Arquitetura e Organização de Computadores

Guilherme Henrique de Souza Nakahata

Universidade Estadual do Paraná - Unespar

27 de Agosto de 2024

O sistema decimal

- No dia a dia, usamos um sistema baseado em dígitos decimais (0, 1, 2, 3, 4, 5, 6, 7, 8, 9) para a representação de números, e nos referimos a esse sistema como sistema decimal.

O sistema binário

- No sistema binário, temos somente dois dígitos, 1 e 0.
- Dessa maneira, os números no sistema binário são representados para a base 2.

Notação hexadecimal

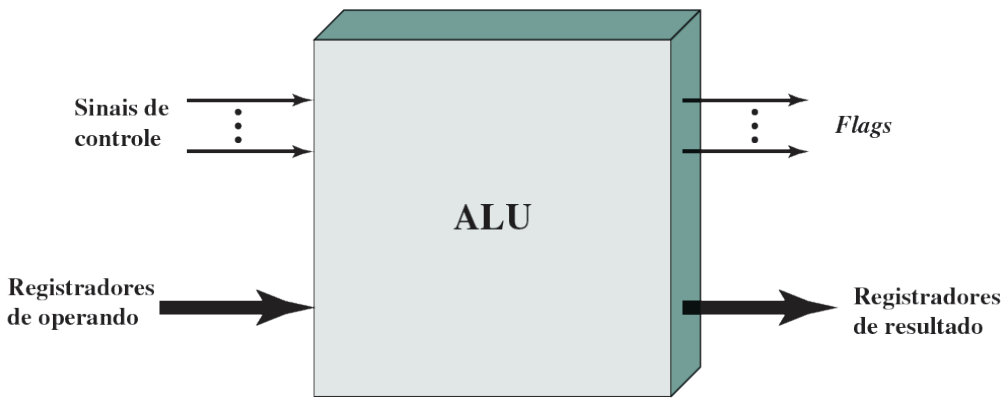
- Os dígitos binários são agrupados em conjuntos de quatro bits, chamados de **nibble**.
- Cada combinação possível de dígitos binários é dada por um símbolo, do seguinte modo:
- Como 16 símbolos são usados, a notação é chamada de hexadecimal, e os 16 símbolos são os dígitos hexadecimais.

Notações

Decimal (base 10)	Binário (base 2)	Hexadecimal (base 16)
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F
16	0001 0000	10
17	0001 0001	11
18	0001 0010	12
31	0001 1111	1F
100	0110 0100	64
255	1111 1111	FF
256	0001 0000 0000	100

Unidade Lógica e Aritmética (ALU)

- A **ALU** é a parte do computador que realmente realiza operações lógicas e aritméticas sobre os dados:



Representação em inteiros

- Para os propósitos de armazenamento e processamento no computador, somente dígitos binários (0 e 1) podem ser usados para representar os números.
- Se estivermos limitados a inteiros não negativos, a representação é direta.

Representação em sinal-magnitude

- A forma de representação mais simples que emprega um bit de sinal é a **representação sinal-magnitude**.
- Em uma palavra de n bits, os n - 1 bits mais à direita representam a magnitude do inteiro.

$$
\begin{aligned}
+18 &= 00010010 \\
-18 &= 10010010 \quad \text{(sinal-magnitude)}
\end{aligned}
$$

Representação em complemento de dois

- Características da representação e aritmética de complemento de dois:

Intervalo	-2^{n-1} a $2^{n-1}-1$
Número de representações de zero	Um
Negação	Apanhe o complemento booleano de cada bit do número positivo correspondente, depois some 1 ao padrão de bits resultante visto como um inteiro sem sinal.
Expansão da extensão em bits	Acrescente posições de bit adicionais à esquerda e preencha com o valor do bit de sinal original.
Regra de *overflow*	Se dois números com o mesmo sinal (positivo ou negativo) são somados, então o *overflow* ocorre se e somente se o resultado tem o sinal oposto.
Regra de subtração	Para subtrair B de A, pegue o complemento de dois de B e some-o a A.

