Architecture des Ordinateurs
Lecture 2

Prof. Yérali Gandica

CY-Tech Cergy Paris Université
2022

Thèorèmes de l'algèbre de Boole

- Théorème d'involution : $\overline{\overline{A}} = A$
  $\overline{\overline{\overline{A}}} = \overline{A}$

- Théorème d'inclusion : $A.B + A.\overline{B} = A$
  $(A + B).(A + \overline{B}) = A$

  Démonstration : mettre A en facteur (distributivité « à l'envers ») :
  $A.B + A.\overline{B} = A.(B + \overline{B}) = A$
  $(A + B).(A + \overline{B}) = A + B.\overline{B} = A$

- **Théorème d'allégement** : $\mathbf{A.(\overline{A} + B) = A.B}$
  $\mathbf{A + \overline{A}.B = A + B}$

  Démonstration : utiliser la distributivité (du ET et du OU) :
  $A.(\overline{A} + B) = A.\overline{A} + A.B = A.B$
  $A + \overline{A}.B = (A + \overline{A}).(A + B) = A + B$

Thèorèmes de l'algèbre de Boole

- **Théorème d'absorption** : $\mathbf{A.(A+B) = A}$
  $\mathbf{A+(A.B) = A}$

Démonstration par la distributivité du ET (*utilisée dans les 2 sens*) :

$A.(A+B) = A.A + A.B$ (distributivité du ET)
$= A + A.B$ (2ème forme du théorème d'absorption)
$= A.(B+1)$ (mise en facteur de A : distributivité du ET « à l'envers »)
$= A.1$
$= A$

Démonstration par la distributivité du OU (*utilisée dans les 2 sens*) :

$A+A.B = (A+A).(A+B)$ (distributivité du OU)
$= A.(A+B)$ (1ère forme du théorème d'absorption)
$= (A+0).(A+B)$ (pour y voir plus clair dans ce qui va suivre ...)
$= A + (B.0)$ (distributivité du OU à l'envers : « factorisation par l'addition »)
$= A+0$
$= A$

- **Théorème de De Morgan** : $\overline{\mathbf{A.B}} = \overline{\mathbf{A}} + \overline{\mathbf{B}}$ ➨ porte ET-NON
  $\overline{\mathbf{A+B}} = \overline{\mathbf{A}} \,.\, \overline{\mathbf{B}}$ ➨ porte OU-NON

Table de Karnaugh: Simplifications de fonctions booléennes

- Nous allons voir que les fonctions booléennes sont en fait implantées à l'aide de portes logiques constituées de transistors.
- Afin d'économiser de l'espace, de l'énergie et de l'argent, on souhaite utiliser le moins de transistors possibles.
- Pour simplifier, on peut tout simplement utiliser les axiomes et les propriétés de l'algèbre de Boole, afin de passer d'une expression à une autre plus simple.
- La méthode consiste à présenter les états d'une fonction logique, par une simplification visuelle.
- Chaque case du tableau correspond à une combinaison des variables d'entrées, donc à une ligne de la table de vérité.
- Le tableau de Karnaugh aura autant de cases que la table de vérité possède de lignes.

Table de Karnaugh: Simplifications de fonctions booléennes

- Nous allons voir que les fonctions booléennes sont en fait implantées à l'aide de portes logiques constituées de transistors.
- Afin d'économiser de l'espace, de l'énergie et de l'argent, on souhaite utiliser le moins de transistors possibles.
- Pour simplifier, on peut tout simplement utiliser les axiomes et les propriétés de l'algèbre de Boole, afin de passer d'une expression à une autre plus simple.
- La méthode consiste à présenter les états d'une fonction logique, par une simplification visuelle.
- Chaque case du tableau correspond à une combinaison des variables d'entrées, donc à une ligne de la table de vérité.
- Le tableau de Karnaugh aura autant de cases que la table de vérité possède de lignes.

