ECE 43700: Computer Design and Prototyping Lab

Midterm Report

Vaibhav Ramachandran

Asheem Chhetri

Lab Section: 3

GTA: Manik V. Singhal

Due Date: March 4th, 2018

1 Executive Overview

The Single Cycle and the Pipelined processor designs were compared based on their maximum clock frequencies, execution time, instructions executed per clock cycle, performance in millions of instructions executed per second, critical paths and in their utilization of the FPGA resources available. Surprisingly, the Single Cycle design did appear to have some advantages over the Pipelined processor, namely it's reduced latency per instruction and higher number of instructions executed per cycle. The Single Cycle processor was also a lot easier to implement since, at a time, only one instruction was being executed by the design, thereby eliminating the need for hazard detection. However, as expected, the benefits of employing a Pipelined design far outweighed the complexities involved in it's implementation. Since the Pipelined design allowed the simultaneous processing of multiple instructions, the execution time was significantly reduced thanks to the higher clock frequency that was attainable. The introduction of pipeline latches between each stage of the pipeline helped reduce the critical path delay of the design and also made better use of the FPGA's resources. Furthermore, thanks to the implementation of hazard detection and forwarding, the overall throughput of the Pipelined processor was also significantly improved.

The testing program that was used in the comparison of these two designs was the merge-sort program. It was ideal because it tested the two designs against nearly the entire MIPS ISA along with sequences of instructions that contained several dependencies and hazards. These dependencies and other corner cases encountered in the program made it ideal to test the performance of the hazard detection unit and the forwarding unit used in the Pipelined processor. Due to it's long execution time, it also provided a more definitive view of the differences in execution times and throughput between the two designs. In conclusion, it was determined that the Pipelined processor was the better design, thanks to it's higher throughput, reduced execution time, lower critical path and higher clock frequency.

2 Processor Design

Figure 1 corresponds to the data-path block diagram for the Single Cycle Design.

Figure 2 corresponds to the data-path block diagram for the Pipelined Design, with both forwarding and hazard units.

Figures are provided in the appendix section.

3 Results

Following are the results of our comparison between the Single-Cycle and Pipelined designs:

Memory Latency used to generate this table was: 5

ASM File used for the comparison: mergesort.asm

Criterion	Single Cycle	Pipeline
Number of Instructions	5399	5399
Execution Time(μs)	1428.730206	1346.746755
Number of Cycles	55168	87345
IPC(Instr. / cycles)	0.098	0.062
Max Frequency(85c model)(MHz)	38.56	64.66
Critical Path(ns)	32.96	22.418
Latency per instruction(μs)	0.129	0.387
MIPS	3.77888	4.00892
Registers (/114480)		
Total Logic Elements	3178	3594
Total Combinational Functions	2879	3211
Dedicated Logic Registers	1282	1836
Total Registers	1283	1837

Formulas used:

$$\texttt{Instructions Per Cycle} = \frac{\#\ \texttt{instructions}}{\#\ \texttt{cycles}} \tag{1}$$

$$\texttt{ExecutionTime} = \frac{1}{\texttt{fMax}} \ * \ \#\ \texttt{cycle} \tag{2}$$

$$\texttt{MIPS} = \texttt{IPC} \ * \ \texttt{fMax} \tag{3}$$

$$\texttt{Pipeline Latency} = \frac{1}{\texttt{fMax}} \ * \ (\texttt{MemoryLatency}) \ * \ 5 \tag{4}$$

$$\texttt{Single}-\texttt{cycle Latency} = \frac{1}{\texttt{fMax}} \ * \ (\texttt{MemoryLatency}) \ * \ 1 \tag{5}$$

4 Conclusions

From the above table it can be inferred that the Pipelined processor has a smaller execution time for the same set of instructions, thanks to the processing of multiple instructions simultaneously. It also has a higher operating frequency (clock speed), due to the inclusion of the pipeline latches that break up the process into several stages, thereby reducing the critical path of the design. The Pipelined processor also made use of a larger proportion of the FPGA's resources due to the introduction of the hazard and forwarding units. As a result of these improvements, the overall throughput of the Pipelined processor is also increased when compared to that of the Single Cycle design as can be seen by the increased MIPS value. Even though a 5-stage pipeline was implemented, the throughput and maximum frequency did not increase 5-fold because of the increase in the CPI due to stall hazards, flushes and incorrect branches. Nevertheless, it still showed a higher throughput compared to the Single Cycle design.

