TD N°2 : Fonctions logiques

EXERCICE 1

Trois interrupteurs A, B et C commandent l'allumage de deux lampes R et S suivant les conditions suivantes :

- Dés qu'un ou plusieurs interrupteurs sont activés la lampe R doit s'allumer.
- La lampe S ne doit s'allumer que si au moins 2 interrupteurs sont activés.

Déterminer les expressions des fonctions binaires R et S et Donner le système logique correspondant (à l'aide de circuits NAND) .

EXERCICE 2

Trois personnes K, J et T commandent le lancement d'une fusée. Chacune dispose d'un commutateur à deux positions permettant le lancement. Suivant une étude psychologique du stress de ces personnes, le lancement aura lieu aux conditions suivantes :

- Si K, J et T ont leurs commutateurs sur la position active.
- Ou si K et J ont leurs commutateurs sur la position active et T sur la position non active.
- Ou si le commutateur de K est sur la position active et les commutateurs de J et T sont sur la position non active.
- Ou si le commutateur de K est sur la position non active et les commutateurs de J et T sont sur la position active.

1) Etablir la table de vérité correspondante.

2) Donner le schéma non simplifié du circuit de commande.

3) Déterminer à l'aide d'un tableau de Karnaugh l'expression simplifiée de la sortie, ainsi que le schéma correspondant. Conclure.

EXERCICE 3

Chacune de 3 alarmes A, B et C met un fil à la masse (potentiel bas 0 volt ; A, B et C actifs à l'état logique 0).

Réaliser un système logique qui permet (à l'aide de circuits NAND) :

- d'allumer la lampe L (L=1) quand il existe une seule alarme (alarme mineure).
- De déclencher une sonnerie S (S=1) quand il existe au moins deux alarmes (alarme majeure).

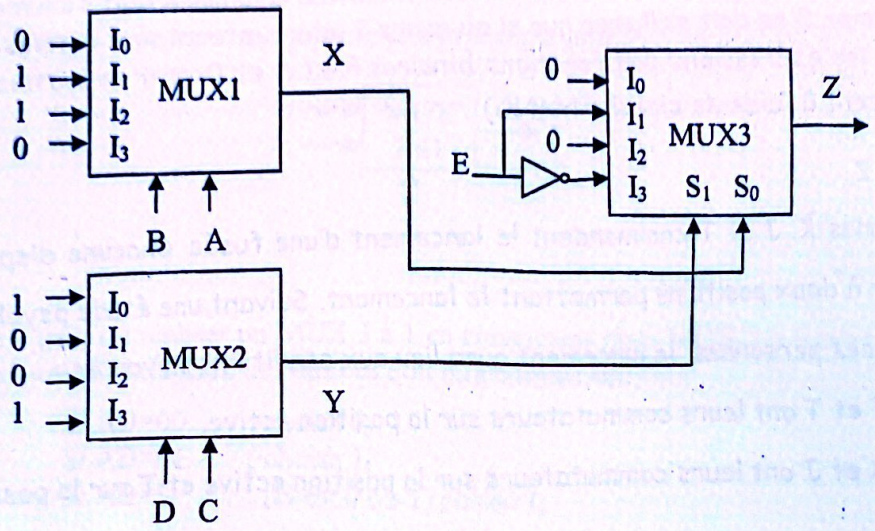
Pour cela on envisagera deux cas :

Scanned with CamScanner

TD2 : Logique Combinatoire

EXERCICE 1: Multiplexeur

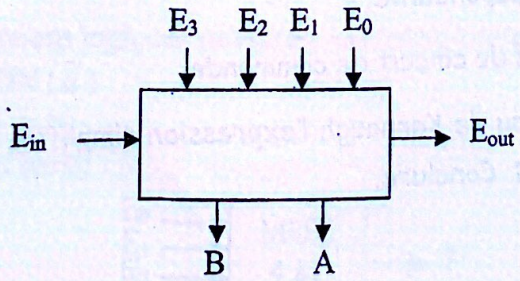
Soit le montage suivant réalisé à partir de trois multiplexeurs : MUX1, MUX2 et MUX3. Chacun de ces multiplexeurs possède quatre entrées.



1) Déterminer l'expression de X en fonction de A et B.
2) Déterminer l'expression de Y en fonction C et D.
3) Déduire l'expression de Z en fonction de A, B, C, D et E.

EXERCICE 2: Décodeur

Soit un circuit combinatoire à cinq lignes d'entré et trois lignes de sortie comme le montre la figure ci-dessous.



Le fonctionnement est le suivant :

- Lorsqu'une seule ligne d'entrée parmi E_0, E_1, E_2, E_3 se trouve au niveau haut, son numéro est codé en binaire sur les sorties BA.
- Si plusieurs lignes sont simultanément au niveau haut, le circuit code le numéro le plus élevé.
- Si toutes les lignes d'entrée sont au niveau bas, le circuit code BA = 00 mais on signale par $E_{out} = 1$ que ce code n'est pas validé. Dans tous les autres cas, $E_{out} = 0$.
- Le fonctionnement décrit jusqu'ici s'observa lorsque $E_{in} = 1$.
  Si $E_{in} = 0$ on a B=A= $E_{out} = 0$.

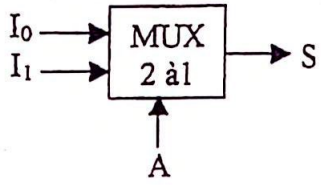
Scanned with CamScanner

Exercice n°1

1. Donner le logigramme d'un demi-additionneur (à base de portes NAND à 2 entrés).
2. Comment connecter 2 demi-additionneurs pour avoir un additionneur complet. Donner le logigramme correspondant (à base de portes NAND à 2 entrés).

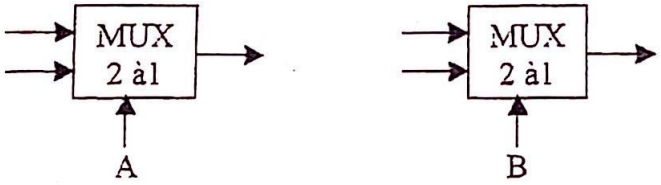
Exercice n°2

1. Ecrire l'équation logique et donner le circuit logique interne d'un MUX 2 à 1.



2. comment peut on réaliser un MUX 3 à 1 en connectant deux MUX 2 à 1 et sans ajouter des portes logiques. La sélection des entrées doit être comme suit :

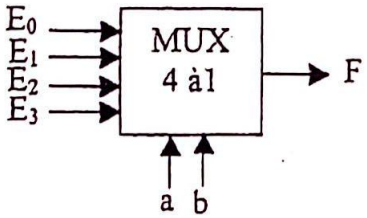
Si AB=00 ; choisir I_0
Si AB=01 ; choisir I_1
Si AB=1- ; (B est 0 ou 1) choisir I_2



Exercice n°3

1. réaliser les fonctions logiques suivantes, chacune à l'aide d'un MUX 4 à 1

- NON ($\bar{a}$)
- ET (a.b)
- OU (a+b)

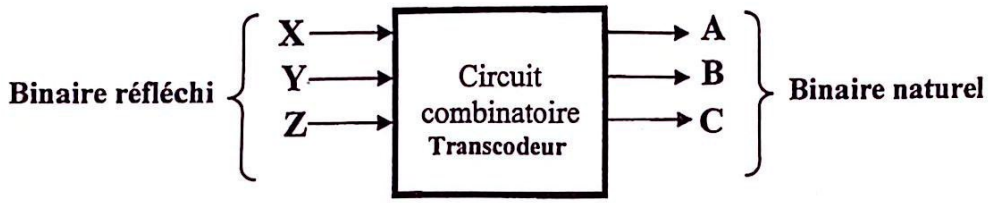


Scanned with CamScanner

1) Donner la table de vérité du décodeur.
2) Donner les expressions logiques des sorties A, B et E_{out} en fonction des entrées E_0, E_1, E_2, E_3 et E_{in}.
3) En déduire le logigramme du décodeur.
4) Comment peut-on obtenir un codeur de priorité à huit entrées à partir de deux décodeurs et de quelques portes logiques nécessaires.

EXERCICE 3: Transcodeur

Réaliser un transcodeur qui permet de convertir le code Gray (binaire réfléchi) à trois bits en code binaire naturel, comme le montre la figure suivante (X et A représentent les bits du poids le plus faible):



1- Etablir la table de vérité du circuit.
2- Simplifier les expressions algébriques par la méthode de karnaugh.
3- Etablir le logigramme correspondant en utilisant des portes élémentaires.
4- Etablir le logigramme correspondant en n'utilisant que des portes " ou exclusive".

