EXPERIMENT #6: XOR IMPLEMENTATION AND PARITY CODE

OBJECTIVES:

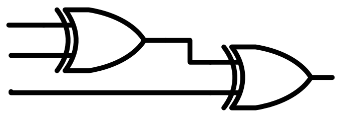
- Design and implement logic circuits using only XOR gates
- Design and implement a simple Parity Code Error Detection System

Equipment and ICs:

- Mini-Lab ML-2001 lab station
- 1 - IC 7493 4-bit Ripple Counter
- 3 - IC 7486 Quadruple 2-input XOR gates
- IC 7404 Hex Inverter gates
- 5 PCB mount SPDT switches

Introduction:

The XOR function for three variables converted to a Boolean expression is shown below:



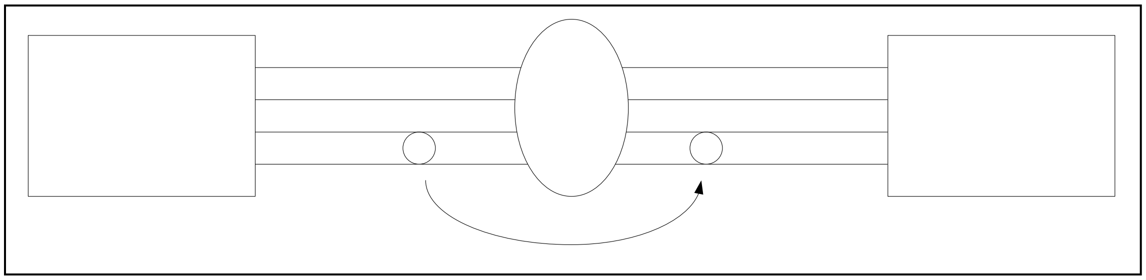
$\oplus$ $\oplus$

$\oplus$ $\oplus$

The Boolean expression and the truth table clearly indicates that the three-variable XOR function is equal to 1 if and only if one variable is equal to 1 or if all three variables are equal to 1. In other words, the XOR function is equal to 1 if an odd number of variables are equal to 1. Hence, the multiple-variable XOR function is defined as an ***odd function***.

Parity Generation and Checking:

A parity bit is used for the purpose of detecting errors during transmission of binary data. A parity bit is an extra bit included with a binary message to make the number of 1's either odd or even. The message, including the parity bit, is transmitted and then checked at the receiving end for errors. An error is detected if the checked parity at the receiver does not correspond with the one transmitted. The circuit that generates the parity bit in the transmitter is called a parity generator. The circuit that checks the parity in the receiver is called a parity checker. Both parity generator and parity checker circuits can be implemented using XOR gates.



Transmitter

Part 1: XOR as an Odd function

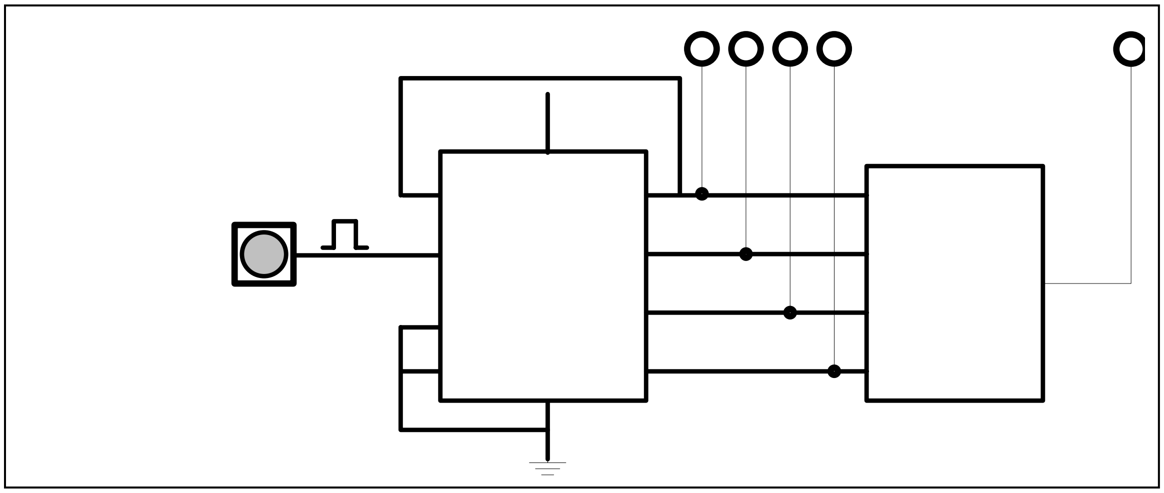
A large room has four doors and a switch near each door controls a light in the room. The light in the room can be turned ON or OFF by changing the state of any of the switches. The light will be ON if exactly one or three switches are closed and it will be OFF if two or four or no switches are closed. If the light is OFF when two switches are closed, then closing the third switch turns the light ON. Let the switches be denoted by A, B, C, and D and the light by L. Assume: Light ON = 1, Light OFF = 0, Switch open = 0, Switch closed = 1.

