CHAPTER 3
LOGIC GATES AND BOOLEAN ALGEBRA

3.0 INTRODUCTION

This chapter explores Boolean Algebra and the logic gates used to implement Boolean equations. Boolean Algebra is an area of mathematics involving operations on two-state (true-false) variables. This type of algebra was first formulated by the English Mathematician George Boole in 1854.

Boolean Algebra is based on the assumption that any proposition can be proven with correct answers to a specific number of true-false questions. Further, Boolean algebra provides a means whereby true-false logic can be handled in the form of Algebraic equations with the questions as independent variables and the conclusion expressed as a dependent variable (recall that in the equation y = A+B that A and B are independent variables and y is a dependent variable). This chapter will introduce you to the use of Boolean algebra and the use of electronic logic gates (circuits) to implement Boolean equations.

3.1 OBJECTIVES

Upon completion of this chapter you should be able to:

- Explain the basic operations of Boolean Algebra.
- Write Boolean equations.
- Use logic circuits to implement Boolean equations.

3.2 DISCUSSION

Boolean algebra is the branch of mathematics which studies operations on two-state variables. For the purposes of this book, an algebra is a system of mathematics where the operations of addition and multiplication can be performed with the results of the operation remaining within the system.

In Boolean algebra, addition and multiplication are the only binary (two-variable) operations which are defined. These two operations also may be performed on more than two independent variables. The only other operation in Boolean algebra is the unary (one-variable) complement function. These three operations are the only operations allowed in Boolean algebra. Mastery of these operations will be critical for understanding modern digital electronic circuitry.

3.2.0 Boolean Variables

Boolean variables are also known as logic state variables. Variables of this type can be in one of two possible states. The states are known as true and false. These true-false variables can be implemented with electronic devices as was illustrated in Chapter one.

Such logic devices are often described as having a 1 or 0, HI or LO, On or Off, True or False input or output. These expressions are convenient ways of noting the state of a particular Boolean variable. Keep in mind that the numbers, 1 and 0, refer to logic states and <u>not</u> binary integers.

Throughout the rest of this text, the terms HI, 1, and On will be used to indicate that a logical variable is True. Likewise the terms LO, 0, and Off, will indicate that the state of a logical variable is False. All exceptions to this convention will be noted.

Again, note that Boolean variables are two-state variables. Having two states allows these variables to be easily represented by electronic two-state switching circuits. Boolean variables are the basis of all modern digital electronic systems.

3.2.1 Truth Tables

Truth tables are useful in describing relationships of Boolean variables. A truth table lists all dependent and independent variables and all possible combinations of their states. The states are listed in mnemonic form.

The independent variables are listed at the top of the truth table to the left. At the top right of the truth table is the dependent variable. Columns of the truth table show all possible states of the associated Boolean variable. Figure 3-1 shows several examples of implementing the truth table for the logical OR function.

$f = A + B$

FIGURE 3-1. Truth Tables.

A	B	f
0	0	0
0	1	1
1	0	1
1	1	1

A	B	f
F	F	F
F	T	T
T	F	T
T	T	T

A	B	f
L	L	L
L	H	H
H	L	H
H	H	H

In Figure 3-1, notice that the state of the independent variables reads right to left for the corresponding state of the dependent variable. Also note that all truth tables will have only one dependent variable.

The truth table will have the number of lines necessary to represent <u>all</u> possible combinations of the independent logic variables.

For Boolean algebra, the number of lines in a truth table will be equal to 2^n where n is the number of independent variables. For example, a truth table for two independent variables require four lines, 2^2, to completely define all the possible combinations of the two variables.

With your knowledge of binary states you should be able to construct the independent part of any truth table at this point.

The following sections of this text will provide the information necessary to complete the dependent portion of the truth table.

3.2.2 The OR Operation

The first Boolean operation to be discussed is the logical OR operation. The OR operator operates on two or more Boolean variables. The result or dependent variable of the OR operation will be true if either one or both of the independent variables is true. The result will be false only if both of the independent variables are false. The truth table for the OR operation is shown in Figure 3-2.

FIGURE 3-2. Truth Table for the OR Operation.

