Computer Architecture

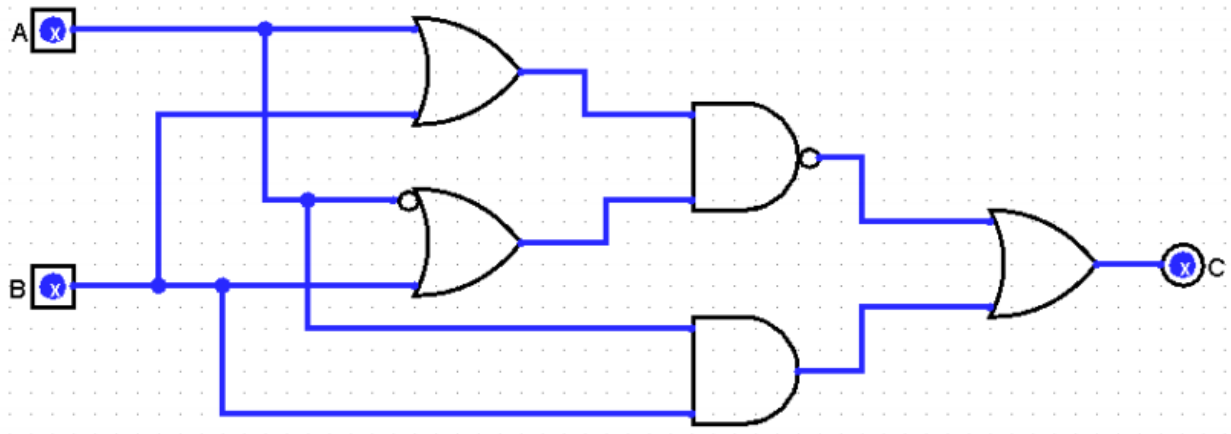
Discussion 12

CB

SID: _____

M2-1: *I couldn't come up with a clever title for SDS.* (10 points)

a) Give the simplest Boolean expression for the following circuit in terms of A and B, using the minimum number of AND, OR, and NOT gates:



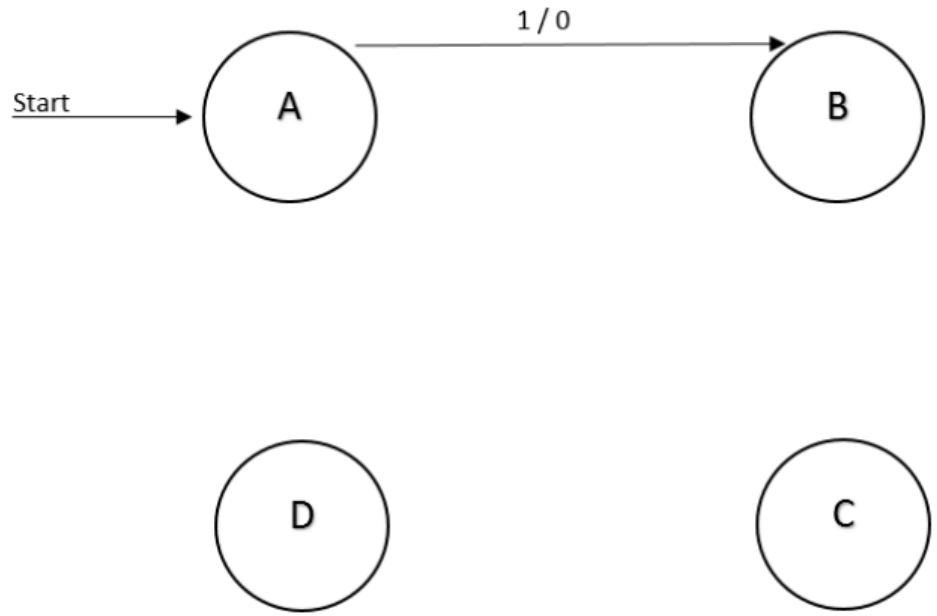
C = _____

(You must show your work above to earn points.)

b) Using as few states as possible, complete the transition table for an FSM that takes an input with 3 values: 0, 1, or 2. The machine will output a 1 when the sum of the inputs seen so far is divisible by 3. Otherwise it should output a 0.

Assume you have seen no digits at the start state. You might not need all of the states, and you should not draw additional states. You must represent your FSM using the table to the left, **the table is the only part that will be graded.** The first transition has been filled in for you.

