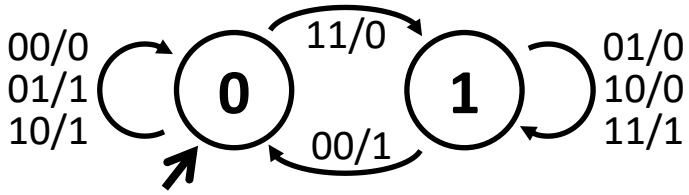
Design of Digital Circuits and Systems, HW1

FSM and SystemVerilog Review

Solution Outlines

Problem 1:



- Transitions shown are **`xy/S`**.
- The state names and binary encodings correspond to the stored value in the flip-flop. The opposite binary encoding could be used, but makes less intuitive sense and would have to be properly documented.
- The reset state wasn't specified, though students should have picked one; state 0 is shown above.

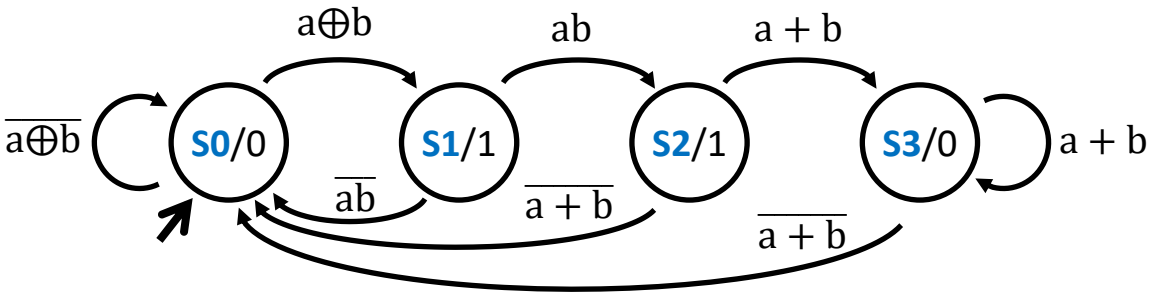
SystemVerilog module(s) and testbench should have been submitted, but are not shown here. Many acceptable solutions exist for the SystemVerilog implementation. We would have suggested connecting the synchronous reset D flip-flop and full adder modules given in 271/369, but students may have coded their FSM instead.

Problem 2:

SystemVerilog module and testbench should have been submitted, but are not shown here. Key points:

- There are 5 states, so 3 state bits are needed, which should have been defined as `parameters` or as part of an `enum`.
- The output logic can be simplified from a Karnaugh map or done via inspection.
- The testbench should check the output and transition from every edge for every state.

Problem 3:

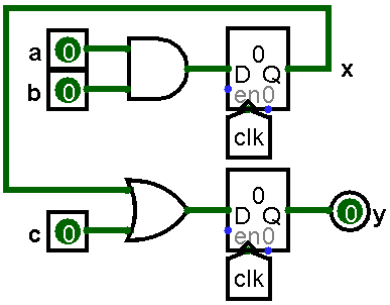


This SystemVerilog code most naturally describes a ***Moore*** machine, since the output depends on the state; one possible representation is shown above. The "equivalent" Mealy machine would have the transitions *into* these states as 1, but this causes the Mealy machine output to appear 1 clock cycle earlier than the Moore machine (this is expected) so Mealy machine with the state outputs on the transitions *out* of the state were also accepted for this assignment.

Problem 4:

Non-blocking assignments:

`x` and `y` update in parallel, using the old/original values of `a`, `b`, `c`, and `x`, meaning that both `circuit1` and `circuit2` imply hardware that looks something like:



Blocking assignments:

The assignments are now implied *in sequence*. If the `y` assignment comes first (*i.e.*, `circuit2`), then it still uses the old value of `x`, so we get the same hardware as above. If the `y` assignment come second (*i.e.*, `circuit1`), then the implied value of `x` is `a & b`, implying hardware that now looks something like: