Chapitre 2: **Circuit Combinatoire**
Algèbre de BOOLE

Objectifs :

À la fin de cette unité, vous comprendrez le fonctionnement des principaux éléments d'un ordinateur : décaleur, additionneur, unité logique et arithmétique. Pour y arriver, vous devrez avoir atteint les objectifs suivants :

- \- décrire le fonctionnement et les propriétés des portes logiques, de circuits combinatoires simples tels que le décodeur, le multiplexeur et le démultiplexeur;
- \- utiliser les théorèmes et les identités de l'algèbre de Boole pour synthétiser un circuit à partir de sa table de vérité et simplifier le résultat obtenu.

5.1 Notion de circuit logique

- **Fonctions logiques**
- Une fonction logique est une fonction qui agit sur une ou plusieurs variables logiques.
- Une variable logique est une variable qui peut prendre l'une de deux valeurs : vrai ou faux, 1 ou 0.

5.1 Notion de circuits logiques

- Les circuits logiques sont des circuits électroniques servant à effectuer physiquement des fonctions logiques.

- Circuits combinatoires

- Les signaux de sortie ne dépendent que des signaux d'entrée présents.

- Circuits séquentiels

- Circuits dans lesquels les signaux de sortie dépendent des signaux d'entrée appliqués antérieurement en plus des signaux d'entrée présents.

- # 5.2 Circuits combinatoires
  - ## 5.2.1 Algèbre booléenne
  - Georges Boole, en 1847, a défini une algèbre qui s'applique à des fonctions logiques de variables logiques. Nous verrons que toute fonction logique peut être réalisée à l'aide d'un petit nombre de fonctions logiques de base aussi appelées opérateurs logiques ou portes (gates).
  - La fonction logique d'un circuit combinatoire peut se définir par le tableau de correspondance entre les états d'entrée et les états de sortie. Un tel tableau est appelé table de vérité.

5.2 Circuits logiques

5.2.1 Algèbre booléenne

- La table de vérité d'une fonction de *n* variables a autant de lignes que d'états d'entrée possibles, soit 2^n. Pour chacun de ces états, la sortie peut prendre la valeur 0 ou 1.
- Donc, pour *n* variables, on a 2n fonctions possibles.

5.2 Circuits logiques

5.2.2 Fonctions d'une variable

Soit *a* une variable logique. On a quatre fonctions possibles :

a	Z_0	Z_1	Z_2	Z_3
0	0	0	1	1
1	0	1	0	1

- $Z_0 = 0$: constante
- $Z_1 = a$: identité
- $Z_2 = \overline{a}$: complément
- $Z_3 = 1$: constante

La seule fonction non triviale est le complément, qu'on réalise au moyen de l'opérateur NON ou inverseur $Z = \overline{a}$.

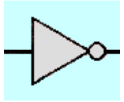
5.2 Circuits logiques

5.2.2 Fonctions d'une variable

- L'opérateur NON ou inverseur

- Table de vérité :

a	$\bar{a}$
0	1
1	0

a  $\bar{a}$

5.2 Circuits logiques

5.2.3 Fonctions de deux variables

- Il y a 16 fonctions possibles de deux variables

00	01	10	11	ab	
0	0	0	0	$F_0 = 0$	Constante 0
0	0	0	1	$F_1 = a.b$	Fonction ET
0	0	1	0	$F_2 = a.\overline{b}$	
0	0	1	1	$F_3 = a$	
0	1	0	0	$F_4 = \overline{a}.b$	
0	1	0	1	$F_5 = b$	
0	1	1	0	$F_6 = a \oplus b$	Fonction XOR
0	1	1	1	$F_7 = a+b$	Fonction OU

5.2 Circuits logiques

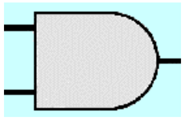
5.2.3 Fonctions de deux variables

00	01	10	11	ab	
1	0	0	0	$F_8 = \overline{a+b} = \bar{a}.\bar{b}$	Fonction NOR
1	0	0	1	$F_9 = \overline{a \oplus b}$	Fonction égalité
1	0	1	0	$F_{10} = \bar{b}$	
1	0	1	1	$F_{11} = a+\bar{b}$	
1	1	0	0	$F_{12} = \bar{a}$	
1	1	0	1	$F_{13} = \bar{a}+b$	
1	1	1	0	$F_{14} = \overline{a.b} = \bar{a} + \bar{b}$	Fonction NAND
1	1	1	1	$F_{15} = 1$	Constante 1

