = 130 = **1000 0010**
m (23) = 0,75 = 2^{-1} + 2^{-2} = **110 0000 0000 0000 0000 0000** = m_X
sem = **0100 0001 0110 0000 0000 0000 0000 0000 = 41 60 00 00$_{(H)}$**

Y = -12 = $-2^k \cdot (1,\cdots)$ = -8 x 1,5 = -2^3 x (1 + 0,5) = $-2^{130-127}$ x (1 + 0,5)
s (1) = **1**
e (8) = 130 = **1000 0010**
m (23) = 0,5 = 2^{-1} = **100 0000 0000 0000 0000 0000** = m_Y
sem = **1100 0001 0100 0000 0000 0000 0000 0000 = C1 40 00 00$_{(H)}$**

2. S = X + Y $\boxed{14 - 12 = 40\ 00\ 00\ 00_{\ (HEXA)}}$ Normalisation : $\boxed{OUI}$

X + Y = 14 - 12 =
```
   1,110 0000 0000 0000 0000 0000 * 2^(130-127)          (14)
-  1,100 0000 0000 0000 0000 0000 * 2^(130-127)          (12)
------------------------------------------------------------
   0,010 0000 0000 0000 0000 0000 * 2^(130-127)
=   1,000 0000 0000 0000 0000 0000 * 2^(128-127)         Normalisation
```
soit pour le résultat : **14 - 12 =**
s (1) = **0**
e (8) = 128 = **1000 0000**
m (23) = **000 0000 0000 0000 0000 0000**
sem = **0100 0000 0000 0000 0000 0000 0000 0000 = 40 00 00 00$_{(H)}$**

Vérification : A comparer au résultat **2** :
2 = $2^k \cdot (1,\cdots)$ = 2 x 1,0 = 2^1 x (1 + 0,0) = $2^{128-127}$ x (1 + 0,0)
s (1) = **0**
e (8) = 128 = **1000 0000**
m (23) = 0,0 = **000 0000 0000 0000 0000 0000**
sem = **0100 0000 0000 0000 0000 0000 0000 0000 = 40 00 00 00$_{(H)}$** **OK**

3. M = XY $\boxed{14x(-12) = C3\ 28\ 00\ 00_{\ (HEXA)}}$ Normalisation : $\boxed{OUI}$

M = X x Y = 14 x(-12) =
signe : **1** (xor entre les 2 signes des 2 opérandes)
exposant : **1000 0010 + 1000 0010 - (127)$_2$ = 1000 0010 + 1000 0010 – 0111 1111**
= 1000 0010 + 1000 0010 + (-127)$_{C2}$ = 1000 0010 + 1000 0010 + 1000 0001
= 1000 101 (= 133$_{10}$) = 6 + 127 **(OK : 6 = 3 + 3)**
mantisse : Multiplication des mantisses m_X et m_Y :
rq: (1 + m_X)(1 + m_Y) = 1 + $m_X.m_Y$ + m_X + m_Y → **m_Z = $m_X.m_Y$ + m_X + m_Y**
```
              1,mX =            1,11 0…0         * 2^(130-127)
    x         1,mY =       x  1,1   0…0          * 2^(130-127)
                              -----------
                                111
                               111
                              -----------
              1,mM =            10,101 0…0       * 2^(133-127) = 1,0101 0…0    * 2^(134-127)       Normalisation
```
→ **m_M = 0101** 0…0 → **m_M (23) = 010 1000 0000 0000 0000 0000**
→ *exposant* : **1000 110 (= 134$_{10}$)**
s (1) : **1**
e (8) : **1000 0110** → **e = 134**
m (23): **010 1000 0000 0000 0000 0000**
sem = 1100 0011 0010 1000 0000 0000 0000 0000 = C3 28 00 00$_{(H)}$

