Logique Combinatoire et Algèbre de BOOLE

Septembre 2017

GOKPEYA NESSEMOU ERIC @ INGÉNIEUR
ASSISTANT UVCI

0.0.1

Table des matières

Objectifs

À la fin de cette leçon, vous serez capable de :

- Définir les variables et fonctions logiques ;
- Manipuler les fonctions combinatoires et les propriétés de l'Algèbre de BOOLE;
- Manipuler l'analyse booléenne des fonctions combinatoire par la résolution de sujet numérique

Introduction

On appel « *système combinatoire* », tout système numérique dont les *sorties* sont *exclusivement définies à partir des variables d'entrées.*

Le Schéma suivant donne une illustration

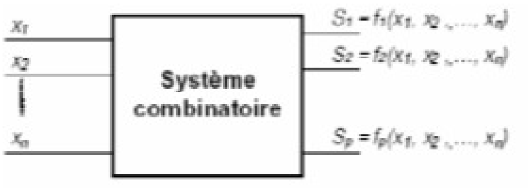


Figure 1 : système combinatoire générant *p sorties Si* à partir des n variables d'*entrées xi.*

Pour réaliser de tel système, on fait appel à des portes dites « *logique* » correspondantes aux *fonctions autorisés par l'algèbre de BOOLE.*

Partie I : Les fonctions logiques de bases (Activités d'Apprentissage)

Objectifs

Définir les variables et les fonctions *logiques de base*

1. 1. Définitions

Variable logique

On appelle variable binaire (ou *logique*), une variable prenant ses valeurs dans l'ensemble $\{0, 1\}$.

Exemple : Etat d'un interrupteur, d'un bouton poussoir, la présence d'une tension, ...

Soit *a,* la variable associée à l'état d'un *bouton poussoir*,

alors $a = 0$ (faux ou bas) signifie qu'il *n'est pas actionné*,

$a = 1$ (vrai ou haut) signifie *qu'il est actionné.*

Équation logique

On appelle *équation logique* une *combinaison de plusieurs variables logiques* donnant l'état d'une variable dite de sortie associée.

Cette combinaison est réalisée à l'aide d'*opérations logiques* :

Soit *xi (i [1, n])* les variables d'entrée.

L'équation $A = f(xi\ )$ définit l'*état* de la *variable de sortie A*.

Table de vérité

La table de vérité représente l'état de la variable de sortie pour chacune des combinaisons des *n variables d'entrée* (2^n *lignes*).

Pour *1* variable, la TdV aura $2^1 = 2$ lignes

- Pour *2* variables, la TdV aura $2^2 = 4$ lignes
- Pour *3* variables, la TdV aura $2^3 = 8$ lignes
- Pour *4* variables, la TdV aura $2^4 = 16$ lignes

Exemple : Table de vérité avec trois variables d'entrées (A,B et C) avec une sortie S

Entrées A, B et C (3)
Sortie S
TVR avec $2^3 = 8$ lignes

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

Variables d'entrée — État de la sortie

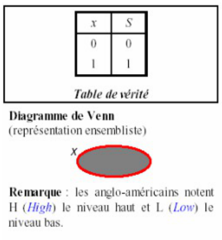
2. 2. La fonction logique " OUI"

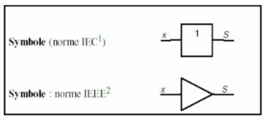
Opérateur " OUI "

L'opération (ou opérateur) *OUI* est dite *unaire* (ne s'applique qu'à une seule opérande).
Elle affecte à la variable de *sortie*, l'*état logique de la variable d'entrée.*

Opérateur OUI

Équation





x est l'entrée, S la sortie : $S = x$

3. 3. La fonction logique " NON " ou inversion

Équation

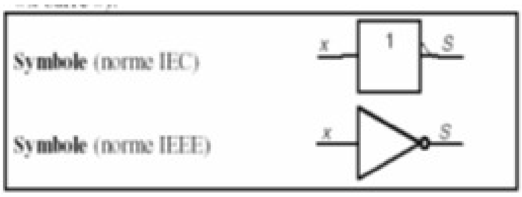
on a :

$$S = \overline{A}$$

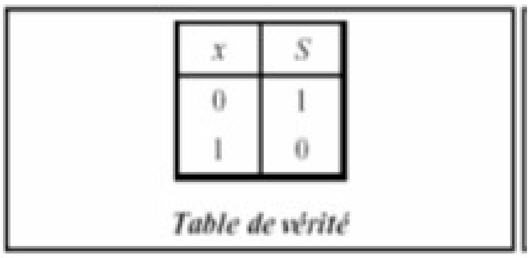
C'est le *contraire* ou l'*opposé* d'une variable logique.
on peut aussi écrire : $S = not(A)$

Symboles

Les deux représentations 'Américaine et Européenne



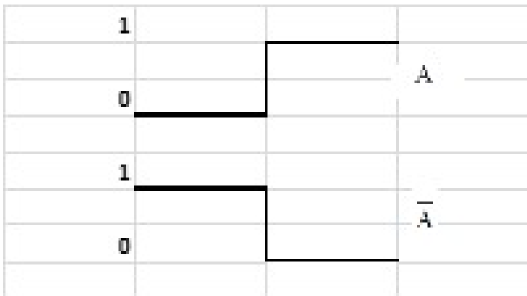
TVR



x	S
0	1
1	0

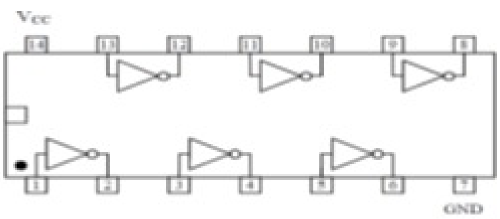
Table de vérité

Chronogramme



Exemple : Circuit intégré (CI) contenant des portes NON

CI 74LS04



Ce CI contient 6 portes inverseurs.

4. 4. La fonction logique "ET" (AND) ou intersection

Cette opération est appelée *multiplication* ou *produit logique*.

On l'exprime par un point " . " (qui se lit ET), par des parenthèses ou par des variables qui se suivent, comme en algèbre :

x et y les entrées, S la sortie. $S = x.y$, c'est aussi une *intersection*.

Symbole

Symboles

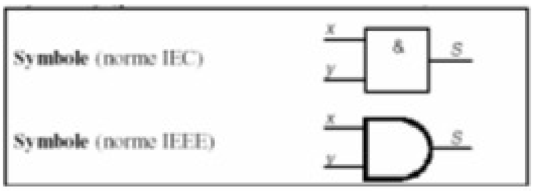


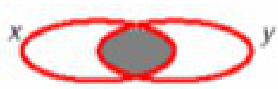
Table de vérité

TVR et Venn

x	y	$x \wedge y$
0	0	0
0	1	0
1	0	0
1	1	1

Table de vérité

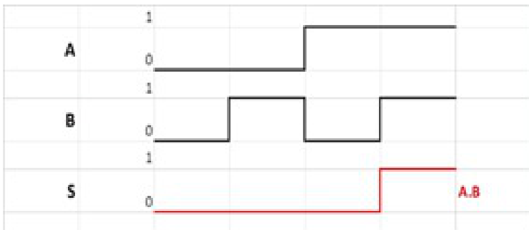
Diagramme de Venn



Remarque : une seule entrée à 0 suffit pour forcer la sortie à 0.

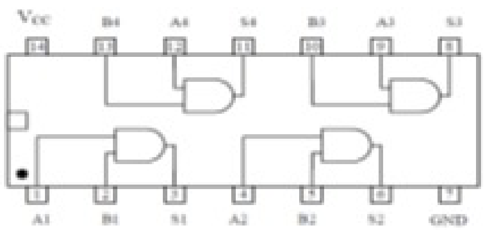
Chronogramme

Signal



Exemple : CI : 74LS08

CI



5. 5. La fonction logique " OU " (OR)

Cette opération est appelée *addition, somme logique ou union.*

Pour l'indiquer, on utilise le signe $+$ *(se lit "ou"). $A + B$ ou encore $A \quad B$.*

Symbole

Symbole

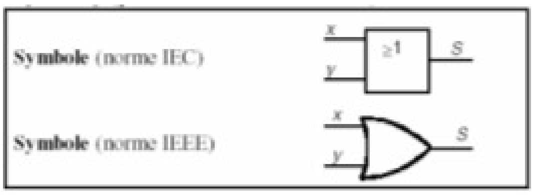
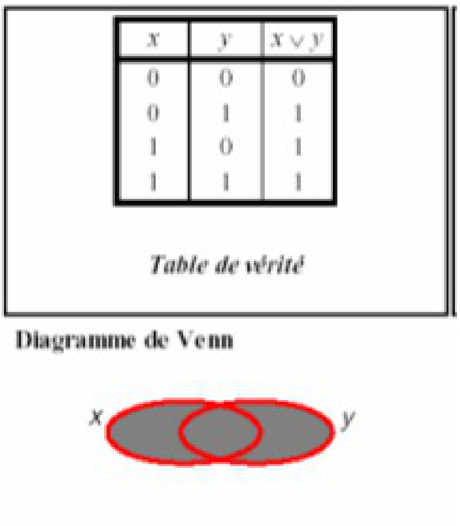


Table de vérité

TVR et Venn

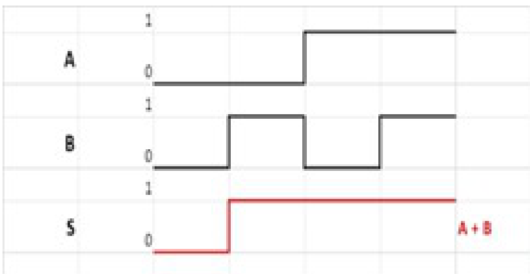


x	y	$x \vee y$
0	0	0
0	1	1
1	0	1
1	1	1

Table de vérité

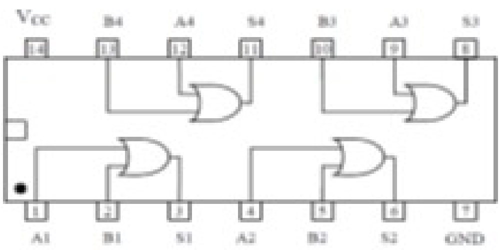
Chronogramme

Signal



Exemple : Exemple de CI : 74LS32

CI



6. 6. La fonction logique dérivée " NON-ET "(NAND)

C'est l'association de la *porte ET avec un inverseur*. Le résultat de la fonction ET est inversé en sortie.

