Architecture des Ordinateurs

Mounir T. El Araki

mounir.elarakitantaoui@uic.ac.ma

Plan du cours

- Historique
- Présentation de l'architecture des ordinateurs
- Représentation interne des informations
- Encodage/décodage de l'information
- **Circuits logiques**
- Mémoires
- Unité centrale de traitement

Objectif

- Maîtriser les bases de l'algèbre booléenne.
- Synthétiser et analyser un circuit combinatoire .
- Connaître les circuits logiques les plus importants.
- Comprendre les principes des circuits séquentiels et des bascules.

Circuit logique

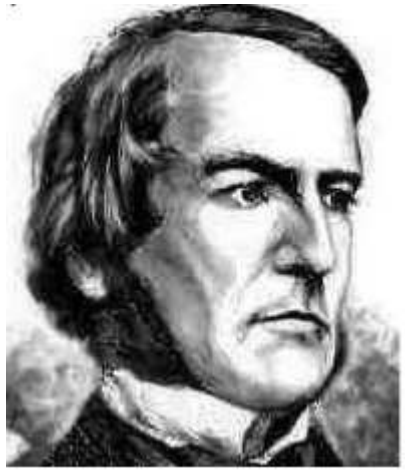
- Les circuits des machines électroniques modernes ont 2 états d'équilibre 0 et 1 (i.e., 2 niveaux de tension)) signal logique
- Une variable binaire est une variable qui ne peut prendre que deux états.

- **Circuit logique**
  - Représentation d'un circuit électronique. Exécute des opérations sur des variables logiques, transporte et traite des signaux logiques.

- **Circuit combinatoire**
  - circuit idéalisé
  - pas de prise en compte du temps de propagation des signaux
  - signaux de sortie ne dépendent que des signaux en entrée

- **Circuit séquentiel**
  - tiens compte du temps de propagation
  - mémoire
  - signaux de sortie dépendent également des signaux en entrée antérieurs

Algèbre de Boole



- Mathématicien britannique (1815-1864). Un des promoteurs de la logique mathématiques contemporaine

- George Boole a défini une algèbre qui s'applique a des fonctions logiques de variables logiques (variables booléennes).
  - Toute fonction logique peut être réalisée à partir de fonctions logiques de base.
  - Les opérations arithmétiques peuvent être réalisées à l'aide d'opérations logiques de base.

- Shannon découvrit que l'algèbre des classes de Boole était un outil puissant, qui permettait d'analyser et de représenter les circuits complexes, basés sur un fonctionnement à deux états.

Algèbre de Boole

- **Fonction logique**
  - Fonction définie par une table de vérité (i.e., tableau de correspondance entre les états d'entrée et les états de sortie)
  - Toutes les combinaisons possibles des variables d'entrées.
  - Représentée sous forme de diagramme ou d'expressions algébrique.
  - Trois operateurs de base : NON, ET, OU

- **Table de vérité**
  - La table de vérité d'une fonction de n variables a autant de ligne que d'états d'entrée, soit 2^n. Comme pour chacun de ces états d'entrées, on peut avoir deux valeurs de sorties (0 et 1), cela nous donne $2^{2pui(n)}$ fonctions possibles a n variables.

  - pour 1 variable, 4 fonctions pour 2 variables, 16 fonctions
  - pour 3 variables, 256 fonctions pour 4 variables, 65536 fonctions

Fonctions d'une variable

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 constante

$Z_2 = \overline{a}$ constante

Z_3 = 1 constante

- Operateur NON
  - La seule fonction logique a une variable non triviale est la fonction de complémentation (Z_2) réalisée par l'opérateur logique NON (ou inverseur)

Table de vérité

Entrée a	NON $\overline{a}$
0	1
1	0

Fonctions à 2 variables

- Il existe 16 fonctions logiques a 2 variables. Les deux non triviales les plus importantes sont les fonctions de produit logique (intersection) et somme logique (réunion) réalisées par les operateurs **ET** et **OU**, notés respectivement **a.b** et **a + b**.

Entrée		ET
a	b	a.b
0	0	0
0	1	0
1	0	0
1	1	1

Entrée		OU
a	b	a+b
0	0	0
0	1	1
1	0	1
1	1	1

Fonctions à 2 variables

Entrée		NOR
a	b	$\overline{a + b}$
0	0	1
0	1	0
1	0	0
1	1	0

Entrée		NAND
a	b	$\overline{a.b}$
0	0	1
0	1	1
1	0	1
1	1	0

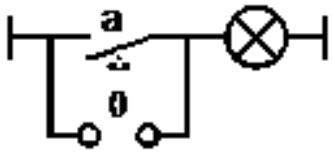
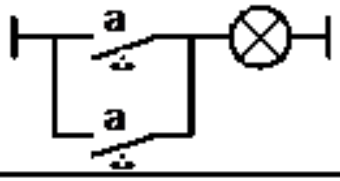
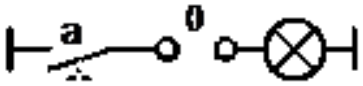
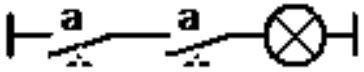
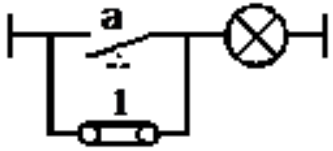
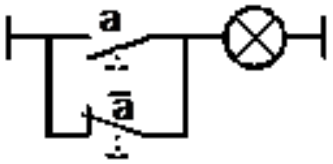
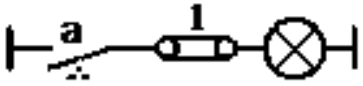
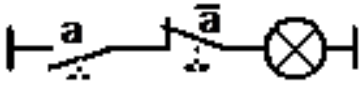
Entrée		XOR
a	b	a xor b
0	0	0
0	1	1
1	0	1
1	1	0

