الجمهورية الجزائرية الديمقراطية الشعبية

République Algérienne Démocratique et Populaire

Ministère de L'enseignement Supérieure de la recherche scientifique

Université 8 Mai 1945 –Guelma-

Faculté des Mathématiques, d'informatique et des Sciences de la Matière

Département d'Informatique

Polycopié

Structure Machine 2

Cours & Exercices

1ère Année Tronc-commun

Mathématique-Informatique

Dr. Bencheriet Chemesse Ennehar

bencheriet.chemesseennahar@univ-guelma.dz

2021/2022

Sommaire

Chapitre 4 : Les Automates Finis

Chapitre 5 : Les Circuits Intégrés

Corrigé des Exercices

Avant-propos

Ce cours est destiné aux étudiants de première année tronc-commun MI. Il est étudié en semestre deux et vient compléter le module de structure machine 1 qui touche principalement l'information et les données, représentant la partie virtuelle de la machine. Le cours de structure machine 2 s'oriente plutôt vers la partie physique de la machine permettant ainsi à l'étudiant de compléter ses connaissances et de faire le lien entre l'information qui circule dans la machine et le matériel qui gère tout cela.

L'objectif principal de ce cours est donc de permettre à l'étudiant d'acquérir les connaissances de base de la structure matérielle de la machine et le principe de fonctionnement de chacun des composants. Ces connaissances vont servir de plateforme pour d'autres aspects en relation avec l'ordinateur (programmation, base de données, réseaux,…).

Prè-requis : Structure Machine 1

MINISTERE DE L'ENSEIGNEMENT
SUPERIEUR ET DE LA RECHERCHE
SCIENTIFIQUE
UNIVERSITE 8 MAI 1945
GUELMA
Faculté de Mathématiques,
D'Informatique et de Sciences de la Matière
Département : Mathématiques

وزارة التعليم العالـي
و البحث العلمي
جامعة 8 مـاي 1945
قالمـة
كليـة الرياضيـات، الإعـلام الآلـي
و علـوم المـادة
قسم: الرياضيات

SYLLABUS

INFORMATION SUR LA MATIERE

Unité d'Enseignement : UEF222 **Matière** : Structure Machine 2
Domaine /Filière : 1ère Année L.M.D (M.I)
Semestre : 2 **Année Universitaire** : 2021/2022.
Crédit : 04 **Coefficient** : 02
Volume Horaire Hebdomadaire Total : 3 Heures
- Cours Magistral (1.5 H)
- Travaux Dirigés (1.5 H)

Langue d'enseignement : Français

Site Web du cours: http://elearning.univ-guelma.dz/moodle/

INFORMATIONS SUR LE RESPONSABLE DE LA MATIERE

Enseignant responsable de la matière : Mme Zeiare-Bencheriet chemesse ennehar
Grade : MCA

E-mail : bencheriet.chemesseennehar@univ-guelma.dz
cbencheriet@yahoo.fr

Bureau : E8.2 **Téléphone (Département Informatique)** : 037 21 67 63
Laboratoire d'attachement : LAIG (Laboratoire d'Automatique et d'Informatique de Guelma)

OBJECTIFS DE LA MATIERE

Acquérir les connaissances de base de l'architecture des ordinateurs et le principe de fonctionnement de chacun des composants. Ces connaissances vont servir de plateforme pour d'autres aspects en relation avec l'ordinateur (programmation, base de données, réseaux,…).

CONTENU DE LA MATIERE

Chapitre 1 : Introduction

- Introduction générale.

Chapitre 2 : La logique combinatoire

- Définition.
- Les circuits combinatoires.
- Etapes de conception d'un circuit combinatoire :
  - Etablissement de la table de vérité.
  - Simplification des fonctions logiques.
  - Réalisation du schéma logique.
- Etude de quelques circuits combinatoires usuels :
  - Le demi-additionneur.
  - L'additionneur complet.
  - L'additionneur soustracteur (en complément vrai)
  - Les décodeurs.
  - Les multiplexeurs.
  - Les encodeurs de priorité.
  - Les démultiplexeurs.
- Autres exemples de circuits combinatoires.

Chapitre 3 : La logique séquentielle

- Définition.
- Les bascules (RS, JK, D)
- Les registres (à chargement parallèle et à décalage)
- Les mémoires.
- Synthèse d'un circuit séquentiel (automates):
  - Automate de Moore et automate de Mealy.
  - Graphe et matrice de transition.
  - Choix des bascules et codage des états.
  - Matrice d'excitation des bascules.
  - Simplification des fonctions logiques.
  - Etablissement du schéma logique.
- Réalisation d'automates :
  - Les compteurs/décompteurs.
  - Autres exemples d'automates.

Chapitre 4 : Les circuits intégrés

- Définition
- Etude des caractéristiques d'un circuit intégré simple (exemple circuit ou 7432)
- Notions sur la réalisation du montage d'un circuit combinatoire simple en utilisant des circuits intégrés.

EVALUATION : CONTROLE DES CONNAISSANCES & PONDERATIONS

contrôle	Pondération (%)
Examen final	**60**
Travaux dirigés (présence et participation)	**20**
Micro – interrogations	**20**
Devoirs à domicile	**-**

REFERENCES BIBLIOGRAPHIQUES

1. Paolo Zanella, Architecture et technologie des ordinateurs, 5ème Edition, Dunod, 2013.
2. Claude BRIE, Informatique industrielle : Logique combinatoire et séquentielle; Méthodes, outils et réalisations, Ellipses, 2003.
3. Jean Jacques Mercier, Bit après bit numération, arithmétique binaire ,logique combinatoire, Ellipses, 2005.
4. Jean Jacques Mercier, Séquence après séquence logique séquentielle circuits asynchrones et synchrones, Ellipses, 2006.
5. Jean Jacques Mercier, Instruction après instruction: logique séquentielle circuits asynchrones et asynchrones et synchrones, Ellipses, 2008.
6. Phillipe Darch, Architecture des Ordinateurs : Logique booléenne et implémentation Technologique. Edition VUIBERT , 2004.
7. John R. Gregg, Ones and Zeros: Understanding Boolean Algebra, Digital Circuits, and the Logic of Sets 1st Edition, Wiley & sons Inc. publishing, 1998.
8. Bradford Henry Arnold, Logic and Boolean Algebra, Dover publication, Inc., Mineola, 2011.

NB : Retrouver les nouveaux ouvrages disponibles dans la bibliothèque à traves le lien :
https://pmb.univ-guelma.dz/opac_css/

Liste des Figures

Liste des Tableaux

Liste des Abréviations

Abréviation	Signification
Bit	**B**inary Dig**IT**
BCD	**B**inary **C**oded **D**ecimal
MSB	**M**ost **S**ignificant **B**it
LSB	**L**east **S**ignificant **B**it
Ck / H	**C**lock / **H**orloge
C / R	**C**arry / **R**etenue
Mux	**Mu**ltiplexeur
Demux	**Dému**ltiplexeur
MEF	**M**achine à **E**tats **F**inis
FSM	**F**init **S**tate **M**achine
CI	**C**ircuit **I**ntégré
DIP	**D**ual **I**nline **P**ackage
DTL	**D**iode **T**ransistor **L**ogic
TTL	**T**ransistor **T**ransistor **L**ogic
ECL	**E**mitter **C**oupled **L**ogic
CMOS	**C**omplementary **M**etal **O**xide **S**emiconductor
DSP	**D**igital **S**ignal **P**rocess
SSI	**S**mall **S**cale **I**ntegration
MSI	**M**edium **S**cale **I**ntegration
LSI	**L**arge **S**cale **I**ntegration
VLSI	**V**ery **L**arge **S**cale **I**ntegration
ULSI	**U**ltra **L**arge **S**cale **I**ntegration
GLSI	**G**iga **L**arge **S**cale **I**ntegration
ASIC	**A**pplication **S**pecific **I**ntegrated **C**ircuit
FPGA	**F**ield **P**rogrammable **G**ate **A**rray
CPLD	**C**omplex **P**rogrammable **L**ogic **D**evice
TV	**T**able de **V**érité
TK	**T**able de **K**arnaugh

Introduction Générale

Les calculs et les traitements effectués par les machines sont élaborés par des circuits, de très petite taille, dits circuits intégrés ayant chacun une fonction spécialisée tel que : l'Unité Arithmétique et Logique (UAL), Décodeur d'instruction, mémoire,…etc. Ces circuits sont conçus à partir de circuits logiques dont le principal rôle est l'exécution des opérations sur des variables logiques (binaires)

Le cours de structure machine 2 s'intéresse principalement à l'analyse et la synthèse des circuits logiques combinatoires et séquentiels. Ce qui permettra de comprendre le fonctionnement de ces circuits élémentaires.

Ce polycopié est scindé en cinq chapitres :

Chapitre 1 : est une introduction générale sur la matière présentant quelques définitions de bases dont principalement les circuits logiques combinatoires et les circuits logiques séquentiels suivi de rappels sur les fondements de base de l'algèbre de Boole.

Chapitre 2 : l'objectif principal de ce chapitre est d'apprendre à synthétiser des circuits combinatoires arithmétiques, de transcodage, de multiplexage et de comparaison en utilisant les principes de l'algèbre de Boole et les portes logiques.

Chapitre 3 : dédié à l'étude des circuits séquentiels en partant de l'élément de base qui est la bascule et les différents types, à la synthèse de circuits séquentiels plus complexes telque les registre et les compteurs.

Chapitre 4 : consacré à l'apprentissage du fonctionnement des automates à état finis ainsi que l'analyse et la synthèse d'un automate en utilisant les circuits séquentiels étudiés dans le chapitre 3.

Chapitre 5 : l'objectif principal de ce chapitre est de familiariser l'étudiant avec le côté matériel des circuits logiques étudiés dans les chapitres précédents. Une description détaillée, sur la constitution interne des circuits intégrés ainsi que leurs utilisations, permettra à l'étudiant de faire une analogie entre le diagramme logique et la réalisation matérielle d'un circuit quelconque.

Chapitre 1
Introduction

Chapitre 1

Introduction

I. Introduction

Les circuits logiques sont des éléments cruciales dans la constitution des ordinateurs et des appareils que nous manipulons quotidiennement ils sont responsables de tous les calculs et les traitements effectués par la machine.

L'objectif de ce chapitre est de donner quelques définitions de bases dont principalement les circuits logiques combinatoires et séquentiels et de rappeler les fondements de base de l'algèbre de Boole.

Les références bibliographiques utilisées dans ce chapitre sont principalement [1][2][3].

II. Quelques définitions

II.1 Ordinateur

L'ordinateur est un dispositif électronique sophistiqué de traitement de l'information. Grace à sa constitution matérielle et logiciel très développée il est capable de réaliser diverses fonctionnalités tel que l'acquisition et le stockage de l'information (textes, sons, images, etc.). Il est également capable d'effectuer des traitements de l'information et de la restituer sous une autre forme.

II.2 Information traitée

Rappelons que les informations traitées par l'ordinateur sont sous forme d'une suite d'impulsions électriques traduisant les symboles introduit codés sous forme d'une suite de bits (signal binaire).

II.3. Matériel et Logiciel (Hardware & Software)

L'ordinateur ou la Machine (nouvelle appellation) est composé de deux parties indissociables qui le font fonctionner : le matériel (hardware) et le logiciel (software).

II.3.1 Partie logiciel

C'est la partie invisible de l'ordinateur, constituée essentiellement des programmes responsables du fonctionnement et de la manipulation de la machine. Ils constituent la partie intelligente de la machine.

Le software est composé d'une suite d'instructions de programme stocké en mémoire et exécuté par la machine ou plus précisément par le processeur. On peut distinguer 3 grandes catégories de logiciel (software), dont le logiciel système (systèmes d'exploitation), le logiciel d'application (logiciel de traitements de texte, de traitement d'images, les tableurs…etc) et le logiciel de programmation (outils de programmation tel C++, Pascal, Java,…etc).

II.3.2 Partie matériel

C'est la partie physique de la machine représentée par l'ensemble des composants électroniques qui sont généralement intégrés dans des circuits appelés « circuits intégrés » où chacun exécute une fonction spécialisés tel que : opérations arithmétiques et logique, mémorisation, décodage des instructions,….etc. Ces circuits sont appelés également « **circuits logiques** ».

II.4 Circuits logiques

L'ensemble des traitements effectués par la machine sont essentiellement réalisés à l'aide d'opérations telles l'addition, la soustraction, la multiplication, la division, la comparaison toutes réalisé par des circuits logiques.

Un circuit logique est un circuit ayant pour but d'exécuter des opérations sur des variables logiques (binaires).

Les circuits logiques nécessaires pour la conception et la réalisation de n'importe quel appareil électronique (ordinateur, imprimante, scanner, …etc) se divisent en deux types selon leur fonctionnement :

- Circuits Combinatoires.
- Circuits Séquentiels.

II.4.1 Circuits Combinatoires

Les circuits combinatoires sont des dispositifs dont les fonctions de sortie s'expriment selon des expressions logiques des seules variables d'entrée. $S(t) = F[E_1(t), E_2(t), \ldots]$

Un circuit combinatoire peut être défini par une ou plusieurs fonctions logiques.

Exemple : parmi les circuits combinatoires les plus répondus on trouve les suivants : Portes logiques, Additionneurs, Soustracteurs, Multiplexeur/ Démultiplexeur, Codeur /Décodeur, Comparateur…etc.

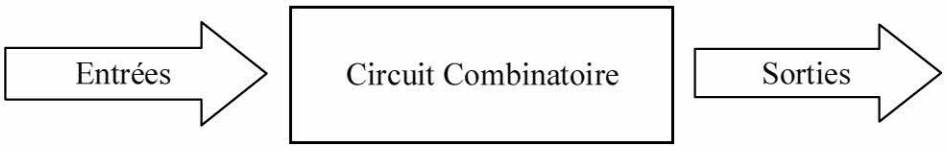


Fig 1.1 : Schéma simplifié d'un circuit combinatoire

II.4.2 Circuits Séquentiels

Les circuits séquentiels sont des dispositifs dont les fonctions de sortie dépendent non seulement de l'état des variables d'entrée mais également de l'état antérieur de certaines variables de sortie (propriétés de mémorisation) $S(t+1) = F[E_1(t), E_2(t),\ldots., S(t)]$

Exemples : parmi les circuits séquentiels les plus répondus on trouve : Bascules, Registres, Mémoires, Compteurs…etc.

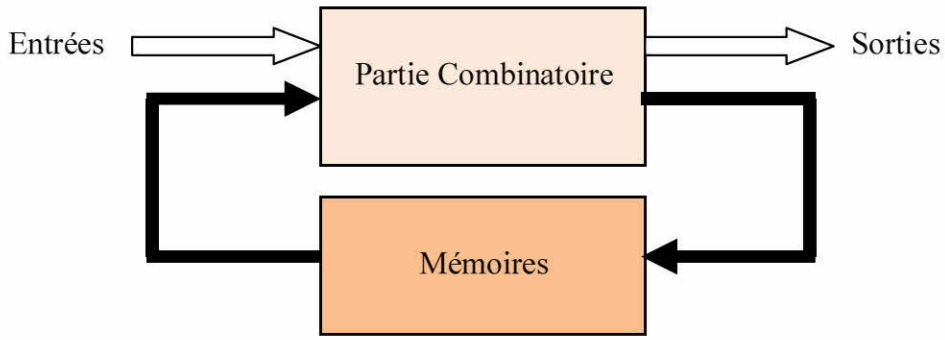


Fig 1.2 : Schéma simplifié d'un circuit séquentiel

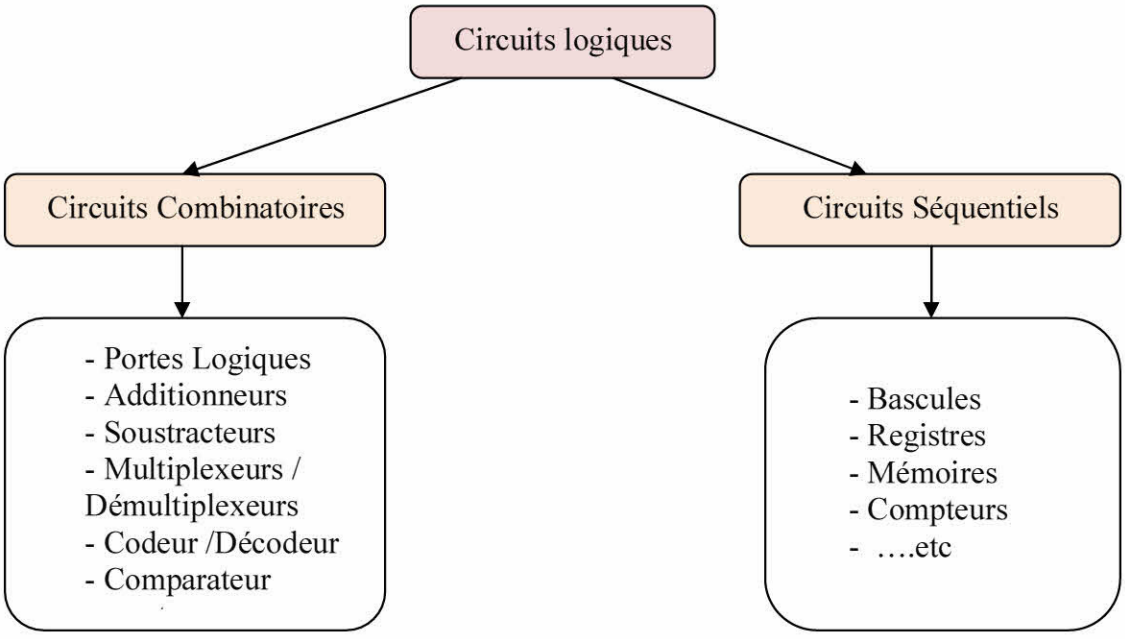


Fig 1.3 : Exemple de circuits logique combinatoires et séquentiels

II.5. Différence entre circuits combinatoires et circuits séquentiels

Pour mieux distinguer entre ces deux types de circuits prenant l'exemple du contrôle de lignes téléphonique suivant :

On suppose avoir un circuit de contrôle de lignes téléphoniques contenant 3 lampes témoins fonctionnant comme suit :

a. Lampe verte allumée si un minimum de lignes est occupé.
b. Lampe bleu allumée si la moitie des lignes sont occupées.
c. Lampe rouge allumée si une majorité des lignes est occupée.

Si la personne qui contrôle les lignes s'absente un moment est au moment de son absence les lignes ont été occupées et en revanchent-elles (lignes) ne le sont plus à son retour, donc le circuit de control ne nous permet pas de mémoriser l'état précédant des lampes témoins lies aux lignes téléphoniques. Donc pour améliorer notre circuit contrôleur nous devons lui rajouter un circuit de mémorisation.

On peut dire que pour:

- Les circuits combinatoires à chaque état des variables d'entrées correspond un seul état des variables de sortie et inversement.
- Les circuits séquentiels un état des variables d'entrées peut correspondre à plusieurs états des variables de sortie parce que le circuit se souvient des états précédents.

III. Rappel sur l'algèbre de Boole

III.1 L'algèbre de Boole

Est un ensemble de variables à deux états, de valeur de vérité 1 (vrai), 0 (faux) muni d'un nombre limité d'opérateurs : NON, ET, OU.

III.2 Table de vérité

Les valeurs des variables soumises aux opérateurs pour toutes les combinaisons possibles des valeurs de ces variables sont indiquées sous forme de tables appelées table de vérité.

III.3 Opération NON

Soit X une variable booléenne ; sa négation NON X, appelée aussi complément de X sera noté $\overline{X}$ (X barre), sera aussi une variable booléenne. Par définition X peut prendre deux valeurs 0 ou 1 donc pour l'opération NON la table de vérité est la suivante :

Entrée	Sortie
X	$\overline{X}$
0	1
1	0

Tab 1.1 : Table de vérité de l'opérateur NON (NOT)

On voit que $\overline{X}$ est une variable booléenne « inversée » par rapport à X d'ou le nom d'**inverseur** donné au dispositif effectuant cette opération, appelé aussi l'opérateur d'inversion

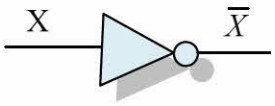


Fig 1.4 : Symbole de l'inverseur (opérateur NON)

III.4 Operation ET (AND)

Soit X et Y deux variables booléennes. Le résultat de X ET Y (X AND Y) est donné selon la table de vérité suivante :

Entrée		Sortie
X	Y	X ET Y
0	0	0
0	1	0
1	0	0
1	1	1

Tab 1.2 Table de vérité de l'opérateur ET (AND)

Le dispositif effectuant cette opération est appelée opérateur (porte) ET (AND) représenté graphiquement (symboliquement) comme suit :

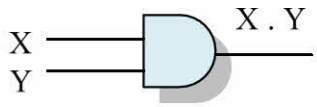


Fig 1.5 : Symbole ET (AND) (porte ET)

III.5 Opération OU (OR)

Soit X et Y deux variables booléennes. Le résultat de X OU Y (X OR Y) est donné selon la table de vérité suivante :

Entrée		Sortie
X	Y	X OU Y
0	0	0
0	1	1
1	0	1
1	1	1

Tab. 1.3 : Table de vérité de l'opérateur OU (OR)

Le dispositif effectuant cette opération est appelée opérateur (porte) OU (OR) représenté graphiquement (symboliquement) comme suit :

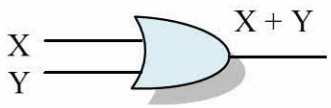


Fig 1.6 : Symbole OU (OR) (porte OU)

III.6 Axiomes ou lois fondamentales de l'algèbre de Boole

Soit A, B, C trois variables booléennes et soit les opérateurs NON, ET et OU ainsi que leurs tables de vérités vue précédemment.

Les lois fondamentales de l'algèbre de Boole sont des règles logiques qui permettent de simplifier l'écriture des expressions algébriques.

III.6.1 Lois de fermeture

a- Le résultat de l'opération Booléenne A . B est une variable booléenne (0 ou 1).

b- Le résultat de l'opération Booléenne A + B est une variable booléenne (0 ou 1).

III.6.2 Lois de commutativité

- A . B = B . A
- A + B = B + A

III.6.3 Lois d'associativité

- (A.B).C = A.(B.C)
- (A+B)+C = A+(B+C)

III.6.4 Lois de distributivité

- A+(B.C) = (A+B).(A+C)
- A.(B+C) = A.B+A.C

III.6.5 Lois d'idempotence

- A.A = A
- A + A = A

III.6.6 Lois de complémentarité

- $A + \overline{A} = 1$
- $A . \overline{A} = 0$

III.6.7 Lois de la double négation (Involution)

- $\overline{\overline{A}} = A$

III.6.8 L'élément neutre

- A . 1 = A donc 1 est l'élément neutre de l'opération (.) AND
- A + 0 = A donc 0 est l'élément neutre de l'opération (+) OU

III.6.9 L'absorption

- A . 0 = 0 donc 0 est l'élément d'absorption de l'opération (.) AND
- A + 1 = 1 donc 1 est l'élément d'absorption de l'opération (+) OU

III.6.10 Théorème de De Morgan

- La négation d'un produit de variables est égale à la somme des négations des variables.

  Exemple : $\overline{xyz}=\overline{x}+\overline{y}+\overline{z}$

- La négation d'une somme de variables est égale au produit des négations des variables.

  Exemple : $\overline{x+y+z}=\overline{x}.\overline{y}.\overline{z}$

III.7 Fonctions logiques

On appel **fonction logique (fonction booléenne)** une entité constitué d'une combinaison de variables booléennes reliées par les opérateurs « NON, ET, OU » acceptant plusieurs valeurs logiques en entrée et dont la sortie (il peut y en avoir plusieurs) peut avoir deux états possibles : 0 ou 1.

Une fonction booléenne peut être définie par :

- Sa table de vérité.
- Son expression.
- Sa table de Karnaugh.
- Son logigramme.

III.7.1 Définition par table de vérité

Une fonction de n variables présente 2n résultats possibles donc on peut décrire cette fonction avec une TV de 2^n lignes, chaque ligne de cette table représente la valeur de la fonction pour une configuration binaire de n variables.

Exemple

M = F(a,b,c) est une fonction booléenne qui prend « 0 » si la majorité des variables vaut « 0 » et « 1 » dans le cas contraire. Donner la TV de la fonction M.

Solution

a	b	c	M
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

Tab 1.4 : Table de vérité de la fonction M

III.7.2 Définition par expression

Il existe une notation plus pratique pour représenter (définir) une fonction booléenne :

- On spécifie toutes les combinaisons des variables d'entrée qui fournissent à la fonction un résultat égal à 1.
- Par convention on place une barre horizontale sur une variable pour indiquer qu'elle vaut 0.
- L'absence de cette barre signifie qu'elle vaut 1.
- On utilise « + » pour exprimer le OU et « . » (ou pas de signe) pour le ET.

Exemple 1

$A\overline{B}+B\overline{C}$ Signifie (A = 1 et B = 0) ou (B = 1 et C = 0)

Exemple 2

Donner l'expression de la fonction booléenne M définie précédemment par la TV (Tab 1.4)

Solution

$$M = \overline{a}bc + a\overline{b}c + ab\overline{c} + abc$$

III.7.3 Définition par Table de Karnaugh

La table de vérité présente un inconvénient : le nombre de ligne croît en même temps que le nombre de variables c.a.d : si on a n variable $\Rightarrow 2^n$ lignes (3 variables $\Rightarrow$ 8 lignes, 4 variables $\Rightarrow$ 16 lignes …etc).

Pour remédier à cet inconvénient on utilise une nouvelle table appeler **Table de Karnaugh (TK).**

- Les lignes et les colonnes doivent être numérotées selon le code binaire réfléchi.
- Le nombre de cases, constituant le tableau, est fonction du nombre de variables mises en jeu

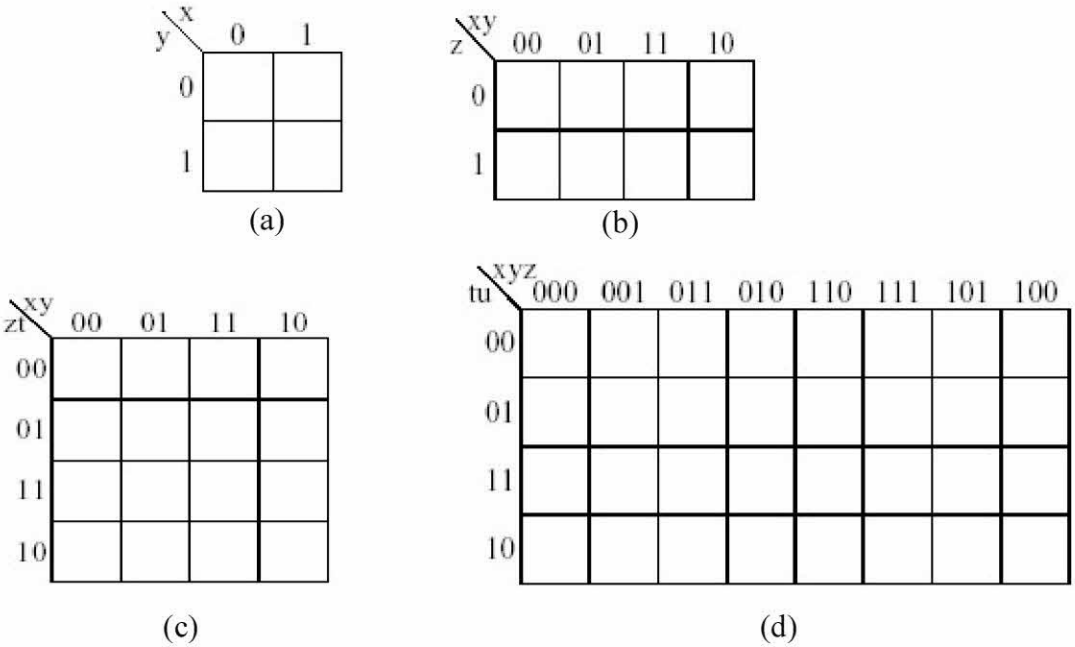


Fig 1.7 : Illustration de quelques tables de Karnaugh (a) Table à 2 variables (b) Table à 3 variables (c) Table à 4 variables (d) Table à 5 variables

Exemple

Soit F = $a\overline{b}+\overline{c}\overline{d}$ donner la table de karnaugh de F

Solution

cd / ab	00	01	11	10
00	**1**	**0**	**0**	**0**
01	**1**	**0**	**0**	**0**
11	**1**	**0**	**0**	**0**
10	**1**	**1**	**1**	**1**

Tab 1.5 : Table de karnaugh de F

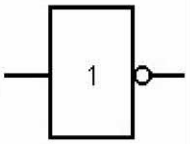
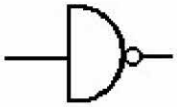
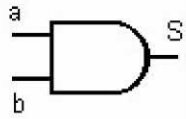
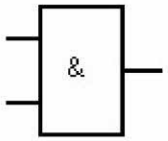
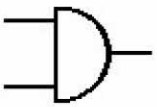
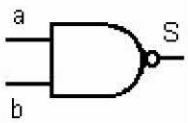
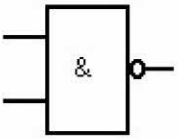
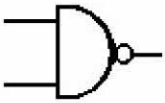
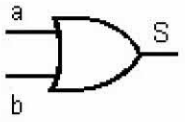
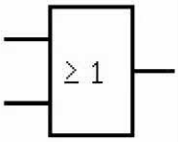
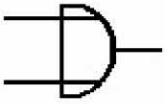
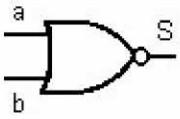
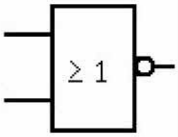
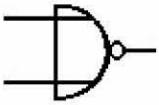
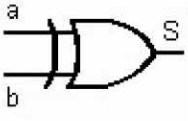
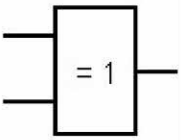
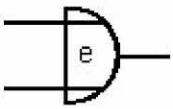
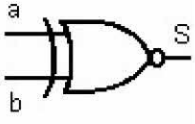
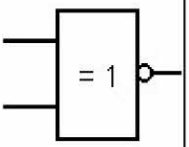
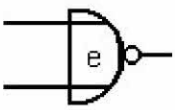
III.7.4 Définition par Logigramme

Connaissant les symboles logiques (voir tableau ci dessous), on peut construire immédiatement le logigramme (digramme logique) d'une fonction booléenne.

NB : Les symboles sur lesquels nous nous basons pour la conception de nos logigrammes sont ceux de la norme internationale.

Remarque

- Les portes ET , OU , NAND , NOR peuvent avoir plus que deux entrées.
- Il n'existe pas de OU exclusif à plus de deux entrées

FONCTION	EQUATION	SYMBOLES			TABLES DE VERITE
		International	Français	Allemand	
NON	$S = \overline{a}$				a S 0 1 1 0
ET	$S = a \cdot b$				a b S 0 0 0 0 1 0 1 0 0 1 1 1
NAND	$S = \overline{a \cdot b}$				a b S 0 0 1 0 1 1 1 0 1 1 1 0
OU	$S = a + b$				a b S 0 0 0 0 1 1 1 0 1 1 1 1
NOR	$S = \overline{a + b}$				a b S 0 0 1 0 1 0 1 0 0 1 1 0
OU Exclusif	$S = a \oplus b$				a b S 0 0 0 0 1 1 1 0 1 1 1 0
NOR Exclusif	$S = \overline{a \oplus b}$				a b S 0 0 1 0 1 0 1 0 0 1 1 1

Tab 1.6 : Table récapitulative des symboles des portes logiques

Exemple

Soit l'expression algébrique de la fonction booléenne S. Dresser le logigramme de S

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

Solution

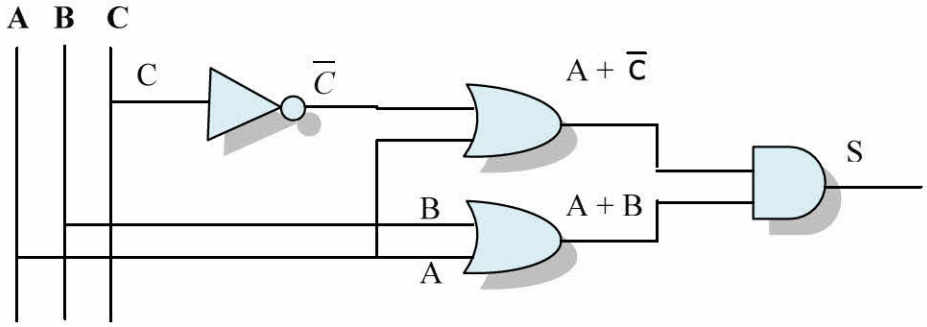


Fig 1.8 : Logigramme de la fonction S

IV. Conclusion

Les circuits logiques sont à la base de la conception de tout système numérique.

Les circuits combinatoires et les circuits séquentiels sont tous les deux complémentaires et essentiels dans les systèmes numériques. Le fonctionnement de ces circuits est basé sur les fondements de l'algèbre de Boole.

Chapitre 2
La logique Combinatoire

Chapitre 2

La Logique Combinatoire

I. Introduction

Les ordinateurs et les systèmes numériques que nous utilisons quotidiennement sont conçus autour de circuits logique dont principalement les circuits combinatoires. Ce sont des circuits ayant comme principal rôle la réalisation des opérations sur des variables logiques (binaires) soumis à des règles de la logique combinatoire.

L'objectif principal de ce chapitre est d'apprendre à synthétiser des circuits combinatoires arithmétiques, de transcodage, de multiplexage et de comparaison en utilisant les principes de l'algèbre de Boole et les portes logiques.

Les références utilisées pour l'élaboration de ce chapitre sont principalement : [3] [4] [5] et [6].

II. Circuits combinatoires

Les circuits combinatoires sont des circuits dont la fonction de sortie s'exprime selon une expression logique des variables d'entrées seulement et où le temps de propagation n'est pas pris en considération.

III. Conception d'un circuit combinatoire

La conception ou la synthèse d'un circuit combinatoire passe par les étapes suivantes :

1. Extraction de la fonction logique par table de vérité.
2. Simplification de la fonction logique par expression ou table de Karnaugh.
3. Implantation de la fonction logique en utilisant un minimum de portes logiques.

On peut classer ces circuits en 4 catégories : opérateurs arithmétiques, opérateurs de transcodage, opérateurs d'aiguillage, opérateurs de comparaison.

IV. Opérateurs Arithmétiques

IV.1 Demi-additionneur

On veut concevoir un circuit capable d'additionner deux bits donc il est capable de générer à la sortie la somme et le report (retenue).

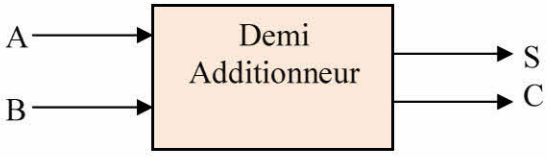


Fig 0.1 : Demi-additionneur

a. Table de vérité

A et B les deux variables d'entrées représentant les bits à additionner. « S » est la somme des deux variables et « C » la retenue (ou carry) correspondante. La TV est la suivante :

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

Tab 0.1 : Table de vérité du Demi-additionneur

b. La mise en équation

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

$C = A.B$

c. Implantation (logigrammes)

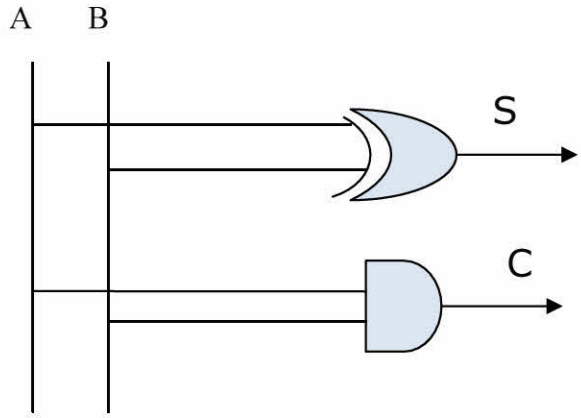


Fig 0.2: logigramme du Demi-Additionneur

IV.2 Additionneur complet

Pour additionner deux nombres binaires il faut tenir compte du report (retenu ou carry) de l'étage précédant (bit précédant). On peut concevoir un additionneur complet ayant comme entrée A_i, B_i, C_{i-1} (retenue de l'étage précédant) et comme sortie S_i et C_i (représentant le résultat de la somme et la retenus de l'étage correspondant).