A	B	f
0	0	0
0	1	1
1	0	1
1	1	1

The OR operation is called the Boolean addition operation. The notation for the OR operation performed on the Boolean variables A and B is f = A+B where f is the dependent variable or result. This logic operation is not the same as adding the binary integers A and B.

3.2.3 The AND Operation

The logical AND operation operates on two or more Boolean variables. The AND operation will only have a true result if both independent variables are true. In all other cases the result will be false. Figure 3-3 is the truth table for the logical AND operation.

FIGURE 3-3. Logical AND Truth Table.

$f = A \bullet B$ (f = A and B)

A	B	f
0	0	0
0	1	0
1	0	0
1	1	1

The logic AND operation is called the Boolean multiplication operation. The AND operation performed on the independent variables A and B is written as f = A•B. This is not the same as multiplying binary integers.

3.2.4 The NOT Operation

The NOT operation is the simplest Boolean operation. It is the only unary operation allowed in Boolean algebra. This means that the operation is performed on only one Boolean independent variable or on one Boolean logic expression. The NOT function returns the complement of the state of the

Boolean variable. A truth table for the NOT operation is shown in Figure 3-4.

$$f = \overline{A}$$

A	f
0	1
1	0

FIGURE 3-4. Truth Table for the NOT Operation.

The NOT operation will result in the opposite logic state of the Boolean variable on which it operates. This means that TRUE Boolean variables will return a value of FALSE when operated on by the NOT operator. The NOT operator is written into Boolean equations by placing a line over the complemented variable. $\overline{A}$ is read as "NOT A."

3.2.5 Logic Equations

The AND, OR, and NOT operators are all of the basic operations in Boolean algebra. All other operations are made from these three basic operations.

Boolean equations involve combinations of these three basic functions according to the rules of Boolean algebra. Boolean algebra has its own set of rules and laws. Boolean equations are written in the same form as other mathematical equations and may involve variables and constants (1 or 0).

Many of the rules of algebra that you already are familiar with will not change in Boolean algebra. Operations in Boolean equations are performed from left to right with logic multiplication first then logic addition after products are completed. Logic equations are frequently written in the form f= A op B op C... where op signifies a logical operation (AND or OR).

Some books will use the symbol Y instead of f for the result. This is not really new to most of you as you have seen the expressions Y and f(x) used interchangeably in other algebra courses. Some common laws of Boolean algebra are listed in Table 3-1.

The Laws of Absorption can be derived from the distributive laws and the laws of tautology. As mentioned previously, the Boolean operations can be combined to solve complex problems. Logic equations are a way of describing and analyzing logical functions.

TABLE 3-1. Basic Laws of Boolean Algebra.

Commutative Laws:

Addition $A+B = B+A$ Multiplication $A \bullet B = B \bullet A$

Associative Laws:

Addition $A+(B+C) = (A+B)+C$

Multiplication $A(B \bullet C) = (A \bullet B)\,C$

Distributive laws:

$A+(B \bullet C) = (A+B) \bullet (A+C)$

$A \bullet (B+C) = AB+AC$

Laws of Tautology:

$A \bullet A = A$

$A+A = A$

when constants are involved:

$A1 = A$ $A \bullet 0 = 0$ $A+1 = 1$ $A+0 = A$

The Laws of Absorption:

$A(A+B) = A$

$A+(A \bullet B) = A$

Identity Theorems:

$A + A \bullet B = A + B = A + B$

$A \bullet (A + B) = A \bullet B$

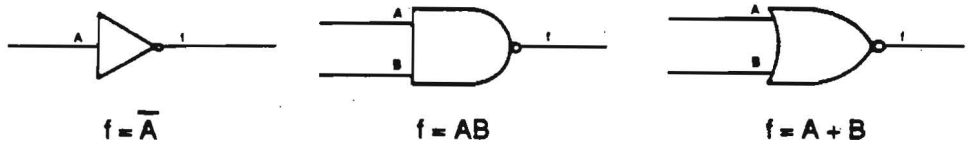
3.2.6 Logic Circuits

When George Boole formulated Boolean algebra, electronic switching had not been invented. For lack of a suitable method of implementing logic equations, Boolean algebra was unused in technology for nearly 100 years. The invention of reliable electronic switches stimulated the use of Boolean algebra for solving logic equations.

