CHAPITRE I :

INTRODUCTION AUX SYSTEMES SEQUENTIELS

I. Introduction :

Pour les circuits combinatoires, l'état de la sortie ne dépend que de l'état présent des entrées, ce qui limite considérablement le champ de leurs applications. C'est là que les circuits séquentiels prennent toute leur importance. Ces derniers permettent la mise au point de systèmes dont le fonctionnement dépend non seulement des entrées reçues, mais également des informations traitées précédemment. Pour traduire cet effet, on introduit une mémoire permettant au circuit de se souvenir des évènements passés et de traiter l'information plus adéquatement.

II. Définition :

Les systèmes séquentiels peuvent être différenciés en fonction de leur mode de fonctionnement qui peut être synchrone ou asynchrone:

- Dans le mode **synchrone**, les éléments de mémorisation sont des bascules. Les modifications d'état du système ne peuvent donc intervenir qu'à des instants très précis déterminés par des signaux d'horloge.
- Dans le mode **asynchrone**, la fonction de mémorisation est réalisée par de simples boucles de rétroaction. L'évolution des états ne dépend donc que des modifications intervenant sur les entrées $E_i$de la machine**.**

Comme illustré sur la ***Figure 1 et Figure 2***, ces systèmes séquentiels peuvent donc être représentés par deux modèles différents :

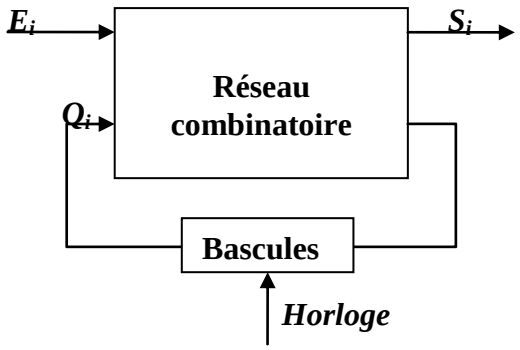


Figure 1système séquentiel synchrone

Figure 2système séquentiel asynchrone

E_i : les entrées de la machine

S_i : les sorties de la machine

Q_i : les états

III. Les machines à nombres finis d'états :

1. Définition

Une machine à états (***M.A.E***.) en anglais Finite State Machine (***F.S.M***.) est un système dynamique, qui peut se trouver, à chaque instant, dans une position parmi un nombre fini de positions possibles. Elle parcourt des cycles, en changeant éventuellement d'état lors des transitions actives de l'horloge. L'architecture générale d'une machine à état est présentée ci-dessous.

2. Architectures des machines

Suivant la façon dont les sorties dépendent des états et des commandes, on distingue deux types de machines à états: les machines de ***Moore*** et les machines de ***Mealy***. Dans les premières les sorties ne dépendent que de l'état actuel, pour les secondes les sorties dépendent de l'état actuel et des entrées.

a. Les machines de Moore

Dans une machine de Moore, la sortie ne dépend que de l'état de la machine : elle ne change que lors d'un changement d'état. Les sorties sont synchrones avec les transitions d'état et les fronts d'horloge.

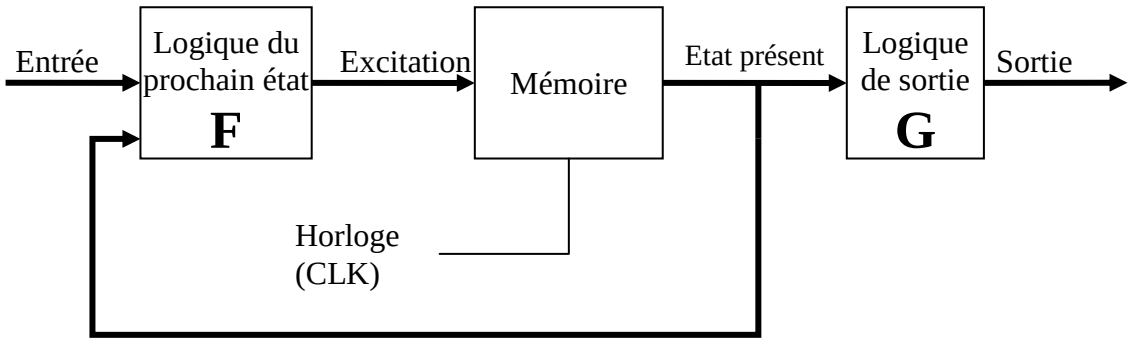


Figure 3*Représentation de la machine de Moore*

Prochain état = **F** (état présent, entrées)

Sortie = **G** (état présent)

b. Les machines de Mealy

Dans une machine de Mealy, la sortie est calculée en fonction de l'état présent et de la valeur présente des entrées : les sorties peuvent changer immédiatement après un changement des entrées, indépendamment de l'horloge.

