Badji Mokhtar University

SCIENCE AND TECHNIQUES DEPARTMENT
1st year ST

Course 2: Information coding systems

by

Dr. Samira LAGRINI

lagrini.samira83@gmail.com

Course link : https://elearning-facsci.univ-annaba.dz/course/view.php?id=469

Academic year:2024/2025

Introduction

- Computers process various types of information, such as numbers, text, images, and videos.
- This information is always represented in binary form (a sequence of 0s and 1s) such as: 01001011, 11000011, and so on.
- The process that allows converting the original representation of information (numbers, text, etc.) into a binary form is called **information encoding**.
- To make this transformation possible, **number systems** are essential.

Number systems

1. What is Number System?

- A number system is a system of writing to express numbers.

It is defined by:

  - A set of symbols
  - Some rules for writing numbers (Juxtaposition of symbols)

- The total number of symbols that are used in a number system is called the base of the number system,

- There are four number systems :

- Binary
- Octal
- Decimal
- Hexadecimal

a. Decimal number system

- The decimal number system contains ten unique symbols {**0, 1, 2, 3, 4, 5, 6, 7, 8, 9**} → **base 10**
- It is a positional weighted system, The value attached to the symbol depends on its location with respect to the decimal point.

For example:

the nomber **5368** is written as: **5368** = $8*10^0+6*10^1+3*10^2+5*10^3$

Polynomial form

$5368,135=5*10^3+3*10^2+6*10^1+8*10^0+1*10^{-1}+3*10^{-2}+5*10^{-3}$

The integer part The fraction part.

b. Binary number system

- The binary number system is a positional weighted system.
- The symbols used are {0,1} ➔ base=2
- The binary point separates the integer and fraction parts.

Example:

$(\underline{1}101110\underline{1})_2$

Most significant bit (MSB) **Less significant bit(LSB)**

Example:

$(11011101)_2 = 2^0*1+2^1*0+2^2*1+2^3*1+2^4*1+2^5*0+2^6*1+2^7*1$

$= (221)_{10}$

$(1110010.01)_2$

c. Octal number system

- It is also a positional weighted system.
- It has 8 independent symbols {0,1,2,3,4,5,6,7}

=> Its base=8

Example:

$$(175)_8 = 8^0*5+8^1*7+8^2*1$$
$$=(125)_{10}$$

d. **Hexadecimal number system**

- The symbols used are : {0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F}

➔The base or radix of this number system is 16,

Example:

- $(AB01)_{16}$
- $(150F)_{16}$

CONVERSION FROM ONE NUMBER SYSTEM TO ANOTHER

Conversion from base 'B' to base 10

- Use polynomial representation
- $X = (a_{n..}a_2a_1a_0)_b \quad = b^0a_0 + b^1a_{1+...}b^na_n = (\sum a_ib^i)_{10}$

Examples:

