الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique

Université Echahid Hamma Lakhder

Faculté des Sciences et de la technologie

Département de Génie électrique

جامعة الشهيد حمة لخضر
كلية العلوم و التكنولوجيا
قسم الهندسة الكهربائية

Support de cours pour la Filière

2ème Année - Génie électrique

Logique Combinatoire et Séquentielle

Préparé Par : Dr GACEM Abdelmalek

TABLE DES MATIERES

RÉSUMÉ

Ce cours est destiné aux étudiants de deuxième année licence en électrotechnique. Il traite le programme du module de logique combinatoire et séquentielle.

Le premier chapitre aborde l'algèbre de Boole et la simplification des fonctions logiques. Il consiste à connaître les variables et fonctions logiques (OR, AND, NOR, NAND, XOR), les lois de l'algèbre de Boole, les théorèmes de De Morgan, ainsi que les notions de fonctions logiques complètes et incomplètes. Il traite également des différentes méthodes de représentation des fonctions logiques à travers les tables de vérité et les tables de Karnaugh, ainsi que de leur simplification en utilisant la méthode algébrique et celle de Karnaugh.

Le deuxième chapitre est consacré aux systèmes de numération et au codage de l'information. L'étudiant devra maîtriser la représentation des nombres à l'aide de différents codes (binaire, hexadécimal, DCB, binaire signé et non signé, etc.), le changement de base ou conversion, ainsi que les codes non pondérés tels que le code de Gray, les codes détecteurs et correcteurs d'erreurs, le code ASCII, etc. Il comprend aussi les opérations arithmétiques effectuées en code binaire.

Le troisième chapitre traite des circuits combinatoires transcodeurs. Il passe en revue les principaux types de circuits combinatoires, en présentant pour chacun une description générale, la liste des circuits intégrés disponibles, les modalités de mise en cascade. Il aborde également les circuits combinatoires de type aiguillage, définit les multiplexeurs et démultiplexeurs, explique leur principe de fonctionnement et leur mise en cascade. Par ailleurs, ce chapitre présente les circuits combinatoires de comparaison, avec les définitions de base, les circuits de comparaison à plusieurs bits, l'analyse des fiches techniques associées et la liste des circuits intégrés utilisables dans ces applications.

Le quatrième chapitre introduit les bascules en amorçant la notion de circuits séquentiels. Sont étudiés les différents types de bascules : RS, RST, D, Maître-Esclave, T et JK.

Enfin, le cinquième chapitre est consacré aux circuits compteurs, en présentant leur définition et leur classification, ainsi qu'aux circuits registres. On y définit ce qu'est un registre, on présente les registres classiques ainsi que les registres à décalage.

Mots clés :

Logique Combinatoire, Séquentielle, L'algèbre de Boole, Systèmes de Numération, Bascules, Compteurs et Registres.

ملخـــــــص

هذا المقرر موجه لطلبة السنة الثانية ليسانس تخصص الهندسة الكهربائية. يتناول محتوى برنامج مقياس المنطق التوافقي والتزامني.

الفصل الأول الجبر البولياني وتبسيط الدوال المنطقية. حيث يهدف إلى التعرف على المتغيرات والدوال المنطقية (OR, AND, NOR, NAND, XOR)، وقوانين الجبر البولياني، ونظريات دي مورغان، بالإضافة إلى مفاهيم الدوال المنطقية الكاملة والناقصة. كما يتطرق إلى مختلف طرق تمثيل الدوال المنطقية باستخدام جداول الحقيقة و كارنو، مع تبسيط هذه الدوال باستعمال كل من الطريقة الجبرية وطريقة كارنو.

أما الفصل الثاني، فهو مخصص لأنظمة الترقيد وترميز المعلومات. يجب على الطالب إتقان تمثيل الأعداد بمختلف الأنظمة (الثنائي، السادس عشر، DCB، الثنائي الموقّع وغير الموقّع، إلخ)، وتحويل القواعد أو الأنظمة، إلى جانب التعرف على الشفرات غير الموزونة مثل شفرةGray ، والشفرات الكاشفة والمصححة للأخطاء، وشفرةASCII ، وغيرها. كما يشمل الفصل العمليات الحسابية المنفذة باستخدام الكود الثنائي.

في الفصل الثالث، يتم تناول الدارات التوافقية من نوع .transcodeurs يستعرض هذا الفصل الأنواع الأساسية من الدارات التوافقية، مع تقديم وصف عام لكل نوع، وقائمة الدارات المتكاملة المتوفرة، وطرق الربط المتسلسل. كما يعالج أيضًا الدارات التوافقية من نوع الموجهات، حيث يعرّف المبدّلات (multiplexeurs) والموزعات(démultiplexeurs) ، ويشرح مبدأ عملها وطريقة توصيلها. إضافة إلى ذلك، يعرض الفصل دارات المقارنة، من خلال التعريفات الأساسية، ودارات المقارنة متعددة البتات، وتحليل البطاقات التقنية الخاصة بها، وقائمة الدارات المتكاملة الممكن استخدامها في هذه التطبيقات.

أما الفصل الرابع، فيقدّم المبدلات (Bascules) مع التمهيد لمفهوم الدارات التزامنية. ويتم التطرق إلى الأنواع المختلفة من المبدلاتRS : ، RST، D، Master-Slave، Tو.JK

وأخيرًا، يُخصص الفصل الخامس لدراسة دارات العدّادات(compteurs) ، من خلال تقديم تعريفها وتصنيفها، بالإضافة إلى دارات السجلات .(registres) حيث يتم تعريف ما هو السجل، وعرض الأنواع التقليدية من السجلات وكذلك سجلات الإزاحة.

الكلمات المفتاحية :
المنطق التوافقي، المنطق التزامني، الجبر البولياني، أنظمة الترقيم، المبدلات، العدادات، السجلات.

CHAPITRE I:

ALGEBRE DE BOOLE ET SIMPLIFICATION DES FONCTIONS LOGIQUES

I.1 Introduction

L'algèbre de Boole, ou calcul booléen, est la partie des mathématiques qui s'intéresse aux opérations et aux fonctions sur les variables logiques. Elle fut inventée par le mathématicien britannique George Boole (1815-1864). Aujourd'hui, l'algèbre de Boole trouve de nombreuses applications en informatique et dans la conception des circuits électroniques.

On appelle B l'ensemble constitué de deux élément s'appelés valeurs de vérité {FAUX,VRAI}. Cet ensemble est aussi noté B ={0, 1}.

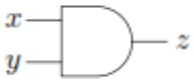
I.2 Portes logiques élémentaires

Une porte logique est un circuit ayant au moins une entrée et exactement une sortie. Les valeurs des entrées et de la sortie sont les valeurs logiques vrai (pour '1') et faux (pour '0'). Les porte logique divisées en deux types :

1. ***Les portes logiques élémentaires qui sont*** *NOT, AND et OR.*
2. ***Les portes logiques complexées qui sont*** *NAND, NOR, XOR et XNOR.*

I.2.1 Porte (AND)

Une porte AND peut avoir un nombre arbitraire d'entrées. Sa sortie vaut 1 si et seulement si toutes les entrées valent 1. Donc, la sortie vaut 0 si et seulement si au moins une des entrées vaut 0.

Symbole	***Table de vérité***	***Sortie***
	X Y Z 0 0 0 0 1 0 1 0 0 1 1 1	$Z = X \,.\, Y$

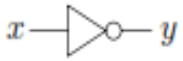
I.2.2 Porte (OR)

Comme la porte AND, la porte OR peut avoir un nombre arbitraire d'entrées. La sortie vaut 1 si et seulement si au moins une des entrées vaut 1. Autrement dit, la sortie vaut 0 si et seulement si toutes les entrées valent 0.

Symbole	***Table de vérité***	***Sortie***
	X Y Z 0 0 0 0 1 1 1 0 1 1 1 1	$Z = X + Y$

I.2.3 Porte (NOT)

La porte NOT ou (inverseur) a exactement une entrée et une sortie. Sa sortie vaut 1 si et seulement si l'entrée vaut 0. Sinon la sortie vaut 0. Autrement dit, la valeur de la sortie est exactement l'inverse de la valeur de l'entrée.

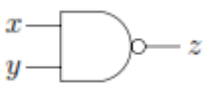
Symbole	***Table de vérité***	***Sortie***
	X Y 0 1 0 0	$Y = \bar{X}$

I.3 Portes logiques complexes

Souvent, il est pratique de combiner plusieurs fonctions élémentaires dans une seule porte plus complexe, par exemple afin de conserver l'espace de dessin dans un diagramme de circuits. Dans cette section, nous présentons quelques portes complexes fréquemment utilisées.

I.3.1 Porte (NAND)

La porte NAND est une porte AND avec un inverseur sur la sortie.

Symbole	***Table de vérité***	***Sortie***
	X Y Z 0 0 1 0 1 1 1 0 1 1 1 0	$Z = \overline{X.Y}$

I.3.2 Porte (NOR)

La porte NOR est une porte OR avec un inverseur sur la sortie.

Symbole ***Table de vérité*** ***Sortie***

X	Y	Z
0	0	1
0	1	0
1	0	0
1	1	0

$$Z = \overline{X + Y}$$

I.3.3 Porte (XOR)

Elle peut avoir un nombre arbitraire d'entrées. Sa sortie vaut 1 si et seulement si exactement une des entrées vaut 1. Sinon la sortie vaut 0.

Symbole ***Table de vérité*** ***Sortie***

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	0

$$Z = \bar{X}.Y + X.\bar{Y} = X \oplus Y$$

I.3.3 Porte (NXOR)

La porte NXOR est une porte XOR avec un inverseur sur la sortie.

Symbole ***Table de vérité*** ***Sortie***

X	Y	Z
0	0	1
0	1	0
1	0	0
1	1	1

$$Z = \bar{X}.\bar{Y} + X.Y = \overline{X \oplus Y}$$

I.4 Propriété l'algèbre de boole

<u>Loi de commutativité :</u>

Les opérations de la somme et le produit sont commutativités.

$$A\,.B = B\,.\,A$$

$$A\, +\, B = B\, +\, A$$

Loi d'associativité :

Les opérations de la somme et le produit sont associativités.

$$(A\,.\,B)\,.\,C = A\,.\,(B\,.\,C)$$

$$(A\ +\ B)\ +\ C = A\ +\ (B\ +\ C)$$

Loi de distributivité :

Chaque des opérations de la somme et le produit est distributive sur l'autre :

$$A + (B\,.\,C) = (A + B)\,.\,(A + C)$$

$$A\,.\,(\,B\ +\ C) = (A\,.\ B)\ + (A\,.\ C)$$

Loi d'impotence :

Le résultat d'une opération entre une variable A et elle-même est égal à cette variable.

$$A\,.\,A = A$$

$$A\ +\ A = A$$

Loi de complémentarité :

Le résultat d'une opération (.) du complémentarité entre une variable A est égale à 0.

$$A\,.\,\bar{A} = 0$$

$$A\ +\ \overline{A} = 1$$

Loi de la double négation :

Le complément du complément d'une variable A est égale à cette variable.

$$\bar{\bar{A}} = A$$

Loi de l'élément neutre :

A chaque opérateur correspond un élément neutre qui, lorsqu'il est opéré avec une variable quelconque A, donne un résultat identique à cette variable.

$$A\,.\,1 = A \text{ (1 élément neutre du AND)}$$

$$A+ 0=A \text{ (0 élément neutre du OR)}$$

Loi de l'absorption :

A chaque opérateur correspond un élément nul qui, lorsqu'il est opéré avec une variable quelconque A, donne un résultat identique à cet élément nul.

$$A\,.\,0 = 0$$

$$A\ +\ 1 = 1$$

I.5 Théorème de Morgan

Théorème :1

Le complémentaire du OR (Somme) des variables est égal au AND (Produit) des complémentaires de chaque variable non (A ou B) = (non A) et (non B).

Exemple :

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

Théorème :2

Le complémentaire du AND (Produit) des variables est égal au OR (Somme) des complémentaires de chaque variable. non (A et B) = (non A) ou (non B).

Exemple :

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

I.6 Dualité

Deux expressions se correspondent par dualité si l'on obtient l'une en changeant dans l'autre, les "AND" par des "OR", les "OR" par des "AND", les "1" par des "0" et les "0" par des "1". La notion de duale intervient dans l'étude des montages CMOS.

