Chapitre II : Les circuits logiques combinatoires

I. Introduction

La transmission de données nécessite fréquemment des opérations de conversion, de transpostage et d'aiguillage. On utilise pour cela des **circuits combinatoires**. Pour réaliser un circuit logique combinatoire, le concepteur doit utiliser plusieurs portes logiques élémentaires. Pour faciliter sa tache, les fabricants fournissent des circuits sous forme intégrés comportant chacun plusieurs portes à des degrés d'intégration différents.

Il existe plusieurs dispositifs logiques combinatoires couramment utilisé dans les systèmes numériques. On peut citer les codeurs, les décodeurs, les transcodeurs, les multiplexeurs, les démultiplexeurs, les comparateurs …

II. Définition

La logique combinatoire concerne l'étude des fonctions dont la valeur de sortie ne dépend que de l'état logique des entrées se traduisant par une modification de la valeur des sorties et non pas non plus de ses états antérieurs : à chaque combinaison des variables d'entrée correspond toujours une seule combinaison des fonctions de sortie.

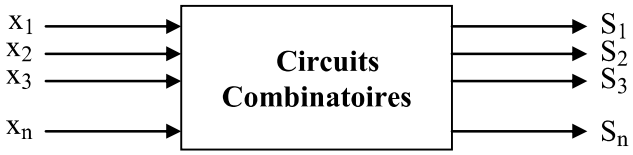


Figure 1 : Circuit combinatoire

III. Les Codeurs

1. Définition

Le **codeur** (ou encodeur) est un circuit logique qui possède 2^N voies entrées, dont une seule est activée et N voies de sorties. Il fournit en sortie le code binaire correspondant.

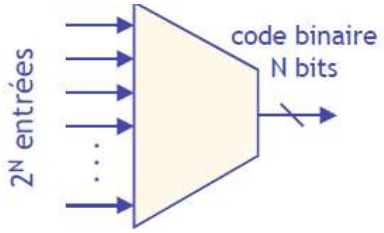


Figure 2: Schéma fonctionnel d'un codeur

2. Principe d'un codeur 4 voies d'entrées et 2 bits de sortie

a. Schéma fonctionnel

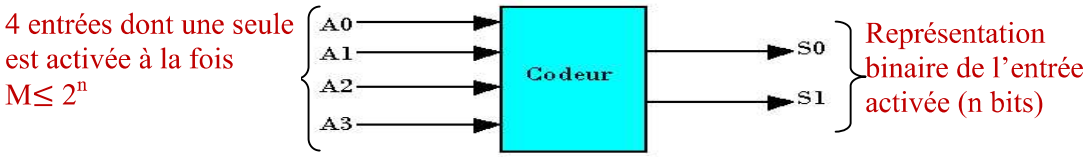


Figure 3 : Schéma fonctionnel d'un codeur 4 voies d'entrées et 2 bits de sortie

b. Table de vérité

Entées				Sorties	
Codage 1 parmi 2^n				Nombre binaire de n bits	
A_3	A_2	A_1	A_0	S_1	S_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

c. Equation des sorties

S_1=1 si (A_2=1) ou (A_3=1) ; $S_1=A_2+A_3$

S_0=1 si (A_1=1) ou (A_3=1) ; $S_0=A_1+A_3$

d. Logigramme

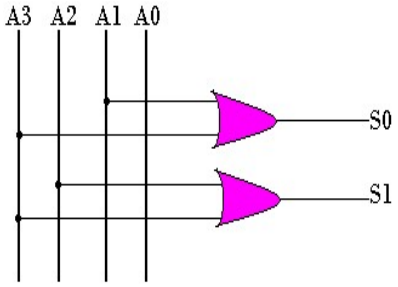


Figure 4: Schéma du logigramme d'un codeur

Si nous activons simultanément les entrées A_1 et A_2 du codeur ci-dessus, les sorties S_1S_0 présente le nombre 11 qui ne correspond pas au code de l'une ou de l'autre des entrées activés. C'est plutôt le code qui représente l'activation de A_3.

Pour résoudre ce problème on utilise un codeur de priorité qui choisit le plus grand nombre lorsque plusieurs entrées sont activées à la fois.

Exemple, lorsqu'A1 et A_2 sont activées simultanément S_1S_0 sera égale à 10 qui représentent l'activation de A0.

3. Codeur de priorité

C'est un dispositif qui réalise le codage du numéro le plus élevé dans le cas ou plusieurs entrées seraient actionnées. Pour cette raison, ce codeur possède des circuits logiques en plus, de sorte que le code de sortie choisi quand deux entrés sont actives soit celui qui correspond au nombre supérieur

Table de vérité

Entrées									**Sorties**			
E_9	E_8	E_7	E_6	E_5	E_4	E_3	E_2	E_1	S_3	S_2	S_1	S_0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	0	1	X	0	0	1	0
0	0	0	0	0	0	1	x	x	0	0	1	1
0	0	0	0	0	1	x	x	x	0	1	0	0
0	0	0	0	1	x	x	x	x	0	1	0	1
0	0	0	1	x	x	x	x	x	0	1	1	0
0	0	1	X	x	x	x	x	x	0	1	1	1
0	1	x	X	x	x	x	x	x	1	0	0	0
1	x	x	X	x	x	x	x	x	1	0	0	1

4. Codeurs en circuits intégrés

a. Codeur BCD de priorité 74147

Le circuit intégré 74147 est un codeur de priorité à 9 entrées. Il est actif à l'état bas et produit à la sortie le code BCD inversé.

Table de vérité

Entrées									**Sorties**			
$\overline{E_9}$	$\overline{E_8}$	$\overline{E_7}$	$\overline{E_6}$	$\overline{E_5}$	$\overline{E_4}$	$\overline{E_3}$	$\overline{E_2}$	$\overline{E_1}$	$\overline{D}$	$\overline{C}$	$\overline{B}$	$\overline{A}$
0	X	x	X	x	x	X	x	x	0	1	1	0
1	0	x	X	x	x	X	x	x	0	1	1	1
1	1	0	X	x	x	X	x	x	1	0	0	0
1	1	1	0	x	x	X	x	x	1	0	0	1

1	1	1	1	0	x	X	x	x	1	0	1	0
1	1	1	1	1	0	X	x	x	1	0	1	1
1	1	1	1	1	1	0	x	x	1	1	0	0
1	1	1	1	1	1	1	0	x	1	1	0	1
1	1	1	1	1	1	1	1	0	1	1	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1

Les sortie de 71747 sont à 1 quand aucune des entrés n'est à son niveau vrai (bas), cela correspond au code inversé du chiffre 0.

