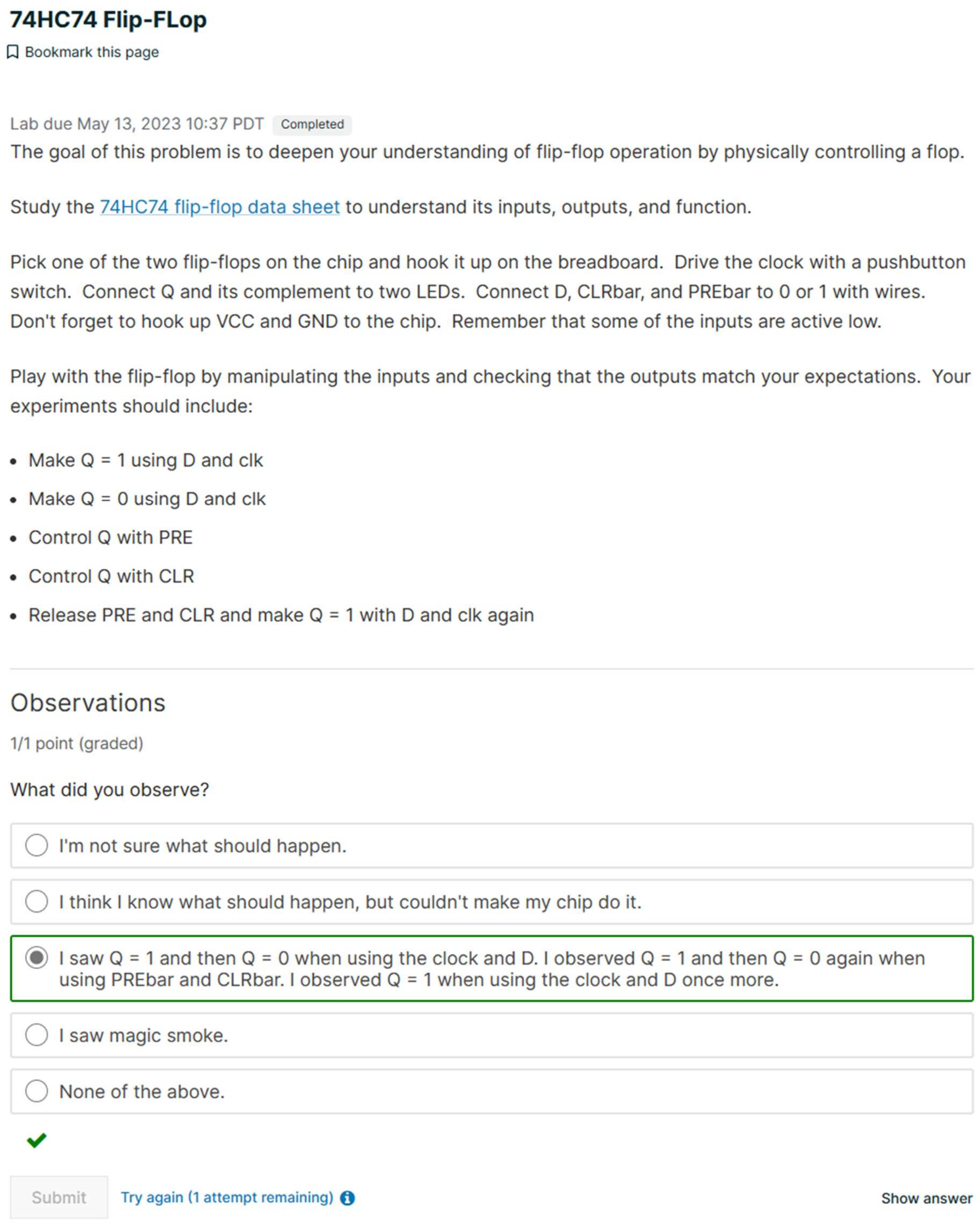
❖ Lab 3: Structural FSM -Tail Lights



74HC74 Flip-FLop

Bookmark this page

Lab due May 13, 2023 10:37 PDT Completed

The goal of this problem is to deepen your understanding of flip-flop operation by physically controlling a flop.

Study the 74HC74 flip-flop data sheet to understand its inputs, outputs, and function.

Pick one of the two flip-flops on the chip and hook it up on the breadboard. Drive the clock with a pushbutton switch. Connect Q and its complement to two LEDs. Connect D, CLRbar, and PREbar to 0 or 1 with wires. Don't forget to hook up VCC and GND to the chip. Remember that some of the inputs are active low.

Play with the flip-flop by manipulating the inputs and checking that the outputs match your expectations. Your experiments should include:

- Make Q = 1 using D and clk
- Make Q = 0 using D and clk
- Control Q with PRE
- Control Q with CLR
- Release PRE and CLR and make Q = 1 with D and clk again

Observations

1/1 point (graded)

What did you observe?

- ○ I'm not sure what should happen.
- ○ I think I know what should happen, but couldn't make my chip do it.
- [✓ I saw Q = 1 and then Q = 0 when using the clock and D. I observed Q = 1 and then Q = 0 again when using PREbar and CLRbar. I observed Q = 1 when using the clock and D once more.]
- ○ I saw magic smoke.
- ○ None of the above.

