Aditya Gopalan
adgo4176@colorado.edu
109052749
11/6/2019

ECEN 2350 - Lab 2: Write-Up

Problem statement:

In this lab, we were able to use our knowledge of latches, clocks and counters to allow the DE10lite board to display the first 99 days of the year. The goal was to allow the board to display the date of the year corresponding to the actual date count of the number of days it is into the year itself by displaying the date in 1Hz intervals on the 7 segment display. To do this, we had to use latches and resets to ensure that once the 99 days had cycled through the board display, it would restart at day 1 as January 1st. To do this, we had to use a reset using KEY0 that allowed the latch to toggle when the button was pressed. When this was done, the frequency at which the time interval of date showed increased from 1 Hz to 10Hz. This, essentially, means that the date would cycle from 1 to 99, 10 times as quickly as before. Another one of the goals was when the the date reset to the first day, we needed to ensure that if the fist number of the date, hypothetically, January 1st as 01/01, we needed the 0's to be displayed as blank on the 7 segment display. In this lab, we made sure that the LED1 is a clock frequency that increases, and LED0 check and shows if the reset was active and toggled or not. In the end, we were working on the extra credit which was the idea that we would make the clock be 5 times as fast rather than 10. In other words, we tried to make the clock go at 5Hz rather than 10Hz when the KEY0 was pressed.

Theory of operation:

One of the first problems that we had to figure out for the lab was to convert the 5MHz clock speed to the 1Hz for which the display would show the date from 1-99. To do this, we used a parameter divide_counter to allow us to vary the clock speed based on what we wanted it to be. In this case, it would be 10Hz, but as seen in the extra credit, we would make it 5Hz. To do this, we would measure at when the positive edge of the clock would occur and would add one for everything the positive edge of the clock would be there. Once we had set the divide by value, we would count the times when the divide by value reached that and would then toggle the output clock value. Due to the fact that the board clock speed was 5MHz, we set the parameter to be 5 million to represent the number of Hz in 5MHz. The after the clock would run every 5 million times, then the clock would be toggled to then get out 1Hz output. We would change this number accordingly to what we wanted it to be based on the situation such as the extra credit.

Next, we had to solve the problem that we needed to display the HEX4 and HEX5 values, for which they represented our date counter for which the display would show what date in the numerical order between 1 and 99 they are in the calendar year. To ensure that we were able to display our HEX4 and HEX5, we had to create a module that took the clock and reset signal from the KEY0 button and used a block to check to see if the reset signal was low. If the reset signal was true, low, then it would set the 10's place of the display date to nothing and set the ones place to 1. This would show in the display as the first date as January 1st. If this was

Aditya Gopalan
adgo4176@colorado.edu
109052749
11/6/2019

false, then it means that the 10’s place would increment by one suggesting that the date would be switching from 9-10, 19-20, etc. Next, we made sure using the counter and bigcounter registers that once the number got to 9 for the counter (1’s place), that the next number would be 0. In the case of the bigcounter (10’s place), if it reached 9, then the next number would be blank so that when the clock resets, it would not show 01, it would just show 1 on the display. To print the 10’s place we had to check if the number was 9. If it was, we toggled the positive edge to ensure we got the right display for the next number. Finally, to print the date counter we had printed using print7seg module which takes in the BCD and outputs value of HEX on the FPGA board.

