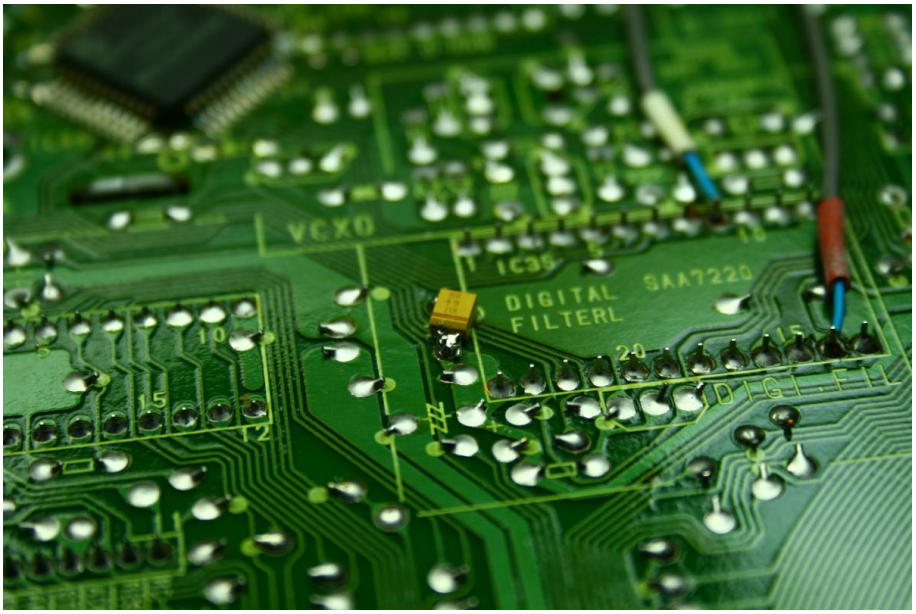
Electronique Numérique

Année académique 2014-2015

Répétitions



Pr suppléant **P. Rousseaux**
Assistants **D. Cerica**, **Q. Massoz**,
V. Pierlot, **T. Schmitz**

Table des matières

1 Rappels théoriques **3**

1.1 Nombres et codes . . . 3

1.1.0.1 Représentation d'un nombre . . . 3

1.1.1 Conversion base r vers base 10 . . . 3

1.1.2 Conversion base 10 vers base r . . . 3

1.1.3 Conversion base r vers base s . . . 4

1.1.4 Cas particuliers : binaire $\leftrightarrow$ octal / binaire $\leftrightarrow$ hexadécimal . . . 4

1.1.5 Compléments . . . 4

1.1.6 Soustraction en binaire . . . 5

1.2 Algèbre booléenne . . . 6

1.2.1 Les opérations . . . 6

1.2.2 Les identités de base . . . 7

1.2.2.1 A une seule variable . . . 7

1.2.2.2 A plusieurs variables . . . 7

1.2.2.3 Théorème de De Morgan . . . 7

1.2.2.4 Théorème du Consensus . . . 7

1.2.3 Recherche du complément d'une fonction logique . . . 7

1.2.4 Implémentation de fonctions . . . 8

1.3 Simplification de fonction . . . 8

1.3.1 Portes universelles NAND et NOR . . . 8

1.3.2 Portes XOR et NXOR . . . 8

1.3.3 Représentation canonique des fonctions booléennes . . . 9

1.3.3.1 Somme de minterms . . . 9

1.3.3.2 Produit de maxterms . . . 9

1.3.3.3 Remarques . . . 9

1.3.4 Tables de Karnaugh . . . 9

2 Electronique combinatoire **11**

3 Electronique séquentielle **25**

4 Langage VHDL **33**
4.1 logique combinatoire 33
4.2 logique séquentielle 43

1 Rappels théoriques

1.1 Nombres et codes

1.1.0.1 Représentation d'un nombre

$$A_r = (A_n A_{n-1} ... A_1 A_0 \bullet A_{-1} A_{-2} ...)_r$$

où r désigne la base, A_i est une valeur comprise entre $[0, r-1]$ et i donne la position.

1.1.1 Conversion base r vers base 10

$$(A)_{10} = A_n \cdot r^n + A_{n-1} \cdot r^{n-1} + ... + A_0 \cdot r^0 \bullet A_{-1} \cdot r_{-1} + ...$$

1.1.2 Conversion base 10 vers base r

$$N = N_e + N_f$$

où N_e désigne la partie entière et N_f la partie fractionnaire

Partie entière

$$\begin{gathered} N_e/r = Q_0 + R_0 \\ Q_0/r = Q_1 + R_1 \\ Q_1/r = Q_2 + R_2 \\ ... \end{gathered}$$

On répète l'opération jusqu'à ce que $Q_{j+1} = 0$.

On obtient :

$$N_e = (R_{j+1} R_j ... R_1 R_0)_r$$

Partie fractionnaire

$$\begin{gathered} N_f \times r = P_{e_0} + P_{f_0} \\ P_{f_0} \times r = P_{e_1} + P_{f_1} \\ P_{f_1} \times r = P_{e_2} + P_{f_2} \\ \ldots \end{gathered}$$

On obtient :

$$N_f = (P_{e_0} P_{e_1} P_{e_2} P_{e_3} \ldots)_r$$

$$\Rightarrow N = (R_{j+1} R_j \ldots R_1 R_0 \bullet P_{e_0} P_{e_1} P_{e_2} P_{e_3} \ldots)_r$$

1.1.3 Conversion base r vers base s

$$\begin{gathered} \text{base } r \rightarrow \text{base } 10 \\ \text{base } 10 \rightarrow \text{base } s \end{gathered}$$

1.1.4 Cas particuliers : binaire $\leftrightarrow$ octal / binaire $\leftrightarrow$ hexadécimal

a. binaire $\leftrightarrow$ octal : 3 chiffres binaires = 1 chiffre octal

b. binaire $\leftrightarrow$ hexadécimal : 4 chiffres binaires = 1 chiffre hexadécimal

Exemple :

$$\begin{gathered} (10111101000)_2 = (??)_8 = (??)_{16} \\ (10\ 111\ 101\ 000)_2 = (2\ 7\ 5\ 0)_8 \\ (101\ 1110\ 1000)_2 = (5\ E\ 8)_{16} = 0x5E8 \end{gathered}$$

1.1.5 Compléments

Soit

$$N = (n.m)_r$$

où n, m et r représentent respectivement les chiffres de partie entière, les chiffres de la partie fractionnaire et la base.

Complément à $r - 1$:

$$r^n - r^{-m} - N = (r^n - N) - r^{-m}$$

Complément à r :

$$r^n - N$$

En binaire, ces formules donnent lieu à des manipulations simples.

— **Complément à 1** : inverser les bits de N.
— **Complément à 2** : laisser les "0" les moins significatifs et le premier "1" inchangés et inverser les autres bits.

1.1.6 Soustraction en binaire

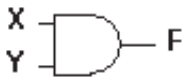
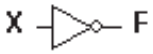
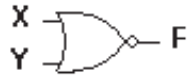
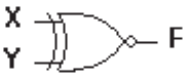
$$M - N = M + \text{ complément à } 2[N] = M + (2^n - N)$$

a. Si $M \geq N$
Le résultat peut s'écrire $2^n + (M - N)$ où $(M - N)$ est le résultat final. Par conséquent, il faut ignorer le bit "1" de poids fort.

b. Si $M < N$
Le résultat peut s'écrire $2^n - (N - M)$ où $N - M > 0 \Rightarrow$ on a calculé le complément à 2 de $(N - M)$. Par conséquent, le résultat final s'écrit : -(complément à 2 du résultat).

1.2 Algèbre booléenne

1.2.1 Les opérations

Nom	Symbole	Equation	Table de vérité
AND		$F = XY$	X Y F 0 0 0 0 1 0 1 0 0 1 1 1
OR		$F = X + Y$	X Y F 0 0 0 0 1 1 1 0 1 1 1 1
NOT		$F = \overline{X}$	X F 0 1 1 0
NAND		$F = \overline{XY}$	X Y F 0 0 1 0 1 1 1 0 1 1 1 0
NOR		$F = \overline{X + Y}$	X Y F 0 0 1 0 1 0 1 0 0 1 1 0
XOR		$F = X \oplus Y$	X Y F 0 0 0 0 1 1 1 0 1 1 1 0
NXOR		$F = \overline{X \oplus Y}$	X Y F 0 0 1 0 1 0 1 0 0 1 1 1

1.2.2 Les identités de base

1.2.2.1 A une seule variable

1. $X + 0 = X$
2. $X + 1 = 1$
3. $X + X = X$
4. $X + \overline{X} = 1$
5. $X.1 = X$
6. $X.0 = 0$
7. $X.X = X$
8. $X.\overline{X} = 0$
9. $\overline{\overline{X}} = X$

1.2.2.2 A plusieurs variables

1. $X + Y = Y + X$
2. $X + (Y + Z) = X + Y + Z$
3. $X.(Y + Z) = X.Y + X.Z$
4. $X.Y = Y.X$
5. $X.(Y.Z) = X.Y.Z$
6. $X + (Y.Z) = (X + Y).(X + Z)$
7. $X + (X.Y) = X.(X + Y) = X$

1.2.2.3 Théorème de De Morgan

$$\overline{X_1 + X_2 + ... + X_n} = \overline{X_1}.\overline{X_2}...\overline{X_n}$$

$$\overline{X_1.X_2...X_n} = \overline{X_1} + \overline{X_2} + ... + \overline{X_n}$$

1.2.2.4 Théorème du Consensus

$$X.Y + \overline{X}.Z + Y.Z = X.Y + \overline{X}.Z$$

$$(X + Y).(\overline{X} + Z).(Y + Z) = (X + Y).(\overline{X} + Z)$$

1.2.3 Recherche du complément d'une fonction logique

— Permuter les "1" et les "0" dans la table de vérité de la fonction.
— Appliquer le théorème De Morgan.

1.2.4 Implémentation de fonctions

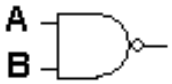
Compromis :
— Minimiser le nombre de portes logiques (coût).
— Minimiser le nombre d'étages (performances).

