Architecture des ordinateurs

Chapitre 2 : A. Circuits logiques

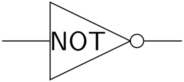
L2 Génie Industriel

Circuit Logique

- Circuit dans lequel seules 2 valeurs logiques sont possibles : 0 ou 1
- Circuit électrique (transistors) :
  - Faible tension = 0
  - Tension élevée = 1
- Composants de base : les portes logiques

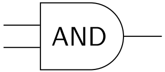
Porte logique

- Permet de combiner les signaux binaires
- Reçoit en entrée une ou plusieurs valeurs binaires (souvent 2)
- Renvoie une unique valeur binaire en sortie
- Exemple : porte NON



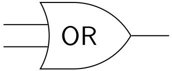
Si la valeur d'entrée est 1, alors la sortie vaut 0.
Si la valeur d'entrée est 0, alors la sortie vaut 1.

Portes ET (AND) et OU (OR)



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

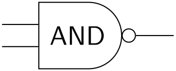
$$S = f(a,b) = a \cdot b = ab$$



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

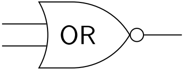
$$S = f(a,b) = a + b$$

Portes NON-ET (NAND) et NON-OU (NOR)



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

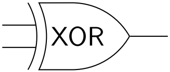
$$S = f(a, b) = \overline{a \cdot b} = \overline{ab}$$



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

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

Porte OU-Exclusif (XOR)



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

$$
\begin{aligned}
S = f(a,b) &= a \oplus b \\
&= (a+b)(\overline{ab}) \\
&= (a+b)(\overline{a}+\overline{b}) \\
&= a\overline{a} + a\overline{b} + b\overline{a} + b\overline{b} \\
&= a\overline{b} + b\overline{a} \\
&= \overline{\overline{a\overline{b}}} + \overline{\overline{b\overline{a}}} \\
&= \overline{\overline{a\overline{b}} + \overline{b\overline{a}}}
\end{aligned}
$$

Ensemble des fonctions booléennes de 2 variables

$f(a,b)$	**00**	**01**	**10**	**11**
0	0	0	0	0
ab	0	0	0	1
$a\overline{b}$	0	0	1	0
a	0	0	1	1
$\overline{a}b$	0	1	0	0
b	0	1	0	1
$a \oplus b$	0	1	1	0
$a + b$	0	1	1	1

$f(a,b)$	**00**	**01**	**10**	**11**
$\overline{a+b}$	1	0	0	0
$\overline{a \oplus b}$	1	0	0	1
$\overline{b}$	1	0	1	0
$a + \overline{b}$	1	0	1	1
$\overline{a}$	1	1	0	0
$\overline{a} + b$	1	1	0	1
$\overline{ab}$	1	1	1	0
1	1	1	1	1

Règles de calcul

Constantes	$a + 0 = a$	$a \cdot 0 = 0$
	$a + 1 = 1$	$a \cdot 1 = a$
Idempotence	$a + a = a$	$a \cdot a = a$
Complémentation	$a + \overline{a} = 1$	$a \cdot \overline{a} = 0$
Commutativité	$a + b = b + a$	$a \cdot b = b \cdot a$
Distributivité	$a + (bc) = (a + b)(a + c)$	
	$a(b + c) = (ab) + (ac)$	
Associativité	$a + (b + c) = (a + b) + c = a + b + c$	
	$a(bc) = (ab)c = abc$	
Lois de Morgan	$\overline{ab} = \overline{a} + \overline{b}$	$\overline{a + b} = \overline{a}\overline{b}$
Involution	$\overline{\overline{a}} = a$	$(a + b)(a + \overline{b}) = a$

Complétude des portes NON-ET et NON-OU

On peut réaliser n'importe quelle fonction booléenne avec seulement des portes :

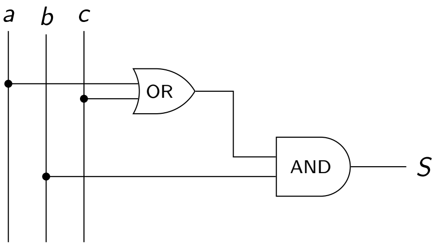
- NON-ET
- NON-OU

Complétude de la porte NON-ET :

- $\overline{a} = \overline{a \cdot a}$
- $a \cdot b = \overline{\overline{a \cdot b}}$
  $= \overline{\overline{ab} \cdot \overline{ab}}$
- $a + b = \overline{\overline{a + b}}$
  $= \overline{\overline{a} \cdot \overline{b}}$

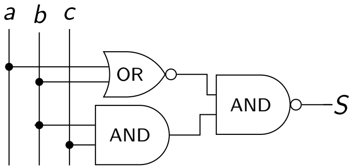
Complétude de la porte NON-OU ?

