Sistema Computacional: (1)

Níveis de um computador

Nível 6 → user

Nível 5 → problem oriented language) compiler

Nível 4 → assembly language) assembler

Nível 3 → operating system machine) operating system

Nível 2 → instruction set architecture

Nível 1 → microarchitecture) direct execution or interpretation (microprogram)

Nível 0 → digital logica) hardware

Diferenças entre compiler, interpreter e ~~machine~~ virtual machine

Compiler → compila alto para baixo (rápido)

interpreter → interpreta e compila (mais lento)

virtual machine → simulador de um sistema de computador que é executado numa maquina física simulando um computador real.

Big-endian, little-endian computers

Big-endian → os valores menos significativos do numero vão para o maior endereço da memória.

Ex: 00010010 00110100 01010110 01111000 (byte 2, byte 0 — menos significativo)

Memória	Big	Little
132104 →	00010010	01111000
132105 →	00110100	01010110
132106 →	01010110	00110100
132107 →	01111000	00010010

Little-endian → os valores menos " " vão para o menor endereço

Representação de informação:

Imagens:

128 x 128 pixels, 3 chanels, 256 diferent cores   1 canal → 8 bits ↳ 8 x 3 = 3 bytes

128 x 128 = 16384 pixeis

3 x 16384 → tamanho em bytes

KB Kilobyte → 10^3
MB megabyte → 10^6
GB gigabyte → 10^9
TB terabyte → 10^{12}

KiB Kibibyte 2^{10} 1024
MiB mebibyte 2^{20}
GiB gibibyte 2^{30}
TiB tebibyte 2^{40}

Sample resolution
É o numero de bits usados para representar o valor para cada sample

$s = f \times r \times t$

f → sampling rate (H_z)
r → is the sample resolution e numero de bits (bits)
t → tempo (s)

Representação de numeros

4 682 na base 10

$4682 = 4 \times 10^3 + 6 \times 10^2 + 8 \times 10^1 + 2 \times 10^0$

Passagem de valores para binário

$4682{,}05 = (\ldots) + 0 \times 0{,}1 + 5 \times 0{,}01$

1 → Dividir por 2
2 → É o resto da divisão
3 → Ler de baixo para cima

$0 - 0_{10}$
$1 - 1_{10}$
$2 - 2_{10}$
$3 - 3_{10}$
$4 - 4_{10}$
$5 - 5_{10}$
$6 - 6_{10}$
$7 - 7_{10}$
$8 - 8_{10}$
$9 - 9_{10}$

Passagem de Binário para Base 10 (Base 2) (decimal)

$\underbrace{1001}_{2^3+2^0 = 9}\ \underbrace{0110}_{2^2+2^1 = 6} = 96$

Passar para base 10

$3410_5 = 3 \times 5^3 + 4 \times 5^2 + 1 \times 5^1 + 0 = 480_{10}$

---

numero numa base $= 1$ digito $\times n^m$

n → base que está m → numero do bit

Exemplos de conversões

$177_{10} = 10110001_2$

$177_{10} = 261_8$

$177 \div 8 = 22$ resto 1
$22 \div 8 = 2$ resto 6
$2 \div 8 = 0$ resto 2

Podemos converter assim
Como podemos converter para base 10 primeiro e depois para a base que queremos

---

$177_{10} = B1_{16}$ → a base 16 é tipo a 10 mas com 16 valores

$177 \div 16 = 11$ resto 1
$11 \div 16 = 0$ resto 11

3

Números negativos.

1 → fazer o número em binário

2 → trocar os 1 para 0 e vice versa

(o primeiro valor do binário diz se é negativo ou positivo

3 → Somar 1

3.22) a) 0011100_2) 1º Completo

$$\begin{array}{r} 1100011 \\ +\ 0000001 \\ \hline \boxed{1100100} \end{array}$$

2º Completo

Somar é tipo normal mas se o valor chegar ao valor da base mete-se a ir um

---

$\{(10^n - 1) - N\}$ 9's complement of the number
n → número de dígitos
N → é o número

1236 → $(10^4 - 1) - 1236 = 10000 - 1 - 1236$
$= 10000 - 1237 = 8763$

3.22)
a) $+18_{10}$

① sinal + magnitude

$18_{10} = 10010_2$

$\boxed{\emptyset}\ \emptyset\ \emptyset\ 1\ 0\ 0\ 1\ 0_2$

+ : 0
− : 1

acrescenta-se porque tem de ser 8 bit

② comp/1
Como o nº é positivo o complemento é o próprio

$\emptyset\ \emptyset\ \emptyset\ 1\ 0\ 0\ 1\ 0_2$

③ comp/2

$\emptyset\ \emptyset\ \emptyset\ 1\ 0\ 0\ 1\ 0_2$

b) $+121_{10}$

121 | 2 → 1, 60; 60 | 2 → 0, 30; 30 | 2 → 0, 15; 15 | 2 → 1, 7; 7 | 2 → 1, 3; 3 | 2 → 1, 1; 1 | 2 → 1, 0

