CSM 61C
Fall 2019

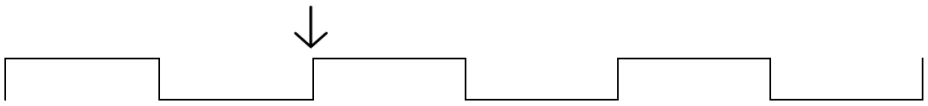
SDSs and FSMs

Mentoring 5: October 14, 2019

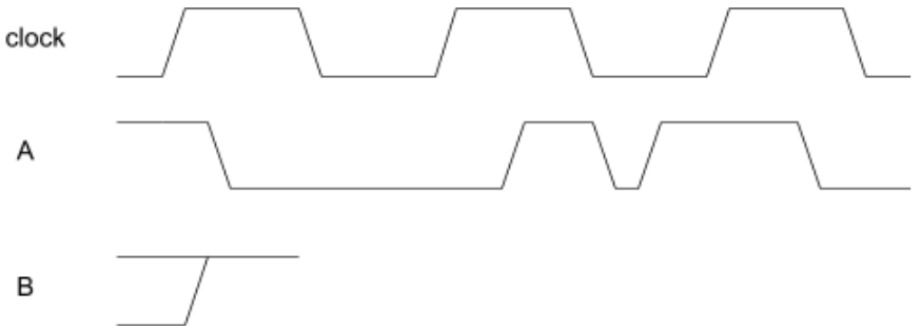
1 Synchronous Digital Systems

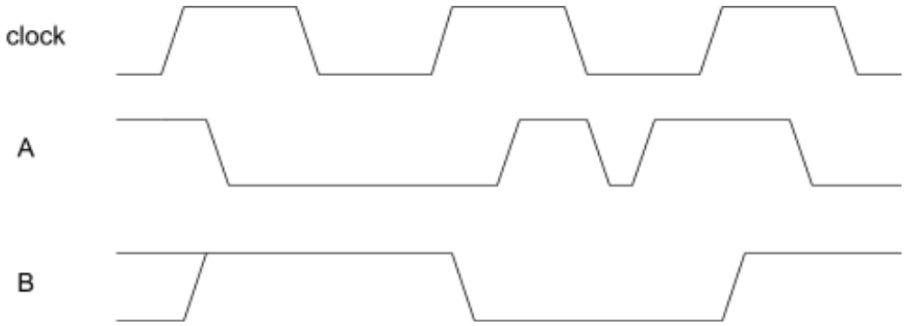
Synchronous Digital Systems (SDSs) are systems designed to process input as it comes in and output it as quickly as it comes in. The input is always a series of pins with either high or low voltage, and the output is another series of pins, each with either high or low voltage. A clock is used to tell the input and output when to switch to the next value; the switch always occurs on the rising edge (low to high voltage) of the clock. This is so that the output is always consistent, otherwise halfway through the switch of the input, the output might be based on the old input or the new, depending on the quality of the circuit and also some random chance based on electron movement. Note: the output always takes a small amount of time to stabilize after the rising edge.

When slowed down, clock cycles will look something like below. This is three clock cycles in a row, and here is the rising edge:



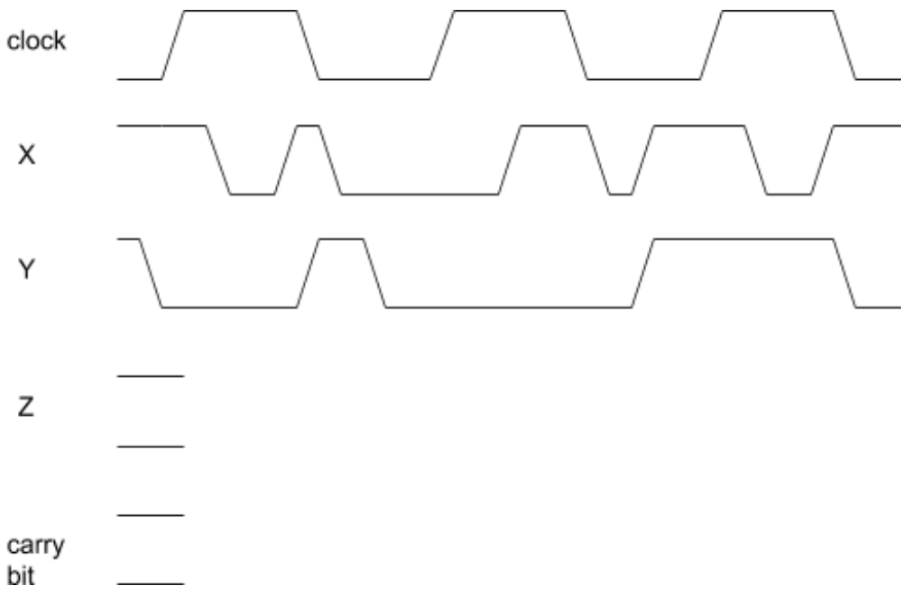
1.1 On the rising edge of the clock, B is set to A and is held constant until the next rising edge. Fill in the diagram for B's value.

