

UNIVERSITÉ CHOUAIB DOUKKALI  
Ecole Supérieure de Technologie  
Sidi Bennour

**Cours: Informatique Industrielle**



## **Informatique Industrielle**

- Généralités
- Logique Combinatoire
- Logique Séquentielle
- Les interruptions
- Programmation en assembleur

## Définitions.

**L'informatique** est un domaine d'activité scientifique, technique et technologique concernant le traitement automatique de l'information, via un programme, par des machines: ordinateurs; systèmes embarqués, robots, automates, etc.

**L'industrie** est l'ensemble des activités socio-économiques tournées vers la production en série de bien. Elle sous-entend :

- une certaine subdivision du travail ;
- une notion d'échelle ;
- l'utilisation de machine, d'abord manuelles puis automatisées.

## Définitions.

Ainsi on définir **l'informatique industrielle** comme étant une branche de l'informatique appliquée qui couvre l'ensemble des techniques de conception et de programmation de systèmes informatisés à vocation industrielle qui ne sont pas des ordinateurs.

## Bref historique.

On peut résumer l'évolution de la technologie en quatre (4) grandes périodes :

1. **Génération 0** ; les *calculateurs mécaniques* (1642-1945) : La première machine est attribué à Pascal (addition et soustraction) améliorée ensuite par Leibniz (multiplication et division).
2. **Génération 1** ; les *tubes à vides* (1945-1955) : Avec l'avènement de l'électronique et l'apparition de la guerre mondiale, on se consacra au chiffage (cryptage et codage) et on aboutira à la construction du premier ordinateur électronique (**COLOSSUS**).

## Bref historique.

3. **Génération 2** ; les *transistors* (1955-1965) : Le prix Nobel de Physique de Bardeen, Brattain et Shockley fut délivré en 1956 pour l'invention en 1948 chez Bell Labs du transistor. Ceci révolutionna définitivement les ordinateurs.
4. **Génération 3** ; les *circuits intégrés* (1965-1980) : Noyce inventa le circuit intégré en 1958.
5. **Génération 4** ; les *VLSI* (1980- ?) : Les VLSI (Very Large Scale Integration) permettent, dans les années 80, l'intégration de milliers, puis de millions de transistors sur une puce, favorisant ainsi la miniaturisation et l'augmentation de la rapidité de traitement.

## Classification des systèmes automatisés.

Il existe deux types de systèmes automatisés :

1. Les **systèmes automatisés continus** (ou **Analogique**) pour asservir et/ou commander des grandeurs physiques de façon précise et sans aide extérieure (l'angle d'une fusée, la vitesse de rotation d'un lecteur CD, la position du bras d'un robot, le pilotage automatique d'un avion).
2. Les **systèmes automatisés à évènements discrets** (ou **Numérique**) pour les commandes en tout ou rien (les distributeurs automatiques, les ascenseurs, le montage automatique dans le milieu industriel, les feux de croisement). Ces systèmes se divisent en deux parties :

## Classification des systèmes automatisés.

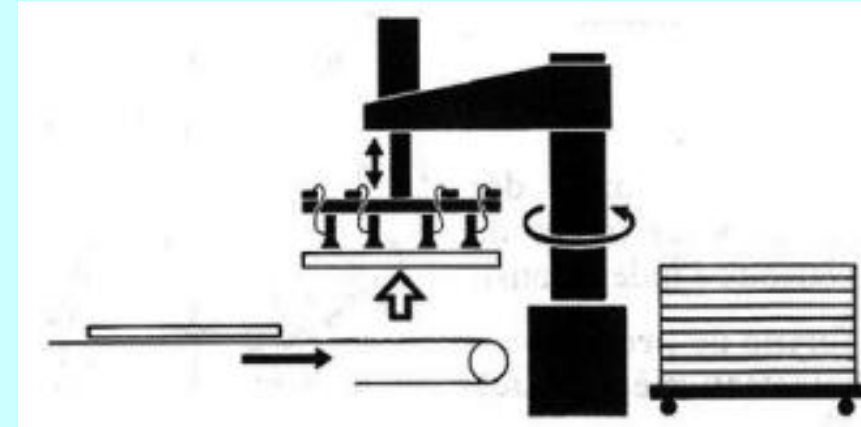
Ces systèmes se divisent en deux parties :

- 1) Les systèmes automatisés à logique câblée pour lesquels la commande est effectuée uniquement selon le câblage électrique du circuit.
  - Les systèmes automatisés combinatoires.
  - Les systèmes automatisés séquentiels.
- 2) Les systèmes automatisés programmables. Ces systèmes sont les plus répandus dans le domaine industriel. L'élément principal s'appelle l'Automate Programmable Industriel ou l'API. Le pilotage des actionneurs se fait selon le programme installé dans la mémoire de l'automate.



## Domaine d'application de l'informatique industrielle.

- ✓ Conditionnement sur palette après emballage.



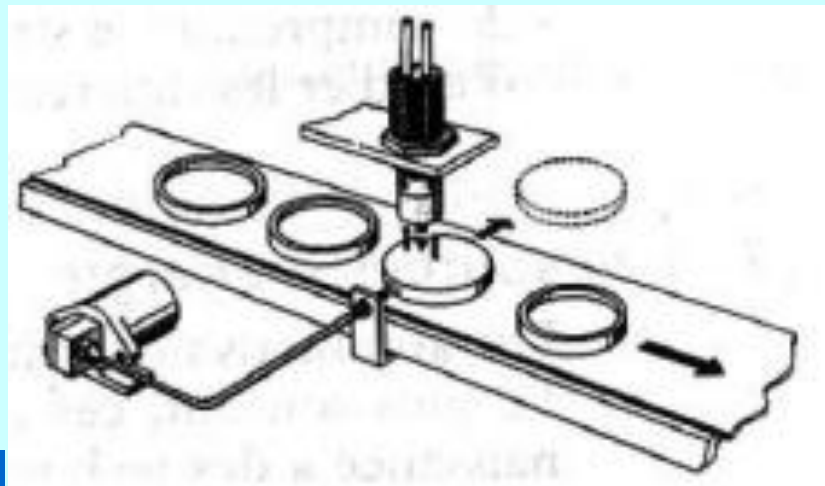
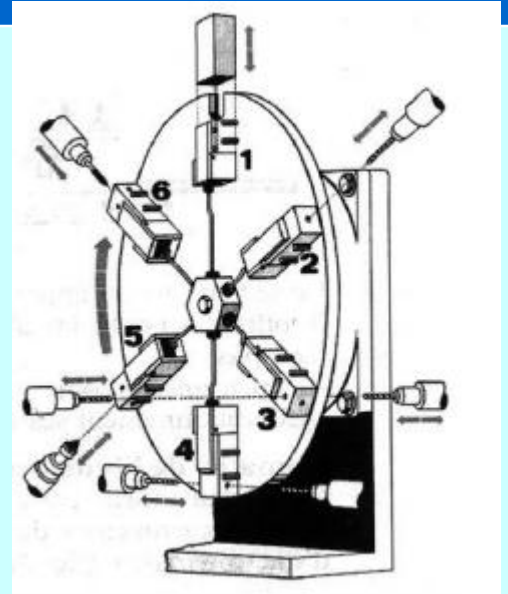
## Classification des systèmes automatisés.

