Surname	
Name	
POLIMI Personal code	
Signature	

SOLUTION

Politecnico di Milano, 5 May 2025

Course on Advanced Computer Architectures

Prof. C. Silvano

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

EXERCISE 1.A - SOFTWARE PIPELINING (2 points)

Consider the following software pipelined loop **SP_LOOP** *and the corresponding start-up and finish-up code:*

```
START_UP:   LD F0, 0 (R1)
            LD F2, 0 (R2)
            FADD F4, F0, F0
            FADD F6, F2, F2
            LD F0, 8 (R1)
            LD F2, 8 (R2)

SP_LOOP:    SD F4, 0 (R1)
            SD F6, 0 (R2)
            FADD F4, F0, F0
            FADD F6, F2, F2
            LD F0, 16 (R1)
            LD F2, 16 (R2)
            ADDUI R1, R1, 8
            ADDUI R2, R2, 8
            ADD R3, R2, R1
            BNE R3, R4, SP_LOOP

FINISH-UP: SD F4, 8 (R1)
            SD F6, 8 (R2)
            FADD F4, F0, F0
            FADD F6, F2, F2
            SD F4, 16 (R1)
            SD F6, 16 (R2)
```

1. *Reconstruct the original (non-pipelined) version of the code:*

Feedback:

```
LOOP: LD F0, 0 (R1)
      LD F2, 0 (R2)
      FADD F4, F0, F0
      FADD F6, F2, F2
      SD F4, 0 (R1)
      SD F6, 0 (R2)
      ADDUI  R1, R1, 8
      ADDUI R2, R2, 8
      ADD R3, R2, R1
      BNE R3, R4, LOOP
```

EXERCISE 1.B - SOFTWARE PIPELINING (3 points)

Given the same software pipelined loop of ***Exercise 1.A:***

```
SP_LOOP: SD F4, 0 (R1)
         SD F6, 0 (R2)
         FADD F4, F0, F0
         FADD $F6, F2, F2
         LD F0, 16 (R1)
         LD F2, 16 (R2)
         ADDUI R1, R1, 8
         ADDUI R2, R2, 8
         ADD R3, R2, R1
         BNE R3, R4, SP_LOOP
```

Consider a **3-issue VLIW** machine with **fully pipelined functional units:**

- **1 Memory Units with 3 cycles latency**
- **1 FP ALUs with 3 cycles latency**
- **1 Integer ALU with 1 cycle latency to next Int/FP & 2 cycle latency to next Branch**

The branch is completed with 1 cycle delay slot (branch solved in ID stage). **No branch prediction.** In the Register File, it is possible to read and write at the same address at the same clock cycle.

1) *Considering one iteration of the* **`SP_LOOP`**, *complete the following table by using the* ***list-based scheduling*** (do NOT introduce any loop unrolling and modifications to loop indexes) on the *3-issue VLIW machine including the BRANCH DELAY SLOT.* Please do not write in NOPs.

	Memory Unit	Floating Point Unit	Integer Unit
C1	`SD F4, 0 (R1)`		
C2			
C3			
C4			
C5			
C6			
C7			
C8			
C9			
C10			
C11			
C12			
C13			
C14			
C15			

2. *How long is the critical path for a single iteration?*

   _____

3. *What performance did you achieve in CPI?*

   _____

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

   _____

5. *How much is the code efficiency?*

   _____

Feedback

	Memory Unit	Floating Point Unit	Integer Unit
C1	`SD F4, 0(R1)`	`FADD F4, F0, F0`	
C2	`SD F6, 0(R2)`	`FADD F6, F2, F2`	
C3	`LD F0, 16(R1)`		`ADDUI R1, R1, 8`
C4	`LD F2, 16(R2)`		`ADDUI R2, R2, 8`
C5			`ADD R3, R2, R1`
C6			
C7			`BNE R3, R4, SP_LOOP`
C8			`(br. delay slot)`