Exemple :

$$A . (A + B) \; par \; l'expression \; duale \; A + (A . B) = A$$

$$A + 0 = A \; par \; l'expression \; duale \; A . 1 = A.$$

I.7 Fonction logique

Les fonctions logiques « combinatoires », bases du calcul booléen, qui résultent de l'analyse combinatoire des variations des grandeurs d'entrées uniquement et peut avoir dans la sortie deux états (0 et 1).

Une fonction booléenne peut être défini par :

1. Sa table de vérité
2. Son expression
3. Sa table de Karnaugh
4. Son logigramme

I.7.1 Définition par sa table de vérité :

Les fonctions de « n » variables présentent 2^n résultats. Donc elle sera décrite par une T/V de 2^n lignes. Chaque ligne représente la valeur de la fonction par une combinaison binaire de « n » variables.

Exemple :

Soit $f(a, b, c)$ est une fonction booléenne qui prend zéro si la majorité des variables vaut zéro, et 'un' si inversement.

3 var$\rightarrow 2^n = 2^3 = 8$ lignes

Table de vérité (3 variable)

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

Alors : $f(a,b,c) = \bar{a}bc + a\bar{b}c + ab\bar{c} + abc$

I.7.2 Définition par expression

Il existe une notation pratique pour représenter une fonction logique :

1. On spécifié toutes les combinaisons d'entrées qui fournissent un « 1 ».
2. Par convention on place une barre horizontale sur la variable qui vaut un zéro «0».
3. L'absence de cette barre signifie qu'elle vaut un « 1 ».

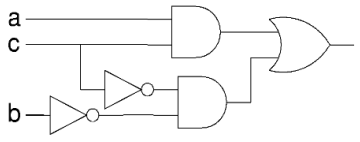
Exemple :

$$\bar{A}BC + A\bar{B}C + \bar{A}\bar{B}\bar{C} + ABC$$

I.7.3 Définition par logigramme

Connaissons les symboles logiques, on peut construire un logigramme.

Exemple :



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

I.8 Les formes canoniques

Une fonction logique exprimée dans une forme canonique est constituée d'un ou plusieurs termes canoniques.

I.8.1 Terme canonique

Un terme algébrique est dit canonique s'il contient une occurrence de chacune des variables impliquées dans la forme. (occurrences de la variable a sont $a\ ou\ \bar{a}$).

I.8.2 Forme canonique

La première forme canonique ou forme disjonctive (Somme des produits), chacun des termes est constitué par une combinaison de toutes les variables c'est un Minterme. La première forme canonique est une réunion de Mintermes.

Minterme mi est l'un quelconque des produits des n variables de la fonction.

Exemple :

A	B	Minterme
0	0	$m_0(\bar{A}\bar{B})$
0	1	$m_1(\bar{A}B)$
1	0	$m_2(A\bar{B})$
1	1	$m_3(AB)$

Deuxième forme canonique ou forme canonique conjonctive (Produits des somme), chacun des termes est constitué par une combinaison de toutes les variables c'est un Maxterme.

Maxterme Mi est l'un quelconque des sommes des n variables impliquées.

Exemple :

A	B	Maxterme
0	0	$m_0(A+B)$
0	1	$m_1(A+\bar{B})$
1	0	$m_2(\bar{A}+B)$
1	1	$m_3(\bar{A}+\bar{B})$

Une fonction booléenne peut être exprimée sous forme algébrique à partir de sa table de vérité:

1. Elle est égale à la somme des Mintermes pour lesquels la fonction vaut 1.
2. Elle est égale au produit des Maxtermes pour lesquels la fonction vaut 0.

I.8.3 Forme canonique NAND

Elle est déduite de la première forme par utilisation du théorème de De Morgan et d'involution, afin d'exprimer la fonction à l'aide des portes NAND uniquement.

Exemple :

$$f(A,B,C) = \bar{A} \cdot B \cdot C + A \cdot \bar{B} \cdot C + A \cdot B \cdot \bar{C} + A \cdot B \cdot C$$

$$= \overline{\overline{\bar{A} \cdot B \cdot C + A \cdot \bar{B} \cdot C + A \cdot B \cdot \bar{C} + A \cdot B \cdot C}} \quad Involution$$

$$= \overline{\overline{\bar{A} \cdot B \cdot C}} \cdot \overline{\overline{A \cdot \bar{B} \cdot C}} \cdot \overline{\overline{A \cdot B \cdot \bar{C}}} \cdot \overline{\overline{A \cdot B \cdot C}} \quad De\ Morgan$$

I.8.4 Forme canonique N0R

Elle est déduite de la deuxième forme par utilisation du théorème de De Morgan et d'involution, afin d'exprimer la fonction à l'aide des portes NOR seulement.

Exemple :

$$f(A.B) = (A + B)\,(\bar{A} + \bar{B})$$

$$= \overline{\overline{(A + B)\,(\bar{A} + \bar{B})}} \quad Involution$$

$$= \overline{\overline{(A + B)}} + \overline{\overline{(\bar{A} + \bar{B})}} \quad De\ Morgan$$

I.9 Simplification des fonctions logiques

I.9.1 Méthode algébrique

Exemple :

Soit à démontrer les ´équations logiques suivantes par la méthode de simplification algébrique :

- $A \cdot B + A \cdot \bar{B} = A \cdot (B + \bar{B})$
  $= A \cdot 1\ car\ B + \bar{B} = 1$
  $= A\ puisque\ A \cdot 1 = A$
- $A + A \cdot B = A \cdot (1 + B)$
  $= A \cdot 1\ car\ 1 + B = 1$
  $= A\ puisque\ A \cdot 1 = A$

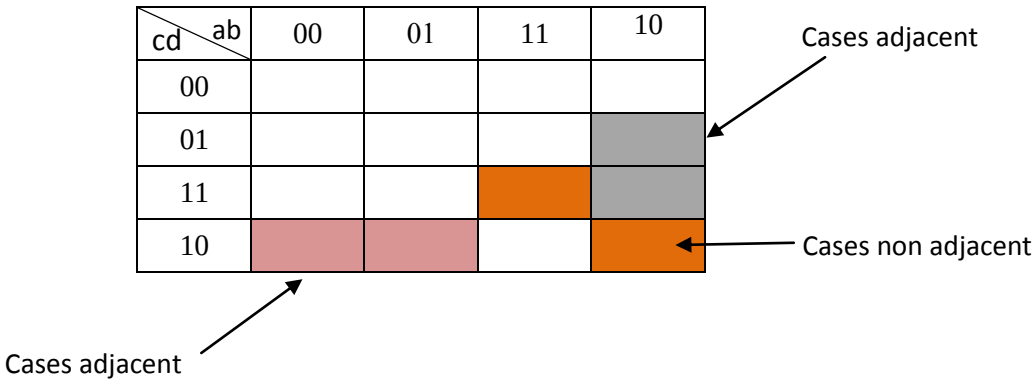
- $A + A \cdot B = A + A \cdot B + A \cdot B\ car\ A = A + A \cdot B$
  $= A + B \cdot (A + A)$
  $= A + B \cdot 1$
  $= A + B$

I.9.2 Méthode graphique (méthode de Karnaugh)

Cases adjacentes :

Deux case sont adjacentes lorsqu'elle sont situées côte à côte, que ce soit a l'horizontale ou à la vertical. De plus, une seule variable doit changer d'état pour que cases soient considérées comme adjacentes.

Exemple :



Règles de regroupement :

Les regroupement des cases adjacentes permet de réduire une équation logique le plus simplement possible pour ce faire, certaines règles doivent être respectée.

Règle 1: Les regroupement des cases adjacentes doit se faire par puissance de deux 2^0, 2^1, 2^2,…etc.

Règle 2: la longue et la hauteur de groupement doivent être de puissance de deux.

Règle 3: Les regroupement des quatre cases ou plus doivent être dispose symétrique par rapport à l'une des ligne de la table Karnaugh.

Règle 4: Les cases des extrémités ou quatre cases des quatre coins de table Karnaugh peuvent être regroupées.

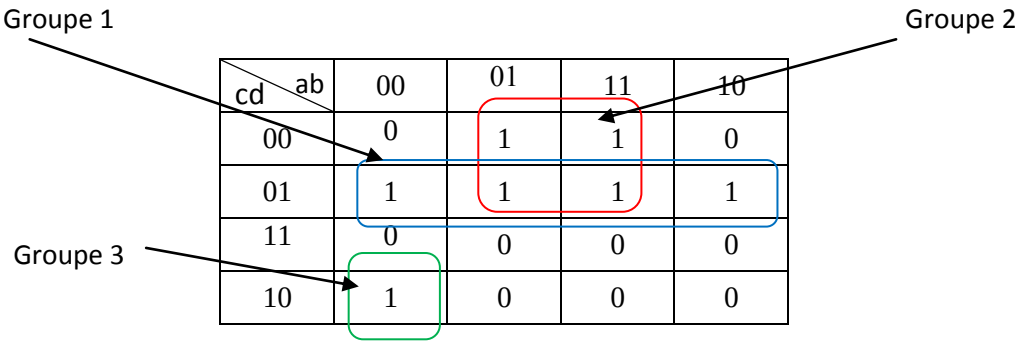
Ecriture des équation à partir de regroupement (somme de produit) :

chaque regroupement de 1 donne le produit logique des variable d'entrée qui n'ont pas changé d'état. L'ensemble de ces regroupements changé est un somme logique.

Règle : $B = 1 \rightarrow$ on la représente par B

$B = 0 \rightarrow$ on la représente par $\bar{B}$

Exemple :



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

Groupe 1:

$$\begin{cases} \text{a et b change d'état} \\ \text{et} \\ \text{c} = 0 \text{ et d} = 1 \text{ ne change pas d'état} \end{cases} \rightarrow \quad \bar{\mathbf{c}} \,.\, \mathbf{d}$$

Groupe 2:

$$\begin{cases} \text{a et d change d'état} \\ \text{et} \\ \text{b} = 1 \text{ et c} = 1 \text{ ne change pas d'état} \end{cases} \rightarrow \quad \mathbf{b} \,.\, \bar{\mathbf{c}}$$

Groupe 3:

$$\begin{cases} \text{a} = 0 \text{ et b} = 0 \text{ ne change pas d'état} \\ \text{et} \\ \text{c} = 1 \text{ et d} = 0 \text{ ne change pas d'état} \end{cases} \rightarrow \quad \bar{\boldsymbol{a}} \,.\, \bar{\boldsymbol{b}} \,.\, \boldsymbol{c} \,.\, \bar{\boldsymbol{d}}$$

Donc l'équation final $\boldsymbol{f} = \bar{\mathbf{c}} \,.\, \mathbf{d} + \mathbf{b} \,.\, \bar{\mathbf{c}} + \bar{\boldsymbol{a}} \,.\, \bar{\boldsymbol{b}} \,.\, \boldsymbol{c} \,.\, \bar{\boldsymbol{d}}$

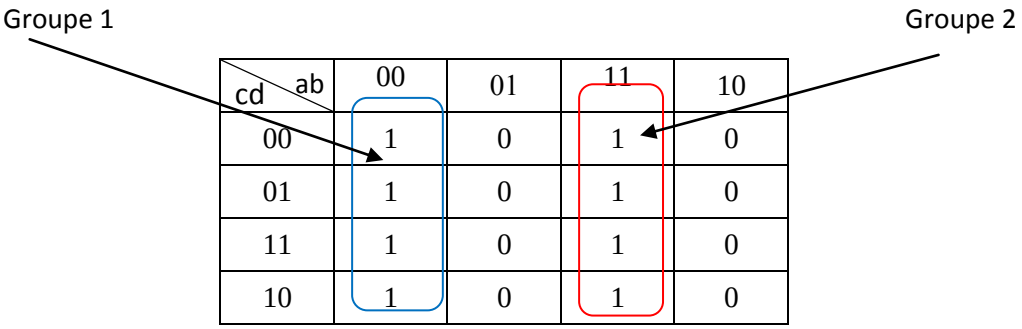
Ecriture des équation à partir de regroupement (produit de somme) :

chaque regroupement de 0 donne la somme logique des variables d'entrées qui n'ont pas changé d'état. L'ensemble de ces regroupements est un produit logique.

