Université de 20 Aout 1955 SKIKDA
Faculté de Technologie
Département de Technologie

Logique combinatoire et séquentielle

Cours

2eme Année Tronc-Commun

Science et Technologie

ST

Dr : Kared Saber

s.kared@univ-sikikda.dz
Karedsaber21@gmail.com

2024-2025

Avant-propos

Ce cours est destiné aux étudiants de deuxième année tronc-commun ST de spécialité génie électrique, électromécanique et génie industrielle.il est étudié en semestre deux, il traite le programme du module de logique combinatoire et séquentielle. Le premier chapitre traite les systèmes de numération et le codage de l'information. L'étudiant savoir définir la base d'un système de numération (binaire, octal, hexadécimale, décimal), savoir convertir de base, codes non pondères (code Gray, code Ascii), effectuer les opérations arithmétiques directement dans le système binaire naturel.

Le deuxième chapitre traite l'initiation à l'algèbre de Boole et à ses applications, consiste à connaitre les opérations logiques de base négation, ET (intersection),OU (union) .connaitre la table de vérité de chacune de ces opérations et leur symbole graphique (porte),les lois fondamentales de l'algèbre de Boole, savoir démontrer ,implanter et appliquer les relation de base de l'algèbre de Boole, connaitre le théorèmes de Morgan ,savoir utiliser la dualité de l'algèbre de Boole pour transposer une relation en une autre, la représentation des fonctions logiques (table de vérité , tables de karnaugh),et la simplification de ces fonction par la méthode algébrique et la méthode de karnaugh.

Le troisième chapitre savoir résoudre les problèmes de logique combinatoire, savoir implanter, à l'aide de circuit intègres, leur fonction solution.(multiplexeurs ,démultiplexeur ,codeur ,décodeur……..).

Le quatrième chapitre dédie à l'étude des circuits séquentiels être capable d'explique la déférence entre circuit combinatoire et circuit séquentielle, et en partant de l'élément de base des bascules, La bascule RS, La bascule D, La bascule Maitre-esclave, La bascule T, La bascule JK. Ainsi des exemples d'applications avec les bascules (analyse de fonctionnement d'une bascule) : Diviseur de fréquence par n, Générateur d'un train d'impulsions, …

Le cinquième chapitre traite les différents types de compteur et décompteurs (synchrones et asynchrones).

Enfin, nous terminons par le huitième chapitre sur les registres et nous espérons que ce cours sera utile pour les étudiants de la deuxième année licence Génie électrique, électromécanique et génie industrielle.

Liste des Tableaux

Chapitre 01 système de numération

1 Base de numération

1.1 Définition :

Un système de numération est un exemple compose d'un certain nombre, de symboles, ce nombre correspond à la base de numération.

Exemple : le système donnée par $\{\mathbf{0}, \mathbf{1} \dots \dots \dots \dots \mathbf{9}\}$ contient 10 chiffres, donc cet ensemble correspond à la bas 10 (décimal)

Ex : $\boldsymbol{M = (2789,34)_{10}}$

$\mathbf{M=2 * 10^3 + 7 * 10^2 + 8 * 10^1 + 9 * 10^0 + 3 * 10^{-1} + 4 * 10^{-2}}$

Soit une base, associes à b symboles $\{\boldsymbol{S_0}, \dots \dots \dots, \boldsymbol{S_{b-1}}\}$

Un nombre M s'écrit dans cette base selon la règle suivant :

$M = (a_n , \dots \dots . a_0, a_{-1}, \dots \dots \dots , a_{-m})_b$

$M = a_n b^n + \cdots + a_0 b^0 + a_{-1} b^{-1} + \cdots + a_{-m} b^{-m}$

$M = \sum_{i=-m}^{i=n} a_i b^i$ Cette forme est appelle : forme polynomiale

a_i : Coefficient

b : La base

1.2 Les différents systèmes de numérations

Les systèmes de numération les plus fréquentent utilises sont :(D, B, O, Hexa).

D ⟶ Système décimal

B ⟶ Système binaire

O ⟶ Système octal

Hexa ⟶ Système hexadécimal

1.3 Le système binaire ou base 2 :

Dans ce système, il comprend 2 symboles (0,1), donc 2 digits 0 et 1 appelé dans ce cas " BIT" (Binary digit) .Ce système est à la base du langage machine (c'est le seul langage compréhensible par l'ordinateur)

Par exemple, le nombre 1011 exprimé en binaire signifie:

$(1011)_2 = 1 * 2^0 + 1 * 2^1 + 0 * 2^2 + 1 * 2^3$

1.4 Système octal ou base 8:

Dans ce système, la base vaut 8 et il y a 8 digits: 0, 1, 2, 3, 4, 5, 6 et 7. Il n'y a pas de chiffres 8 et 9.

Par exemple: le nombre 275 et 174,4 exprimé en octal:

$(275)_8 = 5 * 8^0 + 7 * 8^1 + 2 * 8^2$

$(174{,}4)_8 = 4 * 8^0 + 7 * 8^1 + 1 * 8^2 + 4 * 8^{-1}$

1.5 Système hexadécimal ou base 16:

Dans ce système, la base vaut 16 et il y a 16 digits: 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E et F. Les dix premiers digits de 0 à 9 sont les chiffres du système décimal et les digits de 10 à 15 sont les premières lettres majuscules de l'alphabet.

$A = 10; B = 11; C = 12; D = 13; E = 14; F = 15$

Exemple, le nombre BAC exprimé en hexadécimal :

$(BAC)_{16} = C * 16^0 + A * 16^1 + B * 16^2 = (2988)_{10}$

$(2AF)_{16} =$

1.6 Système décimal ou base 10:

Dans la base 10 "système décimal ", il y a dix digits: 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9 appelés chiffre , c'est système le plus connu et le plus utilise par l`homme

$(1234)_{10} = 4 * 10^0 + 3 * 10^1 + 2 * 10^2 + 1 * 10^3 = 4 + 30 + 200 + 1000$

Conversation : Décimal, Binaire, Octal, Hexadécimal :

Binaire	Octal	Décimal	Hexadécimal
$0000_{(2)}$	$0_{(8)}$	0	$0_{(16)}$
$0001_{(2)}$	$1_{(8)}$	1	$1_{(16)}$
$0010_{(2)}$	$2_{(8)}$	2	$2_{(16)}$
$0011_{(2)}$	$3_{(8)}$	3	$3_{(16)}$
$0100_{(2)}$	$4_{(8)}$	4	$4_{(16)}$
$0101_{(2)}$	$5_{(8)}$	5	$5_{(16)}$
$0110_{(2)}$	$6_{(8)}$	6	$6_{(16)}$
$0111_{(2)}$	$7_{(8)}$	7	$7_{(16)}$
$1000_{(2)}$	$10_{(8)}$	8	$8_{(16)}$
$1001_{(2)}$	$11_{(8)}$	9	$9_{(16)}$
$1010_{(2)}$	$12_{(8)}$	10	$A_{(16)}$
$1011_{(2)}$	$13_{(8)}$	11	$B_{(16)}$
$1100_{(2)}$	$14_{(8)}$	12	$C_{(16)}$
$1101_{(2)}$	$15_{(8)}$	13	$D_{(16)}$
$1110_{(2)}$	$16_{(8)}$	14	$E_{(16)}$
$1111_{(2)}$	$17_{(8)}$	15	$F_{(16)}$

Tableau1.1 : Conversation des systèmes

2 Conversation entre les systèmes de numération

2.1 Passage d`un système (p) an système décimal

On exploite le développement polynomial pour assurer ce passage.

$M = a_n b^n + \cdots + a_0 b^0$

A. Conversion binaire –décimal

$M = (1001)_2 = (\ldots\ldots)_{10}$

$M = (1001)_2 = 1 * 2^3 + 0 * 2^2 + 0 * 2^1 + 1 * 2^0 = (9)_{10}$

B. Conversion octal- décimal

$M = (207)_8 = (\ldots\ldots)_{10}$

$M = (207)_8 = 2 * 8^2 + 0 * 8^1 + 7 * 8^0 = 128 + 7 = (135)_{10}$

C. Conversion hexadécimal - décimal

$M = (76)_{16} = (\ldots\ldots)_{10}$

$M = (76)_{16} = 7 * 16^1 + 6 * 16^0 = 112 + 6 = (118)_{10}$

2.2 Passage d`un système décimal an système (p)

Soit :

$N = (Q_n \ldots\ldots\ldots\ldots Q_0, Q_{-1} \ldots\ldots\ldots Q_{-m}) = (Q_n \ldots\ldots Q_0)_p + (0, Q_{-1}\ldots\ldots Q_{-m})_p$

$(Q_n \ldots\ldots\ldots\ldots Q_0)$: Partie entier n chiffres

$(0, Q_{-1}\ldots\ldots Q_{-m})$: Partie fractionnaire m chiffres

P : base

Pour assurer cette conversion on convertit la PE(N) et la PF(N).

- **PE(N)** :

L`algorithme de conversion et le suivant :

1. Diviser le nombre décimal par la base (P).
2. Prendre le résultat de la division, et le diviser de nouveau par la base (P).
3. Reprendre l'étape 2 jusqu' a obtenu un résultat nul.
4. Le nouveau nombre équivalant s`obtenu en prenant les restes de la division de la dernière a la première, en écrivons de gauche à droite

- **PF(N) :**

1. Multiplier le nombre décimal fractionnaire par la base (P)
2. Reprendre la partie fractionnaire du résultat et la multiplier de nouveau.
3. Reprendre l'étape 2 jusqu`a obtenu une partie fractionnaire nul
4. Le nouveau nombre s`obtient : on prend les partie entières des résultats de multiplication

2.2.1 Conversion d'un nombre en base 2

1- PE(N)=$(26)_{10} = (?)_2$

26 :2=13 et reste 0

13 :2=6 et reste 1

06 :2=3 et reste 0 Donc $(26)_{10} = (11010)_2$

03 :2=1 et reste 1

01 :2=0 et reste 1

Condition d'arrêt

2-PF(N)=$(0.75)_{10} = (?)_2 = (0, Q_{-1} \dots . Q_{-m})_2$

0.75*2=1.5

0.5*2=1

$(0.75)_{10} = (0,11)_2$

Donc si $N = (26.75)_{10}$alors il aura comme valeur $N = (11010,11)_2$

Exemple : I- $(57)_{10} = (?)_2$

$(57)_{10} = (111001)_2$

2.2.2 Conversion d'un nombre en base 8

1-PE(N) = $(207)_{10} = (?)_8$

$207 : 8 = 25$ et reste 7

$25 : 8 = 3$ et reste 1 Donc : $(207)_{10} = (317)_8$

$3 : 8 = 0$ et reste 3

2-PF(N) = $(0.8125)_{10} = (?)_8$

$0.8125 * 8 = 6.5$ Partie entier 6

$0.5 * 8 = 4.0$ Partie entier 4

D`ou $(0.8125)_{10} = (0,64)_8$

2.2.3 Conversion d'un nombre en base 16

$(637)_{10} = (?)_{16}$

637 :16=39 et reste 13 (13 représenté par D en hexadécimal)

39 :16= 2 et reste 7 $(637)_{10} = (27D)_{16}$

2 :16=0 et reste 2

Exemple : $(0.6875)_{10} = (?)_2$

$0.6875 * 2 = 1.375$ Partie entier 1

$0.375 * 2 = 0.75$ Partie entier 0

$0.75 * 2 = 1.5$ Partie entier 1

$0.5 * 2 = 1.0$ Partie entier 1 D`ou $(0.6875)_{10} = (0{,}1011)_2$

2.3 Passage d`un système (P) a un autre $(P)'$

Pour faire ce passage on doit procéder en deux étapes :

- On passe de (P) vers le décimal en appliquant l'algorithme (a)
- On passe de (décimal) vers le système $(P)'$en appliquant l'algorithme (b)

Exemple :

M=$(65)_7 = (?)_4$

Algorithme (a) : $(65)_7 = 5 * 7^0 + 6 * 7^1 = (47)_{10}$

Algorithme (b) : $(47)_{10} = (?)_4$

47 :4=11 et reste 3

11 :4=2 et reste 3 $(47)_{10} = (233)_4$

2 :4=0 et reste 2

Donc : M=$(65)_7 = (233)_4$

2.4 Passage d'un système P à un système P^k

Cette méthode est utilisée pour passer d'une base P à une base P^k, et vice versa. Le cas le plus courant est celui où il s'agit de passer d'une représentation binaire à une représentation octale ou hexadécimal, ou de procéder à l'opération inverse.

Afin de mieux illustrer le concept, considérons les exemples suivants :

$(1101001)_2 = (1\ 101\ 001)_2 = (001\ 101\ 001)_2 = (151)_8$

$(100100111)_2 = (1\ 0010\ 0111)_2 = (0001\ 0010\ 0111)_2 = (127)_{16}$

$(110111101111)_2 = (1101\ 1110\ 1111)_2 = (DEF)_{16}$

Le principe utilisé ici est de regrouper k chiffres pour passer de P à P^k. L'inverse est également applicable, comme l'illustrent les exemples suivants :

$(1EC5)_{16} = (0001\ 1110\ 1100\ 0101)_2 = (0001111011000101)_2$

$= (1111011000101)_2$

$(1672)_8 = (001\ 110\ 111\ 010)_2 = (001110111010)_2 = (1110111010)_2$

Dans ce cas, on traduit chaque chiffre en k chiffres binaires

2.5 passage d'un système P^{k1} à un système P^{k2}

On doit passer par le système P pour réaliser cette conversion :

$N = (1AC)_{16} = (\ldots\ldots)_8$

$N = (1AC)_{16} \longrightarrow (0001\ 1010\ 1100)_2 \longrightarrow (110\ 101\ 100)_2 \longrightarrow (654)_8$

3 Opération arithmétique

A- Addition Binaire

0+ 0 = 0

1+ 0 = 1

0 + 1 = 1

1+ 1 = 0 retourné 1

1+1+1=1 retenue 1

B- Soustraction Binaire

0– 0 = 0

1 – 0 = 1

0 – 1 = 1 retenue 1

1– 1 = 0

1–1–1=1 retenue 1

0–1–1=0 retenue 1

C- Multiplication Binaire

0 * 0 = 0

1* 0 = 0

0 * 1 = 0

1 * 1 = 1

D- Division Binaire

0 / 1 = 0

1 / 1 = 1

Exemple d'application

$110101 + 111100 = 1110001$

$1011 * 101 = 110111$

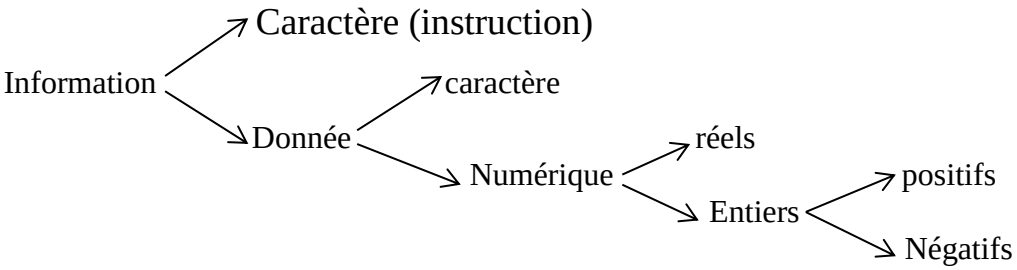
$1011 - 111 = 0110$

$10101/11 = 111$

4 Codage des informations

4.1 Introduction

Dans une machine, et existe plusieurs types d'information. Les informations sont codes selon leur type.



4.2 Codage des entiers

4.2.1 Représentation d'un entier naturel (non signé)

Un entier naturel est un entier positif ou nul, coder cet entier revient à le convertir en binaire et d'utiliser un nombre des bits suffisant pour le représenter. D'une manière générale un codage sur n bits permet de représenter 2^n nombres naturels dont la valeur est comprise entre 0 et $2^n - 1$

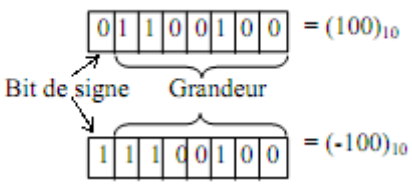
Exemple : Pour coder des nombres naturels compris entre 0 et 7, il faut utiliser 3 bits car ($2^3 = 8$ Positions).

4.2.2 Représentation d'un entier avec signe (signé)

Un entier signé est un nombre qui peut être positif ou négatif. Il faut le coder de telle façon que l'on puisse savoir s'il s'agit d'un nombre positif ou négatif, et de plus il faut conserver les règles d'addition (le nombre + son négatif = 0).

Pour coder cet entier, on réserve le bit de poids le plus fort (le bit le plus à gauche) pour le signe, il prend la valeur 0 pour le signe positif et 1 pour le signe négatif. Ce qui implique que la plus grande valeur codée avec n bit est $2^{n-1} - 1$. Par contre la plus petite valeur est $-(2^{n-1} - 1)$. Le nombre des entiers signés codés sur n bit est égal à 2^n,

Exemple : voici deux exemples de nombres écris en représentation signe-grandeur ayant la Même grandeur mais de signe opposé :



4.2.3 Complément à 1 :

Le complément à 1 est obtenu en inversant bit par bit le nombre en question, de ce fait, la somme du nombre et de son complément sera égale a $2^n - 1$.

$C1(A) = \text{not}(A)$.

Exemple :

Le complément a 1 de $(1001100)_2$ égale$(0110011)_2$

La somme : $(1001100)_2 + (0110011)_2 = (11111111)_2$

$n = 8 \Rightarrow 255 = 2^8 - 1$

4.2.4 Complément à 2 :

C'est la représentation la plus utilisée. Il est égal au complément à 1 majoré de 1.

Avec des mots de n bits, on obtient 2^n valeurs différentes, de 0 a $2^{n-1} - 1$ pour les valeurs positives, et de 1- a 2^{n-1} pour les valeurs négatives.

Exemple :

$+18$: 010010

$C1(18)$: 101101

$+1$:

$C2(18)$: 101110

Pour calculer la soustraction en complément a 2 de $(25)_2 - (18)_2$,il est équivalent à calculer $[25 + C2(18)]$.

$+25$: 011001

$+C2(18)$: 101110

$+7$: 000111 ⟹ **Résultat signé +7**

On écarte la dernière retenue

4.3 Codage BCD

Ce code conserve les avantages du système Décimal et du code binaire. Il est utilisé par les machines à calculer.

On fait correspondre à chaque caractère du système décimal un mot du code binaire de 4 bits, on a alors :

Code décimal	0	1	2	3	4	5	6	7	8	9
Code BCD	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

Remarque :

Les notions (1010, 1011, 1100, 1101, 1110, 1111) sont interdits en BCD

Exemple : conversion décimale BCD

$(19)_{10} = (00011001)_{BCD}$

4.4 Code Gray ou Binaire réfléchi

Le code Gray est construit de telle façon que le passage d'une valeur à la suivante ne nécessite la modification d'un seul bit, Tableau suivant.

Code binaire				Décimal	Code Gray			
0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	1
0	0	1	0	2	0	0	1	1
0	0	1	1	3	0	0	1	0
0	1	0	0	4	0	1	1	0
0	1	0	1	5	0	1	1	1
0	1	1	0	6	0	1	0	1
0	1	1	1	7	0	1	0	0
1	0	0	0	8	1	1	0	0
1	0	0	1	9	1	1	0	1
1	0	1	0	10	1	1	1	1
1	0	1	1	11	1	1	1	0
1	1	0	0	12	1	0	1	0
1	1	0	1	13	1	0	1	1
1	1	1	0	14	1	0	0	1
1	1	1	1	15	1	0	0	0

Tableau1.2 : La représentation en code Gray des nombres écrient sur 4 bits

La procédure de construction du tableau du code Gray pour différents nombres de bits (1, 2, 3 et 4) est illustrée sur la Figure suivante.

Remarquer du Tableau qu'entre le dernier nombre 15 =1000 et le premier nombre 0=0000 il y a un seule bit qui change donc en appliquant la définition de ce code le nombre 15 est suivi par 0 et la séquence se répète; d'où la qualification ***code cyclique.*** Cette propriété est exploitée dans les dispositifs fournissant en numérique la position angulaire d'une pièce en rotation.

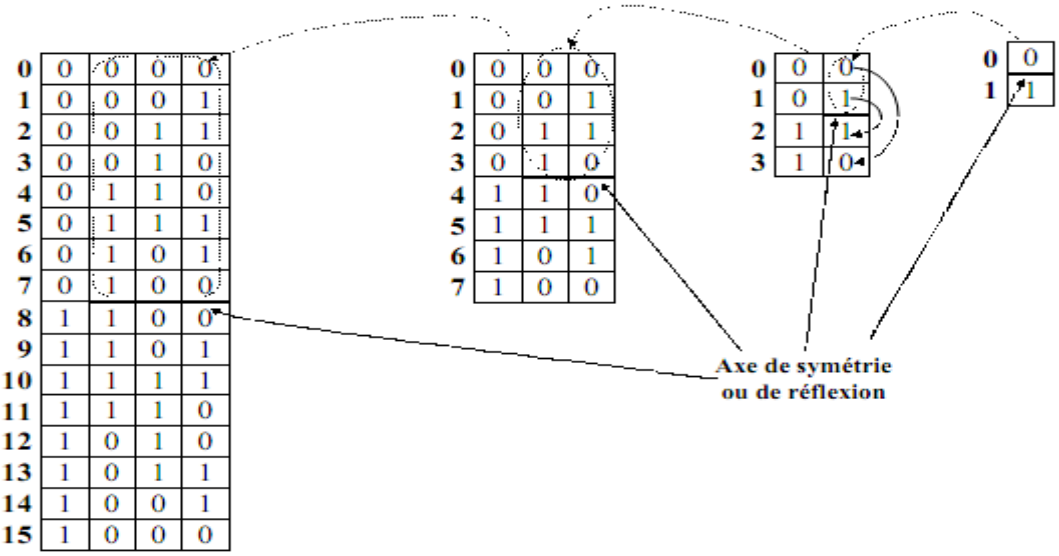


Tableau1.3 : Procédure de construction du tableau du code Gray

4.5 Conversation Binaire-Gray

1- Le MSB gray garde la même valeur du MSB binaire .

2- Sommer le MSB binaire avec le bit de degré inferieur(ne considère pas la retenue).

3- Continuer l'addition des bits adjacents jusqu'à atteindre le LSB.

4- Si le bit a_{n+1} et b_{n+1} ont la même valeur, alors $g_n = 0$.

5- Si le bit a_{n+1} est diffèrent de b_{n+1}, alors $g_n = 1$.

Exemple :

$N = (110110)_{2,\text{naturel}}$

Naturel= 1→1→0→1→1→0

Gray = 1 0 1 1 0 1

4.6 Conversation Gray-Binaire

1- On garde le même MSB.

