Chapitre 4:Circuits combinatoires

Présenter par:Mme AMGHAR D

Circuits combinatoires

1. Introduction
2. Addition binaire
3. Soustraction
4. Comparaison
5. Multiplexage

Introduction

- Les circuits logiques sont élaborés à partir de composants électroniques – **transistors** .
- Types de circuits logiques:
  - Combinatoires
  - Séquentiels

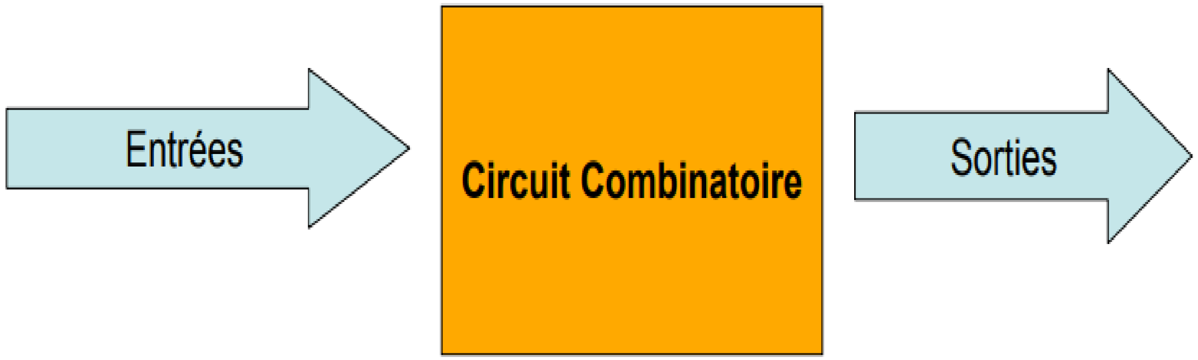
Introduction

- **Circuits logiques séquentiels** : c'est des circuits dont les valeurs de sortie dépendent d'entrée appliquées ultérieurement.

- **Circuits logiques combinatoires** : c'est des circuits dont les valeurs de sortie **ne dépendent que de ses valeurs d'entrée.**

Circuit combinatoire

- Un circuit combinatoire est constitué d'éléments logiques élémentaires appelés portes logiques(logic gates), elle reçoivent des signaux appliqués en entrée et produisent des signaux en sortie.



Circuit combinatoire

L'étude des circuits combinatoires se résume en deux questions :

- **Synthèse** : réaliser le circuit combinatoire à partir de l'énoncé décrivant les fonctions ou le rôle du circuit, en question.
- **Analyse** : déterminer le rôle du circuit combinatoire à partir de son logigramme.

Circuit combinatoire

les étapes à suivre pour réaliser la synthèse d'un circuit logique combinatoire :

1. Établir la table de vérité de chacune des fonctions impliquées dans le problème à traiter

2. Établir les équations logiques.

3. Simplifier les équations de chacune des fonctions logiques.

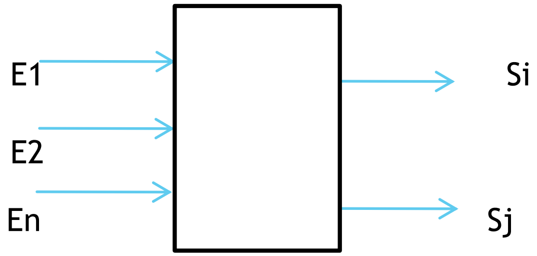
4. Établir le logigramme du circuit logique.

Circuits combinatoires

- Un circuit combinatoire est un circuit numérique dont les sorties dépendent uniquement des entrées.

$S_i = F(E_i)$

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



Circuits combinatoires

Parmi les principaux circuits combinatoires, on distingue:

- les circuits d'opérations arithmétiques (addition, soustraction)
- Les circuits logiques (décodage, multiplexage, comparaison).

Remarque: Il est possible d'utiliser des circuits combinatoires pour réaliser d'autres circuits plus complexes.

Additionneurs(Adder):

- Additionneurs :

– Demi additionneur : 2 entrées sur 1 bit,

2 sorties sur 1 bits.

– Additionneur complet : 3 entrées sur 1 bit,

2 sorties sur 1 bits.

– Additionneur sur n bits.

Additionneurs(Adder):

Demi-additionneur **(half adder):** :

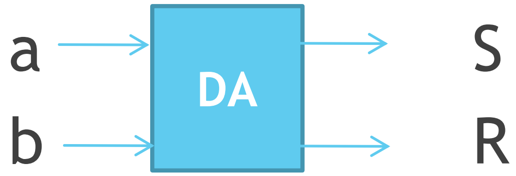
- On commence par l'addition de 2 bits ***a*** et ***b*** en entrée ,avec en sortie la somme ***S*** et une retenue ***R***.
- On l'appelle demi additionneur , parce qu'il ne tient pas compte de la **retenue** qui peut provenir des calculs précédents.

Entrées A B	Sorties Somme (S)	Retenue (R_{sor})
0 + 0	0	0
0 + 1	1	0
1 + 0	1	0
1 + 1	0	1

Additionneurs(Adder):

Demi-additionneur **(half adder): :**

Entrées A B	*Sorties* Somme (S)	Retenue (R_{sor})
0 + 0	0	0
0 + 1	1	0
1 + 0	1	0
1 + 1	0	1



$$S = a \oplus b$$

$$R_{sor} = a.b$$

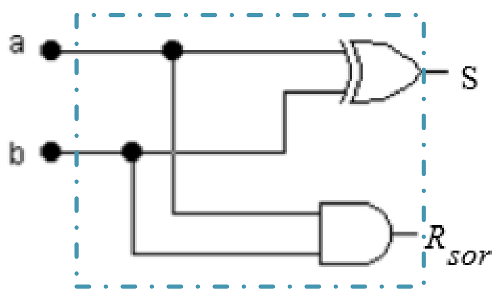
Additionneurs(Adder):

Demi additionneur (half adder):

- Expressions logiques:

$$S = A \oplus B$$
$$R_{sor} = A.B$$

- Logigramme :



Demi-Additionneur

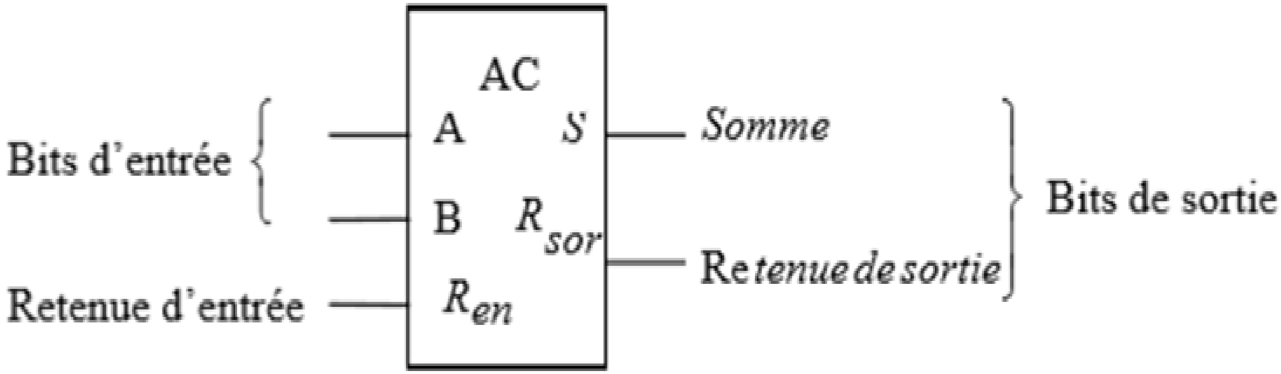
Additionneurs(Adder):

