Unidade de Ponto Flutuante - FPU

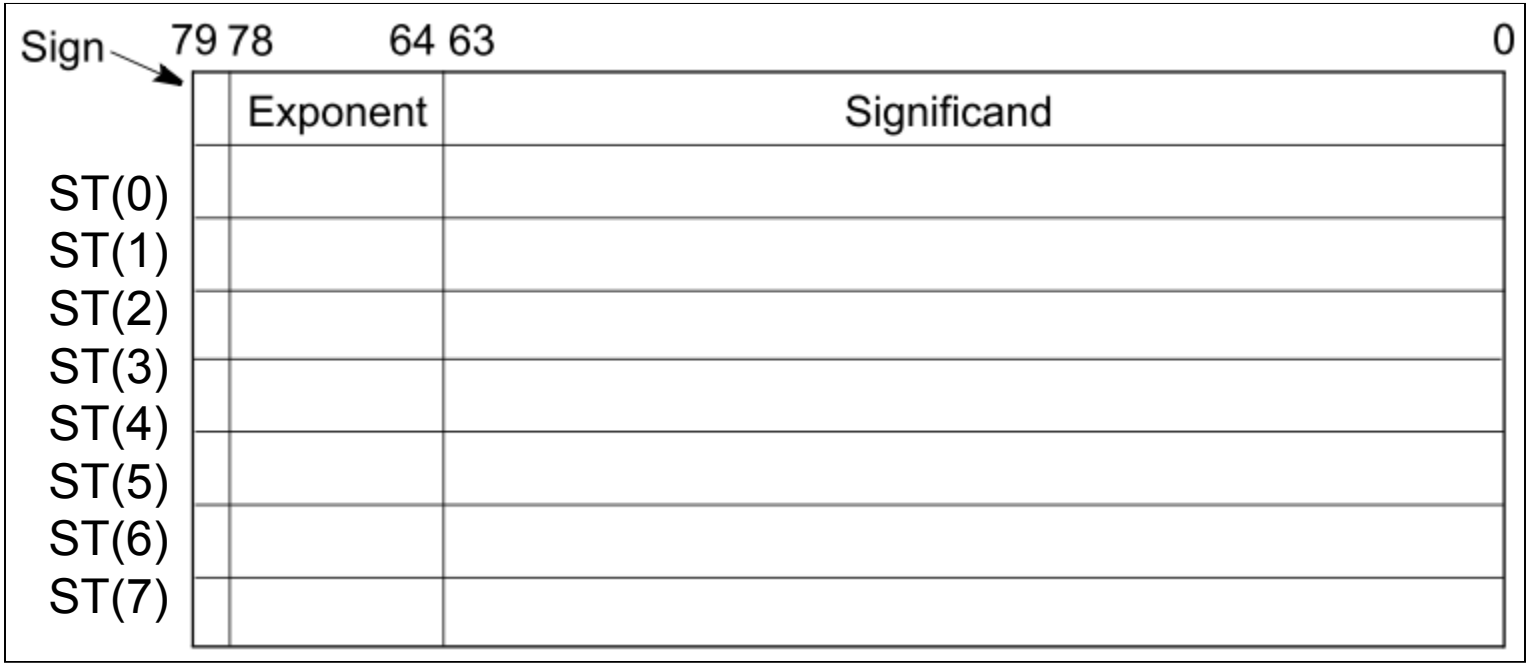
Baseado nos slides do Prof. Eduardo Tavares

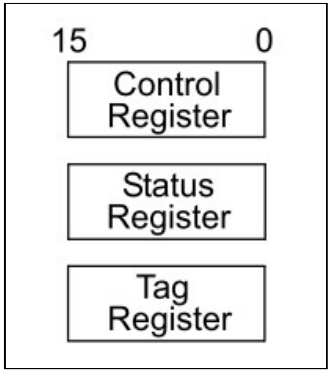
eagt@cin.ufpe.br

FPU

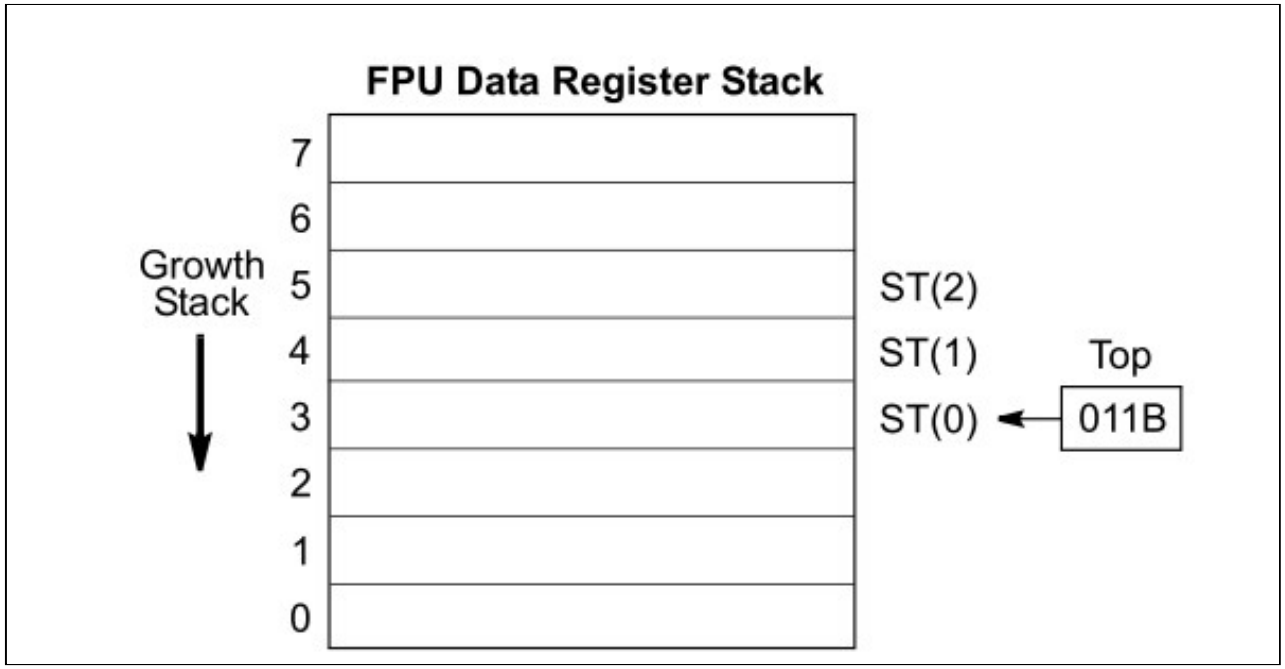
- Os processadores atuais da família x86 possuem uma FPU (Unidade de Ponto Flutuante) dedicada para efetuar operações de ponto flutuante
  - No hardware, algumas arquiteturas podem compartilhar elementos entre a FPU e a CPU (ex.: Pentium)
- Números em ponto flutuante adotam o formato IEEE-754
- A FPU possui 8 registradores de dados de 80 bits cada para armazenar números em ponto flutuante