- ✓ L'industrie automobile avec l'utilisation de robots industriels pour effectuer l'assemblage et la peinture des carrosseries.



## Classification des systèmes automatisés.

- ✓ Machine-outil dans les unités de perçage.
- ✓ Contrôle de produits : Détection de défauts en bout de chaîne de production.



## Avantage et inconvénient des systèmes automatisés.

### ❖ *Les avantages.*

- ✓ La capacité de production accélérée ;
- ✓ L'aptitude à convenir à tous les milieux de production ;
- ✓ La souplesse d'utilisation ;
- ✓ La réduction des coûts de production.
- ✓ La création de postes d'automaticiens.

### ❖ *Les inconvénients.*

- ✓ La complexité de la maintenance : elle doit être structurée ;
- ✓ La suppression d'emplois

## Objectifs :

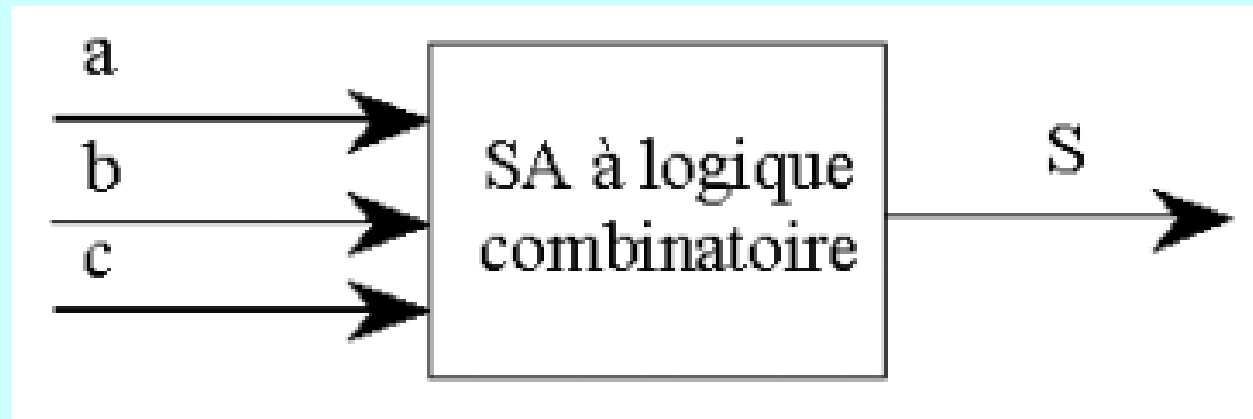
À la fin de cette unité, vous comprendrez le fonctionnement des principaux éléments d'un ordinateur : décaleur, additionneur, unité logique et arithmétique. Pour y arriver, vous devrez avoir atteint les objectifs suivants :

- décrire le fonctionnement et les propriétés des portes logiques, de circuits combinatoires simples tels que le décodeur, le multiplexeur et le démultiplexeur;
- utiliser les théorèmes et les identités de l'algèbre de Boole pour synthétiser un circuit à partir de sa table de vérité et simplifier le résultat obtenu.

## La Logique Combinatoire : Définition

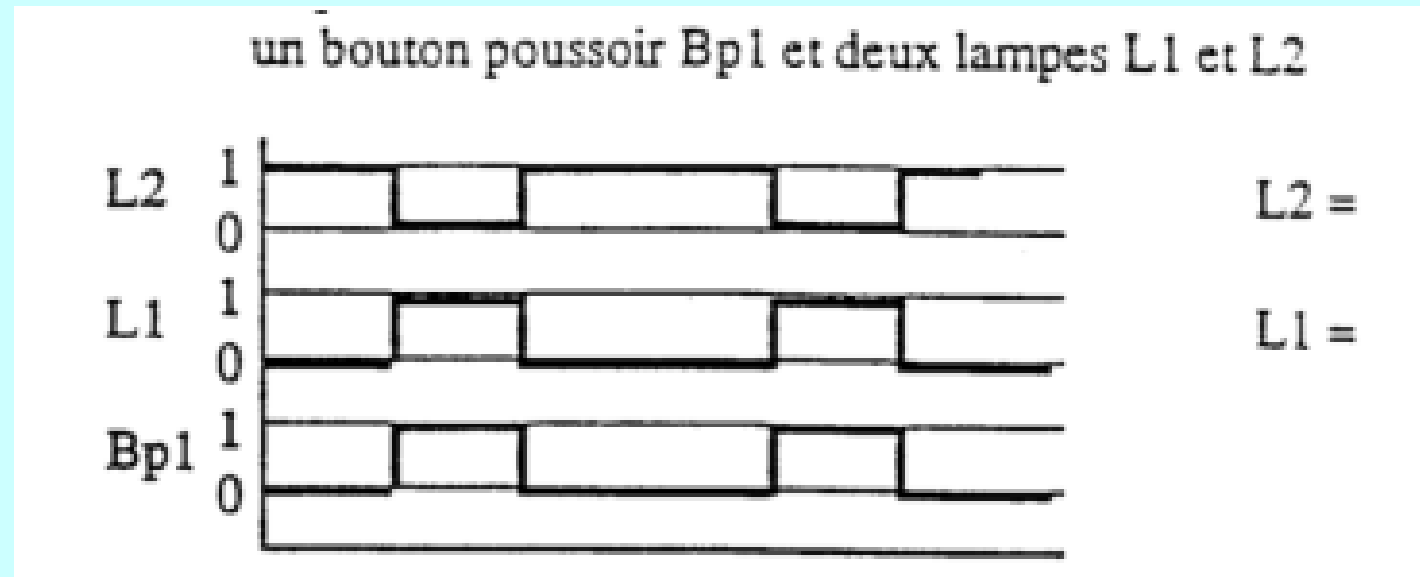
C'est une logique de combinaison de variable, c'est à dire que pour une combinaison d'entrées donnée, il ne correspond qu'une et une seule combinaison de sortie.

**Schéma :**



**Exemple d'application:** lorsque le conducteur d'un véhicule fait un appel de phares, les phares resteront allumés tant que le conducteur appuiera sur la commande de phares. Nous sommes donc en présence d'un système combinatoire.

