**COAL Lab # 03**

**Introduction to Assembly Language & Venus Simulator**

**Name:** Muhammad Usman

**Class:** BSCS 3C1

**Reg No:** cs211208

**Literature Review:**

Assembly Language:

Assembly language is the human-readable representation of the computer’s native language. Each assembly language instruction specifies both the operation to perform and the operands on which to operate. We introduce simple arithmetic instructions and show how these operations are written in assembly language. We then define the RISC-V instruction operands: registers, memory, and constants.

Venus simulator:

Venus is an emulator for RISC-V computers. It allows users to run programs written in RISC-V assembly language without having access to a computer running a RISC-V chip.

Registers:

Instructions need to access operands quickly so that they can run fast, but operands stored in memory take a long time to retrieve. Therefore, most architectures specify a small number of registers that hold commonly used operands. The RISC-V architecture has 32 registers, called the register set, stored in a small multi ported memory called a register file. The fewer the registers, the faster they can be accessed.

Machine Language:

Assembly language is convenient for humans to read. However, digital circuits understand only 1’s and 0’s. Therefore, a program written in assembly language is translated from mnemonics to a representation using only 1’s and 0’s, called machine language. RISC-V makes the compromise of defining four main instruction formats: R-type, I-type, S/B-type, and U/J-type.

Lab Exercise 01

**Task:**

Translate the Given High Level Code to Assembly Language

|  |  |
| --- | --- |
| **High - Level Code** | **RISC-V Assembly Code** |
| a = b − c; | sub x18,x19,x29 |
| f = (g + h) − (i + j); | add x5,x18,x19  add x6,x20,x21  sub x22,x5,x6 |

Lab Exercise 02

**Task:**

Convert the following RISC-V Assembly code into Machine Code

**Assembly:**

add x9, x5, x6

andi x10, x8, 0x6

or x5, x6, x7

slli x10, x6, 0x8

**Machine Code:**

**0000 0000 0110 0010 1000 0100 1011 0011**

**0000 0000 0110 0100 0111 0101 0001 0011**

**0000 0000 0111 0011 0110 0010 1011 0011**

**0000 0000 1000 0011 0001 0101 0001 0011**

**Hexa Code:**

0x006284B3

0x00647513

0x007362B3

0x00831513

In Lab Tasks

**Task:**

Convert the Given High Level Code on RISC-V Assembly and run it on Venus Simulator. Also write down its Machine Code

1. a = a + 4;

b = a − 12;

**Assembly:**

addi x18, x18, 0x4

addi x19, x18, -12

**Machine Code:**

**0000 0000 0100 1001 0000 1001 0001 0011**

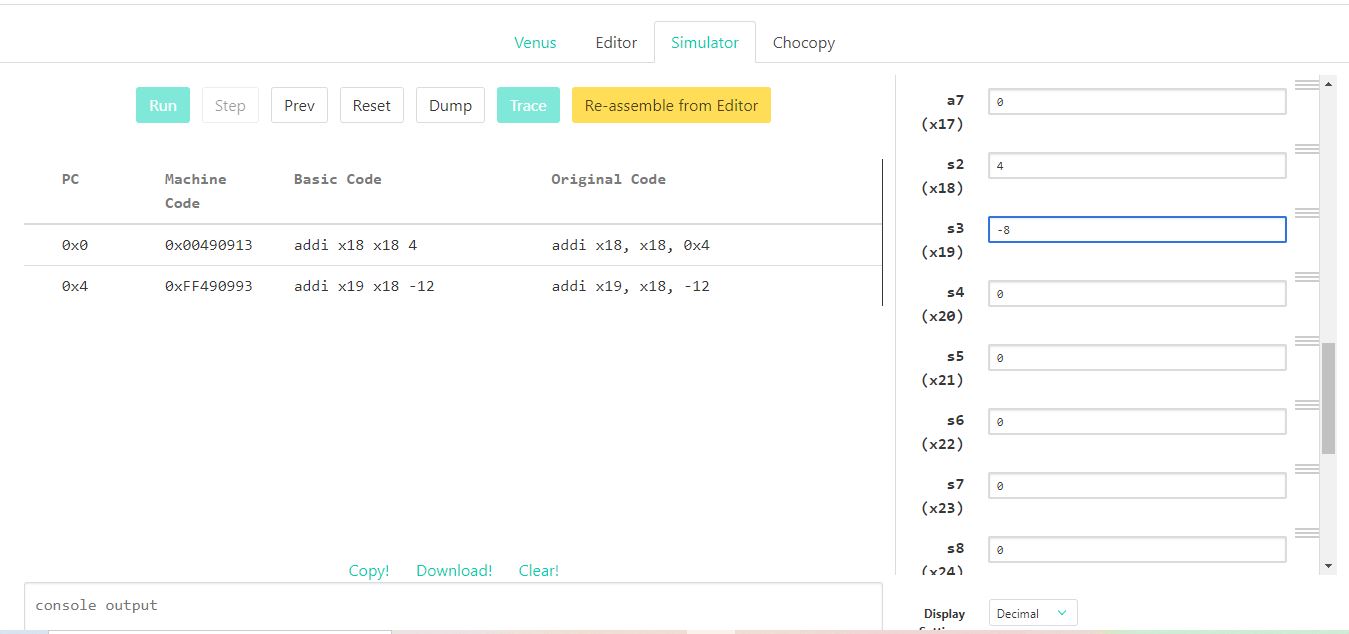
**1111 1111 0100 1001 0000 1001 1001 0011**