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

on peut aussi écrire :

S = not(A.B)

symbole

Symbole

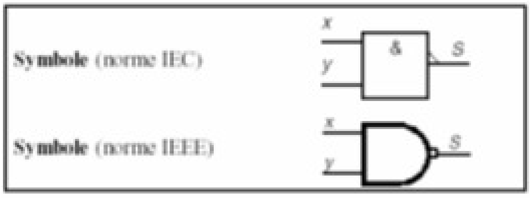
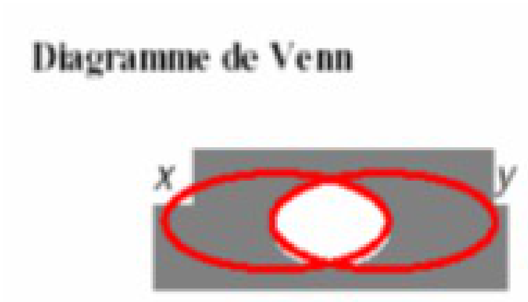


Table de vérité

TVR

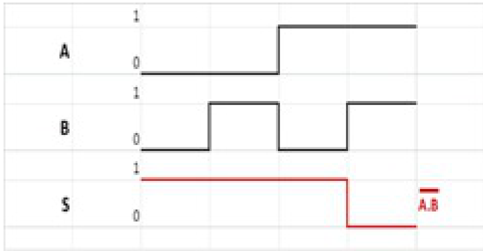
x	y	$x \uparrow y$
0	0	1
0	1	1
1	0	1
1	1	0

Diagramme de Venn



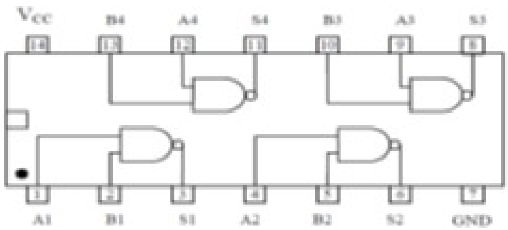
Chronogramme

Chronogramme



Exemple : CI : 74LS32

CI



7. 7. La fonction logique dérivée " NON-OU " (NOR)

C'est l'association de la porte OU avec un inverseur. Le résultat de la fonction OU est inversé en sortie.

S =

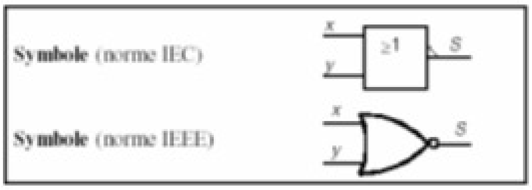
$$\overline{A+B}$$

on peut aussi écrire :

S = not(A + B)

Symbole

Symbole



Autre Symbole

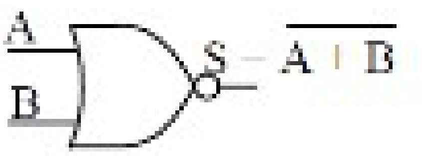
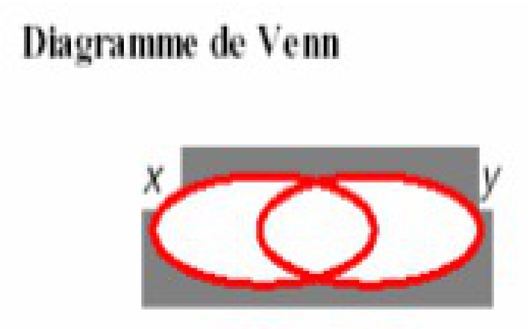


Table de vérité

TVR

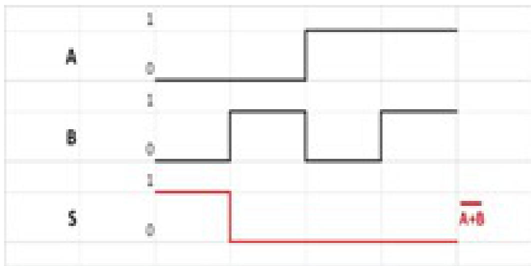
x	y	$x \downarrow y$
0	0	1
0	1	0
1	0	0
1	1	0