L’additionneur complet **(full adder):**

- il faut tenir compte de **la retenue** provenant des bits de **poids inférieurs**
- donc l’additionneur complet :
- 3 entrées :***a***, ***b*** et ***r***
- 2 sorties ***S*** et ***R***

Additionneurs(Adder):

additionneur Complet (full adder) (AC):



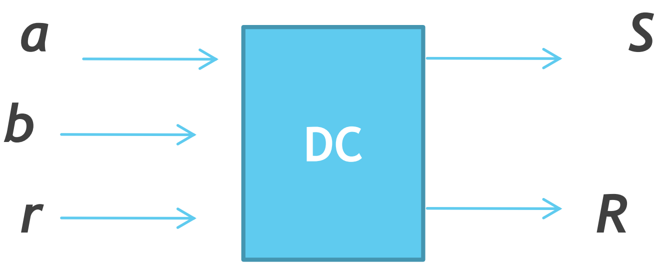
Symbole logique d'un AC

Additionneurs(Adder):

L'additionneur complet(**full adder**):

TV de l'additionneur complet :

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



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

$$\boxed{R = a.b + r(a \oplus b)}$$ (après simplification)

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

$$\boxed{S = (a \oplus b) \oplus r}$$

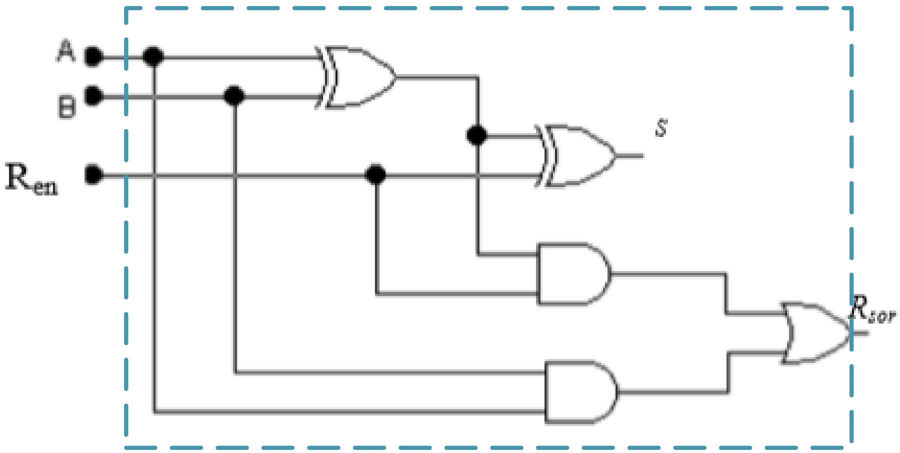
Additionneurs(Adder):

Additionneur complet (full adder): :

Expressions logiques :

$$S = (A \oplus B) \oplus R_{en}$$
$$R_{sor} = A.B + (A \oplus B).R_{en}$$

- Logigramme :



Additionneur complet

Additionneurs(Adder):

Additionneur complet(full adder) :

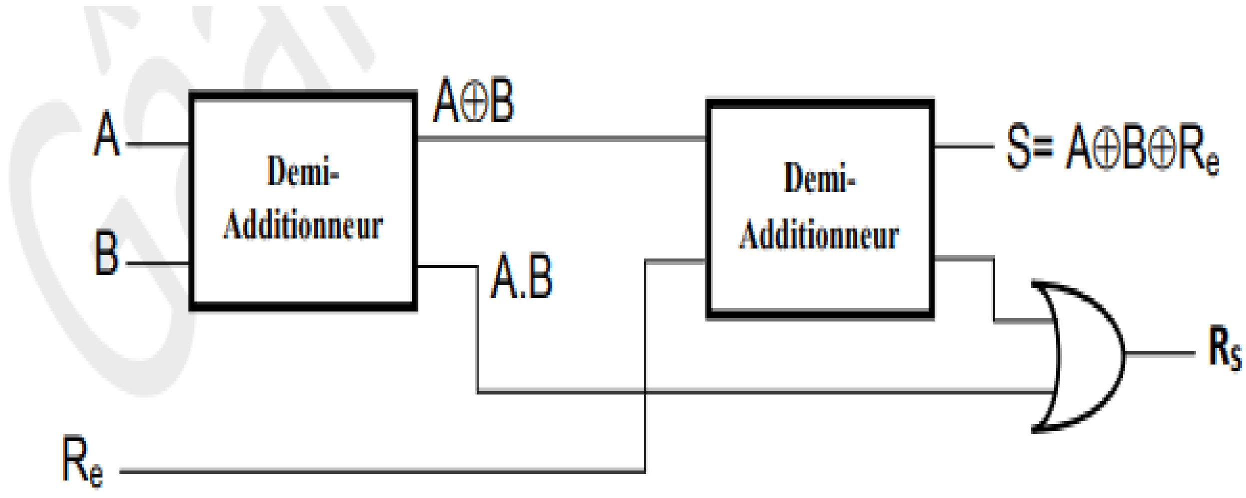
- Logigramme avec 2 demi-additionneurs :

DA $S = a \oplus b$

$R_{sor} = a.b$

CA $S = (a \oplus b) \oplus r$

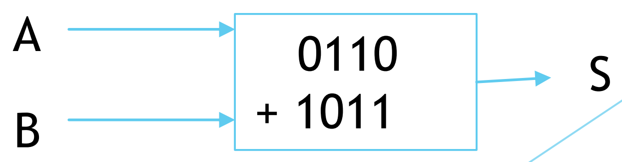
$R = a.b + r(a \oplus b).$



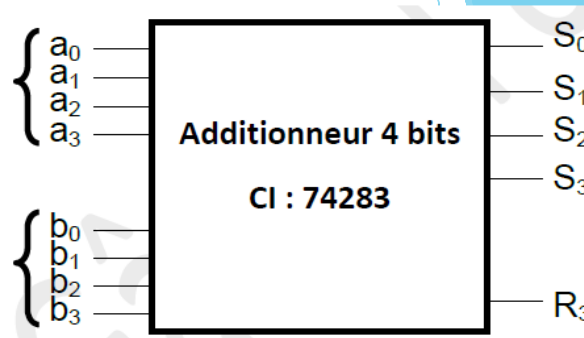
Additionneurs(Adder):

Additionneur parallèle:

- Pour additionner 2 nombres
- Il consiste à mettre des additionneurs **1 bit** en série, avec la retenue sortante de l'un qui devient entrante du suivant, ce qui correspond à la propagation de retenue.



