Computer Architecture

Fall, 2019

Week 4

2019.9.30

Group:

組員簽名：_____

[group2] (對抗賽)

Please translate the following C code into MIPS assembly code(i in $s1) :

1.

```
switch(i){
    case 1: i++;
            break;
    case 2: i+=2;
            break;
    case 3: i+=3;
            break;
}
```

Ans :

```
            addi $s4, $zero, 1
            bne $s1, $s4, C2_COND
            j C1_BODY
C2_COND:    addi $s4, $zero, 2
            bne $s1, $s4, C3_COND
            j C2_BODY
C3_COND:    addi $s4, $zero, 3
            bne $s1, $s4, EXIT
            j C3_BODY
C1_BODY:    addi $s1, $s1, 1
            j EXIT
C2_BODY:    addi $s1, $s1, 2
            j EXIT
C3_BODY:    addi $s1, $s1, 3
EXIT:
```

[group3] (對抗賽)

2. Please convert the following hexadecimal number to binary number, and(1)disassemble the machine code to assembly language in MIPS.(2)Which type is this instruction format?
   $2213FFA8_{16}$

Ans :

(1)

001000 10000 10011 1111111110101000

Addi $19 $16 -88

(2)

I-Format instruction

[group5]

3. 下列有關 MIPS instructions 何者正確？
   (A) R-format 與 I-format 的 rt field 是意義相同的
   (B) I-format 的 immediate field 可以表示$\pm 2^{15}$的數字，最左邊的 bit 如果是 1 的話代表是正數，0 為負數
   (C) I-format 進行 load 與 store 的時候，base register 永遠放在 rs field
   (D) shift right logical 補 0，而 shift right arithmetic 補 1

Ans: **(C)**

[group7] (對抗賽)

4. 關於 MIPS 的指令，以下敘述何者正確？如果敘述錯誤，請簡述原因。
   (A) MIPS 有“branch on less than”的指令
   (B) beq 跟 j 都屬於一種決策指令，只是前者是有條件的，後者是無條件的
   (C) beq 跟 j 都屬於 J-type instrution format.
   (D) R-type instrution format 中的 shamt field 有 6 bits.

Ans : (B)

[group11]

5. What is the MIPS machine language code for these three instruction?

```
lw   $t0, 1200($t1)
add   $t0, $s2, $t0
sw   $t0, 1200($t1)
```

Ans:

Binary:

OP	rs	rt	rd	shamt	funct
100011	01001	01000	0000 0100 1011 0000		
000000	10010	01000	01000	00000	100000
101011	01001	01000	0000 0100 1011 0000		

[group8] (對抗賽)

6. (1) 由於 MIPS 不提供 branch if greater and equal than(bge),但我們可以藉由使用 beq,bne,slt 等指令將其造出,則 bge $s1,$s2,L1 在 MIPS 中可以寫成?

Ans :

```
slt $t0,$s1,$s2
beq $t0,$zero,L1
```

(2) $t0 = 0x55555555 , $t1 = 0x12345678, 在經過以下順序的指令後 , $t2 的值為?

```
sll $t2,$t0,4
or $t2,$t2,$t1
```

Ans :

0x57755778

[group6] (對抗賽)

7. Translate the following loop into C. Assume that the C-level integer i is held in register $t1, $s2 holds the C-level integer called result, and $s0 holds the base address of the integer MemArray.

```
add $t1, $0, $0
LOOP : lw $s1, 0($s0)
        add $s2, $s2, $s1
        addi $s0, $s0, 4
        addi $t1, $t1, 1
        slti $t2, $t1, 100
        bne $t2, $zero, LOOP
```

Ans:

```
i = 0;
do{
    result+= MemArray[i];
    i++;
}while(i<100)
```

[group12] (對抗賽)

8. 請問下列敘述哪些有誤，若敘述有誤，請說明原因？

(a)MIPS 不存在 branch on less than

(b)lw 和 sw 屬於 I-format 的 instruction

(c)NOT a 可以視為 a NAND 0

(d)MIPS 的 3 個 shift instructions 分別為 sll, srl, sla

(e)不論是 lw 或 sw，base register 皆放在 rs field

A:

(C) a NOR 0

(D) sll, srl, sra，左移不需考慮 sign

[group4]

9. Please explain R-Format Instructions' fields

opcode	rs	rt	rd	shamt	funct

Ans :

opcode : partially specifies what instruction it is

rs : the first register source operand

rt : the second register source operand

rd : the destination operand which will receive result of computation

shamt : contains the amount a shift instruction will shift

funct : combined with opcode to specify the instruction

Instruction		Function
add	rd, rs, rt	100000
addu	rd, rs, rt	100001
and	rd, rs, rt	100100
break		001101
div	rs, rt	011010
divu	rs, rt	011011
jalr	rd, rs	001001
jr	rs	001000
mfhi	rd	010000
mflo	rd	010010
mthi	rs	010001
mtlo	rs	010011
mult	rs, rt	011000
multu	rs, rt	011001
nor	rd, rs, rt	100111
or	rd, rs, rt	100101
sll	rd, rt, sa	000000
sllv	rd, rt, rs	000100
slt	rd, rs, rt	101010
sltu	rd, rs, rt	101011
sra	rd, rt, sa	000011
srav	rd, rt, rs	000111
srl	rd, rt, sa	000010
srlv	rd, rt, rs	000110
sub	rd, rs, rt	100010
subu	rd, rs, rt	100011
syscall		001100
xor	rd, rs, rt	100110

Instruction		Opcode	Notes
addi	rt, rs, immediate	001000	
addiu	rt, rs, immediate	001001	
andi	rt, rs, immediate	001100	
beq	rs, rt, label	000100	
bgez	rs, label	000001	rt = 00001
bgtz	rs, label	000111	rt = 00000
blez	rs, label	000110	rt = 00000
bltz	rs, label	000001	rt = 00000
bne	rs, rt, label	000101	
lb	rt, immediate(rs)	100000	
lbu	rt, immediate(rs)	100100	
lh	rt, immediate(rs)	100001	
lhu	rt, immediate(rs)	100101	
lui	rt, immediate	001111	
lw	rt, immediate(rs)	100011	
lwc1	rt, immediate(rs)	110001	
ori	rt, rs, immediate	001101	
sb	rt, immediate(rs)	101000	
slti	rt, rs, immediate	001010	
sltiu	rt, rs, immediate	001011	
sh	rt, immediate(rs)	101001	
sw	rt, immediate(rs)	101011	
swc1	rt, immediate(rs)	111001	
xori	rt, rs, immediate	001110	