

# **Architecture des processeurs**

## **Soutien « Partie matérielle »**

### **CE312/CE318**

---

**Module « Systèmes matériels et logiciels »**

**Crédit : 2**

**Vincent Berouille**

Bureau : B206

[vincent.berouille@esisar.grenoble-inp.fr](mailto:vincent.berouille@esisar.grenoble-inp.fr)

# Informations préliminaires

---

- Volumes horaires : « Partie matérielle »
  - CE312 : 5 séances de cours de **soutien** (en parallèle du cours)
  - CE318 : (environ) 2 séances d'**harmonisation** (en début du cours)
- Divers documents sur Chamilo : rechercher les codes CE312 ou CE318

# I Introduction

Plan global de la partie soutien

---

## I Représentation des nombres

- Entier positif
- Bit vs Byte
- Entier négatif : complément à 2
- Virgule fixe

II Circuits combinatoires

III Circuits séquentiels

IV Machines à états finis

# I Représentation des nombres

## Codage binaire – entier positif

---

- Le code binaire pour les **entiers positifs** :
  - N bits =>  $2^N$  valeurs possibles => en partant de zéro et jusqu'à  $2^N - 1$ .
  - Exemple 8 bits =>  $2^8 = 256$  éléments de 0 à 255.

$$X = \sum_{n=0}^{N-1} x_n 2^n$$

- Exemple de décomposition dans la base 2 :

$$\begin{aligned} 01011010 &= 2^7 * 0 + 2^6 * 1 + 2^5 * 0 + 2^4 * 1 + 2^3 * 1 + 2^2 * 0 + 2^1 * 1 + 2^0 * 0 \\ &= 128 * 0 + 64 * 1 + 32 * 0 + 16 * 1 + 8 * 1 + 4 * 0 + 2 * 1 + 1 * 0 \\ &= 64 + 16 + 8 + 2 = 90 \end{aligned}$$

# I Représentation des nombres

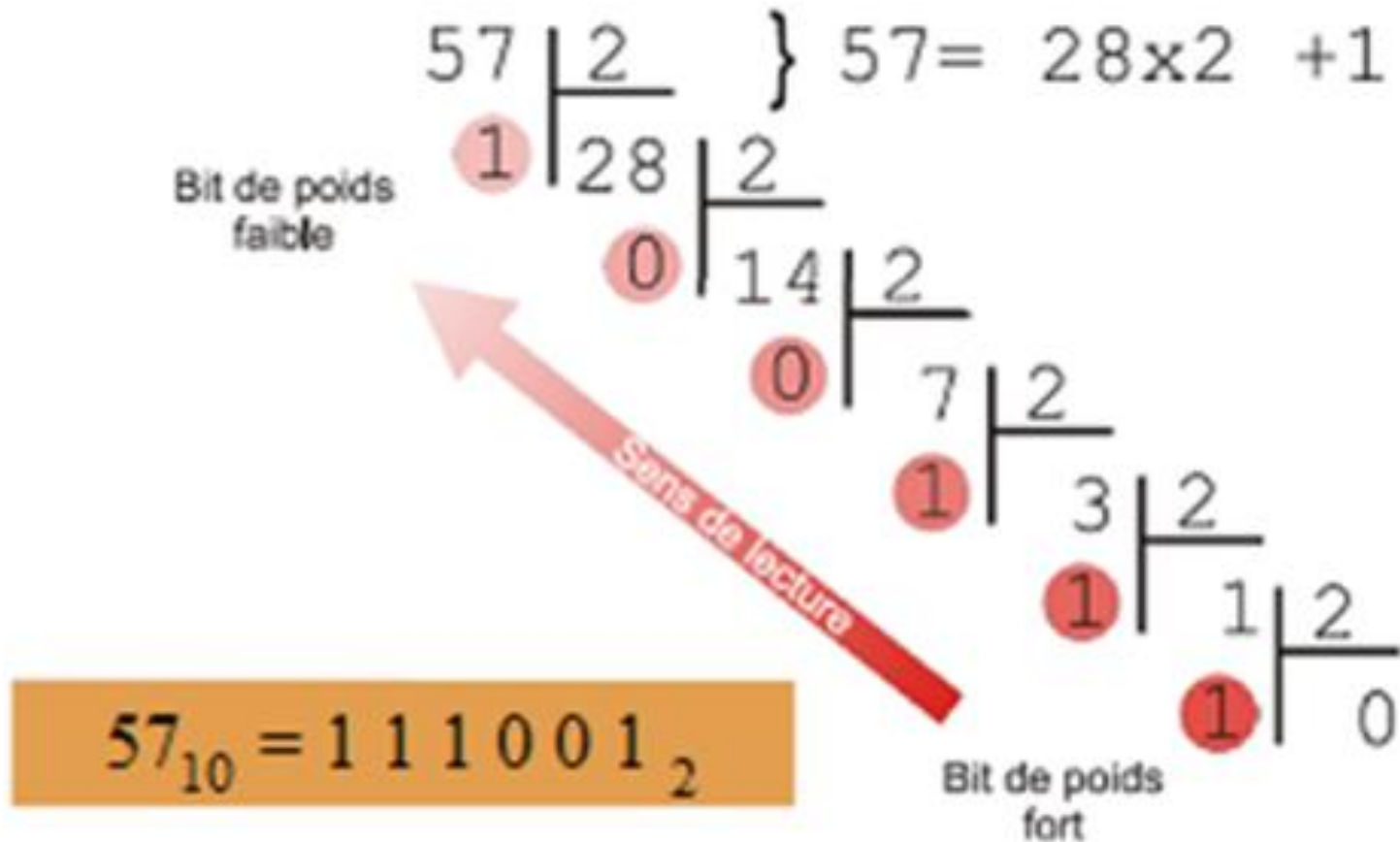
## Codage binaire – bit et byte

---

- Byte : mot de 8 bits
- “kB” for kilobyte vs. “kb” for kilobit
- On parle de kilo, Méga, Giga, Téra bits :
  - $2^{10} = 1024$  valeurs = 1 kibibit = 1 Kbit  $\approx$  1 kilobits = 1 kbit = 1000 bits
  - $2^{20} = 1048576$  = 1 mebibit = 1 Mibit  $\approx$  1 Mégabits = 1 Mbit =  $10^6$  bits

# I Représentation des nombres

Codage binaire – méthode itérative de conversion



# I Représentation des nombres

## Exercices

---

- Convertissez les nombres binaires suivants en leur équivalent décimal ou inversement.
  - 11001  $\Leftrightarrow$
  - 10010  $\Leftrightarrow$
  - 10011001101  $\Leftrightarrow$
  - $\Leftrightarrow$  23
  - $\Leftrightarrow$  42
- Quel nombre maximal peut-on atteindre avec 10 bits ? Combien de bits faut-il pour compter jusqu'à 511 ?

# I Représentation des nombres

## Codage binaire – entier négatif : **complément à 2**

---

- Le code binaire pour les entiers relatifs positifs **et négatifs** : **Code complément à 2**
  - Passage nombre entier positif vers son opposé :
    - **On complémente puis on ajoute 1**
  - Passage nombre entier négatif vers son opposé :
    - **On complémente et on ajoute 1**
  - Par exemple : on veut coder -5.
    - 5 se code par '0101'.
    - On obtient l'opposé en inversant les '0' et les '1' puis en ajoutant 1.
    - On obtient -5  $\Leftrightarrow$  '1011'



# I Représentation des nombres

## Codage binaire - Complément à 2

---

- Remarques :

- le MSB donne le signe.

- Avec N bits on code donc de  $\underbrace{-2^{N-1} \text{ à } 2^{N-1} - 1}_{\text{Inconvénient : échelle asymétrique}}$ .

$$X = \begin{cases} \sum_{n=0}^{N-2} x_n 2^n & X \geq 0 \\ -2^{N-1} + \sum_{n=0}^{N-2} x_n 2^n & X < 0. \end{cases}$$

Par exemple, « 1011 » vaut  $-2^3 + 2^1 + 2^0 = -8 + 2 + 1 = -5$

- **GROS AVANTAGE DU COMPLEMENT A 2 : la soustraction peut se faire en utilisant un additionneur classique**

# I Représentation des nombres

## Codage des nombres à virgule fixe

Partie entière    Partie fractionnaire

M bits , N bits

$$x = -2^m S + \sum_{i=-n}^{m-1} b_i 2^i$$

Exemples :

1,0 = 01,00

-1,0 = 11,00

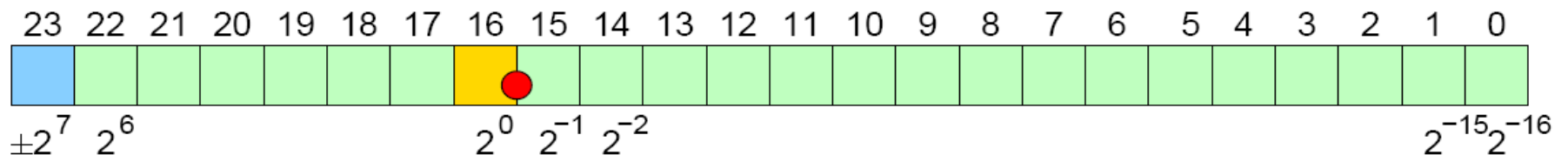
0,5 = 00,10

-0,5 = 11,10

Complément à  
2 de 1

Complément à  
2 de 0,5

réels 8Q16



# I Représentation des nombres

## Exercices

---

- Convertissez les nombres binaires en complément à 2 suivants en leur équivalent décimal ou inversement.
  - 11001  $\Leftrightarrow$
  - 10010  $\Leftrightarrow$
  - 10011001101  $\Leftrightarrow$
  - $\Leftrightarrow -23$
  - $\Leftrightarrow -42$
- Quel nombre maximal peut-on atteindre avec 10 bits ? Combien de bits faut-il pour compter jusqu'à -511 ?

# PLAN

---

I Représentation des nombres

## **II Circuits combinatoires**

- Définitions
- Portes logiques
- Transistor
- Circuits usuels
  - Décodeur, multiplexeur, démultiplexeur, additionneur/soustracteur, multiplieur

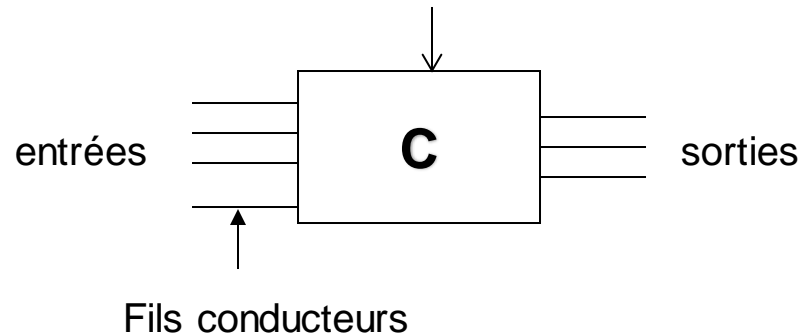
III Circuits séquentiels

IV Machines à états finis

# Définitions

- Circuit combinatoire **C**  $\Leftrightarrow$  Fonction Booléenne **F**  
                électronique                      mathématique
- Les sorties d'un circuit combinatoire peuvent être déterminées à partir de la connaissance de ses entrées

## Composants combinatoires (ou, et, inverseur...)



Vecteurs d'entrées  $\rightarrow$  vecteurs de sorties  
(léger décalage dans le temps, temps de propagation du signal)

# II Circuits combinatoires

## Portes logiques combinatoires

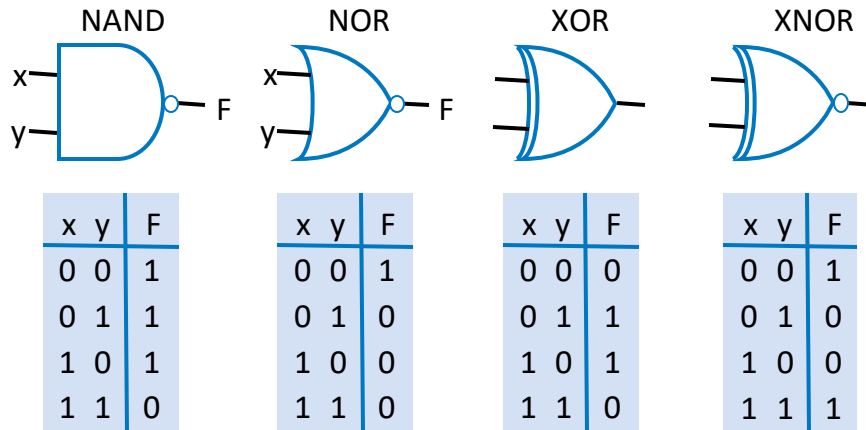
---

- Bibliothèque de portes de base :

INV

OR

AND



**Les portes logiques  
sont les éléments de  
base des circuits  
numériques**

# II Circuits combinatoires

## Portes logiques combinatoires - exercice

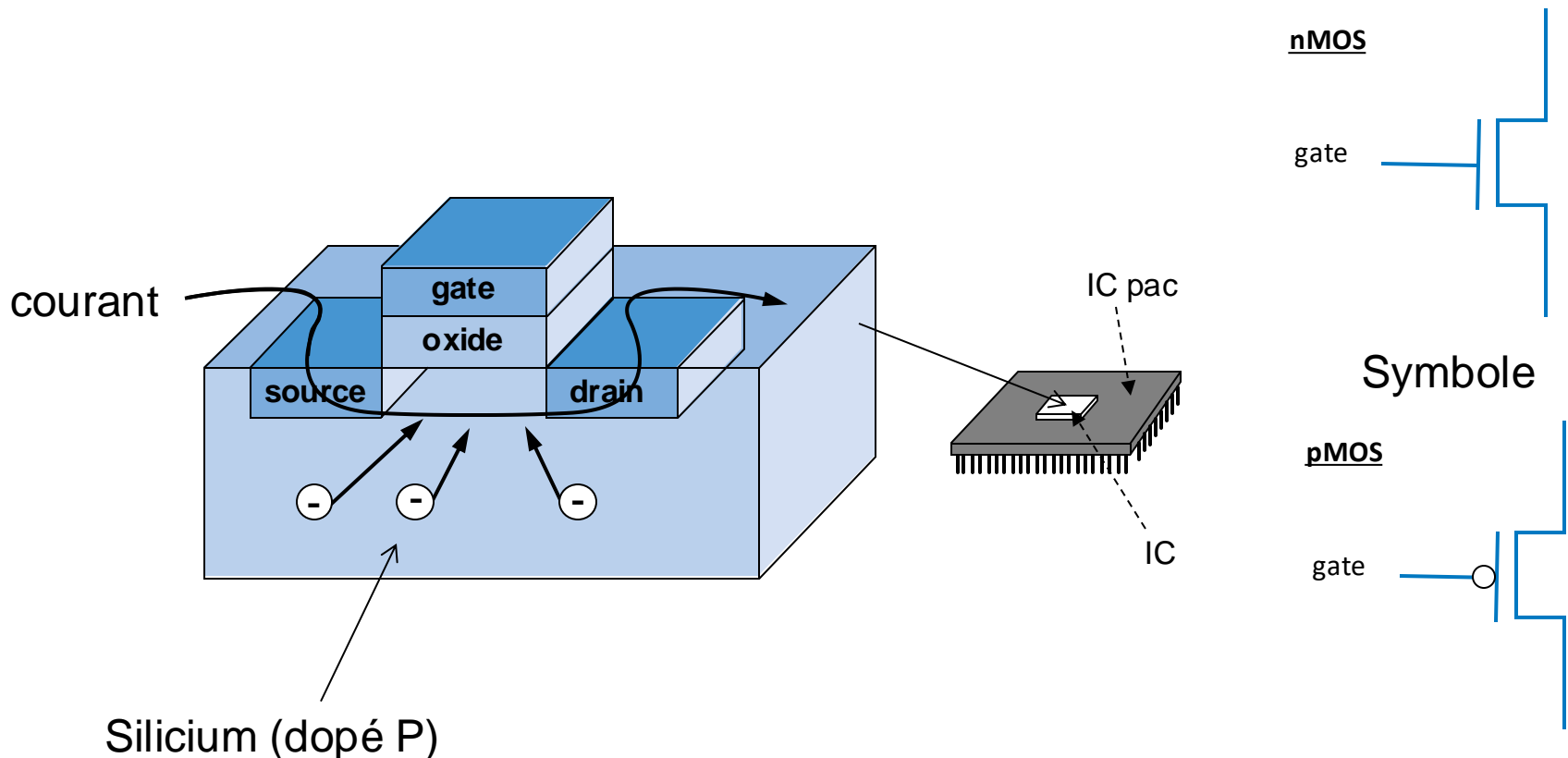
---

- Représentez l'équation suivante avec un circuit logique :
- $F = a \text{ AND NOT}( b \text{ OR NOT}(c) )$

