Chapitre 5: Circuits combinatoires Élémentaires

5.2 Circuits logiques

5.2.4 Synthèse d'un circuit combinatoire

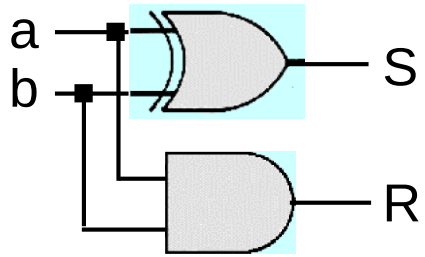
Synthèse d'un demi-additionneur binaire

Table de vérité du demi-additionneur (qui ne tient pas compte d'une retenue antérieure)

a b	S	R	
0 0	0	0	
0 1	1	0	$\bar{a}.b$
1 0	1	0	$a.\bar{b}$
1 1	0	1	$a.b$

$R = a.b$

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



Additionneur binaire

5.2 Circuits logiques

5.2.4 Synthèse d'un circuit combinatoire

Synthèse d'un additionneur binaire de 1 bit

Table de vérité de l'additionneur 1 bit

a	b	R	S	R'
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

$R' = \bar{a}.b.R + a.\bar{b}.R + a.b.\bar{R} + a.b.R$

$S = \bar{a}.\bar{b}.R + \bar{a}.b.\bar{R} + a.\bar{b}.\bar{R} + a.b.R$

On simplifie :

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

$R' = (a \oplus b).R + a.b$

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

$S = \overline{(a \oplus b)}.R + (a \oplus b).\bar{R}$

$S = (a \oplus b) \oplus R$

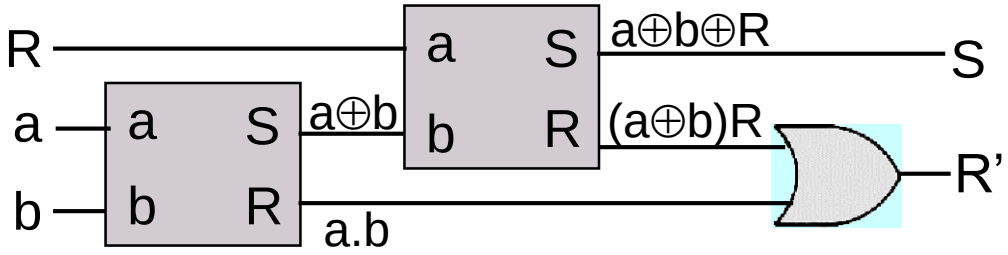
Demi-additionneurs

5.2 Circuits logiques

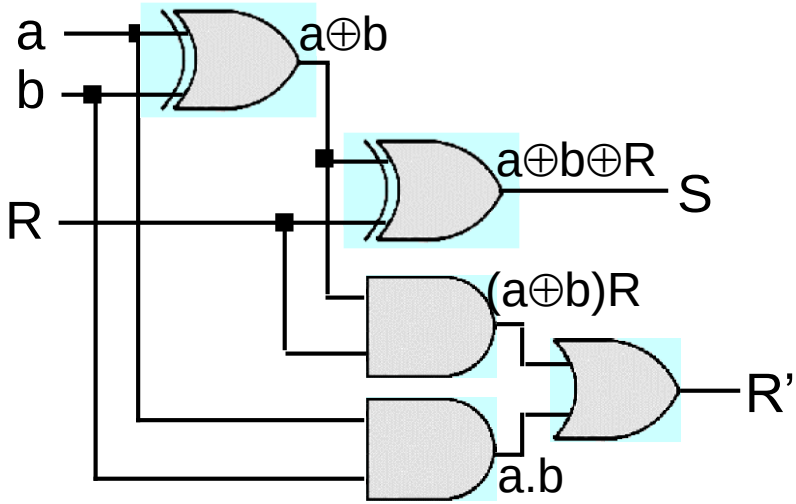
5.2.4 Synthèse d'un circuit combinatoire

