# COL-216
# ASSIGNMENT-3
# MIPS - INTERPRETER

Made By

*Dishant Dhiman 2019CS10347*

*R Hari Shankar 2019CS10386*

## DESIGN DESCRIPTION

We created an interpreter in C++ that reads MIPS assembly language program as input and executes the operations indicated by instruction. We have used an array (registers) to keep track of the register value during program execution . Initially all these values are 0. “clockCycles” and “countOfInstructions” keep track of the number of cycles and instructions respectively.

We first read the file line by line and recognise labels, comments and instruction parameter in the process and store the labels in a map. Now we iterate upon the instructions using a program counter (PC) and according to the instruction given (add , sub , mult , beq , bne , slt , lw , sw , addi , j ) the required conversion to the register value and memory is done.

“getRegister” function is used to locate the value of the register. printRegister is used to print the register values at every instruction .

If some erroneous instruction is found then an error is thrown. “lw” and “sw” make changes to the memory so a memory array of size $2^{20}$ bytes is used to keep check that memory doesn’t overflow.

The register values are returned after every instruction in hexadecimal format.

Once all the instructions are read and computed the number of cycles and number of each instructions executed is returned.

*Comments have been added to the code to make it as easily readable and understandable as possible.*

## HOW TO RUN THE PROGRAM

1) Set your directory to the directory in which the program is present using your terminal/console.
2) Now type g++ interpreter.cpp -o interpreter.out
3) Now type ./interpreter.out <input file>
   Eg) ./interpreter.out input.txt
4) The input file is the file that contains the MIPS instruction.

5) The output will be shown in your console.

## OUTPUT FORMAT

If the input is of correct format then the output will be displayed else an error message will be displayed stating that the input format is incorrect.

FORMAT:

All the 32 Register values are returned after every instruction.

After all the instruction are evaluated the number of clock cycles and number of times each instruction is executed is printed in the console.

## TESTING STRATEGY

To check the correctness of our code as an interpreter of the mips assembly language we have used exhaustive test cases so that our program runs correctly in all cases (including corner cases).

Test Cases:

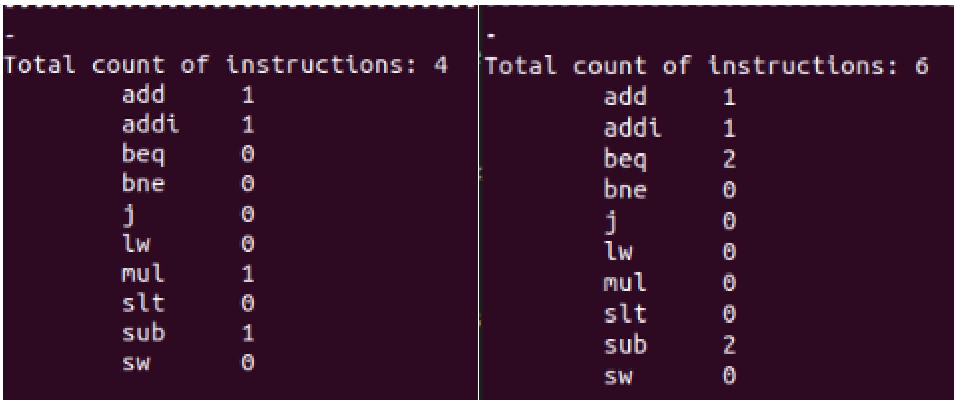
- When correct format of file is given using only the add ,sub ,mult ,beq ,bne ,slt ,lw ,sw ,addi ,j instructions .

Invalid Input Cases:

- When input file is not found.
- When input file is empty.
- Incorrect register name is given.
- Syntax error for various instruction.
- Memory used more than $2^{20}$ bytes.
- When instruction other than add ,sub ,mult ,beq ,bne ,slt ,lw ,sw ,addi ,j is used.

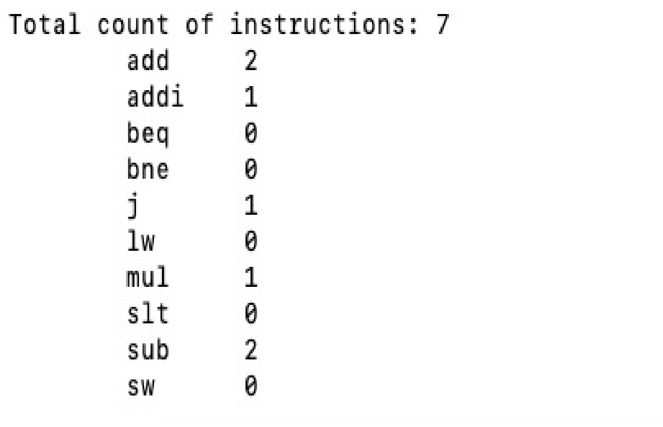
*All the testcases used to check correctness of program are given in the file testcases.txt.*

The expected output are verified both manually and using QtSPIM MIPS.



```
-
Total count of instructions: 4
        add     1
        addi    1
        beq     0
        bne     0
        j       0
        lw      0
        mul     1
        slt     0
        sub     1
        sw      0
```

```
-
Total count of instructions: 6
        add     1
        addi    1
        beq     2
        bne     0
        j       0
        lw      0
        mul     0
        slt     0
        sub     2
        sw      0
```



```
Total count of instructions: 7
        add     2
        addi    1
        beq     0
        bne     0
        j       1
        lw      0
        mul     1
        slt     0
        sub     2
        sw      0
```

Fig: Some examples of test cases