ITI 1500 Hiver 2013

Systèmes Numériques I

Cours

Lundi	11:30 - 13:00	SCS E217
Jeudi	13:00 - 14:30	SCS E217

TUTORIAL Mardi 8 :30-10:00 salle: LMX-221

Professeur : Dr A. Karmouch, Bureau: CBY-A508

Chapitre 5

Circuits Logiques Séquentiels

(Rappel chapitre 2) : Stockage binaire et Registres

• Dans l'ordinateur les informations binaires doivent avoir une existence physique sur une médium de stockage.

• Cellule Binaire

- un dispositif physique avec deux états
- Capable de stocker UN BIT d'information

→ **L'entrée** de la cellule reçoit un signal excitation qui la force a se mettre dans un des deux états.

→ **la sortie** de la cellule permet de fournir un des deux états

→ L'information stockée dans la cellule est « 1 » quand elle est dans un état et « 0» quand elle est dans l'autre état.

(Rappel chapitre 2) : Registres

- Un registre est un groupe de cellule binaire
- un registre avec *n* cellules peut stocker *n* bits

Exemple:

- Un Registre avec 8 cellules peut stocker 8 bits qui peut un des 2^8 états possible

0	0	0	0	0	0	1	1

Le contenu peut être interprété différemment. Ici c'est un entier positive dont la valeur est (+3)

(Rappel chapitre 2) : Transfert de données entre registres

- Tout system numérique est composé
  - registres
  - Unité de traitement de données
- les informations binaires sont transférée d'un ensemble de registres a un autre
- Transfert peut être effectuer directement ou via l'unité de traitement pour faire une opération

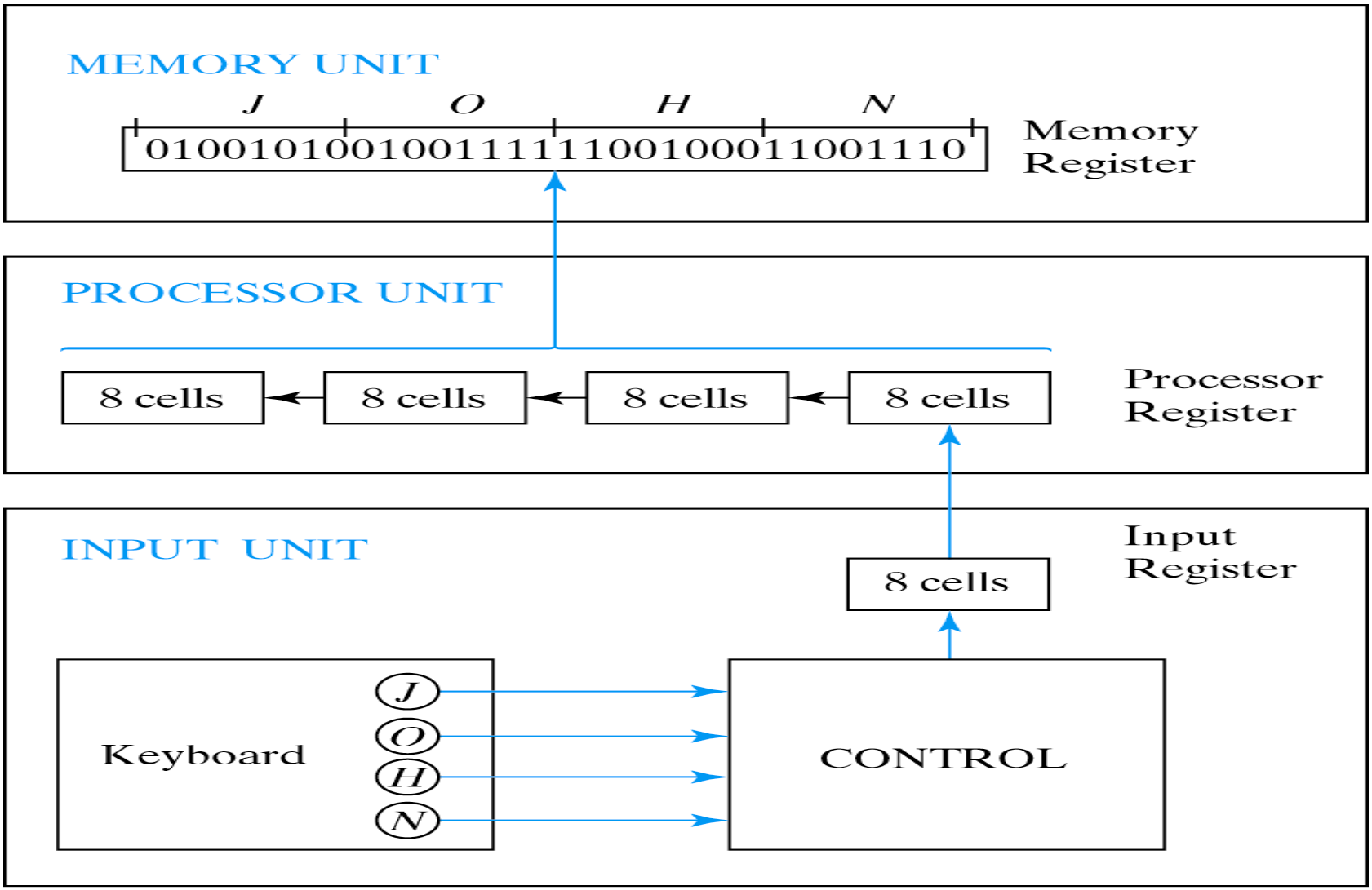


Fig. 1-1 Transfer of information with registers

(Rappel chapitre 2) :Transfert de données entre registres [2]

• ***pour traiter des information binaires l'ordinateur doit disposer de:***

1- dispositifs physique pour mémoriser l'information:

→ ***REGISTRE*** **(**le plus communément utilisé **)**

2- les Circuit pour manipuler les bits d'information:

→ **Circuits logiques numériques**

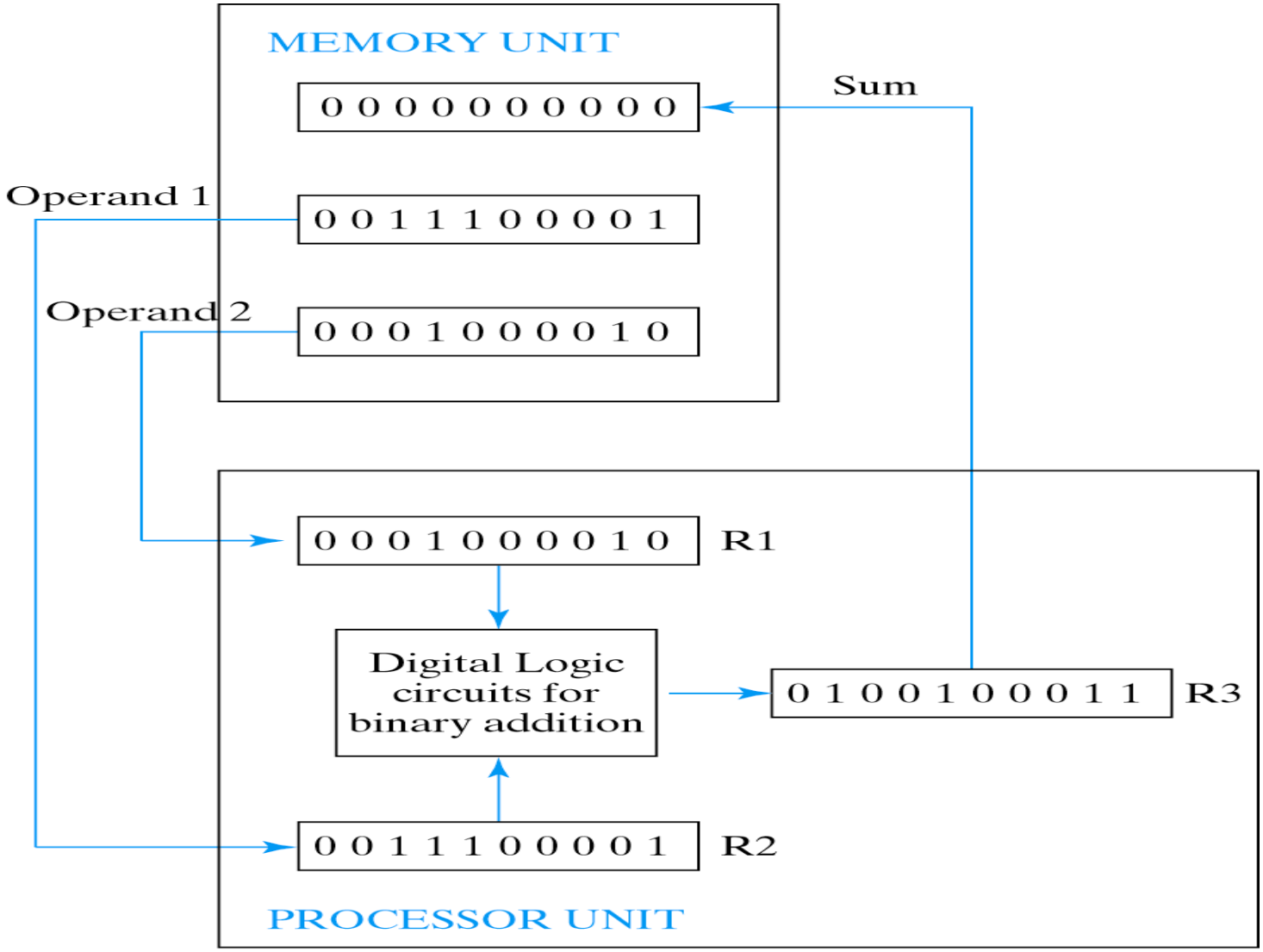


Fig. 1-2 Example of binary information processing

Circuits logiques Séquentiels -Définitions

→Fournissent en sorties la fonction logique des entrées et des états précédents du circuit (mémoire)

→ après changement des entrées les nouvelles sorties apparaissent après un signal d'horloge

→ **boucle de retour**

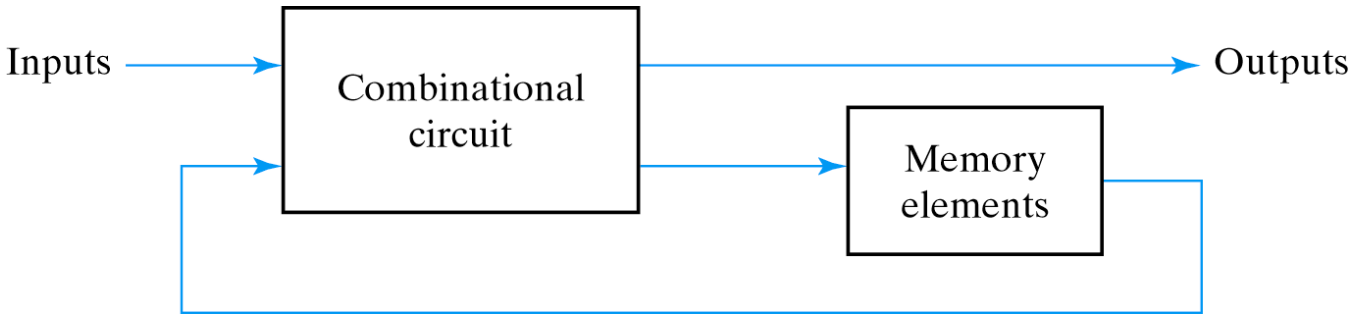


Fig. 5-1 Block Diagram of Sequential Circuit

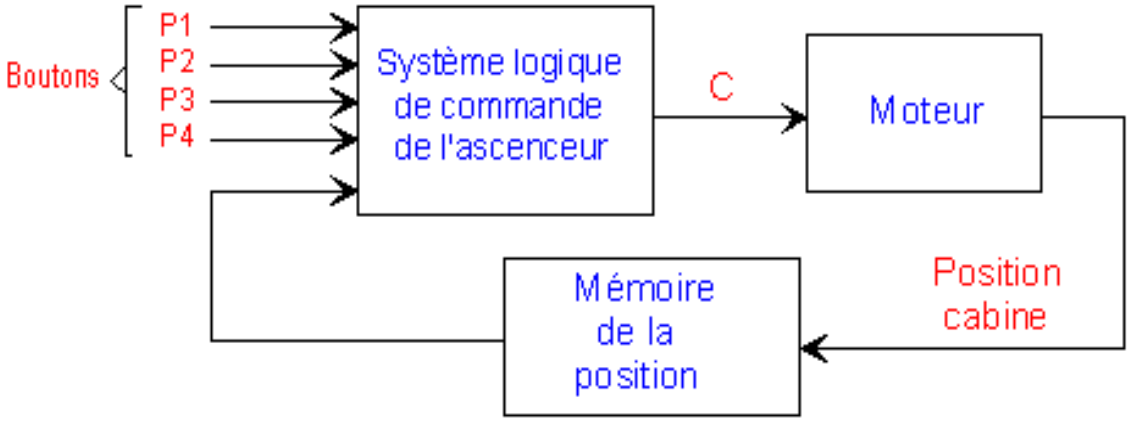
Circuits logiques Séquentiels -Exemple

-- Commande d'une cabine d'ascenseur--

•Les boutons sont les variables d'entrée du système

• la position est aussi une variable d'entrée secondaire.