1.3 Simplification de fonction

1.3.1 Portes universelles NAND et NOR

— **NAND** :

$$F = \overline{A.B} = \overline{A} + \overline{B} \quad \leftrightarrow \quad F = \text{somme de produits}$$



— **NOR** :

$$F = \overline{A + B} = \overline{A}.\overline{B} \quad \leftrightarrow \quad F = \text{produit de sommes}$$

1.3.2 Portes XOR et NXOR

— **XOR** :

$$F = A \oplus B = \overline{A}B + A\overline{B}$$

— **NXOR** :

$$F = \overline{A \oplus B} = \bar{A}\bar{B} + AB$$

1.3.3 Représentation canonique des fonctions booléennes

1.3.3.1 Somme de minterms

$$
\begin{aligned}
Ex : F(X,Y) &= \overbrace{\bar{X}.\bar{Y}}^{minterm\ 0} + \overbrace{\bar{X}.Y}^{minterm\ 1} + \overbrace{X.\bar{Y}}^{minterm\ 2} + \overbrace{X.Y}^{minterm\ 3} \\
&= \sum m(0,1,2,3)
\end{aligned}
\tag{1.1}
$$

1.3.3.2 Produit de maxterms

$$
\begin{aligned}
Ex : F(X,Y) &= \overbrace{(\bar{X}+\bar{Y})}^{maxterm\ 3}.\overbrace{(\bar{X}+Y)}^{maxterm\ 2}.\overbrace{(X+\bar{Y})}^{maxterm\ 1}.\overbrace{(X+Y)}^{maxterm\ 0} \\
&= \prod M(3,2,1,0)
\end{aligned}
\tag{1.2}
$$

1.3.3.3 Remarques

— $M_i = \overline{m_i}$
— Une fonction à n variables possède 2^n minterms/maxterms

1.3.4 Tables de Karnaugh

Principe : groupements de rectangles de 2^n éléments

— 2 variables

m_0	m_1
m_2	m_3

X \ Y	0	1
0	$\bar{X}\bar{Y}$	$\bar{X}Y$
1	$X\bar{Y}$	XY

Figure 1.1 – 2 variables

— 3 variables

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6

		00	01	11	10
	0	$\bar{X}\bar{Y}\bar{Z}$	$\bar{X}\bar{Y}Z$	$\bar{X}YZ$	$\bar{X}Y\bar{Z}$
X	1	$X\bar{Y}\bar{Z}$	$X\bar{Y}Z$	XYZ	$XY\bar{Z}$

Y, Z

Figure 1.2 – 3 variables

— 4 variables

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6
m_{12}	m_{13}	m_{15}	m_{14}
m_8	m_9	m_{11}	m_{10}

		00	01	11	10
	00	m_0	m_1	m_3	m_2
	01	m_4	m_5	m_7	m_6
W	11	m_{12}	m_{13}	m_{15}	m_{14}
	10	m_8	m_9	m_{11}	m_{10}

Y, X, Z

Figure 1.3 – 4 variables

— 5 variables

		00	01	11	10
	00	0	1	3	2
	01	4	5	7	6
B	11	12	13	15	14
	10	8	9	11	10

D, C, E

A = 0

		00	01	11	10
	00	16	17	19	18
	01	20	21	23	22
B	11	28	29	31	30
	10	24	25	27	26

D, C, E

A = 1

Figure 1.4 – 5 variables

2

Electronique combinatoire

1. Convertir de la base 2 vers la base 10 :

$$(1011)_2 = (\qquad)_{10}$$
$$(0.101)_2 = (\qquad)_{10}$$

2. Convertir de la base 10 vers la base 2 :

$$(514.75)_{10} = (\qquad)_2$$

3. Convertir de la base 10 vers la base 3 :

$$(1.3)_{10} = (\qquad)_3$$

4. Convertir de la base 2 vers la base 8 et la base 16 :

$$(011101111001)_2 = (\qquad)_8 = (\qquad)_{16}$$

5. Convertir vers la base 2 :

$$(47.3)_8 = (\qquad)_2$$
$$(AB.C)_{16} = (\qquad)_2$$

6. Additionner en base 2 et vérifier en passant par la base 10 :

$$(1001101)_2 + (0111001)_2 = (\qquad)_2$$

7. Soustraire en base 8 et vérifier en passant par la base 10 :

$$(1267.4)_8 - (452.3)_8 = (\qquad)_8$$

8. Multiplier en base 16 et vérifier en passant par la base 10 :

$$(13A)_{16} \cdot (452.3)_8 = (\qquad)_8$$

9. Diviser en base 8 et vérifier en passant par la base 10 :

$$(1250)_8/(4)_8 = (\qquad)_8$$

10. Calculer les compléments à 9 et à 10 de :

$$(72532.17)_{10}$$

11. Calculer les compléments à 1 et à 2 de :

$$(101100)_2$$

Calculer le complément à 2 de :

$$(100)_2$$
$$(0.001)_2$$

12. Soustraire en utilisant les compléments :

$$(11010)_2 - (1101)_2 = (\qquad)_2$$
$$(1101)_2 - (11010)_2 = (\qquad)_2$$

13. Simplifier les expressions suivantes :
    a. $X + \overline{X}Y$
    b. $\overline{X} \cdot (X + Y)$
    c. $ABC + AB\overline{C} + A\overline{B}C$
    d. $\bar{A}\bar{B}\bar{C} + \overline{A}BC + ABC + A\bar{B}\bar{C} + A\overline{B}C$
    e. $(A + C + D)(A + C + \overline{D})(A + \overline{C} + D)(A + \overline{B})$
    f. $\overline{(A + B)} \cdot \overline{(\overline{A} + \overline{B})}$
    g. $\overline{(\overline{CD} + A)} + A + AB + CD$
    h. $\overline{[(\overline{X} + Z) \cdot (\overline{X} + \overline{Z}) \cdot \overline{Y}] + [\overline{(\overline{X} + Z)} + \overline{Y + Z}]}$

14. Chercher le complément de la fonction F (Fig.2.1) de 2 manières différentes.

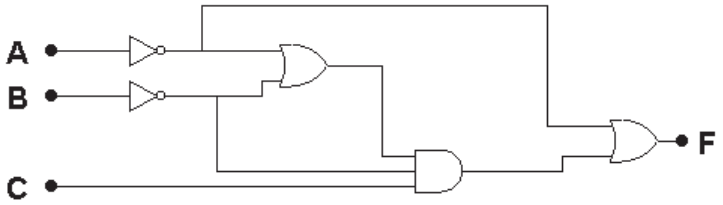


Figure 2.1 – Ex. 14

15. Chercher le complément de la fonction
    a. $F = BC + A(B + C)$
    b. $F = (M + N)(\overline{M} + P)(\overline{N} + \overline{P})$
    c. $F = [(\overline{AB}) \cdot A][(\overline{AB}) \cdot B]$
16. Implémenter les fonctions suivantes de manière minimale.
    a. $F = AB + \overline{A}\overline{B} + \overline{A}C$
    b. $F = (A + B)(C\overline{D} + \overline{C}D)$
    c. $F = \overline{RST} \cdot \overline{(R + S + T)}$
17. Que vaut la fonction duale du circuit suivant (Fig.2.2)

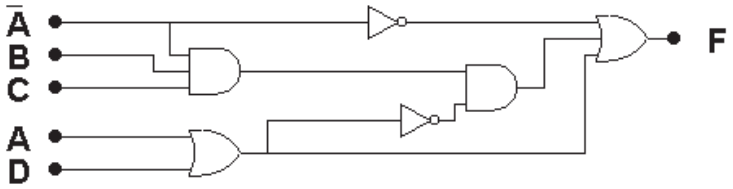


Figure 2.2 – Ex.17

18. Implémenter les fonctions suivantes au moyen d'interrupteurs, en utilisant dans un premier temps des portes simples et ensuite en utilisant des portes complexes.
    a. $F_1 = \overline{A + BC}$
    b. $F_2 = \overline{A} + \overline{B}\overline{C}$
19. Implémenter le circuit suivant (fig.2.3) au moyen de portes NAND

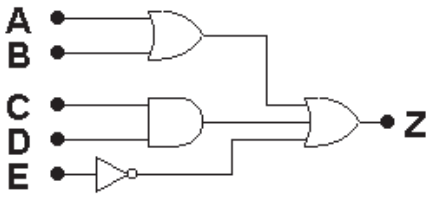


Figure 2.3 – ex.19

20. Implémenter le circuit suivant (fig.2.4) au moyen de portes NOR
21. Pour quelles conditions $MEM = 0$ (fig.2.5) ?
22. Pour quelles conditions $X = 1$ (fig.2.6) ?
23. Montrer que $A \oplus B \oplus C = A \oplus (B \oplus C)$
24. Quel pourrait être le schéma éléectrique de la porte XOR ?
25. Représenter la fonction suivante et son complément sous forme de minterms et de maxterms :

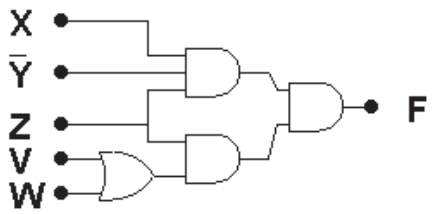


Figure 2.4 – ex.20

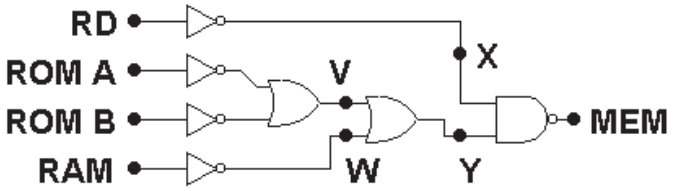


Figure 2.5 – ex.21

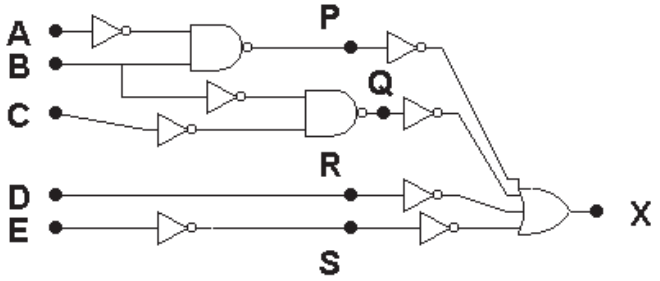


Figure 2.6 – ex.22

a. $F(A, B, C) = A\overline{C} + \overline{B}C$
b. $F(W, X, Y, Z)) = XYZ + \overline{Y}Z + WX$

26. Réduire par Karnaugh et vérifier votre réponse :

$$F(X, Y, Z) = \sum m(2, 3, 4, 6, 7)$$

27. Réduire par Karnaugh :

a. $F(A, B, C, D) = \sum m(1, 5, 6, 7, 11, 12, 13, 15)$
b. $F(A, B, C, D) = \sum m(4, 6, 7, 8, 10, 15)$
c. $F(A, B, C, D) = \sum m(1, 5, 6, 7, 8, 9, 10, 14)$

28. Réduire par Karnaugh. Exprimer F et $\overline{F}$ sous forme de produit de somme(s).

$$F(A, B, C, D) = \sum m(0, 1, 2, 3, 4, 5, 6, 9, 11, 13)$$

29. Simplifier :

a. $F(A, B, C, D) = \bar{A}\bar{C}\bar{D} + \overline{B}C + \overline{A}BC\overline{D} + A\overline{BD} + \bar{A}\bar{B}D + \bar{A}\bar{B}C\bar{D}$
b. $F(A, B, C, D, E) = \sum m(0, 1, 8, 9, 13, 16, 17, 25, 27, 29, 31)$

30. Exprimer la fonction sous forme de produit de somme(s).

$$F(A, B, C, D, E) = \sum m(0, 1, 2, 3, 5, 7, 8, 10, 12, 14, 16, 17, 18, 20, 21, 24, 26, 28, 30)$$

31. Que valent F et $\overline{F}$ (fig.2.7) ?

		C		
$A\{$	1	0	1	1
	X	0	0	X
			B	

Figure 2.7 – ex.31

32. Simplifier :

a. $F(A, B, C, D, E) = \sum m(0, 5, 16, 19, 21, 25, 26) + \sum d(13, 17, 27, 29)$
b. $F(A, B, C, D, E) = \sum m(3, 5, 7, 9, 17, 27, 29, 31) + \sum d(10, 15, 24, 30)$
c. $F(A, B, C, D, E) = \sum m(1, 6, 9, 11, 12, 14, 16, 18, 21, 26, 31)$
$+ \sum d(3, 4, 5, 7, 15, 22, 23, 24, 25, 29, 30)$

33. Analyser la fig.2.8 :

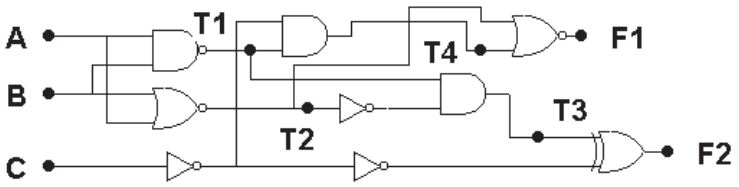


Figure 2.8 – ex.33

34. Analyser le circuit suivant (fig.2.9), donner la table de vérité des sorties et les implémenter de manière optimale à l'aide d'un nombre minimal de portes **NAND**. Les entrées complémentées sont disponibles.

