Dipartimento di Elettronica e Informazione

Politecnico di Milano

20133 Milano (Italia)
Piazza Leonardo da Vinci, 32
Tel. (+39) 02-2399.3400
Fax (+39) 02-2399.3411

Advanced Computer Architecture

August 31, 2023

Prof. D. Conficconi, and M. D. Santambrogio

Name
Last Name
Professor:

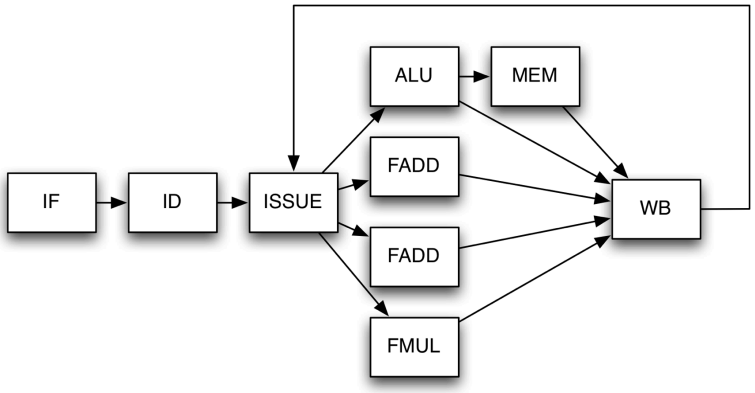
The exam will last 90’

NOTE: the exam is passed IFF (10pts from EXE && 6pts from Theory && min grade 18 pts)

Problem 1 (6pt) – EXE	
Problem 2 (6pt) – EXE	
Problem 3 (6pt) – EXE	
Problem 4 (5pt) – Theory	
Problem 5 (5pt) – Theory	
Problem 6 (5pt) – Theory	
Total (33pt)	

Problem 1

Considering the following single-issue out-of-order processor:



- All functional units are pipelined
- There is no register renaming. No forwarding
- Instructions are fetched, decoded and issued in order
- The ISSUE stage is a buffer of unlimited length that holds instructions waiting to start execution
- An instruction will only enter the issue stage if it does not cause a WAR or WAW hazard
- Only one instruction can be issued at a time, and in the case multiple instructions are ready, the oldest one will go first
- Program Counter calculation for branches and jumps has been anticipated in the ISSUE stage
- The target address for a branch is available in the FETCH stage and there is a branch prediction technique.

Consider the following code:

```
I1: loop: ld f1, 0(r6)
I2: fsub f1, f1, f2
I3: ld f3, 0(r7)
I4: fadd f2, f1, f3
I5: st f2, 0(r6)
I6: addi r6, r6, 4
I7: addi r7, r7, 4
I8: bne r6, r8, loop
I9: next loop iteration
```

1. Highlights all the conflicts

2. Is the following execution table correct? If yes, how many functional units and which latency they have?

	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13
I1	F	D	IS	EX	EX	EX	WB							
I2		F	S(D)	S(D)	S(D)	D	IS	EX	EX	EX	WB			
I3			S(F)	S(F)	S(F)	S(F)	F	D	IS	EX	EX	EX	WB	
I4								F	D	S(IS)	S(IS)	S(IS)	IS	EX
I5									F	D	S(IS)	S(IS)	S(IS)	S(IS)
I6										F	S(D)	S(D)	S(D)	S(D)
I7														
I8														
I9														

NOTE: The table continues below!!!

