BADJI MOKHTAR – ANNABA UNIVERSITY
FACULTY OF EARTH SCIENCES
DEPARTMENT OF GEOLOGY

جامعة باجي مختار- عنابـــــــــة
كلية علوم الأرض
قسم الجيولوجيا

Représentation numérique de l'information

Dr BOUROUGA Moncef

I. Introduction

- Le traitement de l'information (Image, mot, nombre,…etc.) est effectué par l'ordinateur ou calculateur électronique. Les circuits électroniques qui les constituent ne peuvent prendre que deux états représentés par **0** et **1**. Donc, le langage utilisé dans ces machines est appelé système de numération binaire (**suite de 0 et de 1**).

II. Différents systèmes de numération

Il existe quatre systèmes de numération qui sont :

- Système décimal ;
- Système binaire ;
- Système octal ;
- Système hexadécimal.

Information encoding

Information encoding is the process of **converting** data into a specific format that can be efficiently processed, stored, and transmitted by **electronic systems**.

This transformation typically involves the use of **binary codes**, which represent information in the form of **0s** and **1s**.

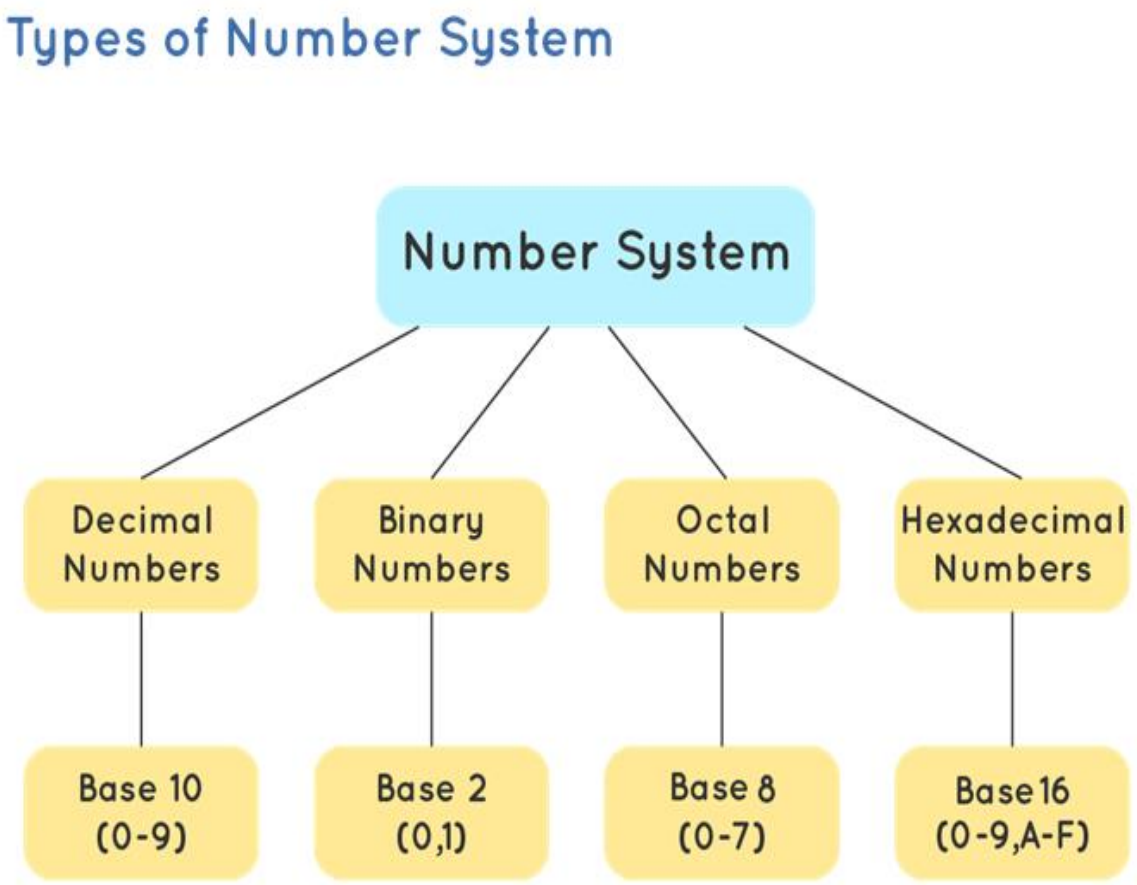
By encoding information, it becomes possible to facilitate communication between devices, ensuring that the **data** can be accurately **interpreted** and utilized by various applications.

This **process** is fundamental in computer science, telecommunications, and data management, as it allows for the effective handling of information across different platforms and technologies.

Different types of number systems

Types of number system

- **In computing, several number systems are used to represent and manipulate data.**
- **The most common types include:**
  1. **Binary Number System:**
  2. **Decimal Number System**
  3. **Octal Number System:**
  4. **Hexadecimal Number System:**



Different types of number systems

1. Binary Number System:

The fundamental system used in computers, consisting of only two digits: **0** and **1**. This **base-2 system** is essential for digital **electronics and computing**, as it aligns with **the on/off states of transistors**.

2. Decimal Number System:

The base-10 system that we use in everyday life, which includes the **digits 0-9**. It is the most familiar number system for **humans** and is often used in user **interfaces and applications**.

3. Octal Number System:

A **base-8 system** that uses digits from **0 to 7**. Octal is sometimes used in computing as a **shorthand** (abreviation) for binary, as each octal digit corresponds to three binary digits, making it more compact for certain applications.