2- On compare le bit binaire naturel de rang n+1 au bit réfléchi de rang n(on met 1s'ils sont différents et 0 si sont égaux).

Exemple :

Gray = 1 1 0 1 0

Naturel= 1 0 0 1 1

4.7 LE CODE A.S.C.I.I.

ASCII vient de « American Standard Code for Inter change of Information ». C'est un code universellement adopté qui permet de représenter sur 7 bits l'ensemble des caractères alphanumériques, des symboles et des commandes de transmission.

Pour connaître le code binaire correspondant à un symbole, il suffit de le localiser dans le tableau (voir en annexe) et de noter dans l'ordre :

• les bits de la colonne $b7, b6$ et $b5$

• puis les bits de la ligne $b4, b3, b2$ et $b1$.

Remarque Le code ASCII est aujourd'hui toujours utilisé, mais il tend à être remplacé par le standard uni code. Si on regarde bien le tableau, on s'aperçoit que la définition du code correspond bien à la langue américaine. Par contre toutes les langues ayant des accents passent à la trappe, par conséquent la nôtre.

Mais que dire du â allemand qui n'y est même pas représenté. Quant aux langues qui n'ont pas d'alphabet latin (comme l'arabe ou le chinois), il n'y a aucun moyen de les coder.

Aujourd'hui cependant le standard 'unicode' code chaque caractère sur 16 bits, ce qui laisse 65536 possibilités. Cela en laisse assez pour coder toutes les langues du mondes (ou presque) ainsi que des caractères spéciaux.

				b7 →	0	0	0	0	1	1	1	1
				b6 →	0	0	1	1	0	0	1	1
				b5 →	0	1	0	1	0	1	0	1
b4	b3	b2	b1		**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	0	0	0	**0**	NUL	DLE	SP	0	@	P		p
0	0	0	1	**1**	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	**2**	STX	DC2	"	2	B	R	b	r
0	0	1	1	**3**	ETX	DC3	#	3	C	S	c	s
0	1	0	0	**4**	EOT	DC4	$	4	D	T	d	t
0	1	0	1	**5**	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	**6**	ACK	SYN	&	6	F	V	f	v
0	1	1	1	**7**	BEL	ETB	'	7	G	W	g	w
1	0	0	0	**8**	BS	CAN	(	8	H	X	h	x
1	0	0	1	**9**	HT	EM	)	9	I	Y	i	y
1	0	1	0	**A**	LF	SUB	*	:	J	Z	j	z
1	0	1	1	**B**	VT	ESC	+	;	K	[	k	{
1	1	0	0	**C**	FF	FS	,	<	L	\	l	|
1	1	0	1	**D**	CR	GS	-	=	M	]	m	}
1	1	1	0	**E**	SO	RS	.	>	N	^	n	~
1	1	1	1	**F**	SI	US	/	?	O	_	o	DEL

NUL	Nul	**DLE**	Echappement transmission
SOH	Début d'en tête	**DC1**	Commande d'appareil
STX	Début de texte	**DC2**	Commande d'appareil
ETX	Fin de texte	**DC3**	Commande d'appareil
EOT	Fin de transmission	**DC4**	Commande d'appareil
ENQ	Interrogation	**NAK**	Accusé de réception négatif
ACK	Acquittement	**SYN**	Synchronisation
BEL	Sonnerie ou alarme	**ETB**	Fin de bloc de transmission
BS	Espacement arrière	**CAN**	Annulation
HT	Tabulation horizontale	**EM**	Fin de support
LF	Interligne	**SUB**	Substitution
VT	Tabulation verticale	**ESC**	Echappement
FF	Présentation de formule	**FS**	Séparateur de fichier
CR	Retour chariot	**GS**	Séparateur de groupe
SO	Hors code	**RS**	Séparateur d'article
SI	En code	**US**	Séparateur de sous article
DEL	Oblitération		

Tableau1.4 : les premiers caractères de la table ASCII

Exemple : le code binaire est donne par $b_7b_6b_5b_4b_3b_2b_1$,(b_1 est le bit de poids faible).
Le code de :
U est 55_{16} soit 1010101 en binaire.
F est 46_{16} soit 1000110 en binaire.
K est $4B_{16}$ soit 1001011 en binaire

TD 01
(Systèmes de numération et opérations arithmétique binaires)

Exercice 1:

1- Déterminer la valeur décimale des nombres suivants :

$N_2 = (101101)_2$,$N_2 = (1100)_2$,$N_2 = (1011101)_2$

$N_8 = (6734)_8$,$N_8 = (15)_8$,$N_8 = (275)_8$

$N_5 = (3.42)_5$,$N_{16} = (A732)_{16}$, $N_{16} = (2AF)_{16}$

2- Convertir les nombres fractionnaires suivant en décimale :

$(111100010)_2$, $(10001.11111)_2$, $(111110.000011)_2$

Exercice 2 :

1- Convertir les nombres suivant en binaire, en octal, et en hexadécimal :

$(1420)_{10}$ $(345.235)_{10}$ $(777.625)_{10}$

2- Convert the following fractional numbers to decimal:

$(1011.0011)_2$, $(122.23)_4$,$(7.7)_8$,$(4B.CC)_{16}$, $(14.82)_9$

3- Considérant les nombres suivant exprimes en octal

- Trouvez leurs équivalent en binaire puis en hexadécimal.

a-$(553)_8$ b-$(157)_8$

Exercice 3 :

Effectuer les opérations arithmétiques suivants en binaire :

a- $(11.01)_2 + (101.1)_2$

b- $(111.01)_2 + (1101.10)_2$

c- $01111 + 10010$

d- $010111 - 110010$

e - $11010 - 100.11$

f - $10110101 * 1011$

g- $110111/1011$

Exercice 4:

Convertir les nombres décimaux suivants en binaire naturel puis convertir en binaire réfléchi:

a- 25+36 b- 27 c- 21

Exercice 5:

Soit les nombres expriment en binaire naturel.

$(10101)_2$, $(11001)_2$,$(10001)_2$.

1- Donner leur équivalent en BCD

2- Convertir les nombres décimaux suivants en binaire BCD :

3- $(15)_{10} + (10)_{10}$, $(75)_{10} + (68)_{10}$, $(99)_{10} + (99)_{10}$

4- Convertir les nombres binaires suivant en code (GARY) :

$(11011)_2$,$(1001010)_2$, $(101101)_2$

1- Convertir les nombres binaires réfléchis (GRAY) en binaire naturel :

1010 , 10010 , 100111

Solution

Exercice 1

I- les valeurs décimale

$N_2 = (101101)_2 = 1 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 0 \times 2^3 + 1 \times 2^4 = (45)_{10}$

$N_2 = (1100)_2 = 0 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 1 \times 2^3 = (12)_{10}$

$N_2 = (1011101)_2 = (93)_{10}$

$N_8 = (6734)_8 = 4 \times 8^0 + 3 \times 8^1 + 7 \times 8^2 + 6 \times 8^3 = (3548)_{10}$

$N_8 = (15)_8 = 5 \times 8^0 + 1 \times 8^1 = (13)_{10}$

$N_8 = (275)_8 = (189)_{10}$

$N_5 = (3{,}42)_5 = 3 \times 5^0 + 4 \times 5^{-1} + 2 \times 5^{-2} = (3{,}88)_{10}$

$N_{16} = (A732)_{16} = 2 \times 16^0 + 3 \times 16^1 + 7 \times 16^2 + 10 \times 16^3 = (42802)_{10}$

$N_{16} = (2AF)_{16} = 2 \times 16^0 + 10 \times 16^1 + 15 \times 16^2 = (687)_{10}$

II- octal puis en hexadécimal

7 4 2

$(111100010)_2 = (742)_8 = (1E2)_{16}$

1 E 2

2 1 , 7 6

$(10001{,}11111)_2 = (21{,}76)_8 = (11, F8)_{16}$

1 1 , F 8

7 6 , 0 3

$(111110{,}000011)_2 = (76{,}03)_8 = (3E, 0C)_{16}$

3 E , 0 C

Exercice 2

1-

Décimale$()_{10}$	**Binaire $()_2$**	**Octal$()_8$**	**Hexadécimal$()_{16}$**
$(\mathbf{1420})_{10}$	$(\mathbf{10110001100})_2$	$(\mathbf{2614})_8$	$(\mathbf{158C})_{16}$
$(\mathbf{345,235})_{10}$	$(\mathbf{101011001,0011})_2$	$(\mathbf{531,1702})_8$	$(\mathbf{159,3C28})_{16}$
$(\mathbf{777,625})_{10}$	$(\mathbf{1100001001,101})_2$	$(\mathbf{1411,5})_8$	$(\mathbf{309,A})_{16}$

2-

$(1011.0011)_2 = 1 \times 2^0 + 1 \times 2^1 + 0 \times 2^2 + 1 \times 2^3 + 0 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} + 1 \times 2^{-4} = (11{,}1875)_{10}$

$(122.23)_4 = (26{,}6875)_{10}$

$(7.7)_8 = (7{,}875)_{10}$

$(4B.CC)_{16} = (75{,}796875)_{10}$

$(14.82)_9 = (13{,}91358024)_{10}$

Exercice 3

a- $(11{,}01)_2 + (101{,}1)_2 = (1000{,}11)_2$

b- $(111{,}01)_2 + (1101{,}10)_2 = (10100{,}11)_2$

c- $01111 + 10010 = 100001$

d- $010111 - 110010 = 100101$

e- $11010 - 100{,}11 = 10101{,}01$

f- $10110101 * 1011 = 11110000111$

g- $110111/1011 = 101$

Exercice : 04

$25 + 36 = 61 = (111101)_2 = (100011)_{Gray}$

$(27)_{10} = (11011)_2 = (10110)_{Gray}$

$(21)_{10} = (10101)_2 = (11111)_{Gray}$

Exercice : 05

1- convertir en code BCD

$(10101)_2 = (15)_{BCD}$

1 5

$(11001)_2 = (19)_{BCD}$

1 9

$(10001)_2 = (11)_{BCD}$

1 1

2- convertir en code BCD

a) $(15)_{10} + (10)_{10} = \underbrace{00010101}_{1\ 5} + \underbrace{00010000}_{1\ 0} = \underbrace{00100101}_{2\ 5}$

3- convertir en code Gray

$(11011)_2 = (10110)_{Gray}$

$(1001010)_2 = (1101111)_{Gray}$

$(101101)_2 = (111011)_{Gray}$

4- convertir en code binaire

$(1010)_{Gray} = (1100)_2$

$(10010)_{Gray} = (11100)_2$

$(100111)_{Gray} = (111010)_2$

Chapitre 2
Algèbre de Boole et Simplification des Fonction Logique

Chapitre 2 : Algèbre de Boole et simplification des fonctions logiques

1 Définition

« L'algèbre de Boole est un ensemble de variables à deux états de vérités : 1 (vrai) et 0 (faux), manipuler par un nombre limité d'opérateurs : et, ou, non. ». Il contient un ensemble de théorèmes mathématiques qui précisent les fondements théoriques de la logique binaire ou booléenne.

2 Logique combinatoire

2.1 Introduction :

Les circuits des machines électromécaniques ont 2 états d'équilibres qui sont 0 et 1, ils sont caractérisés par 2 niveaux de tension qui définissent un signal logique

2.2 Variable logique :

C`est un symbole qui représenté les deux états de l`équilibre (stable) 0 et 1 .

Exemple :

- Une vanne est ouverte ou fermée

Niveau	Tension
0:Bas (B)	0v
1:Haut (H)	5v

- Une lampe est allumée ou éteinte.
- Un interrupteur est ouvert ou fermé.

Remarque : deux cas logique :

- Logique négatif (0 actif ,1 inactif)
- Logique positif (1 actif, 0 inactif)

3 Fonction logique

3.1 Définition

C'est une expression logique (de valeur 0 ou 1) qui combine un ensemble de variables booléennes à l'aide des opérateurs logiques ou, et, non.

3.2 Présentation

Une fonction logique peut être présentée par :

Une table de vérité :

C'est une table qui décrit toutes les combinaisons des entrées et la valeur de la fonction (sortie) pour chaque entrée.

Exemple :

a	b	c	S
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

a,b,c sont des variable d'entrée
S est la variable de sortie

Valeur de sortie

Combinaisons des variables d`entrée

Tableau2.1 : Exemple de la table de vérité

4 Les opérations de l'algèbre de Boule

4.1 L'operateur ET :

L'opérateur AND (ET) porte sur deux variables d'entrée. Si A et B sont les variables d'entrée, alors S = A.B. S est vrai si A **ET** B sont vraies. L'opérateur AND est symbolisé par le point (.) comme la multiplication en mathématique (c'est d'ailleurs l'opération réalisée en binaire).

Le circuit intègre CMOS : 4081 TTL : 7408

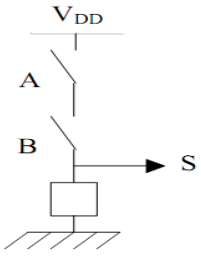
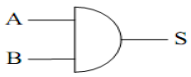
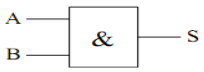
Table de vérité			Montage	Symbole traditionnel	Symbole normalisé
A	B	S = A.B			
0	0	0			
0	1	0			
1	0	0			
1	1	1			

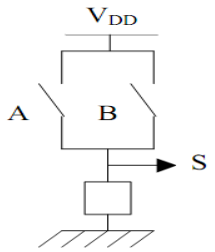
Tableau2.1 : Représentation d'opérateur ET

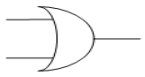
4.2 L'operateur OU

L'opérateur OR (OU) porte sur deux variables d'entrée. Si A et B sont les variables d'entrée, alors S = A+B. S est vrai si A **OU** B sont vraies. L'opérateur OR est symbolisé par le plus (+) comme l'addition en mathématique. Le circuit intègre TTL : 7432.

Table de vérité	Montage	Symbole traditionnel	Symbole normalisé

A	B	S = A+B
0	0	0
0	1	1
1	0	1
1	1	1





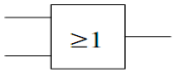


Tableau2.2 : Représentation d'opérateur OU

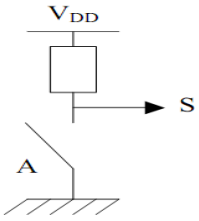
4.3 L'operateur NoN :

Il est aussi appelle Inverseur, c'est une fonction a un variable.

$F(A) = A^{-}$, Il se lit A Barre ou NoN A (l'interrupteur ouvert vaut 0 et l'interrupteur fermé vaut(1)

Table de vérité	Montage	Symbole traditionnel	Symbole normalisé

A	$S = \overline{A}$
0	1
1	0



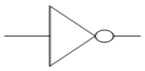


Tableau2.3 : Représentation d'opérateur NON

4.4 La porte OU-exclusif (XOR)

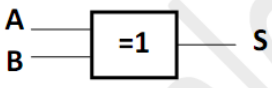
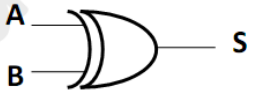
Symbole logique		Equation	Circuit intégré
Symbole International (CEI)	Symbole Européen (MIL)		
		$S=A\oplus B$ $=\overline{A}B*A\overline{B}$	TTL : 7486 CMOS : 4070

Tableau2.4 : Représentation d'opérateur XOR

La fonction OU-exclusif vaut 1 si une seule des entrées est à l'état 1 et l'autre est l'état 0.

La fonction OU-exclusif vaut 1 si une seule des entrées est à l'état 1 et l'autre est l'état 0.

Table de vérité		
A	**B**	**S**
0	0	0
0	1	1
1	0	1
1	1	0

4.5 Généralisations de la fonction OU-EXCLUSIF :

La sortie de la fonction OUEXCLUSIF prend l'état logique 1 si un nombre impair des variables d'entrée est à l'état logique 1.

Exemple : OU-exclusif a trois entrées

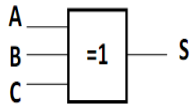
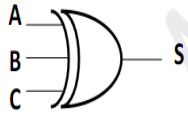
Symbole logique		Equation	Circuit intégré
Symbole International (CEI)	Symbole Européen (MIL)		
		S=A⊕B⊕C	TTL : 74386

Table de vérité			
A	B	C	S
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Tableau2.5 : Représentation d'opérateur XOR a trois entrées

5 Les portes universelles

Autre que les portes logiques de base (ou élémentaires), il existe des portes appelées portes logique universelles (complètes) telles que les portes NON-ET et NON-OU.

5.1 La porte NON-ET (NAND)

C`est l`operateur inverse de l`operateur <ET> il est notée par convention / il est dit A **nand** B

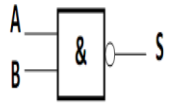
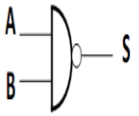
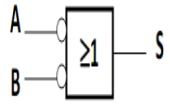
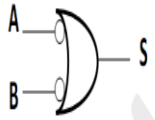
Symbole logique		Equation	Circuit intégré
Symbole International (CEI)	Symbole Européen (MIL)		
		$S=\overline{A.B}$ $S=\overline{A}+\overline{B}$	TTL : 7400 CMOS : 4011-4093
			

Table de vérité		
A	B	S
0	0	1
0	1	1
1	0	1
1	1	0

Tableau2.6 : Représentation d'opérateur NAND

5.2 La porte NON-OU (NOR)

C`est l`operateur inverse de l`operateur <OU> il est notée par convention ↓

Il est dit A **nord** B

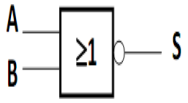
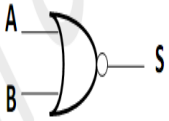
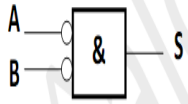
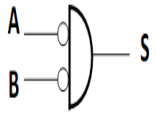
Symbole logique		Equation	Circuit intégré
Symbole International (CEI)	Symbole Européen (MIL)		
		$S=A\downarrow B$ $S=\overline{A+B}$ $S=\overline{A}.\overline{B}$	TTL : 7402 CMOS : 4001
			

Table de vérité		
A	B	S
0	0	1
0	1	0
1	0	0
1	1	0

Tableau2.7 : Représentation d'opérateur NOR

6 Théorème fondamental et propriété de l'algèbre de Boole

Deconstane (neutre Absorption)	$A+0=A$ $\quad A*1=A$ $A+1=1$ $\quad A*0=0$
Idempotence	$A+A=A$ $A*A=A$
Complementation	$A*\overline{A}=0$ $A+\overline{A}=1$
Commutative	$A*B=B*A$ $A+B=B+A$
Distribution	**ET sur OU** $A*(B+C)=(A*B)+(A*C)$ **OU sur ET** $A+(B*C)=(A+B)*(A+C)$
Association	$A*(B*C)=(A*B)*C$ $A+(B+C)=(A+B)+C$

Tableau2.8 : Théorème fondamental et propriété de l'algèbre de Boole

7 Théorème de Morgan

Théorème 1 : la négation d'un produit de variable est égale à la somme des négations des variables

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

Théorème 2 la négation d'une somme de variables est égale au produit des négations des variables

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

Remarque :

Généralisation des 2 théorèmes :

Le complément de expression s`obtient en inversant les variables utilises et en permettant les opérateurs <ET>et <OU>

$\boldsymbol{F(\overline{x},+,*) = F(\overline{x},*,+)}$

8 Dualité

Deux expressions se correspondent par la dualité si l'on obtient l`une en changeant dans l'autre, les (ET) par les (OU), les (OU) par des (ET), les (1) par (0), les (0) par les (1).

Exemple :

$xy + x\bar{y} = x$ Par expression duale $\boldsymbol{(x+y).(x+\overline{y}) = x}$

$\boldsymbol{\overline{xyz} = \overline{x} + \overline{y} + \overline{z}}$ Par expression duale $\boldsymbol{\overline{x+y+z} = \overline{x}.\overline{y}.\overline{z}}$

1+A=1 Par expression duale **0.A=0**

A+A=A Par expression duale **A.A=A**

9 Théorème de consensus

$AB + \bar{A}C + BC = AB + \bar{A}C$

$$(A + B)(\bar{A} + C)(B + C) = (A + B)(\bar{A} + C)$$

10 Mode de représentation des fonctions logiques

Une fonction booléenne peut être représentée par :

- ✓ Ecriture algébrique.
- ✓ Table de vérité.
- ✓ Table de Karnaugh.
- ✓ Logigramme.
- ✓ Chronogramme.

10.1 Ecriture algébrique

Soit les fonctions :

$$\boldsymbol{S = x\bar{y}\bar{c} + \bar{x}y\bar{c} + \bar{x}\bar{y}c + xyc}$$

$$\boldsymbol{C = xy\bar{c} + x\bar{y}c + \bar{x}yc + xyc}$$

Ces expressions sont sous forme algébrique.

10.2 Table de vérité

Une fonction a (n)variables présente 2^n résultats de sortie. Ainsi, elle sera représentée par une table de vérité de 2^n lignes.

Exemple :

On peut représenter ces deux fonctions précédant sous forme d'une table de vérité. Les entres sont dans l'ordre binaire naturel. Nous avons trois variables x, y et c, donc $2^3 = 8$ combinaisons possibles .

c	y	x	S	C
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

10.3 Table de Karnaugh

La table de vérité présente un inconvénient ;le nombre de ligne croit en même temps que le nombre de variable .si n le nombre de variable, le nombre de lignes est 2^n .pour trois variable on a huit ligne , pour quatre variable on a seize ligne etc ,On contourne cette difficulté en plaçant les variables d'entrés aux entrées d'une table aussi car possible. On a donc une table dite Karnaugh a doublé entrée, une pour les lignes et une pour les colonnes .soit n variable, la table sera la plus carrée possible pour p et q entiers tels que $p + q = n$ lorsque

$p = q = n/2$ pour n pair ,ou $p - q = 1$ pour n impair.

La table comportera alors 2^p colonnes 2^q lignes.

Donc à partir de la table de vérité, on inscrit dans les cases les 0 et les 1 de la fonction S de la façon suivante. La premier case en haut à gauche correspond à $x = y = c = 0$, on inscrit donc 0 dans cette case. La case intersection de la 1er ligne et la 2eme colonne correspond a $x = 1, y = c = 0$ .on inscrit donc 1 dans cette case et ainsi de suite.

