LOGIQUE COMBINATOIRE ET SEQUENTIELLE

Djamal Gozim, Kamel Guesmi

RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université ZianeAchor de Djelfa

Faculté de Science et de la Technologie

Département de Génie Electrique

LOGIQUE COMBINATOIRE ET SEQUENTIELLE

Support de cours pour les étudiants en Licence de la filière génie électrique

Dr. Djamal GOZIM

Pr. Kamel GUESMI

Ver. 1.0.0

Table des matières

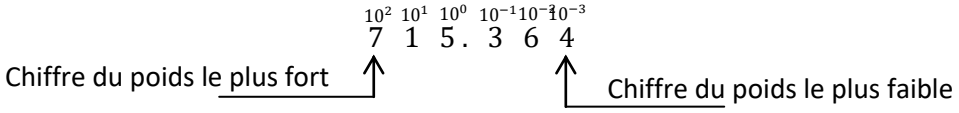
1.1 Définition

Les systèmes de numérotation les plus utilisées sont : décimal, Binaire, Octal et Hexadécimal

1.2 Système Décimal

Ce système est composé par des chiffres (symboles) de zéro (0) à neuf (9).

Exemple



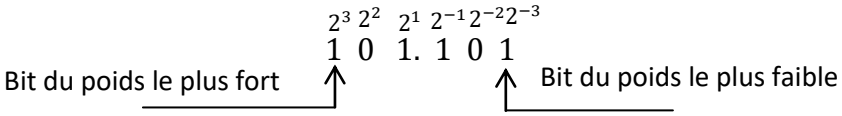
En peut écrire ce numéro à la base Décimal sous la forme :

$$715.364 = 7 * 10^2 + 1 * 10^1 + 5 * 10^0 + 3 * 10^{-1} + 6 * 10^{-2} + 4 * 10^{-3}$$

Remarque : Avec nChiffre, on peut compter jusqu'à 10^n nombres différents

1.3 Système Binaire : Ce système est composé de deux chiffres : zéro (0) et un (1)

Exemple



On peut écrire:

$$101.101 = 1\times2^2 + 0\times2^1 + 1\times2^0 + 1\times2^{-1} + 0\times2^{-2} + 1\times2^{-3}$$

$$= 4 + 0 + 1 + 0.5 + 0 + 0.125$$

Donc

$(101.101)_2 = (5.625)_{10}$

Avec n bits on peut compter 2^n nombres différents.

- **Conversion Binaire – Décimal**

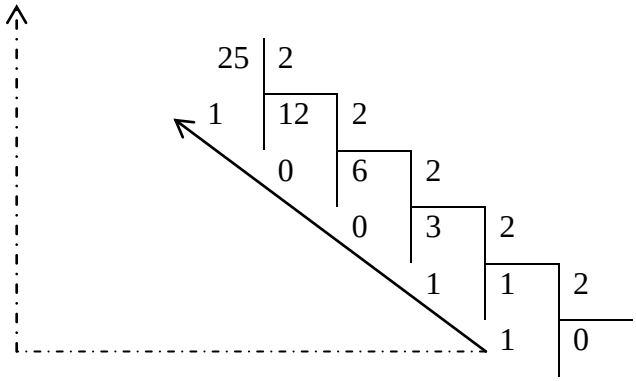
Pour passer du système binaire au décimal, il suffit d'additionner les poids correspondants aux '1' du nombre binaire:

$$(1101)_2 = 1\times2^3 + 1\times2^2 + 0\times2^1 + 1\times2^0 = 8 + 4 + 0 + 1 = 13_{10}$$

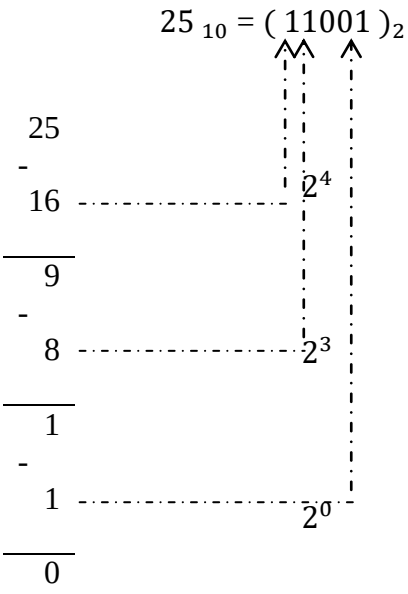
Pour convertir un nombre décimal au binaire il y a deux méthodes:

a) Division successives par '2'

$25_{10} = (11001)_2$



b) Soustraction successives des poids binaires



Exemple (1)

Convertir en nombres binaires les décimaux suivants: 127 et 1024.

Solution

$(127)_{10} = (1111111)_2 = 2^7 - 1$

$(1024)_{10} = (100\ 0000\ 0000)_2 = 2^{10}$

Exemple (2)

Quelle est la plus grande valeur décimale qui peut être représenté par un nombre binaire de 8 bits (1octes) et de 16 bits.

Solution

La plus grande valeur est

$2^8 - 1 = 255$

$2^{16} - 1 = 65535 = 2^6 \times 2^{10}$ (64 ko).

1.4 Systèmes Octal et Hexadécimal

On peut résumer le comptage aux systèmes octal de base 8 (2^3) et le système hexadécimal de base 16 (2^4) dans le tableau suivant :

Décimal	Binaire	Octal	Hexadécimal
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10
17	10001	21	11

- **Conversion Octal- Binaire**

Pour convertir un nombre octal en binaire, on converti chaque chiffre séparément en un nombre binaire de trois bits.

Exemple (1)

Pour convertir 251_8 en binaire, on procède suit

2	5	1	Décimal
010	101	001	Binaire

Donc $251_8 = (\ 010101001\ )_{\ 2}$

Exemple (2)

Convertir 291_8 en base binaire.

- 291 n'est pas un nombre octal.

- **Conversion Binaire-Octal**

Pour convertir un nombre binaire en octal, il suffit de convertir chaque 3 bits du nombre binaire en octal en commençant par le bit du poids le plus faible

100	011	001	001	Binaire
4	3	1	1	Octal

- **Conversion Hexadécimal- Binaire**

De la même façon que pour un nombre octal, pour convertir un nombre hexadécimal en binaire, on convertir chaque chiffre séparément en un nombre binaire de quatre bits.

Exemple

Pour convertir $\mathrm{E1A3E9}_{16}$ en binaire, on procède comme suit:

E	1	A	3	E	9	Hexadécimal
1110	0001	1010	0011	1110	1001	Binaire

Donc:

$\mathrm{E1A3E9}_{16} = 111000011010001111101001_2$

- **Conversion Binaire-Hexadécimal:**

Pour convertir un nombre binaire en hexadécimal, il suffit de convertir chaque quatre bits de nombre binaire en hexadécimal en commençant par le bit du poids le plus faible

0100	0011	0001	0001	Binaire
4	3	1	1	Hexadécimal

Pourquoi le codage hexadécimal

- Les ordinateurs et les circuits digitaux utilisent les nombres binaires pour représenter les données, les adresses mémoire et E/S, les instructions, les indicateurs d'état,….
- Le code hexadécimal permet de simplifier et de compacter les nombres binaires.
- La conversion est très facile entre les deux codes.

1.5Codage BCD (Binary Coded Decimal):

Le système BCD est utilisé pour faciliter la conversion binaire – décimal qui devient très difficile pour les grands nombres. Dans ce système, chaque chiffre de nombre décimal est représenté par son équivalent en binaire de quatre bits. Ce code est utilisé dans les appareils digitaux qui utilisent les nombres décimaux en entrée ou en sortie comme les voltmètres digitaux, les horloges numériques et les calculatrices électroniques….

Exemple (1)

Pour coder $(251)_{10}$ en BCD, on procède comme suit:

2	5	1
0010	0101	0001

Donc $(251)_{10} = (001001010001)_{2}$

Exemple (2)

$137_{10} = (10001001)_{2}$

$137_{10} = (000100110111)_{BCD}$

1.6 Codage Alphanumérique

En plus des chiffres, un ordinateur est capable de reconnaître les lettres et les caractères spéciaux en utilisant un codage alphanumérique.

ASCII (American Standard Code for Information Interchange)

L'ASCII est le code le plus utilisé par les ordinateurs pour la communication avec les périphériques entrée/sortie (clavier, écran, imprimantes...)

1.7 Opérations arithmétiques

1.7.a Addition Binaire

On additionne le chiffre de même poids en commençant par le mois significatif et rapportant la retenue qui doit être rajoutée à l'addition des deux prochains chiffres.

```
   5.625          101.101
 +              +
   1.625          001.101
 -------        -------
   7.250          111.010
```

1.7.b Soustraction Binaire

La plupart des micro-ordinateurs effectuent la soustraction à l'aide d'une addition en utilisant le complément à 2 (remplacer les 1 par 0 et les 0 par 1, en suite ajouter un 1). Le nombre à soustraire est remplacé par son complément à 2, et après l'addition on néglige la retenue.

```
   1100        11          1100
 -              -   ⇔    +
   1001        9           0111   Complément à 9 plus1
   ----        --          ----
   0011        3          1 0011
                          ↑
                Retenue à négliger
```

1.7.c Nombres signés

Dans le code machine les nombres sont représentés par un ensemble de bits qui ne peuvent prendre que 0 ou 1, les ordinateurs manipulent les nombres négatives autant que les nombres positives, un bit dans le codage binaire des nombre est réservé pour le signe (0 pour un nombre positive et 1 pour un nombre négative).

Dans un nombre binaire le bit de signe est le bit le plus significatif.Avec un système utilisent le complément à deux, le nombre négatif binaire équivalent est seulement le complément à deux.

$00110100_2 = +52_{10}$
11001011 (inverser les bits)
$+ \quad 1$
$11001100 = -52_{10}$ (complément à deux)

Négation

La négation et l'opération de convertir un nombre positif en son nombre négatif équivalent.

$01001_2 = +9_{10}$
10110 (inverser les bits)
$+ \quad 1$
$10111 = -9_{10}$ (complément à deux)

Avec le système de complément à 2, on peut vérifier par exemple que dans un système de 5 bits :

$-(-9_{10}) = +9_{10}$

$10111_2 = -9_{10}$
01000 (inverser les bits)
$+ \quad 1$
$01001 = +9_{10}$ (complément à deux)

Exemple

Représenter les nombres décimaux signés suivants avec des nombres signés de 5 bits (compris le bit de signe) :-6, -1 et -8.

$-6_{10} = (11010)_2$; $-1_{10} = (11111)_2$; $-8_{10} = (11000)_2$

1.8 Exercice

Exercice (1)

a) Citer les chiffres de base des systèmes numérique suivants

-Système Binaire, Octal, Hexadécimal

- Système de Base 5

- Système de Base 7

b) Classer les nombres suivants aux bases Hexadécimal (16), Octal 8 et Binaire 2

100111, 12113, 7632, 11A

Exercice (2)

a) Convertir les nombres suivants pat deux méthodes(Divisons successive et soustraction successive

- $(146)_{10}=$ ($)_2$
- $(236)_{10}=$ ($)_8$
- $(529)_{10}=$ ($)_{16}$
- $(56)_{10}=$ ($)_9$
- $(126)_{10}=$ ($)_5$

b) Faire les conversions suivantes

c) $(01110011)_2=$ ($)_{10}=($ $)_8=($ $)_{16}$

d) $(342)_5=$ ($)_{10}$

e) $(1267)_9=$ ($)_{10}$

f) $(1017)_8=$ ($)_{10}=($ $)_2$

g) $(A001)_{16}=$ ($)_{10}=($ $)_2$

h) $(170CDF)_{16}=$ ($)_{10}=($ $)_2$

Exercice (3)

- Convertir les nombres suivants:

a. $(7852)_{10}$ en base hexadécimal puis en binaire.

b. $(1101001011)_2$ en hexadécimal puis en décimal

c. $(2EA)_{16}$ en binaire puis en décimal.

d. $(5F)_{16}$ =( )$_{10}$ = ( )$_2$ = ( )$_8$

Exercice (4)

Faire la soustraction par complément à 2 :

$$\begin{array}{rr} & 111011 \\ - & 11101 \\ \hline & \end{array}$$

Effectuer les opérations suivantes en complément à 2 sur 8 bits. Vérifier les résultats et indiquer les éventuels débordements. Comment peut on détecter que le résultat est faux ?

a. 125 – 26

b. 105 + 35

c. 40-60

d. -38-96

Exercice (5)

On représente des entiers signés sur 16 bits.

