Aula 02 - Representações Avançadas em Binário

Circuitos Digitais - CRT 0384

Prof. Rennan Dantas
Ciência da Computação

2020.1

Na Aula Anterior...

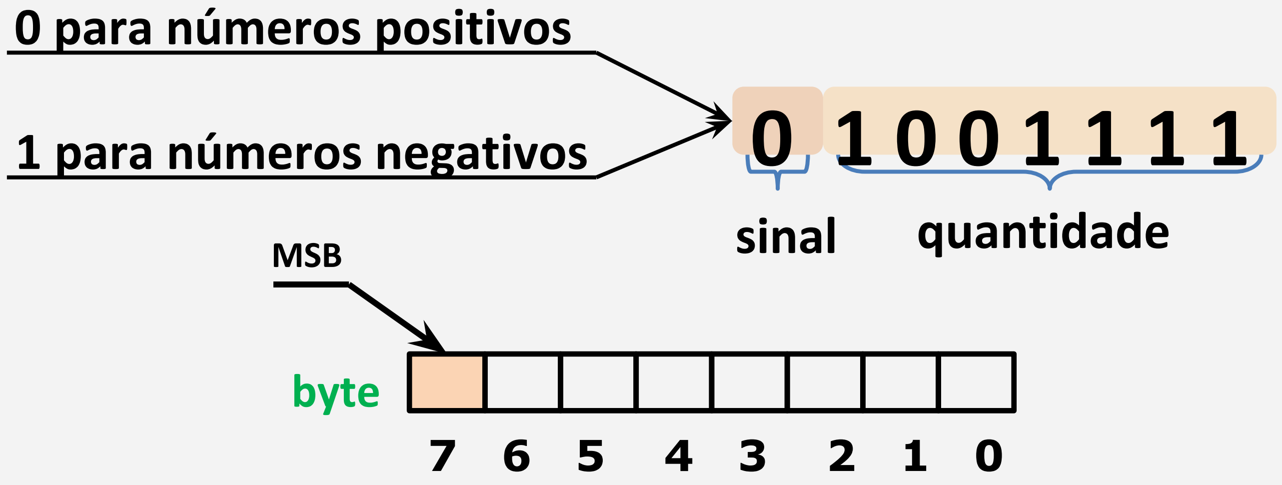
- Fundamentação dos sistemas Numéricos Posicionais
- Sistema Numéricos
  - Decimal
  - Binário
  - Octal
  - Hexadecimal
- Conversão de bases

Nesta Aula

- Representação de números negativos em binário;
- Representação de números reais em base binária;
- Conversão de bases de números reais;
- Complementos de 1 e 2;
- Extensão do sinal em complemento de 2;
- Notação de ponto flutuante;
- Motivação para Códigos Binários;
- Código BCD;
- Código Johnson;
- Código Excesso de 3;
- Código Gray;
- Código ASCII.

Números Inteiros Sinalizados

- Utiliza-se um tamanho fixo de palavra;
- Geralmente o bit mais significativo é reservado para o sinal do número;



Exemplos

1	0 0 0 0 0 0 1	⇨	-1_{10}
1	0 0 0 1 0 1 0	⇨	-10_{10}
0	0 1 0 1 0 1 0	⇨	$+42_{10}$
1	0 1 0 1 0 1 0	⇨	-42_{10}

Representações Alternativas para Números Inteiros Sinalizados

- Os números de magnitude com sinal são fáceis de entender, mas eles requerem demasiado hardware para adição e subtração. Isso tem levado ao uso amplo de complementos para aritmética binária.
- Existem dois tipos de complemento:
  - Complemento de 1
  - Complemento de 2

Complemento de 1

- O complemento de 1 é calculado pela inversão de cada um dos bit do número;
- Existe duas possíveis representações par o número 0.

0 0 1 0 ⇨ $+2_{10}$

1 1 0 1 ⇨ -2_{10}

Decimal	Comp. 1
7	0111
6	0110
5	0101
4	0100
3	0011
2	0010
1	0001
0	0000
-1	1110
-2	1101
-3	1100
-4	1011
-5	1010
-6	1001
-7	1000
-0	1111

Complemento de 2

- O complemento de 2 é calculado pela inversão de cada um dos bits do número. Subsequentemente soma-se 1 ao valor dos bits invertidos;

$$
\begin{array}{rl}
 & 0\,0\,1\,0 \rightarrow +2_{10} \\
 & 1\,1\,0\,1 \\
+ & 0\,0\,0\,1 \\
\hline
 & 1\,1\,1\,0 \rightarrow -2_{10}
\end{array}
$$