Venn



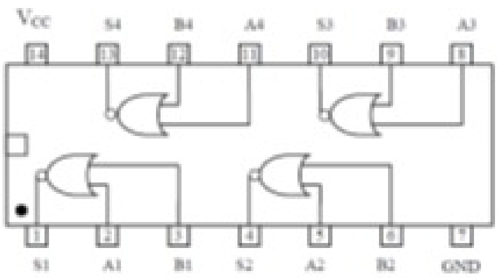
Chronogramme

Signal



☛ *Exemple : CI : 74LS02CI : 74LS02*

CI

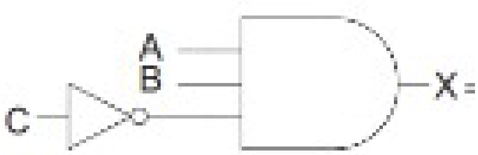


8. Exercice

[Solution n°1 p 29]

Donnez l'équation de sortie des fonctions représentées par les logigrammes suivants

Figure :



- X = A.B.not(C)
- X = C.not(B).A
- X= A + B + C.B

9. Exercice

[Solution n°2 p 29]

Donnez l'équation de sortie de la fonction représentée par le logigramme suivant

Figure

- X = A.B.not(CB)
- X= (not(not(A) + BC)).A.B
- X= (not(A) + BC).A.B.C

Partie II : Les propriétés de l'algèbre de BOOLE

II

Objectifs

Manipuler les fonctions combinatoires et les propriétés de l'Algèbre de BOOLE;

En électronique numérique on manipule des variables logiques conventionnellement repérées par les valeurs 0 ou 1.

Ces grandeurs obéissent à des règles d'algèbre particulières qu'il est indispensable de maîtriser avant d'entreprendre l'analyse ou la synthèse de circuits numériques.

Dans cette partie nous énoncerons les principes et les règles de calcul de l'algèbre logique, appelé aussi *algèbre de Boole*, puis nous les appliquerons à l'écriture et à la manipulation des fonctions logiques.

1. 1. Propriétés de bases de l'algèbre de BOOLE

Résumé des Propriétés de bases

Théorèmes

- Commutativité

  $a + b = b + a$ (commutativité de l'opération OU)

  $a.b = b.a$ (commutativité de l'opération ET)
- Associativité

  $a + (b + c) = (a + b) + c = a + b + c$ (associativité de l'opération OU)

  $(ab)c = a(bc) = abc$ (associativité de l'opération ET)
- Distributivité

  $(a + b).c = ac + bc$ (distributivité du produit logique sur la somme logique)

  $ab + c = (a + c).(b + c)$ (distributivité de la somme logique sur le produit logique)

Les parenthèses imposent une priorité supérieure.

Les autres propriétés

Théorèmes

$a + 0 = a$	**Element neutre**	$a \,.\, 1 = a$
$a + 1 = 1$	**Element absorbant**	$a \,.\, 0 = 0$
$a + a = a$	**Idem potence (redondance)**	$a \,.\, a = a$
$a + \bar{a} = 1$	**Propriété du complément**	$a \,.\, \bar{a} = 0$
$a + b = b + a$	**Commutativité**	$a \,.\, b = b \,.\, a$
$a + b + c = a + (b + c) = (a + b) + c$	**Associativité**	$a\,b\,c = a\,(b\,c) = (a\,b)\,c$

Les propriétés déduites

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

Exemple : x + x y = x

Règle du multiple du complément : a +not(a).b = (a + not(a)) (a + b) = a + b (*Absorption 2*)

Résumé des Propriétés

Propriétés

Propriété	OU	ET
Identité	$a+0=a$	$a.1=a$
Élément neutre	$a+0=a$	$a.1=a$
Élément absorbant	$a+1=1$	$a.0=0$
Idempotence	$a+a=a$	$a.a=a$
Complémentation	$a+\overline{a}=1$	$a.\overline{a}=0$
Involution	$\overline{\overline{a}}=a$	$\overline{\overline{a}}=a$
Commutativité	$a+b=b+a$	$a.b=b.a$
Associativité	$a+(b+c)$ $=(a+b)+c$	$a.(b.c)$ $=(a.b).c$
Distributivité	$a+(b.c)$ $=(a+b).(a+c)$	$a.(b+c)$ $=(a.b)+(a.c)$
Absorption 1	$a+a.b=a$	$a.(a+b)=a$
Absorption 2	$a+\overline{a}.b=a+b$	$a.(\overline{a}+b)=a.b$
Consensus	$a.b+\overline{a}.c+bc$ $=a.b+\overline{a}c$	$(a+b).(\overline{a}+c).(b+c)$ $=(a+b).(\overline{a}+c)$
	$(a+b).(\overline{a}+b)=(a.b)+(\overline{a}.b)$	
De Morgan	$\overline{a+b}=\overline{a}.\overline{b}$	$\overline{a.b}=\overline{a}+\overline{b}$

2. 2. Théorème de DE MORGAN

But

Exprimer les opérateurs ET, OU et NON exclusivement à l'aide d'opérateurs *NOR* seul ou *NAND* seul.

On dit que les opérateurs *NAND* et *NOR* sont *universels ou complets.*

Formule

Théorème de De Morgan :

1. *Le complément d'un produit est égal à la somme des compléments des termes du produit :* $\overline{S}=\overline{a.b}=\overline{a}+\overline{b}$
2. *Le complément d'une somme est égal au produit des compléments des termes de la somme :* $\overline{S}=\overline{a+b}=\overline{a}.\overline{b}$

Ce théorème est très utile pour simplifier des expressions ;

Il est aussi valable également si a ou b sont des expressions contenant plusieurs variables.

