

1A-SRI

"Systèmes à événements discrets"

Intervenants :

G. CHERIF(Cours) - gcherif@laas.fr

C. Briand (TD) - briand@laas.fr

Volumes :

- 12 h cours
- 12 h TD
- 12h TP (3 manipulations de 4 heures)

Introduction

- ❑ Étude des SED = Un champ de l'Automatique (modélisation, de l'analyse, de la commande et, de la régulation des systèmes dynamiques)
- ❑ Le modèle du système diffère selon la façon de traiter le temps
 - Temps continu
 - Temps discrétisé
 - à événements discrets
- ❑ Caractéristiques principales d'un SED
 - Le système est caractérisé par son état $X \in \{state_0, \dots, state_n\}$

L'état d'un SED est défini par une variable discrète prenant valeur dans un domaine discret fini
 - La notion d'événement
 - le passage d'un état à un autre (transition d'état) se produit à l'occurrence d'événements issus de l'environnement et/ou par des conditions logiques
 - Un événement est instantané (pic de Dirac). Par hypothèse, deux événements ne peuvent se produire simultanément
 - Un comportement (temps implicite)
 - Quel que soit l'état considéré, on connaît toutes les transitions d'état possibles, chaque transition étant associée à l'occurrence d'un événement.

Introduction

❑ Champs applicatifs ?

- Électronique numérique
 - Description du comportement de composants (ex : Processeurs, Cartes d'E/S)
- Informatique
 - Analyse de comportement de processus informatiques (programmation concurrente)
 - Description du fonctionnement d'IHM
 - Représentation de protocole de communication entre systèmes distants (réseaux)
- Sciences des organisations
 - Modélisation de l'enchaînement d'un ensemble d'activités sous contraintes de ressources (processus d'organisation)
Ex : Construction de bâtiments, Développement de logiciels, Logistique...
- Automatique
 - Commande de systèmes complexes
Ex : robotique, ateliers de production automatisés, ...

Introduction

❑ Exemple 1: robot explorateur

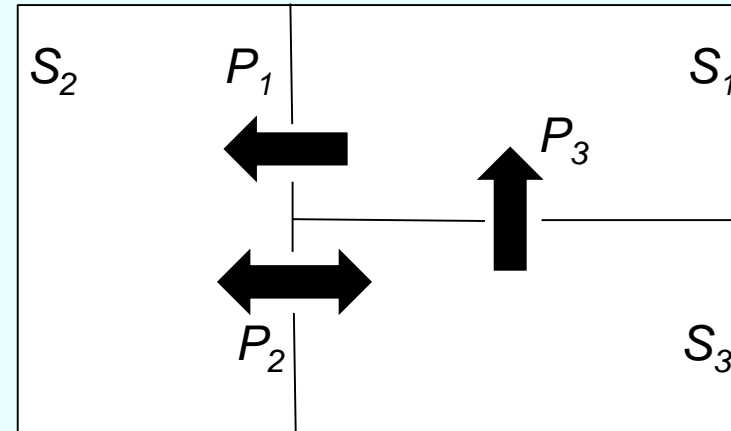
Un robot se déplace d'une manière spontanée à l'intérieur d'un labyrinthe. Les salles S_i communiquent par portes unidirectionnelles P_1 et P_3 et bidirectionnelle P_2 .

On note P_i l'événement

« le robot passe par la porte P_i »

3 situations possibles :

- Le robot dans la salle S_1 .
- Le robot dans la salle S_2 .
- Le robot dans la salle S_3 .



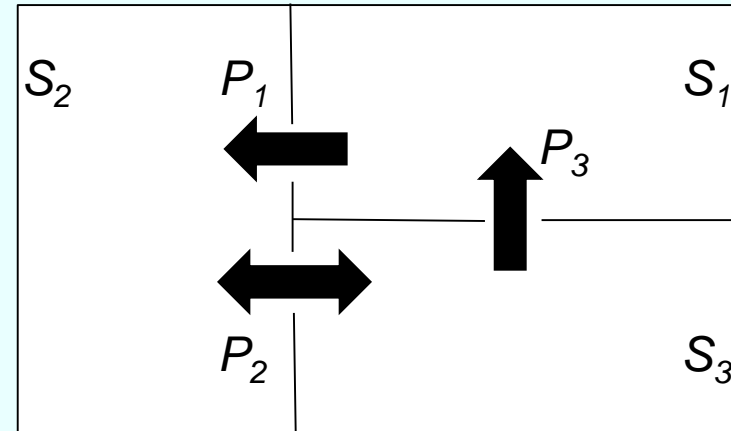
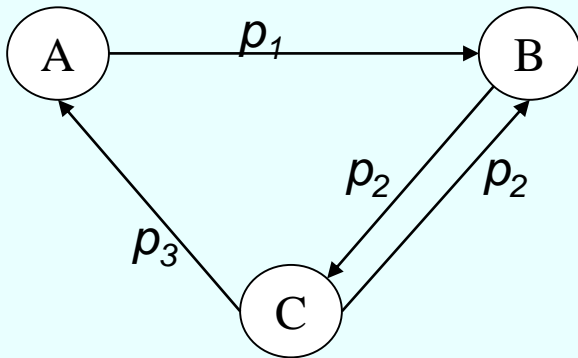
Ce qui définit trois états différents (A , B et C), relatifs aux possibilités d'occupation des salles.

→ L'espace d'états s'écrit $X=\{A, B, C\}$

Introduction

❑ Exemple 1: robot explorateur

Soit E l'ensemble des événements : $E = \{p_1, p_2, p_3\}$



Le passage de l'état « robot en S_1 » (état A) à l'état « robot en S_2 » (état B) nécessitent l'occurrence de l'événement p_1 .

Introduction

➤ Deux besoins

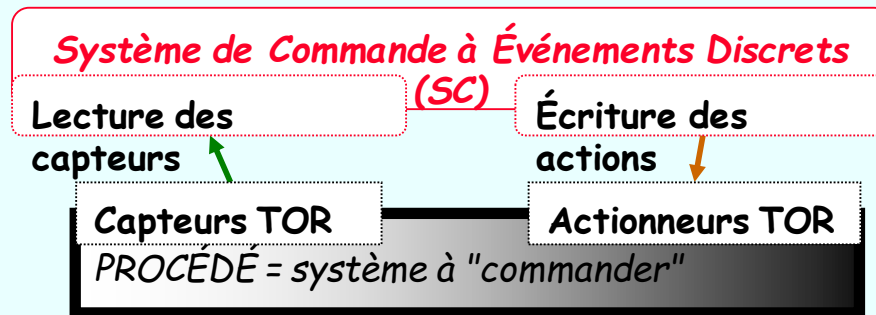
- **Modélisation du comportement d'un système existant**

Objectif : Analyser le comportement d'un système et comprendre ses interactions avec son environnement. Être capable de modéliser / simuler son fonctionnement.

Ex : Modélisation des flux de production dans un atelier de fabrication

- **Modélisation d'un système de commande**

Objectif : Concevoir un système de commande d'un procédé qui satisfasse un ensemble de besoins exprimés dans un cahier des charges



REMARQUE : possibilité de discrétiser un procédé (commande robot, production de fluides, ...)

Classes de SED

❑ Systèmes combinatoires

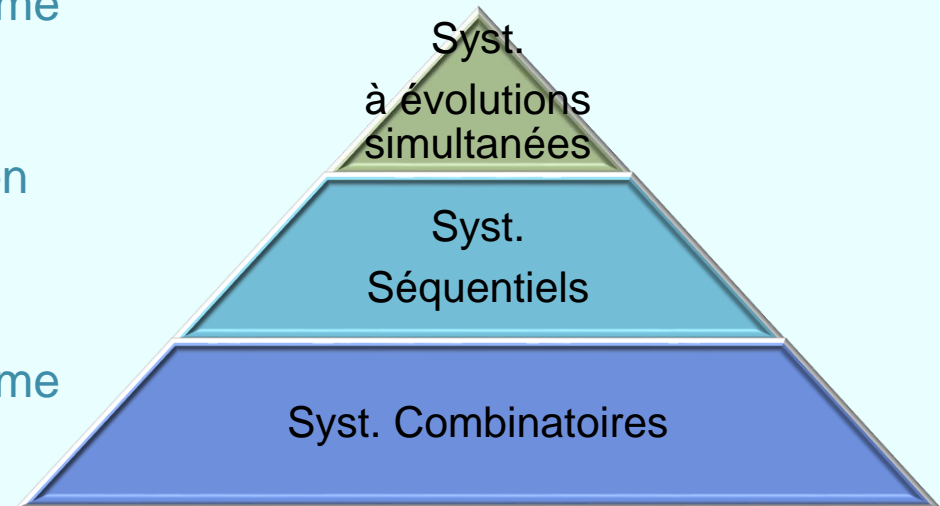
- Les sorties produites par le système de commande sur son environnement dépendent exclusivement des événements en entrée.

❑ Systèmes séquentiels

- Les sorties produites par le système de commande sur son environnement dépendent des événements en entrée ET DE SON ÉTAT INTERNE.

❑ Systèmes à évolutions simultanées

- Les sorties produites par le système de commande sur son environnement dépendent des événements en entrée ET DE SES SOUS-ÉTATS INTERNES.



Objectifs

❑ Objectifs

- Maîtriser le formalisme des SED
- Savoir analyser un système séquentiel existant
- Savoir modéliser une commande satisfaisant un cahier des charges
- Savoir mettre en œuvre une commande en utilisant de multiples technologies

❑ Références

- Machines à états (modélisation, mise en œuvre, ...)
 - Logique combinatoire et séquentielle : méthodes, outils et réalisations / Claude Brie,... - Ellipses, 2002
 - Circuits logiques programmables, A. Nketsa, Informatique Industrielle, Ellipses, 1998.
- VHDL
 - **VHDL** Langage, modélisation, synthèse - 2ème édition - Presses Polytechniques et Universitaires Romandes - 02/1998 - 568 pages ISBN: 2-88074-361-3

"Systèmes à événements discrets"



- Rappels de logique combinatoire et séquentielle



- Les systèmes séquentiels logiques



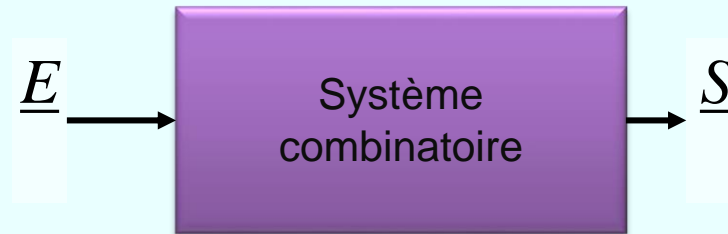
- Mise en œuvre de systèmes séquentiels logiques



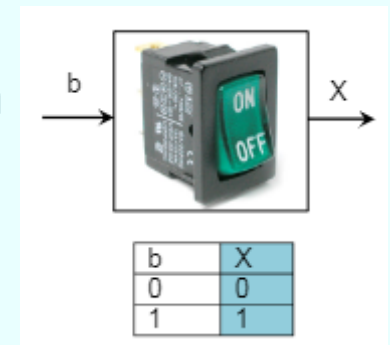
- Systèmes à évolutions simultanées (M1 SMI)

