Aritmética do Computador

Inteiros
Ponto Flutuante

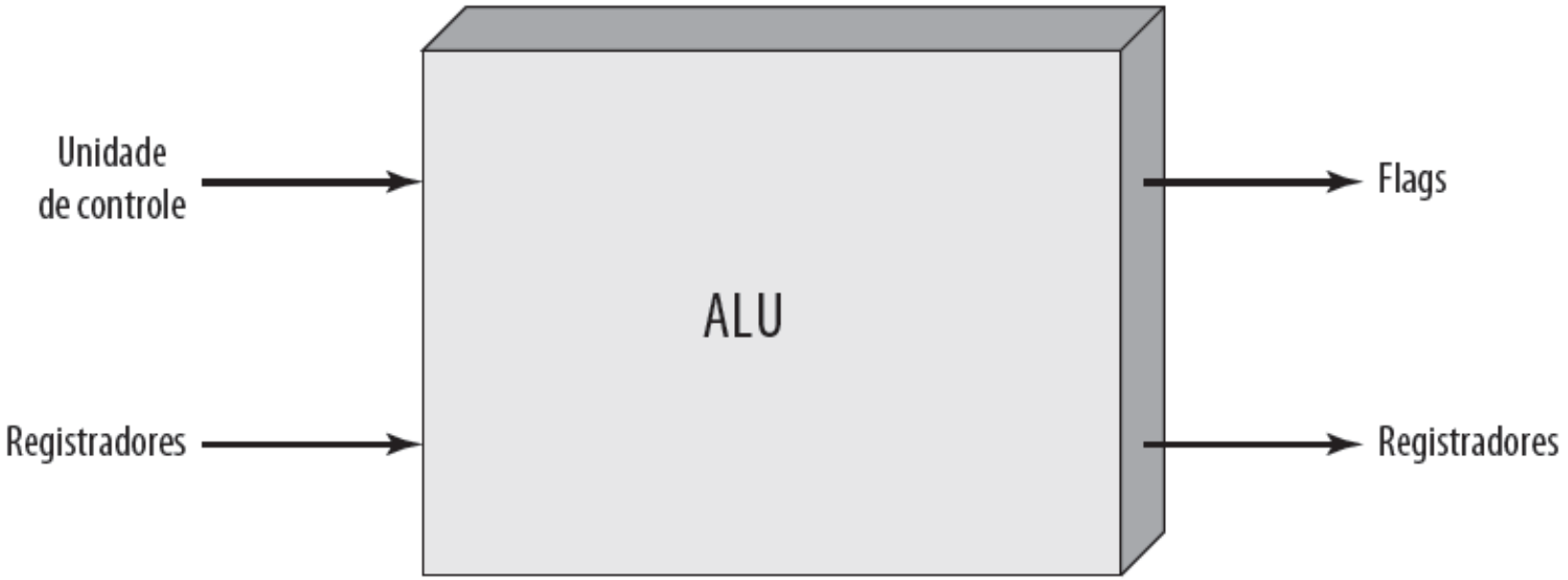
Coordenadoria de Informática
Sistemas de Informação
Profs. Vitor Faiçal Campana
Francisco Rapchan
Material cedido pelo Prof. Flávio Giraldeli

2019/2

Unidade Lógica e Aritmética (ULA) Aritmetic and Logic Unit (ALU)

- Faz os cálculos.
- Tudo o mais no computador existe para atender a essa unidade.
- Trata de inteiros.
- Pode tratar de números de ponto flutuante (reais).
- Pode ser FPU separada (coprocessador matemático).
- A partir do Intel 486DX, é integrada a CPU.

Entradas e saídas da ALU



Aritmética de Inteiros

Um mundo "ideal"...

Representação de inteiros

- No sistema numérico binário, números quaisquer podem ser representados apenas com os algarismos 0 e 1, o sinal de menos e a vírgula, ou vírgula fracionada.

  $-1101{,}0101_2 = -13{,}3125_{10}$
- Para finalidades de armazenamento e processamento no computador, porém, não temos o benefício dos sinais de menos e vírgula (que, como veremos, podem ser representados de outras formas)
- Se nos limitarmos a não negativos, a representação é direta.

```
00000000 =   0
00000001 =   1
00101001 =  41
10000000 = 128
11111111 = 255
```

- De modo geral, se temos uma sequencia de n bits $a_{n-1}a_{n-2}a_1a_0$ como um inteiro sem sinal, seu valor A será:

$$A = \sum_{i=0}^{n-1} 2^i a_i$$

Sinal-Magnitude

- Bit mais à esquerda é bit de sinal.
  - 0 significa positivo.
  - 1 significa negativo.
  - Exemplos:
    - +18 = 00010010.
    - -18 = 10010010.
- Problemas:
  - Precisa considerar sinal e magnitude na aritmética.
  - Duas representações de zero (+0 e -0).
    - Mais difícil de testar se um resultado é zero (operação muito comum em computação)
  - Por isso, raramente é usada na prática.