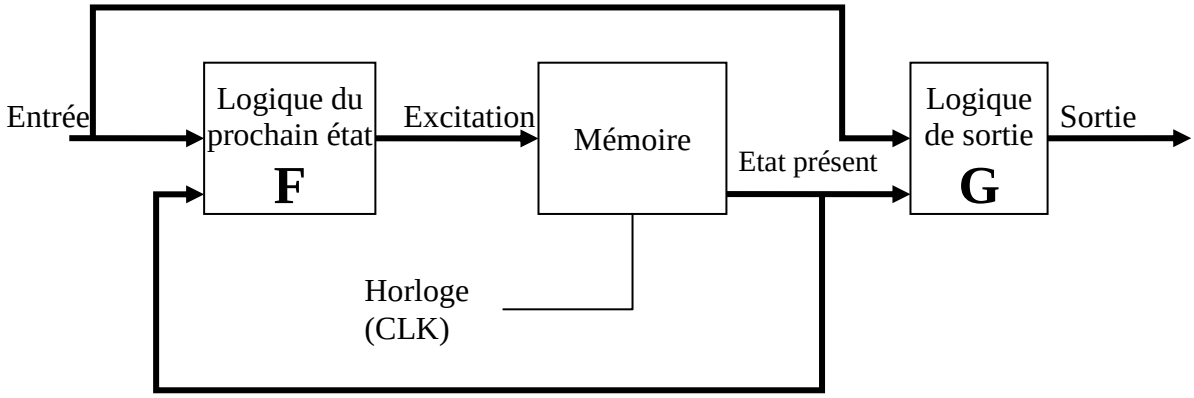


Figure 4*Représentation de la machine de Mealy*

Prochain état = **F** (état présent, entrées)

Sortie = **G** (état présent, entrées)

IV. Méthode de synthèse d' Huffman

Dans cette partie, nous généraliserons les concepts évoqués précédemment afin de permettre le passage d'un cahier des charges quelconque au circuit correspondant. De plus nous établirons des règles de minimisation permettant d'optimiser le nombre de bascules utilisées pour la réalisation du circuit.

La méthode proposée, connue sous le nom de méthode d'Huffman se décompose en plusieurs étapes.

Ces étapes sont les suivantes :

- Modélisation du cahier des charges
  - ✓ Graphe d'état
  - ✓ Table d'état
- Minimisation du nombre d'états
  - ✓ Règles de minimisation
  - ✓ Détermination du nombre de bascules minimum
- Codage
  - ✓ Codage des états

- ✓ Codage des entrées de bascules

➢ Synthèse

- ✓ Synthèse des entrées de bascules et des sorties de la machine
- ✓ Implantation technologique

1. Modélisation du cahier des charges

Le cahier des charges d'un système est généralement donné en langage courant.

Exemple :

Le système considéré a une entrée ***(E)*** et une sortie ***(S)***. Il reçoit sur son entrée des bits arrivant en série. La sortie ***(S)*** doit passer à **1** chaque fois qu'une séquence **010** apparaît sur l'entrée **(E)** puis repasser à **0** sur le bit suivant quel que soit sa valeur.

Pour faire la synthèse d'un tel cahier des charges, la première étape est de le modéliser.

a. Graphe d'état

Le modèle généralement utilisé pour représenter le cahier des charges d'un système est un graphe appelé graphe d'état ou graphe de fluence. Les nœuds de ce graphe représentent les états, un nom symbolique étant affecté à chacun des états. Les arcs du graphe sont orientés. Ils représentent les possibilités de passage entre états. Ces changements d'états se font sur un front d'horloge en fonction des valeurs d'entrée. La structure générale du graphe représentant l'évolution des états d'une machine ayant une entrée ***E*** est représentée sur la **Figure 5**.

