الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي
République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Ecole Supérieure en Génie
Electrique et Energétique d'Oran
Département du cycle préparatoire
ESG2E ORAN
المدرسة العليا في الهندسة الكهربائية و الطاقوية
بوهران
قسم التكوين التحضيري

Manuel de Cours, Travaux Dirigés et Travaux Pratiques

Ingénierie 2

Ce manuel est destiné aux étudiants de 2éme année
Des classes préparatoires en Génie Electrique et Energétique d'Oran

Réalisé par

Dr HENDEL MOUNIA

MAITRE DE CONFERENCE "A" A L'ESG2E

Année Universitaire 2022/2023

Avant- propos

Conformément au contenu du programme pédagogique de l'Ecole Supérieure en Génie Electrique et Energétique d'Oran, validé par arrêté ministériel, du ministère de l'enseignement supérieur et de la recherche scientifique, ce Manuel de cours, de Travaux Dirigés (TD) et de Travaux Pratiques (TP) s'adresse aux étudiants de la deuxième Année, module "Ingénierie 2". Ce support a pour principale objectif, la maitrise des notions fondamentaux des systèmes automatisés circuits logiques combinatoires et séquentiels, Grafcet et des Automates Programmables Industrielles (API), et ceux au tour de 6 chapitres, de 3 fiches de TD et une fiche de TP, des cours en ligne sont également disponibles et dont les liens sont en attachés à la fin de chaque chapitre :

- CHAPITRE 1 : LES SYSTEMES AUTOMATISES.
- CHAPITRE 2 : CIRCUITS COMBINATOIRES.
- CHAPITRE 3 : SYSTEMES LOGIQUES SEQUENTIELS.
- CHAPITRE 4 : COURS COMPTEURS/DECOMPTEURS.
- CHAPITRE 5 : LE GRAFCET.
- CHAPITRE 6 : LES AUTOMATES PROGRAMMABLES INDUSTRIELS (API).
- Fiche TD 1 : Logique combinatoire et Logique Séquentielle.
- Fiche TD 2 : Les compteurs/Décompteurs synchrones/Asynchrone.
- Fiche TD 3 : GRAFCET.
- TP 1 : Grafcet et API sous AUTOMGEN.

Avant- propos :

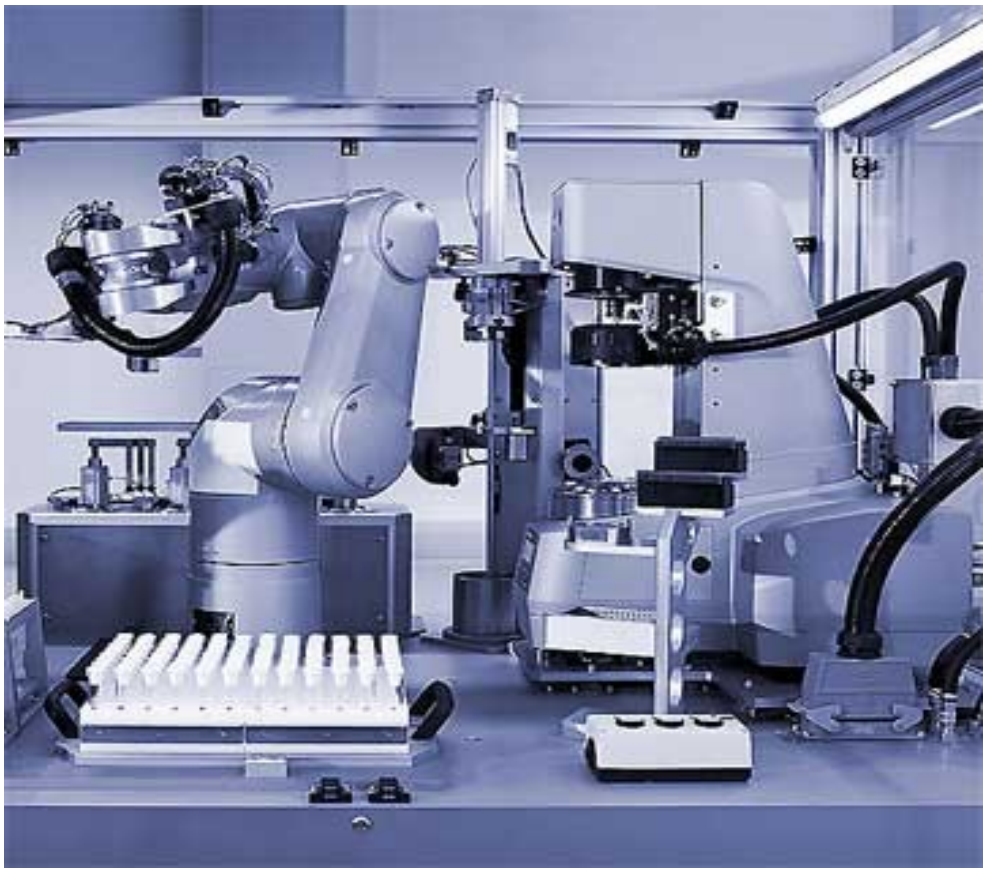
Conformément au contenu du programme pédagogique de l’Ecole Supérieure en Génie Electrique et Energétique d’Oran, validé par arrêté ministériel, du ministère de l'enseignement supérieur et de la recherche scientifique, ce Manuel de cours, de Travaux Dirigés (TD) et de Travaux Pratiques (TP) s'adresse aux étudiants de la deuxième Année, module "Ingénierie 2". Ce support a pour principale objectif, la maîtrise des notions fondamentaux des systèmes automatisés circuits logiques combinatoires et séquentiels, Grafcet et des Automates Programmables Industrielles (API), et ceux au tour de 6 chapitres, de 3 fiches de TD et une fiche de TP, des cours en ligne sont également disponibles et dont les liens sont en attachés à la fin de chaque chapitre :

- CHAPITRE 1 : LES SYSTÈMES AUTOMATISES.
- CHAPITRE 2 : CIRCUITS COMBINATOIRES.
- CHAPITRE 3 : LOGIQUES SÉQUENTIELS.
- CHAPITRE 4 : COURS COMPTEURS/DÉCOMPTEURS.
- CHAPITRE 5 : LE GRAFCET.
- CHAPITRE 6 : LES AUTOMATES PROGRAMMABLES INDUSTRIELS (API).
- Fiche TD 1 : Logique combinatoire et Logique Séquentielle.
- Fiche TD 2 : Les compteurs/Décompteurs synchrones/Asynchrone.
- Fiche TD 3 : GRAFCET.
- TP 1 : Grafcet et API sous AUTOMGEN.

Table des matières

CHAPITRE 1 : LES SYSTÈMES AUTOMATISES.

"L’apprentissage est une porte ouverte sur l’infini, chaque nouveau savoir étant une clé qui nous rapproche un peu plus de la compréhension du monde."



CHAPITRE 1 : LES SYSTEMES AUTOMATISES

1.1 Définition d'système automatisé :

C'est un système qui effectue le même cycle de travail pour lequel il a été programmé.

C'est un système en mesure d'exécuter des tâches sans l'intervention de l'humain.

1.2 La structure d'un système automatisé :

La structure d'un système automatisé est constituée principalement : d'un pupitre, d'une Partie Commande, d'une Partie Opérative, et deux éléments d'interface dénommé Pré-actionneurs et Capteurs (Figure 1.1).

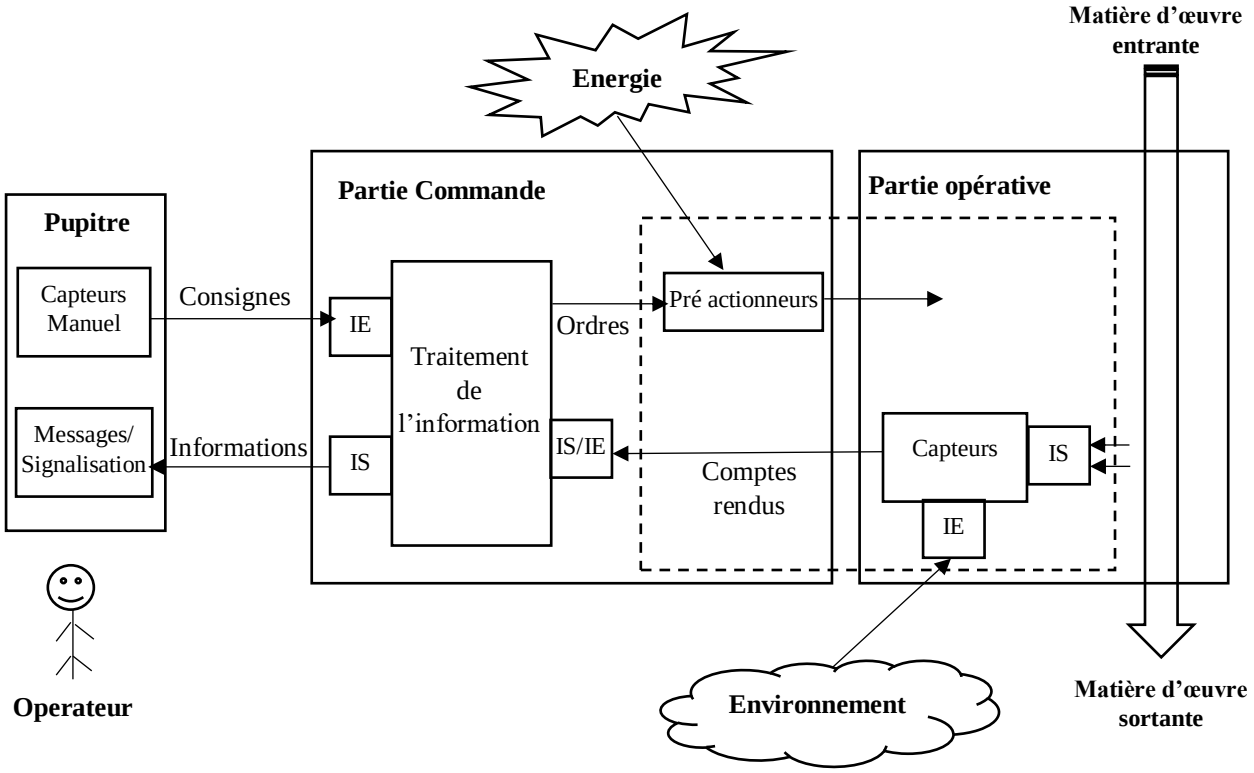


***Figure 1.1*:** Structure générale d'un système automatisé

1.2.1 Pupitre : Assure le dialogue Homme/Partie-commande. L'utilisateur envois des consignes operateur (à partir du bouton Marche/Arrêt ou à partir d'un clavier), et reçoit des informations visuelles (à travers des signalisations et des messages…etc.).

1.2.2 Partie Commande (PC): Elle représente l'ensemble des Moyens de traitement de l'information. La partie commande est considérée comme étant le "cerveau" qui gère le système automatisé. Elle adresse des ordres (consignes opératives) a la partie opérative, à partir :

- Du programme dont elle dispose.

- Des informations renvoies par les capteurs.
- Et les consignes données par l'utilisateur.

1.2.3 Partie Opérative (PO) : Il s'agit de la partie qui effectue le travail dénommé "Machine", elle constitue l'ensemble des moyens techniques qui agissent sur la matière d'œuvre entrante pour avoir la matière d'œuvre sortante. De même qu'elle transmet à la partie commande les comptes rendu concernant l'état du système ou de l'environnement.

1.2.4 Eléments d'interface : Assure l'interaction entre la partie commande et la partie opérative. On distingue deux organes à savoir : les pré-actionneurs et les capteurs (plus de détail dans la section 4)

1.3. Les modes de commandes:

On déduit de la section précédente qu'un échange d'informations est obligatoire entre la PC et la PO ; Ces informations sont respectivement de type Ordres et Comptes rendus. En effet, la PO transfert à la PC des comptes rendus concernant l'état du système ou de l'environnement, et en se basant sur ces informations la PC envois des ordres a la PO.

Une fois les ordres (actions) établis par la PC, la PO effectue les actions demandées. On distingue donc, deux modes de commandes qui peuvent être utilisés par un système automatisé:

1.3.1 Mode de commande directe :

Dénommée aussi système a chaine ouverte; la PC adresse des ordres a la PO, sans vérifier si les actions ont bien été réalisé.

Par exemple, les feux tri colores, le système ne vérifie pas si les voitures se sont bien arrêtées.

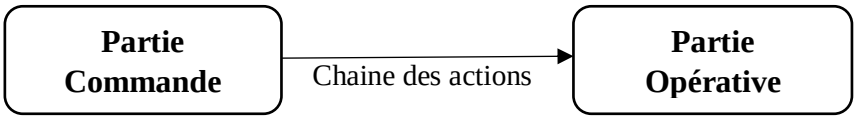


***Figure 1.2*:** Mode de commande directe

1.3.2 Mode de commande avec compte rendu d'exécution :

Dénommé aussi système a chaine fermée, la partie commande du système vérifie que les actions demandées ont été bien réalisé par les PO. Ce type de systèmes sont considéré meilleur et plus fiable.

Par exemple, le passage à niveau, la barrière se lève à condition que le système est vérifier que le train est bien passé.

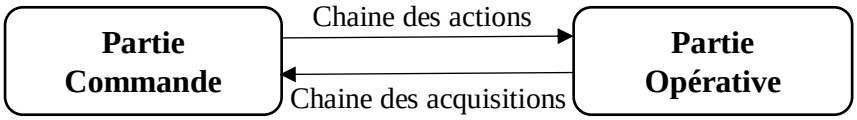


***Figure 1.3*: Mode de commande avec compte-rendu d'exécution**

1.4 Structure et fonctionnement d'une chaîne fonctionnelle :

Tout système automatisé peut être constitue d'une ou de plusieurs chaînes fonctionnelles. Par exemple un système automatisé d'assemblage de petite voiture pour enfant (voir figure 1.4) est constitué de quatre chaînes fonctionnelles suivantes :

- Montage du châssis,
- Avance du plateau,
- Sertissage,
- Montage des roues.

Chaque chaîne fonctionnelle est subdivisée en **une chaîne d'information** et **une chaîne d'énergie (Figure 1.4).**

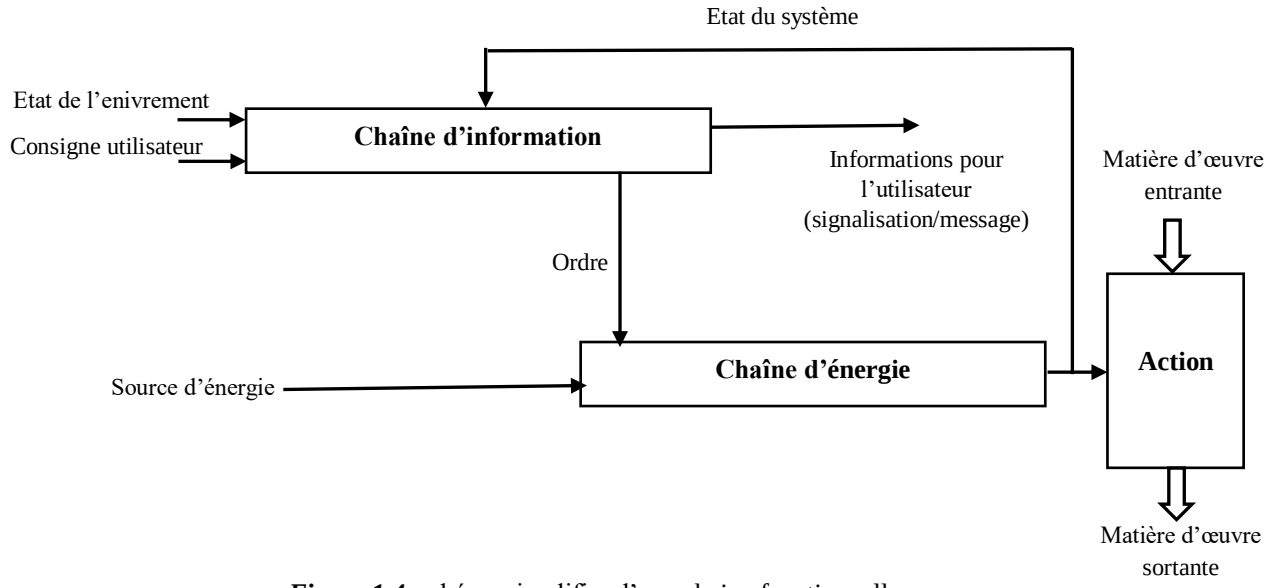


Figure 1.4: schéma simplifier d'une chaine fonctionnelle

1.4.1 Chaîne d'information :

La chaîne d'information est constituée de trois blocs représentant chacun une fonction.

a) Fonction "Acquérir": elle va servir à recevoir les informations provenant de l'utilisateur par l'intermédiaire de capteurs manuels: boutons poussoirs, interrupteurs, claviers, télécommandes…etc.

De même, qu'elle va recevoir des informations du système ou de l'environnement extérieur via les capteurs interne est externe.

b) *Fonction "Traiter" :* C'est la partie intelligente du système automatisé. Elle va permettre de décider ce que le système doit faire, et ceux en fonction des informations reçu du système et de l'utilisateur.

Dans cette partie on peut trouver des éléments tel des : cartes électroniques, automates programmables, Ordinateurs…etc.

c) *Fonction "Communiquer":* la fonction va permettre d'une part d'informé le système de ce qui doit être fait par la suite (des ordres).

Mais également, elle va communiquer des informations visuelles ou sonore sous forme: logique (lampe ou alarme), numérique (afficheur), ou analogique (compteur); afin de lui indiquer par exemple ce qui a étais fait.

Dans ce bloc on peut trouver des liaisons filaires (câbles, fils), ou des liaisons sans fils (wifi, radio, bleutooth).

Remarque : les capteurs sont des éléments embarqué dans les systèmes automatisés, ils permettent de transformer la variation d'une grandeur physique participant au fonctionnement de l'automatisme en un signal électrique.

1.4.2 Chaîne d'énergie :

La chaîne d'énergie est constituée de quatre blocs représentant chacun une fonction bien déterminé.

a) *Fonction alimenter:* elle consiste à alimenter le système en énergie afin d'assurer l'action sur la matière d'œuvre.

La source d'énergie pour alimenter un système automatisé peut se présenter sous plusieurs natures: électrique, hydraulique, fossile…etc.

Ainsi le système d'alimentation doit être en adéquation avec le type d'énergie à distribuer, par exemple:

- Pour une énergie électrique, le système doit être menu d'une pile, une batterie, ou un secteur électrique.
- Pour une énergie fossile le système doit contenir un réservoir.
- Pour une énergie hydraulique, le système d'alimentation doit être de type pompe hydraulique.

b) Fonction distribuer : la fonction permet de distribuer l'énergie a la partie opérative, en fonction des ordres issue de la chaine d'information (c'est le rôle des pré-actionneur). Dans ce module on peut trouver par exemple :

- Des fils ou des câbles si le système utilise de l'énergie électrique.
- Des vannes ou des tuyaux si le système utilise une énergie fossile ou hydraulique.

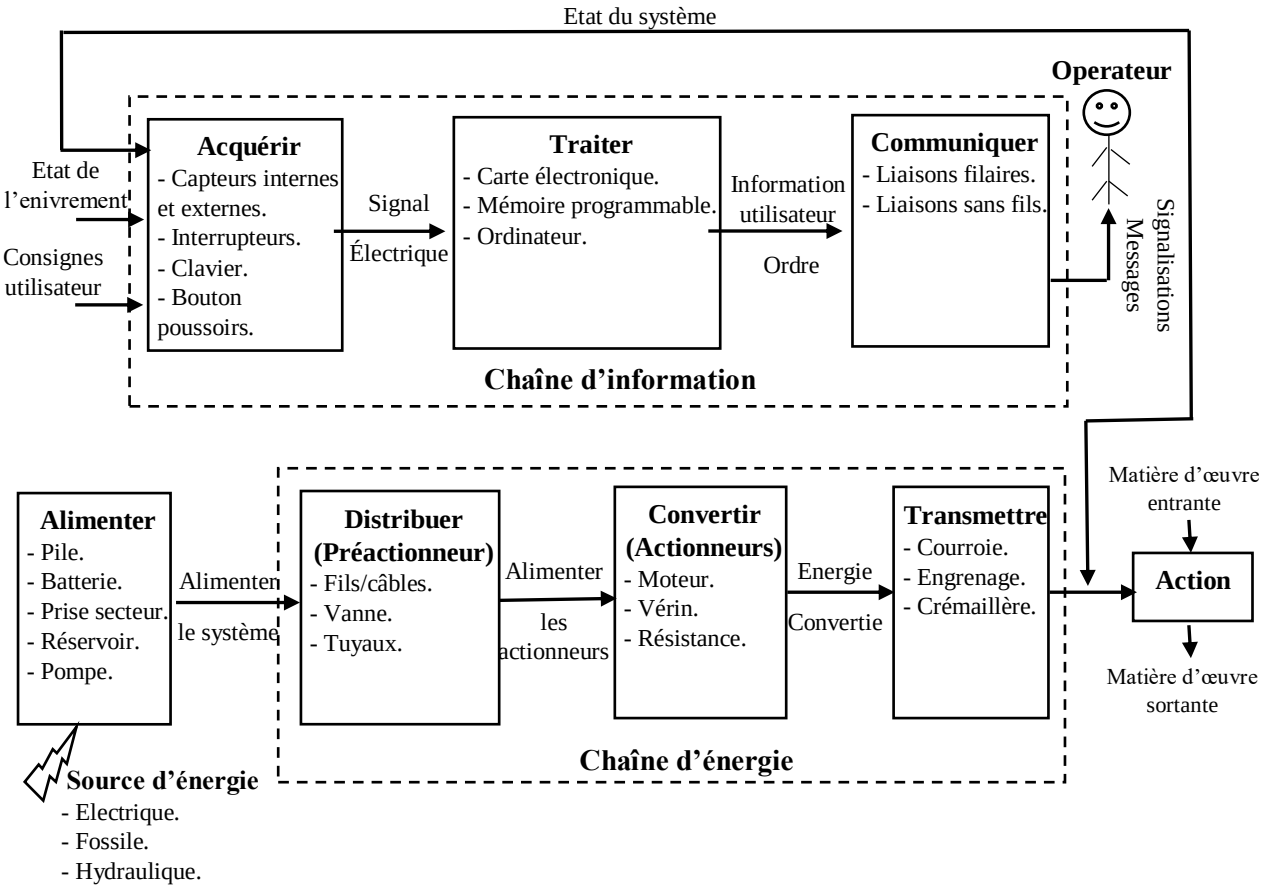
c) Fonction convertir : elle va permettre de transformer l'énergie distribué en une autre forme d'énergie, utilisable par les effecteurs de la partie opérative (c'est le rôle des actionneurs). Un actionneur peut par exemple être:

- Un moteur qui va transformer une énergie électrique en une énergie de rotation.
- Un vérin qui va recevoir de l'énergie électrique ou hydraulique, et en sortie il donne une énergie mécanique de translation.
- Une résistance qui va transformer une énergie électrique en une énergie thermique.

d) Fonctions transmettre et action : c'est deux fonctions sont en générale exécutées par des mécanismes, en effet, ces blocs sont constitués des pièces relier entre elles par des liaisons mécaniques.

Le rôle de ces liaisons mécanismes est de transmettre la nouvelle énergie obtenue à partir des actionneurs au effecteurs afin d'obtenir l'action voulus. Parmi les types de liaisons mécaniques existantes, on peut citer:

- Les courroies et les engrenages, qui va nous permettre d'accentuer ou de diminuer la cadence d'un mouvement de rotation.
- Des crémaillères qui va nous permettre de transformer un mouvement de rotation en un mouvement de translation, etc.

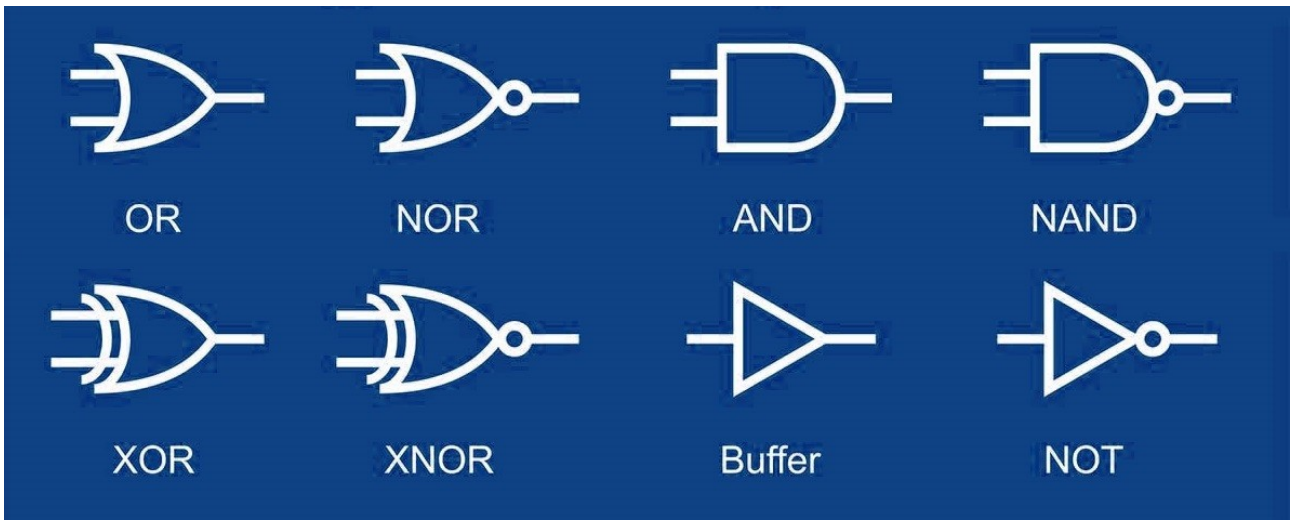

Etat du système
Operateur
Acquérir
- Capteurs internes et externes.
- Interrupteurs.
- Clavier.
- Bouton poussoirs.
Etat de l'enivrement
Consignes utilisateur
Signal
Électrique
Traiter
- Carte électronique.
- Mémoire programmable.
- Ordinateur.
Information utilisateur
Ordre
Communiquer
- Liaisons filaires.
- Liaisons sans fils.
Signalisations
Messages
Chaîne d'information
Alimenter
- Pile.
- Batterie.
- Prise secteur.
- Réservoir.
- Pompe.
Alimenter le système
Distribuer (Préactionneur)
- Fils/câbles.
- Vanne.
- Tuyaux.
Alimenter les actionneurs
Convertir (Actionneurs)
- Moteur.
- Vérin.
- Résistance.
Energie Convertie
Transmettre
- Courroie.
- Engrenage.
- Crémaillère.
Matière d'œuvre entrante
Action
Matière d'œuvre sortante
Chaîne d'énergie
Source d'énergie
- Electrique.
- Fossile.
- Hydraulique.

Biobibliographie :

[1] "Systèmes automatisés : Principes et applications" par Jean-Pierre Desbiens.
[2] "Introduction to Automation and Control Systems" par William S. Levine.
[3] "Fundamentals of Automation and Robotics" par Hassan A. Karimi**.**

CHAPITRE 2 : CIRCUITS COMBINATOIRES.

""Le véritable apprentissage ne consiste pas seulement à emmagasiner des connaissances, mais à transformer chaque découverte en une opportunité d'évoluer"



CHAPITRE 2 : CIRCUITS COMBINATOIRES

2.1 Définition :

Un circuit combinatoire est un système composé d'un ensemble de portes logiques, et dont l'état de(s) sortie(s) à l'instant t est fonction exclusivement des variables présentées à son entrée au même instant: $\boldsymbol{S(t) = F(E(t))}$**.** Ainsi l'application à l'entrée d'une même combinaison de variables conduira forcement a une sortie inchangée.

Fig. 2.1 : *schéma représentatif d'un circuit combinatoire*.

2.2 Synthèse d'un circuit combinatoire

La synthèse d'un circuit logique combinatoire consiste à proposer un logigramme répondant à un cahier des charges, les étapes de conception sont les suivantes :

- Identifier les entrées et les sorties (IN / OUT) du circuit.
- Construire la table (les tables) de vérité.
- Identifier chaque fonction à partir de la table de vérité.
- Simplifier chaque fonction.
- Dessiner le schéma du circuit.

2.3 Les circuits de transcodage :

Un circuit de transcodage est un circuit qui transforme une information présente en entrée sous une forme donnée : ***code1,*** en une information équivalente en sortie mais sous une autre forme : ***code 2***
Dans cette section nous allons faire la synthèse de trois circuits de transcodage couramment utilisés, à savoir :

- Les codeurs (encodeurs).
- Les décodeurs.
- Les transcodeurs Binaire/Gray.

a. Codeurs (Encodeur)

Un codeur est un circuit à $\boldsymbol{N = 2^n}$ entrées et $\boldsymbol{n}$ sorties qui code en binaire le rang de la seule entrée activée.
Application pratique : codeurs de clavier numérique.

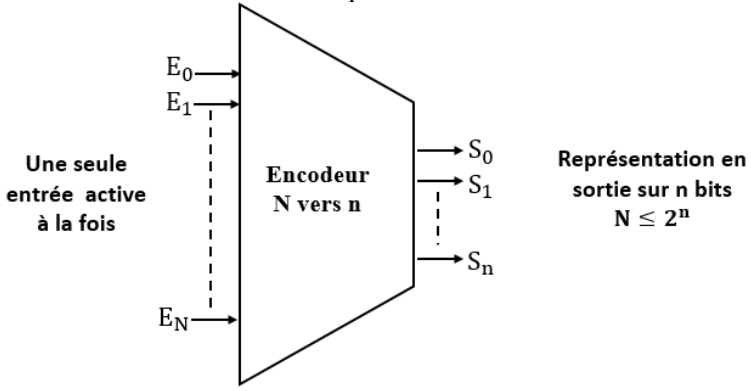


Fig. 2.2 : *schéma représentatif d'un codeur.*

Exemple 2.1 : Faire la synthèse d'un encodeur sur n=3 bits $\rightarrow 2^3 = 8$ entrées et 3 sorties .

1. Identification des entrées et des sorties (IN / OUT) du circuit :

- *Entrées* : encodeur sur n=3 bits → N = 2^3= 8 entrées: E_0, E_1,….., E_7.
- *Sorties* : n= 3 → 3 sorties: S_0, S_1, S_2.

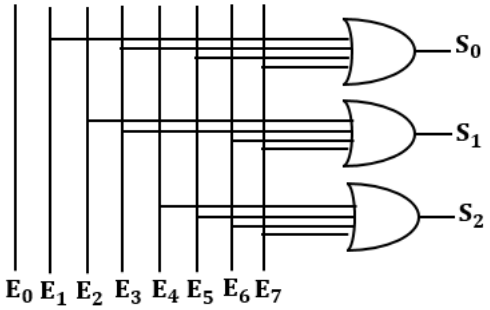
2. Construction de la table (les tables) de vérité :

	E_0	E_1	E_2	E_3	E_4	E_5	E_6	E_7	S_0	S_1	S_2
0	1								0	0	0
1		1							0	0	1
2			1						0	1	0
3				1					0	1	1
4					1				1	0	0
5						1			1	0	1
6							1		1	1	0
7								1	1	1	1

3. Identifier chaque fonction à partir de la table de vérité et 4. simplifier chaque fonction :

$S_2 = E_4 + E_5 + E_6 + E_7$.
$S_1 = E_2 + E_3 + E_6 + E_7$.
$S_0 = E_1 + E_3 + E_5 + E_7$.

5. Dessiner le schéma du circuit :



b. Décodeur

Joue le rôle inverse d'un codeur, il est constitué de :

- $\boldsymbol{n}$ Entrées de données : Code sur n bits
- 2^n Sorties (lignes)
- A chaque code en entrée, une ligne de sortie est sélectionnée

Application pratique : Adressage d'une mémoire**,** Génération de fonctions

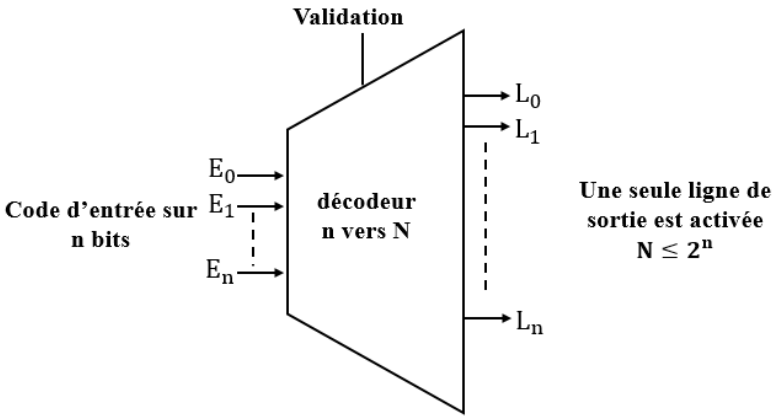
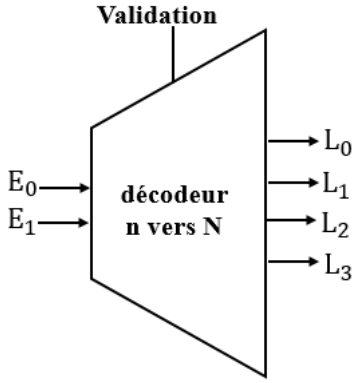


Fig. 2.3 : *schéma représentatif d'un décodeur.*

Exemple 2.2 : Faire la synthèse d'un décodeur 4 voies (2 vers 4).



