http://www.electronics-tutorials.ws

Digital Logic Gates **Tutorial: 1 of 10**

**Introduction to Digital Logic Gates**

A **Digital Logic Gate** is an electronic device that makes logical decisions based on the different combinations of digital signals present on its inputs. A digital logic gate may have more than one input but only has one digital output. Standard commercially available digital logic gates are available in two basic families or forms, **TTL** which stands for *Transistor-Transistor Logic* such as the 7400 series, and **CMOS** which stands for *Complementary Metal-Oxide-Silicon* which is the 4000 series of chips. This notation of TTL or CMOS refers to the logic technology used to manufacture the integrated circuit, (IC) or a "chip" as it is more commonly called.



**Digital Logic Gate**

Generally speaking, **TTL** IC's use NPN (or PNP) type[**Bipolar Junction Transistors**](http://www.electronics-tutorials.ws/transistor/tran_1.html)while **CMOS** IC's use [**Field Effect Transistors**](http://www.electronics-tutorials.ws/transistor/tran_1.html)or FET's for both their input and output circuitry. As well as TTL and CMOS technology, simple digital logic gates can also be made by connecting together diodes, transistors and resistors to produce **RTL**, Resistor-Transistor logic gates, **DTL**, Diode-Transistor logic gates or **ECL**, Emitter-Coupled logic gates but these are less common now compared to the popular **CMOS** family.

**Integrated Circuits** or IC's as they are more commonly called, can be grouped together into families according to the number of transistors or "gates" that they contain. For example, a simple AND gate my contain only a few individual transistors, were as a more complex microprocessor may contain many thousands of individual transistor gates. Integrated circuits are categorised according to the number of logic gates or the complexity of the circuits within a single chip with the general classification for the number of individual gates given as:

**Classification of Integrated Circuits**

* Small Scale Integration or (SSI) - Contain up to 10 transistors or a few gates within a single package such as AND, OR, NOT gates.
* Medium Scale Integration or (MSI) - between 10 and 100 transistors or tens of gates within a single package and perform digital operations such as adders, decoders, counters, flip-flops and multiplexers.
* Large Scale Integration or (LSI) - between 100 and 1,000 transistors or hundreds of gates and perform specific digital operations such as I/O chips, memory, arithmetic and logic units.
* Very-Large Scale Integration or (VLSI) - between 1,000 and 10,000 transistors or thousands of gates and perform computational operations such as processors, large memory arrays and programmable logic devices.
* Super-Large Scale Integration or (SLSI) - between 10,000 and 100,000 transistors within a single package and perform computational operations such as microprocessor chips, micro-controllers, basic PICs and calculators.
* Ultra-Large Scale Integration or (ULSI) - more than 1 million transistors - the big boys that are used in computers CPUs, GPUs, video processors, micro-controllers, FPGAs and complex PICs.

While the "ultra large scale" ULSI classification is less well used, another level of integration which represents the complexity of the Integrated Circuit is known as the **System-on-Chip** or (**SOC**) for short. Here the individual components such as the microprocessor, memory, peripherals, I/O logic etc, are all produced on a single piece of silicon and which represents a whole electronic system within one single chip, literally putting the word "integrated" into integrated circuit. These chips are generally used in mobile phones, digital cameras, micro-controllers, PICs and robotic applications, and which can contain up to 100 million individual silicon-CMOS transistor gates within one single package.

**Moore's Law**

In 1965, Gordon Moore co-founder of the Intel corporation predicted that "*The number of transistors and resistors on a single chip will double every 18 months*" regarding the development of semiconductor gate technology. When Moore made his famous comment way back in 1965 there were approximately only 60 individual transistor gates on a single silicon chip or die. Today, the Intel Corporation have placed around **2.0 Billion** individual transistor gates onto its new Quad-core **Itanium** 64-bit microprocessor chip and the count is still rising!.

**Digital Logic States**

The **Digital Logic Gate** is the basic building block from which all digital electronic circuits and microprocessor based systems are constructed from. Basic digital logic gates perform logical operations of AND, OR and NOT on binary numbers. In digital logic design only two voltage levels or states are allowed and these states are generally referred to as Logic "1" and Logic "0", High and Low, True and False and which are represented in [**Boolean Algebra**](http://www.electronics-tutorials.ws/boolean/bool_1.html) and **Truth Tables** by the binary digits of "1" and "0" respectively. A good example of a digital signal is a simple light as it is either "ON" or "OFF" but not both at the same time.

Most *digital logic gates* and logic systems use "Positive logic", in which a logic level "0" or "LOW" is represented by a zero voltage, 0v or ground and a logic level "1" or "HIGH" is represented by a higher voltage such as +5 volts, with the switching from one voltage level to the other, from either a logic level "0" to a "1" or a "1" to a "0" being made as quickly as possible to prevent any faulty operation of the logic circuit. There also exists a complementary "Negative Logic" system in which the values and the rules of a logic "0" and a logic "1" are reversed but in this tutorial section about digital logic gates we shall only refer to the positive logic convention as it is the most commonly used.

In standard TTL (transistor-transistor logic) IC's there is a pre-defined voltage range for the input and output voltage levels which define exactly what is a logic "1" level and what is a logic "0" level and these are shown below.

**TTL Input & Output Voltage Levels**

|  |
| --- |
| TTL Logic Levels |

There are a large variety of logic gate types in both the bipolar 7400 and the CMOS 4000 families of digital logic gates such as 74Lxx, 74LSxx, 74ALSxx, 74HCxx, 74HCTxx, 74ACTxx etc, with each one having its own distinct advantages and disadvantages compared to the other. The exact switching voltage required to produce either a logic "0" or a logic "1" depends upon the specific logic group or family. However, when using a standard +5 volt supply any TTL voltage input between 2.0v and 5v is considered to be a logic "1" or "HIGH" while any voltage input below 0.8v is recognised as a logic "0" or "LOW". The voltage region in between these two voltage levels either as an input or as an output is called the*Indeterminate Region* and operating within this region may cause the logic gate to produce a false output. The CMOS 4000 logic family uses a different level of voltages compared to the TTL types with a logic "1" level operating between 3.0 and 18 volts and a logic "0" level below 1.5 volts.

Then from the above observations, we can define the ideal **Digital Logic Gate** as one that has a "LOW" level logic "0" of 0 volts (ground) and a "HIGH" level logic "1" of +5 volts and this can be demonstrated as:

**Ideal Digital Logic Voltage Levels**

|  |
| --- |
| Ideal Digital Logic Voltage Levels |

Where the opening or closing of the switch produces either a logic level "1" or a logic level "0" with the resistor R being known as a "pull-up" resistor.