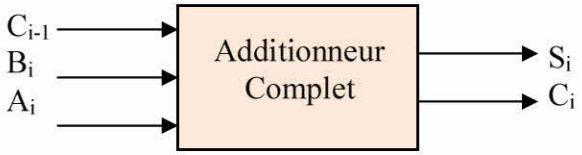


Fig 0.3: Additionneur Complet

a. Table de vérité

C_{i-1}	B_i	A_i	S_i	C_i
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

Tab 0.2: Table de vérité de l'additionneur complet

b. Mise en équation

$S_i = C_{i-1} \oplus B_i \oplus A_i$

$C_i = A_i B_i + C_{i-1}(A_i \oplus B_i)$

c. Logigramme

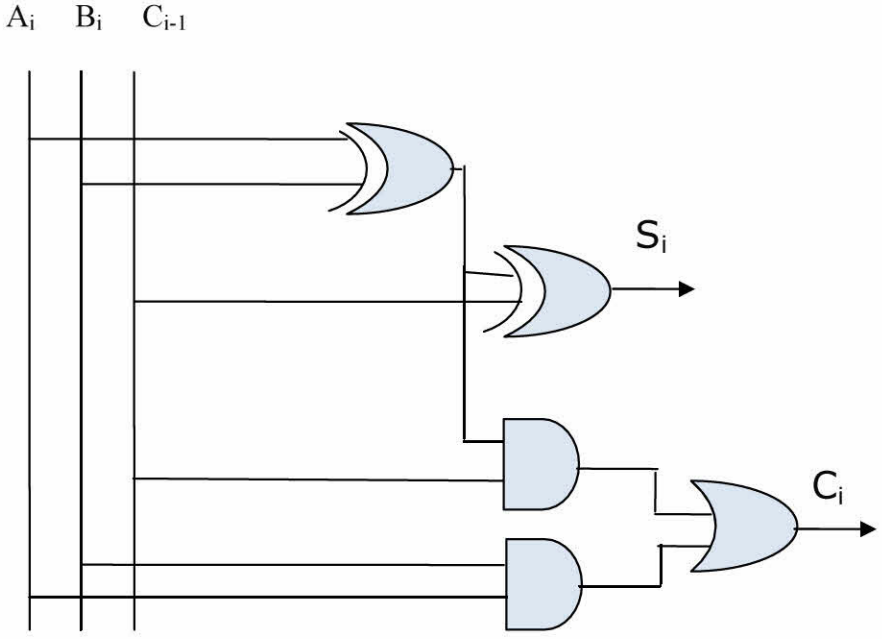


Fig 0.4: Logigramme de l'additionneur complet

IV.3 Demi-soustracteur

On veut concevoir un circuit capable de soustraire un bit d'un bit, il nous donne donc en sortie : leur différence « D » et leur retenue « C ».

Soit A et B deux bits, la différence D = A – B et la retenue C.

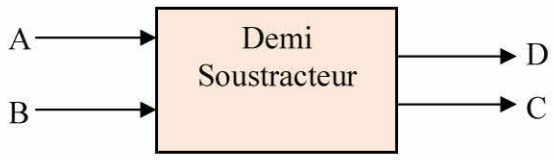


Fig 0.5: Demi-Soustracteur

a. Table de vérité

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

Tab 0.3: Table de vérité du Demi-soustracteur

b. Mise en équation

$D = \overline{A}B + A\overline{B} = A \oplus B$

$C = \overline{A}B$

c. Logigramme

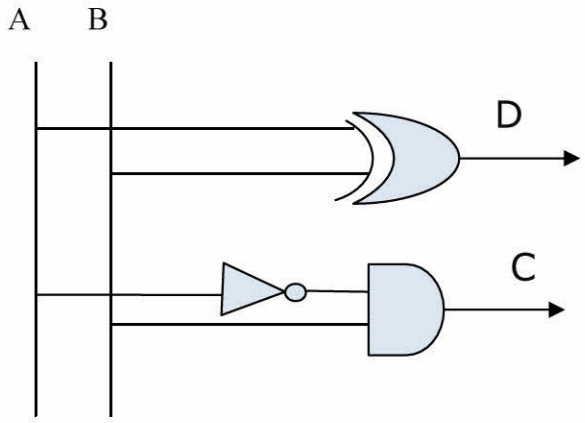


Fig 0.6 : logigramme du Demi-soustracteur

IV.4 Soustracteur complet

Le but c'est d'avoir la différence D_i de deux bits avec $D_i = A_i - B_i$ en tenant compte de la retenue C_{i-1} de l'étage précédant (i-1) et générer la retenue de l'étage correspondant C_i.

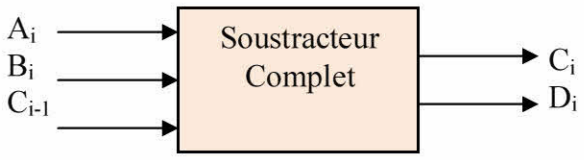


Fig 0.7 : Soustracteur Complet

a. Table de vérité

A_i	B_i	C_{i-1}	D_i	C_i
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Tab 0.4: Table de vérité du soustracteur complet

b. Mise en équation

$$D_i = A_i \oplus B_i \oplus C_{i-1}$$

$$C_i = \overline{A}_i (B_i \oplus C_{i-1}) + B_i C_{i-1}$$

c. Logigramme

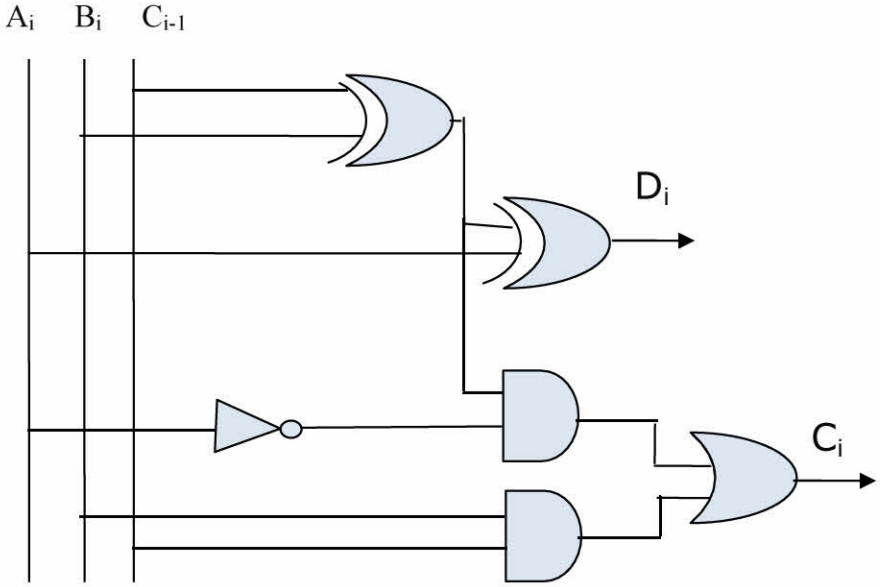


Fig 0.8: Logigramme du soustracteur complet

V. Opérateurs de Transcodage

Un opérateur de transcodage est un circuit transformant une information présente en entrée sous une forme (code 1) en la même information en sortie mais sous une autre forme (code 2).

V.1 Les codeurs (Encodeurs)

V.1.1 Définition

L'encodeur (codeur) est un circuit combinatoire à $\mathbf{2^N}$ entrées et **N** sorties. Une seule entrée est active à la fois et le code binaire équivalent au numéro de cette entrée est délivré sur les **N** sorties. Ce circuit traduit donc le rang de l'entrée active en un code binaire à la sortie.

Il faut noter que dans ce type de circuit seulement une entrée est « 1 » à la fois et les autres sont à « 0 » c'est le format d'entrées « **One-hot** »

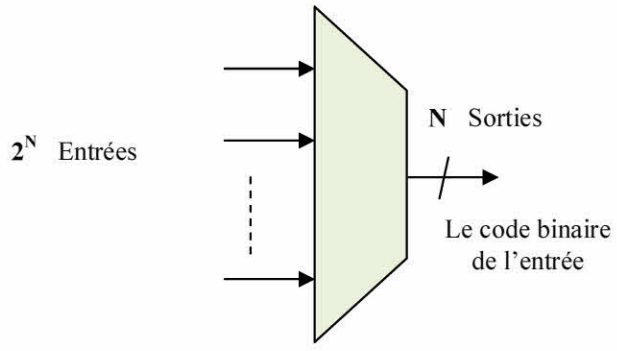


Fig 0.9 : Codeur (Encodeur)

V.1.2 Principe d'un encodeur (4 entrées vers 2 sorties)

Afin de comprendre le fonctionnement d'un encodeur prenons à titre d'exemple la réalisation d'un encodeur avec 4 entrées et 2 sorties (Fig 2.10).

a. Table de vérité

e_3	e_2	e_1	e_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

Tab 0.5 : Table de vérité du codeur 4 vers 2

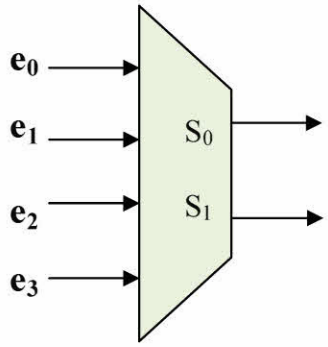


Fig 0.10: Codeur 4 vers 2

b. Mise en équation

$S_1 = 1$ si $(e_3 = 1)$ ou $(e_2 = 1)$; $S_1 = e_3 + e_2$

$S_0 = 1$ si $(e_1 = 1)$ ou $(e_3 = 1)$; $S_0 = e_1 + e_3$

c. Logigramme

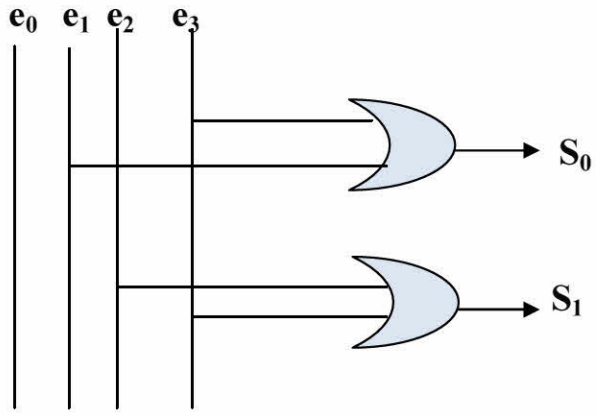


Fig 0.11: Logigramme du codeur 4 vers 2

Remarque

- Si nous activons simultanément les entrées e_2 et e_1 les sorties S_0 S_1 présenteront la combinaison 11 qui ne correspond pas au code de l'une ou de l'autre des entrées activées. C'est plutôt le code qui représente l'activation de e_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. Si e_2 et e_1 sont activent en même temps les sorties S_1 S_0 présenteront la combinaison 10 qui correspond à l'activation de l'entrée e_2.

V.2 Les encodeurs de priorité

V.2.1 Définition

Ce type de codeur fixe un ordre de priorité entre les entrées. Pour un codage en binaire pur, le codeur prioritaire donne en principe la priorité à l'entrée de poids le plus élevé.

V.2.2 Principe d'un encodeur de priorité (4 entrées vers 2 sorties)

Afin de comprendre le fonctionnement d'un encodeur de priorité prenons à titre d'exemple la réalisation d'un encodeur de priorité avec 4 entrées et 2 sorties. La table de vérité décrivant le fonctionnement de l'encodeur de priorité est illustrée sur la table (Tab 2.6).

a. Table de vérité

e_3	e_2	e_1	e_0	S_1	S_0
0	0	0	0	0	0
0	0	0	1	0	0
0	0	1	0	0	1
0	0	1	1	0	1
0	1	0	0	1	0
0	1	0	1	1	0
0	1	1	0	1	0
0	1	1	1	1	0
1	0	0	0	1	1
1	0	0	1	1	1
1	0	1	0	1	1
1	0	1	1	1	1
1	1	0	0	1	1
1	1	0	1	1	1
1	1	1	0	1	1
1	1	1	1	1	1

⇒

e_3	e_2	e_1	e_0	S_1	S_0
0	0	0	0	0	0
0	0	0	1	0	0
0	0	1	x	0	1
0	1	x	x	1	0
1	x	x	x	1	1

Tab 0.6: Table de vérité de l'encodeur de priorité et sa simplification

b. Mise en équation

e_3 e_2 \ e_1 e_0	00	01	11	10
00	0	0	1	1
01	0	0	0	0
11	1	1	1	1
10	1	1	1	1

Tab 0.7: Simplification par tables de Karnaugh de la sortie S_0 de l'encodeur de priorité

$$S_0 = e_3 + \overline{e_2}\, e_1$$

e_3 e_2 \ e_1 e_0	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

Tab 0.8: Simplification par tables de Karnaugh de la sortie S_1 de l'encodeur de priorité

$$S_1 = e_3 + e_2$$

c. Logigramme

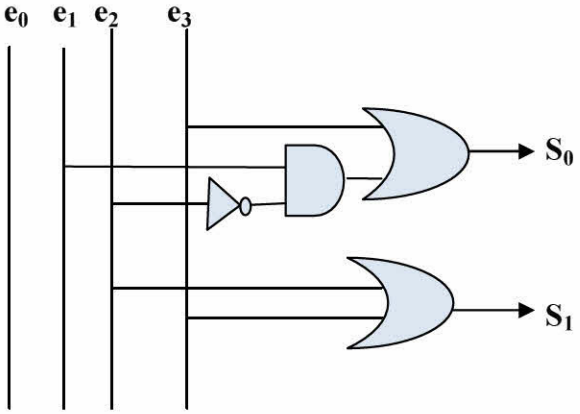


Fig 0.12: Logigramme de l'encodeur de priorité

Remarque

- En générale le nombre d'entrées d'un codeur respecte la condition : entrées $\leq 2^N$
- Les encodeurs et les encodeurs de priorité peuvent être appliqués dans : le clavier la calculatrice, la télécommande ; l'ordinateur : clavier (texte/ASCII),…etc.

V.3 Les décodeurs

V.3.1 Définition

Un décodeur est un circuit combinatoire constitué de **N** entrées binaire ayant 2^N combinaisons et **M** sorties telles que $\mathbf{M \leq 2^N}$.

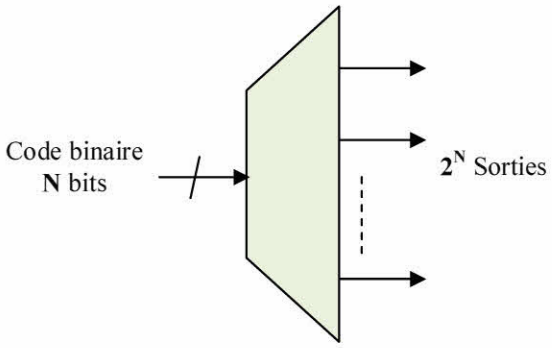


Fig 0.13: Décodeur

V.3.2 Principe d'un décodeur (1 parmi 4)

Afin de comprendre le fonctionnement d'un décodeur prenons à titre d'exemple la réalisation d'un décodeur un parmi quatre sorties (Fig 2.14)

Pour pouvoir activer les 4 sorties on a besoin de 2 bits à l'entrée ($4 = 2^2$).

La table de vérité illustrant le fonctionnement du décodeur est donnée par Tab 2.9.

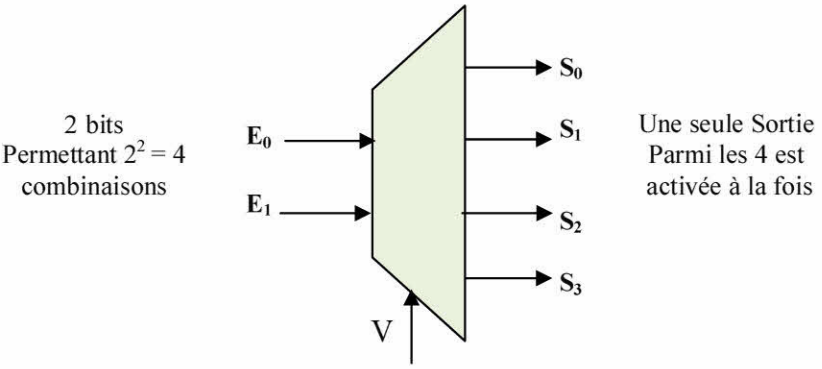


Fig 0.14: Décodeur 1 parmi 4

a. Table de vérité

V	E_1	E_0	S_3	S_2	S_1	S_0
0	x	x	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

Tab 0.7: Table de vérité du décodeur 1 parmi 4

b. Mise en équation

$S_0 = V(\overline{E}_1\overline{E}_0)$; $S_1 = (\overline{E}_1 E_0)$; $S_2 = V(E_1\overline{E}_0)$; $S_3 = V(E_1 E_0)$

c. Logigramme

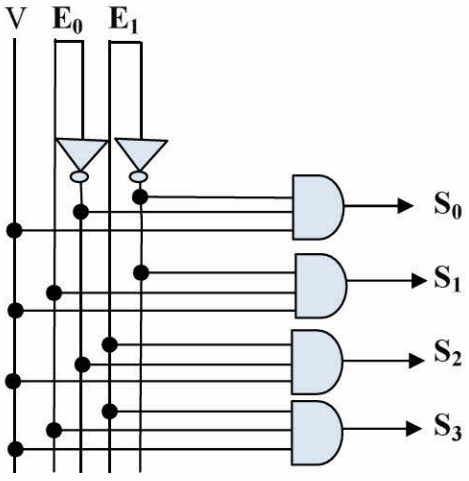


Fig 0.15: Logigramme du décodeur 1 parmi 4

Remarque

Un décodeur peut être utilisé de deux manières où la sortie peut traduire la fonction:

- De convertisseur de code : à un code d'entrée correspond un code de sortie. Ex : décodeur binaire – octal, binaire-décimale, …etc

- Sélecteur de sortie: Une seule sortie parmi les M disponibles est activée à la fois en fonction de la valeur binaire affichée à l'entrée.

V.4 Les transcodeurs

V.4.1 Définition

Un transcodeur est un circuit combinatoire qui transforme un code X (sur **n** bits) en entrée en un autre code **Y** (sur **m** bits) en sortie.

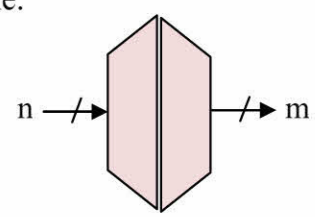


Fig 0.16: Transcodeur

V.4.2 Principe d'un transcodeur Binaire-Gray à 2 bits

Nous voulons réaliser un transcodeur Binaire-Gray à deux bits. La table de vérité illustrant le fonctionnement du transcodeur est donnée par la table (Tab 2.10).

a. Table de Vérité

Binaire		Gray	
B_1	B_0	G_1	G_0
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Tab 0.8: Table de vérité du transcodeur Binaire-Gray à 2 bits

b. Mise en équation

$G_1 = B_1$; $G_0 = B_1 \oplus B_0$

c. Logigramme

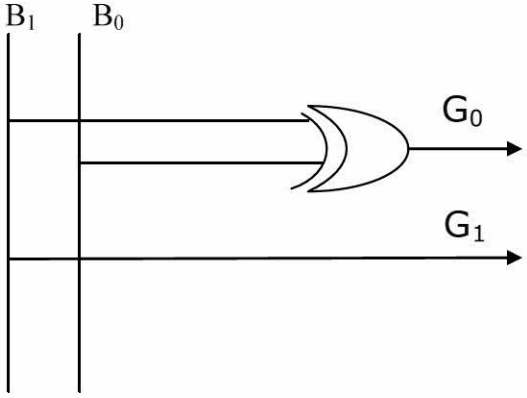


Fig 0.17: Logigramme du transcodeur Binaire-Gray à 2 bits

VI. Opérateurs d'Aiguillage

Le multiplexage (aiguillage) est un dispositif qui permet de transmettre sur une seule ligne des informations en provenance de plusieurs sources ou à destination de plusieurs cibles.

Multiplexeur Démultiplexeur

Fig 0.18 : Multiplexage

VI.1 Multiplexeur

Ce circuit sélectionne une entrée parmi **N** et transmet l'information portée par cette ligne à un seul canal de sortie S.

VI.1.1 Exemple de multiplexeur 4 vers 1

Considérons un multiplexeur à 4 entrées (4 lignes d'entrées)

$\mathbf{4 = 2^2} \Rightarrow$ on aura 2 lignes d'adresse (A et B)

On veut avoir à la sortie S l'état des lignes E_0 E_1 E_2 E_3.

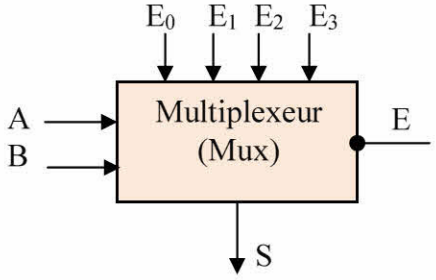


Fig 0.19: Multiplexeur 4 vers 1

La TV de S est la suivante :

A	B	S
0	0	E_0
0	1	E_1
1	0	E_2
1	1	E_3

Tab 0.9 : Table de vérité du multiplexeur 4 vers 1

- Supposons que nous souhaitons également valider les données avec un signal de contrôle E (Enable).
- Par convention nous choisissons de prendre en compte les données pour E = 0 (active à l'état bas)

a. Table de vérité

La TV de S devient :

A	B	E	S
x	x	1	0
0	0	0	E_0
0	1	0	E_1
1	0	0	E_2
1	1	0	E_3

Tab 0.10: Table de vérité du multiplexeur 4 vers 1 avec l'entrée de validation

b. Mise en équation

$$S = \overline{E}\,\overline{A}.\overline{B}.E_0 + \overline{E}.A.B.E_1 + \overline{E}\,A.\overline{B}.E_2 + \overline{E}\,A.B.E_3$$

c. Logigramme

L'expression de S correspond au logigramme de la figure 2.20.

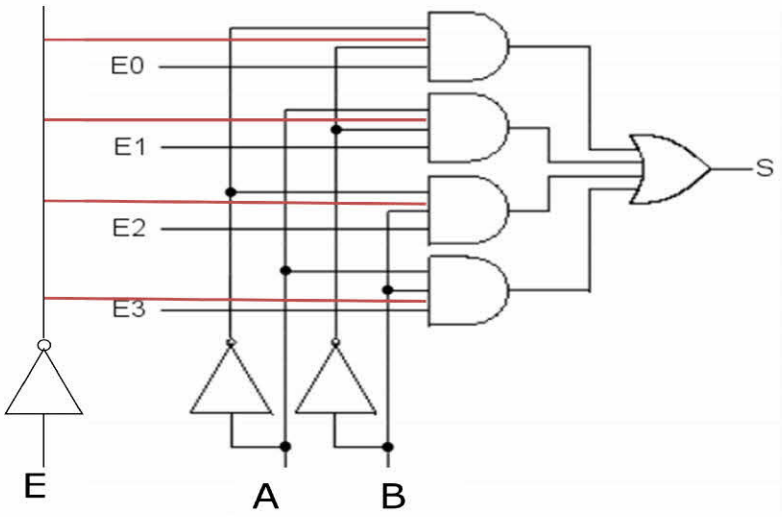


Fig 0.20: Logigramme du multiplexeur 4 vers 1

VI.2 Démultiplexeur

C'est un circuit comportant une entrée et N sorties et qui met en relation cette dernière (entrée) avec une sortie et une seule. Pour pouvoir sélectionner cette sortie il faut également des lignes d'adressages : le code porté par ces lignes identifie les lignes de sortie à utiliser.

VI.2.1 Exemple de démultiplexeur 1 vers 4

Considérons un Démultiplexeur à 4 lignes de sorties. Supposons aussi que nous souhaitons également valider les données avec un signal de contrôle E. Par convention nous choisissons de prendre en compte les données pour E = 0.

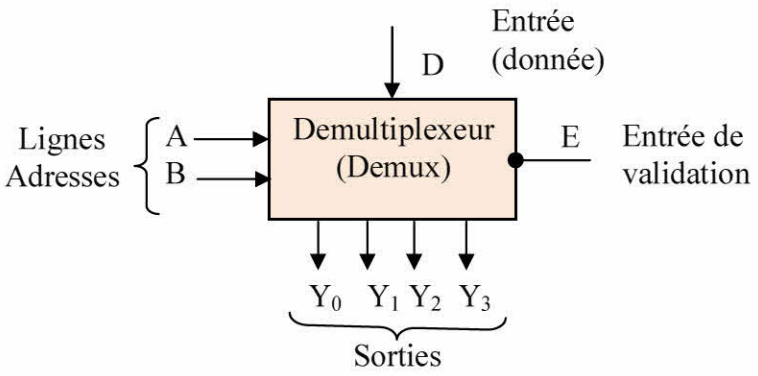


Fig 0.21: Démultiplexeur 1 vers 4

a. Table de vérité

A	B	E	Y_0	Y_1	Y_2	Y_3	
x	x	1	0	0	0	0	
0	0	0	D	0	0	0	$Y_0 = \overline{A}\overline{B}\overline{E}D$
0	1	0	0	D	0	0	$Y_1 = \overline{A}B\overline{E}D$
1	0	0	0	0	D	0	$Y_2 = A\overline{B}\overline{E}D$
1	1	0	0	0	0	D	$Y_3 = AB\overline{E}D$

Tab 0.11 : Table de vérité du démultiplexeur 1 vers 4

b. Logigramme

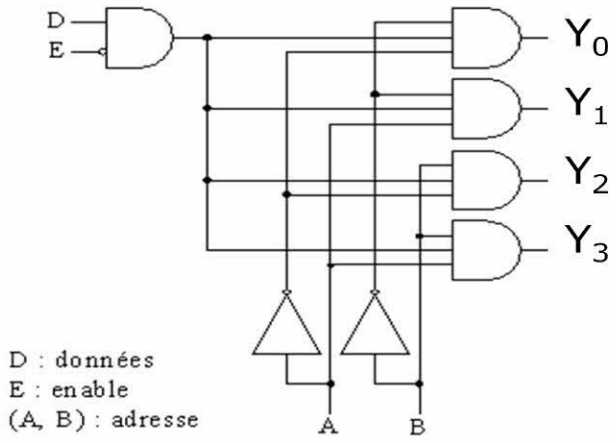


Fig 0.22: Logigramme du démultiplexeur 1 vers 4

Remarque

Les Démultiplexeurs existent sous forme de circuits intégrés avec 2, 4, 16 lignes de sorties, pour constituer des Démultiplexeurs d'ordre supérieur on peut les cascader.

VII. Opérateurs de comparaison

VII.1 Définition

Le comparateur est un opérateur capable de détecter l'égalité et de comparer deux nombres.

VII.2 Exemple de comparateur à un bit

Considérons un circuit de comparaison de deux nombres A et B de 1 bit chacun.

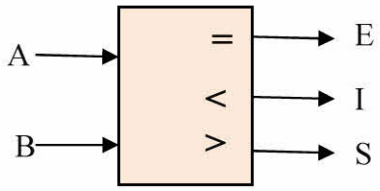


Fig 0.23 : Comparateur à un bit

a. Table de vérité

A	B	E	I	S
0	0	1	0	0
0	1	0	1	0
1	0	0	0	1
1	1	1	0	0

Tab 0.12 : Table de vérité du comparateur à un bit

b. Mise en équation

$E = \overline{A}\overline{B} + AB = \overline{A \oplus B}$

$I = \overline{A}B$

$S = A\overline{B}$

c. Logigramme

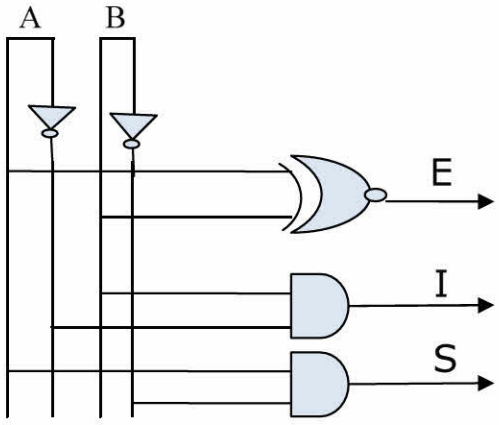


Fig 0.24 : Logigramme du comparateur à un bit

VIII. Conclusion

Les circuits combinatoires étudiés peuvent être classés en quatre classes : les circuits arithmétiques, les circuits de transcodage, les circuits de multiplexage et les circuits de comparaison. D'autres circuits combinatoires seront étudiés dans les exercices proposés dans ce chapitre.

Exercices

Exercice 2.1

1. Soit le schéma bloc d'un additionneur à 2 bits où A (a_1 a_0) et B (b_1 b_0) sont deux nombres constitués de 2 bits chacun et S (S_1 S_0) et C sont respectivement la somme et la retenue.
   a. Tracer la table de vérité du circuit additionneur.
   b. Donner les expressions de sorties simplifiées par table de karnaugh et la méthode algébrique si nécessaire.
   c. Réaliser l'architecture interne (logigramme) de ce circuit.
2. On suppose que le circuit possède une troisième entrée de retenue antérieure C_{i-1}(figure 2), dresser le logigramme d'un additionneur complet de deux nombres à 8 bits chacun en utilisant le circuit de la figure 2.

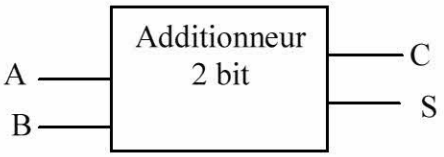


Figure 1

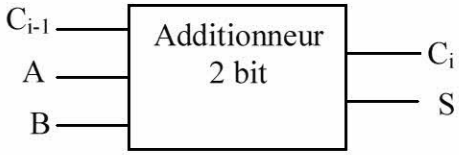


Figure 2

Exercice 2. 2

Soit le schéma bloc d'un circuit codeur Octal-Binaire de la figure 3.

Où V est une entrée de validation du circuit.

1. Tracer la table de vérité reliant les entrées et sorties de ce circuit.
2. Donner les expressions de sorties.
3. Dresser le logigramme de ce circuit.

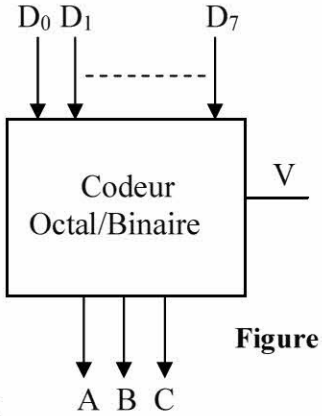


Figure 3

Exercice 2.3

Les afficheurs 7-segments sont un type d'afficheur particulièrement présent sur les circuits à affichage numérique (montres, calculatrices,…etc).

Les caractères s'écrivent en allumant ou en éteignant des segments, au nombre de sept. Les segments sont le plus souvent désignés par les lettres : a, b, c, d, e, f, g (figure 4).

Nous voulons réaliser un circuit qui fait le décodage du code BCD vers le code 7 segments (figure 5).

1. Dresser la TV décrivant le fonctionnement du décodeur.
2. Donner les expressions simplifiées des 7 sorties du décodeur en utilisant la table de karnaugh.

3. Dresser le diagramme logique du décodeur.

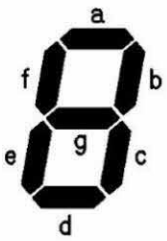


Figure 4

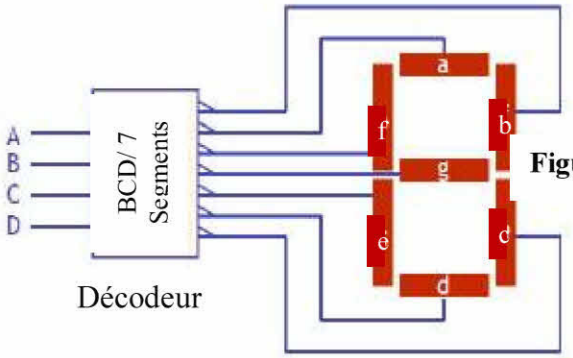


Figure 5

Exercice 2.4

Soit le circuit transcodeur BCD /Excess-3 de la figure 6 :

1. Dresser la table de vérité définissant le fonctionnement de ce circuit.
2. Donner les expressions des sorties x, y, z, t simplifiées.
3. Dresser le logigramme du transcodeur avec un minimum de portes logiques

NB : rappelons que le code Excess-3 = BCD + 3.

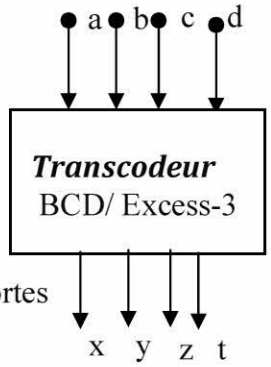


Figure 6

Exercice 2.5

Soit le circuit d'aiguillage représenter par la figure 7 :

1. Comment appel-t-on ce circuit et quel est le rôle de chacune de ses entrées et sortie ?
2. Dresser la table de vérité et donner l'expression logique de la sortie de ce circuit.
3. Dresser le logigramme de ce circuit.
4. Soit F une fonction a trois variables définie par l'expression suivante : $F(x,y,z) = \sum(1,3,5,7)$. Réaliser cette fonction en utilisant ce circuit.
5. Réaliser ce circuit en utilisant des Mux 1 parmi 4.

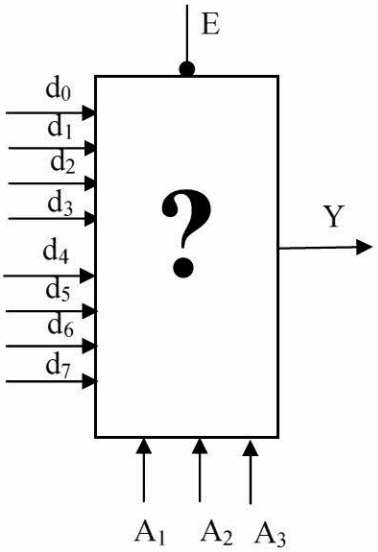


Figure 7

Exercice 2.6

Soit le schéma symbolique d'un démultiplexeur de la figure 8.

1. Le circuit sous cette forme ne fonctionne pas correctement pourquoi ? Proposer une solution et dresser le nouveau circuit.
2. Dresser la table de vérité du circuit.

3. En déduire les expressions des sorties Y_i.
4. Dresser le logigramme du DEMUX.
5. Utilisez ce démultiplexeur pour réaliser la fonction logique $F_1(a,b,c,d) = \sum(0,2,4,6,8,10,13,14,15)$
6. Refaire la question (4) pour $F_2(a,b,c) = \sum(1,2,3,5,7)$

NB : Prendre le MSB la variable A_2

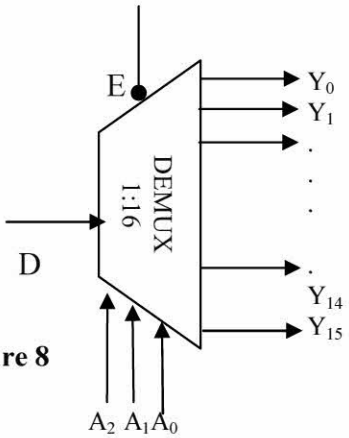


Figure 8

Exercice 2.7

Soit A et B deux nombres binaires à deux bits chacun où $A = a_1a_0$ et $B = b_1b_0$. Nous voulons réaliser le circuit comparateur (Figure 9) à deux bits dont les entrées sont respectivement les nombres A et B et les sorties sont E, I, S où :

- E (égalités) = 1 si A = B
- I (Inférieur) = 1 si A < B
- S (Supérieur) = 1 si A > B

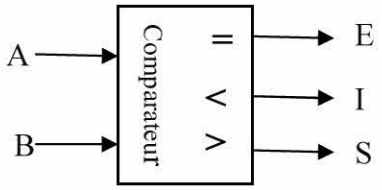


Figure 9

1. Dresser la table de vérité du comparateur.
2. Donner les expressions simplifiées de E, I et S.
3. Dresser le logigramme du comparateur en utilisant des portes à deux entrées.

Exercice 2.8

Un distributeur de boissons chaudes (figure 10) permet de distribuer du café ou du thé, avec ou sans lait, ou du lait seul. Son fonctionnement est décrit comme suit :

- Trois boutons permettent de commander le distributeur : « café » B1, « thé » B2, «lait» B3.
- Pour obtenir l'une de ces boissons seules, il suffit d'appuyer sur le bouton correspondant.
- Pour obtenir une boisson avec lait, il faut appuyer en même temps sur le bouton correspondant à la boisson choisie et sur le bouton « lait ».
- De plus, le distributeur ne fonctionne que si une pièce de 20 DA a préalablement été introduite dans la fente (P) de l'appareil.
- Une fausse manœuvre après introduction du jeton (par exemple, appui simultané sur « café » et « thé ») provoque la restitution de la pièce de 20 DA (R).
- Le lait étant gratuit, la pièce de 20 DA est également restituée si du lait seul est choisi.

