Department of Electrical and Computer Engineering
The University of Texas at Austin

EE 460N Fall 2022
Instructor: Yale N. Patt
TAs: Kayvan Mansoorshahi, Michael Chen
Exam 1
October 12, 2022

Name:_____

Problem 1 (30 points):_____

Problem 2 (10 points):_____

Problem 3 (30 points):_____

Problem 4 (30 points):_____

Total (100 points):_____

Note: Please be sure that your answers to all questions (and all supporting work that is required) are contained in the space provided.

Note: Please be sure your name is recorded on each sheet of the exam.

Please read the following sentence, and if you agree, sign where requested:
I have not given nor received any unauthorized help on this exam.

Signature:_____

GOOD LUCK!

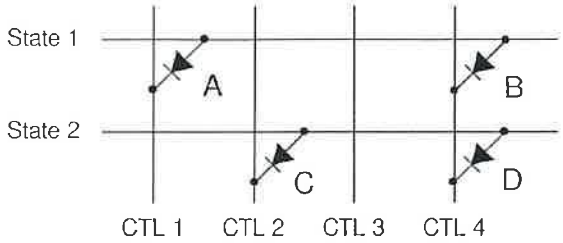
Name:_____

Problem 1 (30 points): Note: For each of the six answers below, if you leave the box empty, you will receive one point of the five.

Part a (5 points): The x86 ISA is a 2-address machine. What inconvenience does that introduce? How do we handle it?

The inconvenience is that one of the two sources was clobbered by the operate instruction. The way we handle it (if we still need to later use that source operand) is to write it to a different register before we execute the operate instruction

Part b (5 points): The diagram below, taken from Professor Maurice Wilkes diode matrix, shows two states State 1 and State 2, four control signals CTL1, CTL2, CTL3, and CTL4, and four diodes labeled A, B, C, D.



Is the paragraph below true or false? Explain.

The control signals asserted by State 2 are CTL1, CTL2, and CTL4. CTL1 is asserted by the voltage on State 2 as follows: State 2 is connected via diode D to CTL4, the voltage on CTL4 connected via diode B to State 1, and the voltage on State 1 connected via diode A to CTL1.

False. The voltage on CTL4 can not drive State 1 via diode B because that would require running diode B in the opposite direction.

Name:_____

Part c (5 points): Predication has an advantage over branch prediction in that predication will never suffer from mis-prediction penalty. However, there are two good reasons why predication may be worse than branch prediction. Name one of them and explain why it is worse.

1. If the merge point is distant, too many useless instructions would have to be brought into on-chip storage. Waste of excessive energy and waste of on-chip storage.
2. If the branch predictor is yielding accurate predictions, predication would have to wait for the predicate to be computed. Branch prediction would not have to wait.

Part d (5 points): Is the branch predictor part of the ISA or part of the microarchitecture? Explain.

Part of the microarchitecture. Nowhere in the ISA is branch prediction specified. It is "underneath the hood."

Part e (5 points): What does the addition of a Reorder Buffer (ROB) prevent from occurring during processing? Please be specific.

The ROB prevents out-of-order retirement from occuring, allowing precise exceptions to be maintained.

Part f (5 points): An out-of-order pipelined microarchitecture starts processing instructions in program order, then changes to out-of-order, and then changes a second time, to finish processing in-order. The microarchitecture requires a structure at each of the two transitions.

From in-order to out-of-order, what is the structure?

Reservation Station

From out-of-order back to in-order, what is the structure?

Reorder Buffer

Name:_____

Problem 2 (10 points): We wish to add a gshare predictor to the LC-3b. The branch history register will contain the directions of the last 3 branches. That means we need 3 bits of the PC. We will use bits 8, 5, and 2. During the execution of a program, we enter the loop shown below which is executed a large number of times. All branch instructions are explicitly identified.

```
LOOP    ...
        ...
    A   BRz  R ; branch is never taken
        ...
        ...
        ...
    B   BRp  W
        ...
        ...
    W   ...
    C   BRnp Z ; branch is always taken
        ...
    Z   ...
        ...
    D   BRn  Y
        ...
    Y   ...
        ...
    E   BRz  LOOP
```

- When the loop is entered, the branch history register contains 101.
- The branches at A and D are always predicted not taken.
- The branches at B and C are always predicted taken.
- Prediction accuracy is 100%.
- The branch at E is accurately predicted taken 200 times, after which it is not taken and we exit the loop.
- Bits 8, 5, and 2 of addresses A and B are all 0.
- Bits 8, 5, and 2 of addresses C, D and E are all 1.

