Organização e Arquitetura de Computadores I

Aritmética Computacional

Sumário

- Unidade Lógica e Aritmética
- Representação de Números Inteiros
- Aritmética de Números Inteiros
- Representação de Números de Ponto Flutuante
- Aritmética de Números de Ponto Flutuante

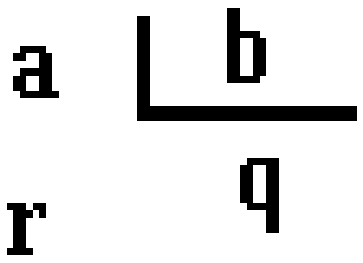
Aritmética de Números Inteiros

- **Divisão**
  - A divisão é mais complexa que a multiplicação, embora seja baseada nos mesmos princípios gerais. A operação envolve repetidas execuções de adição, subtração e deslocamento.
    - Os bits do dividendo são examinados, da esquerda para a direita, até que se obtenha um conjunto de bits que represente um número maior ou igual ao divisor.
    - Enquanto não ocorrer são inseridos 0's no quociente, da esquerda para a direita.

Aritmética de Números Inteiros

- **Divisão**
  - Quando encontrar um número igual ou maior (diz-se que o divisor divide o número) é colocado um 1 no quociente e o divisor é subtraído do dividendo parcial.
  - O resto é chamado de resto parcial. A partir desse ponto, a divisão segue um padrão cíclico. A cada ciclo, bits adicionais do dividendo são anexados ao resto parcial, até que o resultado seja maior ou igual ao divisor.
  - O processo continua até que todos os bits do dividendo tenham sido examinados.

