Universidade Tecnológica Federal do Paraná – UTFPR
Bacharelado em Ciência da Computação

BCC32B – Elementos de Lógica Digital

Prof. Rodrigo Hübner

Aula 02 – Representação de Dados

Representação de Dados

Números de Ponto Fixo (Inteiros)

Números de Ponto Fixo Sem Sinal: usam representação binária convencional

Exemplo:

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

O valor do número é inteiro.

Nenhum bit é usado para representar sinal.

Representação de Dados

Números de Ponto Fixo (Inteiros)

Números de Ponto Fixo Com Sinal

Existem 4 Métodos de Representação:

1. **Sinal Magnitude**
2. **Complemento de 1**
3. **Complemento de 2**
4. **Notação em Excesso**

Representação de Dados

Números de Ponto Fixo (Inteiros)

Representação Sinal Magnitude:

- Em decimal para representarmos as quantias +12 e -12 ⇒ usamos os sinais **+** e **–** para indicar se o número é positivo ou negativo
- Em Sinal Magnitude: Bit mais significativo (mais à esquerda) indica o sinal do número representado

0 indica número positivo

1 indica número negativo

Os bits restantes representam a Magnitude (valor do dado)

Representação de Dados

Números de Ponto Fixo (Inteiros)

Exemplo na Representação Sinal Magnitude:

$+12_{10} \Rightarrow \boxed{0}0001100_2$

$-12_{10} \Rightarrow \boxed{1}0001100_2$

Só muda o bit de sinal

Os bits restantes representam a Magnitude (valor do dado)

Representação de Dados

Números de Ponto Fixo (Inteiros)

Observações para a Representação Sinal Magnitude:

1. Há 2 representações para o número 0

$+0_{10} \Rightarrow 00000000_2$

$-0_{10} \Rightarrow 10000000_2$

- Pode gerar erros de programação
- Requer hardware mais complexo para comparar com os dois 0s.
- Dificulta testes

Representação de Dados

Números de Ponto Fixo (Inteiros)

Observações para a Representação Sinal Magnitude:

2. Intervalo de representação é menor, isto é, a quantidade de números representáveis é menor

011	+3
010	+2
001	+1
000	+0
100	-0
101	-1
110	-2
111	-3

Exemplo: $2^3 = 8$
Isso significa que com 3 bits poderíamos representar até 8 valores diferentes, mas com duas representações do valor 0 (+0 e -0) podemos representar até 7 valores diferentes

Representação de Dados

Números de Ponto Fixo (Inteiros)

Representação em Complemento de 1:

- **Na representação em Complemento de 1, nós complementamos (invertemos) todos os bits 1 por 0 e os bits 0 por 1**
- **Exemplo:**

$+12_{10} \Rightarrow 00001100_2$

$-12_{10} \Rightarrow 11110011_2$

Os números positivos também têm o bit mais significativo em 0 e os números negativos em 1

Representação de Dados

Números de Ponto Fixo (Inteiros)

Observações para a Representação Complemento de 1:

1. Há também 2 representações para o número 0

$+0_{10} \Rightarrow 00000000_2$

$-0_{10} \Rightarrow 11111111_2$

- Pode gerar erros de programação
- Requer hardware mais complexo para comparar com os dois 0s.

Dificulta testes → `A != 00000000 E A != 11111111`

Representação de Dados

Números de Ponto Fixo (Inteiros)

Observações para a Representação Complemento de 1:

2. Intervalo de representação é menor, isto é, a quantidade de números representáveis é menor

011	+3
010	+2
001	+1
000	+0
111	-0
110	-1
101	-2
100	-3

Exemplo: $2^3 = 8$
Isso significa que com 3 bits poderíamos representar até 8 valores diferentes, mas com 2 representações do valor 0 (+0 e -0) podemos representar até 7 valores diferentes

Representação de Dados

Números de Ponto Fixo (Inteiros)

Representação em Complemento de 2:

- Na representação em Complemento de 2, nós complementamos (invertemos) todos os bits 1 por 0 e os bits 0 por 1 e somamos 1 ao resultado do Complemento de 1
- Exemplo:

$+12_{10} \Rightarrow 00001100_2$

Em Complemento de 2 os números positivos também têm o bit mais significativo em 0 e os números negativos em 1

$$
\begin{array}{lll}
-12_{10} \Rightarrow C1 = & 11110011_2 \\
 & \quad\quad\quad +1 \\
\hline
-12_{10} \quad = & \underbrace{11110100_2}_{-12_{10} \text{ em Complemento de 2}}
\end{array}
$$

