System

Washing m/c

car

Computer

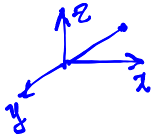
CPU ← Computer

ALU

Co-Processor

ASIC

System



universal gates

Basic gates

- AND ---- For AND operation on two variables, say *X* and *Y*, the operation results in 1 ← true if and only if both the variables are 1 and the result is denoted by $\mathrm{F} = X \cdot Y$, where F is the result of the operation. If there are more than two variables, then the operation results in 1 iff all the variables are 1, else it will be 0. ← false

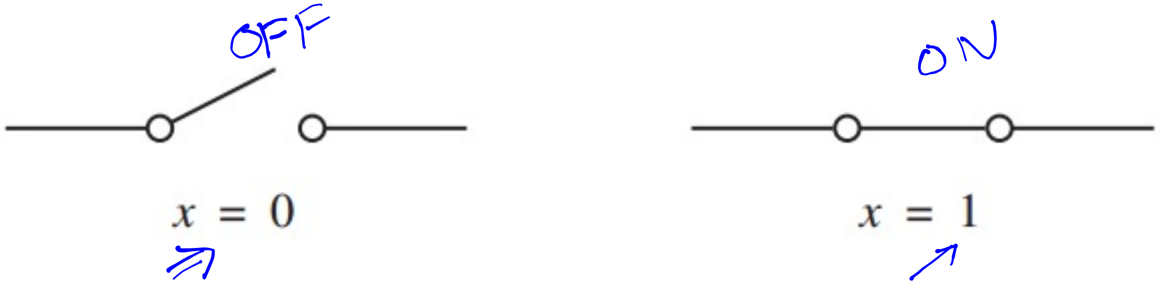
→ ∩ ← intersection

- OR ---- For OR operation on two variables, say *X* and *Y*, the operation results in 0 if and only if both the variables are 0 and the result is denoted by $\mathrm{F} = X + Y$, where F is the result of the operation. If there are more than two variables, then the operation results in 0 iff all the variables are 0, else it will be 1.

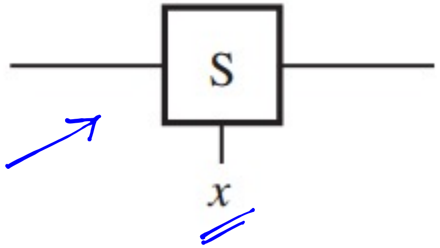
1 0 A B A∩B even numbers nr. divisible by 3

- NOT ---- For operation on one variable only, say X. This operation results in 0 if the variable is 1 and results in 1 if the variable is 0 and the result is denoted by $\mathrm{F} = \overline{X}$ or $F = X'$, where F is the result of the operation. This is also called complementing and F is called the logical complement of X. 0 → 1 1 → 0
- We can combine gates into fancier combinational circuits such as the multiplexer. This device allows us to steer any one of a number of inputs to a single output.