1. Quel est le plus grand entier positif que l'on puisse écrire? Quel est le plus petit entier négatif que l'on puisse écrire?
2. Ecrire, en valeur absolue, les entiers précédents en base hexadécimal et décimal.
3. Donner les compléments à 1 et 2 de l'entier le plus grand.

2.1 Définition

Dans un système logique (les entrées et sorties ne peuvent prendre que (0) ou(1) comme valeur) combinatoire, les sorties ne sont fonctions que des entrées.

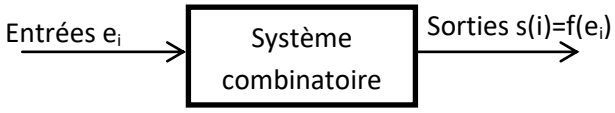


Schéma d'un système combinatoire

Exemple

Soit le schéma électrique suivant

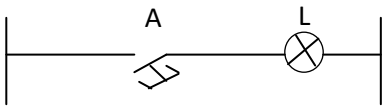


Schéma électrique

Table de vérité

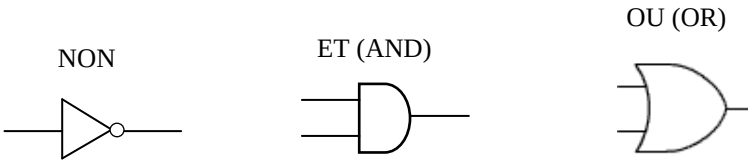
A	L
0	0
1	1
0	0

Nombre d'états de la sortie dépend de nombre des entrées

- Si nombre des entrées 1 → nombre d'états de la sortie est $2^1=2$
- Si nombre des entrées 2 → nombre d'états de la sortie est $2^2=4$
- Si nombre des entrées 3 → nombre d'états de la sortie est $2^3=8$

2.2 Fonction logique

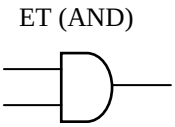
L'outil mathématique qui permet de décrire les systèmes combinatoires est l'**Algèbre de Bool**. Par la combinaison des trois fonctions de base que sont le NON, OU, ET, on va pouvoir décrire chacune des sorties en fonction des entrées.



A) FonctionET (AND)

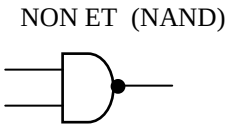
La fonction ET est obtenue par la multiplication des Entrées F= A • B . La table de vérité et le symbole associés à cette fonction sont :

A	B	F= A.B
0	0	0
0	1	0
1	0	0
1	1	1



La fonction NON ET est obtenue en complémentant la fonction ET $F = \overline{A.B}$. La table de vérité et le symbole associés à cette fonction sont :

A	B	F= A.B	$F = \overline{A.B}$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

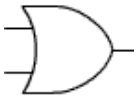


B) Fonction OU(OR)

La fonction OU est obtenue par la somme des entrées du système F = A + B. La table de vérité et le symbole associés à cette fonction sont :

A	B	F= A+B
0	0	0
0	1	1
1	0	1
1	1	1

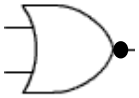
OU (OR)



La fonction NON OU est obtenue en complémentant la fonction OU $F = \overline{A + B}$. La table de vérité et le symbole associés à cette fonction sont :

A	B	F= A+B	$F = \overline{A + B}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

NON OU (NOR)



C) Fonction OU EXCLUSIF (XOR)

La fonction OU EXCLUSIF ne vaut 1 que si les deux entrées sont différentes. Elles'écrit $F = A \oplus B = A \bullet B + A \bullet B$. La table de vérité et le symbole associés à cette fonction sont :

A	B	$F = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

OU EXCLUSIFXOR

2.3 Synthèse d'un circuit combinatoire

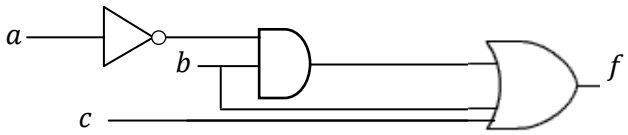
L'objectif est de générer logigramme du circuit à partir de la fonction logique correspondante

Exemple

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

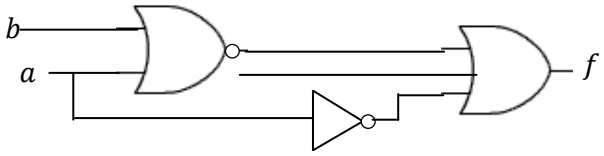
Il y a deux méthodes pour représenter cette fonction logique par logigramme

- *Représentationpar la porte logique* ***Non*** *et la porte logique* ***ET***



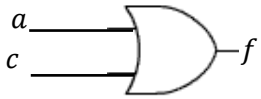
- *Représentation par la porte logique Non et la porte logique OU*

$$f(a,b,c) = a + \overline{\overline{a\bar{b}}} + c = a + \left(\overline{\bar{a} + b}\right) + c$$



On peut aussi réduire ou simplifier l'équation précédente en utilisant les caractéristiques d'algèbre de **BOOL** sous la forme :

$$f(a,b) = a + a\bar{b} + c = a + \left(1 + \bar{b}\right) + c = a + c$$



Cette solution est la plus économique en termes de portes logiques utilisées. Donc, avant d'effectuer la synthèse d'un circuit combinatoire, on doit d'abord poses les questions suivantes:

- Quelle représentation de la fonction choisir
- Quelles portes logiques utilisées pour la représentation schématique
- Est-ce que la fonction est simplifiée.

2.4 Détermination d'une équation logique

Soit la table de vérité qui contient trois entrées a,b et c et deux sortie x et y

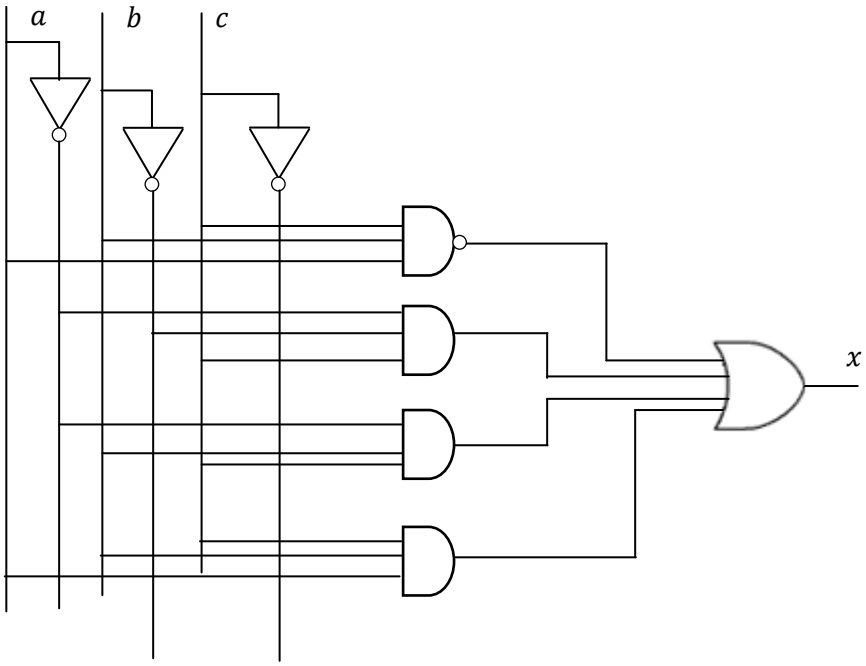
a	b	c	X	y
0	0	0	1	1
0	0	1	1	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	1
1	1	0	0	0
1	1	1	1	0

- *Détermination de l'équation logique par la méthode AND -OR*

On fait la somme de la multiplication des entrées ayant à la sortie 1.

$$x = \overline{abc} + \bar{a}\bar{b}c + \bar{a}bc + abc$$

Logigramme



- *Détermination de l'équation logique par la méthode OR-AND*

On fait la multiplication de la somme des entrées ayant à la sortie 0.

$$x = \overline{abc} + \left(a + \bar{b} + c\right)(\bar{a} + b + c)(\bar{a} + b + \bar{c})\left(\bar{a} + \bar{b} + c\right)$$

2.5 Simplification d'une fonction logique

On peut simplifier les équations logiques par deux méthodes :

- ***Simplification par les lois d'algèbre de BOOL***
- ***Simplification par table de KARNAUGH***

a) Table de KARNAUGH pour un système de deux entrées

Exemple

Soit la table de vérité suivant

a	b	S
0	0	1
0	1	0
1	0	0
1	1	1

***Table de* KARNAUGH** Correspond

a / b	0	1
0	1	0
1	0	1

Equation logique

$$x = \bar{a}\bar{b} + ab$$

b) Table de KARNAUGH pour un système de trois entrées

Table de vérité

a	b	c	S
0	0	0	1
1	0	0	0
0	1	0	0
1	1	0	0
0	0	1	1
1	0	1	1
0	1	1	0
1	1	1	1

Table de KARNAUGH

c \ *ba*	*00*	*01*	*11*	*10*
0	*1*	*0*	*0*	*0*
1	*1*	*1*	*1*	*0*

Equation logique

$$x = \bar{a}\bar{b} + bc$$

c) Table de KARNAUGH pour quatre entrées

Exemple(1)

dc \ *ab*	00	01	11	10
00	0	0	1	1
01	1	1	0	0
11	1	1	0	0
10	1	0	0	1

Equation logique $x = \bar{a}d + \bar{b}c\bar{d} + \bar{c}\bar{d}a$

Exemple(2)

$d\ c$ \ $a\ b$	00	01	11	10
00	0	1	1	0
01	1	1	1	1
11	0	0	0	0
10	0	1	1	0

Equation logique $x = b\bar{d} + \bar{c}d$

Exemple(3)

$d\ c$ \ $a\ b$	00	01	11	10
00	1	0	0	1
01	0	1	1	0
11	0	1	1	0
10	1	0	0	1

Equation logique $x = \bar{b}\bar{d} + bd$

Exemple(4)

$d\ c$ \ $a\ b$	00	01	11	10
00	1	1	0	0
01	1	1	0	0
11	1	1	0	0
10	1	1	0	0

Equation logique $x = \bar{a}$

Exemple(5)

d c \ a b	00	01	11	10
00	1	1	1	1
01	1	1	1	1
11	0	0	0	0
10	0	0	0	0

Equation logique $x = \bar{d}$

2.6 Circuits logiques

En pratique il n'est pas nécessaire de réaliser des fonctions logiques complexes en utilisant les portes logiques de base. On peut réaliser des circuits logiques intégrés où il y a plusieurs portes logiques intégrées. Nous verrons simplement ici le multiplexeur, l'encodeur le décodeur, Additionneur, soustracteur.

