

Machine à Etat Finis

Rappels: Définition des systèmes électroniques

Un système électronique est caractérisé par

- Ses entrées : e0; ::;; ei; ::;; en
- Son état électrique E
- Ses sorties : s0; ::;; si; ::;; sn

Il existe deux type de systèmes électroniques

- Les systèmes combinatoires construits à l'aide de logique combinatoire
- Les systèmes séquentiels construits à l'aide de logique séquentielle

Machines à états

- Les combinaisons des entrées conduisent à un nombre fini de combinaisons de sortie.
- D'où l'appellation machine à nombre d'états finis ou Machine A Etats.
- Vecteur d'entrée : Combinaison des variables d'entrée
- Vecteur de sortie : Combinaison des variables de sortie

1. Introduction

**Les machines à états finis
(Finite State Machine, FSM)**

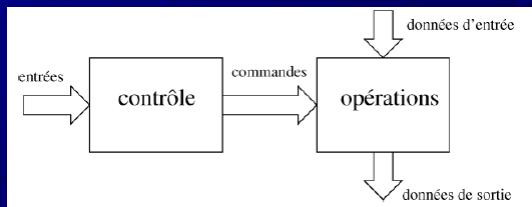
couramment utilisées

modéliser des séquenceurs ou des blocs de contrôle

Signaux à traiter : les données

Signaux pilotant le traitement:
les commandes

Servant exclusivement à générer des signaux de commande.



La machine à état représente la partie **contrôle**, c'est à dire le **cerveau** du système électronique et la partie **opérative**, **les jambes**.

Machines à états

Notations :

- Entrée : E
- Etat Présent : EP
- Etat Futur : EF
- Sortie : S

Définitions

- Etat : Indicateur de position dans le temps
- Registre d'Etat : Composé de bascules permettant de mémoriser les valeurs des états
- Etat Présent : sortie stable du registre d'état à l'instant présent
- Etat Futur : état dans lequel se trouvera la machine après une impulsion

Machines à états

Table de transition : deux parties indiquant le présent et le futur.

- E : Entrée à l'instant n
- EP : Etat Présent à l'instant n
- EF : Etat Suivant à l'instant n + 1
- S : Fonction de Sortie à l'instant n + 1
- Table de transition

E	EP	EF	S
Entrée	Etat Présent	Etat Suivant	Sortie
Avant Impulsion Horloge			Après Impulsion Horloge

Les états de la machine à états

Déf: Les états de la machine à états représentent toutes les valeurs que peuvent prendre les variables internes du circuit de logique séquentielle

Exple: Pour la machine à café

Les états peuvent être



1. Attente de pièce
2. Descendre le gobelet
3. Verser la poudre de café
4. Verser l'eau chaude
5. Indiquer que c'est prêt

Cette machine peut se compliquer en prenant en compte



- Le choix de la boisson,
- Le dosage du sucre

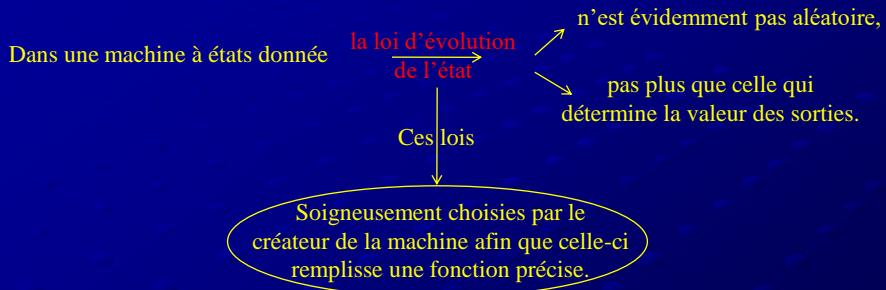
elle reste néanmoins très simple par rapport à certaines machines à états industrielles.



- La conduite d'une centrale nucléaire
- L'automatisation d'un usine de production.

