Université Mohammed V
Faculté des Sciences
Département de Mathématiques et Informatique
Filière : SMI

Algèbre binaire et Circuits logiques

(2007-2008)

Prof. Abdelhakim El Imrani

Plan

1. Algèbre de Boole
2. Circuits logiques
3. Circuits combinatoires
4. Circuits séquentiels

Algèbre de Boole

George Boole (1815-1864) est un mathématicien autodidacte anglais qui voulait faire un lien entre la logique (étude de la validité du raisonnement) et la représentation symbolique utilisée en mathématique.

Il a écrit deux ouvrages sur le sujet :

- ***Mathematical Analysis of Logic*** **(1847)**
- ***An Investigation of the Laws of Thought*** **(1854)**

Ces travaux n'ont pas connu d'intérêt particulier auprès de la communauté mathématique et scientifique de son époque, mis à part chez les logiciens

Algèbre de Boole

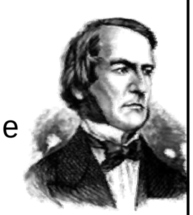
- C'est 70 ans plus tard que les travaux de Boole gagnent l'intérêt de tous, lorsque Claude Shannon fait le lien entre l'algèbre de Boole et la conception des circuits.
- Claude Shannon montre que l'algèbre de Boole peut-être utilisée pour **optimiser** les circuits. Cette nouvelle avenue de recherche va ouvrir la voie à l'ère numérique.

  → « En utilisant **l'algèbre de Boole** avec le **système binaire**, on peut concevoir des **circuits** capables d'effectuer des **opérations arithmétiques et logiques**

- Boole repose sur des axiomes, des postulats et des théorèmes qu'il faut connaître par coeur !

Algèbre de Boole

Propositions vraie ou fausses
et opérateurs sur ces préposition → Algèbre de Boole



- Systèmes binaires: Vrai=1, Faux=0
- C'est le cas des systèmes numériques (circuits logiques)

- L'ordinateur est constitué de circuits logiques
- Élément de base est le transistor, deux états: Bloqué=0, Conducteur=1.

Transistor → Porte logique → Circuit logique → Unité d'un système informatique

Algèbre binaire

Définitions:

- **États logiques :** 0 et 1, Vrai et Faux
- **Variable logique :** Symbole pouvant prendre comme valeur des états logiques (A, b, c, ...)
- **Opérateurs logiques**: Or, And, Not, ...
- **Fonction logique :** Expression de variables et d'opérateurs logiques. (f = not(a) or (b OR c and d)

Éléments de base

- **Variables d'entrée**
  - Les variables d'entrée sont celles sur lesquelles on peut agir directement. Ce sont des variables logiques indépendantes.
- **Variable de sortie**
  - Variable contenant l'état de la fonction après l'évaluation des opérateurs logiques sur les variables d'entrée.
- **Simplification d'une fonction logique**
  - Trouver la représentation (l'écriture) la plus simple de la fonction réalisée: ***Algèbre de Boole***

Algèbre de Boole sur [0,1] = algèbre binaire

Structure d'algèbre de boole

- 2 lois de composition interne (Or, And)
- 1 application unaire (Not)

2 Lois de Composition Interne : ET, OU

Somme (OU, Réunion) $s = a + b$ = a **or** b

Produit (ET, intersection) $s = a \cdot b = ab$ = a **and** b

Nb: a+b se lit « a OU b » pas « a PLUS b »

Application unaire :

Not (complémentation, inversion) $s = \overline{a}$ = **not**(a)

NB: $\overline{a}$ se lit « a barre » ou « non a »

Fonctions logiques

Fonction logique à n variables f(a,b,c,d,...,n)

$$[0,1]^n \longrightarrow [0,1]$$

- Une fonction logique ne peut prendre que deux valeurs (0, 1)
- Les cas possibles forment un ensemble fini (card = 2^n)

La table de fonction logique = table de vérité

Définition : (a, b, c, ..., n) = vecteur d'entrée

Table de vérité

- **Table de vérité**: Enumération ligne par ligne des valeurs prises par f en fonction des valeurs de ses paramètres.