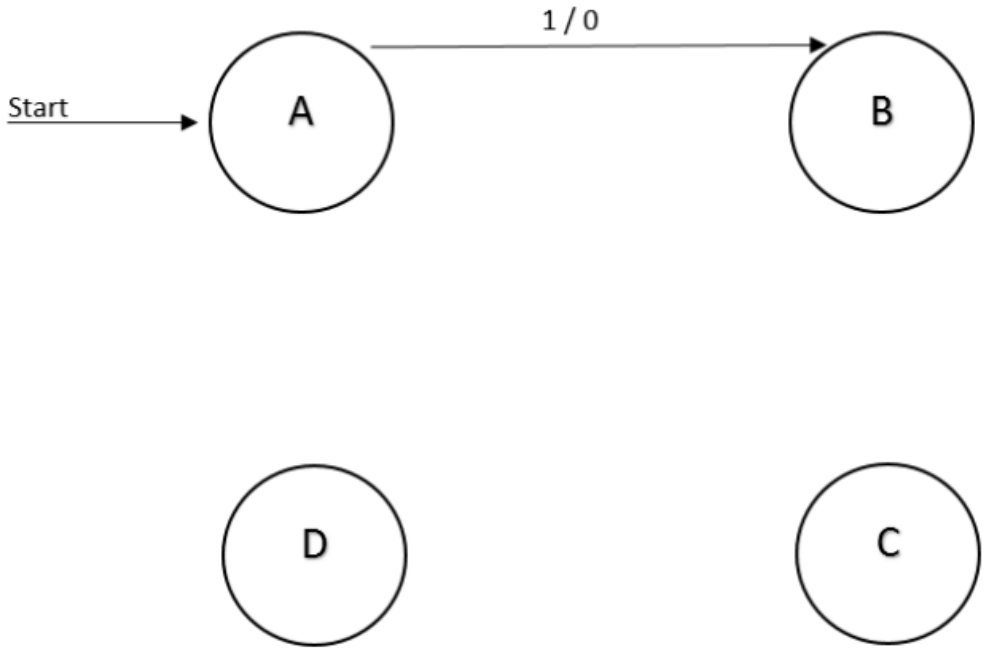
Current State	Input/Output	Next State
A	1/0	B



b) Using as few states as possible, complete the transition table for an FSM that takes an input with 3 values: 0, 1, or 2. The machine will output a 1 when the sum of the inputs seen so far is divisible by 3. Otherwise it should output a 0.

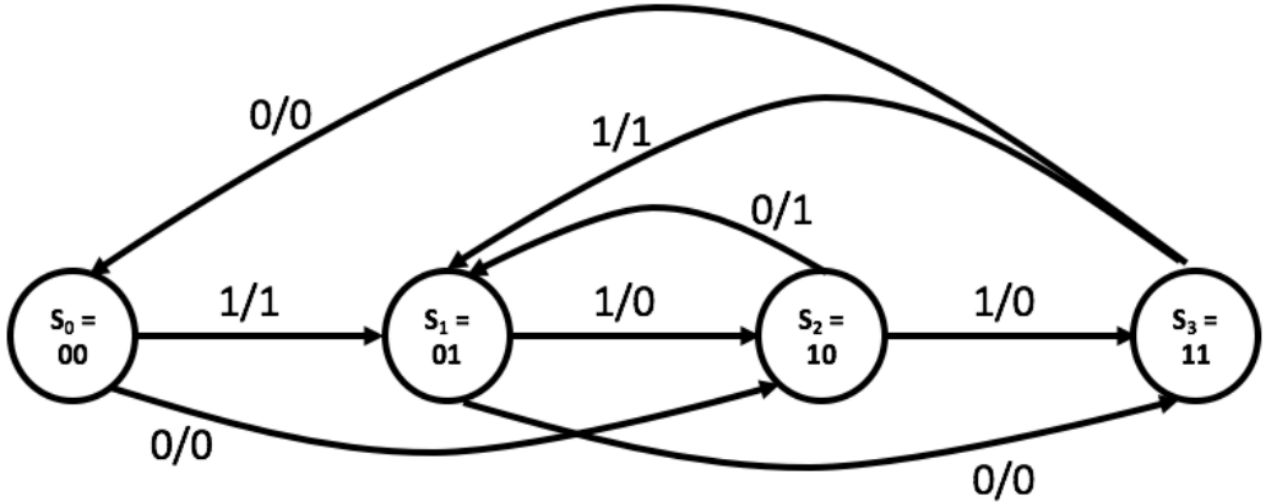
Assume you have seen no digits at the start state. You might not need all of the states, and you should not draw additional states. You must represent your FSM using the table to the left, **the table is the only part that will be graded.** The first transition has been filled in for you.

