HW4.3. RISC-V Bit Manipulation and Endianness

Feel free to check out the guide that we have prepared to help you in this problem.

1: What does the following code do? Feel free to test random numbers on the two registers.

```
xor x1, x1, x2
xor x2, x1, x2
xor x1, x1, x2
```

Q1.1:

- (a) Sets `x1` to `x1 & x2`, and sets `x2` to `x1 | x2`
- (b) Sets `x1` to `x1 ^ x2`, and sets `x2` to `~(x1 ^ x2)`
- (c) Sets `x1` to `~(x1 ^ x2)`, and sets `x2` to `x1 ^ x2`
- (d) Sets `x2` to -1
- (e) Sets `x1 = x2`
- (f) Sets `x1` to 0
- (g) Sets `x2` to 0
- (h) Sets `x1` to `x1 | x2`, and sets `x2` to `x1 & x2`
- (i) Swaps `x1` and `x2` ✓
- (j) Sets `x1` to -1

✓ 100%

2: After the following code is run, what is contained in `x12`? Answer with 8 digits of hexadecimal.

Hint: Note that RISC-V registers contain 32-bit data. `srai` performs a sign-extended right shift.

```
li x10, 0x84FF
slli   x12, x10, 0x10
srai   x13, x12, 0x10
srli   x14, x12, 0x08
and    x12, x13, x14
```

Q2.1: x12=0x 00848400 ? ✓ 100%

3: Given the following code sequence (assume `x5` is somewhere on the stack):

```
addi x11, x0, -1261
sw x11, 0(x5)
sh x11, 2(x5)
lb x12, 1(x5)
lbu x13, 2(x5)
```

What is the 32-bit value stored in `x12` and `x13`? Answer with 8 digits of hexadecimal.

Hint: RISC-V uses little endian. Take note of sign-extension when doing load instructions.

Q3.1: x12=0x FFFFFFFB ? ✓ 100%

Q3.2: x13=0x 00000013 ? ✓ 100%

Try a new variant

Correct answer

Feel free to check out the guide that we have prepared to help you in this problem.

1: What does the following code do? Feel free to test random numbers on the two registers.

```
xor x1, x1, x2
xor x2, x1, x2
xor x1, x1, x2
```

Q1.1:
(i) Swaps `x1` and `x2`

This is called the XOR swap algorithm. Feel free to look it up!

2: After the following code is run, what is contained in `x12`? Answer with 8 digits of hexadecimal.

Hint: Note that RISC-V registers contain 32-bit data. `srai` *performs a sign-extended right shift.*

```
li x10, 0x84FF
slli   x12, x10, 0x10
srai   x13, x12, 0x10
srli   x14, x12, 0x08
and    x12, x13, x14
```

Q2.1: x12=0x `00848400`

`slli x12, x10, 0x10` shifts the value at x10 16 times to the left. **x10 = 0x84FF0000**

`srai x13, x12, 0x10` shifts the value at x12 16 times to the right (with sign-extension). **x13 = 0xFFFF84FF**

`srli x14, x12, 0x08` shifts the value at x12 8 times to the right (without sign-extension). **x14 = 0x0084FF00**

`and x12, x13, x14` bitwise AND of x13 and x14. **x12 = 0x00848400**

3: Given the following code sequence (assume `x5` is somewhere on the stack):

```
addi x11, x0, -1261
sw x11, 0(x5)
sh x11, 2(x5)
lb x12, 1(x5)
lbu x13, 2(x5)
```

What is the 32-bit value stored in `x12` and `x13`? Answer with 8 digits of hexadecimal.

Hint: RISC-V uses little endian. Take note of sign-extension when doing load instructions.

Q3.1: x12=0x `FFFFFFFB`
Q3.2: x13=0x `00000013`

`addi x11, x0, -1261` sets **x11 = 0xFFFFFB13**

`sw x11, 0(x5)` stores x11 to memory address pointed to by x5. **FF FF FB 13** in memory (assuming B3:B2:B1:B0 byte ordering)

`sh x11, 2(x5)` stores least significant half of x11 to memory address pointed to by x5 + 2. **FB 13 FB 13** now in memory.

`lb x12, 1(x5)` loads a byte (signed) from memory address pointed to by x5 + 1. Byte 1 is 0xFB. Sign-extend since instruction is not unsigned. **x12 = 0xFFFFFFFB**

`lbu x13, 2(x5)` loads a byte (unsigned) from memory address pointed to by x5 + 2. Byte 2 is 0x13. No sign-extension needed (will not affect result anyway). **x13 = 0x00000013**

Submitted answer 2 **correct: 100%**
Submitted at 2022-09-20 23:47:56 (PDT)

hide

Feel free to check out the guide that we have prepared to help you in this problem.

1: What does the following code do? Feel free to test random numbers on the two registers.

```
xor x1, x1, x2
xor x2, x1, x2
xor x1, x1, x2
```

Q1.1:
(i) Swaps `x1` and `x2` ✓100%

2: After the following code is run, what is contained in `x12`? Answer with 8 digits of hexadecimal.

Hint: Note that RISC-V registers contain 32-bit data. `srai` *performs a sign-extended right shift.*

```
li x10, 0x84FF
slli   x12, x10, 0x10
srai   x13, x12, 0x10
srli   x14, x12, 0x08
and    x12, x13, x14
```

Q2.1: x12=0x `00848400` ✓100%

3: Given the following code sequence (assume `x5` is somewhere on the stack):

```
addi x11, x0, -1261
sw x11, 0(x5)
sh x11, 2(x5)
lb x12, 1(x5)
lbu x13, 2(x5)
```

What is the 32-bit value stored in `x12` and `x13`? Answer with 8 digits of hexadecimal.

Hint: RISC-V uses little endian. Take note of sign-extension when doing load instructions.

Q3.1: x12=0x `FFFFFFFB` ✓100%
Q3.2: x13=0x `00000013` ✓100%

Submitted answer 1 **partially correct: 25%**
Submitted at 2022-09-20 23:38:34 (PDT)

show