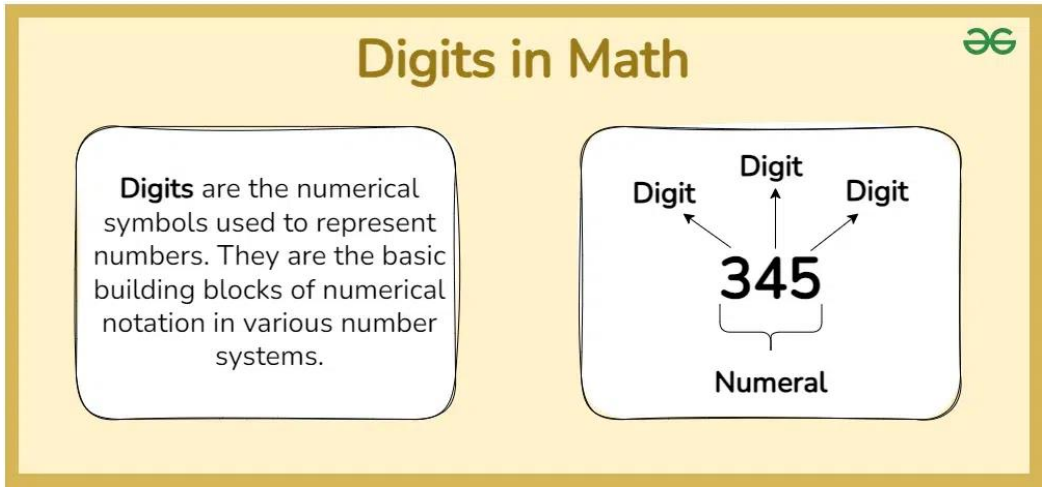
4. Hexadecimal Number System:

A **base-16 system** that includes digits **0-9** and letters **A-F** (representing values 10 to 15). Hexadecimal is widely used in **programming** and computer science because it can **represent large binary numbers more succinctly**, making it **easier** for **programmers** to **read and write code**.

Different types of number systems

Nom :	Symboles utilisés	Référence :
Binaire	0 1	2
Octal	0 1 2 3 4 5 6 7	8
Décimal	0 1 2 3 4 5 6 7 8 9	10
Hexadécimal	0 1 2 3 4 5 6 7 8 9 A B C D E F	16

Digits ''Les chiffres''



https://www.geeksforgeeks.org/digits/

Digits are the building block of mathematics, as all the numbers are made up of digits which can also be called numerals i.e., 0 to 9.

Les chiffres sont les éléments de base des mathématiques, car tous les nombres sont constitués de chiffres qui peuvent également être appelés numéraux, c'est-à-dire de 0 à 9

In math, each **digit** in a **number** holds a particular place value. Place value means the **worth of a digit** in a number based on **where it stands** in that number.

For instance, take the number 345. In this number, the digit 4 has a place value of 40 because it is in the tens place. Meanwhile, the digit 5 has a place value of 5 because it is in the unit place. The place value depends on the digit's position in the number, and it helps us understand how much that digit contributes to the total value of the number.

$$345 = 3x10^{2} + 4x10^{1} + 5x10^{0}$$

The Decimal System

- The decimal system, also known as the base -10 number system, is the most widely used numeration system in the world.
- It is based on ten digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9. Each digit's position in a number determines its value, which is a fundamental characteristic of positional numeral systems.

Key Features of the Decimal System:

- **Base 10:** The decimal system operates on a base of 10, meaning that each position in a number represents a power of ten.
- For example, in the number 345, the digit 3 is in the hundreds place (3×10^2), the digit 4 is in the tens place (4×10^1), and the digit 5 is in the units place (5×10^0).

5	**7**	**3**	**4**	**5**
↓	↓	↓	↓	↓
10^4	10^3	10^2	10^1	10^0

Le Système décimal

- C’est un système utilisé dans le monde extérieur, les chiffres utilisés sont des entiers composés par des valeurs de 0 à 9.
- **La base de ce système est 10.**

Exemple

1995 : ce nombre s’écrit dans le système **à** base 10 sous la forme :

$1x10^3 + 9\ x10^2 + 9\ x10^1 + 5\ x10^0$

Le Système binaire

- C’est le système qui est utilisé par la machine (**l’ordinateur**), il est composé seulement par des suites **binaires** (**1** et **0**).
- **La base de ce système est 2**.

Exemple

$(1001101)_2$: la valeur décimale de cette suite est :

$1x\ 2^6 + 0\ x\ 2^5 + 0\ x\ 2^4 + 1x\ 2^3 + 1x\ 2^2 + 0\ x\ 2^1 + 1x2^\circ = (77)_{10}$

Chaque chiffre binaire (0 ou 1) est appelé **bit** qu’on note par b, et toutes combinaisons binaires composées de huit bits est appelée **Octet**

Example: $(11011)_2 = (\ ?\ )_{10}$

The Binary System

The binary system, also known as the **base-2 number** system, is a positional numeral system that uses only two symbols: **0** and **1**.

This system is fundamental in **computing** and **digital electronics** because it aligns perfectly with the **on/off** states of **electronic circuits**.

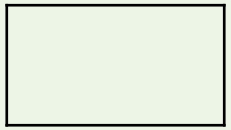
Each binary digit (**0** or **1**) is called a **bit**, denoted by b, and any binary combination composed of eight **bits** is called a **Byte**.

Notions of bit and Byte

In computing, bits and bytes are fundamental units of digital information.

Bit

A bit (short for binary digit) is the smallest unit of information in a computer. It can hold one of two values: 0 or 1.
This binary representation is essential for all digital data processing and storage



0 or 1

Byte

A byte is a collection of 8 bits. It is the standard unit used to represent a character in computer systems, such as a letter or a number. A byte can represent 256 different values (from 0 to 255), which is useful for encoding characters in various character sets

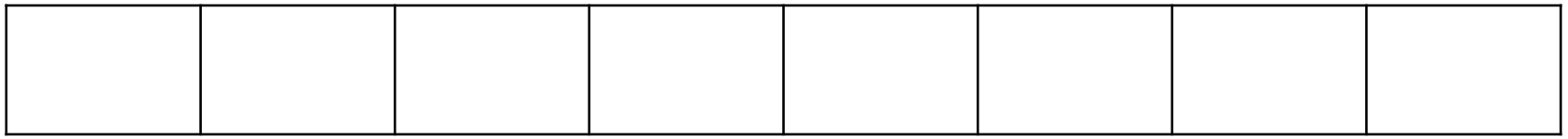
1	1	0	0	1	0	1	0

8 Bits = 1 Byte

La notion de bit et octet

- Un **bit** est l'unité de base de l'information en informatique et en communication numérique. Le terme "**bit**" est une contraction de "**binary digit**".