1. *How long is the critical path for a single iteration? (*)*

   8 cycles

   (*) When considering many iterations, the next iteration could start at cycle C8, therefore the critical path would be 7 cycles and the CPI_{AS} = 7 / 10 = 0.7

2. *What performance did you achieve in CPI?*

   CPI = # cycles / IC = 8 / 10 = 0.8

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

   FP ops per cycles = # FP ops / # cycles = 2 / 8 = 0.25

4. *How much is the code efficiency?*

   Code efficiency = IC / (# cycles x # issues) = 10 / (8 x 3) = 10 / 24 = 0.42

EXERCISE 2.A – DEPENDENCY ANALYSIS (2 points)

1. *Consider the same software pipelined loop of **Exercise 1.A** containing multiple types of intra-loop dependences. Complete the following table by inserting all types of true-data-dependences, anti-dependences and output dependences for each instruction:*

I#	TYPE OF INSTRUCTION	ANALYSIS OF DEPENDECES: 1. True data dependence with I# for $Fx 2. Anti-dependence with I# for $Fy 3. Output-dependence with I# for $Fz
I1	`SP_LOOP: SD F4, 0 (R1)`	None
I2	`SD F6, 0 (R2)`	None
I3	`FADD F4, F0, F0`	Anti dependence with **I1** for **F4**
I4	`FADD F6, F2, F2`	Anti dependence with **I2** for **F6**
I5	`LD F0, 16 (R1)`	Anti dependence with **I3** for **F0**
I6	`LD F2, 16 (R2)`	Anti dependence with **I4** for **F2**
I7	`ADDUI R1, R1, 8`	Anti dependence with **I1** and **I5** for **R1**
I8	`ADDUI R2, R2, 8`	Anti dependence with **I2** and **I6** for **R2**
I9	`ADD R3, R2, R1`	True data dependence with **I7** for **R1** True data dependence with **I8** for **R2**
I10	`BNE R3, R4, SP_LOOP`	True data dependence with **I9** for **R3**

EXERCISE 2.B – TOMASULO (3 points)

*Consider the same assembly code to be executed on a CPU with dynamic scheduling based on **TOMASULO algorithm** with all cache HITS, a single Common Data Bus and:*

- *2 RESERV. STATIONS **(RS1, RS2)** with 2 LOAD/STORE units **(LDU1, LDU2)** with latency **4***
- *2 RESERVATION STATION **(RS3, RS4)** with 1 FP unit1 **(FPU1)** with latency **4***
- *2 RESERVATION STATION **(RS5, RS6)** with 2 INT_ALU/BR units **(ALU1, ALU2)** with latency **2***

Please complete the following table:

INSTRUCTION	ISSUE	START EXEC	WRITE RESULT	Hazards Type	RSi	UNIT
`I1: SD F4, 0 (R1)`	1	2	6	None	RS1	LDU1
`I2: SD F6, 0 (R2)`	2	3	7	None	RS2	LDU2
`I3: FADD F4, F0, F0`	3	4	8	(WAR with I1 for F4 solved*)	RS3	FPU1
`I4: FADD F6, F2, F2`	4	**9**	13	**STR. FPU1** (WAR with I2 for F6 solved*)	RS4	FPU1
`I5: LD F0, 16 (R1)`	**7**	8	12	**STR. RS1** (WAR with I3 for F0 solved*)	RS1	LDU1
`I6: LD F2, 16 (R2)`	8	9	**14**	**STR. WR. CDB** (WAR with I4 for F2 solved*)	RS2	LDU2
`I7: ADDUI R1, R1, 8`	9	10	**15**	**STR. WR. CDB** (WAR with I5 for R1 solved*)	RS5	ALU1
`I8: ADDUI R2, R2, 8`	10	11	**16**	**STR. WR. CDB** (WAR with I6 for R2 solved*)	RS6	ALU2
`I9: ADD R3, R2, R1`	**16**	17	19	**STR. RS5** (RAW R1 & R2 solved)	RS5	ALU1
`I10: BNE R3,R4, SPLOOP`	17	**20**	22	**RAW with I9 for R3**	RS6	ALU2

(*) In Tomasulo, WAW and WAR hazards are already solved by Register Renaming in ISSUE stage.