Aritmética com inteiros

- Na representação sinal-magnitude, a regra para formar a **negação** de um inteiro é simples: inverta o bit de sinal.
- Na notação de complemento de dois, a negação de um inteiro pode ser formada com as seguintes regras:
  - Apanhe o complemento booleano de cada bit do inteiro. Ou seja, defina cada 1 como 0, e cada 0 como 1.
  - Trate o resultado como um inteiro binário sem sinal, some 1.
  - Esse processo em duas etapas é conhecido como a operação de complemento de dois.

Aritmética com inteiros

- **Adição** de números na representação em complemento de dois:

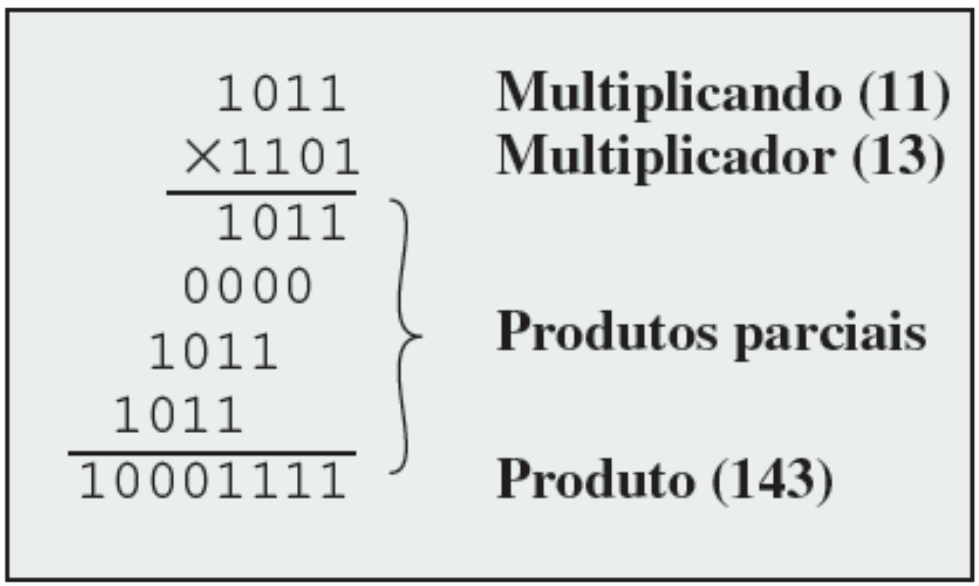
1001 = −7 +0101 = 5 1110 = −2 (a) (−7) + (+5)	1100 = −4 +0100 = 4 10000 = 0 (b) (−4) + (+4)
0011 = 3 +0100 = 4 0111 = 7 (c) (+3) + (+4)	1100 = −4 +1111 = −1 11011 = −5 (d) (−4) + (−1)
0101 = 5 +0100 = 4 1001 = Overflow (e) (+5) + (+4)	1001 = −7 +1010 = −6 10011 = Overflow (f) (−7) + (−6)