Synthèse d'un additionneur binaire

Réalisation au moyen de 2 demi-additionneurs



Réalisation complète d'un additionneur 1 bit

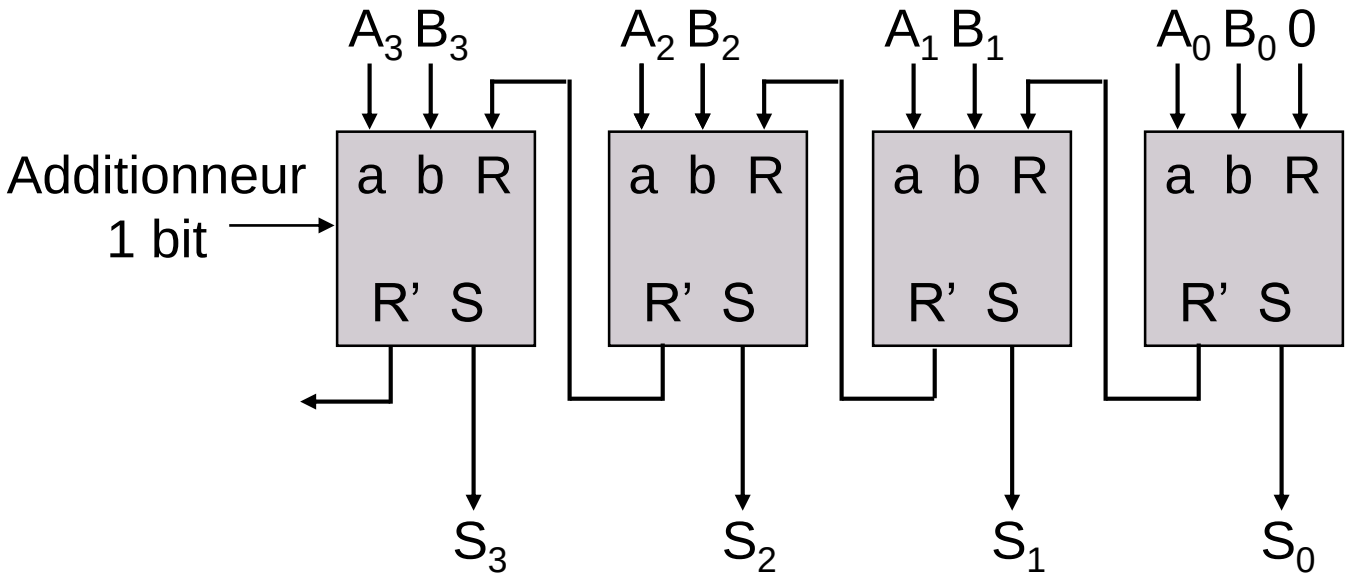


Additionneur à plusieurs bits

5.2 Circuits logiques

5.2.4 Synthèse d'un circuit combinatoire

Additionneur à plusieurs bits

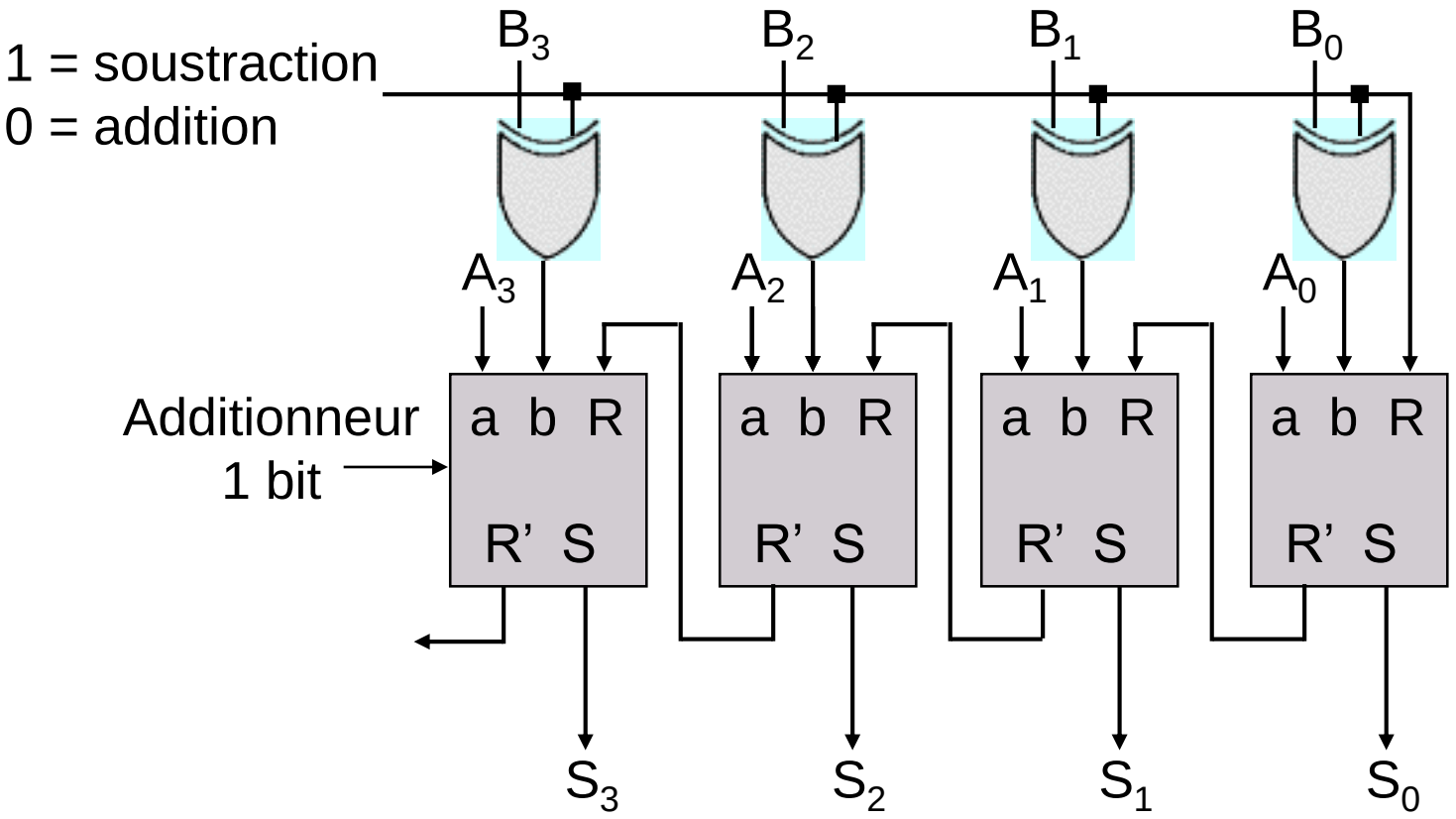


Additionneur/soustracteur 4 bits

5.2 Circuits logiques

5.2.4 Synthèse d'un circuit combinatoire

Additionneur/soustracteur 4 bits



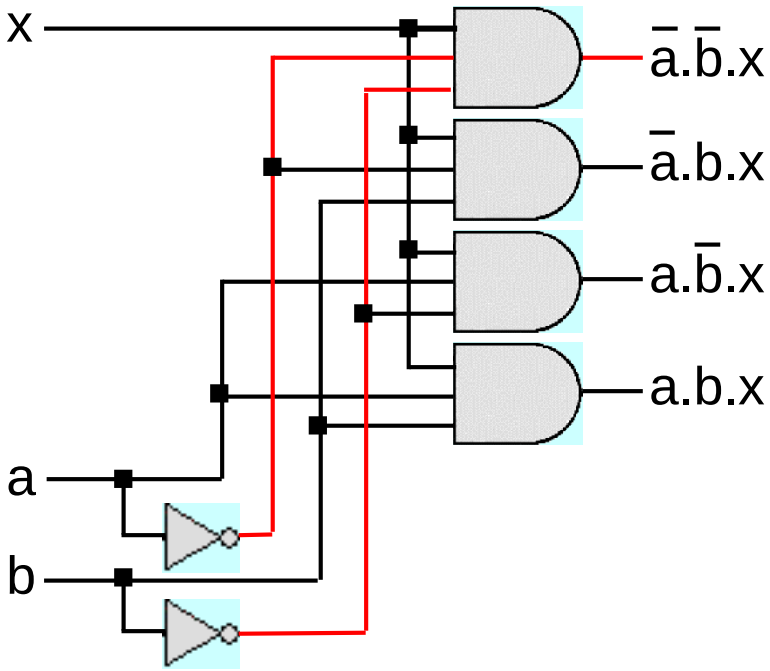
Multiplexeurs et démultiplexeurs

5.2 Circuits logiques

5.2.6 Multiplexeurs et démultiplexeurs

Démultiplexeur 4 bits ou 1 vers 4

Ce circuit est utile pour choisir la destination d'un signal.



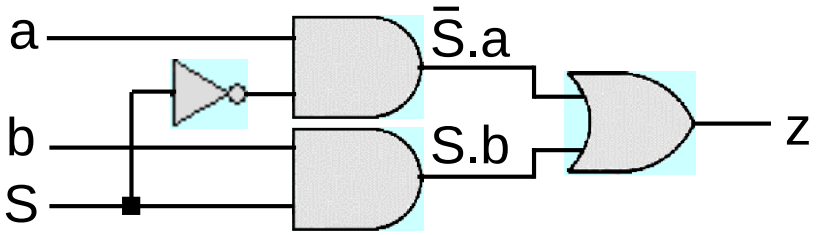
Multiplexeurs et démultiplexeurs

5.2 Circuits logiques

5.2.6 Multiplexeurs et démultiplexeurs

Multiplexeur 2 bits ou 2 vers 1

Ce circuit est utile pour choisir la source d'un signal.



