# Unit 5 Logic gates.

- The voltages which are at the inputs to logic gates are determined by the integrated circuit technology in use.
- Interface circuits allow the voltage levels to be changed for conversion from one family type to another.
- Logic gates are represented by symbols which are independent of the type of logic technology in use.

The integrated circuit technologies discussed in the first four units form a logic family tree as shown in Figure 5.1. Each of these families has its own operating voltages and voltage signal levels. In order to convert from one family to another special translator integrated circuits have been developed which allow, say, ECL logic levels to be converted to TTL logic levels so that different technology ICs can be used in different parts of a large circuit.

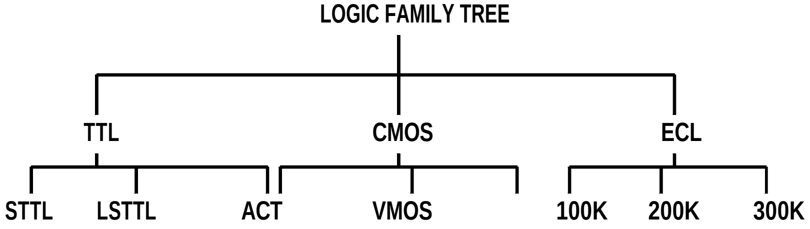

Figure 5.1: Logic gate families.

In general, the number of such conversions is minimized but a good hardware system design would attempt to optimize the most efficient mix of speed, power dissipation and price with ECL, CMOS and TTL all working together. ECL would be used for the speed critical applications, CMOS would be used to keep the system power consumption down and TTL would be used to keep the system costs down. This system optimization process is complicated by considerations of the size of the production runs for the circuit under consideration. It may be that, for large numbers of a specific circuit, the design costs of a customized IC can be justified with the resulting dramatic cost drop which, in a mass market, can result in increased product sales.

The ever increasing expectation of improved product performance can lead to the introduction of advanced technologies in surprisingly mundane applications. For instance, a high definition 1024 × 1024 digital TV display will require of the order of 80 million switching operations per second which is best implemented using the ECL technology ICs.

Every circuit will therefore be a compromise between available technologies and external technical and commercial constraints. However, the internal logical principles of the circuits will remain invariant and it is these principles which we will study in this text.

No matter which of the logic families from Figure 5.1 is employed, there will be an agreed convention about the voltages which represent the Boolean terms, logic 0 and logic 1 and the voltage representing the inputs and outputs of the ICs. You will encounter other names which are used to designate these same Boolean terms. For instance, in C programming the terms TRUE and FALSE are used. In electronics the terms HIGH and LOW are used. In switch circuits the terms ON or OFF are used which leads to a possible table of equivalences:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Logic 0 | OFF | FALSE | LOW | Not OK | STOP | RED | NO | 0 V |
| Logic 1 | ON | TRUE | HIGH | OK | GO | GREEN | YES | +5 V |

Great care must be taken to be unambiguous about the meanings of the terms. It has been said that in China, in honour of the revolution, a red traffic light means go and a green traffic light means stop which is not the convention followed in most other countries. Similarly, in Sri Lanka a vertical nodding of the head is reputed to signify no. In most other cultures it means yes. The table might have to be rewritten if it were to be used in these countries. The 0 V and 5 V are correct for TTL circuits but may not always be correct for CMOS circuits depending on the supply voltages used and 0 V and 5 V have no validity in ECL circuits which use negative voltages.

When a circuit or program has been setup or constructed, there are many well documented procedures which can be used to test the system and verify that it performs to the design specification. There are, however, few procedures which enable you to check a real world event actually translates to the expected logic signal. The only real test is to examine the response to every possible real event. This is easy for frequent events such as testing the indicator light which shows that the wheels of an aircraft have been lowered prior to landing—simply lower the wheels and the light should come on. It is not so easy in the case of rare events such as warning of fire in the engine of the aircraft. You can not light a test fire in the engine. There have been cases of cross wired alarms where an actual fire in the starboard engine was indicated as a fire in the port engine leading to a shutdown of the only working engine. It is therefore essential that all parts of digital circuits be tested not only for logical conformity to the design but also for correct connection to the external world. A high proportion of catastrophic system failures occur due to faults originating at the interface.

We will now assume that we have operating electronic hardware, of whatever type, whose functioning can be represented by a table such as the one above and we will simplify the representation to use only logic 0 and logic 1. In constructing circuits we will use a small number of primitive logic gate symbols together with their truth tables. These logic gates are the AND and the the negated output version NAND gate. The negation of an input or output is represented by a circle at the input or output of the gate as is shown in Figure 5.3.

| A | B | Q |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

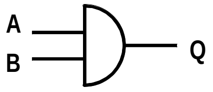

Figure 5.2: The AND gate.

| A | B | Q |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

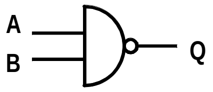

Figure 5.3: The NAND gate.

The OR gate also has an negated output version as shown in Figures 5.4 and 5.5.

| A | B | Q |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

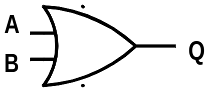

Figure 5.4: The OR gate.

| A | B | Q |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

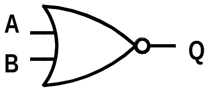

Figure 5.5: The NOR gate.