Aritmética com inteiros

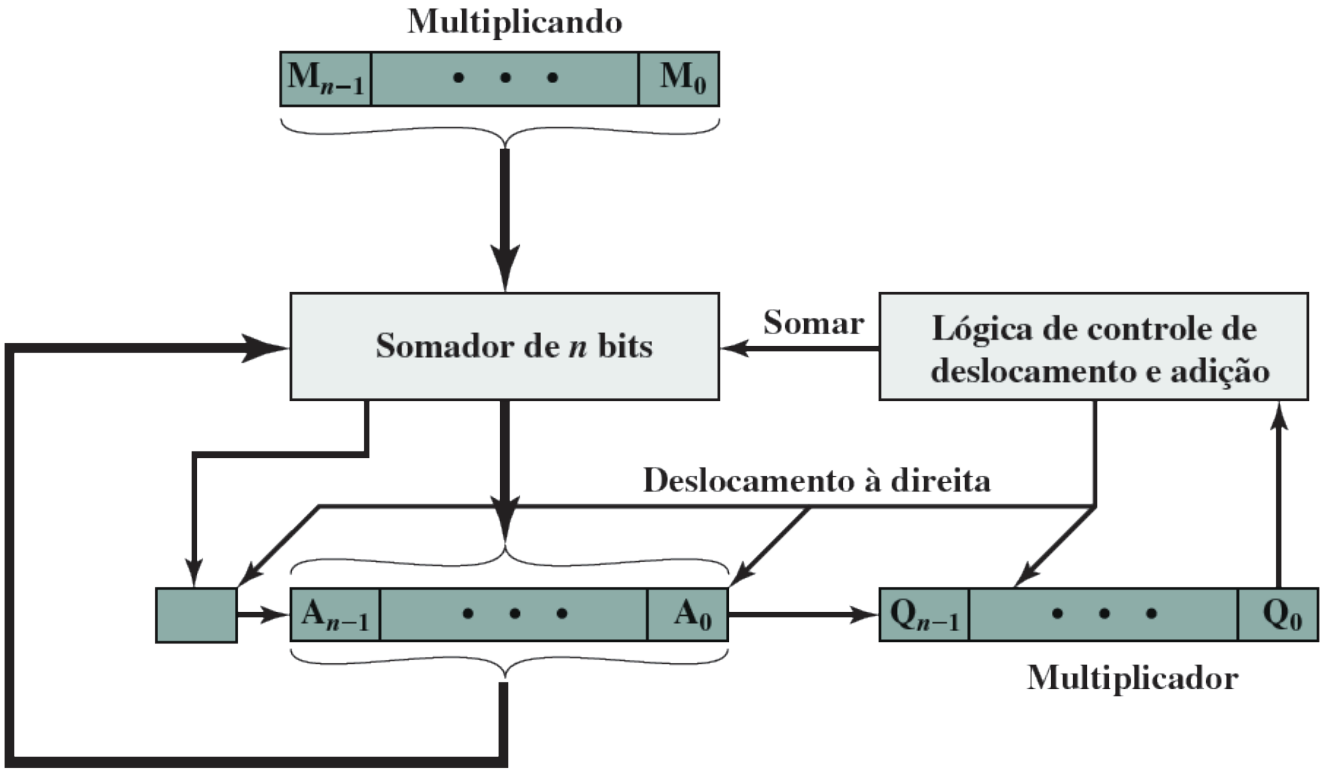
- Em qualquer adição, o resultado pode ser maior do que pode ser mantido no tamanho utilizado da palavra.
- Essa condição é chamada de **overflow**.
- **Regra do overflow**: se dois números são somados e ambos são positivos ou ambos são negativos, então o overflow ocorre se, e somente se, o resultado tiver o sinal oposto.
- **Regra da subtração**: para subtrair um número (subtraendo) de outro (minuendo), pegue o complemento de dois (negação) do subtraendo e some-o ao minuendo.

