REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

Université Larbi Ben M'hidi Oum el Bouaghi
ISTA Ain M'lila
Département Réseaux et Télécommunications

1ère Année Licence
Module : Informatique 1

Logique combinatoire et séquentielle

© Dr BENACER Imad Jan 2021

Année Universitaire : 2020 / 2021

Logique combinatoire

Objectifs

- Apprendre la structure de quelques circuits combinatoires souvent utilisés (demi additionneur , additionneur complet,……..).
- Apprendre comment utiliser des circuits combinatoires pour concevoir d’autres circuits plus complexes.

I. Les Circuits combinatoires

Introduction

- On appel circuit combinatoire un circuit logique dont les sorties dépendent uniquement des entrées.

- $S_i=F(E_i)$
- $S_i=F(E_1,E_2,\ldots,E_n)$

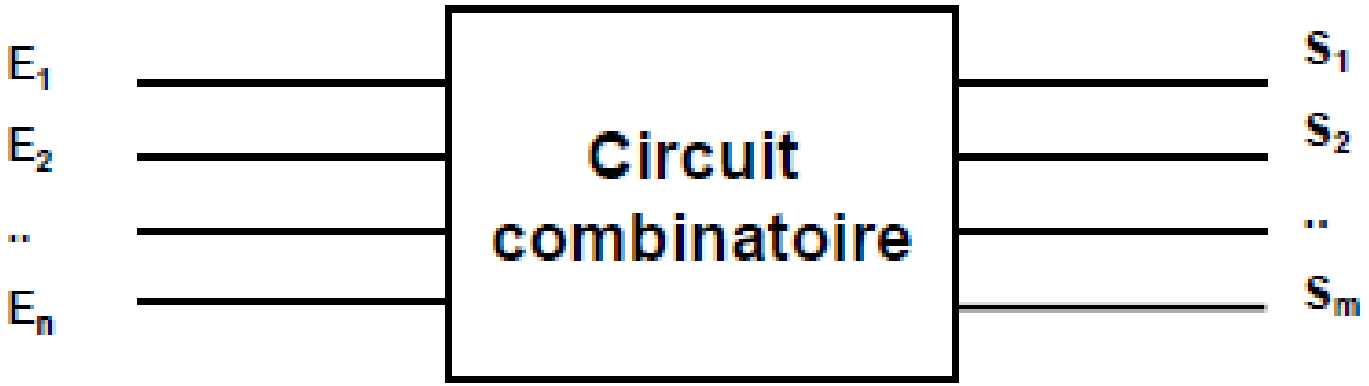


Schéma Bloc

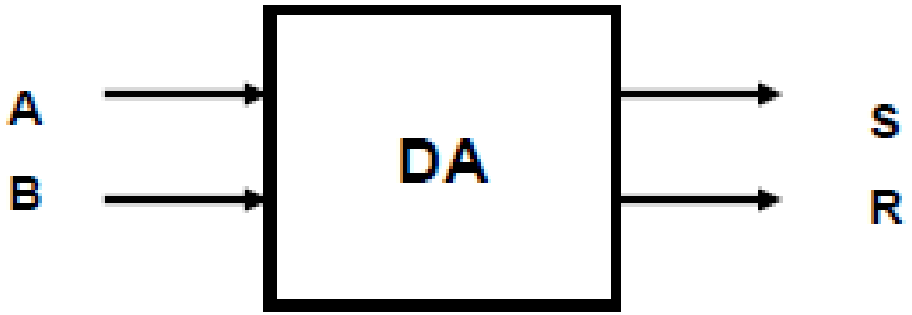
I. Les Circuits combinatoires

Exemple de Circuits combinatoires

1. Multiplexeur
2. Demultiplexeur
3. Encodeur
4. Décodeur
5. Transcodeur
6. Demi Additionneur
7. Additionneur complet
8. Comparateur

2. Demi Additionneur

- Le demi additionneur est un circuit combinatoire qui permet de réaliser la somme arithmétique de deux nombres A et B chacun sur un bit.
- A la sotie on va avoir la somme S et la retenu R (Carry).



Pour trouver la structure (le schéma) de ce circuit on doit en premier dresser sa table de vérité

- En binaire l'addition sur un seul bit se fait de la manière suivante:

$$\begin{cases} 0+0 = 00 \\ 0+1 = 01 \\ 1+0 = 01 \\ 1+1 = 10 \end{cases}$$

- La table de vérité associée :

A	B		R	S
0	0		0	0
0	1		0	1
1	0		0	1
1	1		1	0

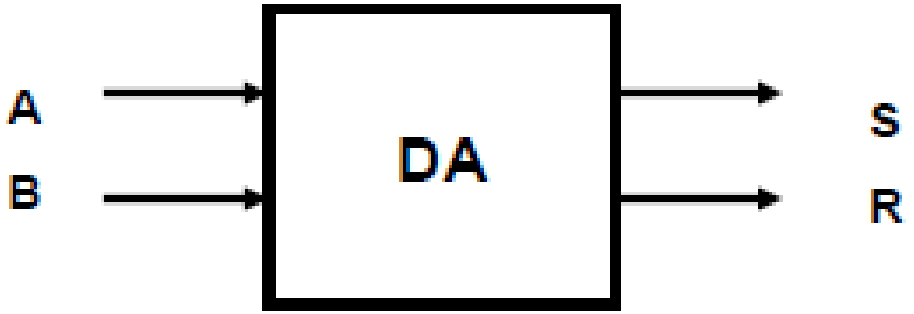
De la table de vérité on trouve :

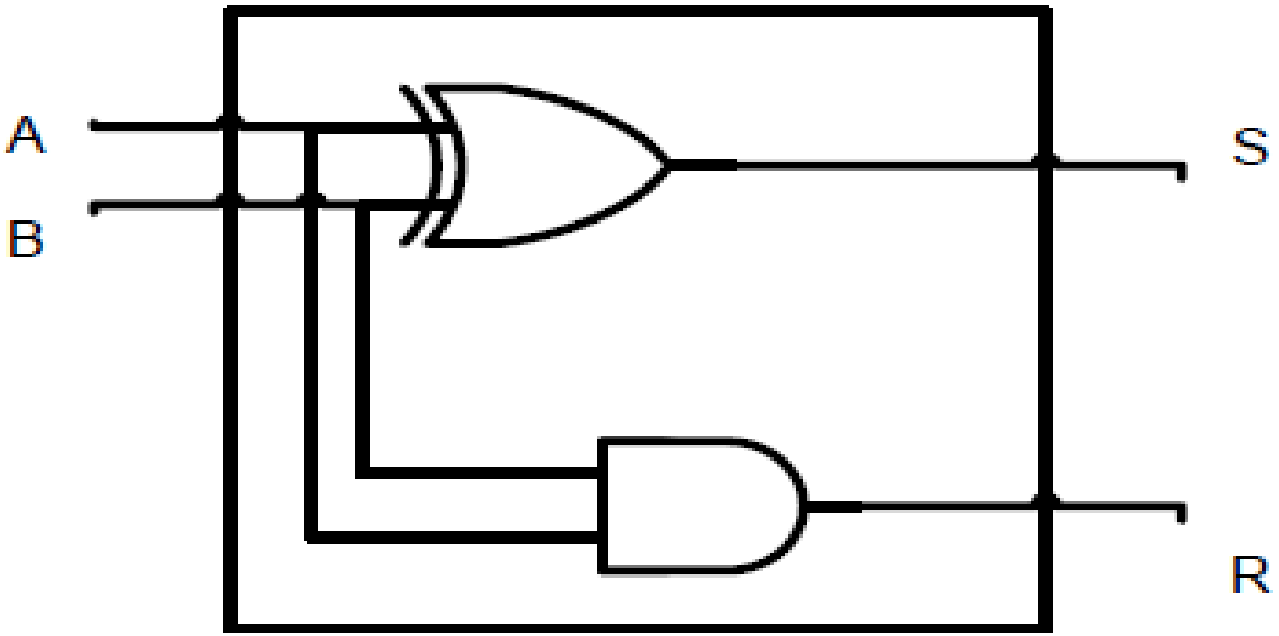
$$R = A.B$$

$$S = \overline{A}.B + A.\overline{B} = A \oplus B$$

$$R = A.B$$

$$S = A \oplus B$$





3. L'additionneur complet

- En binaire lorsque on fait une addition il faut tenir en compte de la retenue entrante.

r_4	r_3	r_2	r_1	$r_0 = 0$
	a_4	a_3	a_2	a_1
+	b_4	b_3	b_2	b_1
r_4	s_4	s_3	s_2	s_1

	r_{i-1}	
	a_i	
+	b_i	
r_i	s_i	

3.1 Additionneur complet 1 bit

- L'additionneur complet un bit possède 3 entrées :
  - a_i : le premier nombre sur un bit.
  - b_i : le deuxième nombre sur un bit.
  - r_{i-1} : le retenue entrante sur un bit.
- Il possède deux sorties :
  - S_i : la somme
  - R_i la retenue sortante

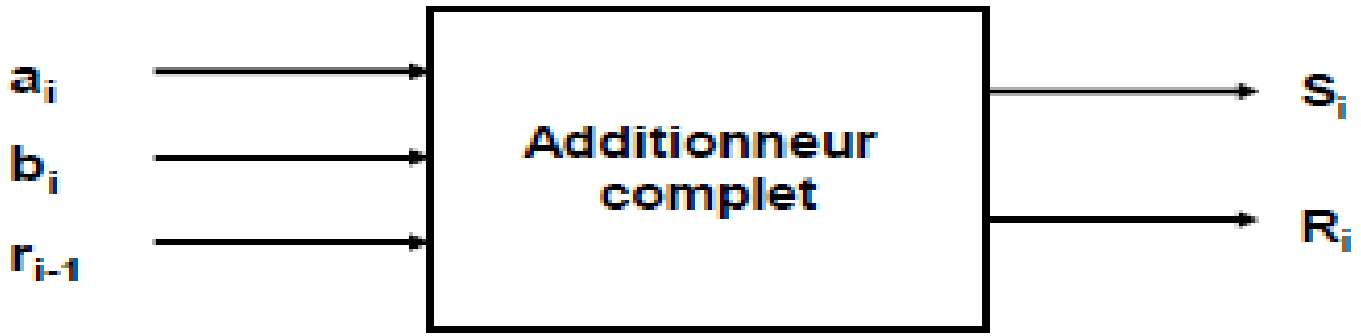
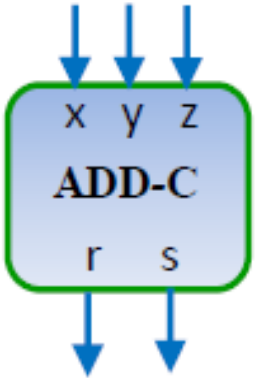


Schéma bloc:



Les entrées:

x , y, z (la retenue entrante) sur 1 bit.

Les sorties:

s est la somme, sur 1 bit.

r, la retenue sortante, sur 1 bit.

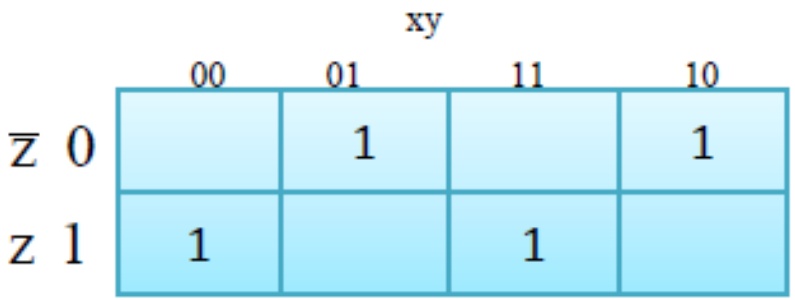
La fonction: est claire

$$\begin{array}{rr} & z \\ + & y \\ + & x \\ \hline & r \quad s \end{array}$$

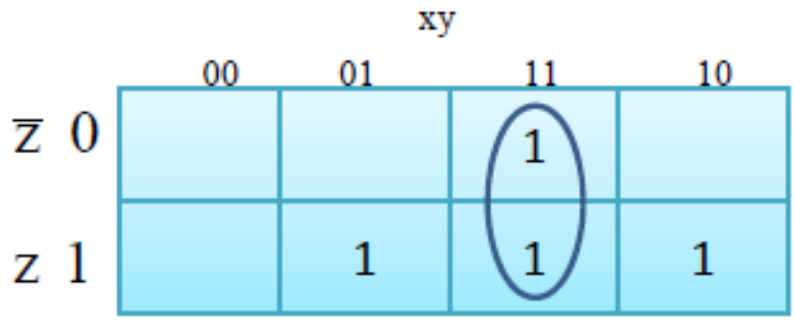
Table de vérité

x	y	z	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

- ***Synthèse du circuit additionneur complet.***



Tab. K. de s



Tab. K. de r

x	y	z	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

$$s = \bar{x}.\bar{y}.z + \bar{x}.y.\bar{z} + x.y.z + x.\bar{y}.\bar{z} = \bar{z}.(x.\bar{y} + \bar{x}y) + z.(\bar{x}.\bar{y} + x.y)$$

$$s = \bar{z}.(x \oplus y) + z.\overline{(x \oplus y)} = (x \oplus y) \oplus z$$

$$r = x.y + y.z + x.z$$

on peut également l'écrire :