• Ce schéma très simplifié montre que la fonction commande 'C' dépend des états des boutons d'appel ET de la position de la cabine, donc des *états antérieurs du système* (Etats avant l'activation des boutons d'appel).



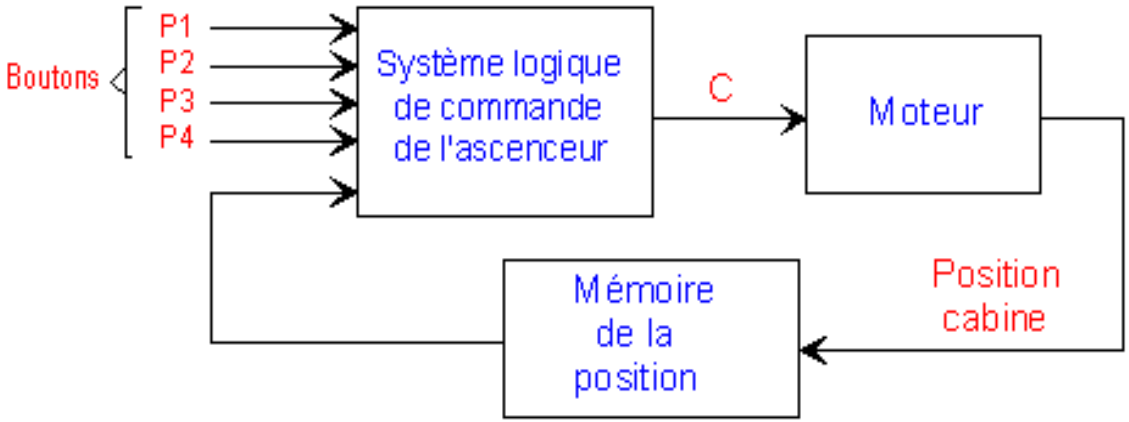
Circuits logiques Séquentiels -Exemple (2)

-- Commande d'une cabine d'ascenseur--

Cet exemple correspond à un système logique SEQUENTIEL c'est à dire que la sortie C à un instant donné dépend

1 - des variables d'entrées (variables principales)

2 - **ET** des ETATS ANTERIEURS (ETATS INTERNES).



Circuits logiques Séquentiels -Exemple (3)

-- Commande d'une cabine d'ascenseur--

→ Par exemple si l'on appuie sur le bouton du cinquième étage, l'action à effectuer sera différente si la cabine est au premier étage ou au quatrième étage.

→On peut dire que *le système garde la mémoire* de la séquence d'événements qui ont *précédé*.

→ Par suite à une *même combinaison des variables d'entrée* correspondent *différents états de la sortie*.

Circuits logiques Séquentiels : temps et mémoire

→ *Par rapport aux circuits combinatoires, deux paramètres nouveaux interviennent:*

1 - *Le temps*

On doit tenir compte des temps de propagation des circuits.

2 - ***L'élément mémoire***

Il est nécessaire pour garder la trace des résultats précédents et imposer un ordre prédéterminé.

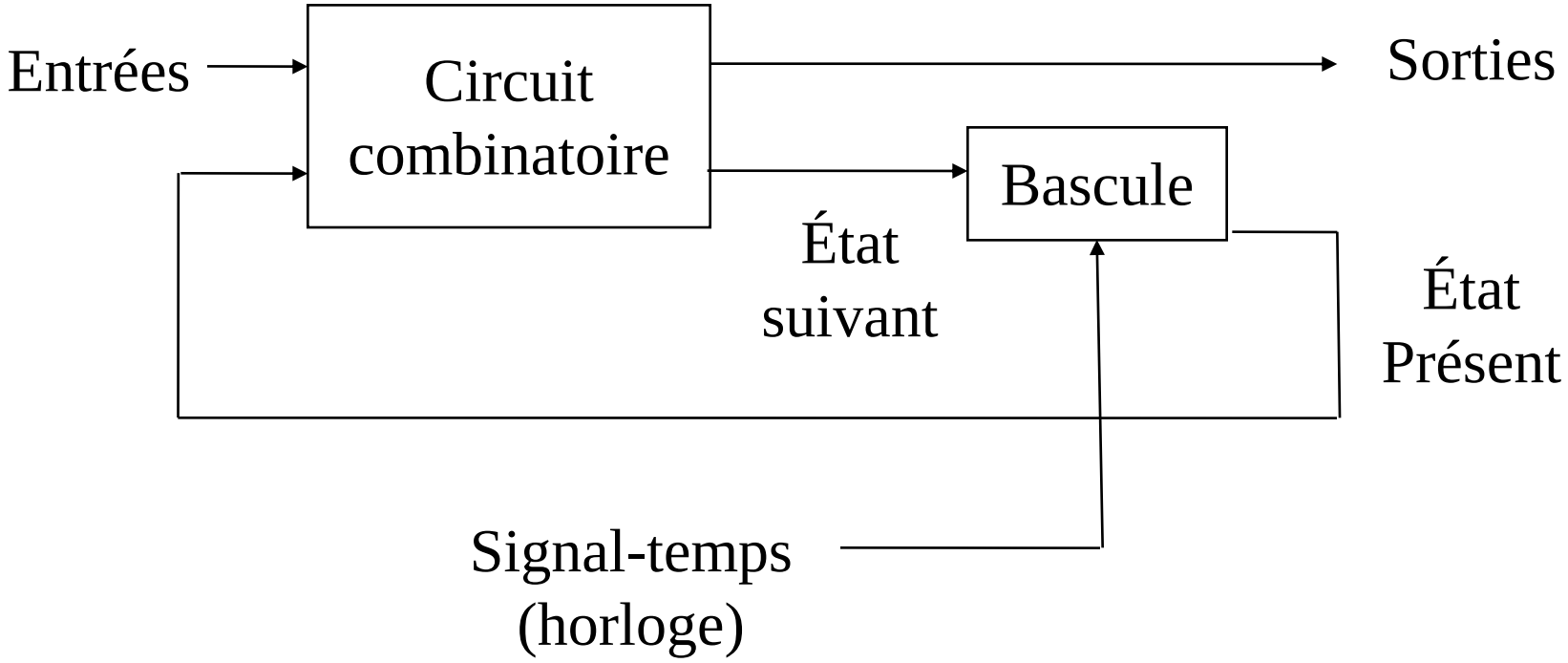
→ D'une manière générale la fonction mémoire consiste à STOCKER DES INFORMATIONS sous forme binaire .

Circuits logiques Séquentielles

• La cellule mémoire qui stocke des informations de "un bit" s'appelle une BASCULE (Flip-flop).

• Un ensemble de bascules commandées simultanément permet de stocker un mot de plusieurs bits; Il s'agit alors d'un REGISTRE.

• Deux modes de fonctionnement: Asynchrone et Synchrone (exemple ci dessous)

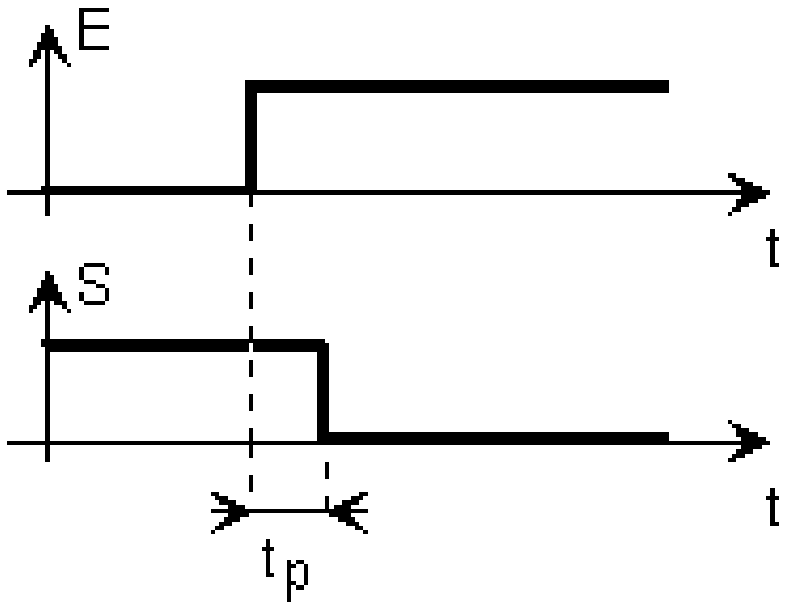


Circuits logiques Séquentiels

1- Fonctionnement Asynchrone

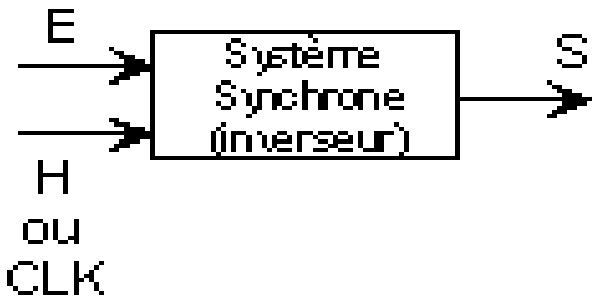
• La sortie réagit "immédiatement" à la modification des variables d'entrée

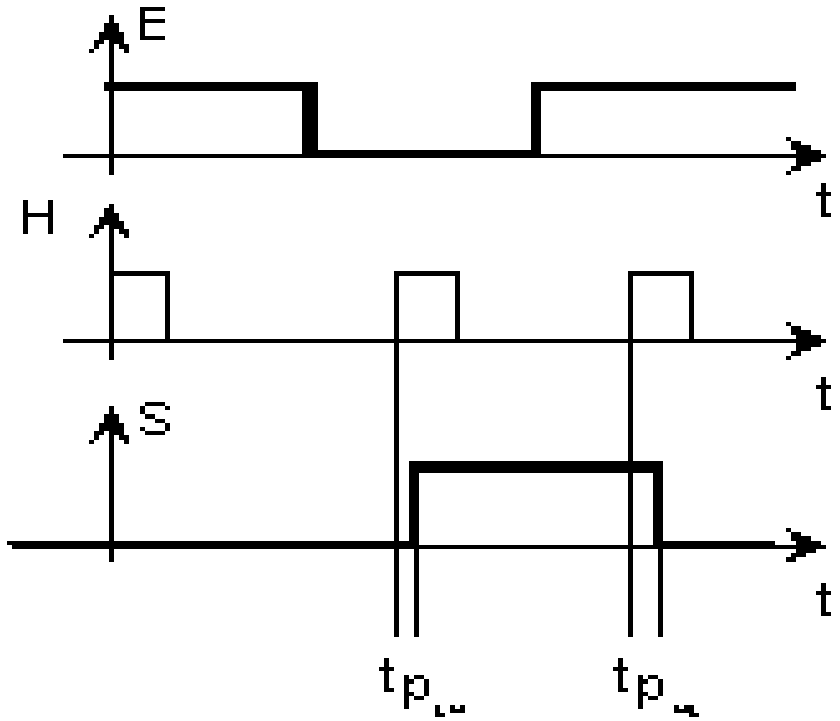
→ le retard propre des opérateurs ne peut cependant pas être négligé --- > t_p



2- Fonctionnement Synchrone

• Dans ce mode de fonctionnement, la modification de la configuration des variables d'entrées n'est **EFFECTIVE** que lorsqu'un signal supplémentaire appelé **HORLOGE** devient actif.