Early switches implemented with electron tube technology could solve Boolean equations more quickly than humans but large arrays of such switches were bulky and consumed enormous amounts of electrical power. The invention of semiconductor switches was eventually responsible for the widespread use of logic circuits evident today.

Semiconductor switches have been widely used because they provide a compact, efficient, economical, and reliable method of solving logic equations. A large number of logic equations have been implemented as digital integrated circuits. These devices use several transistor switches to solve logic equations. The electrical output of these circuits represents the state of the dependent variable in the logic equation. The electrical inputs to these circuits are the states of the independent variables.

Digital logic circuits are often referred to as logic gates. The schematic symbols for the gates used to implement the basic Boolean logic equations are shown in Figure 3-5.



$f = \overline{A}$ $f = AB$ $f = A + B$

Note: The circle is used to indicate the NOT function.

FIGURE 3-5. Schematic Symbols for Boolean Equations.

A logic equation can be solved by arranging logic circuits to perform the logical operations indicated by the logic equation. The electrical inputs to the circuit can be varied between the allowed states while the circuit output can be observed. This technique is used to create a truth table for the entire logic equation.

An example of converting logic equations to logic circuits is shown in Figure 3-6.

LOGIC EQUATION: $Y = A \bullet B+C$

LOGIC CIRCUIT: The variables A and B must be ANDed then the result ORed with the variable C.

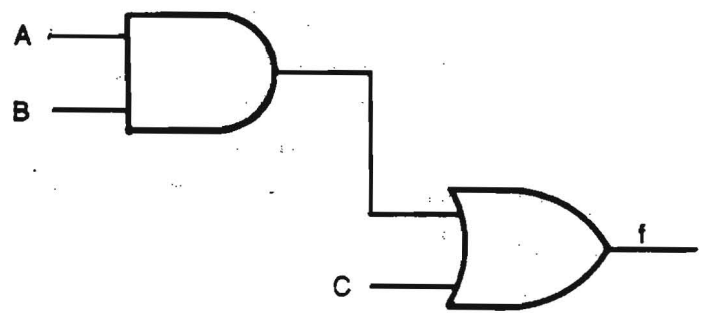


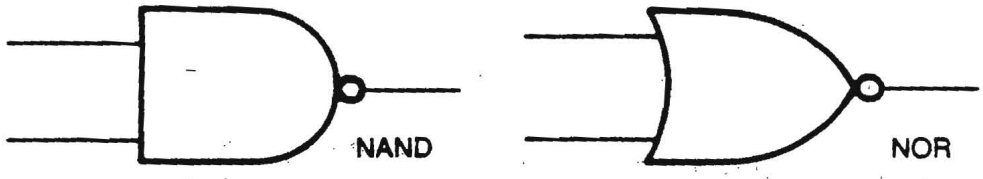
FIGURE 3-6. Designing Circuit From Logic Equations.

The electrical output of logic circuits will be one of tw. voltages. When the more positive of these voltages is used to represent a logic 1, HI or TRUE state, the device is termed to use positive logic. Devices using the opposite convention are said to use negative logic. Negative logic circuits are sometimes called "LO true." You will become familiar with constructing circuits from logic equations. This process is crucial to understanding digital systems.

3.2.7 NOR and NAND Gates

NOR and NAND gates are two of the simpler combinational logic circuits that are commonly available. These combinational logic circuits are used to combine logic functions for decision making. The logic equation for a NAND gate is $f = \overline{A \bullet B}$. The logic equation for a NOR gate is $f = \overline{A+B}$. The schematic symbols for the NAND and NOR gates are shown in Figure 3-7.

IGURE 3-7. Schematic Symbols for NAND and NOR Gates.



The NAND and NOR gates are the same as the AND and OR gates previously studied except that an inverter is built into the output of each circuit. Any digital circuit no matter how complex can be constructed entirely from only NAND or NOR gates. This can be readily demonstrated by implementing all the logic functions using only these gates. This fact is largely responsible for the popularity of these gates.

