Quaid-e-Awam University of Engineering, Science and Technology

بســـــــمِ اللهِ الرَّحمٰنِ الرَّحِيم

In the name of Allah, the Most Merciful, the Most Kind

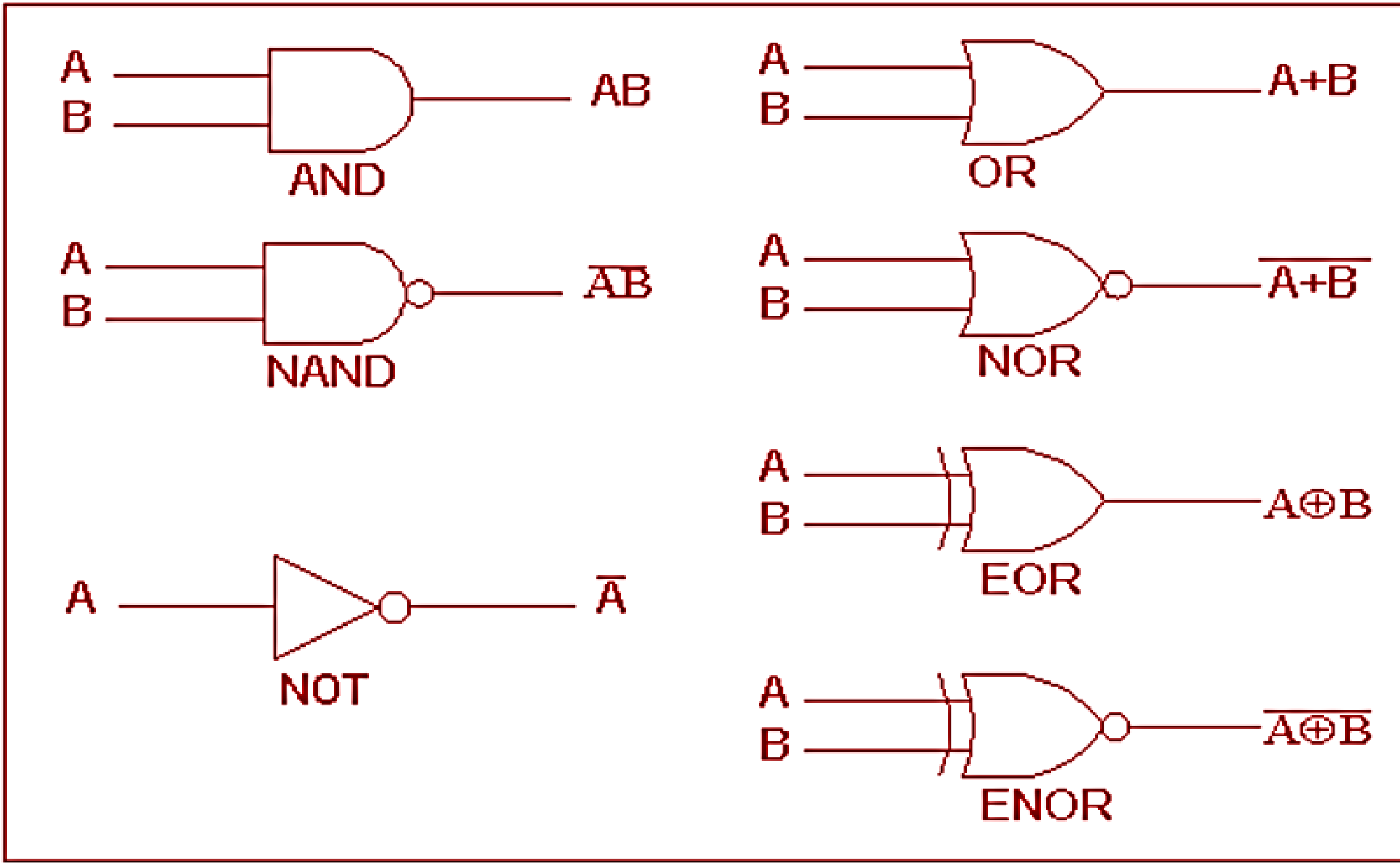
Date: 27-09-2021

BCS 103
Digital Logic & Computer Architecture

Lecture 13 and 14

Department of Computer Science

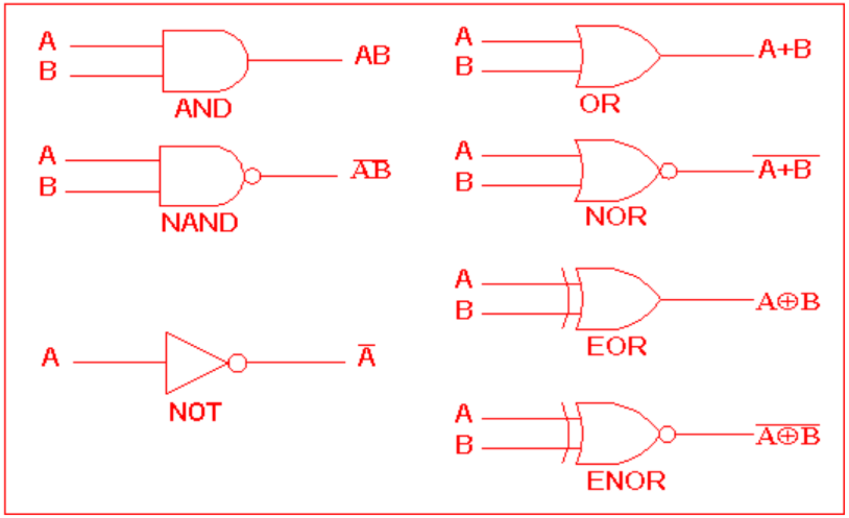
Logic Gates



A
B
AB
AND
A
B
$\overline{AB}$
NAND
A
$\overline{A}$
NOT
A
B
A+B
OR
A
B
$\overline{A+B}$
NOR
A
B
A⊕B
EOR
A
B
$\overline{A \oplus B}$
ENOR

Logic Gates

Logic gates are the basic building blocks of any digital system. It is an electronic circuit having one or more than one input and only one output. The relationship between the input and the output is based on a certain logic. Based on this, logic gates are named as AND gate, OR gate, NOT gate etc.



Logic Gates

Boolean functions may be practically implemented by using electronic gates. The following points are important to understand.

Electronic gates require a power supply.

Gate INPUTS and OUTPUTS are driven by voltages having two nominal values, e.g. 0V and 5V representing logic 0 and logic 1 respectively.

In general, there is only one output to a logic gate except in some special cases.

There is always a time delay between an input being applied and the output responding.

Logic Gates

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) technology or the much faster and low power CMOS based MOSFET transistor logic.