Fonctions à 2 variables

- Il y a 16 fonctions possibles de deux variables a,b

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

Relations particulières

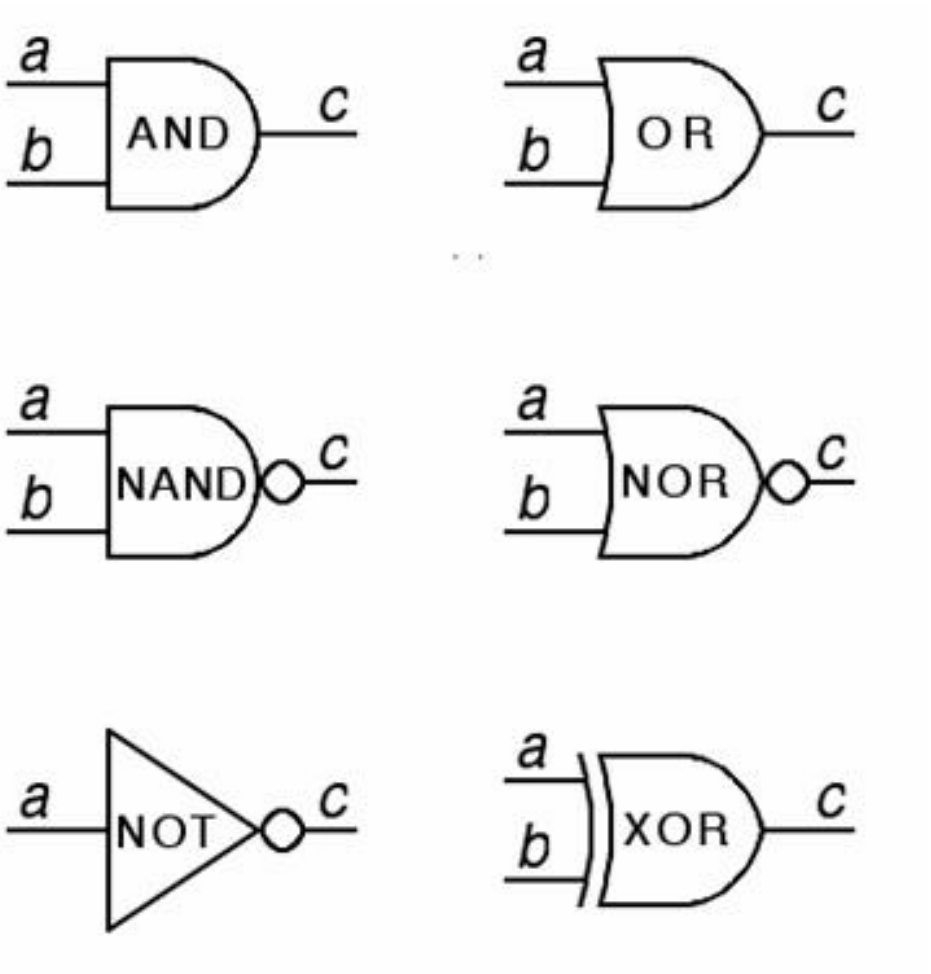
Représentation électrique	Equation	Représentation électrique	Equation
	$a + 0 = a$		$a + a = a$
	$a \cdot 0 = 0$		$a \cdot a = a$
	$a + 1 = 1$		$a + \bar{a} = 1$
	$a \cdot 1 = a$		$a \cdot \bar{a} = 0$

Opérateurs complets

- En pratique, [ET, OU , NON] permet bien d'exprimer tous les operateurs, mais il n'est pas minimal.
  - On peut réaliser la fonction ET avec des OU et des NON et la fonction OU avec des ET et des NON.

- Deux autres operateurs important du point de vue théorique dans l'algèbre de Boole :
  - les operateurs NAND (non et) et NOR (non ou).
  - Ces fonctions forment un ensemble complet ou minimal, c'est a dire qu'ils peuvent exprimer tous les operateurs.

Symboles des opérateurs logiques



Construire la table de vérité

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

a	b	c	$\overline{b.c}$	f(a,b,c)
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	0
1	0	0	0	1
1	0	1	1	1
1	1	0	0	1
1	1	1	0	1

Théorème fondamentaux de l'algèbre de Boole

- Théorème des constantes
  - $a + 0 = a$ et $a.0 = 0$
  - $a + 1 = 1$ et $a.1 = a$
- Idempotence
  - $a + a = a$
  - $a.a = a$
- Complémentation
  - $a + \overline{a} = 1$
  - $a.\overline{a} = 0$
- Commutativité
  - $a + b = b + a$
  - $a.b = b.a$
- Distributivité
  - $a + (bc) = (a + b)(a + c)$
  - $a(b + c) = (ab) + (ac)$
- Associativité
  - $a + (b + c) = (a + b) + c = a + b + c$
  - $a(bc) = (ab)c = abc$