Règle : $B = 1$ → on la représente par $\bar{B}$

$B = 0$ → on la représente par B

Exemple :



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

Groupe 1:

$$\begin{cases} \text{a et b ne change pas d'état} \\ \text{et} \\ \text{c et d change d'état} \end{cases} \rightarrow \boldsymbol{a + b}$$

Groupe 2:

$$\begin{cases} \text{a et b ne change pas d'état} \\ \text{et} \\ \text{c et d change d'état} \end{cases} \rightarrow \boldsymbol{\overline{a} + \overline{b}}$$

Donc l'équation final $\boldsymbol{f = (a + b) \,.\, (\overline{a} + \overline{b})}$

CHAPITRE II:

SYSTEMES DE NUMERATION ET CODAGE DE L'INFORMATION

II.1 Introduction

Un système de numération est un système de comptage et de représentation des nombres. Il est généralement pondéré. La définition d'un système de numération pondéré repose sur trois notions [1] :

La base : est un entier naturel supérieur ou égal à 2, désignant le type de système de numération. Et aussi le nombre de symboles (chiffres) que comporte ce système.

Les bits (digits ou chiffres) : ce sont des symboles, tous différents d'une base B donnée.

Le poids : il est déterminé en fonction de la position de chaque bit lorsqu'on écrit un nombre dans une base donnée.

Exemple:

$N = (a_{n-1}a_{n-2}a_{n-3} \ldots\ldots a_0)_b$

Où

a_0 correspondant au poids le plus faible i=0.

a_{n-1} correspondant au poids le plus fort i=n

II.2 Différents systèmes de numération

Il existe différents systèmes de numération utilisés à travers le monde, mais les plus couramment utilisés sont:

1. Système décimal.
2. Système binaire.
3. Système octal.
4. Système hexadécimal.

II.2.1 Système décimal

C'est le système de numération usuel dans la vie quotidienne. Dans ce système , tout nombre N est exprimé à partir des dix chiffres : 0, 1, 2, ..., 9. On dit alors que la base de numération est $B = 10$.

II.2.2 Système binaire

Un système binaire est un système de numération qui utilise seulement deux chiffres : 0 et 1. On dit alors que la base de numération est $B = 2$. Il est à la base du fonctionnement des ordinateurs et des systèmes numériques

II.2.3 Système octal

Le système octal est un système de numération en base 8, ce qui signifie qu'il utilise huit chiffres : 0,1,2,3,4,5,6,7

II.2.4 Système hexadécimal

Le système hexadécimal est un système de numération en base 16, ce qui signifie qu'il utilise 16 symboles distincts pour représenter les nombres. Ces symboles sont : $0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F$

Où : $A = 10, B = 11, C = 12, D = 13, E = 14, F = 15$

II.3 Conversions entre bases

La conversion entre bases est un processus permettant de représenter un nombre écrit dans un système de numération donné (par exemple, décimal) en un autre système (par exemple, binaire, hexadécimal, etc.).

II.3.1 Conversion binaire-décimal

Soit $N = (a_n a_{n-1} a_{n-2} a_{n-3} \ldots\ldots a_0)_b$

Pour avoir la représentation en décimal du nombre N exprimé dans une base B quelconque, il suffit d'effectuer le calcul suivant :

$$(N)_{10} = (a_n . b^n + a_{n-1} . b^{n-1} + \cdots .. a_1 . b^1 + a_0 . b^0)_b$$

La formule générale s'écrit comme suit : $(N)_{10} = \sum_{i=0}^{n} a_i . b^i$

Où : (i) étant le rang du chiffre ai.

Exemple:

$(11{,}01101)_2 = (?)_{10}$

$11{,}01101 = 1x2^1 + 1x2^0 + 0x2^{-1} + 1x2^{-2} + 1x2^{-3} + 0x2^{-4} + 1x2^{-5}$

$= 2 + 1 + 0 + 0{,}25 + 0{,}125 + 0{,}03125$

$= 3{,}40625$

$(11{,}01101)_2 = (3{,}40625)_{10}$

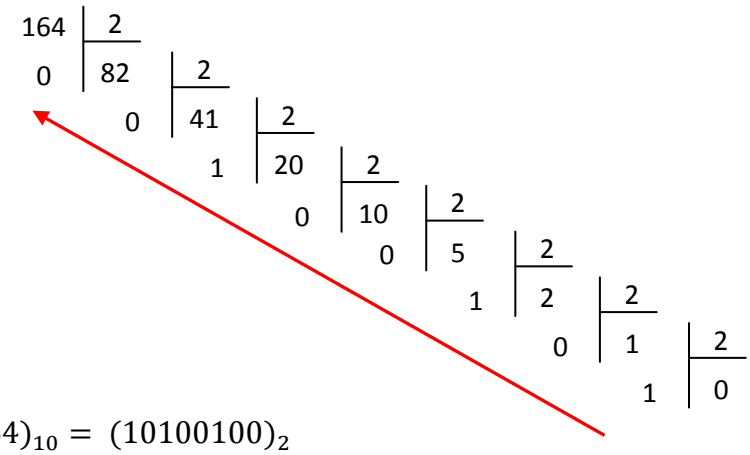
II.3.2 Conversion décimal - binaire

La méthode des divisions successives par deux: On divise le nombre décimal par la base $B = 2$, puis le quotient obtenu de nouveau par $B = 2$ jusqu'à ce qu'il devienne nul. Les restes successifs lus de bas en haut représentent le nombre dans la base $B = 2$.

Exemple:

$(116)_{10} = (?)_2$

On va utiliser la méthode des divisions successives par 2 :



$Donc\ (164)_{10} = (10100100)_2$

II.3.3 Conversion fractions décimal - binaire

La méthode des multiplications successives par deux:

Exemple:

$(0{,}285)_{10} = (\,?\,)_2$

$0,285x2 = 0{,}75$ $\qquad$ $0{,}28x2 = 0{,}56$

$0{,}75x2 = 1{,}14$ $\qquad$ $0{,}56x2 = 1\,,12$

$0{,}14x2 = 0{,}28$ $\qquad$ $0{,}12x2 = 0{,}25$

Donc : $(0{,}6425)_{10} = (0{,}1010010)_2$

Ce processus continu jusqu'à la précision souhaitée.

II.3.4 Conversion binaire-hexadécimal

Le plus grand caractère en hexadécimal est F, son équivalent binaire est $(1111)_2$, il est présenté sur 4 bits donc il est facile de traiter les données par groupe de 4 bits. On divise le nombre binaire par tranche de 4 bits an allant vers la gauche pour la partie entière et vers la droite pour la partie fractionnaire, puis on substitue à chaque groupe son équivalent hexadécimal.

Exemple:

$$(\,1110011101{,}01110001)_2 = (\,39D, 71\,)_{H\ ou\ 16}$$

II.3.5 Conversion hexadécimal - binaire

$$(7A1F, B46)_H = (\,0111101000011111{,}101101000110)_2$$

II.3.6 Conversion hexadécimal - décimal

$(EA7)_H = (\,?\,)_{10}$

$= Ex16^2 + Ax16^1 + 7x16^0$

$= 14x256 + 10x16 + 7x1$

$= 3751$

Donc:

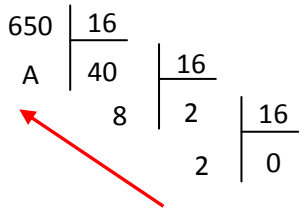
$(EA7)_H = (3751)_{10}$

II.3.7 Conversion décimal -hexadécimal

Là encore, il est question d'appliquer la méthode des divisions successives sue la base 16.

Exemple:

$(650)_{10} = (?)_H$



Donc:

$(650)_{10} = (28A)_H$

II.3.8 Conversion octal-décimal:

$(134)_8 = (?)_{10}$

$= 1x82 + 3x81 + 4x80$

$= 64 + 24 + 4$

$= 92$

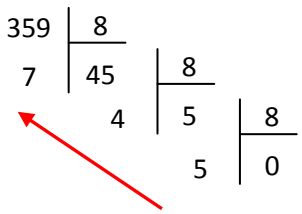
Donc : $(134)_8 = (92)_{10}$

II.3.9 Conversion décimal- octal:

Là encore, il est question d'appliquer la méthode des divisions successives sue la base 8.

Exemple:

$(359)_{10} = (?)_8$



$(359)_{10} = (547)_8$

II.3.10 Conversion octal-binaire:

Le plus grand caractère en octal est 7, son équivalent binaire est $(111)_2$, il est présenté sur 3 bits donc il est facile de traiter les données par groupe de 3 bits. On divise le nombre binaire par tranche de 3 bits an allant vers la gauche pour la partie entière et vers la droite pour la partie fractionnaire, puis on substitue à chaque groupe son équivalent octal.

Exemple:

$$(75{,}246)_8 = (111101{,}010100110)_2$$

II.3.11 Conversion binaire- octal:

$$(1110011101{,}011100001)_2 = (1635{,}341)_8$$

II.4 Les opérations arithmétiques binaires:

Comme tout autre système de numération, le système binaire permet d'effectuer les 4 opérations arithmétiques élémentaires : l'addition, la soustraction, la multiplication et la division.

II.4.1 L'addition :

L'addition binaire repose sur 4 règles :

$0 + 0 = 0$ $\quad\quad$ $0 + 1 = 1$

$1 + 0 = 1$ $\quad\quad$ $1 + 1 = 0$ $(avec\ une\ retenue\ r)$.

II.4.2 La soustraction:

La soustraction binaire se base sur les règles suivantes :

$0 - 0 = 0$ $\quad\quad$ $0 - 1 = 1$ $(avec\ une\ retenue\ r = 1)$.

$1 - 0 = 1$ $\quad\quad$ $1 - 1 = 0$

II.4.3 La multiplication:

Elle repose sur les règles suivantes :

$0\ x\ 0 = 0$ $\quad\quad$ $0\ x\ 1 = 0$

$1\ x\ 0 = 0$ $\quad\quad$ $1\ x\ 1 = 1$

II.4.4 La division binaire:

Elle repose sur les règles suivantes :

$0 / 0 = (cas\ indéterminés)$ $1 / 0 = (cas\ indéterminés)$

$0 / 1 = 0$ $1 / 1 =$

II.5 Complément à 1 et Complément à 2 d'un nombre binaire

Les compléments sont utilisés principalement pour représenter les nombres négatifs en binaire et effectuer des opérations arithmétiques.

II.5.1 Complément à 1:

Le complément à 1 d'un nombre binaire est obtenu en inversant tous ses bits (remplacer 0 par 1 et 1 par 0).

Exemple:

Nombre binaire : 101100

Complément à 1 : 010011

NB: Le complément à 1 est souvent utilisé dans la soustraction binaire et la représentation des nombres négatifs.

II.5.2 Complément à 2:

Le complément à 2 est obtenu en ajoutant 1 au complément à 1.

Exemple:

Nombre binaire : 101100

Complément à 1 : 010011

Ajouter 1 : $010011 + 1 = 010100$

NB: Le complément à 2 est la méthode la plus utilisée pour représenter les nombres négatifs dans un système binaire signé.

Exemple:

Convertir $(-13)_{10}$ en complément à 2 sur 8 bits

A- Représentation de $(13)_{10}$ en binaire sur 8 bits : 00001101

B- Complément à 1 : 11110010

C- Ajouter 1 : $11110010 + 1 = 11110011$

Résultat:

$(-13)10 = (11110011)$ (Complément à 2 sur 8 bits)

II.6 Codage de l'information :

II.6.1 Le code BCD :

Ce système n'utilise que 10 codes et permet d'effectuer facilement les conversions entre le décimal et le binaire. Le code BCD fournit une excellente interface aux systèmes numériques (les afficheurs numérique).

Dans ce code chaque chiffre décimal (de 0 à 9) est représenté par un code binaire de 4 bits.

Décimal	0	1	2	3	4	5	6	7	8	9
BCD	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

Les 6 combinaisons qui restent $(1010, 1011, 1100, 1101, 1110, 1111)$ ne sont pas valides dans le code BCD.

NB : Ne pas confondre le BCD avec le binaire.

Exemple:

$$(98)_{10} = (10011000)_{BCD} = (1100010)_2$$

$$(10000110)_{BCD} = (86)_{10}$$

II.6.2 Le code de Gray :

Ce code est non pondéré et ne convient pas aux calculs arithmétiques, sa caractéristique permet de passer d'un nombre au suivant par le changement d'un seul bit à la fois.

