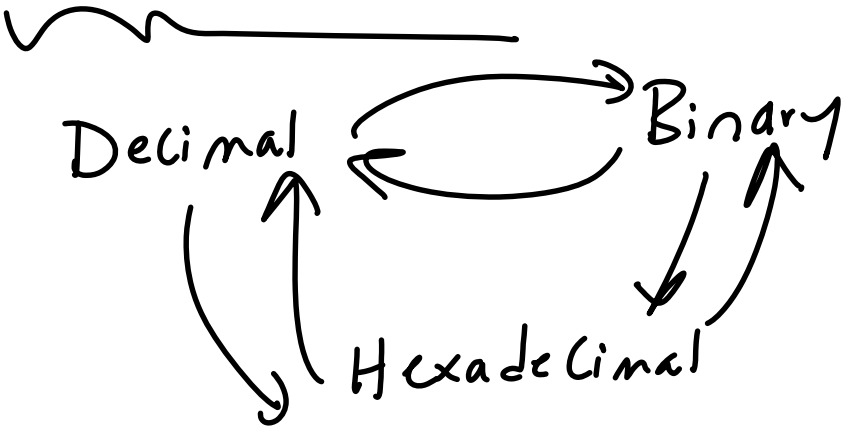
Data Representation

- Binary information in digital computers is stored in memory or processor registers
- Registers contain either data or control information
- Control information is a bit or group of bits used to specify the sequence of command signals nceded for manipulation of data in other registers
- Data
  - numbers
  - letters
    - lower case
    - upper case
  - special characters
  - binary_coded form

