# Midterm Report

ECE 43700

Jordan Huffaker & Lucas Krull

Nick Pfister

10/14/16

## 1. Overview

The power of computation is becoming increasingly relevant in people's lives. The speed at which hardware can complete a set of commands in the MIPS instruction set is a foundational limitation to the speed at which an end user can achieve their desired goal. The traditional method, completing each instruction in a single clock cycle, is limited by the critical path of the longest instruction. We introduce a pipelined processor design that overcomes this barrier by splitting instructions into multiple stages and by completing different stages of multiple instructions at the same time.

We compared the effectiveness of a pipelined approach over the traditional method by comparing several metrics such as average IPC, MIPS, instruction latency, number of logic gates, and number of flip-flops. These metrics were gathered while completing a common task programs must complete for end users to use a computer: merge sorting an array. We found that the pipelined approach was more effective in its MIPS performance, however, it came at the cost of average IPC, instruction latency, number of logic gates, and number of flip-flops.

## 2. Processor Design

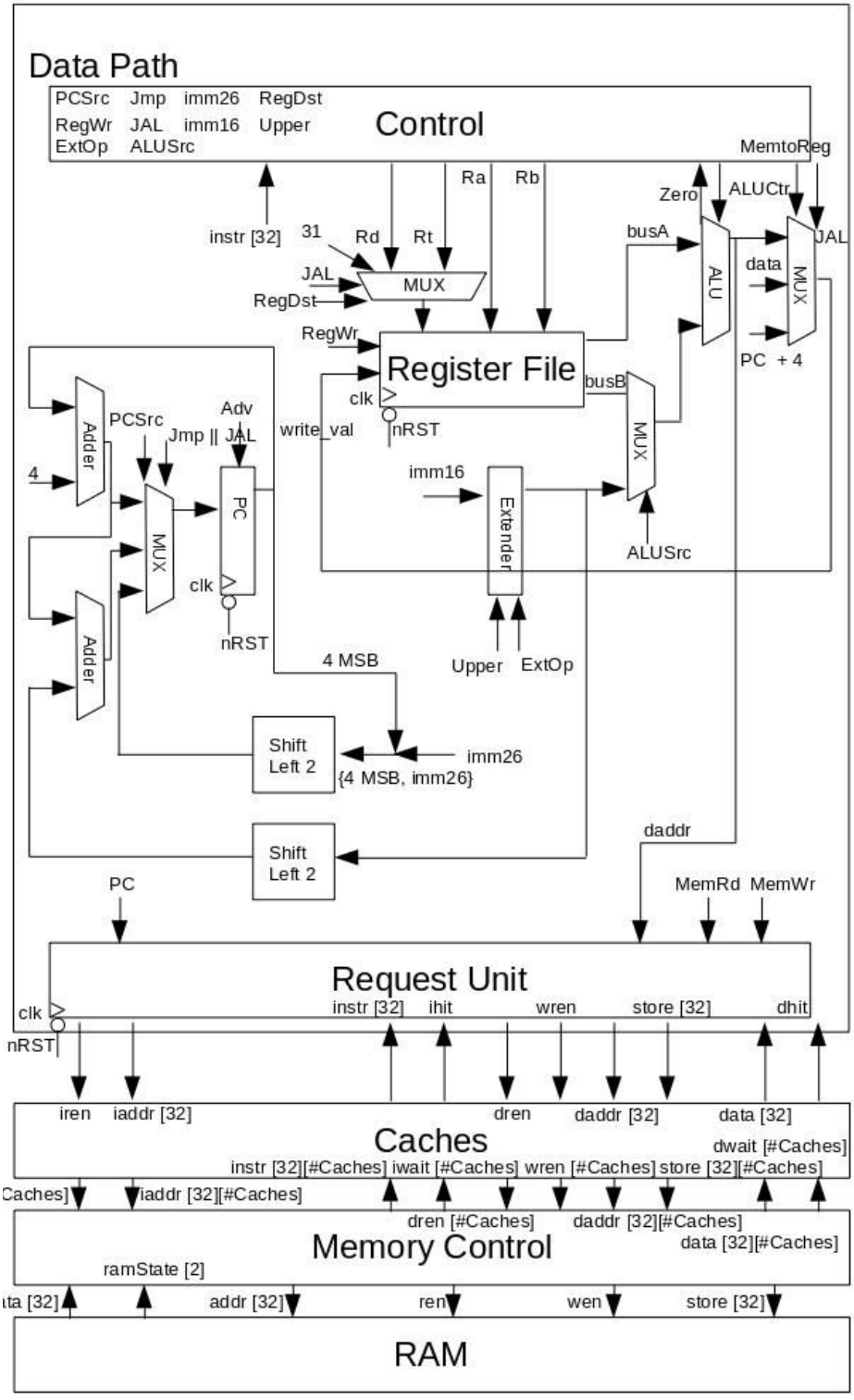


**Figure 1: Single Cycle Block Diagram**

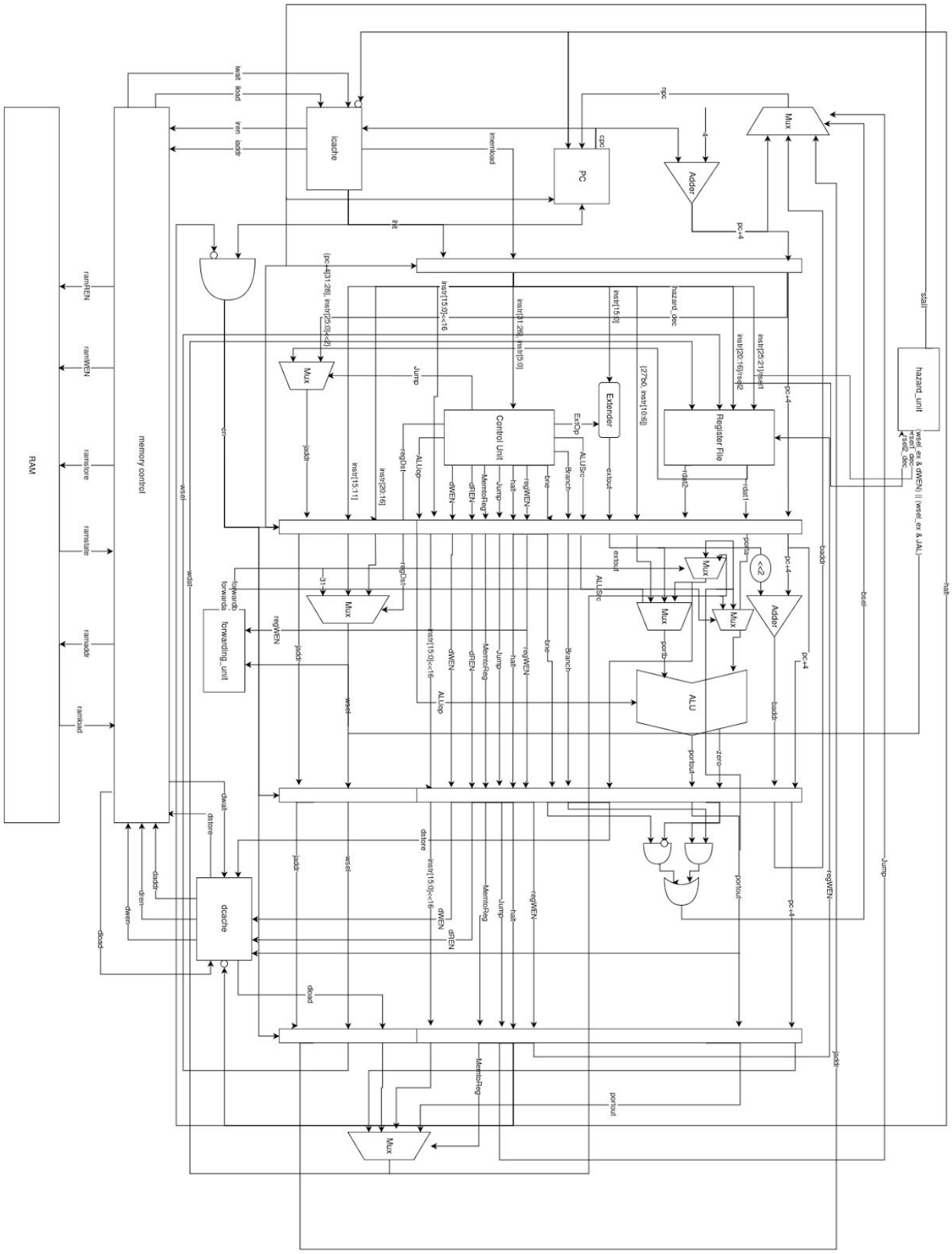


**Figure 2: Pipelined Block Diagram**

## 3. Results

| Metric | SingleCycle | Pipelined |
|---|---|---|
| Max Frequency (MHz) | 63.42 | 133.28 |
| Max Test Bench Freq. (MHz) | 50 | 500 |
| Average IPC | 0.39 | 0.31 |
| *MIPS Performance | 24.73 | 41.32 |
| *Latency of One Instr. (ns) | 15.77 | 37.5 |
| # Logic Gates Required | 3,153 | 3,497 |
| # Flip-Flops Required | 1,311 | 1,750 |

*uses the “Max Frequency” parameter in its calculation

- Max Frequency was calculated with the formula $min(CLK,\ CPUCLK * 2)$ with CLK and CPUCLK obtained from the system.log files
