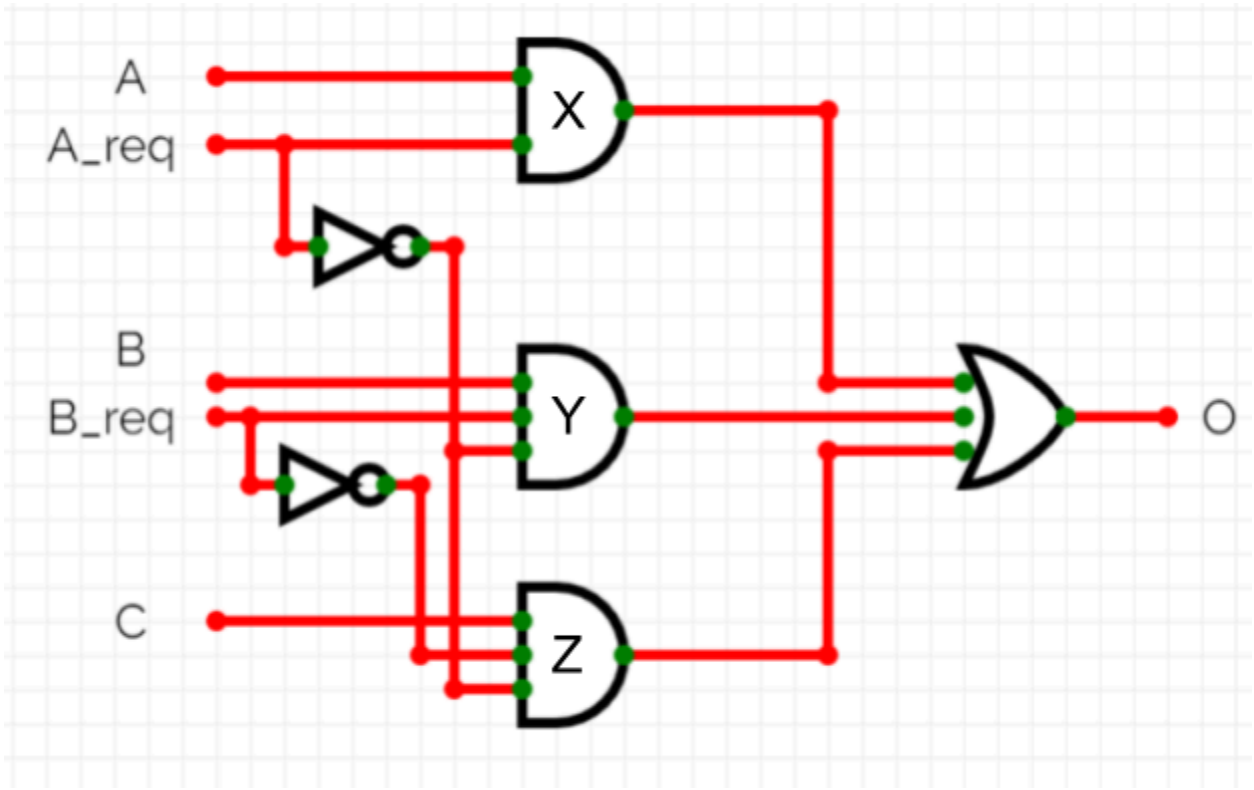
Problem 1: Circuit Delay *[15 points]* Topics: Timing Diagrams, Propagation Delay

Scribe: **[Scribe's name here]**

Consider the following naive implementation of a priority selector. This priority selector takes 3 data inputs, “A,” “B,” and “C” and one data output, “O.” It has 2 requester inputs, “A_req,” “B_req”. If A_req is asserted, O outputs the value of A. If A_req is not asserted but B_req is, O outputs the value of B. If no requesters are asserted, C is outputted.

