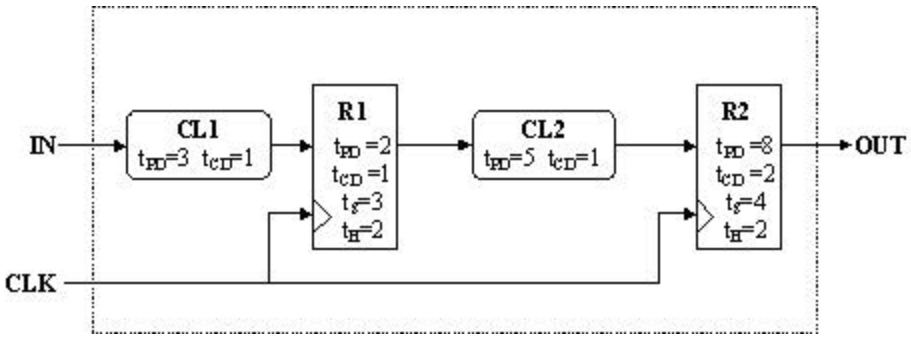
Show All Answers Hide All Answers

Memory components and finite-state machines

★ indicates problems that have been selected for discussion in section, time permitting.

Problem 1. Consider the following diagram of a simple sequential circuit:



The components labeled CL1 and CL2 are combinational; R1 and R2 are edge-triggered flip flops. Timing parameters for each component are as noted.

A. ★ Write the timing specifications (tS, tH, tCD, tPD, tCLK) for the system as a whole using the timing specifications for the internal components that are given in the figure.

Hide Answer

It's a good idea to check if the circuit will work for any clock period. We check this by checking if the total contamination delay from R1 to R2 is long enough to cover the hold time of R2. In other words,

tH,R2 <= tCD,R1 + tCD,CL2
2 <= 1 + 1
2 <= 2

The inequality is satisfied, so we can determine the timing specifications of the system.

The setup time and hold time of the system is determined by the setup time and hold time required for the signal IN, which is the input to CL1. Thus,

tS = tPD,CL1 + tS,R1 = 6, and
tH = tH,R1 - tCD,CL1 = 1.

The contamination and propagation delay of the system is determined by the contamination and propagation delay of the signal OUT, which is the output of register R2. Thus,

tCD = tCD,R2 = 2, and
tPD = tPD,R2 = 8.

The clock period for the system is determined by adding all the propagation delays from R1 to R2, and the setup time for R2.

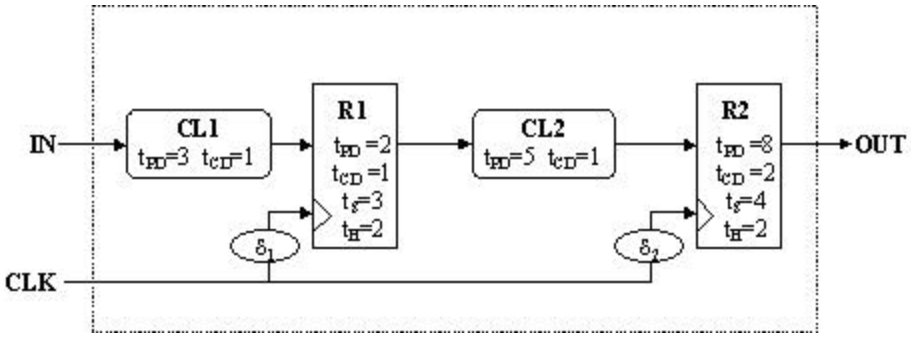
tCLK >= tPD,R1 + tPD,CL2 + tS,R2
tCLK >= 2 + 5 + 4
tCLK >= 11

B. ★ Suppose you had available a faster version of CL2 having a propagation delay of 3 and a contamination delay of zero. Could you substitute the faster CL2 for the one shown in the diagram? Explain.

Hide Answer

You could not substitute the faster CL2 because our hold time constraint for R2 would not be met. The sum of the contamination delays between R1 and R2 must be greater than the hold time of R2. Using the faster CL2 would reduce the contamination delay sum to 1 which is not greater than the hold time, 2, of R2.

C. We've been treating wires as idealized components that introduce no delay of their own. In the real world, wires have resistance, capacitance and inductance that will cause different frequencies to propagate along the wire at different rates. This means that wires will delay the arrival of sharp rising and falling transitions (which you'll remember from Fourier analysis have signal components at many different frequencies). This effect is particularly bothersome in connection with clock signals since the clock may arrive at separate parts of the circuit at slightly different times. This difference in arrival times of the clock is called **clock skew**, which we'll model in our simple circuit above as explicit delays along each clock path:



D. Rewrite the timing specifications for the system as a whole taking into account d1 and d2. Don't make any assumption about the relative sizes of the two delays.

Hide Answer

The delay d1 causes all timing specifications associated with register R1 to be shifted later in time by d1. Likewise the delay d2 causes all timing specifications associated with register R2 to be shifted later in time by d2. Note that we still use the original clock signal as our reference for the system, so the setup times for the registers R1 and R2 become shorter by d1 and d2, respectively, and the hold times for R1 and R2 become longer by d1 and d2.