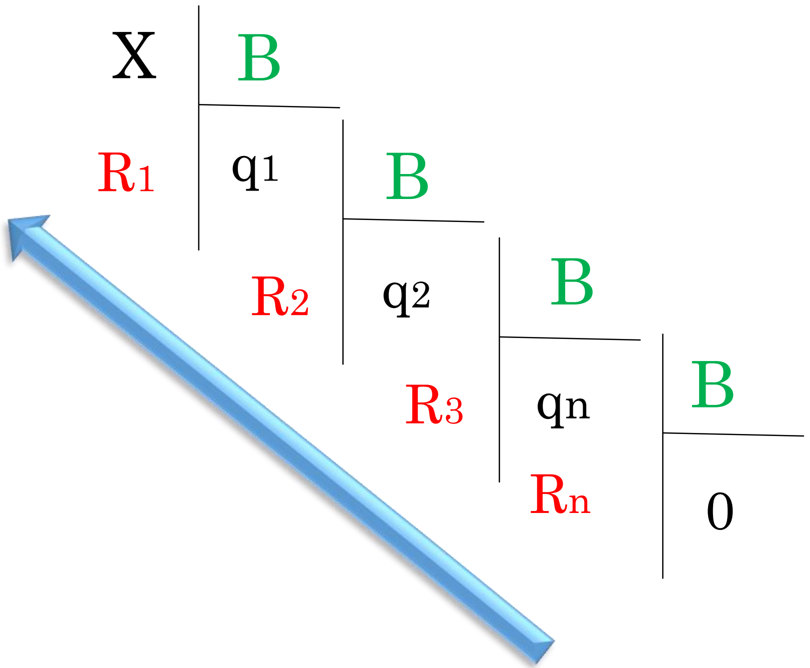
- ❖ $(11011101,1)_2 = 2^{-1}*1 + 2^0*1 + 2^1*0 + 2^2*1 + 2^3*1 + 2^4*1 + 2^5*0 + 2^6*1 + 2^7*1 = (221,5)_{10}$
- ❖ $(175,26)_8 = 8^{-1}*2 + 8^{-2}*6 + 8^0*5 + 8^1*7 + 8^2*1$
- ❖ $(14)_{16} = 16^0*4 + 16^1*1 = (20)_{10}$

Conversion from base 10 to another base B

- The number is converted to the desired base ‘B’ using successive division by the

Base ‘B’.

- Take the remainders of successive divisions on the base X in the opposite direction.

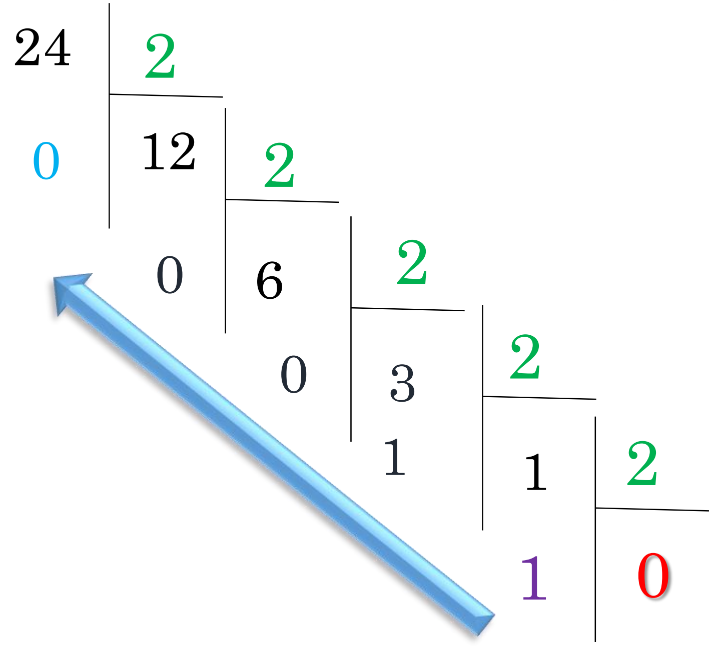


$$(X)_{10}=(R_n..R_3R_2R_1)_B$$

Conversion: decimal to base (2,8,16)