I.1 Rappels de logique Combinatoire

- Un Σ combinatoire est tel que les sorties \underline{S} du Σ sont entièrement déterminées par la connaissance des entrées appliquées \rightarrow aucun effet mémoire, ni de notion d'état



- Une même cause produit toujours le même effet. Un même état des entrées donne toujours le même état des sorties.
- Exemple 2: Système élémentaire de commande d'un dispositif par un bouton à deux positions stables
 - $b=1$ si la position du bouton est sur ON,
 - $b=0$ si la position du bouton est sur OFF



I.1 Rappels de logique Combinatoire

- ❑ Un Σ combinatoire peut être décrit sous diverses formes (types de représentations)
 - Un ensemble d'expressions booléennes (algèbre de BOOLE)
 - Une représentation tabulaire (table de vérité, table de Karnaugh)
 - Un logigramme, un programme ... Mais là, il s'agit de représentation correspondant déjà à la **mise en œuvre** d'une modélisation
 - On peut passer d'une forme de représentation à une autre forme.
- ❑ Prérequis à ce cours: algèbre de Boole et portes logiques de base

I.1 Rappels de logique Combinatoire

□ Axiomes

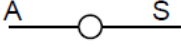

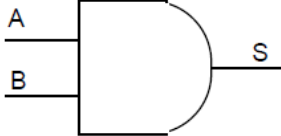

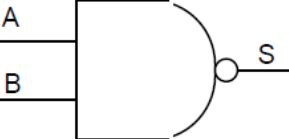

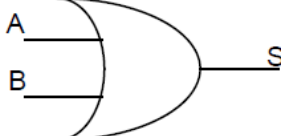
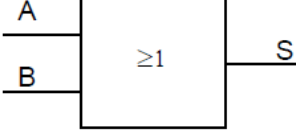
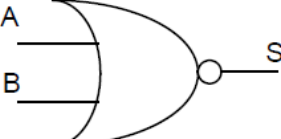
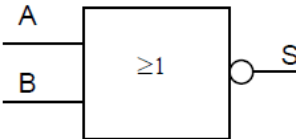
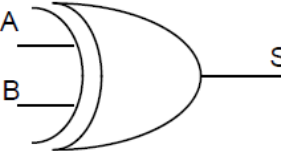
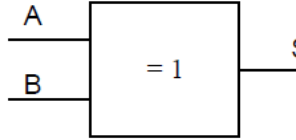
A1	Associativité	$(A + B) + C = A + (B + C)$	$(A . B) . C = A . (B . C)$
A2	Commutativité	$A + B = B + A$	$A . B = B . A$
A3	Élément neutre	$A + 0 = A$	$A . 1 = A$
A4	Distributivité	$A . (B + C) = A . B + A . C$	$A + B . C = (A + B) . (A + C)$
A5	Complémentation	$A + \neg A = 1$	$A . \neg A = 0$

□ Propriétés

P1	Idem potence	$A . A = A$	$A + A = A$
P2	Unicité du complément		
P3	Involution	$\neg \neg A = A$	
P4	Identité	$A + 1 = 1$	$A . 0 = 0$
P5	Absorption	$A + A . B = A$	$A + \neg A . B = A + B$
P6	Théorèmes de Morgan	$\neg (A + B) = \neg A . \neg B$	$\neg (A . B) = \neg A + \neg B$
P7	Consensus	$A . B + \neg A . C + B . C = A . B + \neg A . C$	

I.1 Rappels de logique Combinatoire

Portes logiques

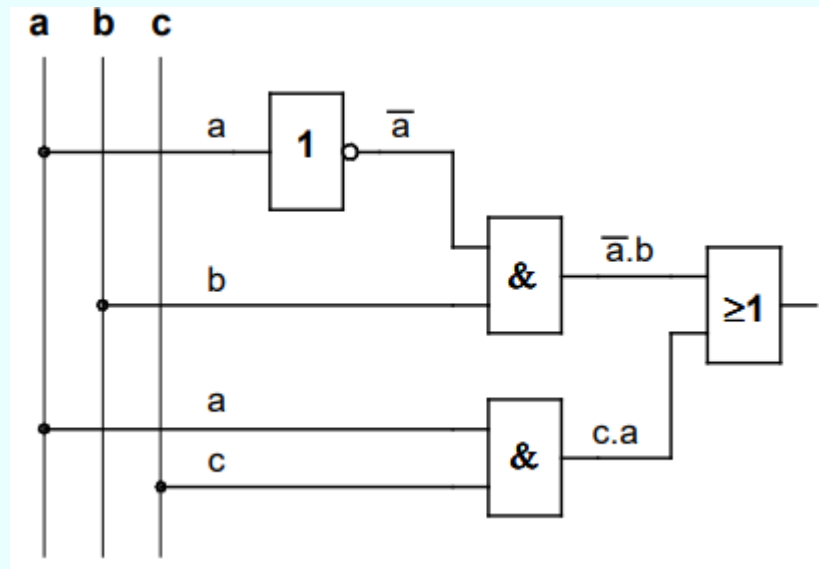
Notations américaine et européenne		Sortie
Inverseur (Circuit 7404)		
		$S = \neg A$
Porte AND (Circuit 7408)		
		$S = A.B$
Porte NAND (Circuit 7400)		
		$S = \neg (A.B)$
Porte OR (Circuit 7432)		
		$S = A + B$
Porte NOR (Circuit 7402)		
		$S = \neg (A + B)$
Porte XOR (Circuit 7486)		
		$S = A.\neg B + \neg A.B$ $= A \oplus B$

I.1 Rappels de logique Combinatoire

❑ Exemple 3: logigramme

- Construire le logigramme de l'expression logique de l'exemple 1:

$$S = \neg a . b + a . c$$



I.1 Rappels de logique Combinatoire

❑ I.2. Représentation tabulaire des Systèmes combinatoires : les tables de Vérités (TV)

➤ Notations

$E = [E_1, E_2, \dots, E_N]$ = entrées, $S = [S_1, S_2, \dots, S_M]$ = sorties

➤ Forme

- 2^N lignes (une par combinaison d'entrées) et $(N+M)$ colonnes

BINAIRE
NATUREL

<u>E : Entrées</u>				<u>S : Sorties</u>	
E_N	...	E_2	E_1	S_1	...
0	...	0	0	0 ou 1	
0	...	0	1	0 ou 1	
0	...	1	0	0 ou 1	
0	...	1	1	0 ou 1	
...	
1	1	1	1	0 ou 1	

A	B	C	S
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

- Exemple : pour $E = [A, B, C]$ et $S = [S]$ → voir table ci-dessus

➤ TV → déduction des expressions booléennes des sorties,

- Si on considère les points à Vrai pour la sortie → forme $\Sigma \Pi$ (somme de monômes)
- Si on considère les points à faux pour la sortie → forme $\Pi \Sigma$ (produit de monal)

I.1 Rappels de logique Combinatoire

❑ INCONVÉNIENTS des TV :

➤ TAILLE de la représentation : 2^N lignes

- Ce nb devient très vite prohibitif pour un système réaliste
- Les expressions logiques obtenues avec une TV sont non simplifiées → il faut encore appliquer les propriétés de l'algèbre de Boole

❑ Tables de Karnaugh (TK)

➤ TK = TV simplement "disposée" autrement

➤ Disposition

- Composantes E_i de \underline{E} scindées en 2 sous-ensembles, codés en binaire RÉFLÉCHI (code de Gray)
 - E_1, E_2, \dots, E_j sont placées en entête des lignes du tableau
 - $E_{j+1}, E_{j+2}, \dots, E_N$ sont placées en entête des colonnes du tableau
- La valeur $f(E_1 E_2 \dots E_N)$ associée par la fonction logique f à une combinaison particulière de \underline{E} est placée dans la case à l'intersection de la ligne et de la colonne relatives à cette combinaison particulière
 - une case de la table de KARNAUGH \Leftrightarrow une ligne de la table de vérité

I.1 Rappels de logique Combinatoire

➤ Exemple à 4 variables

Disposées
BINAIRE
RÉFLÉCHI

$E_3 \ E_4$	00	01	11	10
$E_1 \ E_2$				
00				
01			F (0,1,1,1)	
11				
10				

➤ Intérêt

- Dans une expression logique, 1 monôme formé sur N-p variables de E identifie 2^p points VRAIS.
- Dans une TK, ces 2^p points vrais sont situés dans des cases adjacentes
- ➔ Inversement, un regroupement de cases à « 1 » adjacentes identifie un monôme
- Exemple: N = 3

$p=0 : m_1 = E_1 \cdot \neg E_2 \cdot E_3$ vaut 1 si $(E_1, E_2, E_3) = (1, 0, 1) \rightarrow$ couvre 1 point : 101

$p=1 : m_2 = E_1 \cdot \neg E_2$ vaut 1 si $(E_1, E_2, E_3) = (1, 0, *) \rightarrow$ couvre 2 points : 101, 100

$p=2 : m_3 = E_1$ vaut 1 si $(E_1, E_2, E_3) = (1, *, *) \rightarrow$ couvre 4 points : 101, 100, 110, 111

I.1 Rappels de logique Combinatoire

- Méthodologie pour déterminer une expression logique simplifiée à partir d'une TK
 - 1) Choisir dans la table un point vrai (resp. faux) et regrouper alors autour de ce point en recherchant le plus gros regroupement
 - ➔ Ajouter le monôme correspondant à l'expression logique
 - 2) Choisir un point non encore couvert et recommencer.
 - 3) Continuer jusqu'à avoir couvert TOUS les points vrais.

- Postulats
 - Tous les points vrais (resp. faux) doivent être couverts par au moins un regroupement.
 - Un point peut être couvert par plusieurs regroupements.
 - Lorsqu'un regroupement est le plus grand possible, le monôme obtenu est premier, puisqu'il est impossible de réduire davantage son nombre de variables

On obtient **une base 1ère complète** en faisant TOUS les PLUS GRANDS REGROUPEMENTS
 - Si chaque regroupement a au moins une case qui lui est propre, alors on a obtenu une **base première complète irredondante**.

I.1 Rappels de logique Combinatoire

➤ Exemple 4: Tables de Karnaugh

$E_1 E_2 \backslash E_3 E_4$	00	01	11	10
00	0	1	1	0
01	0	1	0	0
11	1	1	0	0
10	0	1	1	0

Groupement 1: $E_4 \cdot \neg E_2$

Groupement 2: $\neg E_3 \cdot E_1 \cdot E_2$

Groupement 3: $\neg E_3 \cdot E_4$

L'expression logique minimale de la sortie est :

$$S = E_4 \cdot \neg E_2 + \neg E_3 \cdot E_4 + \neg E_3 \cdot E_1 \cdot E_2$$

I.1 Rappels de logique Combinatoire

□ Exemple 6: logigramme

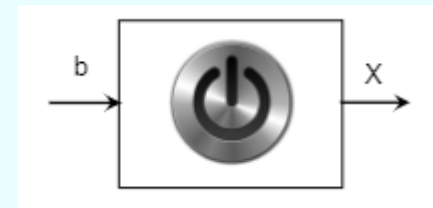
- Construire le logigramme de l'expression logique de l'exemple 1:

$$S = E_4 \cdot \neg E_2 + \neg E_3 \cdot E_4 + \neg E_3 \cdot E_1 \cdot E_2$$

1.2. Rappels de logique séquentielle

- ❑ Un système séquentiel (ou à événements discrets) est un système logique qui n'est pas combinatoire. L'état des sorties dépend de celui des entrées, mais aussi de l'état du système lui-même
- ❑ Exemple 7: Système de commande d'un dispositif par un bouton

- Cette position stable est codée par $b=0$
- si on appuie sur le bouton $b=1$, et $b=0$ sinon.