2.6.1 Additionneur

L'addition de deux nombres binaires s'accomplit en additionnant les bits les moins forts en allant vers ceux des rangs les plus forts.

a) **Demi-additionneur** est l'addition de deux mots d'un bit

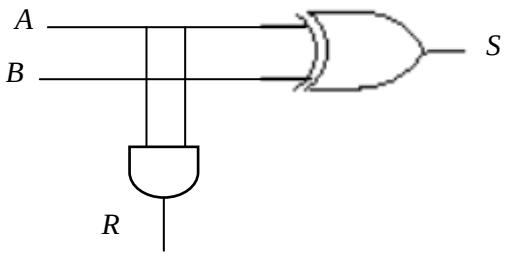
Table de vérité

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

S : Somme

R : Retenue

Logigramme :



b) Additionneur complet

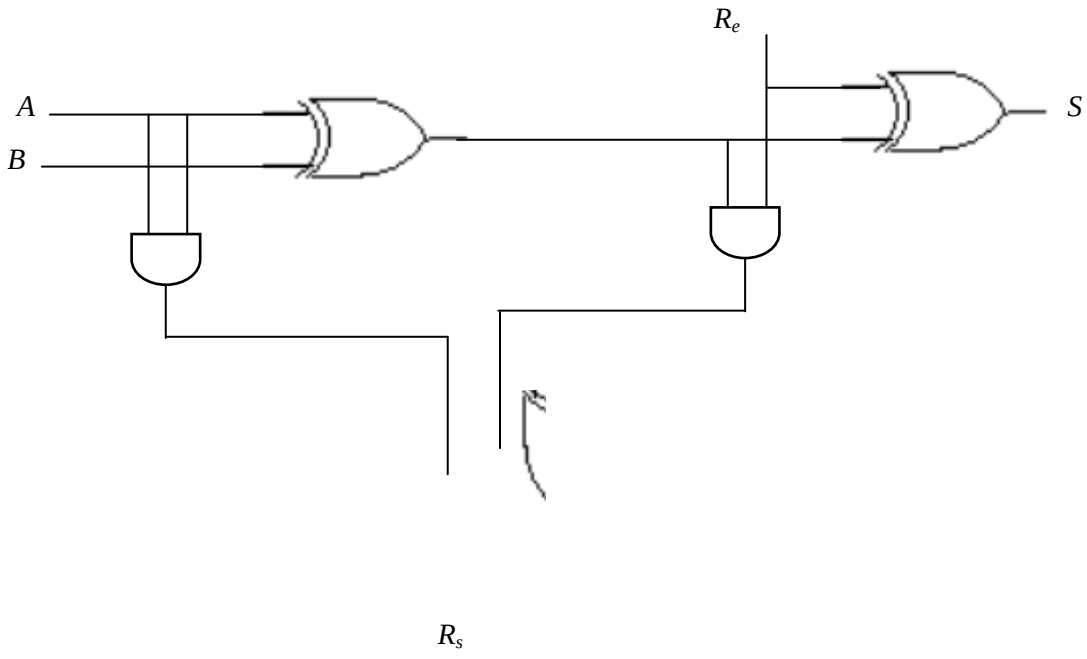
Table de vérité

A	B	R_e	R_s
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

R_e : Retenue en entrée

R_s : Retenue en sortie

Logigramme



2.6.2 Soustracteur

Le même principe que l'additionneur, le soustracteur fait la soustractiondeux nombres binaires

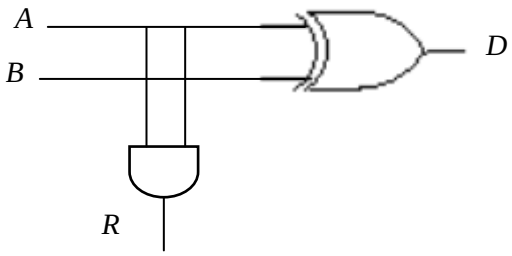
Table de vérité

A	B	R	D
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

R : Retenue

D : Différence

Logigramme :



2.6.3 Multiplexeur (Mux)

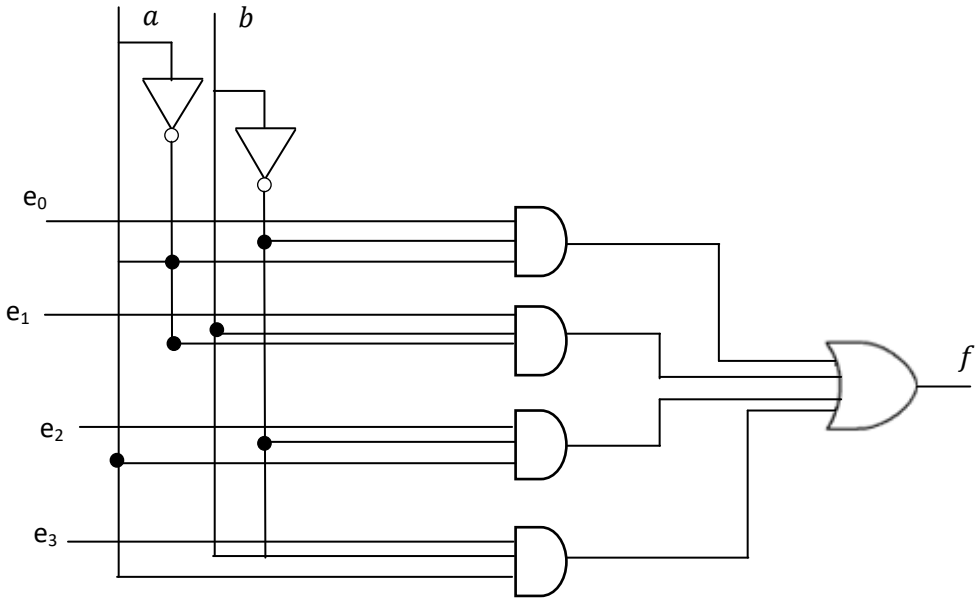
C'est un système combinatoire qui fait la fonction à n variables qui correspond aux n lignes de sélection, un multiplexeur est composé de :

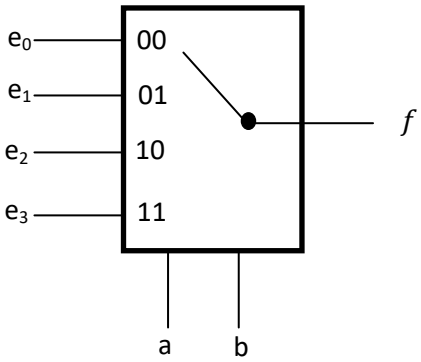
- 2^n entrées
- Une seule sortie
- n lignes de sélection

Exemple

Multiplexeur à deux variables

$f(a,b) = e_0\,\bar{a}\,\bar{b} + e_1\,\bar{a}\,b + e_2\,a\bar{b} + e_3\,ab$





- a et b sont appelées ligne de commande
- e_0, e_1, e_2, $e_3$sont appelées ligne de données
- f est le résultat

Les lignes d commande déterminent quelle entrée se retrouve en sortie ; de ce faite, on dira qu'un multiplexeur est un sélecteur de donnés.

2.6.4 Encodeur prioritaire

C'est un circuit à m entrées et n sorties. Les sorties délivrent le code de l'entrée active si il n'y en a qu'une ou de l'entrée prioritaire si il y en a plusieurs. Si le code est le code binaire standard alors on a m=2n et l'encodeur est dit binaire. Il existe bien entendu des encodeurs pour les codes BCD, GRAY, ASCII ... Dans le cas du code ASCII l'encodeur est utilisé pour coder les touches d'un clavier, c'est à dire pour générer le code ASCII associé au caractère ou à la fonction de la touche enfoncée. Le fonctionnement d'un encodeur prioritaire à quatre entrées est décrit par la table de vérité suivante. Les entrées sont actives au niveau haut (1), lorsque plusieurs entrées Eisont actives, seul le code correspondant à l'entrée d'indice le plus élevé est présent sur les sorties a1a0. Dans cet exemple l'entrée E_3 a la plus forte priorité.

E_0	E_0	E_0	E_0	a_1	a_0
1	0	0	0	0	0
×	1	0	0	0	1
×	1	1	0	1	0
×	×	×	1	1	1

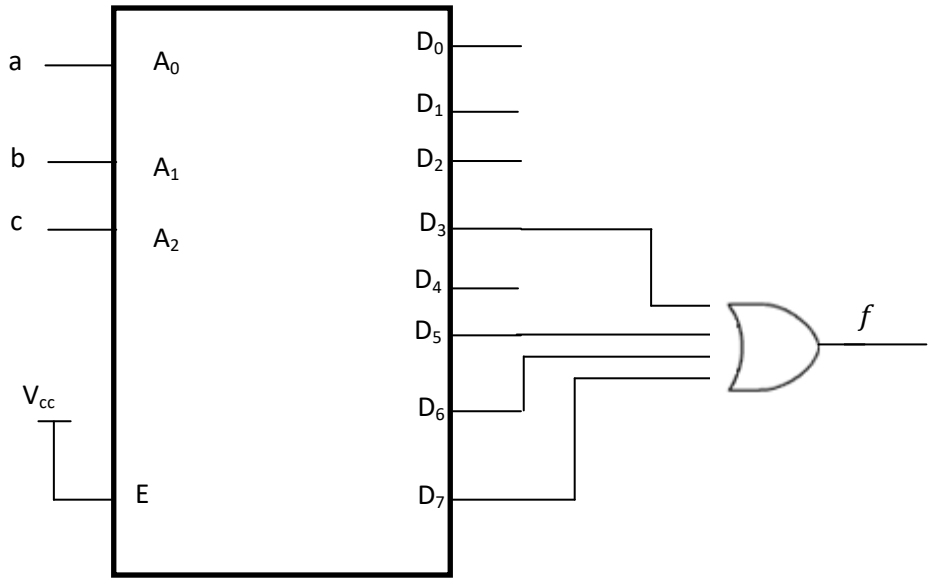
2.6.5 Le décodeur-démultiplexeur

Un démultiplexeur est un aiguilleur à une entrée de donnée, n entrées d'adresse et m sorties. La valeur de l'entrée se retrouve sur la sortie dont le numéro est codé par l'adresse. Dans cette fonction le circuit joue le rôle inverse du multiplexeur.

