**ECE211-L LAB 3**

**Lab 3 Report: Advanced Comparators using 7400-Series Integrated Circuits**

*Efe CIVISOKEN*

*2/26/2025*

*This is a solo project, therefore no collaboration to mention.*

**Time Spent:** 1.5 hours.

**INTRODUCTION**

The objective of this lab was to implement various SystemVerilog modules for comparison operations (greater than and equal to) without the need for complex breadboarding, test their functionality using the Nexys A7 FPGA board, and finally extend them to perform more complex comparison operations, 6 in total. Furthermore, we combined the 1-bit comparator modules to design and implement a 2-bit comparator. The lab aimed to reinforce modular design principles and FPGA-based digital circuit implementation.

**DESIGN**  
To achieve the objectives, we implemented **two fundamental comparator modules** digitally in SystemVerilog:

*module equal\_to(input logic a,*

*input logic b,*

*output logic e);*

*or(e, ~a&~b, a&b);*

*endmodule*

*module greater\_than(input logic a,*

*input logic b,*

*output logic g);*

*and(g, a, ~b);*

*endmodule*

These modules were **instantiated** in the **top-level module (lab03\_top.sv)** in the following way:

module lab03\_top(

input logic [15:0] SW,

output logic [15:0] LED );

// Configure connections between switches and LEDs

assign LED[5:0] = SW[5:0]; //Connect switches SW0-SW5 to LED0-LED5

logic gt;

logic et;

greater\_than instance\_gt( .a(SW[1]), .b(SW[0]), .g(gt));

equal\_to instance\_et( .a(SW[1]), .b(SW[0]), .e(et));

Furthermore, they were used to **derive** **all six relational operations** according to the instructions in the lab demo (see APPENDIX section):

assign LED[15] = et;

assign LED[14] = gt;

assign LED[13] = et | gt;

assign LED[12] = ~gt & ~et;

assign LED[11] = et | ~gt;

assign LED[10] = ~et;

For the simple **2-bit comparator**, we **instantiated** greater\_than and equal\_to modules:

logic gt2, gt3, et0, et1;

greater\_than instance\_gt2n(.a(SW[5]), .b(SW[3]), .g(gt2));

greater\_than instance\_gt3m(.a(SW[4]), .b(SW[2]), .g(gt3));

equal\_to instance\_et0(.a(SW[4]), .b(SW[2]), .e(et0));

equal\_to instance\_et1(.a(SW[5]), .b(SW[3]), .e(et1));

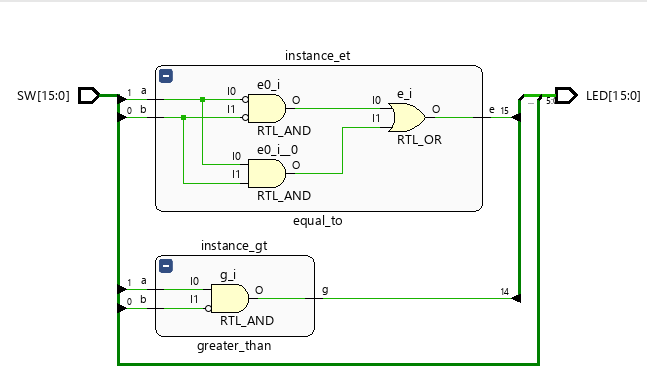
Furthermore, they were used in the following way to **perform the 2-bit comparison** operations:

assign LED[8] = et1&et0;

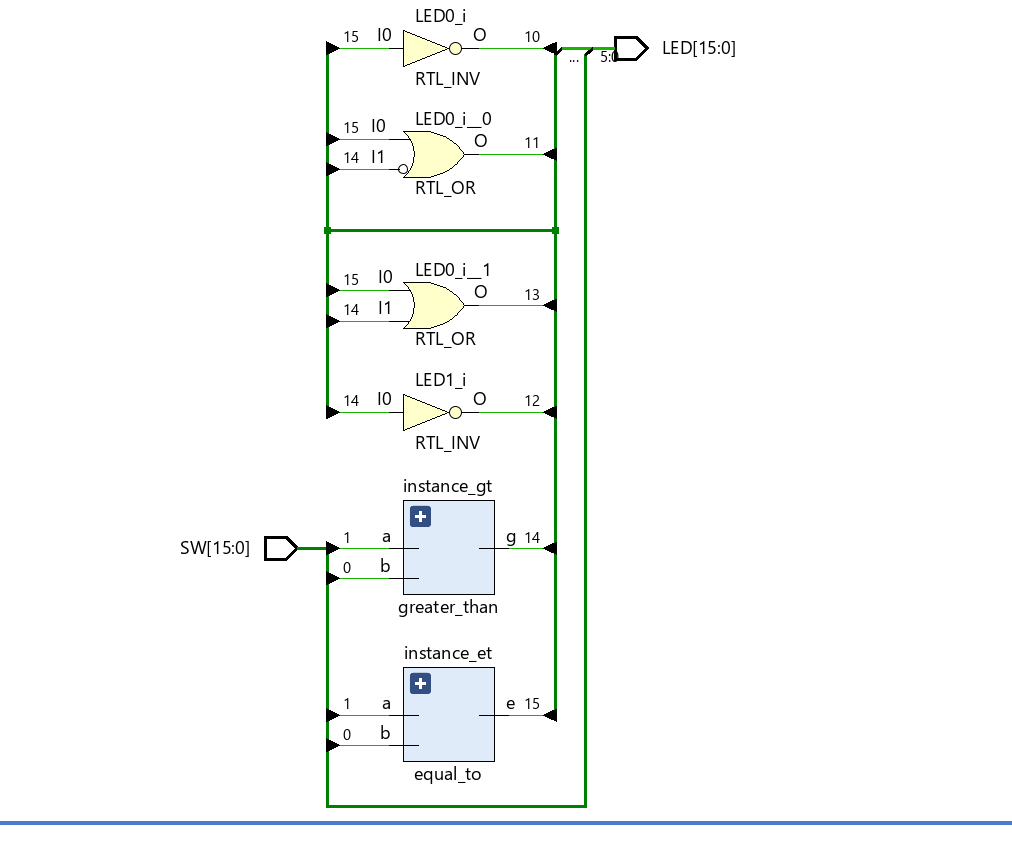
assign LED[7] = (~(et1&et0)) & (gt2|gt3) ;