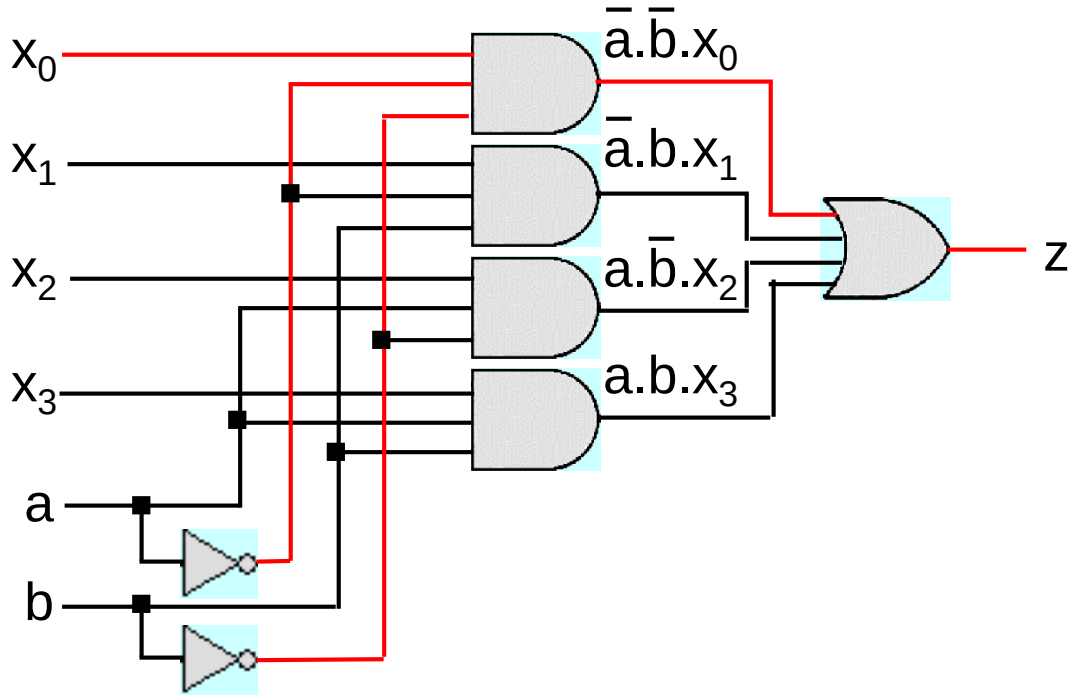
$$z = \overline{S}.a + S.b$$

Multiplexeurs et démultiplexeurs

5.2 Circuits logiques

5.2.6 Multiplexeurs et démultiplexeurs

Multiplexeur 4 bits ou 4 vers 1

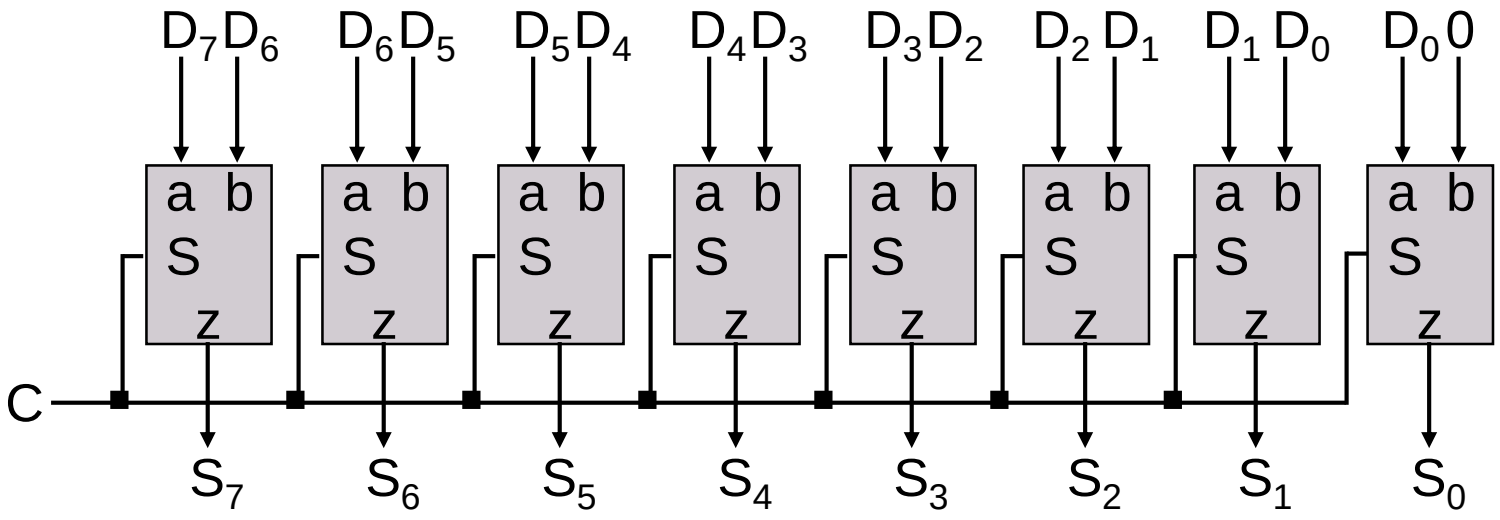


Multiplexeurs et démultiplexeurs

5.2 Circuits logiques

5.2.6 Multiplexeurs et démultiplexeurs

Décaleur de 1 bit vers la gauche

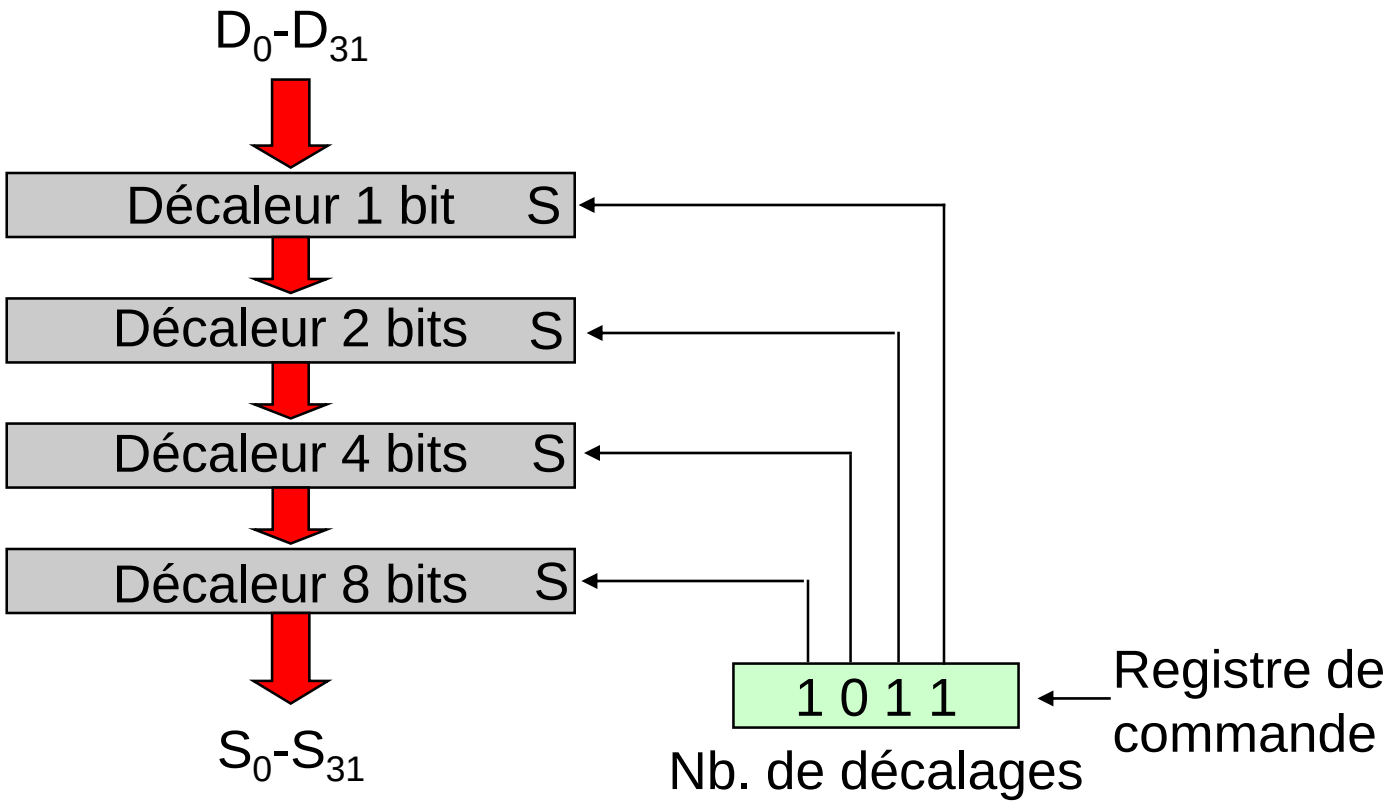


Multiplexeurs et démultiplexeurs

5.2 Circuits logiques

5.2.6 Multiplexeurs et démultiplexeurs

Décaleur à barillet



Multiplexeurs et démultiplexeurs

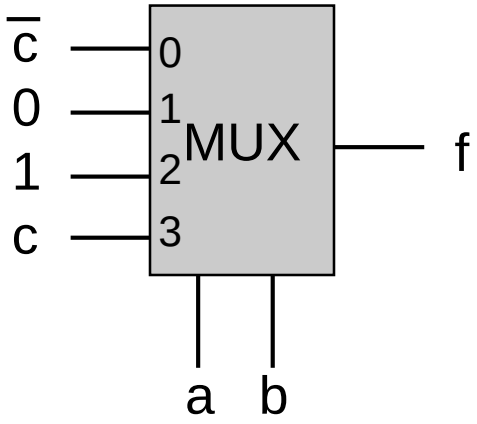
5.2 Circuits logiques

5.2.6Multiplexeurs et démultiplexeurs

