LES AUTOMATISMES

ALGEBRE LOGIQUE

Lycée L.RASCOL 10,Rue de la République
BP 218. 81012 ALBI CEDEX

SOMMAIRE

BASES DE NUMERATION

De tout temps, l'homme a cherché à compter avec plus ou moins de réussite. Les romains ont mis en place un système de numération basé sur des symboles littéraux rendant difficile le calcul.

C'est vers l'an 750 en Inde, qu’est mise en place la numération décimale que nous utilisons aujourd'hui. Elle se généralise en Europe par les arabes vers l'an 1200 (on parle de chiffres arabes).

La numération décimale (Base 10) est la base universelle du fait que nous l'utilisons tout le temps. Elle fait appel à des principes fondamentaux que l'on retrouvera dans toutes les bases de calcul.

Premier principe : Un nombre est constitué de chiffres qui pourront prendre une valeur comprise entre ‘0’ et ‘Base-1’ ; de ‘0’ à ‘9’pour la numération décimale.

Deuxième principe : En fonction de sa position dans le nombre chaque chiffre a une signification parfaitement définie. Soit par exemple le nombre $A_{(10)} = 12041$

1 2 0 4 **1**

1 signifie que l'on a une fois l'unité.

1 signifie que l’on a dix mille fois l’unité.

Le système décimal est dit pondéré 1,10, 100, 1000, 10000, ….

Troisième principe : Le zéro matérialise une position où il y a absence d'éléments. On ne représentera donc que les zéros significatifs.

En informatique industrielle, chaque signal n'ayant que deux états possibles, on utilisera la numération en base deux (ou **numération binaire).**

Elle utilise deux symboles 0 et 1. Cette base est très commode pour distinguer les deux états logiques fondamentaux (vrai et faux).

Un état binaire est appelé **bit** (contraction de *binary digit*), un bit prend les valeurs 0 ou 1.
On associera plusieurs bits pour constituer :

- un quartet 4 bits
- un octet 8 bits
- un mot 16 bits
- un mot double 32 bits
- un mot long 64 bits

On écrit : $A_{(2)}$ = **1**001 011**1**

Les puissances successives de 2 (2^0, 2^1, 2^2,2^3, ….2^n...) sont appelées **poids binaires.**
Le système binaire est pondéré 1, 2, 4, 8, 16, 32, 64, ….
Le bit de poids le plus fort est appelé MSB (***M****ost* ***S****ignificant* ***B****it*).
Le bit de poids le plus faible est appelé LSB (***L****ess* ***S****ignificant* ***B****it*).

En base deux avec un nombre de *n* bits on peut avoir 2^n valeurs différentes :

- la valeur la plus petite est toujours 0,
- la valeur la plus grande correspond à (2^n-1).

Avec un octet on peut réaliser 2^8 = 256 valeurs différentes
valeur minimum 0
valeur maximum 2^8-1=255

CORRESPONDANCE ENTRE LES BASES

Système décimal
- usuellement utilisé,
- il est pondéré 1 . 10 . 100 . 1000 …..

Système binaire
- il est pondéré 1 . 2 . 4 . 8 …..
- il est auto complémentaire,
- il peut apparaître des combinaisons parasites **Aléas**

Système binaire réfléchi « GRAY »
- il ne peut pas apparaître des combinaisons parasites **aucun risque d'Aléas**
- il est auto complémentaire
- **il n'est pas pondéré**

Système octal
- il est pondéré 1 . 8 . 64 . 512 …..

Peut utilisé aujourd'hui, il était utilisé dans les systèmes Informatiques de première génération !