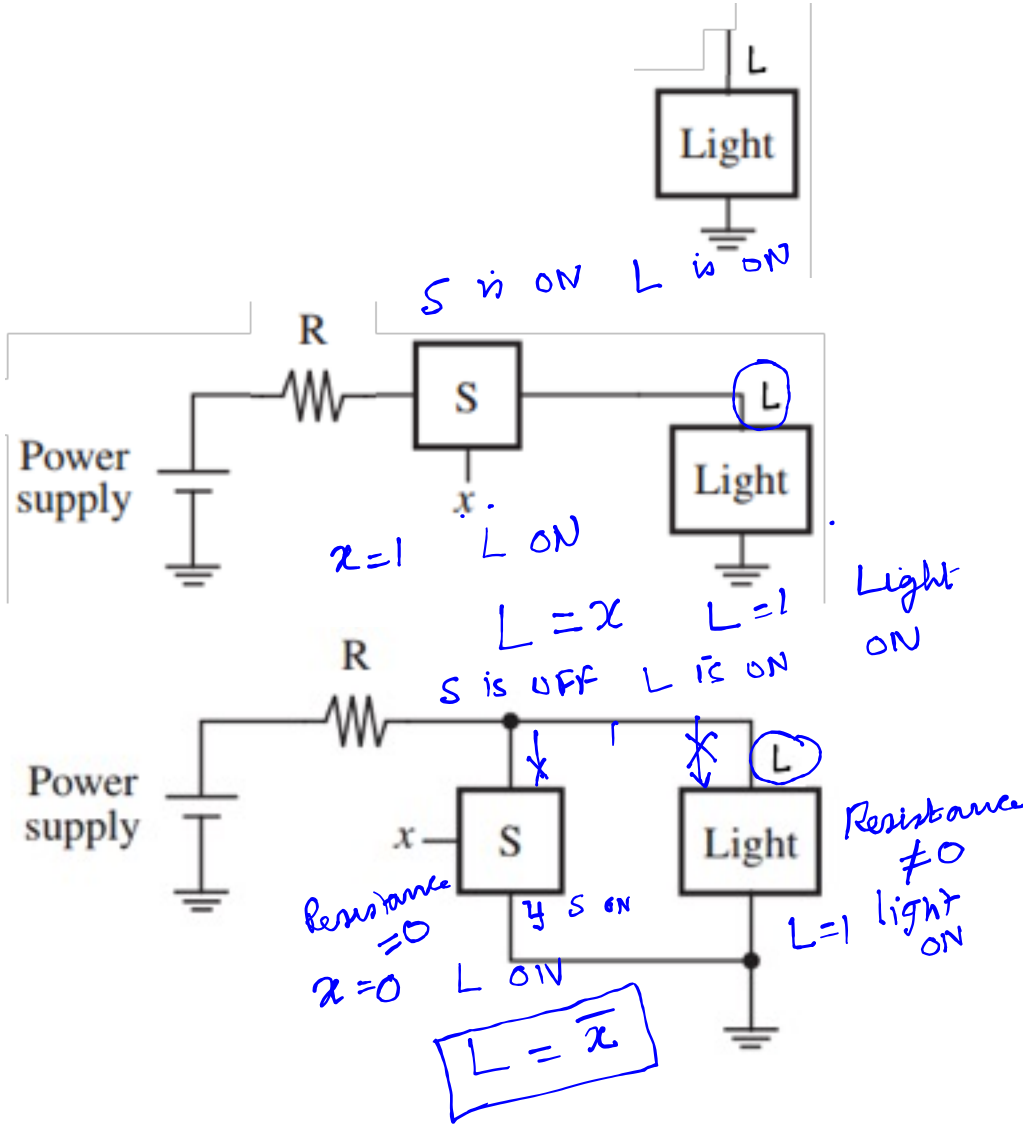
Logic Function (through examples)



