Chapter 1: Numeration systems

I.1 Introduction

The numbering system or the numeral system is the system of naming or representing numbers. A numeral system is a writing system for expressing numbers; that is, a mathematical notation for representing numbers of a given set, using digits or other symbols in a consistent manner. We know that a number is a mathematical value that helps to count or measure objects and it helps in performing various mathematical calculations. There are different types of number systems like decimal number system, binary number system, octal number system, and hexadecimal number system. In this chapter, we are going to learn what is a number system, different types, and conversion procedures with many number system examples.

I.2. What is a computer

A computer is a machine capable of carrying out all kinds of operations and processing such as calculations, handling of texts and images for example. A computer performs automated processing on data: it is capable of transforming, storing and archiving it.

The components of a computer such as machinery that includes wires, transistors, circuits, hard disk are called hardware. Whereas, the programs and data are called software. Everything you do on your computer will rely on both hardware and software.

I.3. What is computer science?

computer science is the study of computers, computing and computational systems, including their theoretical and algorithmic foundations, hardware and software, and their uses for processing information

I.4. The Data

Digital data is data that is represented using the binary number system of ones (1) and zeros (0). There are different kinds of data; such are as follows: Sound, Video, Single character, Number (integer or floating-point), Picture, Boolean (true or false), Text (string).

I.5. The Information

Information is organized or classified data, which has some meaningful values for the receiver. Computer information means information in electronic form which is obtained from or through the use of a computer or which is in a form capable of being processed by a computer.

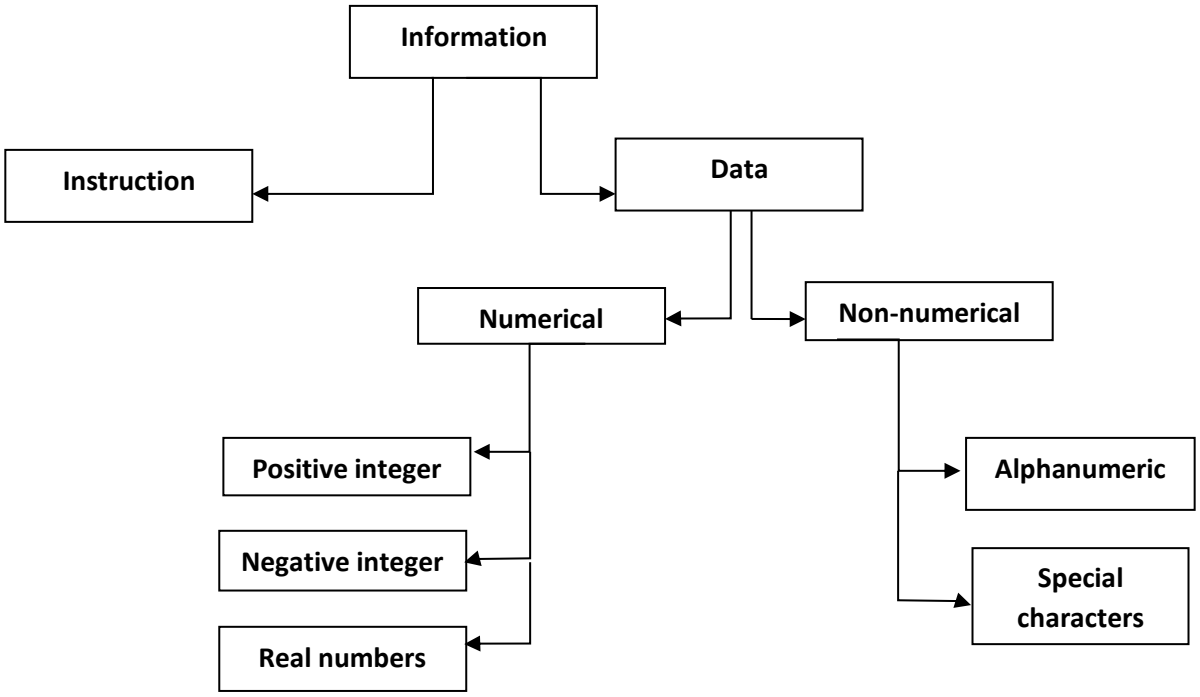
Information is processed by a computer and can be of different types (**instruction** or **data**: text, numbers, etc.). In a computer, information is digitized (digital): it is always represented and manipulated by the computer in binary form. All information will be treated as a sequence of 0s and 1s. The unit of information is the binary digit (0 or 1), which we call **bit** (for binary digit).