Système Hexadécimal
- il est pondéré 1 . 16 . 256 . 4096 …..
- Simplification du codage, avec un seul coefficient on indique la valeur d'un quartet

Système Décimal Codé Binaire « DCB »
- codage binaire des 10 premières valeurs du code décimal,
- il est pondéré 1 . 2 . 4 . 8
- il n'est pas auto complémentaire

Décimal		Binaire				GRAY				Octal		Hexa	DCB			
a_1	a_0	a_3	a_2	a_1	a_0	a_3	a_2	a_1	a_0	a_1	a_0	a_0	a_3	a_2	a_1	a_0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	1	0	1	1	0	0	0	1
0	2	0	0	1	0	0	0	1	1	0	2	2	0	0	1	0
0	3	0	0	1	1	0	0	1	0	0	3	3	0	0	1	1
0	4	0	1	0	0	0	1	1	0	0	4	4	0	1	0	0
0	5	0	1	0	1	0	1	1	1	0	5	5	0	1	0	1
0	6	0	1	1	0	0	1	0	1	0	6	6	0	1	1	0
0	7	0	1	1	1	0	1	0	0	0	7	7	0	1	1	1
0	8	1	0	0	0	1	1	0	0	1	0	8	1	0	0	0
0	9	1	0	0	1	1	1	0	1	1	1	9	1	0	0	1
1	0	1	0	1	0	1	1	1	1	1	2	A				
1	1	1	0	1	1	1	1	1	0	1	3	B				
1	2	1	1	0	0	1	0	1	0	1	4	C				
1	3	1	1	0	1	1	0	1	1	1	5	D				
1	4	1	1	1	0	1	0	0	1	1	6	E				
1	5	1	1	1	1	1	0	0	0	1	7	F				

CHANGEMENT DE BASE

PASSAGE D'UNE BASE « B » VERS LE SYSTEME DECIMAL

*** La base B doit être pondérée**

$$A_{(B)} = a_3 \; a_2 \; a_1 \; a_0 \qquad \text{avec} \qquad 0 < a_i < (B-1)$$

A ce nombre sera associé une valeur numérique en base 10 :

$$A_{(10)} = a_3 B^3 + a_2 B^2 + a_1 B^1 + a_0 B^0$$

$$\mathbf{192}_{(10)} = \mathbf{1}.10^2 + \mathbf{9}.10^1 + \mathbf{2}.10^0$$

A $_{(2)}$ **= 1011**

A $_{(10)}$ **=** $\mathbf{1}.2^3 + \mathbf{0}.2^2 + \mathbf{1}.2^1 + \mathbf{1}.2^0$

A $_{(10)}$ **= 11**

A $_{(8)}$ **= 34**

A $_{(10)}$ **=** $\mathbf{3}.8^1 + \mathbf{4}.8^0$

A $_{(10)}$ **= 28**

PASSAGE DU SYSTEME DECIMAL VERS UNE AUTRE BASE

• ***décimal vers binaire***

A $_{(10)}$ **= 49**

A $_{(2)}$ **= 110001**

• ***décimal vers octal***

A $_{(10)}$ **= 49**

A $_{(8)}$ **= 61**

$A_{(2)}$ = 1011 1100 0011 0100

A (10) = $2^{15} + 2^{13} + 2^{12} + 2^{11} + 2^{10} + 2^{5} + 2^{4} + 2^{2}$

A (10) = 32768 + 8192 + 4096 + 2048 + 1024 + 32 + 16 + 4

A (10) = 48180

48180 | 16
4 | 3011 | 16
3 | 188 | 16
12 | 11

A (16) = BC34

C **B**

A (10) = 928

A (2) = 11 1010 0000

A (8) = 1640

A (16) = 3A0

A (DCB) = 1001 0010 1000

928 |
0 | 464
0 | 232
0 | 116
0 | 58
0 | 29
1 | 14
0 | 7
1 | 3
1 | 1

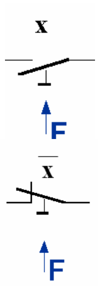
928 | 16
0 | 58 | 16
10 | 3

A

VARIABLE ET OPERATEURS BOOLEENS

VARIABLE

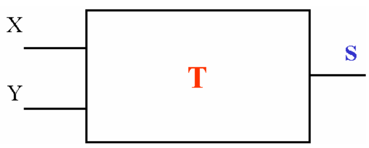
Une variable Booléenne est logique, elle ne peut prendre que deux états 0 ou 1.