Numbering system



$(57)_{10} \Rightarrow$

32	16	8	4	2	1
1	1	1	0	0	1

$(101010)_2 \Rightarrow (42)_{10}$

$(60)_{10} \Rightarrow$

256	16	1
	3	C

$(60)_{10} \Rightarrow 3CH$ ↳ hexadecimal

$2BH \Rightarrow 32+11 = (43)_{10}$

FFH $\Rightarrow (1111\ 1111)_2 \Rightarrow 255$

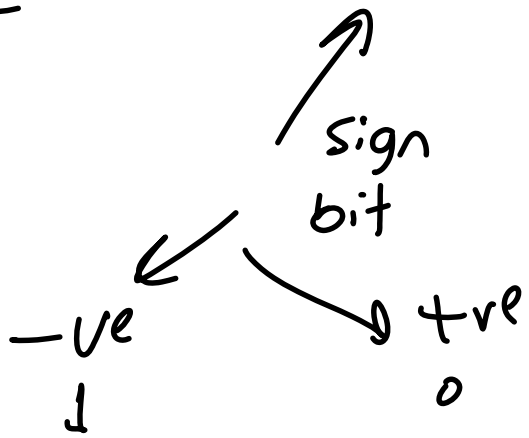
$(1010\ 0000)_2 \Rightarrow$ A0H

$\Rightarrow 160$

unsigned & signed

unsigned $\Leftarrow$ 16 8 4 2 1

signed $\Leftarrow$ -16 8 4 2 1



0101 ⟹ unsigned 5

↳ signed
+5

101010

↓ Signed

-32 16 8 4 2 1
1 0 1 0 1 0

⇓

-22

unsigned

42

Note

1011 = 111 1011 = 111111 1011
in signed form

Decimal code ⇒ every digit need 4 bits

① Binary coded Decimal BCD

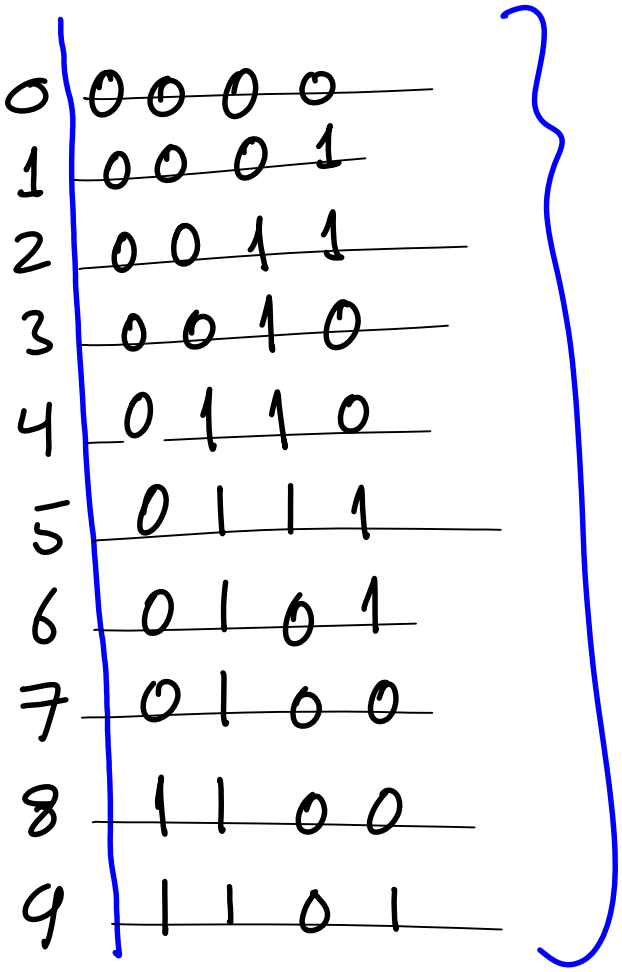
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