2. Le graphe d'états

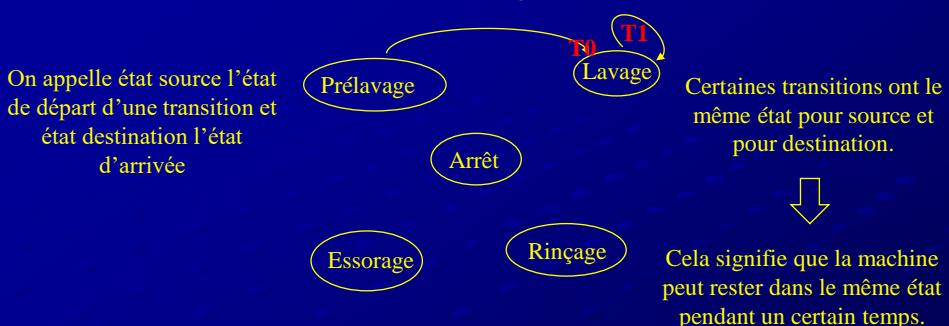
Comment représenter graphiquement le comportement d'une machine à états ?



Le graphe d'états, comme son nom l'indique, représente graphiquement les états d'une machine à états

Chaque état est dessiné sous la forme d'une bulle contenant son nom

Prenons l'exemple d'une machine à laver où on considère 5 états comme illustré dans la figure



On complète le graphe en figurant les transitions possibles par des flèches entre les états.

Muni de toutes les transitions possibles comme représenté dans la figure, le graphe constitue une représentation assez dense de l'évolution possible de la machine au cours du temps. A tout instant la machine est dans l'un des états représentés ;

Le graphe constitue une représentation assez dense de l'évolution possible de la machine au cours du temps.

Pour enrichir encore notre graphe nous devons préciser les spécifications de la machine et, plus particulièrement, la loi d'évolution des variables internes (l'état) en fonction des entrées.

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

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

ENSIAS Rabat Z. ALAOUI ISMAILI 9

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

- prélavage : 10 minutes
- Lavage : 30 minutes
- Rinçage : 10 minutes
- Essorage : 5 minutes

A partir de ces informations complémentaires nous pouvons faire figurer sur le graphe les conditions logiques associées à chaque transition.

lorsque la machine est dans l'état Arrêt elle y reste tant que **M** n'est pas vrai au front montant de l'horloge.

Dès que **M** est vrai \Rightarrow l'horloge la machine change d'état

Elle passe dans l'état Prélavage si **P** est vrai et dans l'état Lavage si **P** est faux.

Notre machine est synchrone sur front montant de l'horloge.

ENSIAS Rabat Z. ALAOUI ISMAILI 10

Or un circuit électronique sans sorties n'est que de peu d'utilité

Notre machine à états possède des entrées mais nous n'avons pas encore étudié les sorties.

celles dont les sorties ne dépendent que de l'état courant  ce sont les machines dites de **Moore**

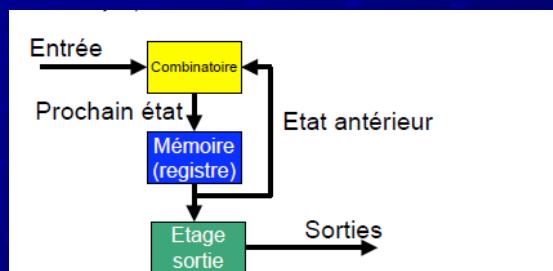
celles dont les sorties dépendent de l'état courant et des entrées  ce sont les machines dites de Mealy