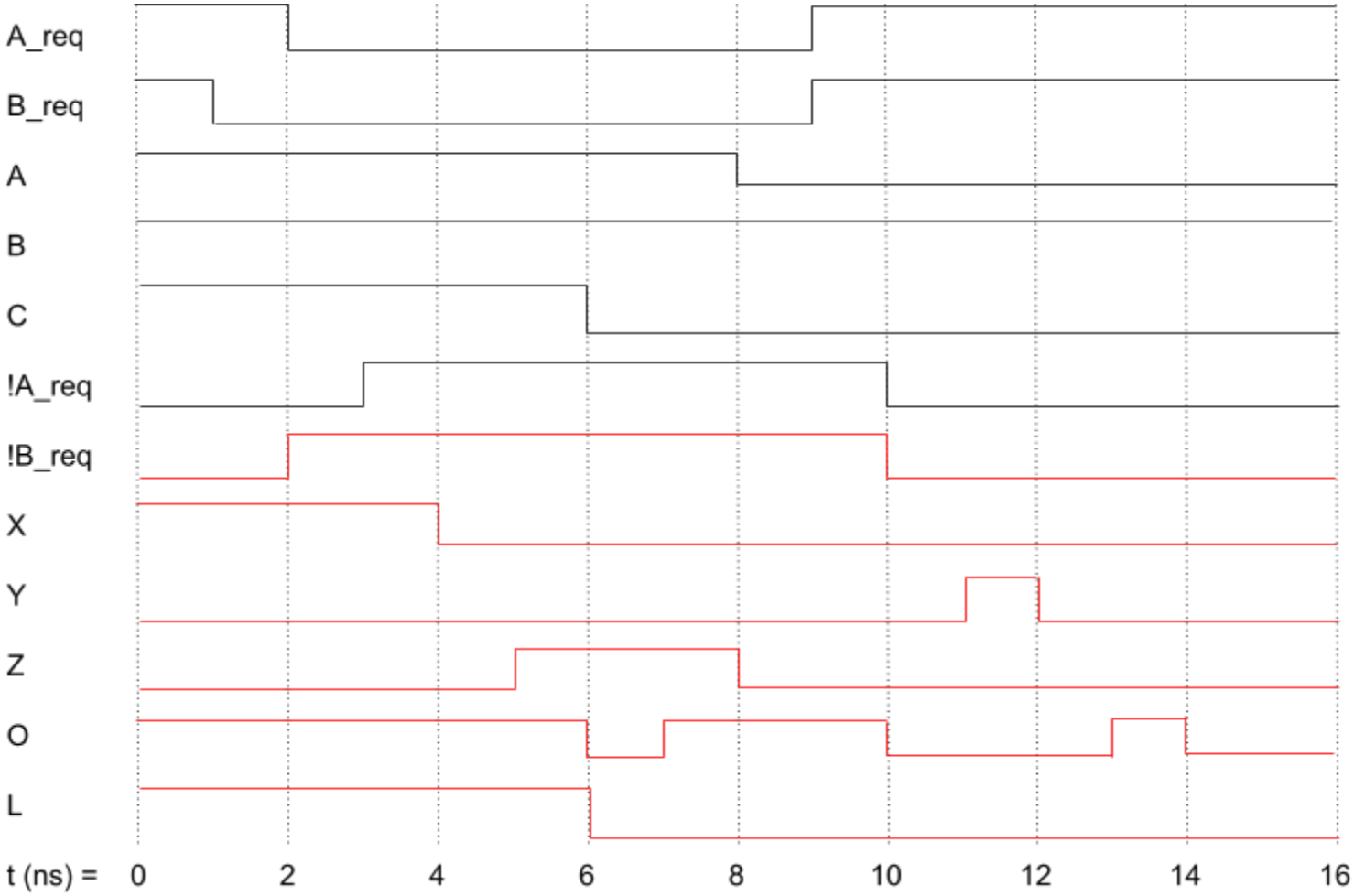


Assume that NOT gates have a 1ns delay, and that OR and AND gates have a 2ns delay regardless of the number of input wires. Assume no wire delay.

Say that for an infinite amount of time before we start measuring, all input signals are “1.”

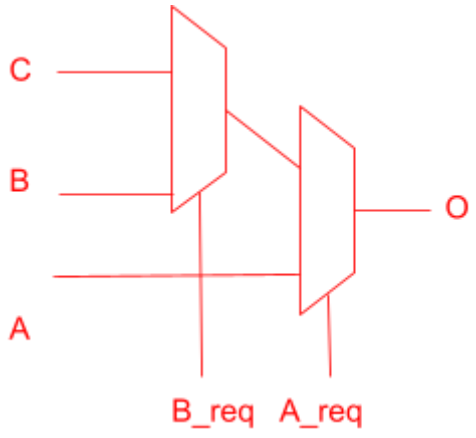
- At 1ns, B_req goes to “0”
- At 2ns, A_req goes to “0”
- At 6ns, C goes to “0”.
- At 8ns, A goes to “0”.
- At 9ns, A_req and B_req go to “1”.

a. Draw a timing diagram from t=0ns to t=16ns showing the input signals (A, B, C), requester signals (A_req, B_req), the output signal (O), and all intermediate gates (!A_req, !B_req, X, Y, Z). The inputs, requesters, and 1 intermediate gate have been done for you. Also include a line "L" (for Logic) depicting what the output signal would be with no delay. *[10 points]*
For drawings, you may draw by hand and insert a photo of your work, or merge PDFs before submission, or just digitally modify the drawing below.