# II Circuits combinatoires

## Portes logiques au niveau transistor

- Technologie CMOS : Complementary Metal Oxyde Semiconductor

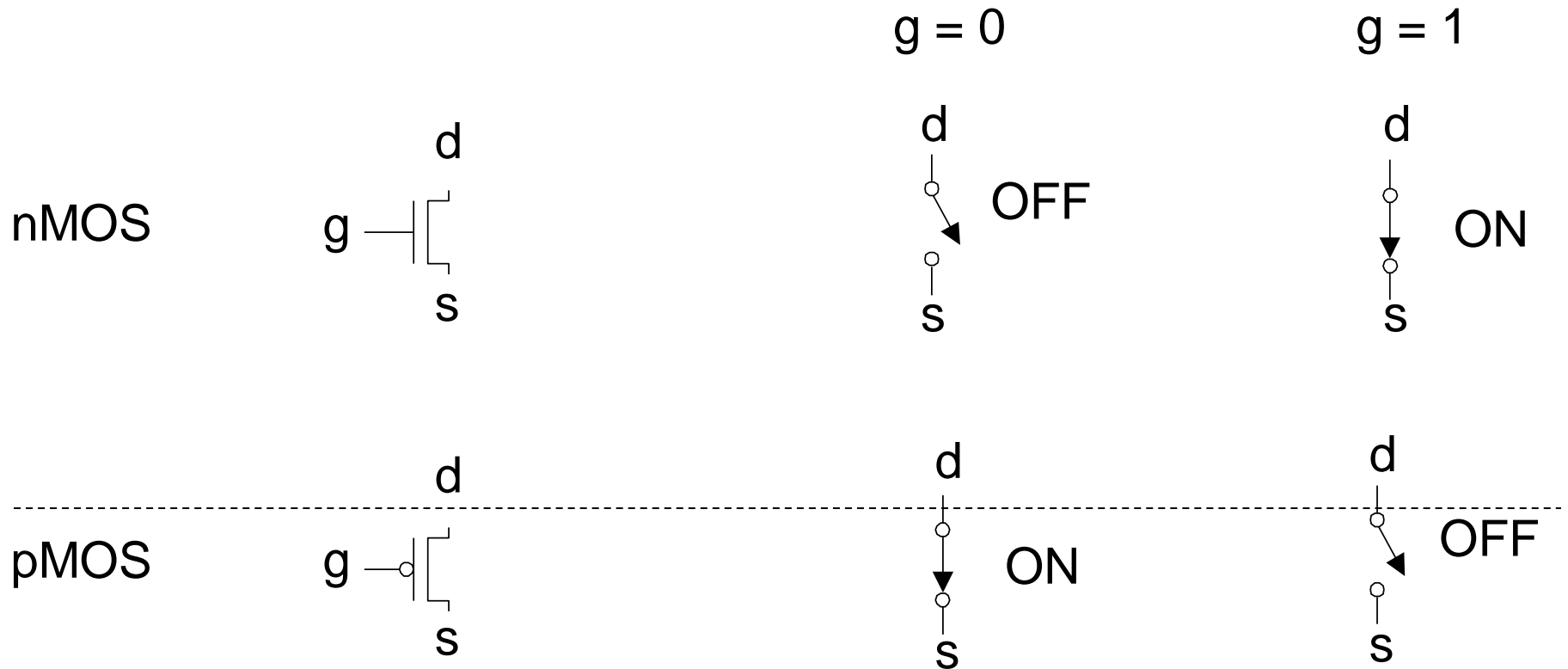




# II Circuits combinatoires

## Portes logiques au niveau transistor

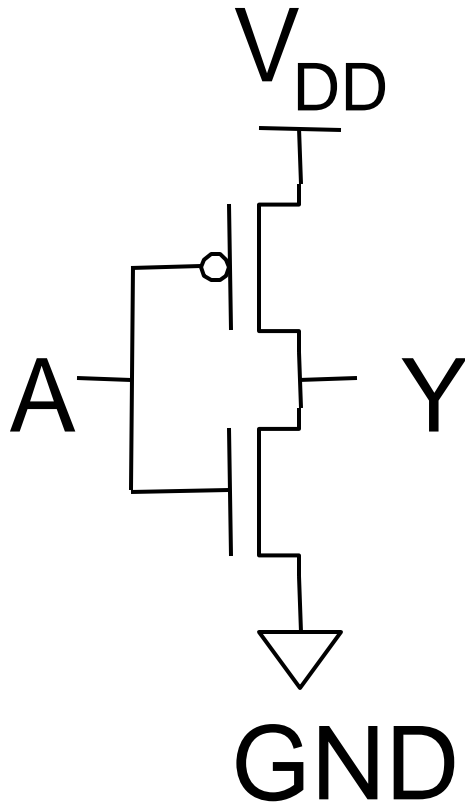
---



# II Circuits combinatoires

## Portes logiques au niveau transistor

---



Complétez la table de vérité

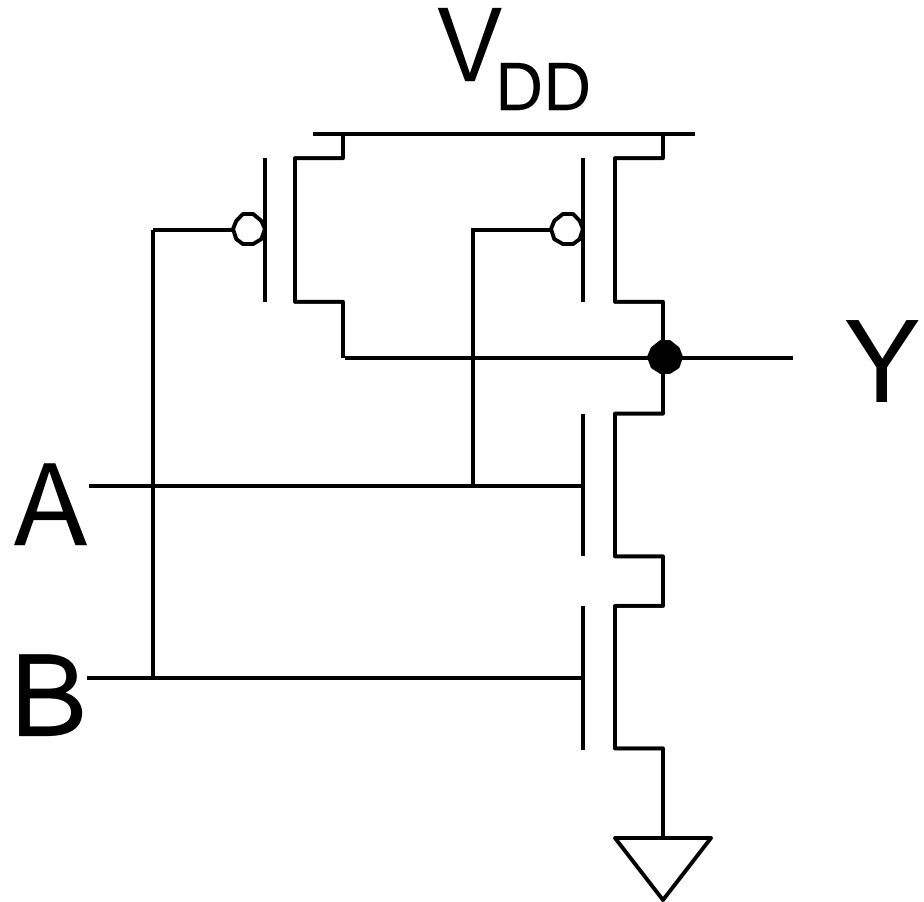
A	Y
0	
1	

# II Circuits combinatoires

## Portes logiques au niveau transistor

---

A	B	Y
0	0	
0	1	
1	0	
1	1	

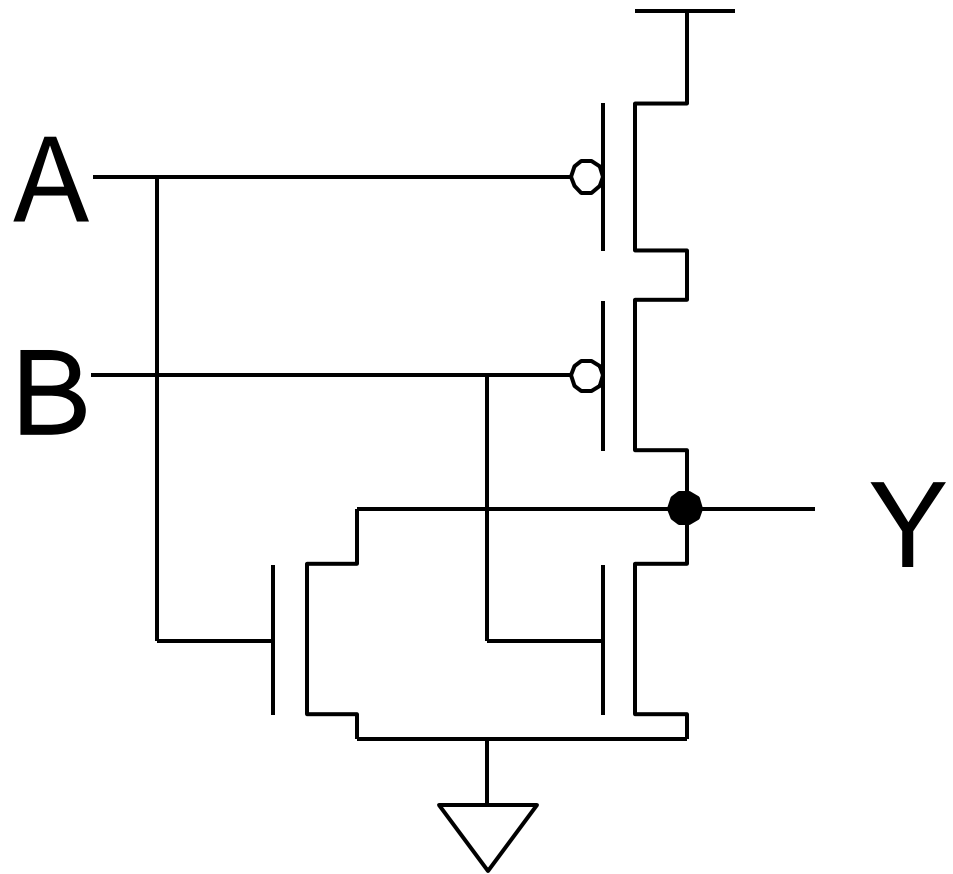


# II Circuits combinatoires

## Portes logiques au niveau transistor

---

A	B	Y
0	0	
0	1	
1	0	
1	1	



# II Circuits combinatoires


## Table de vérité

---

- ▶ Exemple de fonction combinatoire :  $F = A.B + C$

- ▶ Table de vérité :

et logique      ou logique



A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

# II Circuits combinatoires

## Exercice

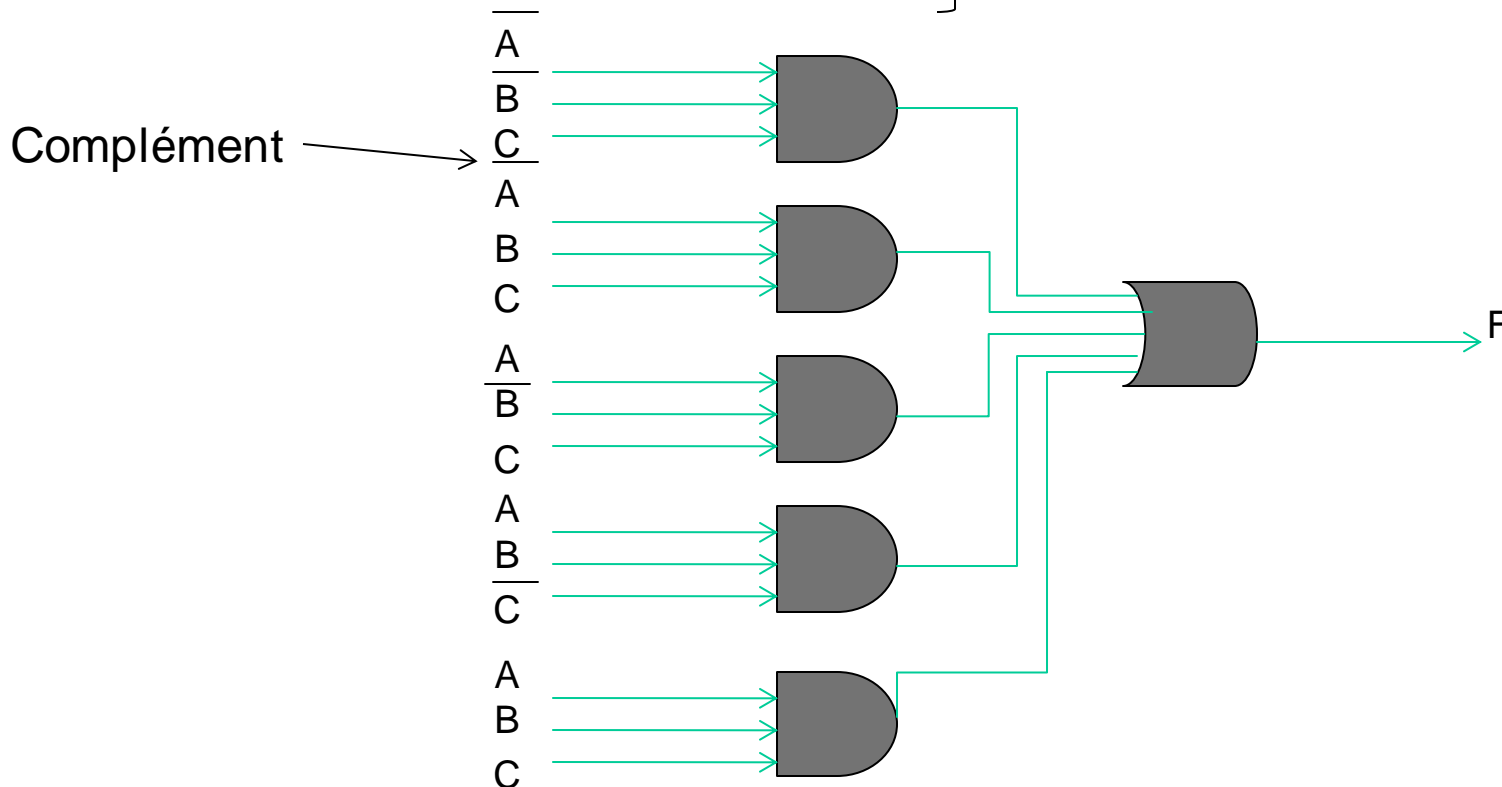
---

- Théorème de De Morgan :  $\overline{A + B} = \overline{A} \cdot \overline{B}$  et  $\overline{A \cdot B} = \overline{A} + \overline{B}$
  - En utilisant des tables de vérité montrez que les 2 égalités ci-dessus sont vraies
-

# II Circuits combinatoires

## Implémentation de $F = A.B + C$

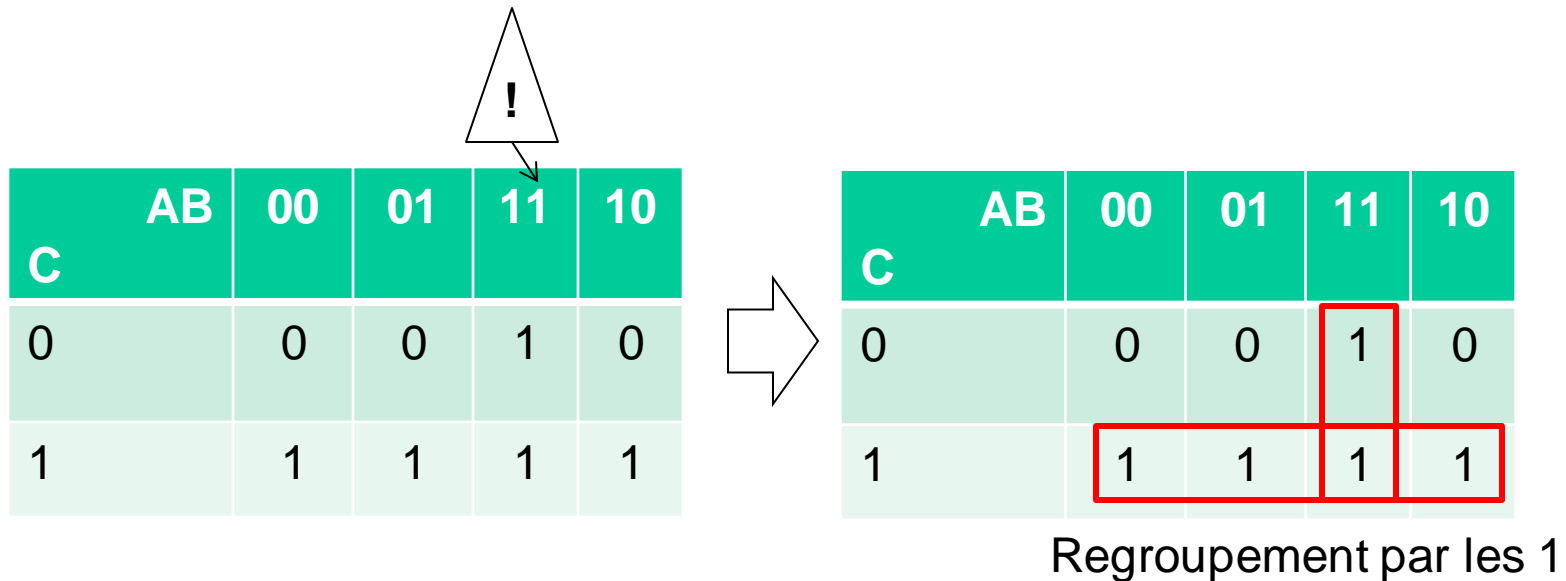
- ▶ On peut choisir d'implémenter toutes les possibilités en entrée d'avoir '1' en sortie
- ▶  $F = \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}C + ABC + ABC$  } 5 '1' en sortie dans la table de vérité  
=> 5 termes produits