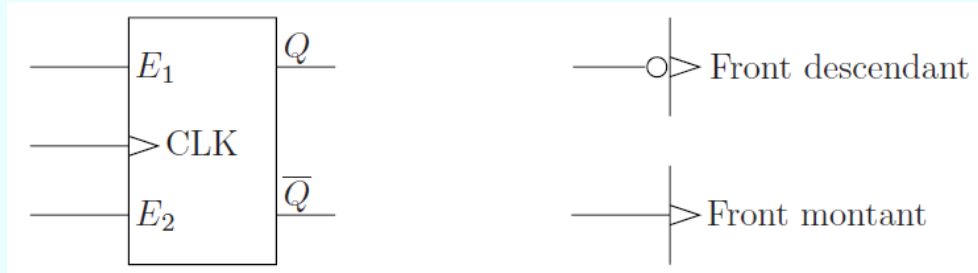
- ➔ Un appui sur le bouton inverse l'état de la variable X de sortie.
- ➔ La conséquence de l'appui sur le bouton ($b=1$) dépend de l'état du système.

I.2. Rappels de logique séquentielle

- ❑ Objectif : Rappeler les fonctions séquentielles de bases et les circuits de réalisation associés
 - Mémorisation d'un bit → bascule
 - Mémorisation de plusieurs bits → registres
 - Comptage d'occurrence d'événements → compteurs

- ❑ 1) Circuits séquentiels élémentaires : les bascules
 - Dispositifs séquentiels permettant de mémoriser un seul bit Q (appeler sortie ou état de la bascule)
 - Bascule synchrone : l'état Q de la bascule ne peut changer qu'à l'occurrence d'un événement sur un signal (front montant ou descendant) → entrée événementielle clk
 - Bascule asynchrone : l'état Q de la bascule change en fonction des niveaux logiques présents sur ses entrées → Pas d'entrée événementielle
 - Dans tous les cas, présence de 2 entrées asynchrones *Set* et *Clear* permettant de forcer la sortie à 1 ou à 0 respectivement

I.2. Rappels de logique séquentielle



➤ Cas de base à envisager sont :

- 1) Maintient de la sortie Q
- 2) Inverser la valeur de Q : fonction "complément" ou "inverseur"
- 3) Forcer la sortie à 1 : mise à un (MU)
- 4) Forcer la sortie à 0 : mise à Zéro (MZ)

➤ Bascule asynchrone (pas de CLK)

- La bascule RS

➤ Bascules synchrones

- La bascule JK
- La bascule D
- La bascule T

I.2. Rappels de logique séquentielle

Bascule RS

Symbolique :

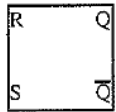


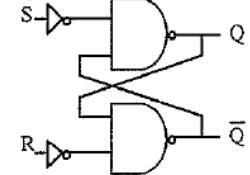
Table de vérité :

R	S	Q_{n+1}
0	0	Q_n
0	1	1
1	0	0
1	1	Interdit

Table d'évolution :

Q_n	Q_{n+1}	R	S
0	0	-	0
0	1	0	1
1	0	1	0
1	1	0	-

Schéma équivalent :



Bascule JK

Symbolique :

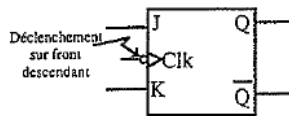


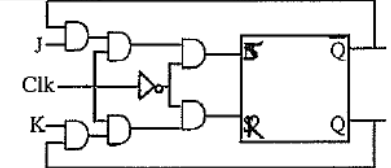
Table de vérité :

J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	$\neg Q_n$

Table d'évolution :

Q_n	Q_{n+1}	J	K
0	0	0	-
0	1	1	-
1	0	-	1
1	1	-	0

Schéma équivalent :



Bascule D

Symbolique :

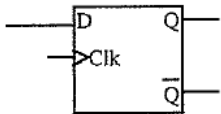


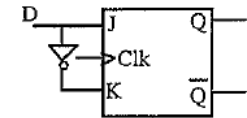
Table de vérité :

D	Q_{n+1}
0	0
1	1

Table d'évolution :

Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

Schéma équivalent :



Bascule T

Symbolique :

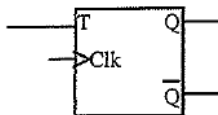


Table de vérité :

T	Q_{n+1}
0	Q_n
1	$\neg Q_n$

Table d'évolution :

Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

Schéma équivalent :



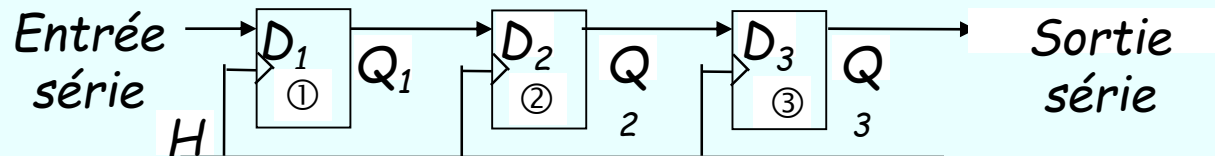
I.2. Rappels de logique séquentielle

❑ 2) Les registres

- Dispositifs séquentiels permettant de mémoriser un "mot" de n bits
 - La mémorisation a lieu à l'occurrence d'un événement (clk)
➔ **Dispositif synchrone**
- Constituant de base
 - 1 bascule D par bit à mémoriser : n bits ➔ n bascules
- 2 principes d'écriture/lecture dans un registre :
 - Principe SÉRIE : à chaque événement, un seul bit est écrit (ou lu) dans le registre, mais par décalage (SHIFT) de une position dans le registre de tous les bits
 - Principe PARALLÈLE : à chaque événement, tous les bits sont écrits (ou lus) en même temps (en parallèle) dans le registre.
- D'où 4 types de registres :
 - Entrée Série – sortie Série (Série-Série)
 - Entrée Série – sortie Parallèle (série – parallèle)
 - Entrée Parallèle – sortie Série (parallèle -Série)
 - Entrée parallèle– sortie Parallèle (parallèle– parallèle)

I.2. Rappels de logique séquentielle

➤ Exemple registre série / série



- à chaque front montant sur H, l'information D_i en entrée de la bascule n° i est transférée sur sa sortie Q_i . Comme cette sortie Q_i est connectée sur D_{i+1} , au prochain front montant sur H, cette même information se retrouve dans la bascule i+1, etc.

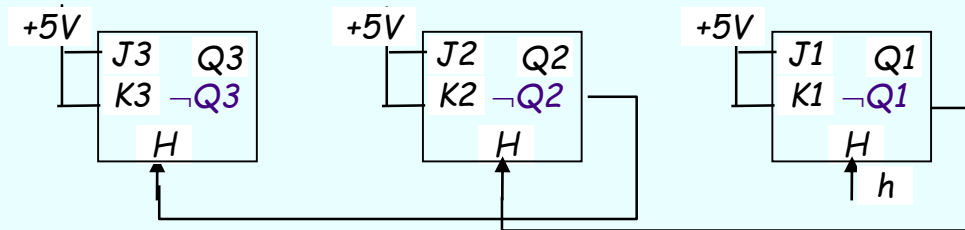
❑ 3) Les compteurs

- Dispositifs séquentiels permettant de compter des occurrences d'événements, occurrences qui provoquent soit l'incrémentation, soit la décrémentation du compteur.
- Particularité
 - il a autant d'états que de valeurs de sorties possibles.
- Constituant de base
 - Bascules : pour un compteur de **X valeurs**, il faut **n bascules** tel que $2^n \geq X$

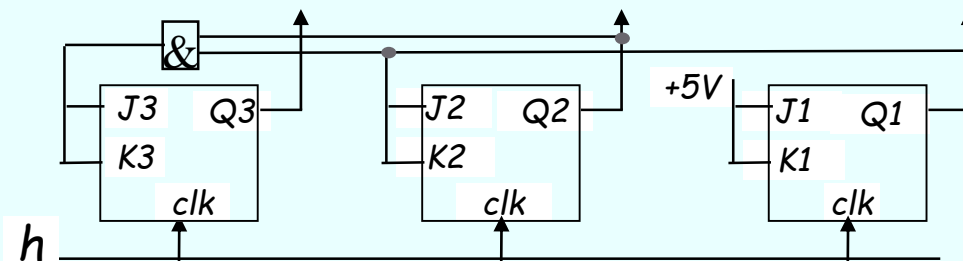
I.2. Rappels de logique séquentielle

❑ 2 types de réalisation : compteur modulo 8

- Asynchrone : le *clk* de la bascule *i* est relié à la sortie de la bascule Q_{i-1} complémentée et les bascules sont montées en diviseur de fréquence



- Synchrone : toutes les bascules partagent la même entrée *clk* d'horloge

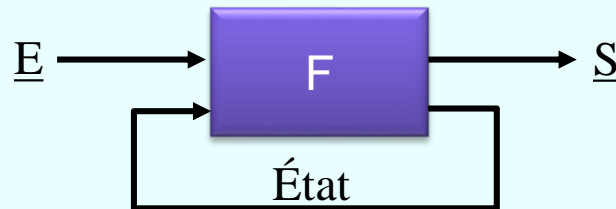


"Systèmes à événements discrets"

- I • Rappels de logique combinatoire et séquentielle
- II • Les systèmes séquentiels logiques
- III • Mise en œuvre de systèmes séquentiels logiques
- IV • Systèmes à évolutions simultanées (sem2)

II.1. Vue informelle des systèmes séquentiels logiques

- ❑ Un système séquentiel est un système dont le comportement est fonction des entrées logiques appliquées ET de son état.
- ❑ Il peut être vu comme un système combinatoire rebouclé sur lui même (dans ce cas certaines sorties correspondent au codage de l'état)

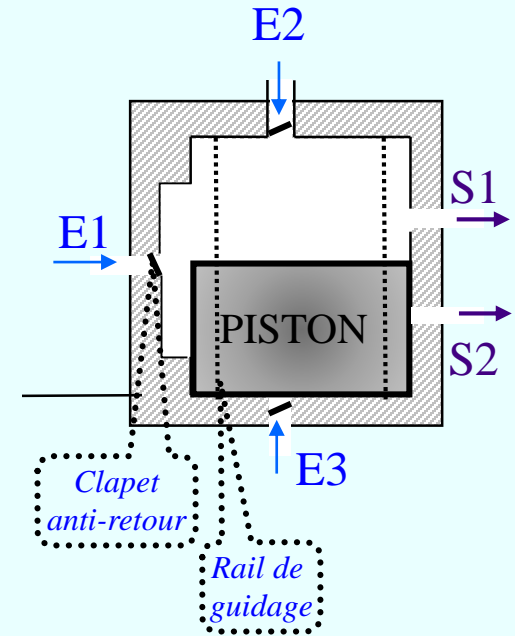


- ❑ L'état du système est susceptible de changer suite
 - à une modification d'une de ses entrées
 - à une modification de l'état lui-même (cascade de changement)
 - lors de ce changement les sorties ne sont pas nécessairement modifiées.
- ❑ Une description doit être faite de tous les états accessibles depuis un état donné
 - But : Connaître le comportement du système et donc prédire l'état atteint suite à une modification donnée en entrée.

