ISA Simulation Program

Mr. Tanapat Limtemap
6014216

ISA Simulation CPU

I’ve created ISA Simulation program by using Java Language. It is a 24-bit ISA.

Opcode 5 bits	Operand 1 3 bits	Binary number 16 bits

Registers

```
private String[] r = { "000", "001", "010", "011", "100", "101", "110", "111" };
```

The Binary of Opcode

```
private String mov = "00001";
private String add = "00010";
private String sub = "00011";
private String mul = "00100";
private String div = "00101";
```

The Clock Cycle of Opcode

```
private int c_mov = 1;
private int c_add = 2;
private int c_sub = 2;
private int c_mul = 4;
private int c_div = 4;
```

Mov

Operand 1 and Operand 2

mov r1 r2 -> to move the register 2 into register 1

Operand I and Value

mov r1 10 -> to move the value 10 into register 1

Add

Operand 1 and Operand 2

add r1 r2 -> to add the register 2 into register 1

Operand 1 and Value

add r1 10 -> to add the value 10 into register 1

Sub

Operand 1 and Operand 2

sub r1 r2 -> to sub the register 2 into register 1

Operand 1 and Value

sub r1 10 -> to sub the value 10 into register 1

Mul

Operand 1 and Operand 2

mul r1 r2 0> to multiply the register 2 into register 1

Operand 1 and Value

mul r1 10 -> to multiply the value 10 into register 1

Div

Operand 1 and Operand 2

div r1 r2 -> To divide the register 2 into register 1

Operand 1 and Value

div r1 10 -> to divide the value 10 into register 1

```
Input order
Opcode : mov , add , sub , mul , div
Operand 1 : R0-R7
Operand 2 : R0-R7 or decimal value
-> end 0 0 <- is the Opcode to stop input order
mov r0 6
mov r1 -5
mov r2 r1
add r3 7
add r4 r1
sub r1 2
sub r0 r1
mul r4 -6
mul r2 r1
div r3 3
div r4 r0
end 0 0
```

```
PC              Decoded      Encoded instructions(24 bit):   Clock cycles
PC[0]-->      mov r0 6     00001 00001 000000000000110            1
PC[1]-->      mov r1 -5     00001 00001 111111111111111            1
PC[2]-->      mov r2 -5     00001 00001 111111111111111            1
PC[3]-->      add r3 7     00010 00010 000000000000111            2
PC[4]-->      add r4 -5     00010 00010 111111111111111            2
PC[5]-->      sub r1 2     00011 00011 000000000000010            2
PC[6]-->      sub r0 -7     00011 00011 111111111111111            2
PC[7]-->      mul r4 -6     00100 00100 111111111111111            4
PC[8]-->      mul r2 -7     00100 00100 111111111111111            4
PC[9]-->      div r3 3     00101 00101 000000000000011            4
PC[10]-->      div r4 13     00101 00101 000000000001101             4

Step of Register
r0 = 6 [0000000000000110]
r1 = -5 [1111111111111111]
r2 = -5 [1111111111111111]
r3 = 7 [0000000000000111]
r4 = -5 [1111111111111111]
r1 = 2 [0000000000000010]
r0 = -7 [1111111111111111]
RM : r4 = -6 [00000000000000001111111111111111]
RM : r2 = -7 [00000000000000001111111111111111]
r3 = 3 [0000000000000011] RE : 1 [0000000000000001]
r4 = 13 [0000000000001101] RE : 4 [0000000000000100]
```

Final Register Result

```
r0 = 13 [0000000000001101]
r1 = -7 [1111111111111111]
r2 = 35 [0000000000100011]
r3 = 2 [0000000000000010]
r4 = 2 [0000000000000010]
r5 = 0 [0000000000000000]
r6 = 0 [0000000000000000]
r7 = 0 [0000000000000000]
```

CPI of the program

2.4545454545454546