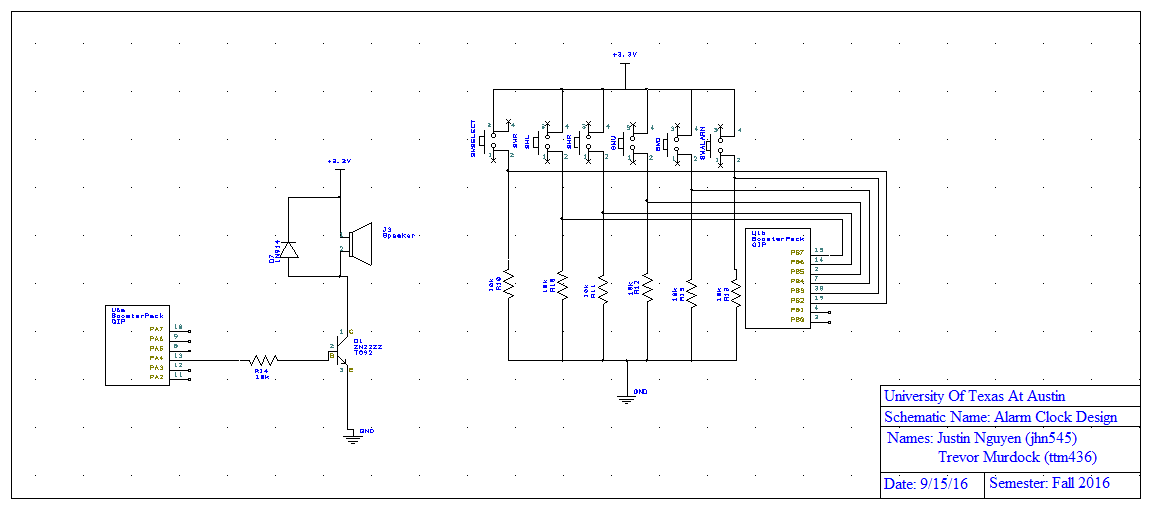
1. **OBJECTIVES**

See the final requirements document.

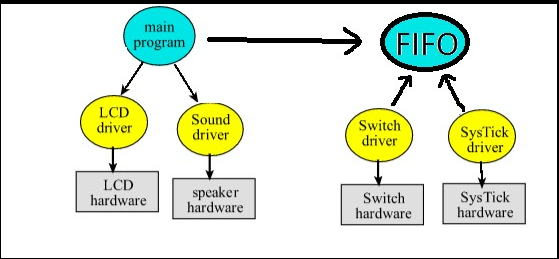
1. **HARDWARE DESIGN**



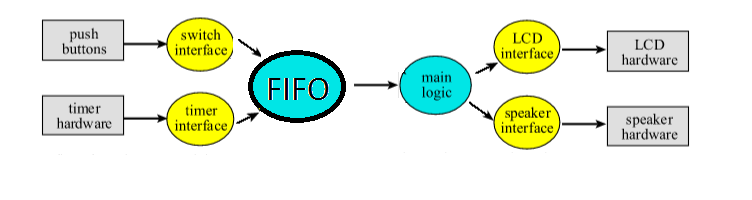
*Figure 1: PCB layout of our alarm clock. 6 switches (left, right, up, down, select, alarm).*

*Speaker circuit is also shown. See the .sch file for details/better view.*

1. **SOFTWARE DESIGN**

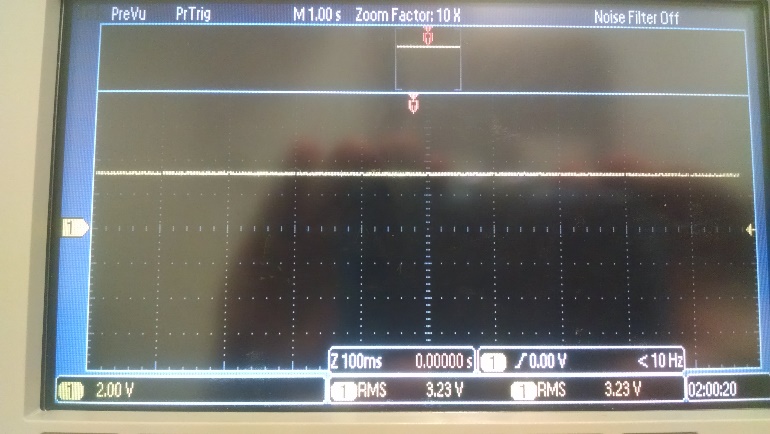


*Figure 2: call graph*

**

*Figure 3: Data Flow Diagram*

1. **MEASUREMENT DATA**

****

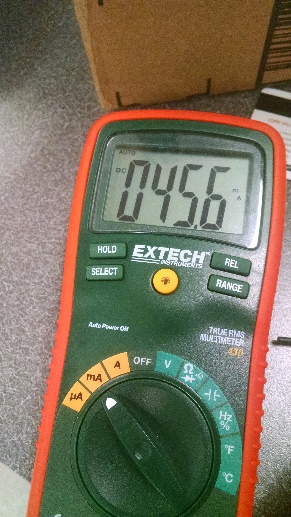
*Figure 4: 3.3 RMS magnitude of supply voltage versus time*

***4.2***

******

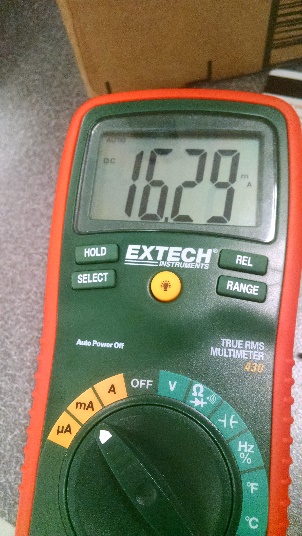
*Figure 5: Speaker voltage during an alarm*

***4.3***

******

*Figure 6: Current required to run the alarm clock with the speaker on*

***4.4***

******

*Figure 7: Current required to run the alarm clock with the speaker off*

1. **ANALYSIS AND DISCUSSION**

**5.1 Give two ways to remove a critical section.**

You can use friendly code (bit specific addressing) or use a communication queue like a FIFO. A bad way to remove a critical section is to disable interrupts.

**5.2 How long does it take to update the LCD with a new time?**

**5.3 What is the disadvantage of updating the LCD in the background ISR?**

The time it takes to return to the main would be too long and increase the chance of critical sections. A long ISR could cause the system’s timing to be off and the system wouldn’t be real time.

**5.4 Did you redraw the entire clock for each output? If so, how could you have**

**redesigned the LCD update to run much faster, and create a lot less flicker?**

We did not redraw the entire clock, but we did redraw both the hour hand and

minute hand. We could only redraw the hands when they are supposed to move.

* 1. **Assuming the system was battery powered, list three ways you could have**

**saved power.**

You could decrease the interrupt frequency (perhaps from every second to every

minute), only update the screen when needed, or use a speaker with a higher resistance (this will lower the volume) and decrease the frequency of the alarm.