1. Dresser la Table de vérité du distributeur.
2. Simplifier par Karnaugh puis en déduire les expressions des fonctions : de restitution de la pièce de 20 DA (**R**), de distribution du café (**C**), du thé (**T**), et du lait (**L**).
3. Etablir le logigramme du distributeur à base de portes logique NAND.

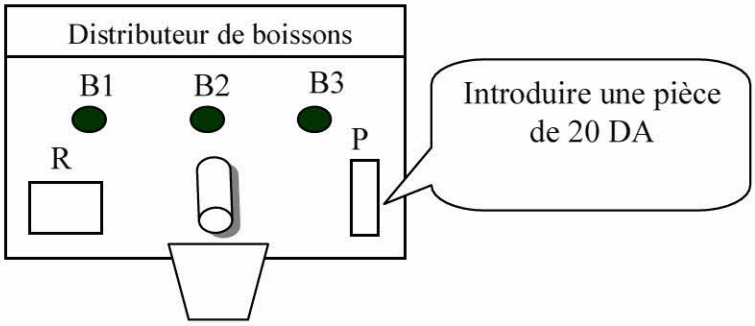


Figure 10

Exercice 2.9

Trois interrupteurs I_1, I_2, et I_3 commandent le démarrage de deux moteurs M_1 et M_2 selon les Conditions suivantes :

- Le moteur M_1 ne doit démarrer que si au moins deux interrupteurs sont fermés ($I_i = 1$),
- Dès qu'un ou plusieurs interrupteurs sont activés, le moteur M_2 doit démarrer.

1. Etablir la T.V des Fonctions représentant les moteurs M_1 et M_2.
2. Simplifier par table de karnaugh M_1 et M_2
3. Etablir le logigramme de M_1 et M_2 a base de portes NAND (NON ET) seulement.
4. Refaire la question 2 en utilisant des portes NOR (NON OU) seulement.

Exercice 2.10

Soit le dé électronique (à 7 diodes LED) représenté par la figure 11.a.

On veut réaliser le circuit logique de commande du « dé » pour allumer les diodes, qui comporte 3 entrées en code binaires (x,y,z) et 7 sorties (a,b,c,d,e,f,g) reliés aux sept diodes du « dé » (figure 11.c).

Les différentes combinaisons d'affichage du dé sont représentées par la figure 1.b. A titre d'exemple, si on veut afficher 2, il faut allumer les diodes a et g (fig 11.b).

On note que pour les combinaisons d'entrée 0 (000) et 7 (111) aucune diode ne doit être allumée.

1. Dresser la table de vérité de fonctionnement du circuit de commande.
2. Donner les expressions des premières formes canoniques des sorties a, b, c, d
3. Donner les expressions simplifiées de sorties (a, b, c, d, e, f, g) du circuit de commande (utiliser la table de karnaugh et/ou la simplification algébrique si nécessaire)
4. Dresser le logigramme du circuit de commande à base de portes logiques NAND.

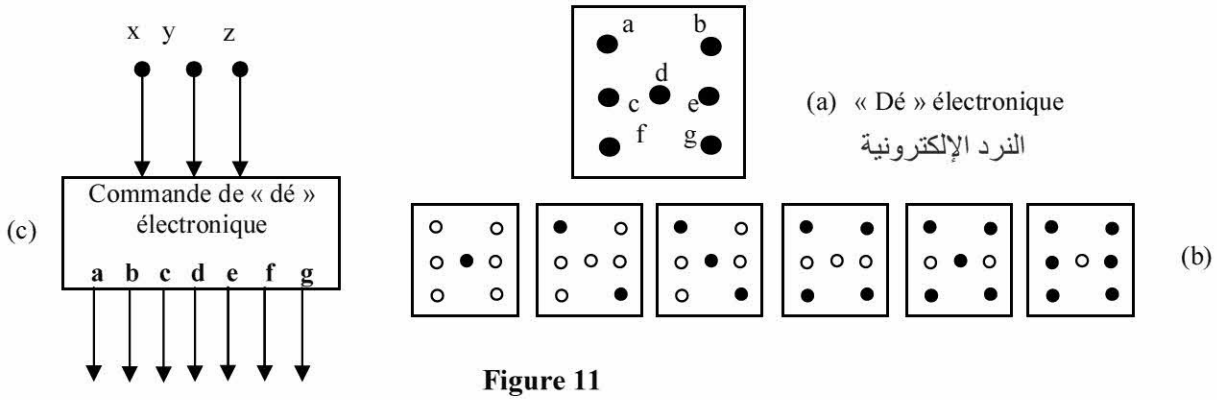


Figure 11

Exercice 2.11

Nous voulons concevoir un circuit de contrôle (Figure 12) de 4 lignes téléphonique (A, B, C, D) d'une société privée.

Les 4 lignes téléphoniques doivent être surveillées comme suit :

Sur le tableau d'affichage (figure 2 se trouve quatre lampes V (vert), J (Jaune), O (Orange), R (Rouge) et un bouton poussoir M (Marche/Arrêt) :

- Lampe V : s'allume lorsqu'aucune ligne n'est occupée.
- Lampe J : s'allume lorsqu'une minorité des lignes sont occupées.
- Lampe O : s'allume lorsque la moitié des lignes sont occupées.
- Lampe R : s'allume lorsque la majorité ou toutes les lignes sont occupées.
- Le contrôleur de lignes téléphoniques ne fonctionne que si l'utilisateur appuis sur le bouton M (Marche (M = 1)/Arrêt (M = 0)).

1. Etablir la T.V des Fonctions représentant les lampes de surveillance.
2. Simplifier puis en déduire les expressions algébriques des fonctions de sorties (lampes de surveillance)
3. Réaliser le logigramme du contrôleur à base de portes NAND (en utilisant des portes NAND (NON-ET) seulement).

Noter que :

- Si une lampe est allumée, les autres sont éteintes.
- Une ligne occupée vaut 1.

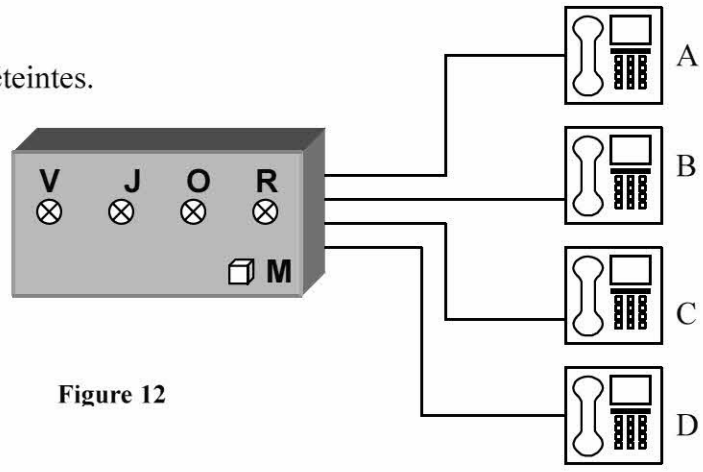


Figure 12

Chapitre 3
La logique séquentielle

Chapitre 3

La logique Séquentielle

I. Introduction

Les circuits logiques combinatoires (vu dans le chapitre 2) ne suffisent pas à eux seuls à la manipulation de l'information dans les systèmes numériques où une forme de mémoire est toujours requise pour permettant au circuit de se souvenir des évènements passés et de traiter l'information plus adéquatement. Cette caractéristique est particulièrement présente dans les circuits séquentiels. Les circuits dits séquentiels sont réalisés à base de cellules mémoire appelées bascules.

L'objectif principale de ce chapitre est d'apprendre à analyser et a syntétiser les circuits séquentiels tel que les registres et les compteurs.

Les références utilisées pour l'élaboration de ce chapitre sont principalement : [5] [7] [8] et [9].

II. Les bascules

La bascule (Latch) est l'élément de base de la logique séquentielle, elle est constitué essentiellement de :

- Un certain nombre d'entrés X_1, X_2,….
- Une sortie Q ou deux sorties complémentaires : Q et $\overline{Q}$

On trouve deux types de fonctionnement de la bascule : synchrone et asynchrone

II.1 Fonctionnement Asynchrone

C'est un mode de fonctionnement où la sortie change d'état uniquement en fonction des grandeurs d'entrée.

II.2 Fonctionnement Synchrone

C'est un mode de fonctionnement où le changement d'état est conditionné par une autorisation donnée par le signal d'**Horloge.** Dans ce mode l'autorisation peut être donnée de trois façons différentes : sur niveau, sur front ou par impulsion.

a- Synchronisation sur niveau

Il suffit d'appliquer le niveau convenable appelé niveau actif pour que la sortie de la bascule puisse changer.

b- Synchronisation sur front

La durée de l'autorisation est réduite à son minimum (c.a.d le temps pour que le signal d'horloge passe d'un niveau a un autre). La sortie prendra la valeur correspondante aux entrées au moment du front actif.

c- Synchronisation par impulsion

Une impulsion est composée de deux fronts : le premier front sert à la synchronisation des entrées, le second à la synchronisation de la sortie.

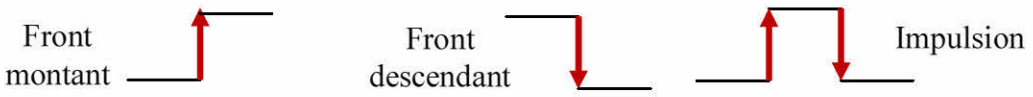


Fig 3.1 : Types de synchronisation d'une bascule

III. Les bascules Asynchrones

III.1 Bascule RS

La bascule RS est la seule bascule asynchrone existante. Elle dispose de deux entrées R et S l'une pour la mise à « 0 » l'autre pour la mise à « 1 » de la sortie **Q**

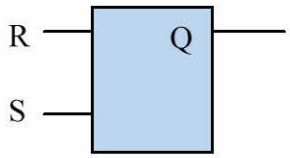


Fig 3.2 : Symbole de la bascule RS

R_t	S_t	Q_{t+}	Fonctionnement
0	0	Q_t	Mémoire
0	1	1	Mise à 1
1	0	0	Mise à 0
1	1		Interdit

Tab 3.1 : Table 1 de fonctionnement de la bascule RS

- La table de fonctionnement à la différence de la table de vérité fait intervenir la notion de temps.

D'après cette table on distingue 3 modes de fonctionnement de la bascule RS :

- Le mode Mémoire « mémorisation » : R = S = 0, la sortie reste dans l'état où elle était (0 ou 1)
- L'écriture d'un 1 ou la mise à 1 de la sortie : S = 1 et R = 0.
- L'écriture d'un 0 ou la mise à zéro de la sortie : S = 0 et R = 1.

Remarque

La combinaison R = S = 1 n'est pas utilisable puisqu'elle conduit en même temps à la mise à 1 et à 0 de la sortie.

Exemple

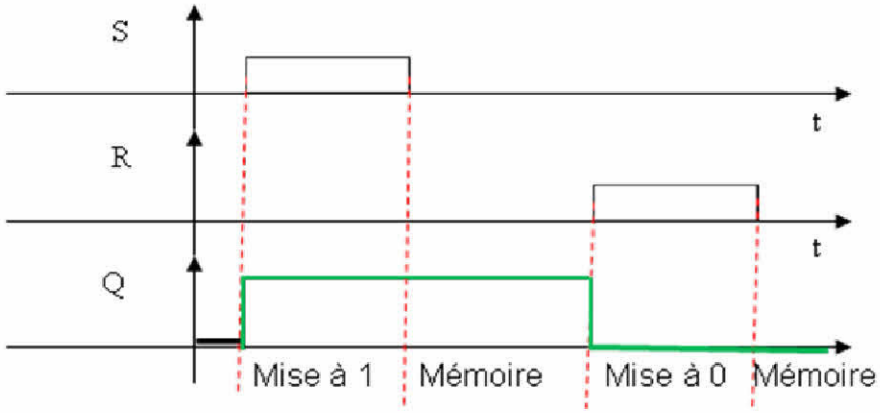


Fig 3.3 : Exemple de chronogramme de la bascule RS

On peut dresser la table de fonctionnement d'une autre manière :

R	S	Q_t	Q_{t+}	
0	0	0	0	Mémoire
0	0	1	1	
0	1	0	1	Mise à 1
0	1	1	1	
1	0	0	0	Mise à 0
1	0	1	0	
1	1	0		Interdits
1	1	1		

$Q_{t+} = f(R,S,Q_t)$
Avec:
Q_{t+}:état présent
Q_t :état passé

Tab 3.2 : Table 2 de fonctionnement de la bascule RS

IV. Les bascules Synchrones

IV.1 Les bascules RS synchronisées ou RST

Cette bascule synchronisée par le niveau haut de l'horloge (T) comporte deux fonctionnements distincts :

- T = 0 : La sortie ne change pas quelles que soient les entrées R et S, c'est le fonctionnement en mémoire. La bascule n'est pas synchronisée.
- T = 1 : La bascule est alors synchronisée : la sortie respects la table de fonctionnement de la bascule RS.

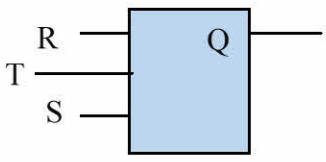


Fig 3.4 : Symbole de la bascule RST

T	R	S	Q_{t+}	Fonctionnement
0	0	0	Q_t	Mémoire
0	0	1	Q_t	
0	1	0	Q_t	
0	1	1	Q_t	
1	0	0	Q_t	Mémoire
1	0	1	1	Mise à 1
1	1	0	0	Mise à 0
1	1	1	x	Interdit

Tab 3.3 : Table de fonctionnement de la bascule RST

Exemple

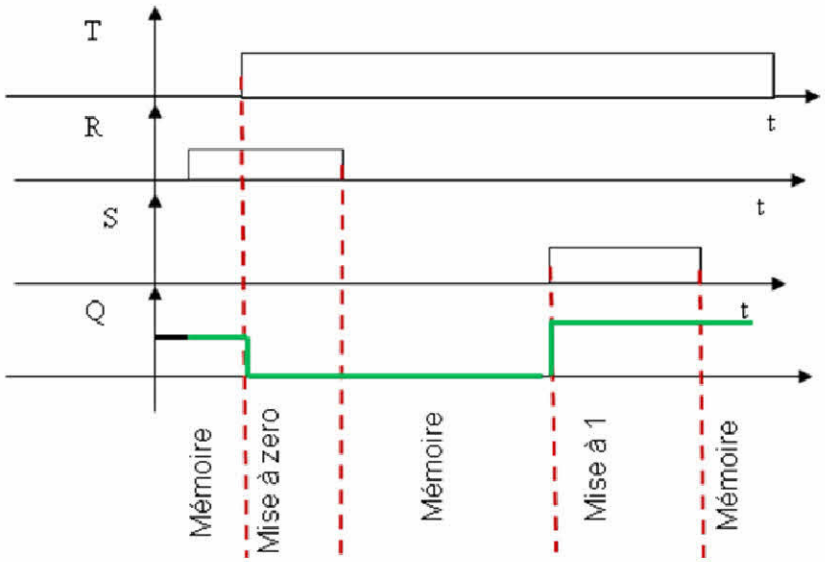


Fig 3.5 : Exemple de chronogramme de la bascule RST

IV.2 La bascule D

Cette bascule dispose d'une seule entrée appelée l'entrée D. Le signal de synchronisation peut être actif soit sur un niveau (D Latch), soit sur un front (edge triggered). Avec une seule entrée on ne peut trouver que deux modes de fonctionnement :

- Si le signal de synchronisation est actif, la sortie recopie l'entrée D.
- Si le signal de synchronisation n'est pas actif, la sortie ne change pas. C'est le mode de fonctionnement en mémoire.

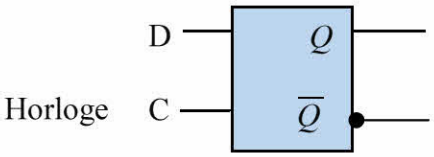


Fig 3.6 : Bascule D Synchronisée Sur niveau (D Latch)

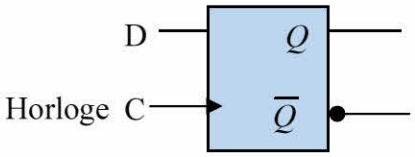


Fig 3.7 : Bascule D Synchronisée Sur Front (D triggered)

D	CK	Q_t	Q_{t+}	fonctionnement
0	0	0	0	Mémorise
0	0	1	1	Q_t
0	1	0	0	Recopie D
0	1	1	0	
1	0	0	0	Mémorise
1	0	1	1	Q_t
1	1	0	1	Recopie D
1	1	1	1	

Tab 3.4 : Table de fonctionnement de la bascule D

Equation de sortie de la bascule D

La bascule D est réalisée à partir d'une bascule RST où les entrées R et S sont liées par la relation : $D = S = \overline{R}$.

- Pendant la phase où l'horloge est inactive on a $Q_{t+} = Q_t$ (Fonction mémoire)
- Pendant la phase où l'horloge est active on a : $Q_{t+} = D_t$ (Fonction de recopie)

Remarque

Le passage en position mémoire veut dire qu'elle (La bascule D) mémorise la dernière valeur recopiée.

Exemple 1 Bascule D avec signal de synchronisation actif sur un niveau haut.

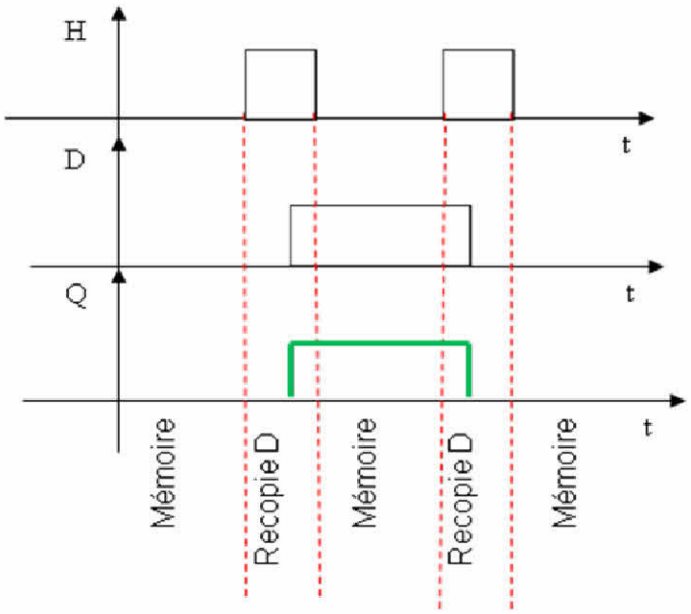


Fig 3.8 : Exemple de chronogramme de la bascule D synchronisée sur niveau haut

Exemple 2 Bascule D avec signal de synchronisation actif sur un front montant.

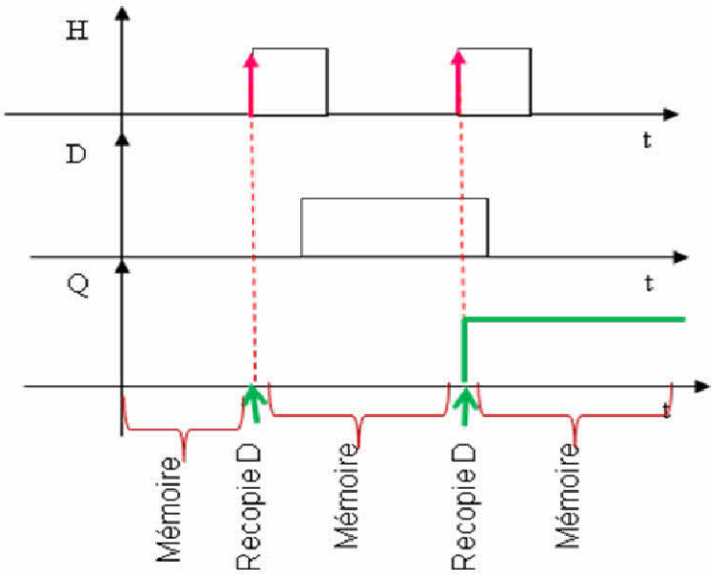


Fig 3.9 : Exemple de chronogramme de la bascule D synchronisée sur front positif

IV.3 Bascule JK

C'est une bascule disposant de deux entrées respectivement appelées J et K, comme pour les bascules RS, l'entrée :

- J pour la mise à 1.
- K pour la mise à 0.

La différence entre la bascule JK et RS réside dans le fait que la JK ne possède pas d'état interdit.

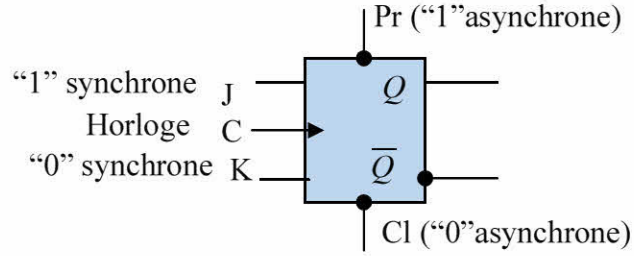


Fig 3.10 : Bascule JK

La table de fonctionnement est la suivante :

CK	J	K	Q_{t+}	Fonctionnement
↑	0	0	Q_t	Mémoire
↑	0	1	0	Mise à 0
↑	1	0	1	Mise à 1
↑	1	1	$\overline{Q_t}$	Basculement

Tab 3.5 : Table de fonctionnement de la Bascule JK

Equation de sortie de la bascule JK

$$Q_{t+}=J\overline{Q}+\overline{K}Q$$

Exemple : Bascule JK avec signal de synchronisation actif sur front descendent

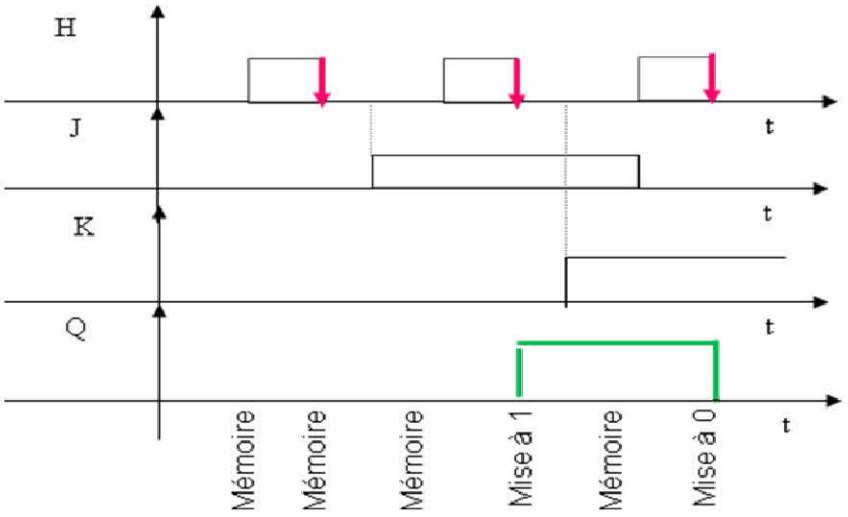


Fig 3.11 : Exemple de chronogramme de la Bascule JK

IV.4 Bascule T

Une bascule fonctionnant suivant le type T dispose d'une commande T, elle change d'état à chaque impulsion de commande.

Equation de fonctionnement

$$Q_{t+} = \overline{Q}_t$$

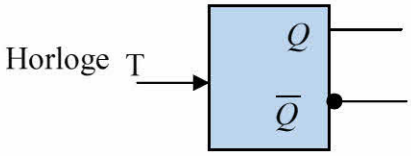


Fig 3.12 : Bascule T

Exemple : Bascule T avec signal de synchronisation actif sur front Montant.

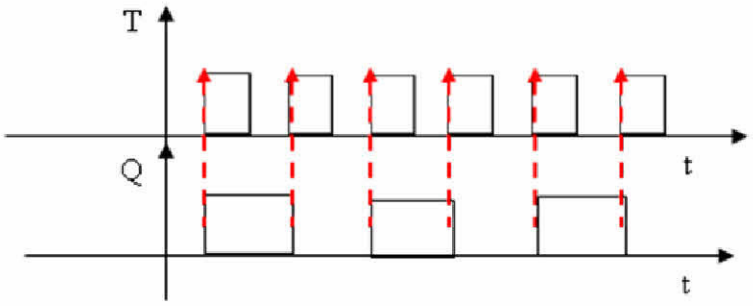


Fig 3.13 : Exemple de chronogramme de la Bascule T synchronisée sur front montant

Remarque

- Si le signal d'entrée est de période $T_H(f_H)$ celui de sortie est de $2T_H$
- Ce circuit est un diviseur de fréquence par 2.

V. Les registres

Un registre est d'abord un ensemble de cases ou cellules mémoires capables de stocker une information. Dans le système binaire, une case mémoire est définie à l'aide d'une bascule. Un registre est donc un ensemble ordonné de bascules (Fig.3.14).

V.1 Fonctionnement

Un registre sert à mémoriser un mot ou un nombre binaire. Le schéma d'un tel système comporte autant de bascules type D que d'éléments binaires à mémoriser. Toutes les bascules sont commandées par le même signal d'horloge (Fig.3.15).

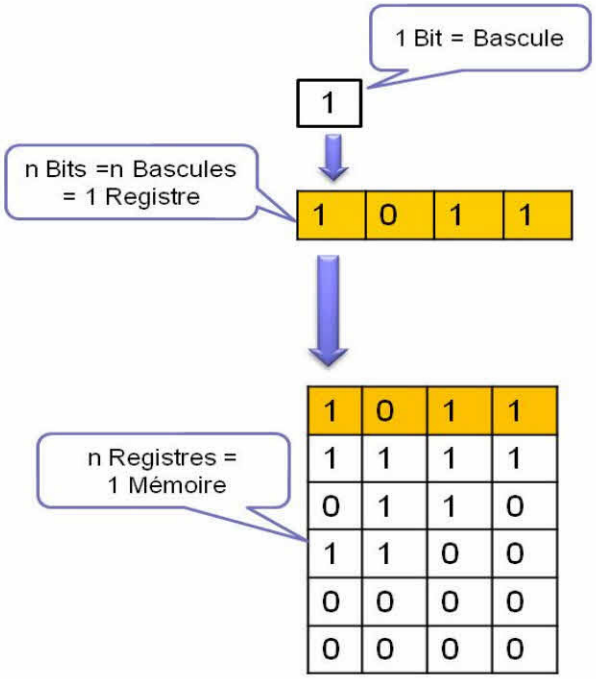


Fig 3.14 : Illustration de la relation bit, registre et mémoire

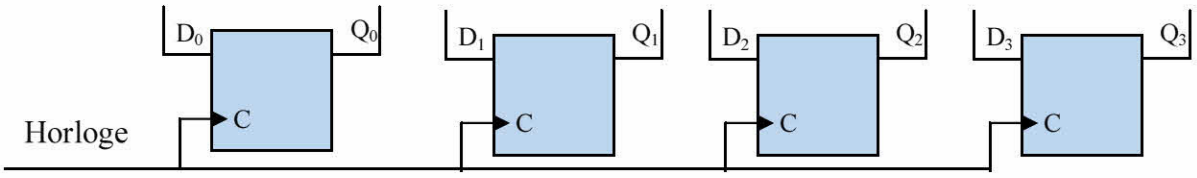


Fig 3.15 : Exemple de registre à 4 bits à base de bascules D

Les principales fonctions d'un registre sont :

- La mémorisation
- Le décalage à droite
- Le décalage gauche

a. Mémorisation : mémoriser l'information telle qu'elle reste sans aucun changement.

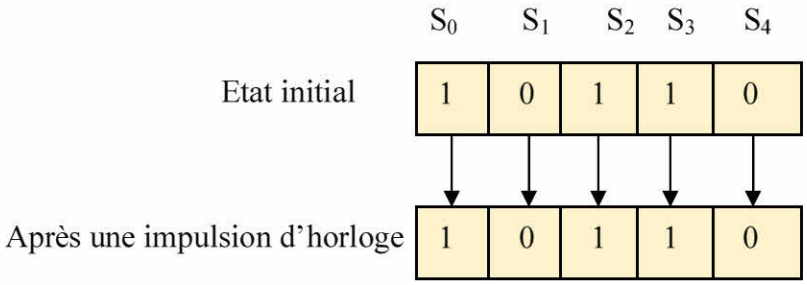


Fig 3.16 : Illustration de la mémorisation

b. Décalage à droite : décaler une information de la gauche vers la droite : Le contenu de la bascule de rang i est transmis à celle de rang i + 1 à chaque impulsion d'horloge.

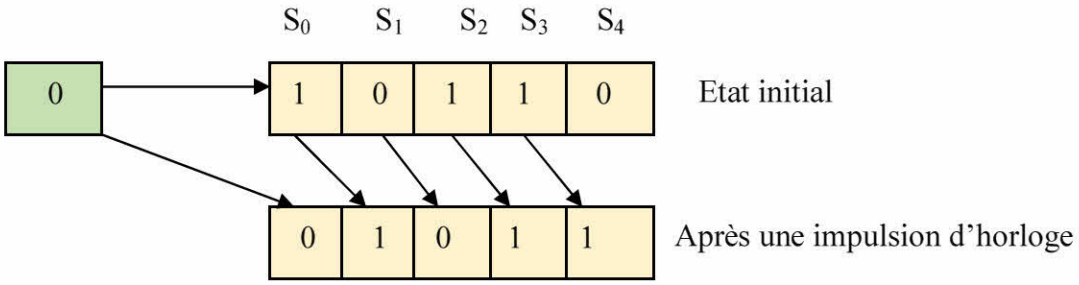


Fig 3.17 : Illustration du décalage à droite

c. Décalage à gauche : décaler une information de droite vers la gauche : la bascule de rang i prend la valeur de sortie de la bascule de rang i + 1 à chaque impulsion horloge.

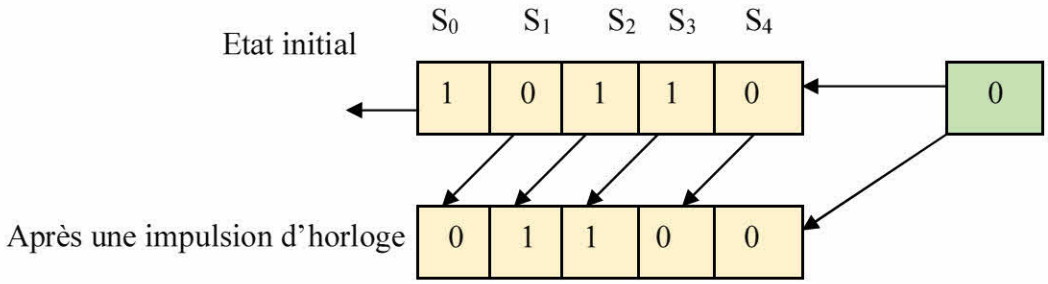


Fig 3.18 : Illustration du Décalage à gauche

Remarque

Parmi les opérations effectuées par les registres :

- La conversion série-parallèle d'une information numérique.
- Les opérations de multiplication et de division par 2.

V.2 Registre de Mémorisation

Le registre de mémoire peut se réaliser sous la forme représentée par la figure 3.19 et qui consiste à interdire l'action de l'horloge en intercalant une porte ET en série.

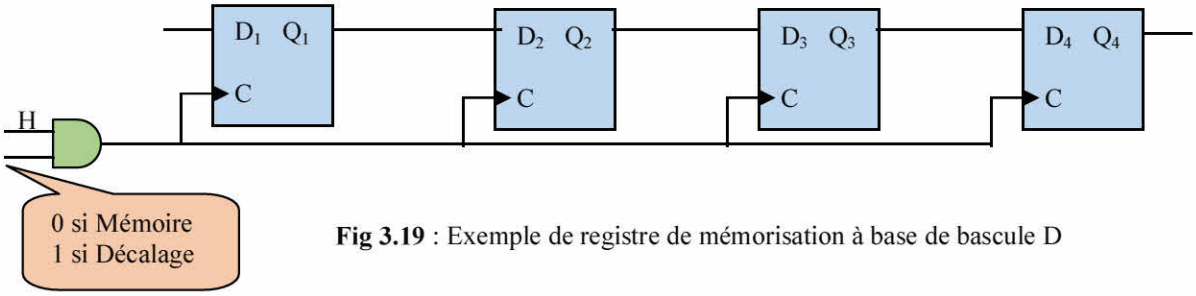


Fig 3.19 : Exemple de registre de mémorisation à base de bascule D

V.3 Registre à décalage à droite

La bascule D de rang i doit recopie la sortie de la bascule de rang i-1. Son entrée D doit être connectée à la sortie i-1, le schéma de l'interconnexion entre les bascules (Fig. 3.20)

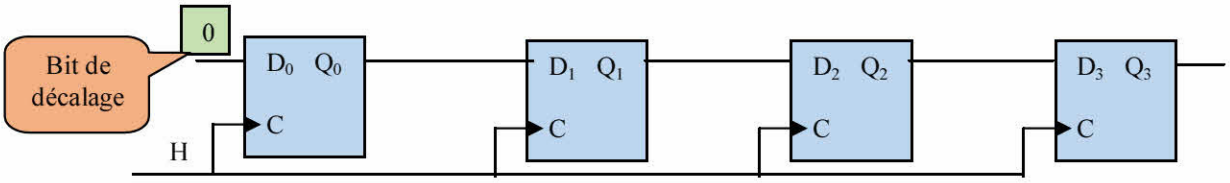


Fig 3.20 : Registre à décalage à droite

V.4 Registre à décalage à gauche

Dans ce type de registre l'entrée D de la bascule de rang i est reliée à la sortie de rang i+1.

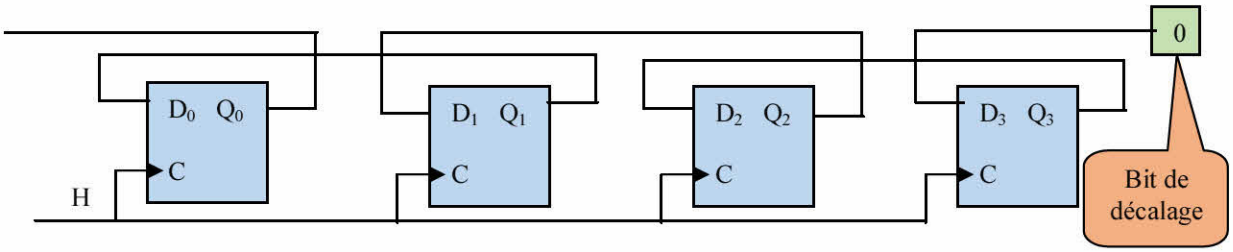


Fig 3.21 : Registre à décalage à gauche

Remarque

Il existe d'autres types de registres tel que :

- Registre à conversion série-parallèle.
- Registre à conversion parallèle -série.
- Registre à délais.

Exemple

Soit le registre à décalage constitué de 3 bascules D suivant :

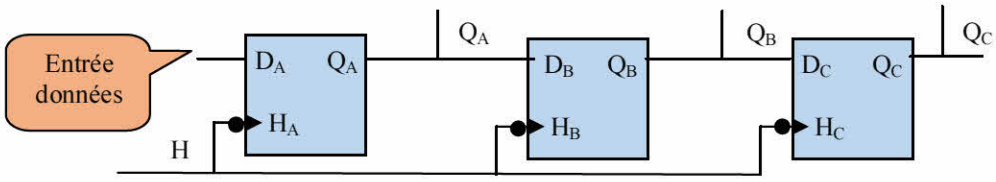


Fig 3.22 : Exemple de registre à décalage à 3 bits

Horloge :N° de l'impulsion	1	2	3	4	5	6	7
Entrée D_A	0	0	1	0	0	0	0
Q_A	0	0	0	1	0	0	0
Q_B	0	0	0	0	1	0	0
Q_C	0	0	0	0	0	1	0

Tab 3.6 : Illustration du décalage du « 1 » dans le registre de décalage à droite

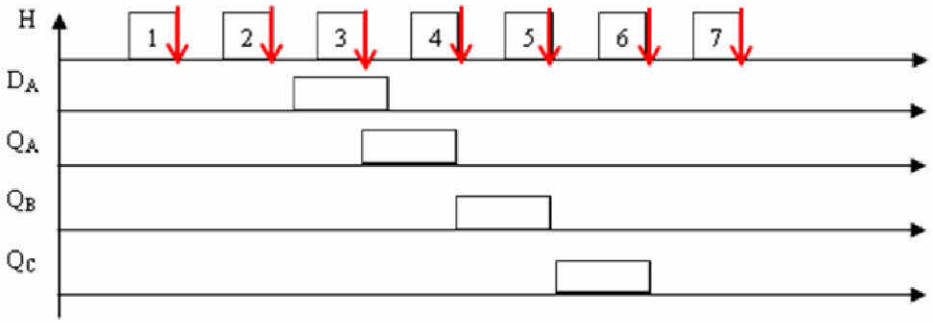


Fig 3.23: Chronogramme de fonctionnement du registre de décalage à droite

VI. Les Mémoires

VI.1 Définition

Nous appelons **Mémoire** tout dispositif capable de stocker des informations (instructions et données) de telle sorte que l'organe qui les utilise puisse à n'importe quel moment accéder à l'information qu'il demande. Les informations peuvent être écrites ou lues. Il y a **écriture** lorsqu'on enregistre des données en mémoire, **lecture** lorsqu'on récupère des informations précédemment enregistrées. La lecture peut être destructive (l'information lue n'est plus en mémoire) ou non.