Le signal d'Horloge : définition

•L'horloge est utilisé dans les circuits logiques synchrones pour provoquer le moment exacte du changement d'état du circuit.

• L'horloge est un signal périodique ou non de type impulsion.

• Ce signal peut agir suivant trois modes (en ce qui nous concerne ici):

- Par Etat
- Par Front montant ou Positif
- Par Front descendant ou Négatif

Le Signal d'Horloge : définition (2)

1-Action du signal par Etat

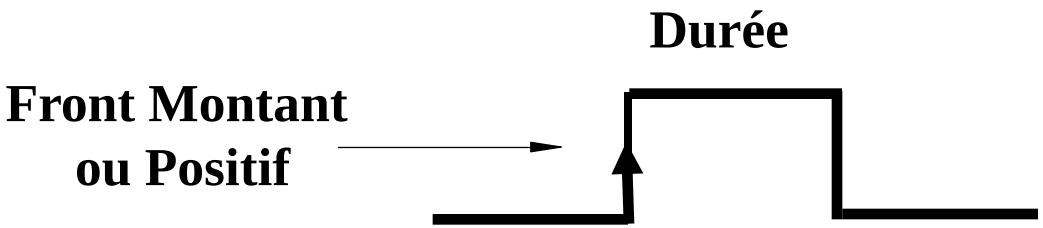
• Tant que Horloge est à l'état logique actif (e.g "1") la commande est effective.

• L'état logique "0" est alors l'état de repos.

Le signal d'Horloge : définition (3)

2- Action du signal par Front Montant

La commande n'a lieu que lors de la *transition d'un état 0 à 1* , dite commande sur front positif ou montant)

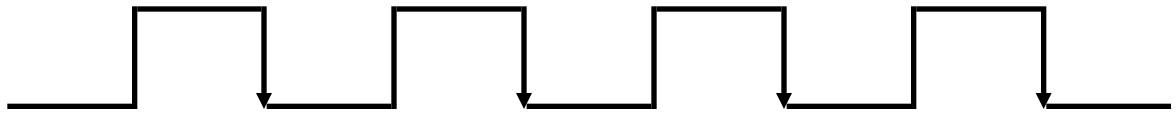


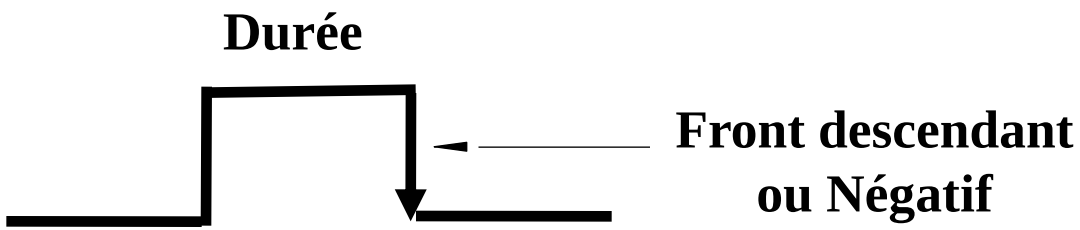
Le signal d'Horloge : définition (4)

3- Action du signal par Front Descendant

La commande n'a lieu que lors de la *transition d'un état 1 à 0* , dite commande sur front négative ou descendant

dans le deux cas La durée de la commande est alors très courte.





Élément de mémoire (Latch)

- Une bascule peut maintenir un état binaire (0 ou 1), tant qu'il y du courant qui est fourni, jusqu' a ce qu'il reçoit une commande de changement d'état.
- Les Bascules les plus basic fonctionnent en asynchrone et sont appelés Latch.
- Les Bascules Latch servent de base pour construire tous autres types de Bascules

Élément de mémoire (Latch SR)

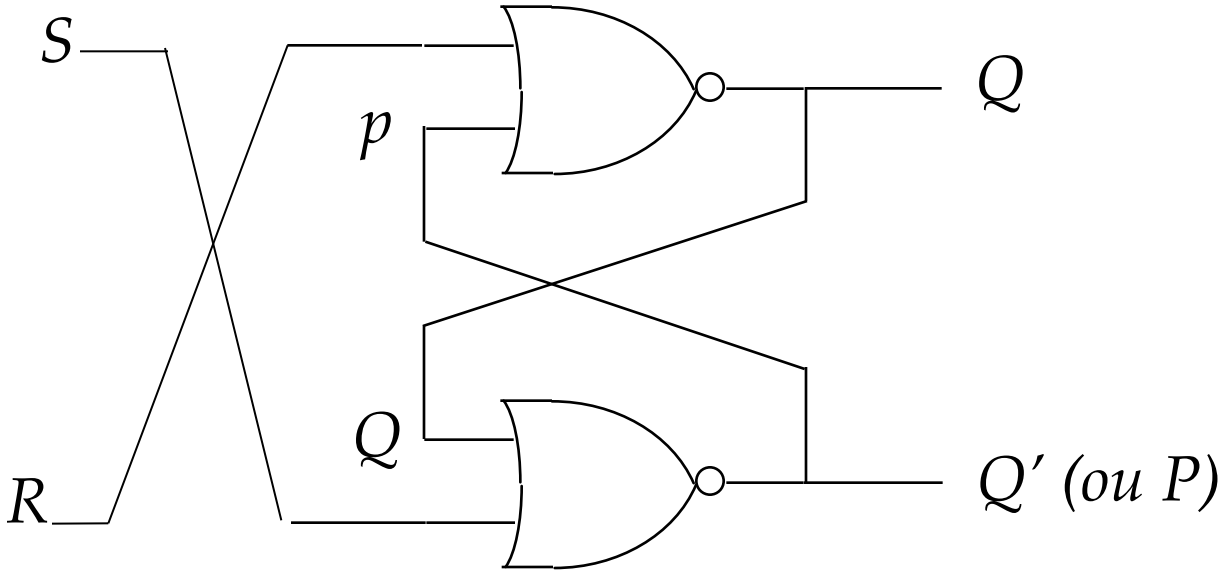
- Bascule (Latch) SR possède deux entrées notées S et R
  - S pour SET
  - R pour RESET

Et deux sorties Q (état normal) et Q’ (état inverse)

Symbole

Élément de mémoire (Latch SR)

1- Implémentation avec des NON-OU



Q = (R+P)'= R'P'

P = (S+Q)'= S'Q'

Élément de mémoire (Latch SR)

Symbole

Table de vérité pour RS avec des NON-OU:

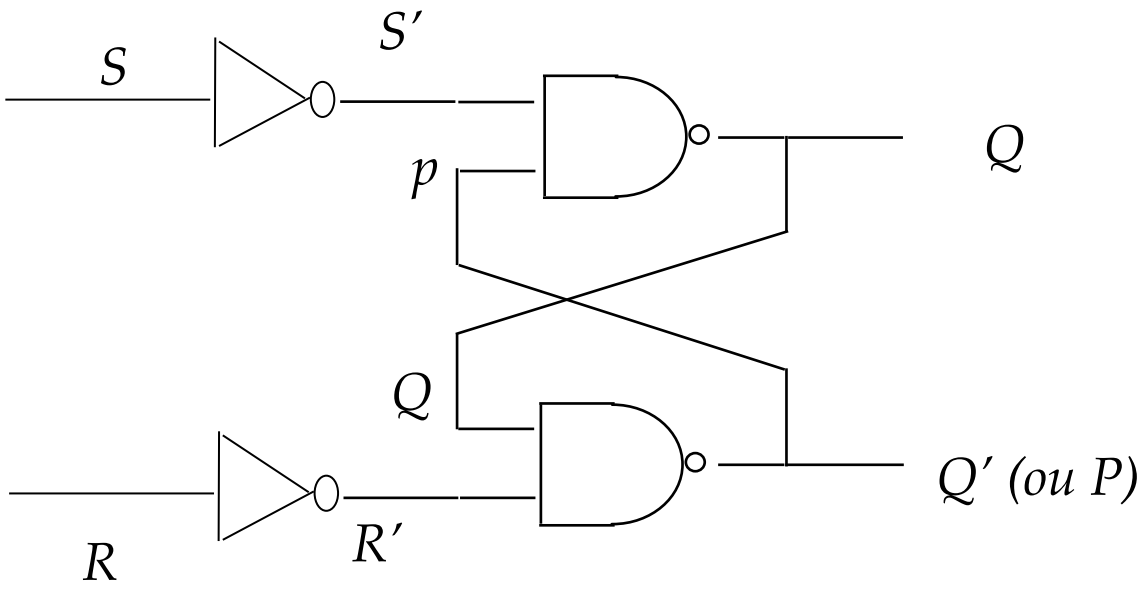
S	R	Q	Q'	
1	0	1	0	Mise à "1"
0	1	0	1	Mise à "0"
0	0	1	0	Sortie inchangée (après SR = 10)
0	0	0	1	Sortie inchangée (après SR = 01)
1	1	?	?	À ne pas faire! (00)

Élément de mémoire (Latch SR)

2- Implémentation avec un Inverseur et des NON-ET

Q= (S'.P)' = S + P'

P= (R'Q)' = R+Q'



Élément de mémoire (Latch SR)

2- Implémentation avec un inverseur et des NON-ET

Aussi dit S'R' Latch

Table de vérité pour RS avec inverseur et des NON-ET :

S	R	Q	Q'	
1	0	1	0	Mise à "1"
0	1	0	1	Mise à "0"
0	0	1	0	Sortie inchangée (après SR = 10)
0	0	0	1	Sortie inchangée (après SR = 01)
1	1	?	?	À ne pas faire! (00)

Élément de mémoire (Latch SR)

2- Implémentation avec des NON-ET

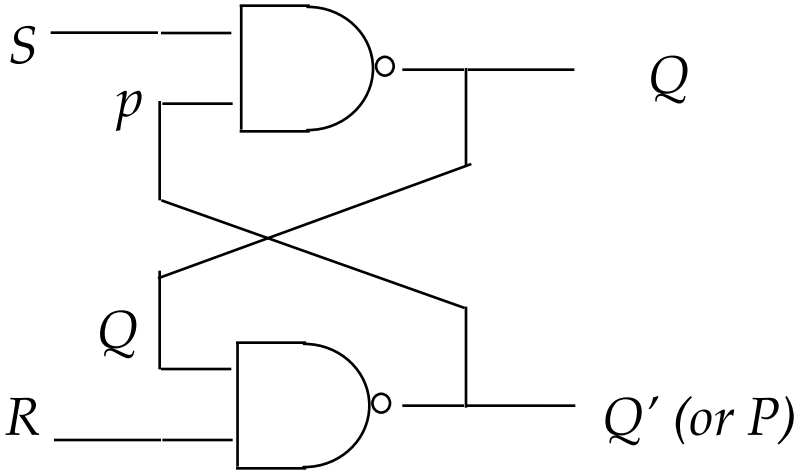


Table de vérité pour RS avec des NON-ET:

S	R	Q	Q'	
1	0	0	1	Mise à « 0 »
1	1	0	1	Sortie inchangée (après SR = 10)
0	1	1	0	Mise à « 1 »
1	1	1	0	Sortie inchangée (après SR = 01)
0	0	?	?	À ne pas faire! (1 1)

Circuits séquentiels synchrones

• Si on ajoute un signal de synchronisation (impulsion d'horloge ***H*** ou ***C***) à l'élément de mémoire RS, on obtient un circuit séquentiel synchrone appelé *bascule bistable RS*.