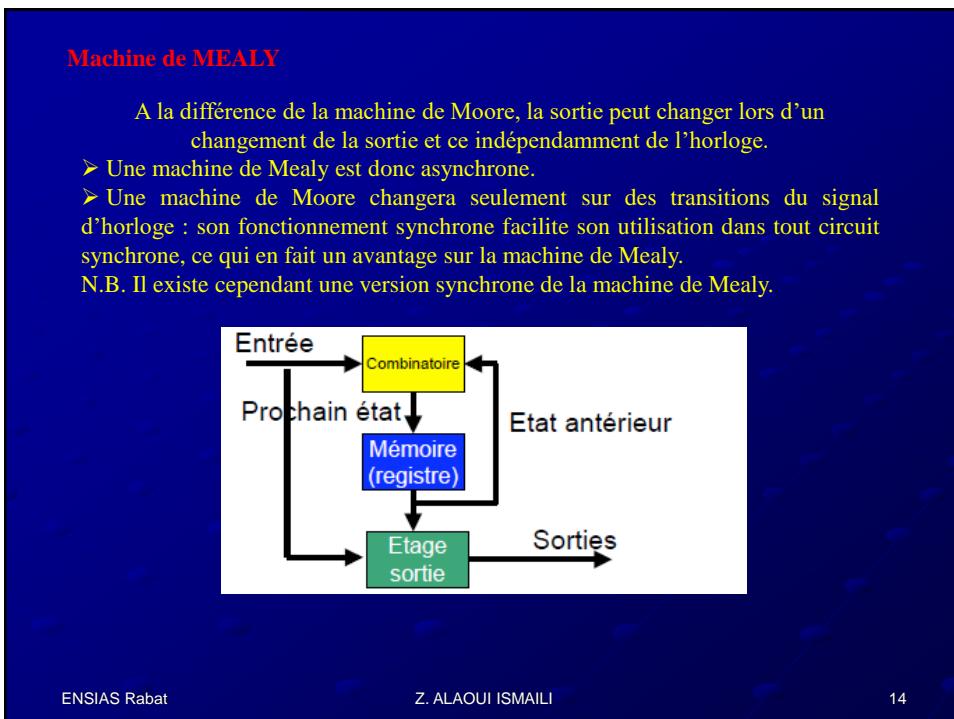
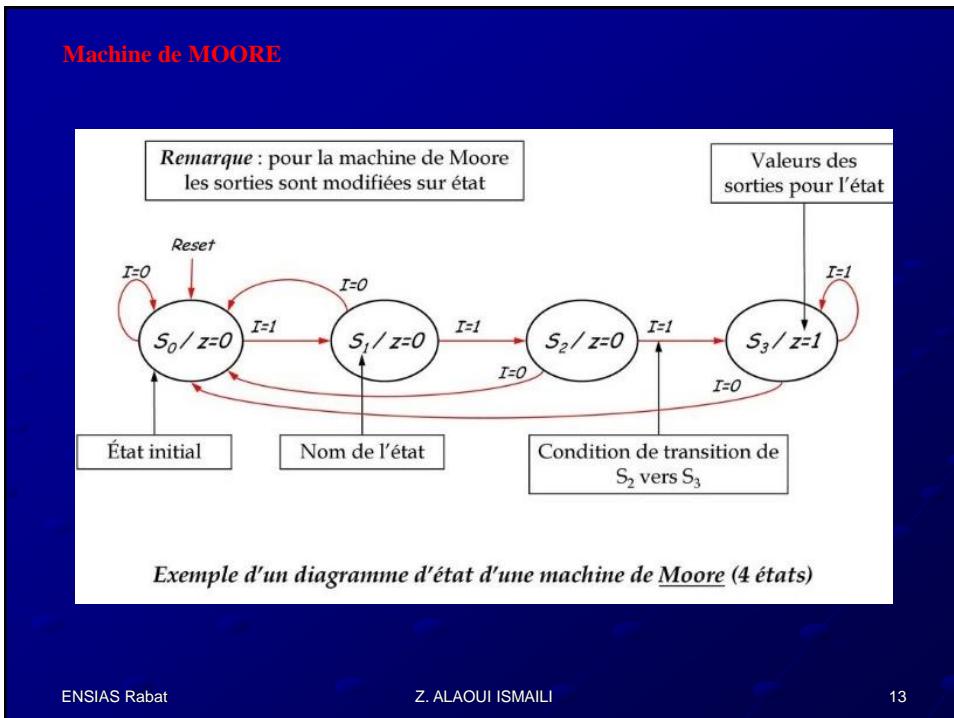
N.B. Le programmeur de notre lave linge est donc une machine de Moore dont les sorties ne dépendent que de l'état courant.

