Surname	
Name	
POLIMI Personal code	
Signature	

SOLUTION

Politecnico di Milano, 6 May 2024

Course on Advanced Computer Architectures

Prof. C. Silvano

EX 1	**(4 points)**	
EX 2	**(5 points)**	
EX 3	**(4 points)**	
QUIZ 4	**(1 point)**	
QUIZ 5	**(1 point)**	
TOTAL	**(15 points)**	
QUIZ 6 OPTIONAL	**+ 3 extra points**	

EXERCISE 1: VLIW (4 points)

Let's consider the following assembly code:

```
INIT:  ADDUI $R1, $R0, 0
       ADDUI $R2, $R0, 40
       ADDUI $R4, $R0, 8

LOOP1: LD $F0, 0 ($R1)
       FADD $F4, $F0, $F2
       SD $F4, 0 ($R1)
       ADDUI $R3, $R0, 0

LOOP2: LD $F6, 0($R3)
       FADD $F8, $F6, $F2
       SD $F8, 0($R3)
       ADDUI $R3, $R3, 4
       BNE $R3, $R4, LOOP2

       ADDUI $R1, $R1, 4
       BNE $R1, $R2, LOOP1
```

1. *Complete the code of the first iteration of the outer loop* ***LOOP1:***

```
LOOP1: LD $F0, 0 ($R1)
       FADD $F4, $F0, $F2
       SD $F4, 0 ($R1)
       ADDUI $R3, $R0, 0
```

2. *Now schedule* ***the first iteration of the outer loop LOOP1*** *by using* ***the list-based scheduling*** *(do NOT introduce any software pipelining, loop unrolling and modifications to loop indexes) on a* ***3-issue VLIW*** *machine with* ***fully pipelined functional units:***

- ***1 Memory Unit with 2 cycle latency***
- ***1 Integer ALU with 1 cycle latency to next Int/FP/LD/SD & 2 cycle latency to next Branch***
- ***1 FP ALU with 2 cycle latency***

There is no branch prediction. *The branch is completed with* ***1 cycle delay slot*** *(branch solved in ID stage).*
In the Register File, it is possible to read and write at the same address at the same clock cycle.
Please do not write in NOPs.

	Memory Unit	Integer Unit	Floating Point Unit
C1	LD $F0, 0 ($R1)		
C2			
C3			
C4			
C5			
C6			
C7			
C8			
C9			
C10			
C11			
C12			
C13			
C14			
C15			
C16			
C17			
C18			
C19			
C20			
C21			
C22			
C23			
C24			
C25			

3. How long is the critical path? ___________________________________

4. What performance did you achieve in CPIas?

   ____

5. What performance did you achieve in FP ops per cycles?

   ____

6. How much is the code efficiency?

   ____

Feedback

1. *Complete the code of the first iteration of the outer loop* ***LOOP1:***

```
LOOP1: LD $F0, 0 ($R1)
       FADD $F4, $F0, $F2
       SD $F4, 0 ($R1)
       ADDUI $R3, $R0, 0
LOOP2: LD $F6, 0($R3)
       FADD $F8, $F6, $F2
       SD $F8, 0($R3)
       ADDUI $R3, $R3, 4
       BNE $R3, $R4, LOOP2
LOOP2: LD $F6, 0($R3)
       FADD $F8, $F6, $F2
       SD $F8, 0($R3)
       ADDUI $R3, $R3, 4
       BNE $R3, $R4, LOOP2
       ADDUI $R1, $R1, 4
       BNE $R1, $R2, LOOP1
```

2. *Now schedule* ***the first iteration of the outer loop LOOP1***

	Memory Unit 1	Integer Unit	Floating Point Unit
C1	LD $F0,0($R1)	ADDUI $R3,$R0,0	
C2	LD $F6,0($R3)		
C3			FADD $F4, $F0, $F2
C4			FADD $F8, $F6, $F2
C5	SD $F4,0(R1)		
C6	SD $F8,0($R3)	ADDUI R3,R3,4	
C7			
C8		BNE $R3,$R4, LOOP2	
C9		Br. delay slot	
C10	LD $F6,0($R3)		
C11			
C12			FADD $F8, $F6, $F2
C13			
C14	SD $F8,0($R3)	ADDUI R3,R3,4	
C15			
C16		BNE $R3,$R4, LOOP2	
C17		Br. delay slot	
C18		ADDUI $R1,$R1,4	
C19			
C20		BNE $R1,$R2, LOOP1	
C21		Br. delay slot	

