Exp 11: A Simple Data Latching and Display Circuit

Objectives:

- To learn about one of the basic sequential circuits in digital design: the flip-flop
- To learn about one of the most common concepts in digital design: memory
- To learn about how Verilog models memory

Background: The D flip-flop is the basic memory storage element in digital design. The notion is that whenever you refer to memory of any type in digital design, you can generally model that memory as a given number of D flip-flops. This experiment aims to get you used to working with D flip-flops by presenting a not overly complicated experiment. In reality, this experiment is similar to a combinatorial logic experiment you did previously, but the notion of memory has been added.

The notion of combinatorial circuits was evident in previous circuits based on the notion that when you moved switches up and down, the display reacted accordingly. In this experiment, you'll use eight D flip-flops to store data. In this way, the data being displayed by the seven segment displays is the data output from the eight D flip-flops, rather than the data output from the switches. Keep in mind, that because D flip-flops are synchronized to a clock edge, you'll need to control when data will be stored by the D flip-flops.

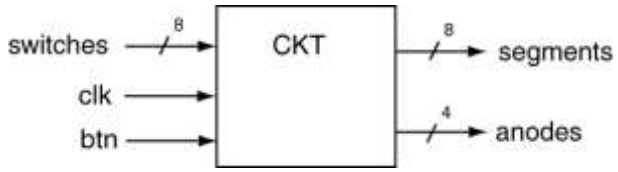


Figure 1: The top-level block diagram for this experiment.

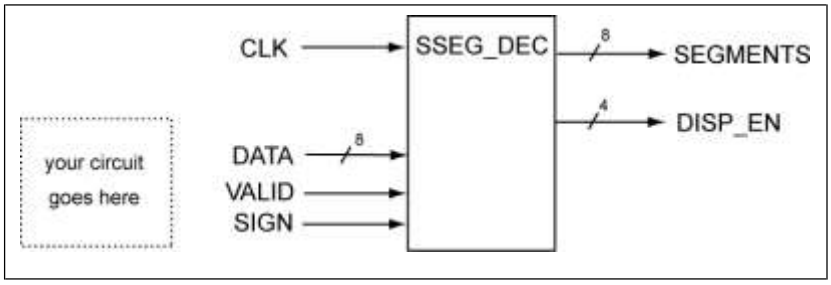


Figure 2: The connections required for the provided modules.

Assignment: Use the sseg_dec.v module to display the output of eight D flip-flops. The eight values are in 8-bit binary and displayed accordingly on the four seven-segment displays. The inputs to the eight D flip-flops are output from the eight switches on the development board; the flips flops latch the switch values when the left-most button is pressed.

Figure 16 shows the highest-level block diagram for this experiment. Your mission is to have a button press latch the data into the D flip-flops, interpret the output of the D flip-flops as an 8-bit number, and display that number on the seven-segment displays.

Questions:

1. When you press the button, how many times is the data effectively "latched" in the associated D flip-flops? HINT: realize that the clock input from the development board operates at 100Mhz.

2. Knowing what you know about Verilog, do you think there is an easier approach to designing the set of eight flip-flops? Briefly speculate and explain.

Exp 12: T Flip-Flop Based Four-Bit Counter

Objectives:

- To learn about one of the basic sequential circuits in digital design
- To gain more practice modeling sequential circuits using Verilog
- To gain more experience designing a common sequential circuit: the counter

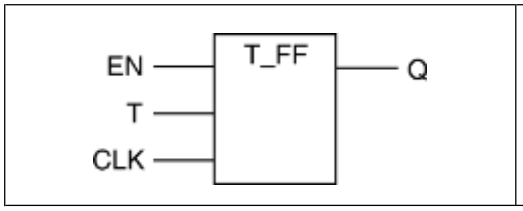
Background: A counter is yet another standard digital design element. Although there are many different ways to implement a counter, in this lab activity, you'll implement the counter using four T flip-flops. This lab activity should be done in stages so as to ensure your final design will work. The three stages are listed below. You're only responsible for demonstrating the final stage.