(**) Only hazards that have caused the introduction of some stalls are reported in the table. Other dependencies are already solved.

*Calculate the **CPI** =* _____

CPI = # clock cycles / IC = 22 / 10 = 2.2

QUESTION 3: LOOP UNROLLING (3 points)

Let's consider the following loop code where registers **R1** and **R2** are initialized to **0** and **R3** is initialized to **400:**

```
LOOP:  LD F2, 0 (R1)
       LD F4, 0 (R2)
       FADD F6, F2, F4
       SD F6, 0 (R1)
       ADDUI R1, R1, 4
       ADDUI R2, R2, 4
       BNE R1, R3, LOOP
```

Answer to the following questions:

How many iterations of the loop?	100 iterations
How many instructions per iteration?	7 instructions
How many instructions due to the loop overhead per iteration?	3 out of 7 instructions
How many instructions are executed globally?	700 instructions
How many branch instructions are executed globally?	100 branches
Let's consider a loop unrolling version of the code with unrolling factor 2.	
How many iterations of the unrolled loop?	50 iterations
How many instructions per iteration?	(4 x 2) + 3 = 11 instructions
How many instructions due to the loop overhead per iteration?	3 out of 11 instructions
How many instructions are executed globally?	550 instructions
How many branch instructions are executed globally?	50 branches
How many more registers are needed due to register renaming?	3 more FP registers
What are the registers that need to be renamed?	F2, F4 and F6

How much is the global instruction count decrease?	(700 – 550) / 700 = 21%
Let's consider a loop unrolling version of the code with unrolling factor 4.	
How many iterations of the unrolled loop?	25 iterations
How many instructions per iteration?	(4 x 4) + 3 = 19 instructions
How many instructions due to the loop overhead per iteration?	3 out of 19 instructions
How many instructions are executed globally?	475 instructions
How many branches are executed?	25 branches
How many more registers are needed due to register renaming?	9 more FP registers
What are the registers that need to be renamed?	F2, F4 and F6 need to be renamed three times
How much is the global instruction count decrease?	(700 – 475) / 700 = 32%
What is the best unrolling factor between 2 and 4? Explain why.	

QUIZ 4 – BRANCH PREDICTION (1 point)

During the execution of a program, the number of mispredictions depends on the initialization and the size of the Branch History Table.

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

Motivate your answer:

_____

_____

_____

_____

_____

_____

QUIZ 5 – BRANCH PREDICTION (1 point)

Consider the following assembly code executed by a processor with 1-entry 1-bit Branch History Table initialized as Taken:

```
            ADDI R1, R0, 0
            ADDI R2, R0, 100

LOOP:       BEQ R1, R2, DONE
            ADD R5, R5, R4
            ADDI R1, R1, 1
            J   LOOP
DONE:
```

*Assuming **only** the branch instruction accesses the BHT:*

- *How many accesses to the Branch History Table?*
- *How many mispredictions?*

(SINGLE ANSWER)

Answer 1: 101 accesses | 2 mispredictions **(TRUE)**

Answer 2: 100 accesses | 1 misprediction

Answer 3: 99 accesses | 1 misprediction

Answer 4: 201 accesses | 2 mispredictions

Motivate your answer:

_____

_____

_____

_____

_____

_____

QUIZ 6: SUPERSCALAR PROCESSORS (3 points) OPTIONAL

Considering superscalar processors, answer **TRUE** or **FALSE** to the following questions, ***motivating your answers***.

- Doubling the number of issues reduces the CPI metric.

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

- Out-of-order execution can be generated by data cache misses **TRUE (TRUE)** **FALSE**

- The introduction of dynamic branch predictors improves the CPI metric **TRUE (TRUE)** **FALSE**