Scanned with CamScanner

EXERCICE 4

On se propose de réaliser un compteur synchrone modulo 8, en utilisant des bascules JK.

Pour ceci :

1) Dresser les tables de vérité nécessaires pour la conception de ce circuit.
4) Donner la structure interne de ce compteur.
5) Tracer le chronogramme du compteur obtenu pour les 8 premières impulsions d'horloge.

Scanned with CamScanner

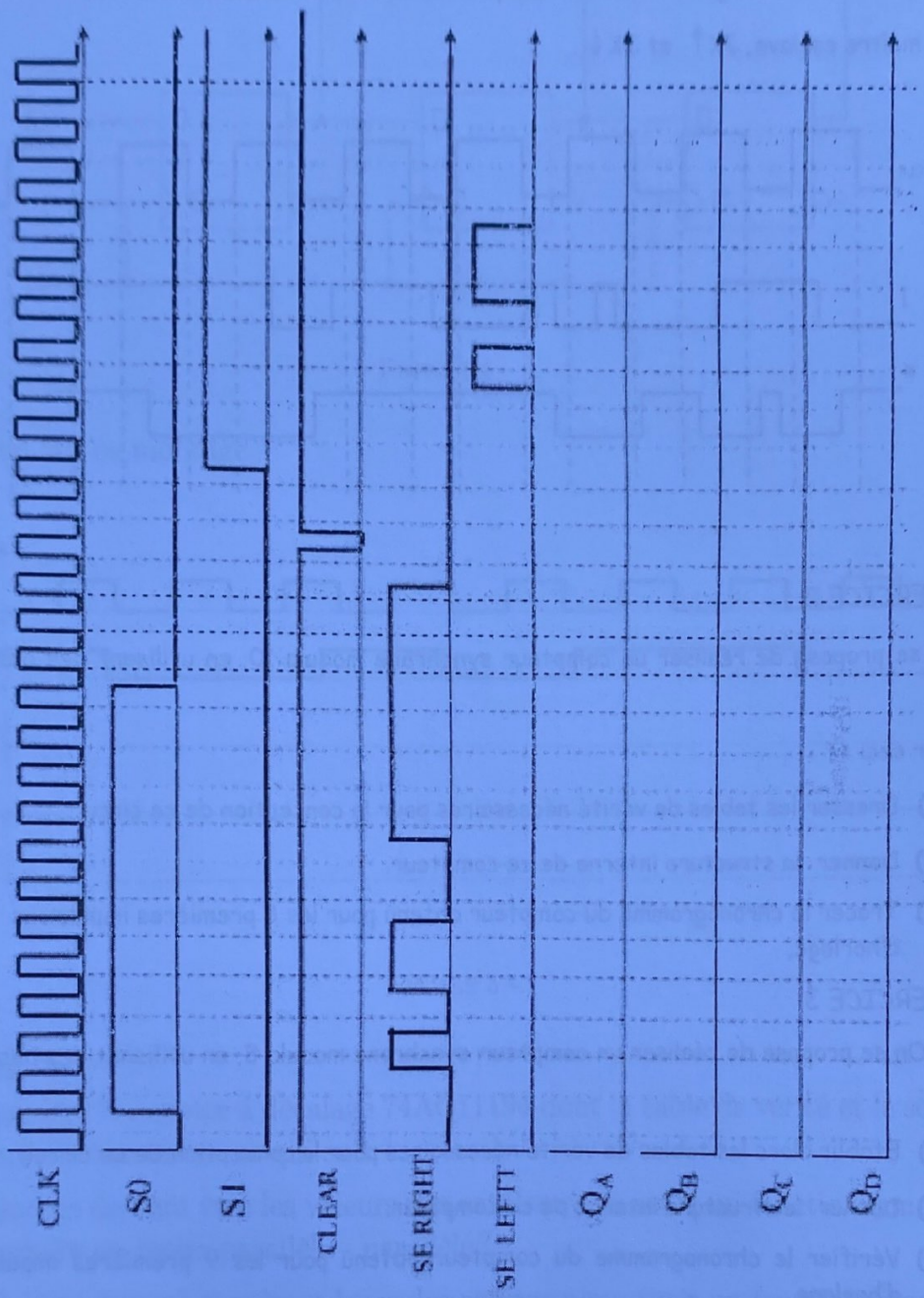


FIGURE 7 –

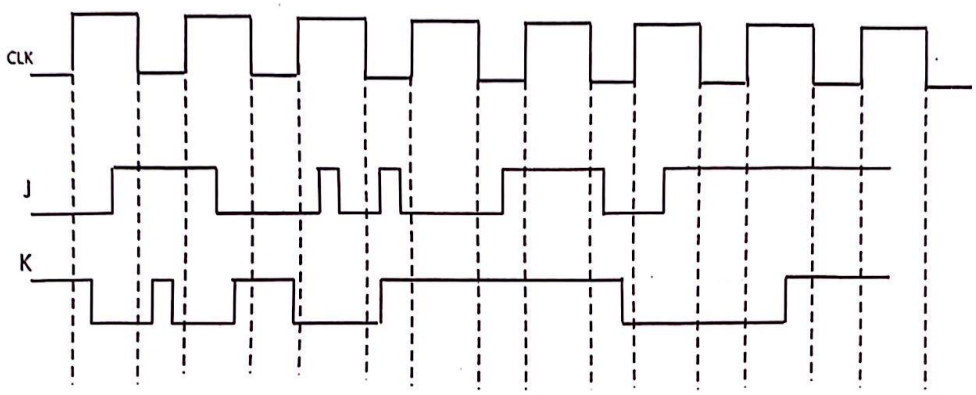
Scanned with CamScanner

TD N°3 : Systèmes séquentiels

EXERCICE 1

Compléter le chronogramme ci-dessous pour chacune des bascules suivantes:

JK maître esclave, JK↑ et JK↓ .



EXERCICE 2

On se propose de réaliser un compteur synchrone modulo 10, en utilisant des bascules JK.

Pour ceci :

1) Dresser les tables de vérité nécessaires pour la conception de ce circuit.
2) Donner la structure interne de ce compteur.
3) Tracer le chronogramme du compteur obtenu pour les 8 premières impulsions d'horloge.

EXERCICE 3

On se propose de réaliser un **compteur synchrone** modulo 8, en utilisant des bascules JK :

1) Etablir alors les tables de vérité nécessaires pour la conception de ce circuit.
2) Donner la structure interne de ce compteur.
3) Vérifier le chronogramme du compteur obtenu pour les 9 premières impulsions d'horloge.
4) Proposer la structure (en utilisant des bascules JK) d'un **compteur asynchrone** qui peut assurer le même comptage.

Scanned with CamScanner

Exercie 4

1. Remplir le chronogramme relatif au montage suivant :

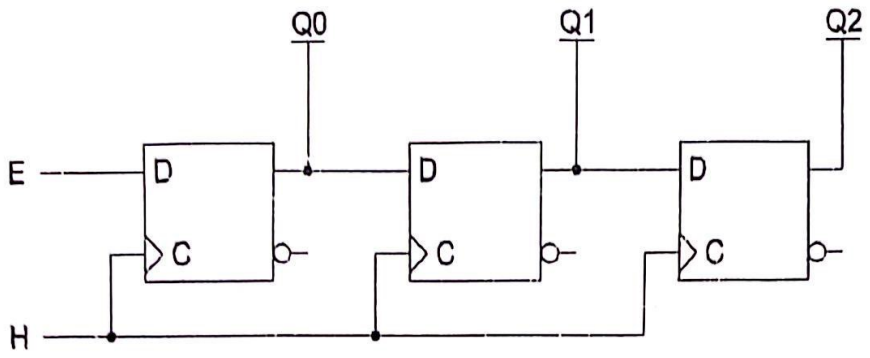


FIGURE 4 –

2. Que fait ce montage ?

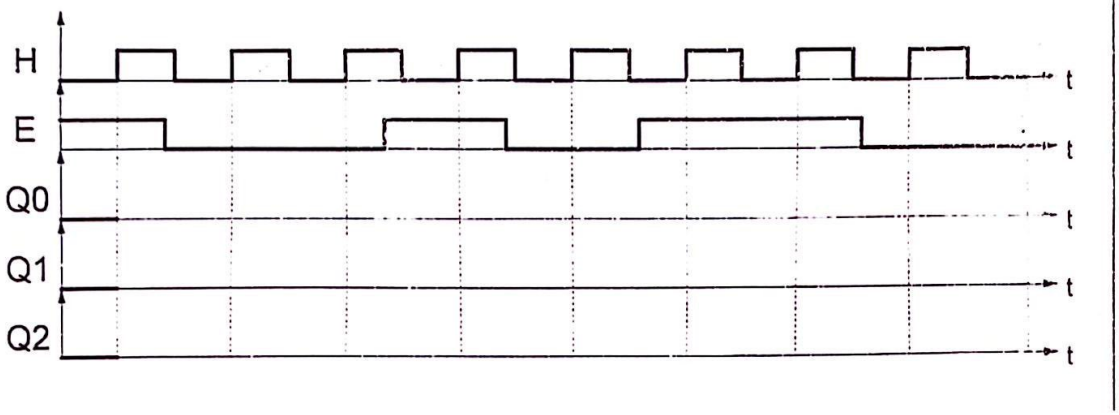


FIGURE 5 –

Exercie 5

On considère le registre à décalage 74AC11194 dont la table de vérité et le schéma logique sont donnés en ce qui suit.