- **Valeurs binaires :** Un **bit** peut prendre l'une des deux valeurs possibles **0** ou **1**.



8 Bits = 1 Octet

Key Features of the Binary System:

- **Base 2**: In the binary system, each digit's position represents a power of two. For example, in the binary number **1011**:
  - The leftmost digit (1) represents $1 \times 2^3 = 8$.
  - The next digit (0) represents $0 \times 2^2 = 0$.
  - The next digit (1) represents $1 \times 2^1 = 2$.
  - The rightmost digit (1) represents $1 \times 2^0 = 1$.
  - Therefore, 1011 in binary equals 8+0+2+1=11 in decimal.

Bits: Each digit in a binary number is called a bit. For example, the binary number 1101 consists of four bits.
Applications: The binary system is used extensively in computer systems, programming, and digital communications. It allows for efficient data processing and storage, as computers operate using binary logic through electronic components like transistors and logic gates.

Le système octal

- 8 symboles sont utilisés dans ce système : {0, 1, 2, 3, 4, 5, 6, 7}
- 8 symbols are used in this system: {0, 1, 2, 3, 4, 5, 6, 7}
- Exemple 1:
- $(127)_8 = 1x8^2 + 2x8^1 + 7x8^0$

- Exemple 2:
- Le nombre 1289 n'existe pas dans la base 8 puisque les symboles 8 et 9 n'appartiennent pas à la base.
- ***The number 1289 does not exist in base 8, since the symbols 8 and 9 do not belong to the base***

Hexadecimal system

- ✓ This system is widely used in computer science to simplify the representation of binary data.
- ✓ hexadecimal uses 16 symbols: **0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F**.
- ✓ In this system, the letters A to F represent decimal values 10 to 15.

Why Use Hexadecimal?

Hexadecimal is particularly useful in computing because it allows for a more concise representation of binary values. For example, one byte (8 bits) can be represented by two hexadecimal digits, making it easier to read and write values.

Usage in Programming

Hexadecimal is often used in web development, particularly for defining colors in HTML. For example, the color red is represented by the hexadecimal code #FF0000, where FF represents the maximum value of red, and 00 for green and blue.

Decimal	Hexadecimal
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	A
11	B
12	C
13	D
14	E
15	F

Hexadecimal system

- $(17)_{16} = 1x16^1 + 7x16^0 = 16 + 7 = (23)_{10}$
- $(AB)_{16} = Ax16^1 + Bx16^0 = 10x16^1 + 11x16^0$
  $= 160 + 11 = (173)_{10}$

For example, in the hexadecimal number 2F3, the value can be calculated as:

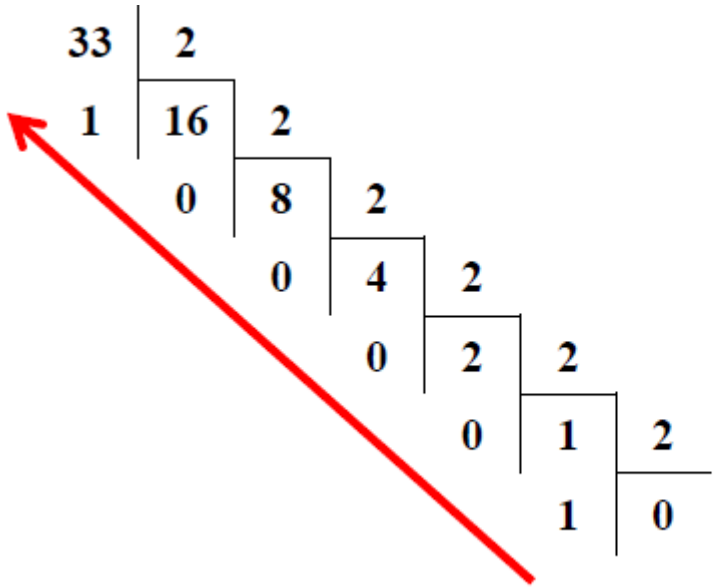
$2x16^2 + 15x16^1 + 3x16^0 = 755$ in decimal

Transformations (conversions)

1 Décimal / Binaire ,

Pour convertir un nombre décimal en un nombre binaire, diviser continûment le nombre décimal par la valeur 2, retenir la suite des restes de chaque division en commençant de **bas en haut** pour avoir le nombre équivalent.

Exemple :



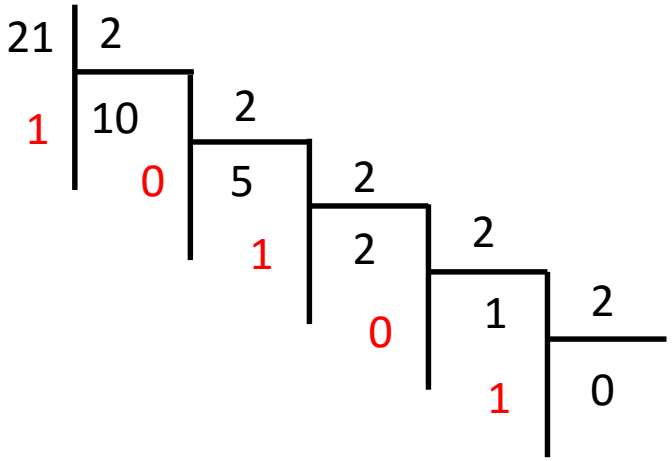
$(33)_{10} = (100001)_2$

Conversion of Decimal number into binary number

Division method:

Example $(21)_{10}$

1. Divid the number by 2
2. Put the **remainder** on right side
3. Keep dividing the quotient till we get the quotient less than 2
4. Repeat the division with the quotient until the quotient becomes 0.
5. The binary representation is obtained by reading the **remainders** from **bottom to top**.

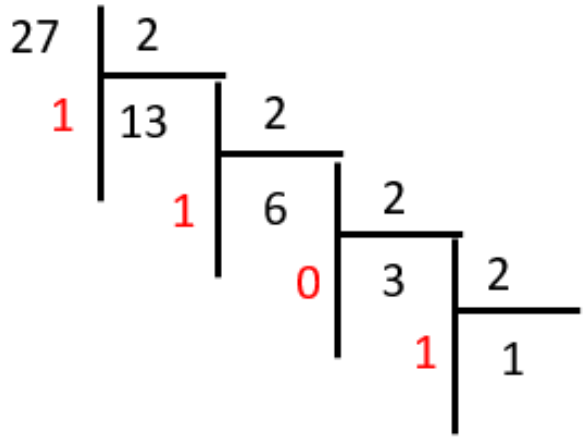


Record remainders from bottom to top to get the answer.