Du circuit logique à la table de vérité

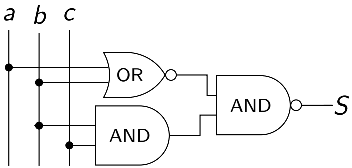


a	b	c	$a+c$	$S = b(a+c)$
0	0	0	0	0
0	0	1	1	0
0	1	0	0	0
0	1	1	1	1
1	0	0	1	0
1	0	1	1	0
1	1	0	1	1
1	1	1	1	1

Du circuit logique à la table de vérité - Exercice



Du circuit logique à la table de vérité - Exercice



a	b	c	$a+b$	$\overline{a+b}$	bc	$S = \overline{\overline{(a+b)} \cdot bc}$
0	0	0	0	1	0	1
0	0	1	0	1	0	1
0	1	0	1	0	0	1
0	1	1	1	0	1	1
1	0	0	1	0	0	1
1	0	1	1	0	0	1
1	1	0	1	0	0	1
1	1	1	1	0	1	1

De la table de vérité au circuit logique

1. Écrire l'équation de la fonction à partir de sa table de vérité

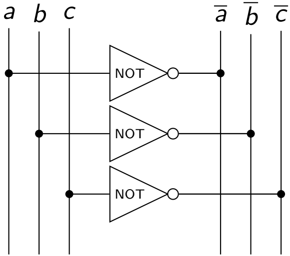
a	b	c	S
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

$$\rightsquigarrow \quad S = f(a, b, c) = \overline{a}b\overline{c} + a\overline{b}c + abc$$

De la table de vérité au circuit logique

2. Réaliser la négation de toutes les variables d'entrée

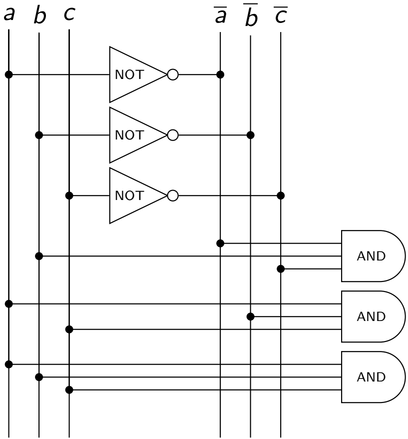
a	b	c	S
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1



De la table de vérité au circuit logique

3. Construire une porte ET pour chacun des termes égal à 1 dans la colonne S
4. Établir le câblage des portes ET avec les entrées appropriées

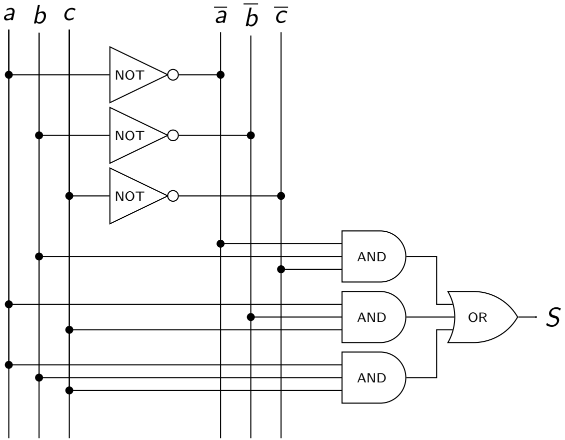
a	b	c	S
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1