However, there were certain parameters in which the Single Cycle design had better results. These were namely, the IPC and latency per instruction. The increased latency comes from the fact that each instruction took at least 5 cycles to execute completely, but the clock frequency was not, at least, 5 times that of the Single Cycle design, where each instruction was executed in one clock cycle. The decreased IPC, despite the implementation of forwarding and hazard detection, was also a direct consequence of the fact that each instruction now took at least 5 cycles to implement. Nevertheless, it was still comparable to the value obtained from the Single Cycle design.

Overall, it can be said that the Pipelined processor's pros more than make up for it's few cons and thus justify it's employment over a Single Cycle design.

5 Contributions

Name	Source Files	Test Benches
Asheem Chhetri	Forwarding Unit	Hazard Unit
Vaibhav Ramachandran	Hazard Unit	Forwarding Unit

Asheem, designed the block diagrams for the pipeline processor in relation to Hazard and Forward unit. He also wrote Forwarding Unit, and test-bench for the hazard unit. Since the single cycle processor, that was used Lab 5 onwards was based on his design, he collected all relevant information required to populate the Results table for single cycle processor. He also helped debugging the process of combining the forward unit with the hazard unit, as was suggested by a TA, since it made more logical sense for our design.

Vaibhav, wrote the Hazard unit and also wrote the test-bench for the forwarding unit. Initially the test-bench was planned to be separated out for each unit. But after the Hazard unit test bench was created, it made more sense to combine the forwarding unit test bench in same file, as we combined our Forward and Hazard unit. Thus, Vaibhav combined the two test-benches. He also collected the required information necessary to populate the Results table for the Pipeline processor. He also helped debugging the merging process of our both units. Furthermore, he also designed and implemented the 4-entry branch target buffer and the 2-bit saturating predictor.