1. Identification des entrées et des sorties (IN / OUT) du circuit :

- *Entrées* : 2 entrées: E_0, E_1.
- *Sorties* : 4 sorties $N = 2^n = 2^2 = L_0, L_1, L_2, L_3$.

2. Construction de la table (les tables) de vérité :

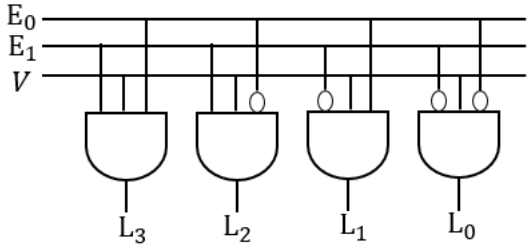
V=0					
E_1	E_0	L_3	L_2	L_1	L_0
0	0	0	0	0	0
0	1	0	0	0	0
1	0	0	0	0	0
1	1	0	0	0	0

V=1					
E_1	E_0	L_3	L_2	L_1	L_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

3. Identifier chaque fonction à partir de la table de vérité et 4. simplifier chaque fonction :

$L_0 = \overline{E}_1.\overline{E}_0.V$, $L_1 = \overline{E}_1.E_0.V$, $L_2 = E_1.\overline{E}_0.V$, $L_3 = E_1.E_0.V$

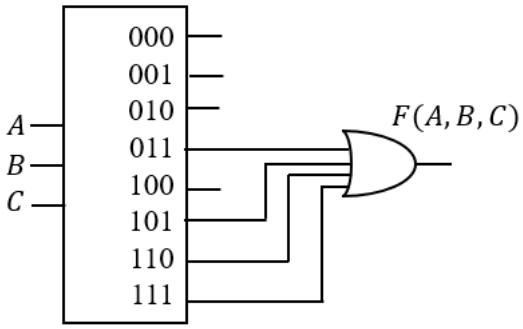
5. Dessiner le schéma du circuit :



Exemple 2.3 : Générer la fonction logique ci-dessous à l'aide d'un décodeur 3 vers 8

$$F(A,B,C) = CB\bar{A} + C\bar{B}A + \bar{C}BA + CBA$$

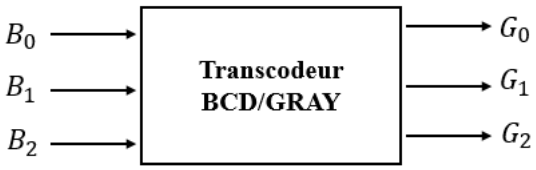
A	B	C	$F(A,B,C)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1



c. Transcodeur Binaire/Gray :

C'est un circuit combinatoire qui permet de renvoyer en sortie en langage Gray (sur n bits), le même chiffre présenté en entrée en code binaire (sur n bits).

Exemple 4 : Faire la synthèse d'un transcodeur binaire / Gray (3bits)



1. Identification des entrées et des sorties (IN / OUT) du circuit :

- *Entrées* : 3 entrées:B_0, B_1, B_2.
- *Sorties* : 3 sorties: G_0, G_1, G_2

2. Construction de la table (les tables) de vérité :

	Entrée binaire			Sortie Gray		
	B_2	B_1	B_0	G_2	G_1	G_0
0	0	0	0	0	0	0
1	0	0	1	0	0	1
2	0	1	0	0	1	1
3	0	1	1	0	1	0
4	1	0	0	1	1	0
5	1	0	1	1	1	1
6	1	1	0	1	0	1
7	1	1	1	1	0	0

3. Identifier chaque fonction à partir de la table de vérité :

$G_2 = B_2\overline{B_1}\ \overline{B_0} + B_2\overline{B_1}B_0 + B_2B_1\overline{B_0} + B_2B_1B_0.$

$G_1=\overline{B_2}B_1\overline{B_0} + \overline{B_2}B_1B_0 + B_2\overline{B_1}\ \overline{B_0} + B_2\overline{B_1}B_0$.

$G_0 = \overline{B_2}\ \overline{B_1}B_0 + \overline{B_2}B_1\overline{B_0} + B_2\overline{B_1}B_0 + B_2B_1\overline{B_0}.$

4. simplifier chaque fonction :

B_2B_1 \ B_0	0	1
00	0	0
01	0	0
11	1	1
10	1	1

$G_2 = B_2$

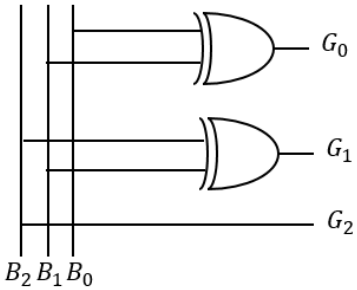
B_2B_1 \ B_0	0	1
00	0	0
01	1	1
11	0	0
10	1	1

$G_1 = B_2\overline{B_1} + \overline{B_2}B_1 = B_2 \oplus B_1$

B_2B_1 \ B_0	0	1
00	0	1
01	1	0
11	1	0
10	0	1

$G_2 = B_1\overline{B_0} + \overline{B_1}B_0 = B_1 \oplus B_0$

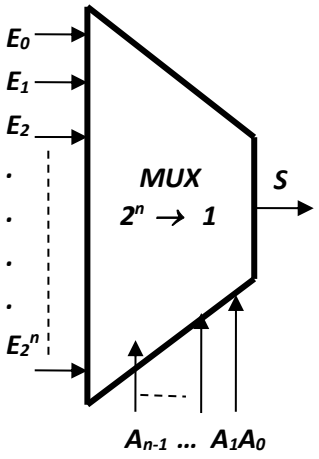
5. Dessiner le schéma du circuit :



2.2 Circuits d'aiguillage

a. Le multiplexeur (sélecteur de données) :

Le multiplexeur est un circuit combinatoire Sélecteur qui possède 2^n entrées d'information, n entrées de commande (Adresse), et une seule sortie. Son rôle consiste à sélectionner, à l'aide de signaux de commande, une des entrées et à l'envoyer à la sortie



Exemple 5 : Multiplexeur à N=2 entrées de commande

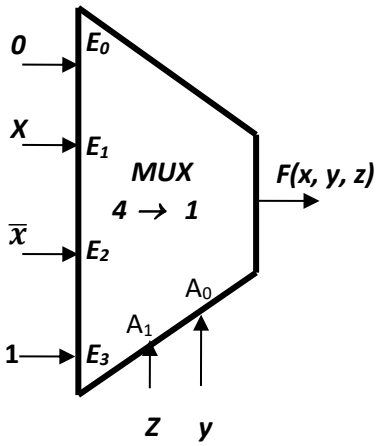
Application pratique : Les multiplexeurs ont de nombreuses applications. Ils peuvent par exemple être utilisés comme :

- Sélecteur de données.
- Convertisseur parallèle-série. Le multiplexeur reçoit en parallèle des données qu'il peut transmettre l'une après l'autre sur sa sortie.
- Générateur de fonctions logiques.
  En fait un multiplexeur à n entrées d'adresses (et donc 2^n entrées de données) peut réaliser toutes les fonctions logiques combinatoires de $n + 1$ variables.

Exemple 6 : réalisation de la fonction $F = x\,y\,\overline{z} + \overline{x}\,\overline{y}\,z + y\,z$ avec un multiplexeur à 2 entrées d'adresse.

On connecte 2 des variables aux entrées d'adresse ou de commande. Par exemple z à A_1 et y à A_0. On doit ensuite connecter convenablement les 4 entrées de données de façon à reproduire les différents termes de la fonction F.

- Pour le terme $x \bullet y \bullet \bar{z}$ on doit relier l'entrée E_1 dont l'adresse est 01 $(z = 0,\ y = 1)$ à **x**.
- Pour le terme $\bar{x} \bullet \bar{y} \bullet z$ on doit relier l'entrée E_2 dont l'adresse est 10 $(z = 1, y = 0)$ à $\bar{x}$.
- Pour le terme $z \bullet y$ on relie l'entrée E_3 dont l'adresse est 11 $(z = 1, y = 1)$ au niveau logique 1 .
- Toutes les autres entrées sont connectées au niveau logique **0** (la masse) puisque $F = 0$ pour les valeurs de y et z correspondantes.



b. Démultiplexeur :

Un démultiplexeur joue le rôle inverse du multiplexeur. Il permet de faire passer une entrée de donnée, dans l'une des sorties selon les valeurs des entrées de commandes.
Il possède :

- une entrée unique d'information (donnée)
- n entrées de commandes (adresses).
- m sorties ($m \leq 2^n$).

Application pratique : Transformation série parallèle.

Exemple 7 : Démultiplexeur à N=2 entrées de commande

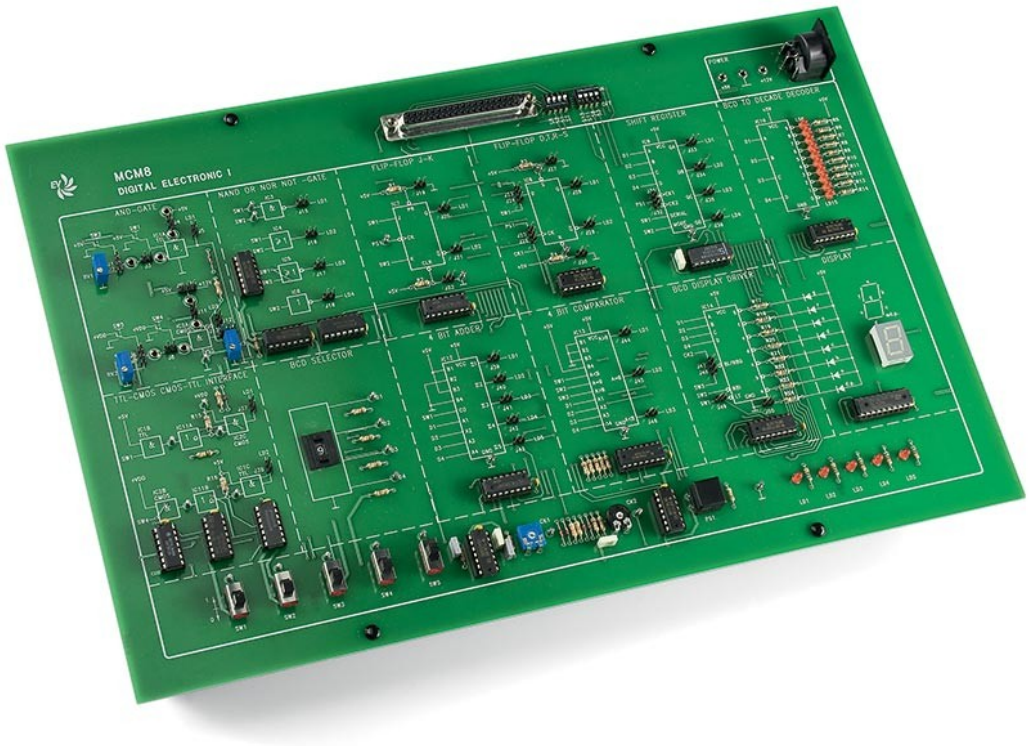
Remarque
La plupart des multiplexeurs/démultiplexeurs sont munis d'une entrée supplémentaire : entrée de validation permettant d'activer ou non le circuit. Lors de la désactivation, les sorties peuvent être placées soit à l'état inactif ("0") soit dans un état dit "haute impédance".

Biobibliographie :

[1] "Digital Logic and Computer Design" par M. Morris Mano.
[2] "Conception des circuits logiques numériques" par M. Morris Mano.
[3] "Circuit numériques : principes et pratique" par John M. Yarbrough.

CHAPITRE 3 : LOGIQUES SÉQUENTIELS.

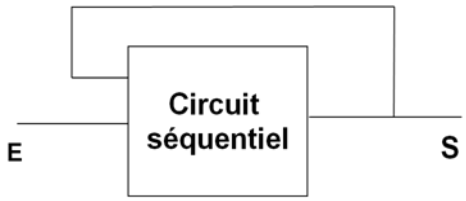
"Chaque jour d'apprentissage est un pas vers une meilleure version de soi-même, un voyage qui ne cesse jamais d'enrichir notre âme."



CHAPITRE 3 : SYSTEMES LOGIQUES SEQUENTIELS

3.1 Définition

Dans les circuits séquentiels, les signaux de sortie dépendent des entrées, mais aussi des états antérieurs : ils ont une mémoire du passé.

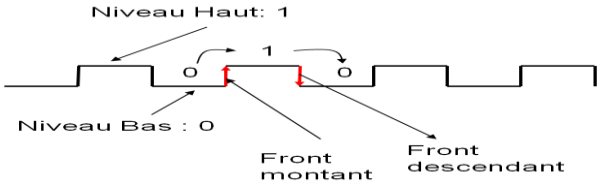


$$S_{t+1} = f(E, S_t)$$

3.2 Circuits asynchrones et synchrones

a. Horloge :

- Une horloge est une variable logique qui passe successivement de 0 à 1 et de 1 à 0 d'une façon périodique.



- L'horloge est notée par **h**, **c** ou **ck** (clock).

b. Circuits asynchrones et synchrones

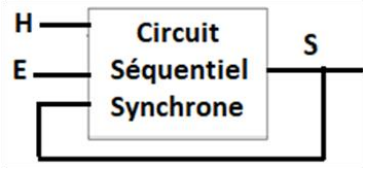
Il existe deux types de circuits séquentiels

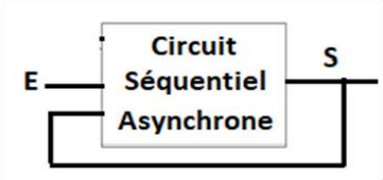
- **Systèmes asynchrones**

Quand un circuit séquentiel n'a pas d'horloge comme variable d'entrée alors il est asynchrone. Dans ce cas, les sorties réagissent immédiatement aux variations des entrées, ce qui peut provoquer des états transitoires, des retards de durées différentes et des risques d'instabilité

- **Systèmes Synchrones**:

Le système mémorise l'état présent sur son entrée si et seulement si une horloge fournit un signal de synchronisation. Les circuits synchrones sont plus simples à synthétiser et à analyser.





3.3 Bascules (Bistables) :

Les bascules sont des circuits de bases de la logique séquentiel. Une bascule est une mémoire unitaire (1 bit) ayant 2 états : **Q et $\overline{Q}$.**

Il existe plusieurs types de bascules : RS, RST, D, JK.

3.3.1 La bascule RS asynchrone

Entrée/sorties

- 2 entrées : ***R et S***
- 2 sorties ***Q et $\overline{Q}$*** qui correspond à l'état stocké et son inverse

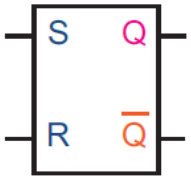
Principe : la valeur de ***Q*** à $t+1$ dépend de ***R***(reset) ***et S***(***set***) et de la valeur de ***Q*** à t

- $S = 1\ et\ R = 0$: $Q\ mis$ à 1
- $S = 0\ et\ R = 1$: $Q\ mis$ à 0
- L'état $R = S = 0$ (mode mémoire) maintient l'état de la sortie.
- L'état $R = S = 1$ (mode interdit)est interdit car il conduit à mettre simultanément la sortie à 1 et à 0.

Table de fonctionnement

R	S	Q^+	
0	0	Q^-	Mémoire
0	1	1	Mise à 1
1	0	0	Mise à 0
1	1	Φ	Interdit

Symbole



Exemple1 : Réalisation de la bascule RS asynchrone à l'aide des portes Nand.

Table de vérité

R	S	Q_n	Q_{n+1}	
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	Φ	Interdit
1	1	1	Φ	

Table de Karnaugh

Q^- \ RS	00	01	11	10
0	0	1	Φ	0
1	1	1	Φ	0

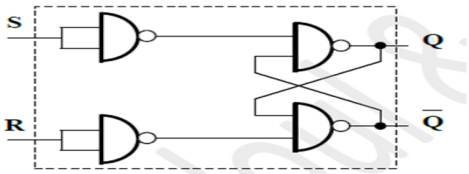
Equation logique

$$Q_{n+1} = S + Q_n\overline{R}$$

$$= \overline{\overline{S + Q_n\overline{R}}}$$

$$= \overline{\overline{S}.\overline{Q_n\overline{R}}}$$

Circuit logique



- Cette bascule RS est prioritaire au 1 car, pour la combinaison R=S=1, la sortie Q est mise à 1 (les Φ ayant été fixés à 1 pour la simplification de Q).

Exemple1 : Réalisation de la bascule RS asynchrone à l'aide des portes Nor.

Table de vérité

R	S	Q_n	Q_{n+1}	
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	Φ	Interdit
1	1	1	Φ	

Table de Karnaugh

Q_n \ RS	00	01	11	10
0	0	1	Φ	0
1	1	1	Φ	0

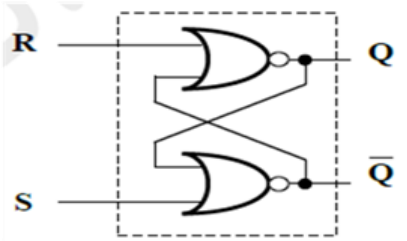
Equation logique

$$Q_{n+1} = \overline{R}.(Q_n + S)$$

$$= \overline{\overline{\overline{R}.(Q_n + S)}}$$

$$= \overline{\overline{R} + \overline{(Q_n + S)}}$$

Circuit logique



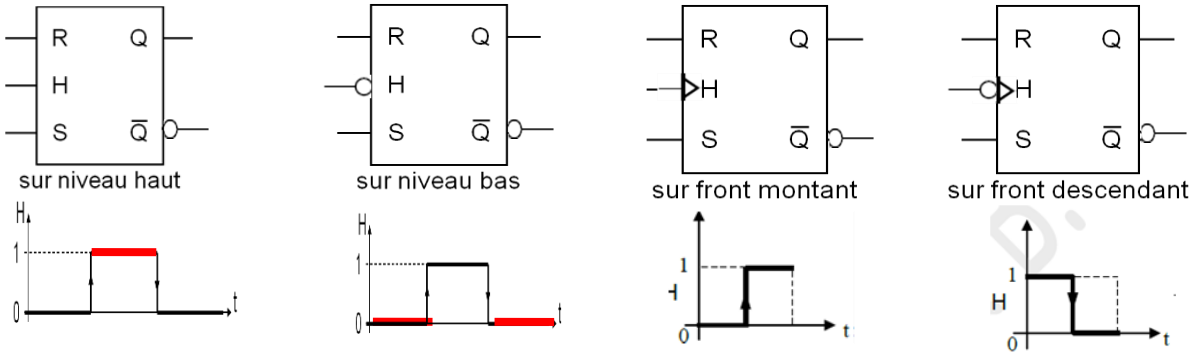
Cette bascule RS est prioritaire au 0 car, pour la combinaison R=S=1, la sortie Q est mise à 0 (les Φ ayant été fixés à 0 pour la simplification de Q).

3.3.2 La bascule RS synchrone

Pour une bascule RS synchrone l'enclenchement et le déclenchement ne sont donc autorisés qu'à la présence du signal d'horloge H.

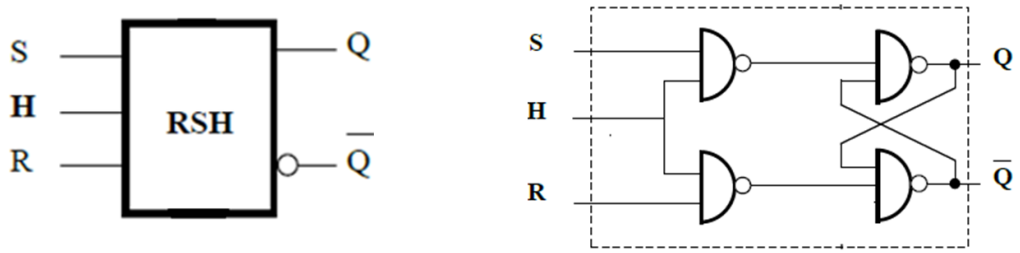
La bascule RS synchrone fonctionne en fonction de l'horloge selon 4 Modes :

- Niveau haut : la bascule n'est active que si l'horloge est à 1
- Niveau bas: la bascule n'est active que si l'horloge est à 0
- Sur le front montant de l'horloge
- Sur le front descendant de l'horloge

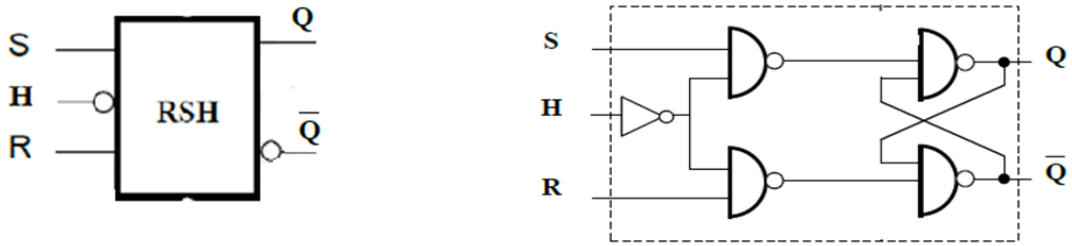


Modes de synchronisation des bascules RST

Exemple 3 : La bascule RST synchronisée par le niveau haut de l'horloge



Exemple 4: La bascule RST synchronisée par le niveau bas de l'horloge



Remarque La synchronisation sur niveau a beaucoup d'inconvénients : la bascule est sensible aux entrées pendant toute la durée de l'état de l'horloge pour niveau haut (ou 0 pour le niveau bas). Si, pendant que H =1 (ou H=0), des parasites apparaissent sur les entrées S et R, ils peuvent entrainer des changements d'état imprévus sur la sortie Q.

Afin de minimiser au maximum la durée de cet état sensible, on s'arrange pour que la bascule reste dans son état mémoire sauf pendant un bref instant, juste au moment où l'entrée passe de 0 à 1 (ou de 1 à 0). La bascule est dite synchronisée sur front.

3.3.3 Bascule J-K

La bascule JK (variante de RS) est une bascule synchrone (le plus souvent sur front) qui possède une entrée J de mise à 1 (S), une entrée K de mise à 0 (R), une entrée d'horloge H, une sortie Q et une sortie complément de Q.

La différence entre la bascule JK et la bascule RS réside dans le fait qu'il n'y a plus l'état indécis. La combinaison J=1 et K=1 est utilisée pour faire basculer la sortie, dans ce cas elle passe à l'état complémentaire : $\boldsymbol{Q^+ = \overline{Q^-}}$ ($\boldsymbol{basculement}$).

3.3.4 Bascule D

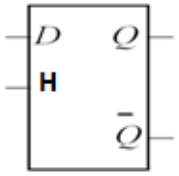
La bascule D est une bascule synchrone qui possède une entrée de donnée D (Data), une entrée d'horloge H, une sortie Q et une sortie complément de Q.

Le signal de synchronisation est actif :

- Soit sur un niveau (haut ou bas) de l'horloge (bascule D latch)
- Soit sur un front (montant ou descendant) de l'horloge (bascule D edge triggered)

Exemple 5 : Bascule ***D latch*** synchronisée par le niveau haut :

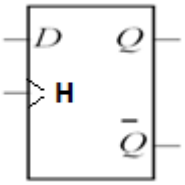
H	D	Q^+
0	0	Q^-
0	1	Q^-
1	0	0
1	1	1



La sortie recopie l'entrée sur le niveau haut d'horloge $\boldsymbol{Q^+ = D}$. Sur le niveau bas, la sortie est mémorisée

Exemple 6 : Bascule ***D*** **edge triggered** synchronisée par le front montant :

H	D	Q^+
0	0	Q^-
1	1	Q^-
↑	0	0
↑	1	1



La sortie recopie l'entrée sur un front montant d'horloge sinon elle ne change pas d'état (maintien de l'état, mémorisation).

Biobibliographie :

[1] "Digital Logic and Computer Design" par M. Morris Mano.
[2] "Digital Systems: Principles and Applications" par Ronald J. Tocci.

Fiche TD 1 : Logique combinatoire et Logique Séquentielle.

Partie 1 : Rappel des principes fondamentaux de l'algèbre de Boole

Propriétés de l'algèbre de Boole	**Théorèmes de l'algèbre de Boole**
La commutativité: A. B=B.A A+B=B+A	Théorème d'involution : $\overline{\overline{A}}$= A $\overline{\overline{A}} = \overline{A}$
L'associativité : (A.B).C=A. (B.C) (A+B)+C=A+(B+C)	Théorème d'inclusion : A.B+A.$\overline{B}$=A (A+B).(A+$\overline{B}$)=A
La priorité: A+B.C=A+ (B.C)	Théorème d'allégement : A.($\overline{A}$+B)=A.B A+($\overline{A}$.B)=A+B
La distributivité : A.(B+C) = (A.B) + (A.C) = A. B+A.C A+(B.C) = (A+B).(A+C)	Théorème d'absorption : A.(A+B) =A A+ (A.B) =A
Les éléments neutres : A.1= A A+0= A	Théorème de Morgan: $\overline{A.B} = \overline{A} + \overline{B}$ $\overline{A+B} = \overline{A}.\overline{B}$
Les éléments absorbants : A.0= 0 A+1=1	
La complémentarité : $\overline{A}$.A=0 $\overline{A}$+A=1	
L'idempotence : A. A=A A+A=A	

Operateurs élémentaires de l'algèbre de Boole

NON

Entrée	Sortie
A	F= $\overline{\overline{A}}$
0	1
1	0

OU (OR)

Les entrées		Sortie
A	B	F=A+B
0	0	0
0	1	1
1	0	1
1	1	1

ET (AND)

Les entrées		Sortie
A	B	F=A.B
0	0	0
0	1	0
1	0	0
1	1	1

NON ET (NAND)

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

NON OU (NOR)

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

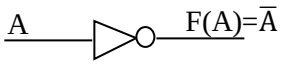
XOR : F(A,B)= A$\oplus$B= $\bar{A}$B+A$\bar{B}$= (A+B)($\bar{A}+\bar{B}$)

Les entrées		Sortie
A	B	F= A$\oplus$B
0	0	0
0	1	1
1	0	1
1	1	0

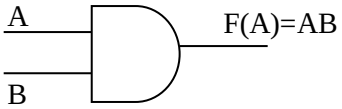
Identité : F(A,B)= A$\otimes$B= $\bar{A}\bar{B}$+AB= (A+$\bar{B}$)($\bar{A}+B$)

Les entrées		Sortie
A	B	F= A$\otimes$B
0	0	1
0	1	0
1	0	0
1	1	1

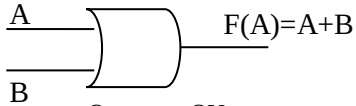
Représentation symbolique des opérateurs logique



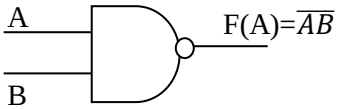
Operateur NON.



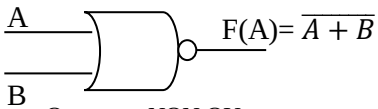
Operateur ET.



Operateur OU.

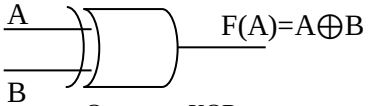


Operateur NON ET.

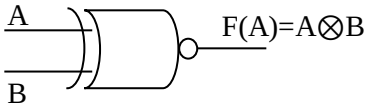


Operateur NON OU.

0



Operateur XOR.



Operateur Identité.

Exercice 1:

a. Donner le circuit logique des fonctions suivantes:

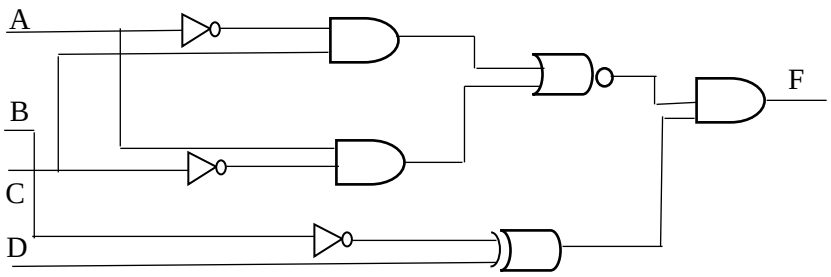
$F(A,B,C,D) = \overline{(A+B).\overline{(B+\bar{C}+D)}.A}$

$F(A,B,C) = \overline{(AB)}.(C \odot B) + A\bar{B}C$

b. Dresser la table de vérité de la fonction suivante:

$F(A,B,C) = \overline{(A.B)}.(C+B) + A\bar{B}C$

c. Donner l'équation de la fonction F réalisée par le circuit logique suivant :



Exercice 2:

A partir de la table de vérité ci-dessous :

A	B	C	F(A,B,C)
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

a. Etablir l'équation logique de la sortie F en fonction des entrées sous sa première forme canonique.
b. Etablir l'équation logique de la sortie F en fonction des entrées sous sa deuxième forme canonique.
c. En utilisant les propriétés de l'algèbre de Boole, simplifier la première forme de F puis schématiser son logigramme à l'aide des portes logiques de base.

Exercice 3:

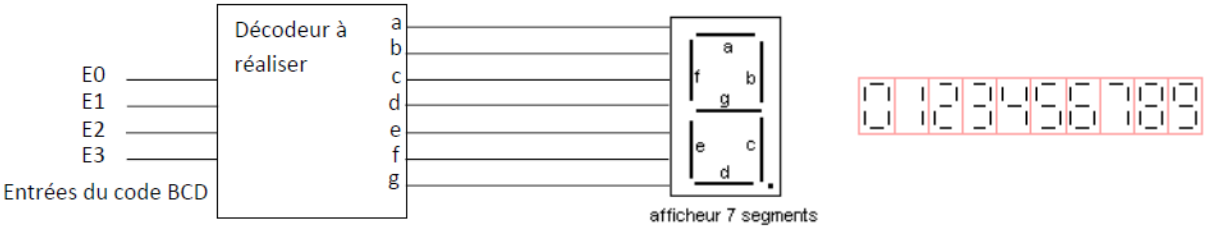
a. Donner la table de Karnaugh qui correspond à la table de vérité ci-dessous, puis utiliser la (TK) pour simplifier la fonction F.

A	B	C	F(A,B,C)
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

b. Pour chacune des fonctions logiques suivantes, établir la (TK), puis simplifier l'expression.
$F(A,B,C) = A\bar{C} + A\bar{B} + ABC + \bar{A}\bar{B}\bar{C}$
$F(A,B,C,D) = \bar{A}B\bar{C} + \bar{A}\bar{C}D + \bar{A}B\bar{D} + AC + BC\bar{D}$
$F(A,B,C,D) = (A+B).(B+\bar{C}).(A+D)$

Partie 2 : Logique combinatoire

Exercice 1



a. Dresser la table de vérité du transcodeur ***BCD/ 7*** segments. Les sorties seront considérées indéterminées (X) pour les combinaisons d'entrée non valides. Elles valent 1 quand le segment doit être allumé.

b. A l'aide du tableau de Karnaugh déterminer les expressions simplifiées des sorties ***a*** et ***d***.

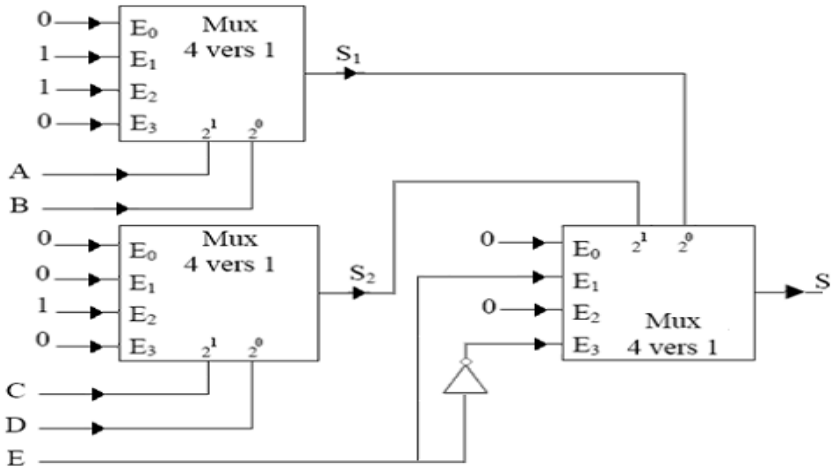
c. Proposer un logigramme pour la sortie ***a*** avec un minimum de portes logiques.