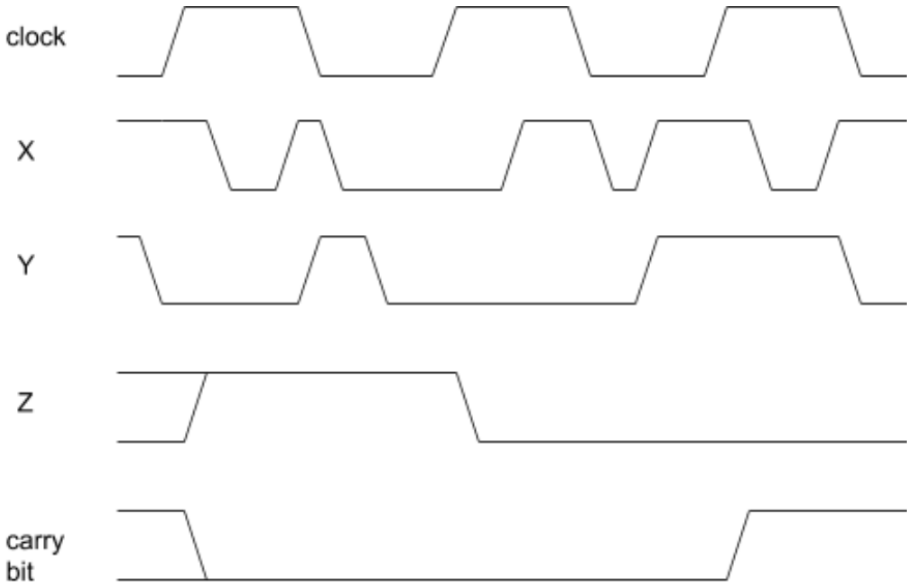




Solution:

1.2 In a clock cycle, complex operations can occur. For example, most of the RISC-V instructions can be done in 1 clock cycle. Below, the diagram shows the equation Z = X + Y where Z is computed on the rising edge of the clock. If Z overflows, place the overflow in the carry bit.





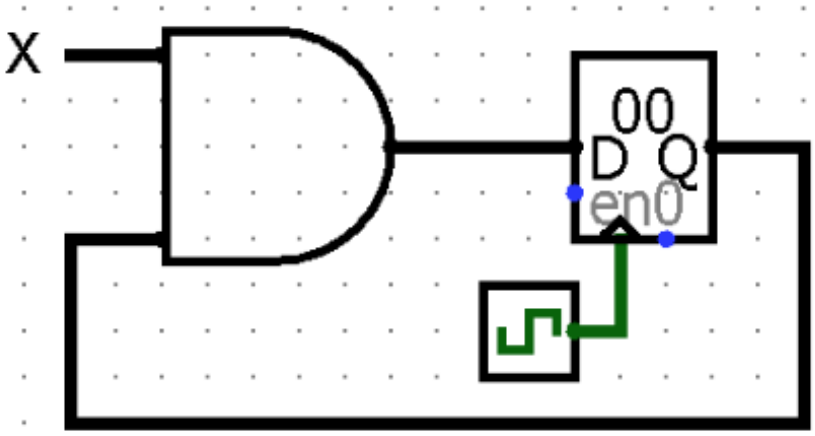
Solution:

2 SDS With Registers

For the circuit below, assume:

1. setup time is 15ns
2. hold time is 30ns
3. AND gate delay is 10ns.