Complemento a dois

- +127 = 01111111
- +3 = 00000011
- +2 = 00000010
- +1 = 00000001
- +0 = 00000000
- -1 = 11111111
- -2 = 11111110
- -3 = 11111101
- -128 = 10000000

Benefícios do complemento a dois

- Uma representação de zero.
- Aritmética funciona com facilidade (ver mais adiante).
- Negação é muito fácil.
  - +3 00000011
  - Complemento Booleano gera 11111100
  - Some 1 ao LSB e tem-se -3 11111101

Negação especial

- Caso 1:
  - 0 = 00000000
  - Not bit a bit 11111111
  - Some 1 ao LSB +1
  - Resultado 100000000
  - Estouro ignorado, portanto, - 0 = + 0
- Caso 2:
  - -128 = 10000000
  - Not bit a bit 01111111
  - Some 1 ao LSB +1
  - Resultado 10000000
  - Portanto:
    - -(-128) = -128 (ERRO!)
    - Monitore MSB (bit de sinal).
    - Ele deve mudar durante a negação.

Intervalo de números

- Complemento a 2 com 8 bits:
  - +127 = 01111111 = 2^7 -1
  - -128 = 10000000 = -2^7
- Complemento a 2 com 16 bits:
  - +32767 = 011111111 11111111 = 2^{15} - 1
  - -32768 = 100000000 00000000 = -2^{15}

Complemento a dois Resumo...

Tabela 9.1 Características da representação e aritmética de complemento a dois

Intervalo	-2^{n-1} até $2^{n-1}-1$
Número de representações de zero	Uma
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 do tamanho 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 estouro ocorre se e somente se o resultado tem o sinal oposto.
Regra de subtração	Para subtrair B de A, apanhe o complemento a dois de B e some-o a A.

Representação de inteiros

Representação decimal	Representação sinal-magnitude	Representação em complemento de dois	Representação polarizada
+8	–	–	1111
+7	0111	0111	1110
+6	0110	0110	1101
+5	0101	0101	1100
+4	0100	0100	1011
+3	0011	0011	1010
+2	0010	0010	1001
+1	0001	0001	1000
+0	0000	0000	0111
–0	1000	–	–
–1	1001	1111	0110
–2	1010	1110	0101
–3	1011	1101	0100
–4	1100	1100	0011
–5	1101	1011	0010
–6	1110	1010	0001
–7	1111	1001	0000
–8	–	1000	–

Conversão entre tamanhos

- Pacote de número positivo com zeros iniciais.
  - +18 = 00010010
  - +18 = 00000000 00010010
- Pacote de números negativos com uns iniciais.
  - -18 = 10010010
  - -18 = 11111111 10010010
- Ou seja, pacote com MSB (bit de sinal).

Aritmética com inteiros

- Adição
  - Adição binária normal.
  - Monitore estouro no bit de sinal.
  - Exemplos:

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

- Subtração
  - Pegue o complemento a dois do subtraendo e some ao minuendo.
    - a - b = a + (-b).
  - Exemplos:

```
      0010 =  2
     +1001 = −7
      1011 = −5

(a) M = 2 = 0010
    S = 7 = 0111
   −S =     1001
```

```
      0101 =  5
     +1110 = −2
     10011 =  3

(b) M = 5 = 0101
    S = 2 = 0010
   −S =     1110
```

```
      1011 = −5
     +1110 = −2
     11001 = −7

(c) M = −5 = 1011
    S =  2 = 0010
   −S =      1110
```

```
      0101 = 5
     +0010 = 2
      0111 = 7

(d) M =  5 = 0101
    S = −2 = 1110
   −S =      0010
```

