

## Chapter 2 Questions - Assignment Questions for Week 2

- 2.1 ;§2.2;** 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);
```

- 2.2 ;§2.2;** Write a single C statement that corresponds to the two RISC-V assembly instructions below.

```
add f, g, h
add f, i, f
```

- 2.3 ;§§2.2, 2.3;** 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];
```

- 2.4 ;§§2.2, 2.3;** 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, 3 // x30 = f*8
add x30, x10, x30 // x30 = &A[f]
slli x31, x6, 3 // x31 = g*8
add x31, x11, x31 // x31 = &B[g]
ld x5, 0(x30) // f = A[f]
addi x12, x30, 8
ld x30, 0(x12)
add x30, x30, x5
sd x30, 0(x31)
```

- 2.5 ;§2.3;** Show how the value `0xabcdef12` would be arranged in memory of a little-endian and a big-endian machine. Assume the data is stored starting at address 0 and that the word size is 4 bytes.

- 2.7 ;§§2.2, 2.3;** 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 8-byte words:

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

**2.10** Assume that registers **x5** and **x6** hold the values **0x8000000000000000** and **0xD000000000000000**.

**2.10.1** What is the value of **x30** for the following assembly code?

```
add x30, x5, x6
```

**2.10.2** Is the result in **x30** the desired result, or has there been overflow?

**2.10.3** For the contents of registers **x5** and **x6** as specified above, what is the value of **x30** for the following assembly code?

```
sub x30, x5, x6
```

**2.10.4** Is the result in **x30** the desired result, or has there been overflow?

**2.10.5** For the contents of registers **x5** and **x6** as specified above, what is the value of **x30** for the following assembly code?

```
add x30, x5, x6
add x30, x30, x5
```

**2.10.6** Is the result in **x30** the desired result, or has there been overflow?

**2.12** Provide the instruction type and assembly language instruction for the following binary value (Hint: Figure 2.20 may be helpful):

```
0000 0000 0001 0000 1000 0000 1011 0011 (two)
```

**2.13** Provide the instruction type and hexadecimal representation of the following instruction:

```
sd x5, 32(x30)
```