Counters

Christopher Hunt

Objectives

The objective of this lab is to design and implement a counter system that utilizes T flip-flops to effectively slow down a 50 MHz clock frequency to achieve a frequency as close to 1 Hz as possible. The DE10-Lite will be utilized to drive all six digits of its seven-segment displays by outputting a counter to each digit, synchronized with the clock signal. This lab aims to enhance our knowledge and skills in digital logic design by introducing the concept of state memory and utilizing the FPGA's clock as a means of logic control. By successfully completing this lab, we will deepen our understanding of digital circuits and gain practical experience in implementing clock-controlled systems.

Equipment

- Quartus Prime Lite Edition V. 18.0/18.1
- DE10-Lite kit with MAX10 10M50DAF484C7G FPGA
- USB to USB-B cable
- The ECE 271 textbook, Digital Design and Computer Architecture by Drs. David and Sarah Harris

---

Design

In the previous lab we developed a logic design that mapped a 4-bit binary number that was inputed via on board switches to a seven segment display which displayed the binary number in hexidecimal. This will be used as a basic building block for the design of a clock-controlled counter (Appendix 1).

T Flip Flops will be used to divide the clock signal. When feeding a clock signal into a TFF the output will effectively output at half the frequency of the input clock. By cascading four of these TFF's we are able to output a four bit counter, the least significant bit being the output clock of the first TFF and the most significant bit being the output clock of the fourth. These outputs will be mapped to the 6 seven-segment displays available on the board, the fastest digit on the right and the 1 Hz digit on the left.

Clock	**Reset**	**T**	**Q**
X	0	X	0
1	1	0	Q
1	1	1	$\sim Q$
1	1	X	Q

Table 1: T Flip-Flop Truth Table

In order to achieve this we need to pay close attention to the amount of TFF's we use. Each TFF is acts as a division by two to the input clock speed. For this implementation we will be using the DE110-Lite's 50 MHz clock. To find the number of TFF's needed to achieve a frequency of near 1 Hz we count how many times we need to divide 50,000,000 until we are as close to 1 as possible. The closest being 26 total TFF's. This should give us a counter frequency of approximately 0.75 Hz(fig. 1).

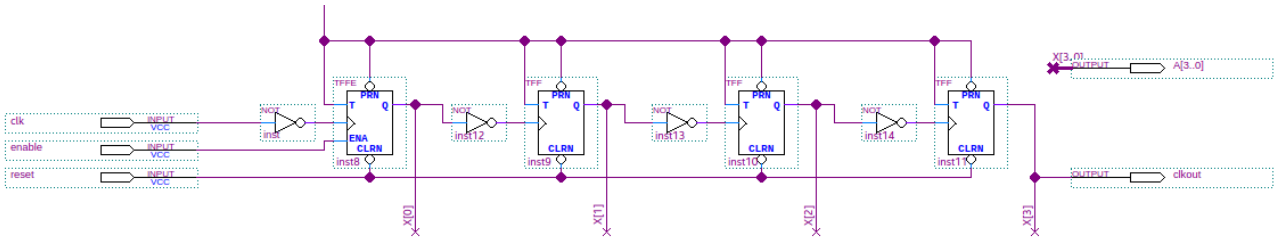

Figure 1: T Flip Flop Counter Design

We know that there will be at least 24 flip flops used to output to the 6 seven-segment displays. This means there must be included some extra TFF's at the beginning of the schematic to act as a divider (fig. 2). When implementing this divider it was built with two TFF's but upon testing the speed of the final digit was faster than 1 Hz. Three more TFF's were added to slow down the clock to better match the desired frequency.

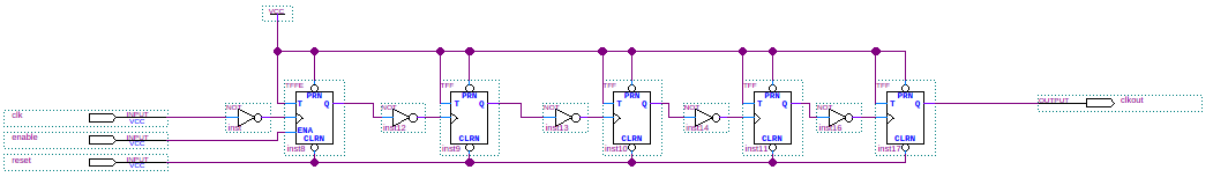

Figure 2: T Flip Flop Clock Divider Design

These two schematics will act as the basic building blocks for the finished schematic. Once these have been exported as Symbol Files the 6 counters will be cascaded in front of the divider with the pins mapped to the seven-segment displays (fig. 3). See tables below for pin mapping.

