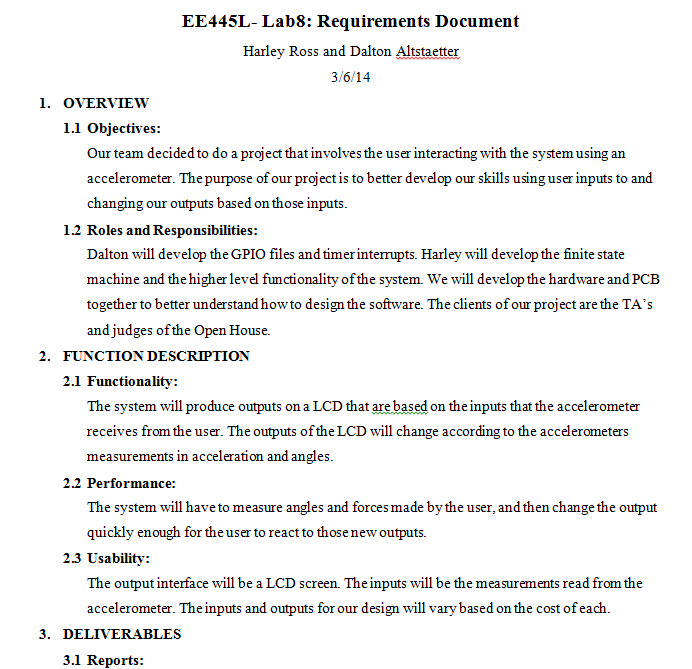
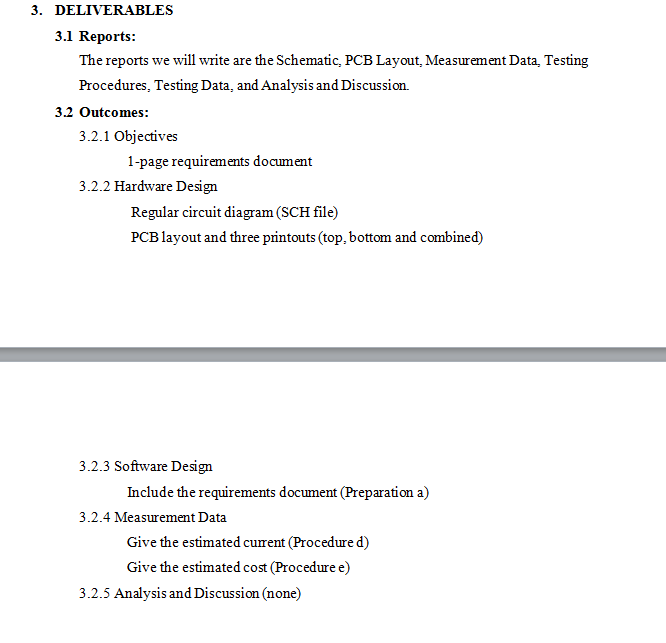
**EE445L – Lab11: Final Embedded Lab**

