Circuitos Digitais

Sistemas de Numeração

Objetivos de Aprendizagem

Reconhecer as diversas bases numéricas utilizadas no dia a dia

Converter números inteiros e fracionários de diversas bases

Conhecer uma forma de representação de números em ponto flutuante e alguns códigos numéricos

Sistemas Numéricos

Utilizamos diversos sistemas numéricos no dia a dia:

- Sistema Decimal (base 10): mais comum, usado em quase todas as aplicações
- Sistema Duodecimal (base 12): aparece na medida de tempo (2 períodos de 12 horas), tamanho (12 polegadas em 1 pé), medidas como dúzia.
- Base Sexagesimal (base 60): aparece em medidas de tempo (minutos em uma hora, segundos em um minuto).
- Base Binária (base 2): representação de informação em sistemas digitais

Valor-Posição

O valor de um dígito específico depende dele próprio e da posição que ele ocupa

Decimal - $\{0, 1, 2, \dots, 9\}$:

$$(123.8)_{10} = 1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0 + 8 \times 10^{-1}$$

Duodecimal - $\{0, 1, 2, \dots, 9, A, B\}$:

$$(4A.6)_{12} = 4 \times 12^1 + 10 \times 12^0 + 6 \times 12^{-1} = (58.5)_{10}$$

Binário - $\{0, 1\}$:

$$(1011.01)_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2}$$
$$= (11.25)_{10}$$

Sistemas Octal e Hexadecimal

Sistemas numéricos com base potência de 2 são facilmente mapeados na base binária, e vice-versa (facilitam a representação de uma sequência de bits):

Octal - $\{0, 1, 2, 3, 4, 5, 6, 7\}$:

$$(271)_8 = (010\ 111\ 001)_2$$

Hexadecimal - $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F\}$:

$$(A0E)_{16} = (1010\ 0000\ 1110)_2$$

Números Binários

Com n bits, o maior número representável é $2^n - 1$

Bit mais à direita é chamado de **LSB** (least significant bit)

Bit mais à esquerda é chamado de **MSB** (most significant bit)

Sequência de 8 bits – **Byte**

Sequência de 4 bits – **Nibble**

Decimal	Binário
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Conversão Binário-Digital

Um número binário de 8 bits inteiros e 5 bits fracionários possui os seguintes pesos por posição:

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}
128	64	32	16	8	4	2	1	0.5	0.25	0.125	0.0625	0.03125

A conversão de binário para decimal pode ser feita simplesmente somando-se os pesos das posições onde o número binário é '1'.

Exemplo

Converta 1101101.1011_2 para decimal

64	32	16	8	4	2	1	0.5	0.25	0.125	0.0625
1	1	0	1	1	0	1	1	0	1	1

$$64 + 32 + 8 + 4 + 1 + 0.5 + 0.125 + 0.0625 = 109.6875$$

Conversão Decimal-Binário (Método de Inspeção)

Decompõe-se o número em soma de potências de 2

Número Decimal	Decomposição	Número Binário
9	$8+1=2^3+2^0$	1001
12	$8+4=2^3+2^2$	1100
25	$16+8+1=2^4+2^3+2^0$	11001
58	$32+16+8+2=2^5+2^4+2^3+2^1$	111010
82	$64+16+2=2^6+2^4+2^1$	1010010

Conversão Decimal-Binário (Método das divisões sucessivas)

Divide-se o número **por 2** até o quociente se tornar 0. Os **restos** das divisões formam o **número binário**, do LSB até o MSB

Dividendo	Quociente	Resto
12	6	0
6	3	0
3	1	1
1	0	1

$12_{10} = 1100_2$

Dividendo	Quociente	Resto
45	22	1
22	11	0
11	5	1
5	2	1
2	1	0
1	0	1

$45_{10} = 101101_2$

Conversão Decimal-Binário Fracionário (Método das Multiplicações sucessivas)

Multiplica-se a parte fracionária do número **por 2**, recuperando-se o **carry**. O primeiro bit produzido é o mais significativo da parte fracionária

0.3125×2	$\mathbf{0}.625$
0.625×2	$\mathbf{1}.25$
0.25×2	$\mathbf{0}.5$
0.5×2	$\mathbf{1}.0$

$$0.3125_{10} = 0.0101_2$$

0.2×2	$\mathbf{0}.4$
0.4×2	$\mathbf{0}.8$
0.8×2	$\mathbf{1}.6$
0.6×2	$\mathbf{1}.2$
0.2×2	$\mathbf{0}.4$
$\vdots$	$\vdots$