Or

$s = a + b$

S est vrai si a OU b est vrai.

a b	s
0 0	0
0 1	1
1 0	1
1 1	1

And

$s = a . b$

S est vrai si a ET b sont vrais.

a b	s
0 0	0
0 1	0
1 0	0
1 1	1

Not

$s = \overline{a}$

S est vrai si a est faux

a	s
0	1
1	0

Notes sur les tables de vérité

f (a, c, d, .., n) fonction logique à N entrées sera représentée par :

- une table à 2^N lignes

a b c	f(a,b,c)
0 0 0	0
0 0 1	1
0 1 0	0
0 1 1	0
1 0 0	1
1 0 1	0
1 1 0	0
1 1 1	1

Propriétés

- **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)

- **Idempotence**
  - a+a = a
  - a.a = a

- **Absorption**
  - a+a.b = a
  - a.(a+b) = a

Démonstration distributivité

$$a.(b+c) \overset{?}{=} a.b+a.c$$

a	b	c	b+c	a.(b+c)	a.b	a.c	a.b+a.c
0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	0	0	0	0
1	0	0	0	0	0	0	0
1	0	1	1	1	0	1	1
1	1	0	1	1	1	0	1
1	1	1	1	1	1	1	1

= ?

Propriétés (2)

- **Élément neutre**
- $a+0 = a$
- $a.1 = a$
- **Élément absorbant**
- $a+1 =1$
- $a.0 = 0$
- **Inverse**
- $a+\overline{a} = 1$
- $a.\overline{a} = 0$

- **Théorème de DE Morgan**
  - $\overline{a+b} = \overline{a} . \overline{b}$
  - $\overline{a.b} = \overline{a} + \overline{b}$

Équations logiques

On exprime f(a, b, c, ...) par une expression en a, b, c.. et des opérateurs logiques.

Exemple: $f = \bar{a}+b.c.(\bar{d}+e)$

Principe de dualité: Une expression reste vraie si on interverti les 1 par des 0 et les ET par des OU

Exemple: si $a+b=1$ alors $\bar{a}.\bar{b}=0$

Je suis riche si je suis bien payé et que je ne dépense pas tout mon argent = Je suis pauvre si je ne suis pas bien payé ou que je dépense tout mon argent

Les opérateurs NAND, NOR

a b	s
0 0	1
0 1	1
1 0	1
1 1	0

$s = \overline{a.b}$

S est vrai si a OU b est faux.

NAND (No-AND)

a b	s
0 0	0
0 1	0
1 0	0
1 1	1

$s = \overline{a+b}$

S est vrai si ni a, ni b ne sont vrais.

NOR (No-OR ou NI)

L'opérateur : XOR

a b	s
0 0	0
0 1	1
1 0	1
1 1	0

$s = a \oplus b = \overline{a}.b + a.\overline{b}$

S est vrai si a OU b
est vrai mais pas les deux.

XOR (Ou-Exclusif) vaut 1 si a est différent de b
Opérateur de différence (disjonction)

Propriétés du XOR

XOR est associatif $s = a \oplus b \oplus c \oplus \ldots\ldots n$

vaut 1 si le nombre de variable à 1 est impaire.

$$s = \overline{a \oplus b} = \overline{a} \oplus b = a \oplus \overline{b} = a \text{ XNOR } b$$

$$\text{XNOR} = \overline{\text{XOR}} \text{ vaut 1 si } a = b$$

$a \oplus 1 = \overline{a}$ $\qquad a \oplus 0 = a$

Propriétés

$$a \oplus c = b \oplus c \Leftrightarrow a = b$$

$$a \oplus x = b \Leftrightarrow x = a \oplus b$$

Écriture des équations logiques

Définitions: Apparition d'une variable = **Lettre**
Produit de variables sous forme simple ou complémentées = **Monôme**
Somme de monômes = **Polynôme**

$z = a + \overline{b.c.(d + e)}$	Expression algébrique
$= a + \overline{b} + \overline{c} + \overline{(d + e)}$	Développement
$= a + \overline{b} + \overline{c} + \overline{d} . \overline{e}$	Polynôme de 4 monômes de 1 et 2 lettres

Fonctions logiques et formes canoniques

f fonction logique de *n* variables

- On appelle «***minterme***» de n variables, l'un des produits de ces variables ou de leurs complémentaires.

exemple $n = 4$ variables $\{a, b, c, d\}$

$m = a \cdot b \cdot c \cdot d$ est un minterme

$m = \overline{a} \cdot \overline{b} \cdot c \cdot d$ est un autre minterme

$m = a \cdot \overline{b} \cdot c$ n'est pas un minterme

- On appelle «***maxterme***» de n variables, l'une des sommes de ces variables ou de leurs complémentaires.