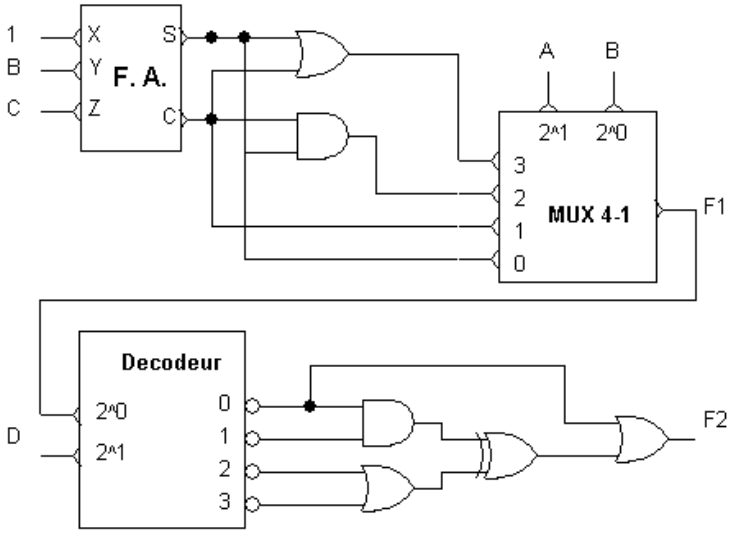


Figure 2.9 – ex.34

35. Analyser le circuit présenté à la figure 2.10. Donner la table de vérité et implémenter la fonction F à l'aide d'un multiplexeur de taille minimale.

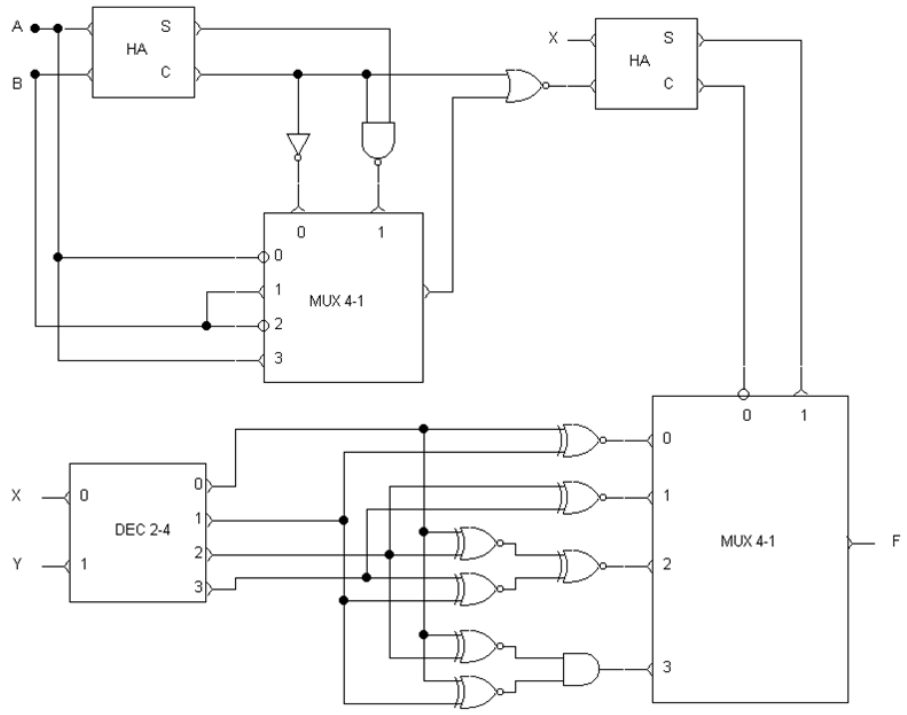


Figure 2.10 – ex.35

36. Analyser le circuit présenté à la figure 2.11. Donner la table de vérité de la fonction de sortie (placer les entrées dans l'ordre alphabétique) et donner son expression canonique sous forme de produit de sommes.

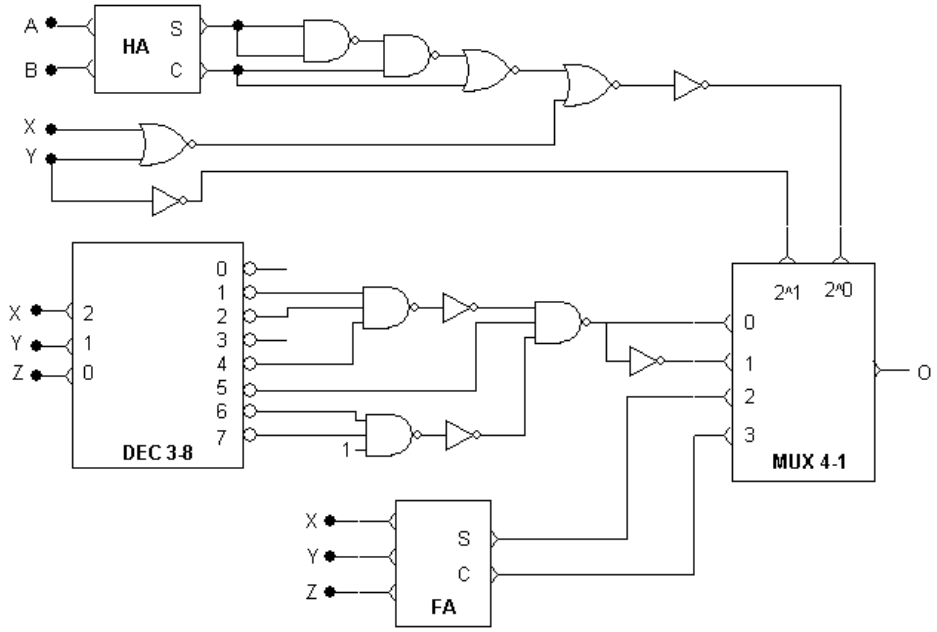


Figure 2.11 – ex.36

37. Analyser le circuit présenté à la figure 2.12. Donner la table de vériré de la sortie F. Implémenter cette sortie à l'aide d'un nombre minimum de portes NOR. Les entrées complémentées sont également disponibles.

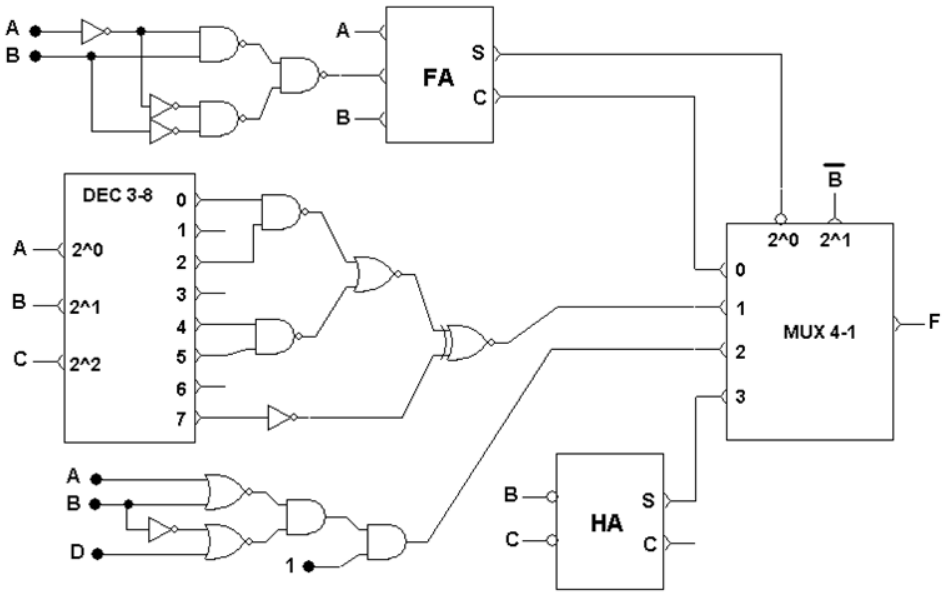


Figure 2.12 – ex.37

38. Analyser le circuit présenté à la figure 2.13. Donner la table de vériré de la sortie F (placer les entrées dans l'ordre alphabétique). Implémenter cette sortie à l'aide d'un nombre minimum de portes NAND. Les entrées complémentées sont disponibles.

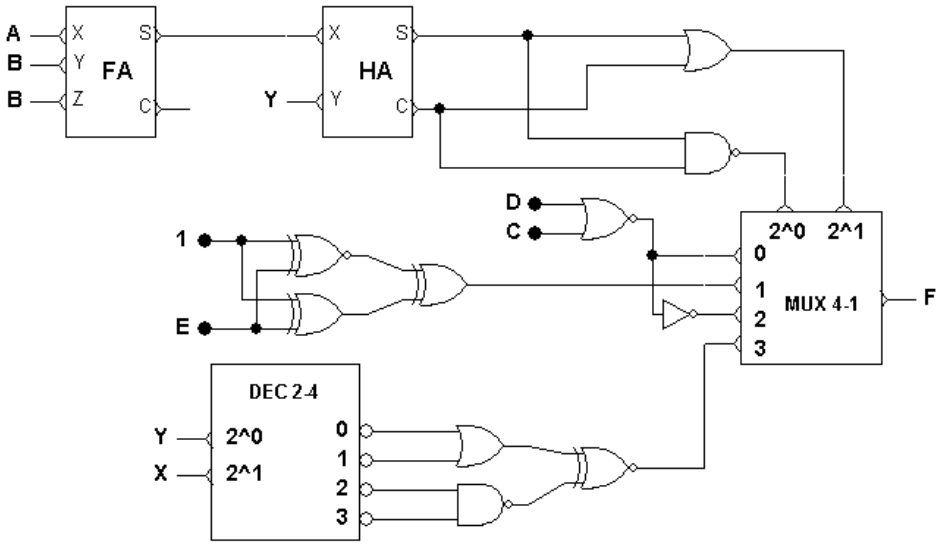


Figure 2.13 – ex.38

39. Analyser le circuit présenté à la figure 2.14. Donner la table de vériré de la sortie F (placer les entrées dans l'ordre alphabétique). Implémenter la sortie à l'aide d'un multiplexeur de taille minimale. Les entrées complémentées sont disponibles.