# II Circuits combinatoires

## Tableau de Karnaugh

- On utilise les tableaux de Karnaugh pour simplifier l'implantation :



$$F = A.B + C$$



# II Circuits combinatoires

## Tableau de Karnaugh

---

Q		a b			
		0 0	0 1	1 1	1 0
c d	0 0	1	0	0	1
	0 1	1	1	1	1
	1 1	1	1	0	0
	1 0	0	0	0	0

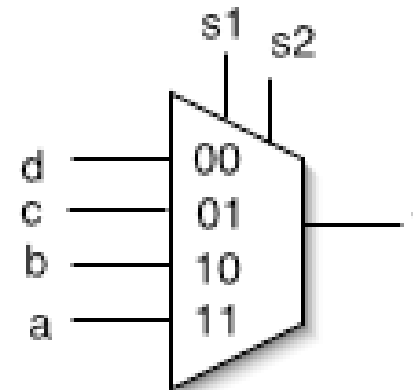
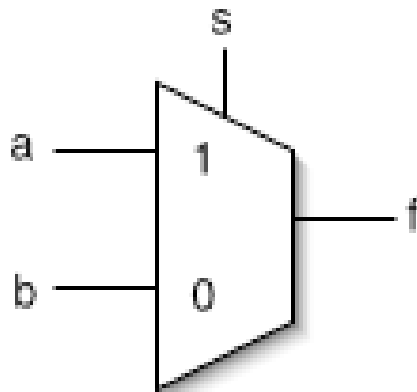
Donnez la fonction  $Q(a,b,c,d)$  simplifiée.

# II Circuits combinatoires

## Circuits usuels : multiplexeur

---

- Multiplexeur : dirige une des entrées vers la sortie
  - 2 vers 1 :  $f(a,b,s) = a.s + b.\bar{s}$
  - 4 vers 1 :  $f(a,b,c,d,s1,s2) = a.s1.s2 + b.s1.\bar{s}2 + c.\bar{s}1.s2 + d.\bar{s}1.\bar{s}2$



# II Circuits combinatoires

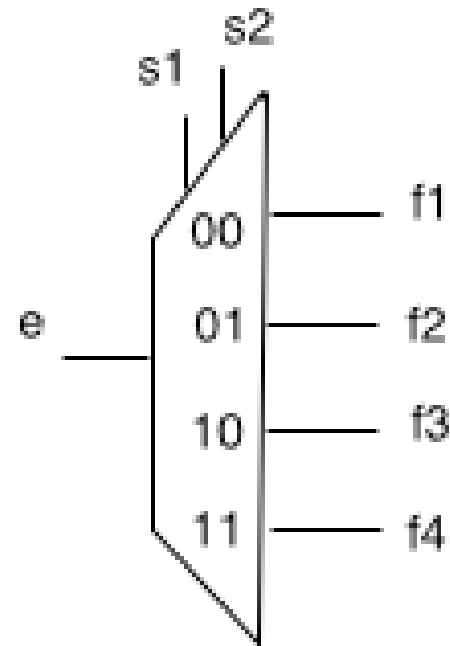
## Circuits usuels : démultiplexeur

---

- Démultiplexeur : dirige l'entrée vers une des sorties

– 1 vers 4 :

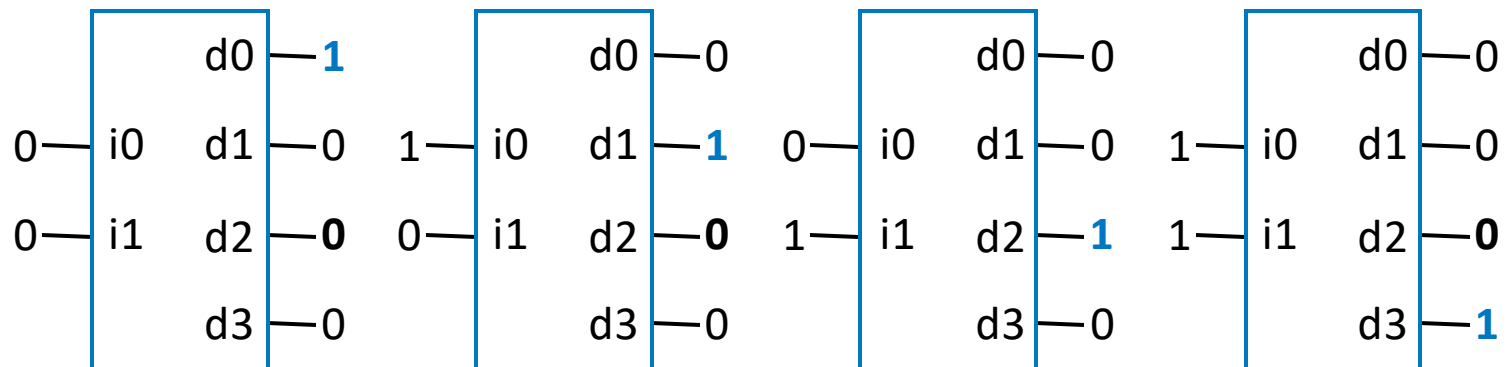
- $f1(e,s1,s2)=e.\overline{s1}.\overline{s2}$
- $f2(e,s1,s2)=e.\overline{s1}.s2$
- $f3(e,s1,s2)=e.s1.\overline{s2}$
- $f4(e,s1,s2)=e.s1.s2$



# II Circuits combinatoires

## Circuits usuels : décodeur

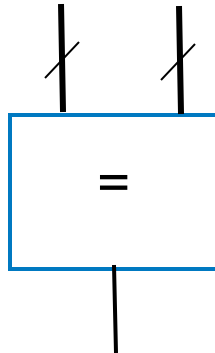
- Décodeur : convertit un nombre binaire en entrée vers une unique sortie haute



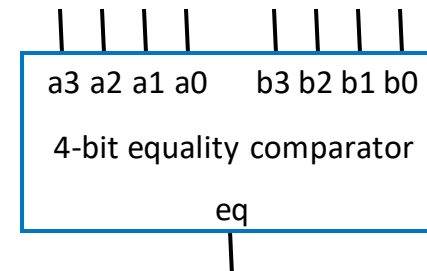
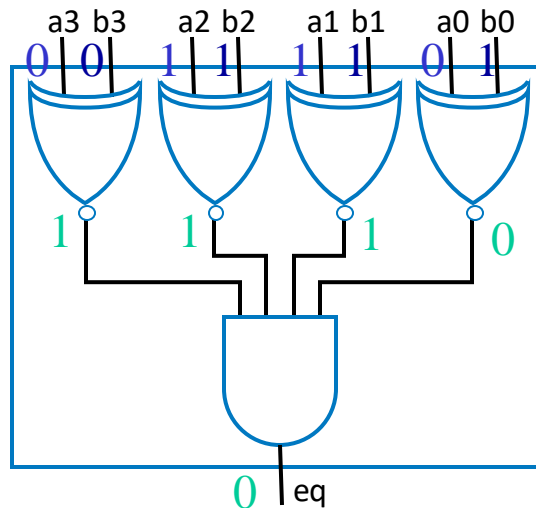
Donnez les équations de d0, d1, d2 et d3

# II Circuits combinatoires

## Circuits usuels : comparateur



0110 = 0111 ?



# II Circuits combinatoires

## Additionneur

---

$1+1 = 0$  avec retenue positive sur le bit suivant

$1+0 = 1$

$0+0 = 0$

- Exemple :  $13 + 5 = 1101 + 0101$   
 $= 10010$

$$\begin{array}{r} 1\ 1\ 1 \\ 1101 \\ +0101 \\ \hline 10010 \end{array}$$

L'addition de deux nombres de 4 bits  
donne un résultat sur 5 bits.

Le MSB est un **indicateur de retenue**.

# II Circuits combinatoires

## Soustracteur

- La soustraction **en complément à 2**

- Principe :  $A - B$  revient à faire  $A + (-B)$

- On code  $A$  et  $-B$  en complément à 2

- On additionne  $A$  et  $-B$

- Exemple :  $13 - 6$

- $13_{c2} = 01101$  on ajoute un bit de signe 0

- $-6_{c2} = /0110 + 0001 = 1001 + 0001 = 11010$

$$\begin{array}{r} 11 \\ 001101 \\ + 111010 \\ \hline 000111 \end{array} \begin{array}{l} \swarrow \text{5 bits + 5 bits} \\ \searrow \text{= 6 bits} \\ \rightarrow +7 \end{array}$$

Exemple :  $6 - 13$

$6_{c2} = 00110$

$-13_{c2} = /1101 + 0001 = 0010 + 0001 = 10011$

$$\begin{array}{r} /111001 + 000001 \\ 000110 \text{ } 000110 + 000001 \\ + 110011 \text{ } (000111)_2 = (7)_{10} \\ \hline 111001 \end{array} \rightarrow -7$$

31

# II Circuits combinatoires

## Exercices

---

- Additionnez les nombres suivants en binaire :
  1.  $25 + 5 =$
  2.  $42 + 13 =$
- puis en complément à deux.
  1.  $-25 + 5 =$
  2.  $-43 + 13 =$
  3.  $13 + (-13) =$



# II Circuits combinatoires

## Multiplieur (nombres entiers positifs)

---

				1	0	1	1
			x	0	1	0	1
				<hr/>			
Retenue de l'addition			1	1	0	1	1
			0	0	0	0	
		1	0	1	1		
	+	0	0	0	0		
		<hr/>					
		0	1	1	0	1	1
						1	1

Exemple de Multiplication Binaire

**Le produit de 2 nombres de n bits positifs génère 2n bits**

# II Circuits combinatoires

## Multiplication binaire (nombres entiers négatifs)

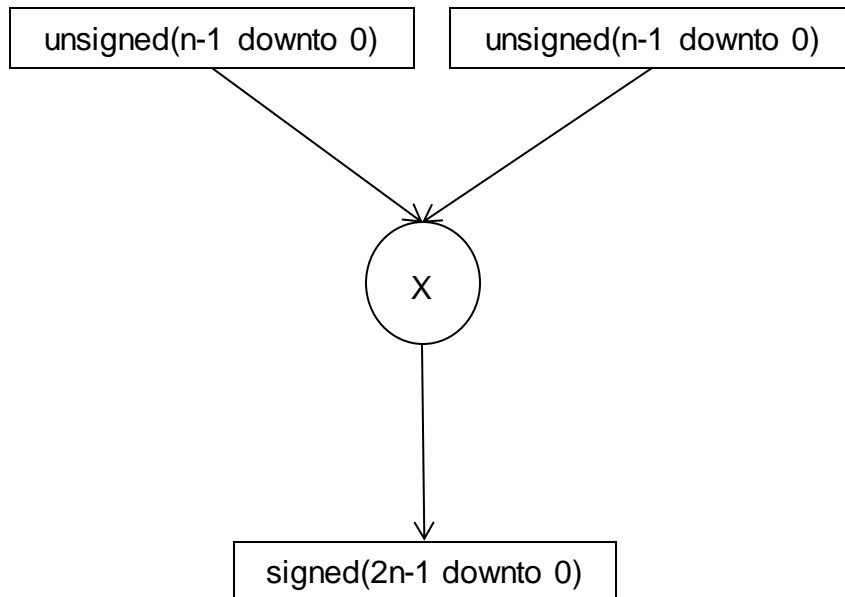
---

- Le produit de 2 nombres négatifs avec un multiplieur classique ne donne pas le bon résultat
- Il faut convertir les nombres négatifs en nombre positifs avant de les multiplier puis appliquer le signe

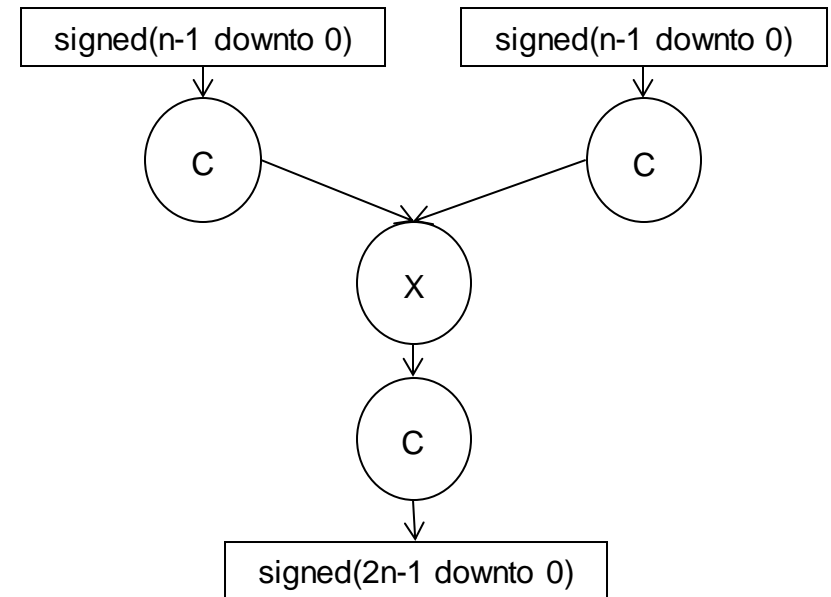
# II Circuits combinatoires

## Multiplieur (nombres négatifs)

### Nombres entiers positifs



### Nombres entiers négatifs



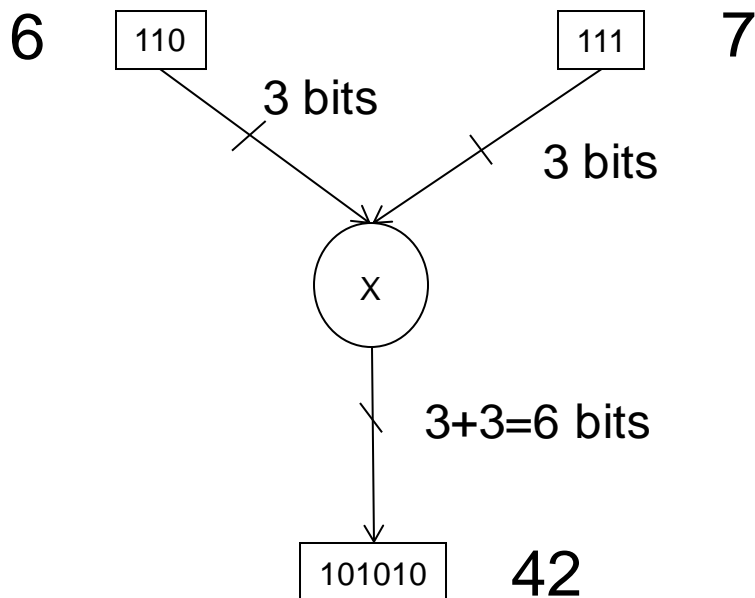
**La « nature non signée ou signée des nombres » a une importance sur la structure matérielle du multiplieur utilisé**

