

CSC3420 Computer System Architectures
2nd Term, Year 2008-2009 Quiz 2

Suggested Solution

Q1) MIPS: One-bit Counter

[23 marks]

```

add    $s1, $zero, $zero    # $s1 = 0;
add    $s2, $zero, $zero    # $s2 = 0;
addi   $s7, $zero, 1        # $s7 = 1;
Loop1: lw    $s3, int_arr($s2) # $s3 = int_arr[$s2];
      beq    $s3, $zero, Done  # while(int_arr[$s2] != 0) {
      addi   $s4, $zero, 1      #   $s4 = 1;
      addi   $s5, $zero, 32     #   $s5 = 32;
Loop2: beq    $s5, $zero, Cont  # while ($s5 != 0) {
      and    $s6, $s3, $s4      #   // $s6 = $s3 & $s4
      beq    $s6, $zero, Skip   #       if ($s3 & $s4) $s1++;
      add    $s1, $s1, $s7      #   // $s1++;
Skip:  sub    $s5, $s5, $s7     #       $s5--;
      sllv   $s4, $s4, $s7     #       $s4 <<= 1;
      j     Loop2              #   }
Cont:  addi   $s2, $s2, 4       #   $s2++;
      j     Loop1              # }
Done:                                     # // the answer is in $s1

```

- [10: general correct approach, i.e. try to count the bits for each integer
- 5: relevant instructions for important parts, e.g. shifting, masking
- 4: relevant instructions for loops, e.g. beq, j,...
- 4: overall correct solution]

Q2) Control Signal

[4*8 marks]

Op/Control	MemToReg	MemRead	MemWrite	JumpReg	Jump	Branch	JumpLink	ALUSrc	ALUOp	RegWrite
R-type	0	X	0	0	0	0	0	0	1	1
I-type	0	X	0	0	0	0	0	1	0	1
LW	1	1	0	0	0	0	0	1	0	1
SW	X	X	1	0	0	0	0	1	0	0
Branch	X	X	0	0	0	1	0	0	X	0
J	X	X	0	0	1	X	0	X	X	0
JR	X	X	0	1	X	1	0	X	X	0
JAL	X	X	0	0	1	X	1	X	X	0

[2 marks for correct '1's, 2marks for correct '0's, -1 mark for incorrect 'X's]

[-3 marks for 2 bit ALUOp]

Remark: RegWrite for JAL is 0. because if RegWrite equals 1, Reg[WriteReg] = WriteData

Q3) MIPS: Tower of Hanoi

[16 marks]

(a) Complete Code

```
void TOH(int n, char from, char dest, char by){
    if (n==1){
        printf("Move the plate from %c to %c \n", from, dest);
    }else{
        TOH(n-1, from, by, dest);
        TOH(1, from, dest, by);
        TOH(n-1, by, dest, from);
    }
}
```

(b) MIPS procedural call

[29 marks]

```
TOH:  addi  $sp, -4
      sw   $s0, 0($sp)      # save $s0 before using it
      addi $s0, $zero, 1    # compute a register with value = 1
      bne $a0, $s0, else    # if (n==1){
```

```
addi  $sp, -4
sw   $ra, 0($sp)
jalmy_print
lw   $ra, 0($sp)
addi  $sp, 4
```

printf("Move the plate from %c to %c \n", from, dest);

```
j  exit
```

```
else: addi  $sp, -28          #}else{
```

```
sw   $ra, 24($sp)
sw   $a3, 20($sp)
sw   $a2, 16($sp)
sw   $a1, 12($sp)
sw   $a0, 8($sp)
sw   $s1, 4($sp)
sw   $s2, 0($sp)
```

Save registers before calling another procedure

```
addi  $a0, $a0, -1
add $s1, $a2, $0
add $a2, $a3, $0
add $a3, $s1, $0
```

Prepare the four input arguments
n-1, from, by, dest

```
jal TOH          # TOH(n-1, from, by, dest);
add $s2, $a0, $0 # save n-1 to $s2
```

```

addi $a0, $0, 1
add $s1, $a2, $0
add $a2, $a3, $0
add $a3, $s1, $0
jal TOH

```

Prepare the four input arguments
1, from, dest, by

TOH(1, from, dest, by);

```

add $a0, $s2, $0
add $s1, $a1, $0
add $a1, $a3, $0
add $a3, $s1, $0

```

Prepare the four input arguments
n-1, by, dest, from

TOH(n-1, by, dest, from);

```

jal TOH
lw $ra, 24($sp)
lw $a3, 20($sp)
lw $a2, 16($sp)
lw $a1, 12($sp)
lw $a0, 8($sp)
lw $s1, 4($sp)
lw $s2, 0($sp)
addi $sp, 28

```

Recover registers after calling another procedure

```

exit: lw $s0, 0($sp)
      addi $sp, 4
      jr $ra

```

recover \$s0 after using it