1. Quelles doivent être les valeurs des entrées pour avoir un fonctionnement du registre en mode parallèle - parallèle ?
2. Quelles doivent être les valeurs des entrées pour avoir un fonctionnement du registre en mode série avec un décalage vers la droite ?
3. Même question pour avoir un décalage vers la gauche.
4. Compléter le chronogramme de la Figure 7. en considérant que toutes les sorties Q_i sont au niveau bas à l'instant initial.

Scanned with CamScanner

Function table

INPUTS										OUTPUTS			
CLEAR	MODE		CLOCK	SERIAL		PARALLEL				Q_A	Q_B	Q_C	Q_D
	S1	S0		LEFT	RIGHT	A	B	C	D				
L	X	X	X	X	X	X	X	X	X	L	L	L	L
H	X	X	↑	X	X	X	X	X	X	Q_{A0}	Q_{B0}	Q_{C0}	Q_{D0}
H	H	H	↑	X	X	a	b	c	d	a	b	c	d
H	L	H	↑	X	H	X	X	X	X	H	Q_{An}	Q_{Bn}	Q_{Cn}
H	L	H	↑	X	L	X	X	X	X	L	Q_{An}	Q_{Bn}	Q_{Cn}
H	H	L	↑	H	X	X	X	X	X	Q_{Bn}	Q_{Cn}	Q_{Dn}	H
H	H	L	↑	L	X	X	X	X	X	Q_{Bn}	Q_{Cn}	Q_{Dn}	L
H	L	L	X	X	X	X	X	X	X	Q_{A0}	Q_{B0}	Q_{C0}	Q_{D0}

H = high level (steady state)
L = low level (steady state)
X = irrelevant (any input, including transitions)
↑ = transition from low to high level
a, b, c, d = the level of steady-state input at inputs A, B, C, or D, respectively.
Q_{A0}, Q_{B0}, Q_{C0}, Q_{D0} = the level of Q_A, Q_B, Q_C, or Q_D, respectively, before the indicated steady-state input conditions were established.
Q_{An}, Q_{Bn}, Q_{Cn}, Q_{Dn} = the level of Q_A, Q_B, Q_C, or Q_D, respectively, before the most recent ↑ transition of the clock.

logic diagram (positive logic)

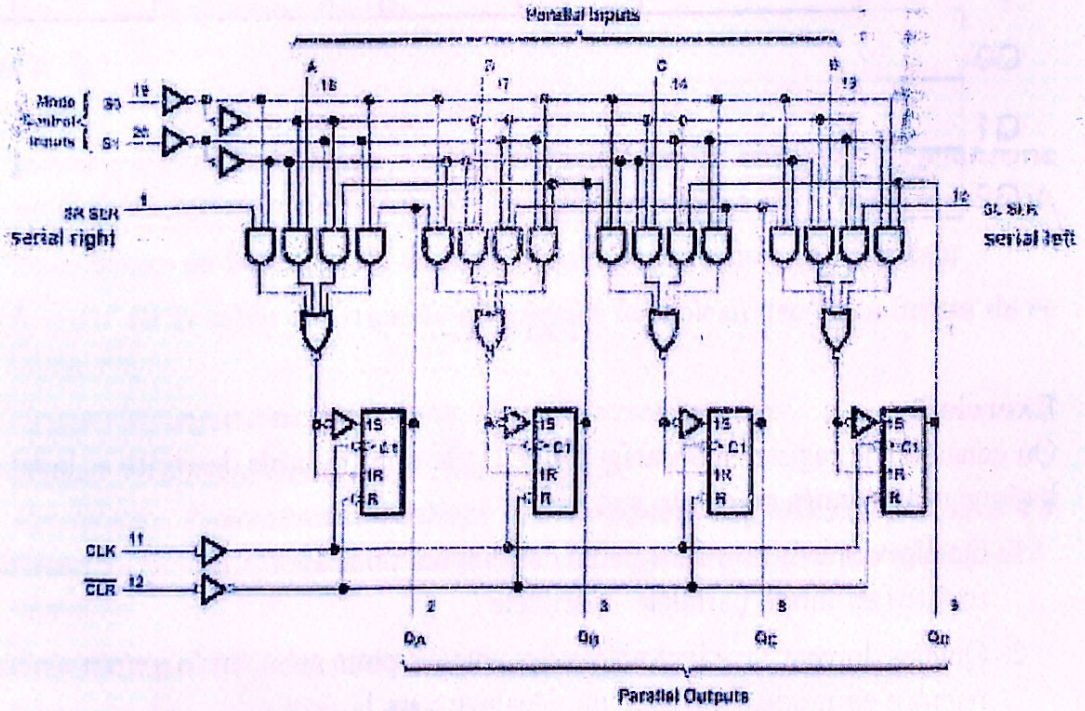


FIGURE 6 –

Scanned with CamScanner

9. Donner le modulo du compteur.

Exercie 2

Soit la figure suivante :

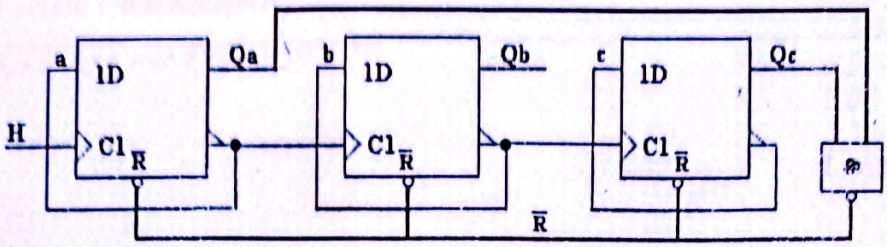


FIGURE 3 –

1. Donner la table de vérité de l'opérateur logique ($/R = f(Q_a, Q_c)$)
2. Quel est le role de l'entrée $/R$?A quel niveau est elle active ?
3. Tracer les chronogrammes des sorties Q_a, Q_b, Q_c et $/R$ (à l'état initial, $Q_a = Q_b = Q_c = 0$).
4. Convertir en décimal les trois bits binaires Q_c, Q_b et Q_a en prenant Q_a pour bit de poids faible.
5. Quelle est la fonction réalisée ?

Exercie 3

1. Donner la table des transitions d'une bascule JK.
   Dans un premier temps, on désire réaliser un compteur synchrone modulo 7 à l'aide de bascules JK synchronisées sur front montant.
2. De combien de bascules JK on aura besoin pour réaliser ce compteur.
3. À l'aide de la table des transitions, remplir le tableau des états futurs de ce compteur.
4. Donner les équations des entrées des différentes bascules.
5. Donner le schéma de câblage correspondant.
   On désire maintenant réaliser un compteur synchrone, modulo 8 en code Gray, à l'aide de bascules JK synchronisées sur front descendant.
6. Remplir le tableau des états futurs de ce compteur.
7. Donner les équations des entrées J_i, K_i correpondantes.

