HIGH PERFORMANCE COMPUTER ARCHITECTURE 05-07-2006 **MATRICULATION NO.**_____

REVISED 23-10-2023 **SURNAME**_____

FIRST NAME_____

1) (POINTS 14/40) Consider the following snippet of code running on a single-dispatch using Tomasulo's algorithm to perform the dynamic scheduling of instructions. The program performs a search inside a vector. Initially, R1 = 0.

```
etic:   LW      R2, 0(R1)       ; read Xi
        ADDI    R2, R2, 1       ; increment R1
        SW      R2, 0(R1)       ; store Yi
        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) but not regarding the branch condition; case A) no-speculation on branch condition; case B) speculation on branch condition;
- the issue stage (I) calculates the address of the actual reads and writes;
- reads require 1 clock cycles; writes take 0 cycles (they are written in a write-buffer + split-cache)
- when accessing memory (M), writes have precedence over reads and must be executed in-order
- there's only one CDB
- dispatch stage (P) and complete stage (W) require 1 clock cycle
- there are separated integer units for the calculation of the actual address, for arithmetic and logical operations, for the evaluation of the branch condition
- the functional units do not take advantage of pipelining techniques internally (reservation stations are busy until the end of CDB-write, except for Stores)
- the load queue has 5 slots; the queue buffer has 5 slots (writes wait for the operand in the queue buffer, i.e., in the execution stage)
- the rest of the processor and has the following characteristics

Type of Functional Unit	No. of Functional Units	Cycles for stage I	No. of reservation stations
Integer (effective addr.)	1	1	2
Integer (op. A-L)	1	1	2
Integer (branch calc.)	1	1	2

Complete the following chart until the end of the third iteration of the code fragment above, both in the case of no speculation on branch condition (case A) that in the case of speculation on branch condition (case B).

Iteraz.	Instruction	P: Dispatch (clock)	I+X: Issue+Exec (start-stop)	M: MEM ACCESS (clock)	W: CDB-write (clock)	C: Complete (clock)	Commenti
1	`LD R2,0(R1)`						
...	...						
...	...						

2) Given the sequence P1: R, P2: R, P3: R, P1: W, P2: W, P3: W (Px:R = read by the processor Px, Px:W write by the processor Px), with respect to a certain variable 'a ', show for each processor the sequence of states, and transactions on the bus that occur in a multiprocessor UMA with write-back caches for each processor and PSCR coherence protocol.

SOLUTION 05-07-2006
(REVISED 23-10-2023)

EXERCIZE 1

CASE A (no speculation on branch condition: dispatch WAITS for branch condition verification):

Iter.	Instruction	P: Dispatch (start-stop)	I+X: Issue+Exec (start-stop)	M: MEM ACCESS (clock)	W: CDB-write (clock)	C: Commit (clock)	Comments
1	`LW R2,0(R1)`	1-4	2	3	4	5	
1	`ADDI R2, R2, 1`	2-6	5		6	7	I waits R2 from 1/LW
1	`SW R2, 0(R1)`	3-3	4-6	7		8	M waits R2 from 1/ADDI_R2
1	`ADDI R1, R1, 4`	4-7	6		7	9	I waits ALU-FU available
1	`BNE R2, R0, etic`	5-7	7			10	I waits R2 from 1/ADDI_R2
2	`LW R2,0(R1)`	8-11	9	10	11	12	P waits decision su 1/BNE, I waits R1 from 1/ADD_R1
2	`ADDI R2, R2, 1`	9-13	12		13	14	I waits R2 from 2/LW
2	`SW R2, 0(R1)`	10-10	11-13	14		15	M waits R2 from 2/ADDI_R2
2	`ADDI R1, R1, 4`	11-14	13		14	16	I waits ALU-FU avail
2	`BNE R2, R0, etic`	12-14	14			17	I waits R2 from 2/ADDI_R2
3	`LW R2,0(R1)`	15-18	16	17	18	19	P waits decision su 2/BNE
3	`ADDI R2, R2, 1`	16-19	19		20	21	I waits R2 from 3/LW; W waits for CDB
3	`SW R2, 0(R1)`	17-17	18-19	21		22	M waits R2 from 3/ADDI_R2
3	`ADDI R1, R1, 4`	18-20	19		20	23	
3	`BNE R2, R0, etic`	19-20	21			24	I waits R2 from 3/ADDI_R2

Note: the dispatch of 2/LW can happen at cycle 6 since there are sufficient Reservation Stations; however at cycle 7 the 2/LW cannot issue since it has to wait the resolution of the condition of the previous branch. Even if there is no speculation, the subsequent instruction enter in the pipeline and will eventually be discarded once the 1/BNE is resolved.

CASE B (speculation: dispatch DOES NOT WAIT for branch condition verification):

Iter.	Instruction	P: Dispatch (start-stop)	I+X: Issue+Exec (start-stop)	M: MEM ACCESS (clock)	W: CDB-write (clock)	C: Commit (clock)	Comments
1	`LW R2,0(R1)`	1-4	2	3	4	5	
1	`ADDI R2, R2, 1`	2-6	5		6	7	I waits R2 from 1/LW
1	`SW R2, 0(R1)`	3-3	4-6	7		8	M waits R2 from 1/ADDI_R2
1	`ADDI R1, R1, 4`	4-7	6		7	9	I waits ALU-FU available
1	`BNE R2, R0, etic`	5-7	7			10	I waits R2 from 1/ADDI_R2
2	`LW R2,0(R1)`	6-10	8	9	10	11	I waits R1 from 1/ADD_R1
2	`ADDI R2, R2, 1`	7-12	11		12	13	I waits R2 from 2/LW & ALU-FU avail.
2	`SW R2, 0(R1)`	8-8	9-12	13		14	M waits R2 from 2/ADDI_R2
2	`ADDI R1, R1, 4`	9-11	10		11	15	
2	`BNE R2, R0, etic`	10-13	13			16	I waits R2 from 2/ADDI_R2
3	`LW R2,0(R1)`	11-15	12	14	15	17	M waits mem
3	`ADDI R2, R2, 1`	12-17	16		17	18	I waits R2 from 3/LW
3	`SW R2, 0(R1)`	13-13	14-17	18		19	M waits R2 from 3/ADDI_R2
3	`ADDI R1, R1, 4`	14-18	15		16	20	
3	`BNE R2, R0, etic`	15-18	18			21	I waits R2 from 3/ADDI_R2

EXERCIZE 2)

P-BLOCKS

CPU-Action	P1	P2	P3	Bus-Action
P1:R	PC	-	-	BusRd(L2=off)
P2:R	I	PC	-	BusRd(L2=off)
P3:R	I	I	PC	BusRd(L2=off)
P1:W	PD	I	I	BusRd(L2=off)
P2:W	I	PD	I	BusRd(L2=on)
P3:W	I	I	PD	BusRd(L2=on)

S-BLOCKS

CPU-Action	P1	P2	P3	Bus-Action
P1:R	PC	-	-	BusRd(L2=off)
P2:R	SC	SC	-	BusRd(L2=on)
P3:R	SC	SC	SC	BusRd(L2=on)
P1:W	SC	SC	SC	BusUpd(L2=on)
P2:W	SC	SC	SC	BusUpd(L2=on)
P3:W	SC	SC	SC	BusUpd(L2=on)