A- Conversion binaire-Gray :

Pour faire la conversion du binaire au code Gray, il faut suivre les règles suivantes :

1. le bit du gauche (MSB) du code Gray est le même que MSB du nombre binaire.
2. Ajouter le MSB du nombre binaire à son voisin immédiat pour obtenir le bit suivant du code Gray (la retenue n'est pas prise en considération).
3. Continuer l'addition des bits à leur voisin de droite jusqu'à atteindre le LSB.

Exemple:

$(0110)_2 = (0101)_{Gray}$

B- Conversion Gray- binaire :

Pour faire la conversion Gray-binaire, il faut suivre les règles suivantes :

1. Le MSB du nombre binaire est le même que le MSB du code Gray.
2. Ajouter le MSB du nombre binaire obtenu au voisin de droite immédiat du code Gray
3. Continuer les additions jusqu'à atteindre le LSB.

Exemple:

$(1001)_{Gray} = (1110)_2$

II.6.3 Le code ASCII :

ASCII est la contraction d'American Standard Code for Information Inter change. C'est un code utilisé dans la transmission d'informations alphanumériques entre un ordinateur et des dispositifs d'entrée/sortie externes (clavier, imprimante….). Il permet de coder 26 lettres minuscules, 26 lettres majuscules, 10 chiffres et environ 25 caractères spéciaux et signes. Chaque code comporte 7 bit.

Exemple:

Caractère	ASCII à 7 bits
M	1001101
blanc	0100000
+	0101011
,	0101001

II.6.4 Les codes détecteurs et correcteurs d'erreurs:

Le processus de transmission de données ne s'accomplit pas sans erreurs, même si les appareils modernes ont été étudiés pour réduire au minimum la probabilité d'erreur. Il existe plusieurs codes permettant la détection ou même la correction des erreurs.

A- **Codes détecteurs d'erreurs :**

L'une des méthodes les plus répondues pour la détection d'erreurs est la méthode de parité paire ou impaire. Le principe consiste à rajouter un bit de contrôle de parité à une donnée. Ce bit de parité est 1 ou 0 selon le nombre de 1 qu'il y a dans la donnée.

La méthode de contrôle de la parité paire :

On fixe le bit de parité pour que le nombre total de 1dans la représentation codée (en tenant compte du bit de parité) soit un nombre pair.

La méthode de contrôle de la parité impaire :

Est une technique tout à fait similaire à la méthode de la parité paire sauf que le nombre total de 1 (en tenant compte du bit de parité) soit impaire.

B- **Codes correcteurs d'erreurs :**

Ce sont des codes qui permettent la détection et la correction des erreurs. On peut citer par exemple le code double parité et le code Hamming.

CHAPITRE III:

CIRCUITS COMBINATOIRES PARTICULIERS

III.1 Introduction

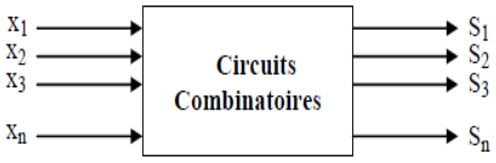
La transmission de données nécessite fréquemment des opérations de conversion, de transpostage et d'aiguillage. On utilise pour cela des circuits combinatoires. Pour réaliser un circuit logique combinatoire, le concepteur doit utiliser plusieurs portes logiques élémentaires. Pour faciliter sa tâche, 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 :

1. Les codeurs, les décodeurs et les transcodeurs.
2. Les multiplexeurs et les démultiplexeurs.
3. Les comparateurs.

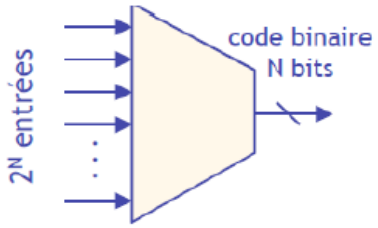
III.2 Définition

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.



III.3 Les Codeurs

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.



III.3.1 Codeur 4 voies d'entrées et 2 bits de sortie

Schéma:

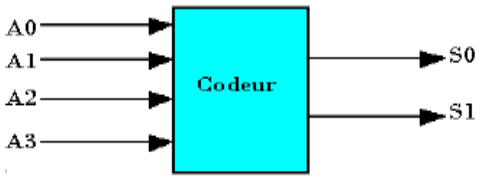


Table de vérité:

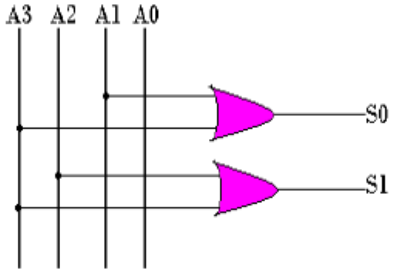
Entrées				Sorties	
Codage 1 parmi 2^n				Nombre binaire n bits	
A_3	A_2	A_1	A_0	S_1	S_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

Equation des sorties:

$S_0 = 1\ si\ (A_1 = 1)\ ou\ (A_3 = 1)\ \rightarrow\ S_1 = A_1 + A_3$

$S_1 = 1\ si\ (A_2 = 1)\ ou\ (A_3 = 1)\ \rightarrow\ S_1 = A_2 + A_3$

Logigramme :



Si nous activons simultanément les entrées A_1et A_2 du codeur ci-dessus, les sorties S_1 S_0 présente le nombre (1,1) 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' A_1 et A_2 sont activées simultanément S_1 S_0 sera égale à 10 qui représentent l'activation de A_0.

III.3.2 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			
A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_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	1	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

III.3.3 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.

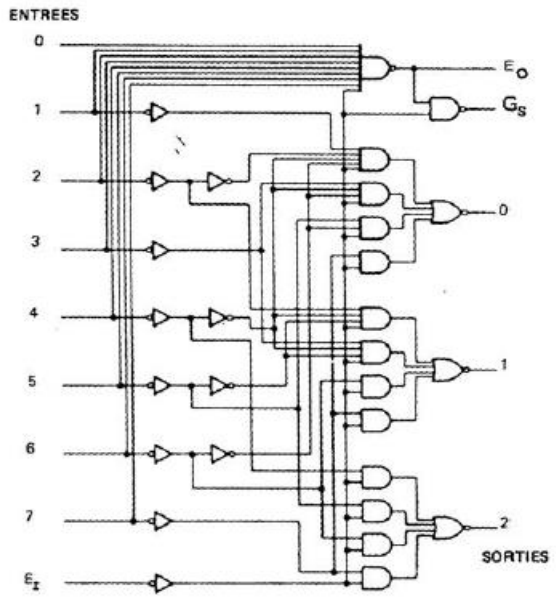
B- Codeur prioritaire à 3 bits 74148

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



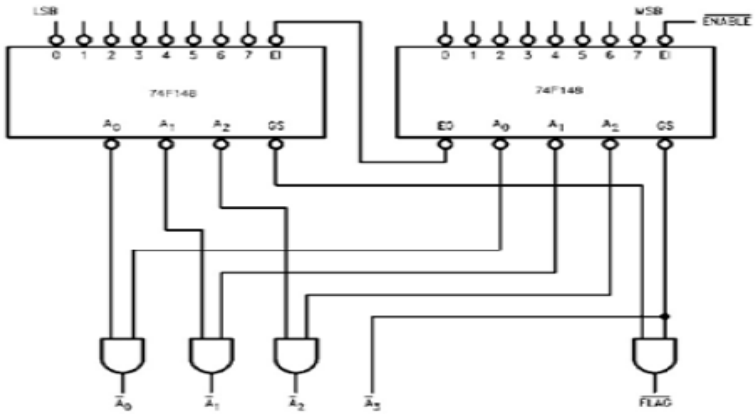
Ce codeur possède en plus des entres classiques du codeur de priorité, trois broches supplémentaires $\overline{E_0}$, $\overline{E_1}$ et $\overline{G_S}$. Le rôle de chacune de ces broches est décrit ainsi:

1. Si l'entrée $\overline{E_1} = 1$, alors le codeur n'est pas validé et les sorties $\overline{A_2} = \overline{A_1} = \overline{A_0} = \overline{E_0} = \overline{G_S}$ sont à 1 quelles que soient les entrés.
2. 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.
3. Si l'entrée $\overline{E_1} = 0$, et si toutes les entrés $\overline{I_i}$ sont à 1(pas d'informations sur les entrées), alors l sortie $\overline{E_0} = 0$ est à l'état bas.
4. 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

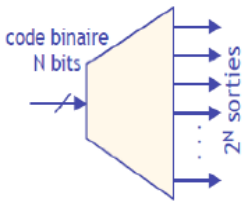
Pour réaliser le codage binaire dans un système à plus de huit entrées, on peut mettre plusieurs codeurs 74148 en cascade.

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



III.4 Les décodeurs

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



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

1. 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).

2. 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.

III.4.1 Décodeur 1 parmi 4

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

Schéma:

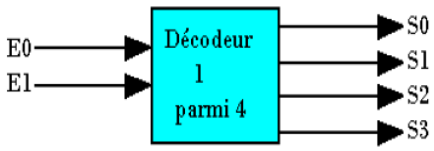


Table de vérité:

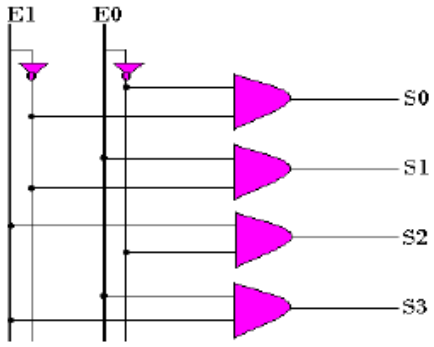
Entrées		Sorties			
Nombre binaire n bits		Codage 1 parmi 2^n			
E_1	E_0	S_3	S_2	S_1	S_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

Equation des sorties:

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

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

Logigramme :



NB: 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

III.4.2 Décodeur 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.

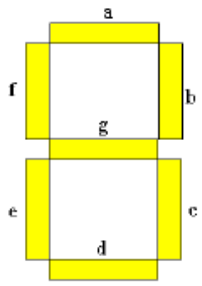


Table de vérité:

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	1	1	7
1	0	0	0	1	1	1	1	1	0	0	8
1	0	0	1	1	1	1	0	0	1	1	9

$$\bar{a} = a_0\,\overline{a_1}\,\overline{a_2 a_3} + a_2\,\overline{a_1}\,\overline{a_0 a_3} = \overline{a_1}\;\overline{a_3}(a_0 \oplus a_3)$$

$$\bar{b} = a_0\,\overline{a_1}\,a_2\,\overline{a_3} + \overline{a_0}\;a_1\,a_2\,\overline{a_3} = a_2\;\overline{a_3}(a_0 \oplus a_1)$$

$$\bar{c} = \overline{a_0}\;a_1\,\overline{a_2}\,\overline{a_3}$$

$$\bar{d} = a_0\,\overline{a_1}\,\overline{a_2}\,\overline{a_3} + \overline{a_1}\,a_2\,\overline{a_0 a_3} + a_0\;a_1\,a_2\,\overline{a_3}$$

$$\bar{e} = \overline{a_1}\,\overline{a_2}\,\overline{a_0 a_3} + \overline{a_0}\;a_1\,\overline{a_2 a_3} + \overline{a_0}\;a_1\,a_2\,\overline{a_3} + a_0\;a_1\,a_2\,\overline{a_3}$$

$$\bar{f} = a_0\overline{a_1}\,\overline{a_2 a_3} + \overline{a_0}\;a_1\,\overline{a_2}\,\overline{a_3} + a_0\;a_1\,\overline{a_2 a_3} + a_0 a_1 a_2 a_3$$

$$\bar{g} = \overline{a_0}\,\overline{a_1}\,\overline{a_2 a_3} + a_0\,\overline{a_1}\,\overline{a_2}\,\overline{a_3} + a_0\;a_1\,a_2\,\overline{a_3}$$

III.4.3 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.

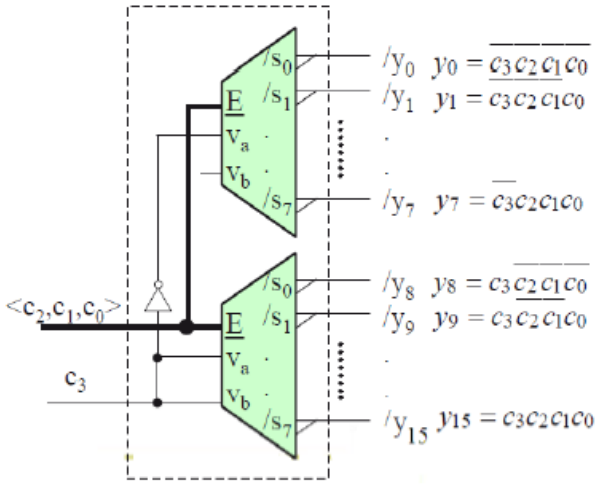
Table de vérité:

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- Décodeur 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.