VI.2 Organisation d'une mémoire

Une mémoire peut être représentée comme une armoire de rangement constituée de différents tiroirs. Chaque tiroir représente alors une case mémoire qui peut contenir un seul élément : des **données**. Le nombre de cases mémoires pouvant être très élevé, il est alors nécessaire de pouvoir les identifier par un numéro. Ce numéro est appelé **adresse**. Chaque donnée devient alors accessible grâce à son adresse

Adresse	Case mémoire
7 = 111	
6 = 110	
5 = 101	
4 = 100	
3 = 011	
2 = 010	
1 = 001	
0 = 000	0001 1010

Fig 3.24 : Organisation d'une mémoire

Avec une adresse de *n* bits il est possible de référencer au plus 2^n cases mémoire. Chaque case est remplie par un mot de données (sa longueur *m* est toujours une puissance de 2). Le nombre de fils d'adresses d'un boîtier mémoire définit donc le nombre de cases mémoire que comprend le boîtier. Le nombre de fils de données définit la taille des données que l'on peut sauvegarder dans chaque case mémoire.

En plus du bus d'adresses et du bus de données, un boîtier mémoire comprend une entrée de commande qui permet de définir le type d'action que l'on effectue avec la mémoire (lecture/écriture) et une entrée de sélection qui permet de sélectionner le boîtier mémoire a utilisé.

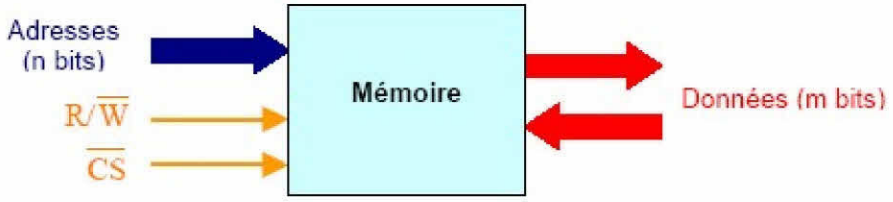


Fig 3.25 : Présentation d'un circuit mémoire

Sur un circuit memoire (figure 2.25) on peut distinguer :

- Les entrées d'adresses
- Les entrées de données
- Les sorties de données
- Les entrées de commandes :

  - une entrée de sélection de lecture ou d'écriture. ($R/\overline{W}$)
  - une entrée de sélection du circuit. ($\overline{CS}$)

Une opération de lecture ou d'écriture de la mémoire suit toujours le même cycle :

1. sélection de l'adresse
2. choix de l'opération à effectuer ($R/\overline{W}$)
3. sélection de la mémoire ($\overline{CS} = 0$)
4. lecture ou écriture de la donnée

Remarque

Les entrées et sorties de données sont très souvent regroupées sur des bornes bidirectionnelles.

VI.3 Caractéristiques d'une mémoire

- **La capacité :** c'est le nombre total de bits que contient la mémoire. Elle s'exprime aussi souvent en octet.
- **Le format des données :** c'est le nombre de bits que l'on peut mémoriser par case mémoire. On dit aussi que c'est la largeur du mot mémorisable.
- **Le temps d'accès :** c'est le temps qui s'écoule entre l'instant où a été lancée une opération de lecture/écriture en mémoire et l'instant où la première information est disponible sur le bus de données.
- **Le temps de cycle :** il représente l'intervalle minimum qui doit séparer deux demandes successives de lecture ou d'écriture.
- **Le débit** : c'est le nombre maximum d'informations lues ou écrites par seconde.
- **Volatilité** : elle caractérise la permanence des informations dans la mémoire. L'information stockée est volatile si elle risque d'être altérée par un défaut d'alimentation électrique et non volatile dans le cas contraire.

VII. Les compteurs

Les compteurs sont des éléments essentiels de la logique séquentielle. L'élément de base des compteurs est, comme pour les registres, la bascule.

L'état du compteur est défini par le nombre binaire formé avec l'ensemble des sorties des bascules.

Les compteurs sont classés en deux catégories suivant le mode de fonctionnement :

- Les compteurs asynchrones ou série.
- Les compteurs synchrones ou parallèles.

VII.1 Compteurs Asynchrones

La caractéristique principale de ces compteurs est la propagation en cascade de l'ordre de changement d'état des bascules. Dans un compteur Asynchrone l'horloge déclenche la bascule B_1 dont la sortie sert de signal d'horloge à la bascule B_2.

Plus généralement le signal d'horloge d'une bascule B_i est issu d'une combinaison logique des sorties des bascules B_j (avec j inférieur à i).

VII.1.1 Compteur binaire

C'est le plus simple des compteurs asynchrones : les sorties des bascules qui le compose évoluent au rythme de l'horloge de manière à représenter la succession croissante des nombres exprimés en base 2.

Exemple

Soit le compteur binaire a base de bascule JK suivant :

On prend J = K = 1 pour que les bascules fonctionnent tel une bascule T et donc comme un compteur Asynchrone :

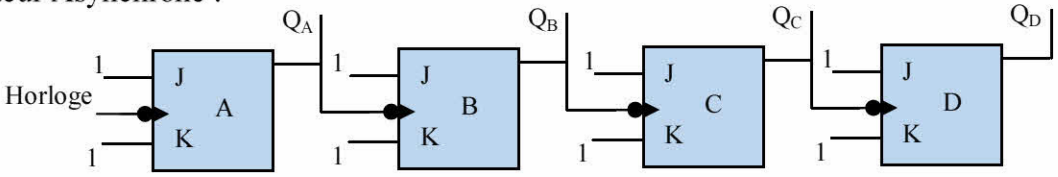


Fig 3.26 : Exemple de compteur binaire asynchrone

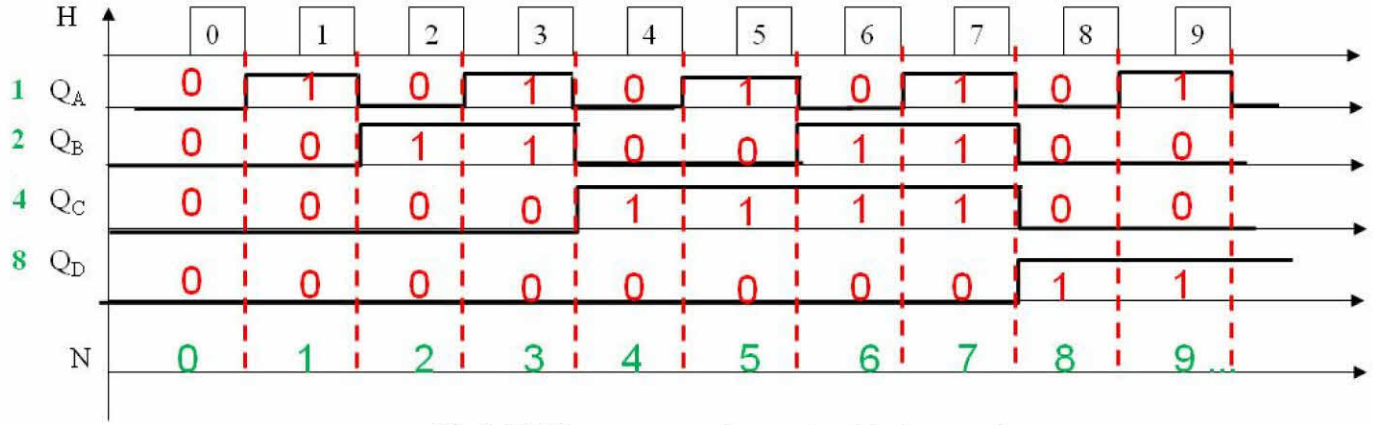


Fig 3.27: Chronogramme du compteur binaire asynchrone

Remarque

Si on affecte les poids 1, 2, 4, 8 respectivement aux sorties Q_A, Q_B, Q_C, Q_D des bascules A, B, C, D donc le nombre binaire ainsi représenté suit l'ordre croissant de 0 à 15 (dans cet exemple nous nous arrêtons à 9).

VII.1.2 Compteur Modulo N

On appel compteur modulo N un compteur décrivant la succession des nombres binaires compris entre 0 et N-1 c.a.d la suite des chiffres d'une base N traduite en binaire.

Par exemple pour un compteur modulo 10 (N = 10) la succession des états du compteur est donnée par le cycle représenté sur la figure 3.28.

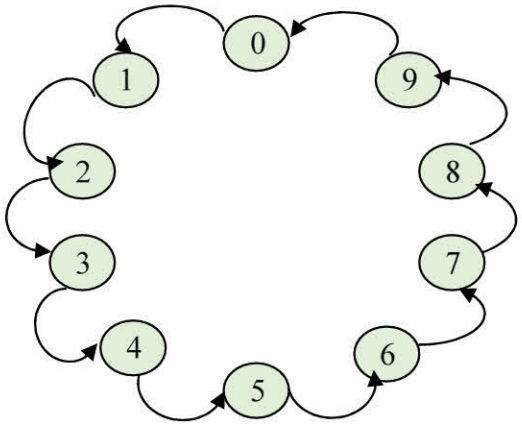


Fig 3.28 : Cycle de comptage d'un compteur binaire modulo 10

Remarque

Un compteur modulo N compte N impulsion d'horloge.

Conception d'un compteur modulo N

Un compteur modulo N est considéré comme un compteur binaire dont le cycle est interrompu.

Exemple

On veut concevoir un compteur modulo 10 c.a.d compte 10 impulsions et fait un rebouclage c.a.d de 0 à 9 et recommence le comptage en arrivant à 9.

Si on laisse le compteur sans contrôle en arrivant à la combinaison binaire équivalente à 9 dans la prochaine impulsion il va inscrire à sa sortie 10 ce qui est indésirable dans notre cas dans ce cas nous devons faire une remise à zéro de toute les bascules du compteur à chaque fois que l'état 10 $(1010)_2$ est détecté pour satisfaire le cycle voulu, ce qui donne le schéma de la figure 3.29.

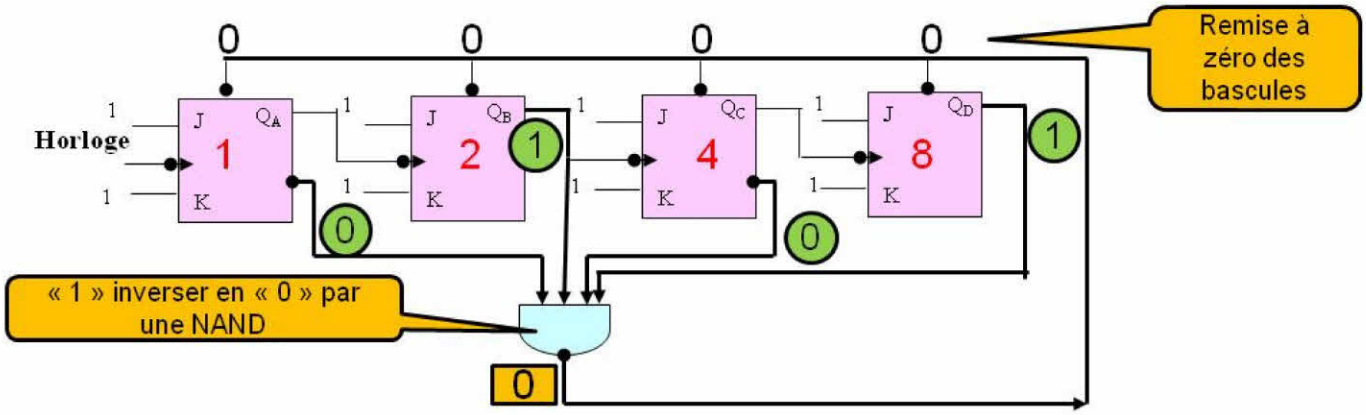


Fig 3.29 : Compteur asynchrone binaire modulo 10

Remarque

1- Le compteur fait un comptage jusqu'à ce que Q_D et Q_B soient à 1, les combinaisons 11 à 15 ne doivent pas apparaître en fonctionnement normal.

2- Il ne faut pas limiter la remise à zéro aux deux seules bascules qui sont à 1 car si seules les bascule B et D sont remisent à 0, le changement d'état de B provoque le changement d'état de C.

VII.2. Compteur Synchrone

Dans les compteurs synchrones le signal d'horloge synchronise toutes les bascules simultanément, ici toutes les bascules doivent être synchronisées par le même signal d'horloge.

Exemple

Supposons qu'on ait à réaliser un compteur synchrone répondant à la table de fonctionnement (Tab.3.7)

Donc selon cette table on décrit :

- ✓ La bascule B change d'état quand $Q_A = 1$.
- ✓ La bascule C change d'état quand $Q_A = Q_B = 1$.
- ✓ La bascule D change d'état quand $Q_A = Q_B = Q_C = 1$.

Etat	Q_D	Q_C	Q_B	Q_A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1
0	0	0	0	0

Tab 3.7 : Table de fonctionnement du compteur synchrone à 4 bits

Supposons qu'on dispose de bascule JK (très utilisées en pratique) :

- La commande de la bascule A doit être effectuée en reliant J_A et K_A à 1.
- La commande de la bascule B doit être effectuée en reliant J_B et K_B à Q_A.
- La commande de la bascule C doit être effectuée en reliant J_C et K_C à Q_A et Q_B par une porte AND c.a.d $J_C = K_C = Q_AQ_B$.
- La commande de la bascule D doit être effectuée par : $J_D = K_D = Q_AQ_BQ_C$.

On aura donc la réalisation du compteur synchrone illustré par la figure 3.30.

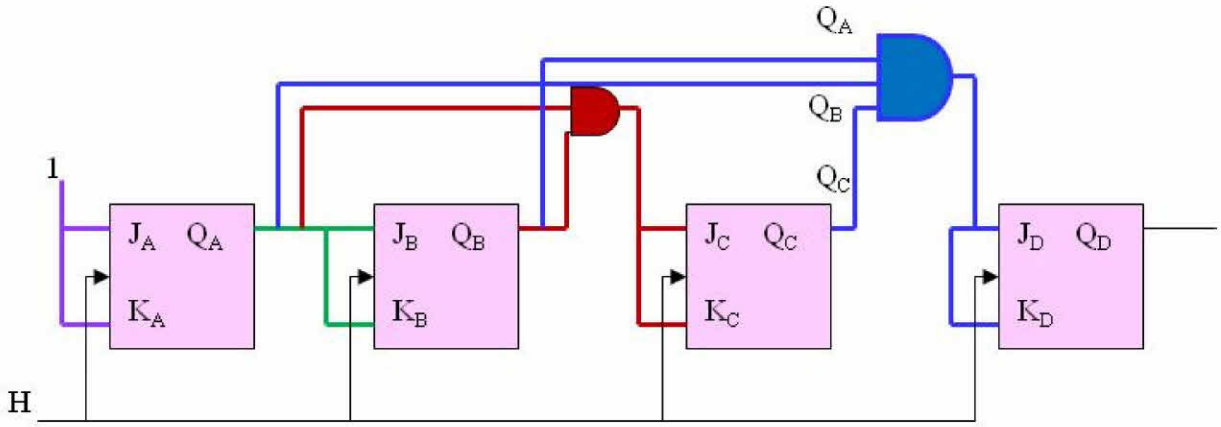


Fig 3.30 : Compteur synchrone à 4 bits

VIII. Conclusion

Les circuits séquentiels sont des éléments cruciaux dans le fonctionnement des systèmes numérique.

La bascule est le cœur de ces circuits grâce elle on peut concevoir des registres, des mémoires, des compteurs…etc.

Exercices

Partie 1 : Les Bascules

Exercice 3.1

1- Completer le chronogramme de la bascule RS .

2- Completer le chronogramme la bascule R S T synchronisée sur niveau haut de H.

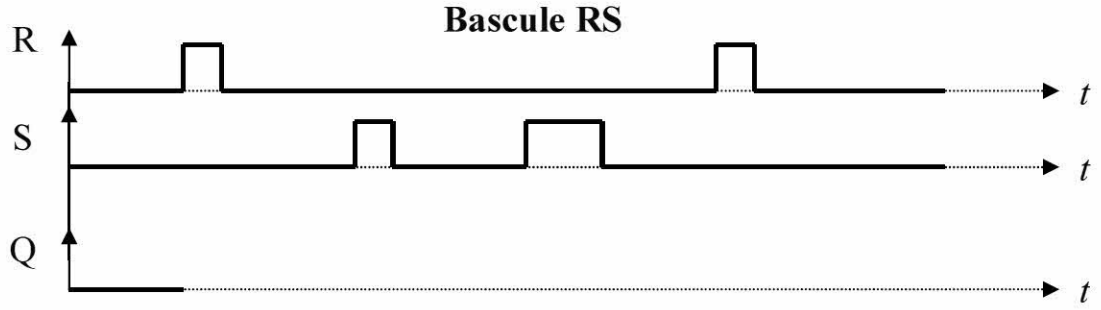


Figure 1

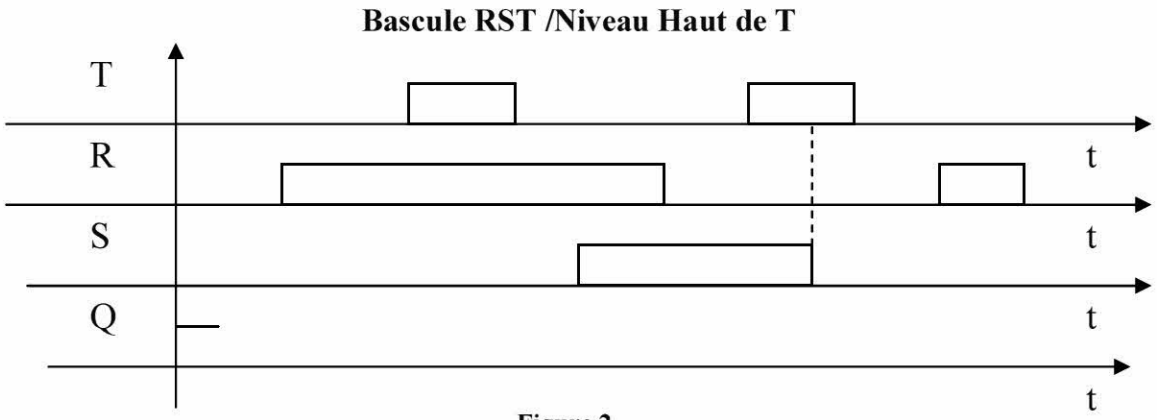


Figure 2

Exercice 3.2

Compléter les chronogrammes déterminant le fonctionnement de la bascule D pour chacun des cas ou la bascule répond aux synchronisations du signal d'horloge désignées.

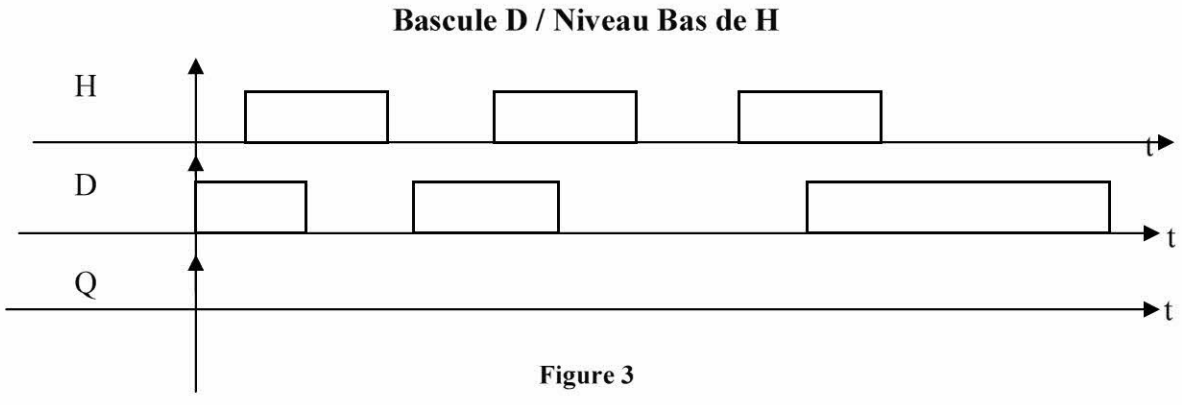


Figure 3

Bascule D / Front Montant de H

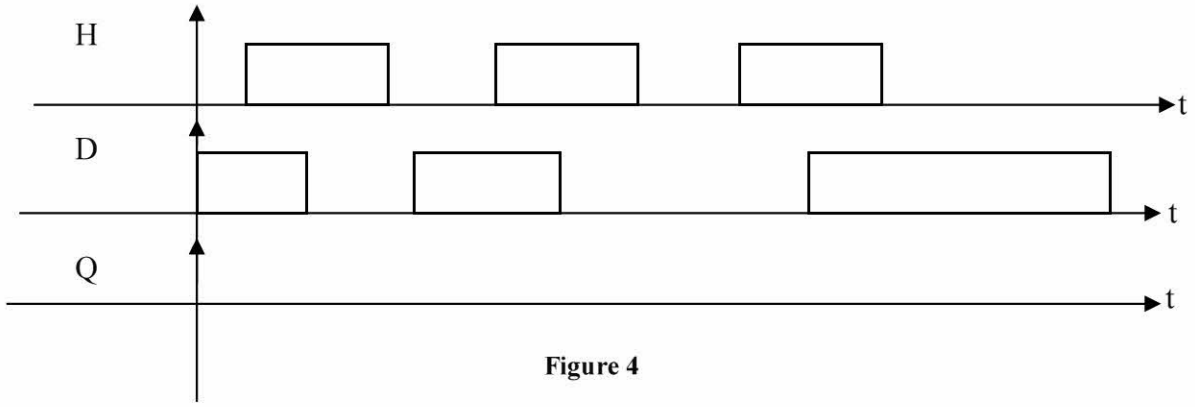


Figure 4

Exercice 3.3

Compléter le chronogramme associé à la bascule JK synchronisé comme suit :

Sur niveau bas de H

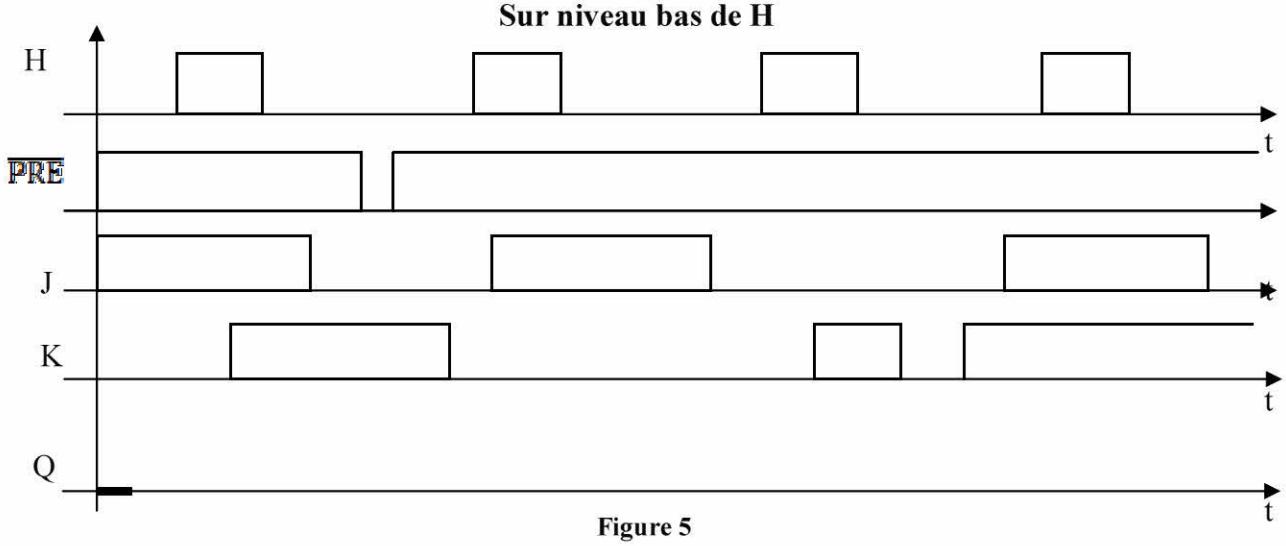


Figure 5

Sur front Descendant de H

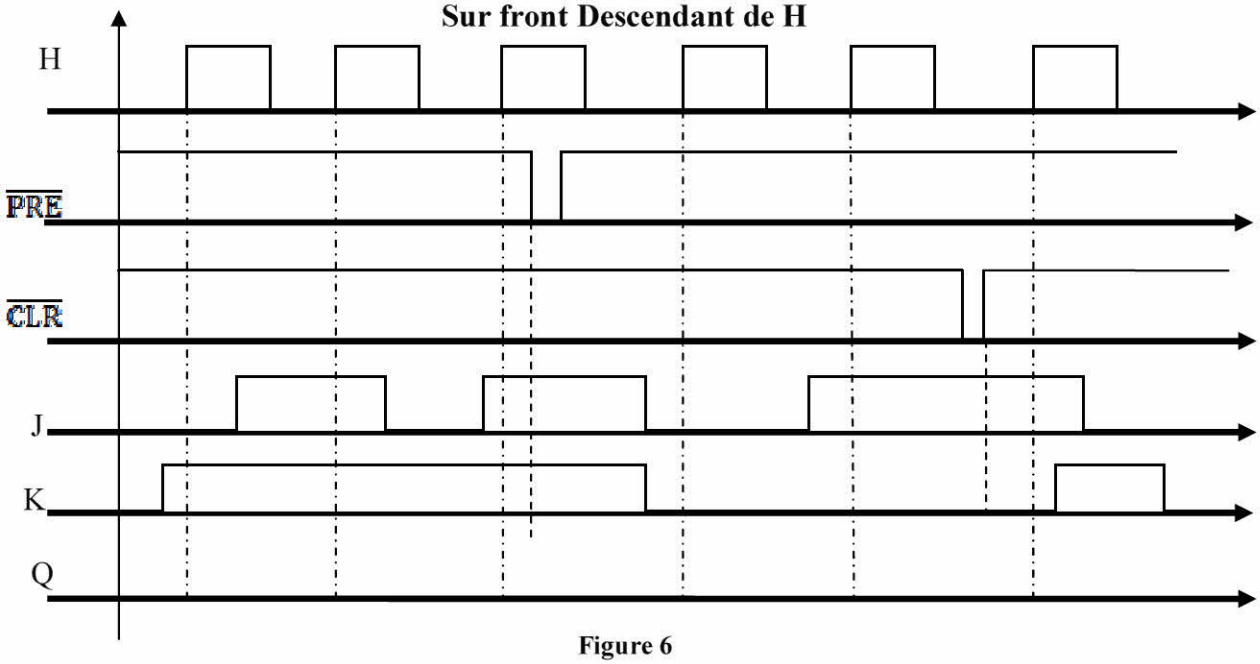


Figure 6

Partie 2 Les registres

Exercice 3.4

Soit le circuit de la figure 7 :

a- Compléter le chronogramme associe à ce circuit

b- En déduire le rôle de ce circuit.

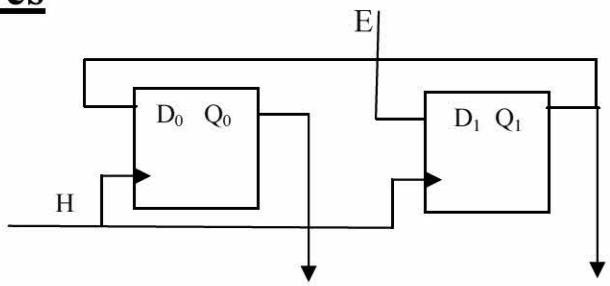


Figure 7

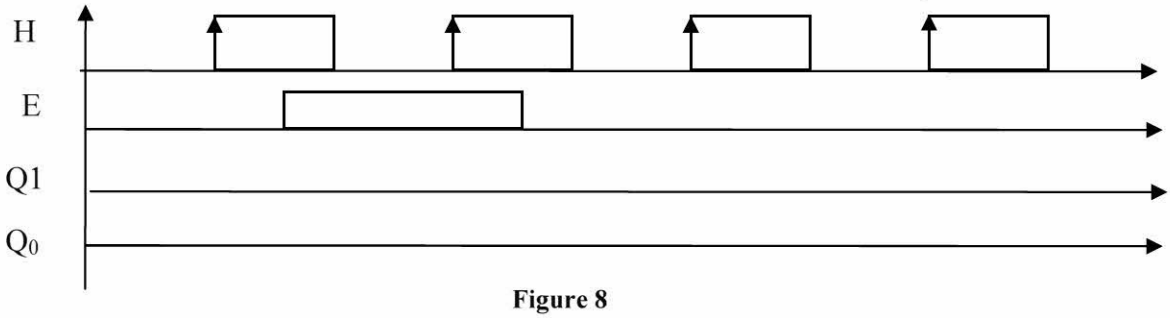


Figure 8

Exercice 3.5

Soit le circuit séquentiel de la figure 9 :

Pour chacun des cas E = 0 et E = 1

1. Compléter le chronogramme correspondant.
2. En déduire le rôle du circuit de la figure 9 (pour chaque cas de E).

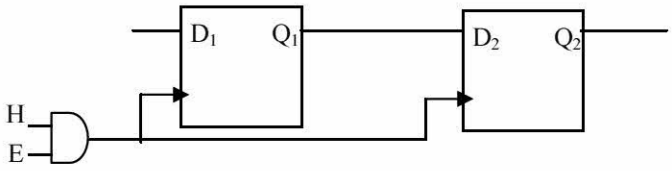


Figure 9

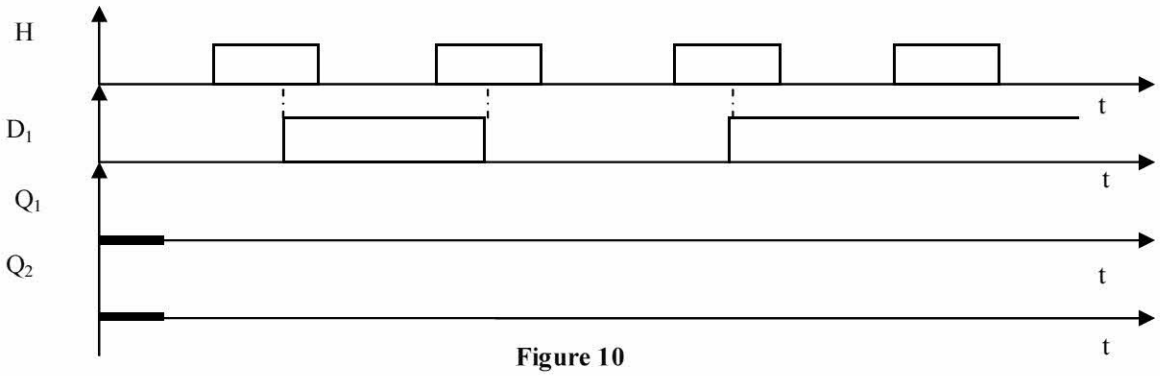


Figure 10

Partie 3 : Les Compteurs

Exercice 3.6

1. Soit le montage de la figure 11.
   a. Compléter le chronogramme associé à ce circuit.
   b. En déduire son rôle.

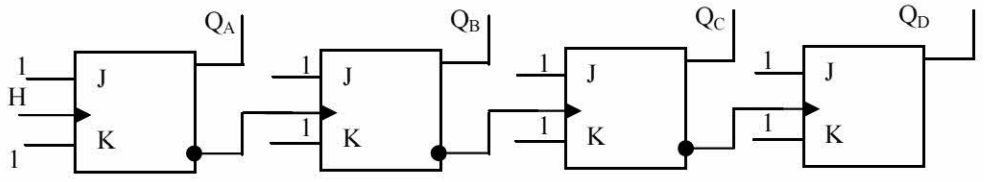


Figure 11

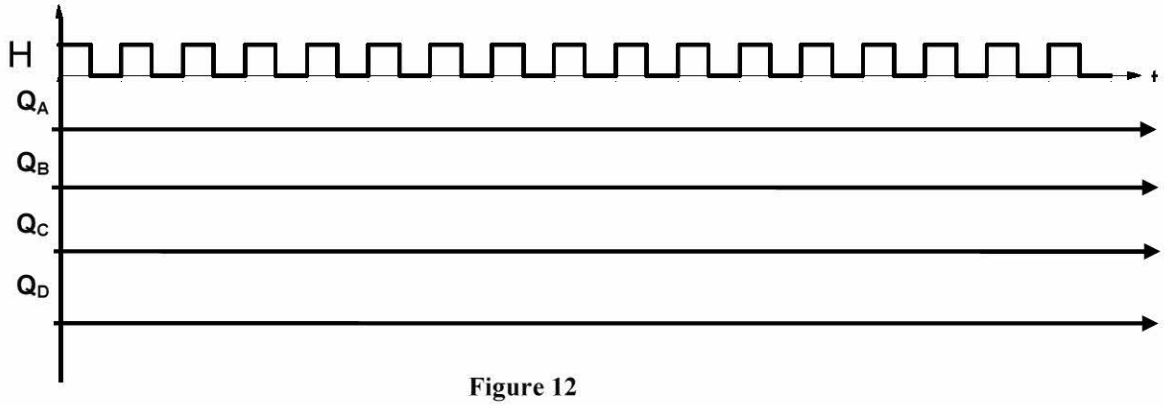


Figure 12

2. On modifie légèrement le montage de la figure 11 afin d'obtenir le montage de la figure 13. Que réalise le circuit de la figure 13 (justifier votre réponse).

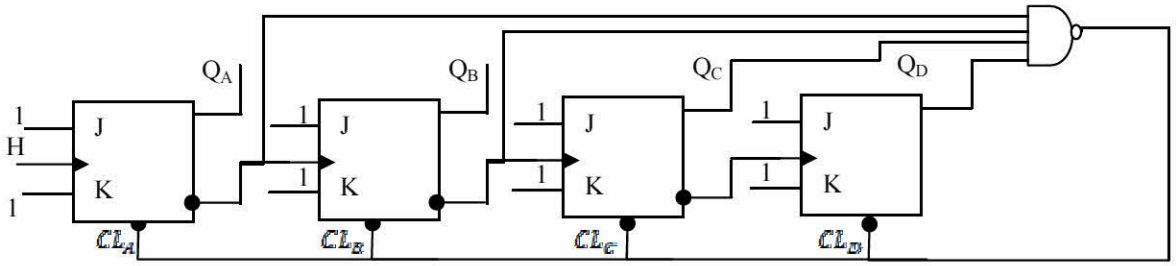


Figure 13

3. Soit le montage representé par le circuit de la figure 14 :
   a. Compléter le chronogramme (fig. 15) associé à ce circuit.
   b. En déduire que fait ce circuit.