Exercice 2 :

a. Réaliser un multiplexeur à 3 entées de commande à l'aide de 2 multiplexeurs à 2 entrées d'adresse.

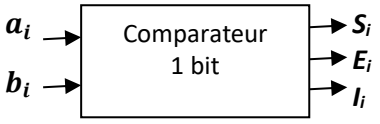
b. Soit le schéma ci-dessous : Donnez l'équation de S en fonction de: A, B, C, D, et E.

c. A l'aide d'un multiplexeur à 3 entrées d'adresses, réaliser la fonction ***d*** de l'exercice précédent (transcodeur ***BCD/ 7*** segments)



Exercice 3:

La figure suivante représente un comparateur de deux nombres binaires a_i et b_i à 1 bit



$$\begin{cases} S_i = 1 \; si \; a_i > b_i \\ E_i = 1 \; si \; a_i = b_i \\ I_i = 1 \; si \; a_i < b_i \end{cases}$$

a. Effectuer la synthèse de ce circuit logique.

b. On prend deux nombres de deux bits chacun en entrée, soit a_1a_0 et b_1b_0 , déduire à partir de **a** les équations des trois sorties de ce comparateur 2 bits.

Partie 3 : logique séquentielle

Exercice 1

a. Rappeler la table de vérité d'une bascule RS.

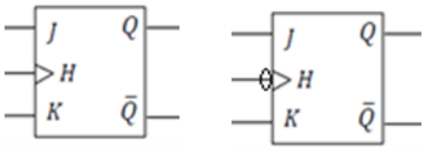
b. Proposer un circuit logique d'une bascule RS asynchrone(vérifier à partir du circuit les deux fonction Q et $\overline{Q}$ a l'instant t+1), ensuite synchrone sur niveau haut de l'horloge en utilisant des portes NAND.

c. Proposer un circuit logique d'une bascule RS asynchrone(vérifier à partir du circuit les deux fonction Qet $\overline{Q}$ a l'instant t+1), ensuite synchrone sur niveau haut de l'horloge en utilisant des portes NOR.

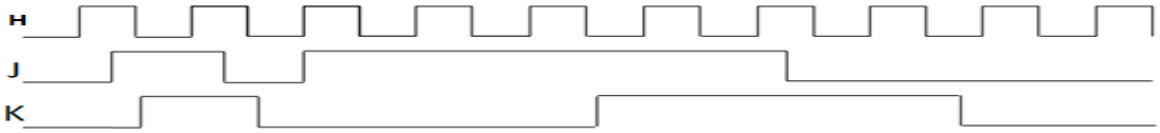
d. Rappeler la table de vérité d'une bascule JK, et proposer ensuite une réalisation de JK à l'aide d'une bascule SR.

Exercice 2

Soient les 2 bascules JK suivantes :

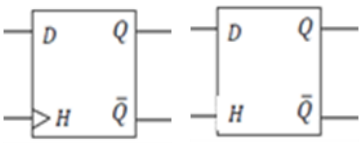


Déterminer La forme d'onde des sorties Q de ces 2 bascules quand on leur applique les entrées illustrées sur la figure ci-dessous (Q initial=0)

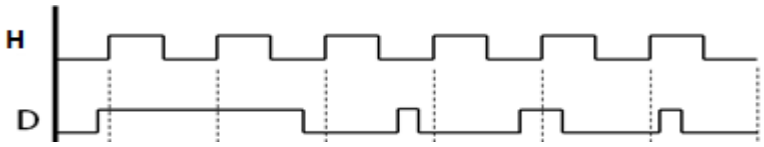


Exercice 3

Soient les 2 bascules D suivantes :



a. Après avoir rappelé la table de vérité d'une bascule D, déterminer La forme d'onde des sorties Q de ces 2 bascules quand on leur applique les entrées illustrées sur la figure ci-dessous (Q initial=0)



b. Comment peut-on réaliser une bascule D à partir d'une bascule JK

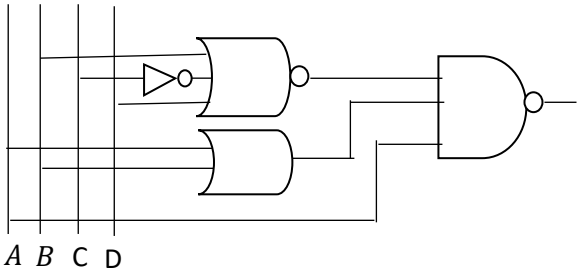
Solution Fiche TD 1: Logique combinatoire et Logique Séquentielle.

Partie 1:

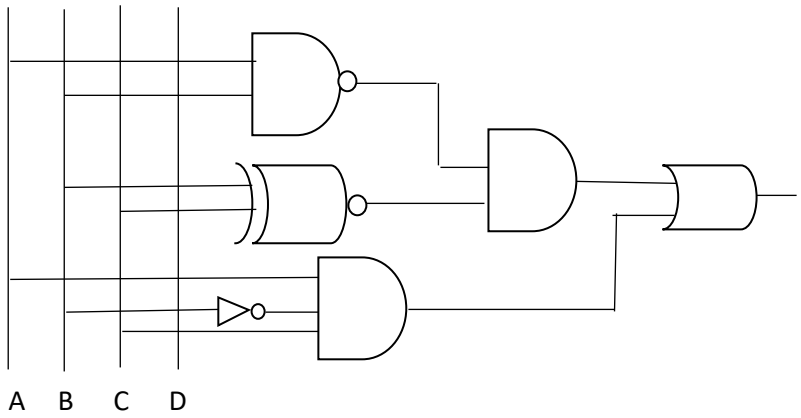
Exercice 1:

a. Donner le circuit logique des fonctions suivantes:

$$F(A,B,C,D) = \overline{(A+B).\overline{(B+\bar{C}+D)}.A}$$



$$F(A,B,C) = \overline{(AB)}.(C \odot B) + A\bar{B}C$$

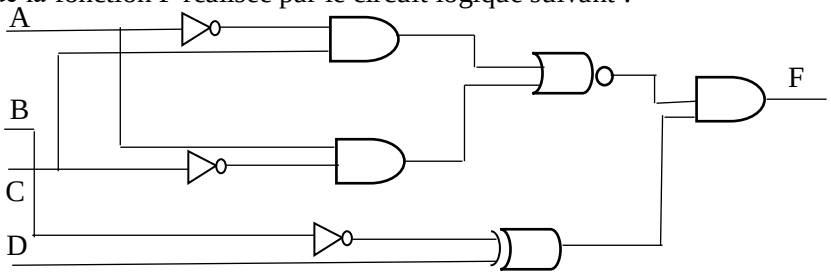


b. Dresser la table de vérité de la fonction suivante:

$$F(A,B,C) = \overline{(A.B)}.(C+B) + A\bar{B}C$$

A	B	C	F(A,B,C)
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

c. Donner l'équation de la fonction F réalisée par le circuit logique suivant :



$$\boldsymbol{F(A,B,C,D)} = \left(\overline{\boldsymbol{\bar{A}C} + \boldsymbol{A\bar{C}}}\right)(\overline{\boldsymbol{B}} \oplus \mathrm{D}) = (\overline{\boldsymbol{A} \oplus \mathrm{C}})(\overline{\boldsymbol{B}} \oplus \mathrm{D}) = (\boldsymbol{A} \otimes \mathrm{C})(\overline{\boldsymbol{B}} \oplus \mathrm{D})$$

Exercice 2 :

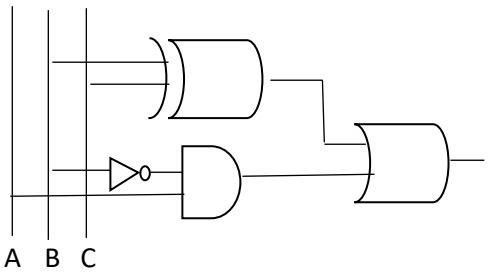
A partir de la table de vérité ci-dessous :

A	B	C	F(A,B,C)
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

a. Etablir l'équation logique de la sortie F en fonction des entrées sous sa première forme canonique.
$F_1(A,B,C)=\bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + A\bar{B}C + AB\bar{C}$

b. Etablir l'équation logique de la sortie F en fonction des entrées sous sa deuxième forme canonique.
$F_2(A,B,C)=(A+B+C)\ (A+\bar{B}+\bar{C})\ (\bar{A}+\bar{B}+\bar{C})$

c. En utilisant les propriétés de l'algèbre de Boole, simplifier la première forme de F puis schématiser son logigramme à l'aide des portes logiques de base.
$F_1(A,B,C)=\bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + A\bar{B}C + AB\bar{C}$
$= B\bar{C}(\bar{A}+A) + A\bar{B}(\bar{C}+C) + \bar{A}\bar{B}C$
$= B\bar{C} + \bar{B}(A + \bar{A}C) = B\bar{C} + \bar{B}(A+C) = B\bar{C} + \bar{B}A + \bar{B}C = \bar{B}A + (B \oplus C)$



Exercice 3 :

c. Donner la table de Karnaugh qui correspond à la table de vérité ci-dessous, puis utiliser la (TK) pour simplifier la fonction F.

A	B	C	F(A,B,C)
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

C / **AB**	**0**	**1**
00	0	1
01	1	0
11	1	0
10	1	1

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

d. Pour chacune des fonctions logiques suivantes, établir la (TK), puis simplifier l'expression.
$F(A,B,C) = A\bar{C} + A\bar{B} + ABC + \bar{A}\bar{B}\bar{C}$

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

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

$F(A,B,C,D) = \bar{A}B\bar{C} + \bar{A}\bar{C}D + \bar{A}\bar{B}\bar{D} + AC + BC\bar{D}$

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

$$F(A,B,C,D) = \bar{A}\bar{C} + \bar{A}\bar{D} + BC + BD$$

$F(A,B,C,D) = (A+B).(B+\bar{C}).(A+D)$

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

$$F(A,B,C,D) = (A+B).(B+\bar{C}).(A+D)$$

Partie 2:

Exercice 1:

a. Dresser la table de vérité du transcodeur ***BCD/ 7*** segments. Les sorties seront considérées indéterminées (X) pour les combinaisons d'entrée non valides. Elles valent 1 quand le segment doit être allumé.

E_0	E_1	E_2	E_3	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ
1	0	1	1	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ
1	1	0	0	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ
1	1	0	1	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ
1	1	1	0	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ
1	1	1	1	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ

b. A l'aide du tableau de Karnaugh déterminer les expressions simplifiées des sorties ***a*** et ***d***.

E_0 E_1 \ E_2 E_3	00	01	11	10
00	1	0	1	1
01	0	1	1	1
11	ϕ	ϕ	ϕ	ϕ
10	1	1	ϕ	ϕ

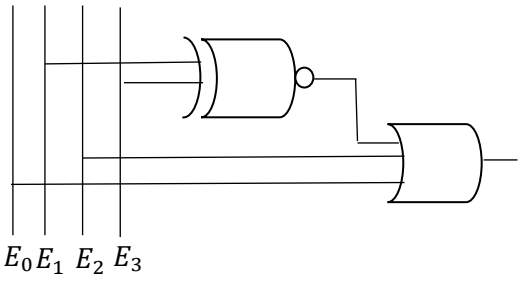
$$a = E_2 + E_0 + E_1E_3 + \overline{E_1}\ \overline{E_3}$$

$E_0\ E_1$ \ $E_2\ E_3$	00	01	11	10
00	1	0	1	1
01	0	1	0	1
11	ϕ	ϕ	ϕ	ϕ
10	1	1	ϕ	ϕ

$$d = E_0 + E_2\overline{E_3} + \overline{E_1}\ E_2 + \overline{E_1}\ \overline{E_3} + E_1\overline{E_2}\ E_3$$

c. Proposer un logigramme pour la sortie ***a*** avec un minimum de portes logiques.

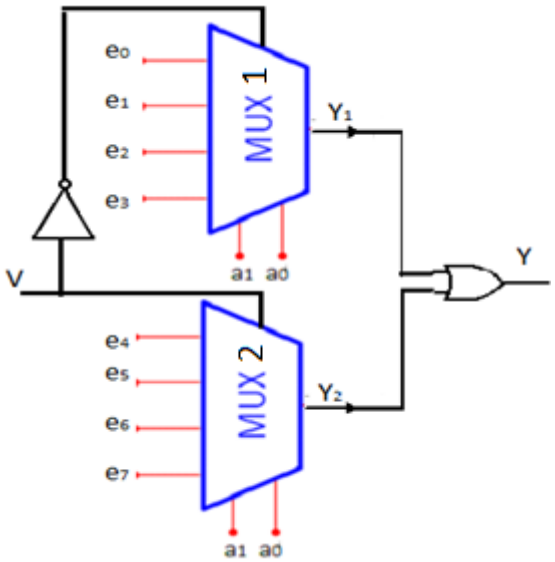
$$a = E_2 + E_0 + E_1E_3 + \overline{E_1}\ \overline{E_3} = E_2 + E_0 + E_1 \otimes E_3$$



Exercice 2:

a. Réaliser un multiplexeur à 3 entées de commande à l'aide de 2 multiplexeurs à 2 entrées d'adresse. On utilise l'entrée de validation ***V*** (Strobe) comme troisième adresse a_2 :

$$Quand\ V = 0\ le\ \ MUX1\ \ est\ activé\ \ (MUX2\ est\ desactivé)\ et\ Y = Y_1$$
$$Quand\ V = 1\ le\ \ MUX2\ \ est\ activé\ \ (MUX1\ est\ desactivé)\ \ et\ Y = Y_2$$



a_2=V	a_1	a_0	Y
0	0	0	e_0
0	0	1	e_1
0	1	0	e_2
0	1	1	e_3
1	0	0	e_4
1	0	1	e_5
1	1	0	e_6
1	1	1	e_7

b. Soit le schéma ci-contre : Donnez l'équation de S en fonction de A, B, C, D, et E .

$S_1 = \overline{A}.B + A\overline{B} = A \oplus B$

$S_2 = C\overline{D}$

$S = \overline{S_2}S_1.E + S_2S_1\overline{E} = S_1(S_2 \oplus E) = A \oplus B(C\overline{D} \oplus E)$

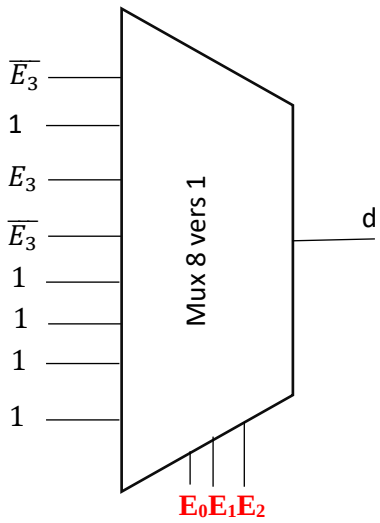
c. A l'aide d'un multiplexeur à 3 entrées d'adresses, réaliser la fonction ***d*** de **l'exercice 1 partie 2** (transcodeur ***BCD/ 7*** segments)

c.1) Table de vérité :

E_0	E_1	E_2	E_3	d
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	ϕ
1	0	1	1	ϕ
1	1	0	0	ϕ
1	1	0	1	ϕ
1	1	1	0	ϕ
1	1	1	1	ϕ

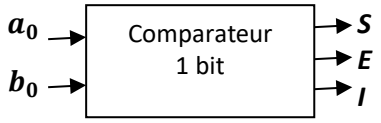
c.2) On prend $E_0E_1E_2$ (l'ordre est important) comme entées d'adresse et E_3 comme entrée de donnée, puis en utilisant la table de vérité on compare l'entrée E_3 avec la sortie d, et on obtient :

Les cas indéterminés sont considérés comme des 1 au moment de la simplification, donc on les met a 1 à l'entrée du Multiplexeur.



Exercice 3:

La figure suivante représente un comparateur de deux nombres binaires $\boldsymbol{a_i}$ $\boldsymbol{et}$ $\boldsymbol{b_i}$ à 1 bit



$$\begin{cases} S = 1\ si\ a_0 > b_0 \\ E = 1\ si\ a_0 = b_0 \\ I = 1\ si\ a_0 < b_0 \end{cases}$$

a. Effectuer la synthèse de ce circuit logique.

a.1) les entrées : a_0 b_0, les sorties S, E, I

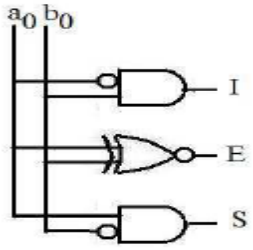
a.2) Table de vérité :

a_0	b_0	S	E	I
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0

a.3) Fonction logique :

S= $a_0\overline{b_0}$ E=$\overline{a_0}\,\overline{b_0} + a_0 b_0 = a_0 \otimes b_0$ I=$\overline{a_0} b_0$

a.4) Logigramme:



b. On prend deux nombres de deux bits chacun en entrée, soit $\boldsymbol{a_1 a_0}$ $\boldsymbol{et}$ $\boldsymbol{b_1 b_0}$ Donner les équations des trois sorties de ce comparateur 2 bits.

$a_1 a_0 = b_{01} b_1\ signifie: a_1 = b_1\ et\ a_0 = b_0\ d'ou\ E = (a_1 \otimes b_1)(a_0 \otimes b_0)$
$a_1 a_0 < b_1 b_0\ signifie: a_1 < b_1\ ou\ (a_1 = b_1\ et\ a_0 < b_0)\ d'ou\ I = \overline{a_1} b_1 + ((a_1 \otimes b_1)\overline{a_0} b_0)$
$a_1 a_0 > b_1 b_0\ signifie: a_1 > b_1\ ou\ (a_1 = b_1\ et\ a_0 > b_0)\ d'ou\ S =$
$= a_1\overline{b_1} + ((a_1 \otimes b_1) a_0 \overline{b_0})$

Partie 3:

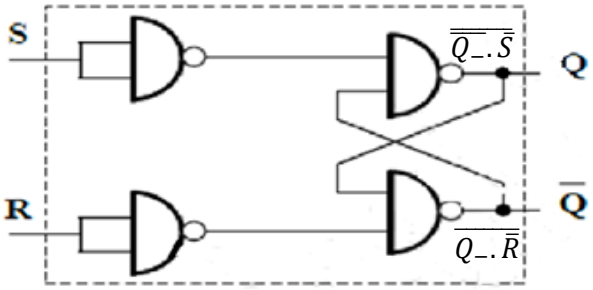
Exercice 1 :

a. Rappeler la table de vérité d'une bascule RS.

R	S	Q_-	Q_+
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	ϕ
1	1	1	ϕ

b. Proposer un circuit logique d'une bascule RS asynchrone(vérifier à partir du circuit les deux fonction Q et $\overline{Q}$ a l'instant t+1), ensuite synchrone sur niveau haut de l'horloge en utilisant des portes NAND.

b.1) Circuit logique d'une bascule RS asynchrone en utilisant des portes NAND.



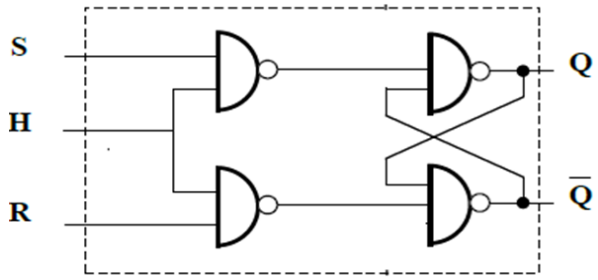
A partir du circuit on peut facilement déduire les équations Q_+et $\overline{Q_+}$:

$Q_+ = \overline{\overline{S}.\overline{Q_-.\overline{R}}}$ (Fonction vérifiée)

$\overline{Q_+} = \overline{\overline{\overline{Q_-}.\overline{S}}.\overline{R}}$ À vérifier avec la fonction obtenue à partir de la table de vérité : $\overline{Q_+} = \overline{S}.\overline{Q_-.\overline{R}}$

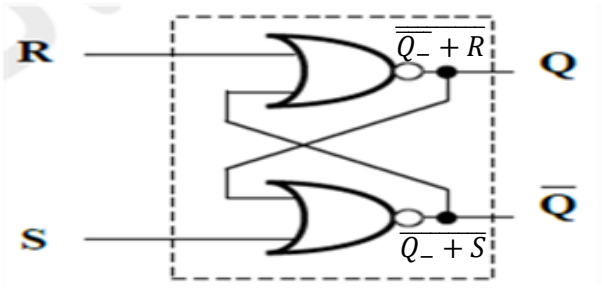
$\overline{Q_+} = \overline{\overline{\overline{Q_-}.\overline{S}}.\overline{R}} = \overline{S}.\overline{Q_-} + R = \overline{\overline{S}.\overline{Q_-} + R} = \overline{S}.\overline{Q_-.\overline{R}}$

b.2) Circuit logique d'une bascule RS synchrone sur niveau haut de l'horloge en utilisant des portes NAND



c. Proposer un circuit logique d'une bascule RS asynchrone(vérifier à partir du circuit les deux fonction Qet $\overline{Q}$ a l'instant t+1), ensuite synchrone sur niveau haut de l'horloge en utilisant des portes NOR.

c.1) Circuit logique d'une bascule RS asynchrone en utilisant des portes NOR



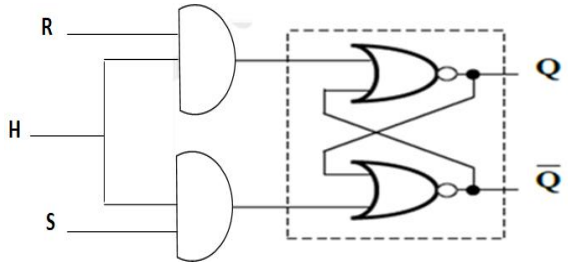
A partir du circuit on peut facilement déduire les équations Q_+et $\overline{Q_+}$:

$Q_+ = \overline{R + (\overline{Q_- + S})}$

À vérifier avec la fonction obtenue à partir de la table de vérité : $\overline{Q_+} = R + (\overline{Q_- + S})$

$\overline{Q_+} = \overline{\overline{R + \overline{Q_-}} + S} = R + \overline{Q_-}.\bar{S} = \overline{\overline{R + \overline{Q_-}.\bar{S}}} = R + (\overline{Q_- + S})$

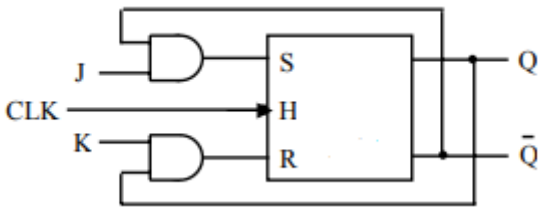
c.2) Circuit logique d'une bascule RS synchrone sur niveau haut de l'horloge en utilisant des portes NOR



d. Rappeler la table de vérité d'une bascule JK, et proposer ensuite une réalisation de JK à l'aide d'une bascule SR.
d.1) Rappeler la table de vérité d'une bascule JK

J	K	Q_-	Q_+
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

d.2) Proposer ensuite une réalisation de JK à l'aide d'une bascule SR.



Le fonctionnement de la bascule JK à partir de SR est comme suit :

1. Si **J = K = 0** Les entrées S et R sont à 0. La bascule RS garde l'état précédent des sorties Q_-et $\overline{Q_-}$.
2. Si **J = 0 et K = 1** L'entrée S est bloquée à 0, tandis que l'entrée R varie selon la valeur de Q_-.
   - Si **Q_- = 0**, l'entrée R =0 les sorties gardent les états précédents.
   - Si **Q_- = 1**, l'entrée R = 1, la sortie Q_+ est mise à 0.

Donc quel que soit la valeur de Q_-, la sortie Q_+ est mise à 0 quand J = 0 et K = 1.

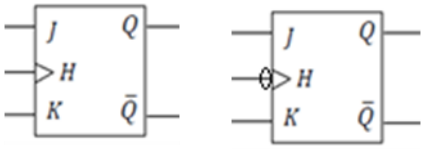
3. Si **J = 1 et K = 0** L'entrée R est bloquée à 0, tandis que l'entrée J varie selon la valeur de $\overline{Q_-}$.
   - Si $\overline{Q_-}$ **= 0**, l'entrée S =0 les sorties gardent les états précédents (Q_- = 1).
   - Si $\overline{Q_-}$ **=1**, l'entrée S = 1, la sortie Q_+ est mise à 1.

Donc quel que soit la valeur de $\overline{Q_-}$, la sortie Q_+ est mise à 1 quand J = 1 et K = 0.

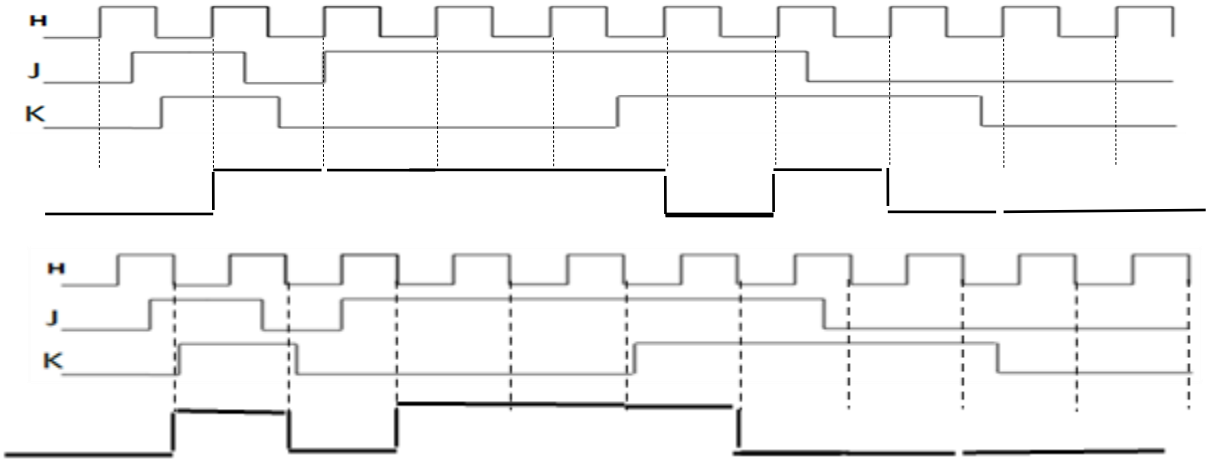
4. Si **J = K = 1** Dans ce cas les retours croisés des sorties Q_- et $\overline{Q_-}$ donne lieu un état de basculement.
   - Si (Q_- **= 0 donc** $\overline{Q_-}$ **= 1**), Au top d'horloge Q_+ =1 et $\overline{Q_+}$ =0.
   - Si (Q_- **= 1 donc** $\overline{Q_-}$ **= 0**), Au top d'horloge Q_+ =0 et $\overline{Q_+}$ =1.

Exercice 2:

Soient les 2 bascules JK suivantes :

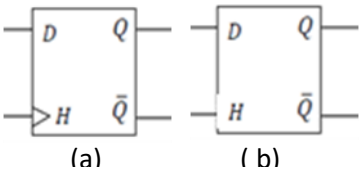


Déterminer La forme d'onde des sorties Q de ces 2 bascules quand on leur applique les entrées illustrées sur la figure ci-dessous (Q initial=0)



Exercice 3 :

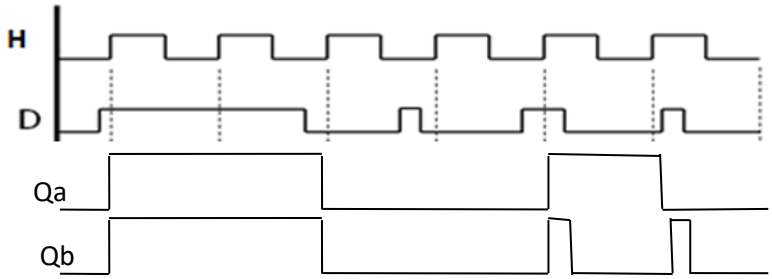
Soient les 2 bascules D suivantes :



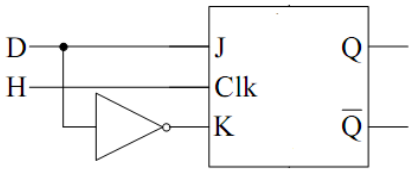
a. Après avoir rappelé la table de vérité d'une bascule D, déterminer La forme d'onde des sorties Q de ces 2 bascules quand on leur applique les entrées illustrées sur la figure ci-dessous (Q initial=0)

D (t)	Q(t + 1)
0	0
1	1

$$Q(t+1) = D(t)$$



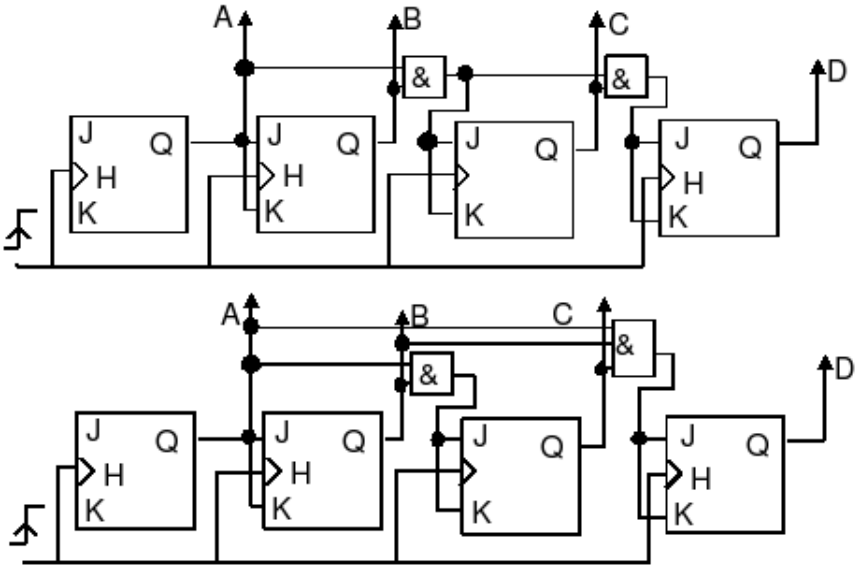
c. On peut réaliser une bascule D à partir d'une bascule JK en envoyant une donnée D sur l'entrée ***J*** et son inverse sur l'entrée ***K***



1. Si **D=1,** l'entrée J=1 et l'entrée K=0, au Top d'horloge Q_+ est mise à 1.
2. Si **D=0,** l'entrée J=0 et l'entrée K=1, au Top d'horloge Q_+ est mise à 0.

CHAPITRE 4 : COURS COMPTEURS/DÉCOMPTEURS.

"Apprendre, c'est faire grandir en soi une soif insatiable de savoir, une quête qui n'a ni début ni fin."



CHAPITRE 4 : COURS COMPTEURS/DECOMPTEURS

4.1 Définitions :

- La réalisation d'un compteur/décompteur nécessite un ensemble de bascules (cellules élémentaires reliées) cadencées au rythme d'une horloge.
- Les bascules considérées pour la conception d'un compteur/décompteur peuvent être généralement de trois natures : bascule JK (utiliser la propriété de basculement quand J = K = 1), bascule D (sortie $\bar{Q}$ rebouclée sur l'entrée D) ou bascule T (mise à 1).
- Chaque bit de comptage est associé à l'état d'une bascule (commutant, ou pas), la concaténation de l'ensemble des états produira un mot binaire particulier qui définira l'état du compteur/décompteur. Le passage d'un état à l'autre du système (hors initialisation) est conditionné par les fronts actifs (montant ou descendant) du signal d'horloge.
- Un compteur modulo $M = 2^n$ est un compteur dont la capacité de comptage est comprise entre $[0, M-1]$ ($[M-1, 0]$ pour un décompteur modulo $M = 2^n$), sa réalisation nécessite par conséquent "n" bascules. Tout de même il est possible de concevoir un compteur/décompteur dont le cycle est modifiable, on parle à ce moment des compteurs/décompteurs programmables.
- Il existe principalement deux types de compteurs/décompteurs : les compteurs/décompteurs asynchrones et les compteurs/décompteurs synchrones. On note que les compteurs/décompteurs sont tous des circuits logiques synchrones, du moment où le passage d'un état à l'autre est conditionné par un top d'horloge. En effet, les deux termes synchrone et asynchrone ont un sens distinct que nous allons éclaircir dans la suite de ce cours.

