Pontos Flutuantes

- · Carregar ponto flutuante da variável para pilha
- · Carregar ponto flutuante da variável para pilha
- · Fazer operação
- · Retirar resultado da pilha e colocar na variável

Pontos Flutuantes

Sentido de empilhamento

SECTION .data

a dq 3.64 b dq 6.25

c dq 0

SECTION .text

fld a fld b fadd fstp c st7 st6 st5 st4 st3

st2

st1

st0

SECTION .data

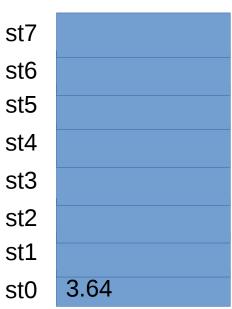
a dq 3.64 b dq 6.25 c dq 0

SECTION .text

fld a

fld b fadd

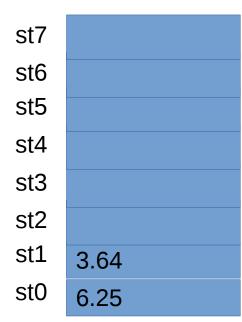
fstp c



SECTION .data a dq 3.64 b dq 6.25 c dq 0

SECTION .text

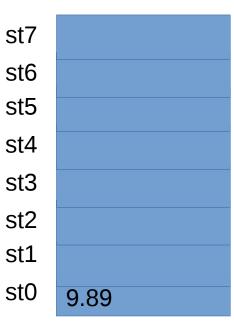
fld a fld b fadd fstp c



SECTION .data a dq 3.64 b dq 6.25 c dq 0

SECTION .text

fld a fld b fadd fstp c



SECTION .data

a dq 3.64

b dq 6.25

c dq 9.89

SECTION .text

fld a fld b fadd

fstp c



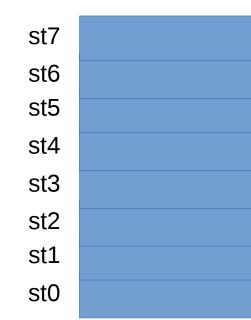
st1

st0

Computar z = (x-1) * (y+3.5), onde x é 1.5 e y é 2.5

SECTION .data

| msg | db | "soma = $\%$ e",0x0a,0x00 |
|------|----|---------------------------|
| Χ | dd | 1.5 |
| У | dd | 2.5 |
| Z | dd | 0 |
| temp | da | 0 |



```
main:
    fld
       dword [x]; st0 <- x
                                                                   st7
    fld1
               : st0 <- 1 st1 <- x
                    : st0 <- x-1
    fsub
                                                                   st6
                                                                   st5
    fld dword [y]
                         ; st0 <- y st1 <- x-1
                                                                  st4
    fld dword [esp]
                         ; st0 <- 3.5 st1 <- y st2 <- x-1
                         ; st0 <- y+3.5 st1 <- x-1
    fadd
                                                                  st3
                         ; st0 <- (x-1) * (y+3.5)
    fmul
                                                                  st2
    fst dword [z]
                         ; armazena a soma em z
                                                                  st1
    fld dword [z] ; transforma z em 64 bits
                                                                   st0
    fstp qword [temp]; armazena z como palavra de 64 bits
```

```
main:
```

```
fld dword [x]; st0 <- x
                                                              st7
fld1
          : st0 <- 1 st1 <- x
                : st0 <- x-1
fsub
                                                              st6
                                                              st5
fld dword [y]
                     ; st0 <- y st1 <- x-1
                                                              st4
fld dword [esp]
                    ; st0 <- 3.5 st1 <- y st2 <- x-1
                     ; st0 <- y+3.5 st1 <- x-1
fadd
                                                              st3
fmul
                     ; st0 <- (x-1) * (y+3.5)
                                                              st2
fst dword [z]
                     ; armazena a soma em z
                                                              st1
fld dword [z] ; transforma z em 64 bits
                                                              st0
                                                                   X
fstp qword [temp]; armazena z como palavra de 64 bits
```

```
main:
    fld
       dword [x]; st0 <- x
                                                                  st7
    fld1
              : st0 <- 1 st1 <- x
                    : st0 <- x-1
    fsub
                                                                  st6
                                                                  st5
    fld dword [y]
                         ; st0 <- y st1 <- x-1
                                                                  st4
    fld dword [esp]
                        ; st0 <- 3.5 st1 <- y st2 <- x-1
                         ; st0 <- y+3.5 st1 <- x-1
    fadd
                                                                  st3
    fmul
                         ; st0 <- (x-1) * (y+3.5)
                                                                  st2
    fst dword [z]
                         ; armazena a soma em z
                                                                  st1
                                                                        X
    fld dword [z] ; transforma z em 64 bits
                                                                  st0
    fstp qword [temp]; armazena z como palavra de 64 bits
```

```
main:
    fld
       dword [x]; st0 <- x
                                                                   st7
    fld1
                : st0 <- 1 st1 <- x
    fsub
                    : st0 <- x-1
                                                                   st6
                                                                   st5