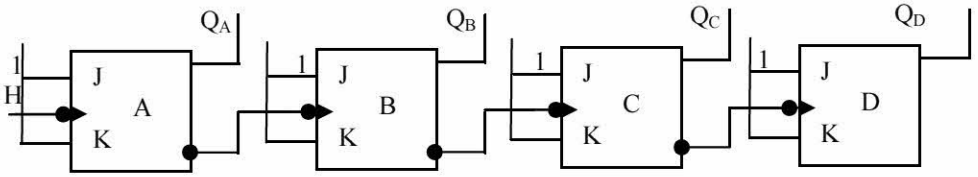


Figure 14

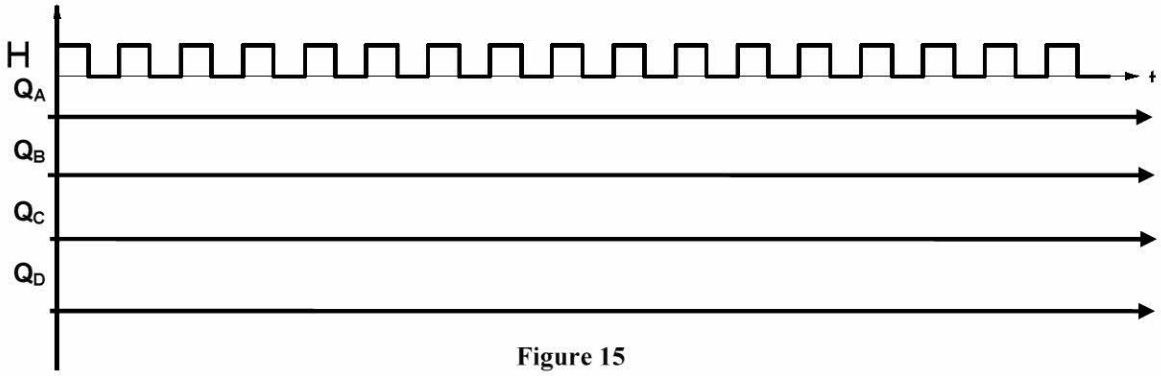


Figure 15

4. Câbler le montage de la figure 16 afin d'obtenir un compteur asynchrone modulo 14.

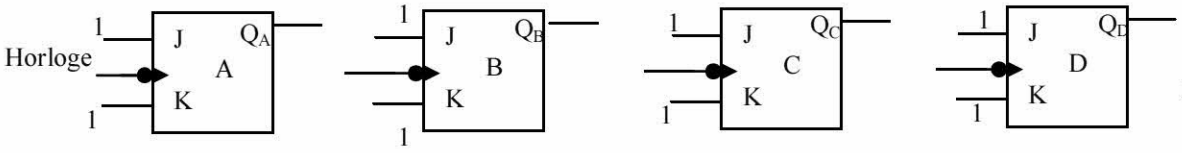


Figure 16

Exercice 3.7

Soit le circuit de la figure 17.

1. Complèter le chronogramme de la figure 18 associé à ce circuit.

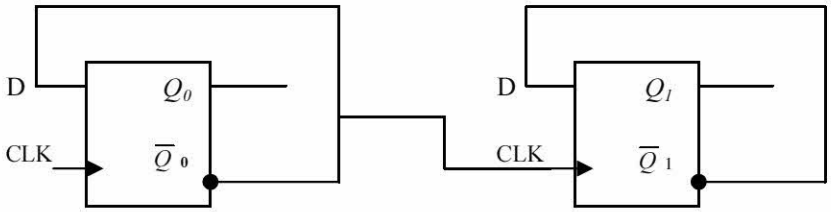


Figure 17

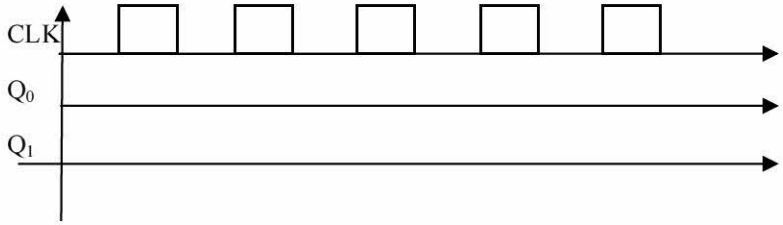


Figure 18

2. Quelle est la séquence des signaux de sortie obtenue ? et quelle est la fonction réalisée ?
3. Etudier et donner le schéma d'un compteur asynchrone modulo 8 avec des bascules D.
4. Soit le nouveau circuit de la figure 19, compléter le chronogramme associé a ce circuit (figure 20)

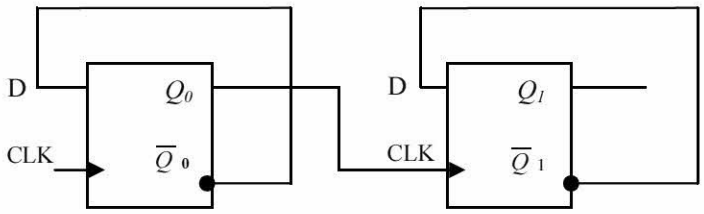


Figure 19

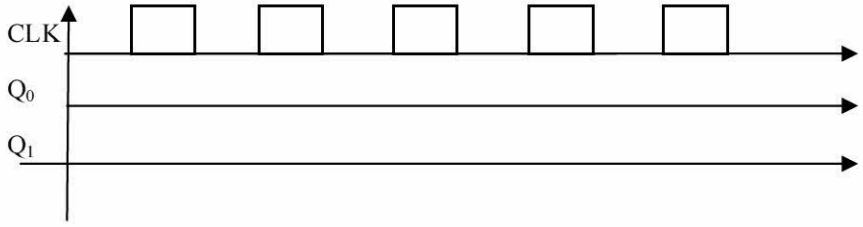


Figure 20

5. Quelle est la séquence des signaux de sortie obtenue ? et quelle est la fonction réalisée ?
6. Etudier et donner le schéma d'un décompteur asynchrone modulo 8 avec des bascules D.

Chapitre 4
Les Automates Finis

Chapitre 4

Automates finis (Machine à état finis)

I. Introduction

Nous avons vu précédemment que les circuits séquentiels possèdent des rétroactions : Les signaux de sortie ne dépendent pas uniquement des entrées mais aussi de leurs séquences. Le circuit se rappelle des entrées et des états précèdent : il a une mémoire du passé.

Si l'étude des circuits combinatoires repose sur l'algèbre de Boole, celle des circuits séquentiels repose sur la théorie des automates finis.

L'objectif principale de ce chapitre consiste à apprendre analyser et synthétiser un automate finis en utilisant les circuits séquentiels étudiés dans le chapitre 3.

Les références utilisées pour l'élaboration de ce chapitre sont principalement : [8] [9] [10] et [11].

II. Où trouve t-on les automates ?

La réponse est simple : partout. La commande d'un ascenseur ou d'un feu tricolore à un carrefour routier est un automate, un digicode est un automate, le bus PCI de vos PC est dirigé par des automates, TCP que vous utilisez sans le savoir ou en le sachant quand vous surfez sur Internet est un automate… etc.

On peut utiliser un automate dans toutes les applications où la gestion d'un processus, l'utilisation d'une ressource, un traitement sur des données peuvent être décrits par une **chronologie** qui fait intervenir un **nombre fini** d'opérations distinctes conditionnées par des entrées extérieures.

III. Concept d'Automate Fini et Automate Fini Synchrone

Un **Automate Fini (MEF : M**achine à **E**tats **F**inis ou en anglais **FSM : F**init **S**tate **M**achine**)** est une machine possédant un nombre **fini d'éléments et de mémoires,** soit 2^n Etats appelés Etats internes, où n est le nombre de bits de mémoire.

Une machine à état finis est définie comme un modèle mathématique permettant de décrire le fonctionnement des Machines (Systèmes) Automatiques, sur différents Etats et le passage d'un Etat à l'autre, le plus souvent commandé par circuit séquentiel.

Un **Automate** est dit **Synchrone** lorsque le passage d'un état (état présent) à l'état suivant (état futur) a lieu sur une transition d'un signal appelé horloge commun à toutes les bascules de l'automate.

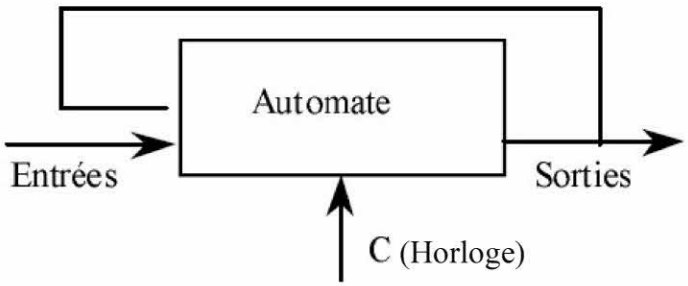


Fig 4.1 : Automate synchrone

Formellement, un automate fini est caractérisé par :

- ✓ Un vecteur des entrées (E),
- ✓ Un vecteur de sorties (S),
- ✓ Une séquence d'états définissant son comportement (Q).
- ✓ Un état de départ lui permettant de débuter son fonctionnement à partir d'un point fixe.

Le bon fonctionnement d'un automate est conditionné par respect des points suivants :

- ➢ L'Automate est un graphe dont les nœuds sont appelés États (State) et les liens appelés Transitions.
- ➢ L'État ne représente pas un composant du système mais le système tout entier dans un État donné **ex** : Si une voiture est un système, sa roue n'est pas un État mais la voiture tout entière, si le moteur est arrêté la voiture est dans l'État Arrêtée.
- ➢ La Transition représente une **condition** ou un **événement** qui fait passer le système d'un État à un autre ex : Tourner la clé de la voiture est l'événement qui fait passer la voiture de l'État Arrêtée vers l'État Démarrée.
- ➢ L'Automate fonctionne de tel-sort que le système reçoit différents **événements**, ces événements déclenchent les Transitions adéquates et le système passe d'un État à l'autre pour chaque événement sans arrêter.
- ➢ Le système ne peut pas être dans 2 États ou plus à un instant donné, Quelques soit le temps de marche, un seul État est actif à la fois sur tout l'Automate.

IV. Description du comportement d'un automate

Le comportement d'un automate est déterminé si l'on connaît soit ses fonctions de transfert, soit ses tables de transitions ou son diagramme d'états:

a. **Ses fonctions de transfert** :
   - ✓ Sa réponse S à l'instant t+1 en fonction de l'entrée E a l'instant t et son état Q a l'instant t : **S(t+1) = f[E(t), Q(t)]**
   - ✓ Son état Q à l'instant t+1 en fonction de son état Q a l'instant t et de son entrée E a l'instant t : **Q(t+1) = g[Q(t), E(t)]**.

b. **Les tables de transitions ou d'états** qui donnent les valeurs des fonctions f et g. Elles doivent contenir l'état Q (ou les états si l'on a plusieurs bascules) à l'instant t, l'entrée E (ou les entrées), Q(t+1) ou Q^+ : l'état futur (ou les états futurs), l'entrée ou les entrées des bascules, les sorties S_t.

c. **Les diagrammes d'états ou de transitions**, où les états sont représentés par des ronds et les transitions entre états par des flèches allant de l'état initial a l'état final.

V. Représentation des automates finis (Machine a état finis)

Les machines à états finis sont généralement représentées par des diagrammes d'états permettant de visualiser les transitions entre les états. Les états sont alors représentés par des cercles avec leur nom inscrit à l'intérieur ; les transitions entre les états sont représentées par des arcs dirigés reliant les cercles ; les conditions (valeurs du vecteur d'entrée) enclenchant ces transitions sont notées sur les arcs ; et finalement la valeur des sorties est généralement indiquée soit sur l'arc (séparée des entrées par un trait oblique : /) ou à l'intérieur du cercle (séparée du nom de l'état par un trait oblique : /). Voici un exemple de machine à états finis où les sorties sont indiquées à l'intérieur des cercles :

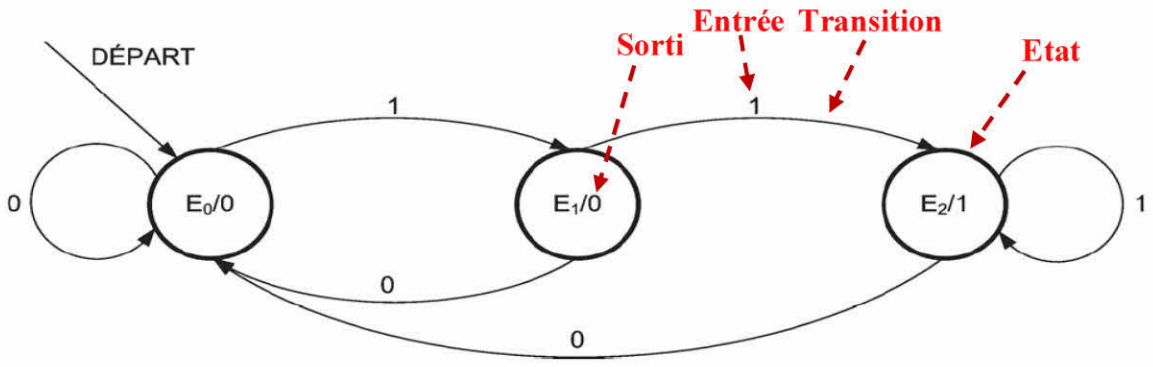


Fig 4.2 : Exemple de graphe d'état

Dans ce graphe (fig. 4.2) on trouve les éléments suivants :

• Trois états notés E_0, E_1 et E_2 ;

• Un état de départ E_0 ;

• Une entrée (dont la valeur est notée sur les arcs) ;

• Une sortie dont la valeur est associée à chaque état, respectivement 0, 0 et 1.

Cette machine permet de détecter une séquence de deux 1 consécutifs en entrée. Considérons en effet son fonctionnement :

- ✓ Au départ, la machine est à E_0 et sa sortie à 0 ;
- ✓ Tant que la machine ne reçoit pas 1, elle demeure à E_0 et sa sortie à 0 ;
- ✓ Si la machine reçoit 1, elle change d'état et part vers E_1 : le début de la séquence 11 est détecté. La sortie demeure à 0 ;
- ✓ Si à cette étape, la machine reçoit en entrée 0, la séquence est brisée. La machine revient à E_0 et tout reprend depuis le début ;
- ✓ Si la machine reçoit 1 alors qu'elle se trouve en E_1, elle transite vers E_2 et sa sortie est alors à 1 (pour indiquer qu'elle a reçu la séquence 11) ;
- ✓ Une fois en E_2, si la machine reçoit 1, elle demeure dans le même état car la séquence (entrée précédente -entrée actuelle) est encore 11 : la sortie est à 1 ;
- ✓ Autrement elle part en E_0 et tout recommence depuis le début.

❖ Dans la pratique, ce type de diagramme est construit par le concepteur pour définir une machine à états finis.

Maintenant considérons le second cas. Voici un exemple de machine à états finis où les sorties sont indiquées sur les arcs (fig. 4.3):

Voici les éléments que l'on y trouve :

- ➢ Deux états notés E_0 et E_1 ;
- ➢ Un état de départ E_0 ;
- ➢ Une entrée (notée sur les arcs) ;
- ➢ Une sortie dont la valeur est associée à chaque transition, 0 pour les transitions de E_0 à E_0 et de E_0 à E_1, 1 pour les transitions de E_1 à E_1 et de E_1 à E_0.

Cette machine (aussi) permet de détecter une séquence de deux 1 consécutifs en entrée. Considérons son fonctionnement :

- ✓ Au départ, la machine est à E_0;
- ✓ Tant que la machine ne reçoit pas 1, elle demeure à E_0 et sa sortie à 0 ;

- ✓ Si la machine reçoit 1, elle change d'état et part vers E_1 : le début de la séquence 11 est détecté. La sortie demeure à 0 mais elle est ici associée à la transition;
- ✓ Si à cette étape, la machine reçoit en entrée 0, la séquence est brisée. La machine revient à E_0 et tout reprend depuis le début ;
- ✓ Si la machine reçoit 1 et qu'elle se trouve en E_1, elle demeure en E_1 et sa sortie est alors à 1 (pour indiquer qu'elle a reçu la séquence 11). Elle à cet état tant et aussi longtemps que l'entrée est 1;
- ✓ Autrement elle part en E_0 et tout recommence depuis le début.

Comme on le voit, dans sa conception, cette machine est différente de la première, mais dans son fonctionnement global, elle lui est exactement identique à un coup d'horloge près.

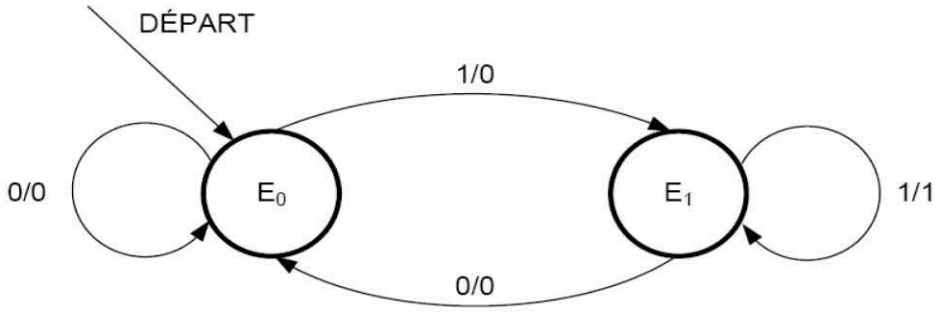


Fig 4.3: Machine à état finis avec sortie sur les arcs

Remarque

Un automate fini forme un graphe orienté étiqueté, dont les états sont les sommets et les transitions les arêtes étiquetées.

Exemple 1 : Un digicode

Un digicode est constitué d'un clavier à 16 touches qui constitue l'organe d'entrée d'un automate dont l'unique sortie commande l'ouverture d'une porte. Nous nous restreindrons à un digicode rudimentaire dans lequel le code secret est fixé une fois pour toutes, sans modification possible.

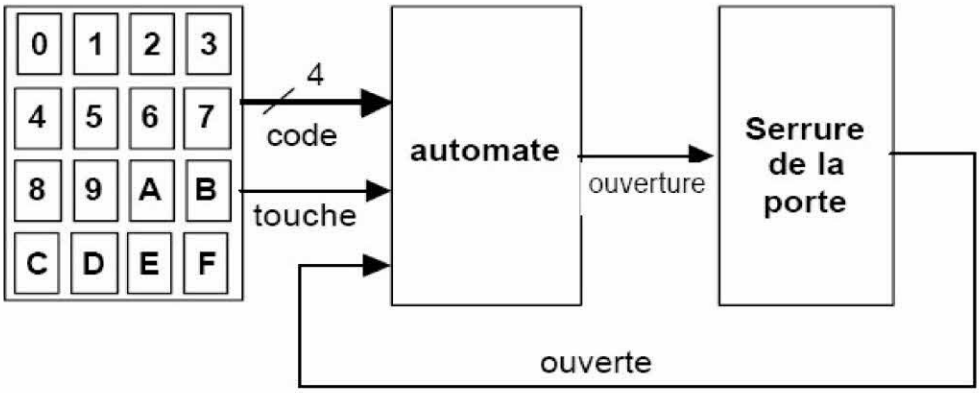


Fig 4.4: Schéma bloc du digicode

Supposons que l'ouverture de la porte soit obtenue en fournissant le code **C94**. Evidemment l'automate doit être informé du fait que l'utilisateur a ouvert la porte, c'est le rôle du signal « ouverte ». Le travail de l'automate peut être résumé par le diagramme suivant:

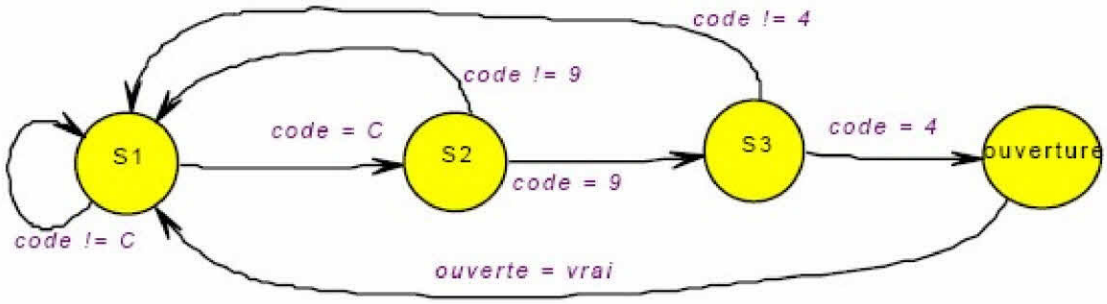


Fig 4.5: Diagramme d'état du digicode

Du moins en première approximation ; il est clair que l'automate doit avoir un moyen de séparer les frappes sur les différentes touches, il doit donc attendre entre deux touches que l'utilisateur relâche le clavier :

NB : En admettant que l'absence de frappe d'une touche provoque T = 0.

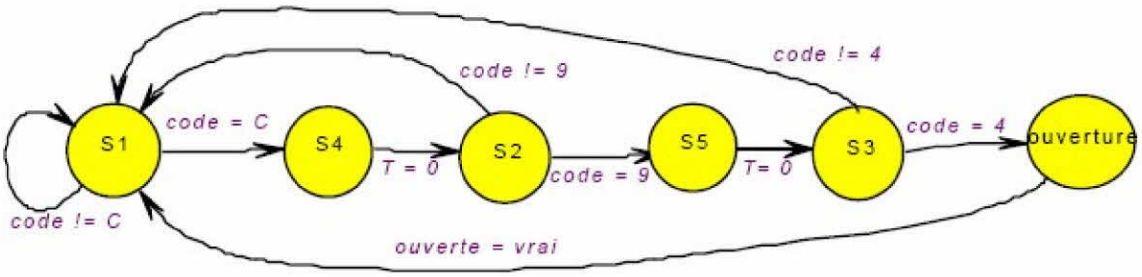


Fig 4.6: Diagramme d'état du digicode avec séparation des frappes

Exemple 2 : Une mémoire Binaire

Mémoire binaire : Entrée : E(0,1)

Sortie : S(0,1)

Etat : Q(0,1)

Dans un tel automate :

- ✓ La sortie à l'instant t+1 ne dépend que de l'état à l'instant t (lecture) S(t+1) = Q(t).
- ✓ L'état à l'instant t+1 ne dépend que de l'entrée à l'instant t (écriture) Q(t+1) = E(t).

Les Tables de transitions de S(t+1) et Q(t+1) sont illustrées respectivement par Tab 4.1 et Tab 4.2

Q(t) \ E(t)	0	1
0	0	0
1	1	1

Tab 4.1 : Table de transition de S(t+1)
Indépendante des entrées

Q(t) \ E(t)	0	1
0	0	1
1	0	1

Tab 4.2 : Table de transition de Q(t+1)
Indépendante des états

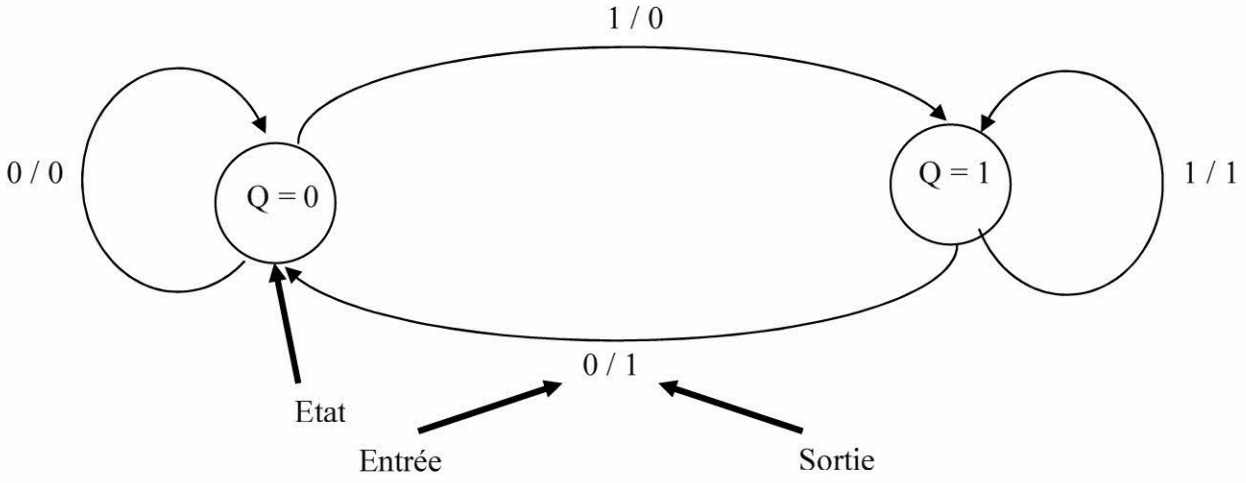


Fig 4.7 : Diagramme d'états de la mémoire

Il existe deux types d'automates finis l'automate de **MOORE** et l'automate de **MEALY**

VI. Machine de Moore et Mealy

Nous avons pu voir que lors de la représentation d'une machine à états finis par un diagramme, il est possible d'indiquer les sorties sur les arcs ou à l'intérieur de l'état. Cette distinction n'est pas purement graphique. Comme on a pu s'en rendre compte en réalisant deux machines selon les deux conventions, les machines sont différentes. Cette différence

réside dans le fait que dans le premier cas, la **sortie** dépend uniquement de **l'état**, tandis que dans le second cas, elle dépend de l'**état** et de l'**entrée**.

VI.1 Automate de MOORE

Lorsque la sortie dépend uniquement de l'état courant, on dit que la machine est une machine de Moore. Le nom de machine de Moore fait référence au professeur Edward F. Moore de l'université de Wisconsin-Madison aux Etats-Unis qui en est l'inventeur (voir fig. 4.8).

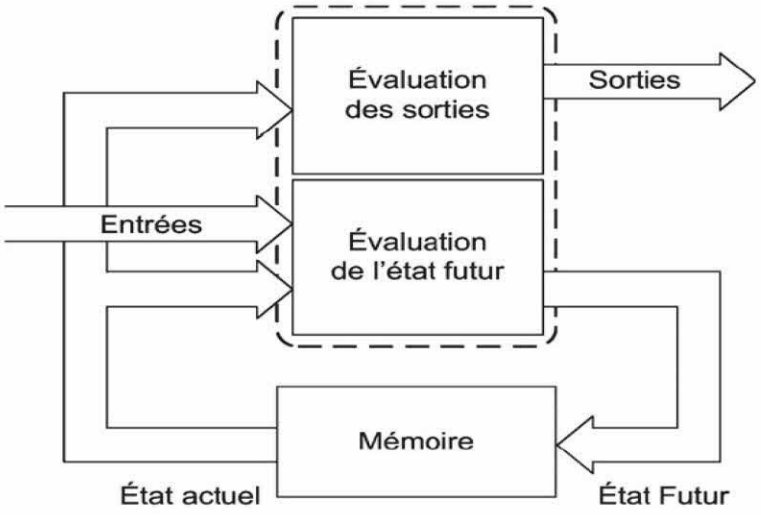


Fig 4.8 : Automate de Moore

VI.2 Automate de MEALY

Lorsque la sortie dépend de l'état courant et de l'entrée, on dit que la machine est une machine de Mealy, en référence à G.H. Mealy (voir fig.4.9).

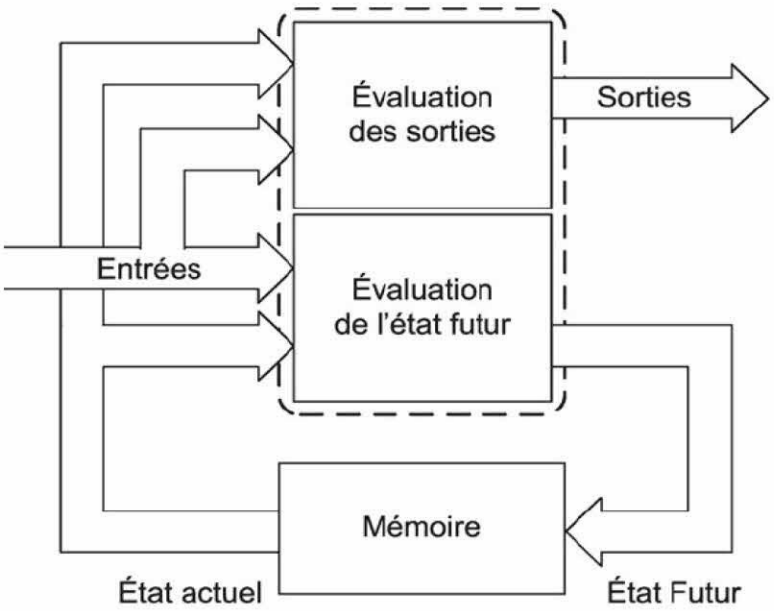


Fig 4.9 : Automate de Mealy

VII. Synthèse d'un circuit séquentiel

Pour faire la synthèse d'un circuit séquentiel il faut :

- ✓ Décrire la fonction du circuit (sa réponse) pour toute entrée valable, a l'aide d'un diagramme de transition
- ✓ Construire la table d'états en indiquant toutes les entrées et toutes les sorties
- ✓ Réaliser les circuits combinatoires associes aux sorties
- ✓ Dessiner le circuit dans sa forme normale (les différentes parties seront représentées de haut en bas : les mémoires (bascules), la logique combinatoire associer au bascules, les entrées et les états présents, la logique combinatoire associée aux sorties et les sorties).

VIII. Analyse d'un circuit séquentiel

Pour analyser un circuit séquentiel il faut suivre les étapes suivantes :

- ✓ Mettre le circuit sous sa forme normale.
- ✓ Déterminer soit sa table d'état en mettant dans chaque case les états futurs et les sorties résultant des états présents et les entrées possibles, soit son diagramme de transition, soit ses fonctions de transfert.

IX. Conclusion

La conception d'un automate à état finis nécessite de faire une analyse détaillée de ce circuit qui permettra de déterminer sa table de transitions, son graphe d'état et sa fonction de transfert.

Exercices

Exercice 4.1

Soit à analyser l'automate de la figure 1 constitué d'une entrée X, 2 bascules de type T (T_0, T_1) et d'une sortie S :

1. Mettre le circuit sous sa forme normale.
2. Remplir la table d'état (Table 1).
3. Déduire les expressions des fonctions de transferts (T_0, T_1, Q_0^+, Q_1^+, S).
4. Est-ce un automate de **Moore** ou de **Mealy** (justifier votre réponse).

Etats Présents		Entrée	Bascules		Etats futurs		Sortie
Q_1	Q_0	X	T_1	T_0	Q_1^+	Q_0^+	S

Table 1

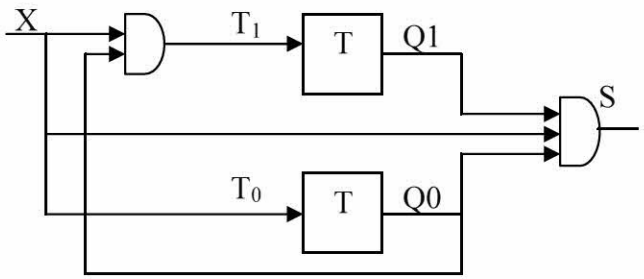


Figure 1

Exercice 4.2

Soit le circuit à analyser de la figure 2 (S = f(X Q_0, Q_1, Q_2)) composer d'une entrée X, de deux bascules T (T_0, T_1) et une sortie S.

1. Mettre le circuit sous sa forme normale.
2. Remplir la table d'états (Table 2) complète.
3. Déduire ses fonctions de transfert (T_0, T_1, Q_0^+, Q_1^+, S)
4. Est-ce un automate de Moore ou de Mealy. (justifier votre réponse).

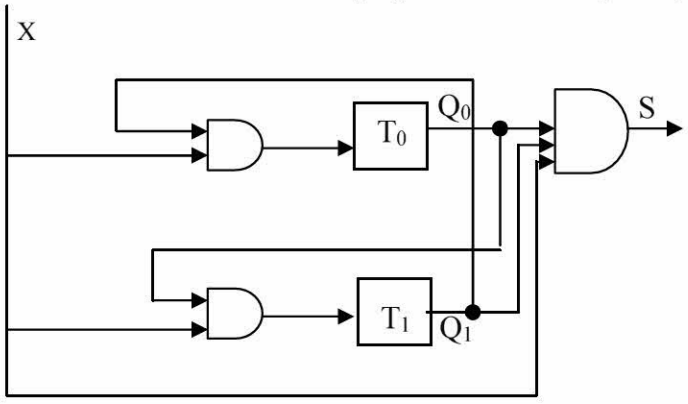


Figure 2

Etats Présents		Entrée	Bascules		Etats futures		Sortie
Q_1	Q_0	X	T_1	T_0	Q_1^+	Q_0^+	S

Table 2

Exercice 4.3

On veut réaliser, à l'aide d'un automate de Moore, un additionneur (figure 3) permettant de calculer la somme de deux opérandes A et B présentés en série. Cet additionneur possède deux entrées notées A et B de 1 bit chacune, correspondant aux deux opérandes. L'additionneur possède également deux sorties sur 1 bit chacune, notées **S** et **R**, correspondant à la somme et à la retenue.

On rappelle que la somme de 3 bits $\mathbf{a_i + b_i + r_{i-1}}$ produit un résultat sur deux bits $\mathbf{r_i\, s_i}$. On a ainsi quatre résultats possibles 00, 01, 10 ou 11.

Le fonctionnement est le suivant : les opérandes sont présentés sur les entrées à raison de 1 bit par période d'horloge **CK**, la somme et la retenue sont disponibles pendant la période qui suit.

L'automate, réalisant L'additionneur série, possède quatre états codés sur deux bits notés $\mathbf{Q_1}$ et $\mathbf{Q_0}$. Le codage et la signification des états sont donnés dans le tableau suivant :

Q_1	Q_0	Etat	Signification r s
0	**0**	**ZERO**	La différence des trois bits vaut 0 0
0	**1**	**UN**	La différence des trois bits vaut 0 1
1	**0**	**DEUX**	La différence des trois bits vaut 1 0
1	**1**	**TROIS**	La différence des trois bits vaut 1 1

Table 3

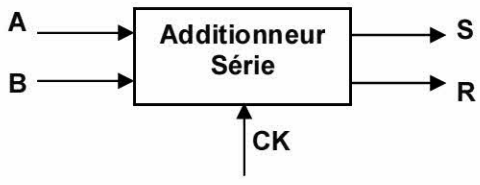


Figure 3

On suppose qu'initialement, l'automate est à l'état **ZERO**.

1. Compléter le graphe de la figure 4, représentant l'automate, en précisant les expressions booléennes attachées à chaque transition. Donner également pour chaque état la valeur des sorties R et S.

2. Compléter la table de transition ci-dessous où: $Q1^+$ et Q_0^+ sont les états futurs et R et S les sorties de l'automate.

3. En utilisant les tables de Karnaugh, donner les expressions booléennes simplifiées des signaux Q_1^+, Q_0^+, R et S.

4. Est-ce qu'il y a des transitions manquantes ? Si oui désigner les sur le graphe d'état et donner les conditions de transition associées.

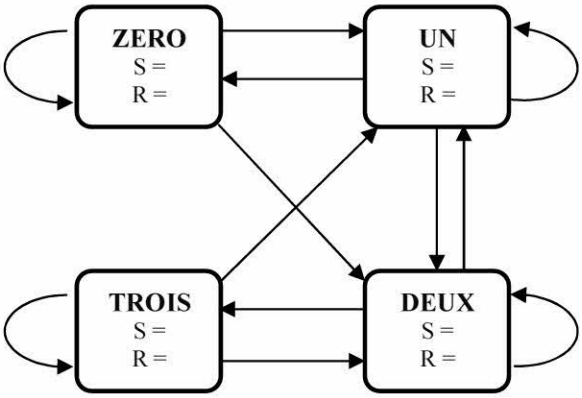


Figure 4

Etat	Q_1	Q_0	A	B	Q_1^+	Q_0^+	Etat Futur	R	S
zéro	**0**	**0**	**0**	**0**					
zéro	**0**	**0**	**0**	**1**					
zéro	**0**	**0**	**1**	**0**					
zéro	**0**	**0**	**1**	**1**					
Un	**0**	**1**	**0**	**0**					
Un	**0**	**1**	**0**	**1**					
Un	**0**	**1**	**1**	**0**					
Un	**0**	**1**	**1**	**1**					
Deux	**1**	**0**	**0**	**0**					
Deux	**1**	**0**	**0**	**1**					
Deux	**1**	**0**	**1**	**0**					
Deux	**1**	**0**	**1**	**1**					
Trois	**1**	**1**	**0**	**0**					
Trois	**1**	**1**	**0**	**1**					
Trois	**1**	**1**	**1**	**0**					
Trois	**1**	**1**	**1**	**1**					

Table 4

Exercice 4.4

Soit le monte charge de la figure 5, on cherche à réaliser la commande du monte-charge pour cela on donne la description suivante :

Le moteur est commandé par deux signaux m et d pour montée et descente. La commande élabore ces commandes à partir de ses entrées B, p1, p0 et init suivant le cahier des charges suivant :

- init provoque la descente du monte-charge jusqu'au palier p0. Ce signal est activé à la mise sous tension. On suppose qu'un plancher empêche le monte-charge d'être à un niveau inférieur à p0.
- Quand la cabine est entre deux étages, la dernière commande (m ou d) est maintenue.
- Quand la cabine est à un étage, si B est inactif elle s'arrête, si B est actif, elle change d'étage.