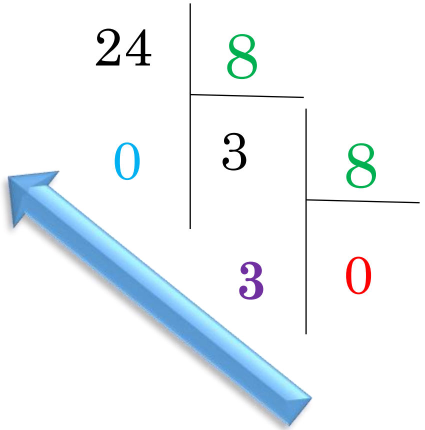
Soit $X=(24)_{10}$

decimal → binary



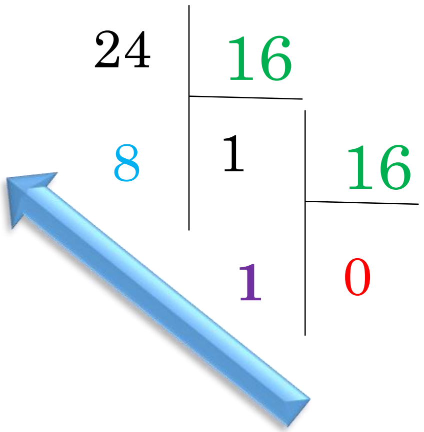
$(24)_{10} = (11000)_2$

decimal → Octal



$(24)_{10} = (30)_8$

decimal → hexadecimal



$(24)_{10} = (18)_{16}$

Trick: decimal to binary

Use the table below to represent the number written in decimal as a sum of powers of 2.

Example

80=64+16= $2^6 + 2^4$ → *the bits of weight 0,1,2,3,5,7 are set to 0*

19=16+2+1= $2^4 + 2^1 + 2^0$ → *the bits of weight 2,3,5,6,7 are set to 1*

	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	256	128	64	32	16	8	4	2	1
80	0	0	1	0	1	0	0	0	0
19	0	0	0	0	1	0	0	1	1

$(80)_{10} = (1010000)_2$

$(19)_{10} = (10011)_2$

Conversion: decimal to binary

- Convert $(80.15)_{10}$ into binary.

Integer part:

$(80)_{10} = (1010000)_2$

Fraction part:

0.15 x 2 = 0.30

0.30 x 2 = 0.60

0.60 x 2 = 1.20

0.20 x 2 = 0.40

0.40 x 2 = 0.80

0.80 x 2 = 1.60

Result of $(80.15)_{10}$ is $(1010000.001001)_2$

Conversion: binary ⇆ octal

Binary → Octal

- Make 3-bit groupings starting from the least significant bit (LSB).
- Replace each grouping with the corresponding value.

$$(10111101)_2$$

2 7 5

=

$$(275)_8$$

Octal → Binary

- Replace each symbol in the octal base with its 3-bit binary value

$$(213)_8$$

2 1 3

010 001 011

Conversion: binary ⇄ hexadecimal

Binary → hexadecimal

- Make 4-bit groupings starting from the least significant
- Replace each grouping with the corresponding value.

$$(1011\,1101,\ 0100)_2$$

B D, 4

=

$$(BD,4)_{16}$$

Hexadecimal → binary

- Replace each symbol in the hexadecimal base with its value in 4-bit binary

F D

11 1 11101

Arithmetic operations (the sum)