action extérieure (physique)	état technologique	état logique	
F = 0	ouvert	0	élément travail
F = 1	fermé	1	
F = 0	fermé	1	élément repos
F = 1	ouvert	0	

OPERATEURS BOOLEENS

Opérateurs sur deux variables



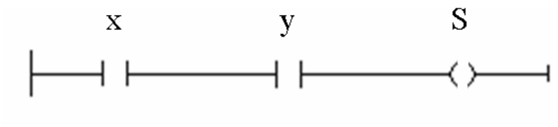
X	Y	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

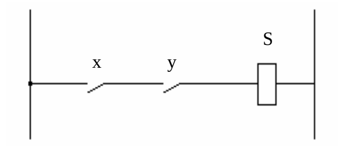
T1	Générateur de 0	**T16** = **/T1**	Générateur de 1
T2	ET	**T15** = **/T2**	ET NON - NAND
T3 et **T5**	Inhibition	**T14** = **/T3** et **T12** = **/ T5**	Implication
T4 et **T6**	Transfert OUI	**T13** = **/T4** et **T11** = **/ T6**	Inverseur NON
T7	OU exclusif - XOR	**T10** = **/T7**	Identité - XNOR
T8	OU	**T9** = **/T8**	OU NON - NOR

REPRESENTATION DES OPERATEURS BOOLEENS

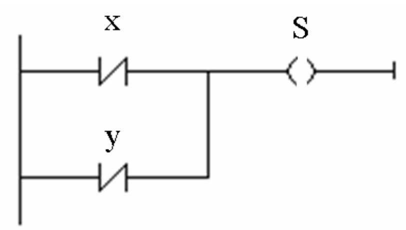
RESEAUX LOGIQUES A CONTACTS

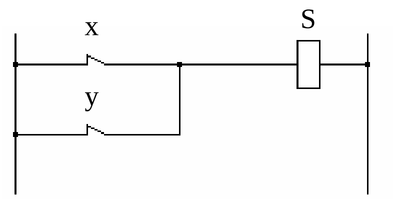
• ***Opérateur ET*** **S = X . Y**





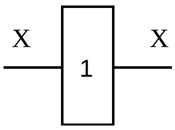
• ***Opérateur NAND*** **S = /(X .Y) = /X + /Y**



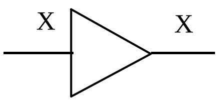


DIAGRAMMES LOGIQUES

Transfert
« OUI »

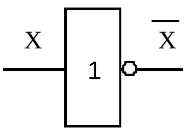


Norme

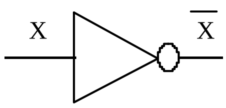


Norme

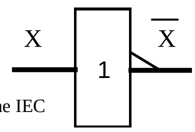
Inverseur - « NON » - « PAS »



Norme AFNOR

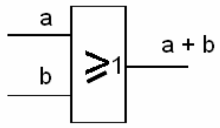


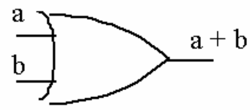
Norme IEEE



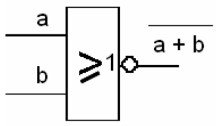
Norme IEC

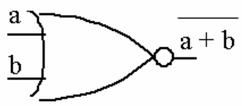
OU



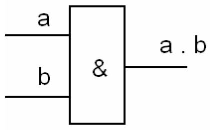


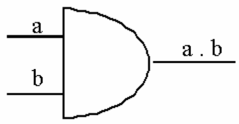
NOR- « OU NON »