Pour obtenir le code B.C.D à partir des sorties de 74147, il faut ajouter un inverseur à chacune des sorties.

b. Codeur prioritaire à 3 bits 74148

Le 74148 est un codeur de priorité à huit entrés, actifs à l'état bas. Le code de sortie est un code en binaire inversé. C'est un codeur très utile car il permet non seulement le codage d'un nombre à huit entrées mais un nombre supérieur.

- **Table de vérité**

Entrées									**sorties**				
$\overline{E_1}$	$\overline{I_7}$	$\overline{I_6}$	$\overline{I_5}$	$\overline{I_4}$	$\overline{I_3}$	$\overline{I_2}$	$\overline{I_1}$	$\overline{I_0}$	$\overline{A_2}$	$\overline{A_1}$	$\overline{A_0}$	$\overline{G_s}$	$\overline{E_0}$
1	x	X	x	x	x	x	x	x	1	1	1	1	1
0	0	X	x	x	x	x	x	x	0	0	0	0	1
0	1	0	X	x	x	x	x	x	0	0	1	0	1
0	1	1	0	x	x	x	x	x	0	1	0	0	1
0	1	1	1	0	x	x	x	x	0	1	1	0	1
0	1	1	1	1	0	x	x	x	1	0	0	0	1
0	1	1	1	1	1	0	x	x	1	0	1	0	1
0	1	1	1	1	1	1	0	x	1	1	0	0	1
0	1	1	1	1	1	1	1	0	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	1	1	0

- **Schéma interne du circuit intégré**

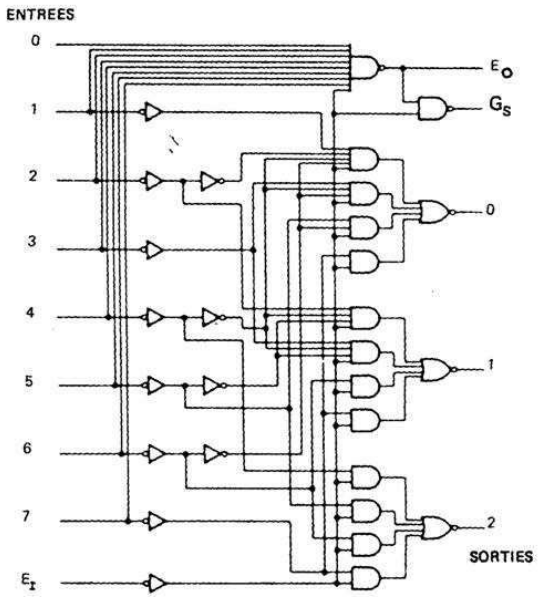


Figure 5: Schéma interne du circuit intégré

Ce codeur possède en plus des entres classiques du codeur de priorité, trois broches supplémentaires $\overline{E_0}$,$\overline{E_1}$ et $\overline{G_s}$. Le rôle de chacune de ces broches est décrit ainsi

- Si l'entrée $\overline{E_1} = 1$,alors le codeur n'est pas validé et les sorties $\overline{A_3} = \overline{A_2} = \overline{A_1} = \overline{E_0} = \overline{G_S}$ sont à 1 quelles que soient les entrés.
- Si l'entrée $\overline{E_1} = 0$,alors le codeur est validé et fournit le code correspondant à l'entrée prioritaire qui se trouve à l'état bas.
- Si l'entrée $\overline{E_1} = 0$, et si toutes les entrés I_i sont à 1(pas d'informations sur les entrées), alors l sortie $\overline{E_0}$ est à l'état bas.
- Les conditions $\overline{G_S} = 0$ et $\overline{E_1} = 0$,indiquent la présence d'au moins une information sur une entrés

Mise en cascade de circuits intégrés 74148

Pour réaliser le codage binaire dans un système à plus de huit entrées, on peut mettre plusieurs codeurs 74148 en cascade.

Exemple : Réalisation d'un codeur prioritaire à 4 bits par assemblage de deux codeurs à 3 bits

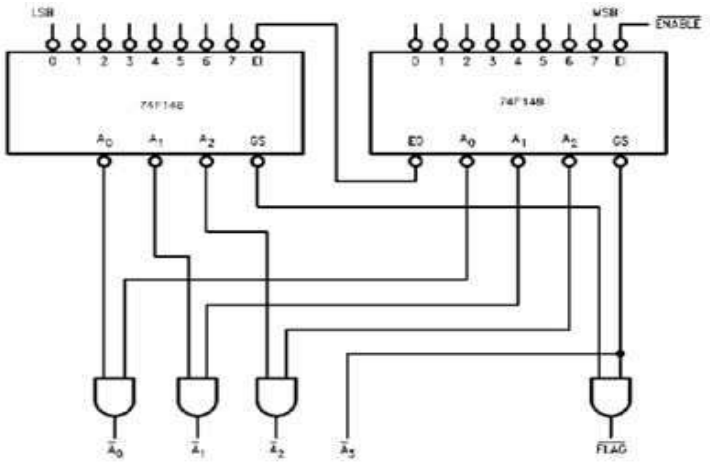


Figure 6: Réalisation d'un codeur prioritaire à 4 bits par avec deux codeurs à 3 bits 74148

IV. Les Décodeurs

1. Définition et fonctionnement

Un décodeur est un circuit logique combinatoire qui a une entrée binaire de n bits permettant 2^n combinaisons et M sorties telles que $2^n \geq M$.

