Settable Audio Alarm Clock

Robert Brown, Jacob Feldpausch, Jacob McCarthy, Nicholas Mojares
Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
e-mails: rbrown3@oakland.edu, jacobfeldpausch@oakland.edu, jacobmccarthy@oakland.edu, nmojares@oakland.edu

Abstract - In order to gain an understanding of how to develop a digital system, a 24-hour alarm clock that can be set to any time is being developed that will sound an alarm at a time set by the user.

I. Introduction

This report will describe in detail the 24-hour settable audio alarm clock that was created. This project aimed to use VHDL in order to create a working digital system incorporating adders, counters, multipliers, comparators, seven segment displays, and more. The alarm clock will be programmed onto a Nexys A7-100T development board and will be programmed using VHDL.

Our motivation was to apply the concepts studied in ECE 2700–such as VHDL code and digital circuit elements–to real-life, hands-on digital design and implementation.

II. Methodology

A. *Design Methodology*

The alarm time is set using the 16 switches on the board, encoded in BCD. This allows the user to quickly input the number they want on a per digit basis. The alarm time will also be displayed in decimal 24-hour time on one set of seven segment displays. The current time will be displayed on another set of seven segment displays. When the current time reaches the set alarm time, an auditory alarm will be sounded which can be stopped using a button on the board. This will be done by comparing the current time (counted in minutes) to the time imputed by the switches.

A PWM signal will be used to generate the alarm sound and will be connected to a buzzer using the PMOD ports on the Nexys board to be plugged into a speaker. The BCD signal from the switches will be converted to be displayed on the seven segment displays, while also being converted into binary and then converted to the total time in minutes. This will be done by multiplying the hours by 60 to get it in minutes where it will then be added to the minutes time to get the total minutes for the alarm. The current time will be tracked by downclocking the 100MHz internal clock to once per minute. This number will then be compared against the alarm time. The once a minute signal will also be used to separately track the current time in standard 24-hour time to be displayed on the seven segment displays. The once a minute clock will be fed into a 60 counter in order to track the current minutes in the hour. Every time the 60 count has completed that will trigger a 24 counter in order to track the current hour in the day. Both of these counts will be then converted into BCD and finally converted to be displayed on the other set of seven segment displays.

III. Experimental Setup

In order to test the functioning of our digital alarm clock, we programmed the Nexys A7-100T board using VHDL. After implementing the code onto the board, we ran sample tests, setting the alarm to go off after a short period of time to see if everything ran properly. After encountering a problem with the alarm, we analyzed the issue in order to determine which part of the code was causing the issue–the counters, the comparator, the multiplier, etc. Ideally, the user will input a time for the alarm to go off, which is displayed on one of the seven segment displays; and when the current time (displayed on another set of seven segment displays) reaches the set alarm time, the alarm will sound. Achieving this end was our goal in experimental testing.

IV. Results

We encountered an issue where the current time stayed at 00:00 for two minutes, and then changed to 00:01. The displayed current time was then essentially one minute behind the actual current time. We determined this was due to an asynchronous reading between a binary to bcd conversion that was feeding the display and a counter output. This was fixed by adding a 1 clock cycle delay between the activation of the counter and the activation of the binary to BCD converter. Another issue encountered was due to the fact that the z output of counters were being used to activate other counters, those counters would be activated for the entire duration of the first counter being high. A monostable circuit was added to ensure signals did not remain high for more than one clock cycle. After going through many iterations and debugging issues, the circuit functioned as intended.

Below is a link to a demo video showing the alarm going off after one minute.

