ECE3700J Introduction to Computer Organization
Lab 1 RISC-V Assembly Instructions

Purpose

This lab is intended to help you have a better understanding of the RISC-V assembly instructions, and get familiarized with Ripes which is a visual computer architecture simulator and assembler built for the RISC-V instruction set architecture.

Tasks

1. Read Ripes Introduction at https://github.com/mortbopet/Ripes/wiki/Ripes-Introduction to learn about the software. Learn more information about Ripes at the wiki page. Everything else about the software can be found at https://github.com/mortbopet/Ripes.

2. Download and install Ripes simulator on your computer. You can find the latest release (v.2.2.2) that suits your operating system on your computer at https://github.com/mortbopet/Ripes/releases/tag/v2.2.2. Play with the software and get familiarized with the software environment.

3. Learn RISC-V assembly syntax at https://github.com/riscv-non-isa/riscv-asm-manual/blob/master/riscv-asm.md. In the software, load example assembly files and learn from the example code.

4. Write a short assembly program. Create a string of characters of your choice in the `.data` segment of memory using `.string` directive. Then copy the string to a different memory section starting at address 0x10000100. Click the Select Processor button on the upper left corner, and select 32-bit Single-cycle processor to debug your program.

Deliverables

This is a 1-week lab. The full score for this lab is 100 points.

1) Demonstrate your program to the TAs before your lab session ends. Go through the program step by step and show corresponding changes in the registers and memory.
2) Upload the source file(s) on Canvas by **22:00 pm, September 24, 2022**.

This is an individual assignment. Your work must be submitted electronically to Canvas before the specified due date. Late submission will result in 0 point for the corresponding deliverables. Source code must be submitted before a grade for this lab can be assigned.

Grading Policy

Demonstration: 80%
Source code: 20%