Utilisation d'un multiplexeur pour réaliser n'importe quelle fonction logique. Exemple :

Table de vérité

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

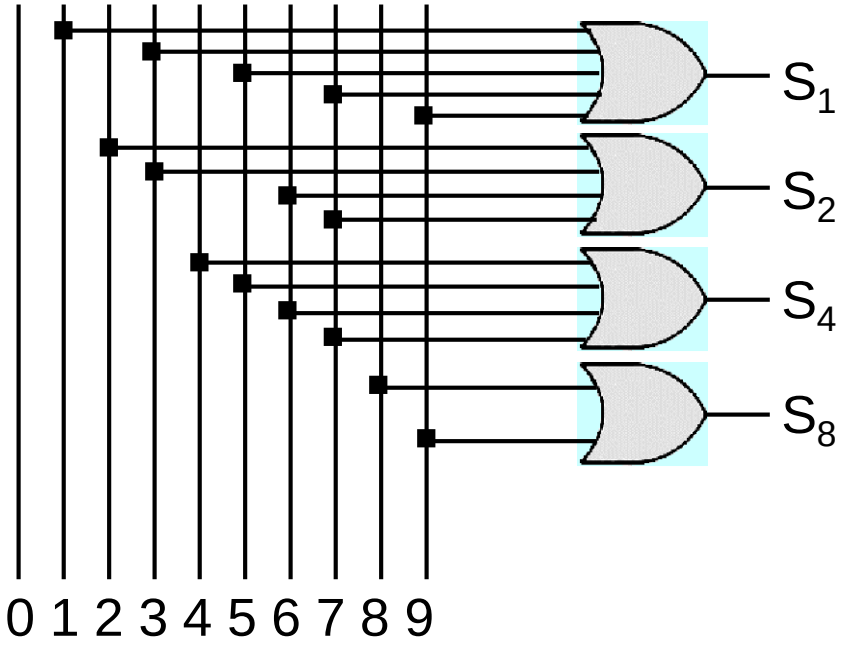


Décodeurs, codeurs, transcodeurs

5.2 Circuits logiques

5.2.7 Décodeurs, codeurs, transcodeurs

Codeur : code en binaire le numéro de la ligne activée.

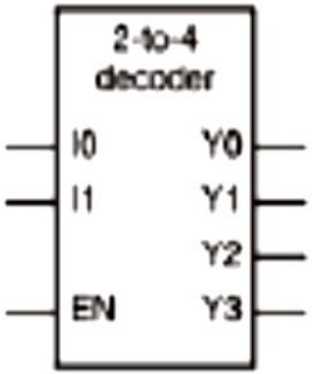


Décodeur 2-à-4 (ou 1 parmi 4)

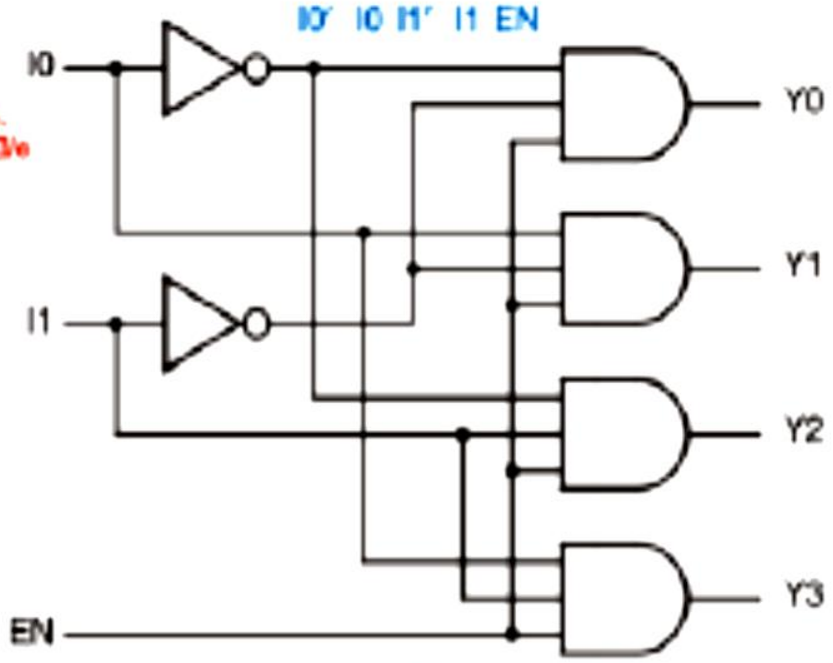
pp. 287 à 295

Table 5-4
Truth table for a 2-to-4 binary decoder.

Inputs			Outputs			
EN	I1	I0	Y3	Y2	Y1	Y0
0	x	x	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0



(a)



(b)

