MATA49
Programação de Software Básico

Sistemas de números

Leandro Andrade
leandrojsa@ufba.br

Conceitos básicos

- Os códigos binário pelo tamanho de caracteres podemos classificar como:
  - Bit: um único dígito
  - Nibble = 4 bits
  - Byte: 8 dígitos = 8 bits
  - Word: 2 bytes = 16 bits
  - Double word: 2 Words = 4 bytes
  - Quad Word: 4 Words = 8 bytes

Representação Decimal

- Número representados na base 10
  - 0, 1, 2, 3, 4, 5, 6, 7, 8 , 9
- O computador na sua visão interna não armazena número decimais
- Assim:

$$234 = 2 \times 10^2 + 3 \times 10^1 + 4 \times 10^0$$

- No mais...sem novidades!

Representação Binária

- Número representados na base 2
  - 0, 1
- Representação utilizada pelos computadores
  - Um número binário com um único dígito é chamado de **bit**
  - Com 8 dígitos é chamado de **byte**
- Exemplo:

$11011 = 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 16 + 8 + 0 + 2 + 1 = 27$

Representação Binária

- Conversão um decimal para binário
  - Divide-se o número sucessivamente por 2 e analisa-se o resto de cada divisão
- Exemplo:

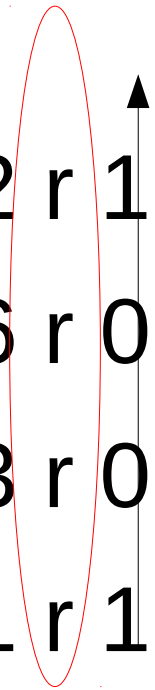
25 / 2 = 12 r 1

12 / 2 = 06 r 0

06 / 2 = 03 r 0

03 / 2 = 01 r 1

01 / 2 = 00 r 1



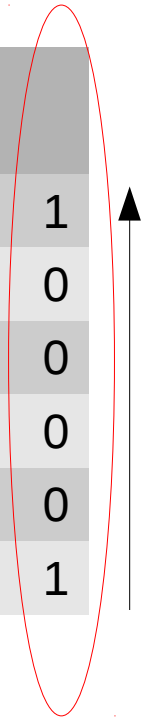
$25_{10} = 11001_{2}$

Representação Binária

- Mais um exemplo:

$33_{10} = ?_{2}$

Nº corrente	Resultado da divisão por 2	Resto
33	16	1
16	8	0
8	4	0
4	2	0
2	1	0
1	0	1



100001

Representação Binária

- Mais exemplos:

$43_{10} = ?_{2}$

43	21	1
21	10	1
10	5	0
5	2	1
2	1	0
1	0	1

101011

$19_{10} = ?_{2}$

19	9	1
9	4	1
4	2	0
2	1	0
1	0	1

10011

$57_{51}0 = {}_{?}{}^{2}$

57	28	1
28	14	0
14	7	0
7	3	1
3	1	1
1	0	1

111001

Representação Binária

- Adição e Subtração:

13 + 9 = 22

```
  1101
+1001
-------
10110
```

13 - 10 = 3

```
  1101
- 1010
---------
  0011
```

Representação Hexadecimal

- Os número representados na base 16
  - [0–9], A(10), B(11), C(12), D(13), E(14), F(15)
- Exemplo:

$2BD_{16} = 2 \times 16^2 + 11 \times 16^1 + 13 \times 16^0$

$2BD_{16} = 512 + 176 + 13$

$2BD_{16} = 701$

Representação Hexadecimal

- Hexadecimal ⟺ Binário:

  0110 0000 0101 1010 0111 1110

  6 0 5 A 7 E

- Decimal para Hexadecimal:

  589 = 24D

Nº corrente	Resultado da divisão por 16	Resto
589	36	13
36	2	4
2	0	2

Representação dos números inteiros

- Representação binária;
- Sem sinal:
  - Simples!
  - Ex: $200_{10} = 11001000_{2}$
- Com sinal:
  - -11001000_{2} e $+11001000_{2}$??
  - Como representar + e – para máquina?

Representação dos números inteiros

- Uso do bit mais significativo para representação do sinal
  - **0** inteiro positivo
  - **1** inteiro negativo
- Mas como obter o valor numérico do inteiro?
  - Temos 3 métodos...

