

# Conception de circuits numériques et architecture des ordinateurs

Équipe pédagogique :

Marie Badaroux (TD), Julie Dumas (TD+TP), Florence Maraninchi (TD+TP),  
Olivier Muller (TD+TP), **Frédéric Pétrot** (CM+TD), Pierre Ravenel (TD+TP)  
Eduardo Tomasi Ribeiro (TP), Lionel Rieg (TD+TP),  
Sébastien Viardot (TD+TP Apprentis)



Année universitaire 2024-2025

- C1 Codage des nombres en base 2, logique booléenne, portes logiques, circuits combinatoires
- C2 Circuits séquentiels
- C3 Construction de circuits complexes
- C4 Micro-architecture et fonctionnement des mémoires
- C5 Machines à état
- C6 Synthèse de circuits PC/PO
- C7 Optimisation de circuits PC/PO
- C8 Interprétation d'instructions - 1
- C9 Interprétation d'instructions - 2
- C10 Interprétation d'instructions - 3
- C11 Introduction aux caches

## Intérêt

La description du comportement temporel d'un système nécessite un formalisme précis.

Dans le cas du matériel, on s'attachera à exprimer des séquences d'opérations ordonnées

## Plan détaillé du cours d'aujourd'hui

### 1 Machines avec un nombre fini d'états

- Définitions
- Représentations usuelles en matériel
- Déterminisme

### 2 Implantation matérielle des machines à états

- Encodages
- Structure matérielle

### 3 Specifications

# Plan

## 1 Machines avec un nombre fini d'états

- Définitions
- Représentations usuelles en matériel
- Déterminisme

## 2 Implantation matérielle des machines à états

- Encodages
- Structure matérielle

## 3 Specifications

## Machines à états

Machine à états (*Finite State Machine, FSM*)

But en matériel :

Décrire de manière synthétique un comportement séquentiel

Constituée :

- d'états, en nombre fini  
un état est représentatif des changements effectués sur les entrées depuis le démarage du système
- de transitions entre états  
ces transitions sont étiquetées par des conditions calculées à partir des entrées
- d'actions, visibles en sortie  
ces actions peuvent être associées aux états ou aux transitions entre états

## Machines à états

Plus formellement :

- $I$  est l'alphabet d'entrée
- $O$  est l'alphabet de sortie
- $S$  est un ensemble non vide d'états, et  $s_0$  l'état *initial*
- $\Theta$  est la fonction de transition entre états  
 $\Theta : I \times S \rightarrow S$
- $\Gamma$  est la fonction de sortie  
 $\Gamma : S \rightarrow O$  pour les machines de *Moore*  
 $\Gamma : I \times S \rightarrow O$  pour les machines de *Mealy*

## Graphe de transitions

### Représentation graphique

FSM souvent représentées sous la forme d'un Graphe de transitions ou *State Transition Graph (STG)*

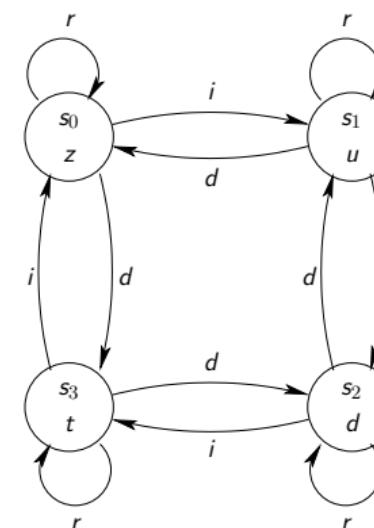
$G(V, A)$  est un graphe orienté

- $v \in V$  est un état, étiqueté avec un nom d'état
- si un ou plusieurs arcs  $(v_i, v_j)$  existent dans  $G$ , alors  $a_{ij}^k \in A$  est une transition,  $k$  identifiant une transition particulière
- les transitions  $a_{ij}^k$  entre états sont étiquetées par des conditions  $c_{ij}^k = t(I)$  dépendantes des entrées
- valeur de sortie notée dans  $v_i$  pour machines de Moore
- valeur de sortie notée sur  $a_{ij}^k$  pour machines de Mealy

## Exemple de STG

### Compteur/décompteur modulo 4

- entrée :  $I = \{r, i, d\}$  soit rien, incrémenter, décrémenter
- états :  $S = \{s_0, s_1, s_2, s_3\}$
- sortie :  $O = \{z, u, d, t\}$ , soit zéro, un, deux, et trois



## Table de transitions

Table de transitions ou *State Transitions Table (STT)*

Représentation tabulaire :

représentation du graphe  $G(V, A)$  sous une forme de table

Utile à la synthèse en vue de l'implantation de  $\Theta$  et  $\Gamma$

- forme générale :

état courant	entrée	état futur	sortie
S0	I0	S1	O0
S0	I1	S2	O3

- question : est-ce une machine de Moore ou de Mealy ?