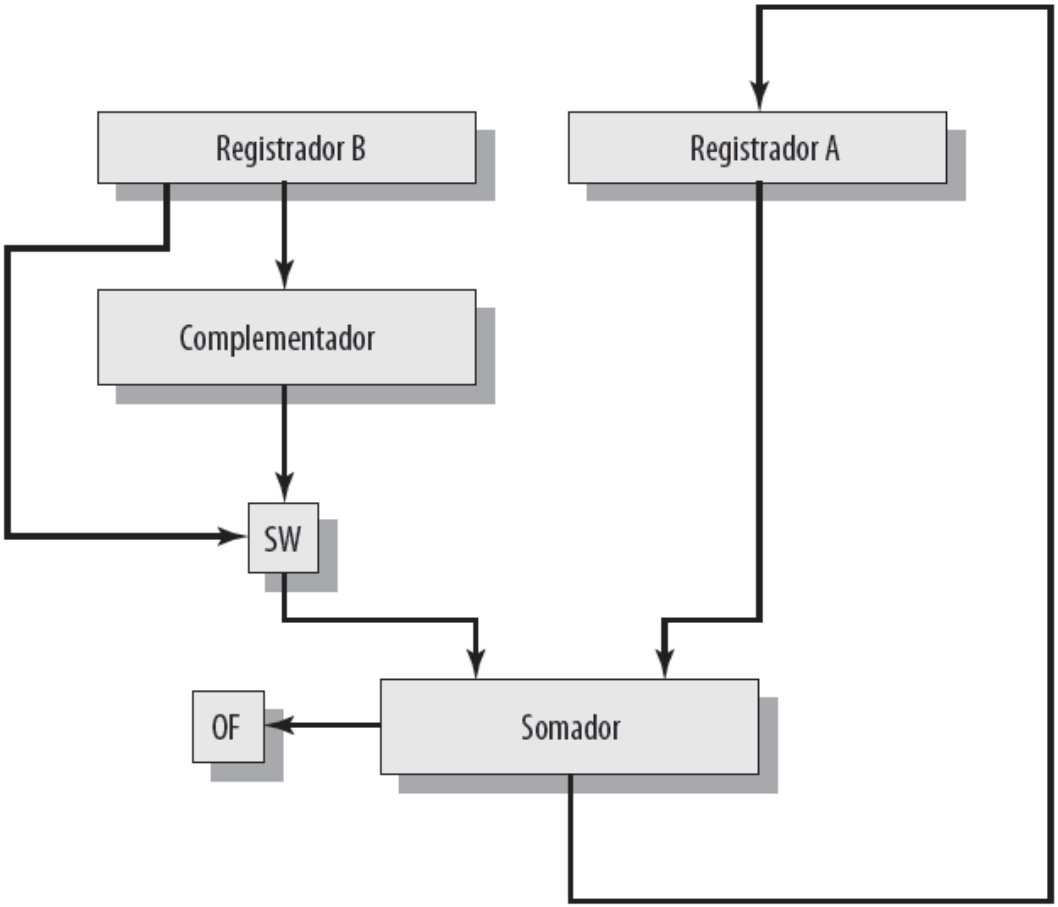
```
      0111 = 7
     +0111 = 7
      1110 = Overflow

(e) M =  7 = 0111
    S = −7 = 1001
   −S =      0111
```

```
      1010 = −6
     +1100 = −4
     10110 = Overflow

(f) M = −6 = 1010
    S =  4 = 0100
   −S =      1100
```

Hardware para adição e subtração



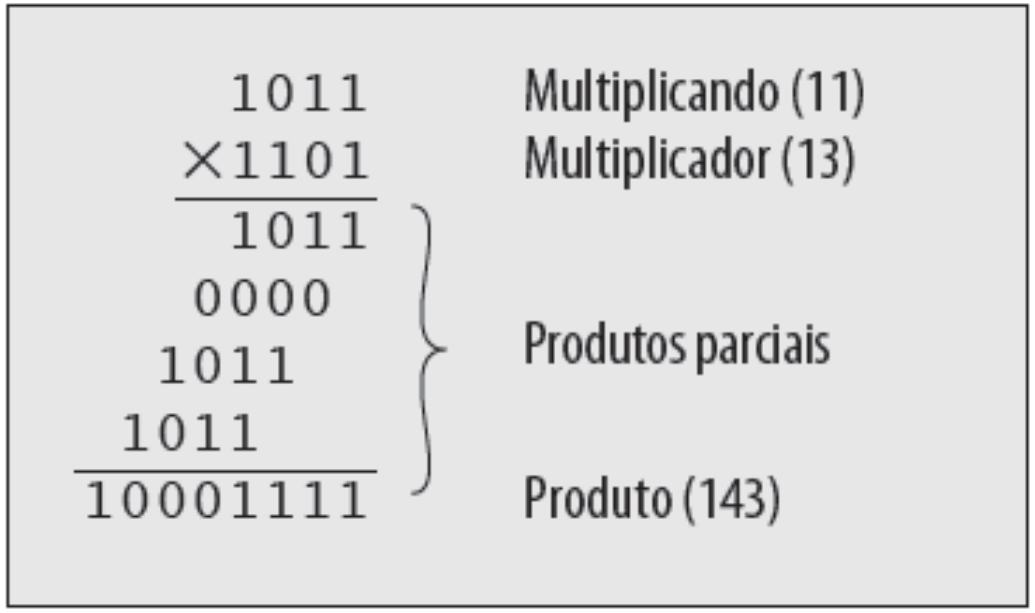
OF = bit de *overflow* (do inglês *overflow bit*)
SW = seletor – multiplexador (seleciona adição ou subtração)