# II Circuits combinatoires

## Multiplieur (nombres entiers négatifs)

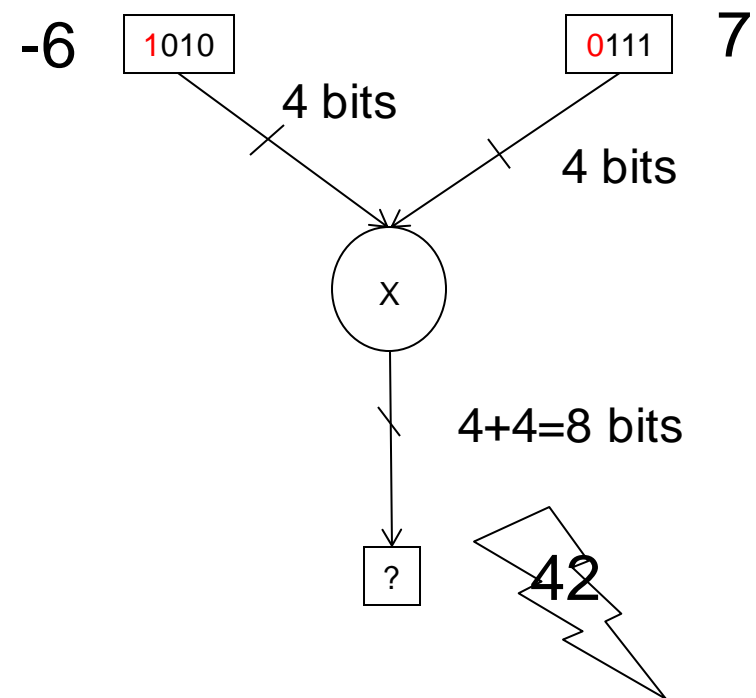
Exemple :  $6 \times 7$

Nombres entiers positifs



Exemple :  $-6 \times 7$

Nombres entiers négatifs  
(complément à 2)

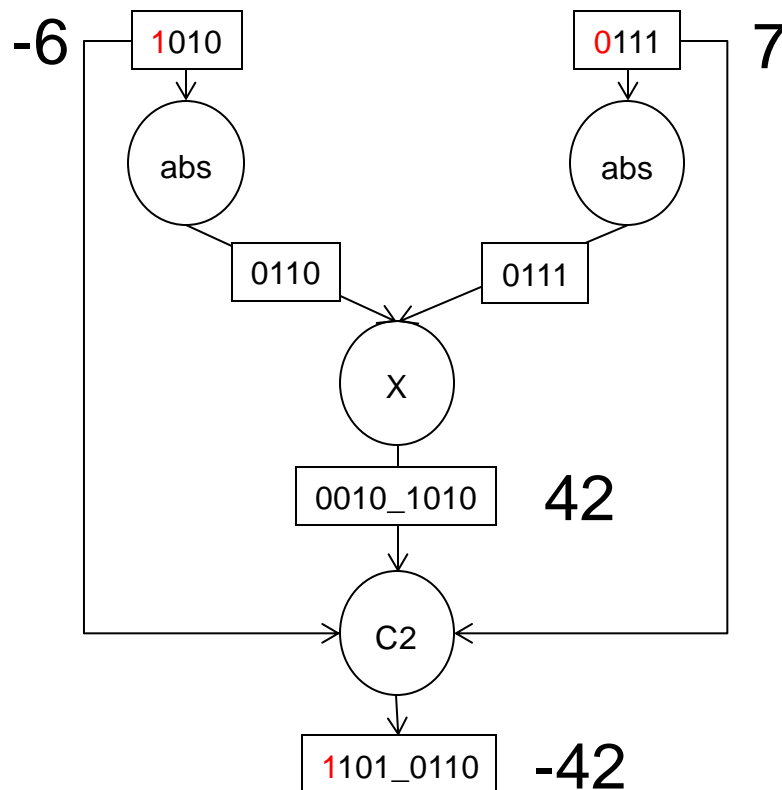


# II Circuits combinatoires

## Multiplieur (nombres entiers négatifs)

Exemple :  $-6 \times 7$

Nombres entiers négatifs  
(complément à 2)



# II Circuits combinatoires

## Exercices

---

- Multipliez les nombres suivants
  - $12 * 3 =$
  - $5 * -6 =$

# PLAN

---

I Représentation des nombres

II Circuits combinatoires

**III Circuits séquentiels**

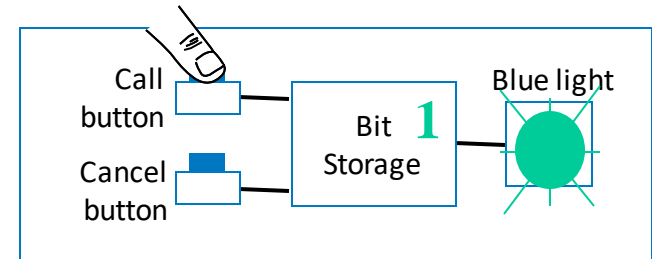
- Définitions
- Bascule verrou (D latch)
- Bascule DFF
- Registres
  - Registre séquentiel, à chargement parallèle
- Compteurs
- Mémoires
  - RAM (SRAM/DRAM), ROM (PROM, EPROM, EEPROM)

IV Machines à états finis

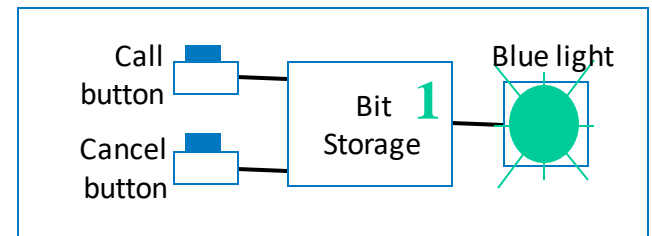
# III Circuits séquentiels

## Définitions

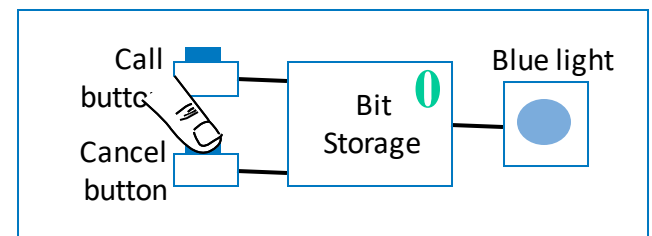
- Circuit séquentiel :
  - Les valeurs des sorties dépendent des valeurs d'entrée **et** de la séquence de valeurs d'entrée depuis le reset.
  - C'est-à-dire  $S = F(E, \text{séquence des entrées})$ .
  - On a donc une **mémorisation** et une **initialisation du circuit**.



1. Call button pressed – light turns on



2. Call button released – light stays on



3. Cancel button pressed – light turns off

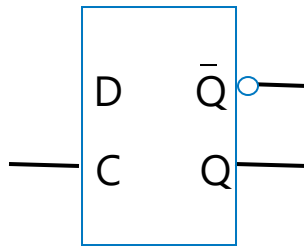


# III Circuits séquentiels

## Élément mémorisant : Bascule verrou (ou D latch)

---

- Fonctionnement (vue de l'extérieur)



**D latch  
symbol**

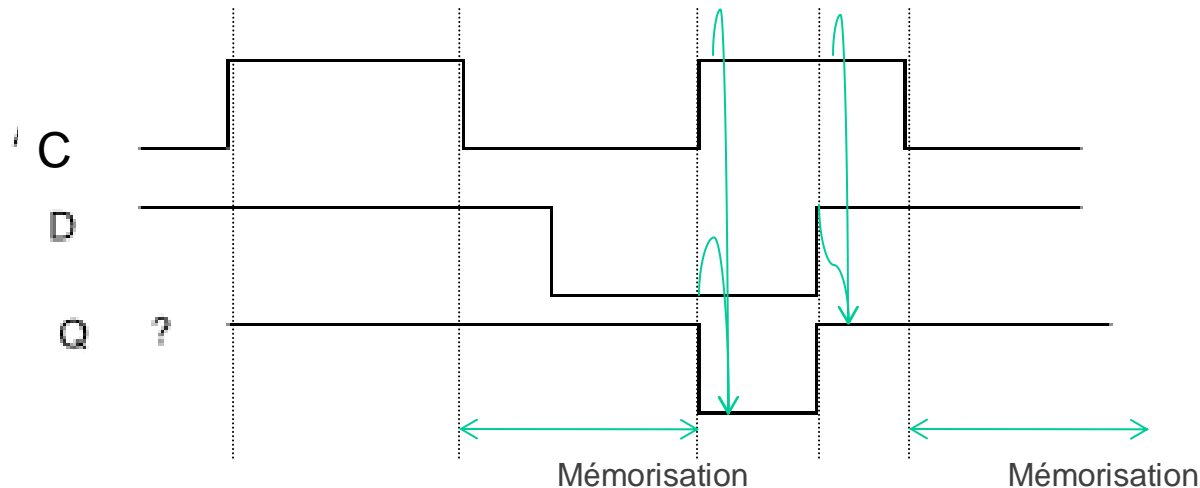
C	D	Q
Oui	0	0
Oui	1	1
Non	*	Q à l'instant précédent

- Sensible au niveau de l'activation (appelé **LATCH** ou **VERROU**)

# III Circuits séquentiels

## Élément mémorisant : Bascule verrou

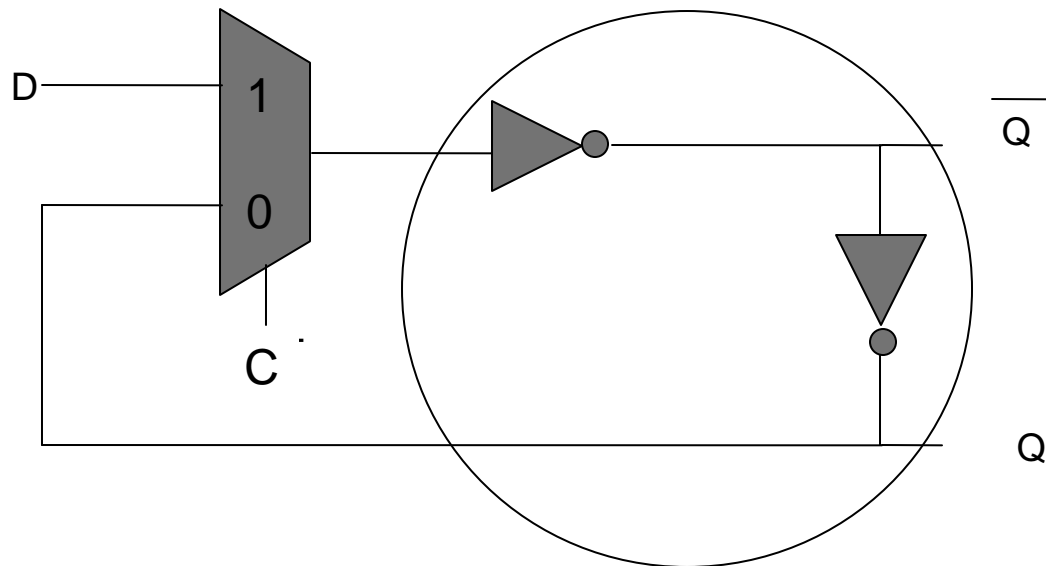
- Exemple de chronogramme sensible au niveau haut de C :



- Problème :
  - Si D change quand C passe de 1 à 0, Q est indéterminé.

## Élément mémorisant : Bascule verrou

► Principe :

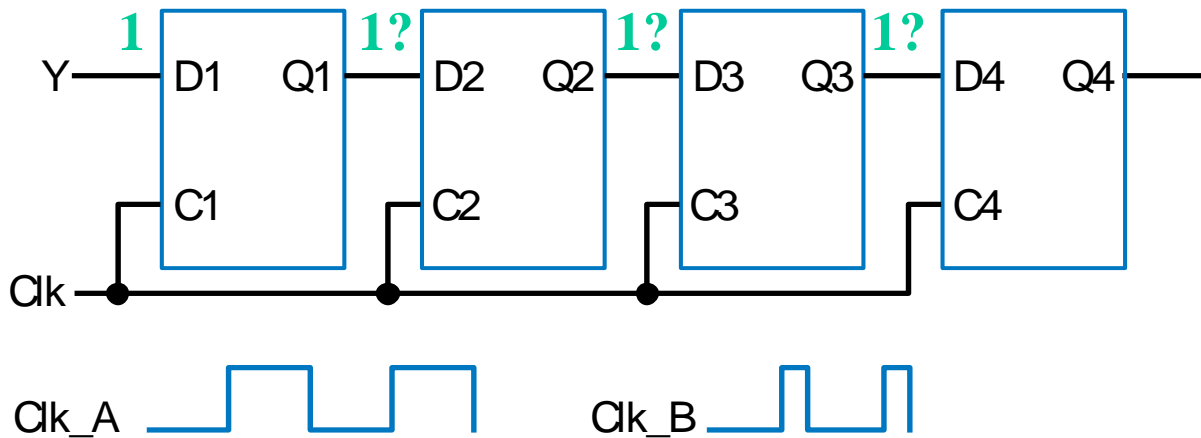


## Elément mémoire

# III Circuits séquentiels

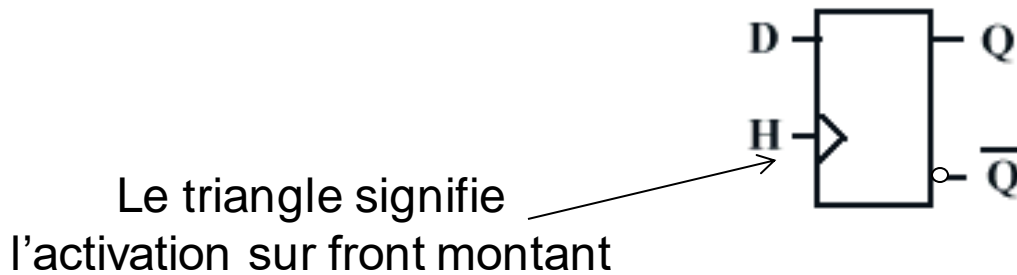
## Élément mémorisant : Bascule verrou

- Les bascules verrous sont “transparentes”
  - Quand  $C=1$ , au travers de combien de bascules le signal va-t-il se propager?
  - Cela dépend de la longueur de l'impulsion  $C=1$ !
- Clk\_A – le signal traverse plusieurs bascules
- Clk\_B – le signal traverse moins de bascules

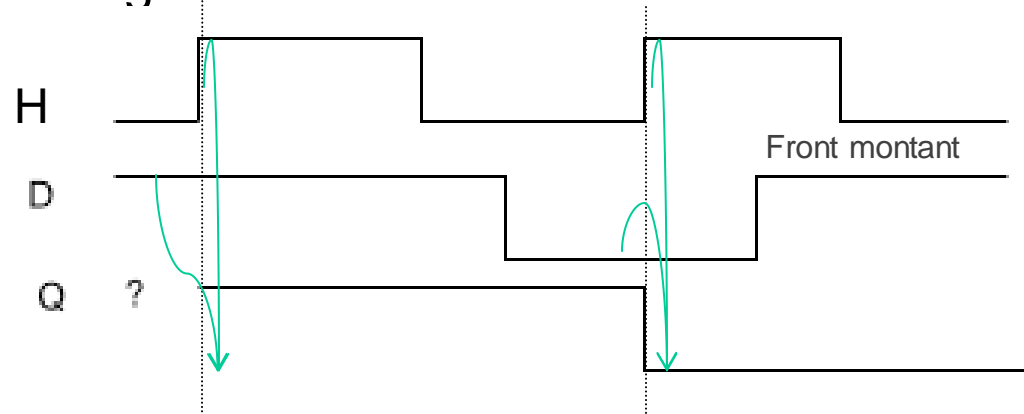


# III Circuits séquentiels

## Élément mémorisant : Bascule D Flip Flop (DFF)



- Exemple de chronogramme sensible au front montant de H :

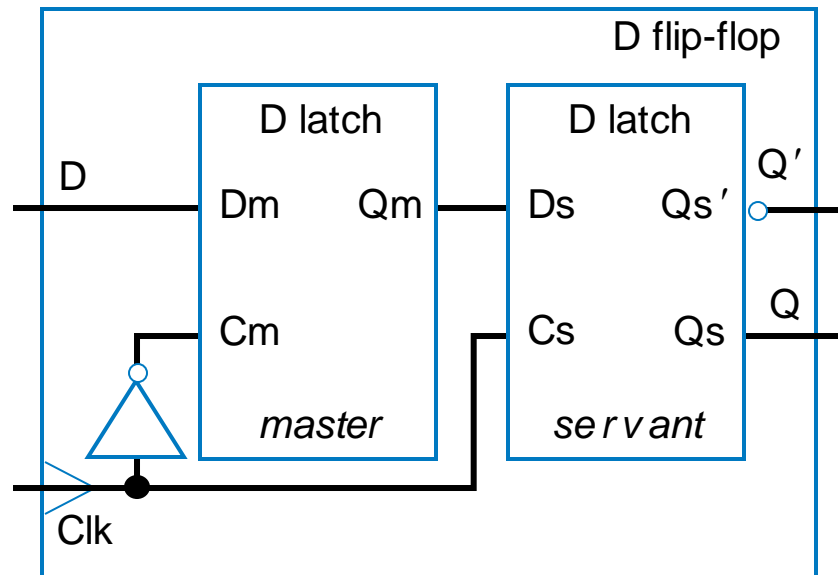


- Problème :
  - Si D change quand H présente un front montant, Q est indéterminé
  - Intervalle de commutation interdit pour D Tsetup avant le front et Thold après

