Binary Logic

2.1 Definition of Binary Logic

Binary logic consists of binary variables and logical operations. The variables are designated by letters such as A, B, C, x, y,z, ..etc., with each variable having two and only two possible values (0,1). There are three basic logical operations: AND, OR, and NOT.

2.2 logic levels

In digital system two voltage levels represent the two binary digits, 1 and 0. if the higher of two voltages represents 1 and the lower represents a 0, this is called positive logic. On other hand, if the lower voltage represents 1 and the higher voltage represents a 0 we have negative logic system, to illustrate, suppose that we have +5 V and 0V as our logic level voltages. We will designate the +5 V as the HIGH level and 0V as the LOW level, so logic can be defined as:

logic
HIGH = 1
LOW = 0

2.3 logic Gates

Logic gates are electronic circuits that operate on one or more input signals to produce an output signal. A logic gate has the basic format shown below in Figure 2-1. Each time the inputs change, the output bit changes in a predictable fashion.

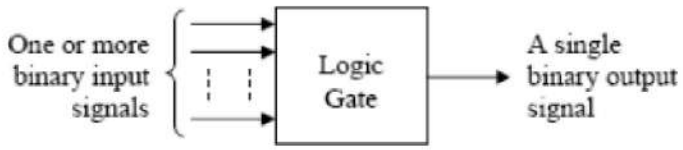


Figure 2-1 Basic Format of a Logic Gate

A number of standard gates exist, each one of which has a specific symbol that uniquely identifies its function. Figure 2-2 presents the symbols for the four primary types of gates that are used in digital circuit design.

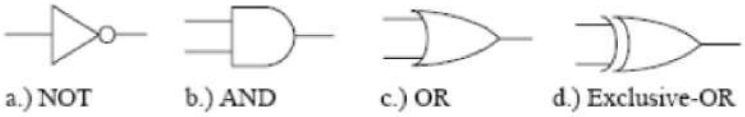


Figure 2-2 Basic Logic Symbols

Not Gate

This logic gate, sometimes referred to as an inverter, it has a single input. Its input goes into the left side of a triangle symbol while its output exits the gate through a small circle placed at the tip of the opposite corner. The NOT gate is used to flip the value of a digital signal. It changes logic 1 input to logic 0 or it changes a logic 0 input to logic 1.

$X=\overline{A}$

Truth table of NOT

A	X
0	1
1	0

AND Gate

The operation of the AND gate is such that its output is a logic 1 only if all of its inputs are logic 1. Otherwise the output is logic 0. The AND gate in Figure 2-2

has only two inputs, but an AND gate may have as many inputs as the circuit requires. Regardless of the number of inputs, all inputs must be logic 1 for the output to be logic 1.

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

X=A.B

OR Gate

An OR gate outputs a logic 1 if any of its inputs are logic 1. An OR gate only outputs a logic 0 if all of its inputs are logic 0. An OR gate may have as many inputs as the circuit requires. Regardless of the number of inputs, if any input is logic 1, the output is logic 1.

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

X=A+B

Exclusive-OR (XOR) Gate

An Exclusive-OR gate is sometimes called a parity checker. Parity checkers count the number of ones being input to a circuit and output logic 1 or 0 based on whether the number of ones is odd or even. The Exclusive-OR (XOR) gate counts the

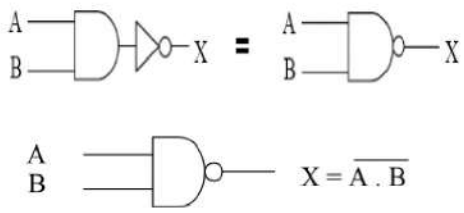
number of ones at its input and outputs logic 1 for an odd count and logic 0 for an even count. The XOR gate may have two or more inputs.

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

$$X = A \oplus B = \overline{A}.B + A.\overline{B}$$

NAND gate

NAND gate is a cascade of AND gate and NOT gate, as shown in the figure below. It has two or more inputs and only one output. The output of NAND gate is HIGH when any one of its input is LOW (i.e. even if one input is LOW, Output will be HIGH).



Boolean Expression for the NAND Function

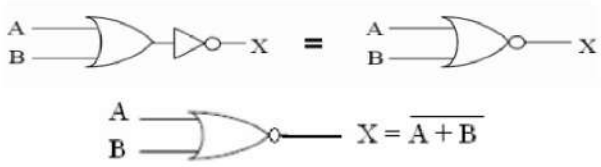
A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

NOR gate

NOR gate is a cascade of OR gate and NOT gate, as shown in the figure below. It has two or more inputs and only one output. The output of NOR gate is HIGH when any all its inputs are LOW (i.e. even if one input is HIGH, output will be LOW).

A	B	X
0	0	1

0	1	0
1	0	0
1	1	0

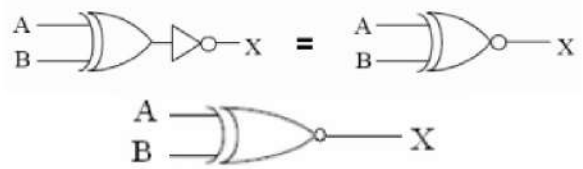


Boolean Expression for the NOR Function

Exclusive-NOR (XNOR) gate

An Exclusive-NOR (XNOR) gate is gate with two or three or more inputs and one output. The output of a two-input XNOR gate assumes a HIGH state if all the inputs assumes same state. This is equivalent to saying that the output is HIGH if both inputs X and input Y is HIGH exclusively or same as input X and input Y is LOW exclusively and LOW when both are not same.

