Final Report

ECE 437
Computer Architecture

TA: Chuan Tan

Dec 9 2016

Adit Ghosh

Overview

We are comparing the performance of our different processor designs. We compare the pipelined processor with and without a cache hierarchy and the pipelined processor with cache and a multicore processor. We will be comparing the estimated synthesis frequency, average instructions/clk, latency of instruction, FPGA resources required and Speedup from sequential to parallel program. We simulated the design using latencies above 5. I am using my own testfile for testing. In this testfile, I add 100 integers sequentially. For testing the dual core I divide it up into two sets of 50 instructions each for each core. This file allows us to test various test various corner cases.

I expect the synthesis frequency to be fastest for the multicore and the caches and then the pipeline. Caches allow the processor to fetch instructions much faster. I expect the instructions per clock to be much greater for caches than the pipeline. This is because caches reduce the need to read from memory, thus minimizing the waste of clock cycles. Latency of each instruction would be greater for caches and greatest for Multicore as each instruction has to pass through extra logic. FPGA resources will also be greater for caches and much greater for Muticore (expecting >> X2). This is because of the extra hardware in caches and twice the wiring of caches in multicore. We expect Multicore to have the highest MIPS because it has twice the throughput of caches. Caches will have a higher MIPS than pipeline because caches reduces memory access time. We expect the speedup of the multicore to be the greatest, followed by caches, followed by pipeline as the average execution time is least for Multicore followed by caches followed by pipeline. This report further contains block and state transition diagrams for the various processor designs, Results, Conclusions and Contributions.

