Computer Organization

Chapter 2

2010/3/15

1. (10%)What hexadecimal number does this binary number represent:
   1100 1010 0000 1110 1111 1010 1100 1110_{two}?

Ans:CA0E FACE

2. (10%)What binary number does this hexadecimal number represent:
   0000 222a_{hex}? What decimal number does it represent?

Ans:

Binary form: 0000 0000 0000 0000 0010 0010 0010 1010

Decimal form: 8746

3. (15%)What is the compiled MIPS assembly code of the following C program?

```
for( i = 0; i < 10, i=i+1 )
    x[i] = i + k;
```

Assume that the base addresses for arrays `x` is found in $s0, `i` is in $s1, and k is in $s2.

```
        and $s1, $s1, $zero    # i = 0
        add $t1, $zero, 10     # $t1 = 10
Loop:   beq $s1, $t1, Exit     # if $s1 equal to $t1 jump to Exit
        add $t2, $s1, $s2      #use the $t2 to store the i + k result
        sll $t3, $s1, 2        #$t3 = 4*$s1
        add $t3, $t3 , $s0     #calculate the location of x[i]
        sw $t2, 0($t3)         #store the $t2 to the x[i]
        addi $s1, $s1, 1       # i++
        j Loop
Exit:
```

4. (20%)A *while* loop was compiled into this MIPS assembler code:

```
        sw      $t8,1000($s2)      # Store Temp reg $t8 to memory
Loop: sll       $t8,$s3,2          # Temp reg $t8 =4 * i
        add     $t8,$t8,$s1        # $t8 = address of save[i]
```

```
        lw      $t0,0($t8)          # Temp reg $t0 = save[i]
        bne     $t0,$s5, Exit       # go to Exit if save[i] ≠ k
        or      $t1, $t0, $s3       #$t1= $t0 OR $s3
        addi    $s3, $s3,16         # i = i + 16
        j       Loop                # go to loop
Exit:
```

If we assume that we place the code starting at location 40000 in memory, what is the MIPS machine code for this code segment? Please determine the instruction format for each instruction and the decimal values of each instruction field.

40000	43	18	24	1000			I
40004	0	0	19	24	2	0	R
40008	0	24	17	24	0	32	R
40012	35	24	8	0			I
40016	5	8	21	3			I
40020	0	8	19	9	0	37	R
40024	8	19	19	16			I
40028	2	10001					J
40032	……						

5. (15%)For each pseudoinstruction in the following table, produce a minimal sequence of actual MIPS instructions to accomplish the same thing.

Pseudoinstruction	What it accomplishes
ble $t3, $t5, L	If ($t3 <= $t5) go to L
bgt $t4, $t5, L	If ($t4 > $t5) go to L
bge $t5, $t3, L	If ($t5 >= $t3) go to L

Ans:

i.
```
slt $at, $t5, $t3
beq $at, $zero, L
```

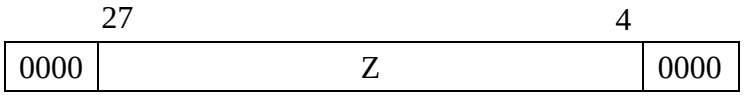
ii.
```
slt $at, $t5, $t4
bne $at, $zero, L
```

iii.
```
slt $at, $t5, $t3
beq $at, $zero, L
```

6. (15%)

OP $t0, $s0, $s1

After this instruction $t0 becomes



where the bit pattern in Z is the same as the bit pattern(27~4) of $s0.

a. Which instruction is suitable for the “OP”?

b. Please write down the content of $s1.

c. Please generate the content of $s1 by assembly language instructions.

Ans:

a. logical operation AND

b. 00001111111111111111111111110000

c. lui $s1, 4095
   ori $s1, $s1, 65520

7. (15%)What is the assembly language statement corresponding to the following machine
instructions?

a. 00AF8022

b. 32510064

c. 00128A80

Ans:

a. sub $s0, $a1, $t7

b. andi $s1, $s2, 100

c. sll $s1, $s2, 10

MIPS register conventions

Name	Register number	Usage	Preserved on call?
$zero	0	The constant value 0	n.a.
$v0-$v1	2-3	Values for results and expression evaluation	No
$a0-$a3	4-7	Arguments	No
$t0-$t7	8-15	Temporaries	No
$s0-$s7	16-23	Saved	Yes
$t8-$t9	24-25	More temporaries	No
$gp	28	Global pointer	Yes
$sp	29	Stack pointer	Yes
$fp	30	Frame pointer	Yes
$ra	31	Return address	Yes

MIPS instruction encoding

op(31:26)								
28-26	0(000)	1(001)	2(010)	3(011)	4(100)	5(101)	6(110)	7(111)
0(000)	R-format	Bltz/gez	jump	Jump&link	Branch eq	Branch ne	blez	bgtz
1(001)	Add immediate	addiu	Set less than imm.	sltiu	andi	ori	xori	Load upper imm
2(010)	TLB	FLPt						
3(011)								
4(100)	load byte	Load half	lwl	load word	lbu	lhu	lwr	
5(101)	store byte	Store half	swl	store word			swr	
6(110)	Lwc0	Lwc0						
7(111)	Swc0	Swc1						

op(31:26) = 010000(TLB),rs(25:21)								
23-21	0(000)	1(001)	2(010)	3(011)	4(100)	5(101)	6(110)	7(111)
0(00)	mfc0		cfc0		mtc0		ctc0	
1(01)								
2(10)								
3(11)								

op(31:26) =000000 (R-format),func(5:0)								
28-26	0(000)	1(001)	2(010)	3(011)	4(100)	5(101)	6(110)	7(111)
0(000)	Shift left logical		Shift right logical	sra	sllv		srlv	srav
1(001)	Jump reg.	jalr			syscall	Break		
2(010)	mfhi	mthi	mflo	Mtlo				
3(011)	Mult	multu	div	divu				
4(100)	add	addu	subtract	subu	and	or	xor	Not
5(101)			Set l. t.	sltu				
6(110)								
7(111)								