Chapter 2 Exercises

2.3 [5] ⟨2.2⟩ 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];
```

RISC-V Translation:

```
sub x5 x28, x29      # f = i - j
slli x5, x5, 2       # f = f << 2      (f = f * 4)
add x6, x10, x5      # g = &A[i-j]
lw x6, 0(x6)         # g = A[i-j]
sw x6, 32(x11)       # B[8] = A[i-j]   (8 * 4 = 32)
```

2.7 [5] ⟨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];
```

RISC-V Translation:

```
slli x28, x28, 2     # i = i << 2
add x5, x10, x28     # f = &A[i]
lw x5, 0(x5)         # f = A[i]
slli x29, x29, 2     # j = j << 2
add x6, x10, x29     # g = &A[j]
lw x6. 0(x6)         # g = A[j]
add x7, x5, x6       # h = f + g
sw x7, 32(x11)       # B[8] = h
```

2.13 [5] ⟨2.5⟩ Provide the instruction type and hexadecimal representation for the following instruction:

```
sw x5, 32(x30)
```

Instruction format S.

$0000001\ 00101\ 11110\ 010\ 00000\ 0100011_{\text{two}} \Rightarrow 25F2023_{\text{hex}}$

2.18 [10] ⟨2.6⟩ 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 = 0xffffffffffffffff`. Doing so may reveal a common oversight.)

```
andi x7, x5, 0x1F800          # Extract bits 16 to 11
slli x7, x7, 15               # Shift bits 16 to 11 to position 31 to 26
andi x6, x6, 0x3FFFFFF        # Clear bits 31 to 26
or x6, x6, x7                 # Replace bits
```