**Homework 4 Answer Sheet**

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

|  |  |  |  |
| --- | --- | --- | --- |
| member | name | SID | email |
| #1 (contact person) | LIU Hengche | 57854329 | hengchliu2-c@my.cityu.edu.hk |
| #2 | XIA Shujun | 57854422 | shujunxia2-c@my.cityu.edu.hk |
| #3 | CHEN Yihuan | 57853935 | yihuachen6-c@my.cityu.edu.hk |

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

Yes.

1. Which version of Logisim was used for your design of the circuits?

Logisim Version and OS: logisim-generic-2.7.1 on Windows

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.

R-type:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| opcode (15:11) | rs (10:8) 2nd | rt (7:5) 3rd | rt (4:2) 1st | Useless funct(0:1) |

1. type:

|  |  |  |  |
| --- | --- | --- | --- |
| opcode (15:11) | rs (10:8) 2nd | rt (7:5) 1st | immediate (5:0) 3rd |

1. type:

|  |  |
| --- | --- |
| opcode (15:11) | target (10:0) |

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

|  |  |
| --- | --- |
| li | Format: 00000 xxx aaa bbbbb  Operation code: 00000 (15:11)  Useless: xxx (10:8)  Write Register: aaa (7:5)  Immediate: bbbbb (4:0)  Control Unit output: 0 1 1 0 0 0 0 0  ALUop: 1001 |
| add | Format: 00001 aaa bbb ccc xx  Operation code: 00001 (15:11)  Rs: aaa (10:8)  Rt: bbb (7:5)  Rd: ccc (4:2)  Funct: xx(1:0)  Control Unit output: 1 1 0 0 0 0 0 0  ALUop: 0001 |
| and | Format: 00010 aaa bbb ccc xx  Operation code: 00001 (15:11)  Rs: aaa (10:8)  Rt: bbb (7:5)  Rd: ccc (4:2)  Funct: xx(1:0)  Control Unit output: 1 1 0 0 0 0 0 0  ALUop: 0010 |
| or | Format: 00011 aaa bbb ccc xx  Operation code: 00011 (15:11)  Rs: aaa (10:8)  Rt: bbb (7:5)  Rd: ccc (4:2)  Funct: xx(1:0)  Control Unit output: 1 1 0 0 0 0 0 0  ALUop: 0011 |
| neg | Format: 00100 aaa bbb ccccc  Operation code: 00100 (15:11)  Rs: aaa (10:8)  Rt: bbb (7:5)  Immediate: ccccc (4:0)  Control Unit output: 0 1 1 0 0 0 0 0  ALUop: 0100 |
| load | Format: 00101 aaa bbb ccccc  Operation code: 00101 (15:11)  Rs: aaa (10:8)  Rt: bbb (7:5)  Immediate: ccccc (4:0)  Control Unit output: 0 1 1 0 1 0 1 0  ALUop: 0000 |
| store | Format: 00110 aaa bbb ccccc  Operation code: 00110 (15:11)  Rs: aaa (10:8)  Rt: bbb (7:5)  Immediate: ccccc (4:0)  Control Unit output: 0 0 1 0 0 1 X 0  ALUop: 0000 |
| move | Format: 00111 aaa bbb ccccc  Operation code: 00111 (15:11)  Rs: aaa (10:8)  Rt: bbb (7:5)  Immediate: ccccc (4:0)  Control Unit output: 0 1 1 0 0 0 0 0  ALUop: 0000 |
| addi | Format: 01000 aaa bbb ccccc  Operation code: 01000 (15:11)  Rs: aaa (10:8)  Rt: bbb (7:5)  Immediate: ccccc (4:0)  Control Unit output: 0 1 1 0 0 0 0 0  ALUop: 0001 |
| andi | Format: 01001 aaa bbb ccccc  Operation code: 01001 (15:11)  Rs: aaa (10:8)  Rt: bbb (7:5)  Immediate: ccccc (4:0)  Control Unit output: 0 1 1 0 0 0 0 0  ALUop: 0010 |
| ori | Format: 01010 aaa bbb ccccc  Operation code: 01010 (15:11)  Rs: aaa (10:8)  Rt: bbb (7:5)  Immediate: ccccc (4:0)  Control Unit output: 0 1 1 0 0 0 0 0  ALUop: 0011 |
| ble | Format: 01011 aaa bbb ccccc  Operation code: 01010 (15:11)  Rs: aaa (10:8)  Rt: bbb (7:5)  Immediate: ccccc (4:0)  Control Unit output: 0 0 0 1 0 0 X 0  BLE: 1 |
| slt | Format: 01100 aaa bbb ccc xx  Operation code: 01100 (15:11)  Rs: aaa (10:8)  Rt: bbb (7:5)  Rd: ccc (4:2)  Funct: xx(1:0)  Control Unit output: 1 1 0 0 0 0 0 0  ALUop: 0110 |
| lsl | Format: 01101 aaa bbb ccc xx  Operation code: 01100 (15:11)  Rs: aaa (10:8)  Rt: bbb (7:5)  Rd: ccc (4:2)  Funct: xx(1:0)  Control Unit output: 1 1 0 0 0 0 0 0  ALUop: 0110 |
| lsr | Format: 01110 aaa bbb ccc xx  Operation code: 01100 (15:11)  Rs: aaa (10:8)  Rt: bbb (7:5)  Rd: ccc (4:2)  Funct: xx(1:0)  Control Unit output: 1 1 0 0 0 0 0 0  ALUop: 0111 |
| jump | Format: 01111 ttttttttttt  Operation code: 01111 (15:11)  Target: ttttttttttt (10:0)  Control Unit output: 0 0 0 0 0 0 0 0  ALUop: 1010 |
| call | Format: 10000 ttttttttttt  Operation code: 10000 (15:11)  Target: ttttttttttt (10:0)  Control Unit output: 0 0 0 0 0 0 0 0  ALUop: 1011 |
| rtn | Format: 10001 ttttttttttt  Operation code: 10001 (15:11)  Target: ttttttttttt (10:0)  Control Unit output: 0 0 0 0 0 0 0 0  ALUop: 1100 |
| reboot | Format: 10010 ttttttttttt  Operation code: 10010 (15:11)  Target: ttttttttttt (10:0)  Control Unit output: 0 0 0 0 0 0 0 0  ALUop: 1101 |
| halt | Format: 10011 ttttttttttt  Operation code: 10011 (15:11)  Target: ttttttttttt (10:0)  Control Unit output: 0 0 0 0 0 0 0 1 |

