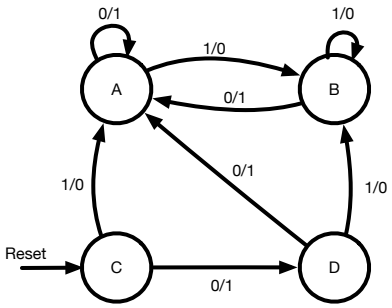
# 2 Finite State Machines [45 points]

## 2.1 Simplifying an FSM [20 points]

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




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. Below is the state transition table of the given state machine:

| Current State | Input | Next State | Output |
|---|---|---|---|
| A | 0 | A | 1 |
| A | 1 | B | 0 |
| B | 0 | A | 1 |
| B | 1 | B | 0 |
| C | 0 | D | 1 |
| C | 1 | A | 0 |
| D | 0 | A | 1 |
| D | 1 | B | 0 |

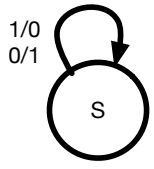
We can see that the states A, B, and D are identical. For all of these states,

- an input of 0 leads to the next state A and the output 1
- an input of 1 leads to the next state B and the output 0

Therefore, we can merge states A, B, and D. Let's use the name X:

| Current State | Input | Next State | Output |
|---|---|---|---|
| X | 0 | X | 1 |
| X | 1 | X | 0 |
| C | 0 | X | 1 |
| C | 1 | X | 0 |

We can further simplify this state machine as both states C and X are identical in terms of their next state and output. As a result, this state machine has *only* one state and the output is always the inverse of the input.



## 2.2 Designing an FSM [25 points]

Design a Moore finite state machine (FSM), where each output is solely determined by the current state of the machine and *not* directly influenced by the inputs. The state machine should have one input and one output. This FSM's goal is to detect a stable transition in the input signal from repeated logic-0 to repeated logic-1. The output should be logic-1 *only* when the input sequence of "0-0-1-1" is observed. The output should be zero in all other cases.

When the circuit is reset, your state machine should assume that the input signal has been high (logic-1) for a long time. 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.

We need to keep track of the bit values in the last four bits. This requires 16 states. However, many of these states behave the same. We can reduce the number of states down to five.

- Since this is a Moore machine, the output should be independent from the input. Therefore, there should be a state for the posedge where the output is "1". All other states will have the output of "0". The FSM goes to the posedge state only when the last four bits are 0-0-1-1. We call this state S0011.
- The FSM should reach to the posedge state from another state where the last three input bits are 0-0-1. We call this state SX001.
- The FSM should reach to the 0-0-1 state from a state where the last two input bits are 0-0. Note that it does not matter what the input bits are, earlier than the last two zeros. We call this state SXX01.
- The FSM should not stay in state S0011 for more than one clock cycle as when the new input comes, the last four bits will not be 0-0-1-1 anymore. If the input is 1, the next state should be SXXX1: the last bit is zero but it is not a posedge and the earlier bits are not important. If the input is 0, the next state should be SXX10: the last two bits are 1-0 and the earlier bits are not important.
- Intuitively, if the state is SXXX1, the FSM should remain at this state if the input is 1 and go to SXX10 if the input is 0.
- If the state is SXX10, the FSM should *not* remain at this state regardless of the input. If the input is 0, the next state is SXX00. If the input is 1, the next state is SXXX1.

Therefore, it is possible to design this state machine with five states. The state diagram is shown below.