Pre-lab Work: (All the Pre-lab work must be shown in the Pre-lab report)

1. Obtain the truth table for function L.
2. Obtain the Boolean expression for function L from the truth table.
3. Draw the logic diagram for function L using AND, OR, NOT gates and compute the cost of the circuit in terms of literals, terms, and gates.
4. Comment on the distribution of 1's in the k-map for function L.
5. Show that function L can be implemented using only XOR gates.
6. Draw and simulate your circuit in LogicWorks. Include your LogicWorks drawing in the pre-lab report.

Lab Work: (All the Lab work must be shown in the Lab report)

1. Implement the logic diagram of L on the proto-board as shown below.
   a. Connect the 4 outputs of IC 7493 to inputs A, B, C, and D.
   b. Connect output F to one LED or indicator lamp.



2. Apply all 16 combinations (0000-1111) of 4 inputs to your circuit through IC 7493 by pushing the Pulser-button as many times. You can observe the input sequence on the 4 indicator lamps connected with the outputs of IC 7493.
3. Observe the output L for all combinations of inputs (switches). Record your observations in a truth table.
4. Compare the truth table above with the truth table obtained in Step 1 of Pre-Lab work and verify the operation of the circuit.

5. Compute the cost of the circuit in terms of gates and ICs.

OBSERVATIONS:

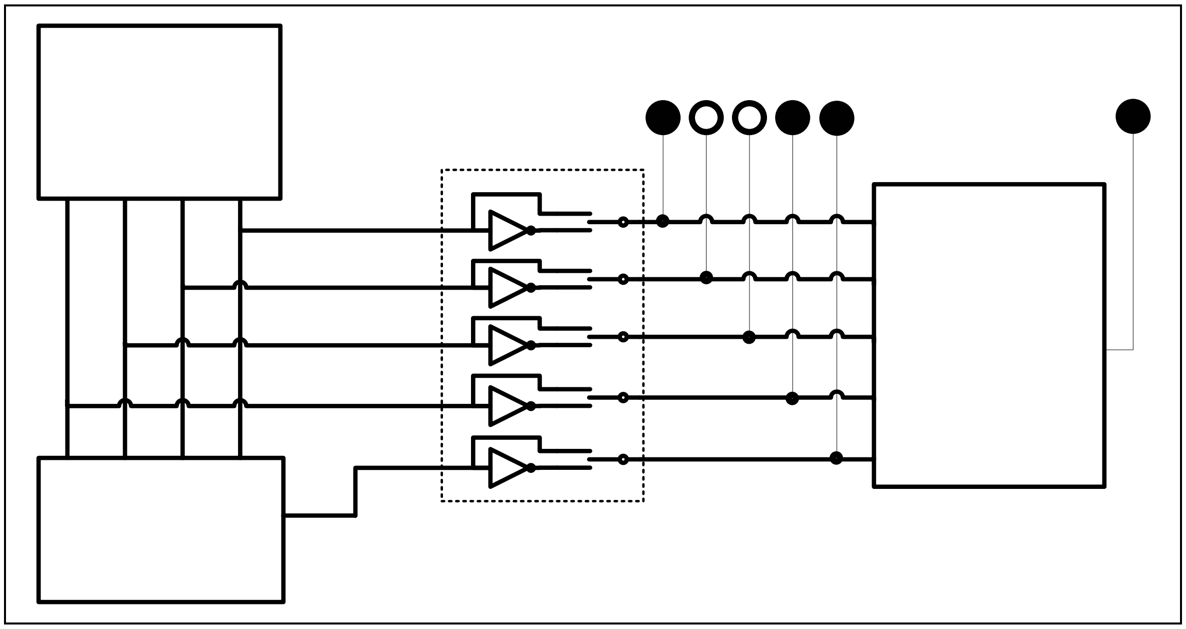
A	B	C	D	L
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

Table: Circuit Cost

Function	Number of			
	Literals	Terms	Gates	ICs
L using basic gates				
L using XOR gates				

Part 2: Parity Bit Error Detection Circuit

Design and implement a circuit that detects errors during transmission of a 4-bit message from point X1. An even parity bit is included with the message before transmission. The 5-bit data including the parity bit is received at point X2 that can detect an odd number of errors in the data received. A block diagram of such a circuit is shown below:



Circuit Description:

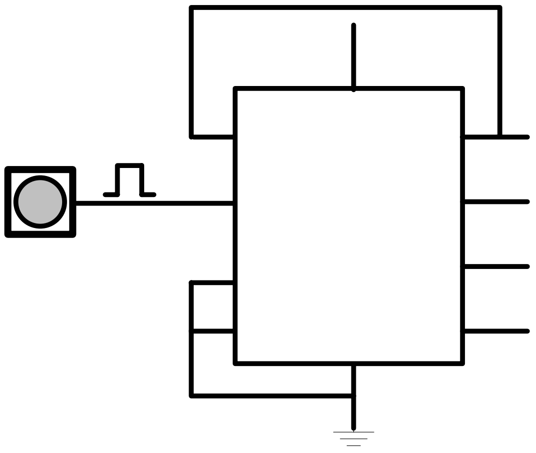
- IC 7493 4-bit Ripple Counter must be connected to count in binary. This IC is used to generate a 4-bit message.
- Inverters are connected (with a bypass switch) in the data transmission path to manually introduce bit errors due to channel noise. This is the simplest way to introduce errors in the data in the transmission path. To let the signal pass with out error flip the switch up, otherwise flip the switch down.
- The **Parity Generator** block at the Transmitter must generate an even parity bit for each 4-bit message received from IC 7493. The 5-bit data together with the parity bit is sent such that it always has an even number if 1's. The data goes through the transmission channel where it may get corrupted due to noise, and as a result a 1 may become 0 and vice versa.
- The **Parity Checker** block at the Receiver generates an error signal if there are errors in the 5-bit data received. If the number of 1's in the data received is odd, then the error P_C = 1, otherwise it is 0.

