ECEN 651: Microprogrammed Control of Digital Systems
Department of Electrical and Computer Engineering
Texas A&M University

Prof. Mi Lu
TA: Ye Wang

Pre-Laboratory Exercise #6

Pipelined MIPS Processor

Objective

The objective of this Pre-laboratory exercise is to acquaint you with the challenges you will face when implementing a pipelined MIPS processor. The questions provided in this document will help to further prepare you for the upcoming lab. Please have these questions answer prior to the next lab.

Background

The single-cycle MIPS processor built in the previous lab was a fairly simple but inefficient design. The long critical path through the entire data path made it difficult to achieve a reasonably high clock rate. Figure 1 shows the typical critical path for the single cycle MIPS processor when executing a load. Note that the portion of the data path which makes up the critical path depends on the logic delay associated with the various subcomponents.

To improve the design, we can break the instruction up into multiple stages and add pipeline registers between each of the stages. For MIPS, the traditional pipeline consists of five stages, which are as follows: Instruction fetch, Instruction Decode, Execute, Memory access, and Write Back. In the instruction fetch stage, the instruction address from the program counter is fed into the instruction memory and the corresponding instruction is read from memory. In instruction decode, the instruction is broken up such that part of it goes to the control unit and is decoded into control signals, while part of it feeds into address buses of

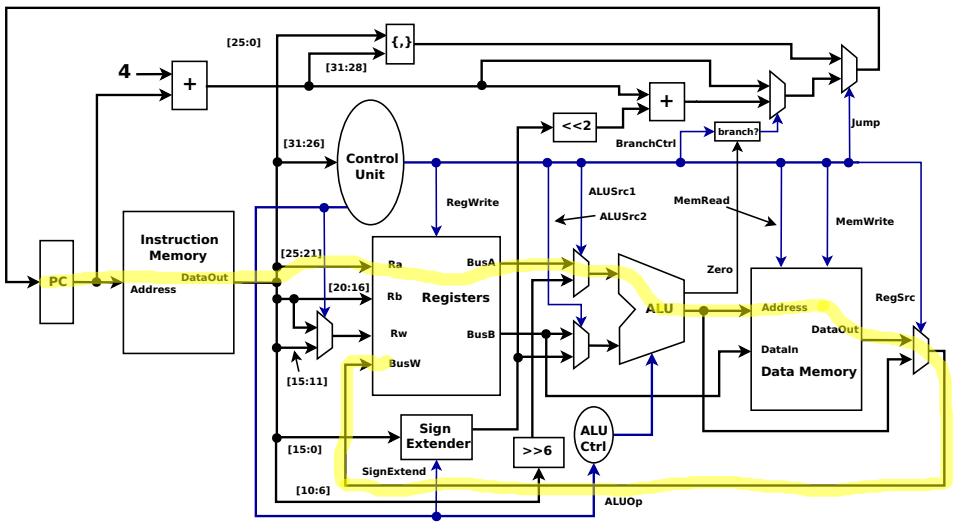

Figure 1: MIPS Single Path for Load Instruction

the register file. Thus, instruction decode includes the register read. The execute stage makes use of the ALU to compute a result for an R-type instruction or calculate an address for a load/store. The memory access stage is where the data memory is read or written to for a load or store respectively. For most R-type instructions, nothing is done in the memory stage. The write back stage is where data from the ALU or memory unit is stored into the register file for an R-type or load respectively. The MIPS pipeline is self-draining in the sense that control signals are generated in the early stages and propagate with the data. Likewise, all instructions must move through all five stages. Figure 2 provides a simplified view of the MIPS pipelined processor. The questions in this prelab will be based on this diagram

1 Prelab Questions

1. A data hazard exists in an inorder processor when an instruction reads from a register that is being written to by a previous instruction. This type of hazard is commonly referred to as a Read-After-Write (RAW) hazard. RAW data hazards must be handled properly in order to maintain correct code execution. One way to deal with data hazards is to stall the pipeline in the ID stage when the hazard is detected. For the data path shown in Figure 2, how many cycles would the pipeline need to be stalled when a RAW exists between two back to back R-type instructions? What about the case where an R-type instruction immediately follows a load? Likewise, can a branch instruction cause a RAW? What about a store instruction? Explain your answers for each of these!

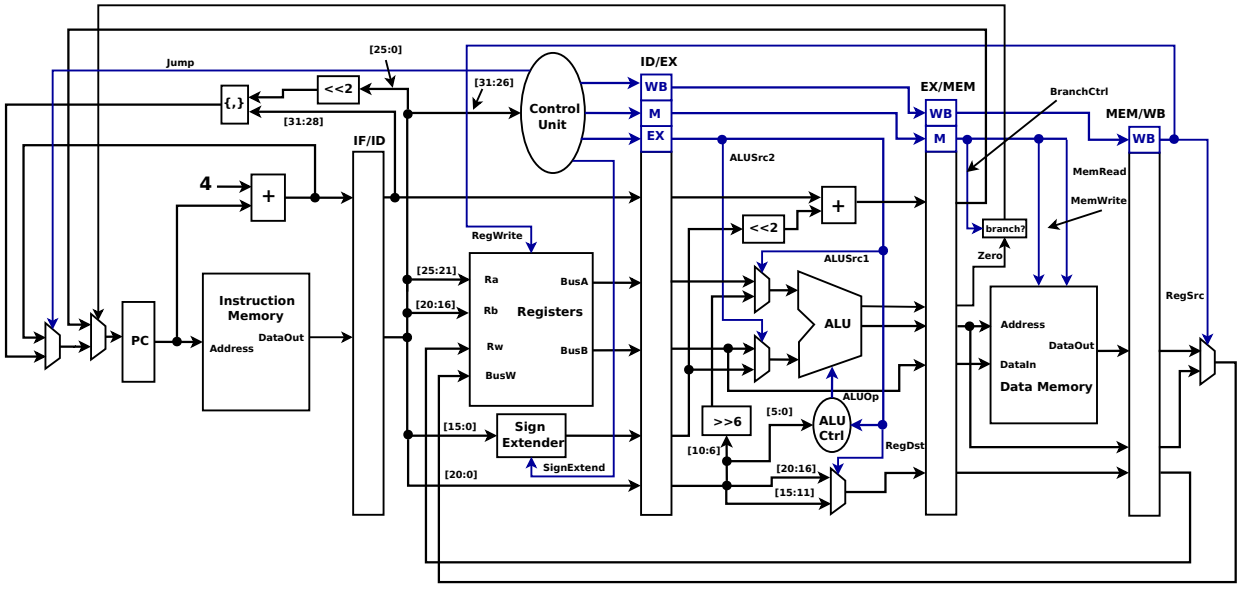

Figure 2: Simplified MIPS Pipeline

2. Simply stalling the pipeline to resolve data hazards increases the processors CPI (clock cycles per instruction), thereby reducing its overall efficiency. What improvement might you make to the design to reduce the number of stalls? For back to back R-type instructions, can we eliminate stalls all together? Why or why not? What about RAW data hazards involving load instructions?

3. Control hazards exist when the processor executes a jump or branch (i.e. changes the path of execution). Explain the reason for a control hazard involving a branch and provide a method for resolving this hazard. Do the same for jump.

4. Structural Hazards exist when two or more instructions need to use a resource in the data path at the same time. Explain how having separate memories for data and instructions avoids a structural hazard in the MIPS pipeline. Do any unresolved structural hazards exist?

5. What modification to Figure 2 would we have to make to handle the aforementioned hazards? Please provide some detail.