**TESTING & RESULT**  
Our testing methodology included the following steps:

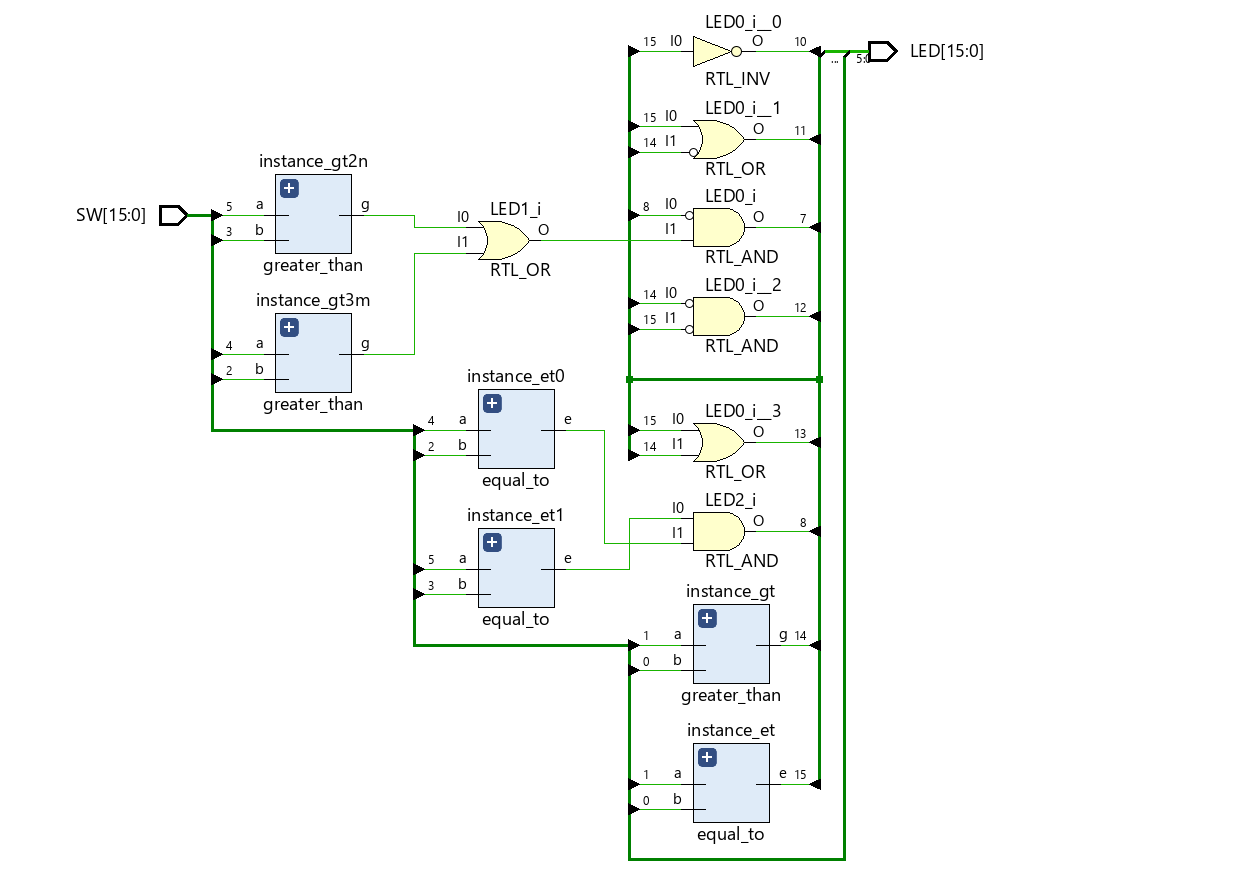
1. **RTL Analysis**: Initially verified the synthesized design in Vivado’s RTL Schematics viewer.