Registradores





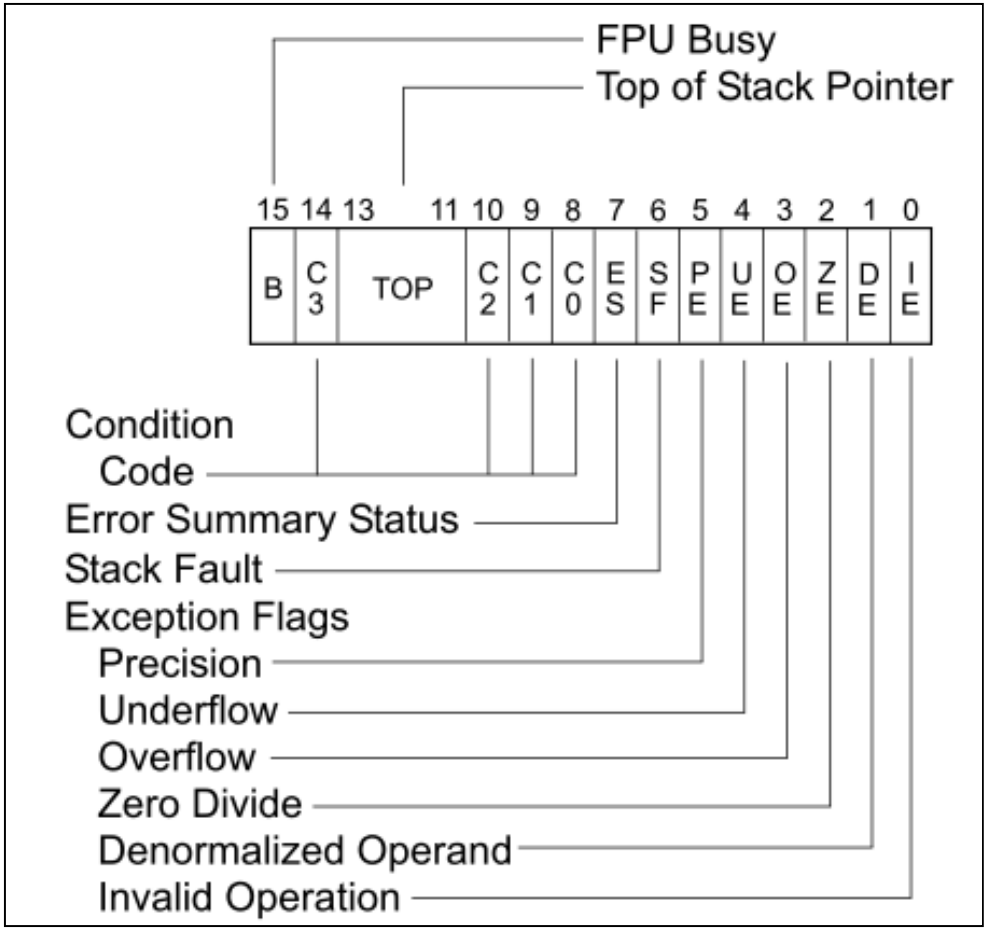
Pilha


FPU Data Register Stack
Growth Stack
7
6
5
4
3
2
1
0
ST(2)
ST(1)
ST(0)
Top
011B

Registrador de Status

FPU Flag	CPU Flag
C0	CF
C2	PF
C3	ZF

C1 = 0 - (stack) underflow
1 - (stack) overflow

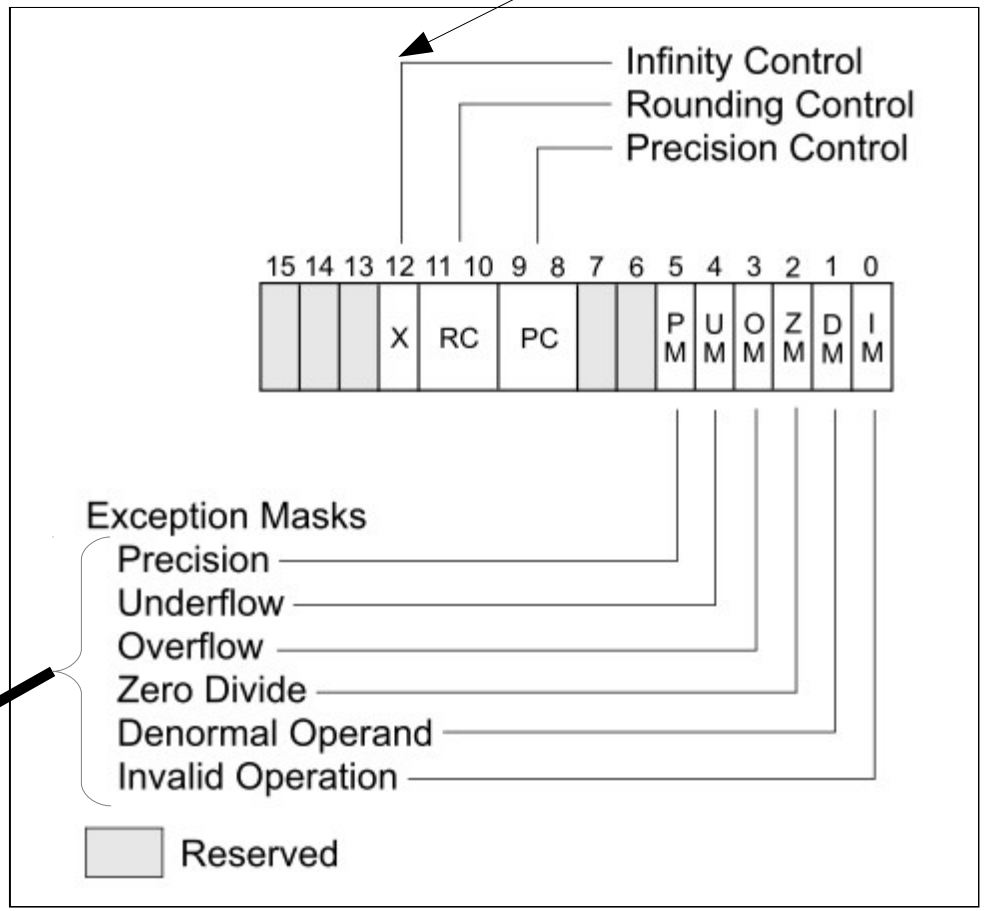


Registrador de Controle

Razões históricas, não mais usado

RC
00 – Arredonda para o mais próximo
01 – Arredonda para baixo
10 – Arredonda para cima
11 – Truncamento

PC (Mantissa)
00 – 24 bits (precisão simples)
01 – Não é adotado
10 – 52 bits (precisão dupla)
11 – 64 bits (precisão extendida)

