LAB 7 – 3300L

Spring 2025 / Dr. Van Blerkom

For this lab, we will design and implement a digital vending machine using a finite state machine in Verilog. The machine will use 7-segment LEDs and the UART to mimic the operation of a real vending machine. The UART receiver will be used to input letters and numbers to simulate the selection of the item, and the input of a quarter or a dime. The UART transmitter will be used to output the amount shown on the 7-segment display, whenever that value changes.

1. **States:**
    - WaitSelection: State where the user inputs an item number to select.
      The price for the item should show on the 7-segment display (and output to the UART).
    - WaitPayment: State where the user inserts payment.
      The remaining amount needed should show on the 7-segment display (and output to the UART).
    - GotQuarter: State when a quarter has been received.
      The remaining amount should be reduced by 25 cents, unless the remaining amount is less than or equal to 25 cents – in that case, the change to be given should be calculated and stored in amount, and the FSM should go to Dispensing.
    - GotDime: State when a dime has been received.
      The remaining amount should be reduced by 10 cents, unless the remaining amount is less than or equal to 10 cents – in that case, the change to be given should be calculated and stored in amount, and the FSM should go to Dispensing.
    - Dispensing: State where the machine dispenses the selected item.
      The UART will send the message “PAID”.
    - ChangeReturn: State where the machine returns change (if any).
      The change returned should be displayed on the 7-segment display (and output to the UART).

2. **Inputs:**
    - For the UART input, the letters that control the actions are:
        1. “0” through “7” == choose the item and show the cost
        2. “S” == Finalize selection and start payment
        3. “Q” == Quarter detected
        4. “D” == Dime detected

3. **Output:**
   For the UART output, the amount on the 7-segment display will be sent in decimal, in ASCII format. This means, for example, that if the amount 125 is shown on the 7-segment display, the three ASCII characters 0x31 (“1”), 0x32 (“2”), 0x35 (“5”) must be

transmitted to the terminal. This should be followed by a carriage-return (0x0D) and a line-feed (0x0A) so that we move to the next line on the terminal.

4. **Functionality:**
    a. Users choose an item number by sending the number over the UART.
    b. The 7-segment display will show the price for that item, and the price will also be sent over the UART.
    c. When "S" is sent, the selection is confirmed, the price is stored in the "amount needed to pay" register, and the system waits for payment.
    d. As quarters and dimes are added, the "amount needed to pay" will decrease, and will be updated on the 7-segment display (and sent over the UART).
    e. When the "amount needed to pay" goes to zero or below, the product will be dispensed.
    f. When dispensing, the UART will send the message "PAID".
    g. The display and UART will then show the amount of change to be given.
    h. The system returns to the WaitSelection state after dispensing the item or returning change.

5. **Pricing and Items:**
    a. There will be 8 different items for sale which can be selected.
    b. The cost for the 8 items is set by the following module:

```
module item_cost(
                input [2:0] item_sel,
                output reg [7:0] item_cost
                );

   always @(*)
     case (item_sel)
       0: item_cost = 125;
       1: item_cost = 100;
       2: item_cost = 85;
       3: item_cost = 150;
       4: item_cost = 225;
       5: item_cost = 185;
       6: item_cost = 50;
       7: item_cost = 135;
       default: item_cost = 100;
     endcase
endmodule
```