
misez sur les
compétences
qui feront la différence.
pôle formation des industries technologiques
CRÉATEUR DE COMPÉTENCES
UI
Union des Industries et Métiers de la Métallurgie
HOPSCOTCH rouge - © Hugo Felix / Shutterstock.com

Circuits logiques 1

Circuits combinatoires

- Algèbre de Boole
- Fonctions d’une variable
- Fonctions de deux variables
- Synthèse d’un circuit combinatoire
- Analyse d’un circuit combinatoire
- Multiplexeurs et démultiplexeurs
- Décodeurs - Codeurs - Transcodeurs

misez sur les compétences qui feront la différence.

Introduction

Les **circuits logiques** exécutent des opérations sur des variables logiques, transportent et traitent des signaux logiques.

On distingue deux types de circuits logiques :

- les circuits **combinatoires** qui sont des circuits idéalisés où le temps de propagation des signaux n’est pas pris en considération. Les signaux de **sortie** ne **dépendent** que des signaux **d’entrée**, appliqués à l’instant considéré ;
- les circuits **séquentiels** qui sont des circuits où il faut **tenir compte du temps de propagation des signaux et de la mémoire du circuit**. Les signaux de sortie dépendent des signaux d’entrée appliqués antérieurement.

Algèbre de Boole

- 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**.
- La table de vérité d’une fonction de **n variables** a autant de **lignes** que d’états d’entrée, soit $\mathbf{2^n}$.

- Toute **fonction logique** peut être **réalisée** à l’aide d’un petit nombre de fonctions logiques de base appelées aussi **opérateurs logiques** ou **portes** [gates].

- Pour chacun de ces états, la **sortie** peut **prendre** la valeur **0** ou **1**. Ainsi, pour n variables on a $(2^2)^n$ fonctions possibles.

Fonctions d’une variable

La table de vérité des fonctions d’une variable a donc deux états d’entrée.

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

$Z_2 = \overline{a}$ complémentation

$Z_3 = 1$ constante

Fonctions logiques d’une variable a

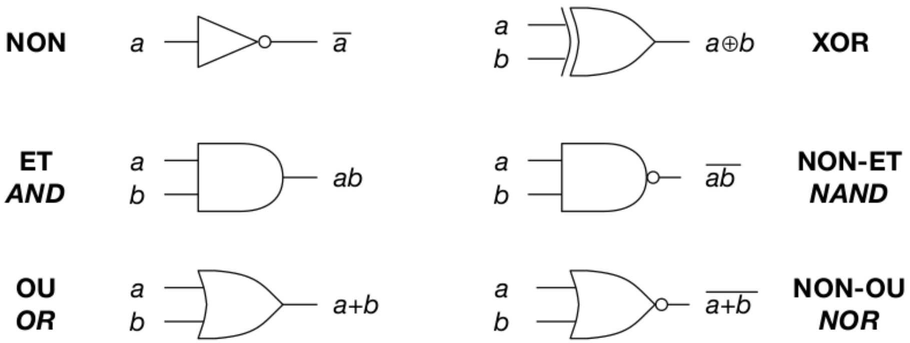
On définit ainsi un ensemble de $2^2 = 4$ fonctions d’une variable.

L’opérateur NON [NOT]

la fonction Z_2, dite de complémentation, est réalisée par l’opérateur NON ou **inverseur**.

Fonctions de deux variables

- La table de vérité des fonctions de deux variables a et b indique qu'il y a **16 fonctions** possibles pour ces deux variables.



Symboles des principaux opérateurs logiques

00	01	10	11	ab	
0	0	0	0	$F_0 = 0$	(constante nulle)
0	0	0	1	$F_1 = ab$	(fonction ET)
0	0	1	0	$F_2 = a\bar{b}$	
0	0	1	1	$F_3 = a$	
0	1	0	0	$F_4 = \bar{a}b$	
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)
1	0	0	0	$F_8 = \overline{a+b} = \bar{a}\,\bar{b}$	(fonction NOR)
1	0	0	1	$F_9 = \overline{a \oplus b}$	
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	$F_{14} = \overline{ab} = \bar{a} + \bar{b}$	(fonction NAND)
1	1	1	1	$F_{15} = 1$	(constante 1)

Fonctions de deux variables

Les opérateurs ET [AND] et OU [OR]

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

Table de vérité des fonctions ET et OU

- La fonction **intersection** ou produit logique $Z = a \times b = ab = a \cap b$ est réalisée par **l'opérateur ET**. Z vaut 1 si et seulement si a et b valent 1 .
- La fonction **réunion** ou somme logique $Z = a + b = a \cup b$ est réalisée par **l'opérateur OU**. Z vaut 1 si a ou b ou les deux valent 1 .
- Les trois fonctions **NON**, **ET**, **OU** sont souvent appelées **opérateurs de base** ; elles **définissent** à elles seules une importante structure algébrique : **l'algèbre de Boole**.