Notons que :

B : bouton de commande

pi : détecteurs de position

m, d : commandes du moteur

init : initialisation

- Décrire le fonctionnement de l'automate par un digramme descriptif.

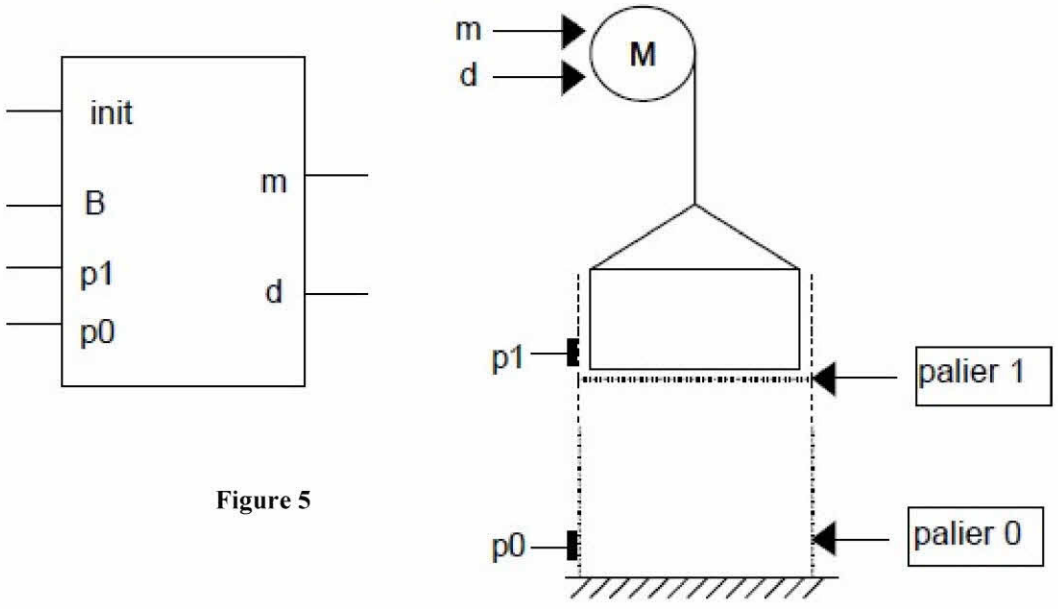


Figure 5

Exercice 4.5 (sans solution)

1. Concevoir le graphe d'état d'une machine à laver automatique muni d'un automate qui gère les étapes de lavages : Prélavage, Lavage, Rinçage, Essorage, Arrêt.

Supposons que les entrées de notre machine soient au nombre de trois :

- ✓ M : variable booléenne qui traduit la position du bouton Marche/Arrêt du lave-linge.
- ✓ P : variable booléenne qui indique si le programme de lavage sélectionné par l'utilisateur comporte ou non une phase de prélavage.
- ✓ C : valeur en minutes d'un chronomètre qui est remis à zéro automatiquement au début de chaque étape de lavage.

Les durées des différentes étapes de lavage sont fixées par le constructeur :

- ✓ Prélavage : 10 minutes
- ✓ Lavage : 30 minutes
- ✓ Rinçage : 10 minutes
- ✓ Essorage : 5 minutes

2. Si on suppose que la machine est dote d'une autre option appeler : Rinçage+ dont les rôles est de prolonger la phase rinçage (rincer plusieurs fois) afin d'éliminer l'effet du détergent. Quel sera alors le nouveau graphe d'état.

NB : on considère que l'automate à réaliser est une machine de Moore (sortie dépendent de l'état courant)

Exercice 4.6 (sans solution)

On veut réaliser, à l'aide d'un automate de Moore, un Soustracteur (figure 6) permettant de calculer la différence de deux opérandes A et B présentés en série poids faibles en tête. Ce soustracteur possède deux entrées notées A et B de 1 bit chacune, correspondant aux deux opérandes. Le soustracteur possède également deux sorties sur 1 bit chacune, notées **D** et **R**, correspondant à la différence et à la retenue.

On rappelle que la différence de 3 bits $\mathbf{a}_i - \mathbf{b}_i - \mathbf{r}_{i-1}$ produit un résultat sur deux bits $\mathbf{r}_i \mathbf{d}_i$. On a ainsi quatre résultats possibles 00, 01, 10 ou 11.

Le fonctionnement est le suivant : les opérandes sont présentés sur les entrées à raison de 1 bit par période d'horloge **CK**, la différence et la retenue sont disponibles pendant la période qui suit.

L'automate, réalisant le soustracteur série, possède quatre états codés sur deux bits notés R1 et R0. Le codage et la signification des états sont donnés dans la table 5.

R1	R0	Etat	Signification r d
0	0	ZERO	La différence des trois bits vaut 0 0
0	1	UN	La différence des trois bits vaut 0 1
1	1	DEUX	La différence des trois bits vaut 1 0
1	0	TROIS	La différence des trois bits vaut 1 1

Table 5

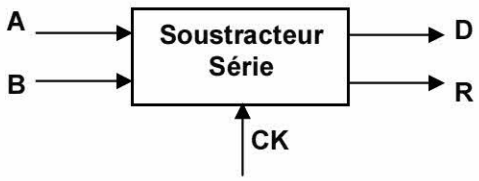


Figure 6

On suppose qu'initialement, lorsque commence la soustraction de deux opérandes (présentation de a0 et b0), l'automate est dans l'état **ZERO**.

1. Compléter le graphe de la figure 7, représentant l'automate, en précisant les expressions booléennes attachées à chaque transition. Donner également pour chaque état la valeur des sorties R et D.

2. Compléter la table de vérité (Table 6), qui définit la fonction de transition (signaux NR1 et NR0 représentant l'état futur), et la fonction de génération (signaux R et D).

3. En utilisant les tables de Karnaugh, donner les expressions booléennes simplifiées des signaux NR1, NR0, R et D.

Etat	R1	R0	A	B	NR1	NR0	Etat Futur	R	D
zéro	0	0	0	0					
zéro	0	0	0	1					
zéro	0	0	1	0					
zéro	0	0	1	1					
Un	0	1	0	0					
Un	0	1	0	1					
Un	0	1	1	0					
Un	0	1	1	1					
Deux	1	1	0	0					
Deux	1	1	0	1					
Deux	1	1	1	0					
Deux	1	1	1	1					
Trois	1	0	0	0					
Trois	1	0	0	1					
Trois	1	0	1	0					
Trois	1	0	1	1					

Table 6

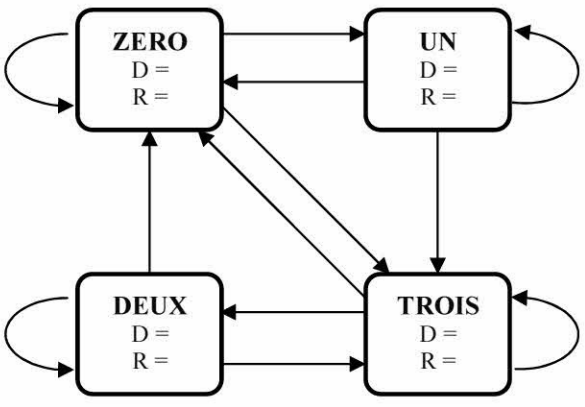


Figure 7

Exercice 4.7 (sans solution)

Soit le circuit à analyser de la figure 8 correspondant à un automate composer de : une entrée X, 3 bascules T (T_0, T_1, T_2) et deux sorties S_1 et S_2.

1- Mettre le circuit sous sa forme normale.

2- Compléter la table d'états (Table 7).

3- Déduire ses fonctions de transfert (états futurs et sorties).

4- Quel type d'automate avons-nous ? (justifier votre réponse)

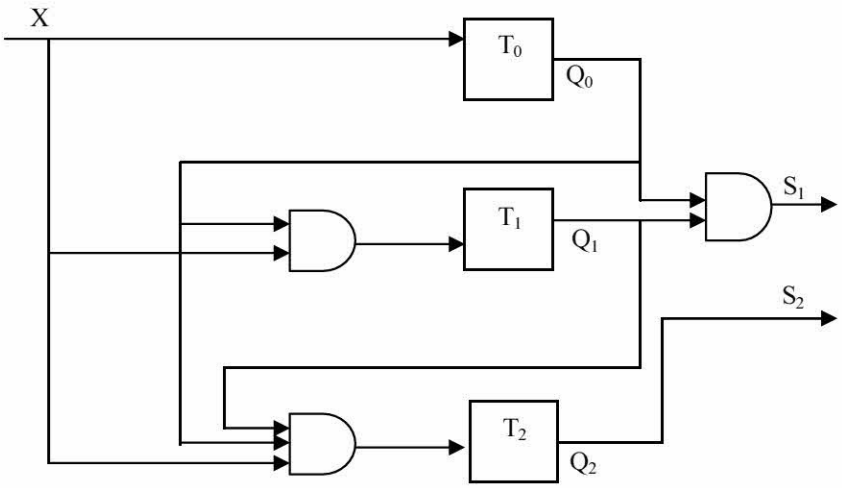


Figure 8

Etats présents			Entrée	Bascules			Etats Futurs			Sortie	
Q_2	Q_1	Q_0	X	T_2	T_1	T_0	Q_2^+	Q_1^+	Q_0^+	S_1	S_2

Table 7

Exercice 4.8

Soit le circuit à analyser de la figure 9 correspondant à un automate de Moore composer d'une entrée X, de 2 bascules T (T0, T1) et deux sorties S_1 et S_2.

1- Mettre le circuit sous sa forme normale.
2- Remplir la table d'état (Table 8).
3- Déduire ses fonctions de transfert.
4- Représenter le graphe d'état correspondant.

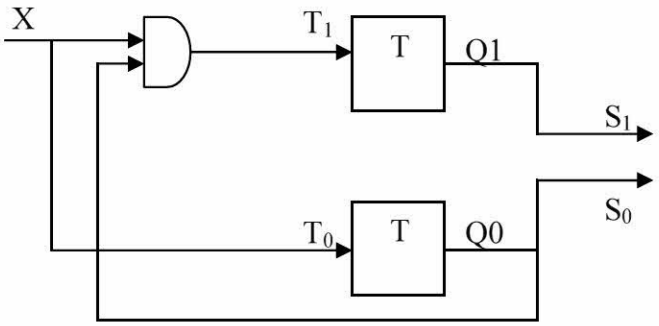


Figure 9

Etats Présents		Entrée	Bascules		Etats futurs	
Q_1	Q_0	X	T_1	T_0	Q_1^+	Q_0^+

Table 8

Chapitre 5
Les Circuits Intégrés

Chapitre 5

Les Circuits Intégrés

I. Introduction

L'invention des circuits intégrés est probablement la plus importante de tous les temps. Des circuits de taille très petite (mesurée en centimètre) intégrant des milliers voir des millions de portes logiques permettant un ainsi une grande révolution technologique dans tous les appareils électroniques en passant des appareils de très grandes tailles à des appareils de tailles de plus en plus petites créant ainsi une nouvelle orientation, des chercheurs des semi-conducteurs, vers la miniaturisation.

L'objectif principal de ce chapitre est de familiariser l'étudiant avec le côté matériel des circuits logiques étudier dans les chapitres précédents. Une description détaillée, sur la constitution interne des circuits intégrés ainsi que leurs utilisations, permettra à l'étudiant de faire une analogie entre le diagramme logique et la réalisation matérielle d'un circuit quelconque.

Les références utilisées pour l'élaboration de ce chapitre sont principalement : [12] [13] et [14].

II. Circuits Intégrés logiques

Les circuits intégrés (CI) logique (appelés aussi puce ou chip (En Anglais)) sont les composants que l'on trouve pratiquement dans tous les appareils électroniques (appareil électroménager, ordinateurs, appareil de mesures,…etc)

Ces composants électroniques sont basés sur un semi-conducteur, reproduisant une, ou plusieurs, fonctions électroniques plus ou moins complexes et intégrant souvent plusieurs types de composants électroniques de base dans un volume réduit (sur une petite plaque), rendant le circuit facile à mettre en œuvre.

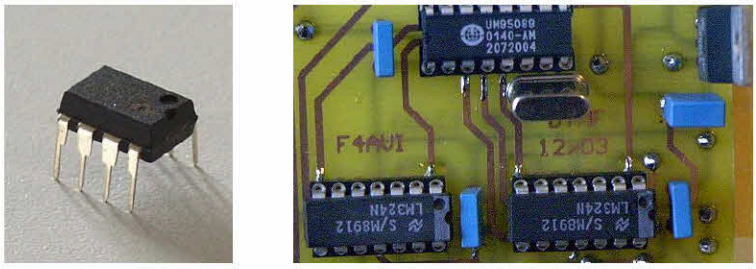


Fig 5.1 : Illustration de quelques circuits intégrés

III. Où trouve t-on les circuits Intégré logique ?

Dans tout appareil fonctionnant numériquement à savoir :

- Les appareils électroménagers : machine à laver, téléviseur, démodulateur, aspirateur, robot de cuisine,…etc.
- Les ordinateurs, tablette, téléphones portables, …etc.
- Les appareils de mesure utilisés dans différents domaines : chimie, physique, géologie, santé,…etc.
- Distributeurs de boissons, journaux, de billet, etc…
- Etc…

IV. Invention des circuits intégrés

En 1958, l'Américain Jack Kilby (1923 – 2005) (travaillant chez Texas Instruments) invente le premier circuit intégré, mettant ainsi les bases du matériel informatique moderne et permettant donc la réalisation de mémoires, ainsi que d'unités logiques et arithmétiques. Cette découverte a valu à Kilby un prix Nobel de physique en 2000.

V. Description matérielles

La partie active (**Die**) d'un circuit logique est formée d'une palette de silicium d'environ 5x5 mm sur laquelle ont été intégrées, par divers procédés technologiques, les portes qui constituent ce circuit.

La puce est enveloppée par un boîtier rectangulaire en plastique ou en céramique. Sur les 2 longs côtés de ce boîtier sont disposées de façon symétrique des broches (ou pattes) permettant d'assurer les connexions électriques (signaux d'entrées/sorties) des circuits logiques internes. Cette représentation classique des puces est connue sous l'appellation de circuit en boîtier **DIP** ou **DIL** (Dual Inline Package/Boitier avec deux lignes).

V.1 Le Die (Circuit intégrés)

Le **Die** est la partie élémentaire, de forme rectangulaire, reproduite comme une copie conforme avec une matrice sur une tranche de silicium en cours de fabrication. Il correspond au circuit intégré qui sera ensuite découpé et qu'on appellera une puce avant qu'elle ne soit encapsulée pour donner un circuit intégré, prêt à être monté sur une carte.

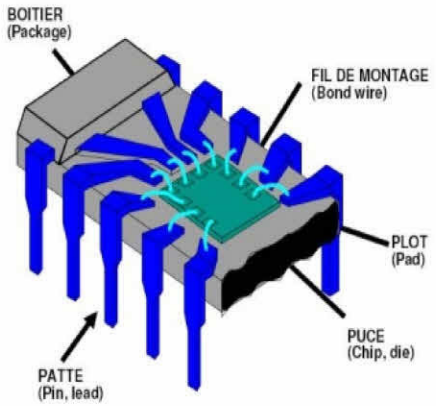


Fig 5.2 : Illustration du Die

V.2 Technologies de fabrication de CI

Le *Die* d'un circuit intégré comprend sous des formes miniaturisées principalement des transistors, des diodes, des résistances, des condensateurs, plus rarement des inductances, car elles sont plus difficilement miniaturisables.

- **DTL** (**D**iode-**T**ransistor-**L**ogic) , Rapidement éliminée en faveur du TTL (plus rapide): technologie obsolète
- **TTL** (**T**ransistor-**T**ransistor-**L**ogic) (Vcc = 5 V, H = 5 V, L = 0 V) Meilleur compromis vitesse / consommation avant l'apparition du CMOS
- **ECL** (**E**mitter-**C**oupled-**L**ogic) Consommation importante, mais très rapide (communications rapides Gbit/s);
- **CMOS** (**C**omplementary **M**etal-**O**xide-**S**emiconductor) (Vcc = 0.8 V à 18 V, H = Vcc, L = 0 V) Très basse consommation; technologie actuellement dominante

Notons que :

Vcc : tension d'alimentation du circuit

H : niveau de tension haut équivalente à 1 logique

L : niveau de tension bas équivalent à 0 logique.

VI. Classement des circuits intégrés

Les circuits intégrés numériques les plus simples sont des portes logiques (*et*, *ou* et *non*), les plus complexes sont les microprocesseurs et les plus denses sont les mémoires.

On trouve de nombreux circuits intégrés dédiés à des applications spécifiques (ou *ASIC* pour *Application-specific integrated circuit*):

- Notamment pour le traitement du signal (traitement d'image, compression vidéo...) on parle alors de processeur de signal numérique (ou ***DSP*** pour *Digital Signal Processor*).
- Une famille importante de circuits intégrés est celle des composants de logique programmable (**FPGA**, **CPLD**). Ces composants sont amenés à remplacer les portes logiques simples en raison de leur grande densité d'intégration.

Les circuits intégrés ont été classés en **six** familles de produits selon la densité d'intégration (nombre de portes par circuit ou nombre de transistors par circuit) :

- **SSI** : (**S**mall **S**cale **I**ntegration) Circuit à faible capacité d'intégration.
- **MSI** : (**M**edium **S**cale **I**ntegration) Circuit à moyenne capacité d'intégration
- **LSI** : (**L**arge **S**cale **I**ntegration) Circuit à haute capacité d'intégration.
- **VLSI**:(**V**ery **L**arge **S**cale **I**ntegration) Circuit à très haute capacité d'intégration.
- **ULSI:** (**U**ltra **L**arge **S**cale **I**ntegration) Circuit à une extrême capacité d'intégration.
- **GLSI:** (**G**iga **L**arge **S**cale **I**ntegration) Circuit à une capacité d'intégration géante.

Nom	Année	Nombre de portes logique	Exemple
SSI	1961-1966	1-10	Portes logiques AND,OR,NOT,..etc
MSI	1967-1971	10-100	Bascules , compteurs, multiplexeurs, décodeurs,... etc
LSI	1972-1980	100-10 000	Mémoire de petite capacité, Circuit logique programmable
VLSI	1981-1990	10 000 – 100 000	Mémoire de capacité importante, micro-processeur
ULSI	1990-2000	100 000- 1000 000	Micro processeur graphique
GLSI	2000-aujourd'hui	> 1 000 000	Pentium Dual Core microprocesseur

Tab 5.1 : Classement des circuits intégrés

VII. Présentation des CI

Un circuit intégré renferme plusieurs portes logiques, dont les entrées et les sorties sont accessibles sur les différentes bornes du circuit intégré.

On peut remarquer sur le haut des circuits intégrés un petit creux appelé « ergo ». L'ergo permet d'orienter correctement le circuit intégré afin de repérer les différentes bornes.

Les broches (pin) d'un circuit intégré sont numérotées. Pour déterminer la position du pin n°1, il faut repérer une encoche sur le composant tel que:

En regardant le circuit intégré avec l'ergo vers le haut, la borne n°1 est la borne située en haut à gauche, les autres bornes sont numérotées en tournant dans le sens inverse des aiguilles d'une montre.

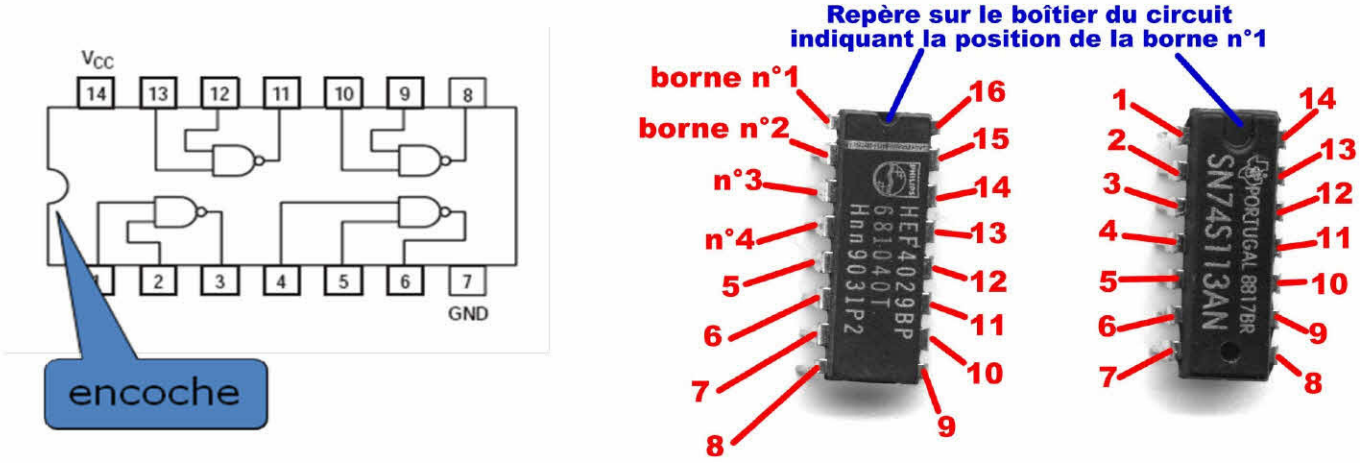


Fig 5.3 : Présentation des circuits intégrés

VIII. Identification des CI

Chaque circuit intégré possède une référence imprimée sur le dessus de son boîtier. Cette référence est composée de 4 à 7 caractères (chiffres et/ou lettres).

La photo ci-contre la référence du circuit de gauche est 4029, et la référence du circuit de droite est 74S113.

- Pour connaître la fonction d'un circuit intégré dont on connaît la référence il faut consulter le **Mémotech électronique [12].**
- Il existe 2 grandes **familles de circuits logiques :**

  Les circuits dont la référence est de la forme **4000, appelé la technologie CMOS.**

  Les circuits dont la référence commence par **74, appelé la technologie TTL**

Exemple 1 : circuit **TTL** nommé **SN74LS00**

SN : signifie que le constructeur est Texas Instrument

74 : désigne les circuits intégrés grands publics qui supportent une température ambiante comprise entre 0 et 70 degré.

LS ou HCT: indiquent la sous famille du circuit TTL

00 : Les derniers chiffres indiquent la fonction logique réalisé par le composant (00=Porte NAND, 02=Porte NOR, 08=Porte AND etc.)

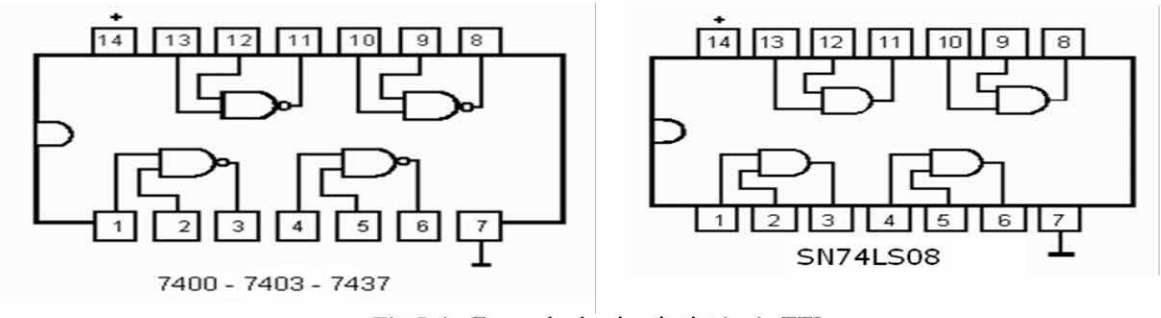


Fig 5.4 : Exemple de circuits intégrés TTL

Exemple 2 : circuit **CMOS** nommé **CD4011BE**.

CD : Préfixe utilisé par Texas Instruments.

4011 : Numéro du circuit. Il s'agit ici d'un quadruple porte NAND (NON-ET) à deux entrées chacune.

B : Indique que la tension maximale est de 18 volts

E : Indication que le circuit est encapsulé dans un boîtier DIP.

Codes Circuit = Préfixe fabricant + numéro du circuit + suffixe + code boîtier

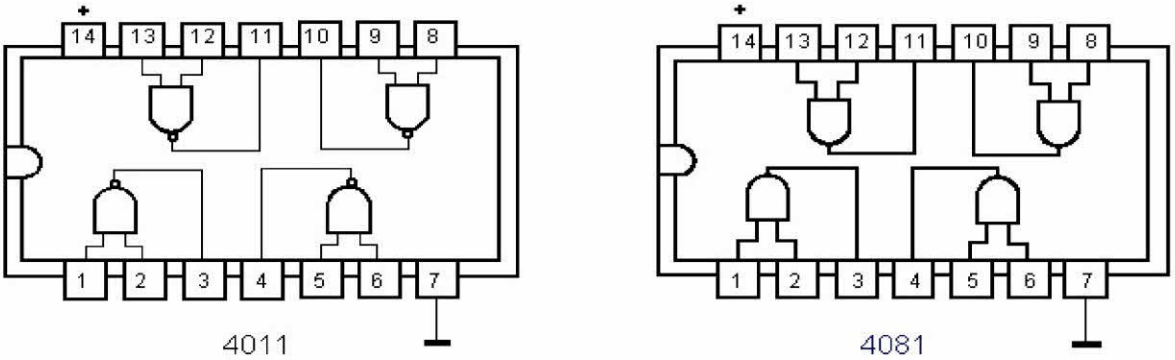


Fig 5.5 : Exemple de circuits intégrés CMOS

Remarque

Les figures 5.6 et 5.7 illustrent quelques exemples de circuits intégrés de portes logiques des familles TTL et CMOS.

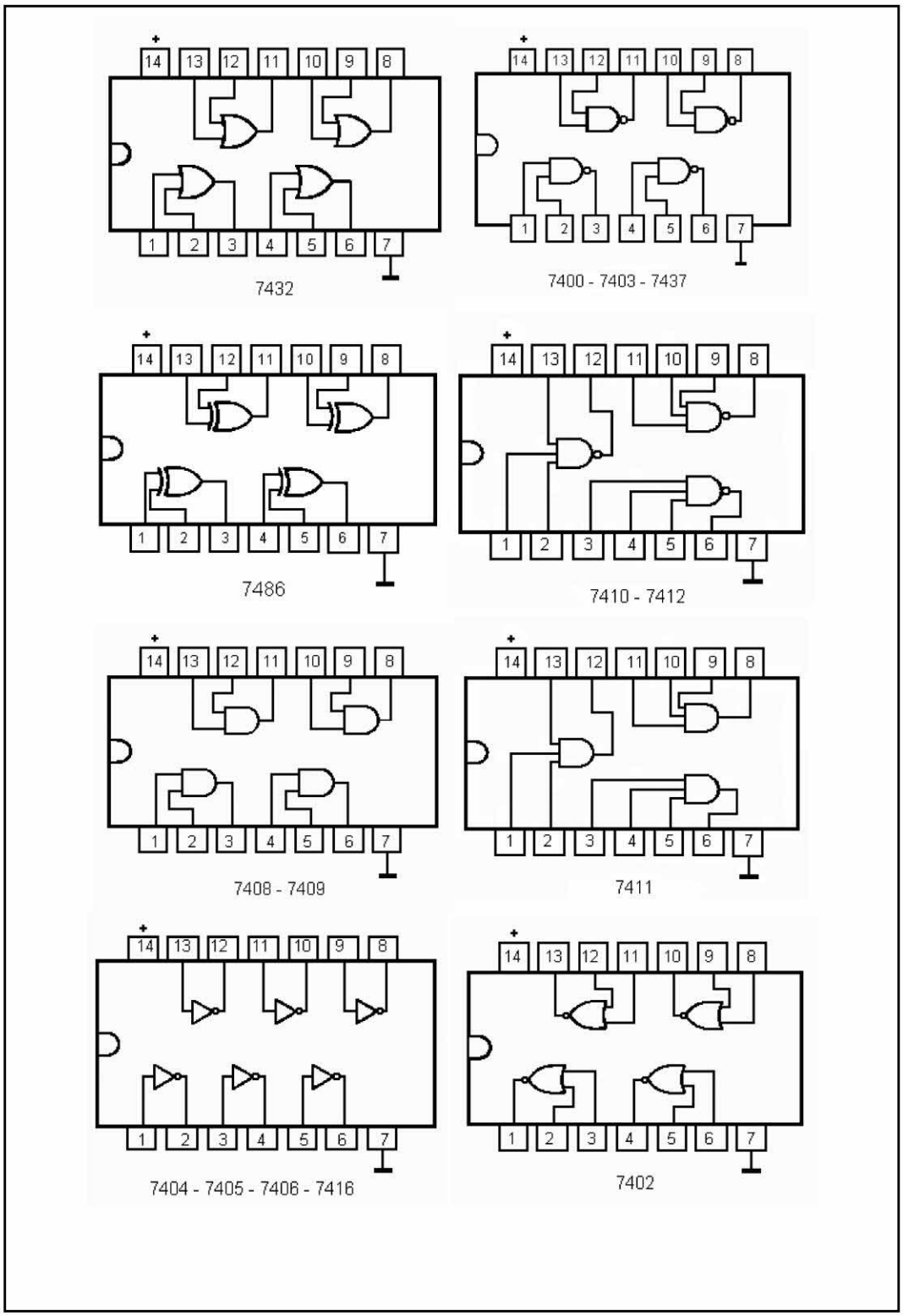


Fig 5.6 : Quelques circuits intégrés TTL

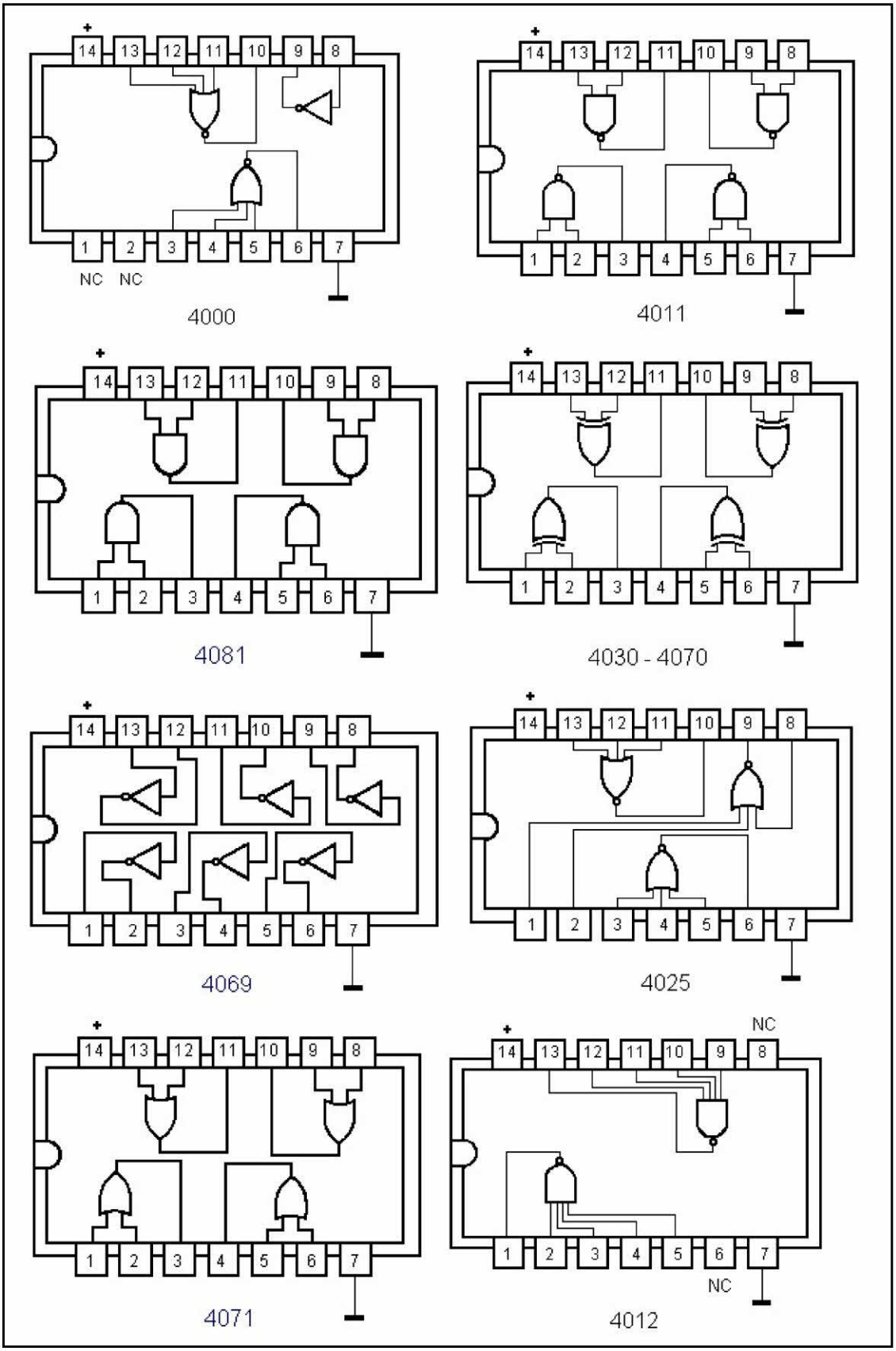


Fig 5.7 : Quelques circuits intégrés CMOS

Exemple

Soit la fonction $X_1(S_1, S_2, S_3)$ définie comme suit : $X_1 = \overline{S_1.S_2}.S_3$

Tracer le logigramme de X_1 et représenter le montage de la fonction à base du circuit 74LS00.

Logigramme de X_1

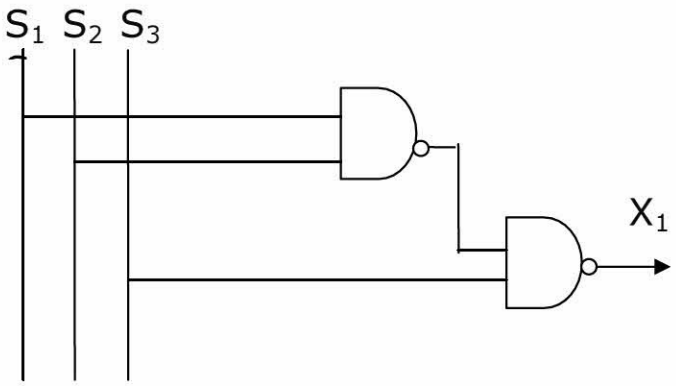


Fig 5.8 : Logigramme de X_1

Montage à base de circuit intégré 74LS00

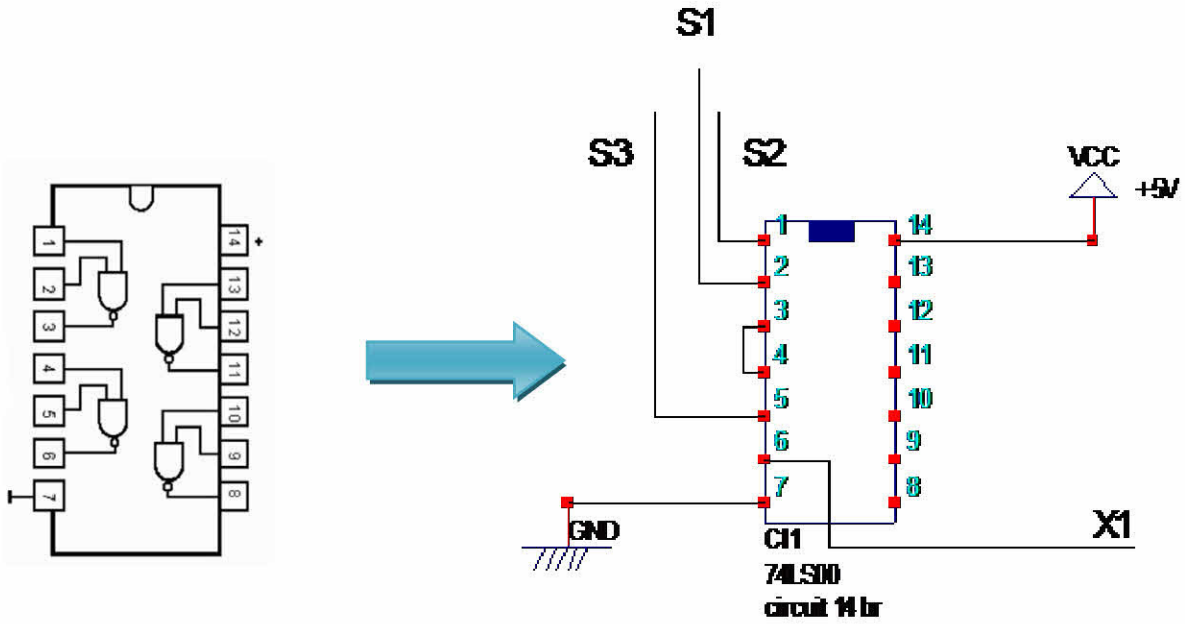


Fig 5.9 : Schéma de câblage de la fonction X_1 à base de CI 74LS00

IX. Conclusion

La réalisation matérielle de n'importe quel système nécessite une bonne connaissance des circuits intégrés, leur famille et comment les disposer et les identifier.