$$r = x.y + x.\bar{y}.z + \bar{x}.y.z = x.y + z.(x \oplus y)$$

- ***Synthèse du circuit additionneur complet.***

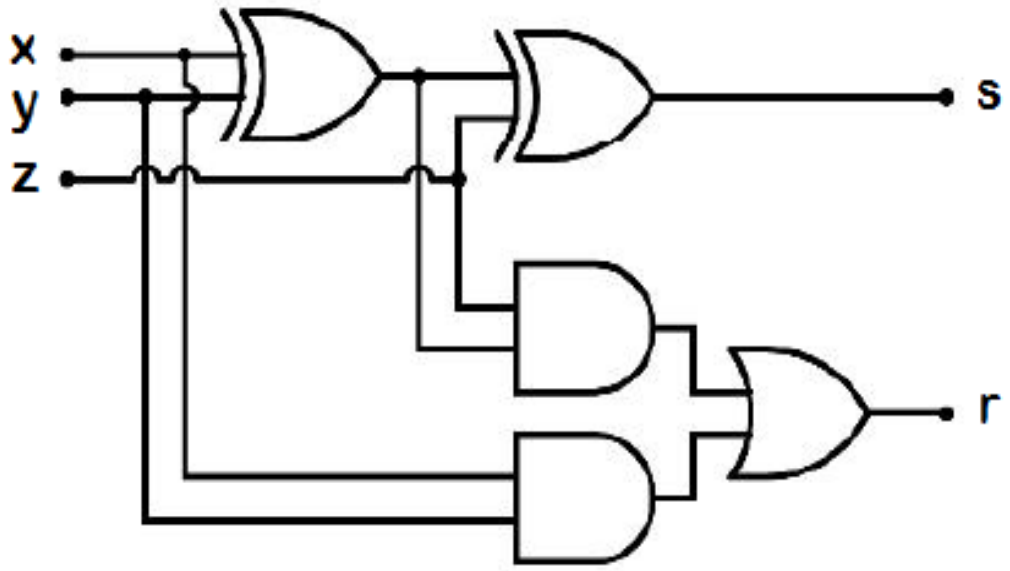
$$s = \bar{x}.\bar{y}.z + \bar{x}.y.\bar{z} + x.y.z + x.\bar{y}.\bar{z} = \bar{z}.(x.\bar{y} + \bar{x}y) + z.(\bar{x}.\bar{y} + x.y)$$

$$s = \bar{z}.(x \oplus y) + z.\overline{(x \oplus y)} = (x \oplus y) \oplus z$$

$$r = x.y + y.z + x.z$$

on peut également l'écrire :

$$r = x.y + x.\bar{y}.z + \bar{x}.y.z = x.y + z.(x \oplus y)$$



x	y	z	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

- ***Synthèse d'un circuit à base d'autres circuits:***

L'additionneur à base de demi-additionneur(s).

les équations de l'additionneur complet, sont reprises ci - après :

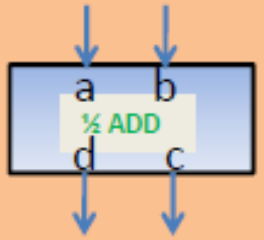
$$s = (x \oplus y) \oplus z$$

$$r = x.y + z.(x \oplus y)$$

Le demi additionneur:

La somme $c = a \oplus b$

La retenue $d = a.b$



On reprend $s = (x \oplus y) \oplus z$ et $r = x.y + z.(x \oplus y)$

Soit $c1 = (x \oplus y)$ et $c2 = c1 \oplus z$

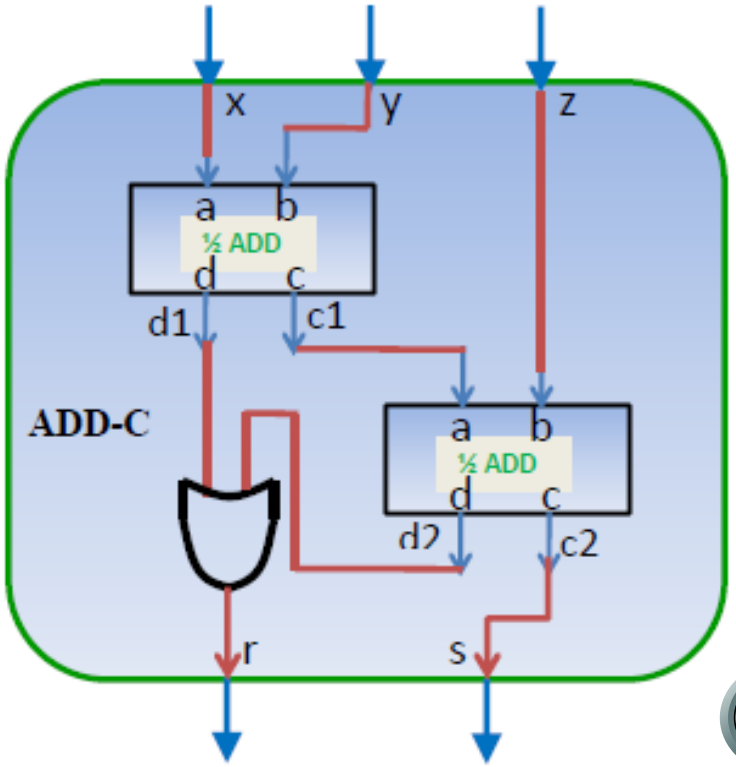
$s = c2$

donc s peut être élaboré grâce aux sorties c de deux (½ ADD)s.

Soit $d1 = xy$ et $d2 = z.c1$;

on peut réécrire $r = d1 + d2$.

donc r est un OU Logique entre les sorties d de deux (½ ADD)s.



3.4 Additionneur sur 4 bits

- Un additionneur sur 4 bits est un circuit qui permet de faire l'addition de deux nombres A et B de 4 bits chacun
  - $A(a_3a_2a_1a_0)$
  - $B(b_3b_2b_1b_0)$

  En plus il tient en compte de la retenu entrante

- En sortie on va avoir le résultat sur 4 bits ainsi que la retenu (5 bits en sortie)

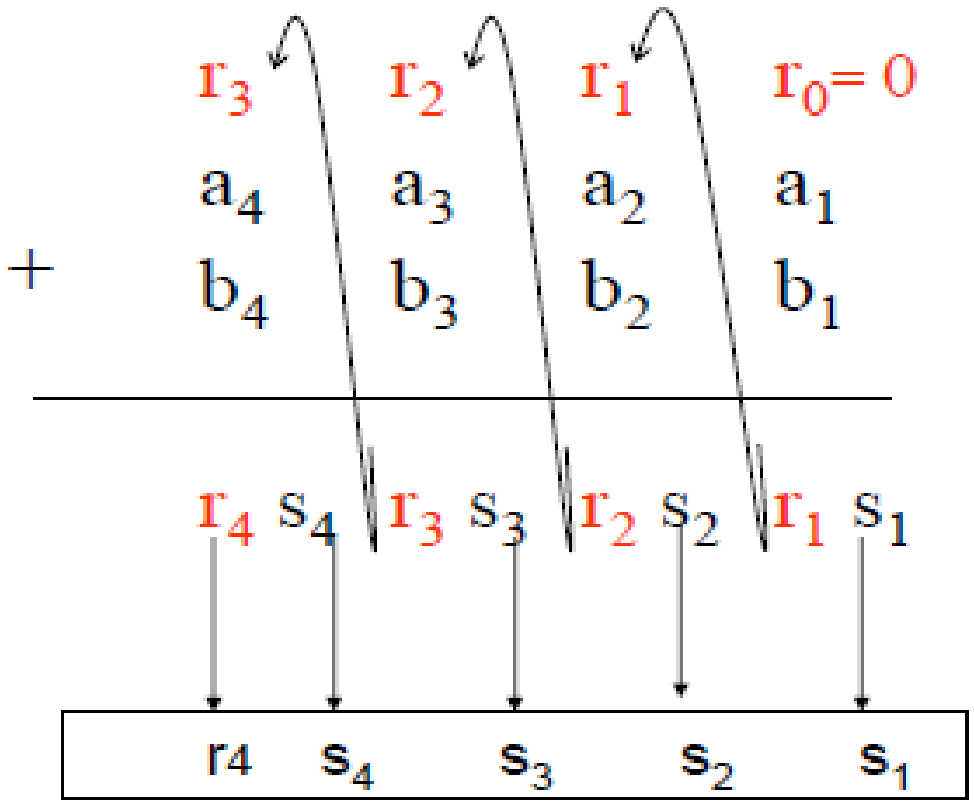
- Donc au total le circuit possède 9 entrées et 5 sorties.

- Avec 9 entrées on a 2^9=512 combinaisons !!!!!! Comment faire pour représenter la table de vérité ?????

- Il faut trouver une solution plus facile et plus efficace pour concevoir ce circuit ?

•Lorsque on fait l'addition en binaire , on additionne bit par bit en commençant à partir du poids fiable et à chaque fois on propage la retenue sortante au bit du rang supérieur.

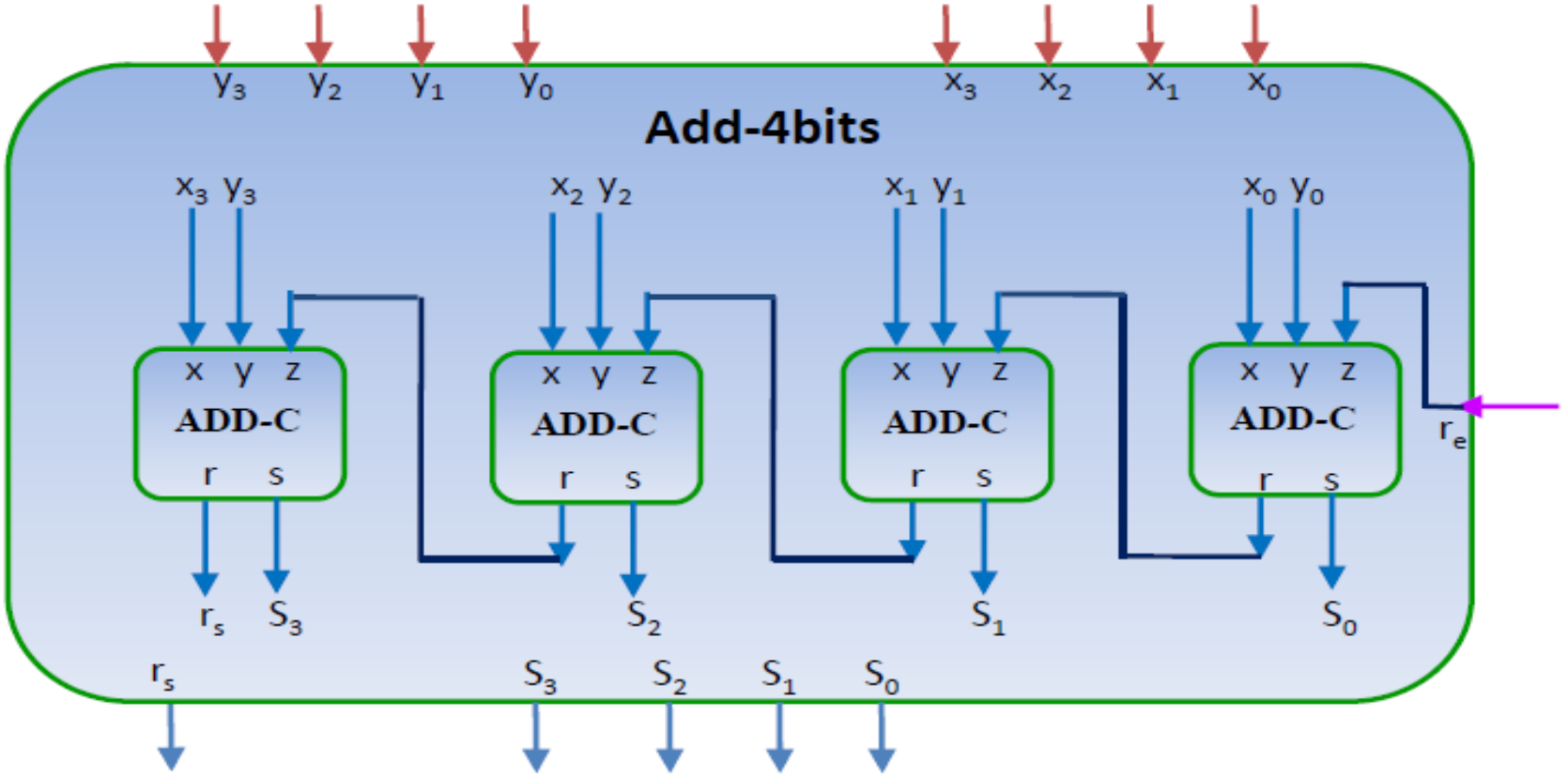
L'addition sur un bit peut se faire par un additionneur complet sur 1 bits.