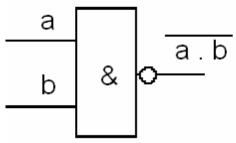


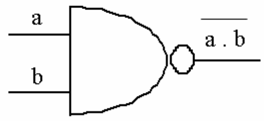
ET



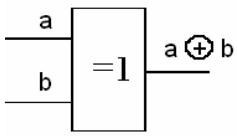


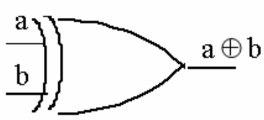
NAND – « ET NON »



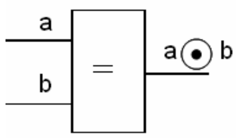


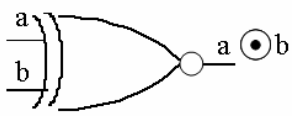
XOR – « OU exclusif »





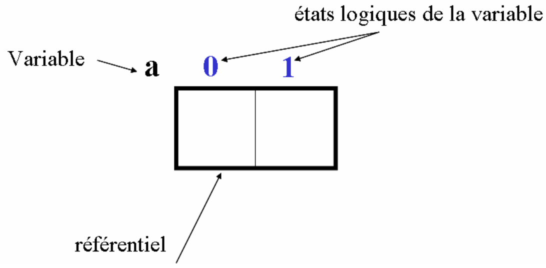
XNOR – « Identité »



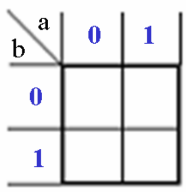


TABLEAUX DE KARNAUGH

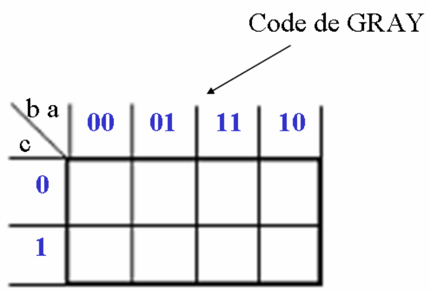
- ***Avec une variable***



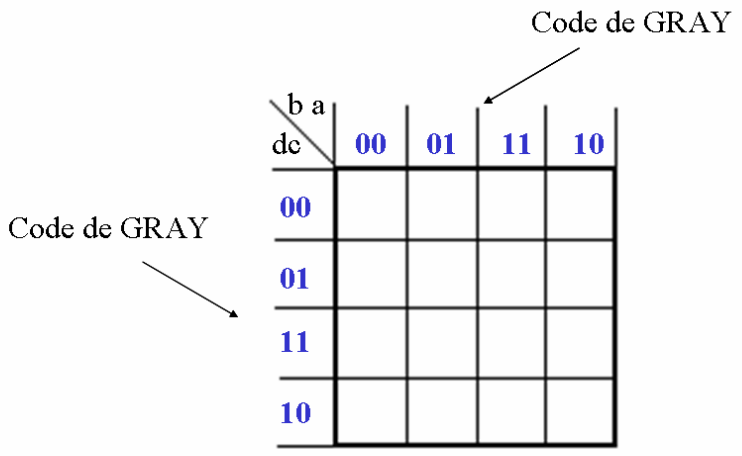
- ***Avec deux variables***



- ***Avec trois variables***



- ***Avec quatre variables***



FONCTIONS BOOLEENNES REPRESENTATION

DEFINITION

On appelle fonction booléenne, une fonction de "n" variables booléennes définies dans l'ensemble (0 - 1), comme les variables, la fonction ne peut donc prendre que deux états possibles 0 ou 1.

variables			$F_1 = f(a,b,c)$	$F_2 = f(a,b,c)$
c	b	a		
0	0	0	0	1
0	0	1	0	0
0	1	0	0	1
0	1	1	1	0
1	0	0	1	1
1	0	1	1	1
1	1	0	1	X
1	1	1	1	X

• Une fonction booléenne est <u>complètement définie</u> si elle prend 0 ou 1 pour chacune des combinaisons possibles des variables.

• Une fonction booléenne est <u>incomplètement définie</u> si certaines combinaisons des variables sont :
- Soit impossible,
- Soit sans incidence sur le système, (elles n'apparaîtront jamais).