**Simple Basic Digital Logic Gates**

Simple digital logic gates can be made by combining transistors, diodes and resistors with a simple example of a Diode-Resistor Logic (DRL) AND gate and a Diode-Transistor Logic (DTL) NANDgate given below.

|  |  |
| --- | --- |
| Diode-Resistor circuit | Diode-Transistor circuit |
| Diode Resistor Logic2-input AND gate | Diode Transistor Logic2-input NAND gate |

The simple 2-input Diode-Resistor AND gate can be converted into a NAND gate by the addition of a single transistor inverting (NOT) stage. Using discrete components such as diodes, resistors and transistors to make digital logic gate circuits are not used in practical commercially available logic IC's as these circuits suffer from propagation delay or gate delay and power loss due to the pull-up resistors, also there is no "Fan-out" facility which is the ability of a single output to drive many inputs of the next stages. Also this type of design does not turn fully "OFF" as a Logic "0" produces an output voltage of 0.6v (diode voltage drop), so the following TTL and CMOS circuit designs are used instead.

**Basic TTL Logic Gates**

The simple Diode-Resistor AND gate above uses separate diodes for its inputs, one for each input. As a transistor is made up off two diode circuits connected together representing an NPN or a PNP device, the input diodes of the DTL circuit can be replaced by one single NPN transistor with multiple emitter inputs as shown.



**2-input NAND gate**

As the gate contains a single stage inverting NPN transistor circuit (TR2) an output logic level "1" at Q is only present when both the emitters of TR1 are connected to logic level "0" or ground allowing base current to pass through the PN junctions of the emitter and not the collector. The multiple emitters of TR1 are connected as inputs thus producing a NAND gate function.

In standard TTL logic gates, the transistors operate either completely in the "cut off" region, or else completely in the saturated region, [**Transistor as a Switch**](http://www.electronics-tutorials.ws/transistor/tran_4.html) type operation.

**Emitter-Coupled Digital Logic Gate**

**Emitter Coupled Logic** or **ECL** is another type of digital logic gate that uses bipolar transistor logic where the transistors are not operated in the saturation region, as they are with the standard TTL digital logic gate. Instead the input and output circuits are push-pull connected transistors with the supply voltage negative with respect to ground. This has the effect of increasing the speed of operation of the ECL gates up to the Gigahertz range compared with the standard TTL types, but noise has a greater effect in ECL logic, because the unsaturated transistors operate within their active region and amplify as well as switch signals.

**The "74" Sub-families of Integrated Circuits**

With improvements in the circuit design to take account of propagation delays, current consumption, fan-in and fan-out requirements etc, this type of TTL bipolar transistor technology forms the basis of the prefixed "74" family of digital logic IC's, such as the "7400" Quad 2-input AND gate, or the "7402" Quad 2-input OR gate. Sub-families of the 74xx series IC's are available relating to the different technologies used to fabricate the gates and they are denoted by the letters in between the 74 designation and the device number. There are a number of TTL sub-families available that provide a wide range of switching speeds and power consumption such as the 74**L**00 or 74**ALS**00 AND gate, were the "L" stands for "Low-power TTL" and the "ALS" stands for "Advanced Low-power Schottky TTL" and these are listed below.

* 74xx or 74Nxx: Standard TTL - These devices are the original TTL family of logic gates introduced in the early 70's. They have a propagation delay of about 10ns and a power consumption of about 10mW.
* 74Lxx: Low Power TTL - Power consumption was improved over standard types by increasing the number of internal resistances but at the cost of a reduction in switching speed.
* 74Hxx: High Speed TTL - Switching speed was improved by reducing the number of internal resistances. This also increased the power consumption.
* 74Sxx: Schottky TTL - Schottky technology is used to improve input impedance, switching speed and power consumption (2mW) compared to the 74Lxx and 74Hxx types.
* 74LSxx: Low Power Schottky TTL - Same as 74Sxx types but with increased internal resistances to improve power consumption.
* 74ASxx: Advanced Schottky TTL - Improved design over 74Sxx Schottky types optimised to increase switching speed at the expense of power consumption of about 22mW.
* 74ALSxx: Advanced Low Power Schottky TTL - Lower power consumption of about 1mW and higher switching speed of about 4nS compared to 74LSxx types.
* 74HCxx: High Speed CMOS - CMOS technology and transistors to reduce power consumption of less than 1uA with CMOS compatible inputs.
* 74HCTxx: High Speed CMOS - CMOS technology and transistors to reduce power consumption of less than 1uA but has increased propagation delay of about 16nS due to the TTL compatible inputs.

**Basic CMOS Digital Logic Gate**

One of the main disadvantages of the TTL logic series is that the gates are based on bipolar transistor logic technology and as transistors are current operated devices, they consume large amounts of power from a fixed +5 volt power supply. Also, TTL bipolar transistor gates have a limited operating speed when switching from an "OFF" state to an "ON" state and vice-versa called the "gate" or "propagation delay". To overcome these limitations complementary MOS called "CMOS" logic gates using "Field Effect Transistors" or FET's were developed.

As these gates use both P-channel and N-channel MOSFET's as their input device, at quiescent conditions with no switching, the power consumption of CMOS gates is almost zero, (1 to 2uA) making them ideal for use in low-power battery circuits and with switching speeds upwards of 100MHz for use in high frequency timing and computer circuits.



**2-input NAND gate**

This CMOS gate example contains 3 N-channel MOSFET's, one for each input FET1 and FET2and one for the output FET3. When both the inputs A and B are at logic level "0", FET1 and FET2 are both switched "OFF" giving an output logic "1" from the source of FET3. When one or both of the inputs are at logic level "1" current flows through the corresponding FET giving an output state at Q equivalent to logic "0", thus producing a NAND gate function.

Improvements in the circuit design with regards to switching speed, low power consumption and improved propagation delays has resulted in the standard CMOS 4000 "CD" family of logic IC's being developed that complement the TTL range. As with the standard TTL digital logic gates, all the major digital logic gates and devices are available in the CMOS package such as the CD4011, a Quad 2-input NAND gate, or the CD4001, a Quad 2-inputNOR gate along with all their sub-families.

Like TTL logic, complementary MOS (CMOS) circuits take advantage of the fact that both N-channel and P-channel devices can be fabricated on the same substrate and connected together to form logic functions. One main disadvantage with the CMOS range of IC's compared to their equivalent TTL types is that they are easily damaged by static electricity so extra care must be taken when handling these devices. Also unlike TTL logic gates that operate on single +5V voltages for both their input and output levels, CMOS digital logic gates operate on a single supply voltage of between +3 and +18 volts.

In the next tutorial about **Digital Logic Gates**, we will look at the digital Logic [**AND Gate**](http://www.electronics-tutorials.ws/logic/logic_2.html) function as used in both TTL and CMOS logic circuits as well as its Boolean Algebra definition and truth tables.

