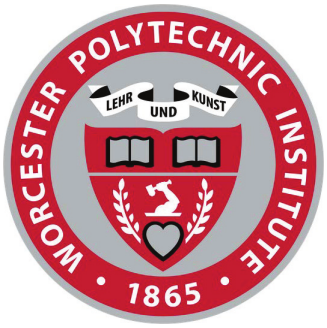
Lab 3: Light Sensor

Liam Snow



Professor Schaumont

ECE 3829 Advanced Digital System Design With FPGAs

WORCESTER POLYTECHNIC INSTITUTE

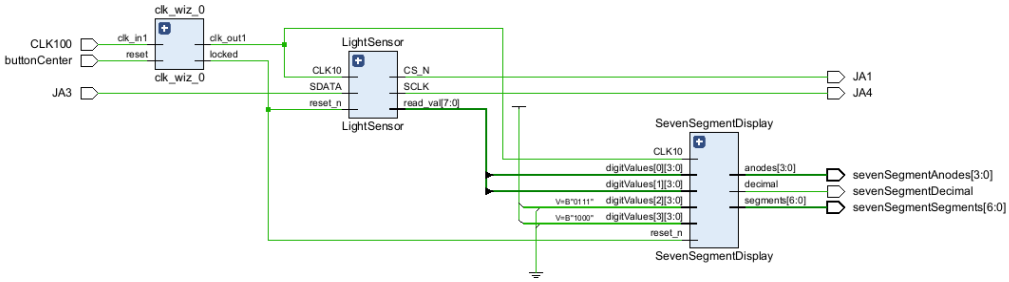
February 2024

Introduction

The goal of this lab was to create an SPI interface to interact with the `ADC081S021` light sensor over the Basys 3's PMOD extension port and display the reading on the seven segment display. I separated out the task into multiple modules to create a more organized approach. I created one module that would handle everything for the light sensor, including the SPI protocol and creating a slower clock for the interface. Then, I created a module to handle displaying numbers on the seven segment display. Finally, I created a module to convert the master $100MHz$ clock to a slower $10MHz$ clock.

Solution

Diagram



Clocking Wizard Module

The `clk_wiz_0` module is used to convert the external $100MHz$ clock to $10MHz$.

Light Sensor Module

The `LightSensor` module contains everything needed to interact with and read values from the `ADC081S021` light sensor, including an SPI interface and slow clock generation.

Seven Segment Display Module

The `SevenSegmentDisplay` module takes an input of 4, 4-bit hex values and displays them on the Basys 3's seven segment display. It does this but displaying one number at time and cycling through all digits hundreds of times per second to create a full 4-digit to the human eye.

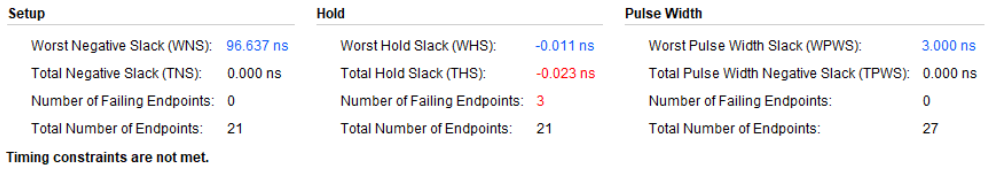
Implementation

Utilization Report

Name	Slice LUTs (20800)	Slice Registers (41600)	Bonded IOB (106)	BUFGCTRL (32)	MMCME2_ADV (5)
Top	65	65	17	3	1
Clocking Wizard	0	0	0	2	1
Light Sensor	43	47	0	0	0
Seven Segment Display	22	18	0	0	0

- Light Sensor:
  - The LUT is expected to be higher than Seven Segment Display because it is a much more complex module.
  - The register usage may seem initially high because there is only 41 bits of declared registers, however, it makes sense because some registers are required for storing `read_val` and the shifting operation into `read_temp_val`.
- Seven Segment Display:
  - The LUT usage is expected. Mostly because the segment decoding (4-bit to 7-bit) requires many LUTs, but also the anode decoding and slow clock creation.
  - The register usage is also expected because it is exactly equal to the amount of register bits, 2 from `currentDigit` and 16 from `slowClockCounter`.

Timing Report



Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	96.637 ns	Worst Hold Slack (WHS):	-0.011 ns	Worst Pulse Width Slack (WPWS):	3.000 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	-0.023 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	3	Number of Failing Endpoints:	0
Total Number of Endpoints:	21	Total Number of Endpoints:	21	Total Number of Endpoints:	27

Timing constraints are not met.