Théorème fondamentaux de l'algèbre de Boole

- Théorème de De Morgan
  - $\overline{ab} = \overline{a} + \overline{b}$
  - $\overline{a + b} = \overline{a}.\overline{b}$
- Autres relations
  - $\overline{\overline{a}} = a$
  - $a + (ab) = a$
  - $a + \overline{a}b) = a + b$
  - $a(a + b) = a$
  - $(a + b)(a + b) = a + b$

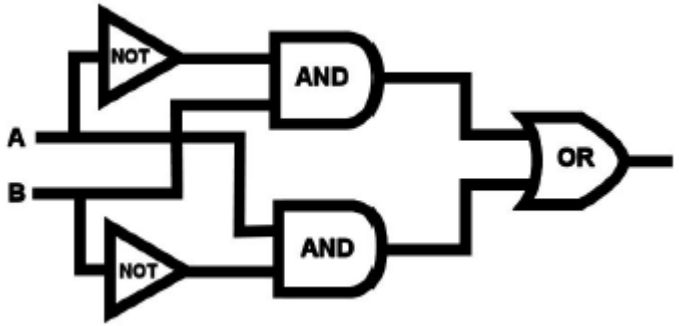
Absorptions

Propriétés du XOR

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

Exemple l'opérateur XOR

- On veut exprimer la fonction XOR (ou exclusif) en n'utilisant que les fonctions ET, OU , NON :
- avec la méthode des minterms : (prochain slide)
  - $a \oplus b = \bar{a}b + a\bar{b}$
- avec la méthode des maxterms : (prochain slide)
  - $a \oplus b = (a + b)(\bar{a} + \bar{b})$



entrées		XOR
a	*b*	$a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0

Méthode des minterms et des maxterms

- A l'aide des théorèmes précédents, il est possible d'exprimer toute fonction logique a l'aide des operateurs NON, ET, OU.

- Méthodes des minterms (i.e., somme logique des produits logiques)
  - La fonction peut être exprimée comme étant la ***somme logique*** des **minterms** correspondant a chaque sortie valant 1 dans la table de vérité. Chaque variable d'entrée est prise telle quelle si elle a la valeur 1, sinon elle est remplacée par son complément.

- Méthodes des maxterms (i.e., produit logique des sommes logiques)
  - La fonction peut être exprimée comme étant le ***produit logique*** des **maxterms** correspondant a chaque sortie valant 0 dans la table de vérité. Chaque variable d'entrée est prise telle quelle si elle a la valeur 0, sinon elle est remplacée par son complément.

- L'expression algébrique obtenu est dite forme normale (ou canonique).

Exemple des minterms

- minterm = produit logique de toutes les variables d'entrées correspondant a une sortie a 1.

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

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

Exemple des maxterms

- maxterm = somme logique de toutes les variables d'entrées correspondant a une sortie a 0.

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

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

Simplification de fonction logique: méthode algébrique

- On utilise les théorèmes de l'algèbre de Boole vu précédemment pour simplifier l'expression algébrique.

- Exemple utilisant la distributivité et la complémentation
- $f(a, b, c) = abc + ab\bar{c} + a\bar{b}c + a\bar{b}\bar{c} + \bar{a}\bar{b}c$

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

$= ab + a\bar{b} + \bar{a}\bar{b}c$

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

$= a + \bar{a}\bar{b}c$

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

$= a + \bar{b}c$

Simplification de fonction logique : Table de Karnaugh

- Basée sur l'inspection visuelle de tables judicieusement construites (table de vérité a 2 dimensions).
  - On attribue la valeur 1 aux cases correspondantes aux états d'entrée ou la fonction est vraie, sinon on attribue 0.
  - Regroupement par blocs rectangulaires de 2, 4 ou 8, 2^n variables, des cases a 1 adjacentes.
    - Attention la table se referme sur elle-même.
    - Une case a 1 peut appartenir a plusieurs blocs.
    - Blocs les plus gros possibles (on utilise un bloc une seule fois).
  - Pour chaque bloc :
    - Si une variable prend comme valeur 0 et 1, on ne la prend pas en compte.
    - On garde les variables dont la valeur ne varie pas.
    - Operateur = ET.
  - **OU** de tous les termes de tous les blocs. (on somme tous les termes des blocs)

Table de Karnaugh à 2 variables

- Table de vérité :
  - Expression algébrique canonique (minterms) :
  - f (a , b) = $\bar{a}b + a\bar{b} + ab$

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

b \ a	0	1
0	0	1
1	1	1

f (a, b) = a + b

Table de Karnaugh à 3 variables

- Table de vérité :
  - Expression algébrique canonique (minterms) :
  - f (a , b) = $\bar{a}\bar{b}c+a\bar{b}\bar{c}+a\bar{b}c+ab\bar{c}+abc$

