EC605: Computer Engineering Fundamentals Fall 2024

Lab 4: RISC-V Assembly

Goal

1. Simple programming in RISC-V assembly.

Overview

In this lab you will program and debug a working assembly project, while implementing simple functions in RISC-V assembly.

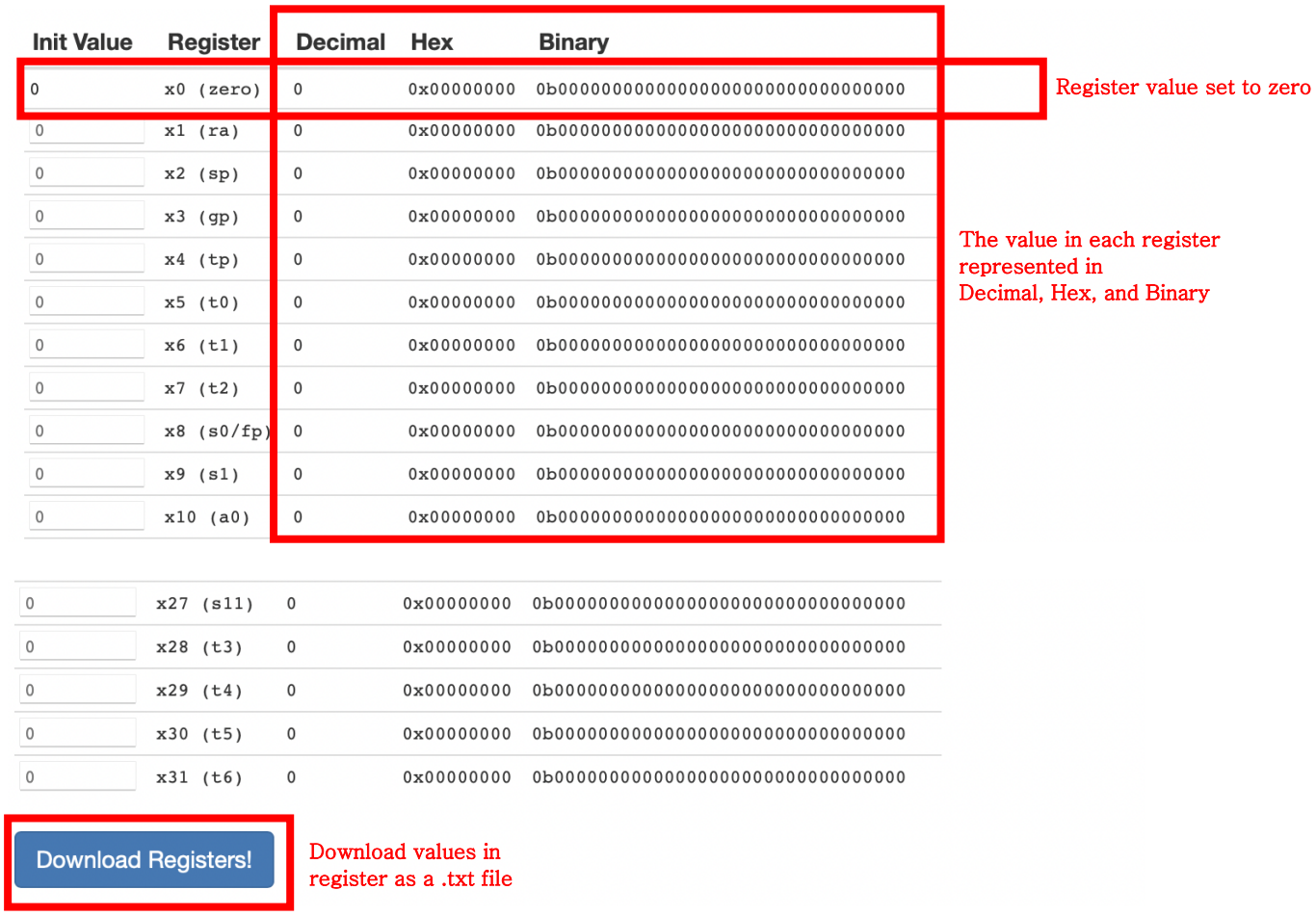
Tasks

Task 0: RISC-V Simulator Environment Tutorial

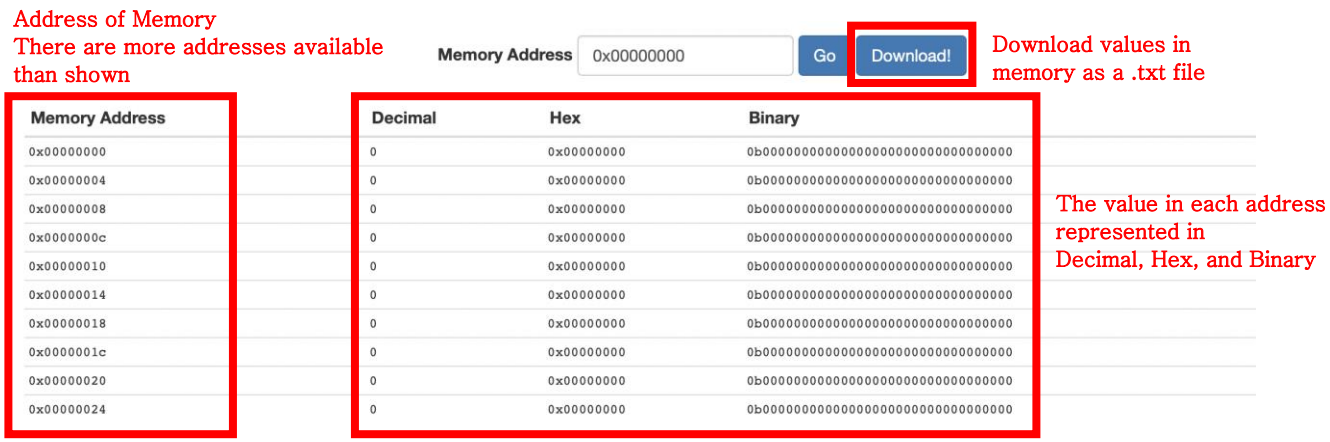
1. Access RISC-V simulator through the following link:
   https://www.cs.cornell.edu/courses/cs3410/2019sp/riscv/interpreter/#

2. Description of the Simulator:

   2-1. Register



2-2. Memory



2-3. Input code window

3. Tips on running your program:
   - **We strongly recommend you use a text editor to save code as you go**
   - Do not use tabs, since the simulator does not accept these, but use spaces instead.
   - Use the standard format for loads and stores, e.g. 'lw rs1, imm(rs2)'.
   - Only use the instructions that are supported by the simulator (not all RISC-V instructions are supported).
   - Do not modify register values manually.
   - When you download the content of the memory, it will be printed out to the point where the rest of the values are zero. Note that more memory is available than what is shown in the simulator window. Do not modify the formatting/spacing of the memory contents.
   - Place labels on separate lines from instructions, for example:
     LOOP:
     add x2, x3, x5
     beq x0, x0, LOOP

Task 1: Array Initialization

1. Implement a Seq_array task to store an array starting at memory address 0x80. The array should be initialized to contain the first 32 elements of the Fibonacci sequence, defined as follows:

   Seq[0] = 0; Seq[1] = 1; Seq[i] = Seq[i-2] + Seq[i-1] for i > 1.

   Your code should utilize a loop and consist of instructions from the green card, along with immediate instructions.

2. Test your code on the RISC-V simulator and download the final program and memory contents after your code successfully runs.

Task 2: Array Copy and Encrypt

Note: X-OR encryption is a simple encryption technique where each element of data is *X-OR*'d with a key before storage. Original data is recovered by simply performing the X-OR operation with the key again. i.e. A ^ B = C ➔ C ^ B = A, where A is an element of data, B represents the key, and C is its encrypted value

1. Implement an Array_copy_encrypt task, such that elements of a new array beginning at memory address 0x00 are the encrypted values of the Task 1 array **in reverse order**. To receive credit for Task 2, you must operate on the array from task 1 by loading its elements from memory. Use your birthyear as the encryption key, or pick a random number between 1969 and 2006.

   seq_encrypted[0] = FibSeq[31] ^ <key > ; seq_encrypted[1] = FibSeq[30] ^ <key> ; etc.

   Note: Execute tasks 1 & 2 as a single program. You should not overwrite task 1 in memory.

2. Test your code on the RISC-V simulator and download the final memory contents after your code successfully runs (download entire memory after both tasks have run).

Deliverables

1. Copy and paste your well commented code and the results for each task in the Lab4_(BUEmailID).txt file given on Blackboard, using the provided format. Rename the file to include your BU email ID and submit it on Blackboard.