Table de Karnaugh: Simplifications de fonctions booléennes

- Nous allons voir que les fonctions booléennes sont en fait implantées à l'aide de portes logiques constituées de transistors.
- Afin d'économiser de l'espace, de l'énergie et de l'argent, on souhaite utiliser le moins de transistors possibles.
- Pour simplifier, on peut tout simplement utiliser les axiomes et les propriétés de l'algèbre de Boole, afin de passer d'une expression à une autre plus simple.
- La méthode consiste à présenter les états d'une fonction logique, par une simplification visuelle.
- Chaque case du tableau correspond à une combinaison des variables d'entrées, donc à une ligne de la table de vérité.
- Le tableau de Karnaugh aura autant de cases que la table de vérité possède de lignes.

Table de Karnaugh: Simplifications de fonctions booléennes

- Nous allons voir que les fonctions booléennes sont en fait implantées à l'aide de portes logiques constituées de transistors.
- Afin d'économiser de l'espace, de l'énergie et de l'argent, on souhaite utiliser le moins de transistors possibles.
- Pour simplifier, on peut tout simplement utiliser les axiomes et les propriétés de l'algèbre de Boole, afin de passer d'une expression à une autre plus simple.
- La méthode consiste à présenter les états d'une fonction logique, par une simplification visuelle.
- Chaque case du tableau correspond à une combinaison des variables d'entrées, donc à une ligne de la table de vérité.
- Le tableau de Karnaugh aura autant de cases que la table de vérité possède de lignes.

Table de Karnaugh: Simplifications de fonctions booléennes

- Nous allons voir que les fonctions booléennes sont en fait implantées à l'aide de portes logiques constituées de transistors.
- Afin d'économiser de l'espace, de l'énergie et de l'argent, on souhaite utiliser le moins de transistors possibles.
- Pour simplifier, on peut tout simplement utiliser les axiomes et les propriétés de l'algèbre de Boole, afin de passer d'une expression à une autre plus simple.
- La méthode consiste à présenter les états d'une fonction logique, par une simplification visuelle.
- Chaque case du tableau correspond à une combinaison des variables d'entrées, donc à une ligne de la table de vérité.
- Le tableau de Karnaugh aura autant de cases que la table de vérité possède de lignes.

Table de Karnaugh: Simplifications de fonctions booléennes

- Nous allons voir que les fonctions booléennes sont en fait implantées à l'aide de portes logiques constituées de transistors.
- Afin d'économiser de l'espace, de l'énergie et de l'argent, on souhaite utiliser le moins de transistors possibles.
- Pour simplifier, on peut tout simplement utiliser les axiomes et les propriétés de l'algèbre de Boole, afin de passer d'une expression à une autre plus simple.
- La méthode consiste à présenter les états d'une fonction logique, par une simplification visuelle.
- Chaque case du tableau correspond à une combinaison des variables d'entrées, donc à une ligne de la table de vérité.
- Le tableau de Karnaugh aura autant de cases que la table de vérité possède de lignes.

Table de Karnaugh: Simplifications de fonctions booléennes

À 3 variables:

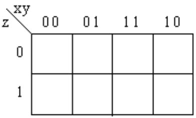


Tableau à 3 variables

À 4 variables

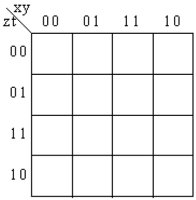


Tableau à 4 variables

Table de Karnaugh: procédure à suivre

- Construire la table de vérité.
- La table (sphérique) est construite, comme indiqué avant, où un seul chiffre change entre les cases adjacentes.
- On remplit la table avec les valeurs de la table de vérité.

$$x.y.z + \bar{x}.y.z + x.\bar{y}.z + x.y.\bar{z}$$

$z \backslash xy$	00	01	11	10
0	0	0	1	0
1	0	1	1	1

Table de Karnaugh: procédure à suivre

- Construire la table de vérité.
- La table (sphérique) est construite, comme indiqué avant, où un seul chiffre change entre les cases adjacentes.
- On remplit la table avec les valeurs de la table de vérité.

$$x.y.z + \bar{x}.y.z + x.\bar{y}.z + x.y.\bar{z}$$

$z \backslash xy$	00	01	11	10
0	0	0	1	0
1	0	1	1	1

Table de Karnaugh: procédure à suivre

- Construire la table de vérité.
- La table (sphérique) est construite, comme indiqué avant, où un seul chiffre change entre les cases adjacentes.
- On remplit la table avec les valeurs de la table de vérité.

$$x.y.z + \bar{x}.y.z + x.\bar{y}.z + x.y.\bar{z}$$

$\frac{xy}{z}$	00	01	11	10
0	0	0	1	0
1	0	1	1	1

Table de Karnaugh: procédure à suivre

- Ensuite, on cherche à recouvrir tous les 1 du tableau en formant des regroupements. Chaque regroupement :
  - ne contient que des 1 adjacents,
  - doit former un rectangle,
  - et le nombre d'éléments d'un regroupement doit être une puissance de deux.