In computing: "information" has a few essential functions:

- **Exchange** information (keyboard, mouse, bus, screen, printer, modem, etc.).
- **Memorize** information (Hard Disk, RAM, ROM, DVD, CD, disc flash, etc.).
- **Calculate** information (processor).

I.5.1. Types of the information

- The treated information by the machine can be:
- **Instructions**: Operations carried out by the microprocessor, in the machine, are represented in binary according to different formats in relation to the microprocessor.
- **The Data**: can be of different types
    - Numerical data (digits)
    - Non-numerical data: char, images, sounds…



I.5.1.a. Numeric Data Types

Numeric data types are types of data that consist of numbers, which can be computed mathematically with various standard operators such as add, minus, multiply, divide and more. Examples of numeric data types are examination marks, height, weight, the number of students in a class, share values, price of goods, monthly bills, fees and others.

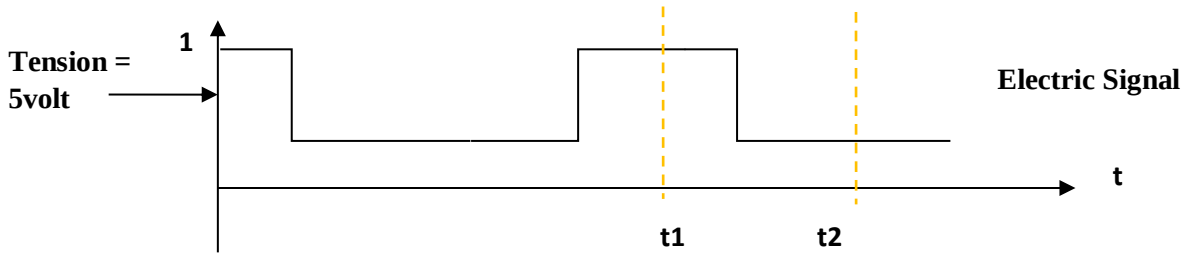
I.5.1.b. Non-numeric Data Types

Nonnumeric data types are data that cannot be manipulated mathematically using standard arithmetic operators. The non-numeric data comprises text or string data types.

I.5.2. Physical representation of the information

Although the information is digitized, its support is physical, therefore analog: typically, in current technology, tension.

Knowledge of a state therefore depends on two thresholds, high and low. With an electrical signal, it shows that there are two significant states, the low state (value 0) when the voltage is lower than a low reference, and a high state (value 1) when the voltage is higher than a high reference, for example, at t1 the signal is high and at t2 the signal is low.



Physical representation of the information

The term bit: means "binary digit", i.e. 0 or 1 in binary numbering. It is the smallest unit of information that can be manipulated by a digital machine. It is possible to physically represent this binary information: by an electric or magnetic signal, which, beyond a certain threshold, corresponds to the value 1.

Remark:

1. With a bit, it is thus possible to obtain two states: either 1 or 0 (2^1)
2. With 2 bits, it is possible to obtain 4 different states (2^2):

00
01
10
11

3. With 3 bits, it is possible to obtain 8 different states: (2^3) binary value on 3 bits

000
001
010
011
100
101
110
111

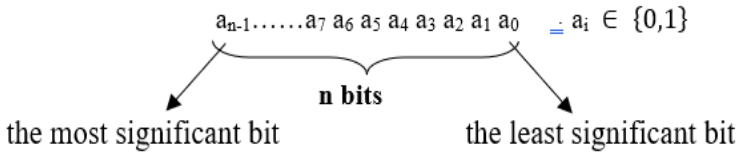
4. For a group of n bits, it is possible to represent (2^n) binary sequences.

000000…….0
0000……..1
0000…..…..10
000……...…..11
000…….…100
. . . .
111111……1

$n\ bits$

Binary word: is a group of fixed number of bits processed as a unit, which varies from computer to computer:

- A word of n bits is a binary sequence (a_i), $0 \leq i \leq n-1$;
- a_0 is the **least significant bit**,
- a_n is the **most significant bit**

