EEL 3701C – Digital Logic & Computer System

LAB 4 – Alarm Clock

Problem 1: Alarm Clock Controller

Goal: In this lab, you will design, develop, and implement a controller for an alarm clock (Figure 1) using VHDL, your FPGA, and extension board.

Description of the Alarm Clock

The alarm clock has a certain number of input switches (S_i on Figure 1 with $i \in \{1..7\}$) and outputs (7-segment display (SSG), LEDs and the buzzer).

INPUTS	**Switch S_1:** Moves the clock into its "Set_time" mode **Switch S_2:** Moves the clock into its "Set_alarm" mode **Switch S_3:** Allows Minutes to be incremented when in a set mode **Switch S_4:** Allows Hours to be incremented when in a set mode **Switch S_5:** Activates the alarm **Switch S_6:** Turns the entire clock on/off **Switch S_7:** Mutes the alarm clock **Push Button PB_1:** Stops the alarm after it has started ringing
OUTPUTS	**The four SSGs:** Displays time or alarm time depending on S_1 and S_2 **The Buzzer:** Buzzes when the alarm is triggered and **not** muted. **LED 0:** Activity depends on clock's current state. **LED 1:** Is on whenever the alarm clock is activated, regardless of the current state.

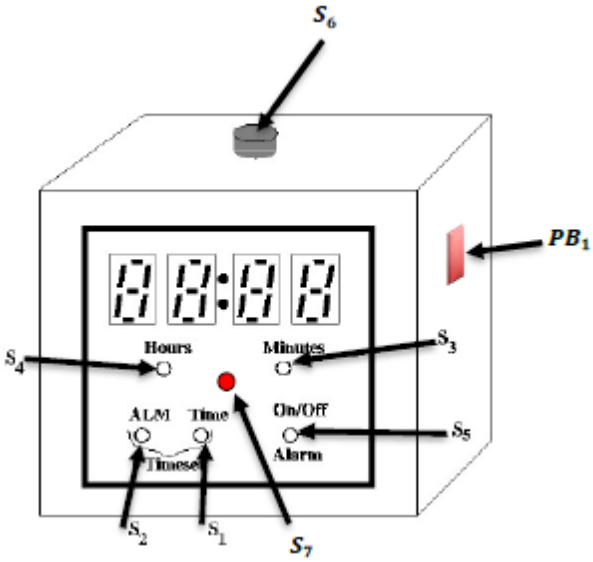

Figure 1 - Potential layout for the alarm clock

The behavior of the alarm clock is specified as a finite state machine with 4 states:

- **Time:** The default state.
  - In this state, seconds, minutes and hours are incremented as normal for a clock.
  - **LED0** blinks with a period of 2s and a 50% duty cycle when the alarm is not ringing.
  - Move to state **Set_time** if **S1** is activated.
  - Move to state **Set_alarm** if **S2** is activated.
  - Move to state **Alarm** if the current time matches the alarm time and the alarm is activated (**S5** is on).
- **Set_time:** Used to set the current time of the clock.
  - **LED 0** is off
  - Activating **S3** then causes minutes to be incremented once per second.
  - Activating **S4** then causes the hours to be incremented once per second.
  - Both can be activated at the same time.
  - By deactivating **S1**, the controller returns to state **Time**.
- **Set_alarm:** Used to set the alarm time.
  - **LED 0** is on
  - Activating **S3** then causes the minutes to be incremented once per second.
  - Activating **S4** then causes the hours to be incremented once per second.
  - Both can be activated at the same time.
  - By releasing **S2**, the controller returns to state **Time**.
- **Alarm:**
  - **LED 0** blinks with a period of 4s and a duty cycle of 75%.
  - **Buzzer** sounds with a period of 2s and a duty cycle of 50% if the alarm is not muted (if **S7** is off).
  - Pressing **PB1** *instantly* moves the clock to state **Time**.

