Lab 4: BCD Counters

Submission Due Dates:

Demo:	2022/10/18 17:20
Source Code:	2022/10/18 18:30
Report:	2022/10/23 23:59

Objective

1. Getting familiar with the 7-segment display and pushbuttons on the FPGA board.
2. Getting familiar with the counter designs and finite-state machines in Verilog.

Action Items

1 lab4_1.v (60%)

Design a 3-digit BCD up/down counter. In this lab, you must design a finite-state machine (FSM) (refer to the Logic Design course) to control the LEDs and 7-segment display based on the inputs from the pushbuttons. The FSM may consist of an **INITIAL** state to initialize all register's values, a **COUNTING** state to increase or decrease the counter value based on the current direction (you may store the direction information in a register), and a **STOP** state for the pausing.

a. I/O list:

- ✓ Input: **clk**, **rst**, **start**, **direction**
  - ✧ Note: Signals from pushbuttons should be processed by debouncing and one-pulse converters properly.
- ✓ Output: **DIGIT[3:0]**, **DISPLAY[6:0]**, **max**, **min**

b. **clk** (connected to **W5**)

The clock input with the frequency of 100MHz, positive-edge-triggered.

c. **rst** (connected to **SW0**)

The positive-edge-triggered (active-high) asynchronous reset signal, which resets the system to the **INITIAL** state.

d. **start** (connected to **BTNU**):

The control signal for changing the INITIAL state to the STOP state or toggling between the COUNTING and STOP states.

e. **direction** (connected to **BTND**)

The control signal for changing the direction between UP and DOWN in the COUNTING state. Also, it can display the current direction in the STOP state.

f. **DISPLAY[6:0]**

The signals control the seven LED segments of the 7-segment display.

g. **DIGIT[3:0]**

The signals control the four digits of the 7-segment display. In this lab, the leftmost digit displays the counting state; the rest rightmost digits indicate the current counter value.

h. **max** (connected to LED0)

Set to 1 if the counter reaches the upper bound (999) and 0 otherwise.

i. **min** (connected to LED1)

Set to 1 if the counter reaches the lower bound (000) and 0 otherwise.

j. **Finite state machine (FSM)**

The FSM may consist of the following states. You can design your own variant instead.

✓ INITIAL state

This state will only be entered upon reset with the **rst** button. First, it initializes the current counter to **050** and the current direction to **UP**. The FSM can enter the STOP state by pressing the **start** button. Example 1 demonstrates the 7-segment display in the INITIAL state.

Example 1 (INITIAL state after reset)

✓ STOP state

- The counter in this state will not be increased or decreased. It will display a dash and current counter value in the 7-segment display, as Example 2 shows.
- Pressing the **start** button can enter the COUNTING state. Pressing & holding the **direction** button in this state will **not** change the counting direction but display the current direction on the 7-segment display, as Example 3 shows. The display will resume to Example 2 when the **direction** button is released.

Example 2 (STOP state after reset)

Example 3 (STOP state when press & hold **direction** button)

✓ COUNTING state

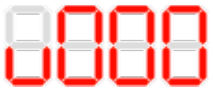
- The counter value will be increased or decreased by one every **0.1 seconds** based on its current direction, UP or DOWN, respectively. The counter will stop at 999 eventually when the direction is UP; the counter will stop at 000

at the end when the direction is DOWN.

- Pressing the **start** button can go back to the STOP state with the counter paused. Note that in the STOP state, the display will be similar to Example 2 with a different counter value.
- Pressing the **direction** button can change the counting direction between UP and DOWN. Examples 4 and 5 show the two possible statuses of the counter in the COUNTING state. Examples 6 and 7 show the cases when the counter reaches the boundary.

Example 4 (COUNTING state with DOWN direction)

Example 5 (COUNTING state with UP direction)



Example 6 (COUNTING state: counting down when reaching 000)

Example 7 (COUNTING state: counting up when reaching 999)

- After counting down to 000, pressing the **direction** button will change the counting direction to UP. The counter will start increasing again.
- After counting up to 999, pressing the **direction** button will change the counting direction to DOWN. The counter will start decreasing again.

k. DEMO video

https://www.youtube.com/watch?v=5zZxY2xwIRw

l. You have to use the following template for your design:

```
module lab4_1 (
    input wire clk,
    input wire rst,
    input wire start,
    input wire direction,
    output reg [3:0] DIGIT,
    output reg [6:0] DISPLAY,
    output reg max,
    output reg min
);
  /* Note that output ports can be either reg or wire.
   * It depends on how you design your module. */
```

```
    // add your design here
endmodule
```

