Computer Architecture

Fall, 2022

Week 4

2022.10.03

組別：_____ 簽名：_____

_____

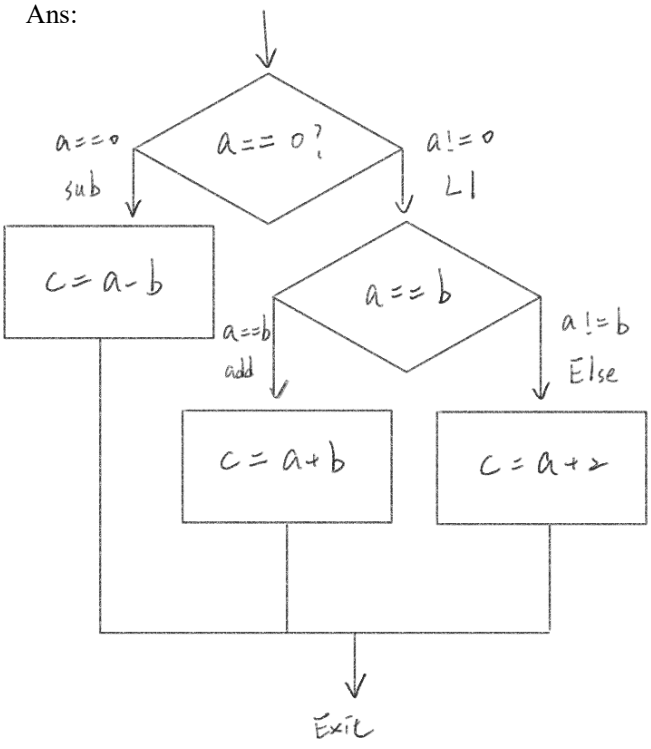
[group2]

Please draw a flowchart of the following MIPS code, and convert it to C language.

a = $s0, b = $s1, c = $s2

```
bne   $s0, $zero, L1
sub   $s2, $s0, $s1
j     Exit
L1:   bne   $s0, $s1, Else
      add   $s2, $s0, $s1
      j     Exit
Else: addi  $s2, $s0, 2
Exit:
```

Ans:




```c
if(a!=0){
  if(a!=b){
      c = a +2;
      return;
  }
  else{
      c = a + b;
      return;
  }
}
else{
  c = a - b;
  return;
}
```

[group10]

True or false:

(a) In R-format instructions, opcode are always the same for different instructions.

(b) If we shift right arithmetic by 4 bits for the following code:

0101 0010 0011 0001

After operation, we will get:

0001 0101 0010 0011

(c) Shamt field is only 5 bits because shifting a 32-bits word by more than 31 is useless.

(d) rt field in I-format instructions is source register when instruction is lw and destination register when instruction is sw.

(e) sll by i bits multiplies by 2^i and srl by i bits divides by 2^i, i bits are signed number.

(f) MIPS has “and”, “or” and “not” three instructions.

Ans:

(a) T

(b) F. fills empties by sign extending

(c) T

(d) F, rt is not a register oprand ⇒ in the formmer description, rt should be target register

(e) F, it is unsigned number

(f) F, "and", "or", "nor"

[group7]

Explain why the following descriptions are INCORRECT?

1) A good compiler will compile $t0 = $t1 / 4; into srl $t0, $t1, 2 if $t0 and $t1 are signed integers.

2) The R-type instruction sub $s0, $t0, $t1 can be represented as:

opcode	rs	rt	rd	shamt	funct
sub	$t0	$t1	$s0	0	0

3) Consider the following code:

```
if ($s1 == 0) $s0 = $t0 - $t1;
else $s0 = $t1 * 8;
```

It has been compiled into MIPS instructions as shown below. We can remove line 3 since it's a redundant instruction produced by the compiler.

```
1        beq $s1, $0, Equl
2        sll $s0, $t1, 3
3        j   Exit
4 Equl: sub $s0, $t0, $t1
5 Exit:
```

Ans: (1) If $t1 is negative, when we perform srl on $t1 the LHS bit will be filled by zero. Which results $t1 not negative anymore.

