Arquitetura e Organização de Computadores

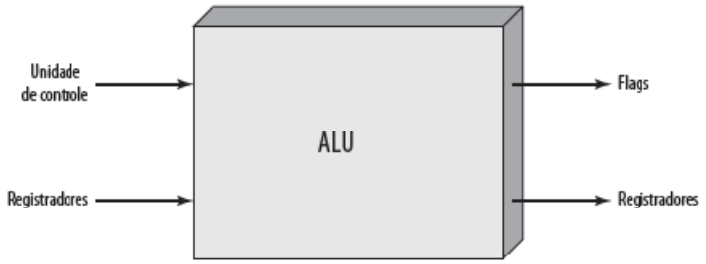
Capítulo 9

Aritmética do computador

Unidade aritmética e lógica

- 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).
- Pode estar em chip de FPU separado (486DX +).

Entradas e saídas da ALU



Representação de inteiros

- Só tem 0 & 1 para representar tudo.
- Números positivos armazenados em binário.
  —P.e., 41=00101001.
- Sem sinal de menos.
- Sem ponto.
- Sinal-magnitude.
- Complemento a dois.

Sinal-magnitude (não usada)

- Bit mais à esquerda é bit de sinal.
- 0 significa positivo.
- 1 significa negativo.
- +18 = 00010010.
- -18 = 10010010.
- Problemas:
  - —Precisa considerar sinal e magnitude na aritmética.
  - —Duas representações de zero (+0 e -0).

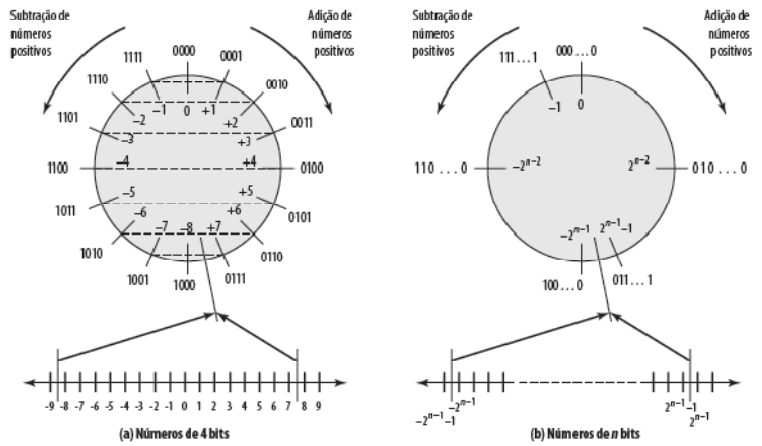
Complemento a dois (usada)

- +3 = 00000011
- +2 = 00000010
- +1 = 00000001
- +0 = 00000000
- -1 = 11111111 (**inverte bit a bit e soma 1**)
- -2 = 11111110
- -3 = 11111101

Benefícios

- 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 11111101

Representação geométrica dos inteiros de complemento a dois



Negação especial – caso 1

- 0 = 00000000
- Not bit a bit 11111111
- Some 1 ao LSB +1
- Resultado 1 00000000
- Estouro ignorado, portanto:
- - 0 = 0 √

Negação especial – caso 2

- -128 = 10000000
- Not bit a bit 01111111
- Some 1 ao LSB +1
- Resultado 10000000
- Portanto:
- -(-128) = -128 X
- 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}

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).

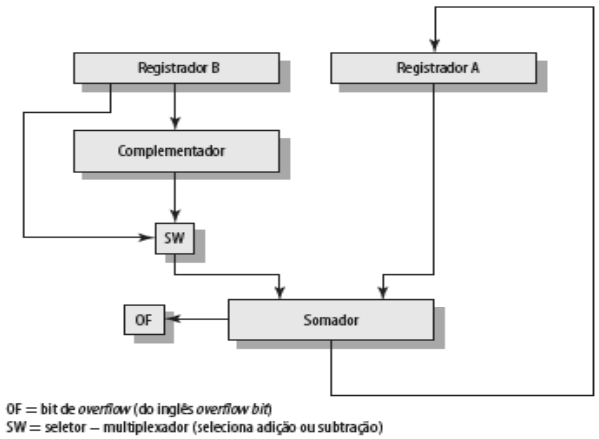
Adição e subtração

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

- Pegue o complemento a dois do subtraendo e some ao minuendo.
  —Ou seja, a - b = a + (-b).

- Assim, **só se precisa de circuitos de adição e complemento**.