$M = a + b + c + d$ est un maxterme

$M = \overline{a} + \overline{b} + c + d$ est un autre maxterme

$M = a + \overline{b} + c$ n'est pas un maxterme

Formes canoniques

Une fonction est sous **forme canonique** (ou **normale**) si chaque terme contient toutes les variables. L'écriture sous forme canonique est unique.

Exemples :

$$f(x,y,z) = \underbrace{x.y.z}_{\textbf{Minterme}} + \overline{x}.\overline{y}.z + \overline{x}.y.z$$

Première forme canonique ou forme normale disjonctive

$$f(x,y,z) = \underbrace{(x+\overline{y}+z)}_{\textbf{Maxterme}}.(\overline{x}+y+\overline{z})$$

Deuxième forme canonique ou forme normale conjonctive

Formes canoniques

Si la fonction n'est pas sous forme normale

i.e. une des variables (au moins) ne figure pas dans un des termes

$\longrightarrow$ La fonction est sous une forme simplifiée

$$f(x,y,z) = xyz + \overline{x}y\overline{z} + xy\overline{z}$$ Première forme canonique

$$= xy(z+\overline{z}) + \overline{x}y\overline{z}$$ Forme simplifiée

$$= y(x+\overline{x}\overline{z})$$ Forme simplifiée

$$= y(x+\overline{z})$$ Forme simplifiée

Formes canoniques: Choix

Première forme canonique = expression des 1 de la fonction

Deuxième forme canonique = expression des 0 de la fonction

Les deux formes canoniques sont équivalentes

On choisit celle qui donne le résultat le plus simple
peu de 0 => deuxième forme / peu de 1 => première forme

Simplification des fonctions

Objectif : Fabriquer un système
- à moindre coût
- rapide
- fiable
- peu consommateur

Méthodes : Algébriques
Graphiques
Programmables

Résultat : on cherche la forme minimale d'une fonction
nombre minimal de monômes/nombre minimal de lettre par monôme

Possibilité de plusieurs formes minimales: formes équivalentes

Simplification algébrique

Applications des principes et propriétés de l'algèbre de Boole

Identités remarquables :

1	$a.b+\bar{a}.b=b$	$(a+b).(\bar{a}+b)=b$
2	$a + a.b = a$	$a.(a+b) = a$
3	$a + \bar{a}.b = a+b$	$a.(\bar{a}+b)=a.b$

Démonstrations : 1 et 2 trivial

3 : $a+\bar{a}.b=\underbrace{a.a+a.b}_{a}+\underbrace{a.\bar{a}}_{0}+\bar{a}.b=(a+\bar{a}).(a+b)=a+b$

Simplification algébrique

Règles de simplification :

(Mintermes adjacents = 1 seule variable qui change)

1 : Deux mintermes adjacents → Il reste l'intersection commune

1': Deux maxtermes adjacents → Il reste la réunion commune

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

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

2: On peut ajouter un terme déjà existant à une expression logique.
→ pas de coefficient en algèbre de Boole.

3: On ne change pas le résultat en multipliant l'un des termes par 1 ou en ajoutant 0.

Méthode algébrique toujours possible mais démarche intuitive qui dépend de l'habileté et de l'expérience.