Décodeurs

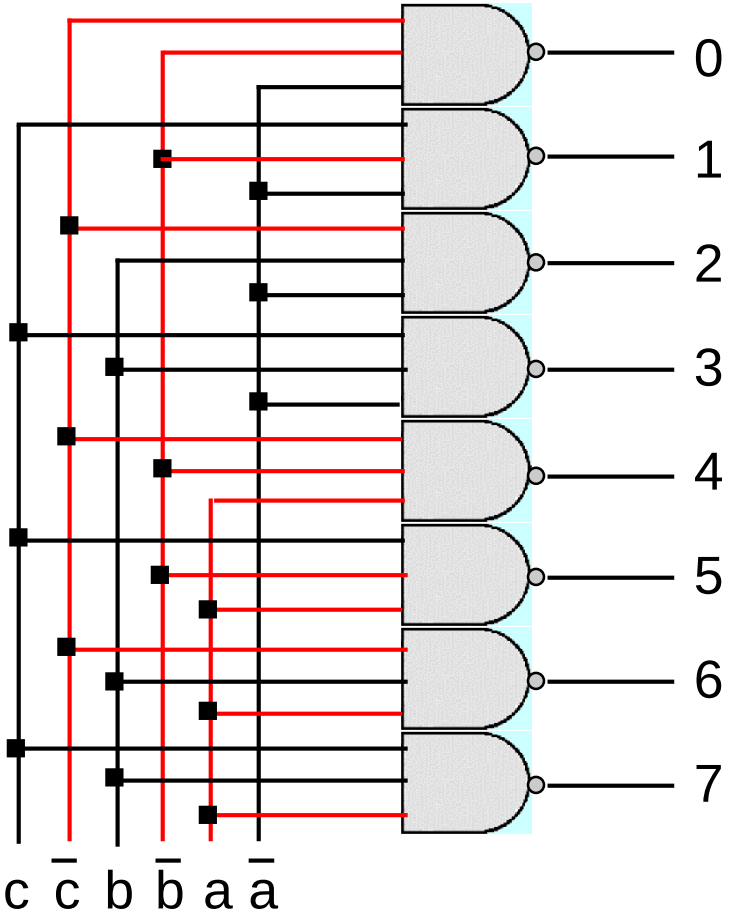
5.2 Circuits logiques

5.2.7 Décodeurs, codeurs, transcodeurs

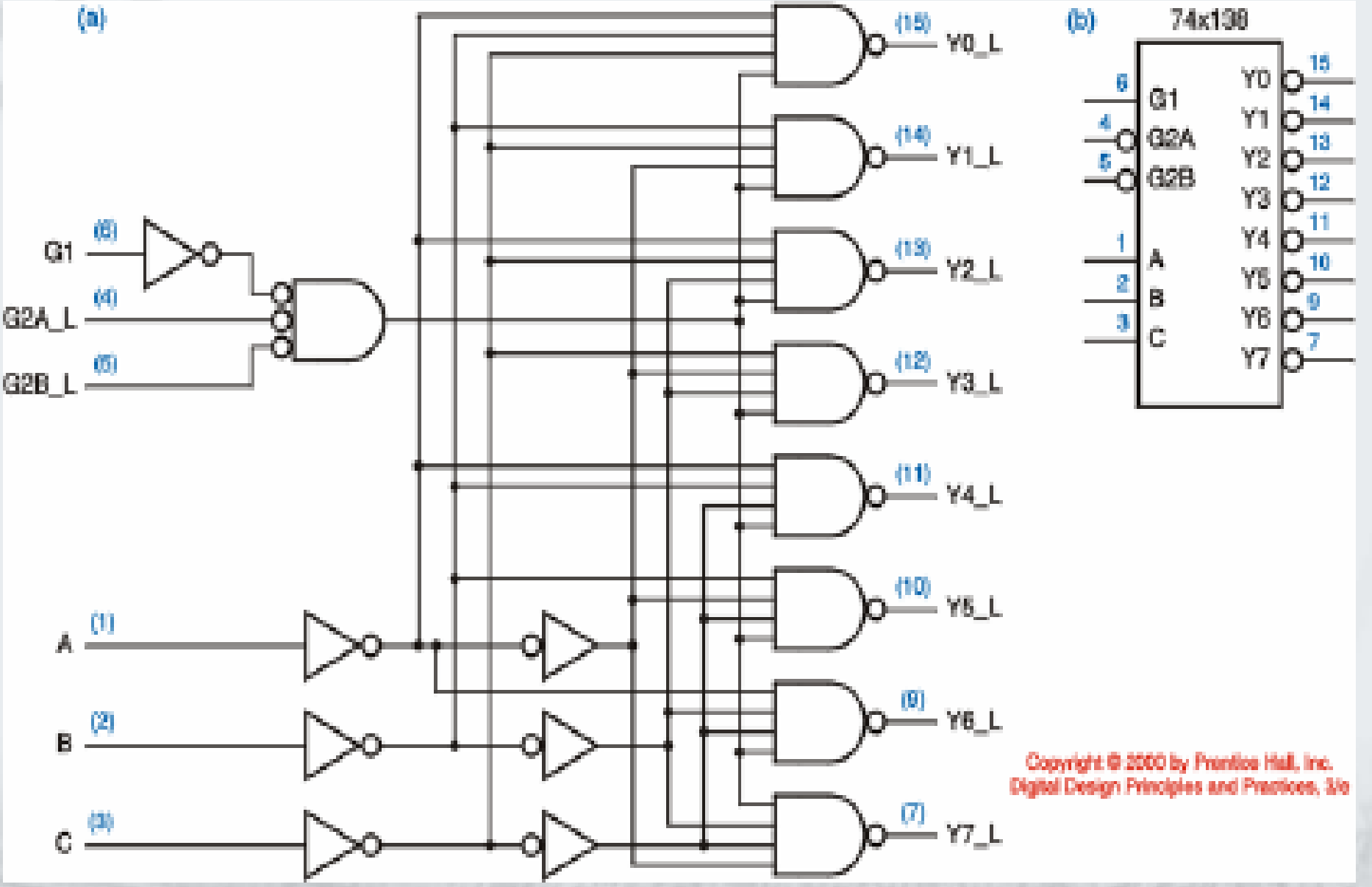
Décodeur 3 vers 8

Une seule sortie à la fois est 0 et est choisie par le code abc.

$a.\overline{b}.\overline{c} = 4$



Décodeur 3-à-8 (1 parmi 8)



Décodeur 3-à-8 (1 parmi 8)

Table 5-7 Truth table for a 74x138 3-to-8 decoder.

Inputs						Outputs							
G1	G2A_L	G2B_L	C	B	A	Y7_L	Y6_L	Y5_L	Y4_L	Y3_L	Y2_L	Y1_L	Y0_L
0	x	x	x	x	x	1	1	1	1	1	1	1	1
x	1	x	x	x	x	1	1	1	1	1	1	1	1
x	x	1	x	x	x	1	1	1	1	1	1	1	1
1	0	0	0	0	0	1	1	1	1	1	1	1	0
1	0	0	0	0	1	1	1	1	1	1	1	0	1
1	0	0	0	1	0	1	1	1	1	1	0	1	1
1	0	0	0	1	1	1	1	1	1	0	1	1	1
1	0	0	1	0	0	1	1	1	0	1	1	1	1
1	0	0	1	0	1	1	1	0	1	1	1	1	1
1	0	0	1	1	0	1	0	1	1	1	1	1	1
1	0	0	1	1	1	0	1	1	1	1	1	1	1

Transcodeur

5.2 Circuits logiques

5.2.7 Décodeurs, codeurs, transcodeurs

Transcodeur BCD-excédent-3

Table de vérité:

A	B	C	D	X	Y	Z	T	I
0	0	0	0	0	0	1	1	0
0	0	0	1	0	1	0	0	0
0	0	1	0	0	1	0	1	0
0	0	1	1	0	1	1	0	0
0	1	0	0	0	1	1	1	0
0	1	0	1	1	0	0	0	0
0	1	1	0	1	0	0	1	0
0	1	1	1	1	0	1	0	0
1	0	0	0	1	0	1	1	0
1	0	0	1	1	1	0	0	0

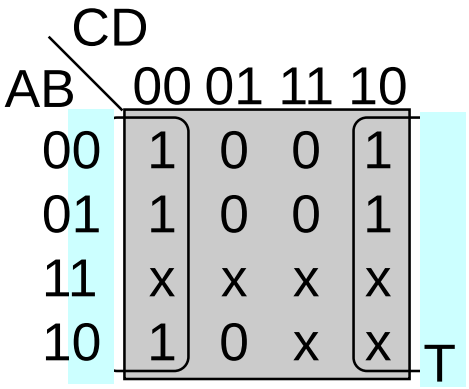
A	B	C	D	X	Y	Z	T	I
1	0	1	0	x	x	x	x	1
1	0	1	1	x	x	x	x	1
1	1	0	0	x	x	x	x	1
1	1	0	1	x	x	x	x	1
1	1	1	0	x	x	x	x	1
1	1	1	1	x	x	x	x	1

