CS152 Exam #2
Fall 2003

Professor Dave Patterson

Question 1: Potpourri (Jack and Dave's Question)

Part A:

TLBs entries have valid bits and dirty bits. Data caches have them also. Which of the following are true? Circle the correct answer(s).

A. The valid bit means the same in both: if valid = 0, it must miss in both TLBs and Caches.
B. The valid bit has different meanings. For caches, it means this entry is valid if the address requested matches the tag. For TLBs, it determines whether there is a page fault (valid=0) or not (valid=1).
C. The dirty bit means the same in both: the data in this block in the TLB or Cache has been changed.
D. The dirty bit has different meanings. For caches, it means the data block has been changed. For TLBs, it means that the page corresponding to this TLB entry has been changed.

Explain (briefly):

Part B:

Buses and networks share some common characteristic yet retain some differences. Which of the following are true? Circle the correct answer(s).

A. Multimaster buses need to resolve arbitration before using the bus, while networks don't.
B. Both buses and networks transfer multiple words to increase communication bandwidth.
C. Networks are often connected in a hierarchy while buses are not connected in such a way.
D. Buses usually connect computers together while networks usually connect I/O peripherals to processors.

Explain (briefly):

Question 1: Potpourri (Jack and Dave's Question) Continued…

Part C:

What would be the bottleneck if we tried to turn an ordinary single issue tomasulo machine into a dual issue machine by changing only the issue (and Icache) unit to issue 2 instructions at once?

Part D:

Briefly explain how the Tomasulo algorithm resolves the following classes of hazards:

RAW:

RAR:

WAW:

WAR:

Question 1: Potpourri (Jack and Dave's Question) Continued…

Part E:

Again, assume we have a dual-issue Tomasulo machine with exactly three reservation stations for arithmetic instructions. The reservation stations/functional units along with their execution latencies are as follows:

adds and subs	-- a cycles
multiply	-- 2a cycles
divide	-- 5a cycles

(Loads, stores, and branches are handled separately.)

What is the minimum number of entries in each of the reservation stations and in the ROB necessary to guarantee that we wont stall on a structural hazard while trying to issue an arithmetic instruction?

Add/sub reservation station: _____

Multiply reservation station: _____

Divide reservation station: _____

Number of entries in the Reorder Buffer: _____

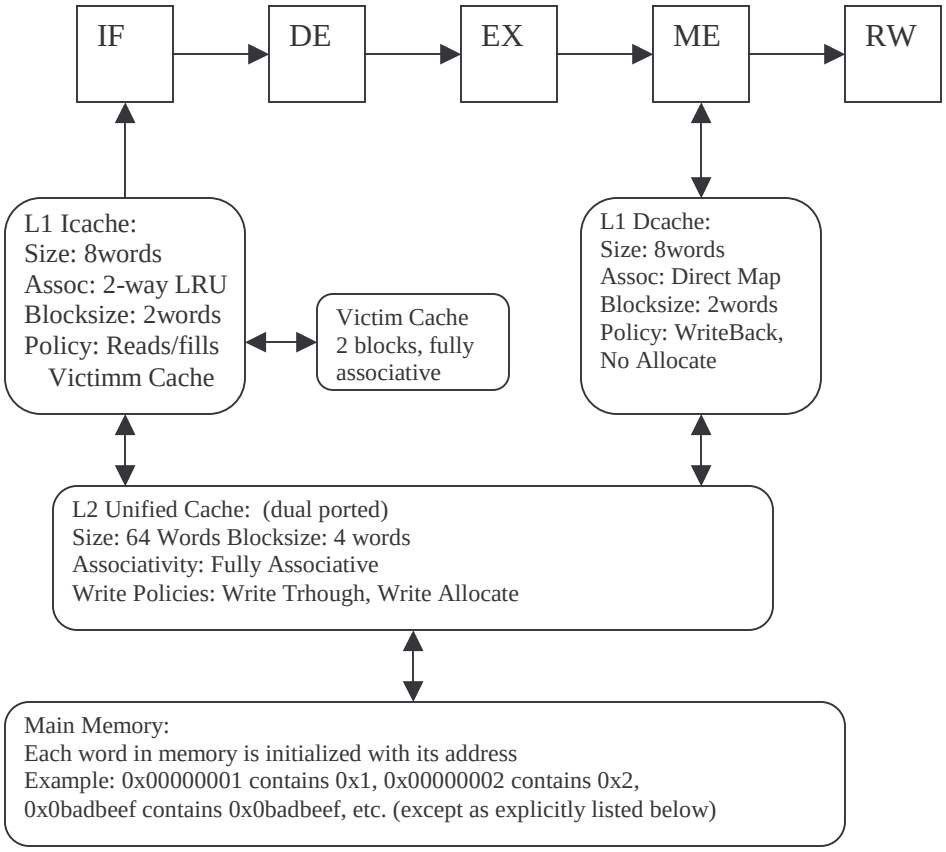
Part F:

The x86 instruction set only has 8 ISA-defined general purpose registers. Assume that each instruction only writes to one register, but may read from up to 3 registers (those darn CISC instructions!) If the maximum number of instructions that can be in flight at any given time is 32, how many physical registers must we have in order to implement explicit register renaming?