Part a (1 points): What is in the branch history register just before A is fetched? 101

Part b (1 points): What is in the branch history register just before C is fetched? 101

Part c (2 points): What is the index into the PHT when the branch at A is fetched? 101

Part d (2 points): What is the state of A's saturating 2-bit counter when we exit the loop? 00

Part e (2 points): What is the index into the PHT when the branch at C is fetched? 010

Part f (2 points): What is the state of C's saturating 2-bit counter when we exit the loop? 11

Name:_____

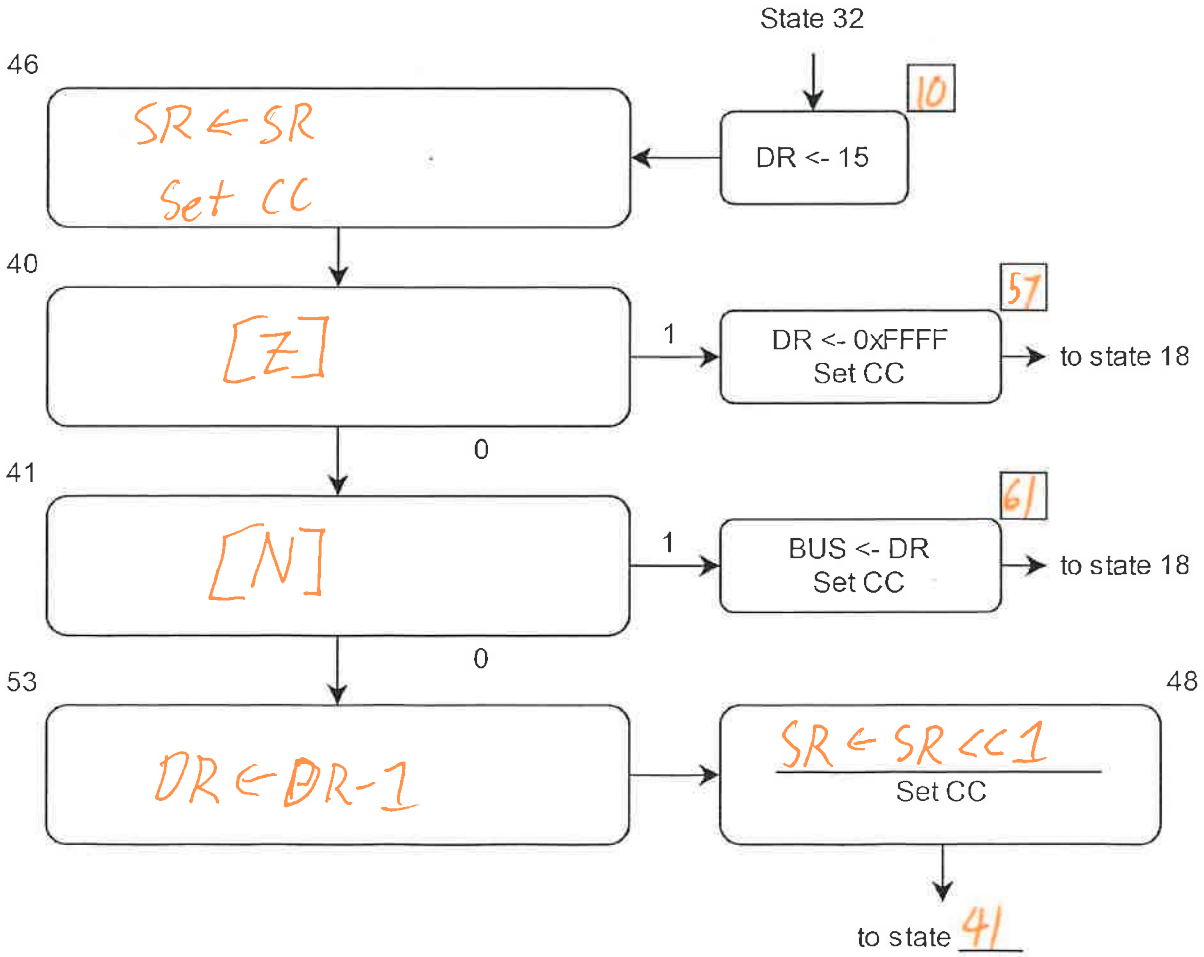
Problem 3 (30 points): In class we discussed several interesting instructions which are part of various ISAs. One was FFS (Find First Set), an instruction in the AMD 29000 ISA. FFS examines the 16 bit value in a register, finds the first bit set to "1" starting from the most significant bit and moving toward the least significant bit. The condition codes of FFS are set based on the result in DR.