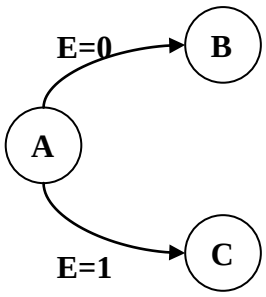


Figure 5*Structure générale du graphe d'état d'un système séquentiel synchrone*

Les caractéristiques de ces graphes sont les suivantes :

- ✓ Chaque état ***(Qi)*** est représenté par un cercle,
- ✓ A chaque état est associé un nom symbolique
- ✓ Le passage d'un état à un autre se fait au coup d'horloge,
- ✓ L'état atteint dépend de l'état de départ et de la valeur des d'entrées ***(Ei)***
- ✓ De chaque état part au plus 2^n arcs, ***n*** étant le nombre d'entrées ***(Ei)***
- ✓ Ce graphe est connexe.

Si l'on considère maintenant la sortie, il faut différencier les deux types de machines que sont machines de ***Moore*** et machines de ***Mealy***. En effet, dans une machine de Moore, les sorties ne dépendent que des états et par conséquent peuvent être consignées à l'intérieur des cercles. Dans une machine de Mealy, les sorties dépendent des états mais également des entrées. Ces sorties doivent donc être consignées sur les arcs du graphe. Ainsi, selon le type de machine, un modèle différent de graphe d'état doit être considéré (**Figure 6**).

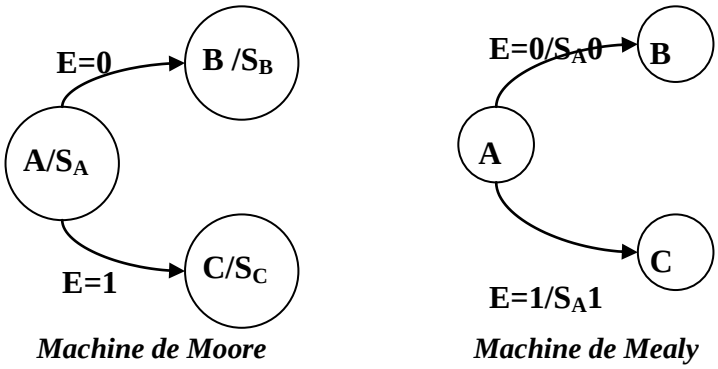


Figure 6*Structure des graphes d'état de Moore et de Mealy*

Exemple :

Selon que le système sera réalisé sous forme de machine de Moore ou sous forme de Machine de Mealy, le graphe d'état représentant le cahier des charges précédent est représenté sur la **Figure 7**.