Hardware para adição e subtração



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

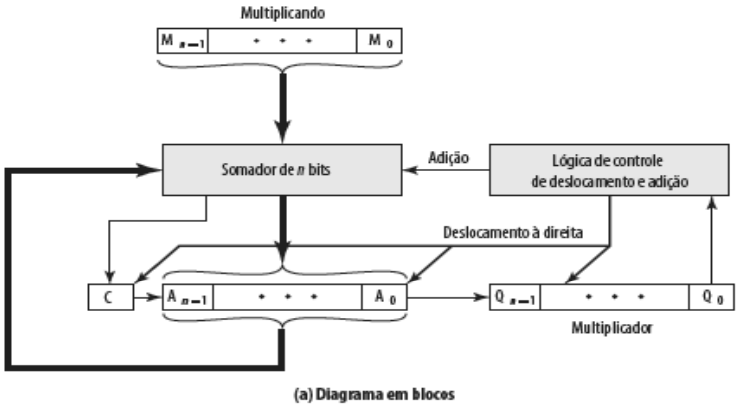
Multiplicação

- Complexa.
- Calcule produto parcial para cada dígito.
- Cuidado com o valor da casa (coluna).
- Some produtos parciais.

Exemplo de multiplicação

- 1011 Multiplicando (11 dec)
- x 1101 Multiplicador (13 dec)
- 1011 Produtos parciais
- 0000 Nota: Se bit multiplicador for 1, copia.
- 1011 Multiplicando (valor da casa)
- 1011 Caso contrário, zero.
- 10001111 Produto (143 dec)
- Nota: precisa de resultado com tamanho duplo.

Multiplicação binária sem sinal

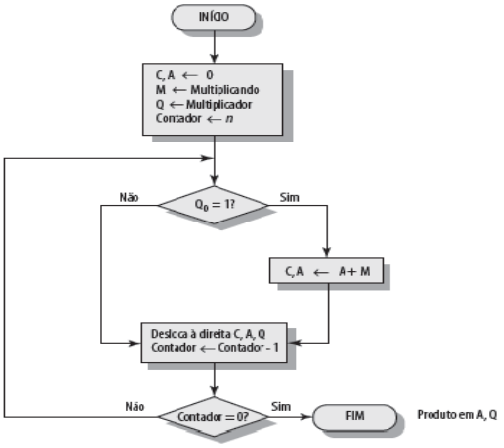


(a) Diagrama em blocos

Execução do exemplo

C	A	Q	M		
0	0000	1101	1011	Valores iniciais	
0	1011	1101	1011	Adição	Primeiro ciclo
0	0101	1110	1011	Desl.	
0	0010	1111	1011	Desl.	Segundo ciclo
0	1101	1111	1011	Adição	Terceiro ciclo
0	0110	1111	1011	Desl.	
1	0001	1111	1011	Adição	Quarto ciclo
0	1000	1111	1011	Desl.	

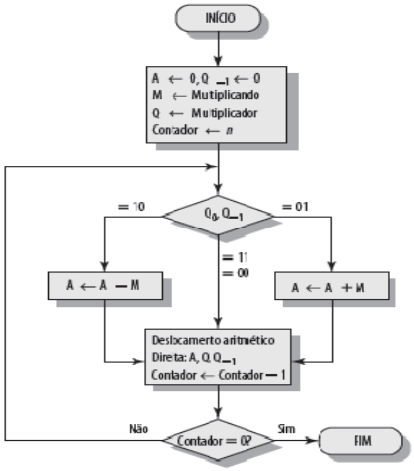
Fluxograma para a multiplicação binária sem sinal



Multiplicando números negativos

- Isso não funciona!
- Solução 1:
  - —Converta para positivo, se for preciso.
  - —Multiplique como antes.
  - —Se sinais diferentes, negue a resposta.
- Solução 2:
  - —Algoritmo de Booth.