Answer :21_{10} = $(10101)_2$

Exercise:

- Convert the decimal number **27** to binary.



Now, we collect the remainders from the last division to the first:
Remainders (from last to first): 1, 1, 0, 1, 1
Thus, the binary representation of the decimal number 27 is 11011.

Conversion of binary number into Decimal number

Expansion method:

1. Multiply each digit with its place value
2. Add these place value
3. The sum is decimal number

Example $(11101)_2$

$= (1*2^4) + (1*2^3) + (1*2^2) + (0*2^1) + (1*2^0)$

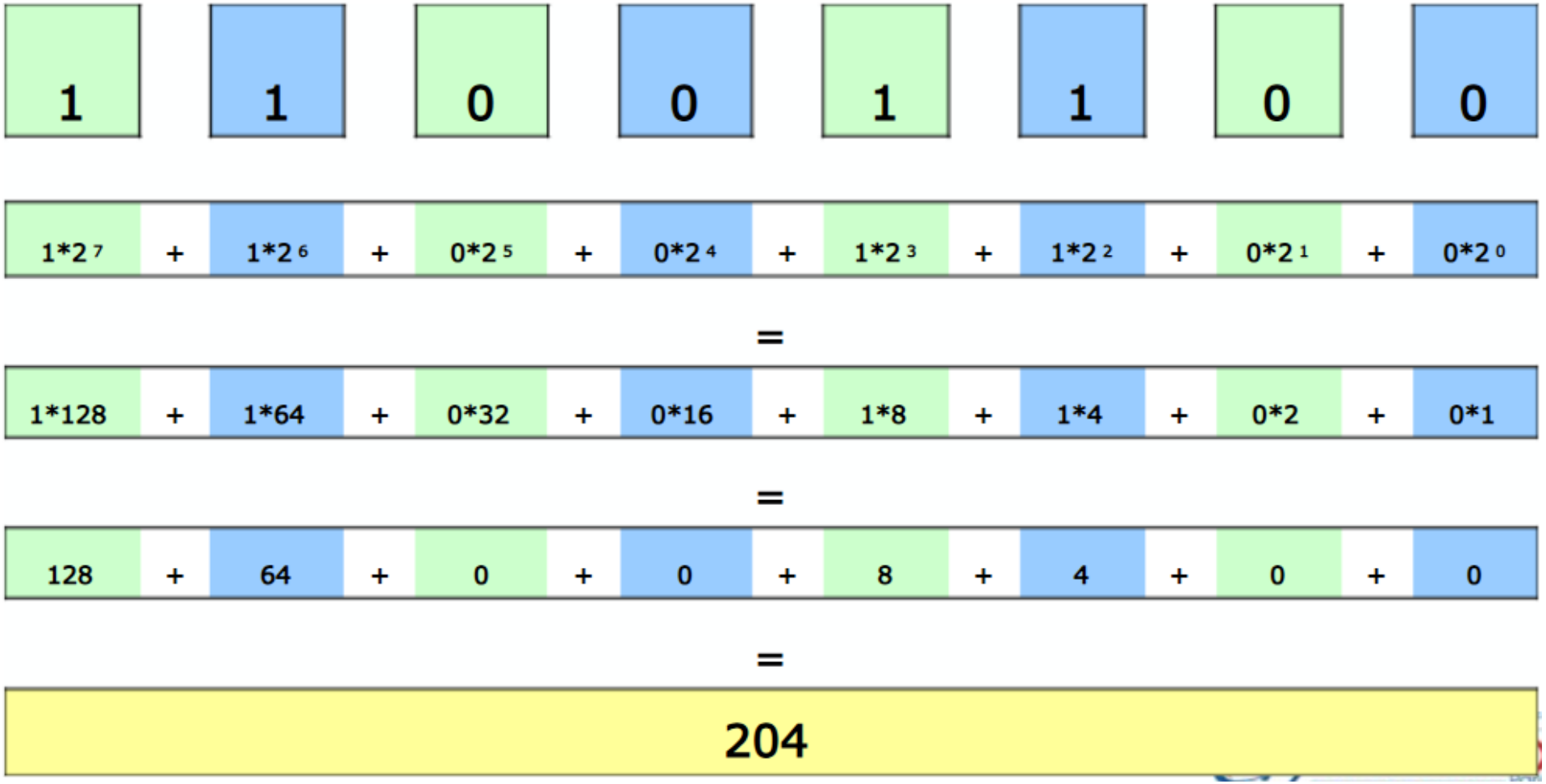
$= (1*16) + (1*8) + (1*4) + (0*2) + (1*1)$

$= 16+8+4+0+1 = \mathbf{29}$

Answer : $(11101)_2 = 29_{10}$

Conversion binaire vers décimal

11001100 en binaire à convertir en décimal



Exercice 1:

Convert the binary number 11010 to decimal.

Solution:

1. Write down the binary number
and assign powers of 2 from right to left:
1×2^4 (16)
1×2^3 (8)
0×2^2 (0)
1×2^1 (2)
0×2^0 (0)

2. Calculate the values:
   $1\times16= 16$
   $1\times8= 8$
   $0\times4= 0$
   $1\times2= 2$
   $0\times1= 0$

3. Add the results together:
   $16 + 8 + 0 + 2 + 0 = 26$

Thus, the binary number 11010 is equal to 26 in decimal.

Exercice 2:

Convert the binary number 101011 to decimal.

Solution:

1. Write down the binary number
and assign powers of 2 from right to left:
   1×2^5 (32)
   0×2^4 (0)
   1×2^3 (8)
   0×2^2 (0)
   1×2^1 (2)
   1×2^0 (1)

2. Calculate the values:
$1\times32=32$
$0\times16=0$
$1\times8=8$
$0\times4=0$
$1\times2=2$
$1\times1=1$

3. Add the results together:
   $32+0+8+0+2+1=4332+0+8+0+2+1=43$

Thus, the binary number `101011` is equal to **43** in decimal.