Aritmética com inteiros

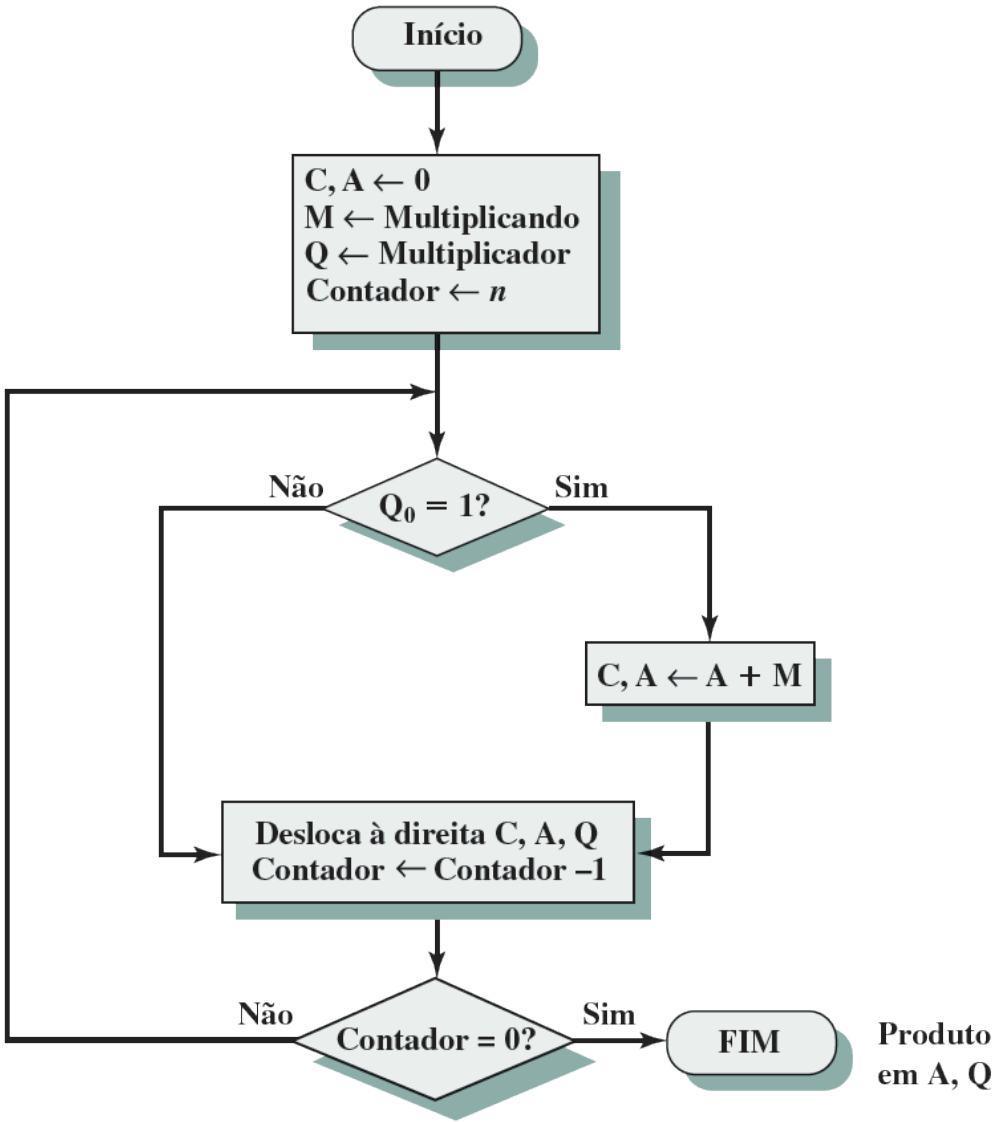
- Em comparação com a adição e a subtração, a **multiplicação** é uma operação complexa, seja ela realizada no hardware ou pelo software.
- Vários algoritmos foram usados em diversos computadores.
- Multiplicação de inteiros binários sem sinal:



- Implementação de hardware da multiplicação binária sem sinal:



- Fluxograma para a multiplicação binária sem sinal:



Aritmética com inteiros

- Multiplicação de dois inteiros de 4 bits sem sinal, gerando um resultado de 8 bits:

```
    1011
  × 1101
----------
00001011   1011 × 1 × 2^0
00000000   1011 × 0 × 2^1
00101100   1011 × 1 × 2^2
01011000   1011 × 1 × 2^3
----------
10001111
```

