**INDIAN INSTITUTE OF TECHNOLOGY**

**GOA**

**COMPUTER ARCHITECTURE LAB (CS 211)**

Faculty : Dr. Sharad Sinha

Teaching Assistant: Prachi Kashikar and Pavitra P. Bhade

**LAB 05**

In this course, you will study the floating point operations in MIPS processor.

This lab exercise will split in to two parts:

In the first part, you will understand the concept of floating point operations on the MIPS processor. In the second part, you will write assembly programs to understand the same.

**PART A**

The MIPS processor has a floating point coprocessor (numbered 1) also called as **F**loating **P**oint **A**ccelerator (FPA) that operates on single precision (32-bit) and double precision (64-bit) floating point numbers. This coprocessor has its own registers, which are numbered $f0--$f31. Modern MIPS chips include floating point operations on the main processor chip.

MIPS also has hardware for double precision (64 bit) floating point operations. For this, it uses pairs of single precision registers to hold operands. There are 16 pairs, named $f0, $f2, ... , $f30. Only the even numbered register is specified in a double precision instruction; the odd numbered register of the pair is included automatically. Some MIPS processors allow only even-numbered registers ($f0, $f2,...) for single precision instructions. However SPIM allows you to use all 32 registers in single precision instructions.

**Representation of Floating Point Numbers in the Floating Point Registers:**

Consider the number 152.3

This is a floating point number. The representation of floating point numbers in MIPS is based on their scientific notation. The scientific notation of this number is as follows:

1.523 x 102

Mantissa Exponent

This is base 10 representation. Its binary equivalent will be saved in the registers(base 2).

In the register, it will be represented as shown in Fig 1

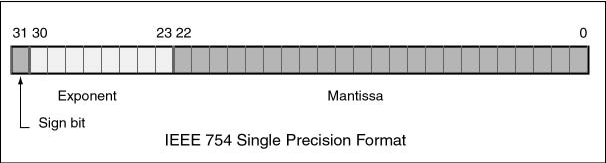


Fig: 01

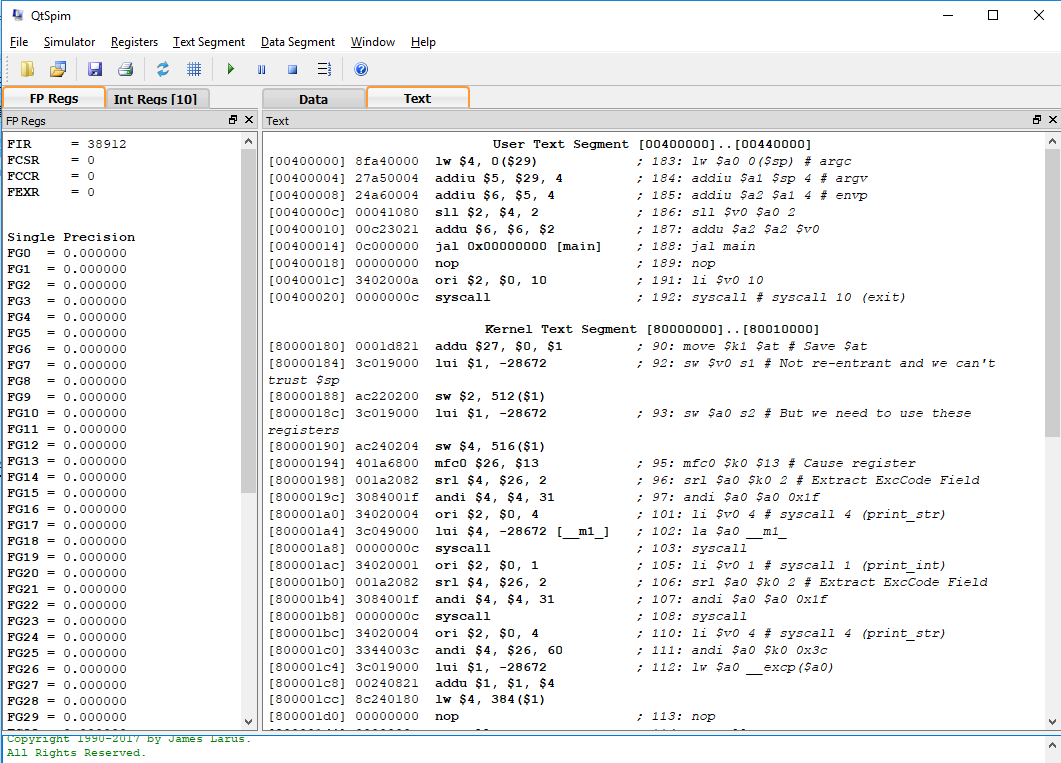
The sign bit is 1 for negative numbers and 0 for positive numbers.

The mantissa represents the 23-bit binary fraction part of a 24-bit number that starts with a "1". The 0th  bit is presumed to contain a 1 and *is not present in the mantissa*.  This trick gives us 24 bits of precision with only 23 bits. For example, the binary number 1.11110000101101101010001 is represented as 11110000101101101010001

The exponent is expressed using a biased integer. The true exponent is after subtracting 127 from the biased exponent (127 is referred to as the bias). The biased integer is encoded using 8-bit unsigned binary.

* A biased exponent of +127 represents the actual exponent 0
* A biased exponent of +128 represents the actual exponent 1
* A biased exponent of +126 represents the actual exponent -1

You can see these registers on the simulator, in the registers area, under that tab named FP Regs as highlighted in Fig 2.

****Fig: 02

**Part B:**

Write MIPS Assembly language programs to study the working of floating point operations in the processor. Use single stepping to understand the changes in the respective registers.

1. Evaluate the expression 2.5x2+4.3x+5. Take x as input from the user. Display the result.
2. Find the square root of a number entered by the user. Apply Newton’s method to perform the calculations (Accurate up to 5 places of decimal).

Hint: Newton's method is a way to compute the square root of a number. Say that n is the number and that x is an approximation to the square root of n.

Then:

x' =(1/2)(x + n/x)

x' is an even better approximation to the square root.

If x reaches the exact value, it stays fixed at that value.