Scanned with CamScanner

Exercie 1

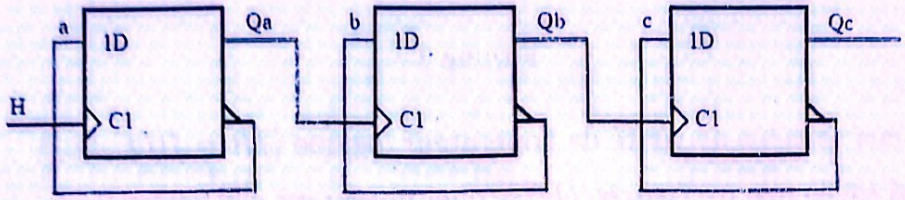


FIGURE 1 –

1. Le fonctionnement de ces bascules est-il synchrone ou asynchrone ? Argumenter votre réponse.
2. Tracer les chronogrammes des sorties Q_a, Q_b et Q_c (à l'état initial, $Q_a = Q_b = Q_c = 0$).
3. Convertir en décimal les trois bits binaires Q_a, Q_b et Q_c en prenant Q_a pour bit de poids faible.
4. Quelle est la fonction réalisée ?
5. Donner le modulo du compteur.
   Soit la figure suivante :

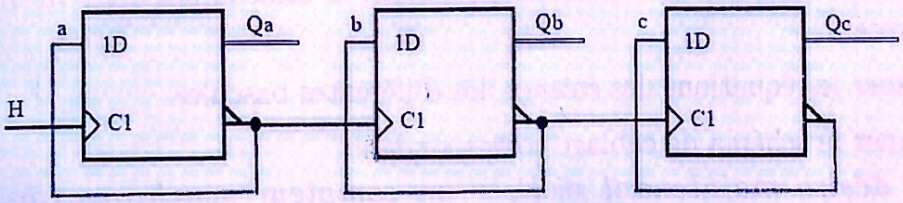


FIGURE 2 –

6. Tracer les chronogrammes des sorties Q_a, Q_b et Q_c (à l'état initial, $Q_a = Q_b = Q_c = 0$).
7. Convertir en décimal les trois bits binaires Q_a, Q_b et Q_c en prenant Q_a pour bit de poids faible.
8. Quelle est la fonction réalisée ? Comparer ce schéma structurel avec celui de la figure précédente et conclure sur l'incidence de la fonction réalisée.

Scanned with CamScanner

9. Donner le modulo du compteur.

Exercie 2

Soit la figure suivante :

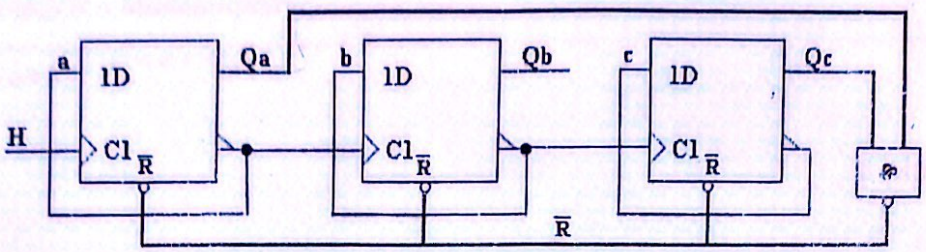


FIGURE 3 –

1. Donner la table de vérité de l'opérateur logique ($/R = f(Q_a, Q_c)$)
2. Quel est le role de l'entrée $/R$?A quel niveau est elle active ?
3. Tracer les chronogrammes des sorties Q_a, Q_b, Q_c et $/R$ (à l'état initial, $Q_a = Q_b = Q_c = 0$).
4. Convertir en décimal les trois bits binaires Q_c, Q_b et Q_a en prenant Q_a pour bit de poids faible.
5. Quelle est la fonction réalisée ?

Exercie 3

1. Donner la table des transitions d'une bascule JK.
   Dans un premier temps, on désire réaliser un compteur synchrone modulo 7 à l'aide de bascules JK synchronisées sur front montant.
2. De combien de bascules JK on aura besoin pour réaliser ce compteur.
3. À l'aide de la table des transitions, remplir le tableau des états futurs de ce compteur.
4. Donner les équations des entrées des différentes bascules.
5. Donner le schéma de câblage correspondant.
   On désire maintenant réaliser un compteur synchrone, modulo 8 en code Gray, à l'aide de bascules JK synchronisées sur front descendant.
6. Remplir le tableau des états futurs de ce compteur.
7. Donner les équations des entrées J_i, K_i correpondantes.

Scanned with CamScanner

Exercice 3 Simplifier les fonctions logiques suivantes en utilisant la méthode des tables de Karnaugh:

$$f_1 = a \bullet \bar{b} + a \bullet b \bullet \bar{d} + a \bullet b \bullet \bar{c} \bullet d$$

$$f_2 = \bar{a} \bullet b \bullet \bar{c} + \bar{a} \bullet \bar{c} \bullet d + \bar{a} \bullet \bar{b} \bullet \bar{d} + a \bullet c + b \bullet c \bullet \bar{d}$$

$$f_3 = a \bullet b \bullet \bar{c} + \bar{c} \bullet d \bullet a + a \bullet \bar{b} \bullet c \bullet d$$

Scanned with CamScanner

. *Exercices.*

Exercice 1

Démontrer les relations suivantes en utilisant les règles de calculs de l'algèbre de Boole

$$x + x \bullet y = x$$
$$x + \bar{x} \bullet y = x + y$$
$$x \bullet (x + y) = x$$
$$x \bullet (\bar{x} + y) = x \bullet y$$
$$x \bullet A + \bar{x} \bullet B + A \bullet B = x \bullet A + \bar{x} \bullet B$$
$$(x + A) \bullet (\bar{x} + B) \bullet (A + B) = (x + A) \bullet (\bar{x} + B)$$

Exercice 2 Ecrire les expressions logiques simplifiées des fonctions F et G définies par les tables de vérités suivantes :

x	y	z	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

x	y	z	G
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

Scanned with CamScanner

Exercice 1

1. Convertir les nombres décimaux suivants en base 2 (base binaire) :
   - 13
   - 0,125
   - 2016
   - 135,625
2. Écrire les nombres binaires suivants en base 10.
   - 11011
   - 101,101
   - 1001110,11
3. Par conversion rapide, écrire les nombres binaires précédents (question 2) en base octale puis en base hexadécimale.
4. Effectuer les opérations arithmétiques suivantes :
   - $(101011)_2 + (1011)_2$
   - $(101011)_2 - (11011)_2$
   - $(101011)_2$ X $(1011)_2$
   - $(101011)_2 \div (1011)_2$
   - $(AF5)_{16} + (10C)_{16}$

Exercice 2

1. Convertir les nombres hexadécimaux suivants en base 2 (base binaire) :
   - 9A
   - 5F3
   - 0,25
   - 1ABC,DE
2. Écrire les nombres précédents dans la base 8.
3. Les nombres et les résultats sont représentés sur 8 bits (on manipule des nombres signés). Faire les opérations binaires suivantes (par addition du complément à 2). Préciser s'il y'a retenue ou débordement et conclure en le justifiant si on doit retenir le résultat ou le rejeter.

   - $11011011 - 01101010$
   - $11101000 + 11110111$
   - $01011111 + 01100001$
   - $01011011 - 10111011$

Scanned with CamScanner

TD N°1 : codes et numérations

EXERCICE 1

1) Effectuer la somme des nombres binaires suivants, en montrant les retenus :

$$\begin{array}{r} 11011101 \\ +\ 1100011 \\ \hline \end{array} \qquad \begin{array}{r} 1110010 \\ +\ 1101101 \\ \hline \end{array}$$

2) Effectuer des soustractions au lieu des additions en montrant les emprunts d'une colonne à l'autre.

3) Effectuer la somme des nombres hexadécimaux suivants :

$$\begin{array}{r} F35B \\ +\ 27E6 \\ \hline \end{array} \qquad \begin{array}{r} 1B90F \\ +\ C44E \\ \hline \end{array}$$

EXERCICE 2

1) Montrer que les relations algébriques suivantes sont correctes :

a) $A + AB = A$

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

2) Donner la représentation binaire sur 8 bits (complément à 2 pour les nombres négatifs) des nombres décimaux suivants :

+18, -49, -3 et +115

3) Déterminer les mots binaires correspondant aux mots code Gray suivants :

a) $(1\,1\,1\,1\,0\,1\,1)_G$ =

b) $(1\,0\,0\,0\,1\,0\,1\,1)_G$ =

EXERCICE 3

1) Effectuer la soustraction des nombres binaires suivants en montrant les emprunts d'une colonne à l'autre :

10110101 -1100011 =

Scanned with CamScanner

2) Effectuer la somme des nombres hexadécimaux suivants :

4F1A5 + B8D5 =

1372 + 4631 =

3) Donner la représentation binaire (complément à 2 pour les nombres négatifs) sur 8 bits des nombres suivants écrit en système décimal :

+79,-49, -100

EXERCICE 4

1) Effectuer la conversion des nombres binaires suivants, en code de Gray :

a) 01110101 b) 11010100 c) 1101110

2) Chacun des nombres suivants est le complément à 2 d'un nombre binaire signé. Trouver sa valeur décimale :

a) 01101 b) 11011 c) 10010

3) Coder en BCD les nombres décimaux suivants :

a) 147_{10} b) $33,25_{10}$

Scanned with CamScanner

2) Effectuer la somme des nombres hexadécimaux suivants :

4F1A5 + B8D5 =

1372 + 4631 =

3) Donner la représentation binaire (complément à 2 pour les nombres négatifs) sur 8 bits des nombres suivants écrit en système décimal :