Number of physical registers: _____

Question 2: Cache This! (Kurt's Question)

Kurt's rinky-dink computer has the following organization:



His computer has 32-bit words and addresses and no virtual memory system.

The worst-case latency of the entire memory system is 1 cycle. (I.e., the cycle time is long enough such that L1Dcache, L1Icache, and L2 can all miss on the same request, fill their blocks, and L1I and L1D can return the requested data all in the same cycle. This assumption is totally unrealistic and defeats the purpose of having a cache, but it will make your calculations easier.)

Assume further that the L2 cache is dual-ported but services Icache requests before Dcache requests.

Question 2: Cache This! (Kurt's Question) Continued…

Part A:

Please show the structure of each of the 4 caches in a table format. We've done the L1 Dcache for you; your answers should contain the same types of information as ours. Be sure to include the size of all fields in the cache.

L1 Icache:

L1 Icache Victim Cache:

L1 Dcache:

Index #	Tag	Word0	Word1
00			
01			
10			
11			

Tag will be 27 bits for each block.
Index will be 2 bits from address.
Block offset will be 1 bit.
Byte offset is 2 bits.

L2 Unified:

Question 2: Cache This! (Kurt's Question) Continued…

Part B:

Assume that Kurt just turned on his computer (with memory initialized as described above except for the addresses below) and then started executing these instructions. For each instruction, consider each of the 4 caches. For each cache, indicate whether the instruction hits in that cache ("H"), misses in that cache ("M"), or is never checked ("X"). For cache hits and misses, also include the cycle number on which the access occurred. Some boxes may have more than one entry. We have filled in a couple entries for you.

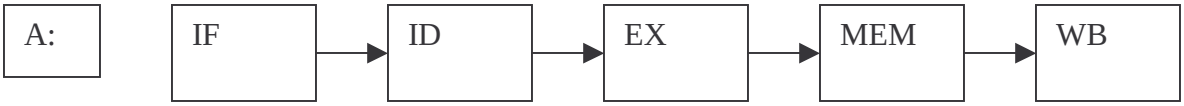
Don't forget that this is a pipelined processor!

Address	Instruction	L1 Icache	L1 Victim	L1 Dcache	L2
0x00000000	Lw $1 0xBAD0($0)	M-1	M-1	M-4	M-1
0x00000004	Lw $2 0xBAD4($0)				
0x00000008	Lw $3 0xBA04($0)				
0x0000000C	Sw $1 0($0)				
0x00000010	Sw $2 0x0C($0)				
0x00000014	Sw $3 0x80($0)				
0x00000018	Sw $4 0x0C($0)				

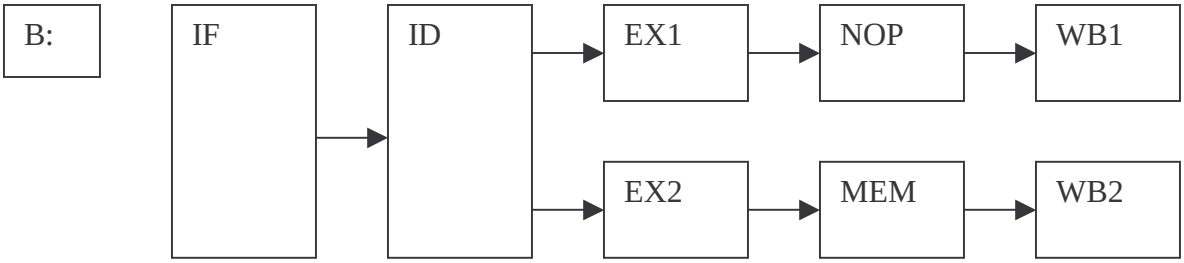
Question 3: Superscalar (John's Question)

You are an engineer working at Advanced Intelligent Devices. Your workers have proposed 3 different MIPS 2000 processor designs to you.

