**Department of Electrical & Computer Engineering**

**University of California, Davis**

**EEC 170 – Computer Architecture**

**Homework 2**

Due Friday Oct 24, 2024, 6PM

**Question 1**

Assume all registers are 32 bits. Assume all numbers are expressed in hexadecimal notation.

Initially Memory [00000064] = 456789AB and register x6 contains 00000064.

What are the contents of registers x5, x6, x7, x8, x9 and memory location 00000064 after the execution of the following RISC-V code?

lw x5, 0(x6)

slli x9, x5, 4

lb x7, 1(x6)

lbu x8, 2(x6)

addi x10, x0, -5

sw x0, 0(x6)

|  |
| --- |
| **x5 = 456789AB**  **x6 =** 00000064  **x7 =** 67  x8 = 00000089  **x9 = 5F6E5D4B2**  x10 = -5  x0 = 0 |

**Question 2**

010 0000 0 1111 0111 0000 0110 1011 0011

* Assume it is a RISC-V instruction, what is it?
* Assume it is single precision FP number, what is it?

|  |
| --- |
| 0100000 01111 01110 000 01101 0110011  R format  funct 3 = 0x0  func 7 32  This is a Sub function  Instruction is:  Sub x13, x14, x15  Single precision FP:  7.719568 |

**Question 3**

Translate the following RISC-V code into its corresponding machine language representation. Show your answer in both binary and hexadecimal.

add x17, x18, x19

sw x17, 100 (x18)

addi x17, x18, 100

|  |
| --- |
| **R format:**  **0000000 10011 10010 000 10001 0110011**  **Hex: 0x013908B3**  **S format:**  **0000011 10010 10001 010 00100 0100011 Hex: 0x0728A223**  **I format:**  **000001100100 10010 000 10001 0010011**  **Hex: 0x06490893** |

**Question 4**

Represent 0.2 in IEEE Floating point standard representation (32 bit). Express your answer in hexadecimal.

|  |
| --- |
| **0.2 \* 2 = 0.4 0**  **0.4 \* 2 = 0.8 0**  **0.8\*2 = 1.6 1**  **0.6\*2 = 1.2 1**  **0.2\*2 = 0.4 0**  **0.4\*2 = 0.8 0**  **0.8\*2 = 1.6 1**  **0.6\*2 = 1.2 1**  **0.00110011**  **1.10011**  **0 01111100 1001100 1100 1100 1100 1101**  **HEX: 3e4ccccd** |

**Question 5**

Assume A = 12.5 and B = 15.25

* Represent A and B in custom 10-bit floating-point representation with

5 bits for exponent with a bias of 15, 4 bits for the fraction and 1 bit for the sign

* Compute A - B using the FP add algorithm.
* Convert your **result** back to decimal and verify that your answer is correct, which means it is equal to -2.75

|  |
| --- |
| **0**  **1100**  **0.1**  **1100.1**  **1.1001\*2^3**  **1.1001\*2^18**  **A**  **= 0 10010 1001**  **1111**  **0.01**  **0.25 \* 2 0.5 0**  **0.5 \*2 1.0 1**  **1111.01**  **1.11101\*2^3**  **1.11101\*2^18**  **B**  **= 0 10010 1110**  **Addition:**  **0 10010 1001**  **0 10010 1110**    **1.10010\*2^18**  **+ (-1.11101\*2^18)**  **negative since the greater value has a negative sign**  **A-B =**  **1 10000 0110**  **which is equal to -2.75** |

**Question 6**

Consider the following RISC-V code

add x6, x9, x0 #Memory Address of this instruction is decimal 500

add x7, x2, x0

jal x1, L

sub x9, x10, x0

…

L: lui x9, 0xAB #Memory address of this instruction is decimal 1000

lw x10, 0(x9)

addi x10, x14, -5

beq x10, x0, L

jalr x0, 0(x1)

* What is the value of PC and register x1 after the execution of **jal L**? Explain how your arrived at the answer?
* What is the value of PC and register x0 after the execution of **jalr x0, 0(x1)**? Explain how you arrived at the answer?
* What would you store in the 12-bit immediate field in the encoding of the **beq** instruction above?

|  |
| --- |
| **The value of PC would be 1000 and X1 would store the address of the next instruction to be executed which is 512 since each instruction is seperated by 4 and the sub instruction would be the 500 + 3\*4 = 512th instruction.**  **The value of PC would be 512 since that is the address stored in X1 and the value of x0 would be 0 since that is always set to 0 and cannot be rewritten.**  **it would be -12**  **111111110100 in 2's complement** |

**Question 7**

// Function to search for an element in the array

int searchElement(int arr[], int size, int x) {

for (int i = 0; i < size; i++) {

if (arr[i] == x) {

return i; // Return index if element is found

}

}

return -1; // Return -1 if element is not found

}

Part (1) Translate this to an efficient RISC-V assembly language program.

Part (2) What is the size of your instruction memory (in bytes)?

Part (3) Assume arr is [4,34,3,3,53,59,39,9,-1,12]. If x = 13, what is the instruction count of your program? What is the instruction count if x = 3?

Part (4) What is the execution time of your program in each case assuming each instruction takes 2 cycles and the clock frequency of the processor is 1GHz?

|  |
| --- |
| Part (1)  **x32 = array**  **x31 = size**  **x30 = x**  **li x29, x0**  **search:**  **bge x29, x31, exit**  **slli x28, x29, 2**  **add x27, x32, x28**  **lw x26, 0(x27)**  **beq x26, x30, findexit**  **addi x29, x29, 1**  **j search**  **exit:**  **li x25, -1**  **j print**  **findexit:**  **sw x25, x29**  **j print**  **print:**  **...**  Part (2)  **12 instructions excluding the print function and initializations**  **12\*4 = 48 bytes**  Part (3)  Case x =13  1 + 7(inside loop)\*10 + 1 + 2 = 74 instructions and 296 bytes  Case x =3  1 + 7(inside loop)\*2 + 5 + 2 = 22 instructions and 88 bytes  Part (4)  **EXE time = CPI \* I \* 1/Clockspeed**  **Case x =13 = 2 \* 74 \* 1/(1\*10^9) = 1.48 \* 10^-7 seconds**  **Case x =3 = 2 \* 22 \* 1/(1\*10^9) = 4.4 \* 10^-8 seconds** |
|  |

(This one could need some additional space, feel free to use additional scratch paper)