5.2 Circuits logiques

5.2.3Fonctions de deux variables

- Fonction ET (AND)



- Table de vérité

a	b	a.b
0	0	0
0	1	0
1	0	0
1	1	1

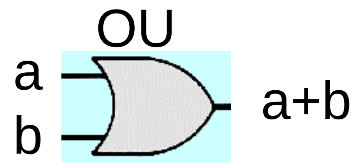
a \ b	0	1
0	0	0
1	0	1

a.b

5.2 Circuits logiques

5.2.3Fonctions de deux variables

- Fonction OU (OR)



- Table de vérité

a	b	a+b
0	0	0
0	1	1
1	0	1
1	1	1

a \ b	0	1
0	0	1
1	1	1

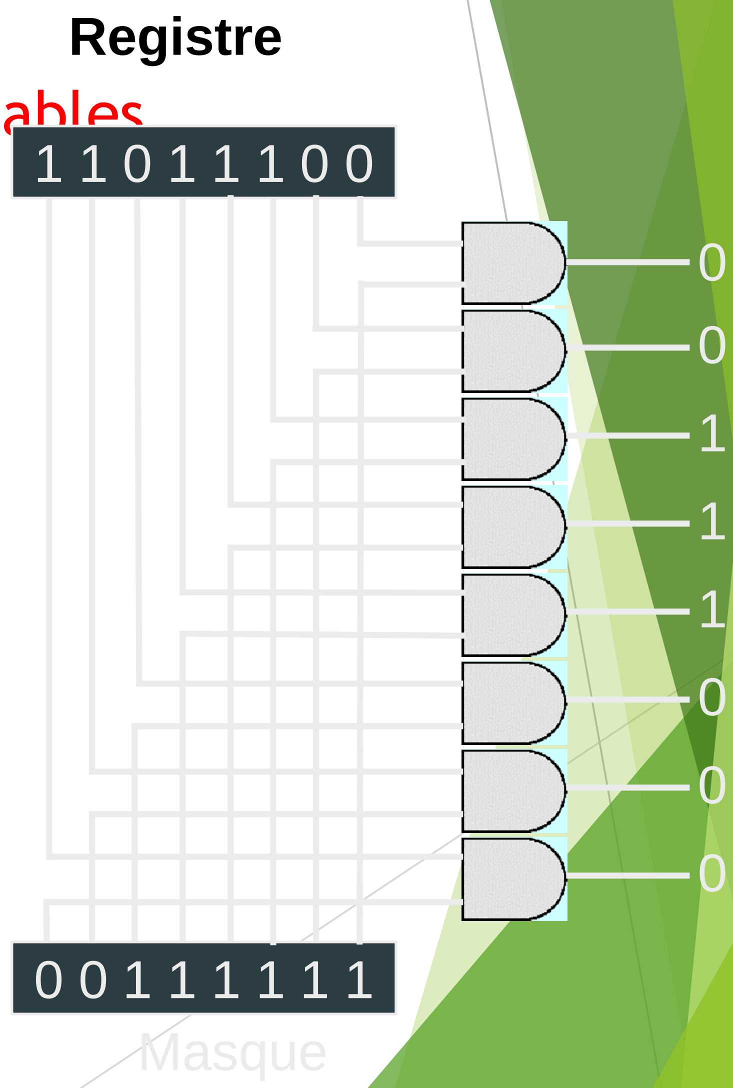
a+b

5.2 Circuits logiques

5.2.3 Fonctions de deux variables

- Application.
- Masquage d'un registre :
- Avec des portes ET, on peut mettre des bits à 0 de façon sélective.

- Avec des portes OU, on pourrait mettre des bits à 1 de façon sélective.