Le nombre de circuits intégrés à utiliser doit respecter les critères : encombrement (espace occupé), consommation électrique, et coût.

Corrigé des Exercices

Chapitre 2

Exercice 2.1

1-a) Table de vérité

a1	a0	b1	b0	S1	S0	C
0	0	0	0	0	0	0
0	0	0	1	0	1	0
0	0	1	0	1	0	0
0	0	1	1	1	1	0
0	1	0	0	0	1	0
0	1	0	1	1	0	0
0	1	1	0	1	1	0
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	1	0
1	0	1	0	0	0	1
1	0	1	1	0	1	1
1	1	0	0	1	1	0
1	1	0	1	0	0	1
1	1	1	0	0	1	1
1	1	1	1	1	0	1

1-b) Tables de Karnaugh

$S_1 = \Sigma(2,3,5,6,8,9,12,15)$

a1a0 \ b1b0	00	01	11	10
00	0	0	1	1
01	0	1	0	1
11	1	0	1	0
10	1	1	0	0

$$S_1 = a_1\overline{b_1}\,\overline{b_0} + a_1\overline{a_0}\,\overline{b_1} + \overline{a_1}\,\overline{a_0}b_1 + \overline{a_1}b_1\overline{b_0} + \overline{a_1}a_0\overline{b_1}b_0 + a_1a_0b_1b_0$$

$$S_1 = a_0b_0 \oplus a_1 \oplus b_1$$

$S_0 = \Sigma(1,3,4,6,9,11,12,14)$

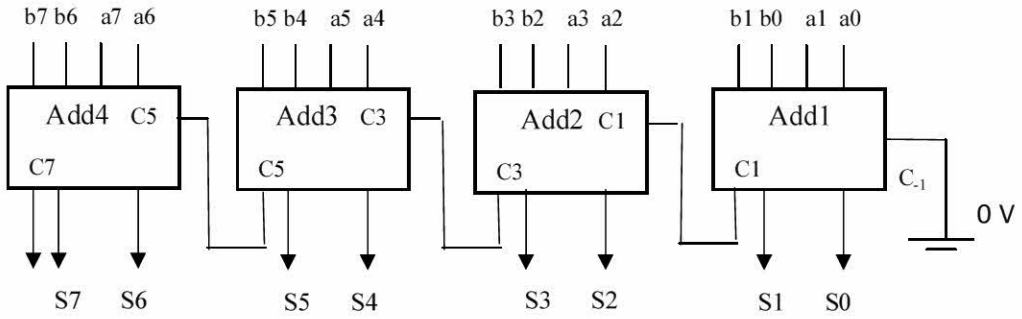
On trouve : $S_0 = \overline{a_0}b_0 + a_0\overline{b_0} = a_0 \oplus b_0$

$C = \Sigma(7,10,11,13,14,15)$

On trouve : $C = a_1a_0b_0 + b_0b_1a_0 + a_1b_1 = b_0a_0(b_1 + a_1) + b_1a_1$

1-c) Logigramme déjà vu en TD.

2) Additionneur à 8 bits :



Exercice 2.2

1) Table de vérité du codeur octal-binaire

V	D7	D6	D5	D4	D3	D2	D1	D0	A	B	C
0	x	x	x	x	x	x	x	x	0	0	0
1	0	0	0	0	0	0	0	1	0	0	0
1	0	0	0	0	0	0	1	0	0	0	1
1	0	0	0	0	0	1	0	0	0	1	0
1	0	0	0	0	1	0	0	0	0	1	1
1	0	0	0	1	0	0	0	0	1	0	0
1	0	0	1	0	0	0	0	0	1	0	1
1	0	1	0	0	0	0	0	0	1	1	0
1	1	0	0	0	0	0	0	0	1	1	1

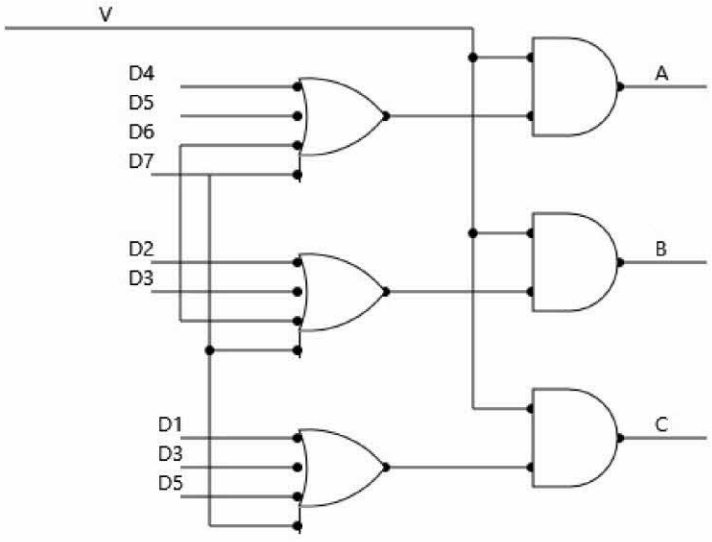
2) Expressions des sorties :

$A = V(D_4 + D_5 + D_6 + D_7)$

$B = V(D_2 + D_3 + D_6 + D_7)$

$C = V(D_1 + D_3 + D_5 + D_7)$

3) Logigramme



Exercice 2.3

1) Table de vérité du décodeur BCD/7-segments :

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

2) Tables de Karnaugh :

a

AB \ CD	00	01	11	10
00	1	0	1	1
01	0	1	1	1
11	X	X	X	X
10	1	1	X	X

$$a = A + C + \overline{B \oplus D}$$

Pour le reste et en suivant la même méthode on trouve :

$$b = \bar{B} + (\overline{C \oplus D})$$

$$c = \bar{C} + D + B$$

$$d = A + C\bar{D} + (B \oplus \bar{C}D)$$

$$e = \bar{D}(C + \bar{B})$$

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

$$g = (B \oplus C) + A\bar{C} + C\bar{D}$$

3) Le logigramme : faire de même que pour l'exercice précédent.

Exercice 2.4

1) Table de vérité :

BCD				Excess-3			
a	**b**	**c**	**d**	**x**	**y**	**z**	**t**
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0

Expressions simplifiées :

X :

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

$$x = bd + bc + a = b(c + d) + a$$

Y :

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

$$y = b\bar{c}\bar{d} + d\bar{b} + \overline{bc}$$

$$y = b \oplus \bar{c}\bar{d}$$

Z :

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

$$z = \bar{c}\bar{d} + cd = \overline{c \oplus d}$$

t :

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

$t = \bar{d}$

2) Logigramme : faire de même que précédemment

Exercice 2.5

1) Ce circuit possède 8 entrées vers une seule sortie et possède 3 lignes d'adressage des 8 lignes d'entrées, donc c'est un multiplexeur (MUX 8 vers 1), où :

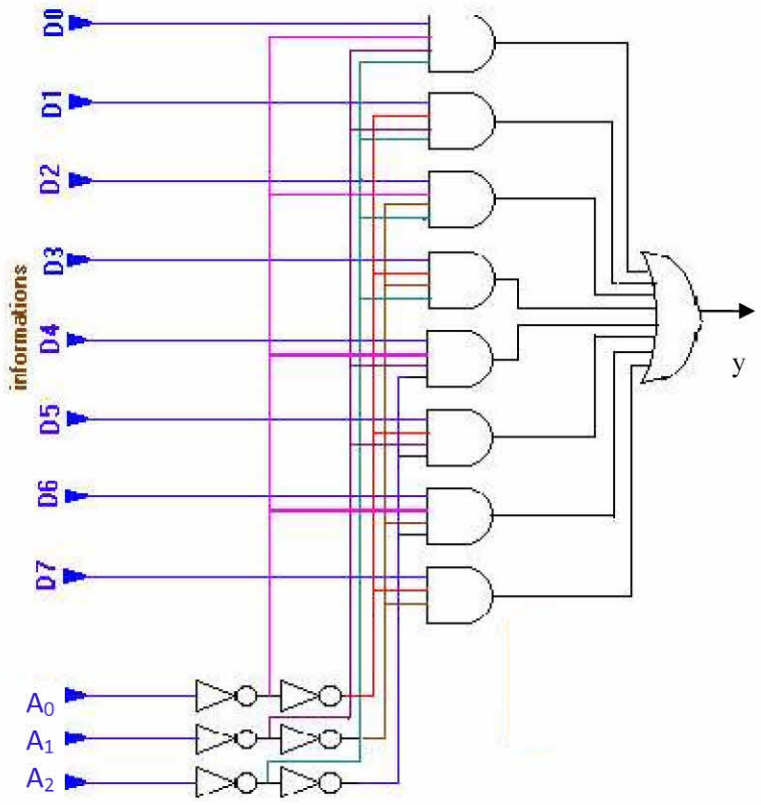
$d_0 - d_7$: lignes d'entrées, y : la sortie, A_0, A_1, A_2 : lignes d'adressage, E : entrée de validation active au niveau bas

2) Table de vérité :

E	A2	A1	A0	Y
1	X	X	X	0
0	0	0	0	d0
0	0	0	1	d1
0	0	1	0	d2
0	0	1	1	d3
0	1	0	0	d4
0	1	0	1	d5
0	1	1	0	d6
0	1	1	1	d7

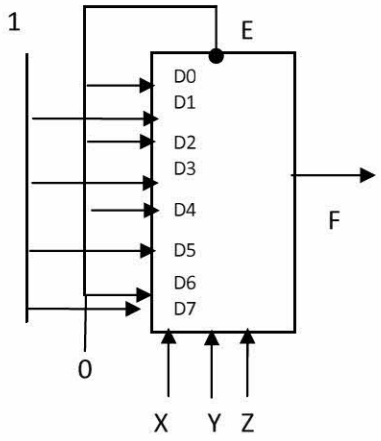
$$y = \bar{E}[(\overline{A_2}\,\overline{A_1}\,\overline{A_0})d_0 + (\overline{A_2}\,\overline{A_1}A_0)d_1 + (\overline{A_2}A_1\overline{A_0})d_2 + (\overline{A_2}A_1A_0)d_3 + (A_2\overline{A_1}\,\overline{A_0})d_4 + (A_2\overline{A_1}A_0)d_5 + (A_2A_1\overline{A_0})d_6 + (A_2A_1A_0)d_7]$$

3) Logigramme



4) $F(x,y,z) = \Sigma(1,3,5,7)$

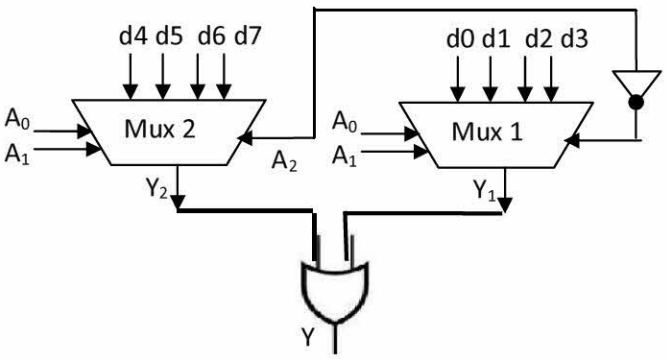
x	y	z	f	di
0	0	0	0	d0 = 0
0	0	1	1	d1 = 1
0	1	0	0	d2 = 0
0	1	1	1	d3 = 1
1	0	0	0	d4 = 0
1	0	1	1	d5 = 1
1	1	0	0	d6 = 0
1	1	1	1	d7 = 1



5) Nous avons besoin de deux Mux 4 vers 1 pour réaliser le Mux 8 vers 1. La table de vérité d'un Mux 8 vers 1 sera partagée entre les deux Mux 4 vers 1 comme suit :

Validation du boitier à utiliser

A2	A1	A0	Y	
0	0	0	d0	Mux 4 vers 1 Niveau 1
0	0	1	d1	
0	1	0	d2	
0	1	1	d3	
1	0	0	d4	Mux 4 vers 1 Niveau 2
1	0	1	d5	
1	1	0	d6	
1	1	1	d7	



Exercice 2.6

1) Le circuit sous cette forme ne fonctionne pas correctement car il dispose de 3 lignes d'adressage ce qui n'est pas suffisant pour adresser les 16 sorties il nous faut donc une quatrième ligne d'adressage ($4^2 = 16$) pour pouvoir adresser toutes les sorties

2) Table de vérité :

E	A3	A2	A1	A0	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7	Y8	…	Y15
1	X	X	X	X	0	0	0	0	0	0	0	0	0		0
0	0	0	0	0	D	0	0	0	0	0	0	0	0		0
0	0	0	0	1	0	D	0	0	0	0	0	0	0		0
0	0	0	1	0	0	0	D	0	0	0	0	0	0		0
0	0	0	1	1	0	0	0	D	0	0	0	0	0		0
0	0	1	0	0	0	0	0	0	D	0	0	0	0		0
0	0	1	0	1	0	0	0	0	0	D	0	0	0		0
0	0	1	1	0	0	0	0	0	0	0	D	0	0		0
0	0	1	1	1	0	0	0	0	0	0	0	D	0		0
0	1	0	0	0	0	0	0	0	0	0	0	0	D	.....	0
0	1	0	0	1	0	0	0	0	0	0	0	0	0		0
0	1	0	1	0	0	0	0	0	0	0	0	0	0		0
0	1	0	1	1	0	0	0	0	0	0	0	0	0		0
0	1	1	0	0	0	0	0	0	0	0	0	0	0		0
0	1	1	0	1	0	0	0	0	0	0	0	0	0		0
0	1	1	1	0	0	0	0	0	0	0	0	0	0		0
0	1	1	1	1	0	0	0	0	0	0	0	0	0	.....	D

1) Expressions des sorties :

$$y_0 = \bar{E}D(\overline{A_3}\,\overline{A_2}\,\overline{A_1}\,\overline{A_0}) \;, y_1 = \bar{E}D(\overline{A_3}\,\overline{A_2}\,\overline{A_1}A_0) \;,$$

$$y_2 = \bar{E}D(\overline{A_3}\,\overline{A_2}\,A_1\overline{A_0}) \;, y_3 = \bar{E}D(\overline{A_3}\,\overline{A_2}\,A_1A_0) \;, y_4 = \bar{E}D(\overline{A_3}\,A_2\overline{A_1}\,\overline{A_0}) \;,$$

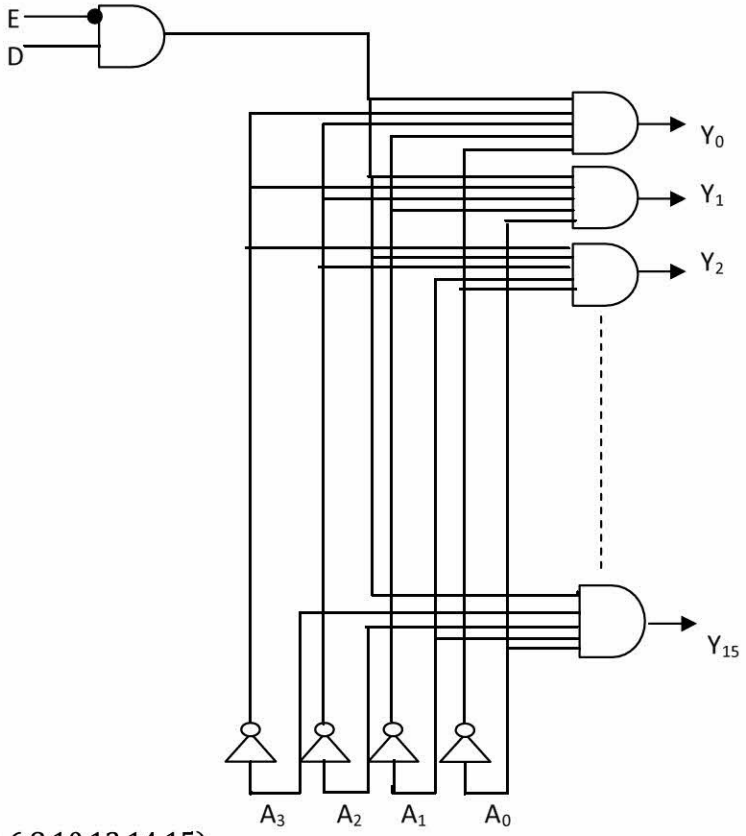
$$y_5 = \bar{E}D(\overline{A_3}\,A_2\overline{A_1}\,A_0) \;, y_6 = \bar{E}D(\overline{A_3}\,A_2\,A_1\overline{A_0}) \;, y_7 = \bar{E}D(\overline{A_3}\,A_2\,A_1A_0) \;,$$

$$y_8 = \bar{E}D(A_3\,\overline{A_2}\,\overline{A_1}\,\overline{A_0}) \;, y_9 = \bar{E}D(A_3\,\overline{A_2}\,\overline{A_1}A_0) \;, y_{10} = \bar{E}D(A_3\,\overline{A_2}\,A_1\overline{A_0}) \;,$$

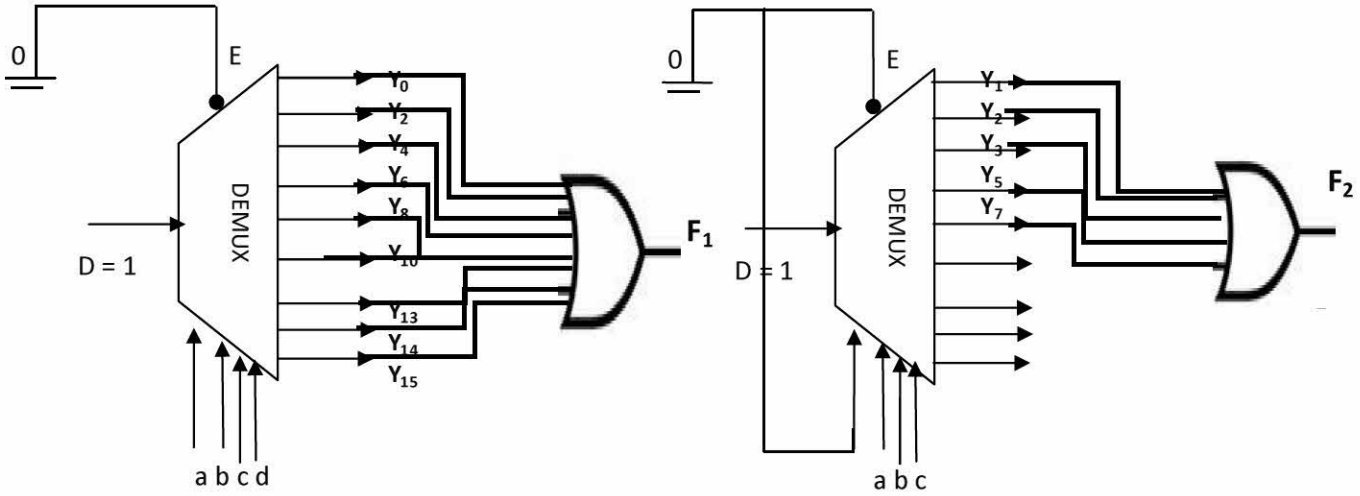
$$y_{11} = \bar{E}D(A_3\,\overline{A_2}\,A_1A_0) \;, y_{12} = \bar{E}D(A_3\,A_2\overline{A_1}\,\overline{A_0}) \;, y_{13} = \bar{E}D(A_3\,A_2\overline{A_1}\,A_0) \;,$$

$$y_{14} = \bar{E}D(A_3\,A_2\,A_1\overline{A_0}) \;, y_{15} = \bar{E}D(A_3\,A_2\,A_1A_0)$$

1) Logigramme



2) $F_1(a,b,c,d) = \sum(0,2,4,6,8,10,13,14,15)$



3) $F_2(a,b,c) = \sum(1,2,3,5,7)$

Il est important de forcer l'entrée de l'adresse de plus fort poids à zero.

Exercice 2.7

1) Table de vérité :

b1	b0	a1	a0	E	I	S
0	0	0	0	1	0	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	0	1	0
0	1	0	1	1	0	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	0	1	0
1	0	0	1	0	1	0
1	0	1	0	1	0	0
1	0	1	1	0	0	1
1	1	0	0	0	1	0
1	1	0	1	0	1	0
1	1	1	0	0	1	0
1	1	1	1	1	0	0

2) Expressions simplifiées :

E : pour E, on ne va pas utiliser la table de Karnaugh par absence d'adjacence.

$$E = \bar{b_1}\,\bar{b_0}\,\bar{a_1}\,\bar{a_0} + \bar{b_1}b_0\bar{a_1}a_0 + b_1\bar{b_0}a_1\bar{a_0} + b_1b_0a_1a_0$$
$$= \bar{b_1}\,\bar{a_1}(\bar{b_0}\,\bar{a_0} + b_0a_0) + b_1a_1(\bar{b_0}\,\bar{a_0} + b_0a_0)$$
$$E = \overline{(b_0 \oplus a_0)}.\overline{(b_1 \oplus a_1)}$$

S :

b1b0 \ a1a0	00	01	11	10
00	0	1	1	1
01	0	0	1	1
11	0	0	0	0
10	0	0	1	0

$$S = \bar{b_1}a_1 + \bar{b_0}a_0(\bar{b_1} + a_1)$$

I :

b1b0 \ a1a0	00	01	11	10
00	0	0	0	0
01	1	0	0	0
11	1	1	0	1
10	1	1	0	0

$$I = b_1\bar{a_1} + b_1b_0\bar{a}_0 + b_0\bar{a}_1\bar{a}_0$$
$$I = b_1\bar{a}_1 + b_0\bar{a}_0(b_1 + \bar{a_1})$$

3) Logigramme : Utiliser les portes à deux entrées.

Exercice 2.8

1) Les entrées du distributeur sont : P(introduction de la pièce), B1(café), B2(thé), B3(lait).

Les sorties sont : R(restitution de la pièce), distribution des boissons : C(café), T(thé), L(lait).

P	B1	B2	B3	C	T	L	R
0	x	x	x	0	0	0	0
1	0	0	0	0	0	0	1
1	0	0	1	0	0	1	1
1	0	1	0	0	1	0	0
1	0	1	1	0	1	1	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	1	0
1	1	1	0	0	0	0	1
1	1	1	1	0	0	0	1

2) Simplification par Karnaugh :

Café : $C = PB_1\overline{B_2}$

PB1 \ B2B3	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	0	0
10	0	0	0	0

En suivant la même méthode, on trouve :

$T = P\overline{B}_1B_2$

$L = P\overline{B_2}B_3 + P\overline{B_1}B_3$

$R = P\overline{B_1}\,\overline{B_2} + PB_1B_2$

3) Logigramme à base de NAND :

$\overline{\overline{C}} = \overline{\overline{PB_1\overline{B_2}}}$ $\qquad\qquad$ $\overline{\overline{T}} = \overline{\overline{P\overline{B_1}B_2}}$

$\overline{\overline{L}} = \overline{\overline{P\overline{B_2}B_3 + P\overline{B_1}B_3}} = \overline{\overline{P\overline{B_2}B_3} \cdot \overline{P\overline{B_1}B_3}}$

$\overline{\overline{R}} = \overline{\overline{P\overline{B_1}\,\overline{B_2} + PB_1B_2}} = \overline{\overline{P\overline{B_1}\,\overline{B_2}} \cdot \overline{PB_1B_2}}$

Logigramme déjà vu en TD.

Exercice 2.9

Les entrées du distributeur sont : I1,I2, I3.

Les sorties sont : M1 et M2

1) Table de vérité :

I1	I2	I3	M1	M2
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	1
1	0	0	0	1
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

2) Simplification par Karnaugh :

$M_1 = I_1I_2 + I_1I_3 + I_2I_3$

I1 \ I2 I3	00	01	11	10
0	0	0	1	0
1	0	1	1	1

$M_2 = I_1 + I_2 + I_3$

I1 \ I2 I3	00	01	11	10
0	0	1	1	1
1	1	1	1	1

3) Logigramme à base de NAND

$\overline{\overline{M_1}} = \overline{\overline{I_1I_2 + I_1I_3 + I_2I_3}} = \overline{\overline{I_1I_2}.\overline{I_1I_3}.\overline{I_2I_3}}$

$\overline{\overline{M_2}} = \overline{\overline{I_1 + I_2 + I_3}} = \overline{\overline{I_1}.\overline{I_2}.\overline{I_3}}$

4) Logigramme à base de NOR

$\overline{\overline{M_1}} = \overline{\overline{\overline{I_1I_2} + \overline{I_1I_3} + \overline{I_2I_3}}} = \overline{\overline{I_1} + \overline{I_2} + \overline{I_1} + \overline{I_3} + \overline{I_2} + \overline{I_3}}$

$\overline{\overline{M_2}} = \overline{\overline{I_1 + I_2 + I_3}}$

Exercice 2.10

1. Tables de vérité

x	y	z	a	b	c	d	e	f	g
0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0
0	1	0	1	0	0	0	0	0	1
0	1	1	1	0	0	1	0	0	1
1	0	0	1	1	0	0	0	1	1
1	0	1	1	1	0	1	0	1	1
1	1	0	1	1	1	0	1	1	1
1	1	1	0	0	0	0	0	0	0

2. Première forme canonique

$a = \bar{x}y\bar{z} + \bar{x}yz + x\bar{y}\bar{z} + x\bar{y}z + xy\bar{z}$

$b = x\bar{y}\bar{z} + x\bar{y}z + xy\bar{z}$

$c = xy\bar{z}$

$d = \bar{x}\bar{y}z + \bar{x}yz + x\bar{y}z$

3. Expressions simplifiées

x \ yz	00	01	11	10
0	0	0	1	1
1	1	1	0	1

x \ yz	00	01	11	10
0	0	0	1	1
1	1	1	0	1

x \ yz	00	01	11	10
0	0	1	1	0
1	0	1	0	0

$a = g = x\bar{y} + \bar{x}y + y\bar{z}$ $\quad b = f = x\bar{y} + x\bar{z}$ $\quad d = \bar{y}z + \bar{x}z$

$c = e = xy\bar{z}$

4. Logigramme à base de NAND

$\bar{\bar{a}} = \bar{\bar{g}} = \overline{\overline{x\bar{y} + \bar{x}y + y\bar{z}}} = \overline{\overline{x\bar{y}}.\overline{\bar{x}y}.\overline{y\bar{z}}}$

$\bar{\bar{b}} = \bar{\bar{f}} = \overline{\overline{x\bar{y} + x\bar{z}}} = \overline{\overline{x\bar{y}}.\overline{x\bar{z}}}$

$\bar{\bar{d}} = \overline{\overline{\bar{y}z + \bar{x}z}} = \overline{\overline{\bar{y}z}.\overline{\bar{x}z}}$

$\bar{\bar{c}} = \bar{\bar{e}} = \overline{\overline{xy\bar{z}}}$

Exercice 2.11

Les entrées du contrôleur des lignes téléphoniques sont : M(Marche/Arrêt), les lignes A, B, C, D. Les sorties sont les lampes : V, J, O, R

1) Table de vérité :

M	A	B	C	D	V	J	O	R
0	x	x	x	x	0	0	0	0
1	0	0	0	0	1	0	0	0
1	0	0	0	1	0	1	0	0
1	0	0	1	0	0	1	0	0
1	0	0	1	1	0	0	1	0
1	0	1	0	0	0	1	0	0
1	0	1	0	1	0	0	1	0
1	0	1	1	0	0	0	1	0
1	0	1	1	1	0	0	0	1
1	1	0	0	0	0	1	0	0
1	1	0	0	1	0	0	1	0
1	1	0	1	0	0	0	1	0
1	1	0	1	1	0	0	0	1
1	1	1	0	0	0	0	1	0
1	1	1	0	1	0	0	0	1
1	1	1	1	0	0	0	0	1
1	1	1	1	1	0	0	0	1

2) Simplification

$V = M\bar{A}\bar{B}\bar{C}\bar{D}$ (Sans table de karnaugh vu l'absence de l'adjacence voir table de K)

$J = M[\bar{A}\,\bar{B}(C \oplus D) + \bar{C}\,\bar{D}(A \oplus B)]$ (Sans table de karnaugh vu l'absence de l'adjacence voir table de K ci-dessous)

CD / AB	00	01	11	10
00	0	1	0	1
01	1	0	0	0
11	0	0	0	0
10	1	0	0	0

CD / AB	00	01	11	10
00	0	0	1	0
01	0	1	0	1
11	1	0	0	0
10	0	1	0	1

$O = M[A\,\bar{B}(C \oplus D) + \bar{A}B(C \oplus D) + AB\bar{C}\bar{D} + \bar{A}\bar{B}CD]$

$= M[(A \oplus B)(C \oplus D) + AB\bar{C}\bar{D} + \bar{A}\bar{B}CD]$ (Sans TK en l'absence d'adjacence des « 1 »)

$R = M[BCD + ACD + ABD + ABC]$ (Par TK)

$= M[CD(A + B) + AB(C + D)]$ (Algébrique)

CD / AB	00	01	11	10
00	0	0	0	0
01	0	0	1	0
11	0	1	1	1
10	0	0	1	0

3) Logigramme à base de NAND

$\bar{\bar{V}} = \overline{\overline{M\bar{A}\bar{B}\bar{C}\bar{D}}}$

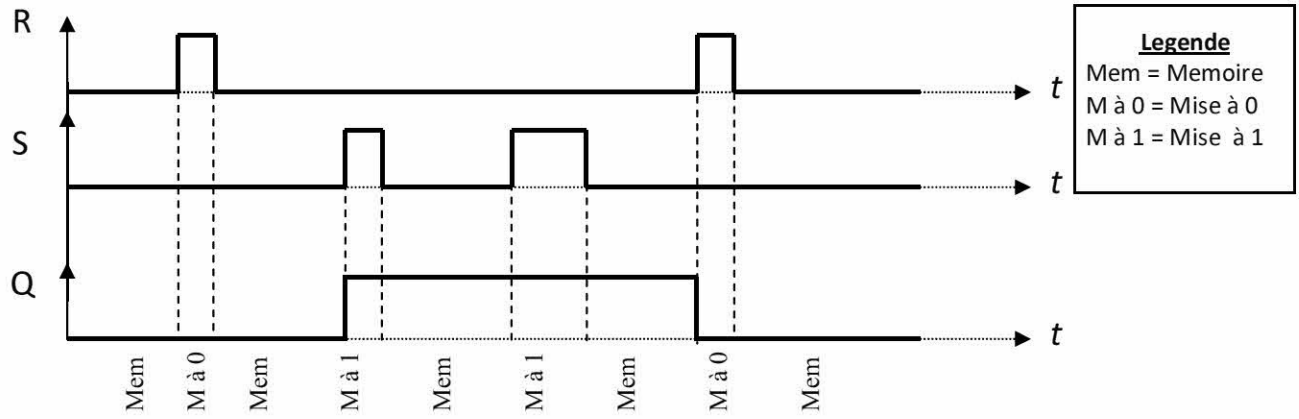
$\bar{\bar{J}} = \overline{\overline{M\bar{A}\,\bar{B}\bar{C}D + M\bar{A}\,\bar{B}C\bar{D} + MA\bar{B}\bar{C}\,\bar{D} + M\bar{A}B\bar{C}\,\bar{D}}} = \overline{\overline{M\bar{A}\,\bar{B}\bar{C}D}.\overline{M\bar{A}\,\bar{B}C\bar{D}}.\overline{MA\bar{B}\bar{C}\,\bar{D}}.\overline{M\bar{A}B\bar{C}\,\bar{D}}}$