4.2 Les compteurs/décompteurs Asynchrones :

Le principe de fonctionnement est celui de la mise au point d'une cascade de diviseurs de fréquence basculant sur fronts. On distingue ici deux catégories, celle des compteurs/décompteurs asynchrones binaires complet et celle des compteurs/décompteurs binaires asynchrones incomplets.

4.2.1 Compteurs asynchrones modulo 2^n (compteur binaire complet) :

- Les bascules JK fonctionnent toutes en mode basculement (J=K=1)
- La première bascule (poids le plus faible) prend comme 3eme entrée l'horloge qui l'active ; l'une de ces sorties (Q ou $\bar{Q}$ selon le front d'activation de l'horloge) servira par la suite d'horloge à la bascule qui la succède et ce processus continue jusqu'à la n^{ième} bascule. (Voir figure 1).

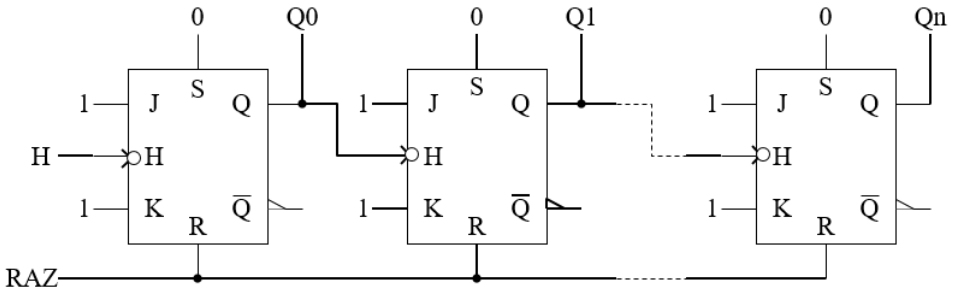


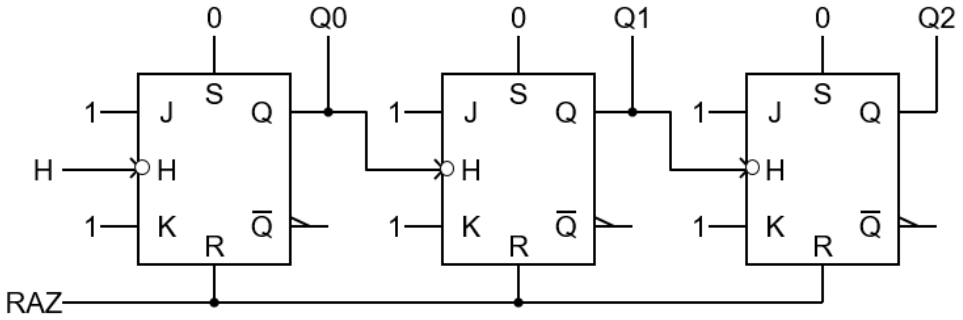
Fig. 1 : *Compteurs asynchrones modulo 2^n à base de bascules JK actives sur front descendant.*

Exemple 1 : synthétiser un compteur asynchrone modulo $2^3 = 8$ active sur front descendant, et réalisé avec des bascules JK.

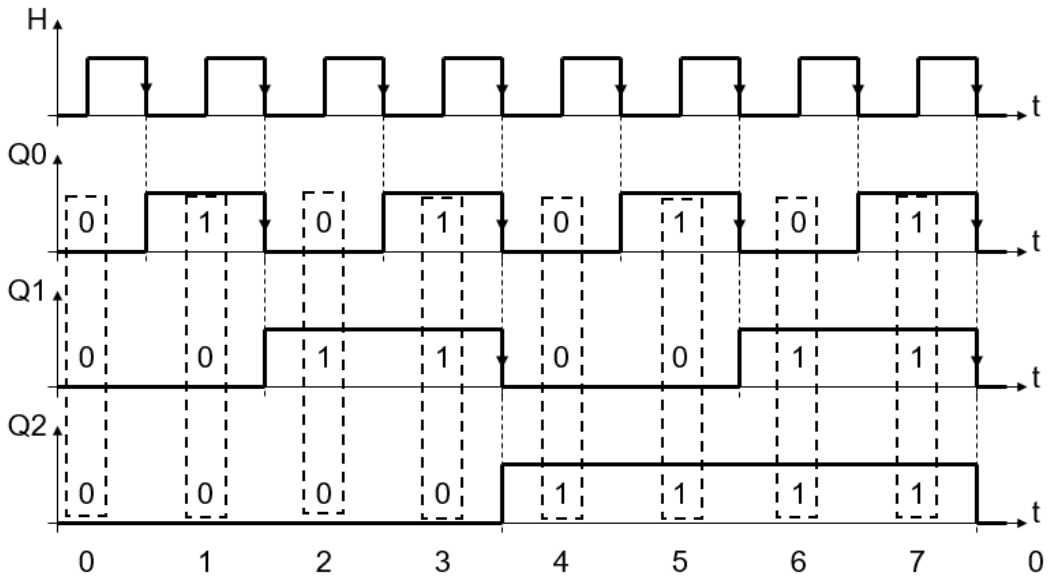
Table de vérité :

R	H	q_0	q_1	q_2	Q_0	Q_1	Q_2
1	X	x	x	x	0	0	0
0	↓	0	0	1	0	0	1
0	↓	0	1	0	0	1	0
0	↓	0	1	1	0	1	1
0	↓	1	0	0	1	0	0
0	↓	1	0	1	1	0	1
0	↓	1	1	0	1	1	0
0	↓	0	0	1	1	1	1

Logigramme :



Chronogramme :



N.B :
- Pour un compteur à activation sur front montant, on connecte la sortie $\overline{Q_n}$ à l'entrée de l'horloge H_{n+1}.
- Entrées de forçage asynchrones R : l'entrée CLEAR ou remise à zéro de l'état du compteur (active R=1 et inhibée R=0).
- Entrées de forçage asynchrones S : L'entrée PRESET ou remise à un de l'état du compteur (active S=1 et inhibée S=0).

4.2.2 Compteurs binaires asynchrones incomplets :

On a régulièrement besoin d'un compteur modulo M (ou $M \neq 2^n$), par exemple dans le cas où on veut dénombrer jusqu'à 12. Dans ce cas il est nécessaire de considérer un compteur asynchrone modulo $2n \geq M$ et d'interrompre le cycle de comptage en provoquant une rétroaction sur les entrées CLEAR, au top d'horloge qui succède la valeur maximale à afficher (M-1).

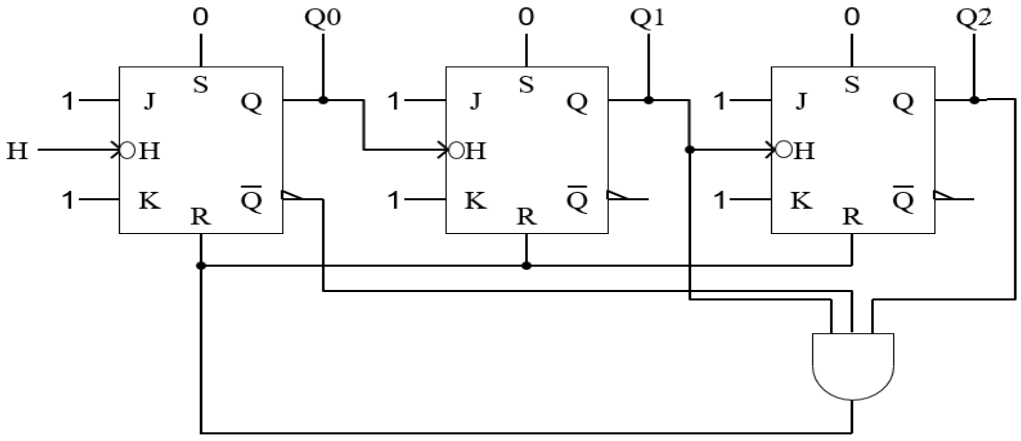
Exemple 2 : synthétiser un compteur asynchrone modulo 6 (compte donc de : 0→5) active sur front descendant, et réalisé à l'aide de bascules JK.

- Etape 1 : réaliser un compteur asynchrone binaire modulo 8
- Etape 2 : forcer la mise à 0 au moment que le compteur voudrait passer à la valeur 6.

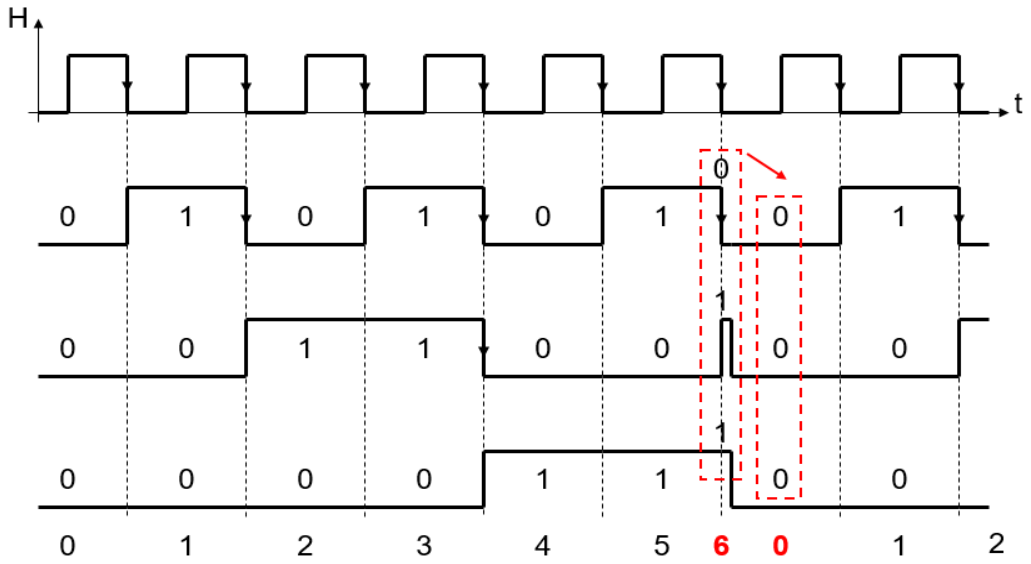
Table de vérité :

R	H	q_2	q_1	q_0	Q_2	Q_1	Q_0
1	x	x	x	x	0	0	0
0	↓	0	0	0	0	0	1
0	↓	0	0	1	0	1	0
0	↓	0	1	0	0	1	1
0	↓	0	1	1	1	0	0
0	↓	1	0	0	1	0	1
1	↓	1	0	1	**0**	**0**	**0**
					Forcer une mise à 0		

Logigramme :



Chronogramme :



4.2.3 Décompteurs asynchrones modulo 2^n (décompteur binaire complet) :

Le même principe est appliqué pour le décompteur, la bascule du poids faible a comme 3eme entrée l'horloge qui l'active ; l'une de ces sorties (Q ou $\bar{Q}$ selon le front d'activation de l'horloge) servira comme horloge à la bascule qui la succède et ce processus continue jusqu'à la nième bascule. (Voir figure 2).

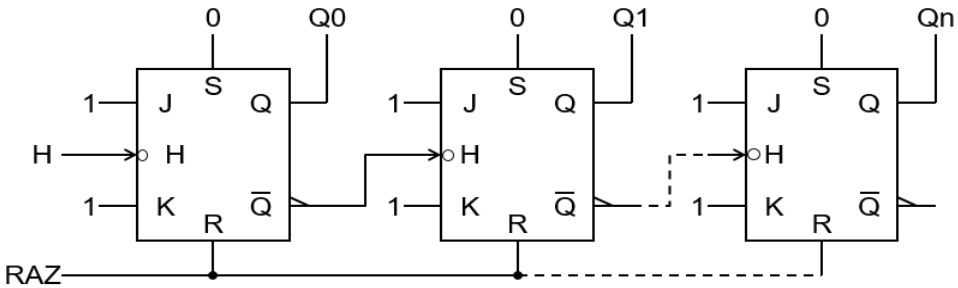


Fig. 2 : *Décompteurs asynchrones modulo 2^n à base de bascules JK active sur front descendant.*

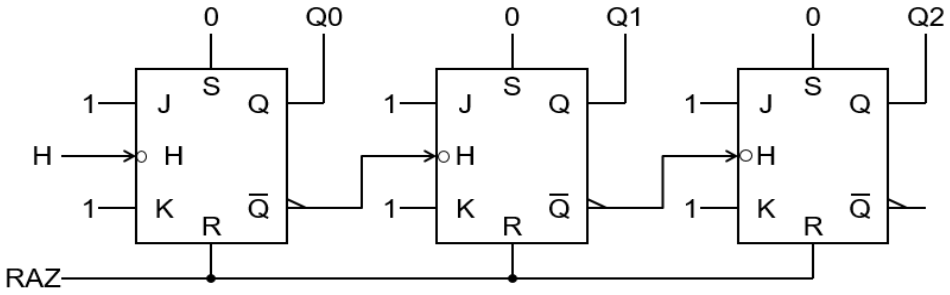
N.B :
- Pour un décompteur à activation sur front montant, on connecte la sortie Q_i à l'entrée de l'horloge H_{i+1} $(i\ allant\ de\ 0\ à\ n-1)$.

Exemple 2 : synthétiser un décompteur asynchrone modulo $2^3 = 8$ active sur front descendant et réalisé avec des bascules JK.

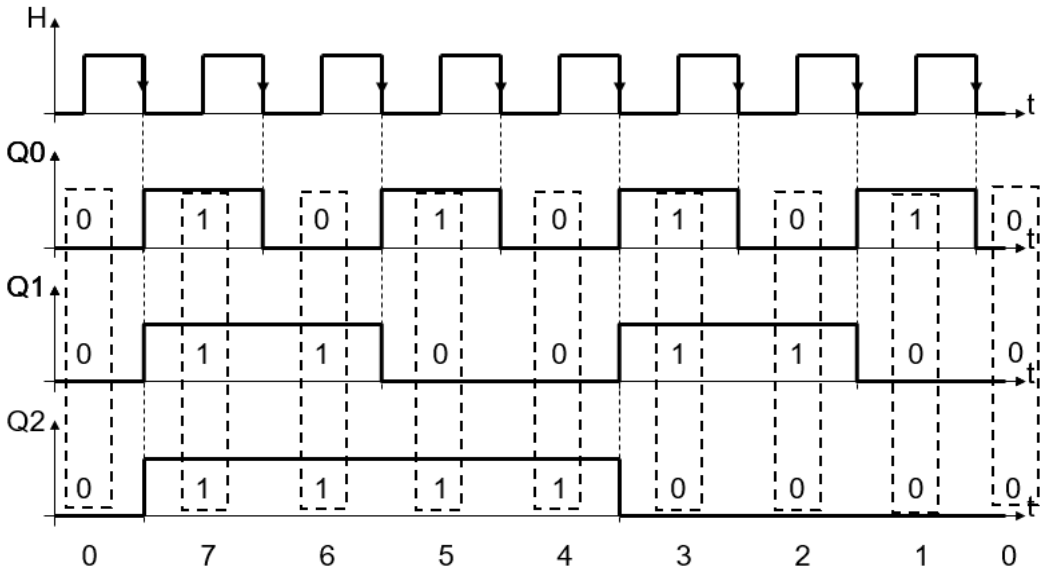
Table de vérité :

R	H	q_0	q_1	q_2	Q_0	Q_1	Q_2
1	x	x	X	x	0	0	0
0	↓	0	0	0	1	1	1
0	↓	1	1	1	1	1	0
0	↓	1	1	0	1	0	1
0	↓	1	0	1	1	0	0
0	↓	1	0	0	0	1	1
0	↓	0	1	1	0	1	0
0	↓	0	1	0	0	0	1
0	↓	0	0	1	0	0	0

Logigramme :



Chronogramme :



4.2.4 Décompteurs binaires asynchrones incomplets :

Il est aussi possible de proposer une configuration pour un cycle de décomptage incomplet, il suffit de forcer le cycle de décomptage à s'initialiser à la valeur maximale souhaitée ; en provoquant cette fois-ci une rétroaction sur les entrées CLEAR et /ou sur les entrées PRESET, au top d'horloge qui succède l'affichage de la valeur finale du cycle (0).

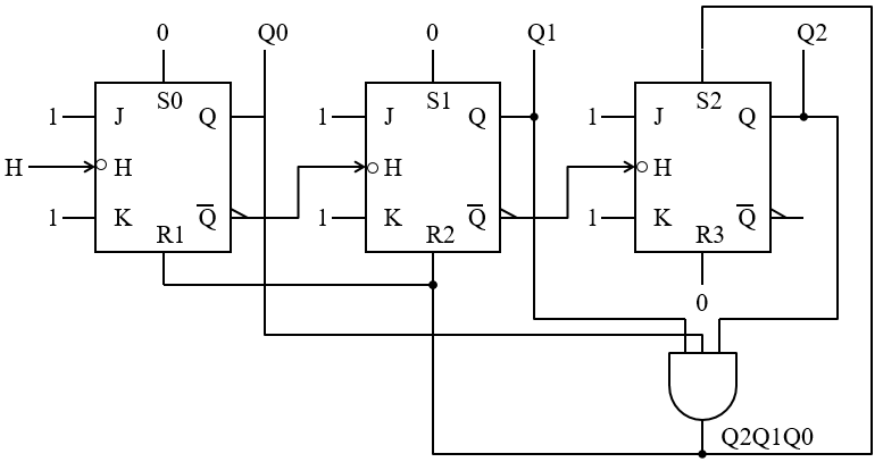
Exemple 4 : synthétiser un décompteur asynchrone modulo 5 (4→0) active sur front descendant et réalisé avec des bascules JK.

- Etape 1 : réaliser un décompteur asynchrone binaire modulo 8
- Etape 2 : forcer la mise à 4 ($Q_0 = 0\ et\ Q_1 = 0\ et\ Q_2 = 1$) au moment où le compteur voudra passer à la valeur 7 ($Q_0 = 1\ et\ Q_1 = 1\ et\ Q_2 = 1$). Pour cela on devrait faire appel aux deux touches PRESET ($S_0 = 0\ et\ S_1 = 0\ et\ S_2 = 1$) et CLEAR ($R_0 = 1\ et\ R_1 = 1\ et\ R_2 = 0$).

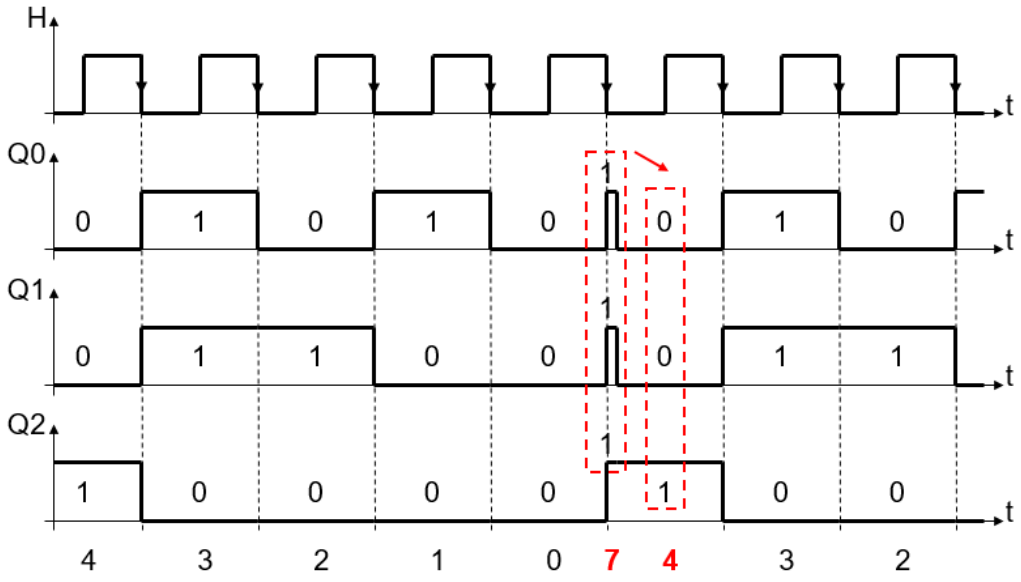
Table de vérité :

H	q_2	q_1	q_0	Q_2	Q_1	Q_0
↓	1	0	0	0	1	1
↓	0	1	1	0	1	0
↓	0	1	0	0	0	1
↓	0	0	1	0	0	0
↓	0	0	0	1	1	1
	1	1	1	1	0	0
x	1	1	0	x	x	X
x	1	0	1	x	x	X

Logigramme :



Chronogramme :



4.2.5 Exercice 1 :

Proposer un logigramme qui synthétise en même temps un compteur et un décompteur asynchrone modulo 4 active sur front descendant en utilisant des bascules JK.

Principe :

On doit prévoir une variable de choix Y qui permettra de trancher le sens du dénombrement (s'il s'agit d'un comptage ou un décomptage). Pour un compteur/décompteur active sur front descendant de l'horloge il est nécessaire de :

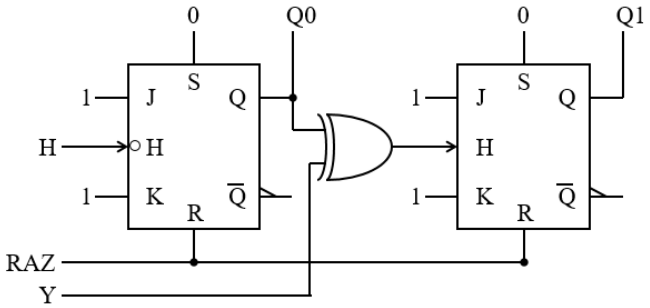
- conduire la sortie Q_n vers l'horloge H_{n+1}, si $Y = 0 \rightarrow$ comptage ;
- conduire la sortie Q_n complémentée vers l'horloge H_{n+1}, si $Y = 1 \rightarrow$ décomptage.

Pour se faire on utilise une porte ou exclusive :

Y	Q_n	H_{n+1}
0	0	0
0	1	1
1	0	1
1	1	0

$$H_{n+1} = \bar{Y}Q_n + Y\overline{Q_n} = Y \oplus Q_n$$

Logigramme :



4.2.6 Les avantages et limites des compteurs/décompteurs asynchrones :

Les compteurs/décompteurs asynchrones sont réputés par leur simplicité d'implémentation et sont généralement recommandés pour des configurations faisant intervenir un nombre limité (faible) de bascules. Au-delà d'un certain nombre de bascules les compteurs/décompteurs asynchrones engendrent des états transitoires non souhaités sur les sorties après chaque front de l'horloge ; et occasionnent une vitesse de fonctionnement très handicapante (limitée) (Voir figure 3). D'un autre côté, il arrive que l'on souhaite énumérer un cycle dont l'ordonnancement est aléatoire (différent du code binaire naturel). Dans ces cas de figures la solution réalisable est : "les compteurs/décompteurs synchrones".

Dans une configuration synchrone, le signal d'horloge est commun à toutes les bascules, en effet, les sorties des bascules changent d'état en même moment (après le front actif de l'horloge).

4.3 Synthèse d'un compteur/décompteur synchrone :

L'un des points fort d'une configuration synchrone est la possibilité de reproduire n'importe quel cycle de comptage, cela signifie qu'il est nécessaire d'effectuer une synthèse complète pour chaque cycle retenu.

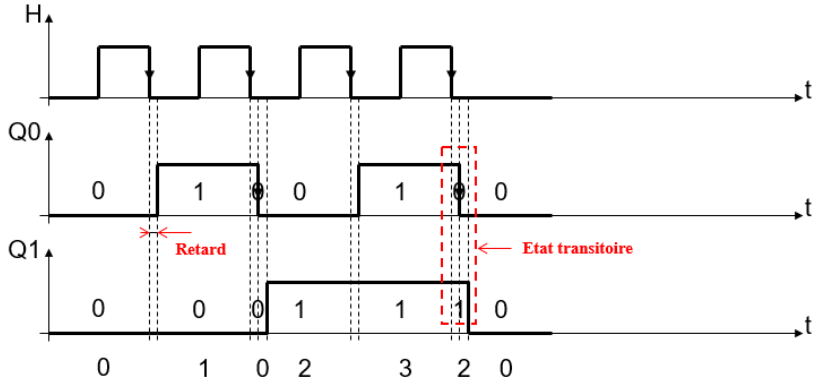


Fig. 2 : *états transitoires et retards engendrés par un compteur asynchrone.*

4.3.1 Table d'excitation d'une bascule :

Sert à spécifier le choix des valeurs à présenter aux entrées d'une bascule afin de permuter sa sortie d'un état à l'autre. Cette section présentera les tables d'excitation de quatre types de bascule.

A) Bascule JK :

Table de vérité			
J	**K**	**Q_-**	**Q_+**
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Table d'excitation			
Q_-	**Q_+**	**J**	**K**
0 →	0	0	x
0 →	1	1	x
1 →	0	x	1
1 →	1	x	0

B) Bascule RS :

Table de vérité			
S	**R**	**Q_-**	**Q_+**
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	x
1	1	1	x

Table d'excitation			
Q_-	**Q_+**	**S**	**R**
0 →	0	0	x
0 →	1	1	0
1 →	0	0	1
1 →	1	x	0

C) Bascule D :

Table de vérité		
D	**Q_-**	**Q_+**
0	0	0
0	1	0
1	0	1
1	1	1

Table d'excitation		
Q_-	**Q_+**	**D**
0 →	0	0
0 →	1	1
1 →	0	0
1 →	1	1

D) Bascule T :

Table de vérité		
T	**Q_-**	**Q_+**
0	0	0
0	1	1
1	0	1
1	1	0

Table d'excitation		
Q_-	**Q_+**	**T**
0 →	0	0
0 →	1	1
1 →	0	1
1 →	1	0

Exemple 5 : synthétiser un compteur synchrone modulo 8 active sur front descendant à base de bascules **JK.**

Table de transition à partir de la table d'excitation :

Q_2	Q_1	Q_0	J_2	K_2	J_1	K_1	J_0	K_0
0↓	0↓	0↓	0	x	0	x	1	x
0↓	0↓	1↓	0	x	1	x	x	1
0↓	1↓	0↓	0	x	x	0	1	x
0↓	1↓	1↓	1	x	x	1	x	1
1↓	0↓	0↓	x	0	0	x	1	x
1↓	0↓	1↓	x	0	1	x	x	1
1↓	1↓	0↓	x	0	x	0	1	x
1	1	1	x	1	x	1	x	1

Table d'excitation			
Q_-	**Q_+**	**J**	**K**
0 →	0	0	x
0 →	1	1	x
1 →	0	x	1
1 →	1	x	0

Tableaux de Karnaugh et équations logiques :

Q_2Q_1 \ Q_0	0	1
00	0	0
01	0	1
11	x	x
10	x	x

Q_2Q_1 \ Q_0	0	1
00	x	x
01	x	x
11	0	1
10	0	0

$J_2 = K_2 = Q_1Q_0.$

Q_2Q_1 \ Q_0	0	1
00	0	1
01	x	x
11	x	x
10	0	1

Q_2Q_1 \ Q_0	0	1
00	x	x
01	0	1
11	0	1
10	x	x

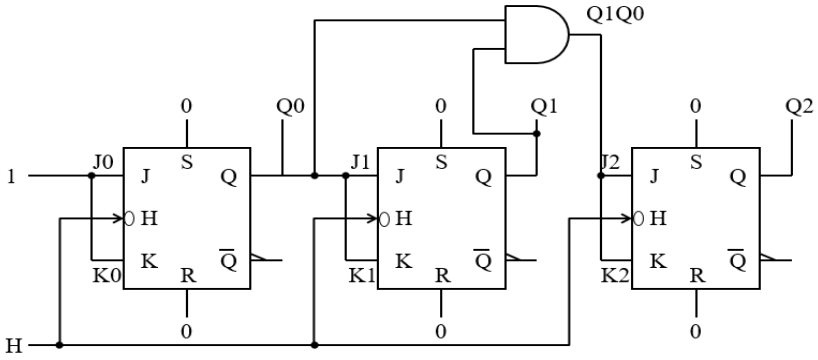
$J_1 = K_1 = Q_0.$

Q_2Q_1 \ Q_0	0	1
00	1	x
01	1	x
11	1	x
10	1	x

Q_2Q_1 \ Q_0	0	1
00	x	1
01	x	1
11	x	1
10	x	1

$J_0 = K_0 = 1.$

Logigramme :



Exemple 6 : synthétiser un décompteur synchrone modulo 8 active sur front montant a base des bascules **RS.**

Table de transition à partir de la table d'excitation :

Q_2	Q_1	Q_0	S_2	R_2	S_1	R_1	S_0	R_0
1↓	1↓	1↓	x	0	x	0	0	1
1↓	1↓	0↓	x	0	0	1	1	0
1↓	0↓	1↓	x	0	0	x	0	1
1↓	0↓	0↓	0	1	1	0	1	0
0↓	1↓	1↓	0	x	x	0	0	1
0↓	1↓	0↓	0	x	0	1	1	0
0↓	0↓	1↓	0	x	0	x	0	1
0	0	0	1	0	1	0	1	0

Table d'excitation			
Q_-	**Q_+**	**S**	**R**
0 →	0	0	x
0 →	1	1	0
1 →	0	0	1
1 →	1	x	0

Tableaux de Karnaugh et équations logiques :

Q_2Q_1 \ Q_0	0	1
00	1	0
01	0	0
11	x	x
10	0	x

Q_2Q_1 \ Q_0	0	1
00	0	x
01	x	x
11	0	0
10	1	0

$S_2 = \overline{Q_2}\,\overline{Q_1}\,\overline{Q_0}$ *et* $R_2 = Q_2\,\overline{Q_1}\,\overline{Q_0}$.

Q_2Q_1 \ Q_0	0	1
00	1	0
01	0	x
11	0	x
10	1	0

Q_2Q_1 \ Q_0	0	1
00	0	x
01	1	0
11	1	0
10	0	x

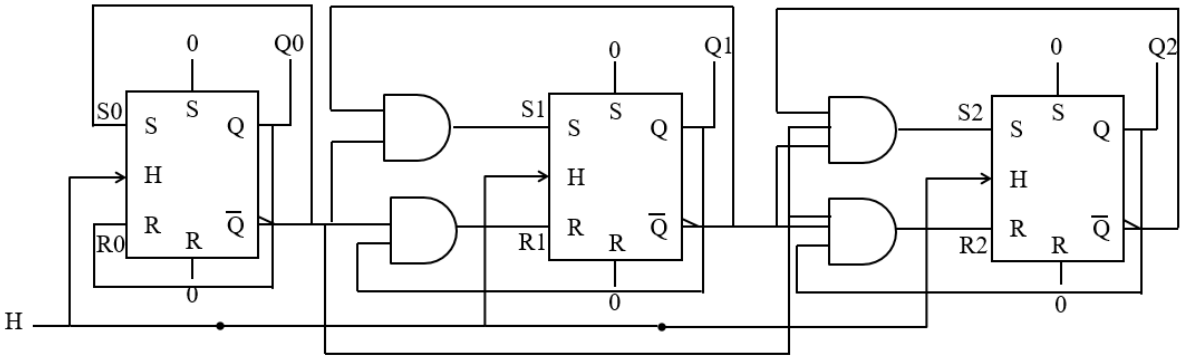
$S_1 = \overline{Q_1}\,\overline{Q_0}$ *et* $R_1 = Q_1\,\overline{Q_0}$.

Q_2Q_1 \ Q_0	0	1
00	1	0
01	1	0
11	1	0
10	1	0

Q_2Q_1 \ Q_0	0	1
00	0	1
01	0	1
11	0	1
10	0	1

$S_0 = \overline{Q_0}$ *et* $R_0 = Q_0$.

Logigramme :



Exercice 2 : synthétiser un compteur synchrone active sur front descendant à base de bascules D, qui nous permet de réaliser la séquence suivante : {2, 5, 1, 4, 8, 2…} :

- Dresser la table de transition de la bascule D à partir de la table d'excitation
- Trouver les équations des entrées D.
- Donner le logigramme.
- Donner le chronogramme.