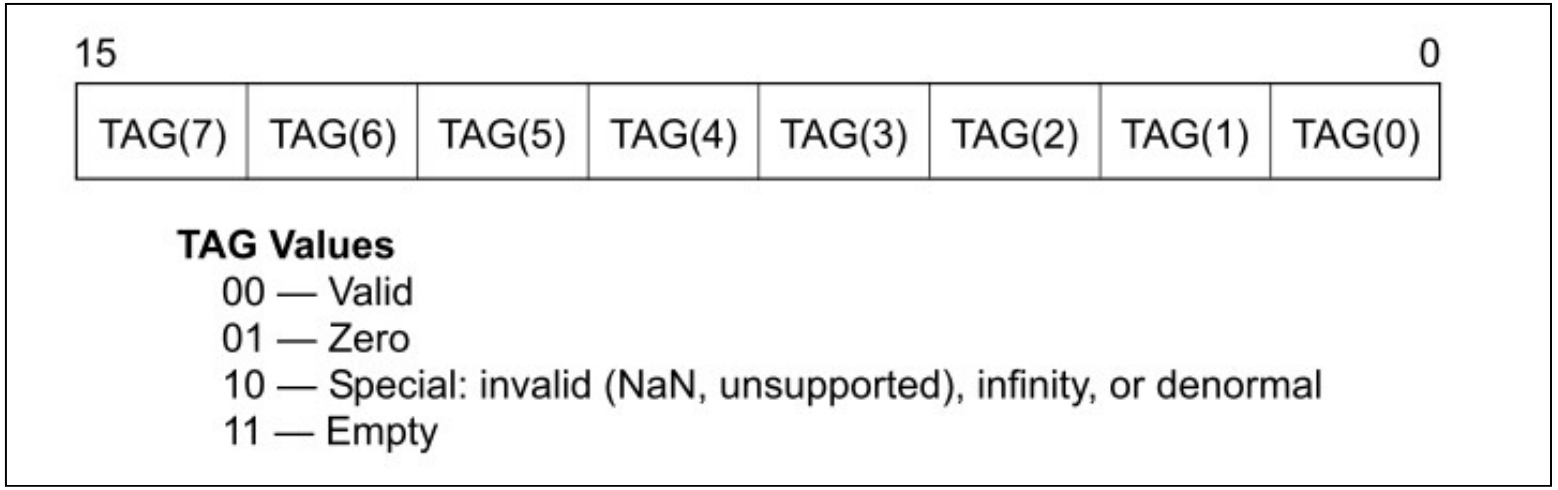


1 – Mascara
0 – Não mascara

Mascara a exceção correspondente

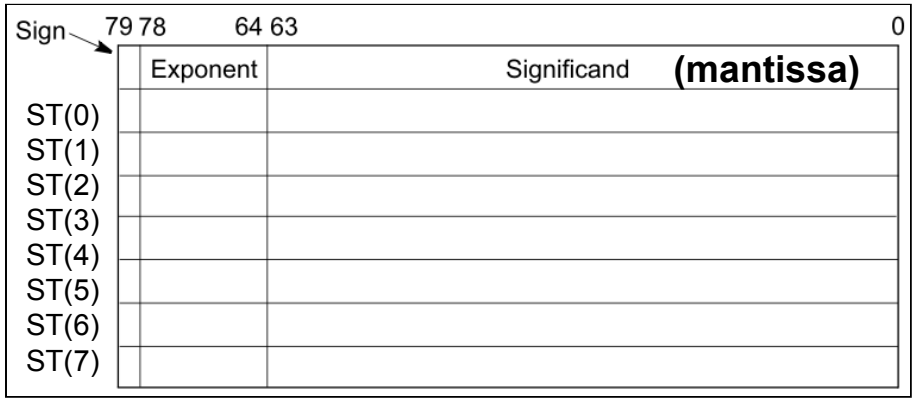
Padrão: 037FH – Todas exceções mascaradas, arred mais próximo, 64bit prec

Registrador de Tag



2 bits para cada registrador da FPU, indica o tipo de valor contido no registrador

Formato IEEE 754



$$N = (-1)^{S} \times 2^{E-127} \times 1.M$$

S-> sinal
E = Expoente
M -> parte fracionaria (mantissa)

Tipo de Dados

Word integer: Magnitude = 15 bits, Sinal =1 bit
Short (doubleword) integer: Magnitude = 31 bits, sinal =1 bit
Long (quadword) integer: Magnitude = 63 bits, sinal =1 bit
Packed BCD: Magnitude = 72 bits, sinal =1 bit (de um byte um único bit é usado)
Short real: Mantissa = 23 bits, Expoente = 8 bits, Sinal = 1 bit
Long real: Mantissa = 52 bits, Expoente = 11 bits, Sinal = 1 bit
Extended real: Mantissa = 64 bits, Expoente = 15 bits, Sinal = 1 bit

Tipo de Dados

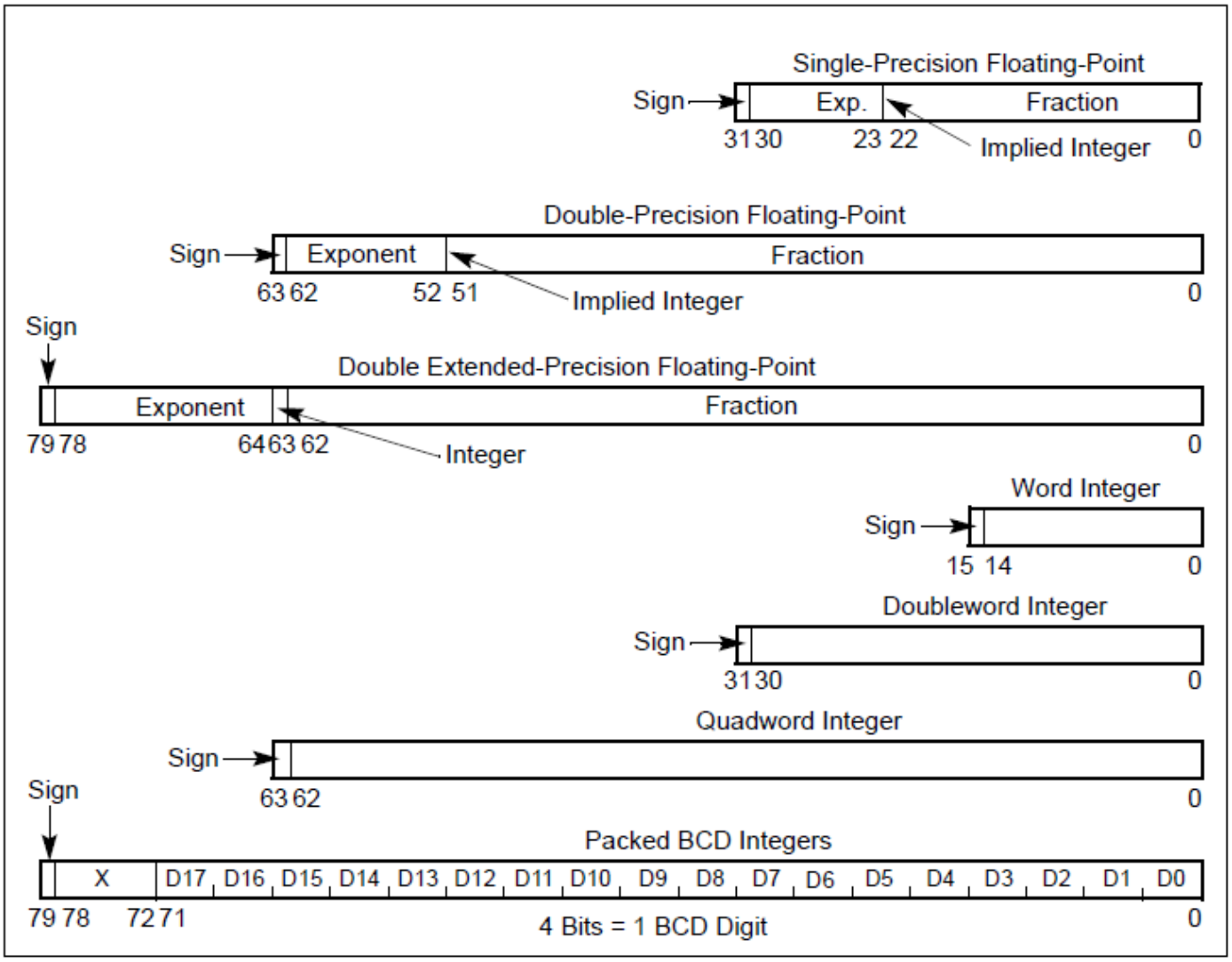


Figure 8-13. x87 FPU Data Type Formats

Tipo de Dados

Exemplo: Representação do Formato Short Real

- N = -1S×2E' ×1.M, onde E'=E-127. Portanto E= E'+127.

Obtenha o número armezando em memória correspondente ao número 209,8125 no formato Short Real do Pentium.

1. Converter para binário: 11010001,11012
2. Normalizar: 1,101000111012 (consistiu em deslocar, neste caso, a vírgula decimal 7 posições à esquerda (+7).
3. Como 23 bits são utilizados para representar a mantissa (no Short Real), preenche-se os demais bits com zero (0): 1,10100011101000000000002.
4. O bit mais significativo não é representado, pois sempre é 1: M=10100011101000000000002

Tipo de Dados

Exemplo: Representação do Formato Short Real

- N = -1S×2E' ×1.M, onde E'=E-127. Portanto E= E'+127.

Obtenha o número armezando em memória correspondente ao número 209,8125 no formato Short Real do Pentium.

5. Soma-se 127 ao expoente obtido na normalização (E'): E=127+ 7=134 = 10000110b.
6. Calcule -1S = (-1)0 = 1 (S=0 pois o número é positivo).
7. Compõem-se as partes e gera-se o número:
   01000011010100011101000000000000b = 43 51 D0 00H
   S E M

Obs.:para os demais tipos de dados o processo é similar.