CURRENT STATE	INPUT/OUTPUT	NEXT STATE
A	1/0	B
A	0/1	A
A	2/0	C
B	2/1	A
B	0/0	B
B	1/0	C
C	1/1	A
C	2/0	B
C	0/0	C



Q1: Finite State Machine (8 points)

Answer the questions below for the finite state machine in this diagram:

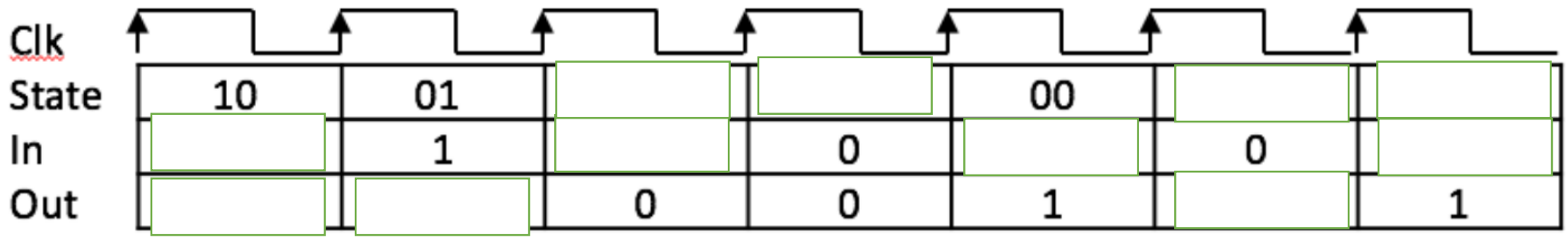


1. Complete the truth table shown below. (2 points)

Input		Output	
State	**In**	**State**	**Out**
S_0 = 00	0	S_2 = 10	0
00	1	01	1
01	0	11	0
01	1	10	0
10	0	01	1
10	1	11	0
11	0	00	0
11	1	01	1

Input		Output	
State	**In**	**State**	**Out**
$S_0 = 00$	0	$S_2 = 10$	0
00	1	01	1
01	0	11	0
01	1	10	0
10	0	01	1
10	1	11	0
11	0	00	0
11	1	01	1

2. Fill in the blanks in the diagram below. (3 points.)

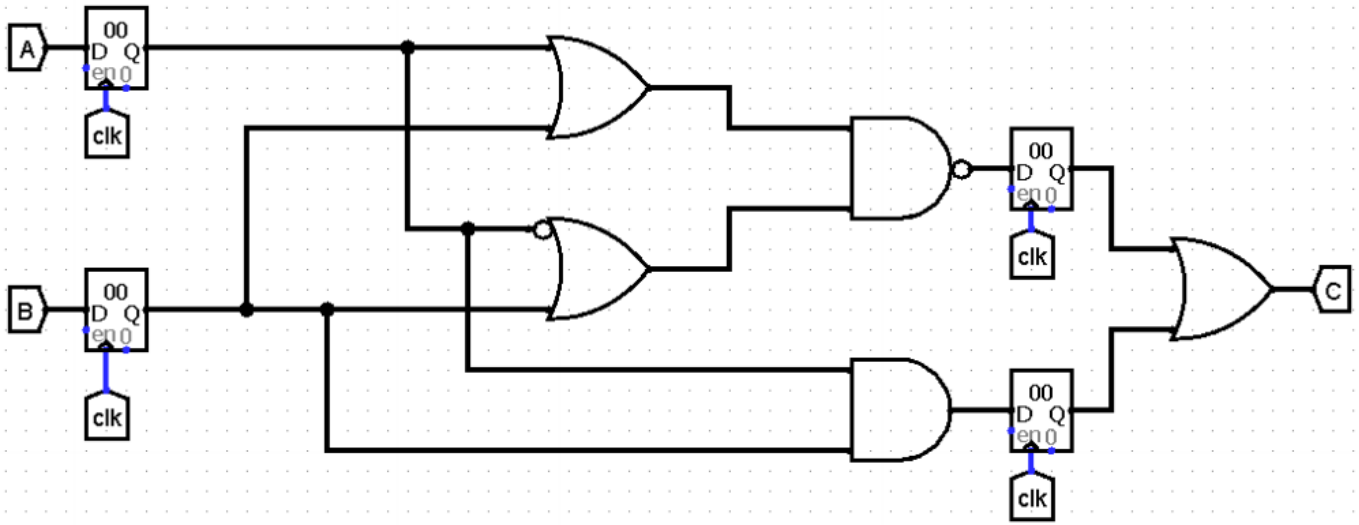


3. The finite state machine is required to operate at a frequency f_{clk} = 5GHz. The finite state machine is realized with combinatorial logic with delay t_c = 120ps and flip-flops with hold and clock-to-Q times of t_{hold} = 50ps and t_{clk2Q} = 70ps, respectively.

   What is the maximum value of the flip-flop setup time t_{setup} that allows the finite state machine to operate at a clock frequency of up to f_{clk} = 5GHz? Suggestion: draw a complete timing diagram. (3 points)

   $t_{setup} \leq$ _____200-120-70=10ps_____ (state the unit of your result).

c) Suppose we add registers to the unoptimized circuit in part A to increase the clock rate (this modification is shown below). What is the longest clock-to-Q that the registers on inputs A and B can have that will result in correct behavior when the circuit is clocked at 10 MHz?