If SR contains 0010010001100101, FFS sets DR to decimal 13, P=1, and N=Z=0.
If SR contains 0000000000000001, FFS sets DR to decimal 0, Z=1, and N=P=0.
If SR contains all 0s, FFS sets DR to 0xFFFF, N=1, and P=Z=0.

Your job: Augment the LC-3b to implement FFS. Use `1010` as the opcode. The FFS instruction has the following format.

Note: SR and DR must be distinct registers. Execution of FFS may destroy the original contents of SR.

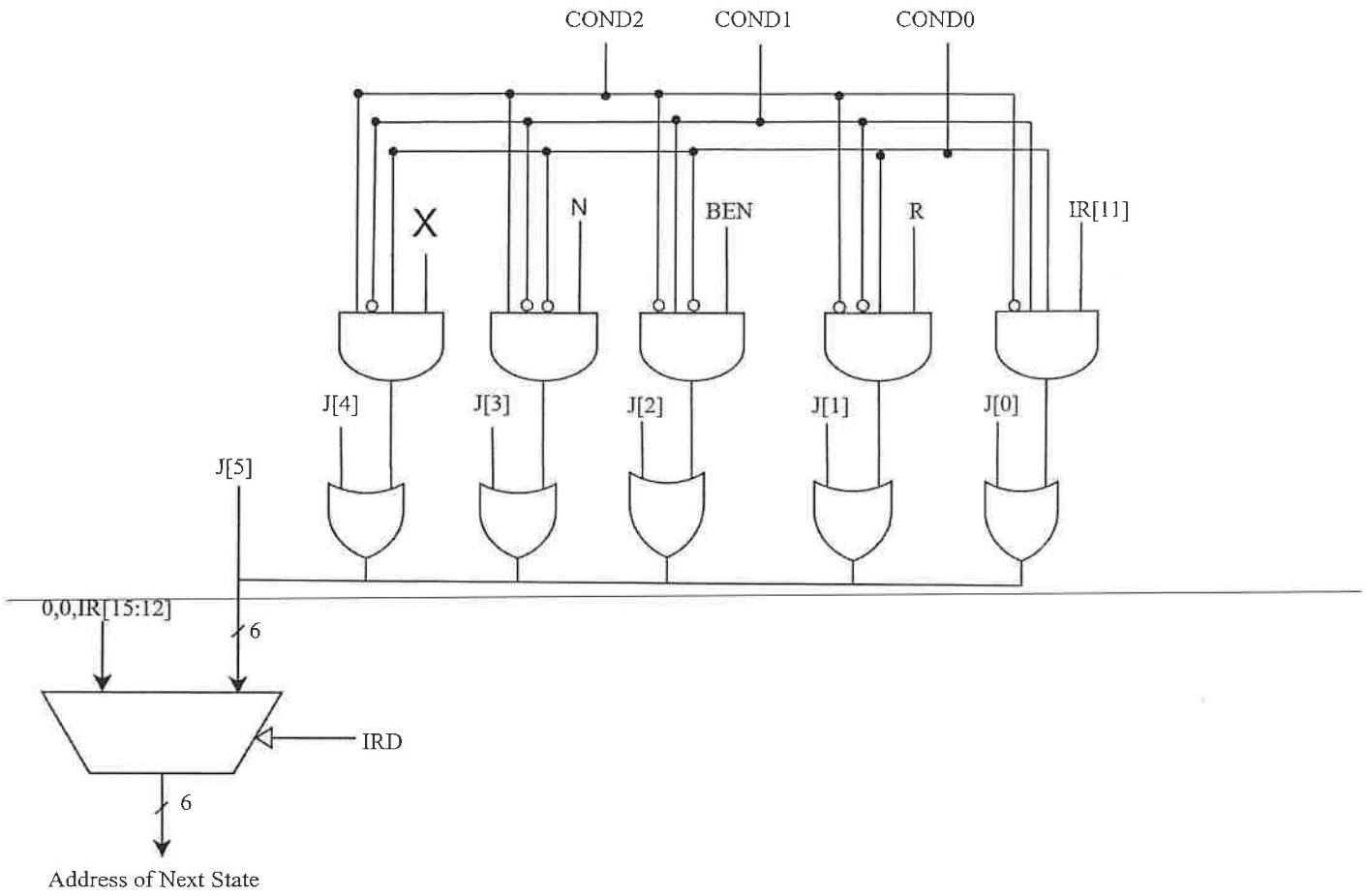
Part a (14 points) The State Machine: Complete the state machine shown below. That means describing what happens in each state, and identifying the state numbers in the boxes shown for those states that are missing state numbers.