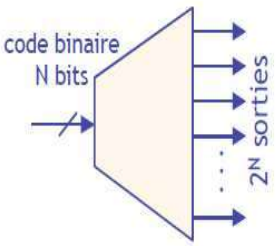


Figure 7: Schéma fonctionnel de décodeur

Suivant le type de décodeur, la sortie peut traduire deux fonctions:

- Convertisseur de code à un code de sortie d'entrée correspond un code de sortie.

Exemple: Un décodeur binaire octal possède 3 bits d'entrés permettant $2^3=8$ combinaisons pour activer chacun des 8 sortie de l'octal.

- Sélecteur de sortie: Une seule sortie parmi les M disponibles est activée à la fois en fonction de la valeur binaire affichée à l'entré. Ces fonctions permettent d'activer (sélectionner) un circuit intégré parmi plusieurs.

2. Principe d'un décodeur 1 parmi 4

Pour pouvoir activer toutes les 4 voies on a besoin de 2 bits à l'entrée car c'est $2^2=4$

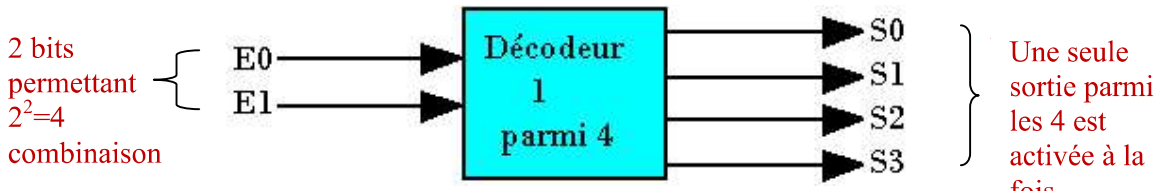


Figure 8: Schéma de principe d'un décodeur

a. Table de vérité

Table de fonctionnement					
Code binaire d'entrée		Codage 1 parmi 4 sorties			
E_1	E_0	S_3	S_2	S_1	S_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

b. Equations de sorties

$S_0 = \overline{E_1}.\overline{E_0}$ $S_2 = E_1.\overline{E_0}$

$S_1 = \overline{E_1}.E_0$ $S_3 = E_1.E_0$

c. Logigramme

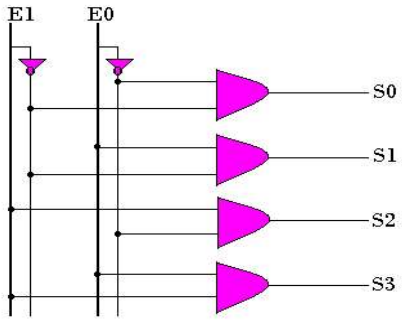


Figure 9 : Schéma de logigramme d'un décodeur

Remarque:

Certains n'utilisent pas toute la gamme de 2^n combinaisons d'entrées possibles. C'est le cas du décodeur DCB décimal qui a 4 bits d'entrée et 10 sorties donc une seule est active dans chacune des 10 représentations du DCB

3. Synthèse de décodeurs DCB 7segments

Les 10 chiffres décimaux (0 à 9) et parfois les caractères de l'hexadécimal (A à F) peuvent être configurés au moyen de 7 segments (voir ci-dessous). Chaque segment est constitué d'un matériau qui émet de la lumière lorsqu'il est traversé par un courant. Les matériaux les plus

utilisés sont les LED et les filaments incandescents.

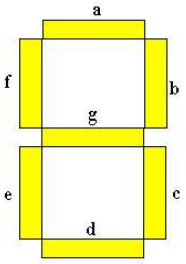


Figure 10: Disposition des 7 segments

a. Table de vérité

Entrées				Sorties							
a_3	a_2	a_1	a_0	A	B	C	D	E	F	G	Affichage
0	0	0	0	1	1	1	1	1	1	0	0
0	0	0	1	0	1	1	0	0	0	0	1
0	0	1	0	1	1	0	1	1	0	1	2
0	0	1	1	1	1	1	1	0	0	1	3
0	1	0	0	0	1	1	0	0	1	1	4
0	1	0	1	1	0	1	1	0	1	1	5
0	1	1	0	0	0	1	1	1	1	1	6
0	1	1	1	1	1	1	0	0	0	0	7
1	0	0	0	1	1	1	1	1	1	1	8
1	0	0	1	1	1	1	0	0	1	1	9

b. Équations logiques de sorties

$$\overline{A} = a_0\overline{a_1}.\overline{a_2}.\overline{a_3} + a_2\overline{a_1}.\overline{a_0}.\overline{a_3} = \overline{a_1}.\overline{a_3}(a_0 \oplus a_2)$$

$$\overline{B} = a_0\overline{a_1}.\overline{a_2}.\overline{a_3} + a_2a_1.\overline{a_0}.\overline{a_3} = a_2.\overline{a_3}(a_0 \oplus a_1)$$

$$\overline{C} = a_1\overline{a_0}.\overline{a_2}.\overline{a_3}$$

$$\overline{D} = a_0\overline{a_1}.\overline{a_2}.\overline{a_3} + a_2\overline{a_1}.\overline{a_0}.\overline{a_3} + a_0a_1a_2.\overline{a_3}$$

$$E = \overline{a_2}a_1.\overline{a_0}.\overline{a_3} + \overline{a_0}.a_1.\overline{a_2}.\overline{a_3} + \overline{a_0}.a_1.a_2.\overline{a_3} + \overline{a_0}.\overline{a_1}.a_3.\overline{a_2}$$

$$\overline{F} = a_0\overline{a_1}.\overline{a_2}.\overline{a_3} + a_1\overline{a_2}.\overline{a_0}.\overline{a_3} + a_0a_1.\overline{a_2}.\overline{a_3} + a_0a_1a_2.\overline{a_3}$$

$$\overline{G} = \overline{a_0}.a_1.\overline{a_2}.\overline{a_3} + a_0\overline{a_1}.\overline{a_2}.\overline{a_3} + a_0a_1a_2\overline{a_3}$$

4. Décodeurs en circuit intégrés

a. Décodeur B.C.D 7442

Le décodeur B.C.D est un décodeur à quatre bits d'entrée et à dix sorties, l'une d'entre elles étant seule validée à zéro. Les dix combinaisons de sortie sur les seize possibles sont employées pour désigner les dix chiffres décimaux 0 à 9.