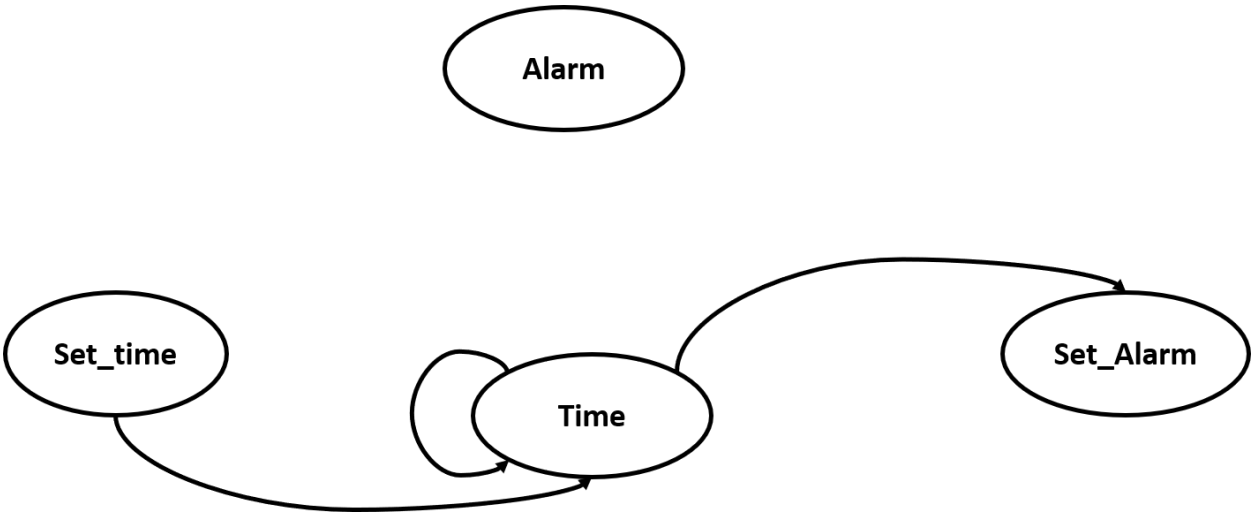

Figure 2 - FSM diagram of the alarm clock

Pre-Lab Homework

1. Complete the state diagram of Figure 2 by adding all transitions, labels and outputs. Assume that the display of numbers on the SSG and the clock are implicit and don't have to be driven in states. **(4 pts)**

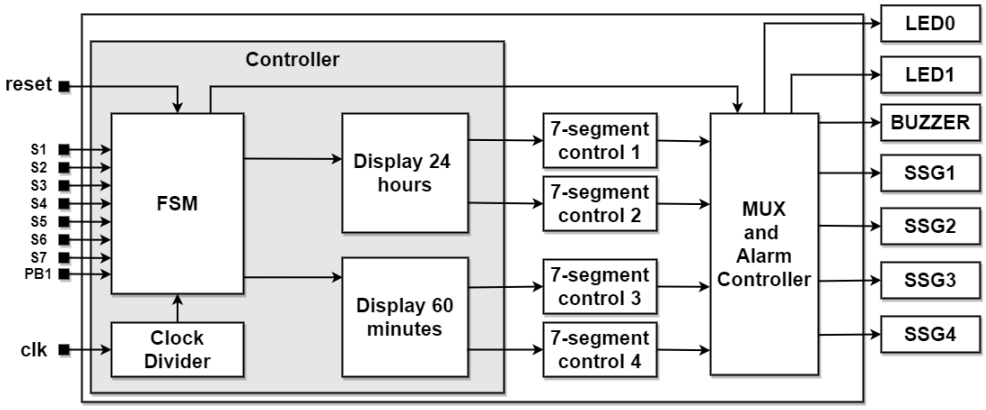

Figure 3 - Architecture of the alarm clock

2. To implement the alarm clock as shown in Figure 3in VHDL, we need a couple of modules.
   a. A two-digit display module that displays a 2-digit decimal number on 2 SSGs: This module will be instantiated twice. One instance will display hours and the other will display the minutes. *Remember: You can instantiate your old 7-segment decoder component inside these 2-digit displays!*
      i. Provide the VDHL design of this module **(4 pts)**.
      ii. Build a **test bench** and simulate your design for correctness **(2 pts)**.
   b. A MUX and alarm controller: As the name indicates, it takes 4 numbers in binary at the input and drive the 4 SSGs. The basis for this was set in the previous lab with problem 3. Additionally, this module passes the FSM's buzzer and LED outputs through to the FPGA pins.
      i. Modify your VHDL for Lab3/Problem3 to implement the MUX and Alarm Controller **(4 pts)**.
      ii. Build a **test bench** and simulate your design for correctness (**2 pts)**.

Make sure in your implementation that minutes are increment for every 60 seconds as shown in Figure 4

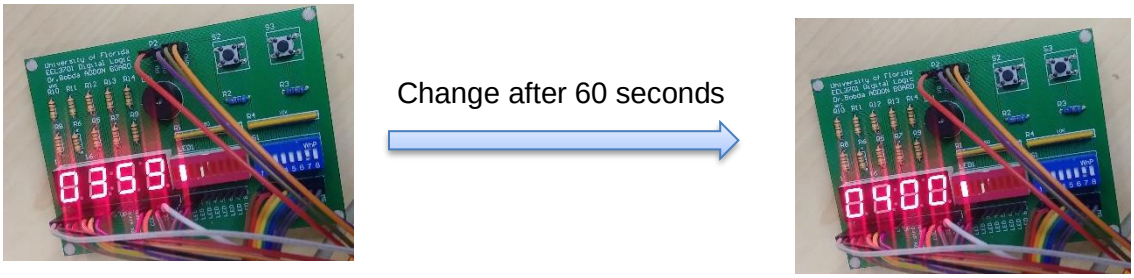

Figure 4 - Proper ticking of the clock

3. Implement your top-level design in VHDL, using a clock divider that output a 1s clock (modify your VHDL code of Lab2/Part2. Properly connect all the I/O from the add-on board to the GPIO pins of the MAX10 board. **(6 pts)**

In-Lab Implementation

1. Compile your design, make sure it is fault free, download it on the FPGA and verifies that the alarm clock works properly **(10 pts)**
2. Demonstrates and explains your whole design to your PI. You must demonstrate that you have mastered the implementation of all components. You PI will ask you specific questions on the operation of your alarm clock **(5 pts)**

Deliverables: Prepare and submit your report on canvas using the template provided on the webpage. Include supporting documents (pictures, video, or compressed sources after cleaning up your project) in your report. If you submit your sources, make sure to describe your pin assignment.

N.B All Pre-Lab Homework must be completed before your lab session. You will not be admitted to the lab without completing this part.