Machine de MOORE

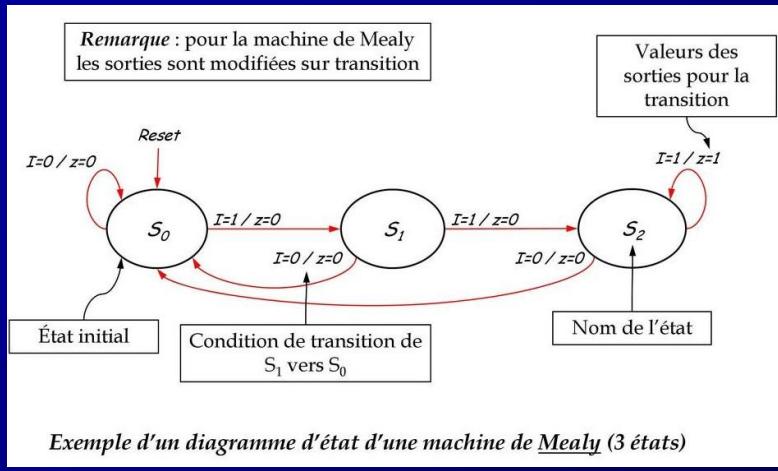
Les sorties dépendent des états présents. Un bloc de logique combinatoire traite les entrées et l'état antérieur des sorties et alimente un bloc de bascules en sortie.

Les sorties changent de manière synchrone sur un front d'horloge.





Machine de MEALY

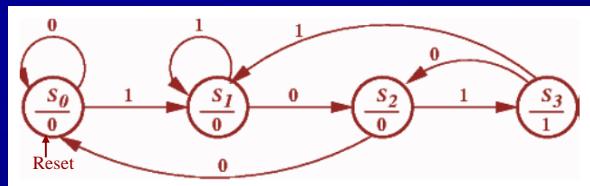


MOORE & MEALY

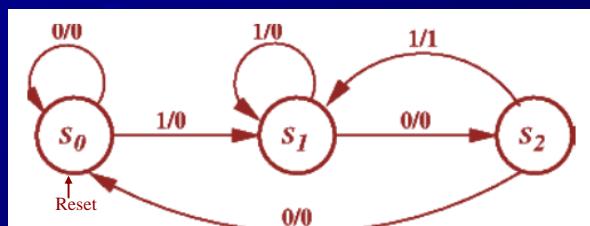
- Machines de Moore et de Mealy peuvent être fonctionnellement équivalentes
 - Une machine de Moore équivalente peut être déduite d'une machine de Mealy et vice-versa
- Machine de Mealy permet une description plus complexe et nécessite un nombre d'état
 - Surface et consommation réduite
- Machine de Mealy : prise en compte immédiate d'un changement en entrée
 - Une machine de Mealy a un temps de réponse réduit d'un cycle par rapport à son équivalent de Moore.
- Une machine de Moore n'a pas de chemin combinatoire liant l'entrée et la sortie
 - Absence d'un chemin critique le plus court

Exemple: Machine à états (de Moore et de Mealy) qui permet de détecter la transition 101 sur une entrée input:

Moore 101 transition



Mealy 101 transition

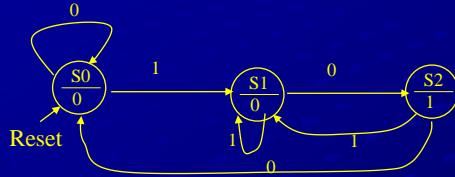


Exercice :

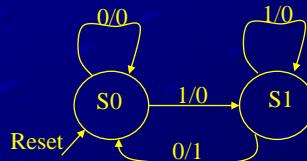
1. Ecrire la machine à états (de Moore et de Mealy) qui permet de détecter la transition 010 sur une entrée input.
2. Même question pour détecter les transitions 01 ou 10.
3. Même question pour détecter la séquence 111
4. Tracer les chronogrammes des entrées et sorties pour chaque cas.

