ECE 3544: Digital Design I
Project 3 (Part A) – Design and Synthesis of a Parity-Checking System

Student Name: **Jonathan Lemarroy**

Honor Code Pledge: I have neither given nor received unauthorized assistance on this assignment.

Jonathan Lemarroy ____________

Grading: The design project will be graded on a 100 point basis, as shown below:

Manner of Presentation (30 points)

_________ Completed cover sheet included with report (5 points)

_________ Organization: Clear, concise presentation of content; Use of appropriate, well-organized sections (15 points)

_________ Mechanics: Spelling and grammar (10 points)

Technical Merit (70 points)

_________ General discussion: *Did you describe the objectives in your own words? Did you discuss your conclusions and the lessons you learned from the assignment?* (5 points)

_________ Design discussion: *Did you discuss the approach you took to designing any original modules? Did you discuss the approach you took to assembling the top-level module?* (10 points)

_________ Testing discussion: *What was your approach to formulating your test benches? How did you verify the correctness of the modules you designed?* (5 points)

_________ Supporting figures: *Waveforms showing correct operation of the top-level module.* (10 points)

_________ Supporting files: *Do the modules pass any tests applied by the grading staff? Modules whose declarations do not conform to the requirements of the project specification cannot be tested, and will receive no credit.* (15 points)

_________ Validation of the final design on the DE1-SOC board (25 points): ***Modules that exhibit the correct behavior but do not represent the correct implementation will receive no credit***.

_________ **Project Grade**

ECE 3544: Digital Design I
Project 3A Validation Sheet

GTA Validation Instructions:
Program the FPGA on the DE1-SoC Nano board with the student's implementation of the parity checking system. When the programming has successfully completed, perform the set of tests described in the table below. For each case, indicate whether or not the student' design demonstrates the behavior described.

Procedure and *Expected Result*	Correct Operation (**Yes** or **No**)
Choose a 7-bit value on SW[6:0] as the desired "transmit word." Record the value you applied here: SW[6:0] = ___________________________ Set the values of SW[9:7] = 000. *LED[6:0] should display the value on SW[6:0]. LED[9:7] should all be 0.*	
Change the 7-bit value on SW[6:0]. Record the value you applied here: SW[6:0] = ___________________________ Leave the values of SW[9:7] unchanged. *LED[6:0] should display the value on SW[6:0]. LED[9:7] should all be 0.*	
Change the value of SW[9:7] to a non-zero value. Record the value you applied here: SW[9:7] = ___________________________ Leave the value of SW[6:0] unchanged. *LED[6:0] should display the value on SW[6:0]. LED[9:7] should equal the value on SW[9:7].*	
Change the value of SW[9:7] to a different non-zero value. Record the value you applied here: SW[9:7] = ___________________________ Leave the value of SW[6:0] unchanged. *LED[6:0] should display the value on SW[6:0]. LED[9:7] should equal the value on SW[9:7].*	
Change the 7-bit value on SW[6:0]. Record the value you applied here: SW[6:0] = ___________________________ Leave the values of SW[9:7] unchanged. *LED[6:0] should display the value on SW[6:0]. LED[9:7] should all equal the value on SW[9:7].*	
Change the value of SW[9:7] to a different non-zero value. Record the value you applied here: SW[9:7] = ___________________________ Leave the value of SW[6:0] unchanged. *LED[6:0] should display the value on SW[6:0]. LED[9:7] should equal the value on SW[9:7].*	