+79,-49, -100

EXERCICE 4

1) Effectuer la conversion des nombres binaires suivants, en code de Gray :

a) 01110101 b) 11010100 c) 1101110

2) Chacun des nombres suivants est le complément à 2 d'un nombre binaire signé. Trouver sa valeur décimale :

a) 01101 b) 11011 c) 10010

3) Coder en BCD les nombres décimaux suivants :

a) 147_{10} b) $33,25_{10}$

Scanned with CamScanner

Introduction à l'électronique numérique

I. Introduction

- Les grandeurs physiques que l'on mesure ou que l'on désire modifier sont des ....variables analogiques: elles prennent de nombreuses valeurs en passant en général de l'une à l'autre de manière continue. En électronique numérique et logique (digitale) on manipule les mêmes variables que l'électronique analogique (tensions, courants) mais sous forme de..signaux discrets.... Ces signaux ne peuvent prendre que deux valeurs ....0..;..1..... (variable binaire, bit : binary digit).
- Le système binaire présente de nombreux avantages. Les 4 opérations (+, -, ×, /) y sont simples, on peut câbler un système binaire à l'aide de simples ....ouvert ou fermé.....(c'est d'ailleurs sous cette forme mécanique que les premiers ordinateurs ont été réalisés).
- Les variables binaires sont moins affectées par des signaux parasites et leurs erreurs possibles que ceux analogiques.
- Dans les circuits physiques, ces deux variables binaires sont représentées par des....niveaux de tension bien séparés....., par exemples 0V et 5V.

Remarque :

Logique positive	Logique négative
0V → 0	0V → 1
5V → 1	5V → 0

- Les circuits électroniques numériques et logiques utilisent des composants identiques à ceux de l'électronique analogique, mais *en régime linéaire (commutation)*. Ces composants sont des transistors (d'où l'appellation parfois rencontrée de « Total Transistor Logic » (TTL)) et permettent de réaliser des interrupteurs (avec des résistances, des diodes...).
- L'élément binaire de base s'appelle le Binary Digit (BIT) ; il peut prendre deux valeurs : 0 ou 1. Il peut être interprété en tant que nombre binaire (écrit en base 2, contenant uniquement des 0 et des 1), et cela donne lieu à l'application numérique de cette électronique (Exp : fonction addition).
- Les circuits de base (comportant un nombre réduit de transistors) de l'électronique numérique et logique réalisent des opérations logiques simples : ET (AND), OU (OR), NON (NOT), XOR....
- A partir de ces portes logiques de base, on peut ensuite réaliser toutes les ....fonctions logiques complexes.....
- Ces opérations de base sont définies par leurs ....Table de vérité..... (un peu comme les opérations de calcul de base sont définies par les tables d'addition et de multiplication).

Scanned with CamScanner

- Il faut effectivement de nombreux bits pour coder la moindre information. On manipule alors des « mots » binaires de plusieurs bits (8, 16, 32, 64, ...).
- L'électronique numérique traite de nombres binaires (en base 2 : $1100011=2^0+2^1+\ldots\ldots+2^6=99$).

Si A1B1C1D1 et A2B2C2D2 représentent des nombres (de 4 bits : de 0 à 15), on peut calculer leur somme, leur différence, leur produit .....

- Les calculatrices, les ordinateurs,..., sont basés sur l'utilisation de ces opérations.

Exemples :

De circuits numériques simples : le 74LS83 permet de réaliser l'addition de 2 nombres de 4 bits.

Le 74LS85 permet de comparer 2 nombres de 4 bits (<, >, =...).

❖ *L'électronique logique*

L'information binaire sert à représenter des grandeurs binaires : Vrai ou Faux, Blanc ou Noir, Oui ou Non, 0 ou 1. Cela permet par exemple le contrôle de processus à partir de capteurs qui donnent des informations sur l'état du système.

Un exemple (très) simple : un plafonnier de voiture s'allume en fonction de l'information fournie par deux variables D et G qui représentent l'état de l'ouverture ou de la fermeture des portes, si P la variable qui détermine l'allumage de la lampe au plafond, l'équation logique donnant P est : ...P=D.ou.P=G.....

A partir de cette équation, on représente le schéma symbolique de la fonction plafonnier puis son schéma électronique.

Quelques exemples de fonctions logiques simples : boitiers de 4 portes ET à 2 entrées : 74LS08 ; boitier de 4 portes OU à 2 entrées : 74LS32.....

II. *De l'analogique au numérique/logique*

1. *le monde Analogique (A) et le monde Digital (D)*

- Ce qui semble une limitation apporte en effet un avantage énorme pour la transmission et le traitement des données (on peut facilement détecter une erreur).
- Pour les signaux analogiques la présence ...du bruit (signaux parasites)... superposé au signal, empêche l'utilisateur de déterminer une valeur précise.
- En électronique comme en physique, on détermine une mesure avec une certaine ...incertitude..... Il est alors illusoire de chercher à distinguer deux valeurs qui ne sont pas suffisamment séparées l'une de l'autre car le bruit les « mélange » (de même qu'il est impossible de chercher ce qu'il y a entre 2 et 3 dans le monde des entiers). Ce qui dans le monde analogique permet d'établir un « pont » entre l'analogique et le numérique et de « modéliser » le monde analogique par sa représentation numérique.

2. *Critèrse de passage de l'Analogique au Digital*

Le traitement des signaux digitaux se fait sur ...des signaux en tension..... Si le signal initial n'est pas une tension, on utilise un capteur qui transforme de manière continue le signal mesuré (pression, température, vitesse,...) en tension. Le passage d'une tension analogique (continue) à sa version digitale (numérisée) se fait par l'intermédiaire d'un circuit appelé ...convertisseur analogique numérique..... (ADC en anglais). La réalisation de cette opération se fait en deux temps : échantillonnage puis numérisation et doit respecter deux critères :

Scanned with CamScanner

a. *Critère temporel-Cadence d'échantillonnage (Théorème de SHANNON)*

Si la tension d'origine varie entre deux échantillons, on perd de l'information. Inversement, si la tension d'origine ne varie pas, il est utile de prendre de nouveaux échantillons. Il faut bien choisir la cadence d'échantillonnage qui permet de « passer » les parties HF du signal.

$$T_e < {}^1/_2 (f_{max})^{-1}$$

b. *Critère en tension*

- Le passage de la tension analogique à sa représentation digitale se fait ..sans perte d'information....., si le niveau minimum entre deux valeurs significatives est le même des deux côtés du circuit. Du côté digital, il s'agit du LSB (de l'écart entre un nombre et son plus proche voisin), du côté analogique, il s'agit de la valeur de bruit superposé au signal.
- De même qu'il n'y a pas d'information entre deux nombres entiers, il n'y a pas d'information entre deux valeurs de tension analogique qui ne sont pas espacées d'une quantité ..sup..au..bruit..... (on ne peut pas les distinguer).
- Si ΔV est l'amplitude du signal à l'entrée de convertisseur, avec bruit b et N_{max} le nombre de la valeur numérique en sortie ($N_{max} = 2N_B$ où N_B est le nombre de bits du convertisseur), on doit prendre :

$$N_{max} > \frac{\Delta V}{b}$$

Scanned with CamScanner

Algèbres de BOOLE et fonctions logiques

I . Introduction à l'algèbre de Boole

Un processeur est composé de transistors permettant de réaliser des fonctions sur des signaux numériques. Ces transistors, assemblés entre eux forment des composants permettant de réaliser des fonctions très simples. A partir de ces composants il est possible de créer des circuits réalisant des opérations très complexes. L'**algèbre de Boole** (du nom du mathématicien anglais *Georges Boole 1815 - 1864*) est un moyen d'arriver à créer de tels circuits.

L'algèbre de Boole est une algèbre se proposant de traduire des signaux en expressions mathématiques. Pour cela, on définit chaque signal élémentaire par des variables logiques et leur traitement par fonctions logiques. Des méthodes (Table de vérité, ...) permettent de définir les opérations que l'on désire réaliser, et à transcrire le résultat en une expression algébrique. Grâce à des règles appelées lois de composition, ces expressions peuvent être simplifiées. Cela va permettre de représenter grâce à des symboles un circuit logiques, c'est-à-dire un circuit qui schématise l'agencement des composants de base (au niveau logique) sans se préoccuper de la réalisation au moyen de transistors (niveau physique).

1) Variable logique

Un ordinateur ne manipule que des donnees binaire, on appelle donc variable logique une donnée binaire, c'est-à-dire une donnée ayant deux états possibles: 0 ou 1.

2) Fonction logique

On appelle « **fonction logique** » une entité acceptant plusieurs valeurs logiques en entrée et dont la sortie (il peut y en avoir plusieurs) peut avoir deux états possibles : 0 ou 1.