• Ce type de circuit est plus complexe, mais plus fiable que les circuits séquentiels asynchrones.

• Bascules bistables RS, D, T, JK (en anglais: *flip-flops*)

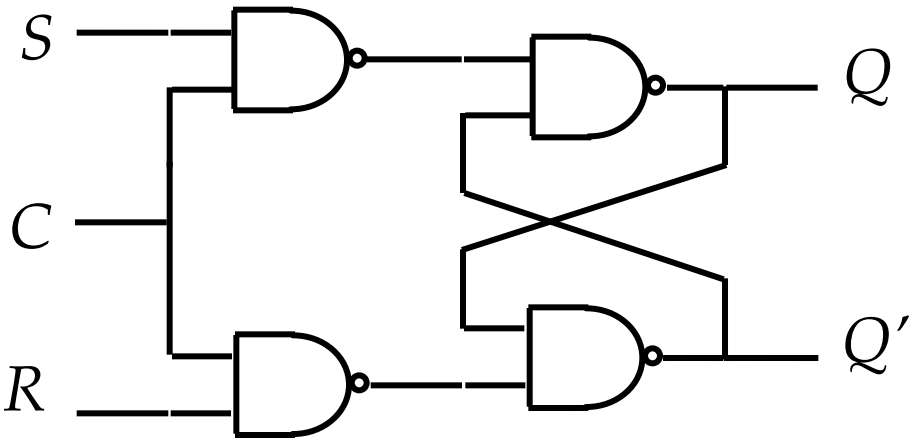
Nous allons étudier ces bascules uniquement

SR avec une entrée de contrôle

• l'entrée de contrôle est utilise de deux manière.

1- comme signal ON/OFF

2- comme un signal de synchronisation.

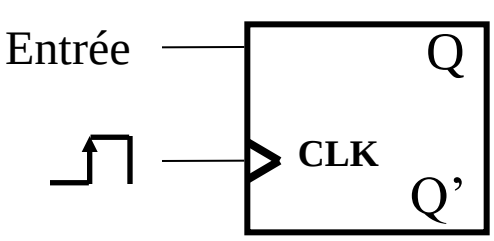


C	S	R	Qt_{+1}	$Q't_{+1}$	Fonction
1	0	0	Qt	Q't	inchangé
1	0	1	0	1	RESET
1	1	0	1	0	SET
1	1	1	0	0	Non-auto
0	X	X	Qt	Q't	Inhibé.

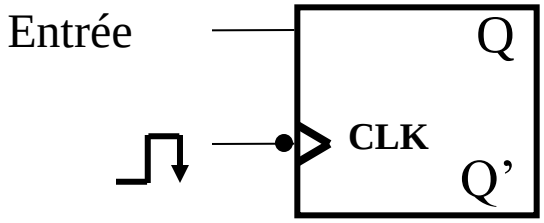
- Il est utile de ne pas permettre le changement d'état
- *Quand C = 0* → empêche tout changement d'état dans le Latch
- Signal de control *permet* de changer l'état quand C = 1
- Le côté droit du circuit est le même qu'un S-R Latch avec NON-ET

Bascules Synchrone :définitions

- C'est un Latch avec une entrée horloge.
- La sortie du circuit change quand son entrée Horloge détecte un Front (transition)
- Sensible au front plutôt qu'au niveau
- Symbole est un triangle sur l'entrée CLK (clock) de la Bascule

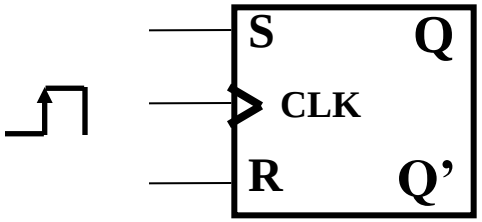


Symbol avec Front Montant

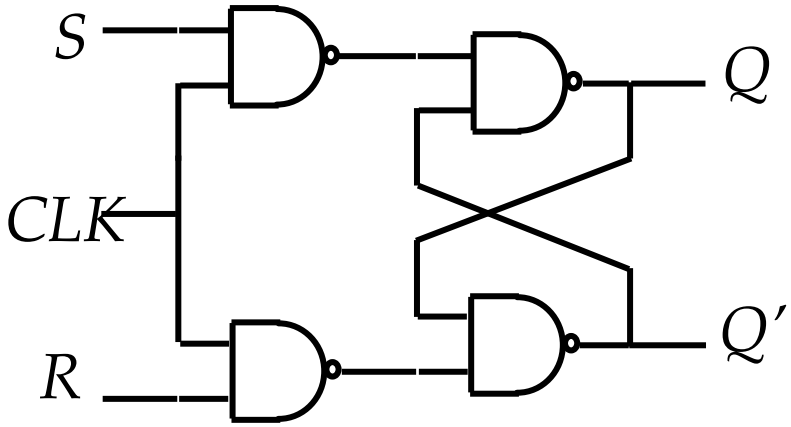


Symbol avec front descendant

La Bascule SR Synchrone



Symbol

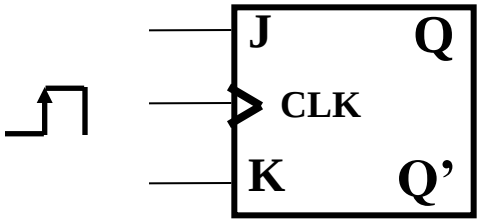


S	R	Q_{n+1}	Function
0	**0**	Q_n	**inchangé**
0	**1**	**0**	**RESET**
1	**0**	**1**	**SET**
1	**1**	**?**	**Non permis**

- Q_n = état avant front montant
- Q_{n+1} = état après front montant

La Bascule JK

Symbol



J	K	Q_{n+1}	Function
0	0	Q_n	inchangée
0	1	0	RESET
1	0	1	SET
1	1	Q'_n	complément

Même que SR sauf pour

K=J= 1, flip-flop JK passe a l'opposé Q_n. (complément)

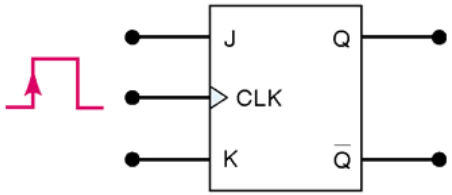
- Q_n = état avant front montant
- Q_{n+1} = état après front montant

Si $Q_{n.} = 1$ alors $Q_{n+1} = 0$

si $Q_{n.} = 0$ alors $Q_{n+1} = 1$

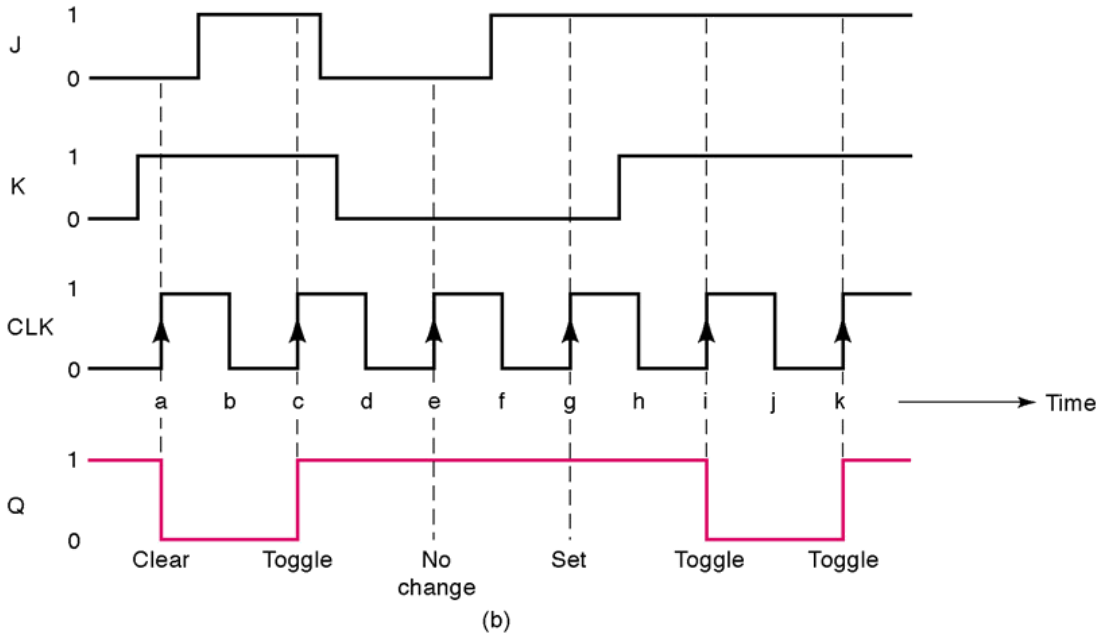
La Bascule JK Synchrone

- Deux entrées de données, J et K
- J -> set, K -> reset, if J=K=1 sortie est complémentée



J	K	CLK	Q
0	0	↑	Q_0 (no change)
1	0	↑	1
0	1	↑	0
1	1	↑	$\overline{Q}_0$ (toggles)

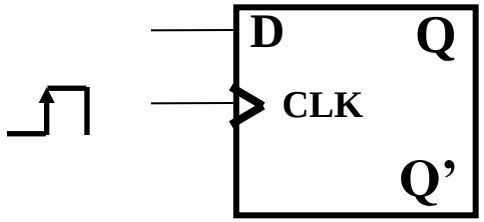
(a)



(b)

La bascule D

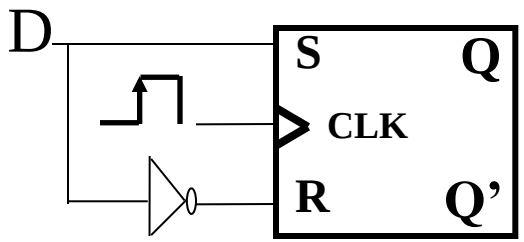
• Les sorties changent seulement au niveau de la transition d'horloge



Symbol

D	Q_{n+1}
0	0
1	1

D	CLK	Q	Q'
0	↑	0	1
1	↑	1	0
X	0	Q_0	Q_0'



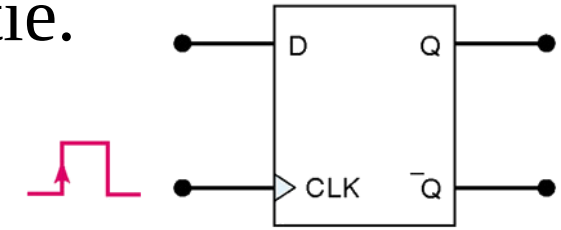
La bascule D peut être implémentée avec SR

S = D

R =D'

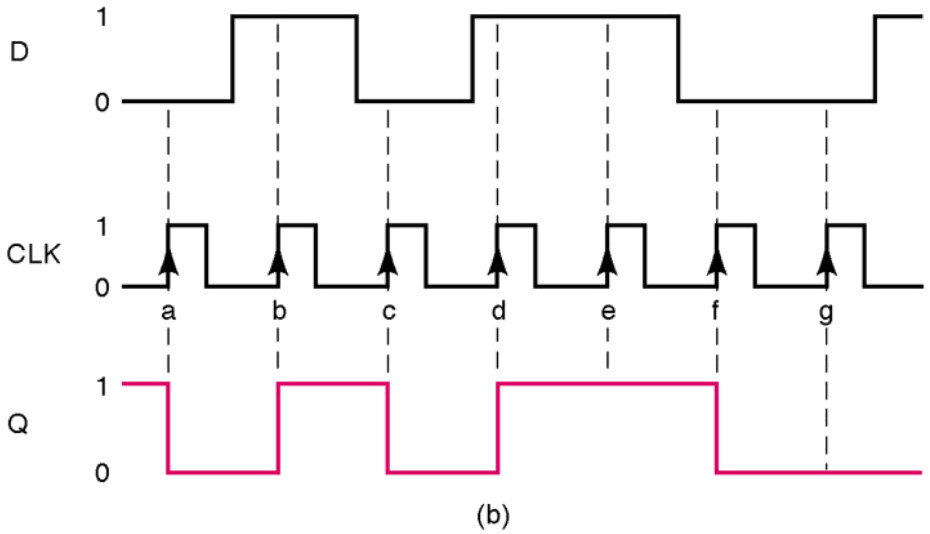
La Bascule D

- Stocke des valeurs lors de la transition front montant de l'horloge.
- Les changements hors front montant n'ont aucun effet sur la sortie.



