计算机体系架构　第二周作业

范云潜 18373486

微电子学院 184111 班

日期：2020 年 9 月 17 日

作业内容：2.10, 2.11, 2.13, 2.14; 2.15, 2.16, 2.21, 2.29;

Problem 2.10

```
    .data
  .align 2
jtab: # jump table
  .word L0, L1, L2, L3, L4 # exit

  .text

.macro ret # return
  jr    $ra
.end_macro



# f g h j i k
# 0 1 2 3 4 5

main: # f-k -> s0-s5
  li  $t2, 4 # test code
  li  $s5, 1 # test code
  li  $s1, 1 # test code
  li  $s2, 2 # test code
  li  $s3, 3 # test code
  li  $s4, 4 # test code
  sub   $t1, $t2, $s5 # t1 = 4-k
  slt $t0, $zero, $t1 #  0 < 4 - k should be 1
  beq   $t0, $zero, L4 # check fail then exit
  slt   $t0, $s5, $zero # k < 0 should be 0
  bne $t0, $zero, L4 # check fail then exit
  mul $t1, $t2, $s5 # calculate the bias
  la  $t0, jtab # t0 = addr_of_switch + 4 * k
  add   $t1, $t0, $t1 # advance the pointer
  lw  $t0, ($t1) # load the dest memory addr
  jalr  $t0 # start switch
  j   L4 # exit

L0:   add   $s0, $s3, $s4 # case 0
  ret
L1:   add $s0, $s1, $s2 # case 1
  ret
L2:   sub $s0, $s1, $s2 # case 2
  ret
L3: sub   $s0, $s3, $s4 # case 3
  ret
L4:   li      $v0, 10 # exit
  syscall
```

Problem 2.11

SubProblem a

```
if (k == 0) {
    f = i + j;
}else if(k == 1) {
    f = g + h;
}else if (k == 2) {
    f = g - h;
}else if (k == 3) {
    f = i - j;
}else {
    return 0; // check failed
}
```

SubProblem b

```
  .data
  .align 2
jtab:
  .word L0, L1, L2, L3, L4 # exit

  .text

.macro ret
  jr    $ra
.end_macro

.macro exit
```

```
  li      $v0, 10 # exit
  syscall
.end_macro
# f g h j i k
# 0 1 2 3 4 5

main: # f-k -> s0-s5
  li  $t2, 4 # test code
  li  $s5, 1 # test code
  li  $s1, 1 # test code
  li  $s2, 2 # test code
  li  $s3, 3 # test code
  li  $s4, 4 # test code
  beq   $t1, $zero, L0 # cmp with 0
  subi  $t1, $s5, 1
  beq   $t1, $zero, L1 # cmp with 1
  subi  $t1, $s5, 2
  beq   $t1, $zero, L2 # cmp with 2
  subi  $t1, $s5, 3
  beq   $t1, $zero, L3 # cmp with 3
  j   L4 # else: return



L0:   add   $s0, $s3, $s4 # case 0
  exit
L1:   add $s0, $s1, $s2 # case 1
  exit
L2:   sub $s0, $s1, $s2 # case 2
  exit
L3: sub   $s0, $s3, $s4 # case 3
  exit
L4:
  exit
```

SubProblem 3

对于跳转表，算数类：6 ，传输类：3 ，分支类：2，跳转类：3 ，共 17.2 clk 。
对于 if-else ，算数类：4 ，传输类：1 ，分支类：4，跳转类：1 ，共 13.4 clk 。
但是对于更多分支类型的表达式，跳转表会更加迅速。