Résultat final

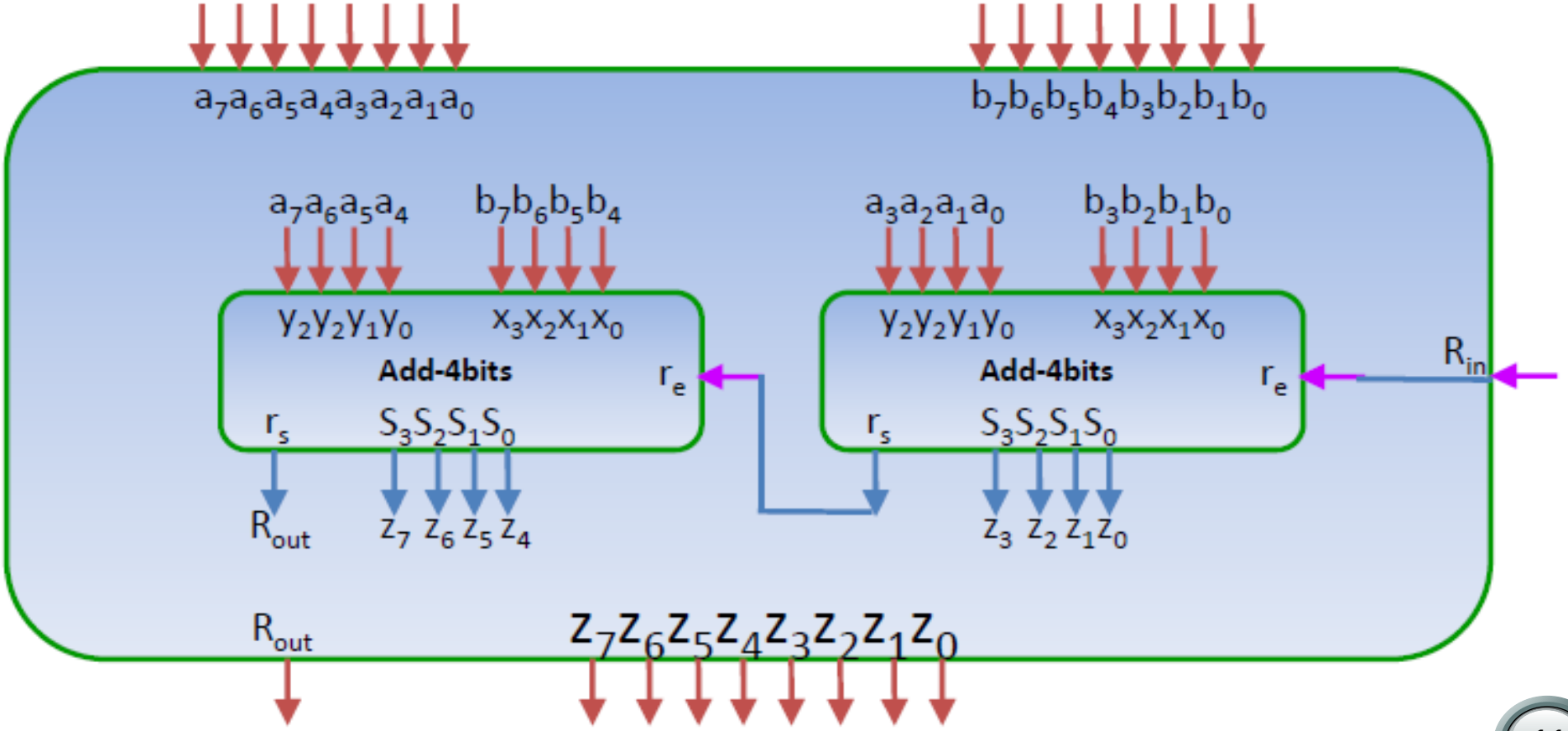
additionneur binaire pur 4 bits.

	r_2	r_1	r_0	r_e
+	x_3	x_2	x_1	x_0
+	y_3	y_2	y_1	y_0
r_s	S_3	S_2	S_1	S_0



- ***Exemple d'utilisation de circuit additionneur binaire pur 4 bits ADD-C.*** Soit à réaliser un circuit additionneur de deux nombre entiers naturels A et B codés chacun sur 8 bits, et produisant Z. **Z = A+B**

Schéma bloc, entrées et sorties sont évidents.



- ***Les comparateurs.*** *(de valeurs absolues ou d'entiers naturels).*

Comparateur de deux nombres A et B codés chacun sur un bit.

Schéma bloc

Entrées A et B.

Trois sorties exclusives

- fe : égalité (A=B)
- fi : inférieur (A < B)
- fs : supérieur (A > B)

Ceci est une façon de représenter le résultat de la comparaison.

A	B	fs	fe	fi
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0

$$fs = A.\overline{B}$$

$$fi = \overline{A}B$$

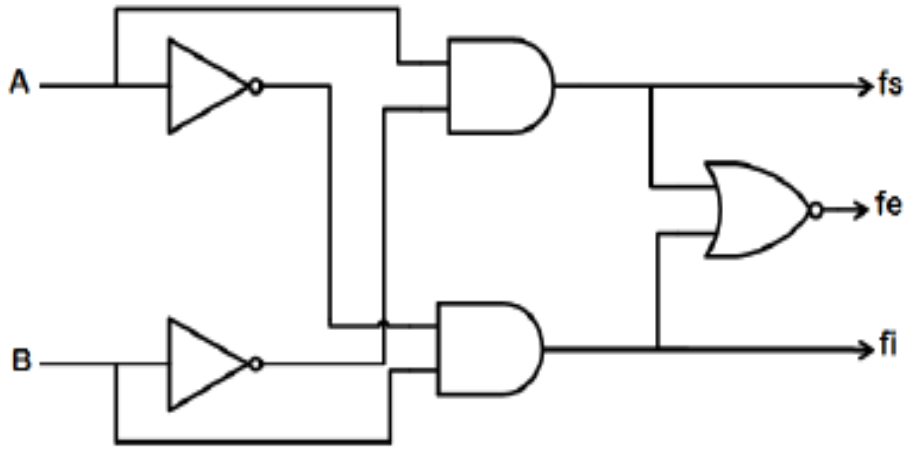
$$fe = \overline{A}\,\overline{B} + AB = \overline{A \oplus B} = \overline{fs + fi}$$

Schéma interne de la boite noire.

$$fs = A.\overline{B}$$

$$fi = \overline{A}.B$$

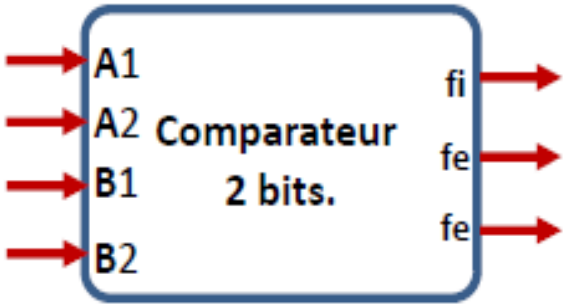
$$fe - \overline{A}.\overline{B} + A.B - \overline{A \oplus B} - \overline{fs + fi}$$



- ***Les comparateurs.*** *(de valeurs absolues ou d'entiers naturels).*

Comparateur de deux nombres A et B codés chacun sur deux bits.
Schéma bloc et table de vérité, les équations, sans les schémas.

Exercice : Réalisez ce comparateur à base du précédent.



A2	A1	B2	B1	fe	fs	fi
0	0	0	0	1	0	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	0	1	0
0	1	0	1	1	0	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	0	1	0
1	0	0	1	0	1	0
1	0	1	0	1	0	0
1	0	1	1	0	0	1
1	1	0	0	0	1	0
1	1	0	1	0	1	0
1	1	1	0	0	1	0
1	1	1	1	1	0	0

1. A=B si

A2=B2 et A1=B1

$$fe = (\overline{A2 \oplus B2}).(\overline{A1 \oplus B1})$$

2. A>B si

A2 > B2 ou (A2=B2 et A1>B1)

$$fs = A2.\overline{B2} + (\overline{A2 \oplus B2}).(A1.\overline{B1})$$

3. A<B si

A2 < B2 ou (A2=B2 et A1<B1)

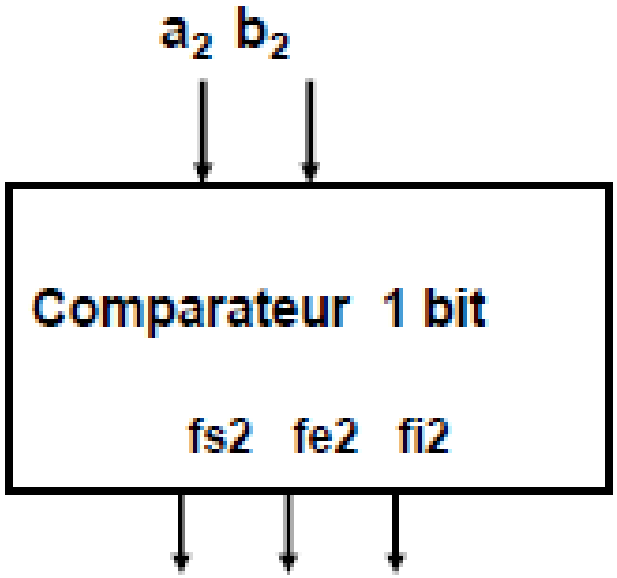
$$fi = \overline{A2}.B2 + (\overline{A2 \oplus B2}).(\overline{A1}.B1)$$

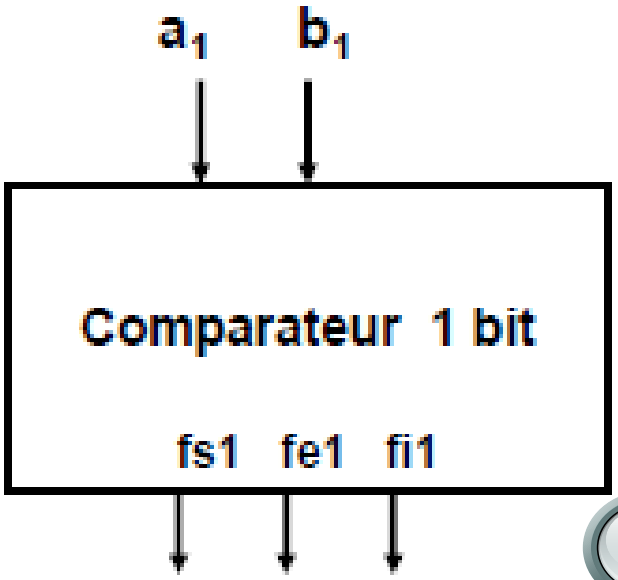
comparateur 2 bits avec des comparateurs 1 bit

•C'est possible de réaliser un comparateur 2 bits en utilisant des comparateurs 1 bit et des portes logiques.

•Il faut utiliser un comparateur pour comparer les bits du poids faible et un autre pour comparer les bits du poids fort.

•Il faut combiner entre les sorties des deux comparateurs utilisés pour réaliser les sorties du comparateur final.





1. A=B si

A2=B2 et A1=B1

$$fe = \overline{(A2 \oplus B2)}.\overline{(A1 \oplus B1)} = fe2.fe1$$

2. A>B si

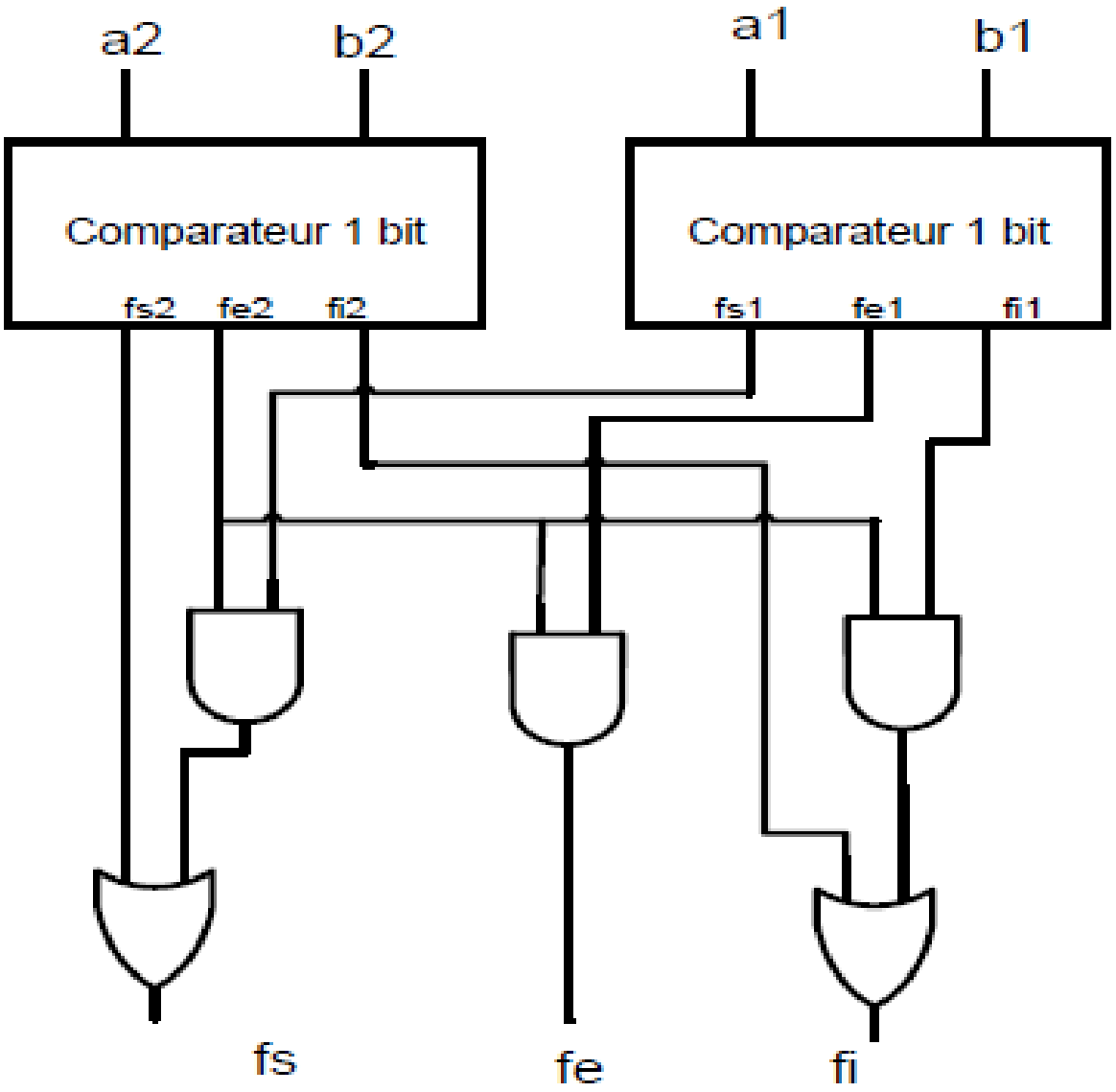
A2 > B2 ou (A2=B2 et A1>B1)

$$fs = A2.\overline{B2} + \overline{(A2 \oplus B2)}.(A1.\overline{B1}) = fs2 + fe2.fs1$$

