COAL Lab-03

Introduction to Assembly Language & Venus Simulator

Name: Saad Nisar Butt

Reg. no: cs211246

Class: BSCS-3C-1

Literature Review:

Assembly Language:

Assembly Language is closer to low level language which is machine language or Machine. It is human readable representation of computer's native language. It is a command based language and 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 architecture specifies 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 Code:

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 1:

Task:

Translate the Given High Level Code to Assembly Language

High - Level Code	RISC-V Assembly Code
a = b − c;	sub s0, s1, s2
f = (g + h) − (i + j);	add t0, s3, s4 add t1, s5, s6 sub s7, t0, t1

Lab Exercise 2:

Task:

Convert the following RISC-V Assembly code into Machine Code

```
add x9, x5, x6

andi x10, x8, 0x6

or x5, x6, x7

slli x10, x6, 0x8
```

Solution:

add x9, x5, x6

0000000	00110	00101	000	01001	0110011

Machine Code: 0000 0000 0110 0010 1000 0100 1011 0011

Hexadecimal Code: 006284B3

andi x10, x8, 0x6

000000000110	01000	111	01010	0010011

Machine Code: 0000 0000 0110 0100 0111 0101 0001 0011

Hexadecimal Code: 00647513

or x5, x6, x7

0000000	00111	00110	110	00101	0110011

Machine Code: 0000 0000 0111 0011 0110 0010 1011 0011

Hexadecimal Code: 007362B3

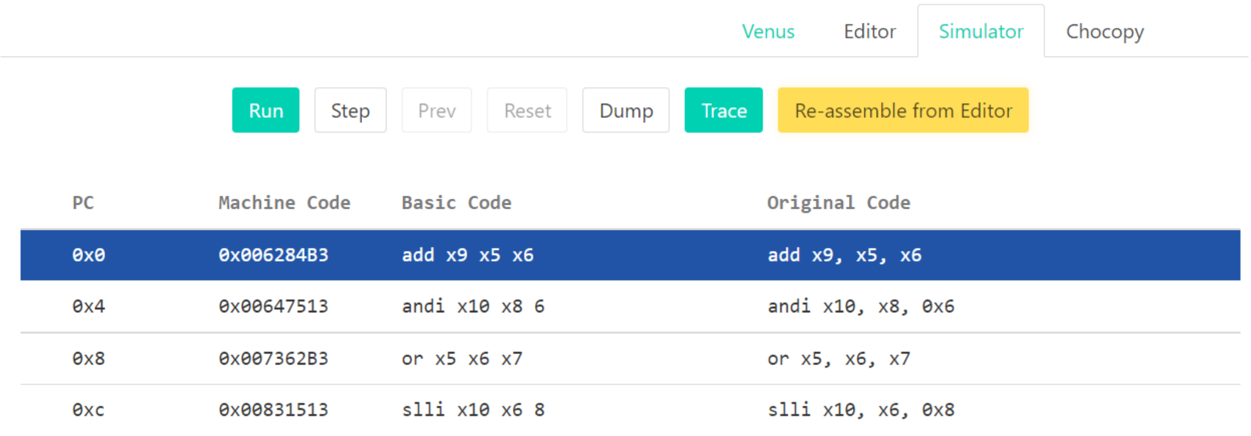
slli x10, x6, 0x8

000000001000	00110	001	01010	0010011

Machine Code: 0000 0000 1000 0011 0001 0101 0001 0011

Hexadecimal Code: 00831513

Venus Simulation:



PC	Machine Code	Basic Code	Original Code
0x0	0x006284B3	add x9 x5 x6	add x9, x5, x6
0x4	0x00647513	andi x10 x8 6	andi x10, x8, 0x6
0x8	0x007362B3	or x5 x6 x7	or x5, x6, x7
0xc	0x00831513	slli x10 x6 8	slli x10, x6, 0x8

In-Lab Task:

Task:

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

a)