The timing specifications of the system, taking d1 and d2 into account, are:

tS = tPD,CL1 + tS,R1 - d1 = 6 - d1
tH = tH,R1 - tCD,CL1 + d1 = 1 + d1
tCD = tCD,R2 + d2 = 2 + d2
tPD = tPD,R2 + d2 = 8 + d2

Finally,

tCLK >= tPD,R1 + tPD,CL2 + tS,R2
tCLK >= 2 + d1 + 5 + 4 - d2
tCLK >= 11 + d1 - d2

E. The relative clock skew (d2 - d1) between two registers connected in a "pipeline" - where the output of the first register is connected, usually through logic, to the input of the second register - can also affect the design of a circuit. Explain how relative clock skew affects the maximum clock frequency of the circuit shown above. Remember that the relative skew might be positive or negative.

Hide Answer

As shown in part (C), tCLK >= 11 - (d2 - d1), rewritten to show the relative clock skew term.

One can see that as the relative clock skew becomes positive, the maximum clock frequency increases. Conversely, as the relative clock skew becomes negative, the maximum clock frequency decreases.

F. [Why clock skew keeps integrated circuit designers awake at night.] If d2 > d1, the circuit shown above will not operate correctly. Explain why. Will changing the frequency of CLK solve the problem? Why or why not?

Hide Answer

Let's revisit the constraint that the contamination delay from R1 to R2 must cover the hold time of R2:

tH,R2 <= tCD,R1 + tCD,logic
2 + d2 <= 1 + d1 + 1
d2 <= d1

Thus, if d2 > d1, then the hold time of R2 is no longer being satisfied. Lengthening the clock period doesn't change the fact that the hold time constraint isn't met. The clock period doesn't even enter our equations above.

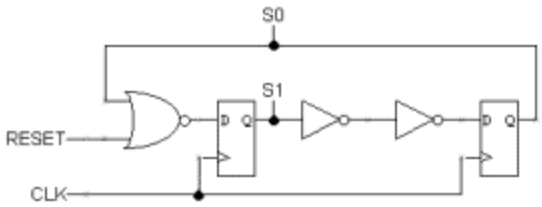
G. Suggest a way for the designer to change his circuit to guarantee correct operation given an upper bound, tskew > abs(d2 - d1), on the maximum relative clock skew. Assume that the timing parameters of the registers cannot be adjusted.

Hide Answer

Adding additional contamination delay, such as a buffer between R1 and R2, will

work. This leads to a circuit which has a larger clock period overall.

Problem 2. The following circuit diagram implements a FSM with two state bits, S0 and S1:



inverter: t_{CD}=1ns, t_{PD}=2ns
nor2: t_{CD}=1.5ns, t_{PD}=2ns
D register: t_{CD}=0ns, t_{PD}=2ns, t_H=1ns, t_S=3ns

A. ★ What is the smallest clock period for which the circuit still operates correctly?

Hide Answer

There are two contraints to check:

tPD,REG + tPD,INV + tPD,INV + tS,REG <= tCLK
tPD,REG + tPD,NOR2 tS,REG <= tCLK

The first constraint requires tCLK >= 9ns.

B. ★ A sharp-eyed student suggests optimizing the circuit by removing the pair of inverters and connecting the Q output of the left register directly to the D input of the right register. If the clock period could be adjusted appropriately, would the optimized circuit operate correctly? If yes, explain the adjustment to the clock period that would be needed.

Hide Answer

No, the circuit won't operate correctly since tCD,REG < tHOLD,REG, i.e., the output of the left register doesn't meet the required hold time when connected directly to the input of the right register.

C. ★ When the RESET signal is set to "1" for several cycles, what state is the FSM set to? (Give values for S0 and S1.)

Hide Answer

S0 = 0, S1 = 0.

D. ★ Assuming the RESET signal has been set to "0" and will stay that way, what is the state following S0=1 and S1=1?

Hide Answer

S0 = 1, S1 = 0.

E. ★ Now suppose there is *skew* in the CLK signal such that the rising edge of CLK always arrives at the left register exactly 1ns before it arrives at the right register. What is the smallest clock period for which the FSM still operates correctly?

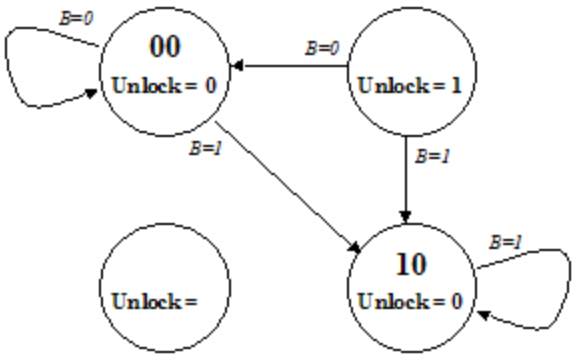
Hide Answer

Fortunately the skew doesn't introduce any hold time problems with the input to the right register. tCLK can now be as small as 8ns (both paths between registers fit exactly).

Problem 3. The ACME Company has recently received an order from a Mr. Wiley E. Coyote for their all-digital Perfectly Perplexing Padlock. The P3 has two buttons ("0" and "1") that when pressed cause the FSM controlling the lock to advance to a new state. In addition to advancing the FSM, each button press is encoded on the B signal (B=0 for button "0", B=1 for button "1"). The padlock unlocks when the FSM sets the UNLOCK output signal to 1, which it does whenever the last N button presses correspond to the N-digit combination.