Table de transition de la bascule D à partir de la table d'excitation :

Q_3	Q_2	Q_1	Q_0	D_3	D_2	D_1	D_0
0↓	0↓	1↓	0↓	0	1	0	1
0↓	1↓	0↓	1↓	0	0	0	1
0↓	0↓	0↓	1↓	0	1	0	0
0↓	1↓	0↓	0↓	1	0	0	0
1↓	0↓	0↓	0↓	0	0	1	0

Table d'excitation		
Q_-	**Q_+**	**D**
0 →	0	0
0 →	1	1
1 →	0	0
1 →	1	1

Equations des entrées D :

Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	x	0	x	0
01	1	0	x	x
11	x	X	x	x
10	0	X	x	x

$$D_3 = Q_2\overline{Q_0}$$

Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	x	1	x	1
01	0	0	x	x
11	x	x	x	x
10	0	x	x	x

$$D_2 = \overline{Q_3}\,\overline{Q_2}$$

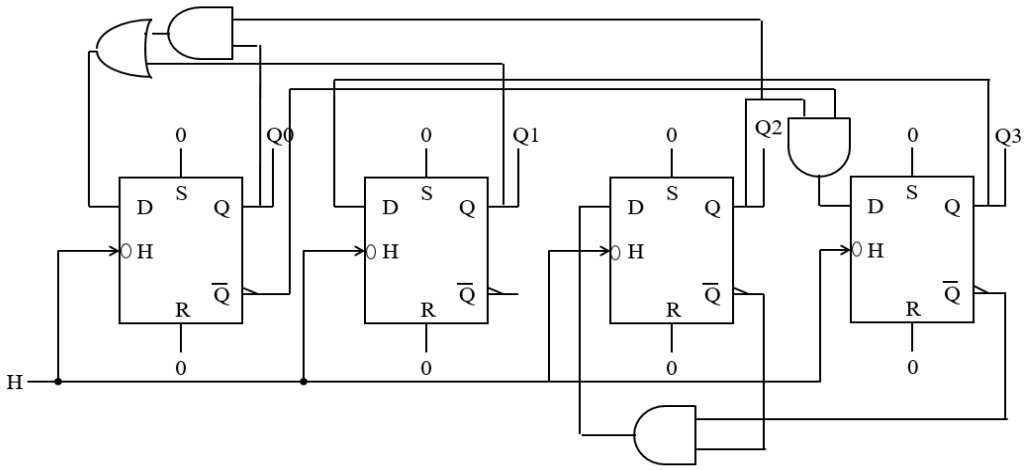
Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	x	0	x	0
01	0	0	x	x
11	x	x	x	x
10	1	x	x	x

$$D_1 = Q_3$$

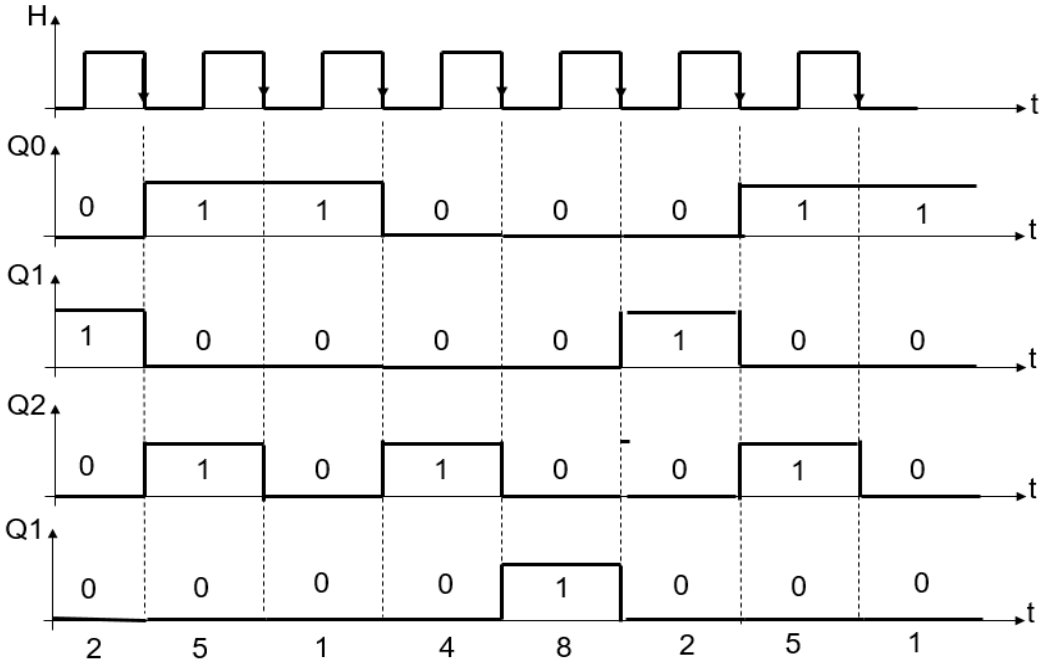
Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	x	0	x	0
01	0	0	x	x
11	x	x	x	x
10	1	x	x	x

$$D_0 = Q_1 + Q_2Q_0$$

Logigramme :



Chronogramme :



N.B : pour un compteur à cycle quelconque il est nécessaire :

- D'initialiser le compteur à une valeur incluse dans le cycle à modéliser. Dans l'exemple précèdent le compteur a été initialisé à la valeur 2 qui est incluse dans le cycle, on peut bien évidement choisir n'importe quelle valeur du cycle à condition de respecter l'ordonnancement.
- D'avoir un nombre suffisant de bascules pour pouvoir représenter la valeur la plus élevée du cycle. Pour notre exemple nous avons utilisé 4 bascules car la plus grande valeur du cycle est 8 et dont le codage en binaire nécessite 4 bits (1000).
- D'affecter aux états non inclus dans le cycle une valeur non déterminée (x/ Φ) sachant que ces états sont censés non appartenir au cycle. En remplissant dans notre exemple la table de Karnaugh nous avons attribué des valeurs non déterminées, à chaque combinaison non incluse dans l'ensemble {2, 5, 1, 4, 8}.

Biobibliographie :

[1] "Digital Logic and Computer Design" par M. Morris Mano.
[2] "Conception des circuits logiques numériques" par M. Morris Mano.
[3] "Circuit numériques : principes et pratique" par John M. Yarbrough.

Fiche TD 2 : Les compteurs/Décompteurs synchrones/Asynchrone.

Exercice 1 :

- Rappeler le principe de fonctionnement de la bascule T, Puis synthétiser un compteur asynchrone modulo $2^3 = 8$ active sur front descendant (réalisé avec des bascules T).
- Rappeler le principe de fonctionnement de la bascule D, puis synthétiser un compteur asynchrone modulo 6 active sur front descendant (réalisé avec des bascules D).

Exercice 2 :

- Synthétiser un décompteur asynchrone modulo $2^2 = 4$ active sur front descendant et réalisé avec des bascules JK.
- Synthétiser un décompteur asynchrone modulo 6 active sur front descendant et réalisé avec des bascules JK.

Exercice 3 :

Synthétiser un compteur synchrone active sur front montant à base de bascules T, qui nous permet de réaliser la séquence suivante : {0, 2, 3, 5, 6, 0…} :

- Dresser la table de transition de la bascule T à partir de la table d'excitation
- Trouver les équations des entrées T.
- Donner le logigramme.
- Donner le chronogramme.

Solution exercice 1 :

Rappeler le principe de fonctionnement de la bascule T (table de vérité), Puis synthétiser un compteur asynchrone modulo $2^3 = 8$ active sur front descendant (réalisé avec des bascules T).

Table de vérité de la bascule T :

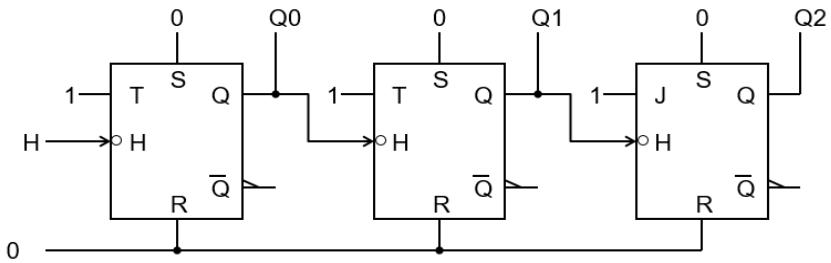
Table de vérité		
T	**Q_-**	**Q_+**
0	0	0
0	1	1
1	0	1
1	1	0

Synthèse d' un compteur asynchrone modulo $2^3 = 8$ active sur front descendant (réalisé avec des bascules T).

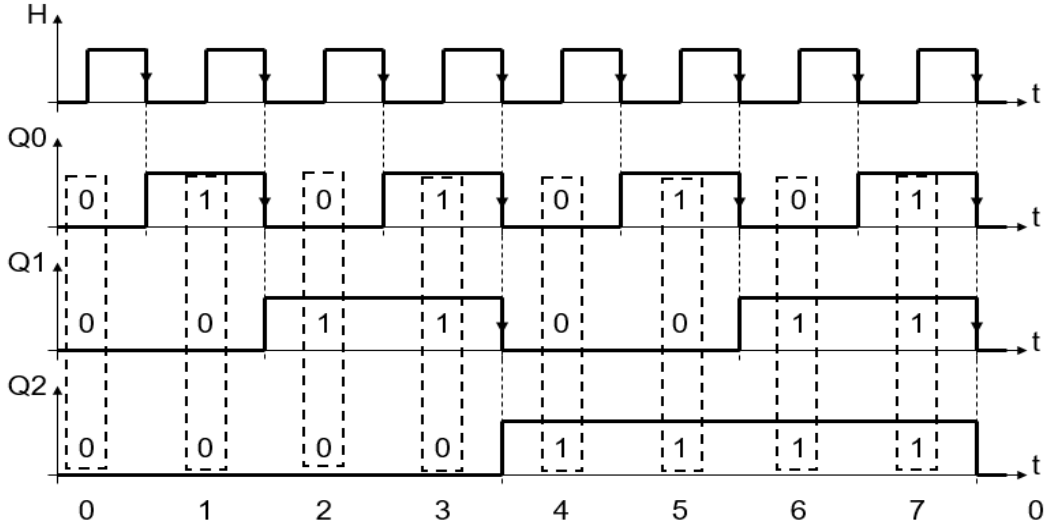
Table de vérité :

R	H	q_0	q_1	q_2	Q_0	Q_1	Q_2
1	x	x	x	X	0	0	0
0	↓	0	0	1	0	0	1
0	↓	0	1	0	0	1	0
0	↓	0	1		0	1	1
0	↓	1	0	0	1	0	0
0	↓	1	0	1	1	0	1
0	↓	1	1	0	1	1	0
0	↓	0	0	1	1	1	1

Logigramme :



Chronogramme :



Principe de fonctionnement de la bascule D :

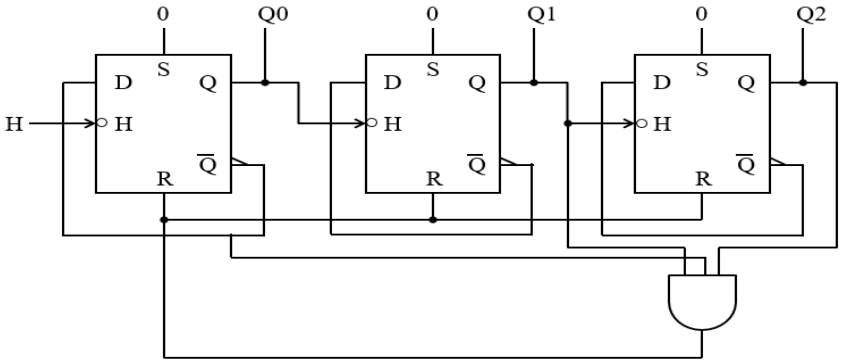
Table de vérité		
D	**Q_-**	**Q_+**
0	0	0
0	1	0
1	0	1
1	1	1

Synthétise d'un compteur asynchrone modulo 6 active sur front descendant (réalisé avec des bascules D).

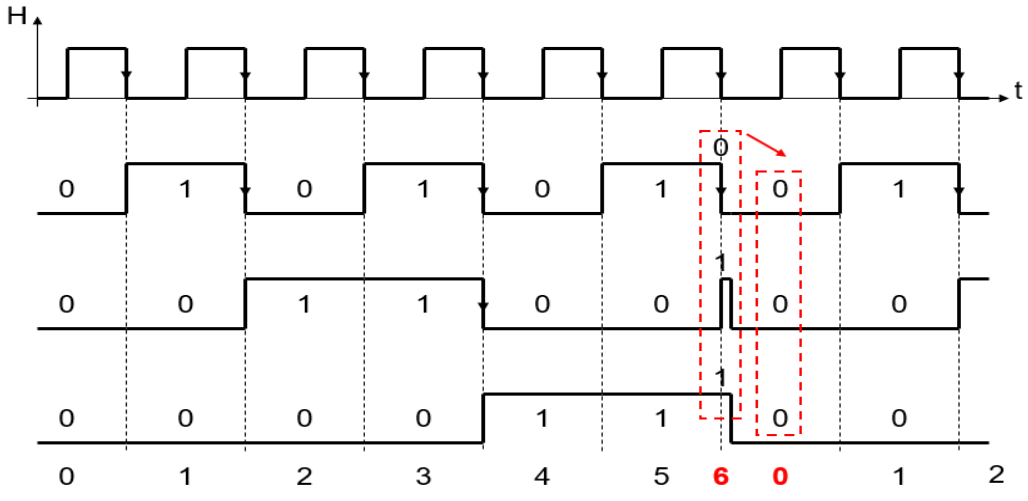
Table de vérité :

R	H	q_2	q_1	q_0	Q_2	Q_1	Q_0
1	x	x	x	x	0	0	0
0	↓	0	0	0	0	0	1
0	↓	0	0	1	0	1	0
0	↓	0	1	0	0	1	1
0	↓	0	1	1	1	0	0
0	↓	1	0	0	1	0	1
1	↓	1	0	1	**0**	**0**	**0**
					Forcer une mise à 0		

Logigramme :



Chronogramme :



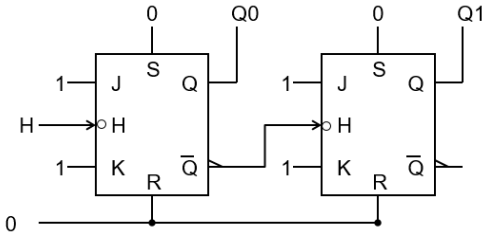
Solution exercice 2 :

Synthétiser un décompteur asynchrone modulo $2^2 = 4$ active sur front descendant et réalisé avec des bascules JK.

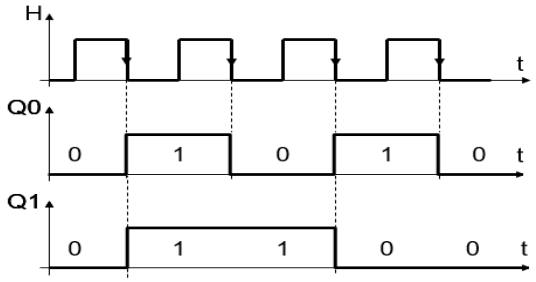
Table de vérité :

R	H	q_0	q_1	Q_0	Q_1
1	x	x	x	0	0
0	↓	0	0	1	1
0	↓	1	1	1	0
0	↓	1	0	0	1
0	↓	0	1	0	0

Logigramme :



Chronogramme :

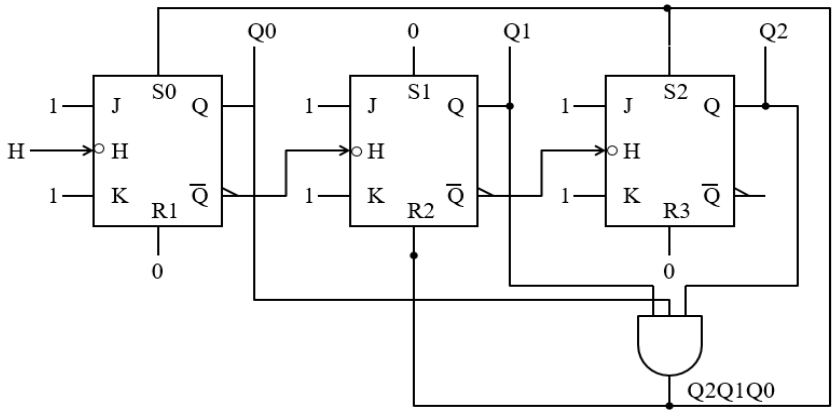


Synthétiser un décompteur asynchrone modulo 6 active sur front descendant et réalisé avec des bascules JK.

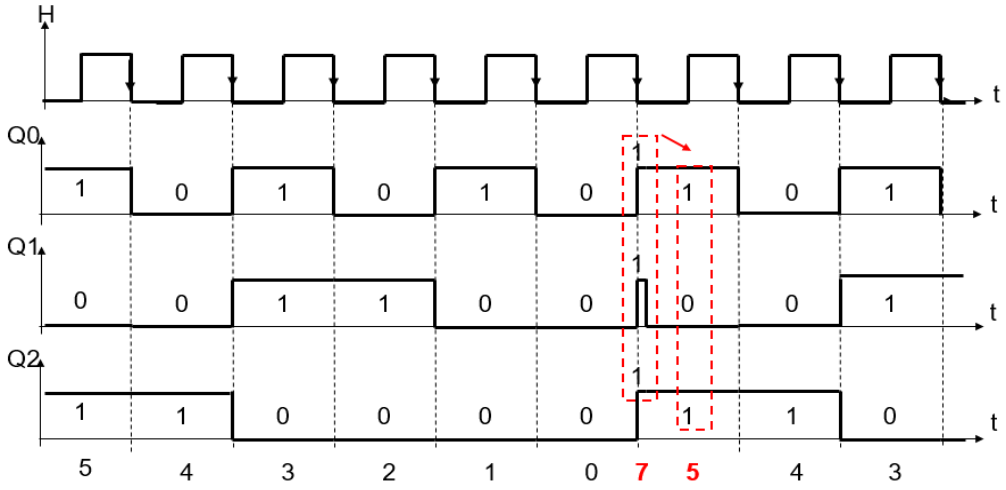
Table de vérité :

H	q_2	q_1	q_0	Q_2	Q_1	Q_0
↓	1	0	1	1	0	0
↓	1	0	0	0	1	1
↓	0	1	1	0	1	0
↓	0	1	0	0	0	1
↓	0	0	1	0	0	0
↓	0	0	0	1	1	1
	1	1	1	1	0	1
x	1	1	0	x	x	x

Logigramme :



Chronogramme :



Solution exercice 3 :

Table de transition de la bascule T à partir de la table d'excitation :

Q_3	Q_2	Q_1	T_2	T_1	T_0
0↓	0↓	0↓	0	1	0
0↓	1↓	0↓	0	0	1
0↓	1↓	1↓	1	1	0
1↓	0↓	1↓	0	1	1
1↓	1↓	0↓	1	1	0

Table d'excitation		
Q_-	**Q_+**	**T**
0 →	0	0
0 →	1	1
1 →	0	1
1 →	1	0

Equations des entrées T :

Q_2Q_1 \ Q_0	0	1
00	0	x
01	0	1
11	1	x
10	x	0

$T_2 = Q_2\overline{Q_0} + Q_1Q_0$.

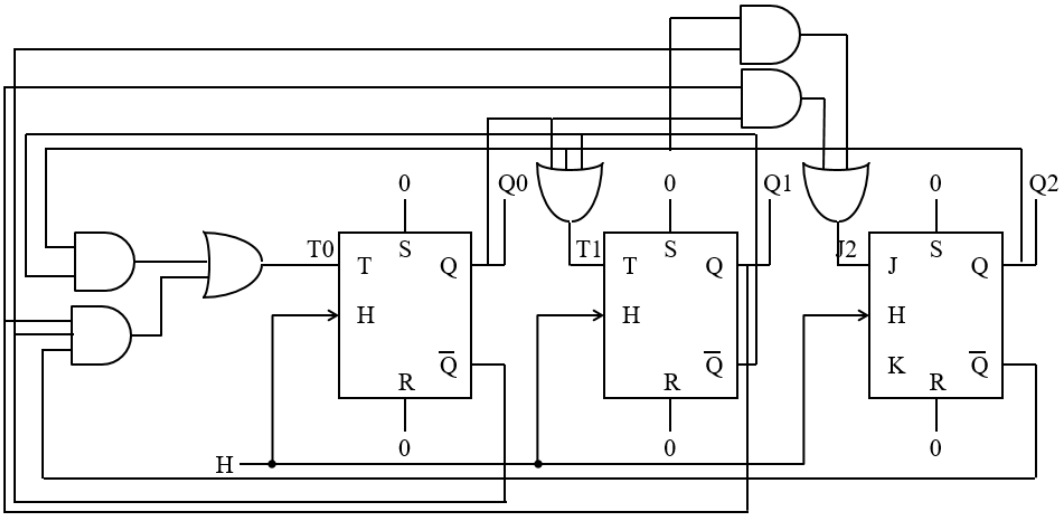
Q_2Q_1 \ Q_0	0	1
00	1	x
01	0	1
11	1	x
10	x	1

$$T_1 = Q_0 + Q_2 + \bar{Q}_1.$$

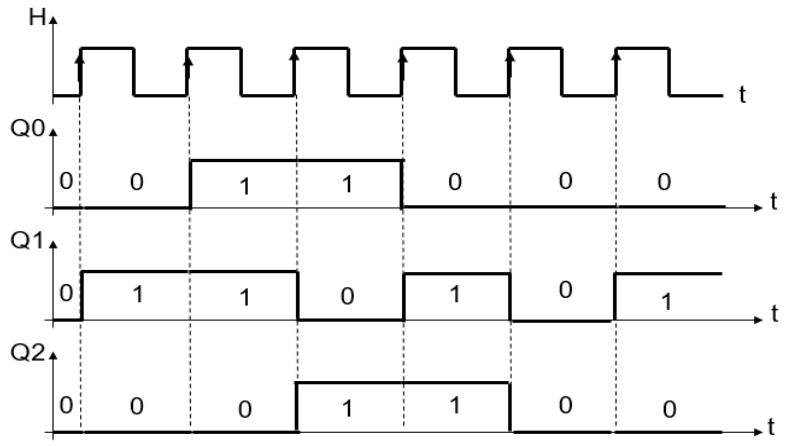
Q_2Q_1 \ Q_0	0	1
00	0	x
01	1	0
11	0	x
10	x	1

$$T_0 = Q_2\bar{Q}_1 + \bar{Q}_2Q_1\bar{Q}_0$$

Logigramme :

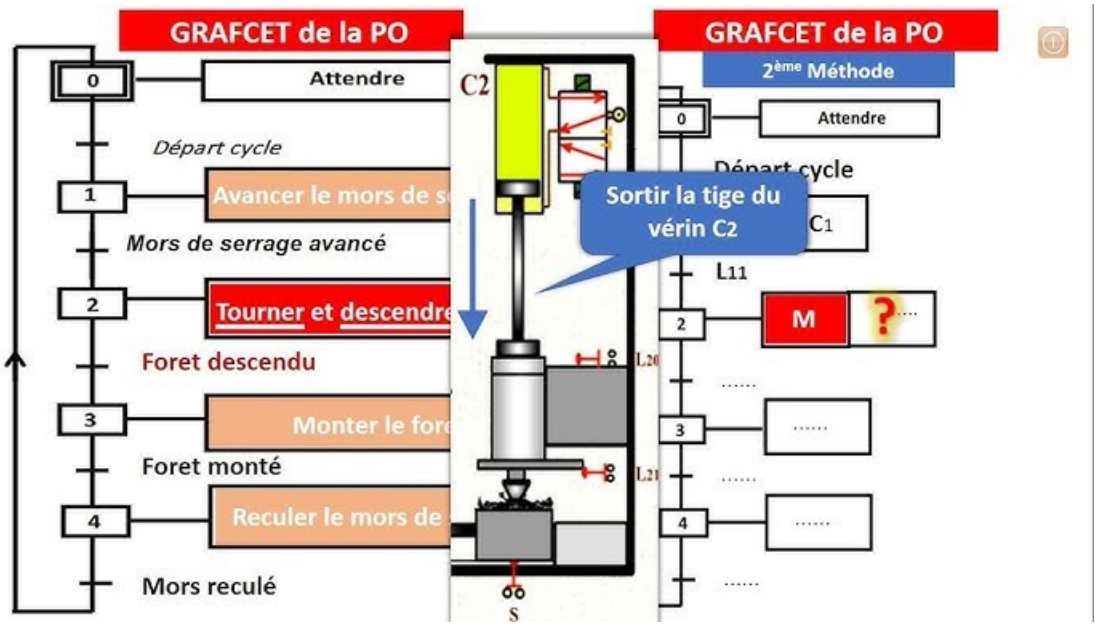


Chronogramme :



CHAPITRE 5: LE GRAFCET.

"L’apprentissage n’est pas un fardeau, mais un chemin vers l’accomplissement personnel et la découverte de notre potentiel."



CHAPITRE 5 : LE GRAFCET

5.1 C'est quoi le GRAFCET ?

Signification : GRAphe **F**onctionnel de **C**ommande **E**tape **T**ransition

Comment est-il né: en 1977 en France, du travail d'un groupe d'universitaires et d'industriels du l'AFCET : **A**ssociation **F**rançaise pour la **C**ybernétique **E**conomique et **T**echnique.

Rôle: Le Grafcet est un outil graphique permettant la représentation du comportement d'un automatisme à évolution séquentielle (c'est-à-dire décomposable en étapes). C'est un langage universel ; qui peut être implémenté sur automate ou sur ordinateur.

5.2 Les éléments de base du GRAFCET :

- Un Grafcet est une représentation graphique composé d'**étapes**, de **transitions** et de **liaisons.**
- Un Grafcet est une interprétation engendrée par les **actions** (associées aux étapes) et les **réceptivités** (associées aux transitions).

5.2.1 Etape :

Permet de définir une situation bien précise d'un automatisme, elle est symbolisée par un carré et distinguée des autres étapes par un numéro unique. A chaque étape i, on associe une variable logique X_i où X_i=1 (si l'étape i est active et repérée par un point noir placé sous le chiffre) et X_i=0 (si l'étape i est inactive sans point).

Remarque:

- L'étape initiale (étape initialement activée) est représentée par un carré double.

Etape X_1=0 inactive

Etape X_2=1 active

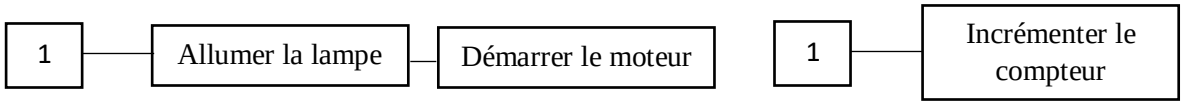
Etape Initiale

5.2.3 Action :

Une ou plusieurs actions peuvent être associées à une étape ; Ces actions expriment ce qui doit être réalisé à chaque fois que l'on active l'étape à laquelle elles sont associées. Les actions peuvent être externes (ordres émis vers la partie opérative ou vers les éléments externes, afin de commander l'automatisme) ou internes (fonctions spécifiques de l'automatisme temporisation, comptage, etc.). Les actions sont mentionnées dans un ou plusieurs rectangles reliés par un trait et positionné à droite de l'étape.

Remarque :

- Une étape peut ne pas avoir d'actions (attente de la fin d'une autre action par exemple), elle est nommée étape vide (étape d'attente).



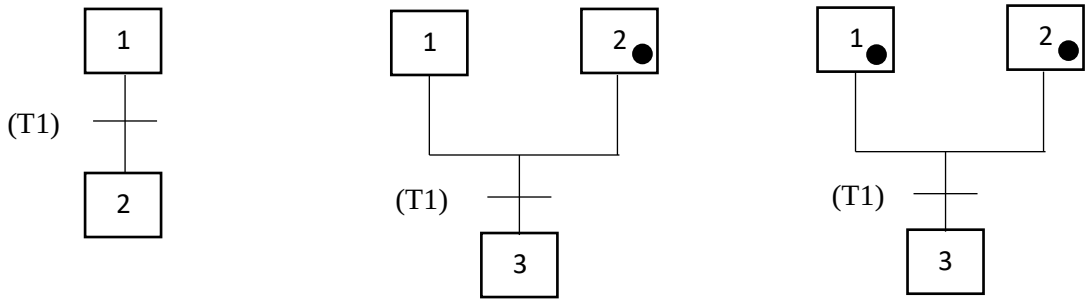
Action externe *Action interne*

5.2.4 Transitions :

Elles expriment les possibilités d'évolution d'une étape à une ou plusieurs étapes suivantes. Une transition est représentée par un segment orthogonal (perpendiculaire) aux liaisons orientées (liaisons joignant 2 étapes, voir section 2.5)

Remarque :

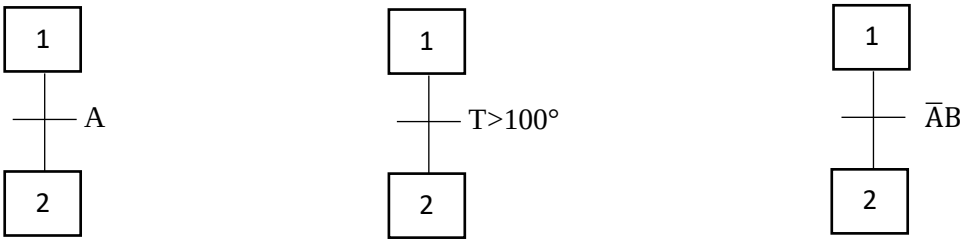
- Une transition est dite validée quand toutes les étapes précédentes immédiatement reliées à cette transition sont achevées (actives), et dite non validée, dans le cas contraire.
- Il existe une et une seule transition entre deux étapes.
- Afin de faciliter la lecture du GRAFCET, on peut attribuer à chaque transition un identifiant mis entre parenthèses et repérer à gauche de la transition.



Transition T1 *Transition T1 non validée* *Transition T1 validée*

5.2.5 Réceptivités :

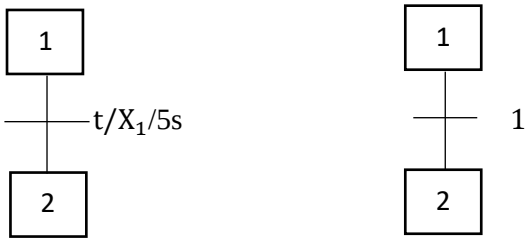
On associe à chaque transition une proposition logique ou une fonction combinatoire appelée réceptivité qui peut prendre soit la valeur = 1 (vraie), soit la valeur = 0 (fausse). La réceptivité est fonction d'informations externes (provenant des capteurs internes ou externes), ou fonction d'informations internes (compteurs, temporisations, étapes actives ou inactives). Les réceptivités s'écrivent à droite des transitions.



Remarque :

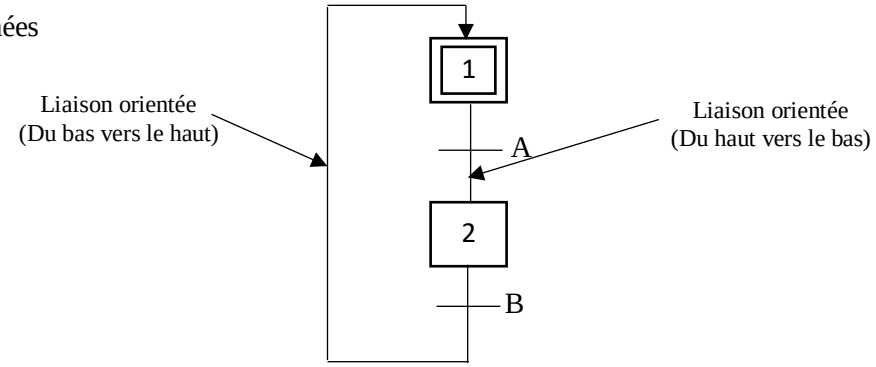
- Pour faire intervenir la temporisation dans une réceptivité on applique l'expression suivante: $\delta = t/X_i/q$ où X_i est la variable logique associée à l'étape i et q la durée de la temporisation.

- On appelle "réceptivité toujours vrais" une transition dont la réceptivité =1.



5.2.6 Liaisons orientées :

Les liaisons orientées permettent de relier les étapes aux transitions et les transitions aux étapes. Les liaisons orientées de haut en bas ne sont pas fléchées, contrairement aux liaisons orientées de bas en haut qui sont fléchées



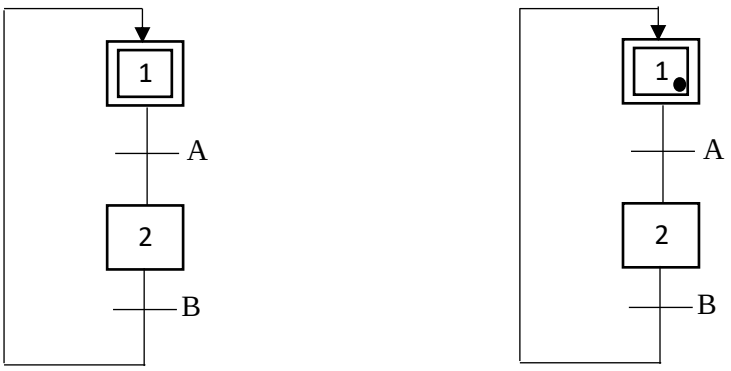
Remarque :

- Une liaison orientée ne peut jamais relier deux étapes ou deux transitions.
- Une liaison relie toujours une étape à une transition, et la même transition à une autre étape.