Signal	PIN
clk	PIN_P11
enable	PIN_C10
reset	PIN_C11

Signal	PIN
sa0	PIN_C14
sa1	PIN_C18
sa2	PIN_B20
sa3	PIN_F21
sa4	PIN_F18
sa5	PIN_J20

Signal	PIN
sb0	PIN_E15
sb1	PIN_D18
sb2	PIN_A20
sb3	PIN_E22
sb4	PIN_E20
sb5	PIN_K20

Signal	PIN
sc0	PIN_C15
sc1	PIN_E18
sc2	PIN_B19
sc3	PIN_E21
sc4	PIN_E19
sc5	PIN_L18

Signal	PIN
sd0	PIN_C16
sd1	PIN_B16
sd2	PIN_A21
sd3	PIN_C19
sd4	PIN_J18
sd5	PIN_N18

Signal	PIN
se0	PIN_E16
se1	PIN_A17
se2	PIN_B21
se3	PIN_C20
se4	PIN_H19
se5	PIN_M20

Signal	PIN
sf0	PIN_D17
sf1	PIN_A18
sf2	PIN_C22
sf3	PIN_D19
sf4	PIN_F19
sf5	PIN_N19

Signal	PIN
sg0	PIN_C17
sg1	PIN_B17
sg2	PIN_B22
sg3	PIN_E17
sg4	PIN_F20
sg5	PIN_N20

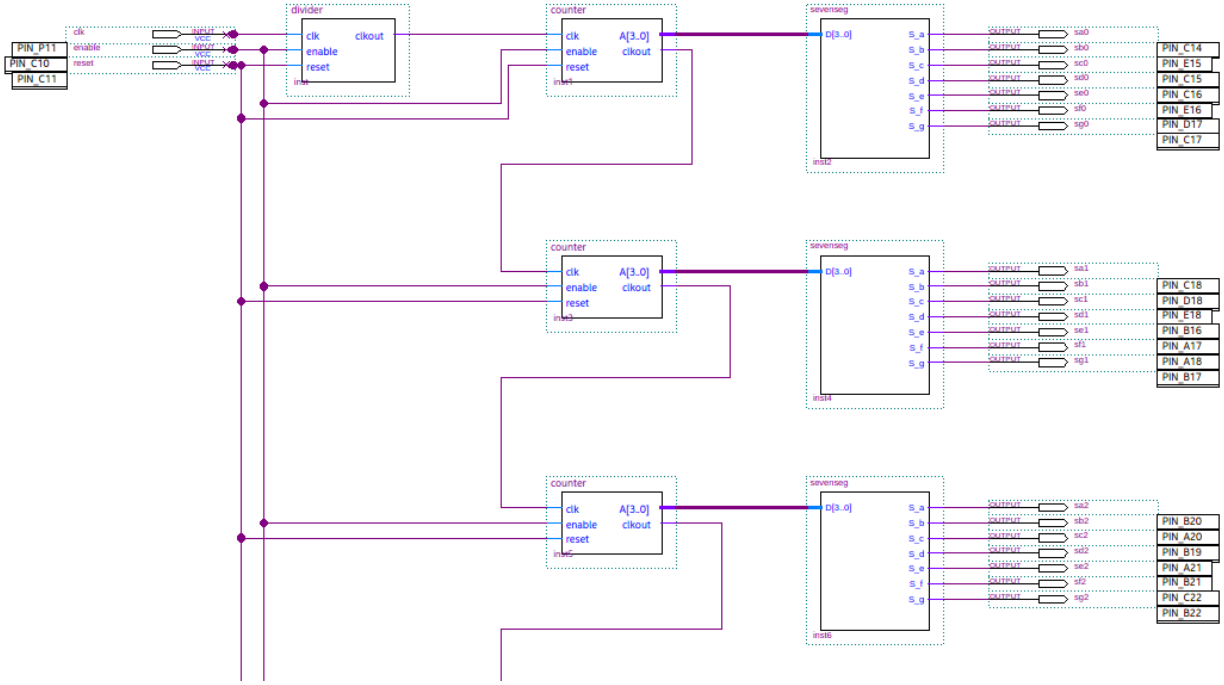

Figure 3: Partial View of T Flip Flop Counter Design

Once completed, the design is compiled and then Verilog files were exported. These were then used in ModelSim to verify the that the design works as anticipated.