****

**Schematics 1.** Basic Implementation and Instantiation of Greater\_Than and Equal\_To Modules



**Schematics 2.** Implementation After the Addition of 6-Comparison Operations.



**Schematics 3.** Overall Final Implementation after Adding 2-Bit Comparison Operators.

1. **Bitstream Generation & FPGA Testing**:

* Used the Nexys A7 board to test all six comparison operations.
* Verified the 2-bit comparator with all combinations of inputs using the Nexys A7 board.

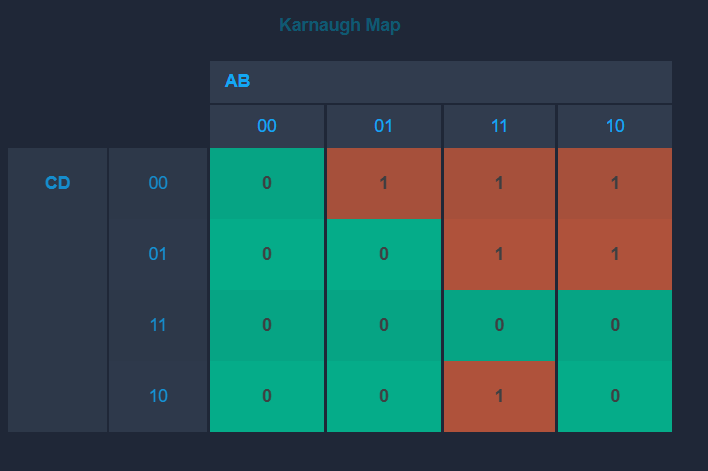
**Test-Case Demonstration:**

**6-Operation Comparison Test Cases**

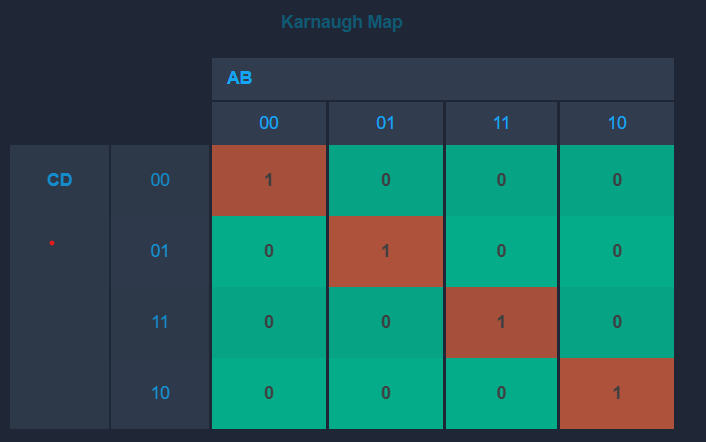
| **SW[1]** | **SW[0]** | **G (>)**  **LED[14]** | **E (==)**  **LED[15]** | **L (<)**  **LED[12]** | **GE (>=)**  **LED[13]** | **LE (<=)**  **LED[11]** | **NE (!=)**  **LED[10]** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | **0** | **0** | **1** | **0** | **1** | **1** | **0** |
| **0** | **1** | **0** | **0** | **1** | **0** | **1** | **1** |
| **1** | **0** | **1** | **0** | **0** | **1** | **0** | **1** |
| **1** | **1** | **0** | **1** | **0** | **1** | **1** | **0** |

**Table 1.** Truth Table for 6-Operation Comparison Test Cases, with LEDs and SWs on the board specified.

**2-Bit Comparator Test-Cases (With K-Map for Easier Demonstration)**



**Table 2**. K-Map of 2-Bit Greater-Than Function, where A and B stand for Switches 5 and 4, respectively, and C and D stand for Switches 3 and 2, respectively. The function’s value determines the output on the LED[7].



**Table 3**. K-Map of 2-Bit Equal-To Function, where A and B stand for Switches 5 and 4, respectively, and C and D stand for Switches 3 and 2, respectively. The function’s value determines the output on the LED[8].

For both functionalities, the visual results in the board aligned perfectly with the expected outputs in the truth table above, confirming the successful implementation of the design, which was also confirmed by the dear lab instructor Prof. Biernacki.

**DISCUSSION & CONCLUSION**

This lab reinforced my understanding of digital comparators and modular design, especially using SystemVerilog/Vivado syntax. Implementing 1-bit comparators first allowed for a structured approach to extending functionality to 2-bit comparisons. The FPGA board provided a hands-on platform to validate our designs. A key takeaway was the importance of **structured module instantiation** to ensure reusability in multi-bit designs.

**Challenges Faced and Their Resolution:**

* Some syntax errors in the last part of the logic implementation, especially after generating the bitstream cost me some time when I was testing the full functionality.

Other than that, I, thankfully, had no issue with the design, implementation, and testing processes.

In conclusion, this lab successfully demonstrated the design and implementation of modular comparators, and the functionality was confirmed through extensive FPGA testing.

**APPENDIX:**

* The Demo of Lab 3 can be found in Moodle.