Representação de Dados

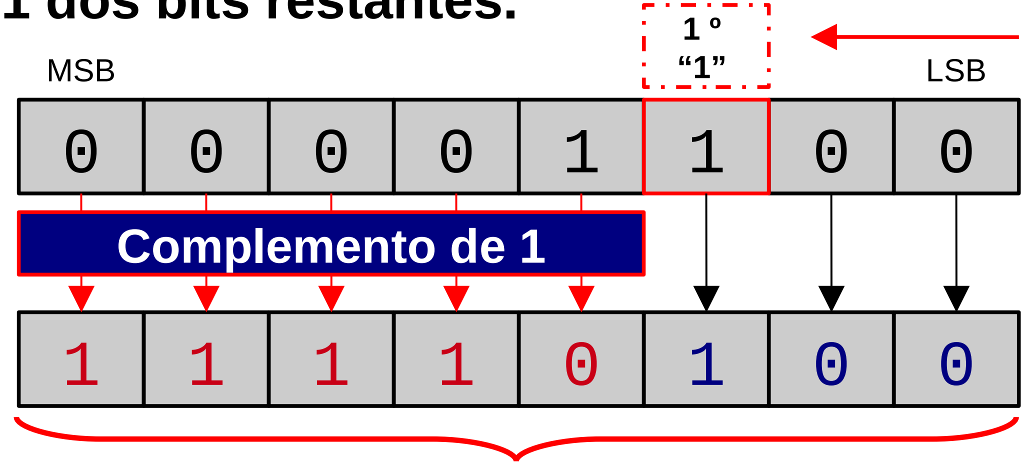
Representação em Complemento de 2

Método Alternativo:

- **Troque todos os bits à esquerda do bit 1 menos significativo.**
  1) Comece à direita com o LSB e escreva os bits como eles aparecem até o primeiro 1 (inclusive ele);
  2) Tome o complemento de 1 dos bits restantes.
- **Exemplo:**

$+12_{10} \Rightarrow 00001100_2$

LSB - *Least significant bit*
MSB - *Most significant bit*



-12_{10} em Complemento de 2

Representação de Dados

Números de Ponto Fixo (Inteiros)

Observações para a Representação Complemento de 2:

1. Há somente 1 representação para o número 0

$$\boxed{+0_{10} \Rightarrow 00000000_2}$$

$$-0_{10} \Rightarrow C1 = \quad 11111111_2$$

$$+1$$

$$-0_{10} = \quad \underbrace{1\,00000000_2}_{-0_{10} \text{ em Complemento de 2}}$$

Carry é ignorado na conversão do número

-0_{10} em Complemento de 2

Representação de Dados

Números de Ponto Fixo (Inteiros)

Observações para a Representação Complemento de 2:

2. Intervalo de representação é maior que dos outros métodos de representação anteriores, porque só há uma representação para o 0

011	+3
010	+2
001	+1
000	+0
000	-0
111	-1
110	-2
101	-3
100	-4

Intervalo maior: 8 representações diferentes

Representação de Dados

Números de Ponto Fixo (Inteiros)

Representação em Excesso (Bias ou Deslocamento):

- **A representação em Excesso, tem o efeito de deslocar o número a ser representado, de forma que, o menor valor (negativo) corresponda à representação com todos os bits em zero e os valores sejam representados em ordem crescente, a partir do menor**
- **Exemplo em Excesso de 128:**

$$+12_{10} \Rightarrow +12+128 = 140 = 10001100_2$$

$$-12_{10} \Rightarrow -12+128 = 116 = 01110100_2$$

Representação de Dados

Números de Ponto Fixo (Inteiros)

Observações para a Representação Excesso:

1. Há somente 1 representação para o número 0
2. Intervalo de representação maior

Com 8 bits pode-se representar 2^8 = 256 números (de 0 a 255)

$+127_{10} \Rightarrow +127 + 128 = 255 = 1111.1111_2$

...

$0_{10} \Rightarrow +0 + 128 = 128 = 1000.0000_2$

...

$-127_{10} \Rightarrow -127 + 128 = 1 = 0000.0001_2$

$-128_{10} \Rightarrow -128 + 128 = 0 = 0000.0000_2$

Ordem crescente facilita comparações entre os números

Representação de Dados

Resumo das Representações de Dados

Decimal	Sem Sinal	Sinal Magnitude	Complemento de 1	Complemento de 2	Excesso de 4
+7	111				
+6	110				
+5	101				
+4	100				
+3	011	001	011	011	111
+2	010	010	010	010	110
+1	001	001	001	001	101
+0	000	000	000	000	100
-0	-	100	111	000	100
-1	-	101	110	111	011
-2	-	110	101	110	010
-3	-	111	100	101	001
-4	-			100	000

Representação de Dados

Números em Ponto Flutuante (Reais)

Problema: Ponto Fixo requer uma quantidade muito grande de dígitos para representar números muito grandes ou muito pequenos

- **Exemplo:** Para representar 1 Trilhão ⇒ Requer 40 bits à esquerda do ponto fixo