# III Circuits séquentiels

## Élément mémorisant : Bascule D Flip Flop (DFF)

---

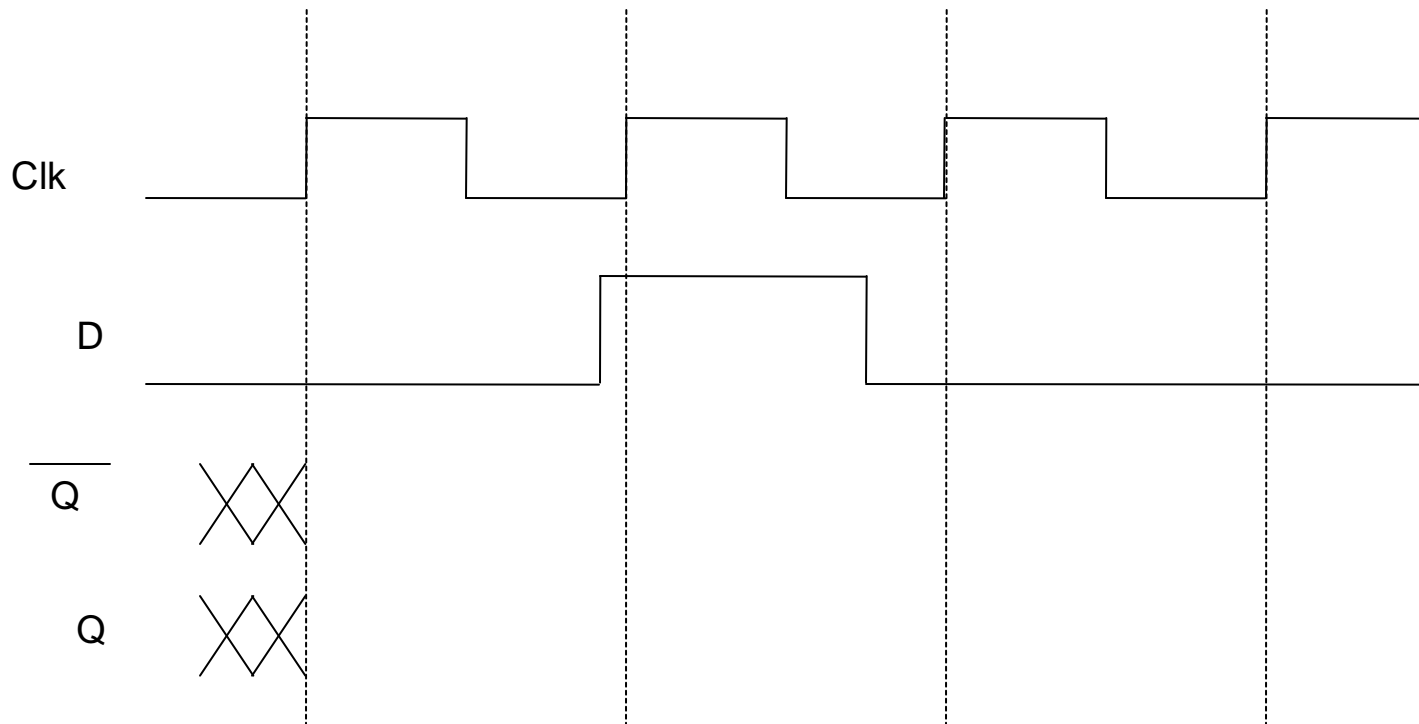


# III Circuits séquentiels

## Élément mémorisant : Bascule D Flip Flop

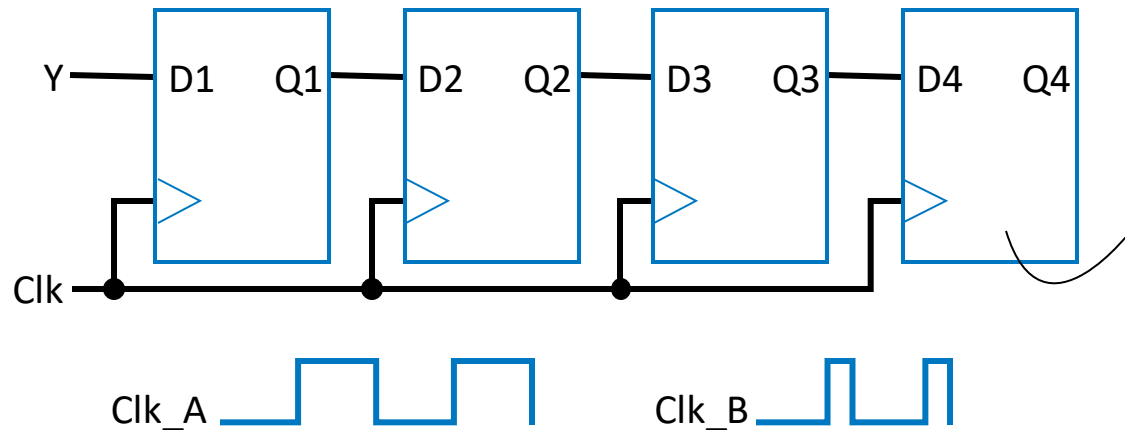
---

- Complétez le chronogramme suivant:



# III Circuits séquentiels

## Élément mémorisant : Bascule D Flip Flop



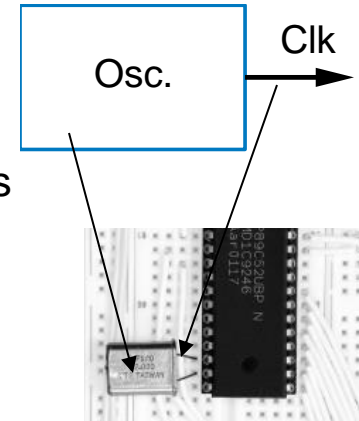
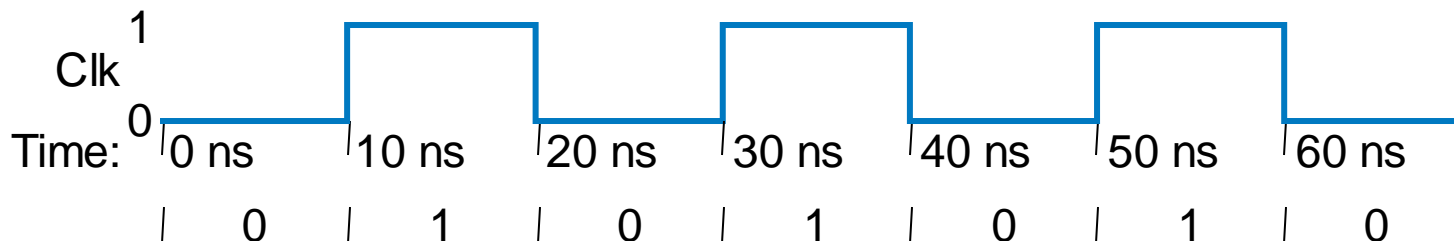
- Que ce passe-t-il pour les 2 horloges Clk\_A et Clk\_B?
- Les bascules DFF résolvent le problème de la transparence des bascules verrous



# III Circuits séquentiels

## Signal d'horloge

- Le signal d'horloge provient toujours d'un oscillateur
  - L'oscillateur génère un signal périodique
    - Ci-dessous : "Période" = 20 ns, "Frequence" =  $1/20 \text{ ns} = 50 \text{ MHz}$
    - Un "cycle" est la durée d'une période (20 ns); ci-dessous 3,5 cycles sont représentés

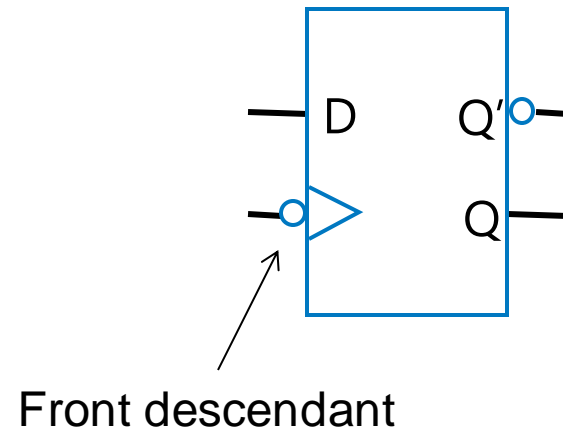
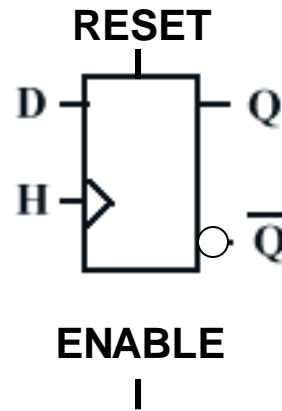


Period/Freq : Se souvenir de  $1 \text{ ns} \rightarrow 1 \text{ GHz}$

Freq.	Period
100 GHz	0.01 ns
10 GHz	0.1 ns
<b>1 GHz</b>	<b>1 ns</b>
100 MHz	10 ns
10 MHz	100 ns

# III Circuits séquentiels

## Élément mémorisant : Bascule D

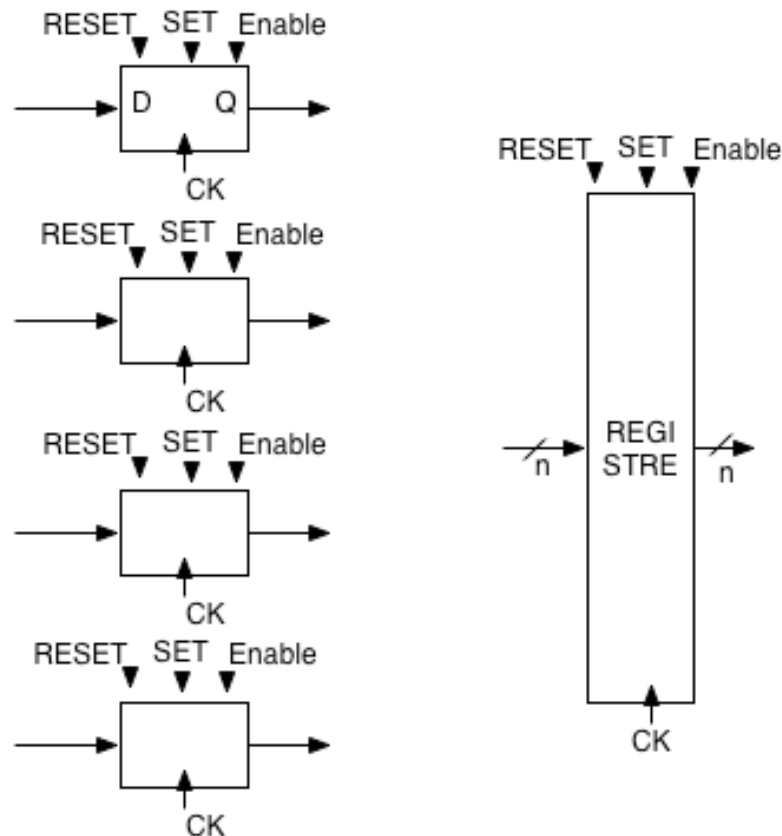


- Initialisation (**TOUJOURS PRIORITAIRE SUR ENABLE**)
  - SET / RESET : mise à '1' ou à '0' de Q
  - Synchrone / Asynchrone  $\Leftrightarrow$  il faut un front montant de H / indépendant de H
- Enable
  - à '1' : autorise le fonctionnement de la bascule comme vu précédemment.
  - à '0' : inhibe le fonctionnement de la bascule : pas de chargement au front montant.
- Activation = front d'horloge

# III Circuits séquentiels

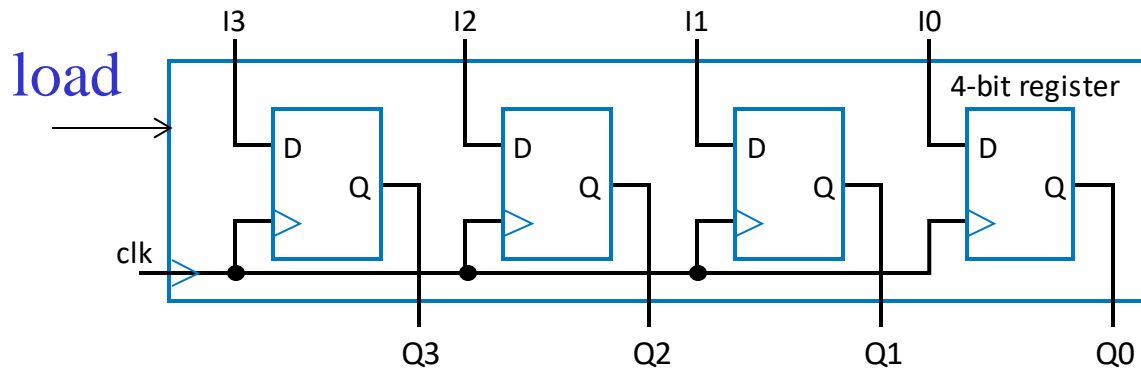
## Registres

- Un registre  $n$  bits =  $n$  bascules avec les mêmes entrées Set, reset, clk, enable ... (c'est-à-dire même initialisation et activation)

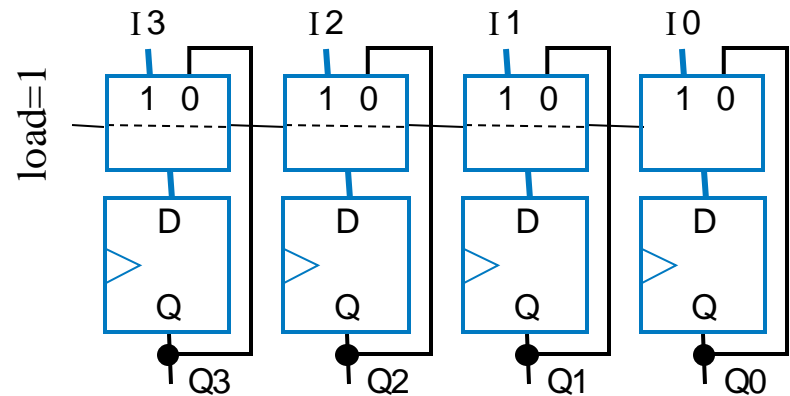
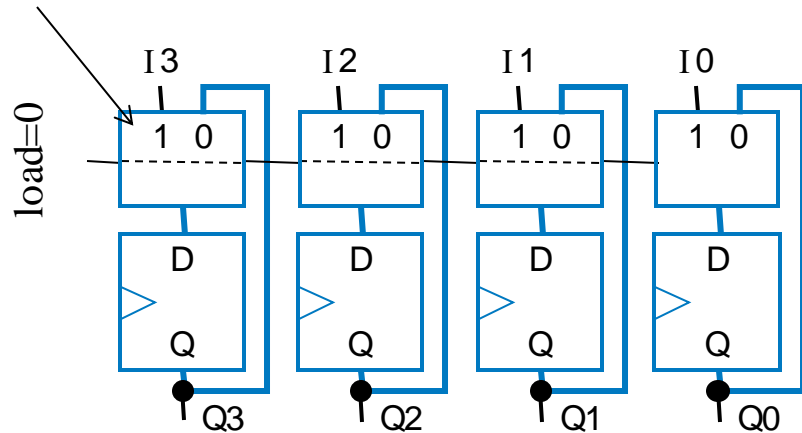