Fonctions de deux variables

Théorèmes fondamentaux de l'algèbre de BOOLE

Théorème des constantes	$a+0=a$	$a\times 0=0$
	$a+1=1$	$a\times 1=a$
Idempotence	$a+a=a$	$a\times a=a$
Complémentation	$a+\overline{a}=1$	$a\times\overline{a}=0$
Commutativité	$a+b=b+a$	$a\times b=b\times a$
Distributivité	$a+(bc)=(a+b)(a+c)$	
	$a(b+c)=(ab)+(ac)$	
Associativité	$a+(b+c)=(a+b)+c=a+b+c$	
	$a(bc)=(ab)c=abc$	
Théorèmes de De Morgan	$\overline{ab}=\overline{a}+\overline{b}$	$\overline{a+b}=\overline{a}\ \overline{b}$
Autres relations	$\overline{\overline{a}}=a$	$a+(ab)=a$
	$a+(\overline{a}b)=a+b$	$a(a+b)=a$
	$(a+b)(a+\overline{b})=a$	

- Voici, en résumé et sans démonstration, **les principales propriétés** de cette structure algébrique.

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

Fonctions de deux variables
L'opérateur XOR

a	b	$a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0

Table de vérité du XOR (Z = a ⊕ b)

- **L'opérateur XOR**, appelé aussi **OU exclusif** [eXclusive OR] réalise une fonction de deux variables où Z **vaut 1** si et seulement si **une seule des deux variables vaut 1** .

- **minterms** : $Z = a \oplus b = \overline{a}b + a\overline{b}$.
- **maxterms** : $Z = a \oplus b = (\overline{a} + \overline{b})(a + b)$.

pôle formation des industries technologiques
CRÉATEUR DE COMPÉTENCES

Fonctions de deux variables

Propriétés du XOR

$$a \oplus b = \bar{a}b + a\bar{b} \qquad \overline{a \oplus b} = ab + \bar{a}\,\bar{b}$$

$$a \oplus 0 = a \qquad a \oplus a = 0$$

$$a \oplus 1 = \bar{a} \qquad a \oplus \bar{a} = 1$$

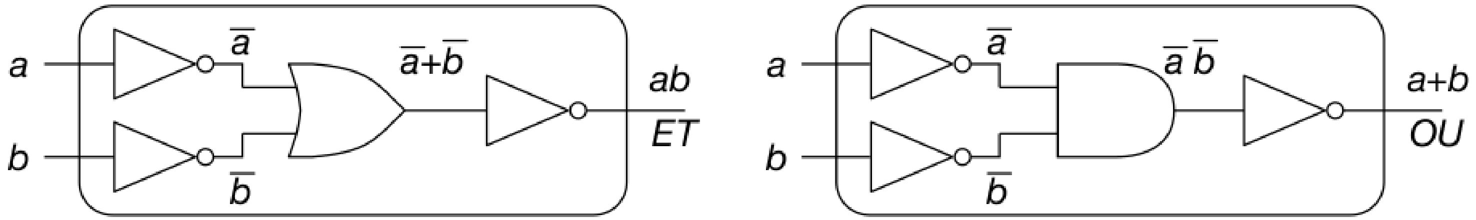
$$a \oplus b = b \oplus a \qquad (a \oplus b) \oplus c = a \oplus (b \oplus c)$$

$$a \oplus b = \overline{\bar{a}\,\bar{b} + ab} = (a+b)(\bar{a}+\bar{b}) = (a+b)\overline{ab}$$

Propriétés du XOR

Fonctions de deux variables
Les opérateurs complets

- Il existe des opérateurs complets tels que différents assemblages d'un même opérateur (complet) **permettent** de **réaliser** les trois fonctions **ET**, **OU** et **NON**.
- Ces **opérateurs complets** sont les opérateurs **NAND et NOR**.



