Introduction to Digital Logic Circuits

An Online Continuing Education Course for Engineers

Course Number: IC-1003

Credit: 1 Hour / 1 PDH / 1 CPD

© 2023 Decatur Professional Development, LLC. All rights reserved.

Short History of Digital Logic

In 1835, Joseph Henry invented the electro-mechanical relay. The relay is a device in which a small current in the relay coil can close the relay contacts and allow larger currents to flow in an electrical circuit. It is a common component in digital logic circuits. In 1845, George Boole developed a mathematical theory of logic which is fundamental to the design of digital logic circuits. It wasn't until 1937 that Claude Shannon, an Electrical Engineer at MIT noted that Boolean algebra applies to relay and switching circuits. He wrote *"A Symbolic Analysis of Relay and Switching Circuits"* as his master's thesis, which he later published. The significance of this is that Claude Shannon is considered the founder of practical digital circuit design theory. Shannon provided several examples using his logic design methods: an electric combination lock, an adder, a voting machine and a circuit that would find factors and prime numbers. (He suggested that relay logic, operating at five operations per second, could more accurately and faster perform what a mathematician took twenty years to accomplish. He estimated it would take two months for his device to accomplish the same work.)

In 1947, Bardeen, Braittain and Shockley invented the transistor at Bell Labs. This resulted in solid state switching, that is much faster and more reliable than relays. This enabled the creation of large and powerful computers. In 1958, Jack Kilby and Robert Noyce invented integrated circuits that enabled more and less expensive digital circuits in a smaller package. These were used in the space program in which weight is an important factor. In 1969, Dick Morley invented the first Programmable Logic Controller (PLC), the MODICOM Model 84. The PLC is designed for the more rugged applications and more power that is required for manufacturing. These devices have replaced control relays in many manufacturing areas. In 1971, Robert Noyce and Gordon Moore introduced a "Computer on a chip". It executed 60,000 operations per second which is substantially more than the 5 operations per second for Shannon's relay logic device. Improvements in integrated circuits and microprocessors have enhanced the functionality of Programmable Logic Controllers. In mid 1970's through 1980's, Allen Bradley produced PLC1 through PLC5 series of Programmable Logic Controllers using integrated circuits and microprocessors. In 1980, IBM started production of the IBM Personal Computer (PC) which made computing available to all. The PC is useful for both programming Programmable Logic Controllers and for the analysis and design of digital logic circuits. In addition to computers and the PLC, digital circuits are used in cell phones and other mobile devices, automobiles, medical devices, security systems, household appliances, energy management systems and High Definition Television (HDTV).

Boolean Algebra Fundamentals

Unlike other algebras, Boolean algebra allows only two possible values: 0 or 1. The two value representation is often referred to as a bit (**Bi**nary digi**t**). The bit can represent on or off, true or false, or yes or no. Input variables such as A, B, C are used in logic expressions to represent switches, relay contacts and sensor inputs. The input variables can represent push button switches, limit switches, temperature sensitive switches, pressure switches, level switches, proximity sensors, light sensors, and weight sensors. The results of the logical expression are to perform some action: turn on or off solenoids, relays, motors, lights, buzzers, alarms and other output devices. Boolean algebra has only three basic operations: NOT or Negation, AND (·), and OR (+).

The NOT operation changes the value to the complement of the value. It changes a 1 to a 0, and a 0 to a 1. It can be represented by a bar over the variable. This is also referred to as inversion or negation.

The AND operation is a Boolean multiply and is 1 only when all values are 1. This is represented in Boolean variables as $A \cdot B = Y$ or $AB = Y$. The AND is like the old string of Christmas lights. All the lights have to work to have the string light. The AND operations for $A \cdot B = Y$ are shown for each input value.

$$A \cdot B = Y$$
$$0 \cdot 0 = 0$$
$$0 \cdot 1 = 0$$
$$1 \cdot 0 = 0$$
$$1 \cdot 1 = 1$$

The OR operation results in a 1 if any of the values are 1. In Boolean algebra, $1 + 1 = 1$ is valid. Logically the statement means if there are one or more true values, the result is true.

$$A + B = Y$$
$$0 + 0 = 0$$
$$0 + 1 = 1$$
$$1 + 0 = 1$$
$$1 + 1 = 1$$

Special Properties Associated with Boolean Logic Operations

There are properties of Boolean operations that are important in working with logic expressions. Using these properties, Shannon showed that logical expression can be expanded and simplified which in turn resulted in simpler logic and reduced the number or relays and switches.