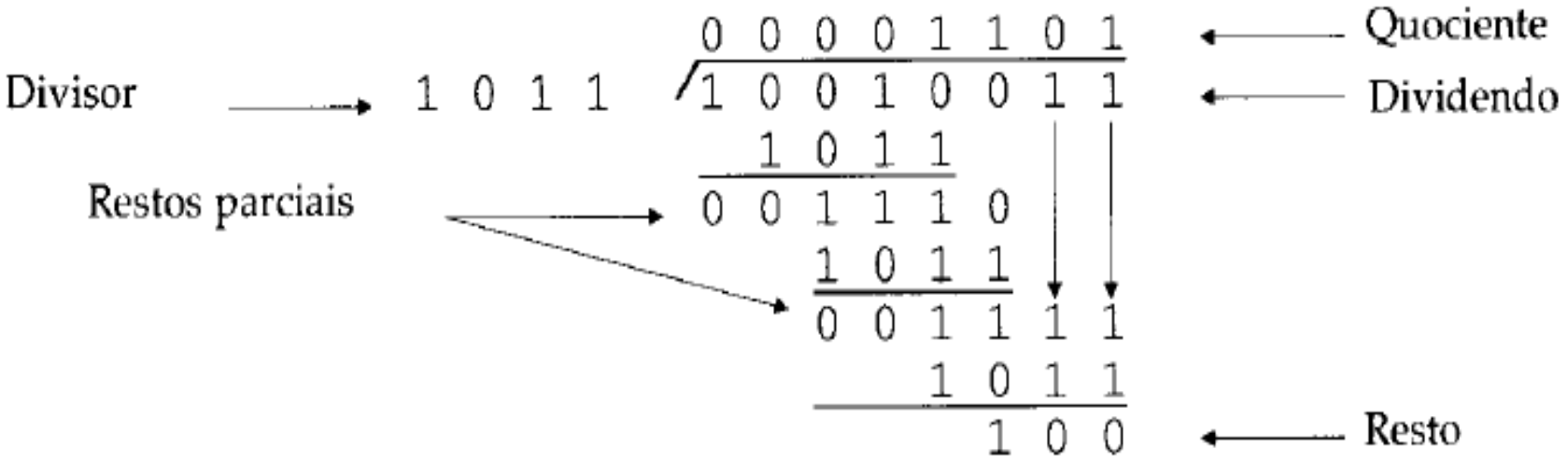
Aritmética de Números Inteiros



- **a**, dividendo
- **b**, divisor
- **q**, quociente
- **r**, resto

Aritmética de Números Inteiros

- **Divisão**

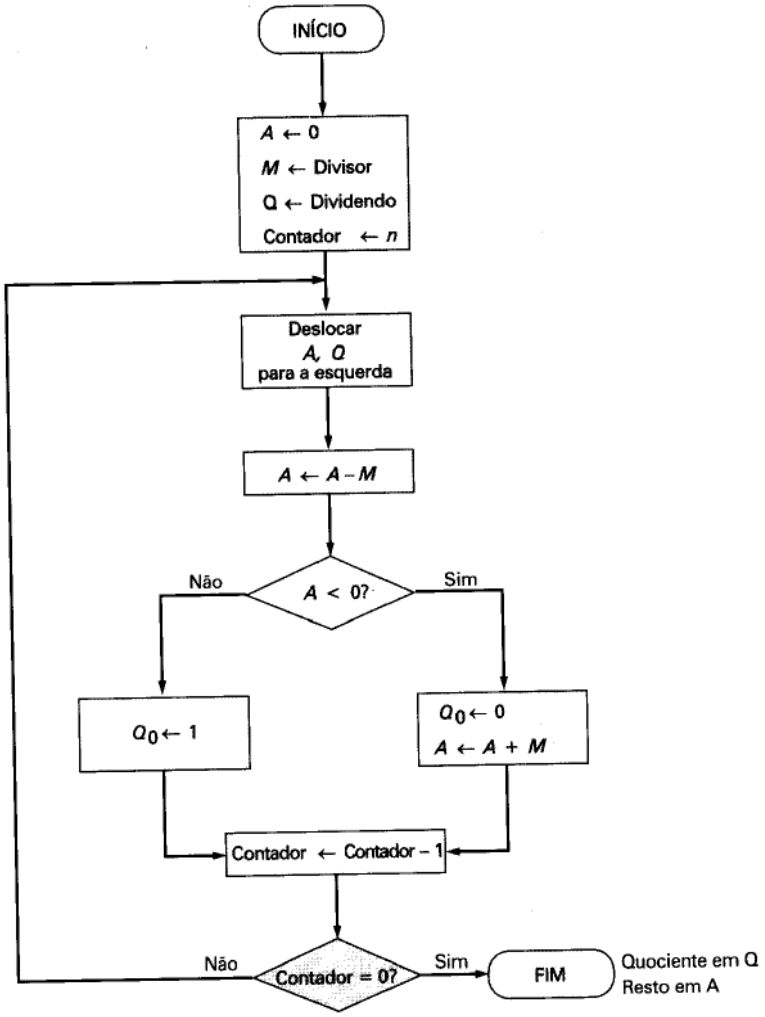


Aritmética de Números Inteiros

Divisão

- Algoritmo de máquina para o processo de divisão:
  1. O divisor é colocado no registrador **M** e o dividendo nos registradores **A** e **Q** (o dividendo será expresso como um número em complemento de 2).
  2. Deslocar o conteúdo dos registradores A e Q, um bit à esquerda.
  3. Se **M** e **A** têm o mesmo sinal, **A<-A-M**; senão, **A<-A+M**.
  4. A operação anterior será bem sucedida se o sinal de **A** for o mesmo antes e depois da operação
     - Se (**A_{n-1} antes == A_{n-1} depois) ou (A=0...0)**, faça **Q_0=1**;
     - Senão, se **A_{n-1} antes != A_{n-1} depois** (sinal do valor contido em **A** antes diferente do sinal de depois), restaure o valor de **A**.
  5. Repita 2 a 4 de acordo com o tamanho da palavra armazenada em Q.
  6. No fim, o resto estará em **A**. Se o divisor e o dividendo tiverem o mesmo sinal, o quociente estará em **Q**; caso contrário, o quociente correto é o complemento de 2 do número armazenado em **Q**.

Organização e Arquitetura de Computadores I



Aritmética de Números Inteiros

7 ÷ 3

A	Q	M = 0011
0000	0111	Valor inicial
0000	1110	Deslocar
1101		Subtrair
0000	1110	Restaurar
0001	1100	Deslocar
1110		Subtrair
0001	1100	Restaurar
0011	1000	Deslocar
0000		Subtrair
0000	1001	Fazer $Q_0 = 1$
0001	0010	Deslocar
1110		Subtrair
0001	0010	Restaurar

Aritmética de Números Inteiros

7 ÷ (-3)

A	Q	M = 1101
0000	0111	Valor inicial
0000	1110	Deslocar
1101		Adicionar
0000	1110	Restaurar
0001	1100	Deslocar
1110		Adicionar
0001	1100	Restaurar
0011	1000	Deslocar
0000		Adicionar
0000	1001	Fazer $Q_0 = 1$
0001	0010	Deslocar
1110		Adicionar
0001	0010	Restaurar

Rep. de Números de Ponto Flutuante

- Usando uma notação de ponto fixo, é possível representar certa faixa de números inteiros positivos e negativos, centrada em 0. Esse formato permite também a representação de números com parte fracionária, bastando fixar uma posição adequada para a vírgula que separa a parte inteira e a parte fracionária.
- **Em operações com números muito grandes a parte fracionária pode ser perdida.**

Rep. de Números de Ponto Flutuante

- A solução é a utilização de notação científica.

  o número 976.000.000.000.000

  pode ser representado como $9{,}76 \times 10^{14}$

  o número 0,000000000000976

  pode ser representado como $9{,}76 \times 10^{-14}$

- Na notação científica, a vírgula é deslizada dinamicamente para uma posição conveniente e é usado um expoente de 10 adequado, para representar o mesmo valor original.

Rep. de Números de Ponto Flutuante

- A mesma abordagem pode ser usada para números binários.

$$\pm M \times B^{\pm E}$$

Onde:

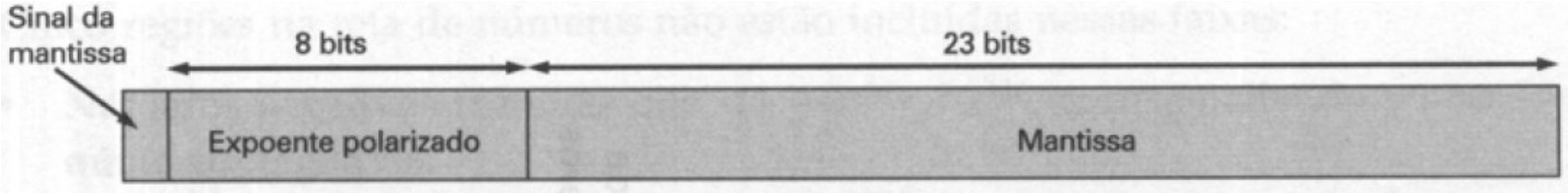
±: sinal

M: mantissa

E: expoente

B: base (implícita e não precisa ser armazenada pois é a mesma para todos os números)