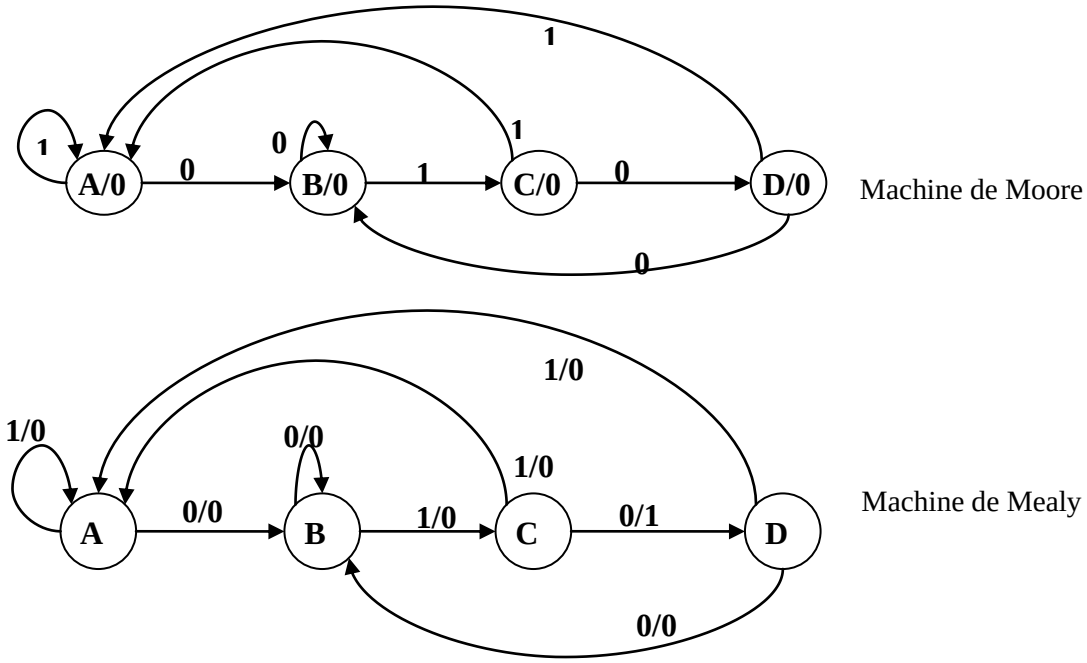


Figure 7*Graphes d'état de la machine détectant les séquences 010*

Remarque:La structure des deux graphes paraît identique. En fait, il est toujours possible de passer d'un graphe de Moore à un graphe de Mealy. Pour cela il, suffit de reporter les sorties associées à chaque état (Moore) sur les arcs arrivant à chacun de ces états. Nous verrons par la suite que l'inverse, c'est à dire passer d'un graphe de Mealy à un graphe de Moore n'est pas toujours possible.

Ce modèle permet de représenter le cahier des charges sous une forme exploitable, mais permet également de figer le fonctionnement du circuit dans tous les cas particuliers non prévus dans le cahier des charges initial.

Dans le cas de cet exemple, le graphe correspond à une machine qui détecte les séquences 010 en mots disjoints (Arc 1 de D à A) et non en mots imbriqués.

A ce niveau de la synthèse, ce qui est important c'est de s'assurer que le graphe correspond bien au cahier des charges. Le nombre d'état utilisés pour représenter ce cahier des charges importe peu. Il sera minimisé par la suite.

b. Table d'état

Le cahier des charges d'un système peut également être modélisé sous une forme tabulaire qui est plus facile à manipuler qu'une représentation sous forme de graphe. Cette représentation tabulaire, appelée table d'état, est directement déductible du graphe d'état. Elle représente les différentes possibilités d'états suivants de chacun des états du système et ceci en fonction des entrées. Les sorties associées à chaque état sont également représentées sur cette table. Elles dépendent ou non des entrées selon qu'il s'agit d'une machine de Mealy ou d'une machine de Moore.

Exemple : Les tables d'état correspondant au graphe de la figure 6 sont présentées sur le tableau.

Etats	Etats suivants		Sortie
	E=0	E=1	
A	**B**	**A**	**0**
B	**B**	**C**	**0**
C	**D**	**A**	**0**
D	**B**	**A**	**1**

Machine de Moore

Etats	Etats suivants		Sortie	
	E=0	E=1	E=0	E=1
A	**B**	**A**	**0**	**0**
B	**B**	**C**	**0**	**0**
C	**D**	**A**	**1**	**0**
D	**B**	**A**	**0**	**0**

Machine de Mealy

Tableau 1 : Tables d'état de la machine détectant les séquences 010

2. Minimisation du nombre d'états

Le nombre d'états de la machine influe directement sur le nombre de bascules nécessaires pour réaliser ce système. Or, le nombre d'états utilisés pour représenter le cahier des charges, que ce soit sur le graphe d'état ou sur la table d'état, n'est pas nécessairement minimum.

a. Règles de minimisation

Deux règles permettent de déterminer les états équivalents et par conséquent de minimiser le nombre d'états nécessaires à la réalisation du circuit.

- ✓ ***Règle R_1*** : Deux états sont équivalents si pour chaque combinaison d'entrée, ils ont mêmes sorties et mêmes états suivants.
- ✓ ***Règle R_2*** : Les états sont regroupés en différentes classes selon les valeurs de sorties associées. Ainsi, deux états ayant mêmes sorties (pour chaque combinaison d'entrée) sont dans la même classe. Les états appartenant à une même classe sont équivalents s'ils ne peuvent être séparés. Or les états appartenant à une même classe doivent être séparés si les états suivants associés à chacun d'eux sont dans des classes différentes.