Design

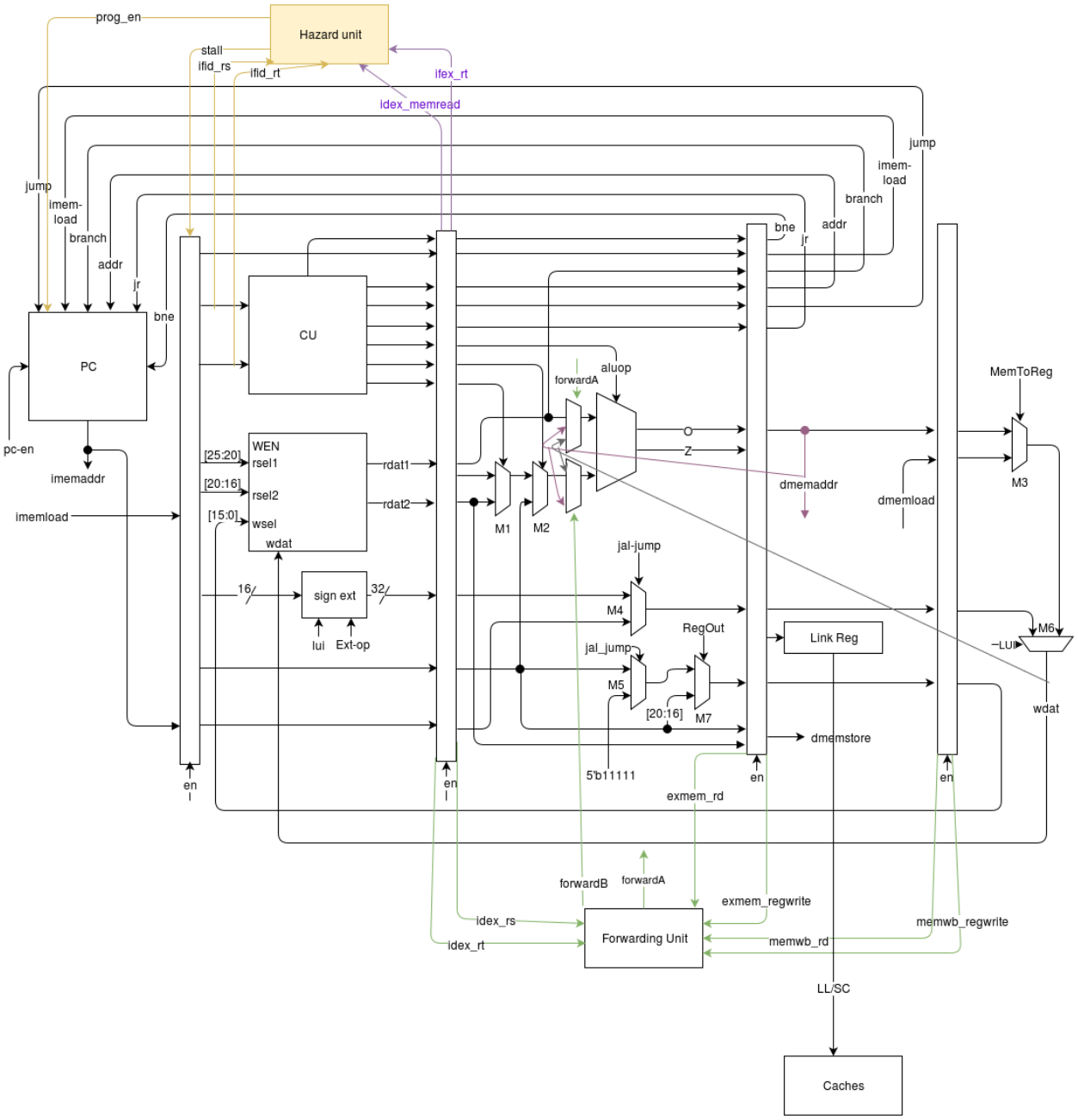

Fig. Pipeline processor design

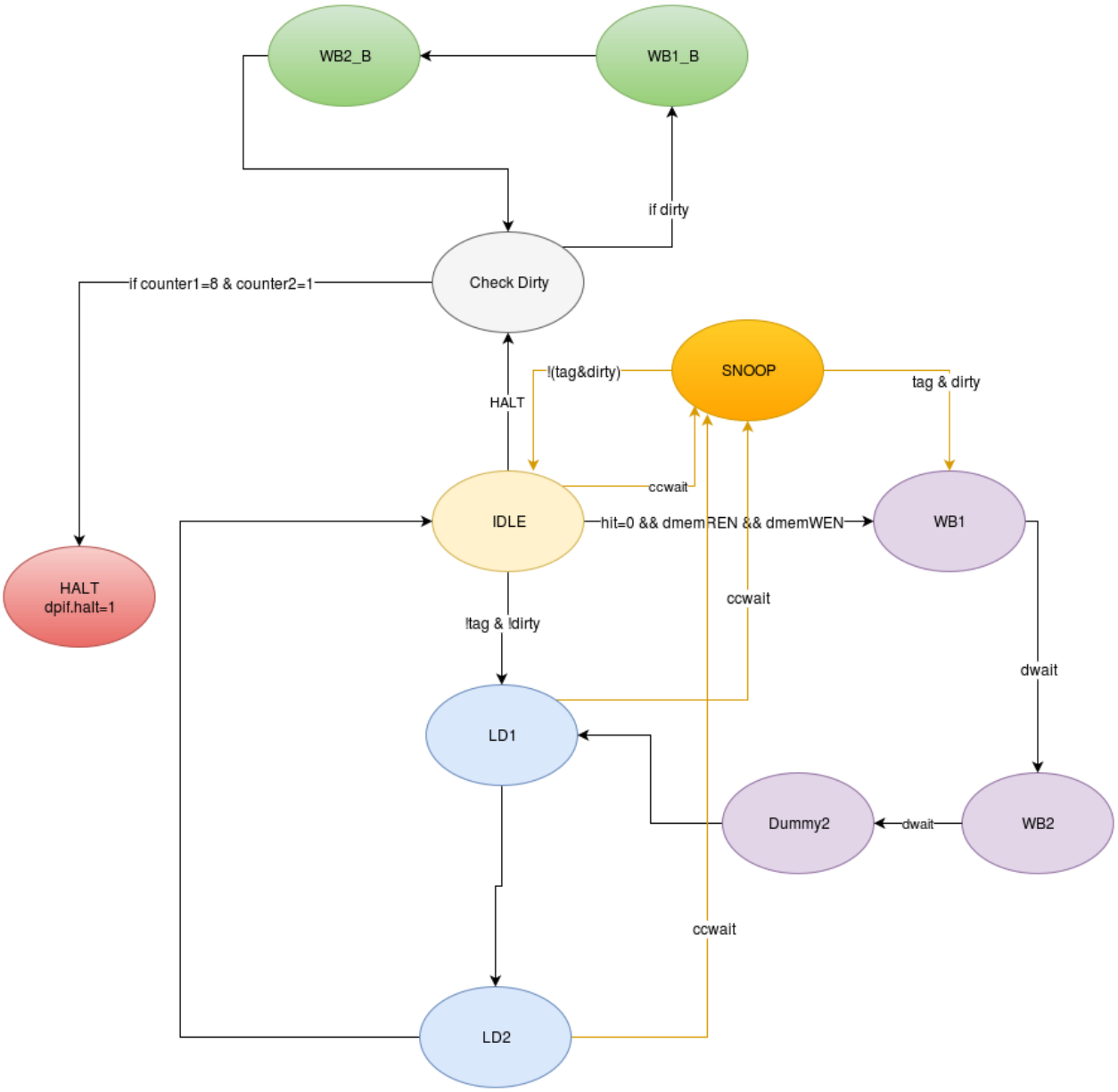

Fig Dcache state diagram

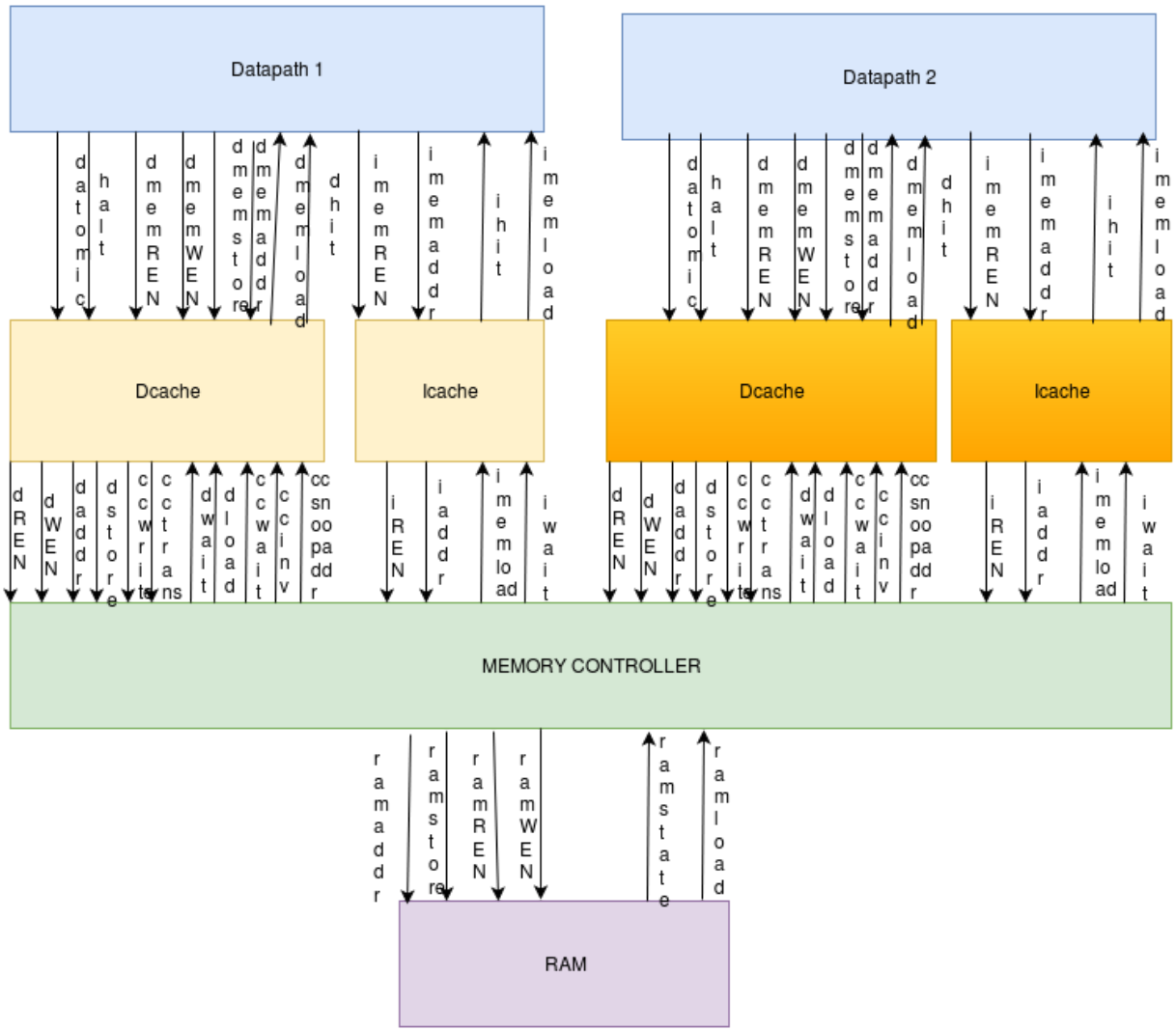

Fig Dual Core Block diagram

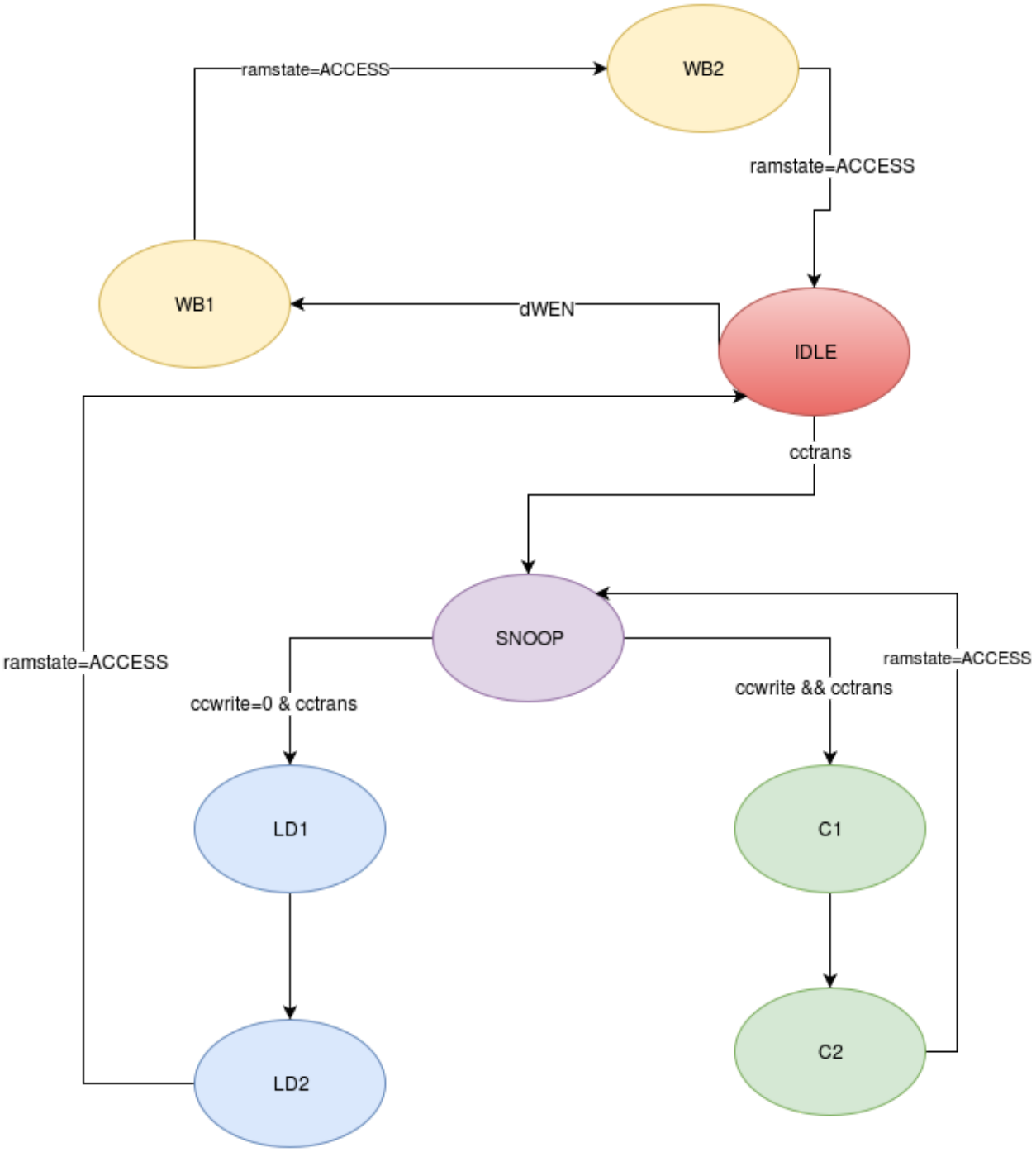

Fig Coherence controller

Results

	Max Clk Freq MHz	Instruction/ clk	Latency	FPGA resources	MIPS	Speedup
Pipeline	46	0.61	0.120	3225	23.5	1
Pipeline with Caches	53	0.75	0.45	4215	42.3	2.4
Multicore	60	0.84	0.65	8100	55	2.95

Conclusion

We can conclude from the above results that we achieve greatest throughput when using Multicore. Caches reduce the latency of store and load instructions making data available much faster as the processor does not have to go to memory every time.

We observed a Max Clock frequency of our multicore and caches to be 80 MHz which is faster than our pipeline at 47 MHz. Multicore has a faster clock as predicted. Latency is greater for Multicore, 0.65, and caches, 0.45, because instructions and data has to go through more logic than pipeline, 0.12, as our caches and multicore is built on our pipeline design. For the FPGA resources we observed that the multicore layout has approximately twice(8100) the resources as the caches(4215). This is to incorporate Cache coherence and the additional core. Caches has significantly more resources than Pipeline(3225) to incorporate Dcache and Icache. The Multicore has a MIPS of about 55 whereas pipeline has about 23.5 this is a result of higher throughput of a multicore design. Lastly we achieve a speedup of around 2.95 using our dual core design.

Contribution

This project was completed individually for all steps.