|  |  |  |  |
| --- | --- | --- | --- |
| **Instruction** | **Functionality** | **Opcode** | **Parity Bit** |
| load *imm* | $R4 = Mem[imm] | 011 iiii | 0 |
| store *Rx, Ry* | Mem[Ry] = Rx | 000 xx yy | 1 |
| srl Rx | Rx shift right one bit, 0 shifted into MSB | 001 00 xx | 0 |
| addi Rx, *imm* | Rx = Rx + imm | 100 xx ii | 0 |
| subi Rx | Rx = Rx + (-imm) | 101 10 xx | 0 |
| bne Rx | $R5 = Rx (!=) 0 | 110 11 xx | 1 |
| slt Rx | $R5 = Rx (<) 0 | 100 01 xx | 1 |
| BezDec *imm* | If $R5 == 0, then PC = PC + imm, else $R5 = $R5 – 1, PC = PC + 1 | 0100 iii | 0 |
| xori *Rx, imm* | $R5 = Rx (EXCL) with imm | 110 xx ii | 0 |
| andi *Rx, imm* | $R5 = Rx (AND) with imm | 111 xx ii | 0 |
| jump *‘branch’* | PC = PC - imm | 010 iiii | 1 |
| add Rx, Ry | Rx = Rx + Ry | 001 xx yy | 1 |
| sub Rx, Ry | Rx = Rx – Ry | 101 xx yy | 1 |
| subIn $R4 | $R4 = $R4 - 1 | 0110110 | 1 |
| bne $R4 | $R5 = $R4 (!=) 0 | 1110000 | 1 |
| jump ‘first branch’ | PC = 12 | 1010101 | 0 |
| halt | Stop | 000 00 00 | 0 |

**Machine Code for Program 1:**

#Assume everything is equal to zero at first

#$t0 = 00

#$t1 = 01

#$t2 = 10

#$t3 = 11

addi $t0, 1 100 00 01

addi $t1, 1 100 01 01

load 0(0x2000) 011 0000

addi $t2, 3 100 10 11

addi $t2, 2 100 10 10

addi $t3, 3 100 11 11

addi $t3, 3 100 11 11

addi $t3, 3 100 11 11

addi $t3, 3 100 11 11

addi $t3, 3 100 11 11

addi $t3, 2 100 11 10

loop:

bne $s0 1110000

BezDec 7 0100 111

next:

bne $t2 110 11 10

BezDec 4 0100 100

add $t1, $t0 001 01 00

subi $t2 101 10 10

jump ‘*next’* 010 1000

next2:

sub $t3, $t1 101 11 01

bne $s0 1110000

BezDec 7 0100 111

slt $t3 100 01 11

add $t3, $t1 001 11 01

BezDec 3 0100 011

sub $t1, $t3 101 01 11

jump *‘next2’* 010 0111

down:

addi $t2, 3 100 10 11

addi $t2, 2 100 10 10

bne $s0 1110000

BezDec 5 0100 101

subIn $s0, 1 0110110

sub $t0, $t0 101 00 00

add $t0, $t1 001 00 01

jump *‘loop’* 1010101 #this line may be wrong

exit:

store $t1, 0(0x2004) 000 0100

halt 000 0000