Lab 05: The Awesome Vending Machine

Submission deadlines:

Source Code:	2019/10/29 18:30
Report:	2019/11/03 23:59

Objective

- ➢ To be familiar with modeling finite state machines with Verilog.
- ➢ To be familiar with the FPGA design flow, the demo-board I/Os

Description

Design a controller for the Awesome Vending Machine which is very similar to an actual vending machine. It behaves as follows:

- The basic concept of how this vending machine works:
  - Initially, the machine will show “0000” on the 7-segment display. All LEDs will be turned off.
  - When the customer presses the money button, the corresponding coin will be deposited into the machine. The two **rightmost** 7-segment digits will show the balance (the amount of deposited money). If the balance is more than the price of the drinks, the corresponding LEDs will light up.
  - When the customer presses the cancel button, the machine will return the deposited money.
  - When the customer presses the drink button **once**, the two **leftmost** 7-segment digits will show the price of the corresponding drink. When the customer presses the same drink button **again**, the drink will come out if the deposited money is enough.
  - When dropping the drink, the 7-segment display will show its name. And all LEDs should be turned off.
  - When the drink has been dropped, the change should be returned. The 7-segment display will show the decreasing balance and return the change coin by coin with corresponding LEDs lighting up.

- The detailed spec:
  - INITIAL state: After being reset, the machine goes to the

INITIAL state. The 7-segment display will show “0000”. There are LEDs corresponding to *enough_A*, *enough_B*, and *drop_money(10 LEDs)*. All the LEDs will be off initially. Then, the machine goes to the DEPOSIT state**.**

- DEPOSIT state:
  - First, the customer may deposit $5 coin by pressing the *money_5* button or $10 by pressing the *money_10* button. Every time after the customer deposits a coin ($10 or $5), the two **rightmost** 7-segment digits will show the total amount of money that has been deposited so far (*balance*). The price of the two drinks: *drink_A* and *drink_B* are *price_A* = $20 and *price_B* = $25, respectively. The machine will return the deposited money if the customer presses the *cancel* button. The *balance* cannot be over 99. The LED *enough_A* or *enough_B* will light up if the *balance* is more than the price of *drink_A* or *drink_B*, respectively.
  - If the customer presses *drink_A* or *drink_B* **once**, the two **leftmost** 7-segment digits (*price*) will show the *price_A* or *price_B*. If the customer presses *drink_A* first and then presses *drink_B*, the price will be *price_B*, vice versa.
  - If the customer presses the same drink button twice, the machine will enter the BUY state if the *balance* is enough.
- BUY state:
  - The LED *enough_A* and *enough_B* would be turned off.
  - The 7-segment display shows the name of the selected drink for one clock cycle (**with the frequency of clk/(2^26)**). The names of *drink_A* and *drink_B* can be defined freely. But limited to 4 characters. You can refer to the following graph.

- CHANGE state: The two rightmost 7-segment digits will show the change to be returned to the customer (*balance* – *price*). If there are more than or equal to $10 to return, the change will be decreased by $10 (to simulate dropping a coin of $10), and the ten leftmost LEDs corresponding to the *drop_money* (LD15~LD6) will light up at each clock cycle (**with the frequency of clk/(2^26)**) until the two rightmost 7-segment digits show "00" or "05". If there is less than $10 to be returned, the change will be decreased by $5 once and the five leftmost LEDs corresponding to the *drop_money* (LD15~LD11) will light up for one clock cycle (**with the frequency of clk/(2^26)**).
- *Bonus: In the DEPOSIT mode, if the customer doesn't take any action for about 5~10 seconds (see Hint c), the leftmost 7-segment will show "00", and the machine will return the change the customer has deposited.

I/O signal specification:

- ***clk***: clock signal with the frequency of 100MHz (connected to pin **W5**).
- ***rst***: active-high reset (connected to **SW0**).

✧ Each signal from the pushbutton in the following have to be processed by a debouncer, and then a one_pulse circuit.

- ***money_5*:** deposit a $5 coin (connected to **BTND**).
- ***money_10*:** deposit a $10 coin (connected to **BTNU**).
- ***cancel***: cancel current drink choice and return the *balance* if

there is any. (connected to **BTNC**).

- ***drink_A***: show *price_A* or confirm the purchase of *drink_A* (connected to **BTNL**).
- ***drink_B***: show *price_B* or confirm the purchase of *drink_B* (connected to **BTNR**).

- ***enough_A***: indicating if the *balance* is more than the *price_A* (connected to **LD1**).
- ***enough_B***: indicating the *balance* is more than the *price_B* (connected to **LD0**).
- ***drop_money[9:0]***: the drop_money signal (connected to **LD15 ~ LD6**).
- **DIGIT[3:0]**: signals to enable one of the 7-segment digits.
- **DISPLAY[6:0]**: signals to show the digits on the 7-segment display.

Note:

1. **All the signals of the pushbutton should be properly processed with the debounce and one pulse.**
2. **The clock frequency of each debouncer or one-pulse circuit is clk/(2^16)**
3. **The clock frequency of the seven-segment display controller is clk/(2^13)**
4. **Demo video:**
   https://www.youtube.com/watch?v=UO2x2e1i4js&feature=youtu.be

Hint:

1. You must design at least one finite state machine (FSM).
   a) There should be at least four states. Think carefully for the operation frequencies of the pushbuttons and 7-segment display.
   b) The proper duration for each state may be:
      1. INITIAL: clk/2^16
      2. DEPOSIT: clk/2^16
      3. BUY: clk/2^26
      4. CHANGE: clk/2^26

      You may operate at a higher clock rate and use a counter to slow down the display rate (e.g., the FSM can operate at the

clock rate of clk/2^16. But every time when it enters the BUY or CHANGE state, it will stay for at least 2^10 cycles before changing to other states). Or you may use a clock selector as a quick fix.

c) clk/2^29 is about 5 seconds. You may design a counter to measure this period for the bonus.

2. You can use the following template for your design.

```
module lab05(clk, rst, money_5, money_10, cancel, drink_A,
drink_B, drop_money, enough_A, enough_B, DIGIT, DISPLAY);
    input clk;
    input rst;
    input money_5;
    input money_10;
    input cancel;
    input drink_A;
    input drink_B;
    output reg[9:0] drop_money;
    output reg enough_A;
    output reg enough_B;
    output [3:0] DIGIT;
    output [6:0] DISPLAY;
    //add you design here
endmodule
```

Attention:

1. You should hand in only one Verilog file, lab05.v. If you have several modules in your design, integrate them in lab05.v. (Please do not hand in any compressed files, which will be considered as an incorrect format.)
2. You should also hand in your report as lab05_report_StudentID.pdf (i.e., lab05_report_107066666.pdf).
3. You should be able to answer questions of this lab from TA during the demo.
4. You need to generate bitstream before demo.