Multiplicação

- Se comparada a adição e subtração, a multiplicação é uma operação complexa.
- Dois casos a se considerar:
  - Inteiros sem sinal
  - Inteiros com sinal

Multiplicação de inteiros sem sinal

- É realizada tal qual todos nós já aprendemos com lápis e papel aplicada aos números decimais. No entanto é até mais simples!
- Exemplo:



Multiplicação de inteiros com sinal

- O método estudado não funciona!
- Solução 1:
  - Converta para positivo, se for preciso.
  - Multiplique como antes.
  - Se sinais diferentes, negue a resposta.
- Solução 2:
  - Algoritmo de Booth
    - É computacionalmente mais leve, porém, por ser bem mais complexo de entender, não será estudado. É preciso simplificar o capítulo para que possamos cumprir com a ementa.

Divisão

- Ainda mais complexa que a multiplicação.
- Números negativos são realmente maus!
- Baseada na divisão longa.
- Logo...

Aritmética de Ponto Flutuante

Um mundo “nem tão ideal”...

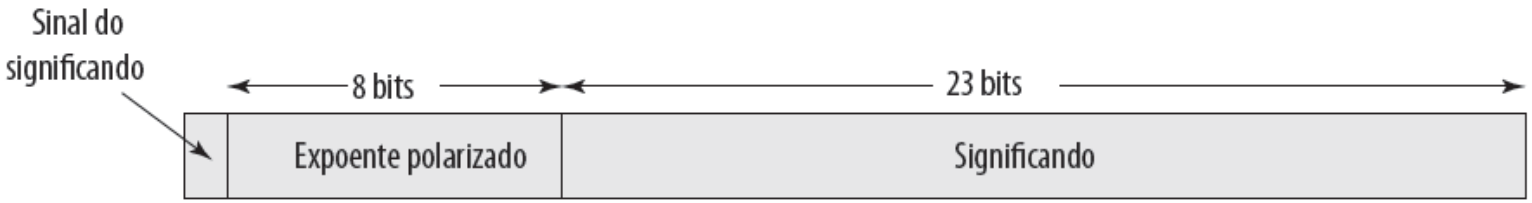
Representação de Ponto Flutuante

- Com a notação de ponto fixo é possível representar um intervalo de inteiros positivos e negativos centrados em 0.
- Assumindo um binário fixo e ponto fracionário, esse formato permite a representação de números também com componente fracionário.
- Limitações:
  - Números muito grandes não podem ser representados, nem frações muito pequenas.
  - A parte fracionária do quociente em uma divisão de dois números grandes poderia ser perdida.
- Solução? Representação em **ponto flutuante** (ideia derivada da notação científica).

Ponto Flutuante

- Formato:
  - $\pm S \times B^{\pm E}$
- Três campos:
  - Sinal: mais (0) ou menos (1).
  - Significando S.
  - Expoente E.
- A base B é implicita e não precisa ser armazenada.

Formato típico de ponto flutuante de 32 bits



(a) Formato

$1.1010001 \times 2^{10100} = 0\ 10010011\ 10100010000000000000000 = 1.6328125 \times 2^{20}$

$-1.1010001 \times 2^{10100} = 1\ 10010011\ 10100010000000000000000 = -1.6328125 \times 2^{20}$

$1.1010001 \times 2^{-10100} = 0\ 01101011\ 10100010000000000000000 = 1.6328125 \times 2^{-2}$

$-1.1010001 \times 2^{-10100} = 1\ 01101011\ 10100010000000000000000 = -1.6328125 \times 2^{-2}$