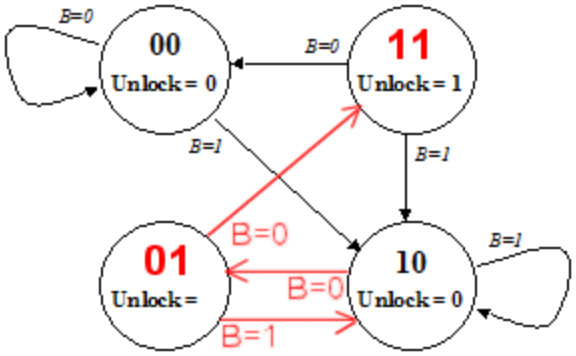
A. ★ Unfortunately the design notes for the P3 are incomplete. Using the specification above and clues gleaned from the partially completed diagrams below fill in the information that is missing from the state transition diagram with its accompanying truth table. When done

- each state in the transition diagram should be assigned a 2-bit state name S1S0 (note that in this design the state name is not derived from the combination that opens the lock),
- the arcs leaving each state should be mutually exclusive and collectively exhaustive,
- the value for UNLOCK should be specified for each state, and
- the truth table should be completed.



S_1	S_0	B	S'_1	S'_0	UNLOCK
0	0	0	0	0	
0	0	1	1	0	
0	1	0	1	1	0
0	1	1			0
1	0	0	0	1	
1	0	1			
1	1	0			1
1	1	1			1

Hide Answer



S_1	S_0	B	S'_1	S'_0	UNLOCK
0	0	0	0	0	0
0	0	1	1	0	0
0	1	0	1	1	0
0	1	1	1	0	0
1	0	0	0	1	0
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	1

B. What is the combination for the lock?

Hide Answer

100

Problem 4. Construct a "divisible-by-3" FSM that accepts a binary number entered one bit at a time, most significant bit first, and indicates with a light if the number entered so far is divisible by 3.

A. ★ Draw a state transition diagram for your FSM indicating the initial state and for which states the light should be turned on. Hint: the FSM has 3 states.

Hide Answer

If the value of the number entered so far is N, then after the digit b is entered, the value of the new number N' is 2N + b. Using this fact:

if N is 0 mod 3 then for some p, N = 3p + 0. After the digit b is entered, N' = 6p + b. So N' is b mod 3.

if N is 1 mod 3 then for some p, N = 3p + 1. After the digit b is entered, N' = 6p + 2 + b. So N' is b+2 mod 3.

if N is 2 mod 3 then for some p, N = 3p + 2. After the digit b is entered, N' = 6p + 4 + b. So N' is b+1 mod 3.

This leads to the following transition diagram where the states are labeled with the value of N mod 3.

B. ★ Construct a truth table for the FSM logic. Inputs include the state bits and the next bit of the number; outputs include the next state bits and the control for the light.

Hide Answer

S1	S0	b	S1'	S0'	light
0	0	0	0	0	1
0	0	1	0	1	1
0	1	0	1	0	0
0	1	1	0	0	0
1	0	0	0	1	0
1	0	1	1	0	0

C. Draw a logic schematic for the FSM.

Hide Answer

$light = \overline{S1}*\overline{S0}$

$S1' = \overline{S1}*S0*\overline{b} + S1*\overline{S0}*b$

$S0' = \overline{S1}*\overline{S0}*b + S1*\overline{S0}*\overline{b}$

Problem 5.

A. An FSM, M, is constructed by connecting the output of a 3-state FSM to the inputs of an 9-state FSM. M is then reimplemented using a state register with the minimum number of bits. What is the maximum number of bits that may be needed to reimplement M?

Hide Answer

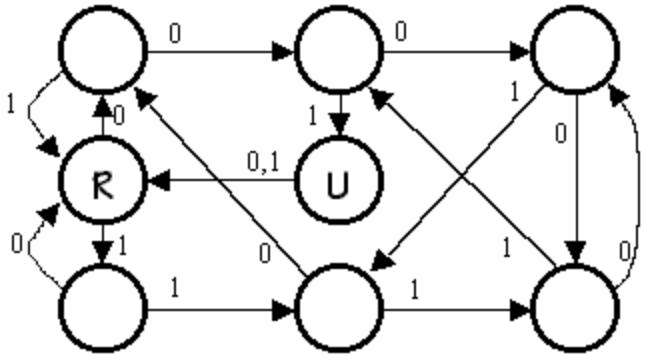
M has 27 states which require a total of 5 bits in the state register (not 2 + 4 bits!).

B. You connect M N-state FSMs, each have 1 input and 1 output, in series. What's an upper bound on the number of states in the resulting FSM?

Hide Answer

Each FSM can in theory be in one of its N states, so an upper bound on the number of states in the combined machine is N^M.

Problem 6. Ben Bitdiddle has designed an electronic lock with three buttons: "reset", "0" and "1". He has provided the following state transition diagram showing how the lock responds to a sequence of inputs.



The lock makes a transition from its current state to a new state whenever one of the three buttons is pressed and released. It ignores its inputs if more than one button is pressed. Pressing "reset" returns the lock to the state marked "R" in the diagram (arcs showing the transitions to the reset state have been omitted from the diagram to make it easier to read). Pressing "0" or "1" will cause the lock to follow the appropriately labeled transition from its current state. The lock opens if it reaches the state marked "U".