3. A<B si

A2 < B2 ou (A2=B2 et A1<B1)

$$fi = \overline{A2}.B2 + \overline{(A2 \oplus B2)}.(\overline{A1}.B1) = fi2 + fe2.fi1$$


a2
b2
a1
b1
Comparateur 1 bit
Comparateur 1 bit
fs2
fe2
fi2
fs1
fe1
fi1
fs
fe
fi

Comparateur avec des entrées de mise en cascade

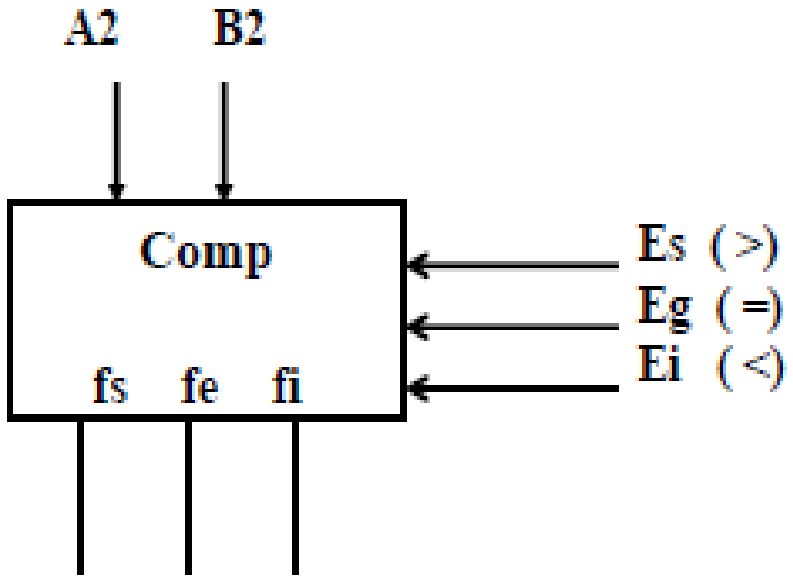
- On remarque que :
  - Si A2 >B2 alors A > B
  - Si A2<B2 alors A < B

- Par contre si A2=B2 alors il faut tenir en compte du résultat de la comparaison des bits du poids faible.

- Pour cela on rajoute au comparateur des entrées qui nous indiquent le résultat de la comparaison précédente.

- Ces entrées sont appelées des entrées de mise en cascade.

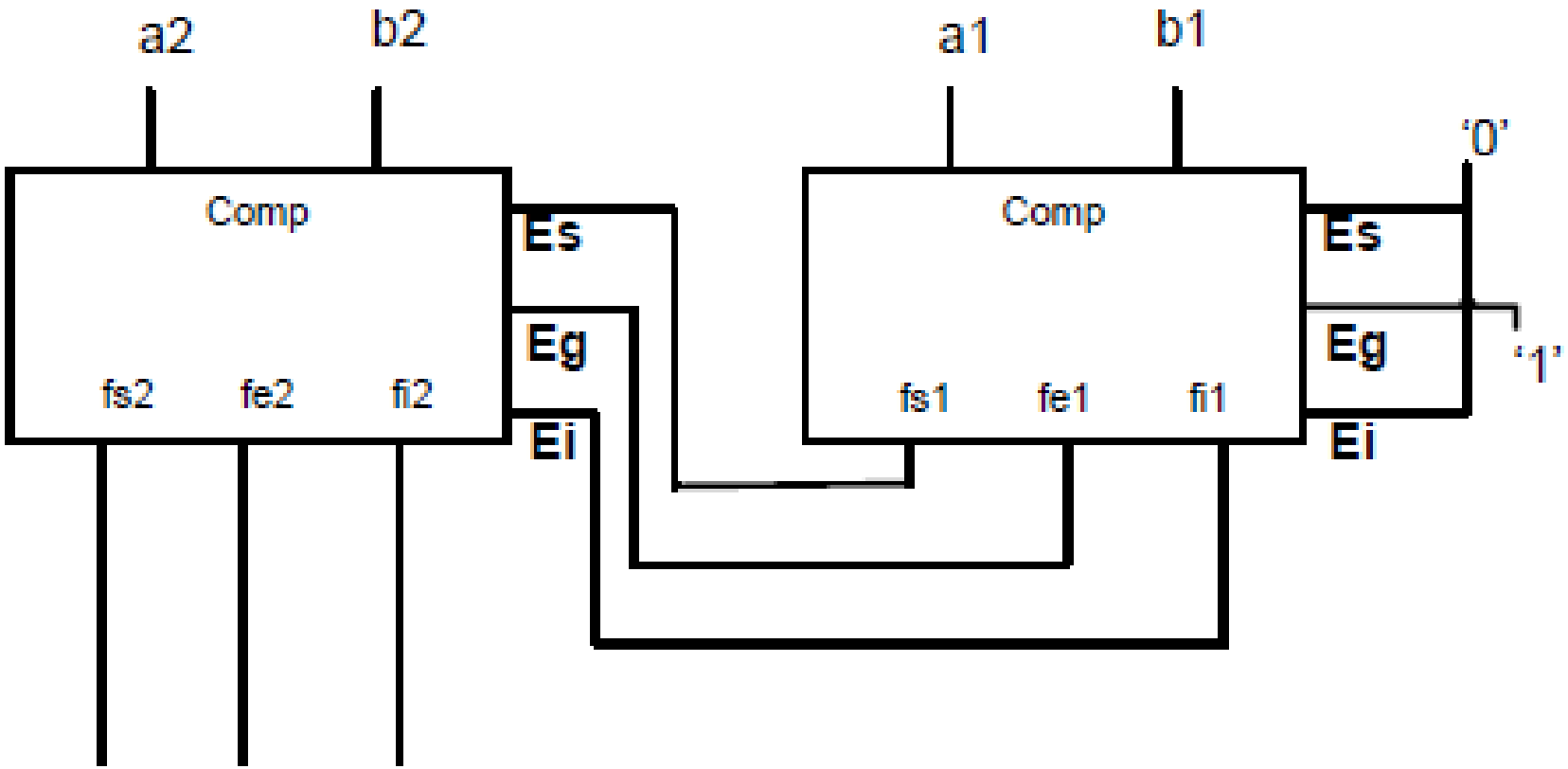
A2	B2	Es	Eg	Ei		fs	fe	fs
A2>B2		X	X	X		1	0	0
A2<B2		X	X	X		0	0	1
A2=B1		1	0	0		1	0	0
		0	1	0		0	1	0
		0	0	1		0	0	1



fs= (A2>B2) ou (A2=B2).Es

fi= (A2<B2) ou (A2=B2).Ei

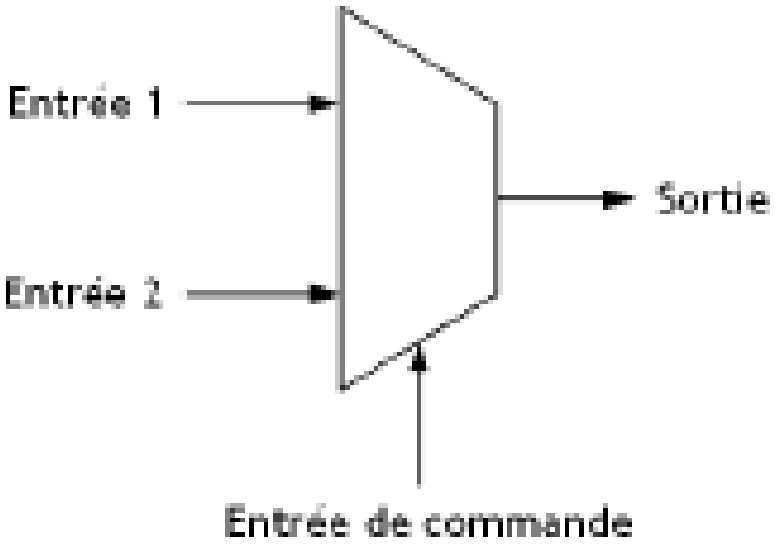
fe=(A2=B2).Eg


a2
b2
a1
b1
'0'
Comp
Comp
Es
Es
Eg
Eg
'1'
fs2
fe2
fi2
fs1
fe1
fi1
Ei
Ei

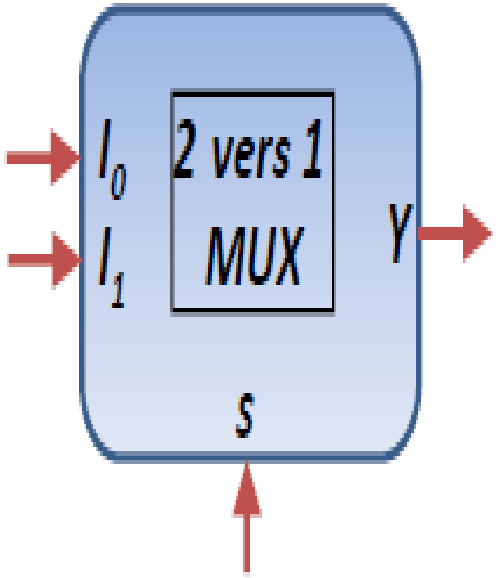
5. Multiplexeurs (MUX)

Un multiplexeur (MUX) est généralement un organe constitué d'un ou plusieurs circuits qui reçoit N entrées (E) et transmet par sa sortie (S) une de ses entrées au choix. Pour sélectionner cette entrée le multiplexeur reçoit une adresse codée (A). On pourra de plus trouver une ou plusieurs entrées de validation (V).

L'entrée de validation V: elle permet d'autoriser ou non le fonctionnement du multiplexeur.

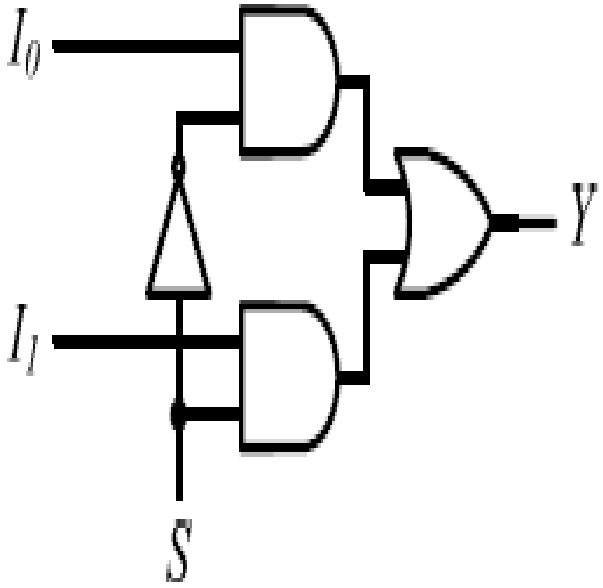


Le multiplexeur 2 vers 1.