II.1. Vue informelle des systèmes séquentiels logiques

❑ Exemple : cas d'un système pneumatique

- entrées E1, E2 et E3
- sorties S1 et S2
- Le piston (sur rail) se déplace sous la pression de l'air comprimé appliqué en E2 et E3.
- E1 ne cause pas de déplacement vertical.
- Les entrées sont munies de clapets anti-retour.
- On ne peut connaître l'état des sorties uniquement à partir de la connaissance des entrées ...



II.2. Définition formelle d'un système séquentiel logique

- ❑ Définition : un Système Séquentiel logique est un quintuplet $\langle e, s, Q, \delta, \omega \rangle$
- $e = \{e_1, e_2, \dots, e_k\}$ est l'alphabet d'entrée. Chaque symbole correspond :
 - ① Soit à une expression logique formée sur les composantes de E
Ex : $e_i = \neg B. \neg C$
 - ② soit à un événement (front \uparrow ou \downarrow) sur une composante donnée de E , associé éventuellement à une expression logique Ex. : $e_i = C \downarrow. \neg(B.A)$
 - $s = \{s_1, s_2, \dots, s_q\}$ est l'alphabet de sortie. Chaque symbole correspond :
 - ① soit une valeur donnée des composantes de S , et s'exprime sous forme de vecteur :
Ex. : $s_i = [0, 0, 1]$
 - ② soit un événement (front \uparrow ou \downarrow) sur une composante donné de S et s'exprime : Ex. : $s_i = X$
 - $Q = \{Q_1, Q_2, \dots, Q_r\}$ est l'ensemble des états du système
 - δ est la fonction de transition $\delta : e \times Q \rightarrow Q$
Détermine la prochaine valeur de Q en réponse à la situation définie par la paire (e_i, Q_j)
 - ω est la fonction de sortie $\omega : e \times Q \rightarrow s$
Détermine la valeur à générer pour les sorties en fonction de la paire (e_i, Q_j)

II.2. Définition formelle d'un système séquentiel logique

❑ Retour au système pneumatique

⇒ vecteur des entrées : $E = [E_1, E_2, E_3]$

⇒ vecteur des sorties : $S = [S_1, S_2]$

⇒ alphabet d'entrée : $e = \{ /E_1./E_2./E_3, /E_1./E_2.E_3, \dots, E_1.E_2.E_3 \}$,
soient les 8 combinaisons possibles

⇒ alphabet de sortie $s = \{ (/S_1, /S_2), (/S_1, S_2), (S_1, /S_2) \}$

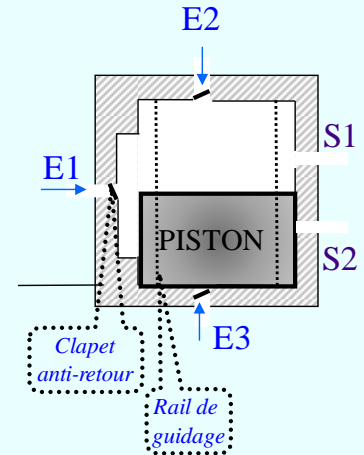
⇒ vecteur d'état $Q = \{Q_{Haut}, Q_{Bas}\}$ selon la position du piston

⇒ $\delta(E_1, E_2, E_3, Q_j) = ?$: → 16 cas

- $\delta(0, 0, 0, Q_{Haut}) = Q_{Haut}$ → pas de changement d'état
- $\delta(0, 0, 0, Q_{Bas}) = Q_{Bas}$ → pas de changement d'état
- $\delta(0, 0, 1, Q_{Haut}) = Q_{Haut}$ → pas de changement d'état
- $\delta(0, 0, 1, Q_{Bas}) = Q_{haut}$ → changement d'état
- ...

⇒ $\omega(E_1, E_2, E_3, Q_j) = (S_1, S_2) ?$: → 16 cas

- $\omega(0, 0, 0, Q_{Haut}) = (0, 0)$
- $\omega(0, 0, 0, Q_{Bas}) = (0, 0)$
- $\omega(0, 0, 1, Q_{Haut}) = (0, 1)$
- $\omega(0, 0, 1, Q_{bas}) = (0, 0)$... puis, dans le temps (0,1) ...
en effet, l'air comprimé entrant dans E3 ne sortira par S2
qu'après que le piston soit suffisamment monté ...on verra plus
tard comment ce type de situation se représente (est pris en
compte).
- ...



II.2. Définition formelle d'un système séquentiel logique

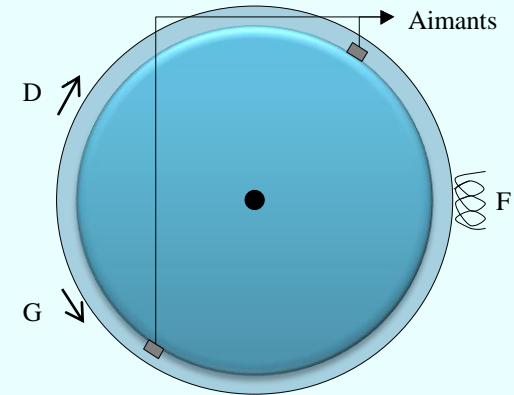
❑ Autre exemple

➤ Malaxeur

- Tambour muni d'aimants diamétralement opposés
- ➔ Une impulsion dans le bobinage F

➤ Cahier des charges

- Le tambour doit faire 1/2 tour à droite, ...
- ... puis 1/2 tour à gauche



➤ Représentation

- ⇒ Le système n'évolue qu'à l'occurrence d'un événement → on ne considère donc pas les cas où il ne se produit pas d'événement (le système reste dans son état courant)
- ⇒ "Vecteur" des entrées : $F\uparrow$ (une seule entrée)
- ⇒ vecteur des sorties : $S = [G, D]$
- ⇒ alphabet des SYMBOLES d'entrée : $e = \{F\uparrow\}$
- ⇒ alphabet des SYMBOLES de sortie : $s = \{ (/G, D), (G, /D) \}$
- ⇒ vecteur d'état $Q = \{Q_G, Q_D\}$
- ⇒ $\delta(e_j, Q_i) = ?$: $\delta(F\uparrow, Q_G) = Q_D$ et $\delta(F\uparrow, Q_D) = Q_G$
- ⇒ $\omega(*, Q_i) = ?$: $\omega(*, Q_D) = (0, 1)$ et $\omega(*, Q_G) = (1, 0)$

II.2. Définition formelle d'un système séquentiel logique

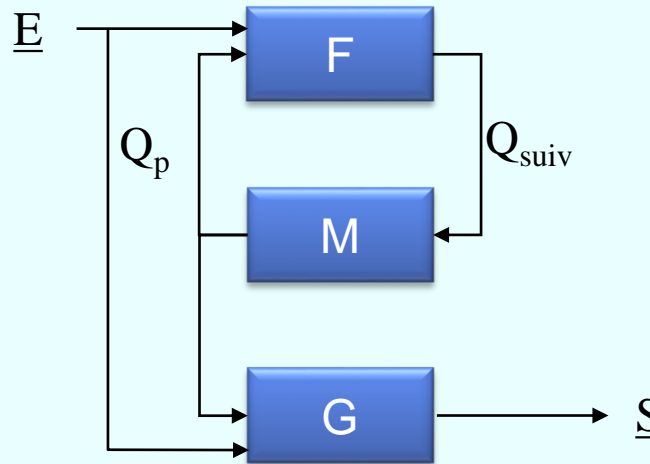
❑ Classification

Faite en considérant les alphabets d'entrée e et de sortie s

- Cas où e est un ensemble de combinaisons logiques de E
 - On dit que le système fonctionne en mode **ASYNCHRONE**, aussi appelé mode **FONDAMENTAL**
 - Les entrées sont dites de type "NIVEAU" (pas d'entrées dites "impulsionnelles" ou événementielles) : la valeur de ces entrées est fournie par des capteurs de type TOUT ou RIEN (TOR)
- Cas où e est une "liste" d'ÉVÉNEMENTS
 - On dit que le système fonctionne en mode **SYNCHRONE**, aussi appelé mode **PULSÉ**
 - Une au moins des entrées de \underline{E} est IMPULSIONNELLE (ou ÉVÉNEMENTIELLE)
 - Machine de MOORE si s = ensemble de combinaisons logiques
 - Machine de MEALY si s = liste d'événements
- Remarque : un système séquentiel fonctionnant en mode SYNCHRONE est une ABSTRACTION d'un système séquentiel en mode ASYNCHRONE

II.3. Représentation symbolique par schéma bloc

□ Le schéma bloc FMG



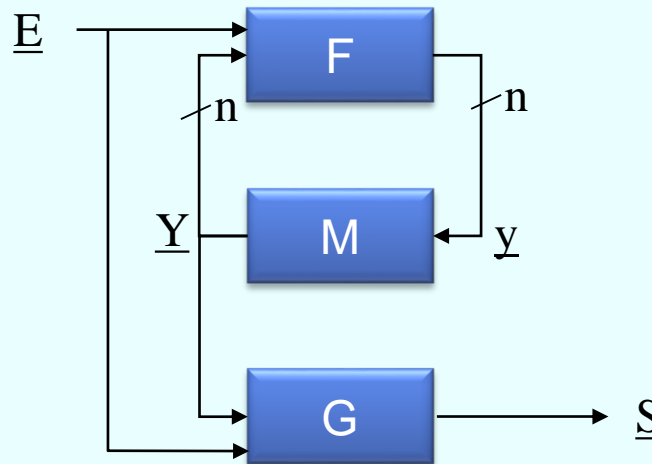
- ⇒ Notations des **états présent et suivant** : Q_p et Q_s , ou Q_n et Q_{n+1}
- ⇒ **Blocs F et G** : purement combinatoires, et supposés réagir instantanément (temps de propagation des signaux nul → Vue idéale : dans la réalité ce n'est pas le cas.
- ⇒ **Bloc M** : Le caractère séquentiel est « symbolisé » par le bloc **mémoire** M, dont la nature détermine le mode de fonctionnement du système : FONDAMENTAL ou PULSÉ (Asynchrone ou Synchrone respectivement)

II.3. Représentation symbolique par schéma bloc

❑ Mise en œuvre électrique de systèmes séquentiels

À chaque composante des vecteurs \underline{E} et \underline{S} est associé un signal électrique.

Chaque état q_i de $Q = \{q_1, \dots, q_r\}$ est codé par un **vecteur de n bits noté \underline{Y}** tel que $2^n \geq r$. Les **composantes Y_i de \underline{Y}** sont appelée **variables internes** (VI) et sont chacune mise en œuvre par un signal électrique



II.3. Représentation symbolique par schéma bloc

❑ Nature du bloc M et mode de fonctionnement d'un système séquentiel

➤ Combien de temps dure la mémorisation de l'état présent ?