Rep. de Números de Ponto Flutuante



- Para o expoente utiliza-se a representação polarizada, um valor fixo, chamado de polarização, é subtraído do expoente polarizado para sabermos o verdadeiro valor do campo.
- A mantissa é representada no último campo com uma particularidade, o bit mais à esquerda da mantissa é sempre 1 e ele fica implícito. Temos assim 24 bits para representar a mantissa (23 + 1).
- O sinal da mantissa é representado por um bit, 0 para positivo e 1 para negativo.

Rep. de Números de Ponto Flutuante

- Exemplos de representação de números de ponto flutuante de 32 bits:

  $0{,}11010001 \times 2^{10100} = 0\ \ 10010011\ \ 10100010000000000000000$

  $-0{,}11010001 \times 2^{10100} = 1\ \ 10010011\ \ 10100010000000000000000$

  $0{,}11010001 \times 2^{-10100} = 0\ \ 01101011\ \ 10100010000000000000000$

  $-0{,}11010001 \times 2^{-10100} = 1\ \ 01101011\ \ 10100010000000000000000$

- Características:
  - O sinal é armazenado no primeiro bit da palavra
  - O primeiro bit da mantissa verdadeira é sempre 1 e está implícito
  - O valor 127 é adicionado ao expoente verdadeiro para ser armazenado no campo do expoente
  - A base é 2

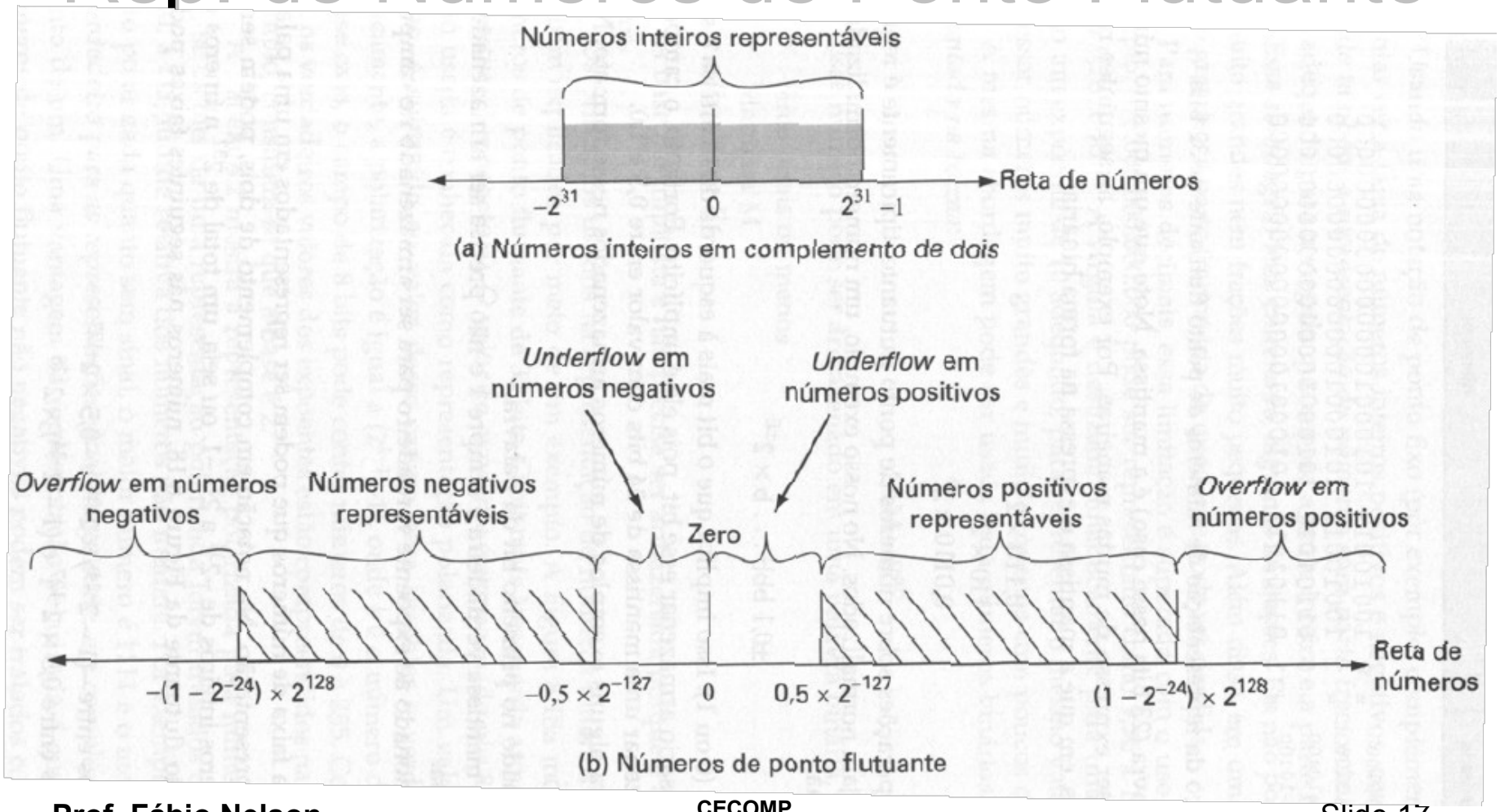
Rep. de Números de Ponto Flutuante

- Na notação em complemento de dois, podem ser representados todos os números inteiros de **-2^{31} a $2^{31}-1$**, ou seja, um total de **2^{32} números distintos**.
- No formato de ponto flutuante, números nas seguintes faixas podem ser representados:

  Números negativos entre $-(1 - 2^{-24}) \times 2^{128}$ e $-0,5 \times 2^{-127}$

  Números positivos entre $0,5 \times 2^{-127}$ e $(1 - 2^{-24}) \times 2^{128}$

Rep. de Números de Ponto Flutuante



Rep. de Números de Ponto Flutuante

- Cinco regiões na reta de números não estão incluídas
- nessas faixas:
  - Números negativos menores que $-(1 - 2^{-24}) \times 2^{128}$ (***overflow* em números negativos**)
  - Números negativos maiores que $-0,5 \times 2^{-127}$ (***underflow* em números negativos**)
  - **Zero**
  - Números positivos menores que $-0,5 \times 2^{-127}$ (***underflow* em números positivos**)
  - Números positivos maiores que $(1 - 2^{-24}) \times 2^{128}$ (***overflow* em números positivos**)

Rep. de Números de Ponto Flutuante

- Representações de ponto flutuante incluem um padrão de bits especial para designar zero.
- O *overflow* ocorre quando a magnitude do resultado é maior do que o maior valor que pode ser expresso com expoente igual a 128 (por exemplo, $2^{120} \times 2^{100} = 2^{220}$).
- A ocorrência de *underflow* é um problema menos sério, pois o resultado geralmente pode ser aproximado satisfatóriamente por 0.

Rep. de Números de Ponto Flutuante

- Os números representados na notação de ponto flutuante não estão igualmente distribuídos ao longo da reta de números, como é o caso de números de ponto fixo.
  - **Maior quantidade de valores representáveis próximo à origem**
  - **A quantidade diminui com a distância à origem**
- Se aumentarmos o número de bits do expoente, expandimos a faixa de valores representáveis. Com a diminuição da quantidade de números representáveis.
- A maioria dos computadores oferece, pelo menos, números de precisão simples e números de precisão dupla (por exemplo, 32bits e 64bits).

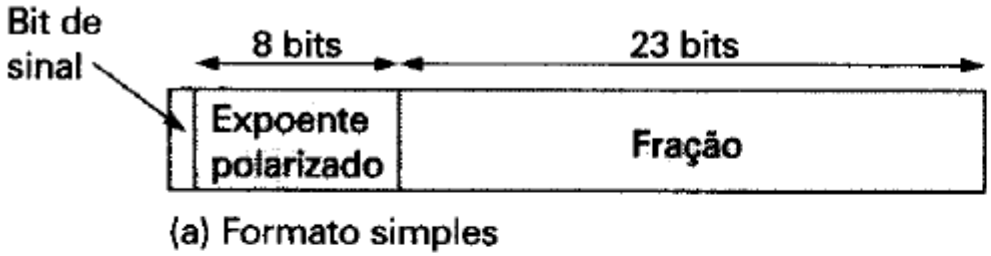
Rep. de Números de Ponto Flutuante

- Os cálculos em ponto flutuante, tendem a serem arredondados para os valores mais próximos que a notação possibilitar.

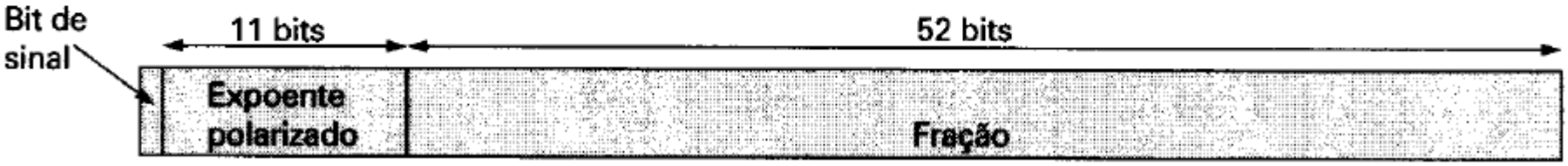
Rep. de Números de Ponto Flutuante

- **Padrão IEEE**
  - A mais importante representação de ponto flutuante é definida no padrão IEE 754 de 1985.
  - Desenvolvido para facilitar a portabilidade de programas entre processadores distintos.
  - **Tem sido largamente adotado, sendo usado em quase todos os processadores e co-processadores aritméticos modernos.**
  - Define um formato simples de 32 bits e um formato duplo de 64 bits, com expoentes de 8 e 11 bits, respectivamente.

Rep. de Números de Ponto Flutuante



(a) Formato simples



(b) Formato duplo

Rep. de Números de Ponto Flutuante

- **Padrão IEEE**
  - Foram criados dois formatos estendidos, simples e duplo, cujo formato exato é dependente da implementação.
  - Os formatos estendidos incluem bits adicionais no expoente (alcance estendido) e na mantissa (precisão estendida).
  - Evita o arredondamento excessivo.