Programação utlizando a FPU

Utilizando Operandos Implícitos (Formato pilha)

- A forma clássica de endereçamento acessa os registradores via o formato “pilha”.
- Os operandos não são especificados. O (primeiro) operando é o registrador ST(0) – fonte – e o segundo operando (se houver) é o registrador ST(1).
- O resultado da operação é armazenado no operando destino e o operando fonte é desempilhado. Portanto, o resultado fica no topo da pilha de registradores.

Obs: Pode-se acessar os registradores individualmente

Mnemônicos das instruções da FPU começam com F

Programação utilizando a FPU

Utilizando Operando em Memória

- Os elementos são carregados no topo da pilha ou desempilhados na memória.

Utilizando Operandos na forma de Registradores.

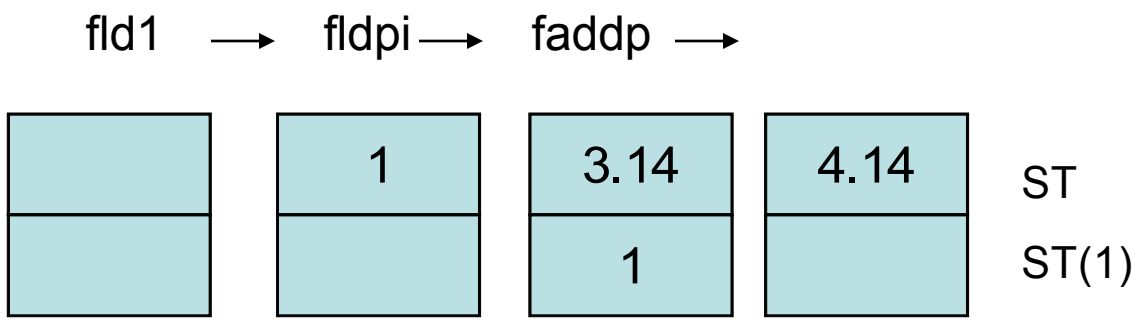
- Nesta forma os operandos são especificados através dos seus nomes.
- O primeiro operando é o destino.
- O segundo operando é a fonte.

Programação utilizando a FPU

```
SEGMENT .text
global _start
_start:

fld1
fldpi
faddp st1

mov eax,1
mov ebx,0
int 80h
```



Movimentação

Instrução FLD

- Carrega um número de 32, 64 ou 80 bits na pilha.
- De fato, FLD, inicialmente decrementa o TOS (bits 11,12 e 13 do registrador de *status*) e posteriormente armazena a informação no registrador apontado por TOS.
- Esta instrução converte o número do formato de memória para o formato de ponto flutuante.
  - fld mem
  - fld st(i) ; No NASM: fld sti (sem parêntese)

Movimentação

```
SEGMENT .data
dado1 dq 3.0
dado2 dq 4.0
SEGMENT .text
global _start
_start:

fld qword[dado1]
fld qword[dado2]
fld st1

mov eax,1
mov ebx,0
int 80h
```

Movimentação

Instrução FST e FSTP

- Copia valor do topo da pilha em um registrador ST(i) ou posição de memória (32, 64 ou 80 bits).

- A instrução FSTP desempilha o valor do topo da pilha.

```
            d1   dq   3.0
                 ...
            d2        resq  1
        ...
    fld  qword[d1]
    fst qword[d2]
```

Movimentação

Instrução FXCH

- Permuta valores entre registradores de ponto flutuante. Tem-se duas formas: um operando e dois operandos
  - fxch st(i) ; permuta valores entre st e st(i).

Instrução FILD

- Carrega um operando inteiro de 16, 32, 64 bits (complemento de dois) no ST.
  - fild mem

Movimentação - Inteiro

Instrução FIST

- Copia o valor do registrado ST para região de memória (no formato inteiro).O destino pode ser de 16, 32 ou 64 bits.
  - fist mem

Instrução FISTP

- Copia o valor do registrado ST para região de memória (no formato inteiro) e desempilha o valor de ST. O destino pode ser de 16, 32 ou 64 bits.
  - fist mem

Programação utilizando a FPU

```
SEGMENT .data
m1    dq   1.0
m2    dq   2.0
m3    dq   3.0
m4     dq   4.0

SEGMENT .text
global _start
_start:

fld    QWORD [m4]
fld    QWORD [m3]
fld    QWORD [m1]
fld    st2
fst    QWORD [m2]
fxch   st2
fstp   QWORD [m1]
```

```
mov eax,1
mov ebx,0
int 80h
```

Movimentação - BCD

Instrução FBLD

- Carrega informação de 80 bits (formato BCD) da memória para o registrador ST
  - fbld mem

Instrução FBSTP

- Armazena informação contida no ST na posição de memória (o operando destino deve ser de 80 bits). A informação armazenada na memória estará no formato BCD e desempilha o valor de ST.
  - fbstp mem

Carregamento

Instrução FLDZ

- Carrega informação 0 no ST

Instrução FLD1

- Carrega informação 1 no ST

Instrução FLDPI

- Carrega informação Π no ST

Instrução FLDL2E

- Carrega log2e no ST

Instrução FLDL2T

- Carrega informação log210 no ST

Instrução FLDLG2

- Carrega informação log102 no ST

Instrução FLDLN2

- Carrega informação loge2 no ST

Adição

Formas:

- FADD src (ST0 = ST0 + src)
- FADD ST(i),ST
- FADD ST, ST(i)
- FADD mem
  - Resultado é armazenado em ST.
- FIADD mem
  - Soma inteira. Resultado armazenado em ST.
- FADDP ST(i), ST
  - Soma ST(i) e ST, armazena o resultado em ST(i).

Programação utilizando a FPU

```
 1:  /*************************************************************
 2:   * This program reads SIZE values into an array and calls
 3:   * an assembly language program to compute the array sum.
 4:   * The assembly program is in the file "arrayfsuma.asm".
 5:   *************************************************************/
 6:  #include        <stdio.h>
 7:
 8:  #define  SIZE  10
 9:
10:  int main(void)
11:  {
12:      double     value[SIZE];
13:      int        i;
14:      extern double array_fsum(double*, int);
15:
16:      printf("Input %d array values:\n", SIZE);
17:      for (i = 0; i < SIZE; i++)
18:          scanf("%lf",&value[i]);
19:
20:      printf("Array sum = %lf\n", array_fsum(value,SIZE));
21:
22:      return 0;
23:  }
```

Programação utilizando a FPU

```
 1:  ;-------------------------------------------------------
 2:  ; This procedure receives an array pointer and its size
 3:  ; via the stack. It computes the array sum and returns
 4:  ; it via ST0.
 5:  ;-------------------------------------------------------
 6:  segment .text
 7:  global  array_fsum
 8:
 9:  array_fsum:
10:      enter   0,0
11:      mov     EDX,[EBP+8]       ; copy array pointer
12:      mov     ECX,[EBP+12]      ; copy array size
13:      fldz                      ; ST0 = 0 (sum is in ST0)
14:  add_loop:
15:      jecxz   done
16:      dec     ECX               ; update the array index
17:      fadd    qword[EDX+ECX*8]  ; ST0 = ST0 + arrary_element
18:      jmp     add_loop
19:  done:
20:      leave
21:      ret
```

Subtração

- FSUB ST(i),ST
- FSUB ST, ST(i)
- FSUB mem (ST – [mem])
  - Resultado é armazenado em ST.
- FISUB mem (ST – [mem])
  - Subtração inteira. Resultado armazenado em ST.
- FSUBP ST(i), ST ou FSUBP ST(i)
  - Subtrai ST de ST(i) (ST(i)-ST), armazena o resultado em ST(i) e desempilha ST.