S	Y
0	I_0
1	I_1

$$Y = \overline{S}.I_0 + S.I_1$$

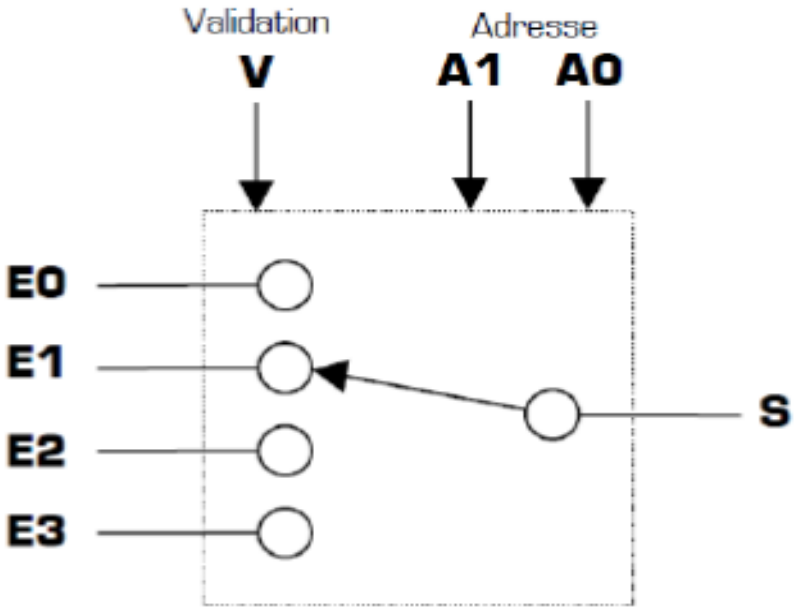


Le multiplexeur 4 vers 1

Multiplexeur à 4 entrées, aussi appelé multiplexeur **4 vers 1**.

Le schéma ci-dessous donne une image d'un multiplexeur

- 4 entrées de donnée **E0**, **E1**, **E2** et **E3**
- 2 entrées d'adresse **A0** et **A1**
- 1 sortie **S**
- 1 entrée de validation **V**

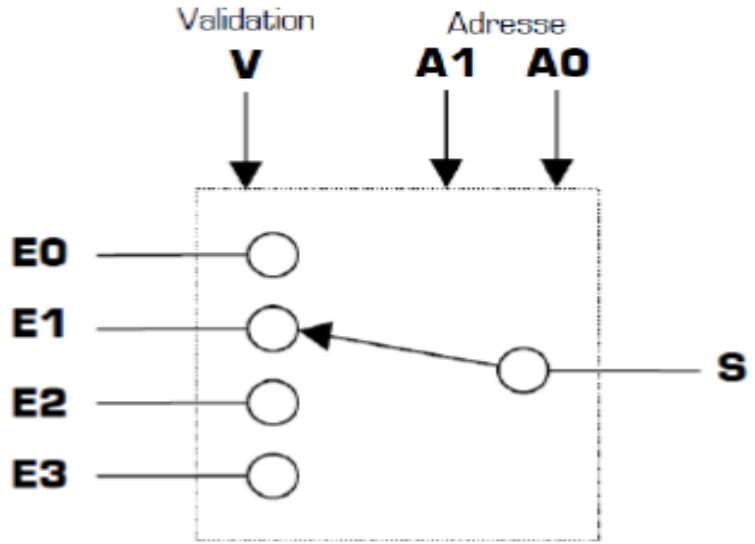


Le multiplexeur 4 vers 1

Fonctionnement

Le multiplexage consiste à envoyer sur une même ligne de transmission des informations provenant de sources différentes.

- S=E0 si A0=0 et A1=0 et V=1
- S=E1 si A0=1 et A1=0 et V=1
- S=E2 si A0=0 et A1=1 et V=1
- S=E3 si A0=1 et A1=1 et V=1



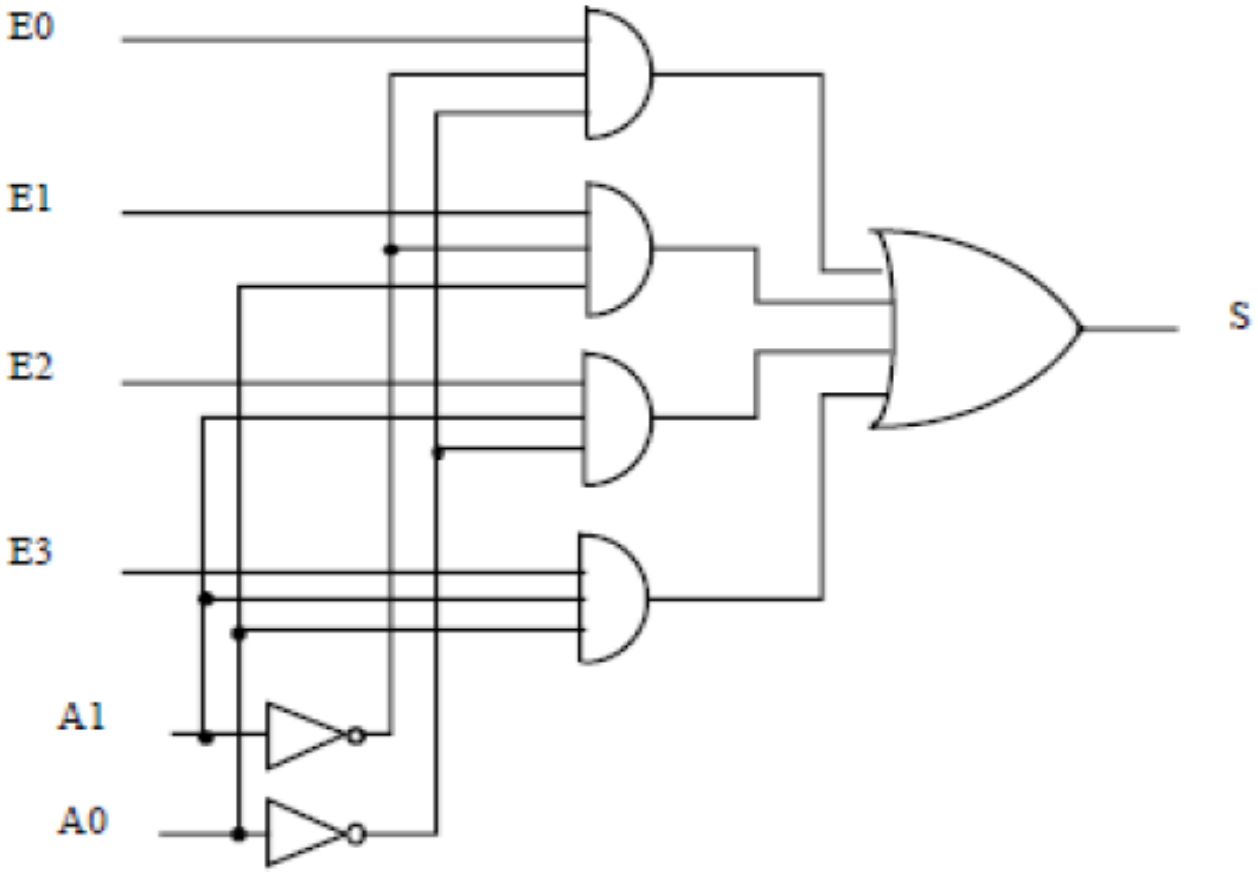
Le fonctionnement de cette fonction multiplexeur peut être résumé dans la table suivante :

	V	A1	A0	S
	1	0	0	E0
	1	0	1	E1
	1	1	0	E2
	1	1	1	E3
MUX non validé	0	X	X	0

L'équation qui donne la sortie en fonction des entrées se déduit directement du tableau

$$S = (\overline{A1} \cdot \overline{A0} \cdot E0) + (\overline{A1} \cdot A0 \cdot E1) + (A1 \cdot \overline{A0} \cdot E2) + (A1 \cdot A0 \cdot E3)$$

Ce qui aboutit au logigramme ci-dessous :



$$S = (\overline{A1} \cdot \overline{A0} \cdot E0) + (\overline{A1} \cdot A0 \cdot E1) + (A1 \cdot \overline{A0} \cdot E2) + (A1 \cdot A0 \cdot E3)$$

On remarquera sur ce logigramme que les portes ET fonctionnent comme des commutateurs dont la validation est assurée par les entrées d'adresse A1 et A0.

Décodage des adresses.

Adresse	
A1	A0
0	0
0	1
1	0
1	1

Sélection			
s3	s2	s1	s0
0	0	0	1
0	0	1	0
0	1	0	0
1	0	0	0

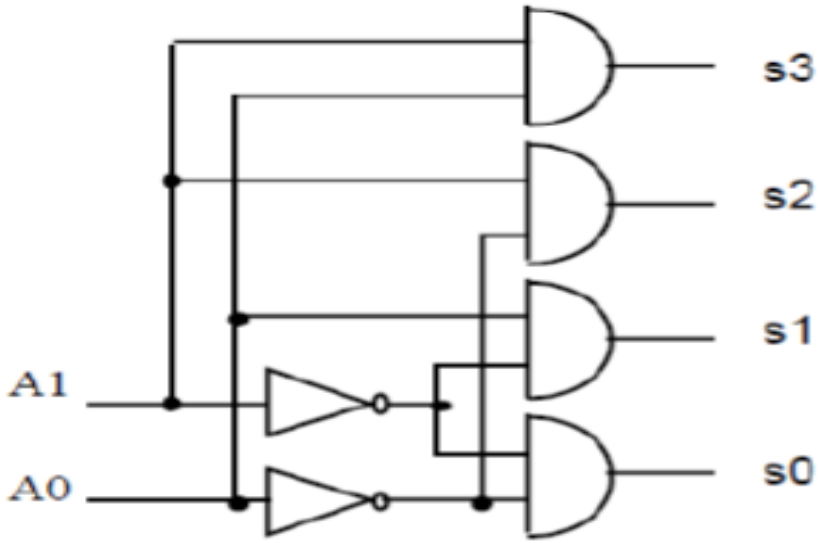
Equations déduites:

$$S0 = \overline{A1} \cdot \overline{A0}$$

$$S1 = \overline{A1} \cdot A0$$

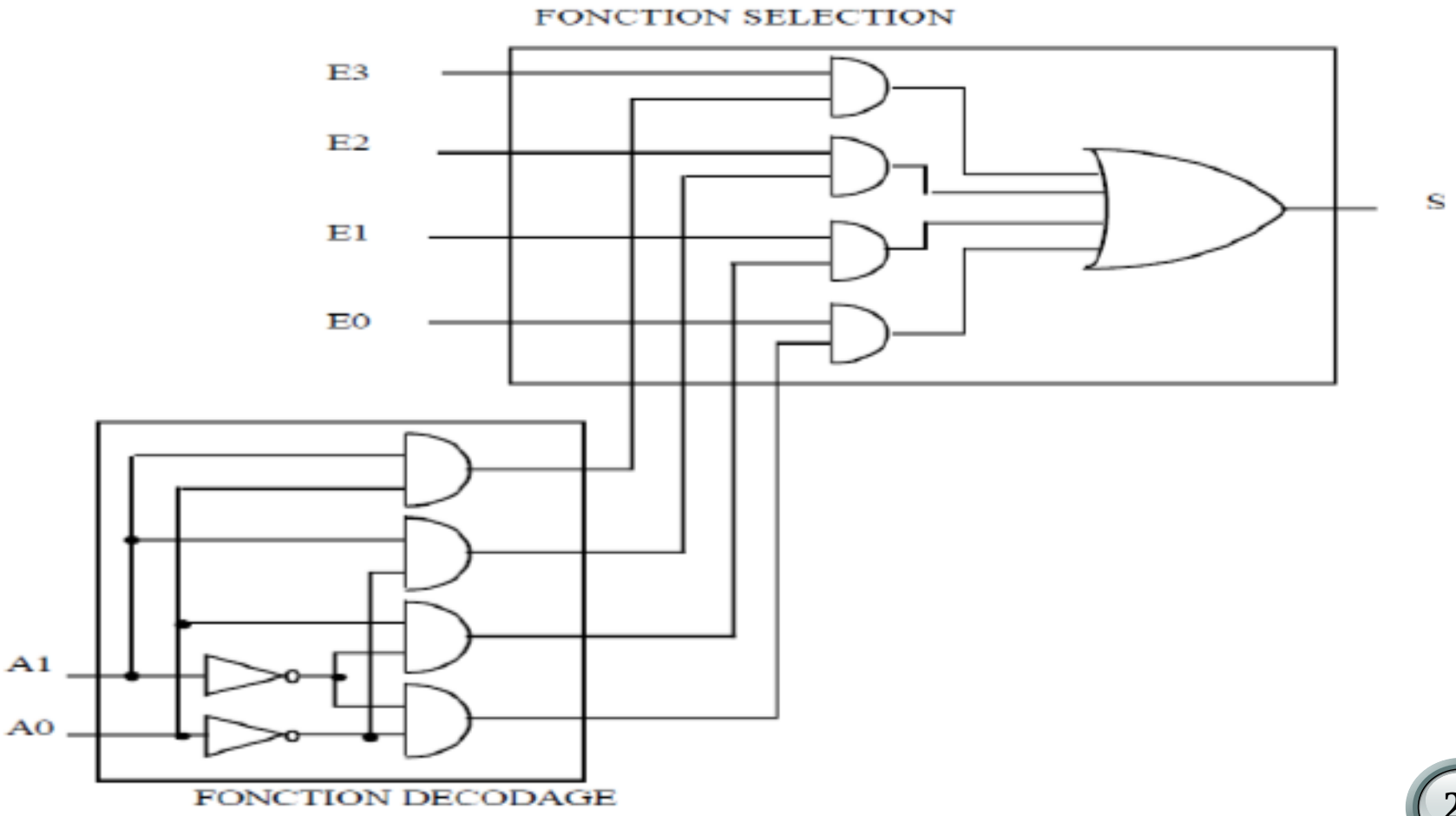
$$S2 = A1 \cdot \overline{A0}$$

$$S3 = A1 \cdot A0$$



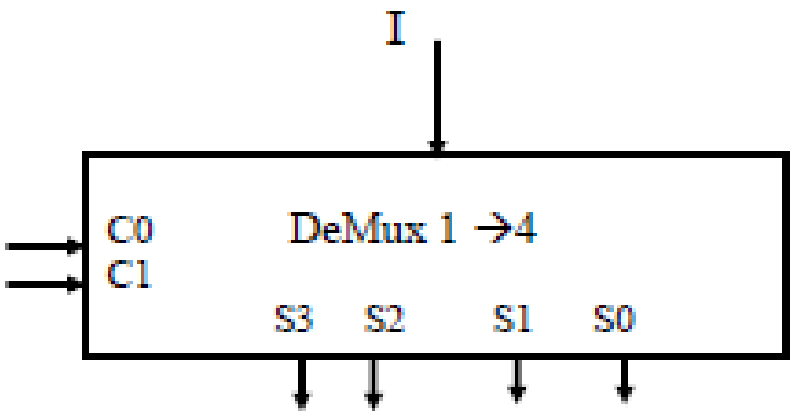
Fonction Décodage

Le schéma ci-dessous fait apparaître le multiplexeur réalisé à l'aide de deux sous-ensembles, d'une part le décodeur d'adresses et d'autre part le sélecteur d'entrée.