## Exemple de STT

Compteur/décompteur modulo 4

état courant	entrée	état futur	sortie
$s_0$	$r$	$s_0$	$z$
$s_0$	$i$	$s_1$	$z$
$s_0$	$d$	$s_3$	$z$
$s_1$	$r$	$s_1$	$u$
$s_1$	$i$	$s_2$	$u$
$s_1$	$d$	$s_0$	$u$
$s_2$	$r$	$s_2$	$d$
$s_2$	$i$	$s_3$	$d$
$s_2$	$d$	$s_1$	$d$
$s_3$	$r$	$s_3$	$t$
$s_3$	$i$	$s_0$	$t$
$s_3$	$d$	$s_2$	$t$

## Déterminisme : complétude et orthogonalité

Contrainte en matériel :

Les machines à états sont déterministes :

pour un état initial donné et une séquence d'entrées donnée, c'est toujours la même séquence de sortie qui est produite

Assuré par deux propriétés sur les arcs sortant d'un état

Soit  $Succ_i \subset S$  l'ensemble des successeurs de  $s_i \in S$

- complétude :  $\sum_{s_j \in Succ_i} c_{ij} = 1$

En d'autres termes, le *ou* les conditions sur tous les arcs sortant donnent 1

Cela signifie que, dans l'état courant, pour toutes les configurations d'entrées, l'état futur est connu

## Déterminisme : complétude et orthogonalité

- orthogonalité :  $\forall s_j, s_k \in Succ_i \times Succ_i$ , avec  $j \neq k$ ,  $c_{ij} \cdot c_{ik} = 0$

En d'autres termes, il n'y a pas 2 arcs sortants pour lesquels une combinaison des entrées est vraie

Cela signifie que pour une configuration des entrées données, un état et un seul est l'état futur

Ces deux propriétés seront forcément imposées par l'implantation

# Plan

## 1 Machines avec un nombre fini d'états

- Définitions
- Représentations usuelles en matériel
- Déterminisme

## 2 Implantation matérielle des machines à états

- Encodages
- Structure matérielle

## 3 Specifications

## Encodages

Pourquoi :

Les alphabets d'entrée et de sortie sont des symboles

Le matériel manipule des bits, pas des symboles

⇒ nécessité d'encoder ces symboles en chaînes de bits

Encodage des entrées/sorties

- souvent imposé par le matériel utilisé avec la FSM

Encodage des états

- étant donnés  $n$  états, comment choisir leur représentation binaire ?
- influence majeure sur la complexité de  $\Theta$  et  $\Gamma$

## Encodage des états

Problème :

choisir une injection des états vers une chaîne de bits

- taille  $b$  de la représentation binaire :  $\lceil \log_2 n \rceil \leq b \leq n$
- nombre  $m$  d'encodages *distincts* pour  $b = \lceil \log_2 n \rceil$  :  $m = \frac{(2^b - 1)!}{(2^b - n)!b!}$  †  
 $m$  croît très vite
- toute injection fait l'affaire
- certaines ont des implantations plus simples que les autres

---

†. E. J. McCluskey, and S. H. Unger, "A note on the number of internal variable assignments for sequential switching circuits", IRE Trans. on Electronic Computers, vol. EC-8, pp. 439-440, December 1959.

## Encodage 1 parmi $n$

Ou encore *One-Hot coding* :  $b = n$

Un état est représenté par un unique bit à 1

On note le bit  $i$  de la chaîne de bits  $r_i$

- machine dans l'état  $s_i$  si  $r_i = 1$
- machine dans un état et un seul  $\Rightarrow \forall j \neq i, r_j = 0$
- état initial :  $s_k$  pour lequel  $r_k = 1$
- + encodage unique
  - la permutation des  $r_i$  ne change pas les équations, seulement les noms
- +  $\Theta$  simple à calculer
  - soit  $Pred_i \subset S$  l'ensemble des prédécesseurs de  $s_i$ , alors  $r_i = \sum_{s_h \in Pred_i} r_h c_{hi}^k$
- + distance de Hamming  $\Delta_h = 2$ 
  - $n$  bits au lieu de  $\lceil \log_2 n \rceil$