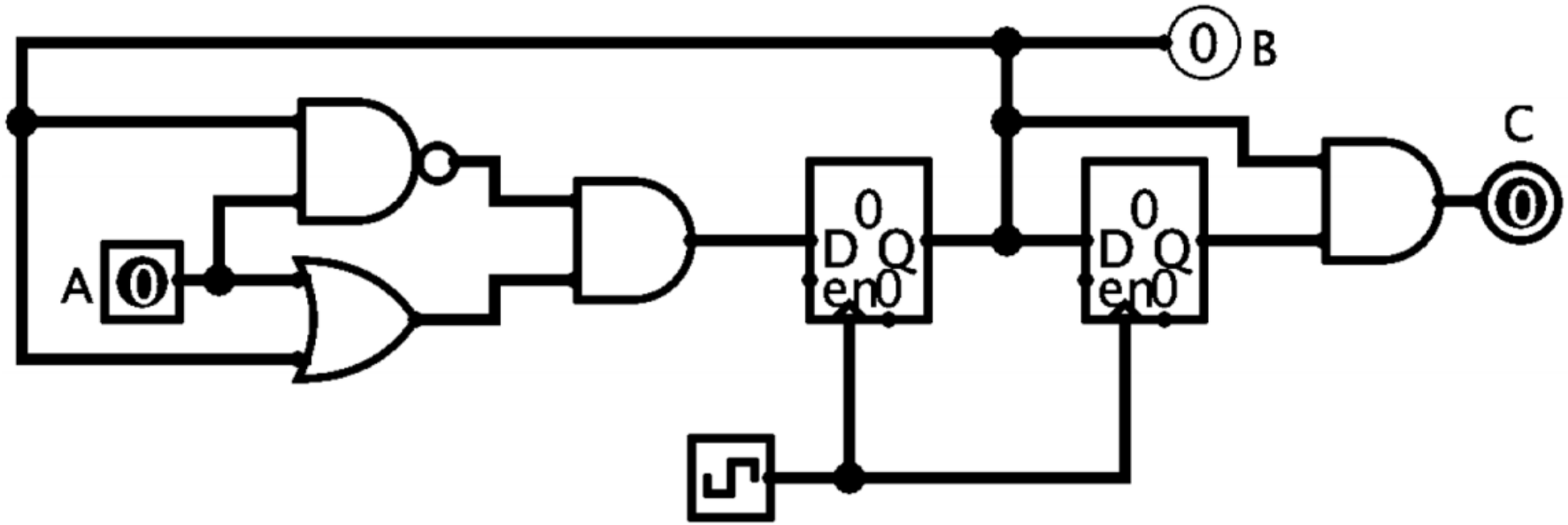


Assumptions:

- Assume that clock-to-Q > hold time
- All registers have a setup time of 2 ns
- All logic gates have a delay of 25 ns
- Bubbles on gates do not introduce additional delay

Answer: _____48ns_____

You are given the following digital circuit. The registers have a setup time of 5ns, a hold time of 3ns, and a CLK-to-Q delay of 5ns, and all logic gates have a delay of 20ns. Assume inputs A and B are driven by registers with the same specifications.



a) What is the critical path delay, and what is the maximum clock frequency at which the circuit will operate correctly? You may leave answers as fractions.

(CLK-to-Q + CL + CL + setup) = 50 ns
1 / (50 ns) = 20 MHz

b) Someone meddles with the circuit, increasing the register hold time to 10ns and setting the clock rate to 1 Hz. Will the circuit still work after these changes? Explain your reasoning.

Hold time violation. CLK-to-Q < hold time so the second register will fail

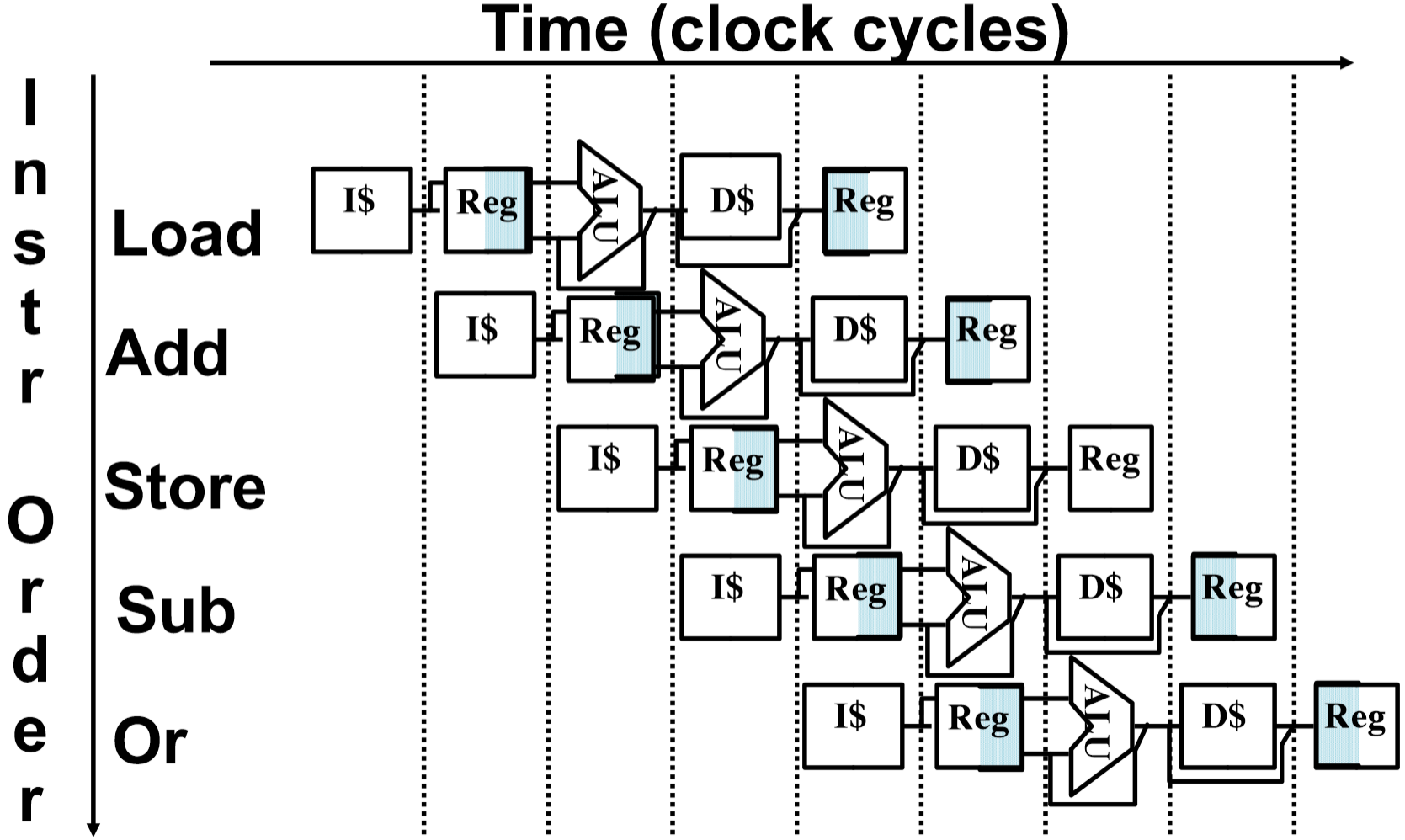
c) What simplification could be made to this circuit to decrease the critical path delay without changing the exact sequence of outputs?

Replace the left combinational logic with a single XOR gate

Note that adding a register would change the exact sequence of outputs (delaying by one)