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	1
1	1	0	1
1	1	1	1

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

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

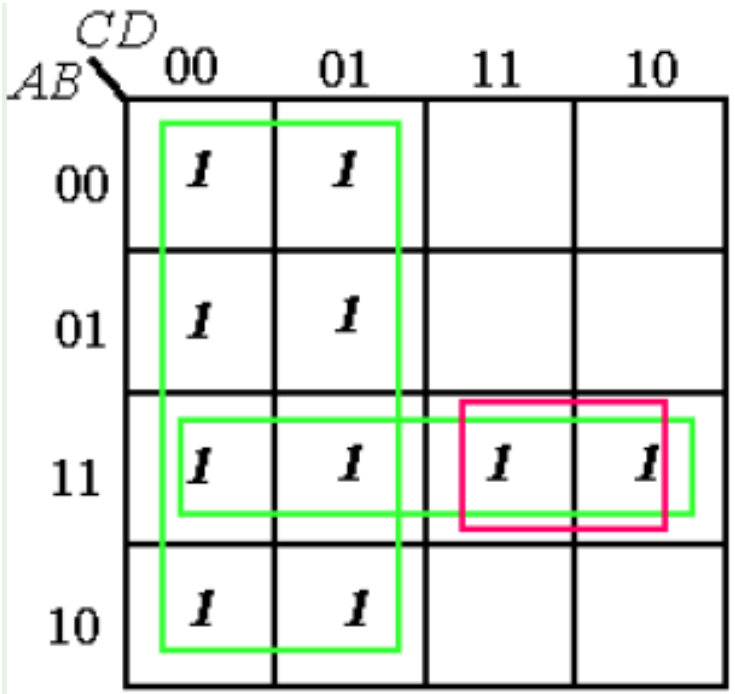
Table de Karnaugh à 4 variables

- Table de vérité :
- Expression algébrique canonique (minterms) :
  - $f(a; b; c; d) = \bar{a}\bar{b}\bar{c}\bar{d} + \bar{a}\bar{b}\bar{c}d + \bar{a}\bar{b}c\bar{d} + \bar{a}\bar{b}cd + a\bar{b}\bar{c}\bar{d} + a\bar{b}\bar{c}d + a\bar{b}cd + \bar{a}bcd + a\bar{b}cd + \bar{a}bc\bar{d} + abcd$

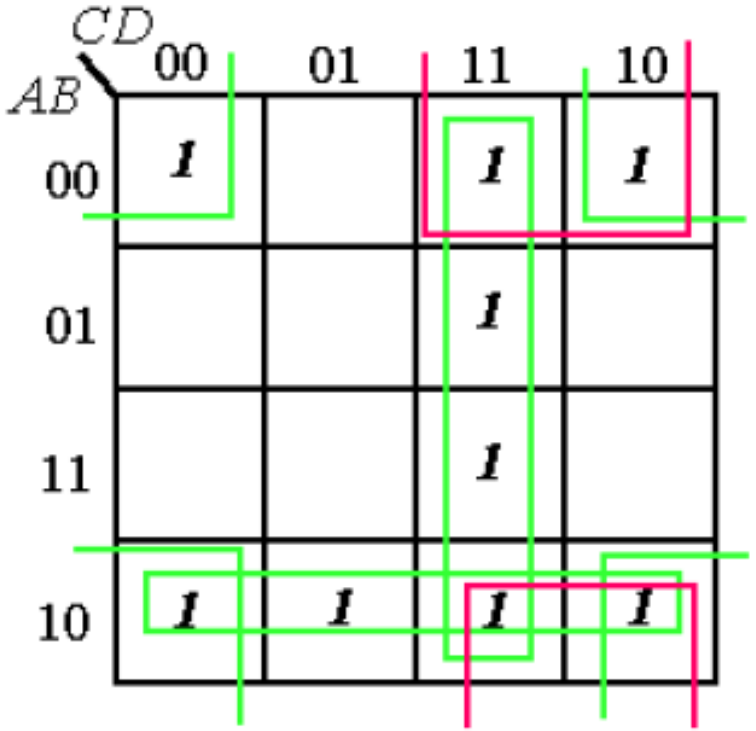
cd \ ab	00	01	11	10
00	1	0	0	1
01	0	1	1	1
11	0	0	0	1
10	1	0	0	1

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

Autres exemples



$\overline{C} + AB$



$A\overline{B} + CD + \overline{B}\overline{D}$

Synthèse d'un circuit combinatoire

Méthode de synthèse

- A partir d'une fonction logique, déterminer un circuit logique réalisant cette fonction et obtenir le meilleur (i.e., le plus simple en nombre de portes, de connexions) :

1. construire la table de vérité de la fonction logique ;
2. dériver une expression algébrique (par exemple par la méthode des minterms) ;
3. simplifier cette expression (méthode algébrique ou tables de Karnaugh) ;
4. réaliser la fonction logique a l'aide d'operateurs divers (NON, ET, OU, XOR, NAND, NOR, etc.) pour obtenir un logigramme.