Commutative: $A + B = B + A$

$A \cdot B = B \cdot A$

Changing the order of the input variables does not change the result.

Associative: $(A + B) + C = A + (B + C)$

$(A \cdot B) \cdot C = A \cdot (B \cdot C)$

Changing the order of the logical operations does not change the result.

Distributive: $A(B + C) = A \cdot B + A \cdot C$ Sum of Products (SOP)

$A + B \cdot C = (A + B)(A + C)$ Product of Sums (POS)

The input variables are distributed to the individual terms.

Identity: $1 \cdot A = A$, $0 + A = A$

1 AND a variable is the same as the variable, 0 OR a variable is the same as the variable.

Complement: $A + \bar{A} = 1$, $A \cdot \bar{A} = 0$

The OR of a complement is 1 and the AND of a complement is 0.

De Morgan's Theorem: $\overline{A \bullet B} = \bar{A} + \bar{B}$

The NOT of A AND B is the same as NOT A OR NOT B

Example

Using these properties, logic expressions can be simplified. The expression $A \cdot B + \bar{A} \cdot B$ can be rewritten as $B(A + \bar{A})$, which is $B \cdot 1$ or simply B. This shows that instead of two switches, A and B, only one switch, B is needed.

Solutions and Contradictions

Each variable in a logic equation has two possible values. For each variable added to the equation, the number of combinations and possible solutions doubles. The number of solutions related to the number of variables is exponential: 2^n, where n is the number of variables. For example, a circuit with ten switches requires ten variables and can have 2^{10} or 1024 possible combinations and solutions. As the number of switches and sensors increase, the design becomes much more difficult.

Algebraic equations do not always have a solution. When an equation does not have a solution it is called a **contradiction.** An example of a contradiction is $A = \bar{A}$. This equation states that A is equal to NOT A, or A is not A. When A is 1, $\bar{A}$ is not 1. There is no combination of one's and zero's that will make the equation correct, so this equation is a contradiction. There is a similar equation, but the results are a unique solution: $A \cdot B = \bar{A}$. This equation is correct only when A = 1 and B = 0, and no other combination. Even though this is a valid equation, since A and B values are fixed, neither A or B are switches. If switches were used, switch A would always be on, and switch B would always be off. This would be a waste of two switches. In more complicated logic, there can be situations where the logical result does not change whether the switch is on or off. In these cases, it is likely the switch is not needed.

Ladder Logic and Logic Gates

Originally digital logic was accomplished only through switches and relays. Switches and Relays are still commonly represented in Ladder Logic Diagrams (LLD). In Ladder Logic, logic is defined using rungs. On the left, contacts define the logic operations. On the right is the output of the logic. These are shown below for the basic logic operations: NOT, AND and OR. Digital logic is also implemented using digital electronic circuits called logic gates. Gates serve as the building blocks to more complex electronic digital logic circuits. They are fundamental to the design of computers. Digital logic using transistors is often referred as Transistor-Transistor Logic or TTL. The basic electronic digital operations are also NOT, AND, OR, but there are combinations of these that are also commonly used: the NAND, NOR, and XOR.

NOT Logic

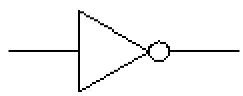
In Ladder Logic, a normally closed contact represents NOT logic.

Normally closed contact   Power On

|------------------]\[------------------------------------( )----------|

With the Normally closed contact not activated, the Power On output is on. When the Normally closed contact is activated or is “on”, the Normally closed contact opens and the Power On output is off. NOT gates that perform the same function have a single input and a single output. A symbol for a NOT gate is a triangle with a circle on the right. The circle indicates "negation". The NOT logic gate symbol is given below along with the Boolean equation:

$$\bar{A} = Y$$



A **truth table** shows all the possible combinations and the outputs for a logic equation. The truth table for the NOT operation is shown below:

A	Y
0	1
1	0

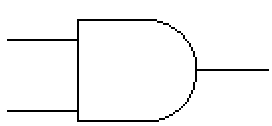
AND Logic

In Ladder Logic, the AND operation consists of two or more normally open contacts in series.