### Chronogramme :



### Introduction: Variables et fonctions logiques

- Un système numérique complexe est réalisé à partir d'un assemblage hiérarchique d'opérateurs logiques élémentaires réalisant des opérations simples sur des variables logiques.
- Une variable logique est une variable qui ne peut avoir que deux états.
  - 1 et 0 en général

ou encore

- Vrai et Faux
- Fermé et Ouvert
- -5V et + 5V
- .....





## **Introduction: Variables et fonctions logiques**

Vous connaissez déjà de nombreux systèmes physiques qui travaillent à partir de grandeurs ne pouvant prendre que deux états:

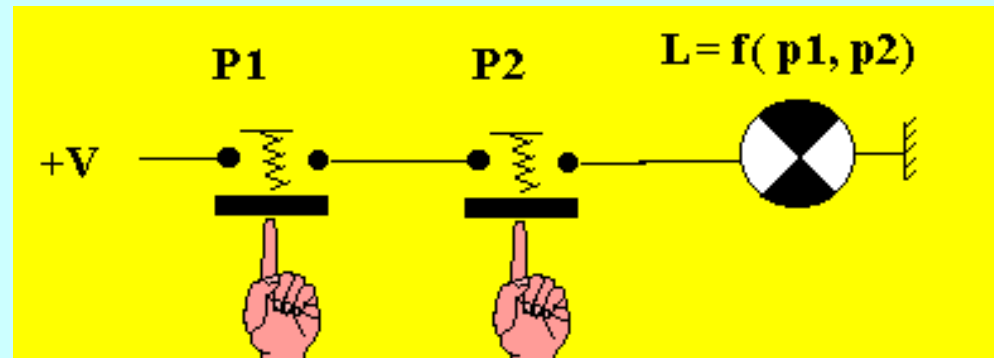
- Interrupteur ouvert ou fermé,
- Tension présente ou non,
- Lampe allumée ou non, objet éclairé ou non,
- Moteur en marche ou arrêté,

## Fonction Logique:

On appelle **fonction logique** (ou booléenne) une fonction définie sur  $2^n$  combinaisons de  $n$  variables logiques

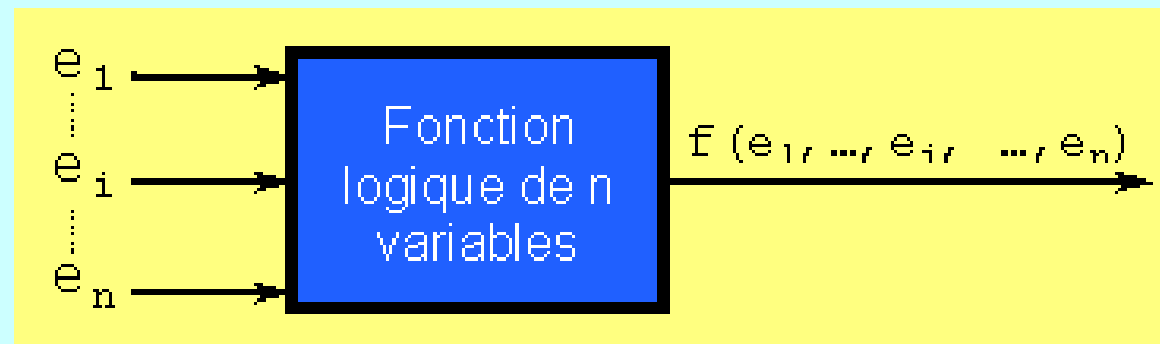
- Une fonction logique est donc une fonction de variables logiques
- Une fonction logique peut prendre 2 valeurs notées 0 et 1

**Exemple:**  $L$  (état de la lampe) est une fonction logique des variables  $p1$  et  $p2$  liées aux poussoirs



## Fonction Logique Combinatoire:

Une fonction logique est dite **combinatoire** lorsque l'état de la sortie est uniquement définie par la combinaison de l'état des variables logiques d'entrées quelque soit l'instant.



### Table de vérité

- Une **fonction logique** peut être représentée par une table donnant pour toutes les combinaisons des états des variables, l'état correspondante de la fonction.
- Elle comporte  **$2^n$**  lignes (ou  $n$  est le nombre de variable). Cette table est appelée table de vérité.
- Exemple de table de vérité:

a	b	f(a,b)

a	b	f(a,b)
0	0	
0	1	
1	0	
1	1	

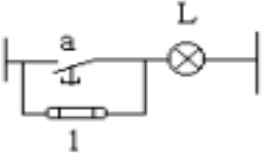
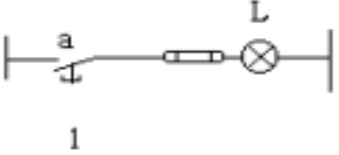
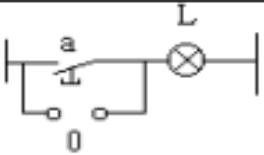
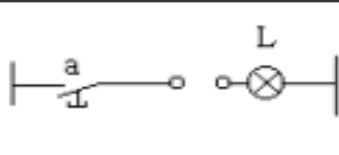
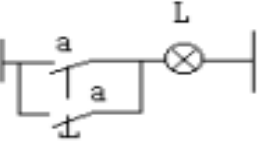
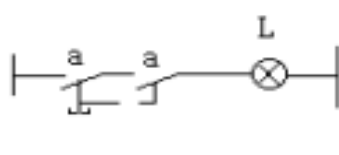
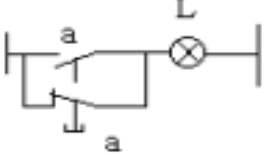
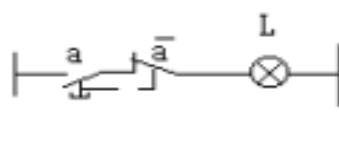
a	b	f(a,b)
0	0	0
0	1	0
1	0	0
1	1	1

## Equation logique:

- Une **fonction logique** peut s'exprimer algébriquement en utilisant **l'algèbre de Boole** c'est à dire par un groupe de variables reliées par des opérateurs logiques (NON, ET, OU)
- On définit tous les états où la fonction est égale à 1 par l'état de toutes les entrées.
- Exemple d'équation :

$$F = \bar{c}.b.\bar{a} + \bar{c}.b.a + c.\bar{b}.a + c.b.a$$

### Relations Caractéristiques De La Logique Booléenne: « algèbre de Boole »

Propriété des sommes logiques	Propriétés des produits logiques	Propriété de la complémentation
 $a + 1 = 1$	 $a . 1 = a$	$f(a) = /a$ Se dit <b>a barre</b> ou <b>non a</b>
 $a + 0 = a$	 $a . 0 = 0$	$/1 = 0$
 $a + a = a$	 $a . a = a$	$/0 = 1$
 $a + /a = 1$	 $a . /a = 0$	$//a = a$