Algoritmo de Booth



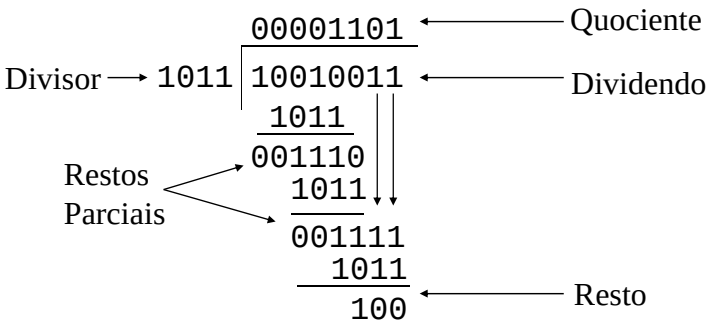
Exemplo do algoritmo de Booth

A	Q	Q_{-1}	M		
0000	0011	0	0111	Valores iniciais	
1001	0011	0	0111	A ← A − M	Primeiro ciclo
1100	1001	1	0111	Deslocamento	
1110	0100	1	0111	Deslocamento	Segundo ciclo
0101	0100	1	0111	A ← A + M	Terceiro ciclo
0010	1010	0	0111	Deslocamento	
0001	0101	0	0111	Deslocamento	Quarto ciclo

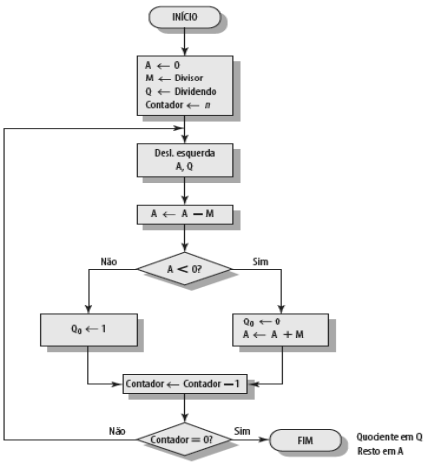
Divisão

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

Divisão de inteiros binários sem sinal



Fluxograma para divisão binária sem sinal



Números reais

- Números com frações.
- Poderia ser feito em binário puro.
  - —$1001.1010 = 2^4 + 2^0 + 2^{-1} + 2^{-3} = 9{,}625$
- Onde está o ponto binário?
- Fixo?
  - —Muito limitado.
- Móvel?
  - —Como você mostra onde ele está?

Ponto flutuante

(a) Formato

- +/- significando x $2^{\text{exponente}}$
- Nome impróprio
- Ponto é realmente fixo entre bit de sinal e corpo da mantissa
- Expoente indica valor da casa (posição do ponto)

Exemplos de ponto flutuante

(a) Formato

$1.1010001 \times 2^{10100} = 0\ 10010011\ 10100010000000000000000 = 1.6328125 \times 2^{20}$
$-1.1010001 \times 2^{10100} = 1\ 10010011\ 10100010000000000000000 = -1.6328125 \times 2^{20}$
$1.1010001 \times 2^{-10100} = 0\ 01101011\ 10100010000000000000000 = 1.6328125 \times 2^{-20}$
$-1.1010001 \times 2^{-10100} = 1\ 01101011\ 10100010000000000000000 = -1.6328125 \times 2^{-20}$

(b) Exemplos

Sinais para ponto flutuante

- Mantissa é armazenada em complemento a dois.
- Expoente está em notação de excesso ou viesado.
  - —P.e., excesso (viés) 128 significa campo de expoente com 8 bits.
  - —Intervalo de valor puro 0-255.
  - —Subtraia 128 para obter valor correto.
  - —Intervalo de -128 a +127.

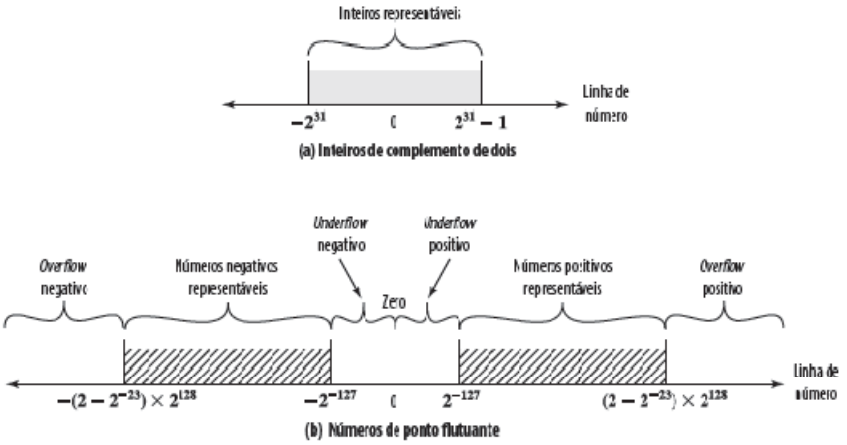
Normalização

- Números de PF geralmente são normalizados, ou seja, expoente é ajustado de modo que bit inicial (MSB) da mantissa seja 1.
- Por ser sempre 1, não é preciso armazená-lo.
- (c.f. notação científica, onde os números são normalizados para um único dígito antes do ponto decimal, p.e., $3{,}123 \times 10^3$)