Decimal	Comp. 2
7	0111
6	0110
5	0101
4	0100
3	0011
2	0010
1	0001
0	0000
-1	1111
-2	1110
-3	1101
-4	1100
-5	1011
-6	1010
-7	1001
-8	1000

Extensão de Sinal Positivo

- Considere por exemplo a representação do número 12 em complemento de 2

$$0\ 1\,1\,0\,0 \Rightarrow 12_{10}$$

- No computador, por conveniência de arquitetura, o tamanho da palavra binária (número de bits) é sempre múltiplo de 2 (4, 8, 16, 32, ...)

- Para acomodar um número de 5 bits em uma palavra de 8 bits, basta estender o sinal para os demais bits

$$0\ 0\,0\,0\,1\,1\,0\,0 \Rightarrow 12_{10}$$

Extensão de Sinal Negativo

- Considere por exemplo a representação do número -12 em complemento de 2

1 0 1 0 0 → -12_{10}

- Se completarmos os bits restantes para uma palavra de 8 bits com zeros, o número deixará de ser zero
- Em complemento de 2, basta que completemos os demais bits com o bit de sinal

1 1 1 1 0 1 0 0 → 12_{10}

Números Reais em Binário

- Extensão simples do sistema posicional;
- A parte inteira fica inalterada, a parte fracionária utiliza potências negativas.

$$10{,}5_{10} \Rightarrow \begin{array}{c|c|c} 1 & 0 & 5 \\ \hline 10^1 & 10^0 & 10^{-1} \end{array}$$

$$\ldots \underset{2^3}{\_\_} \; \underset{2^2}{\_\_} \; \underset{2^1}{\_\_} \; \underset{2^0}{\_\_} \, , \underset{2^{-1}}{\_\_} \; \underset{2^{-2}}{\_\_} \ldots$$

Pot.	valor
2^{-1}	0,5
2^{-2}	0,25
2^{-3}	0,125
2^{-4}	0,0625
2^{-5}	0,03125
2^{-6}	0,015625
2^{-7}	0,0078125
2^{-8}	0,00390625

Conversão (Reais) Decimal - Binário

$42{,}42_{10} \Rightarrow 42_{10} + 0{,}42_{10}$

$$
\begin{array}{r}
0{,}42 \\
\times\ \ 2 \\
\hline
\boxed{0}{,}84 \\
\times\ \ 2 \\
\hline
\boxed{1}{,}68 \\
\times\ \ 2 \\
\hline
\boxed{1}{,}36 \\
\times\ \ 2 \\
\hline
\boxed{0}{,}72 \\
\vdots
\end{array}
$$

$101010{,}0110_2$

Um Exemplo Mais Simples

$10{,}25_{10} \Rightarrow 10_{10} + 0{,}25_{10}$

$$
\begin{array}{r}
0{,}25 \\
\times \quad 2 \\
\hline
0{,}50 \\
\times \quad 2 \\
\hline
1{,}00
\end{array}
$$

$1010{,}01_2$

condição de parada

Conversão binário →decimal

$$1010{,}01_2$$

$$0\text{x}2^{-1} + 1\text{x}2^{-2}$$

Notação em Ponto Flutuante

- Fundamentada na notação numérica científica;

$$42{,}42 = 42{,}42x10^{0} = 4{,}242x10^{1} = 0{,}4242x10^{2}$$

- Utilização otimizada do espaço de representação;
- Note que o sinal fracionário “flutua” dependendo do expoente associado a base;

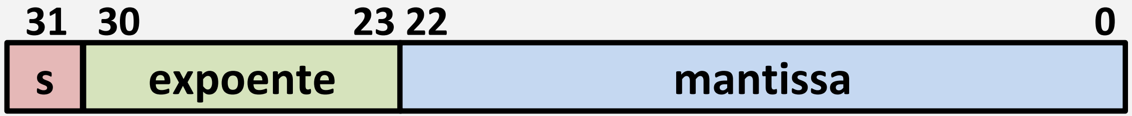
$$+/\_\,0,mantissa \times base^{+/\_\,\mathrm{exp}oente}$$

- A mantissa está contida no intervalo [0,1[
- É importante notar que a notação em ponto flutuante pode induzir à erros de arredondamento.