High - Level Code	RISC-V Assembly Code
a = a + 4;	addi x5, x5, 0x4
b = a - 12;	addi x6, x5, -12

addi x5, x5, 0x4

000000000100	00101	000	00101	0010011

Machine Code: 0000 0000 0100 0010 1000 0010 1001 0011

Hexadecimal Code: 00428293

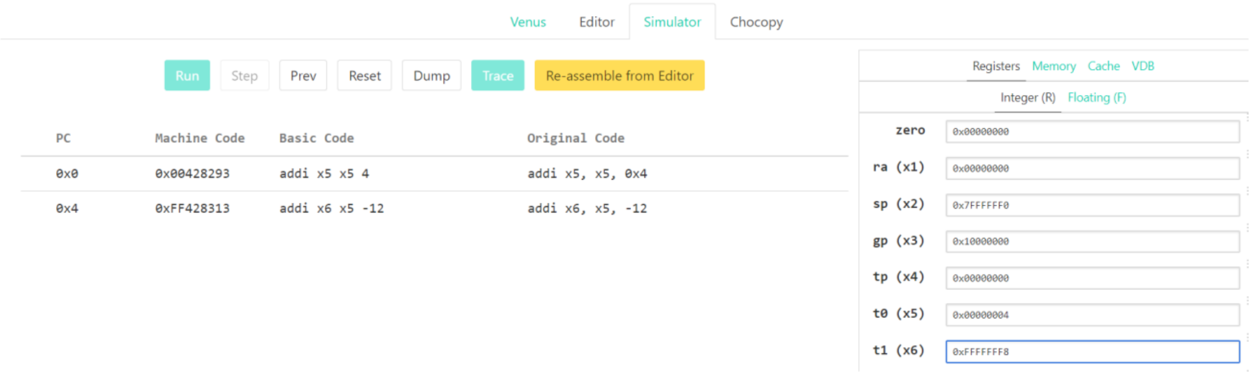
addi x6, x5, -12

111111110100	00101	000	00110	0010011

Machine Code: 1111 1111 0100 0010 1000 0011 0001 0011

Hexadecimal Code: FF428313

Venus Simulation:



PC	Machine Code	Basic Code	Original Code
0x0	0x00428293	addi x5 x5 4	addi x5, x5, 0x4
0x4	0xFF428313	addi x6 x5 -12	addi x6, x5, -12

zero	0x00000000
ra (x1)	0x00000000
sp (x2)	0x7FFFFFF0
gp (x3)	0x10000000
tp (x4)	0x00000000
t0 (x5)	0x00000004
t1 (x6)	0xFFFFFFF8

b)

High - Level Code	RISC-V Assembly Code

If (g < h)
g = g + 1
else
h = h - 1
	j main
	Label:
	addi x5, x5, 0x1
	main:
	blt x5, x6, Label
	addi x6, x6, -1

blt x5, x6, Label

1111111	00110	00101	100	11101	1100011

Machine Code: 1111 1110 0110 0010 1100 1110 1110 0011

Hexadecimal Code: FE62CEE3

addi x5, x5, 0x1

000000000001	00101	000	00101	0010011

Machine Code: 0000 0000 0001 0010 1000 0010 1001 0011

Hexadecimal Code: 00128293

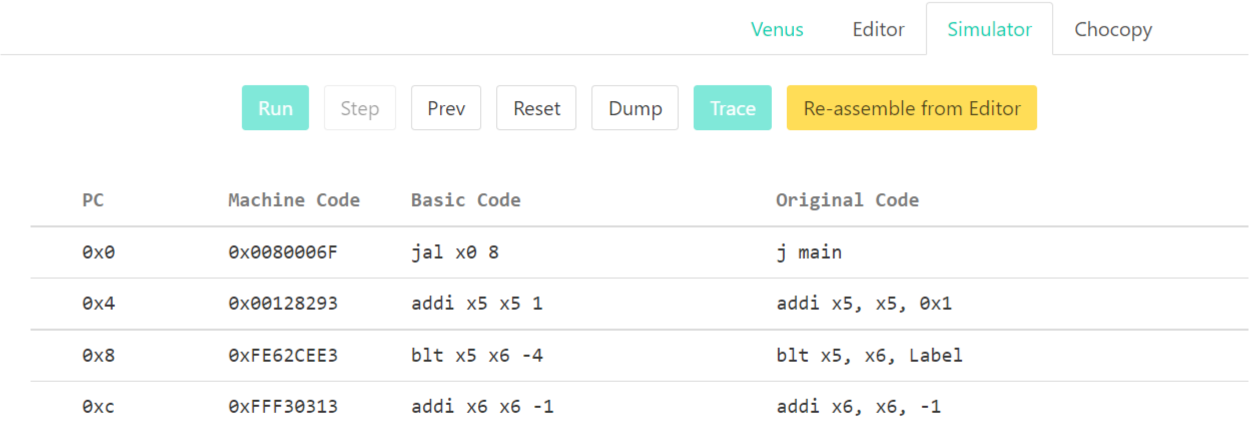
addi x6, x6, -1

111111111111	00110	000	00110	0010011

Machine Code: 1111 1111 1111 0011 0000 0011 0001 0011

Hexadecimal Code: FFF30313

Venus Simulation:



