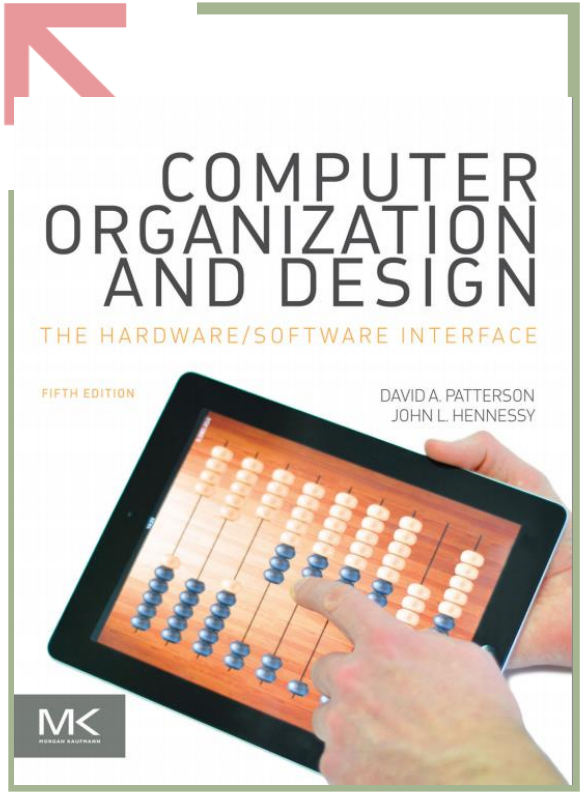
Assembly programming

floating point process

wangw6@sustc.edu.cn



Topics

- Floating point number
- Floating point instructions
- exercise

IEEE 745 on floating point number

```
.data
fneg1:    .float -1
wneg1:    .word -1
fpos1:    .float 1
wpos1:    .word  1
```

$$\pm 1.xxxxxxx_2 \times 2^{yyyy}$$

single: 8 bits
double: 11 bits

single: 23 bits
double: 52 bits

$$x = (-1)^S \times (1 + Fraction) \times 2^{(Exponent - Bias)}$$

Exponents 00000000 and 11111111 reserved

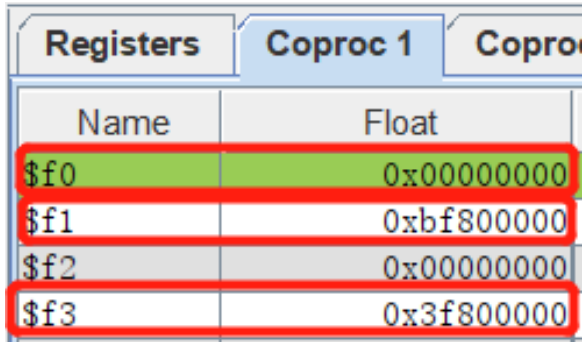
Label	Address ▲
float_rw.asm	
fneg1	0x10010000
wneg1	0x10010004
fpos1	0x10010008
wpos1	0x1001000c

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)
0x10010000	0xbf800000	0xffffffff	0x3f800000	0x00000001

Coprocessor 1 in MIPS

which one will trigger an exception ?
which one will get the right answer



Runtime exception at 0x00400004: first register must be even-numbered

```
.data
        fneg1:    .float -1
        fpos1:    .float 1
.text
        lwc1 $f1,fneg1
        lwc1 $f3,fpos1
        add.s $f0,$f1,$f3

        li $v0,10
        syscall
```

```
.data
        fneg1:    .double  -1
        fpos1:    .double  1
.text
        l.d $f1,fneg1
        l.d $f3,fpos1
        add.s $f0,$f1,$f3

        li $v0,10
        syscall
```

```
.data
        fneg1:    .double  -1
        fpos1:    .double  1
.text
        l.d $f0,fneg1
        l.d $f2,fpos1
        add.s $f0,$f1,$f3

        li $v0,10
        syscall
```

Floating-point instructions

The floating-point coprocessor has these classes of instructions:

- Load and Store Instructions: Load values and move data between memory and coprocessor registers.
- Move Instructions: Move data between registers.
- Computational Instructions: Do arithmetic and logical operations on values in coprocessor registers.
- Relational Instructions: Compare two floating-point values

Infinite vs NaN (floating-point)

```
.data
        sdata: .word 0xff7f7fff
        fneg1: .float -1
        zdata: .word 0x007fffff
.text
        lw $t0,sdata
        mtc1 $t0,$f1
        mul.s $f12,$f1,$f1

        li $v0,2
        syscall

        lwc1 $f2,fneg1
        mul.s $f12,$f12,$f2

        li $v0,2
        syscall

        li $v0,10
        syscall
```

which one will get an infinite value?
which one will get the NaN

```
.data
        sdata: .word 0xffff7fff
        fneg1: .float -1
        zdata: .word 0x007fffff
.text
        lw $t0,sdata
        mtc1 $t0,$f1
        mul.s $f12,$f1,$f1

        li $v0,2
        syscall

        lwc1 $f2,fneg1
        div.s $f12,$f12,$f12

        li $v0,2
        syscall

        li $v0,10
        syscall
```

Lab exercise

1. There are 5 students in a class , every students attend 10 lab classes and got its score(integer from 0 to 10). All the scores are stored in a two-dimensional （5*10） array. Print out the index of the lab class whose performance is not so good(the average score of the lab is lower than the total average score)

2. Calculate the square root of an integer number without using “sqrt.s” and “sqrt.d”

- Get the input data and the precision value from input device
- If the input data is a negative number, print out the warning message and exit
- If the input data is a positive number, calculate its square root value which can satisfy the accuracy requirement and print it out