Padrões de Representação

IEEE Standard for Floating-Point Arithmetic, IEEE 754’2008

- Precisão Simples



- Precisão Dupla

Conversão (Precisão simples)

- Expoente possui um bias de 127 (01111111_2);
- Ao contrário da notação científica tradicional, que coloca todos os dígitos significativos a direita da vírgula, em ponto flutuante deixamos um '1' a esquerda da vírgula.
- Equação para conversão binário →decimal:

$$n = (-1)^s \times \left(1 + \sum_{i=1}^{23} b_{23-i} \times 2^i\right) \times 2^{e-127}$$

Exemplo

- $10{,}25_{10}$ ⇨ $1010{,}01_2$ ⇨ $1{,}01001 \times 2^3$
  - sinal → +
  - expoente → 127+3 = 130 → (01111111+11) = 10000010
  - mantissa → 01001000000000000000000

31	30 … 23	22 … 0
0	10000010	01001000000000000000000

Casos Especiais

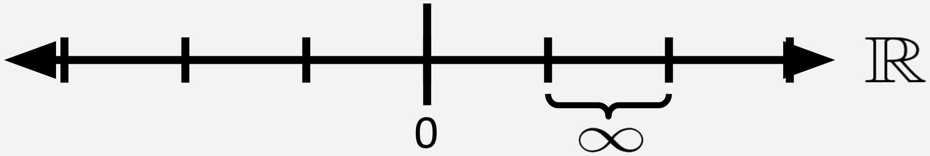
- Números (não normalizados)

	31	30 ... 23	22 ... 0
zero	x	00000000	00000000000000000000000
infinito	x	11111111	00000000000000000000000
NaN	x	11111111	01000000010000000001000

Pelo menos 1 bit
da mantissa
diferente de zero

Números Representáveis

- Em matemática, o conjunto dos números reais é infinito;
- Entre dois números reais quaisquer, há infinitos números reais;
- Para tal, infinitos dígitos devem ser potencialmente utilizados;
- A representação de números reais utilizando a notação de ponto flutuante, utiliza um número finito de bits;
- Por definição, apenas números racionais podem ser representados em ponto flutuante;



Números Representáveis

- $0.1_{10} \rightarrow 0.0001100110011 \ldots.$

$$Fra = \frac{1}{2^4} + \frac{1}{2^5} + \frac{1}{2^8} + \frac{1}{2^9} + \frac{1}{2^{12}} + \frac{1}{2^{13}} \cdots \rightarrow 0.1$$

- s = 0 | m = 1.1001100110011 … e = -4

31	30 23	22 0
0	**01111100**	**10011001100110011001100**

- Convertendo de volta para decimal …
- m = 0,100000001490116119384765625
- erro = 0,000000001490116119384765625

Exercícios

Converta para representação em ponto flutuante (precisão simples)

- $42{,}42_{10}$
- $0{,}11100110_2 x 2^2$
- $0{,}11100111_2 x 2^2$
- $3{,}6_{10}$

Códigos Binários

- O computador trabalha apenas com números;
- Estes números são sempre em binário, devido a aspectos de construção;
- Códigos binários fornecem uma forma de representar outros conceitos que não números, de maneira a serem mapeados diretamente para suas representações em binário, e desta forma, passiveis de serem processados pelo computador.

BCD 8421

- BCD significa “Binary Coded Decimal”, ou seja,
- Representa números de 0-9 em binário;
- Utiliza quatro bits para cada dígito decimal;
- Para representar o número 10 por exemplo, são necessários oito bits em BCD 8421;
- 8421 referem-se as potências de cada uma das quatro casas do sistema de codificação.

BCD 8421

Decimal	Binário Puro	BCD 8421
0	0000	0000
1	0001	0001
2	0010	0010
3	0011	0011
4	0100	0100
5	0101	0101
6	0110	0110
7	0111	0111

Decimal	Binário Puro	BCD 8421
8	1000	1000
9	1001	1001
10	1010	0001 0000
11	1011	0001 0001
12	1100	0001 0010
13	1101	0001 0011
14	1110	0001 0100
15	1111	0001 0101

Código de Johnson

- Muito utilizado na construção de circuitos contadores;

Dec	Johnson	Binário
0	00000	0000
1	00001	0001
2	00011	0010
3	00111	0011
4	01111	0100
5	11111	0101
6	11110	0110
7	11100	0111
8	11000	1000
9	10000	1001

Código Excesso de 3

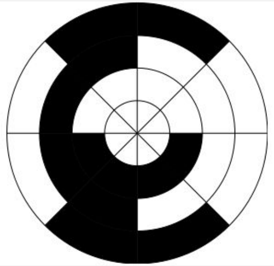
- Código simples, soma-se 11_2 ao número binário puro;