In binary

The sum

+	0	1
0	0	1
1	1	1 0

a carry

```
  1 111
   11011
+
   10110
  110001
```

The product

*	0	1
0	0	0
1	0	1

In Octal

```
  1 1
  375
+
   33
(430)8
```

In hexadecimal

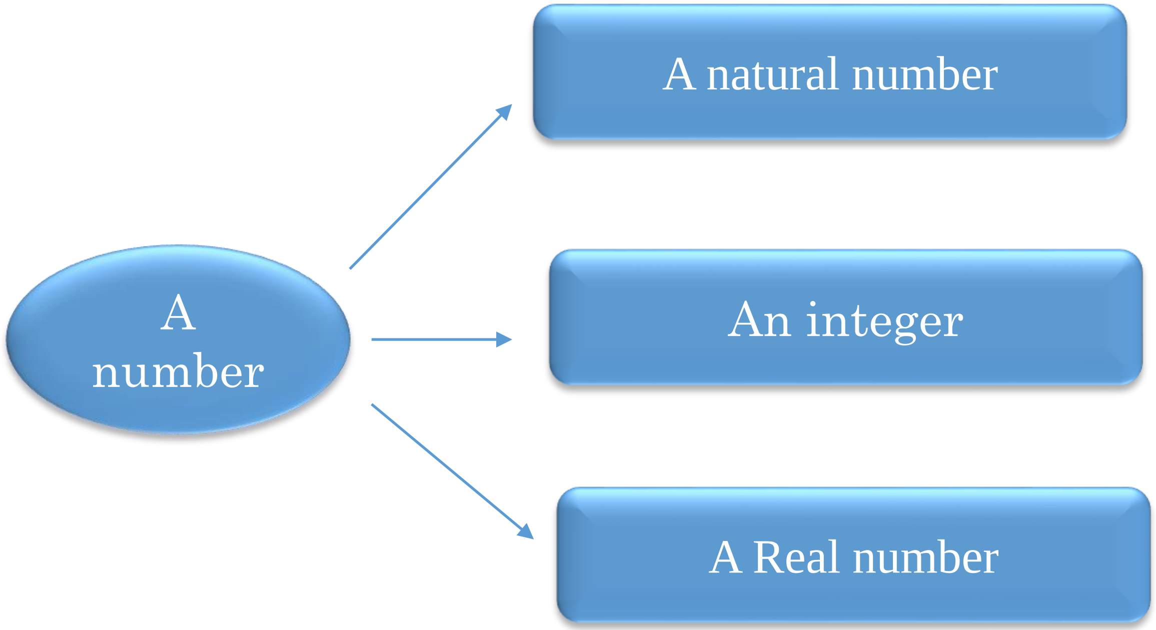
```
  1 1
  1BA
+
   F6
(2B0)16
```

Exercise

- Perform the following operations and transform the result to decimal
- $(1101{,}111)_2+(11{,}1)_2=(?)_2$
- $(43)_8+(76)8=(?)_8$
- $(AB1)_{16}+(237)_8=(?)_{16}$

Information Encoding

1.Coding of digital data



1.**Coding of natural number – 'The pure binary code'**

- A natural number is a positive integer or zero.
- To encode natural integers we use <u>the pure binary code (PBC)</u> :
- according to (PBC), the natural number is represented in <u>base 2 on N bits</u>.
- The choice of how many bits to use depends on the range of numbers to be used.

Exemple:

- on one byte (8 bits) , $(17)_{10}$ is encoded in pure binary as follows: 00010001

- On 1 byte (8 bits): we can code 2^8 values : [0 ; 255]
- On 2 bytes(16 bits): we can code 2^{16} values : [0 ; 2^{16}-1]
- On n bytes : we can code 2^n values : [0 ; 2^n-1]

2. Coding of signed integers

Two’s Complement

- An integer is a whole number which may be negative.
- ‘The two's complement’ is one of the techniques used to represent integers.

The representation of a number ‘X’ in 2's complement on ‘n’ bits is done as follow:

- if (X>= 0) then X is encoded in the same way as in pure binary.
- if (X<0) then :

1. Code |X| in binary by completing on the left with 0 to obtain an n-bit code
2. Invert all bits of the binary representation (one's complement);
3. Add 1 to the result (two's complement or C2)

Two's Complement-(2nd Method)-

if X <0 then its 2's complement is equal to $(2^n + X)$ coded in binary on n-bit

- *Example*: code -24 en 2's complement on 8 bit

First method

|-24|=24= 00011000

Reverse the bits (1's complement)= 11100111

Then add 1 to the result: 11100111+1

-24 = 11101000 (c à 2)

2^{nd} method

2^8-24=256-24=232

232= $(11101000)_2$

(-24)=11101000 (2's complement)

The 2's Complement -(Tip)-

Transforming a binary number into its 2's complement can be done as follows:

Look at the number from right to left, leaving the bits before the first '1' unchanged, then invert all subsequent bits.

Example: code the number -24 in 2's complement on 8 bits

24= $(00011000)_2$

- Invert the left part after the first 1 (written in red) : **1110**1000

➔ -24 : 11101000

Comments

- The highest-weighted bit is 1➔ it is a negative number.
- If you add 5 and -5 (00000101 + 11111011) the sum is 0 (with remainder 1).

3.Real Numbers Encoding

How to represent a number with a decimal point in binary?