Table de S

yx / *c*	00	01	11	10
0	0	1	0	1
1	1	0	1	0

Table de C

yx / c	00	01	11	10
0	0	0	1	0
1	0	1	1	1

Pour obtenir la fonction de chaque case il suffit d'effectuer pour chaque case le produit des variables en complémentant chaque variable de valeur 0 ce qui donne pour S

yx / c	00	01	11	10
0	$\bar{x}\bar{y}\bar{c}$	$x\bar{y}\bar{c}$	$xy\bar{c}$	$\bar{x}y\bar{c}$
1	$\bar{x}\bar{y}c$	$x\bar{y}c$	xyc	$\bar{x}yc$

10.4 Définition par logigramme:

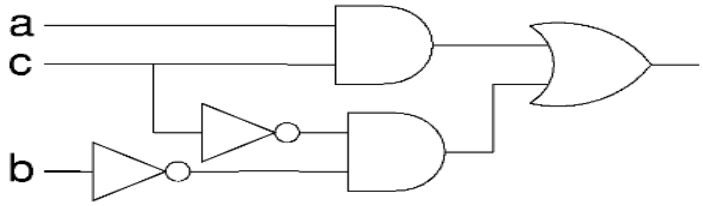


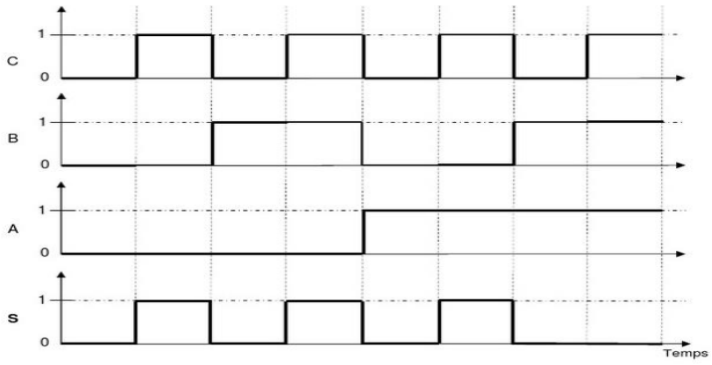
Figure2.1 : Logigramme logique

Alors la fonction de sortie F est egale :

$F = (a, b, c) = ac + \bar{b}\bar{c}$

10.5 Représentation par chronogramme:

Le chronogramme est une représentation graphique permettant de visualiser, en fonction du temps, toutes les combinaisons d'états logiques possibles des entrées avec l'état correspondant de la sortie.



A	B	C	S
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

Figure2.2 : Chronogramme logique

11 Formes Canoniques :

C'est une équation qui permet de localiser directement chaque case du tableau de Karnaugh ou table de vérité comportant un « 1 » logique ou un « 0 » logique. On distingue principalement deux formes canoniques qui sont :

11.1 Premières formes canoniques : Somme de Produit (mintermes)

Considérant la table de vérité ou le tableau de Karnaugh de la fonction logique. A chaque 1 logique de la variable de sortie, on fait correspondre le produit des n variables d'entrées.

Dans ce produit, chaque variable sera sous forme normale si elle est à 1 et sous forme complémentée si elle est à 0. L'expression de la fonction sera la somme des produits élémentaires ainsi formés.

Exemple :

$F = 1\ si\ (a,b,c) = (0,1,1)\ ou\ (1,1,1)\ ou\ (1,0,0)\ ou\ (1,0,1)$

Si on note n = (abc) alors F vaut 1 si et seulement si n = 3 ou 7 ou 4 ou 5.

On écrit alors f (a, b, c) = (3,4,5,7)

Dans notre cas précédant la table de vérité de la fonction S permet d`écrire :

a	b	c	S
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Table de vérité :

$$S = x\bar{y}\bar{c} + \bar{x}y\bar{c} + \bar{x}\bar{y}c + xyc = (2,3,5,8)$$

Exemple d`application

Etablir l'équation logique du système S (a,b,c) = (0,1,2,6,7)

L'équation de la fonction sous la 1ère forme canonique :

$$s = m_0 + m_1 + m_2 + m_6 + m_7$$
$$s = \bar{a}\bar{b}\bar{c} + \bar{a}\bar{b}c + \bar{a}b\bar{c} + ab\bar{c} + abc$$

11.2 Deuxième forme canonique: Produit de Somme(Maxtermes)

Considérant la table de vérité ou le tableau de Karnaugh de la fonction logique. A chaque 0 logique de la variable de sortie, on fait correspondre la somme des n variables d'entrées.

Dans cette somme, chaque variable sera sous forme normale si elle est à « 0 » et sous forme complimentée si elle est à « 1 ».

L'expression de la fonction sera le produit des sommes élémentaires ainsi formés.

Exemple: établir l'équation logique du système f(a,b,c) = (0,1,2,6,7)

a	b	c	F
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Table de vérité :

L'équation de la fonction sous la 2eme forme canonique :

$$f = M_3 . M_4 . M_5$$
$$f = (a + \bar{b} + \bar{c}).(\bar{a} + b + c).(\bar{a} + b + \bar{c})$$

12 Simplification des fonctions logiques

12.1 Définition :

On appelle forme minimale d'une expression logique l'expression sous forme réduite (Somme de produit) qui comporte :

1- Le nombre minimal de terme

Le nombre minimal de variable dans chaque terme.

On dispose de plusieurs outils de simplification de fonction logique dont on va citer le plus importants.

12.2 Simplification algébrique

Dans cette première méthode, on se base essentiellement sur les théorèmes de l'algèbre de Boole pour simplifier les expressions logiques.

Malheureusement, il n'est pas toujours facile de savoir quel théorème faut-il évoquer pour obtenir la simplification minimale.

Exemples : simplifier les fonctions suivantes :

$F = (ab + \bar{a}c + bc)$

$F = (ab + \bar{a}c + bc.1) = \big(ab + \bar{a}c + bc(a + \bar{a})\big) = (ab + \bar{a}c + bca + bc.\bar{a})$

$F = \big(ab + bca + \bar{a}c(1 + b)\big) = (ab(1 + c) + \bar{a}c(1 + b))$

$\boldsymbol{F = (ab + \bar{a}c)}$

12.3 Simplification méthode de Karnaugh

On peut simplifier une fonction logique représentée par un tableau de karnaugh en effectuant des regroupements de 2, 4, 8, 16, ... cases adjacentes remplies toutes avec des 1 logiques. Ceci va nous permettre de simplifier 1 ou 2 ou 4 ou plusieurs variables logiques.

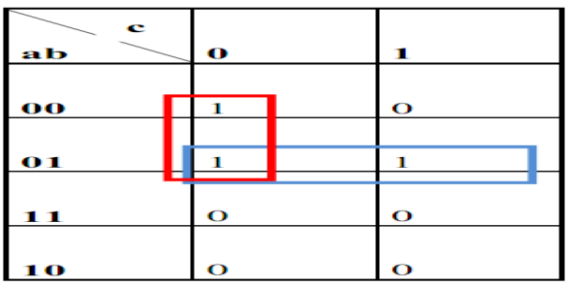
D'une manière générale, pour une fonction de n variables, un regroupement de 2^k Cases nous donnera une équation de (n-k) variables.

Remarque :

Un groupement de 1 permet d'obtenir l'équation de F ,un groupement de 0 permet d'obtenir l'équation de $\overline{F}$.

12.3.1 Regroupement de doublets

Le regroupement de deux cases adjacentes, verticalement ou horizontalement,ou symétriques



remplies des 1 logiques simplifie une variable dans l'expression de la fonction. L'expression canonique de cette fonction est :$f = \bar{a}\bar{b}\bar{c} + \bar{a}b\bar{c} + \bar{a}bc$

L'expression réduite (simplifiée de la fonction) ; $f = \bar{a}\bar{c} + \bar{a}b$

12.3.2 Regroupement de quartets

Un groupement de 4 cases adjacentes ou symétriques remplies des 1 logiques va simplifier 2 variables dans l'expression canonique de la fonction logique.

ba / dc	00	01	11	10
00	0	0	0	0
01	1	1	0	0
11	1	1	0	0
10	0	0	0	0

La forme canonique de F est :$F = \bar{a}\bar{b}\bar{d}c + \bar{a}\bar{b}cd + \bar{b}a\bar{d}c + \bar{b}adc$
L'expression réduite $F = \bar{b}c$

12.3.3 Regroupement d'octets

Un groupement de 8 cases adjacentes ou symétriques remplies des 1 logiques va simplifier 3 variables logiques dans l'expression canonique de la fonction logique.

ab / cd	00	01	11	10
00	1	1	1	1
01	0	0	0	0
11	0	0	0	0
10	1	1	1	1

L'expression canonique de F est :

$$F = \bar{a}\bar{b}\bar{c}\bar{d} + \bar{a}b\bar{c}\bar{d} + ab\bar{c}\bar{d} + a\bar{b}\bar{c}\bar{d} + \bar{a}\bar{b}cd + \bar{a}bcd + abcd + a\bar{b}cd$$

L'expression réduite F = $\bar{d}$

TD 02

(Algèbre de Boole et Fonction Logique)

Exercice : 01

1- Simplifiez les functions suivante

$Z = (a + b)(\bar{b} + c)(\bar{a} + c)$ $Z = (a + b)(\bar{a} + \bar{b} + c)(a + c)$

2- Ecrire l`égalité D des égalités E ci-dessous

$E = a + bc + ab = a + bc$ $E = ab(\bar{c}d + \bar{b}c) = ab\bar{c}d$

Exercice : 02

1. Ecrire les deux fonctions canoniques de l'expression suivant

   $Z = ab + \bar{a}\bar{b}c$

2. Ecrire les formes canoniques de l'expression Z suivante donnée par sa table de karnaugh

ba / c	00	01	11	10
0	1	1	0	1
1	1	0	0	0

Exercice : 03

Représenter les expressions E_1, E_2 sous forme de :

$E_1 = ab + a\bar{c}$ $E_2 = \bar{a}bc + a\bar{b}c + ab\bar{c}$

1. Table de vérité dans l'ordre binaire naturel.
2. Table de karnaugh.
3. Logigramme.

Exercice : 04

Représenter l`expression $E = (a + b)(\bar{a} + \bar{b} + \bar{c})(a + c)$ par :

1. Sous la forme d'une table de Karnaugh
2. Sous la première forme canonique
3. Sous la de la deuxième forme canonique
4. Sous la forme simplifier
5. Sous la forme d'un logigramme en n`utilisant que les portes NON-ET

Exercice : 05

I- Utiliser la table de Karnaugh pour simplifier l'expression suivante

$E = \bar{A}\bar{B}CD + \bar{A}B\bar{C}\bar{D} + \bar{A}BCD + \bar{A}BC\bar{D} + AB\bar{C}\bar{D} + AB\bar{C}D$

II- Soit l'expression $y = \bar{a}\bar{b}cd + abc + \bar{a}\bar{c}d$

1- Représenter sous forme de table de karnaugh

2- Déterminer la forme simplifier de $\bar{y}$

Solution

Exercice 01 :

1- $Z = (a+b)(\bar{b}+c)(\bar{a}+c) = (a\bar{b}+b\bar{b}+ac+bc)(\bar{a}+c)$

$= \left(a\bar{b}+ac+bc\right)(\bar{a}+c) = \left(a\bar{a}\bar{b}+a\bar{a}c+\bar{a}bc\right)+(a\bar{b}c+acc+bcc)$

$= \bar{a}bc+a\bar{b}c+ac+bc = bc(\bar{a}+1)+ac(\bar{b}+1)$

Donc :$Z = bc+ac = c(b+a)$

$Z = (a+b)(\bar{b}+\bar{b}+c)(a+c)$

$Z = \left(a\bar{a}+b\bar{a}+a\bar{b}+b\bar{b}+ac+bc\right)(a+c) = (b\bar{a}+a\bar{b}+ac+bc)(a+c)$

$Z = a\bar{a}b+aa\bar{b}+aac+abc+\bar{a}bc+a\bar{b}c+acc+bcc$

$Z = a\bar{b}+ac+abc+\bar{a}bc+a\bar{b}c+ac+bc$

$Z = a\bar{b}+ac+abc+\bar{a}bc+a\bar{b}c+bc$

$Z = ac(1+b)+a\bar{b}+bc(1+\bar{a})+a\bar{b}c$

$Z = ac+a\bar{b}+bc+a\bar{b}c = c(a+b)+a\bar{b}(1+c)$

$Z = c(a+b)+a\bar{b}$

Donc :$Z = c(a+b)+a\bar{b}$

2- L'égalité D de l'égalité E

$D_1 = a.(b+c).(a+b) = a.(b+c)$

$D_2 = (a+b)+(\bar{c}+d)\left(\bar{b}+c\right) = a+b+\bar{c}+d$

Exercice 02 :

$$Z = ab+\bar{a}\bar{b}c$$

La première forme canonique

$Z = ab+\bar{a}\bar{b}c$

$Z = ab(c+\bar{c})+\bar{a}\bar{b}c$

$Z = abc+ab\bar{c}+\bar{a}\bar{b}c = m_7+m_6+m_1$ (mintermes)

- La deuxième forme canonique

$Z = ab+\bar{a}\bar{b}c$

$Z = (ab+\bar{a})(ab+\bar{b}c)$

$Z = (\bar{a}+a)(b+\bar{a})(ab+\bar{b})(ab+c)$

$Z = (\bar{a}+a)(b+\bar{a})(a+\bar{b})(b+\bar{b})(a+c)(b+c)$

$Z = (\bar{a}+b)(a+\bar{b})(a+c)(b+c)$

$Z = (\bar{a}+b+c\bar{c})(a+\bar{b}+c\bar{c})(a+c+b\bar{b})(b+c+a\bar{a})$

$Z = (\bar{a} + b + c)(\bar{a} + b + \bar{c})(a + \bar{b} + c)(a + \bar{b} + \bar{c})(a + b + c)(a + c + \bar{b})(a + b + c)(\bar{a} + b + c)$

3- Les deux forms canonique

$Z = \bar{a}\bar{b}\bar{c} + a\bar{b}\bar{c} + \bar{a}b\bar{c} + \bar{a}\bar{b}c$ (Minterme)

$Z = (\bar{a} + \bar{b} + c) + (\bar{a} + b + \bar{c}) + (\bar{a} + \bar{b} + \bar{c}) + (a + \bar{b} + \bar{c})$ (Maxterme)

Exercice 4

$$E = (a + b)(\bar{a} + \bar{b} + \bar{c})(a + c)$$

1- Sous la forme d'une table de karnaugh

bc	00	01	11	10
0	0	0	1	0
1	1	1	0	1

2eme méthode 2eme forme canonique :

$E = (a + b)(\bar{a} + \bar{b} + \bar{c})(a + c)$

$E = (a + b + c\bar{c})(\bar{a} + \bar{b} + \bar{c})(a + c + b\bar{b})$

$E = (a + b + c)(a + b + \bar{c})(\bar{a} + \bar{b} + \bar{c})(a + b + c)(a + \bar{b} + c)$

$E = (a + b + c)(\bar{a} + b + \bar{c})(\bar{a} + \bar{b} + \bar{c})(a + \bar{b} + c) = M_0 + M_1 + M_7 + M_2$

2- Sous la première forme canonique

$E = a\bar{b}\bar{c} + a\bar{b}c + \bar{a}bc + ab\bar{c} = m_4 + m_5 + m_3 + m_6$

3- Sous la deuxième forme canonique

$E = (a + b + c)(\bar{a} + b + \bar{c})(\bar{a} + \bar{b} + \bar{c})(a + \bar{b} + c) = M_0 + M_1 + M_7 + M_2$

4- Sous la forme simplifier

$E = \bar{a}bc + a\bar{b} + a\bar{c}$

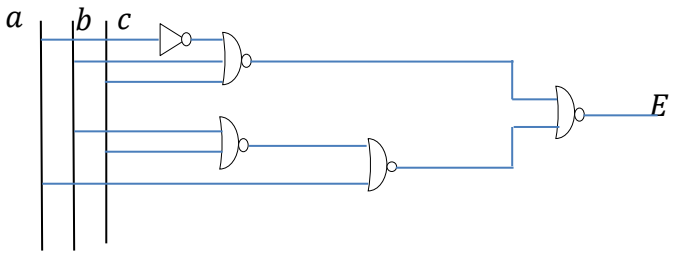
$E = \bar{a}bc + a(\bar{b} + \bar{c}) = \bar{a}bc + a\overline{bc}$

5- Sous la forme d'un logigramme en utilisant que les porte NON-ET

$E = \bar{\bar{E}}$ donc :

$E = \bar{a}bc + a\overline{bc} = \overline{\overline{\bar{a}bc + a\overline{bc}}}$

$E = \overline{\bar{a}bc . a\overline{bc}}$



Exercice 5

1- Table de Karnaugh

$$E = \bar{A}\bar{B}CD + \bar{A}B\bar{C}\bar{D} + \bar{A}BCD + \bar{A}BC\bar{D} + AB\bar{C}\bar{D} + AB\bar{C}D$$

Donc :

$$E = B\bar{C} + \bar{A}B + \bar{A}CD$$

CD / AB	00	01	11	10
00	0	0	1	0
01	1	1	1	1
11	1	1	0	0
10	0	0	0	0

2- $y = \bar{a}\bar{b}cd + abc + \bar{a}\bar{c}d$

$y = \bar{a}\bar{b}cd + abc(d + \bar{d}) + \bar{a}\bar{c}d(b + \bar{b})$

$y = \bar{a}\bar{b}cd + abcd + abc\bar{d} + \bar{a}b\bar{c}d + \bar{a}\bar{b}\bar{c}d$

Table de Karnaugh

La forme simplifié de $\bar{y}$:

$$\bar{y} = a\bar{c} + \bar{a}bc + \bar{d}$$

ba / dc	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	0
10	1	0	0	1

Chapitre 3

Technologie Des Circuits Intègres CMOS et TTL

Chapitre 3 : Technologie des Circuit Intègres CMOS et TTL

1- Définition

Les circuits intègres logiques (CI) appelés aussi puce ou chip (en anglais) sont des composants que l'on trouve dans tous les appareils électroniques (ordinateurs, appareil de mesures,……etc).

Ces composants électroniques sont bases sur un semi-conducteur, reproduisant une, ou plusieurs, fonction électronique plus ou moins complexes et intégrant souvent plusieurs types de composants électronique de base dans un volume réduit (sur une petite plaque), rendant le circuit facile à mettre en œuvre.

Il existe une très grande variété de ces composants divisés en deux grandes catégories : analogique et numérique.

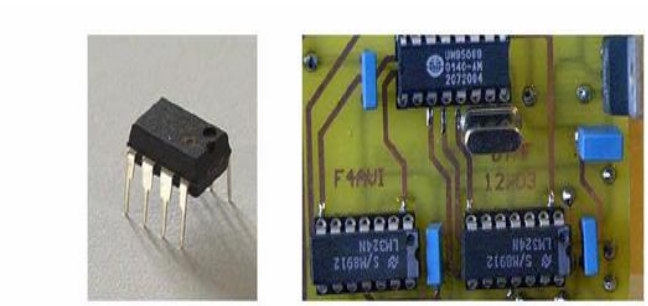


Figure 3.1 : Circuit Intègre

2- Description matérielles :

La partie active (Die) d'un circuit logique est formée d'une palette de silicium d'environ 5x5 mm sur laquelle ont été intégrées par divers procèdes technologiques, les portes qui constituent ce circuit.

La Die est la partie élémentaire, de forme rectangulaire, reproduite comme une copie conforme avec une matrice sur une tranche de silicium en cours de fabrication.

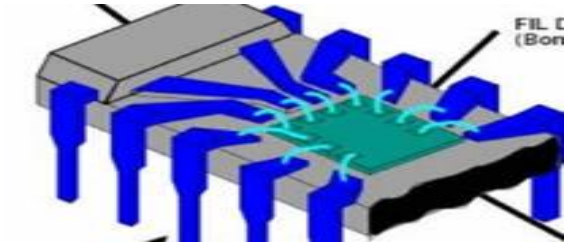


Figure 3.2 : La Die de Circuit Intègre

3- Technologies de fabrication de CI

La fabrication des circuits intégrés est un processus complexe qui implique la création de minuscules dispositifs électroniques, tels que des transistors, sur une plaquette de silicium. Ce processus se déroule dans des "salles blanches" où la propreté et les conditions environnementales sont rigoureusement contrôlées.

- DTL (Diode-Transistor-Logic) : est une famille logique qui a été développes dans les années 1960 .elle utilise des diodes pour réaliser la fonction logique principale (AND), et un transistor pour l'amplification (inversion) c'est une étape intermédiaire entre RTL(Resistor-Tronsistor Logic) et TTL(Transistor-Tronsistor logic) .
- TTL (Transistor-Tronsistor logic) : est une famille de circuit logique ou les transistor sont utilisés à la fois pour la fonction logique et l'amplification.
  Elle a été développée pour remplacer les technologies de DTL et RTL.
- ECL(Emitter-coupled-logic) :c'est famille de circuit logiques numériques basée sur les transistors bipolaires (BJT) ,très grande vitesse de commutation mais avec consommation d'énergie élevée même au repos ,car le courant circule en permanence
- CMOS (Complementary-Metal-Oxide-Semiconducteur) :c'est une technologie utilisée pour fabrique la majorité des circuits intègres numériques, comme les microprocesseurs, les mémoires, et la plupart des circuits logique modernes. Avec très faible consommation d'énergie statique, haute densité d'intégration, donc on peut mettre beaucoup de transistors sur une puce. Mais avec plus sensible aux perturbations électriques.

4- Présentation des Circuits intégrés :

Un circuit intègres referme plusieurs portes logiques, dont les entrées et les sorties sont accessibles sur différentes bornes des circuits intégrés.

Les broches (pins en anglais) d'un circuit intégré sont numérotées. Pour déterminer la position du pin 1 il faut repérer une encoche sur le composant tel que :

En regardant le circuit intégré avec l'encoche vers le haut, la numérotation des broches se fait dans le sens antihoraire comme suit :

Broche1 : en bas à gauche (juste à gauche de l'encoche), tu continues à monter sur le cote gauche jusqu'à la dernière broche de ce cote, en suit, tu passes en haut à droite, et tu descends.

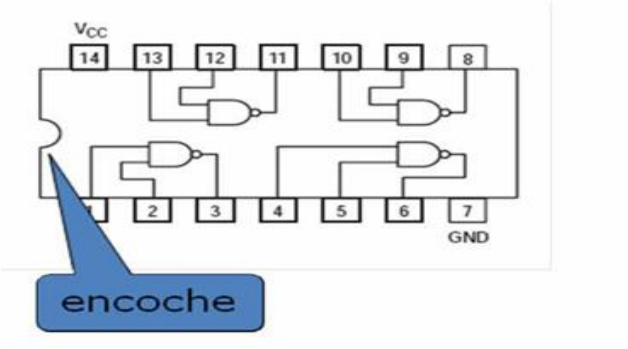


Figure 3.3 : Présentation des Circuit intégré

5- Identification des Circuit intégré

Chaque circuit intégré possède une référence inscrite sur le dessus de son boitier cette référence, composée de 4 à 7 caractères (chiffres et /ou lettres), permet d'identifier facilement le composant.

Par exemple la photo citée la référence du circuit de gauche est 4029 et la référence du circuit de droite est 74S113.

