RGB LED Control with PWM from External Sensors

Richard Pinto, Adrian Sinishtaj, Dustin Cankar
Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
richardpinto@oakland.edu adriansinishtaj@oakland.edu dcankar@oakland.edu

Abstract— This project incorporates many aspects learned over the duration of Computer Hardware Design. Such as reading data from the built in accelerometer on the Nexys A7 FPGA and implementing the data found. Also the knowledge obtained in advanced use of a Finite State Machine will be critical to this project. Firstly a Finite State Machine with the input from a user interface through the use of switches will dictate which mode the system is operating in. After the mode has been selected data will be read from either the onboard Accelerometer or Temperature Sensor. This data will then be used as an input for the control of the pulse width modulation of the onboard RGB LED.

I. Introduction

This project report covers the design and experimentation of PWM controlled LEDS through the implementation of the onboard switches, accelerometer and temperature sensor. It also covers findings related to FPGA implementation and debugging. Many logic components were used from the course completed over the semester, such as registers, counters, logical computation units, multiplexers, microcontrollers, a temperature sensor, a PWM controller, and two FSMs to control each sensor . Results of experimentation, implementation and simulation of controlling the onboard LED colors with the help of the accelerometer or temperature sensor will be discussed in detail to understand future improvements.

II. Methodology

The circuit consisted of four primary components, the light emitting diode (LED), I^2C (Temperature Sensor), SPI (Accelerometer), and a pulse-width modulation (PWM) controller. A pulse-width modulation signal either controlled by an accelerometer, temperature sensor or switches controls the light intensity and color of the onboard RGB LED. The use of either component is decided by the onboard switches. The temperature sensor will provide a signal to the PWM controller in order to display a color corresponding to the PWM input, in this instance the use of red, purple and blue will be used to dictate room temperature.

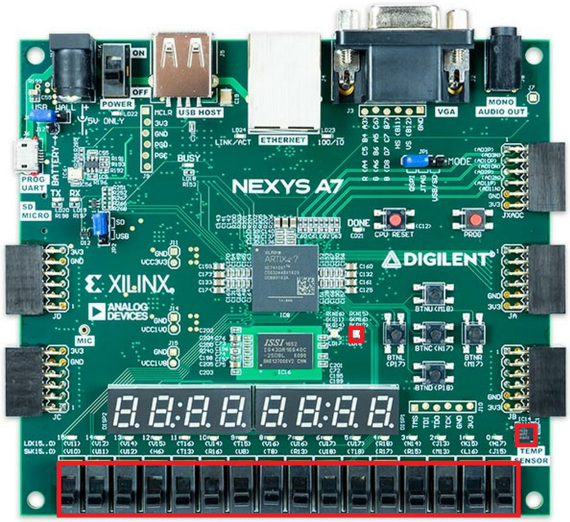

Figure 1: Nexys A-7 100-T Board

The outlined onboard components are the primary elements that the group decided to focus on for the project.

A. LEDs

Several colors can be produced using different combinations of red, green and blue LEDS, which can be achieved by varying the light intensity through pulse width modulation. The Nexys 7 board has two tri-color LEDs which have three input signals that drive the smaller internal LEDS: one red, one blue, and one green. Driving these signals corresponding to one of these colors illuminates the internal LED. The tri-color LED will emit a color depending on the combination of internal LEDs that are being illuminated. For example if green and red

signals are driven high and blue is low, the tri-color LED will emit a yellow color.

B. I^2C

To alter the duty cycle of the onboard LED using the temperature sensor built into our board, a 2-wired synchronous serial interface is required in order to input the proper values to the PWM signals. Data is read/written via a register based interface involving two bytes per bus transaction. The output is 16 bits signed FX which can measure temperatures between -40C and 150C. With logic signal SCL (Serial clock) and SDA (Bi-directional serial data) the FPGA board (master) gives the temperatures sensor (slave) on the bus while the slave device provides a matching address for the master. The slave address is conigurable in this instance the address was altered to ultimately match the input of the PWM component 4 bit in order to achieve functionality.