If the clock rate is 10 MHz and x updates 25ns after the rising edge of the clock,what are the minimum and maximum values for the clk-to-Q delay to ensure proper functionality?



2.1 Min: _____ ns

2.2 Max: _____ ns

Min: 20 ns

Max: 75 ns

If the clk-to-Q delay is too fast, the input to the register will change before the hold time is finished. Thus, the minimum clk-to-Q delay is t_{hold}–t_{and} = 30–10 = 20 ns. On the other hand, we must make sure the critical path is no longer than the clock period, which is $100ns (= 1/(10MHz))$. In other words, $t_{setup}+t_{and}+t_{clk-to-Q} \leq 100ns$, or $tclk-to-Q \leq 100ns-t_{setup}-t_{and}$. Solving yields $t_{clk-to-Q} \leq 75ns$.

3 Finite State Machines

Suppose we want to design a FSM that takes a single bit (1/0) as input, and outputs a single bit (1/0). We want the FSM to output true (1) only if it has seen three consecutive 1's as its input. Let's design it!

3.1 Given the following input streams, fill in what the FSM should output at each time step:

(a)

In	0	1	0	1
Out				

In	0	1	0	1
Out	0	0	0	0

The FSM has only seen individual 1's (non-consecutive), so it should output only zeros.

(b)

In	1	1	0	0
Out				

In	1	1	0	0
Out	0	0	0	0

The FSM has only seen two consecutive 1's, but not three. It should still output only zeros.

(c)

In	0	1	1	1
Out				

In	0	1	1	1
Out	0	0	0	1

The FSM has finally seen three consecutive 1's, but it should only output 1 (true) after it has seen the third 1.

3.2 Let's consider the design of the FSM with more formality.

(a) If a 0 is input into the FSM, what should the FSM output?

The FSM should always output 0. We only care about outputting 1 when the input is 1.

(b) If a 1 is input into the FSM, what does the FSM need to remember to make the correct decision?

The FSM needs to remember how many 1's were input before the current 1. More concretely, it needs to know whether no 1's, one 1, or two 1's were previously inputted.

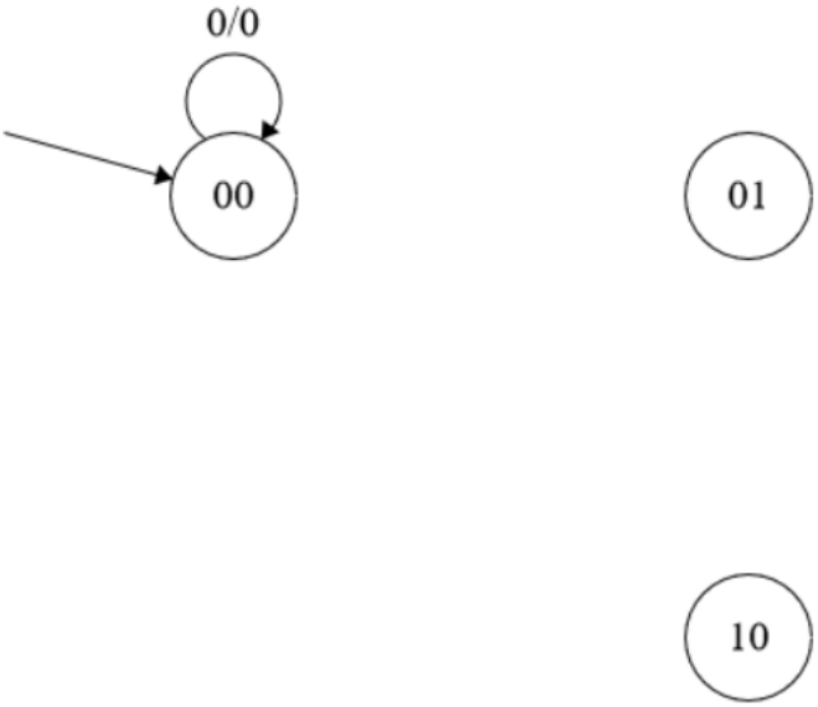
(c) How many unique states does the FSM need?

