Pipelined vs. Singlecycle Processor Design

ECE 437

Grant Gumina and Warren Getlin

Pranav

10/17/2014

Overview

In this report, two major processor designs are being compared. The first is a simply singlecycle processor and the second is that same singlecycle design, but with pipelines added to it. Specifically, we're comparing the latency and throughput of each processor as well as how many FPGA resources were required to actually synthesize each design. Also found in this report will be an architectural comparison of each design.

The above metrics were gathered by synthesizing each Verilog implemented design onto an FPGA development board. Results were recorded and sanity checked (using our knowledge of what the theoretical values for each metric should be) to make sure our designs weren't implemented wrong. What we found was that each design was properly implemented (ie as close to a singlecycle/pipelined processor as reasonably possible) and that our pipeline design performed significantly better than our singlecycle design.

Processor Design

Below is a RTL diagram for our pipeline processor implementation (Figure 2). Below that, in Figure 1, is a RTL diagram for our singlecycle processor implementation.

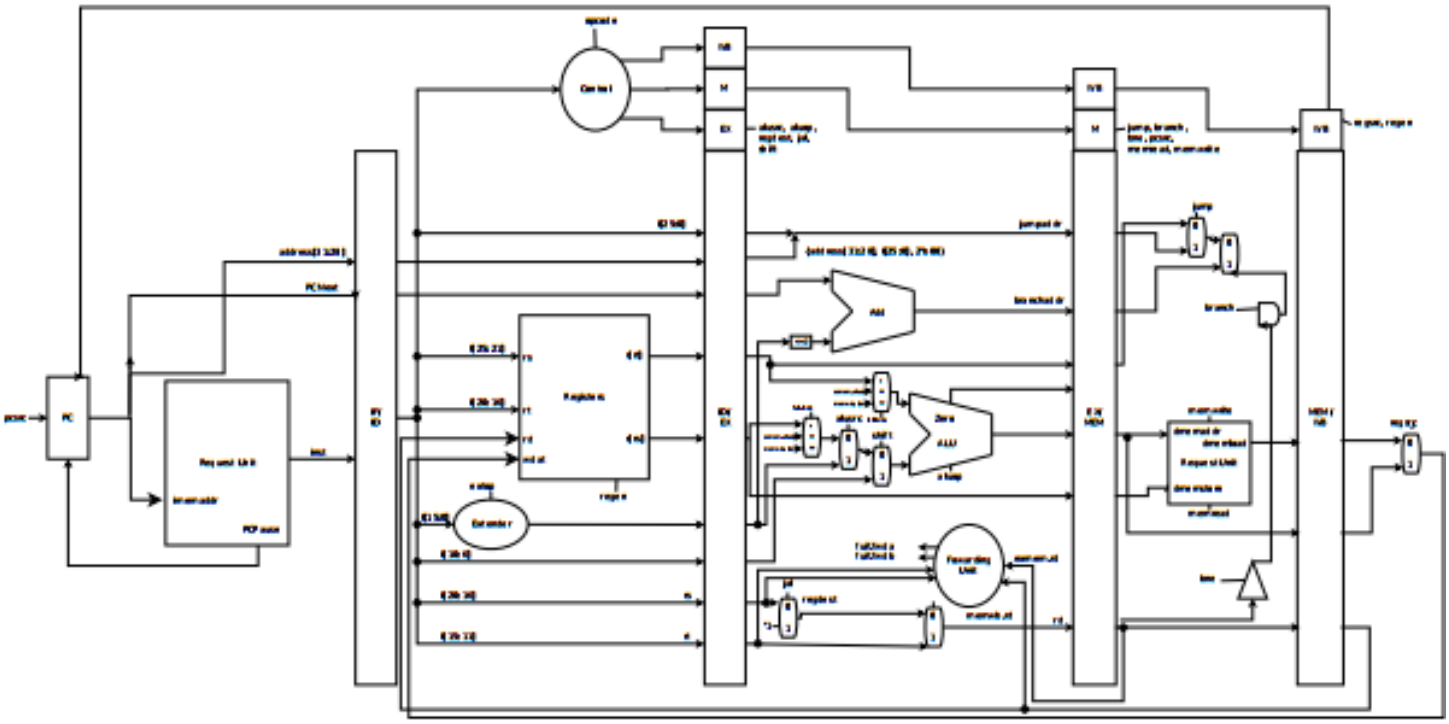

Figure 1 RTL Diagram for Pipeline Processor