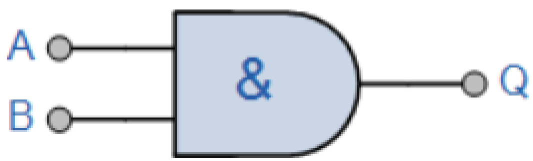
Truth Tables

Truth tables help understand the behaviour of logic gates.

- They show how the input(s) of a logic gate relate to its output(s).
- The gate input(s) are shown in the left column(s) of the table with all the different possible input combinations. This is normally done by making the inputs count up in binary.
- The gate output(s) are shown in the right hand side column

Truth Table		
B	A	Q
0	0	0
0	1	0
1	0	0
1	1	1

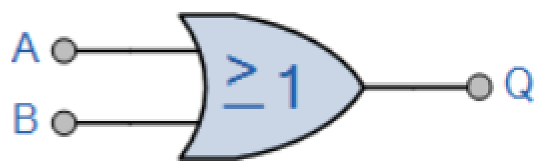
AND Gate

<table>
<tr><th>Symbol</th><th colspan="3">Truth Table</th></tr>
<tr><td rowspan="5">

2-input AND Digital Logic Gate</td><td>B</td><td>A</td><td>Q</td></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
<tr><td>Boolean Expression Q = A.B</td><td colspan="3">Read as A AND B gives Q</td></tr>
</table>

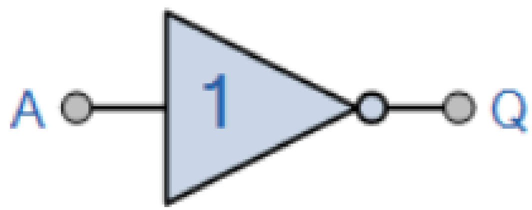
The AND gate is an electronic circuit that gives a **high** output (1) only if **all** its inputs are high. A dot (.) is used to show the AND operation i.e. A.B. Bear in mind that this dot is sometimes omitted i.e. AB

OR Gate

Symbol	Truth Table		
	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		

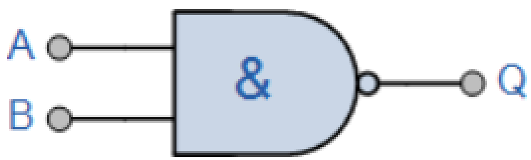
The OR gate is an electronic circuit that gives a high output (1) if one or more of its inputs are high. A plus (+) is used to show the OR operation.