Conséquences

Résumé

1. une porte NON-OU est une porte ET avec ses entrées inversées :

2. une porte NON-ET est une porte OU avec ses entrées inversées :

Remarque : Universalité des portes NAND et NOR

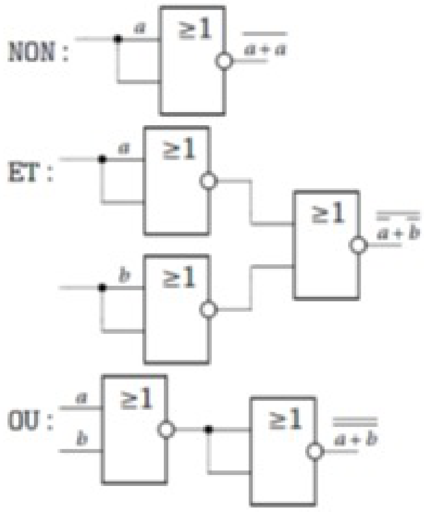
Résultat :

Universalité des portes NON-ET et des portes NON-OU :
Toutes les portes logiques élémentaires (ET, OU, NON) peuvent être réalisées avec des portes NON-OU ou NON-ET.

3. 3. Universalité des portes NON-OU et NON-ET

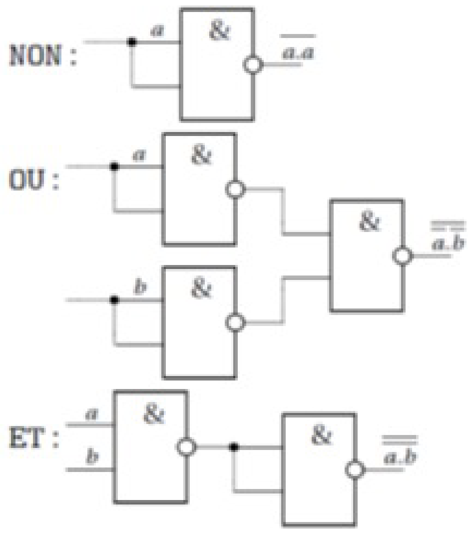
Porte NON-OU

Représentations



Porte NON-ET

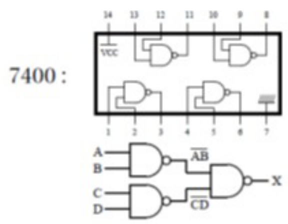
Représentations

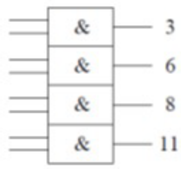


Exemple

Réaliser la fonction $X=AB+CD$ à l'aide du CI (circuit intégré) NON-ET :

$X=AB+CD$



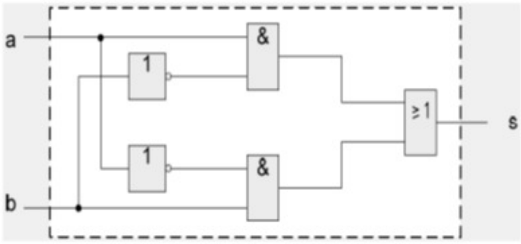


Logigramme

Le logigramme est représentation symbolique des fonctions logique.

Exemple : Logigramme

Exemple : s = a.not(b) + not(a).b

s : 

4. 4. Application à la simplification des fonctions par la méthode algébrique

La simplification algébrique consiste à l'application des théorèmes et propriétés de l'algèbre de BOOLE.

Exemple : 1.

Simplification

$$x+\bar{x}y = x(1+y)+\bar{x}y = x+xy+\bar{x}y = x+y \quad \text{(théorème d'allègement)}$$
$$x.(x+y) = x+xy = x \quad \text{(absorption)}$$
$$ABC + A\bar{B}C + AB\bar{C} + \bar{A}BC = AC + AB + BC$$

Exemple : 2.

Simplification

$Z = ABC \# A\bar{B}(\overline{\bar{A}\bar{C}})$	
$Z = ABC \# A\bar{B}(\bar{\bar{A}} \# \bar{\bar{C}})$	théorème 18 (De Morgan)
$Z = ABC \# A\bar{B}(A \# C)$	annulation de la double complémentation
$Z = ABC \# A\bar{B}A \# A\bar{B}C$	distribution du ET
$Z = ABC \# A\bar{B} \# A\bar{B}C$	théorème 3
$Z = AC(B \# \bar{B}) \# A\bar{B}$	mise en facteur
$Z = AC(1) \# A\bar{B}$	théorème 8
$Z = AC \# A\bar{B}$	théorème 2

5. Exercice

[Solution n°3 p 29]

Quelle est la simplification possible de la fonction ci-dessous :

Z :

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

- ◯ Z = A(B + C)
- ◯ Z = BA + CB
- ◯ Z = BC + CAB

Partie III : Réalisation de circuit logique Combinatoire

III

Objectifs

Manipuler l'analyse booléenne des fonctions combinatoire par la résolution de problèmes numériques

1. 1. Conception de circuit logique combinatoire

Les Étapes

Les étapes de conception d'un système logique combinatoire à partir d'un cahier de charge sont les suivantes :

- Construire la table de vérité selon le cahier de charge.
- Écrire l'expression Booléenne relative à la table de vérité.
- Simplifier l'équation.
- Réaliser le schéma de l'équation simplifiée.

a - Construire la table de vérité selon le cahier de charge.