Rep. de Números de Ponto Flutuante

- **Padrão IEEE**
  - Parâmetros do formato IEEE 754

Parâmetro	Formato			
	Simples	Simples estendido	Duplo	Duplo estendido
Tamanho da palavra (bits)	32	≥43	64	≥79
Tamanho do expoente (bits)	8	≥11	11	≥15
Polarização do expoente	127	Não especificado	1023	Não especificado
Expoente máximo	127	≥1023	1023	≥16383
Expoente mínimo	–126	≤–1022	–1022	≤–16382
Faixa de números (base 10)	10^{-38}, 10^{+38}	Não especificado	10^{-308}, 10^{+308}	Não especificado
Tamanho da mantissa (bits)*	23	≥31	52	≥63
Número de expoentes	254	Não especificado	2046	Não especificado
Número de frações	2^{23}	Não especificado	2^{52}	Não especificado
Número de valores	$1{,}98 \times 2^{31}$	Não especificado	$1{,}99 \times 2^{63}$	Não especificado

* Não inclui o bit implícito

Rep. de Números de Ponto Flutuante

- **Padrão IEEE**
  - Classes de números representados:
    - Se o intervalo de valores do expoente é de 1 a 254 no formato simples ou de 1 a 2.046 no formato duplo, são representados números de ponto flutuante normalizados (1,bbb...)
    - Um expoente igual a zero junto com uma fração igual a zero representa zero, podendo ser positivo ou negativo.
    - Um expoente totalmente preenchido com uns junto com uma fração igual a zero representa infinito, positivo ou negativo.
    - Um expoente de valor zero junto com uma fração diferente de zero representa um número não-normalizado (0,bbb...).
    - Um expoente totalmente preenchido com uns junto com uma fração diferente de zero representa o valor NaN (*Not a Number* – não é um número).

Organização e Arquitetura de Computadores I

	Precisão simples (32 bits)				Precisão dupla (64 bits)			
	Sinal	Expoente polarizado	Fração	Valor	Sinal	Expoente polarizado	Fração	Valor
Zero positivo	0	0	0	0	0	0	0	0
Zero negativo	1	0	0	–0	1	0	0	–0
Infinito positivo	0	255 (todos 1s)	0	∞	0	2047 (todos 1s)	0	∞
Infinito negativo	1	255 (todos 1s)	0	–∞	1	2047 (todos 1s)	0	–∞
NaN silencioso	0 ou 1	255 (todos 1s)	≠ 0	NaN	0 ou 1	2047 (todos 1s)	≠ 0	NaN
NaN sinalizador	0 ou 1	255 (todos 1s)	≠ 0	NaN	0 ou 1	2047 (todos 1s)	≠ 0	NaN
Diferente de zero, normalizado positivo	0	0 < e < 255	f	$2^{e-127}(1,f)$	0	0 < e < 2047	f	$2^{e-1023}(1,f)$
Diferente de zero, normalizado negativo	1	0 < e < 255	f	$-2^{e-127}(1,f)$	1	0 < e < 2047	f	$-2^{e-1023}(1,f)$
Não-normalizado positivo	0	0	f ≠ 0	$2^{e-126}(0,f)$	0	0	f ≠ 0	$2^{e-1022}(0,f)$
Não-normalizado negativo	1	0	f ≠ 0	$-2^{e-126}(0,f)$	1	0	f ≠ 0	$-2^{e-1022}(0,f)$

Aritmética de Números de Ponto Flutuante

- Na aritmética de ponto flutuante, a adição e a subtração são operações mais complexas que a multiplicação e a divisão. **Devido a necessidade de alinhar os operandos, de modo que torne seus expoentes iguais**.
- Possui quatro fases:
  - Verificar se algum operando é zero
  - Alinhar as mantissas
  - Adicionar ou subtrair as mantissas
  - Normalizar o resultado