Logic "AND" Gate **Tutorial: 2 of 10**

**The Logic "AND" Gate**

**Definition**

A **Logic AND Gate** is a type of digital logic gate that has an output which is normally at logic level "0" and only goes "HIGH" to a logic level "1" when **ALL** of its inputs are at logic level "1". The output of a **Logic AND Gate** only returns "LOW" again when **ANY** of its inputs are at a logic level "0". The logic or Boolean expression given for a logic AND gate is that for *Logical Multiplication* which is denoted by a single dot or full stop symbol, (.) giving us the Boolean expression of: A.B = Q.

Then we can define the operation of a 2-input logic AND gate as being:

**"If both A and B are true, then Q is true"**

**2-input Transistor AND Gate**

A simple 2-input logic AND gate can be constructed using RTL Resistor-transistor switches connected together as shown below with the inputs connected directly to the transistor bases. Both transistors must be saturated "ON" for an output at Q.

|  |
| --- |
| 2-input Transistor AND Gate |

**Logic AND Gates** are available using digital circuits to produce the desired logical function and is given a symbol whose shape represents the logical operation of the AND gate.

**The Digital Logic "AND" Gate**

**2-input AND Gate**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| 2-input AND gate**2-input AND Gate** | B | A | Q |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| Boolean Expression **Q = A.B** | Read as A **AND** B gives Q |

**3-input AND Gate**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| 3-input AND gate**3-input AND Gate** | C | B | A | Q |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |
| Boolean Expression **Q = A.B.C** | Read as A **AND** B **AND** C gives Q |

Because the Boolean expression for the logic AND function is defined as (.), which is a binary operation, AND gates can be cascaded together to form any number of individual inputs. However, commercial available AND gate IC's are only available in standard 2, 3, or 4-input packages. If additional inputs are required, then standard AND gates will need to be cascaded together to obtain the required input value, for example.

|  |
| --- |
| 6-input AND Gate |

The Boolean Expression for this 6-input AND gate will therefore be: **Q = (A.B).(C.D).(E.F)**

If the number of inputs required is an odd number of inputs any "unused" inputs can be held HIGH by connecting them directly to the power supply using suitable "Pull-up" resistors.

Commonly available digital logic AND gate IC's include:

|  |  |
| --- | --- |
| TTL Logic Types* 74LS08 Quad 2-input
* 74LS11 Triple 3-input
* 74LS21 Dual 4-input
 | CMOS Logic Types* CD4081 Quad 2-input
* CD4073 Triple 3-input
* CD4082 Dual 4-input
 |

**Quad 2-input AND Gate 7408**

|  |
| --- |
| 2-input AND gate 7408 |

In the next tutorial about **Digital Logic Gates**, we will look at the digital logic [**OR Gate**](http://www.electronics-tutorials.ws/logic/logic_3.html) function as used in both TTL and CMOS logic circuits as well as its Boolean Algebra definition and truth tables.

Logic "OR" Gates **Tutorial: 3 of 10**

**The Logic "OR" Gate**

**Definition**

A **Logic OR Gate** or Inclusive-OR gate is a type of digital logic gate that has an output which is normally at logic level "0" and only goes "HIGH" to a logic level "1" when **ANY** of its inputs are at logic level "1". The output of a **Logic OR Gate** only returns "LOW" again when **ALL** of its inputs are at a logic level "0". The logic or Boolean expression given for a logic OR gate is that for *Logical Addition* which is denoted by a plus sign, (+) giving us the Boolean expression of: A+B = Q.

Then we can define the operation of a 2-input logic OR gate as being:

**"If either A or B is true, then Q is true"**

**2-input Transistor OR Gate**

A simple 2-input logic OR gate can be constructed using RTL Resistor-transistor switches connected together as shown below with the inputs connected directly to the transistor bases. Either transistor must be saturated "ON" for an output at Q.

|  |
| --- |
| 2-input Transistor OR Gate |

**Logic OR Gates** are available using digital circuits to produce the desired logical function and is given a symbol whose shape represents the logical operation of the OR gate.

**The Digital Logic "OR" Gate**

**2-input OR Gate**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| 2-input OR gate**2-input OR Gate** | B | A | Q |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |
| Boolean Expression **Q = A+B** | Read as A **OR** B gives Q |

**3-input OR Gate**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| 3-input OR gate**3-input OR Gate** | C | B | A | Q |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |
| Boolean Expression **Q = A+B+C** | Read as A **OR** B **OR** C gives Q |

The OR function can have any number of individual inputs. However, commercial available OR gates are available in 2, 3, or 4 inputs types. Additional inputs will require gates to be cascaded together for example.

|  |
| --- |
| 6-input OR Gate |

The Boolean Expression for this 6-input OR gate will therefore be: **Q = (A+B)+(C+D)+(E+F)**

If the number of inputs required is an odd number of inputs any "unused" inputs can be held LOW by connecting them directly to ground using suitable "Pull-down" resistors.

Commonly available OR gate IC's include:

|  |  |
| --- | --- |
| TTL Logic Types* 74LS32 Quad 2-input
 | CMOS Logic Types* CD4071 Quad 2-input
* CD4075 Triple 3-input
* CD4072 Dual 4-input
 |

**Quad 2-input OR Gate 7432**

|  |
| --- |
| 2-input OR gate 7432 |

In the next tutorial about **Digital Logic Gates**, we will look at the digital logic [**NOT Gate**](http://www.electronics-tutorials.ws/logic/logic_4.html) function as used in both TTL and CMOS logic circuits as well as its Boolean Algebra definition and truth table.

Logic "NOT" Gates **Tutorial: 4 of 10**

**The Digital Logic "NOT" Gate**

**Definition**

The digital **Logic NOT Gate** is the most basic of all the logical gates and is sometimes referred to as an **Inverting Buffer** or simply a **Digital Inverter**. It is a single input device which has an output level that is normally at logic level "1" and goes "LOW" to a logic level "0" when its single input is at logic level "1", in other words it "inverts" (complements) its input signal. The output from a NOT gate only returns "HIGH" again when its input is at logic level "0" giving us the Boolean expression of: A = Q.

Then we can define the operation of a single input logic NOT gate as being:

**"If A is NOT true, then Q is true"**

**Transistor NOT Gate**

A simple 2-input logic NOT gate can be constructed using a RTL Resistor-transistor switches as shown below with the input connected directly to the transistor base. The transistor must be saturated "ON" for an inversed output "OFF" at Q.

|  |
| --- |
| Transistor NOT Gate |

**Logic NOT Gates** are available using digital circuits to produce the desired logical function. The standard NOT gate is given a symbol whose shape is of a triangle pointing to the right with a circle at its end. This circle is known as an "inversion bubble" and is used in NOT, NAND and NOR symbols at their output to represent the logical operation of the NOT function. This bubble denotes a signal inversion (complementation) of the signal and can be present on either or both the output and/or the input terminals.