Ces circuits sont aussi des décodeurs : si l'entrée est maintenue active, le numéro de la sortie reflète le code de l'adresse. Dans le cas d'un code binaire on a= 2^n . Mais il existe aussi des décodeurs pour les codes BCD, GRAY , …

Un démultiplexeur dont l'entrée E est maintenue active au niveau 1 peut également servir de générateur de fonctions logiques. En effet, puisque l'on retrouve sur les sorties toutes les combinaisons possibles des entrées d'adresse, une porte OU suffit pour fabriquer une fonction logique sous sa première forme canonique

Soit la fonction logique de trois variables $f = \bar{a}\, b\, c + a\bar{b}c + \bar{c}ab + abc$. Le codage des sorties avec l'adresse (c b a) correspond aux sorties 6, 5 , 3 et 7. D'où le schéma suivant :



3.1 Définition

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

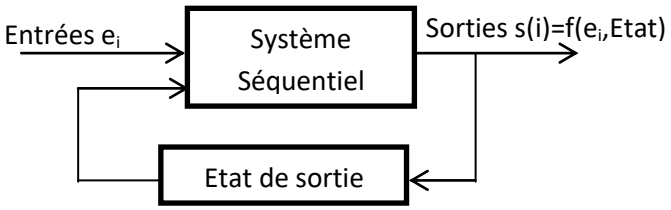


Schéma d'un système séquentiel

Exemple

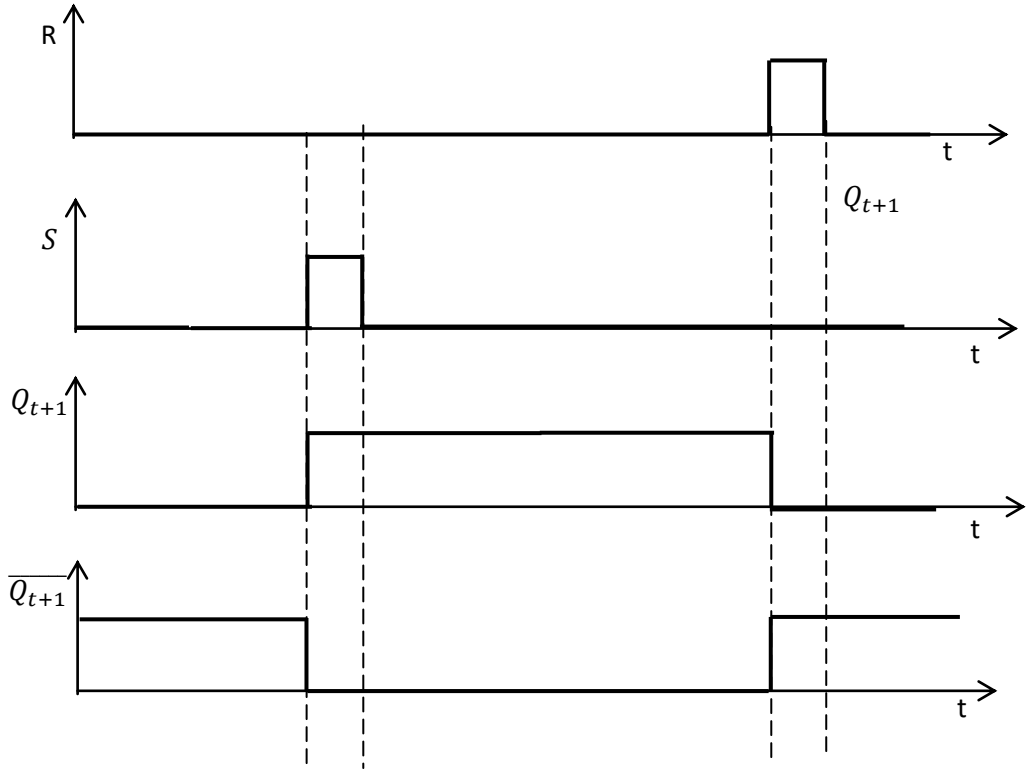
Le cahier de charge d'un système est comme suit :

L'impulsion sur un bouton poussoir (A) provoque démarrage du moteur et à libération du bouton le moteur reste en marche. En suite l'impulsion sur un bouton poussoir (B) provoque l'arrêt du moteur et à son libération le moteur reste en arrêt.

Table de vérité

B	A	M
0	0	0
0	1	1
0	0	1
1	0	0
0	0	0

Chronogramme

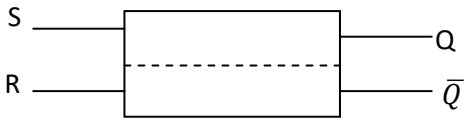


Nous constatons que pour l'état des entrées A=0 et B=0 la sortie M porte 0 et 1. Donc, on peut dire que le système est criée une mémoire qui garder l'état précèdent du système.

3.2 Bascules

3.2.1 Bascule RS (Bascule Asynchrone sans Horloge)

Est constituée par deux entrées (S mise à 1 (Set)) et (R mise à 0 (Rset)) et de deux sorties Q et $\overline{Q}$



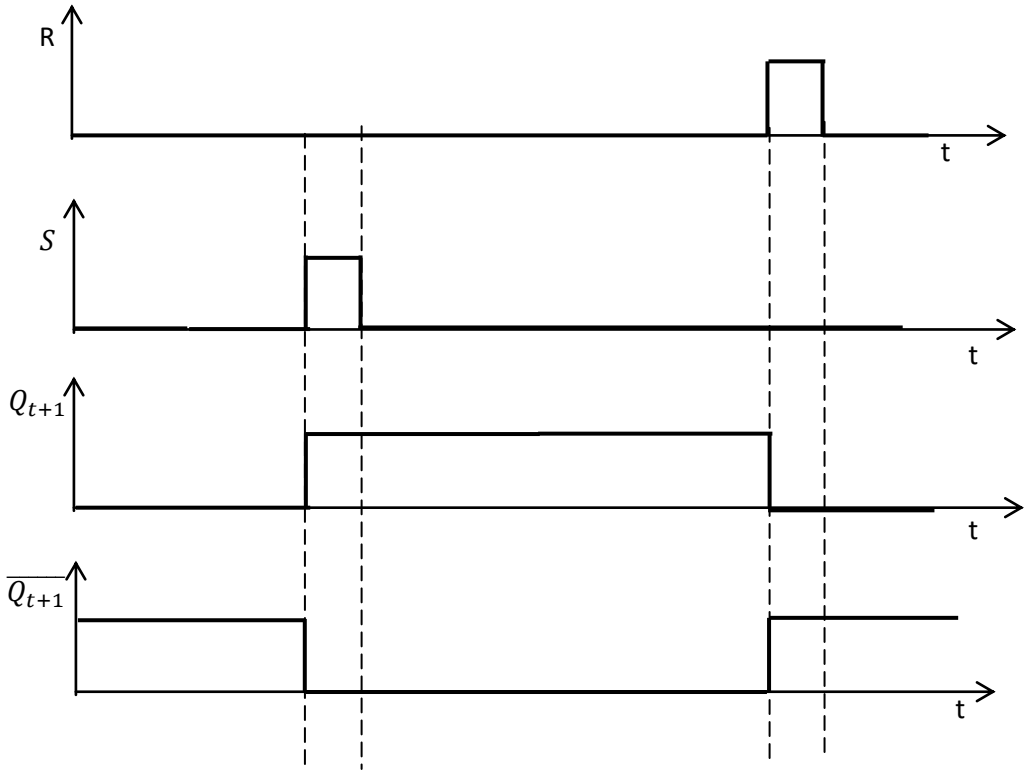
S mise à 1 implique que Q=1 (Q est forcé à un par S)

R mise à 0 implique que Q=0 (Q est forcé à zéro par R)

- **Table de vérité**

S	R	Q_t	Q_{t+1}	$\overline{Q}_{t+1}$	
0	0	0	0	1	Garder l'état précédent
0	0	1	1	0	
1	0	0	1	0	Mise à 1
1	0	1	1	0	
0	1	0	0		Mise à 0
0	1	1	0		
1	1	0	×	×	Indéterminé
1	1	1	×	×	

- **Chronogramme**



- **Equation logique**

$Q_{t+1} = f(R, S, Q_t)$

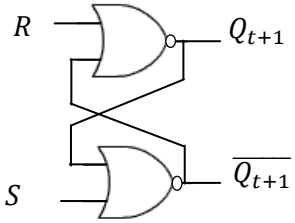
Q_t \ RS	**00**	**01**	**11**	**10**
0	**0**	**1**	×	**0**
1	**1**	**1**	×	**0**

$Q_{t+1} = \bar{R}Q_t + S$ou$Q_{t+1} = \bar{R}(Q_t + S)$

- **Schéma logique de bascule RS**

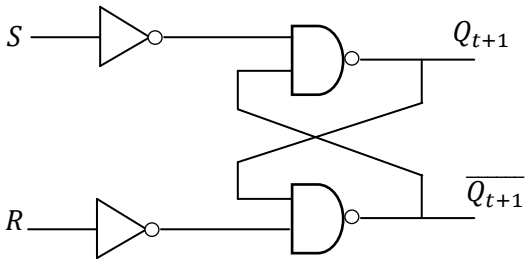
A) Représentation par les portes logique NOR

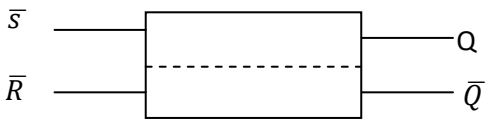
$$Q_{t+1} = \bar{R}(Q_t + S) = \overline{R + \overline{(Q_t + S)}}$$

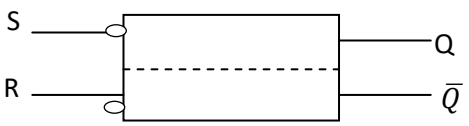


B) Représentation par les portes logique NAND

$$Q_{t+1} = \overline{\overline{\bar{R}(Q_t + S)}} = \overline{(\bar{R}Q_t).\bar{S}}$$



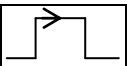




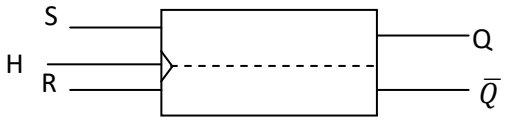
3.2.2 Bascules Synchrones

Sont des bascules asynchrone avec une troisième entrée pour le temps (t) (Horloge **H**) avec une période T, le changement de l'état de la sortie dépend de l'horloge H et l'état des entrées

L'Horloge (H) est un signal électrique périodique, on distingue quatre types d'horloge :

T 	Horloge influé sur le niveau Haut (H)	
T 	Horloge influé sur le niveau Bas (L)	
T 	Horloge influé lorsqu'il y a un passage de niveau Bas au Haut (Front montant)	
T 	Horloge influé lorsqu'il y a un passage de niveau Haut au Bas (Front descendent)	

a) Bascules RST Bascule à verrouillage (Latch)



Si T=0 → $Q_{t+1}=Q_t$

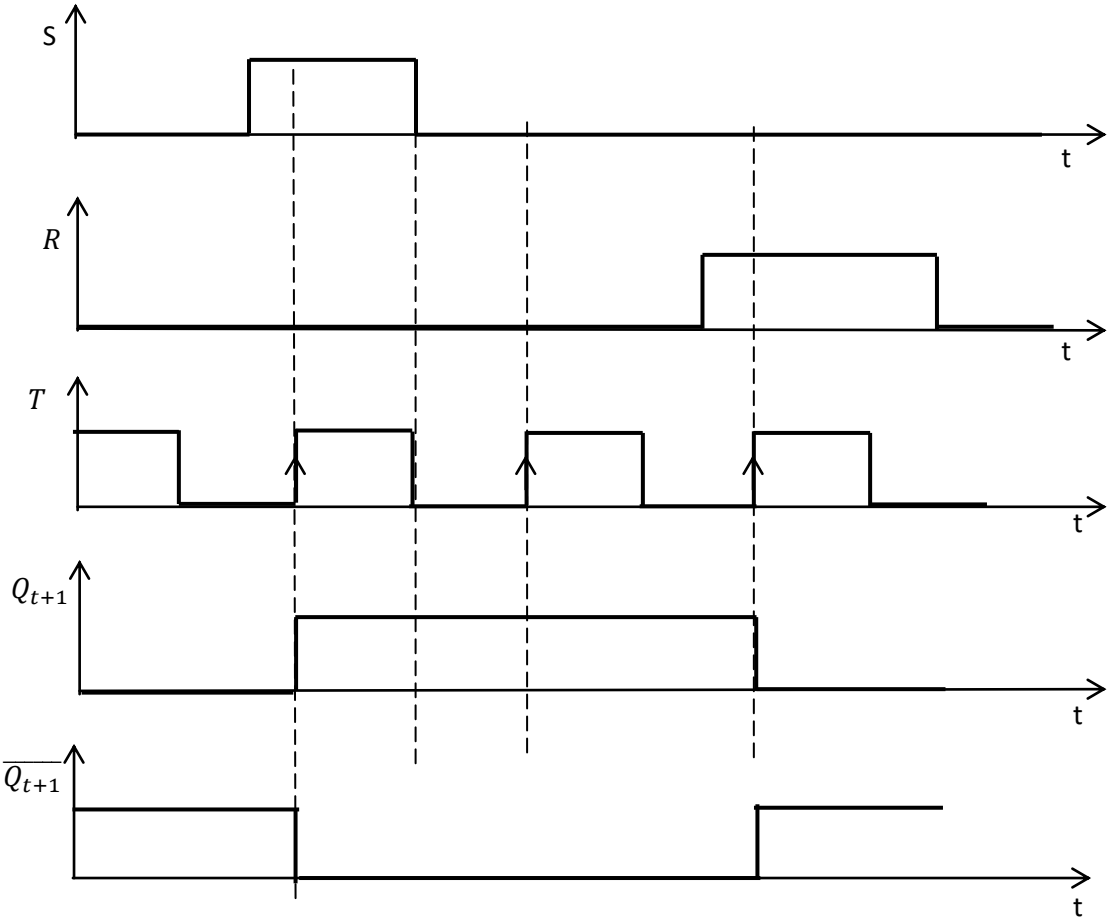
Si T=1 → Q_{t+1} dépend de l'état des entrées RS

