Lab 5

Seven-Segment Display Driver

The goal of this lab is to build a complete seven-segment display driver, which will allow you to utilize the display in any future projects or labs. You will also test the driver using a simple up/down/load counter.

Part 1 (Rudimentary seven-segment display driver – revisited)

You built a rudimentary seven-segment display driver (`first_sseg_driver`) in lab 3. The driver generates all necessary control signals (i.e. `sseg`, `AN`, `DP`). It displays a 4-bit number (`num`) on a digit specified by (`active_digit`). You also tested this driver by connecting it to an external 3-bit signal `X` then changing that signal to display the sequence (0, 1, 2,…7).

Modify the application (`first_sseg_driver_test`) so that `X` is automatically generated by a 3-bit counter. The counter should utilize the 100 MHz clock available on the FPGA board; however, it should change its count at a slower frequency. The count frequency can be specified by a timer, Figure 1 shows a circuit that will allow you to implement this functionality.

- Search for a timer value (interval) that will allow you to see all generated numbers on all seven-segment digits simultaneously.
- Remember the value. You will use it for the complete seven-segment driver in Part 2.
- Use either of the provided timer modules (`timer_input` or `timer_parameter`) for the timer
- Use the up/down/load counter provided (`udl_counter`) for the 3-bit counter.

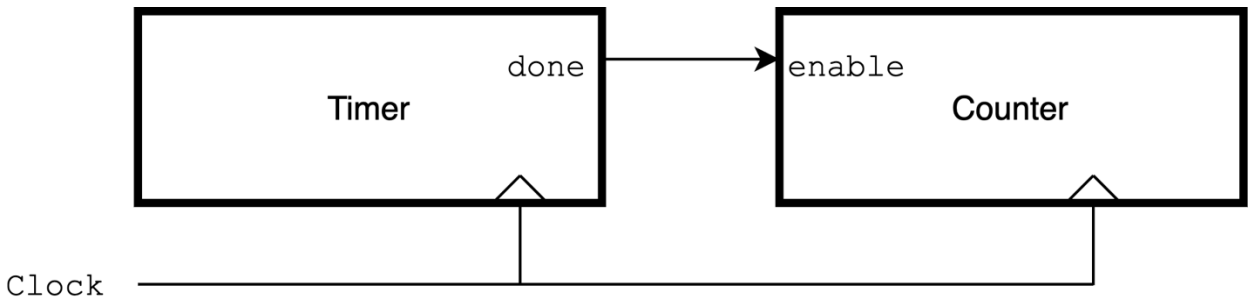

Figure 1: Timer-controlled Counter

Part 2 (Complete seven-segment display driver)

The technique used in part 1 is called time-multiplexed display. In this part you will use this technique to build a complete seven-segment driver with the following specifications:

- The driver accepts 8 inputs/numbers that can be displayed on each of the available seven-segment digits
- The size of each of the inputs `I` should be 6-bits.
    - `I[0]` should be used to specify the status of the decimal point
    - `I[4:1]` should be used for the actual HEX number
    - `I[5]` should be used to enable/disable the digit
- The driver utilizes a 100 MHz clock
- The driver should generate all necessary control signals (i.e. `sseg`, `AN`, `DP`)

Figure 2 shows a possible implementation of a complete seven-segment display driver. Use the provided starter code to write a Verilog module (`sseg_driver`) that describes the whole system.

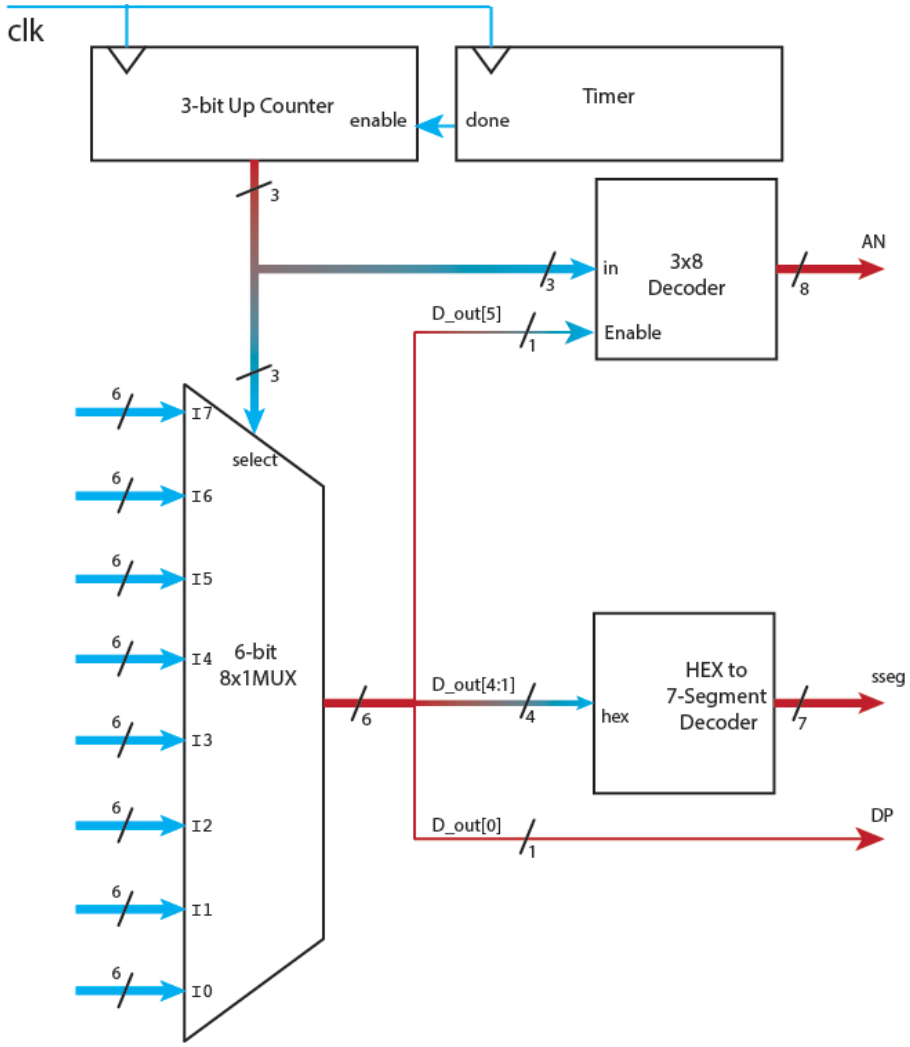

Figure 2: Time-multiplexed display driver

Part 3 (Display the content of an up/down/load counter)

Use the driver from part 2 to display the content of an up/down/load counter. The system should have the following specifications:

- Your top file should be called `counter_application.v`
- The counter range should be (at least) between 0 and 255.
- The counter functionality should be controlled as follows:
    - Up push button to count up
    - Down push button to count down
    - Middle push button to load a number specified by the switches
    - CPU reset push button to reset the counter to 0
- You might want to use `button.v` with the different push buttons
- You might find bin2bcd (from lab 2) useful

Submission check list:

[ ] All Verilog code you generated or modified
[ ] All testbenches written
[ ] Embed all screenshot of your testbench output in your README.md
[ ] Embed all block diagram generated in your README.md
[ ] Short videos demonstrating each of the parts you implemented on the FPGA