UCSD CSE140L Fall 2013

LAB#3

(Due: See course web page)

Instructor: Dr. Choon Kim

Objective

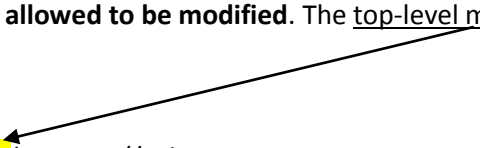
- Based on the experience from LAB#1&2, learn how to design, simulate, synthesize, program on FPGA and test **FSM(Finite State Machine)** digital system using Altera Quartus II CAD SW and DE1 FPGA board.

- Learn and become familiar with logic design using **Verilog Hardware Description Language**

Instructions

1. Your LAB#3 project name should be L3Cyyy, where yyy=your CID(e.g., L3C079 if your CID=079*)*. The golden solution .pof and .sof files are provided. **Student should play with golden solution as a reference whenever he/she has a question during design.**

2. Use Verilog HDL design. Use the following Verilog top-level module interface code for your design. **No part of this code is allowed to be modified**. The top-level module name must be same as your LAB project name.



```
module L3Cyyy(          // where yyy=your CID. For example, L3C079 if your CID=079
    input  [9:0] sw,    // ten up-down switches, SW9 - SW0
    input  [3:0] key,   // four pushbutton switches, KEY3 - KEY0
    input   clock,      // 24MHz clock source on Altera DE1 board
    output [9:0] ledr,  // ten Red LEDs, LEDR9 - LEDR0
    output [7:0] ledg,  // eight Green LEDs, LEDG8 - LEDG0
    output reg [6:0] hex0, hex1, hex2, hex3      // four 7-segment, HEX3 - HEX0
);
```

3. Like LAB#2, our acceptable timing margin for real-time clock operation is -30 and +30%.

Soda VM(Vending Machine) Controller Design

****

A vending machine company requests you to design a soda Vending Machine(VM) controller circuit following specifications below. The price of a soda was set to **35 cents**.

LAB#3 Project Operation Flow

Warning: Following operations are ***__**prerequisite**__*** conditions. You will get **zero(0) point** for LAB#3 if you fail these operations.

1) **Initial setting before turning on power**: Whenever you turn on power, you must always make the following initial setting before turning on power.
   - all sw are in DOWN position
   - no key is PRESSED

2) When the power is turned on, your design is in initial state with following conditions:
   - all leds(i.e., ledg and ledr) are OFF
   - VM is disabled
   - HEX[3:0] displays your CID. For example, HEX[3:0]=**0097** if your CID=097
     (Reminder: Golden solution has HEX[3:0]=0353 since it's CID=353)

3) To start VM from the initial state:
   Press key[3] once. The hex[3:0] should display "**0000**" as an initial display.

4) To stop VM and return to initial state:
   Turn off power first. Then turn on power(with initial setting).

VM Operation Specifications

(Reminder: You should check with golden solution for more details or whenever necessary)

1) How to deposit money to VM:

Money deposit to VM is made by setting up amount of money first followed by pressing EnterKey.

Setting up amount of money

```
sw[8]    //  1 for Reset input (=clearing Deposit balance to 0)
sw[4]    //  1 for Credit-card input(= makes 35 cents immediately regardless of current balance)
sw[3]    //  1 for One-dollar bill input
sw[2]    //  1 for Quarter input
sw[1]    //  1 for Dime input
sw[0]    //  1 for Nickel input
```

Enter key

```
key[3]    // one pressing deposits above money input amount one time.
```

2) HEX[3:0] Display Specifications:

IF sw[9] = 0 {

  IF special cases (See 4. Special cases) {

    hex[3:1] = "**Err** "

    hex[0] = off

  }

  ELSE { // normal operation...

    hex[3:2] = Deposit balance. It displays value up to "35"(when dispensing occurs).

    hex[1:0] = Change balance e.g., 15

      e.g., hex[3:0]=3515 when 50 cents were deposited

  }

}

ELSE IF sw[9] = 1 (with all other sw are down) {

- hex[0] displays the total number of dispensing made only by **coin input** since board power was turned on.
- The number must be in hexadecimal using modulo-16 format.
- DO NOT make One-dollar-bill input or Credit-card input before testing **sw[9]=1** function. This is a test for only coin input condition!
- hex[3:1] = all OFF
- This operation should NOT CHANGE the value of other variables in your design, such as Deposit balance or Change balance.

}

3) ledr[9:0] --- dispensing indicator

- All blinking(**half-second period** with **50% duty cycle**) when dispensing,
- All OFF when NOT dispensing

4) Special cases: Your design should be able to handle the following special cases

4.1) ****Multiple inputs case: ****
When more than one sw are UP among sw[0,1,2,3,4,8] AND EnterKey is pressed, hex[3:0] should display "Err ". For example, sw[2]=sw[3]= 1.

4.2) ****Consecutive one-dollar bill inputs or Consecutive credit-card inputs case: ****
Two consecutive one-dollar bill inputs or Two consecutive credit-card inputs should display "Err " on hex[3:0]. For example, one-dollar bill input followed by one-dollar bill input, or credit-card input followed by credit-card input.
(However, note that one-dollar bill input followed by credit-card input, or credit-card input followed by one-dollar bill input is O.K.)

4.3) ****Credit Card input when hex[3:0]=3500(i.e., when Deposit=35 and Change=00) ****
This is another case when hex[3:0] should display "Err ".

The "Err " display is cleared to "**0000**"(=Deposit and Change balance are and cleared to zero) by either pressing EnterKey or **Reset** input. Then normal operation can continue.

Checking Items during Demo

Warning: **ledr[9:0] blinking operation**(See Sec.3) is a dispensing indicator. Therefore it must be working correctly during demo. You will get **zero(0)** point for the **Part** if the ledr[9:0] blinking operation does not work correctly.

PART 1(3 pts) : **Coin input** operation

PART 2(3 pts) : **sw[9] = 1** (with all other sw are down) operation

PART 3(3 pts) : **Credit-card** and **Reset** operations

PART 4(3 pts) : **One-dollar bill** operation

PART 5(3 pts) : **"Err " display** operation