Analyse d'un circuit combinatoire

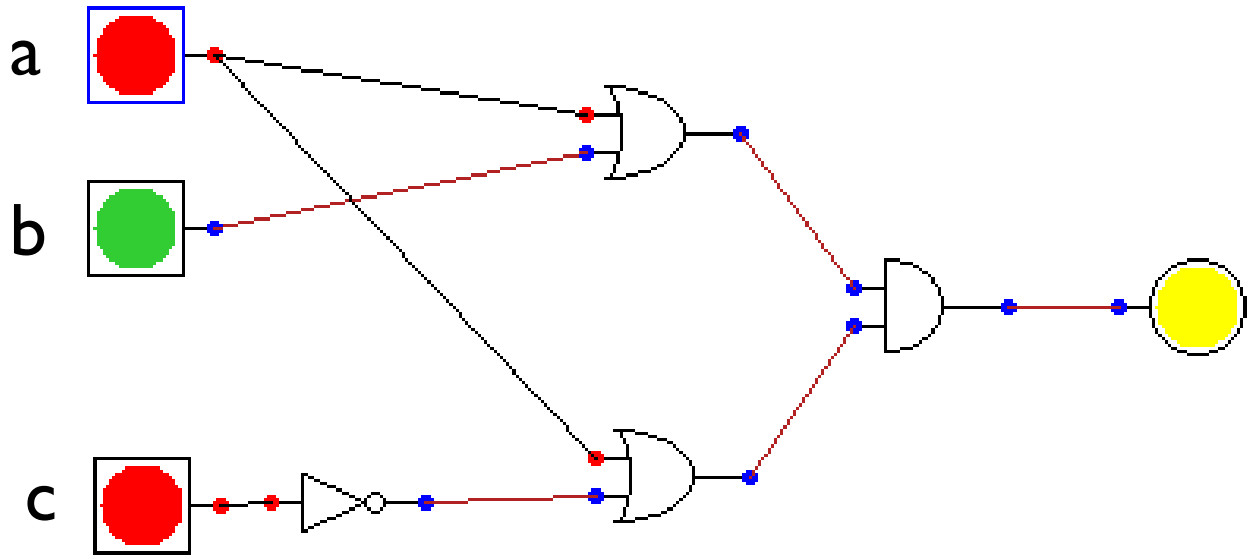
- L'analyse est l'opération inverse de la synthèse.

Méthode de d'analyse

- Retrouver la fonction d'un circuit dont on connaît uniquement le logigramme :

1. En procédant des entrées vers les sorties, donner, pour chaque operateur l'expression de sa sortie en fonction de ses entrées, jusqu'à obtention d'une expression pour chaque fonction réalisée par le circuit ;
2. Donner la table de vérité correspondante ;
3. En déduire le rôle du circuit.

Exemple d'analyse



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

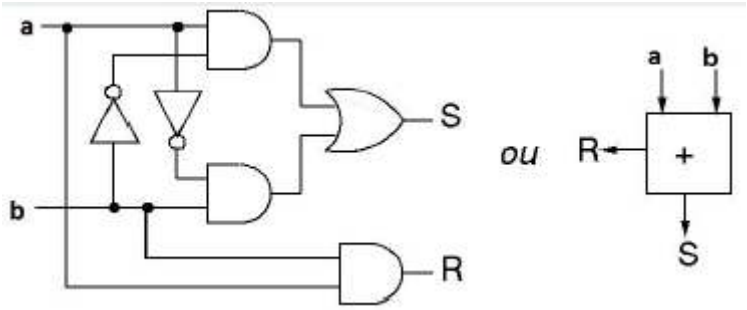
Circuits logiques les plus importants

- Demi-additionneur (addition sans gestion de la retenue) et additionneur complet (addition avec gestion de la retenue) ;
- Multiplexeur (plusieurs signaux en entrées, 1 seule sortie) et démultiplexeur (un seul signal en entrée et plusieurs sorties) ;
- Décodeur, codeur et transcodeur (e.g., conversion de base).

Synthèse d'un demi additionneur

- Circuit logique capable de faire la somme de 2 nombres binaires mais qui ne tient pas compte de la retenue éventuelle provenant d'une opération précédente.

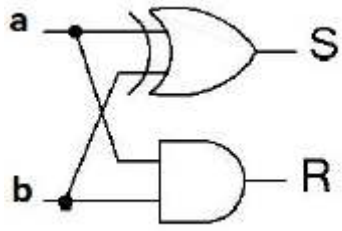
a	b	Sortie S	Retenue R
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



Expression algébrique canonique (minterms) :

$S = a\bar{b} + \bar{a}b = a \oplus b$

$R = ab$



Synthèse d'un étage additionneur

- Circuit logique capable de faire la somme de 2 nombres binaires et d'une retenue provenant d'une opération précédente.