6. Démultiplexeurs (DMUX)

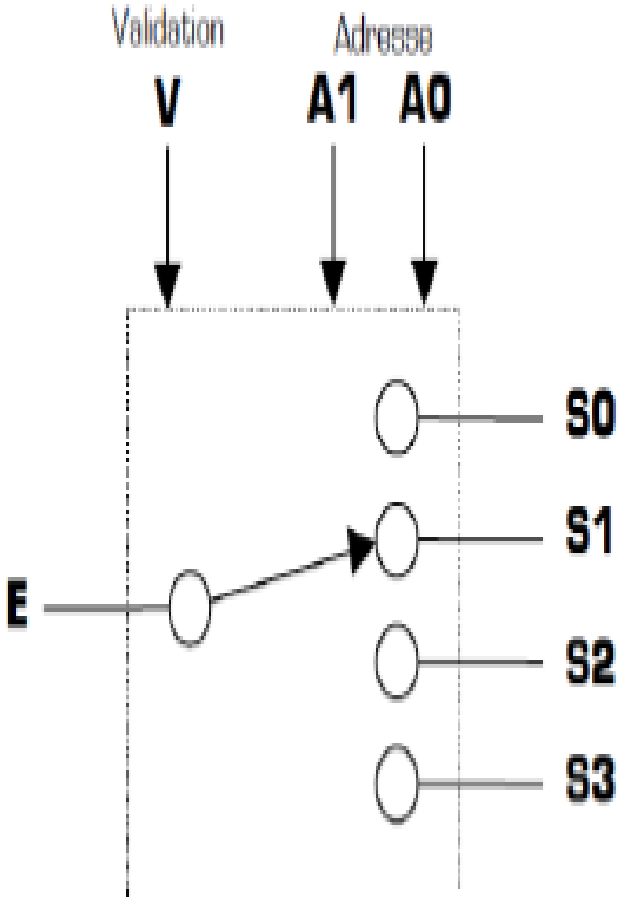
- Il joue le rôle inverse d'un multiplexeurs, il permet de faire passer une information dans l'une des sorties selon les valeurs des entrées de commandes.
- Il possède :
  - une seule entrée
  - 2^n sorties
  - N entrées de sélection (commandes)



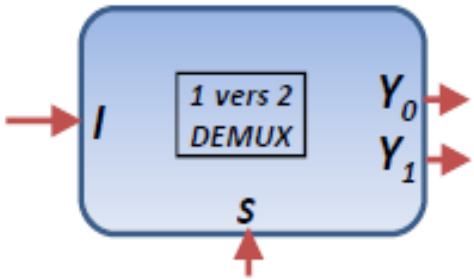
Démultiplexeur à 4 sorties, aussi appelé démultiplexeur **1 vers 4**.

Le schéma ci-dessous donne une image d'un démultiplexeur

- 1 entrée de donnée **E**
- 4 sorties **S0**, **S1**, **S2** et **S3**
- 2 entrées d'adresse **A0** et **A1**
- 1 entrée de validation **V**



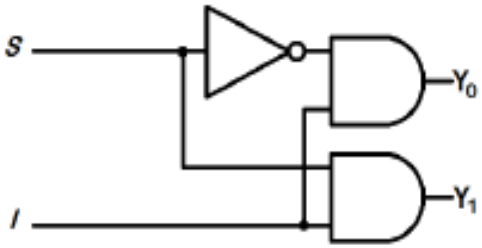
Demux (1 vers 2) & (1 vers 4)

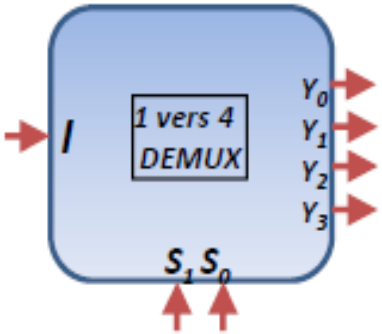


I	S	Y_1	Y_0
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	0

$$Y_1 = S.I$$

$$Y_0 = \overline{S}.I$$





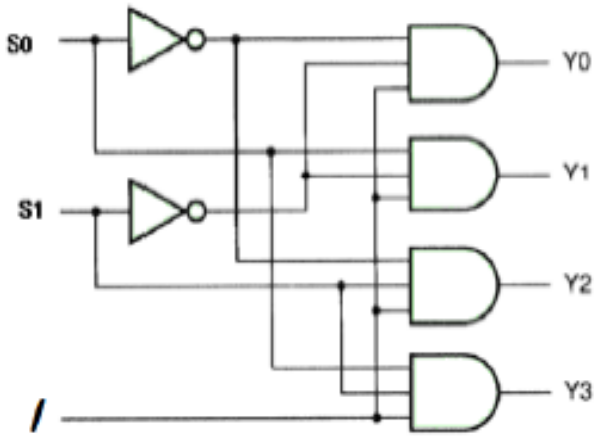
S_1	S_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	/
0	1	0	0	/	0
1	0	0	/	0	0
1	1	/	0	0	0

$$Y_0 = \overline{S_1}.\overline{S_0}.I$$

$$Y_1 = \overline{S_1}.S_0.I$$

$$Y_2 = S_1.\overline{S_0}.I$$

$$Y_3 = S_1.S_0.I$$



Le démultiplexage consiste à répartir sur plusieurs lignes des informations qui arrivent en série sur une même ligne.

- S0=E si A0=0 et A1=0 et V=1
- S1=E si A0=1 et A1=0 et V=1
- S2=E si A0=0 et A1=1 et V=1
- S3=E si A0=1 et A1=1 et V=1

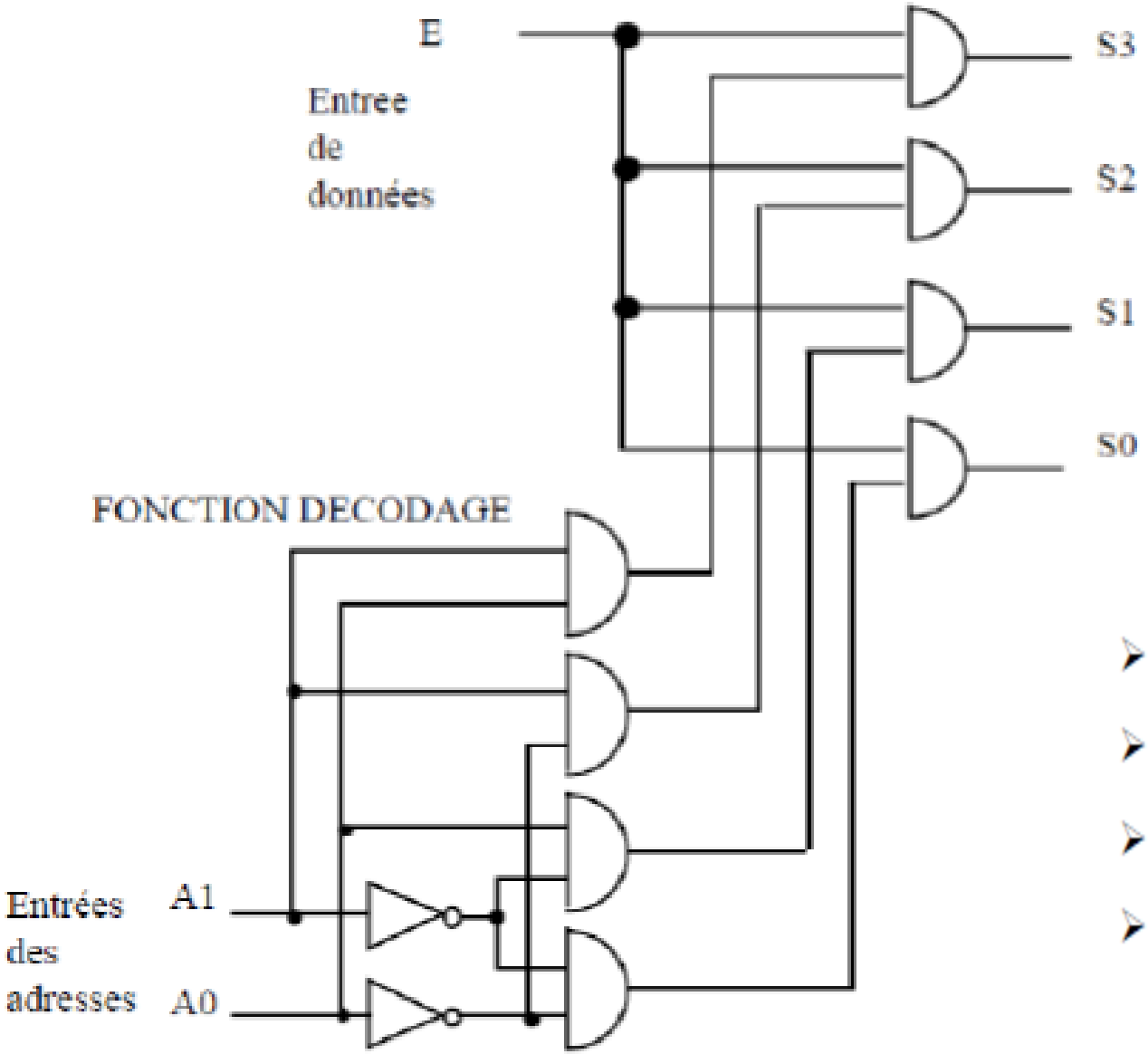
Le fonctionnement de cette fonction multiplexeur peut être résumé dans la table suivante :

	V	A1	A0	S0	S1	S2	S3
	1	0	0	E	0	0	0
	1	0	1	0	E	0	0
	1	1	0	0	0	E	0
	1	1	1	0	0	0	E
DMUX non validé	0	X	X	0	0	0	0

	V	A1	A0	S0	S1	S2	S3
	1	0	0	E	0	0	0
	1	0	1	0	E	0	0
	1	1	0	0	0	E	0
	1	1	1	0	0	0	E
DMUX non validé	0	X	X	0	0	0	0

Les équations qui donnent les sorties en fonction d'entrée se déduit directement du tableau

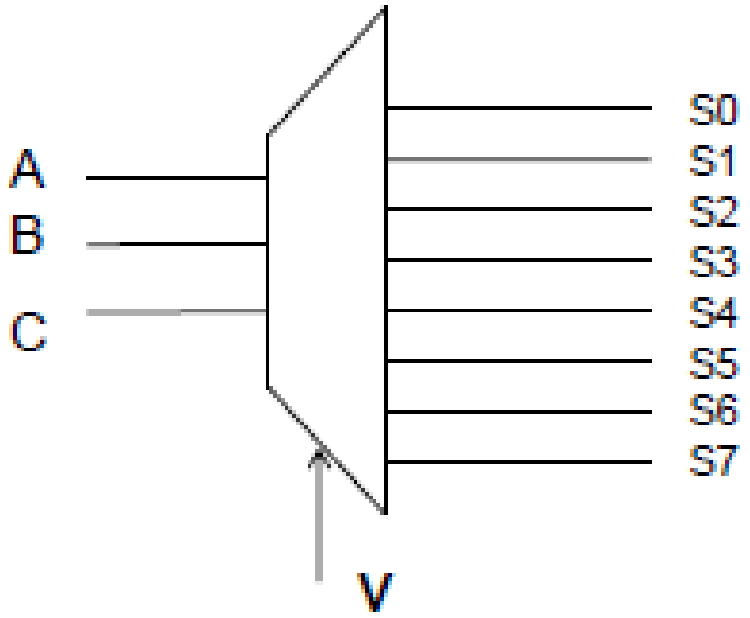
- $S0 = \overline{A1} \cdot \overline{A0} \cdot E$
- $S1 = \overline{A1} \cdot A0 \cdot E$;
- $S2 = A1 \cdot \overline{A0} \cdot E$
- $S3 = A1 \cdot A0 \cdot E$