# III Circuits séquentiels

## Registres à chargement parallèle



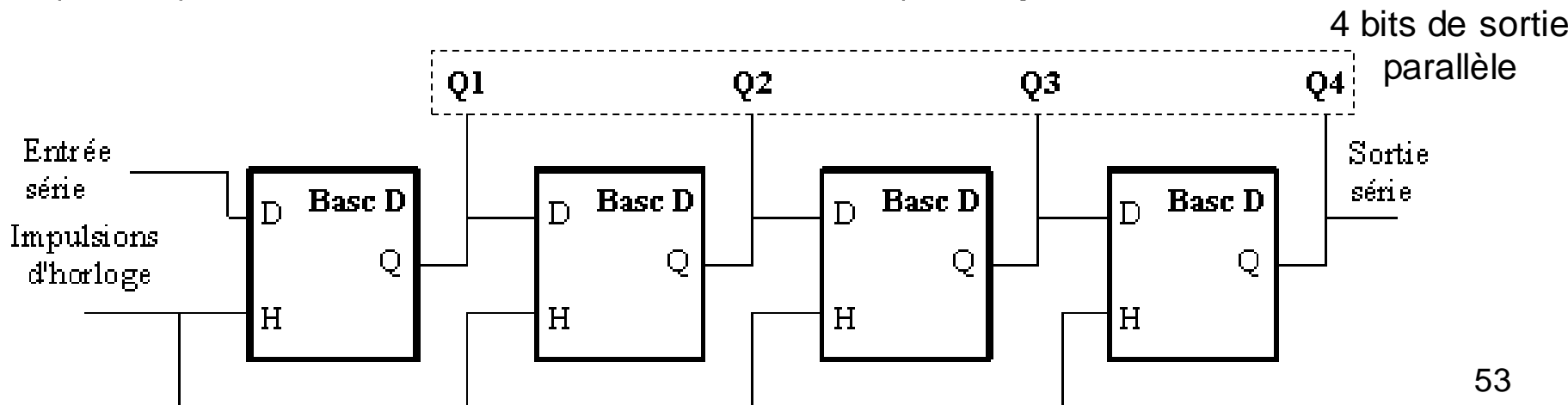
MUX 2x1



# III Circuits séquentiels

## Registres à décalage

- **Un registre à décalage  $N$  bits** est un registre  $N$  bits dans lequel les bits sont décalés à chaque coup d'horloge.
- Principe : la sortie de la bascule  $N$  est reliée à l'entrée de la bascule  $N-1$  etc...
- Exemple ci-dessous : Registre 4 bits « Serial IN – Serial OUT » (**SISO**) ET « Serial IN – Parallel OUT » (**SIPO**)

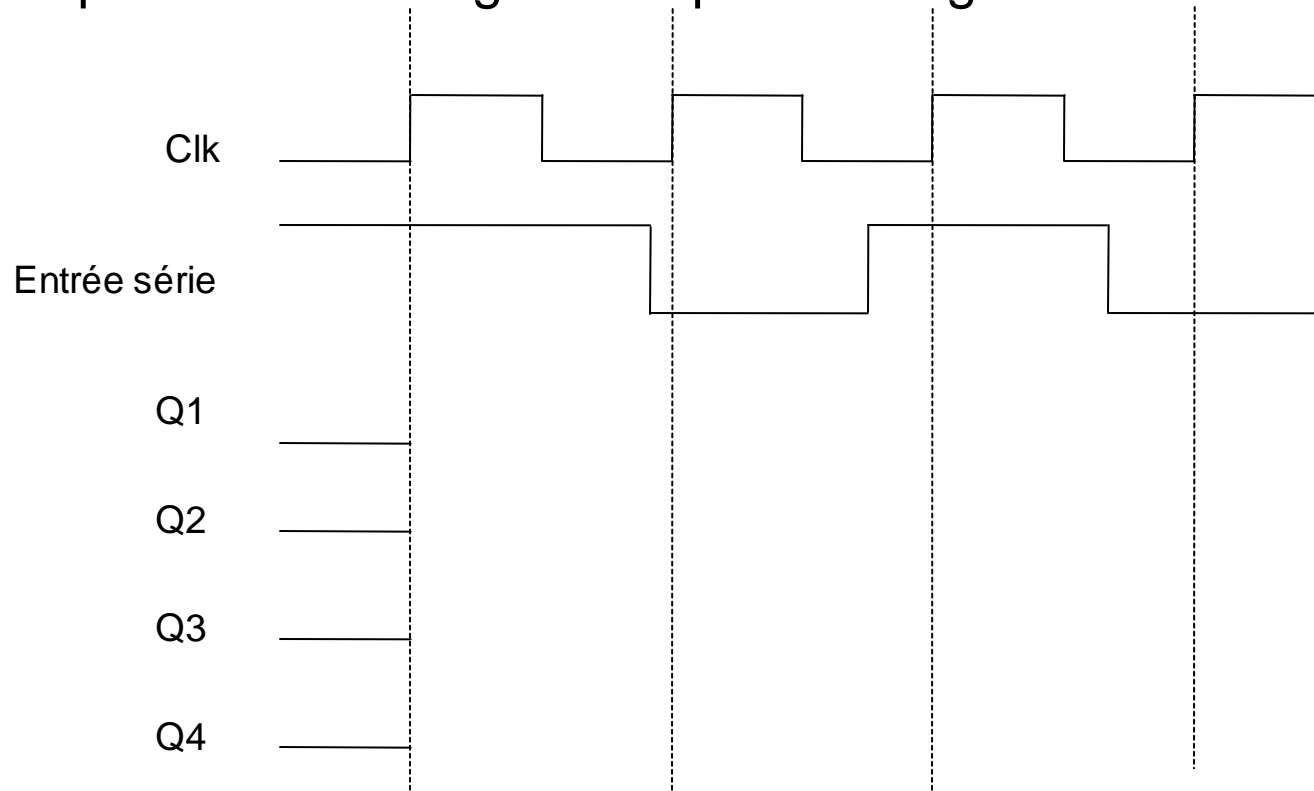


# III Circuits séquentiels

## Registres à décalage

---

- ▶ Complétez le chronogramme pour un registre SIPO :



# III Circuits séquentiels

## Registres à décalage

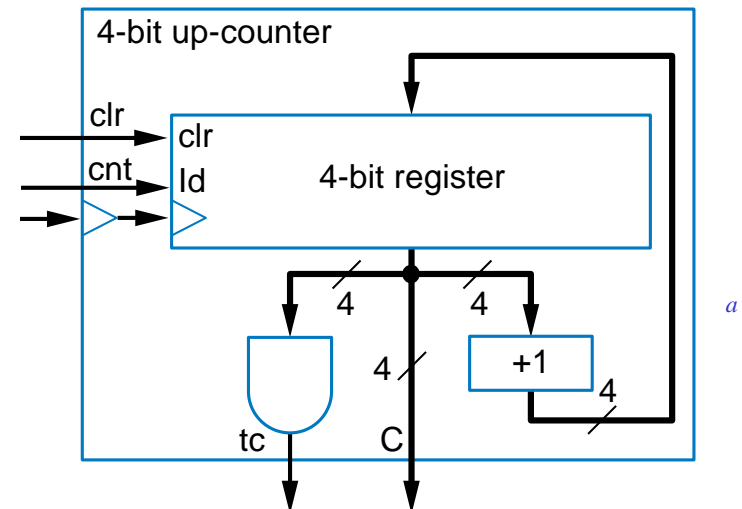
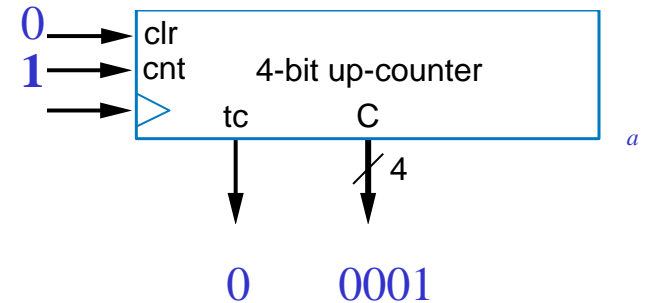
---

Using four registers, design a circuit that stores the four values present at an 8-bit input D during the previous four clock cycles. The circuit should have a single 8-bit output that can be configured using two inputs s1 and s0 to output any one of the four registers. (Hint: use an 8-bit 4x1 mux.)

# III Circuits séquentiels

## Compteurs

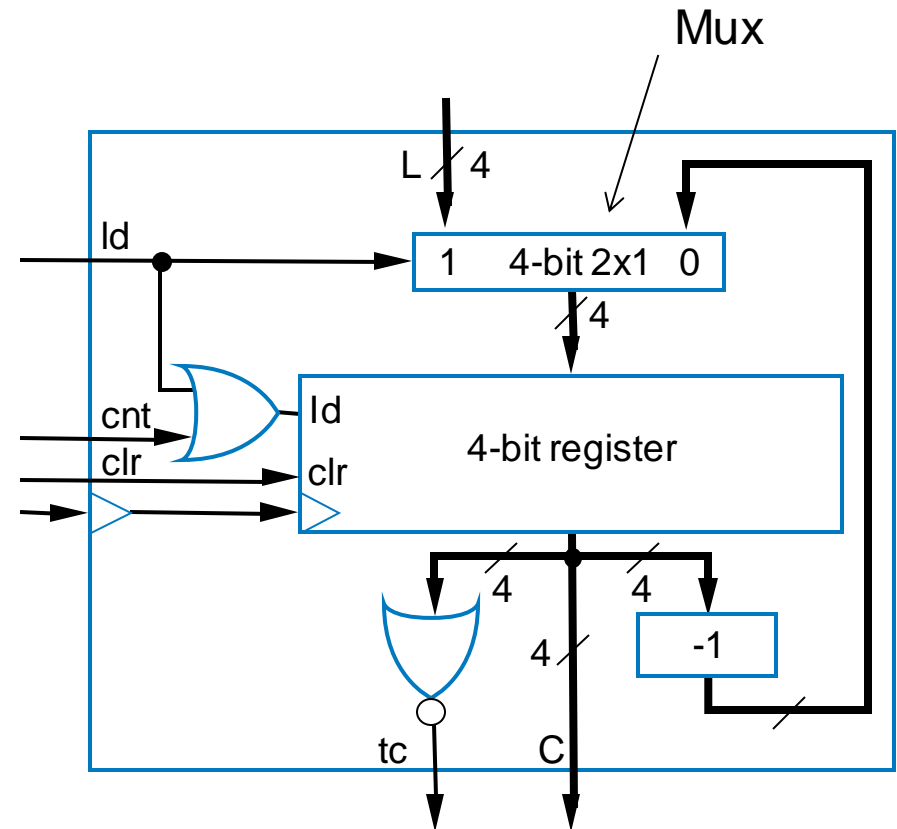
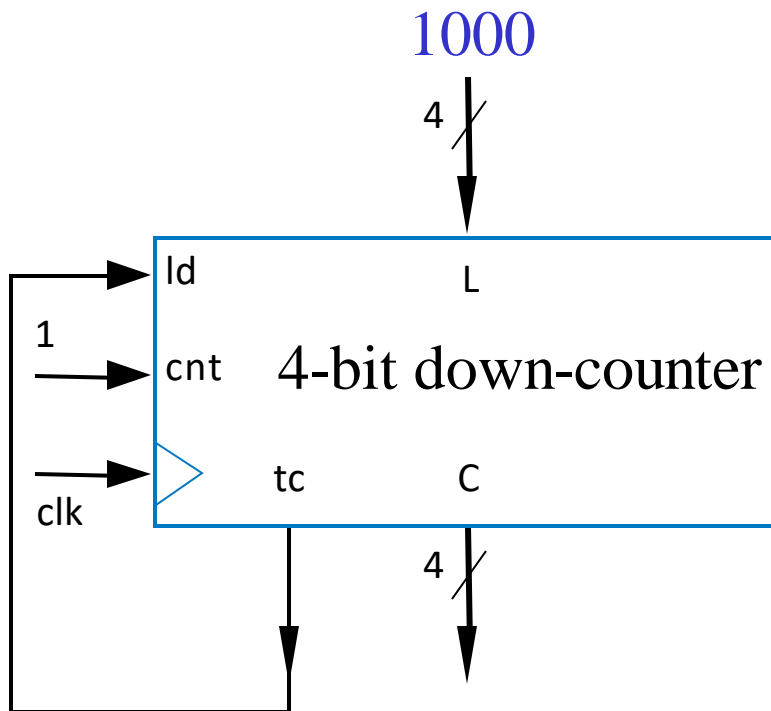
- Compteur 4 bits, principe :
  - un signal incrémentation **cnt** est envoyé au compteur (autorisation de comptage)





# III Circuits séquentiels

## Décompteur avec chargement parallèle

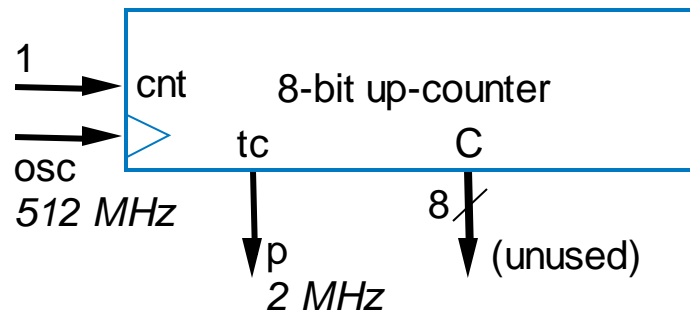


# III Circuits séquentiels

## Compteur – Diviseur de fréquence

---

- Supposons que la fréquence de l'horloge est de 512 MHz mais que l'on veut générer des impulsions de 2 MHz
  - Il faut diviser la fréquence par 256
  - Utilisation d'un compteur 8-bit (de 0 à 255)
  - tc est l'impulsion voulue (2 MHz)



# III Circuits séquentiels

## Mémoires

- Principe :
  - Choix de l'adresse par le bus d'adresses.
  - Ecriture/lecture des données par un bus données.

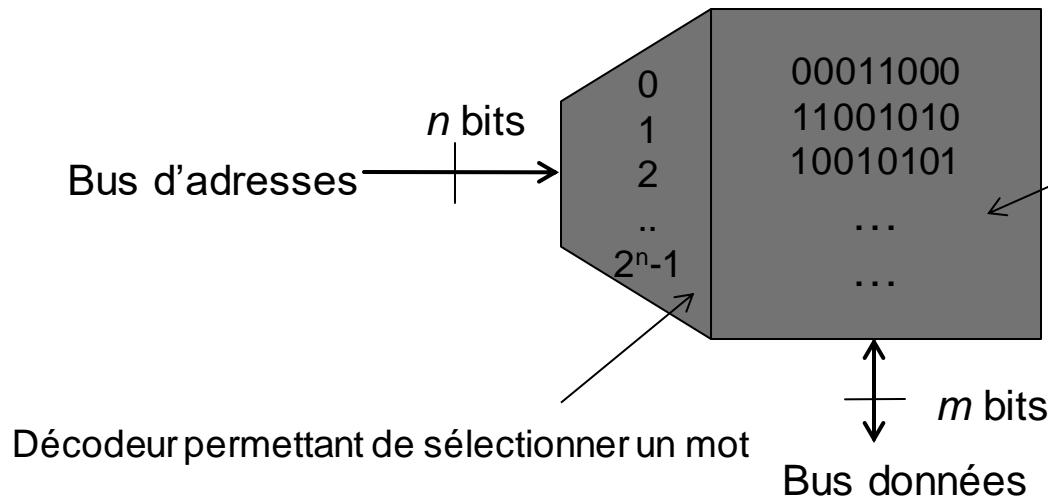


Tableau constitué d'un ensemble de registres à chargement parallèle :  
**architecture optimisée (très compacte)**

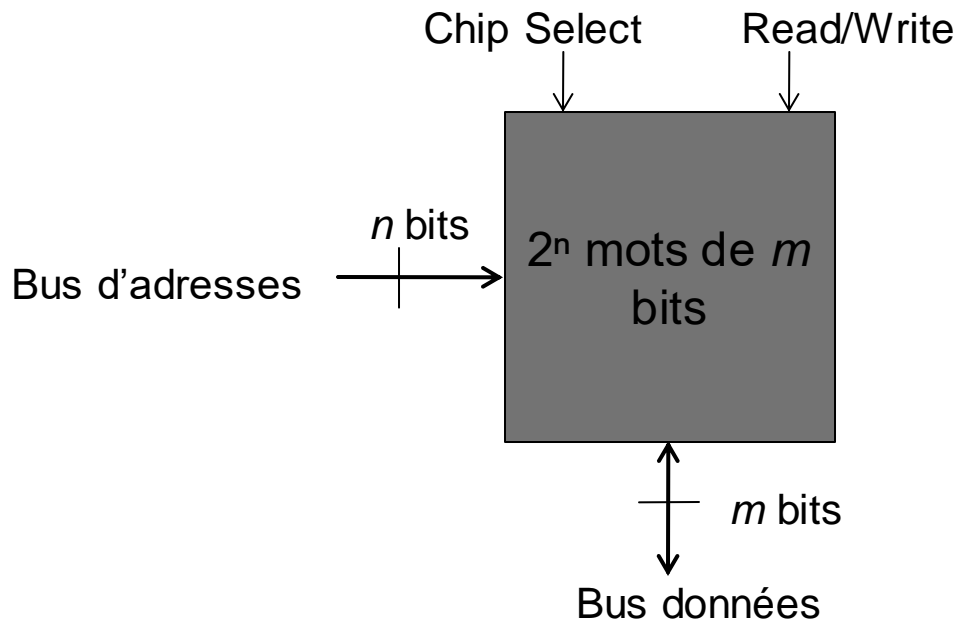
- Bus d'adresses de largeur  $n$  \* Bus données de largeur  $m$   
=>  $2^n$  mots de  $m$  bits.