A. After pressing the "reset" button what is the length of the shortest sequence of button presses that will open the lock?

Hide Answer

3 button presses will open the lock: 0, 0, 1.

B. After pressing the "reset" button what is the length of the longest sequence of button presses that will cause the lock to open after the last button in the sequence is pressed but not open any earlier in the sequence?

Hide Answer

The longest such sequence is unbounded: any number of 0's followed by 111 or 1111 will cause the lock to open for the first time.

C. After much use, the "reset" button breaks. Is it still possible to open the lock using only the "0" and "1" buttons assuming you know nothing about the lock's state (except that its locked!) when you start?

Hide Answer

Yes. A sequence of 1's will open the lock. You have to try the lock after each press of "1" since a different number of 1's is required depending on the starting state.

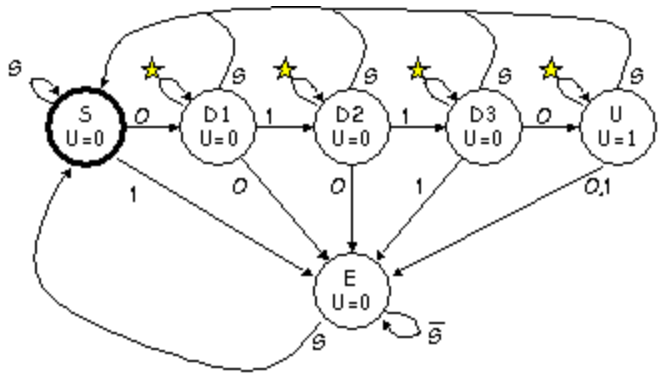
D. Suppose Ben wanted to design a lock that required exactly 10 button presses to open after pressing "reset". Not counting the "reset" and "unlock" states, what is the

minimum number of state his FSM would need need?

Hide Answer

His FSM would need 9 states in addition to "reset" and "unlock".

Problem 7. Stimulated by Tuesday's lecture, you have decided to cover MIT's steep tuition costs by selling simple digital locks based on the neat six-state FSM used as an example:



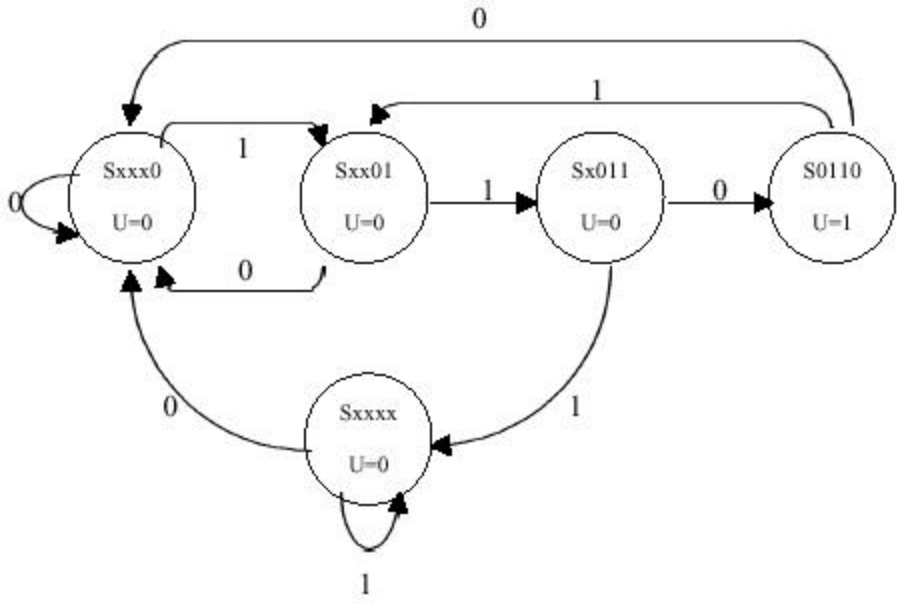
☆= no buttons pressed

Recall that this design has three buttons labeled "0", "1", and "Start", and generates an unlock signal U=1 when the user presses Start followed by the sequence 0,1,1,0.

Unfortunately your partner, Mark Ting, insists that the 6.004 design is way too complex for normal users to understand. After asking you to help figure out how to make his watch stop beeping ("I never could figure out how to operate this damned thing"), Mark questions the need for a Start button. If 0110 is the combination, he argues, why can't I just walk up and enter 0,1,1,0 and have the lock open? After some reflection, you conclude that he may have a point.

A. Design a FSM whose inputs are simply "0" and "1" buttons, whose output is the U (unlock) signal, and which has the property that U=1 if and only if the last four button presses correspond to the sequence 0,1,1,0. Show the state transition diagram corresponding to your design. [HINT: 5 states are sufficient].

Hide Answer



The name of each state represents how many digits in the sequence have been input. State Sxxxx indicates that the sequences has not begun, Sxxx0 indicates that the first 0 has been input, etc.

B. Is it possible that an equivalent FSM might be implemented in fewer than 5 states? Explain.

Hide Answer

Since 4 transitions are required for 4 button pushes, at least 5 states are needed to implement the FSM. Having only 4 states would make 3 the mininum number of transitions to the unlock state.

