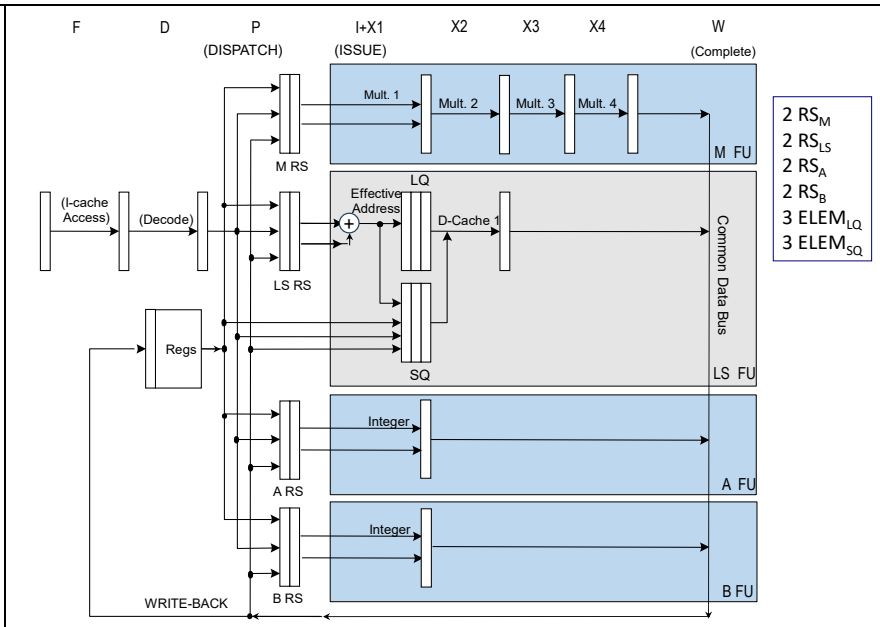
HIGH PERFORMANCE COMPUTER ARCHITECTURE midterm exam 30-10-2019 **MATR.NO.**_____

SURNAME_____

(REVISED 26-10-2023) **FIRST NAME**_____

1) (POINTS 27/30) Consider a **triple-dispatch (3 instruction per cycle)** processor using Tomasulo's algorithm to perform the dynamic scheduling of instructions on the pipeline shown in the following figure. This pipeline is executing the following program, which performs a search within a vector (initially, R1=0).

```
etic: LW   R2, 0(R1)     ; read Xi
      MULI R2, R2, 3     ; multiplies Xi by 3
      SW   R2, 0(R1)     ; write Xi
      ADDI R1, R1, 4     ; update  R1
      BNE  R2, R0, etic  ; continue to loop if
false
```



Working hypothesis:

- the loop executes speculatively in terms of direction (always taken) and regarding the branch condition; high-performance fetch breaks after fetching a branch
- the issue stage (I) calculates the address of the actual read/write and push it into load/store queues; only 1 instruction is issued per cycle
- **reads require 5 clock cycles**; writes take 1 cycle
- when accessing memory (M), **reads** have precedence over writes and must be executed in-order
- **there is a single CDB**
- dispatch stage (P) and complete stage (W) require 1 clock cycle
- **ASSUME** that the reservation stations could be freed right before the start of issue phase (therefore extending the duration of P stage)
- only 1 instruction is committed (C stage) per cycle
- there are separated integer units: one for the calculation of the actual address, one for arithmetic and logical operations, one of the integer multiplication and one for the evaluation of the branch condition, as illustrated in this table:

Type of Functional Unit	No. of Functional Units	Cycles for stage I+X	No. of reservation stations
LS: Integer (effective addr.)	1	1	2
A: Integer (op. A-L)	1	1	2
B: Integer (branch calc.)	1	1	2
M: Integer Multiplication	1	**4**	2

- the functional units TAKE advantage of pipelining techniques internally
- the load queue has 3 slots; the store queue has 3 slots (writes wait for the operand in the store queue, i.e., in the execution stage)

Complete the following chart until the end of the FOURTH iteration of the above code fragment in the case of dynamic scheduling with speculation. Also add the instruction that occupies a certain reservation station (one of the 8) as indicated:

Instr. No..	Instruction name	ALU RS1	ALU RS2	LS RS1	LS RS2	BU RS1	BU RS2	MU RS1	MU RS2	P: disPatch (clock)	I+X:Issue+Exec (start-stop)	M: MEM.ACCESS (start-stop)	W: CDB-write (clock)	C: Commit (clock)	Comments
I01	**LW R2,0(R1)**			I01 1-1						1	2-2	3-7	8	9	
...	...														
...	...														