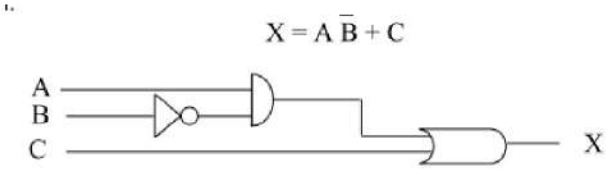
A	B	X
0	0	1
0	1	0
1	0	0
1	1	1



Boolean Expression for the XNOR Function $X = \overline{A \oplus B}$

2.4Networks and Expressions

Combining AND gates, OR gates and inverters, we can construct a logic network representing a given Boolean expression. The logic network for the Boolean expression:



Home work:

Draw the logic circuit represented by each of the following expressions:

1- $A + B + C$

2- ABC

3- $AB + CD$

4- $\overline{A}B(C + \overline{D})$

2.5Truth Tables for Combinational Logic

Not all digital circuits lend themselves to quick conversion to a truth table. For example, input B in the digital circuit shown in Figure 2.6 passes through four logic gates before its effect is seen at the output.

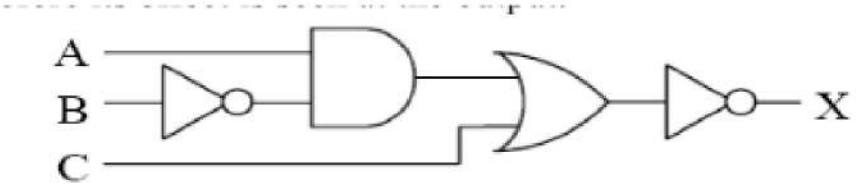
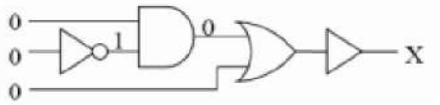


Figure 2-6 Sample of Multilevel Combinational Logic

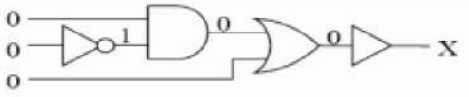
So how do we convert this circuit to a truth table? One method is to go through each pattern of ones and zeros at the input and fill in the resulting output in the truth table row by row. Figure 2.7 takes A=0, B=0, and C=0 through each gate to determine its corresponding output.

a.) A 0 is input to the first inverter which outputs a 1.

b.) The 1 coming from the inverter is combined with a 0 in the AND gate to output a 0.



c.) The OR gate receives a 0 from the AND and a 0 from the inputs which makes it output a 0.



d.) The 0 output from the OR gate passes through the inverter output a 1.

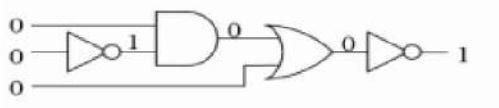


Figure 2-7 Process of Passing Inputs Through Combinational Logic

Figure 2.7 represents only one row of a truth table with eight rows. Add another input and the truth table doubles in size to sixteen rows. There is another way to determine the truth table. Notice that in Figure 2.7, we took the inputs through a sequence of steps passing it first through the inverter connected to the input B, then through the AND gate, then through the OR gate, and lastly through the inverter connected to the output of the OR gate. These steps are labeled (a), (b), (c), and (d) in Figure 2.8.

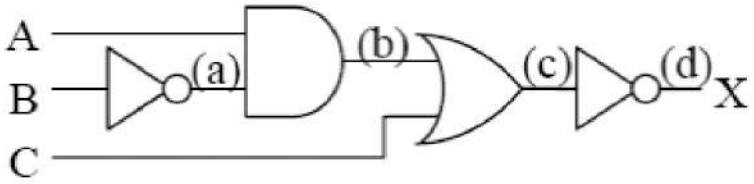


Figure 2-8 Steps That Inputs Pass Through in Combinational Logic

Begin by creating the input columns of the truth table listing all of the possible combinations of ones and zeros for the inputs. In the case of our sample circuit, that gives us a truth table with eight rows.

Next, add a column for each layer of logic. Going back to Figure 2.8, we begin by making a column representing the (a) step. Since (a) represents the output of an inverter that has B as its input, fill the (a) column with the opposite or inverse of each condition in the B column.

Then, step (b) is the output of an AND gate that takes as its inputs step (a) and input A. Add another column for step (b) and fill it with the AND of columns A and (a).

Step (c) is the output from the OR gate that takes as its inputs step (b) and the input C. Add another column for (c) and fill it with the OR of column C and column (b).

Last of all, the final output is the inverse of the output of the OR gate of step (c). Make a final column and fill it with the inverse of column (c). This will be the final output column for the truth table.

A	B	C	(a)=Not of B	(b)=(a) AND A	(c)=(b)OR C	X=(d)=NOT C
0	0	0	1	0	0	1
0	0	1	1	0	1	0
0	1	0	0	0	0	1
0	1	1	0	0	1	0
1	0	0	1	1	1	0
1	0	1	1	1	1	0
1	1	0	0	0	0	1
1	1	1	0	0	1	0

Home Work:

Develop the truth table of the combinational logic circuit shown below.