Light Intensity Measuring Device

Read from 7 Segment Displays

List of Authors (Christian Lauderback, Matthew Foltz, Bradley Jarbo)
Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
e-mails: christianlauder@oakland.edu, mfoltz@oakland.edu, bjarbo@oakland.edu

***Abstract*— The purpose of this experiment is to design and implement an externally interfaced system which involves a light intensity sensor that will be connected to the NEXYS 4 DDR board which will relay the information collected and display this data on a 7 segment display. This information will be displayed on a scale of 0 to 255. To complete our project we found that a finite state machine (FSM) was necessary for our system to function properly as a whole. After completing this component and connecting each component, we found that our system functioned properly.**

I. Introduction

The group is to implement a light sensor (PMOD ALS) to collect ambient light intensity in a room. This information will be stored in 8 bits which will be decoded and processed to output on the NEXYS 4 DDR 7 segment display. Topics learned in class that this project will include is use of shift registers to collect the data from the light intensity sensor. Use of decoders to process the information collected to the 7 segment display will also be essential for this project which was also something covered in class. One application our project could be useful for is to create a climate controlled environment for plants that displays ambient light intensity and allows to user to keep their plants alive in a light controlled atmosphere.

II. Methodology

Through collaborative discussion, the decision to design a light intensity sensor was reached by the team. Once this decision was made, a search for appropriate input sensors was started. The search culminated in the decision to use the Vishay Semiconductor's TEMT6000X01 ambient light sensor as integrated in the Digilent PMOD ALS. This sensor was chosen because of its integrated ADC and connection compatibility with the Nexys 4 Artix 7 FPGA board by the same manufacturer [1]. This connection compatibility allows for the removal of planning connections from our task list leaving the team free to only consider programming decisions and inputs already integrated into the Artix 7. The programming language used to construct components in the FPGA will be VHDL. The components needing to be constructed will solve the issues of communication, data manipulation, and data display. The block diagram shown in Figure 1 represents the system as a whole, and the design of each of the three major components are discussed in the following sections.

A. Communication

The PMOD ALS has an integrated Texas Instrument's ADC081S021 chip. This chip utilizes the Serial Peripheral Interface or SPI [2]. The SPI Protocol is a full duplex synchronous communication standard where a master device can either read and write information from a connected slave. The connection from the master to slave requires a Master Out /Slave In (MOSI), Master In /Slave Out (MISO), Channel Select, and Communication clock. These connections are made by the connector from the PMOD to the FPGA. The data transfer will occur based on the shared communication clock, and will be transferred one bit per communication cycle. The communication cycle must be slower than the master clock of the system, so a clock divider will be needed to reduce the master clock to 3.13MHz for the communication clock. The ADC has an eight bit resolution, so a total of 8 bits will need to be read by the FPGA through use of a shift register. This register will use counters as well as the communication clock to control the on and off of the data stream. Finally, the data read from the ADC will be stored in a synchronous load register. The key to this communication will be the timing of the shifts, and the channel select lines to ensure the correct information is captured.

B. Data Manipulation

Once stored, the data will need to be processed from a one byte unsigned integer binary number to a number in decimal. The team will design and implement an 8 to 12 binary to BCD decoder. This decoder will be based on the double dabble algorithm [source needed]. To implement

this algorithm the use of adders, and shift registers will be crucial. Also, the algorithm is well established, so once implemented properly the component will work consistently.

C. Data Display

The final component will consist of three BCD to seven segment decoders. The 12 bit BCD data will be split into packets of 4 bits, and converted to the proper LED placement by the seven segment decoder assigned to each of the three displays. Further, each 4 bit packet will be sent to a series of NOR and AND gates to ensure that only the needed seven segment displays are lit when in use. This will lower the total power usage of the machine.

Once all the individual components are programmed and tested in VHDL, a top file will be created to integrate all of the components into a single machine.