Lorsque plusieurs états sont équivalents, il suffit de garder qu'un seul représentant par classe d'équivalence et de renommer les états suivants en conséquence.

Exemple : Les règles de minimisation appliquées à la table d'état de la machine de Mealy précédente donnent :

R_1: ***A*** et ***D*** on mêmes sorties et mêmes états suivants, ils sont donc équivalents. L'état ***D*** peut par exemple être éliminé. En renommant les états suivants en conséquence, c'est à dire en remplaçant ***D*** par ***A***, la table d'état devient :

Etats	Etats suivants		Sortie	
	E=0	E=1	E=0	E=1
A	**B**	**A**	**0**	**0**
B	**B**	**C**	**0**	**0**
C	**A**	**A**	**1**	**0**

Tableau 2 : Table d'état réduite

Au sens de la règle ***R_1*** il n'y a pas d'autres états équivalents.

R_2 : les états peuvent être regroupés en deux classes (classe 1 et classe 2).

(1)	(2)	Classes
(A, B)	(C)	Etats
BA	BC	Etats suivants
11	12	Classes des états suivants

Les états A et B doivent être séparés. Il y a maintenant qu'un seul état par classe. Il n'y a donc plus d'états équivalents. Cette machine peut être réalisée avec 3 états.

Remarque : S'il est toujours possible de passer du graphe représentant une machine de Moore à un graphe représentant la même machine en Mealy, l'exemple précédent montre que

l'inverse n'est pas toujours possible. En effet, une machine de Mealy peut comporter moins d'état qu'une machine de Moore.

Le nombre d'états nécessaire à la réalisation d'une machine de Mealy pouvant être inférieur à celui nécessaire à la réalisation d'une machine de Moore, le nombre de bascules peut l'être également. D'où l'avantage qu'il peut y avoir à réaliser une machine de Mealy plutôt qu'une machine de Moore. Ceci dit, les machines de Mealy peuvent avoir des inconvénients liés au fait que les sorties dépendent directement des entrées. En effet, lors du passage d'un état à un autre, les entrées ne doivent pas varier. Il se produit donc un instant entre le changement d'état et le changement d'entrée ou le système se trouve dans le nouvel état mais en présence de l'entrée ayant conduit à cet état, c'est à dire de l'entrée précédente. Puisqu'en machine Mealy, les sorties dépendent directement de l'état et des entrées, elles peuvent donc être soumise à des commutations parasites.

b. Détermination du nombre de bascules minimum

Le nombre minimum d'états "q" étant déterminé, on peut en déduire le nombre minimum "n" de variables d'état et par conséquent de bascules nécessaires au codage de ces états à partir de la double inéquation suivante : $2^{n-1} < q < 2^n$

Exemple : Pour la machine de Mealy précédente, le nombre minimum d'état étant de 3, le nombre de variables d'état nécessaire au codage de ces états est 2. Deux bascules sont donc nécessaires pour réaliser ces systèmes.

3. Codage

a. Codage des états

Chaque état peut être codé par une combinaison de ces n variables d'état. Chaque état doit avoir un code différent des autres mais le codage des états peut être quelconque. Notons toutefois que le codage influence la structure de la future machine et peut donc influencer sa complexité. Le problème de l'optimisation de la machine résultante passe donc par un choix judicieux du codage des états. Ce problème ne sera pas développé ici.

Une fois les états codés, la table d'état peut être exprimée en fonction de ces codes.

Exemple: Nous appellerons les variables d'état (sorties des 2 bascules) de la machine détectant la séquence 010, Q_1 et Q_2. En considérant un codage donné (celui indiqué dans la colonne "Etats"), la table d'état codée de la machine de Mealy correspondant à la table d'état du tableau 2 est représentée sur le tableau 3.

Etats $Q_1Q_2(n)$	Etats suivants E=0	E=1	Sortie E=0	E=1
00 (A)	01	00	0	0
01 (B)	01	11	0	0
11 (C)	00	00	1	0

Tableau 3: Tables d'état codées de la machine de Mealy

b. Codage des entrées de bascules

Le code des états étant déterminé, les entrées de bascules peuvent l'être. Pour chaque bascule i nous connaissons l'état suivant $Q_{i(n+1)}$ (après le coup d'horloge) en fonction de l'état présent $Q_{i\,(n)}$ et des entrées.

