**Homework 4 Answer Sheet**

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

|  |  |  |  |
| --- | --- | --- | --- |
| member | name | SID | email |
| #1 (contact person) |  |  |  |
| #2 |  |  |  |
| #3 |  |  |  |

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

1. Which version of Logisim was used for your design of the circuits?
2. 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 5bits(15:11) | rs 3bits(10:8) | rt 3bits(7:5) | rd 5bits(4:0) |

I-type

|  |  |  |  |
| --- | --- | --- | --- |
| Opcode 5bits(15:11) | rs 3bits(10:8) | rt 3bits(7:5) | Immediate num 5bits(4:0) |

J-type

|  |  |  |
| --- | --- | --- |
| Opcode 5bits(15:11) | Unuse 6bits(10:5) | Immediate num 5bits(4:0) |

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

|  |  |
| --- | --- |
| li r1 n | 00000 xxx rrr iiiii  ALUControl:0000  RegDst:0  RegWrite:1  ALUSrc:1  Branch:0  MemRead:0  MenWrite:0  MemtoReg:0  Li:1  Write$ra:0  Read$ra:0  EnReboot:0  Enjump:0 |
| Add r1 r2 r3 | 00001 r2r2r2 r3r3r3 00r1r1r1  ALUControl:0001  RegDst:1  RegWrite:1  ALUSrc:0  Branch:0  MemRead:0  MenWrite:0  MemtoReg:0  Li:1  Write$ra:0  Read$ra:0  EnReboot:0  Enjump:0 |
| and r1 r2 r3 | 00010 r2r2r2 r3r3r3 00r1r1r1  ALUControl:0010  RegDst:1  RegWrite:1  ALUSrc:1  Branch:0  MemRead:0  MenWrite:0  MemtoReg:0  Li:1  Write$ra:0  Read$ra:0  EnReboot:0  Enjump:0 |
| or r1 r2 r3 | 00011 r2r2r2 r3r3r3 00r1r1r1  ALUControl:0011  RegDst:1  RegWrite:1  ALUSrc:1  Branch:0  MemRead:0  MenWrite:0  MemtoReg:0  Li:1  Write$ra:0  Read$ra:0  EnReboot:0  Enjump:0 |
| Neg r1 r2 | 00100 r2r2r2 r1r1r1 xxxxx  ALUcontrol:0100  RegDst:0  RegWrite:1  ALUSrc:0  Branch:0  MemRead:0  MenWrite:0  MemtoReg:0  Li:1  Write$ra:0  Read$ra:0  EnReboot:0  Enjump:0 |
| Load r1 r2 | 00101 r2r2r2 r1r1r1 xxxxx  ALUcontrol:1001  RegDst:0  RegWrite:1  ALUSrc:0  Branch:0  MemRead:1  MenWrite:0  MemtoReg:1  Li:1  Write$ra:0  Read$ra:0  EnReboot:0  Enjump:0 |
| store r1 r2 | 00110 r2r2r2 r1r1r1 xxxxx  ALUcontrol:1001  RegDst:0  RegWrite:1  ALUSrc:0  Branch:0  MemRead:0  MenWrite:1  MemtoReg:0  Li:1  Write$ra:0  Read$ra:0  EnReboot:0  Enjump:0 |
| move r1 r2 | 00111 r2r2r2 r1r1r1 xxxxx  ALUcontrol:1001  RegDst:0  RegWrite:1  ALUSrc:0  Branch:0  MemRead:0  MenWrite:0  MemtoReg:0  Li:1  Write$ra:0  Read$ra:0  EnReboot:0  Enjump:0 |
| addi r1 r2 n | 01000 r2r2r2 r1r1r1 iiiii  ALUcontrol:0001  RegDst:0  RegWrite:1  ALUSrc:1  Branch:0  MemRead:0  MenWrite:0  MemtoReg:0  Li:1  Write$ra:0  Read$ra:0  EnReboot:0  Enjump:0 |
| andi r1 r2 n | 01001 r2r2r2 r1r1r1 iiiii  ALUcontrol:0010  RegDst:0  RegWrite:1  ALUSrc:1  Branch:0  MemRead:0  MenWrite:0  MemtoReg:0  Li:1  Write$ra:0  Read$ra:0  EnReboot:0  Enjump:0 |
| ori r1 r2 n | 01010 r2r2r2 r1r1r1 iiiii  ALUcontrol:0011  RegDst:0  RegWrite:1  ALUSrc:1  Branch:0  MemRead:0  MenWrite:0  MemtoReg:0  Li:1  Write$ra:0  Read$ra:0  EnReboot:0  Enjump:0 |
| ble r1 r2 n | 01011 r1r1r1 r2r2r2 iiiii  ALUcontrol:0101  RegDst:0  RegWrite:1  ALUSrc:1  Branch:0  MemRead:0  MenWrite:0  MemtoReg:0  Li:1  Write$ra:0  Read$ra:0  EnReboot:0  Enjump:0 |
| slt r1 r2 r3 | 01100 r2r2r2 r3r3r3 00r1r1r1  ALUcontrol:0110  RegDst:1  RegWrite:1  ALUSrc:0  Branch:0  MemRead:0  MenWrite:0  MemtoReg:0  Li:1  Write$ra:0  Read$ra:0  EnReboot:0  Enjump:0 |
| lsl r1 r2 r3 | 01101 r2r2r2 r3r3r3 00r1r1r1  ALUcontrol:0111  RegDst:1  RegWrite:1  ALUSrc:0  Branch:0  MemRead:0  MenWrite:0  MemtoReg:0  Li:1  Write$ra:0  Read$ra:0  EnReboot:0  Enjump:0 |
| lsr r1 r2 r3 | 01110 r2r2r2 r3r3r3 00r1r1r1  ALUcontrol:1000  RegDst:1  RegWrite:1  ALUSrc:0  Branch:0  MemRead:0  MenWrite:0  MemtoReg:0  Li:1  Write$ra:0  Read$ra:0  EnReboot:0  Enjump:0  Enjump:0 |
| jump n | 01111 xxxxxx iiiii  ALUcontrol:xxxx  RegWrite:0  ALUSrc:0  Branch:0  MemRead:0  MenWrite:0  MemtoReg:0  Li:1  Write$ra:0  Read$ra:0  EnReboot:0  Enjump:1 |
| call n | 10000 xxxxxx iiiii  ALUcontrol:xxxx  RegWrite:0  ALUSrc:0  Branch:0  MemRead:0  MenWrite:0  MemtoReg:0  Li:1  Write$ra:1  Read$ra:0  EnReboot:0  Enjump:1 |
| rtn | 10001 xxxxxxxxxx  ALUcontrol:xxxx  RegWrite:0  ALUSrc:0  Branch:0  MemRead:0  MenWrite:0  MemtoReg:0  Li:1  Write$ra:0  Read$ra:1  EnReboot:0  Enjump:0 |
| reboot | 10010 xxxxxxxxxx  RegWrite:0  ALUSrc:0  Branch:0  MemRead:0  MenWrite:0  MemtoReg:0  Li:1  Write$ra:0  Read$ra:0  EnReboot:1  Enjump:0 |
| halt | 10011 xxxxxxxxxx  RegWrite:0  ALUSrc:0  Branch:0  MemRead:0  MenWrite:0  MemtoReg:0  Li:0  Write$ra:0  Read$ra:0  EnReboot:0  Enjump:0 |