PROBLEM CONTINUES ON NEXT PAGE

Name:_____

The Microsequencer: X and N have been added to the microsequencer.



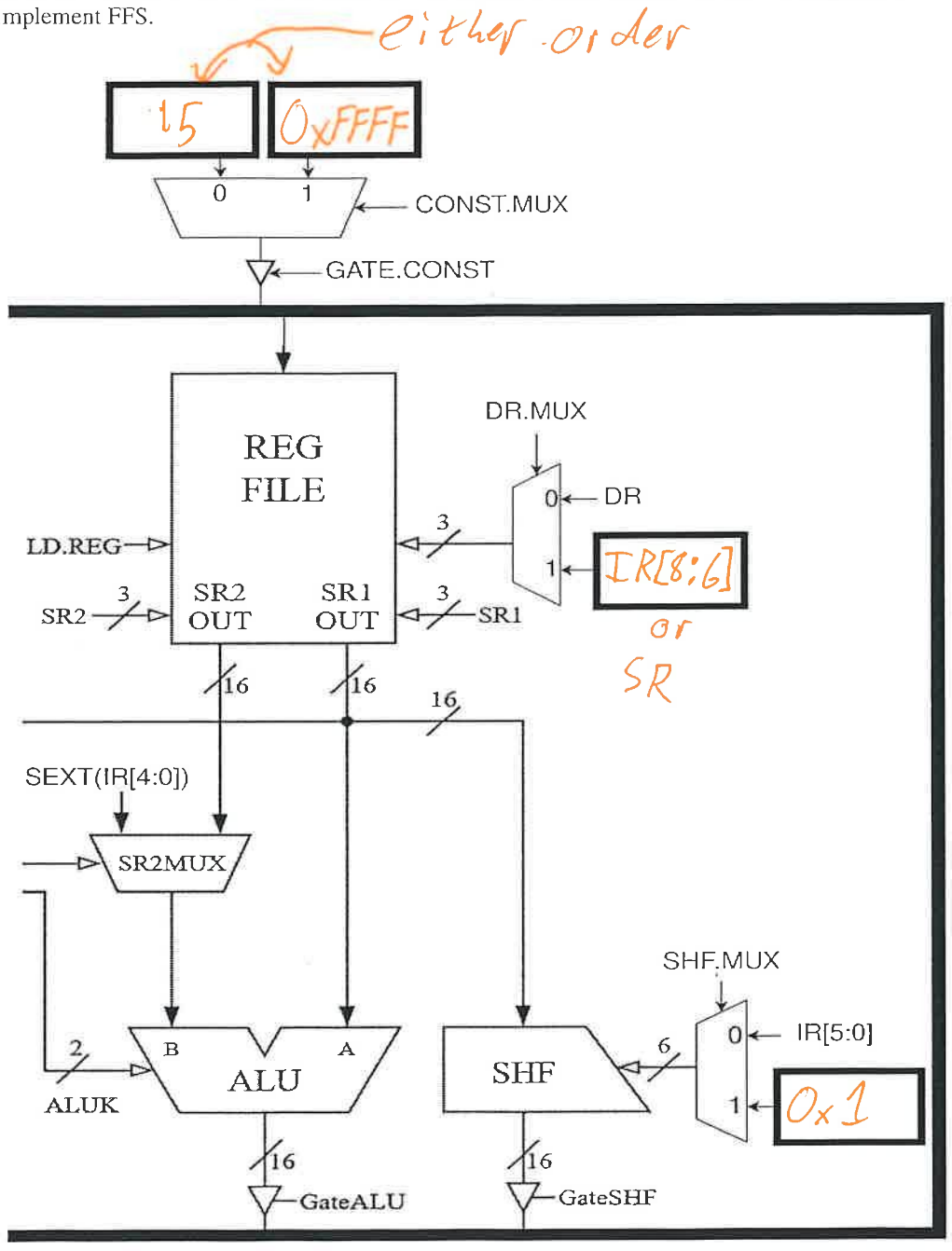
Part b (3 points): What is "X"?