Simulation

For the simulation, the Quartus project files were exported as Verilog files and the lab was simulated in ModelSim (fig. 4). Once the design has passed simulation, the counter will be programmed onto the board.

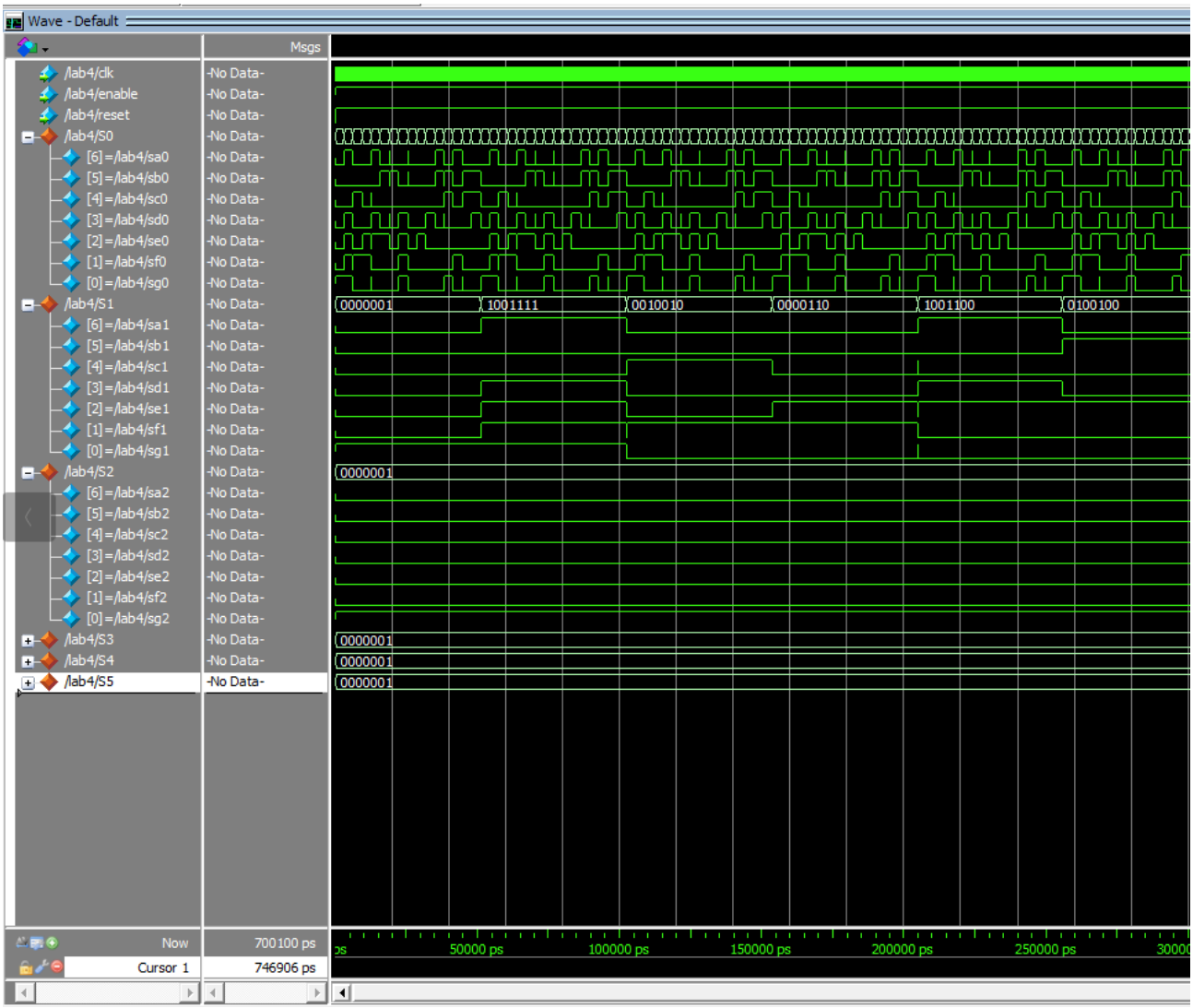

Figure 4: Counter Design Simulation

Implementation

The DE10-Lite was programmed successfully with the counter design. In it's implementation the slowest counter nearly reached a frequency of 1 Hz.