IC
7493

Q_A Q_B Q_C Q_D

Pre-lab Work:

1. Obtain the truth table for 4-bit Parity Generator function P_g.
2. Implement function P_g using minimum number of XOR gates. Draw the logic diagram.
3. Obtain truth table for 5-bit Parity Checker function P_C.
4. Implement function P_C using minimum number of XOR gates. Draw the logic diagram.
5. Redraw Figure 3 shown above by replacing the Parity Generator and Parity Checker blocks with their respective logic diagrams.
    - Use an SPDT switch with an inverter in the transmission channel.
    - Replace the IC 7493 block with the following figure:



6. Draw and simulate your circuit in LogicWorks. Include your LogicWorks drawing in the pre-lab report.

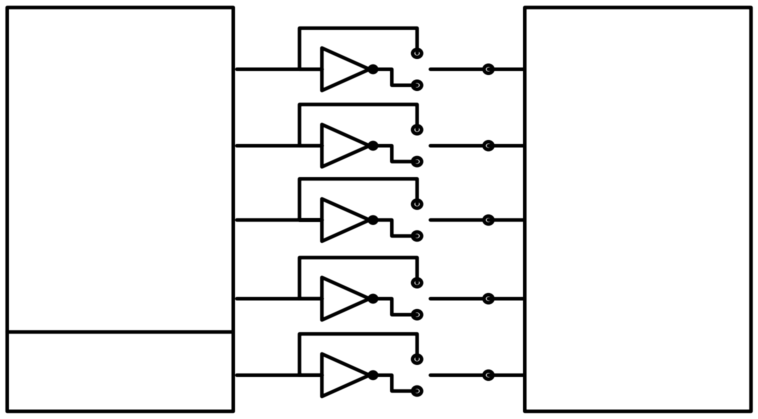
Lab Work: (Make sure to switch off power while making/breaking any connection)

Parity Generation:

1. Connect the 4 outputs of IC 7493 to the inputs of Parity Generator circuit (Remember that output Q_A is the least significant bit of the counter).
2. Connect the 4 outputs of IC 7493 to four indicators lamps to observe the input sequence. Connect the output of Parity Generator circuit also to one LED or indicator lamp.
3. Apply all 16 combinations (0000-1111) of 4 inputs to your circuit through IC 7493 by pushing the Pulser-button as many times. You can observe the input sequence on the 4 indicator lamps connected with the outputs of IC 7493.
4. Observe the output P_g for all combinations of inputs. Record your observations in a truth table.
5. Compare the truth table above with the truth table obtained in Step 1 of Pre-Lab work and verify the operation of the circuit.

Channel Simulation:

6. Now connect the 4 outputs of IC 7493 to four inverters and the output of the Parity Generator circuit to another inverter in the order as shown in Figure 5.
7. Connect 5 SPDT switches on the proto-board before the Parity Checker circuit as shown in Figure 5 below.
8. Connect each output of IC 7493 and the Parity Generator circuit to the top pin of one SPDT switch in the order as shown in Figure 5.
9. Connect the output of each inverter to the bottom pin of one SPDT switch respectively.
10. Connect the middle pin of each SPDT switch to the input of Parity Checker circuit in the order as shown in Figure 5.
11. Connect the middle pin of each SPDT switch to one lamp indicator to observe the data received by the Parity Checker circuit.



12. Flip the SPDT switch **UP** to let the signal pass through to the Parity Checker with out error. Flip the switch **DOWN** to invert the signal going to the Parity Checker circuit.

Parity Checking:

13. Push the pulser-button connected to IC 7493 to generate the 4-bit message and parity bit.
14. Verify that when the 5 SPDT switches are in the 'UP" position, the message is received at the Parity Checker with out error. Observe the indicator lamps.
15. Verify that when an odd (1, 3, 5) number of switches are in the "DOWN" position the Parity Checker generates an error. Observe the indicator lamp connected to the output of the Parity Checker.
16. Verify that when an even (2, 4) number of switches are in the "DOWN" position the Parity Checker does not generate an error. Observe the indicator lamp connected to the output of the Parity Checker.
17. Observe the output P_C for all combinations of inputs. Record your observations in a truth table.
18. Compare the truth table above with the truth table obtained in Step 3 of Pre-Lab work and verify the operation of the circuit.