SCC 124 - Introdução à Programação para Engenharias

Códigos numéricos

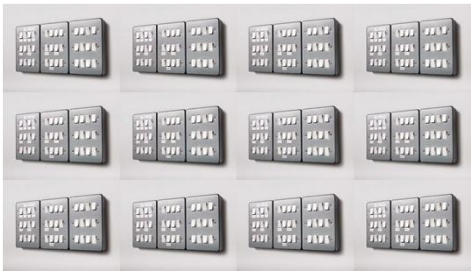
Professor: André C. P. L. F. de Carvalho, ICMC-USP
Pos-doutorando: Isvani Frias-Blanco
Monitor: Henrique Bonini de Britto Menezes

Aula de hoje

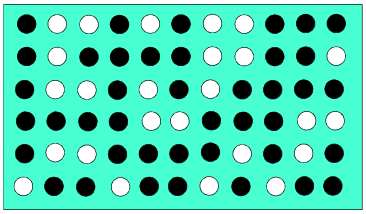
- Sistemas numéricos
- Conversões entre sistemas numéricos
- Aritmética com números binários
  - Soma
  - Subtração
  - Multiplicação
  - Divisão
- Códigos binários

Armazenamento de dados

Armazenamento de dados



Armazenamento de dados



Códigos binários

- Computadores trabalhem internamente com números em códigos binários
  - Dispositivos de I/O trabalham geralmente com números decimais
  - Maioria dos circuitos lógicos aceitam apenas sinais com dois valores
    - Cada dígito decimal deve ser codificado por valores binários

Sistemas numéricos

- Principais sistemas numéricos
  - Decimal
    - 0, 1, ..., 9
  - Binário
    - 0,1 (1011_b)
  - octal
    - 0, 1, ..., 7 (230.4_o)
  - Hexadecimal
    - 0, 1, ..., 9, A, B, C, D, E, F ($3B2_h$)

Sistema binário

- Base binária
  - Números são combinações de valores binários (0 e 1)
  - Fácil conversão de, e para, outras bases
    - Ex. Decimal
  - Existem outros códigos binários
    - Usados com outros propósitos
    - Ex. Código cinza

Decimal	Binário	Código cinza
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

Sistemas mais usados

Decimal	Binário	Octal	Hexadecimal
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Conversão base x - base 10

- num_d = parte-inteira$_d$. parte-fracionária$_d$
  = $a_n a_{n-1} \ldots a_0.a_{-1}a_{-2} \ldots a_{-m}$
- $num_d = a_n*x^n + a_{n-1}*x^{n-1} + \ldots + a_0x^0 + a_{-1}*x^{-1} + a_{-2}*x^{-2} + \ldots + a_{-m}x^{-m}$
  - Valor de x: base utilizada
  - Cálculo: soma de multiplicações
- Converter para a base 10:
  - 1011_b =
  - 230.4_o =

Conversão base x - base 10

- num_d = parte-inteira$_d$. parte-fracionária$_d$
  = $a_n a_{n-1} \ldots a_0.a_{-1}a_{-2} \ldots a_{-m}$
- $num_d = a_n*x^n + a_{n-1}*x^{n-1} + \ldots + a_0x^0 + a_{-1}*x^{-1} + a_{-2}*x^{-2} + \ldots + a_{-m}x^{-m}$
  - Valor de x: base utilizada
  - Cálculo: soma de multiplicações
- Converter para a base 10:
  - $1011_b = a_3*x^3 + a_2*x^2 + a_1*x^1 + a_0*x^0$
  - $230.4_o = a_2*x^2 + a_1*x^1 + a_0*x^0 + a_{-1}*x^{-1}$

Conversão base x - base 10

- num_d = parte-inteira$_d$. parte-fracionária$_d$
  = $a_n a_{n-1} \ldots a_0.a_{-1}a_{-2} \ldots a_{-m}$
- $num_d = a_n*x^n + a_{n-1}*x^{n-1} + \ldots + a_0x^0 + a_{-1}*x^{-1} + a_{-2}*x^{-2} + \ldots + a_{-m}x^{-m}$
  - Valor de x: base utilizada
  - Cálculo: soma de multiplicações
- Converter para a base 10:
  - $1011_b = 1*2^3 + 0*2^2 + 1*2^1 + 1*2^0$
  - $230.4_o = 2*8^2 + 3*8^1 + 0*8^0 + 4*8^{-1}$