A   B   Out
|---------------] [--------------] [-------------------( )----------|

The above shows two contacts on the rung. Additional contacts could be added to the rung as required. An AND gate can have two or more inputs and has a single output. The output of the AND gate is 1 only if all inputs are 1. Otherwise, the output is 0. An AND gate symbol for a two input AND gate is shown below along with the Boolean equation.

$$A \cdot B = Y \text{ or } AB = Y$$



The truth table of a two input AND logic is the following:

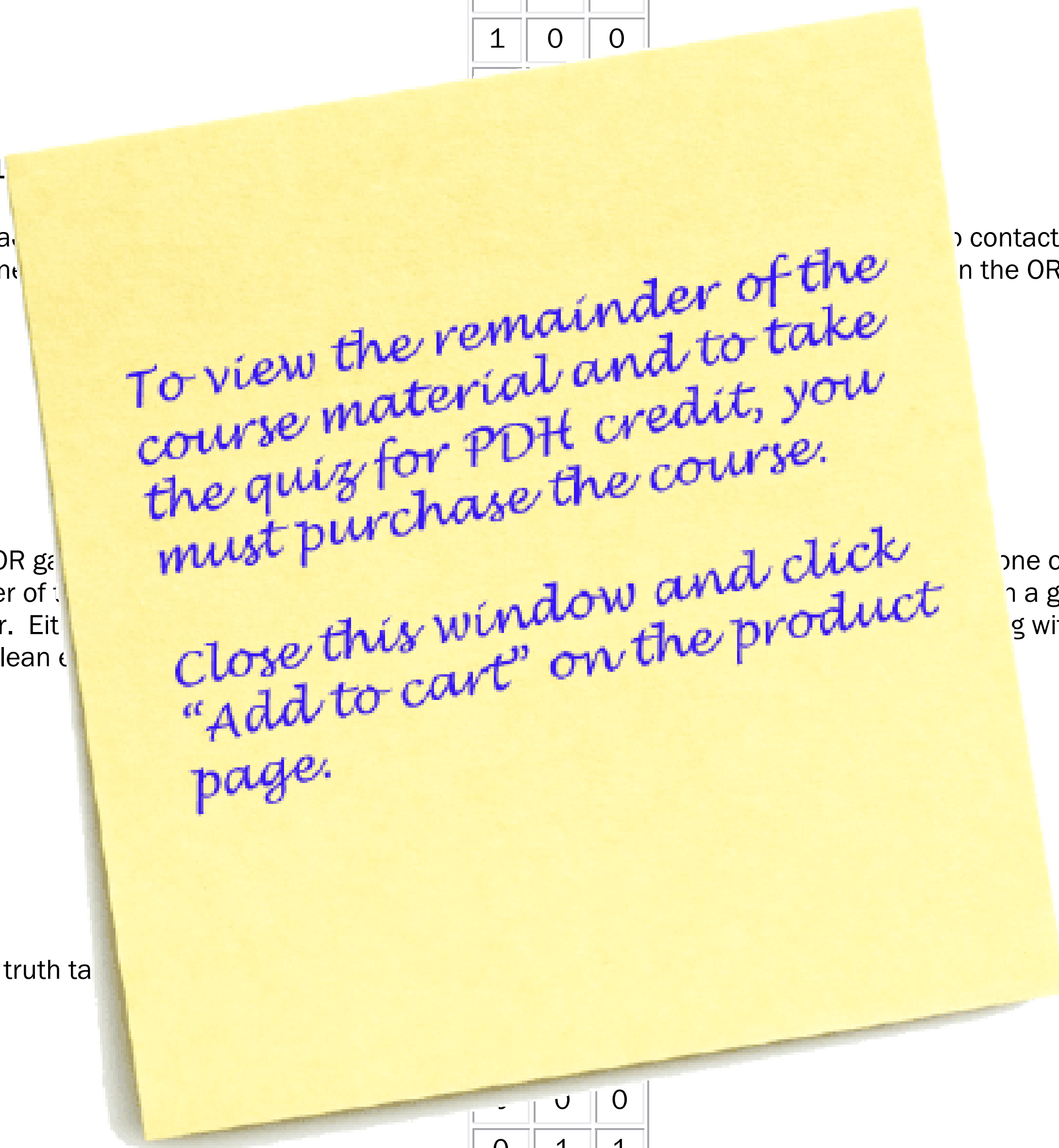
A	B	Y
0	0	0
0	1	0
1	0	0

OR L

In La … contacts
conn … n the OR logic.

An OG ga … one or any
other of … n a garage
door. Eit … g with the
Boolean e

The truth ta



[illegible]	[illegible]	0
0	1	1
1	0	1
1	1	1