- **Table de vérité**

Entrées				Sorties									
D	C	B	A	S_9	S_8	S_7	S_6	S_5	S_4	S_3	S_2	S_1	S_0
0	0	0	0	1	1	1	1	1	1	1	1	1	0
0	0	0	1	1	1	1	1	1	1	1	1	0	1
0	0	1	0	1	1	1	1	1	1	1	0	1	1
0	0	1	1	1	1	1	1	1	1	0	1	1	1
0	1	0	0	1	1	1	1	1	0	1	1	1	1
0	1	0	1	1	1	1	1	0	1	1	1	1	1
0	1	1	0	1	1	1	0	1	1	1	1	1	1
0	1	1	1	1	1	0	1	1	1	1	1	1	1
1	0	0	0	1	0	1	1	1	1	1	1	1	1
1	0	0	1	0	1	1	1	1	1	1	1	1	1

On note que pour toute combinaison supérieur à 9 (1001) à l'entrée, aucune sortie n'est validée (toutes les sorties sont à l'état haut).

b. Décodeurs de grande capacité

Compte tenu du nombre limité de connexions sur un circuit intégré, il est souvent utile de mettre en cascade les décodeurs pour permettre le décodage d'un grand nombre de combinaisons. Grace aux entrés de validation, on peut augmenter notablement la capacité du système de décodage.

Exemple :

Réaliser un décodeur 1 parmi 16 à l'aide de décodeurs 1 parmi 8

Solution

Accroissement de capacité de décodage par assemblage de deux décodeurs 3 bits pour réaliser un seul décodeur à 4 bits. Deux décodeurs traitent en parallèle les bits c_2, c_1, c_0 . Le bit c_3 sélectionne les sortis celui qui doit être actif

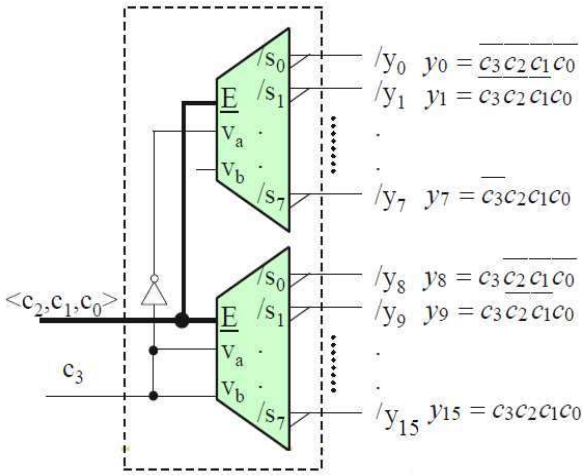


Figure 11 : Schéma de deux décodeurs en cascade

V. Transcodeur

1. Définition

Un **transcodeur** transforme une information disponible en entrée sous forme donnée (généralement un code) en la même information, mais sous une autre forme (généralement un autre code).

Il existe trois types de transcodeurs :

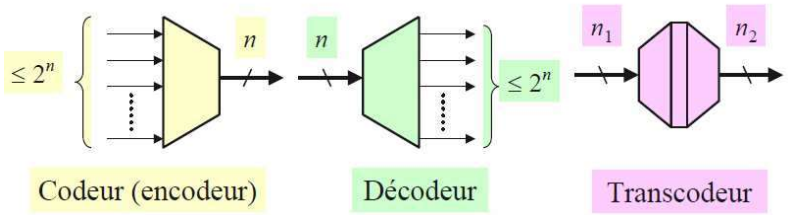


Figure 12 : Schéma d'un transcodeur

Les deux plus importantes applications des transcodeurs sont : la conversion de code et l'affichage par segment.

2. Conversion de code

a. Transcodeur binaire Gray

Pour passer d'un code à un autre, on utilisera un convertisseur de code. A titre d'illustration nous allons étudier le transcodeur binaire Gray.

Cherchons le circuit d'un transcodeur qui permet de convertir le code binaire 2 bits par exemple en code Gray.

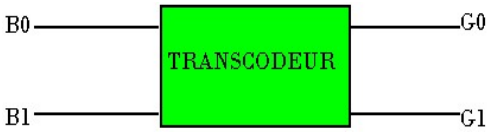


Figure 13 : Transcodeur binaire

Table de vérité

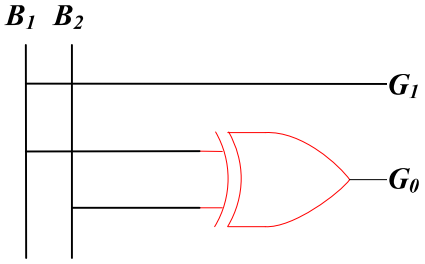
ENTREES		SORTIES	
B_1	B_0	G_1	G_0
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Equations de sorties

$$G_1 = B_1.\overline{B_0} + B_1.B_0 = B_1$$

$$G_0 = B_0.\overline{B_1} + B_1.\overline{B_0} = B_1 \oplus B_0$$

Logigramme



b. Transcodeur BCD – 7 segments

Un domaine d'application considérable des transcodeurs est celui de la conversion de donné binaire en une forme se prêtant à un affichage numérique. Les dix chiffres 0 à9 sont affichés au moyen d'un dispositif appelé afficheur à 7 segment lumineux qui sont des diodes électroluminescentes (D E L).les variables A,B,C,D sont écrites en BCD les variables de sortie a,b,c,d,e,f,g correspondent à chacun des segments de l'afficheur.

