1. CSE 140L Lab 2

(JiKun Wang, A17164596); (Chuning Liu 2,A17168379); (Wensheng Yu, A16163418);

Academic Integrity

Your work will not be graded unless the signatures of all members of the group are present beneath the honor code.

To uphold academic integrity, students shall:

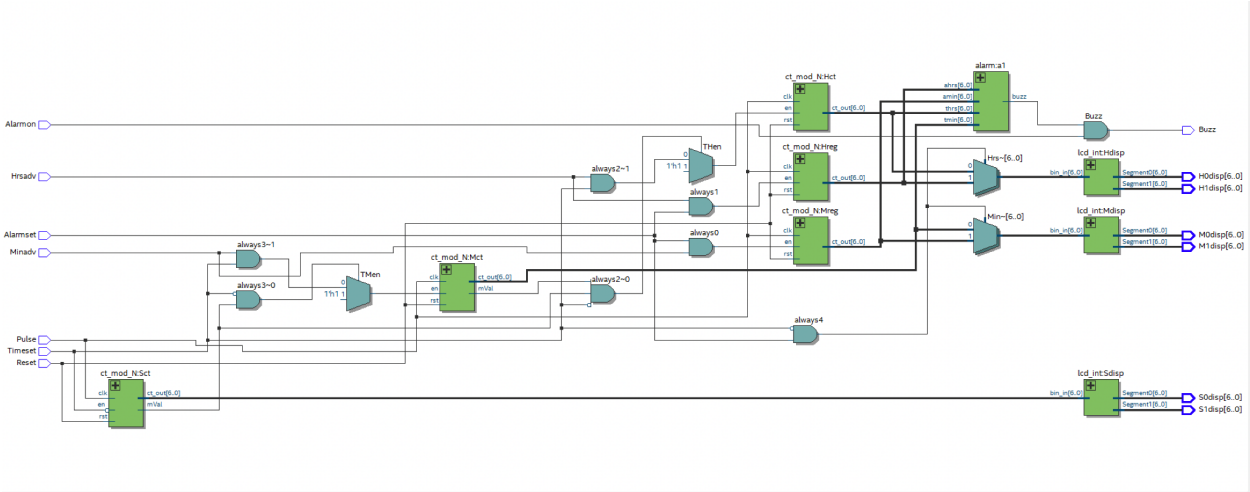
- Complete and submit academic work that is their own and that is an honest and fair representation of their knowledge and abilities at the time of submission.
- Know and follow the standards of CSE 140L and UCSD.

Please sign (type) your name(s) below the following statement:

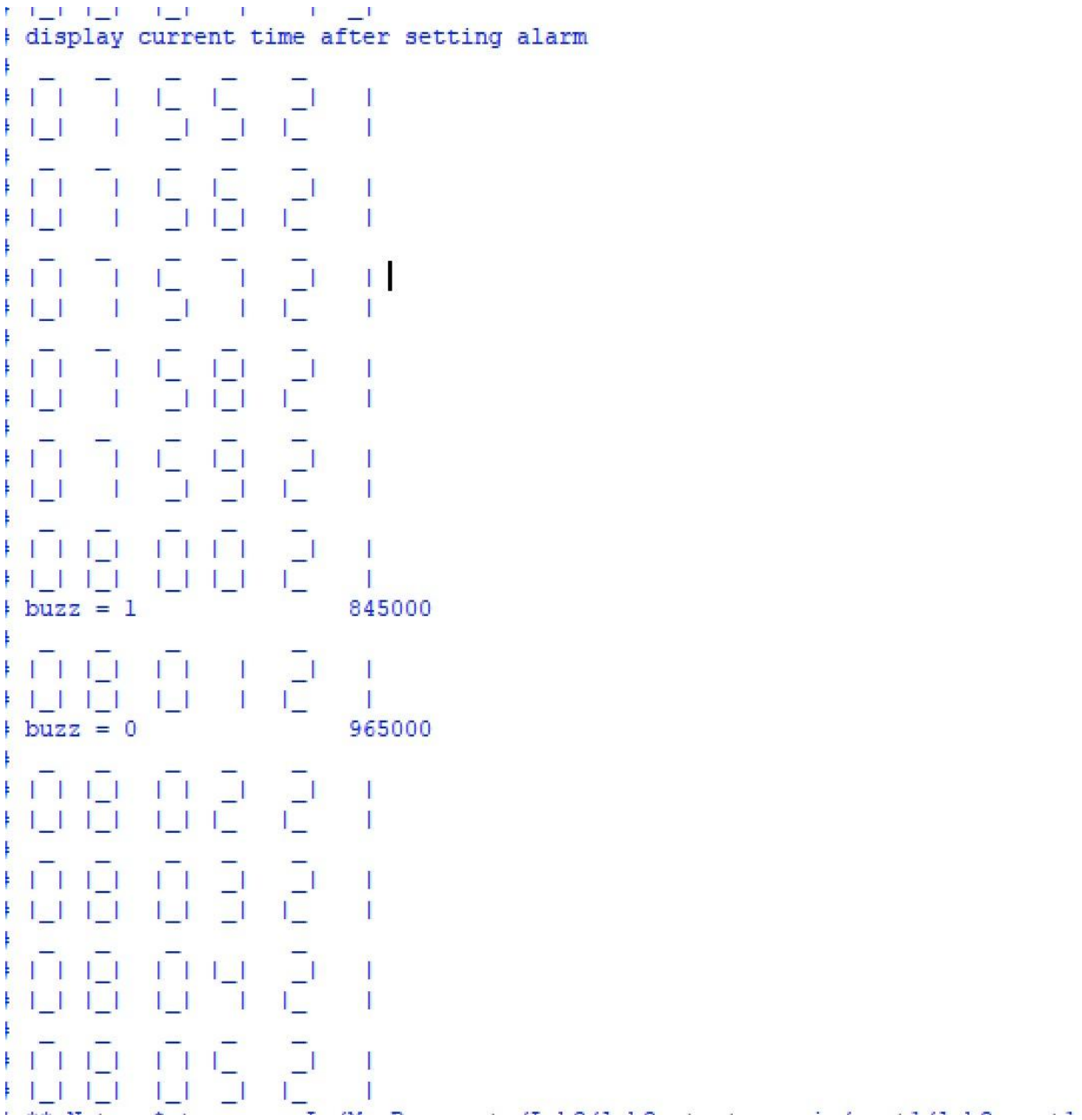
I pledge to be fair to my classmates and instructors by completing all of my academic work with integrity. This means that I will respect the standards set by the instructor and institution, be responsible for the consequences of my choices, honestly represent my knowledge and abilities, and be a community member that others can trust to do the right thing even when no one is watching. I will always put learning before grades, and integrity before performance. I pledge to excel with integrity.

(JiKun Wang)
(Chuning Liu)
(Wensheng Yu)

1. A screenshot of the RTL viewer top level schematic/block diagram in Quartus for Part 1. (2 pts)



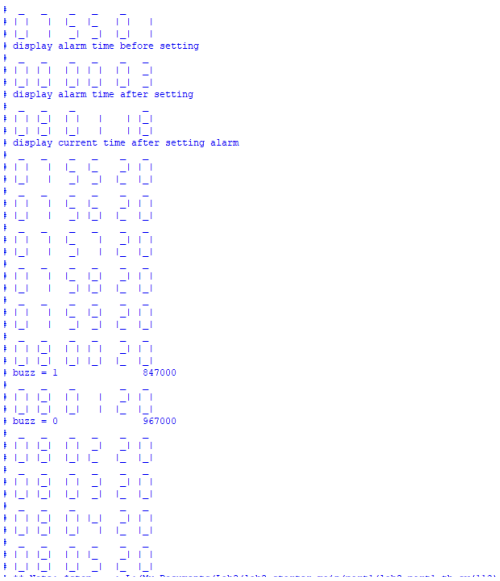
2. A screenshot of the Modelsim transcript, showing correct alarm functionality for Part 1. (2 pts)



3. A summary paragraph explaining how you tested your alarm clock. Modify the testbench and be sure to include details on how you tested cases other than the given testbench. In particular, what other cases did you feel you needed to test and how did you do it? (3 pts)