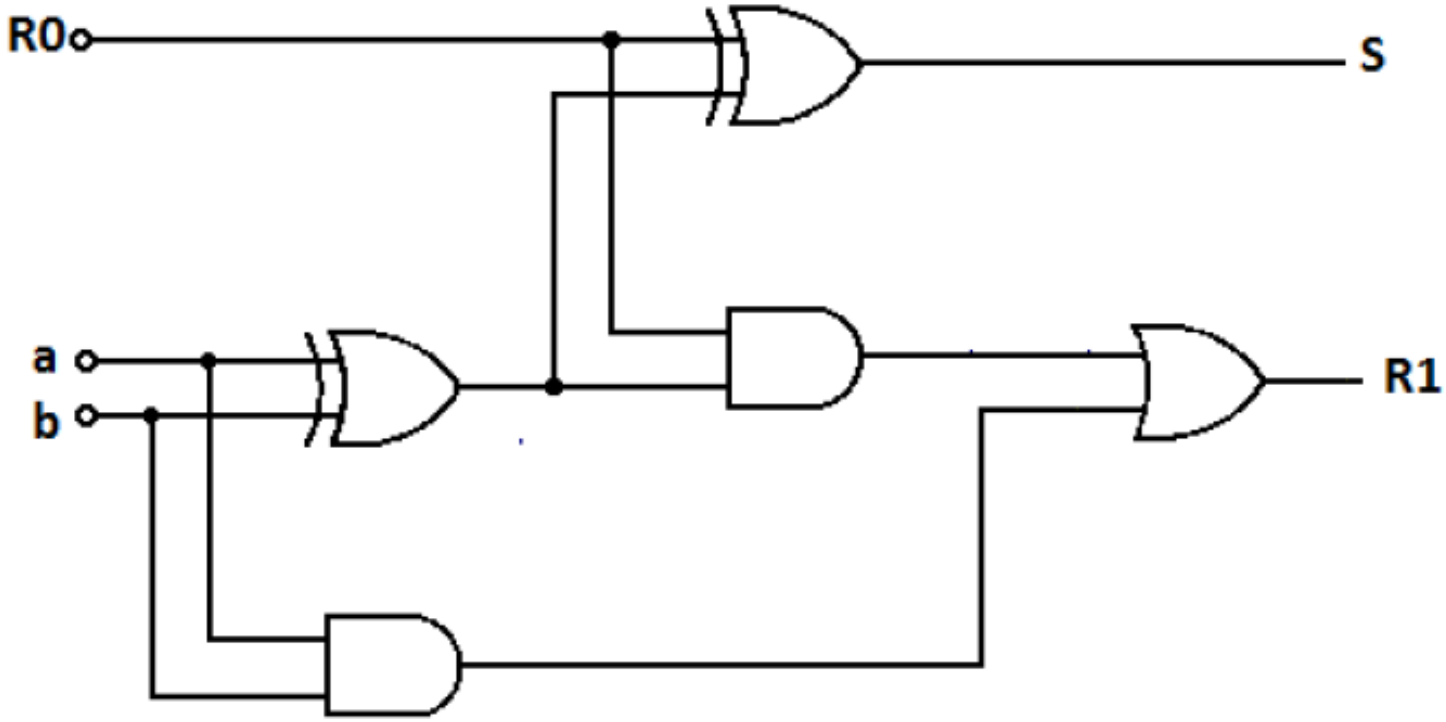
a	b	R0	Sortie S	R1
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Expression algébrique canonique (minterms) :

$$
\begin{aligned}
S &= \bar{a}\bar{b}R_0 + \bar{a}b\,\overline{R_0} + a\overline{bR_0} + abR_0 \\
&= R_0(\bar{a}\bar{b} + ab) + \overline{R_0}(\bar{a}b + a\bar{b}) \\
&= R_0(\overline{a \oplus b}) + \overline{R_0}(a \oplus b) \\
&= R_0 \oplus (a \oplus b)
\end{aligned}
$$

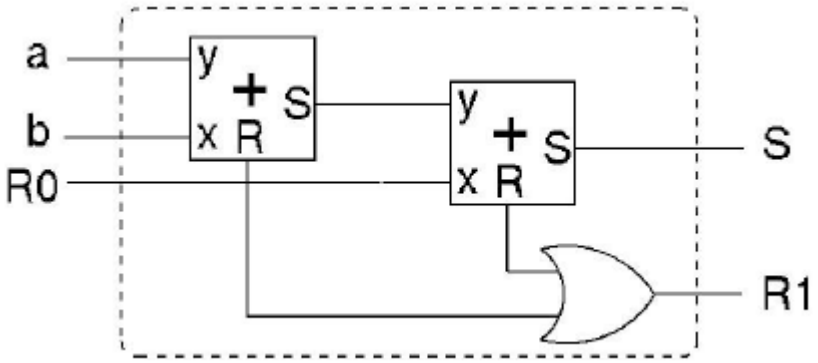
$$
\begin{aligned}
R_1 &= \bar{a}bR_0 + a\bar{b}R_0 + ab\overline{R_0} + abR_0 \\
&= R_0(\bar{a}b + a\bar{b}) + ab(R_0 + \overline{R_0}) \\
&= R_0(a \oplus b) + ab
\end{aligned}
$$

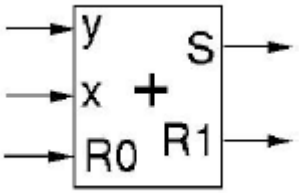
Logigramme d'un étage additionneur



Additionneur binaire complet

- L'étage d'additionneur est compose de 2 demi-additionneurs et d'un OU. Il fait la somme de 2 bits en tenant compte d'une éventuelle retenue.