9 ⇒ 1001

10 ⇒ 0001 0000

23 ⇒ 0010 0011

② Gray – Reflected – cyclic



only one bit change

9 ⇒ 1101

20 ⇒

0011 0000

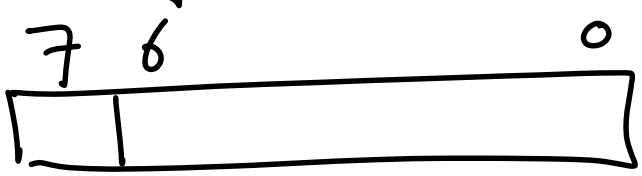
36 ⇒

0010 0101

Alphanumeric Code

① ASCII

- 8-bit
- 7-bit



Sign bit

Parity bit

A character ⇒ 41 H

100 0001

Space bar ⇒ 20 H ⇒ 010 0000

3 ⇒ 33 H ⇒ 011 0011

Note

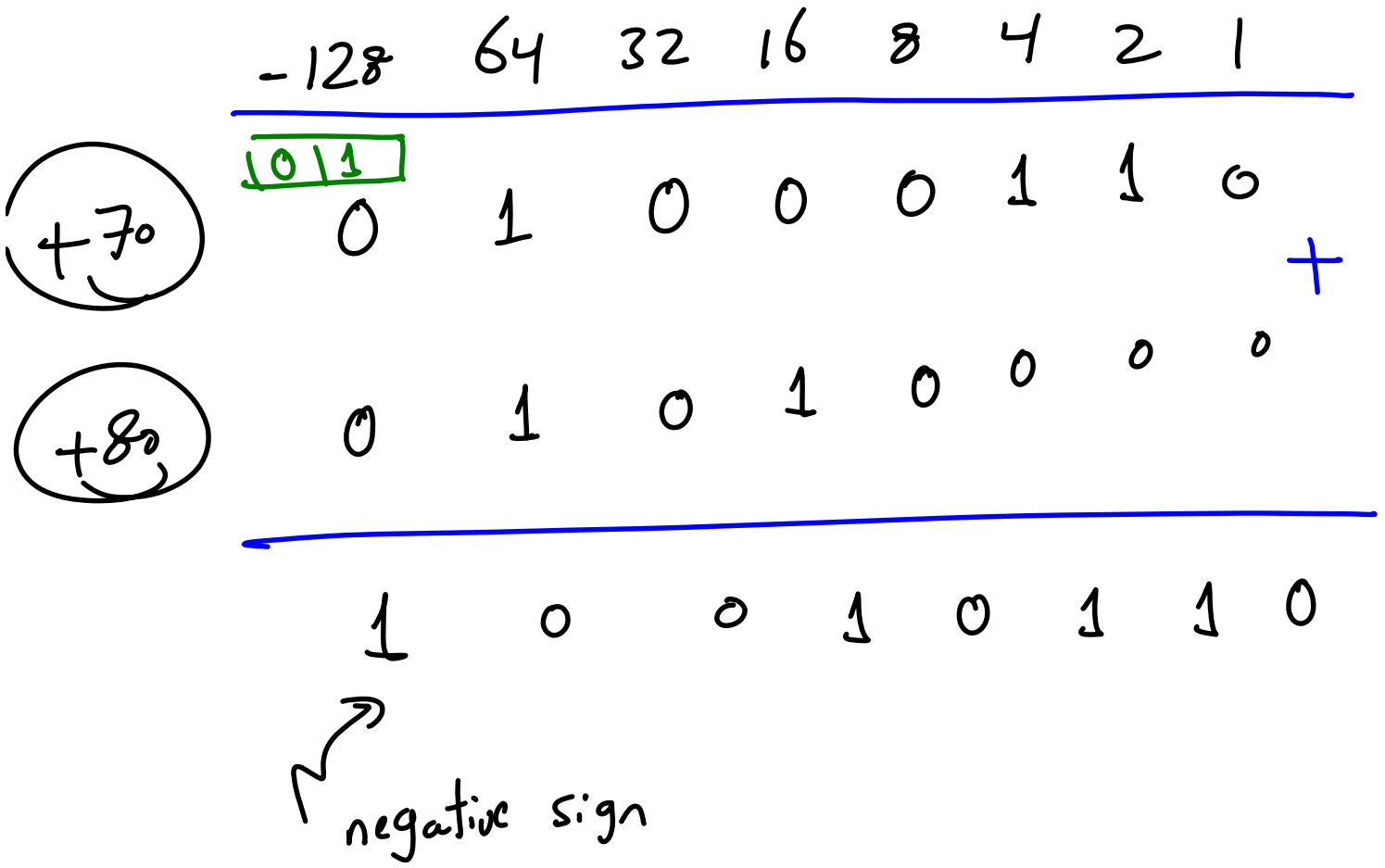
- Binary codes can be formulated for any set of discrete elements such as the musical notes and chess pieces and their positions on the chessboard
- Binary codes are also used to formulate instructions that specify control information for the computer

Overflow

- When two numbers on n-digits are added and the sum occupies $n+1$ digits $\longrightarrow$ we say that an overflow occurred

- When the addition is performed with paper and pencil, an overflow is not a problem since there is no limit to the width of the page to write down the sum
- an overflow is a problem in digital computers because the width of register is finite

