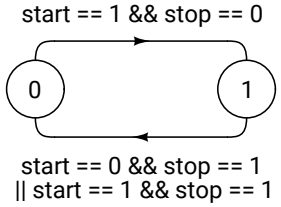
State Machines

Exercise 1:



Draw the schematic for this state machine. Write an **`always_ff`** statement that implements this state machine.

Exercise 2: Simplify the solution using the fact that addition of 1 to 3 “wraps” a 2-bit value to 0. What would you change so the counter does not “wrap around” from 2’b11 to 2’b00?

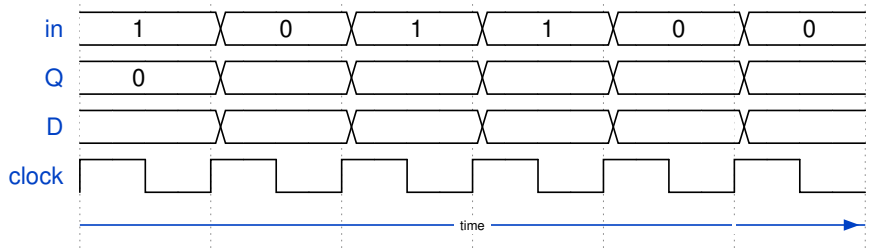
Exercise 3: What value of N would give a 20 ms delay if the clock frequency is 50 MHz? How many bits are needed for this timer's register?

Exercise 4: Assume the timer above is reset to $N - 1$ each time it reaches 0. For how long is the register value 0? What are the period and frequency of a signal that is inverted each time the count reaches 0?

Exercise 5: Write the state transition table for the state machine for the **`lights`** output.

Exercise 6: The example above is an N-bit shift register that shifts the bits right. Draw a block diagram and write the Verilog for a 6-bit shift register that shifts left.

Exercise 7:



Fill in the diagram above for a 4-bit ($N = 4$) right-shift shift register. Assume the initial value is zero. Which bit is the oldest (first) value in the D waveform? Which bit of the shift register holds the oldest value?

Exercise 8: Draw a block diagram and write the Verilog for a circuit that sets an output named **`detect`** high when the sequence of values 1, 1, 0, 1 has appeared on an input named **`in`** on successive rising edges of the clock.

Exercise 9: For which states would **`falling`** be asserted? **`rising`**? Draw the schematic and write the Verilog for this state machine. Assume an input **`in`** and a 2-bit register **`bits`** that holds the two most recent input values.

Exercise 10: How could you modify the code so that **`digits`** is only updated when an **`enable`** input is asserted?

Exercise 11: How many states can this state machine have?

Exercise 12: Draw the state transition diagram for this simpler implementation. How many states are there? Write the Verilog using a 3-bit **`count`** state variable.

Exercise 13: Write `always_ff` statements that implement these two state machines.

Exercise 14: What is the size of the expression `sqrt*sqrt`? Of `{8'b0,sqrt}*sqrt`?

Exercise 15: Draw the state transition diagram for **done**.

Exercise 16: If we used 8-bits of state information, how many states could be represented? What if we used 8 bits of state but used a “one-hot” encoding?