Additionneurs(Adder):

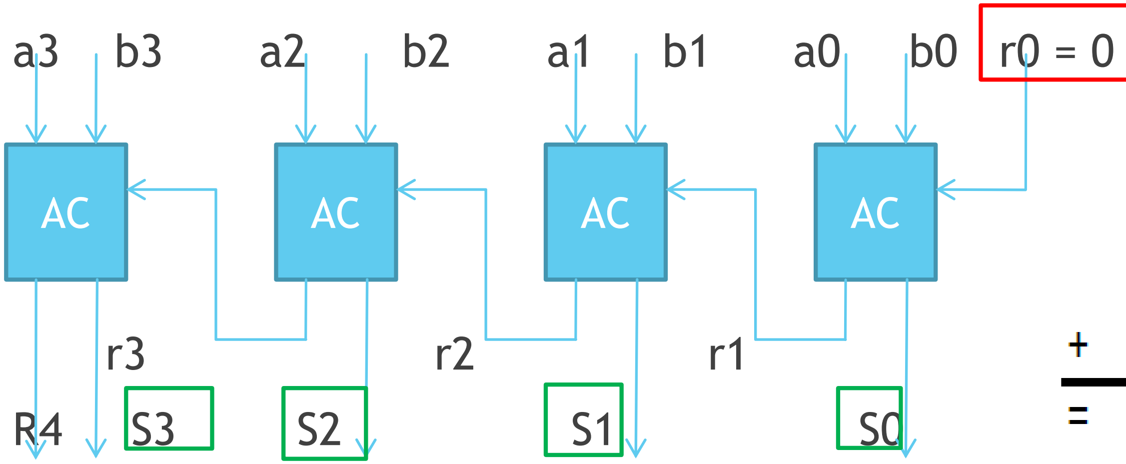


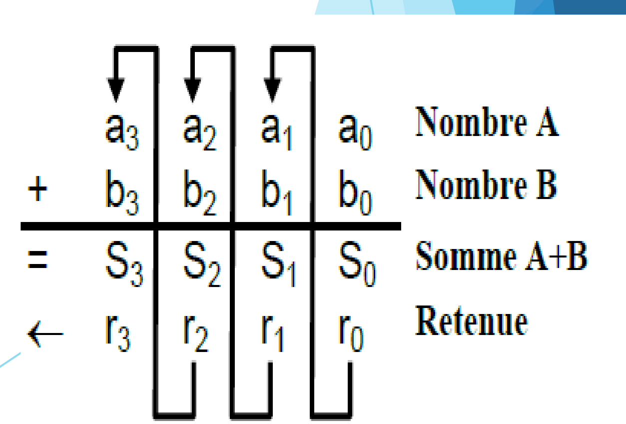
Additionneur parallèle:
Exemple : additionneur parallèle à 4 bits.
Soit à additionner les nombres binaires :

$$A = a_0a_1a_2a_3$$

$$B = b_0b_1b_2b_3$$

Additionneur 4 bits:





Soustracteurs (substractor)

- **Demi soustracteur :** 2 entrées sur 1 bit
  2 sorties sur 1 bits.
- **Soustracteur complet** : 3 entrées sur 1 bit
  2 sorties sur 1 bits.
- **Soustracteur sur n bits.**

Soustracteurs (substractor)

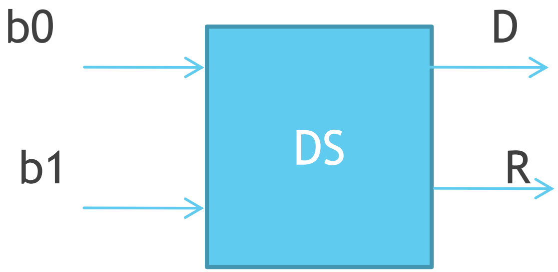
- **Demi soustracteur (half substractor):**
- C'est un circuit qui fait la soustraction de deux bits **b0** et **b1** de même poids,
- Le demi-soustracteur est un soustracteur binaire qui ne tient pas compte de la retenue provenant des bits de poids inférieurs.
- **D** représente le résultat de la différence (A-B) et **R** le report.

Soustracteurs (substractor)

Demi soustracteur (half substractor):

TV

b0	b1	D	R
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0



$$D = \bar{b}_0 . b_1 + b_0 . \overline{b_1}$$

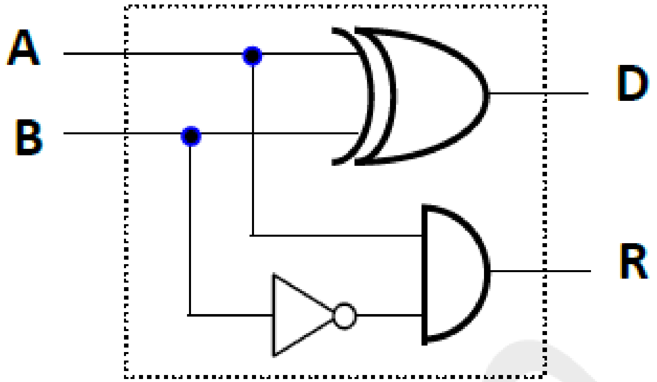
$$D = b_0 \oplus b_1$$

$$R = \bar{b}_0 . b_1$$

Soustracteurs (substractor)

- **Demi soustracteur (half substractor):**
- Expressions logiques : $D = b_0 \oplus b_1$

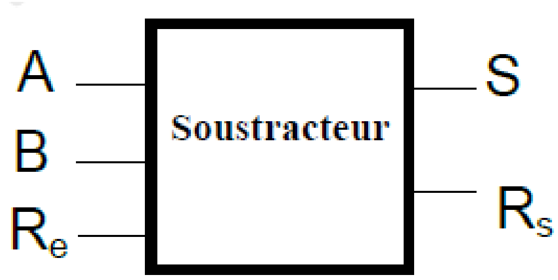
  $R = \bar{b}_0 . b_1$
- Logigramme :



Soustracteurs

Soustracteur complet (full substractor):

- c'est un circuit qui fait la soustraction de deux bits **b0** et **b1** de même poids plus le report de l'étape précédente **R0**



Soustracteurs (substractor)

Soustracteur complet (full substractor) :

$$D = \bar{A}\bar{B}R_e + \bar{A}B\overline{R_e} + A\bar{B}\overline{R_e} + ABR_e$$