La table de vérité permet de spécifier le fonctionnement du circuit.

Construire une table de vérité c'est donc *traduire une donnée de problème ou un cahier de charge d'un client en un tableau attribuant des 1 ou des 0 comme valeur des variables d'entrée et de sortie.*

Il faut être donc capable d'identifier les entrées et sorties du système ensuite prévoir toutes les éventualités sur les variables d'entrée. C'est certainement le point le plus important de la conception. Une table de vérité ne peut être correcte que si les données sont parfaitement comprises.

Les variables d'entrée ne sont autres que les *conditions liées au fonctionnement du système.*

Les variables de sorties sont les *résultats attendus.* Un système qui a *n variables* d'entrées présentera un maximum de 2^n *éventualités* (ou combinaisons).

b - Écrire l'expression Booléenne relative à la table de vérité (TVR)

La TVR permet de faire sortir l'expression de l'équation du(des) sortie(s) en fonction des entrées.

Méthode

- Pour chaque cas de la table qui donne 1 en sortie, on écrit le produit logique (terme ET) qui lui correspond,
- On somme logiquement (opérateur OU) ensuite tous les produits logiques constitués, ce qui donne l'expression définitive de la sortie.

Exemple

TVR

No	C	B	A	X	Equation minterme
0	0	0	0	0	
1	0	0	1	0	
2	0	1	0	1	$\overline{C}B\overline{A}$
3	0	1	1	1	$\overline{C}BA$
4	1	0	0	0	
5	1	0	1	0	
6	1	1	0	0	
7	1	1	1	1	CBA

$$X = \overline{C}B\overline{A} + \overline{C}BA + CBA$$

c – Simplification de l'équation

Dès qu'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 moins de variables par terme.

Cette nouvelle équation peut alors servir de modèle pour construire un circuit entièrement équivalent au circuit original mais qui requiert moins de portes ;

--> L'objectif est donc de réaliser le *circuit le plus simple possible.*

U*ne première façon est fondée sur l'application des théorèmes de l'algèbre booléenne (simplification algébrique); cette méthode a été déjà présenté.*

Limite :

Malheureusement, il n'est pas toujours facile de savoir quels théorèmes il faut invoquer pour obtenir le résultat minimal.

--> D'où l'utilisation d'une méthode graphique dite " *Tableau de KARNAUGH* "

2. 2. Méthode de KARNAUGH

a. Principe de la méthode (ADJACENCE DE DEUX TERMES)

Deux termes sont *adjacents* quand ils ne diffèrent l'un de l'autre que par *une seule variable.*

ABC et AB.not(C), (soit 111 et 110) sont adjacents.

Un digramme de *KARNAUGH* est une *table* d'implication logique disposée de telle manière que deux termes *logiquement adjacents* soient également *adjacents géométriquement.*

Comme la table de vérité, il met en évidence le *rapport entre les entrées et les sorties.*

Chaque ligne de la table de vérité correspond à une case du diagramme de KARNAUGH.

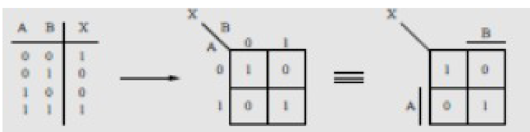
C'est un tableau de 2^n *cases, n étant le nombre de variables d'entrées.*

Dans chacune des cases, on place l'état de la sortie pour les combinaisons d'entrée correspondante.

Exemple : Tableau de KARNAUGH de deux (2) variables

Pour 2 variables, on a $2^2 = 4$ cases de la table de KARNAUGH

TVR convertit en Tableau de KARNAUGH

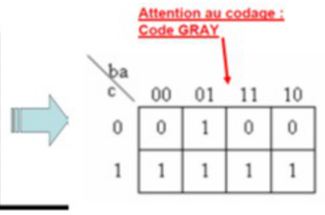


Exemple : Tableau de KARNAUGH de trois (3) variables

Pour 3 variables, on a $2^3 = 8$ cases de la table de KARNAUGH

TVR convertit en Tableau de KARNAUGH

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



Remarque : Valeur binaire des variables d'entrées

Le positionnement des valeurs binaires des variables d'entées obéit au *code binaire GRAY.*

Pour deux variables on a : 00, 01, 11 et 10.

Les valeurs dans chaque case, représente l'état possible de la sortie en fonction des combinaisons des variables d'entées.