$0\,1\,1\,1_2 \Rightarrow 1\,0\,1\,0_{e3}$

Dec	Exc 3	Binário
0	0011	0000
1	0100	0001
2	0101	0010
3	0110	0011
4	0111	0100
5	1000	0101
6	1001	0110
7	1010	0111
8	1011	1000
9	1100	1001

Código Gray

- Sistema de numeração binário no qual dois valores sucessivos diferem em apenas 1 bit;
- Aplicado em correção de erros, controle de dispositivos eletromecânicos, etc.



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

Tabela ASCII

000		(nul)	016	►	(dle)	032	sp	048	0	064	@	080	P	096	`	112	p
001	☺	(soh)	017	◄	(dc1)	033	!	049	1	065	A	081	Q	097	a	113	q
002	☻	(stx)	018	↕	(dc2)	034	"	050	2	066	B	082	R	098	b	114	r
003	♥	(etx)	019	‼	(dc3)	035	#	051	3	067	C	083	S	099	c	115	s
004	♦	(eot)	020	¶	(dc4)	036	$	052	4	068	D	084	T	100	d	116	t
005	♣	(enq)	021	§	(nak)	037	%	053	5	069	E	085	U	101	e	117	u
006	♠	(ack)	022	▬	(syn)	038	&	054	6	070	F	086	V	102	f	118	v
007	•	(bel)	023	↨	(etb)	039	'	055	7	071	G	087	W	103	g	119	w
008	◘	(bs)	024	↑	(can)	040	(	056	8	072	H	088	X	104	h	120	x
009		(tab)	025	↓	(em)	041	)	057	9	073	I	089	Y	105	i	121	y
010		(lf)	026		(eof)	042	*	058	:	074	J	090	Z	106	j	122	z
011	♂	(vt)	027	←	(esc)	043	+	059	;	075	K	091	[	107	k	123	{
012	♀	(np)	028	∟	(fs)	044	,	060	<	076	L	092	\	108	l	124	|
013		(cr)	029	↔	(gs)	045	-	061	=	077	M	093	]	109	m	125	}
014	♫	(so)	030	▲	(rs)	046	.	062	>	078	N	094	^	110	n	126	~
015	☼	(si)	031	▼	(us)	047	/	063	?	079	O	095	_	111	o	127	⌂

Tabela ASCII

128	Ç	143	Å	158	₧	172	¼	186	║	200	╚	214	╓	228	Σ	242	≥
129	ü	144	É	159	ƒ	173	¡	187	╗	201	╔	215	╫	229	σ	243	≤
130	é	145	æ	160	á	174	«	188	╝	202	╩	216	╪	230	µ	244	⌠
131	â	146	Æ	161	í	175	»	189	╜	203	╦	217	┘	231	τ	245	⌡
132	ä	147	ô	162	ó	176	░	190	╛	204	╠	218	┌	232	Φ	246	÷
133	à	148	ö	163	ú	177	▒	191	┐	205	═	219	█	233	Θ	247	≈
134	å	149	ò	164	ñ	178	▓	192	└	206	╬	220	▄	234	Ω	248	°
135	ç	150	û	165	Ñ	179	│	193	┴	207	╧	221	▌	235	δ	249	∙
136	ê	151	ù	166	ª	180	┤	194	┬	208	╨	222	▐	236	∞	250	·
137	ë	152	ÿ	167	º	181	╡	195	├	209	╤	223	▀	237	φ	251	√
138	è	153	Ö	168	¿	182	╢	196	─	210	╥	224	α	238	ε	252	^{n}
139	ï	154	Ü	169	⌐	183	╖	197	┼	211	╙	225	ß	239	∩	253	^{2}
140	î	155	¢	170	¬	184	╕	198	╞	212	╘	226	Γ	240	≡	254	■
141	ì	156	£	171	½	185	╣	199	╟	213	╒	227	π	241	±	255	
142	Ä	157	¥														

Leitura obrigatória

- Leitura: (Tocci) 6.2 (pgs. 254-259)
- Leitura: (Tocci) 2.4-2.8 (pgs. 31-38)
- Exercícios: (Tocci): E={2.19 – 2.26 }

Aula 02 - Representações Avançadas em Binário

Circuitos Digitais - CRT 0384
Prof. Rennan Dantas
Ciência da Computação

2020.1