VE370

Bingcheng HU

516 021 910 219

HW2

☆PC是当前的指令

1.

	6 op	5 RS	5 rt	5 rd	5 shamt	6 funct	Location
LOOP: R(set)	0	$zero 0	$t0 8	$t2 10	0	16×2+10=42 ✓ 2a 0010 ✓ 1010	
I(bne)	5	$t2 10	$zero 0	ELSE		PC	0x10000400
J(j)	2	DONE					0x10000404
ELSE: I(addi)	8	$~~zero~~ s2 18	$ ~~t2~~ s2 18	2			0x10000408
I(addiu)	9	$ t0 8	$ t0 8	1			0x1000040C
J(j)	2	LOOP					0x10000410
DONE:							0x10000414
							0x10000418

translate to binary code:

000000 ~~00000~~ 00000 ~~00110~~ 01000 ~~00000~~ 01010 ~~10110~~ 00000 1 01010

~~01010~~ 001001 ~~00000~~ 01010 00000 [(~~40C-402~~ 40C-404) - 4]/4 = ~~2~~ ← 即中间隔除

000010 00000 00000 | 00000 | 0 0100 | 0001 1000

18=2^4+2

(0000418)/4 = 右移2位

001000 10010 10010 00000 | 00000 | 000010

9=8+1

001001 01000 01000 00000 00000 00001

(0000400)/4 = 0000100

000010 00000 00000 0000 0001 0000 0000

Q2: lui $t0 4096 ($4096_{(10)} = 1\overset{12个0}{0\cdots0}_{(2)}$) 结果 t0 = 0x10000000

lb $s1 1($t0)

Q3: 2.19.1

How to write recursion?

First → sw (for preserved value: $s0, $sp, $ra) ↑ → stack point ↓

(for not preserved: $t0, $a0, $v0)

```
fib_iter:
  # first, save to stack:
  addi $sp, $sp, -16   # adjust 4 byte

save all preserved value that will be used:
  sw $s0 , 0($sp)
  sw $s1 , 4 ~~1~~ ($sp)   —— 注意 X4 !!
  sw $s2 , 8 ~~2~~ ($sp)
  sw $ra , 12 ~~3~~ ($sp)

put a → s
  addi $s0 , $a0 , 0    # int a → s0.
  addi $s1 , $a1 , 0    # int b → s1.
  addi $s2 , $a2 , 0    # int n → s2.
  ~~addi~~
比条件跳出
  addi $t0 , $0 , 0
  beq $s2 , $t0 , exit   # if (n==0) exit.

递归
  add $a0, $s0, $s1
  addi $a1 , $s1 , 0
  addi $a2, $s2 -1
  jal fib-iter.
exit:
跳出
  lw { lw $s0 , 0($sp)
       ⋮
       lw $ra, 3($sp)
                    , addi $sp. $sp. 16 销栈
  addi $v0 , $s1 , 0   # return V0 = s1 = b
  j $ra
```

2.19.2

?

2.19.3

old $sp → 0x7ffffffc

-4	$s0
-8	$s1
-12	$s2
$sp ⟶ -16	$ra.

2.23.1

存ra

存 t0←a0

设 S0←0 #为数.

Loop: ~~查 t0='\0'?跳~~

t0>'9' 跳

t0<'0' 跳

s0+=t0

(~~t0~~ - '0')×10 ← t0下一位='\0',完成 DONE:

DONE: 取ra,

存 V0←S0

给栈

跳

MAIN:

addi $sp $sp -4

sw $ra ,0($sp)

add $s0, $0, $0

— — — — —

~~LOOP~~: add $t0, $a0, $0.

add $t4, 0($t0)

add $t1, $0, '0'

add $t2, $0, '9'

slt $t3, $t4, $t1

b~~eq~~ne $t3, $0, DONE

slt $t3, $t2, $t4

bne $t3, $0, DONE.

不能直接将数组指针当数用,要先把出

~~add $s0, $s0,~~

sub $s0, $t0, $t1

beq 1($t0), $0, DONE.

mul $s0, $s0, 10

j Loop

DONE:

lw $ra, 0($sp).

add $sp, $sp, 4.

add $v0, $s0, $0

jr $ra.

2-24.3.

$\$t_2 = \$t_1 + 10_{(2)} = \$t_1 + 2_{(10)}$

$t1 → $0x\underline{1000}\,\underline{0000}$ → $\underline{110000FE}$ = $\underline{0001}\ \underline{0001}\ \underline{0}\ \underline{0}\ \underline{0}\ \underline{0}\ \underline{1111}\ \underline{1111}$

$t2 → $0x\underline{1000}\,\underline{0010}$ → ~~[illegible]~~ 00 → 04 → 08 $\overset{10}{\downarrow}$ 0C → 00

↑

5555 5555.

2.26.1 use beq $s0, $t0, Loop: 16 bit:

Loop = PC + 4 + Relative Addr * 4

$2^{16} \times 4 >$ 0x0000 1000, 1 branch.

2.26.2

LOOP = PC + 4 + RA * 4

⇒ RA = (LOOP − 4 − PC)/4

$$\left.\begin{array}{r} 0X0002\,0000. \\ -\ 0X0000\,0000 \\ -\quad 4 \\ \hline \end{array}\right/ 4 = \begin{array}{c} 0X0000\,8000 \\ -1 \end{array}$$

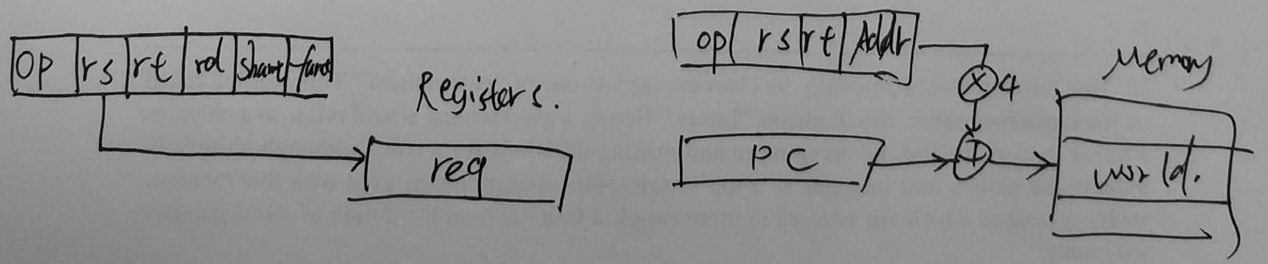
~~0X000~~

[illegible] 0X000 8000

2~~8~~

2x16 ÷ 4 = 8

2.27.1


op
rs
rt
rd
shamt
funct
Registers.
reg
op
rs
rt
Addr
×4
+
PC
Memory
world.