**The Digital Inverter or NOT gate**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| The NOT gate**Inverter or NOT Gate** | A | Q |
| 0 | 1 |
| 1 | 0 |
| Boolean Expression **Q = not A or A** | Read as inverse of **A** gives Q |

Logic NOT gates provide the complement of their input signal and are so called because when their input signal is "HIGH" their output state will **NOT** be "HIGH". Likewise, when their input signal is "LOW" their output state will **NOT** be "LOW". As they are single input devices, logic NOT gates are not normally classed as "decision" making devices or even as a gate, such as the AND or OR gates which have two or more logic inputs. Commercial available NOT gates IC's are available in either 4 or 6 individual gates within a single i.c. package.

The "bubble" (o) present at the end of the NOT gate symbol above denotes a signal inversion (complimentation) of the output signal. But this bubble can also be present at the gates input to indicate an *active-LOW* input. This inversion of the input signal is not restricted to the NOT gate only but can be used on any digital circuit or gate as shown with the operation of inversion being exactly the same whether on the input or output terminal. The easiest way is to think of the bubble as simply an inverter.

**Signal Inversion using Active-low input Bubble**

|  |
| --- |
| Signal Inversion NOT gate**Bubble Notation for Input Inversion** |

**NAND and NOR Gate Equivalents**

An **Inverter** or logic NOT gate can also be made using standard NAND and NOR gates by connecting together **ALL** their inputs to a common input signal for example.

|  |
| --- |
| Inverter using NAND and NOR Gates |
| Single Stage Transistor Inverter | Also a very simple inverter can also be made using just a single stage transistor switching circuit as shown. When the transistors base input at "A" is high, the transistor conducts and collector current flows producing a voltage drop across the resistor R thereby connecting the output point at "Q" to ground thus resulting in a zero voltage output at "Q". When the transistors base input at "A" is low, the transistor now switches "OFF" and no collector current flows through the resistor resulting in an output voltage at "Q" high at a value near to +Vcc. |

Then, with an input voltage at "A" HIGH, the output at "Q" will be LOW and an input voltage at "A" LOW the resulting output voltage at "Q" is HIGH producing the complement of the input signal.

**Hex Schmitt Inverters**

A standard **Inverter** or **Logic NOT Gate**, is usually made up from transistor switching circuits that do not switch from one state to the next instantly, there is some delay. Also as a transistor is a basic current amplifier, it can also operate in a linear mode and any small variation to its input level will cause a variation to its output level or may even switch "ON" and "OFF" several times if there is any noise present in the circuit. One way to overcome these problems is to use a **Schmitt Inverter** or **Hex Inverter**.

We know from the previous pages that all digital gates use only two logic voltage states and that these are generally referred to as **Logic "1"** and **Logic "0"** any TTL voltage input between 2.0v and 5v is recognised as a logic "1" and any voltage input below 0.8v is recognised as a logic "0" respectively. A **Schmitt Inverter** is designed to operate or switch state when its input signal goes above an "Upper Threshold Voltage" limit in which case the output changes and goes "LOW", and will remain in that state until the input signal falls below the "Lower Threshold Voltage" level in which case the output signal goes "HIGH". In other words a Schmitt Inverter has some form of **Hysteresis** built into its switching circuit. This switching action between an upper and lower threshold limit provides a much cleaner and faster "ON/OFF" switching output signal and makes the Schmitt inverter ideal for switching any slow-rising or slow-falling input signal either an analogue or digital signal.

**Schmitt Inverter**

|  |
| --- |
| Schmitt Inverter |

A very useful application of Schmitt inverters is when they are used as oscillators or sine-to-square wave converters for use as square wave clock signals.

**Schmitt Inverter Oscillator & Converter**

|  |
| --- |
| Schmitt Inverter Oscillator and Square Wave Generator |

The first circuit shows a very simple low power RC type oscillator using a Schmitt inverter to generate square waves. Initially the capacitor C is fully discharged so the input to the inverter is "LOW" resulting in an inverted output which is "HIGH". As the output from the inverter is fed back to its input and the capacitor via the resistor R the capacitor begins to charge up. When the capacitors charging voltage reaches the upper threshold limit of the inverter, the inverter changes state, the output becomes "LOW" and the capacitor begins to discharge through the resistor until it reaches the lower threshold level were the inverter changes state again. This switching back and forth by the inverter produces a square wave output signal with a 33% duty cycle and whose frequency is given as: ƒ = 680/RC.

The second circuit converts a sine wave input (or any oscillating input for that matter) into a square wave output. The input to the inverter is connected to the junction of the potential divider network which is used to set the quiescent point of the circuit. The input capacitor blocks any DC component present in the input signal only allowing the sine wave signal to pass. As this signal passes the upper and lower threshold points of the inverter the output also changes from "HIGH" to "LOW" and so on producing a square wave output waveform. This circuit produces an output pulse on the positive rising edge of the input waveform, but by connecting a second Schmitt inverter to the output of the first, the basic circuit can be modified to produce an output pulse on the negative falling edge of the input signal.

Commonly available logic NOT gate and Inverter IC's include

|  |  |
| --- | --- |
| TTL Logic Types* 74LS04 Hex Inverting NOT Gate
* 74LS04 Hex Inverting NOT Gate
* 74LS14 Hex Schmitt Inverting NOT Gate
* 74LS1004 Hex Inverting Drivers
 | CMOS Logic Types* CD4009 Hex Inverting NOT Gate
* CD4069 Hex Inverting NOT Gate
 |

**Inverter or NOT Gate 7404**

|  |
| --- |
| Inverter NOT gate 7404 |

In the next tutorial about **Digital Logic Gates**, we will look at the digital logic [**NAND Gate**](http://www.electronics-tutorials.ws/logic/logic_5.html) function as used in both TTL and CMOS logic circuits as well as its Boolean Algebra definition and truth tables.

Logic "NAND" Gate **Tutorial: 5 of 10**

**The Logic "NAND" Gate**

**Definition**

The **Logic NAND Gate** is a combination of the digital logic AND gate with that of an inverter or NOT gate connected together in series. The NAND (Not - AND) gate has an output that is normally at logic level "1" and only goes "LOW" to logic level "0" when **ALL** of its inputs are at logic level "1". The **Logic NAND Gate** is the reverse or "*Complementary*" form of the AND gate we have seen previously.

**Logic NAND Gate Equivalence**

|  |
| --- |
| 2-input NAND Gate |

The logic or Boolean expression given for a logic NAND gate is that for *Logical Addition*, which is the opposite to the AND gate, and which it performs on the *complements* of the inputs. The Boolean expression for a logic NAND gate is denoted by a single dot or full stop symbol, (.) with a line or *Overline*, ( ‾‾ ) over the expression to signify the NOT or logical negation of the NAND gate giving us the Boolean expression of: A.B = Q.