Faire de même pour O et R

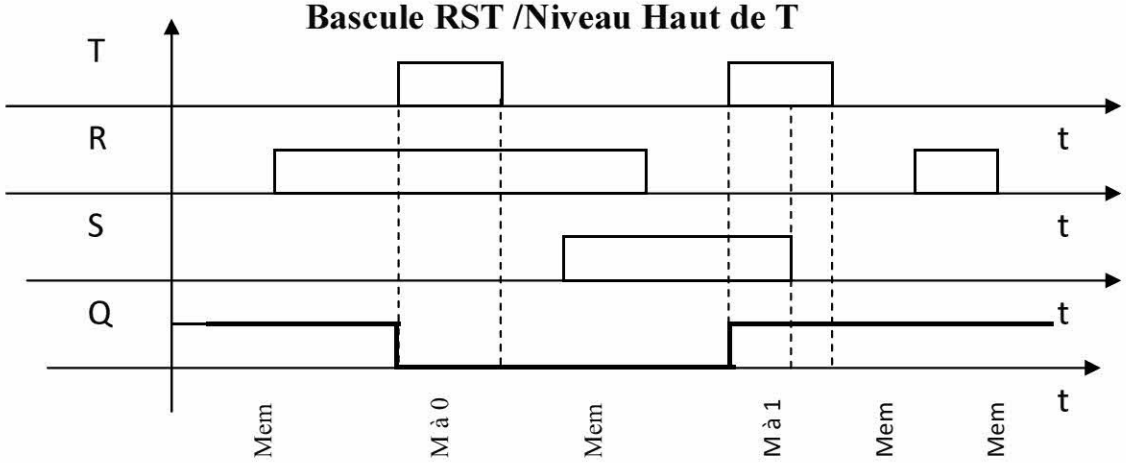
Chapitre 3

Exercice 3.1

Bascule RS

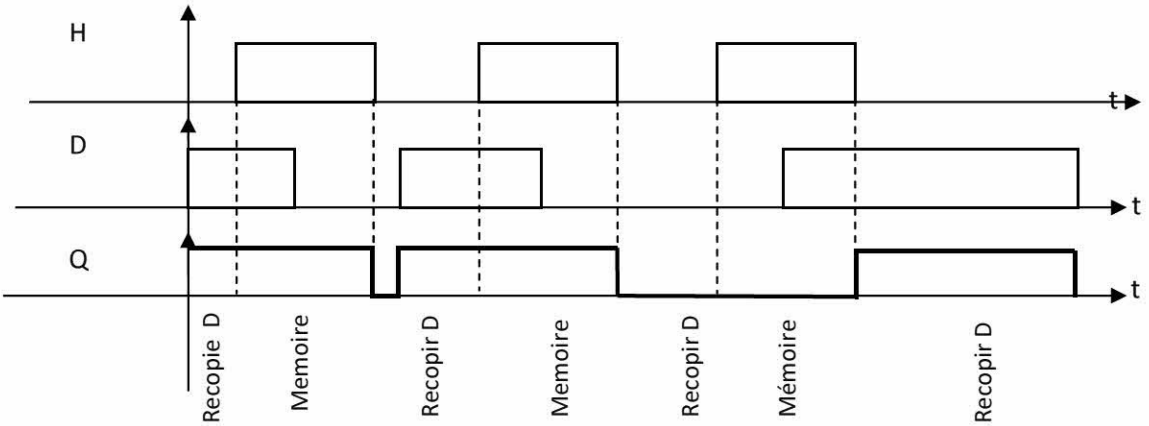


Bascule RST /Niveau Haut de T

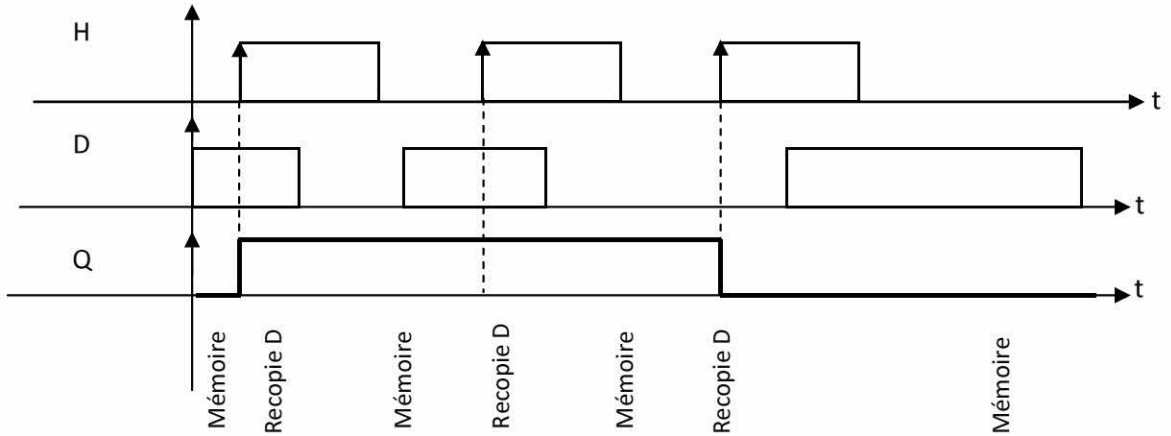


Exercice 3.2

Bascule D / Niveau Bas de H

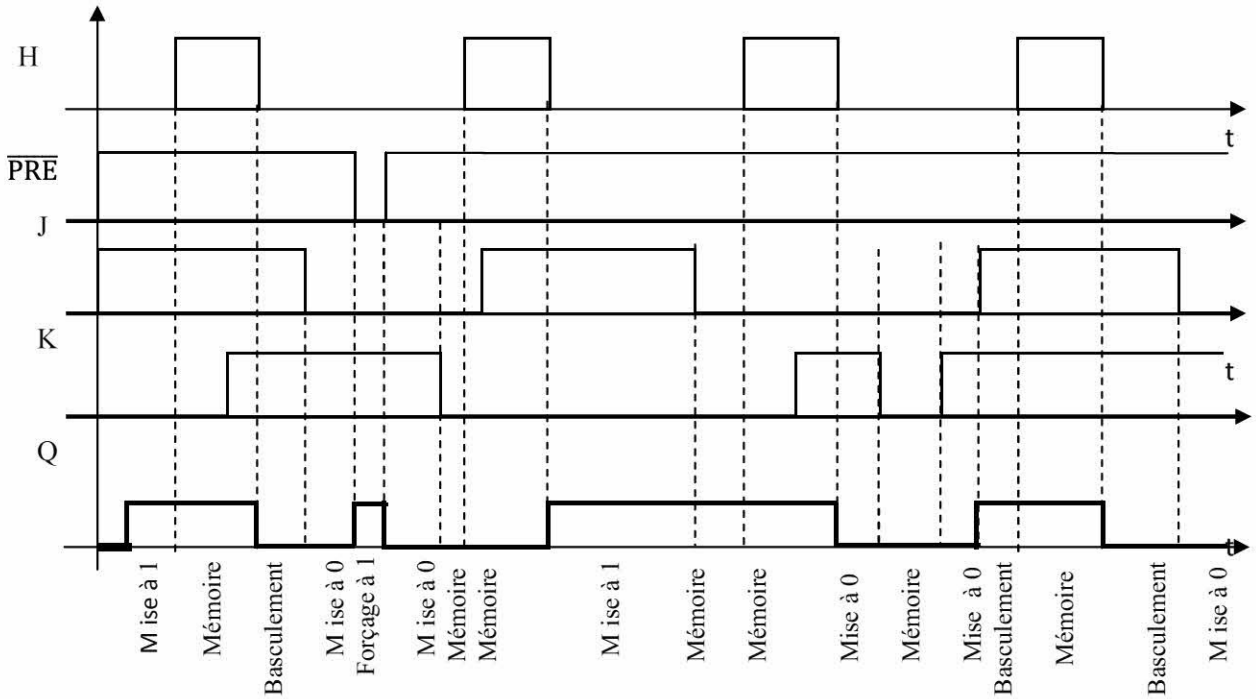


Bascule D / Front Montant de H

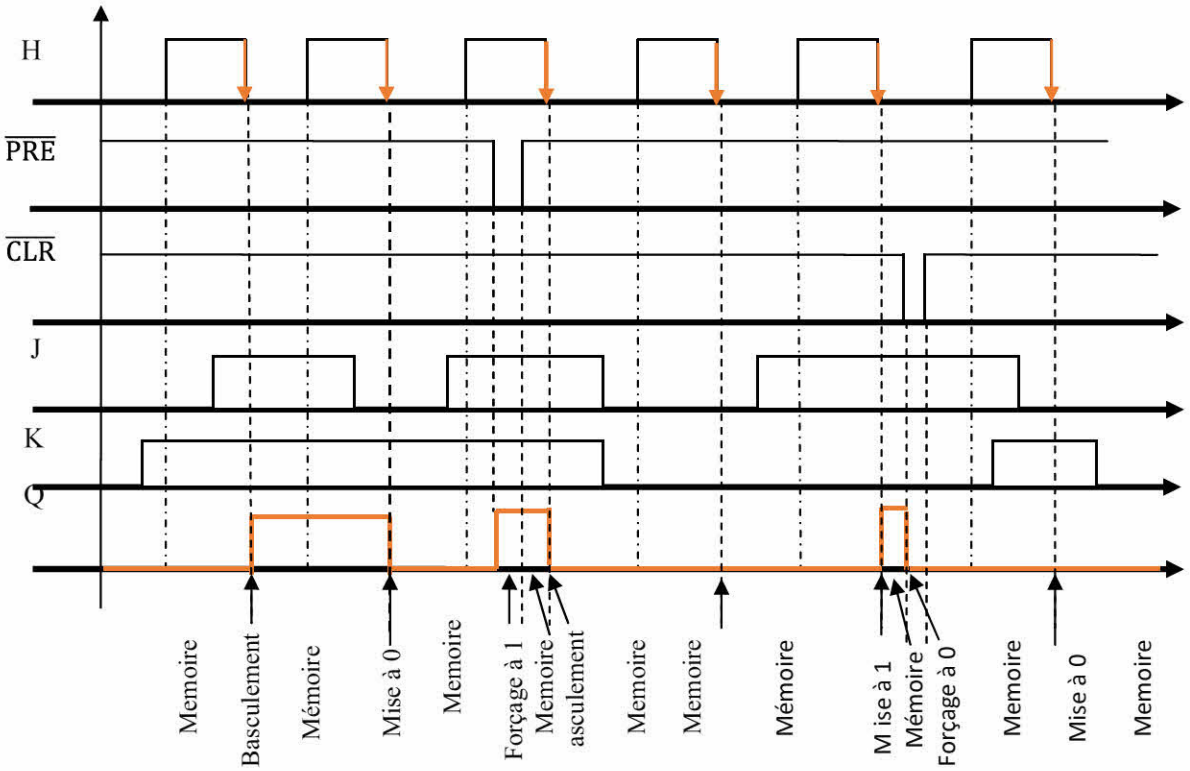


Exercice 3.3

Sur niveau bas de H



Sur front descendant de H

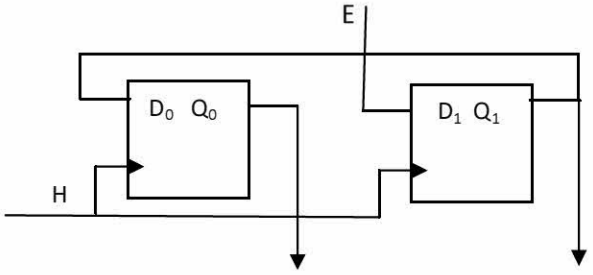


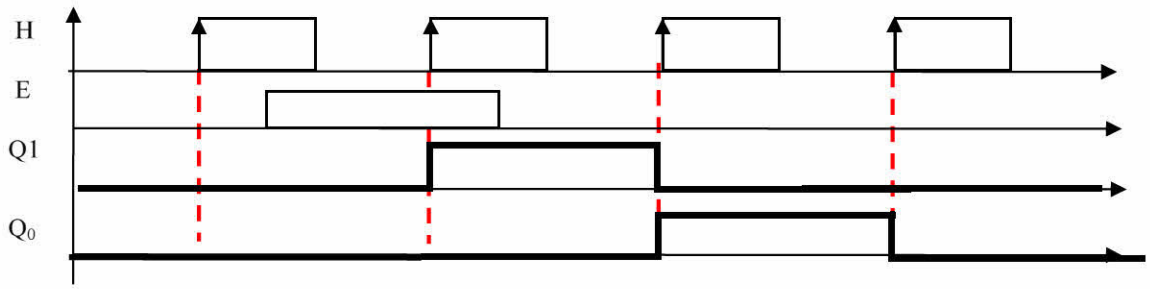
Partie 2 Les registres

Exercice 3.4

Soit le circuit de la figure 1 :

a- Compléter le chronogramme associe à ce circuit



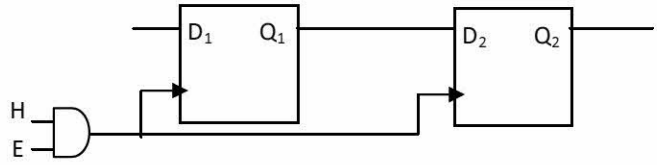


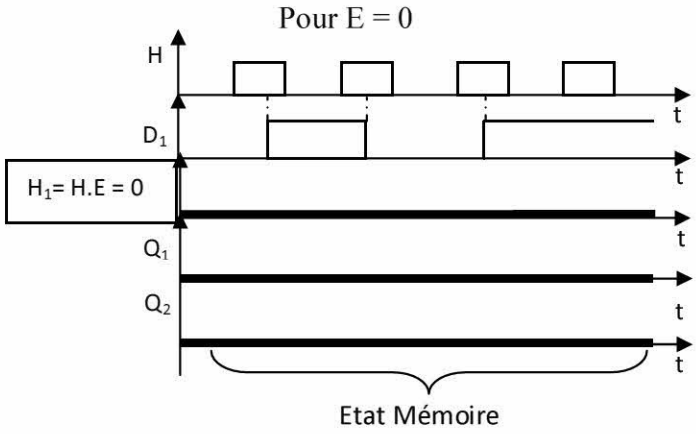
b- C'est un registre à décalage à gauche de 2 bits

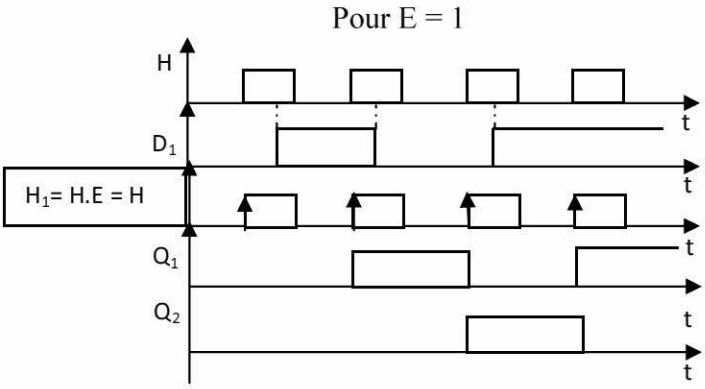
Exercice 3.5

Soit le circuit séquentiel de la figure 2:

1. Compléter le chronogramme correspondant.







2. Dans le cas E = 0, le circuit fonctionne comme registre de mémorisation.
   Dans le cas E = 1, le circuit fonctionne comme registre de décalage à droite

Partie 3 : Les Compteurs

Exercice 3. 6

1. Soit le montage de la figure 3.

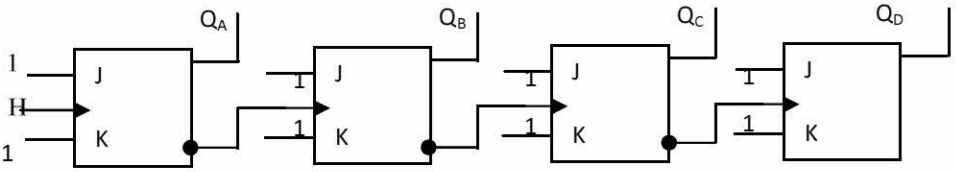


Figure 3

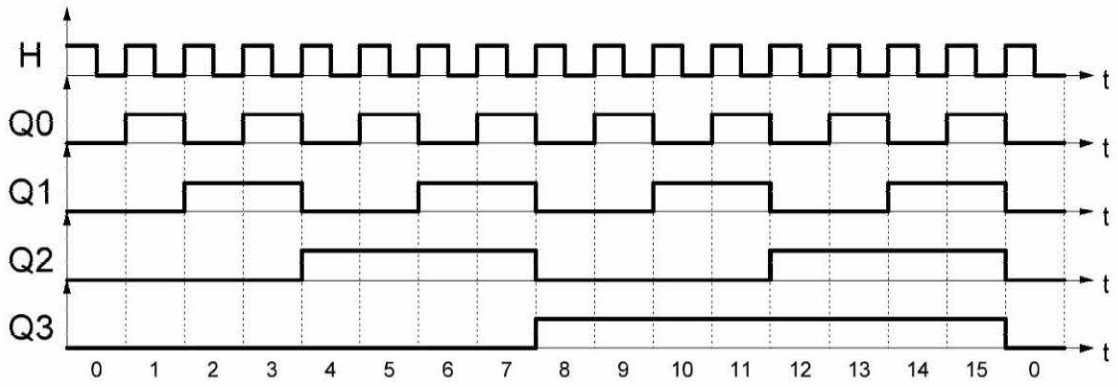
Les bascules JK sont synchronisées sur **front montant** et câblées en **basculement permanent** (**J** et **K** sont toujours a 1) :

• La sortie **Q0** bascule sur chaque front montant de **H** ;

• La sortie **Q1** bascule sur chaque front montant de **Q0** (donc chaque **front descendant** de **Q0**) ;

• La sortie **Q2** bascule sur chaque front montant de **Q1** (donc chaque **front descendant** de **Q1**) ;

• La sortie **Q3** bascule sur chaque front montant de **Q2** (donc chaque **front descendant** de **Q2**).



b. en déduire son rôle?

A chaque front d'horloge, la valeur présente sur les sorties est incrémentée de un. Ce montage est un **compteur asynchrone modulo 16**. Il compte de 0 à 15.

2. On modifie légèrement le montage de la figure 3 afin d'obtenir le montage de la figure 4. Que réalise le circuit de la figure 3 (justifier votre réponse).

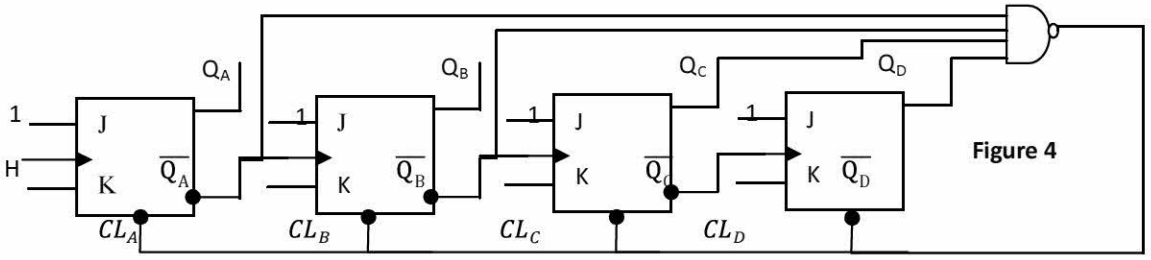


Figure 4

La porte NON-ET sert à détecter la valeur 12 et à la remplacer par la valeur 0.

Soit **M**, la sortie de la porte NON-ET. Pour rappel, la sortie d'une porte NON-ET est a 0 uniquement lorsque ses deux entrées sont a 1. **M** passera donc a 0 lorsque **Q2** et **Q3** seront a 1 en même temps.

Le passage de **M** a 0 aura pour effet de provoquer un *reset* sur le compteur et donc de le faire repartira 0.

Les sorties **Q2** et **Q3** passent a 1 pour la première fois sur la valeur 12. Le *reset* s'effectue donc au moment ou le compteur atteint la valeur 12. Cette valeur ne reste pas et est immédiatement remplacée par la valeur 0. **M** repasse alors à 1 et le compteur se remet à compter.

Ce montage est un compteur asynchrone modulo 12. Il compte de 0 à 11.

3. Soit le montage la figure 5.
    a. Compléter le chronogramme associé à cette figure.
    b. En déduire que fait ce circuit.

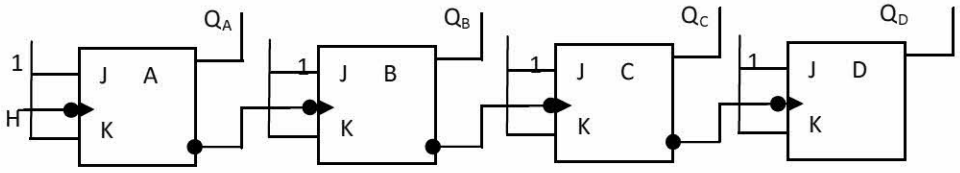


Figure 5

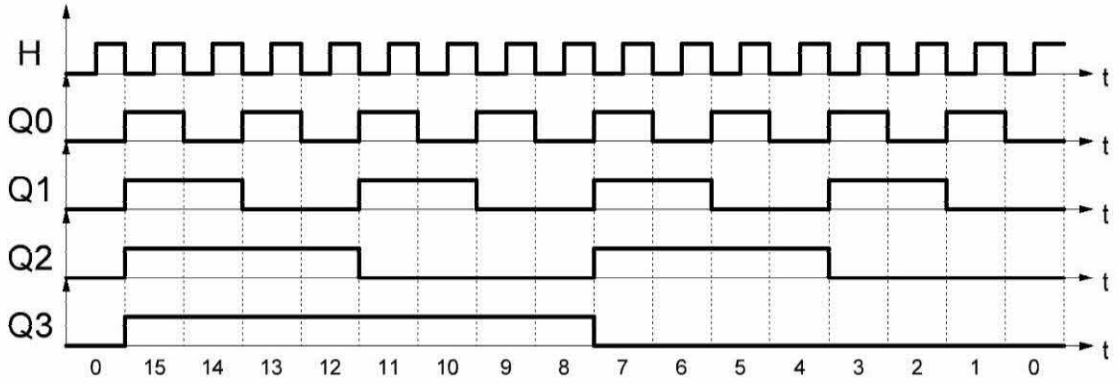
a. Les bascules JK sont synchronisées sur **front descendant** et câblées en **basculement permanent** (**J** et **K** sont toujours a 1) :

•La sortie Q0 bascule sur chaque front descendant de H ;

•La sortie **Q1** bascule sur chaque front descendant de **Q0** (donc chaque **front montant** de **Q0**)

•La sortie **Q2** bascule sur chaque front descendant de **Q1** (donc chaque **front montant** de **Q1**)

•La sortie **Q3** bascule sur chaque front descendant de **Q2** (donc chaque **front montant** de **Q2**)



b. A chaque front d'horloge, la valeur présente sur les sorties est décrémentée de un. Ce montage est un décompteur asynchrone modulo 16. Il décompte de 15 à 0.

4. Câbler le montage de la figure 6 afin d'obtenir un compteur asynchrone modulo 14.

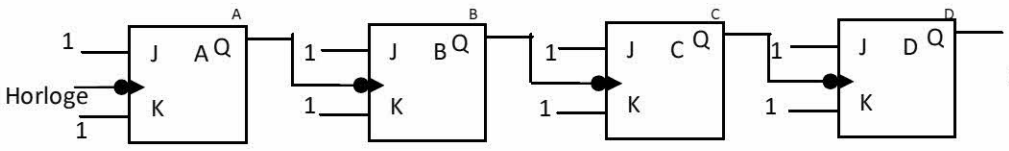
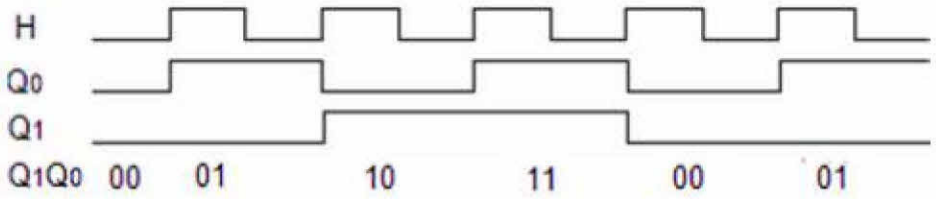


Figure 6

Exercice 3.7

Soit le circuit de la figure 7.

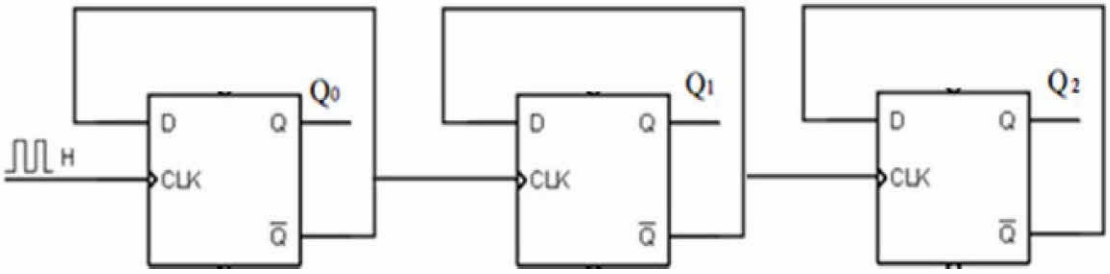
1. Compléter le chronogramme de la figure 8 associé à ce circuit.



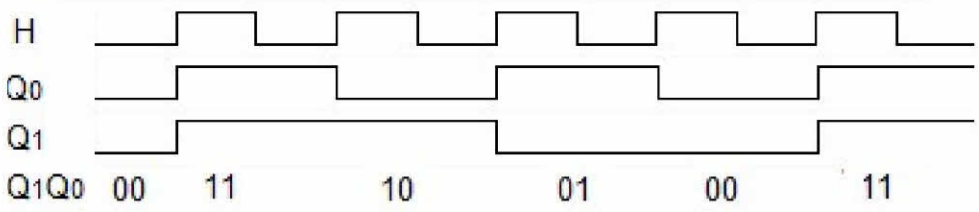
2. Quelle est la séquence des signaux de sortie obtenue ? et quelle est la fonction réalisée ?

   La séquence obtenue est : 0 – 1 – 2 – 3 – 0. On a un compteur asynchrone modulo 4.

3. Etudier et donner le schéma d'un compteur asynchrone modulo 8 avec des bascules D.
   Pour avoir un décompteur asynchrone modulo 8 ; il suffit d'ajouter une 3ème bascule D, comme suit :

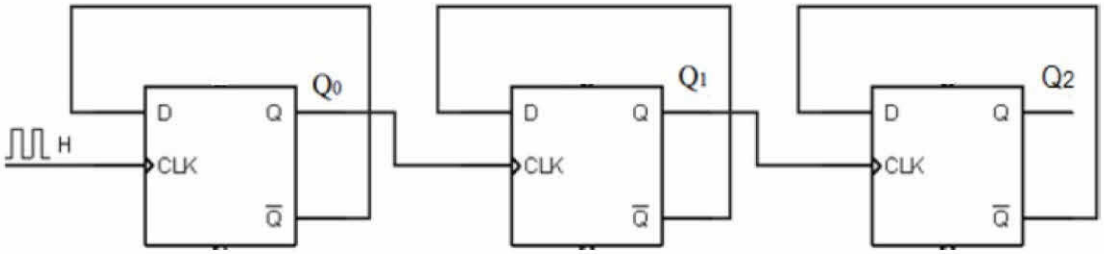


4. Soit le nouveau circuit de la figure 9, compléter le chronogramme associé a ce circuit (figure 10)



5. Quelle est la séquence des signaux de sortie obtenue ? et quelle est la fonction réalisée ?
   La séquence obtenue est : 0 – 3 – 2 – 1 – 0. On a un décompteur asynchrone modulo 4.
6. Etudier et donner le schéma d'un décompteur asynchrone modulo 8 avec des bascules D.

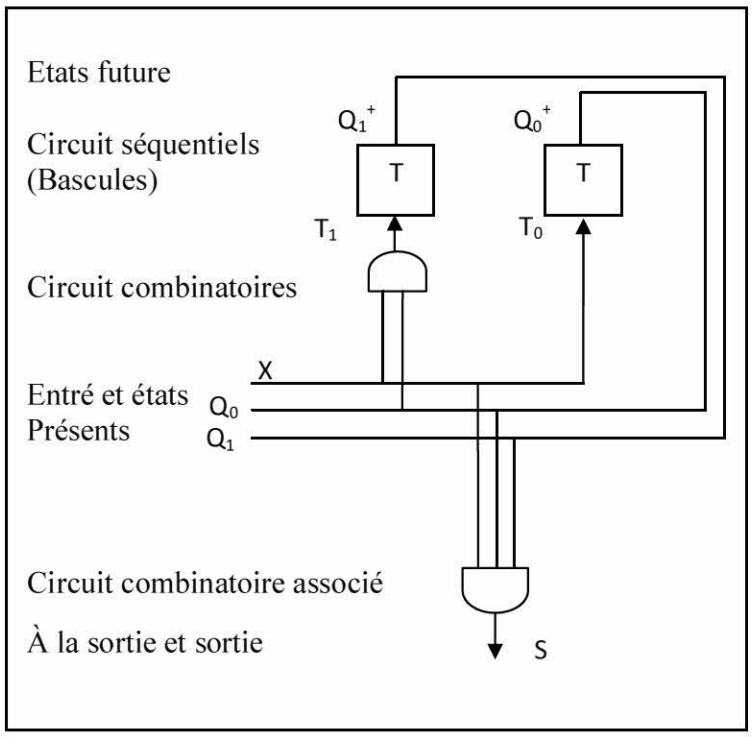
Pour avoir un décompteur asynchrone modulo 8 ; il suffit d'ajouter une 3ème bascule D, comme suit :



Chapitre 4

Exercice 4.1

1. Circuit sous sa forme normale



2. Table D'état

Etats Présents		Entrée	Bascules		Etats futurs		Sortie
Q_1	Q_0	X	T_1	T_0	Q_1^+	Q_0^+	S
0	0	0	0	0	0	0	0
0	0	1	0	1	0	1	0
0	1	0	0	0	0	1	0
0	1	1	1	1	1	0	0
1	0	0	0	0	1	0	0
1	0	1	0	1	1	1	0
1	1	0	0	0	1	1	0
1	1	1	1	1	0	0	1

3. Expressions des fonctions de transferts (T_0, T_1, Q_0^+, Q_1^+, S)

D'après le logigramme on peu déduire les expressions de :

$T_0 = X$, $T_1 = X.Q_0$, $S = X.Q_0.Q_1$

Les expressions de Q_0^+, Q_1^+ peuvent être déduites par table de karnaugh

Q_1 \ Q_0X	00	01	11	10
0	0	0	1	0
1	1	1	0	1

Q_1 \ Q_0X	00	01	11	10
0	0	1	0	1
1	0	1	0	1

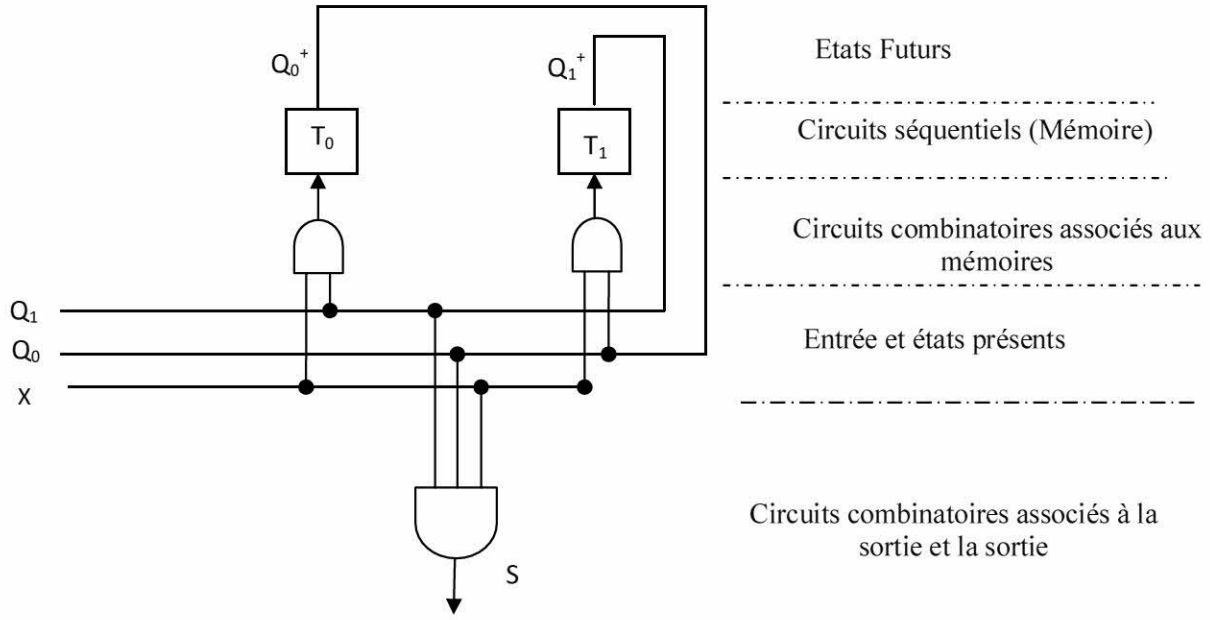
$Q_1^+ = Q_1 \oplus Q_0X = Q_1 \oplus T_1$

$Q_0^+ = Q_0 \oplus X = Q_0 \oplus T_0$

4. Puisque "S" est fonction de l'entrée et des états présents donc c'est un automate de **Mealy.**

Exercice 4.2

1. Circuit sous sa forme normale



2. Table d'états

Etats Présents		Entrée	Bascules		Etats futures		Sortie
Q_1	Q_0	X	T_1	T_0	Q_1^+	Q_0^+	S
0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	1	0	0	0	0	0	0
0	1	1	1	0	1	0	0
1	0	0	0	0	1	0	0
1	0	1	0	1	1	1	0
1	1	0	0	0	1	1	0
1	1	1	1	1	0	0	1

3. Fonctions de transferts :

$\mathbf{T_1} = Q_0.X$

$\mathbf{T_0} = Q_1.X$

$\mathbf{S} = Q_0.Q_1.X$

$Q_0^+ = Q_1(Q_0 \oplus X)$ (simplification par expression)

$Q_1^+ = Q_1 \oplus T_1$ (simplification par table de karnaugh et expression)

4. Puisque "S" est fonction de l'entrée et des états présents donc c'est un automate de **Mealy**.

Exercice 4.3

1. Graphe de transitions

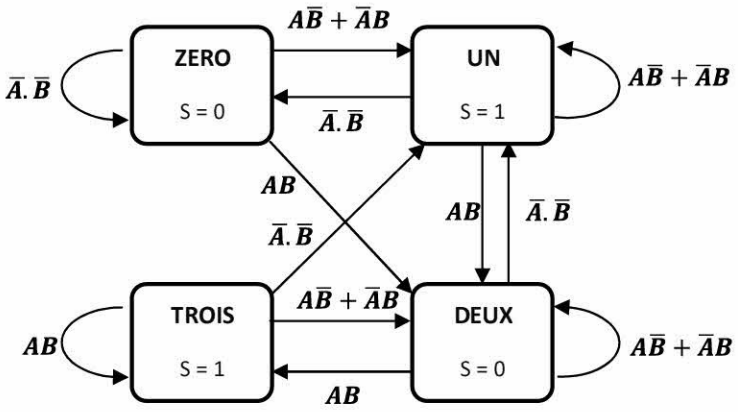


Figure 4

2. Table de transition

Etat	Q_1	Q_0	A	B	Q_1^+	Q_0^+	Etat Futur	R	S
zéro	0	0	0	0	0	0	Zéro	0	0
zéro	0	0	0	1	0	1	Un	0	0
zéro	0	0	1	0	0	1	Un	0	0
zéro	0	0	1	1	1	0	Deux	0	0
Un	0	1	0	0	0	0	Zéro	0	1
Un	0	1	0	1	0	1	Un	0	1
Un	0	1	1	0	0	1	Un	0	1
Un	0	1	1	1	1	0	Deux	0	1
Deux	1	0	0	0	0	1	Un	1	0
Deux	1	0	0	1	1	0	Deux	1	0
Deux	1	0	1	0	1	0	Deux	1	0
Deux	1	0	1	1	1	1	Trois	1	0
Trois	1	1	0	0	0	1	Un	1	1
Trois	1	1	0	1	1	0	Deux	1	1
Trois	1	1	1	0	1	0	Deux	1	1
Trois	1	1	1	1	1	1	Trois	1	1

3. $R = Q_1$ (de la table de transition)

$S = Q_0$ (de la table de transition)

$Q_1^+ = Q_1B + Q_1A + AB$

$Q_1^+ = Q_1(B + A) + AB$

$Q_0^+ = Q_1 \oplus A \oplus B$

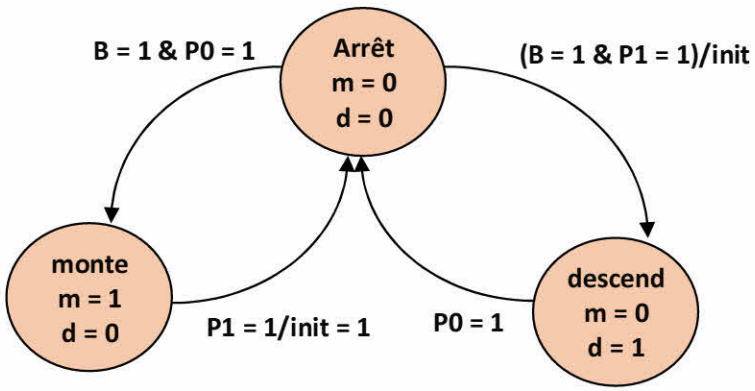
A B / Q1Q0	00	01	11	10
00	0	0	1	0
01	0	0	1	0
11	0	1	1	1
10	0	1	1	1

A B / Q1Q0	00	01	11	10
00	0	1	0	1
01	0	1	0	1
11	1	0	1	0
10	1	0	1	0

4. Il n'ya aucune transition manquante

Exercice 4.4

Diagramme de fonctionnement du monte charge



Conclusion Générale

Acquérir le module de « Structure Machine 2 » permettra à l'étudiant d'avoir des connaissances de base sur les circuits logiques et sera en mesure d'analyser et de synthétiser aussi bien les circuits combinatoires, comme les circuits : arithmétiques, de transcodage ou de multiplexage, que les circuits séquentiels comme les registres, les compteurs et les automates à états finis.

Le chapitre des circuits intégrés permettra à l'étudiant d'avoir une vu approfondie sur les circuits utilisés à la réalisation des systèmes numériques.

L'étudiant aura donc acquit les pre-requis nécessaires pour pouvoir étudier aisément les modules d'informatique tel que : l'architecture des ordinateurs, les réseaux, les systèmes d'exploitation,…etc.

Bibliographie

[1] John R. Gregg, « Ones and Zeros: Understanding Boolean Algebra, Digital Circuits, and the Logic of Sets », 1st Edition, Wiley & sons Inc. publishing, 1998.

[2] Bradford Henry Arnold, « Logic and Boolean Algebra », Dover publication, Inc., Mineola, 2011.

[3] Bencheriet Chemesse ennehar, « Codage et Représentation de l'Information », Polycopié de cours, édition OPUG, 2015.

[4] Paolo Zanella, « Architecture et technologie des ordinateurs », 5ème Edition, Dunod, 2013.

[5] Claude BRIE, Informatique industrielle : « Logique combinatoire et séquentielle; Méthodes, outils et réalisations », Ellipses, 2003.

[6] Jean Jacques Mercier, « Bit après bit numération, arithmétique binaire ,logique combinatoire », Ellipses, 2005.

[7] Jean Jacques Mercier, « Séquence après séquence logique séquentielle circuits asynchrones et synchrones », Ellipses, 2006.

[8] Jean Jacques Mercier, « Instruction après instruction: logique séquentielle circuits asynchrones et asynchrones et synchrones », Ellipses, 2008.

[9] Phillipe Darch, « Architecture des Ordinateurs : Logique booléenne et implémentation Technologique ». Edition VUIBERT , 2004.

[10] Yliès Falcone, Jean-Claude Fernandez, « Automates à états finis et langages réguliers: Rappels des notions essentielles et plus de 170 exercices corrigés », Dunod, 2020.

[11] Michel Marchand, « Outils mathématiques pour l'informaticien: Mathématiques » DeBoeck, 2005.

[12] https://electronique-et-informatique.fr/Electronique-et Informatique/Digit/Data_book_CMOS.php (date dernière consultation 01-01-2022).

[13] Laurent Pichon, « Bases de l'électronique analogique - Du composant au circuit intégré. Cours et exercices corrigés », Ellipses, 2015.

[14] Sedra, Smith, « Circuits microélectronique », DeBoeck, 2016.

NB : Retrouver les ouvrages disponibles dans la bibliothèque à traves le lien : http://www.univ-guelma.dz/recherche-bibliographique