- $S0 = \overline{A1} \cdot \overline{A0} \cdot E$
- $S1 = \overline{A1} \cdot A0 \cdot E$;
- $S2 = A1 \cdot \overline{A0} \cdot E$
- $S3 = A1 \cdot A0 \cdot E$

7. Décodeurs

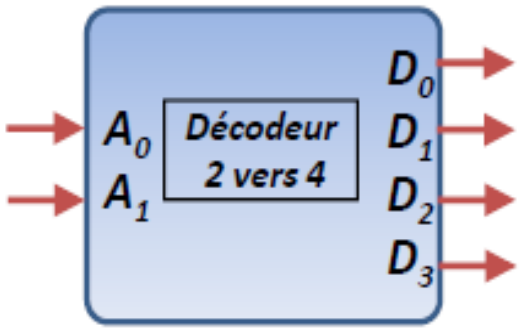
- C'est un circuit combinatoire qui est constitué de :
  - N : entrées de données
  - 2^n sorties
  - Pour chaque combinaison en entrée une seule sortie est active à la fois



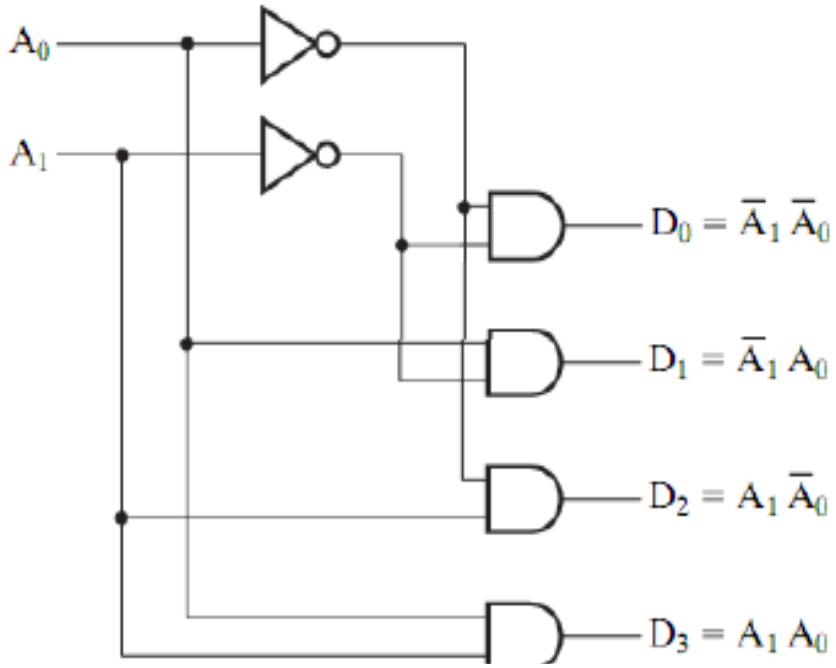
Un décodeur 3→8

Le décodeur 2 vers 4.

Deux entrées A_1, A_0.
Quatre sorties D_0, D_1, D_2, D_3.
Schéma bloc, table de vérité, schéma interne:



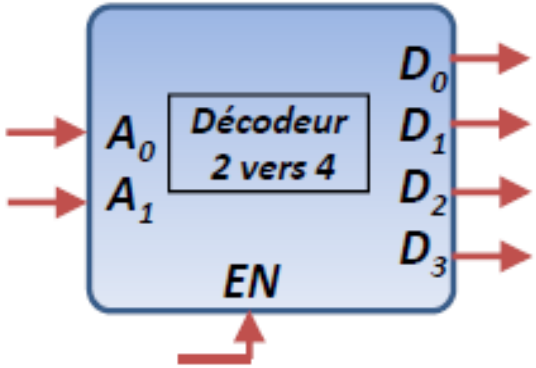
A_1	A_0	D_0	D_1	D_2	D_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1



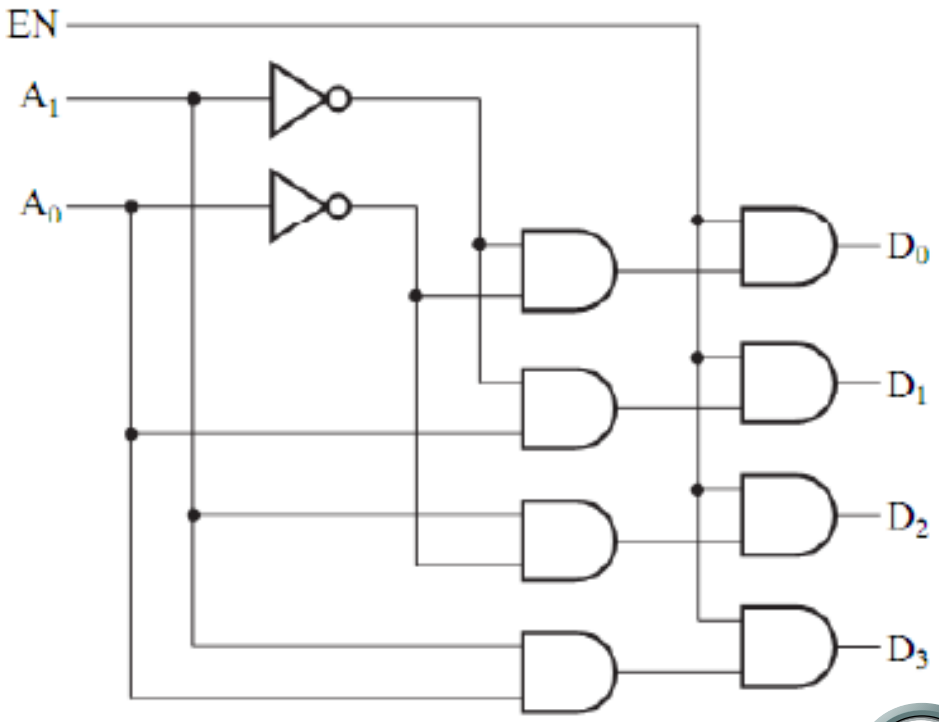
Le décodeur 2 vers 4 avec l'entrée "E: Enable". (V: Validation)

Deux entrées A_1, A_0 et l'entrée *"E: Enable"*.
Quatre sorties D_0, D_1, D_2, D_3.
Schéma bloc, table de vérité, schéma interne:

$$D0 = \overline{A0}\ \overline{A1}\ EN$$
$$D1 = \overline{A0}\ A1\ EN$$
$$D2 = A0\ \overline{A1}\ EN$$
$$D3 = A0\ A1\ EN$$

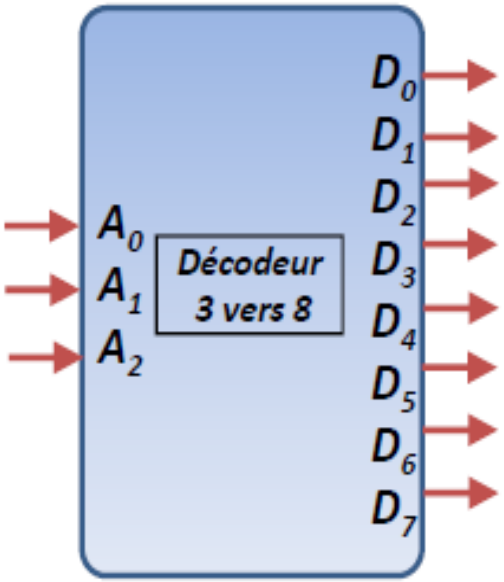


EN	A_1	A_0	D_0	D_1	D_2	D_3
0	X	X	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1



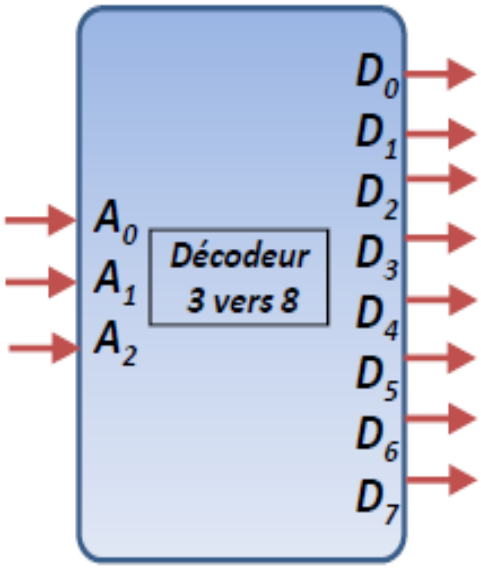
Le décodeur 3 vers 8.

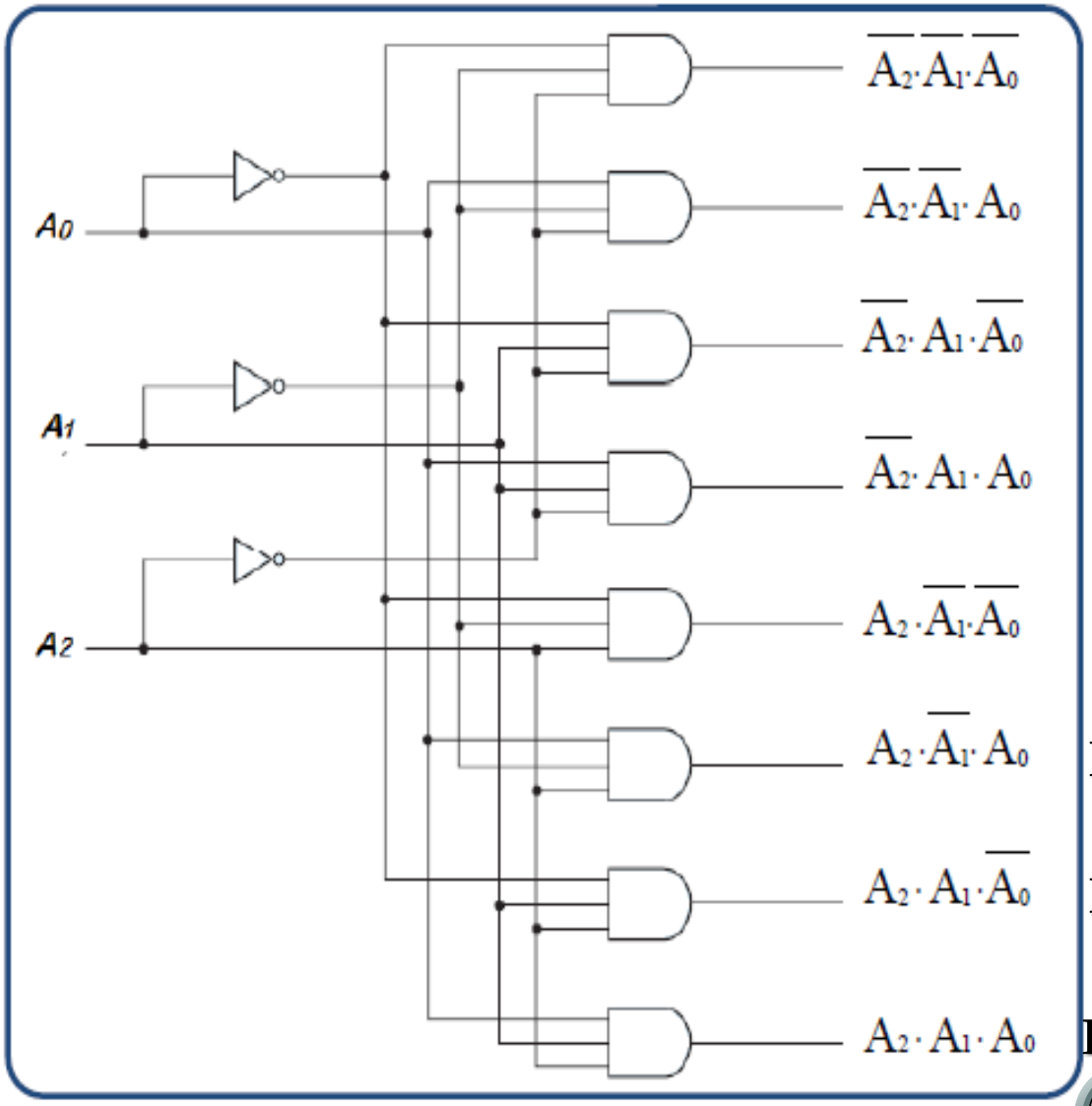
Schéma bloc et table de vérité



A_2	A_1	A_0	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Schéma bloc et schéma interne.



