**Homework 4 Answer Sheet**

Please state the name, SID and email of each member of your group.

|  |  |  |  |
| --- | --- | --- | --- |
| member | name | SID | email |
| #1 (contact person) | Ip wing kai | 58120400 | kaiwip3-c@my.cityu.edu.hk |
| #2 | LIN Weiyuan | 58537684 | weiyualin3-c@my.cityu.edu.hk |
| #3 | LEE Kin Wai | 57925345 | kwlee232-c@my.cityu.edu.hk |

Logisim Version: logisim-win-2.7.1.exe

OS (Window/MAC/Linux):\_Windows 11\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Do all members make significant contributions to this homework? If not, please specify the details.

Yes, all members make significant contributions to this project homework.

1. Please explain how many types of instructions are supported in your processor, and explain the format of each type of instructions (e.g., which bits are used as the operation or function code, which bits are used to index the 1st, 2nd or 3rd operand, and which bits are used to store the immediate number). You can draw figures to better explain your answer.

X = 58120400 + 58537684 + 57925345 = 174,583,429

Y = X mod 7 = 6

There are 3 types of instructions are supported: R-type, I-type, and J-type.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R-type | opcode | rt | | | rs | | | rd | | | opcode | | | | | | |
| I-type | immediate | rt | | | rd | | | immediate | | | | | opcode | | | | |
| J-type | Immediate | | | | | | | | | | | | opcode | | | | |

Note: Opcode = operation code, rs = 1st operand, rt = 2nd operand, rd = 3rd operand, immediate = immediate number.

Specific instructions:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| I-type | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| move | unused | rt | | | rd | | | unused | | | | | opcode | | | |
| load | unused | rt | | | rs | | | unused | | | | | opcode | | | |
| store | unused | rt | | | rs | | | unused | | | | | opcode | | | |

1. Please explain the format of each instruction (including the format of this instruction and its operation codes, and other information if needed).

x = unused, s = operand 1, t = operand 2, d = destination, i = immediate

|  |  |  |
| --- | --- | --- |
|  | format | Opcode |
| li | i xxx ddd iiiii 1000 | 1000 |
| add | 0 ttt sss ddd 00 0000 | 0 00 0000 |
| and | 0 ttt sss ddd 01 0010 | 0 10 0010 |
| or | 0 ttt sss ddd 10 0100 | 0 01 0100 |
| load | x ttt sss xxxxx 1100 | 1100 |
| store | x ttt sss xxxxx 1010 | 1010 |
| move | x ttt ddd xxxxx 1001 | 1001 |
| addi | i ttt ddd iiiii 0001 | 0001 |
| andi | i ttt ddd iiiii 0011 | 0011 |
| ori | i ttt ddd iiiii 0101 | 0101 |
| ble | i ttt sss iiiii 0110 | 0110 |
| bne | i ttt sss iiiii 1110 | 1110 |
| jump | iiiiiiiiiiii 1101 | 1101 |
| call | iiiiiiiiiiii 1011 | 1011 |
| rtn | xxxxxxxxxxxx 0111 | 0111 |
| halt | xxxxxxxxxxxx 1111 | 1111 |

1. Fill the following tables with the machine codes of each instruction of the testing programs:

**Test program 1:**

|  |  |  |
| --- | --- | --- |
| instruction | machine code (binary) | machine code (hex) |
| li $r1, 1 | 0000 0000 0001 1000 | 0018 |
| li $r2, 2 | 0000 0010 0010 1000 | 0228 |
| li $r3, 10 | 0000 0100 1010 1000 | 04a8 |
| add $r2, $r1, $r2 | 0001 0000 0100 0000 | 1040 |
| ble $r2, $r3, -1 | 1 010 001 11111 0110 | a3f6 |
| halt | 0000 0000 0000 1111 | 000f |

**Test program 2:**

|  |  |  |
| --- | --- | --- |
| Instruction | machine code (binary) | machine code (hex) |
| li $r1, 6 | 0000 0000 0110 1000 | 0068 |
| li $r2, 5 | 0000 0010 0101 1000 | 0258 |
| andi $r3, $r1, 3 | 0 000 010 00011 0011 | 0433 |
| ori $r4, $r3, 8 | 0 010 011 01000 0101 | 2685 |
| halt | 0000 0000 0000 1111 | 000f |

**Test program 3:**

|  |  |  |
| --- | --- | --- |
| instruction | machine code (binary) | machine code (hex) |
| li $r1, 6 | 0 000 000 00110 1000 | 0068 |
| li $r2, 5 | 0 000 001 00101 1000 | 0258 |
| and $r3, $r1, $r2 | 0 001 000 010 010010 | 1092 |
| li $r8, 0 | 0 000 111 00000 1000 | 0e08 |
| store $r3, $r8 | 0 111 010 00000 1010 | 740a |
| or $r4, $r1, $r2 | 0 001 000 011 000100 | 10c4 |
| li $r8, 1 | 0 000 111 00001 1000 | 0e18 |
| store $r4, $r8 | 0 111 011 00000 1010 | 760a |
| li $r8, 1 | 0 000 111 00001 1000 | 0e18 |
| load $r7, $r8 | 0 111 110 00000 1100 | 7c0c |
| halt | 0000 0000 0000 1111 | 000f |

**Test program 4:**

|  |  |  |
| --- | --- | --- |
| instruction | machine code (binary) | machine code (hex) |
| li $r1, 6 | 0 000 000 00110 1000 | 0068 |
| li $r2, 4 | 0 000 001 00100 1000 | 0248 |
| call 7 | 0000 0000 0111 1011 | 007b |
| move $r4, $r3 | 0 010 011 00000 1001 | 2609 |
| li $r1, 7 | 0 000 000 00111 1000 | 0078 |
| li $r2, 8 | 0 000 001 01000 1000 | 0288 |
| call 3 | 0000 0000 0011 1011 | 003b |
| move $r5, $r3 | 0 010 100 00000 1001 | 2809 |
| jump 3 | 0000 0000 0011 1101 | 003d |
| add $r3, $r1, $r2 | 0 001 000 010 000000 | 1080 |
| rtn | 0000 0000 0000 0111 | 0007 |
| halt | 0000 0000 0000 1111 | 000f |