- **Exemplo:** Para representar 1 Trilionésimo no mesmo processador ⇒ Requer 40 bits à direita do ponto fixo

- **No total precisamos de 80 bits por número**

Representação de Dados

Números em Ponto Flutuante (Reais)

- **Exemplo: Número de Avogadro $+6{,}023 \times 10^{23}$**
- **Intervalo: 10^{23}**
- **Precisão: $6{,}023$ (3 dígitos de precisão)**
- **Representação do Número em Notação Científica:**

ponto decimal

expoente

sinal $+\underline{6.023} \times \underline{10}^{23}$

Mantissa

raiz (base)

Representação de Dados

Números em Ponto Flutuante (Reais)

- **Obs: Há várias maneiras de representar o mesmo número**
- **Exemplos:**

$$\underbrace{\boxed{3584{,}1 \times 10^{0} = 3{,}5841 \times 10^{3} = 0{,}35841 \times 10^{4}}}_{\text{Várias representações dificultam cálculos e comparações}}$$

↓

Necessidade de normalização da representação

Representação de Dados

Números em Ponto Flutuante (Reais)

Normalização: o ponto é deslocado ("flutua") para a esquerda do dígito diferente de `0` mais à esquerda (bit mais significativo), e o expoente é ajustado

- **Exemplo:**

$$\boxed{3584{,}1 \times 10^{0} = 3{,}5841 \times 10^{3} = 0{,}35841 \times 10^{4}}$$

$$\boxed{0{,}35841 \times 10^{4}} \longleftarrow \text{Forma normalizada}$$

- **Obs.: Para representar "`0`" usa-se a mantissa com todos os valores em `0`**

Representação de Dados

Números em Ponto Flutuante (Reais)

Observações para representação em Ponto Flutuante:

- Para representar 0 usa-se a mantissa com todos os valores em 0
- Em binário, não há necessidade de se armazenar o dígito 1 “mais significativo” da mantissa (já se sabe que ele é 1) ⇒ esse bit é chamado de “bit escondido”. Sobra mais espaço para o número ser representado ⇒ aumenta a precisão

Exemplo

$01101\ ...\ x\ 2^x$

$0,1101...\ x\ 2^y$

Bit escondido

$\downarrow$

$101...\ x\ 2^z$

Representação de Dados

Números em Ponto Flutuante (Reais)

Padrão IEEE 754

- **1980: Padronização da representação em Ponto Flutuante pela IEEE (*Institute of Electrical and Electronics Engineers*)**

- **Padronização:**
  - Facilita a troca de dados entre diferentes computadores
  - Facilita os algoritmos aritméticos de PF, pois tratam os os números sempre no mesmo formato;
  - Melhora a precisão dos números representados devido ao bit escondido

Representação de Dados

Números em Ponto Flutuante (Reais)

Padrão IEEE 754
(ANSI/IEEE 754-1985)

- **São três formas:**
  - **Precisão Simples:** 32 bits {S → 1, E → 8, M → 23}
    - E é representado em excesso de 127 e M efetivamente possui 24 bits, um bit escondido.
  - **Precisão Dupla:** 64 bits {S → 1, E → 11, M → 52}
    - E é representado em excesso de 1023 e M efetivamente possui 53 bits, um bit escondido.
  - **Precisão estendida:** 80 bits {S → 1, E → 15, M → 64}
    - E é representado em excesso de 16383 e M possui 64 bits, não tem bit escondido.

Representação de Dados

Números em Ponto Flutuante (Reais)
Formatos do Padrão IEEE 754

Precisão Simples

1	8	23
S	E	M

bit de sinal

expoente: excesso de 127

mantissa: (bit escondido)

Precisão Dupla

1	11	52
S	E	M

bit de sinal

expoente: excesso de 1023

mantissa: (bit escondido)

Precisão Estendida

1	15	64
S	E	M

bit de sinal

expoente: excesso de 16383

mantissa

Representação de Dados

Números em Ponto Flutuante (Reais)

Padrão IEEE 754

- **Exceções:**
  - O número `0,0` é representado por `0`s em todas as posições.
  - E *infinito* é representado com `1`s em todas as posições do expoente e `0`s em todas as posições da mantissa.

Representação de Dados

Formatos do Padrão IEEE 754

Precisão simples

Exemplo: converter o número decimal para binário

$3,248 \times 10^4 = 32480 = 111111011100000_2 = \boxed{1}\boxed{11111011100000} \times 2^{14}$

O MSB não ocupa a posição de um bit porque ele é sempre 1

Mantissa

11111011100000000000000

Expoente (polarizado → em excesso de 127):

$14 + 127 = 141 = \boxed{10001101_2}$

O número completo representado em ponto flutuante:

S	Expoente							Mantissa																						
0	1	0	0	0	1	1	0	1	1	1	1	1	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Próxima Aula

- **Fundamentos de Lógica**