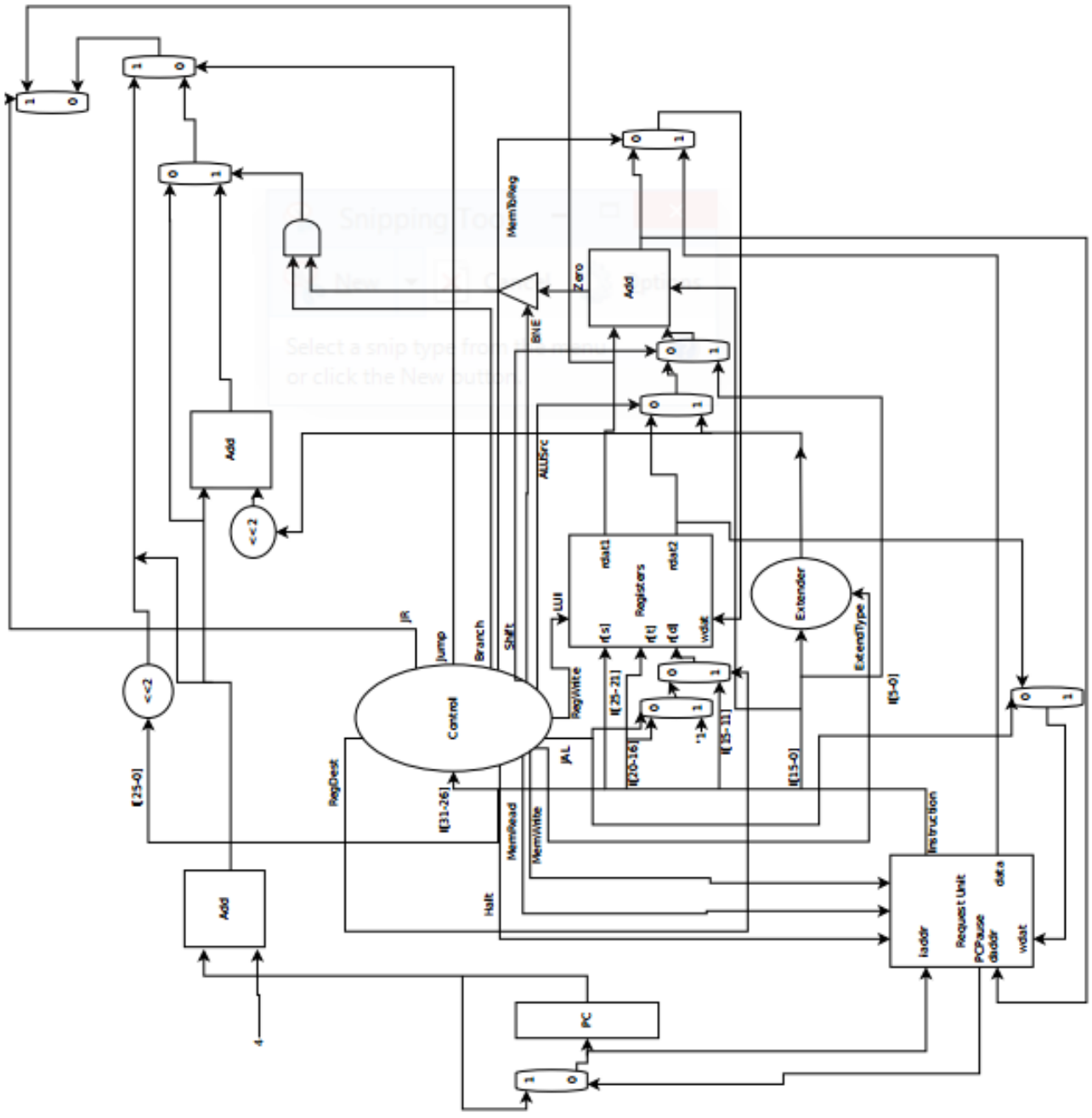

Figure 2 RTL Diagram for Singlecycle Processor

Results

	Target Frequency (MHz)	Max Frequency (MHz)	Max Period	Run Time	Throughput (MIPS)	Latency (sec)
SS	50	50	20	429000	12.58	1.00E-07
	75	50.5	19	487000	10.9	9.50E-08
	100	Not working	N/A	N/A	N/A	N/A
PL	50	58.1	17	437184	16.4	8.60E-09
	75	63.22	15	546500	16.4	7.90E-09
	100	67.57	14	546500	16.4	7.40E-09

Conclusions

As seen from the data, and as derived theoretically in class, the pipeline processor performs significantly better than the singlecycle one. Throughput is consistently higher with our pipeline processor implementation, as expected. Also, latency is reduced by factors of ten and one hundred compared to the singlecycle processor being run at 50 and 100 MHz, respectively. While it's unfortunate that we weren't able to get our singlecycle processor to work at 100 MHz, the trends the data shows are correct. Lastly, it's worth noting that the periods for the pipeline processor decrease continuously as the target frequency for our simulation increases.

All results shown above show expected behavior for a pipeline processor. In general, pipeline processors increase throughput and decrease latency when compared to a simple singlecycle processor we built during the first part of class.