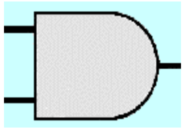


5.2 Circuits logiques

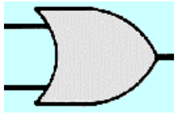
5.2.3 Fonctions de deux variables

- On peut généraliser les fonctions logiques à trois variables ou davantage :

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



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



5.2 Circuits logiques

5.2.3 Fonctions de deux variables

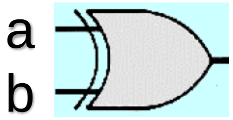
Théorèmes fondamentaux de l’algèbre de Boole

Identités	$a + 0 = a$	$a \,.\, 0 = 0$
	$a + 1 = 1$	$a \,.\, 1 = a$
Commutativité	$a + b = b + a$	$a \,.\, b = b \,.\, a$
Distributivité	$a+(b.c) = (a+b).(a+c)$	$a.(b+c) = a.b + a.c$
Associativité	$a+(b+c) = (a+b)+c = a+b+c$	$a.(b.c) = (a.b).c = a.b.c$
Idempotence	$a + a = a$	$a \,.\, a = a$
Complémentation	$a + \overline{a} = 1$	$a \,.\, \overline{a} = 0$
De Morgan	$\overline{a.b} = \overline{a} + \overline{b}$	$\overline{a + b} = \overline{a} \,.\, \overline{b}$
Autres	$a = a$	
Absorption	$a + (a \,.\, b) = a$	$a.(a + b) = a$
	$a + (\overline{a} \,.\, b) = a + b$	$(a + b).(a + b) = a + b$

5.2 Circuits logiques

5.2.3Fonctions de deux variables

- La fonction XOR (OU-exclusif ou OU-disjonctif) ou fonction inégalité



- Table de vérité

a	b	a ⊕ b
0	0	0
0	1	1
1	0	1
1	1	0

a \ b	0	1
0	0	1
1	1	0

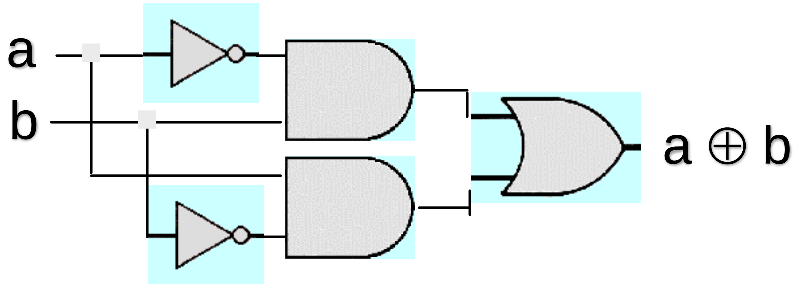
a⊕b

5.2 Circuits logiques

5.2.3 Fonctions de deux variables

La fonction XOR. Propriétés :

- $a \oplus b = a.\overline{b} + \overline{a}.b$ $\qquad$ $\overline{a \oplus b} = a.b + \overline{a}.\overline{b}$
- $a \oplus 0 = a$ $\qquad$ $a \oplus \overline{a} = 1$
- $a \oplus 1 = \overline{a}$ $\qquad$ $a \oplus a = 0$
- $a \oplus b = b \oplus a$ $\qquad$ $(a \oplus b) \oplus c = a \oplus (b \oplus c)$



- Réalisation :

5.2 Circuits logiques

5.2.3 Fonctions de deux variables

Minterm

- Un minterm est le produit logique de toutes les variables d'entrée apparaissant chacune sous la forme vraie (si la variable vaut 1) ou sous la forme complémentée (si la variable vaut 0).

- Ainsi, dans la table de vérité suivante, il y a quatre minterms :

a	b	$a \oplus b$	
0	0	0	$\bar{a} \cdot \bar{b}$
0	1	1	$\bar{a} \cdot b$
1	0	1	$a \cdot \bar{b}$
1	1	0	$a \cdot b$

5.2 Circuits logiques

5.2.3 Fonctions de deux variables

Théorème

- Un circuit logique peut être représenté par la somme logique de tous les minterms pour lesquels la sortie est 1 ou par le produit logique de tous les maxterms pour lesquels la sortie est 0.