(2) The funct of sub instruction is "100010", opcode = 000000

(3) If we remove line 3, when we enter the statement of "$s1 = 0" it would contine process after line 2, rather than Exit.

[group9]

True/False questions: Which of the following statement are True? Please provide the reason for the False statements.

(1) The opcode in R-format and I-format are similar in number of bits (6 bits), and its integer value which is equal to 0 for all instructions.

(2) Shifting a 32-bit word by more than 31 is useless, so the field shamt is only 5 bits.

(3) Shift right arithmetic by 8 bits:

1100 0001 0010 0100 0111 0010 0111 1000 -> 0000 0000 1100 0001 0010 0100 0111 0010

(4) The hexadecimal strings of the following binary strings:

1100 0001 0010 0100 0111 0010 0111 1000 is c124 7278

(5) In I-format, the field immediate has 16 bits -> can be used to represent immediate up to 2^{16} different values.

(6) Fred constructs a MIPS code using the NOT operations nor $t0, $t1, $zero then he gets the answer for the register $t0 as below:

$t1 0100 1010 0000 0110 0011 1000 0011 1010

$t0 1011 0101 1111 1001 1000 0111 1101 0101

NOT ($t1 OR 0)
= NOT $t1

(7) Each register field is exactly 5 bits, which means that it can specify any unsigned integer in the range 1-32.

(8) We have a C code multiplies by a power of 2 as x *= 512; and compiles it to a shift left instruction in MIPS as sll $s0, $s0, 7

Ans:

(1) The opcode of I-format insturction is not equal to 0 for all.

(2) T

(3) Perform sr| 8 bits on this number, it will be filled by 1s in the LHS.

(4) T

(5) T

(6)

$t1 0100 1010 0000 0110 0011 1000 0011 1010

$t0 1011 0101 1111 1001 1000 0111 1100 0101

(7) [0, 31]

(8) sll $s0, $s0, 9

[group13]

Which statements are correct ?

(a) rs ,rt ,rd and shamt only have 5bits because registers are 32 bits

(b) In logical operations, we would perform NOT a by a NOR b(b is $zero)

(c) A good complier would use multiplication than shifting since shifting is slower.

(d) Design principle 4 is simplicity favors regularity

Ans:

(a) F

Each register field(rs,rt,rd and shamt) is exactly 5 bits,
which means that it can specify any unsigned integer in the range 0-31.
Each of these fields specifies one of the 32 registers by number.

(b) T

(c) F. shift is faster

(d) F. Good design demands good compromises.

[group3]

Multiple choice question (may have two or more correct choice)

1. MIPS only has three instruction formats, so it's not as functional as other ISAs

2.In MIPS, we can use only opcode field to identify what the instruction is doing except R-format instruction.

3.To identify if a number is even, we can use [and] instruction to do it in MIPS.

4.Let $s0 = -1023 , then these two instructions{srl $t0 $s0 4, sra $t0 $s0 4 } will output the same outcome.

Ans:

1. It may be functional to others.

4. it will be a postive number

2, 3 #

[group12]

Write a program to count the number of bits that are set to 1 in a nonnegative integer in MIPS. The number is stored in Memory M[0], and the result should restore into the Memory M[1]. The base address of the Memory is stored in the register $s0, $t1 is clear to 0 for the variable temp_bit, and $s1 is clear to zero for the variable num_bits.

(Hint: Translate the following C program into MIPS)

```
int num_bits = 0;
int temp_bit = 0;
while(x) {
        temp_bit = x & 1;
        num_bits += temp;
        x >>= 1;
}
return num_bits;
```

$s0 ⇒ M[0]
$t1 ⇒ temp_bit
$s1 ⇒ num_bit

Ans:

```
1 lw $t0, 0(s0)
2 LOOP:   beq $t0, $zero, SAVE
3         andi $t1, $t0, 1
4         add $s1, $s1, $t1
5         srl $t0, $t0, 1
6         j   LOOP
7 SAVE:   sw $s2, 4(s0)
8 EXIT:
```