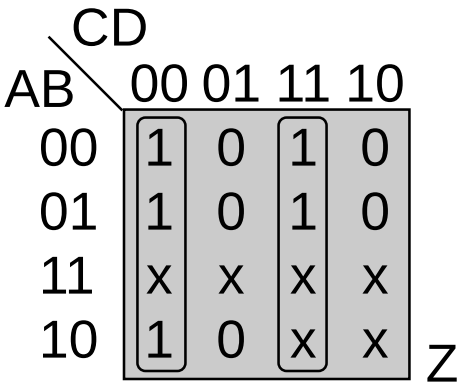
Transcodeur BCD-excédent-3

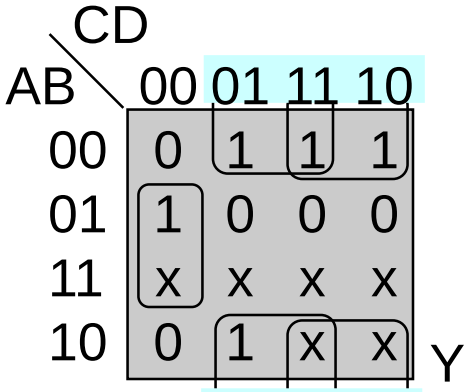
5.2 Circuits logiques

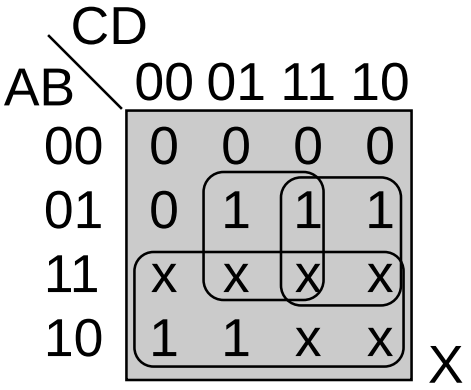
5.2.7 Décodeurs, codeurs, transcodeurs

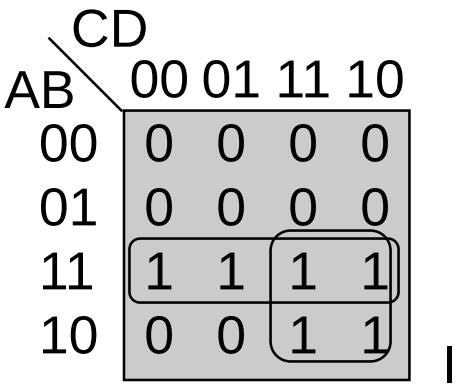
Transcodeur BCD-excédent-3











$T = \overline{D}$

$Z = C\overline{D} + \overline{C}\,\overline{D} = \overline{C \oplus D}$

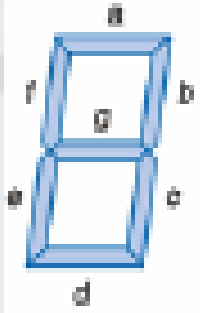
$Y = B\overline{C}\,\overline{D} + \overline{B}C + \overline{B}D$

$X = A + BC + BD$

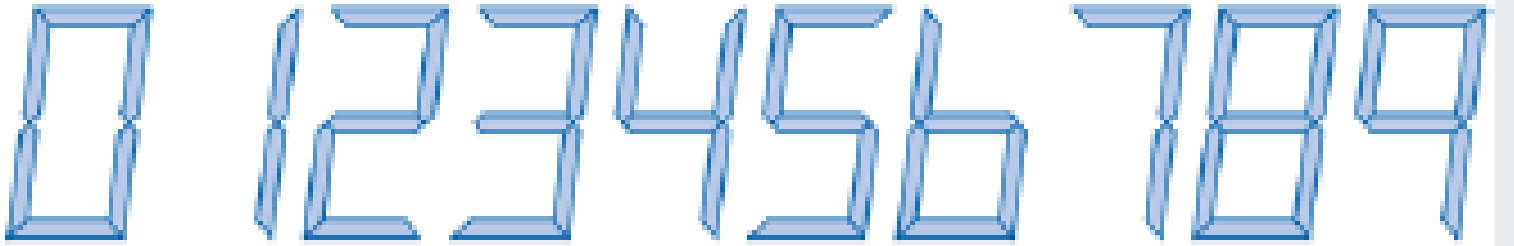
$I = AB + AC$

Décodeur DCB-7 segments

DCB : Décimal codé binaire



(a)

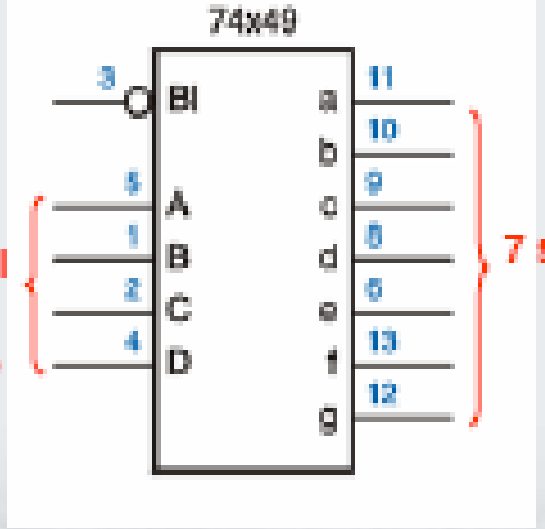


(b)

Table 4-21 Truth table for a 7-bit 8 seven-segment decoder.

Inputs					Outputs						
BI_L	D	C	B	A	a	b	c	d	e	f	g
0	x	x	x	x	0	0	0	0	0	0	0
1	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
1	0	0	1	0	1	1	0	1	1	0	1
1	0	0	1	1	1	1	1	1	0	0	1
1	0	1	0	0	0	1	1	0	0	1	1
1	0	1	0	1	1	0	1	1	0	1	1
1	0	1	1	0	0	0	1	1	1	1	1
1	0	1	1	1	1	1	1	0	0	0	0
1	1	0	0	0	1	1	1	1	1	1	1
1	1	0	0	1	1	1	1	0	0	1	1
1	1	0	1	0	0	0	0	1	1	0	1
1	1	0	1	1	0	0	1	1	0	0	1
1	1	1	0	0	0	1	0	0	0	1	1
1	1	1	0	1	1	0	0	1	0	1	1
1	1	1	1	0	0	0	0	1	1	1	1
1	1	1	1	1	0	0	0	0	0	0	0

(b)