In other words, how to encode real numbers???

IEEE standard 754 defines how to encode real numbers.

IEEE standard 754

- This standard offers a way to code a real number using 32 bits (simple precision).
- IEEE 754 defines three components:

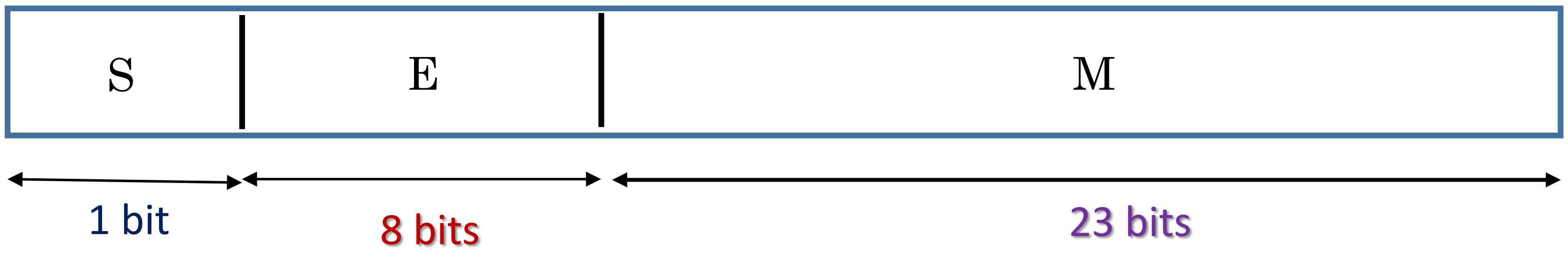
(S; E ;M)

S: represents the sign (0: positive/1: negative).

It is represented by one bit, the highest-weighted bit

E: the exponent is encoded using 8 bits immediately after the sign

M: the mantissa (the bits after the decimal point) on the remaining 23 bits



Steps for representation under IEE standard 754

1. Encode X in binary in the form :

$$X = \pm 1{,}M \cdot 2^{dec}$$

2. Compute the exponent E

$$E = dec + 127$$

3. Represent the 3 components (S, E, M) on 32 bits

IEEE standard 754-(examples)-

- *Example1*: compute the binary representation of (8,25)10 under IEEE standard 754

Solution

8,25 is positive, so the first bit will be 0 (S=0)

- Its representation in base 2 is:
- (8,25)10=(1000, 01)2

$=1,00001*2^3$ ← dec

$8,25=1,\underline{00001}*2^3$

↑ The mantissa (M)

IEEE standard 754 -(examples)-

- $E = dec + 127 = 3 + 127 = (130)_{10}$

  $= (10000010)_2$

As the mantissa must take up 23 bits, zeroes must be added to complete it:

- M=00001000000000000000000

0	10000010	00001000000000000000000
1 bit	8 bits	23 bits

The binary representation of 8,25 under IEEE standard 754 is therefore:

01000001000001000000000000000000

$= (41040000)_{16}$

IEEE standard 754

Example 2:
The value $(20,5)_{10}$ is to be encoded under IEEE standard 754

$(20,5)_{10}=(10100,1)_2$
$= + 1,01001 * 2^4$

- S =0
- E= 127+4 =131=10000011
- M= 01001

The binary representation of the number 20,5 under IEEE standard 754 is:
$(01000001101001000000000000000000)_2$
$= (41A40000)_{16}$

C. Conversion from IEEE Standard 754 to Decimal

To convert a number ‘X’ coded according to the IEEE standard 754 to decimal, you simply need to decompose this number into its elements:S,E,M, then estimate its representation in floating point format (X= ± 1,M . 2dec)

Example

- X=(01000001011010000000000000000000)
- X= 0 10000010 11010000000000000000000

(S → 0, E → 10000010, M → 11010000000000000000000)

S=0 => a positive number

E=$(10000010)_2$=130 ; E= dec+ 127 => dec=3

X= + 1,M * 2^3 = 1,11010000000000000000000 * 2^3 (dec=3)

X=+ 1110,10 = $(14,5)_{10}$

2. Character encoding

Character →
- letter (A-Z , a-z),
- number (0 ,1,2,3,4,5,6,7,8,9),
- punctuation (;. ? ! ….)
- symbol (&, $, %,,…)

Characters encoding is the process of converting characters(letters, numbers, punctuation, and symbols) into unique format for transmission or storage in computers.