(b) Exemplos

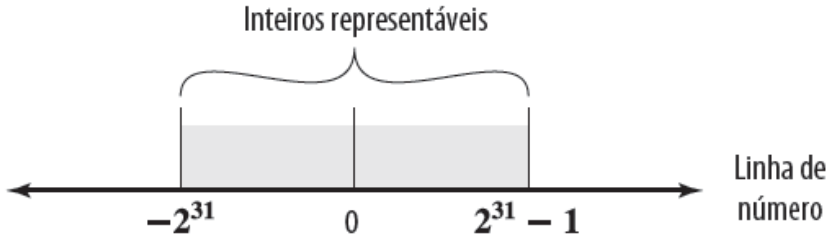
Ponto Flutuante

- Mantissa é armazenada em complemento a dois.
- Expoente usa a chamada representação polarizada.
  - Um valor fixo, chamado de polarização, é subtraído do campo para obter o valor verdadeiro do expoente. Normalmente é igual a $2^{k-1}-1$.
  - Exemplo:
    - k = 8 (bits) → Polarização = 127 (2^7-1)
    - Intervalo de valor puro 0-255.
    - Subtraia 127 para obter valor correto.
    - Intervalo de -127 a +128.

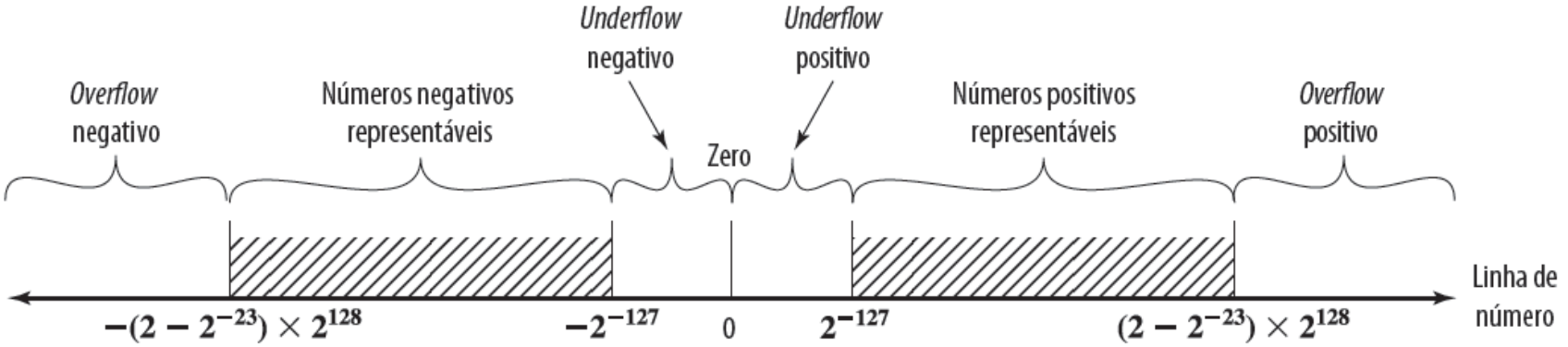
Ponto Flutuante: Normalização

- Números de PF geralmente são normalizados, ou seja, expoente é ajustado de modo que bit inicial (MSB) do significando seja 1.
- Por ser sempre 1, não é preciso armazená-lo.
- Ponto flutuante normalizado:
  - $\pm 1.bbb \ldots b \times 2^{\pm E}$
  - Campo de 23 bits é usado para armazenar um significando de 24 bits com um valor no intervalo meio aberto [1,2)

Ponto Flutuante: Intervalos



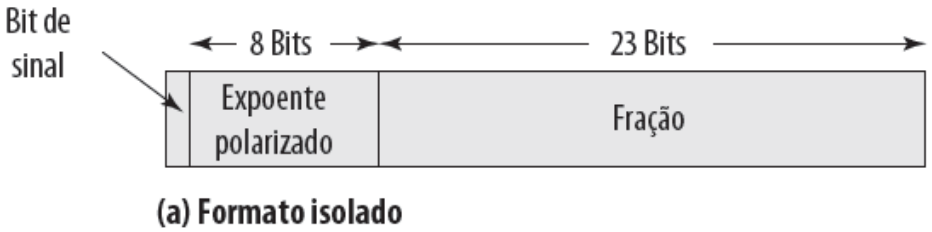
(a) Inteiros de complemento de dois



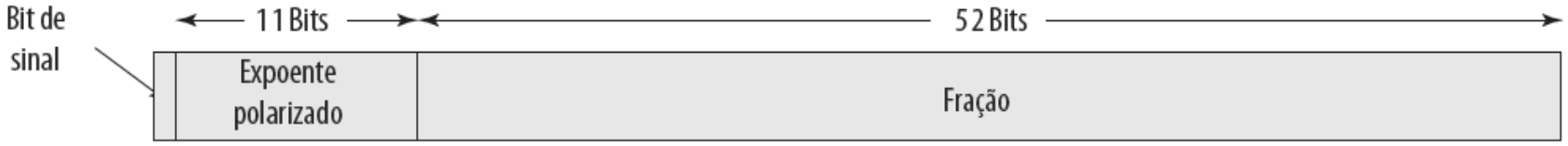
(b) Números de ponto flutuante

Padrão IEEE para representação binária de ponto flutuante (IEEE 754)

