University of Southern California

Department of Electrical Engineering

EE557 Fall 2025

Instructor: Murali Annavaram

Homework #0, Due: 11:59 PM., Sunday, Sept. 14th

TOTAL SCORE: 180

Problem 1 [Power/Energy] (20 pts.)

Problem 2.1 in the book, with the following modifications, considering the following designs for the problem:

A. 5-stage pipeline clocked at 4f

B. 5-stage pipeline clocked at f

C. 5-way multiprocessor in which each processor is a 5-stage pipeline clocked at 4f

D. 5-way multiprocessor in which each processor is a 5-stage pipeline clocked at f

Compare these four designs with the base machine, which is the single-cycle processor clocked at f.

Problem 2 [Amdahl’s law] (30 pts.)

The combination of two enhancements is considered to boost the performance of a chip multiprocessor. The enhancements are: 1) adding more cores or 2) adding more shared level-2 cache. **The base chip has 2 cores and 4 L2 cache banks**. L2 cache size can be increased by adding cache banks. Each cache bank uses **five times** the area of a core. Here is what we also know from all kinds of sources:

1) The workload is evenly distributed across all 2 cores.
2) **90%** of the time each core executes instructions that can be fully parallelized; the remaining **10%** cannot.
3) The core stall time due to L2 misses accounts for **50%** of each core's execution time in the base configuration.
4) The cycles spent in (2) and (3) above may overlap. For example, some instructions can run in parallel and at the same time be stalled due to L2 misses.
5) It is suspected if the amount of shared L2 cache per core remains constant, the miss rate will be the same.
6) Simulations have also determined that the miss rate of L2 decreases as the square root of its size per core. A conjecture is that the stall time in each core will also decrease as the square root of L2 size per core.

The company that pays your paycheck has acquired a new technology to build large micro-chips, so that the next generation chips will have **six times the area of current chips** to dedicate to cores and L2 caches. Given what you know, what kind of best "first cut" design would you propose? A design is characterized by (# of cores, # of L2 cache banks). These numbers can be any integer. The design should be contained in the new chip. Estimate the maximum speedup of the best design that takes advantage of the new chip real estate.

Problem 3 [Amdahl’s law (reversed)] (20pts)

A baseline microprocessor is enhanced as follows.
1) The core is replicated 4 times to form a 4-way CMP.
2) A floating-point co-processor is added to each core. This co-processor speeds up all floating-point operations in each core by a factor 8 and is attached to the cores. While the floating point co-processor is active, the host core is idle and the presence of the co-processor does not affect instructions that are not part of floating point operations.

We observe the following on the NEW machine:
1) the floating-point co-processor is used during 45% of the execution time of each core.
2) 20% of the time only one core is active, 25% of the time 2 cores are active, and the rest of the time the 4 cores are active. Note that a core is considered active whenever it or its FP co-processor is active.

What is the speedup of the new machine over the base machine?

Problem 4 [Data hazard] (30 pts.)

Exercise 3.4 in the textbook, but for the following program:

```
SEARCH:    LW R5,0(R3)          /I1   Load item
           SUB R6,R5,R2         /I2   Compare with key
           ADDI R1,R1,#1        /I3   Count matches
           BEZ R6,MATCH         /I4   Check for match
           SUBI R1,R1,#1        /I5   Decrement matches
MATCH:     ADDI R3,R3,#4        /I6   Next item
           BNE R4,R3,SEARCH     /I7 Continue until all items
```

Sub-problem (e) should also change as follows: replace I5 and BNEZ with I6 and BEZ respectively.

Problem 5 [Multi-pipeline design] (20 pts.)

Consider a machine with three pipelines as below in Figure 1: integer/load/store at the top, FP arithmetic in the middle, and integer multiplication at the bottom. Each of the integer multiplication stages (M1 and M2) takes two clocks with a counter attached to each of the stages showing how many cycles spent for a valid instruction. Register values whenever available are forwarded to the EX, FP1 and M1 stages. Branches are predicted as always untaken and branch outcome is known in the EX stage. Instructions are stalled only in the ID stage if necessary.

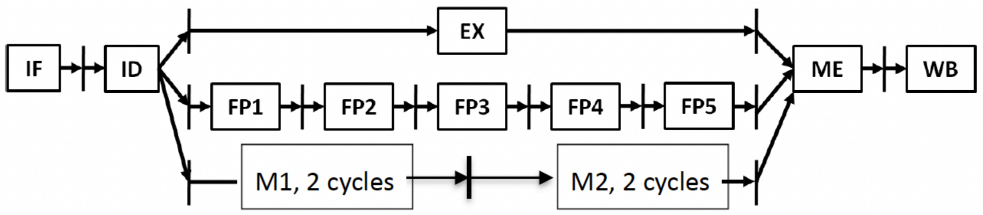

Figure 1. Machine with three pipelines

a. What are the operation latencies of the following instructions: 1) Reg-to-reg, 2) load, 3) FP arithmetic and 4) integer multiplication?

b. How are structural hazards, if any, handled? Provide sources of structural hazards and pipeline registers to be checked in the ID stage for possible stalls.

c. Which pipeline registers must be checked in the ID stage to detect both RAW and WAW hazards?

d. How is branch misprediction handled? What is the branch misprediction penalty in clock cycles?