REPRESENTATION DES FONCTIONS BOOLEENES

Représentation par écriture canonique

A partir de la table de vérité ou tableau des états on tire l'équation de la fonction F1.

$$F1 = a\,b\,/c + /a\,/b\,c + a\,/b\,c + /a\,b\,c + a\,b\,c$$

La fonction F1 est écrite en somme de produits, c'est la **première forme canonique** ou forme disjonctive, chacun des termes est constitué par une combinaison de toutes les variables c'est un **minterme**.
La première forme canonique est une réunion de mintermes.

A partir de la table de vérité ou tableau des états on tire l'équation de la fonction F1 = //F1.

$$\mathbf{F1 = // F1 = (a + b + c) . (/a + b + c) . (a + /b + c)}$$

On obtient ici la **deuxième forme canonique** de F1 ou forme canonique conjonctive, chacun des termes est constitué par une combinaison de toutes les variables c'est un **maxterme**.

Les deux formes canoniques précédentes font intervenir les opérateurs ET,OU, PAS.
Il existe deux autres formes canoniques faisant intervenir chacune qu'un seul opérateur :

- 3° forme canonique (NAND)

- 4° forme canonique (NOR)

Représentation par rectangles de Karnaugh

c \ b a	00	01	11	10
0	0	0	1	0
1	1	1	1	1

F1

c \ b a	00	01	11	10
0	1	0	0	1
1	1	1	X	X

F2

Représentation par écriture numérique

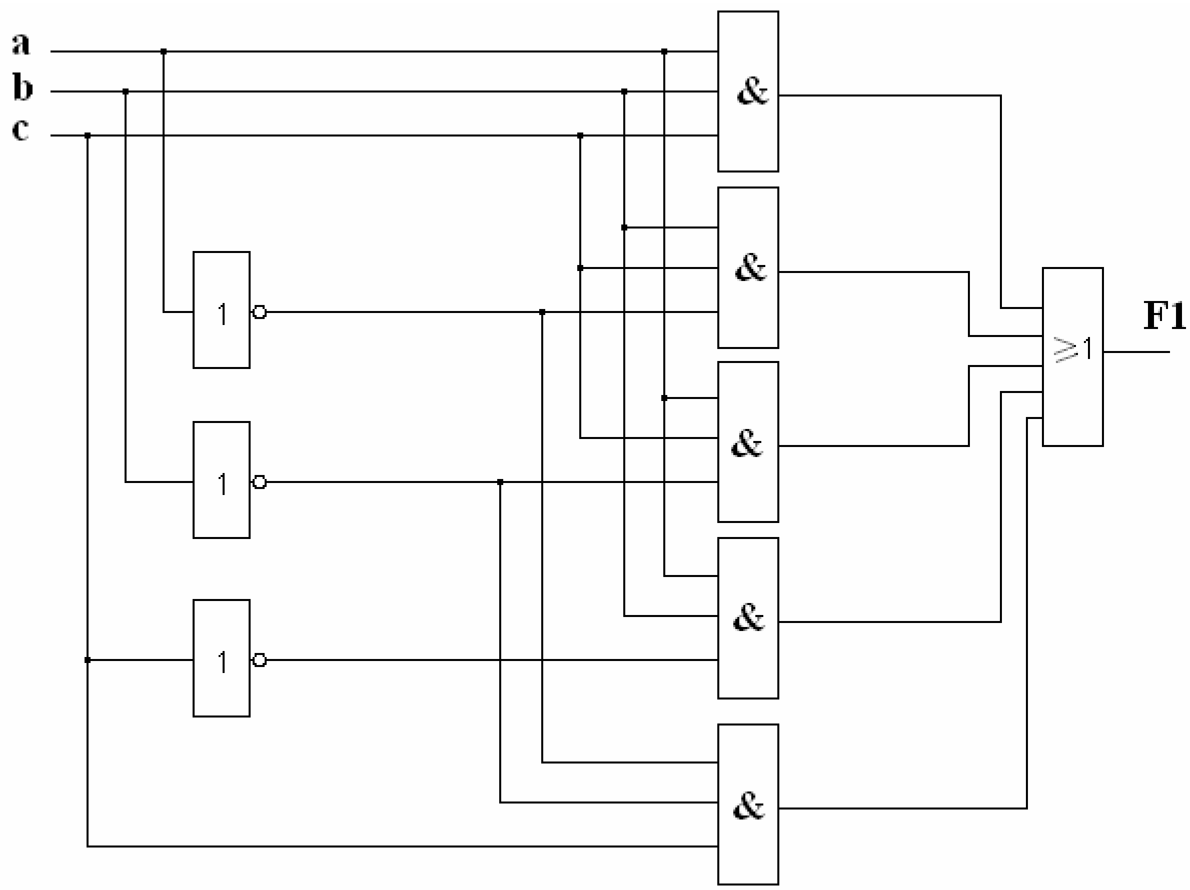
$\mathbf{F1} = \mathbf{R}\ (3, 4, 5, 6, 7)$