Activity

Convert $(11011)_2$ into its decimal equivalent

Answer should be $(27)_{10}$

Conversion un chiffre fractionné ” nombre à virgule flottante” (Converting a float number) :

- **1. Conversion de la partie entière :**
  - Divisez la partie entière du nombre par 2.
  - Notez le reste.
  - Répétez le processus avec le quotient jusqu'à ce que le quotient soit 0.
  - Lisez les restes à l'envers pour obtenir la représentation binaire de la partie entière.
- **Exemple :** Pour le nombre 12,5 :12 divisé par 2 donne 6, reste 0.
  - 6 divisé par 2 donne 3, reste 0.
  - 3 divisé par 2 donne 1, reste 1.
  - 1 divisé par 2 donne 0, reste 1.
  - **La partie entière 12 en binaire est donc 1100.**
- **2. Conversion de la partie fractionnaire :**
  - **Multipliez** la partie fractionnaire par 2.
  - Notez la partie entière du résultat (0 ou 1).
  - Répétez le processus avec la nouvelle partie fractionnaire jusqu'à ce que vous atteigniez la précision désirée ou que **la partie fractionnaire devienne 0**.

Exemple : Pour 0,5 :0,5 multiplié par 2 donne **1,0** (partie entière 1).

- La partie fractionnaire est maintenant 0, donc nous arrêtons ici.
- **Combinaison des résultats :**
  - Combinez les résultats des deux parties. Pour 12,5, cela donne 1100 (partie entière) et 1 (partie fractionnaire), donc 12,5 en binaire est **1100.1**.

Cette méthode vous permet de convertir n'importe quel nombre décimal avec virgule en binaire de manière systématique et précise .

Convertir le nombre décimal 21.25 en binaire,

1. Conversion de la partie entière (21) :

- 21 divisé par 2 donne 10, reste 1.
- 10 divisé par 2 donne 5, reste 0.
- 5 divisé par 2 donne 2, reste 1.
- 2 divisé par 2 donne 1, reste 0.
- 1 divisé par 2 donne 0, reste 1.

- En lisant les restes à l'envers, la partie entière **21** en binaire est **10101.2.**

2. Conversion de la partie fractionnaire (0.25) :

- 0.25 multiplié par 2 donne 0.5 (partie entière 0).
- 0.5 multiplié par 2 donne 1.0 (partie entière 1).

Nous arrêtons ici car la partie fractionnaire est maintenant 0.

3. Combinaison des résultats :

La partie entière est **10101** et la partie fractionnaire est **01**.

- Donc, **21.25** en binaire est **10101.01**.

Résultat final

21.25 en binaire est : 10101.01.

Hexadécimal	Binaire	Hexadécimal	Binaire
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

Converting Decimal Fractions to Binary

Step 1: Convert the Whole Number Part

- **Divide the whole number by 2** and record the quotient and the remainder.
- **Repeat the division** with the quotient until the quotient becomes 0.
- The binary representation of the whole number is obtained by reading the remainders from bottom to top.

Step 2: Convert the Fractional Part

- **Multiply the fractional part by 2**.
- The whole number part of the result (either 0 or 1) is the next binary digit.
- **Repeat the multiplication** with the new fractional part (the part after the decimal point).
- Continue this process until you reach the desired precision or the fractional part becomes 0.

Example: Convert 12.625 to Binary

- **Step 1: Convert the Whole Number (12)**
  - **12 ÷ 2** = 6 (remainder 0)
  - **6 ÷ 2** = 3 (remainder 0)
  - **3 ÷ 2** = 1 (remainder 1)
  - **1 ÷ 2** = 0 (remainder 1)

Reading the remainders from bottom to top gives us **1100**.

Whole number part: 1100

- **Step 2: Convert the Fractional Part (0.625)**
  - **0.625 × 2** = 1.25 → Whole number part: **1**
  - **0.25 × 2** = 0.5 → Whole number part: **0**
  - **0.5 × 2** = 1.0 → Whole number part: **1**

Fractional part: 101

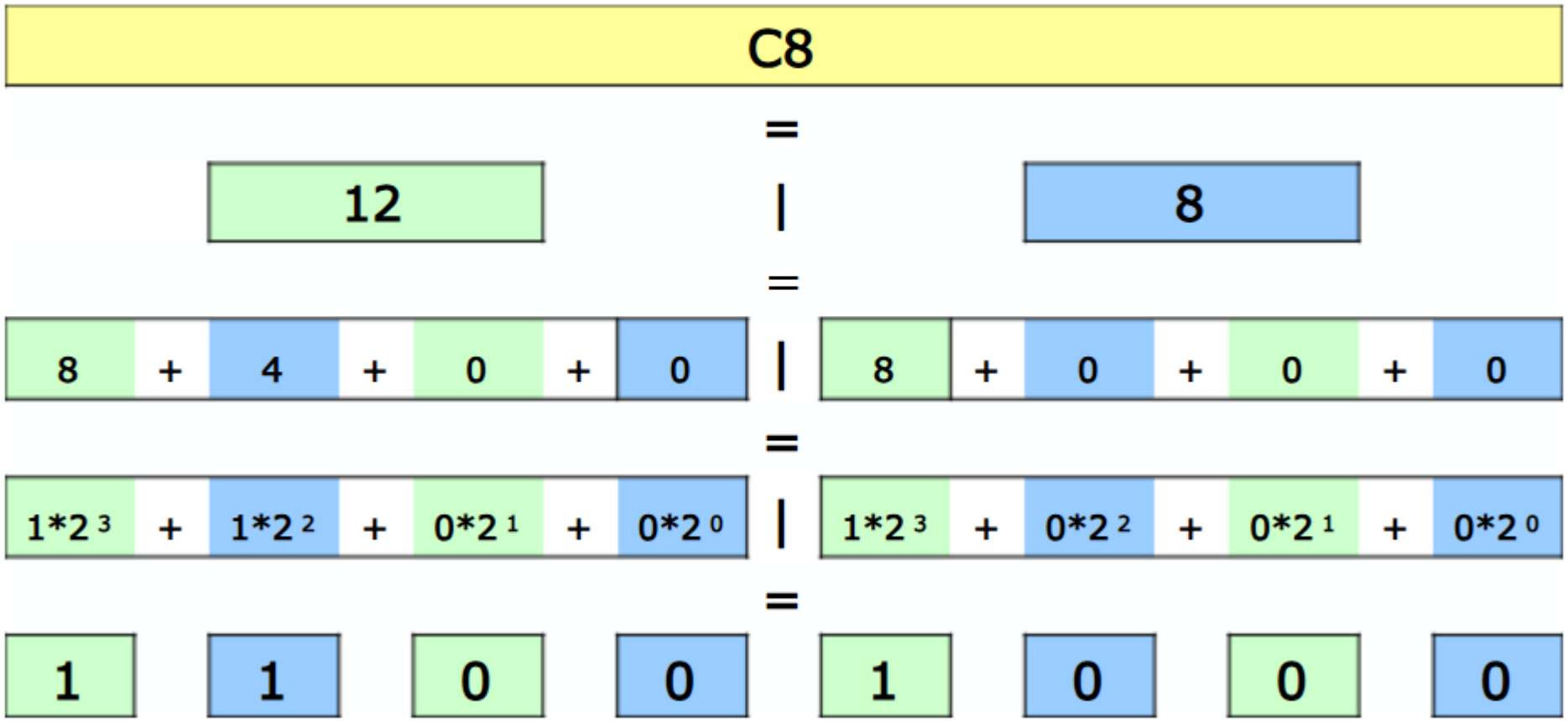
Now, we stop here since the fractional part is now 0.