**Hexa Code:**

0x00490913

0xFF490993

**Venus Output:**



b) if (g < h)

g = g + 1;

else

h = h − 1;

**Assembly:**

blt x18,x19,Label

addi x19, x19, -1

Label:

addi x18,x18,0x1

**Machine Code:**

**0000 0001 0011 1001 0100 0100 0110 0011**

**1111 1111 1111 1001 1000 1001 1001 0011**

**0000 0000 0001 1001 0000 1001 0001 0011**

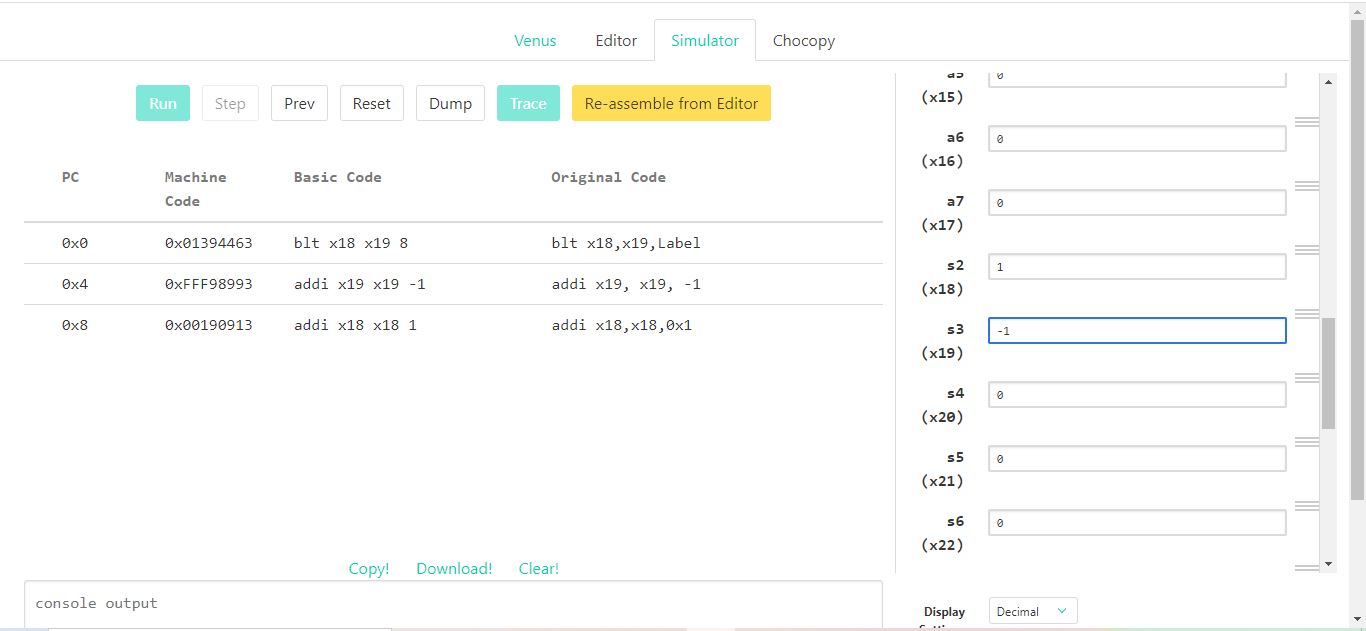
**Hexa Code:**

0x01394463

0xFFF98993

0x00190913

**Venus Output:**

****

Post Lab Tasks

**Task:**

Write down a simple C program to add, multiply and divide two integer numbers. Convert the C code into assembly and machine code and stimulate it on Venus.