Graphical Pipeline Representation

- RegFile: left half is write, right half is read

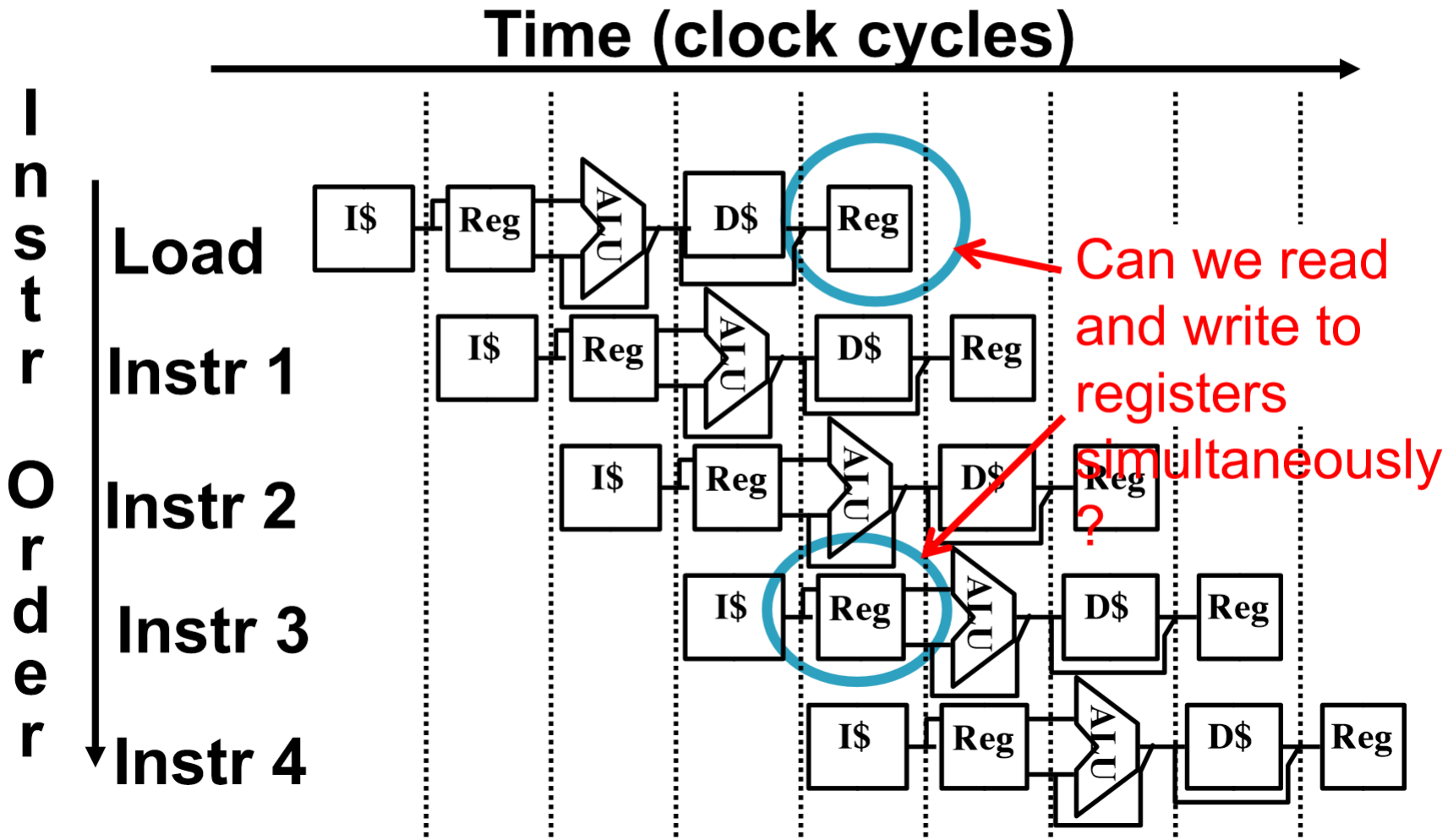


Pipelining Hazards

A *hazard* is a situation that prevents starting the next instruction in the next clock cycle

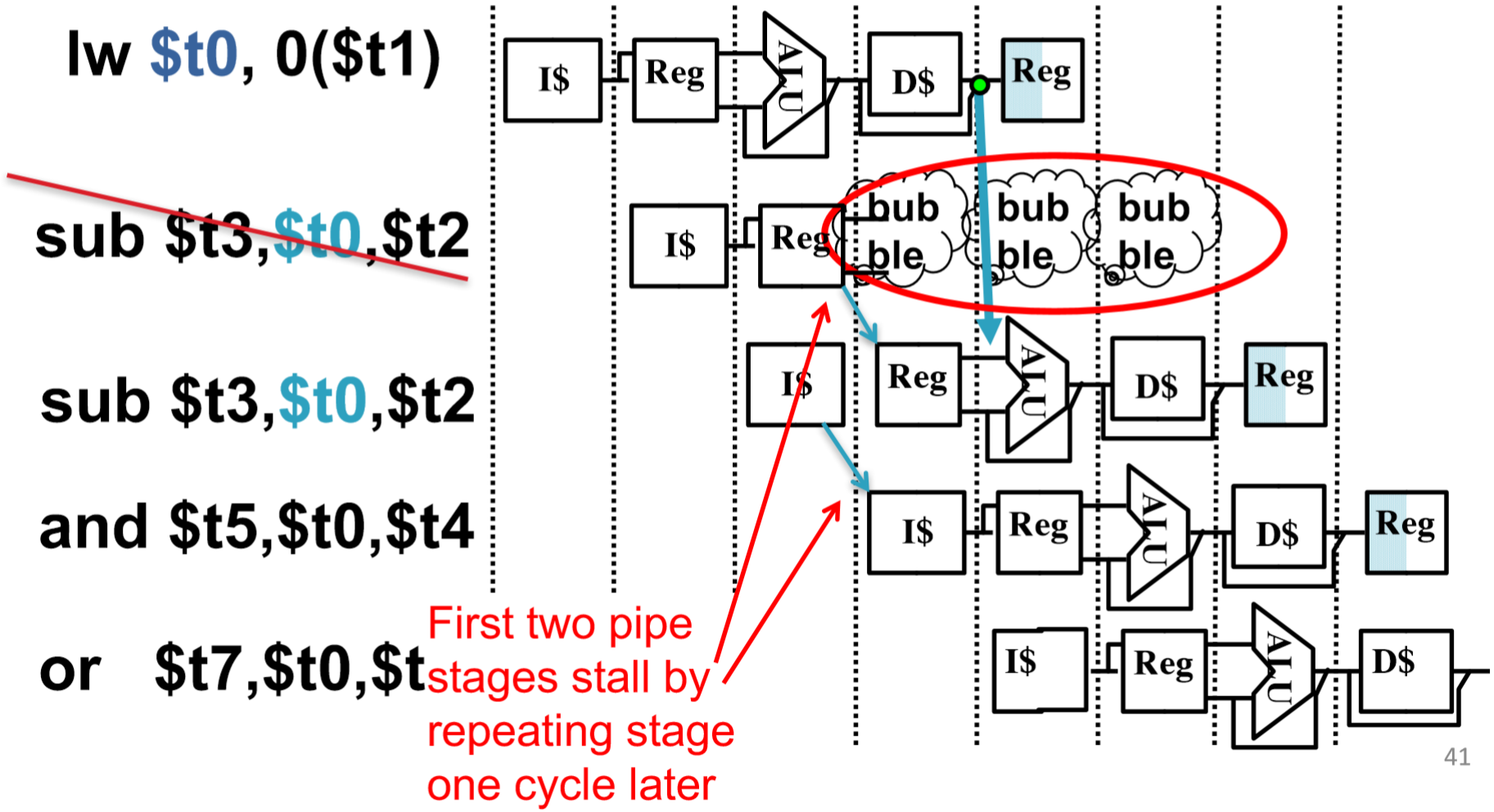
1) *Structural hazard*
   - A required resource is busy (e.g. needed in multiple stages)
2) *Data hazard*
   - Data dependency between instructions
   - Need to wait for previous instruction to complete its data read/write
3) *Control hazard*
   - Flow of execution depends on previous instruction