- **Table de vérité**

T	R	S	Q_{t+1}	$\overline{Q}_{t+1}$	
0	×	×	Q_t	$\overline{Q}_t$	Mémorisation de l'état précèdent
1	0	0	Q_t	$\overline{Q}_t$	
1	0	1	1	0	Mise à 1
1	1	0	0	1	Mise à 0
1	1	1	×	×	Interdit

- **Chronogramme**

Le changement de l'état de la sortie de la bascule RST est en fonction de l'impulsion d'horloge (H) soit : en front montant ou en front descendent et l'état des entrées R et S.



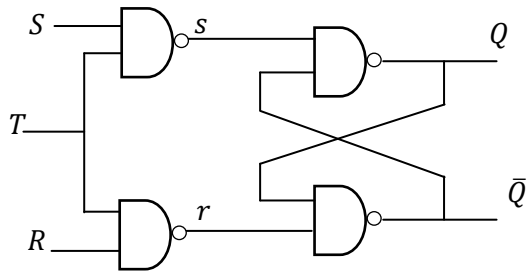
- **Equation logique**

$$Q_{t+1} = S\bar{R}T + \bar{T}Q_t + \bar{S}\bar{R}TQ_t$$

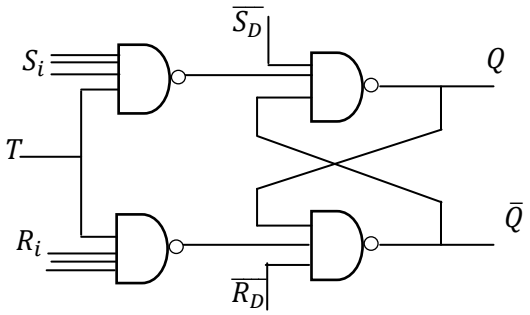
- **Schéma logique**

Représentation par les portes NAND

$s = \overline{T.S} = \bar{T} + \bar{S}$ $\quad r = \overline{T.R} = \bar{T} + \bar{R}$



b) Bascule RST à plusieurs entrées



S_i et R_i des entrées synchronisées avec l'horloge

S_D et R_D des autres entrées non synchronisé avec T est appelé les entrées de forcement (mise à 1 par l'entrée S_D=1 et mise à zéro par R_D=1).

TQ \ RS	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	1	1	×	0
10	0	1	×	0

$Q = S\overline{T}$

3.2.3Bascule JK

Bascule **JK** est une bascule **RS** mais dans le cas où la valeur de deux entrées est 1 la sortie est porté la négation de l'état précédente.

Telle que S→J et R→K

Table de vérité

K	J	Q_{t+1}
0	0	Q_t
1	0	0
0	1	1
1	1	$\bar{Q}_t$

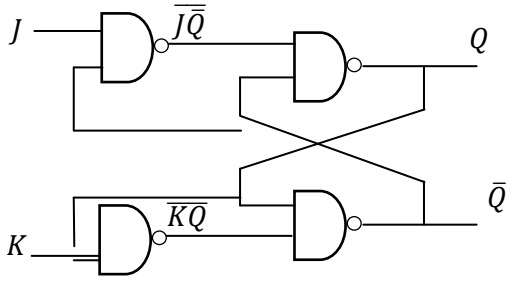
Table de KARNAUGH

Q_t \ KJ	00	01	11	10
0	0	1	1	0
1	1	1	0	0

Equation logique

$$Q_{t+1} = \overline{J}\,\overline{K}Q_t + J\overline{K} + JK\overline{Q_t}$$

Schéma logique

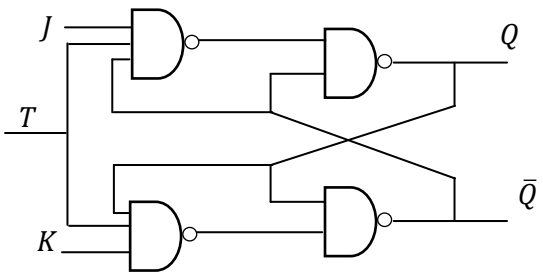


3.2.4Bascule JK synchrone

Table de vérité

T	K	J	Q_{t+1}	$\bar{Q}_{t+1}$
0	×	×	Q_t	$\bar{Q}_t$
1	0	0	Q_t	$\bar{Q}_t$
1	0	1	1	0
1	1	0	0	1
1	1	1	$\bar{Q}_t$	Q_t

Schéma logique



3.2.5Bascule D

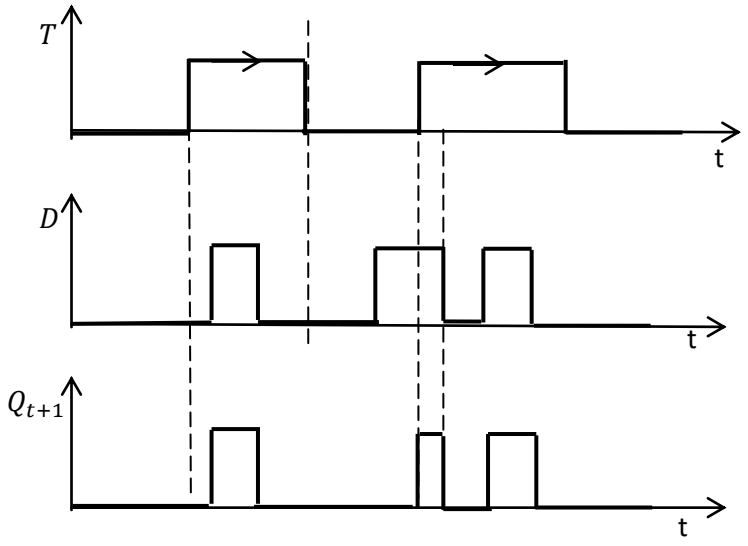
Bascule D est utilisée pour éliminer l'état indéterminé (interdit), il est résumé les deux entrées R et S par une seule entrée D telle que $S = D \;\; et \;\; R = \overline{D}$

Table de vérité

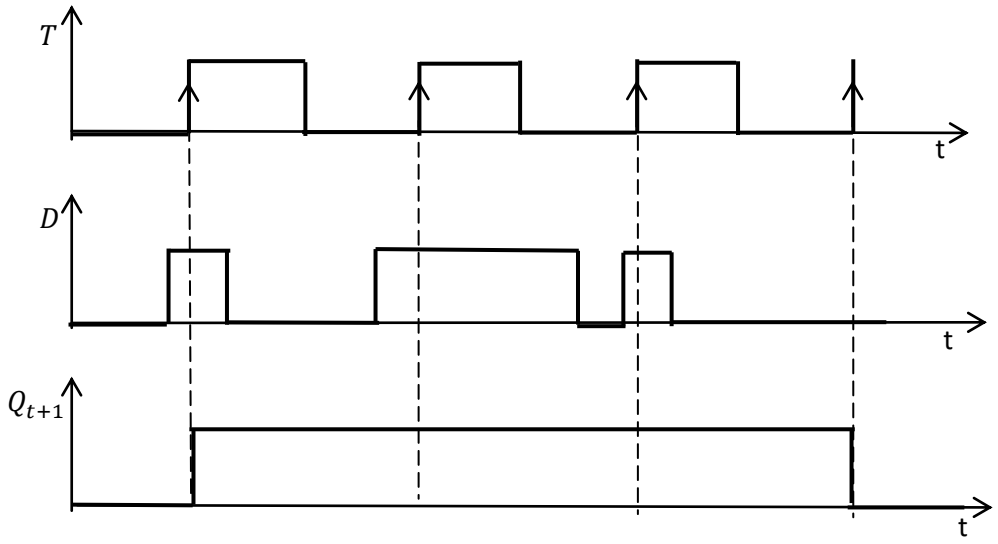
T	D	Q_{t+1}	$\overline{Q}_{t+1}$
0	0	Q_t	$\overline{Q}_t$
0	1	Q_t	$\overline{Q}_t$
1	0	0	1
1	1	1	0

Chronogramme

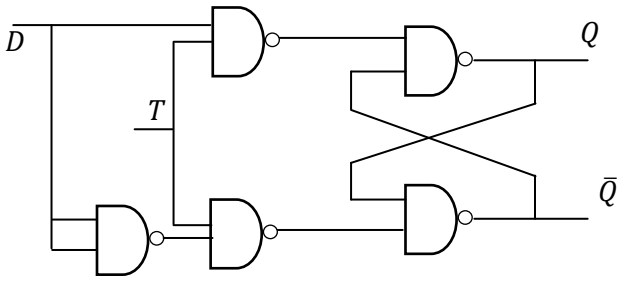
Exemple(1)



Exemple(2)



- **Schéma logique par les portes logiques NAND**



3.2.6Bascule T

Si T=0 → mémorisation de l'état précédent

Si T=1 → la sortie de la bascule sera changé tel que 0→1 et 1→0

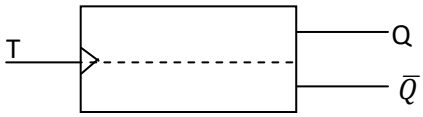


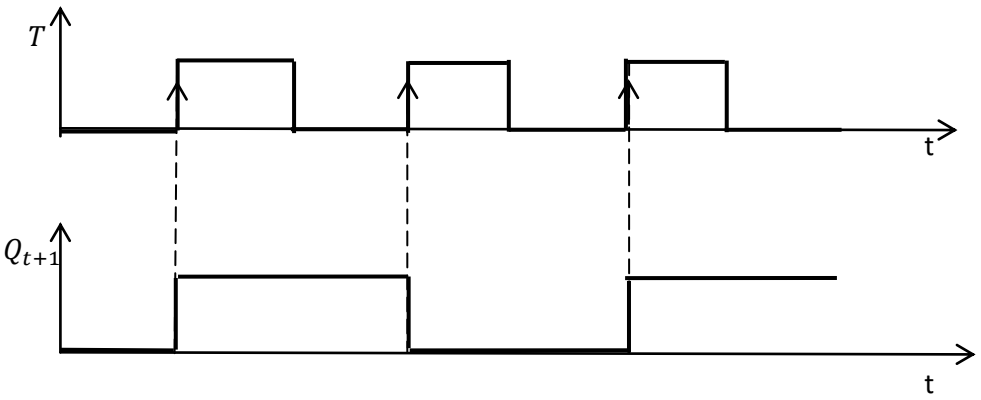
Table de vérité

T	Q_t	Q_{t+1}	$\overline{Q}_{t+1}$	
0	0	0	1	Mémorisation
0	1	1	0	de l'état
1	0	1	0	Fonctionnement
1	1	0	1	en bascule T

Equation logique

$Q_{t+1} = \overline{Q}_t$

Chronogramme



3.3Compteurs

3.3.1Définition

Les compteurs sont des ensembles des bascules montées en série ou en parallèle pour compter soit incrémenter ou décrémenté, on distingue deux types : les compteurs Asynchrone et Synchrone.