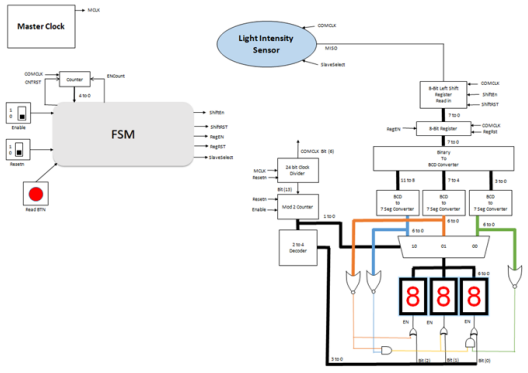
III. Experimental Setup

The software that is used is a test bench that will be used to check for possible values. The hardware that is used is an adjustable brightness LED that will be used to test the function of the 7 segment display. An arduino is a tool that is used in this project to check and make sure that the light sensitivity sensor is reading proper values. The specific configuration of the tools used is when the adjustable LED is connected we will run a timing diagram and look for any unexpected behavior. The expected results for this is that when the LED intensity changes, the seven segment display shows the correct number from 0 to 255, 1 being the lowest intensity level and 255 being the highest intensity level. We expect all segments to turn off when the LED is adjusted to 0.

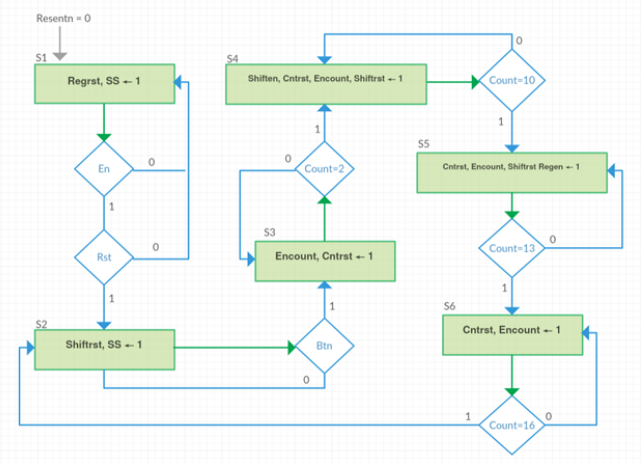
If the intensity level is in 100's or above then all 7 segments should be on, 99-10 then only 2 segments should be on, and if 10-1 then only 1 segment should be on at that time.

IV. Results

Our results indicate that our system works and is fully functional. Io accomplishing this, we designed the following system.



In order to properly read the information data from the sensor, we had to use the timing diagram from the PMOD ALS data sheet and 2 shift registers (one only used to shift bits and one to store these bits) and create a state diagram. This state diagram can be seen below.



Upon our first implementation of our system on the NEXYs board we noticed that our register was cutting off one bit coming from the PMOD ALS. We fixed this by adjusting the counter reading in the state machine so that it is able to capture all significant bits from the sensor and provide a proper output (0-255) on the seven segment today.

The concluding results indicate that our system is fully functional and all output results are as expected. A video of our functioning final project can be seen here (https://www.icloud.com/photos/#05yvi1Hp5V3QLvRncmeAEXAVA) .

Conclusions

The main takeaway point from this project would be choosing the correct components to make our project work. We also had to find the proper data sheet for the PMOD ALS and apply it to our project using the information given on the data sheet. Learning how to use data found on a data sheet and applying it to the project was something we all learned from this project. There are no further issues that remain to be solved, but we had a problem with the button working slowly and getting the register to hold data. We fixed the button issue by attaching it to a faster running clock signal. The shift register issue was solved when we found that one control signal was reversed, so we switched it to make it work properly. Improvements that can be made would be to change the 0-255 read out to actual lux..

References

[1] "Electrical Engineering Store, FPGA, Microcontrollers and Instrumentation." Digilent, store.digilentinc.com/.

[2] "ADC081S021 Single-Channel, 50-ksps to 200-ksps, 8-Bit A/D Converter" Digilent, http://www.ti.com/lit/ds/symlink/adc081s021.pdf.