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 0000 0001 | 0001 |
| li $r2, 2 | 0000 0000 0010 0010 | 0022 |
| li $r3, 10 | 0000 0000 0100 1010 | 004A |
| add $r2, $r1, $r2 | 0000 1000 0010 0100 | 0824 |
| ble $r2, $r3, -1 | 0101 1010 0011 1111 | 5A3F |
| slt $r4, $r3, $r2 | 0110 0010 0010 1100 | 622C |
| halt | 1001 1000 0000 0000 | 9800 |

**Test program 2:**

|  |  |  |
| --- | --- | --- |
| instruction | machine code (binary) | machine code (hex) |
| li $r1, 3 | 0000 0000 0000 0011 | 0003 |
| li $r2, 5 | 0000 0000 0010 0101 | 0025 |
| andi $r3, $r1, 3 | 0100 1000 0100 0011 | 4843 |
| ori $r4, $r3, 8 | 0101 0010 0110 1000 | 5268 |
| neg $r5, $r4 | 0010 0011 1000 0000 | 2380 |
| lsl $r6, $r5, $r1 | 0110 1100 0001 0100 | 6C14 |
| lsr $r7, $r5, $r2 | 0111 0100 0011 1000 | 7438 |
| halt | 1001 1000 0000 0000 | 9800 |

**Test program 3:**

|  |  |  |
| --- | --- | --- |
| instruction | machine code (binary) | machine code (hex) |
| li $r1, 6 | 0000 0000 0000 0110 | 0006 |
| li $r2, 5 | 0000 0000 0010 0101 | 0025 |
| and $r3, $r1, $r2 | 0001 0000 0010 1000 | 1028 |
| li $r8, 0 | 0000 0000 1110 0000 | 00e0 |
| store $r3, $r8 | 0011 0111 0100 0000 | 3740 |
| or $r4, $r1, $r2 | 0001 1000 0010 1100 | 182c |
| li $r8, 1 | 0000 0000 1110 0001 | 00e1 |
| store $r4, $r8 | 0011 0111 0110 0000 | 3760 |
| li $r8, 1 | 0000 0000 1110 0001 | 00e1 |
| load $r7, $r8 | 0010 1111 1100 0000 | 2fc0 |
| reboot | 1001 0000 0000 0000 | 9000 |
| halt | 1001 1000 0000 0000 | 9800 |

**Test program 4:**

|  |  |  |
| --- | --- | --- |
| instruction | machine code (binary) | machine code (hex) |
| li $r1, 6 | 0000 0000 0000 0110 | 0006 |
| li $r2, 4 | 0000 0000 0010 0100 | 0024 |
| call 7 | 1000 0000 1000 0111 | 8007 |
| move $r4, $r3 | 0011 1010 0110 0000 | 3A60 |
| li $r1, 7 | 0000 0000 0000 0111 | 0007 |
| li $r2, 8 | 0000 0000 0010 1000 | 0028 |
| call 3 | 1000 0000 1000 0011 | 8003 |
| move $r5, $r3 | 0011 1010 1000 0000 | 3A80 |
| jump 3 | 0111 1000 0000 0011 | 7803 |
| add $r3, $r1, $r2 | 0000 1000 0010 1000 | 0828 |
| rtn | 1000 1000 0000 0000 | 8800 |
| halt | 1001 1000 0000 0000 | 9800 |