



**INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR**  
**Class Test 2 - 2023**

Date: 11-04-23      Timing: 2:00 pm - 3:00 pm      Place: NR223      Total Marks = 30

Subject No : CS60003      HIGH PERFORMANCE COMPUTER ARCHITECTURE

Department/Centre/School : Computer Science and Engineering

Answer all questions. In case of reasonable doubt, make assumptions and state them upfront.

Marks will be deducted for claims without proper reasoning. Write your answers (with all analysis, justification, calculation etc) within the allotted time.

---

1. Assume we have a MIPS machine that uses the speculative version of Tomasulo's algorithm. It can issue up to two instructions per clock cycle. Make the following assumptions about the machine:

- FP addition executes in 3 cycles
- FP multiplication executes in 5 cycles
- Integer operations and effective address addition execute in 1 cycle
- Data can be accessed in the data cache in 1 cycle
- The FPUs are fully pipelined and there are 2 FP adders (FA1,FA2), 1 FP multiplier (FM1)
- There are two ALUs (A1,A2), which handle effective address calculation along with all other integer operations.
- There are 8 entries in the reorder buffer
- The Issue and Commit columns of the table can handle up to two instructions (each) per cycle
- Assume all the functional units are fully pipelined the Common Data Bus (CDB) can handle up to two instructions per clock cycle
- At most one instruction may be accessing the data cache in each clock cycle
- The remaining FUs all have two reservation stations (RS), named: FMUL[1,2], FAD[1,2], ALU[1,2]
- Similarly two entries for store and load buffers, namely: ST[1,2], LD[1,2]

Fill in the following table indicating the cycles in which each instruction performs each step. Note that this table has been augmented to allow you to track the functional unit, reservation station and ROB entry being used, which should ease your job of finding hazards. Notice that the FU/EX entry indicates what functional unit is performing computation for the instruction, and what cycles it begins on and ends on.

Instruction	Issue	FU/EX	MEM	CDB	Commit	ROB/RS
L.D F1, 0(R1)	1	A1 / 2-2	3	4	5	#1 / LD1
L.D F2, 0(R2)						
ADD.D F2, F1, F2						
MUL.D F4, F1, F2						
ADD.D F5, F2, F3						
S.D F5, 0(R3)						

[10]

2. Prof. Shonku has built a system around a processor with in-order execution that runs at 1.1 GHz and has a CPI of 0.7 excluding memory accesses. The only instructions that read or write from memory are loads (20% of all instructions) and stores (5% of all instructions). The memory system for this computer is composed of a split L1 cache that imposes no penalty on hits. Both the I-cache and D-cache are direct mapped and hold 32 KB each. The I-cache has a 2% miss rate and 32-byte blocks, and the D-cache is write-through with a 5% miss rate and 16-byte blocks. There is a write buffer on the D-cache that

eliminates stalls for 95% of all writes. The 512 KB write-back, unified L2 cache has 64-byte blocks and an access time of 15 ns. It is connected to the L1 cache by a 128-bit data bus that runs at 266 MHz and can transfer one 128-bit word per bus cycle. Of all memory references sent to the L2 cache in this system, 80% are satisfied without going to the main memory. Also, 50% of all blocks replaced are dirty. The 128-bit wide main memory has an access latency of 60 ns, after which any number of bus words may be transferred at the rate of one per cycle on the 128-bit wide 133 MHz main memory bus.

In light of the above-specified system design, answer the following:

- What is the average memory access time for instruction accesses?
- What is the average memory access time for data reads?
- What is the average memory access time for data writes?
- What is the overall CPI, including memory accesses?

[10]

- A particular architecture implementation has 32-bit virtual addresses, 32-bit physical addresses and a page size of  $2^{12}$  bytes. A test program has been running on this architecture and has been halted just before execution of the following instruction at location 0x1FFC:

LD R31, 0x34C8(R1) | PC = 0x1FFC

ST R1, 0x6004(R31) | PC = 0x2000

The first 8 locations of the page table at the time execution was halted are shown below; the least recently used page (“LRU”) and next least recently used page (“next LRU”) are as indicated. Assume that all the pages in physical memory are in use. Also assume that in case of a page fault, the LRU page is reused and its previous entry is invalidated. Execution resumes and the LD and ST instructions are executed.

Page Number	Dirty Bit	Valid Bit	Frame Number
0	1	1	0x1
1	0	1	0x0
LRU → 2	1	1	0x6
3	-	0	-
Next LRU → 4	0	1	0x4
5	0	1	0x2
6	0	1	0x7
7	0	1	0x3

- Which physical pages, if any, are needed to be written to disk during the execution of the LD and ST instructions?
- What is the 32-bit physical memory address of LD instruction (in hex)?
- What is the 32-bit physical memory address of data read by LD (in hex)?
- What is the 32-bit physical memory address of ST instruction (in hex)?
- What is the 32-bit physical memory address of data written by ST (in hex)?

[10]