LAB ASSIGNMENT-02

Computer Architecture

Ritik Tiwari

B21CS098

MARS SOFTWARE

MARS stands for "MIPS Assembly and Runtime Simulator." It is a piece of software made for writing in MIPS assembly language. One of its best features is that it has an integrated development environment (IDE) that lets you write, assemble, and run MIPS assembly code. This makes it a useful tool for workers and students who want to learn or work with MIPS architecture.

Key features of MARS

1) Assembler: Users can write MIPS assembly code in MARS and then have it put together into machine code that can be run. It checks for errors in real time and highlights incorrect grammar to help you find problems in the code.
2) Runtime Simulator: There is a runtime simulator in the program that can run MIPS machine code. This model lets users see the MIPS CPU in great detail, including the program counter, registers, and memory. They can then step through their code, set breakpoints, and watch the program run.
3) Interactive Interface: The MARS software incorporates a user-friendly graphical interface with a code editor, an output terminal, and windows for monitoring CPU and memory operations during simulation.
4) Debugging Tools: The MARS suite provides debugging features including single-step execution, breakpoints, and variable watches. These utilities facilitate the process of detecting and resolving defects in MIPS assembly code.
5) Support for Syscalls: MARS provides a variety of system calls (syscalls) that emulate input/output (I/O) tasks, such as keyboard reading or console writing, frequently needed in MIPS assembly applications.

MIPS PROGRAMMING

MIPS programming, short for Microprocessor without Interlocked Pipeline Stages, is the act of developing software using the MIPS assembly language tailored for programming processors that are based on the MIPS architecture. MIPS is a prevalent RISC (Reduced Instruction Set Computer) architecture.

Overview of the MIPS Architecture:

1) RISC Architecture: MIPS is a RISC architecture, meaning it uses a small, highly optimized set of instructions. This simplicity allows for faster instruction execution and easier instruction pipelining.
2) 32 Registers: MIPS processors typically have 32 general-purpose registers (`$0` to `$31`), where most computations take place. Register `$0` is hardwired to the value 0.
3) Load/Store Architecture: In MIPS, arithmetic operations can only be performed on data stored in registers. To work with memory, data must first be loaded into a register, manipulated, and then stored back to memory if needed.
4) Fixed Instruction Length: Each MIPS instruction is 32 bits long, making instruction decoding straightforward and uniform.

To install the MARS software on your PC, you must have Java installed, and it should be the latest version. Initially, I had Java version 18 installed, but I was unable to run the MARS software until I upgraded Java to the latest version. After upgrading Java, the software ran without issues. To download the MARS JAR file, visit the official MARS website. The link is mentioned in the "Resources Used" section.

The assignment consists of two components. I am required to execute mathematical operations in task 1, while in task 2 I am tasked with creating the payroll system.

Task 1:

1) Subtask 1: addition of two numbers that are stored in registers.
2) Subtask 2: Multiplication of two numbers that are stored in registers.
3) Subtask 3: Subtraction of two numbers that are stored in registers.

   In the above three tasks, they have the same logic; just I have to change the instruction name, like for addition, the instruction is add, while for subtraction, the instruction is sub and for multiplication, the instruction is mul.

   Here is the basic procedure for the above three subtask codes.
   1) Load integers
   2) Instruction (add, sub, mul)
   3) Program Exit

4) Subtask 4: This task is to take the user input on what operation he wants to perform and then take two input numbers on which chosen operation was performed.

   Here is the basic procedure of the code, which I have written.
   1) Prompt for operation
   2) Prompt for numbers
   3) Perform Operation
   4) Print result
   5) Program Exit

Task 2: This involves making the employee payroll calculator, which involves the employee's details and for each employee we get three inputs from the user: The regular hours worked by the employee, The number of overtime hours worked, The hourly wage of the employee. In this question, my approach is to take the number of employees from the user and then for each employee I have taken the above-mentioned inputs. After that, I have calculated the payroll calculation, which has three components. 1) Gross Salary 2) Deduction calculation 3) Net Salary calculation. Here is the procedure to calculate all these calculations.

