Lab 09: Machine Language
CSE-031-01 **Points: 20**

Overview

Since computer hardware can only communicate in 0's and 1's, our programs written in MIPS must be translated into machine code containing only 0's and 1's so that they can be executed. In this lab, we will practice the conversions between MIPS and machine code.

Getting Started

Before we begin any activities, create a directory (`Lab_9`) inside the `CSE31` directory we created in the first lab. You will save all your work from this lab here. **Note that all the files shown in green below are the ones you will be submitting for this assignment.**

You must have a clear idea of how to answer the TPS questions before leaving lab to receive participation score.

MIPS ↔ Machine Code

TPS (Think-Pair-Share) activity 1: Discuss questions 1 – 6 (50 minutes) while paired with your classmates assigned by your TA (you will be assigned to groups of 3-4 students) and record your answers in a text file named `tpsAnswers.txt` under a section labelled "TPS 1" (***you will continue to use this file to record your answers to all the TPS questions that follow in the lab handout***):

1. Download "**MIPS_Reference_Sheet**" from CatCourses. We will need to refer to this sheet to complete all the exercises in this lab.
2. Load `proc1.s` in MARS and study the code. This is similar to `compare.s` from Lab 06.
3. After assembling the program, study the Text Segment window and see how your source code is translated into True Assembly Language (Basic) as well as machine code (Code).
4. In true assembly language, every single instruction can be translated into a machine instruction. How many bits does a machine instruction contain?
5. To utilize the limited number of bits efficiently, all machine instructions are categorized into different types (or formats). How many types are there? What are they? Give 2 operations for each type as examples.
6. Now, locate the instruction in line #7 of `proc1.s`. Let us translate this instruction into machine code.
   a. What instruction type is this? How many fields does this type of instruction have? What are the names of these fields?
   b. Refer to the **MIPS sheet**, what is the value of the ***opcode*** of this instruction in **hex**? What register is `rs`? What is the value of this register in **hex**? What register is `rt`? What is the value of this register in **hex**? What is the value of ***immediate*** in **hex**?
   c. Construct the machine code of ***line #7*** using the values obtained from **part b**. Write your answer in both **binary** and **hex** formats. You can verify your answer with the Code column in Text Segment window.
7. Now, let us convert a machine code to a MIPS instruction. Locate address `0x00400010` from the Text Segment window.
   a. What is the machine code at this address in **hex**? Convert this code into **binary**.
   b. From the binary version of this machine code. What is the instruction type? How can you tell? How many fields are there in this instruction type? What are the names of these fields?
   c. According to the binary machine code, what is the value of each field in **hex**?
   d. Refer to the **MIPS sheet**, what operation is this instruction? How can you tell? What is the mapping of the registers being used in this instruction?
   e. What is the final MIPS instruction? Is it the same as the Source Column in the Text Segment window?
8. Now, let us take a look at ***line #17*** of `proc1.s`.

a. What format is this instruction?
b. What are the values of **opcode**, `rs`, and `rt` of this instruction in **hex**?
c. What is the name of the target label if it takes the branch? What is the address of this label in **hex**? (Hint: you can find it in the Text Segment window.)
d. So, do we put this address as the value of the ***immediate field*** of the instruction? Why?
e. How do we find the value of the ***immediate field***? What is this value?
f. What is the machine code of this instruction in **binary** and **hex** formats? Does your answer match the Code column in the Text Segment window?

9. Finally, let us convert the j instruction in ***line #20***.
    a. What format is this instruction? How many fields are there in this format?
    b. What is the opcode of this instruction in **hex**?
    c. To what ***label*** and ***address*** does this instruction jump?
    d. How many bits can you use in the address field of the instruction? How can we "squeeze" the address into this field? What are the reasons behind this approach? What is the value of the address field in **binary**?
    e. What is the machine code of this instruction in **binary** and **hex**? Is it the same as what's in the Code Column of the Text Segment window?

Your TA will "invite" one of you randomly after the activity to share what you have discussed.

Individual Assignment 1: Conversion in `proc2.s`

Convert the following line in `proc2.s` to machine code and then back to MIPS instructions (note your answers in a text file named `individualAssign1.txt`):

- Line #7
- Line #14
- Line #17
- Line #20

You must show all the steps including values of the instruction fields when you demo this lab.

Verify your answers with the Text Segment window.

Collaboration

You must credit anyone you worked with in any of the following three different ways:

1. Given help to
2. Gotten help from
3. Collaborated with and worked together

What to hand in

When you are done with this lab assignment, submit all your work through CatCourses.

Before you submit, make sure you have done the following:

- Attached your `individualAssign1.txt` and `tpsAnswers.txt`.
- Filled in your collaborator's name (if any) in the "Comments…" textbox at the submission page.

Also, remember to demonstrate your code to the TA or instructor before the end of the grace period.