➤ Mode fondamental (ou asynchrone)

M = retard (temps de propagation des signaux dans le bloc F)

Le retard n'est pas identique pour chaque VI \Rightarrow 2 VI ne peuvent pas commuter simultanément ! \Rightarrow **phénomène de course**

➤ Mode pulsé (ou synchrone)

M = échantillonneur-bloqueur (registre de n bits)

La valeur de y est échantillonnée à l'occurrence d'un événement (front) sur un signal construit à partir des composantes du vecteur d'entrée \underline{E} (l'alphabet d'entrée e comporte des événements)

\Rightarrow toutes les VI commutent simultanément \Rightarrow **pas de course**

- Machine de Moore : l'alphabet de sortie s ne contient pas d'événement
- Machine de Mealy : l'alphabet de sortie s contient des événements

II.4. Trois représentations formelles d'un système séquentiel logique

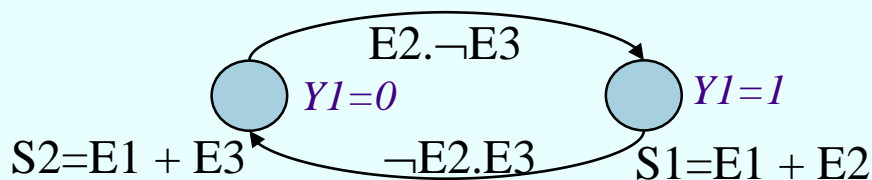
□ La représentation algébrique

- Il s'agit de déterminer les équations logiques régissant le fonctionnement des blocs F et G

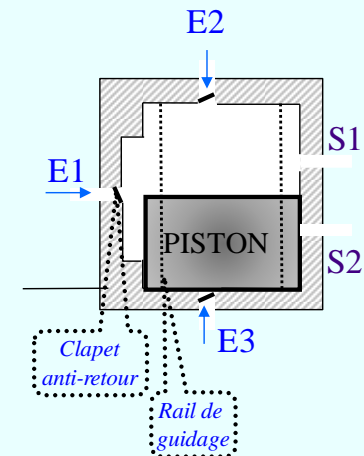
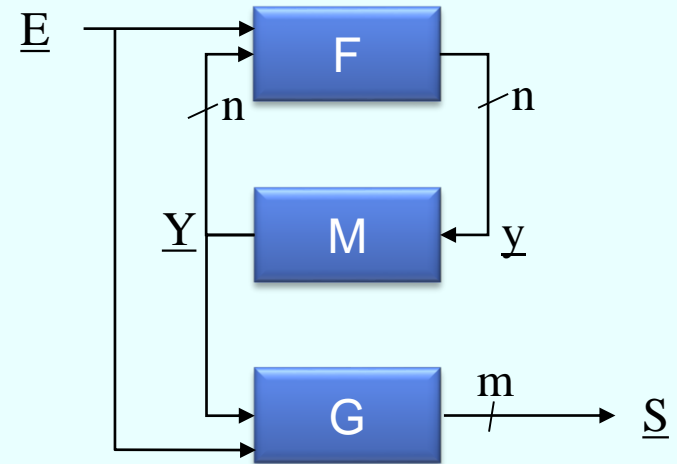
$$y_i = F(\underline{Y}, \underline{E}) \text{ pour } i = 1 \dots n$$

$$s_j = G(\underline{Y}, \underline{E}) \text{ pour } j = 1 \dots m$$

- Exemple du système pneumatique



$$\Leftrightarrow \begin{cases} y1 = E2 \cdot \neg E3 + Y1 \cdot (E2 + \neg E3) \\ S1 = Y1 \cdot (E1 + E2) \\ S2 = \neg Y1 \cdot (E1 + E3) \end{cases}$$



II.4. Trois représentations formelles d'un système séquentiel logique

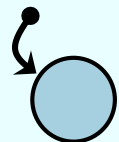
□ La représentation graphique (machine à états / graphe d'états)

➤ Système séquentiel → **graphe** tel que :

- Sommet = un état q_i (représenté par un cercle)
- Arc orienté et étiqueté entre 2 sommets = changement d'état
- Une étiquette de transition
 - un événement si le mode de fonctionnement est pulsé
 - une condition logique si le mode de fonctionnement est fondamental
- Les sorties sont associées :
 - Aux états (sortie niveau) : machine de Moore
 - Aux transitions (sortie événementielle) : machine de Mealy

➤ Conventions

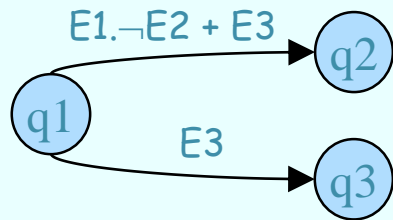
- À tout instant, un système ne peut être que dans un seul état à la fois (dit état "actif")
- Un changement d'état ne peut avoir lieu que pour l'état actif, lorsque la condition portée sur un des arcs sortant de cet état est vraie (mode fondamental) où lorsqu'un des événements étiquetant les arcs sortants se produit (mode pulsé).
- L'état initialement actif est indiqué par un arc ne venant d'aucun état



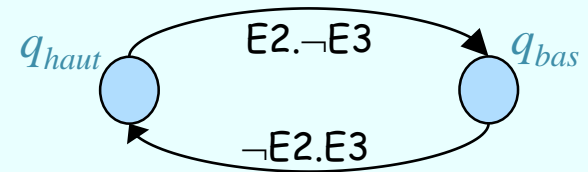
II.4. Trois représentations formelles d'un système séquentiel logique

➤ Étiquettes de transition

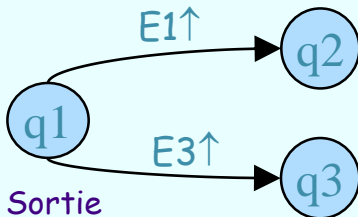
- Mode fondamental (asynchrone)



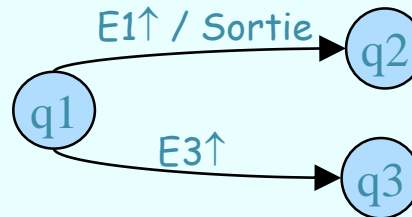
Ex. Syst. Pneumatique



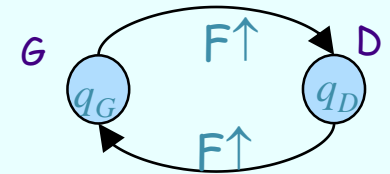
- Mode pulsé



Machine de MOORE
(sorties niveaux)



Machine de MEALY
(sorties impulsionnelles)



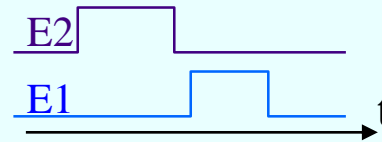
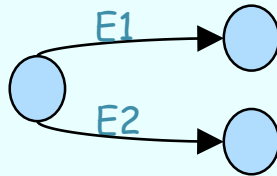
Application à
l'exemple du Malaxeur

- ATTENTION : les changements d'états doivent être **déterministes**
 - Les étiquettes de transition doivent être **mutuellement exclusives**
 - ex : ici, dans l'exemple du monde fondamental, si E3 est vraie, le syst. peut évoluer soit vers l'état 2, soit vers l'état 3 => comportement indéterministe

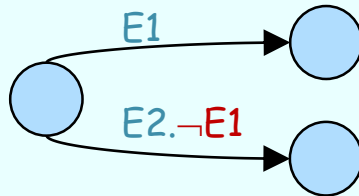
II.4. Trois représentations formelles d'un système séquentiel logique

➤ Comment assurer l'exclusion mutuelle de conditions logiques

- Temporellement : il est parfois physiquement impossible que 2 entrées soient simultanément vraies (ex : capteur d'étage pour une cabine d'ascenseur)



- Définir une priorité : si l'exclusion physique n'est pas assurée, il faut définir logiquement une priorité

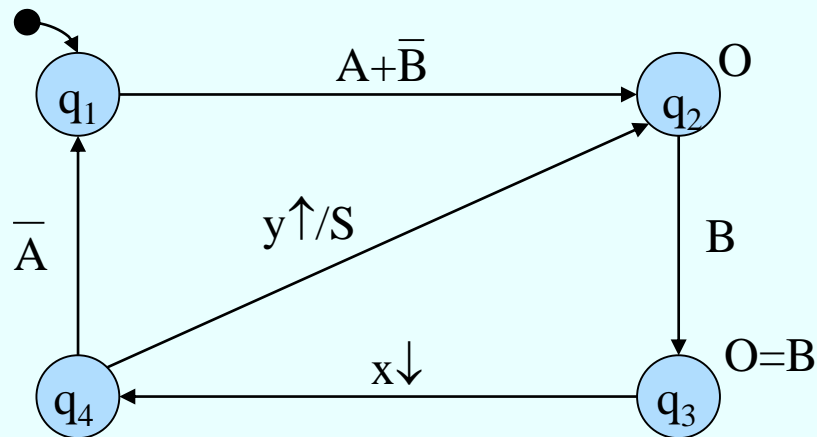


E1 prioritaire sur E2

II.4. Trois représentations formelles d'un système séquentiel logique

➤ Représentation des sorties

- Sorties niveaux ➔ associées aux états
 - On les représente à côté du sommet du graphe
 - Par convention, la sortie n'est indiquée à côté d'un état que si elle est **active** sur cet état
 - La sortie peut être conditionnelle : dans ce cas on indique à côté de l'état l'équation (dépendant uniquement des entrées) qui décrit le comportement de la sortie lorsque l'état est actif
- Sorties événementielles ➔ associées aux transitions
 - On les représente en dessous des étiquettes de transition
- Exemple (pas beau!)



II.4. Trois représentations formelles d'un système séquentiel logique

□ La représentation tabulaire

➤ Représentation tabulaire pour le mode fondamental

⇒ On utilise une table de Karnaugh avec :

- en ligne : les combinaisons du vecteur d'état
- en colonne : les combinaisons du vecteur d'entrée

États
PRÉSENTS
Décrits par
leur "codes"
placés en
binaire
réfléchi

$\begin{matrix} \text{Y} \\ \text{E} \end{matrix}$	\underline{E}_1	...	\underline{E}_i	...	\underline{E}_k
0...0					
...					
0...1					
...					
1...					

État
suivant

sorties

Table des états CODÉS

$\begin{matrix} \text{Q}_p \\ \text{E} \end{matrix}$	\underline{E}_1	...	\underline{E}_i	...	\underline{E}_k
\underline{Q}_1					
...					
\underline{Q}_j			$\begin{matrix} q \\ \text{sortie} \end{matrix}$		
...					
\underline{Q}_m					

Table des états NOMMÉS

⇒ 1 LIGNE de la table = 1 ÉTAT INTERNE de la Machine à états fini

⇒ 1 CASE de la table = 1 ÉTAT TOTAL de la Machine à états fini → il y a 2 cas :

- Si $\delta(\underline{E}_i, \underline{Q}_j) = \underline{Q}_j$, l'état total est STABLE → On entoure la case par un cercle
- Si $\delta(\underline{E}_i, \underline{Q}_j) \neq \underline{Q}_j$, l'état total est TRANSITOIRE



II.4. Trois représentations formelles d'un système séquentiel logique

⇒ Cheminement dans une table

- ① Pour un état interne (une ligne de la table), un **changement du vecteur d'entrée** se traduit dans la table par un **changement de colonne**, sur la même ligne
 - ② La case à l'intersection de la ligne considérée et de la nouvelle colonne indique l'état suivant vers lequel évoluera le système.
 - ③ Au bout d'un temps Δt , cet état suivant devient le nouvel état présent → dans la table, cela se traduit par un **changement de ligne**
 - ④ Pour trouver l'état stable finalement, il faut répéter ce principe de "cheminement" jusqu'à trouver, dans la colonne considérée, un **état stable** (cerclé dans la table).
- On dit qu'il y a une **COURSE**, quand, suite à un changement en entrée, le Σ passé par plusieurs états transitoires avant d'atteindre un état stable
 - La **cOURSE** est **INIFINIE**, si le Σ n'atteint jamais d'état stable => Σ Instable

Cheminement de base

E \ Q	...	E ₁	E ₂	...
...				
Q ₃		Q ₃	Q ₄	
Q ₄		Q ₄	Q ₄	
...				