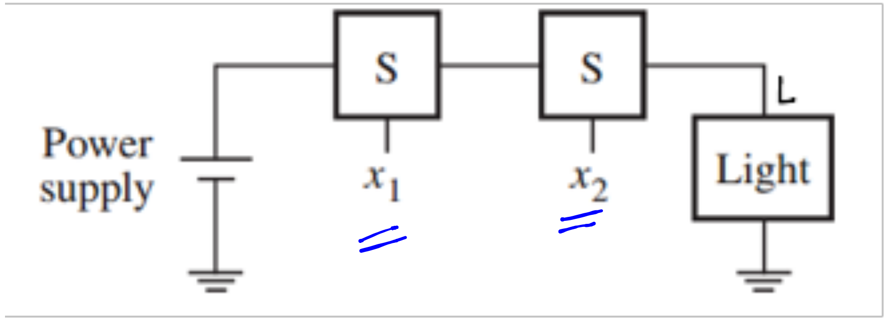
Two states of a switch



Symbol for a switch



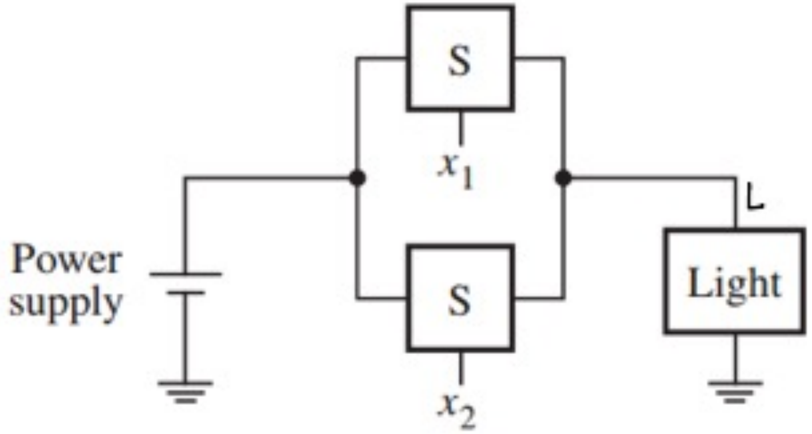
Boolean Expression



Intersection (∩) $L = x_1 \cdot x_2$

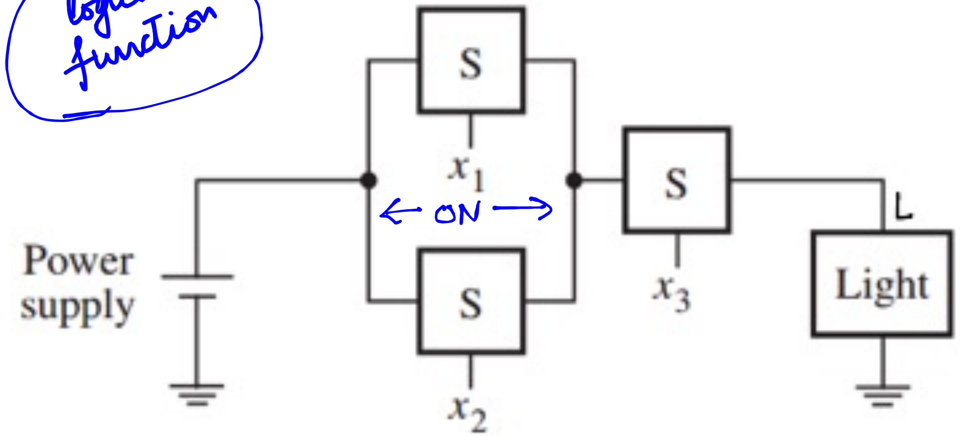
AND (·)

logical function



Union (∪) $L = x_1 + x_2$

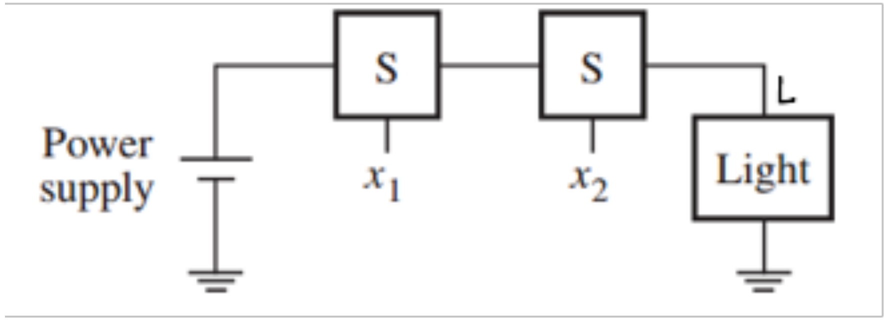
OR (+)