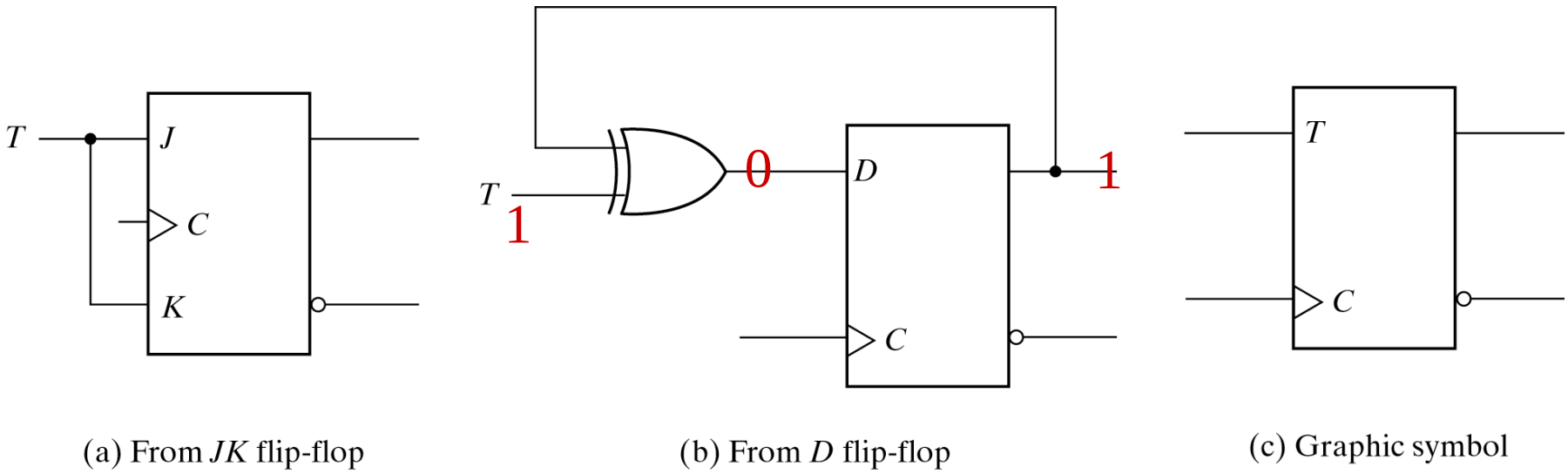
D	CLK	Q
0	↑	0
1	↑	1

(a)



(b)

La bascule T



(a) From *JK* flip-flop (b) From *D* flip-flop (c) Graphic symbol

Fig. 5-13 T Flip-Flop

°**Peut être implémentée avec JK ou D**

T	C	Q	Q'
0		Q_n	Q_n'
1			

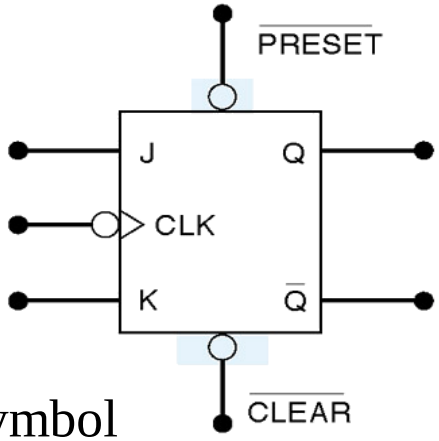
Complement

Les Entrées Asynchrones

- Jusqu'a présent nous avons considéré les entrées synchrones uniquement (e.g. D, S, R, J., K et T) → leur effet sur la sortie est synchronisée avec l'horloge.

- **Les bascules possèdent les entrées asynchrones. Elles opèrent indépendamment des entrées synchrones et de l'horloge.**

  - ***utilisés pour mettre la bascule a 1 or 0 a tout instant.***

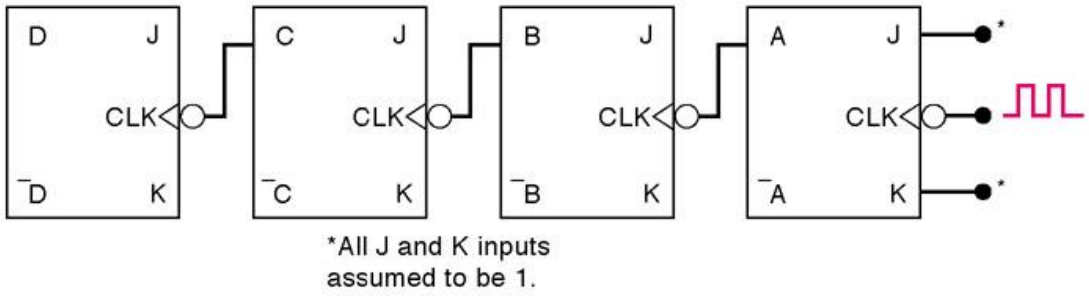


Symbol

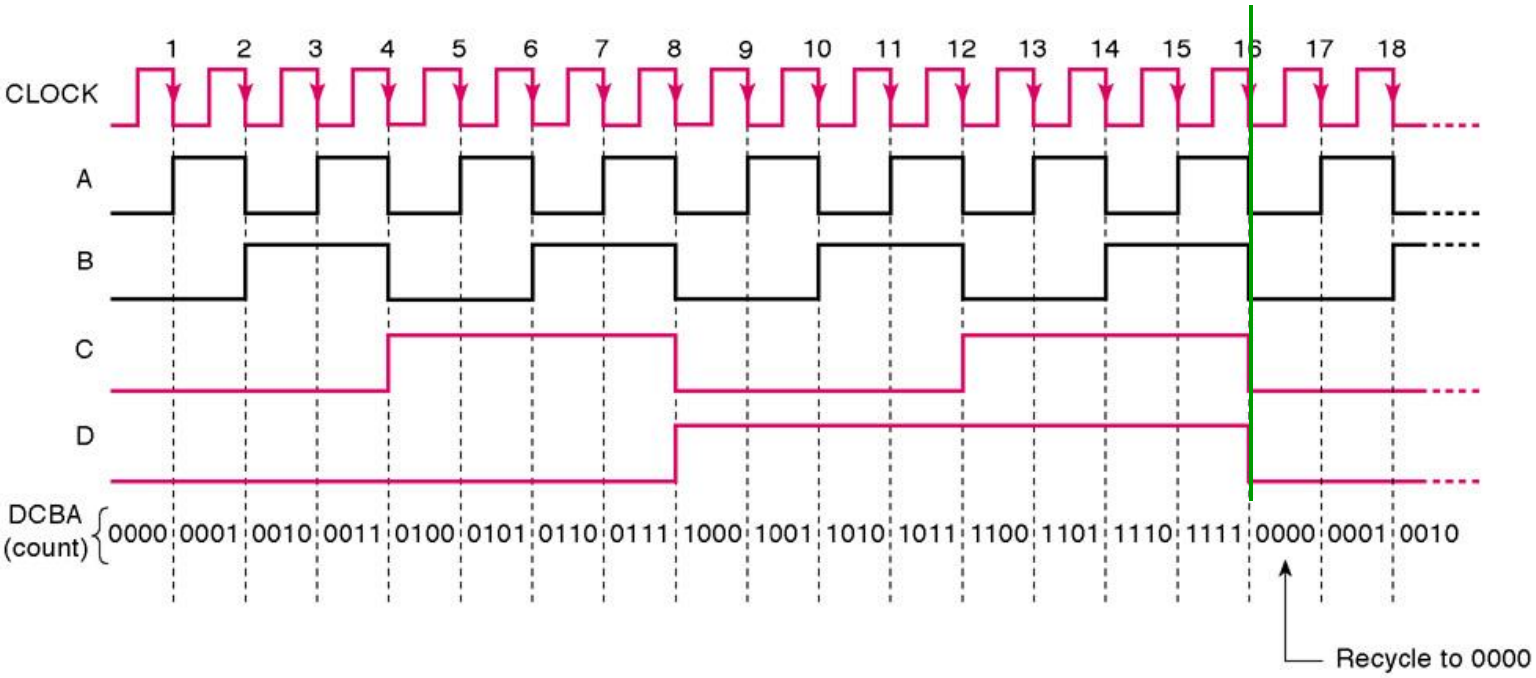
PRESET	CLEAR	FF response
1	1	Clocked operation*
0	1	Q = 1 (regardless of CLK)
1	0	Q = 0 (regardless of CLK)
0	0	Not used

*Q will respond to J, K, and CLK

Exemple d'utlisation des bascules



*All J and K inputs assumed to be 1.



Analyse des circuits séquentiels synchrone

- L'analyse d'un circuit décrit ce que fait le circuit sous certains conditions d'une opération