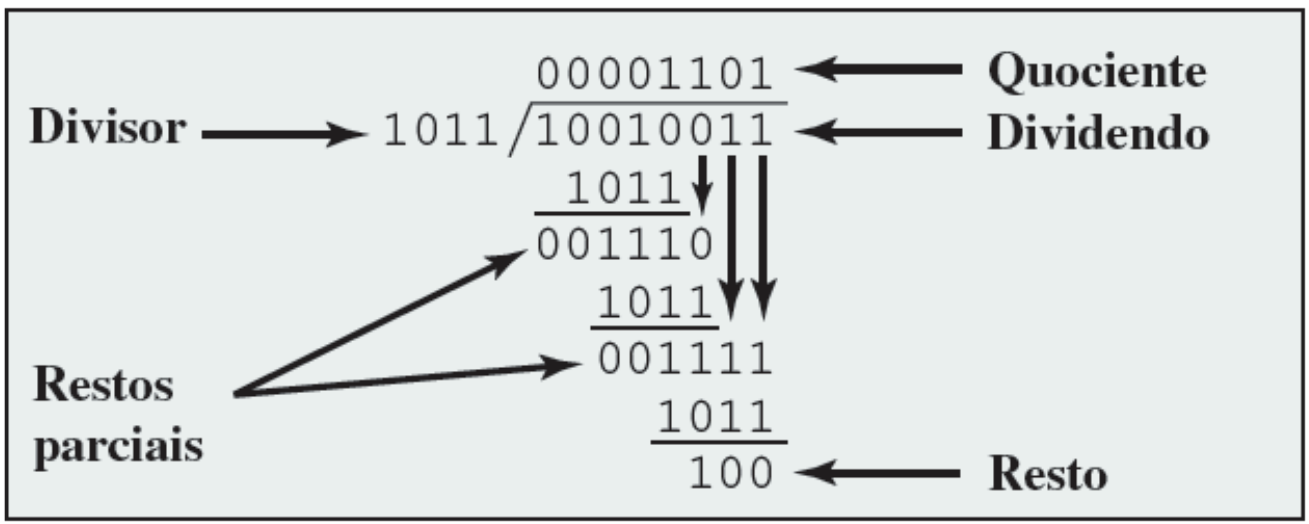
Aritmética com inteiros

- Comparação da multiplicação de inteiros sem sinal e em complemento de dois:

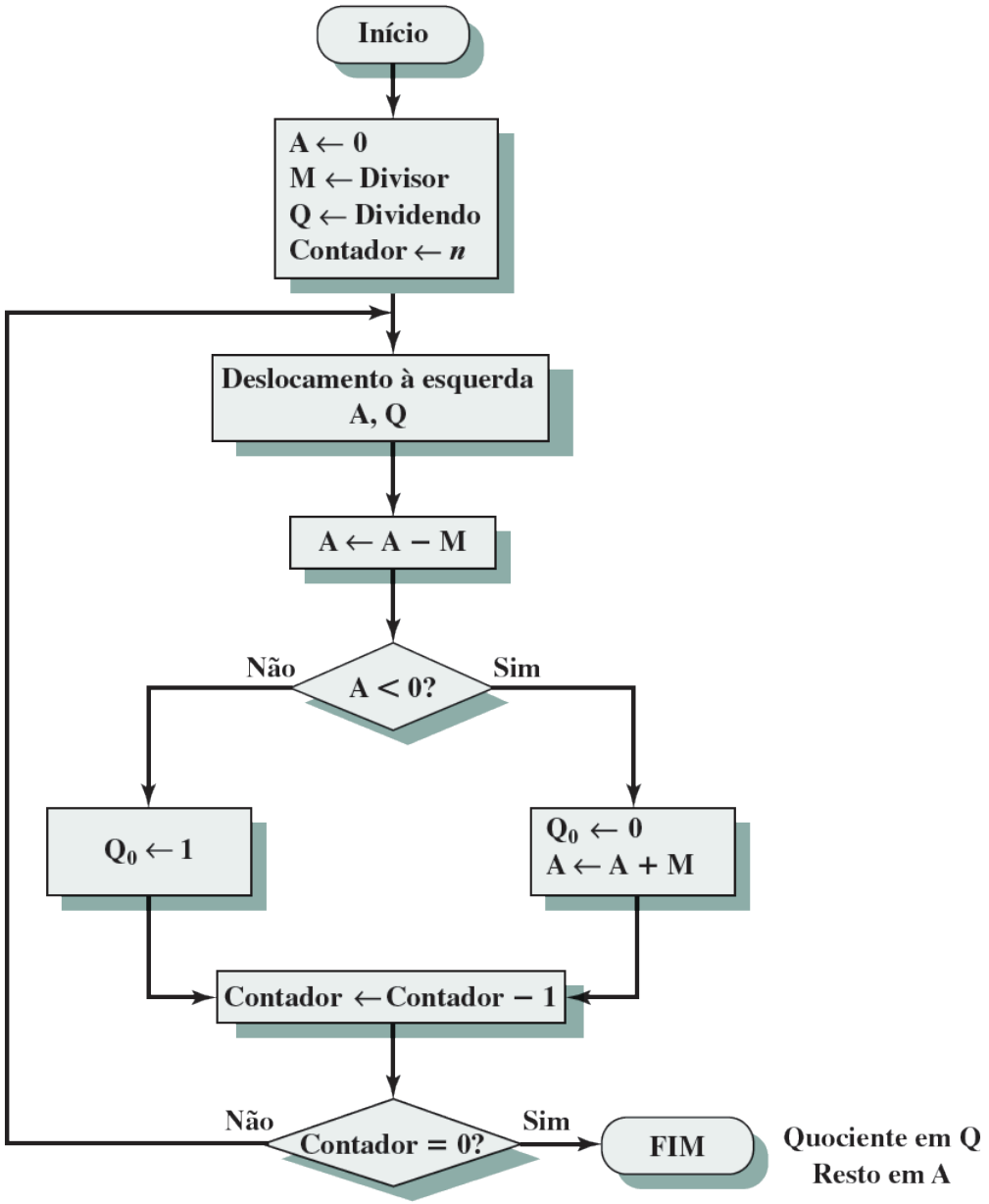
1001 (9)	1001 (−7)
× 0011 (3)	× 0011 (3)
00001001 1001 × 2^0	11111001 (−7) × 2^0 = (−7)
00010010 1001 × 2^1	11110010 (−7) × 2^1 = (−14)
00011011 (27)	11101011 (−21)