1. Machine à états (de Moore et de Mealy) qui permet de détecter la transition 010 sur une entrée input

Moore transition 10

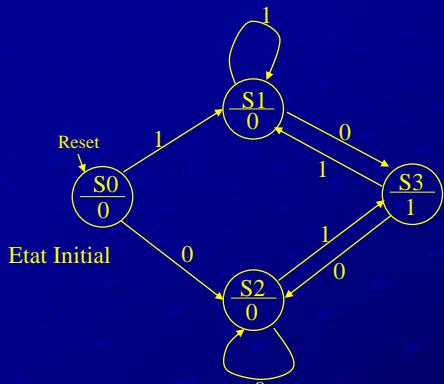


Mealy transition 10

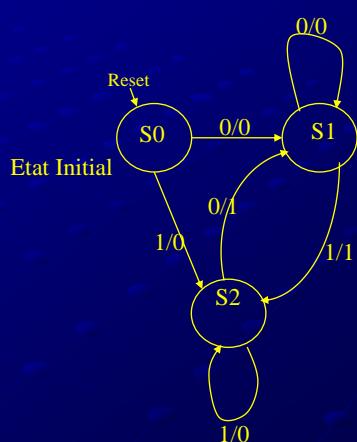


2.b. Détection de la transition 01 ou 10

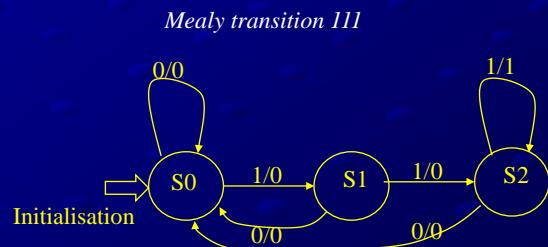
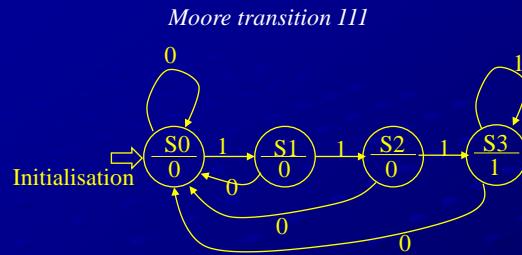
Moore transition 01 ou 10



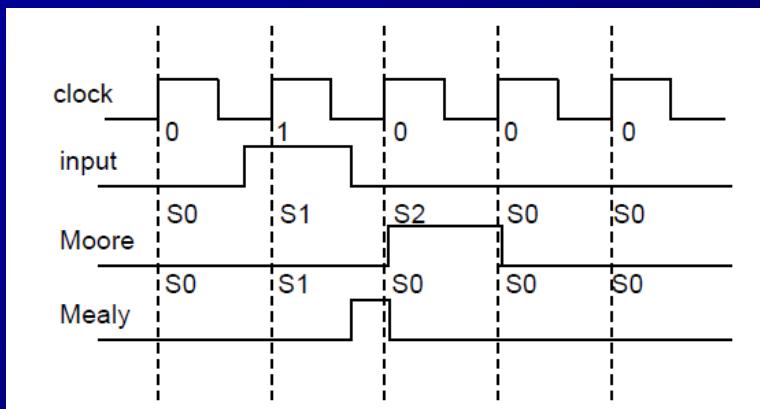
Mealy transition 01 ou 10



3. Détection de la transition III

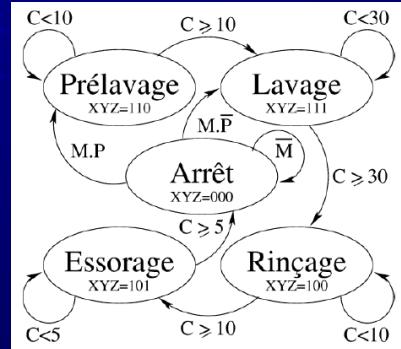


Chronogramme séquence 10 par moore et mealy



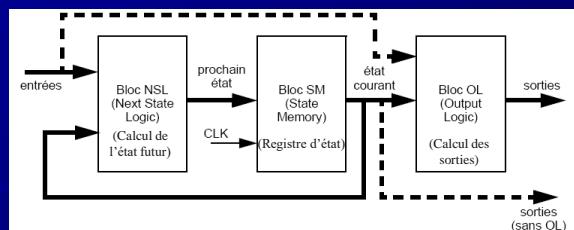
Nous supposerons que ses sorties sont trois signaux booléens, X, Y et Z destinés à piloter les différents moteurs du lave-linge.