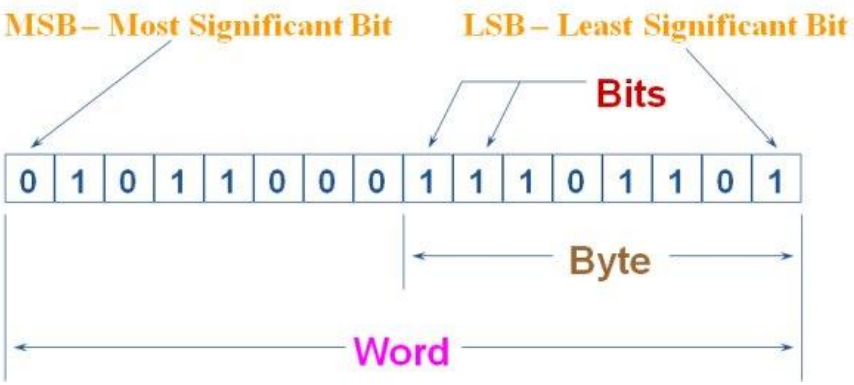


Example :

Consider ***N*** = $a_{15}\ldots\ldots a_7\ a_6\ a_5\ a_4\ a_3\ a_2\ a_1\ a_0$ = ***1110001000011110 :*** is a binary sequence of size ***n=16 bits***

The byte: is a unit of information composed of 8 bits. For example, it allows you to store a character, such as a letter or a number. For a byte:

- The smallest number is 0 (represented by eight zeros 00000000),
- And the greatest is 255 (represented by eight digits "one" 1111 1111),
- This represents 256 possibilities of different binary values.



weight of bits

In binary numbers, each bit has a value, called weight, depending on the position of the bit. Each bit has a weight twice that of the previous bit. The least significant bit has a weight of 1, then next 2 and so on. The LSB represents a value of 1 and then the successive bits have values of the order 2^{n-1}. Therefore; the weight of a bit increases by a power of two going from right to left as shown in the following table:

Let the Binary number represented by $a_7\ a_6\ a_5\ a_4\ a_3\ a_2\ a_1\ a_0 : a_i \in \{0,1\}$

	the bit	The weight	Value
Least significant bit	a_0	2^0	1
	a_1	2^1	2
	a_2	2^2	4
	a_3	2^3	8
	a_4	2^4	16
	a_5	2^5	32
	a_6	2^6	64
Most significant bit	a_7	2^7	128

Here are the standard units:

1 Byte = 8 Bits

1 kilobyte (kB) = 2^{10} bytes = 1024 bytes

1 Megabyte (MB) = 2^{20} bytes = 1024 ko = 1 048 576 bytes

1 Gigabyte (GB) = 2^{30} bytes = 1024 Mo = 1 073 741 824 bytes

1 Terabyte (TB) = 2^{40} bytes = 1024 Go = 1 099 511 627 776 bytes

Unit	Shortened	Capacity
Bit	b	1 or 0 (on or off)
Byte	B	8 bits
Kilobyte	KB	1024 bytes = **2^{10} bytes**
Megabyte	MB	1024 kilobytes = **2^{20} bytes**
Gigabyte	GB	1024 megabytes = **2^{30} bytes**
Terabyte	TB	1024 gigabytes = **2^{40} bytes**
Petabyte	PB	1024 terabytes = **2^{50} bytes**
Exabyte	EB	1024 petabytes = **2^{60} bytes**
Zettabyte	ZB	1024 exabytes = **2^{70} bytes**
Yottabyte	YB	1024 zettabytes = **2^{80} bytes**

I.6. Numeral systems

Before approaching the representation of the different types of data (characters, natural numbers, real numbers), it is advisable to become familiar with the representation of a number in any base. Subsequently, we will often use the bases 2, 8, 10 and 16 that are the most used in computing.

Usually, we use the base 10 to represent numbers, that is to say that we write using 10 distinct symbols: digits.

In base b, we use b digits. Let's denote a_i the digits used to write a number

$$N = a_{n-1}...a_1a_0$$

- In **Decimal**, b = 10, $a_i \in \{0,1,2,3,4,5,6,7,8,9\}$ 10 decimal digits
- In **binary**, b = 2, $a_i \in \{0,1\}$: 2 binary digits, or bits
- In **octal**, b = 8, $a_i \in \{0,1,2,3,4,5,6,7\}$: 8 octal digits
- In **hexadecimal**, b = 16, $a_i \in \{0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F\}$ 16 hexadecimal digits (we use the first 6 letters "A,B,C,D,E,F" as numbers for the numbers 10,11, 12, 13,14, 15 respectively).

Notation of numbers written in hexadecimal

Let us observe that the writing 1753 is ambiguous: it can be a number written in decimal or in hexadecimal, or in octal. The result is very different. Depending on the case, several notations are used to indicate that a number is written in hexadecimal:

- 1753_{16}
- **0x**1753 (0=zéro)
- 1753**H**

Examples:

- $(01101110)_2$ is a binary number with size n=8 bits (represented in one byte)

- $(2801687)_{10}$ is a decimal number represented in n=7 digits
- $(2AC017)_{16}$ is hexadecimal number represented in n=6 digits
- $(20157)_8$ is an octal number of size n=5 digits

I.6.1. Convert a number N from any base "b" to a decimal number

Let N = $(a_{n-1}...a_1a_0)_b$ sequence of digits in a base b system. To convert a word in any base to a decimal number, simply multiply the value of each digit by its weight, then add each result e.i We multiply each digit of the number with its place value and add the products.

$$\mathbf{N_{10}} = (a_{n-1}) * \mathbf{b}^{n-1} + \ldots. (a_2 * b^2) + (a_1 * b^1) + (a_0 * b^0)$$

$$= \sum_{i=0}^{n-1} (a_i \times b^i)$$

Example :

- Thus, the binary word $(0101)_2$ in decimal is: $2^3x0 + 2^2x1 + 2^1x0 + 2^0x1 = 8x0 + 4x1 + 2x0 + 1x1 = 5$
- The hexadecimal word $(2AC017)_{16}$ is worth in decimal: $16^5x\ 2 + 16^4x10 + 16^3x12 + 16^2x0 + 16^1x1 + 16^0x7 = (2801687)_{10}$
- The octal word $(20157)_8$ is worth in decimal: $8^4x2 + 8^3x0 + 8^2x1 + 8^1x5 + 8^0x7 = (8303)_{10}$

decimal	octal	hexadecimal	Binary
0	0	0	0000
1	1	1	0001
2	2	2	0010
3	3	3	0011
4	4	4	0100
5	5	5	0101
6	6	6	0110
7	7	7	0111
8	10	8	1000
9	11	9	1001
10	12	A	1010
11	13	B	1011
12	14	C	1100
13	15	D	1101
14	16	E	1110
15	17	F	1111
16	20	10	10000
17	21	11	10001

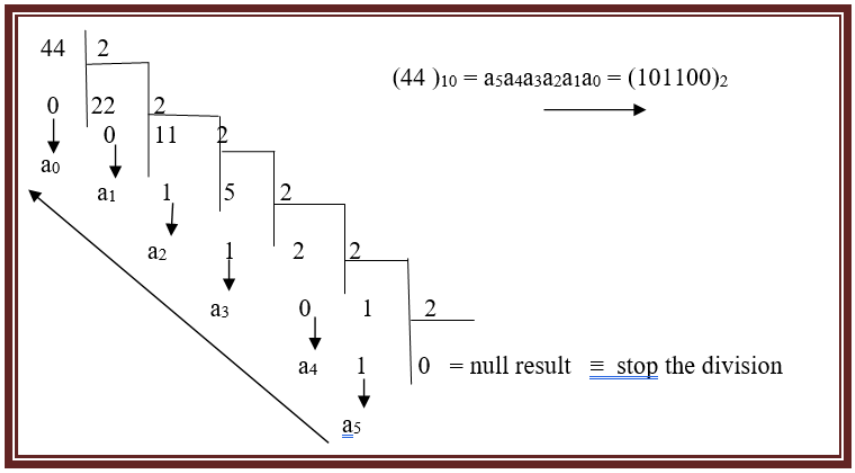
I.6.2. Convert a decimal number N to a number in any base system b

For a positive decimal number, we proceed by successive divisions. We divide the number by the base, then the quotient obtained by the base, and so on until the quotient becomes 0. The sequence of remainders obtained corresponds to the digits in the numeration system of the target base b. Write the remainders **from bottom to top** and **from the left to the right**

Example:

Convert the number $(44)_{10}$ to base 2, then to the base 8, and then to the base 16 system

- To binary system:



I.6.3. Convert a binary number to a hexadecimal number and to an octal number

These bases: 2, 8 and 16 correspond to powers of 2 (2^1, 2^3 and 2^4), hence very simple passages from one to the other.

I.6.3.1. convert a binary number to an octal number

Starting from the right to the left, we partition the binary number into groups of three bits. Then, we convert each group of binary numbers to octal and write them in the same order.

Example: $(110\ 101\ 001\ 011)_2 = (?)_8$

Let's split it into groups of 3 digits:	110	101	001	011
Let's translate each of the groups into octal	6	5	1	3
Group values in decimal	6×8^3	5×8^2	1×8^1	3×8^0

So: $(110101001011)_2 = (6513)_8$.