Aritmética de Números de Ponto Flutuante

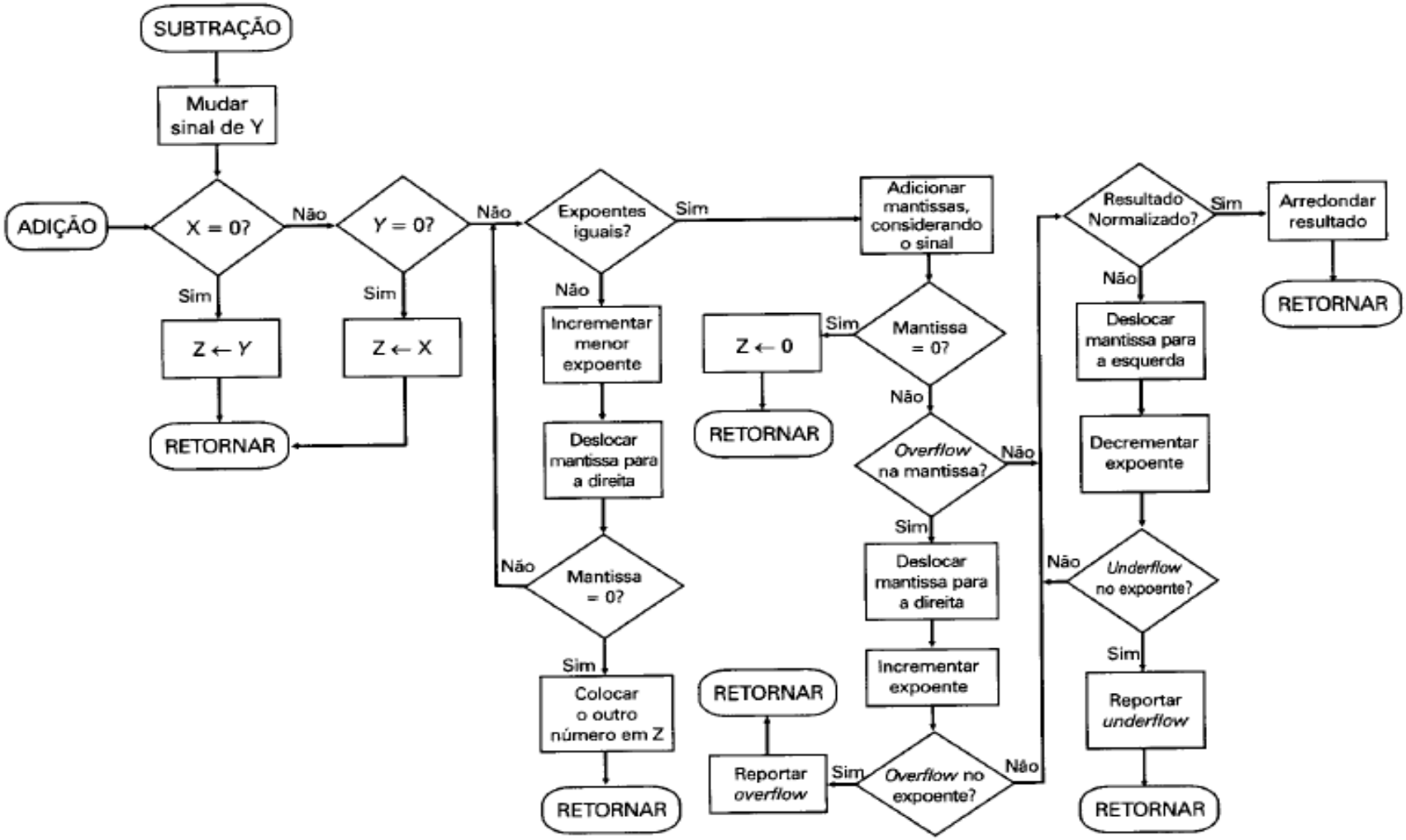
- **Passos para uma adição ou subtração**:
  - A adição e a subtração são idênticas, exceto pela mudança de sinal, ou seja, caso seja uma subtração, o sinal do subtraendo é mudado.
  - Manipular os números de modo que seus expoentes se tornem iguais.
  - Soma das duas mantissas, levando em conta o seus sinais:
    - Pode ocorrer um *overflow* na mantissa por um dígito, então os dígitos são movidos para a direita e o expoente é aumentado em um dígito (abrindo a possibilidade de um *overflow* no expoente, gerando um erro).
  - Normalização do resultado, deslocar os dígitos da mantissa para a esquerda, até que o dígito mais significativo seja diferente de zero, alterando também o expoente.
  - O resultado é arredondado.

Aritmética de Números de Ponto Flutuante

$$(123 \times 10^{0}) + (456 \times 10^{-2}) =$$

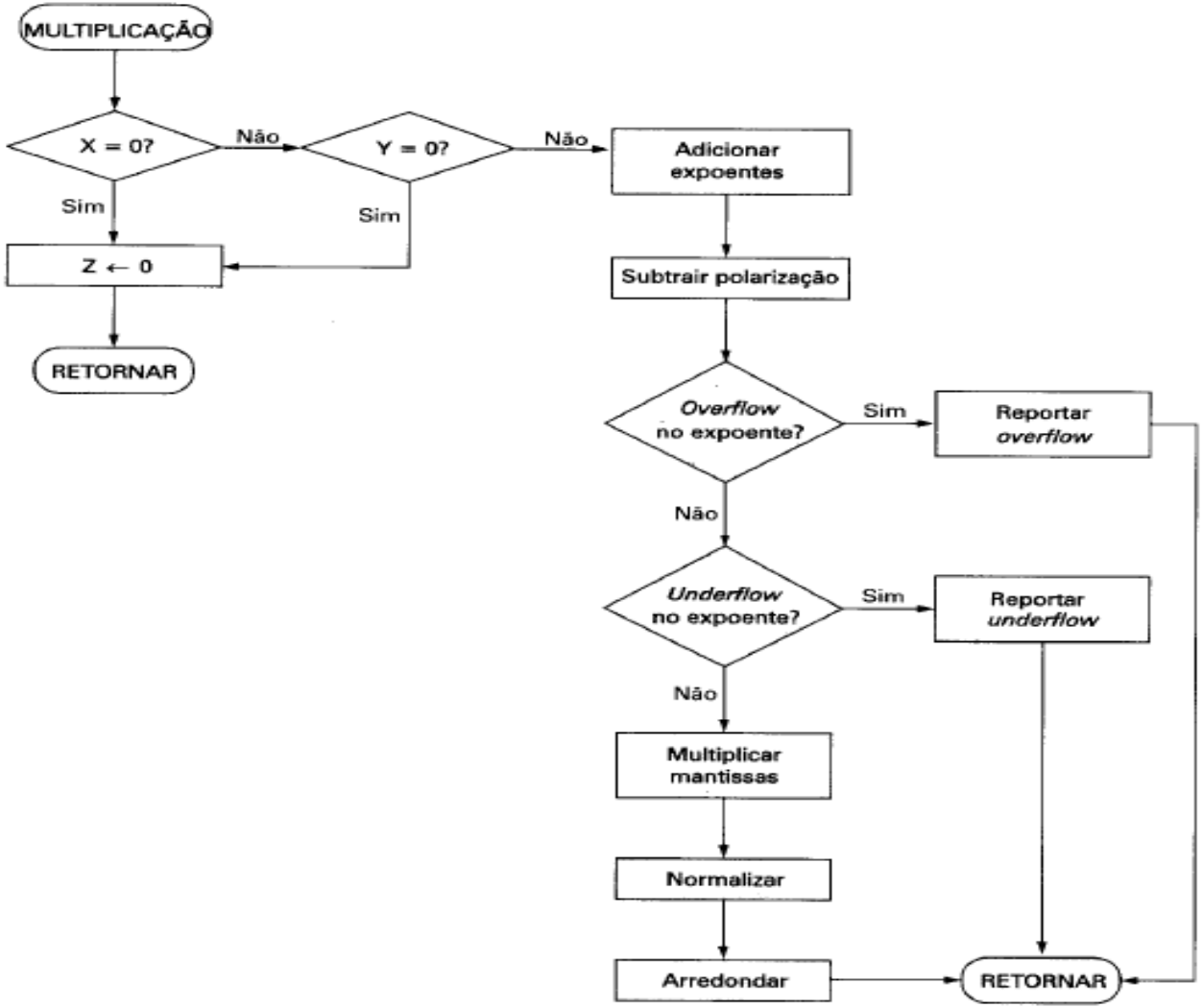
$$(123 \times 10^{0}) + (4{,}56 \times 10) =$$