Subtração

Formas:

- FSUBR ST(i),ST (ST – ST(i))
- FSUBR ST, ST(i) (ST(i) – ST)
- FSUBR mem
  - Resultado é armazenado em ST.
- FISUBR mem ([mem] - ST)
  - Subtração inteira. Resultado armazenado em ST.
- FSUBRP ST(i), ST ou FSUBRP ST(i)
  - Subtrai ST(i) de ST (ST-ST(i)), armazena o resultado em ST(i) e desempilha ST.

Multiplicação

- FMUL ST(i),ST
- FMUL ST, ST(i)
- FMUL mem
  - Resultado é armazenado em ST.
- FIMUL mem (ST * [mem])
  - Multiplicação inteira. Resultado armazenado em ST.
- FMULP ST(i), ST ou FMULP ST(i), ST
  - Multiplica ST e ST(i) (ST(i)*ST), armazena o resultado em ST(i) e desempilha ST.

Divisão

- FDIV src
  - ST=ST/src
- FDIV dest,src
  - dest=dest/src
- FDIV ST, ST(i)
- FDIV mem
  - Resultado é armazenado em ST.
- FIDIV mem (ST / [mem])
  - Divisão inteira. Resultado armazenado em ST.
- FDIVP dest
  - dest=dest/ST
- FDIVP ST(i), ST
  - Divisão ST(i) por ST (ST(i)/ST), armazena o resultado em ST(i) e desempilha ST.

Divisão

- FDIVR ST(i),ST
- FDIVR ST, ST(i)
- FDIVR mem
  - Resultado é armazenado em ST.
- FIDIVR mem ([mem]/ST)
  - Divisão inteira. Resultado armazenado em ST.
- FDIVRP ST(i), ST OU FDIVRP ST(i)
  - Divisão ST por ST(i) (ST/ST(i)), armazena o resultado em ST(i) e desempilha ST.

Outras instruções aritméticas

FSQRT

- Calcula a raiz quadrada do valor do registrador apontado por TOS e armazena o resultado em ST.

FSCALE

- Armazena no topo da pilha (ST) o valor $ST \times 2ST(1)$

FRNDINT

- Arredonda o valor de ST para inteiro, segundo especificado no registrador de controle.

FABS

- Torna o sinal de ST como positivo.

FCHS

- Inverte o sinal de ST.

Exemplo Raiz Quadrada

```
SEGMENT .data
m1    dq   1.0
m2    dq   2.0
m3    dq   0.0

SEGMENT .text
global _start
_start:

 fld QWORD[m1]
 fld st0
 fmulp st1,st0
 fld QWORD[m2]
 fld st0
 fmulp st1,st0
 faddp st1,st0
 fsqrt
 fst QWORD[m3]
```

```
mov eax,1
mov ebx,0
int 80h
```

Instruções de comparação

Após FCOM	C3	C2	C0
ST>fonte	0	0	0
ST<fonte	0	0	1
ST=fonte	1	0	0
N. Comp.	1	1	1

FCOM ST(i)
- ST – ST(i)

FCOM mem
- ST – [mem]

FCOMP ST(i)
- ST – ST(i)

FCOMPP
- ST – ST1, depois remove ambos da pilha

FICOM mem
- ST – [mem]; [mem] é um inteiro de 16 ou 32 bits

FICOMPP mem
- Compara ST e ST1, depois os remove da pilha

.

Instruções de controle

FINIT

- “Reseta” o FPU.

FCLEX

- Limpa flags de erro.

FSTSW [mem]

- Armazena o registrador de status na memória.

FSTSW AX

- Armazena o registrador de status em AX

FSTCW [mem]

- Armazena registrado de controle na memória

FLDCW [mem]

- Carrega o registrador de controle.

Instruções de controle

FXAM

- Examina o número em STO
- C1 sinal (0 – positivo e 1 – negativo)

Após FXAM	C3	C2	C0
Unsuportted	0	0	0
NaN	0	0	1
Normal	0	1	0
Infinity	0	1	1
Zero	1	0	0
Empty	1	0	1
Denormal	1	1	0

Desvios condicionais

Até o Pentium, para realizar desvios condicionais devia-se carregar o registrador de status de ponto flutuante no EFLAGS (figura a seguir)

Pode-se armazenar os conteúdo do registrador de status de ponto flutuante para memória e posteriormente carregá-lo no registrador de *flags* padrão (via SAHF) ou testar as informações armazenadas na memória com instruções TEST e usar Jcc.

```
fstsw  mem
mov   ax,mem
sahf
```

Desvios condicionais

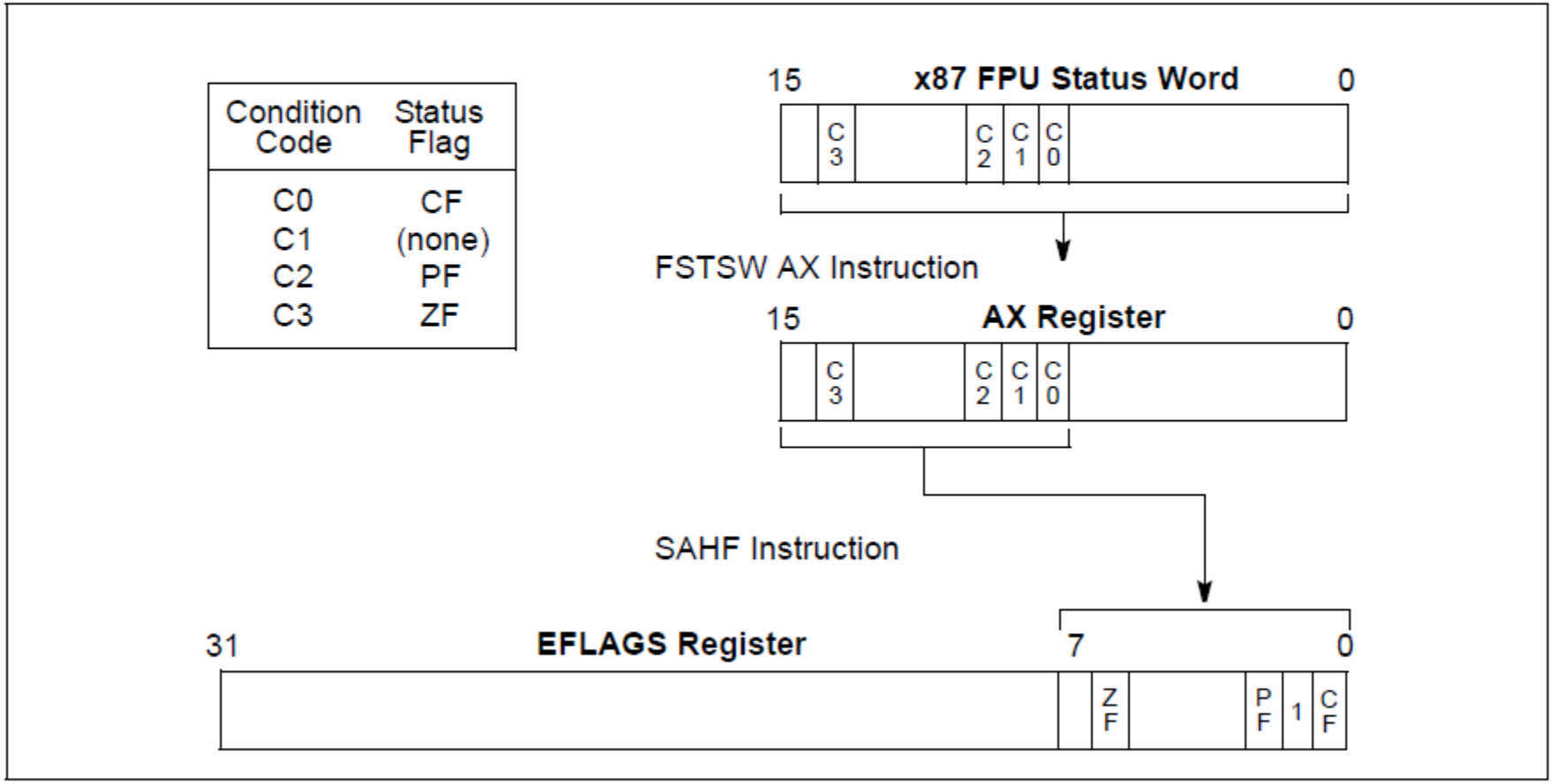


Figure 8-5. Moving the Condition Codes to the EFLAGS Register

Desvios condicionais

- A partir do Pentium Pro, novas instruções de comparação de ponto flutuante permitem setar diretamente o EFLAGS
  - FCOMI
  - FCOMIP
  - FUCOMI
  - FUCOMIP

    P = and pop

    U/sem U – mudança de comportamento (invalid operation exception) em relação a valores NaN

Instruções Transcendentais

FPTAN

- Calcula o tangente do valor de ST. Guarda o valor em ST e empilha 1.