✓

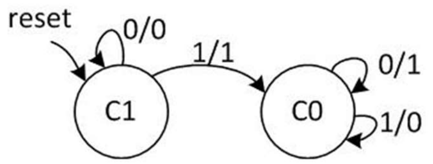
Submit Try again (1 attempt remaining) Show answer

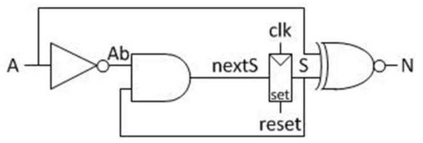
Twos Complementer

Bookmark this page

Lab due May 13, 2023 10:37 PDT Completed

Recall the Serial Two's Complementer you designed as a Mealy FSM in the Chapter 3 practice problems. The FSM takes a binary number one bit at a time, starting with the least significant bit, and puts out its negative (two's complement), one bit at a time. The state transition diagram and circuit are shown below.





The goal of this lab problem is for you to code the Mealy FSM in structural SystemVerilog. Use only flip-flops, AND, OR, and NOT gates in your solution.

Start with the files below. Modify the twoscomp module to describe the schematic above. Notice that SystemVerilog code for an ordinary flip-flop and for asynchronously resettable and settable flip-flops are provided; you will need to instantiate one of these for your state register. The test bench takes input 0010 and produces an expected output of 1110. Don't change the testbench or test vectors because that would mess up the hash.

Simulate your design and debug any discrepancies.

twoscomp.sv

twoscomp.tv

Hash

1/1 point (graded)

What hash did your testbench produce? Enter the hash as a 2-digit hexadecimal number with lowercase letters where applicable.

0e ✓

Submit Try again (1 attempt remaining) Show answer

Another Hash

1/1 point (graded)

What hash did your testbench produce with your new test vector? Enter the hash as a 2-digit hexadecimal number with lowercase letters where applicable.

08 ✓

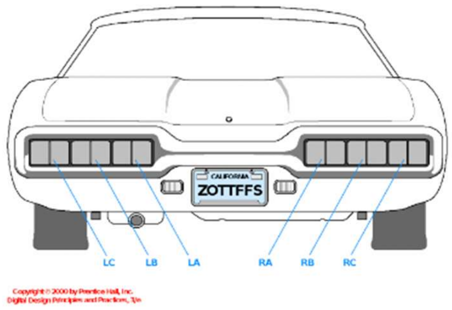
Submit Try again (1 attempt remaining) Show answer

Tail Light FSM

Bookmark this page

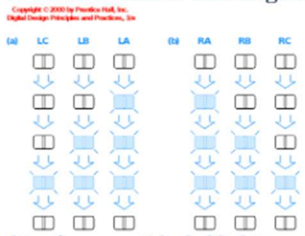
Lab due May 13, 2023 10:37 PDT Completed

The 1965 Ford Thunderbird is noted for its spiffy tail light turn signal sequence. There are three lights on each side that operate in sequence to indicate the direction of a turn. Figure 1 shows the tail lights and Figure 2 shows the flashing sequence for (a) left turns and (b) right turns. (This material is derived from an example by John Wakerly from the 3rd Edition of *Digital Design*.)



ZOTTFFS
LC
LB
LA
RA
RB
RC
Copyright © 2000 by Prentice Hall, Inc. Digital Design Principles and Practices, 3/e

Figure 1. Thunderbird Tail Lights



Copyright © 2000 by Prentice Hall, Inc. Digital Design Principles and Practices, 3/e
(a)
LC
LB
LA
(b)
RA
RB
RC

Figure 2. Flashing Sequence (shaded lights are illuminated)

The goal of this lab problem is for you to design a gate-level implementation of a turn signal Moore FSM, model it with structural SystemVerilog, and debug your design. Use only flip-flops, AND, OR, and NOT gates in your solution. (You may also use buffers, called "buf" in SystemVerilog, if you want to create another signal with a different name but the same value as a signal you already have.) You may need to copy a flip-flop module from elsewhere, and then instantiate it in your FSM multiple times for each flop in your system.

Start with the files below. Modify the lightfsm module but don't change the testbench or test vectors because that would mess up the hash. If you have questions about the desired timing of your FSM, see the test vector file for expectations.

lightfsm.sv

lightfsm.tv

***Hint**: Take a methodical approach to design:*

- *Sketch a state transition diagram (please take the time to do this, even though it's not computer-gradable)*
- *Choose state encodings*
- *Write the next state and output logic*
- *Write Boolean equations for next state in terms of current state and inputs*
- *Write Boolean equations for outputs in terms of current state*
- *Sketch a schematic implementing the equations*
- *Write structural SystemVerilog matching your schematic*
- *Simulate your design with the test bench provided*
- *Debug any discrepancies*

Hash

1/1 point (graded)

What hash did your testbench produce? Enter the hash as a 2-digit hexadecimal number with lowercase letters where applicable.

21 ✓

Submit Try again (1 attempt remaining) Show answer