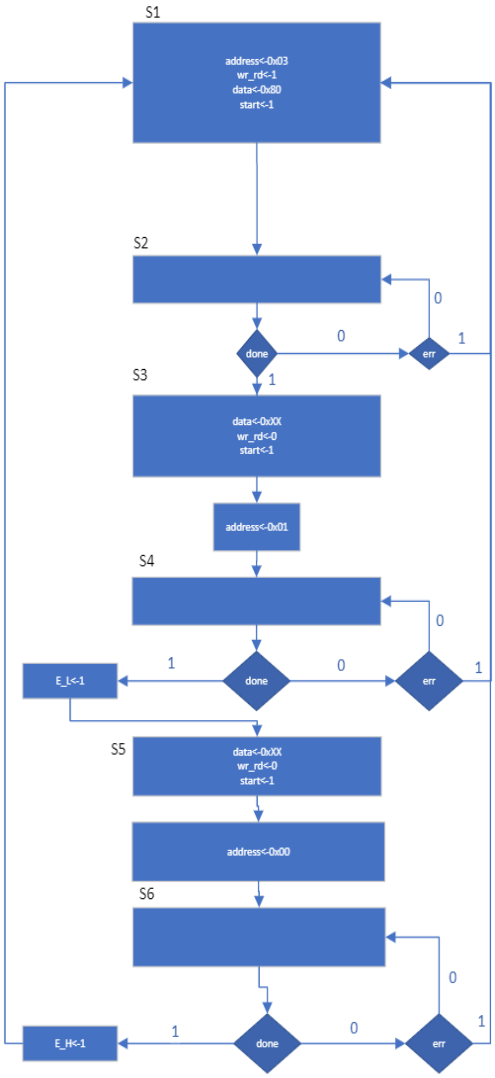

Figure 2: Temperature Sensor FSM

The following Fsm issues commands to configure the ADT7420 register and then read two of four 8-bit ADT7420 registers. This allows the component to read high and low temps mapped to 8-bits.

C. SPI

A 4-wired synchronous serial interface consisting of a serial clock, master input/output and chip select is used as another input for PWM. The SPI, a slave device, reads/writes data via a register-based interface with an output resolution of 12 bits. The controller is able to handle SPI communication based on address data and write/read decisions. Asserting a start signal initiates the

transaction, in this instance a signal decided by the use of the onboard switches. After the operation is complete, the signal done is asserted for one clock cycle. The controller can handle data consisting of 8-bits matching with the input signal of the PWM controller. In order to get different values altering the orientation of the FPGA board the use of this component is necessary.