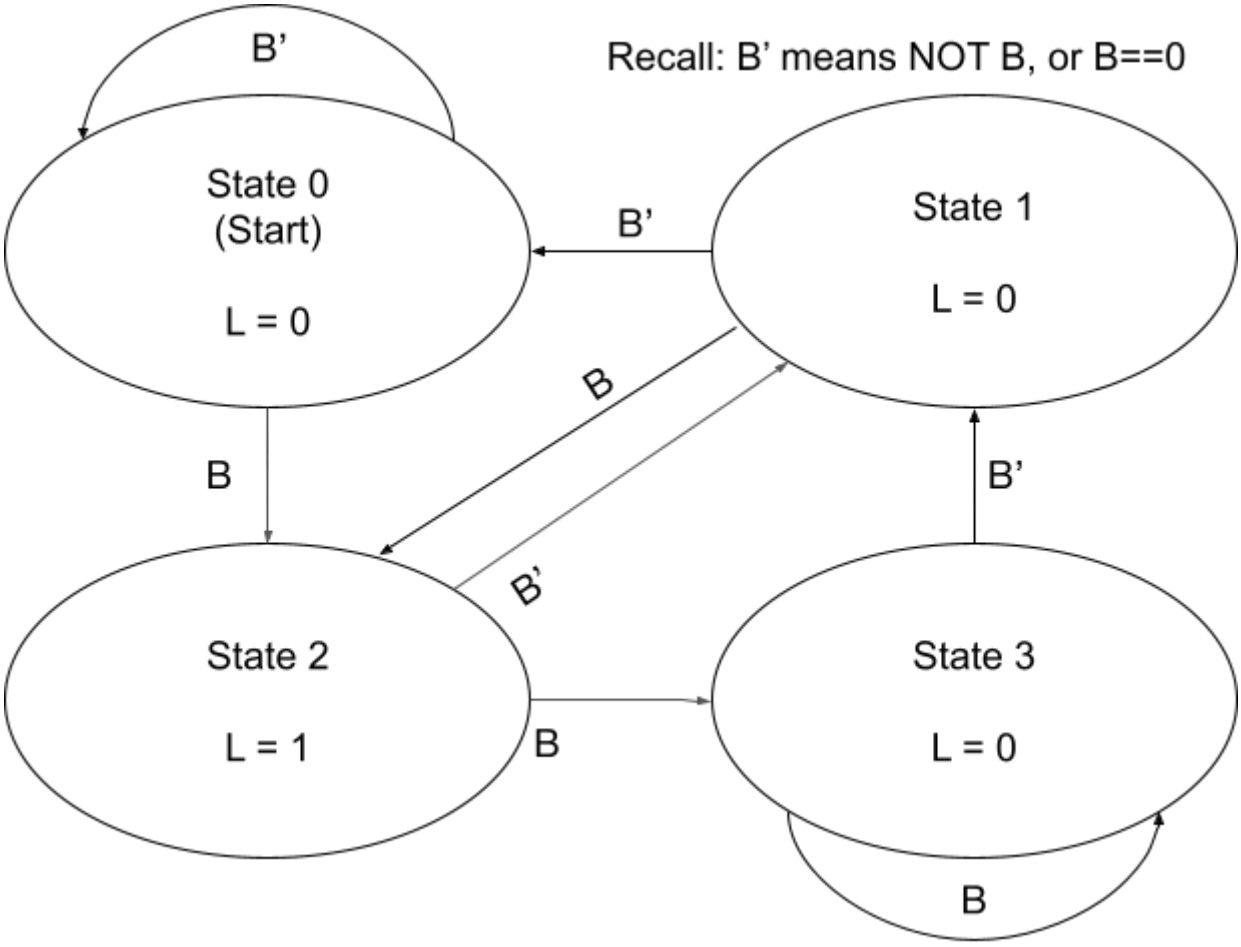
b. Draw a circuit using only two 2-input Muxes that implements the circuit. *[5 points]*
The definition of the 3-input priority selector is repeated:

- If A_req is asserted, O outputs the value of A.
- If A_req is not asserted but B_req is, O outputs the value of B.
- If no requesters are asserted, C is outputted.