3. How long is the critical path?

21 cycles

4. What performance did you achieve in CPIas?

CPIas = (# cycles) / IC = 21 / 16 = 1,31

5. What performance did you achieve in FP ops per cycles?

(# FP ops) / cycles = 3 / 21 = 0,14

6. How much is the code efficiency?

Code_eff = IC/ (# cycles * # issues) = 16 / (21 * 3) = 0,25

EXERCISE 2: DYNAMIC BRANCH PREDICTION (5 points)

Let’s consider the same assembly code used for **EXERCISE 1:**

```
INIT:  ADDUI $R1, $R0, 0
       ADDUI $R2, $R0, 40
       ADDUI $R4, $R0, 8

LOOP1: LD $F0, 0 ($R1)
       FADD $F4, $F0, $F2
       SD $F4, 0 ($R1)
       ADDUI $R3, $R0, 0

LOOP2: LD $F6, 0($R3)
       FADD $F8, $F6, $F2
       SD $F8, 0($R3)
       ADDUI $R3, $R3, 4
       BNE $R3, $R4, LOOP2

       ADDUI $R1, $R1, 4
       BNE $R1, $R2, LOOP1
```

1. *How many iterations for the outer loop **LOOP1?***

   ____

2. *How many iterations for the inner loop **LOOP2?***

   ____

3. *How many branch instructions are executed in the code?*

   ____

4. Assuming there is **no branch prediction** and each branch costs **2 cycle penalty** to fetch the correct instruction, *how many branch penalty cycles are needed to execute both loops?*

   ____

5. Assuming to execute the code on a pipelined processor with a dynamic **Branch Prediction Unit (BPU)** in the **IF-stage** composed of:
   - **2-entry 2-bit Branch History Table**
   - **2-entry Branch Target Buffer**

Let's assume the 2 branch instructions **do not collide** so they are allocated to the 2 entries of the BPU where the **BTB hit**, there are 4 cases for each conditional branch with the related **branch penalty cycles:**

	Branch Outcome	
	Taken	Not Taken
Strongly Taken	1 cycle	2 cycles
Weakly Taken	1 cycle	2 cycles
Strongly Not Taken	2 cycles	0
Weakly Not Taken	2 cycles	0

Let's assume the 2-entries do not collide with BTB hit and are initialized as **Strongly Taken**, *please complete the following table:*

Explain the branch behavior considering the inner LOOP2 in isolation.	***How many branch penalty cycles to execute the LOOP2 in isolation?***	***Calculate the branch misprediction rate to execute the LOOP2 in isolation.***
Explain the branch behavior considering both loops.	***How many branch penalty cycles to execute both loops?***	***Calculate the global branch misprediction rate to execute both loops.***

Feedback

1. *How many iterations for the outer loop* ***LOOP1?***
   The outer loop LOOP1 is executed **10** times.

2. *How many iterations for the inner loop* ***LOOP2?***
   The inner loop LOOP2 is executed 2 times for each iteration of LOOP1 => Globally LOOP2 is executed **20** times.

3. *How many branch instructions are executed in the code?*
   There are **20** branches for LOOP2 and **10** branches for BNE-LOOP1=> Globally **30** branches executed.

4. Assuming there is **no branch prediction and** each branch costs **2 cycle penalty** to fetch the correct instruction, *how many branch penalty cycles are needed to execute both loops?*
   Globally **30** branch instructions are executed introducing 2 cycles penalty each => Globally there are **60** branch penalty cycles to execute the code.