C. The flip flops used to hold your FSM state contain random values when power is first applied to your lock. Does this constrain your handling of unused states? Explain.

Hide Answer

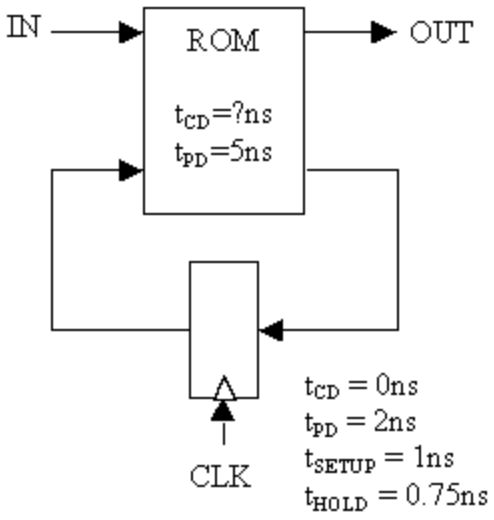
Since we have 5 states, 3 bits are required to encode the states, resulting in 3 unused states. If during power up it is possible to being in an unknown state, our FSM must include transitions from unknown states to known states. If the machine beings in an unknown state and a 0 in input, we should transition to state Sxxx0; if a 1 is input, we should transition to Sxxxx.

D. In a table (similar to that shown in lecture), give the contents of a ROM that might be used in an implementation of your design. Completely specify the ROM contents, including those corresponding to unused states.

Hide Answer

Current State	Input	Next State	Output
000	0	001	0
000	1	000	0
001	0	001	0
001	1	010	0
010	0	011	0
010	1	001	0
011	0	100	0
011	1	000	0
100	0	001	1
100	1	010	1
101	0	001	0
101	1	000	0
110	0	001	0
110	1	000	0
111	0	001	0
111	1	000	0

Problem 8. A possible implementation of a finite state machine with one input and one output is shown below.



A. If the register is N bits wide, what is the appropriate size of the ROM? Give the number of locations and the number of bits in each location.

Hide Answer

2^{N+1} locations of N+1 bits.

B. If the register is N bits wide, what is the least upper bound on the number of states in a FSM implemented using this circuit?

Hide Answer

2^N

C. What is the smallest value for the ROM's contamination delay that ensures the necessary timing specifications are met?

Hide Answer

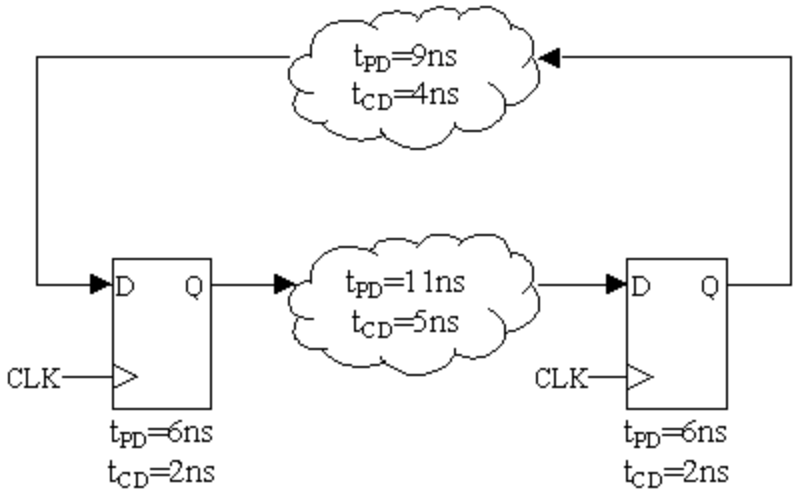
tCD,REG + tCD,ROM >= tH,REG, so tCD,ROM >= 0.75ns.

D. Assume that the ROM's tCD = 3ns. What is the smallest clock period that ensures that the necessary timing specifications are met.

Hide Answer

tPD,REG + tPD,ROM + tS,REG = 2 + 5 + 1 = 8ns.

Problem 9. The following schematic has two flip-flops and two blocks of combinational logic with the indicated timing specifications. Assume that the flip-flops are identical and that the clock has zero rise and fall time.



A. Assuming that the clock period is 25ns, what is the maximum setup time for the flip-flops for which this circuit will operate correctly?

Hide Answer

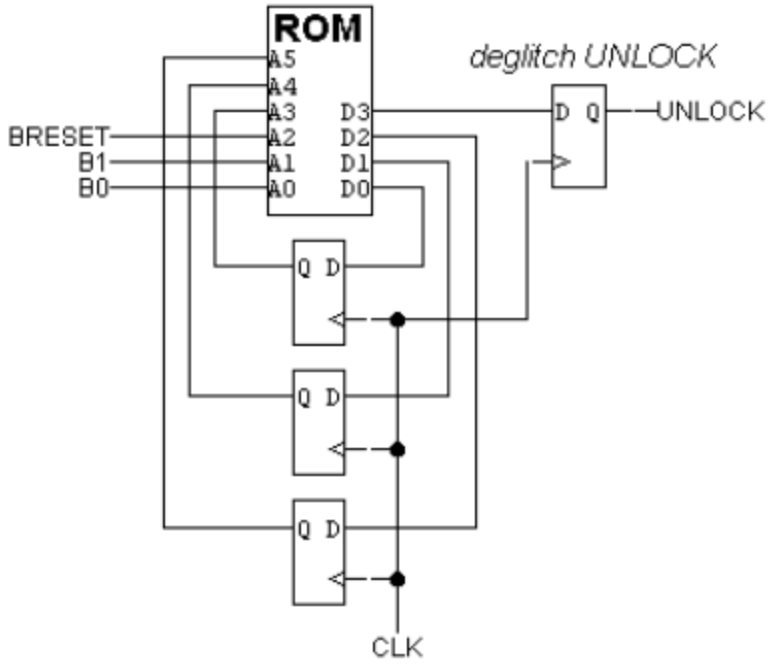
tS <= tCLK - tPD,REG - tPD,LOGIC = 25 - 6 - max(11,9) = 8ns.