    fld dword [y]
                         ; st0 <- y st1 <- x-1
                                                                   st4
    fld dword [esp]
                         ; st0 <- 3.5 st1 <- y st2 <- x-1
                         ; st0 <- y+3.5 st1 <- x-1
    fadd
                                                                   st3
    fmul
                         ; st0 <- (x-1) * (y+3.5)
                                                                   st2
    fst dword [z]
                         ; armazena a soma em z
                                                                   st1
    fld dword [z] ; transforma z em 64 bits
                                                                        x-1
                                                                   st0
    fstp qword [temp]; armazena z como palavra de 64 bits
```

```
main:
    fld
       dword [x]; st0 <- x
                                                                   st7
    fld1
                    : st0 <- 1 st1 <- x
                     : st0 <- x-1
    fsub
                                                                   st6
                                                                   st5
    fld dword [y]
                         ; st0 <- v st1 <- x-1
    fld dword [esp] ; st0 <- 3.5 st1 <- y st2 <- x-1
                                                                   st4
                         ; st0 <- y+3.5 st1 <- x-1
    fadd
                                                                   st3
    fmul
                         ; st0 <- (x-1) * (y+3.5)
                                                                   st2
    fst dword [z]
                         ; armazena a soma em z
                                                                   st1
                                                                         x-1
    fld dword [z] ; transforma z em 64 bits
                                                                   st0
    fstp qword [temp]; armazena z como palavra de 64 bits
```

```
main:
    fld
       dword [x]; st0 <- x
                                                                   st7
    fld1
                : st0 <- 1 st1 <- x
                    : st0 <- x-1
    fsub
                                                                   st6
                                                                   st5
    fld dword [y]
                         ; st0 <- y st1 <- x-1
    fld dword [esp] ; st0 <- 3.5 st1 <- y st2 <- x-1
                                                                   st4
                        ; st0 <- y+3.5 st1 <- x-1
    fadd
                                                                   st3
                        t = 10 < -(x-1) * (y+3.5)
    fmul
                                                                        x-1
                                                                   st2
    fst dword [z]
                         ; armazena a soma em z
                                                                        У
                                                                   st1
    fld dword [z] ; transforma z em 64 bits
                                                                        3.5
                                                                   st0
    fstp qword [temp]; armazena z como palavra de 64 bits
```

```
main:
    fld
       dword [x]; st0 <- x
                                                                    st7
    fld1
                    : st0 <- 1 st1 <- x
                     : st0 <- x-1
    fsub
                                                                    st6
                                                                    st5
    fld dword [y]
                         ; st0 <- y st1 <- x-1
                                                                    st4
    fld dword [esp]
                         ; st0 <- 3.5 st1 <- y st2 <- x-1
    fadd
                         ; st0 <- y+3.5 st1 <- x-1
                                                                    st3
    fmul
                         ; st0 <- (x-1) * (y+3.5)
                                                                    st2
    fst dword [z]
                         ; armazena a soma em z
                                                                    st1
                                                                         x-1
    fld dword [z] ; transforma z em 64 bits
                                                                         y+3.5
                                                                    st0
    fstp qword [temp]; armazena z como palavra de 64 bits
```

```
main:
    fld
        dword [x]; st0 <- x
                                                                    st7
