Title: XOR, XNOR Gates, Parity Generator & Parity Checker Circuits

Objectives

- To verify the input-output relationships of XOR and XNOR gates by constructing their truth tables.
- To implement and verify a 3-bit odd parity generator circuit.
- To implement and verify a 4-bit odd parity checker circuit for a 3-bit message.

Introduction

XOR (Exclusive OR) and XNOR (Exclusive NOR) gates are fundamental logic gates used in digital circuits for operations like binary addition and error detection. The XOR gate outputs logic 1 when an odd number of inputs are 1, while the XNOR gate outputs logic 1 when an even number of inputs are 1. Parity circuits, built using these gates, are used to detect errors in data transmission. A 3-bit odd parity generator adds a parity bit to ensure an odd number of 1s in the output, while a 4-bit even parity checker verifies the parity of a 3-bit message plus its parity bit.

This experiment involves implementing these circuits using the AT-700 portable laboratory with 7486 (XOR) and 7481 (XNOR) ICs. The setup uses a breadboard, data switches for inputs, and an LED light to observe outputs. The results are verified by comparing experimental truth tables with theoretical ones.

Procedures

XOR and XNOR Gates

1. **Set Up Components:** I installed the components for the XOR and XNOR circuits (using 7486 XOR and 7481 XNOR ICs) on the AT-700 breadboard. I ensured proper connections, with each IC's pin 14 connected to +5V and pin 7 to GND.

2. **Connect Inputs and Outputs:** I connected Data switches “0” and “1” to input points x and y of the XOR and XNOR circuits (Fig. 3-1), respectively. The LED Light’s “0” and “1” were connected to the XOR and XNOR outputs, respectively.

3. **Test the Circuit:** I toggled Data switches “0” and “1” between “0” and “1” positions and observed the LED light. A lit LED indicated logic 1, and a dark LED indicated logic 0.

4. **Record Results:** I recorded the observed outputs in the truth table for XOR and XNOR gates and compared them with the theoretical truth table.

3-Bit Odd Parity Generator

1. **Set Up Components:** I installed the components for the 3-bit odd parity generator circuit (Fig. 3-1(a)) on the AT-700 breadboard using 7486 XOR ICs. I ensured pin 14 was connected to +5V and pin 7 to GND.

2. **Connect Inputs and Outputs:** I connected Data switches to the three input points of the circuit and the LED Light to the parity output.

3. **Test the Circuit:** I toggled the Data switches between “0” and “1” positions and observed the LED light to confirm the output ensured an odd number of 1s.

4. **Record Results:** I recorded the results in a truth table and verified the parity generator’s functionality.

4-Bit Even Parity Checker

1. **Set Up Components:** I installed the components for the 4-bit odd parity checker circuit (Fig. 3-1(b)) on the AT-700 breadboard using 7486 XOR ICs. I ensured pin 14 was connected to +5V and pin 7 to GND.

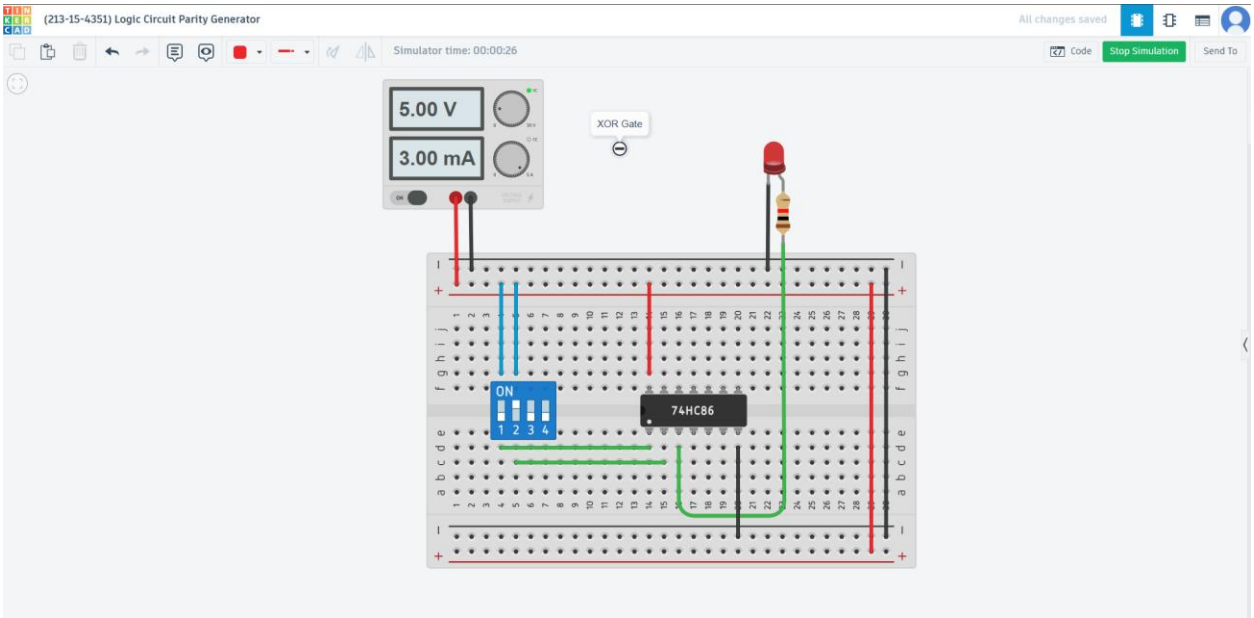
2. **Connect Inputs and Outputs:** I connected Data switches to the four input points (three message bits plus one parity bit) and the LED Light to the checker output.