Structural Hazard #2: Registers (1/2)

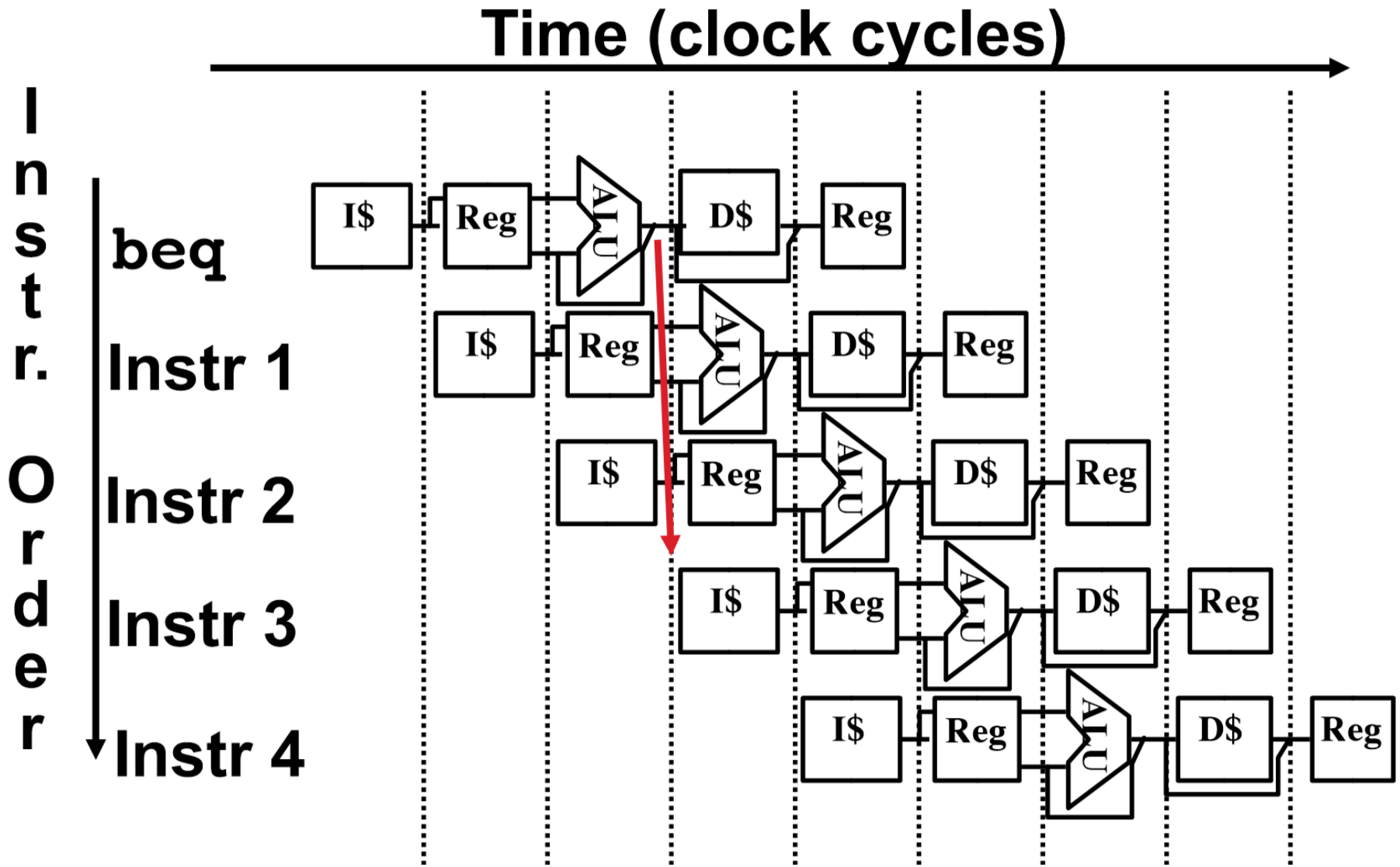


Data Hazard: Loads (3/4)