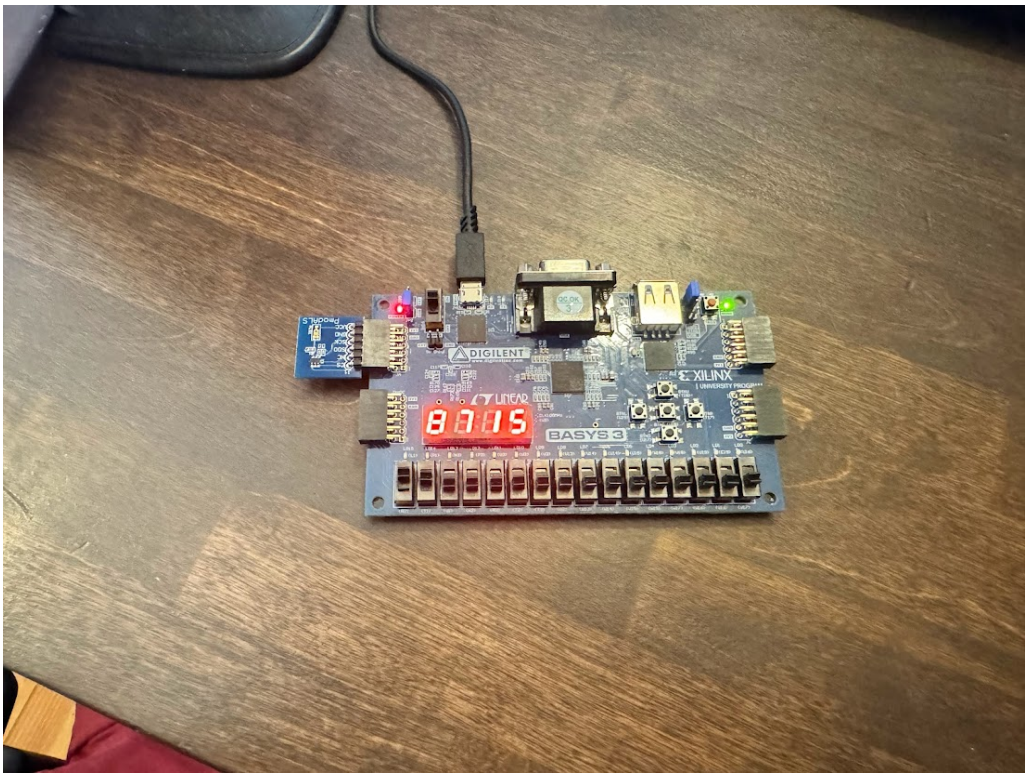
- The setup timing issues are caused by the Seven Segment Display's `slowClockCounter`. To solve this I would use a clock wizard (PLL or MCMM) to generate a slower clock next time.
- The hold timing issues are caused by the Light Sensor's `serialClockCounter`. To solve this I would also a clocking wizard module.