En réalité ces fonctions sont assurées par des composants électronique admettant des signaux électriques en entrée, et restituant un signal en sortie. Les signaux électroniques peuvent prendre une valeur de l'ordre de 5 Volts (c'est l'ordre de grandeur général) que l'on représente par un 1, ou 0 V que l'on représente par un 0.

3) Les portes logiques

Les fonctions logiques de bases sont appelées **portes logiques.** Il s'agit de fonctions ayant une ou deux entrées et une sortie :

- La fonction **OU** (en anglais *OR*) positionne sa sortie à 1 si l'une ou l'autre des ses entrée a 1
- La fonction **ET** (en anglais *AND*) positionne sa sortie à 1 si ses deux entrée sont a 1

Scanned with CamScanner

- La fonction ...ou exclusif... (en anglais *XOR*) positionne sa sortie à 1 si l'une ou l'autre de ses entrées est à 1 mais pas les deux simultanément
- La fonction **NON** (appelée aussi *inverseur*) positionne sa sortie à 1 si son entrée est à 0, et vice-versa

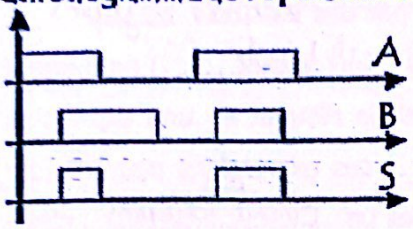
On définit généralement les fonctions **NON OU** (couramment appelée *NOR*) et **NON ET** (*NAND*) comme étant la composition respective d'un NON avec un OU et un ET.

4) Chronogramme

Un chronogramme est un diagramme montrant l'évolution des entrées et des sorties en fonction du temps.

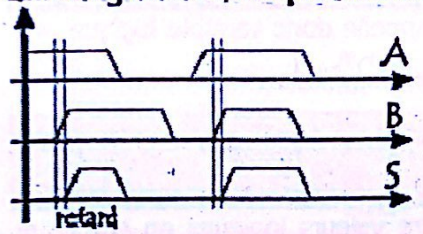
Voici par exemple ce à quoi pourrait ressembler un chronogramme de l'opérateur ET :

Chronogramme de l'opérateur ET



Ce chronogramme en un chronogramme idéal, en réalité les signaux électriques ne passent pas instantanément de 0 à 1, les pentes (ici verticales) sont obliques, et le traitement des entrées cause un retard sur les sorties :

Chronogramme de l'opérateur ET



5) Expression algébrique

Le but de l'algèbre de Boole est de ...décrire les traitements des signaux sous forme... Comme nous l'avons vu, les signaux sont représentés par des noms de variables. Les fonctions logiques sont représentées par des opérateurs :

- la fonction **OU** est représenté par un *plus*: $+$
- la fonction **ET** est représenté par un *point*: $\cdot$
- la fonction **NON** est représenté par une *barre* au-dessus de la variable inversée: $\overline{A}$

Elle est parfois représentée par un / devant la variable inversée

- la fonction **OU EXCLUSIF** est représentée par un *plus encerclé*: $\oplus$

Une expression algébrique sera donc une expression du type :

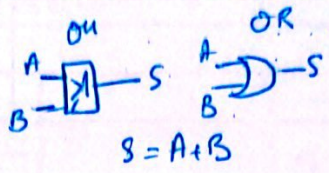
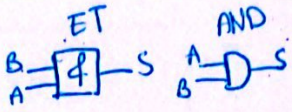
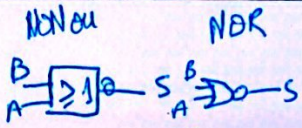
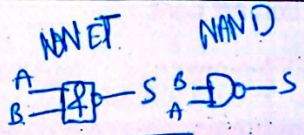
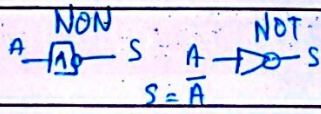
$S = \overline{A} \oplus (B.C)$

Scanned with CamScanner

6) Table de vérité

Une **table de vérité** est un tableau permettant de décrire toute les poss de sortie en fonction des entrées. On place donc les variables d'entrées dans les colonnes de gauche en les faisant varier de telle façon à couvrir l'ensemble des possibilités. La colonne (ou les colonnes si la fonction a plusieurs sorties) de droite décrit la sortie.

Voici par exemple les tables de vérités des portes logiques :

Nom de la porte	Entrée		Sortie
	A	B	S
OU 	0	0	0
	0	1	1
	1	0	1
	1	1	1
ET 	0	0	0
	0	1	0
	1	0	0
	1	1	1
NON OU 	0	0	1
	0	1	0
	1	0	0
	1	1	0
NON ET 	0	0	1
	0	1	1
	1	0	1
	1	1	0
NON 	0		1
	1		0

Il est possible à partir de la table de vérité d'une fonction d'écrire son expression algébrique

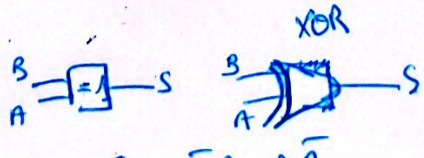
Soit la table de vérité suivante :

Entrée		Sortie
A	B	S
0	0	0
0	1	0
1	0	1
1	1	0

La sortie vaut 1 lorsque A vaut 1 et B vaut 0, l'expression algébrique de cette fonction est donc :

$S = A.\overline{B}$

---

ou exclusif : 

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

$S = \overline{A}B + A\overline{B}$

Scanned with CamScanner

Prenons maintenant la table de vérité suivante :

Entrée			Sortie
A	B	C	S
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

La sortie vaut 1 lorsque :

- A vaut 0
- B vaut 1
- C vaut 0

ou

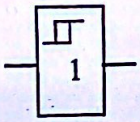
- A vaut 1
- B vaut 1
- C vaut 0

L'expression algébrique de cette fonction est donc :

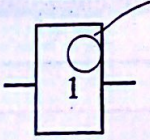
$S = \bar{A}.B.\bar{C} + A.B.\bar{C}$

II. Les fonctions logiques et algèbres de BOOLE

1) Fonctions de base : *OUI - NON - ET - OU*



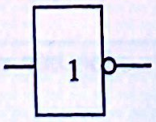
⊓ : Entrée avec TRIGGER de SCHMITT pour résoudre le problème de signaux d'entrée à transition lente

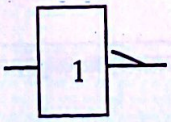


> : Sortie amplifiée (bufférisée). courant de sortie max du niveau plus important

⊄ : sortie à collecteur ouvert. commander une charge fonctionnant sous un potentiel différent de Vcc ou pour relier des sorties ensemble

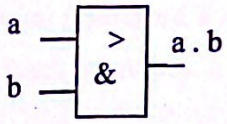
= : sortie à 3 états : haut, bas, haute impédance

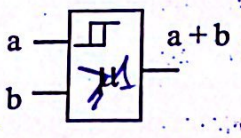




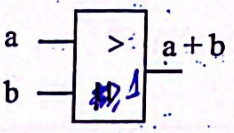
Fonction NON

Scanned with CamScanner





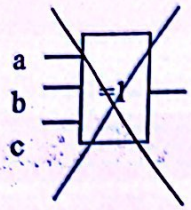
ou



ou

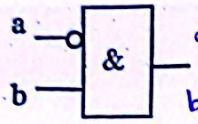
2) Fonction *OU Exclusif* et fonction *Inhibition*

S = $a \oplus b = \bar{a}\ b + a\ \bar{b}$ (va et vient).





S = $(a \oplus b) \oplus c$



S = $\bar{a} . b$ => Inhibition

3) Fonction NOR et NAND

3.1) Théorème de DE MORGAN

Le complément d'une somme est égal au produit de tous les termes complémentés

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

Le complément d'un produit est égal à somme de termes complémentés

$$\overline{a \bullet b} = \bar{a} + \bar{b}$$

3.2) Fonction NOR

a	b	a+b	$\overline{a+b}$
0	0	1	0
0	1	0	1
1	0	0	1
1	1	0	1

Pour réaliser une expression logique à l'aide de fonctions NOR il faut mettre cette expression sous la forme d'un produit de somme

Scanned with CamScanner

3.3) Fonction NAND

a	b	a • b	$\overline{a \bullet b}$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Pour réaliser une expression logique à l'aide de fonctions NAND il faut mettre cette expression sous la forme de somme de produit

III. Les lois de composition

Les lois de composition sont des règles logiques qui permettent de simplifier l'écriture des expressions algébriques.