Representação dos números inteiros

- Magnitude do sinal (*Signed magnitude*):
  - O bit mais significativo representa o sinal do número. Quando for **0 é positivo** e **1 é negativo**
  - Ex: +56 = **0**0111000 e -56 = **1**0111000
  - Problemas!
  - Duas representações para o zero: +0 (**0**0000000) e −0 (**1**0000000)
  - Para operar a soma entre -10 e +56 é necessário aplicar uma subtração

Representação dos números inteiros

- Complemento de 1
  - Inverte todos os bits
  - Ex: +56 = **0**0111000 e -56 = **1**1000111
  - Ainda problemas...
  - Aritmética complicada
  - E duas representações do zero (00000000 e 11111111)

Representação dos números inteiros

- Complemento de 2:
  - Utilizado pelas arquiteturas atuais
  - Resolve os problemas dos métodos anteriores
    - Única representação binária do zero
    - Simplifica operações aritméticas

Representação dos números inteiros

- Complemento de 2:
  - Conserva a representação dos números positivos (EX: +56 = **0**0111000)
  - Para os números negativos:
    - Aplica-se o complemento de 1
    - Soma-se ao resultado 1_2 (2_{10})

Representação dos números inteiros

- Complemento de 2:
  - Por exemplo:

$$00111000\ (+\ 56_{10})$$

$$11000111\ (\text{Complemento de 1})$$

$$+1\ (\text{Soma 1})$$

$$11001000\ (-\ 56_{10})$$

  - Se aplicarmos o complemento de 2 sobre o resultado obteremos $+56_{10}$

Representação dos números inteiros

- Complemento de 2:
  - Operações aritméticas sem alteração:

    11001000 (-56)

    00110000 (+48)

    11111000 (-8 em complemento 2)

    00000111 (complemento de 1)

    \+ 1

    00001000 (+ 8)

Representação dos números inteiros

Representação dos números inteiros

- Seja x um número inteiro positivo. Logo:

  x + y = 0 se e somente y = -x

- Assim: y = 0 – x

  O valor “0” pode ser representado 1000_2 se a representação numérica for de 3 dígitos

  - O último digito é desprezado
  - 1000 = 000 (em uma escala de 3 dígitos)
  - 100000000 = 00000000 (escala de 8 dígitos)

Representação dos números inteiros

- Assim como temos y = 0 – x

  Supondo x = 3 e uma escala de 3 dígitos

  y = 0 – 3

  y = 000 – 011

  y = (111 + 001) – 011

  y = 111 + 001 – 011

  y = 111 – 011 + 001

  y = 100 + 001 (aplicado complemento de 1)

  y = 101 (aplicado complemento de 2)

Representação dos números inteiros

- Complemento de 2:
  - Única representação do zero

    00000000

    11111111 (complemento de 1)

    \+ 1

    100000000 (1 mais a esquerda descartado)

Representação dos números inteiros

- Exemplos:

- 49	**- 37**	**- 62**
00110001 (+49)	00100101 (+37)	00111110 (+62)
11001110	11011010	11000001
+1	+1	+1
---	---	---
11001111	11011011	11000010

Representação interna de caracteres

- São definidos a partir de padrões
- Um conjunto de binários específicos são reservados para representar os caracteres
  - Ex: No padrão Bla 'A' = 100101
  - Mas como o processador diferencia se o código 100101 está representando 'A' ou o inteiro 37?
    - O programa que informa como esse dado deve ser interpretado

Representação interna de caracteres: ASCII

- ASCII: American Standard Code for Information Interchange
  - Padrão de codificação de caracteres e códigos
  - Definido em 1977
  - Inicialmente usava 7-bits e representava 128 caracteres
  - Exemplo: ‘A’ = (1000001) = (65)

Representação interna de caracteres: ASCII

	000	001	010	011	100	101	110	111
0000	NULL	DLE		0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EDT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

Representação interna de caracteres: ASCII

- Foi posteriormente estendido para 8-bits (1 byte)
  - Passou a codificar 256 caracteres e códigos
- Alguns códigos comumente usados:
  - NULL 00 null
  - DEL 7F delete
  - LF 0A avanço de linha
  - HT 09 tabulação horizontal

Representação interna de caracteres: ASCII

- Exemplo:

	Binary		Hexadecimal		Decimal
H =	01001000 =		48	=	72
e =	01100101 =		65	=	101
l =	01101100 =		6C	=	108
l =	01101100 =		6C	=	108
o =	01101111 =		6F	=	111
, =	00101100 =		2C	=	44
=	00100000 =		20	=	32
w =	01110111 =		77	=	119
o =	01100111 =		67	=	103
r =	01110010 =		72	=	114
l =	01101100 =		6C	=	108
d =	01100100 =		64	=	100

Representação interna de caracteres: UNICODE

- 256 códigos para mapear caracteres não comportam todas os idiomas
- Formou um consórcio internacional para definir um padrão de codificação para todas as línguas
- Cada caractere é representada por 16 bits
  - 65536 combinações diferentes
  - Os primeiros 256 caracteres têm o mesmo valor do ASCII