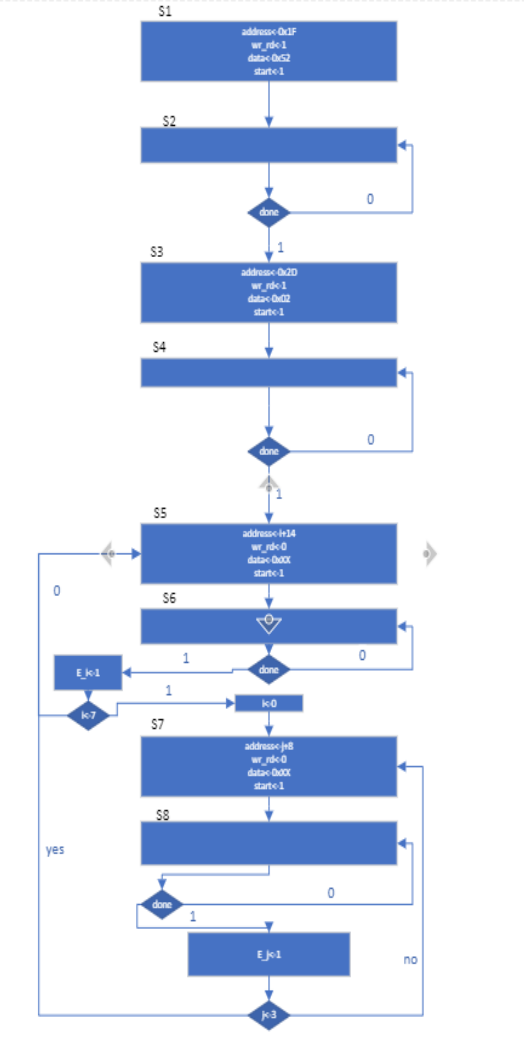

Figure 3: Accelerometer FSM

The following FSM issues commands to configure the two ADXL362 registers and reads cyclically from one of 6 8-bit ADXL362 registers. This data is fetched on the output register so we are able to read low-precision 8-bit X,Y, and Z measurements.

D. PWM

To use the onboard tri-color LEDS, a PWM controller alters the RGB color by varying the voltage to the LED using pulse width modulation.. This is done by converting an analog value, the average pulse width over a 3dB interval, into a digital value. Digital control is used to generate a square wave, a signal decided by the percentage of duty-cycle or the duration of time between the designated signal being changed on/off. This signal can enhance or dim the brightness of the LED. In the case of this project, it is used to display all the outputs of the sensors and switches. In order to display colors appropriately bits are mapped to the RGB LEDs (4-bits). The PWM circuit produces a 8-bit DC signal that allows the integrator to generate a sinusoidal wave, ultimately controlled by the frequency. The use of a 2 KHz frequency is used to provide good color variation in order to keep the steadiness of the brightness of the LEDs.

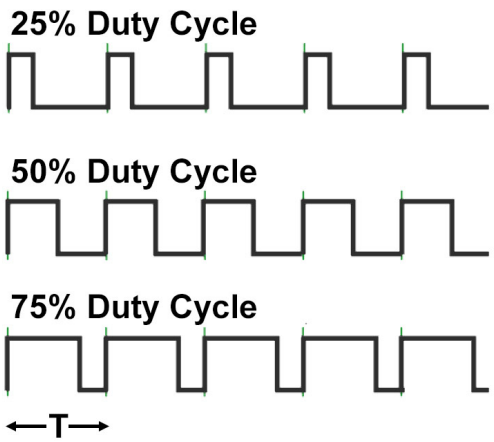

Figure 4: PWM Duty Cycle

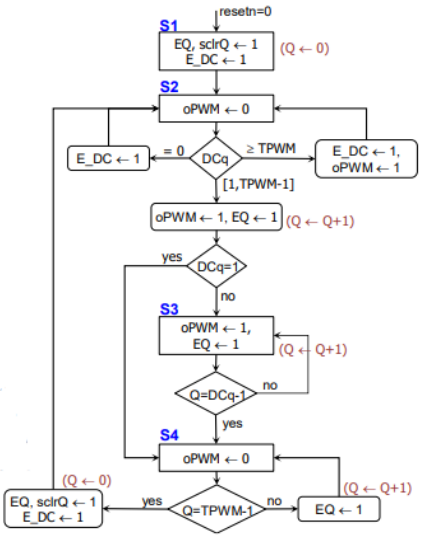

Figure 5: PWM FSM

The following FSM controls the output of PWM. TPWM is the period of PWM in units of Tclock and is a generic in the code. In the case of this project, a TPWM of 10 was used in order to comply with the encoder output of 4-bits. nDC is a function of the logarithm of TPWM which is an input component to the mypwm component. DCq is converted from TPWM to DC to get low or high signals. PWM is a signal that varies between 0-255 bits and has a duty cycle set between 0 and 100%, we can use PWM to vary the brightness of the LED, therefore changing the color, based on orientation, temperature, switch value, etc.