III.5 Les transcodeurs

III.5.1 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.

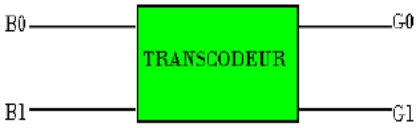


Table de vérité:

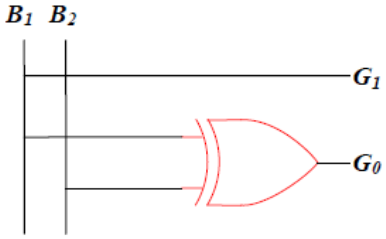
Entrées		Sorties	
Nombre binaire		Nombre Gray	
B_1	B_0	G_1	G_0
0	1	0	0
1	0	0	1
1	0	1	1
1	1	1	0

Equations des sorties:

$$G_0 = B_1\overline{B_0} + B_1B_0 = B_1$$

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

Logigramme :



III.5.2 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 à 9 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.

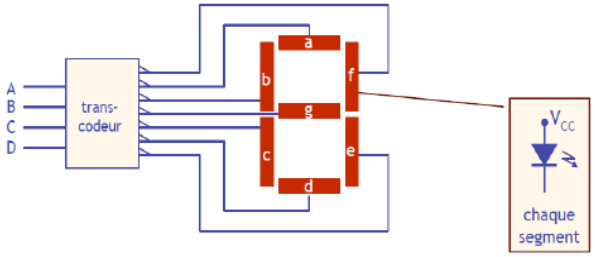


Table de vérité:

Entrées					Sorties						
Affichage	A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	0	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	1	1
8	1	0	0	0	1	1	1	1	1	0	0
9	1	0	0	1	1	1	1	0	0	1	1

Equations des sorties:

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

III.6 Multiplexeur

Le multiplexeur est un sélecteur de données qui permet d'aiguiller à l'aide des entrées de sélection $(C_0, C_1, \dots, C_n)$ des données de provenances diverses $(E_0, E_1, \dots, E_n)$ vers une seule sortie S. L'entrée sélectionnée par son adresse.

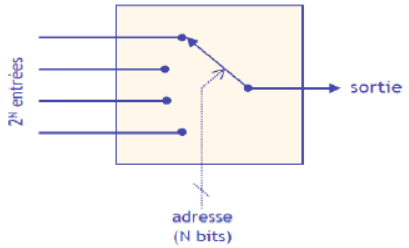


Table de vérité:

Décimal	C_2	C_1	C_0	S
0	0	0	0	E_0
1	0	0	1	E_1
2	0	1	0	E_2
3	0	1	1	E_3
$2^n - 1$	1	1	1	E_n

III.6.1 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, C_1

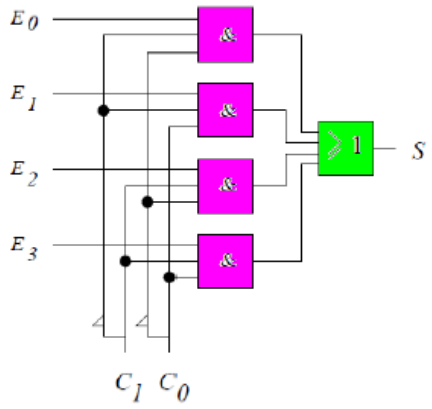
Table de vérité:

Décimal	C_1	C_0	S
0	0	0	E_0
1	0	1	E_1
2	1	0	E_2
3	1	1	E_3

Equation de sortie

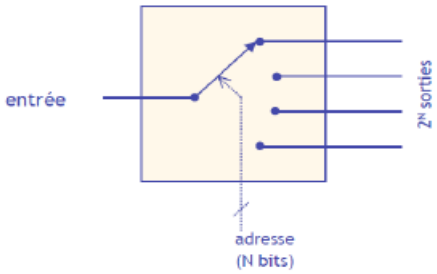
$$G_0 = \overline{C_1}\,\overline{C_0}E_0 + \overline{C_1}\,C_0E_1 + C_1\,\overline{C_0}E_2 + C_1\,C_0E_0$$

Logigramme :



III.7 Démultiplexeur

Le démultiplexeur réalise l'inverse d'un multiplexeur : 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 $(C_0, C_1, \dots, C_n)$ et 2^n sorties $(S_0, S_1, \dots, S_{2^n})$



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

Table de vérité:

Décimal	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
$2^n - 1$	1	1	1	*0*	0		E

NB : Dans certains cas on trouve :

1. S_i à 1 lorsqu'elles ne sont pas sélectionnées à la place de 0
2. $\bar{E}$ à la palace de E dans les S_i, lorsqu'elles sont sélectionnés.

Démultiplexeur en circuit intégré

1. Démultiplexeur (décodeur) 8-vers-1 :74138
2. Décodeur /démultiplexeur : 74154

III.8 Comparateur

C'est un circuit permettant de comparer 2 mots de n bits chacun en indiquant sur ses sorties S1, S2 ou S3 si le premier mot est égal, plus grand ou plus que le second.

III.8.1 Comparateur 2 bits

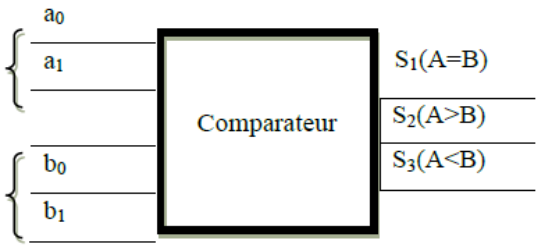


Table de vérité:

b_1	b_0	a_1	a_0	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

$$S_1 = \overline{a_0}\ \overline{a_1}\ \overline{b_0}\ \overline{b_1} + a_0\ \overline{a_1}\ b_0\ \overline{b_1} + \overline{a_0}\ a_1\ \overline{b_0}\ b_1 + a_0\ a_1\ b_0\ b_1$$

$$= \overline{(a_0 \oplus b_0) + (a_1 \oplus b_1)}$$

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

$$= a_1\ \overline{b_1} + a_0\ \overline{b_0}\ \overline{b_1} + a_0\ a_1\ \overline{b_0}$$

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

Comparateur en circuit intégrés

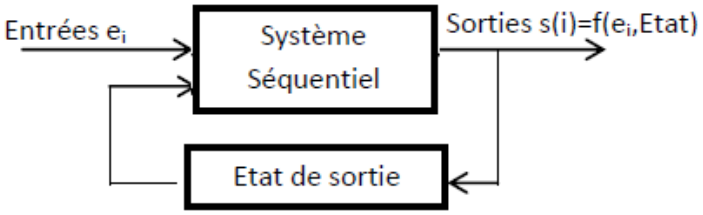
1. 74 85 TTL (8bits)
2. 54 85 TTL (8bits)
3. 40 05 CMOS (8bits)

CHAPITRE IV:

CIRCUITS LOGIQUES SEQUENTIELS

IV.1 Introduction

Un circuit séquentiel est un système dont les sorties à l'instant (t) dépendent à la fois de L'état des entrées et aussi de l'état précédent de la sortie. On peut représenter un système séquentiel par le schéma suivant



Un circuit séquentiel est composé d'un circuit combinatoire et d'éléments mémoire appelés BASCULES. On distingue deux types de circuits séquentiels :

1. Les circuits séquentiels synchrones
2. Les circuits séquentiels asynchrones

IV.2 Définition d'une bascule

Une bascule est un élément capable de stocker un bit et le restituer au temps voulu. Les bascules sont utilisées pour réaliser des circuits de mémorisation.

On distingue quatre bascules de base :

1. La bascule R-S
2. La bascule J-K
3. La bascule T
4. La bascule D

Dans ce qui suit, on va étudier ces éléments de base (bascule), l'une après l'autre, dans le but de comprendre le principe de leur fonctionnement et leurs points de différence.

IV.3 Les bascules de base

IV.3.1 La bascule R-S

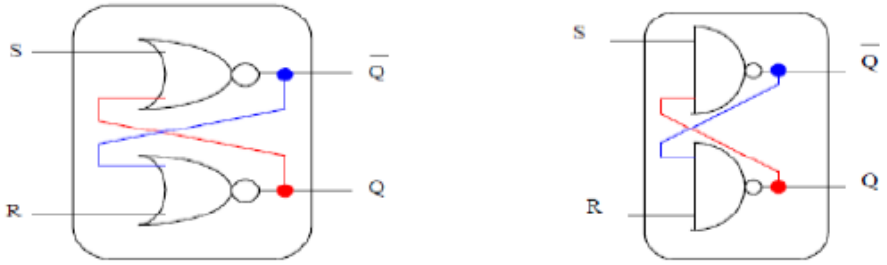
La bascule R-S est un circuit formé de deux portes logiques NOR ou NAND. Ce circuit possède deux entrées et deux sorties :

Les entrées : S pour la mise à 1 de la bascule

R pour la mise à 0 de la bascule

Les sorties : $Q\ et\ \bar{Q}$

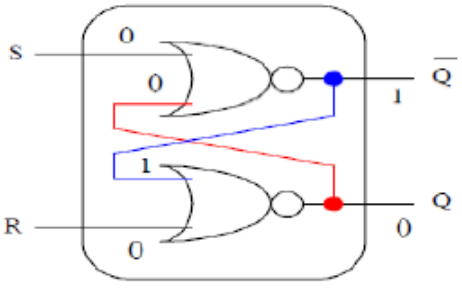
Le schéma de la bascule R-S est le suivant :



Pour comprendre le fonctionnement de la bascule R-S, on va étudier le comportement des variables de sortie ($Q\ et\ \bar{Q}$) en fonction des variables d'entrée (R et S). Pour cela, on désigne par :

1. Q_t: la variable de sortie à l'instant t (l'état présent de la variable).
2. Q_{t+1}: la variable de sortie à l'instant t+1 (l'état futur de la variable).

Pour un exemple : quand S=R=0 ; on suppose que $Q_t = 0$. Dans ce cas, on aura le schéma suivant :

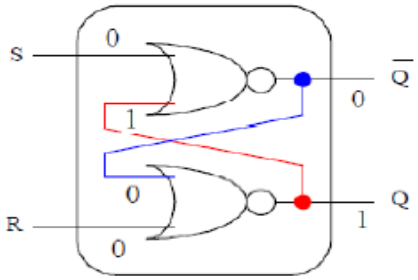


Ce schéma montre bien que, lorsque $Q_t = 0$, la variable de sortie de la porte NOR supérieure est à 1 ($\overline{S + Q_t} = \overline{0 + 0} = 1$). Donc, on aura $\overline{Q_{t+1}} = 1$.

Du moment que $\overline{Q_t} = 1$, la variable de sortie de la porte NOR inférieure sera à 0 ($\overline{\overline{Q_t} + R} = \overline{1 + 0} = 1$) et on a $Q_{t+1} = 0$. C'est-à-dire $Q_{t+1} = Q_t$

On dit que l'état de la bascule stable.

On ne supposant que $Q_t = 1$. Dans ce cas, on aura le schéma suivant :



(S=0 et $Q_t = 1 \rightarrow \overline{S + Q_t} = \overline{0 + 1} = 0$). Donc, on aura $Q_{t+1} = 0$.

(R=0 et $\overline{Q_t} = 0 \rightarrow \overline{\overline{Q_t} + R} = \overline{0 + 0} = 1$), et on aura : $\overline{Q_{t+1}} = 0$

Donc dans ce cas aussi, on a $Q_{t+1} = Q_t$.

On peut résumer le comportement de la bascule R-S par la table vérité suivante :

R	S	Q_t	Q_{t+1}	
0	0	0	0	$Q_{t+1} = Q_t$
0	0	1	1	
0	1	0	1	$Q_{t+1} = 1$ Mise à1
0	1	1	1	
1	0	0	0	$Q_{t+1} = 0$ Mise à 0
1	0	1	0	
1	1	0	X	Cas indéterminé
1	1	1	X	

Cette table est appelée, table caractéristique de la bascule R-S.