Pour réaliser ce système il reste à déterminer les entrées de chaque bascule.

Avec des bascules D, les entrées D_i peuvent être déterminée directement à partir de la relation : $D_{i(n)} = Q_{i(n+1)}$

Exemple: Reprenons la machine de Mealy précédente. Les entrées D_1et D_2 des bascules Q_1 et Q_2 sont représentées sur le tableau 4:

Etats $Q_1Q_2(n)$	Etats Suivants $Q_1Q_2(n+1)$		Sortie		Entrée Bascules $D_1D_2(n)$	
	E=0	E=1	E=0	E=1	E=0	E=1
00(A)	01	00	0	0	01	00
01(B)	01	11	0	0	01	11
11(C)	00	00	1	0	00	00

Tableau4: Détermination des entrées de bascules

4. Synthèse

a. Synthèse des entrées de bascules et des sorties de la machine

Sur la table précédente on dispose des sortie et entrées de bascules exprimées en fonction des entrées et des variables d'état (sorties des bascules). Il suffit donc maintenant d'exprimer les fonctions logiques relatives aux sorties et entrées de bascules.

Exemple: Reprenons la machine de Mealy précédente et déterminons les équations de la sortie S et des entrée de bascules D_1et D_2 (Tableau 5)

Q_1Q_2 \ E	0	1
00	0	0
01	0	1
11	0	0
10	ϕ	ϕ

$D_1=E.\ Q_1$

Q_1Q_2 \ E	0	1
00	1	0
01		1
11	0	0
10	ϕ	ϕ

$D_2=E\overline{\overline{Q_2}}+\overline{Q_1}Q_2$

Q_1Q_2 \ E	0	1
00	0	0
01	0	0
11	1	0
10	ϕ	ϕ

$S=\bar{E}.\ Q_1.\ \overline{Q_2}$

Tableau5: Synthèse de D_1 et D_2

b. Implantation technologique

L'implantation technologique de fonctions logiques est un problème en soit qui sera traité dans un chapitre spécifique. Nous pouvons nous contenter ici, d'une implantation à portes obtenue directement à partir de ces équations déterminée précédemment (Figure 8).

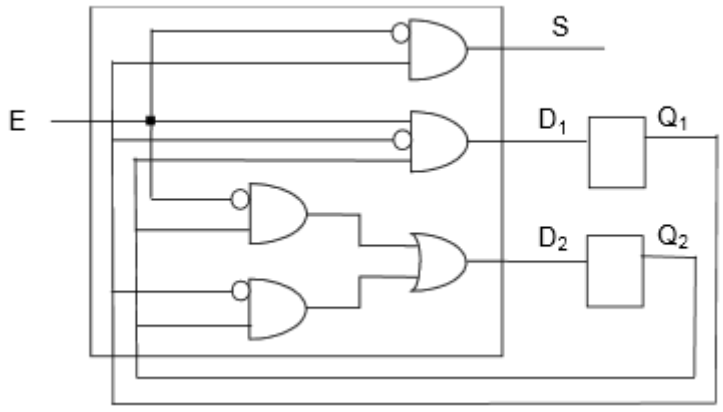


Figure 8*Circuit logique du détecteur de séquence 010*

V. Application

Application : Analyse d'un circuit séquentiel, Machine Mealy.

Soit le circuit suivant :