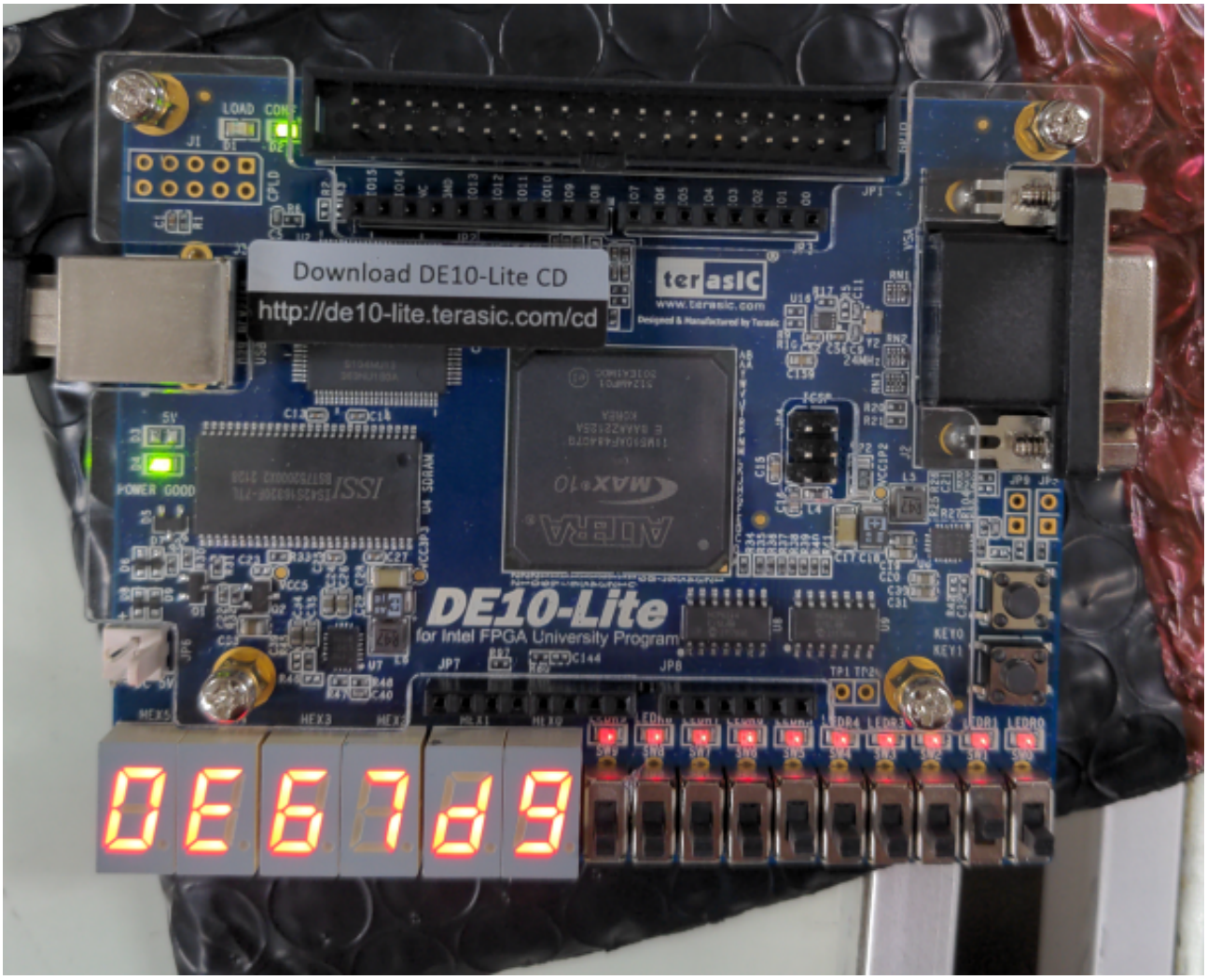

Figure 5: Counter Design Implementation

Observations

In this lab the most challenging aspect was acquiring precise timing. When first implementing the design it was calculated that 26 total TFF's would be needed to achieve 1 Hz. The divider initially used had 2 TFF's which, with the 6 cascaded 4 TFF counter, would have met that total. When implemented the counters frequency was faster than 1 Hz. After adding 3 more, the frequency reached it's closest value.

