Chapter 2

Exercises

2.1 [5] <§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 [5] <§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 [5] <§§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];
```

2.5 [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 are stored starting at address 0 and that the word size is 4 bytes.

2.11 Assume that `x5` holds the value 128_{ten}.

2.11.1 [5] <§2.4> For the instruction `add x30, x5, x6`, what is the range(s) of values for `x6` that would result in overflow?

2.11.2 [5] <§2.4> For the instruction `sub x30, x5, x6`, what is the range(s) of values for `x6` that would result in overflow?

2.11.3 [5] <§2.4> For the instruction `sub x30, x6, x5`, what is the range(s) of values for `x6` that would result in overflow?

2.19 [5] <§2.6> Provide a minimal set of RISC-V instructions that may be used to implement the following pseudoinstruction:

```
    not x5, x6      // bit-wise invert
```

2.25 [10] <§2.7> 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;
```