- Exemple :
  - Le XOR peut être exprimé par
  - $a \oplus b = a.\bar{b} + \bar{a}.b$
  - ou
  - $a \oplus b = (a + b).(\bar{a} + \bar{b})$

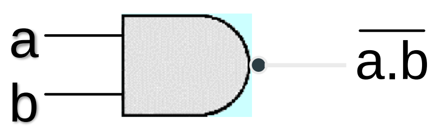
5.2 Circuits logiques

5.2.3 Fonctions de deux variables

Les fonctions NAND et NOR

Le théorème précédent montre que tout circuit logique peut être réalisé avec trois types de portes : ET, OU et NON. On peut aussi les réaliser avec un seul type de porte si on utilise les portes complètes NAND ou NOR.

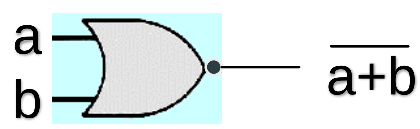
NAND



a \ b	0	1
0	1	1
1	1	0

$\overline{a.b}$

NOR



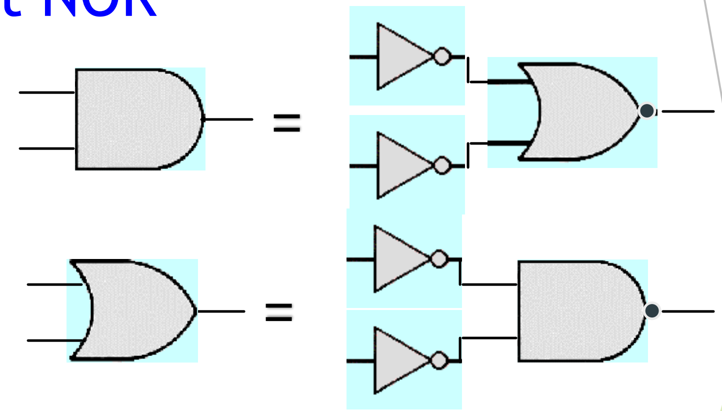
a \ b	0	1
0	1	0
1	0	0

$\overline{a+b}$

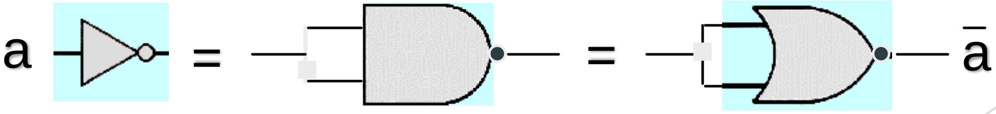
5.2 Circuits logiques

5.2.3 Fonctions de deux variables

Les fonctions NAND et NOR



- En effet :
  - $\overline{\overline{a.b}} = a.b = \overline{\bar{a}+\bar{b}}$
  - et
  - $\overline{\overline{a+b}} = a+b = \overline{\bar{a}.\bar{b}}$
- Aussi, puisque a.a = a et a+a = a



- ## 5.2 Circuits logiques
  - ### 5.2.4 Synthèse d'un circuit combinatoire

  - Pour effectuer la synthèse d'un circuit combinatoire, on part de sa table de vérité.
  - On en extrait les minterms des valeurs pour lesquelles la fonction est vraie (1) et on réalise cette fonction en faisant la somme logique de ces minterms,
  - ou encore, on en extrait les maxterms des valeurs pour lesquelles la fonction est fausse (0) et on réalise cette fonction en faisant le produit logique de ces maxterms.

  - Cette réalisation n'est pas toujours optimale. On aura donc la plupart du temps à simplifier les expressions au moyen de l'algèbre booléenne.

5.2 Circuits logiques

5.2.4 Synthèse d'un circuit combinatoire

- Exemple : soit la table de vérité suivante :

a	b	c	f	minterms
0	0	0	0	
0	0	1	1	$\bar{a}.\bar{b}.c$
0	1	0	0	
0	1	1	0	
1	0	0	0	
1	0	1	1	$a.\bar{b}.c$
1	1	0	1	$a.b.\bar{c}$
1	1	1	1	$a.b.c$