## Encodage logarithmique

Ou *logarithmic encoding*, donc  $b = \lceil \log_2 n \rceil$

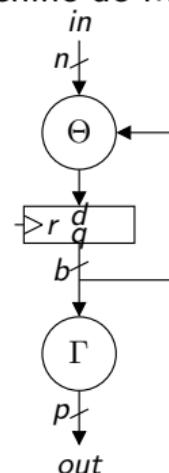
Un état est représenté par un produit de littéraux ( $r_i$  ou  $\bar{r}_i$ )

- quelques exemples :
  - ordre naturel :  $s_0 \rightarrow 0, s_1 \rightarrow 1, \dots, s_{n-1} \rightarrow n - 1$
  - ordre naturel inverse :  $s_0 \rightarrow n - 1, s_1 \rightarrow n - 2, \dots, s_{n-1} \rightarrow 0$
  - tirage aléatoire sans remise
- quel encodage minimise les équations logiques ?
  - il faut essayer les  $m$  pour savoir, ...
  - non soluble en temps polynômial
  - on se contentera d'une approche intuitive dans les TDs
- $\Theta$  et  $\Gamma$  sont des fonctions de potentiellement tous les  $r_i$

## Structure matérielle

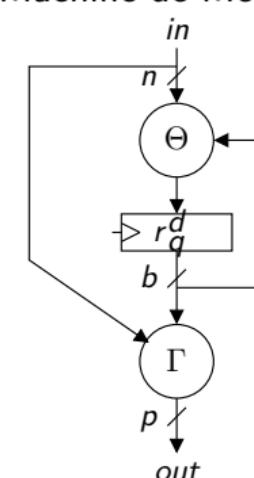
- $in$  : ensemble des signaux encodant les éléments de  $I$
- $r$  : ensemble des registres encodant les éléments de  $S$
- $out$  : ensemble des signaux encodant les éléments de  $O$

Machine de Moore



Les sorties ne dépendent  
que de l'état courant

Machine de Mealy



Les sorties dépendent  
de l'état courant et des entrées

## Synthèse de l'automate

*in* :

- fournie par l'extérieur, uniquement utilisée en entrée de  $\Theta$  et  $\Gamma$

*r* :

- $q_{b-1\dots 0}$ , sortie de la bascule  $b$  bits code l'état courant
- $d_{b-1\dots 0}$ , entrée de la bascule  $b$  bits code l'état futur

*out* :

- calculée en fonction de l'état courant  $q_{b-1\dots 0}$
- *in* intervient dans les machines de Mealy
- l'état futur  $d_{b-1\dots 0}$  JAMAIS !

## Synthèse de l'automate

Choix des encodages : exemple du compteur modulo 4

- entrées : 3 valeurs codées sur 2 =  $\lceil \log_2 3 \rceil$  signaux

$$c = \begin{cases} 1 & \text{compte,} \\ 0 & \text{ne compte pas.} \end{cases} \quad p = \begin{cases} 1 & \text{incrémente,} \\ 0 & \text{décrémente.} \end{cases}$$

choix arbitraire : dicté par l'« expérience »

- sorties : 4 valeurs codées sur 2 =  $\lceil \log_2 4 \rceil$  signaux  $o_1 o_0$

$$\{z, u, d, t\} = \{00, 01, 10, 11\}$$

choix arbitraire : dicté par l'« expérience » car c'est directement l'encodage de la valeur voulue

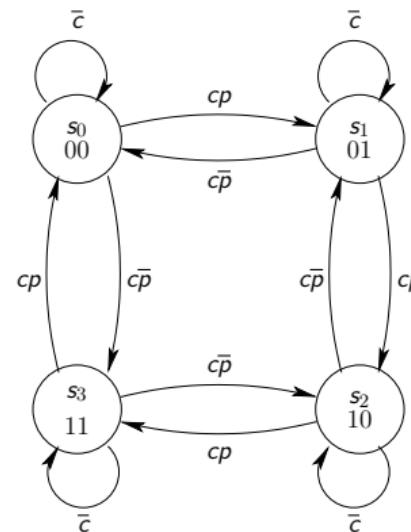
- états : 4 états codées sur 2 =  $\lceil \log_2 4 \rceil$  signaux

$$\{s_0, s_1, s_2, s_3\} = \{00, 01, 10, 11\}$$

choix minimisant la fonction de sortie

## Synthèse de l'automate

Graphe de transitions :