1) Associativité

- (A.B).C est équivalent à A.(B.C)
- (A+B)+C est équivalent à A+(B+C)

2) Absorption

- A.(A+B) est équivalent à A
- A+A.B est équivalent à A

3) Commutativité

- A.B est équivalent à B.A
- A+B est équivalent à B+A

4) Distributivité

- A+(B.C) est équivalent à (A+B).(A+C) (de la somme par rapport au produit)
- A.(B+C) est équivalent à A.B+A.C (du produit par rapport à la somme)

5) Idempotence (redondance)

- A.A est équivalent à A
- A + A est équivalent à A

Scanned with CamScanner

6) Identité (Elément neutre)

- 1.A est équivalent à A
- 0+A est équivalent à A

7) Inversion

- A./$\overline{A}$ est équivalent à 0
- A+/$\overline{A}$ est équivalent à 1

8) Nullité (Elément absorbant)

- 0.A est équivalent à 0
- 1+A est équivalent à 1

9) Théorème de *De Morgan*

- $\overline{A.B}$ est équivalent à $\overline{A}+\overline{B}$
- $\overline{A+B}$ est équivalent à $\overline{A}.\overline{B}$

10) Règle du multiple du complément

- $a + \bar{a}\,b = (a + \bar{a})(a + b) = a + b$
- $a + a\,\bar{b} + \bar{a}\,b\,c + a\,b\,\bar{c} = a(1 + b + b\,\bar{c}) + \bar{a}\,b\,c = a + b\,c$

Remarques

<u>Loi d'absorption :</u>

A titre informatif, il est possible de prouver la loi d'absorption avec les autres lois :

A.(A+B) = A.A + A.B

= A + (A.B)

= A(1+B)

= .....

= A

Sur la deuxième ligne, la deuxième loi d'absorption apparaît.

Lorsqu'une somme logique contient un terme et un de ses multiples, on peut négliger le multiple.

Exemple : $x + x\,y = x$

$a\,\bar{b} + a\,\bar{b}\,c\,d = a\,\bar{b}$

$x + x\,y + x + x\,\bar{y}\,\bar{z} =$ ....x....

$a\,b + a\,b\,c + a\,b\,\bar{c}\,d =$ ....ab....

$x + \bar{x} + x\,y + z + x\,y\,z =$ ....1....

Scanned with CamScanner

$a \oplus a = 0$ $\quad \bar{a} \oplus a = 1$

$a \oplus 0 = a$ $\quad a \oplus 1 = \bar{a}$

Commutativité : $a \oplus b = b \oplus a$

Associativité : $(a \oplus b) \oplus c = a \oplus (b \oplus c)$

Complément : $\bar{a} \oplus \bar{b} = a \oplus b$ $\quad \overline{a \oplus b} = \bar{a} \oplus b = a \oplus \bar{b}$

Exemple de simplification d'expression

Soit la simplification suivante :

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

IV. Application à la simplification des équations et des circuits logiques

Dés que l'on dispose de l'expression d'un circuit logique, il peut être possible de la minimiser pour obtenir une équation comptant moins de termes ou de variables par terme. Cette simplification peut se faire de deux façons différentes :

- par l'utilisation des ...Théorème de l'algèbre de Boole...;
- par l'utilisation des ...Tableau de carnot...

Dans les deux cas, il est indispensable d'exprimer l'équation sous la forme d'une somme de produit

Remarque : Le signe de complémentation (-) ne peut pas surmonter plus d'une variable à la fois.

1) Simplification par l'algèbre de Boole

$P = (a + \bar{b})(b + \bar{c})(c + \bar{a})$ $\quad Q = (a+b+c)(a+\bar{b}+c)(a+\bar{b}+\bar{c})$

$R = abc + a\bar{b}(\overline{\bar{a}\ \bar{c}})$ $\quad S = \bar{a}c(\overline{\bar{a}bd}) + \bar{a}b\bar{c}\bar{d} + a\bar{b}c$

$T = abc + ab\bar{c} + a\bar{b}c$ $\quad U = (\bar{a}+b)(a+b+d)\bar{d}$

$V = (a+b)(a+c) + (b+c)(b+a) + (c+a)(c+b)$

2) Simplification par les tableaux de KARNAUGH

Rappel : Les tableaux de KARNAUGH permettent la simplification des équations logiques. Ils comportent 2^n cases, n étant le nombres de variables d'entrée, organisés selon le code ...GRAY... (ex : 4 variables donnent 16 cases).

Chaque case correspond à une combinaison possible des variables d'entrée;

Chaque combinaison exprimée dans l'équation ...sera représentée par un 1 ds la case correspondante

Il est ensuite possible de regrouper les cases par 2, 4, 8, 2^n afin d'éliminer les variables qui change d'état dans le regroupement :

- un regroupement de 2 cases élimine 1 variable;
- un regroupement de 2^x cases élimine x variables.

Scanned with CamScanner

Propriétés de la somme modulo 2 :

$a \oplus a = 0$ $\quad\quad$ $\bar{a} \oplus a = 1$

$a \oplus 0 = a$ $\quad\quad$ $a \oplus 1 = \bar{a}$

Commutativité : $a \oplus b = b \oplus a$

Associativité : $(a \oplus b) \oplus c = a \oplus (b \oplus c)$

Complément : $\bar{a} \oplus \bar{b} = a \oplus b$ $\quad\quad$ $\overline{a \oplus b} = \bar{a} \oplus b = a \oplus \bar{b}$

Exemple de simplification d'expression

Soit la simplification suivante :

$\bar{A}.B.C + A.B.C + A.\bar{B}.C + \bar{A}.B.\bar{C}$

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

$= \bar{A}.B + A.C$

IV. Application à la simplification des équations et des circuits logiques

Dés que l'on dispose de l'expression d'un circuit logique, il peut être possible de la minimiser pour obtenir une équation comptant moins de termes ou de variables par terme. Cette simplification peut se faire de deux façons différentes :

- par l'utilisation des ..Théorème de l'algèbre de Boole.....;
- par l'utilisation des ..Tableau de Carnot.....

Dans les deux cas, il est indispensable d'exprimer l'équation sous la forme d'une somme de produit

Remarque : Le signe de complémentation (-) ne peut pas surmonter plus d'une variable à la fois.

1) Simplification par l'algèbre de Boole

$P = (a+\bar{b})(b+\bar{c})(c+\bar{a})$ $\quad\quad$ $Q = (a+b+c)(a+\bar{b}+c)(a+\bar{b}+\bar{c})$

$R = abc + a\bar{b}\,\overline{(\bar{a}\ \bar{c})}$ $\quad\quad$ $S = \bar{a}\,c\,\overline{(\bar{a}\,b\,d)} + \bar{a}\,b\,\bar{c}\,\bar{d} + a\,\bar{b}\,c$

$T = abc + ab\bar{c} + a\bar{b}c$ $\quad\quad$ $U = (\bar{a}+b)(a+b+d)\bar{d}$

$V = (a+b)(a+c) + (b+c)(b+a) + (c+a)(c+b)$

2) Simplification par les tableaux de KARNAUGH

Rappel : Les tableaux de KARNAUGH permettent la simplification des équations logiques. Ils comportent 2^n cases, n étant le nombres de variables d'entrée, organisés selon le code .GRAY.... (ex : 4 variables donnent 16 cases).

Chaque case correspond à une combinaison possible des variables d'entrée;

Chaque combinaison exprimée dans l'équation sera représentée par un 1 ds la case correspondante

Il est ensuite possible de regrouper les cases par 2, 4, 8, 2^n afin d'éliminer les variables qui change d'état dans le regroupement :

- un regroupement de 2 cases élimine 1 variable;
- un regroupement de 2^x cases élimine x variables.