I.6.3.2. convert a binary number to a hexadecimal number

We group the bits from right to left 4 bits per group, then we replace the 4 bits of each group by a corresponding hexadecimal digit.

Example: $(1101\ 0100\ 1011\ 0111)_2=(?)_{16}$

Let's split it into groups of 4 digits:	1101	0100	1011	0111
Let's translate each of the groups into hexadecimal	D	4	B	7
Group values in decimal	13×16^3	4×16^2	11×16^1	7×16^0

So $(1101\ 0100\ 1011\ 0111)_2$= $(D4B7)_{16}$=$(54\ 465)_{10}$.

I.6.4. Convert octal number to binary number

To convert an octal number to binary, we write 3-bit binary equivalent of each octal digit in the same order. Therefore; We replace (represent) each octal digit of the number by the corresponding 3 binary digits

I.6.5. Convert hexadecimal number to binary number

Convert hexadecimal to binary is a conversion of a number in a hexadecimal number system to an equivalent number in the binary number system.

We write 4-bit binary equivalent of each hexadecimal digit in the same order. Therefore; We replace (represent) each hexadecimal digit of the number by the corresponding 4 binary digits

I.6.6. Representation of fractional numbers (unsigned real numbers)

Fractional numbers are those that have digits after the decimal point. Example: In the decimal system, we write

$$\mathbf{(12,\ 346)_{10}} = 1*10^1 + 2*10^0 + 3*10^{-1} + 4*10^{-2} + 6*10^{-3}.$$

I.6.6.1. Convert fractional number from any base b to decimal number

Consider the fractional number "N" written in base "b" as follows: $N=(a_{n-1}...a_1a_0\ ,\ a_{-1}a_{-2}...a_{-p})_b$

The decimal equivalent is obtained by the following formula:

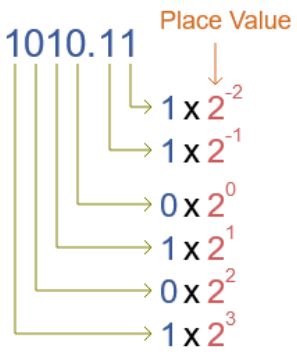
$$N_{10}=[(a_{n-1}*b^{n-1}+...+a_0*b^0)+(a_{-1}*b^{-1}+...+a_{-p}*b^{-p})]_{10}$$

Then the value of any decimal number will be equal to the sum of its digits multiplied by their respective weights,

Example: Convert the number $N=(1010.11)_2$ to decimal number

We multiply each binary digit with its place value and add the products.

$\mathbf{(1010.11)_2} = (\mathbf{1}\times\mathbf{2^3}) + (\mathbf{0}\times\mathbf{2^2}) + (\mathbf{1}\times\mathbf{2^1}) + (\mathbf{0}\times\mathbf{2^0}) + (\mathbf{1}\times\mathbf{2^{-1}}) + (\mathbf{1}\times\mathbf{2^{-2}}) = \mathbf{(10.75)_{10}}$



N= $(1010.11)_2$ = $(10.75)_{10}$

I.6.6.2. Convert a fractional decimal number to any base b

We convert the integer part and fractional part separately and then combine the results. We multiply the **fractional part** by the base "b", by repeating the operation on the fractional part of the product until it becomes zero (or until the desired precision is reached). For the **integer part**, we proceed by divisions as for an integer.

Example*:* Convert the number N= $(85.375)_{10}$ to binary number (base 2)

Integer part

The integer part of 85.375 is 85. Divide this number repeatedly by 2 until the quotient becomes 0. Write the remainders **from bottom to top** :

$$(\mathbf{85})_{10} = (\mathbf{1010101})_2$$

Fractional part

The fractional part of 85.375 is 0.375. Multiply the fractional part repeatedly by 2 until it becomes 0.

- **0.375 × 2 = 0.750**
- **0.750 × 2 = 1.500**
- **0.500 × 2 = 1.000**

From top to bottom, write the integer parts of the results to the fractional part of the number in base 2: $(\mathbf{0.375})_{10} = (\mathbf{0.011})_2$

Overall result

Combine the whole number and fractional parts to obtain the overall result.

$$N = (\mathbf{85.375})_{10} = (\mathbf{1010101})_2 + (\mathbf{0.011})_2 = (\mathbf{1010101.011})_2$$

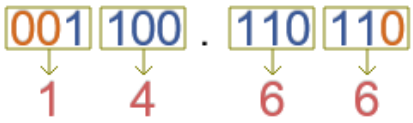
I.6.6.3. Convert fractional number from binary to octal system

- Starting from the binary point, we partition the binary number into groups of three bits.
- In the **integer part**, we proceed to the left. To complete the leftmost group of bits, we append **two zeros** to the left.
- In the **fractional part**, we proceed to the right. To complete the rightmost group of bits, we append **a zero** to the right.
- We convert each group of binary numbers to octal and write them in the same order.