NOT Gate

<table>
<tr><th>Symbol</th><th colspan="2">Truth Table</th></tr>
<tr><td rowspan="3"></td><td>A</td><td>Q</td></tr>
<tr><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td></tr>
<tr><td>Boolean Expression Q = not A or $\overline{A}$</td><td colspan="2">Read as inverse of A gives Q</td></tr>
</table>

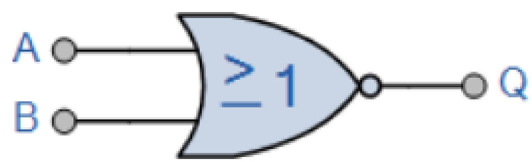
The NOT gate is an electronic circuit that produces an inverted version of the input at its output. It is also known as an inverter. If the input variable is A, the inverted output is known as NOT A. This is also shown as A', or A with a bar over the top, as shown at the outputs.

NAND Gate

Symbol	Truth Table		
	B	A	Q
	0	0	1
	0	1	1
	1	0	1
	1	1	0
Boolean Expression $Q = \overline{A\,.\,B}$	Read as A **AND** B gives **NOT** Q		

This is a NOT-AND gate which is equal to an AND gate followed by a NOT gate. The outputs of all NAND gates are high if any of the inputs are low. The symbol is an AND gate with a small circle on the output. The small circle represents inversion.

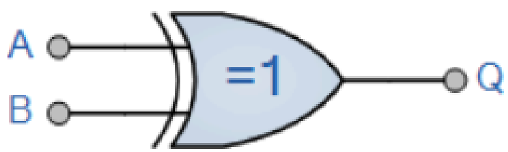
NOR Gate

<table>
<tr><th>Symbol</th><th colspan="3">Truth Table</th></tr>
<tr><td rowspan="5">
</td><td>B</td><td>A</td><td>Q</td></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>0</td></tr>
<tr><td>Boolean Expression $Q = \overline{A + B}$</td><td colspan="3">Read as A OR B gives NOT Q</td></tr>
</table>

This is a NOT-OR gate which is equal to an OR gate followed by a NOT gate. The outputs of all NOR gates are low if any of the inputs are high.

The symbol is an OR gate with a small circle on the output. The small circle represents inversion.

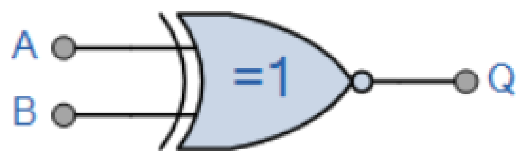
XOR Gate



Symbol	Truth Table		
	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 (odd)		

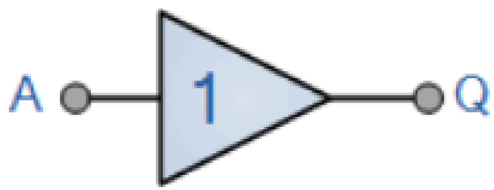
The 'Exclusive-OR' gate is a circuit which will give a high output if either, but not both, of its two inputs are high. An encircled plus sign () is used to show the XOR operation.

XNOR Gate

Symbol	Truth Table		
	B	A	Q
	0	0	1
	0	1	0
	1	0	0
	1	1	1
Boolean Expression $Q = \overline{A \oplus B}$	Read if A **AND** B the **SAME** gives Q (even)		