Let's assume the 2-entries do not collide with BTB hit and are initialized as **Strongly Taken**, *please complete the following table:*

Explain the branch behavior considering the inner LOOP2 in isolation.	***How many branch penalty cycles to execute the LOOP2 in isolation?***	***Calculate the branch misprediction rate to execute the LOOP 2 in isolation.***
Being the predictor initialized as **Strongly Taken,** the first iteration of LOOP2 is correctly predicted as taken, while there is a misprediction at the second iteration (exit) of the inner LOOP2 and the prediction is turned to **Weakly Taken.**	There are: (1 + 2) = 3 branch penalty cycles.	There is 1 misprediction out of 2 branch predictions => misprediction rate 50%.
Explain the branch behavior considering both loops.	***How many branch penalty cycles to execute both loops?***	***Calculate the global branch misprediction rate to execute both loops.***
As explained above, being the predictor initialized as **ST**, the first iteration of LOOP2 is correctly predicted as taken, while there is a misprediction at the second iteration (exit) of the inner LOOP2 and the prediction is turned to **WT.** Exiting from the inner LOOP2 with the prediction as **WT,** this does not influence the LOOP1 because the 2 branch instructions **do not collide.** The BHT entry used for LOOP1 is initialized as **ST,** so there are 9 iterations correctly predicted as taken, while we have a misprediction at the last iteration of the outer LOOP1 and its prediction is turned to WT. Exiting from the outer LOOP1 with the prediction as **WT,** this does not influence the LOOP2 because the 2 branch instructions **do not collide.** When re-entering in the inner LOOP2 with the its prediction as **WT,** the first iteration is taken, the prediction is turned to ST and we have a misprediction at the second iteration (exit) of the inner LOOP2 and the prediction bit is turned to **WT.**	There are: (1 + 2) = 3 BP cycles to execute LOOP2 for 10 iterations of the outer LOOP1 => **30** BP cycles. For the outer LOOP1, there are: (9 + 2) = **11** BP cycles. Globally there are **41** BP cycles.	We have 1 misprediction for the BNE-LOOP2 only at the exit of LOOP2 times 10 iterations of the outer LOOP1 => globally 10 mispredictions. For the outer LOOP1, we have only 1 mispredictions for BNE-LOOP1 at the last iteration of LOOP1 Globally (10 + 1) = 11 mispredictions, while there are 30 predictions (20 for BNE-LOOP2 and 10 for BNE-LOOP1) => 11 mispredictions out of 30 predictions => 36.67% misprediction rate.

EXERCISE 3 – SCOREBOARD (4 points)

1. *Let's consider the following assembly code containing multiple types of dependences. Complete the following table by inserting all types of data-dependences, anti-dependences and output dependences for each instruction:*

INSTRUCTION	ANALYSIS OF DEPENDECES
`I0: LD   $F2,A($R6)`	--
`I1: FADD $F3,$F2,$F6`	True data dependence with I0 for $F2
`I2: SD   $F3,A($R7)`	
`I3: LD   $F3,B($R6)`	
`I4: SD   $F3,C($R7)`	
`I5: ADDUI $R6,$R6,4`	
`I6: ADDUI $R7,$R7,4`	

2. *Schedule the code on a CPU with dynamic scheduling based on* ***OPTIMIZED SCOREBOARD*** *with the following assumptions:*

- 2 LOAD/STORE Units (LDU1, LDU2) with latency 3 cycles
- 1 FP Unit (FPU1) with latency 3 cycles
- 1 ALU/BR Unit ALU1 with latency 1 cycle
- Register Filw with 2 read ports and 1 write port
- Check for WAR and WAW hazards postponed to the WRITE BACK phase
- Forwarding

INSTRUCTION	ISSUE	READ OPs	EXEC COMPL.	WRITE BACK	Hazards Type Forwarding	UNIT
`I0: LD   $F2,A($R6)`	1	2	5	6		LDU1
`I1: FADD $F3,$F2,$F6`	2	6	9	10	RAW $f2 by forw.	FPU1
`I2: SD   $F3,A($R7)`						
`I3: LD   $F3,B($R6)`						
`I4: SD   $F3,C($R7)`						
`I5: ADDUI $R6,$R6,4`						
`I6: ADDUI $R7,$R7,4`						

3. *Express the formula and calculate the CPI:*
CPI = ____

Feedback

1. *Let's consider the following assembly code containing multiple types of dependences. Complete the following table by inserting all types of data-dependences, anti-dependences and output dependences for each instruction:*