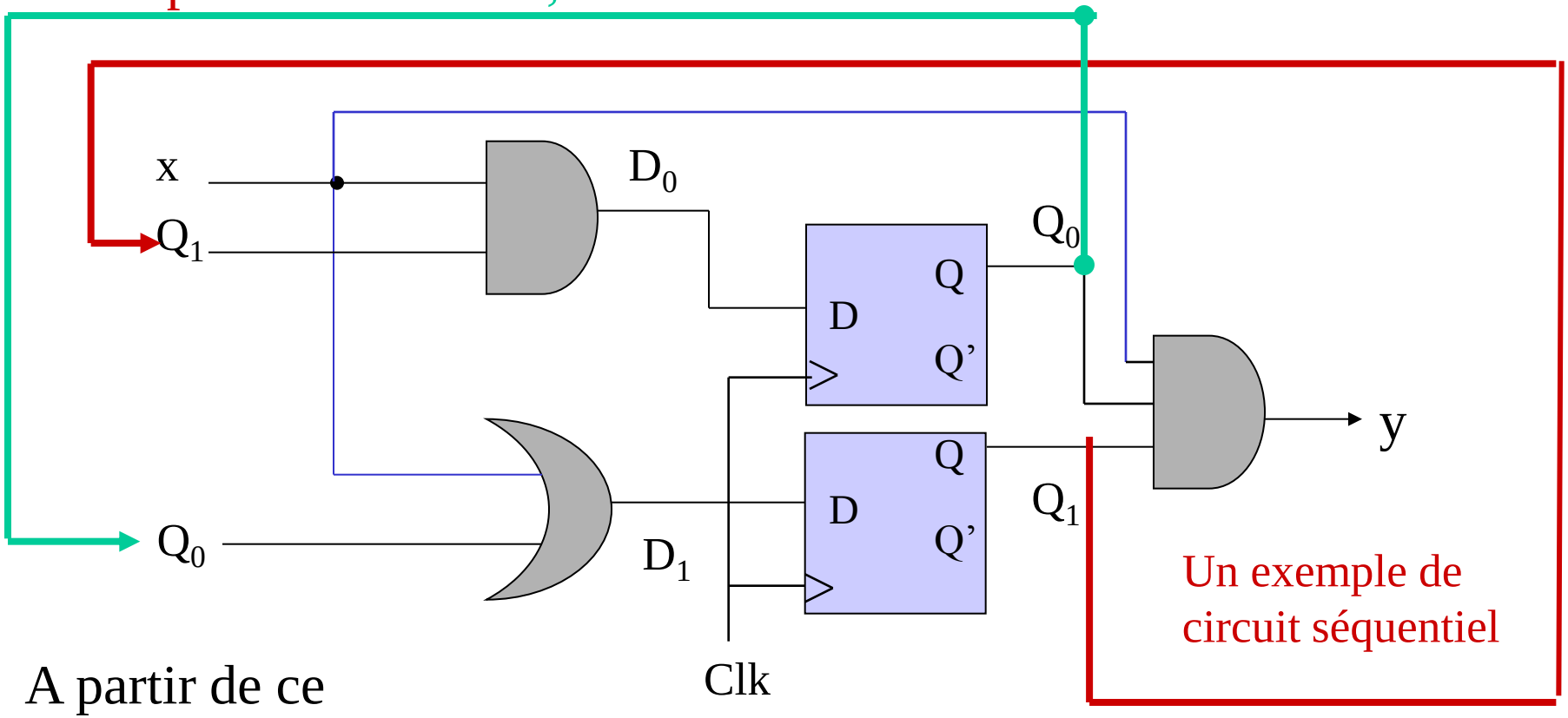
- Le comportement d'un circuit séquentiel synchrone est déterminé :

  - Les entrées,
  - Les sorties et
  - Les états

  des bascules

État dans les Bascules

° Le comportement du circuit séquentiel peut être déterminé a partir des entrées, des sorties et des états des bascules.



A partir de ce circuit on peut

Obtenir: :

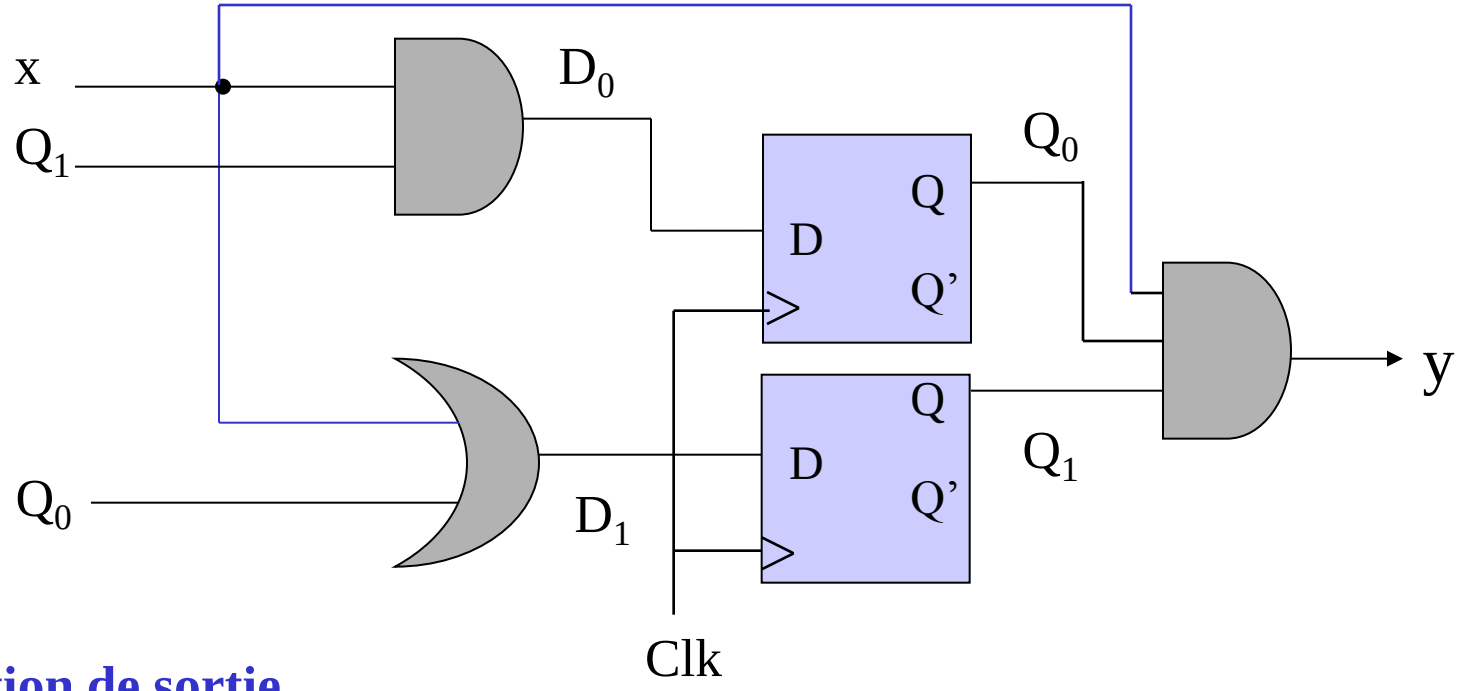
$y(t) = x(t)Q_1(t)Q_0(t)$

$Q_0(t+1) = D_0(t) = x(t)Q_1(t)$

$Q_1(t+1) = D_1(t) = x(t) + Q_0(t)$

Equations de Sorties et d'Eats

- **L'état suivant dépend (instant t+1) de l'état actuel (instant t).**



équation de sortie

$y(t) = x(t)Q_1(t)Q_0(t)$

équations d'État

$Q_0(t+1) = D_0(t) = x(t)Q_1(t)$

$Q_1(t+1) = D_1(t) = x(t) + Q_0(t)$

Tables des Etats

- Séquences des sorties, entrées, et états des bascules sont listées dans la « table des états »
- État Présent indique la valeur actuelle de la bascule
- État suivant indique la valeur des bascules après la prochaine impulsion d'horloge
- La sortie représente la valeur après l'impulsion actuelle d'horloge

Table des États

État présent	État suivant		Sortiet	
	x=0	x=1	x=0	x=1
0 0	00	10	0	0
0 1	10	10	0	0
1 0	00	11	0	0
1 1	10	11	0	1

$Q_1(t)$ $Q_0(t)$ $Q_1(t+1)$ $Q_0(t+1)$

Table des États

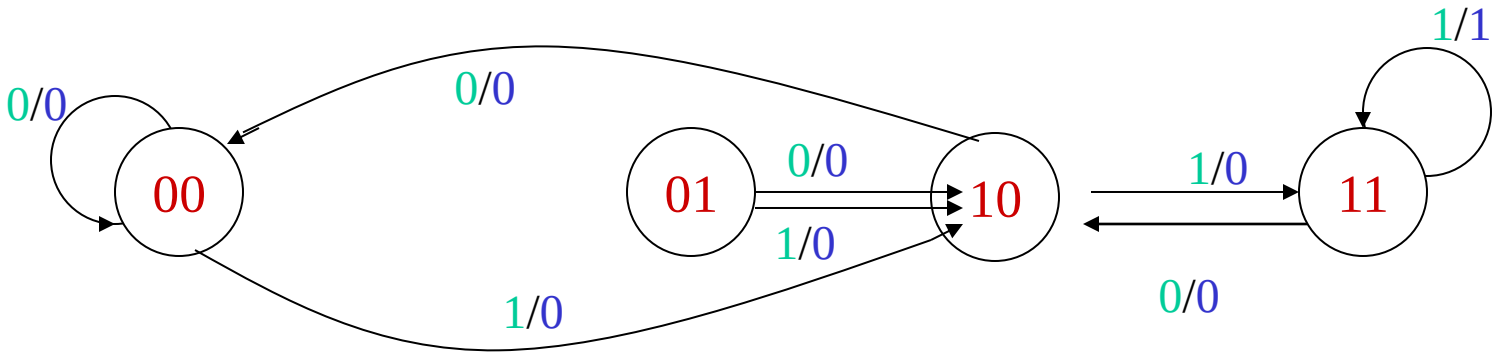
- Toutes les combinaisons en entrées sont listées
- Toutes les combinaisons des états sont représentées
- Une colonne séparée pour chaque valeur de sortie
- Plus facile d'utiliser des noms pour les états

soit:	État Présent	État suivant x=0	État suivant x=1	Sortie x=0	Sortie x=1
$s_0 = 00$	s_0	s_0	s_2	0	0
$s_1 = 01$	s_1	s_2	s_2	0	0
$s_2 = 10$	s_2	s_0	s_3	0	0
$s_3 = 11$	s_3	s_2	s_3	0	1

Diagramme des États

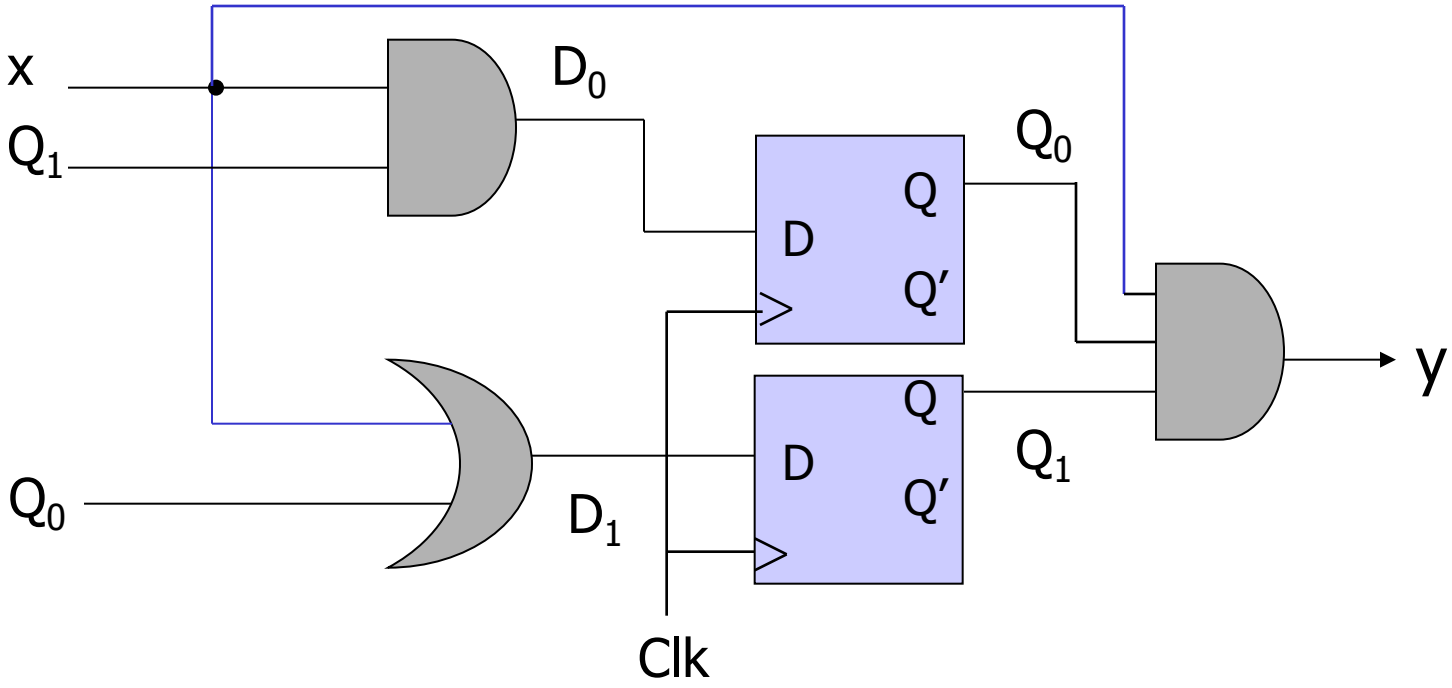
- Cercles indique l'État actuel
- Les flèches pointent vers l'État suivant
- Pour x/y, x l'entrée et y est la sortie