$$D = R_e(\bar{A}\bar{B} + AB) + \overline{R_e}(\bar{A}B + A\bar{B})$$

$$\boxed{D = R_e \oplus (A \oplus B)}$$

$$R_s = \bar{A}\bar{B}R_e + \bar{A}B\overline{R_e} + \bar{A}BR_e + ABR_e$$

$$R_s = R_e(\bar{A}\bar{B} + \mathrm{AB}) + \bar{A}B(\overline{R_e} + R_e)$$

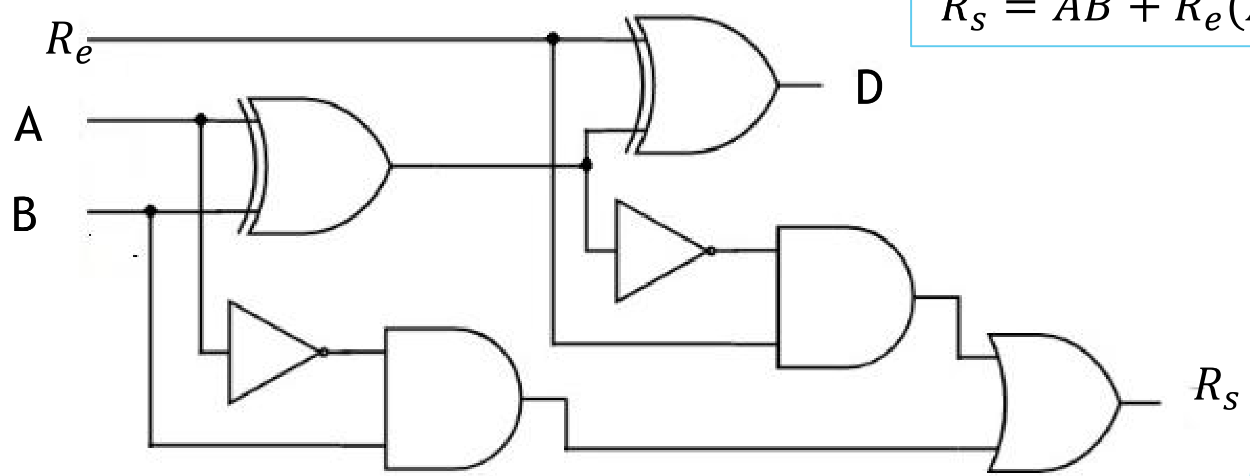
$$\boxed{R_s = \bar{A}B + R_e(\overline{A \oplus B})}$$

Soustracteurs (substractor)

- **Soustracteur complet (full substractor):**
- Logigramme :

$$D = R_e \oplus (A \oplus B)$$

$$R_s = \bar{A}B + R_e(\overline{A \oplus B})$$



Soustracteurs (substractor)

- **Soustracteur complet (full substractor):**

DS

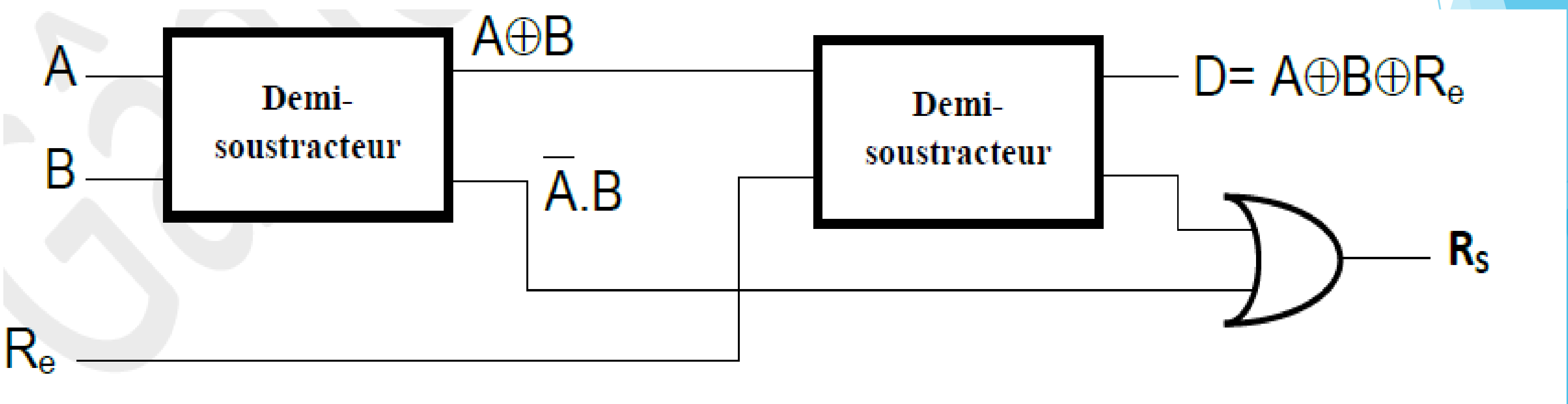
$D = A \oplus B$

R= $\bar{A}B$

SC

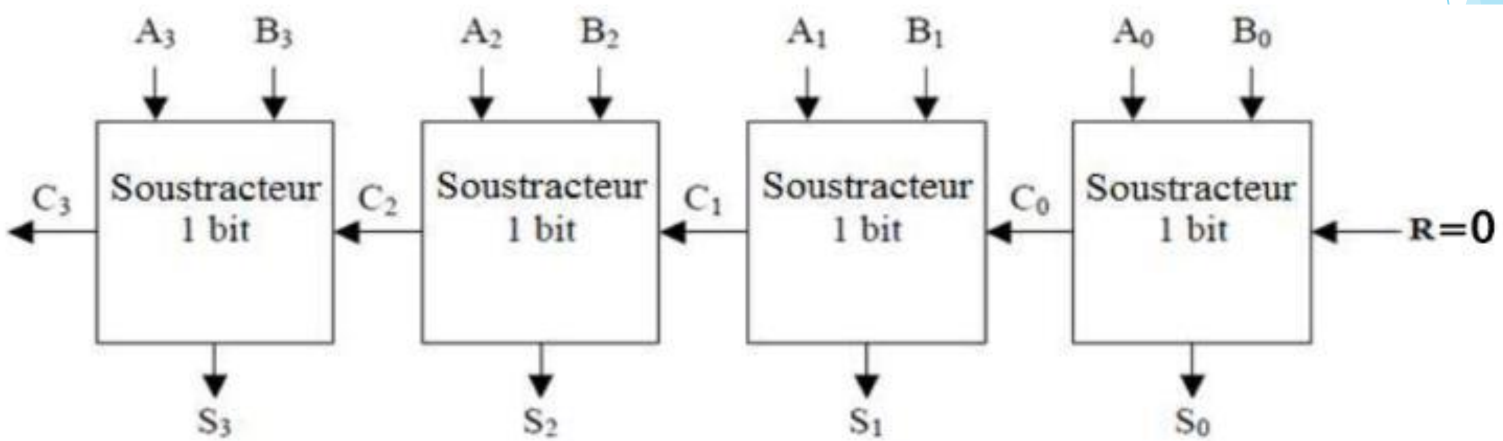
$D = R_e \oplus (A \oplus B)$

$R_s = \bar{A}B + R_e(\overline{A \oplus B})$



Soustracteurs (substractor)

- **Soustracteur sur n bits :**
- **Exemple :** additionneur parallèle à 4 bits. Soit à additionner les nombres binaires : A= A0 A1 A2 A3   et   B = B0 B1 B2 B3



Soustracteurs (substractor)

Remarque:

En binaire les nombres négatifs sont représentés en complément à 2

or sur n bits $2^n = 0$. ➔ $-A = \bar{A} + 1$

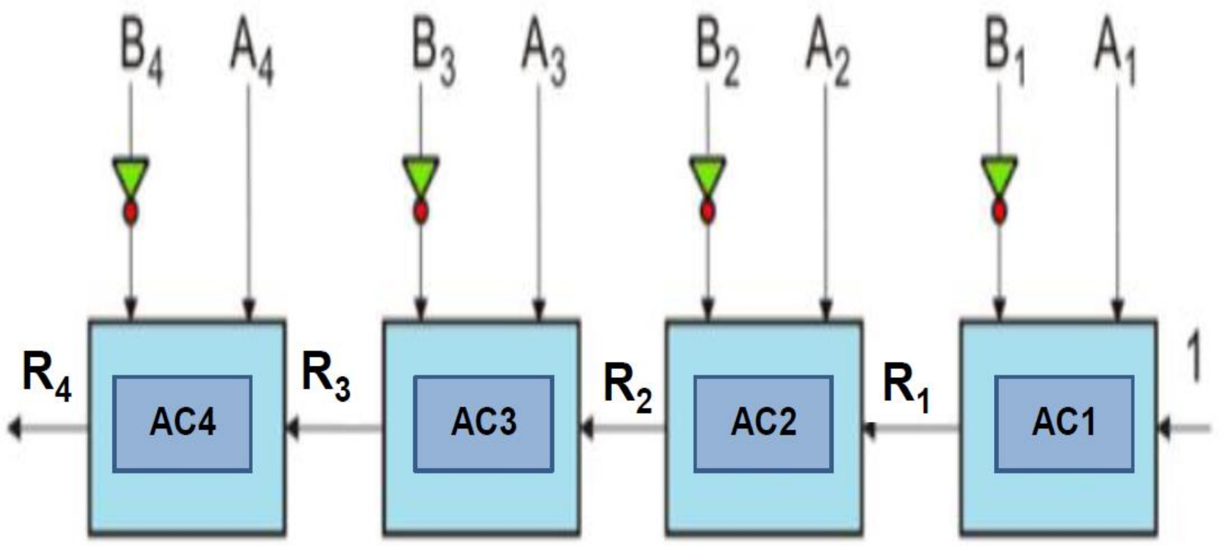
- Ce qui veut dire que l'on peut transformer la soustraction en addition:

$$\boxed{A - B = A + \bar{B} + 1}$$

- On utilise un additionneur au lieu d'un soustracteur.

Soustracteurs (substractor)

- **Exercice 1 :**



Soustracteurs (substractor)

- **Exercice :**

Concevoir un circuit qui permet de faire l'addition ou la soustraction (additionneur/soustracteur) de deux nombres binaires A et B de 1 bit.

On rappelle que dans la représentation en complément à 2,

$$\boxed{A - B = A + \bar{B} + 1}$$

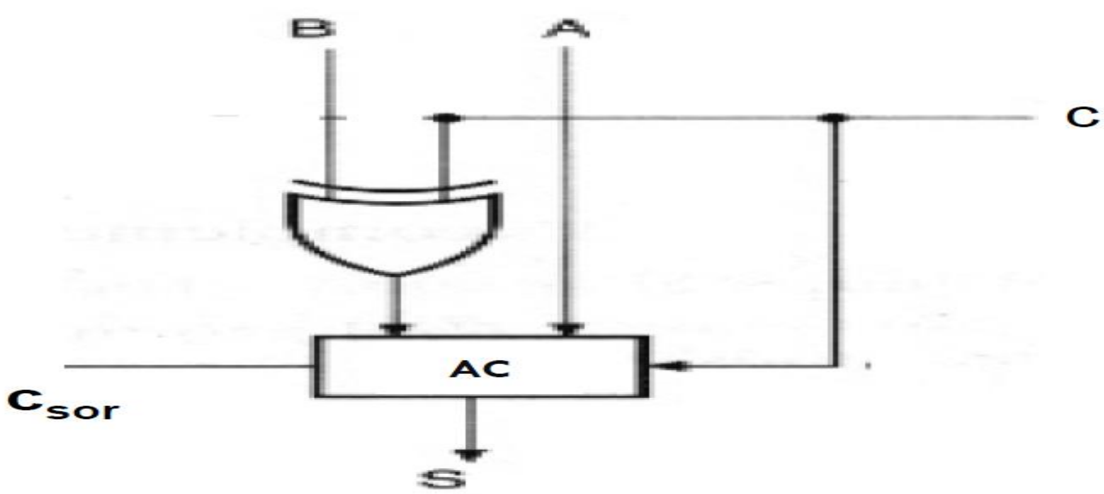
- Cet additionneur/soustracteur possèdera une entrée de commande C qui sera utilisée comme suit :
- C=0 fonctionnement en addition.
- C=1 fonctionnement en soustraction.

Soustracteurs (substractor)

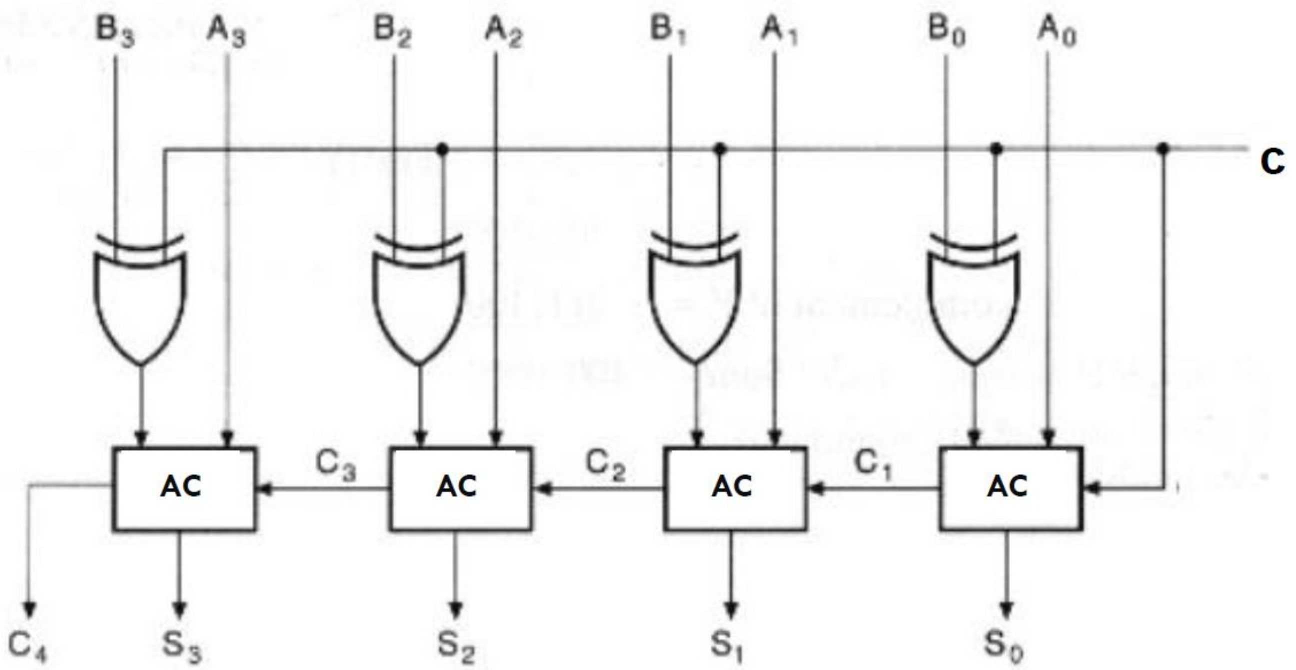
- **Exercice :**

  - C=0 fonctionnement en addition. — $B_{in}=B$
  - C=1 fonctionnement en soustraction — $B_{in}=/B$

B	C	B_{in}
0	0	0
0	1	1
1	0	1
1	0	0



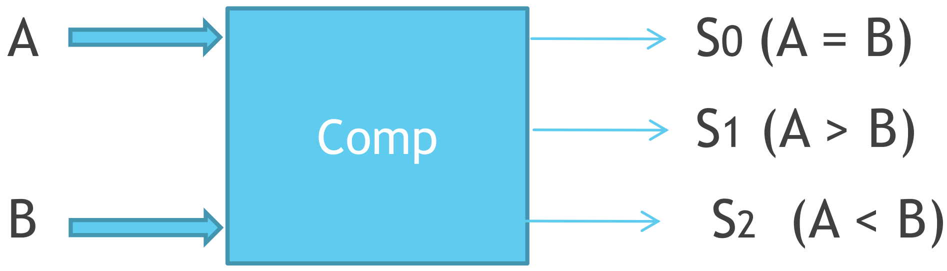
Soustracteurs (substractor)



Circuits combinatoires

Comparateur:

Il s'agit de comparer 2 nombres A et B de n bits chacun, avec comme résultat 3 sorties correspondant à $S_0(A = B)$, $S_1(A < B)$ et $S_2(A > B)$.