Synthèse *one-hot* au tableau

# Synthèse de l'automate

Table de transition :

état courant	$q_1 q_0$	entrée	$cp$	état futur	$d_1 d_0$	sortie	$o_1 o_0$
$s_0$	00	$r$	0-	$s_0$	00	$z$	00
	00	$i$	11	$s_1$	01	$z$	00
	00	$d$	10	$s_3$	11	$z$	00
$s_1$	01	$r$	0-	$s_1$	01	$u$	01
	01	$i$	11	$s_2$	10	$u$	01
	01	$d$	10	$s_0$	00	$u$	01
$s_2$	10	$r$	0-	$s_2$	10	$d$	10
	10	$i$	11	$s_3$	11	$d$	10
	10	$d$	10	$s_1$	01	$d$	10
$s_3$	11	$r$	0-	$s_3$	11	$t$	11
	11	$i$	11	$s_0$	00	$t$	11
	11	$d$	10	$s_2$	10	$t$	11

## Synthèse de l'automate

Exploitation de la table de transitions :

Θ

calcul des expressions des  $d_i = t_i(q_{b-1}, q_{b-2}, \dots, q_0, in_{p-1}, in_{p-2}, \dots, in_0)$ ,  
pour  $i = 0, \dots, n - 1$

Γ

Machines de Moore, calcul des expressions des  
 $o_i = g_i(q_{b-1}, q_{b-2}, \dots, q_0)$ , pour  $i = 0, \dots, n - 1$

Machines de Mealy, calcul des expressions des

$o_i = g_i(q_{b-1}, q_{b-2}, \dots, q_0, in_{p-1}, in_{p-2}, \dots, in_0)$ , pour  $i = 0, \dots, n - 1$

## Synthèse de l'automate

Mise en œuvre sur l'exemple :

$d_0$	$\bar{q}_1 \bar{q}_0$	$\bar{q}_1 q_0$	$q_1 q_0$	$q_1 \bar{q}_0$
$\bar{c}\bar{p}$		1	1	
$\bar{c}p$		1	1	
$c\bar{p}$	1			1
$c\bar{p}$	1			1

$d_1$	$\bar{q}_1 \bar{q}_0$	$\bar{q}_1 q_0$	$q_1 q_0$	$q_1 \bar{q}_0$
$\bar{c}\bar{p}$			1	1
$\bar{c}p$			1	1
$c\bar{p}$		1		1
$c\bar{p}$	1		1	

# Plan

## 1 Machines avec un nombre fini d'états

- Définitions
- Représentations usuelles en matériel
- Déterminisme

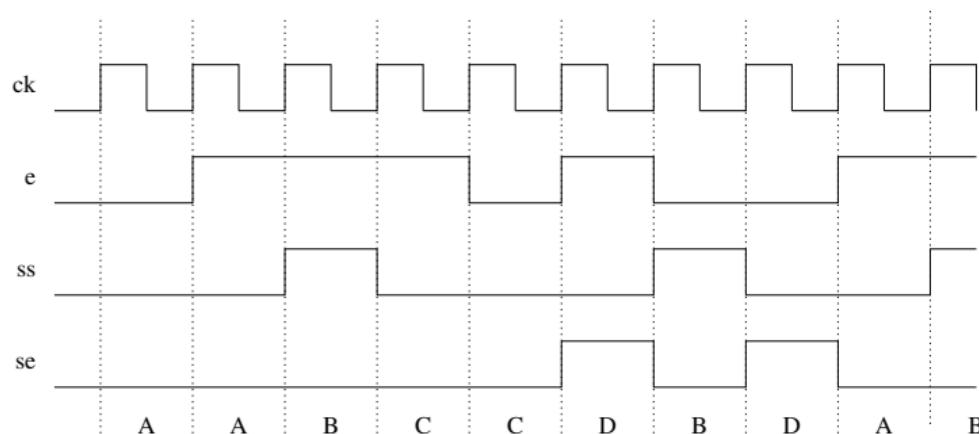
## 2 Implantation matérielle des machines à états

- Encodages
- Structure matérielle

## 3 Specifications

## Grâce à des diagrammes de temps

Extraction de l'automate :



## À partir de spécifications textuelles

- on cherche à détecter le passage à 1 d'un signal e
- on cherche à détecter une certaine séquence d'entrée, genre *digicode*

Réponse au tableau