https://www.youtube.com/shorts/NEbo9oE8Lkk

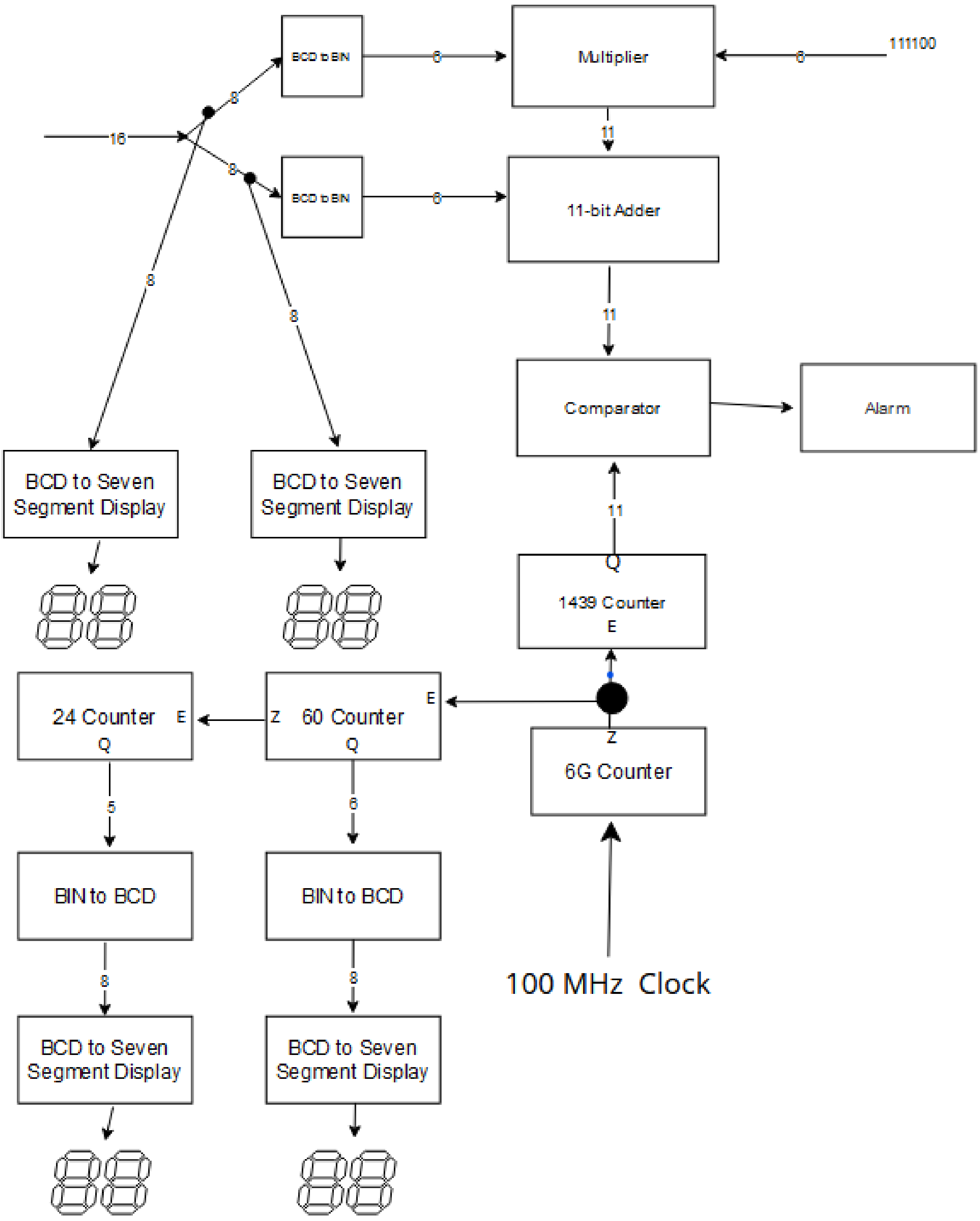
Conclusion

After collaborating with our group and combining our code for each component together we were able to make a functioning alarm clock, using the Nexys Board. This alam worked based on a 24 hour (or standard time) clock which allowed us to simplify our design as well as fit it within the 8, 7 segment display. We were also able to produce an audio for the alarm using an external audio device from the board's output. During the work on the finalized version of our code we ran into a few problems. The first being an error that involved the clock, and the Resetn. We were able to determine that we forgot to add the clock signal to one of our FSM (finite state machines), so it would seem that VHDL was trying to use the Resetn as a type of clock signal but could not resolve it as one, this was simply fixed by added the clock, however it was a interesting learning experience because none of us had encountered the error before. The other problem we encountered was a delay from the current time and the time being displayed. We knew this because we timed how long it should take, as well as the fact that the buzzer was going off as expected. The problem ended up being that the binary to BCD module was not outputting the signal from our counters as discussed previously in our results. For each of these problems we were able to utilize synthesis, simulation and testing on the board to determine the problem, as well as solve it.

