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

June 13, 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 (7pt) – Theory	
Problem 4 (6pt) – EXE	
Problem 5 (8pt) – Theory	
Total (33pt)	

Problem 1

Please consider the program in the table be executed on a CPU with dynamic scheduling based on SCOREBOARD BASIC SCHEME with:

- 2 LOAD/STORE units (LDU1, LDU2) with latency 4 cycles
- 1 FP UNIT (FPU1) with latency 3 cycles
- 1 ALU/BR UNIT (ALU1) with latency 1 cycle

1. Please complete the SCOREBOARD TABLE by assuming all cache HITS

ISTRUZIONE	ISSUE	READ OPS	EXEC COMPL	WRITE BACK	Hazards Type	UNIT
I0 lw $f5,0($r1)						
I1 lw $f6,0($r1)						
I2 fadd $f6,$f6,$f5						
I3 sw $f6,0($r1)						
I4 lw $f6,4($r1)						
I5 fadd $f6,$f6,$f6						
I6 sw $f6,4($r1)						
I7 addi $r1,$r1,8						

2. Express the formula then compute the following metrics:
   - CPI =
   - IPC =

Problem 2

In this problem, you will port code to a simple **3-issue VLIW machine,** and schedule it to improve performance.
Details about the 3-issue VLIW machine with 3 fully pipelined functional units:

- Integer ALU with **1** cycle latency to next Integer/FP and **2** cycle latency to next Branch
- Memory Unit with **3** cycle latency
- Floating Point Unit with **3** cycle latency (each FPU can complete one add or one multiply per clock cycle)
- Branch completed with **1 cycle delay slot** (branch solved in ID stage)

C Code:

```
for(int i=0; i<N; i++)
     A[i] = A[i] + B[i];
```

Assembly Code:

```
loop:ld   f1, 0(r6)
     ld   f2, 0(r6)
     fadd f2, f2, f1
     st   f2, 0(r6)
     addi r6, r6, 4
     bne  r6, r7, loop
```

ESERCISE 2.A

Considering **one iteration** of the loop, **schedule** the assembly code for the 3-issue VLIW machine in the following table by using the **list-based scheduling,** but neither using software pipelining nor loop unrolling **nor modifying loop indexes.** Please do not need to write in NOPs (can leave blank).

	Integer ALU	Memory Unit	FPU
C0			
C1			
C2			
C3			
C4			
C5			
C6			
C7			
C8			
C9			
C10			
C11			
C12			
C13			

How long is the Critical Path?

What performance did you achieve in FP ops per cycle?

EXERCISE 2.B

Considering the unrolling of two iterations of the loop as follows:
Assembly Code:

```
loop:ld   f1, 0(r6)
     ld   f2, 0(r6)
     fadd f2, f2, f1
     st   f2, 0(r6)
     ld   f3, 4(r6)
     ld   f4, 4(r6)
     fadd f4, f4, f3
     st   f4, 4(r6)
     addi r6, r6, 8
     bne  r6, r7, loop
```

Considering **one iteration of the unrolled loop,** schedule the assembly code by using the **list-based scheduling,** but neither using software pipelining nor loop unrolling **nor modifying loop indexes.** Please do not need to write in NOPs (can leave blank).

	Integer ALU	Memory Unit	FPU
C0			
C1			
C2			
C3			
C4			
C5			
C6			
C7			
C8			
C9			
C10			
C11			
C12			
C13			
C14			
C15			

How long is the Critical Path? 13 cycles

What performance did you achieve in FP ops per cycle? 2/13

Problem 3

3.1. Superscalar processors today use an approach that exploits both Instruction Level Parallelism and Thread Level Parallelism. Briefly explain the approach

3.2. Which HW mechanisms of dynamic scheduled processors intrinsically support this approach?

3.3. Which modifications to a generic architecture of dynamically scheduled superscalar processors are required?

Problem 4

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	P0 cache block state	P1 cache block state	Memory at block 0 up to date?	Memory at block 1 up to date?
0	P0: read block 1				
1	P1: read block 0				
2	P0: write block 0				
3	P0: write block 1				
4	P1: read block 1				
5	P1: write block 1				
6	P0: read block 0				
7	P0: write block 0				
8	P1: read block 0				
9	P0: write block 1				
10	P1: write block 1				
11	P0: read block 0				
12	P1: write block 1				
13	P1: read block 1				
14	P0: read block 1				
15	P1: write block 1				

Question 5 (8 points)

Answer with **True** or **False** to the following questions:

Q1) SIMD architectures can exploit significant data parallelism in a single operation

Answer: T F

Q2) Multithreading aims at speeding up the execution time of each individual thread

Answer: T F

Q3) When using Explicit Register Reaming the ROB is no longer necessary but it can still be used to achieve precise exceptions in an easier way.

Answer: T F

Q4) Mark the True answers to the following question, note that It might have multiple true answers.
Which complications are introduced to MIPS pipelined processors by adding out-of-order execution and out-of-order completion?