**C program:**

int a = 5+2;

int m = 2\*2;

int d = 6/2;

**Assembly:**

addi x18,x18,0x5

addi x18,x18,0x2

addi x19,x19,0x2

mul x19,x19,x19

addi x25,x25,0x6

addi x26,x26,0x2

div x20,x25,x26

**Machine Code:**

**0000 0000 0101 1001 0000 1001 0001 0011**

**0000 0000 0010 1001 0000 1001 0001 0011**

**0000 0000 0010 1001 1000 1001 1001 0011**

**0000 0011 0011 1001 1000 1001 1011 0011**

**0000 0000 0110 1100 1000 1100 1001 0011**

**0000 0000 0010 1101 0000 1101 0001 0011**

**0000 0011 1010 1100 1100 1010 0011 0011**

**Hexa Code:**

0x00590913

0x00290913

0x00298993

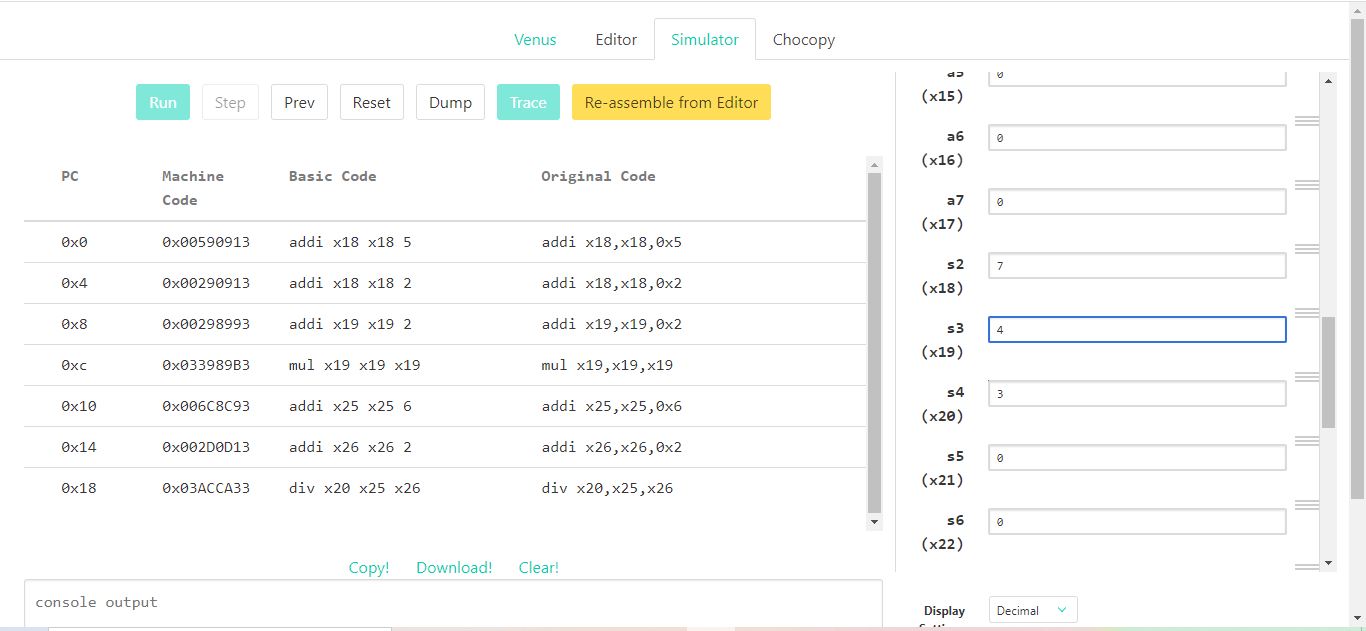
0x033989B3

0x006C8C93

0x002D0D13

0x03ACCA33

**Venus Output:**



****

**Task:**

Write down a simple C program to find out the prime numbers between 1-10 and give the final count of prime numbers. Convert the C code into assembly and machine code and stimulate it on Venus.

**C code:**

#include <stdio.h>

int main() {

    int c = 0;

    int flag;

    for(int i=2; i<11; i++){

        flag = 1;

        for(int j=2; j<(i/2); j++){

            if((i%j) == 0){

                flag = 0;

            }

        }

        if(flag){

            c++;

        }

    }

    printf("%d",c);

    return 0;

}

**Assembly:**

j main

main:

    addi x18,x18,0x2 # int i=2;

    addi x25,x0,0x1  # int 1 = 1;

    addi x22,x22,50 # int 11 = 11

    addi x21,x21,0x1 # int flag = 1

    Counter:

    beq x21,x25,CounterAdd # c++

    OuterLoop:

    addi x18,x18,0x1 # i++

    addi x21,x0,0x1 # flag = 1;

    addi x19,x0,0x2 # j = 2

    div x31,x18,x19

    blt x18,x22,Outer # for(int i=2; i<11);

    j End # if outer loop condition false the end

# if Outer loop condition true

InnerLoop:

    addi x19,x19,0x1

    Outer:

    blt x19,x18,Condition # for(int j=2; j<i);

    j Counter

Condition:

    rem x24,x18,x19

    beq x24,x0,PrimeFalse

    j InnerLoop

PrimeFalse:

    addi x21,x0,0x0

    j OuterLoop

CounterAdd:

    addi x20,x20,0x1

    j OuterLoop

End:

**Machine Code:**

**0000 0000 0010 1001 0000 1001 0001 0011**

**0000 0000 0001 0000 0000 1100 1001 0011**

**0000 0000 1011 1011 0000 1011 0001 0011**

**0000 0000 0001 1010 1000 1010 1001 0011**

**0000 0011 1001 1010 1000 1110 0110 0011**

**0000 0000 0001 1001 0000 1001 0001 0011**

**0000 0000 0001 0000 0000 1010 1001 0011**

**0000 0000 0010 0000 0000 1001 1001 0011**

**0000 0011 0011 1001 0100 1111 1011 0011**

**0000 0001 0110 1001 0100 0110 0110 0011**

**0000 0000 0001 1001 1000 1001 1001 0011**

**0000 0001 0010 1001 1100 0100 0110 0011**

**0000 0011 0011 1001 0110 1100 0011 0011**

**0000 0000 0000 1100 0000 0100 0110 0011**

**0000 0000 0000 0000 0000 1010 1001 0011**

**0000 0000 0001 1010 0000 1010 0001 0011**

**Hexa Code:**

0x00290913

0x00100C93

0x00BB0B13

0x001A8A93

0x039A8E63

0x00190913

0x00100A93

0x00200993

0x03394FB3

0x01694663

0x00198993

0x0129C463

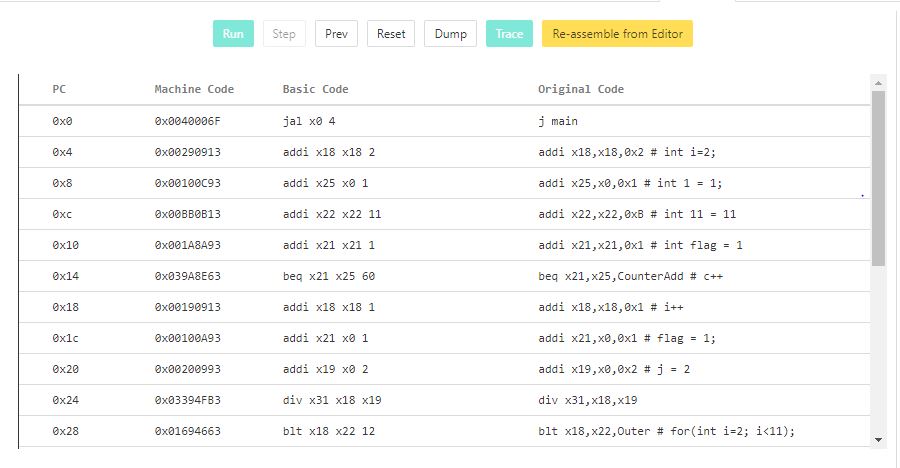
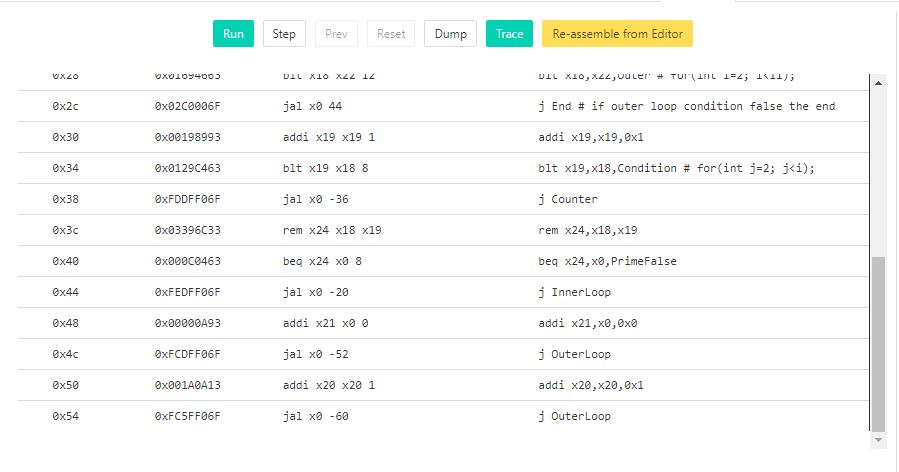
0x03396C33

0x000C0463

0x00000A93

0x001A0A13

**Venus Output:**

**** **** final output register

**C:\Users\DELL\Desktop\8.JPG**