Pour connaitre la fonction précise d'un CI à partir de sa référence, il est nécessaire de consulter le Memotech électronique ou la datasheet correspondant.

On distingue principalement deux grandes familles :

- Technologie CMOS : les circuits dont la référence commence par 4000(ex : 4011,4029) ces circuit se caractérisent par une faible consommation d'énergie et une large plage de tension.
- Technologie TTL : les circuits dont la référence commence par 74(ex : 7400,74LS00, 74S113) ces circuit sont plus rapides mais consomment davantage.

Exemple 1:Circuit CMOS nomme CD4011BE

CD : Préfixe utilise par Texas Instruments.

4011 : Numéro du circuit,il s'agit ici d'un quadruple porte NAND(NOT-ET) a deux entrées chacune.

B : Indique que la tension maximale est de 18 V

E : Indication que le circuit est encapsule dans un boitier DIP

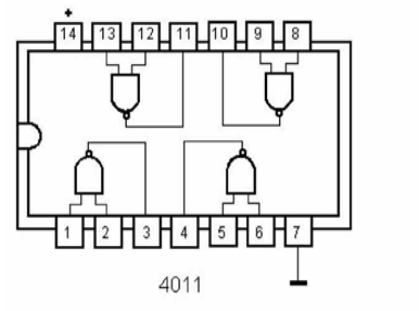


Figure 3.4 : Exemple Circuit intégré type CMOS

Exemple 2 : Circuit TTL nomme SN74LS00

SN : Signifie que le constructeur est Texas Instrument

74 : Désigne les circuits intègres grands publics qui supportent une température ambiante comprise entre 0 et 70 degré.

LS : Indiquent la sous famille du circuit TTL

00 : Les derniers chiffres indiquent la fonction logique réalisé par le composant (00=Porte NAND, 02=Porte NOR, 08=Porte AND etc).

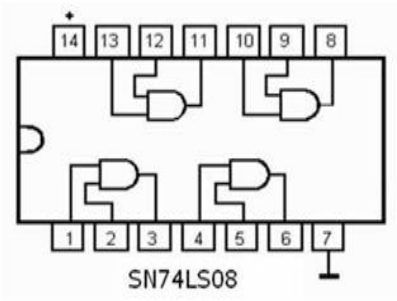


Figure 3.5 : Exemple Circuit intégré type TTL

Chapitre 4

Les Circuits Combinatoires

Chapitre 4 : Les Circuits Combinatoires

1 Circuit combinatoire :

Les circuits logiques combinatoires (CLC) sont la première forme de circuits que nous aurons à étudier dans ce cours. Les circuits logiques combinatoires sont des circuits caractérisés par leur propriété déterministe, associant à toute combinaison d'entrées une seule et même combinaison de sorties. Pour cette raison, les circuits combinatoires sont souvent vus comme des boîtes noires régies par ce fonctionnement déterministe entrées/sorties et dont l'implémentation sera pour nous sujette à discussion.

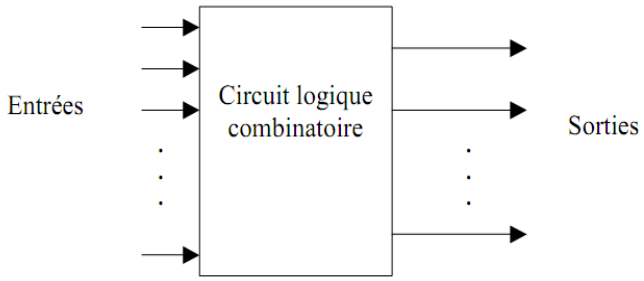


Figure4.1 : Circuit Combinatoire

Les circuits combinatoires peuvent servir par exemple :

- à traduire des bits en chiffres représentant un nombre (ou des lettres ou un code particulier). On appelle ces circuits des codeurs (ou bien des décodeurs pour l'opération inverse). Par exemple, un codeur Gray ou bien BCD.
- à effectuer des opérations arithmétiques sur des nombres. Par exemple, un additionneur ou un multiplieur.
- à transmettre ou recevoir des informations sur une ligne unique de transmission (une ligne série), ce qui nécessite de transformer un nombre écrit sous forme parallèle en une suite de bits mis en série et vice-versa. C'est le rôle des circuits multiplexeur/démultiplexeur. Voici par exemple une transformation série/parallèle suivie d'une transformation parallèle/série :

Dans ce chapitre, nous allons réaliser des circuits combinatoires qui permettent d'établir les opérations d'addition, de soustraction et de comparaison de deux nombres binaires.

2 Additionneur

C`est un circuit logique permette de réaliser une addition ce circuit et présenter dans les ordinateurs pour les calcule arithmétique, également pour le calcul d`adresses.

Une addition met en œuvre deux sorties :

- La somme, généralement notée **S**
- La retenue, généralement notée **R**

La figure suivante montre la décomposition de l'addition de deux nombres binaires de 4 bits.

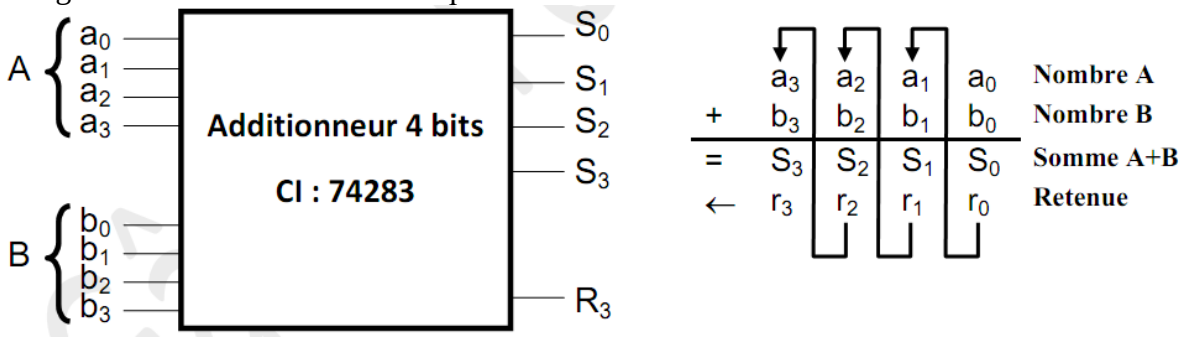


Figure4.2 : Additionneur 4 bits

2.1 Demi additionneur

Le demi additionneur permette de réaliser une somme arithmétique de deux nombres a et b sur un bit.il a deux sorties : la somme S et la retenue R.

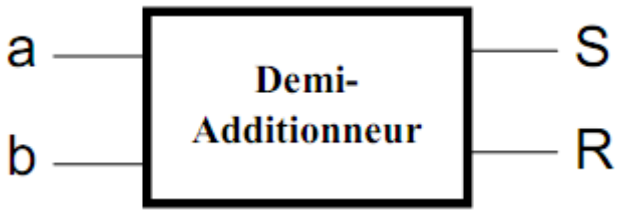
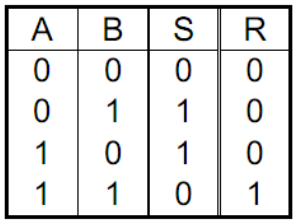
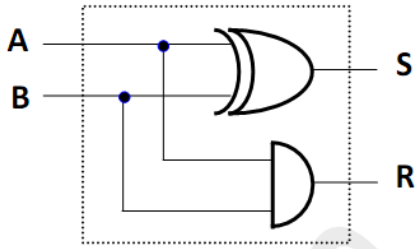


Figure4.3 : Demi-Additionneur

Table de vérité	Equation des sorties	Logigramme
	$S=\bar{A}B+A\bar{B}=A\oplus B$ $R=AB$	

A	B	S	R
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Tableau4.1 : Représentation Demi additionneur

2.2 Additionneur complet :

Un additionneur est un circuit combinatoire qui présente la structure suivante :

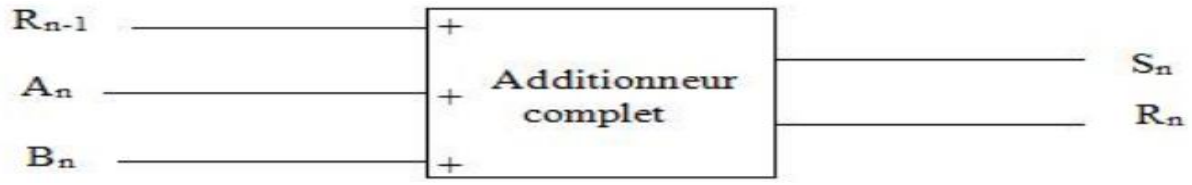


Figure4.4 : Schéma d'un additionneur complet

Circuit intégré : 74LS183

Où :

A_n et B_n sont les deux bits du rang n à additionner

R_{n-1} est une retenue de l'étage précédent qui doit être prise en considération dans Addition

S_n est le résultat de l'opération d'addition du rang n

R_n est la retenue provoquée par l'addition et renvoyée vers l'étage suivant

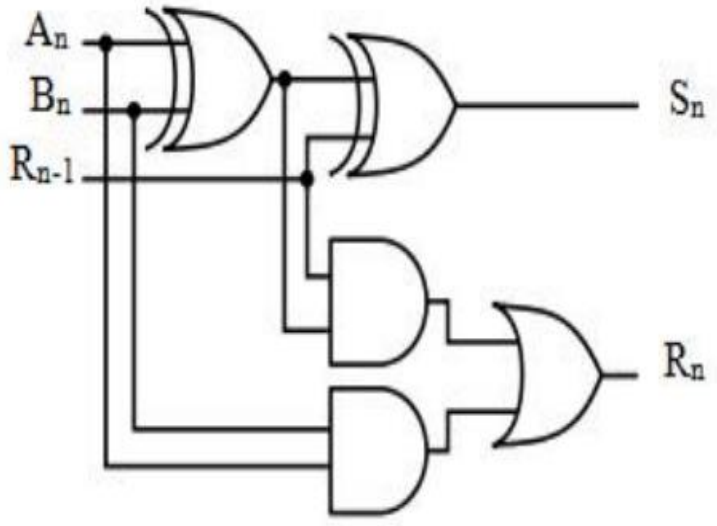
Application

Etablir la table de vérité d'un additionneur complet 1 bit

(AC : élémentaire). Donner le logigramme de cet additionneur à l'aide des portes logiques de votre choix.

Solution

Logigramme d'Addition complet



La table de vérité d'Addition complet

A_n	B_n	R_{n-1}	S_n	R_n
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$S_n = (A_n \oplus B_n) \oplus R_{n-1}$ $R_n = R_{n-1}(A_n \oplus B_n) + A_n B_n$

3 Les soustracteurs

3.1 Demi-soustracteur

C`est un circuit capable de faire la soustraction de deux nombre binaire d`un bit chacun. Le circuit aura deux entrées A, B et deux sorties D, R

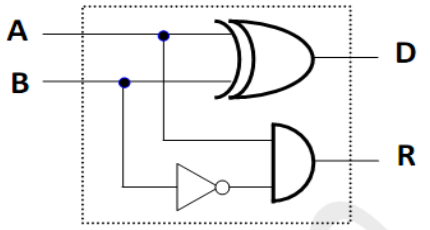
Table de vérité	Equation des sorties	Logigramme
<table><tr><th>A</th><th>B</th><th>D</th><th>R</th></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td><td>0</td></tr></table>	$D=\bar{A}B+A\bar{B}=A\oplus B$ $R=\bar{A}B$	

Tableau4.2 : Représentation Demi soustracteur

3.2 Soustracteur complet

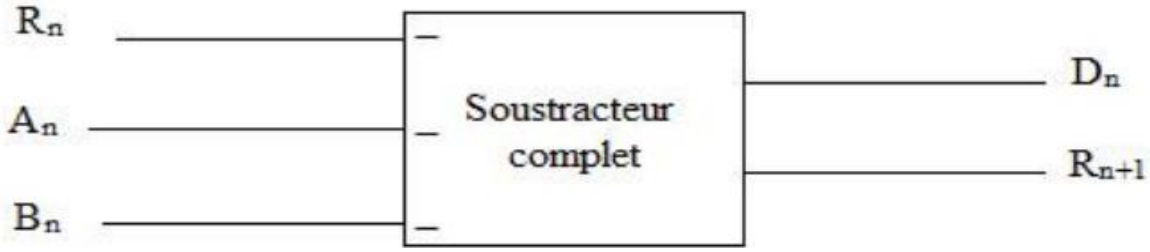


Figure4.5 : Schéma d'une soustraction complet

Où :

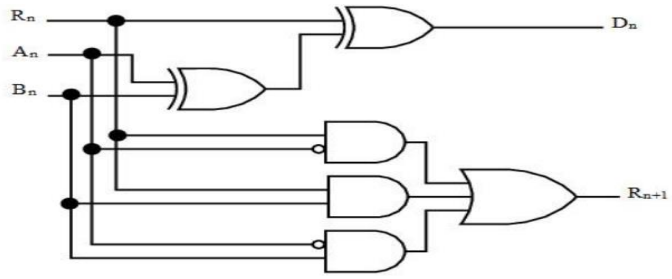
- A_n B_n sont les deux bits du rang n à soustraire
- R_n est une retenue engendrée de l'étage précédent qui doit être prise en considération dans la soustraction
- $D_n$est le résultat de l'opération de soustraction du rang n
- R_{n+1} est la retenue renvoyée vers l'étage suivant

Application :

Etablir la table de vérité d'un soustracteur 1 bit. Donner le logigramme de cette soustraction à l'aide des portes logiques de votre choix.

Pour remplir la colonne de la sortie D_n, pour chaque ligne de la table de vérité il faut appliquer l'équation suivante :$D_n = A_n - (B_n + R_n)$

Logigramme de soustraction complet



$D_n = (A_n \oplus B_n) \oplus R_n$

La table de vérité de soustraction complet

R_n	A_n	B_n	D_n	R_{n+1}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	0
0	1	1	0	0
1	0	0	1	1
1	0	1	0	1
1	1	0	0	0
1	1	1	1	1

$R_{n+1} = R_n(\overline{A_n \oplus B_n}) + \bar{A}_n B_n$

4 Les comparateurs

Ce sont des circuits combinatoires standards qui servent pour la comparaison de deux nombres binaires.

4.1 Principe de la comparaison

Soit $A = A_n A_{n-1} A_{n-2} \dots \dots \dots \dots . . A_1 A_0$

$B = B_n B_{n-1} B_{n-2} \dots \dots \dots \dots . . B_1 B_0$

Deux nombres binaires à comparer. Le processus commence par la comparaison des bits de poids fort :

Si $A_n > B_n$ alors $A > B$

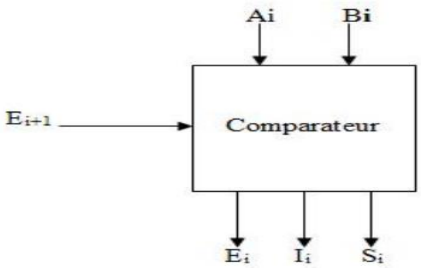
Si $A_n < B_n$ alors $A < B$

Si An = Bn alors il faut passer à la comparaison de $An - 1$ et $B\,n - 1$

On a alors besoin d'un comparateur élémentaire 2 bits

Si $E_{i+1} = 1$ alors la comparaison de A_i et B_i se fait normalement

Si $E_{i+1} = 0$alors toutes les sorties seront à zéro, le résultat de la comparaison est déjà donne par la comparaison des bits précédents.



Si $A_i > B_i$ alors $E_i = I_i = 0$ et $S_i = 1$.

Si $A_i < B_i$ alors $E_i = S_i = 0$ et $I_i = 1$.

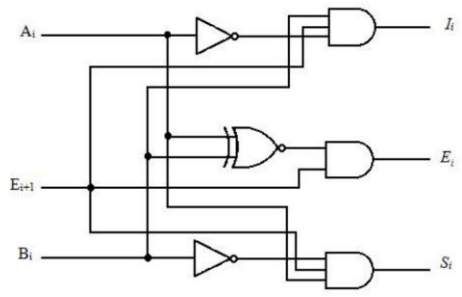
Si $A_i = B_i$ alors $I_i = S_i = 0$ et $E_i = 1$.

Application :

Etablir la table de vérité de ce comparateur et donner les équations de ses sorties avec leurs câblages.

Solution :

L'entrée E_{i+1} joue le rôle d'une entrée de validation pour le circuit. Si elle est égale à 0 le circuit reste bloqué.



Logigramme d'un comparateur

E_{i+1}	A_i	B_i	S_i	I_i	E_i
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	0	0	1
1	0	1	0	1	0
1	1	0	1	0	0
1	1	1	0	0	1

Tableau de vérité d'un comparateur

Les équations des sorties

$E_i = E_{i+1}\bar{A}_i\bar{B}_i + E_{i+1}A_iB_i = E_{i+1}(\bar{A}_i\bar{B}_i + A_iB_i) = E_{i+1}(A_i \otimes B_i)$

$I_i = E_{i+1}\bar{A}_iB_i$

$S_i = E_{i+1}A_i\bar{B}_i$

5 Les circuits de codage

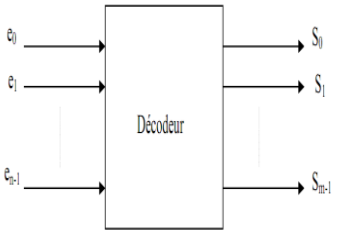
5.1 Le décodeur

Un décodeur est un circuit logique qui établit la correspondance entre un code d'entrée binaire de n bits et m lignes de sortie $(m \leq 2^n)$ Pour chacune des combinaisons possibles des entrées une seule ligne de sortie est validée.

Pour une combinaison binaire de n entrées => une seule ligne sera mise à 1

Exemple :

Supposons qu'on cherche à reconnaître le code binaire 1001, Dans ce cas, il faut réaliser un circuit qui implémente la fonction R = x3.x2.x1.x0.On remarque que R = 1 si l'entrée est 1001 et 0 sinon, ce circuit permet de décoder le code 1001.

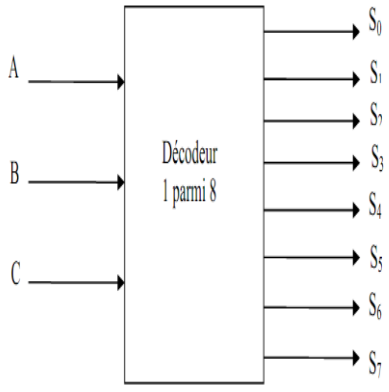


Exemples d'application :

Décodeur 1 parmi 8 :

C'est un circuit combinatoire à trois entrées et $2^3 = 8$ sortie

Avec $X_0 = A, X_1 = B, X_2 = C$



X0	X1	X2	sorties
0	0	0	S0
0	0	1	S1
0	1	0	S2
0	1	1	S3
1	0	0	S4
1	0	1	S5
1	1	0	S6
1	1	1	S7

Figure3.6 : Décodeur 1 parmi 8

Equation logique :

$$S_0 = \bar{A}\bar{B}\bar{C}\ ;\ S_1 = \bar{A}\bar{B}C\ ;\ S_2 = \bar{A}B\bar{C}\ ;\ S_3 = \bar{A}BC\ ;\ S_4 = A\bar{B}\bar{C}\ ;\ S_5 = A\bar{B}C\ ;\ S_6 = AB\bar{C}\ ;$$

$$S_7 = ABC$$

5.2 Décodeur DCB – Décimal (1 parmi 10) :

(Si) est activée si la valeur i est présente en binaire en entrée et elle est valide.

Si i est non valide => aucune sortie n'est activée.

Table de vérité :

A	B	C	D	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9
0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	0
0	0	1	1	0	0	0	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0	1	0	0	0	0	0
0	1	0	1	0	0	0	0	0	1	0	0	0	0
0	1	1	0	0	0	0	0	0	0	1	0	0	0
0	1	1	1	0	0	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	0	0	0	0	1

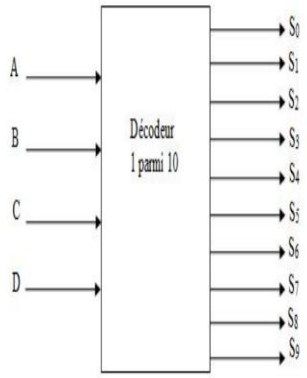


Figure4.7 :Decodeur 1 parmi 10

Les équations simplifiées des sorties de ce circuit sont :

$$S_0 = \bar{A}\bar{B}\bar{C}\bar{D}\,, S_1 = \bar{A}\bar{B}\bar{C}D, S_2 = \bar{A}\bar{B}C\bar{D}, S_3 = \bar{A}\bar{B}CD\,, S_4 = \bar{A}B\bar{C}\bar{D}\,, S_5 = \bar{A}B\bar{C}D\,, S_6 = \bar{A}BC\bar{D}, S_7 = \bar{A}BCD, S_8 = AB\bar{C}\bar{D}\,, S_9 = AB\bar{C}D$$

Exemple : Réalisation de fonctions logiques

A l'aide d'un décodeur approprié et des portes logiques, réaliser la fonction logique suivante :

$F = \bar{A}B\bar{C} + \bar{A}\bar{B}\bar{C} + AB\bar{C} + ABC$

Solution

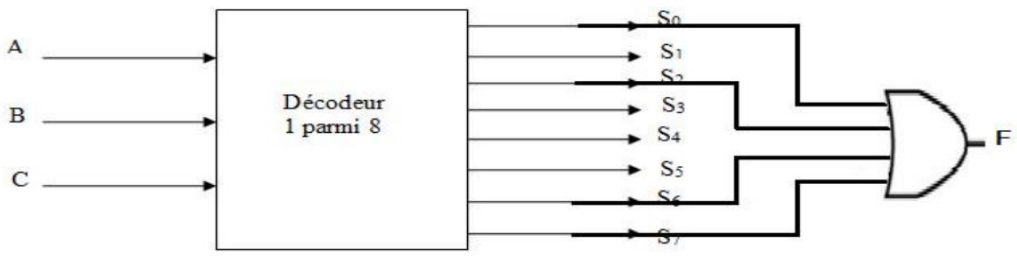
D'après l'équation de la fonction F, nous aurons besoin d'un décodeur à trois entrées

(A, B et C) ; donc on utilisera un décodeur 1 parmi 8.

Par identification avec les sorties d'un tel décodeur, on peut conclure que :

$F = S_2 + S_0 + S_6 + S_7$

Donc en obtient le logigramme suivant :



5.3 Codeur

Les Codeurs sont utilisés pour la compression des données, Le principe de fonctionnement d'un codeur est le suivant : lorsqu'une entrée est activée, les sorties affichent la valeur correspondant au numéro de l'entrée dans le code binaire choisi. Un codeur peut être vu comme un convertisseur du code décimal vers un code binaire.