Answer 1: WAR and WAW hazards	T
Answer 2: RAW hazards	T
Answer 3: Interrrupt/Exception handling	T
Answer 4: Control hazards	T
Answer 5: Instruction Reordering	T

1

```
lw $f5,0($r1)
lw $f6,0($r1)
fadd $f6,$f6,$f5

sw $f6,0($r1)
lw $f6,4($r1)

fadd $f6,$f6,$f6

sw $f6,4($r1)
addi $r1,$r1,8
```

MEM 4cc LDU1, LDU2

FP 3cc FPU1

ALU 1cc ALU1

RAW $F5 11-13 WAW $F6 12-13

RAW $F6 12-13 WAW $F6 15-16

RAW $F6 12-14

RAW $F6 15-16 WAR $r1 17-18

RAW $F6 16-17 WAR $F6 13-15

ISTRUZIONE	ISSUE	READ OPS	EXEC COMPL	WRITE BACK	Hazards Type	UNIT
I0 lw $f5,0($r1)	1	2	6	7		LDU1
I1 lw $f6,0($r1)	2	3	7	8		LDU2
I2 fadd $f6,$f6,$f5	9	10	13	14	●	FPU1
I3 sw $f6,0($r1)	10	15	19	20	●	LDU1
I4 lw $f6,4($r1)	15	16	20	21	●	LDU2
I5 fadd $f6,$f6,$f6	22	23	26	27	●	FPU1
I6 sw $f6,4($r1)	23	28	32	33	●	LDU1
I7 addi $r1,$r1,8	24	25	26	29	●	ALU1

$$CPI = \frac{\#c.c.}{\#I} = \frac{33}{8} = 4,125$$

$$IPC = \frac{1}{CPI} = 0,24$$

2

A

	Integer ALU	Memory Unit	FPU
C0		ld F1, 0(r6)	
C1		ld F2, 0(r6) 1	
C2		2	
C3		3	
C4			1 Fadd F2, F2, F1
C5			2
C6			3
C7	addi r6, r6, 4 1	st F2, 0(r6)	
C8	2		
C9	bne r6, r7, loop		
C10	DELAY		
C11	NEXT INSTRUCTION		
C12			
C13			

CRITICAL PATH = 11 cc

$$\frac{FP}{CYCLES} = \frac{1}{11}$$

B

	Integer ALU	Memory Unit	FPU
C0		ld F1, 0(r6)	
C1		ld F2, 0(r6) 1	
C2		ld F3, 4(r6) 2	
C3		ld F4, 4(r6) 1 3	
C4		2	1 Fadd F2, F2, F1
C5		3	2
C6		st F2, 0(r6) 1	3 fadd F4, F4, F3
C7			2
C8			3
C9	addi r6, r6, 4	st F4, 4(r6)	
C10	DELAY		
C11	bne r6, r7, loop		
C12	NEXT INSTRUCTION		
C13			

CRITICAL PATH = 13 cc

$$\frac{FP}{CYCLES} = \frac{2}{13}$$

3

a) MIMD ARCHITECTURES ENSURE TLP; MODERN SUPERSCALAR ARCHITECTURES AIM ON DYNAMICALLY PARALLELIZE SET OF INSTRUCTION. THE IDEAL MULTITHREADING ARCHITECTURE TO EXPLOIT IS SIMULTANEOUS MULTITHREADING SMT

b) WE CAN THINK ABOUT DYNAMIC APPROACHES TECHNIQUES THAT CAN ENSURE ILP. EXAMPLES ARE:

- SCOREBOARDS WITH EXPLICIT REGISTER RENAMING
- TOMASULO WITH ROB
- SCOREBOARD AND TOMASULO THEMSELVES, AS OUT-OF-ORDER EXECUTION

c)
- SCOREBOARD AND TOMASULO WITH MULTIPLE P.C.
- SEPARATION OF ROB

4

COMMIT 0 TO WORK ON 1

Cycle	After Operation	P0 cache block state	P1 cache block state	Memory at block 0 up to date?	Memory at block 1 up to date?
0	P0: read block 1	E(1)	I	YES	YES
1	P1: read block 0	E(1)	E(0)	YES	YES
2	P0: write block 0	M(0)	I	NO	YES
3	P0: write block 1	M(1)	I	YES	NO
4	P1: read block 1	S(1)	S(1)	YES	YES
5	P1: write block 1	I	M(1)	YES	NO
6	P0: read block 0	E(0)	M(1)	YES	YES
7	P0: write block 0	M(0)	M(1)	NO	NO
8	P1: read block 0	S(0)	S(0)	YES	NO
9	P0: write block 1	M(1)	E(0)	YES	NO
10	P1: write block 1	I	M(1)	YES	NO
11	P0: read block 0	E(0)	M(1)	YES	NO
12	P1: write block 1	E(0)	M(1)	YES	NO
13	P1: read block 1	E(0)	M(1)	YES	NO
14	P0: read block 1	S(1)	S(1)	YES	YES
15	P1: write block 1	I	M(1)	YES	NO

P1 OWNS 0 ALONE

5

Q1) SIMD architectures can exploit significant data parallelism in a single operation

Answer: T ✓ F

Q2) Multithreading aims at speeding up the execution time of each individual thread

Answer: T F ✓ AIMS AT SPEEDING UP OVERALL EXECUTION

Q3) When using Explicit Register Reaming the ROB is no longer necessary but it can still be used to achieve precise exceptions in an easier way.

Answer: T ✓ F

Q4) Mark the True answers to the following question, note that It might have multiple true answers.
Which complications are introduced to MIPS pipelined processors by adding out-of-order execution and out-of-order completion?

Answer 1: WAR and WAW hazards	T ✓
Answer 2: RAW hazards	T
Answer 3: Interrrupt/Exception handling	T ✓
Answer 4: Control hazards	T
Answer 5: Instruction Reordering	T