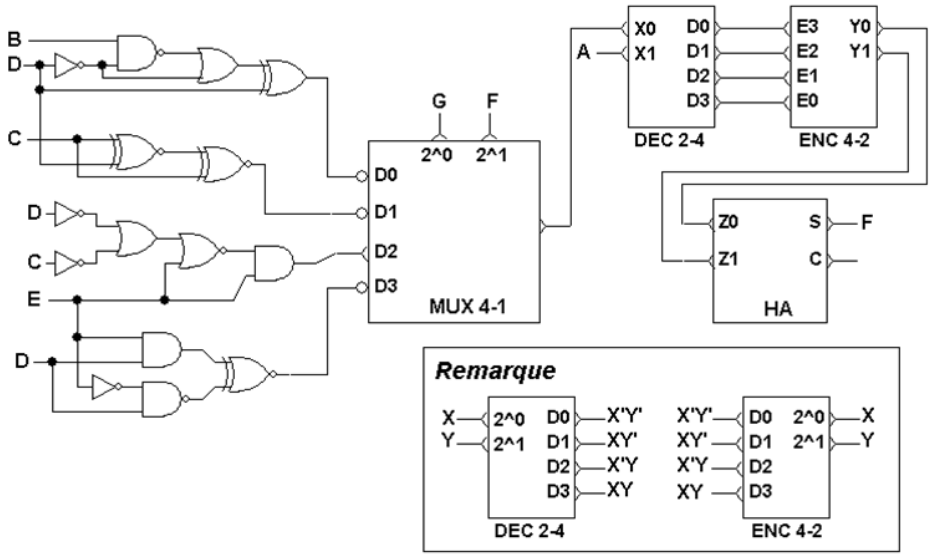


Figure 2.14 – ex.39

40. Soit $F(z, y, x, w) = \sum m(0, 3, 5, 7, 8, 11, 13, 15) + \sum d(2, 10)$
    (a) Simplifier la fonction F en utilisant la méthode de Karnaugh.
    (b) Implémenter la fonction F à l'aide d'un multiplexeur de taille minimale. Chaque entrée du multiplexeur peut comporter **une et une seule** des portes logiques suivantes : AND, OR, XOR, NXOR.
41. Soit $F(d, c, b, a) = \sum m(1, 3, 4, 5, 7, 12, 14) + \sum d(6, 9)$
    (a) Simplifier la fonction F en utilisant la méthode de Karnaugh.
    (b) Implémenter la fonction F à l'aide d'un multiplexeur de taille minimale. Chaque entrée du multiplexeur peut comporter **une et une seule** des portes logiques suivantes : AND, OR, XOR, NXOR.
42. Implémenter la fonction F de manière optimale à l'aide d'un multiplexeur.
    $$F(a, b, c, d) = \sum m(1, 3, 10, 11, 14) + \sum d(4, 6, 7, 15)$$
43. Faire la synthèse du circuit suivant : un circuit combinatoire à 3 entrées dont la sortie vaut le carré de l'entrée.
44. Faire la synthèse du circuit suivant.
    Les critères de recrutement d'un compagnie aérienne sont les suivants :
    — être un homme célibataire et belge ;
    — être célibataire, belge et avoir moins de 25 ans ;
    — être une célibataire étrangère ;
    — être un homme de moins de 25 ans ;
    — être une femme célibataire de plus de 25 ans.
45. La figure 2.15 nous montre l'intersection entre une route principale et une route secondaire. Des capteurs de voitures ont été placés le long des voies C et D (route principale) et des voies A et B (route secondaire). Les sorties de ces capteurs sont à 0 quand il n'y a pas de voiture et à 1 quand il y en a. Le feu de circulation se trouvant à cette intersection est commandé par les règles de décision suivante :
    — Le feu E-O est vert quand il y a des voitures dans les deux voies C et D.
    — Le feu E-O est vert quand il y a des voitures dans C ou D mais si les bandes A et B ne sont pas toutes deux occupées.
    — Le feu N-S est vert quand il y a des voitures dans les voies A et B mais si les bandes C et D ne sont pas toutes deux occupées.
    — Le feu N-S est également vert quand il y a des voitures dans A ou B et qu'il n'y a pas de voiture dans C et D.
    — Le feu E-O est vert quand il n'y a pas de voiture du tout.
    En utilisant les tensions de sortie des capteurs A, B, C et D comme entrées, concevez un circuit logique qui commande le feu de circulation. Ce circuit a deux sorties, soit E-O et N-S, qui prennent la valeur haute quand le feu doit être vert. Simplifiez le plus possible le circuit et illustrez toutes les étapes.
46. On désire réaliser un système d'alarme qui est constitué de deux détecteurs dans un hall (un à gauche et un à droite), 2 détecteurs dans un bureau (un à gauche et un à droite) et deux sirènes (une extérieure et une intérieure). En supposant que les deux détecteurs d'une même pièce ne peuvent déclencher simultanéement, le fonctionnement du système est régi par les lois suivantes :

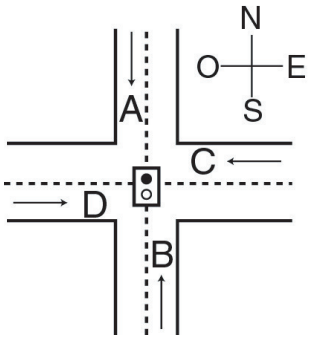


Figure 2.15 – ex.45

— La sonnerie intérieure retentira si un des détecteurs du bureau est enclenché sans qu'aucun détecteur du hall ne déclenche.
— Si un nombre impair de détecteurs s'enclenchent, la sirène intérieure retentira également. Par contre, si un nombre pair de détecteurs est sollicité, la sonnerie intérieure ne retentira que si les deux détecteurs à gauche ou les deux détecteurs à droite sont enclenchés.
— La sirène extérieure est, quant à elle, insensible aux détecteurs du bureau. Il en résulte qu'elle ne retentira qu'au moment où un individu pénètre dans le hall.
— Quand aucun détecteur n'est enclenché, les sirènes sont à l'arrêt.

Déterminer les équations booléennes de sortie de ce circuit. Implémenter la commande de la sirène intérieure au moyen de portes AND et XOR uniquement et la commande de la sirène extérieure au moyen d'un multiplexeur de taille minimale.

47. On vous demande d'implémenter le jeu "pierre-papier-ciseaux". Ce jeu oppose deux joueurs qui doivent, à chaque part, choisir l'une des trois possibilités : pierre, papier ou ciseaux. La victoire est attribuée comme suit :
— la pierre casse les ciseaux ;
— les ciseaux coupe le papier ;
— le papier emballe la pierre ;
— il y a égalité lorsque les deux joueurs choisissent le même objet.

On considère que les deux joueurs choisissent toujours un objet. Déterminer les entrées/sorties du système, établir la table de vérité ainsi que la(les) équation(s) de la (des) sortie(s).

48. On considère un robot dont la plateforme se compose de 4 capteurs disposés suivant la figure 2.16. On vous demande de réaliser un circuit combinatoire capable de modifier le mouvement du robot chaque fois que celui-ci rencontre un obstacle. Par défaut, le mouvement du robot est la marche avant. Le système est régi par les lois suivantes :
— Si le capteur F ou les 3 capteurs à l'avant sont activés, le robot se dégagera en marche arrière.
— Lorsque les capteurs F et D (resp. F et E) sont activés simultanément, le robot tournera vers la gauche (resp. droite).
— Quand seul le capteur D (ou seul le capteur E) est activé, le robot tourne dans le sens opposé au choc.
— Si le capteur A est activé, le robot repart en marche avant.
— Si les capteurs D et E sont activés simultanément, le mouvement du robot sera la marche arrière.

On considère que le capteur A ne sera jamais activé en même temps que l'un des capteurs D,

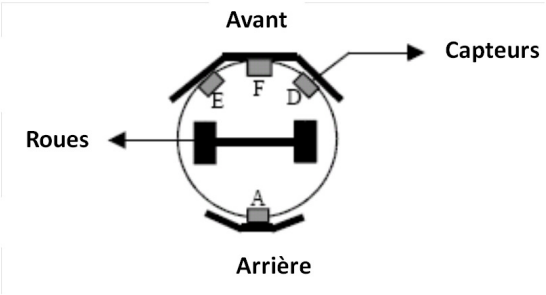


Figure 1 - Robot (vue du dessus)

Figure 2.16 – ex.48

E et F. Déterminer les entrées/sorties du système, établir la table de vérité ainsi que la(les) équation(s) de la (des) sortie(s). Implémenter la sortie (ou une des sorties) à l'aide d'un multiplexeur.

49. Implémenter les fonctions suivantes au moyen de 3 HA :
    a. $F_1 = A \oplus B \oplus C$
    b. $F_2 = \overline{A}BC + A\overline{B}C$
    c. $F_3 = AB\overline{C} + (\overline{A} + \overline{B})C$
    d. $F_4 = ABC$
50. Implémenter les fonctions suivantes au moyen d'un décodeur et de deux portes NAND :
    a. $F_1 = \sum m(0, 3)$
    b. $F_2 = \sum m(1, 2, 3)$
51. Montrer comment un multiplexeur à 8 entrées peut générer la fonction :
$$F = \bar{A}\bar{B}\bar{C}D + \bar{A}\bar{B}C\bar{D} + \bar{A}B\bar{C}\bar{D} + \bar{A}B\bar{C}D$$
52. Construire un additionneur/soustracteur (4 bits) à partir de 4 FA et de portes NAND. On utilisera 2 bits de contrôle (ADD=1 $\Rightarrow$ addition, SUB=1 $\Rightarrow$ soustraction).
    $ADD \neq SUB$ en tout temps. Les entrées pouvant être utilisées sont B_0 à B_3, $\overline{B_0}$ à $\overline{B_3}$, A_0 à A_3. La sortie est représentée par S_0 à S_3 et C_4.