$(x_1 + x_2) \cdot x_3 = L$

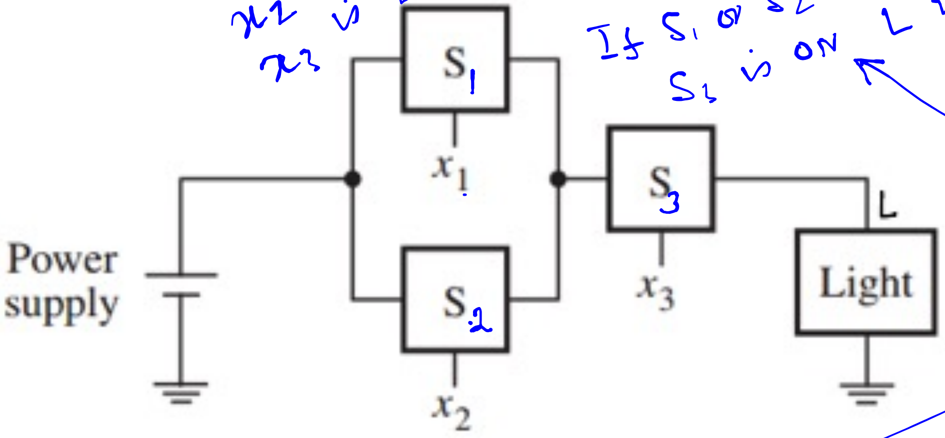
Combination of AND and OR

Boolean Expression

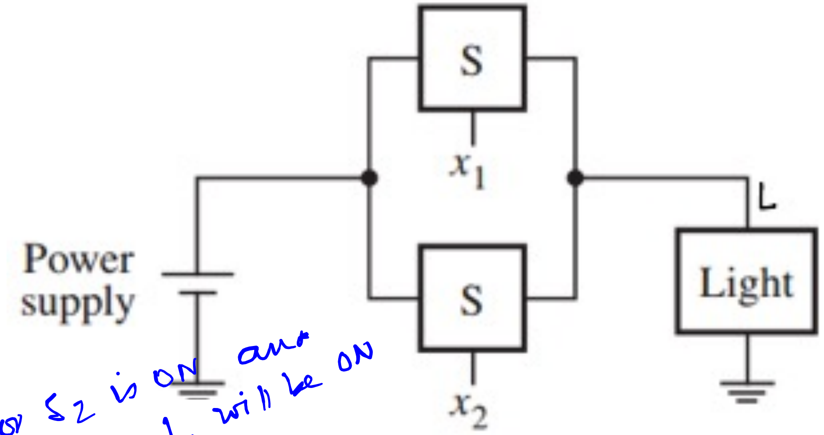


$$L = x_1 \cdot x_2$$

x_1 is 1 S_1 is ON
x_2 is 1 S_2 is ON
x_3 is 1 S_3 is ON



If S_1 or S_2 is ON and S_3 is ON L will be ON



$$L = x_1 + x_2$$

"If x_1 or x_2 is 1 and x_3 is 1 Light will be ON."

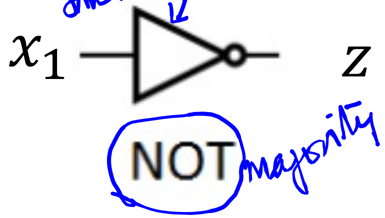
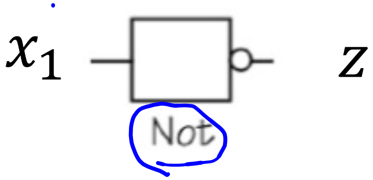
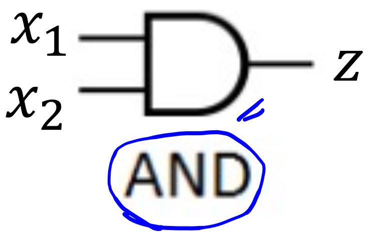
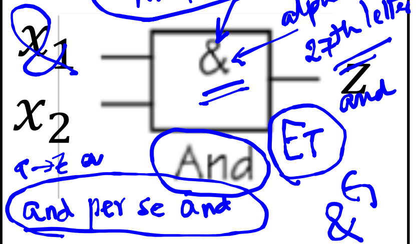
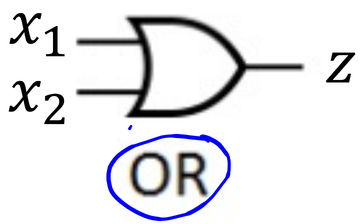
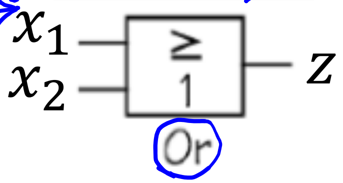
$$L = \underline{(x_1 + x_2)} \cdot \underline{x_3}$$

In logic form

- AND ---- For AND operation on two variables, say *X* and *Y*, the operation results in 1 if and only if both the variables are 1 and the result is denoted by $\mathrm{F} = X \cdot Y$, where F is the result of the operation. If there are more than two variables, then the operation results in 1 iff all the variables are 1, else it will be 0.
- OR ---- For OR operation on two variables, say *X* and *Y*, the operation results in 0 if and only if both the variables are 0 and the result is denoted by $\mathrm{F} = X + Y$, where F is the result of the operation. If there are more than two variables, then the operation results in 0 iff all the variables are 0, else it will be 1.
- NOT ---- For operation on one variable only, say X. This operation results in 0 if the variable is 1 and results in 1 if the variable is 0 and the result is denoted by $\mathrm{F} = \overline{X}$ or $F = X'$, where F is the result of the operation. This is also called complementing and F is called the logical complement of X.
- We can combine gates into fancier combinational circuits such as the multiplexer. This device allows us to steer any one of a number of inputs to a single output.