Exemple de réalisation des opérateurs ET et OU

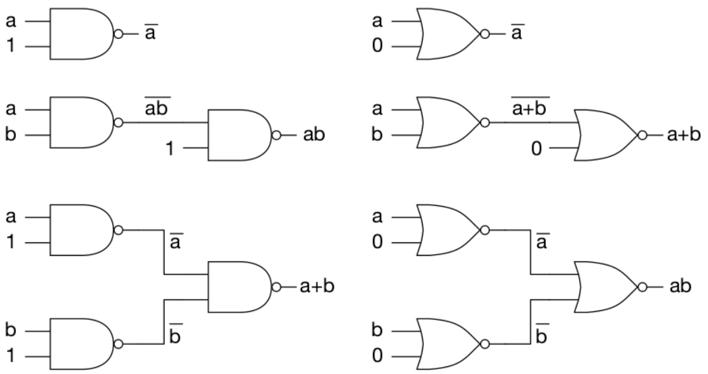
Fonctions de deux variables
Les opérateurs NAND et NOR

- On peut les **exprimer** les opérateurs **NON**, **ET** et **OU** à partir **d'un seul opérateur**, soit **NAND**, soit **NOR**.
- on **utilise** ainsi une **logique NAND** ou une **logique NOR**.

NAND $(Z = \overline{ab} = \bar{a} + \bar{b})$ et **NOR** $(Z = \overline{a+b} = \bar{a}\,\bar{b})$

a	b	a NAND b	a NOR b
0	0	1	1
0	1	1	0
1	0	1	0
1	1	0	0

Table de vérité du NAND et du NOR



Réalisation de ET, OU, NON avec des NAND et des NOR

Synthèse d’un circuit combinatoire

- Le **problème** est le suivant : à partir de la **définition** d’une **fonction** logique, par exemple sa table de vérité, il faut **déterminer** un **logigramme** (représentation graphique d’un circuit logique) qui **réalise** cette **fonction**.

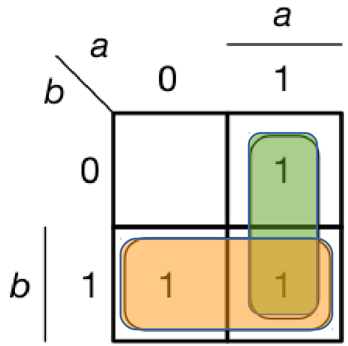
- La **marche à suivre** pour faire la synthèse d’un circuit combinatoire est la suivante :
  - **construire** la **table de vérité** de la fonction logique ; en dériver une expression algébrique (par exemple, somme logique des minterms);
  - **simplifier** cette **expression** en la transformant en une expression équivalente plus simple (par exemple, par passage de la forme canonique à un polynôme contenant un nombre minimal d’opérateurs). Il existe plusieurs méthodes de simplification : tables de **Karnaugh**, théorèmes de l’algèbre de Boole ;
  - **réaliser** la **fonction** logique à l’aide **d’opérateurs** divers (NON, ET, OU, XOR, NAND, NOR, etc). Il existe de nombreuses solutions.

- Les **tables** (ou diagrammes) de **Karnaugh** permettent de **simplifier** des **fonctions** Logiques. Cette méthode est particulièrement utile avec un nombre de **variables inférieur** à **6**.

- Soit une fonction définie par sa **table de vérité**

$$Z(a,b) = \bar{a}b + a\bar{b} + ab$$

a	b	Z
0	0	0
0	1	1
1	0	1
1	1	1



- Selon le théorème d’idempotence on peut

écrire : $Z = \bar{a}b + a\bar{b} + ab + ab$

d’où : $Z = a(b+\bar{b}) + b(a+\bar{a}) = a + b$

- Pour **remplir** la **table de Karnaugh** à partir de la table de vérité, on **attribue** la valeur **1** aux cases correspondantes aux **états** d’entrée où la **fonction** est **vraie**.

- La **méthode** de **simplification** consiste à **encercler** tout ensemble de **cases occupées**, **adjacentes** sur la **même ligne** ou sur la **même colonne**.
Les **recouvrements** sont **permis**. Dans l'exemple Z = a + b

Synthèse d'un circuit combinatoire
Table de Karnaugh avec 3 variables

- Soit la fonction Z suivante, exprimée sous sa forme canonique :

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

b \ ac	00	01	11	10
0	1		1	1
1			1	

Table de Karnaugh à trois variables

- L'expression simplifiée est $Z = ac + \overline{b}\,\overline{c}\,.$

Synthèse d'un circuit combinatoire
Table de Karnaugh avec 4 variables

- Soit la fonction Z suivante, donnée sous sa forme canonique :

$$Z(a,b,c,d) = \bar{a}\bar{b}\bar{c}\bar{d} + \bar{a}\bar{b}\bar{c}d + \bar{a}\bar{b}cd + \bar{a}b\bar{c}d + ab\bar{c}d + a\bar{b}\bar{c}d$$
$$+ a\bar{b}\bar{c}\bar{d} + \bar{a}bcd + abcd + a\bar{b}cd + \bar{a}bc\bar{d} \ .$$

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

Table de Karnaugh à quatre variables

- L'expression simplifiée est $Z = d + \bar{b}\bar{c} + \bar{a}bc$.