3 Electronique séquentielle

1. Soit un verrou et un flip-flop D. Comment évolue la sortie si le système est soumis aux entrées de la figure 3.1 ?

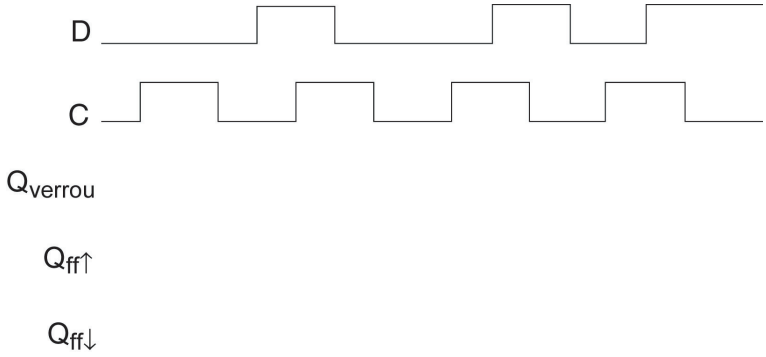


Figure 3.1 – ex.1

2. Même question pour un flip-flop JK. Q=1 au départ (figure 3.2).

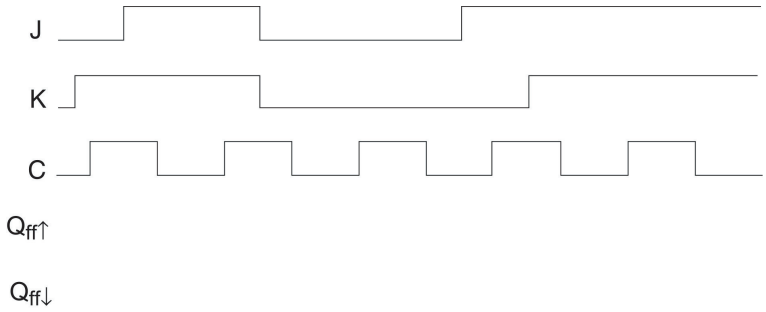


Figure 3.2 – ex.2

3. Analyser le circuit présenté à la figure 3.3.

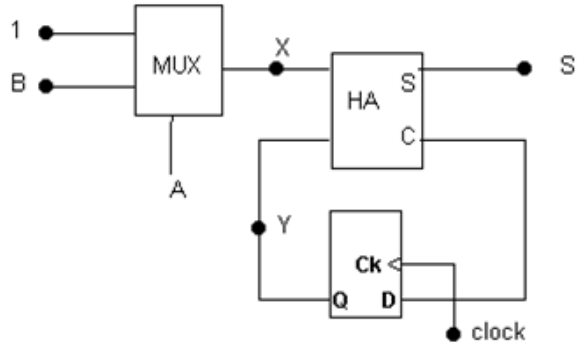


Figure 3.3 – ex.3

4. Analyser le circuit présenté à la figure 3.4.

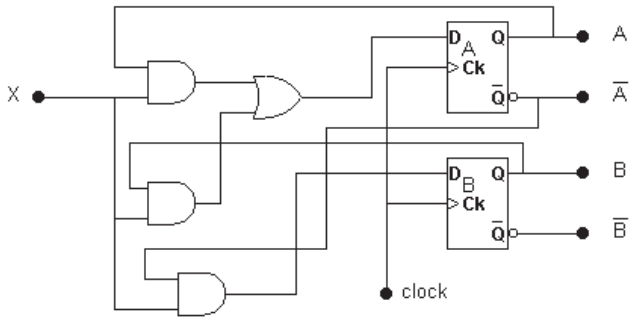


Figure 3.4 – ex.4

5. Soit le circuit synchrone défini comme suit :

$$D_A = \overline{X}Y + XA$$
$$D_B = \overline{X}B + XA$$
$$Sortie = Z = B$$

   a. Dessiner la table d'état.
   b. Dessiner le diagramme d'état.

6. Analyser le circuit présenté à la figure 3.5.

7. Analyser le circuit présenté à la figure 3.6.
8. Analyser le circuit présenté à la figure 3.7.
9. Déterminer les états du circuit présenté à la figure 3.8 pour toutes les possibilités de $Q1$, $Q0$.

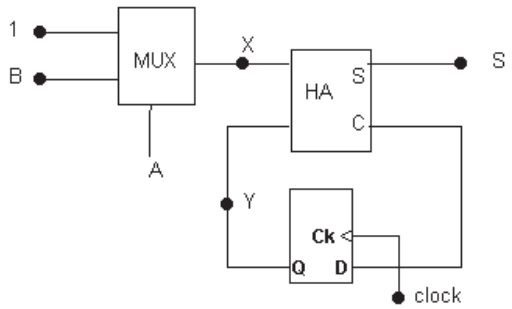


Figure 3.5 – ex.6

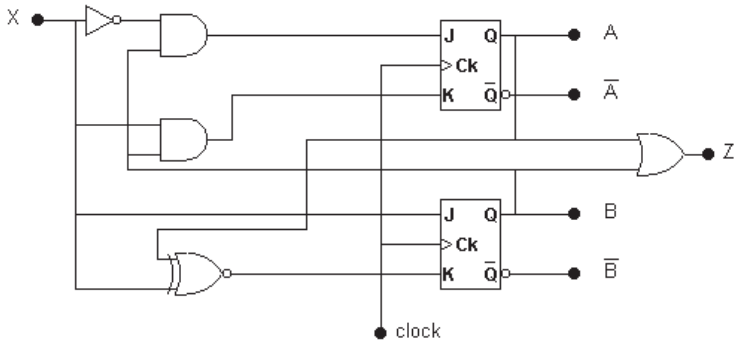


Figure 3.6 – ex.7

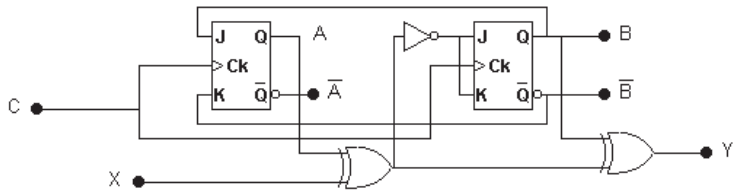


Figure 3.7 – ex.8

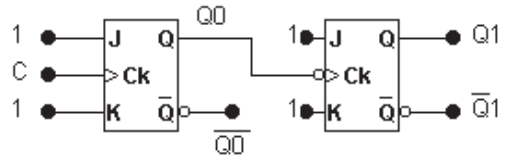


Figure 3.8 – ex.9

10. Déterminer les états pour le système à 4 F-F présenté à la figure 3.9.
11. Soit le système asynchrone de la figure 3.10. Déterminer les états suivants de tous les FF $Q3$, $Q2$, $Q1$, $Q0$ possibles.