1. Fill the following tables with the machine codes of each instruction of the testing programs:
2. **Test program 1:**

|  |  |  |
| --- | --- | --- |
| instruction | machine code (binary) | machine code (hex) |
| li $r1, 1 | 00000 000 000 00001 | 0001 |
| li $r2, 2 | 00000 000 001 00010 | 0022 |
| li $r3, 10 | 00000 000 010 00110 | 0046 |
| add $r2, $r1, $r2 | 00001 000 001 00001 | 0821 |
| ble $r2, $r3, -1 | 01011 001 010 10001 | 5951 |
| slt $r4, $r3, $r2 | 01100 010 001 00011 | 6223 |
| halt | 10011 000 000 00000 | 9800 |

1. **Test program 2:**

|  |  |  |
| --- | --- | --- |
| instruction | machine code (binary) | machine code (hex) |
| li $r1, 3 | 00000 000 000 00011 | 0007 |
| li $r2, 5 | 00000 000 001 00101 | 0025 |
| andi $r3, $r1, 3 | 01001 000 010 00011 | 4843 |
| ori $r4, $r3, 8 | 01010 010 011 00100 | 5264 |
| neg $r5, $r4 | 00100 011 100 00000 | 2380 |
| lsl $r6, $r5, $r1 | 01101 010 000 00101 | 6a05 |
| lsr $r7, $r5, $r2 | 01110 010 001 00110 | 7226 |
| halt | 10011 000 000 00000 | 9800 |

1. **Test program 3:**

|  |  |  |
| --- | --- | --- |
| instruction | machine code (binary) | machine code (hex) |
| li $r1, 6 | 00000 000 000 00110 | 0006 |
| li $r2, 5 | 00000 000 001 00101 | 0025 |
| and $r3, $r1, $r2 | 00010 000 001 00010 | 1022 |
| li $r8, 0 | 00000 000 111 00000 | 00e0 |
| store $r3, $r8 | 00110 010 111 00000 | 32e0 |
| or $r4, $r1, $r2 | 00011 000 001 00011 | 1823 |
| li $r8, 1 | 00000 000 111 00001 | 00e1 |
| store $r4, $r8 | 00110 011 111 00000 | 33e0 |
| li $r8, 1 | 00000 000 111 00001 | 00e1 |
| load $r7, $r8 | 00101 111 110 00000 | 2fc0 |
| reboot | 10010 000 000 00000 | 9000 |
| halt | 10011 000 000 00000 | 9800 |

1. **Test program 4:**

|  |  |  |
| --- | --- | --- |
| instruction | machine code (binary) | machine code (hex) |
| li $r1, 6 | 00000 000 000 00110 | 0006 |
| li $r2, 4 | 00000 000 001 00100 | 0024 |
| call 7 | 10000 000 000 00111 | 8007 |
| move $r4, $r3 | 00111 010 011 00000 | 3a60 |
| li $r1, 7 | 00000 000 000 00111 | 0007 |
| li $r2, 8 | 00000 000 001 01000 | 0024 |
| call 3 | 10000 000 000 00011 | 8003 |
| move $r5, $r3 | 00111 010 100 00000 | 3a80 |
| jump 3 | 01111 000 000 00011 | 7803 |
| add $r3, $r1, $r2 | 00001 000 001 00010 | 0822 |
| rtn | 10001 000 000 00000 | 8800 |
| halt | 10011 000 000 00000 | 9800 |