Exercice 1

Remplissez la table de vérité suivante pour prouver le théorème de DeMorgan :

x	y	xy	$\overline{xy}$	$\overline{x}$	$\overline{y}$	$\overline{x}+\overline{y}$
0	0	**0**	**1**	**1**	**1**	**1**
0	1	**0**	**1**	**1**	**0**	**1**
1	0	**0**	**1**	**0**	**1**	**1**
1	1	**1**	**0**	**0**	**0**	**0**

Exercice 2

Considérons la fonction F définie par la table de vérité suivante :

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

Mintermes

$$F = \overline{x}\,y\,z + x\,\overline{y}\,z + x\,y\,\overline{z} + x\,y\,z$$

$$\begin{aligned} F &= \overline{x}\,y\,z + x\,\overline{y}\,z + x\,y\,\overline{z} + x\,y\,z \\ &= (\overline{x}\,y\,z + x\,y\,z) + (x\,\overline{y}\,z + x\,y\,z) + (x\,y\,\overline{z} + x\,y\,z) \\ &= y\,z\,(\overline{x} + x) + x\,z\,(\overline{y} + y) + x\,y\,(\overline{z} + z) \\ &= x\,y + y\,z + z\,x \end{aligned}$$

Exercice 3

- On désire concevoir un circuit qui permet de gérer les notes des examens, on donne: Examen final (45 %), Examen Partiel (35 %), TPs (20 %).
- Un étudiant est admis s'il dispose d'un pourcentage >= 55 %).
  - Exemple: Final=11, Partiel=8, Tps=10 → F=1, P=0, T=1 ⇒ Pourcentage = 65 % → R=1 (étudiant admis).

- Donner la table de vérité.
- Donner la fonction logique correspondante. Simplifier le fonction obtenue.

Simplification graphique: Karnaugh

- La méthode de Karnaugh permet de visualiser une fonction et d'en tirer naturellement une écriture simplifiée.
- L'élément de base de cette méthode est la table de Karnaugh qui représente toutes les combinaisons d'états possibles pour un nombre de variables donné.
- La table de Karnaugh est un outil graphique qui permet de simplifier de manière méthodique des expressions booléennes.
- La construction des tables de Karnaugh exploite le codage de l'information et la notion d'adjacence

Karnaugh – simplification graphique

Principe:

Mettre en évidence sur un graphique les mintermes (ou maxtermes) adjacents. Transformer les adjacences logiques en adjacences «géométriques».

Trois phases:

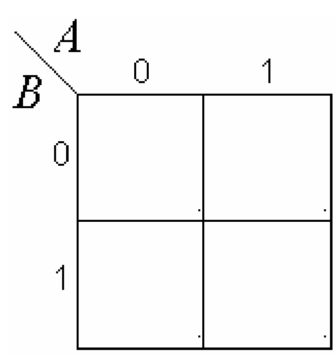
Transcrire la fonction dans un tableau codé, recherche des adjacents pour simplification équations des groupements effectués

Description: **Table de vérité** vs **Tableau de Karnaugh**

Table de vérité	Tableau de Karnaugh
1 ligne	1 case
n variables	2^n cases

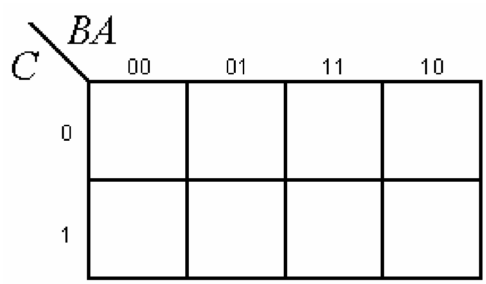
Diagrammes de Karnaugh

- Avec n = 2:
  - Entrées A et B
  - 4 cases



Diagrammes de Karnaugh

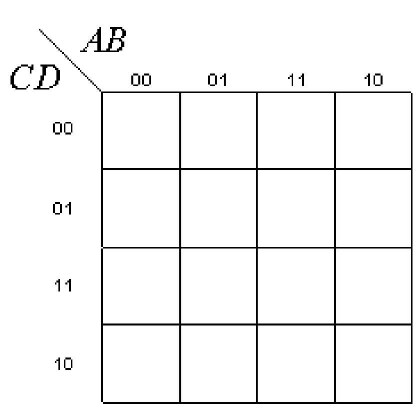
- **Avec n = 3:**
  - Entrées C, B et A
  - 8 cases