Final Binary Representation

Combining both parts, we have:

Thus, the binary representation of **12.625** is **1100.101**

C8 en Hexadécimal a convertir en binaire :



Conversion hexadecimal to decimal : 9D

$$9 \qquad D$$

$$= 9 \times 16^{1} + 13 \times 16^{0}$$

$$= 9 \times 16 + 13 \times 1$$

$$= 144 + 13$$

$$= 157$$

$$(9D)_{16} = (157)_{10}$$

le nombre donné dispose d'une partie fractionnaire

Dans le cas où le nombre donné dispose d'une partie fractionnaire, l'opération de conversion se fait de la manière suivante :

- ✓ Multiplier la partie, fractionnaire du nombre donné par 2 et retenir la partie entière du résultat trouvé.
- ✓ Continuer à multiplier successivement par la valeur 2 les parties fractionnaires obtenues pendant chaque opération tout en retenant leurs parties entières, jusqu'à ce qu'on obtienne une partie fractionnaire nulle ou déjà obtenue, sinon ce sera une suite binaire infinie et dans ce cas , on peut se limiter à quelques bits (problème d'arrondi).
- ✓ Le résultat de cette conversion est l'ensemble, des parties entières conservées à chaque étape de multiplication.

Exemple :

Transformer le numéro 23 en binaire ?

$(23)_{10} = (10111)_2$

Exemple :

Convertir le nombre (23.625) de la base 10 à la base 2.

La partie entière 23 est déjà déterminée, son résultat est : $(23)_{10} = (10111)_2$

Pour la partie fractionnaire 0.625, on procède comme suit :

0.625 * 2 = 1.250

0.250 * 2 = 0.500

0.500 * 2 = 1.00

On constate que la dernière partie fractionnaire est nulle, donc le résultat sera :

$(0.625)_{10} = (0.101)_2$

Le résultat final : $(23.625)_{10} = (10111.101)_2$

Prenons le cas infini par exemple $(23.56)_{10}$

Pour $(23)_{10}$ c'est toujours (10111): mais pour la partie fractionnaire $(0.56)_{10}$, on aura : 0.56 *2=1.12

Activity

Convert $(1011.011)_2$ into its decimal equivalent

Answer should be $(\quad)_{10}$

Activity

Convert $(29.15)_{10}$ into its binary system

Answer should be $(\quad)_2$

Data Representation in Computer

- In modern computers, all information is represented using binary values.
- Each storage location (cell): has two states
  - low-voltage signal => 0
  - High-voltage signal => 1
  - i.e., it can store a binary digit, i.e., bit
- Eight bits grouped together to form a byte
- Several bytes grouped together to form a word
  - Word length of a computer, e.g., 32 bits computer, 64 bits computer

What is a bit (binary digit)?

- A bit (binary digit) is the smallest unit of data that a computer can process and store. A bit is always in one of two physical states, similar to an on/off light switch. The state is represented by a single binary value, usually a 0 or 1. However, the state might also be represented by yes/no, on/off or true/false.
- Bits are stored in memory through the use of capacitors that hold electrical charges. The charge determines the state of each bit, which, in turn, determines the bit's value.

Quelques notions

- Le fonctionnement du microprocesseur et de la Ram
  - ✓ Les informations sont traitées par des transistors qui reçoivent des micro-impulsions électriques (mode binaire)
- Le fonctionnement des unités de stockage
  - ✓ Les informations sont stockées sur des supports magnétiques ou optiques en mode binaire.

Different types of number systems

1. Binary Number System:

The fundamental system used in **computers**, consisting of only two digits: **0** and **1**. This **base-2 system** is essential for digital **electronics and computing**, as it aligns with **the on/off states of transistors**.

2. Decimal Number System:

The base-10 system that we use in everyday life, which includes the **digits** **0-9**. It is the most familiar number system for **humans** and is often used in user **interfaces and applications**.

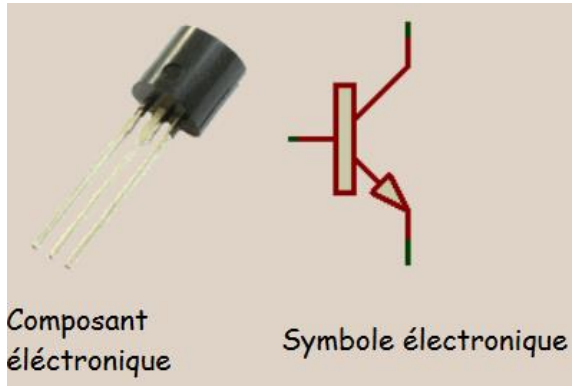
3. Octal Number System:

A **base-8 system** that uses digits from **0 to 7**. Octal is sometimes used in computing as a **shorthand** (abreviation) for binary, as each octal digit corresponds to three binary digits, making it more compact for certain applications.