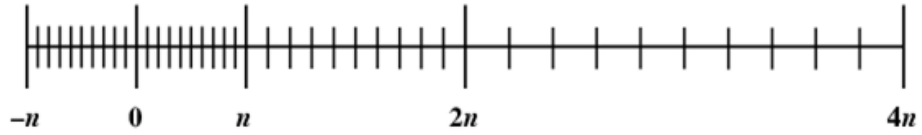
Intervalos de PF

- Para um número de 32 bits:
  - —Expoente de 8 bits.
  - —+/- $2^{256} \approx 1{,}5 \times 10^{77}$
- Precisão:
  - —O efeito de alterar LSB da mantissa.
  - —Mantissa de 23 bits $2^{-23} \approx 1{,}2 \times 10^{-7}$.
  - —Cerca de 6 casas decimais.

Números representáveis



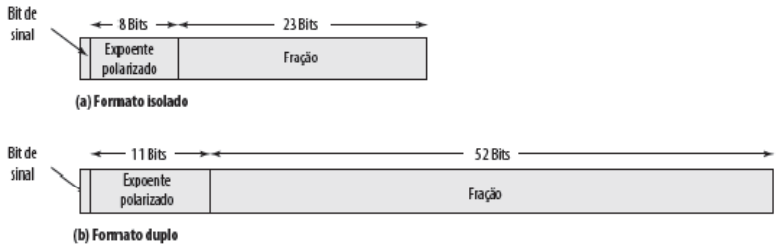
Densidade dos números de ponto flutuante



IEEE 754

- Padrão para armazenamento de ponto flutuante.
- Padrões de 32 e 64 bits.
- Expoente de 8 e 11 bits, respectivamente.
- Formatos estendidos (mantissa e expoente) para resultados intermediários.

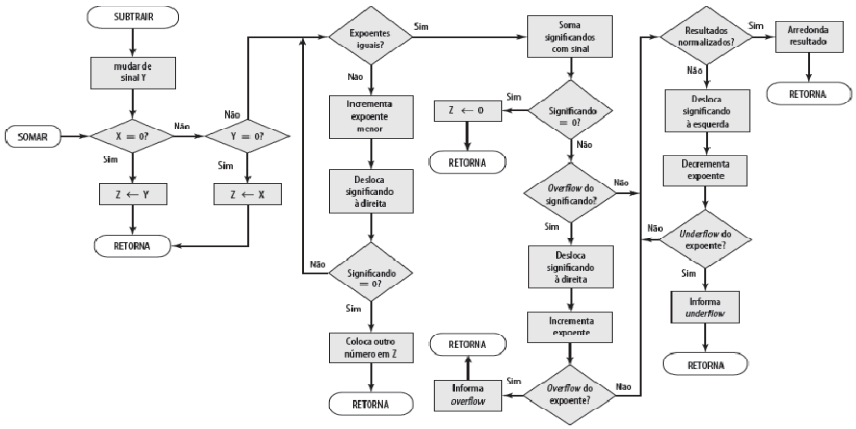
Formatos IEEE 754



Aritmética de ponto flutuante (+/-)

- Verifique zero.
- Alinhe significandos (ajustando expoentes).
- Soma ou subtraia significandos.
- Normalize resultado.

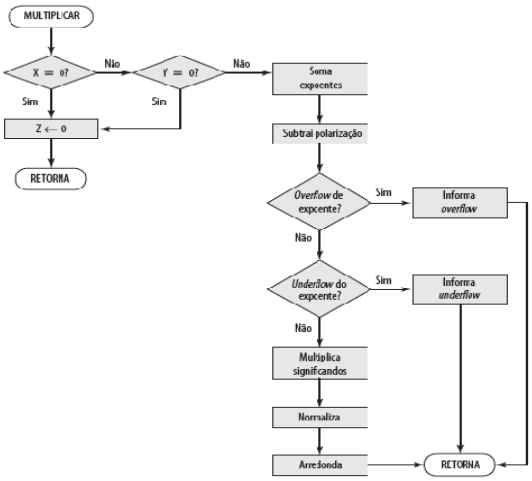
Fluxograma da adição e subtração de ponto flutuante



Aritmética de ponto flutuante (x/÷)

- 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 ponto flutuante



Divisão de ponto flutuante