	[…]	**C14**	**C15**	**C16**	**C17**	**C18**	**C19**	**C20**	**C21**	**C22**	**C23**	**C24**	**C25**
I1													
I2													
I3													
I4		EX	EX	EX	WB								
I5		S(IS)	S(IS)	S(IS)	IS	EX	EX	EX	WB				
I6		S(D)	S(D)	S(D)	S(D)	S(D)	S(D)	D	IS	EX	WB		
I7								F	D	IS	EX	WB	
I8									F	D	IS	EX	WB
I9										F	D	IS	EX

Problem 2

Please consider the program in the table be executed on a CPU with dynamic scheduling based on TOMASULO. Consider an architecture with:

- 4 RESERVATION STATIONS (RS1, RS2, RS3, RS4) + 2 LDU units (LDU1, LDU2) with latency 2 cycles
- 1 RESERVATION STATION (RS5) + 1 MULT (MULT1) with latency 10 cycles
- 1 RESERVATION STATIONS (RS6) + 1 Integer ALU (ALU1) with latency 1 cycle
- In case of hazard on the CDB, assume that the oldest instruction has priority.

Consider the following code:

I0: lw $2, 32($fp)

I1: sw $2, 24($fp)

I2: lw $3, 28($fp)

I3: mult $2, $3, $2

I4: sw $2, 28($fp)

I5: lw $4,26($fp)

I6: addi $4, $4, -1

I7: sw $4, 26($fp)

1. Please complete the TOMASULO TABLE by assuming all cache HITS, apart from I7 where a cache miss happens and I7 needs 6cc to complete the execution.

INSTRUCTION	ISSUE	START EXE	EXE COMP.	WRITE BACK	Hazards Type	RS	UNIT
I0: lw $2, 32($fp)							
I1: sw $2, 24($fp)							
I2: lw $3, 28($fp)							
I3: mult $2, $3, $2							
I4: sw $2, 28($fp)							
I5: lw $4,26($fp)							
I6: addi $4, $4, -1							
I7: sw $4, 26($fp)							

2. Compute the following metrics:

- CPI = (#clock cycles / IC) =
- IPC = 1/CPI =

- **Problem 3**
- Consider the following access pattern on a two-processor system with a direct-mapped, write-back cache with one cache block and a two cache block memory.
- Assume the **MESI protocol** is used, with **write-back** caches, **write-allocate,** and **write-invalidate** of other caches.

Cycle	After Operation	Px cache block state	Py cache block state	Memory at block 0 up to date?	Memory at block 1 up to date?
0	Px read block 0				
1	Px read block 0				
2	Py read block 0				
3	Py write block 0				
4	Py write block 0				
5	Py read block 0				
6	Px write block 1				
7	Py write block 1				
8	Px read block 1				
9	Py read block 1				
10	Py write block 1				
11	Py write block 1				
12	Py read block 1				
13	Px write block 0				
14	Px read block 0				
15	Py: write block 1				

Question 4 (5 points)

Explain the main characteristics of static scheduling and its implication. Also discuss the trace scheduling and software pipelining techniques.

Question 5 (5 points)

Describe the Flynn's taxonomy providing for each class the main features and real examples.

Question 6 (5 points)

Which problem solve the branch prediction? Please list a technique for each of the two main classess of branch prediction and describe it briefly.

1

```
I1: loop: ld f1, 0(r6)
I2: fsub f1, f1, f2
I3: ld f3, 0(r7)
I4: fadd f2, f1, f3
I5: st f2, 0(r6)
I6: addi r6, r6, 4
I7: addi r7, r7, 4
I8: bne r6, r8, loop
I9: next loop iteration
```

RAW F1 I1-I4
RAW F1 I2-I4
RAW r6 I6-I8
RAW F3 I3-I4
RAW F2 I4-I5

WAR r6 I1-I6
WAR r6 I5-I6
WAR r7 I3-I7
WAR F2 I2-I4
WAR F1 I1-I2

	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13
I1	F	D	IS	EX	EX	EX	WB							
I2		F	S(D)	S(D)	S(D)	D	IS	EX	EX	EX	WB			
I3			S(F)	S(F)	S(F)	S(F)	F	D	IS	EX	EX	EX	WB	
I4								F	D	S(IS)	S(IS)	S(IS)	IS	EX
I5									F	D	S(IS)	S(IS)	S(IS)	S(IS)
I6										F	S(D)	S(D)	S(D)	S(D)
I7														
I8														
I9														

OK IF 1 ALU

NOTE: The table continues below!!!