Remarque: Une seule variable change d'état entre 2 cases adjacentes

Diagrammes de Karnaugh

- **Avec n = 4:**
  - Entrées D, C, B et A
  - 16 cases



Diagrammes de Karnaugh

- **Avec n = 5:**
  - Entrées x, y, z, t et u
  - 32 cases

tu \ xyz	000	001	011	010	110	111	101	100
00								
01								
11								
10								

Simplification graphique

Exemple: Depuis une table de vérité

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

a \ bc	00	01	11	10
0	0	1	1	1
1	0	0	0	0

Exemple (Karnaugh)

Entrées			Sortie
C	B	A	S
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

TABLE DE VÉRITÉ

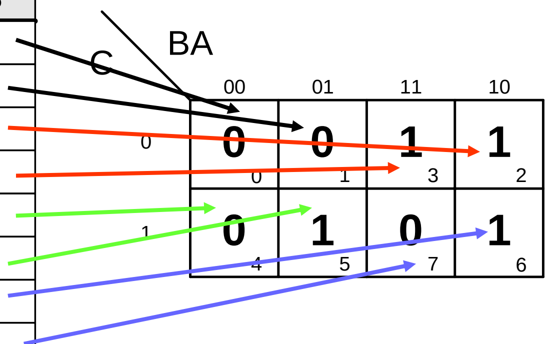


DIAGRAMME DE KARNAUGH

Simplification graphique

Exemple 2: Par une première forme canonique (Par les 1)

a \ bc	00	01	11	10
0		1	1	
1			1	

$$f(a,b,c) = \bar{a}.\bar{b}.c + \bar{a}.b.c + a.b.c$$

Simplification graphique

Exemple 2: Par une deuxième forme canonique (Par les 0)

a \ bc	00	01	11	10
0	0			
1		0		0

$$f(a,b,c) = (a+b+c).(\overline{a}+b+\overline{c}).(\overline{a}+\overline{b}+c)$$

Simplification graphique

Règles de simplification

1 : Les groupements comportent une puissance de deux cases,

2 : Les 2^k cases forment un rectangle,

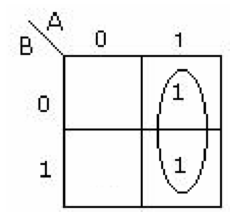
3 : On élimine variable(s) qui change(nt) d'état
Groupement de 2^k cases → On élimine k variables
2 cases → on élimine 1 variable;
4 cases → on élimine 2 variables;
8 cases → on élimine 3 variables;

4 : Il faut utiliser au moins une fois chaque 1, le résultat est donné par la réunion logique de chaque groupement,

5 : Expression minimale si :
- les groupements les plus grands possibles
- utiliser les 1 un minimum de fois

Exemple 1

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

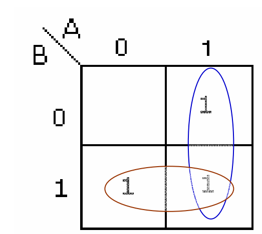


$S = AB + A\overline{B}$, simplification algébrique → $S = A\,(B + \overline{B}) = A$

Karnaught:
Groupement de 2 cases: on élimine variable qui change d'état (B) → S=A

Exemple 2

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



Premier groupement: On élimine B

Deuxième groupement: On élimine A

→ S = A + B

Exemple 3

Tous les 1 sont groupés !

a \ bc	00	01	11	10
0	0	1	1	0
1	1	1	1	0

Equation : $F(a,b,c) = a.\overline{b} + c$

Exemple 4

Par les 0

a \ bc	00	01	11	10
0	0	1	1	0
1	1	1	1	0

Equation : $F(a,b,c) = (a + c).(\overline{b} + c)$

Exemple 5

z \ xy	00	01	11	10
0			1	1
1	1	1	1	1

Z X

z \ xy	00	01	11	10
0	1			1
1	1			1

z \ xy	00	01	11	10
0		1		
1	1		1	1

S = x + Z

Exemple 6

x	y	z	t	S
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

zt \ xy	00	01	11	10
00	1			1
01		1	1	1
11				1
10	1			1

zt \ xy	00	01	11	10
00	1			1
01		1	1	1
11				1
10	1			1

Exercice 1

CD \ BA	00	01	11	10
00	1	1		
01	1	1		1
11	1	1		1
10	1	1		

CD \ BA	00	01	11	10
00	1	1		1
01	1	1		
11	1	1		
10	1	1		1

Exercice 2

CD \ BA	00	01	11	10
00			1	
01	1	1	1	1
11		1	1	1
10				

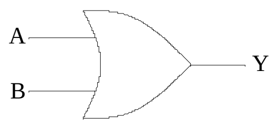
CD \ BA	00	01	11	10
00			1	
01	1	1	1	
11		1	1	1
10		1		

Circuits logiques

Circuit logique = Ensemble de portes logiques reliées entre elles correspondant à une expression algébrique.

