COMPUTER ARCHITECTURE AND SOFTWARE EXECUTION PROCESS

DATA REPRESENTATION

🎓 Bachelor in Artificial Intelligence, Data and Management Sciences
🏛 CentraleSupelec and ESSEC Business School - 2023/2024

Idir AIT SADOUNE
idir.aitsadoune@centralesupelec.fr

OUTLINE

- Character Encoding
- Number Encoding

Back to the outline - Back to the begin

WHAT DATA TO ENCODE?

- Characters and strings
- Natural, integers and fixed point numbers
- Floating point numbers
- Pictures, sounds, videos...

OUTLINE

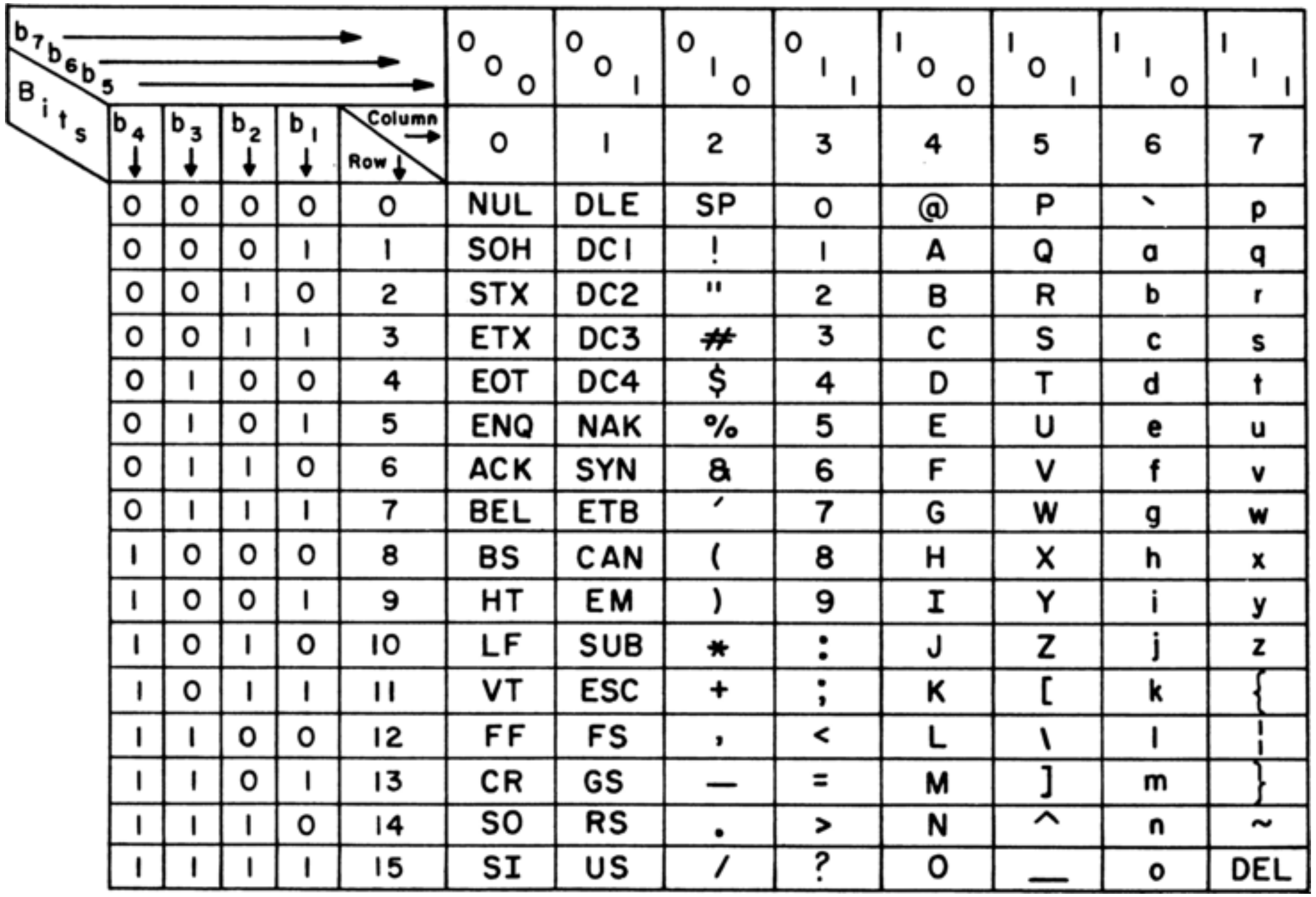
- Character Encoding
- Number Encoding

Back to the outline - Back to the begin

ASCII CHARACTER ENCODING

- **American Standard Code for Information Interchange** - **ASCII**
  - **ASCII** is a character encoding standard for electronic communication.
- **ASCII** codes represent text in computers, telecommunications equipment, and other devices.
- The first edition of the **ASCII standard** was published in 1963.
- **ASCII** has just 128 code points (7 bits + 1 as parity bit).
  - Of the 2^7=128 codes, 33 were used for controls, and 95 for printable characters

ASCII CHARACTER ENCODING



Bits				$b_7 b_6 b_5$ →	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b_4 ↓	b_3 ↓	b_2 ↓	b_1 ↓	Column → Row ↓	0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	´	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[	k	{
1	1	0	0	12	FF	FS	,	<	L	\	l	¦
1	1	0	1	13	CR	GS	—	=	M	]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	_	o	DEL

EXTENDED ASCII

- **Extended ASCII** is a repertoire of character encodings that include the original **ASCII** character set, plus up to 128 additional characters.
- In 1987 , the **ISO** published a set of standards for **8-bit ASCII** extensions, **ISO 8859**
  - **ISO 8859-1**: for the most common Western European languages.
  - **ISO 8859-2**: for Eastern European languages.
  - **ISO 8859-xxx**: ...