PC	Machine Code	Basic Code	Original Code
0x0	0x0080006F	jal x0 8	j main
0x4	0x00128293	addi x5 x5 1	addi x5, x5, 0x1
0x8	0xFE62CEE3	blt x5 x6 -4	blt x5, x6, Label
0xc	0xFFF30313	addi x6 x6 -1	addi x6, x6, -1

Post-Lab Tasks

Task # 1:

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 Code:

```
int main()
{
    int a = 4;
    int b = 2;

    int add = a + b;

    int mul = a * b;

    int div = a / b;

    printf("Add: %d\n", add);
    printf("Mul: %d\n", mul);
    printf("Div: %d\n", div);

    return 0;
}
```

RISC-V Assembley Code:

```
addi x5, x0, 0x4
addi x6, x0, 0x2

add x7, x5, x6

mul x28, x5, x6

div x29, x5, x6
```

addi x5, x0, 0x4

000000000100	00000	000	00101	0010011

Machine Code: 0000 0000 0100 0000 0000 0010 1001 0011

Hexadecimal Code: 00400293

addi x6, x0, 0x2

000000000010	00000	000	00110	0010011

Machine Code: 0000 0000 0010 0000 0000 0011 0001 0011

Hexadecimal Code: 00200313

add x7, x5, x6

0000000	00110	00101	000	00111	0110011

Machine Code: 0000 0000 0110 0010 1000 0011 1011 0011

Hexadecimal Code: 006283B3

mul x28, x5,x6

0000001	00110	00101	000	11100	0110011

Machine Code: 0000 0010 0110 0010 1000 1110 0011 0011

Hexadecimal Code: 02628E33

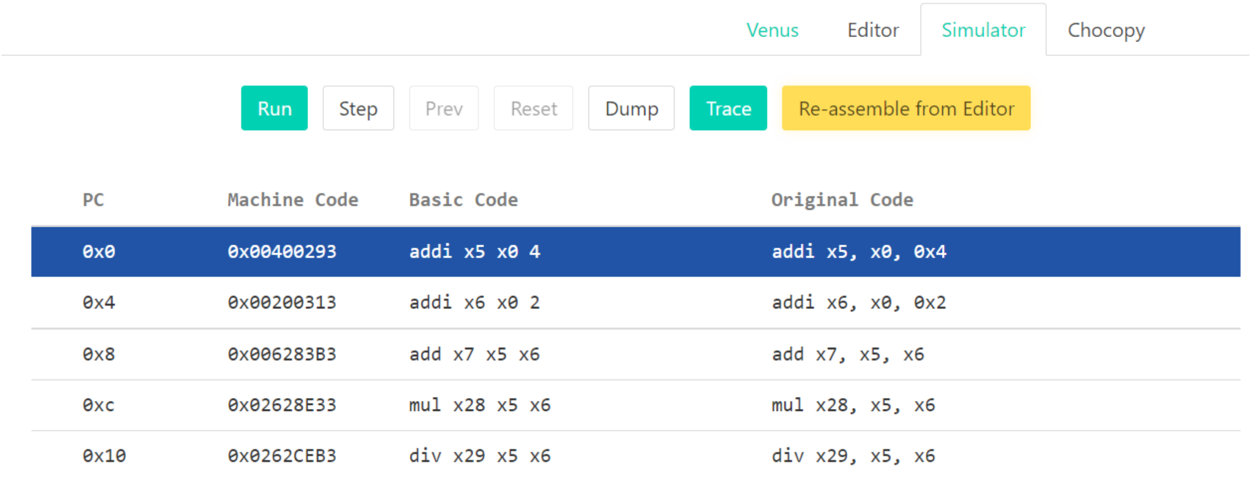
div x29, x5,x6

0000001	00110	00101	100	11101	0110011

Machine Code: 0000 0010 0110 0010 1100 1110 1011 0011

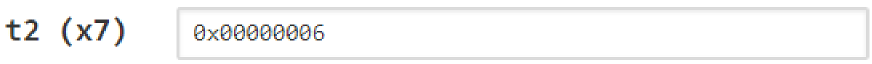
Hexadecimal Code: 0262CEB3

Venus Simulation:



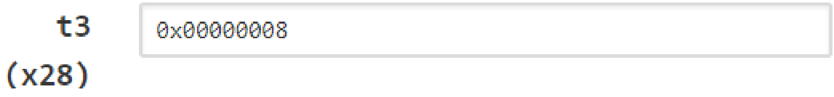
PC	Machine Code	Basic Code	Original Code
0x0	0x00400293	addi x5 x0 4	addi x5, x0, 0x4
0x4	0x00200313	addi x6 x0 2	addi x6, x0, 0x2
0x8	0x006283B3	add x7 x5 x6	add x7, x5, x6
0xc	0x02628E33	mul x28 x5 x6	mul x28, x5, x6
0x10	0x0262CEB3	div x29 x5 x6	div x29, x5, x6

Addition



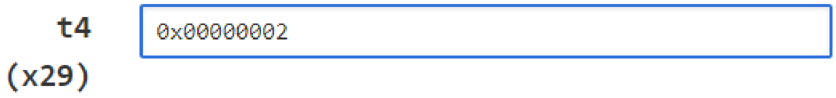
t2 (x7) 0x00000006

Multiplication



t3 (x28) 0x00000008

Division



t4 (x29) 0x00000002

Task # 2:

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:

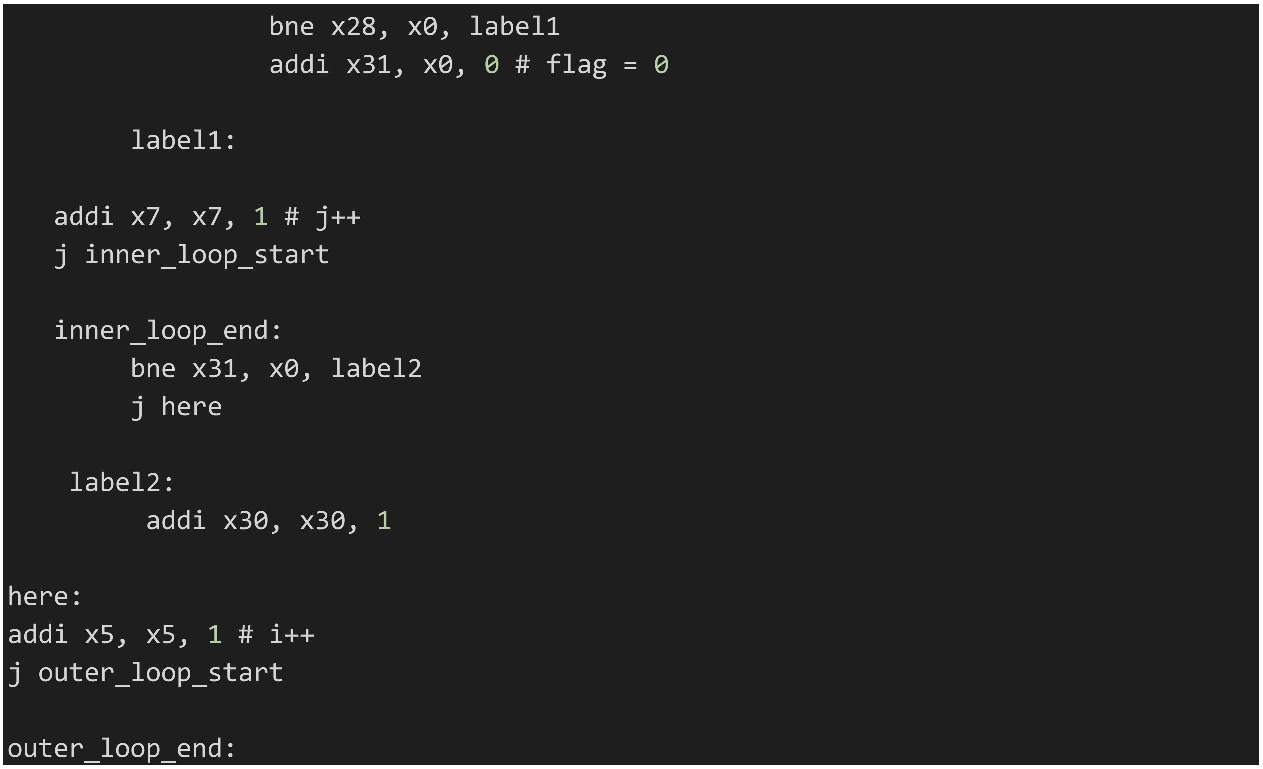
```
int main()
{
    int c = 0;
    int flag;
    for (int i = 2; i < 11; i++) {
        flag = 1;
        for (int j = 2; j <= (i / 2); j++) {
            if ((i % j) == 0) {
                flag = 0;
            }
        }
        if (flag) {
            c++;
        }
    }
    printf("Prime numbers are %d times from 1 to 10", c);

    return 0;
}
```