- Average IPC was calculated with the formula $\frac{Total\ Instruction}{Total\ Cycles}$ with Total Instructions obtained from running “sim -t” and Total Cycles obtained from the report generated after running the synthesized simulation
- Latency of One Instr. was calculated with the formula $\#\ Stages\ *\ Period$ with Period calculated from Max Frequency in table
- MIPS was calculated with the formula $\frac{Average\ IPC * Max\ Frequency}{1{,}000{,}000}$ with Average IPC and Max Frequency from table

## 4. Conclusions

As was expected, our pipeline design achieved a significant performance improvement over our single cycle design. Pipelining allowed our design to more than double its maximum clock frequency. Additionally, specifically for mergesort.asm, it achieved an increase in maximum clock frequency of 10x. This improvement in clock speed resulted in an improvement in MIPS performance of nearly 2x for our pipeline design over our single cycle design.

However, some metrics did worsen due to pipelining, though these were expected. The average IPC and latency of a single instruction decreased from our single cycle design to our pipeline design. These are affected by stalling and the number of stages, which for our pipeline design was 5 and for single cycle was 1. Although, the average IPC decreased due to the necessity for stalling, the overall increase in clock frequency more than compensates for this and results in an overall increase in performance. Additionally, our pipeline design requires more FPGA resources compared to our single cycle design. This is due to the necessity for pipeline registers and the forwarding and hazard units.

## 5. Contributions

Jordan Huffaker

- Updated block diagram
- Coded pipeline registers
- Coded hazard unit and testbench
- Cooperated on debugging
- Wrote overview, inserted block diagrams, and aided in result compilation for report

Lucas Krull

- Created/updated block diagram
- Coded datapath
- Coded forwarding unit
- Cooperated on debugging
- Wrote conclusion and aided in result compilation for report