1) Gross Salary Calculation: This involves two things: the salary from the regular hours worked and the overtime hours worked. So if we calculate these separately and add them together, we easily get the gross salary. So I have done the same in my code: first I calculate the regular salary, which will be (number of regular hours) * (hourly wage) and to calculate the overtime salary, it will be (number of overtime hours) * (hourly wage) * 1.5, which is the bonus of working the extra hours.
2) Tax Deduction Calculation: This involves also two things. 1. Tax deduction 2. Insurance Deduction. Both have different parameters to calculate so to calculate the tax deduction, this will be x percent of gross salary, where x is the (last two-digit roll number) mod 30 and in my case, my roll number is 98%30, which is equal to 8. For the calculation of insurance, there is a slight change in the parameter, which is (last two digits of roll number + 8) mod (30) and in my case it’s 16.
3) Net Salary Calculation: This will be a simple calculation of gross salary minus total deduction (tax + insurance deduction).

So this is the overall procedure for each of the calculations. The basic steps that I have done in my code are given below. I have written the overall step-wise procedure for what I have done.

1) Initialize and load constants
2) Employee loop
3) Gross Salary Calculation
4) Deduction Calulation
5) Net Salary Calculation
6) End of the loop
7) Program Exit

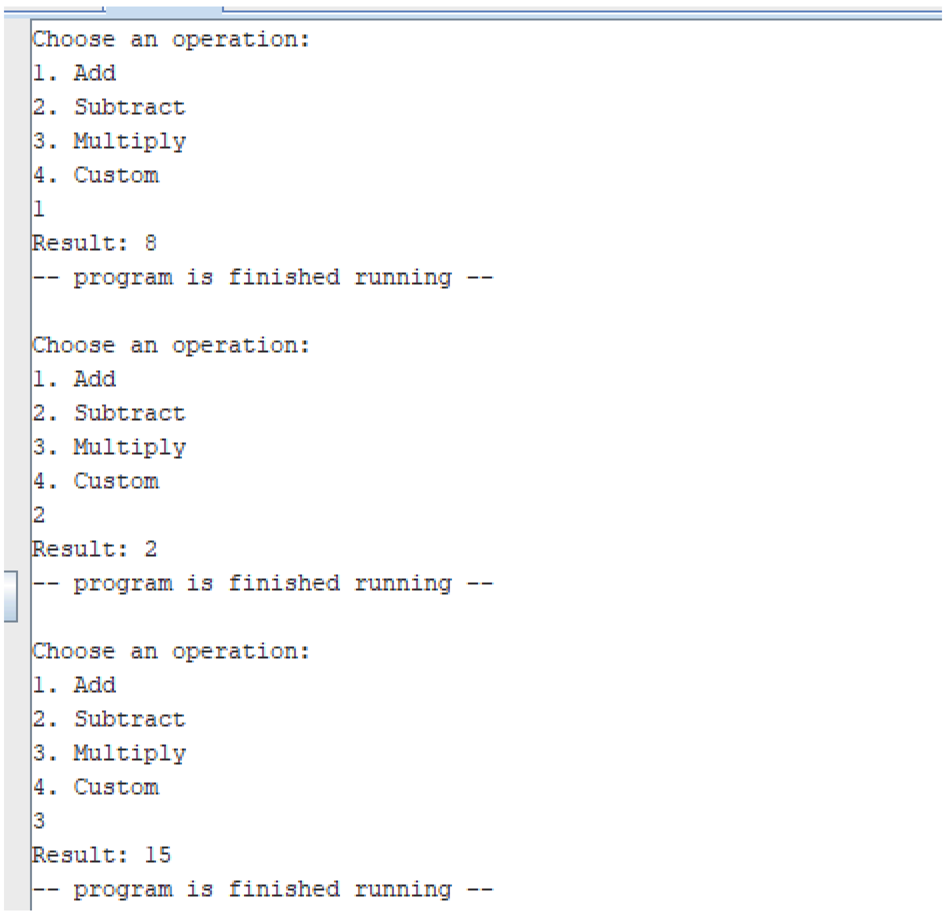
So this is all about task 2 and in this assignment I have to do only these two tasks with the code I have also submitted.

Challenges Faced: I have not faced such a specific challenge but yes, this is the new language, which I have learned so familiar with the syntax it takes a little bit of time otherwise the MIPS programming is completely logical.