6 Appendix

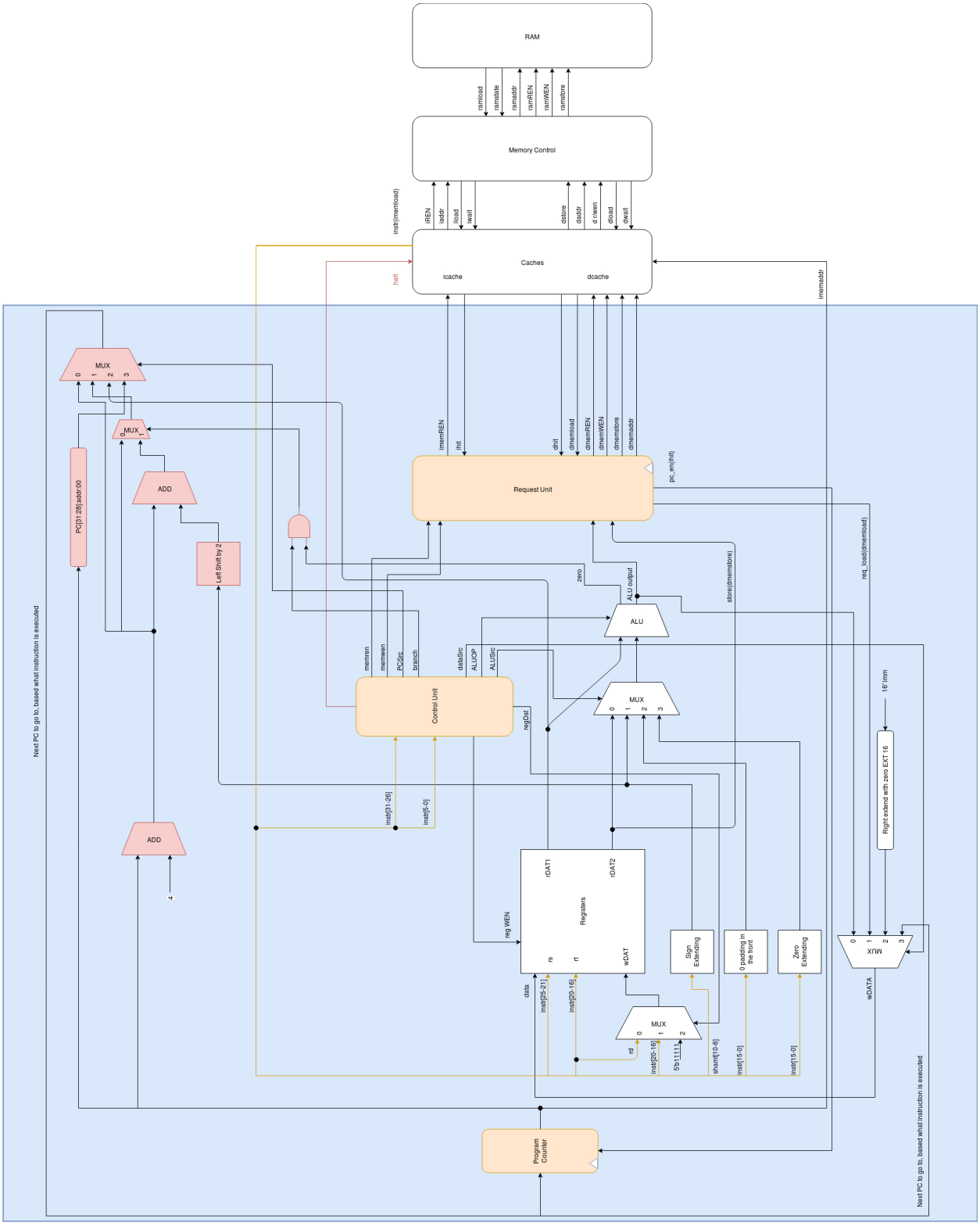

Figure 1: Single Cycle Block Diagram

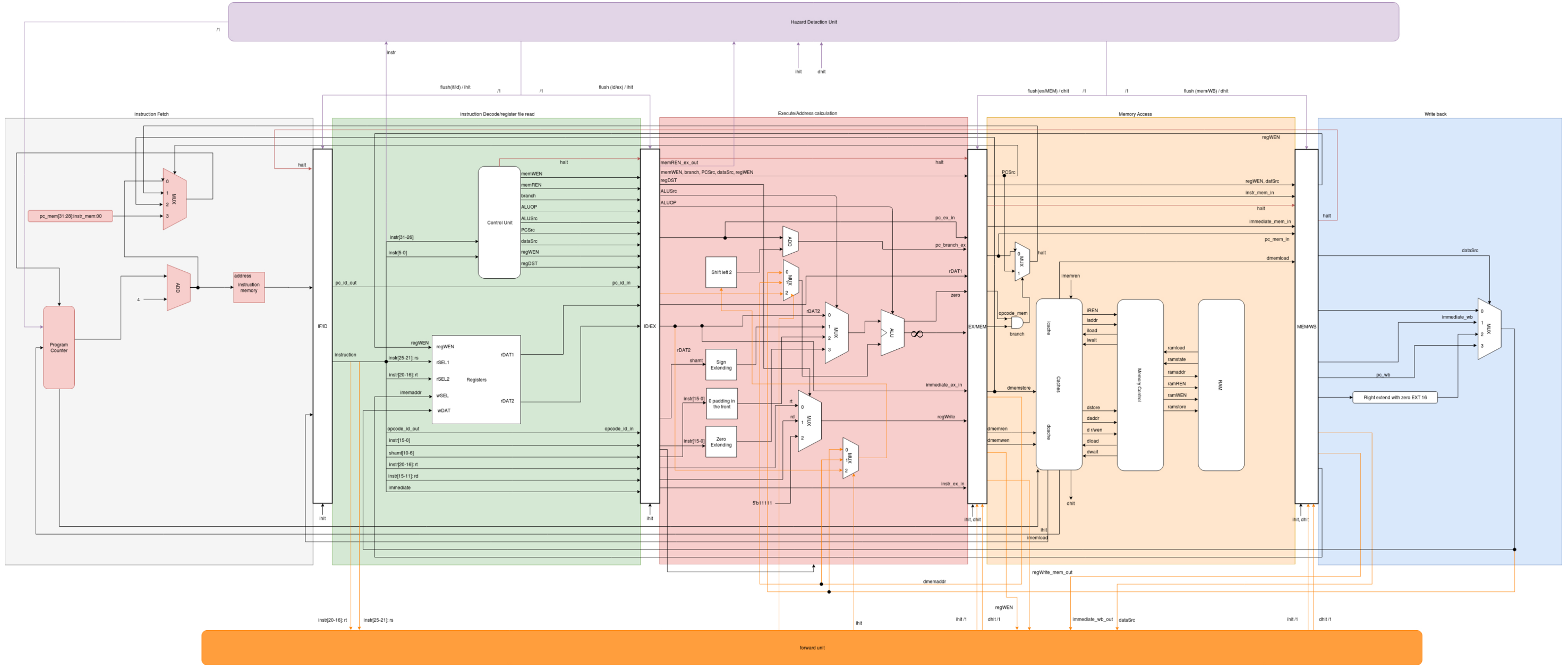

Figure 2: Pipeline with Hazard and Forward unit Block Diagram