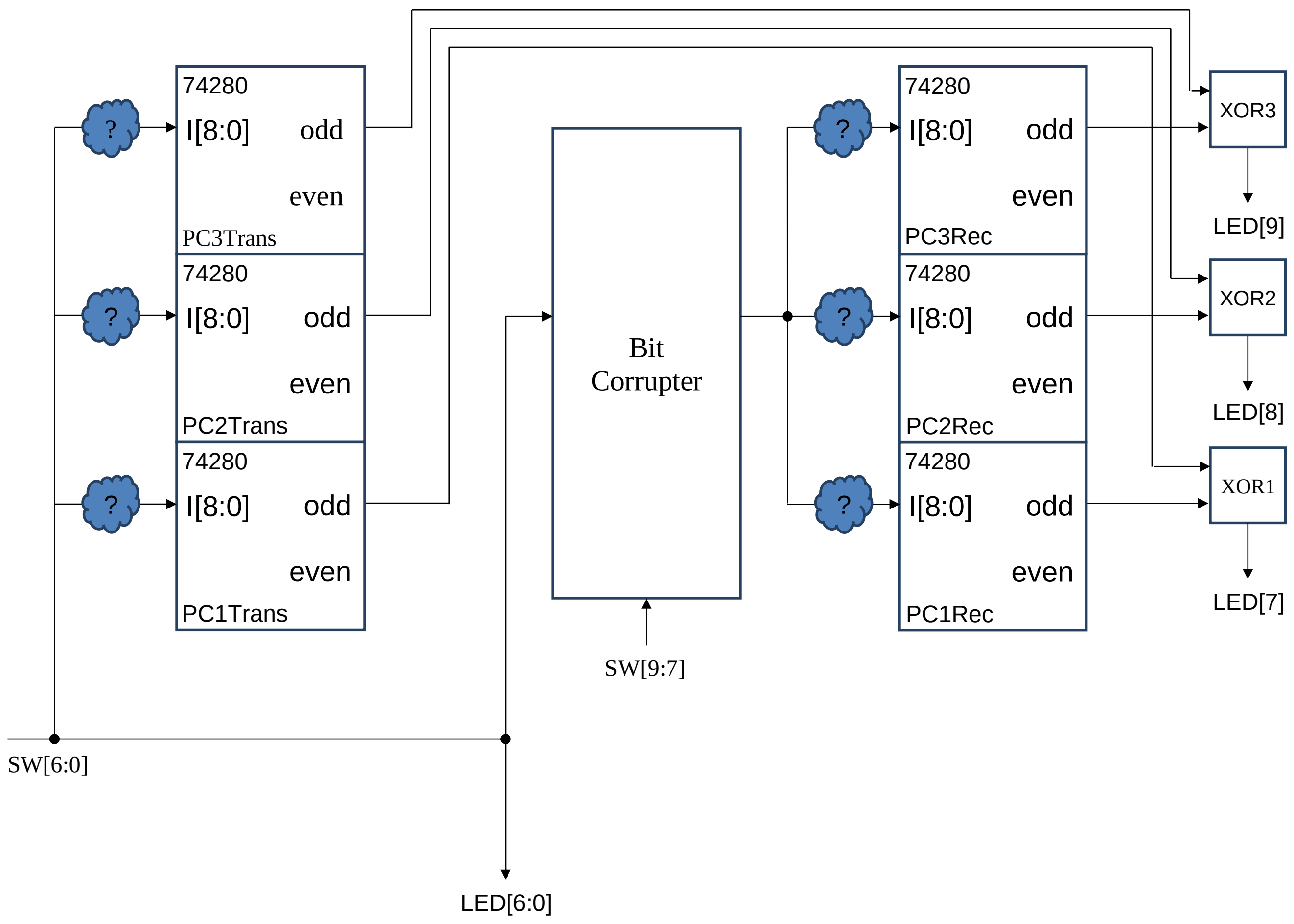
Purpose:

The objective of this assignment is to gain experience synthesizing Verilog modules using Quartus onto the DE1-SoC board, which will require learning how to setup the pin assignments correctly. Also, it is to become familiar with a bit error detection algorithm known as the Hamming code.

Project Description:

Design and synthesize a bit error detection Verilog module described by the circuit below.