### Equation logique:

$$a + a = a$$

$$a + 1 = 1$$

$$a + 0 = a$$

$$a + \bar{a} = 1$$

$$a . a = a$$

$$a . 1 = a$$

$$a . 0 = 0$$

$$a . \bar{a} = 0$$

### Propriétés des équations:

- Les termes entre parenthèses **sont prioritaires sur tout le reste** La fonction ET est **prioritaire** sur la fonction OU

Commutativité	$a + b = b + a$ $a . b = b . A$	
Associativité	$a . (b . c) = (a . b) . c$ $a + (b + c) = (a + b) + c$	
Distributivité de ./+	$a . (b + c) = (a . b) + (a . c)$	
Distributivité de +/-	$a + (b . c) = (a + b) . (a + c)$	
Simplification par absorption	$a . (a + b) = a$ $a + (a . b) = a$	$\underline{a} . (/a + b) = a . b$ $\underline{a} + (/a . b) = a + a$
Simplification par développement	$(a . b) + \bar{b} = a + b$	

### Théorèmes d'Augustus De Morgan :

- Le complément d'un produit logique de variables est égal à la somme logique des compléments de variables.

$$\overline{(a \cdot b)} = \bar{a} + \bar{b}$$

- Le complément d'une somme logique de variables est égal au produit logique des compléments de variables.

$$\overline{(a + b)} = \bar{a} \cdot \bar{b}$$

- Exercices d'application Simplifier à l'aide du théorème l'équation suivante :  $S = \overline{a \cdot b + c \cdot d}$



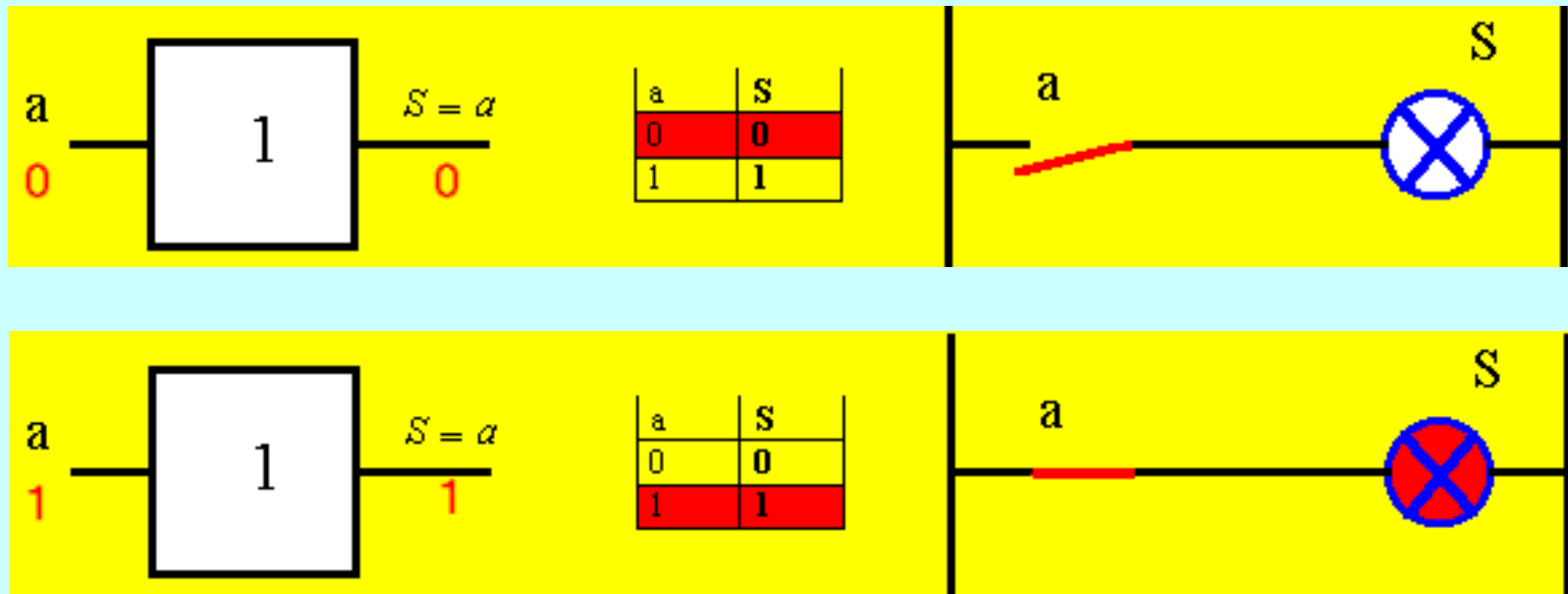
### Fonctions Logiques de base:

- OUI
- Equation :  $S=a$
- Symbole

*Logigrammes*

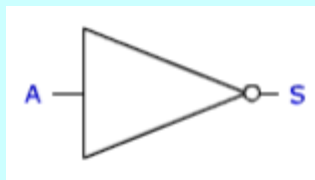
*Tables de vérité*

*Schémas électriques*



### Fonctions Logiques de base:

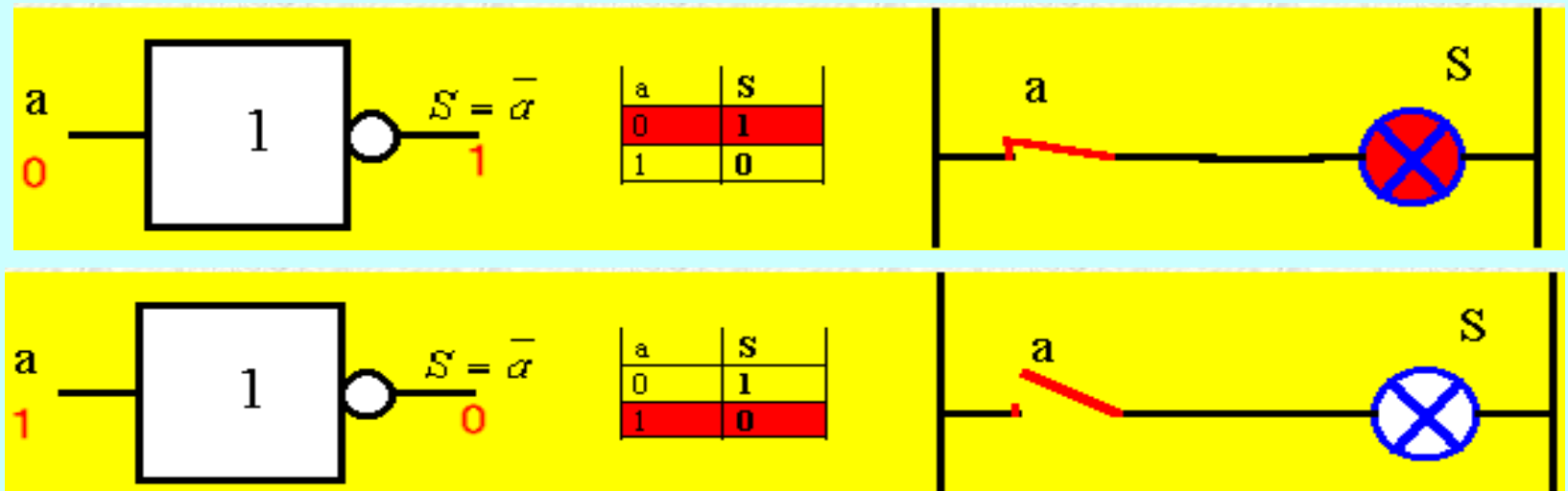
- Appellation: PAS, NON, NOT, complémentation
- Equation:  $S = \bar{a}$
- Symbole