  On choisit toujours:
  - les plus grands regroupements possibles,
  - et le plus petit nombre possible de regroupements recouvrant les 1 du tableaux.
- On simplifie finalement $(a.b) + (a.\bar{b}) = a.(b + \bar{b}) = a$...et voilà!!

0	0	1	0
0	1	1	1

Table de Karnaugh: procédure à suivre

- Ensuite, on cherche à recouvrir tous les 1 du tableau en formant des regroupements. Chaque regroupement :
  - ne contient que des 1 adjacents,
  - doit former un rectangle,
  - et le nombre d'éléments d'un regroupement doit être une puissance de deux.

  On choisit toujours:
  - les plus grands regroupements possibles,
  - et le plus petit nombre possible de regroupements recouvrant les 1 du tableaux.
- On simplifie finalement $(a.b) + (a.\bar{b}) = a.(b + \bar{b}) = a$...et voilà!!

0	0	1	0
0	1	1	1

Circuits combinatoires

Un circuit combinatoire est un circuit physique élaboré à partir de composants électroniques.

- Il comporte des entrées et des sorties.
- Les entrées et sorties sont des valeurs booléennes et chaque sortie est valeur d'une fonction booléenne fonction des entrées.
- Les circuits combinatoires sont construits à partir de "portes logiques".

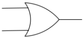
Portes logiques

- Les portes logiques sont des fonctions booléennes élémentaires; elles disposent d'entrées (à gauche sur les dessins) et d'une sortie (à droite).
- Des signaux arrivent sur les entrées (0 ou 1) et un signal est produit sur la sortie.
- Les tables de vérité donnent la valeur de la sortie pour chacune des portes en fonction de la valeur de entrées.

Porte ET

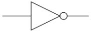
0	0	0
0	1	0
1	0	0
1	1	1

Porte OU



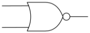
0	0	0
0	1	1
1	0	1
1	1	1

Porte NON



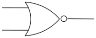
0	1
1	0

Porte NON-ET



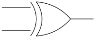
0	0	1
0	1	1
1	0	1
1	1	0

Porte NON-OU



0	0	1
0	1	0
1	0	0
1	1	0

Porte OU-X



0	0	0
0	1	1
1	0	1
1	1	0

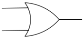
Portes logiques

- Les portes logiques sont des fonctions booléennes élémentaires; elles disposent d'entrées (à gauche sur les dessins) et d'une sortie (à droite).
- Des signaux arrivent sur les entrées (0 ou 1) et un signal est produit sur la sortie.
- Les tables de vérité donnent la valeur de la sortie pour chacune des portes en fonction de la valeur de entrées.

Porte ET

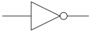
0	0	0
0	1	0
1	0	0
1	1	1

Porte OU



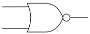
0	0	0
0	1	1
1	0	1
1	1	1

Porte NON



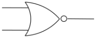
0	1
1	0

Porte NON-ET



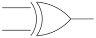
0	0	1
0	1	1
1	0	1
1	1	0

Porte NON-OU



0	0	1
0	1	0
1	0	0
1	1	0

Porte OU-X



0	0	0
0	1	1
1	0	1
1	1	0

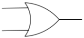
Portes logiques

- Les portes logiques sont des fonctions booléennes élémentaires; elles disposent d'entrées (à gauche sur les dessins) et d'une sortie (à droite).
- Des signaux arrivent sur les entrées (0 ou 1) et un signal est produit sur la sortie.
- Les tables de vérité donnent la valeur de la sortie pour chacune des portes en fonction de la valeur de entrées.