3.3 SUMMARY

In this chapter the concepts for the use and understanding of Boolean algebra were introduced. The three basic operations used in Boolean algebra were defined and explained. The use of logic equations and some basic rules of Boolean algebra were presented.

Implementing logical equations with electronic digital circuits was explained and the schematic symbols for some common logic circuits were identified. The NOR and NAND combinational logic circuits were introduced and the flexibility of these circuits explained. The concepts learned in this chapter will be used throughout the remainder of this book and your association with digital electronics.

3.4 REVIEW QUESTIONS

1. Who first formulated Boolean algebra ?

2. What is the practical use of Boolean algebra ?

3. What are the three basic operations allowed in Boolean algebra ?

   1.

   2.

   3.

4. How many variables does the AND function operate on?

5. How many variables does the NOT function operate on?

6. What is a truth table ?

7. Draw the truth table for the NAND function.

8. Draw the schematic symbols for the three basic logic operations.

9. Draw the schematic of a circuit that will perform th operations in the following logical equation: f = C+DF.

10. Draw and complete a truth table for the equation from question nine.

11. Name the combinational logic circuits .

_____

_____

12. What are combinational logic circuits used for ?

_____

13. Why have digital integrated circuits become so popular ?

_____

_____

LAB EXERCISE 3.1
The NOT Circuit (Inverter)

Objectives

After completion of this experiment you will understand the operation of logic inverters (NOT gates). You will be able to use the 74LS04 IC and explain its operation.

Materials

LD-2 Logic Designer

74LS04 Hex Inverter

Jumper Wires

TTL Data Book

Procedure

This section will begin your experiments with logic gates. You will learn some general characteristics of logic circuits then study the 74LS04 TTL hex inverter.

All logic circuits will have connections for power and ground. Logic circuits are usually seen as Dual Inline Package

Integrated Circuits known as DIP ICs. The term dual inline package describes the pin arrangement for the integrated circuit inputs and outputs.

All DIP ICs have one end or corner marked in a special way. This marking is used to show integrated circuit pin orientation. With the marked end of the IC facing away from you the pins are numbered counterclockwise from the upper left corner. Figure 3-8 shows how ICs are marked and how the pins are numbered.

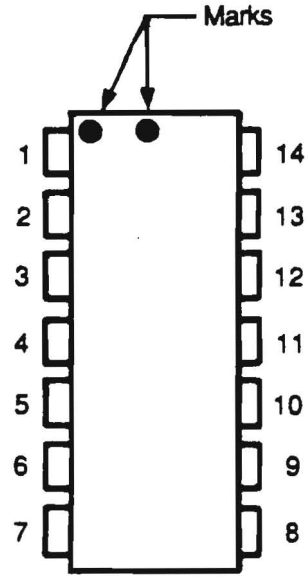
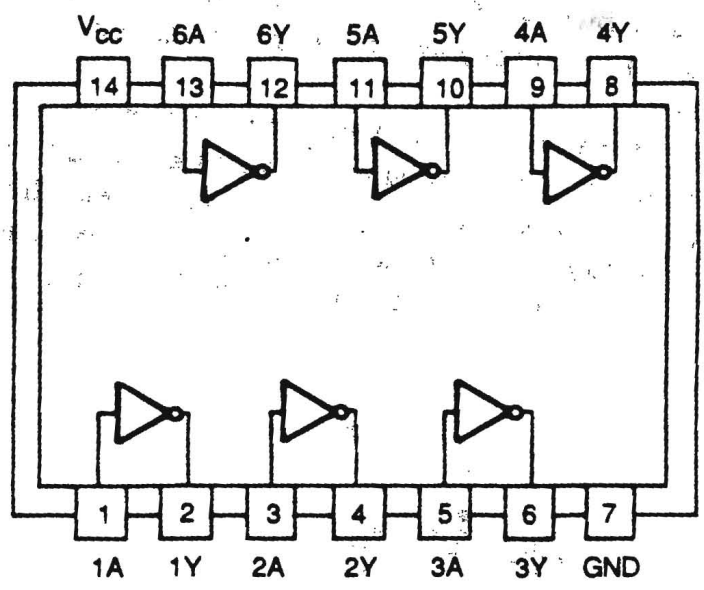


FIGURE 3-8. IC Orientation and Pin Numbering.