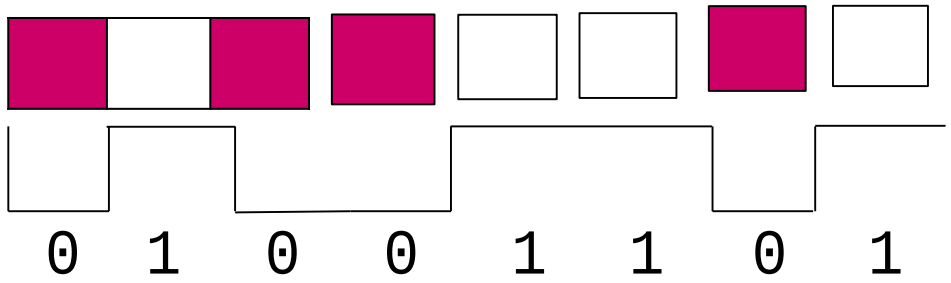
4. Hexadecimal Number System:

A **base-16 system** that includes digits **0-9** and letters **A-F** (representing values 10 to 15). Hexadecimal is widely used in **programming** and computer science because it can **represent large binary numbers more succinctly**, making it **easier** for **programmers** to **read and write code**.

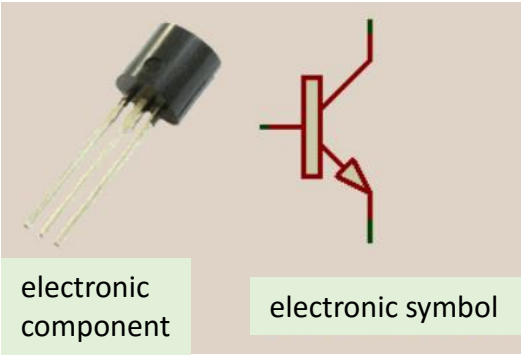
Formalisation du binaire



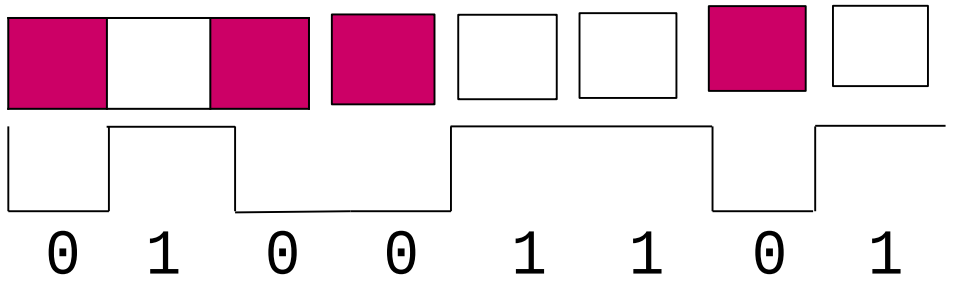
- Pour les mémoires centrales
  - L' élément de base c'est le **transistor**
  - Or le **transistor** est une pièce électronique qui fonctionne à la manière d'un interrupteur (Il est sensible aux impulsions électriques).
  - Dans son fonctionnement le plus simple le **transistor** peut prendre deux états :
    - **Un état bloqué** : dans cet état, le transistor ne permet pas le passage du courant.
    - **Un état passant** (on dit aussi saturé) : le transistor est en état de conduction. Le courant passe.
  - Il code en binaire des impulsions et des non impulsions électriques



Formalization of the binary



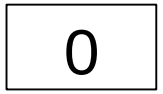
- For central memories
  - The basic element is the **transistor**
  - Now the **transistor** is an **electronic part** that works like a **switch** (It is sensitive to electrical impulses).
  - In its simplest operation the transistor can take two states:
    - **A blocked state**: in this state, the transistor does not allow the passage of current.
    - **A passing state** : the transistor is in a conduction state. The current passes.
  - It codes electrical impulses and non-impulses in binary

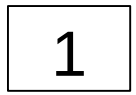


La capacité de codage

- **La notion de case binaire**
  - Les valeurs binaires (0,1) peuvent être stockées dans des emplacements fictifs que nous allons qualifier de case binaire.
  - Dans chaque case binaire on peut stocker indifféremment un 0 ou un 1.
  - Une case binaire code donc pour 2 états distincts.

  - Une case binaire = Bit (Binary Unit) 





- **The concept of binary box**
  - Binary values (0,1) can be stored in fictitious locations that we will call binary boxes.
  - In each binary box, we can store either a 0 or a 1.
  - A binary box therefore codes for 2 distinct states

Storage Capacity in Binary Mode

The concept of storage capacity in binary mode is fundamental to understanding how data is represented and stored in computing systems. At its core, binary storage relies on the use of bits, which are the smallest units of data in computing, represented as either a 0 or a 1.

Binary Representation

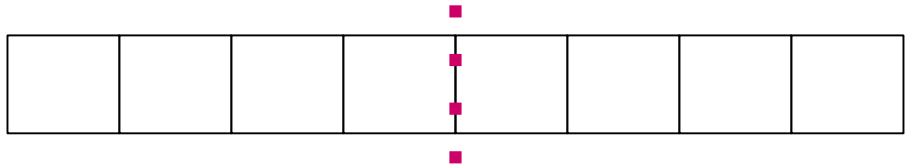
In binary mode, all types of data, whether text, images, or audio are converted into sequences of bits. Each bit can represent two states, allowing for the encoding of complex information through combinations of these bits. For example, a single byte, which consists of 8 bits, can represent 256 different values (from 0 to 255).

La notion de bit et octet (The concept of bit and byte)

- C’est l’enchainement de 8 cases binaires (It is the sequence of 8 binary boxes)

Avec 8 cases binaires (8 bits)

8 Bits = 1 Octet = 1 Byte



2^8 → 256 possibilités

L’octet est l’unité de base de codage de l’information

The byte is the basic unit of information coding.