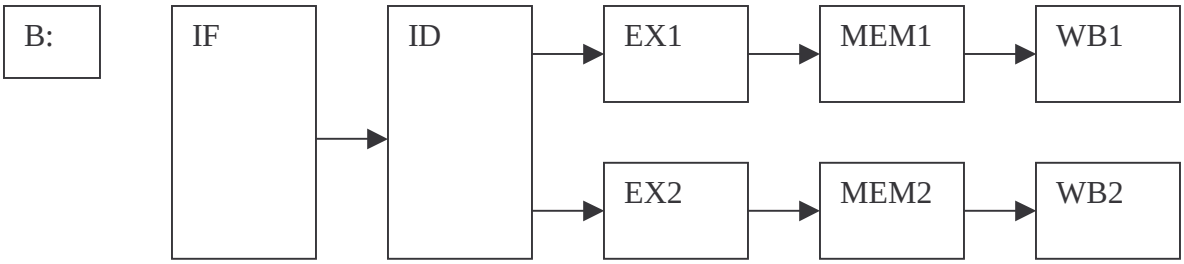
Processor A is a 5-stage pipeline identical to the one described in your Fall 2003 CS152 class. It has a full five stages and writes to the register file can be read during the same cycle:



Processor B is a limited superscalar processor that can handle branches and jumps only in the first pipeline and memory operations only in the second pipeline. Integer operations can be executed in both pipelines as long as they are not dependent. Instructions are only issued if they are not dependent. The first pipeline always executes earlier instructions and the second pipeline always executes later instructions:



Processor C is a full superscalar processor that has no restrictions on placement of branches, jumps, or memory operations. The only restriction is that like the limited superscalar pipeline, earlier instructions are always in the first pipeline:



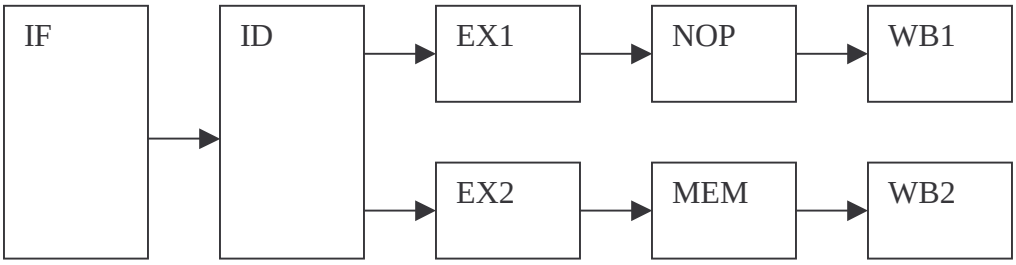
Question 3: Superscalar (John's Question) Continued…

In the 5-stage pipeline, when instruction 0x40000000c is completing WB for the first time, the pipeline looks like this.

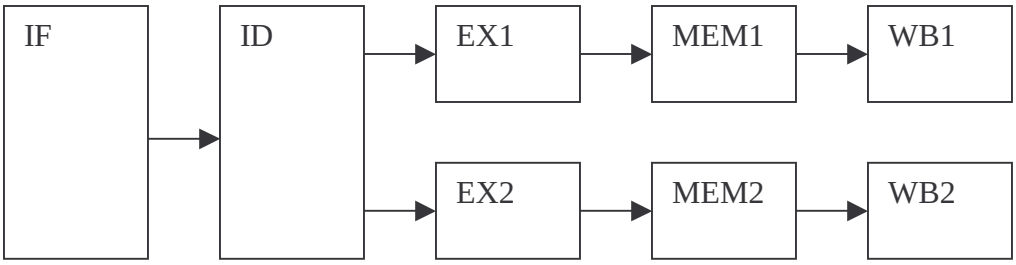
Now fill in the stages for each of the pipelines as instruction 0x40000034 is completing WB for the second time. (If you like, you can use just the first two fields of the instruction.)

Processor A:

Processor B:



Processor C:



Question 3: Superscalar (John's Question) Continued…

Part B:

If you were asked to reorder the instructions to improve performance for either processor B or processor C, which would be easier and why?

Part C:

Now reorder the instructions for the processor that you thought was easier. Indicate where the stalls will occur (if there are any left). You may not change the code size.

Address	Label	Instruction
0x40000000		
0x40000004		
0x40000008		
0x4000000c		
0x40000010		
0x40000014		
0x40000018		
0x4000001c		
0x40000020		
0x40000024		
0x40000028		
0x4000002c		
0x40000030		
0x40000034		
0x40000038		
0x4000003c		
0x40000040		
0x40000044		
0x40000048		