Dans le cas précédent : la valeur "1" sur la ligne au dessus, correspond pour *ba=01 et c=1*

b. Simplification

La simplification par KARNAUGH, consiste à faire des regroupement des cases adjacents (voisins) qui sont à "1".

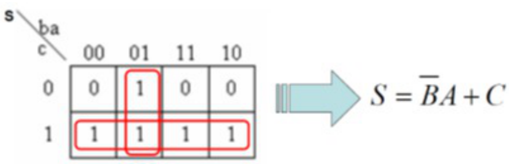
Il est possible de regrouper les cases par *2, 4, 8,* 2^n afin d'éliminer les variables qui changent d'état dans le regroupement :

- un regroupement de *2 cases* élimine *1 variable* ;

- un regroupement de 2^n *cases* élimine *n variables.*

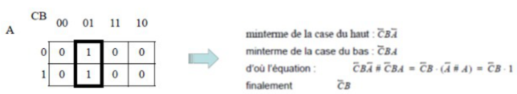
Exemple : Pour 3 variables, Tableau de KARNAUGH

Les regroupements



Les réunions de doublets (deux variables voisins à 1)

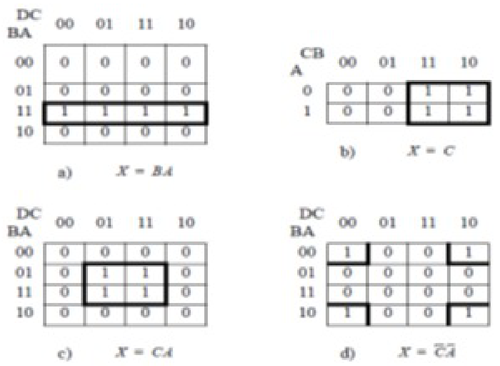
S :



Réunion de quartets (groupes de quatre)

On peut avoir les différents cas de réunion de quartets suivant

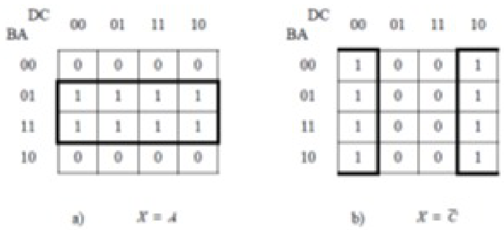
S :



Réunion d'octets (groupes de huit)

On peut avoir les différents cas de réunion de d'octets suivant

S :



Méthode : Simplification

1 Dessinez la table de KARNAUGH et placez des 1 dans les carrés correspondant aux lignes de la table de vérité dont la sortie est 1. Mettez des 0 dans les autres carrés

2 Etudiez la table de KARNAUGH et repérez tous les groupes possibles. Trouvez les plus grands.

3 Commencez ensuite par encercler les 1, dit isolés, qui ne font parties que d'un seul groupe. Cela signifie que pour ces 1, il n'existe qu'une seule possibilité de groupement.

4 Continuer ensuite à prendre les groupes les plus grands qui incluent des 1 (au minimum un seul) qui ne font pas partie d'un autre groupe.

5 Vous devez prendre tous les 1 de la table de KARNAUGH. Il est possible d'utiliser plusieurs fois le même 1.

6 Effectuez un OU logique entre tous les termes résultant des réunions.

3. Exercice

[Solution n°4 p 29]

1. *Quelle est la table de KARNAUGH correspondant à la table de vérité ci dessous :*

TVR :

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

a -

B \ A	0	1
0	1	1
1	0	0

b -

B \ A	0	1
0	0	1
1	1	1

c -

B \ A	0	1
0	1	1
1	1	0

- ○ a
- ○ b
- ○ c

4. Exercice

[Solution n°5 p 29]

2. Quelle est l'équation résultante de cette table de KARNAUGH ?

- ○ a. X = BAD + CD
- ○ b. X = BAC + CDB
- ○ c. X = A + B
- ○ d. X = AD + BD

5. Exercice

[Solution n°6 p 30]

Quelle est la table de KARNAUGH correspondant à la table de vérité

TVR :

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

a -

A \ CB	00	01	11	10
0		1		
1	1			1

b -

A \ CB	00	01	11	10
0		1		
1	1	1	1	1

c -

A \ CB	00	01	11	10
0		1	1	
1	1		1	1

- ○ a -
- ○ b -
- ○ c -

6. Exercice

[Solution n°7 p 30]

Quelle est l'équation résultante de cette table de KARNAUGH ?

NB : not(X) = inverse de X.

- ○ S = BAD + CD
- ○ S = BAC + CDB
- ○ S = AB + BC
- ○ S = not(C).B + A

IV

Partie IV : Application

Objectifs

- Manipuler l'analyse et la résolution de problèmes numérique

1. 1. Méthode de résolution d'un problème numérique

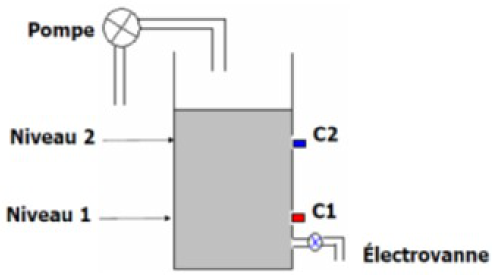
Procédure de résolution de Problèmes

- Lire et comprendre le sujet
- Faire ressortir les variables d'entrées et les variables de sorties
- Établir la table de vérité qui donne les sorties en fonction des entrées
- Faire la simplification par KARNAUGH pour obtenir le circuit le plus optimal possible.
- ÉcrireEcrire l'équation définitive et réaliser le schéma ou logigramme du circuit.