RISC-V Assembly Code:

```
addi x30, x0, 0 # c = 0
addi x31, x0, 0 # flag = 0;

addi x5, x0, 0x2 # i=2
addi x6, x0, 0xa # condition 10
outer_loop_start:
bge x5, x6, outer_loop_end # if (i >= 10)
addi x31, x0, 0x1 # flag = 1
   addi x7, x0, 0x2 # j=2
   div x9, x5, x7 # i/2
   inner_loop_start:
                 bgt x7, x9, inner_loop_end # if(j <= (i/2))
                 rem x28, x5, x7
```



```
                 bne x28, x0, label1
                 addi x31, x0, 0 # flag = 0

        label1:

   addi x7, x7, 1 # j++
   j inner_loop_start

   inner_loop_end:
        bne x31, x0, label2
        j here

    label2:
         addi x30, x30, 1

here:
addi x5, x5, 1 # i++
j outer_loop_start

outer_loop_end:
```

addi x30, x0, 0

000000000000	00000	000	11110	0010011

Machine Code: 0000 0000 0000 0000 0000 1111 0001 0011

Hexadecimal Code: 00000F13

addi x31, x0, 0

000000000000	00000	000	11111	0010011

Machine Code: 0000 0000 0000 0000 0000 1111 1001 0011

Hexadecimal Code: 00000F93

addi x5, x0, 0x2

000000000010	00000	000	00101	0010011

Machine Code: 0000 0000 0010 0000 0000 0010 1001 0011

Hexadecimal Code: 0020293

addi x6, x0, 0xA

000000001010	00000	000	00110	0010011

Machine Code: 0000 0000 1010 0000 0000 0011 0001 0011

Hexadecimal Code: 00A00313

bge x5, x6, outer_loop_end

0000001	00110	00101	101	11100	1100011

Machine Code: 0000 0010 0110 0010 1101 1110 0110 0011

Hexadecimal Code: 0262DE63