Building off of part b), we need 4 states in total for the following purposes:

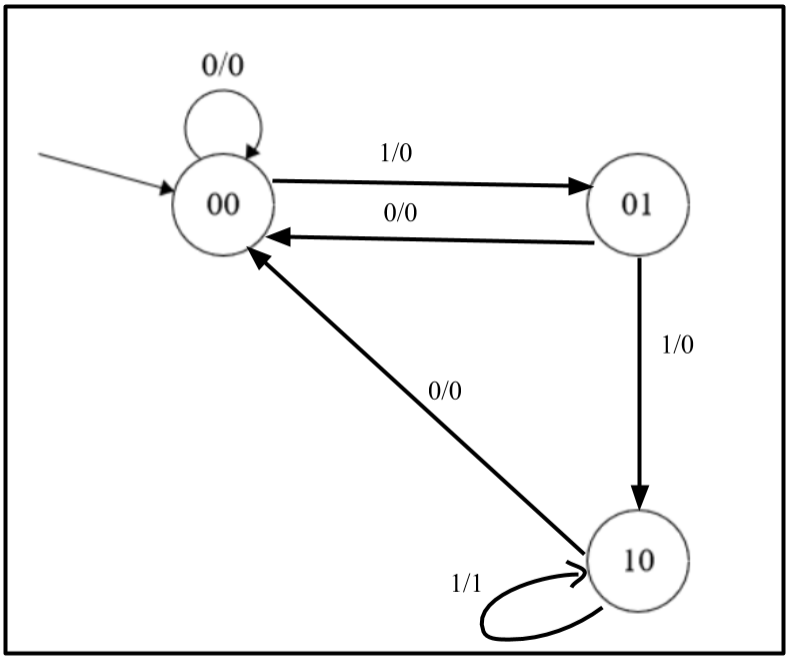
- Start/Reset: A starting point for the machine as well as a reset if we ever see 0
- One 1: We've seen one 1 so far.
- Two 1's: We've seen two 1's so far. This needs to be distinguishable from only seeing one 1.
- Three 1's: We've seen three 1's so far. This needs to be distinguishable from the other states

Meta: Note that three consecutive 1's is indistinguishable from seeing four, five, six, etc. consecutive 1's.

3.3 Fill in the FSM.

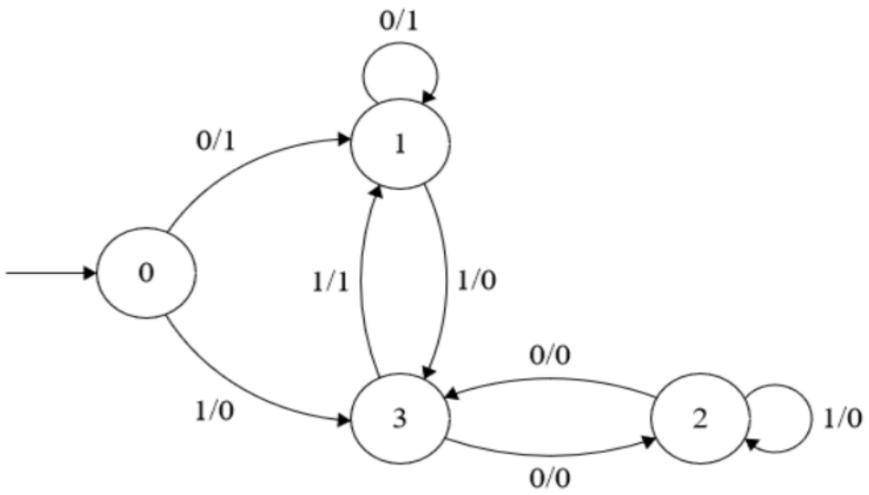


Solution:



Meta: The states can be named in decimal, not necessarily binary. Meta 2: Please excuse the poorly drawn self-loops. No (simple) drawing application likes them apparently.

3.4 Consider the following FSM. What does it do?



Meta: This will likely need more scaffolding. The FSM outputs a 1 whenever the input is a multiple of 3. Some ways to approach analyzing:

- 1 is a less common output than 0 → what causes a 1 to be output?
- Trace along the paths → what causes a 0 to be output?
  - Consider 0, 0, 1 → 0b001 = 1 is not a multiple of 3
  - Consider 1, 1 → 0b11 = 3 is a multiple of 3
  - Consider 1, 1, 1 → 0b111 = 7 is not a multiple of 3
- Brute-force: draw out the truth table