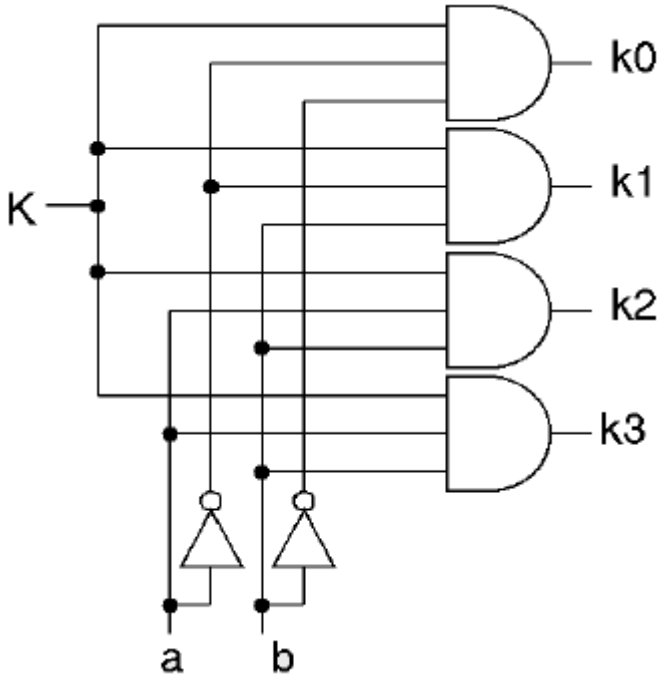


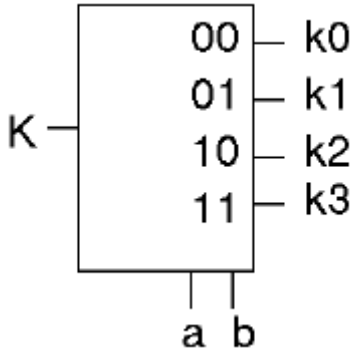


- L'additionneur complet est obtenu en utilisant en parallèle plusieurs étages additionneurs (il faut autant d'étages que de bits composants les nombre binaires a additionner).

Démultiplexeur

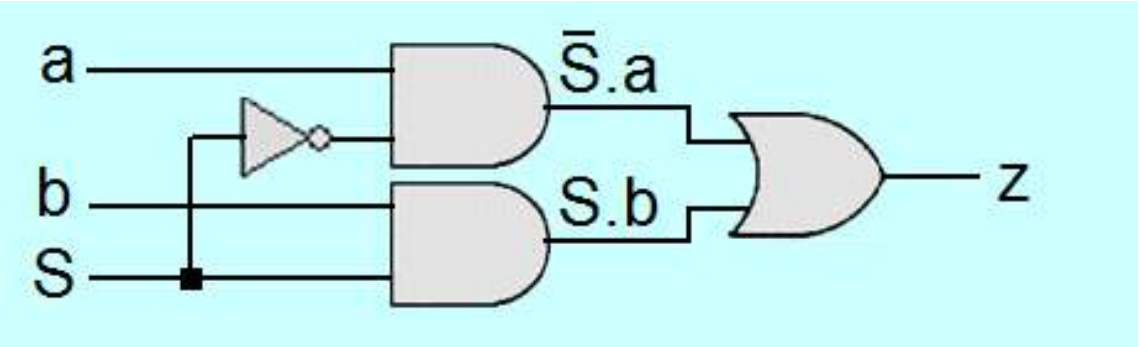
- 1 entrée, n variables, 2^n sorties
- Une des sorties prend la valeur de l'entrée (K) selon la valeur des n variables : la variable K est aiguillée sur l'une des 4 sorties.
- Utile pour choisir la source d'un signal





Multiplexeur

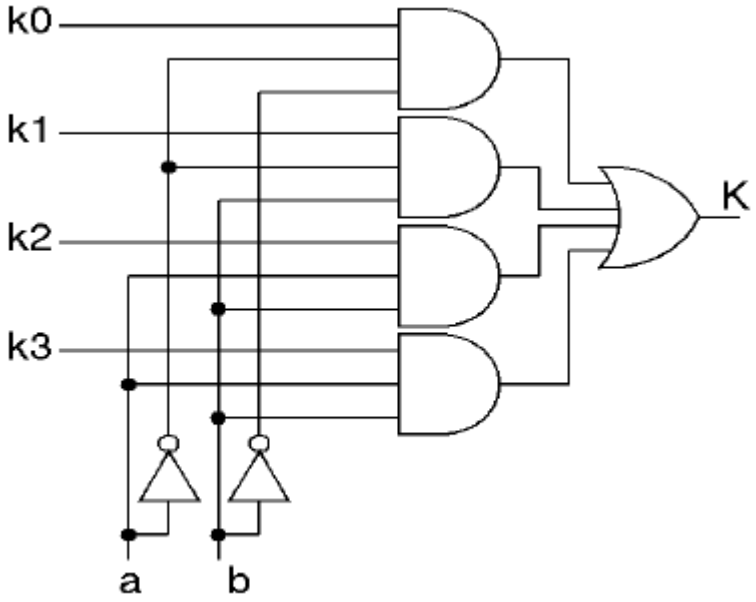
- Multiplexeur 2 bits ou 2 vers 1

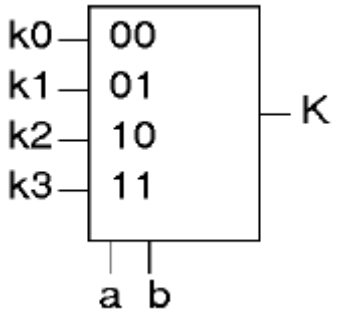


$$z = \overline{S}.a + S.b$$

multiplexeur

- 2^n entrées, n variables, 1 sortie
- La sortie (K) prend la valeur d'une des entrées selon la valeur des n variables : une des 4 entrées est aiguillée sur la sortie K.
- Utile pour choisir la source d'un signal