addi x30, x0, 0

000000000001	00000	000	11111	0010011

Machine Code: 0000 0000 0001 0000 0000 1111 1001 0011

Hexadecimal Code: 00100F93

addi x30, x0, 0

000000000010	00000	000	00111	0010011

Machine Code: 0000 0000 0010 0000 0000 0011 1001 0011

Hexadecimal Code: 00200393

div x9, x5, x7

0000001	00111	00101	100	01001	0110011

Machine Code: 0000 0010 0111 0010 1100 0100 1011 0011

Hexadecimal Code: 00272C4B3

bgt x7, x9, inner_loop_end

0000000	00111	01001	100	11000	1100011

Machine Code: 0000 0000 0111 0100 1100 1100 0110 0011

Hexadecimal Code: 0074CC63

rem x28, x5, x7

0000001	00111	00101	110	11100	0110011

Machine Code: 0000 0010 0111 0010 1110 1110 0011 0011

Hexadecimal Code: 0272EE33

bne x28, x0, label1

0000000	00000	11100	001	01000	1100011

Machine Code: 0000 0000 0000 1110 0001 0100 0110 0011

Hexadecimal Code: 000E1463

addi x31, x0, 0

000000000000	00000	000	11111	0010011

Machine Code: 0000 0000 0000 0000 0000 1111 1001 0011

Hexadecimal Code: 00000F93

addi x7, x7, 1

000000000000	00111	000	00111	0010011

Machine Code: 0000 0000 0000 0011 1000 0011 1001 0011

Hexadecimal Code: 00038393

bne x31, x0, label2

0000000	00000	11111	001	01000	1100011

Machine Code: 0000 0000 0000 1111 1001 0100 0110 0011

Hexadecimal Code: 000F9463

addi x30, x30, 1

000000000001	11110	000	11110	0010011

Machine Code: 0000 0000 0001 1111 0000 1111 0001 0011

Hexadecimal Code: 001F0F13