# III Circuits séquentiels

## Mémoires

---

- Signaux de contrôle de la lecture/écriture en mémoire :



Read/Write	CS	
0	1	Écriture
1	1	Lecture
1/0	0	Rien

# III Circuits séquentiels

## Mémoires RAM

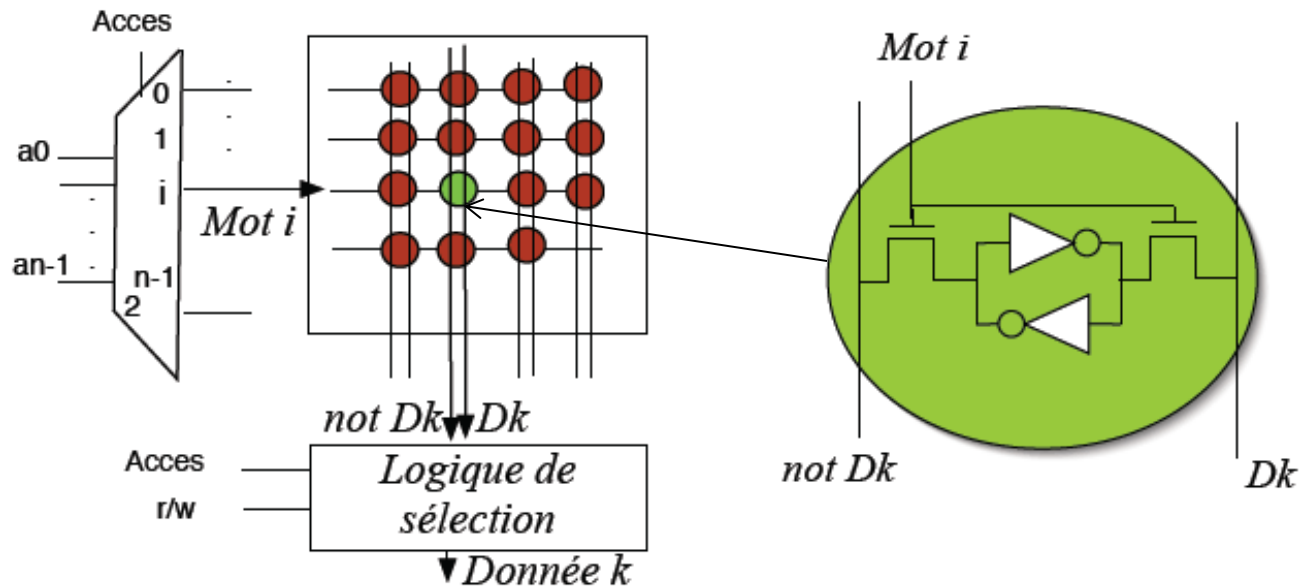
---

- Les RAMs (mémoires vives)  $\Leftrightarrow$  Random Acces Memory
  - Volatile (les informations mémorisées disparaissent quand l'alimentation disparaît)
  - Se sont des registres stockant un grand nombre de mots
- Mémoires Statiques RAMs (SRAM) :
  - Réalisation avec deux inverseurs (voir D-latch)
  - En tout 6 transistors pour stocker un bit (voir suivant)
  - Rapide  $\Rightarrow$  utilisation en cache.
- Mémoires Dynamiques RAMs (DRAM) :
  - Moins couteuses mais moins rapides que la SRAM
  - Meilleure intégration

# III Circuits séquentiels

## Mémoires SRAM

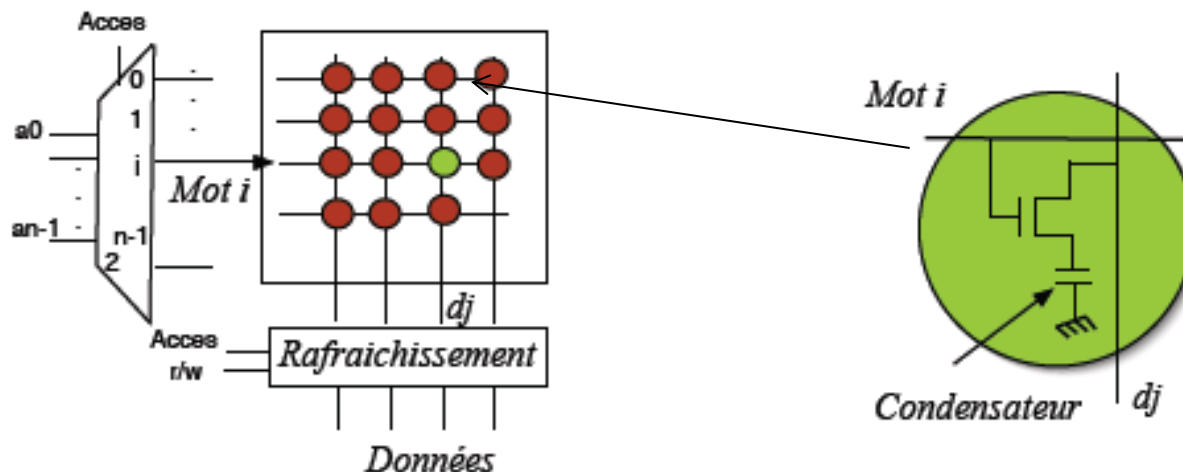
- Mémoire Statique RAMs :



# III Circuits séquentiels

## Mémoires DRAM

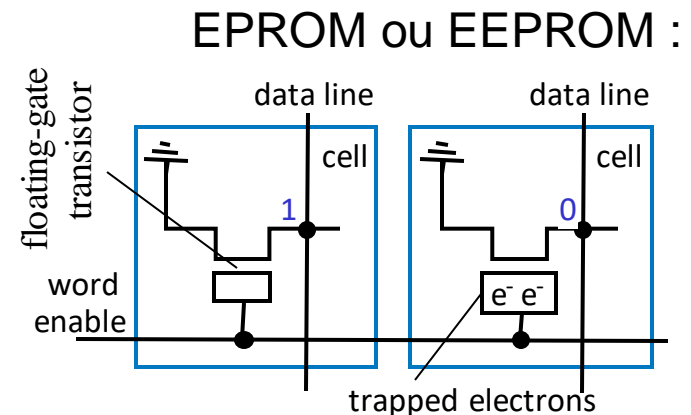
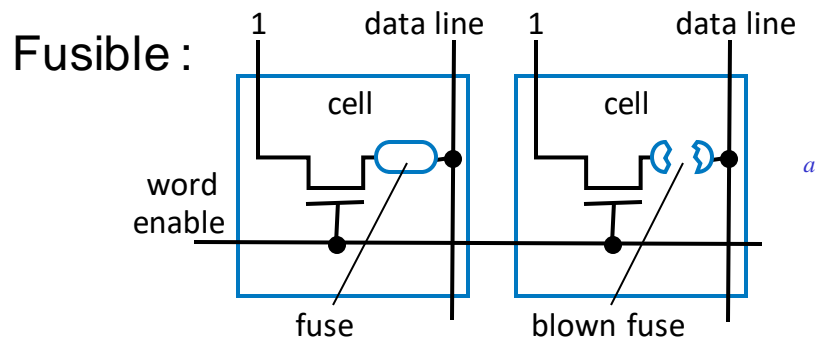
- Dynamique RAMs :
  - Réalisation avec transistor + condensateur => petit point mémoire (stockage d'un bit)
  - Besoin de rafraîchir pour palier à la décharge du condensateur toutes les 2 à 4 ms.
  - Lecture puis re-écriture par circuit spécialisé de rafraîchissement interne à la mémoire
  - Le rafraîchissement consomme ~5% du temps d'accès à la mémoire => long temps d'accès dans le pire cas



# III Circuits séquentiels

## Mémoires ROM

- Les ROMs (mémoires mortes)  $\Leftrightarrow$  Read Only Memory :
  - Non volatile (les informations mémorisées sont conservées quand l'alimentation disparaît)
  - Pas de signal Read/Write (inutile car *chip select* suffit).
  - Ce sont des circuits combinatoires.
- Les différentes ROMs :
  - Non configurables (fusible)
  - Configurables électriquement : EPROMs (programmation définitive).
  - Configurables et effaçable électriquement : EEPROMs (Flash Memory)
  - Temps d'écriture plus long que pour les RAM

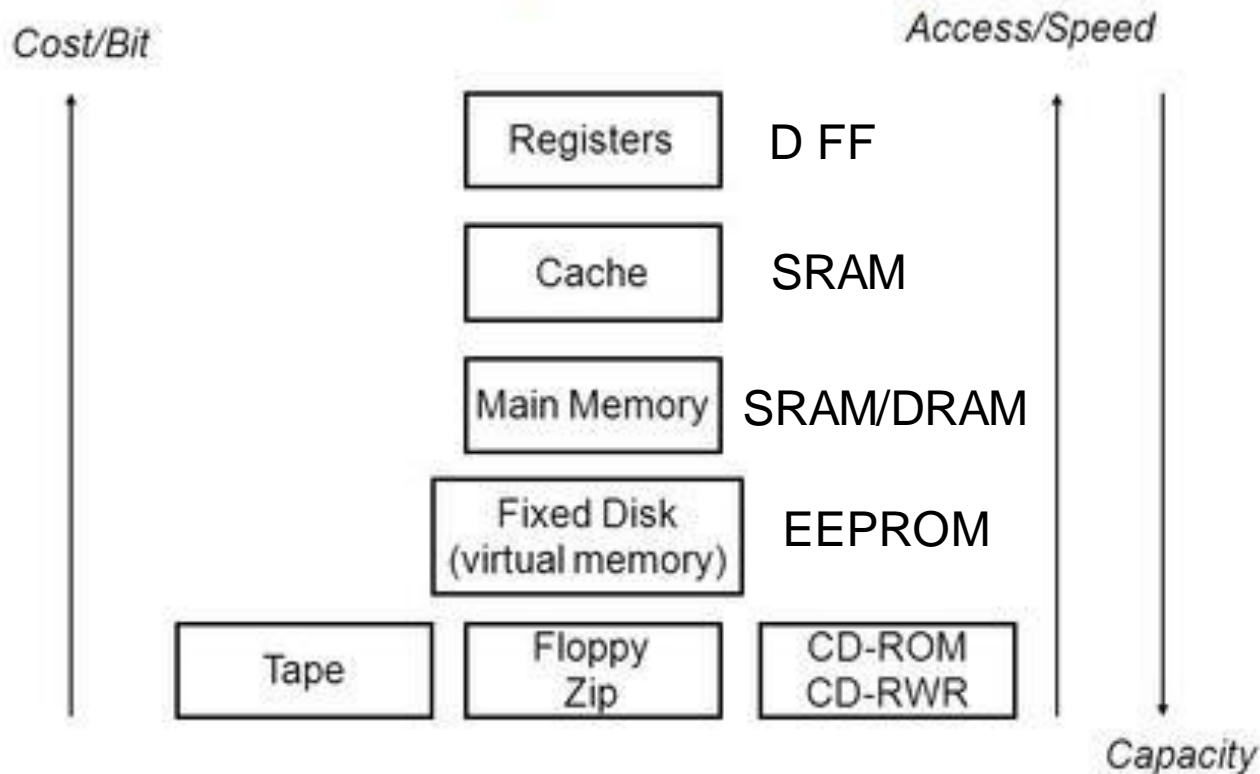




# III Circuits séquentiels

## Les mémoires : Conclusion

### Memory Hierarchy



# PLAN

---

I Représentation des nombres

II Circuits combinatoires

III Circuits séquentiels

**IV Machine à états finis**

- Définitions
- Exemples
- Architecture de Moore et Mealy
- Représentations

# IV Machine à états finis

## Définitions

---

- Automates ou Machine à états finis (Finite State Machine = FSM) souvent utilisés en informatique/automatique
- Représentation graphique simple d'un système discret à entrées/sorties
- On va s'intéresser ici à leurs réalisations à l'aide de **circuits numériques séquentiels**
- 2 principaux types de FSM :
  - **Moore**
  - **Mealy**

# IV Machine à états finis

## Représentation : diagramme de transitions

- On utilise un graphe d'états (ou diagramme de transitions) :
  - État : 1 rond + 1 nom à l'intérieur (le nom de l'état)
  - État initial : « flèche venant de nulle part avec rst » et/ou double cercle
  - Transition : une flèche portant la valeur condition associée à cette transition, la transition n'est franchie que lors du front actif de l'horloge
  - Sortie :
    - soit associée à l'état (Moore)
    - soit associée à une transition (e/s) (Mealy)

### Exemple :

Quand il n'y a pas de condition alors la condition est toujours vraie

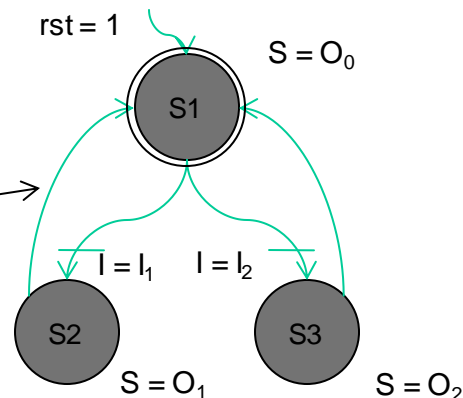


Diagramme de transitions  
d'une machine de Moore :

1 état = 1 sortie

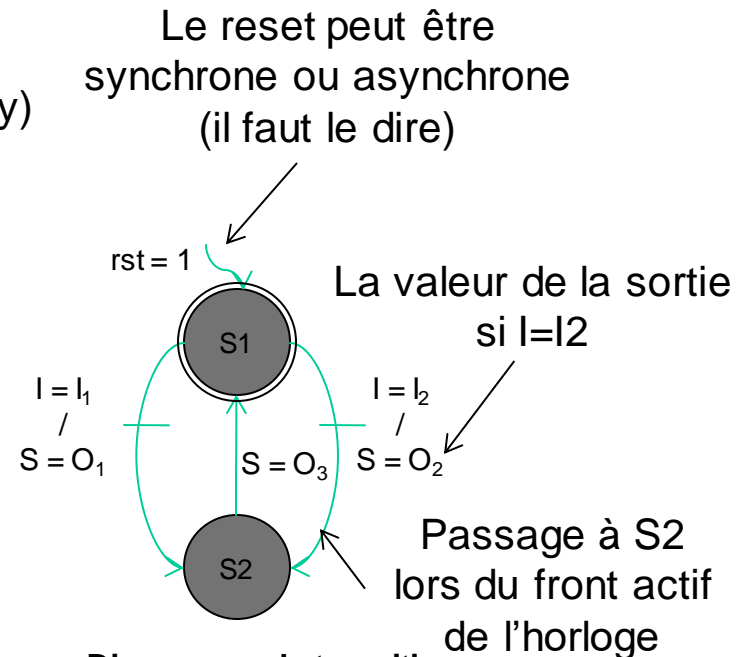


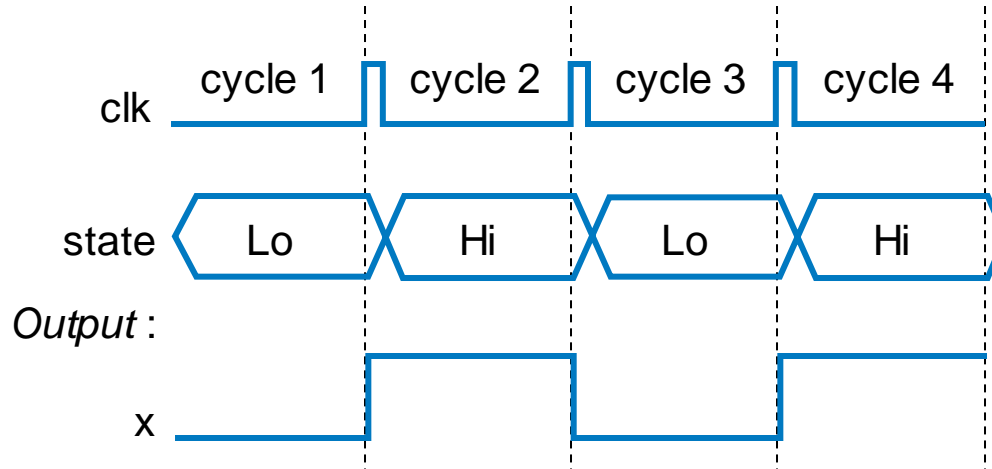
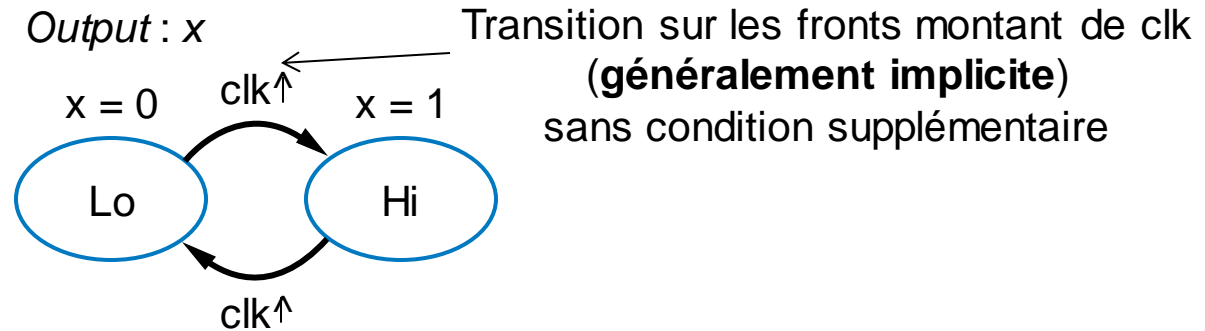
Diagramme de transitions  
d'une machine de Mealy :

