COEN 210 Project, Spring 2017

In this project, you are going to define your own 16-bit instruction set, along with the control signals for a pipeline implementation of this instruction set.

The context for defining your instructions and your control logic is that you must define something that executes two of the following functions:

A: accumulate all the values in an array
B: find the last occurrence of a specific value in an array
C: count the number of occurrences of a specific value in an array
D: create a new array with the reverse contents of the original array
E: add a specific value to each element in an array

In each case, you can start with the assumption that you have the starting address of the array (or both arrays in the case of D), the size of the array, and any constant value that might apply (the search value in B and C, and the value to add in E). You can decide which registers will contain these values.

A, B, and C all have an output value as well. You can decide which register will have this output value.

You will be assigned your two functions in class.

As part of defining your instruction set, you will define how many registers you need. All registers are 32 bits; 2's complement is assumed.

In your report, the followings are expected.

1. Definition of your instruction set
2. Assembly code for your assigned functions, using your own instructions.
3. Control logic of the MIPS datapath, as pictured on page 325 of the textbook. Note this diagram is missing some detail. You need to account for the missing details. *
4. Examples of a control hazard , a data hazard stall, and data forwarding

*One thing that does need to change from the datapath in the book: how the PC gets updated, given that the instructions are only 16 bits. So assume the calculation of the next PC does PC+2 rather than PC+4. And for branches, assume the sign-extended immediate value only gets shifted by 1 rather than by 2 before it is sent into the adder that computes the branch target.

Extra credit if you make changes to the datapath HW. You'll need to make it clear what the changes are.

The report is due on the last day of class.

Project Report Template

1. Instruction set architecture
   - 1.1 Instruction format(s)

     A description of the general encoding scheme for the instructions, including clarity on how many registers there are in your register file
   - 1.2 Instruction definitions

     For each instruction that you need to run your assigned functions, provide a brief description of the semantics of the instruction and the Op code for the instruction
2. Application
   - 2.1 First function
     - 2.1.1 Define input and output registers
     - 2.1.2 Assembly code
   - 2.2 Second function
     - 2.2.1 Define input and output registers
     - 2.2.2 Assembly code
3. Datapath control
   - 3.1 Any changes to datapath HW, for extra credit; the rest of this section should be specified in the context of these changes
   - 3.2 Instruction Decode
     - 3.2.1 Definition of Inputs
     - 3.2.2 Definition of Outputs
     - 3.2.3 Logic equations for outputs as a function of inputs
   - 3.3 Hazard Detection
     - 3.3.1 Definition of Inputs
     - 3.3.2 Definition of Outputs
     - 3.3.3 Logic equations for outputs as a function of inputs
   - 3.4 Forwarding
     - 3.4.1 Definition of Inputs
     - 3.4.2 Definition of Outputs
     - 3.4.3 Logic equations for outputs as a function of inputs
4. Hazards
   - 4.1 Control Hazard
     - 4.1.1 Show a snippet of code from section 2 that must flush instructions due to a control hazard
     - 4.1.2 Explain how your hazard detection logic causes the flush and how many cycles are flushed
   - 4.2 Data Hazard - Stall
     - 4.2.1 Show a snippet of code that must stall due to a data hazard
     - 4.2.2 Explain how your hazard detection logic causes the stall
   - 4.3 Data Hazard - Forwarding
     - 4.3.1 Show a snippet of code where data is forwarded
     - 4.3.2 Explain how your forwarding logic allows this to happen