Scanned with CamScanner

Simplifier les équations suivantes à l'aide de carnot :

$T1 = x\,y\,z + x\,y\,\bar{z} + \bar{x}\,y\,\bar{z} + \bar{x}\,y\,z$

$T2 = x\,\bar{y}\,\bar{z} + x\,y\,\bar{z} + x\,y\,z + \bar{x}\,y\,\bar{z}$

$T3 = y\,w + z\,w + \bar{z}\,w + \bar{x}\,y\,\bar{z}\,\bar{w} + x\,y\,\bar{z}$

$T4 = x\,y\,z + z\,(x\,\bar{y} + \bar{x}\,y)$

3) Réalisation de logigrammes

Réaliser les logigrammes, uniquement en portes NAND (à deux entrées) puis en portes NOR (à deux entrées) correspondant aux équations suivantes :

$N = a\,\bar{d} + \bar{d}\,b + \bar{c}\,b + c\,\bar{a}$

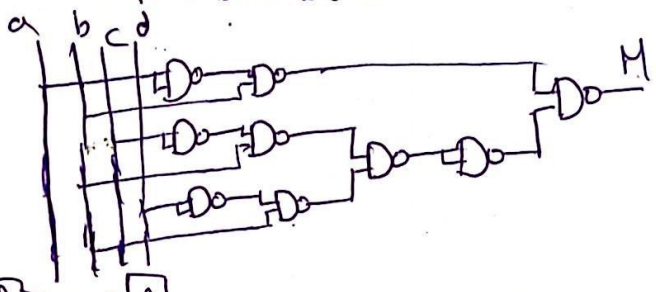
$= \overline{a\bar{d} + \bar{d}b + \bar{c}b + c\bar{a}}$

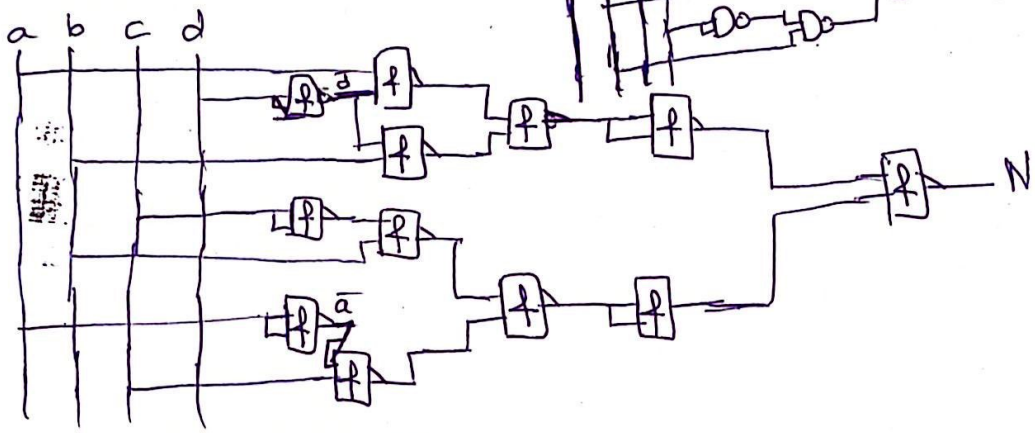
$= \overline{a\bar{d}} \cdot \overline{\bar{d}b} \cdot \overline{\bar{c}b} \cdot \overline{c\bar{a}}$

$= \overline{a\bar{d}} \cdot \overline{\bar{d}b} \cdot \overline{\bar{c}b} \cdot \overline{c\bar{a}}$

$M = \bar{a}\,b + b\,\bar{c} + b\,\bar{d}$

$= \overline{\bar{a}b} \cdot \overline{b\bar{c}} \cdot \overline{b\bar{d}} = \overline{\bar{a}b} \cdot \overline{b\bar{c}} \cdot \overline{b\bar{d}}$

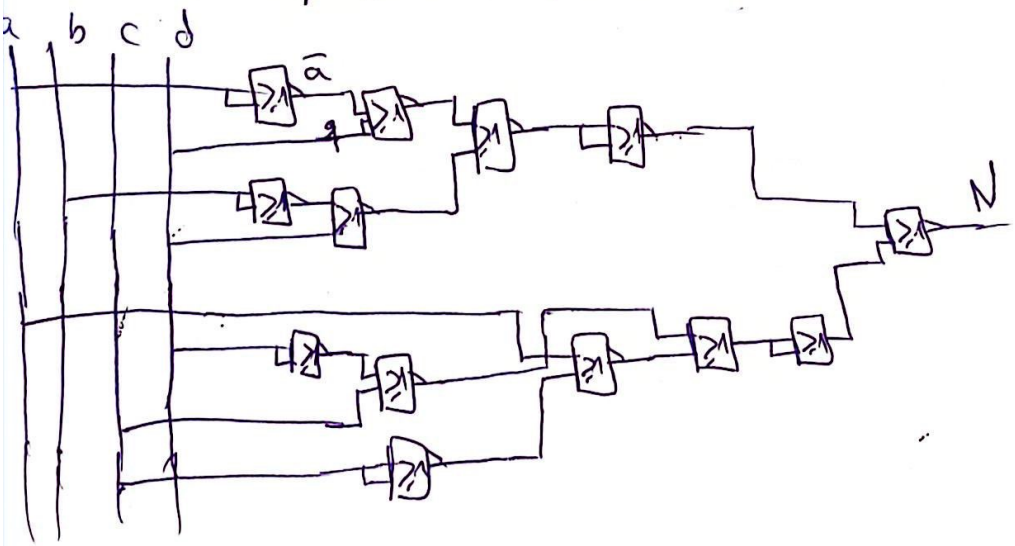




$N = a\bar{d} + \bar{d}b + \bar{c}d + c\bar{a}$

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

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



Scanned with CamScanner

Sortir les équations simplifiées en utilisant les tableaux de KARNAUGH.

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

$M = \bar{c} + b$

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

$N = d\bar{b} + \bar{b}\bar{d}$

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

$P = \bar{a}d + \bar{c}d + \bar{c}\bar{b}$

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

$R = cd + ab + d\bar{b}$

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

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

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

$T = c\bar{d} + a\bar{b} + c\bar{a}b + a\bar{c}d + \bar{c}d\bar{b} + \bar{d}\bar{a}b$

de \ abc	000	001	011	010	110	111	101	100
00	0	0	0	1	1	1	1	0
01	0	0	0	1	1	1	1	0
11	0	0	0	1	1	1	1	0
10	0	0	0	1	1	1	1	0

$H = b\bar{c} + ac$

de \ abc	000	001	011	010	110	111	101	100
00	0	1	1	0	0	1	1	0
01	0	1	1	0	0	1	1	0
11	0	1	1	0	0	1	1	0
10	0	1	1	0	0	1	1	0

$J = C$

de \ abc	000	001	011	010	110	111	101	100
00	1	0	0	1	1	0	0	1
01	0	1	0	0	0	0	1	0
11	0	1	0	0	0	0	1	0
10	1	0	0	1	1	0	0	1

$K = e\bar{b}c + \bar{e}\bar{c}$

de \ abc	000	001	011	010	110	111	101	100
00	1	1	1	0	0	1	1	1
01	0	1	1	0	0	1	1	0
11	1	1	1	1	1	1	1	1
10	1	0	0	1	1	0	0	1

$L = de + cd + \bar{e}\bar{b}\bar{c} + d\bar{c}$

Scanned with CamScanner

Rappels sur les systèmes de codage

1) Systèmes binaires.

En binaire, on distingue trois principaux systèmes de codage :

- binaire pur : (1-2-4-8);
- binaire réfléchi : (code GRAY);
- binaire D C B ou B C D

a) Code binaire naturel.

Dans ce codage, on utilise le poids binaire de chaque chiffre en fonction de son rang. Nous pouvons faire l'analogie entre le système binaire et le système décimal.

1	9	9	9	⟶	1	1	1	1	1	0	0	1	1	1	1
10^3	10^2	10^1	10^0		2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

b) Code binaire réfléchi.

Dans ce codage, un seul bit change d'état lorsque l'on passe d'un terme au suivant.

A l'apparition d'une variable supplémentaire on fait la symétrie du code déjà obtenu plus le nouveau bit à 1.

Le code peut se refermer sur lui-même sans perdre ses propriétés dans la mesure ou le dernier terme se situe juste avant un .....

Intérêt:.....

.....

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

c) Code binaire D C B (Décimal Codé Binaire).

Dans ce codage, chaque chiffre décimal est converti en binaire, indépendamment des autres chiffres.

1	9	9	9
0 0 0 1	1 0 0 1	1 0 0 1	1 0 0 1

Ce code est utilisé dans les systèmes traitant des nombres décimaux uniquement :

- .....
- Dans les petites calculettes de poche.

Inconvénient: Il nécessite plus de bits que le binaire naturel pour coder le même nombre décimal.

Scanned with CamScanner

2) Système hexadécimal.

Le codage héxadécimal est trés utilisé dans les systèmes à microprocesseur car il simplifie l'écriture des nombres binaires.

Ce codage utilise 16 symbôles [0 . . 9 et A . . F]

L'analogie avec le système décimal peur être faite .

1	9	9	9	⟶	7	C	.	F
10^3	10^2	10^1	10^0		16^2	16^1		16^0

Chaque chiffre héxadécimal est défini par quatre bits.

Quartet (4 bits) ⟶ 16 combinaisons;

Octet (8 bits) ⟶ 256 combinaisons.

3) Exercices.

Convertir en binaire, puis en hexadécimal les nombres suivants :

77				0	1	0	0	1	1	0	1		4	D
1256														
4096														
4095														
255														

Scanned with CamScanner