1 transition = 1 sortie

# IV Machine à états finis

## Exemple

Diagramme de transitions  
d'une machine de Moore :

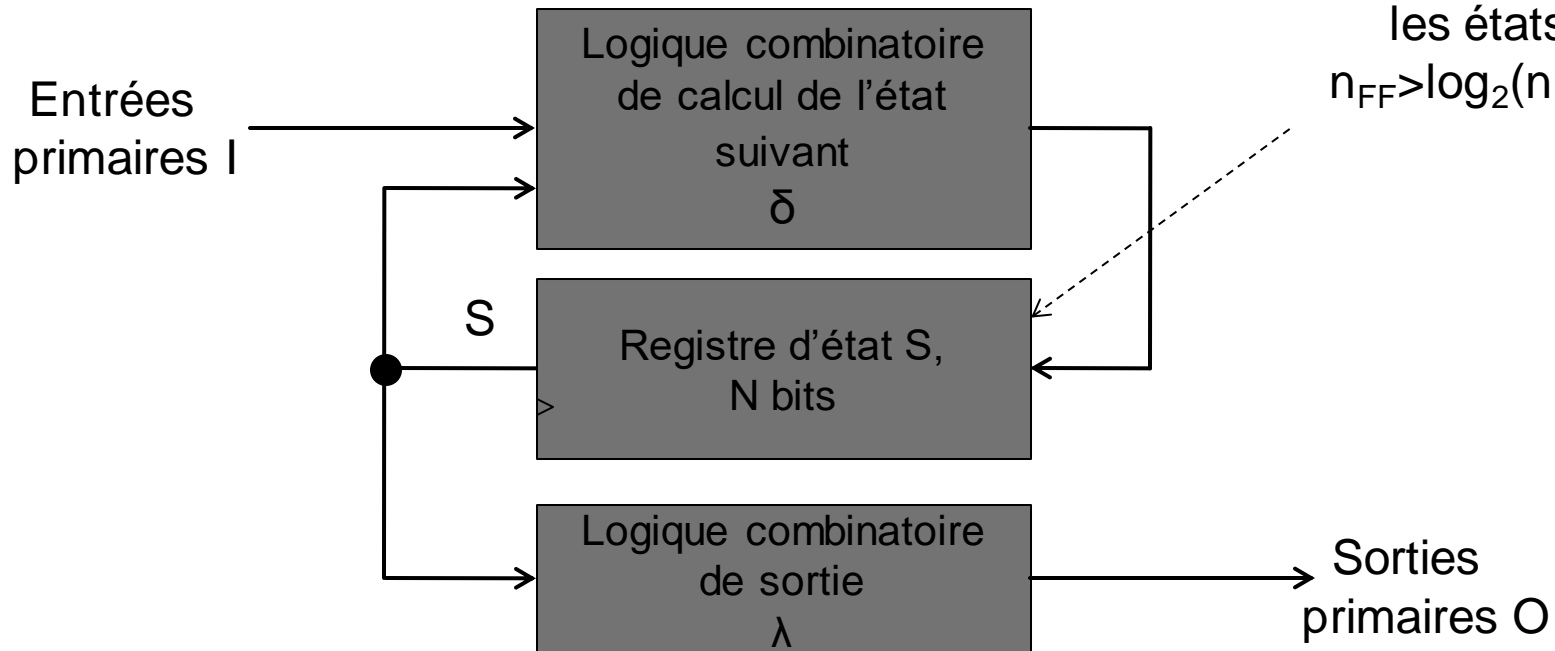


# IV Machines à états finis

## Architecture Machine de Moore

- Sorties calculées uniquement avec l'état courant :
    - $\lambda : S \rightarrow O$
- Le nombre de Flip-Flop doit être suffisant pour permettre d'identifier tous les états

$$n_{FF} > \log_2(n_{Etat})$$



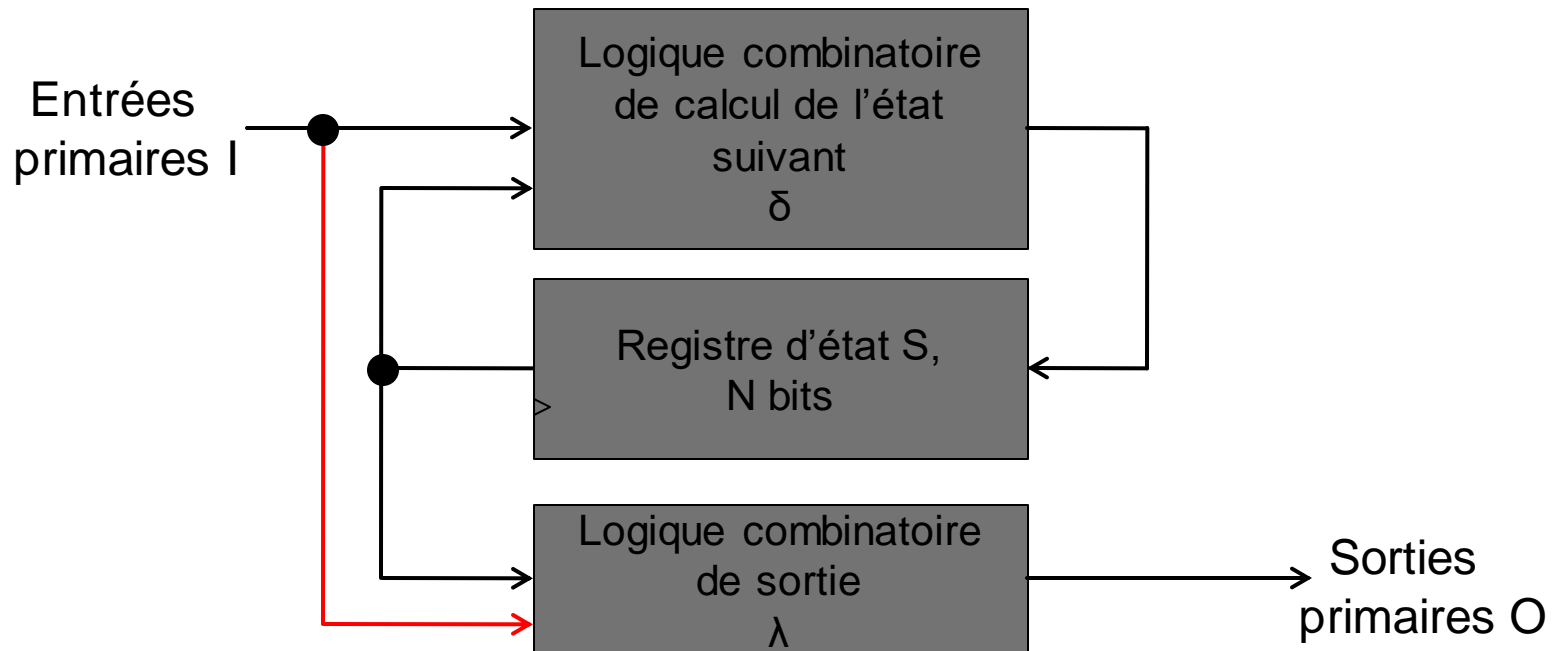
**Dans une machine de Moore la sortie dépend seulement de l'état :  
la sortie O est synchrone**

# IV Machines à états finis

## Architecture Mealy

---

- Sorties calculées avec l'état courant et les entrées primaires I :
  - $\lambda : S \times I \rightarrow O$



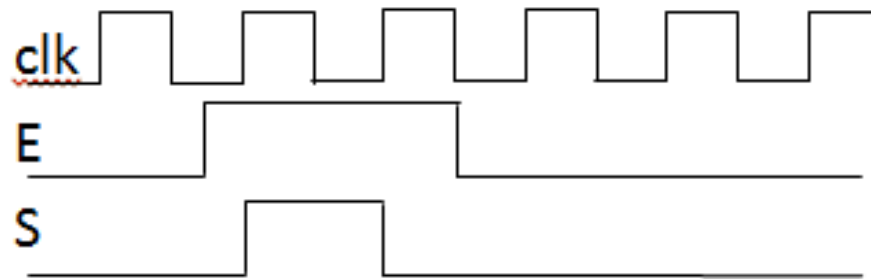
**Dans une machine de Mealy la sortie dépend de l'état et des entrées :  
La sortie O est asynchrone**

# IV Machines à états finis

## Exercices

---

- Donnez les diagrammes de transition des machines à états de Moore puis de Mealy correspondant à un détecteur de front montant

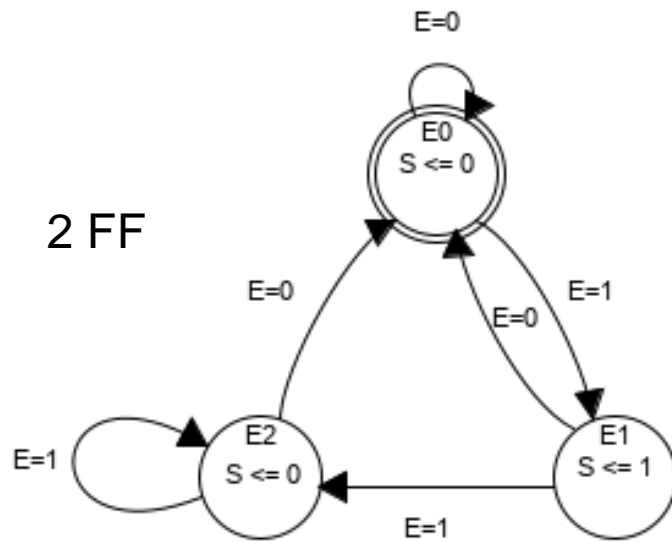


- Pour chaque machine complétez le chronogramme avec les signaux **clk**, **E**, **S**, et **Etat**
- Combien de bascules sont nécessaires pour chaque machine?

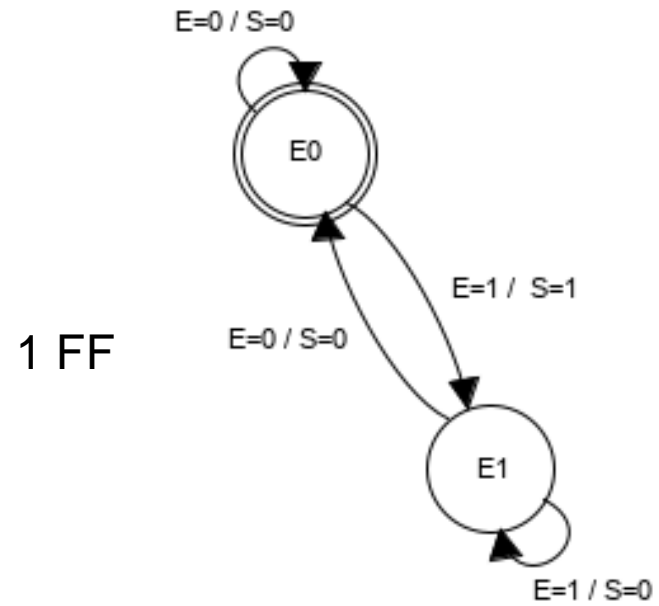
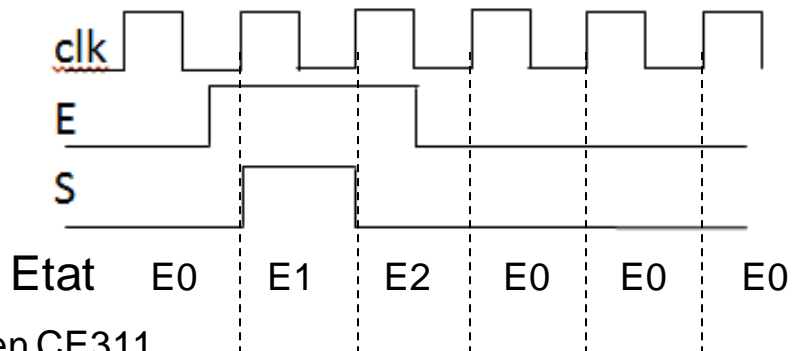


# IV Machines à états finis

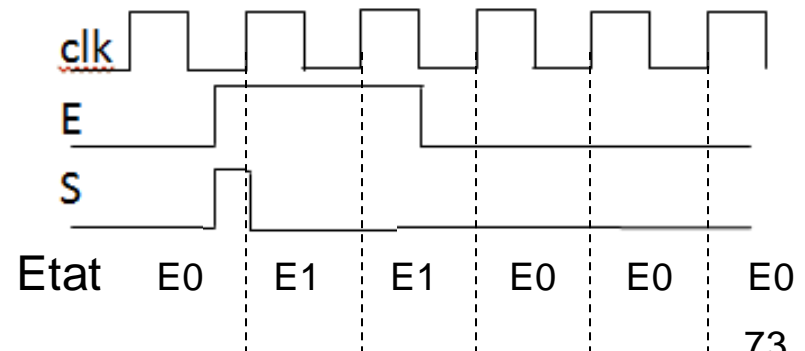
## Exercices : détecteur de front montant



Machine de Moore



Machine de Mealy

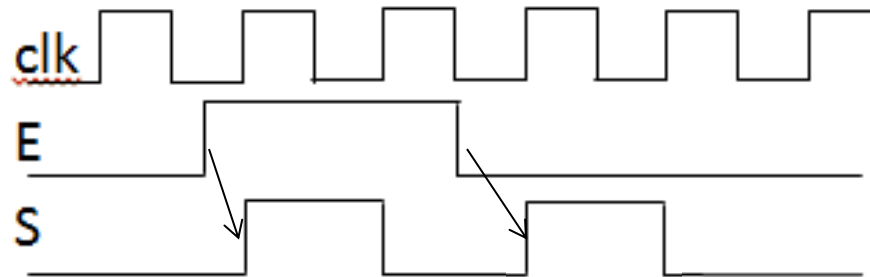


# IV Machines à états finis

## Exercices

---

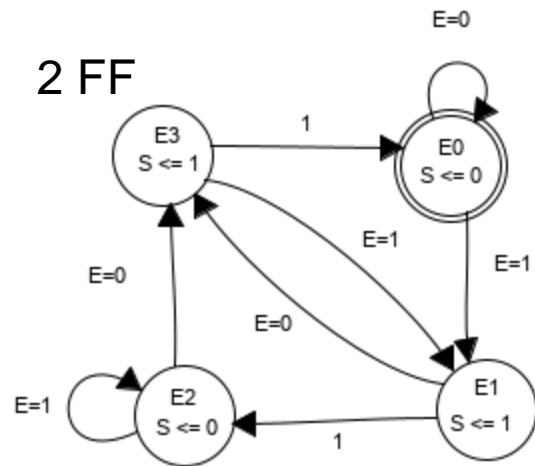
- Donnez les diagrammes de transition des machines à états de Moore et de Mealy correspondant à un détecteur de front montant et descendant



- Combien de bascules sont nécessaires pour chaque machine?

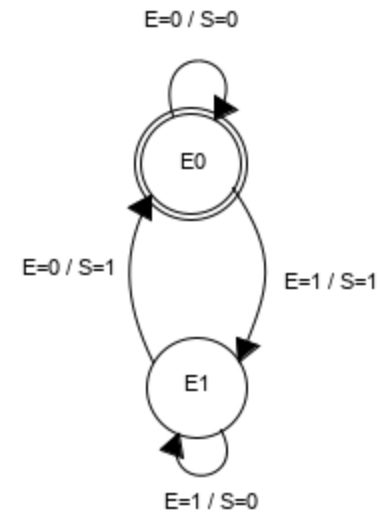
# IV Machines à états finis

Exercices : détecteur de front montant et descendant



Machine de Moore

1 FF



Machine de Mealy