CMPE 260 Laboratory Exercise 5

Memory & Writeback Stage

Andrei Tumbar
Performed: March 30th
Submitted: April 17th

Lab Section: 1
Instructor: Moskal
TA: Jacob Meyerson
Dennis Lam

Lecture Section: 1
Professor: Cliver

By submitting this report, you attest that you neither have given nor have received any assistance (including writing, collecting data, plotting figures, tables or graphs, or using previous student reports as a reference), and you further acknowledge that giving or receiving such assistance will result in a failing grade for this course.

Your Signature: _____

Abstract

In this laboratory exercise, the data memory, memory stage, and writeback stages of the MIPS datapath were implemented. The data memory was essentially equivalent to the instruction memory however it was written to be word addressable. This implementation initialized 1024 words in data memory. The writeback stage will simply select between the ALU output and the memory data output to write back to the register file.

Design Methodology

The given Verilog implementation of memory stage (bar the seven segment display, and switches) is very similar to the design of the instruction memory. The only major difference that can be seen is the addressablilty of data memory. Instead of concatenating four consecutive bytes as done with the instruction memory, the data memory will address by word meaning adjecent addresses will be 4-bytes apart.

The Verilog implementation of the writeback stage is fairly straightforward. There are a series of passthrough signals which a simply copied or inverted to the outputs. The VHDL implementation of the writeback will differ slightly because the output signal `reg_write` will be manually controlled by an input signal instead of by the inverted `mem_write` signal. Other than the passthrough signals, the writeback stage will control the writeback data output by selecting between the ALU result data and the memory read data to write back to the register file.

Results & Analysis

Memory Stage

To test the memory stage, a testbench was written to verify its functionality by sequentially writing and then reading from memory. An extra read cycle was added to verify that the first read cycle did not overwrite the memory at the target address.

Table 1: Memory Stage Testcases

Address	Operation	Value
0x1B	WRITE	0xAAAA5555
0x1C	WRITE	0x5555AAAA
0x1B	READ	0xAAAA5555
0x1C	READ	0x5555AAAA
0x1B	READ	0xAAAA5555
0x1C	READ	0x5555AAAA

As shown in Table 1, the first two operations will write to two difference memory addresses. The next four operations will verify the values of these addresses by performing read operations.
A set of waveforms was generated to show the simulation output.

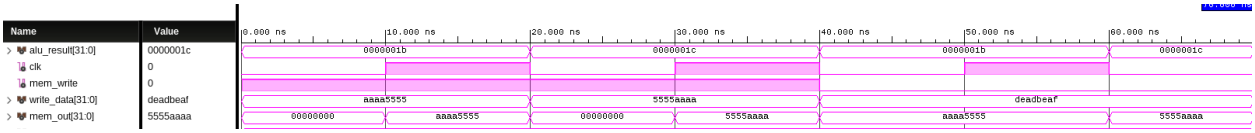

Figure 1: Memory Stage Behavioural Waveform

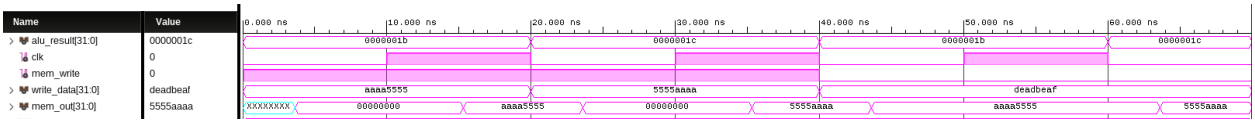

Figure 2: Memory Stage Post-Synthesis Timing Waveform

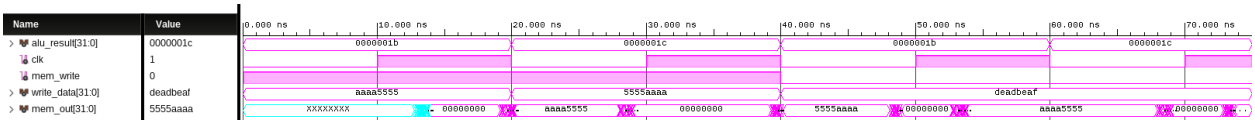

Figure 3: Memory Stage Post-Implementation Timing Waveform

Figures 1 2 and 3 show the testcases described in Table 1. Looking at the two timing simulations in Figures 2 and 3, we cna see that the shortened clock period of 20ns is still long enough to provide ample setup time for the writeback stage.

Writeback Stage

To test the writeback stage, the two different possible testcases need to be run. The writeback stage will select between the ALU result and the read memory output. A waveform was generated to show the proper functionality of the writeback stage.

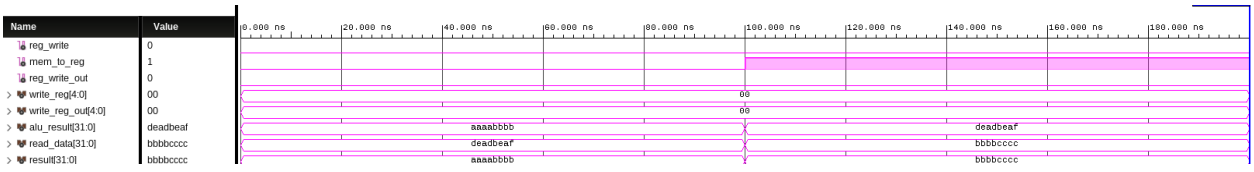

Figure 4: Writeback Stage Behavioural Waveform

Figure 4 will show that the mux selection for the writeback stage will properly select from the two input signals: `alu_result` and `mem_result`.