	[...]	C14	C15	C16	C17	C18	C19	C20	C21	C22	C23	C24	C25
I1													
I2													
I3													
I4		EX	EX	EX	WB								
I5		S(IS)	S(IS)	S(IS)	IS	EX	EX	EX	WB				
I6		S(D)	S(D)	S(D)	S(D)	S(D)	S(D)	D	IS	EX	WB		
I7								F	D	IS	EX	WB	
I8									F	D	IS	EX	WB
I9										F	D	IS	EX

1 ALU 1cc

1 FP+ 4cc

1 MEM 3cc

1 FP- 3cc

2

```
I0: lw $2, 32($fp)
I1: sw $2, 24($fp)
I2: lw $3, 28($fp)
I3: mult $2, $3, $2
I4: sw $2, 28($fp)
I5: lw $4,26($fp)
I6: addi $4, $4, -1
I7: sw $4, 26($fp)
```

RAW $2 10-11 RAW $3 12-13

WAW $2 10-13 WAW $4 15-16

RAW $2 13-14 RAW $4 16-17

WAR $2 13-14 CAVOIDED IN TOMASULOS

- 4 RESERVATION STATIONS (RS1, RS2, RS3, RS4) + 2 LDU units (LDU1, LDU2) with latency 2 cycles
- 1 RESERVATION STATION (RS5) + 1 MULT (MULT1) with latency 10 cycles
- 1 RESERVATION STATIONS (RS6) + 1 Integer ALU (ALU1) with latency 1 cycle
- In case of hazard on the CDB, assume that the oldest instruction has priority.

INSTRUCTION	ISSUE	START EXE	EXE COMP.	WRITE BACK	Hazards Type	RS	UNIT
I0: lw $2, 32($fp)	1	2		4		RS1	LDU1
I1: sw $2, 24($fp)	2	5		7		RS2	LDU2
I2: lw $3, 28($fp)	3	5		8	STRUCT LDU1 STRUCT CDB	RS3	LDU1
I3: mult $2, $3, $2	4	9		19		RS5	MULT1
I4: sw $2, 28($fp)	5	20		21		RS1	LDU1
I5: lw $4,26($fp)	6	8		10	STRUCT LDU	RS4	LDU1
I6: addi $4, $4, -1	7	11		12		RS6	ALU1
I7: sw $4, 26($fp)	8	13		22	LATENCY STRUCT CDB	RS2	LDU2