Then we can define the operation of a 2-input logic NAND gate as being:

**"If either A or B are NOT true, then Q is true"**

**Transistor NAND Gate**

A simple 2-input logic NAND gate can be constructed using RTL Resistor-transistor switches connected together as shown below with the inputs connected directly to the transistor bases. Either transistor must be cut-off "OFF" for an output at Q.

|  |
| --- |
| 2-input Transistor NAND Gate |

**Logic NAND Gates** are available using digital circuits to produce the desired logical function and is given a symbol whose shape is that of a standard AND gate with a circle, sometimes called an "inversion bubble" at its output to represent the NOT gate symbol with the logical operation of the NAND gate given as.

**The Digital Logic "NAND" Gate**

**2-input NAND Gate**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| 2-input NAND gate**2-input NAND Gate** | B | A | Q |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |
| Boolean Expression **Q = A.B** | Read as A **AND** B gives **NOT** Q |

**3-input NAND Gate**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| 3-input NAND gate**3-input NAND Gate** | C | B | A | Q |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |
| Boolean Expression **Q = A.B.C** | Read as A **AND** B **AND** C gives **NOT** Q |

As with the AND function seen previously, the NAND function can also have any number of individual inputs and commercial available NAND Gate i.c.´s are available in standard 2, 3, or 4 input types. If additional inputs are required, then the standard NAND gates can be cascaded together to provide more inputs for example.

**A 4-input NAND Function**

|  |
| --- |
| 4-input NAND Gate |

The Boolean Expression for this 4-input logic NAND gate will therefore be: **Q = A.B.C.D**

If the number of inputs required is an odd number of inputs any "unused" inputs can be held HIGH by connecting them directly to the power supply using suitable "Pull-up" resistors.

The **Logic NAND Gate** function is sometimes known as the **Sheffer Stroke Function**and is denoted by a vertical bar or upwards arrow operator, for example, A NAND B = A|B or A↑B.

**The "Universal" NAND Gate**

The **Logic NAND Gate** is generally classed as a "Universal" gate because it is one of the most commonly used logic gate types. NAND gates can also be used to produce any other type of logic gate function, and in practice the NAND gate forms the basis of most practical logic circuits. By connecting them together in various combinations the three basic gate types of AND, OR and NOT function can be formed using only NAND's, for example.

**Various Logic Gates using only NAND Gates**

|  |
| --- |
| Logic Gates using NAND Gates |

As well as the three common types above, Ex-Or, Ex-Nor and standard NOR gates can be formed using just individual NAND gates.

Commonly available logic NAND gate IC's include:

|  |  |
| --- | --- |
| TTL Logic Types* 74LS00 Quad 2-input
* 74LS10 Triple 3-input
* 74LS20 Dual 4-input
* 74LS30 Single 8-input
 | CMOS Logic Types* CD4011 Quad 2-input
* CD4023 Triple 3-input
* CD4012 Dual 4-input
 |

**Quad 2-input NAND Gate 7400**

|  |
| --- |
| 2-input NAND gate 7400 |

In the next tutorial about **Digital Logic Gates**, we will look at the digital logic [**NOR Gate**](http://www.electronics-tutorials.ws/logic/logic_6.html) function as used in both TTL and CMOS logic circuits as well as its Boolean Algebra definition and truth tables.

Logic "NOR" Gate **Tutorial: 6 of 10**

**The Logic "NOR" Gate**

**Definition**

The **Logic NOR Gate** or Inclusive-NOR gate is a combination of the digital logic OR gate with that of an inverter or NOT gate connected together in series. The NOR (Not - OR) gate has an output that is normally at logic level "1" and only goes "LOW" to logic level "0" when **ANY** of its inputs are at logic level "1". The **Logic NOR Gate** is the reverse or "*Complementary*" form of the OR gate we have seen previously.

**NOR Gate Equivalent**

|  |
| --- |
| 2-input NOR Gate |

The logic or Boolean expression given for a logic NOR gate is that for *Logical Multiplication* which it performs on the *complements* of the inputs. The Boolean expression for a logic NOR gate is denoted by a plus sign, (+) with a line or *Overline*, ( ‾‾ ) over the expression to signify the NOT or logical negation of the NOR gate giving us the Boolean expression of: A+B = Q.

Then we can define the operation of a 2-input logic NOR gate as being:

**"If both A and B are NOT true, then Q is true"**

**Transistor NOR Gate**

A simple 2-input logic NOR gate can be constructed using RTL Resistor-transistor switches connected together as shown below with the inputs connected directly to the transistor bases. Both transistors must be cut-off "OFF" for an output at Q.

|  |
| --- |
| 2-input Transistor NOR Gate |

**Logic NOR Gates** are available using digital circuits to produce the desired logical function and is given a symbol whose shape is that of a standard OR gate with a circle, sometimes called an "inversion bubble" at its output to represent the NOT gate symbol with the logical operation of the NOR gate given as.

**The Digital Logic "NOR" Gate**

**2-input NOR Gate**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| 2-input NOR gate**2-input NOR Gate** | B | A | Q |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |
| Boolean Expression **Q = A+B** | Read as A **OR** B gives **NOT** Q |

**3-input NOR Gate**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| 3-input NOR gate**3-input NOR Gate** | C | B | A | Q |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |
| Boolean Expression **Q = A+B+C** | Read as A **OR** B **OR** C gives **NOT** Q |

As with the OR function, the NOR function can also have any number of individual inputs and commercial available NOR Gate IC's are available in standard 2, 3, or 4 input types. If additional inputs are required, then the standardNOR gates can be cascaded together to provide more inputs for example.

**A 4-input NOR Function**

|  |
| --- |
| 4-input NOR Gate |

The Boolean Expression for this 4-input NOR gate will therefore be: **Q = A+B+C+D**

If the number of inputs required is an odd number of inputs any "unused" inputs can be held LOW by connecting them directly to ground using suitable "Pull-down" resistors.

The **Logic NOR Gate** function is sometimes known as the **Pierce Function** and is denoted by a downwards arrow operator as shown, A↓B.

**The "Universal" NOR Gate**

Like the NAND gate seen in the last section, the NOR gate can also be classed as a "Universal" type gate. NOR gates can be used to produce any other type of logic gate function just like the NAND gate and by connecting them together in various combinations the three basic gate types of AND, OR and NOT function can be formed using only NOR's, for example.

**Various Logic Gates using only NOR Gates**

|  |
| --- |
| Logic Gates using NOR Gates |

As well as the three common types above, Ex-Or, Ex-Nor and standard NOR gates can also be formed using just individual NOR gates.