FPATAN

- Calcula o arco-tangente da relação Y/X. X deve estar em ST e Y em ST(1). O resultado é devolvido em ST(1) e ST removido da pilha.

Instruções Transcendentais

FSIN

- Calcula o seno do valor de ST (radianos) e devolve em ST o resultado.

FCOS

- Calcula o cosseno do valor de ST (radianos) e devolve em ST o resultado.

FSINCOS

- Calcula o seno e o cosseno do valor de ST (radianos). O resultado do cosseno é devolvido em ST. O seno é devolvido em ST(1).

Exemplo – Volume Cilindro

```
SEGMENT .data
radio   dq 2.0
height  dq 3.0

SEGMENT .bss
vol     rest 1
vol2    resq 1

SEGMENT .text
global _start
_start:

 fldpi
 fld qword [radio]
 fmul st0,st0
 fmul st0,st1
 fld qword [height]
 fmul st0,st1
```

```
fst qword [vol2]
fbstp [vol]

mov eax,1
mov ebx,0
int 80h
```

Implementação no NASM de int solvep2(double a, double b, double c, double *raiz1, double *raiz2)
Em C: extern int solvep2 (double a, double b, double c, double *raiz1, double *raiz2);

$$\text{root1} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$\text{root2} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
%define   a       qword [EBP+8]
%define   b       qword [EBP+16]
%define   c       qword [EBP+24]
%define   raiz1    dword [EBP+32]
%define   raiz2    dword [EBP+36]

segment .text
global solvep2

solvep2:
      enter 0,0

            • • •
```

```
                  • • •
mov eax,1
jmp done
no_real_roots:
      sub eax, eax
done:
      leave
      ret
```

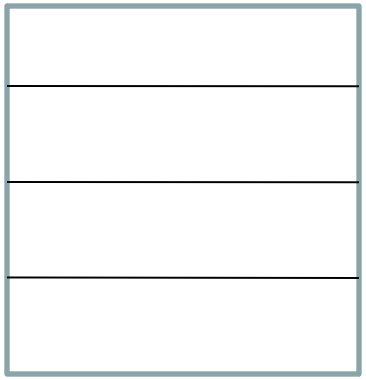
Exemplo

```
%define   a       qword [EBP+8]
%define   b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1   dword [EBP+32]
%define   root2   dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$\text{root1} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$\text{root2} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
fsqrt
fld b
fchs
fadd ST1
fdiv ST2
mov EAX,root1
fstp qword[EAX]
fchs
fld b
fsubp ST1
fdivrp ST1
mov EAX,root2
fstp qword[EAX]
```



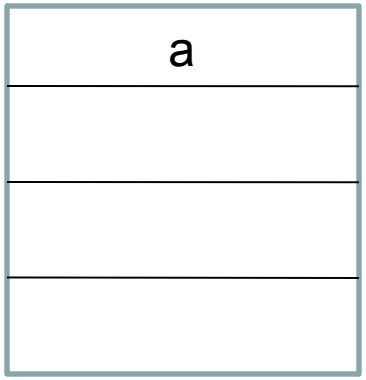
Exemplo

```
%define   a       qword [EBP+8]
%define  b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$\text{root1} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$\text{root2} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
fsqrt
fld b
fchs
fadd ST1
fdiv ST2
mov EAX,root1
fstp qword[EAX]
fchs
fld b
fsubp ST1
fdivrp ST1
mov EAX,root2
fstp qword[EAX]
```



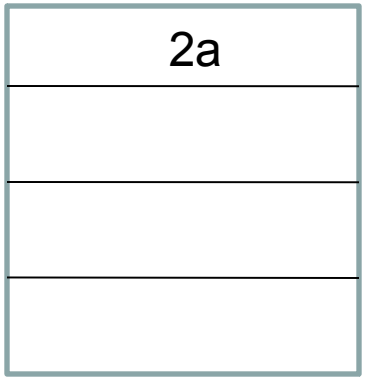
Exemplo

```
%define   a       qword [EBP+8]
%define  b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$root1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$root2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
fsqrt
fld b
fchs
fadd ST1
fdiv ST2
mov EAX,root1
fstp qword[EAX]
fchs
fld b
fsubp ST1
fdivrp ST1
mov EAX,root2
fstp qword[EAX]
```



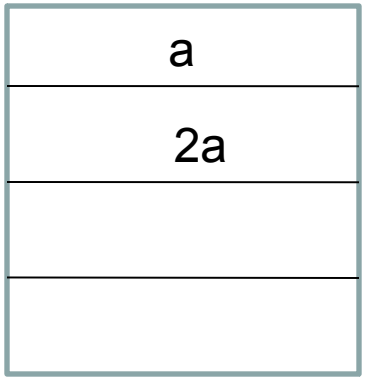
Exemplo

```
%define   a       qword [EBP+8]
%define  b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$\text{root1} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$\text{root2} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
 fsqrt
 fld b
 fchs
 fadd ST1
 fdiv ST2
 mov EAX,root1
 fstp qword[EAX]
 fchs
 fld b
 fsubp ST1
 fdivrp ST1
 mov EAX,root2
 fstp qword[EAX]
```



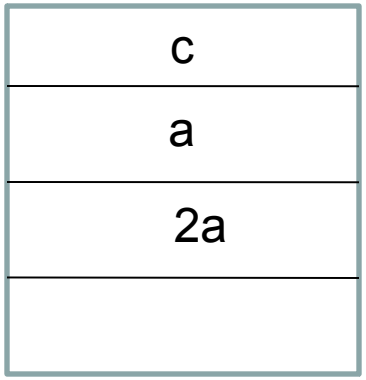
Exemplo

```
%define   a       qword [EBP+8]
%define  b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$root1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$root2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
 fsqrt
 fld b
 fchs
 fadd ST1
 fdiv ST2
 mov EAX,root1
 fstp qword[EAX]
 fchs
 fld b
 fsubp ST1
 fdivrp ST1
 mov EAX,root2
 fstp qword[EAX]
```



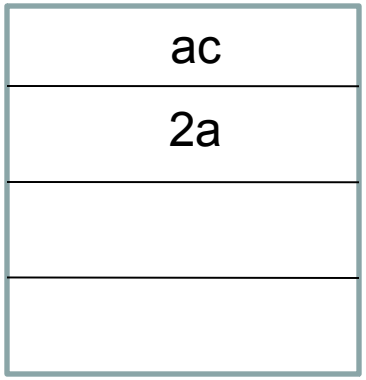
Exemplo

```
%define   a       qword [EBP+8]
%define  b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$root1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$root2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
 fsqrt
 fld b
 fchs
 fadd ST1
 fdiv ST2
 mov EAX,root1
 fstp qword[EAX]
 fchs
 fld b
 fsubp ST1
 fdivrp ST1
 mov EAX,root2
 fstp qword[EAX]
```



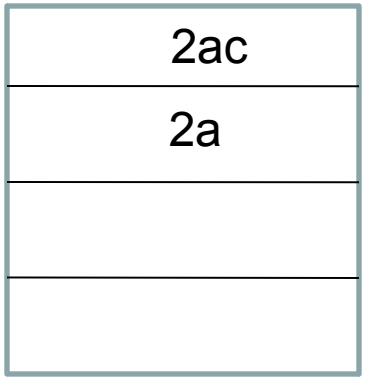
Exemplo

