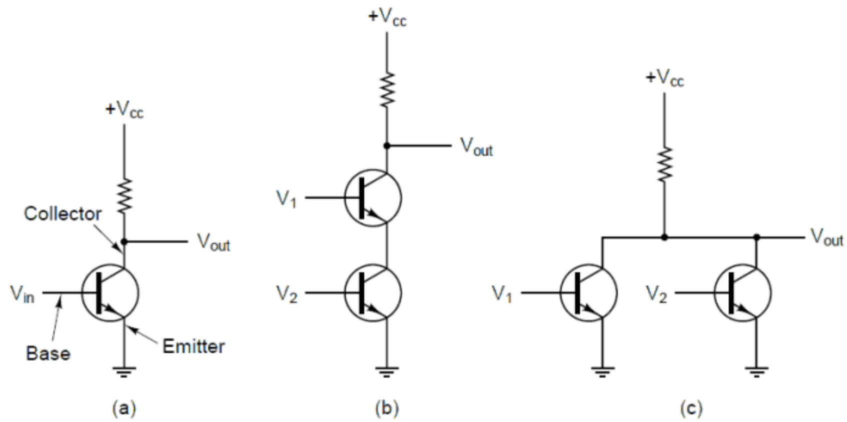
Ch 3.1 Logic Design

NOTE08: Introduction to logic gates and Boolean algebra

Text reference: ***Section 3.1***

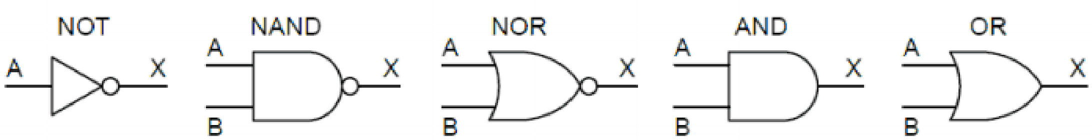
Remember these? Transistors designs for inverter, nand and nor gates



Abstraction! We'll work at the logic gate level.

3 representations of combinatorial logic: truth table, Boolean equations, and gates.

Like these:



A	X
0	1
1	0

A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

NOT	NAND	NOR	AND	OR
f = A'	f = (AB)'	f = (A+B)'	f = AB	f = AB

Some identities of Boolean algebra:

Name	AND form	OR form
Identity law	$1A = A$	$0 + A = A$
Null law	$0A = 0$	$1 + A = 1$
Idempotent law	$AA = A$	$A + A = A$
Inverse law	$A\overline{A} = 0$	$A + \overline{A} = 1$
Commutative law	$AB = BA$	$A + B = B + A$
Associative law	$(AB)C = A(BC)$	$(A + B) + C = A + (B + C)$
Distributive law	$A + BC = (A + B)(A + C)$	$A(B + C) = AB + AC$
Absorption law	$A(A + B) = A$	$A + AB = A$
De Morgan's law	$\overline{AB} = \overline{A} + \overline{B}$	$\overline{A + B} = \overline{A}\,\overline{B}$

Use these identities to reduce the size number of gates to implement logic.

The XOR function: f = AB’ + A’B. Equivalent representations are shown below:

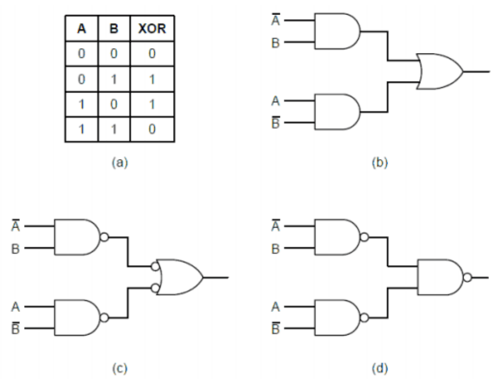


Figure 3-8. (a) The truth table for the XOR function.
(b)–(d) Three circuits for computing it.

Transformations between each of the forms: truth table, equations, gates

Terms:

- **complete** - a set of gates is complete if they can describe any Boolean function
- **dual** - replacing AND/OR and 0/1… see identities table above
- **equivalent** - two Boolean forms are equivalent if they have the same truth table