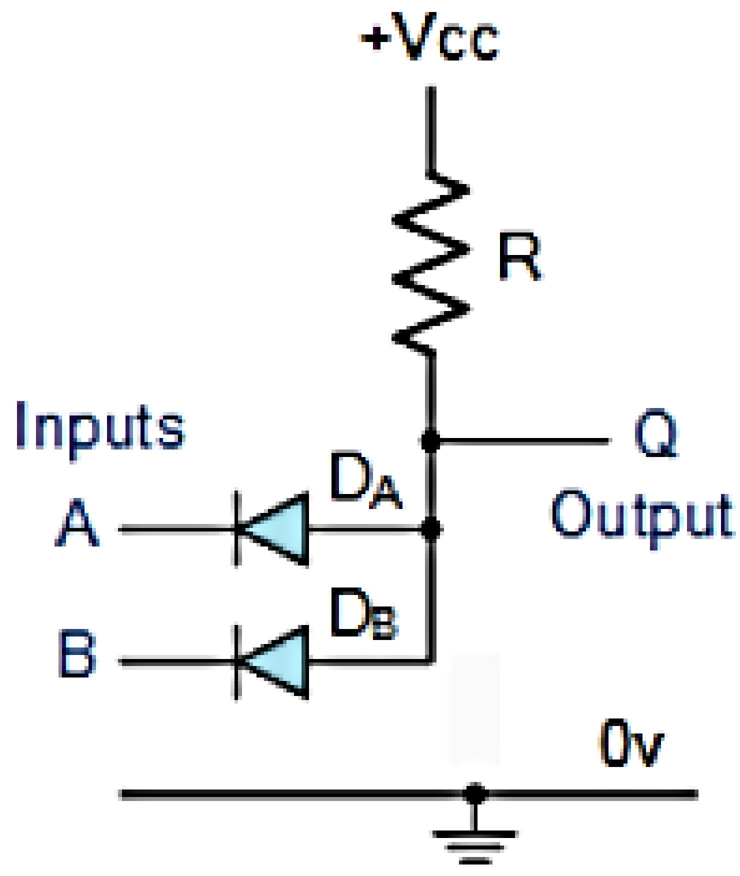
The 'Exclusive-NOR' gate circuit does the opposite to the EOR gate. It will give a low output if either, but not both, of its two inputs are high. The symbol is an EXOR gate with a small circle on the output. The small circle represents inversion.c

Buffer

<table>
<tr><th>Symbol</th><th colspan="2">Truth Table</th></tr>
<tr><td rowspan="3">
</td><td>A</td><td>Q</td></tr>
<tr><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td></tr>
<tr><td>Boolean Expression Q = A</td><td colspan="2">Read as A gives Q</td></tr>
</table>

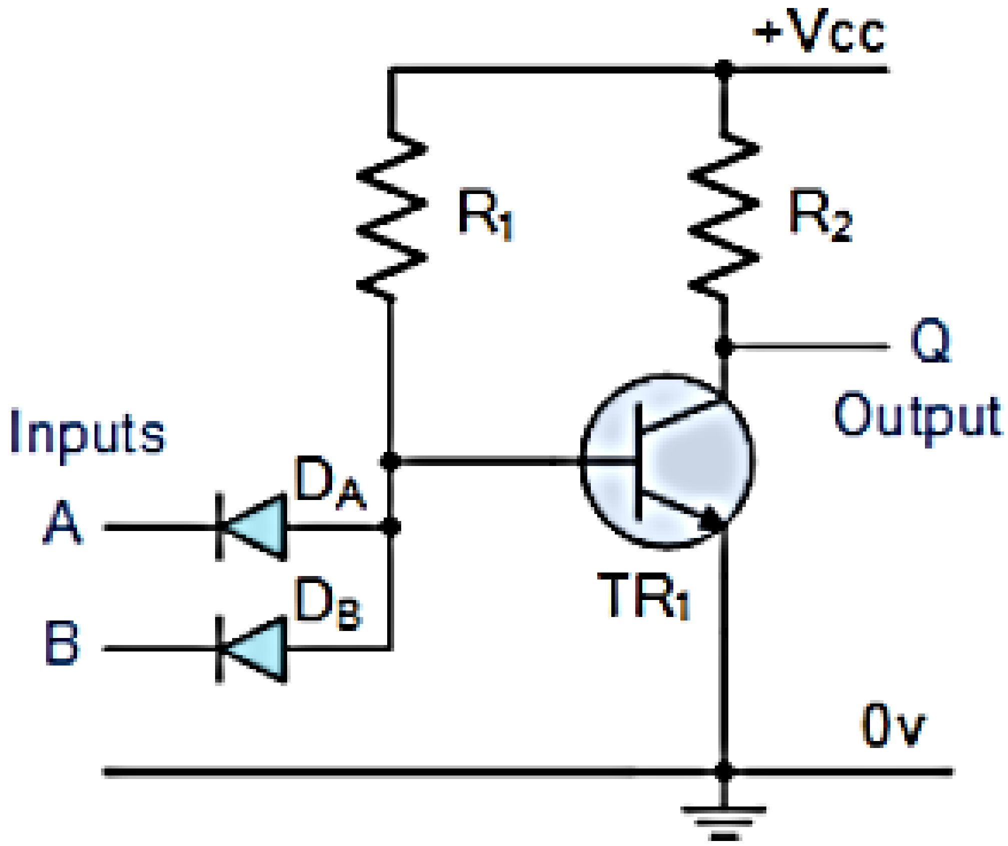
A buffer has only a single input and a single output with behavior that is the opposite of an NOT gate. It simply passes its input, unchanged, to its output. In a boolean logic simulator, a buffer is mainly used to increase propagation delay. In a real-world circuit, a buffer can be used to amplify a signal if its current is too weak.