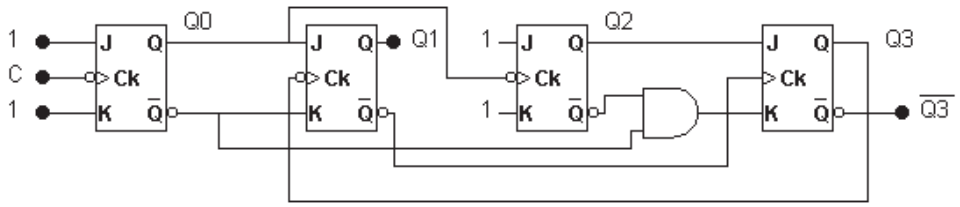


Figure 3.9 – ex.10

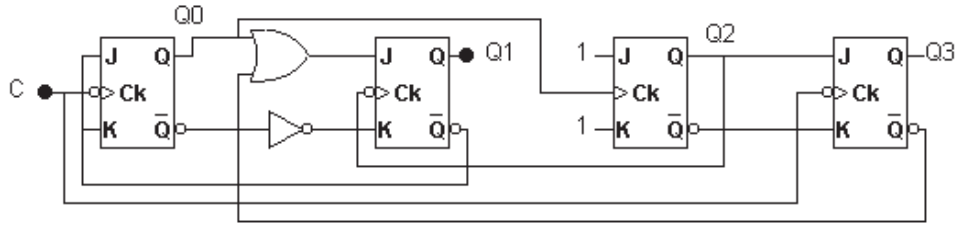


Figure 3.10 – ex.11

12. Soit le système illustré à la figure 3.11. Déterminer les états suivants de tous les F-F A, B, C, D.

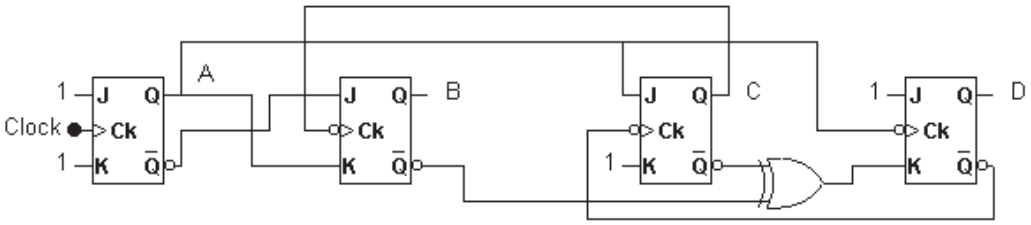


Figure 3.11 – ex.12

13. Analyser le circuit présenté à la figure 3.12.

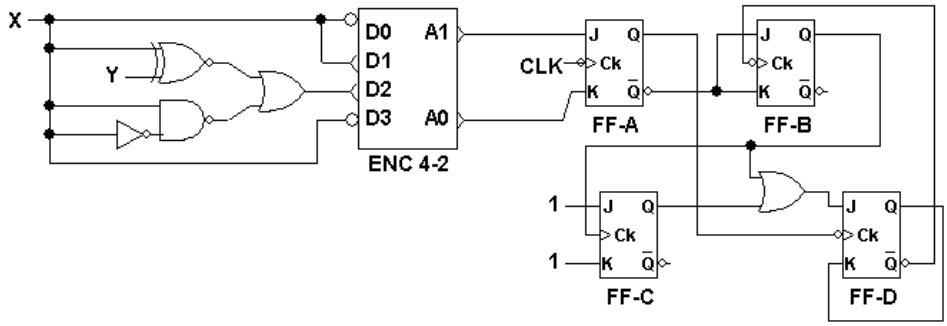


Figure 3.12 – ex.13

14. Analyser le circuit présenté à la figure 3.13.

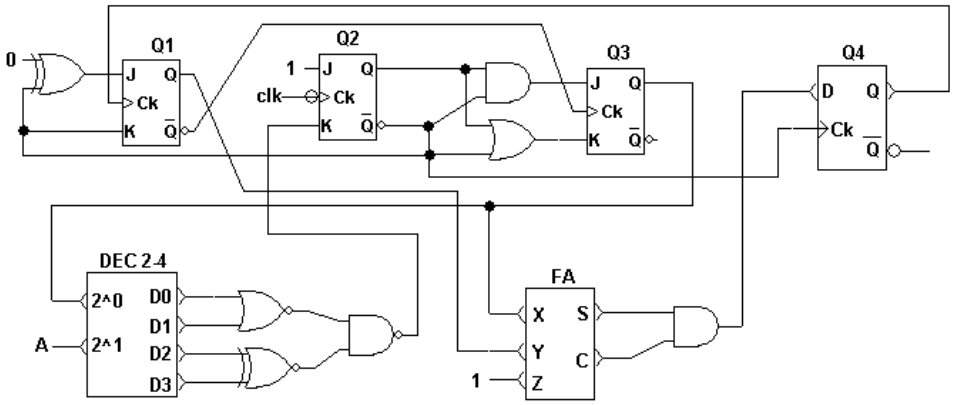


Figure 3.13 – ex.14

15. Analyser le circuit présenté à la figure 3.14.

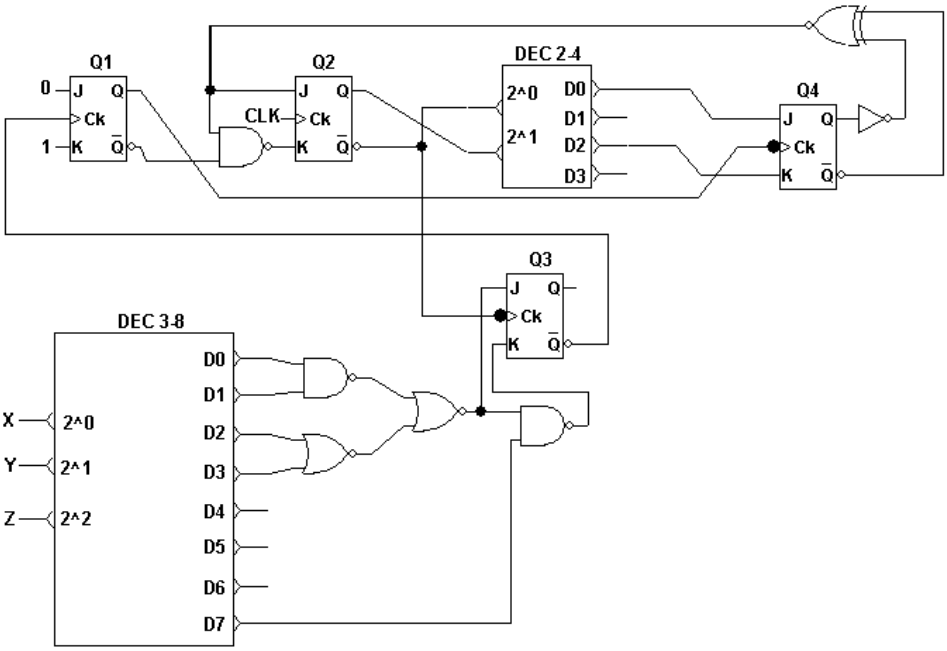


Figure 3.14 – ex.15

16. Réaliser le design d'un circuit qui a le comportement suivant :
    — l'entrée $= 1 \Rightarrow$ le circuit reste dans le même état.
    — l'entrée $= 0 \Rightarrow$ le circuit génère la séquence $00 \rightarrow 11 \rightarrow 01 \rightarrow 10 \rightarrow 00$ ...
    Utiliser des flips-flops D, la sortie est égale à l'état présent.

17. Réaliser le design d'un circuit dont les sorties génèrent la séquence suivante : $0, 1, 3, 7, 6, 4, 0, \ldots$
    On associe les sorties aux états.

18. Réaliser le design d'un circuit qui possède 2 entrées et qui a le comportement suivant :
    — si $E = 0$, le circuit reste dans le même état sans regarder la valeur de X ;
    — si $E = 1$ et $X = 1$, le circuit passe dans les états suivants : $00, 01, 10, 11, 00, \ldots$;
    — si $E = 1$ et $X = 0$, le circuit passe dans les états : $00, 11, 10, 01, 00, \ldots$
    Utiliser des flips-flops JK. Associer les sorties aux états.

19. Faire le design du circuit présenté à la figure 3.15. On associe les états aux sorties.

20. Des nombres compris entre 0 et 7 sont transmis sous forme binaire sur une ligne sérielle. L'envoi de chaque nombre commence toujours par le bit de poids fort. On demande de réaliser un circuit dont la sortie $z = 1$ lors de la détection du bit de poids faible d'un 0 ou d'un 7. Une nouvelle détection commencera après le passage de 3 bits. Déterminer la table d'état.

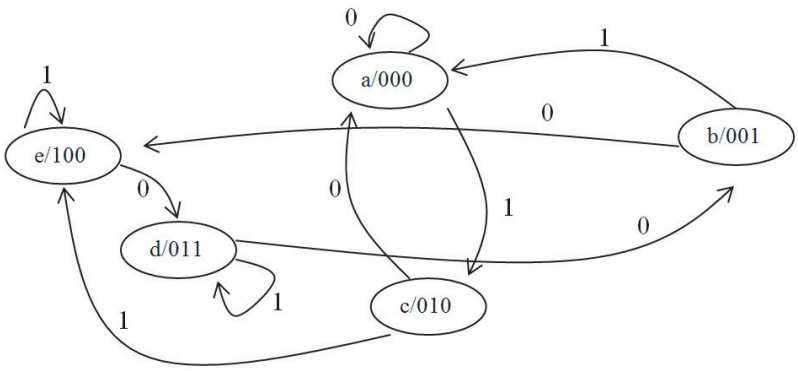


Figure 3.15 – ex.19

21. On vous demande de réaliser la synthèse d'un système séquentiel régi par une horloge à 0.05 Hz et dédié à la surveillance de la température des lignes haute tension.

    Tant qu'il y a du vent (vitesse $\geq$ à 1 m/s) et que la valeur du courant reste inférieure ou égale à 1200A, il n'y a rien à craindre.

    Lorsque la vitesse du vent tombe à moins d'1m/s et que la valeur du courant reste inférieure ou égale à 1200A, le système passe en mode de surveillance et informe l'utilisateur par un témoin lumineux. De même, si le courant dépasse 1200A et que la vitesse du vent reste supérieure ou égale à 1m/s. Ce mode reste actif tant que l'une de ces deux conditions est vérifiée.

    L'état d'alerte doit être enclenché si la vitesse du vent chute à moins d'1m/s et que la valeur du courant est supérieure à 1200A pendant 40s ou plus. L'alerte est donnée à l'utilisateur au moyen d'une alarme. On maintient l'alerte tant que la valeur du courant reste supérieure à 1200A.

    Donner le nombre de bit(s) d'entrée(s)/sortie(s) et leur signification. Etablir la machine d'états, les équations des FF-D et des sorties de ce système.

22. Afin de connaitre le sens de rotation d'un arbre tournant, deux photodiodes (R1 et R2) sont utilisées. Pour ce faire, on regarde l'ordre dans lequel elles reçoivent un pulse de lumière. Par exemple, si un pulse est observé par R1, puis par R2, l'arbre tourne dans le sens trigonométrique, et vice-versa.

    On vous demande de décrire le système qui détermine le sens de rotation à partir des signaux des 2 diodes et envoie cette information à un autre système.

    Pour chaque diode, on représente l'observation d'un pulse de lumière par un " 1 " logique et l'absence de pulse par un " 0 " logique. Le système ne doit renvoyer le sens de rotation qu'après avoir observé le pulse sur la deuxième diode, et un signal d'attente le reste du temps.

    Il arrive qu'on observe un pulse sur une diode qui ne soit pas suivi d'un pulse sur l'autre diode ; cela correspond à une détection erronée qui ne doit pas influencer la prise de décision. La disposition des éléments fait que les deux photodiodes n'observeront jamais de pulse en même temps.

    On demande de déterminer les entrées-sorties du système, le diagramme d'états et la table d'états.

23. On vous demande de réaliser le design du système séquentiel synchrone suivant :
    — L'appui sur le bouton "start" démarre un compteur 2 bits (de 0 à 3). Deux LEDs permettent de visualiser la valeur du compteur (2 bits). Tout appui ultérieur de ce bouton remet le compteur à 0.
    — L'appui du second bouton d'entrée décrémente d'une unité la valeur du compteur (jusque 0, pas de boucle telle que ...1 $\rightarrow$ 0 $\rightarrow$ 3...), pour autant que celui-ci soit en fonctionnement.
    — L'appui simultané sur les 2 boutons place le système en attente du bouton "start", le compteur affiche la valeur 2.

    Donner le nombre d'entrées/sorties du système ainsi que leur signification. Etablir la machine d'états du système. On suppose que les flip-flops sont déclenchés sur un flanc montant de l'horloge, que les boutons sont anti-rebonds et qu'ils fournissent un '1' logique lorsqu'on appuie dessus.

24. Dessiner la table d'état du système décrit dans la lettre suivante et proposer une solution aux problèmes mentionnés.

    Monsieur,

    La voiture que je vous ai récemment achetée à grand frais ne donne guère satisfaction. Le système électrique est complètement déficient et provoque 2 types de problèmes : le moteur broute et le klaxon retentit intempestivement.

    Chaque minute, chaque problème est présent ou absent. Ce que chacun d'eux fera au cours de la prochaine minute dépend de ce qui s'est passé dans la minute précédente.

    — *Le moteur fonctionnera de la même façon que dans la présente minute sauf si pendant cette minute j'ai allumé les phares et fait fonctionner les essuie-glace, auquel cas, le moteur changera de mode de fonctionnement.*
    — *Quant au klaxon, si les essuie-glace ne fonctionnent pas et si les phares sont éteints, il retentira ou non selon que le moteur broutait ou pas. Par contre, dans tous les autres cas, le klaxon retentira si le moteur ne broutait pas et inversément.*

25. Soit un circuit qui détermine le nombre de 1 contenu dans un mots de 4 bits. Les données arrivent de manière sérielle. La sortie du circuit passe à 1 si le nombre total de 1 compris dans le mot est pair. Après chaque mot, le circuit est supposé retrouver son état initial.

4 Langage VHDL

4.1 logique combinatoire

1. Implémenter et simuler le code
   — d'un Half-Adder (achitecture "flot de données", voir cours VHDL).
   — d'un Half-Adder mais de 2 autres manières (achitectures "structurée" et "procédurale", voir cours VHDL).
   — d'un Full-Adder (voir cours VHDL).
   — d'un dual MUX 4-1 (un double multiplexeur dont leur sortie dépend des mêmes entrées de sélection, comme le composant 74HC153).
   — ...

2. **Question interro de logique combinatoire 25-10-2007**

   Compléter le code VHDL et la simulation (figure 4.1). Ce programme doit implémenter la fonction $F(X, Y, Z) = \sum m(0, 2, 7) + \sum d(1, 6)$.
   Remarque : la simulation doit illustrer la table de vérité de la fonction programmée en VHDL.

```
-- librairies
Library ieee ;


-- entité
entity myFunction is


end myFunction ;

-- architecture
architecture arch_myFunction of myFunction is

begin
  process
```

```
begin





end process;
```

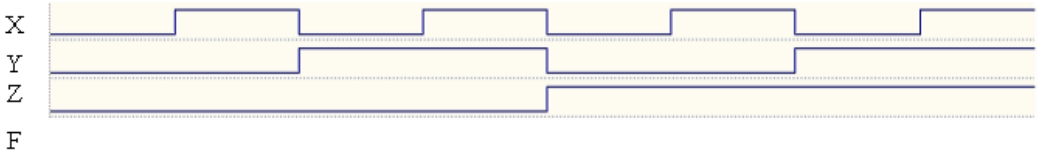


Figure 4.1 – Simulation de la fonction F.

3. **Question interro de logique combinatoire 31-10-2007**

Soit une fonction F implémentée par un décodeur (figure 4.2) et une partie de la description VHDL de cette fonction, complétez le code.

