AYESHA ABBAS **CS211245**

LAB NUMBER # 03

LAB TITLE: Introduction to Assembly Language & Venus Simulator

Literature Review

In this lab we have learned about basic Assembly language and machine code.

Assembly Language

An assembly language statement is a line of text that translates into a single machine instruction. Assembly Language is expressed in a more human readable form than the binary instructions and names are allowed for memory locations, registers, operations etc.

Machine Language

Machine language is the language understood by a computer. It is very difficult to understand, but it is the only thing that the computer can work with. All programs and programming languages eventually generate or run programs in machine language.

VENUS

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.

LAB EXERCISE#01

Translate the Given High Level Code to Assembly Language

High - Level Code	RISC-V Assembly Code

a = b − c;
f = (g + h) − (i + j);	

ASSEMBLY

sub x8, x9 , x18

add x5, x19, x20

add x6, x21, x22

sub x7, x5, x6

LAB EXERCISE#02

Q. 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

MACHINE CODE

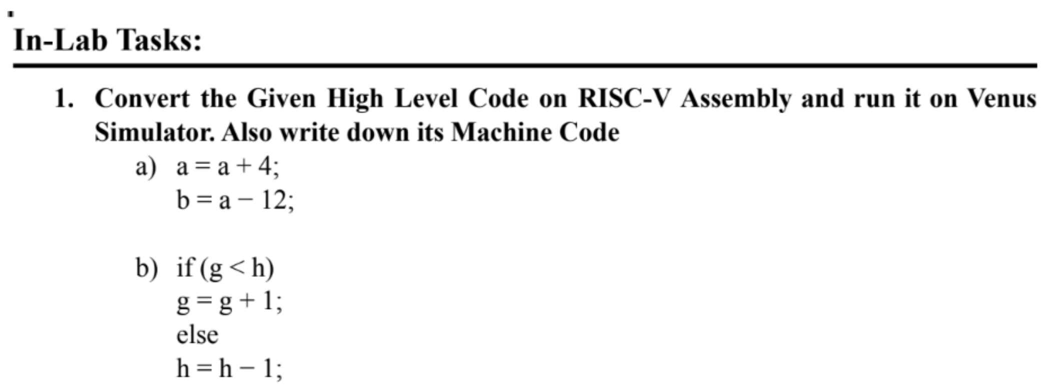
0000000 00110 00101 000 0100 10110011

000000000110 01000 111 01010 0010011

0000000 00111 00110 110 00101 0110011

0000000 01000 00110 001 01010 0010011

In-Lab Task#01



In-Lab Tasks:

1. **Convert the Given High Level Code on RISC-V Assembly and run it on Venus Simulator. Also write down its Machine Code**
   a) a = a + 4;
      b = a − 12;

   b) if (g < h)
      g = g + 1;
      else
      h = h − 1;

a) a=a + 4;
b = a – 12;

ASSEMBLY

```
addi x18,x5,0x6

addi x19,x18,-12
```

MACHINE CODE

```
0000000000110  00101  000  10010  0010011

111111110100  10010  000  10011  0010011
```

HEXA-CODE

```
00628913

FF490993
```

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

```
01394463

FFF98993

00190913
```

LAB ASSIGNMENT

Q. 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 = 10+6;

int b = 2*5;

int c = 4/2;
```

ASSEMBLY

```
addi x18,x18,0x10
addi x18,x18,0x6

addi x20,x20,0x2
addi x21,x21,0x5
mul x29,x20,x21

addi x27,x27,0x4
addi x26,x26,0x2
div x28,x27,x26
```

MACHINE CODE

```
000000001010  10010  000  10010  0010011
000000000110  10010  000  10010  0010011
000000000010  10100  000  10100  0010011
000000000101  10101  000  10101  0010011
0000001  10101  10100  000  11101  0110011
000000000100  11011  000  11011  0010011
000000000010  11010  000  11010  0010011
0000001  11010  11011  100  11100  0110011
```

HEXA-CODE

```
00A90913
00690913
002A0A13
005A8A93
```

```
035A0EB3

004D8D93

002D8D13

03ADBE33
```

Q . 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-PROGRAM CODE

```
int main() {
   int c = 0;
   int flag;
   for(int i=2; i<11; i++){
      flag = 1;
      for(int j=2; j<i; j++){
         if((i%j) == 0){
            flag = 0;
         }
      }
      if(flag){
         c++;
      }
   }
   printf("%d",c);
 return 0;
}
```

ASSEMBLY

```
j main

main:

addi x18, x18, 0x2    # i=2

addi x19, x19, 0x0    # c=0

addi x20,x20,0x1   # flag=1

addi x21,x21,0x2   # j=2

addi x24,x24,0xB   # 11

addi x23 , x23 , 0x1  # 1



#condition

counter:

  beq x20, x23,counteraddition #if flag==1 do c++



outer:

  addi x18,x18,0x1  #i++

  addi x21,x0,0x2  #j=2

  addi x20,x0,0x1  #flag=1

  BLT x18,x24,outerloop  #i<11

  j end


outerloop:

  blt x21,x18,innerloop   #j<i

  j counter


innerloop:

  rem x22,x18,x21    #i%j
  beq x22,x0,primefalse   #i%j==0 (if true means not prime)
  addi x21,x21,0x1  #j++
  j outerloop

primefalse:
  addi x20,x20,0x0  #flag=0
  j outer

counteraddition:
  addi x19,x19,0x1  #c++
  j outer



end:
```

MACHINE CODE

0000 0000 0010 1001 0000 1001 001 0011

0000 0000 0000 1001 1000 1001 1001 0011

0000 0000 0001 1010 0000 1010 0001 0011

0000 0000 0010 1010 1000 1010 1001 0011

0000 0000 1011 1100 0000 1100 0001 0011

0000 0000 0001 1011 1000 1011 1001 0011

0000 0011 0111 1010 0000 1100 0110 0011

0000 0000 0001 1001 0000 1001 0001 0011

0000 0000 0010 0000 0000 1010 1001 0011

0000 0000 0010000 0000 1010 0001 0011

0000 0001 1000 1001 0100 0100 0110 0011

0000 0001 0010 1010 1100 0100 0110 001

0000 0011 0101 1001 0110 1011 0011 0011

0000 0000 0000 1011 0000 0110 0110 0011

0000 0000 0001 1010 1000 1010 1001 0011

0000 0000 0000 1010 0000 1010 0001 0011

0000 0000 0001 1001 1000 1001 1001 0011

HEXA-CODE

```
00290913

00098993

001A0A13

002A8A93

00BC0C13

001B8B93

037A0C63

00190913

00200A93

00100A13

01894463

012AC463

03596B33

000B0663

001A8A93

000A0A13

00198993
```