De la table de vérité au circuit logique

5. Réunir l'ensemble des sorties des portes ET vers une porte OU, dont la sortie et le résultat de la fonction

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	1
1	1	0	0
1	1	1	1

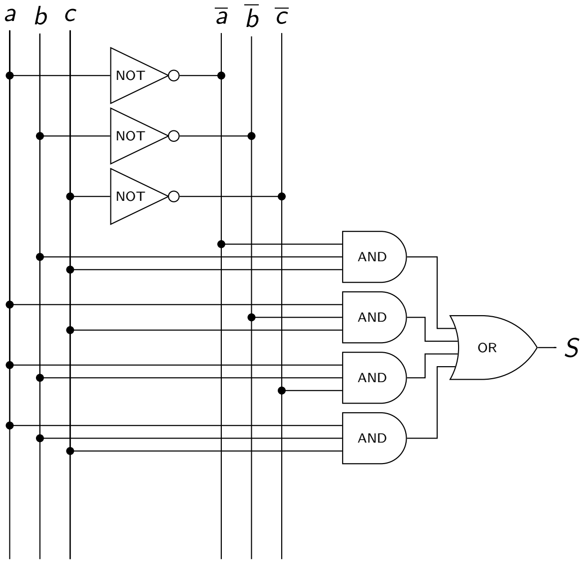


De la table de vérité au circuit logique - Exercice

a	b	c	S
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

De la table de vérité au circuit logique - Exercice

a	b	c	S
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



Simplification

⇒ Diminuer le nombre d'opérateurs

⇒ Diminuer le nombre de portes logiques (et donc le coût)

Deux approches :

- Méthode algébrique (algèbre de Boole)

  Exemple : fonction majoritaire

$$
\begin{aligned}
f(a,b,c) &= \overline{a}bc + a\overline{b}c + ab\overline{c} + abc \\
&= (\overline{a}b + a\overline{b})c + ab(c + \overline{c}) \\
&= (a + b)(\overline{a} + \overline{b})c + ab \\
&= (ac + bc)\overline{ab} + ab \\
&= (ab + ac + bc)(\overline{ab} + ab) \\
&= ab + ac + bc
\end{aligned}
$$

- Méthode des tableaux de Karnaugh

Méthode de Karnaugh (1/3)

- Permet de visualiser une fonction et d'en tirer naturellement une écriture simplifiée.
- Représentation de toutes les combinaisons d'états possibles pour un nombre de variables donné.
- Outil graphique qui permet de simplifier de manière méthodique des expressions booléennes.
- Exploite le codage de l'information et la notion d'adjacence.

Méthode de Karnaugh (2/3)

Principe :

- Mettre en évidence sur un graphique les mintermes ou maxtermes adjacents.
- Transformer les adjacences logiques en adjacences géométriques.

Trois phases :

- Transcription de la fonction dans un tableau codé
- Recherche des adjacences pour simplification
- Mise en équations des groupements effectués

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

1 ligne	1 case
n variables	2^n cases

Méthode de Karnaugh (3/3)

1. Écrire la table de vérité sous la forme d'un code de Gray (ou binaire réfléchi) : les valeurs des entrées ne diffèrent que d'un seul bit entre chaque ligne

a	b	c	S
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

$\Rightarrow$

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

Méthode de Karnaugh (3/3)

2. Compacter la table

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

$\Rightarrow$

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

Méthode de Karnaugh (3/3)

3. Entourer tous les 1 dans des rectangles :
   - Les plus grands possibles
   - Tels que leur taille est une puissance de 2
   - Éventuellement les bords

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

Méthode de Karnaugh (3/3)

4. En déduire la formule et le circuit
   - Une somme (OR) des formules de chaque rectangle
   - La formule d'un rectangle est un produit (AND) :
     - Des variables qui valent toujours 1 dans ce rectangle
     - Des négations de celles qui valent toujours 0
     - Les autres variables n'apparaissent pas dans le produit

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

$$f(a,b,c) = bc + ac + ab$$

Méthode de Karnaugh - Exercice

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

Méthode de Karnaugh - Exercice

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

$f(a, b, c, d) = \overline{a}\overline{b} + cd + a\overline{bd} + \overline{a}b\overline{c}d$ $\rightsquigarrow$ formule pas simplifiée au maximum !

Méthode de Karnaugh - Exercice

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

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