B. Assuming that the clock period is 25ns, what is the maximum hold time for the flip-flops for which this circuit will operate correctly?

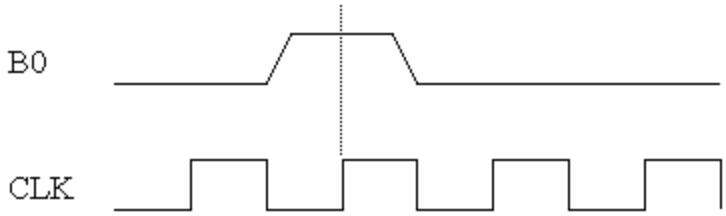
Hide Answer

tH <= tCD,REG + tCD,LOGIC = 2 + min(4,5) = 6ns.

Problem 10. Ben Bitdiddle has designed an electronic lock with three buttons: "Breset", "B0" and "B1". He has provided the following circuit diagram showing how the lock is implemented from a ROM and 3 flip-flops.



The button circuitry converts each button press into a single pulse guaranteed to be stable the required amount of time before and after the rising edge of the clock. For example, pressing "B0" once produces the following waveform:



In answering the questions below, assume that the value of the UNLOCK output is only a function of the current state.

A. What is the total number of bits in the ROM?

Hide Answer

256 bits total: 2^6 locations, 4 bits wide.

B. The timing specifications for components are:

ROM: tCD=3ns, tPD=11ns

D flip-flop: tCD=2ns, tPD=4ns, tS=3ns, tH=3ns

How long before the rising edge of CLK must the button circuitry guarantee that the button signals are stable?

Hide Answer

tPD,ROM + tS = 14ns.

C. Assume that all combinations start with pressing the "Breset" button. Ben wants to program the lock with the longest possible combination. Not counting the "Breset" button press, what is the longest combination Ben can achieve?

Hide Answer

7 assuming no looping combinations.

D. If the lock is programmed not to change state if no buttons are pressed, what is the next state field of ROM location 48 (i.e., the location corresponding to A5,A4,A3,A2,A1,A0 = 110000)?

Hide Answer

The current state appears on A5,A4,A3 = 110. So we want the next state field of the ROM (D2,D1,D0) to specify the same state = 110.

E. The following table shows one possible contents of the first 32 locations of the ROM; assume that all other locations have the value "0010". The location is listed as A5,A4,A3,A2,A1,A0, the data is listed as D3,D2,D1,D0.

Location	Data
000000	0000
000001	0011
000010	0000
000011	0000
000100	0010
000101	0010
000110	0010
000111	0010
001000	1001
001001	1001
001010	1001
001011	1001
001100	1010
001101	1010
001110	1010
001111	1010

Location	Data
010000	0010
010001	0011
010010	0000
010011	0010
010100	0010
010101	0010
010110	0010
010111	0010
011000	0011
011001	0010
011010	0001
011011	0011
011100	0010
011101	0010
011110	0010
011111	0010

If the lock is programmed with this ROM data, what happens when "B0" and "B1" are pressed at the same time? Assume that "Breset" is not pressed.

Hide Answer

State stays the same since all addresses of the form XXX011 transition to state XXX.

F. If the lock is programmed with this ROM data, what is the shortest combination that opens the lock after "Breset" has been pressed?

Hide Answer

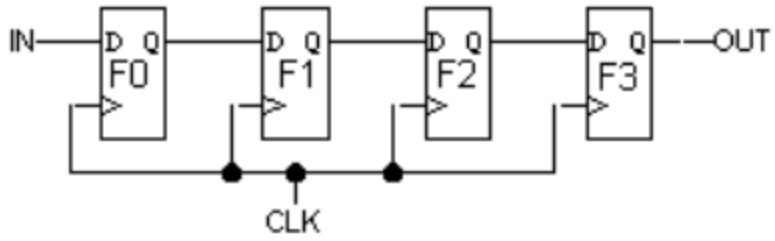
press B0, then B1.

G. Suppose that the "Breset" button breaks while the lock is locked. Is it still possible to open the lock using a predetermined sequence of presses of the "B0" and "B1" buttons? Assume you know nothing about the lock's state (except that it's locked!) when you start.

Hide Answer

Yes, you can open the lock. Noting that the UNLOCK state loops to iteself, B1-B0-B1 is one of many sequences that takes us from any state to UNLOCK.

Problem 11. Use the following circuit in answering the questions below.



Each of the edge-triggered D flip-flops has a setup time of tS, a hold time of tH, a propagation delay of tPD and a contamination delay of tCD. Assume that IN is stable tS before the rising edge of CLK and tH after the rising edge of CLK.