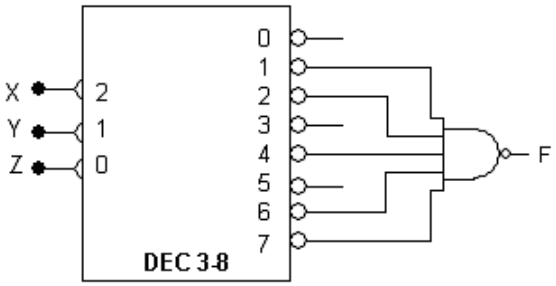


Figure 4.2 – Implémentation de la fonction F à l'aide d'une décodeur.

```
-- librairie
library ieee ;
use ieee.std_logic_1164.all ;

-- entité : entrées-sorties du DEC3-8
entity DEC3_8 is
port (X,Y,Z : in std_logic ;
 M     : out std_logic_vector(7 downto 0)) ;
end DEC3_8 ;
```

```
-- architecture du DEC 3-8
architecture arch_DEC3_8 of DEC3_8 is

signal INPUT : std_logic_vector(2 downto 0)

begin


process
  begin
    case INPUT is
      "000" => M <= "11111110" ;
      "001" => M <= "11111101" ;
      "010" => M <= "11111011" ;
      "011" => M <= "11110111" ;
      "100" => M <= "11101111" ;
      "101" => M <= "11011111" ;
      "110" => M <= "10111111" ;


end case ;
end process ;
end arch_DEC3_8 ;


-- librairie
library ieee ;


-- entité : entrées-sorties de la fonction F
entity       is



end myF ;





-- architecture de la fonction F
architecture arch_myF of myF is

component
```

```
signal minterm : std_logic_vector(7 downto 0) ;

begin
process(X,Y,Z,minterm)
  begin


F <= not(minterm(1) and minterm(2) and
) ;
```

4. **Question interro de logique combinatoire 29-10-2008**

a) complétez le code VHDL ci-dessous d'une fonction combinatoire *compl2* prenant un nombre *I* de 4 bits en entrée et donnant son complément à 2 en sortie *O*.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity compl2 is
port(
  I : in  std_logic_vector(3 downto 0);
  O : out std_logic_vector(3 downto 0));
end compl2;

architecture arch_compl2 of compl2 is

begin
..
..
..
  when "0000" => O <= "0000";
  when "0001" => O <= "1111";
  when "0010" => O <= "1110";
  when "0011" => O <= "1101";
  when "0100" => O <= "1100";
  when "0101" => O <= "1011";
  when "0110" => O <= "1010";
  when "0111" => O <= "1001";
  when "1000" => O <= "1000";
  when "1001" => O <= "0111";
  when "1010" => O <= "0110";
  when "1011" => O <= "0101";
  when "1100" => O <= "0100";
```

```
  when "1101" => O <= "0011";
  when "1110" => O <= "0010";
..
..
..
end process;
end arch_compl2 ;
```

b) complétez le code VHDL d'une fonction combinatoire prenant 2 nombres de 4 bits en entrée, A et B, et donnant le résultat de la soustraction des 2 entrées à l'aide de 5 bits : le signe S et la valeur absolue du résultat $|A - B|$. **L'opération de soustraction s'effectue à l'aide du complément à 2** (point a.), comme vu à la première séance d'exercice.

Suggestion : Complétez le schéma qui illustre les étapes de cette opération arithmétique. Par exemple, le test $A \geq B$ renvoie un 1 ou 0 logique qui est inversé pour donner la sortie S, tel que décrit dans le code VHDL.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity sub is port(
A,B : in  std_logic_vector(3 downto 0); -- opérandes
O   : out std_logic_vector(3 downto 0); -- valeur |A-B|
S   : out std_logic);  -- signe S=1=>O<0
                       -- signe S=0=>O>=0
end sub;

architecture arch_sub of sub is -- effectue A-B

..
..
..
..

signal ..

begin
..
..
..
..
begin
if (A >= B) then
     S <= '0';
..
else
```

```
  S <= '1';
..
end if;
end process;
end arch_sub;
```

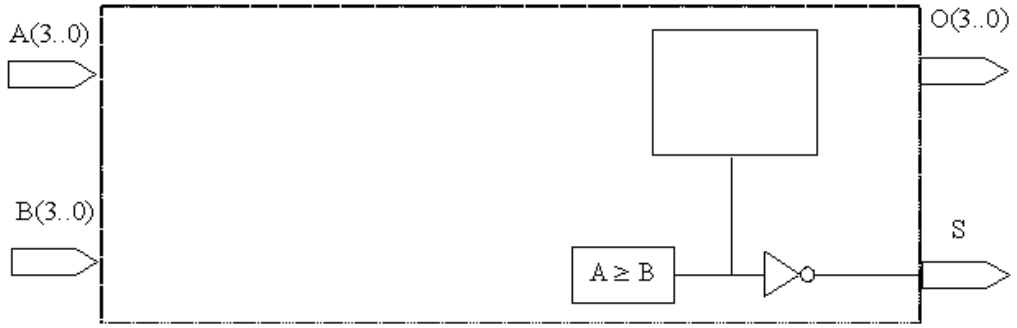


Figure 4.3 – Schéma de la soustraction ($A - B$).

5. **Question interro de logique combinatoire 18-11-2009**

   Complétez le code VHDL ci-dessous qui décrit le schéma de la figure 5.

```
-- description du MUX 4-1
library ieee ;
use ieee.std_logic_1164.all ;
-- entité
entity MUX_4_1 is port(
A,B,C,D,S0,S1 : in std_logic;
F : out std_logic);
end MUX_4_1;
-- architecture
architecture arch_MUX_4_1 of MUX_4_1 is
-- déclarations des composants et signaux internes
signal SEL : std_logic_vector(1 downto 0);
begin
  SEL <= ;
  process

    case SEL is
when "00" => F <= A;
when "01" => F <= B;
when "10" => F <= C;
when "11" => F <= D;
```

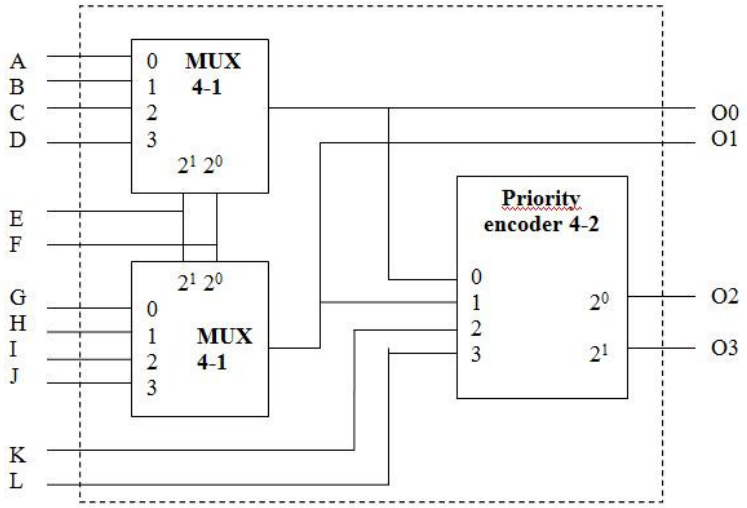


Figure 4.4 – VHDL 2009

```
    end case;
  end process;
end arch_MUX_4_1;

-- description de la function de la figure 1
library ieee;
use ieee.std_logic_1164.all;
-- entité
entity I1 is port(


);
end I1;
-- architecture
architecture arch_I1 of I1 is





begin
```

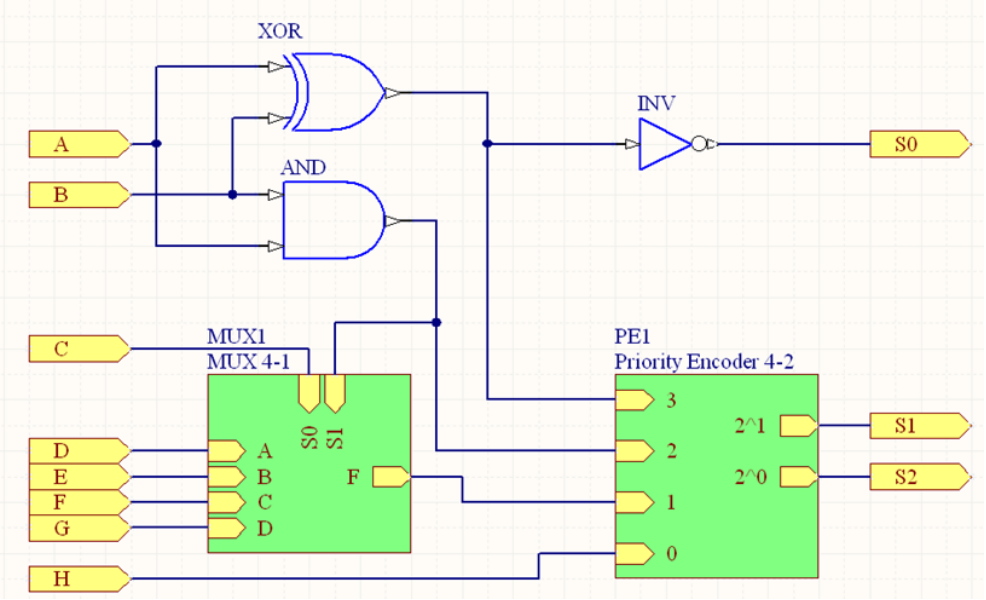


Figure 4.5 – VHDL 2010

```
process( ) -- encodeur à prorité
begin
if
```

```
end if;
end process;
end arch_I1;
```

6. **Question interro de logique combinatoire 10-11-2010**

   Complétez le code VHDL ci-dessous qui décrit le schéma de la figure 6

```
-- description du MUX 4-1
library ieee ;
use ieee.std_logic_1164.all ;
-- entité
entity MUX_4_1 is port(
A,B,C,D,S0,S1 : in std_logic;
F : out std_logic);
end MUX_4_1;
```

```
-- architecture
architecture arch_MUX_4_1 of MUX_4_1 is
-- déclarations des composants et signaux internes
signal SEL : std_logic_vector(1 downto 0);
begin
  SEL <= . . . & . . . ; -- concaténation
  Process . . .
    . . .
    case SEL is
when "00" => F <= A;
when "01" => F <= B;
when "10" => F <= C;
when "11" => F <= D;
. . .
    end case;
  end process;
end arch_MUX_4_1;

-- description de la function de la figure 1
library ieee;
use ieee.std_logic_1164.all;
-- entité
entity I1 is port(
. . .
. . .
);
end I1;
-- architecture
architecture arch_I1 of I1 is

component . . .
. . .
. . .
end component;

. . .

begin

. . .

. . .

. . .

. . .
```

```
process( . . . . . .) -- encodeur à prorité
begin
if . . .

. . .

. . .

. . .

else . . .
end if;
end process;
end arch_I1;
```

4.2 logique séquentielle

1. Implémenter et simuler le code
   — d'un compteur 4 bits avec reset synchrone.
   — d'un compteur 4 bits avec reset **a**synchrone comptant de 0 à 11.
   — d'un compteur-décompteur 4 bits avec reset synchrone comptant de 0 à 13 puis décomptant jusque 0, une seule fois après reset.
   — d'un compteur-décompteur 4 bits avec reset asynchrone comptant de 0 à 11 puis décomptant jusque 0, et ce indéfiniment.
   — d'un compteur-décompteur 4 bits avec reset synchrone comptant de 0 à 13 puis décomptant jusque 3, et ce indéfiniment.
   — d'un compteur 4 bits avec reset synchrone et possibilité de charger une valeur.
   — d'un compteur 4 bits dont les sorties sont utilisées comme table de vérité d'une fonction de votre choix.
   — d'un registre à décalage 3-bits, chargement sériel, sortie sérielle.
   — d'un registre à décalage 4-bits, chargement parallèle, sortie sérielle.
   — d'un registre à décalage 3-bits, chargement sériel, sortie parallèle.
   — d'un registre à décalage 4-bits, chargement parallèle, sortie parallèle.
   — ...
2. **Question interro de logique séquentielle 13-12-2006**