5.3.1 Codeur 4 vers 2 :

C'est un codeur qui possède 4 entrées et deux sorties. Pour chaque entrée activée, son code binaire est affichée sur les sorties.

Exemple : Dressez la table de vérité de ce circuit et déduire les équations logiques simplifiées des différentes sorties.

Equations de soties :

$S_1 = A_2 + A_3$

$S_0 = A_1 + A_3$

Logigramme

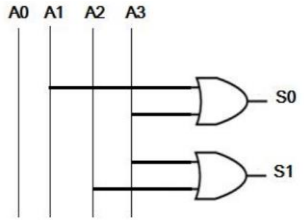


Table de vérité

A_3	A_2	A_1	A_0	S_1	S_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

Figure4.8 : Codeur 4 vers 2

5.4 Transcodeur :

Un **transcodeur** transforme une information disponible en entrée sous forme donnée (généralement un code) en la même information, mais sous une autre forme (généralement un autre code). Il existe trois types de transcodeurs :

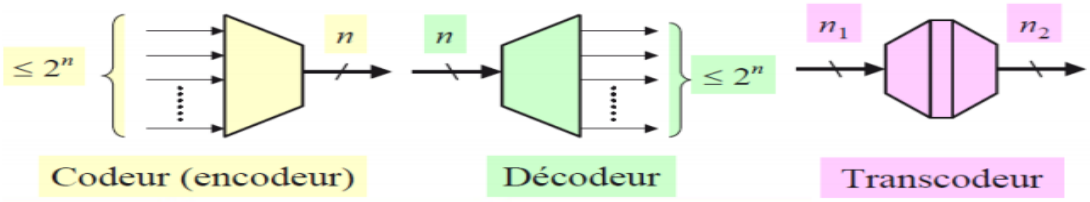


Figure4.9 :Transcodeur

Les deux plus importantes applications des transcodeurs sont : la conversion de code et l'affichage par segment.

- Transcodeur binaire Gray

Pour passer d'un code à un autre, on utilisera un convertisseur de code. A titre d'illustration nous allons étudier le transcodeur binaire Gray.

Exemple : Réaliser un transcodeur binaire/Gray à trois bits :

- Dresser sa table de vérité

- Donner les équations de ses sorties

Equation de sortie :

$X = A$

$Y = \bar{A}\,B + \bar{B}A = A \oplus B$

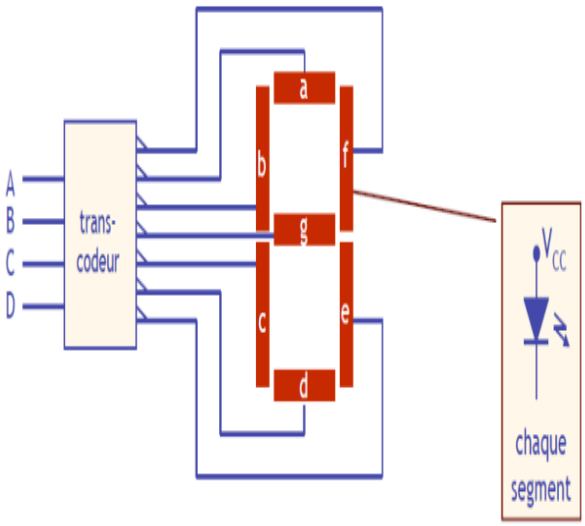
$Z = \bar{B}\,C + \bar{C}B = B \oplus C$

A	B	C	X	Y	Z
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	1	0	1
1	1	1	1	0	0

Figure4.10 : Table de vérité du transcodeur 3bits binaires –Gray

- Transcodeur BCD – 7 segments

Un domaine d'application considérable des transcodeurs est celui de la conversion de donné binaire en une forme se prêtant à un affichage numérique. Les dix chiffres 0 à 9 sont affichés au moyen d'un dispositif appelé afficheur à 7 segment lumineux qui sont des diodes électroluminescentes (D E L).les variables A, B, C, D sont écrites en BCD les variables de sortie a, b, c, d, e, f, g correspondent à chacun des segments de l'afficheur



Chiffres	ABCD	a	b	c	d	e	f	g
0	0000	1	1	1	1	1	1	0
1	0001	0	1	1	0	0	0	0
2	0010	1	1	0	1	1	0	1
3	0011	1	1	1	1	0	0	1
4	0100	0	1	1	0	0	1	1
5	0101	1	0	1	1	0	1	1
6	0110	1	0	1	1	1	1	1
7	0111	1	1	1	0	0	0	0
8	1000	1	1	1	1	1	1	1
9	1001	1	1	1	1	0	1	1

Figure4.11 : Table de vérité afficheur 7 segment

6 Les circuits d'aiguillage :

6.1 Le multiplexeur

Un multiplexeur est un circuit qui a pour rôle de faire circuler sur une seule voie les informations provenant de plusieurs sources.

D'une façon générale, un multiplexeur possède n entrées de commandes (d'adresses ou de sélection) qui permettent de sélectionner l'une des 2^n entrées de données possibles et de l'envoyer vers l'unique sortie.

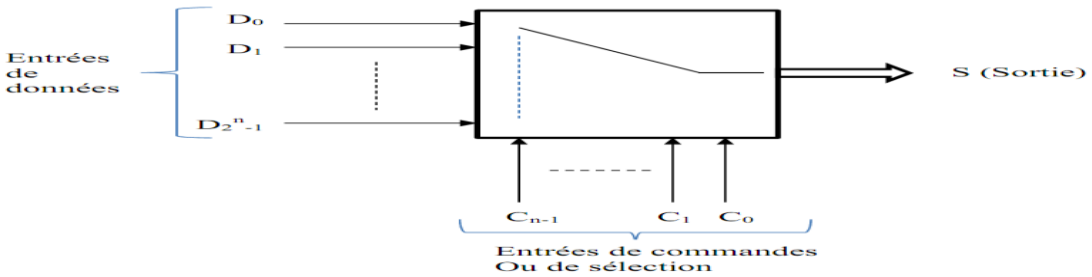


Figure4.12 :Multiplexeur

Exemple 01 :

1°/ Dresser la table de vérité d'un multiplexeur à 3 entrées de sélections et des entrées de données sur 1bit.

2°/ Etablir l'équation simplifiée de la sortie Z.

3°/ Etablir le logigramme de la sortie avec des portes logiques de votre choix.

Solution

Il possède :

2^n entrées d'information

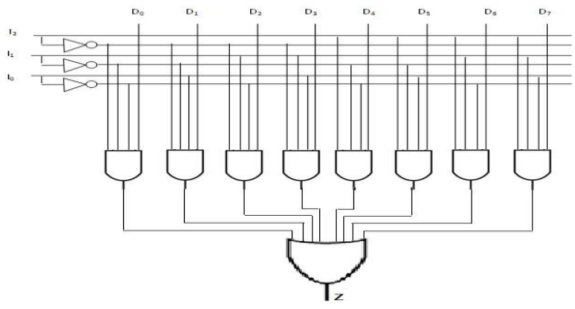
Une seule sortie

N entrées de sélection (commandes)

Un multiplexeur à trois entrées de commandes (I2, I1et I) possède 8 entrées de données (D0…D7)

$$Z = \overline{I_2}\overline{I_1}\overline{I_0}D_0 + \overline{I_2}\overline{I_1}I_0D_1 + \overline{I_2}I_1\overline{I_0}D_2 + \overline{I_2}I_1I_0D_3 + I_2\overline{I_1}\overline{I_0}D_4 + I_2\overline{I_1}I_0D_5 + I_2I_1\overline{I_0}D_6 + I_2I_1I_0D_7$$

Logigramme

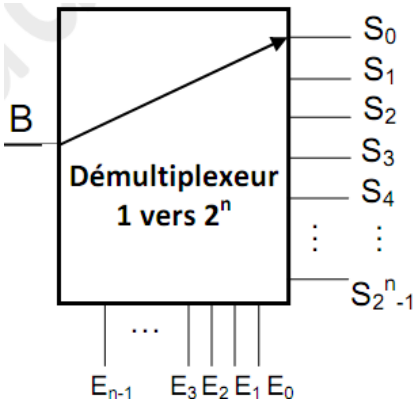


T .V : Mux 8vers 1

I_2	I_1	I_0	Z
0	0	0	D_0
0	0	1	D_1
0	1	0	D_2
0	1	1	D_3
1	0	0	D_4
1	0	1	D_5
1	1	0	D_6
1	1	1	D_7

6.2 Le Démultiplexeur

Un démultiplexeur (DEMUX) est un circuit logique qui possède une seule entrée **B**, **n** entrées de sélection (**E_0, E_1, E_2, … E_{n-1}**) et 2^n sorties (**S_0, S_1, S_2, … S_{2^n-1}**). Il est dit : **DEMUX 1 vers 2^n** ou **DEMUX 1 x 2^n**. Il effectue la fonction inverse d'un multiplexeur, il transmet la donnée d'entrée vers une des sorties selon le mot écrit aux entrées de sélection, il fonctionne comme un commutateur.



Circuit intégré :

4067 DEMUX 1 vers 16
74LS154 DEMUX 1 vers 16
74LS138 DEMUX 1 vers 8
74LS156 DEMUX 1 vers 4

Figure4.13 : Demultiplexeur

Exemple :

1- Dresser la table de vérité d'un démultiplexeur à deux entrées de sélection.
2- Etablir les équations de ses sorties et construire son logigramme.

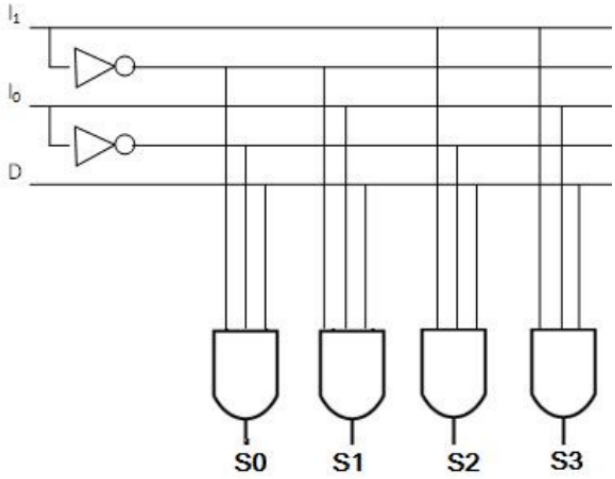
Solution

Equations des sorties :

$$S_0 = \bar{I}_1\bar{I}_0D \quad S_1 = \bar{I}_1I_0D \quad S_2 = I_1\bar{I}_0D \quad S_3 = I_1I_0D$$

Logigramme

T .V : DEMUX 1vers 4



I_1	I_0	S_3	S_2	S_1	S_0
0	0	0	0	0	D
0	1	0	0	D	0
1	0	0	D	0	0
1	1	D	0	0	0

TD 03
(Circuit Combinatoire)

Exercice : 01

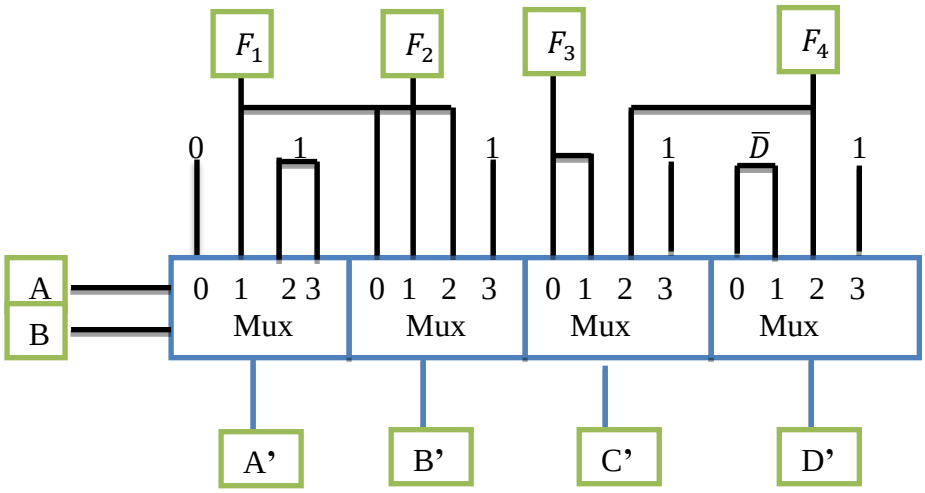
A l'aide d'une table de vérité, trouver les équations logiques des fonctions somme S et retenue R dans le cas de l'addition de 2 chiffres binaires. Réaliser ensuite les circuits logiques qui engendrent R et S :

1. Avec des portes AND, OR et NOT.
2. Avec des portes OU Exclusif et AND.
3. Avec des portes NAND a 2 entrées.

Exercice 2

Un transcodeur convertit un nombre ABCD code en binaire pur en nombre A'B'C'D' dans un autre code. Le logigramme de transcodeur est celui de la figure suivante, sachant que :

$F_1 = C + DF_2 = \bar{C}\bar{D}F_3 = F_2 + CDF_4 = C + \bar{D}$



1- Réaliser les fonctionsF_1, F_2, F_3, F_4 à l'aide de Mux a 2 voies avec D comme commande

2- Faire le logigramme général obtenu.

Exercice 3

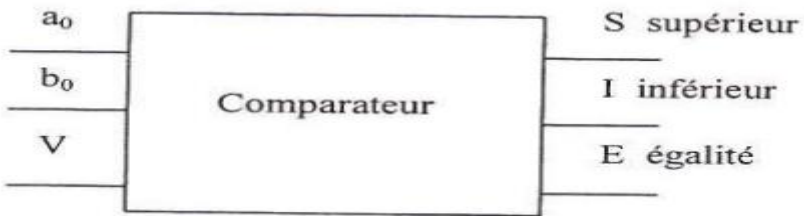
On souhaite réaliser un comparateur complet de deux mots de quatre bits.

1. Donner la table de vérité d'un comparateur de deux éléments binaires qui fonctionne façon suivante :

   S supérieur est égal 1

   I inférieur ou égal à 1

E=1 si les deux éléments binaires sont égaux.
L'entrée V est une entrée de validation, le comparateur fonctionne si V est égal à 1 si non toutes les sorties sont égales à zéro.



2. Réaliser ce comparateur de 1 élément binaire avec les portes de votre choix.
3. Réaliser le schéma d'un comparateur en cascade de deux mots de quatre bit A $(a_0a_1a_2a_3)$ et B $(b_0b_1b_2b_3)$à l'aide de comparateurs de 1 bit et de quelques portes logiques.

Exercice 4 :

Le Multiplexeur a 4 entre peut être utilise comme générateur de fonction logique avec 8 entrées (3 lignes d`adresse), on aura pour le signal de sortie

$S = \bar{A}.\bar{B}.\bar{C}.D_0 + \bar{A}.\bar{B}.C.D_1 \ldots\ldots\ldots\ldots\ldots.. + A.B.C.D_7$

Avec :D_0 ,D_1 , ……D_7 entrees du multiplexeur, et A,B,C ligne d'adresse du multiplexeur.

Réaliser à l`aide de multiplexeur a 8 entrées les fonctions logiques suivantes :

$S = \bar{X}\bar{Y}Z + \bar{X}Y\bar{Z} + \bar{X}YZ + XY\bar{Z}$

$F = AC + \bar{A}\bar{B} + AB$

$M = A\bar{B}\bar{C}\bar{D} + \bar{A}B\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + \bar{A}\bar{B}\bar{C}D + \bar{A}BCD + A\bar{B}CD + AB\bar{C}D + ABC\bar{D}$

Exercice 5

Deux pompe alimentent un même réservoir, deux contacts x et y lies a des flotteurs sont actionnes lorsque le niveau d`eau est inférieur à celui des contacts .le contact x est dessus de y, le fonctionnement dépend du niveau d`eau dans le réservoir.

1. Réservoir plein y=0, x=0, aucune pompe ne fonctionne.
2. Réservoir à moitié plein y=0, x=1, une seule pompe fonctionne.
3. Réservoir vide y=1, x=1, les deux pompes fonctionnent en même temps.

Un contacteur c a deux position, précise le numéro de pompe qui doit fonctionner seule : c=0(P1), c=1(P2).

1- Etablir la table de vérité de ces pompes.
2- Simplifiez P1 et P2 avec la table de karnaugh.
3- Réaliser le schéma logique de ces pompes puis le circuit électrique.

Solution

Exercice 01 :

Analyse : on a ′a′ et ′b′ les deux chiffres binaires S : la somme, R : le retenue

Table de vérité :

a	b	S	R
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

L'équation logique : S

	0	1
0	0	1
1	1	0

$S = \bar{a}b + a\bar{b}$

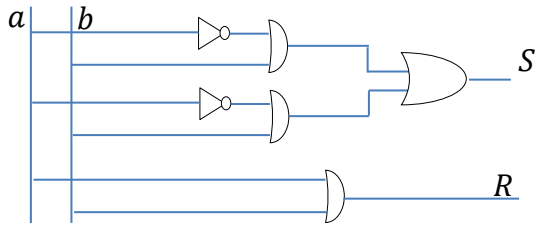
$S = a \oplus b$

l'équation logique R

	0	1
0	0	0
1	1	1

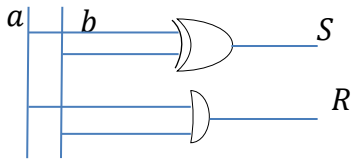
$R = a.b$

Circuit logique l'aide des portes : And,Ou,et Not



Circuit logique l'aide des portes : Ou exclusif, et And

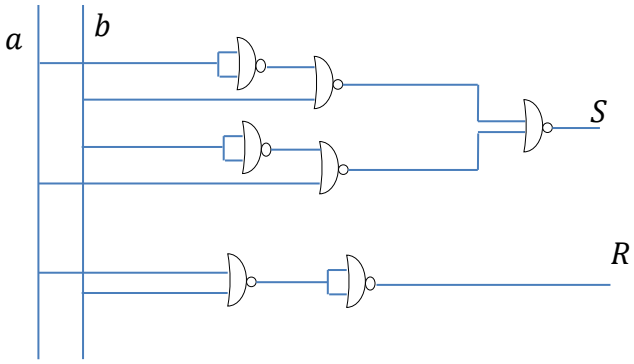
$S = a \oplus b \quad R = a.b$



Circuit logique l'aide des portes : Nand a 2 entres

$S = \bar{\bar{S}} = \overline{\overline{\bar{a}b + a\bar{b}}} = \overline{\overline{\bar{a}b}.\overline{a\bar{b}}}$

$R = \bar{\bar{R}} = \overline{\overline{a.b}}$



Exercice 2

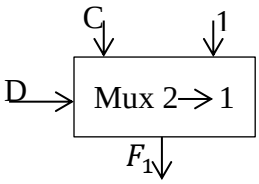
Réalisation des fonctions avec Mux à 2 voies

➢

$F_1 = C + D = C(D + \bar{D}) + D$

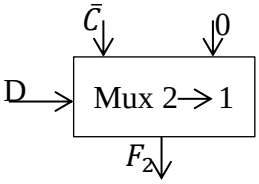
$F_1 = C.D + C.\bar{D} + D = D(C + 1) + C.\bar{D}$

$F_1 = D.1 + C.\bar{D}$



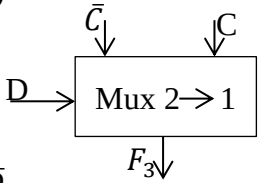
➢

$F_2 = \bar{C}\bar{D} = \bar{C}\bar{D} + D.0$



➢

$F_3 = F_2 + C.D = \bar{C}.\bar{D} + C.D$

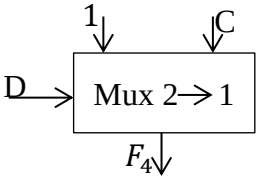


➢

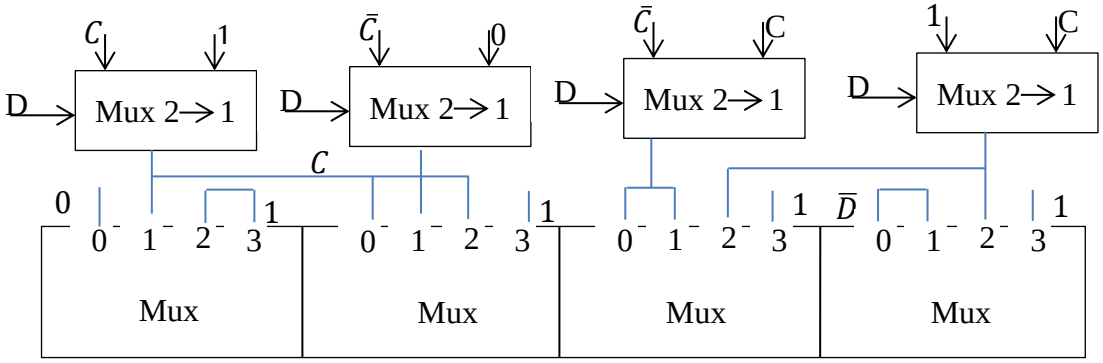
$F_4 = C + \bar{D} = C(D + \bar{D}) + \bar{D}$

$F_4 = CD + C\bar{D} + \bar{D} = CD + \bar{D}(1 + C)$

$F_4 = C.D + 1.\bar{D}$



2 logigramme générale obtenue



Exercice 3

TV d'un comparateur de 2 éléments binaire

V	a_0	b_0	S	I	E
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	0	0	1
1	0	1	0	1	0
1	1	0	1	0	0
1	1	1	0	0	1