E. On-Board Switches

The input of switches fourteen and fifteen are port mapped to the input of the multiplexer. This multiplexer, based on the input of the switches, dictates which reading will be used for the control of the LED. Therefore, these two switches act as a mode selection. Switches zero to eleven are used for direct PWM control once the mode is selected. From this set of twelve switches they were broken into groups of four. Each group was then designated to the control of a single color of the RGB. Switches zero to three for red, four to seven for green and eight to eleven for blue. Switches 12 and 13 were used for debugging purposes. When “00” the value of the temperature sensor was displayed on the 16 on board LEDs. The next three combinations were used to display one of the accelerometer axes on the LEDs. This was used mainly to debug and test the data outputted by each of the sensors.

F. Encoders

For converting the output signals of the accelerometer, and temperature sensor into readable values for PWM_LEDS, the implementation of this encoder was necessary. The encoder itself was split into 16 different ranges in which the outputs of each of these two components would fall into. This was done to account for all the possible combinations of PWM since it has a 4-bit input. The encoder takes the input for one of the sensors then converts the logic vector to an integer. One more conversion is necessary since both sensors output signed data. This now converted data is mapped to the 16 ranges and with the use of if statements is mapped down to a 4 bit output. The encoder for the temperature sensor converts 16 bit data to 8 bit data for the output data of the temperature sensor. This number was found by using the onboard LEDs to determine the amount of bits needed when max temperature was found with the use of a hair dryer. The outdata will be mapped the following way, with the four most significant bit values mapped to red and the four least significant bits of this signal mapped to blue. Combining these signals and mapping to the select of pwmleds allows for the use of the RGB LED.

III. Experimental Setup

This project used an encoder, onboard switches, an accelerometer and temperature sensor to produce the output of the RGB LEDS using PWM and the Nexys 4 DDR board. Each component was evaluated to ensure functioning capacity as code was synthesized before implementation in order to prove functionality. If all were successful, bitstream generation commenced in order to see if the code functioned as planned. Switch 14 and 15 were used as select values. If in mode “00” the board would be off. If in mode “01” the board would be in switch mode, where switches 0 to 11 were used simply as input data for the PWM controller and were able to display colors based on switch value. If in mode “10” the board would be in accelerometer mode, where

changing the orientation of the FPGA board would alter the color of the RGB LED according to different X,Y, and Z values. If in mode “11”, the board would be in temperature sensing mode, where the onboard temp sensor could be used to increase or decrease the brightness of the RGB LED from “base” color of blue to red for higher temperatures. The use of a hair dryer and air duster canister was important to display the functionality of the temperature sensor when trying to display data.

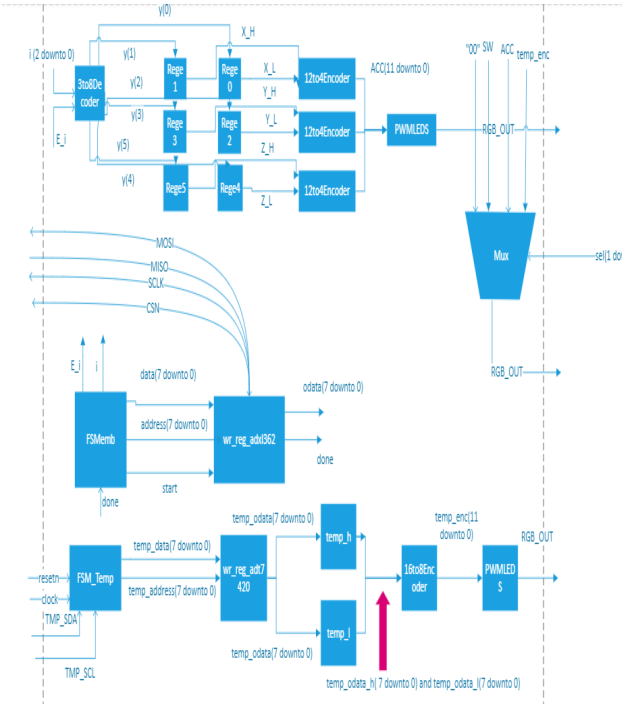

Figure 6: Block Diagram

The block diagram shown above displays how the circuit was implemented with the previous components.