Porte ET

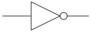
0	0	0
0	1	0
1	0	0
1	1	1

Porte OU



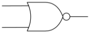
0	0	0
0	1	1
1	0	1
1	1	1

Porte NON



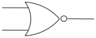
0	1
1	0

Porte NON-ET



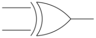
0	0	1
0	1	1
1	0	1
1	1	0

Porte NON-OU



0	0	1
0	1	0
1	0	0
1	1	0

Porte OU-X

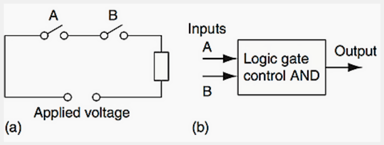


0	0	0
0	1	1
1	0	1
1	1	0

Exemples de circuits

Fonction logique ET

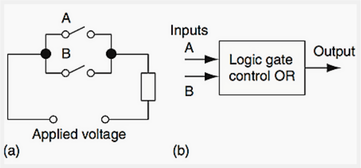
Contributions		Sortie
UNE	B	
0	0	**0**
0	1	**0**
1	0	**0**
1	1	**1**



Exemples de circuits

Fonction logique OU

Contributions		Sortie
UNE	B	
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**1**



Circuits combinatoire

- est défini par un ensemble de portes reliées les unes aux autres.
- les sorties des portes sont reliées aux entrées d'autres portes (définissant une orientation des connexions)
- peut être vu comme une porte logique (à plusieurs sorties).

De l'idéal à la réalité... Les circuits combinatoires sont **une idéalisation** dans lesquels

— le temps de propagation n'est pas pris en compte
— la sortie est disponible dès que les entrées sont présentes

En réalité le temps de passage de 0 à 1 **n'est pas**

— immédiat (temps de parcours du courant électrique)
— instantané (temps de réponse d'une porte)

Circuits combinatoire

- est défini par un ensemble de portes reliées les unes aux autres.
- les sorties des portes sont reliées aux entrées d'autres portes (définissant une orientation des connexions)
- peut être vu comme une porte logique (à plusieurs sorties).

De l'idéal à la réalité... Les circuits combinatoires sont **une idéalisation** dans lesquels

— le temps de propagation n'est pas pris en compte
— la sortie est disponible dès que les entrées sont présentes

En réalité le temps de passage de 0 à 1 **n'est pas**

— immédiat (temps de parcours du courant électrique)
— instantané (temps de réponse d'une porte)

Circuits combinatoire

- est défini par un ensemble de portes reliées les unes aux autres.
- les sorties des portes sont reliées aux entrées d'autres portes (définissant une orientation des connexions)
- peut être vu comme une porte logique (à plusieurs sorties).

De l'idéal à la réalité... Les circuits combinatoires sont **une idéalisation** dans lesquels

— le temps de propagation n'est pas pris en compte

— la sortie est disponible dès que les entrées sont présentes

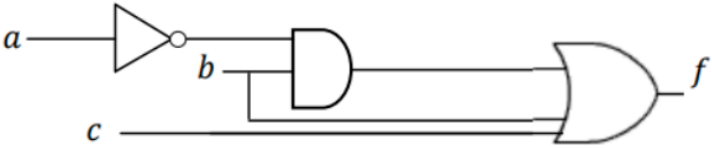
En réalité le temps de passage de 0 à 1 **n'est pas**

— immédiat (temps de parcours du courant électrique)

— instantané (temps de réponse d'une porte)

Example de circuits combinatoire

$\overline{a}b + b + c$



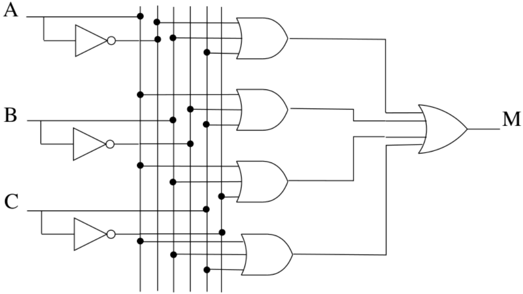
Example de circuits combinatoire

$\overline{a}b + b + c$

On peut aussi simplifier l'équation précédente en utilisant les caractéristiques de l'algèbre de Bool sous la forme:

$\overline{a}b + b + c = b.(\overline{a} + 1) + c = b + c$.