Conversão base x - base 10

- num_d = parte-inteira$_d$. parte-fracionária$_d$
  = $a_n a_{n-1} \ldots a_0 . a_{-1} a_{-2} \ldots a_{-m}$
- $num_d = a_n * x^n + a_{n-1} * x^{n-1} + \ldots + a_0 x^0 + a_{-1} * x^{-1} + a_{-2} * x^{-2} + \ldots + a_{-m} x^{-m}$
  - Valor de x: base utilizada
  - Cálculo: soma de multiplicações
- Converter para a base 10:
  - $1011_b = 1*8 + 0*4 + 1*2 + 1*1 = 11_d$
  - $230.4_o = 2*64 + 3*8 + 0*1 + 4/8 = 152.5_d$

Exercício

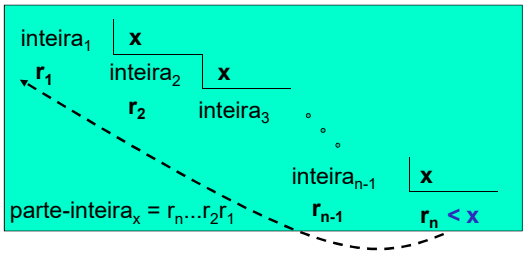
- Converter o número 4A3B.3F$_h$ para a base decimal

Exercício

- Converter o número 4A3B.3F$_h$ para a base decimal
  - 19003.24609375

Conversão base 10 - base x

- Parte inteira

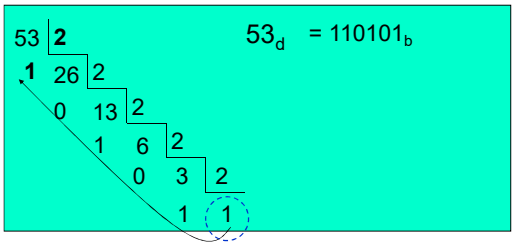


Exemplo

- Converter 53_d para base binária

Exemplo

- Converter 53_d para base binária



Conversão base 10 - base x

- Parte fracionária

$fracionária_1 * x = v_1 . fracionária_2$

$fracionária_2 * x = v_2 . fracionária_3$

$fracionária_{m-1} * x = v_{m-1} . fracionária_m$

$parte\text{-}fracionáriaria_x = 0.v_1v_2...v_{m-1}$

Exemplo

- Converter 0.3_d para base binária

Exemplo

- Converter 0.3_d para base binária

$0.3_d = 0.010011001..._b$

0.3 * 2 = 0.6
0.6 * 2 = 1.2
0.2 * 2 = 0.4
0.4 * 2 = 0.8
0.8 * 2 = 1.6
0.6 * 2 = 1.2
...

Exercício

- Converter o número 34.7_d para a base octal
- Converter 231.3 da base 4 para a base 7

Conversão binário para octal

- Pode ser realizada visualmente
  - Cada três dígitos binários representa um dígito octal
  - Começar no ponto decimal
  - O que faltar, completar com 0s
    - Ex. 01101.1101 = 001 101 . 110 100
      = 15.64

Conversão octal para binário

- Pode ser realizada visualmente
  - Cada dígito octal corresponde a três dígitos binários
  - Começar no ponto decimal
    - Ex. 14.213 = 001 100 . 010 001 011
      = 1100.010001011

Outras conversões

- Conversão hexadecimal para binário
  - Semelhante a conversão octal para binário
    - Utiliza quatro dígitos binários
- Conversão binário para hexadecimal
  - Semelhante a conversão binário para octal
    - Utiliza quatro dígitos binários

Exercício

- Converter o número $4A3B.3F_h$ para a base binária

Exercício

- Converter o número $4A3B.3F_h$ para a base binária
  - 100101000111011.00111111