A	B	S_0	S_1	S_2
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	0	0

Circuits combinatoires

Comparateur:

TV d'un comparateur 1 bit:

$S_0(A = B)$, $S_1(A < B)$ et $S_2(A > B)$.

A	B	S_0	S_1	S_2
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	0	0

On déduit:

$$S0 = \bar{a}.\bar{b} + a.b = \bar{a} \oplus b$$

$$S_1 = a.\bar{b}$$

$$S_2 = \bar{a}.b$$

$$\boxed{S_0 = \overline{S1 + S2}}$$

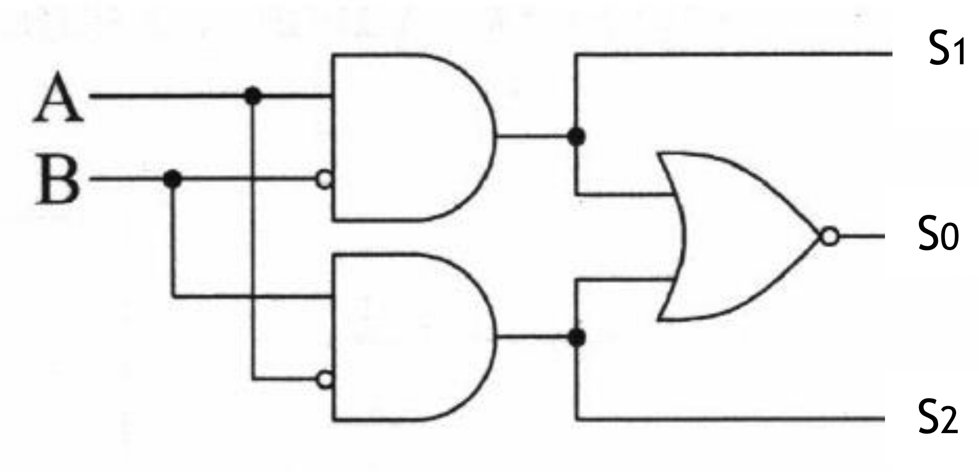
Circuits combinatoires

$S1 = a.\bar{b}$

$S2 = \bar{a}.b$

$\boxed{S0 = \overline{S1 + S2}}$

Comparateur:



Circuits combinatoires

- ***Exemple : Comparateur à 2 bits***
  - Etablir la TV du circuit;
  - Générer les équations de sortie: S0,S1,S2
  - Réaliser le même circuit à l'aide de circuits comparateurs 1 bit et des portes.

On dit que:

A = B(S0=1)   Si A1 = B1, **et** A0 = B0;

A > B (S1=1) Si A1 > B1 **ou** A1 = B1 **et** A0 > B0;

A < B (S2=1) Si A1 < B1 **ou** A1 = B1 **et** A0 < B0.

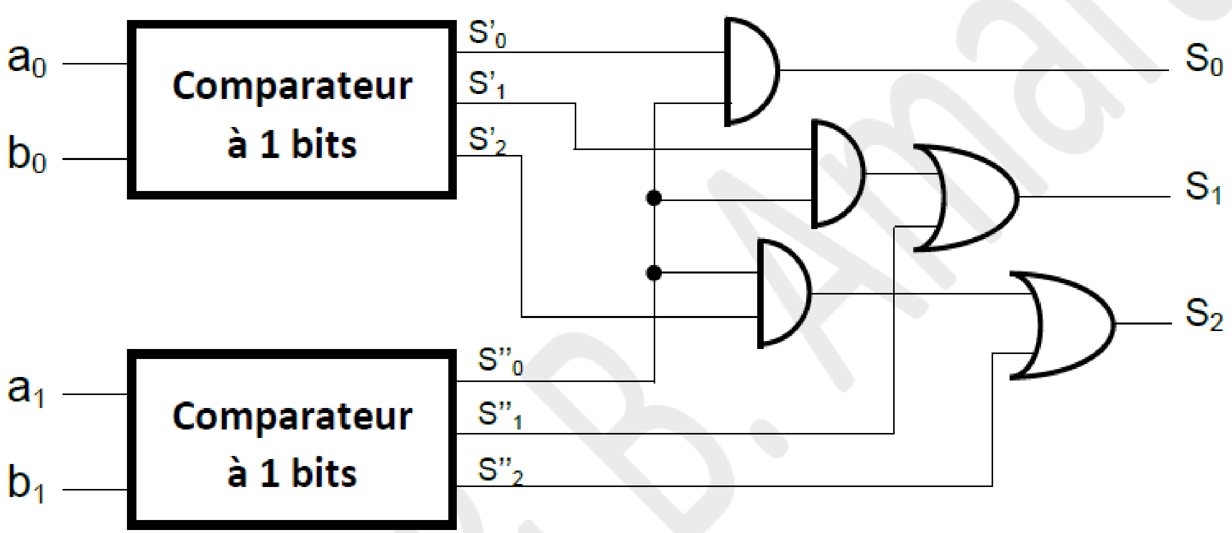
Circuits combinatoires

Le comparateur de 2 bits

S_0 vaut 1 si (a_1=b_1 et a_0=b_0) $S_0 = S_0'.S_0''$

Et S_1 vaut 1 si a_1>b_1 ou si (a_1=b_1 et a_0>b_0) $S_1 = S_1'' + S_0''S_1'$

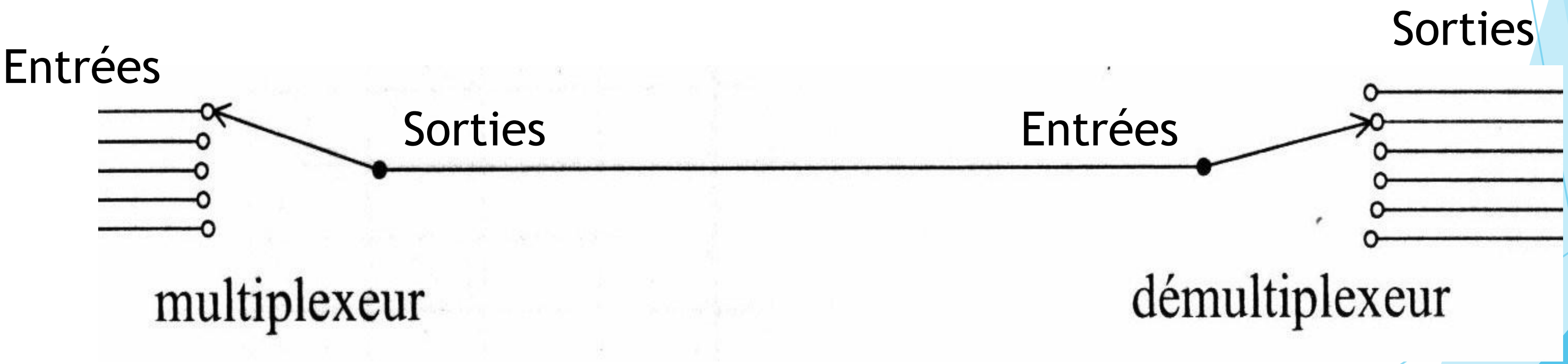
Et S_2 vaut 1 si a_1<b_1 ou si (a_1=b_1 et a_0<b_0) $S_2 = S_2'' + S_0''S_2'$