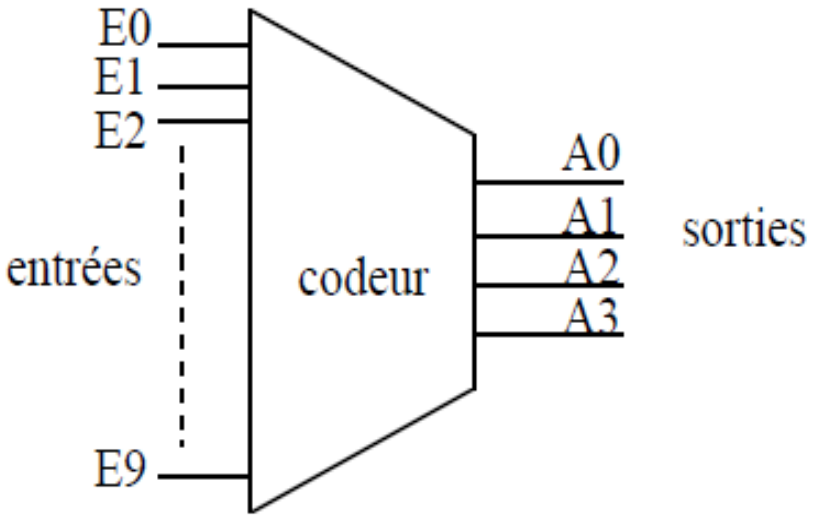


8. Les codeurs

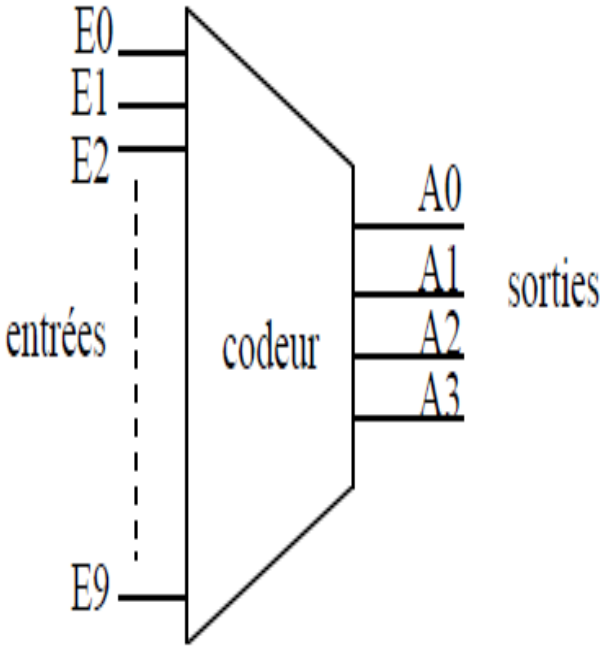
Le principe de fonctionnement d'un codeur est le suivant : lorsqu'une entrée est activée, les sorties affichent la valeur correspondant au numéro de l'entrée dans le code binaire choisi. Un codeur peut être vu comme un convertisseur du code décimal vers un code binaire.

- **Exemple 1 : codeur décimal vers binaire (10 entrées vers 4 sorties)**

Ce codeur reçoit un chiffre décimal sur une des dix entrées et génère l'équivalent binaire sur les sorties $A0$ à $A3$. Une seule entrée doit être active à la fois.



Entrée activée (= 1)	Sorties			
	A3	*A2*	*A1*	*A0*
E0	0	0	0	0
E1	0	0	0	1
E2	0	0	1	0
E3	0	0	1	1
E4	0	1	0	0
E5	0	1	0	1
E6	0	1	1	0
E7	0	1	1	1
E8	1	0	0	0
E9	1	0	0	1



$$A0 = E1 + E3 + E5 + E7 + E9$$

$$A1 = E2 + E3 + E6 + E7$$

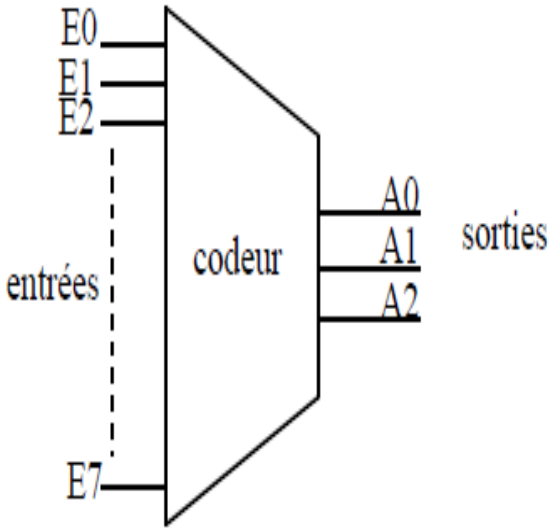
$$A2 = E4 + E5 + E6 + E7$$

$$A3 = E8 + E9$$

- **Exemple 2 : codeur binaire 8 vers 3**

Ce codeur reçoit une information codée sur une de ses huit entrées et génère l'équivalent binaire sur les sorties $A0$ à $A2$. Une seule entrée doit être active à la fois.

	Sorties		
Entrée activée (= 1)	$A2$	$A1$	$A0$
$E0$	0	0	0
$E1$	0	0	1
$E2$	0	1	0
$E3$	0	1	1
$E4$	1	0	0
$E5$	1	0	1
$E6$	1	1	0
$E7$	1	1	1



$$A0 = E1 + E3 + E5 + E7$$

$$A1 = E2 + E3 + E6 + E7$$

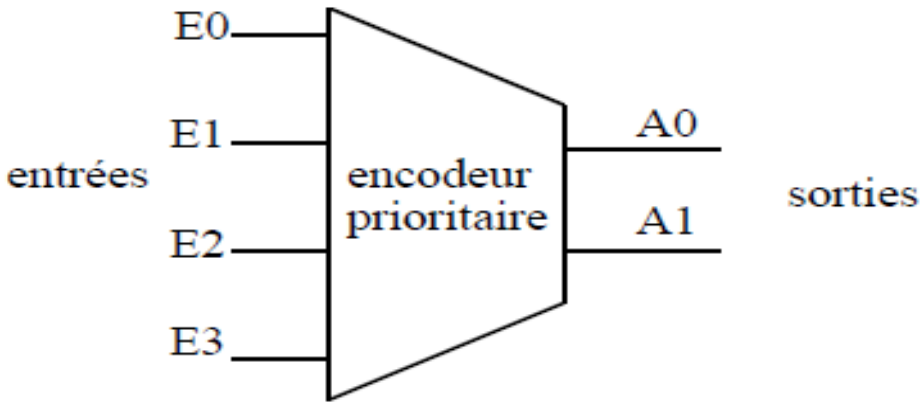
$$A2 = E4 + E5 + E6 + E7$$

8. 1 Codeur prioritaire

Ce type de codeur fixe un ordre de priorité entre les entrées. Dans le cas d'un encodage en binaire pur, le codeur prioritaire donne en général la priorité à l'entrée de poids le plus élevé. Par exemple, si les entrées 2, 8 et 9 sont activées simultanément, le codage de sortie correspondra à l'entrée 9. Ce circuit permet de détecter le rang du premier bit positionné à 1 dans un mot d'entrée.

- **Exemple : encodeur prioritaire 4 vers 2**

L'opérateur de la figure 4.3 est un encodeur prioritaire possédant 4 entrées ; l'entrée $E3$ correspond à l'entrée la plus prioritaire, et l'entrée $E0$ correspond à l'entrée la moins prioritaire

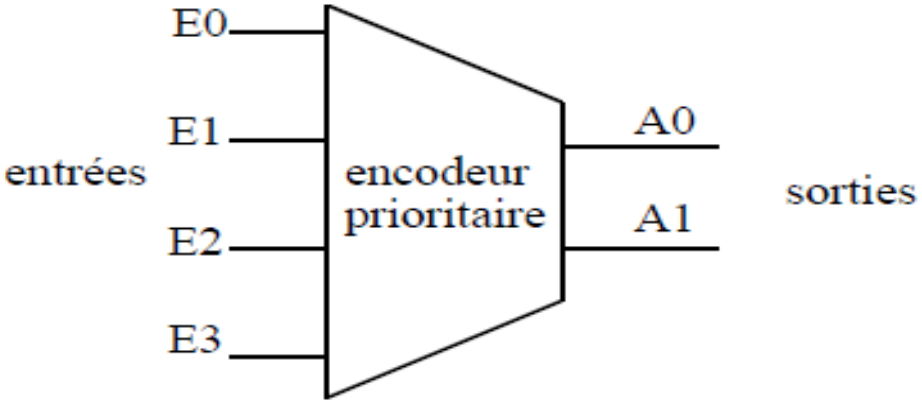


$$A + (A . B) = A$$

$$A . (A + B) = A$$

$$(A + B) . (A + \overline{B}) = A$$

$$A + \overline{A} . B = A + B$$



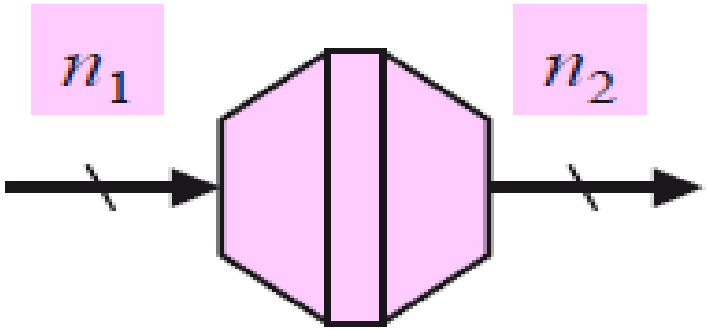
$E3$	$E2$	$E1$	$E0$	$A1$	$A0$
1	X	X	X	1	1
0	1	X	X	1	0
0	0	1	X	0	1
0	0	0	1	0	0

$$A0 = E3 + \overline{E2}\, E1$$

$$A1 = E3 + E2$$

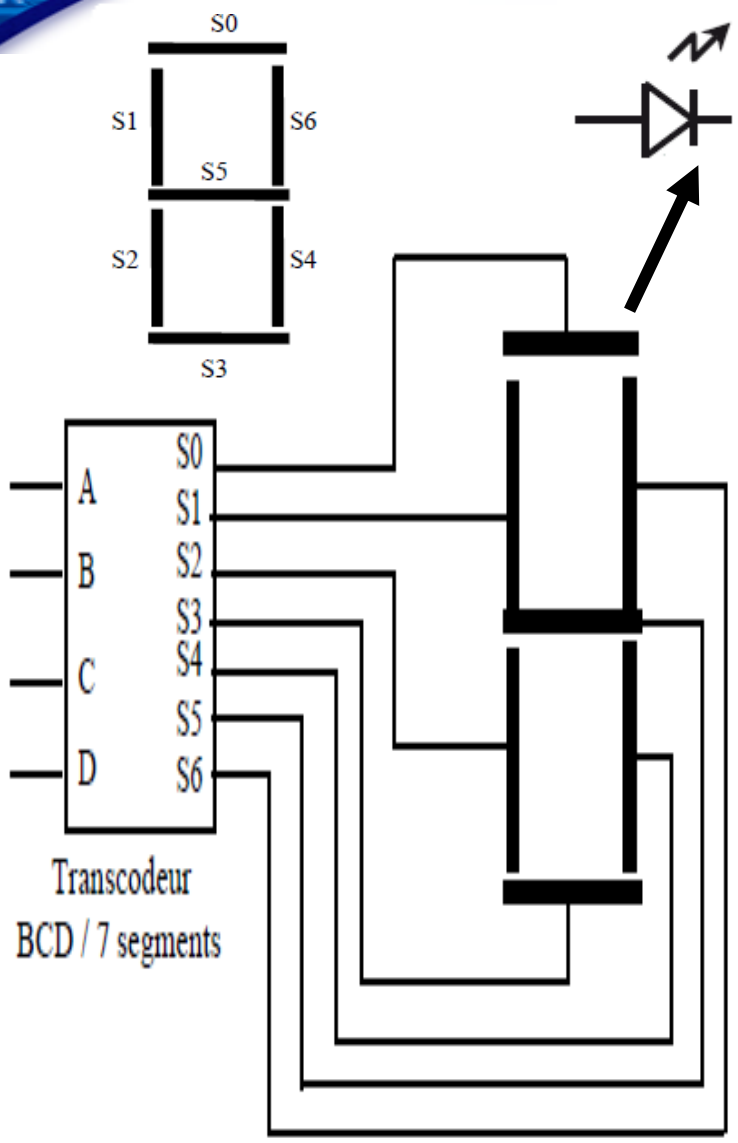
9. Le transcodeur

Ces opérateurs permettent de convertir un nombre écrit dans un code C1 vers un code C2



Exemple 1 : le transcodeur BCD/7 segments

Le transcodeur BCD/7 segments permet de commander un afficheur alphanumérique possédant 7 segments (des diodes électroluminescentes, par exemple). Cet afficheur permet la visualisation des chiffres 0 à 9 codés en binaire naturel sur 4 bits *D*, *C*, *B*, et *A*, où *A* représente le bit de poids le plus faible.



Code BCD				7 segments						
				S6	S5	S4	S3	S2	S1	S0
0	0	0	0	1	0	1	1	1	1	1
0	0	0	1	1	0	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	1	1	1	0	0	1	0
0	1	0	1	0	1	1	1	0	1	1
0	1	1	0	0	1	1	1	1	1	1
0	1	1	1	1	0	1	0	0	0	1
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1

$$S0 = \overline{A}.\overline{C} + A.C + B + D$$

$$S1 = \overline{A}.C + \overline{A}.\overline{B} + \overline{B}.C + D$$

$$S2 = \overline{A}.\overline{C} + \overline{A}.B$$

$$S3 = A.\overline{B}.C + \overline{A}.\overline{C} + \overline{A}.B + B.\overline{C} + D$$

$$S4 = A + \overline{B} + C$$

$$S5 = \overline{B}.C + \overline{A}.B + B.\overline{C} + D$$

$$S6 = A.B + \overline{C}$$