A. In order for the circuit shown above to operate correctly what constraints on tH and tS are necessary? Express them in terms of tCD, tPD and the clock period.

Hide Answer

ensure hold time is met at each register: tH <= tCD

B. What is the minimum clock period at which this circuit can be clocked and still be guaranteed to work? Express your answer in terms of tH, tS, tCD and tPD. Assume that timing constraints that do not depend on the clock period are met.

Hide Answer

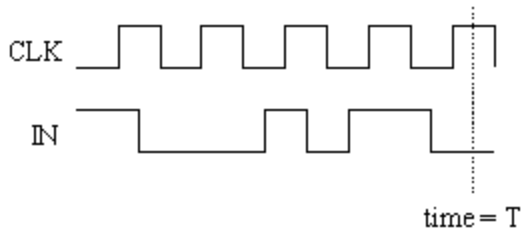
ensure setup time is met at each register: tPD + tS <= tCLK

C. For just this question suppose there is skew in the CLK signal such that the rising edge of CLK arrives at the flip-flop labeled F1 1ns before it arrives at the other three flip-flops. Assume that hold times are not violated. How does this change the minimum clock period at which the circuit above can be clocked and still be guaranteed to work?

Hide Answer

The minimum clock period increases by 1ns, i.e., we have to have an extra 1ns between clock edges to ensure that the setup time at F1 is met.

D. Consider following waveform plot for the circuit above. Assume that IN is stable tS before the rising edge of CLK and tH after the rising edge of CLK and that time T is more than tPD after the preceding rising edge of CLK.



What is the value of OUT at time T?

Hide Answer

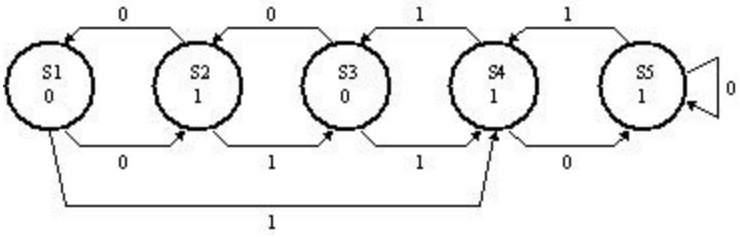
At time T, OUT = 0 (ie, the value of IN four clock edges earlier).

E. View the circuit above as an FSM with one input and one output. How many non-equivalent states does it have?

Hide Answer

4 bits of state give us 2^4 = 16 states.

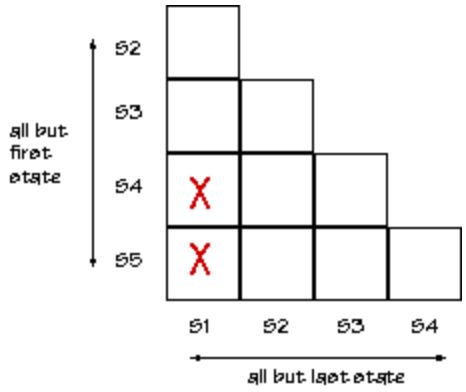
Problem 12. Consider the following FSM state transition diagram:



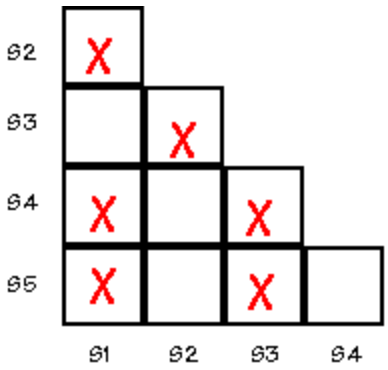
Let's see if there is an equivalent state machine with fewer states by checking to see if any

states in the diagram above are equivalent. Two states are equivalent if (1) they have identical outputs and (2) for each possible combination of inputs they transition to equivalent states.

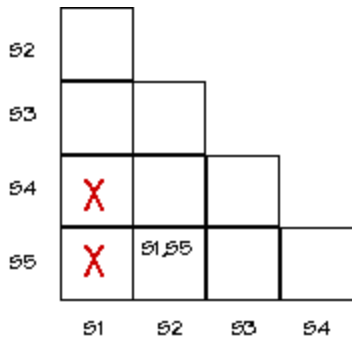
A. Start by filling in a "compatibility table" like the one shown below. Place an "X" in square (SI,SJ) if SI produces a different output from SJ.



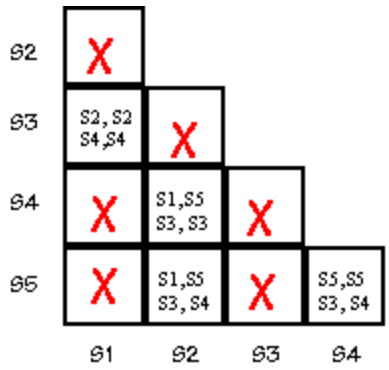
Hide Answer



B. For each non-X square (SI,SJ) write in pairs of states that have to be equivalent in order for SI and SJ to be equivalent. For example, for S2 to be equivalent to S5, then S1 (where S2 goes with a "0" input) has to be equivalent to S5 (where S5 goes with a "0" input).