Formato:



(a) Formato isolado



(b) Formato duplo

Convertendo...
Decimal ↔ IEEE 754 (32 bits)

$$-26{,}5 \xrightarrow{binário} -11010{,}1 \times 2^0 \xrightarrow{normalização} -1{,}\underline{10101} \times 2^{\underline{4}}$$

$4 + 127 = 131$

1.10000011.10101000000000000000000

$$-7{,}1875 \xrightarrow{binário} -111{,}0011 \times 2^0 \xrightarrow{normalização} -1{,}\underline{110011} \times 2^{\underline{2}}$$

$2 + 127 = 129$

1.10000001.11001100000000000000000

$$+20{,}8 \xrightarrow{binário} +10100{,}[1100] \times 2^0 \xrightarrow{normalização} +1{,}0[1001] \times 2^{\underline{4}}$$

$4 + 127 = 131$

0.10000011.01001100110011001100110

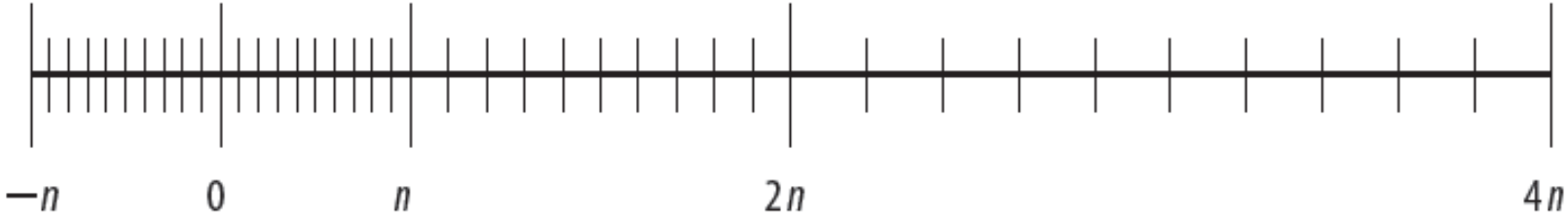
Parâmetros de formato IEEE 754

Parâmetro	Formato			
	Isolado	Estendido isolado	Duplo	Estendido duplo
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
Exponente mínimo	–126	≤ –1022	–1022	≤ –16382
Intervalo numérico (base 10)	10^{-38}, 10^{+38}	Não especificado	10^{-308}, 10^{+308}	Não especificado
Tamanho do significando (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 incluso o bit implícito.

Ponto Flutuante: Densidade

- Números Inteiros (ponto fixo) = Espaçamento linear (uniforme) ao longo de todo intervalo.
- Números de Ponto Flutuante = Espaçamento não uniforme.
  - Mais próximo da origem (zero) = maior densidade (ou seja, os números possíveis de serem representados ficam mais próximos).
  - Mais afastado da origem = menor densidade.



Ponto Flutuante: intervalo ou precisão?

- Ponto flutuante de 32 bits:
  - 1 bit de sinal
  - 8 bits de expoente
  - 23 bits de significando
- Se aumentarmos o número de bits no expoente, expandimos o intervalo de números representáveis.
- Mas como apenas um número fixo de valores diferentes podem ser representados, reduzimos a densidade desses números e, portanto, a precisão.
- O único modo de aumentar esses dois fatores é usar mais bits.
  - Comumente usados:
    - 32 bits (precisão simples)
    - 64 bits (precisão dupla)

