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. The solution .pof and .sof files are provided to you. Your design must behave exactly same as this solution except solution identifier(s). You should use the given solution as a golden reference during your design, testing & debugging your project. You must run this solution on your board and compare with your design whenever you have any question during your work. If there is any discrepancy between the solution and the written instruction, the solution is correct.

2. Use Verilog HDL. Like previous project, your project name should be LxGzz where x=LAB number, zz=your group number. For example, a project name, L3G09, is a name of LAB#3 Project done by Group#9.

3. Use following top-level module interface code in your design. No part of this code is allowed to be modified.
   
   **Hint:** The top-level module name must be same as the project name (e.g., L3G09) in our Quartus.

   ```
   module LxGzz(
     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] hex3, hex2, hex1, hex0 // four 7-segment, HEX3 - HEX0  
   );
   ```

4. Rules for this LAB
4.1 In the LAB Report, replace *Functional & timing simulation waveforms* with a *Compilation Report - Flow Summary* of your design. Clearly mark the Percentage(%) value of "Total logic elements".

4.2 No tpd timing setup is required.

5. Our acceptable timing margin (or tolerance) for real-time operation is -30 and +30%.
   For example, in case of 1 second required in Part4&5 of this LAB, a time period between 0.7 sec (= -30%) and 1.3 sec (= +30%) is acceptable as 1 sec period. A time period beyond this range is an error and will get zero point.

6. Use of simulation for debugging
   Simulation is one of the *most effective debugging methods* in digital design engineering. Like many professional engineers, you have to use simulation to figure out the cause of the problem *when your design behaves differently than you expect*. Without simulation, you may have difficulties in solving your problem and waste your time.

7. Any modification or change on any LAB project, if occurs, will be posted on our course web page.

---

**Soda VM(Vending Machine) Controller Design**

*****************************************************************************
Design Specifications

You design a soda Vending Machine (VM) controller circuit following specifications below. The price of a soda is **35 cents**.

**VM Operation Flow**
(Nota: Use the solution for details. Also watch the instructor's demonstration during the class)

1. **[When Power is turned on (= Initial State)]**
   All SWs are down. NO key is pressed. All leds (both green and red) are OFF. hex[3:0] displays "L3xx" where xx=your group number. For example, "L399".

2. **[Starting VM]**
   Press key[3] once. Then hex[3:0] should display "0000".

3. **[VM operation]**
   - An input (Coin, One-dollar bill, Credit card, and Return request) is made by a sw setup followed by pressing EnterKey (=key[3]).
   - Output(s) is displayed on hex[3:0] and led green and red

**Modes:**
- sw[9] // 1 for NO change mode (i.e., "Use Exact Change Only") -- hex[1:0] is OFF,
  // 0 for normal mode

**Input:**
- sw[8] // 1 for Return indicator,
- sw[4] // 1 for credit-card input indicator,
- sw[3] // 1 for one-dollar bill indicator,
- sw[2] // 1 for Quarter input indicator,
- sw[1] // 1 for Dime input indicator,
- sw[0] // 1 for Nickel input indicator,
- key[3] // EnterKey, **press this key[3] AFTER you set only one sw[8, 4:0].**

**Output:**
- hex[3:0] // displaying **deposit balance** on hex[3:2], **change balance** on hex[1:0]
- ledg[7:0] // all blinking = dispensing

- When the deposit balance becomes 35 cents or above, a soda and change (if any) is dispensed.
- The dispensing action is indicated by all LEDGs(ledg[7:0] blinking with half-second period with 50% duty cycle.
- **[Special Case]** Two consecutive one-dollar bill inputs should display "Err " on hex[3:0]

Example (with EnterKey),
- sw[0]=1 makes 5₵ accepted by VM,
- sw[2]=1 makes 25₵ accepted by VM,
- sw[3]=1 makes $1 accepted by VM,
- sw[4]=1 makes necessary amount of balance is charged to Credit Card by VM,
- sw[8]=1 clears deposit & change balance
- sw[9]=1 makes hex[1:0] OFF since NO change is returned (EnterKey is NOT needed for this)
Checking Items & Grading Points

**PART 1:** Initial state with GN display operation & Starting VM operation

**PART 2:** coin operation

**PART 3:** credit card operation

**PART 4:** Return operation & NO change mode (i.e., "Use Exact Change Only") operation

**PART 5:** One-dollar bill operation with [Special Case]

------------------------ The End of LAB3 ------------------------