Diagram illustrating the basic pathfinding process. It shows a state transition table with columns for input vectors (E) and rows for internal states (Q). The path starts at Q₃ for E₁, moves to Q₄ for E₂, and then stays at Q₄ for E₁. The final state Q₄ is circled, indicating it is a stable state.

Course

E \ Q	...	E ₁	E ₂	...
...				
Q ₃		Q ₃	Q ₄	
Q ₄			Q ₅	
Q ₅			Q ₅	

Diagram illustrating a finite course. The path starts at Q₃ for E₁, moves to Q₄ for E₂, then to Q₅ for E₁, and finally stays at Q₅ for E₂. The final state Q₅ is circled, indicating it is a stable state.

Course infinie

E \ Q	...	E ₁	E ₂	...
...				
Q ₃		Q ₃	Q ₄	
Q ₄			Q ₅	
Q ₅			Q ₃	

Diagram illustrating an infinite course. The path starts at Q₃ for E₁, moves to Q₄ for E₂, then to Q₅ for E₁, and finally back to Q₃ for E₂. The path never reaches a stable state, as it keeps cycling through different states.

Exemple : avec comme point de départ, l'état stable Q₃ et la combinaison d'entrée E₁

II.4. Trois représentations formelles d'un système séquentiel logique

➤ Représentation tabulaire pour le mode pulsé

- Remarque préliminaire : pour ce mode, il n'y a pas de phénomène de course puisque le système est **STABLE** entre 2 événements

e_i Q	N	...	e_2	...	N	...	e_2	N	...	e_2
...					...			0		
Q_i	(Q_i)	q_i	q_k		S_1		$S_1 + S_2$	0		S_1
...		Δt			...	Δt		0	Δt	
Q_k	(Q_k)		q_k		S_2			0		
...					...			0		

États suivants

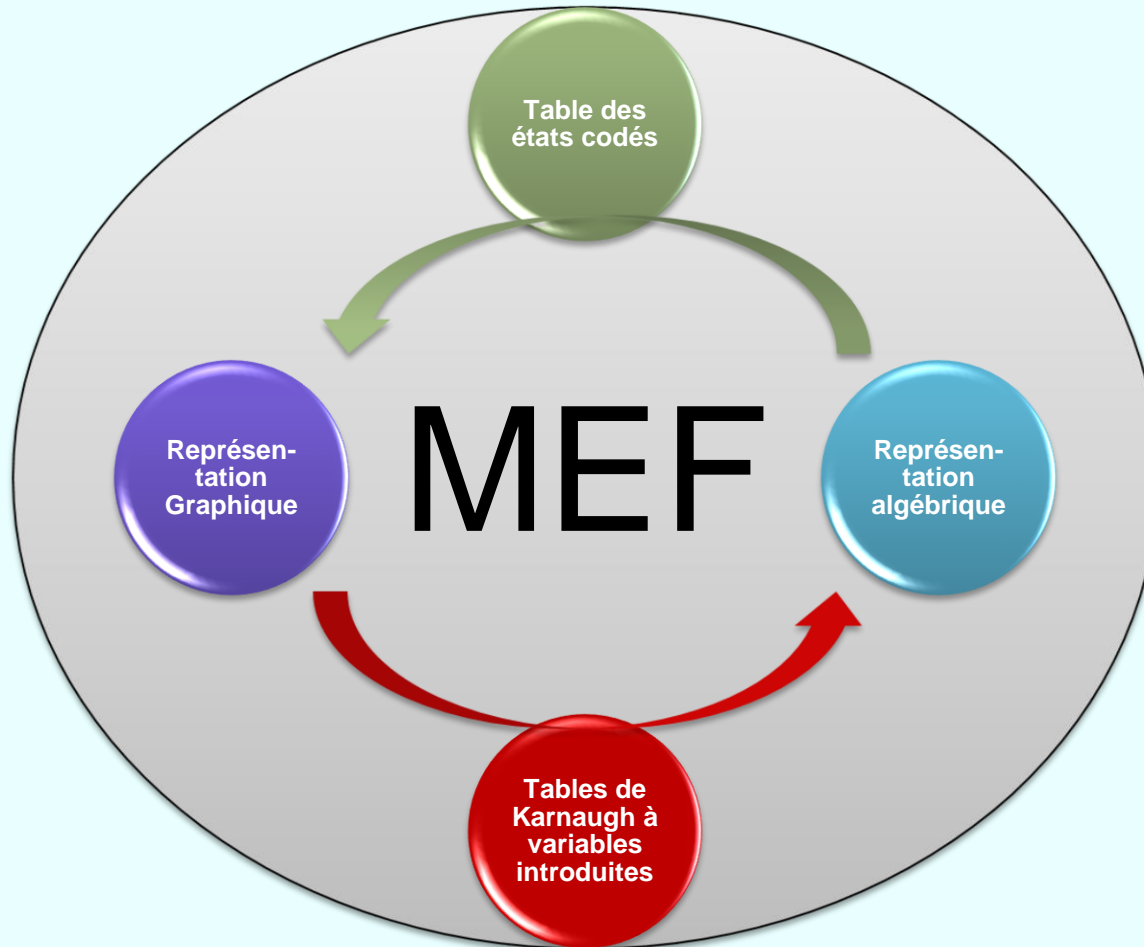
Sorties
Niveaux (sur état)
(machine de MOORE)

Sorties événementielles
(sur arc)
(machine de MEALY)

- Remarques
 - Δt = temps de commutation
 - La colonne N n'est pas toujours représentée
 - Sorties niveaux : définies sur les états
 - Sorties "événementielles" ou impulsionnelles : placées sur les arcs (transitions entre états) et sont donc nécessairement inactives sur les états d'où la colonne N à 0.

II.5. Passage d'une représentation à une autre

❑ Principe



II.5. Passage d'une représentation à une autre

❑ Tables de Karnaugh à variables introduites (TKVI)

➤ Inconvénients des TEC classiques

- taille fonction du nombre des états et surtout des entrées du systèmes
 - Taille = nombre d'états $\times 2^{\text{nombre d'entrées}}$
 - 8 entrées \rightarrow 256 colonnes
- La TK est généralement "creuse" (beaucoup de case inspecifiées), car le système n'est réactif qu'à une très petite partie de toutes les combinaisons d'entrées possibles (certaines ne pouvant même pas se produire)

➤ Principe TKVI

- On fait 1 TKVI pour chaque variable interne et pour chaque sortie
- TKVI=TEC=TK **MAIS** on ne fait plus apparaître les entrées dans les colonnes
- Composantes Y_i de \underline{Y} scindées en 2 sous-ensembles, codés en binaire RÉFLÉCHI
 - Y_1, Y_2, \dots, Y_j sont placées en entête des lignes de la TKVI
 - $Y_{j+1}, Y_{j+2}, \dots, Y_N$ sont placées en entête des colonnes de la TKVI
- Chaque case de la TKVI correspond donc à un état interne et on indique pour la VI considérée sa valeur suivante **en l'exprimant comme une fonction logique, fonction dépendant des valeurs des composantes de E**

II.5. Passage d'une représentation à une autre

➤ Forme d'une TKVI : exemple avec 4VI

$Y_1 Y_2 \backslash Y_3 Y_4$	00	01	11	10
00	-	-	-	-
01	-	$f_1(\underline{E})$	$f_2(\underline{E})$	-
11	-	0	$f_3(\underline{E})$	-
10	0	1	1	-

y1

➤ Dédution des équations

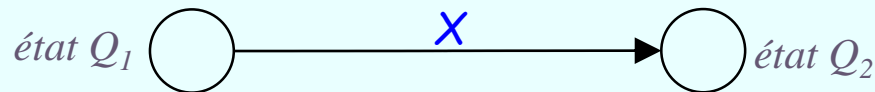
- ① Considérer toutes les Expressions Logiques Introduites (ELI) comme à 0, et faire les plus grands regroupement de 1, y compris, le cas échéant, avec des valeurs inspecifiées (les '-').
- ② Considérer une (et une seule) des ELI comme étant à 1 et toutes les autres à 0, et faire alors les plus grands regroupement sur la base de cette ELI (en utilisant si besoin est, des points à 1, voire des valeurs inspecifiées)
- ③ Recommencer l'étape ② avec une autre ELI, jusqu'à avoir traité toutes les ELI.

➤ Exemple

II.5. Passage d'une représentation à une autre

➤ Remplissage d'une TKVI à partir du diagramme d'états

- tout changement d'état est conditionné par le passage à Vrai d'une *condition* (passage de 0 \rightarrow 1 de cette *condition*)
- Quand le système passe de Q_i à Q_k , il faut déterminer le lien entre "l'évolution" de la valeur de Y_i et la *condition*, source du changement d'état
- Comme on est dans le monde binaire, il n'y a que 3 types de liens
 - ① la variable ne change pas de valeur lors du changement d'état : aucun lien $y_i=0$ ou 1
 - ② évolution identique : $y_i = X$
 - ③ évolution opposée : $y_i = \neg X$



$$Y_I = 0$$

$$Y_I = 1$$

$$Y_I = 0$$

$$Y_I = 1$$

$$Y_I = 0$$

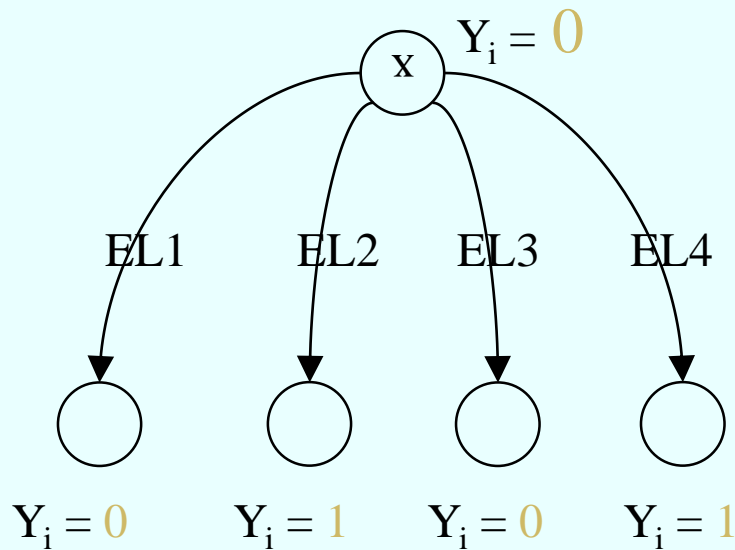
$$Y_I = 1$$

$$Y_I = 1$$

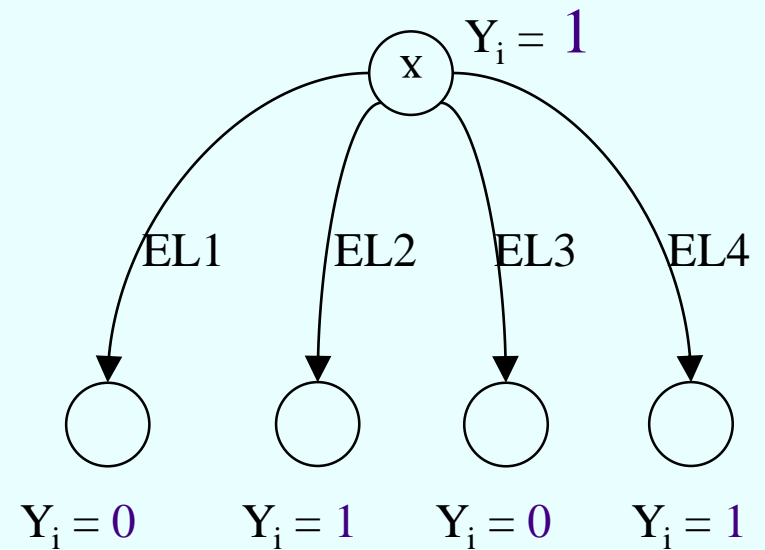
$$Y_I = 0$$

II.5. Passage d'une représentation à une autre

➤ Remplissage d'une TKVI à partir du diagramme d'états



Dans la TKVI de y_i , dans la case
Correspondant au code de l'état x , il faut
écrire : $EL2 + EL4$