2. 2. Application 1 : Remplissage d'un réservoir

Montage

soit :



On se propose d'automatiser par un circuit logique le remplissage et la vidange d'un réservoir

Lorsque le niveau d'eau est inférieur au niveau 1 (Capteur C1), on déclenche la pompe pour remplir le réservoir.

Lorsque Niveau d'eau est supérieur au niveau 2, on commande l'électrovanne pour vider le réservoir.

Les questions suivantes peuvent être posées :

1. Donner le circuit équivalent (sans simplification)
2. Donner le circuit simplifié.

Résolution

La lecture attentive du sujet doit nous permettre de ressortir les variables d'entées et de sorties.

Les variables d'entrées sont les paramètres dont le changement d'état impact le résultat attendu, c'est-à-dire le ou les sorties.

En variables d'entrées : on peut noter les capteurs de niveaux.

C1 : pour indiquer le niveau haut

C2 : pour indiquer le niveau bas.

Sorties :

nous pouvons noter deux sorties :

P : la pompe qui doit s'ouvrir pour le remplissage du réservoir.

La pompe est ouverte : P=1, si le niveau de l'eau est inférieur à C1, soit C1= 0.

La pompe est Fermé : P=0, si le niveau de l'eau est supérieur à C2, soit C2 = 1.

V : Électrovanne, elle est s'ouvre (V =1) quand le niveau de l'eau est supérieur à C2. Elle est fermé dans le cas contraire.

L'étape suivante consiste à établir la table de vérité, des sortie en fonction des entrées.

La table de vérité permettra de faire resortir les équations des sorties en fonction des variables d'entrées.

Ensuite, on simplifie les expressions en utilisant la méthode algébrique ou le tableau de KARGNAUGH.

3. Exercice

[Solution n°8 p 30]

Remplissage de réservoir.

En s'inspirant des explications, quelle est l'état de P et V si on a :

C1 = 1 et C2 = 1

- ○ P = 1 et V = 0
- ○ P = 1 et V = 1
- ○ P = 0 et V = 0
- ○ P = 0 et V = 1

Solutions des exercices

> Solution n°1

Exercice p. 14

- (•) X = A.B.not(C)
- ( ) X = C.not(B).A
- ( ) X= A + B + C.B

La variable " C ", est inversé avant la porte ET

> Solution n°2

Exercice p. 14

- ( ) X = A.B.not(CB)
- (•) X= (not(not(A) + BC)).A.B
- ( ) X= (not(A) + BC).A.B.C

> Solution n°3

Exercice p. 18

- (•) Z = A(B + C)
- ( ) Z = BA + CB
- ( ) Z = BC + CAB

on met le terme "AB" en facteur, soit ABC + AB.not(C)= AB(C + not(C)), or d'après les propriétés de l'algèbre de BOOLE, C + not (C) = 1.

D'ou : Z = AB + A.not(B).C

= A(B + not(B).C), d'après les propriétés de boole x + not(x).y = x + y

Alors Z = A(B + C).

> Solution n°4

Exercice p. 23

- ( ) a
- (•) b
- ( ) c

> Solution n°5

Exercice p. 24

- ( ) a. X = BAD + CD

○ b. X = BAC + CDB

◉ c. X = A + B

○ d. X = AD + BD

X :

B \ A	0	1
0	0	1
1	1	1

La réunion verticale des deux "1" donne : A

La réunion horizontale des deux "1" donne : B

D'où la solution : X = A+B

> Solution n°6

Exercice p. 25

○ a -

◉ b -

○ c -

S = 1 dans les cas suivants : BCA = 010, 001, 011, 111 et 101.

> Solution n°7

Exercice p. 26

○ S = BAD + CD

○ S = BAC + CDB

○ S = AB + BC

◉ S = not(C).B + A

S :

A \ CB	00	01	11	10
0		1		
1	1	1	1	1

La réunion de doublets verticale donne : not(C).B, (not (C) = inverse de C) car dans le passage de la première ligne à la deuxième, la variable A change de valeur, elle est donc annulée.

La réunion de quartets horizontale donne : A, car sur le long de la ligne, les deux autres variables changent de valeur au moins une fois.

D'où la solution : S =not(C).B + A

> Solution n°8

Exercice p. 28

○ P = 1 et V = 0

○ P = 1 et V = 1

- ○ P = 0 et V = 0
- ◉ P = 0 et V = 1

si C2 = 1, cela signifit que que l'eau a atteint le niveau haut .
il faut donc fermer la pompe (P=0) et ouvrir l'électrovanne (V=1)