Synthèse d'un circuit combinatoire

Table de Karnaugh avec 4 variables

- Dans une table de Karnaugh, les 4 coins sont des cases adjacentes. Par exemple, avec la forme canonique suivante :

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

- D'une façon générale, la **méthode** de **simplification** d'une fonction de quatre variables par Karnaugh est la suivante :
  - **encercler** d'abord les **cases** à **1** qui ne sont **pas adjacentes** à d'autres 1 et ne peuvent donc pas former des blocs de deux cases ;
  - **encercler** celles qui peuvent former des **groupes** de **deux cases** mais pas de quatre cases ;
  - **encercler** celles qui peuvent se combiner en blocs de **quatre cases** mais pas de huit cases ;
  - **enfin**, **encercler** les **groupes** de **huit**.

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

Table de Karnaugh avec les quatre coins occupés

- Les quatre coins peuvent être regroupés. L'expression simplifiée est donc la suivante : $Z=\overline{b}\,\overline{d}$.

Synthèse d'un circuit combinatoire
Synthèse d'un additionneur binaire

- L'**additionneur binaire** est un circuit logique capable de **faire** la **somme** de **deux nombres binaires** selon le principe de la table d'addition suivante :

a	+	b	=	Somme	
0	+	0	=	0	
0	+	1	=	1	
1	+	0	=	1	
1	+	1	=	0	$\Rightarrow$ avec une retenue = 1

Table d'addition

- Le **demi-additionneur ne tient pas compte** de la **retenue** éventuelle provenant d'une opération précédente.