A partir de cette table, on peut déduire les expressions algébriques de Q_{t+1}et $\overline{Q_{t+1}}$ et pour obtenir des fonctions simplifiées, on utilise une table de karnaugh :

R \ SQ_t	00	01	11	10
0	0	1	1	1
1	0	0	X	X

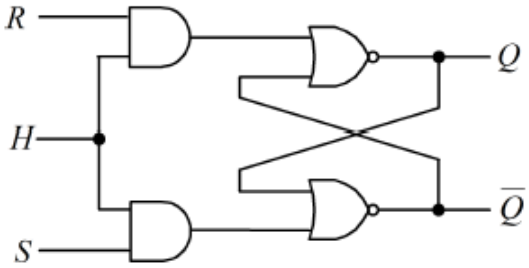
L'expression simplifiée de Q_{t+1} est $Q_{t+1} = \bar{R}Q_t + S$

De même, on obtient la forme simplifiées de $\overline{Q_{t+1}} = \bar{S}\,\bar{Q}_t + R$

Cette équation est appelée équation caractéristique de la bascule R-S

Synchronisation de la bascule R-S:

L'utilisation d'une horloge permet de synchroniser les changement d'état de la bascule.

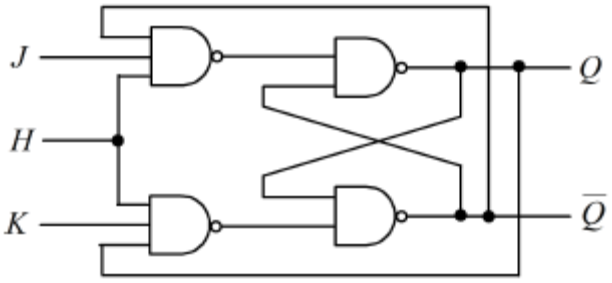


Il faut noter toutefois, que l'entrée S et R n'ont aucun d'effet que si H est à l'état 1. La table caractéristique de la bascule R-S synchrone est la suivante :

H	R	S	Q_t	Q_{t+1}	
0	0	0	0	0	$Q_{t+1} = Q_t$ L'état de bascule ne change pas
0	0	0	1	1	
0	0	1	0	0	
0	0	1	1	1	
0	1	0	0	0	
0	1	0	1	1	
0	1	1	0	0	
0	1	1	1	1	
1	0	0	0	0	$Q_{t+1} = Q_t$
1	0	0	1	1	
1	0	1	0	1	$Q_{t+1} = 1$ Mise à1
1	0	1	1	1	
1	1	0	0	0	$Q_{t+1} = 0$ Mise à 0
1	1	0	1	0	
1	1	1	0	X	Cas indéterminé
1	1	1	1	X	

IV.3.2 La bascule J-K

La bascule J-K diffère de la bascule R-S du fait que quand les deux variables d'entrée passent simultanément à 1 l'état de la bascule n'est pas indéterminé. En effet, quand J=K=1, on obtient la fonction de complémentation $Q_{t+1} = \overline{Q_t}$.



La table caractéristique de la bascule J-K

J	K	Q_t	Q_{t+1}	
0	0	0	0	$Q_{t+1} = Q_t$
0	0	1	1	
0	1	0	1	$Q_{t+1} = 1$
0	1	1	1	Mise à1
1	0	0	0	$Q_{t+1} = 0$
1	0	1	0	Mise à 0
1	1	0	1	$Q_{t+1} = \overline{Q_t}$
1	1	1	0	complément

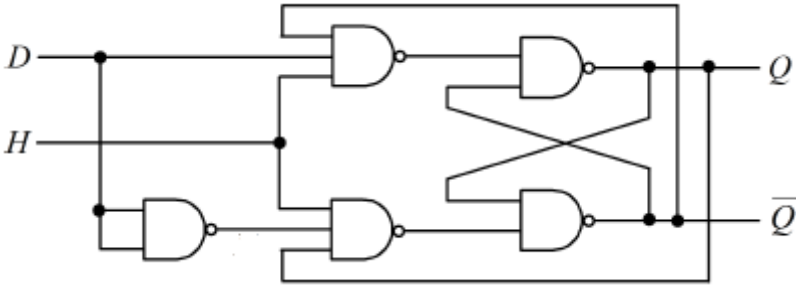
A partir de la table de karnaugh, on obtient l'expression de Q_{t+1} dans le cas de la bascule J-K :

J \ KQ_t	00	01	11	10
0	0	1	0	0
1	1	1	0	1

L'expression simplifiée de Q_{t+1}est: $Q_{t+1} = \overline{K}\ Q_t + J\overline{Q_t}$

IV.3.3 La bascule D

Une autre manière de résoudre le problème d'ambiguïté rencontrée ave la bascule R-S lorsque R=S=1, est de faire en sorte que ce cas ne se présente jamais à l'entrée de la bascule. Pour cela, on n'utilise seule variable d'entrée externe D et on parle alors de la bascule D. elle est illustrée par le schéma suivant :



La table caractéristique de la bascule D

H	D	Q_t	Q_{t+1}	
0	0	0	0	$Q_{t+1} = Q_t$
0	0	1	1	
0	1	0	0	$Q_{t+1} = 0$ Mise à 0
0	1	1	0	
1	0	0	1	$Q_{t+1} = 1$ Mise à1
1	0	1	1	
1	1	0	1	Q_{t+1} = D = 1
1	1	1	1	

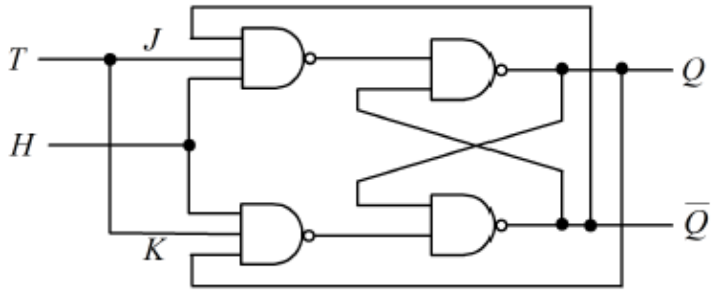
A partir de la table de karnaugh, on obtient l'expression de Q_{t+1} dans le cas de la bascule D :

D \ HQ_t	00	01	11	10
0	0	1	0	0
1	0	1	1	1

L'expression simplifiée de Q_{t+1}est: $Q_{t+1} = \overline{H}\ Q_t + DH$

IV.3.4 La bascule T

La bascule T ressemble assez à une bascule J-K à une seule entrée. Son schéma est le suivant :



Chaque fois qu'une impulsion arrive, les états de la bascule (sorties) sont inversés (complémentés).

1. Si T=0, alors, pas de changement.
2. Si T=1, il y a complémentation des variables de sortie.

D	Q_t	Q_{t+1}	
0	0	0	$Q_{t+1} = Q_t$
0	1	1	
1	0	1	$Q_{t+1} = \overline{Q_t}$
1	1	0	

A partir de la table de karnaugh, on obtient l'expression de Q_{t+1} dans le cas de la bascule D :

T \ Q_t	0	1
0	0	1
1	1	0

L'expression simplifiée de Q_{t+1}est: $Q_{t+1} = \overline{T}\ Q_t + T\ \overline{Q_t} = T \oplus Q_t$

CHAPITRE V:

LES COMPTEUR ET LES REGISTRES

V.1 Introduction

Un compteur est un système séquentiel (constitué par des bascules) qui possède "N" états différents (appelé compteur modulo N), dont à chaque top d'horloge le compteur passe de l'état E_i à l'état E_{i+1}, d'une manière cyclique jusqu'à revenir à l'état initial E_0.

V.2 Constitution

Un compteur est constitué de n Bascules avec $2^n \geq$ N

V.3 Classification des compteurs

Compteur asynchrone

Dans un compteur asynchrone l'impulsion d'horloge est appliquée à la première bascule. Les autres entrées d'horloges reçoivent les impulsions des sorties des étages (bascules) précédentes (dont les bascules sont toujours reliées en cascade).

Compteur synchrone

Pour ce type de compteur, l'horloge est appliquée directement et en même temps à toutes les bascules.

Compteur progressif

Un compteur binaire est dit progressif, si son contenu passe d'une valeur binaire b à la valeur $b + 1$ (dans un sens croissant) après application d'une impulsion d'horloge.

Compteur régressif

Un compteur est dit régressif, si son contenu passe d'une valeur binaire b à la valeur $b - 1$ (sens décroissant), après chaque application d'une impulsion d'horloge.

V.3.1 Compteur synchrone à cycle complet à 3 bit

C'est un compteur modulo 8 constitué de trois bascules ($2^3 \geq 8$), synchronisées par le même signale d'horloge, il peut être réalisé par n'importe quel type de bascule.

A. Réalisation avec la bascule D :

Table d'état

Valeur	Etat présent			Etat future			Entrées des bascules		
	Q_2	Q_1	Q_0	Q_2^+	Q_1^+	Q_0^+	D_2	D_1	D_0
0	0	0	0	0	0	1	0	0	1
1	0	0	1	0	1	0	0	1	0
2	0	1	0	0	1	1	0	1	1
3	0	1	1	1	0	0	1	0	0
4	1	0	0	1	0	1	1	0	1
5	1	0	1	1	1	0	1	1	0
6	1	1	0	1	1	1	1	1	1
7	1	1	1	0	0	0	0	0	0

Maintenant on charge les équations de $D_2, D_1 et\ D_0$ en fonction de $Q_2, Q_1 et\ Q_0$

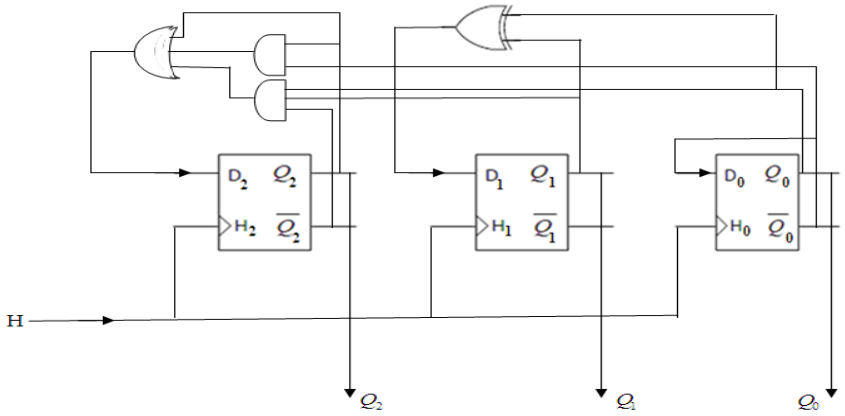
Alors on trouve :

$D_2 = \overline{Q_2}\, Q_1\, Q_0 + Q_2\overline{Q_0} + Q_2$

$D1 = Q_1\overline{Q_0} + \overline{Q_1}Q_0 = Q_1 \oplus Q_0$

$D_0 = \overline{Q_0}$

Logigramme



B. Réalisation avec la bascule JK :

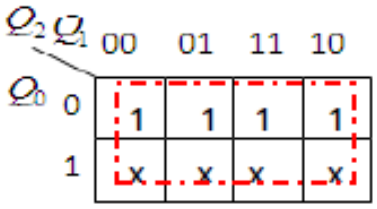
Considérons un compteur synchrone progressif formé de trois bascules type JK. En utilisant la table d'excitation de la bascule JK, ci-contre, nous pouvons obtenir la table d'implication (table d'état) de notre compteur, comme suit :

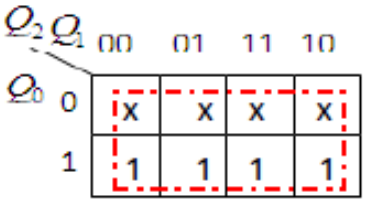
Q	Q^+	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Table d'état

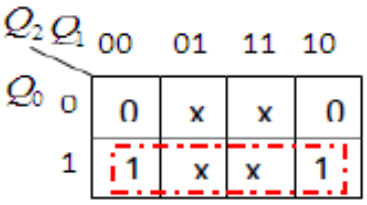
Valeur	Etat présent			Etat future			Entrées des bascules					
	Q_2	Q_1	Q_0	Q_2^+	Q_1^+	Q_0^+	J_2	K_2	J_1	K_1	J_0	K_0
0	0	0	0	0	0	1	0	X	0	X	1	X
1	0	0	1	0	1	0	0	X	1	X	X	1
2	0	1	0	0	1	1	0	X	X	0	1	X
3	0	1	1	1	0	0	1	X	X	1	X	1
4	1	0	0	1	0	1	X	0	0	X	1	X
5	1	0	1	1	1	0	X	0	1	X	X	1
6	1	1	0	1	1	1	X	0	X	0	1	X
7	1	1	1	0	0	0	X	1	X	1	X	1