    fld1
                    : st0 <- 1 st1 <- x
                     : st0 <- x-1
    fsub
                                                                    st6
                                                                    st5
    fld dword [y]
                         ; st0 <- y st1 <- x-1
                                                                    st4
    fld dword [esp]
                         ; st0 <- 3.5 st1 <- y st2 <- x-1
                         ; st0 <- y+3.5 st1 <- x-1
    fadd
                                                                    st3
    fmul
                         ; st0 <- (x-1)*(y+3.5)
                                                                    st2
    fst dword [z]
                         ; armazena a soma em z
                                                                    st1
    fld dword [z] ; transforma z em 64 bits
                                                                    st0
                                                                         (x-1)*(y+3.5)
    fstp qword [temp]; armazena z como palavra de 64 bits
```

```
main:
    fld
       dword [x]; st0 <- x
                                                                  st7
    fld1
               : st0 <- 1 st1 <- x
                    : st0 <- x-1
    fsub
                                                                  st6
                                                                  st5
    fld dword [y]
                        ; st0 <- y st1 <- x-1
                                                                  st4
    fld dword [esp]
                        ; st0 <- 3.5 st1 <- y st2 <- x-1
                        ; st0 <- y+3.5 st1 <- x-1
    fadd
                                                                  st3
                         ; st0 <- (x-1) * (y+3.5)
    fmul
                                                                  st2
    fst dword [z]
                         ; armazena a soma em z
                                                                  st1
    fld dword [z] ; transforma z em 64 bits
                                                                  st0
    fstp qword [temp]; armazena z como palavra de 64 bits
```

```
main:
    fld
       dword [x]; st0 <- x
                                                                   st7
    fld1
                : st0 <- 1 st1 <- x
                    : st0 <- x-1
    fsub
                                                                   st6
                                                                   st5
    fld dword [y]
                         ; st0 <- y st1 <- x-1
                                                                   st4
    fld dword [esp]
                         ; st0 <- 3.5 st1 <- y st2 <- x-1
                         ; st0 <- y+3.5 st1 <- x-1
    fadd
                                                                   st3
    fmul
                         ; st0 <- (x-1) * (y+3.5)
                                                                   st2
    fst dword [z]
                         ; armazena a soma em z
                                                                   st1
    fld dword [z] ; transforma z em 64 bits
                                                                   st0
                                                                        Ζ
    fstp qword [temp]; armazena z como palavra de 64 bits
```

```
main:
    fld
       dword [x]; st0 <- x
                                                                   st7
    fld1
                : st0 <- 1 st1 <- x
                    : st0 <- x-1
    fsub
                                                                   st6
                                                                   st5
    fld dword [y]
                         ; st0 <- y st1 <- x-1
                                                                   st4
    fld dword [esp]
                         ; st0 <- 3.5 st1 <- y st2 <- x-1
                         ; st0 <- y+3.5 st1 <- x-1
    fadd
                                                                   st3
                         ; st0 <- (x-1) * (y+3.5)
    fmul
                                                                   st2
    fst dword [z]
                         ; armazena a soma em z
                                                                   st1
    fld dword [z] ; transforma z em 64 bits
                                                                   st0
    fstp gword [temp] ; armazena z como palavra de 64 bits
```

Diferenças entre código 32 e 64 bits

```
printf("soma = %e\n", temp);

msg db "sum = %e",0x0a,0x00

push dword [temp+4]
push dword [temp]
push dword msg
call printf
add esp, 12
```

```
mov rdi, msg
movq xmm0 , qword [temp]
mov rax,1
call printf
```