Aritmética de Ponto Flutuante

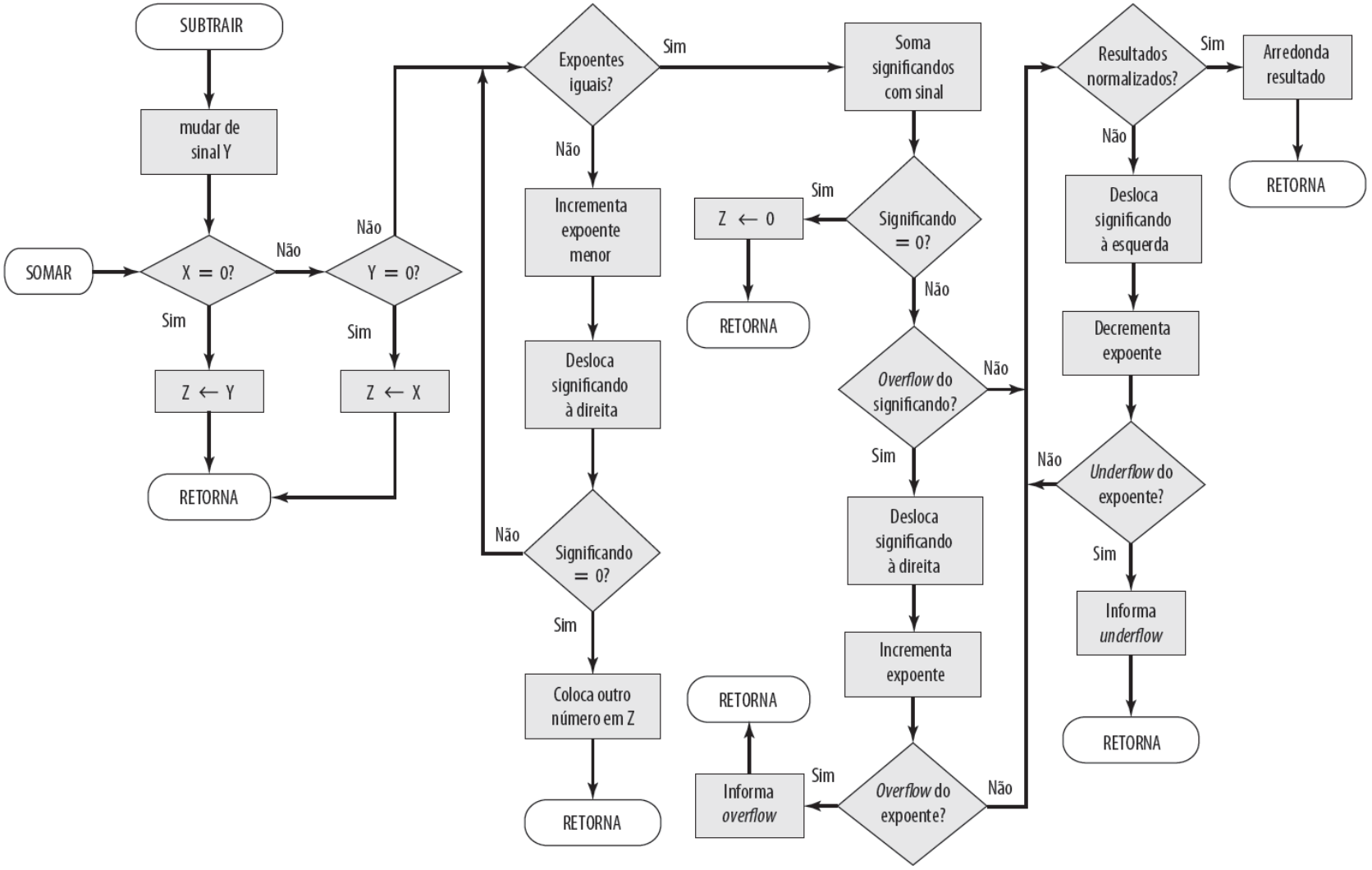
Números de ponto flutuante	Operações aritméticas
$X = X_S \times B^{X_E}$ $Y = Y_S \times B^{Y_E}$	$X + Y = (X_S \times B^{X_E - Y_E} + Y_S) \times B^{Y_E}$ $X - Y = (X_S \times B^{X_E - Y_E} - Y_S) \times B^{Y_E}$ } $X_E \leq Y_E$ $X \times Y = (X_S \times Y_S) \times B^{X_E + Y_E}$ $\frac{X}{Y} = \left(\frac{X_S}{Y_S}\right) \times B^{X_E - Y_E}$

- Anomalias possíveis:
  - ***Overflow* de expoente**: expoente positivo maior que o máximo possível.
  - ***Underflow* de expoente**: expoente negativo menor que o mínimo possível (ex: -200 é menor que -127). Ou seja, o número é muito pequeno para ser representado.
  - ***Underflow* de significando**: no alinhamento dos significandos, os dígitos podem sair pela extremidade direita do significando.
  - ***Overflow* de significando**: adição de dois significandos com o mesmo sinal pode resultar em um carry pelo bit mais significativo.