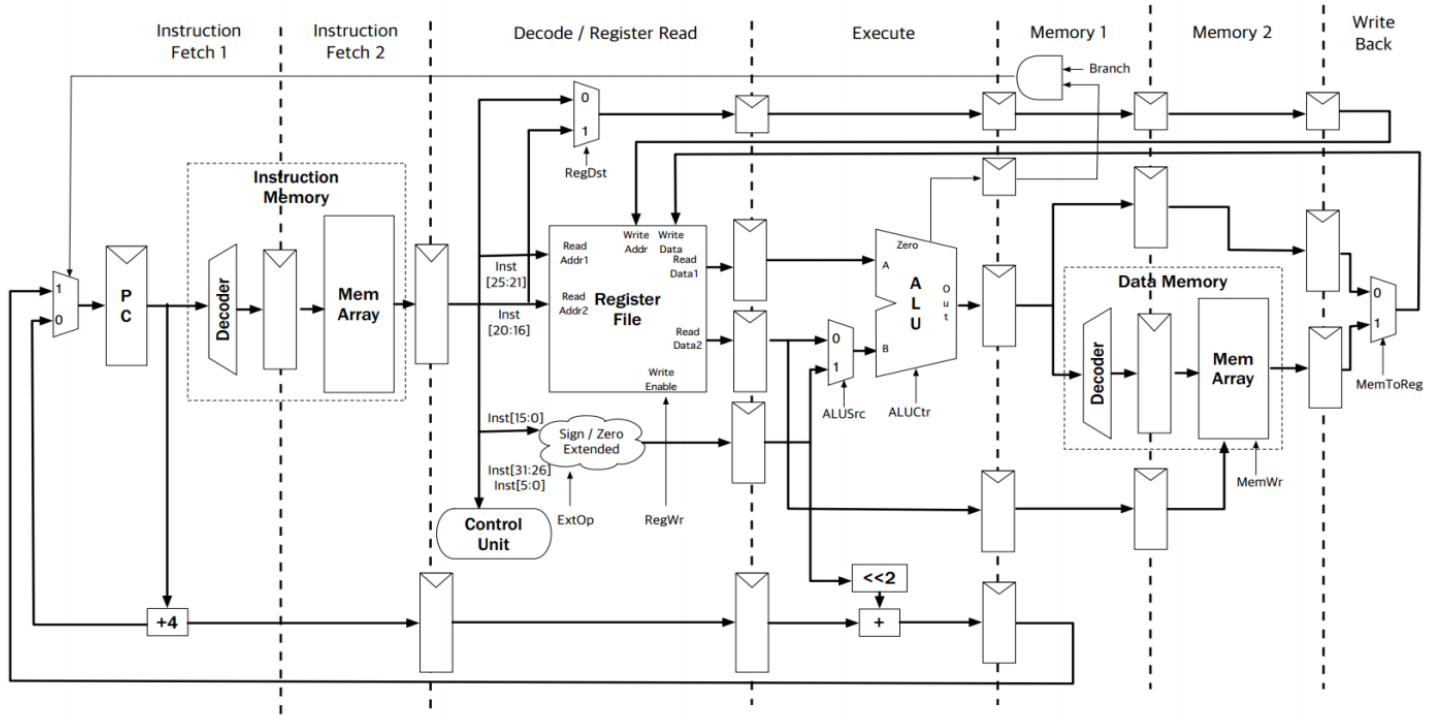
- Stalled instruction converted to “bubble”, acts like nop



Stall => 2 Bubbles/Clocks



We found that the instruction fetch and memory stages are the critical path of our 5-stage pipelined MIPS CPU. Therefore, we changed the IF and MEM stages to take **two** cycles while increasing the clock rate. You can assume that the register file is written at the falling edge of the clock.



Assume that no pipelining optimizations have been made, and that branch comparisons are made by the ALU. Here's how our pipeline looks when executing two add instructions:

Clock Cycle #	1	2	3	4	5	6	7	8
add $t0, $t1, $t2	IF1	IF2	ID	EX	MEM1	MEM2	WB	
add $t3, $t4, $t5		IF1	IF2	ID	EX	MEM1	MEM2	WB

Make sure you take a careful look at the above diagram before answering the following questions:

a. How many stalls would a data hazard between back-to-back instructions require?

3 stalls

b. How many stalls would be needed after a branch instruction?

4 stalls

c. Suppose the old clock period was 150 ns and the new clock period is now 100ns. Would our processor have a significant speedup executing a large chunk of code…

i. Without any pipelining hazards? Explain your answer in 1-2 sentences.

Yes, due to 1.5x throughput

ii. With 50% of the code containing back-to-back data hazards? Explain your answer in 1-2 sentences.

Yes, penalty is 300 ns per hazard in both cases, so our new processor will still have higher throughtput.

THX >.<