5.3 Les règles d'évolutions du GRAFCET :

A l'ensemble des règles de syntaxe vue en section 2, s'ajoutent cinq règles d'évolution qui précisent les conditions d'activation des étapes et le franchissement des transitions.

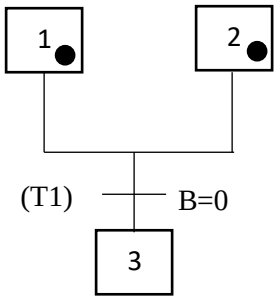
- **Règle 1** (Initialisation)**:** La situation initiale caractérise en général une situation de repos, elle est activée dès que l'automatisme (partie commande) est opérationnel.



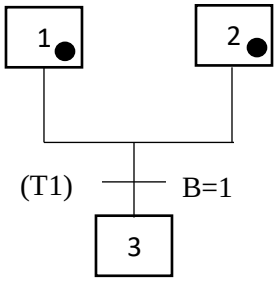
Automatisme à l'arrêt

Automatisme opérationnel au repos

- **Règle 2** (Franchissement d'une transition)**:** Lorsqu'une transition est **validée** (voir section 2.3) et que la **réceptivité** qui lui est associée est **vraie**, la transition devient franchissable et est obligatoirement franchie.

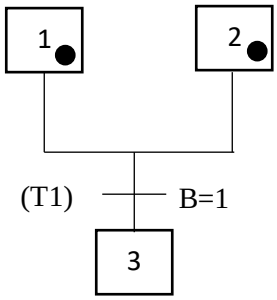


Transition T1 validée
La réceptivité de T1 est fausse
T1 Non franchissable

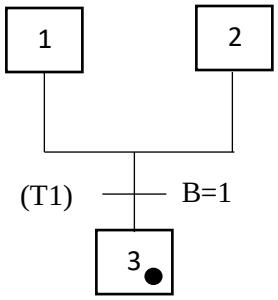


Transition T1 validée
La réceptivité de T1 est vraie
T1 franchissable

- **Règle 3** (Evaluation des étapes actives) **:** Le franchissement d'une transition engendre au même moment la désactivation de toutes les étapes précédant immédiatement la transition et l'activation de toutes les étapes suivant immédiatement la transition.



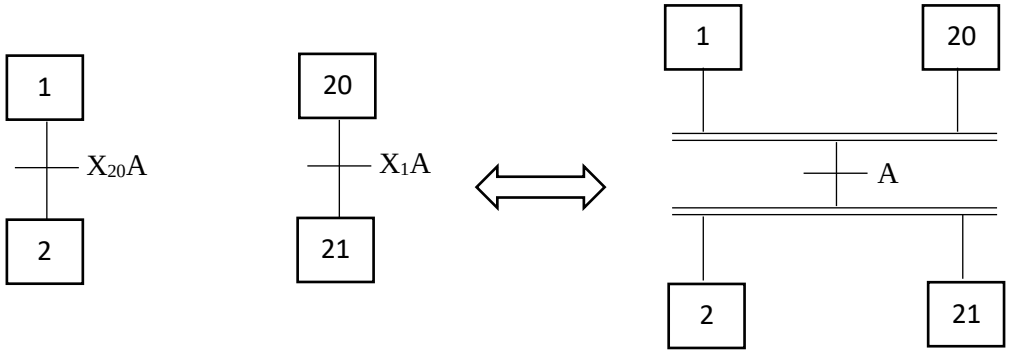
Transition T1 validée
La réceptivité de T1 est vraie
T1 franchissable



Etape X_1 et Etape X_2 désactivées
Etape X_3 activée
T1 franchie

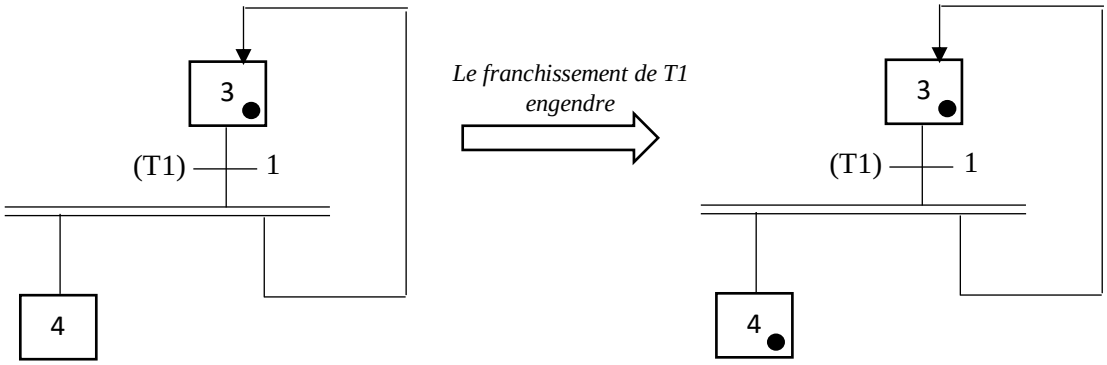
- **Règle 4** (Evolutions simultanées)**:** Plusieurs transitions simultanément franchissables sont simultanément franchies.

La règle 4 nous donne la possibilité de subdiviser un grafcet en un ensemble de diagrammes indépendants.



Evolutions simultanées

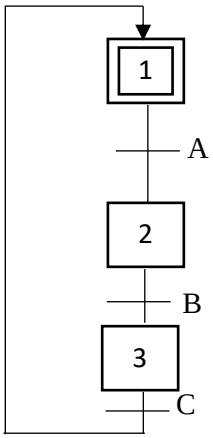
- **Règle 5** (Activation et désactivation simultanées)**:** Si une même étape doit être à la fois activée et désactivée, elle reste toujours activée.



5.4 Les séquences de base du GRAFCET :

5.4.1 Grafcet à séquence unique :

Un GRAFCET à séquence unique (appelé aussi GRAFCET linéaire), est un GRFCET, dont chaque étape est reliée à une seule transition et chaque transition est reliée à une seule étape. Autrement dit, le GRAFCET passe d'une étape à l'autre en franchissant une seule transition. Dans ce type de séquences, une et une seule étape peut être active à la fois.



GRAFCET a séquence unique

Remarque :

- La séquence est dite active si au moins une des étapes est active, elle est dite inactive dans le cas contraire.

Exemple 1 :

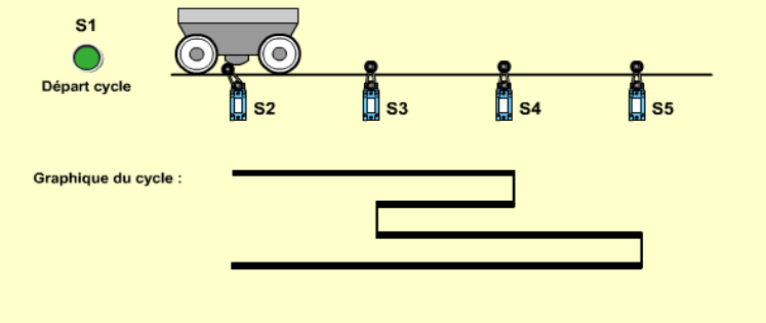


Figure 1: LE WAGONNET.

Fonctionnement:

Un chariot se déplace sur un rail et permet de chercher des produits à trois endroits repéré pas trois capteurs S3, S4, S5 puis revenir à son point de départ repéré par le capteur S2.

Quand on appuie sur le bouton de départ du cycle (S1), le chariot va se déplacer jusqu'à (S4), puis revenir en arrière jusqu'à (S3), ensuite il va avancer jusqu'à (S5) où il va marquer un temps d'arrêt de 30s, en fin il revient à son point de départ (S2).

a) Définir les capteurs et les actions.

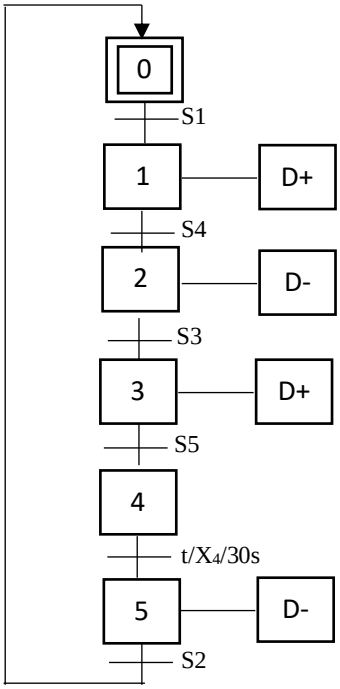
b) Définir le GRAFCET.

Solution :

a) Définir les capteurs et les actions:

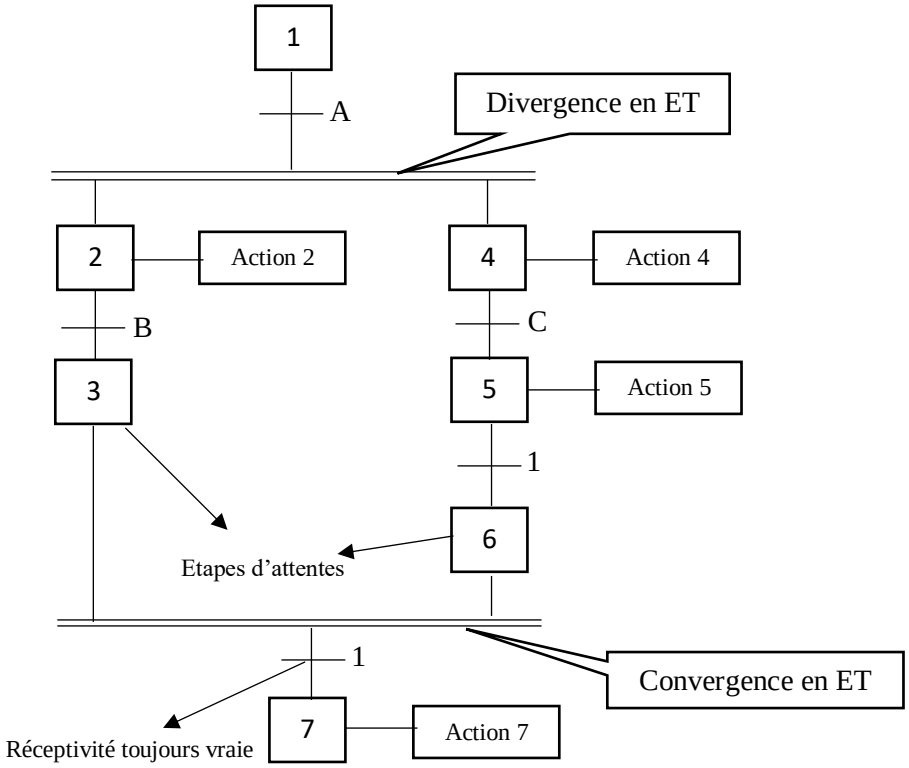
Capteurs	Actions
S1	D+ : Déplacer à droite
S2	D- : Déplacer à gauche
S3	
S4	
S5	

b) Définir le GRAFCET:



5.4.2 Grafcet à séquences simultanées (aiguillage en ET/ parallélisme structurel):

On parle "d'aiguillage en ET ", si le franchissement d'une transition conduit au déroulement parallèle de plusieurs séquences indépendantes.



Remarque:

- On prévoie généralement à la fin de chaque séquence indépendante une étape d'attente, cela va permettre la désactivation de ces séquences en même temps.
- Une divergence en ET se termine toujours par une convergence en ET.
- Une convergence en ET est généralement suivie par une "réceptivité toujours vraie".
- Le nombre de branches parallèles peut être supérieur à 2.

Exemple 2:

Quand on appui sur le bouton **dcy,** les chariots (CH1et CH2) effectuent un cycle aller-retour. Le prochain cycle ne peut être relancé que si les deux chariots sont en leurs positions de départ (complétement à gauche).

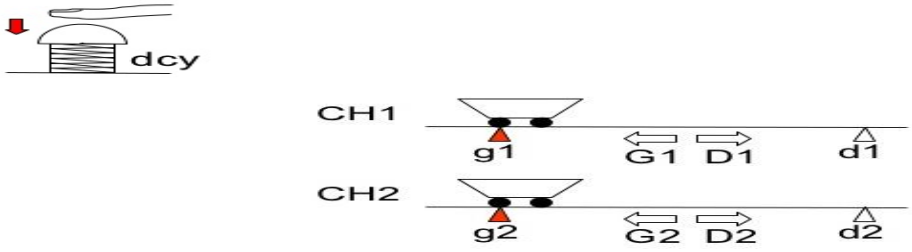
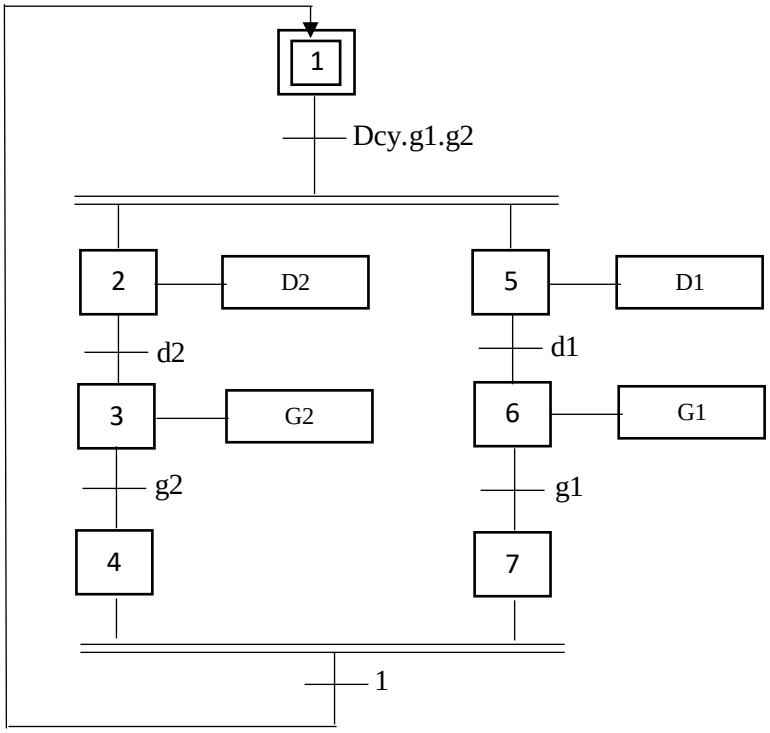


Figure 1: Exemple séquences simultanées (aiguillage en ET)

Capteurs	**Actions**
dcy	D1 : Déplacer à droite (CH1)
g1: capteur "position gauche"(CH1)	G1 : Déplacer à gauche(CH1)
d1: capteur "position droite"(CH1)	D2 : Déplacer à droite (CH2)
g2: capteur "position gauche"(CH2)	G2 : Déplacer à gauche(CH2)
d2: capteur "position droite"(CH2)	

a) Définir le GRAFCET.

Solution :

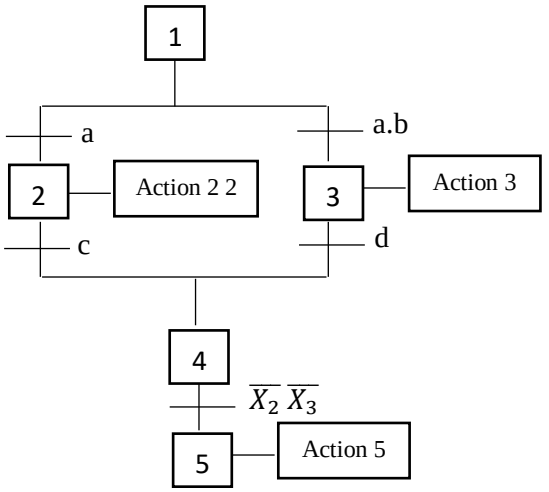


5.4.3 Grafcet avec choix de séquences (aiguillage en OU) :

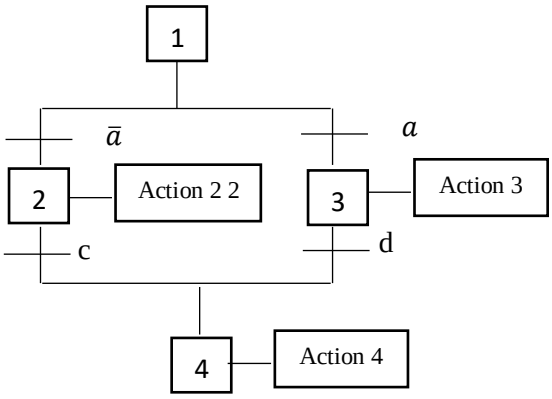
Il s'agit d'un aiguillage ou d'une sélection de séquence, établi en fonction des conditions données par les réceptivités associées aux transitions. On distingue deux cas :

a) Divergence en OU "non exclusive": Dans ce cas les conditions portées par les réceptivités permettent des évolutions simultanées de séquences.

b) Divergence en OU "exclusive": Dans ce cas les réceptivités sont exclusives: et des évolutions simultanées de séquences ne sont pas possibles.



Divergence en OU " non exclusive"



Divergence en OU "exclusive"

En divergence en ou non exclusive:

- Si a est vraie ET b est fausse alors 2 est active et 3 est inactive
- Si a est vraie ET b est vraie alors 2 et 3 sont actives
- Pour la synchronisation l'étape 4 **(étape** d'attente) est prévue avec la réceptivité: $\overline{X_2}\,\overline{X_3}$ (étape 2 et 3 désactivées)

En divergence en ou exclusive:

- Si a est vraie alors 3 est active et 2 est inactive.
- Si a est faux alors 2 est active et 3 est inactive.

Remarques:

- Après une divergence en OU, on trouve une convergence en OU (convergence vers une étape commune).
- Le nombre de branches peut être supérieur à 2.
- La convergence de toutes les branches ne se fait pas obligatoirement au même endroit.

Exemple 3:

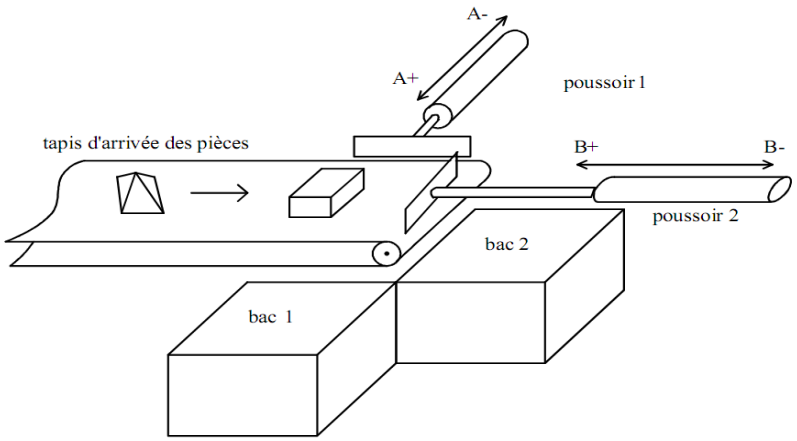


Figure 3: TRI DE PIÈCES

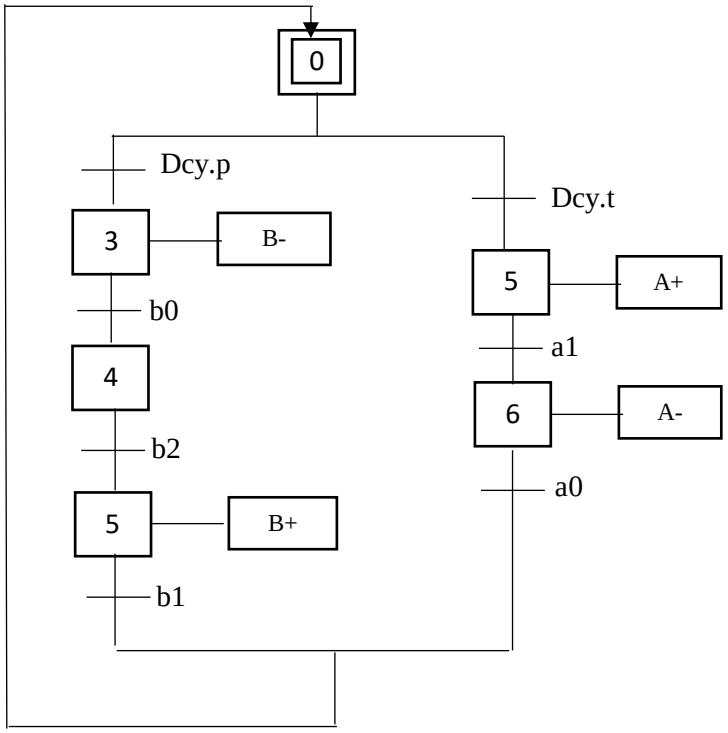
Cycle de fonctionnement:

- Apres la mise en marche du système (dcy=1) le tapis amène une pièce.
- Une fois la pièce contre le poussoir 2, on se retrouve avec 2 possibilités :
    - Possibilté1: Si la pièce est de type "pyramidale", le poussoir 1 avance pour la mettre dans le bac 1, puis le poussoir 1 revient à sa position de départ.
    - Possibilté2: Si la pièce est de type "prismatique", le poussoir 2 recule, puis le tapis avance afin de mettre la pièce dans le bac2, ensuite le poussoir 2 revient à sa position de départ.

Remarque: On ne prend pas en compte le fonctionnement du tapis pour les GRAFCET.

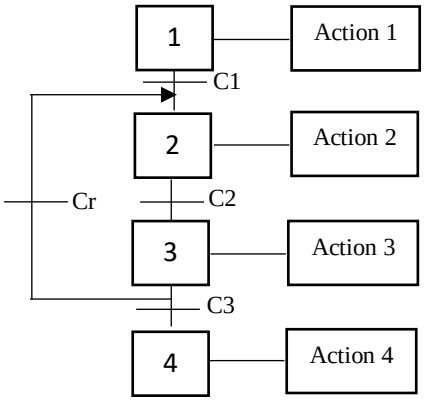
Capteurs	**Actions**
dcy	A+: Avancer le poussoir 1
a0 et b0: poussoir 1 et 2 rentrés.	A-: Reculer le poussoir 1
a1 et b1: poussoir 1 et 2 sortis.	B+: Avancer le poussoir 2
t: pièce de type "pyramidale" contre le poussoir 2.	B-: Reculer le poussoir 2
p: pièce de type " prismatique" contre le poussoir 2.	
b2: pièce tombée dans le bac 2.	

Solution:

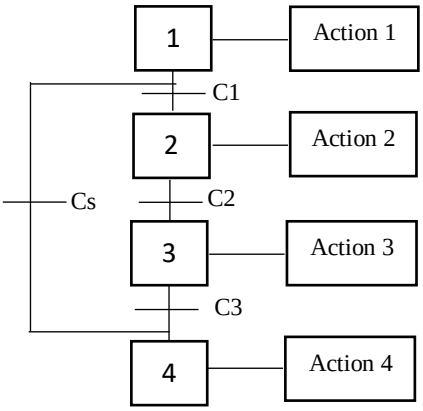


5.4.4 Grafcet à saut (et/ou) reprise de séquences :

Le saut d'étapes a pour fonction de sauter une ou plusieurs étapes lorsque les actions associées ne doivent pas être réalisées, La reprise de séquences (nommée aussi boucle) permet de reprendre la séquences N fois (N>=1), tant qu'une condition n'est pas validée.



Reprise d'étapes



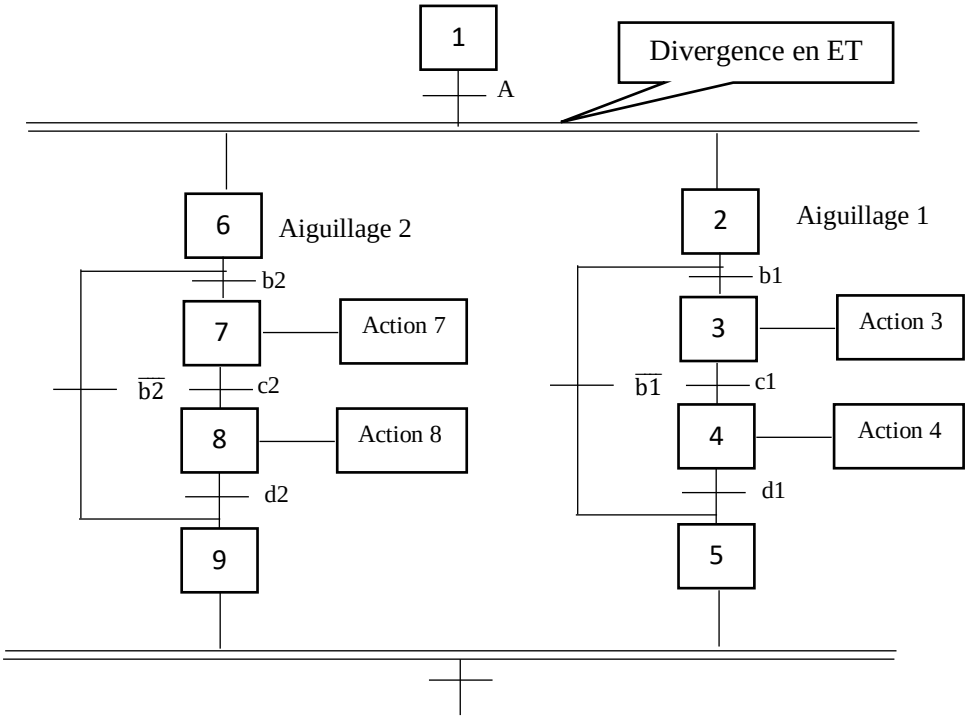
Saut d'étapes

5.4.5 Aiguillage après activation simultanée des séquences :

On trouve ce type de structure quand un ou plusieurs aiguillages doivent s'effectuer après les activations simultanées de ces séquences.

Remarque :

- Aucune action n'est associée aux étapes qui viennent immédiatement après la divergence en ET.



Exemple 4:

Le système présenté sur la figure 4 ci-dessous, illustre le fonctionnement relatif au remplissage et bouchage de bouteilles, le système est constitué:

- Tapi roulant, dont le fonctionnement est d'acheminer les bouteilles.
- Un sous-système de remplissage "P1" constitué d'une électrovanne "Ev".
- Un sous-système de bouchage "P2" constitué d'un vérin presseur "1D".

Cycle de fonctionnement:

- L'appui sur le bouton "dcy" permet la mise en fonction du système.
- Le moteur avance le tapi jusqu'à l'activation du capteur FCTP (ce dernier permet de fixer le pas d'avance du tapi). Ainsi, une bouteille est présentée à chacun des sous-systèmes P1 (bouteille détectée par pbv) et P2 (bouteille détectée par pbp).
- Les deux opérations de remplissage et de bouchage sont réalisées simultanément.
- Le remplissage se fera en deux étapes :
    - Etape 1: l'ouverture de l'électrovanne "Ev".
    - Etape 2: remplissage de la bouteille (niveau de remplissage contrôler par "br"), puis fermeture de l'électrovanne (fermeture contrôler par le capteur "Fr").
- Le bouchage des bouteilles se fera en deux étapes:
    - Etape 1: descente du vérin presseur "1D" (décente contrôlée par le capteur "vb")

- Etape 2: Remonté du vérin presseur "1D" (remonté contrôlée par le capteur "vh")

Remarque : Un nouveau cycle ne peut être relancé que si les deux opérations Remplissage et bouchage sont achevées.

a) Dans un tableau définissez les capteurs et les actions.

b) Réaliser le Grafcet associé au système présenté.

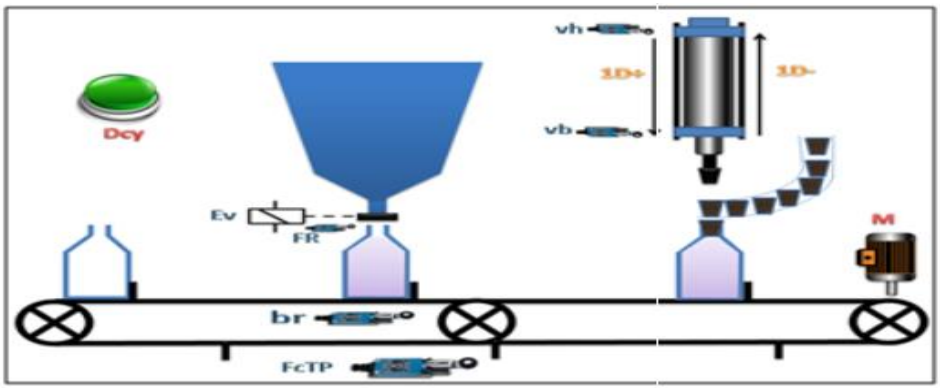


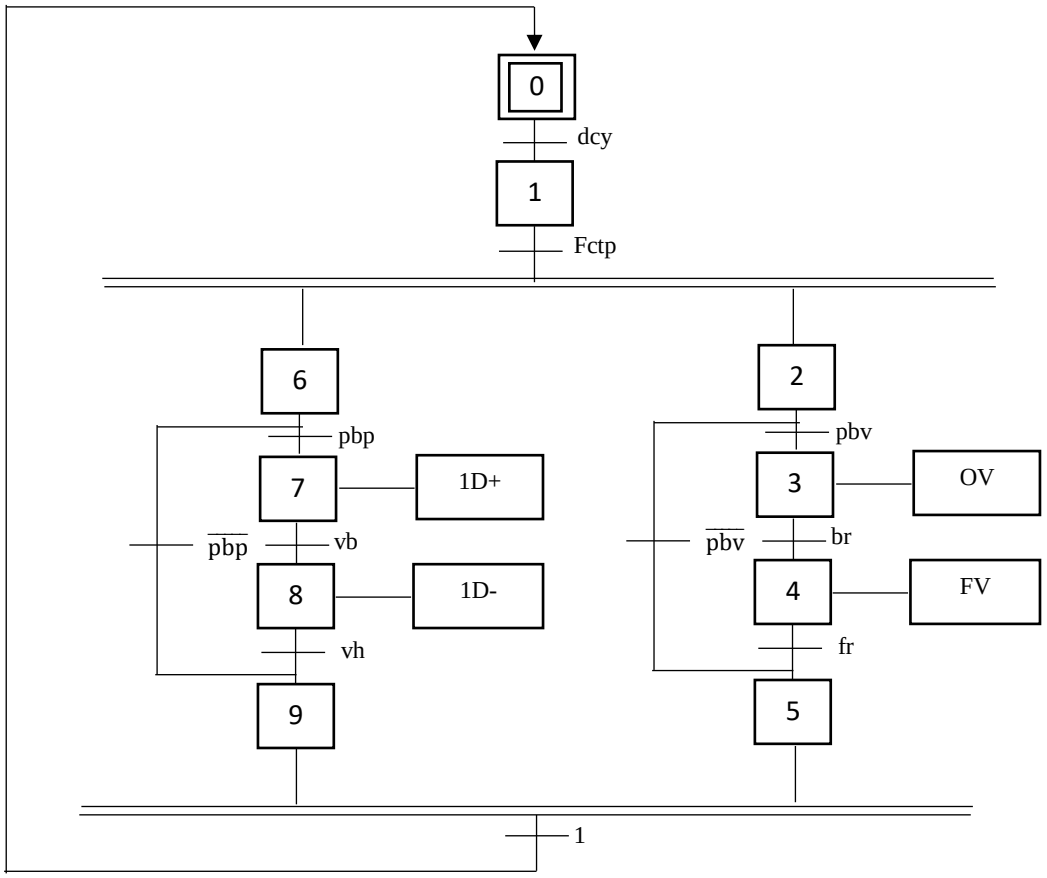
Figure 4: remplissage et bouchage automatique de bouteilles

Solution:

a. Dans un tableau définissez les capteurs et les actions.

Actions	Capteurs
OV: ouverture de l'électrovanne	dcy
FV: fermeture de l'électrovanne	Fctp
1D+ : descente du vérin presseur	pbv
1D- : Remonté du vérin presseur	pbp
	br
	fr
	vb
	vh

b. Réaliser le Grafcet associé au système présenté.

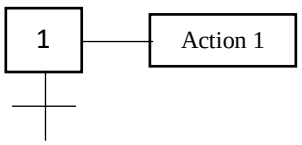


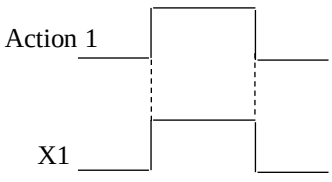
5.5 La classification des actions d'un GRAFCET :

5.5.1 Action continue inconditionnelle :

L'exécution de l'action 1 se poursuit tant que l'étape à laquelle elle est associé est active :

Action 1=X_1.

