ECE 4750 PSET 4

Tim Yao (ty252)

Nov 25, 2015

1 Tree Network Topologies

1.a Baseline I3L Microarchitecture

Cycle:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
mul r1, r2, r3	F	D	I	Y0	Y1	Y2	Y3	W																					
mul r4, r1, r5		F	D	I	I	I	I	Y0	Y1	Y2	Y3	W																	
div r6, r7, r8			F	D	D	D	D	I	Z	Z	Z	Z	W																
div r9, r10, r11				F	F	F	F	D	I	I	I	I	Z	Z	Z	Z	W												
div r12, r13, r14								F	D	D	D	D	I	I	I	I	Z	Z	Z	Z	W								
mul r15, r12, r16									F	F	F	F	D	D	D	D	I	I	I	I	Y0	Y1	Y2	Y3	W				
mul r17, r15, r18													F	F	F	F	D	D	D	D	I	I	I	I	Y0	Y1	Y2	Y3	W

Figure 1: Pipeline Diagram for Baseline I3L Architecture

The total issue to commit cycle count is 27.

1.b Schedule Oldest Ready Instruction First on IO2L Microarchitecture

Cycle:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
mul r1, r2, r3	I	Y0	Y1	Y2	Y3	W	C																	
mul r4, r1, r5					I	Y0	Y1	Y2	Y3	W	C													
div r6, r7, r8		I	Z	Z	Z	Z	W	r	r	r	r	C												
div r9, r10, r11						I	Z	Z	Z	Z	W	r	C											
div r12, r13, r14										I	Z	Z	Z	Z	W	C								
mul r15, r12, r16														I	Y0	Y1	Y2	Y3	W	C				
mul r17, r15, r18																		I	Y0	Y1	Y2	Y3	W	C

Figure 2: Pipeline Diagram for IO2L Architecture

The total issue to commit cycle count is 24.

1.c Optimal Scheduling on IO2L Microarchitecture

Cycle:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
mul r1, r2, r3		I	Y0	Y1	Y2	Y3	W	C										
mul r4, r1, r5						I	Y0	Y1	Y2	Y3	W	C						
div r6, r7, r8					I	Z	Z	Z	Z	W	r	r	C					
div r9, r10, r11									I	Z	Z	Z	Z	W	C			
div r12, r13, r14	I	Z	Z	Z	Z	W	r	r	r	r	r	r	r	r	r	C		
mul r15, r12, r16							I	Y0	Y1	Y2	Y3	W	r	r	r	r	C	
mul r17, r15, r18											I	Y0	Y1	Y2	Y3	W	r	C

Figure 3: Pipeline Diagram for IO2L Architecture with Optimized Scheduling

The total issue to commit cycle count is 18.

1.d Scheduling Comparison

The optimized scheduler is able to achieve higher performance due to better exploitation of ILP. In the instruction sequence, there are two main sequences of data dependencies, so the optimal schedule will be to interleave these two sequences, along with the two independent div instructions, to maximize commits per cycle. Also because we have an un-pipelined divider, we will also want to start the divide instructions as early as possible (which is what the optimal schedule does). We might be able to implement this in hardware by using a mixed priority scheduling algorithm. The scheduler will have to place divide instructions at the highest priority (with the oldest div instruction issuing first). The next highest priority will be the oldest ready instruction that is not a divide. This will implicitly interleave instructions with data dependencies.