$$(127{,}56 \times 10^{0})$$


SUBTRAÇÃO
Mudar sinal de Y
ADIÇÃO
X = 0?
Não
Y = 0?
Não
Expoentes iguais?
Sim
Adicionar mantissas, considerando o sinal
Sim
Z ← Y
Sim
Z ← X
Não
Incrementar menor expoente
Z ← 0
Sim
Mantissa = 0?
RETORNAR
RETORNAR
Não
Deslocar mantissa para a direita
RETORNAR
Overflow na mantissa?
Não
Não
Mantissa = 0?
Sim
Deslocar mantissa para a direita
Sim
Colocar o outro número em Z
RETORNAR
RETORNAR
Incrementar expoente
Reportar overflow
Sim
Overflow no expoente?
Não
Resultado Normalizado?
Sim
Arredondar resultado
RETORNAR
Não
Deslocar mantissa para a esquerda
Decrementar expoente
Não
Underflow no expoente?
Sim
Reportar underflow
RETORNAR

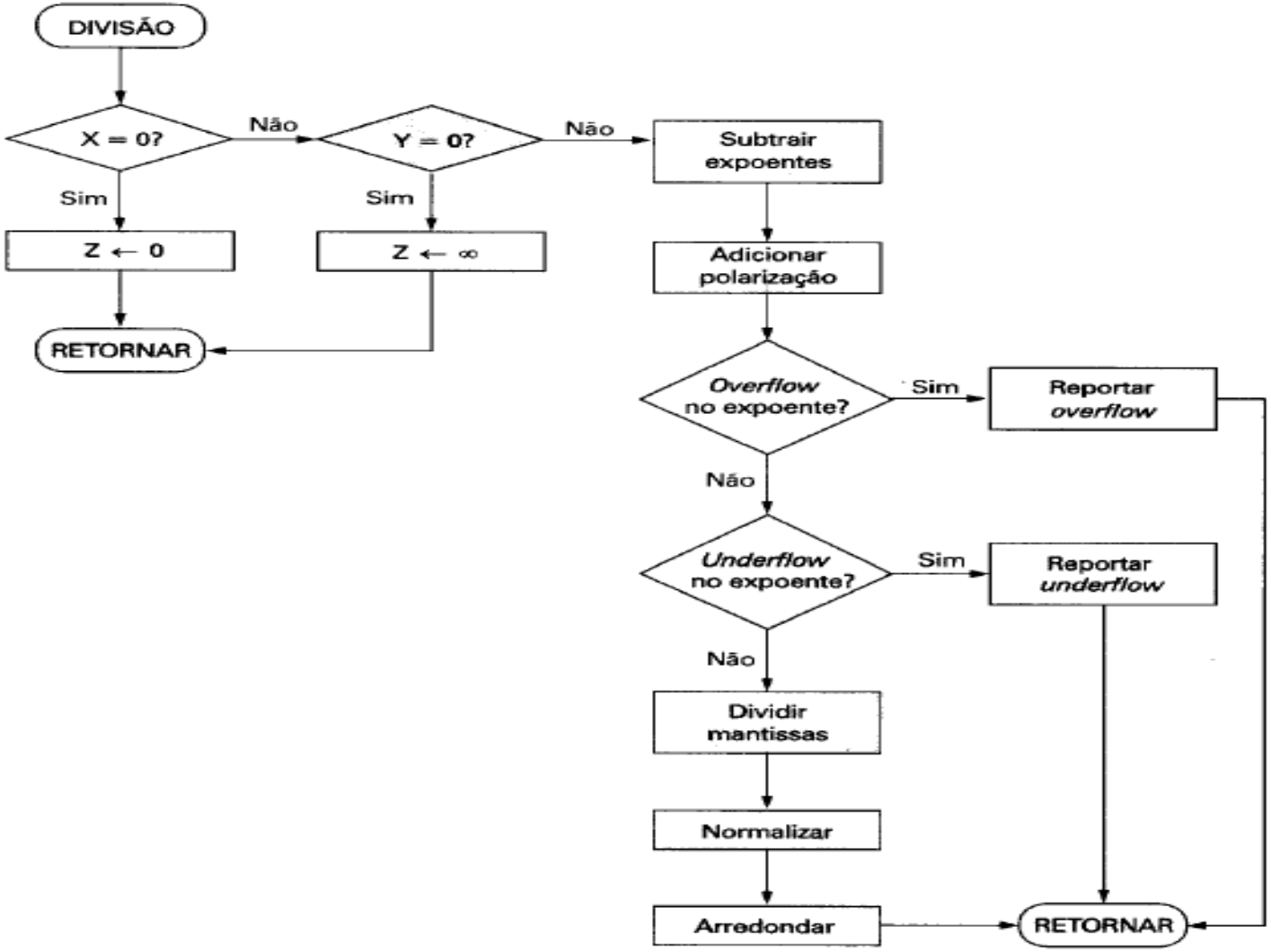
Aritmética de Números de Ponto Flutuante