Problem 6 [Static scheduling] (30pts.)

In this problem, we compare execution times with different static scheduling techniques done at compile time. The code snippet below will be run on the machine in Figure 1 above. Compare the execution cycles of (a) no scheduling, (b) local scheduling, and (c) loop unrolling twice. The BNE at the bottom is taken always and resolved in the EX stage. Show all codes after scheduling with any displacement adjustment if necessary. What are the speedups after local scheduling and loop unrolling by the compiler with respect to the code with no compiler scheduling? Branches are not delayed.

```
LOOP:        LW R3,0(R1)
             MUL R4,R3,R3
             ADDI R6,R6,#2
             ADD R4,R4,R6
             SW R4,0(R2)
             ADDI R1,R1,#4
             ADDI R2,R2,#4
             BNE R5,R2,LOOP
```

Problem 7 [Tomasulo Algorithm] (30pts.)

This problem concerns Tomasulo's algorithm with non-speculative execution. Tomasulo's algorithm and the processor that supports it are described in section 3.4.1, pictured in Figure 3.15 of your book and you can find below a copy of Figure 3.15 for your reference.

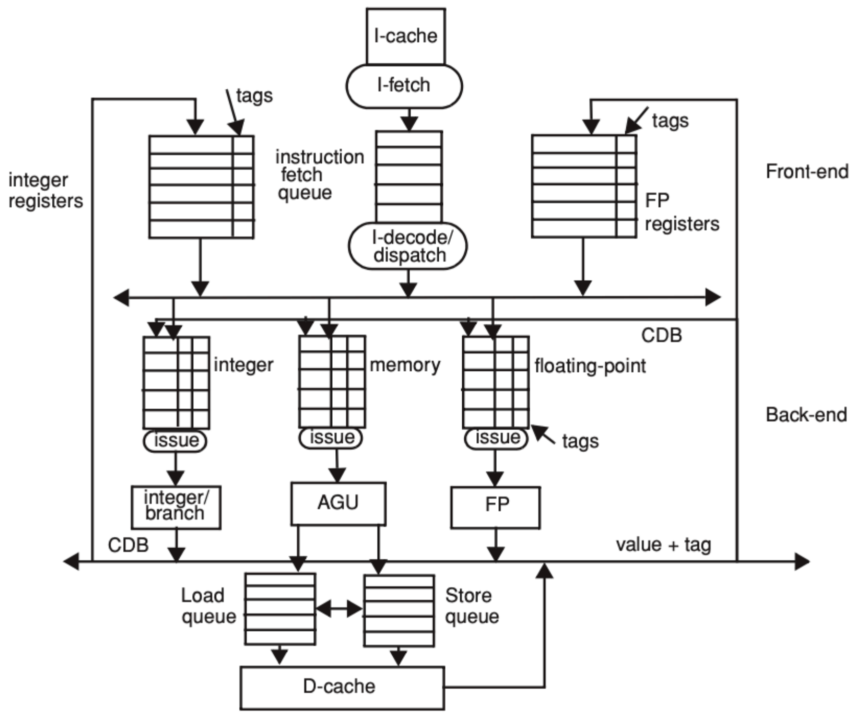

Figure 3.15. Hardware for Tomasulo algorithm.

Assume **ADDI, MUL.D take 1 cycle**. Please fill out the execution sequence table based on the following instruction sequence:

```
LOOP:   L.D F0,0(R1)          /X[i] loaded in F0
        L.D F2,0(R2)          /Y[i] loaded in F2
        MUL.D F8,F2,F0        /Multiply X by Y
        ADDI R1,R1,#8         /update address registers
        ADDI R2,R2,#8
        ADDI R3,R3,#8
        S.D F8, 0(R3)         /store in Z[i]
        BNE R4,R1,LOOP/       /(R4)-8 points to the last element of X
```

	Instruction	Dispatch	Issue	Exec start	Exec complete	Cache	CDB	COMMENTS
I1	L.D F0,0(R1)							
I2	L.D F2,0(R2)							
I3	MUL.D F8,F2,F0							
I4	ADDI R1,R1,#8							
I5	ADDI R2,R2,#8							
I6	ADDI R3,R3,#8							
I7	S.D-A F8, 0(R3)							
I8	S.D-D F8, 0(R3)							
I9	BNE R4,R1,LOOP							
I10	L.D F0,0(R1)							