ECE 4750 Computer Architecture

Topic 6: Advanced Processors – Superscalar Execution

http://www.csl.cornell.edu/courses/ece4750
School of Electrical and Computer Engineering
Cornell University

revision: 2024-11-09-17-52

List of Problems

Problem 1. In-Order Superscalar Processors

Consider the "abstract pipelines" for three processor microarchitectures shown in Figure 1. Figure 1(a) illustrates the canonical five-stage in-order single-issue TinyRV1 processor. Figure 1(b) illustrates the canonical in-order dual-issue TinyRV1 processor discussed recently in lecture. Figure 1(c) illustrates a new in-order quad-issue TinyRV1 processor. The quad-issue processor has four functional units (or "pipes"). Figure 2 illustrates which instructions can use which functional unit in the dual-issue TinyRV1 processor. Figure 3 illustrates which instructions can use which functional unit in the quad-issue TinyRV1 processor. Notice that the quad-issue TinyRV1 processor has much higher computational throughput; it can execute up to two `mul` instructions and two memory instructions per cycle. Recall that our in-order processors must always use aligned fetch blocks, and we do not let fetch-blocks "slip" in the F and D stages.

Consider the following assembly sequence. Assume that there are no instruction cache nor data cache misses, and that `x10` is initially a very large number (i.e., there are many iterations of the loop).

```
1  loop:
2    0x1000 lw   x1, 0(x2)
3    0x1004 lw   x3, 0(x4)
4    0x1008 mul  x1, x1, x6
5    0x100c mul  x3, x3, x7
6    0x1010 add  x8, x1, x3
7    0x1014 add  x9, x9, x8
8    0x1018 addi x2, x2, 4
9    0x101c addi x4, x4, 4
10   0x1020 addi x10, x10, -1
11   0x1024 bne  x10, x0, loop
```

Part 1.A Pipeline Diagram for Single-Issue TinyRV1 Processor

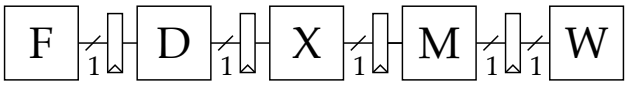
Draw a pipeline diagram to illustrate the how the given assembly sequence executes on the canonical single-issue TinyRV1 processor. Calculate the CPI and IPC for many iterations of this loop.

Part 1.B Pipeline Diagram for Dual-Issue TinyRV1 Processor

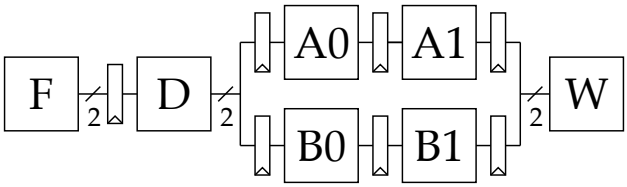
Draw a pipeline diagram to illustrate the how the given assembly sequence executes on the canonical dual-issue TinyRV1 processor. Calculate the CPI and IPC for many iterations of this loop.

Part 1.C Optimized Pipeline Diagram for Dual-Issue TinyRV1 Processor

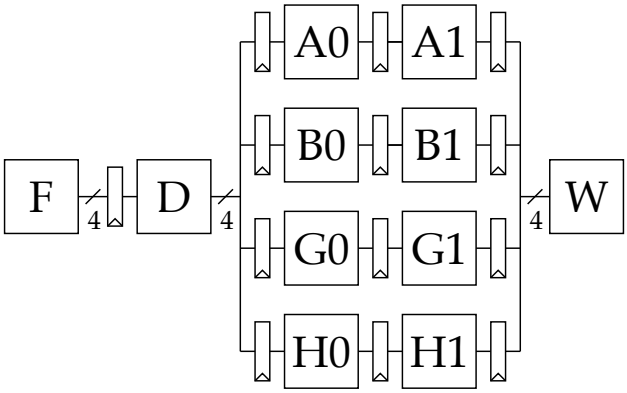
Re-schedule the given assembly code to optimize the performance when executing on the dual-issue TinyRV1 processor. Draw a pipeline diagram to illustrate the how the given assembly sequence executes on the canonical dual-issue TinyRV1 processor. Calculate the CPI and IPC for the optimized version of many iterations of this loop.



(a) In-Order Single-Issue TinyRV1 Processor



(b) In-Order Dual-Issue TinyRV1 Processor



(c) In-Order Quad-Issue TinyRV1 Processor

Figure 1: Abstract Pipelines for Three In-Order Processors

	`add`	`addi`	`mul`	`lw`	`sw`	`jal`	`jr`	`bne`
A-Pipe	✓	✓	✓			✓	✓	✓
B-Pipe	✓	✓		✓	✓	✓	✓	

Figure 2: Dual-Issue Mapping of Instructions to Functional Units

	`add`	`addi`	`mul`	`lw`	`sw`	`jal`	`jr`	`bne`
A-Pipe	✓	✓	✓			✓	✓	✓
B-Pipe	✓	✓		✓	✓	✓	✓	
G-Pipe	✓	✓	✓			✓	✓	
H-Pipe	✓	✓		✓	✓	✓	✓	

Figure 3: Quad-Issue Mapping of Instructions to Functional Units

Part 1.D Optimized Pipeline Diagram for Quad-Issue TinyRV1 Processor

Re-schedule the given assembly code to optimize the performance when executing on the quad-issue TinyRV1 processor. Draw a pipeline diagram to illustrate the how the given assembly sequence executes on the quad-issue TinyRV1 processor. Calculate the CPI and IPC for the optimized version of many iterations of this loop.

Part 1.E Instruction Level Parallelism

When we say an instruction sequence has high *instruction level parallelism* (ILP) we mean that it should be possible to execute many instructions in parallel. We can measure the ideal ILP by constructing a instruction dependency graph where each node in the graph is an instruction, and each edge in the graph is a RAW dependency. The ideal ILP is then simply the total number of nodes divided by the longest path through the graph. The ideal ILP roughly corresponds to the average number of instructions per cycle when executing an instruction sequence on an ideal superscalar processor with perfect software scheduling, perfect branch prediction, and an infinite number of single-cycle functional units.

Draw an instruction dependency graph for a single iteration of the loop. There should be 10 nodes in the graph (one per instruction). What is the ideal ILP for a single iteration?

Draw an instruction dependency graph for a three iterations of the loop. There should be 30 nodes in the graph (one per instruction). What is the ideal ILP for three iterations? Note that you should not draw any edges for control dependencies; every edge should correspond to a RAW dependency.

Derive a simple equation for the ideal ILP for N iterations of the loop. Discuss why the IPC achieved on the quad-issue TinyRV1 processor is less than the ideal ILP.