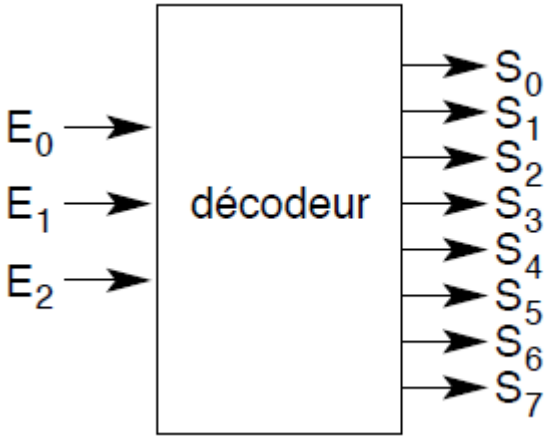


Application de multiplexeurs

- Fonction universelle (i.e., un multiplexeur a n variables peut réaliser les $2^{2puis(n)}$ fonctions logiques a n variables ;
- Multiplexage (i.e., concentrer plusieurs lignes en une seule ou faire l'opération inverse) ;
- Codage, décodage, transcodage.

Décodeur

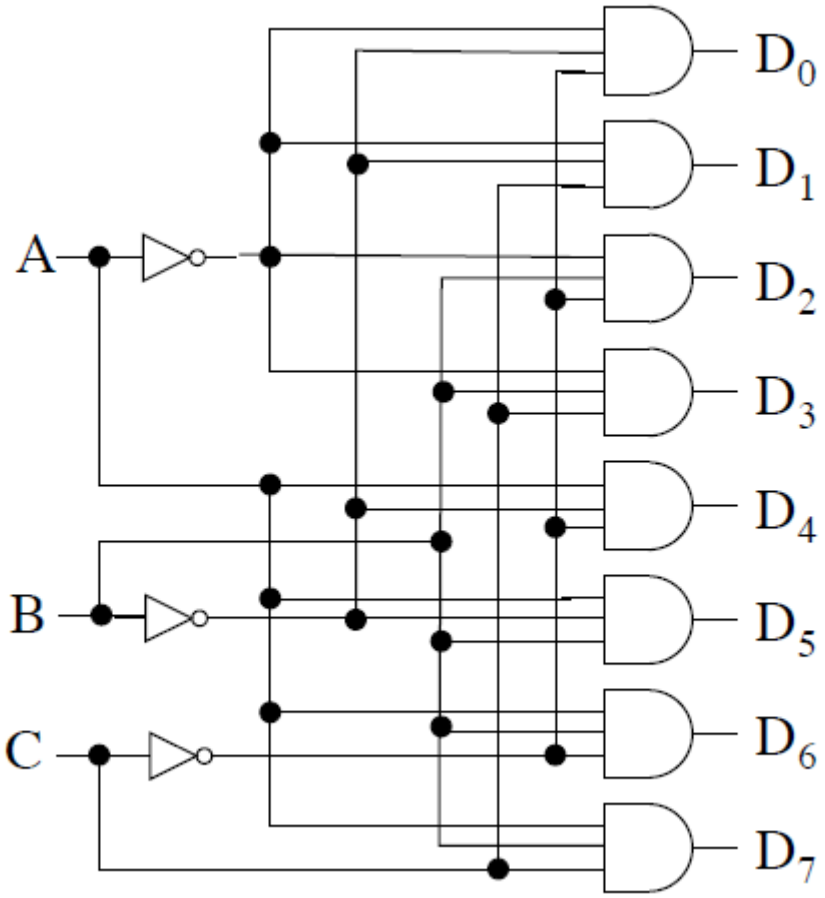
- Fait correspondre a un code en entrée (sur n lignes) une seule sortie active (i.e., a 1) parmi les 2^n sorties possibles.



entrées			sorties							
E_2	E_1	E_0	S_7	S_6	S_5	S_4	S_3	S_2	S_1	S_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

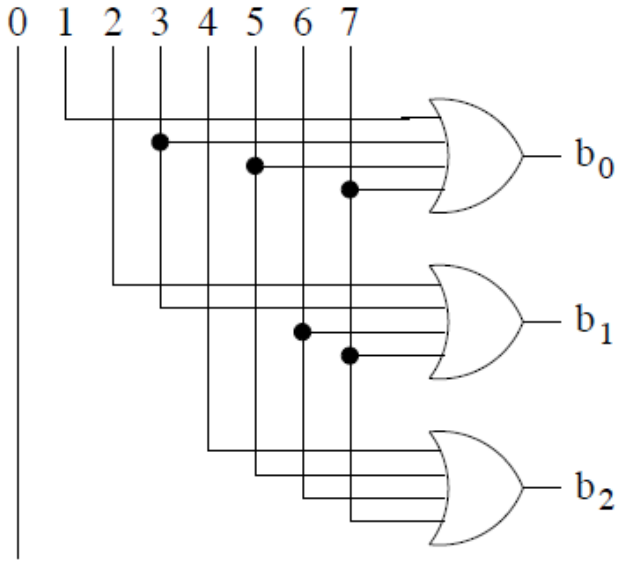
- Le décodeur peut être utilise pour convertir un nombre binaire en nombre décimal ou pour adresser une mémoire.

Décodeur



Codeur

- Fait correspondre a une entrée active, parmi les 2^n entrées, un code sur n lignes en sortie.



- Un transcodeur fait correspondre une entrée sur n lignes correspondant a un certain codage, une sortie sur m lignes correspondant a un autre codage.