Dept. of CSE, Bennett University

ECSE217L – Microprocessor and Computer Architecture

Assignment – 3

The MIPS assembly language simply refers to the processor's assembly language. Microprocessor without Interlocked Pipeline Stages (MIPS) is an acronym for Microprocessor without Interlocked Pipeline Stages. It's a MIPS Technologies-developed reduced-instruction set architecture.
To begin, you first require an excellent Integrated Development Environment to compile and execute your MIPS assembly language code. MARS (MIPS Assembler and Runtime Simulator) will be used in this lab for this purpose.
You will learn how MARS MIPS works and will simulate some sample assembly programs. This will help you to learn assembly programming in depth.

MARS MIPS (https://courses.missouristate.edu/KenVollmar/mars/) is lightweight IDE for programming in MIPS assembly language.

It can be obtained from https://courses.missouristate.edu/KenVollmar/mars/download.htm.

Download and double click to open it (Tutorial is available in 'Help' tab).

1. Using MARS simulator, write a basic assembly language code for printing a single character using the '.byte' directive. (To learn more about all of the directives, go to the Help tab).

   Example: A single character 'B' can be printed using '.byte' directive.

2. Insert a String like "The first String of MIPS Code" in the same file of Question 1. **Print the new string in a new line**. Use '.byte' directive to print the given string "The first String of MIPS Code". If it is showing some errors, identify the correct directive to solve the problem.

   Example: Say the output of your previous program was: **B**
   Your new output will be: **B**
   The first String of MIPS Code

3. Assume the values of b, c, d in the following equations according to your choice and convert following set of arithmetic operations into assembly instructions.

   given code:
   ```
   a = b + c - d;
   a = a + 5;
   e = a * 3
   ```

   Example: If value of b, c, d are taken as 7, 8, 9 in the program then after
   1st Instruction: a = 7 + 8 – 9 = 6
   2nd Instruction: a = 6 + 5 = 11
   3rd Instruction: e = 11 * 3 = 33

4. Repeat Question No. 3 by taking input given by user at runtime.
   Note: User needs to give the inputs after assembling the program.

5. Write the assembly language code which will check a number whether it is even or odd (input should be given by user).
   Example: Enter the number to be checked for even or odd: 23
   23 is odd.

Submission Instructions:

- Submit your .asm files in your respective batches in LMS. Save all the files as per the format **RollNo_Lab#_QuestionNo.asm (Example: E19CSE632_Lab5_Q2.asm)**.
- Write your Name and Roll No. as comment before starting of each program.
- Make it sure that in each program, you have mentioned enough comments regarding the explanation of program instructions.
- In the LMS please submit in your respective batch's submission portal. Submission in other batch's submission portal will not be checked.
- Write your Name and Roll No in the .m file itself (Use # to insert comment lines). Without this you will score zero for that particular question.
- Late submission will lead to penalty.
- Any form of plagiarism/copying from peer or internet sources will lead penalty.