$\mathbf{F2} = \mathbf{R}\ (0, 2, 4, 5)\ \ \mathbf{R}_X\ (6, 7)$

Représentation par logigramme

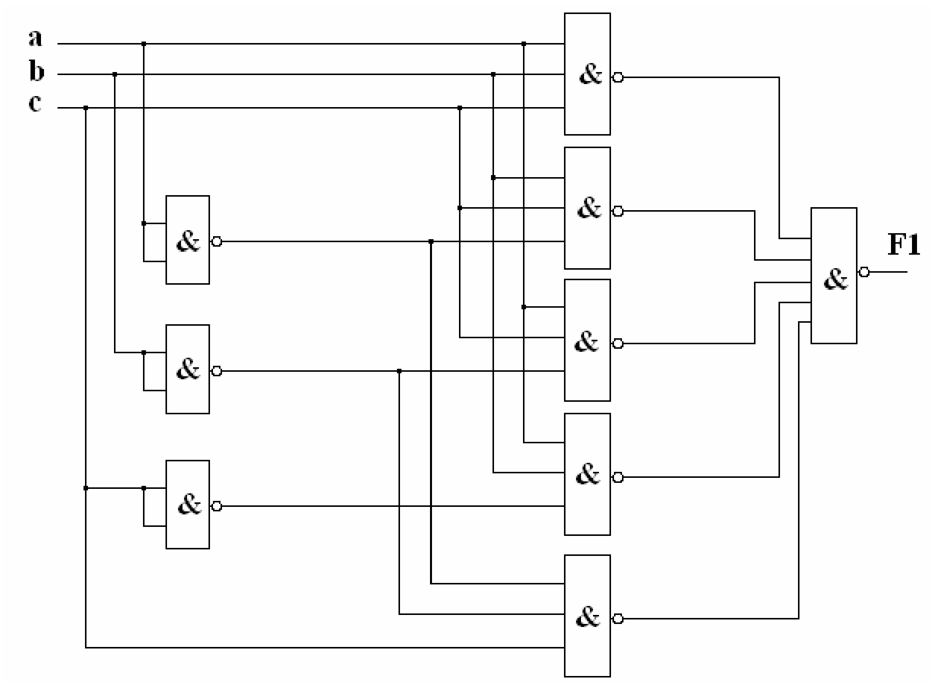
$$F1 = a\ b\ /c + /a\ /b\ c + a\ /b\ c + /a\ b\ c + a\ b\ c$$