The ICs used in this experiment are TTL ICs. This means the voltage of the two logic states are 0 and 5 volts. A large number of compatible integrated circuits have been manufactured as the 7400 series of TTL ICs. Correct connection of power and ground pins is crucial to circuit operation. Many 14 pin DIP ICs use pin 7 for ground and pin 14 for Vcc. Some 14 pin DIPs use pin 11 for ground and pin 4 for Vcc. Most 16 pin DIP ICs use pin 8 for ground and pin 16 for Vcc. If you connect the power and ground connections incorrectly the IC will be destroyed. For this reason, a basing diagram of all ICs used in experiments is provided. The basing diagram explains all connections to an IC and uses schematic symbols to indicate logic functions performed by the circuit. A basing diagram for the 74LS04 hex inverter is shown in Figure 3-9.

iURE 3-9. Basing Diagram for 74LS04 IC.



If you have trouble during the experiment, remove power from the IC and check circuit wiring. Consult with your instructor if after repeated attempts to correct the problem fail.

1. Place 1 74LS04 IC onto the LD-2 breadboard.

2. Use the basing diagram of the 74LS04 to locate the power and ground pins.

3. Connect pin 7 to ground and pin 14 to +5 VDC. Power and ground connections are provided on the right most two-row breadboard section of the LD-2.

4. Connect pin 1 to PB1. PB1 is on the left most two-row breadboard. There are two PB1 connections. You should use the left one.

5. Connect pin 1 to L1 on the right most two-row breadboard. This will allow monitoring the inverter input.

6. Connect pin 2 to L2 (next to L1). This allows monitoring the inverter output.

7. Check circuit wiring. When you are certain that the circuit is correctly wired, connect power to the LD-2 and position the On/Off switch (the right corner of the LD-2 near the power plug) to on. Two lights should be on. D1 indicates power is on. L1 indicates the state of the inverter input. If both lights are off, disconnect power and check the circuit wiring particularly power and ground. If D1 is on and L1 is off turn the power off using the On/Off switch. Check circuit wiring paying careful attention to pins 1 and 2. If no problem is noted proceed to step 2. If you had problems retry step one.

8. Record the states of L1 and L2 (NOTE: an on light indicates a logic one). 1 0

9. Push PB1, the upper pushbutton at the lower left side of the LD-2. Record the states of L1 and L2. 0 0

10. Turn off power. Leave the 74LS04 IC connected till after you have finished the following questions.

Questions

1. Construct a truth table for the 74LS04 hex inverter.

2. Why is the 7404 called a hex inverter ? (hint: look at the basing diagram)

_____

_____

3. Are the lights L1 and L2 ever on simultaneously ?

LAB EXERCISE 3.2
The AND Gate

Objectives

In this laboratory you will learn the use of the 74LS08 quad two-input AND gate. You will observe and record the AND gate's logic characteristics.

Materials

LD-2 Logic Designer

74LS08 Quad Two-input AND Gate

Jumper Wires

TTL Data Book

Procedure

1. Insert the 74LS08 IC into the breadboard.

2. Wire pin 7 to ground and pin 14 to Vcc.

3. Wire S1 on the left two-row breadboard to pin 1 on the 74LS08 and L1 on the right two-row breadboard. This allows setting the state of pin 1 with switch S1 and observing it's state on L1.

4. Wire S2 to pin 2 on the 74LS08 and to L2. This allows setting and observing the state of pin 2.

5. Wire pin 3 on the 74LS08 to L3. This allows observation of the AND gate output.

6. Place S1 and S2 in their off state (toward the words "LOGIC SWITCHES" printed on the LD-2 circuit board).

7. Connect and turn on power. D1 should be lit. If D1 is not lit or other lights are on, then turn off power and recheck circuit interconnection.

8. Move S1 to ON. L1 should light. If L1 does not light check wiring to pins 1 and 2.

9. Turn S1 to OFF and S2 to ON. L2 should light.

10. Now use S1 and S2 to determine the truth table for the 74LS08. Record your results. Observe the circuit output on L3.