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

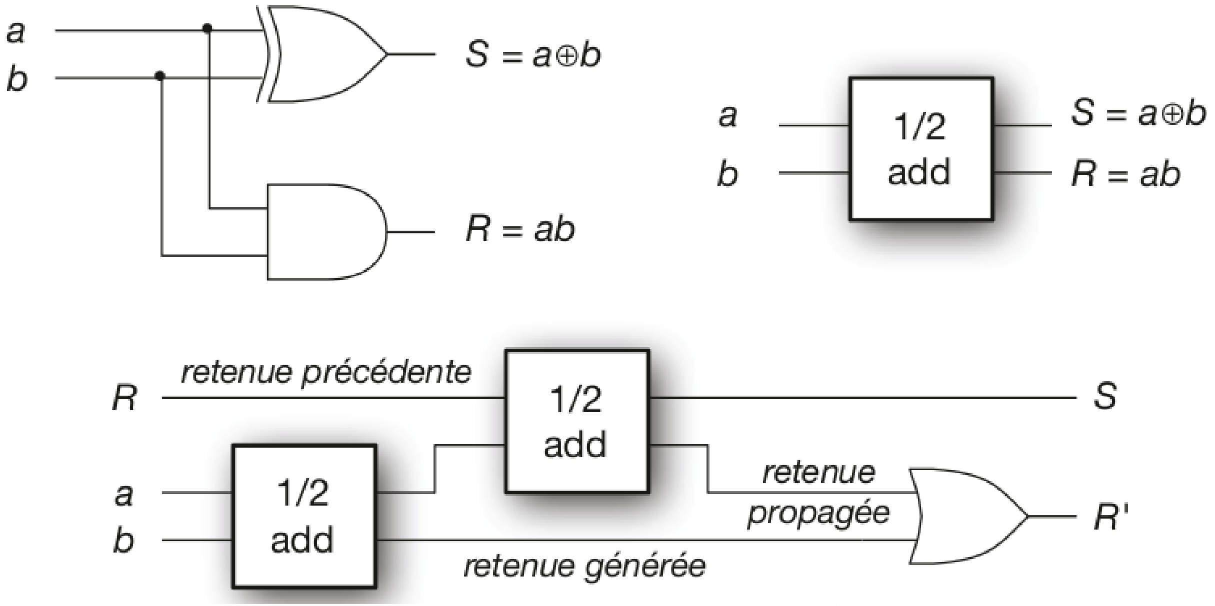
S = Somme
R = Retenue

Table de vérité du demi-additionneur

$$S = \bar{a}b + a\bar{b} = a \oplus b \quad \text{et} \quad R = ab$$

Synthèse d’un circuit combinatoire
Synthèse d’un additionneur binaire

- L’**étage d’additionneur** est **composé** de **deux demi-additionneurs** et d’une **porte OU**.
- L’**additionneur complet** est **obtenu** en utilisant en **parallèle** plusieurs **étages additionneurs** (il faut **autant d’étages** que de **bits** composant les **nombres binaires**).
- Ces **étages** doivent être **connectés** : il suffit de connecter chaque **sortie R’** à l’**entrée R** de l’étage suivant.



Demi-additionneur et étage d’additionneur

Synthèse d'un circuit combinatoire

Analyse d'un circuit combinatoire

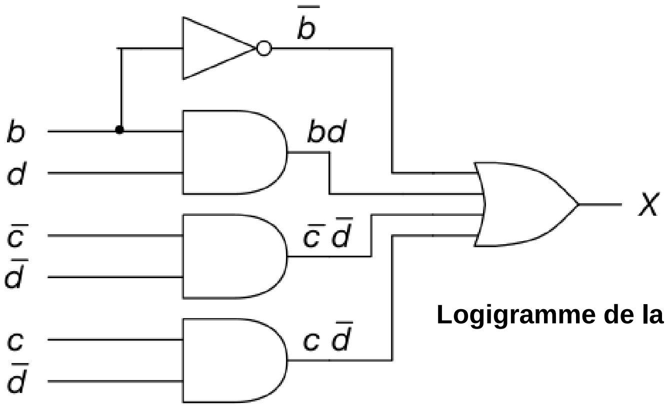
- **L'analyse** consiste à **retrouver** la **fonction** d'un circuit dont on connaît uniquement le **logigramme.** Cette fonction est unique.

- La **marche à suivre** pour faire l'analyse d'un circuit combinatoire est la suivante :
  - en **procédant** des **entrées vers** les **sorties**, donner, **pour** chaque **opérateur** l'expression de sa **sortie** en **fonction** de ses **entrées**, jusqu'à obtention d'une expression pour chaque fonction (sortie) réalisée par le circuit ;
  - **donner** la **table de vérité** correspondante ;
  - en **déduire** le **rôle** du **circuit.**

Analyse d’un circuit combinatoire

Exemple d’analyse d’un circuit

- Étant donné le logigramme présenté dans la figure, déterminer la fonction X .



Logigramme de la fonction X

b	d	$\bar{b}$	bd	$\bar{d}$	X
0	0	1	0	1	1
0	1	1	0	0	1
1	0	0	0	1	1
1	1	0	1	0	1

Table de vérité

- Expression de la fonction : $X = \bar{b} + bd + \bar{c}\bar{d} + c\bar{d}$.
- On peut simplifier : $X = \bar{b} + bd + \bar{d}(c + \bar{c}) = \bar{b} + bd + \bar{d}$.
- On obtient : $X = \bar{b} + d + \bar{d} = \bar{b} + 1 = 1$.

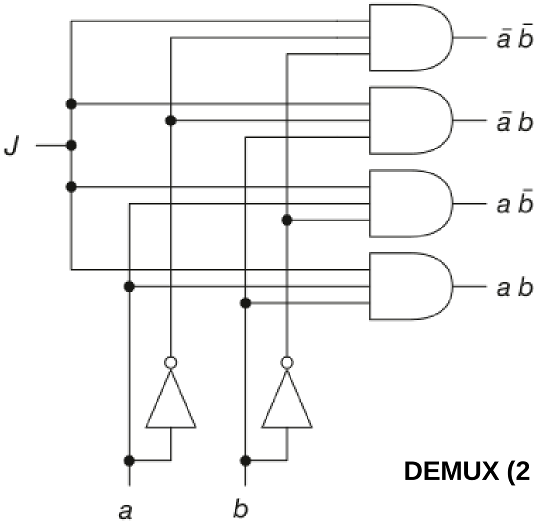
Synthèse d'un circuit combinatoire
Multiplexeurs et démultiplexeurs

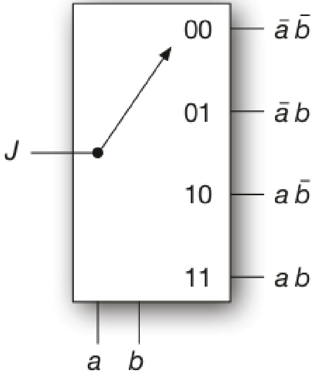
- Mis à part l'additionneur, d'autres circuits combinatoires jouent un **rôle important** dans l'ordinateur, en particulier les **multiplexeurs** et les **démultiplexeurs**.