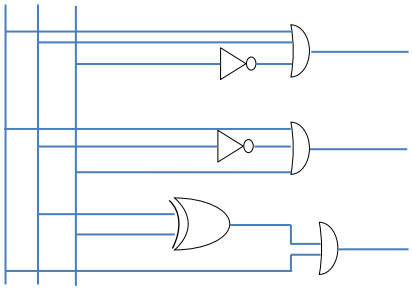
$S = Va_0\bar{b}_0$

$I = Vb_0\bar{a}_0$

$E = V\bar{a}_0\bar{b}_0 + Va_0b_0$

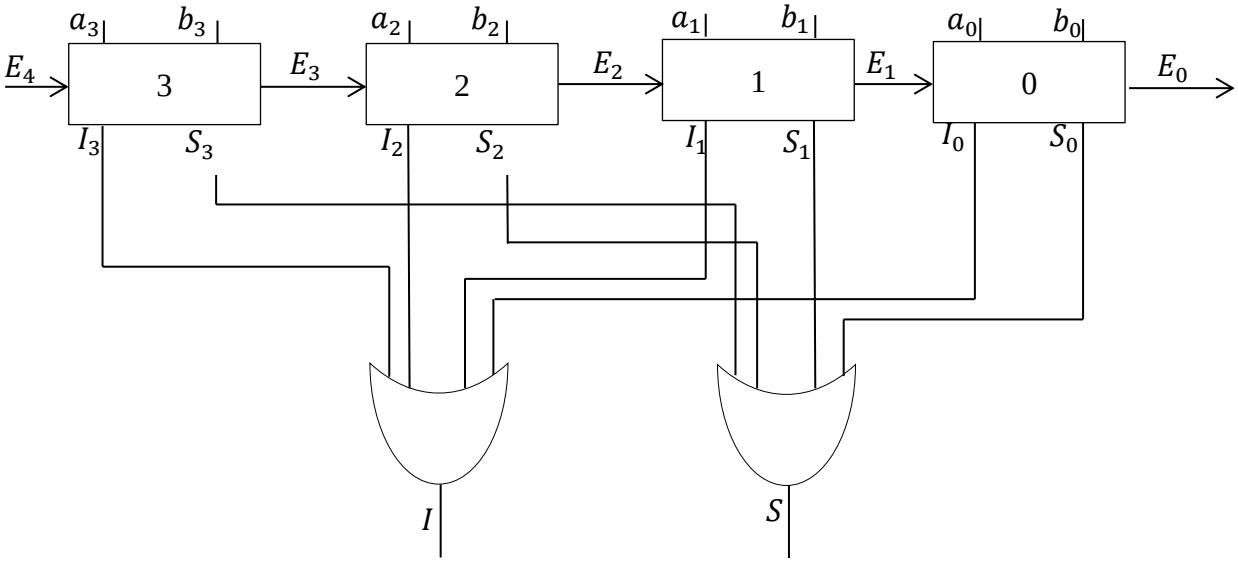
$E = V(\overline{a_0 \oplus b_0})$

1 comparateur d'éléments binaire (logigramme)

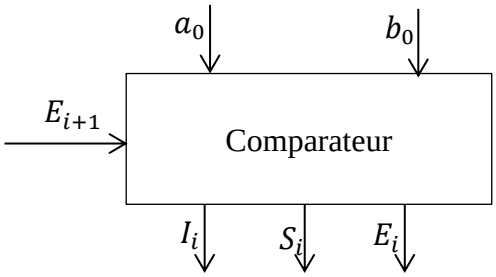


2 Comparateur en cascade de 2 Mots de 4 bits à l'aide d'un comparateur de 1 bit

$A(a_0, a_1, a_2, a_3)$ $B(b_0, b_1, b_2, b_3)$



Avec :



Exercice 4

Réalisation des fonctions logique l'aide d'un multiplexeur a 8 entrées

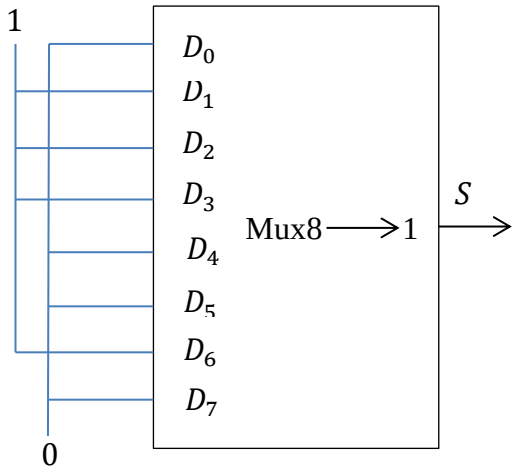
$S = \bar{x}\bar{y}z + \bar{x}y\bar{z} + \bar{x}yz + xy\bar{z}$

Table de vérité

x	y	z	D_i	S
0	0	0	D_0	0
0	0	1	D_1	1
0	1	0	D_2	1
0	1	1	D_3	1
1	0	0	D_4	0
1	0	1	D_5	0
1	1	0	D_6	1
1	1	1	D_7	0

Pour savoir la fonction S il faut attribuer aux entrées $D_0, D_1, D_2, D_3, D_4, D_5, D_6, D_7$ les valeurs suivant : $D_1 = D_2 = D_3 = D_6 = 1$

$D_0 = D_4 = D_5 = D_7 = 0$



- $F = AC + \bar{A}\bar{B} + AB$

1- Ecrire F à la 1ère forme canonique :

$F = AC(B + \bar{B}) + \bar{A}\bar{B}(C + \bar{C}) + AB(C + \bar{C})$

$F = ABC + A\bar{B}C + \bar{A}\bar{B}C + \bar{A}\bar{B}\bar{C} + ABC + AB\bar{C}$

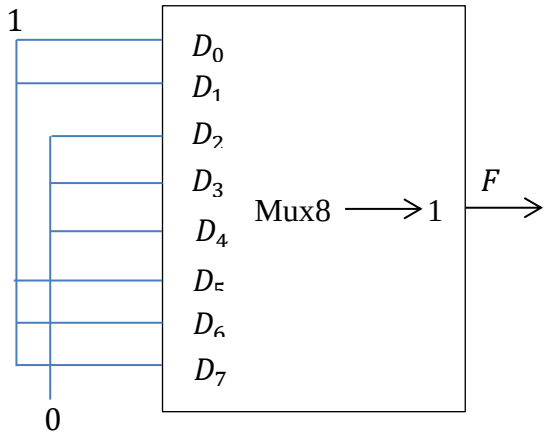
$F = \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + A\bar{B}C + AB\bar{C} + ABC$

$F = \bar{A}\bar{B}\bar{C}D_0 + \bar{A}\bar{B}CD_1 + A\bar{B}CD_5 + AB\bar{C}D_6 + ABCD_7$

Avec : $D_0 = D_1 = D_5 = D_6 = D_7 = 1$

$D_2 = D_3 = D_4 = 0$

A	B	C	D_i	F
0	0	0	D_0	1
0	0	1	D_1	1
0	1	0	D_2	0
0	1	1	D_3	0
1	0	0	D_4	0
1	0	1	D_5	1
1	1	0	D_6	1
1	1	1	D_7	1

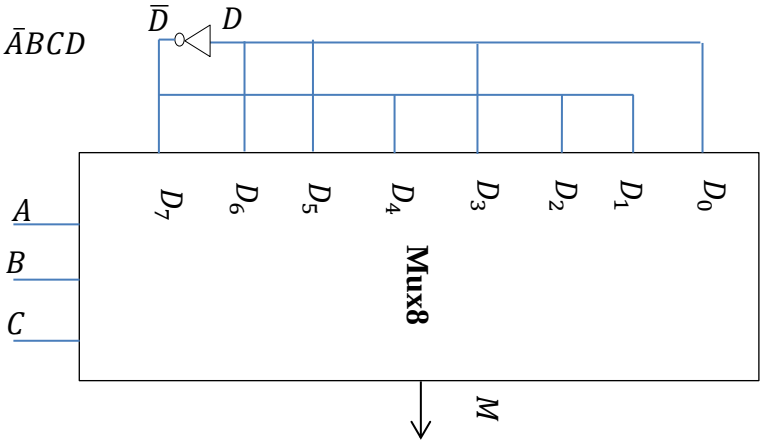


$M = A\bar{B}\bar{C}\bar{D} + \bar{A}B\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + \bar{A}\bar{B}\bar{C}D + \bar{A}BCD$

$+A\bar{B}CD + AB\bar{C}D + ABC\bar{D}$

$M = (A\bar{B}\bar{C} + \bar{A}B\bar{C} + \bar{A}\bar{B}C + ABC)\bar{D}$
$+(\bar{A}\bar{B}\bar{C} + \bar{A}BC + A\bar{B}C + AB\bar{C})D$

$D = D_0 = D_3 = D_5 = D_6$
$\bar{D} = D_4 = D_2 = D_1 = D_7$



Exercice 5

- pompes alimente un même réservoir
- niveau d'eau supérieur (réservoir plein $x = 0, y = 0$)
- niveau d'eau supérieur a x ($> x$) :
  Les deux pompes ne fonctionnement pas)
- niveau d'eau ($< y$): les deux pompes fonctionnement
- niveau d'eau ($< x, > y$): seul pompe fonctionne

$$\begin{cases} P_1 \text{fonctionne si: } c = 0 \\ P_2 \text{fonctionne si: } c = 1 \end{cases}$$

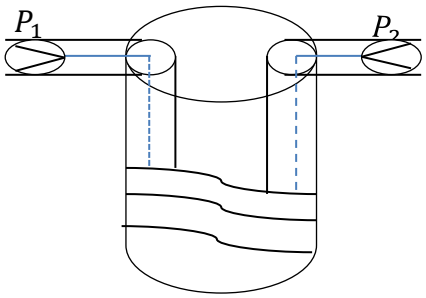


Table de vérité

y	x	c	P_1	P_2
0	0	0	0	0
0	0	1	0	0
0	1	0	1	0
0	1	1	0	1
1	0	0	X	X
1	0	1	X	X
1	1	0	1	1
1	1	1	1	1

Avec :
X : état indéterminée

$P_1 = \bar{y}x\bar{c} + yx\bar{c} + yxc$
$P_2 = \bar{y}xc + yx\bar{c} + yxc$

Tableaux de karnaugh

On peut utiliser les Φ (état indéterminé) dans les regroupements puisque les combinaisons correspondantes sont impossibles dans la réalité.

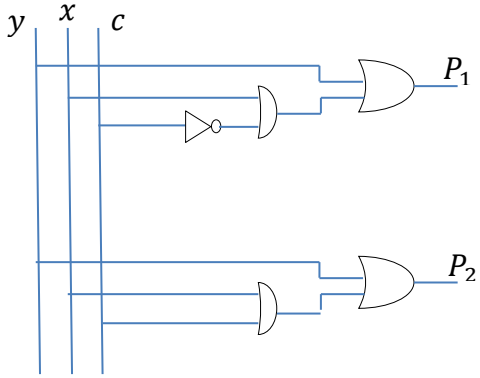
c \ yx	00	01	11	10
0	0	1	1	Φ
1	0	0	1	Φ

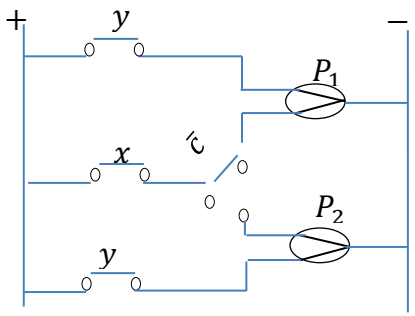
$P_1 = y + x\bar{c}$

c \ yx	00	01	11	10
0	0	0	1	Φ
1	0	1	1	Φ

$P_2 = y + xc$

Circuit logigramme et circuit électrique





Chapitre 5
Les Circuit Séquentiels

Chapitre 5: Les Circuits séquentiels

1 Introduction

Dans un circuit combinatoire, une sortie est uniquement fonction des entrées, pour les mêmes entrées on a toujours les mêmes sorties. Par contre, dans un circuit séquentiel, une sortie est une fonction des entrées mais aussi des sorties du circuit. Il y a rebouclage (rétroaction) des sorties sur les entrées. Cela signifie qu'un circuit séquentiel garde la mémoire des états passés.

Tout circuit séquentiel est constitue d'un circuit combinatoire couple à une mémoire comme indique par la figure suivante:

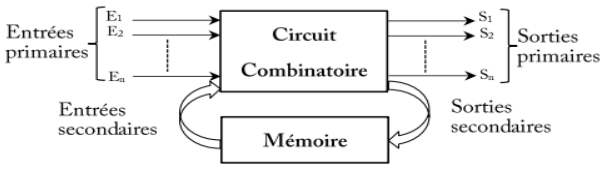


Figure 5.1 : Circuit séquentiel

Il existe deux grandes catégories de circuit séquentiel :

- **Le circuit séquentiel asynchrone** : Les sorties du montage peuvent changer à tout moment dès qu'une ou plusieurs entrées changent.
- **Le circuit séquentiel synchrone** : Le changement sur les sorties se produit après le changement d'état (front montant ou descendant) d'un signal maître (synchronise) nomme signal (Horloge) note H ou CLK.

2 Les Bascule asynchrone :

La bascule est le circuit de mémorisation le plus répandu, capable de stoker un bit et le restituer au temps nécessaire.

Elle est un système séquentiel constitue par une ou deux entrées et deux sorties complémentaire.

2.1 Bascule RS :

Une basculeRS comporte deux entrées: Une entrée R (Reset) de mise à zéro et une entrée S (Set) de mise à un. Schématise sous la forme suivante :

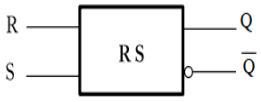


Figure 5 .2 : Bascule RS

Explication

Une impulsion sur **S (set)** → Mise à 1 de **Q** (marche)

Une impulsion sur **R (Reset)** → Mise à 0 de **Q** (Arrêt)

Entrées			Sorties		Mode de fonctionnement
R	S	Q_n	Q_{n+1}	Q_{n+1}	
0	0	0	0	1	Etat précèdent
0	0	1	1	0	Etat précèdent
0	1	0	1	0	Enclenchement
0	1	1	1	0	Maintien à 1
1	0	0	0	1	Maintien a 0
1	0	1	0	1	Déclenchement
1	1	0	-	-	Interdit
1	1	1	-	-	Interdit

Tableau5.1 : Table de vérité de la bascule RS

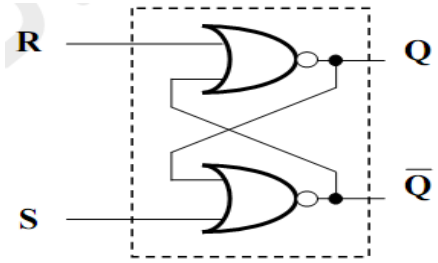


Figure 5.3 : Logigramme bascule RS

L'état R=S=1 est un état interdit puisqu'il nous donne le deux sorties complémentaires Q et $\bar{Q}$ au même état ce qui n'est pas logique.

L'équation caractérisée (simplifier) de la bascule RS est :$Q_{n+1} = \bar{R}Q_n + S$

2.2 Bascule D :

La bascule D possède une seule entrée Schématise sous la forme suivante :

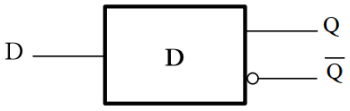


Figure 5 .4 : Bascule D

Explication

Un appui sur **D** → Mise à 1 de **Q**

Un relâchement de **D** → Mise à 0 de **Q**

Entrées		Sorties		Mode de fonctionnement
D	Q_n	Q_{n+1}	Q_{n+1}	
0	0	0	1	Maintien à 0 : μ_0
0	1	0	1	Déclenchement : δ
1	0	1	0	Enclenchement : ε
1	1	1	0	Maintien à 1 : μ_1

Tableau5.2 : Table de vérité de la bascule D

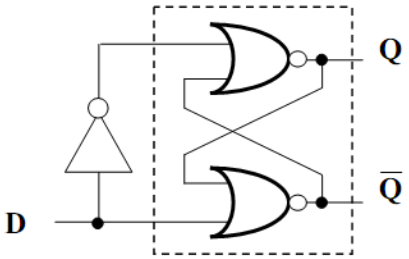


Figure 5.5 : Logigramme bascule D

L'équation caractérisée de la bascule D est :$Q_{n+1} = D$

Remarque : En mettant **S=D et R=D** dans l'équation de la bascule **RS** on aura :

$\boldsymbol{Q_{n+1} = DQ_n + D = D(1 + Q_n) = D.}$

Ainsi on obtient une bascule **D** en rajoutant un inverseur entre **S** et **R**.

2.3 Bascule JK :

Contrairement à la bascule **RS**, la condition **J=K=1**, ne donne pas lieu à une condition indéterminée, mais par contre la bascule passe à l'état opposé.

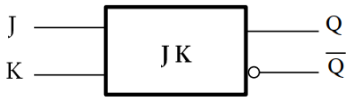


Figure 5 .6 : Bascule JK

Entrées			Sorties		Mode de fonctionnement
J	K	Q_n	Q_{n+1}	$\overline{Q_{n+1}}$	
0	0	0	0	1	Etat précèdent
0	0	1	1	0	Etat précèdent
0	1	0	0	1	Maintien à 0 : μ_0
0	1	1	0	1	Déclenchement : δ
1	0	0	1	0	Enclenchement : ε
1	0	1	1	0	Maintien à 0 : μ_1
1	1	0	1	0	Enclenchement : ε
1	1	1	0	1	Déclenchement : δ

Tableau5.3 : Table de vérité de la bascule JK

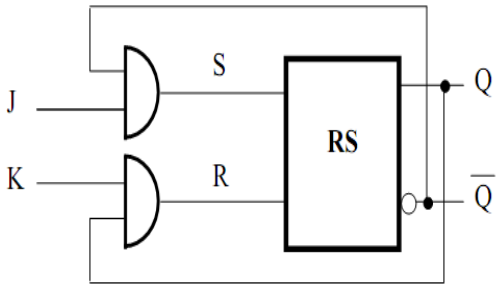


Figure 5.7 : Logigramme bascule JK

L'équation caractérisée de la bascule JK est :$Q_{n+1} = J\bar{Q}_n + \bar{K}Q_n$

2.4 Bascule T :

La bascule **T** est obtenue en reliant les entrées **J** et **K** d'une bascule **JK**

Figure 5 .8 : Bascule T

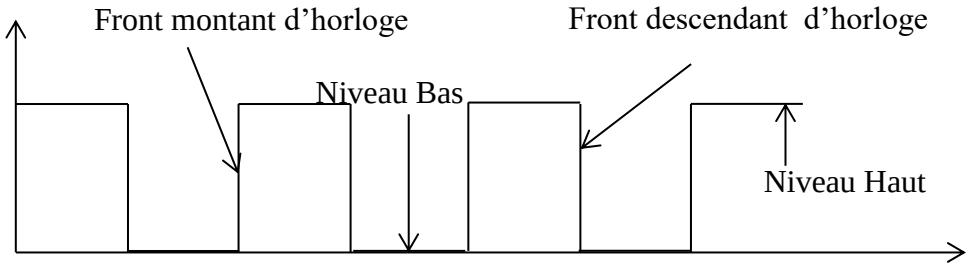
Entrées		Sorties		Mode de fonctionnement
T	Q_n	Q_{n+1}	$\overline{Q_{n+1}}$	
0	0	0	1	Maintien à 0 : μ_0
0	1	1	0	Maintien à 1 : μ_1
1	0	1	0	Enclenchement : ε
1	1	0	1	Déclenchement : δ

Tableau5.4 : Table de vérité de la bascule T

L'équation caractérisée de la bascule T est :$Q_{n+1} = \bar{T}Q_n + \bar{Q}_n T = T \oplus Q_n$

3 Les Bascule synchrone

Les sorties d'une bascule synchrone se produit après le changement d'état (front montant ou descendant) d'un signal maître (synchronise), dit signal d'horloge note H ou CLK (voir figure ci-dessous)



3.1 Synchronisation sur niveau bas :

Dans le cas du niveau bas :

- ✓ Si H=1 : Q maintient son état
- ✓ Si H=0 : Q la bascule est fonctionné normale

3.2 Synchronisation sur niveau haut

Dans le cas du niveau haut :

- ✓ Si H=1 : la bascule fonctionne en mode normale, les sorties obéissent aux entrées.
- ✓ Si H=0 : la sortie maintient son état, quelle que soit les valeurs des entrées.

3.3 Synchronisation sur front

Tous les changements d'état se font sur un front d'impulsion fixé (montant ou descendant), le passage du niveau bas vers le niveau haut s'appelle front montant et le passage du niveau haut vers niveau bas s'appelle front descendant.

3.4 Bascule JK sur front montant

Symbole :

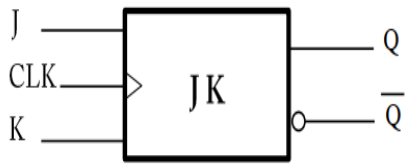


Table de vérité

Entrées			sorties		Mode de Fonctionnement
CLK	J	K	Q_{t+1}	$\overline{Q_{t+1}}$	
↑	0	0	Q_t	$\overline{Q_t}$	Etat précèdent
↑	0	1	0	1	Déclenchement Mise a 1
↑	1	0	1	0	Enclenchement Mise a 0
↑	1	1	$\overline{Q_t}$	Q_t	Changement d'état

Figure 5 .9 : bascule JK synchrone

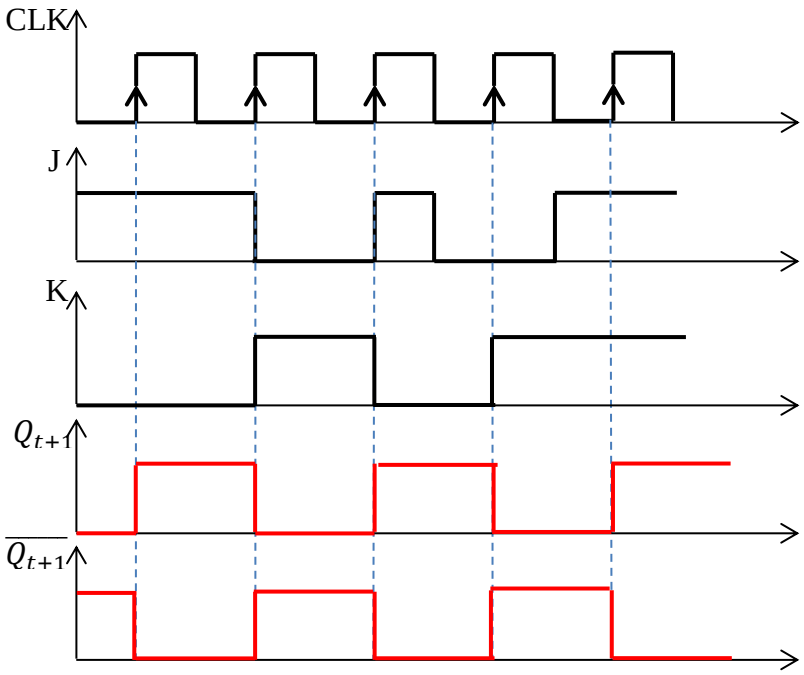


Figure 5.10 : Chronogramme de la bascule JK synchronisé sur front montant

3.5 Bascule D sur front montant

La bascule D synchrone est déclenchée par le signal d'horloge H. L'unique entrée D (Data) détermine l'état de la bascule. La sortie Q prend la même valeur que celle présente à l'entrée D quand le signal d'horloge effectue une transition.