11. Remove power from the LD-2 and remove the circuit used for this experiment.

LAB EXERCISE 3.3
The OR Gate

Objectives

In this laboratory you will learn about the 74LS32 two-input OR gate.

Materials

LD-2 Logic Designer

74LS32 Quad Two-input OR Gate

Jumper Wires

TTL Data Book

Procedure

1. Insert the 74LS32 IC into the breadboard.

2. Wire pin 7 to ground and pin 14 to Vcc.

3. Wire S1 on the left two-row breadboard to pin 1 on the 74LS32 and L1 on the right two-row breadboard. This allows setting the state of pin 1 with switch S1 and observing it's state on L1.

4. Wire S2 to pin 2 on the 74LS32 and to L2. This allows setting and observing the state of pin 2.

5. Wire pin 3 on the 74LS32 to L3. This allows observation of the OR gate output.

6. Place S1 and S2 in their off state (toward the words "LOGIC SWITCHES" printed on the LD-2 circuit board).

7. Connect and turn on power. D1 should be lit. If D1 is not lit or other lights are on, then turn off power and recheck circuit interconnection.

8. Place S1 in the ON position. L1 and L3 should light.

9. Place S1 to OFF and S2 to ON. L2 and L3 should light.

10. Place S2 to OFF. Use the switches and lights to determine the truth table for the 74LS32. Record your observations.

11. Remove power from the LD-2 and remove the circuits used for this laboratory.

LAB EXERCISE 3.4
The NAND Gate

Objectives

In this laboratory you will learn the operation of the 74LS00 two-input NAND gate.

Materials

LD-2 Logic Designer

74LS00 Quad Two-input NAND Gate

Jumper Wires

TTL Data Book

Procedure

1. Insert the 74LS00 IC into the breadboard.

2. Wire pin 7 to ground and pin 14 to Vcc.

3. Wire S1 on the left two-row breadboard to pin 1 on the 74LS00 and L1 on the right two-row breadboard. This allows setting the state of pin 1 with switch S1 and observing it's state on L1.

4. Wire S2 to pin 2 on the 74LS00 and to L2. This allows setting and observing the state of pin 2.

5. Wire pin 3 on the 74LS00 to L3. This allows observation of the NAND gate output.

6. Place S1 and S2 in their off state (toward the words "LOGIC SWITCHES" printed on the LD-2 circuit board).

7. Connect and turn on power. D1 and L3 should be lit.

8. Move S1 to the ON position. L1 should light.

9. Move S1 to OFF and S2 to ON. L2 should light.

10. Move S1 to OFF. Use S1, S2 and L3 to determine the truth table of the 74LS00 IC. Record your observations here.

11. Remove power and then disassemble the laboratory circuit.

LAB EXERCISE 3.5
The NOR Gate

Objectives

In this experiment you will learn the use of the 74LS02 two-input NOR gate.

Materials

LD-2 Logic Designer

74LS02 Quad Two-input NOR Gate

Jumper Wires

TTL Data Book

Procedure

1. Insert the 74LS02 IC into the breadboard.

2. Wire pin 7 to ground and pin 14 to Vcc.

3. Wire L1 on the right two row breadboard to pin 1 on the 74LS02. This allows observing the state of pin 1 (NOR gate output) with L1.

4. Wire S2 to pin 2 on the 74LS02 and to L2. This allows setting and observing the state of pin 2.

5. Wire pin 3 on the 74LS00 to L3 and S3. This allows setting and observing the state of pin 3.

6. Place S3 and S2 in their off state (toward the words "LOGIC SWITCHES" printed on the LD-2 circuit board).

7. Connect and turn on power. D1 and L1 should be lit.

8. Place S2 to ON. L2 should light (L1 will go OFF).

9. Turn S2 OFF and S3 ON. L3 should light.

10. Turn S3 OFF. Use S2, S3 and L1 to determine the truth table of the 74LS02 IC. Record your observations.

11. Remove power and disassemble the laboratory circuit.

LAB EXERCISE 3.6 Using NAND and NOR Gates

Objectives

In this experiment you will confirm that NOR and NAND gates can be used to perform any logic function.