Exercício

- Converter o número 101101.011010 da base binária para a base octal e para a base hexadecimal
- Converter o número A44E.39 da base hexadecimal para a base octal e para a base binária
- Converter 7456.362 da base octal para a base hexadecimal e para a base binária
- Converter o número 72.43 da base octal para a base decimal
- Converter o número 132.21 da base decimal para a base hexadecimal

Aritmética binária

- Operações aritméticas em sistemas digitais são geralmente feitas em binário
  - Projetar circuitos lógicos para aritmética binária é bem mais fácil do que para aritmética decimal
- Aritmética binária é realizada de forma semelhante à aritmética decimal

Soma binária

- Tabela de adição:
  - 0 + 0 = 0
  - 0 + 1 = 1
  - 1 + 0 = 1
  - 1 + 1 = 0 (e vai 1 para a próxima coluna)
- Exemplo:

$$\begin{array}{rll} & 1101 & (13_d) \\ + & \underline{1011} & (11_d) \\ & 11000 & (24_d) \end{array}$$

Subtração binária

- Tabela de subtração:
  0 - 0 = 0
  0 - 1 = 1 (e empresta 1 da próxima coluna)
  1 - 0 = 1
  1 - 1 = 0
- Exemplo:

$$\begin{array}{rll} & 11000 & (24_{10}) \\ - & 1011 & (11_{10}) \\ \hline & 1101 & (13_{10}) \end{array}$$

Multiplicação binária

- Semelhante à multiplicação de decimais
- Tabela de multiplicação:
  0 x 0 = 0
  0 x 1 = 0
  1 x 0 = 0
  1 x 1 = 1
- Exemplo: 1101 (13_{10}) x 1011 (11_{10})

Multiplicação binária

- Exemplo:

$$\begin{array}{rll} & 1101 & (13_{10}) \\ \times & 1011 & (11_{10}) \\ \hline & 1101 & \\ & 1101 & \\ & 0000 & \\ & 1101 & \\ \hline & 10001111 & (143_{10}) \end{array}$$

Divisão binária

- Semelhante à divisão de decimais
- Exemplo: 10001111 (143_{10})/1011 (11_{10}) = 1101 (13_{10})

Divisão binária I

$$\begin{array}{l|l} 10001111 & 1011 \\ \underline{1011} & \quad 1 \\ 00110\mathbf{1} & \end{array}$$

10001111/1011 = 1101
$143_{10} / 11_{10} = 13_{10}$

Divisão binária I

$$\begin{array}{l|l} 10001111 & 1011 \\ \underline{1011} & \quad 11 \\ 001\mathbf{1101} & \\ \underline{\ 1011} & \\ 00001\mathbf{01} & \end{array}$$

10001111/1011 = 1101
$143_{10} / 11_{10} = 13$

Divisão binária I

```
10001111 | 1011
 1011    | 110
00110 1
  1011
-------
00001 01
-------
00001011
```

10001111/1011 = 1101

$143_{10} / 11_{10} = 13$

Divisão binária I

```
10001111 | 1011
 1011    | 1101
00110 1
  1011
-------
00001 01
-------
00001011
 1011
 0000
```

10001111/1011 = 1101

$143_{10} / 11_{10} = 13$

Divisão binária II

```
10010001 | 1011
```


Divisão binária II

```
145 | 11
```


Divisão binária II

```
10010001 | 1011
 1011    | 1101
00111 0
  1011
-------
00001 10
-------
00001101
 1011
 0010
```

10010001/1011 = 1101
resto = 0010

$145_{10} / 11_{10} = 13$
resto = 2

Exercício

- Calcular o resultado de:
  - 10101101 + 1010
  - 10101101 - 1010
  - 10101101 / 1010
  - 10101101 * 1010
- Fazer em casa (resultado em o e em h):
  - $A35F282_h + 436_o$
  - $A35F282_h - 436_o$
  - $A35F282_h / 436_o$
  - $A35F282_h * 436_o$

Valores binário

- Como representar valores negativos?
  - Geralmente, acrescenta-se um novo bit, no início da sequência de bits
    - Bit de sinal
    - Valor positivo? Bit de sinal = 0
    - Valor negativo? Bit de sinal = 1
  - Exemplo:
    - Número binário formado por três bits para valor e um bit para sinal
    - $+3_d = 0011_b$
    - $-3_d = 1011_b$