Diode-Resister Logic



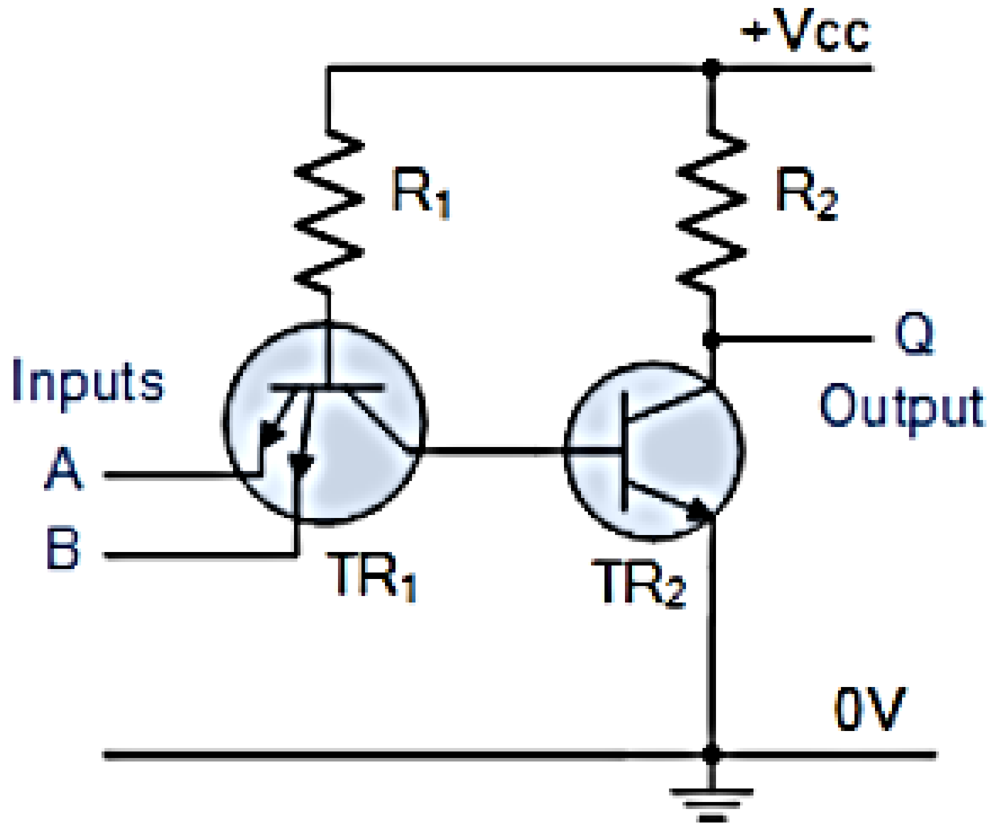
2-input AND Gate

Diode-Transistor Logic



2-input NAND Gate

Transistor-Transistor Logic

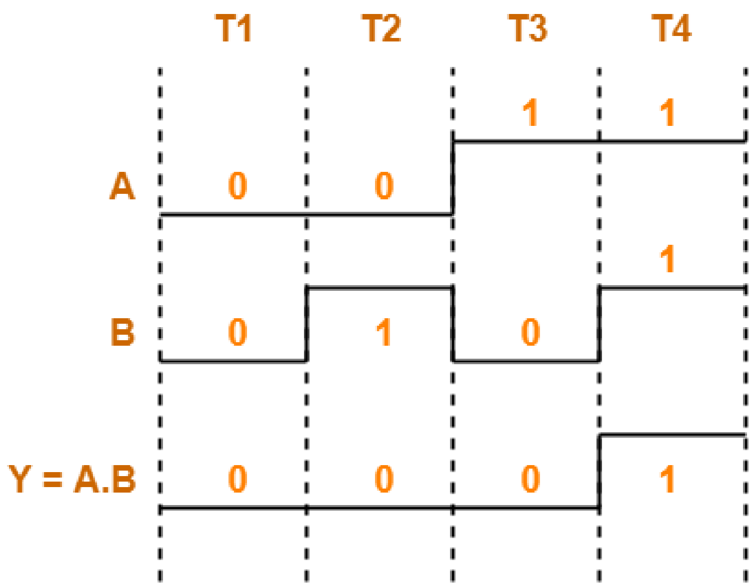


2-input NAND Gate

AND Gate Timing Diagram

A	B	Y = A.B
0	0	0
0	1	0
1	0	0
1	1	1

Truth Table

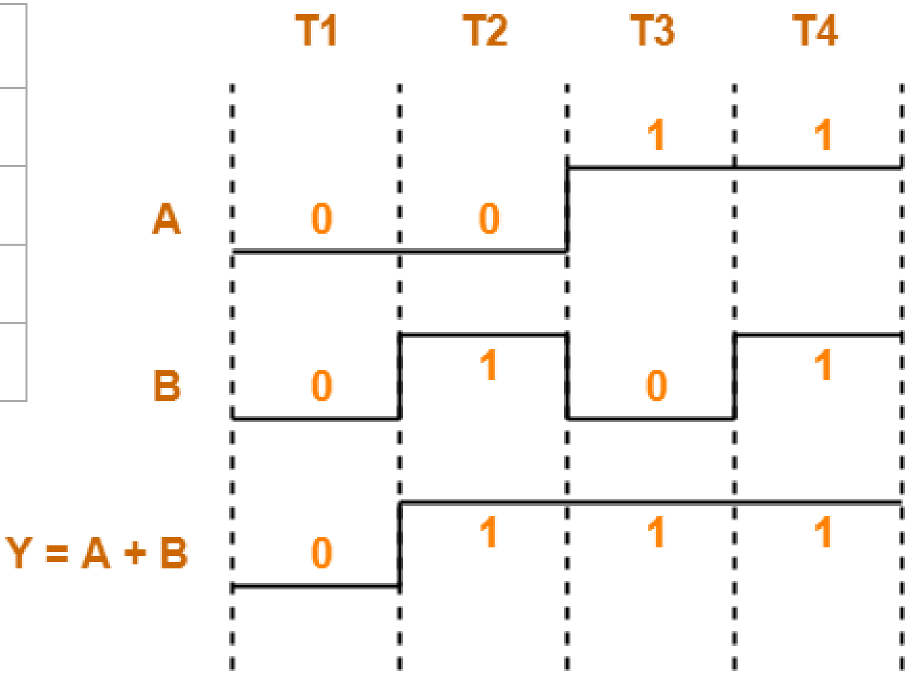