Example de circuits combinatoire: Le circuit "Majorité"



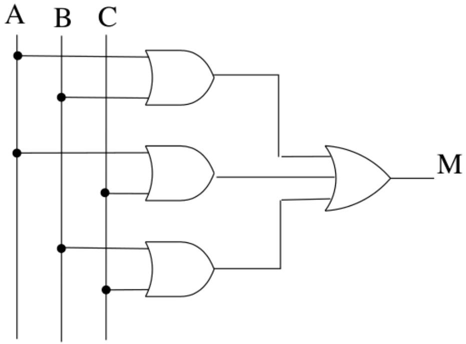
A	B	C	M
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

La forme normale disjonctive de cette fonction est :

$$\bar{A}.B.C + A.\bar{B}.C + A.B.\bar{C} + A.B.C$$

Example de circuits combinatoire: Le circuit "Majorité"

En utilisant un tableau de Karnaugh, on obtient la forme simplifiée suivante : $AB + BC + CA$
Le circuit correspondant est le suivant :



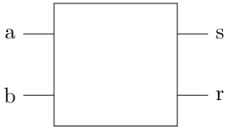
Circuits combinatoires:

- Les additionneurs
- Les décodeurs
- Les multiplexeurs et les démultiplexeurs

Les additionneurs: Le demi-additionneur

- Un demi-additioneur est un circuit qui prend en entrée deux bits (a et b) et qui produit la somme (addition) de ces deux nombres s et la retenue éventuelle r.

- Alors, entrées : a et b; et sorties : s la somme et r la retenue.



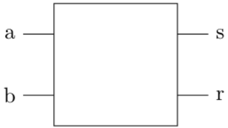
a	b	s	r
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$s = a \oplus b$$
$$r = a.b$$

Les additionneurs: Le demi-additionneur

- Un demi-additionneur est un circuit qui prend en entrée deux bits (a et b) et qui produit la somme (addition) de ces deux nombres s et la retenue éventuelle r.

- Alors, entrées : a et b; et sorties : s la somme et r la retenue.



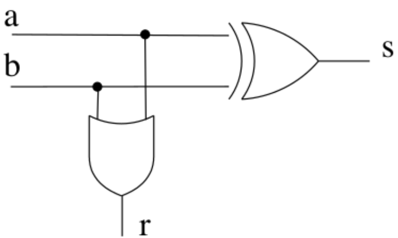
a	b	s	r
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$s = a \oplus b$
$r = a.b$

Les additionneurs: Le demi-additionneur

- Un demi-additionneur est un circuit qui prend en entrée deux bits (a et b) et qui produit la somme (addition) de ces deux nombres s et la retenue éventuelle r.

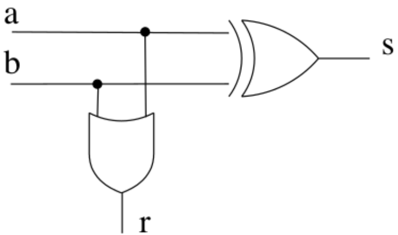
- Voici le logigramme correspondant:



Les additionneurs: Le demi-additionneur

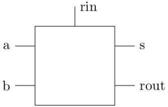
- Un demi-additionneur est un circuit qui prend en entrée deux bits (a et b) et qui produit la somme (addition) de ces deux nombres s et la retenue éventuelle r.

- Voici le logigramme correspondant:



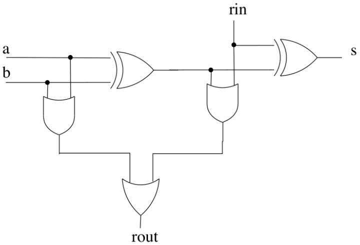
Le additionneur

- C'est un additionneur qui prend en compte la retenue précédente.
- Exemple: Additionneur Complet (2bits)



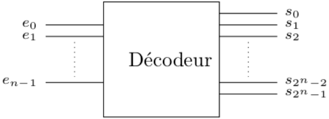
a	b	rin	s	$rout$
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

- Voici le logigramme correspondant:



Le décodeur

Un décodeur décode un nombre codé en binaire en activant la ligne correspondant à ce nombre. Il comprend n entrées et 2^n sorties. La $i$ème sortie de décodeur vaut 1 si les n entrées forment l'entier i, *ie* $(e_n e_{n-1} \dots e_1 e_0)_2 = (i)_{10}$.



un décodeur 2 vers 4 : Voici un décodeur 2 vers 4 (2 entrées - 4 sorties) représenté par sa table de vérité (à gauche) et les expressions booléennes de chacune des sorties (à droite).

e_1	e_0	s_3	s_2	s_1	s_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

$$s_0 = \overline{e_0}.\overline{e_1}$$
$$s_1 = \overline{e_1}.e_0$$
$$s_2 = e_1.\overline{e_0}$$
$$s_3 = e_0.e_1$$

Multiplexeur (Mux)

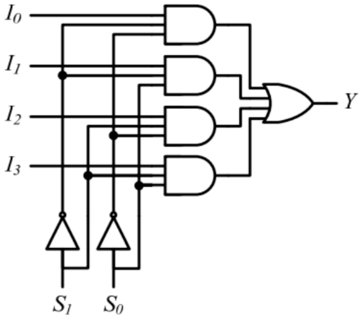
- Le multiplexeur est un circuit combinatoire qui comporte 2^n entrées et une seule sortie.
- Il permet de choisir une entrée parmi les 2^n et de la recopier sur la sortie.
- Pour la sélection de l'entrée, il y a n entrées de sélection qui codent le numéro en binaire de l'entrée voulue.
- L'intérêt du multiplexeur est de pouvoir connecter plusieurs entrées à un même circuit et de sélectionner l'entrée voulue simplement en indiquant son adresse sur les lignes de commande.

Multiplexeur 4 vers 1

- 4 entrées I_0, I_1, I_2, I_3
- 2 entrées de sélection S_1, S_0
- 1 sortie Y

<u>Table de vérité</u>

S1	S0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3



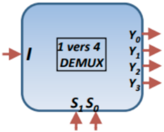
$$Y = \overline{S_1}.\overline{S_0}.I_0 + \overline{S_1}.S_0.I_1 + S_1.\overline{S_0}.I_2 + S_1.S_0.I_3$$

21

Démultiplexeur (Demux)

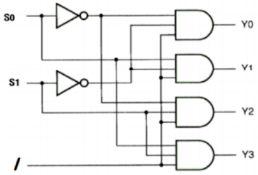
- Un démultiplexeur est un aiguilleur à une entrée de donnée.
- Il contient:
  - Une entré de donnée
  - n entrées d'adresse (bits de sélection)
  - 2^n sorties.
- La valeur de l'entrée se retrouve sur la sortie dont le numéro est codé par l'adresse.
- Dans cette fonction le circuit joue le rôle inverse du multiplexeur.

Multiplexeur 4 vers 1



S_1	S_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	*I*
0	1	0	0	*I*	0
1	0	0	*I*	0	0
1	1	*I*	0	0	0

$$Y_0 = \overline{S_1} \cdot \overline{S_0} \cdot I$$
$$Y_1 = \overline{S_1} \cdot S_0 \cdot I$$
$$Y_2 = S_1 \cdot \overline{S_0} \cdot I$$
$$Y_3 = S_1 \cdot S_0 \cdot I$$



Circuits séquentiels

- Ce sont ceux où les sorties à l'instant t dépendent non seulement des entrées à l'instant t mais aussi des entrées précédentes ou de leur état interne.
- **Le temps** est donc un paramètre des circuits séquentiels.
- Ils permettent de stocker une information au cours du temps et **sont donc l'élément principal des mémoires.**
- Ils sont souvent synchrones: leurs sorties ne changent que sur autorisation d'un signal particulier qui marque les instants significatifs dans le temps.
- Les circuits logiques séquentiels sont bâtis au moyen des circuits combinatoires et des éléments, basiques, de mémorisation appelés bascules.

