計算機架構_CH2_HW1

B0529060 蘇筠雅

2.26 Consider the following MIPS loop:

```
LOOP: slt $t2, $0, $t1
      beq $t2, $0, DONE
      sub $t1, $t1, 1
      addi $s2, $s2, 2
      j     LOOP
DONE:
```

2.26.1Assume that the register $t1 is initialized to the value 10. What is the value in the register $s2 assuming $s2 is initially zero?

Ans: $s2 = 20

2.26.2For each of the loops above, write the equivalent C code routine. Assume that the registers $s1, $s2, $t1, and $t2 are integers A, B, i, and temp, respectively.

Ans:

```
while(i > 0){
    i= i-1
    B = B + 2
}
```

2.26.3 For the loops written in MIPS assembly above, assume that the reigster $t1 is initialized to the value N. How many MIPS instructions are executed?

Ans: 2 個

2.27Translate the following C code to MIPS assembly code. Use a minimum number of instructions. Assume that the values of a, b, i, and j are in registers ＄s0, ＄s1, ＄t0, and ＄t1, respectively. Also, assume that register ＄s2 holds the base address of the array D.

```
for(i=0; i<a; i++)
for(j=0; j<b; j++)
D[4*j] = i + j;
```

Ans：

```
        add $t0, $0, $0          # i = 0
L1:     slt  $t2, $t0, $s0       # i < a
             beq $t2, $0, Exit        # $t2 == 0, go to Exit
             add $t1, $0, $0          # j = 0
L2:     slt  $t2, $t1, $s1       # j < b
             beq $t2, $0, L3          # if $t2 == 0, go to L3
             add $t2, $t0, $t1        # i+j
             sll  $t4, $t1, 2         # $t4 = 4*j
             add $t3, $t4, $s2        # $t3 = &D[4*j]
             sw   $t2, 0($t3)         # D[4*j] = i+j
             addi $t1, $t1, 1         # j = j+1
             j    L2
L3:     addi $t0, $t0, 1         # i = i+1
             j    L1
Exit:
```

2.34 Translate function f into MIPS assembly language. If you need to use registers $t0 through $t7, use the lower numbered registers first. Assume the function declaration for func is "int func(int a, int b);" .The code for function f is as follows:

```
int f(int a, int b, int c, int d){
        return func(func(a, b), c + d);
}
```

Ans:

```
f:      addi    $sp, $sp, -12
        sw      $ra, 8($sp)
        sw      $s1, 4($sp)
        sw      $s0, 0($sp)
        move    $s1, $a2
        move    $s0, $a3
jal func
        move    $a0, $v0
        add     $a1, $s0, $s1
jal func
        lw      $ra, 8($sp)
        lw      $s1, 4($sp)
        lw      $s0, 0($sp)
        addi    $sp, $sp, 12
jr $ra
```