Materials

LD-2 Logic Designer

74LS02 Quad Two-input NOR IC

74LS00 Quad Two-input NAND IC

Jumper Wires

TTL Data Book

Procedure

1. Install the 74LS02 IC on the breadboard.

2. Wire ground to pin 7 and Vcc to pin 14.

3. Wire S2 to pin 2 and S3 to pin 3.

4. Wire S2 to L2 and S3 to L3.

5. Wire pin 1 to pins 5 and 6. Shorting pins five and six

causes gate 2 of the quad IC to act as an inverter. (You may want to convince yourself of this.)

6. Wire pin 6 to L6. This is the inverter input.

7. Wire pin 4 to L4. This is the inverter and overall circuit output.

8. Turn S2 and S3 to OFF.

9. The schematic for the circuit you have just constructed is shown in Figure 3-10.

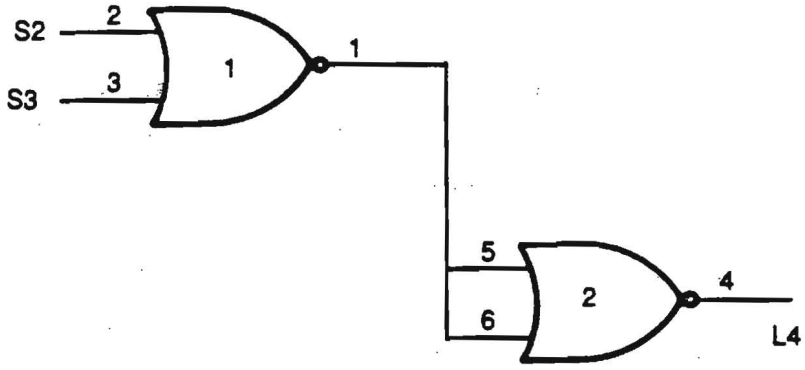


FIGURE 3-10. Circuit Schematic.

10. Turn on power. D1 and L6 should light.

11. L2 and L3 monitor the state of S2 and S3. L6 monitors the NOR gate output. L4 monitors the circuit output. Use S2, S3 and L4 to create a truth table for this circuit. Record this truth table. What logic operation is this ?

---

12. Turn off power. You will now wire inverters onto the inputs of the existing circuit.

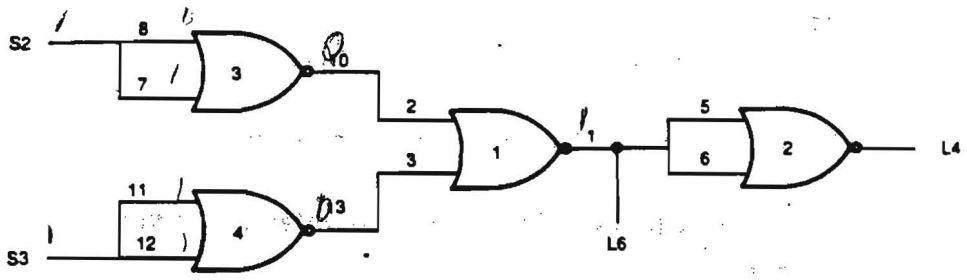
13. Remove the wire at S2 and connect it to pin 10.

14. Remove the wire at S3 and connect it to pin 13.

15. Connect S3 to pins 11 and 12.

16. Connect S2 to pins 8 and 9. The schematic for this circuit is shown in Figure 3-11.

FIGURE 3-11. Circuit Schematic.



17. Turn S1 and S2 OFF. Turn on power. Use S2, S3 and L4 to make a truth table for the circuit. Record your observations. What logic function is performed by this circuit ?

_____

18. Use S2, S3 and L6 to make a truth table for the circuit consisting of gates 4, 3 and 1. Record your observations here. Which logic function is implemented by this circuit ?

_____

19. Flip switches S2 and S3 ON and OFF together while watching L6 and L4 (output). Record your observations in the form of a truth table.

Questions

1. Which logic function is performed by the circuit observed in step 19? _____

2. All of the basic Boolean functions have been demonstrated using the 74LS02 quad two-input NOR gate. Design a circuit to implement the basic Boolean functions. Use the 74LS00 quad two-input NAND as your IC. Describe which gate combinations perform which Boolean functions. Breadboard your circuit and check it's operation.