EECS 112 & CSE 132, FALL 2017

Homework 2

Due date: October, 26, 2017

Student ID: 7 7 9 7 0 3 8 Name: Michael Chai

1) Write the RISC-V assembly code for the following C statement. Assume that elements of arrays A and B are 32 bits. Also, what is the hexadecimal machine code for those instructions?

$$B[2*k] = A[i - j]$$

Hint: Assume that variables i, j and k are assigned to registers X_{10} , X_{11} and X_{12} . Also, the base addresses of arrays A and B are registers X_5 and X_6 . Suppose that register X_{30} is used as the temporary register.

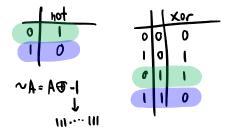
sub x30, x10, x11 slli x30, x5, x30 lw x30, 0(x30) slli x31, x12,3 add x31, x6, x31 sw x30, 0(x31)

2) Translate the following RISC-V code to C. Assume that the variable f is assigned to register X_5 . Also, the base address of array A is in register X_{10} . X_{30} and X_{31} are the temporary registers.

addi
$$x_{30}$$
, x_{10} , 8
 x_{30} = $A + 8 \cdot I = ACI$
addi x_{31} , x_{10} , 0
 x_{31} = $A + 8 \cdot 0 = ACI$
sd x_{31} , 0(x_{30})
 ACI] = ACO]
1d x_{30} , 0(x_{30})
 x_{30} = ACI]
add x_{5} , x_{30} , x_{31}
 x_{5} , x_{30} , x_{31}

3) Implement the following RISC-V pseudo instructions using the recommended instructions.

Pse	eudo Instruction	Meaning	Recommended Instruction
not	rd, rs	rd = ~rs	xori rd, rs, -1
nop		Neither the registers nor the memory contents change	addi x0, x0, G
mov	rd, rs	rd = rs	addi rd, rs, O
j	label1	Jump to a label (not linking)	jal x0, labell



4) Translate the following C code to RISC-V assembly code. Assume that the values of a, b, i and j are in registers X_5 , X_6 , X_7 and X_{29} , respectively. Also, assume that register X_{10} holds the base address of the array D.

```
//i = 0
                li \times 7,0
                                                   11 i ? a?, if so, then end outer loop
              bge x7, x5, endloopl
                                                   // j=0
              li x29,0
                                                   1/j 7 b ?, if so, then end inner loop
                bge x29, x6, endloop2
                                                  //temp, = i + i = 2 i
//temp, = temp, + i = 3 i
               add x30, x7, x7
               add x30, x30, x7
                                                  // temp2 = j << 5 = (4 \cdot j) \cdot 8 = (2^3 \cdot j) \cdot 2^3
               slli x31, x29,5
                                                  // tempz = basep+(4·j)·8 = basep(4·j)
// D[x3|+0] = x30 → D[4·j]=3·i
               add x31, x10, x31
                sd x30, 0(x31)
addi x29, x29, l
j loop2
endloop2: addi x7, x7, l
j loop1
                                                  11 then repeat loop
                                                  11 itt repeat loop
```

) Fill out the following table:

IEEE 754 single precision	Smallest absolute value (excluding 0) s=x E(8):-126 M(13)=0 ±1.0.2-126	Largest absolute value (excluding infinity) 5 × ((3) = 127 M(23) = 11 ··· 11 ± 0.0 · 7 27
IEEE 754 double precision	5=x E(11)= +022 M(52)=0 ±1.0.7-1022	s-x E(8)=1023 M(62)=1111··· 11

$$\pm 1.2 \cdot 10^{-38}$$
 $\pm 3.4 \cdot 10^{38}$ $\pm 1.2 \cdot 10^{-308}$ $\pm 1.8 \cdot 10^{308}$

6) Convert the following numbers to IEEE 754 single precision binary representation:

7) To add two floating point numbers a and b with equal exponents, we can simply add the mantissa together, and then adjust the exponent by 1 if there is a carry. When exponents are not equal, we should modify the mantissa of one of the two numbers a and b (by shifting it) such that the exponents become equal, and then we can add the mantissa together. In doing so, we should not add any significant zeros to either a or b (See 'significant figure' on Wikipedia if you don't know what it is. You may also want to consult Google on 'floating point addition').

Consider a variation of IEEE 754 with 4 bits for exponent, 8 bits for fraction, and a bias of 7. Add the following numbers, once assuming to have unlimited precisions, and once under this variation of IEEE 754. Calculate the numerical error between the two cases.

$$A = (1.01100000)_{2} \times 2^{-3}, \quad B = (1.01100001)_{2} \times 2^{-4}$$

$$A = 0.00[0][00000]$$

$$B = 0.000[0][0000]$$

$$0.0100,001 0,0001$$

$$B = 0.000[0][0000]$$

$$S = 0$$

$$E = 0.000$$

Good Luck