INSTRUCTION	ANALYSIS OF DEPENDECES
`I0: LD $F2,A($R6)`	--
`I1: FADD $F3,$F2,$F6`	True data dependence with I0 for $f2
`I2: SD $F3,A($R7)`	True data dependence with I1 for $f3
`I3: LD $F3,B($R6)`	Output dependence with I1 for $f3 Anti dependence with I2 for $f3
`I4: SD $F3,C($R7)`	True data dependence with I3 for $f3
`I5: ADDUI $R6,$R6,4`	Anti dependence with I0, I3 for $r6
`I6: ADDUI $R7,$R7,4`	Anti dependence with I2, I4 for $r7

2. *Schedule the code on a CPU with dynamic scheduling based on OPTIMIZED SCOREBOARD with the following assumptions:*

- 2 LOAD/STORE Units (LDU1, LDU2) with latency 3 cycles
- 1 FP Unit (FPU1) with latency 3 cycles
- 1 ALU/BR Unit ALU1 with latency 1 cycle
- Register File with 2 read ports and 1 write port
- Check for WAR and WAW hazards postponed to the WRITE BACK phase
- Forwarding

INSTRUCTION	ISSUE	READ OPs	EXEC COMPL.	WRITE BACK	Hazards Type	UNIT
`I0: LD $F2,A($R6)`	1	2	5	6		LDU1
`I1: FADD $F3,$F2,$F6`	2	6	9	10	RAW $f2 I0 by forw	FPU1
`I2: SD $F3,A($R7)`	3	10	13	14	RAW $f3 I1 by forw	LDU2
`I3: LD $F3,B($R6)`	7	8	11	12	Check STRUCT LDU1 in ISSUE (Check WAW $f3 I1 in WB ok) (Check WAR Sf3 I2 in WB ok)	LDU1
`I4: SD $F3,C($R7)`	13	14	17	18	Check STRUCT LDU1 is ISSUE (Check RAW $f3 I3 ok)	LDU1
`I5: ADDUI $R6,$R6,4`	14	15	16	17	(Check WAR $r6 I3 in WB ok)	ALU1
`I6: ADDUI $R7,$R7,4`	18	19	20	21	Check STRUCT ALU1 in ISSUE (Check WAR $r7 I4 ok)	ALU1

3. *Calculate the CPI: CPI = (#clock cyles / IC) = 21/7 = 3*

QUIZ 4 – SPECULATIVE TOMASULO (1 point)

In the speculative Tomasulo architecture, exceptions are taken when the instruction that generated them reaches the head of the ROB.
(TRUE/FALSE ANSWER)
1 point

Answer: **TRUE** **FALSE**

Motivate your answer:

____

____

____

Feedback:
*The answer is **TRUE**: When the instruction that has generated the exceptions has reached the head of the ROB, it is ready to commit. This means that the instruction is no longer speculative and all its previous instructions have already been committed. This mechanism represents a precise interrupt/exception model.*

QUIZ 5 – CACHE PERFORMANCE (1 point)

Let us consider a computer architecture with L1 and L2 caches with the following parameters:

- Processor Clock Frequency = 1 GHz
- Hit Time L1 = 1 clock cycle
- Hit Rate L1 = 95%
- Hit Time L2 = 5 clock cycles
- Hit Rate L2 = 90%
- Miss Penalty L2 = 15 clock cycles

How much is the Global Miss Rate for Last Level Cache?
(SINGLE ANSWER)
1 point

Answer 1: 5% **T**

Answer 2: 0.5% **T** **(TRUE)**

Answer 3: 10% **T**

Answer 4: 7.5% **T**

Answer 5: 25% **T**

Motivate your answer:

Feedback:

Global Miss Rate = Miss Rate $_{L1\ L2}$ = Miss Rate$_{L1}$ x Miss Rate$_{L2}$= 0.05 x 0.1 = 0.005 **=** 0.5%

QUIZ 6: CACHE MEMORIES (3 points) OPTIONAL

Given a cache of a given capacity, associativity and block size, answer **TRUE** or **FALSE** to the following questions, ***motivating your answers***.

- Doubling the cache capacity of a direct mapped cache usually reduces conflict misses

  Answer: **TRUE** **(TRUE)** **FALSE**

- Doubling the block size reduces compulsory misses **TRUE** **(TRUE)** **FALSE**

- Change the nesting of loops in the code to access data in order stored in memory will increase spatial locality, possibly reducing the miss rate **TRUE** **(TRUE)** **FALSE**