Lab 2

Lab 2

Pseudocode

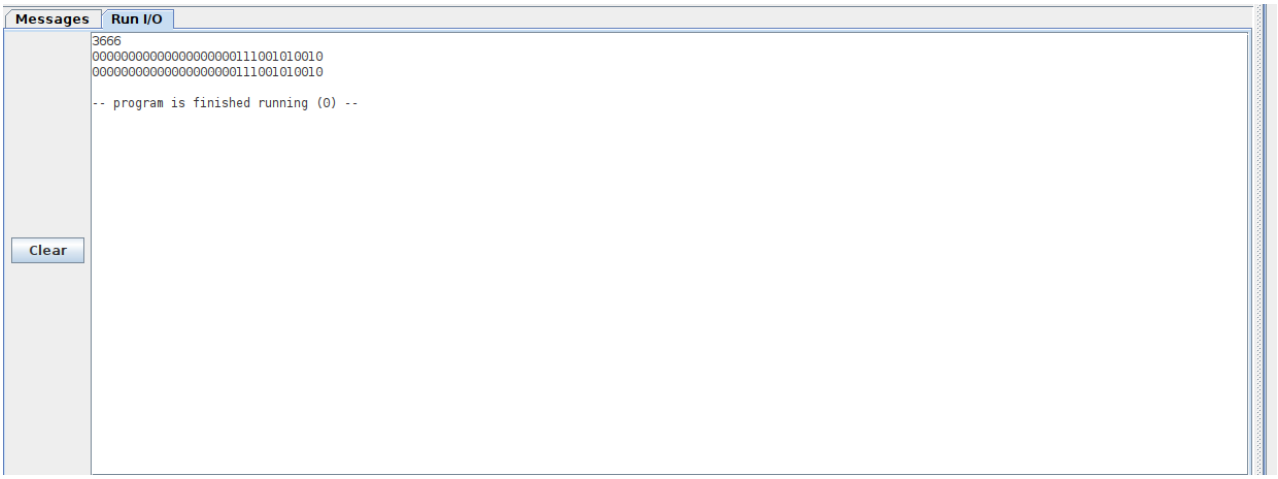
```
newline
Set mask
Loop:
    if nomask:
        End
    Else:
         if Mask & s1
                print 0
        else:
            print 1
    shift right
    return loop
End:
 newline
```

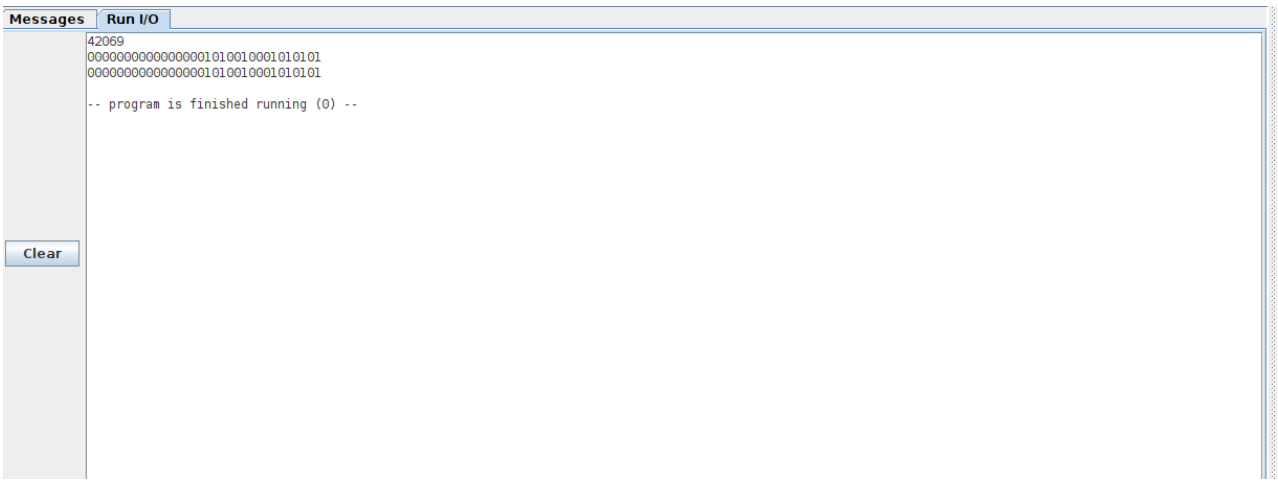
Code in RISC-V

```
 lui s1, 0x80000
Loop:   # if s0 == 0, End
    beq s0, x0, End # if s0 == 0, Ened

    and s2, s0, s1 # and s1 and s0 into s2

    bne s2, x0, Else # if s2 is 1 then go to the Else

    # if you made it here, you are the critical section
    li a7, 1    #load service to print integer
    addi a0, x0, 0      # load 0 (desired value) in register a0
    ecall # make the call

    beq x0, x0, Endif

Else:
    # we need to print 1
    li a7, 1 # load service to print integer
```

```
    addi a0, x0, 1 # load 1 into a0
    ecall # make the call

Endif:
    # shift s0 to the right
    srli s0, s0, 1

    beq x0, x0, Loop # back to the top of the loop agiain
End:
    # add ascii chars
    addi a7, x0, 11
    addi a0, x0, '\n'
    ecall
```

I/O Results

- Test input: 3666
  - 

- Test input: 42069
  - 

- Test input: -1, -3000000

```
Reset: reset completed.

-1
11111111111111111111111111111111
11111111111111111111111111111111

-- program is finished running (0) --


Reset: reset completed.

-300000
11111111111110110110110000100000
11111111111110110110110000100000

-- program is finished running (0) --
```

- Test input: 80084321

```
        lui s1, 0x80000
Loop:   # if s0 == 0, End
        beq s0, x0, End # if s0 == 0, Ened

        and s2, s0, s1 # and s1 and s0 into s2

        bne s2, x0, Else # if s2 is 1 then go to the Else

        # if you made it here, you are the critical section
        li a7, 1        #load service to print integer
        addi a0, x0, 0  # load 0 (desired value) in register a0
        ecall # make the call

        beq x0, x0, Endif

Else:
        # we need to print 1
        li a7, 1 # load service to print integer
        addi a0, x0, 1 # load 1 into a0
        ecall # make the call

Endif:
        # shift s0 to the right
        srli s0, s0, 1

        beq x0, x0, Loop # back to the top of the loop agiain
End:
        # add ascii chars
        addi a7, x0, 11
        addi a0, x0, '\n'
        ecall

        # exit
exit:   addi    a7, x0, 10
        ecall
```

```
ine: 27 Column: 24 Show Line Numbers
Messages | Run I/O
80084321
00000100110001011111110101100001
00000100110001011111110101100001

-- program is finished running (0) --

Clear
```

Analysis

This code first calls new line, runs the loop 31 times, moving the mask to the right and extracting the bit, resulting in either loading and printing a 1, or loading and printing a 0, and runs newline again. I tested it on varius outputs, including positive outputs, negative outputs and those shown in the lab2.md file. This code passes all of the inputs given in the lab2.md, working for both positive and negative cases. It cannot however take hex numbers, or else it raises an execution error. The code is also conscisely commented following the example code.