Symbole :

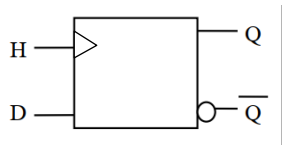


Figure 5.11 : Bascule D synchrone

Table de vérité

Entrées		Sorties	
H	D	Q_{t+1}	$\overline{Q_{t+1}}$
↑	0	0	1
↑	1	1	0

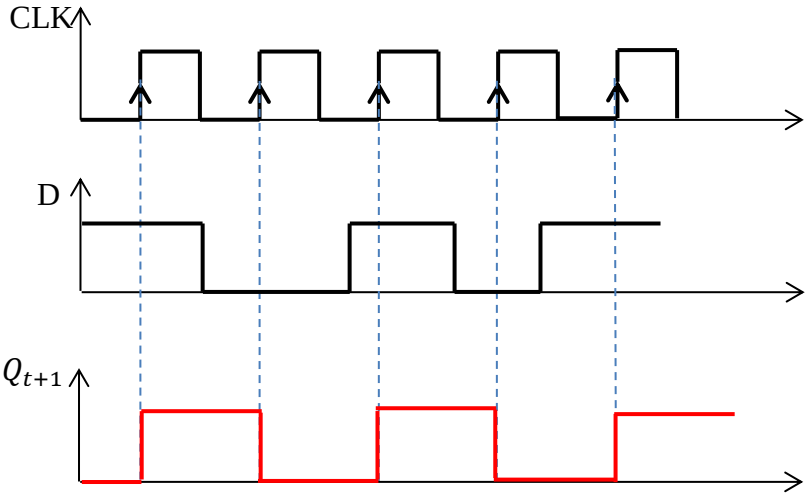


Figure 5.12 : Chronogramme de la bascule D synchronisé sur front montant

3.6 Bascule T sur front montant

La bascule T synchrone est déclenchée par le signal d'horloge H. L'unique entrée T (Trigger) commande l'état de la bascule. La sortie Q change d'état chaque fois que l'entrée T passe à l'état logique 1 et conserve son état le reste du temps.

Symbole :

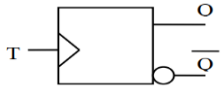


Figure 5.13 : Bascule T synchrone

Table de vérité

Entrees	Sorties
H	Q_{t+1}
$\uparrow$	$\overline{Q_t}$

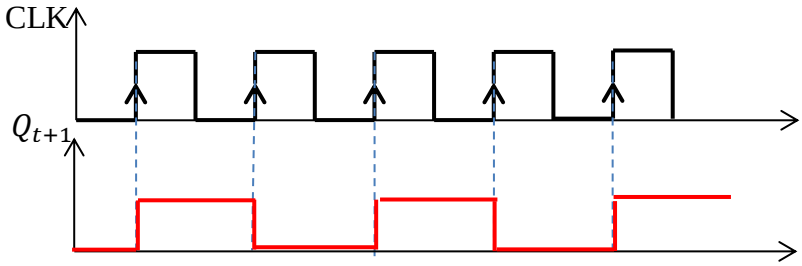


Figure 5.14 : Chronogramme de la bascule T synchronisé sur front montant

3.7 Bascule D de structure Maitre Esclave

La bascule D de structure Maitre Esclave est constituée de deux bascules D à verrouillage (ou latch) placées l'une à la suite de l'autre. La première est appelée Maitre, la seconde est appelée Esclave.

- ✓ Si CLK=1 : la première bascule fonctionne et la deuxième bloque
- ✓ Si CLK=0: la première bascule est bloqué et la deuxième fonctionne
- ✓ Les deux bascules fonctionnent ensemble au moment de passage de CLK de 1 à 0 c'est-à-dire au moment du front descendent.

Symbole :

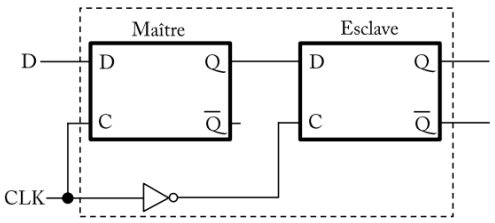


Table de vérité

Entrees		Sorties	
D	CLK	Q_{t+1}	$\overline{Q_{t+1}}$
X	0	Q_t	$\overline{Q_t}$
X	1	Q_t	$\overline{Q_t}$
0	$\uparrow$	0	1
1	$\uparrow$	1	0

Figure 5.15 : Bascule D Maitre Esclave déclenchement Sur front descendant d'horloge

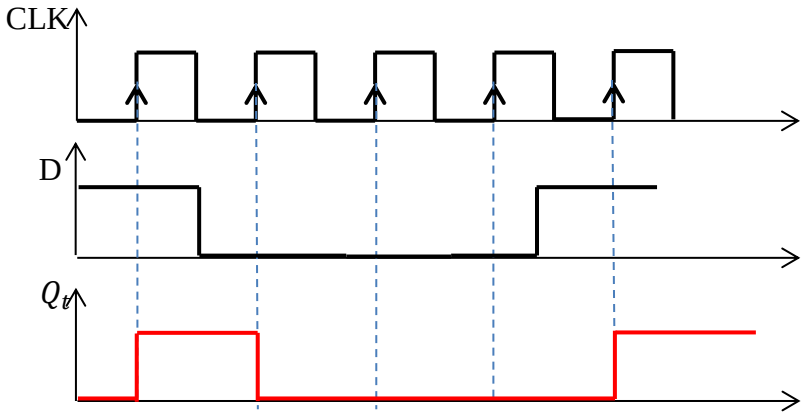
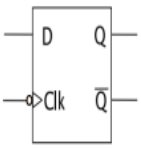


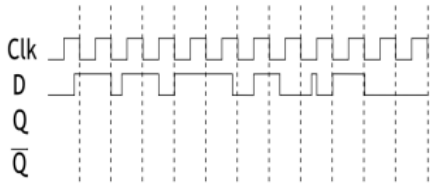
Figure 5.16 : Chronogramme de la bascule maitre esclave D activée par front montant

TD : 04
(Les Bascules)

Exercice 01 :

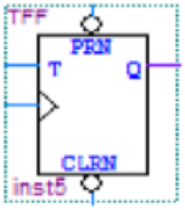
1- **D**onner le chronogramme correspondant aux sorties complémentaires Q et $\bar{Q}$ de la bascule D



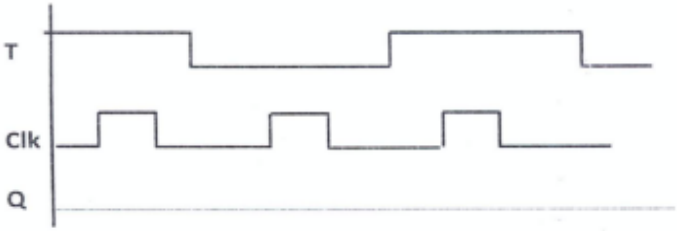


2- On considère la bascule T, ayant la table de vérité suivante. Concevoir cette bascule à l'aide de la bascule D et une porte logique.

Entrée T	État futur (Sortie Q^+)
0	Q
1	$\bar{Q}$

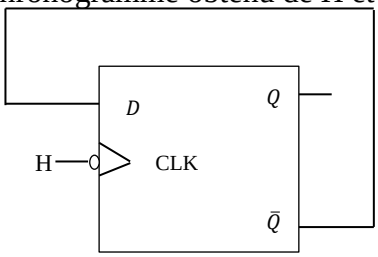


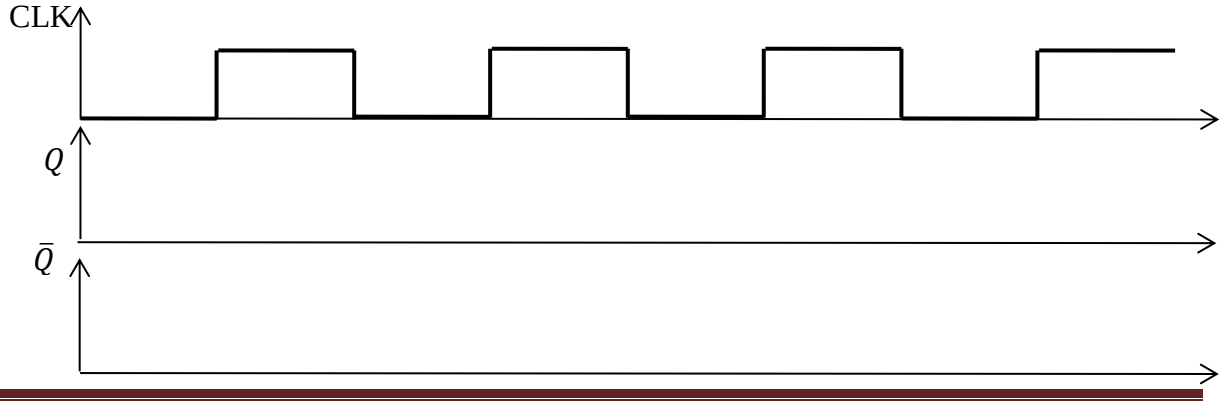
3- Compléter le chronogramme ci-après.



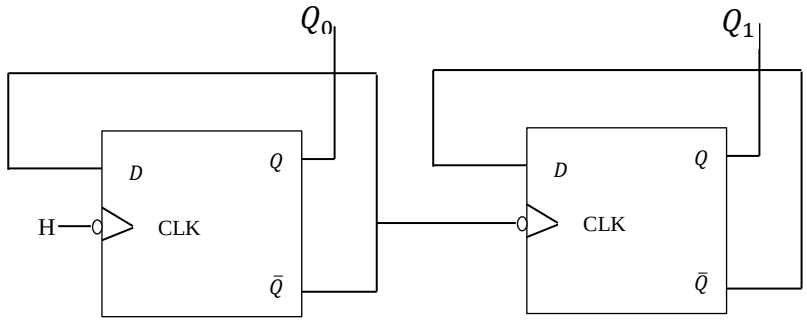
Exercice 2 :

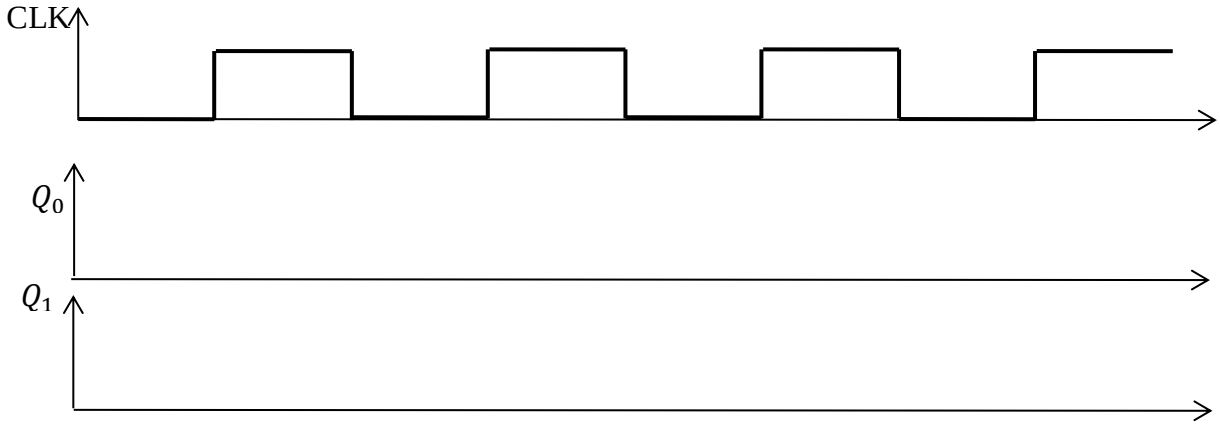
1- Tracer le chronogramme obtenu de H et Q de la figure suivante





2- Quelle est la fréquence f_Q de Q par rapport à la fréquence f_H de H
3- Tracer les chronogrammes de , Q_0 et Q_1 pour la figure suivante :





4- Quelle est la fréquence f_{Q_1} de Q_1 par rapport à la fréquence f_H de H

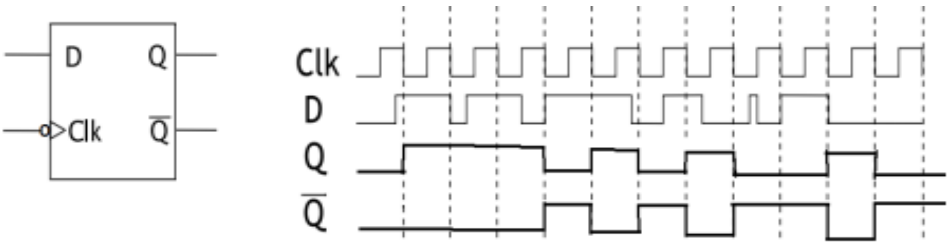
Exercice 3

Réaliser une bascule JK synchronisée en front montant d'horloge à l'aide d'un MUX 8 vers 1 et le minimum des portes logique.

Solution

Exercice 1

Les chronogrammes correspondant aux sorties complémentaires Q et $\bar{Q}$ de la bascule D, sachant que cette dernière réagit au front descendant, sont comme suit :



Rappelons la table de vérité associée à la bascule T. Cette TV peut être encore réarrangée de la façon portée au tableau 1. En ce qui concerne la bascule la bascule D, sa TV est également

Clk	Q	Q^+	T
↑	0	0	0
↑	0	1	1
↑	1	0	1
↑	1	1	0

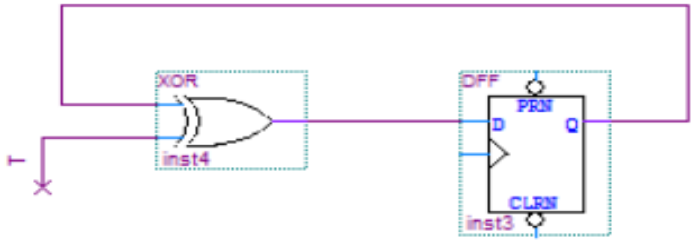
TABLE 1 –

rappelée dans le tableau 2. Les deux TV sont équivalents (fonctions équivalentes) ; on a alors

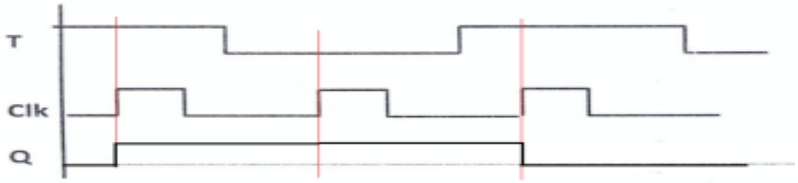
Clk	D	Q	Q^+	$D \oplus Q$
↑	0	0	0	0
↑	0	1	0	1
↑	1	0	1	1
↑	1	1	1	0

TABLE 2 –

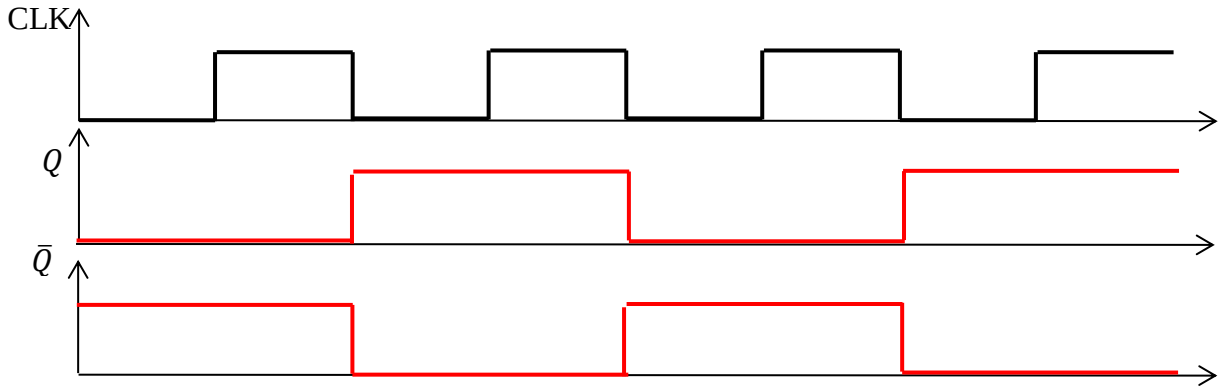
$T = D \oplus Q$, autrement écrit : $D = T \oplus Q$. La Figure suivant présente le schéma équivalent de la bascule T, obtenu à partir d'une bascule D.



Le chronogramme en question est ci-après.



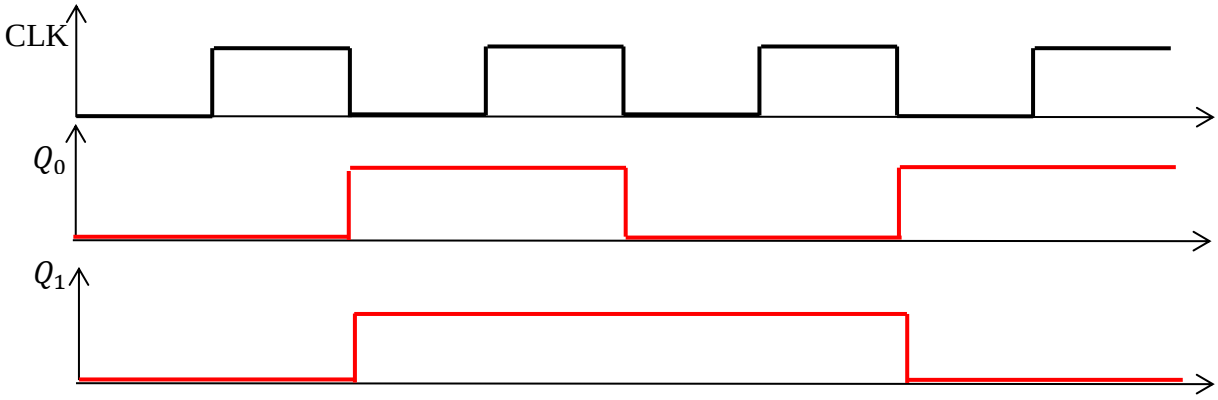
Exercice 2



2- La fréquence de f_Q par rapport f_H

$T_H = 2T_Q$ donc : $f_Q = {f_H}/{2}$ ou $f_H = 2f_Q$

3-



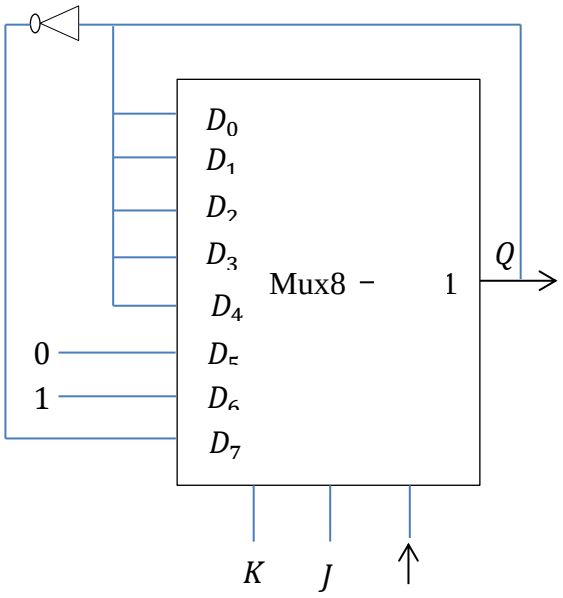
4- La fréquence de f_Q par rapport f_H

$T_{Q_0} = 2T_H$ et $T_{Q_1} = 2T_{Q_0}$ donc : $T_{Q_1} = 4T_H$

Donc : $f_{Q_1} = {f_H}/{4}$ ou $f_H = 4f_{Q_1}$

Exercice 3

Front	J	K	Q_{t+1}	D_i
0	0	0	Q_t	$D_0 = Q_t$
0	0	1	Q_t	$D_1 = Q_t$
0	1	0	Q_t	$D_2 = Q_t$
0	1	1	Q_t	$D_3 = Q_t$
1	0	0	Q_t	$D_4 = Q_t$
1	0	1	0	$D_5 = 0$
1	1	0	1	$D_6 = 1$
1	1	1	$\overline{Q_t}$	$D_7 = \overline{Q_t}$



Chapitre 6
Les Compteurs

Chapitre 6 : Les Compteurs

1 Introduction

Le comptage d'événements est une opération indispensable dans de nombreux automatismes dès lors que ces événements peuvent être traduits par des impulsions électriques.

Les compteurs sont composes d'un ensemble de bascules montée en série ou en parallèle. Avec une bascule synchrone, soit de type$T$,D ou JK .en distingues deux types :

- Compteurs-décompteurs asynchrones.
- Compteurs-décompteurs synchrones.

2 Compteurs et décompteurs asynchrones :

Dans la réalisation des compteurs asynchrones, les bascules ne changent pas d'état en même temps .Techniquement, si on utilise des bascules JK, on les place en cascade en fixant toutes les entrées J et K égales à 1. La première bascule qui contient le bit de poids le plus faible reçoit l'entrée de l'horloge. Pour les autres bascules, la sortie de chaque bascule sert de signal d'horloge pour la bascule suivante.

2.1 Compteurs asynchrones modulo 2^n

Pour construire un compteur modulo N qui compte de 0 jusqu'à $(N-1)$, on cherche le nombre n de bascules tel que : $2^n > N$

Par exemple :

- pour un compteur octal qui compte de 0 jusqu'à 7 ; il faut 3 bascules.
- pour un compteur décimal qui compte de 0 jusqu'à 9 ; il faut 4 bascules.