Aritmética com inteiros

- A divisão é um pouco mais complexa que a multiplicação, mas é baseada nos mesmos princípios gerais.
- Exemplo de divisão de inteiros binários sem sinal:



Fluxograma para divisão binária sem sinal:

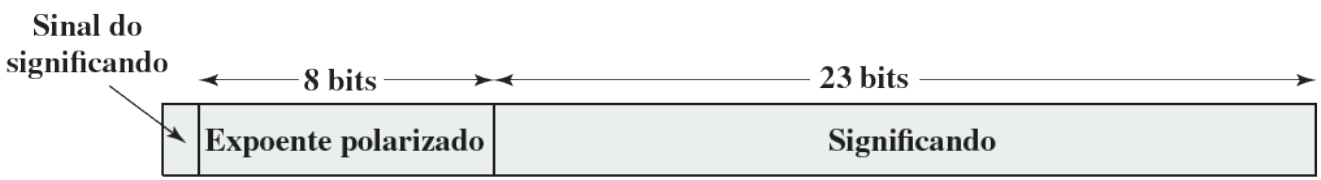


- Exemplo de divisão por restauração em complemento de dois (7/3):

A	Q	
0000	0111	**Valor inicial**
0000	1110	**Deslocamento**
1101		**Use dois complementos de 0011 para a subtração**
1101		**Subtraia**
0000	1110	**Restaure, faça $Q_0 = 0$**
0001	1100	**Deslocamento**
1101		
1110		**Subtraia**
0001	1100	**Restaure, faça $Q_0 = 0$**
0011	1000	**Deslocamento**
1101		
0000	1001	**Restaure, faça $Q_0 = 1$**
0001	0010	**Deslocamento**
1101		
1110		**Subtraia**
0001	0010	**Restaure, faça $Q_0 = 0$**