IV. Results

The results from this project demonstrated application of the course material with the use of VHDL and the Nexys Artix A7 board. Being able to successfully display the LED color control with the use of the onboard switches, accelerometer and temperature sensor to adjust brightness and color of the onboard LEDs. Initial experimentation led to great improvements with the use of the test bench and on board display, as if something wasn’t working appropriately, functionality of the board implementation not working appropriately, this information was used to debug the code. Errors were constantly encountered that group members were able to solve with a meaningful consensus. The largest issue was coming up with a way to manipulate the components codependently (accelerometer and temperature sensor) as bit sizes kept being mismatched. This was fixed with the editing of the circuit, the use of two variations of encoders and addition of modes designated by the onboard switches.

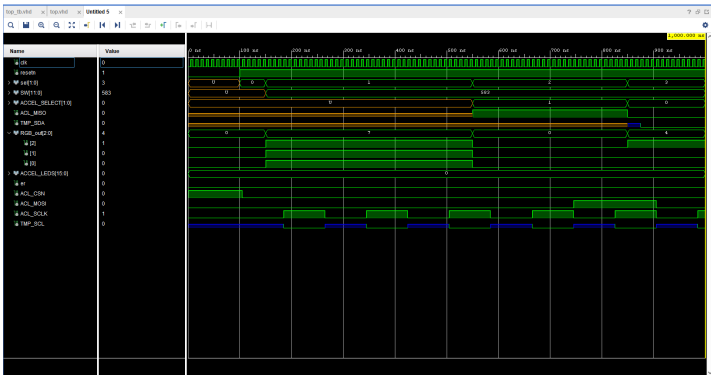

Figure 7: Results Simulation

This picture shows the results from simulating in Vivado. The stimulus consists of changing the modes via the switches, beginning in off mode where the RGB_out signal is zero for each of the three colors, which portrays the LED being off. Then, we switch the mode into controlling with the switches, and because the stimulus included one’s in each color range, the simulation portrays the RGB LED as all of the colors being on. Next, accelerometer mode where the stimulus is setting the signal for the Master In, Slave Out as high. Finally for the temperature sensor mode, we set SCL to a “high Z” state and we can see that the LED is showing that the LED is displaying a blue color.

The following picture displays the results of mode “01”, switch mode. Here switch 0 (dim red) and switch 8 (dim blue) are both on, combining and producing the color purple on the RGB LED.

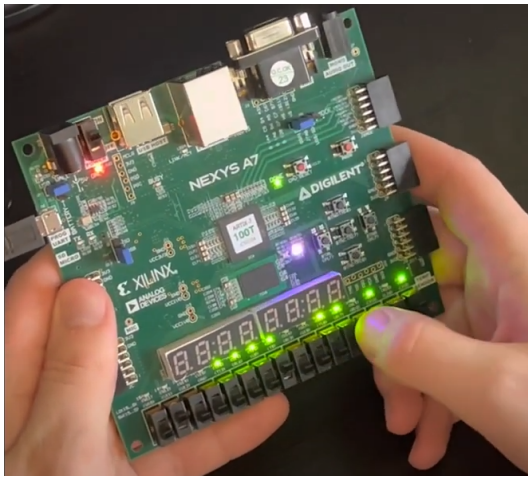

Figure 8: Color Combination of Two Inputs

The constant changing colors through use of the accelerometer was unable to be documented through pictures, however, worked as intended. The RGB LED would increase/decrease in red brightness if the board was moved in the X direction, change in green brightness if moved in the Y direction and change in blue brightness if moved in the Z direction. Manipulating multiple directions at once would cause a combination of colors as expected.