État Présent	État suivant		Sortie	
	x=0	x=1	x=0	x=1
0 0	00	10	0	0
0 1	10	10	0	0
1 0	00	11	0	0
1 1	10	11	0	1



Équations des Entrées des bascules

- **L'expression Booléenne qui indique l'entrée des bascules**

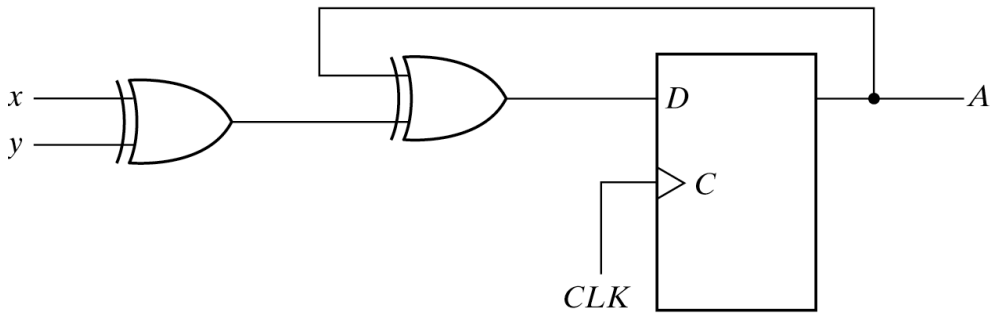


$D_{Q0} = xQ_1$

$D_{Q1} = x + Q_0$

Exemple avec bascule D

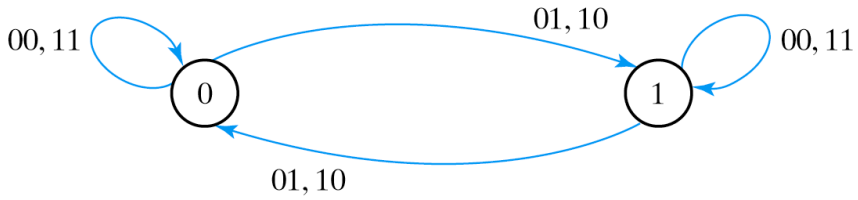
- **Identifier l'équations d'entrées des bascules** $D_A = A \oplus x \oplus y$
- **Identifier équation de sortie (pas de sortie dans cet exemple)**



(a) Circuit diagram

Present state	Inputs		Next state
A	*x*	*y*	*A*
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

(b) State table



(c) State diagram

Fig. 5-17 Sequential Circuit with *D* Flip-Flop

Représentation des États

- Equations des états
- Table des états
- Diagramme des états

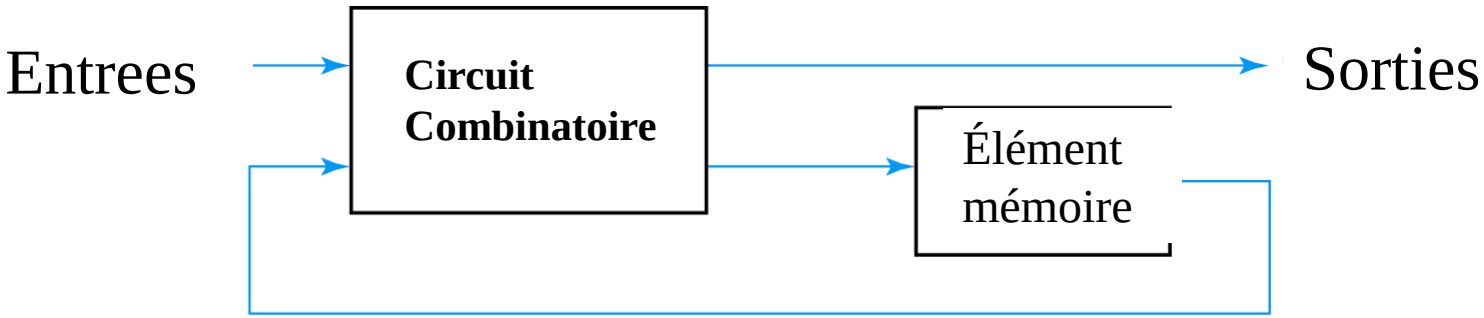


Fig. 5-1 Block Diagram of Sequential Circuit

Modèles de Mealy et de Moore

- le model général d'un circuit séquentiel possède

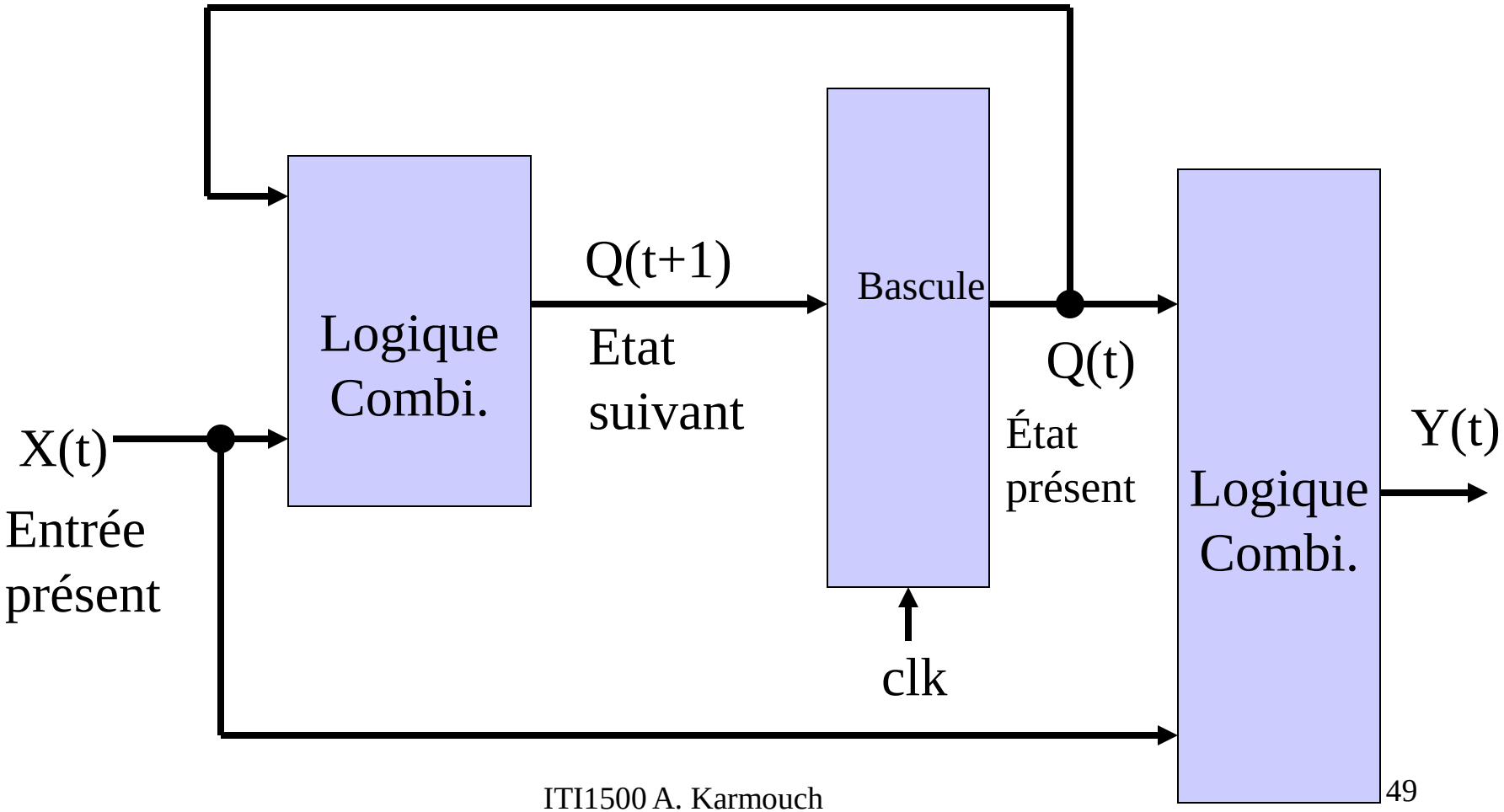
  - Des entrées
  - Des sorties
  - Des États internes

- Il y a deux modèles "standard "

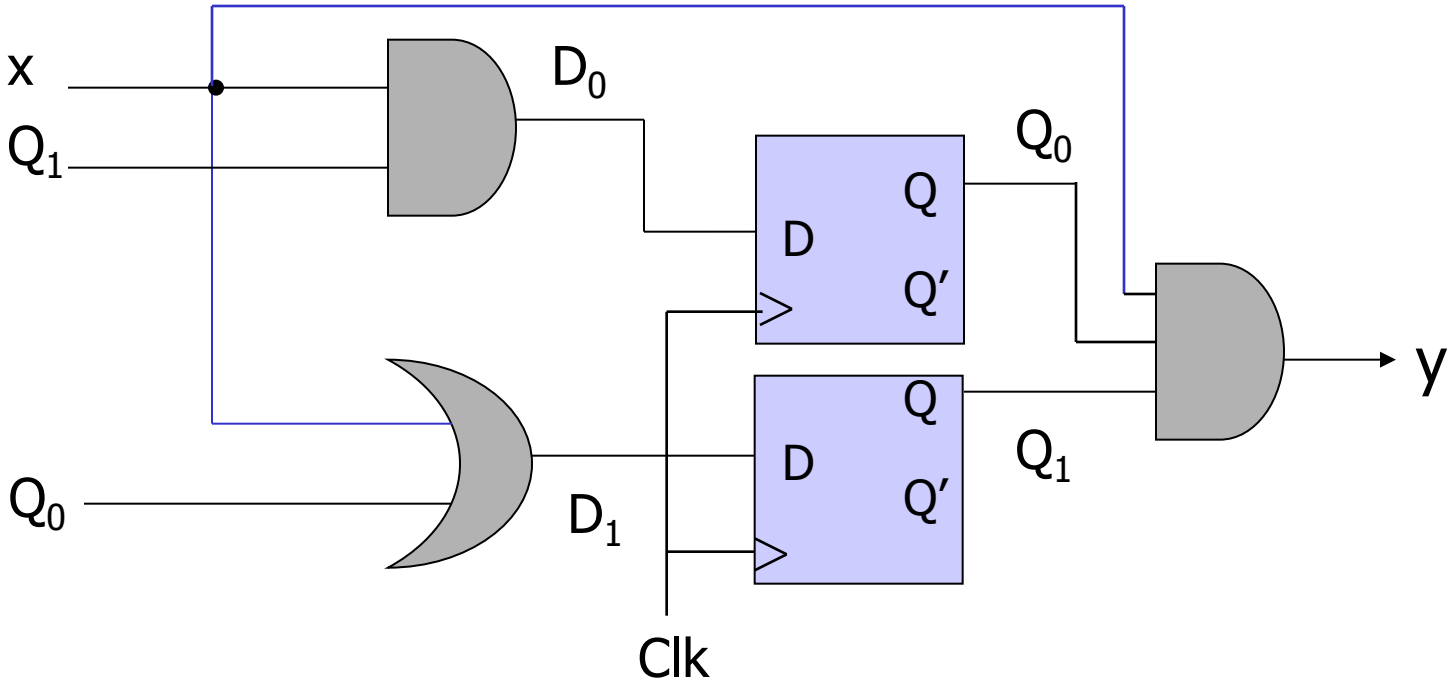
  - Le modèle de Mealy connu sous le terme de la machine d'état finis de Mealy (ou Mealy machine)
  - Modèle de Moore connu sous le terme de la machine d'état finis de Moore (ou Moore Machine)