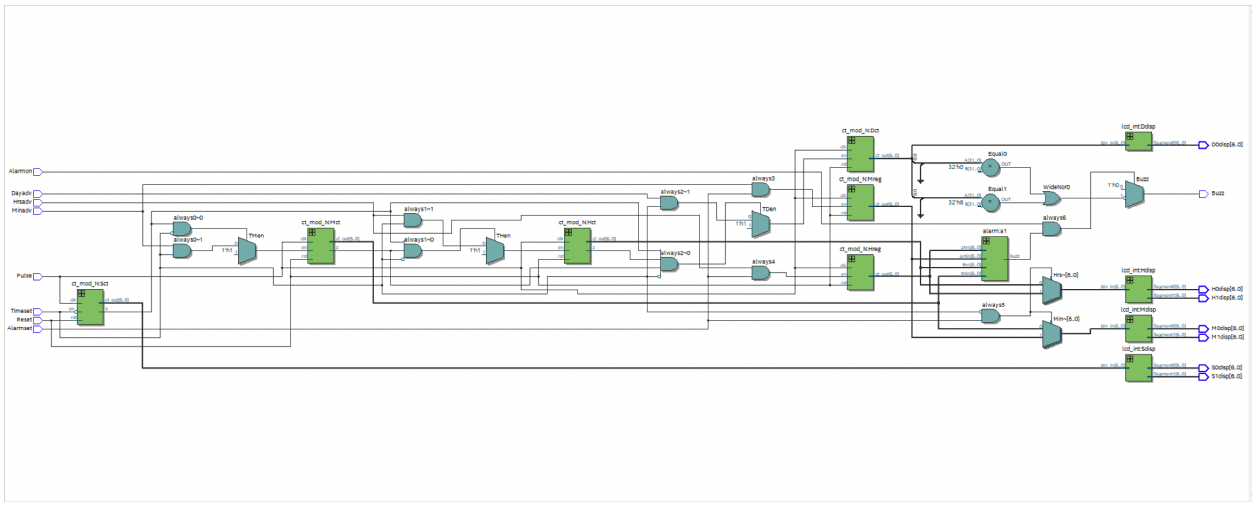
We test when the alarmset and Timeset both 1, what the display is gonna be. As designed, it will display the current time as default. After that, test if Buzz goes high when Alarmon is low. Modify the testbench by setting the Alarmon to 0, and add the monitor to check the states of Buzz. Moreover, we test if the clock will go back to 0:00 when the time passes 23:59. We change the coefficient of ns in the testbench, so the time will be set to 23:59. Then check the output to see the time change.

When alarmset and Timeset both1:

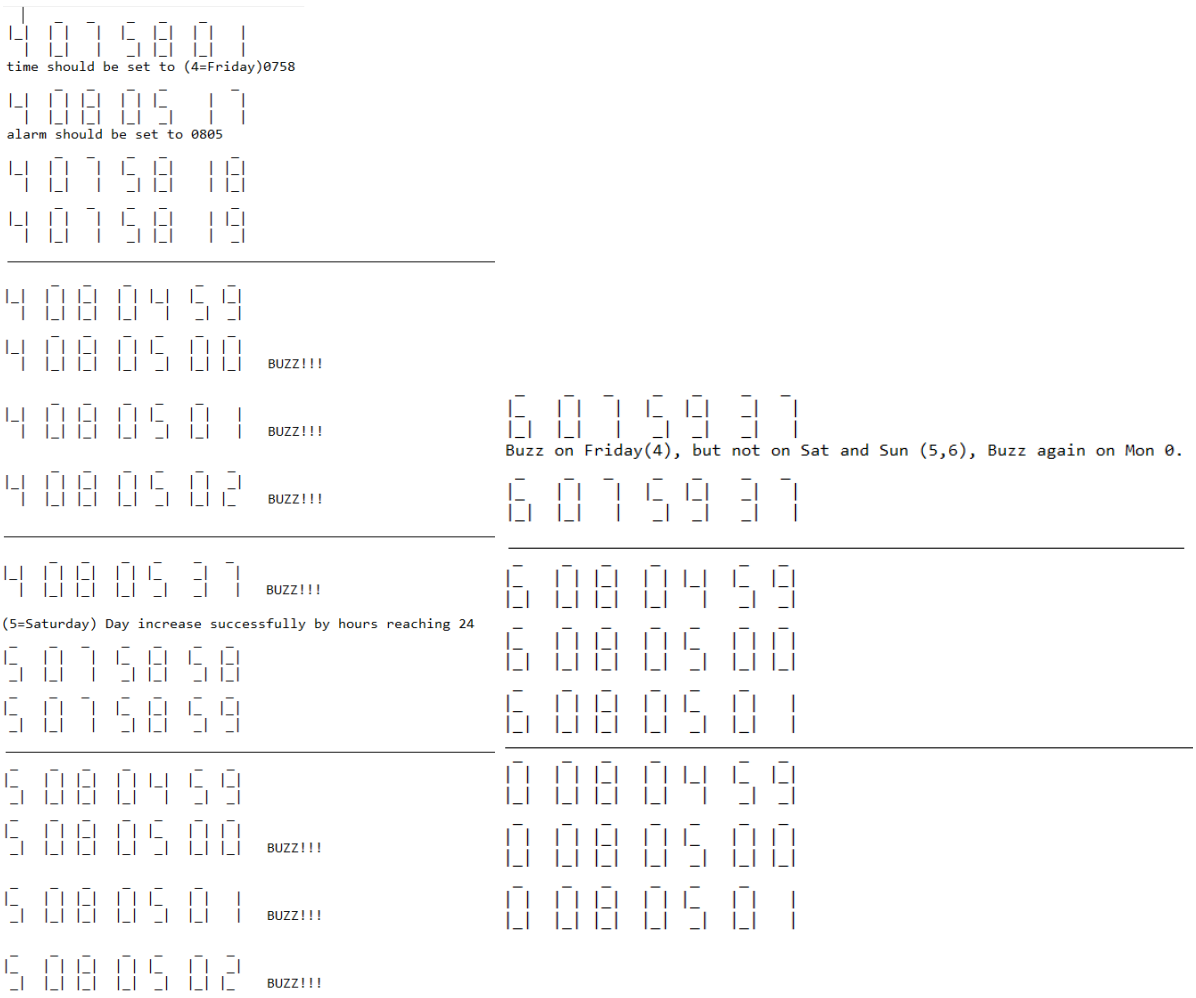


When time passes 23:59:

4. A screenshot of the RTL viewer top level schematic/block diagram in Quartus for Part 2. (2 pts)



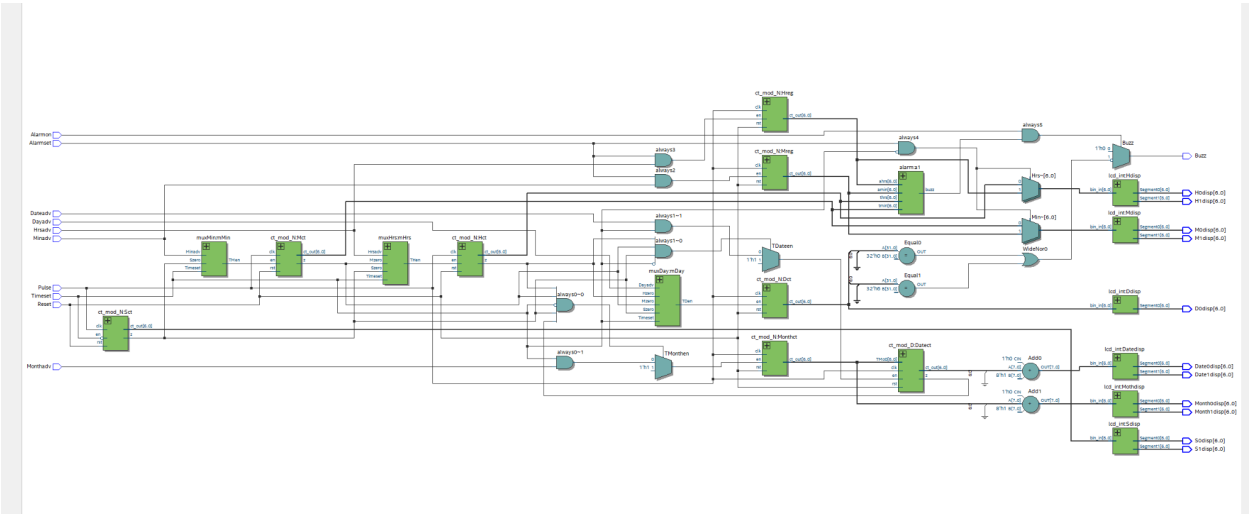
5. A screenshot of the ModelSim transcript showing correct day of the week functionality for Part 2. (2 pts)