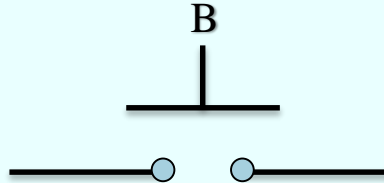
Dans la TKVI de y_i , dans la case
Correspondant au code de l'état x , il faut
écrire : $\neg EL1 . \neg EL3$

II.5. Passage d'une représentation à une autre

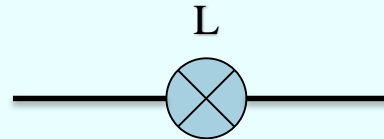
❑ Exemple : Télerrupteur

Système :

- Un bouton poussoir B



- Une lampe L



Fonctionnement :

- 1 appui sur B allume la lampe.
- Nouvel appui sur B, la lampe s'éteint.
- ...

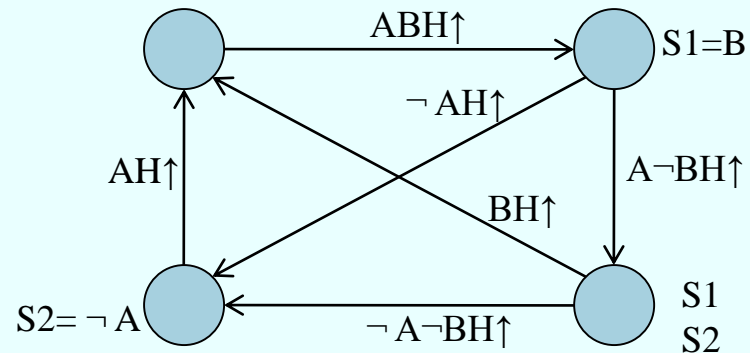
Machine de traitement : machine de Moore !

II.5. Passage d'une représentation à une autre

❑ Exemple : représentation graphique \leftrightarrow algébrique

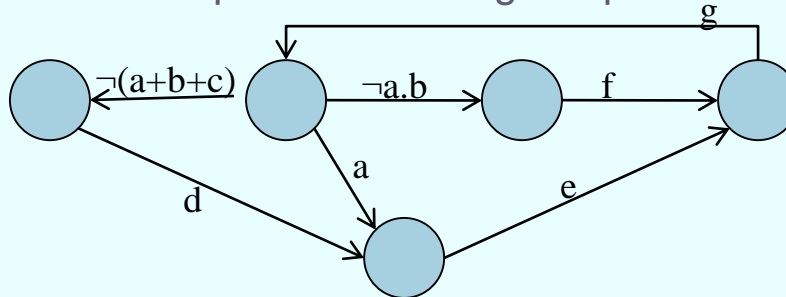
➤ Cas synchrone

- Passer à une représentation algébrique



➤ Cas asynchrone

- Passer à une représentation algébrique



- Passer à une représentation graphique

$$\begin{cases} y_1 = Y_2 + B.Y_1 + A.Y_1 \\ y_2 = \neg Y_1.Y_2 + A.\neg Y_1 + \neg A.Y_1 + B.Y_2 \\ S = Y_1.\neg Y_2 \end{cases}$$

II.6. Analyse d'un système séquentiel logique

□ Définition

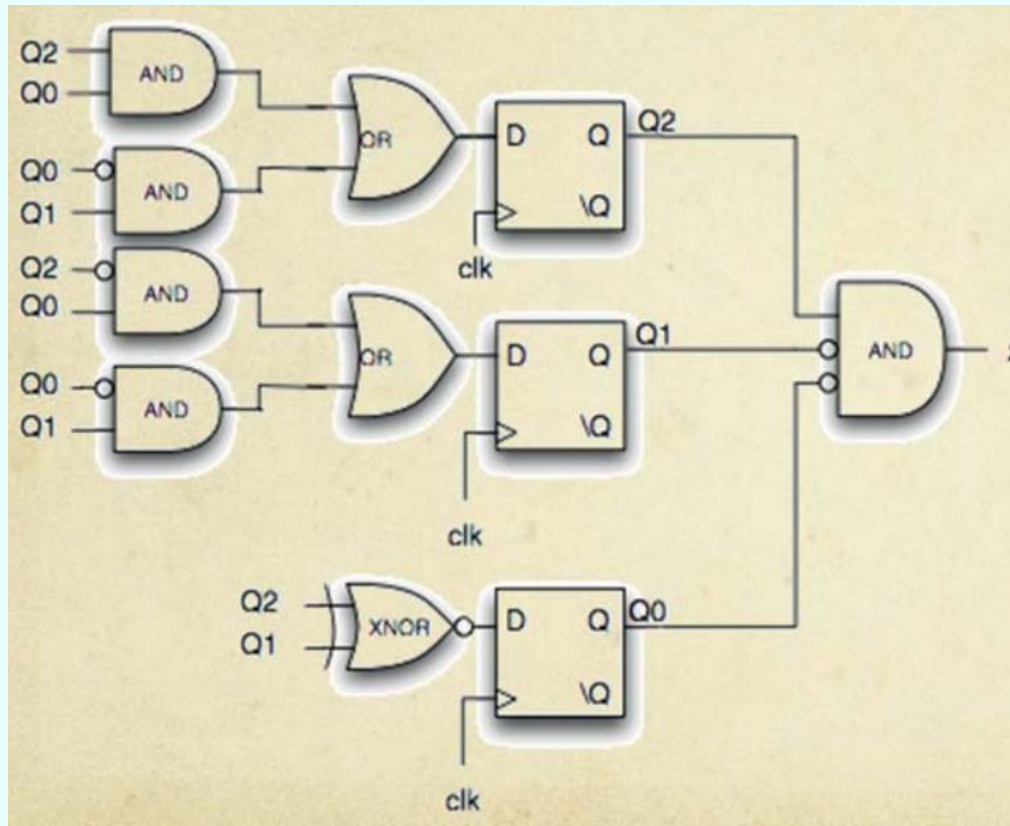
- Analyse = étudier le fonctionnement d'une réalisation déjà existante pour comprendre le système correspondant à cette réalisation → partant d'une réalisation, "remonter" à un modèle "abstrait" de son fonctionnement (ici ce sera une MEF), voire au Cahier des Charges
- La "réalisation" à analyser peut être sous une forme de : schéma électrique (ou logigramme), chronogramme, programme informatique (automate, langage de haut niveau, ...) ou autre encore ... Le cours présente la démarche au travers d'un exemple de réalisation électrique, les autres formes seront vues en TD.

□ Méthode

1. Identifier les Entrées et les Sorties du système
2. Identifier les Variables Internes utilisées pour coder les états (internes) du Σ
3. Déterminer les Expressions Logiques représentatives des blocs F et G (modèle FMG)
$$\underline{y} = F(\underline{E}, \underline{Y}) \quad \text{et} \quad \underline{S} = G(\underline{E}, \underline{Y})$$
4. Construire la table des états CODÉS (pour les états suivants et pour les sorties)
5. {En déduire la table des états NOMMÉS, après avoir choisi une association code \leftrightarrow état}
6. En déduire le Graphe d'état (la MEF = Machine à États Fini)
7. Donner une interprétation textuelle du fonctionnement → rédiger le Cahier des Charges

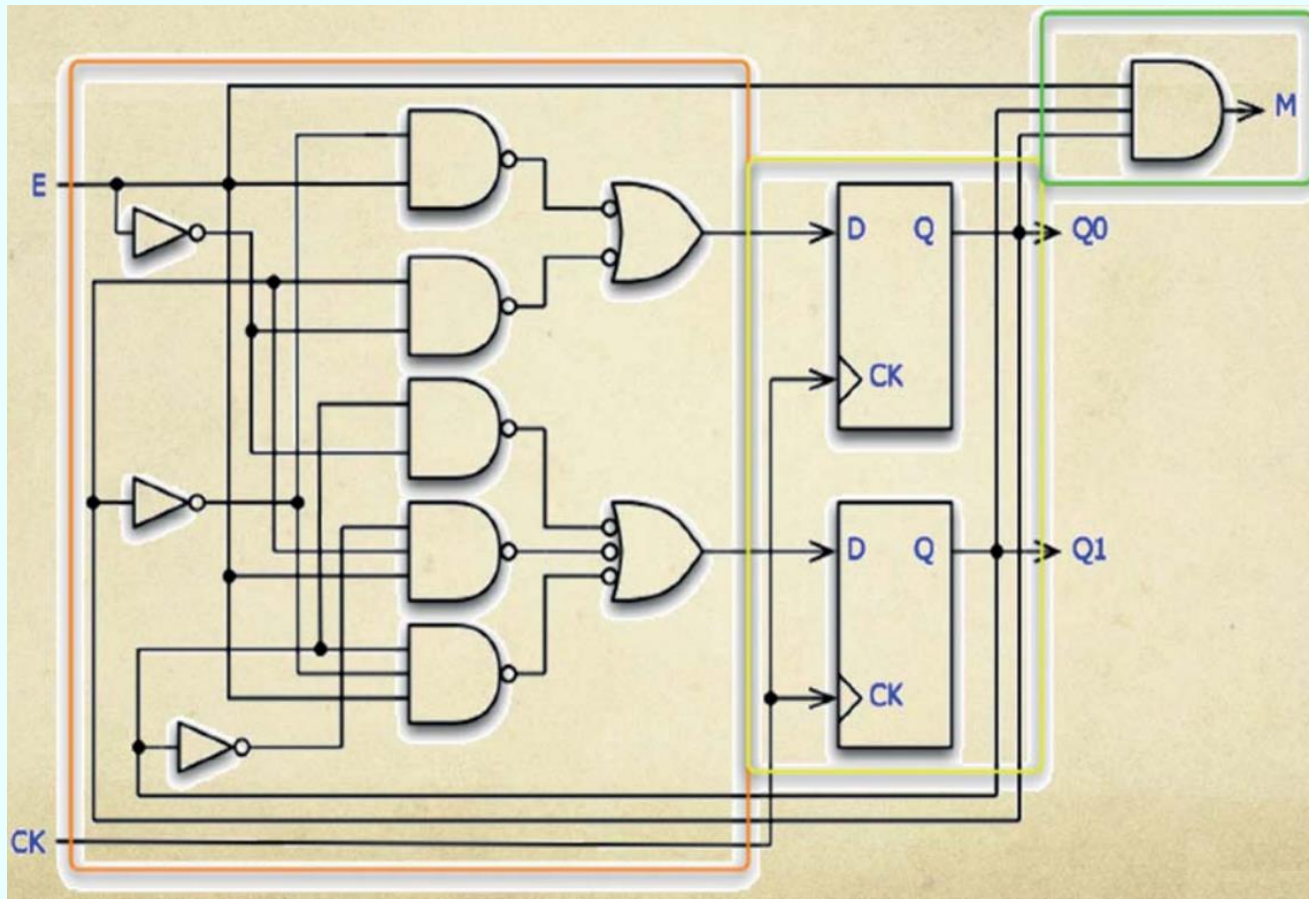
II.6. Analyse d'un système séquentiel logique