Circuits combinatoires

Multiplexage/Démultiplexage:

-
- C'est un dispositif qui permet de transmettre sur une seule ligne des informations en provenance de plusieurs sources possibles à destination de plusieurs cibles



Multiplexeurs

a) **Multiplexeur:**

C'est un circuit qui met en relation **1** entrée parmi **n**, avec la sortie, d'où la nécessité de sa sélection.

- MUX 2:1, MUX 4:1, MUX 8:1, MUX 16:1………..

Exemple de Mux 2 vers 1 :

Soient **2 entrées(E1,E2)** d'information et **1 sortie (S)**

donc: **1 commande (C0)** pour sélectionner **l'une des 2 entrées.**

Multiplexeurs

Table de vérité

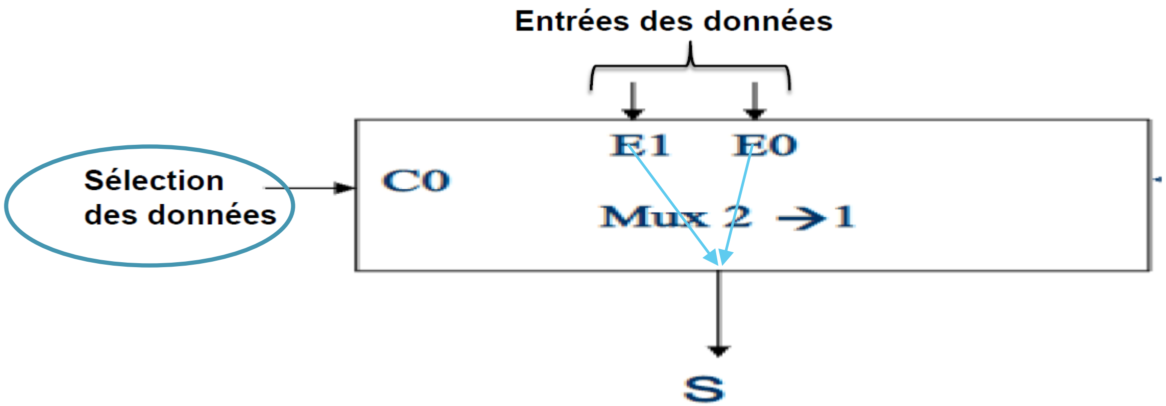
C0	S
0	E0
1	E1

$S = E0.\bar{C}0$

$S = E1.C0$

Equation

$S = E0.\overline{C0} + E1.C0$

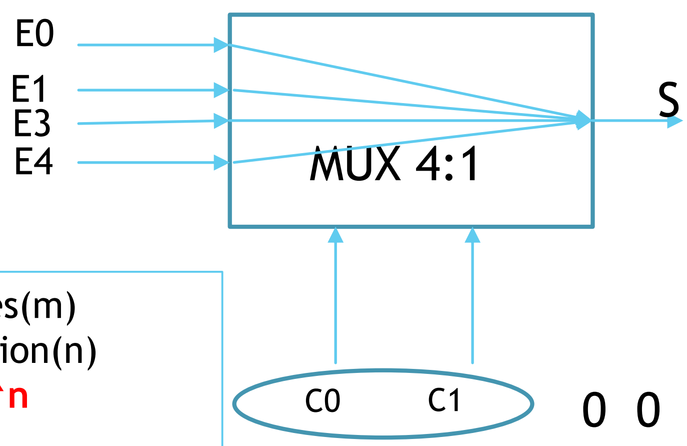


Multiplexeurs

- **Multiplexeur 4 à 1 (MUX 4 :1)**

Soient **4 entrées** d'information et **1 sortie**,

donc: 2 commande pour sélectionner **l'une des 4 entrées**



C0	C1	S
0	0	E0
0	1	E1
1	0	E2
1	1	E3

4 ligne d'entrées(m)
=2 ligne de section(n)
m=2^n

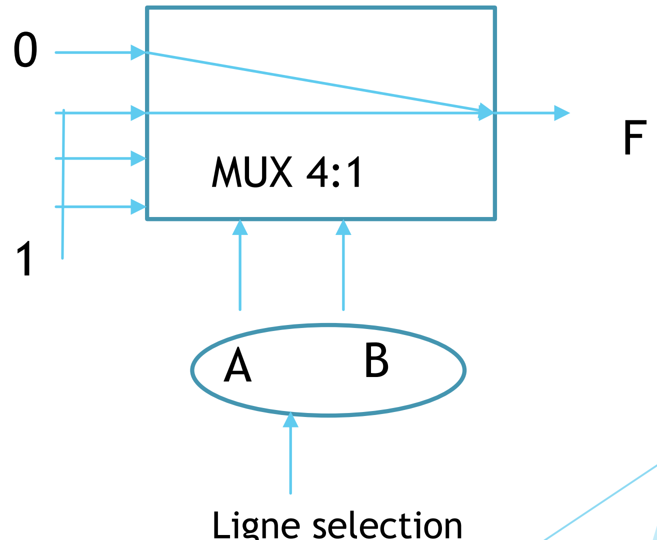
0 0

$$S = \overline{C1}.\overline{C0}.(E0) + \overline{C1}.C0.(E1) + C1.\overline{C0}.(E2) + C1.C0.(E3)$$

Multiplexeurs

- Exemple: TV

A	B	F
0	0	$\boxed{0}$
0	1	1
1	0	1
1	1	1

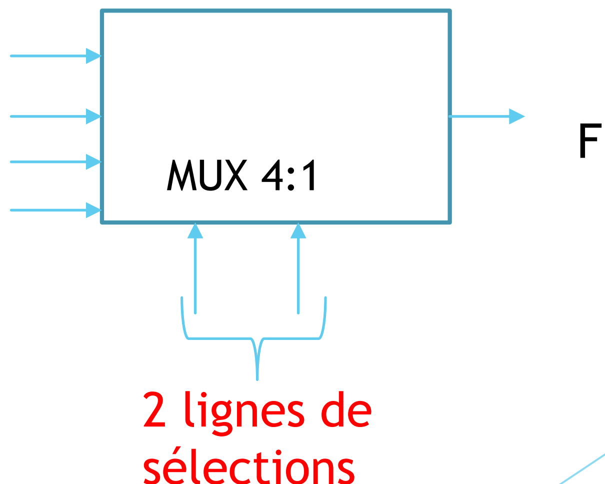


Si la fonction(5 variable):
MUX 2^5:1(MUX 32:1)