Logigrammes

Tables de vérité

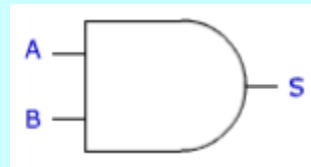
Schémas électriques



### Fonctions Logiques de base:

- Appellation: ET , AND , produit logique
- Notation: rien, . , &

- Symbole

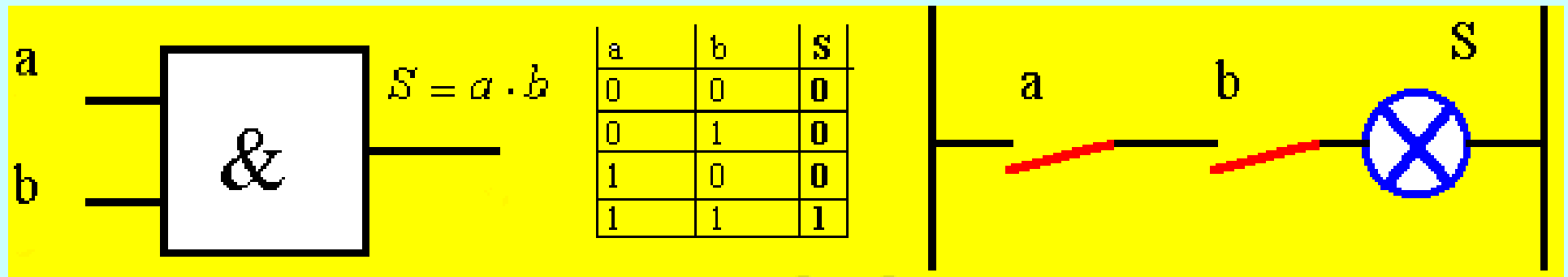


- Equation:  $S = a.b$

*Logigrammes*

*Tables de vérité*

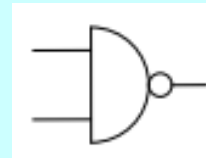
*Schémas électriques*



### Fonctions Logiques de base:

- **NONET, NOT AND et NAND**

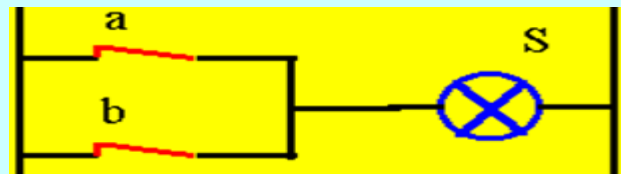
- ✓ **Symbole**



- ✓ **Logigrammes**



- ✓ **Schémas électriques**



- ✓ **Equation**

$$S = \overline{a \cdot b} = \overline{a} + \overline{b}$$

- ✓ **Tables de vérités**

a	b	S
0	0	1
0	1	1
1	0	1
1	1	0

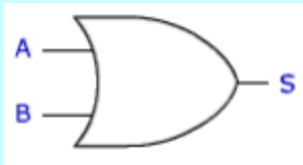
### Fonctions Logiques de base:

- Appellation: OU , OR , somme logique

- Notation: +

- Equation:  $S = a + b$

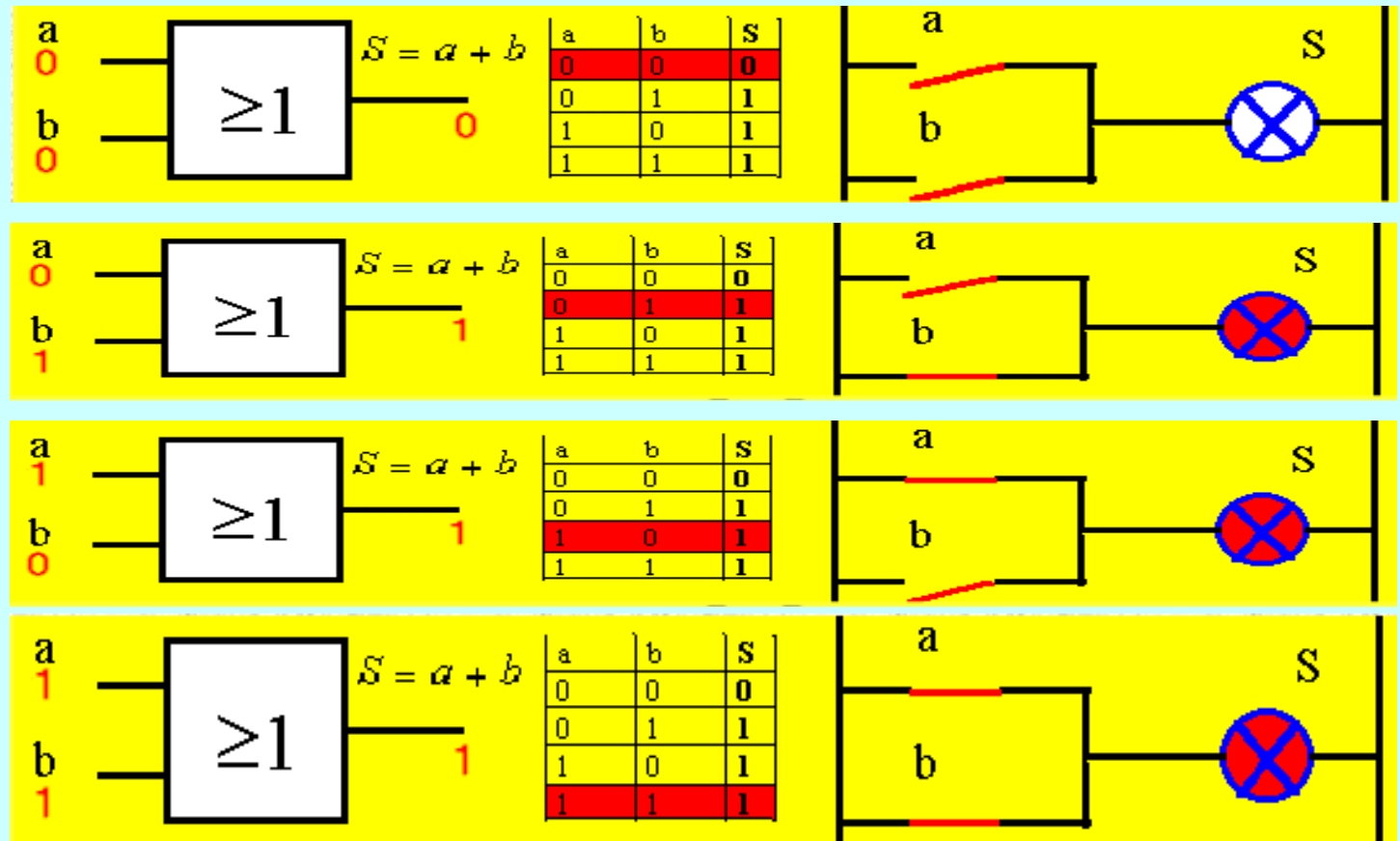
- Symbole:



Logigrammes

Tables de vérité

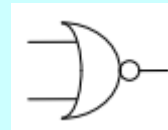
Schémas électriques



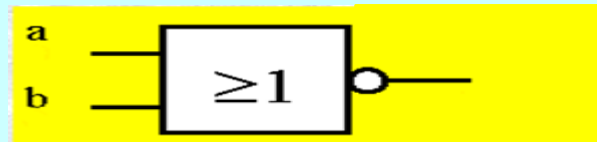
### Fonctions Logiques de base:

- NONOU, NOT OR et NOR

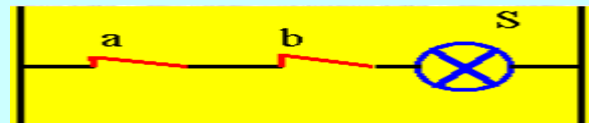
- ✓ **Symbole**



- ✓ **Logigrammes**



- ✓ **Schémas électriques**



- ✓ **Equation**

$$S = \overline{a + b} = \bar{a} \cdot \bar{b}$$

- ✓ **Tables de vérités**

a	b	S
0	0	1
0	1	0
1	0	0
1	1	0

### Fonctions Logiques de base:

#### ■ Xor

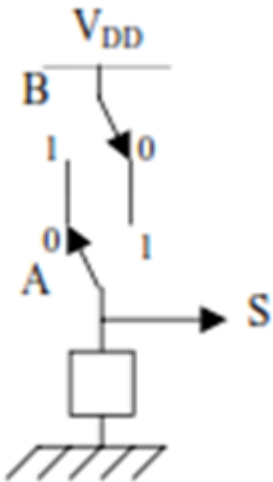


✓ **Symbole**

✓ **Logigrammes**

✓ **Schémas électriques**

✓ **Equation:  $S = A \oplus B$**

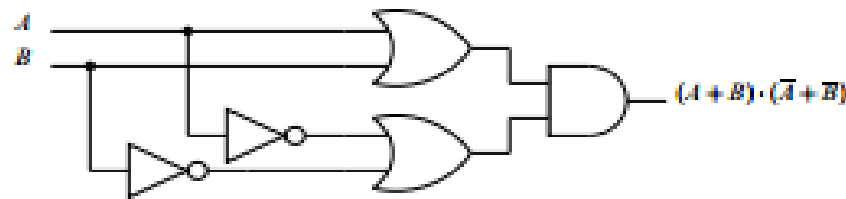
✓ **Tables de vérités**

Table de vérité	Montage	Symbole traditionnel	Symbole normalisé															
<table><tr><th>A</th><th>B</th><th><math>S = A \oplus B</math></th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	$S = A \oplus B$	0	0	0	0	1	1	1	0	1	1	1	0			
A	B	$S = A \oplus B$																
0	0	0																
0	1	1																
1	0	1																
1	1	0																

### Exercice d'application :

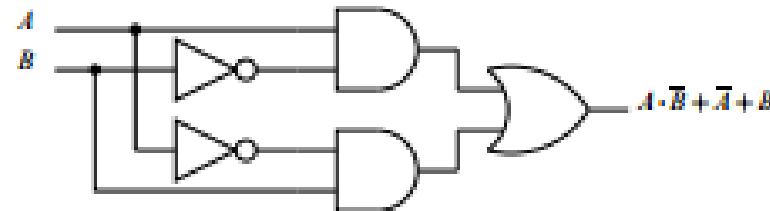
1. Exprimer la fonction **xor** comme un produit de sommes et réaliser le circuit logique correspondant.

**Correction :**  $A \oplus B = (A + B) \cdot (\overline{A} + \overline{B})$



2. Même question en exprimant **xor** comme une somme de produits.

**Correction :**  $A \oplus B = A \cdot \overline{B} + \overline{A} \cdot B$





3. Réaliser un circuit logique qui implémente la fonction F.

$$F = (A + B + C) \cdot (A + \overline{B} + \overline{C}) \cdot (\overline{A} + \overline{B} + \overline{C})$$

Correction: "pendant la séance de cours"