3.3.2 Compteurs Asynchrone

Sont formé par des bascules montées en série tel que chaque bascule donne l'impulsion à la bascule suivante

Un compteur modulo **n** peut compter de zéro jusqu'à (**n-1**), tel que : Si n=8 donc est compté de 0 à 7, et si n=10 donc est compté de 0 à 9.

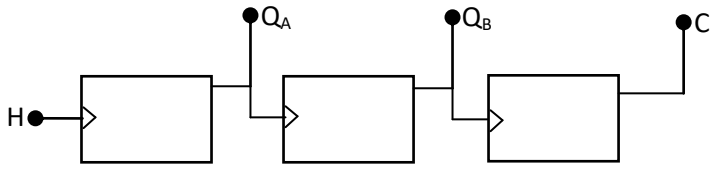


Schéma d'un compteur Asynchrone

Chronogramme

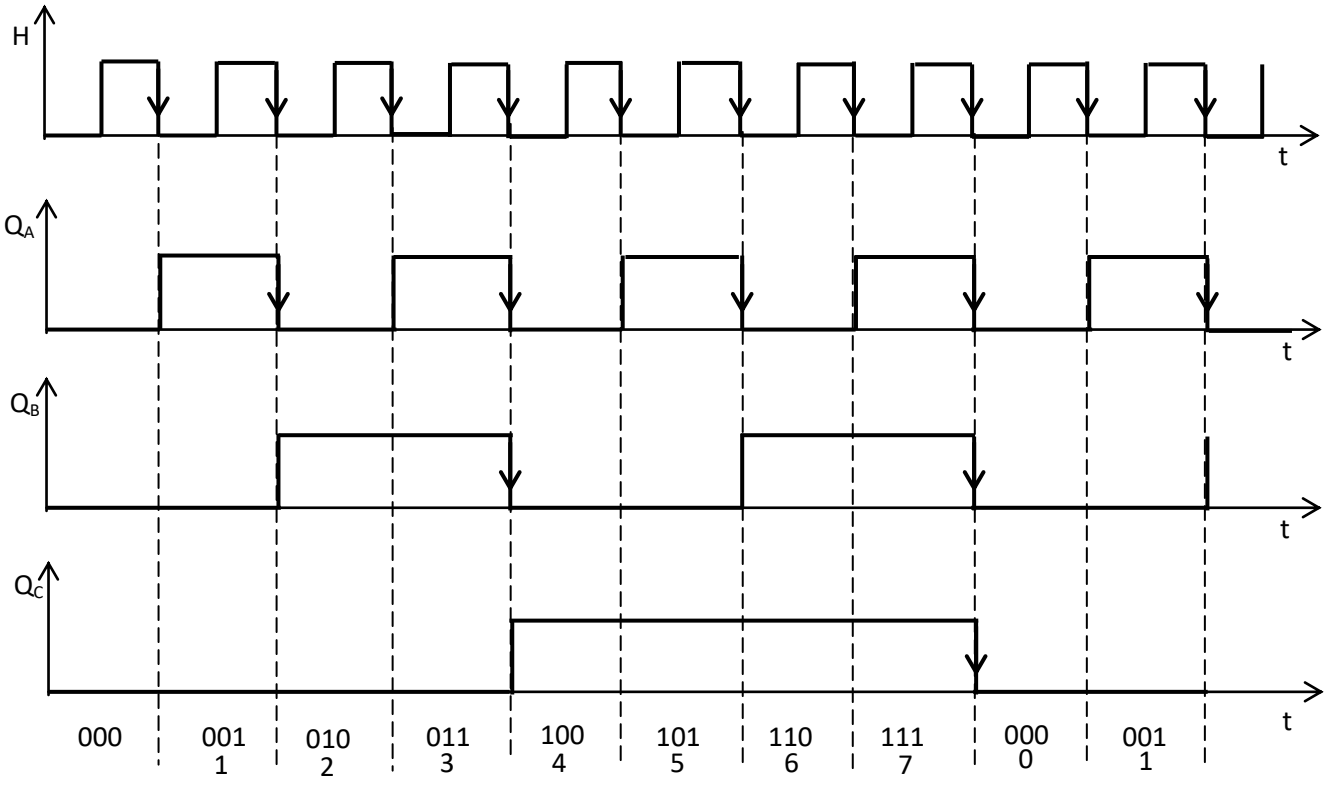
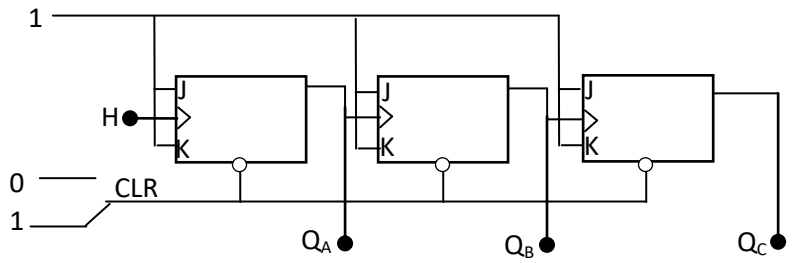


Table de vérité

Q_A	Q_B	Q_C	H
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

- On peut réaliser le compteur Asynchrone par les bascules **JK**, avec J=K=1



Compteur Asynchrone avec Bascule JK

Les entrées **CLR** forcent le compteur à l'état **0** (mise à 0)

a) Compteur Asynchrone décimal (modulo 10)

Ce compteur est compté de 0 à 9, le nombre des bascules utilisées pour compter de 0 à 9 est déterminé par l'opération suivante:

$$2^0 = 1 < 10, \quad 2^1 = 2 < 10,$$
$$2^2 = 4 < 10, \quad 2^3 = 8 < 10$$
$$2^4 = 16 > 10$$

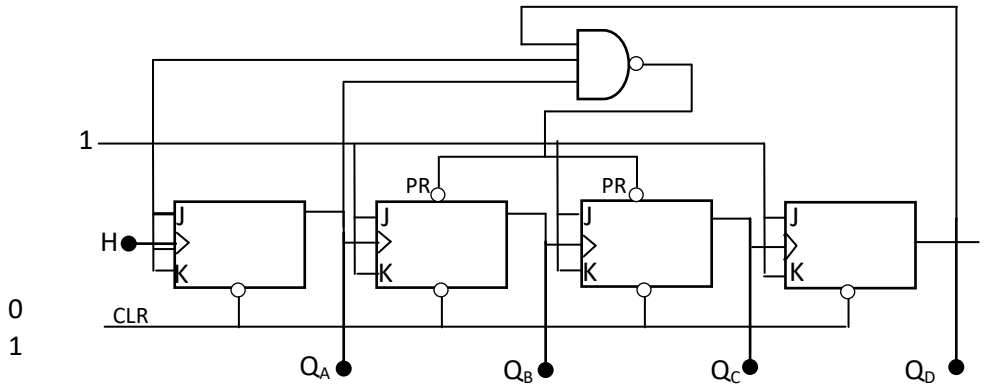
Donc le nombre des bascules utilisées au compteur décimal (modulo 10) est **4**

Table de vérité

Q_A	Q_B	Q_C	Q_D	H
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9

On remarque que les sorties des bascules peut atteindre jusqu'à 15 (1 1 1 1) mais notre compteur décimal est compté de (0 à 9), donc il est un compteur de tour incomplète.

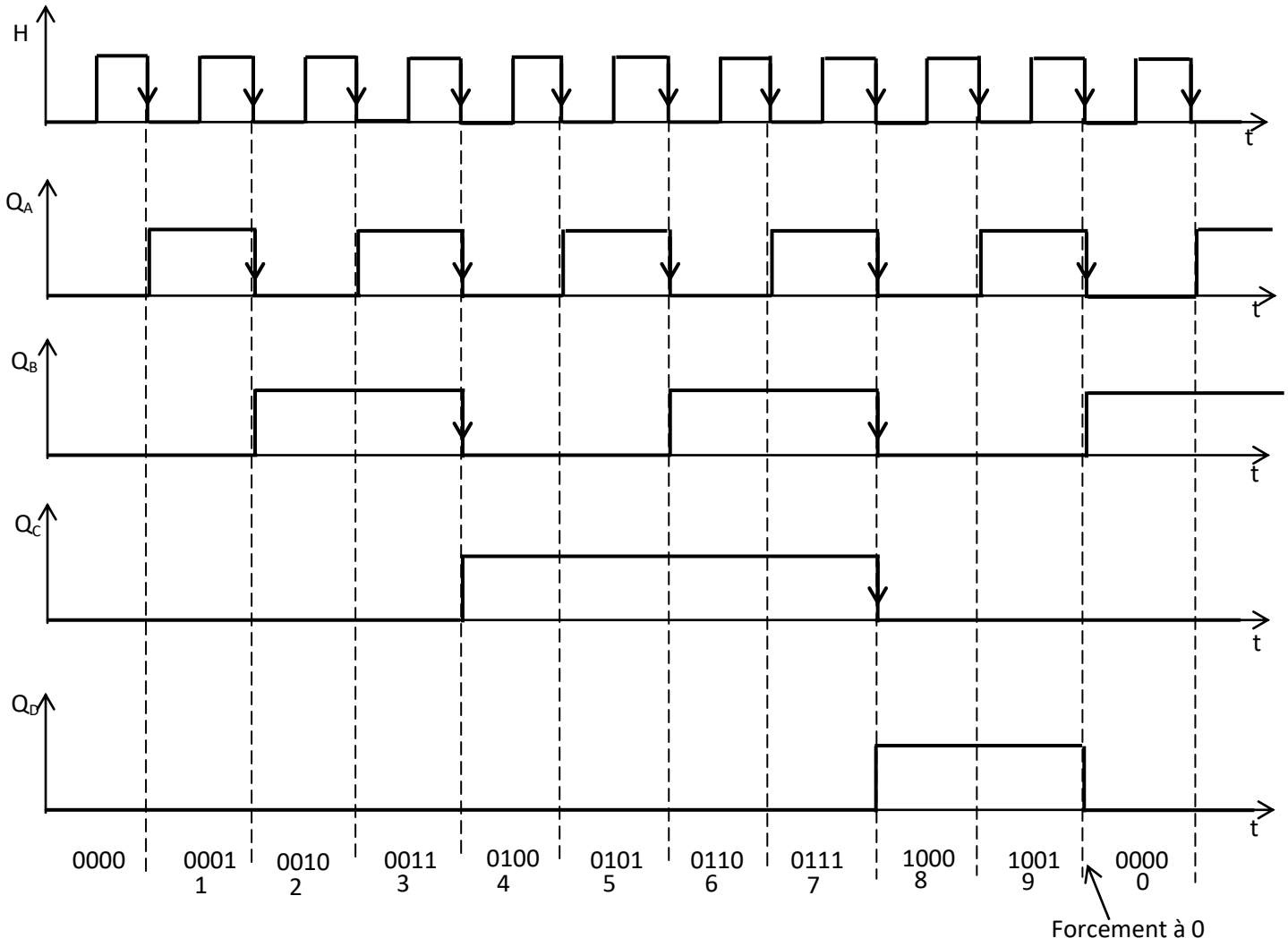
Schéma logique



Ce compteur est compté de zéro**(0000)** jusqu'à 9 **(1001).**

Donc il faut forcer les sorties des bascule Q_B et Q_C d'être 1, et en prend les sorties des bascules Q_A et Q_D avec l'horloge H comme entrée d'une porte logique (NAND) et sa sortie est montée sur l'entrée PR (Prest) des bascules ayant à sa sortie zéro qui sont dans ce cas Q_B et Q_C.