2 lab4_2.v (40%)

Implement a configurable timer based on **lab4_1**. You can configure the counter value digit by digit in the STOP state.

a. I/O list:

- ✓ Input: **clk**, **rst**, **start**, **direction**, **increase**, **decrease**, **select**
- ✓ Output: **digit[3:0]**, **display[6:0]**, **max**, **min**, **d2**, **d1**, **d0**

b. **increase** (connected to BTNR)

The signal increases the digit value in the STOP state.

c. **decrease** (connected to BTNL)

The signal decreases the digit value in the STOP state.

d. **select** (connected to BTNC)

The signal selects the digit to be changed by the **increase** or **decrease** buttons in the STOP state.

This button has **no effect** when in the INITIAL or COUNTING state.

e. **d2** (connected to LED7), **d1** (connected to LED6), **d0** (connected to LED5)

The three LED indicates which digit is currently selected. The counter value has three digits. If Digit 2 is selected, LED7 will be on. LED6 and LED5 will be off. These three LEDs are off when the FSM is in other states.

f. Configuration:

- The configuration can only be done in the STOP state. That is, the three new buttons, **increase**, **decrease**, and **select**, have no effect in the INITIAL and COUNTING states.
- When entering the STOP state, the default digit to be configured will be Digit 0. LED5 is on accordingly. You can press **select** to switch to Digit 1. And LED6 will turn on. Press **select** again to switch to Digit 2 (with LED7 on). Press **select** once again to go back to Digit 0.
- With one of the digits selected, press **increase** (**decrease**) can add (minus) its value by one. For example, if Digit 1 is selected and the current counter is 5**3**2, pressing **increase** will change the counter value to 5**4**2. By contrast, pressing **decrease** will change the counter from 5**3**2 to 5**2**2.
- When the selected digit's value is 9, pressing **increase** will reset its value to 0. For example, if Digit 1 is selected and the current counter value is 5**9**2, pressing

increase will change the value to 502. Similarly, when the selected digit's value is 0, pressing **decrease** will set its value to 9. For example, if Digit 2 is selected and the current counter value is 092, pressing **decrease** will change the value to 992.

g. DEMO video

https://www.youtube.com/watch?v=BHgnqshJstU

h. You have to use the following template for your design:

```
module lab4_2 (
    input wire clk,
    input wire rst,
    input wire start,
    input wire direction,
    input wire increase,
    input wire decrease,
    input wire select,
    output reg [3:0] DIGIT,
    output reg [6:0] DISPLAY,
    output reg max,
    output reg min,
    output reg d2,
    output reg d1,
    output reg d0
);
  /* Note that output ports can be either reg or wire.
   * It depends on how you design your module. */
  // add your design here
endmodule
```

3 Questions and Discussion

Please answer the following questions in your report.

A. Please explain why we need the **debounce** module. Does its clock rate need to be the same as the **one-pulse** module's clock rate? And why? You may draw waveforms to help with the explanation.

B. What happens when the one-pulse module's clock rate is faster or slower than the FSM's? You may draw waveforms to help with the explanation.

C. Please explain how the 7-segment can simultaneously display different content for each digit to the human eyes. Hint: frames per second (FPS) for human eyes.

Note: You can use Wavedrom, an open-source waveform editor, to draw the

waveforms. You can also use PowerPoint or any other drawing tools.

4 Guidelines for the report

Refer to the guidelines in the report template (or in the previous lab assignments).
Grading policy (subject to change): Part (A): **35%**; Part (B): **50%**; Part (C): **10%**; (D): **5%**

Attention

- ✓ DO NOT copy-and-paste code segments from the PDF materials. It may also paste invisible non-ASCII characters, leading to hard-to-debug syntax errors.
- ✓ In this lab, we provide you **onepulse.v**, **debounce.v** and **clock_divider.v**. Do not include them in your submissions. We will take care of them when verifying your source code.
- ✓ If you have two or more modules used for any specific lab, merge them into one Verilog file before the submission.
- ✓ You should submit two source files, including **lab4_1.v and lab4_2.v**. Upload each source file individually. DO NOT hand in any compressed ZIP files, which will be considered an incorrect format.
- ✓ You should also hand in your report as **lab4_report_StudentID.pdf** (i.e., lab4_report_108456789.pdf).
- ✓ You should be able to answer questions of this lab from TA during the demo.
- ✓ You need to prepare the bitstream files before the lab demo to make the demo process smooth.
- ✓ Feel free to ask any questions about the specification on the EECLASS forum.