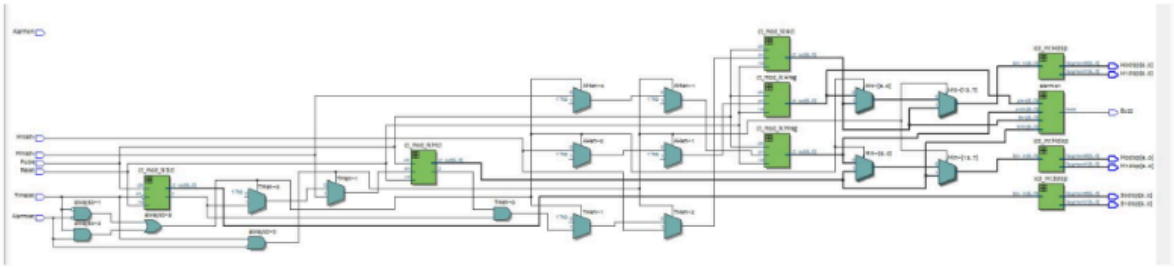
Part 1

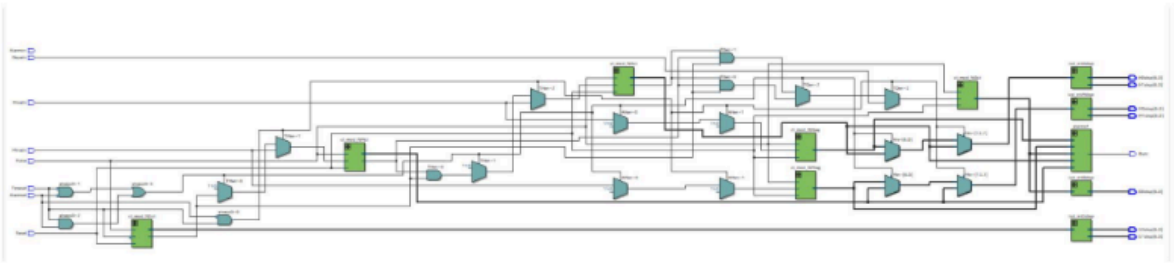
Screenshot of the RTL viewer top level schematic/block diagram in Quartus
Or submit your Mentor Precision netlist file if using EDA Playground (3 pts)



Please include the output file of part 1 testbench with your code submission, and name it “output1.txt” (3 pts). Nothing is required in the writeup for this question.

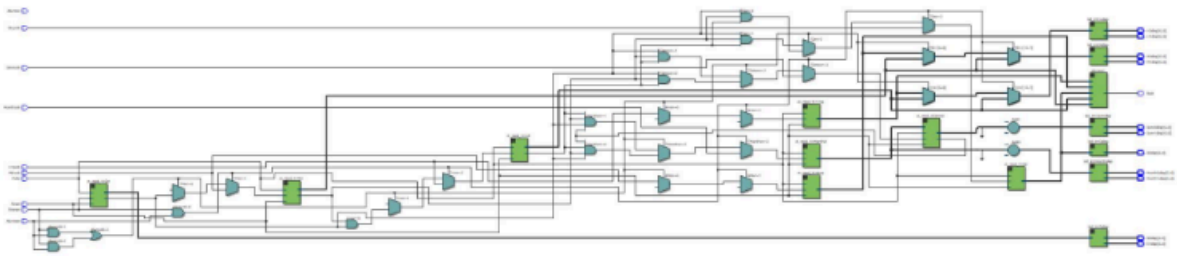
Part 2

Screenshot of the RTL viewer top level schematic/block diagram in Quartus
Or submit your Mentor Precision netlist file if using EDA Playground (3 pts)



Part 3

Screenshot of the RTL viewer top level schematic/block diagram in Quartus
Or submit your Mentor Precision netlist file if using EDA Playground (3 pts)



Please include the output messages (from log/transcript) of part 3 testbench with your code submission, and name it “output3.txt” (3 pts). Nothing is required in the writeup for this question.

1. Please write a summary paragraph explaining how you tested your alarm clock in part 1. (4 pts)

It is insufficient to say “I ran the testbench and it worked.” Please tell us what is happening in the testbench and other testing methods that you may have implemented. Word limit: 200 words.

Test method 1 is simulating on modelsim. The testbench helps us decide what to put in the struct code. Modelsim said “3 connections expected, 2 found only”. We readded empty D1display. The most common typos was matching variable names. We had an issue with month parameter. We want to pass in month as input not a constant. We compare our output txt files with the given .txt files.
Test method 2 is running on gradescope. It makes sure the folder hierarchy is correct and everything compiles/passes.
Test method 3 is synthesis on Quartus. We got errors about assuming inputs. We must initialize each variable even unused ones. We got errors about not purely combinational logic. Combinational cannot remember what a variable was. We have to initialize variables for each condition (if statements). When it said Alarmset, we don’t use TMen and THen since we want to display alarm time not regular time. But we must set TMen and THen =0. Leaving connections empty gives errors.

2. Please write a summary paragraph explaining the day of the week enhancement and how it was implemented. (4 pts)

Word limit: 200 words.

We modified “alarm.sv” by introducing a new input “days”. We modified the output statement “buzz” by anding it with (days >=0 && days <= 4), so the clock only buzzes when it is alarm time and current day is weekday.
We modified “struct_diag.sv”, introducing a new input “Dayadv” which is used when user is setting time. We added a new output D0disp. For the internal connections, we added a logic[6:0] TDay (represents the clock’s day in the week) and Day (used as bit_in for display).

We added a day counter in “struct_diag.sv” which takes in the parameter ND=7 (because there are 7 days in a week). The enable bit is TDen; output is TDay; z is Dzero. TDen is Hzero && Mzero && Szero, because the day counter only updates when it is the end of the day (which is 23:59:59).

We also added some features for setting time and alarm.. While the user is setting the time, TDen is Dayadv. While the user is setting the alarm, TDen is 0.
For the display, we used the submodule lcd_int, making .Segment1(), .Segmen0(D0disp), because day only has one bit in decimal.

3. Please write a summary paragraph explaining the date enhancement and how it was implemented. (4 pts)

Word limit: 200 words.

We added inputs Monthadv and Dateadv (used when the user is setting time) in “struct_diag.sv”. We added the display feature for date and month, which has bit_in of Date+1 and Month+1 because we used 0 for January in the logic, but in display it should be 1. Similar for date.

For the internal connections, we added logic[6:0] TDate and TMonth, Date and Month (used as bit_in for display).

We added a new file ct_mod_d (used as date counter), which is similar to ct_mod_N, but instead of taking a constant parameter, we used the input TMonth and determine the mod number based on it, because different months have different number of days. So in “struct_diag.sv” we added ct_mod_d Datect which takes TDateen as enable bit; the output is TDate and z Datezero.

We added a month counter in “struct_diag.sv” using ct_mod_N, which takes in the parameter NM=12.

We also added some features for setting time and alarm. While the user is setting the time, TDen is Dayadv, TMonthen is Monthadv. While the user is setting the alarm, TDen and TMonthen are 0.

For the display, we added Monthdisp and Datedisp based on lcd_int.