Mealy Machine

• La sortie est fonction de l'état et de l'entrée actuelle (présent)



Exemples de Mealy Machine

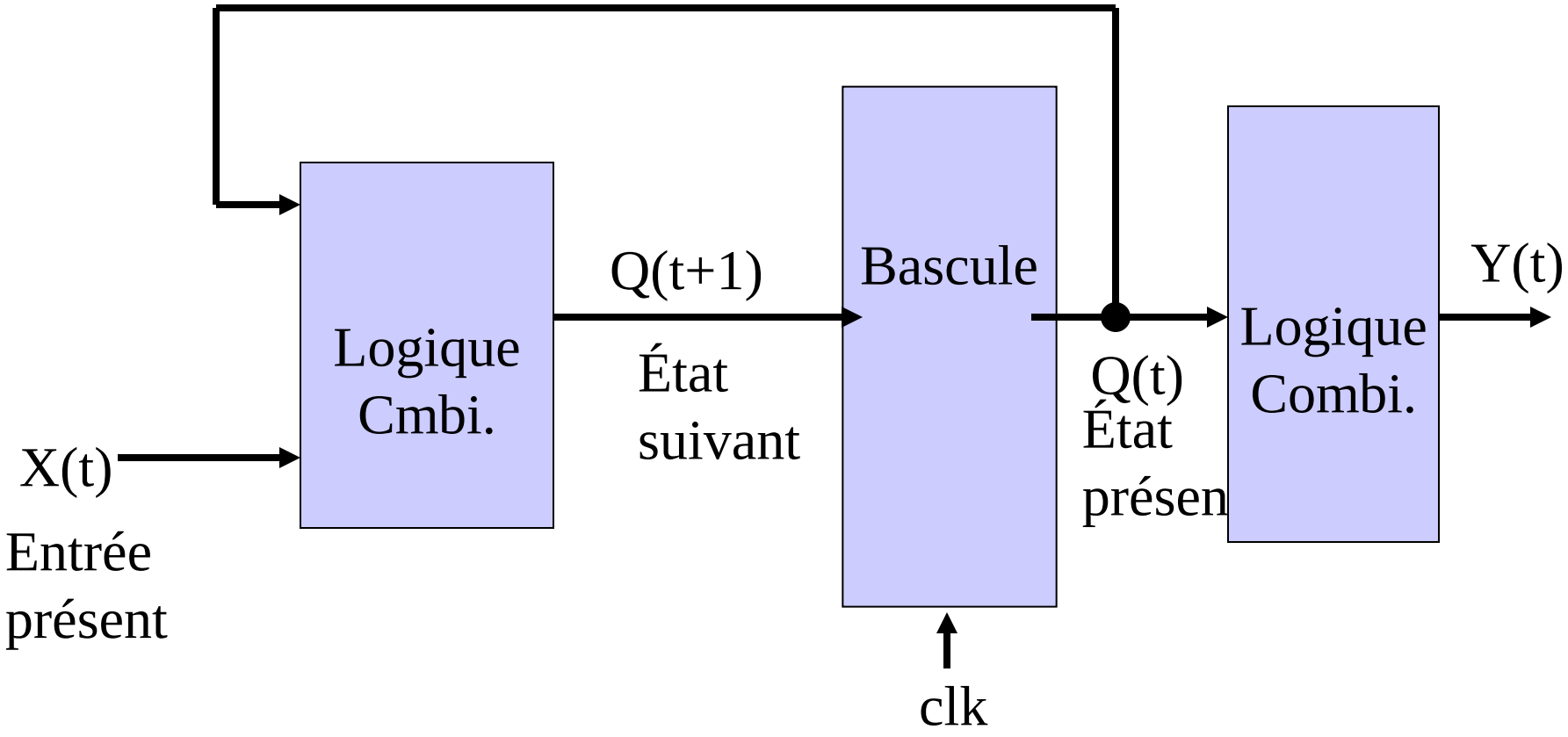


$D_{Q0} = xQ_1$

$D_{Q1} = x + Q_0$

Moore Machine

- Sortie est basée sur l'état présent seulement



Exemple de Moore Machine

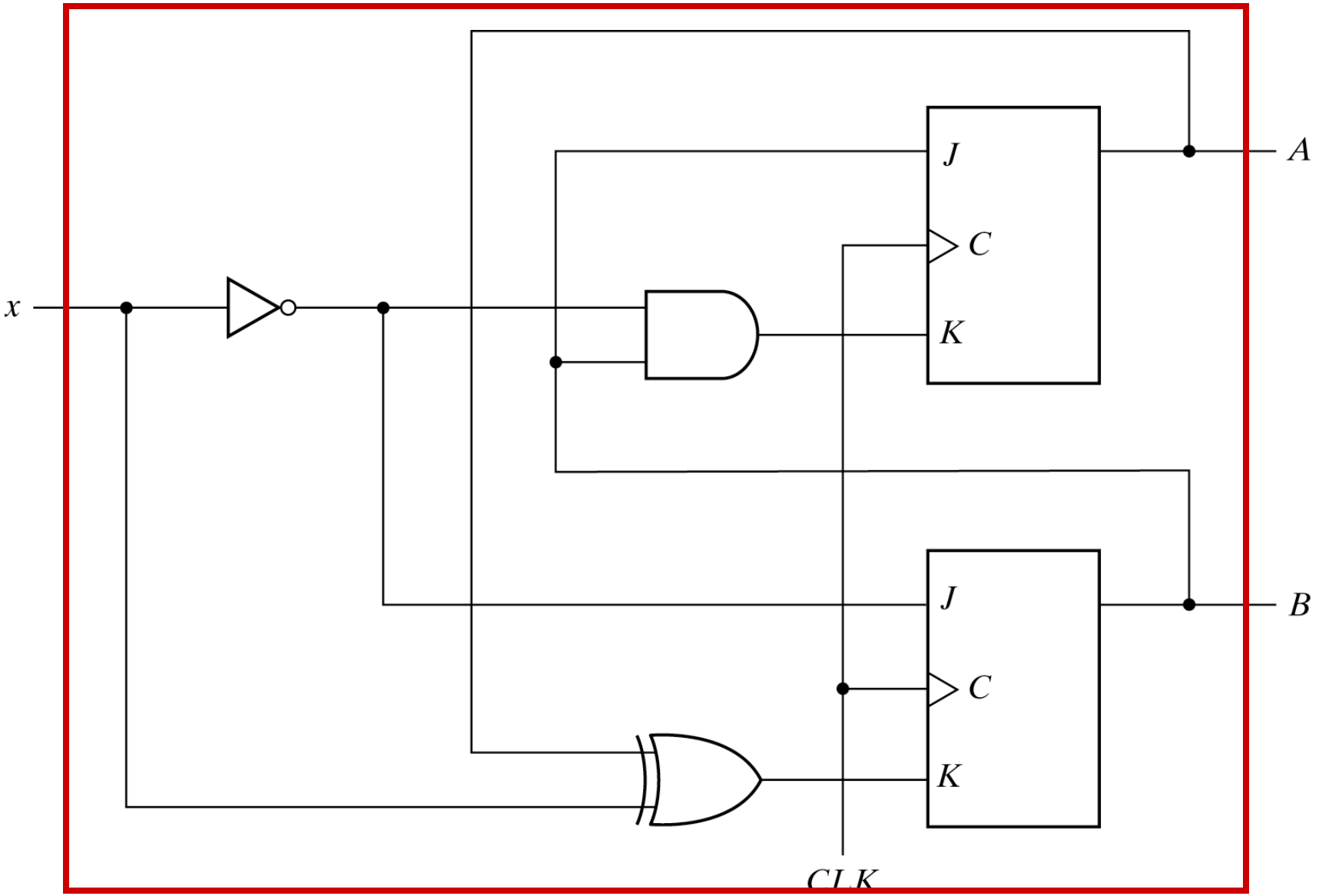
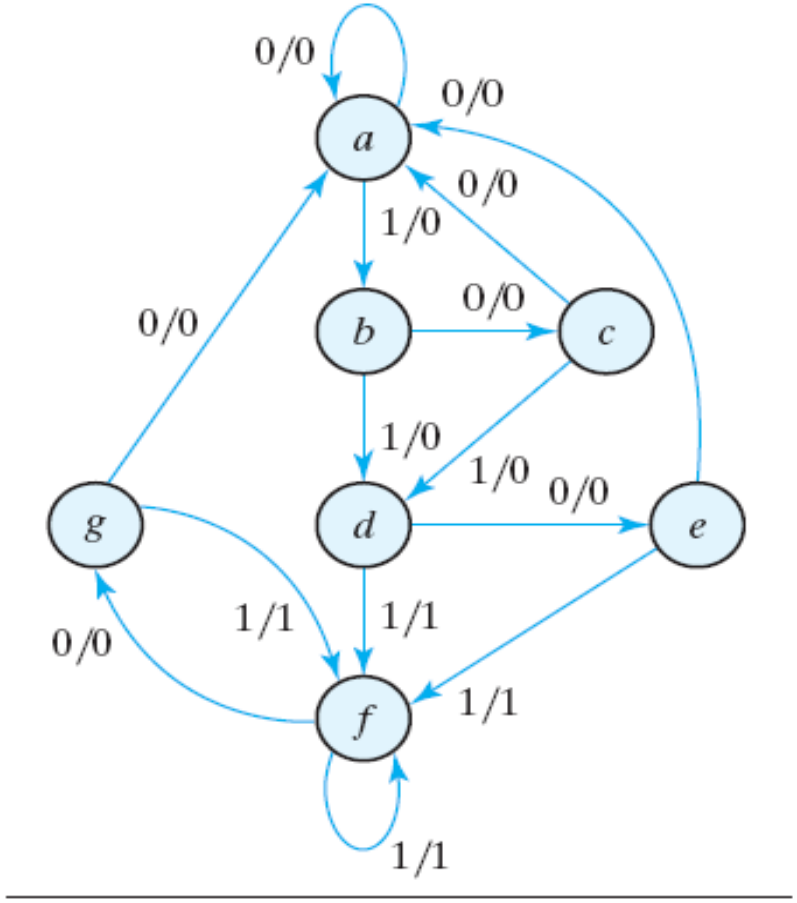


Fig. 5-18 Sequential Circuit with *JK* Flip-Flop

Réduction des Etats (1)

- Considérer un circuit séquentiel avec le diagramme d'état suivant



Réduction des états (2)

- Obtention de la table des états a partir du diagramme des états donne dans le transparent précédent

Etats e et f sont équivalents→ ont les états suivants et même sorties pour x=0, x=1

State Table

Present State	Next State $x=0$	Next State $x=1$	Output $x=0$	Output $x=1$
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	g	f	0	1
g	a	f	0	1

Reducing the State Table

Present State	Next State $x=0$	Next State $x=1$	Output $x=0$	Output $x=1$
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	e	f	0	1

retirer g et le remplacer par e dans les états suivant restants

Réduction des états (3)

Reducing the State Table

Present State	Next State $x=0$	Next State $x=1$	Output $x=0$	Output $x=1$
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	e	f	0	1

f et d sont équivalents

Reduced State Table

Present State	Next State $x=0$	Next State $x=1$	Output $x=0$	Output $x=1$
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	d	0	1
e	a	d	0	1

Fin du chapitre 5

Note: Plusieurs exemples sont discutés en classe mais ne figurent pas dans ce document