Circuits séquentiels

- Ce sont ceux où les sorties à l'instant t dépendent non seulement des entrées à l'instant t mais aussi des entrées précédentes ou de leur état interne.
- **Le temps** est donc un paramètre des circuits séquentiels.
- Ils permettent de stocker une information au cours du temps et **sont donc l'élément principal des mémoires.**
- Ils sont souvent synchrones: leurs sorties ne changent que sur autorisation d'un signal particulier qui marque les instants significatifs dans le temps.
- Les circuits logiques séquentiels sont bâtis au moyen des circuits combinatoires et des éléments, basiques, de mémorisation appelés bascules.

Circuits séquentiels

- Ce sont ceux où les sorties à l'instant t dépendent non seulement des entrées à l'instant t mais aussi des entrées précédentes ou de leur état interne.
- **Le temps** est donc un paramètre des circuits séquentiels.
- Ils permettent de stocker une information au cours du temps et **sont donc l'élément principal des mémoires.**
- Ils sont souvent synchrones: leurs sorties ne changent que sur autorisation d'un signal particulier qui marque les instants significatifs dans le temps.
- Les circuits logiques séquentiels sont bâtis au moyen des circuits combinatoires et des éléments, basiques, de mémorisation appelés bascules.

Circuits séquentiels

- Ce sont ceux où les sorties à l'instant t dépendent non seulement des entrées à l'instant t mais aussi des entrées précédentes ou de leur état interne.
- **Le temps** est donc un paramètre des circuits séquentiels.
- Ils permettent de stocker une information au cours du temps et **sont donc l'élément principal des mémoires.**
- Ils sont souvent synchrones: leurs sorties ne changent que sur autorisation d'un signal particulier qui marque les instants significatifs dans le temps.
- Les circuits logiques séquentiels sont bâtis au moyen des circuits combinatoires et des éléments, basiques, de mémorisation appelés bascules.

Circuits séquentiels

- Ce sont ceux où les sorties à l'instant t dépendent non seulement des entrées à l'instant t mais aussi des entrées précédentes ou de leur état interne.
- **Le temps** est donc un paramètre des circuits séquentiels.
- Ils permettent de stocker une information au cours du temps et **sont donc l'élément principal des mémoires.**
- Ils sont souvent synchrones: leurs sorties ne changent que sur autorisation d'un signal particulier qui marque les instants significatifs dans le temps.
- Les circuits logiques séquentiels sont bâtis au moyen des circuits combinatoires et des éléments, basiques, de mémorisation appelés bascules.

Circuits séquentiels

- De même que les portes logiques constituent les briques de base des circuits combinatoires, les circuits séquentiels sont construits à partir de cellules élémentaires servant à mémoriser un bit, connues sous la dénomination générique de bascules, car elles possèdent deux états stables.
- Une bascule est un circuit pouvant prendre deux états logiques 0 et 1. L'état de la bascule peut être modifiée en agissant sur une ou plusieurs entrées.
- Le nouvel état de la bascule dépend de l'état précédent. La bascule peut conserver sont état pendant une durée, c'est pour ça que elle peut être utilisée comme mémoire.
  - La bascule RS (asynchrone et synchrone)
  - La bascule D
  - La bascule JK

Circuits séquentiels

- De même que les portes logiques constituent les briques de base des circuits combinatoires, les circuits séquentiels sont construits à partir de cellules élémentaires servant à mémoriser un bit, connues sous la dénomination générique de bascules, car elles possèdent deux états stables.
- Une bascule est un circuit pouvant prendre deux états logiques 0 et 1. L'état de la bascule peut être modifiée en agissant sur une ou plusieurs entrées.
- Le nouvel état de la bascule dépend de l'état précédent. La bascule peut conserver sont état pendant une durée, c'est pour ça que elle peut être utilisée comme mémoire.
  - La bascule RS (asynchrone et synchrone)
  - La bascule D
  - La bascule JK

Circuits séquentiels

- De même que les portes logiques constituent les briques de base des circuits combinatoires, les circuits séquentiels sont construits à partir de cellules élémentaires servant à mémoriser un bit, connues sous la dénomination générique de bascules, car elles possèdent deux états stables.
- Une bascule est un circuit pouvant prendre deux états logiques 0 et 1. L'état de la bascule peut être modifiée en agissant sur une ou plusieurs entrées.
- Le nouvel état de la bascule dépend de l'état précédent. La bascule peut conserver sont état pendant une durée, c'est pour ça que elle peut être utilisée comme mémoire.
    - La bascule RS (asynchrone et synchrone)
    - La bascule D
    - La bascule JK

Bascule RS asynhrone

Le fonctionnement de la bascule RS peut être résumé comme suit : mettre S à 1 (et R à 0) met la sortie à 1 (set) tandis que mettre R à 1 (et S à 0) met la sortie à 0 (reset). Lorsque les deux entrées sont à 0, la bascule restitue en sortie la dernière action mémorisée sur la sortie (set ou reset).