Commonly available NOR gate IC's include:

|  |  |
| --- | --- |
| TTL Logic Types* 74LS02 Quad 2-input
* 74LS27 Triple 3-input
* 74LS260 Dual 4-input
 | CMOS Logic Types* CD4001 Quad 2-input
* CD4025 Triple 3-input
* CD4002 Dual 4-input
 |

**Quad 2-input NOR Gate 7402**

|  |
| --- |
| 2-input NOR gate 7402 |

In the next tutorial about **Digital Logic Gates**, we will look at the digital logic Exclusive-OR gate known commonly as the [**Ex-OR Gate**](http://www.electronics-tutorials.ws/logic/logic_7.html) function as used in both TTL and CMOS logic circuits as well as its Boolean Algebra definition and truth tables.

Exclusive-OR Gate **Tutorial: 7 of 10**

**The Exclusive-OR Gate**

**Definition**

Previously, we have seen that for a 2-input OR gate, if A = "1", **OR** B = "1", **OR BOTH** A + B = "1" then the output from the gate is also at logic level "1" and this is known as an Inclusive-OR function because it *includes* the case of Q = "1" when both A and B = "1". If however, an output "1" is obtained **ONLY** when A = "1" or when B = "1" but **NOT** both together at the same time, then this type of gate is known as an Exclusive-OR function or an Ex-Or function for short because it *excludes* the "**OR BOTH**" case of Q = "1" when both A and B = "1".

In other words the output of an Exclusive-OR gate **ONLY** goes "HIGH" when its two input terminals are at "**DIFFERENT**" logic levels with respect to each other and they can both be at logic level "1" or both at logic level "0" giving us the Boolean expression of: Q = (A B) = A.B + A.B

The **Exclusive-OR Gate** function is achieved is achieved by combining standard gates together to form more complex gate functions. An example of a 2-input Exclusive-OR gate is given below.

**The Digital Logic "Ex-OR" Gate**

**2-input Ex-OR Gate**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| 2-input Ex-OR Logic Gate**2-input Ex-OR Gate** | B | A | Q |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |
| Boolean Expression Q = A B | Read as A **OR** B but NOT **BOTH** gives Q |

Then, the logic function implemented by a 2-input Ex-OR is given as "either A OR B but NOT both" will give an output at Q. In general, an Ex-OR gate will give an output value of logic "1" ONLY when there are an **ODD** number of 1's on the inputs to the gate. Then an Ex-OR function with more than two inputs is called an "odd function" or modulo-2-sum (Mod-2-SUM), not an Ex-OR. This description can be expanded to apply to any number of individual inputs as shown below for a 3-input Ex-OR gate.

**3-input Ex-OR Gate**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| 3-input EX-OR gate**3-input Ex-OR Gate** | C | B | A | Q |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |
| Boolean Expression Q = A B C | Read as "any **ODD** number of Inputs" gives Q |

The symbol used to denote an Exclusive-OR function is slightly different to that for the standard Inclusive-OR gate. The logic or Boolean expression given for a logic OR gate is that of logical addition which is denoted by a standard plus sign. The symbol used to describe the Boolean expression for an **Exclusive-OR** function is a plus sign, ( + ) within a circle, ( Ο ). This exclusive-OR symbol also represents the mathematical "direct sum of sub-objects" expression, with the resulting symbol for an *Exclusive-OR* function being given as: ( ).

We said previously that the Ex-OR function is a combination of different basic logic gates and using the 2-input truth table above, we can expand the Ex-OR function to: Q = (A B) = (A+B).(A.B) which means we can realise this new expression using the following individual gates.

**Ex-OR Gate Equivalent Circuit**

|  |
| --- |
| 2-input EX-Or Gate Equivalent |

One of the main disadvantages of implementing the Ex-OR function above is that it contains three different types logic gates OR, NAND and finally AND within its design. One easier way of producing the Ex-OR function from a single gate is to use our old favourite the NAND gate as shown below.

**Ex-OR Function Realisation using NAND gates**

|  |
| --- |
| 2-input EX-OR Gate Equivalent using NAND Gates |

**Exclusive-OR Gates** are used mainly to build circuits that perform arithmetic operations and calculations especially **Adders** and **Half-Adders** as they can provide a "carry-bit" function or as a controlled inverter, where one input passes the binary data and the other input is supplied with a control signal.

Commonly available **Exclusive-OR** gate IC's include:

|  |  |
| --- | --- |
| TTL Logic Types* 74LS86 Quad 2-input
 | CMOS Logic Types* CD4030 Quad 2-input
 |

**Quad 2-input Ex-OR Gate 7486**

|  |
| --- |
| 2-input Ex-OR gate 7486 |

In the next tutorial about **Digital Logic Gates**, we will look at the digital logic Exclusive-NOR gate known commonly as the [**Ex-NOR Gate**](http://www.electronics-tutorials.ws/logic/logic_8.html) function as used in both TTL and CMOS logic circuits as well as its Boolean Algebra definition and truth tables.

Exclusive-NOR Gate **Tutorial: 8 of 10**

**The Exclusive-NOR Gate**

**Definition**

The **Exclusive-NOR Gate** function or Ex-NOR for short, is a digital logic gate that is the reverse or complementary form of the Exclusive-OR function we look at in the previous section. It is a combination of the Exclusive-OR gate and the NOT gate but has a truth table similar to the standard NOR gate in that it has an output that is normally at logic level "1" and goes "LOW" to logic level "0" when **ANY** of its inputs are at logic level "1". However, an output "1" is also obtained if **BOTH** of its inputs are at logic level "1". For example, A = "1" and B = "1" at the same time giving us the Boolean expression of: Q = (A B) = A.B + A.B

In other words, the output of an Exclusive-NOR gate **ONLY** goes "HIGH" when its two input terminals, A and B are at the "**SAME**" logic level which can be either at a logic level "1" or at a logic level "0". Then this type of gate gives and output "1" when its inputs are "*logically equal*" or "*equivalent*" to each other, which is why an **Exclusive-NOR** gate is sometimes called an **Equivalence Gate**. The logic symbol for an Exclusive-NOR gate is simply an Exclusive-OR gate with a circle or "inversion bubble", ( ο ) at its output to represent the NOT function. Then the **Logic Exclusive-NOR Gate** is the reverse or "*Complementary*" form of the Exclusive-OR gate, ( ) we have seen previously.

**Ex-NOR Gate Equivalent**

|  |
| --- |
| 2-input Ex-NOR Gate |

The **Exclusive-NOR Gate** function is achieved by combining standard gates together to form more complex gate functions and an example of a 2-input Exclusive-NOR gate is given below.