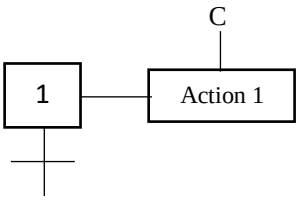


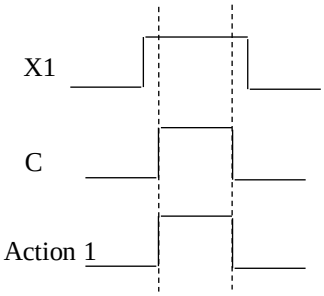


5.5.2 Action continue conditionnelle :

L'exécution de l'action 1 est soumise à une condition logique (l'action 1 n'est exécuté que si la condition associée est vraie et X_1 est active) :

Action 1=X_1.C

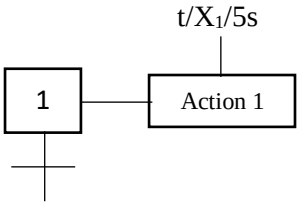


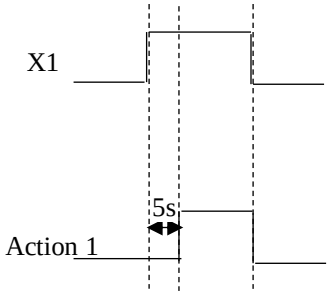


5.5.3 Action continue retardée :

La condition logique d'une action continue retardée est une temporisation, elle a pour objectif de retarder l'action par rapport à l'activité de l'étape qui lui est associée. On note, $t/X_i/q$ dans laquelle "X_i" indique l'étape prise comme origine du temps et "q" est la durée du retard :

Action 1= X_1. $t/X_1/5s$

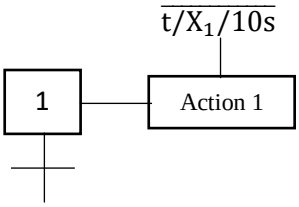


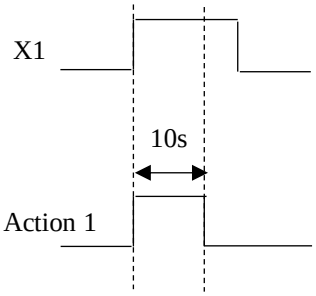


5.5.4 Action continue à durée limitée :

L'exécution de l'action est lancée dès l'activation de l'étape à laquelle elle est associée. Cependant, la durée de cette action est fixée à une valeur donnée.

Action 1= X_1. $\overline{t/X1/10s}$

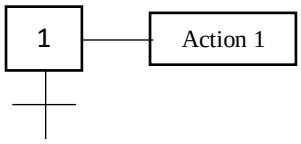


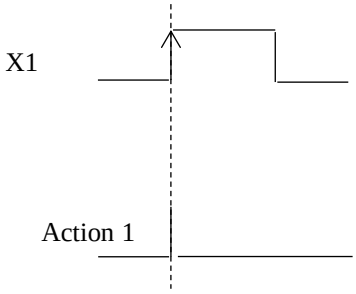


5.5.5 Action impulsionnelle :

L'exécution de l'action dans un laps de temps juste sur le front montant de l'activation de l'étape qui lui est associée.

Action 1= $\uparrow X_1$

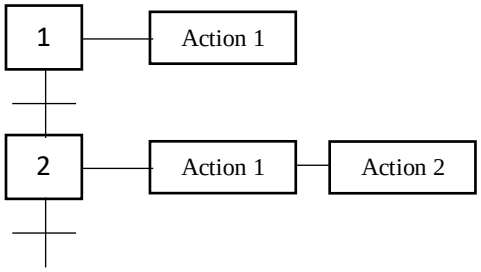


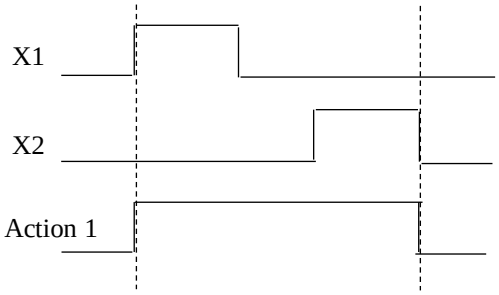


5.5.6 Action maintenu :

Consiste à maintenir une action, sur la durée d'activation de plusieurs étapes consécutives.

Action 1= X1+X2





Biobibliographie :

[1] "Automatismes et régulation - Systèmes à événements discrets : GRAFCET et ses applications" par Pierre Pires.
[2]"Automatismes industriels - GRAFCET, PLC et systèmes à événements discrets" par Jean-Pierre Dufresne.
[3] "L'automatisme et la commande des systèmes : GRAFCET, PLC et applications" par Bernard Dufresne.

Fiche TD 1: GRAFCET

Exercice 1: *(Grafcet à séquence unique)*

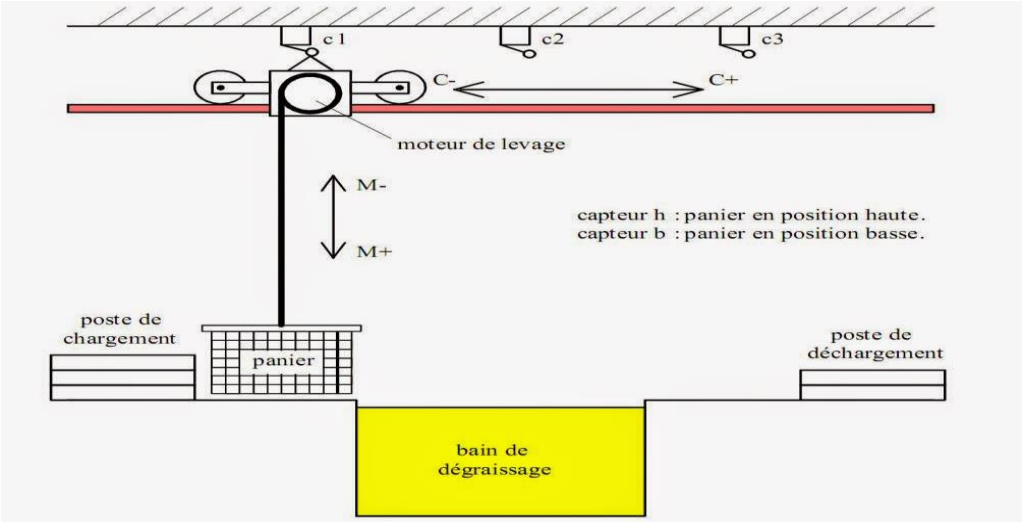


Figure 1: BAIN DE DÉGRAISSAGE

Cycle de fonctionnement:

Le système a pour but de nettoyer des pièces qui sont mises dans un panier en les trempant dans un bain de dégraissage. La chaîne de fonctionnement est comme suit:

- L'appui sur le bouton "dcy" va permettre au chariot de se déplacer jusqu'au capteur "c2", afin de se positionner au-dessus du bac de dégraissage.
- Le panier descend ensuite dans le bain de dégraissage, où il reste 30s.
- Une fois les 30s achevées, le panier remonte. Puis le chariot va jusqu'au capteur "c3" pour le décharger.
- Quand le déchargement est terminé, le système revient à sa position de départ (complétement à gauche et en position haute)

N.B: du fait que le chargement et le déchargement se font d'une manière manuelle, un bouton poussoir "p" est prévus afin de valider le déchargement.

a) En s'appuyant sur le cycle de fonctionnement et sur la figure 1, déterminer dans un tableau les capteurs et les actions.
b) Donner le GRAFCET correspondant.

Exercice 2: *(Grafcet à séquences simultanées : aiguillage en ET)*

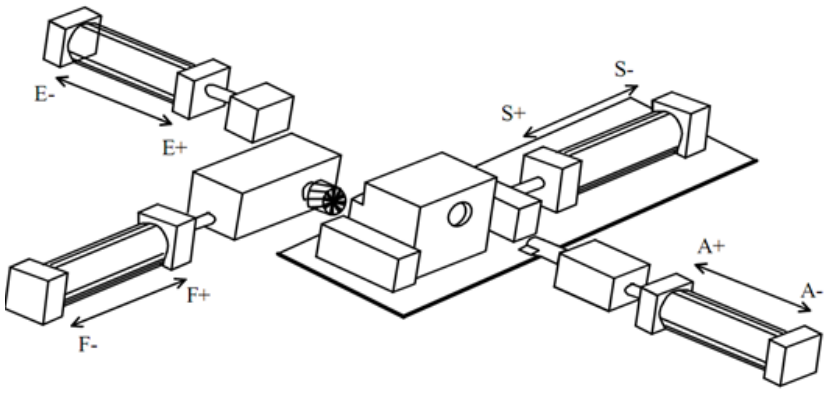


Figure 2: MACHINE SPÉCIALE D'USINAGE.

Cycle de fonctionnement:

La mise en fonctionnement du système illustré par la figure 2 ci-dessus, nécessite la présence d'une pièce "P" et l'appui sur le bouton "dcy". Le fonctionnement du système est comme suit:

- Le vérin "S" avance pour serrer la pièce.
- Puis les deux opérations de fraisage et lamage s'effectue simultanément.
- L'opération de fraisage s'effectue en deux étapes :
    - Le vérin "F" avance en vitesse lente
    - Puis recule en vitesse rapide.
- L'opération de lamage s'effectue en 3etapes:
    - Le vérin "A" avance à petite vitesse.
    - Attend 10s pour avoir un fond plat.
    - Puis retourne à sa position initiale à grande vitesse.
- La pièce est ensuite desserrée.
- En fin, le vérin "E" éjecte la pièce et revient à sa position initiale.

Capteurs	Actions
Dcy : mise en marche du système.	A+, F+ : Avancer lentement le vérin (A, F) respectivement.
a0, e0, f0, s0: vérin A, E, F, S avancé respectivement .	A-, F- : Reculer rapidement le vérin (A, F) respectivement.
a1, e1, f1, s1: vérin A, E, F, S reculé respectivement .	E+, S+: Avancer le vérin (E, S) respectivement.
P: pièce présente dans le montage.	E-, S- : Reculer le vérin (E, S) respectivement.

Exercice 3: *(choix de séquences : aiguillage en OU)*

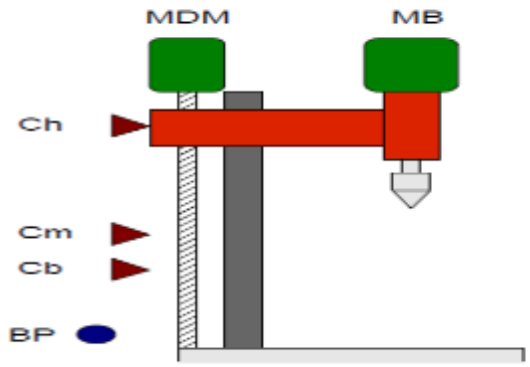


Figure 3.1: VERSION 1.

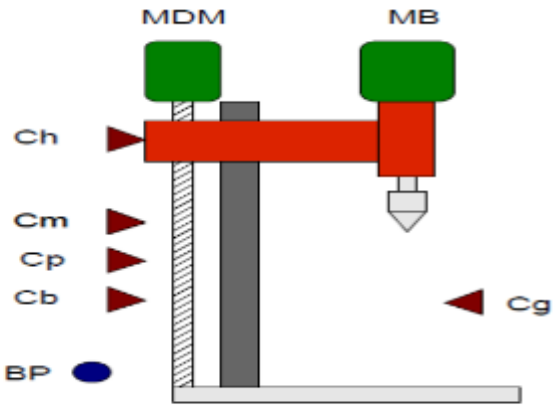


Figure 3.2: VERSION 2.

Figure 3: PERCEUSE AUTOMATIQUE

Cycle de fonctionnement:

Elle réalise un perçage et revient en position initiale. Elle est mise en mouvement par deux moteurs, l'un pour la rotation de la broche "MB" et l'autre pour la montée ou la descente du foret "MDM" (Pour des raisons de simplicité, on ne tiendra pas compte du fonctionnement des deux moteurs).

Partie 1: Lorsque la pièce à usiner est en place (détecté par le capteur "p"), l'opérateur lance le cycle par action sur "BP". La broche se met en rotation (RB) et elle descend à grande vitesse (DGV) jusqu'à "Cm",

ensuite elle poursuit sa course à petite vitesse (DPV) jusqu'à "Cb". Puis elle remonte à grande vitesse (MGV) pour s'arrêter en Ch. **(voir Figure 3.1)**

Partie 2: Le fonctionnement de la perceuse est transformé pour traiter deux sortes de pièces, des grandes et des petites. Les pièces hautes sensibilisent le capteur "Cg" contrairement aux basses.
Pour les pièces basses, le fonctionnement est le même. Cependant pour les pièces hautes, la décente de la broche se fait toujours à petite vitesse et s'arrête en "Cp" puis la broche remonte à petite vitesse afin de retrouver sa position initiale. **(Voir figure 3.2)**

En s'appuyant sur le cycle de fonctionnement et la figure 3, définir pour les deux parties :

a) Les capteurs et les actions dans un tableau.
b) Le GRAFCET correspondant.

Exercice 4: *(Reprise de séquence)*

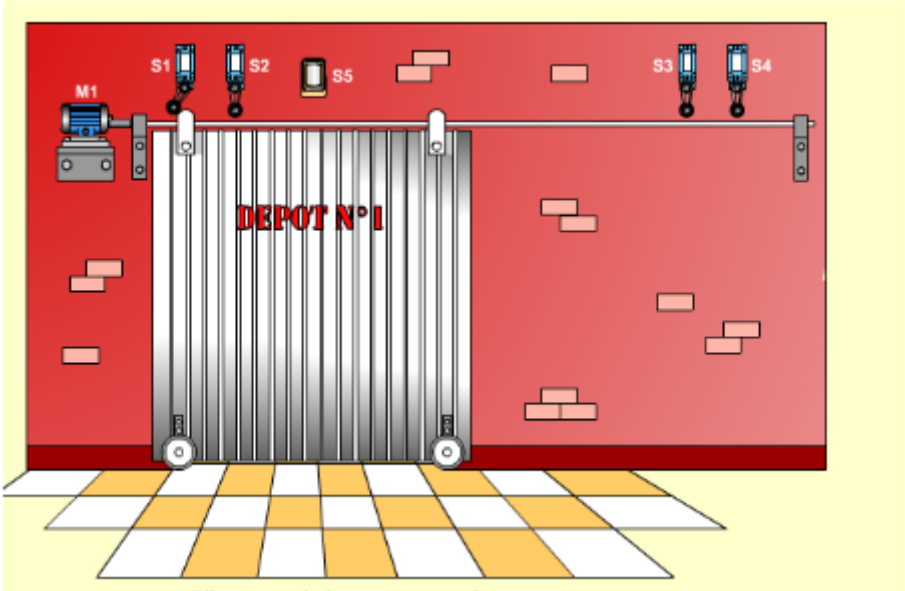


Figure 4: PORTE COULISSANTE.

Cycle de fonctionnement:
Le mode de fonctionnement de la porte coulissante représentée par la figure 4 est comme suit:

- La présence d'une personne devant la porte est détectée par le capteur "S5".
- La porte s'ouvre alors en deux temps:
  - Elle s'ouvre à grande vitesse (OGV) jusqu'au capteur "S3".
  - Puis elle continue son ouverture à petite vitesse (OPV) jusqu'au capteur "S4".
- La porte reste ouverte pendant 15 Seconde, puis au moment de sa fermeture :
  - Si une personne se présente à nouveau devant la porte, elle reste ouverte 15s de plus.
  - Sinon elle se referme en deux temps: d'abord en Grande Vitesse (FGV) jusqu'au capteur "S2", puis en petite vitesse (FPV) jusqu'au capteur "S1".

a) En s'appuyant sur le cycle de fonctionnement et la figure 4, déterminer dans un tableau les capteurs et les actions.
b) Donner le GRAFCET correspondant.

Exercice 4:

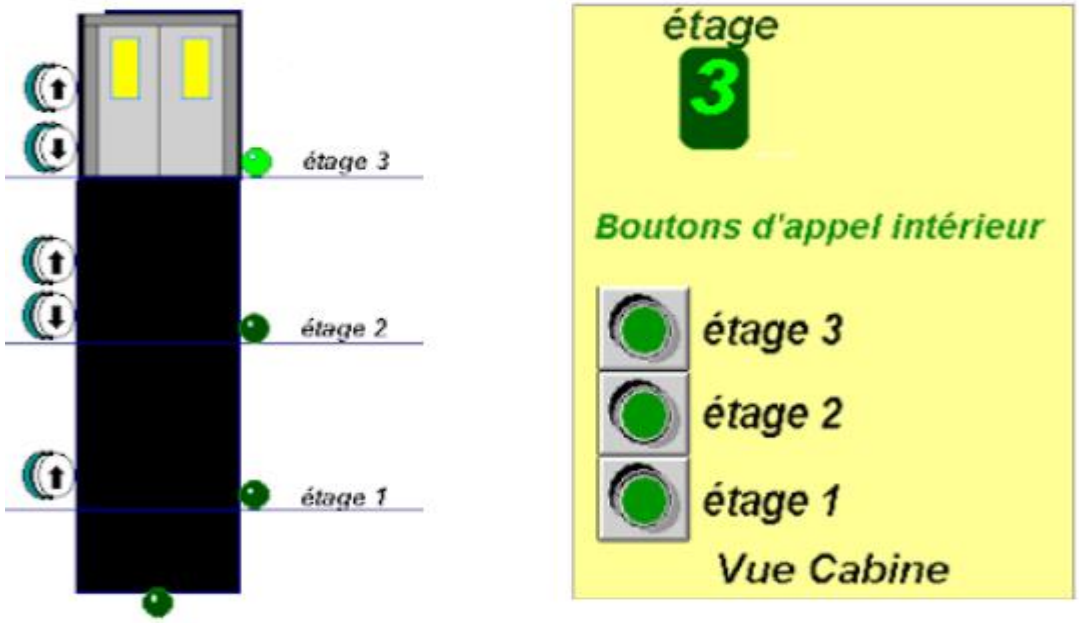


Figure 5: ASCENSEUR.

Le but de cet exercice est de modéliser en utilisant un GRAFCET le fonctionnement d'un ascenseur à 3 étages.
N.B: seul les appels de l'intérieur de la cabine sont étudiés.

Capteurs	Actions
E1: Cabine au premier étage.	MC: Monter la cabine
E2: Cabine au deuxième étage.	DC: Descendre la cabine
E3: Cabine au troisième étage.	FC: Fermeture de la cabine
PF: Porte fermée.	OC: Ouverture de la cabine
PO: Porte Ouverte.	
Etage 1: bouton poussoir d'appel d'étage 1	
Etage 2: bouton poussoir d'appel d'étage 2	
Etage 3: bouton poussoir d'appel d'étage 3	

a) Définir le GRAFCET correspondant.

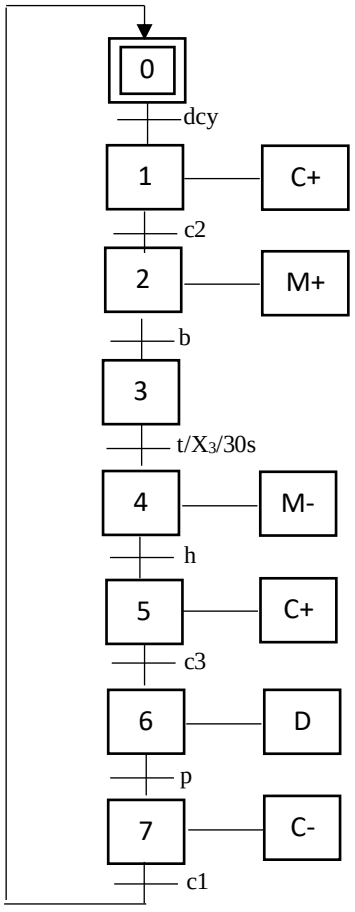
Solution Fiche de TD 1: GRAFCET.

Exercice 1:

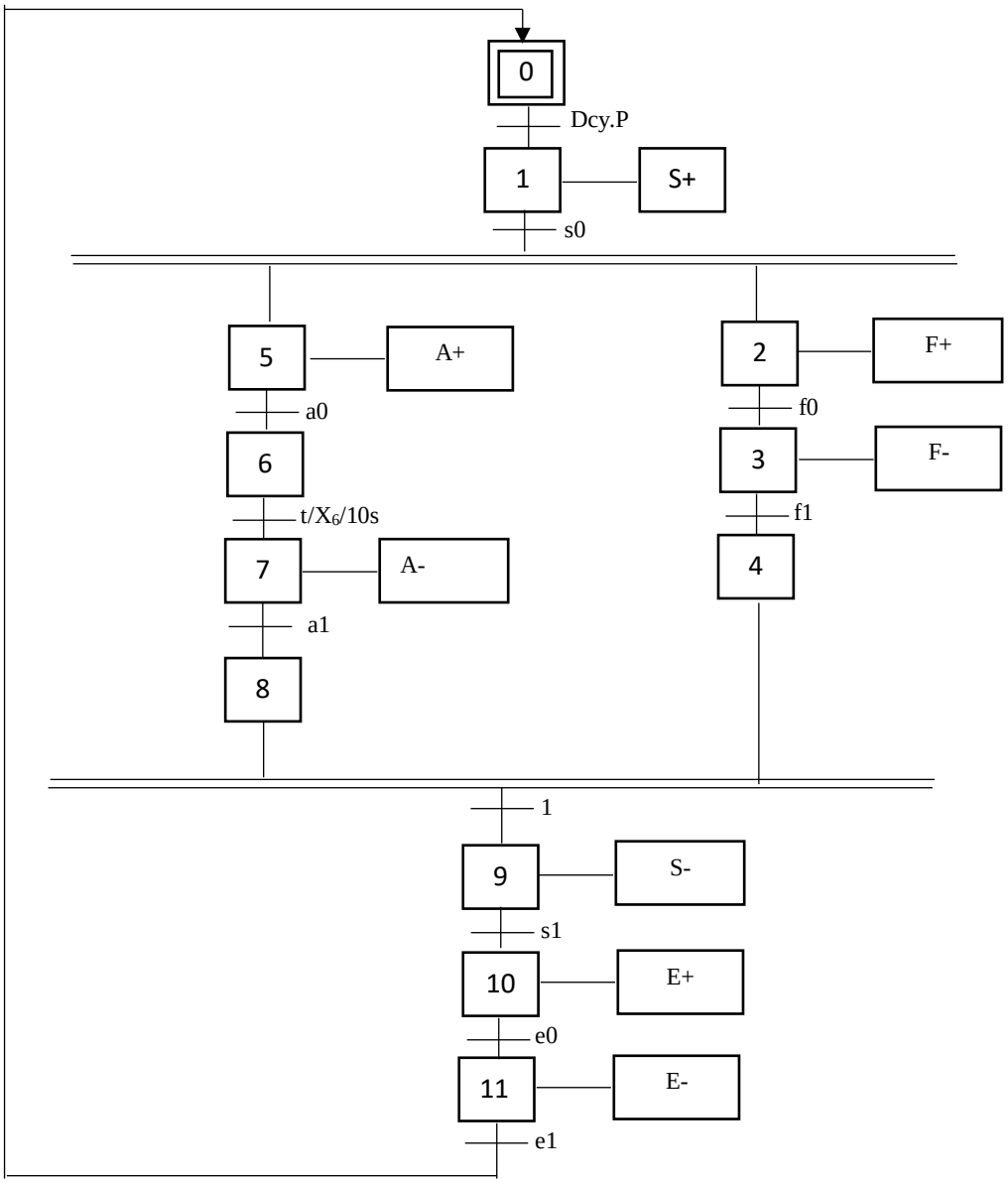
a) En s'appuyant sur le cycle de fonctionnement et la figure 1, déterminer dans un tableau les capteurs et les actions.

Capteurs	Actions
dcy	M+: descendre le panier
c1	M-: Remonter le panier
c2	C+: Déplacer le chariot à droite
c3	C-: Déplacer le chariot à gauche
h	D: Déchargement
b	
p	

b) Donner le GRAFCET correspondant.



Exercice 2:



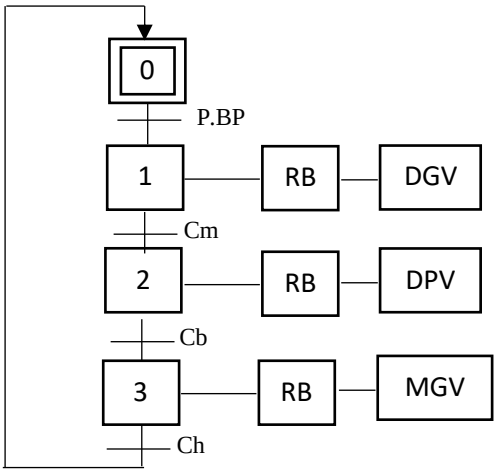
Exercice 3:

Partie1:

a) Les capteurs et les actions dans un tableau.

Capteurs	Actions
P	RB: mettre la broche en rotation
BP	DGV: Descendre à grande vitesse
Cm	DPV: Descendre à petite vitesse
Cb	MGV: Monter à grande vitesse.
Ch	

b) Le GRAFCET correspondant.

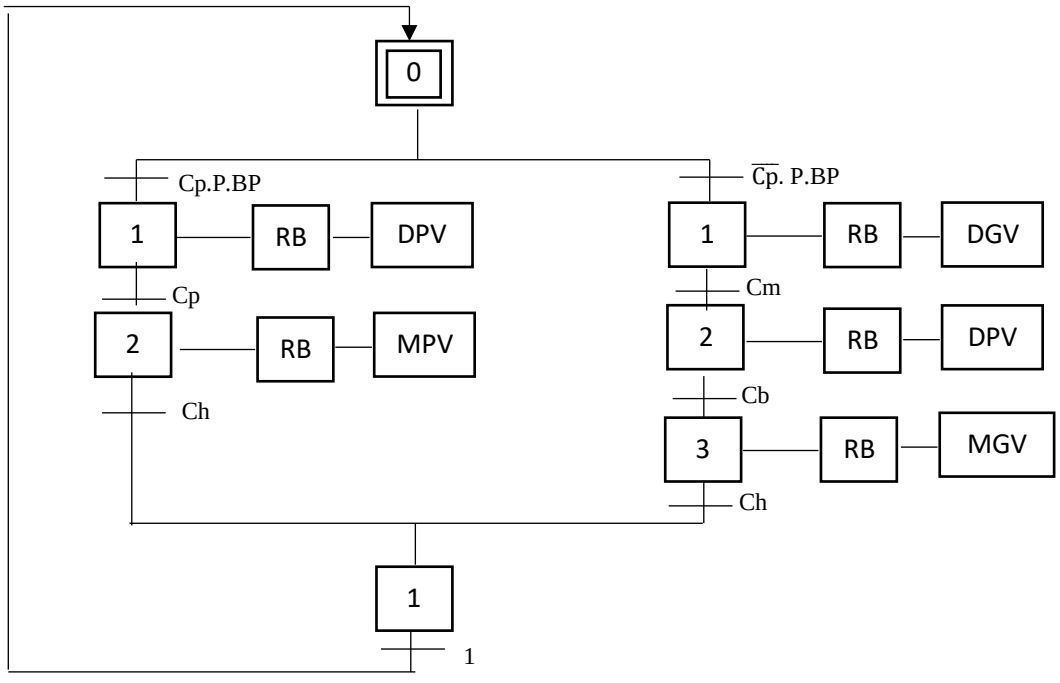


Partie1:

a) Les capteurs et les actions dans un tableau.

Capteurs	**Actions**
P	RB: mettre la broche en rotation
BP	DGV: Descendre à grande vitesse
Cm	DPV: Descendre à petite vitesse
Cb	MGV: Monter à grande vitesse.
Ch	**MPV: Monter à petite vitesse.**
Cg	
Cp	

b) Le GRAFCET correspondant.

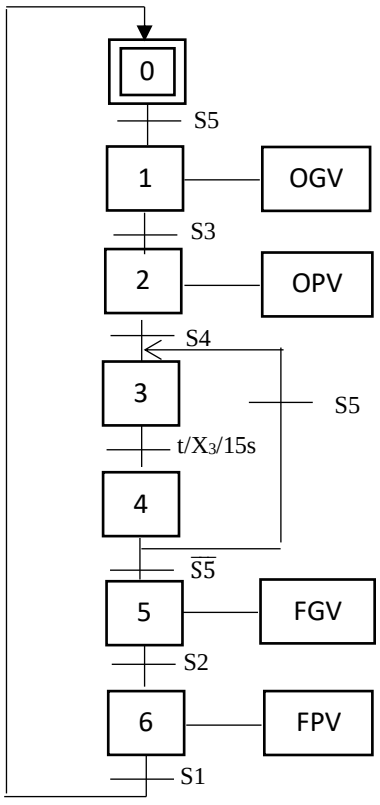


Exercice 4:

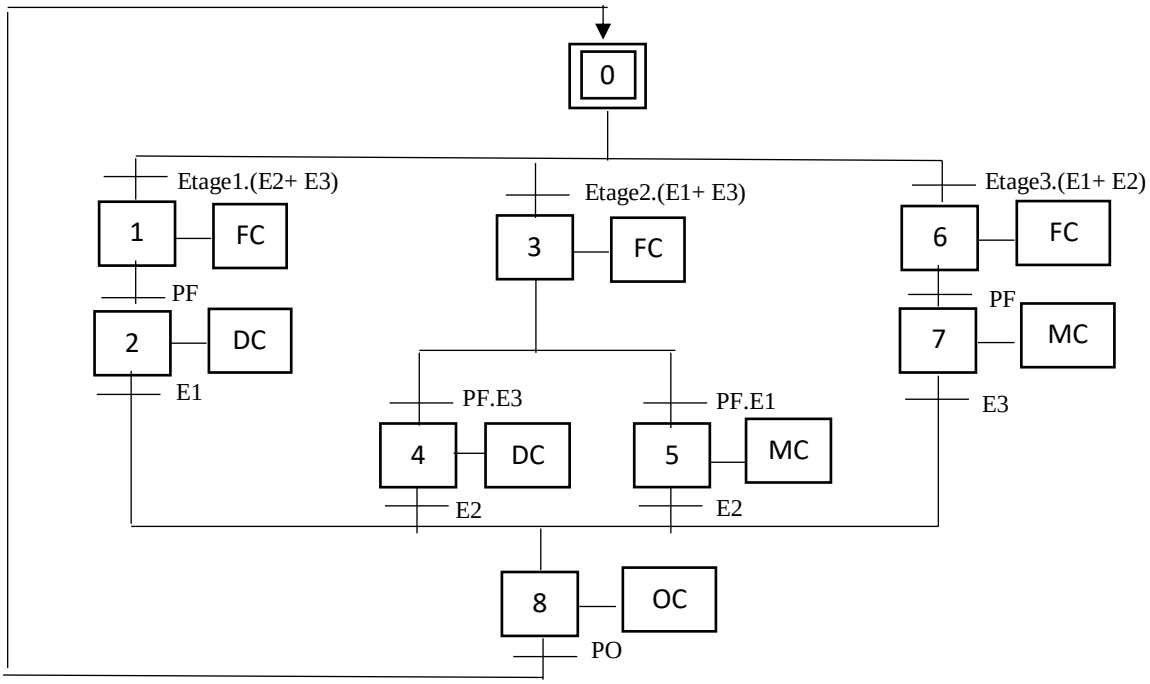
a) En s'appuyant sur le cycle de fonctionnement et la figure 4, déterminer dans un tableau les capteurs et les actions.

Capteurs	Actions
S5	OGV: Ouverture de la porte à grande vitesse
S3	OPV: Ouverture de la porte à petite vitesse
S4	FGV: fermeture de la porte à grande vitesse
S2	FPV: fermeture de la porte à petite vitesse
S1	

b) Donner le GRAFCET correspondant.