Quartet de poids fort

Heavyweight quartet

Quartet de poids faible

Low weight quartet

- Les unités
- The units

Units of Measurement

The capacity of storage devices is typically measured in bytes and their multiples. Here are some common units:

1 Byte (B) = 8 bits

1 Kilobyte (KB) = 1,000 Bytes

1 Megabyte (MB) = 1,000 KB

1 Gigabyte (GB) = 1,000 MB

1 Terabyte (TB) = 1,000 GB

These units help quantify how much data can be stored on various devices, from small USB drives to large hard drives and cloud storage solutions.

Les unités

L’unité élémentaire le bit (valeur symbolique)

L’unité de base l’octet 2^8 Informations élémentaires

Le kilo-octet (Ko) 2^{10} octets	1024 octets
Le méga-octet (Mo) 2^{20} octets	1 048 576 octets
Le giga-octet (Go) 2^{30} octets	

Ce sont avec ces unités que l’on mesure les capacités de stockage de l’information par l’ordinateur.

Les performances des mémoires centrales (mémoires vives). Les performances des mémoires de stockage (disque dur…). Le poids des documents informatiques (fichiers).

Le code ASCII (American Standard Code for Information Interchange)

Aperçu de l'ASCII (American Standard Code for Information Interchange)

L'**ASCII** est une **norme de codage** de caractères qui joue un rôle crucial dans la façon dont les ordinateurs représentent le texte.

Qu'est-ce que l'ASCII ?

- **ASCII** signifie **American Standard Code for Information Interchange**. C'est un schéma de codage de caractères qui utilise des valeurs numériques pour représenter des caractères, permettant aux ordinateurs de stocker et de manipuler du texte.
- La norme **ASCII** originale utilise **7 bits**, ce qui permet de représenter **128 caractères uniques**, allant de 0 à 127.

Représentation des Caractères

Le code ASCII comprend :

Caractères de Contrôle :

Ce sont des caractères non imprimables utilisés pour contrôler des périphériques (par exemple, retour chariot, saut de ligne, del, esc ….).

Caractères Imprimables :

Cela inclut :

Lettres Majuscules : 'A' à 'Z' sont représentées par les codes 65 à 90.

Lettres Minuscules : 'a' à 'z' sont représentées par les codes 97 à 122.

Chiffres : '0' à '9' sont représentés par les codes 48 à 57.

Caractères Spéciaux : Comme les signes de ponctuation et les symboles (par exemple, espace, virgule, point).

Comment Fonctionne l'ASCII

Chaque caractère est assigné à une valeur numérique unique. Par exemple:

- ✓La lettre 'A' est représentée par la valeur décimale **65** (binaire 01000001).
- ✓Pour convertir une lettre majuscule en sa version minuscule, on peut ajouter **32** à sa valeur ASCII. Par exemple, 'A' (65) devient 'a' (97) en ajoutant 32.

ASCII étendu

Alors que la norme ASCII originale utilise 7 bits, une version **étendue** utilise 8 bits, permettant de représenter **256 caractères**. Cela inclut des symboles supplémentaires et des caractères utilisés dans diverses langues, rendant le système plus polyvalent pour une utilisation internationale.

Importance de l'ASCII

L'ASCII est fondamental pour la représentation du texte dans les ordinateurs et est largement utilisé dans la programmation, la transmission de données et les formats de fichiers. Il assure la compatibilité entre différents systèmes et dispositifs, ce qui le rend essentiel pour l'informatique moderne.

Conclusion
Comprendre l'ASCII est crucial pour quiconque apprend les sciences informatiques et la programmation. Cela fournit la base de la façon dont le texte est codé et traité dans les systèmes numériques.

Overview of ASCII (American Standard Code for Information Interchange)

ASCII is a character encoding standard that plays a crucial role in how computers represent text.

What is ASCII?

ASCII stands for **American Standard Code for Information Interchange**. It is a character encoding scheme that uses numerical values to represent characters, allowing computers to store and manipulate text. The original ASCII standard uses **7 bits**, which allows for **128 unique characters**, ranging from 0 to 127.

Character Representation

The ASCII code includes:

Control Characters: These are non-printable characters used for controlling devices (e.g., carriage return, line feed).

Printable Characters: This includes:

- **Uppercase Letters**: 'A' to 'Z' are represented by codes 65 to 90.
- **Lowercase Letters**: 'a' to 'z' are represented by codes 97 to 122.
- **Digits**: '0' to '9' are represented by codes 48 to 57.
- **Special Characters**: Such as punctuation marks and symbols (e.g., space, comma, period).

How ASCII Works

Each character is assigned a unique numerical value. For example:

The letter 'A' is represented by the decimal value **65** (binary 01000001).

To convert a capital letter to its lowercase equivalent, you can add **32** to its **ASCII** value. For instance, '**A**' (65) becomes 'a' (**97**) by adding 32.

Extended ASCII

While the original ASCII standard uses 7 bits, an **extended version** uses 8 bits, allowing for **256 characters**. This includes additional symbols and characters used in various languages, making it more versatile for international use.

Importance of ASCII

ASCII is foundational for text representation in computers and is widely used in programming, data transmission, and file formats. It ensures compatibility across different systems and devices, making it essential for modern computing.

Conclusion

Understanding **ASCII** is crucial for anyone learning about computer science and programming. It provides the basis for how text is encoded and processed in digital systems.

Table des codes ASCII

Char.	ASCII	Char.	ASCII	Char.	ASCII
@	64	U	85	j	106
A	65	V	86	k	107
B	66	W	87	l	108
C	67	X	88	m	109
D	68	Y	89	n	110
E	69	Z	90	o	111
F	70	[	91	p	112
G	71	\	92	q	113
H	72	]	93	r	114
I	73	^	94	s	115
J	74	_	95	t	116
K	75	`	96	u	117
L	76	a	97	v	118
M	77	b	98	w	119
N	78	c	99	x	120
O	79	d	100	y	121
P	80	e	101	z	122
Q	81	f	102	{	123
R	82	g	103	|	124
S	83	h	104	}	125
T	84	i	105	~	126