Adição e Subtração de PF

- Passo-a-passo:
  - 0. É uma operação de subtração?
    - Troque o sinal do subtraendo.
  - 1. Verificação de zero.
  - 2. Alinhamento do significando.
  - 3. Adição.
  - 4. Normalização

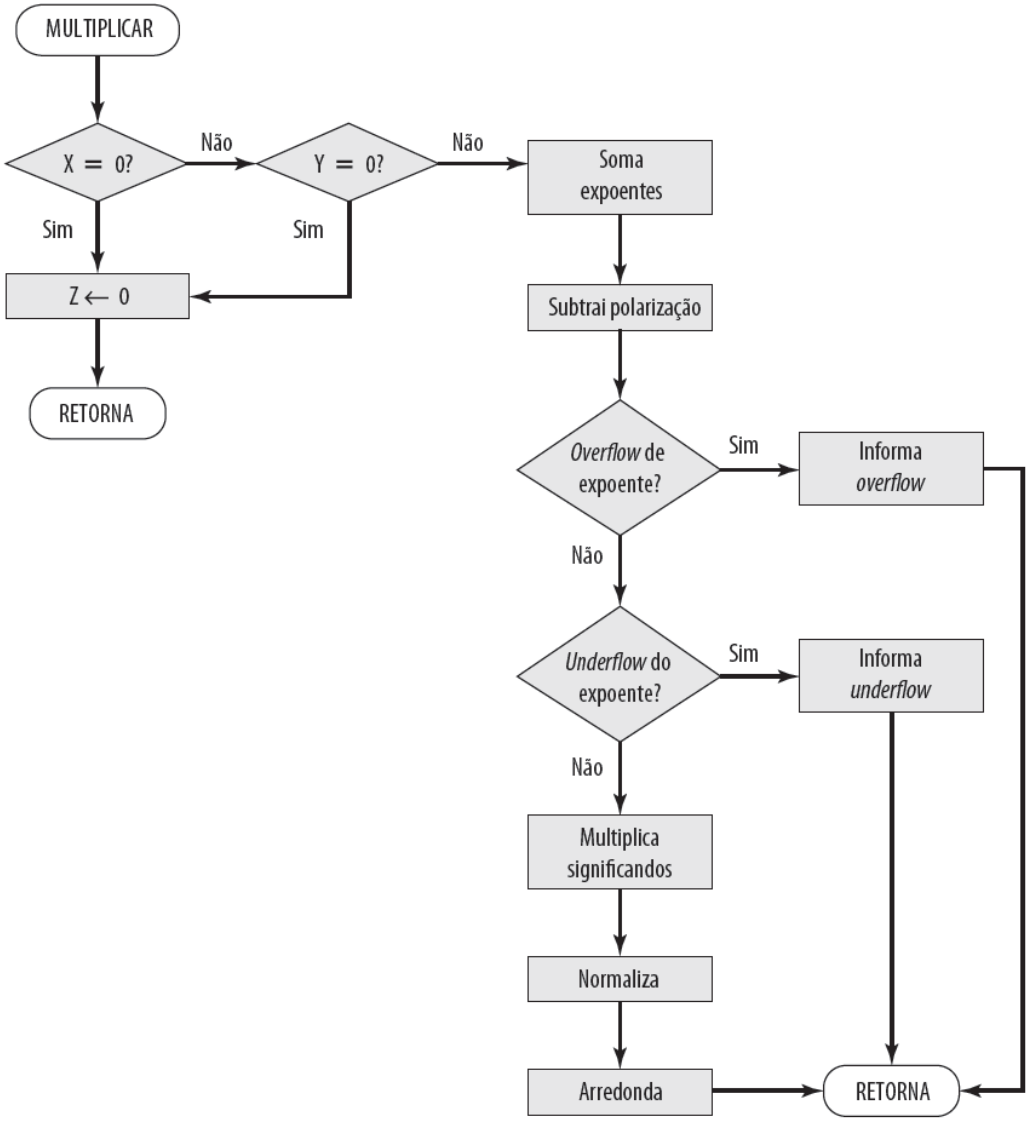
Adição e Subtração de PF



Multiplicação e Divisão de PF

- Passo-a-passo
  - Verifique zero.
  - Soma/subtraia expoentes .
  - Multiplique/divida significandos (observe sinal).
  - Normalize.
  - Arredonde.
  - Todos os resultados intermediários devem ser em armazenamento de tamanho duplo.

Multiplicação de PF



Divisão de PF