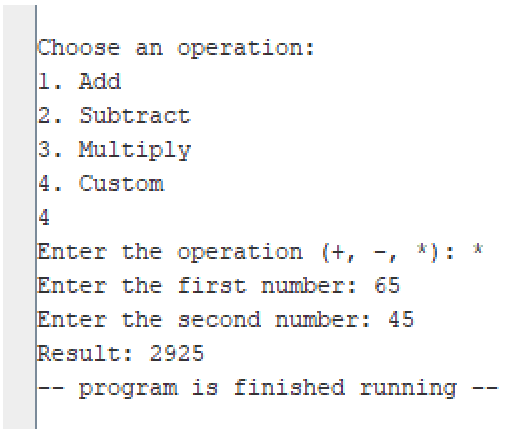
Program Output: For the first task where I have to perform the arithmetic operations, the outputs will be based on the operation performed and we can also see the outputs in the simulator, each by step if the program teaks multiple steps and also values how changes in the register. While the second task is more visualized because it evolves multiple steps to get into the final result. The second task is making more intensive use of registers because it involves extensive calculation. Both the task outputs can be visualized easily.

Please find the screenshots of the outputs of both tasks:

Task 1 Output

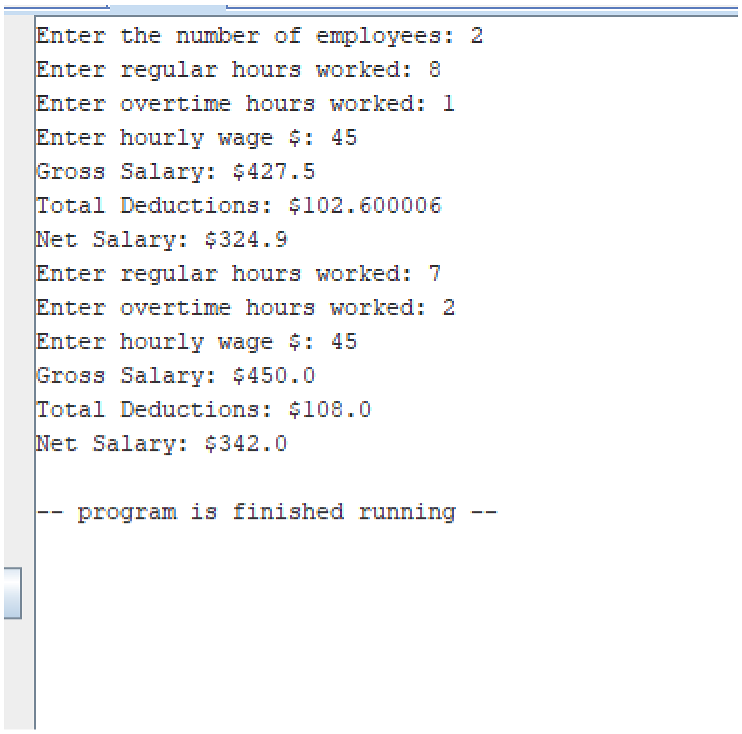


```
Choose an operation:
1. Add
2. Subtract
3. Multiply
4. Custom
1
Result: 8
-- program is finished running --

Choose an operation:
1. Add
2. Subtract
3. Multiply
4. Custom
2
Result: 2
-- program is finished running --

Choose an operation:
1. Add
2. Subtract
3. Multiply
4. Custom
3
Result: 15
-- program is finished running --
```



```
Choose an operation:
1. Add
2. Subtract
3. Multiply
4. Custom
4
Enter the operation (+, -, *): *
Enter the first number: 65
Enter the second number: 45
Result: 2925
-- program is finished running --
```

<u>Task 2 Output</u>



```
Enter the number of employees: 2
Enter regular hours worked: 8
Enter overtime hours worked: 1
Enter hourly wage $: 45
Gross Salary: $427.5
Total Deductions: $102.600006
Net Salary: $324.9
Enter regular hours worked: 7
Enter overtime hours worked: 2
Enter hourly wage $: 45
Gross Salary: $450.0
Total Deductions: $108.0
Net Salary: $342.0

-- program is finished running --
```

Learning from the assignment and use of MIPS in real-world scenarios:

So this assignment focuses on the basics of the introduction of the MIPS programming, where I have learned the different registers, where some of them are for special purposes while others are for general purposes, and also learn the syntax involved in MIPS, like how to access the registers value stored in them, and also what are the different instructions the MIPS have. So this assignment involves basic learning and this also contains one real-life problem: the payroll, which may be used by small businesses. We can also make the calculator using MIPS, which has basic operation so these are some of the real-life uses of the MIPS programming.

Resources Used:

1) https://courses.missouristate.edu/kenvollmar/mars/download.htm
2) https://student.cs.uwaterloo.ca/~cs241/mips/mipsasm.html
3) https://learnxinyminutes.com/docs/mips/