- Le **multiplexeur** (MUX) est un circuit qui accepte **plusieurs** signaux logiques (données) en **entrée** et **n'autorise** qu'un **seul** d'entre eux en **sortie**,
- le **démultiplexeur** (DEMUX) a **une** seule ligne d'**entrée** et de **nombreuses** lignes en **sortie**. Il **transmet l'entrée** sur une seule ligne en **sortie**.

Démultiplexeur

- On appelle **démultiplexeur** (DEMUX) tout **système combinatoire réalisant** les **2^n minterms** de n variables qui **correspondent** aux **n lignes** de **sélection**.

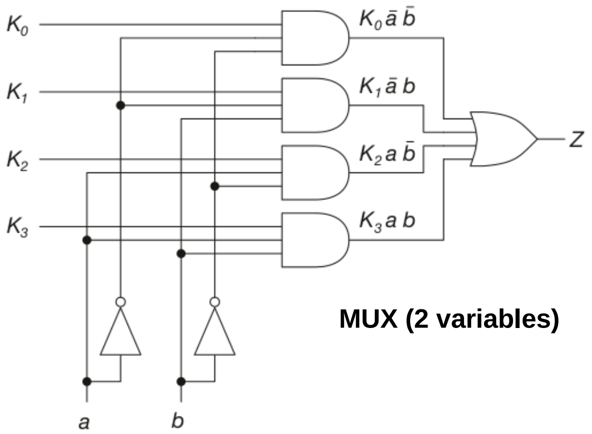




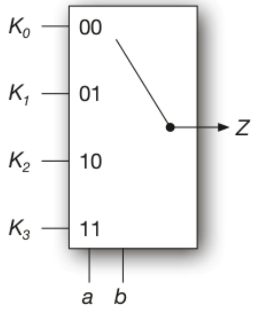
DEMUX (2 variables)

Multiplexeur

- On appelle **multiplexeur** (MUX) tout **système combinatoire réalisant** la **fonction universelle de n variables** qui correspondent aux n lignes de sélection.



MUX (2 variables)



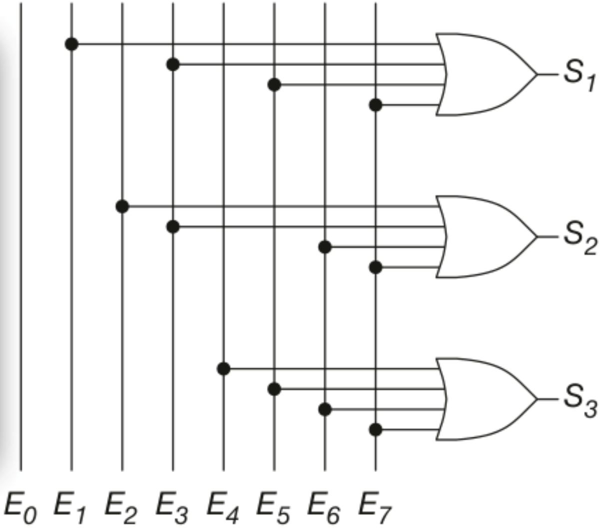
- Dans le cas de **deux variables**, la **fonction universelle** est définie de la manière suivante : $Z(a,b) = K_0\bar{a}\bar{b} + K_1\bar{a}b + K_2a\bar{b} + K_3ab$.

- On appelle **a** et **b** les **lignes** de **commande** et **K_0 , K_1 , K_2 et K_2** les **lignes** de **données**.

- Le MUX est donc un **sélecteur de données**.

Exemple de codeur

	E_0	E_1	E_2	E_3	E_4	E_5	E_6	E_7	S_3	S_2	S_1
0	1								0	0	0
1		1							0	0	1
2			1						0	1	0
3				1					0	1	1
4					1				1	0	0
5						1			1	0	1
6							1		1	1	0
7								1	1	1	1



Codeur à 8 entrées

Synthèse d'un circuit combinatoire
TD / TP