Vérification du résultat : **M**
M (exact) = -168 = $-2^k \cdot (1,\cdots)$ = -128 x 1,3125 = -2^7 x (1 + 0,3125) = $-2^{134-127}$ x (1 + 0,3125)
s (1) = **1**
e (8) = 134 = **1000 0110** = **7** % 127
m (23) = 0,3125 = 0.25 + 0.0625 = 2^{-2} + 2^{-4} = **010 1000 0000 0000 0000 0000**

sem = **1100 0011 0010 1000 0000 0000 0000 0000 = C3 28 00 00$_{(H)}$** **OK**

4. Pipelining

1. Combien de bulles sont insérées dans le pipeline pour la méthode 1 ? $\boxed{6}$ bulles
- *l'instruction 2 engendre 2 bulles*
- *l'instruction 3 engendre 0 bulle*
- *l'instruction 4 engendre 2 bulles*
- *l'instruction 5 engendre 2 bulles*

2. Indiquer le coût temps calcul, exprimé *en nombre de cycles*, du programme de la méthode 1 : $\boxed{11}$ *cycles*
- *Nombre de cycles = nombre d'instructions + nombre de bulles*

3. Combien de bulles sont insérées dans le pipeline pour la méthode 2 ? $\boxed{3}$ bulles
- *l'instruction 2 engendre 0 bulle*
- *l'instruction 3 engendre 1 bulle*
- *l'instruction 4 engendre 0 bulle*
- *l'instruction 5 engendre 2 bulles (et non pas 3 : les bulles ne s'ajoutent pas !)*

4. Indiquer le coût temps calcul, exprimé *en nombre de cycles*, du programme de la méthode 2 : $\boxed{8}$ *cycles*
- *Nombre de cycles = nombre d'instructions + nombre de bulles*

5. Mémoire et suite arithmétique

1. Indiquer le nombre de mots nécessaires pour stocker un terme de la suite en mémoire $\boxed{8}$ *8 octets (64 bits)*

2. Indiquer le nombre de bits d'adresse $\boxed{16}$ *Capacité = 64 Ko avec 1 octet par adresse (@) → 64x1024 =* 2^{16} *@ →16 bits d'@*

3. Combien de termes de la suite arithmétique peut-on stocker dans cette mémoire ? $\boxed{2^{13} = 8192}$ *8 @ / terme →* $\frac{2^{16}}{8}$ *termes*

4. A quelle adresse (en hexadécimal) débute le dernier terme de la suite contenu dans la mémoire ? $\boxed{0xFFF8}$
Dernière adresse *(16 bits = 4 caractères Hexa)* = $0xFFFF$ *→ dernier terme stocké en* $0xFFFF - 8 + 1 = 0xFFF8$.

6. Microprogrammation

1. Quelle est l'addition réalisée ? Donner la valeur des opérandes et du résultat en décimal :

$$\boxed{D_1 + D_2 = 0x27 + 0xFA = 39_{10} + (-6)_{10} = 33_{10} = 0x21}$$

Opération réalisée : 0x27+0xFA=0x21 soit en décimal : 39 + (-6) = 33

$$R_2 + R_3 = 0x7B + 0xFF = A = 0x7A \quad \rightarrow \quad (A) = 0x27 = D_1$$
$$R_1 = 0x4A \quad \rightarrow \quad (R_1) = 0xFA = D_2$$
$$0xFA - 1 = 0xF9 \quad \xrightarrow{Cà1} \quad +6 \quad \rightarrow \quad 0xFA = -6 \text{ en code Cà2}$$
$$0x27 = 2x16 + 7 = 39_{10}$$

	Micro-instruction	R1	R2	R3	MDR	MAR	Y	Bus	
	État initial	0x4A	0x7B	0xFF	0xAA	0x02	0x03	0x03	
1	R2 out							0x7B	Y <= R2
2	Y in						0x7B		
3	R3 out							0xFF	Y <= R2 + R3
4	ADD						0x7A		
5	Y out							0x7A	MDR <= (R2+R3)
6	MAR in					0x7A			
7	Lecture; Attente				0x27				
8	MDR out							0x27	Y <= MDR (1ère donnée D1)
9	Y in						**0x27**		
10	R1 out							0x4A	MDR <= (R1) 2$^{\text{ème}}$ donnée D2 sur le bus
11	MAR in					0x4A			
12	Lecture; Attente				0xFA				
13	MDR out							**0xFA**	
14	ADD						0x21		Y <= D1 + D2
15	Y out							**0x21**	(R1) <= Y Ecriture somme à l'@ R1 (déjà dans MAR)
16	MDR in								
17	Ecriture; Attente								

Execute