Chronogramme



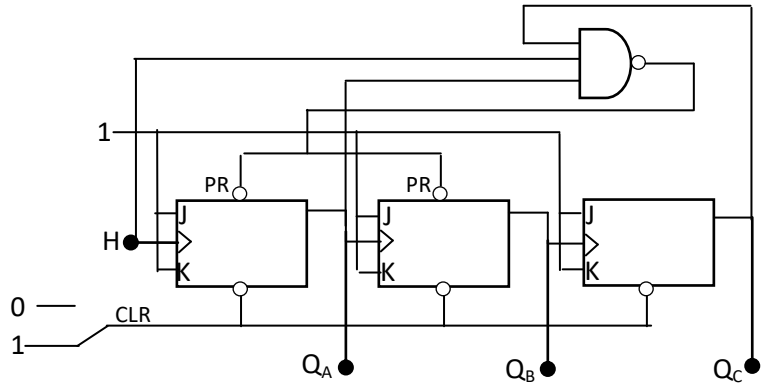
Exemple

Réaliser le schéma logique d'un compteur modulo 5 (càd compté de 0 à 4) en utilisant les bascule **JK.**

$$2^0 = 1 < 5, \quad 2^1 = 2 < 5,$$
$$2^2 = 4 < 5, \quad 2^{\mathbf{3}} = 8 > 5$$

Donc le nombre des bascules utilisées pour le compteur modulo 5 est trois (3).

- **Schéma logique**

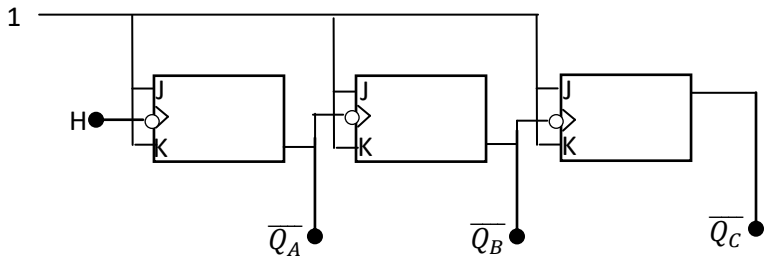


b)Décomptage Asynchrone

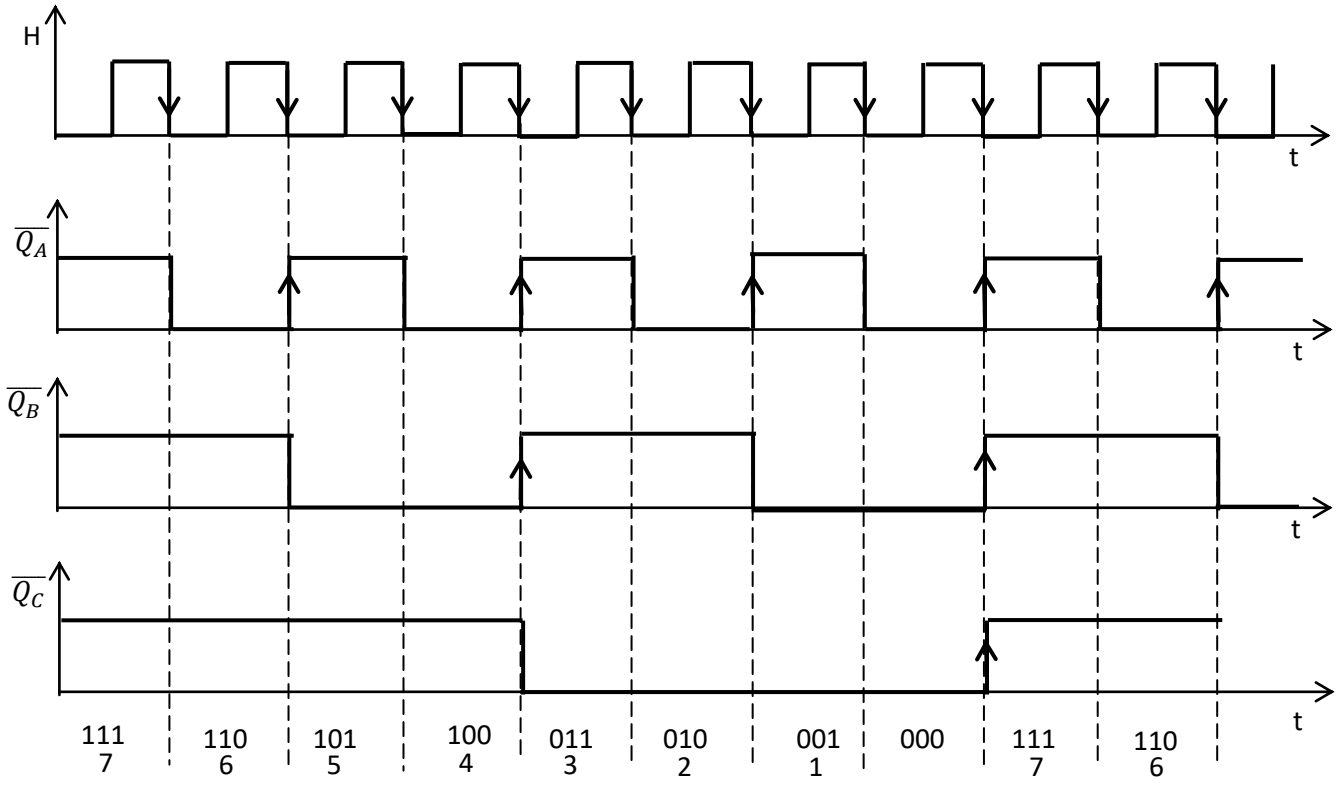
Dans ce cas, on utilise les sorties complémentaire $\bar{Q}$ Comme entrée à la bascule suivante.

Exemple

- *Décompteur Asynchrone modulo 8* **:** Comme nous avons vu ce compteur nécessite trois bascules



Chronogramme

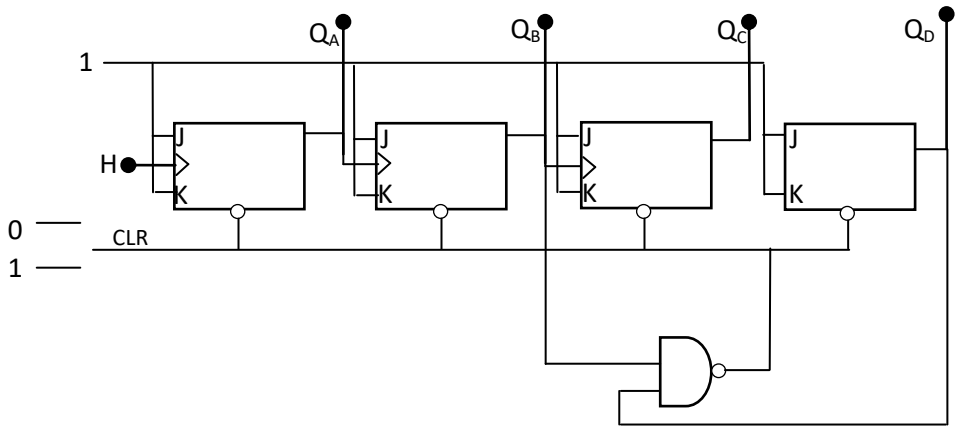


- *Compteur décimal modulo 10 avec l'utilisation des entrées de forcement CLR (Clear)*

Pour compter de 0 à 9 (1001), il faut à n=10 (1010) les sorties de compteur de3ennent zéro (0000).

Donc :

On prend les un (Q_B, Q_D) comme entrée à une porte logique (NAND) et sa sortie forcé les entrées de forcement CLR à zéro.



Compteur Asynchrone modulo 10 en utilisant les entrées de forcement CLR

3.4 Les Registres

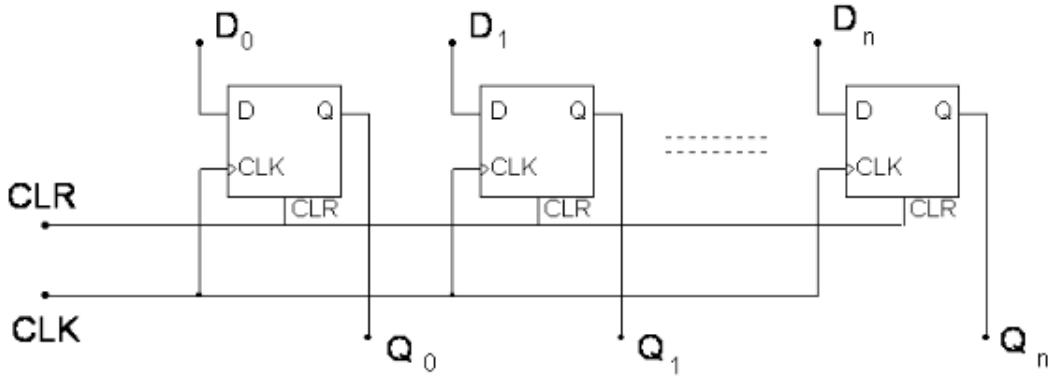
3.4.1 Définitions

Les registres sont les éléments de base des mémoires réalisées avec des semi-conducteurs. On peut se représenter un registre comme un ensemble de mémoires élémentaires susceptibles de stocker chacune un bit. L'entrée des informations dans un registre peut se faire soit en série (les unes après les autres) soit en parallèle (toutes au même moment). De la même façon la présentation des informations sur les sorties peut se faire soit en série soit parallèle. On aboutit ainsi à 4 types de fonctionnements différents pour les registres (parallèle-série, parallèle-parallèle, série-parallèle et série-série).

3.4.2 Les Registres tampon

Les registres tampon sont des registres de type parallèle-parallèle constitués de n bascules de type D commandées par une même horloge.

Au signal d'horloge (impulsion sur CLK) les entrées D sont recopiées sur les sorties Q_i. Une entrée asynchrone CLR permet, de façon prioritaire, d'effacer le contenu du registre et d'écrire $Q_i = 0$. Entre deux impulsions les sorties sont parfaitement isolées des entrées.



Registre tampon à base de bascules D

3.4.3 Les Registres à décalage

Les registres à décalage sont des registres de type série-série ou série parallèle, dans lesquels les informations sont décalées d'une bascule vers la suivante au rythme des impulsions d'une horloge. Ils sont généralement réalisés avec des bascules RS de type maître-esclave.

