CAPÍTULO 3 - REPRESENTATION OF NUMBERS

Positional Numeral Systems (pns)

Numeral Systems

↳ writing system for expressing numbers

↳ it's a notation for representing numbers of a given set, using digits or other symbols

↳ binary aritmetic system and decimal numeral system are both positional numeral systems

↳ they represent any number by a sequence of juxtaposed digits

dígito ou algarismo é o mesmo ≠ número
número é constituído por algarismo(s).

Digit and Radix

↳ digit is a symbol used to represent nºs in pns

↳ the position of each digit has a corresponding weight

↳ the mathematical value of a number is the weighted sum of all digits

↳ for decimal numbers the weights are powers of ten equal to 10^i

↳ 10 is called radix (or base) of the numeral systems

Natural Numbers

↳ pns can use any integer ≥ 2 for the base

↳ the digit in position i has weight r^i

general form of a n-digit number D system is $d_{n-1}, d_{n-2} \ldots d_1 d_0$.

↳ the value of D (natural nº) is $\sum_{i=0}^{n-1} d_i \cdot r^i$

↳ digital devices adopt the binary base ($r=2$)

↳ the representation for natural number involves a fixed number of bits and is known as the natural binary numeral system

Decimal Numbers

- Fractions are considered by using negative powers of radix
- the integer part of a number is separated from its fractional part by a radix point
- binary fractions have a bynary point

Least / Most Significant Digits

LSD - the rightmost digit (least significant)

MSD - the leftmost digit (most significant)

LSB - in binary numbers LSD is least significant bit

MSB - in binary numbers MSD is most significant bit.

posição 0

$$00110101_2 = 53_{10}$$

$$V = 1\times2^5 + 1\times2^4 + 1\times2^2 + 0\times2^1 + 1\times2^0$$
$$= 32 + 16 + 4 + 1 = 53$$

$$00110101_2.011$$

$$V = 1\times2^5 + 1\times2^4 + 1\times2^2 + 0\times2^1 + 1\times2^0 + 1\times2^{-2} + 1\times2^{-3}$$
$$= 32 + 16 + 4 + 1 + \frac{1}{4} + \frac{1}{8} = 53{,}375_{10}$$

Octal and Hexdecimal Numbers

Bases 8 and 16

(base 10 is used in everyday activities by humans)

- Base 2 is also relevant, as binary numbers are processed by digital devices.
- Binary notation is too verbose so base 8 and 16 are used instead.
- They offer a shorthand representation of binary nºs
- The octal system uses 8 digits: From 0 to 7
- The hexadecimal system uses 16 digits: From 0 to 9 and From A to F

Converting from Base 2 to 8-16

Bases 8 and 16 are useful for representing multi-bit numbers since they're integer powers of 2

Octal digits are represented by 3-bit strings

Hexadecimal digits are 4-bits strings

$$51_8 = 00101001 = 29_{16}$$

5 1 em binário

Converting binary into octal starts at the end and forms groups of 3 to the left. Zeros can be added on the left bcs they don't change the value.

If the binary has digits to the right of the binary point, groups of 3 or 4 are formed from the right and zeros can be added to the right of the rightmost bit

$.11001_2 = .110\ 010_2 = .62_8$

$= .1100\ 1000_2 = .C8_{16}$

Conversions between Different Bases

177 | 16
17 11 | 16
1 11 0

$177_{10} = B1_{16}$

177 | 8
17 22 | 8
1 6 2 | 8
2 0

4,1º bit

$177_{10} = 261_8$

$261_8 = 2 \times 8^2 + 6 \times 8 + 1$

$= 128 + 48 + 1 = 177_{10}$

$120.01_3 = 1 \times 3^2 + 2 \times 3^1 + 0 \times 3^0 + 0 \times 3^{-1} + 1 \times 3^{-2} = 15.(1)_{10}$

$AB3_{16} = 10 \times 16^2 + 11 \times 16 + 3 \times 16^0 = 2739_{10}$

Often it's easier to convert to decimal base and then to the target base

Negative Numbers

Representations - requires the inclusion of a sign bit

Sign-Magnitude:

The most intuitive method. Sign-magnitude uses:

the MSB for the sign bit

the remaining bits → magnitude (its absolute value)

8 bits 0 — 255 → 256 numbers

128 negativos | zero | 127 positivos

1 1 1 1 1 0 0 0

64 32 16 8

80 + 40 = 120

0000 0000 = +0

1000 0000 = -0

By convention: 1 - negative

0 - positive

Sign-magnitude contains same nº of positive and negative

" integer representation w/ n bits ranges from

$-(2^{n-1}-1)$ to $+(2^{n-1}-1)$

For 8 bits: -127 to +127

" calculations are the same as traditional addition

One's Complement

generally WORKS for decimals

```
  135          + 135          + 135
 +977          -  22           978
 ----          -----          -----
 X112          + 113          X113
+   1
 ----
 +113
```

↳ Complementar para 9 (1000 - 1)

↳ Complementar para 10

O "1" adicionado é o "carry"

WORKS similarly in the one's complement numeral system

FOR binary: MSB is $2^{n-1}+1$ instead of 2^{n-1}

$0110011_2 = 1 \cdot 16 + 1 \cdot 8 + 1 \cdot 1 = 25_{10}$?

