Laboratory 6

Objective

The objective of this laboratory is to implement data forwarding and hazard detection in hardware within your pipelined CPU. The result is a CPU that does not require the compiler to be aware of forwarding issues and load hazards. Turn in one model that incorporates the results of both Problem 1 and Problem 2.

Problem 1

Implement a data forwarding unit for your pipelined KURM processor. The data forwarding unit should detect when source data for the ALU is not available from the register file and forward the data from the appropriate pipeline stage to the ALU. Your forwarding unit should compare source register identifiers from the instruction in the register access stage with the destination register identifier from instructions further along in the pipeline. When equal, data should be appropriately forwarded to ALU inputs.

Implement your forwarding unit using behavioral VHDL. Integrate your forwarding unit into your KURM implementation from Problem 3 of Laboratory 5 by adding multiplexers controlled by the forwarding unit.

Problem 2

Implement a load-word hazard detection unit for your pipelined KURM processor. The hazard detection unit should determine when the processor should be stalled. When stalling is necessary, the program counter should be prevented from loading a new value, and the equivalent of a “no-op” instruction should be generated for the initial instruction instance.

Implement your hazard detection unit using behavioral VHDL. Integrate your hazard detection unit into your KURM implementation from Problem 1 by adding multiplexers and control signals as required.