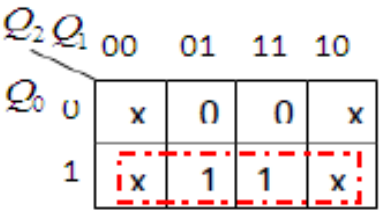
Les équations logiques



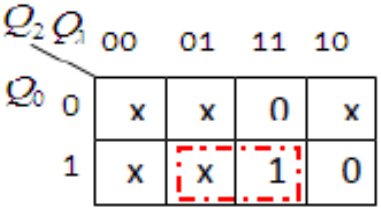


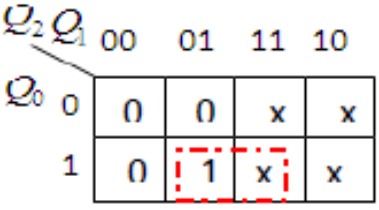
$K_0 = 1 \quad et \quad J_0 = 1$





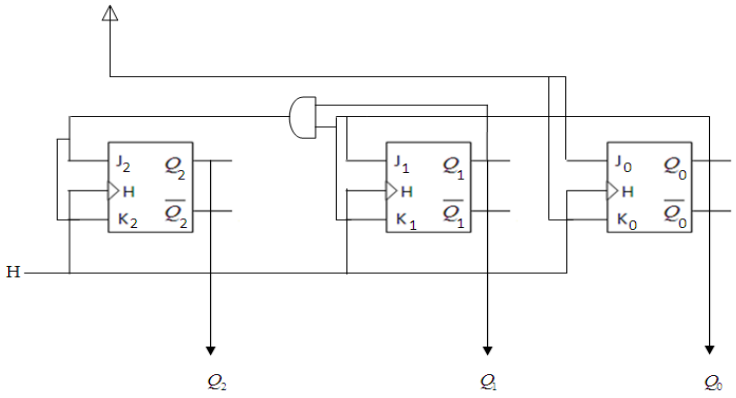
$K_1 = Q_0 \quad et \quad J_1 = Q_0$





$K_2 = Q_0.Q_1 \quad et \quad J_2 = Q_0.Q_1$

Logigramme



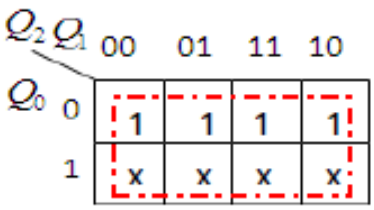
V.3.2 Compteur synchrone régressif :

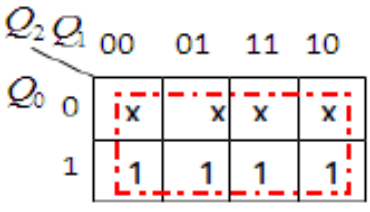
Considérons un compteur synchrone régressif, formé de trois bascules type JK. La table d'implication de notre compteur est la suivante :

Table d'état

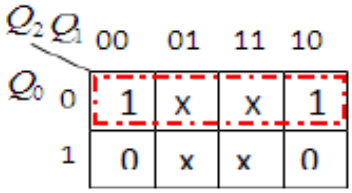
Valeur	Etat présent			Etat future			Entrées des bascules					
	Q_2	Q_1	Q_0	Q_2^+	Q_1^+	Q_0^+	J_2	K_2	J_1	K_1	J_0	K_0
7	1	1	1	1	1	0	X	0	X	0	X	1
6	1	1	0	1	0	1	X	0	X	1	1	X
5	1	0	1	1	0	0	X	0	0	X	X	1
4	1	0	0	1	0	0	X	1	1	X	1	X
3	1	0	0	0	1	1	0	X	X	0	X	1
2	0	1	1	0	1	0	0	X	X	1	1	X
1	0	1	0	0	0	0	0	X	0	X	X	1
0	0	0	0	1	1	1	1	X	1	X	1	X

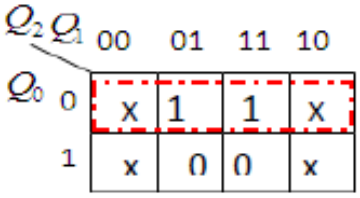
Les équations logiques



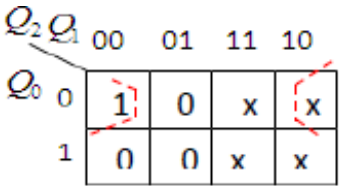


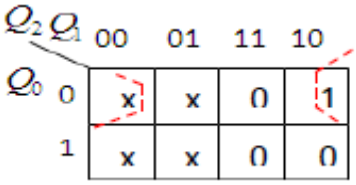
$K_0 = 1 \quad et \quad J_0 = 1$





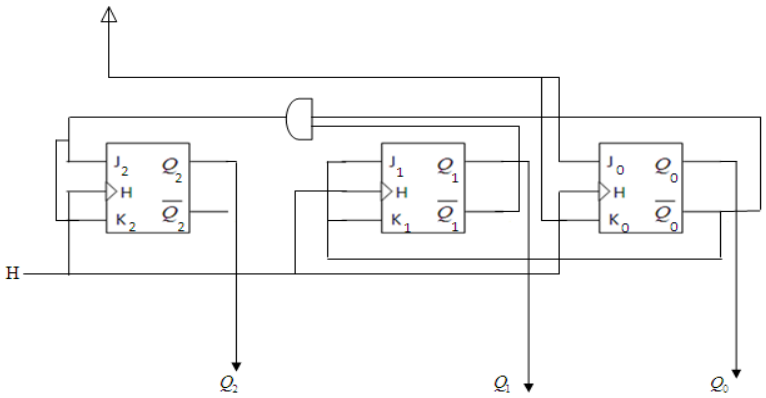
$K_1 = \overline{Q_0}$ *et* $J_1 = \overline{Q_0}$





$K_2 = \overline{Q_0}.\overline{Q_1}$ *et* $J_2 = \overline{Q_0}.\overline{Q_1}$

Logigramme



V.3.3 Compteur synchrone à cycle incomplet :

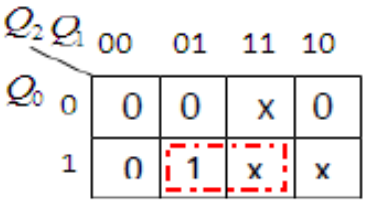
Considérons un compteur synchrone progressif modulo 5, formé de trois bascules de type D. La table d'implication de ce compteur est la suivante :

Table d'état

Valeur	Etat présent			Etat future			Entrées des bascules		
	Q_2	Q_1	Q_0	Q_2^+	Q_1^+	Q_0^+	D_2	D_1	D_0
0	0	0	0	0	0	1	0	0	1
1	0	0	1	0	1	0	0	1	0
2	0	1	0	0	1	1	0	1	1
3	0	1	1	1	0	0	1	0	0
4	1	0	0	0	0	0	0	0	0
5	1	0	1	X	X	X	X	X	X
6	1	1	0	X	X	X	X	X	X
7	1	1	1	X	X	X	X	X	X

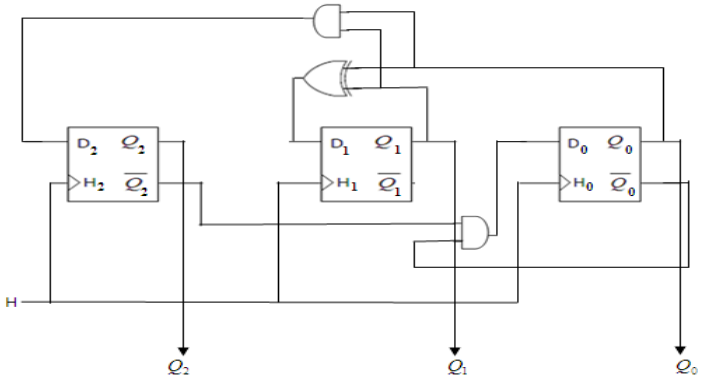
Les équations logiques

$D_0 = \overline{Q_0\, Q_2}$ *et* $D_1 = \overline{Q_0}Q_1 + Q_0\overline{Q_1} = Q_1 \oplus Q_0$



$D_2 = Q_0Q_1$

Logigramme



V.4 Les Registres

Un registre est un circuit séquentiel synchrone, capable de stocker temporairement des informations binaires en utilisant un ensemble de bascules.

V.4.1 Applications :

Les applications des registres sont nombreuses, parmi lesquelles :

1. Stockage temporel de l'information.
2. Conversion série – parallèle des données binaires.
3. Multiplication par une puissance de deux (décalage vers la gauche).
4. Division par une puissance de deux (décalage vers la droite).

V.4.2 Type de registres

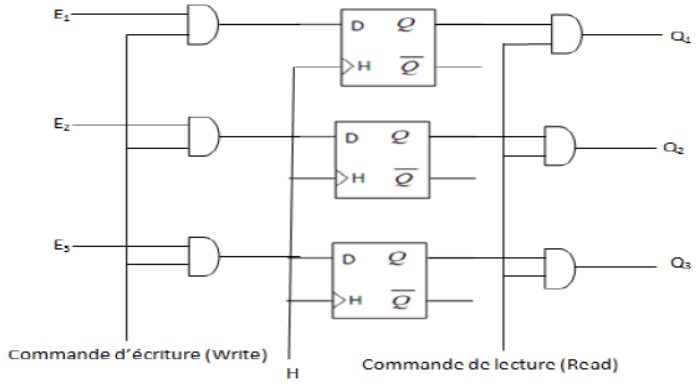
On distingue deux types de registres :

1. Registre de mémorisation.
2. Registre de décalage.

A- Registre de mémorisation

Le registre de mémorisation est un registre capable de stocker ou mémoriser un mot de n bits.

Exemple



B- Registre à décalage

Ce type de registre est un registre capable de décaler à droite ou à gauche sont contenu. Ces types de registres peuvent être utilisés pour effectuer des multiplications ou divisions par une puissance de deux, ou encore pour effectuer une conversion série-parallèle.

Exemple (La multiplication par 2)

Soit $N = (3)_{10} = (0011)_2$

$2 \times N = 6 = 0110$ (décalage à gauche)

$2 \times (2 \times N) = 12 = 1100$

On constate que pour effectuer la multiplication d'un nombre binaire par 2 il suffit de décaler tous les bits vers la gauche (c'est-à-dire vers les bits de poids fort).