Problem 2.13

如 图 1。

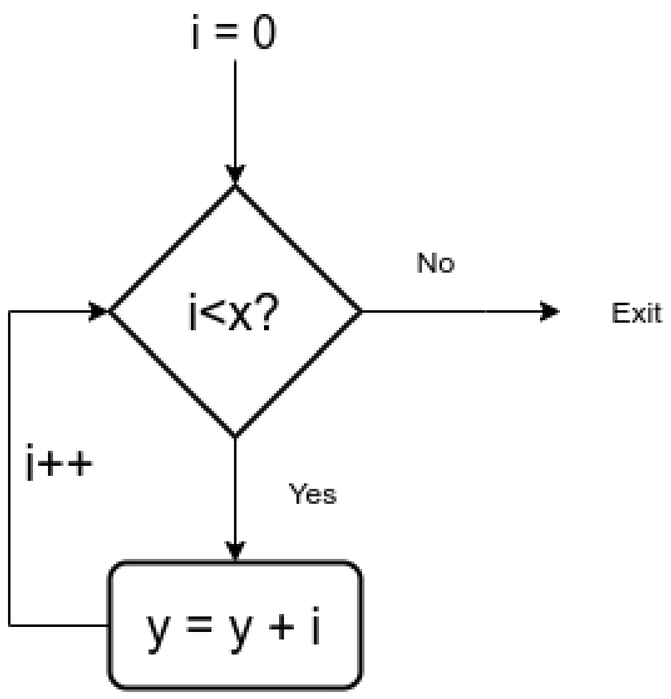

图 1: 循环流程图

Problem 2.14

```
    .data
  .align 2
jtab: # jump table
  .word Loop, Exit

  .text
.macro ret # return
  jr    $ra
.end_macro

.macro exit
  li      $v0, 10 # exit
  syscall
.end_macro

# s3 -> i
# s6 -> save
# s5 -> k

main:

Loop:
    sll      $t1, $s3, 2 # t1 = i * 4
    add      $t1, $t1, $s6
    lw       $t0, 0($t1)
  sub   $t1, $t2, $s5 # t1 = 4-k
  slt $t0, $zero, $t1 #  0 < 4 - k should be 1
  beq   $t0, $zero, L4 # check fail then exit
  slt   $t0, $s5, $zero # k < 0 should be 0
  bne $t0, $zero, L4 # check fail then exit
  mul $t1, $t2, $s5 # calculate the bias
  la  $t0, jtab # t0 = addr_of_switch + 4 * k
  add   $t1, $t0, $t1 # advance the pointer
  lw  $t0, ($t1) # load the dest memory addr

  jalr  $t0 # start switch
  j   L4 # exit

L0:   add   $s0, $s3, $s4 # case 0
  ret
L1:   add $s0, $s1, $s2 # case 1
  ret
L2:   sub $s0, $s1, $s2 # case 2
  ret
L3: sub   $s0, $s3, $s4 # case 3
  ret
L4:   li      $v0, 10 # exit
  syscall
```

Problem 2.15

```
    .text

.macro ret # return
  jr    $ra
.end_macro

# int i in $s0

set_array:
  # allocate space for: fp/ra/array[10]/num = 4 * (1+1+10+1) = 52
  addi  $sp, $sp, -52
  # store fp, ra, num for the caller,
  # num is the only args, in a0
  sw  $fp, 48($sp)
  sw  $ra, 44($sp)
  sw  $a0, 40($sp)
  # init the fp for the stack
  addi  $fp, $sp, 48
  # set i = 0, max = 10
  add   $s0, $zero, $zero
  addi  $t0, $zero, 10


loop:
  # set bias as 4*i to index array
  sll   $t1, $s0, 2
  add   $t1, $t1, $sp
  # i++
  addi  $s0, $s0, 1
  # set num and i as args
  add   $a0, $a0, $zero
  add   $a1, $s0, $zero
  jal   compare
  sw  $v0, ($t1)
  bne   $s0, $t0, loop

  # loop end, then restore the stack for caller
  lw  $fp, 48($sp)
  lw  $ra, 44($sp)
  lw  $a0, 40($sp)
  addi  $sp, $sp, 52
  ret

compare:
  # allocate for fp/ra
  addi  $sp, $sp, -8
  sw  $fp, 4($sp)
  sw  $ra, ($sp)

  jal Sub
  # if (v0 < 0)==1 < 1 == 0, then return 0; else return 1
  slt $v0, $v0, $zero
  slti  $v0, $v0, 1

  # restore the frame for caller
  lw  $fp, 4($sp)
  lw  $ra, ($sp)
  addi  $sp, $sp, 8
  ret


Sub:
  # allocate for fp/ra
  addi  $sp, $sp, -8
  sw  $fp, 4($sp)
  sw  $ra, ($sp)
```

```
sub $v0, $a0, $a1

# restore the frame for caller
lw  $fp, 4($sp)
lw  $ra, ($sp)
addi  $sp, $sp, 8
ret
```