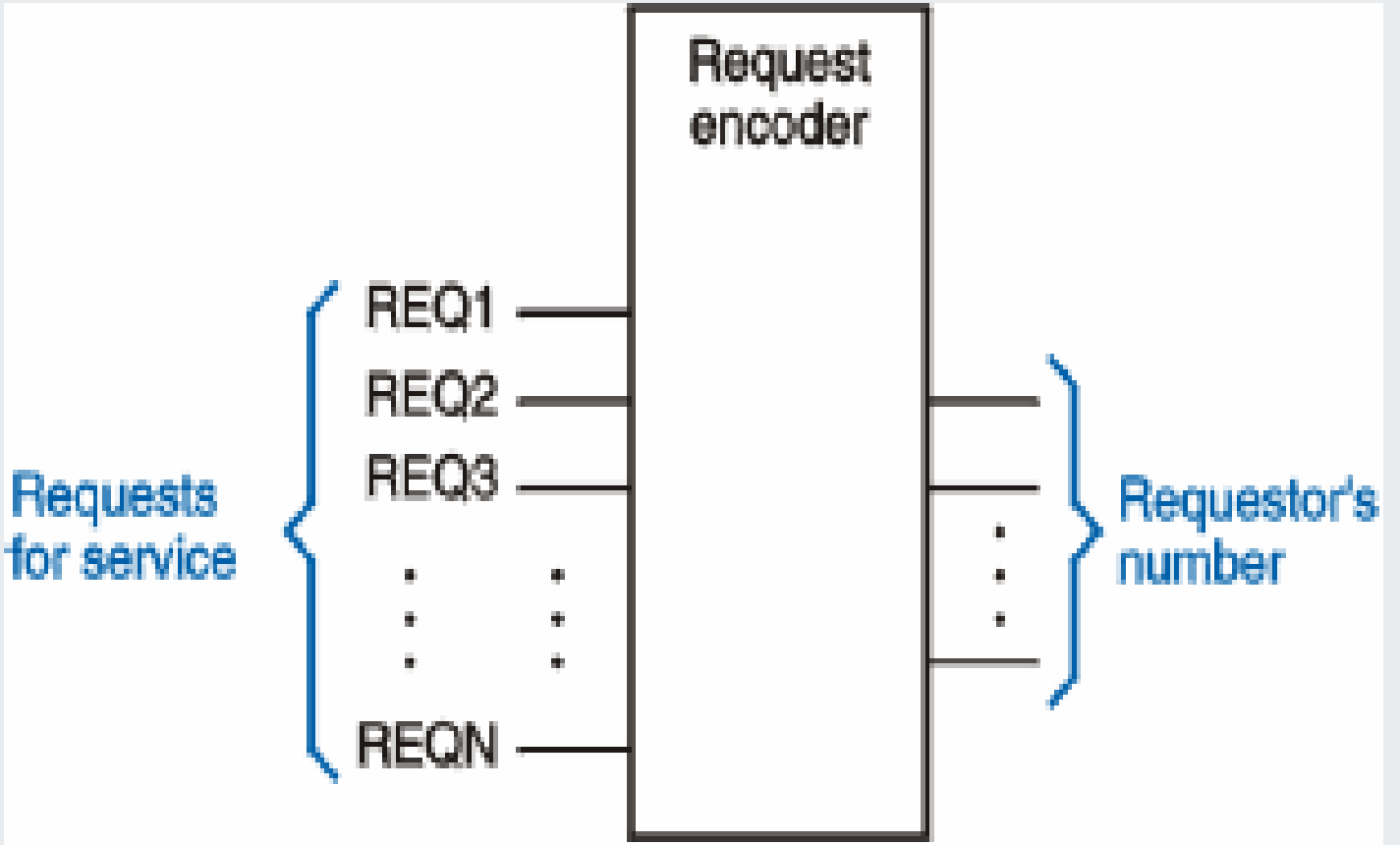


DCB décimal codé binaire

7 segments

Encodeur de priorité*

pp. 296 à 300



Encodeur de priorité à 8 entrées

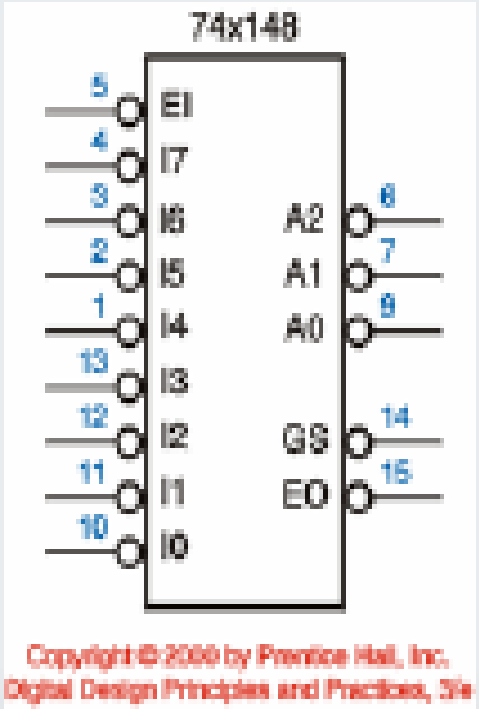


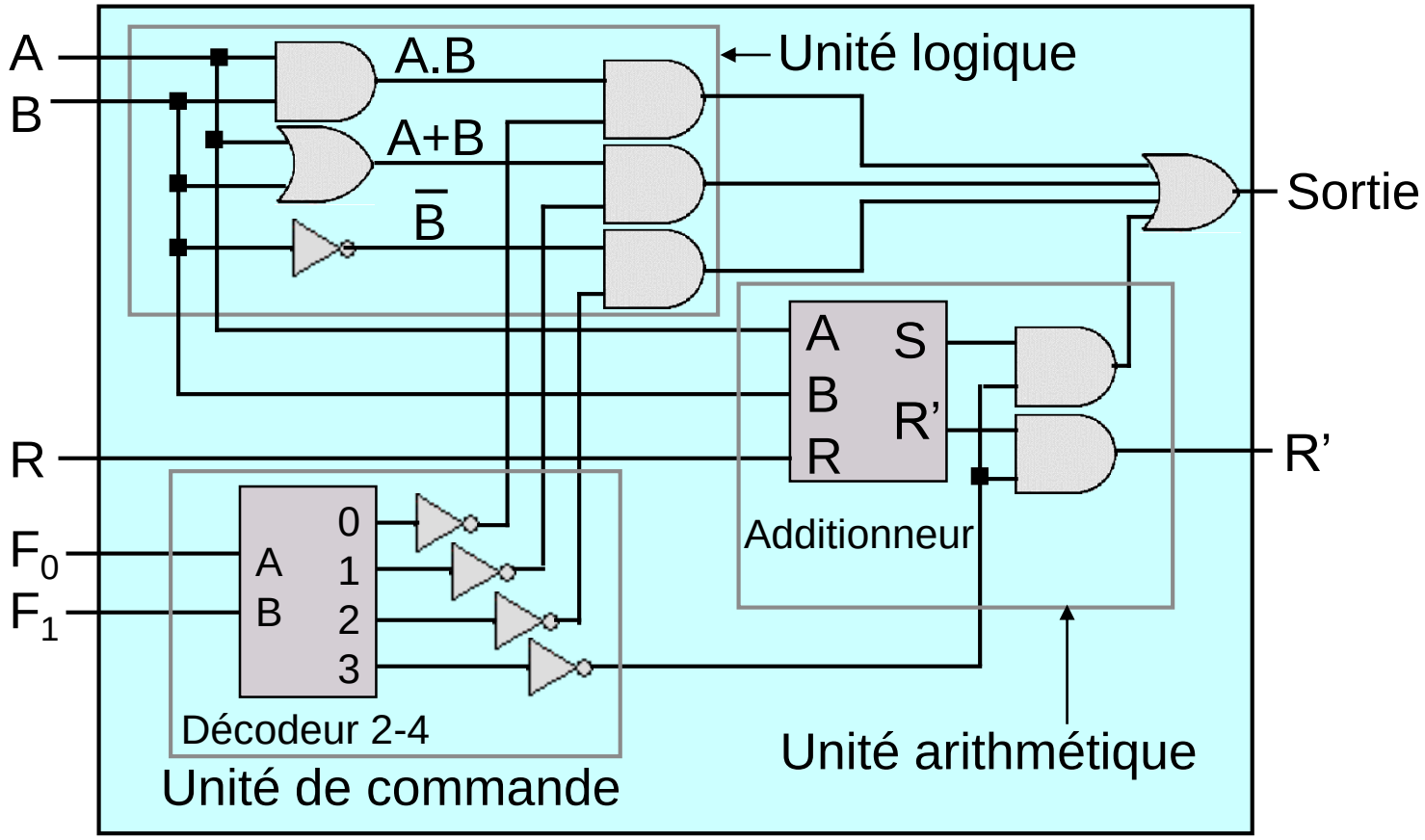
Table 5-23 Truth table for a 74x148 8-input priority encoder.