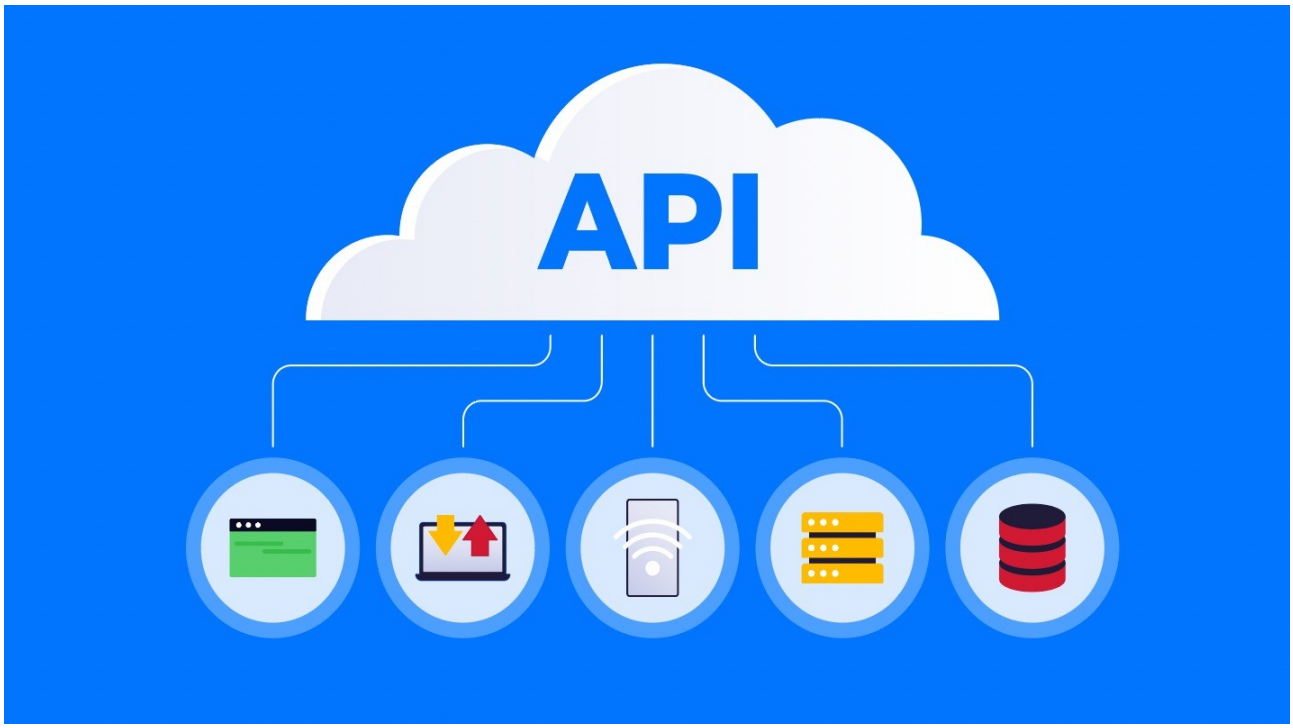


Exercice 5:


0
Etage1.(E2+ E3)
Etage2.(E1+ E3)
Etage3.(E1+ E2)
1
FC
3
FC
6
FC
PF
PF
2
DC
7
MC
E1
PF.E3
PF.E1
E3
4
DC
5
MC
E2
E2
8
OC
PO

CHAPITRE 6: LES AUTOMATES PROGRAMMABLES INDUSTRIELS (API).

"Chaque leçon est une occasion de grandir, chaque erreur, une chance de progresser."



CHAPITRE 6 : LES AUTOMATES PROGRAMMABLES INDUSTRIELS (API)

6.1 Introduction :

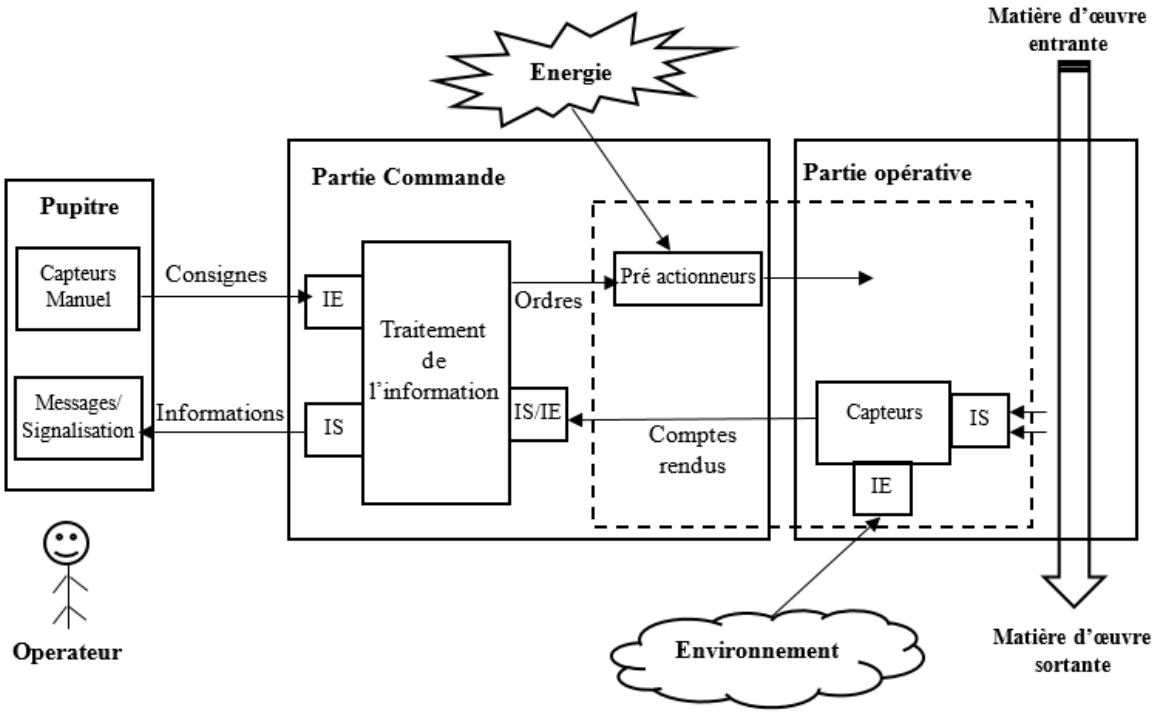


Figure 1 **:** Structure générale d'un système automatisé

Comme mentionné dans le cours 1, Un système automatisé est constitué de 4 parties, à savoir :

1. La Partie Commande (PC),
2. Partie Opérative (PO),
3. Le Pupitre,
4. Les éléments d'interface.

Dans ce cours nous allons nous focaliser sur **la Partie Commande**.

En effet, la PC est considérée comme étant le cerveau de la machine, Elle adresse des ordres à la PO, et des informations à l'utilisateur en fonction des données renvoyées par les capteurs, et les consignes émises par l'utilisateur.

A ces fins, **diverses solutions technologiques** ont été proposées pour permettre à l'UC de **traiter** les données reçues (de l'utilisateur et des capteurs), puis les communiquer à l'utilisateur (informations), et à la PO (ordres). Les solutions proposées sont subdivisées en deux catégories :

a. La logique câblée :

Le traitement est réalisé par des circuits électriques ou des cartes électroniques, qui traduisent le comportement d'une fonction logique séquentielle ou combinatoire.

La logique câblée compte quelques inconvénients, d'où l'intérêt de la logique programmée. Parmi les inconvénients en peut citer :

- Les fonctions sont réalisées par voie matériel (circuit), la taille du circuit accroit avec la complexité du problème. Par conséquent, la logique câblée est utilisée dans des automatismes simples, et utilisée en général pour les traitements de sécurité (arrêt d'urgence par exemple).
- La moindre modification du problème entraine la mise au point d'un nouveau circuit.

b. La logique programmée :

Les schémas électroniques et électriques sont remplacés par une suite d'instructions qui constitue le programme ; ainsi la logique programmée pali aux problèmes de la logique câblée :

- La logique programmée nécessite un minimum de matériel, puisque les fonctions logiques sont directement réalisées par un programme.
- En cas de modification du problème, l'installation ne comporte aucune modification, on se contente d'une modification du programme.

Parmi les unités de traitement programmables : **Les automates programmables industriels (API)**, ces derniers seront détaillés dans la suite de ce cours.

API à usage simple

API à usage complexe

Figure 2 : Exemples automates programmables industriels (API).

Remarque : L'échange de données (entrées/sorties) entres l'unité de traitement et les constituants de l'automatisme s'effectue :

- A l'aide de "câblage" pour la logique câblée.
- A l'aide de "bus de terrain" pour la logique programmée.

6.2 Qu'est-ce qu'un API ?

Appareil électronique programmable (par un personnelle non informaticien), destiné à commander des processus industriels par un traitement séquentiel.

6.3 Les constituants d'un API :

L'API est structurée autour : d'un microprocesseur, d'une mémoire, de dispositifs de communication, d'une console de programmation, et d'une source d'alimentation.

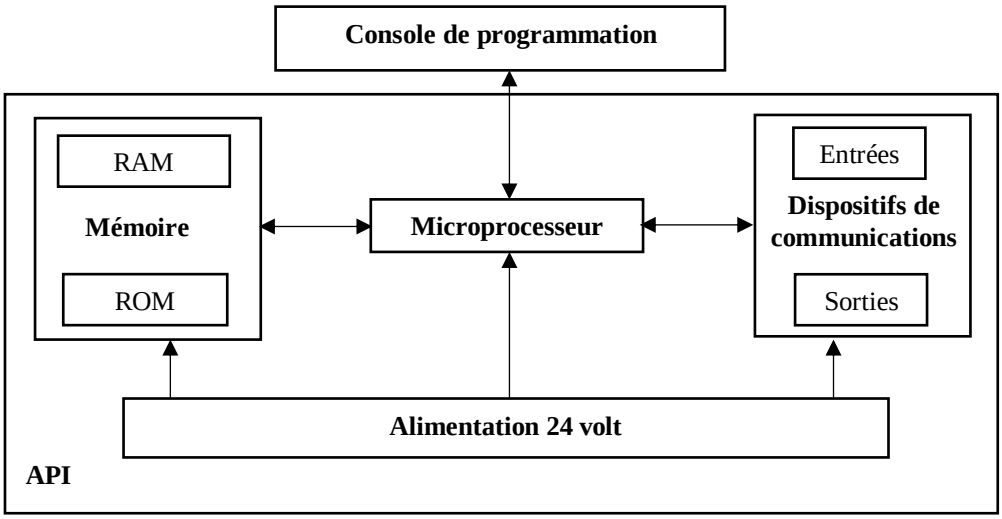


Figure 3 **:** Structure générale d'un API.

6.3.1 Le microprocesseur : Réalise les fonctions logiques et de calcul, mais aussi les opérations de comptage et de temporisation qui sont importantes à la synchronisation du système. De même qu'il organise l'interconnexion entre la mémoire et les interfaces d'entrées/sorties.

6.3.2 Mémoire : contient le système d'exploitation (ROM) ; les instructions du programme à exécuter et les données (RAM). Les données sont reparties en trois catégories :

1. Variables images des entrées.
2. Variables images des sorties.
3. Variables internes : compteurs, temporisateurs, bits internes, etc

La capacité mémoire peut être augmentée, il suffit d'ajouter des barrettes mémoire.

6.3.3 Dispositifs de communication : les interfaces modulaires ou encore les cartes d'entrées/sorties, se subdivisent en :

a) Le module des entrées : comporte un ensemble d'adresses d'entrée où chaque adresse est reliée à un capteur. Le module a pour fonction de transmettre les données issues des capteurs et de l'utilisateur à l'automate.

b) Le module des sorties : comporte un ensemble d'adresses de sortie où chaque adresse est reliée à un pré-actionneur ou un dispositif de visualisation. Le module a pour fonction d'émettre des ordres aux pré-actionneurs et des informations à l'utilisateur.

La modularité des cartes d'entrées/sorties (nombre d'entrées ou sorties) varie entre : 8, 16, 32 vois.

6.3.4 Bus internes : le processeur échange ders informations avec les autres éléments du système via des "Bus" qui véhiculent des informations sous forme binaire.

6.3.5 Module d'alimentation : il permet de distribuer aux différents éléments de l'API une énergie électrique nécessaire au fonctionnement et qui est égale à 24 volts.

6.3.6 Console de programmation : C'est le matériel (ordinateur par exemple) dans lequel il est installé le logiciel de programmation en relation à l'API.

Le dialogue entre le microprocesseur et la console de programmation s'effectue à l'aide du "Module de communication".

6.4 Cycle de fonctionnement d'un API :

Durant son fonctionnement l'API exécute le même cycle de traitement qu'on appelle "cycle automate". Le cycle comporte trois opérations successives qui se répètent comme suit :

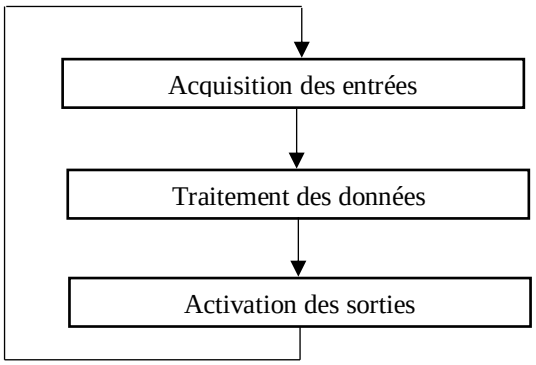


***Figure 4* :** Cycle d'exécution d'un API.

6. 4.1 Acquisition des entrées : la lecture s'effectue en trois étapes :

- Acquisition des données issues des capteurs et consignes des utilisateurs. Ces données sont sous forme : numérique, analogique ou logique.
- Transformer les données reçues en un forma compréhensif par l'API (format binaire).
- Stocker les données transformées en mémoire dans la zone "Image des entrées". Ces données seront nommées "variables image des entrées".

6.4.2 Traitement des données : l'exécution des programmes s'effectue en deux étapes :

- Le microprocesseur exécute les instructions adéquates de la "mémoire programmes" en fonction des "variables image des entrées".
- Les résultats de traitement sont stockés sous forme binaire dans la zone "Image des sorties". Ces résultats sont nommées "variables image des sorties".

6.4.3 Activation des sorties : l'affectation des sorties s'effectue en deux étapes :

- Les "variables image des sorties" sont tout d'abord transférées dans le module de sortie.
- Ces variables sont ensuite converties en signaux électriques pour assurer la commande des pré-actionneurs et les dispositifs de visualisation.

6.5 Critères de choix d'un API :

Le choix de l'API ce fait généralement sur :

- **Le nombre d'entrées/sorties :** plus le nombre d'entrées ou sorties est élevé plus on peut connecter plusieurs capteurs et plusieurs pré-actionneurs.
- **Type de microprocesseur :** les critères d'évaluation du microprocesseur sont relatifs à : la vitesse de traitement, le jeu d'instructions (type d'instructions) qu'il peut offrir, et la taille de sa mémoire.
- **Fonction de communication :** l'API doit pouvoir communiquer avec les autres systèmes de commande, et offrir des possibilités de communication avec des standards normalisés (Bus de terrain).

6.6 Langage de programmation d'un API :

La programmation d'un API consiste à traduire le fonctionnement que doit entreprendre l'automatisme dans un langage spécialisé de l'automate. On cite ci-dessus cinq langages de programmation des API les plus utilisés :

a. Les langages graphiques :

- LD : Ladder Diagram (Langage à contacts).
- SFC : Séquentiel Function Chart (GRAFCET).
- FBD : Function Block Diagram (Logigrammes).

b. Les langages textuels :

- IL : Instruction List (Listes d'instructions).
- ST : Structured Text (Texte structuré).

TP: Grafcet et API sous AUTOMGEN

Exercice 1:

Partie 1 : On désire simuler un GRAFCET sous AUTOMGEN qui va nous permettre l'allumage et l'extinction d'une lampe.

1) Proposez un **GRAFCET** et simulez ce dernier sous le logiciel **AUTOMGEN.**

Partie 2 : On désire Maintenant modifier le GRAFCET précédent afin que la lampe reste allumée pendant 15 s.

2) Simulez un GRAFCET qui répond à ces modifications.

Partie 3 : On désire cette fois ci utiliser un interrupteur, L'allumage de la lampe s'effectue à la première impulsion donnée par l'interrupteur et l'extinction s'effectue à la deuxième impulsion donnée par le même interrupteur.

3) Reprendre le GRAFCET de la partie 1, puis à l'aide du module **IRIS 2D**, créez un pupitre de visualisation comprenant un interrupteur.

Partie 4 : Dans cette dernière partie on voudrait compléter la partie 3 en ajoutant un API.

4) A l'aide du module **AUTOMSIM**, réaliser la simulation en deux dimension de l'allumage d'une lampe en utilisant un API.

Exercice Supplémentaire :

Ces exercices devraient en temps normal être fait individuellement, puis revus et corrigés aves l'enseignant du TP.

Exercice 1 :

On désire simuler sous AUTOMGEN un grafcet d'un distributeur de boisson, Sachant, que le distributeur peut servir une seule boisson à la fois:

- Coca Cola en maintenant l'appui sur un bouton poussoir "Noir".
- Fanta en maintenant l'appui sur un bouton poussoir "vert".
- Eau en maintenant l'appui sur un bouton poussoir "Bleu".

De même qu'il faut prévoir un interrupteur pour la mise en marche du système.

Exercice 2 :

On désire simuler sous AUTOMGEN un grafcet **pour l'allumage de 3 lampes,** Sachant, que les trois lampes s'allument en parallèle et que la durée d'allumage de chaque lampe et de :

- 10s pour la "lampe 1",
- 15 s pour la "lampe 2" et,
- 5 s pour la "lampe 3".

La mise en marche de ces trois lampes est assurée par un bouton poussoir M.

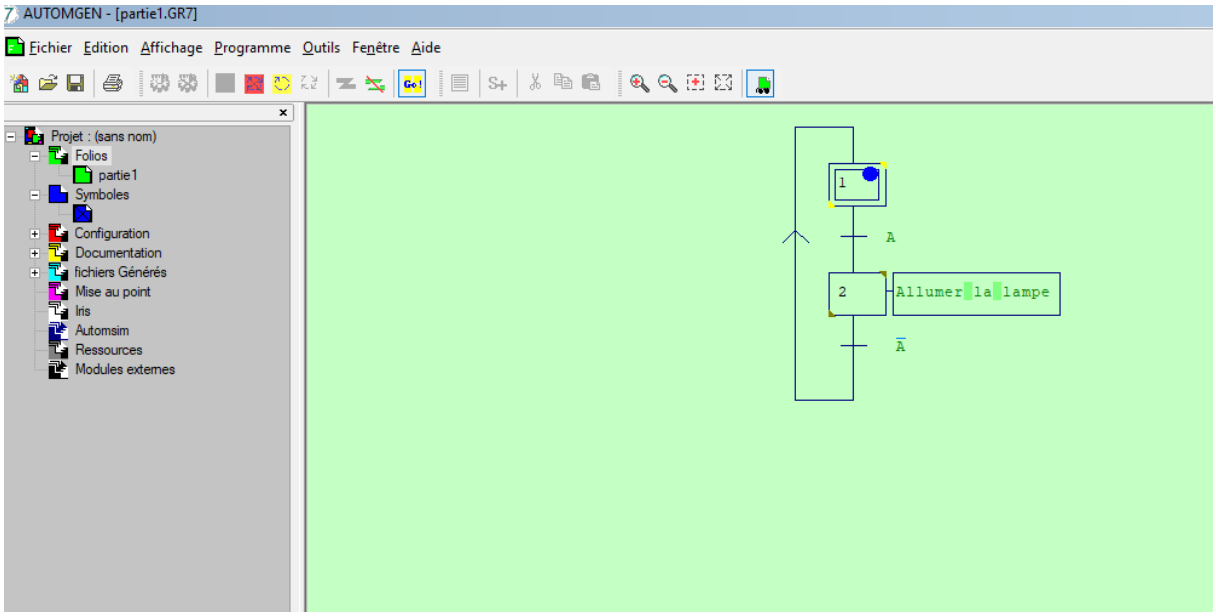
Solution TP : Grafcet et API sous AUTOMGEN

Lien pour Télécharger AUTOMGEN:
http://fractale.gecif.net/si/logiciels/automgen/

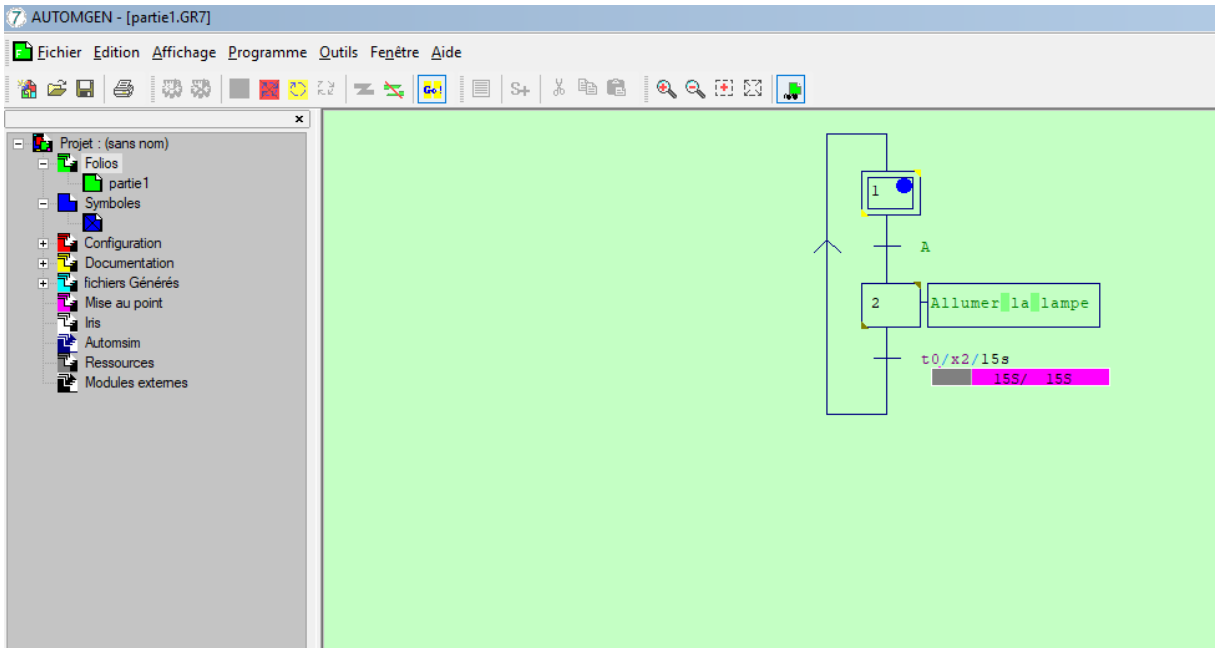
N.B: Afin de voir la solution détaillée avec explication de la résolution de l'exercice 1; merci de consulter les vidéos que j'ai postés sur "Moodle " et que vous pouvez aussi trouver au liens suivants:
https://www.youtube.com/watch?v=HhjRmWtU0bM
https://www.youtube.com/watch?v=Wy3FUhmdWhc

Solution Exercice 1 :

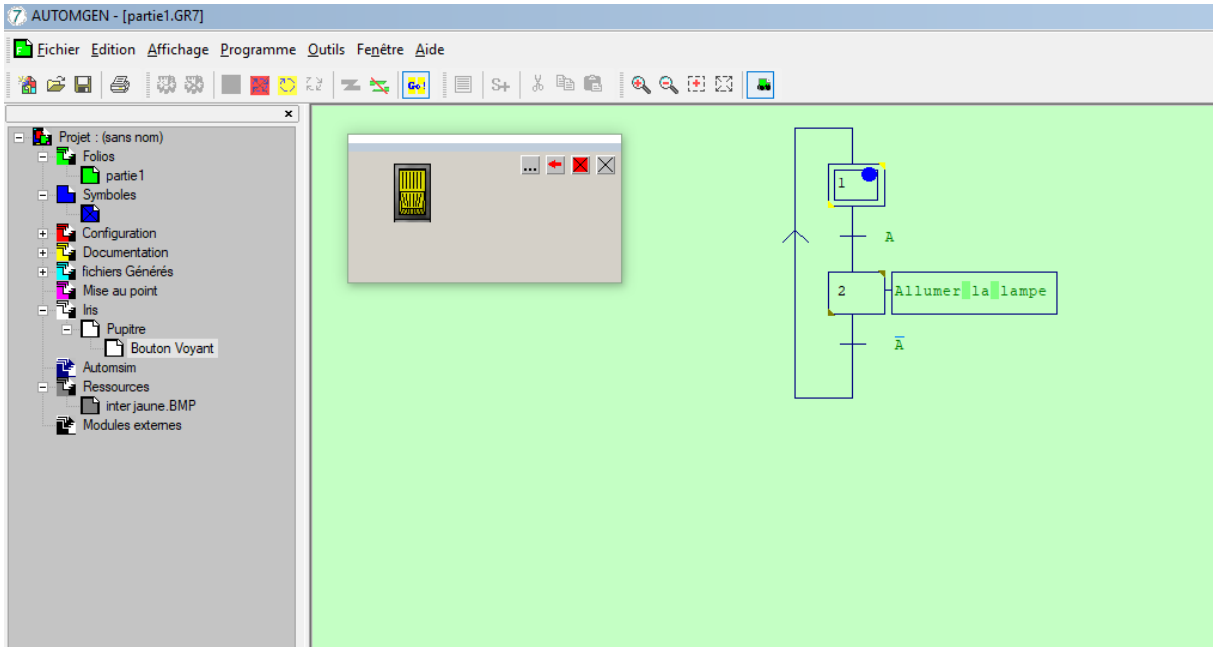
Partie1 :



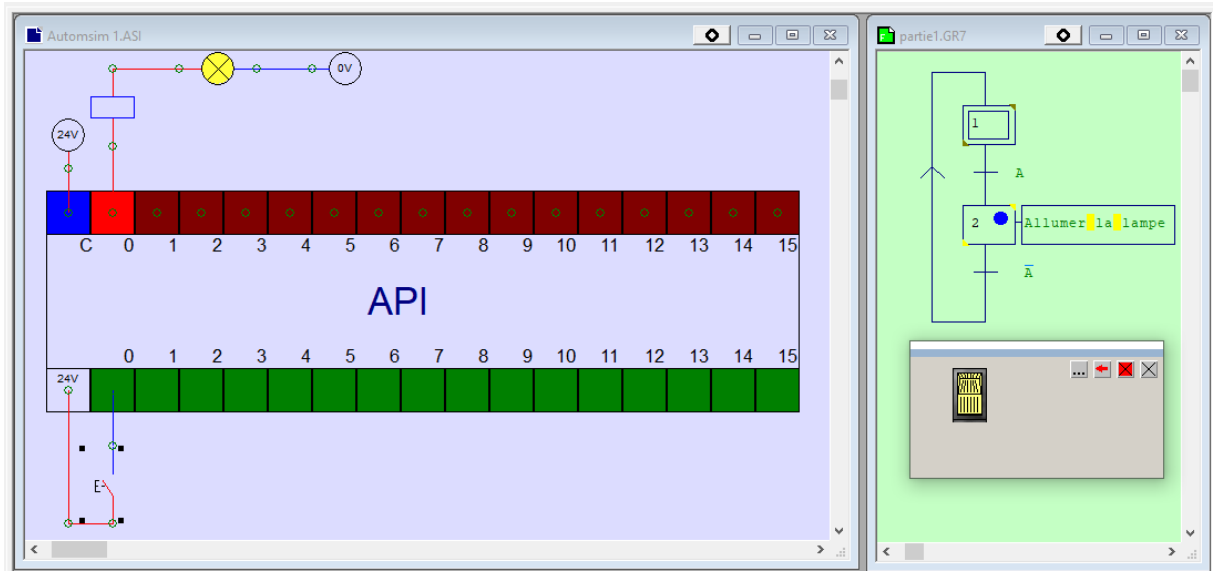
Partie 2 :



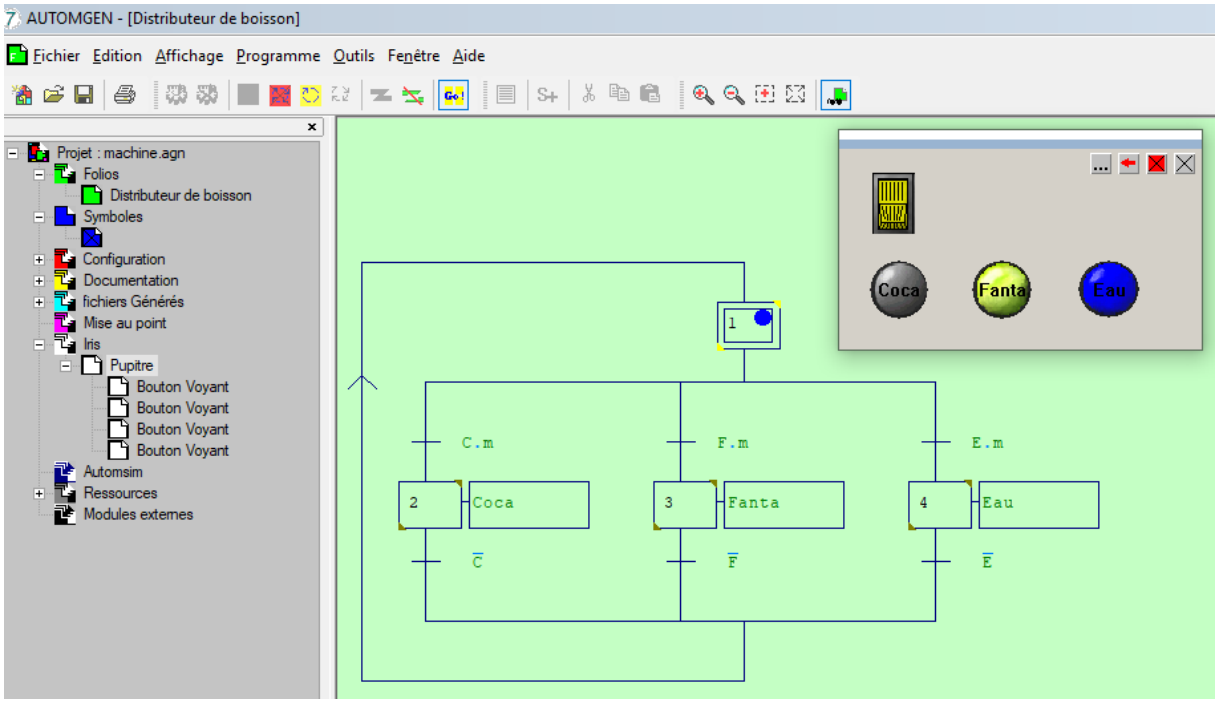
Partie 3:



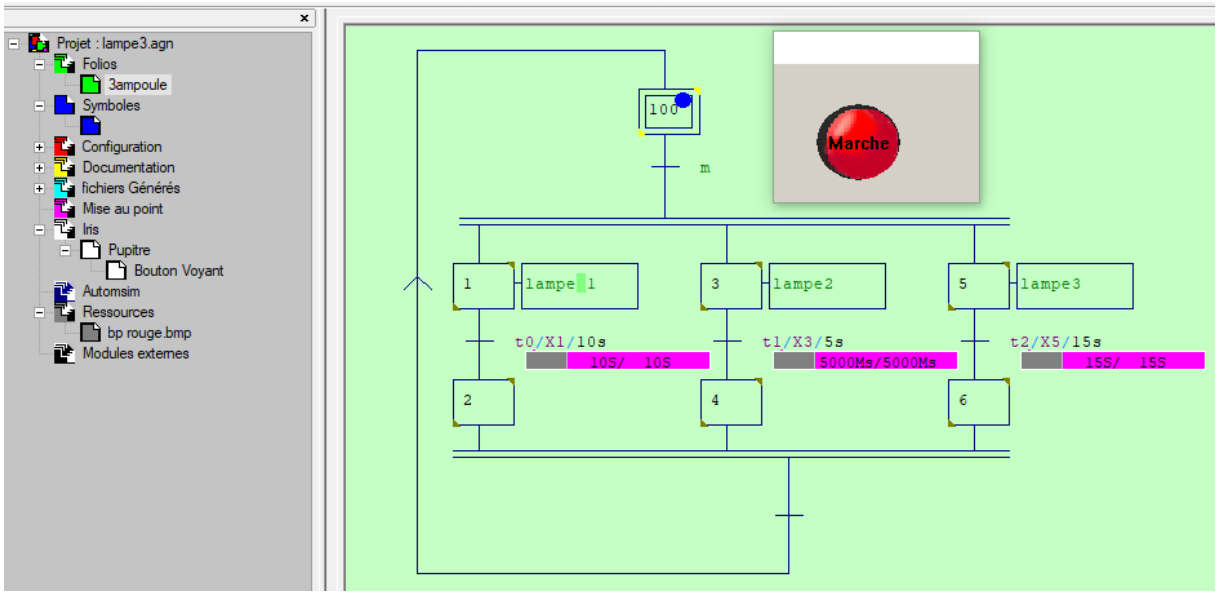
Partie 4:



Exercice 2:



Exercice 3:



Biobibliographie :

[1] "Les automates programmables industriels – Conception et mise en œuvre" par Jean-Pierre Dufresne.
[2] "Automatismes industriels – Automates programmables et régulation" par Michel F. Gosselin.