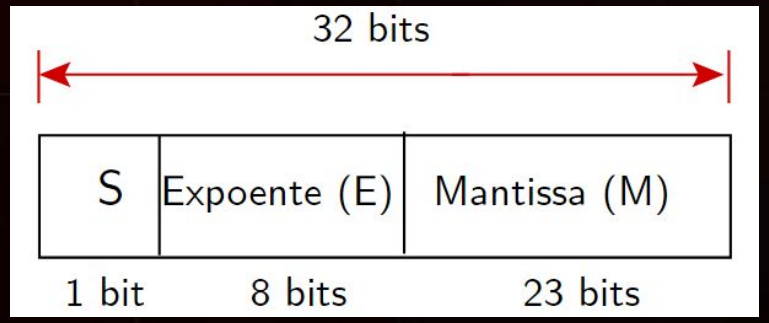
$$0.2_{10} = (0.00110011\ldots)_2$$

Representação de números em ponto flutuante

Números muito grandes ou muito pequenos exigem grande quantidade de bits. Representação em ponto flutuante é mais adequada

Padrão IEEE-754:

- Precisão simples (32 bits)
- Precisão dupla (64 bits)
- Precisão estendida (80 bits)



$$\#: (-1)^S \times (1 + M) \times 2^{E-127}$$

S é o bit de sinal

E o expoente polarizado, obtido somando-se 127 ao expoente real

M é a mantissa e representa a parte fracionária do número

Exemplo

Represente o número positivo 1001110011001 no padrão IEEE-754 de precisão simples

Observe que:

$$100111011001 = 1.00111011001 \times 2^{11}$$

Logo:

$$S = 0$$

$$E = 11 + 127 = 138 = (10001010)_2$$

$$M = 00111011001000000000000$$

Código BCD

O código BCD 8421 implica que cada dígito decimal é decomposto em 4 bits, com pesos $2^3\ 2^2\ 2^1\ 2^0$.

Decimal	0	1	2	3	4	5	6	7	8	9
BCD 8421	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

- As outras combinações $(1010 - 1111)$ são códigos inválidos.
- A conversão é feita dígito a dígito

$$35_{10} = 0011\ 0101_{BCD}$$

$$0.2_{10} = 0.\overline{0011}_2 = 0.0010_{BCD}$$

Código de Gray

O código de Gray não é aritmético nem posicional. Apenas 1 bit é alterado em palavras sequenciais

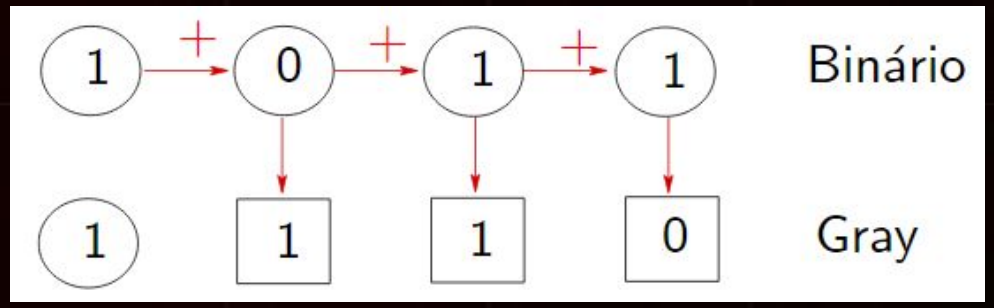
4 bits:

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

Conversão Binário-Gray

Conversão de binário para código Gray

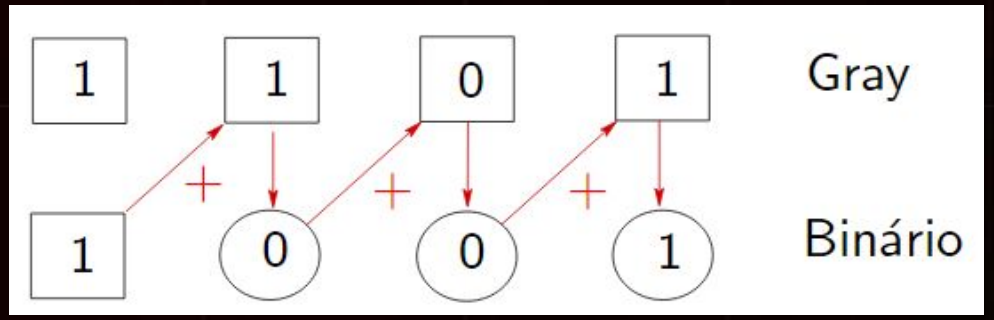
1. Copia-se o MSB
2. Da esquerda p/ direita, adiciona-se o par de bits da codificação binária, descartando-se o carry



Conversão Gray-Binário

Conversão de código Gray para binário

1. **Copia-se o MSB**
2. **Da esquerda p/ direita, adiciona-se o bit gerado com o próximo bit da codificação Gray. Descarta-se o carry.**



Referências

TOCCI, R. T.; WIDMER, N. S; MOSS, G. L. | Sistemas Digitais: Princípios e Aplicações, ed. 12