- $f = \bar{a}.\bar{b}.c + a.\bar{b}.c + a.b.\bar{c} + a.b.c$

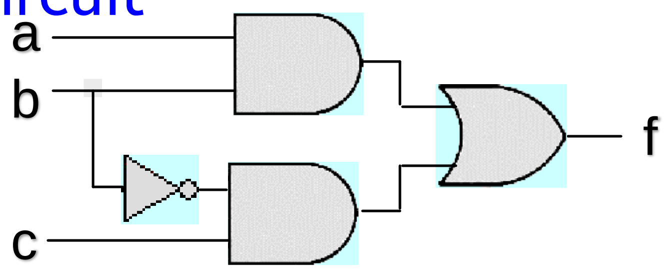
5.2 Circuits logiques

5.2.4 Synthèse d'un circuit combinatoire

Simplification

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

Circuit



- 5.2 Circuits logiques
  - 5.2.4 Synthèse d’un circuit combinatoire
  - Simplification

  - La simplification des équations logiques au moyen de l’algèbre booléenne n’est pas toujours simple, et on ne sait pas toujours si on a atteint une solution optimale.
  - Les tables de Karnaugh permettent de systématiser ce processus.

5.2 Circuits logiques

5.2.4 Synthèse d'un circuit combinatoire

Tables de Karnaugh

ab	c	f
00	0	0
00	1	1
01	0	0
01	1	1
10	0	0
10	1	0
11	0	1
11	1	1

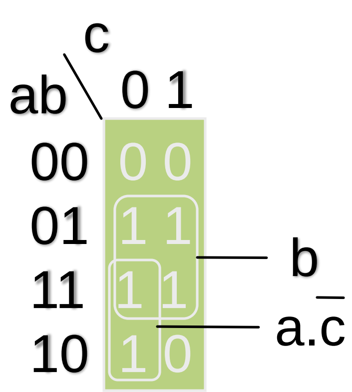
ab \ c	0	1	
00	0	1	$\bar{a}.c$
01	0	1	
11	1	1	$a.b$
10	0	0	

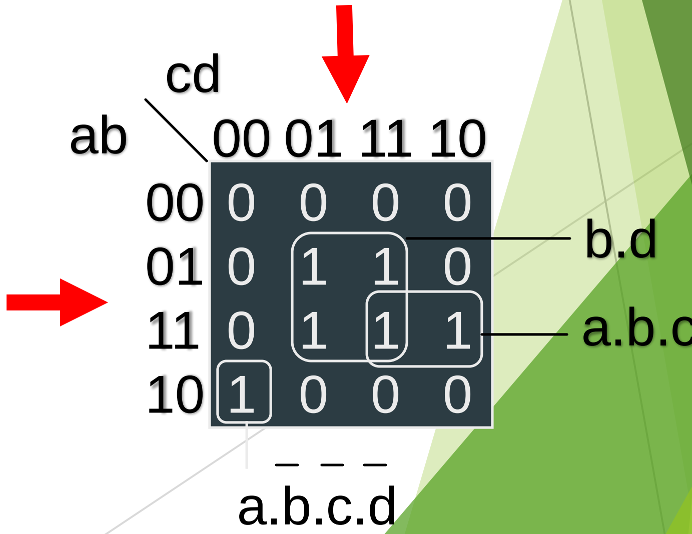
Donc $f = a.b + \bar{a}.c$

▶5.2 Circuits logiques

▶5.2.4 Synthèse d'un circuit combinatoire

- Chaque boucle doit être rectangulaire et doit contenir le maximum possible de 1 qui soit une puissance de 2 : 1, 2, 4, 8, 16, etc. et ne contenir aucun 0.
- La boucle est caractérisée par les combinaisons qui sont vraies pour tous les éléments de la boucle.
- Les recouvrements sont possibles.





Logique combinatoire

5.2 Circuits logiques

5.2.4 Synthèse d'un circuit combinatoire

- Les boucles peuvent «faire le tour» de la table

ab c	f
0 0 0	0
0 0 1	1
0 1 0	0
0 1 1	0
1 0 0	0
1 0 1	1
1 1 0	0
1 1 1	0

Donc $f = \overline{b}.c$

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

$\overline{b}.c$

Logique combinatoire

5.2 Circuits logiques

5.2.4 Synthèse d'un circuit combinatoire

- Les boucles peuvent «faire le tour» de la table

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

$b.\overline{d}$

$\overline{b}.d$

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

$\overline{b}.\overline{d}$

5.2 Circuits logiques

5.2.4 Synthèse d’un circuit combinatoire

États indifférents

Dans certains cas, la sortie pour un état d’entrée donné est indifférente, soit parce que cet état d’entrée ne peut jamais se produire, soit parce que la sortie correspondante ne nous intéresse pas. On inscrit alors un x dans la table de Karnaugh. On peut s’en servir pour minimiser le circuit comme si c’étaient des 1.