Soma de valores

- Se têm sinal diferente
  - Deixando de fora o bit sinal, subtrair o menor do maior
  - Por no resultado sinal do maior dos operandos
- Se têm o mesmo sinal
  - Deixando de fora o bit sinal, soma os dois valores
  - Por no resultado sinal dos operandos
  - Atenção para estouro de valor

Valores binário

- Alternativas
  - Notação complemento a 1 e notação complemento a 2
    - Número positivo
      - Representado como ele é
    - Número negativo
      - Representado pelo complemento de um número positivo

Complemento a 1

- Inverter todos os bits do número utilizando seu valor absoluto
- Ex.: C1 (00000001) = 11111110
- Para um sistema com 8 bits
  - Maior número positivo: 0 1111111 +127
  - Maior número negativo: 1 0000000 -127
  - Dois valores para zero: 0 0000000 e 1 1111111

Complemento a 1

- Cálculo do valor negativo em decimal
  - Fórmula: $-x = 2^n - x - 1$
  - Exemplo para n = 8 (8 bits)

$12_d = 00001100_b$

$-12_d = 2^8 - 12 - 1$

$= 256 - 12 - 1$

$= 243$

$= 11110011$

Complemento a 2

- Inverter todos os bits do número utilizando seu valor absoluto e adiciona o valor 1
- Ex.: C2 (00000001) = 11111110 + 1
  = 11111111
- Para um sistema com 8 bits
  - Maior número positivo: 0 1111111 +127
  - Maior número negativo: 1 0000000 -128
  - Zero: 0 0000000
- Facilita operação de soma

Complemento a 2

- Cálculo do valor negativo em decimal
  - Fórmula: $-x = 2^n - x$
  - Exemplo

$12_d = 00001100_b$

$-12_d = 2^8 - 12$

$= 256 - 12$

$= 244$

$= 11110100$

Vantagens de complemento a 2

- Um único zero
- Mais fácil manipular em hardware
- Facilita a soma de números quando existem valores negativos
  - Basta somar os valores
  - No complemento a 1, deve ser somado o valor 1 depois

Exercício

- Calcular 7 - 3, usando 4 bits
  - Complemento a 1
  - Complemento a 2
  - Com bit de sinal (entre os 4 bits)

Valores binário

- Comparação para 4 bits

Valor	Bit sinal	Compl. 1	Compl. 2
+7	0111	0111	0111
+6	0110	0110	0110
+5	0101	0101	0101
+4	0100	0100	0100
+3	0011	0011	0011
+2	0010	0010	0010
+1	0001	0001	0001
+0	0000	0000	0000

Valores binário

- Comparação para 4 bits

Valor	Bit sinal	Compl. 1	Compl. 2
-0	1000	1111	-
-1	1001	1110	1111
-2	1010	1101	1110
-3	1011	1100	1101
-4	1100	1011	1100
-5	1101	1010	1011
-6	1110	1001	1010
-7	1111	1000	1001
-8	-	-	1000

Código binário BCD

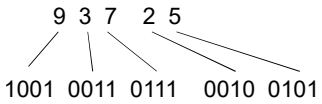
- Binary-coded-decimal ou 8-4-2-1
- Cada dígito decimal é substituído pelo seu equivalente em binário
  - Codifica cada valor decimal por 4 bits
  - Não utiliza todas as combinações de valores binários
    - Ex. 1111

Código BCD

- Vantagens
  - Permite conversão rápida
    - De decimal para binário
    - De binário para binário
  - Facilita impressão ou apresentação no monitor
- Desvantagens
  - Desperdício de espaço
  - Implementação em hardware é mais complexa
  - Não segue as regras da aritmética

Código BCD

- Ex.: codificar 937.25 para binário



8-4-2-1 BCD

Conclusão

- Sistemas numéricos
- Conversões entre sistemas numéricos
- Aritmética com números binários
  - Soma
  - Subtração
  - Multiplicação
  - Divisão
- Outros códigos binários

Perguntas

Códigos binários para caracteres

- Códigos ASCII, EBCDIC e Unicode
  - Transformam caracteres em valores binários

Código ASCII

Mais significativos

	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