❑ Exemple : machine de Moore



II.6. Analyse d'un système séquentiel logique

❑ Exemple : machine de Mealy



II.7. Modélisation de systèmes séquentiels logiques

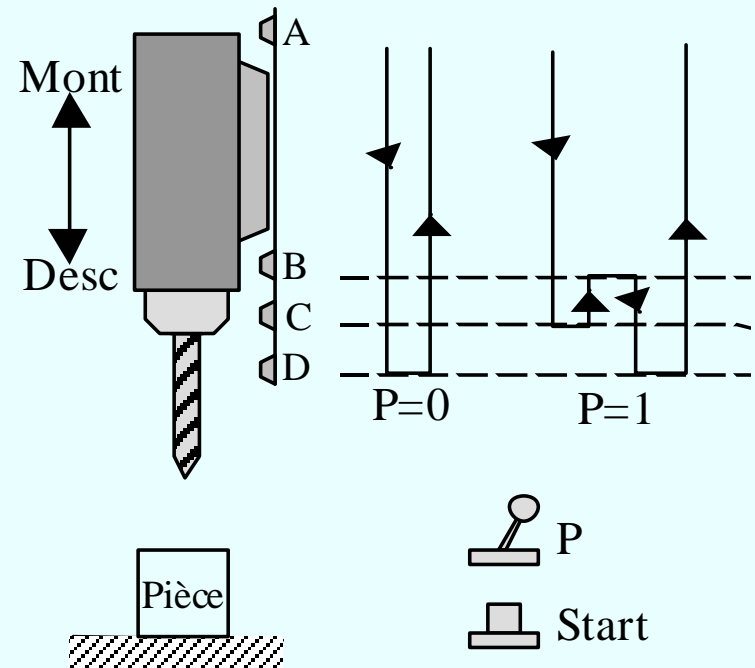
❑ Objectif

- Concevoir une MEF respectant un besoin exprimé dans un cahier des charges
- Expérience du concepteur prépondérante

❑ Exemple

➤ Cahier des charges

- Commander le mandrin (Mont/Desc)
- Respect des 2 cycles
 - Pièce mine $\Rightarrow P=0$
 - Pièce épaisse $\Rightarrow P=1$
- Start est pris en compte ssi A est vrai



II.7. Modélisation de systèmes séquentiels logiques

☐ Méthode

- a) Identifier les Entrées et Sorties : $\underline{E} = [A, B, C, D, P, \text{Start}]$, $\underline{S} = [\text{montée, descente}]$
- b) Identifier les états et "entrées" significatives → remplir le tableau suivant

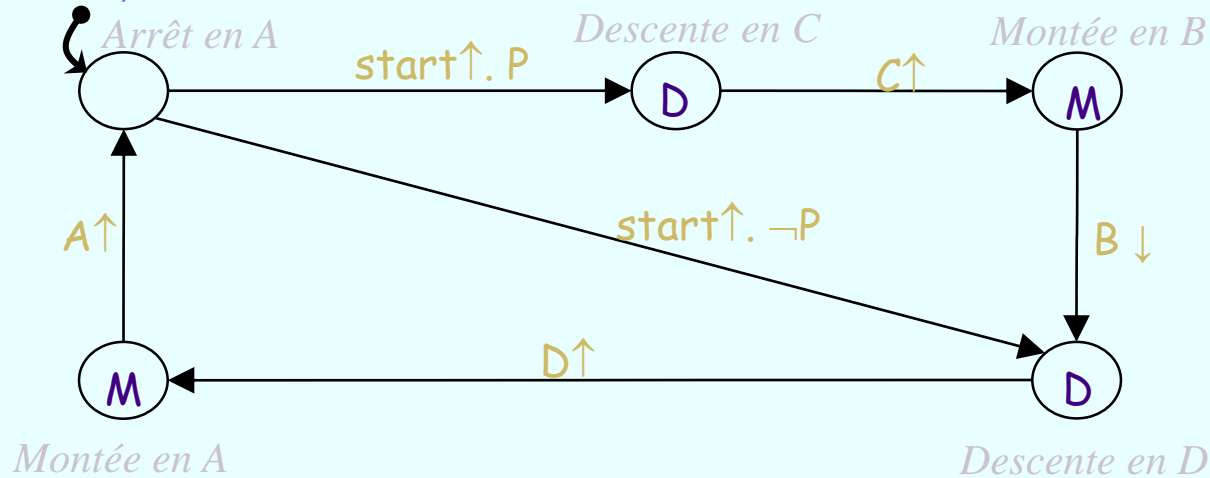
états	Événements significatifs	États suivants
Arrêt en A	Start. $\neg P$	Descente en D
	Start.P	Descente en C
Descente en D	D	Montée en A
Descente en C	C	Montée en B
Montée en A	A	Arrêt en A
Montée en B	B	Descente en D

- c) Déterminer les modes de fonctionnement possibles et en choisir un → ici, on peut choisir le mode synchrone (pulsé), mais on aurait aussi pu choisir le mode fondamental (les choix faits ci-dessous ne changent rien).
- $\{A, B, C, D, \text{Start}\}$ = événements (impulsion) → c'est un point de vue abstrait car, en réalité, par nature, ce sont des niveaux
 - P = niveau

II.7. Modélisation de systèmes séquentiels logiques

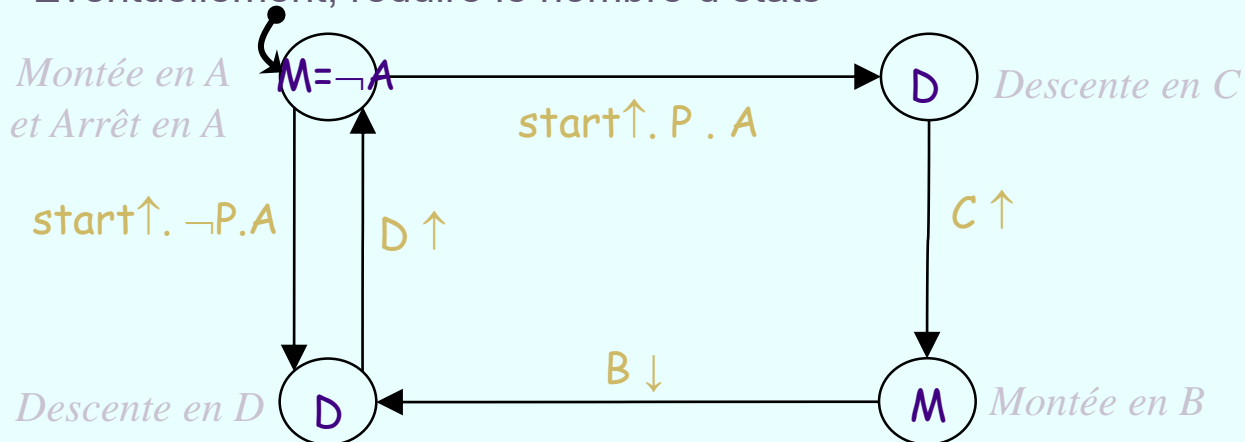
□ Méthode (suite)

d) Modéliser



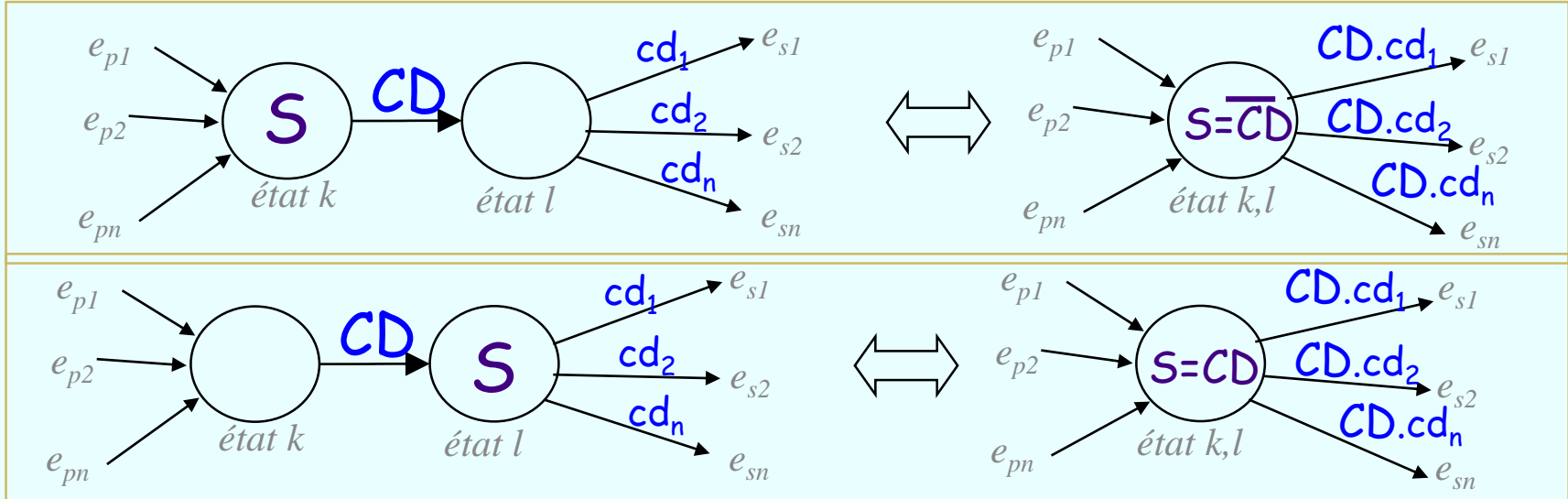
RAPPEL : pour un état avec plusieurs arcs sortants, toujours s'assurer que les conditions étiquetant ces arcs sont mutuellement exclusives

e) Éventuellement, réduire le nombre d'états



II.7. Modélisation de systèmes séquentiels logiques

❑ Quelques mots sur la réduction de MEF



ATTENTION :

1. le regroupement n'est possible que si les conditions (sur les arcs), ici **CD** et cd_i ne sont pas mutuellement exclusives \rightarrow sinon blocage
2. L'entrée **CD** doit se maintenir pendant tout le temps où le système est dans l'état *état l*
3. **Cas où un regroupement est totalement impossible**