Z

PROBLEM CONTINUES ON NEXT PAGE

Name:_____

Part c (7 points) The Data Path and its control: We have provided "boxes" for you to augment the data path as needed to implement FFS.



Part d (6 points): In the table below, each row represents a state, and each column the value of that control signal in that state. Provide the values for the control signals for states 41, 53, and 48. If the value of a control signal does not matter in a state, label that entry as X.

State	COND	ALU.k	LD.REG	CONST.MUX	SHF.MUX	DR.MUX
41	100	X	0	X	X	X
53	0	00	1	X	X	0
48	0	X	1	X	1	1

Name:_____

Problem 4 (30 points): Two Aggies built an out-of-order machine, as defined by Tomasulo; i.e., they did not include a reorder buffer (ROB).

The machine has a five stage pipeline (FETCH, DECODE, REGISTER RENAME, EXECUTE, WRITEBACK). All stages except EXECUTE take one clock cycle each.

- ADD takes one cycle for EXECUTE.
- MUL takes two cycles for EXECUTE.
- DIV takes eight cycles for EXECUTE.
- For DIV, SR1 is the dividend and SR2 is the divisor. (DR = SR1/SR2)
- Assume no functional units are pipelined, BUT you have enough of each to guarantee that an instruction never has to wait to start execution.
- All three functional units have three-entry reservation stations that are initially empty, and are allocated in a top-to-bottom manner. Assume SR1 will populate the left entry and SR2 the right.
- Entries are put into reservation stations at the end of REGISTER RENAME and removed at the end of WRITE-BACK.
- The result of a functional unit is broadcast during WRITEBACK. In the same cycle, instructions in the RS that need this result will set the valid bit for that operand so that they can start executing in the next cycle if both operands are ready.
- The write-back bus supports only one result being stored at a time. Earlier instructions take priority for WRITE-BACK.

The machine supports two types of exceptions, both are detected during the WRITEBACK stage.

1. **Divide-by-zero**
2. **Multiply overflow** - when a MUL produces a result not representable with 16 bits.

Part a (24 points). The table below contains a sequence of 7 instructions to be executed as a program.

	OP	DR	SR1	SR2
I1	Div	R7	R3	R1
I2	ADD	R1	R1	R4
I3	MUL	R2	R0	R5
I4	ADD	R3	R2	R1
I5	Div	R0	R4	R3
I6	Mul	R6	R6	R6
I7	MUL	R2	R6	R6

The instructions are not fully specified. To help you fill in the table, the following information is provided: the contents of the register file before execution starts, partial contents of the register file at the end of cycle 8 and at the end of cycle 13, and partial contents of the reservation stations at the end of cycle 8 and at the end of cycle 13.

Your job: Complete the table of instructions, and fill in the **bold spaces** of the register file and the reservation stations at the end of cycle 8 and cycle 13 shown on the next page and the timing diagram on the page after. Note that none of the boxes should contain a dash.