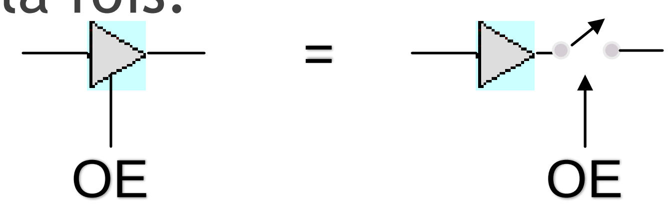
ab \ cd	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	x	x	x
10	x	0	1	x

a.b + a.c au lieu de
$a.b.\overline{c}.\overline{d} + a.\overline{b}.c.d$

▶5.2 Circuits logiques

▶ Logique à trois états

▶Il faut souvent appliquer à un même fil la sortie de l'une ou l'autre d'un ensemble de sorties. Pour éviter l'interférence entre les différents circuits, par exemple une sortie qui tenterait d'appliquer 1 à une ligne alors qu'une autre sortie tenterait d'y appliquer 0, on utilise la logique trois états, dans laquelle la sortie peut être 0, 1, ou haute impédance (comme si elle n'était pas connectée). On ajoute une entrée Output Enable (OE) à chaque circuit et on n'en active qu'un à la fois.



5.2 Circuits logiques

Logique programmable

- Les circuits de logique programmable PLA (Programmable Logic Array), PLD (Programmable Logic Devices), EPLD (Eraseble PLD), etc. sont basés sur le fait que toute fonction logique peut être exprimée comme une somme de minterms.

- Le circuit contient un réseau de portes logiques ET à *n* variables, et un réseau de portes logiques OU, suivi, le cas échéant, d'une couche de bistables. Des appareils spécialisés permettent la programmation du réseau.

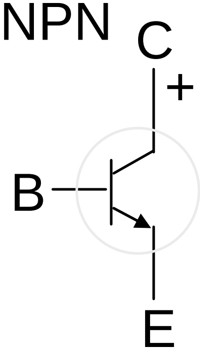
Technologie des semi conducteurs

Transistors

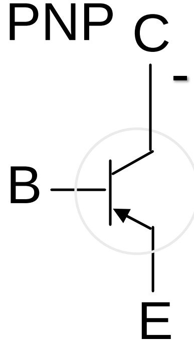
Transistors bipolaires

-

NPN

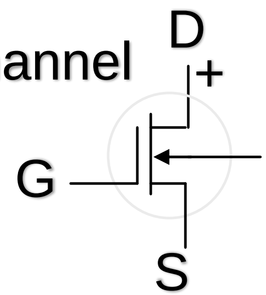


PNP

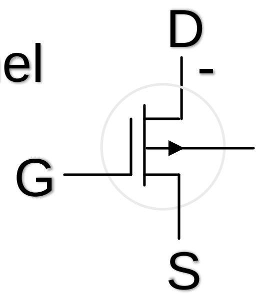


- Transistors unipolaires (à effet de champ)