### Exercices d'application : 2

Simplifier les équations suivantes:  $S = a + a.b$

$$S = (a + a.b)(b + a.b) + a.b \quad a . b + a . b$$

$$S = c . (a.b + a . b) + a . b . c \quad c.(a.b) + a.b.c$$

### Exercices d'application : 3

#### Exercice 1:

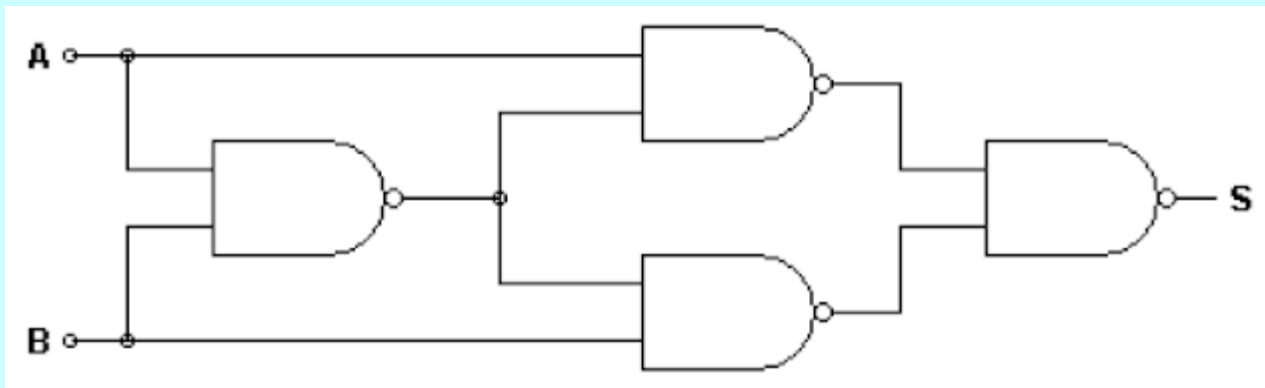
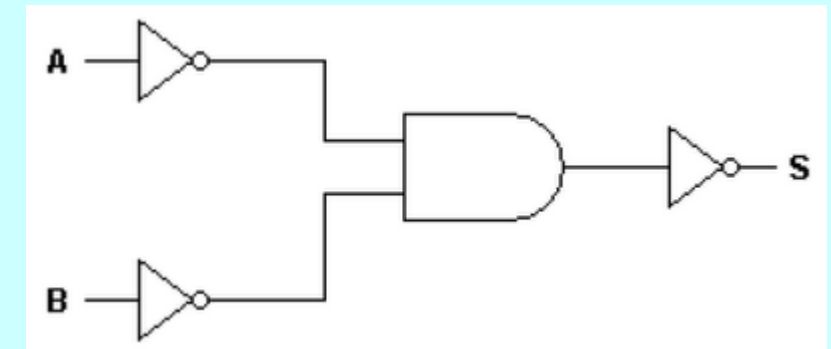
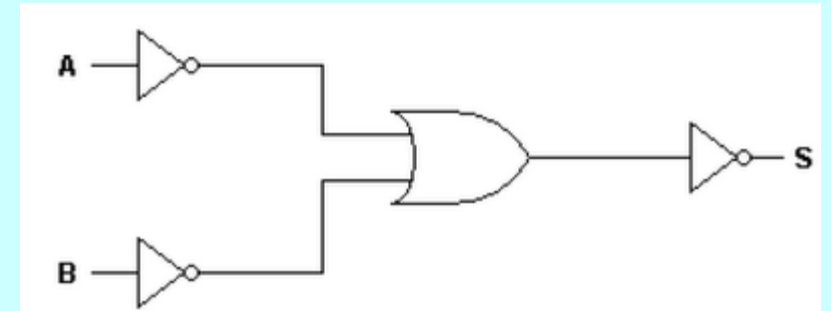
1) a. Déterminer l'équation du circuit de la figure suivante :

b. Dresser la table de vérité de ce circuit ?

c. Quelle est la fonction logique réalisée et quel est son symbole ?

2) Mêmes questions pour le circuit de la figure suivante ?

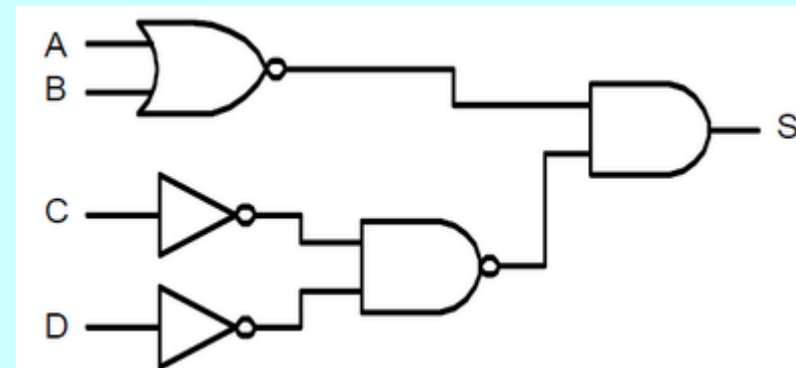
3) Mêmes questions pour le circuit de la figure suivante ?



### Exercices d'application : 3

#### Exercice 2:

1) Déterminer l'équation du circuit de la figure suivante :



2) Transformez le circuit ci-dessus en portes NON-ET à deux entrées. ( à faire)

# *Informatique Industrielle*

## *Représentation des nombres « Codage »*

### **Objectifs:**

- ✓ Nous verrons comment les instructions et les données sont représentées dans l'ordinateur.
- ✓ Nous apprendrons à représenter les nombres dans différentes bases.
- ✓ Nous verrons un codage permettant de corriger un codage.

# *Informatique Industrielle*

## *Représentation des nombres « Codage »*

### **Introduction:**

- Un des principes fondamentaux de l'informatique est que : Toute information (instructions ou donnée) est représentée dans un ordinateur par des nombres.
- Ces nombres sont des nombres binaires.
- Une information élémentaire correspond à un chiffre binaire (0 ou 1) appelé bit.
- Le codage de l'information permet d'établir une correspondance entre la représentation externe de l'information et sa représentation binaire.

# *Informatique Industrielle*

## *Représentation des nombres « Codage »*

**Base d'un système:**

**La base d'un système numérique est le nombre de chiffre de l'ensemble.**

$$(N)_r = [(partie\ entière) , (fraction)]_r$$

$r \rightarrow$  base

$N \rightarrow$  nombre

$$Ex. = [124, 659]_{10}$$

### Notation juxtaposé :

Soit une base  $b$  contenant les chiffres :  $s_0, s_1, s_2, s_3, \dots, s_{b-1}$

Un nombre  $N$  s'écrit dans la base  $b$  comme suit:

$$N = (a_n \dots a_1 a_0, a_{-1} a_{-2} \dots a_{-m})_b \quad \text{avec } a_i \in \{s_0, s_1, s_2, s_3, \dots, s_{b-1}\}$$

La signification de cette écriture est :

$$N = (a_n b^n + \dots + a_2 b^2 + a_1 b^1 + a_0 b^0 + a_{-1} b^{-1} + a_{-2} b^{-2} + \dots + a_{-m} b^{-m})$$

Remarques:

Cette écriture s'appelle forme **polynomiale**.

Les chiffres avant la virgule forment la **partie entière**, ceux après la virgule forment la partie **fractionnaire** de  $N$ .



# Informatique Industrielle

## Représentation des nombres « Codage »

### Les principales bases:

	Décimal	Binaire	Octal	Hexadécimal	
Base	(10)	(2)	(8)	(10)	
Symboles	0 à 9	0 à 1	0 à 7	0 à F	
	0	0	0	0	B
P	1	1	1	1	C
R	2	10	2	2	D
O	3	11	3	3	E
G	4	100	4	4	F
R	5	101	5	5	
E	6	110	6	6	
S	7	111	7	7	
S	8	1000	10	8	
I	9	1001	11	9	
O	10	1010	12	A	
N					

### La base 2 ou système de numération binaire:

La base est  $b=2$ . les chiffres sont 0 et 1 et s'appellent bits ou chiffres binaires.

