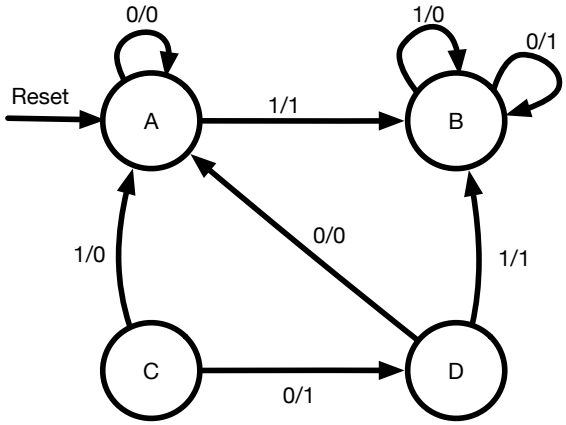
3 Finite State Machines [45 points]

3.1 Simplifying an FSM [20 points]

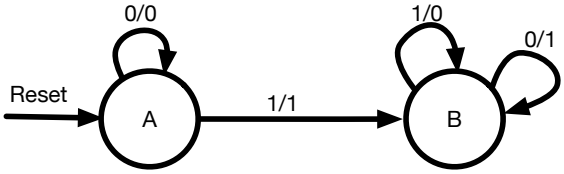
You are given the Mealy state machine of a *one input / one output* digital circuit design. Answer the following questions for the given state diagram.



(a) [10 points] Is it possible to simplify this state diagram and reduce the number of states? If so, simplify it to the minimum number of states. Explain each step of your simplification. Draw the simplified state diagram. If not, explain why it is not possible to simplify the state diagram.

Yes, it is possible.

- There is no way the state goes to C, so it is a non-used state.
- After deleting C, there is no way the state goes to D, so D is also a useless state.
- We can simplify the state diagram as shown next:



(b) [10 points] Assume this state machine is used to process binary numbers from the least significant bit to the most significant bit. You are given an input bit stream: "10110100". Please show the output bit stream produced by this FSM.

"01001100"
When processing a bit stream from the least significant bit to the most significant bit, this state machine keeps the bits unchanged until the first "1" comes. Then, all the bits are flipped (not include the first "1") after the first "1" comes. Therefore, the output is "01001100".

3.2 Designing an FSM [25 points]

Design a Moore finite state machine (FSM) with one input and one output. The input provides an unsigned binary number in a bit-serial fashion from the most-significant bit to the least-significant bit. The output should be logic-1 in a clock cycle if the provided input so far is divisible by 8 (i.e., [the input number] $\bmod 8 = 0$). (Hint: Recall that the output depends only on the current state in a Moore FSM.)

Below are some example bit-streams that should output a logic-1 value.

- 1000
- 10000
- 11000
- 111000
- 101000

To start an input bit stream, the user should reset the FSM. Draw the state diagram and explain why it works. Your state machine should use as few states as possible and each state should have a precise definition and output.

From the given examples, we can see that strings can be exactly divided by 8 are all ended with "000" (i.e., three "0"s). Then, we define S_0, a state where the number is ended with "000".
If "1" comes, then the number cannot be exactly divided by 8 and it lacks three "0"s at the end. We define this state as "E" state (S_1), which means no zero at the end.
When there is a "0", then the number lacks two "0"s to be exactly divided by 8. Therefore, we define the state as "0" (S_2).
When there are two "0"s, then the number lacks one more "0" to be exactly divided by 8. Therefore, we define the state as "00" (S_3) .
Based on the analysis above, we can draw the finite state machine whose output (i.e., O) is "1" at S_0 (is "0" at other states):