n-channel



p-channel

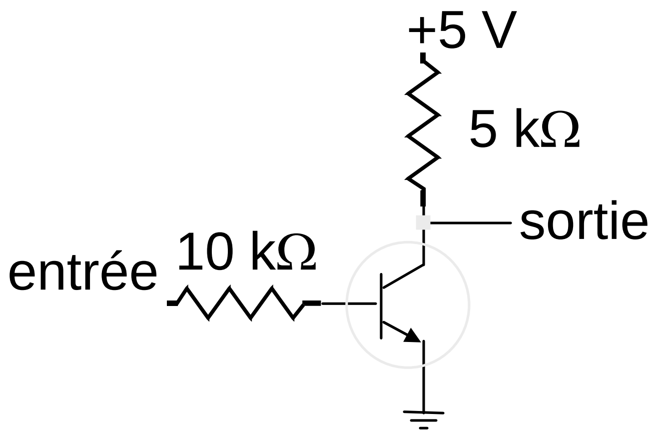


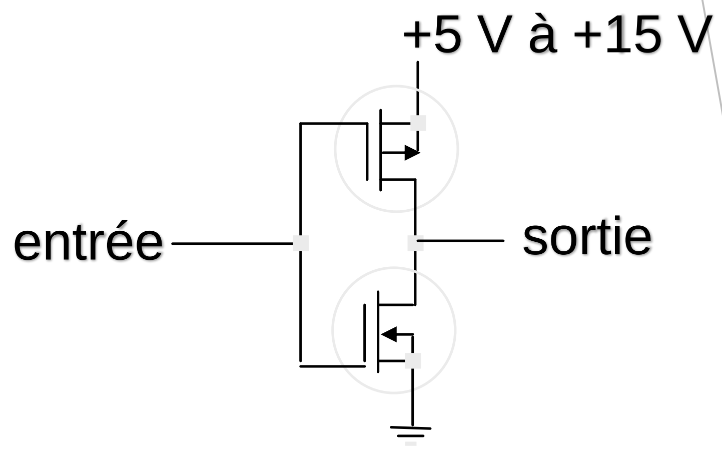
MOSFET ou MOS, CMOS

Logique combinatoire

- **Technologie des semiconducteurs**
  - Inverseur TTL

Inverseur CMOS

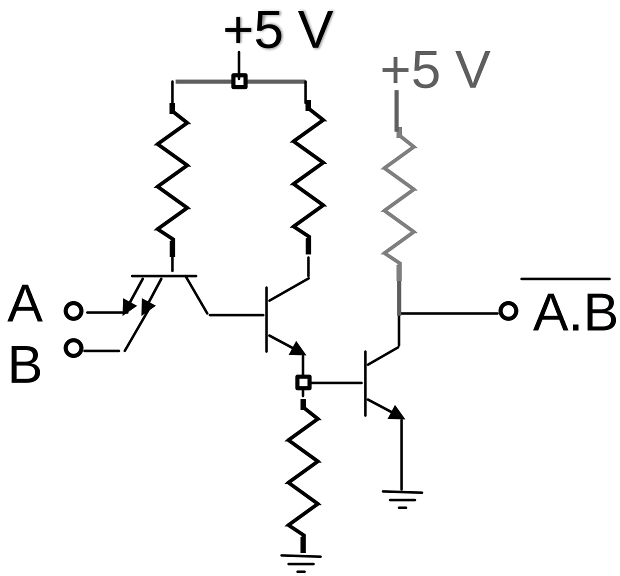


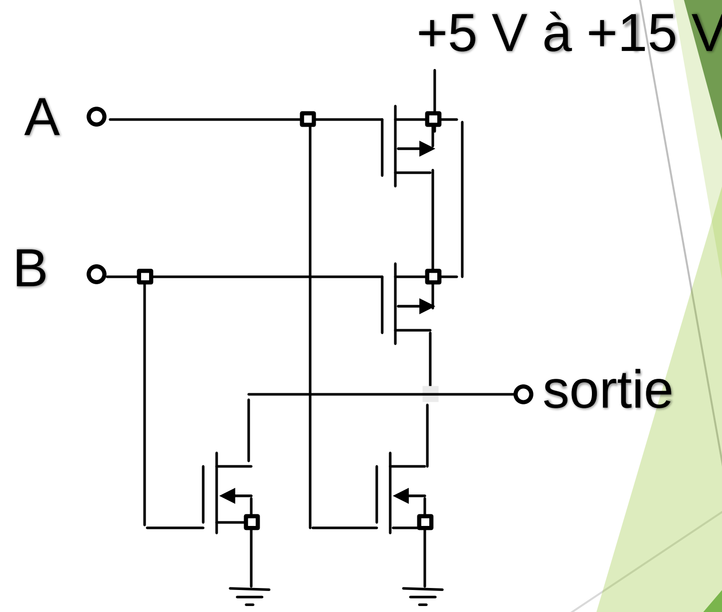


Logique combinatoire

Technologie des semiconducteurs

- NAND TTL NOR CMOS





F I N

Chapitre 3: Circuits combinatoires