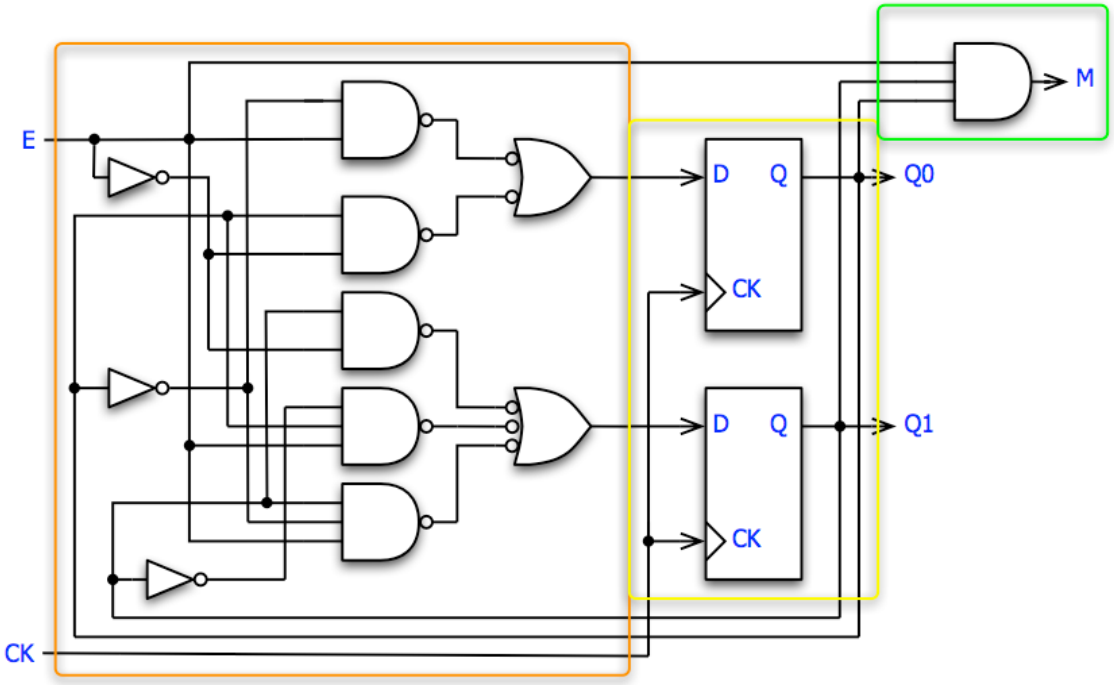


Figure 9*circuit séquentiel, Machine Mealy*

Equations du système:

Généralités à propos des machines d'état:

→ Le prochain état d'une machine est définit comme Q+. On peut aussi utiliser Q(t + 1).

→ La transition d'un état à un autre se fait à chaque coup d'horloge.

Pour déterminer la valeur future d'une bascule, il faut connaître d'abord l'état présent.

Analyse du circuit:

→ Les signaux D_0 et D_1 dans la figure précédente fournissent l'excitation aux bascules D à chaque coup d'horloge.

→ On peut définir, selon le diagramme, des *équations d'excitation*. Ces équations sont des équations qui décrivent les signaux d'excitation en fonction de l'état présent et des entrées. On obtient donc les équations suivantes

$$D_0 = Q_0.\overline{E} + \overline{Q_0}.E$$
$$D_1 = Q_1.\overline{E} + \overline{Q_1}.Q_0.E + Q_1.\overline{Q_0}.E$$

→Dans le cas d'une bascule D, la fonction qui relie la sortie de la bascule à son entrée est (Q+ = D) .Donc les équations qui décrivent le prochain état sont:

$$Q_0^+ = Q_0.\overline{E} + \overline{Q_0}.E$$
$$Q_1^+ = Q_1.\overline{E} + \overline{Q_1}.Q_0.E + Q_1.\overline{Q_0}.E$$

On appelle ces équations les *équations de transition*. Pour le cas de la bascule D, les équations de transition sont faciles à déterminer, puisque la relation entre l'entrée de la bascule et sa sortie est simple. Pour des bascules J-K, par contre, le processus est un peu plus complexe.

→ Pour chaque combinaison d'état présent et d'entrée, les équations de transition nous donnent le prochain état. Chaque état est décrit à l'aide de 2 bits, les valeurs présentes de Q_0 et Q_1: (Q0 Q1) = 00, 01, 10, ou 11. Pour chaque état, il y a seulement 2 entrées possibles, soit E = 0 ou E = 1, donc on a au total **8 combinaisons état/entrée.**

La table d'états

→La prochaine étape est de dessiner la table d'états. Ce tableau donne toutes les combinaisons état/entrée. On a donc, pour chaque état présent, le prochain état pour chaque entrée. Pour le circuit étudié, la table de transition est:

	E	
Q1Q0	**0**	**1**
00	00	01
01	01	10
11	11	00
10	10	11
	$Q1^+Q0^+$	

On peut déterminer la fonction de cette machine à l'aide de la table d'états. Cette machine est un compteur à 2 bits.