Ex: Two signed binary numbers are stored in two 8-bit register $+70, +80$

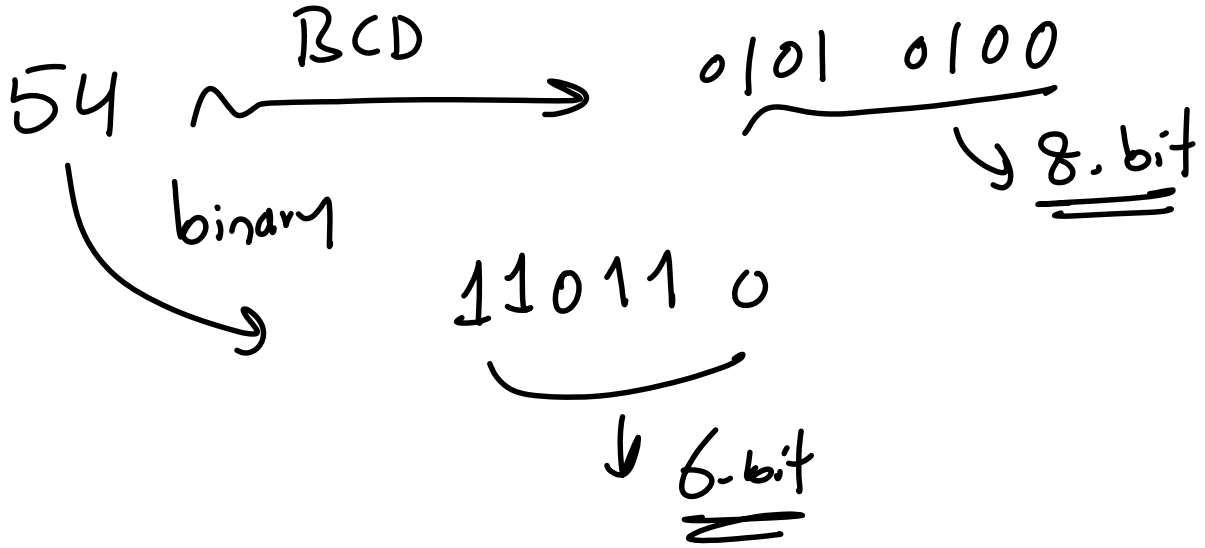


How processor know that overflow happened

- Signed number
- Same sign
- different carry

Note

By representing numbers in decimal we are wasting a considerable amount of storage space since the number of bits need to store a decimal number in a binary code is greater than the number of bits needed for its equivalent binary representation



— also, the circuits required to perform decimal arithmetic are more complex

— but, there are some advantages in the

use of decimal representation

↳ Computer input & output are generated by people who use the decimal system

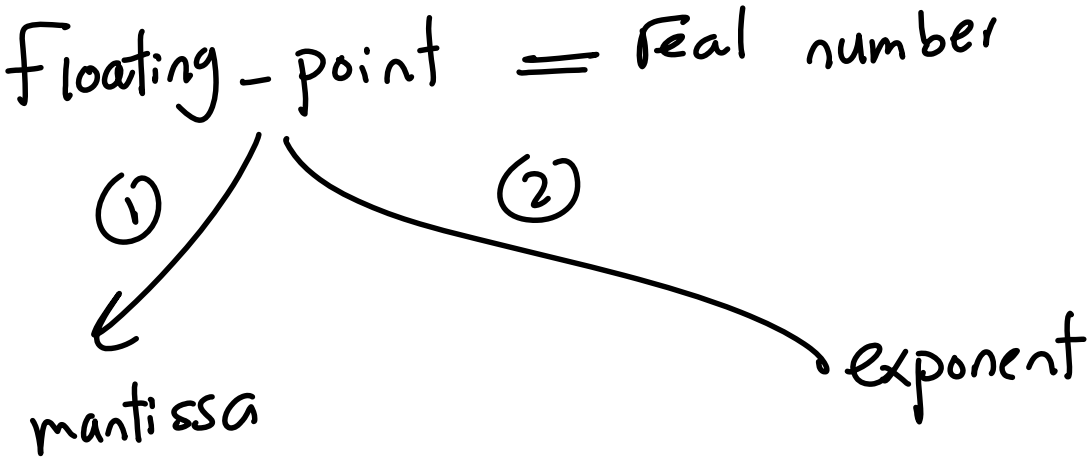
Floating-point Representation

- To represent very large integer (whole) numbers — many bits are required
- There is also a problem when numbers with both integer and fractional parts — 23.5612