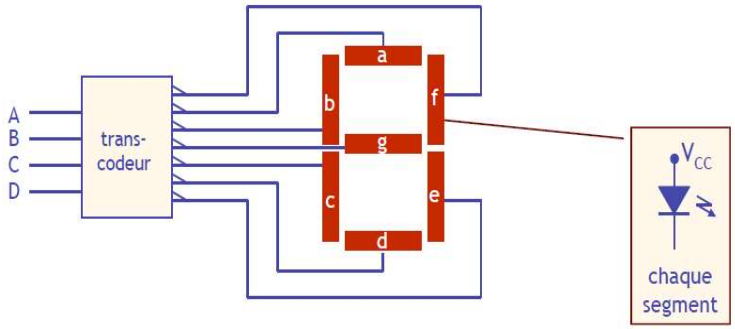


Figure 14 : Schéma fonctionnel de transcodeur BCD 7 segment

Table de vérité

Chiffres	ABCD	a	b	c	d	e	f	g
0	0000	1	1	1	1	1	1	0
1	0001	0	1	1	0	0	0	0
2	0010	1	1	0	1	1	0	1
3	0011	1	1	1	1	0	0	1
4	0100	0	1	1	0	0	1	1
5	0101	1	0	1	1	0	1	1
6	0110	1	0	1	1	1	1	1
7	0111	1	1	1	0	0	0	0
8	1000	1	1	1	1	1	1	1
9	1001	1	1	1	1	0	1	1

Pour obtenir les équations logiques de ce transcodeur, il faut établir le diagramme relatif à l'expression de chaque segment. On aura sept diagrammes. Etant donné que les 0 sont moins nombreux que les 1 dans ce diagramme de a,b,c,d,e,f,g correspondant à l'extinction des segments.

	$\overline{B}.\overline{A}$	$\overline{B}.A$	$B.A$	$B.\overline{A}$
$\overline{D}.\overline{C}$	1	0	1	1
$\overline{D}.C$	0	1	1	0
$D.C$	X	X	X	X
$D.\overline{C}$	1	1	X	X
$\overline{a} = A.\overline{B}.\overline{C}.\overline{D} + \overline{A}.C$				

	$\overline{B}.\overline{A}$	$\overline{B}.A$	$B.A$	$B.\overline{A}$
$\overline{D}.\overline{C}$	1	1	1	1
$\overline{D}.C$	1	0	1	0
$D.C$	X	X	X	X
$D.\overline{C}$	1	1	X	X
$\overline{b} = A.\overline{B}.C + \overline{A}.B.C$				

	$\overline{B}.\overline{A}$	$\overline{B}.A$	$B.A$	$B.\overline{A}$
$\overline{D}.\overline{C}$	1	1	1	0
$\overline{D}.C$	1	1	1	1
$D.C$	X	X	X	X
$D.\overline{C}$	1	1	X	X
$\overline{c} = \overline{A}.B.\overline{C}$				

	$\overline{B}.\overline{A}$	$\overline{B}.A$	$B.A$	$B.\overline{A}$
$\overline{D}.\overline{C}$	1	0	1	1
$\overline{D}.C$	0	1	0	1
$D.C$	X	X	X	X
$D.\overline{C}$	1	0	X	X
$\overline{d} = \overline{A}.\overline{B}.C + A.B.C + A.\overline{B}.\overline{C}$				

	$\overline{B}.\overline{A}$	$\overline{B}.A$	$B.A$	$B.\overline{A}$
$\overline{D}.\overline{C}$	1	0	0	1
$\overline{D}.C$	0	0	0	1
$D.C$	X	X	X	X
$D.\overline{C}$	1	0	X	X
$\overline{e} = \overline{B}.C + A$				

	$\overline{B}.\overline{A}$	$\overline{B}.A$	$B.A$	$B.\overline{A}$
$\overline{D}.\overline{C}$	1	0	0	0
$\overline{D}.C$	1	1	0	1
$D.C$	X	X	X	X
$D.\overline{C}$	1	1	X	X
$\overline{f} = A.\overline{C}.\overline{D} + A.B + B.\overline{C}$				

	$\overline{B}.\overline{A}$	$\overline{B}.A$	$B.A$	$B.\overline{A}$
$\overline{D}.\overline{C}$	0	0	1	1
$\overline{D}.C$	1	1	0	1
$D.C$	X	X	X	X
$D.\overline{C}$	1	1	X	X
$\overline{g} = \overline{B}.\overline{C}.\overline{D} + A.B.C$				

c. Transcodeur B .C.D- 7 segments en circuits intégré : MC-144495

Le MC-144495 est un transcodeur très souvent utilisable avec les afficheurs 7 segments.

Les sorties de ce transcodeurs et actives à l'état haut, pour cela il faut utiliser des afficheurs 7 segments à cathodes communes (la cathode commune est relié à la masse)

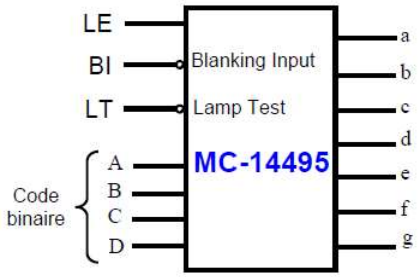


Figure 15 : Symbole logique

Table de vérité :

$\overline{LT}$	$\overline{BI}$	LE	Fonctionnement
0	X	X	Les 7 segments sont allumés
1	0	X	Les 7 segments sont éteints
1	1	1	Verrouillage des 7 segments sur le code d'entrée
1	1	0	Affiche en hexadécimal le code d'entrée

VI. Multiplexeur

1. Définition

Le multiplexeur (MUX) est un sélecteur de données qui permet d'aiguiller à l'aide des entrées de sélection (C_1, C_2,…, C_n) des données de provenances diverses (E_1, E_2,…, E_n) vers une seule sortie **S**. L'entrée sélectionnée est définie par son adresse.