THE UNICODE STANDARD

- **Unicode** is a text encoding standard maintained by the **Unicode Consortium** designed to support the use of text written in all of the world's major writing systems.

- **Unicode** is used to encode the vast majority of text on the Internet, including most web pages.

- 149 813 code points in the last published version. (15.1, September 2023)

THE UNICODE STANDARD

Forme	Used bits	Code points
0xxxxxxx	7	0 to 127
110xxxxx 10xxxxxx	11	128 to 2 047
1110xxxx 10xxxxxx 10xxxxxx	16	2 048 to 65 535
11110xxx 10xxxxxx 10xxxxxx 10xxxxxx	21	65 536 to 1 114 111

STRINGS ENCODING

- The most used representation is a **character array**.
  - but an array is not directly manipulated by the processor.
- The processor needs to know the address of the beginning of the array and the index of the element it wants to access.
  - the first memory word contains the number of characters,
  - or the **ASCII** code 0 (**NULL**) indicates the end of the string (like in **C**)

OUTLINE

- Character Encoding
- Number Encoding

Back to the outline - Back to the begin

NATURAL NUMBERS

- Classic binary representation: generally used.
- **Example:** case of using 32-bit encoding
  - 214 = 00000000 00000000 00000000 11010110

- BCD, Binary-Coded Decimal :
  a class of binary encodings of decimal numbers where each digit is represented by a fixed number of bits (4 or 8).
- **Example:** case of using 32-bit encoding
  - 214 = 00000000 00000010 00000001 00000100

INTEGER NUMBERS

- Signed number representation
  - the sign bit is a bit in a signed number representation that indicates the sign of a number
  - number = sign bit + absolute value
  - **example:** $5 = 00000101$ and $-5 = 10000101$

✘ Two representations of 0

✘ Arithmetic operations cannot be implemented in electronic circuits.

INTEGER NUMBERS

- Two's complement
  - the most common method of representing signed integers

- The two's complement of an integer is computed by:
  1. starting with the binary representation of the number;
  2. inverting all bits – changing every 0 to 1, and every 1 to 0;
  3. adding 1 to the entire inverted number, ignoring any overflow

- **Example**: to calculate the number -6 in binary from the number 6
  1. 6 in decimal is 00000110 in binary (using 8-bit encoding)
  2. flip all bits in 00000110, giving 11111001.
  3. add the value 1 to the obtained number 11111001, giving 11111010.

REAL NUMBERS

- Fixed-point arithmetic
  - A fixed-point representation of a fractional number is essentially an integer that is to be implicitly multiplied by a fixed scaling factor.
  - 18.625 = 10010.101 (with 8 as a fixed scaling factor) $\rightarrow$ 10010101
    - $2^4 + 2^1 = 18$
    - $2^{-1} + 2^{-3} = 0.625$

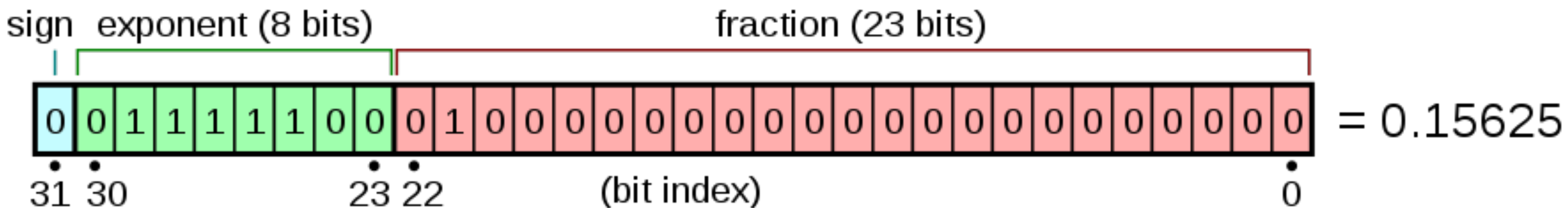
REAL NUMBERS

- floating-point arithmetic
  - is arithmetic that represents real numbers using an integer with a fixed precision (significand), scaled by an integer exponent of a fixed base.

$$x = 3.14159265359 = \underbrace{314159265359}_{\text{significand}} \times \underbrace{10}_{\text{base}}^{\overbrace{-11}^{\text{exponent}}}$$

REAL NUMBERS

- floating-point arithmetic
  - IEEE 754 standard: binary32
    - Sign bit: 1 bit
    - Exponent width: 8 bits
    - Significand precision: 24 bits (23 explicitly stored)



    - $(-1)^{sign} \times 2^{exponent-127} \times 1.fraction$
    - $exponent = 124$ and $fraction = 25$
    - $2^{124-127} \times 1.25 = 0.15625$

THANK YOU

Back to the begin - Back to the outline