Conclusion

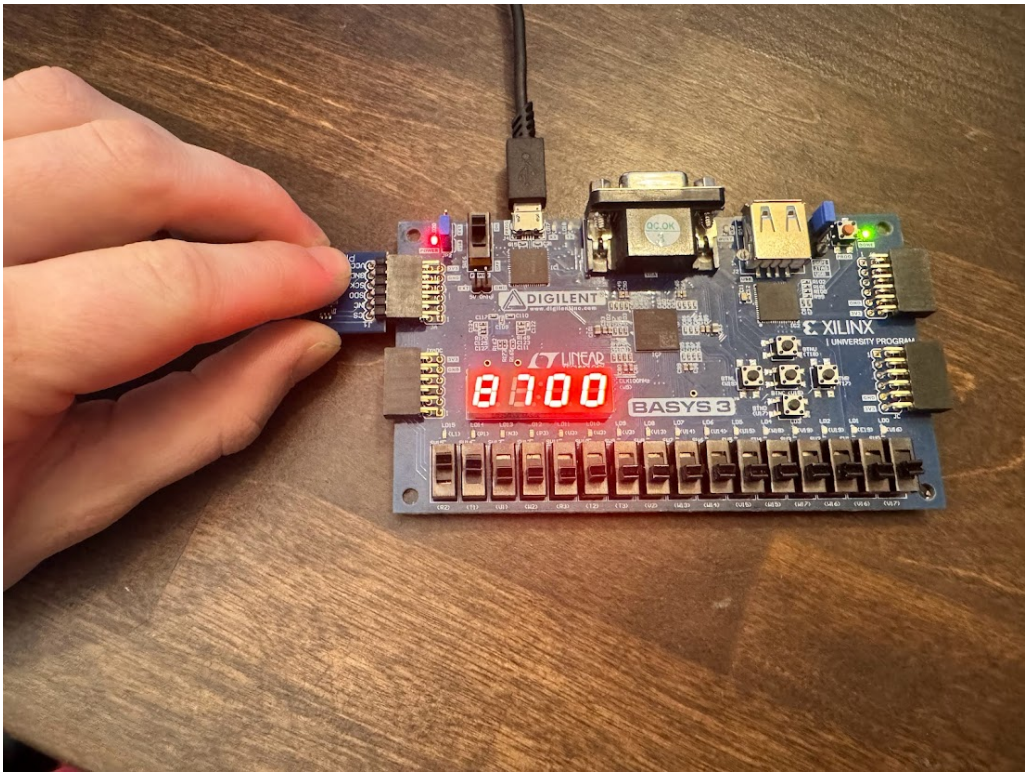
This was a great lab that challenged me a lot. It forced me create test benches to test my code, which was initially a hassle, but eventually helped me iterate much quicker and get more data from my testing. I struggled to find a good way to implement a state machine, which I think this project could have done better without. I encountered many small bugs that took awhile to debug, for example `sample_trig_counter` being too small and `read_seq_ptr` being set in different `always_ff` blocks causing it to work in simulation but not synthesis. Overall, I learned that carefully writing and reading Verilog is very important and small issues can break the entire module in unexpected ways.

Results

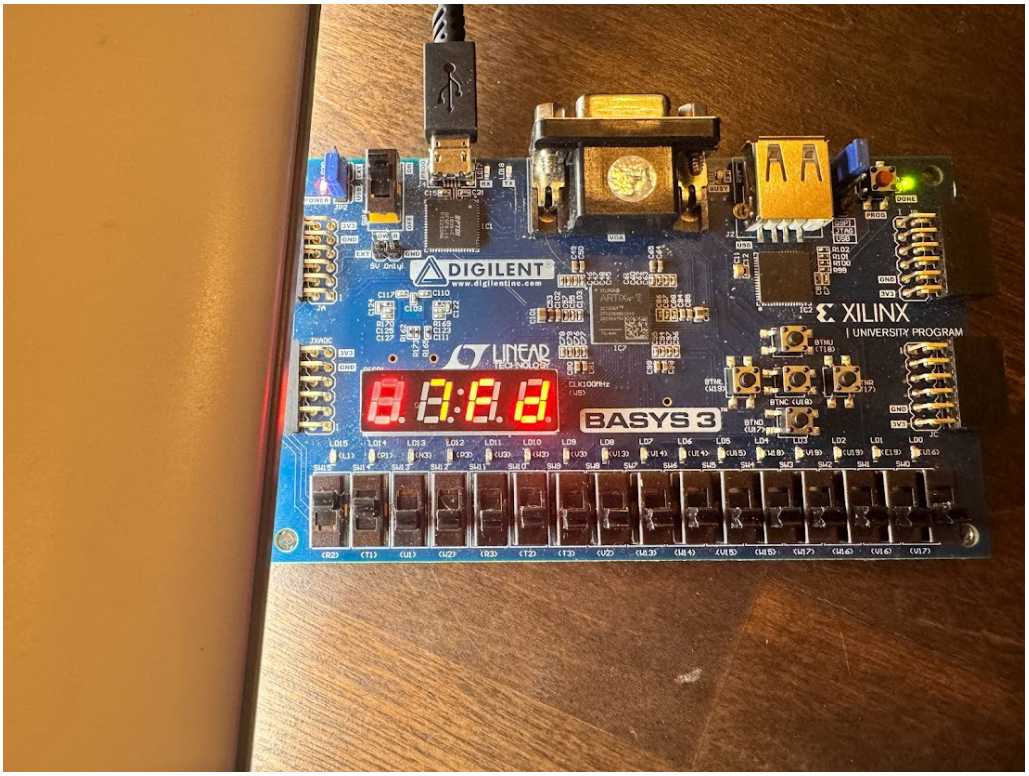
In Ambient Light



In No Light (Covered)



In Bright Light


DIGILENT
www.digilentinc.com
XILINX
UNIVERSITY PROGRAM
BASYS 3