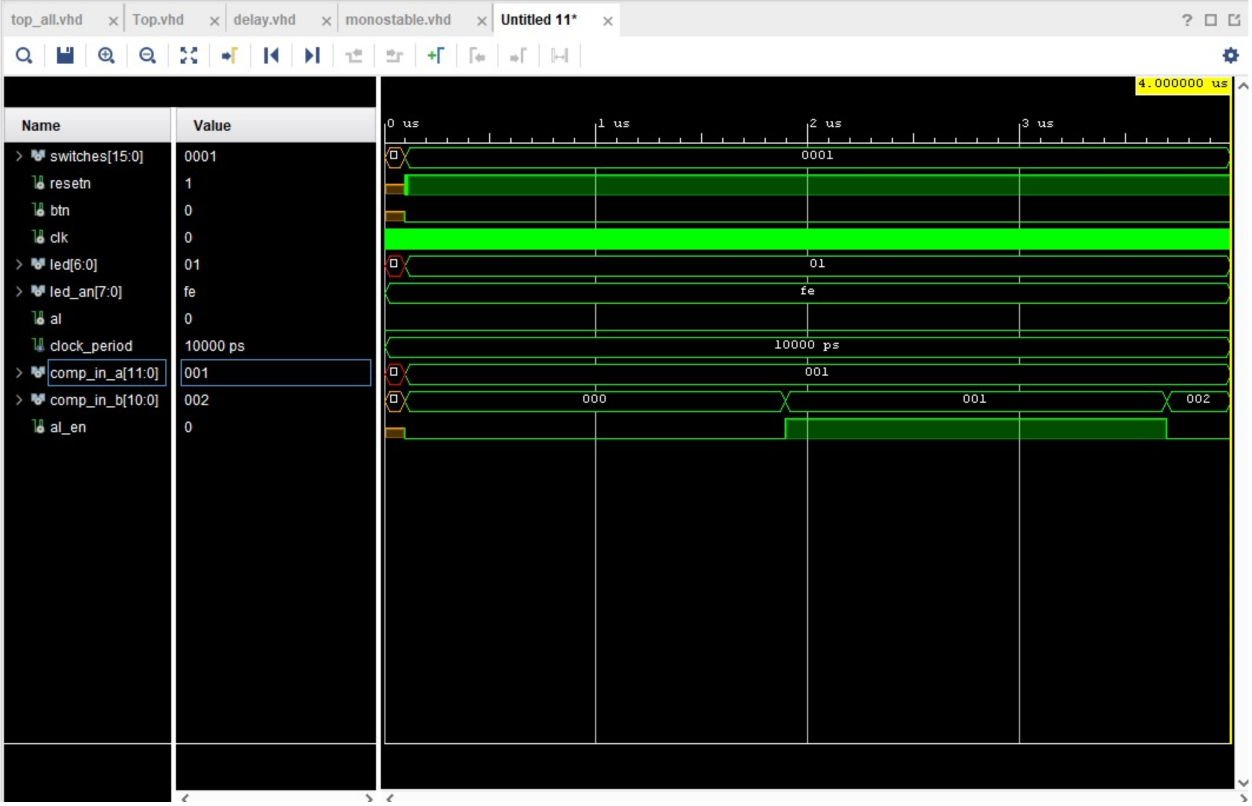
For our design of an alarm clock there are some areas we could improve. The first improvement we could have made would be allowing the time to be set. In our current design, the time starts at 0 when the reset button is clicked. We could potentially include this function with a different mode and use the switches the same way the alarm is set. We would need to include some kind of memory in order to do this. Another possible improvement would be a different way to set the alarm time through a different input like a computer. This would give the advantage of not needing to convert the desired time into BCD.

By accelerating the speed of the system to a microsecond scale, it could be easily tested using a testbench file. The alarm time is set to 00:01 initially and the system is simulated for several microseconds, once the current time reaches 00:01, the alarm is enabled until the current time reaches 00:02 and it turns off. A screenshot of this simulation is included below.

Overall designing and implementing this design taught us new ways in which the Nexys board works and we tackled new problems by doing so. We learned how to serialize an output for our 7 segment displays, and we learned how to send an external output from the board. Those of which have not been done so far in our lab.



Block Diagram of the Entire Alarm System



Name	Value
switches[15:0]	0001
resetn	1
btn	0
clk	0
led[6:0]	01
led_an[7:0]	fe
al	0
clock_period	10000 ps
comp_in_a[11:0]	001
comp_in_b[10:0]	002
al_en	0

Accelerated Simulation Results Showing Functionality of the System