Exemple : Compteur modulo 8 à bascule D synchronisées en front montant

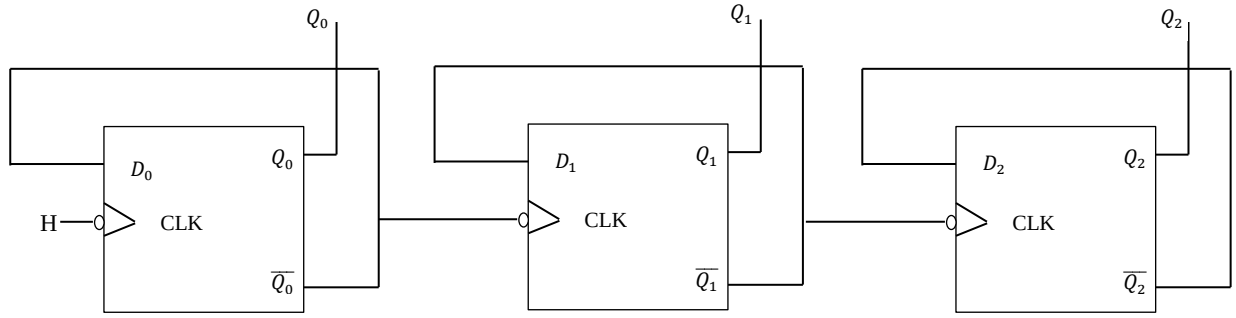


Figure 6.1 : Compteur Modulo 8 à bascule D synchronisés

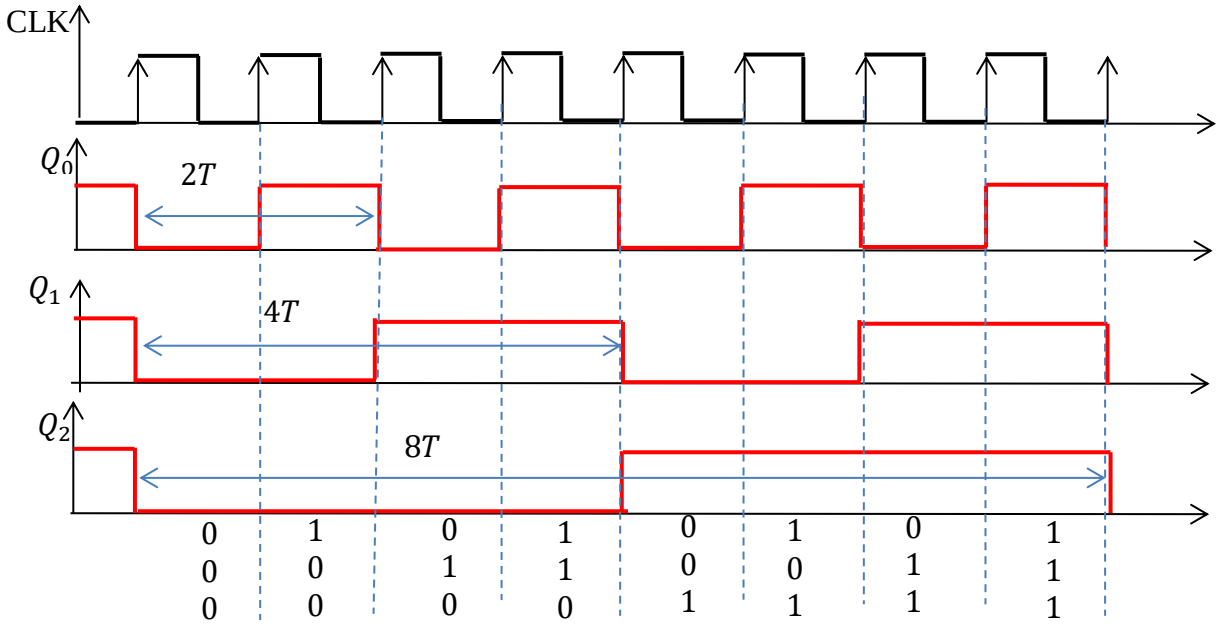


Figure 6.2 : Chronogramme compteur Modulo 8 à bascule D synchronisés

2.2 Décompteur asynchrone modulo 8 : (Avec front montant)

On à $\mathbf{2^3} = 8$ donc il nous faut 3 bascules pour la réalisation de ce décompteur asynchrone modulo 8.

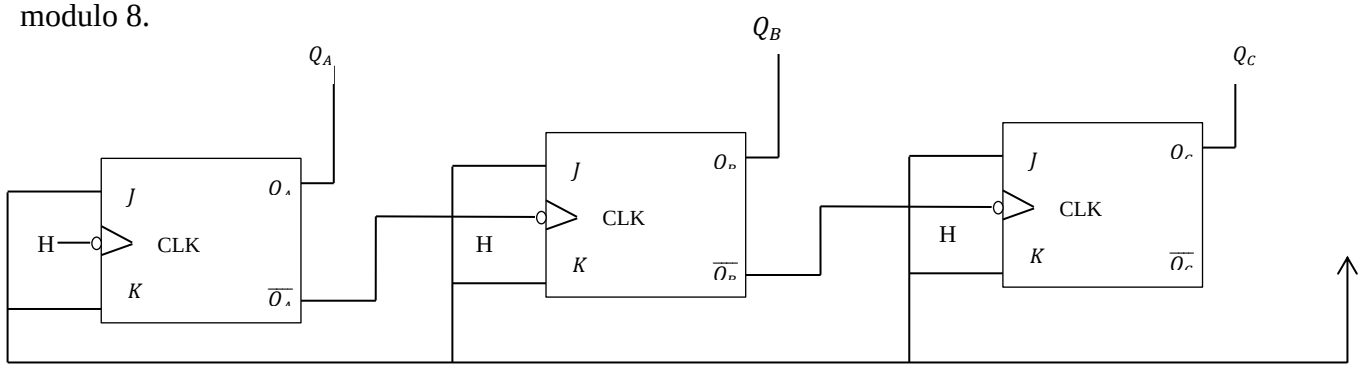


Figure 6.3: Décompteur Asynchrone modulo 8

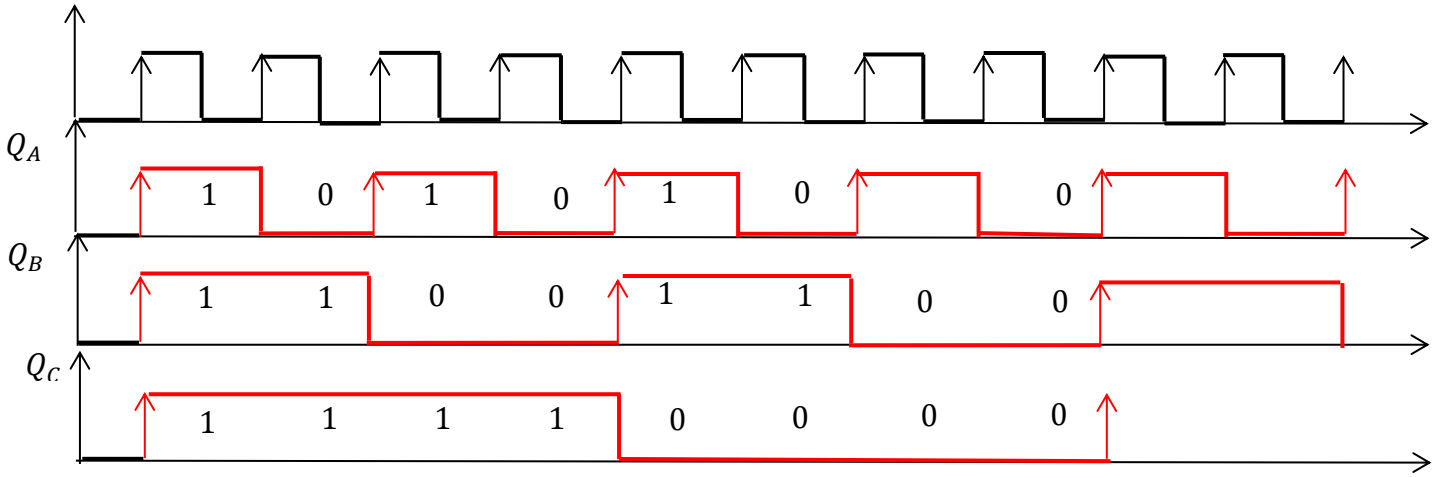


Figure 6.4 : Chronogramme du décompteur asynchrone modulo 8

3 Compteur synchrone modulo :

Dans ce cas on doit commander par le signal d'horloge toutes les bascules en mêmes temps. Les états des bascules JK, par exemple, dépendent des entrées J et K. pour un compteur modulo2^3, par exemple, nécessitant donc trois bascules JK (C,B ,A).

La premier bascule (A) change donc l'état a chaque impulsion d'horloge.la deuxième bascule (B) change d'état tous les deux impulsions d'horloge. Son état dépend de (A).la troisième bascule(C) dépend de (A), (B).

Exemple : le circuit logique et chronogramme d'un compteur synchrone modulo 8, fonctionnant sur front montant sont donne dans les figures suivante :

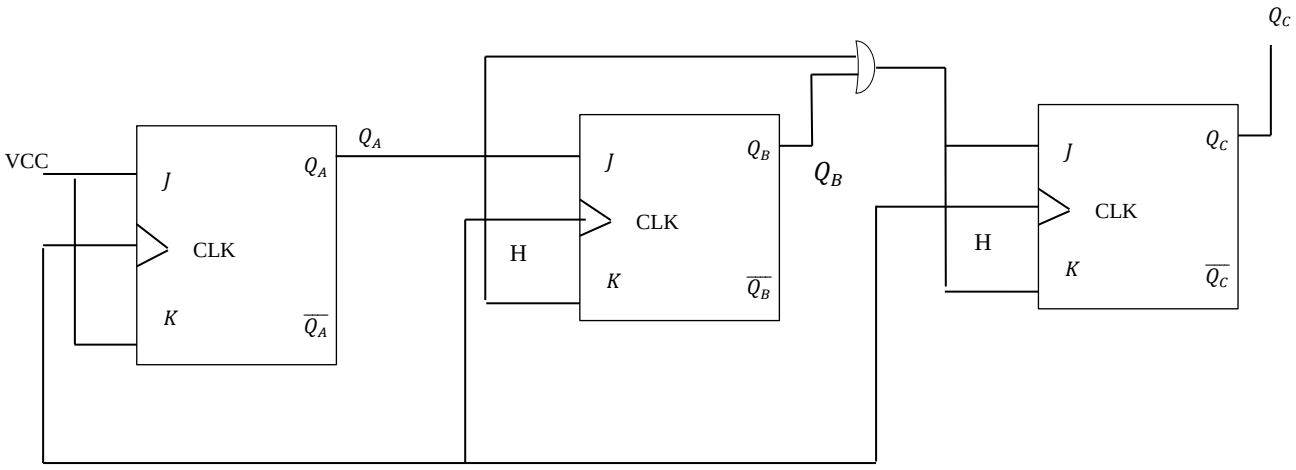


Figure 6.5 : Compteur synchrone modulo 8

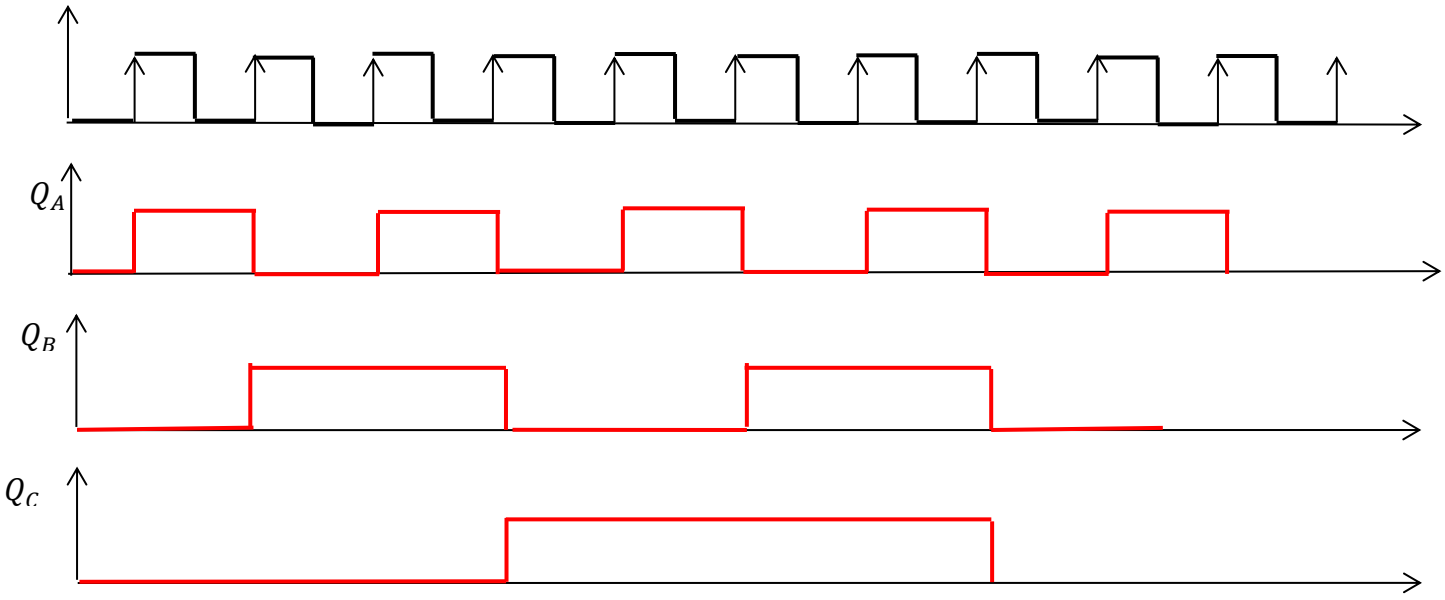


Figure 6.6 : Chronogramme Compteur synchrone modulo 8

Chapitre 7
Les Registres

Chapitre 7 : Les Registres

1 Introduction

Un registre est un circuit séquentiel synchrone, capable de stocker temporairement des informations binaires en utilisant un ensemble de bascules.

2 Définition :

Un registre est un circuit séquentiel synchrone, constitué de n bascules permettant déstocker temporairement un mot (une information) binaire de n bits dans l'objectif de son transfert dans un autre circuit (pour affichage, m´mémorisation, traitement, etc.).

Les registres sont classés par :

- Des registres à entrées parallèles et sorties parallèles : PIPO (Parallel IN-Parallel OUT).
- Des registres à entrées parallèles et sorties séries : PISO (Parallel IN-Serial OUT).
- Des registres à entrées séries et sorties parallèles : SIPO (Serial IN- Parallel OUT).
- Des registres à entrées séries et sorties séries : SISO (Serial IN- Serial OUT).

Il y a donc deux types de registres : les registres de mémorisation et les registres de décalage.

2.1 Registre de mémorisation :

Un registre de mémorisation (ou registre de données) est un registre dans lequel les différents étages sont indépendants les uns des autres, cependant certains signaux agissent sur l'ensemble des étages ; tel que remise à 0 et remise à 1.

Exemple Registre de mémorisation 4 bits

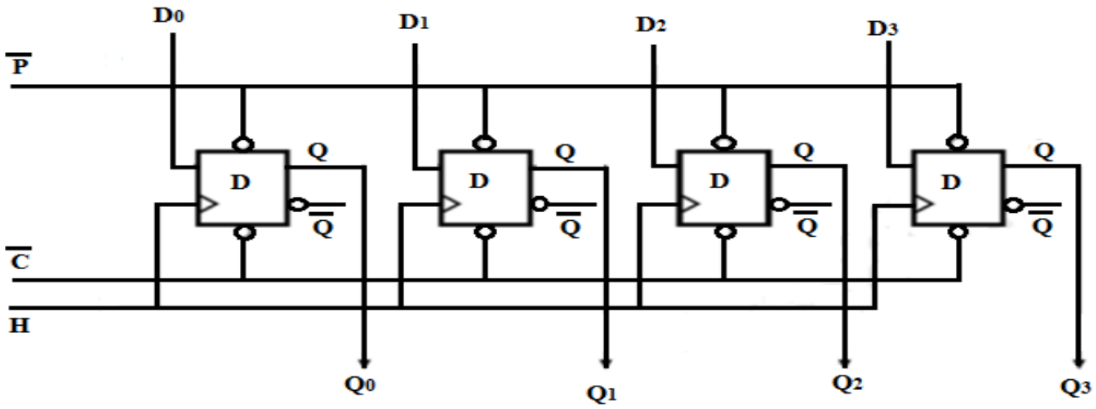


Figure 7.1 : Registre de mémorisation 4 bits

Dans l'exemple ci-dessous, les 4 bascules sont chargées en parallèle et lues en parallèle en synchrone avec le signal d'écriture H. Ce type de registre est appelé aussi registre PIPO.

La figure suivante présenté le schéma fonctionnel d'un registre type PIPO

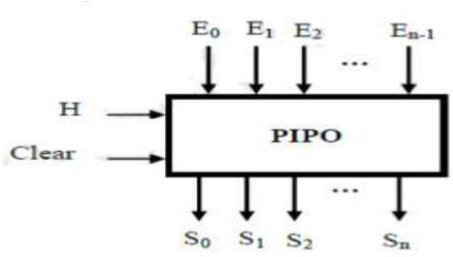


Figure 7.2 : Schéma fonctionnel type PIPO

2.2 Registre à décalage

Parmi les caractéristiques des registres à décalage sont :

- Les trois types de décalage : à droite, à gauche, ou dans ces deux directions par une commande de mode (registre bidirectionnels).
- Deux types de chargements des registres : parallèle ou série .en parallèle on entre le nombre dans les bascule d'un seul coup. en série, on entre le premier bit du nombre dans la première bascule, de gauche ou de droite, on le décale en même temps que l'on entre le deuxième bit etc
- Les deux types de sortie des registres, parallèle ou série.
- Les nombre de bit dans les registres est de quatre, cinq ou huit

2.2.1 Registre à décalage à gauche :

Son objectif est de reproduire à la sortie de la bascule de rang i l'état logique de la sortie de la bascule de rang $i + 1$, tout en appliquant le même signal d'horloge à toutes les bascules.

Il s'agit d'un registre possédant une seule entrée à droite E_{dr} et n sorties **(**Q_1 **,** Q_2 **…….** Q_n **)**

2.2.1.1 Registre à décalage à gauche a 4 bascules D :

Solon la caractéristique de décalage à gauche $Q_i^+ = Q_{i+1}$ et de la caractéristique de la bascule D $Q_i^+ = D_i$, on déduit que $D_i = Q_{i+1}$ donc les équations logiques correspondant sont :

$$\begin{cases} D_1 = Q_2 \\ D_2 = Q_3 \\ D_3 = Q_4 \\ D_4 = E_{dr} \end{cases}$$

Le schéma du registre a décalage à gauche a 4 bascules D

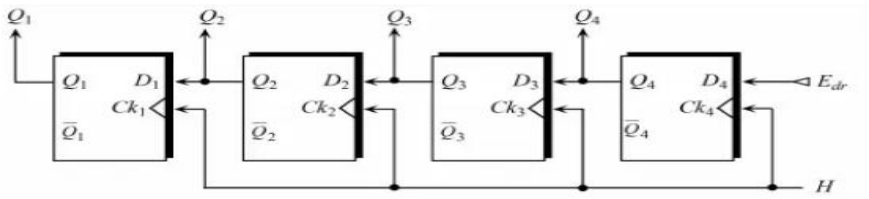


Figure 7.3 : Registre à décalage à gauche a 4 bascules D

2.2.2 Registre à décalage à droit :

2.2.2.1 Registre à décalage à droite à 4 bascules D :

Solon la caractéristique de décalage à droite $Q_i = Q_{i+1}^+$ et de la caractéristique de la bascule D $Q_{i+1}^+ = D_{i+1}$, on déduit que $D_{i+1} = Q_i$ donc les équations logiques correspondant sont :

$$\begin{cases} D_4 = Q_3 \\ D_3 = Q_2 \\ D_2 = Q_1 \\ D_1 = E_{dr} \end{cases}$$

Le schéma du registre à décalage à gauche à 4 bascules D a front montant est représenté

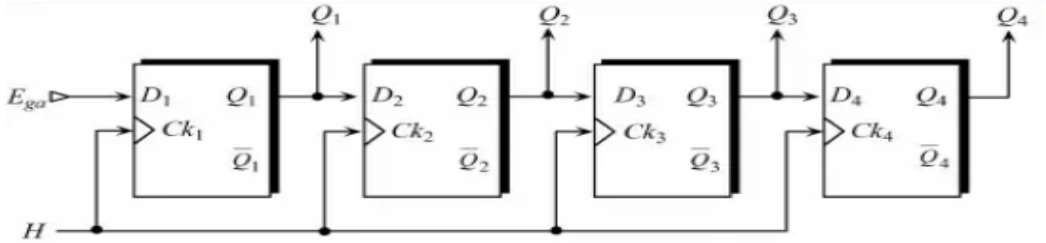


Figure 7.4 : Registre à décalage à droite a 4 bascules D

2.2.3 Registre à décalage à gauche ou à droite :

Dans ce cas en rajoutant en entrée de sélection S offrant la possibilité de choisir le sens de décalage.

2.2.3.1 Registre à décalage à gauche ou à droite à 4 bascules D :

Il est caractérisé par l'entrée de sélection logique S, tel que :

- S=0 décalage à gauche
- S=1 décalage à droite

Les équations logiques correspondant sont :

$$\begin{cases} D_1 = \bar{S}Q_2 + SE_{ga} \\ D_2 = \bar{S}Q_3 + SQ_1 \\ D_3 = \bar{S}Q_4 + SQ_2 \\ D_4 = \bar{S}E_{dr} + SQ_3 \end{cases}$$

Le schéma du registre à décalage à gauche ou à droite à 4 bascules D à front montant est représenté.

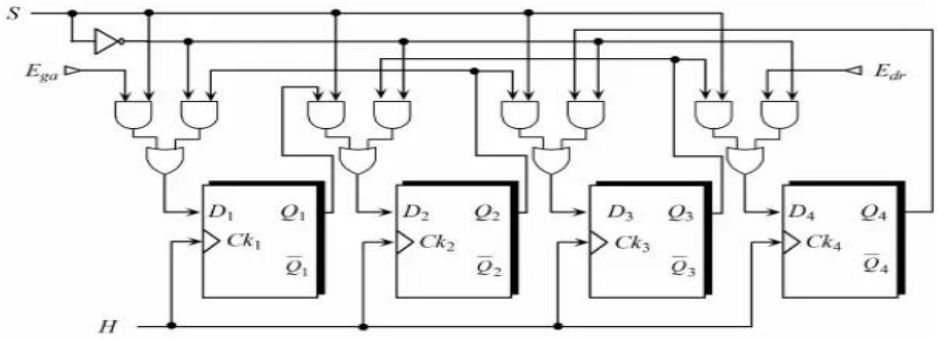


Figure 7.5: Registre à décalage à gauche ou à droite a 4 bascules D

Bibliographie

[1] : WILLIAM I.FLETCHER ;An Engineering Approach To Digital Design,Utah State University Logan ,Utah.

[2] : JEAN LETOCHA; Introduction Aux Circuits Logique, McGraw-Hill, Editeurs.

[3]: H.Curry,Combinatory Logic II , North-Holland,1972 .

[4]: PHILLIPE DARCH, Architecture des Ordinateurs : Logique Booléenne et Implémentation Technologique. Edition VUIBERT, 2004.

[5] : JEAN JACQUES MERCIER, Introduction après instruction: Logique Séquentielle Circuits asynchrones et synchrones, Ellioses, 2008.

[6] : CLAUDE BRIE , Informatique Industrielle :Logique Combinatoire et Séquentielle, méthode ,outils et réalisation, Ellipses,2003.

[7] : A.LAHRECH, Cours d'électronique Numérique, Circuit Séquentiels 2022.

[8] : Hassen BOUZGOU, Cours logique combinatoire et séquentielle Université Batna.

[9] : Hocine Amimeur, logique combinatoire et séquentielle, polycopie de cours, Faculté de Technologie, Département de Génie Electrique, Université de Bejaia, 2016.

[10] : M. Sbaï, Electronique numérique, logique combinatoire et composants numérique, Ellipses Editions Marketing, Paris, France, 2013.

[11] : VILLEMEJANE. J. cours du codage des nombres à la logique séquentielle, Institut Universitaire de Technologie de creteil-vitry. Département de Génie Électrique et Informatique Industrielle. Année universitaire 2013-2014