INDIAN INSTITUTE OF TECHNOLOGY GOA

COMPUTER ARCHITECTURE LAB (CS 211)

Faculty: Dr. Sharad Sinha
Teaching Assistant: Prachi Kashikar
1st Contributor: Pavitra Bhade

LAB 02

In this course, you will write and simulate assembly language programs, to display characters on the console screen, using the MIPS32 simulator QtSpim. A number of system services, mainly for input and output, are available for use by your MIPS program. In this course, you will learn about these system services by printing characters on the console screen. You will also be using control flow instructions to perform the following exercise.

We will be splitting this lab exercise in to two parts:

PART A:
In this part, you will become familiar with the concept of printing characters on the console screen by working with a simple MIPS assembly language program to print 'Hello World'.

STEPS:

1. Open the text editor and type the demo code shown in Fig. 1

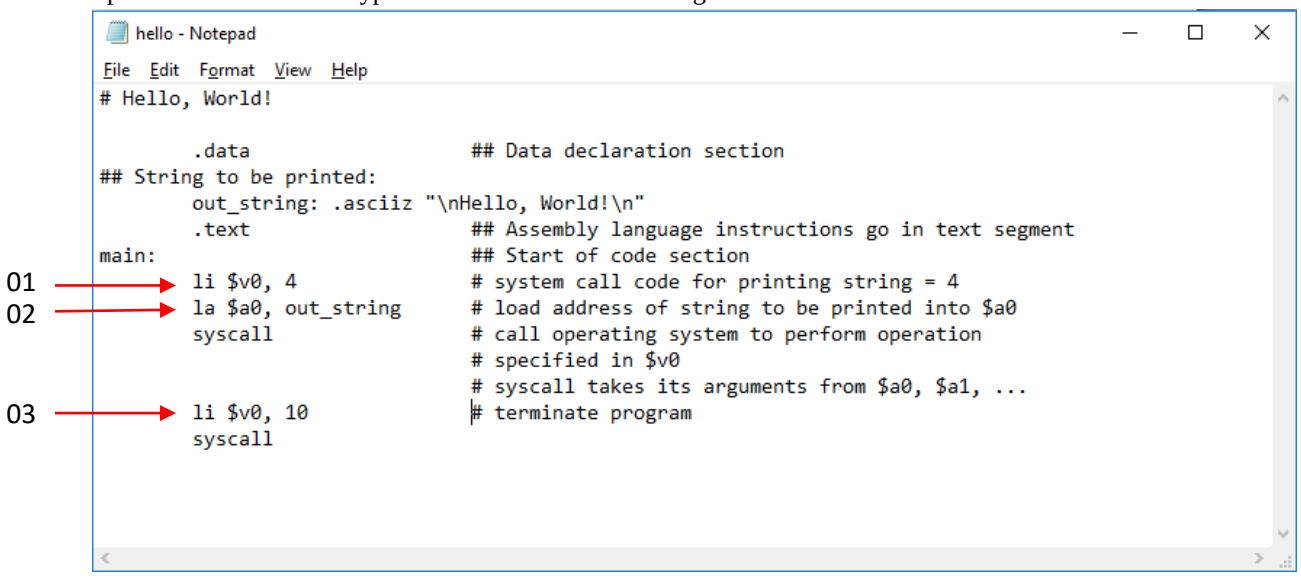

Fig. 1

To use SYSTEM call services, the following steps have to be followed:

a. Load the service number in register $v0 (Every service has a particular service number. See Table 1 for details).
b. Load argument values, if any, in $a0, $a1, $a2, or $f12 as specified.
c. Issue the SYSCALL instruction.
d. Retrieve return values, if any, from result registers as specified.

Service	System Call Code	Arguments	Result
print integer	1	$a0 = value	(none)
print float	2	$f12 = float value	(none)
print double	3	$f12 = double value	(none)
print string	4	$a0 = address of string	(none)
read integer	5	(none)	$v0 = value read
read float	6	(none)	$f0 = value read
read double	7	(none)	$f0 = value read
read string	8	$a0 = address where string to be stored $a1 = number of characters to read + 1	(none)
memory allocation	9	$a0 = number of bytes of storage desired	$v0 = address of block
exit (end of program)	10	(none)	(none)
print character	11	$a0 = integer	(none)
read character	12	(none)	char in $v0

Table. 1

In Fig 1, three instructions have been highlighted. They are related to the SYSCALL system services.
Instruction 01 loads the immediate value 4 into the register $v0. This is done to ensure that the register $v0 holds the service number 4, which is the system call code for printing strings.
Instruction 02 loads the address of the string to be printed, into the register $a0. The string to be printed is specified as out_string.
Once the system call is done to print the given string, the program has to be terminated.
Instruction 03 loads the immediate value 10 into the register $v0. 10 is the service number code to terminate the program.

2. Save this code and simulate it on the QtSpim simulator. Check the output on the console.

PART B:

In this part, you will write MIPS assembly language programs to understand control flow instructions.

1) Complete the following code snippet to add 10 numbers stored consecutively in data memory. Print the result.

```
.data
array:  .word  10,12,15,-10,13,82,-9,4,3,-7          #array={10,12,15,-10,13,82,-9,4,3,-7}
length:  .word 10                                    #load the length of the array as 10
sum: .word 0                                         #initialise sum to 0
.text
main:
la $t3, array                          # load base address of the array
# $t3 has the base address of data. All the subsequent data can be accessed using respective offset values.
#Add your code here
```

Count the total number of machine instructions executed to complete this task.

2) Include the following numbers in the array data segment of question 1.
10,20,30,40,50,77

Now you have 16 numbers residing in the array (data memory). Add these numbers and display the result.
Count the total number of instructions.
Compare and analyse the relation between the number of data elements and total number of machine instructions executed.

3) Compute the Euler Phi function for the number 21.

Euler's Phi function for an input n, denoted as (phi(n)) is the count of numbers in {1, 2, 3, …, n} that are relatively prime to n, i.e, the numbers whose GCD (Greatest Common Divisor) with n is 1.
Examples:
phi(1)=1, (gcd(1,1)=1)
phi(2)=1, (gcd(1,2)=1, but gcd(2,2)=2)
phi(3)=2, (gcd(1,3)=1, gcd(2,3)=1, gcd(3,3)=3)
phi(4)=2, (gcd(1,4)=1, gcd(2,4)=2, gcd(3,4)=1, gcd(4,4)=4)
phi(5)=4, (gcd(1,5)=1, gcd(2,5)=1, gcd(3,5)=1, gcd(4,5)=1, gcd(5,5)=5)

(Hint: The logic for your code would be as follows

```
phi = 0;
trial = 1;
while ( trial < N)          #where N is the number under consideration (given : 21)
{
   if ( gcd(N,trial) == 1 ) phi++;
} )
```

The computed value, which is phi(21), should be printed on the screen.