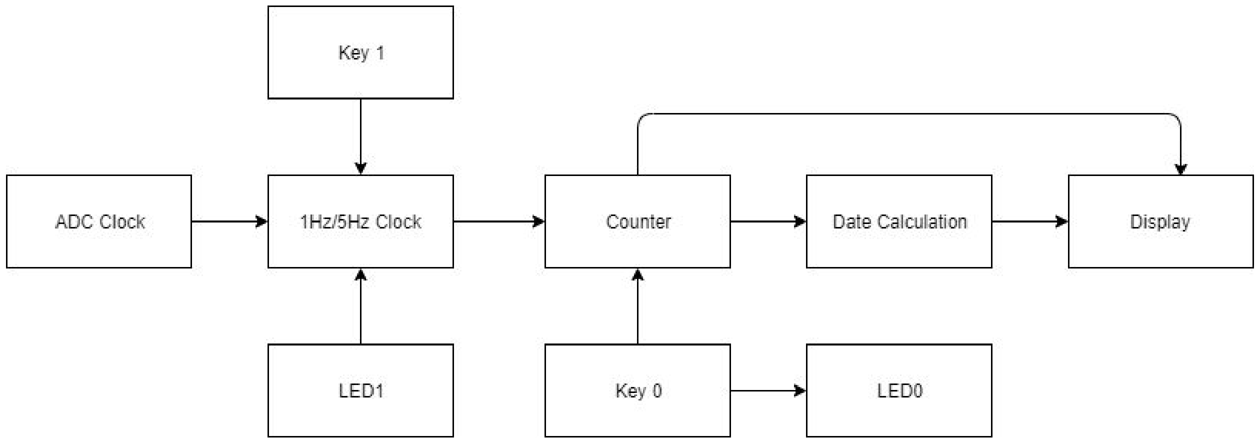
Finally, for the date itself we had to consider that each month had a different number of days in the month, January - 31, February - 28, March - 31, and April - 9 since day 99 is April 9th. To allow the counter to know when to switch the month we needed to assign the days to a certain value of dateones and datetens registers. For January, we had to ensure that the counter was able to count the ones and tens place to January 30th without issue. However, we couldn't go over 31st since there is no January 32nd, so we had to add an if block to allow an exception to say that any day from the times that the counter goes to 30 it would switch to a new month. But weit would include the 31st of both the January and March before it switched to the new month. To make things less confusing, we let the counter count till the 29th of the month for January since that way it would follow the counter correctly. However, it would get weird once it hit January 30th since there isn't a full 10 days in the 30’s for January, we set the big counter (10’s place) to reset to 0 and switch to the new month. However, we couldn't forget about the 30th and 31st of January and March so we had an if statement saying that if the date counter is less than 30, then it would continue to count up to the 31st and only then reset to February 1st and continue from there. We repeated this process for each of the months. Once we got to April, we didn't have to worry since days 90-99 are all in April, we just needed to let the counter run until it got to day 99 and then the loop will start all over again to repeat the process for the next cycle of January 1st - April 9th. This does sound confusing, however it is pretty simple once you get how to ensure that the counters don't go too far before they need to reset for the days in the months.

In the extra credit of the lab, we did a fairly similar to what we did for the 10Hz however we needed to change the delay and the clock speed to ensure that once we pressed the button it would trigger the pos and neg edge at the correct times. The process was similar to what was mentioned before.

In the testbench, we had to include another create_clock and toplab2 verilog files because we had not included a reset clock in either of these modules since the board had already done this for us. However, since the testbench cannot read or know what to do with the latches that it is given, we had to add some extra points to allow the testbench to be able to simulate the clocks without having to try and do it without the latches and the pos-edges of the counter clocks.

Aditya Gopalan
adgo4176@colorado.edu
109052749
11/6/2019

Block diagram:



Hierarchy of source files:

Top Level File: toplab2.v
Lower-Level Files: create_clock.v, create_clock2.v, counter.v, print7seg.v, toplab2test.v, date.v, and testbenchlab2.v

Testbench operation and output:

The creation of the testbench was the hardest part of the lab by far. One of the biggest issues was the conversion from the ADC clock to be 5MHz for the delay to allow the text file to display what the date should be. When we created the testbench, we had to initialize out ADC_CLK_10 and KEY0 and KEY1 as 0. Once that was done, we had to ensure that the ADC clock was alternating between on and off to ensure that the output was being displayed every 1 second. At the end of the test bench, we had to show the output of what it was referencing. To do this we had a monitor block that displayed HEX0-HEX2 and the LEDR to show what the date was being outputted after it had done all the computations in the other modules. One of the biggest issues that we had for this was the fact that we weren't able to get this to display 99 times. It would display once and be done with it, while the FPGA board was working perfectly fine. We figured out that the reason that this was happening was the fact that our create_clock module did not have a reset clock. In the case of the board, it knew that the module needed to be reset every Hz but the testbench did not know that so we needed to ensure that all the modules had that reset block to ensure that the latch was being toggled for all modules so that it was able to display 99 times.

Aditya Gopalan
adgo4176@colorado.edu
109052749
11/6/2019

Summary of project success, what works and what doesn't:

In the end, we were able to get the project to work in completion. We were not able to get all of the extra credit done with since we didn't have enough time to work on it. However, we were able to get the 5Hz clock speed done with relatively easily. One of the biggest challenges, as mentioned above, was the testbench. We got the code for all the other modules to compile and run perfectly on the board, however, we were having a lot of difficulty with getting the testbench to print more than once. Turns out it was a small error that we eventually got fixed, but the majority of our time went into that debugging. The one thing that we weren't able to get done was the leap year extra credit. The main reason that we weren't able to figure that out was the lack of time since the majority of our time went into the debugging for the rest of the modules and testbench.

Overall in this lab, we learned a lot about the structure and the way latches and BCD's worked in Verilog. Though it was challenging for a lot of the ways to code the FPGA board, it was helpful to see the way that logic was able to play a role in this code. The most frustrating part of the lab was the fact that some of the random solutions that the board would produce would be wrong. However, I feel that if I had a slight bit more time to work on the lab, I would have been able to eventually figure out what the issue could have been and could have been able to solve it.