Le schéma de principe d'un registre à décalage (vers la droite) avec entrée série est présenté sur la Figure suivante. Au fur et à mesure des impulsions d'horloge les données présentes sur l'entrée série E sont transférées sur les différentes bascules. La présence de l'inverseur entre R et S assure toujours S = R. Les bascules ne sont donc jamais en mode mémoire mais toujours en mode SET ou RESET. Dans ces conditions la sortie Qi recopie, au moment de l'impulsion d'horloge, la valeur présente sur l'entrée Si

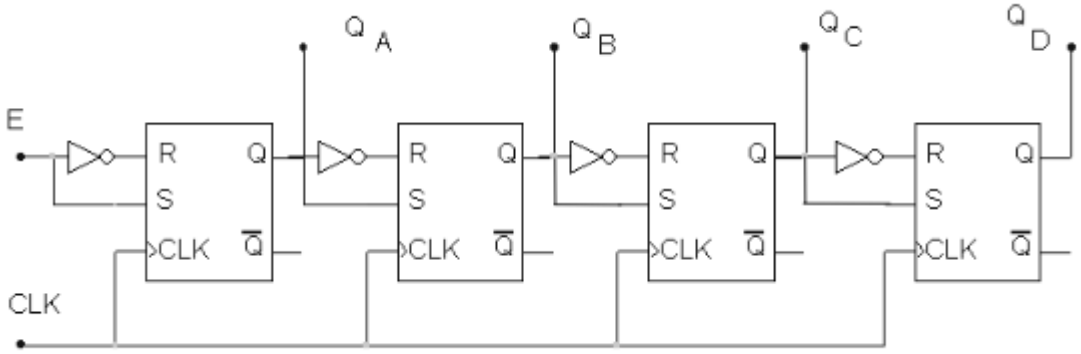


Schéma de principe d'un registre à décalage avec une entrée série (E). Le décalage se fait de Q_A vers Q_B

Donnons un exemple de fonctionnement pour fixer les idées. On suppose que la situation de départ est la suivante: E = 0 et seule la sortie Q_A de la première bascule est au niveau 1 les autres étant au niveau 0 ($Q_AQ_BQ_CQ_D$ = 1000)

- 1ére impulsion : Chaque bascule recopie sur sa sortie Q la valeur présente sur son entrée S. On obtient donc après l'impulsion (Q_A Q_B Q_C Q_D =0100).
- 2ème impulsion : Seule l'entrée S de la troisième bascule est au niveau haut. Après l'impulsion on a donc (Q_A Q_B Q_C Q_D= 0010).

On raisonne de la même façon pour la troisième impulsion et l'on aboutit à (Q_A Q_B Q_C Q_D= 0001). Ainsi au fur et à mesure des impulsions sur l'entrée d'horloge (CLK) le 1 présent initialement sur Q_A a été progressivement décalé vers la droite.

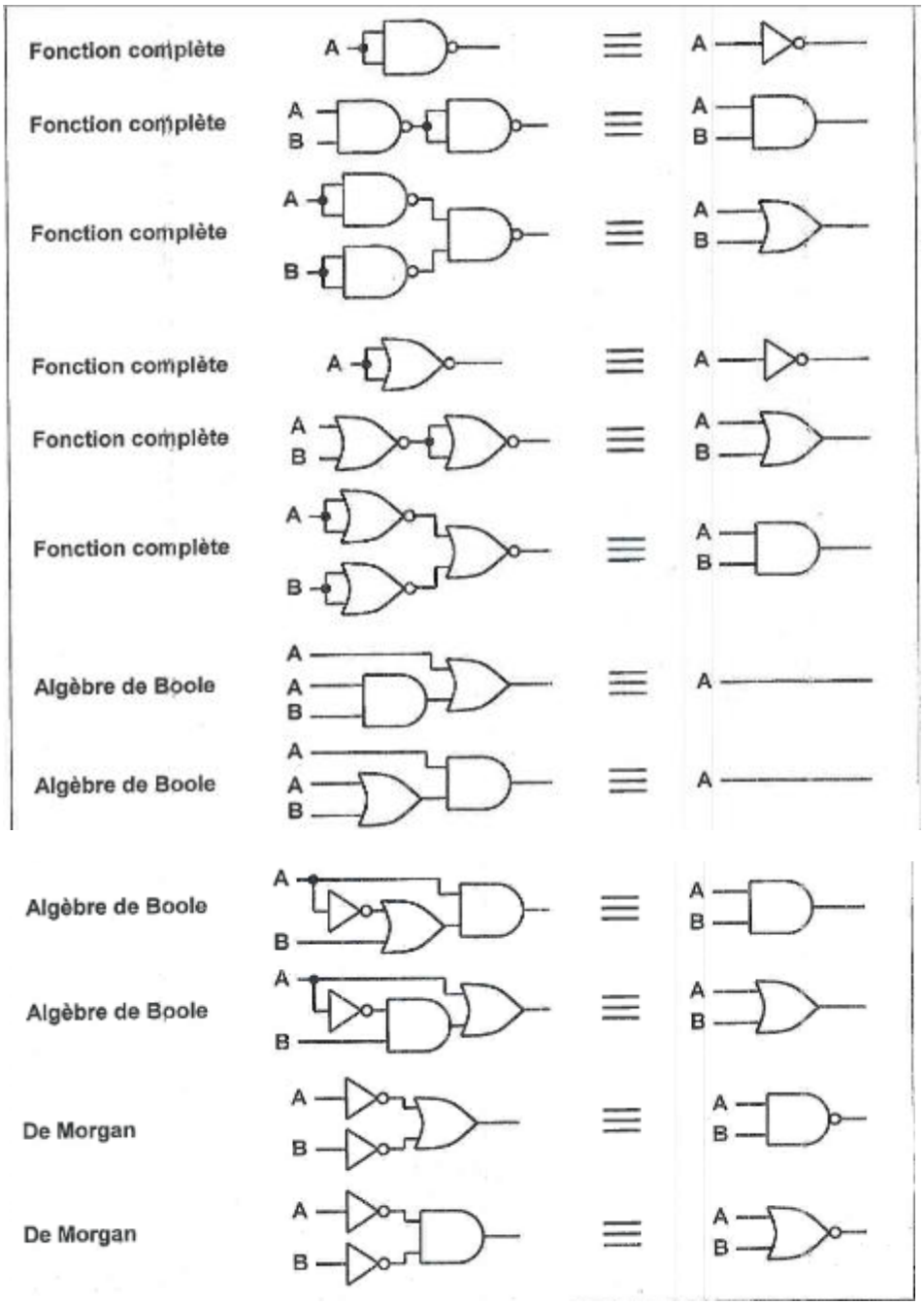
3.4.4 Les registres universels

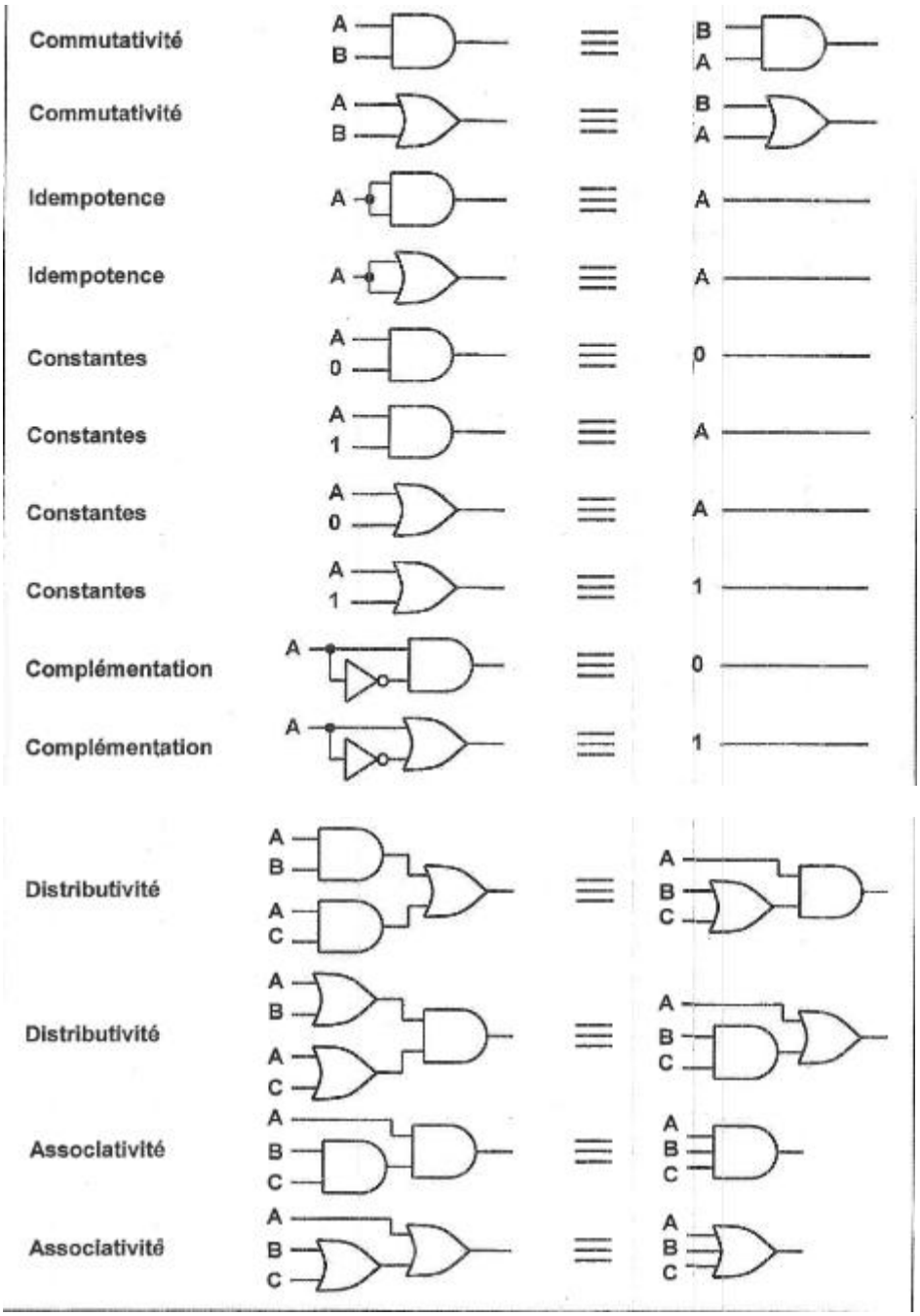
Dans la pratique il est inutile d'effectuer la synthèse des registres à décalage : un choix très vaste est offert par les constructeurs. A titre d'exemple citons les registres universels de type 194 dont le schéma est présenté sur la page suivante. Ce sont des registres 4 bits à chargement parallèle ou série. Outre une remise à zéro asynchrone (CLEAR) ils offrent la possibilité de déplacer l'information vers la droite (de Q_A vers Q_D ou vers la gauche ou de Q_D vers Q_A). Le mode de fonctionnement est choisi avec les entrées synchrones S_0 et S_1

[01] MERZOUK S, BOUZOURANE H, AMAROUCHE A, HAMOUDI D, Logique combinatoire séquentielle*. Les pages Bleues Internationale 2010*

[02] Luc MUSEUR, Electronique numérique 'Logique combinatoire et séquentielle', Université Paris 13, Institut Galilée.

Système Logique




Commutativité
A
B
≡
B
A
Commutativité
A
B
≡
B
A
Idempotence
A
≡
A
Idempotence
A
≡
A
Constantes
A
0
≡
0
Constantes
A
1
≡
A
Constantes
A
0
≡
A
Constantes
A
1
≡
1
Complémentation
A
≡
0
Complémentation
A
≡
1
Distributivité
A
B
A
C
≡
A
B
C
Distributivité
A
B
A
C
≡
A
B
C
Associativité
A
B
C
≡
A
B
C
Associativité
A
B
C
≡
A
B
C