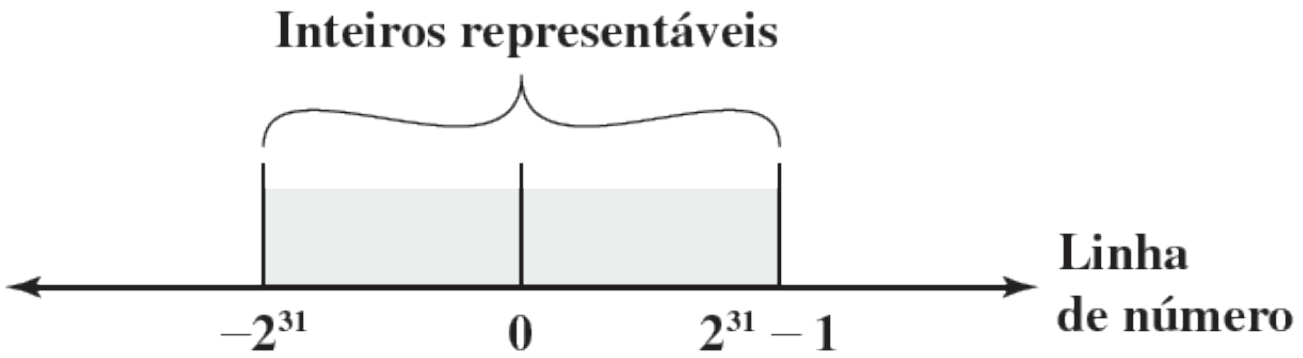
Representação em ponto flutuante

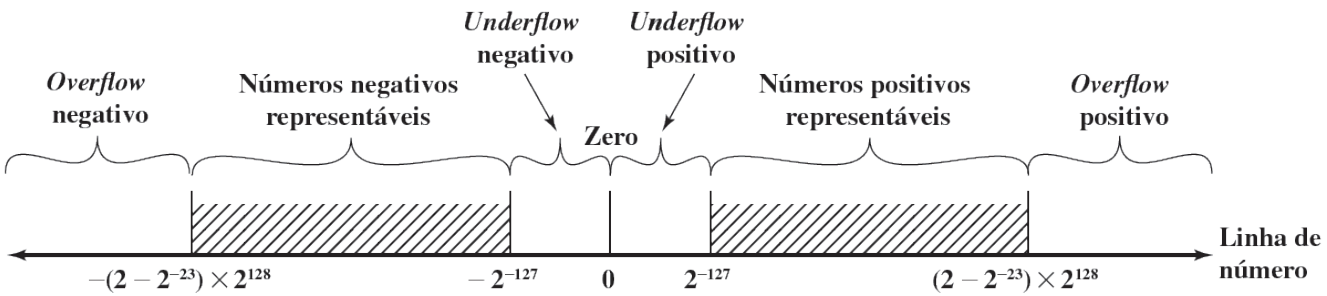
- Abaixo vemos um formato típico de ponto flutuante com 32 bits.
- O bit mais à esquerda armazena o sinal do número (0 = positivo, 1 = negativo).
- O valor do expoente é armazenado nos 8 bits seguintes.
- A representação usada é conhecida como representação polarizada.



Representação em ponto flutuante

- Números expressos em formatos típicos de 32 bits:



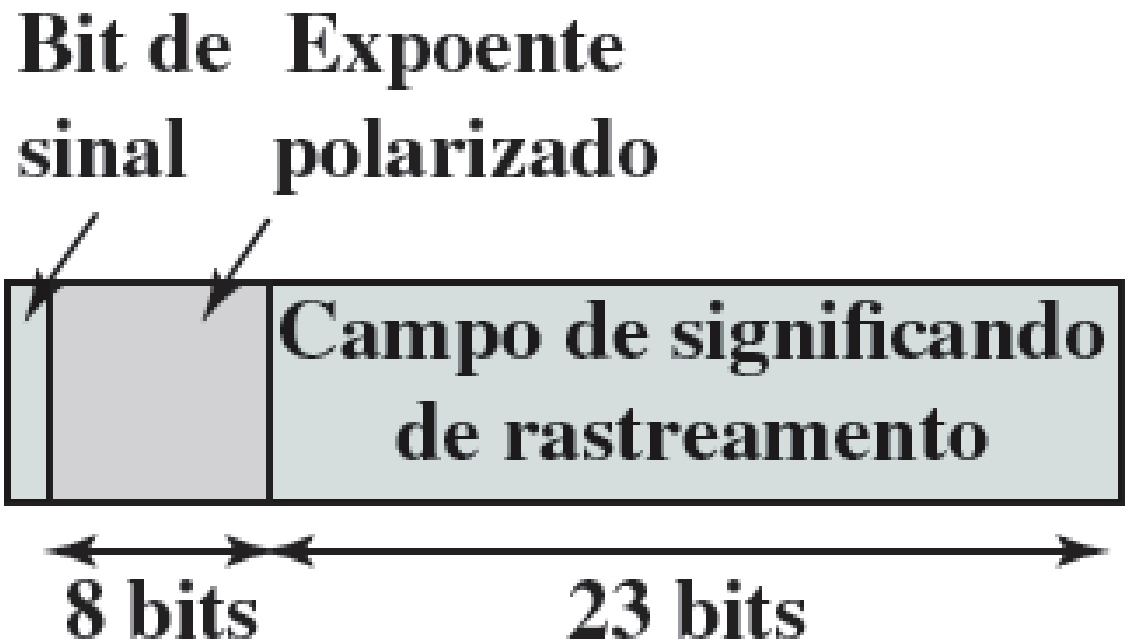


Padrão do IEEE para a representação binária em ponto flutuante

- O IEEE 754 de 2008 define os seguintes tipos diferentes de formatos de ponto flutuante:
- **Formato aritmético:** Todas as operações obrigatórias definidas pelo padrão são aceitas pelo formato.
- **Formato básico:** Cobre cinco representações de ponto flutuante, três binárias e duas decimais, cujas codificações são especificadas pelo padrão e podem ser usadas pela aritmética.
- **Formato de intercâmbio:** Uma codificação de tamanho fixo completamente especificada que possibilita a troca de dados entre plataformas distintas.

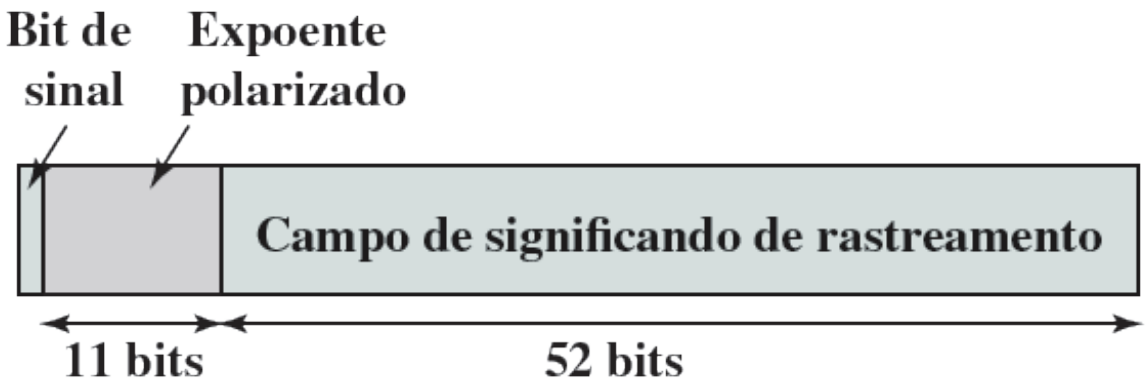
Padrão do IEEE para a representação binária em ponto flutuante

- Formatos do IEEE 754:
- Formato binário32



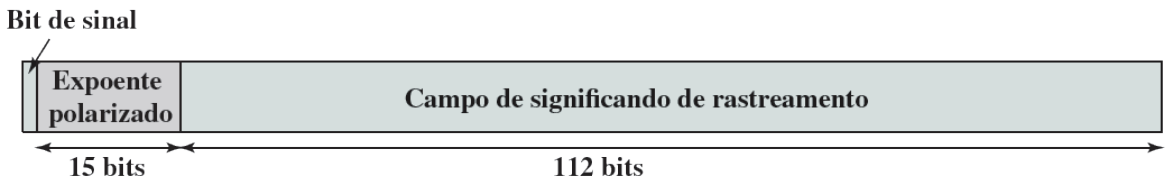
Padrão do IEEE para a representação binária em ponto flutuante

- Formatos do IEEE 754:
- Formato binário64



Padrão do IEEE para a representação binária em ponto flutuante

- Formatos do IEEE 754:
- Formato binário128



Bibliografia Básica

- STALLINGS, W. **Arquitetura e Organização de Computadores**. 10 ed. São Paulo: Pearson, 2017;
- TANENBAUM, A. S. **Organização Estruturada de Computadores**. 5 ed. Pearson 2007;
- HENNESY, J. PATTERSON, D. **Organização e Projeto de Computadores**. 3 ed. Editora Campus, 2005.

Obrigado!
Dúvidas?

Guilherme Henrique de Souza Nakahata

guilhermenakahata@gmail.com

https://github.com/GuilhermeNakahata/UNESPAR-2024