$121_{10} = 1111001$ $\quad \emptyset\ 1\ 1\ 1\ 1\ 0\ 0\ 1$

Comp 1 igual, Comp 2 igual

c) -33   4   (repetição ✱)

33 | 2 → 1, 16   16 | 2 → 0, 8   8 | 2 → 0, 4   4 | 2 → 0, 2   2 | 2 → 0, 1   1 | 2 → 1, 0

$33_{10} = 100001_2$ $\cancel{1}\cancel{1}1\,0000\,1_2$

0001111 0

000 11111

R: 011111 ,, $2^4+2^3+2^2+2^1+2^0 = 16+8+4+2+1 =$

Complemento de 2

- Número positivo: bit de sinal 0 e a magnitude representada na forma binária direta
- Número negativo: bit de sinal 1 e a magnitude representada na forma de Complemento de 2

Ex: $\underbrace{0}\ \underbrace{0110}_{mag} = +6$ mas 1 0110 não é 6 porque é na forma do Complemento de 2

~~(seja se quiser -6 e)~~

1 0110 → -10

  0 1001
+ 0 0001
= 0 1010 → +10

---

1 1001 → 00110
+ 00001
= 00111 → +7 logo 11001 é o -7

$0\times2^4+0\times2^3+2^2+2^1+2^0 = 4+2+1 = 7$

de + para - é fazer o (+1) e depois inverso

✱ é mudar e depois +1 se for negativo

c) -33 $33_{10} = 100001_2$ 0100001, 1011110

$\boxed{0100001_2}$

  0100001
- 0000001
= 0100000

1011111$_2$

+1 ... 11011110 → -? $= 2^7+2^6+2^4+2^3+2^2$

$2^4+2^3+2^2+2^1+2^0 = 16+8+4+2+1 = \boxed{31}$ → do 31 porque é o complemento de 2 de 33 o valor

Binário negativo mais +2

Ex: -7 +2 = -5

  1 1001
+ 0 0010
= 1 1011

Confirmação → 00100 + 1 = 00101   0101 = 5 ✓

(5) Overflow

$$\begin{array}{r} +9 \\ +8 \\ \hline +17 \end{array} \qquad \begin{array}{r} 01001 \\ 01000 \\ \hline 10001 \end{array}$$

↳ tá a dar negativo

4 bits só chega até ao 15

logo ↳ $1111 = 2^3 + 2^2 + 2^1 + 2^0$
$= 8 + 4 + 2 + 1$
$= 15$
é o maior

$$\begin{array}{r} 001001 \\ 001000\ + \\ \hline 010001 \end{array}$$

→ +17 acrescentar 1 bit

↳ positivo

Carry (é 0 ou 1) se for no bit do sinal é ignorado

vai 1
fica só 0

$$\begin{array}{r} 01001 \\ +\ 11100 \\ \hline \underline{0}0101 \end{array}$$

---

s, e → número, f → parte fracional

s	e			f				
0								

rever melhor esta última parte

$21,5 = 10101,1_2 = 10101,1_2 \times 2^0 = 1,01011_2 \times 2^4$

s → sign bit (+ ; -)

e → exponent part (representado em base de 2)

f → ~~mantissa~~ fractional part of the mantissa

Número normal

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

$$Bias = 2^{k-1} - 1$$

k → nº de bits do e

Bits:

$+3.14 \times 10^6$

$\underset{s}{\boxed{}}\ \underbrace{\_\ \_\ \_}_{e}\ \underbrace{\_\ \_\ \_\ \_\ \_\ \_}_{f}$

$e \downarrow$ expoente

$f \downarrow$ parte fracionária

Expoente 2

$\boxed{F_1}$

$\underset{s}{\_}\ \underbrace{\_\ \_\ \_\ \_}_{e}\ \underbrace{\_\ \_\ \_}_{f}$

$e \rightarrow$ expoente

$\boxed{F_2}$

$\underset{s}{\_}\ \underbrace{\_\ \_\ \_}_{e}\ \underbrace{\_\ \_\ \_\ \_}_{f}$

bit escondido não tem memória

$11001_2 = \underline{1,1001 \times 2^4}$

está normalizado

$bias = 2^{4-1} - 1 = 2^{n-1} - 1$ $\quad n \rightarrow$ número de bits do expoente

$\boxed{F_1}$ $V_n = (-1)^s \times 1.f_2 \times 2^{e-bias}$ $\quad \underset{s}{\boxed{}}\underset{e}{\boxed{}}\underset{f}{\boxed{}}$

$bias = 2^{4-1} - 1 = 7$

$V_n = (-1)^s \times 1.f \times 2^{e-7}$