Computer Architecture and Programming Languages
Jacobs University Bremen
Fjolla Dedaj

Course: CO20-320241
4th of November, 2019

Homework 7

Problem 7.1

Solution:

x is stored in $a0
y is stored in $a1

my_function:

```
        slti $t0, $a0, 10        ⟹ If $a0 < 10, $t0 = 1, otherwise $t0 = 0
        bne $t0, $zero, ELSE     ⟹ If $t0 ! = 0, goto ELSE
        sub $s3, $a0, $a1        ⟹ $s3 = x - y
        add $v0, $s3, $zero      ⟹ store the value of $s3 as a return value in $v0
        jr $ra                   ⟹ jump to the address contained in register $ra
  ELSE:
        add $s4, $a0, $a1        ⟹ $s4 = x + y
        add $v1, $s4, $zero      ⟹ store the value of $s4 as a return value in $v1
        jr $ra                   ⟹ jump to the address contained in register $ra
```

Problem 7.2

Solution:

```
prod:
    addi $sp, $sp, -4
    sw $s1, 0($sp)
    mul $s1, $a0, $a1
    add $v0, $s1, $0
    lw $s1, $0($sp)
    addi $sp, $sp, 4
    jr   $ra
```

```
is_more_than_fifty:
     addi $sp, $sp, -4
     sw $s0, 0($sp)
     addi $t0, $0, 50
     jal prod
     slt $t1, $t0, $v0
     beq $t1, $t0, ELSE
     addi $so, $0, 1
     j    RETURN
ELSE:
     addi $s0, $0, 0
RETURN:
     add $v0, $s0, $0
     lw $s0, 0($sp)
     addi $sp, $sp, 4
     jr   $ra
```

$is\_more\_than\_fifty$ function is implemented in mips by first saving space for 1 variable and storing the value 50 in another register. Then the prod function is called and we check with slt if the condition holds. If it doesn't hold, we proceed to the ELSE instructions. The RETURN statement is implemented using the return value $v0 and jr (jumping back to the scope where the function was called).

The instructions in the $prod$ function save space for a variable first, then, the multiplication is done using mul and the return value is put into register $v0. Space is readjusted and the jump to the scope where the function was called is done using jr.

Problem 7.3
Solution:

\$s3 = i ; \$s5 = k

```
LOOP:
    slli $t1, $s3, 2            => $t1= 4 * i
    add $t1, $t1, $s6           => $t1 is the address of array[i]
    lw $t0, 0($t1)              => $t0=array[i]
    beq $t0, $s5, END           => skip to END if array[i] != k
    addi $s3, $s3, 1            => i = i + 1
    j     LOOP                  => go to LOOP

END:
```

In C code, that can be written in a more simplified way as:
$while(array[i]! = -1)$
$\quad i = i + 1$

Problem 7.4
Solution:

PC	MACHINE CODE	BINARY MACHINE CODE
6000	0 0 19 9 2 0	000000 00000 10011 01001 00010 000000
60004	0 9 22 9 0 32	000000 01001 10110 01001 00000 100000
60008	35 9 8 0	100011 01001 01000 0000000000000000
60012	4 8 21 2	000100 01000 10101 0000000000000010
60016	8 19 19 1	001000 10011 10011 0000000000000001
60020	2 15000	000010 00000000000011101010011000
60024		

Problem 7.5
Solution:

(i)	0x0C000000
(ii)	0xC4630000

What decimal number does the bit pattern represent:

a) if it is a two complement number?

(i) 0x0C000000 $=!(0000\ 1100\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000) + 1 =$
$= 1111\ 0011\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111 + 1 = 01111010000000000000000000000000 =$
$= -4093640704_{10}$

(ii) 0xC4630000 $=!(1100\ 0100\ 0110\ 0011\ 0000\ 0000\ 0000\ 0000) + 1 =$
$= 0011\ 1011\ 1001\ 1100\ 1111\ 1111\ 1111\ 1111 + 1 = 0011\ 1011\ 1001\ 1101\ 0000\ 0000\ 0000\ 0000 =$
-1000144896_{10}

b) if it is an unsigned integer?

(i) 0x0C000000 $= 0000\ 1100\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000 = 201326592_{10}$

(ii) 0xC4630000 $= 1100\ 0100\ 0110\ 0011\ 0000\ 0000\ 0000\ 0000 = 3294822400_{10}$

c) if this bit pattern is placed into the Instruction Register, what MIPS instruction would it be?

i) 0x0C000000 $=$ 000011 00000 00000 0000000000000000 $=$ JAL target
op
(ii) 0xC4630000 $=$ 11000 00011 00011 0000000000000000 $=$
= op code + base register + destination register + address =
= lwc1 \$3, 0(\$3)