- Si E = 0, le compteur demeure au même état.
- Si E = 1, le compte monte de 1 à chaque coup d'horloge.

On peut aussi assigner des *noms d'état* à chaque état :

- 00 = A, 01 = B, 10 = C et 11 = D.

Évidemment, on peut nommer les états de plusieurs différentes façons. Il serait mieux de nommer les états par un nom descriptif qui en indique la fonction. Mais dans ce cas, puisqu'on ne sait pas vraiment la fonction du circuit, on assigne aux états une lettre, tout simplement.

Si on remplace les combinaisons de Q_1 et Q_0 dans la table de transition par les noms d'état, on obtient le tableau d'état suivant:

	E	
S=état présent	**0**	**1**
A	A	B
B	B	C
C	C	D
D	D	A
	S^+=état prochain	

Après la table d'état, il ne reste que la sortie logique de la machine à analyser.

Dans cet exemple, il n'y a qu'une seule sortie, donc on aura une seule *équation de sortie*:

$$M = Q_0.Q_1.E$$

On combine le comportement prédit par cette équation avec le tableau d'état pour produire le *tableau état/sortie*:

	E	
S	**0**	**1**
A	A,0	B,0
B	B,0	C,0
C	C,0	D,0
D	D,0	A,1
	S^+,M	

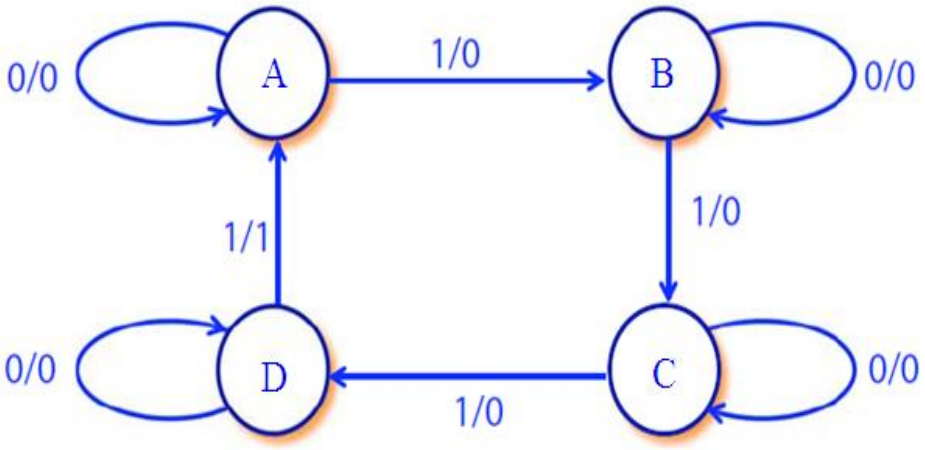
Ce tableau donne toute l'information nécessaire pour comprendre le comportement du circuit. Pour chaque état, le tableau nous donne le prochain état et la sortie en fonction de l'entrée.

Si la sortie aurait été de **type Moore**, le tableau état/sortie est plus simple:

	E		
S	**0**	**1**	**MS=** Q1·Q0 : sortie type Moore
A	A	B	0
B	B	C	0
C	C	D	0
D	D	A	1
	S^+		

On peut aussi représenter l'information de la table état/sortie de façon graphique, à l'aide d'un Graphe des états:

Graphe des états



Ici, 0/0 veut dire que l'entrée = 0, et la sortie = 0. Le premier chiffre désigne la valeur de l'entrée, et le deuxième veut dire la valeur de la sortie. Cette forme n'est utilisée que si on a qu'une entrée et une sortie.

Étapes d'analyse d'une machine d'état:

Donc, en résumé, les étapes pour analyser une machine d'état synchrone sont:

1. Déterminer les équations d'excitation pour les entrées aux bascules
2. Substituer les équations d'excitation dans les équations caractéristiques des bascules pour obtenir les équations de transition.
3. Utiliser les équations de transition pour construire une table de transition.
4. Déterminer les équations de sortie.
5. Ajouter les valeurs de la sortie à la table de transition pour obtenir la table de transition/état.
6. Nommer les états et substituer ces noms pour les combinaisons état/variable dans la table de transition/état pour obtenir la table état/sortie.
7. Dessiner le diagramme d'état.