```
%define   a       qword [EBP+8]
%define  b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$root1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$root2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
fsqrt
fld b
fchs
fadd ST1
fdiv ST2
mov EAX,root1
fstp qword[EAX]
fchs
fld b
fsubp ST1
fdivrp ST1
mov EAX,root2
fstp qword[EAX]
```



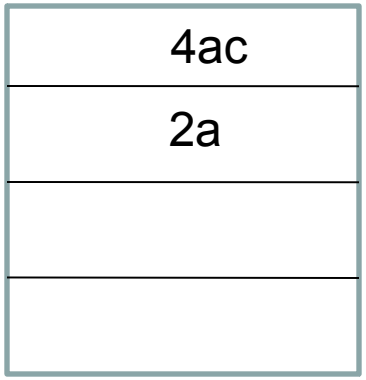
Exemplo

```
%define   a       qword [EBP+8]
%define  b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$\text{root1} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$\text{root2} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
 fsqrt
 fld b
 fchs
 fadd ST1
 fdiv ST2
 mov EAX,root1
 fstp qword[EAX]
 fchs
 fld b
 fsubp ST1
 fdivrp ST1
 mov EAX,root2
 fstp qword[EAX]
```



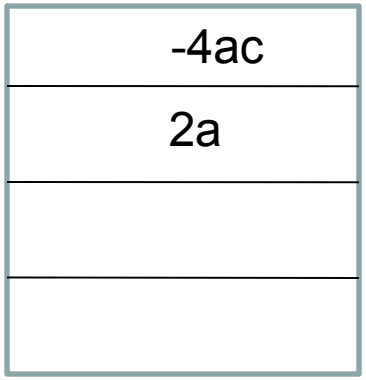
Exemplo

```
%define   a       qword [EBP+8]
%define   b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1   dword [EBP+32]
%define   root2   dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$root1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$root2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
 fsqrt
 fld b
 fchs
 fadd ST1
 fdiv ST2
 mov EAX,root1
 fstp qword[EAX]
 fchs
 fld b
 fsubp ST1
 fdivrp ST1
 mov EAX,root2
 fstp qword[EAX]
```



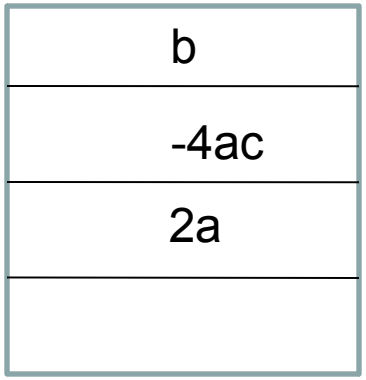
Exemplo

```
%define   a       qword [EBP+8]
%define   b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1   dword [EBP+32]
%define   root2   dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$root1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$root2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
 fsqrt
 fld b
 fchs
 fadd ST1
 fdiv ST2
 mov EAX,root1
 fstp qword[EAX]
 fchs
 fld b
 fsubp ST1
 fdivrp ST1
 mov EAX,root2
 fstp qword[EAX]
```



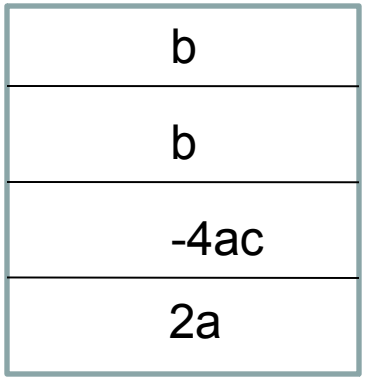
Exemplo

```
%define   a       qword [EBP+8]
%define  b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$root1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$root2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
fsqrt
fld b
fchs
fadd ST1
fdiv ST2
mov EAX,root1
fstp qword[EAX]
fchs
fld b
fsubp ST1
fdivrp ST1
mov EAX,root2
fstp qword[EAX]
```



Exemplo

```
%define   a       qword [EBP+8]
%define   b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$root1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

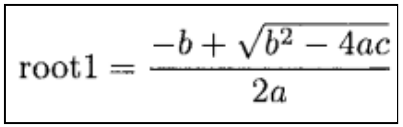
$$root2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
 fsqrt
 fld b
 fchs
 fadd ST1
 fdiv ST2
 mov EAX,root1
 fstp qword[EAX]
 fchs
 fld b
 fsubp ST1
 fdivrp ST1
 mov EAX,root2
 fstp qword[EAX]
```

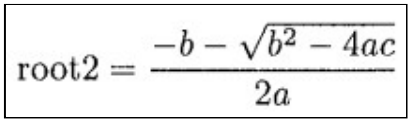
b*b
-4ac
2a

Exemplo

```
%define   a       qword [EBP+8]
%define  b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

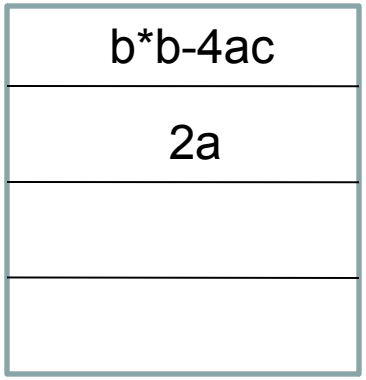


$$root1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$



$$root2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
 fsqrt
 fld b
 fchs
 fadd ST1
 fdiv ST2
 mov EAX,root1
 fstp qword[EAX]
 fchs
 fld b
 fsubp ST1
 fdivrp ST1
 mov EAX,root2
 fstp qword[EAX]
```



Exemplo

```
%define   a       qword [EBP+8]
%define   b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$\text{root1} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$\text{root2} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
fsqrt
fld b
fchs
fadd ST1
fdiv ST2
mov EAX,root1
fstp qword[EAX]
fchs
fld b
fsubp ST1
fdivrp ST1
mov EAX,root2
fstp qword[EAX]
```

b*b-4ac
2a

Exemplo

```
%define   a       qword [EBP+8]
%define  b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$\text{root1} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$\text{root2} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
fsqrt
fld b
fchs
fadd ST1
fdiv ST2
mov EAX,root1
fstp qword[EAX]
fchs
fld b
fsubp ST1
fdivrp ST1
mov EAX,root2
fstp qword[EAX]
```

sqrt(b*b-4ac)
2a

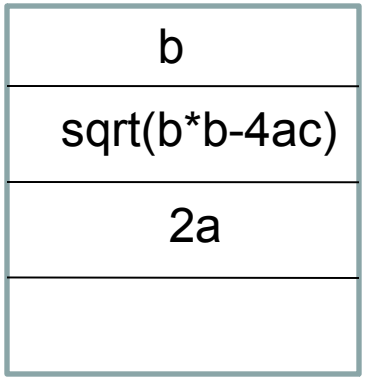
Exemplo

```
%define   a       qword [EBP+8]
%define   b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$\text{root1} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$\text{root2} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
fsqrt
fld b
fchs
fadd ST1
fdiv ST2
mov EAX,root1
fstp qword[EAX]
fchs
fld b
fsubp ST1
fdivrp ST1
mov EAX,root2
fstp qword[EAX]
```



Exemplo

```
%define   a       qword [EBP+8]
%define  b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$\text{root1} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$\text{root2} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
fsqrt
fld b
fchs
fadd ST1
fdiv ST2
mov EAX,root1
fstp qword[EAX]
fchs
fld b
fsubp ST1
fdivrp ST1
mov EAX,root2
fstp qword[EAX]
```

-b
sqrt(b*b-4ac)
2a

Exemplo