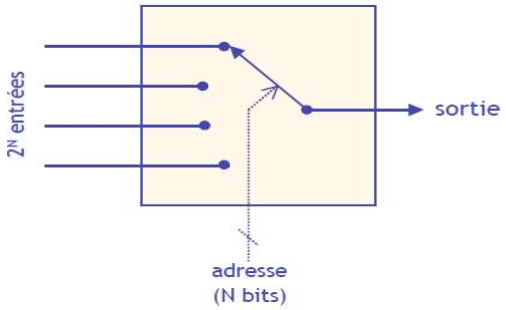


Figure 16: Multiplexeur 2^n vers 1

Table de vérité

Décimale	Cn	C2	C1	S
0	**0**	**0**	**0**	E0
1	**0**	**0**	**1**	E1
2	**0**	**1**	**0**	E2
3	**0**	**1**	**1**	E3
2n-1	**1**	**1**	**1**	En

2. Applications des multiplexeurs

- Conversion parallèle/série : aiguiller les informations présentes en parallèle à l'entrée du MUX en des informations de type série en sortie ; toutes les combinaisons d'adresses sont énumérées une par une sur les entrées de sélection.
- Réalisation de fonctions logiques : toute fonction logique de *N* variables est réalisable avec un multiplexeur de 2^N vers 1

a. Multiplexeur à 4 entrées (4 vers 1)

Un multiplexeur 4 vers 1 est un circuit logique qui est formé de 4 entrées E_0, E_1, E_2, E_3 qui sont transmises selon le choix indiqué par l'une des quatre combinaisons possibles des sorties de sélection C_0 et C_1

Table de fonctionnement

Décimale	C_0	C_1	S
0	0	0	E_0
1	0	1	E_1

2	1	0	E_2
3	1	1	E_3

Equation boolienne de sortie

$$S = \overline{C_1}.\overline{C_0}.E_0 + \overline{C_1}.C_0.E_1 + C_1.\overline{C_0}.E_2 + C_1.C_0.E_3$$

Circuit logique

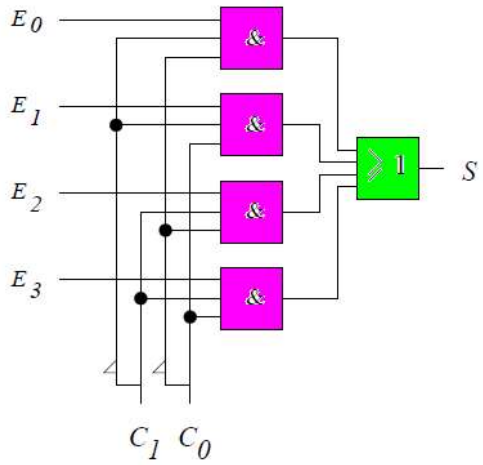


Figure 17 : Logigramme d'un multiplexeur

b. Multiplexeur en circuit intégré

- Multiplexeur 4-vers-1 : 74153
- Multiplexeur 8-vers-1 : 74151
- Multiplexeur 16-vers-1 : 74150

VII. Démultiplexeur

Le démultiplexeur réalise l'inverse d'un MUX : il aiguille une seule entrée vers une parmi 2^n vois de sorties. Les démultiplexeur fonctionnent comme un commutateur. Ils comportent une entrée de donné **E**, n entrées de sélection (C_1, C_2,…, C_n) et 2^n sorties (S_1, S_2,…, S_2^n)

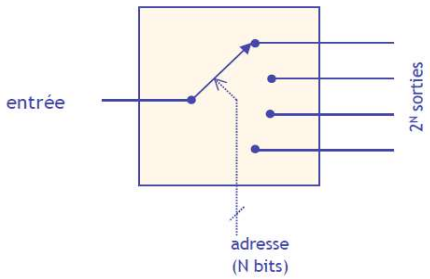


Figure 18: Démultiplexeur 1 vers 2^n

Les démultiplexeurs sont surtout utilisés dans les conversions série - parallèle. Ils peuvent aussi faire office de décodeur.

Table de vérité

Décimale	C_n	C_2	C_1	S_1	S_2		S_2^n
0	**0**	**0**	**0**	E	0		0
1	**0**	**0**	**1**	0	E		0
2	**0**	**1**	**0**	0	0		0
3	**0**	**1**	**1**	0	0		0
							0
2n-1	**1**	**1**	**1**	0	0		E

Remarque
Dans certains cas on trouve :

S_i à 1 lorsqu'elles ne sont pas sélectionnées à la place de 0

$\overline{E}$ à la palace de E dans les S_i, lorsqu'elles sont sélectionnés.

Démultiplexeur en circuit intégré

Démultiplexeur (décodeur) 8-vers-1 :74138

Décodeur /démultiplexeur : 74154

VIII. Comparateur

C'est un circuit permettant de comparer 2 mots de n bits chacun en indiquant sur ses sorties S_1, S2 ou S_3 si le premier mot est égal, plus grand ou plus que le second.

1. Comparateur 2 bits

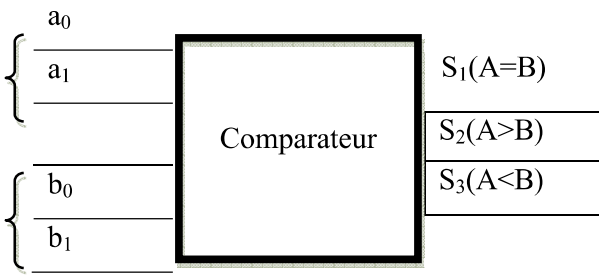


Figure 19 : Schéma fonctionnel de comparateur 2 bits

Table de vérité

b_1	b_0	a_1	a_2	S_1	S_2	S_3
0	**0**	**0**	**0**	**1**	**0**	**0**
0	**0**	**0**	**1**	**0**	**1**	**0**
0	**0**	**1**	**0**	**0**	**1**	**0**
0	**0**	**1**	**1**	**0**	**1**	**0**

0	1	0	0	0	0	1
0	1	0	1	1	0	0
0	1	1	0	0	1	0
0	1	1	1	0	1	0
1	0	0	0	0	0	1
1	0	0	1	0	0	1
1	0	1	0	1	0	0
1	0	1	1	0	1	0
1	1	0	0	0	0	1
1	1	0	1	0	0	1
1	1	1	0	0	0	1
1	1	1	1	1	0	0

Equations logiques de sorties

$$S_1 = \overline{a_0}\,\overline{a_1}.\overline{b_0}.\overline{b_1} + a_0\overline{a_1}.b_0.\overline{b_1} + a_1\overline{a_0}.b_1.\overline{b_0} + a_0a_1b_0b_1 = \overline{(a_0 \oplus b_0)+(a_1 \oplus b_1)}$$

$$S_2 = a_0\overline{a_1}.\overline{b_0}.\overline{b_1} + a_1\overline{a_0}.\overline{b_0}.\overline{b_1} + a_0a_1\overline{b_1}.\overline{b_0} + \overline{a_0}a_1b_0\overline{b_1} = a_1\overline{b_1} + a_0.\overline{b_0}.\overline{b_1} + a_0a_1\overline{b_0}$$

$$S_3 = \overline{S_1 + S_2}$$

Comparateur en circuit intégrés

74 85 TTL (8bits)