3. **Test the Circuit:** I toggled the Data switches and observed the LED light to confirm the output was 1 for valid parity (odd number of 1s) and 0 for invalid parity.

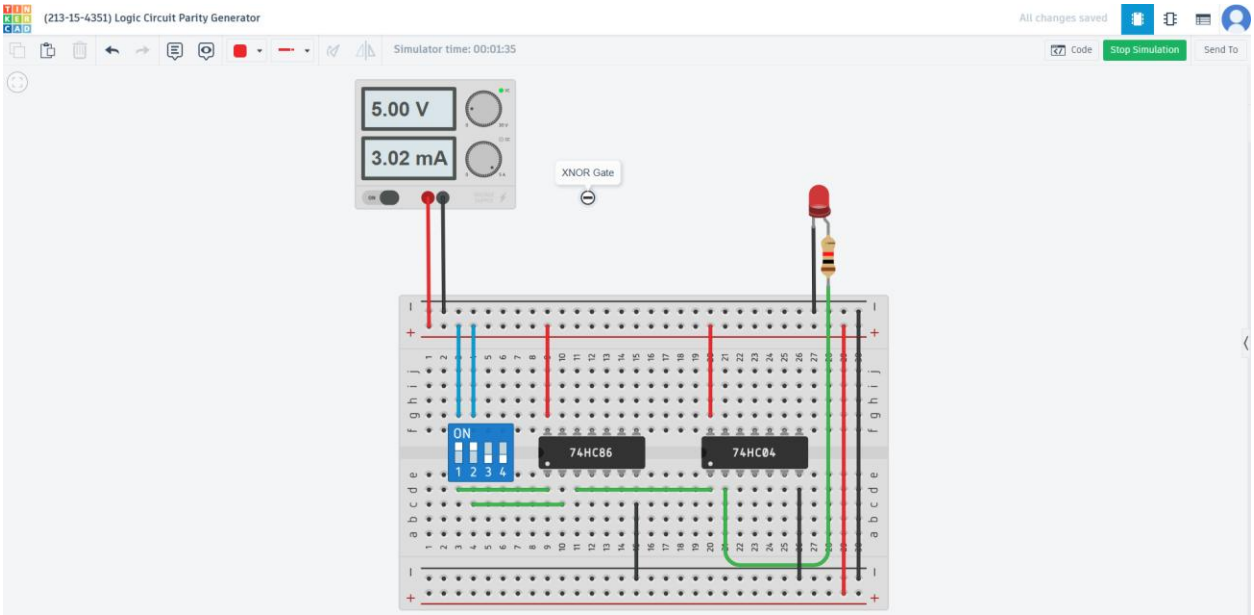
4. **Record Results:** I recorded the results in a truth table and verified the parity checker’s functionality.