栈的示意图如 **图 2**

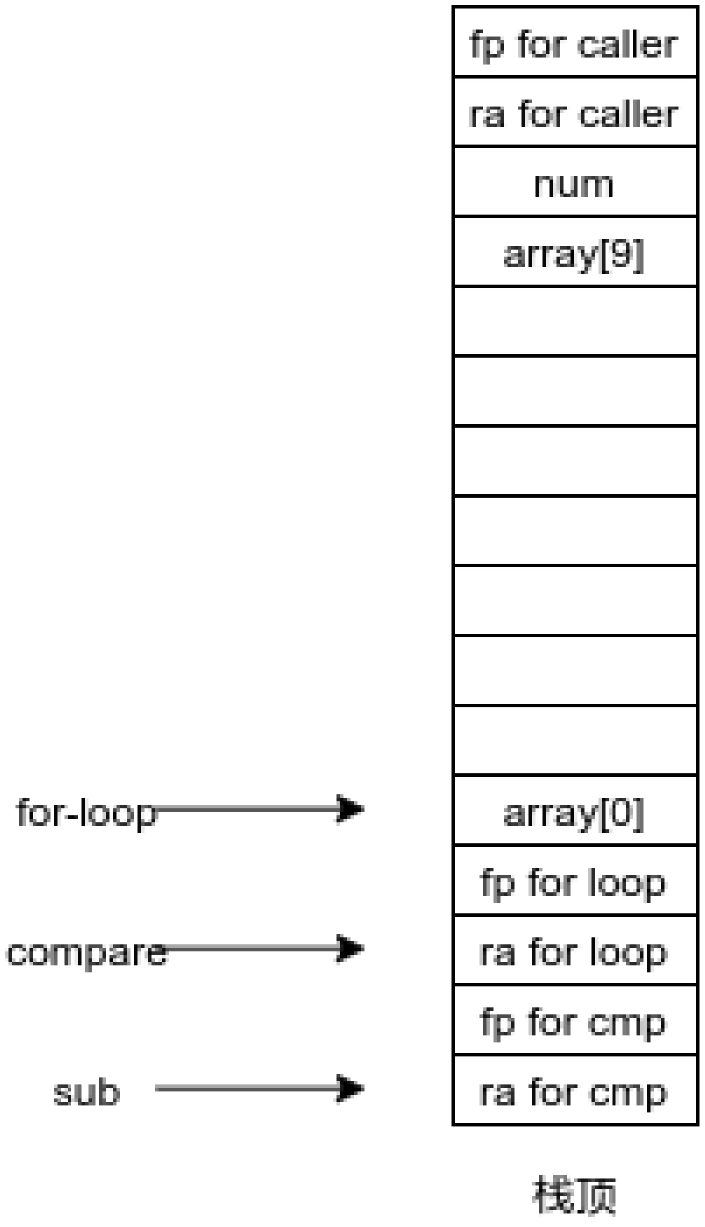

图 2: stack 示意图

Problem 2.16

```
# n stored in $a0

  .data
msg: .asciiz "the ans is "

  .text

  addi  $s0, $zero, 1
  # li  $a0, 7
  # la  $ra, prt
fib:
  addi  $sp, $sp, -12
  # push ra and n to stack
  sw  $ra, 8($sp)
  sw  $a0, 4($sp)
  # if n == 0, return 0;
  beq $a0, $zero, n0
  # if n == 1, return 1;
  beq $a0, $s0, n1
  # else fib(n-1)
  addi  $a0, $a0, -1
  jal fib
  # t0 = fib(n-1)
  add   $t0, $v0, $zero
  sw  $t0, ($sp)
  addi  $a0, $a0, -1
  jal fib
  # v0 = fib(n-1) + fib(n-2)
  # store the answer
  lw  $t0, ($sp)
  add   $v0, $t0, $v0
  # pop ra and n to stack
done: lw  $ra, 8($sp)
  lw  $a0, 4($sp)
  addi  $sp, $sp, 12

  # return to the caller
  jr  $ra




n0: add $v0, $zero, $zero
  j done
n1:   addi  $v0, $zero, 1
  j   done
prt:  add   $a0, $zero, $v0
  li      $v0, 1
  syscall
```

Problem 2.21

其 C 原型为

```
int str2int(char * str) {
    const char ascii_0 = '0';
    const char ascii_9 = '9';
    const int digit = 10;
    int ret = 0;

    while(1) {
    char now = *str;
    str++;
        if (now != '\0') {
            if (now <= ascii_9 && now >= ascii_0) {
                ret *= 10;
                ret += now - ascii_0;
            }
            else {
                return -1;
            }
        }else {
            return ret;
        }
    }
}
```

对此进行翻译

```
  .data
str:  .asciiz "1-3"

  .text
  la  $ra, end
  la  $a0, str
# j   puts
str2int:
  addi  $t0, $zero, 47 # 47 is the '0' - 1
  addi  $t1, $zero, 58 # 58 is the '9' + 1
  addi  $t2, $zero, 10 # set the number mult
  addi  $t4, $zero, 48 # 48 is the '0',
  add   $v0, $zero, $zero # set ret as 0
loop:
  lb  $t3, ($a0) # now = *str
  addi  $a0, $a0, 1 # str++
  beq   $t3, $zero, ret # if end, then return
  slt $t5, $t3, $t1 # if now <= '9'
  slt   $t6, $t0, $t3 # if '0' <= now
  and $t5, $t5, $t6 # and the condition
  bne $t5, 1, err # fail then return -1
  mul $v0, $t2, $v0 # ret *= 10
  sub $t3, $t3, $t4
  add $v0, $v0, $t3
  j loop

err:
  addi  $v0, $zero, -1
  j ret

ret:
  jr  $ra

end:
  add $a0, $v0, $zero
  j   puti

puti:
        li      $v0, 1           # specify Print Integer service
        syscall                  # Print it
        jr      $ra              # Return



puts:
        li      $v0, 4           # specify Print String service
        syscall                  # Print it
        jr      $ra              # Return
```

Problem 2.29

计算 a * b + 100 。

```
    add $t0, $zero, $zero # set $t0 to 0
loop: beq   $a1, $zero, finish # if b == 0, then goto finish
    add $t0, $t0, $a0 # else t0 += a
    sub $a1, $a1, 1 # b--
    j   loop
finish  addi  $t0, $t0, 100 # t0 += 100
    add $v0, $t0, $zero # return t0
```