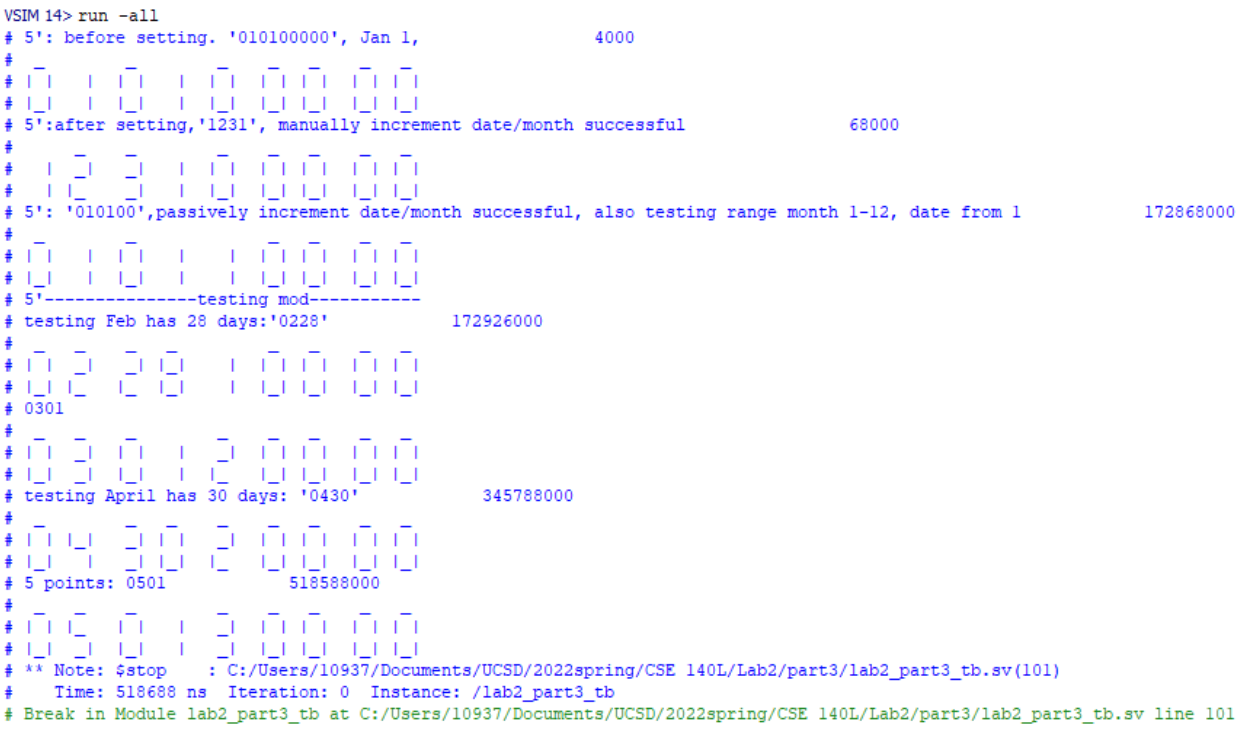
6. A summary paragraph explaining day of the week enhancement and how it was implemented. (3 pts)

We add another counter to count the number of days in a week. On this counter, we pass 7 as the parameter. Then apply the similar logic as we implemented the hours counter. If hrszero = 1, Szero = 1, Mzero =1, and timeset = 0. Or if timeset = 1 and dayAdv = 1. The day will increase. So we add a logic “Den” and pass to “en” in the day counter. For the display part, it is similar to other display modules, we connect the current day and Segment0. However, we left Segment1 blank, since we don’t need it.

7. A screenshot of the RTL viewer top level schematic/block diagram in Quartus for Part 3. (2 pts)



8. A screenshot of the ModelSim transcript showing correct functionality for Part 3. (2 pts)



9. A summary paragraph explaining date enhancement and how it was implemented. (3 pts)

We add a counter to count the number of dates in a month, which has a parameter 28, 30, or 31 according to the value of the month counter. The date will increase if hrszero = 1, Szero = 1, Mzero =1, and timeset = 0, or if timeset = 1 and Dateadv = 1. The month will increase when ,hrszero = 1, Szero = 1, Mzero =1, Dzero = 1 and timeset = 0, or if timeset = 1 amd Monthadv = 1. We implement these functions by adding a logic Monthen. For the display part, we connect the month segments with {Month1disp, Month0disp}; and we connect date segments with {Date1disp, Date0disp}.

10. Describe how you would change your logic if February had 29 days (i.e. it was a leap year). How would your design be modified if you had to keep track of leap years (once every 4 years) - you don't have to implement this functionality. (4 pts)

Add a counter of year and logic to store the years. There is two way to store the years. Directly or break into two centuries and years. If we store it directly, add a module that calculates the number of years mod 4, 100 and 400. If the number is divisible by 4 and (not divisible by 100 or divisible by 400) then it is a leap year.

11. Please mention the contribution of the individual team members to the assignment. (0 pts)

Student 1 - Jikun Wang
Student 2 - Chuning Liu
Student 3 - Wenshen Yu