20200152 김규빈

2024-1 Computer Architecture Homework #2

Due: 4/19 (Fri) 11:59 p.m.

1. [25] Translate the following C code to MIPS. Assume that the variables f, g, h, i, and j are assigned to registers $s0, $s1, $s2, $s3, and $s4, respectively. Assume that the base address of the arrays A and B are in registers $s6 and $s7, respectively. Assume that the elements of the arrays A and B are 8-byte words:

```
B[8] = A[i] + A[j];
```

2. [10] Provide the type and assembly language instruction for the following binary value:0000 0010 0001 0000 1000 0000 0010 0000two . Hint: Figure 2.20 may be helpful.

3. [25] Consider a proposed new instruction named rpt. This instruction combines a loop's condition check and counter decrement into a single instruction. For example, rpt $s0, loop would do the following:

```
if (x29 > 0) {
        x29 = x29 − 1;
        goto loop
}
```

a. [10] If this instruction were to be implemented in the MIPS instruction set, what is the most appropriate instruction format?

b. [15] What is the shortest sequence of MIPS instructions that performs the same operation?

4. [20] 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.

```
        addi $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, $s0, LOOP
```

5. [20] For the following code:

```
lbu $t0, 0($t1)
sw $t0, 0($t2)
```

Assume that the register $t1 contains the address 0x10000000 and the data at address is 0x11223344.

a. [10] What value is stored in 0x10000004 on a big-endian machine?

b. [10] What value is stored in 0x10000004 on a lile-endian machine?

1. [25] Translate the following C code to MIPS. Assume that the variables f, g, h, i, and j are assigned to registers $s0, $s1, $s2, $s3, and $s4, respectively. Assume that the base address of the arrays A and B are in registers $s6 and $s7, respectively. Assume that the elements of the arrays A and B are 8-byte words:

```
B[8] = A[i] + A[j];
```

1.

```
sll $t0, $s3, 3    # $t0 = i * 8
sll $t1, $s4, 3    # $t1 = j * 8
add $t0, $t0, $s6  # $t0 = &A[i]
add $t1, $t1, $s7  # $t1 = &B[j]
lw $t2, 0($t0)     # $t2 = A[i]
lw $t3, 0($t1)     # $t3 = B[j]
add $t4, $t2, $t3  # $t4 = A[i] + B[j]
sw $t4, 64($s7)    # B[8] = A[i] + B[j]
```

2. [10] Provide the type and assembly language instruction for the following binary value:0000 0010 0001 0000 1000 0000 0010 0000two . Hint: Figure 2.20 may be helpful.

2. 31-29 bits: 000, 28-26 bits: 000

∴ The format of instruction is R-format instruction

5-3 bits: 000, 2-0 bits: 000

∴ This instruction is add

Register number of $s0 is 16.

R-format	Opcode	rs	rt	rd	shamt	func
	000000	10000	10000	10000	000000	100000
	R-format	16	16	16	0	add
	R-format	$s0	$s0	$s0	0	add

∴ add $s0, $s0, $s0

3. [25] Consider a proposed new instruction named rpt. This instruction combines a loop's condition check and counter decrement into a single instruction. For example, rpt $s0, loop would do the following:

```
if (x29 > 0) {
        x29 = x29 − 1;
        goto loop
}
```

a. [10] If this instruction were to be implemented in the MIPS instruction set, what is the most appropriate instruction format?

a. rpt $s0, loop

We need
: 1. a register operand ($s0)
2. a target address to jump (loop)

∴ I-format is appropriate format for rpt.
I-format can use a register operand.
I-format can use an address (offset added to base address)

opcode (6bits): rpt
rs (5bits): $s0
rt (5bits): used for special settings
constant or address (16 bits): offset for the branch.

∴ I-format is the most appropriate instruction format.

b. [15] What is the shortest sequence of MIPS instructions that performs the same operation?

```
b.  bgtz $s0, L1        # if(x29 > 0)
L1: addi $s0, $s0, -1   # x29 = x29 - 1
    j loop              # goto loop
```

4. [20] 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.

```
        addi $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, $s0, LOOP
```

$zero

4.
```
addi $t1, $0, 0   # i = 0
LOOP: lw $s1, 0($s0)   # Load the current array index into $s1
      add $s2, $s2, $s1   # result += MemArray[i]
      addi $s0, $s0, 4   # Move to the next array element
      addi $t1, $t1, 1   # i++
      slti $t2, $t1, 100   # if i<100 set $t2 to 1, else set $s2 to 0
      bne $t2, $zero, LOOP   # if ($t2 != 0) go to LOOP
```

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

5. [20] For the following code:

lbu $t0, 0($t1)

sw $t0, 0($t2)

Assume that the register $t1 contains the address 0x10000000 and the data at address is 0x11223344.

a. [10] What value is stored in 0x10000004 on a big-endian machine?

a.

In a big-endian system, MSB is stored at the lowest address.

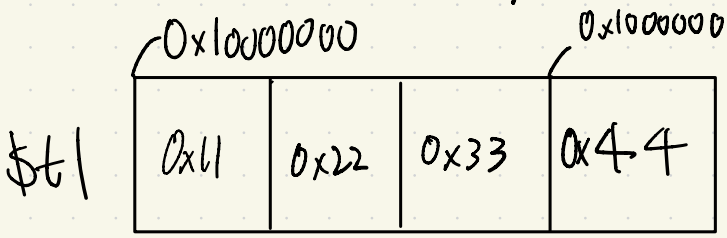
lbu $t0, 0($t1): The data at address 0x10000000 is 0x11223344.

On a big-endian machine, MSB is stored at the lowest address.

∴ the byte loaded into $t0 is 0x11

sw $t0, 0($t2): store the value in $t0 (0x11) at 0($t2).

∴ 0x10000000 : 0x11
0x10000001 : 0x22
0x10000002 : 0x33
0x10000003 : 0x44



∴ Big-endian: $t2 stored 0x00000011

∴ 0x00 is stored in 0x10000004.

b. [10] What value is stored in 0x10000004 on a lile-endian machine?

b.

In a little-endian machine, LSB is stored at the lowest memory address

For the address 0x10000000 holding the value 0x11223344.

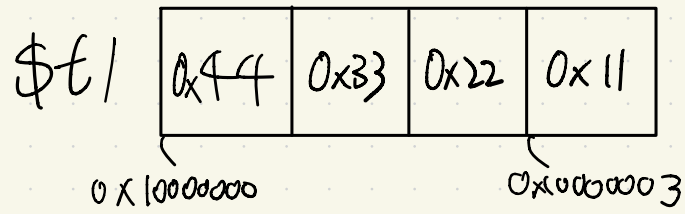
lbu $t0, 0($t1): The data at address 0x10000000 is 0x11223344.

On a little-endian machine, LSB is stored at the lowest address.

∴ the byte loaded into $t0 is 0x44

sw $t0, 0($t2): store the value in $t0 (0x44) at 0($t2).

∴ 0x10000000 : 0x44
0x10000001 : 0x33
0x10000002 : 0x22
0x10000003 : 0x11



∴ Little endian : $t2 stored 0x00000044

∴ 0x44 is stored in 0x10000004