Avec tous les opérateurs nécessaires

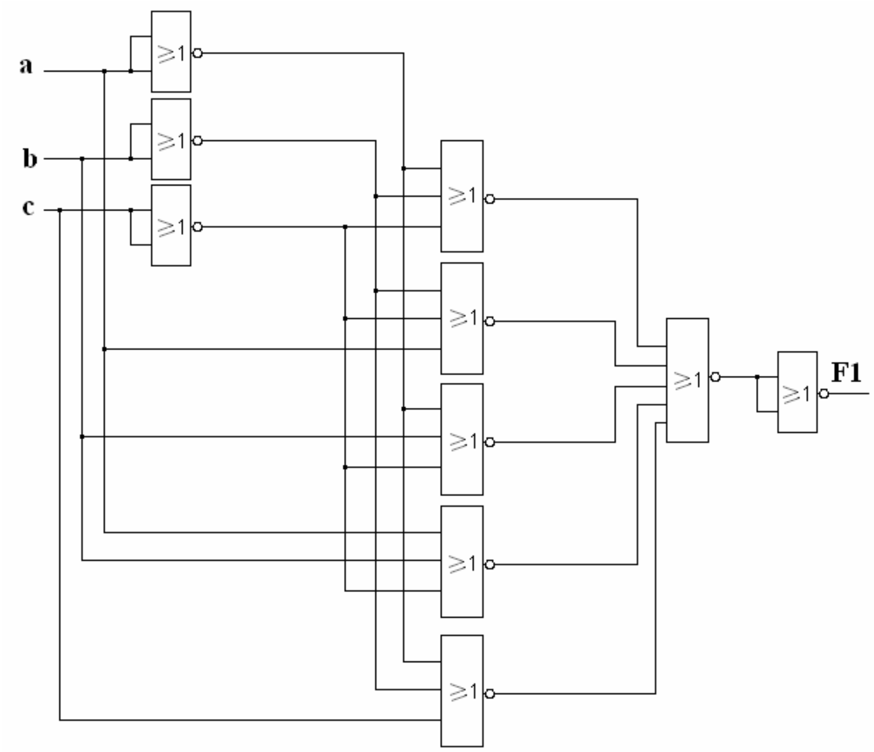


F1 = a b /c + /a /b c + a /b c + /a b c + a b c

Avec uniquement des opérateurs NAND



Avec uniquement des opérateurs NOR



FONCTIONS BOOLEENNES SIMPLIFICATION

On va rechercher la formule la plus condensée, avec le moins de symboles, donc conduisant à une réalisation matérielle plus compacte.

SIMPLIFICATION ALGEBRIQUE

Principaux postulats de l'algèbre de BOOLE

.

• Fonction d'une variable avec 0, 1, elle-même:

$$x + 0 = x \qquad x \,.\, 0 = 0$$

$$x + 1 = 1 \qquad x \,.\, 1 = x$$

$$x + x = x \qquad x \,.\, x = x$$

• Propriété d'une fonction avec son complément

$$x + \overline{x} = 1 \qquad x \,.\, \overline{x} = 0$$

• Théorème de MORGAN

Le complément d'une somme logique est égal au produit de chaque terme complémenté.

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

Le complément d'un produit logique est égal à la somme de chaque terme complémenté.

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

La simplification algébrique repose sur beaucoup d'astuce, au-delà de trois variables cette méthode est pratiquement impossible à mettre en œuvre on peut :

- supprimer les associations de variables multiples.
- mettre en facteur pour faire apparaître des termes complémentaires.
- mettre en facteur pour faire apparaître des termes inclus.
- ajouter une expression qui figure déjà.

F1 = a b c + /a b c + a /b c + a b /c	$F1 = ab + c(a \oplus b)$
F2 = a b + /a b + a c + a /c	F2 = a + b
F3 = (a + b) . (a + c)	F3 = a + bc
F4 = a b c + /a b c + a /b /c + a b /c	F4 = bc + ac
F5 = a b c + a b /c + a /b c	F5 = a (b + c)
F2 = a b + a /b + a c + a /c	F2 = a
F3 = (a + b) . (a + /b)	F3 = a