Figure 1: Top-level block diagram of the Hamming code system

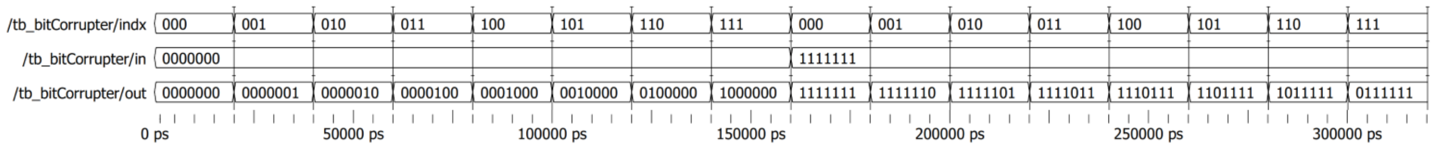


This circuit can be divided into three sections, a transmitter, a bit corrupter, and a receiver. The transmitter consists of 3 **74280** parity-bit checkers that are imported from Project 2. The bit-corrupter is to be a combinational circuit that flips the bit position determined by the switch signals SW[9:7] as an unsigned binary code. And lastly the receiver consists of the same design as the transmitter. The transmitter's and receiver's 3 parity bits are to be then compared to determine which bit was corrupted. This scrambled bit's location is to then be displayed as an unsigned binary code using LED[9:7]. The other LED signals LED[6:0] are to be connected directly to switches SW[6:0]. The **"?"** are for the designer to determine which bits should be connected to each of the parity checkers to produce the correct output in the LEDs.

Design:

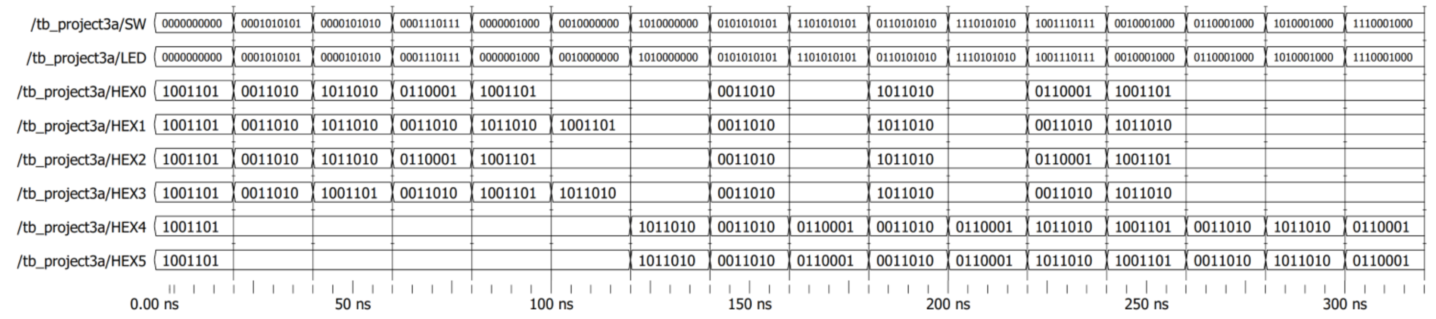
Starting with the bit corrupter which required using a conditional assignment for each bit, using the switches SW[9:7] as the condition. The results of the design were then tested in simulation, see **Waveform 1**. The test bench was designed to show that regardless if the data input, it would only invert the bit selected by the index.

Waveform 1: Bit Corruptor's Simulation



The next stage was determining which bits should be connected to each of the parity checkers. This was accomplished by making a comparison of the LED[9:7] combinations and then matching those to the SW[6:0] signals they would represent if the bit was incorrect. From this it can be deduced that the signals SW[6], SW[4], SW[2], and SW[0] are the only ones that would cause LED[7] to become a 1 if any of those bits were corrupted. Likewise it can also be determined the same way that LED[8] is influenced by signals SW[6], SW[5], SW[2], and SW[1]. And LED[9] is influenced by signals SW[6], SW[5], SW[4], and SW[3]. The rest of the design was simply wiring up the circuit as shown in **Figure 1**. This design was then tested using random signals for SW[6:0] and testing it against every combination of bit corrupter's signals SW[9:7], see **Waveform 2**.

Waveform 2: Hamming Code's Simulation



Analysis & Conclusion:

After simulating the modules and testing the results on the DE1-SoC board, it can be confirmed that everything should be in accord with the spec requirements, provided those specs were understood correctly.

Upon completion of this project, I can now assign pins to their corresponding module signals via referencing the board's manual. I can then compile and synthesize a module onto the board. And lastly I can run simulations via Quartus calls to run in ModelSim.