   Donner le code VHDL du système séquentiel synchrone possédant une horloge cadencée à 1Hz, deux boutons d'entrées (B1 et B2, supposés sans rebond) ainsi que 2 sorties (LED rouge et LED jaune). Ce système se comporte comme suit :
   bullet Dans l'état initial, les LEDs sont éteintes.
   bullet Le système attend l'appui successif des deux boutons. Si le premier bouton enclenché était B1, la LED rouge est allumée en premier suivie de la jaune, et inversément si le premier bouton était B2. Si l'ordre des boutons d'entrées n'est pas respecté ou que l'utilisateur oublie d'appuyer sur un bouton, le système retourne à son état initial.
   bullet Après cette intervention humaine, les sorties clignotent en alternance et à la moitié de la fréquence de l'horloge. Seul l'appui simultané des deux boutons permet de sortir de cette séquence et de réinitialiser le système par la même occasion.
   Donner la machine d'état simplifiée (clair, SVP !) et le code VHDL complet. On suppose que les flip-flops sont déclenchés sur un flanc montant de l'horloge.
3. **Question interro de logique séquentielle 12-12-2007**

   Donner le code VHDL du système séquentiel synchrone suivant :
   bullets L'appui sur le bouton *start* démarre un compteur 2 bits (de 0 à 3). Trois LEDs permettent de visualiser la valeur du compteur (2 bits) et l'activation du système (1 bit). Cette dernière LED est allumée une fois le bouton *start* appuyé (système ON). Tout appui ultérieur de ce bouton remet le compteur à 0.
   bullets L'appui du second bouton d'entrée décrémente (jusque 0, pas de boucle telle que ...1 0 3 ...) la valeur du compteur, pour autant que celui-ci soit en fonctionnement.
   bullets L'appui simultané sur les 2 boutons place le système en attente du bouton *start*, le compteur est remis à zéro et la LED témoin est éteinte (système OFF).

Donner la machine d'états simplifiée (clair, SVP !) et le code VHDL complet (limiter la description de la machine d'états aux 2 premiers états, suivie de "etc. . ."! !). On suppose que les flip-flops sont déclenchés sur un flanc montant de l'horloge, que les boutons sont anti-rebonds et qu'ils fournissent un $'1'$ logique lorsqu'on appuie dessus.

4. **Question interro de logique séquentielle 10-12-2008**

Soit un circuit électronique composé d'un composant mémoire, contenant des mots hexadécimaux, une mémoire à accès sériel ainsi qu'une PLD d'interfaçage entre les 2 composants mémoire (Fig. 4.6). Nous ne nous soucierons que du processus d'écriture vers la mémoire à accès sériel, le processus de lecture étant fort analogue. Initialement, la PLD est en attente d'une demande d'écriture (ligne à $'Z'$, $'0'$ pour une demande d'écriture, $'1'$ pour une lecture). Une fois la demande reconnue, la PLD applique un niveau bas sur la ligne de la mémoire (Chip Select, $\overline{CS} =' 0'$) un cycle d'horloge avant de démarrer le processus d'écriture sérielle. La ligne passe de $'Z'$ à $'0'$ et le bit de poids faible est transmis sur la ligne sérielle DIO. La mémoire cible est sensible aux flancs montants du signal d'horloge appliqué. De plus, elle envoie un accusé de réception ACK pour signaler à la PLD qu'elle a bien écrit le bit (cette dernière peut donc poursuivre le processus en transmettant le bit suivant). Cet ACK est complémenté par la mémoire à chaque écriture d'un bit. Une fois le dernier bit correctement transmis (le bit de poids fort), la ligne est mise à $'1'$. La Fig. **??** illustre les conditions initiales lors d'un reset ainsi qu'une erreur au temps 650 ns où le bit de poids fort est transmis de nouveau, les autres signaux étant masqués.

Donnez la machine d'états qui sera implémentée dans la PLD. Complétez l'architecture de la description VHDL. L'entité de la PLD vous est donnée.

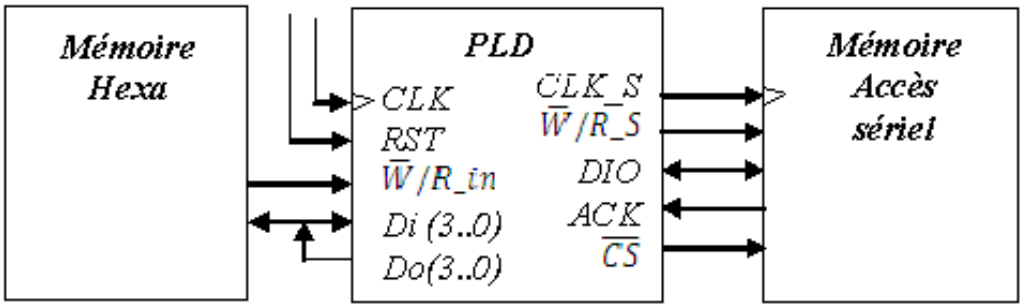


Figure 4.6 – Circuit électronique composé d'une mémoire à accès parallèle de mots hexadécimaux, d'une mémoire à accès sériel, ainsi que d'une PLD d'interfaçage.

Code VHDL à compléter :

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

entity test_serial is
 port(
     CLK : in STD_LOGIC; -- clock
     RST : in STD_LOGIC; -- asynchronous reset, active high
```

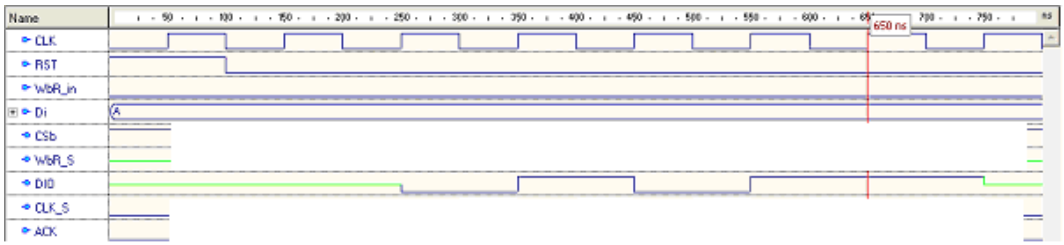


Figure 4.7 – Simulation partiellement masquée de la PLD. Conditions initiales au moment du reset asynchrone (actif haut) : WbR_S et DIO sont en haute impédance. Au temps 650ns, une erreur est apparue dans la mémoire sérielle, le bit de poids fort a été transmis de nouveau.

```
    WbR_in : in STD_LOGIC; -- Read or write
    Di : in STD_LOGIC_VECTOR(3 downto 0); -- Hexa word
    Do : out STD_LOGIC_VECTOR(3 downto 0); -- not used here
    CLK_S : out STD_LOGIC; -- serial CLK for serial MEM
    WbR_S : out STD_LOGIC; -- read or write serial process
    DIO : inout STD_LOGIC; -- serial data IO
    ACK : in STD_LOGIC; -- ACK from serial MEM
    CSb : out STD_LOGIC -- chip select (serial MEM)
    );
end test_serial;

architecture arch_test_serial of test_serial is
-- à completer sur une feuille à part, merci
end arch_test_serial;
```

5. **Question interro de logique séquentielle 16-12-2009**

Complétez l'architecture ci-dessous décrivant le système séquentiel synchrone suivant : Une entrée START est mise à '0' provoquant le lancement d'un compteur jusqu'à une valeur N. Pendant cette période de comptage le système comptabilise le nombre de flancs montants d'une seconde entrée A. Sa sortie Z passe à '1' dès que le comptage est terminé et jusqu'à ce que START repasse à 1. Le système est initialisé lorsque START vaut '1'.

Sous-questions :
a) La fréquence d'horloge sera de 100Hz. La période de comptage sera de 1s. Combien de cycles (valeur de N) dure la période de comptage ?
b) Quel est l'intervalle de fréquence du signal d'entrée A ?
c) Quel sera le plus grand nombre de flancs montants détectés ?

Suggestion : un vecteur binaire "00001100" peut s'écrire en hexadécimal : x"0C"

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
use ieee.std_logic_arith.all;
```

```
use ieee.std_logic_unsigned.all;

entity I2 is port(
CLK, START, A : in std_logic;
Z   : out std_logic);
end I2;

architecture arch_I2 of I2 is

signal .. .. .. .. ..
signal .. .. .. .. ..
signal .. .. .. .. ..

begin
-- process comptage du temps et des flancs montants
process(.. .. .. .. .. ..)
begin
if .. .. .. .. .. .. then
..    ..    ..
..    ..    ..
..    ..    ..
elsif rising_edge( .. .. .. ) then
..    ..    ..
..    ..    ..
..    ..    ..
..    ..    ..
..    ..    ..
..    ..    ..
..    ..    ..
end if;
end process;
-- process sortie Z
process( .. .. .. .. )
begin
if .. .. .. ..  then .. .. .. .. ..
else .. .. .. .. ..
end if;
end process;
end arch_I2;
```

6. **Question interro de logique séquentielle 01-12-2010**

Soit un composant de type SPLD (small programmable logic device) utilisé comme mémoire sérielle de 4 bits, en mode esclave, régi par les règles suivantes :

— En cas de reset (asynchrone et actif-bas), la sortie sérielle est en haute impédance et le registre remis à " 0000 ";
— La sortie sérielle sera également en haute impédance si le composant n'est pas activé via son entrée de sélection CS (actif-bas également) ;
— Le mot mémorisé dans le SPLD sera transmis de manière sérielle et sera remplacé par un nouveau mot de 4 bits reçu aussi de manière sérielle. La transmission/réception dans le composant SPLD s'effectue sur flanc descendant et du bit le plus significatif au moins significatif.

Suggestion : Dissociez la sortie sérielle de la mémoire de 4 bits.
Remarques : Une transmission sérielle s'effectue bit à bit. Le mode esclave signifie que le composant fonctionne de manière passive. L'horloge est fournie par le maître et n'est donc pas un train d'impulsions sans fin ! Les données émises par le maître sont supposées valides.

Complétez le code VHDL ci-dessous :

```
-- description VHDL
library ieee;
use ieee.std_logic_1164.all;

-- entité
entity I2 is port(

 . . .

 . . .
);
end I2;
-- architecture
architecture arch_I2 of I2 is

. . .
begin

SORTIE : process(. . . . . .)
begin
if . . . . . then . . . . . .;

else                . . . . . .;
end if;
. . .
```

```
MEMOIRE : . . . . . .
begin
  if  . . .  then  -- reset asynchrone

  . . . . . .

  elsif . . . . . .  then

      . . . . . .

 . . . . . .

  else . . . . . .;
  end if;
end process;
end arch_I2;
```