VE370 HW1 5193709100847 Wang Lan

1. (5 points) For the following C statement, write the corresponding RISC-V assembly code. Assume that the C variables f, g, and h, have already been placed in registers x5, x6, and x7 respectively. Use a minimal number of RISC-V assembly instructions.

```
f = g + (h - 5);
```

```
addi x7, x7, -5
add  x5, x6, x7
```

2. (5 points) For the following C statement, write the corresponding RISC-V assembly code. Assume that the variables f, g, h, i, and j are assigned to registers x5, x6, x7, x28, and x29, respectively. Assume that the base address of the arrays A and B are in registers x10 and x11, respectively.

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

```
sub  x5, x7, x28
slli x6, x5, 2
add  x6, x6, x10
lw   x5, (0)x6
sw   x5, (32)x11
```

3. (10 points) For the RISC-V assembly instructions below, what is the corresponding C statement? Assume that the variables f, g, h, i, and j are assigned to registers x5, x6, x7, x28, and x29, respectively. Assume that the base address of the arrays A and B are in registers x10 and x11, respectively.

```
slli x30, x5, 2
add  x30, x10, x30
slli x31, x6, 2
add  x31, x11, x31
lw   x5, 0(x30)
addi x12, x30, 4
lw   x30, 0(x12)
add  x30, x30, x5
sw   x30, 0(x31)
```

```
B[g] = A[f+1] + A[f];
```

4. (5 points) Show how the value `0xabcdef12` would be arranged in memory of a little-endian and a big-endian machine. Assume the data are stored starting at word address 0.

	0x00000000	0x00000001	0x00000002	0x00000003
little endian:	12	ef	cd	ab
big endian:	ab	cd	ef	12

5. (5 points) Find the shortest sequence of RISC-V instructions that extracts bits 16 down to 11 from register x5 and uses the value of this field to replace bits 31 down to 26 in register x6 without changing the other bits of registers x5 or x6. (Be sure to test your code using x5 = 0 and x6 = `0xffffffff`. Doing so may reveal a common oversight.)

```
addi x28, x0, 0x3f
slli x28, x28, 11
slli x29, x28, 15
and  x30, x5, x28
slli x30, x30, 15
xori x29, x30, -1
and  x6, x6, x29
add  x6, x6, x30
```

6. (10 points) Assume x5 holds the value 0x01010000. What is the value of x6 after the following instructions?

```
        bge x5, x0, ELSE
        jal x0, DONE
ELSE:   ori x6, x0, 2
DONE:   ......
```

0x00000002

7. Consider the following RISC-V loop:

```
LOOP: beq x6, x0, DONE
      addi x6, x6, -1
      addi x5, x5, 2
      jal x0, LOOP
DONE: ......
```

(1) (10 points) Assume that the register x6 is initialized to the value 10. What is the final value in register x5 assuming the x5 is initially zero?

(2) (10 points) For the loop above, write the equivalent C code. Assume that the registers x5 and x6 are integers acc and i, respectively.

(3) (5 points) For the loop written in RISC-V assembly above, assume that the register x6 is initialized to the value N. How many RISC-V instructions are executed?

(4) (5 points) For the loop written in RISC-V assembly above, replace the instruction "`beq x6, x0, DONE`" with the instruction "`blt x6, x0, DONE`" and write the equivalent C code.

(1) LOOP is executed 10 times. x5 is 20 finally.

(2)
```
while (i != 0) {
    i = i-1;
    acc = acc+2;
}
```

(3) LOOP will be executed N times ⇒ 4N

add the last judgement ⇒ 4N+1

(4)
```
while (i >= 0) {
    i = i-1;
    acc = acc+2;
}
```

8. (20 points) Translate function f into RISC-V assembly language. Assume the function declaration for `g` is `int g(int a, int b)`. The code for function f is as follows:

```
int f(int a, int b, int c, int d){
   return g(g(a,b), c+d);
}
```

(a: x11, b: x12, c: x13, d: x14)

return value of f and g: x10

```
f: addi sp, sp, -8
   sw   x1, 4(sp)
   add  x5, x13, x14   // x5 = c+d
   sw   x5, 0(sp)
   jal  x1, g          // x10 = g(a,b)
   lw   x12, 0(sp)
   addi x11, x10, 0    // x11 = x10
   jal  x1, g          // x10 = g(g(a,b), c+d)
   lw   x1, 4(sp)
   addi sp, sp, 8
   jalr x0, 0(x1)
```

9. (10 points) Right before your function `f` from Problem 8 returns, what do we know about contents of registers `x10-x14`, `x8`, `x1`, and `sp`? Keep in mind that we know what the entire function `f` looks like, but for function `g` we only know its declaration.

x10 = g(g(a,b), c+d)
x11 = g(a,b)
x12 = c+d
x13 = c
x14 = d
x8 = 0
x1 = the address of the next line after where calling f.
sp = 0x7fff fffc (top of empty heap)