Example: Convert the number N= $(1100.11011)_2$ to octal number

N= $(001100.110110)_2$

- $(001)_2 = (1)_8$
- $(100)_2 = (4)_8$
- $(110)_2 = (6)_8$
- $(110)_2 = (6)_8$



$(1100.11011)_2 = (14.66)_8$

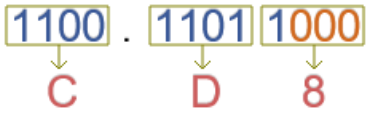
I.6.6.4. Convert fractional number from binary to hexadecimal system

- Starting from the binary point, we partition the binary number into groups of four bits.
- In the **integer part**, we proceed to the left.
- In the **fractional part**, we proceed to the right. To complete the rightmost group of bits, we append **three zeros** to the right.
- We convert each group of binary numbers to octal and write them in the same order.

Example: Convert the number N= $(1100.11011)_2$ to hexadecimal system

N= $(1100.11011000)_2$

- $(1100)_2 = (C)_{16}$
- $(1101)_2 = (D)_{16}$
- $(1000)_2 = (8)_{16}$



$N = (1100.11011)_2 = (C.D8)_{16}$

I.6.6.5. Convert fractional number from octal or hexadecimal to binary system

To convert an octal number to binary, we write 3-bit binary equivalent of each octal digit in the same order:

$(162.05)_2 = (001110010.000101)_8$

To convert a hexadecimal number to binary, we write 4-bit binary equivalent of each hexadecimal digit in the same order:

$(A46.09)_2 = (101001000110.00001001)_{16}$

I.7. Arithmetic operations on binary numbers (naturel numbers)

Binary operation is an operation that requires two inputs. These inputs are known as operands. The binary operation of addition, multiplication, subtraction and division takes place on two operands.

I.7.1. Definition of "Carry" and "overflow"

- In elementary arithmetic, a carry is a digit that is transferred from one column of digits to another column of more significant digits. We say that there is a carry if an arithmetic operation generates a carry at the end.
- We say that there is an overflow: if the result is too big to fit in the available digits i.e the number of bits used is insufficient to contain the result. In other words, the result exceeds the range of values on the used n bits.

I.7.2. The addition

The addition is done digit by digit from least significant bit to the most significant bit by propagating the carry

Carry

We have:

0 + 0	0 + 1	1 + 0	^{1} 1 + 1
= 0	= 1	= 1	= 0 The result = $(2)_{10}$ = $(10)_2$

Example:

0011010 + 001100 = 00100110

```
   1 1          carry
 0 0 1 1 0 1 0  = 26₁₀
+0 0 0 1 1 0 0  = 12₁₀
---------------
 0 1 0 0 1 1 0  = 38₁₀
```

I.7.3. The subtraction

The subtraction is done digit by digit from least significant bit to the most significant bit.

We have

0 - 0	^{2} 0 - ^{1} 1	1 - 0	1 - 1
= 0	= 1	= 1	= 0

Borrow the base

Example:

0011010 - 001100 = 00001110

```
  1 1      borrow
0 0 1 1 0 1 0   = 26(10)
- 0 0 0 1 1 0 0 = 12(10)
-----------------
  0 0 0 1 1 1 0 = 14(10)
```

I.7.4. The multiplication

As in decimal system by series of additions (the sum must be in binary)

0 x 0	0 x 1	1 x 0	1 x 1
= 0	= 0	= 0	= 1

Example:

0011010 x 001100 = 100111000

```
  0 0 1 1 0 1 0   = 26(10)
x 0 0 0 1 1 0 0   = 12(10)
-----------------
    0 0 0 0 0 0 0
  0 0 0 0 0 0 0
0 0 1 1 0 1 0
0 0 1 1 0 1 0
-----------------
1 0 0 1 1 1 0 0 0   = 312(10)
```

I.7.5. The division

As in the decimal system, the number we want to divide must be greater than the divisor, otherwise the result is 0

Example :

```
101100 | 110
-110   |----
01010  | 111
 -110  |
 01000 |
  110  |
  0    |
```

$(44)_{10}$	$(6)_{10}$
2	7