- A multiplicação e a divisão de ponto flutuante são operações muito mais simples do que a adição e a subtração.
  - Caso qualquer um dos operandos for 0, o resultado será 0.
  - Soma-se os expoentes. Se eles forem armazenados na forma polarizada, a soma dobrará a polarização (o valor da polarização deve ser subtraído da soma dos expoentes).
  - Se o expoente do produto estiver dentro da faixa de números representáveis, as mantissas devem ser multiplicadas (mesma forma dos números inteiros), levando em conta seus sinais.
  - Ocorre a normalização e depois o arredondamento.


MULTIPLICAÇÃO
X = 0?
Não
Y = 0?
Não
Adicionar expoentes
Sim
Sim
Z ← 0
Subtrair polarização
RETORNAR
Overflow no expoente?
Sim
Reportar overflow
Não
Underflow no expoente?
Sim
Reportar underflow
Não
Multiplicar mantissas
Normalizar
Arredondar
RETORNAR

Aritmética de Números de Ponto Flutuante

- No caso de divisão temos o seguintes passos:
  - Testar se algum dos operandos é zero. Caso seja o divisor, o resultado é um erro ou infinito. Caso o dividendo seja zero, o resultado será zero.
  - O expoente do divisor será subtraído do expoente do dividendo (isso remove a polarização que deve ser adicionada ao resultado).
  - Em seguida, são efetuados testes de *overflow* e *underflow* no expoente.
  - Divisão das mantissas.
  - Normalização e arredondamento.


DIVISÃO
X = 0?
Não
Y = 0?
Não
Subtrair expoentes
Sim
Sim
Z ← 0
Z ← ∞
Adicionar polarização
RETORNAR
Overflow no expoente?
Sim
Reportar overflow
Não
Underflow no expoente?
Sim
Reportar underflow
Não
Dividir mantissas
Normalizar
Arredondar
RETORNAR

Aritmética de Números de Ponto Flutuante

- **Política de Arredondamento**
  - Afeta a precisão do resultado.
  - O resultado de qualquer operação sobre as mantissas geralmente é armazenado em um registrador de tamanho maior.
  - Quando o resultado é colocado novamente no formato de ponto flutuante, os bits extras devem ser descartados.
  - Diversas técnicas para arredondamento foram exploradas. De fato, o padrão IEEE relaciona quatro abordagens alternativas:
    - **Arredondar para o mais próximo**: o resultado é arredondado para o número representável mais próximo.
    - **Arredondar para cima (+∞)**: o resultado é arredondado para cima, na direção de infinito positivo.
    - **Arredondar para baixo (- ∞)**: o resultado é arredondado para baixo, na direção de infinito negativo.
    - **Arredondar para 0**: o resultado é arredondado na direção de zero.

Aritmética de Números de Ponto Flutuante

- O padrão IEEE traz também procedimentos específicos para que a aritmética de ponto produza resultados uniformes, previsíveis e independente de plataformas:
  - **Infinito**:

5 + (+∞) = +∞ 5 - (-∞) = -∞ 5 x (-∞) = -∞ 5 ÷ (+∞) = 0

  - **NaN silencioso e NaN sinalizador**: se propaga sem gerar exceção e gera exceção de operação inválida, respectivamente.
  - **Números não-normalizados**: trata casos de *underflow* de expoente.

Aritmética de Números de Ponto Flutuante

Operação	NaN silencioso produzido por
Qualquer	Qualquer operação sobre um NaN sinalizador
Adição ou subtração	Subtração de números de magnitude infinita $(+\infty) + (-\infty)$ $(-\infty) + (+\infty)$ $(+\infty) - (+\infty)$ $(-\infty) - (-\infty)$
Multiplicação	$0 \times \infty$
Divisão	$\frac{0}{0}$ ou $\frac{\infty}{\infty}$
Resto	x REM 0 ou ∞ REM y
Raiz quadrada	$\sqrt{x}$ onde $x < 0$