**Lab 3: Applications of XOR and XNOR Gates**

Freddie Boadu

ECEN 328

Mansi Bhavsar

February 28,2019

**Introduction:** The overall objective/purpose of this lab was getting students comfortable with the idea of even and odd bit parity using XOR and XNOR gates to achieve this. The XOR and XNOR gate has many common applications such as: comparators and parity generations. The parity generation is what this lab was focused around. The student learned the two types of parity bits: even parity bit, and odd parity bit. Parity is dependent on the amount of 1’s within a binary number. If there are an even amount of 1’s we have even parity and vice versa. A parity generator also allows us to detect errors. Error detection occurs when there is odd parity occurring. The error is a result of the parity bit being incorrect as well as being transmitted incorrectly.

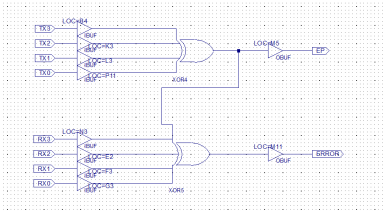
**Materials:**

• Xilinx ISE software, student or professional edition V14.7

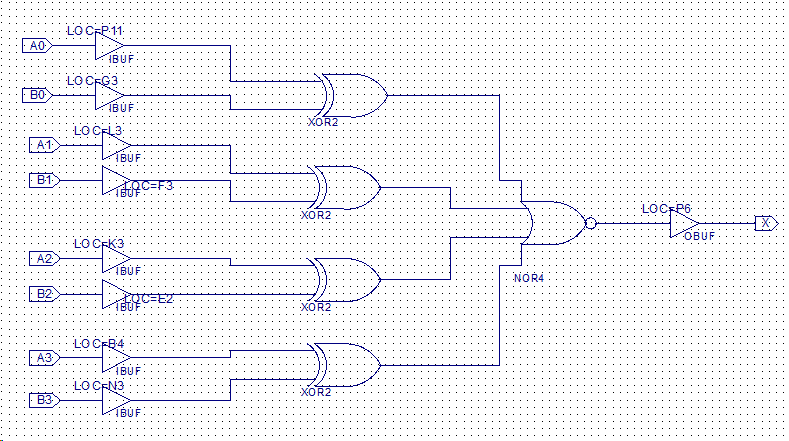
• PC with Pentium III or higher, 128+ MB RAM and 8+ GB hard drive

• Digilent Basys2 board with an XC3S100E device.

**Methods:**

The student is to open the design program, Xilinx ISE 14.7. In the project create a new project called GEN\_CHK and create a new schematic in that project with that same name. The student is first supposed to make a parity generator/checker. The generator would consist of XOR4, XOR5, IBUF, and OBUF gates. After the schematic is complete it should look like this: 

Switches 0-7 are used to represent TX0-TX3, and RX0-RX3. And LED0 and LED1 are represented by Even parity and error. Pins are added to all inputs and outputs of the schematic. Once the schematic is complete it should be uploaded to the board and tested.

The next schematic is a collection of XOR gates used in a comparator. The schematic should end up being similar to this: 

In the schematic There are IBUF,XOR2 and NOR4 gates used to make this schematic. A0-A3 and B0-B3 are represented by switches 0-7 respectively. Once the schematic is complete it should be uploaded to the board and tested.

**Data:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **TX0** | **TX1** | **TX2** | **TX3** | **RX0** | **RX1** | **RX2** | **RX3** | **EP** | **ERROR** |
| **SW0** | **SW1** | **SW2** | **SW3** | **SW4** | **SW5** | **SW6** | **SW7** | **LD0** | **LD1** |
| **0** | **1** | **1** | **0** | **1** | **0** | **0** | **1** | **0** | **0** |
| **1** | **0** | **1** | **0** | **1** | **0** | **1** | **0** | **0** | **0** |
| **0** | **0** | **1** | **1** | **1** | **1** | **0** | **0** | **0** | **0** |
| **0** | **1** | **0** | **0** | **0** | **1** | **0** | **1** | **1** | **1** |
| **1** | **1** | **0** | **1** | **1** | **1** | **0** | **1** | **0** | **1** |
| **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |
| **0** | **1** | **1** | **1** | **0** | **1** | **1** | **1** | **1** | **0** |
| **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0** | **0** | **1** |
| **1** | **0** | **1** | **1** | **0** | **0** | **1** | **1** | **1** | **1** |

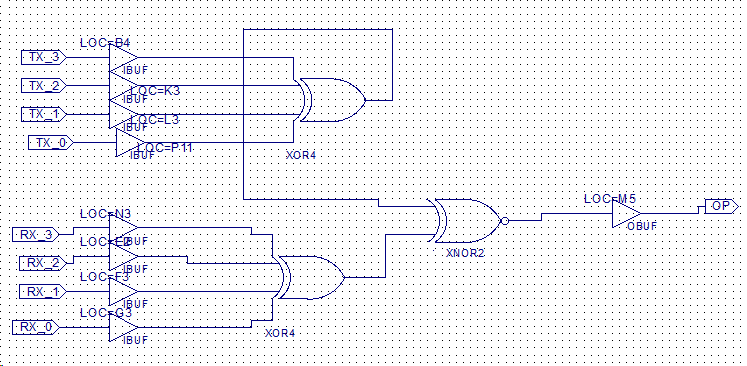
**Table 1: Checking for Parity**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **A0** | **A1** | **A2** | **A3** | **B0** | **B1** | **B2** | **B3** | **X** |
| **SW0** | **SW1** | **SW2** | **SW3** | **SW4** | **SW5** | **SW6** | **SW7** | **LD3** |
| **0** | **1** | **1** | **0** | **1** | **0** | **0** | **1** | **0** |
| **1** | **0** | **1** | **0** | **1** | **0** | **1** | **0** | **1** |
| **0** | **0** | **1** | **1** | **1** | **1** | **0** | **0** | **0** |
| **0** | **1** | **0** | **0** | **0** | **1** | **0** | **1** | **0** |
| **1** | **1** | **0** | **1** | **1** | **1** | **0** | **1** | **1** |
| **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **1** |
| **0** | **1** | **1** | **1** | **0** | **1** | **1** | **1** | **1** |
| **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0** | **0** |
| **1** | **0** | **1** | **1** | **0** | **0** | **1** | **1** | **0** |

**Table 2: Comparing Words A and B**

**Results and Discussions:**

After completing the demonstrations with the correct schematics, students were able to complete the truth tables in the lab document. The first table revealed two things: when RX and EP have an even number of ones, then no error was present other wise there was an error, also when TX had an even number of 1’s EP was 0 and when the numbers were odd TX would output a 1 on the ep. This showed a parity generator in full effect. Table 2 displayed a Comparator. The results are supposed to display high when the bits of A0-A3 equal the ones of the B0-B3 otherwise the answer was low. If the result is high or 1 this means that the LED turns on, if the results is low or 0 this means the LED does not turn on.

**Design challenge:** The design challenge required the student to construct a logic circuit that will generate odd parity for an 8-bit number. The schematic ends up displaying this:

This schematic end ups representing an 8-bit number with an odd parity. It consists of 2 XOR4 gates, and a XNOR2 gate.

**Conclusion:** The experiment above allows the hypothesis to be proven. The experiment shows the different ways XNOR and XOR gates can be used in combinational logic. It was discovered that the gates can be used to determine parity and make a comparator. The gates can also be used to make odd parity be displayed on the board as well. The software helped the students to understand how the different combinations of XOR and XNOR gates can come together to have multiple applications.