Un nombre  $N$  s'écrit dans la base 2 sous forme **polynomiale**:

$$N = a_n 2^n + \dots + a_2 2^2 + a_1 2^1 + a_0 2^0 + a_{-1} 2^{-1} + a_{-2} 2^{-2} + \dots + a_{-m} 2^{-m} \quad \text{avec } a_i \in \{0, 1\}$$

Utilité: c'est la base utilisée dans les circuits électriques numériques.

Exemple:

Soit  $N_2=1001$  ; trouver  $N_{10}$

$N_2=10101$  ; trouver  $N_{10}$

### La base 8 ou système de numération octale:

La base est  $b=8$ . les chiffres sont 0, 1, 2, 3, 4, 5, 6, 7.

Un nombre  $N$  s'écrit dans la base 8 sous forme **polynomiale**:

$$N = a_n 8^n + \dots + a_2 8^2 + a_1 8^1 + a_0 8^0 + a_{-1} 8^{-1} + a_{-2} 8^{-2} + \dots + a_{-m} 8^{-m} \quad \text{avec } a_i \in \{0, 1, 2, 3, 4, 5, 6, 7\}$$

Utilité : cette base est utile pour compresser l'écriture de la base 2.

Exemple:

Soit  $N_8 = 52$  ; trouver  $N_{10}$

$N_2 = 26$  ; trouver  $N_{10}$

### La base 16 ou système de numération décimale:

La base est  $b=10$ . les chiffres sont 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

Un nombre  $N$  s'écrit dans la base 10 sous forme **polynomiale**:

$$N = a_n 10^n + \dots + a_2 10^2 + a_1 10^1 + a_0 10^0 + a_{-1} 10^{-1} + a_{-2} 10^{-2} + \dots + a_{-m} 10^{-m} \quad \text{avec } a_i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

Utilité : ce système de numération est utilisé dans la vie courante.

Exemple:

Soit  $N_{10} = 526$  ; trouver  $N_2$

$N_{10} = 269$  ; trouver  $N_2$

### La base 16 ou système de numération hexadécimale:

La base est  $b=16$ . les chiffres sont 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

Un nombre N s'écrit dans la base 16 sous forme **polynomiale**:

$$N = a_n 16^n + \dots + a_2 16^2 + a_1 16^1 + a_0 16^0 + a_{-1} 16^{-1} + a_{-2} 16^{-2} + \dots + a_{-m} 16^{-m}$$

$$\text{avec } a_i \in \{ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F \}$$

Utilité : cette base est utile pour compresser l'écriture de la base 2.

Exemple:

Soit  $N_{16} = 24$  ; trouver  $N_{10}$

$N_{16} = 32$  ; trouver  $N_{10}$

# *Informatique Industrielle*

## *Représentation des nombres « Codage »*

### **Passage d'une base à une autre:**

#### **1. Passage d'une base $b$ à la base 10**

on applique directement l'écriture polynomiale



### 2. Passage de la base 10 à une autre base b

On traite les parties entière (PE) et fractionnaire (PF) séparément.

#### ▪Partie entière:

On divise le nombre décimal successivement par b, les restes obtenus forment les chiffres de l'écriture de ce nombre dans la base b.



Exemples:

- soit  $N_{10}=86$ ; trouver  $N_2 =$  
- soit  $N_{10}=183$ ; trouver  $N_8 =$  

#### ▪Partie fractionnaire:

On la multiplie successivement par b. les chiffres obtenus à gauche de la virgule forment le nombre dans la base b.

Exemples:

- soit  $N_{10} = 0,8125$ ; trouver  $N_2 =$  
- soit  $N_{10} = 0,8$ ; trouver  $N_8 =$  

# Informatique Industrielle

## Représentation des nombres « Codage »

### ▪ Nombre quelconque:

On juxtapose la partie entière et la partie fractionnaire:

$$N_b = PE_b , PF_b$$

Exemples:

- soit  $N_{10} = 86,8125$  ; trouver  $N_2$
- soit  $N_{10} = 183,8$  ; trouver  $N_8$



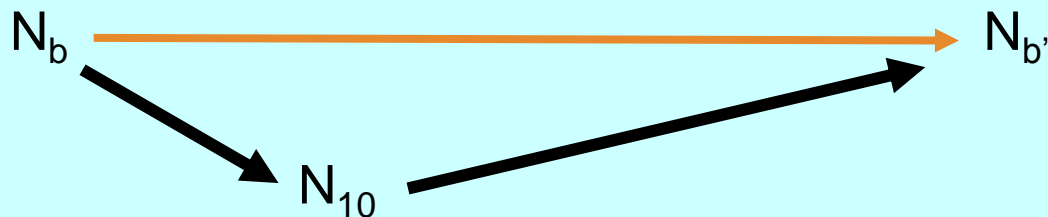
### Passage d'une base $b$ à une base $b'$

#### Règle générale:

On divise la partie entière et on multiplie la partie fractionnaire successivement par  $b'$ .

- **Inconvénient:** il faut faire les opérations dans le système de numérotation  $b$  (difficile).
- **Mieux**

On passe par la base 10.



- Exceptions

Pour passer de la base 2 aux bases 8 et 16 ou inversement, il y a une méthode plus simple.

# *Informatique Industrielle*

## *Représentation des nombres « Codage »*

### Passage de la base 2 à la base 8

- **Règle**

On regroupe les bits par 3 à partir de la virgule. On écrit ensuite chaque groupe en décimal.

- **Exemples:**

$N_2 = 1100011,1001101$  ; trouver  $N_8$ .

$N_2 = 1011010101,11$  ; trouver  $N_8$

# *Informatique Industrielle*

## *Représentation des nombres « Codage »*

### Passage de la base 2 à la base 16

- **Règle**

On regroupe les bits par 4 à partir de la virgule. On écrit ensuite chaque groupe en hexadécimal.

- **Exemples:**

$N_2 = 1100011,1001101$  ; trouver  $N_{16}$ .

$N_2 = 1011010101,11$  ; trouver  $N_{16}$

# *Informatique Industrielle*

## *Représentation des nombres « Codage »*

### Passage de la base 8 à la base 2

- **Règle**

On éclate des chiffres octaux en trois bits. Les zéros de la partie entière figurant complètement à gauche et ceux de la partie fractionnaire figurant complètement à droite peuvent être éliminés.

- **Exemples:**

$N_8 = 273,154$  ; trouver  $N_2$ .

## Passage de la base 16 à la base 2

- **Règle**

On éclate les chiffres hexadécimaux en quatre bits.

- **Exemples:**

$N_{16} = 2B,1C$  ; trouver  $N_2$ .