Experiment Pictures

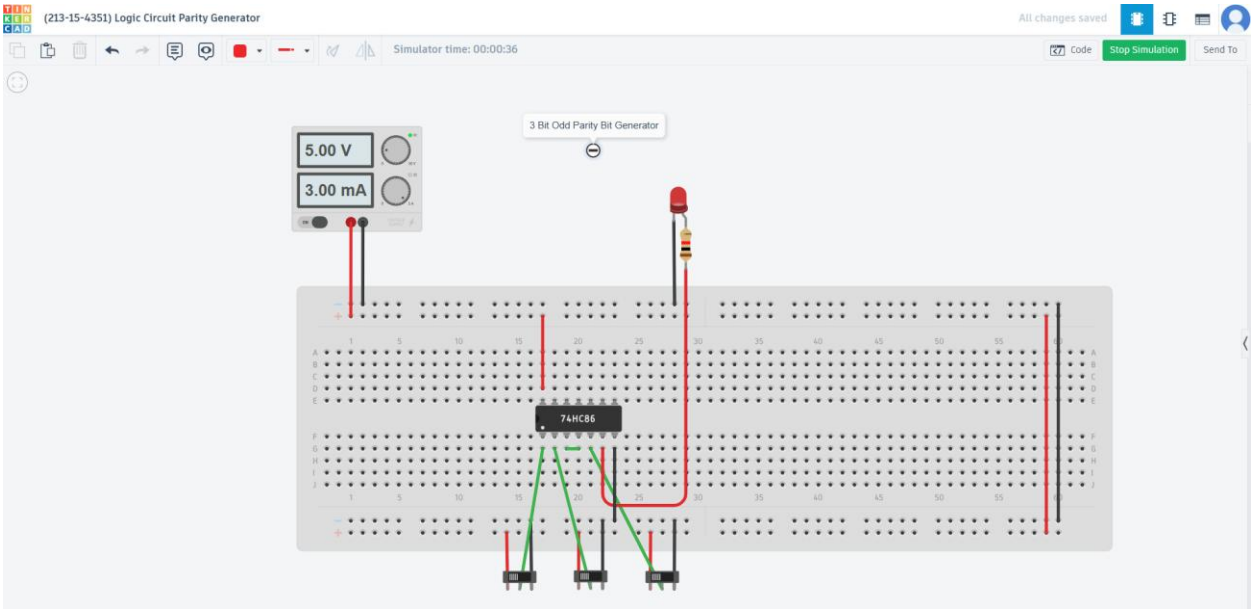
XOR Gate



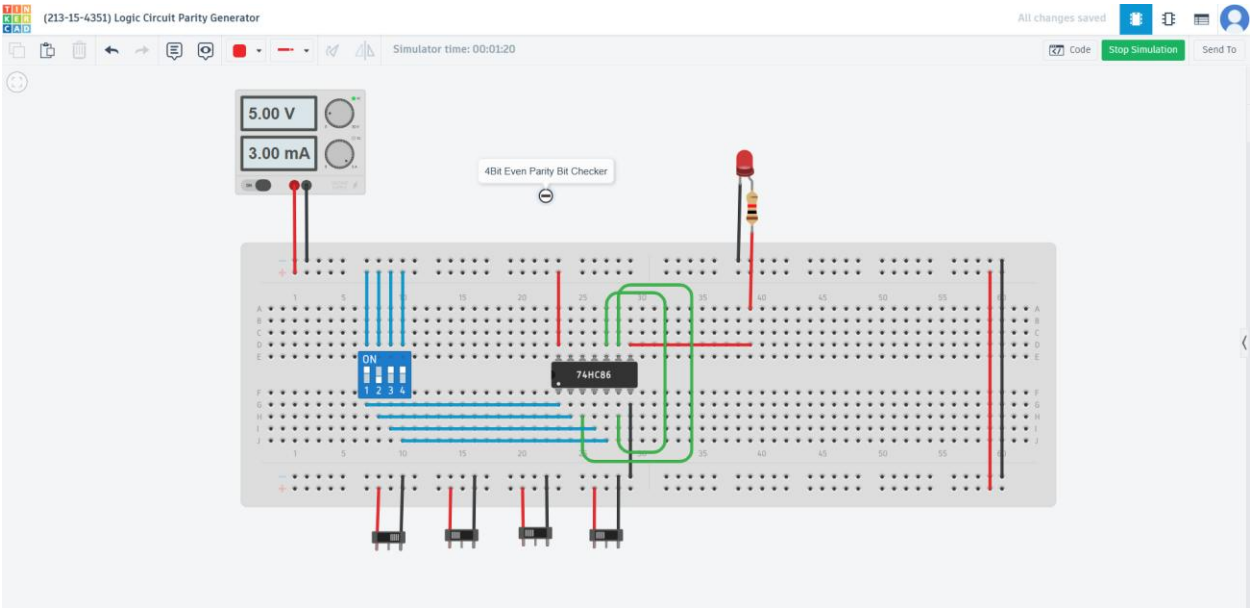
XNOR Gate:



3 Bit Parity Generator:



4 Bit Parity Checker:



Tinkercad Link

https://www.tinkercad.com/things/2O573KRjXKZ-213-15-4351-logic-circuit-parity-generator?sharecode=wXfDcLeCjy2JbUFPxVRjCEJa6g86P-xBjqsHH_LnyFA

Experimental Results

I recorded the results based on the LED outputs observed during the experiment.

XOR and XNOR Truth Table

x	y	XOR (x ⊕ y)	XNOR (¬(x ⊕ y))
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

3-Bit Odd Parity Generator Truth Table

A	B	C	Parity (P)
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

4-Bit Even Parity Checker Truth Table

A	B	C	P	Checker Output
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0

0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

The LED states (lit for logic 1, dark for logic 0) matched the theoretical truth tables, confirming that I correctly implemented the XOR, XNOR, parity generator, and parity checker circuits.

Conclusion

I successfully implemented and verified the XOR and XNOR gates, as well as the 3-bit odd parity generator and 4-bit odd parity checker circuits. The XOR and XNOR gates functioned as expected, with outputs matching their theoretical truth tables. The parity generator correctly produced a parity bit to ensure an odd number of 1s, and the parity checker accurately identified valid and invalid parity cases. The Tinkercad simulation helped me confirm the circuit designs before physical setup. This experiment enhanced my understanding of XOR and XNOR gate operations and their application in error detection through parity circuits.