LAB 4: A Pipelined VHDL Calculator

CPEG 324

Vinay Vazir
Jason Reynolds

ABSTRACT

The goal for this project was to develop a simple three stage, 8-bit, pipelined calculator combining the skills learned in the earlier labs into one project. This calculator being pipelined instead of single staged means that it is more reminiscent of what is used in practical applications.

DIVISION OF LABOR

The first step in completing this lab was to outline the datapath. Next, to develop any new components that had not already been developed for previous projects. Lastly, assembling the components into the datapath design and testing.

For the initial datapath design stage, Vinay and Jason worked together examining the data path created for the single stage calculator and determining what would need to change in order to create a pipelined datapath instead.

The second part of the project was to design components that were needed for the pipelined datapath but had not been created for previous assignments. Fortunately, the only new parts that were needed were register components for the interstage registers, and a new modified control system. The registers had in fact been created already for the single cycle calculator and gone unused. Vinay handled the bulk of the creation of the new control system with some input from Jason.

The last part of the project was putting all the components together and developing a testbench. Jason worked on putting the components developed in step two and from the previous single cycle calculator together and connecting them together with signals. Vinay and Jason worked together to troubleshoot issues as they arose.

STRATEGY

For the data path, we began with our single cycle datapath from lab three. Next we added two interstage registers, one between the register stage and the execution stage, and one between the execution stage and the write back stage (Denoted as ID/EXE and EXE/WB). Two more interstage registers were added into the control block that passed forward the opcode for these stages.

The output of the calculator was changed to accomodate the lab description. Three new outputs were added. One signal would indicate if a skip was occurring, and the other signal would display how much it was skipping by. The final signal was used to indicate a NOP instruction and was used for the very last opcode, as two NOPs were added to the end of every testbench to flush out the pipeline.

Functionally, we decided to avoid changing as much as possible in the execution stage of the pipeline, and we only added added a few mux's to handle data forwarding. Major changes happened to the control block as many of the old control signals had to be changed to accept input from the new internal interstage registers that held the old opcodes.

The testbench itself also changed, with the output of the ALU always displayed on a line marked ALU Output. During normal display instructions this line will not be printed. During a skip instruction where the skip condition is true, the test bench will display how many instructions will be skipped.

RESULTS

We were successfully able to develop a pipelined datapath and implement all the logic needed to control our 8-bit calculator.

CONCLUSION

We successfully created the pipelined datapath. We discovered that the most difficult part of the project was developing the control signals, as there were many aspects of the datapath that we did not account for at first, such as the fact that we needed to data forward the output of the ALU during display operations instead of just add and subtract operations.

This lab was fairly straightforward to do because of all the extra work we did in previous labs. The only part of the lab that became very difficult was the control block, as it required a lot of debugging to get through all the small issues that cropped up. All the rest of the functional blocks were already completed, including the code to generate the test benches.

This lab was a very fun and interesting lab to complete, as seeing it all come together on gtkwave was very nice. It was very satisfying to finally complete the lab and have a fully developed pipelined calculator.

NOTES ON EVALUATING

Use the command: *python builder.py bench.jv -s* to build the calculator and run the testbench. (The -s will tell python to launch gtkwave with bench.vcd after running the testbench and creating the vcd file)

APPENDICES

Appendix I:

Name	Hours spent
Jason Reynolds	20
Vinay Vazir	20

Appendix II:
All code located here:
https://github.com/Syncla/CPEG324/tree/master/lab4