$$CPI = \frac{\#CC}{\#I} = \frac{22}{8} = 2,75$$

$$IPC = \frac{1}{CPI} = 0,36$$

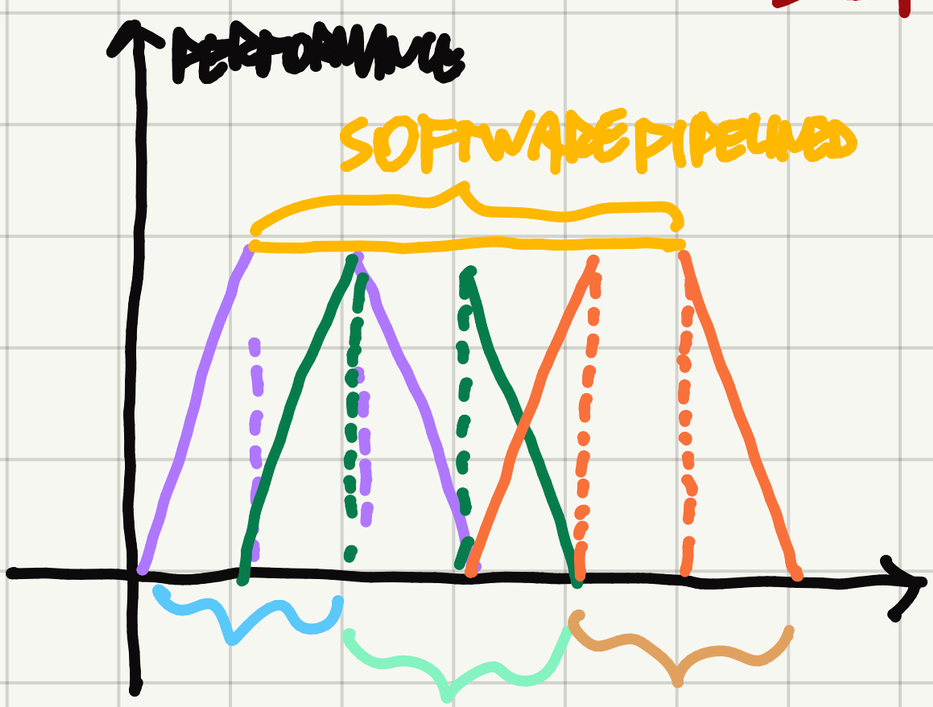
3

Cycle	After Operation	Px cache block state	Py cache block state	Memory at block 0 up to date?	Memory at block 1 up to date?
0	Px read block 0	E(0)	I	YES	YES
1	Px read block 0	E(0)	I	YES	YES
2	Py read block 0	S(0)	S(0)	YES	YES
3	Py write block 0	I	M(0)	NO	YES
4	Py write block 0	I	M(0)	NO	YES
5	Py read block 0	I	M(0)	NO	YES
6	Px write block 1	M(1)	M(0)	NO	NO
7	Py write block 1	I	M(1)	YES	NO
8	Px read block 1	S(1)	S(1)	YES	YES
9	Py read block 1	S(1)	S(1)	YES	YES
10	Py write block 1	I	M(1)	YES	NO
11	Py write block 1	I	M(1)	YES	NO
12	Py read block 1	I	M(1)	YES	NO
13	Px write block 0	M(0)	M(1)	NO	NO
14	Px read block 0	E M(0)	M(1)	Y NO	NO
15	Py: write block 1	E M(0)	M(1)	Y NO	NO

4

CODE ANALYSIS AND REORDERING TO MAXIMIZE THE F.U.'S USAGE FOR EACH CLOCK CYCLE AND MINIMIZE INACTIVE CYCLES.

