# Logic Gates and Boolean Algebra

### Introduction

In 1847 the mathematician George Boole introduced a shorthand notation for expressing and dealing with a system of logic. Logical Statements are considered as being either **True** or **False**.

In Boolean Logical a statement that can be either True or False is considered to be a logical variable.

**Examples**

* All the AS Computing students are present. (Statement X)
* The car park is full. (Statement Y)

|  |  |  |
| --- | --- | --- |
| X | | Meaning |
| True | 1 | All the AS Computing students are present |
| False | 0 | Not all the AS Computing students are present |

Boolean algebra was of little use until digital electronics and computers were invented.

Boolean algebra only uses True or False. The Binary Number System only uses 1s or 0s. Hence:

* 1 = True
* 0 = False

### Truth Tables

The binary states of Boolean variables can be conveniently represented a simple switch. 1 = Switch Closed 0 = Switch Open

= 0

= 1

### OR Function

X Lamp (Q)

Y

If X or Y is closed the Q is on. This is written as the Boolean equation:

X + Y = Q

|  |  |  |
| --- | --- | --- |
| X | Y | Q  X OR Y |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

**OR**

X

Y

Q

(This is an OR Symbol)

**Q = X OR Y = X + Y**

### AND Function

X Y Lamp (Q)

If X and Y are closed the Q is on. This is written as the Boolean equation:

X . Y = Q

|  |  |  |
| --- | --- | --- |
| X | Y | Q  X AND Y |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

X

**AND**

Q

Y

**Q = X AND Y = X.Y**

(The above is an And Symbol)

### NOT Function

This is an operator that reverses the outcome. When a switch is open the lamp is on and when it is closed it is off.

Q = Not X or 

**NOT**

|  |  |
| --- | --- |
| X | Q  Not X |
| 0 | 1 |
| 1 | 0 |

### Combinations of AND and OR

Y

X Q

Z

We define condition Q as Q = 1 when the current can flow and Q = 0 when the current can not flow.

By inspection if X and Y are closed or X and Z are closed then the current can flow.

Q = (X AND Y) OR (X AND Z)

Q = X.Y + X.Z

Alternatively

Q = X AND (Y OR Z)

Q = X.(Y + Z)

This tells us that normal algebra rules apply.

i.e. Q = X.(Y + Z)

Q = X.Y + X.Z

### Distributive Law

Q = X.(Y + Z) = X AND (Y OR Z)

Q = X.Y + X.Z = (X AND Y) OR (X AND Z)

### Logic Gates

Electronic circuits that perform Boolean algebra are called Logical Gates.

### NAND Function

This is the Boolean function which is simply a combination of the AND and NOT function. This can be represented using the logic circuit below.

A A.B 

**AND**

**NOT**

B

|  |  |  |
| --- | --- | --- |
| Input A | Input B | Output  **NAND** |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

### NOR Function

This is the Boolean function which is simply a combination of the NOT and OR function. This can be represented using the logic circuit below.

**NOT**

A A + B 

**OR**

B

|  |  |  |
| --- | --- | --- |
| Input A | Input B | Output  **NOR** |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

### XOR Function

This function differs from OR in that the output is 1 when one or other of the inputs is 1 but not both.

|  |  |  |
| --- | --- | --- |
| Input A | Input B | Output |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

This is written as 

### Exercise 1

Given that an output Q is generated from possibly four inputs A, B, C and D, write out the expression using OR, AND and NOT and then draw the logic diagram for the following.

1. Q = A.B + C.D
2. Q = 
3. Q = 
4. Q = 
5. Q = 
6. Q = 

### De Morgan’s Law

De Morgan’s laws enable Boolean expressions to be converted to forms requiring only the OR and Not functions or only the AND and NOT functions. This means that any Boolean expression may be implemented using only OR and Not gates (NOR gates) or using only AND and NOT gates (NAND gates). Therefore NOR gates alone or NAND gates alone can be used to implement any Boolean function. The fabrication on a single chip of many NOR gates or many NAND gates is possible with integrated circuit technology.









### Exercise 2

Prove these two laws by constructing truth tables.

not (P and Q) = (not P) or (not Q)

not (P or Q) = (not P) and (not Q)

### Other Useful Identities

These identities can be proved by construction a truth table or can be demonstrated using a circuit diagram.



















### Exercise 3

Select four identities to prove by drawing the circuit diagram and four identities to prove by constructing a truth table.

### Example 1



This means **A or Not A = True**

A

Q

Not (A)

So this means that if A is open Not A will be closed and vice-versa. Hence the current will always flow.

### Example 2

0 + A = A

This means **FALSE or A = A**

|  |  |  |
| --- | --- | --- |
| FALSE | A | 0 or A |
| 0 | 1 | 1 |
| 0 | 0 | 0 |

### Exercise 4

1. Consider the two identities below. Draw the circuit diagram for each and then construct the truth tables to prove they are equivalent.  
   1. (A.B) + A
   2. (A + B).A
2. Draw the circuit diagram to represent each of the following expressions.  
   1. P or Q or R
   2. A and (B or C)
   3. (L or M) and (L or P)
3. Given the Boolean expression 
   1. Draw a logic diagram for the expression.
   2. Simplify the expression using laws and theorems of Boolean algebra
   3. Draw the logic diagram of the simplified expression.
4. Given the Boolean expression 
   1. Draw a logic diagram for the expression.
   2. Simplify the expression using laws and theorems of Boolean algebra
   3. Draw the logic diagram of the simplified expression