De la même façon pour réaliser la division d'un nombre par 2 il suffit de décaler tous les bits de nombre vers la droite (vers les bites de poids faible)

100 – – – – – – – – – – – – –> 010 (décalage à droite)

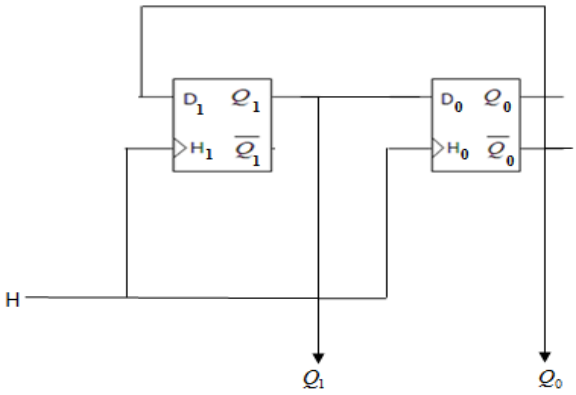
Soit la table caractéristique (d'état) d'un registre :

Etat présent		Etat future		Entrées de registre	
Q_1	Q_0	Q_1^+	Q_0^+	D_1	D_0
0	0	0	0	0	0
0	1	1	0	1	0
1	0	0	1	0	1
1	1	1	1	1	1

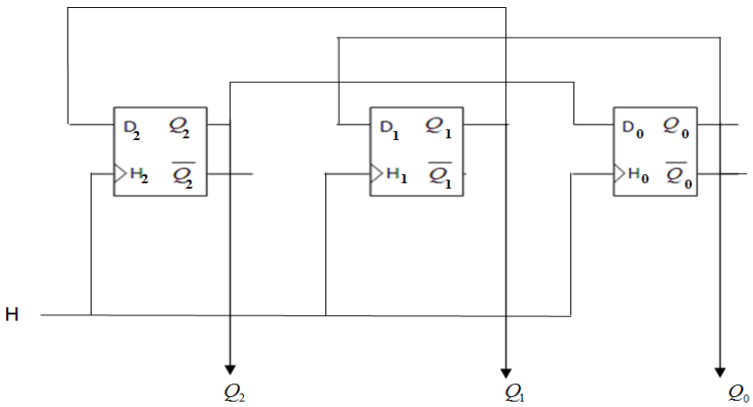
On remarque que les bits (1) sont décalés vers la droite à chaque fois donc il s'agit d'un registre à décalage à droite, dont :

$D_0 = Q_1$ et $D_1 = Q_0$

Logigramme



Soit le circuit suivant :



Les équations d'entrées du système sont :

$\mathrm{D}_0 = \mathrm{Q}_2, \quad D_1 = Q_0 \quad et \ D_1 = Q_0$

La table caractéristique du circuit est :

Etat présent			Etat future		Entrées de bascule			
Q_2	Q_1	Q_0	Q_2^+	Q_1^+	Q_0^+	D_2	D_1	D_0
0	0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	1	0
0	1	0	1	0	0	1	0	0
0	1	1	1	1	0	1	1	0
1	0	0	0	0	1	0	0	1
1	0	1	0	1	1	0	1	1
1	1	0	1	0	1	1	0	1
1	1	1	1	1	1	1	1	1

On remarque que les bits (1) sont décalés vers la gauche à chaque fois ($exemple\ 101 - - - - - - - - - - - - -> 011$)

Donc il s'agit d'un registre à décalage à gauche sur 3 bits, en plus il y a une rotation des bits , alors c'est un registre à décalage à gauche avec rotation.

Réaliser un registre à décalage à droite avec rotation sur 2 bits à base des bascules RS.

V.4.3 Type de registre à décalage :

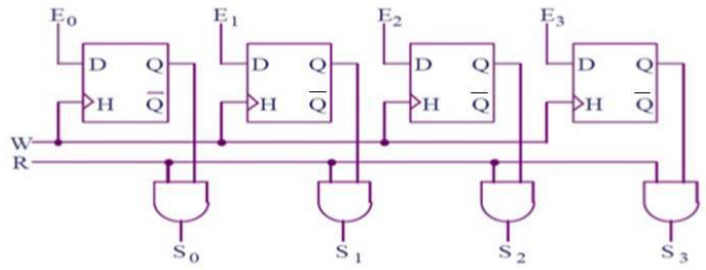
On distingue 4 types de registre à décalage :

1- Registre a entrée série et sortie série (SISO)
2- Registre a entrée série et sortie parallèle (SIPO)
3- Registre a entrée parallèle et sortie série (PISO)
4- Registre a entrée parallèle et sortie parallèle (PIPO)

A- Registre a entrée parallèle et sortie parallèle (PIPO)

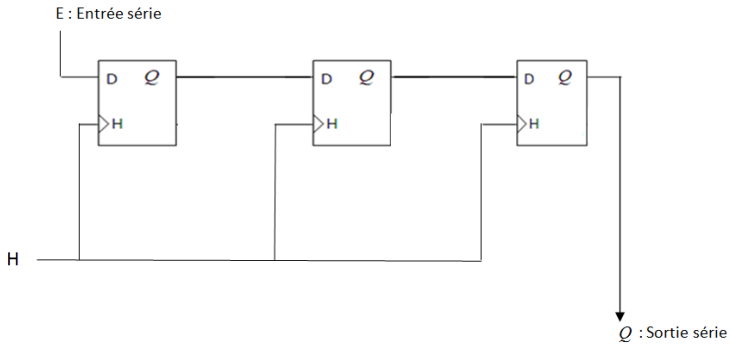
Il peut charger une information sur N bits en même temps, dont chaque bascule prend la valeur de l'information. Ce type de registre possède une entrée de chargement W tels que :

1- $Si\ W = 0\ - -$> état de mémorisation
2- $Si\ W\ = 1\ - -$> état de chargement

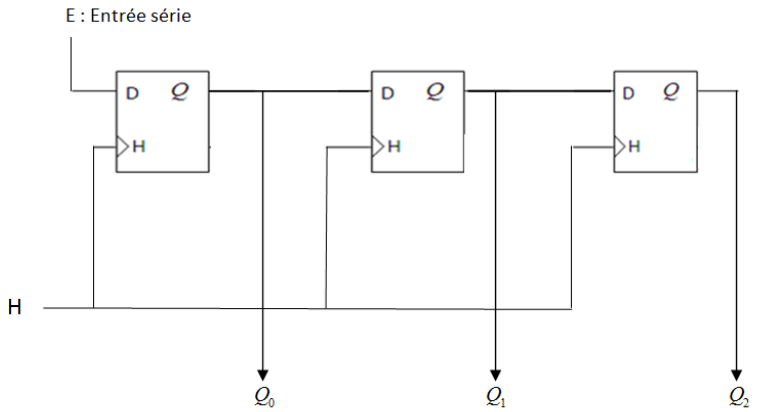


B- Registre a entrée série et sortie série (SISO)

Dans ce cas l'information (E) est introduite bit par bit (en série)



C- Registre a entrée série et sortie parallèle (SIPO)

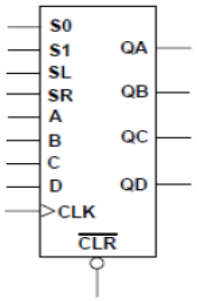


V.4.4 Registres universels

Il existe des circuits intégrés regroupant les quatre types de registres précédents. Ils permettent les modes de fonctionnement suivants :

1- chargement et lecture parallèles.
2- chargement série et décalages à droite ou à gauche, lecture série ou parallèle.

3- chargement parallèle et décalages à droite ou à gauche, lecture série ou parallèle. Par exemple, le circuit intégré de référence 74194 possède la représentation symbolique suivante :



Les entrées A, B, C, D sont les entrées parallèles. Les entrées SL et SR sont respectivement les entrées/sorties série gauche et droite. Les entrées S0 et S1 permettent de choisir le mode de fonctionnement de ce registre (blocage, décalage à droite, décalage à gauche, chargement parallèle). L'entrée CLR (active sur niveau bas) permet une remise à zéro asynchrone des sorties. L'entrée CLK est l'entrée horloge de synchronisation. Les sorties sont QA, QB, QC, QD.

BIBLIOGRAPHIE

[1] : INEL FOUAD, "Cours logique combinatoire et séquentielle ", Université de 20 Aout 1955 –Skikda-.

[2 : Bouchemel Ammar, " Cours logique combinatoire et séquentielle ", UNIVERSITE 8 MAI 1945 GUELMA, 2019.

[3] : Letocha ; Introduction aux circuits logiques, 2e Edition Mc-Graw Hill. 3 novembre 1997.

[4] : AMIMEUR Hocine, " Cours logique combinatoire et séquentielle ", Université abderrahmane mira bejaia, 2017.

[5] : MEHARRAR AOUED, " Cours logique combinatoire et séquentielle ", Centre Universitaire El-Wancharissi de Tissemsilt 2019.

[6] : Ouerghemmi N & Tarhouni W, " Les circuits logiques combinatoires ", Office des publications universitaire..

[7] : Djamal GOZIM, " Cours logique combinatoire et séquentielle ", Université Ziane Achor de Djelfa 2005.

Semestre: 4
Unité d'enseignement: UEF 2.2.1
Matière 2: Logique combinatoire et séquentielle
VHS: 45h00 (Cours: 1h30, TD: 1h30)
Crédits: 4
Coefficient: 2

Objectifs de l'enseignement:

Connaître les circuits combinatoires usuels. Savoir concevoir quelques applications des circuits combinatoires en utilisant les outils standards que sont les tables de vérité, les tables de Karnaugh. Introduire les circuits séquentiels à travers les circuits bascules, les compteurs et les registres.

Connaissances préalables recommandées

Aucune.

Contenu de la matière :

Le nombre de semaines affichées sont indiquées à titre indicatif. Il est évident que le responsable du cours n'est pas tenu de respecter rigoureusement ce dimensionnement ou bien l'agencement des chapitres.

Chapitre 1 : Algèbre de Boole et Simplification des fonctions logiques **2 semaines**

Variables et fonctions logiques (OR, AND, NOR, NAND, XOR). Lois de l'algèbre de Boole. Théorème de De Morgan. Fonctions logiques complètes et incomplètes. Représentation des fonctions logiques: tables de vérité, tables de Karnaugh. Simplification des fonctions logiques : Méthode algébrique, méthode de Karnaugh.

Chapitre 2 : Systèmes de numération et Codage de l'information **2 semaines**

Représentation d'un nombre par les codes (binaire, hexadécimal, DCB, binaire signé et non signé, ...) changement de base ou conversion, codes non pondérés (code de Gray, codes détecteurs et correcteurs d'erreurs, code ascii, ...), opérations arithmétiques dans le code binaire.

Chapitre 3 : Circuits combinatoires transcodeurs **2 semaines**

Définitions, les décodeurs, les encodeurs de priorité, les transcodeurs, Mise en cascade, Applications, Analyse de la fiche technique d'un circuit intégré décodeur, Liste des circuits intégrés de décodage.

Chapitre 4 : Circuits combinatoires aiguilleurs **2 semaines**

Définitions, les multiplexeurs, les démultiplexeurs, Mise en cascade, Applications, Analyse de la fiche technique d'un circuit intégré d'aiguillage, Liste des circuits intégrés.

Chapitre 5 : Circuits combinatoires de comparaison **2 semaines**

Définitions, circuit de comparaison à 1 bit, 2 bits et 4 bits, Mise en cascade, Applications, Analyse de la fiche technique d'un circuit intégré de comparaison, Liste des circuits intégrés.

Chapitre 6 : Les bascules **2 semaines**

Introduction aux circuits séquentiels. La bascule RS, La bascule RST, La bascule D, La bascule Maitre-esclave, La bascule T, La bascule JK. Exemples d'applications avec les bascules : Diviseur de fréquence par n, Générateur d'un train d'impulsions, ...
Il est conseillé de présenter pour chaque bascule la table de vérité, des exemples de chronogrammes ainsi que les limites et imperfections.

Chapitre 7 : Les compteurs **2 semaines**

Définition, Classification des compteurs (synchrone, réguliers, irréguliers, asynchrone, cycles complets et incomplets). Réalisation de compteurs binaires synchrones complets et incomplets, Tables d'excitation des bascules JK, D et RS, Réalisation de compteurs binaires asynchrones modulo (n) :

complets, incomplets, réguliers et irréguliers. Compteurs programmables (démarrage à partir d'un état quelconque).

Chapitre 8. Les Registres **1 Semaine**

Introduction, les registres classiques, les registres à décalage, chargement et récupération des données dans un registre (PIPO, PISO, SIPO, SISO), décalage des données dans un registre, un registre universel, le 74LS194A, les circuits intégrés disponibles, Applications : registres classiques, compteurs particuliers, files d'attente.

Mode d'évaluation :

Contrôle continu : 40 % ; Examen final : 60 %.

Références bibliographiques:

1- J. Letocha, Introduction aux circuits logiques, Edition McGraw Hill.
2- J.C. Lafont, Cours et problèmes d'électronique numérique, 124 exercices avec solutions, Ellipses.
3- R. Delsol, Electronique numérique, Tomes 1 et 2, Edition Berti
4- P. Cabanis, Electronique digitale, Edition Dunod.
5- M. Gindre, Logique combinatoire, Edition Ediscience.
6- H. Curry, Combinatory Logic II. North-Holland, 1972
7- R. Katz, Contemporary Logic Design, 2nd ed. Prentice Hall, 2005.
8- M. Gindre, Electronique numérique : logique combinatoire et technologie, McGraw Hill, 1987
9- C. Brie, Logique combinatoire et séquentielle, Ellipses, 2002.
10- J-P. Ginisti, La logique combinatoire, Paris, PUF (coll. « Que sais-je? » n°3205), 1997.
11- J-L. Krivine, Lambda-calcul, types et modèles, Masson, 1990, chap. Logique combinatoire, traduction anglaise accessible sur le site de l'auteur.