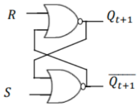
Table de verité

S	R	Q_t	Q_{t+1}	$\bar{Q}_{t+1}$	
0	0	0	0	1	Garder l'état
0	0	1	1	0	précédent
1	0	0	1	0	Mise à 1
1	0	1	1	0	
0	1	0	0		Mise à 0
0	1	1	0		
1	1	0	×	×	Indéterminé
1	1	1	×	×	

Equation logique:

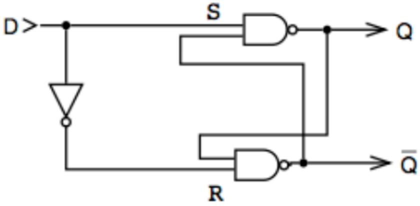
Q_t \ RS	00	01	11	10
0	**0**	**1**	×	**0**
1	**1**	**1**	×	**0**

Représentation avec les portes NOR



Bascule D

Une bonne façon de résoudre l'ambiguité propre à la bascule RS consiste à faire en sorte que l'état indéfini ne soit jamais présenté à l'entrée de la bascule.
C'est l'idée de la bascule D qui ne dispose que d'une seule entrée :



On a $S = D$ et $R = \overline{D}$ ainsi les valeurs de S et R sont toujours complémentaires.

Bascule RS Synchrone

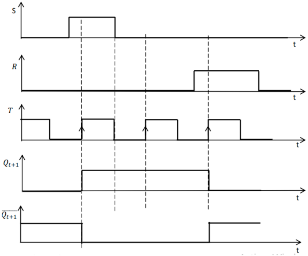
C'est une bascule RS dont la prise en compte de l'état des entrées est synchronisée par une impulsion d'horloge.

• Lorsque H = 0 ou R=S=0 il y a mémorisation de l'état précédent.

Table de vérité à gauche et chronogramme à droite:

Entrées			Sorties	
R	S	Q_n	Q_{n+1}	$\overline{Q_{n+1}}$
0	0	0	0	1
0	0	1	1	0
0	1	0	1	0
0	1	1	1	0
1	0	0	0	1
1	0	1	0	1
1	1	0	-	-
1	1	1	-	-

Table de vérité de la bascule RS



- On considère que l'horloge passe alternativement et de manière périodique d'un niveau haut (1) à un niveau bas (0).
- On supposera que ce passage est instantané (ces passages sont appellés front).
- De plus, on suppose généralement que les temps passés au niveau haut et au niveau bas sont égaux.
- fréquence = 1 / période

Bascule JK

Le fonctionnement est synchrone à une entrée d'horloge H et:

- Pour J = K = 0, il y a conservation du dernier état logique Qn-1 indépendamment de l'horloge : état mémoire.
- Pour J = K = 1, le système bascule à chaque front d'horloge.
- Pour J différent de K, la sortie Q recopie l'entrée J et la sortie non Q recopie l'entrée K à chaque front d'horloge.

Table de vérité

J	K	Q_n
0	0	Q_{n-1}
0	1	**0**
1	0	**1**
1	1	$\overline{Q_{n-1}}$

Table de vérité alternative

Q_{n-1}	J	K	Q_n	remarque
0	**0**	**X**	0	pour que la sortie reste à 0, il faut que J soit à 0, peu importe K.
0	**1**	**X**	1	pour que la sortie passe de 0 à 1, il faut que J soit à 1, peu importe K.
1	**X**	**1**	0	pour que la sortie passe de 1 à 0, il faut que K soit à 1, peu importe J.
1	**X**	**0**	1	pour que la sortie reste à 1, il faut que K soit à 0, peu importe J.