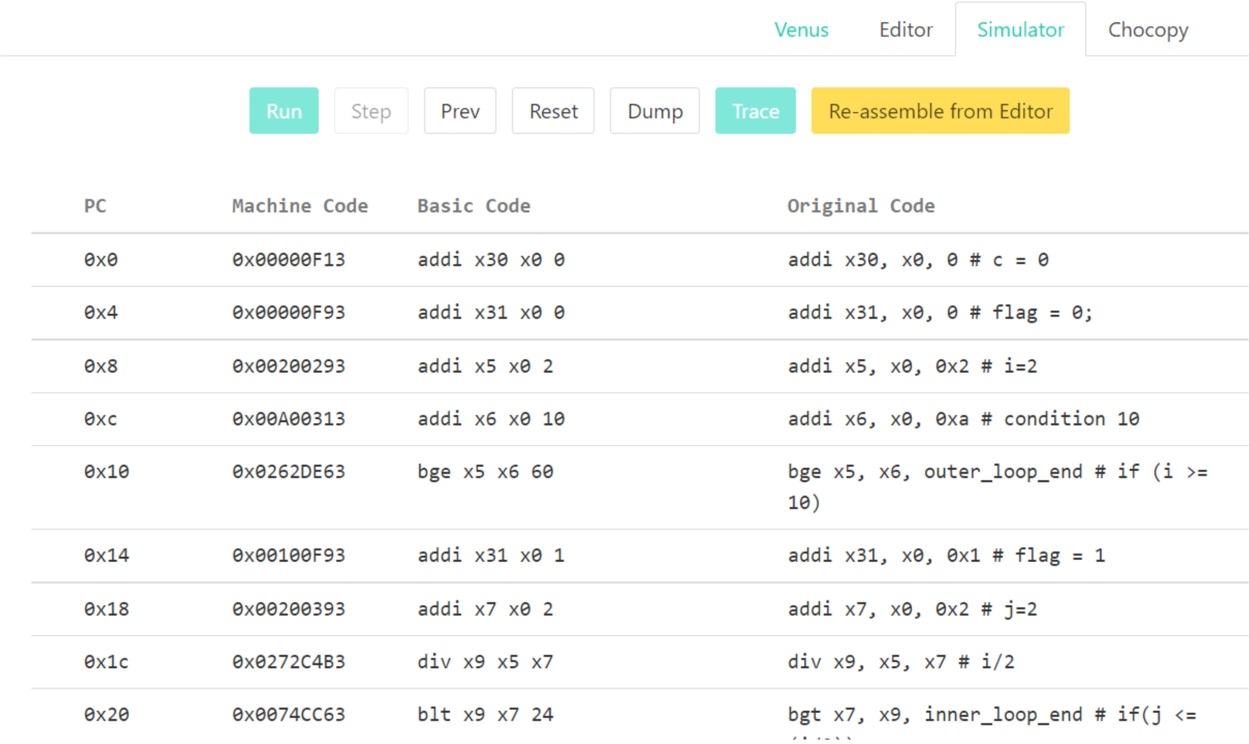
addi x5, x5, 1

000000000001	00101	000	00101	0010011

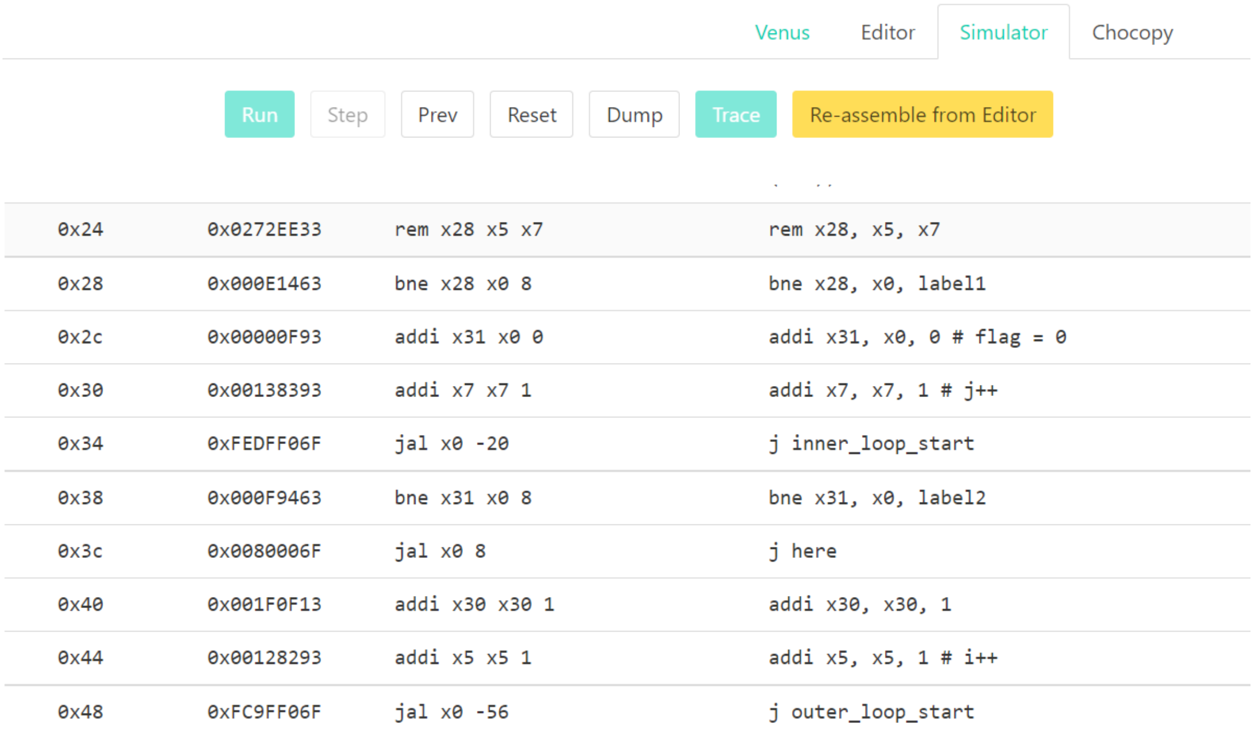
Machine Code: 0000 0000 0001 0010 1000 0010 1001 0011

Hexadecimal Code: 00128293

Venus Simulation:

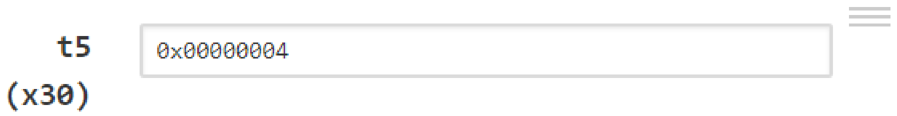


PC	Machine Code	Basic Code	Original Code
0x0	0x00000F13	addi x30 x0 0	addi x30, x0, 0 # c = 0
0x4	0x00000F93	addi x31 x0 0	addi x31, x0, 0 # flag = 0;
0x8	0x00200293	addi x5 x0 2	addi x5, x0, 0x2 # i=2
0xc	0x00A00313	addi x6 x0 10	addi x6, x0, 0xa # condition 10
0x10	0x0262DE63	bge x5 x6 60	bge x5, x6, outer_loop_end # if (i >= 10)
0x14	0x00100F93	addi x31 x0 1	addi x31, x0, 0x1 # flag = 1
0x18	0x00200393	addi x7 x0 2	addi x7, x0, 0x2 # j=2
0x1c	0x0272C4B3	div x9 x5 x7	div x9, x5, x7 # i/2
0x20	0x0074CC63	blt x9 x7 24	bgt x7, x9, inner_loop_end # if(j <=



0x24	0x0272EE33	rem x28 x5 x7	rem x28, x5, x7
0x28	0x000E1463	bne x28 x0 8	bne x28, x0, label1
0x2c	0x00000F93	addi x31 x0 0	addi x31, x0, 0 # flag = 0
0x30	0x00138393	addi x7 x7 1	addi x7, x7, 1 # j++
0x34	0xFEDFF06F	jal x0 -20	j inner_loop_start
0x38	0x000F9463	bne x31 x0 8	bne x31, x0, label2
0x3c	0x0080006F	jal x0 8	j here
0x40	0x001F0F13	addi x30 x30 1	addi x30, x30, 1
0x44	0x00128293	addi x5 x5 1	addi x5, x5, 1 # i++
0x48	0xFC9FF06F	jal x0 -56	j outer_loop_start

Final register value



t5 (x30) 0x00000004

*** ------------------------------------- ***