Finally there is the simple INVERT or NOT gate shown in Figure 5.6.

| A | $\bar{A}$ |
|---|---|
| 0 | 1 |
| 1 | 0 |

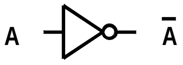

Figure 5.6: The INVERT gate.

A fundamental feature of digital electronics is that any digital function can be implemented using a subset of these primitive gates, that is by using only the gates in any one row of the table below.

| | |
|---|---|
| AND | INVERT |
| OR | INVERT |
| NAND | |
| NOR | |

## 5.1 Examples

5.1 Show that the circuit in Figure 5.7 implements a NAND gate.

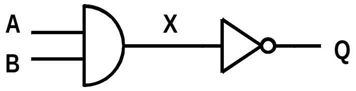

Figure 5.7: Example 5.1.

In solving this example we introduce the concept of intermediate points such as that marked X in the circuit. We include this intermediate point in the form of an extra column in the truth table so that in this case, the output Q is the invert of the signal at point X. We also denote the invert of a signal X by the symbol $\overline{\mathrm{X}}$ or X Bar.

| A | B | X | $Q = \overline{X}$ |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

In this truth table there are two primary inputs A and B each of which have two possible values 0 and 1. The number of rows in the truth table which allows the enumeration of all possible combinations of the inputs A and B is $2^2 = 4$. In the more general case of $n$ independent inputs A, B, C,..., M, we will have a truth table with $2^n$ rows representing all possible combinations of the possible input signals.

Comparison of this truth table with the truth table for a NAND gate shown in Figure 5.3 shows that the circuit does implement a NAND gate as required.

5.2 Analyze the operation of the circuit shown in Figure 5.8 (a).

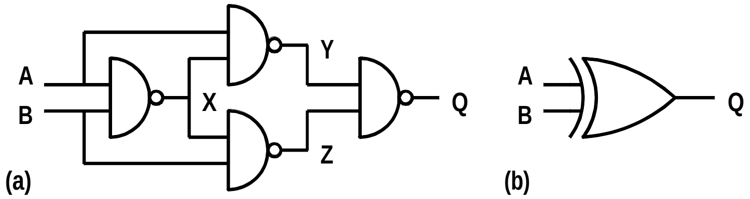

Figure 5.8: The XOR Exclusive OR gate.

We mark the intermediate points X, Y, Z in the circuit and introduce three extra columns into the truth table.

| A | B | X | Y = A NAND X | Z = B NAND X | Q = Y NAND Z |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 |

This is a very important circuit which is given a symbol of its own—the XOR as shown in Figure 5.8 (b). In words this circuit has a logic 1 at the output when either input but not both inputs is an logic 1. It is an exclusive OR function and is called either XOR or EOR in textbooks.

## 5.2 Problems

5.1 Draw a logic gate diagram which uses NOR gates which implements a two input NAND gate. Construct the truth table and show all intermediate points as columns in the truth table.

5.2 Construct the truth table for the circuit shown in Figure 5.9.

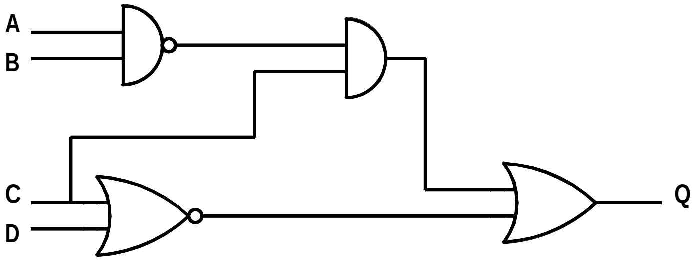

Figure 5.9: Problem 5.2.

5.3 Draw a logic gate diagram, using NOR gates, which implements the XOR gate function.

5.4 Show that the gate circuit in Figure 5.10 forms an XOR gate. Note that the circles at the inputs to the AND gates denote inversion of that input.

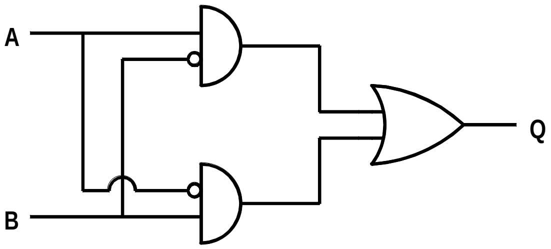

Figure 5.10: Problem 5.4.

5.5 What is the output Q from the logic gate diagram shown in Figure 5.11 for all combinations of A, B, C and D? Expresse Q as a function of A, B, C and D.

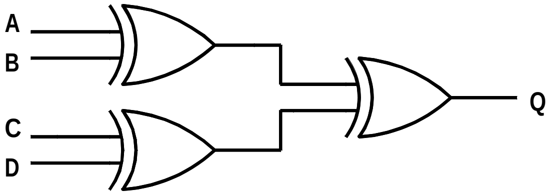

Figure 5.11: Problem 5.5.