SOFTWARE PIPELINING

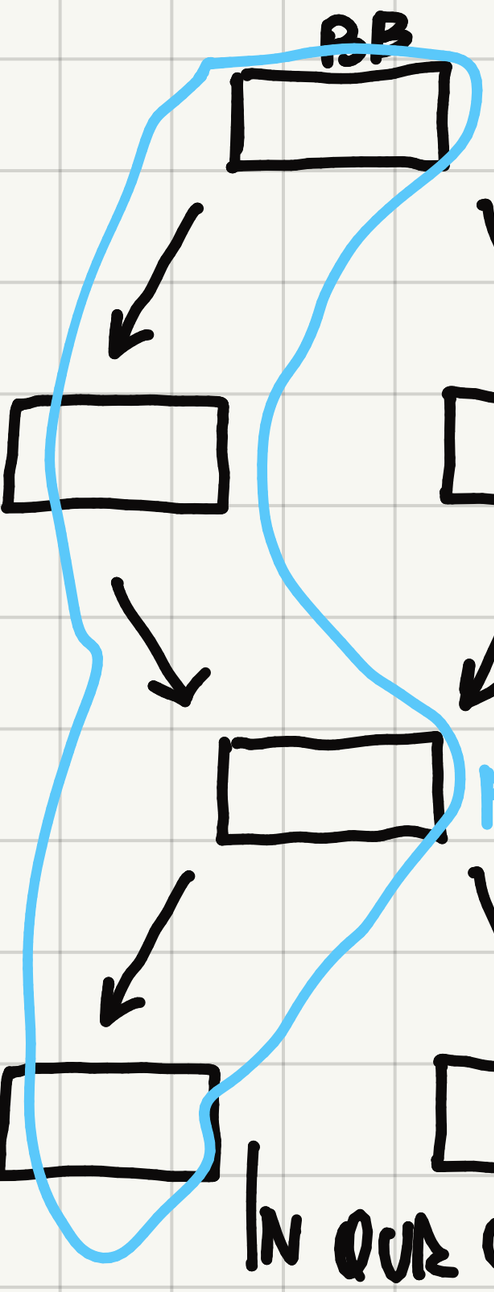


PARALLELIZE LOOP ITERATIONS

- PROLOGO → INIZIALIZZAZIONE PIPELINE
- LOOP ITERATION (G) → ESECUZIONE OTTIMIZZATA
- EPILOGO → SVUOTO PIPELINE

OVERHEAD TIMES "LOST" ONLY ONCE AT THE BEGINNING AND AT THE END, BUT WHAT IF LOOPS DEPEND FROM EACH OTHER?

TRACE SCHEDULING



BB = BASIC BLOCK

TRACE: SEQUENCE OF LINEAR INSTRUCTIONS, NO INTERMEDIATE BRANCHES NOR JUMPS, ONE ENTRY AND ONE EXIT

DATA DEPENDENCIES LIMIT ILP. FOR A BETTER PERFORMANCE, EXPLOIT ILP ON DIFFERENT BB.

IN OUR CONTEXT, FIND RAW/WAR/WAW, SOLVE BY REORDING OR, IF NOT POSSIBLE, WITH NOP

TRACE SCHEDULING CANNOT PROCEED BEYOND A LOOP BARRIER. LOOP UNROLLING CAN OVERCOME THIS LIMITATION BUT IT ADDS EXTRA CODE AND LOSES PERFORMANCES

MORE ADVANCED TECHNIQUES (HYPERBLOCK/SPECULATIVE SCHEDULING) TO ELIMINATE CONTROL DEPENDENCIES

5

FLYNN TAXONOMY DESCRIBES COMPUTER ARCHITECTURES BASED ON INSTRUCTION MANAGEMENT AND DATA FLOW

- SISD) ONLY ONE INSTRUCTIONS AT A TIME, USING ONE DATA
  - COMPILATOR, THAT SEQUENTIALLY ANALYSE A GIVEN CODE
- MISD) MULTIPLE INSTRUCTION IN PARALLEL OVER SAME DATA
  - NO REAL WORLD APPLICATION
- SIMD) ONE INSTRUCTION WORKS WITH MULTIPLE DATA FLOW DLP
  - GRAPH, IMAGE PROCESSING ARRAY, MATRIXES OPERATION
- MIMD) MULTIPLE INSTRUCTION WORKS WITH MULTIPLE DATA FLOW TLP
  - MULTICORES, CLOUD COMPUTING, EMBEDDED DEVICES

6

BRANCH PREDICTION TRIES TO "ANTICIPATE" EVALUATION OF A BRANCH CONDITION AT THE D STAGE, GENERALLY BEFORE THE DATA ARE ACTUALLY WRITTEN BACK BY THE INSTRUCTION THAT ARE WORKING ON THEM

STATIC PREDICTIONS BASES ITS DECISION ON THE CHOSEN APPROACH AND NEVER CHANGES; DYNAMIC PREDICTIONS CHANGES WHETHER TO CHANGE OR NOT THE PREDICTION ON TAKING THE BRANCH OR NOT

STATIC BRANCH PREDICTION: DELAYED BRANCH PREDICTION

ADD USEFUL INSTRUCTIONS (OR NOP IF THERE ARE ANY) AFTER THE BRANCH TO ENSURE DATA AVAILABILITY BEFORE DECIDING WHERE TO JUMP

DYNAMIC BRANCH PREDICTION: 1-BHT WITH COLLISION

ALL BRANCHES ARE RELATED TO EACH OTHER, MEANING THAT A PREDICTION MADE FOR A BRANCH (TAKEN/NOT TAKEN) AFFECTS THE PREDICTION OF THE SUCCESSIVE BRANCH, EVEN IF IT IS RELATED TO ANOTHER INSTRUCTION