1) (POINTS 5/30) On a Linux system, write the SINGLE command line to perform at the BASH shell prompt the following operation (please note that no intermediate files should be used):

- The file 'data1.txt' contains a list of alpha-numerical values to be used as input
- The file 'data2.txt' should contain a list of the lines which contain the string with "ciao"
- The extracted list should also be directed to the printer

HIGH PERFORMANCE COMPUTER ARCHITECTURE midterm exam 30-10-2019 – SOLUTION (REVISED 26-10-2023)

EXERCIZE 1

Instr. No..	Instruction name	ALU RS1 (start-stop)	ALU RS2 (start-stop)	LS RS1 (start-stop)	LS RS2 (start-stop)	BU RS1 (start-stop)	BU RS2 (start-stop)	MU RS (start-stop)1	MU RS2 (start-stop)	P: Dispatch (clock)	I+X: Issue (start-stop)	MEM. ACC. (start-stop)	W: CDB-write (clock)	C: Commit (clock)	Comments
I01	LW R2,0(R1)			I01 1-1						1	2-2	3-7	8	9	
I02	MULI R2,R2,3							I02 1-8		1	9-12	--	13	14	I waits R2 from 1/LW
I03	SW R2,0(R1)				I03 1-2					1	3-3	23	--	24	I waits issue logic; M waits R2 M waits mem
I04	ADDI R1,R1,4	I04 2-3								2	4-4	--	5	25	I waits issue logic;
I05	BNE R2,R0,etic					I05 2-13				2	14-14	--	--	26	I waits R2 from 1/MULI
I06	LW R2,0(R1)	s		I06 3-5						3	6-6	8-12	14	27	I waits R1; M waits mem; W waits for CDB
I07	MULI R2,R2,3								I07 3-14	3	15-18	--	19	28	I waits R2 from 2/LW;
I08	SW R2,0(R1)				I08 3-6					3	7-7	24	--	29	I waits R1; I waits issue logic; M waits R2; M waits mem
I09	ADDI R1,R1,4	I09 4-7								4	8-8	--	9	30	I waits R1 from 1/ADDI; I waits issue logic;
I10	BNE R2,R0,etic						I10 4-19			4	20-20	--	--	31	I waits R2 from 2/MULI;
I11	LW R2,0(R1)			I11 6-9						6	10-10	13-17	18	32	P waits EA-RSs I waits issue logic; I waits R1; M waits mem
I12	MULI R2,R2,3							I12 9-18		9	19-22	--	23	33	P waits M-RSs; I waits R2 from 3/LW
I13	SW R2,0(R1)				I13 9-10					9	11-11	25	--	34	I waits R1; I waits issue logic; M waits R2; M waits mem
I14	ADDI R1,R1,4	I14 9-11								9	12-12	--	15	35	I waits issue logic;
I15	BNE R2,R0,etic					I15 14-23				14	24-24	--	--	36	P waits B-RSs; I waits R2 from 3/MULI
I16	LW R2,0(R1)			I16 15-15						15	16-16	18-22	24	37	I waits R1; M waits mem; W waits for CDB
I17	MULI R2,R2,3								I17 15-24	15	25-28	--	29	38	I waits R2 from 4/LW
I18	SW R2,0(R1)				I18 15-25					15	26-26	30	--	39	I waits R1; I waits issue logic; M waits R2; M waits mem*;
I19	ADDI R1,R1,4	I19 16-16								16	17-17	--	20	40	W waits for CDB
I20	BNE R2,R0,etic						I20 20-29			20	30-30		--	41	P waits B-RSs; I waits R2 from 4/MULI

* I18 (an SW) has to wait to issue until there is space in the SQ (there are 3 slots and they are all occupied by the previous stores until cycle 24), then at 25 and 26 previous instructions are issueing: it issues at 26.

EXERCIZE 2

The requested command line is:

```
grep "ciao" data1.txt  |  tee data2.tx  |  lpr
```