SIMPLIFICATION PAR LES RECTANGLES DE KARNAUGH

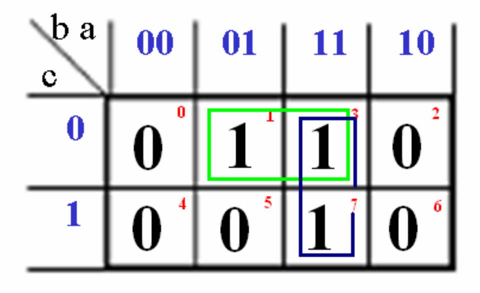
La simplification par les rectangles de KARNAUGH est graphique

- On transporte la table de vérité dans un rectangle de Karnaugh.
- On réalise les groupements possibles de 1 - 2 - 4 – 8 termes en recherchant à avoir le minimum de groupements (le nombre des cases regroupées doit correspondre à une puissance de 2).
- Dans chaque groupement ainsi formé on élimine les termes qui changent d'état (le nombre des termes éliminés doit correspondre à la puissance de 2 du groupement).

Exemples

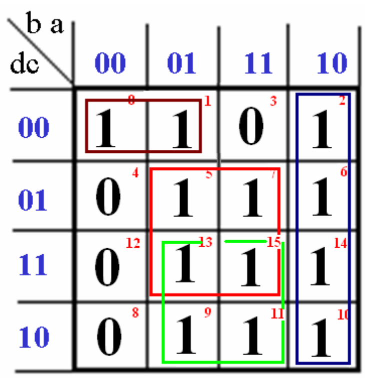
F6 = R (1, 3, 7)

F6 = a /c + ab



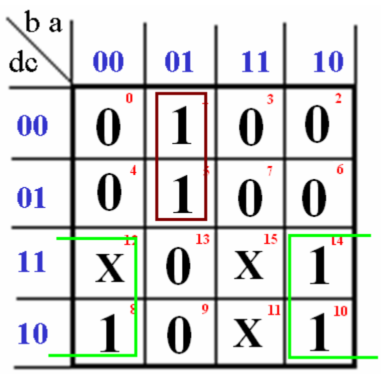
F7 = R (1, 2, 5, 6, 7, 9, 10, 11, 13, 14, 15)

F7 = /a b + a (c + d) + /b /c /d



F8 = R (1, 5, 8, 10, 14) R x (11, 12, 15)

F8 = a /b /d + /a d



EXERCICE *Problème de majorité*

Le bon fonctionnement d'un processus industriel est lié pour des raisons de sécurité à un paramètre (température par exemple) qui doit rester en dessous d'un certain seuil.
Si ce paramètre est surveillé par un seul capteur et si l'arrêt du processus est coûteux on ne peut pas prendre le risque d'arrêter sur avarie du capteur.
On utilise donc simultanément trois capteurs et on regarde si les informations sont concordantes:

- Le processus fonctionnera si une majorité d'informations correctes se dégage,
- Toute discordance sera signalée.

Travail demandé

1) équation de voyant "fonctionnement incorrect".
2) équation du voyant "défaut capteur n°1".
3) équation du voyant "défaut capteur n°2".
4) équation du voyant "défaut capteur n°3".
5) schéma du système en utilisant :
   - uniquement des fonctions NAND,
   - uniquement des fonctions NOR.

c	b	a	fonctionnement incorrect	défaut capteur a	défaut capteur b	défaut capteur c
0	0	0	0	0	0	0
0	0	1	0	1	0	0
0	1	0	0	0	1	0
0	1	1	1	0	0	1
1	0	0	0	0	0	1
1	0	1	1	0	1	0
1	1	0	1	1	0	0
1	1	1	1	0	0	0

c \ b a	00	01	11	10
0	0	1	0	0
1	0	0	0	1

Da = a /b /c + /a b c

c \ b a	00	01	11	10
0	0	0	0	1
1	0	1	0	0

Db = a /b c + /a b /c

c \ b a	00	01	11	10
0	0	0	1	0
1	1	0	0	0

Dc = a b /c + /a /b c

c \ b a	00	01	11	10
0	0	0	1	0
1	0	1	1	1

Fi = a b + ac + b c