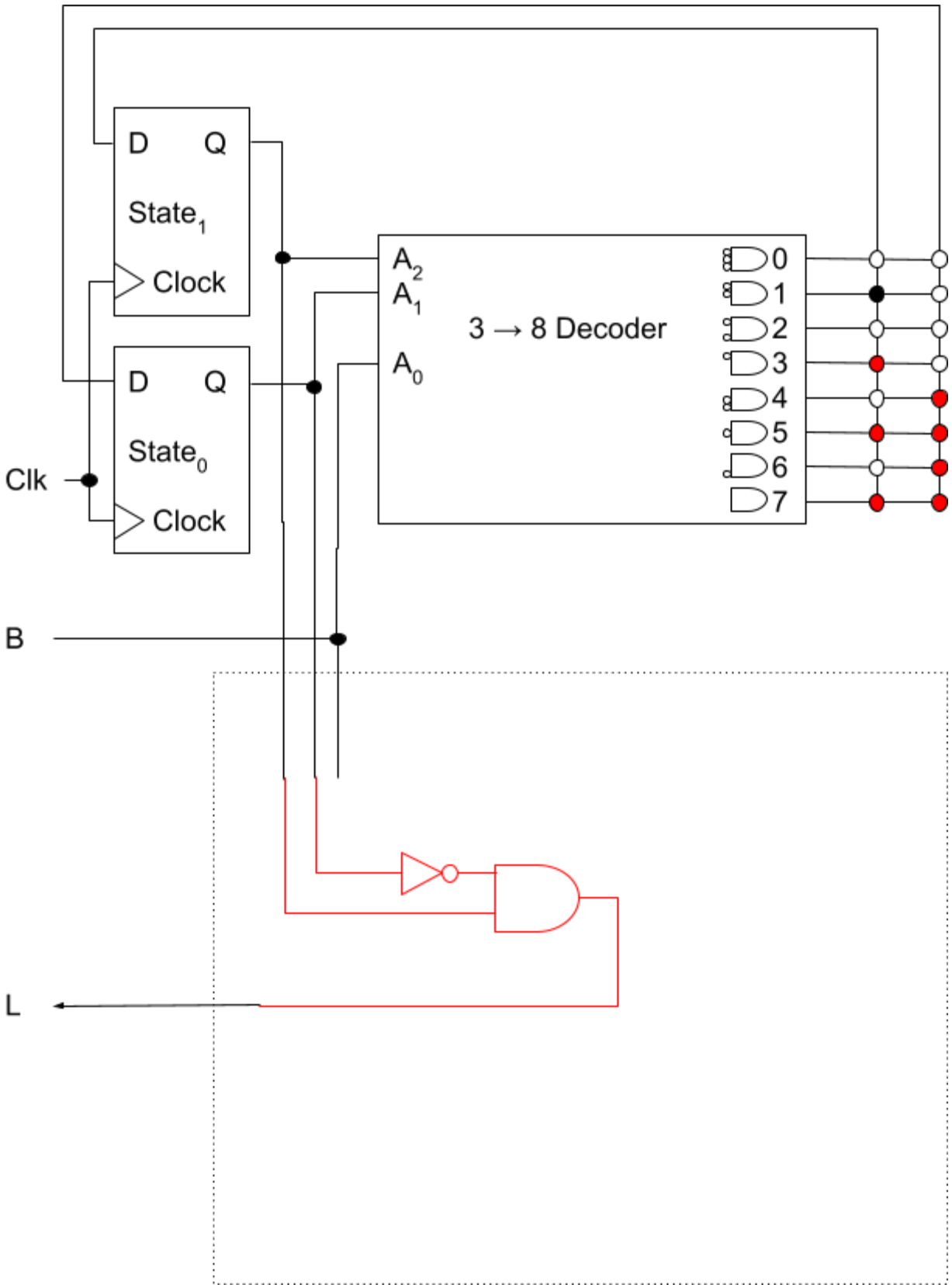
Problem 2: Finely Stated *[15 points]* Topics: FSM, Flip Flops, Decoders
Scribe: **[Scribe's name here]**

Consider the finite state machine diagram below. This FSM tracks the last 2 values of the input B, sets an output L to 1 if a code is received.



On the next page, convert the Finite State Machine into a circuit. Fill in the provided bubbles in the control ROM to determine the next state *[7.5 points]*. Inside the dotted box, add wires and logic gates to complete the rest of the logic for the output L *[7.5 points]*.

- Assume the clock is connected but not shown.
- Assume the flip flops have a starting value of 0.
- Do not add outputs to the control ROM, and do not modify anything outside the ROM bubbles and the dotted box.
- The decoder inputs are {$State_1$, $State_0$, B}
- Filling in one of the control ROM bubbles means there is a connection, so current flows and we get a 1. Otherwise, the wires are disconnected and we get a 0.
    - For example, state 0 is done for you in the first 2 rows. When the FSM is in state 0 and B=0, the next state is 0, so both Next State bits are 0 and not filled in. But when the FSM is in state 0 and B=1, the next state is 2 (binary 10), so the bubble for the Next $State_1$ bit is filled in while the bubble for Next $State_0$ is not.


D
Q
State1
Clock
D
Q
State0
Clock
Clk
A2
A1
A0
3 → 8 Decoder
0
1
2
3
4
5
6
7
B
L