Basic Logic Gates

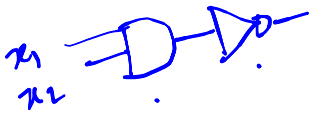
for information Institute of Electrical & Electronics Engineer

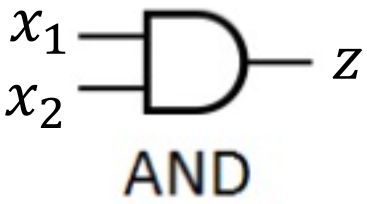
<table>
<tr><th>Name</th><th>Symbol
Normally used</th><th>IEEE
Recommended</th><th>Logical Function</th><th>Truth Table</th></tr>
<tr><td>NOT
Inverter</td><td>Inverter

majority</td><td></td><td>$Z = \overline{x_1}$</td><td><table><tr><th>x_1</th><th>Z</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>for Inverter</td></tr>
<tr><td>AND</td><td>AND gate
</td><td>Ampersand
alphabet 27th letter

and
ET
T→Z or
and per se and
&</td><td>$Z = x_1 \cdot x_2$</td><td><table><tr><th>x_1</th><th>x_2</th><th>Z</th></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></table>Truth Table. for AND</td></tr>
<tr><td>OR</td><td></td><td>both are acceptable
</td><td>$Z = x_1 + x_2$</td><td><table><tr><th>x_1</th><th>x_2</th><th>Z</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>for OR</td></tr>
</table>

Truth Table: Positive Logic Function vs. Negative Logic Function

TT

x_1	x_2	z
0	0	0 → 1
0	1	0 → 1
1	0	0 → 1
1	1	1 → 0



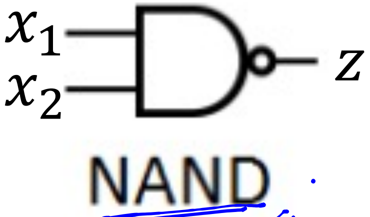


Basic

AND followed by NOT

x_1	x_2	z
0	0	1
0	1	1
1	0	1
1	1	0

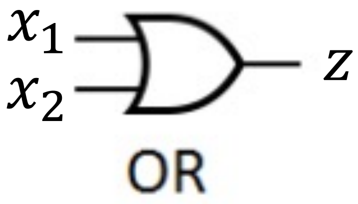
Negative logic function



$Z = \overline{(x_1 \cdot x_2)}$

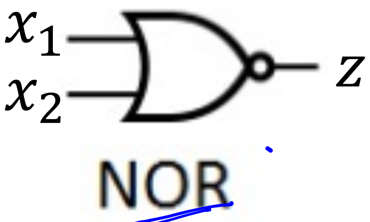
Negation

x_1	x_2	z
0	0	0
0	1	1
1	0	1
1	1	1



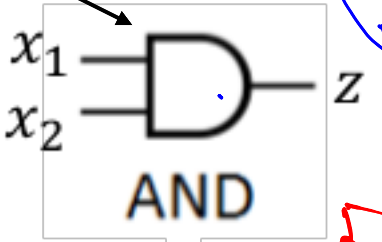
OR followed by NOT

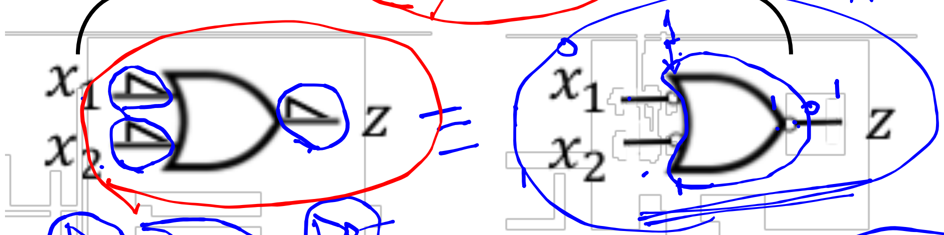
x_1	x_2	z
0	0	1
0	1	0
1	0	0
1	1	0



$Z = \overline{(x_1 + x_2)}$

Positive and Negative Logic Symbol:





Negative gate

-ve gate

high voltage — 1
low voltage — 0 } +ve gate

NAND NOR -ve function

AND

NOR

x_1	x_2	z
0	0	0
0	1	0
1	0	0
1	1	1

x_1	x_2	z
0	0	0
0	1	0
1	0	0
1	1	1

$\bar{x}_1 \cdot \bar{x}_2$

$(x_1 + x_2)$

$x_1 = 1 \quad \bar{x}_1 = 0$

De Morgan's law

$\overline{x_1 \cdot x_2} \to \bar{x}_1 + \bar{x}_2$

$z = \overline{x_1 \cdot x_2}$

symbol → +ve & -ve
logic function → +ve & -ve
gate → +ve & -ve
merged older