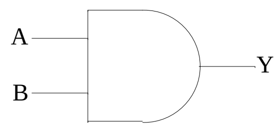
Porte logique (correspond à un opérateur logique)

Porte Or



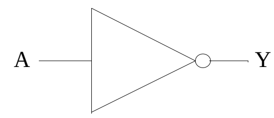
$$Y = A + B$$

Porte And



$$Y = A . B$$

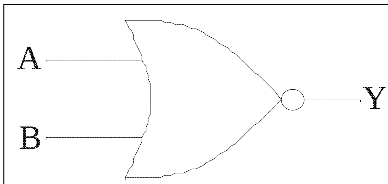
Porte Not



$$Y = \overline{A}$$

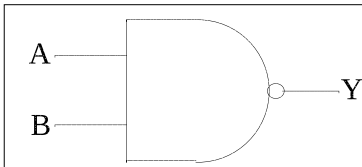
Portes dérivées

Porte Nor



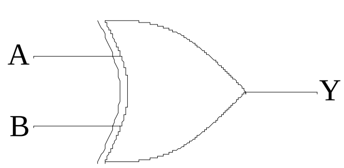
$$Y = \overline{A + B}$$

Porte Nand



$$Y = \overline{A. B}$$

Porte Xor



$$Y = A \oplus B$$

Conception d'un circuit logique

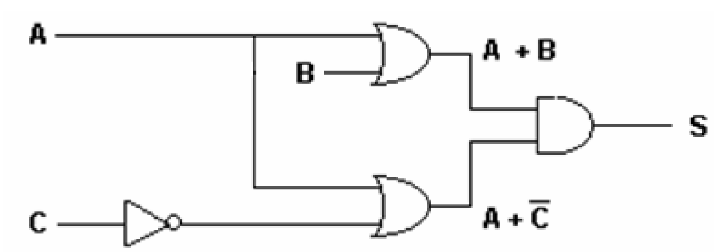
1. **Identifier les entrées et les sorties de la fonction.**
2. **Construire la table de vérité.**
3. **Identifier la fonction à partir de la table de vérité.**
4. **Simplifier la fonction.**
5. **Dessiner le schéma du circuit.**

Réalisation de circuits logiques

Exemple:

Circuit logique correspondant à l'expression algébrique:

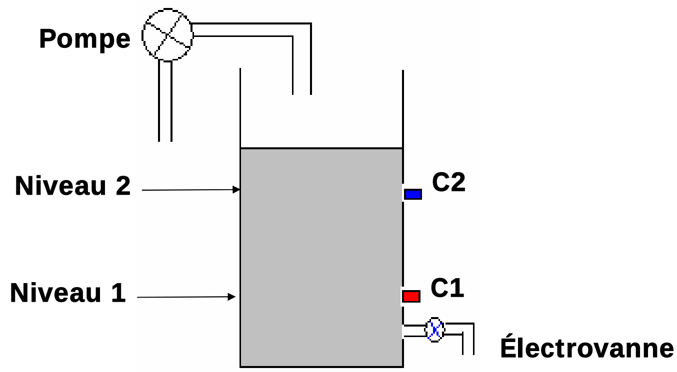
$$(A+B).(A+\overline{C})$$



Exercice 1

Donner le circuit (Exercice 3, simplification algébrique).

Exercice 2



Lorsque le niveau d'eau est inférieure au niveau 1 (Capteur C1), on déclenche la pompe pour remplir le réservoir.
Lorsque Niveau d'eau > Niveau 2, on commande l'électrovanne pour vider le réservoir.

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