Character encoding

Data is represented in computers using:

- ➢ ASCII
- ➢ UTF8
- ➢ UTF32
- ➢ ISCII
- ➢Unicode .

ASCII

- ASCII standard known as American Standard Code for Information Interchange was first published in 1963.
- ASCII is an 8-bit code standard that divides the 256 slots as follows:
  - Codes from 48 to 57 : numbers in order (0,1,…,9)
  - codes from 65 to 90: capital letters (A….Z)
  - Codes from 97 to 122: lowercase letters (a….z).

ASCII

0		24	↑	48	0	72	H	96	`	120	x	144	É	168	¿	192	└	216	╪	240	≡
1	☺	25	↓	49	1	73	I	97	a	121	y	145	æ	169	⌐	193	┴	217	┘	241	±
2	☻	26	→	50	2	74	J	98	b	122	z	146	Æ	170	¬	194	┬	218	┌	242	≥
3	♥	27	←	51	3	75	K	99	c	123	{	147	ô	171	½	195	├	219	█	243	≤
4	♦	28	∟	52	4	76	L	100	d	124	|	148	ö	172	¼	196	─	220	▄	244	⌠
5	♣	29	↔	53	5	77	M	101	e	125	}	149	ò	173	¡	197	┼	221	▌	245	⌡
6	♠	30	▲	54	6	78	N	102	f	126	~	150	û	174	«	198	╞	222	▐	246	÷
7		31	▼	55	7	79	O	103	g	127	⌂	151	ù	175	»	199	╟	223	▀	247	≈
8		32		56	8	80	P	104	h	128	Ç	152	ÿ	176	░	200	╚	224	α	248	°
9		33	!	57	9	81	Q	105	i	129	ü	153	Ö	177	▒	201	╔	225	ß	249	∙
10		34	"	58	:	82	R	106	j	130	é	154	Ü	178	▓	202	╩	226	Γ	250	·
11	♂	35	#	59	;	83	S	107	k	131	â	155	¢	179	│	203	╦	227	π	251	√
12	♀	36	$	60	<	84	T	108	l	132	ä	156	£	180	┤	204	╠	228	Σ	252	ⁿ
13		37	%	61	=	85	U	109	m	133	à	157	¥	181	╡	205	═	229	σ	253	²
14	♫	38	&	62	>	86	V	110	n	134	å	158	₧	182	╢	206	╬	230	µ	254	■
15	¤	39	'	63	?	87	W	111	o	135	ç	159	ƒ	183	╖	207	╧	231	τ	255	a
16	►	40	(	64	@	88	X	112	p	136	ê	160	á	184	╕	208	╨	232	Φ		
17	◄	41	)	65	A	89	Y	113	q	137	ë	161	í	185	╣	209	╤	233	Θ		
18	↕	42	*	66	B	90	Z	114	r	138	è	162	ó	186	║	210	╥	234	Ω		
19	‼	43	+	67	C	91	[	115	s	139	ï	163	ú	187	╗	211	╙	235	δ		
20	¶	44	,	68	D	92	\	116	t	140	î	164	ñ	188	╝	212	╘	236	∞		
21	§	45	-	69	E	93	]	117	u	141	ì	165	Ñ	189	╜	213	╒	237	ø		
22	▬	46	.	70	F	94	^	118	v	142	Ä	166	ª	190	╛	214	╓	238	ε		
23	↨	47	/	71	G	95	_	119	w	143	Å	167	º	191	┐	215	╫	239	∩		

THANK YOU