54 85 TTL (8bits)

40 05 CMOS (8bits)

IX. Les Additionneurs et les soustracteurs

1. Addition binaire

a. Demi-additionneur

Addition et soustraction sont deux opérations arithmétiques de base. Commençons par l'addition de deux nombres binaires, la soustraction sera étudiée dans le prochain paragraphe. En base 2 l'addition de deux bits s'écrit :

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

Comme en décimal, nous devons donc tenir compte d'une éventuelle retenue (carry). La figure ci-dessous montre la décomposition de l'addition de deux nombres binaires de quatre bits.

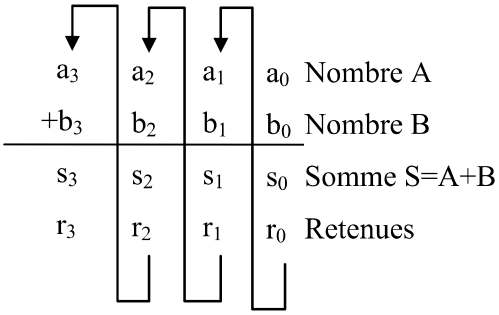
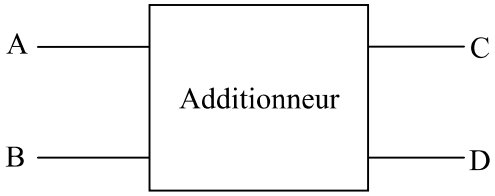


Figure 20: Décomposition de l'addition de deux nombres binaires de quatre bits.

L'addition des deux bits de plus bas poids (LSB : Least Significant Bit) a_0 et b_0, donne un résultat partiel s_0 et une retenue r_0. On forme ensuite la somme des deux bits a_1 et b_1 et de la retenue r_0.

Nous obtenons un résultat partiel s_1 et une retenue r_1. Et ainsi de suite, nous obtenons un résultat sur quatre bits S et une retenue r_3.

Considérons la cellule symbolisée sur la figure suivante, comptant deux entrées A et B, les deux bits à sommer, et deux sorties D le résultat de la somme et C la retenue.



Ce circuit, qui permettrait d'effectuer l'addition des deux bits de plus bas poids est appelé demi-additionneur (Half-Adder). Ecrivons la table de vérité de celui-ci :

Table de vérité

A	B	C	D
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Equations des sorties

Si nous écrivons ces deux fonctions sous leur forme canonique il vient :

$$\begin{cases} D = \overline{A}.B + A.\overline{B} \\ C = A.B \end{cases}$$

Nous reconnaissons pour la sortie D une fonction OU exclusif, donc :

$$\begin{cases} D = A \oplus B \\ C = A.B \end{cases}$$

Logigramme

Ce qui peut être réalisé par le circuit schématisé sur le logigramme de la figure suivante.

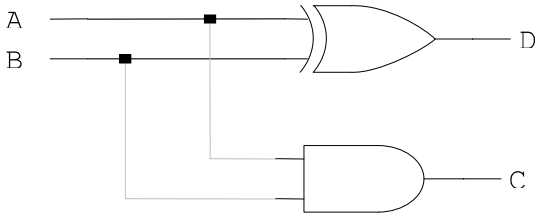


Figure 21 : Logigramme d'un demi-additionneur

b. Additionneur complet

Il faut en fait tenir compte de la retenue des bits de poids inférieurs, un circuit additionneur doit donc comporter trois entrées et deux sorties, comme représenté sur la figure suivante.

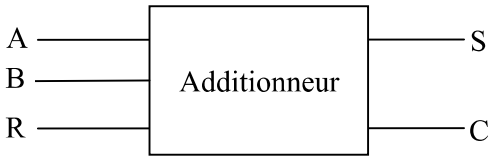


Figure 22: Additionneur à trois entrées et deux sorties

Ce serait possible en combinant deux demi-additionneurs comme présenté par la figure 5. En pratique pour minimiser le nombre de composants, ou de portes dans un circuit intégré, un tel additionneur est réalisé directement.

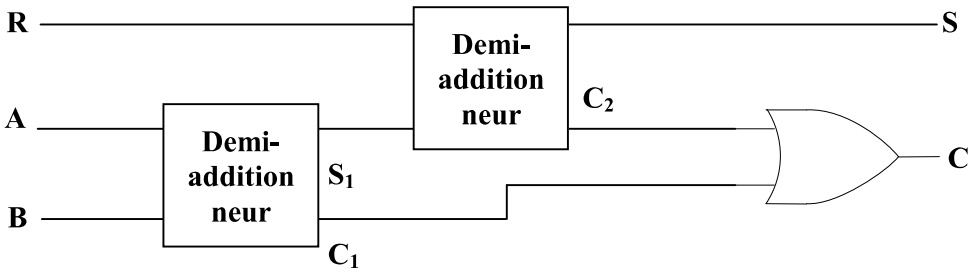


Figure 23 : Schéma fonctionnel d'un additionneur complet

Les entrées A et B représentent les bits à additionner et R le report de la retenue de l'addition des bits de poids inférieurs. La sortie S représente le résultat de la somme et C la retenue. La table de vérité de ce circuit est la suivante :

Table de vérité

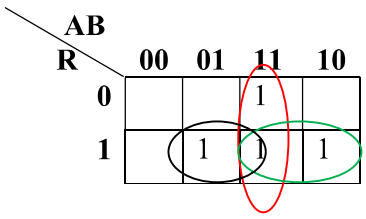
A	B	R	S	C
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Equation logiques de sorties

A partir de cette table nous pouvons écrire pour S et C les expressions booléennes suivantes :

$$\begin{cases} S = \overline{A}.\overline{B}.R + \overline{A}.B.\overline{R} + A.\overline{B}.\overline{R} + A.B.R \\ C = \overline{A}.B.R + A.\overline{B}.R + A.B.\overline{R} + A.B.R \end{cases}$$