The results of the temperature sensor are seen below. The LED is red since a hair dryer was used to increase the temperature being received to the temperature sensor. This is the max LED output that the board will display in this mode.

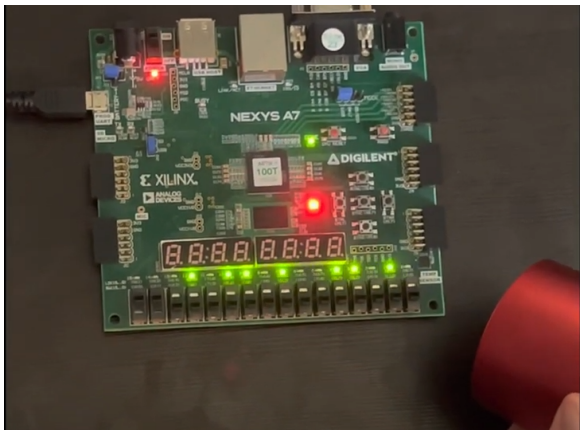

Figure 9: Maximum Output of Temperature Sensor

The following figure shows the results of bringing the board back to its base temperature reading through the use of an air duster canister. This base temperature is displayed as blue on the RGB LED.

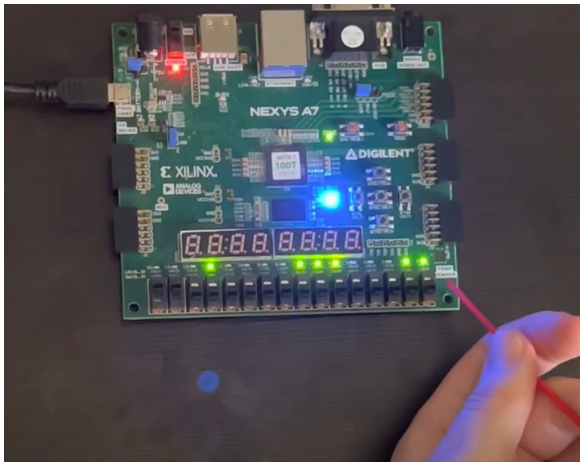

Figure 10: Base Output of Temperature Sensor

The implemented circuit was able to produce multiple color combinations through the use of PWM as well as operate as intended through the use of the programmed select switches.

V. Conclusion

In conclusion, this project was a great way of using multiple components learned about in class in an effective and efficient manner in order to properly use PWM to control the RGB LED of the FPGA board. The group learned much about communication and teamwork through encountering issues and coming up with solutions through meaningful discussion. The mode of the FPGA board was decided through the implementation of a select programmed to two switches. When in switch mode, the board operated appropriately and was able to display multiple color combinations according to the switch value. When in the accelerometer mode, the board was able to control the color and brightness of the RGB LED through the change of its orientation. The temperature sensor also functioned appropriately and was able to increase and decrease the brightness of the RGB LED according to the temperature input. Besides functionality, the group learned a large amount about PWM modulation and ways of manipulating RGB color. Improvements that could have been made would be displaying the according data on the seven segment display and implementing a select mode for the accelerometer to isolate the X,Y, and Z coordinates so no mixture of colors occurs when moving the board if intended. These improvements could have been implemented to give the user more benefit simply for user experience or if

they are trying to figure out how to implement or make something similar themselves. Overall, the application of manipulating the onboard RGB LED was successful through various group discussions, the help of our professor, and the information learned in the course throughout the semester.

REFRENCES

[1] Llamocca, Daniel. VHDL Coding for FPGAs, www.secs.oakland.edu/~llamocca/VHDLforFPGAs.h t ml

[2] Brown, Arthur. “Nexys A7 Reference Manual.” Nexys A7 Reference Manual - Digilent Reference, Digilent, reference.digilentinc.com/reference/programmablelog ic/nexys-a7/reference-manual