CS204 - Computer Architecture
Practice Lab - 3

Date: 3rd Feb 2025. Not Graded!

Deadline: 10th Feb 2025. Complete all the tasks at the earliest!

Working with RISC - V Control Instructions and Procedures using Venus

We will continue working with RISC-V ISA using *Venus* simulator. Over the last two weeks you have been introduced to many RISC - V instructions. A few points:

1. Use the RISC-V Reference Data sheet for any quick look up and class slides to know the role of each instruction.

2. In this lab, the onus will be on usage of control instructions. Below tasks are not exhaustive but are only indicative for your practice.

3. As you start exploring with control instructions, having a visual of PC of each instruction will be very handy. So, start observing PC in *Venus* https://venus.cs61c.org/ (Dont worry about Cache right now, we'll look at it in-depth in Module 3 of the course).

4. Keep up the practice of arranging your code using the assembler directives. Also, continue to preload the data you want to work with.

Task 1: Realizing a 'switch' statement in RISC-V : Part I
Assume that there is a switch statement with three possible cases other than default. Write a RISC-V code to realize this switch.

Task 2: Realizing a loop in RISC-V.
Using the control instructions we have studied in class, write a code that will perform the following tasks on elements in an array. For simplicity, consider an array of 10 elements in range [-50, 50]. As usual, preload the values of the 10 elements into the memory and load one of the registers with the base address of this array.
1. Perform addition of all the elements.
2. Add those elements which are greater than a value, say, 5.
3. Consider only the positive elements which are below a value, say, 40.
4. Add only the negative elements.
5. Add all the elements which are greater than or equal to 35, considering them as unsigned.

Task 3: Write the RISC-V counterparts for a 'while', 'do-while' and 'for' loop instructions in C. What are the differences?

Task 4: Write RISC-V code to the count number of 1's in a word.

Task 5: Write a RISC-V procedure that calculates elements of array `b` as below:

```
b[i] = m * a[i] + c
```

You are required to pass only the values of `m`, `c`, starting addresses of array `a` and `b` as arguments (using registers) to this procedure.

Task 6: Task 1 of Lab 2 was to load the value `0xdeadbeef` at the memory location `0x1000000`. Most likely you would have used instructions like `li` and `la` to fill the registers with such 32bit values. Try to do the same task by avoiding `li` and `la` instructions as they

are pseudo instructions.
a) Think about using logical instructions for this (Task 4 in Lab 2 ?).
b) Use instructions `lui` and `auipc`.

Task 7: In Lab1.2, one of the tasks (Task5) was -
Use the shift and bit-wise operations to complete the task.
Write RISC-V code that extracts bits 16 down to 11 from register x5 and uses the value of this field to replace bits 31 down to 26 in register x6 without changing the other bits of registers x5 or x6. (Be sure to test your code using x5 = 0 and x6 = 0xffffffff. Doing so may reveal a common oversight.)

Let's modularize this task - read the bit positions "16","11","31", "26" from memory locations and perform the same task on values read into registers x5 and x6 from memory. Come up with a procedure for this.

Task 8: Once you are finished with *all* the above tasks, upload a single text file (name it "YourName_YourRollNo_learnings_lab3.txt") documenting your learnings and interesting aspects that you have noticed today. Write point-wise statements and not paragraphs. Upload the file by Friday, 10th Feb 2025, 11.55AM.