Harley Ross and Dalton Altstaetter

5/2/14

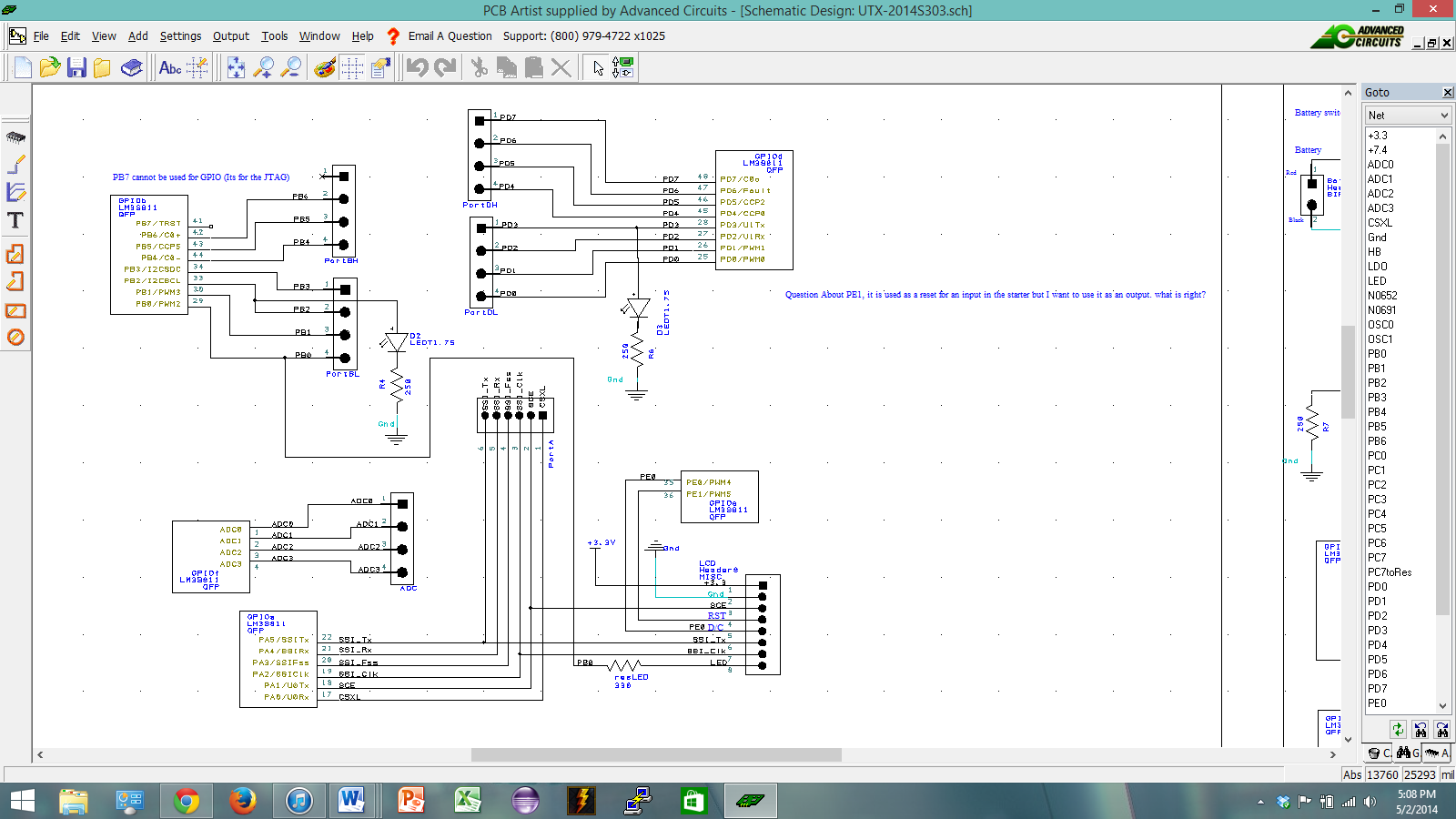
**OBJECTIVES**

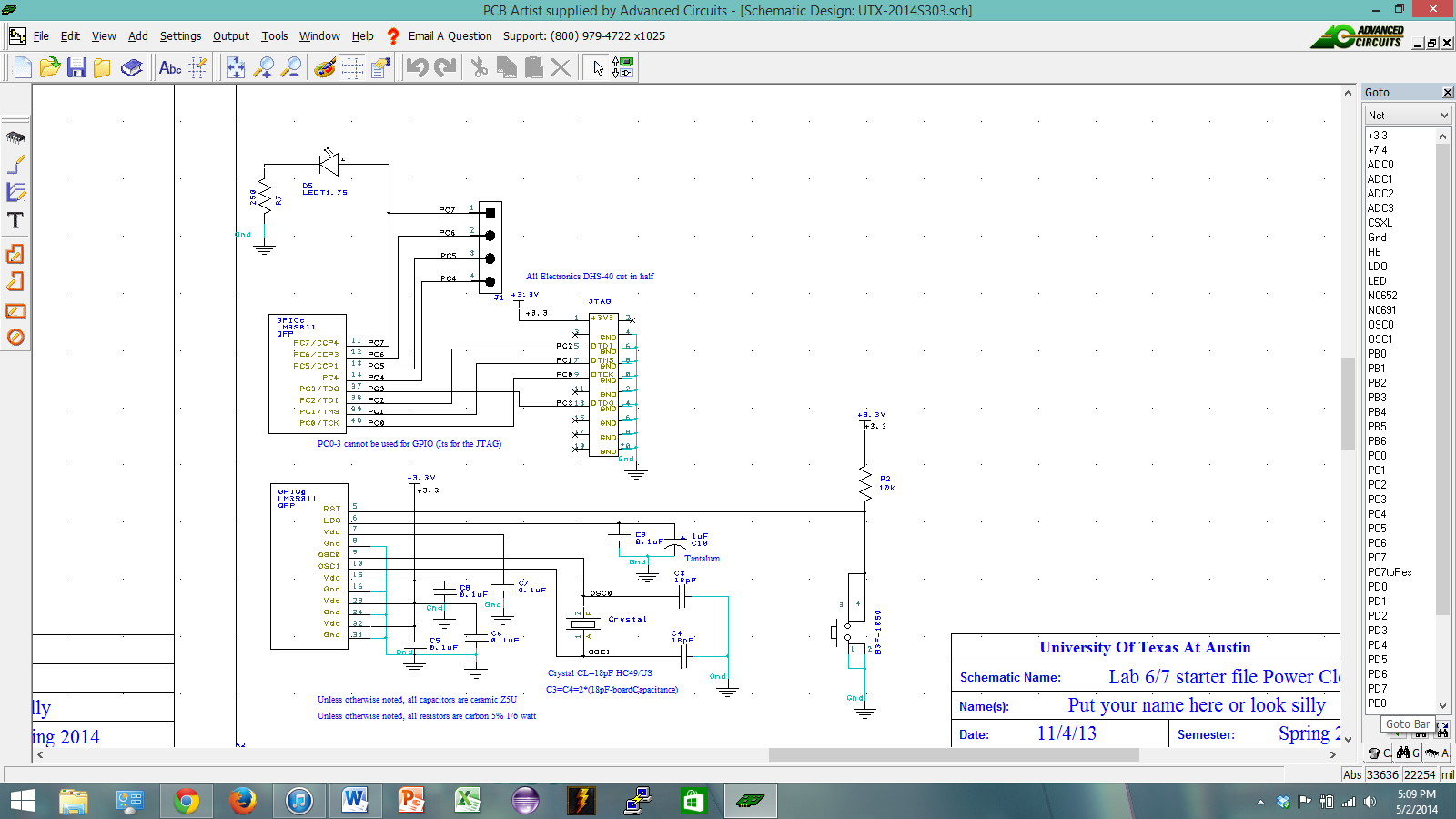


****

**HARWARE DESIGN**

SCH





**SOFTWARE DESIGN**

Accelerometer

The accelerometer software uses the Serial Peripheral Interface (SPI) in mode SPO = 1, SPH = 1 at a SPI clock speed of 10 MHz to communicate with the 9-Degree of Freedom (DOF) LMS9DOF which has a 3-axis accelerometer, a 3-axis gyroscope, a 3-axis magnetometer, and a temperature sensor to communicate with our LM3S1968/LM3S811. The accelerometer software just used the accelerometer since the gyroscope data wasn’t used to calculate orientation. The LMS9DOF stores the x, y, and z axis data in 3 separate 16-bit registers as a two’s complement number. The accelerometer software continually reads these registers at a rate of 760 Hz as set by the initialization of the accelerometer and sends these three 16-bit registers to the microcontroller to be processed to find orientation. To find the orientation of the accelerometer (in fixed-point ranging from 00.00-90.00 degrees with a resolution of Δ = 0.01) I used the C library function from “<math.h>” to call “atan2(y,√ x2+z2appropriated (for my fixed-point math) to turn it into the appropriate angle, this was done for each axis. I then used these angles to set thresholds for the expected motions of the golf swing, the backswing, forward swing, and follow-through. Upon the end of the follow-through I captured the x and y axis accelerations (measured in units of “g’s", earth gravitation units) and normalized them to convert them to vectors to pass to the LCD display, where it would then repeat the cycle for the next swing, if necessary.

Nokia\_LCD

The software to run the Nokia LCD used SSI0 to initialize and send signals to the LCD. From there, bitmaps were created to print an outline of the boundaries for the miniature golf course. This outline was copied to a two-dimensional array that holds all of the boundary locations and ball locations. The ball two dimensional array was reprinted to the screen to show ball movement. The moveBall function called for an X and Y vector component values. From those values, the ball would move up or down by one depending on the Y component and right or left depending on the X component. The function would keep preprinting the movement components became zero. If the position of the ball was near a wall, the function would reverse the direction of the ball to replicate bouncing. The function would also detect the hole where the ball user wanted the ball to go. If the ball’s location was near the hole, the LCD would print the next level and restart the balls starting position.

**ANALYSIS AND DISCUSSION**

Our end goal for the project was to combine the accelerometer software with the LCD to create a miniature golf game. Our team spilt the work up between the LCD and the accelerometer to get the work done faster. Upon merging our code, our team found bugs with the merging that lead to problems with the LCD. Our PCB design has only one SSI port so we planned on having to reinitialize the ports every time we needed to switch between the accelerometer and the LCD. Our LCD ended up being shorted when the LM3S811 chip was soldered on. Because of this, our team was not able to complete our project using the PCB.