Conclusion

In this lab, the objective was to design and implement a counter system using T flip-flops to slow down a 50 MHz clock frequency to achieve a frequency as close to 1 Hz as possible. The DE10-Lite board was used to drive six digits of seven-segment displays by outputting a counter to each digit synchronized with the clock signal.

The lab aimed to enhance knowledge and skills in digital logic design, introduce the concept of state memory, and gain practical experience in implementing clock-controlled systems.

The design involved cascading T flip-flops to create a counter. Each T flip-flop acted as a division by two to the input clock speed. By carefully selecting the number of flip-flops used, a frequency close to 1 Hz was achieved. The design was implemented using Quartus Prime and verified through simulation in ModelSim. The final design was successfully programmed onto the DE10-Lite board.

During the implementation, precise timing was a challenge. Initially, the chosen divider did not produce the desired frequency, and additional flip-flops were added to adjust the timing. The implemented counter achieved a frequency nearly reaching 1 Hz.

In conclusion, this lab provided a practical understanding of digital circuits and the implementation of clock-controlled systems. The objectives of enhancing knowledge and skills in digital logic design were successfully met. The use of T flip-flops proved effective in dividing the clock frequency, and the design was implemented and verified.

Study Questions

1. Why did the counter count down before the inverters were added to the design? Answer this with respect to the operation of the Flip flop.

The output of the T Flip Flop is opposite the clock signal, this is the root of the effect of the counter counting down. When inverters were placed before the clock input, this effectively worked to flip the direction of counting around to counting up.

2. What would happen if the T Flip-flops were replaced with D flip-flops in this design?

The T flip flop specifically toggles it's out put to the opposite value each rising edge of the clock. This has the effect of halving the speed of the input clock signal. D flip flops on the other hand output whatever value the data signal contains when the rising edge of the clock input occurs.

This means that it isn't a guarantee that the output clock of the flip flop will always be the value we want, since it would require very specific timing of all the D inputs for each cascaded flip flop. Additional logic elements would need to be designed in order to accomplish a similar counter, adding to a more complex overall design.

3. Did you use the 10 Mhz or the 50 MHz clock in your design, and why?

I chose to use to the 50 MHz clock in my design because when doing the division beginning at the two respective clock speeds, the 50 MHz clock speed had the least amount of error with respect to 1 Hz when divided.

Appendix

1

```
// Christopher Hunt
// ENGR 271
// HW5
// sevenseg.sv

module sevenseg (
    input logic[3:0] D,
    output logic S_a, S_b, S_c, S_d, S_e, S_f, S_g
);
    assign S_a = ~D[3] & ~D[2] & ~D[1] & D[0] |
                 ~D[3] & D[2] & ~D[1] & ~D[0] |
                 D[3] & D[2] & ~D[1] & D[0] |
                 D[3] & ~D[2] & D[1] & D[0];

    assign S_b = D[2] & D[1] & ~D[0] |
                 D[3] & D[1] & D[0] |
                 D[3] & D[2] & ~D[0] |
                 ~D[3] & D[2] & ~D[1] & D[0];

    assign S_c = D[3] & D[2] & ~D[0] |
                 D[3] & D[2] & D[1] |
                 ~D[3] & ~D[2] & D[1] & ~D[0];

    assign S_d = D[2] & D[1] & D[0] |
                 ~D[3] & ~D[2] & ~D[1] & D[0] |
                 ~D[3] & D[2] & ~D[1] & ~D[0] |
                 D[3] & ~D[2] & D[1] & ~D[0];

    assign S_e = ~D[2] & ~D[1] & D[0] |
                 ~D[3] & D[2] & ~D[1] |
                 ~D[3] & D[2] & D[0] |
                 ~D[3] & ~D[2] & D[0];

    assign S_f = ~D[3] & ~D[2] & D[0] |
                 ~D[3] & ~D[2] & D[1] |
                 ~D[3] & D[1] & D[0] |
                 D[3] & D[2] & ~D[1] & D[0];

    assign S_g = ~D[3] & ~D[2] & ~D[1] |
                 D[3] & D[2] & ~D[1] & ~D[0] |
                 ~D[3] & D[2] & D[1] & D[0] ;
endmodule
```