PROBLEM CONTINUES ON NEXT PAGE

Name:_____

	V	Tag	Val
R0	1	-	-2
R1	1	-	4
R2	1	-	51
R3	1	-	36
R4	1	-	2
R5	1	-	3
R6	1	-	16
R7	1	-	50

Initial

	V	Tag	Val
R0	0	σ	-2
R1	1	α	
R2	1	ϕ	-6
R3			
R4	1	-	2
R5	1	-	3
R6	0		
R7	0	ρ	50

End of cycle 8

	V	Tag	Val
R0			
R1	1	α	
R2	0	ψ	
R3	1	β	0
R4	1	-	2
R5	1	-	3
R6	1	θ	256
R7			9

End of cycle 13

ADD

	V	T	Val	V	T	Val
α	1		4	1		2
β	1	ϕ		1	α	6
γ	-	-	-	-	-	-

End of cycle 8

ADD

	V	T	Val	V	T	Val
α	1		4	1		2
β	1	ϕ	-6	1	α	6
γ	-	-	-	-	-	-

End of cycle 13

MUL

	V	T	Val	V	T	Val
ϕ						
θ						
ψ	-	-	-	-	-	-

End of cycle 8

MUL

	V	T	Val	V	T	Val
ϕ						
θ						
ψ						

End of cycle 13

DIV

	V	T	Val	V	T	Val
ρ				1		
σ			2		β	
τ	-	-	-	-	-	-

End of cycle 8

DIV

	V	T	Val	V	T	Val
ρ				1		
σ			2		β	
τ	-	-	-	-	-	-

End of cycle 13

PROBLEM CONTINUES ON NEXT PAGE

Name:_____

A blank cycle-by-cycle description of each instruction's behavior is provided for you to use in completing the above entries.

1. Use **F** for FETCH, **D** for DECODE, **RR** for REGISTER RENAME, Register number **(R#)** for WRITEBACK, and * if no progress is made during that clock cycle for that instruction.
2. Use **a** for ADD, **m** for MUL, and **d** for DIV.

Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
I1	F	D	RR	d	d	d	d	d	d	d	d	R7								
I2		F	D	RR	a	R1														
I3			F	D	RR	m	m	R2												
I4				F	D	RR	*	*	a	R3										
I5					F	D	RR	*	*	*	d	d	d	d	d	d	d	d	R0	
I6						F	D	RR	M	m	R6									
I7							F	D	RR	*	*	m	m	R2						

Part b (3 points). Does the program generate an exception? If so, what exception and at what clock cycle is it detected?

yes. Cycle 14 multiply-overflow.

Part c (3 points). What happens if we add a reorder buffer? Does the program generate an exception? If so, what exception and at what clock cycle is it detected?

yes. Cycle 19 divide-by-0.

(1) 2 ADD ~~3~~ MUL ~~2~~ DIV

(2) I7 given

(3) I4 must be β.
- R1 is valid and tagged α in cycle 8 but I4 doesn't exec until 9. Thus there is a younger add

(4) I5 must be div #2
- has β in R5. so must be > I4
- I7 is given
- I6 latency is too short.

(5) I6 must be a MUL
- Can't be div (latency)
- Can't be add (only 2 adds total and I4 is the second add)

(6) I1-I3 = 1 ADD, MUL & DIV.

(7) I2 must be add. DR = R1
- Can't be I1 since R3 is 36 (First instr)
- Can't be I3 since R5 is 3 (Never written)
- RAT R1 tagged α

(8) I1 must be Div R7, ...

- MUL writes R2 (RAT tag)
- R7 is written to by cycle 14 (RAT value)
- all other instruction DRs are resolved.

why R7.

- Must be I1 to WB value by cycle 14

why I1.

(9) I3 must be mul R2, ...

- 1st mul writes R2.
- all other instrs resolved.

(10) I1 op2 must be R1

- op1 is 36.
- R7 gets 9.
- R1 is the only 4.

(11) I4 op1, op2 = R2, R1

- dep stall (timing diagram)
- R1 (RAT α tag)
- R2 (mul dep stall resolved)

(12) I5 op1, op2 = R4, R3

- R3, β tag
- R4, since val=2 and no other 2s possible.