Nous pouvons encore compléter le graphe d'états afin d'y faire figurer cette information.



3. Machines à états finis

Une machine à états finis est un circuit séquentiel dont les sorties dépendent d'un état et éventuellement des entrées



le cœur de la machine d'états

Le bloc SM (State Memory) a pour fonction de mémoriser l'état courant ↗ Il a un comportement séquentiel synchronisé sur un signal d'horloge.
de la machine.

Le bloc NSL (Next State Logic) a pour fonction de calculer le prochain état en fonction de l'état courant et des entrées

Le bloc OL (output Logic) a pour fonction la génération des signaux de sortie en fonction de l'état courant et éventuellement des entrées.

La modélisation VHDL d'une machine à états finis doit considérer les points suivants:

- 1. L'utilisation d'un registre d'état explicite ou implicite
 - Dans le premier cas, les états sont énumérés et identifiés par un code explicite
 - Dans le second cas, les états ne sont pas nommés et découlent de la structure du modèle.
 - 2. La répartition en circuits combinatoires et séquentiels.
 - 3. Le choix machine de Moore ou de Mealy
 - Une machine de Mealy offre plus de flexibilité qu'une Machine de Moore → car les sorties peuvent changer dans le cas où le circuit reste dans un même état.
 - Une machine de Moore peut par contre être optimisée de manière à éviter le bloc OL par un codage approprié des états → les sorties sont codées dans les états

ENSIAS Rabat

Z. ALAOUI ISMAILI

25

4. La méthode d'initialisation (*reset*) → qui peut être synchrone ou asynchrone

↓

Un problème relatif est le traitement des états non valides → le circuit ne doit pas être piégé dans un état duquel il ne pourra pas sortir

5. L'encodage des états → La manière de représenter les états (explicites) a un effet important sur les performances du circuit en termes de surface, de délais et de consommation.

La formalisation des différentes formes de machines à états finis utilisera les notations suivantes:

X_i : entrées primaires

Z_i : sorties primaires

S_i : état courant

$f(\dots)$: fonction combinatoire

$f_{clk}(\dots)$: fonction de type flip-flop évaluée sur un flanc d'horloge.

ENSIAS Rabat

Z. ALAOUI ISMAILI

26

3.1. Machine de Moore

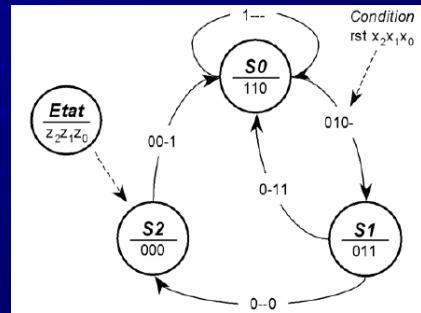
Une machine de Moore traditionnelle est caractérisée par les relations suivantes:

$$\begin{aligned} S_i &= f_{clk}(S_{i-1}, X_i) \\ Z_i &= f(S_i) \end{aligned}$$

La Figure donne un exemple de machine de Moore représentée par un diagramme d'états

Les cercles représentent les états et les flèches représentent les transitions entre les états.

La transition d'un état à l'autre est synchronisée sur le flanc d'un signal d'horloge implicite.



Chaque cercle indique le nom d'un état (en haut) et les valeurs prises par les sorties (en bas) lorsque la machine est dans cet état.

Le Code donne le modèle VHDL correspondant

Le modèle du Code déclare un type énuméré fsm_states à trois valeurs possibles et un signal local state qui représente le registre d'état de la machine.

Les états sont par défauts encodés par des valeurs binaires successives en fonction de l'énumération des états:



L'état "11" est interdit.