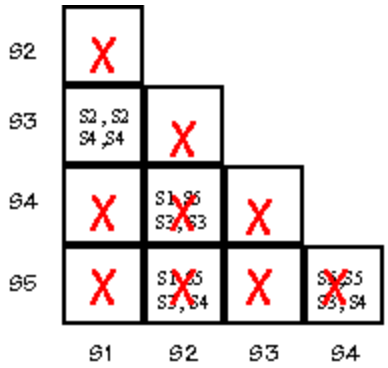


Hide Answer



C. Finally, look at an entry (SI,SJ). If entry is "SM,SN" and if (SM,SN) has an "X", put an "X" in square (SI,SJ). Repeat until no more squares can be X'ed out. The remaining squares indicate equivalent states. Show the final state (no pun intended) of your compatibility table.

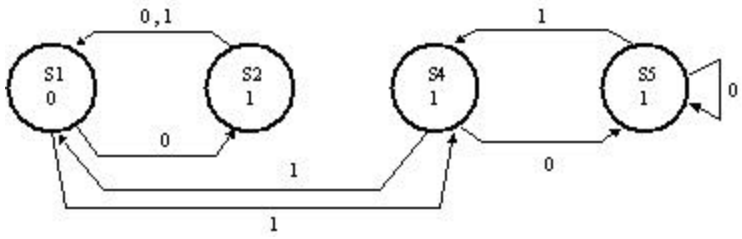
Hide Answer



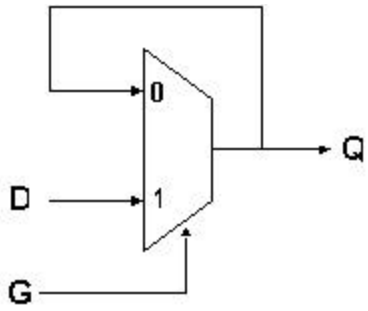
D. Draw the state transition diagram for the simplified FSM.

Hide Answer

Here's the state transition diagram for the simplified FSM (w/o state 3).



Problem 13. In lecture, you saw a static latch constructed from a 2-input lenient MUX as shown in the diagram below.



Recall that the MUX selects the Q output when G=0, and the D input when G=1. The plan is that when G=1, the Q output will follow D after a short delay; when G=0, the current Q output will be "latched" via the feedback path. In this problem we explore assumptions necessary to construct an informal proof that the latch behaves as proposed. Assume, in each of the following, that the MUX is a well-behaved lenient combinational device with a propagation delay of tpd.

Recall that the lenience of the MUX allows us to assume that if any two of its inputs sufficient to determine its output are stable and valid for at least tpd, then the MUX output will be stable and valid.

A. Specify constraints on the two data inputs of the MUX sufficient to guarantee that its output will be stable and valid independently of the value on the select input.

Hide Answer

When both data inputs are equal, valid and stable for tpd, the output is valid and stable independent of the select signal.

B. Specify constraints on a single data input and the select input of the MUX sufficient to guarantee stable and valid output independently of the value on the remaining data input.

Hide Answer

When G is 0 and both G and Q are stable and valid for tpd or when G is 1 and both G and D are stable and valid for tpd, the MUX will have a valid and stable output indenpedent of the other data input.

C. Now we explore the scenario where G=1 and D has been stable and valid for ts seconds prior to a 1-to-0 transition on G, and remains stable and valid until th seconds after the transition on G. Our goal is to establish that, for sufficiently large values of ts and th, the latch behaves as advertised.

Consider the initial interval in the above scenario where G=1. At what point is Q=D guaranteed? Explain why in a sentence or two.

Hide Answer

After G and D have been stable and valid for tpd, the output Q will equal the data input D.

D. Explain why, for sufficiently large values of tS the output Q remains stable despite invalid voltages on G. What is the setup time required to guarantee output validity during the transition on G?

Hide Answer

Part C tells us that Q will equal D tpd after D becomes valid. According to the leniency requirement, the output will be guaranteed valid and stable tpd after the inputs necessary to determine the output become valid and stable. In this case, both Q and D must be valid and stable tpd in order to ensure Q remains stable and valid independent of transitions on G. Thus tS must be greater than 2*tpd.

E. Now assume that D changes tH seconds after the transition on G. Explain why, for a sufficiently large value of tH, the Q output will remain stable independently of D.

Hide Answer

In order to ensure the Q remains stable and valid independent of D, G must be set to a valid logic "0" for tpd before D transitions. Thus, tH must be greater than tpd.

F. Identify which of your previous answers is dependent on the MUX being lenient, giving a single-sentence description of the dependence.

Hide Answer

All of the previous answers are dependent on the MUX being lenient. In each case we discuss the necessity for two inputs to be valid and stable for tpd in order to ensure the output is valid and stable. If the MUX were not lenient, all three inputs would have to be valid and stable for tpd in order for the output to be guaranteed valid and stable.

G. Does the operation of the above latch depend on the contamination delay of the MUX? Explain.

Hide Answer

The latch does not depend on contamination delay. It only depends on the leniency of the MUX and that tS and tH requirements are met.

H. Your analysis has established setup and hold time requirements necessary to guarantee proper operation of the latch. Suppose, in the above scenario, the setup or hold time requirement is violated? What can you say about the value on Q?

Hide Answer

If the setup and hold times are violated, the value of the output cannot be determined.