Nous pouvons simplifier l'expression de C en utilisant un tableau de Karnaugh :



Nous en déduisons :

C = A B + A R + B R

Le bit de carry est égal à 1 si au moins deux des entrées sont à 1. D'autre part, nous pouvons remarquer qu'intervertir les 0 et les 1 dans la table 2 revient à permuter les lignes 1 et 8, 2 et 7, 3 et 6, 4 et 5. La table de vérité reste globalement invariante par inversion des entrées et des sorties, nous avons donc :

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

A partir de cette relation, qui peut également être démontrée en appliquant l'algèbre de Boole, nous pouvons écrire :

$$\begin{cases} A.\overline{C} = A.\overline{B}.\overline{R} \\ B.\overline{C} = \overline{A}.B.\overline{R} \\ R.\overline{C} = \overline{A}.\overline{B}.R \end{cases} \Rightarrow (A+B+C).\overline{C} = A.\overline{B}.\overline{R} + \overline{A}.B.\overline{R} + \overline{A}.\overline{B}.R$$

Ce qui nous permet de réécrire l'expression de S :

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

La figure 24 donne un exemple de réalisation d'un additionneur 1 bit basé sur deux portes AOI (AND OR INVERT), c'est-à-dire un ensemble de portes ET suivies d'une porte NON-OU.

Logigramme

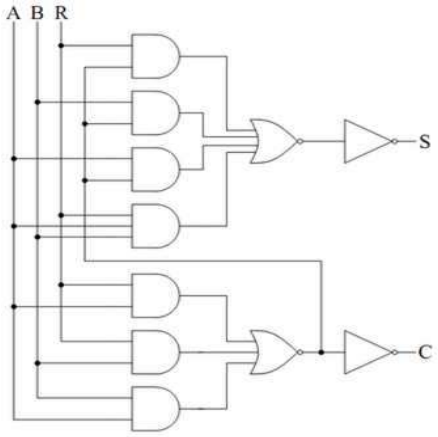


Figure 24: Logigramme d'un additionneur complet

2. Soustraction

a. Demi-soustracteur

La table de vérité pour un demi-soustracteur (ne tenant pas compte d'une éventuelle retenue provenant des bits de poids inférieurs) est la suivante :

Table de vérité

A	B	D	C
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Equations logiques de soties

Où D représente le résultat de la soustraction A − B et C la retenue. Nous en déduisons les expressions logiques définissant D et C :

$$\begin{cases} D = \overline{A}.B + A.\overline{B} = A \oplus B \\ C = \overline{A}.B \end{cases}$$

Logigramme

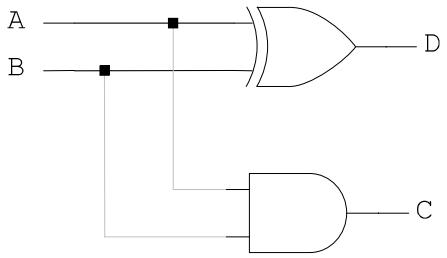


Figure 25 : Logigramme un demi soustracteur

Nous pourrions maintenant étudier un soustracteur prenant en compte la retenue. Nous allons plutôt tirer parti de certaines propriétés de la numération binaire pour traiter de la même manière l'addition et la soustraction.

b. Additionneur-soustracteur

Nous savons qu'avec un mot de n bits nous pouvons représenter un entier positif dont la valeur est comprise entre 0 et 2^{n-1}. Le complémentaire d'un mot de n bits est obtenu entre prenant le complément de chacun des n bits. Ainsi, si nous sommons un nombre et son complément nous obtenons un mot dont tous les bits sont à 1. C'est-à-dire :

$$A + \overline{A} = 2^n - 1$$

Attention : dans ce paragraphe le signe + représente l'opération addition et non la fonction logique OU. Nous pouvons encore écrire :

$$-A = \overline{A} + 1 - 2^n$$

Mais sur n bits l'entier 2^n est identique à 0 :

$$2^n \equiv 0\ (n\ bits)$$

C'est-à-dire qu'il est possible d'écrire un nombre entier négatif comme le "complément à 2" de sa valeur absolue : $-A = \overline{A} + 1$

Nous reviendrons sur les divers codages des entiers signés plus tard. Nous pouvons utiliser cette propriété pour écrire la soustraction de deux mots de n bits sous la forme suivante :

$$A - B = A + \overline{B} + 1 - 2^n \equiv A + \overline{B} + 1 (n\ bits)$$

Ce résultat conduit au schéma de principe présenté sur la figure 13 combinant les fonctions addition et soustraction. Celui-ci est basé sur l'emploi d'un additionneur n bits et d'un multiplexeur à deux lignes d'entrée. Nous étudierons ce type de circuit un peu plus loin dans ce chapitre. Selon le code opération O (0 pour une addition et 1 pour une soustraction) ce multiplexeur permet de sélectionner une des deux entrées, B ou son complémentaire. Le code opération est également injecté sur l'entrée report de retenue de l'additionneur. Pour simplifier

le schéma et éviter de représenter n lignes de connexion parallèles, on ne matérialise qu'une seule ligne. Celle-ci est barrée et accompagnée d'une valeur qui indique le nombre réel de connexions.

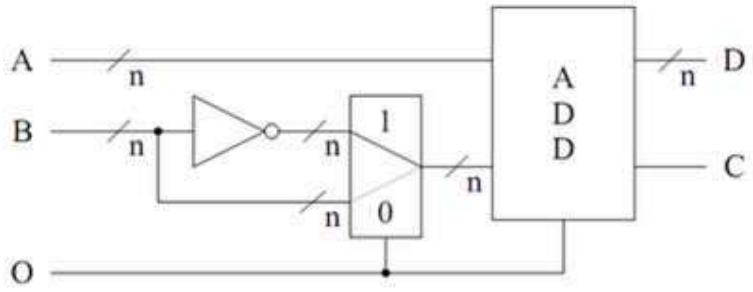


Figure 26: Additionneur soustracteur