AND Gate Timing Diagram

OR Gate Timing Diagram

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

Truth Table

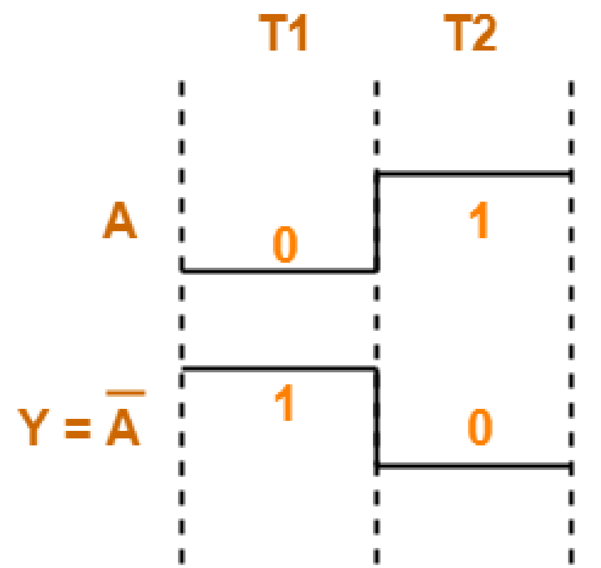


OR Gate Timing Diagram

NOT Gate Timing Diagram

A	Y = A'
0	1
1	0

Truth Table



NOT Gate Timing Diagram

Note: Make timing diagrams for remaining Gates

Thanks