Homework 1

VE370 - Intro to Computer Organization Summer 2022

* Name: Huang Yucheng ID: 519021910885

Exercise 1

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 `x28`, `x29`, and `x30` respectively. Use a minimal number of RISC-V assembly instructions.

```
f = g + ( h - 9 ) ;
```

Answer:

```
addi x28, x30, -9
add x28, x28, x29
```

Exercise 2

Translate the following C code to RISC-V. 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. Assume that the elements of the arrays `A` and `B` are 4-byte words:

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

Answer:

```
slli x28, x28, 2
lw x28, 0(x10)
slli x29, x29, 2
lw x29, 0(x10)
add x29, x28, x29
sw x29, 32(x11)
```

Exercise 3

Translate the following loop into C. Assume that the C-level integer `i` is held in register `x5`, `x6` holds the *C*-level integer called `result`, and `x10` holds the base address of the integer `MemArray`.

```
     addi x6, x0, 0
     addi x29, x0, 100
LOOP:lw x7, 0(x10)
     add x5, x5, x7
     addi x10, x10, 4
     addi x6, x6, 1
     blt x6, x29, LOOP
```

Answer:

```
int i = 0;
while (i < 100)
{
     result = result + Memarray[i];
     i++;
}
return result;
```

Exercise 4

Show how the value `0x12345678` would be arranged in memory of a little-endian and a big-endian machine. Assume the data are stored starting at word address 0 .

Answer:
Since it is a 32-bit number, we have

Big-Endian		Little-Endian	
Address	Word	Address	Word
0xffff_0000	12	0xffff_0000	78
0xffff_0001	34	0xffff_0001	56
0xffff_0002	56	0xffff_0002	34
0xffff_0003	78	0xffff_0003	12

Exercise 5

Assume the following register contents: `x5 = 0x0000AAAA`, `x6 = 0x12345678`

a. For the register values shown above, what is the value of `x7` for the following sequence of instructions?

```
slli x7, x5, 4
or x7, x7, x6
```

Answer:
First, we shift `x5` to the left for 4 bits. `x7 = 0x000AAAA0`. Next, we use `or` function and finally we can get that
`x7 = 0x123EFEF8`

b. For the register values shown above, what is the value of `x7` for the following sequence of instructions?

```
srli x7, x5, 3
andi x7, x7, 0xFEF
```

Answer:
First, we shift `x5` to the right for 3 bits. `x7 = 0x00001555`. Next, we use `andi` function and finally we can get that
`x7 = 0x545`

Exercise 6

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:   ......
```

Answer:
`x6 = 2`

Exercise 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) 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) For the loop above, write the equivalent C code. Assume that the registers `x5` and `x6` are integers `acc` and `i`, respectively.

(3) 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) 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.

Answer:

(1) The final value of `x5` should be 20.

(2)

```
int i = 10, acc = 0;
while (i != 0)
{
     acc = acc + 2;
     i--;
}
```

(3) In each loop, it executed 4 instructions. Before the last judgement, `x6` executed totally 4N instructions. Add the final judgement, it totally cost 4N+1 instructions.

(4)

```
int i = 10, acc = 0;
while (i >= 0)
{
     acc = acc + 2;
     i--;
}
```

Exercise 8

Translate the following C code to RISC-V assembly code. Use a minimum number of instructions. Assume that the values of `a`, `b`, `i`, and `j` are in registers `x5`, `x6`, `x7`, and `x29`, respectively. Also, assume that register `x10` holds the base address of the array D.

```
for(i=0; i<a; i++)
    for(j=0; j<b; j++)
        D[4*j] = i + j
```

Answer:

```
LOOPOUT: addi x7, x0, 0
         bge x7, x5, BREAK
 LOOPIN: addi x29, x0, 0
         bge x29, x6, CONTINUE
         add x1, x7, x29
         addi x2, x10, 0
         sw x1, 0(x2)
         addi x2, x2, 16
         addi x29, x29, 1
         jal x0, LOOPIN
CONTINUE: addi x7, x7, 1
         jal x0, LOOPOUT
   BREAK: ......
```