**The Digital Logic "Ex-NOR" Gate**

**2-input Ex-NOR Gate**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| 2-input Ex-NOR Logic Gate**2-input Ex-NOR Gate** | B | A | Q |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| Boolean Expression Q = A B | Read if A **AND** B the **SAME** gives Q |

Then, the logic function implemented by a 2-input Ex-NOR gate is given as "when both A AND B are the SAME" will give an output at Q. In general, an Exclusive-NOR gate will give an output value of logic "1" ONLY when there are an **EVEN** number of 1's on the inputs to the gate (the inverse of the Ex-OR gate) except when all its inputs are "LOW". Then an Ex-NOR function with more than two inputs is called an "even function" or modulo-2-sum (Mod-2-SUM), not an Ex-NOR. This description can be expanded to apply to any number of individual inputs as shown below for a 3-input Exclusive-NOR gate.

**3-input Ex-NOR Gate**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| 3-input EX-Nor gate**3-input Ex-NOR Gate** | C | B | A | Q |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |
| Boolean Expression Q = A B C | Read as "any **EVEN** number of Inputs" gives Q |

We said previously that the Ex-NOR function is a combination of different basic logic gates Ex-OR and a NOT gate, and by using the 2-input truth table above, we can expand the Ex-NOR function to: Q = A B = (A.B) + (A.B) which means we can realise this new expression using the following individual gates.

**Ex-NOR Gate Equivalent Circuit**

|  |
| --- |
| 2-input EX-NOR Gate Equivalent |

One of the main disadvantages of implementing the Ex-NOR function above is that it contains three different types logic gates the AND, NOT and finally an OR gate within its basic design. One easier way of producing the Ex-NOR function from a single gate type is to use NAND gates as shown below.

**Ex-NOR Function Realisation using NAND gates**

|  |
| --- |
| 2-input EX-NOR Gate Equivalent using NAND Gates |

Ex-NOR gates are used mainly in electronic circuits that perform arithmetic operations and data checking such as **Adders**, **Subtractors** or **Parity Checkers**, etc. As the Ex-NORgate gives an output of logic level "1" whenever its two inputs are equal it can be used to compare the magnitude of two binary digits or numbers and so Ex-NOR gates are used in [**Digital Comparator**](http://www.electronics-tutorials.ws/combination/comb_8.html) circuits.

Commonly available **Exclusive-NOR** gate IC's include:

|  |  |
| --- | --- |
| TTL Logic Types* 74LS266 Quad 2-input
 | CMOS Logic Types* CD4077 Quad 2-input
 |

**Quad 2-input Ex-NOR Gate 74266**

|  |
| --- |
| 2-input Ex-NOR gate 74266 |

In the next tutorial about **Digital Logic Gates**, we will look at the digital [**Tri-state Buffer**](http://www.electronics-tutorials.ws/logic/logic_9.html) also called the non-inverting buffer as used in both TTL and CMOS logic circuits as well as its Boolean Algebra definition and truth table.

Tri-state Buffer **Tutorial: 9 of 10**

**The Digital Tri-state Buffer**

**Definition**

In a previous tutorial we look at the digital [**Not Gate**](http://www.electronics-tutorials.ws/logic/logic_4.html) or **Inverter**, and we saw that the NOT gates output is the "complement" or inverse of its input signal. For example, when its input signal is "HIGH" its output state will **NOT** be "HIGH" and when its input signal is "LOW" its output state will **NOT** be "LOW", in other words it inverts the signal. Another single input logical device used a lot in electronic circuits and which is the reverse of the NOT gate inverter is called a **Buffer**, **Digital Buffer** or **Non-inverting Buffer**.

A **Digital Buffer** is another single input device that does no invert or perform any type of logical operation on its input signal as its output exactly matches that of its input signal. In other words, its Output equals its Input. It is a "Non-inverting" device and so will give us the Boolean expression of: Q = A.

Then we can define the operation of a single input Digital Buffer as being:

**"If A is true, then Q is true"**

**The Tri-state Buffer**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| A Tri-state Buffer**A Tri-state Buffer** | A | Q |
| 0 | 0 |
| 1 | 1 |
| Boolean Expression **Q = A** | Read as **A** gives **Q** |

The **Digital Tri-state Buffer** can also be made by connecting together two NOT gates as shown below. The first will "invert" the input signal A and the second will "re-invert" it back to its original level.

**Double Inversion using NOT Gates**

|  |
| --- |
| Tri-state Buffer using NOT gates |

You may think "what is the point of a Digital Buffer", if it does not alter its input signal in any way or make any logical operations like the AND or OR gates, then why not use a piece of wire instead and that's a good point. But a non-inverting digital Buffer has many uses in digital electronic circuits, as they can be used to isolate other gates or circuits from each other or they can be used to drive high current loads such as transistor switches because their output drive capability is much higher than their input signal requirements, in other words buffers are uses for power amplification giving them a high **fan-out** capability.

**Buffer Fan-out Example**

|  |
| --- |
| Buffer Fan-out example |

**Fan-out** is the output driving capability or output current capability of a logic gate giving greater power amplification of the signal. It may be necessary to connect more than just one logic gate to the output of another or to switch a high current load such as an [**LED**](http://www.electronics-tutorials.ws/diode/diode_8.html), then a Buffer will allow us to do just that by having a high fan-out rating of up to 50.

**The "Tri-state Buffer"**

As well as the standard Digital Buffer seen above, there is another type of digital Buffer circuit whose output can be "electronically" disconnected from its output circuitry when required. This type of Buffer is known as a 3-State Buffer or commonly Tri-state Buffer.

A Tri-state Buffer can be thought of as an input controlled switch which has an output that can be electronically turned "ON" or "OFF" by means of an external "Control" or "Enable" signal input. This control signal can be either a logic "0" or a logic "1" type signal resulting in the Tri-state Buffer being in one state allowing its output to operate normally giving either a logic "0" or logic "1" output. But when activated in the other state it disables or turns "OFF" its output producing an open circuit condition that is neither "High" or "low", but instead gives an output state of very high impedance, **high-Z**, or more commonly Hi-Z. Then this type of device has two logic state inputs, "0" or a "1" but can produce three different output states, "0", "1" or "Hi-Z" which is why it is called a "3-state" device.

There are two different types of Tri-state Buffer, one whose output is controlled by an "**Active-HIGH**" control signal and the other which is controlled by an "**Active-LOW**" control signal, as shown below.

**Active "HIGH" Tri-state Buffer**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| A Tri-state Buffer**Tri-state Buffer** | Enable | A | Q |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| 0 | 0 | Hi-Z |
| 0 | 1 | Hi-Z |
| Read as Output = Input if Enable is equal to "1" |

An **Active-high** Tri-state Buffer is activated when a logic level "1" is applied to its "**enable**" control line and the data passes through from its input to its output. When the enable control line is at logic level "0", the buffer output is disabled and a high impedance condition, Hi-Z is present on the output.