Inputs									Outputs				
EI_L	I0_L	I1_L	I2_L	I3_L	I4_L	I5_L	I6_L	I7_L	A2_L	A1_L	A0_L	GS_L	EO_L
1	x	x	x	x	x	x	x	x	1	1	1	1	1
0	x	x	x	x	x	x	x	0	0	0	0	0	1
0	x	x	x	x	x	x	0	1	0	0	1	0	1
0	x	x	x	x	x	0	1	1	0	1	0	0	1
0	x	x	x	x	0	1	1	1	0	1	1	0	1
0	x	x	x	0	1	1	1	1	1	0	0	0	1
0	x	x	0	1	1	1	1	1	1	0	1	0	1
0	x	0	1	1	1	1	1	1	1	1	0	0	1
0	0	1	1	1	1	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	1	1	0

UAL (Unité Arithmétique et Logique)

5.2 Circuits logiques

UAL élémentaire

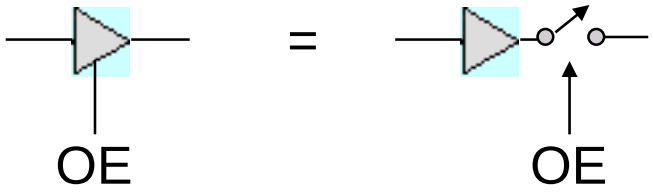


Logique à trois états

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'état pas connectée). On ajoute une entrée Output Enable (OE) à chaque circuit et on n'en active qu'un à la fois.



Logique programmable

5.2 Circuits logiques

Logique programmable

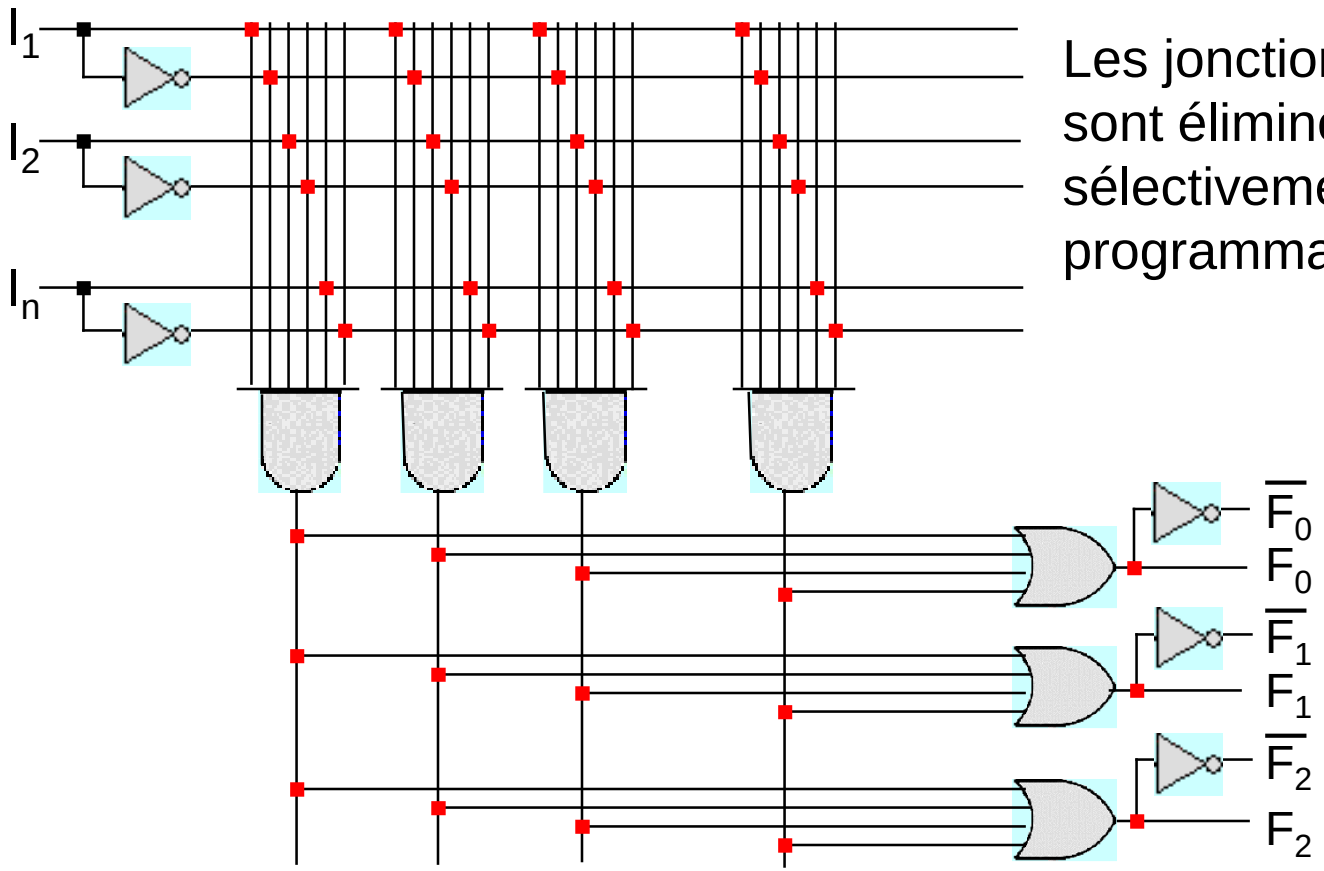
Les circuits de logique programmable PLA (Programmable Logic Array), PLD (Programmable Logic Devices), EPLD (Eraseble PLD), etc. sont basés sur le fait que toute fonction logique peut être exprimée comme une somme de minterms.

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.

Logique programmable combinatoire

5.2 Circuits logiques

Logique programmable combinatoire



Les jonctions en rouge sont éliminées (brûlées) sélectivement lors de la programmation.

ROM (Read Only Memory)

5.2 Circuits logiques

Logique avec ROM

Il est possible de réaliser des circuits logiques au moyen de mémoires ROM (Read-Only Memory). Aucune simplification n'est nécessaire. Les entrées de la table de vérité servent d'adresse dans la ROM et le contenu de chaque adresse est la sortie désirée pour cette combinaison de variables d'entrée, la sortie pouvant avoir un ou plusieurs bits.

Logique avec ROM

5.2 Circuits logiques

Logique avec ROM

Exemple : transcodeur binaire à code Gray

Table de vérité

ABCD	EFGH	ABCD	EFGH
0000	0000	1000	1100
0001	0001	1001	1101
0010	0011	1010	1111
0011	0010	1011	1110
0100	0110	1100	1010
0101	0111	1101	1011
0110	0101	1110	1001
0111	0100	1111	1000