Multiplexeurs

- **Multiplexeur 4 à 1 (MUX 4 :1) :**
- Exemple: En utilisant un MUX 4 :1 et des portes logiques, réaliser la fonction suivante :

$$f(\underbrace{\boxed{A, B, C, D}}_{\text{4 variables}}) = \sum(1,4,5,7,9,12,13)$$



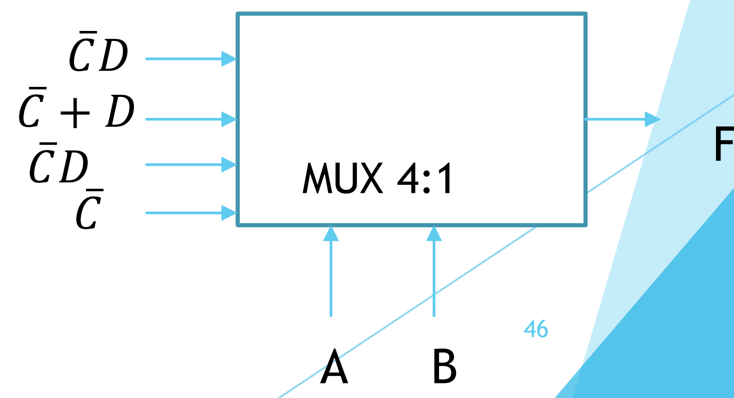
Multiplexeurs

AB / CD	00	01	11	10
00		1	1	
01	1	1	1	1
11		1		
10				

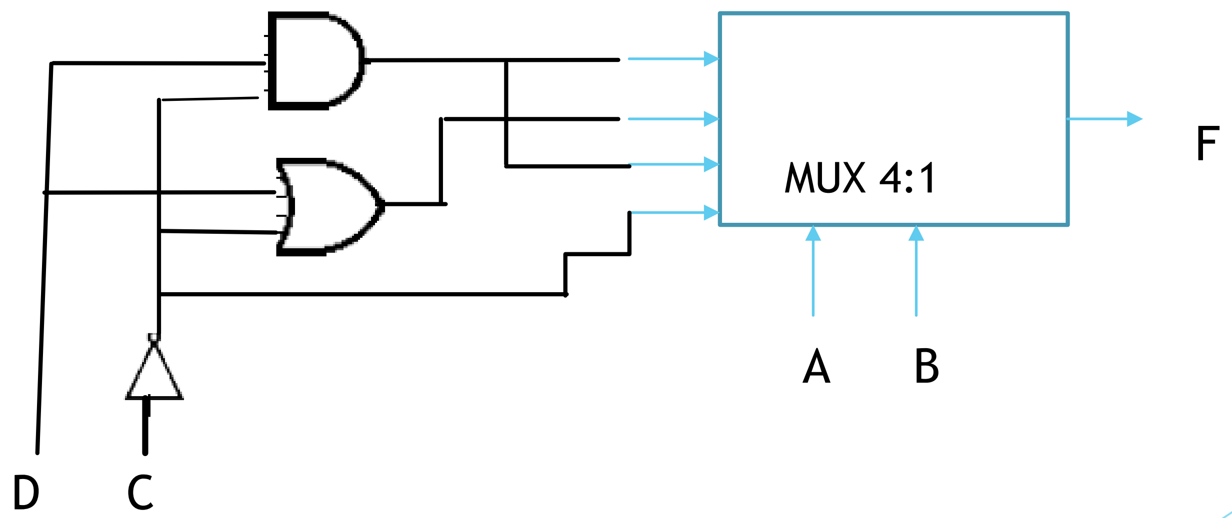
$\bar{C}D$

$\bar{C}D$ $\quad \bar{C}+D$ $\quad \bar{C}$

A	B	F
0	0	$\bar{C}D$
0	1	$\bar{C}+D$
1	0	$\bar{C}D$
1	1	$\bar{C}$

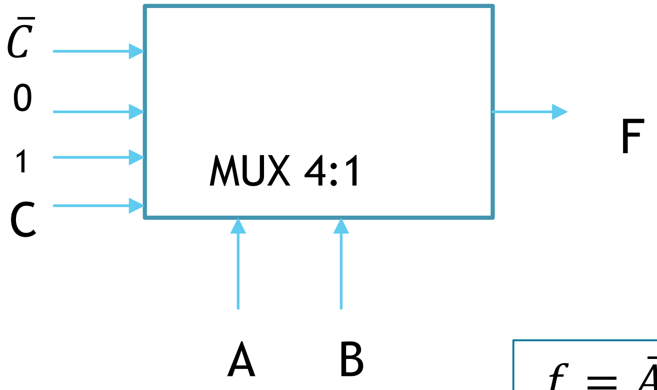


Multiplexeurs



Multiplexeurs

- Exemple: MUX(4:1) de la fonction f(A,B,C)



A	B	F
0	0	$\bar{C}$
0	1	0
1	0	1
1	1	C

$$f = \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}0 + A\bar{B}1 + ABC$$

$$f = \bar{A}\bar{B}\bar{C} + A\bar{B} + ABC$$

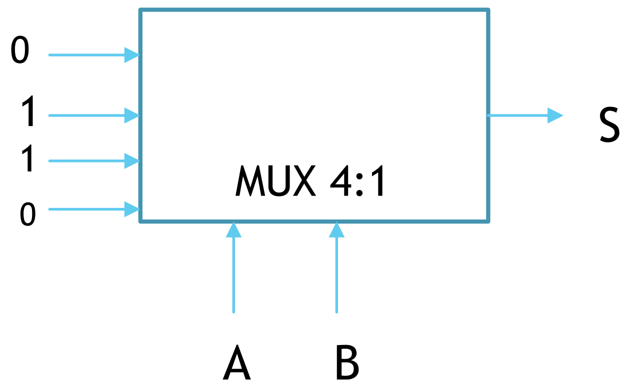
Multiplexeurs

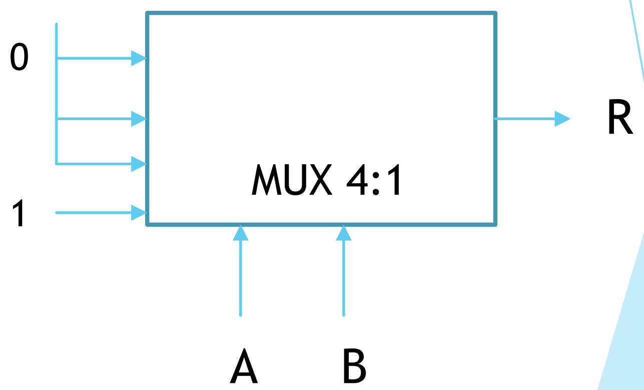
- **Half adder(demi additionneur)**
- En utilisant un MUX 4 1 et des portes logiques:

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

Multiplexeurs

➢ **Half adder(demi additionneur)**





Multiplexeurs

- **Exercice :**

En utilisant un MUX 4 1 et des portes logiques, réaliser les fonctions suivantes :

$$1)\ f1(a,b,c) = \sum m(0,1,2,7) \qquad 2)\ f2(x,y,z) = \prod M(0,1,4,5,7)$$