**Active "LOW" Tri-state Buffer**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| A Tri-state Buffer**Tri-state Buffer** | Enable | A | Q |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | Hi-Z |
| 1 | 1 | Hi-Z |
| Read as Output = Input if Enable is **NOT** equal to "1" |

An **Active-low** Tri-state Buffer is the opposite to the above, and is activated when a logic level "0" is applied to its "**enable**" control line. The data passes through from its input to its output. When the enable control line is at logic level "1", the buffer output is disabled and a high impedance condition, Hi-Z is present on the output.

**Tri-state Buffer Control**

The Tri-state Buffer is used in many electronic and microprocessor circuits as they allow multiple logic devices to be connected to the same wire or bus without damage or loss of data. For example, suppose we have a data line or data bus with some memory, peripherals, I/O or a CPU connected to it. Each of these devices is capable of sending or receiving data onto this data bus. If these devices start to send or receive data at the same time a short circuit may occur when one device outputs to the bus a logic "1" the supply voltage, while another is set at logic level "0" or ground, resulting in a short circuit condition and possibly damage to the devices.

Then, the Tri-state Buffer can be used to isolate devices and circuits from the data bus and one another. If the outputs of several Tri-state Buffers are electrically connected together[**Decoders**](http://www.electronics-tutorials.ws/combination/comb_5.html) are used to allow only one Tri-state Buffer to be active at any one time while the other devices are in their high impedance state. An example of Tri-state Buffers connected to a single wire or bus is shown below.

**Tri-state Buffer Control**

|  |
| --- |
| Tri-state Buffer Control |

It is also possible to connect Tri-state Buffer "back-to-back" to produce a **Bi-directional Buffer** circuit with one "active-high buffer" connected in parallel but in reverse with one "active-low buffer". Here, the "enable" control input acts more like a directional control signal causing the data to be both read "from" and transmitted "to" the same data bus wire.

Commonly available **Digital Buffer** and **Tri-state Buffer** IC's include:

|  |  |
| --- | --- |
| TTL Logic Types* 74LS07 Hex Non-inverting Buffer
* 74LS17 Hex Buffer/Driver
* 74LS244 Octal Buffer/Line Driver
* 74LS245 Octal Bi-directional Buffer
 | CMOS Logic Types* CD4050 Hex Non-inverting Buffer
* CD4503 Hex Tri-state Buffer
* HEF40244 Octal Buffer with 3-state Output
 |

**Digital Non-inverting Buffer 7407**

|  |
| --- |
| Digital Hex Buffer 7407 |

**Octal Tri-state Buffer 74244**

|  |
| --- |
| Octal Tri-state Buffer 74244 |

In the next tutorial about **Digital Logic Gates**, we will look at the digital Logic [**OR Gate**](http://www.electronics-tutorials.ws/logic/logic_3.html) function as used in both TTL and CMOS logic circuits as well as its Boolean Algebra definition and truth tables.

Logic Gates **Tutorial: 10 of 10**

**Digital Logic Gates Summary**

In this section about **Digital Logic Gates**, we have seen that there are three main basic types of digital logic gate, the AND Gate , the OR Gate and the NOT Gate. We have also seen that each gate has an opposite or complementary form of itself in the form of the NAND Gate, the NOR Gate and the Buffer respectively, and that any of these individual gates can be connected together to form more complex [**Combinational Logic**](http://www.electronics-tutorials.ws/combination/comb_1.html) circuits.

We have also seen, that both the NAND gate and the NOR gate can both be classed as "**Universal**" gates as they can be used to construct any other gate type. In fact, any combinational circuit can be constructed using only two or three input NAND or NOR gates. We also saw that NOT gates and Buffers are single input devices that can also have a **Tri-state** High-impedance output which can be used to control the flow of data onto a common data bus wire.

**Digital Logic Gates** can be made from discrete components such as Resistors,Transistors and Diodes to form **RTL** (resistor-transistor logic) or**DTL** (diode-transistor logic) circuits, but today's modern digital 74xxx series integrated circuits are manufactured using **TTL** (transistor-transistor logic) based on NPN bipolar transistor technology or the much faster and low power CMOS MOSFET transistor logic used in the 74Cxxx, 74HCxxx, 74ACxxx and the 4000 series logic chips.

The eight most "standard" individual **Digital Logic Gates** are summarised below along with their corresponding truth tables.

**The Logic AND Gate**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| 2-input AND gate | B | A | Q |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| Boolean Expression Q = A . B | Read as A **AND** B gives Q |

 **The Logic NAND Gate**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| 2-input NAND gate | B | A | Q |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |
| Boolean Expression Q = A . B | Read as A **AND** B gives **NOT** Q |

 **The Logic OR Gate**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| 2-input OR gate | B | A | Q |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |
| Boolean Expression Q = A + B | Read as A **OR** B gives Q |

 **The Logic NOR Gate**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| 2-input NOR gate | B | A | Q |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |
| Boolean Expression Q = A + B | Read as A **OR** B gives **NOT** Q |

 **The Logic Exclusive-OR Gate (Ex-OR)**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| 2-input Ex-Or Logic Gate | B | A | Q |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |
| Boolean Expression Q = A B | Read as A **OR** B but not **BOTH** gives Q |

 **The Logic Exclusive-NOR Gate (Ex-NOR)**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| 2-input Ex-Nor Logic Gate | B | A | Q |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| Boolean Expression Q = A B | Read if A **AND** B the **SAME** gives Q |

 **The Buffer**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| A Buffer | A | Q |
| 0 | 0 |
| 1 | 1 |
| Boolean Expression Q = A | Read as **A** gives **Q** |

 **The NOT gate (Inverter)**

|  |  |
| --- | --- |
| Symbol | Truth Table |
| The NOT gate | A | Q |
| 0 | 1 |
| 1 | 0 |
| Boolean Expression Q = not A or A | Read as inverse of **A** gives Q |

The operation of the above **Digital Logic Gates** and their Boolean expressions can be summerised into a single truth table as shown below. This truth table shows the relationship between each output of the main digital logic gates for each possible input combination. **Truth Table Summary**

|  |  |
| --- | --- |
| Inputs | Truth Table Outputs for 2-input Logic Gates |
| B | A | AND | NAND | OR | NOR | EX-OR | EX-NOR |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |

|  |
| --- |
| Truth Table Output for Single-input Gates |
| A | NOT | Buffer |
| 0 | 1 | 0 |
| 1 | 0 | 1 |

**Pull-up and Pull-down Resistors**

One final point to remember, when connecting together digital logic gates to produce logic circuits, any "unused" inputs to the gates must be connected directly to either a logic level "1" or a logic level "0" by means of a suitable "Pull-up" or "Pull-down" resistor ( for example 1kΩ resistor ) to produce a fixed logic signal. This will prevent the unused input to the gate from "floating" about and producing false switching of the gate and circuit.

|  |
| --- |
| Pull-up and Pull-down Resistors |

|  |
| --- |
|  |