1) Design a T flip-flop.
2) Design, simulate, and test a 4-bit binary counter using four T flip-flops and structural modeling.
3) Visually test the 4-bit counter by using it to drive a SSEG_DEC module.

Stage 1: Implementation of a T Flip-Flop

Design a rising-edge triggered (RET) T flip-flop in Verilog ***by entering the characteristic equation of the T flip-flop*** using Verilog syntax. This flip-flop should contain an enable input that controls when and if the flip-flop will toggle its present state. A description of the T flip-flop is provided in Figure 2. Note that the simulator often times generates an unknown output for sequential circuits such as flip-flops. If this is the case, assign an initial value to your intermediate signal as follows:

```
reg t_Q = 1'b0;
```



EN	CLK	T	Q^+
0	-	-	Q
1	↑	0	Q
1	↑	1	$\overline{Q}$

Figure 2: Schematic diagram and operational characteristics of the T flip-flop.

Stage 2: 4-bit Counter with Enable

Using the T flip-flop you designed in Stage 1, design a 4-bit counter. A block diagram of this counter is shown in Figure 3. This counter should contain an enable input, which when asserted, allows the counter to count in binary increments. The output of the counter increments on the rising edge of the clock signal. If the enable input is not asserted, the counter does not change state (meaning it does not count). Implement the 4-bit counter using Verilog structural modeling using the individual T flip-flops as the components. Test your counter using the simulator to verify its proper operation.

NOTE: This circuit will require additional logic to implement. In determining the necessary logic, keep in mind that the bit at position will toggle only when all lower-order bits are '1' and the enable signal is asserted. There are multiple ways to design this counter.

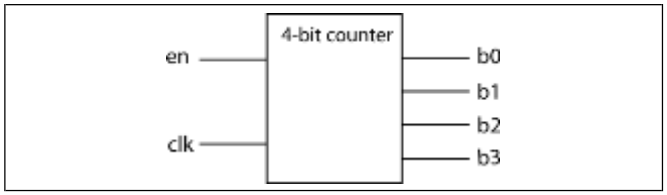


Figure 3: Block diagram of the 4-bit counter.

Stage 3: Visual Testing of 4-Bit Counter

1. Using Verilog structural modeling, create a new circuit comprised of the 4-bit counter designed in Stage 2 and the standard SSEG_DEC module. Be sure to connect the VALID and SIGN inputs accordingly; Figure 4 shows most of the proper connections.

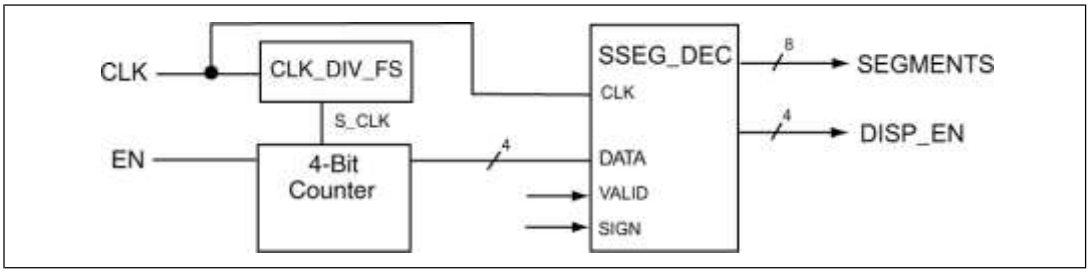


Figure 4: The circuit used for visually testing the 4-bit counter.

Special Deliverables:

1. Be sure to include a screen shot of your iSim waveform for this design.

Questions:

1. How many flip-flops did this FSM design use?

2. Without thinking about it, this counter simply "rolled over" when the highest count was reached. What was it that made it magically roll over? Briefly explain.

3. Describe the changes that would be required to implement this counter using JK flip-flops instead of T flip-flops.

4. This counter is referred to as an "up counter" (counts from low binary number to high binary numbers). Describe the changes that would be required in order to make it a "down counter".