```
%define   a       qword [EBP+8]
%define  b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$\text{root1} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$\text{root2} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
fsqrt
fld b
fchs
fadd ST1
fdiv ST2
mov EAX,root1
fstp qword[EAX]
fchs
fld b
fsubp ST1
fdivrp ST1
mov EAX,root2
fstp qword[EAX]
```

-b+sqrt(b*b-4ac)
sqrt(b*b-4ac)
2a

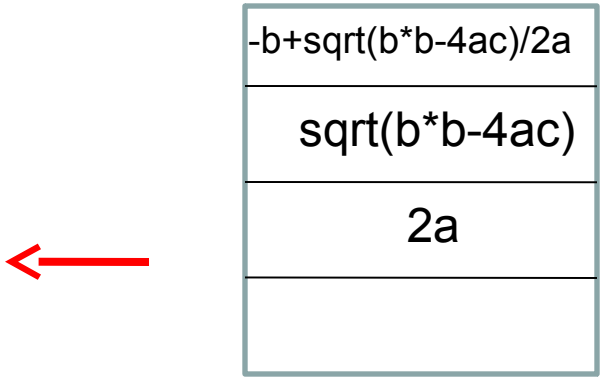
Exemplo

```
%define   a       qword [EBP+8]
%define  b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$\text{root1} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$\text{root2} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
 fsqrt
 fld b
 fchs
 fadd ST1
 fdiv ST2
 mov EAX,root1
 fstp qword[EAX]
 fchs
 fld b
 fsubp ST1
 fdivrp ST1
 mov EAX,root2
 fstp qword[EAX]
```



Exemplo

```
%define   a       qword [EBP+8]
%define  b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$\text{root1} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$\text{root2} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
 fsqrt
 fld b
 fchs
 fadd ST1
 fdiv ST2
 mov EAX,root1
 fstp qword[EAX]
 fchs
 fld b
 fsubp ST1
 fdivrp ST1
 mov EAX,root2
 fstp qword[EAX]
```

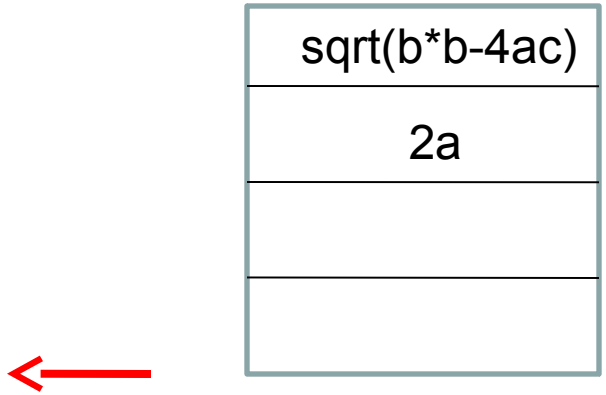
Exemplo

```
%define   a       qword [EBP+8]
%define  b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$\text{root1} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$\text{root2} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
fsqrt
fld b
fchs
fadd ST1
fdiv ST2
mov EAX,root1
fstp qword[EAX]
fchs
fld b
fsubp ST1
fdivrp ST1
mov EAX,root2
fstp qword[EAX]
```



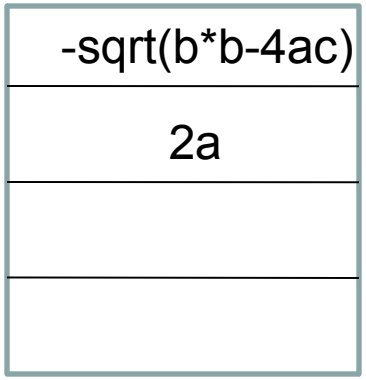
Exemplo

```
%define   a       qword [EBP+8]
%define   b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$\text{root1} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$\text{root2} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
fsqrt
fld b
fchs
fadd ST1
fdiv ST2
mov EAX,root1
fstp qword[EAX]
fchs
fld b
fsubp ST1
fdivrp ST1
mov EAX,root2
fstp qword[EAX]
```



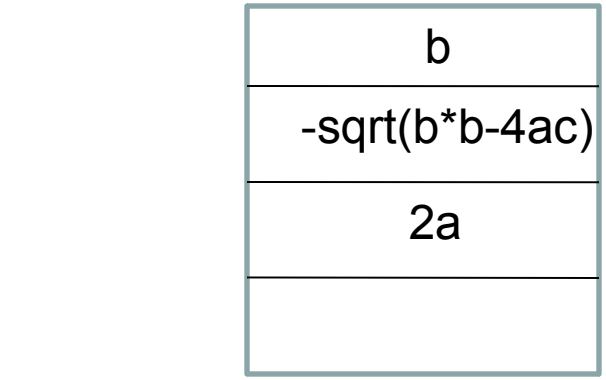
Exemplo

```
%define   a       qword [EBP+8]
%define   b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1   dword [EBP+32]
%define   root2   dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$root1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$root2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
 fsqrt
 fld b
 fchs
 fadd ST1
 fdiv ST2
 mov EAX,root1
 fstp qword[EAX]
 fchs
 fld b
 fsubp ST1
 fdivrp ST1
 mov EAX,root2
 fstp qword[EAX]
```



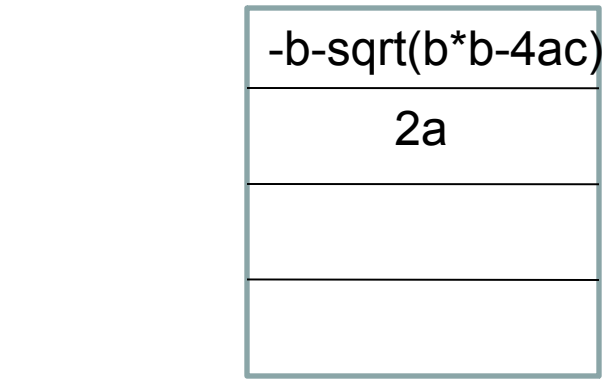
Exemplo

```
%define   a       qword [EBP+8]
%define  b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$\text{root1} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$\text{root2} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
 fsqrt
 fld b
 fchs
 fadd ST1
 fdiv ST2
 mov EAX,root1
 fstp qword[EAX]
 fchs
 fld b
 fsubp ST1
 fdivrp ST1
 mov EAX,root2
 fstp qword[EAX]
```



Exemplo

```
%define   a       qword [EBP+8]
%define  b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$\text{root1} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$\text{root2} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
 fsqrt
 fld b
 fchs
 fadd ST1
 fdiv ST2
 mov EAX,root1
 fstp qword[EAX]
 fchs
 fld b
 fsubp ST1
 fdivrp ST1
 mov EAX,root2
 fstp qword[EAX]
```

-b-sqrt(b*b-4ac)/2a

Exemplo

```
%define   a       qword [EBP+8]
%define  b       qword [EBP+16]
%define   c       qword [EBP+241
%define   root1    dword [EBP+32]
%define   root2    dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$\text{root1} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$\text{root2} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
fsqrt
fld b
fchs
fadd ST1
fdiv ST2
mov EAX,root1
fstp qword[EAX]
fchs
fld b
fsubp ST1
fdivrp ST1
mov EAX,root2
fstp qword[EAX]
```

-b-sqrt(b*b-4ac)/2a

Exemplo

```
%define   a      qword [EBP+8]
%define  b      qword [EBP+16]
%define   c      qword [EBP+241
%define   root1   dword [EBP+32]
%define   root2   dword [EBP+36]
 ...
 fld a
 fadd ST0
 fld a
 fld c
 fmulp ST1
 fadd ST0
 fadd ST0
 fchs
 fld b
 fld b
 fmulp ST1
 faddp ST1
 ftst
 fstsw AX
 sahf
 jb sem_raizes_reais
```

$$\text{root1} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$\text{root2} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
fsqrt
fld b
fchs
fadd ST1
fdiv ST2
mov EAX,root1
fstp qword[EAX]
fchs
fld b
fsubp ST1
fdivrp ST1
mov EAX,root2
fstp qword[EAX]
```