One's complement → subtracting from all ones

Given: number V, base R, n digits

$R^n - 1 - V$ ↳ diminished radix complement

one's complement of $0101_2 = 1111 - 0101 = 1010_2$

$10_{10} = 0000\,1010_2$

8 2 MSB

1 1 1 1 0 1 0 1

-127 | 32 | 4 | 1

16

64

$-127 + 80 + 32 = -10$

(127 - 10)

SE FOR POSITIVO: FICA =

" " NEGATIVO: 0 passa a 1 / 1 passa a 0 } MSB

MSB will have a CARRY { 0 / 1 } that must be added

```
  1 1 1
  0 0 0 1 1 0 0 0      CARRIES
  1 1 1 1 0 1 0 1        +24_10
 ---------------     + (-10_10)
```

```
  0 0 0 0 0 1 1 0 1
          +     1
  ---------------------
  0 0 0 0 0 1 1 1 0
```

$+14_{10}$

```
  ' '   1
      0
  -------
      1
```

Most computers use the two's-complement numerical system

Radix complement $\xrightarrow[\text{intuitive}]{+}$ diminished radix C.

nadix complement:

$r^n - V$ for $V \neq 0$ and 0 for $V = 0$

$01100\,1_2 = 1 \times 16 + 8 \times 1 + 1 \cdot 1 = 25_{10}$

$111001_2 = 1 \times (-32) + 1 \times 16 + 1 \times 8 + 1 \times 1 = -7_{10}$

↓

último bit, smp negativo, último dos algarismos

two's complement is one's complement + 1

↳ for binary: flip bits, add 1

CARRIES are discarted

only negative need to be converted to two's complement

⟶ two's complemen ⟵

Fixed-Point Numbers

0 —— 255

.00000000 —— 11111111

The value of a fixed-point number is an integer that is scaled by a factor determined by the position of the radix point.

$1 \leq \text{mantissa} \leq 10$

$\text{mantissa} \times \text{radix}^{\text{exponent}}$

ex: $3{,}42 \times 10^{2}$

Floating-point is a fixed-point ...

Floating-point representations offer more range and less precision than fixed-points

" consist of three pointes: a sign-bit, mantissa (or significand), and exponent (a power of 2)

If fpn has 16 bit has a sign bit, a 5-bit exponent and a 10-bit fractional part of the mantissa.

float → 32 bits double → 8 bits } in C

The values encoded by a bit pattern are different, depending on the value of the exponent:

1- normal numbers;

2- subnormal numbers;

3- special values;

FORMULA FOR CASE 1:

$$V = (-1)^{s} \times (1+f) \times 2^{e-bias}$$

Mantissa = $1+f$, $1 \leq M < 2$, bcs the 1st bit is 1

Sign bit: if $s=0$ then V is positive

if $s=1$ then V is negative

bias: $2^{K-1}-1$, K is the n° of exponent bits

exponent: encoded in an excess format

any number larger than 15 in the exponent fields represents a positive exp?

For this formula has to be $1 < e < 30$

all zeros or all ones, is not a normal number

$21,5_{10} = 16_{10} + 4_{10} + 1_{10} + 0.5_{10}$

$= 1\ 0\ 1\ 0\ 1\ .\ 1_2 \times 2^0$

$= 1.0\ 1\ 0\ 1\ 1 \times 2^4$

$4 = e - 15 \iff e = 19$

$= 1.0\ 1\ 0\ 1\ 1 \times 2^{19-15}$

$(-1)^0 \times 1.01011\,00000 \times 2^{10011_2 - 15}$

FORMULA FOR CASE 2:

Mantissa está desnormalizada, $0 \leq M < 1$, $M = 0 + f$

$V = (-1)^S \times M \times 2^{-14}$ ← o expoente é fixo

00000_2 ↑? 11111_2 ↓?

SPECIAL VALUES

if: I

+ ∞ e - ∞ if S=0

Can be used to indicate an error OR to represent non-initialised data

contain NaN

↳ M is nonzero

$0.1111111111 \times 2^{-14}$ Biggest subnormal

$0.0000000001 \times 2^{-14}$ Smallest subnormal

$1.0000000000 \times 2^{-14}$ Smallest normal number

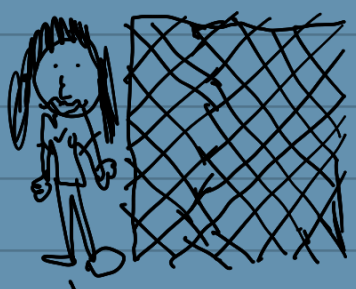
IEE 754 32 bits 1 - 8 - 23

64 bits 1 - 11 - 52

(16 bits 1 - 5 - 10) what we used, not part of IEE 754 (acho?)

↳ Both single and double precision

gorgolu



araújo

gorgolu

araújo