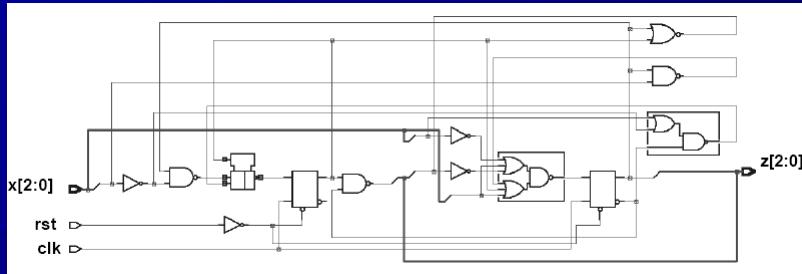
```

library ieee;
use ieee.std_logic_1164.all;

entity fsm is
  port (
    clk, rst : in std_logic;
    x      : in std_logic_vector(2 downto 0); -- entrées primaires
    z      : out std_logic_vector(2 downto 0); -- sorties primaires
  );
end entity fsm;

architecture moore_type of fsm is
  type fsm_states is (S0, S1, S2);
  signal state: fsm_states;
begin
  NSL_SM: process (rst, clk)
  begin
    if rst = '1' then
      state <= S0;
    elsif clk'event and clk = '1' then
      case state is
        when S0 => if x(2) = '1' and x(1) = '0' then
          state <= S1;
        else
          state <= S0;
        end if;
        when S1 => if x(0) = '0' then
          state <= S2;
        else
          state <= S0;
        end if;
        when S2 => if x(2) = '0' and x(0) = '1' then
          state <= S0;
        else
          state <= S1;
        end if;
        when others => state <= S0;
      end case;
    end if;
  end process NSL_SM;
  OL: process (state)
  begin
    case state is
      when S0  => z <= "110";
      when S1  => z <= "011";
      when S2  => z <= "000";
      when others => z <= "----";
    end case;
  end process OL;
end architecture moore_type;
  
```

La Figure montre le circuit synthétisé du code précédent



3.2. Machine de Mealy

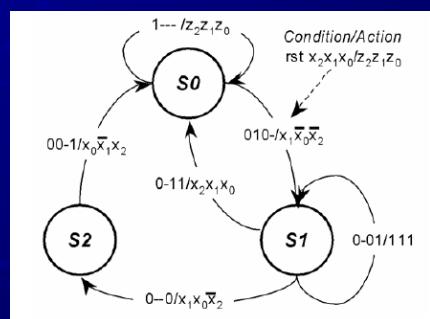
Une machine de Mealy traditionnelle est caractérisée par les relations suivantes:

$$S_i = f_{clk}(S_p, X_i)$$

$$Z_i = f(S_i, X_i)$$

La Figure donne un exemple de machine de Mealy représentée par un diagramme d'états

Les sorties primaires peuvent dépendre directement des entrées primaires et leurs valeurs peuvent changer même si la machine reste dans le même état.



Le Code donne le modèle VHDL correspondant.

```

library ieee;
use ieee.std_logic_1164.all;

entity fsm is
... voir Code 4.36
end entity fsm;

architecture mealy_typ of fsm is
type fsm_states is (S0, S1, S2);
signal state: fsm_states;
begin
NSL_SM: process (rst, clk)
begin
... voir Code 4.36
end process NSL_SM;

```

```

OL: process (state, x)
begin
z <= "----"; -- pour éviter d'inférer des latches
case state is
when S0 => if x(2) = '0' and x(0) = '1' then
z(2) <= x(0);
z(1) <= not x(1);
z(0) <= x(2);
elsif x(1) = '1' and x(0) = '1' then
z(2) <= x(2);
z(1) <= x(1);
z(0) <= x(0);
end if;
when S1 => if x(2) = '1' and x(1) = '0' then
z(2) <= x(1);
z(1) <= not x(0);
z(0) <= not x(2);
elsif x(1) = '0' and x(0) = '1' then
z <= "111";
end if;
when S2 => if x(0) = '0' then
z(2) <= x(1);
z(1) <= x(0);
z(0) <= not x(2);
end if;
when others => z <= "----";
end case;
end process OL;

```

Le changement principal par rapport au modèle de la machine de Moore réside dans le processus OL

→ Le processus est aussi sensible à des événements sur les entrées primaires x et les valeurs des sorties primaires dépendent des valeurs des entrées primaires

La Figure montre le circuit synthétisé du code précédent