Conclusion

This laboratory exercise implemented the memory stage and the writeback stage. The memory stage wraps the data memory by passing the address from the ALU result into the data memory. No special handling is needed for any of the addresses in data memory. The writeback stage will simply select between the memory read output and the ALU result. The generated waveforms from the memory stage and the writeback stage show that the implementation works properly.

Demo results

Part1

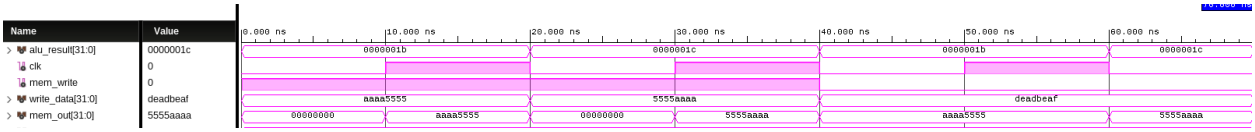

Figure 5: Behavioural simulation of Memory Stage

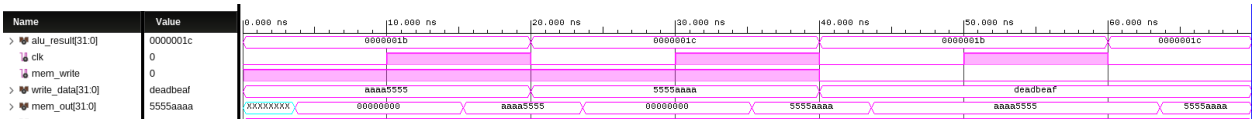

Figure 6: Post-synthesis timing simulation of Memory Stage

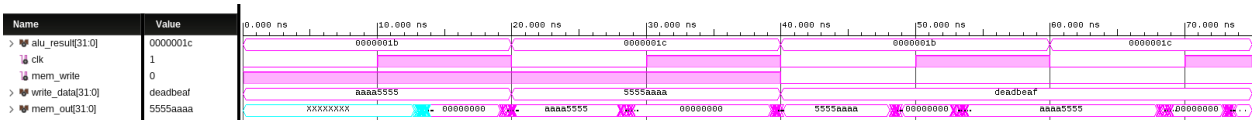

Figure 7: Post-implementation timing simulation of Memory Stage

Part2

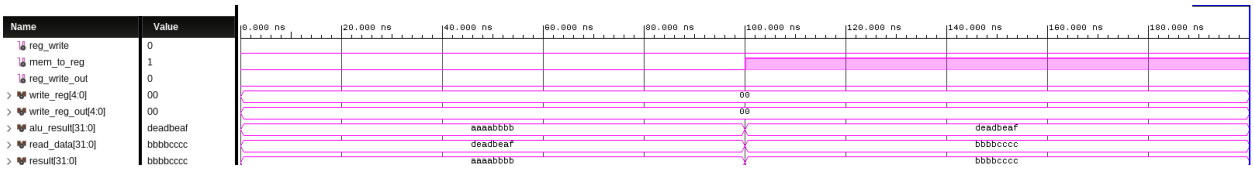

Figure 8: Behavioural simulation of Writeback Stage

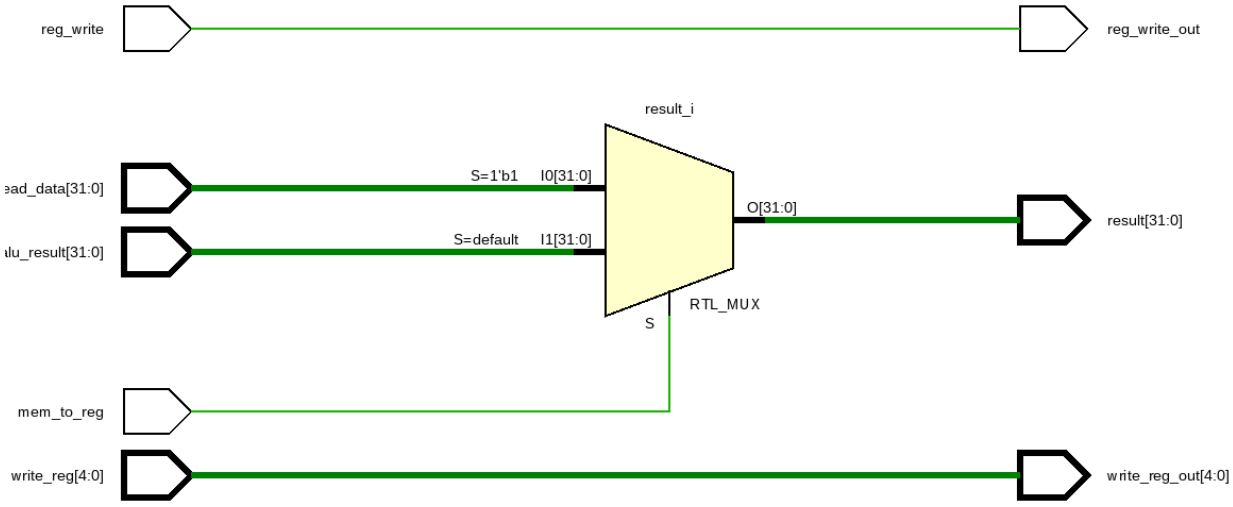

Figure 9: Writeback Stage RTL schematic