- Floating-point number system is capable of representing

Whole integer numbers

fractional number



Ex

241,506,800

↳ to binary

0.2415068×10^{9}

mantissa

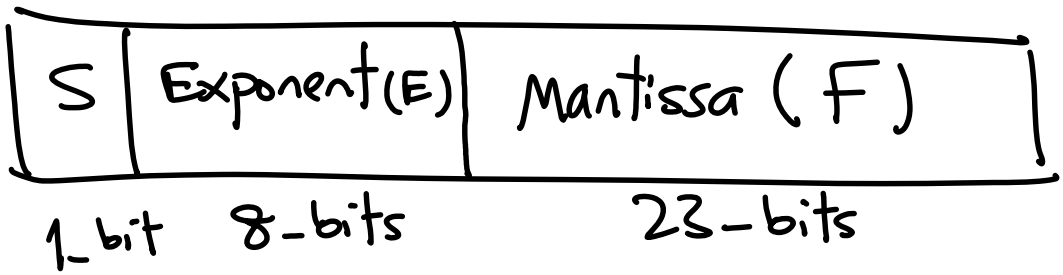
exponent

binary floating-point number

1. single-precision ⇒ 32-bit
2. double-precision ⇒ 64-bit
3. extended-precision ⇒ 80-bit

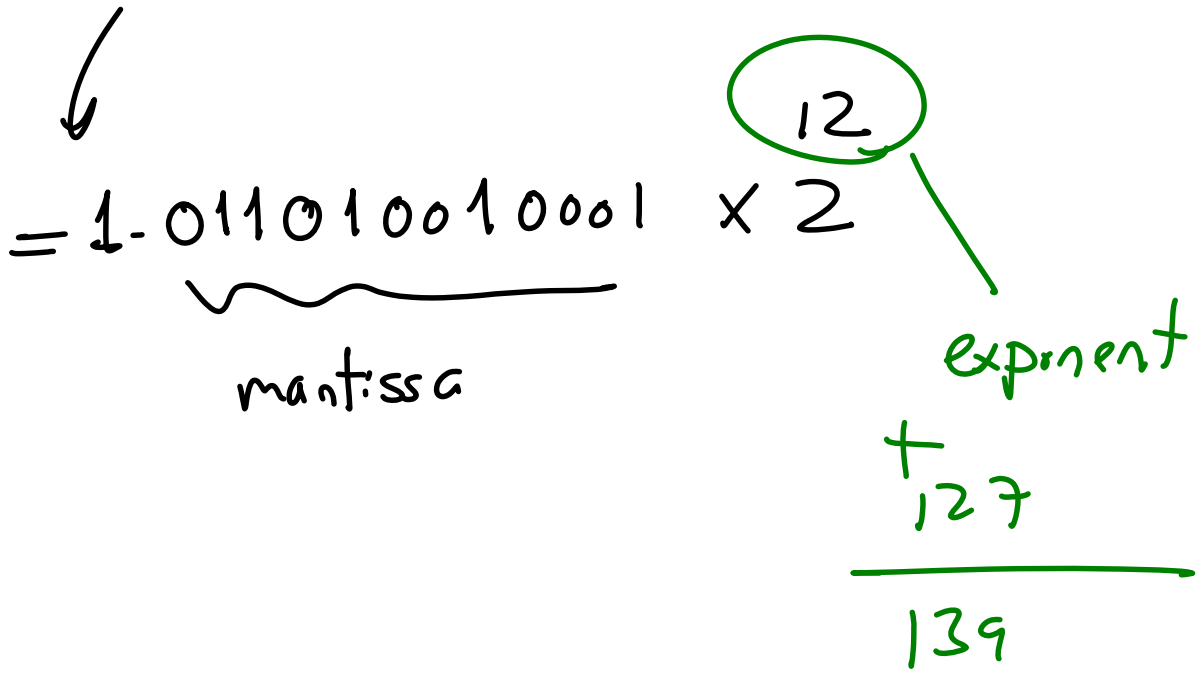
1. Single-precision

S: Sign E: Exponent
F: Fraction



⇓
biased exponent (+127)

Ex ⇒ 1011 0100 10001



S	E	F
0	1000 1011	01101001 0001 0000 0000 000

Ex

S	E	F
1	1001 0001	1001 1100 0010 0000 0000 0000

$$\text{Number} = (-1)^{S}(1+F)(2^{E-127})$$

$$= -1.10001110001 * 2^{145-127}$$

$$= -1.10001110001 * 2^{18}$$

$$= -11000111000100000000$$

$$= -407,688$$

Prob Convert 3.248×10^4 to single-point precision