

Switch Symbol: INVERTER (NOT)

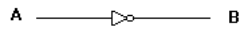
$$B = A'$$



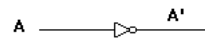
Truth Table:

<u>A</u>	<u>B</u>
0	1
1	0

Digital Logic Gate Symbol: INVERTER (NOT)



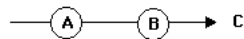
(conceptually)



(required syntax)

Switch Symbol: AND

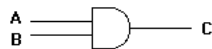
$$C = AB$$



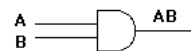
Truth Table:

<u>A</u>	<u>B</u>	<u>C</u>
0	0	0
0	1	0
1	0	0
1	1	1

Digital Logic Gate Symbol: AND



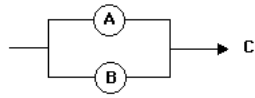
(conceptually)



(required syntax)

Switch Symbol: OR

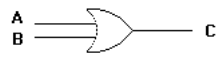
$$C = A + B$$



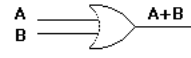
Truth Table:

<u>A</u>	<u>B</u>	<u>C</u>
0	0	0
0	1	1
1	0	1
1	1	1

Digital Logic Gate Symbol: OR



(conceptually)



(required syntax)

XOR (exclusive OR)

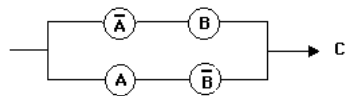
- one or the other is true, but not both

<u>A</u>	<u>B</u>	<u>A ⊕ B</u>
0	0	0
0	1	1
1	0	1
1	1	0

$A \oplus B$ means $A'B + AB'$

Switch Symbol: EXCLUSIVE OR (XOR)

$$C = A \oplus B$$



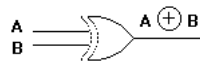
Truth Table:

<u>A</u>	<u>B</u>	<u>C</u>
0	0	0
0	1	1
1	0	1
1	1	0

Digital Logic Gate Symbol: XOR



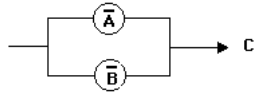
(conceptually)



(required syntax)

Switch Symbol: NAND

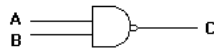
$$C = (AB)'$$



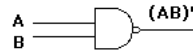
Truth Table:

<u>A</u>	<u>B</u>	<u>AB</u>	<u>C</u>
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Digital Logic Gate Symbol: NAND



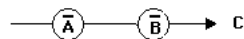
(conceptually)



(required syntax)

Switch Symbol: NOR

$$C = (A + B)'$$



Truth Table:

<u>A</u>	<u>B</u>	<u>A+B</u>	<u>C</u>
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

Digital Logic Gate Symbol: NOR

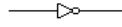


(conceptually)

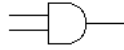


(required syntax)

Digital Logic Gate Symbols:



The INVERTER (or NOT) gate



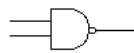
The AND gate



The OR gate



The XOR gate

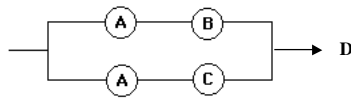


The NAND gate



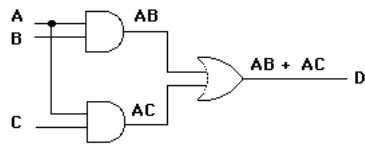
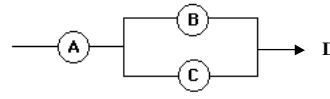
The NOR gate

$D = AB + AC$

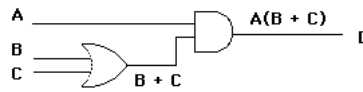


switch diagram

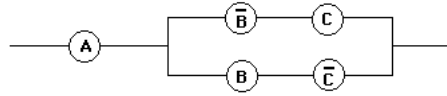
simplifies to $D = A(B + C)$



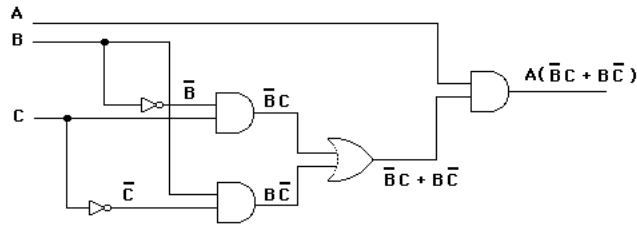
gate diagram



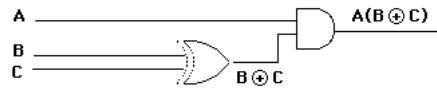
Simplified circuit from handouts: $A(B'C + BC')$



switch diagram

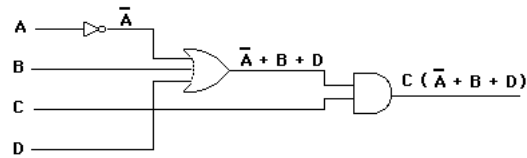


gate diagram

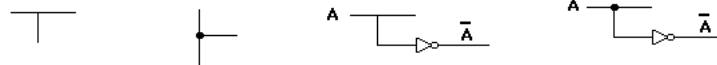


gate diagram using XOR

Gates can have more than 2 inputs:



Connections:



No connection:

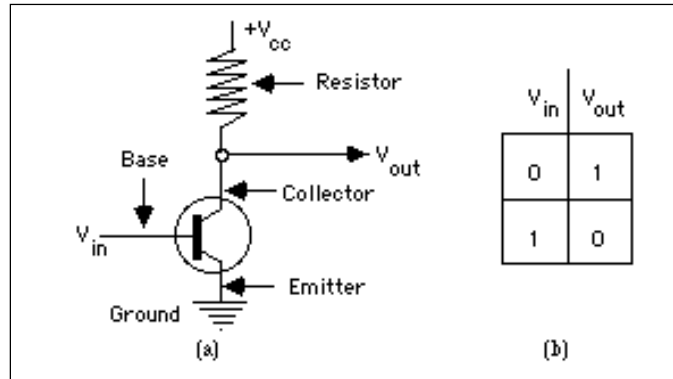


Transistors:

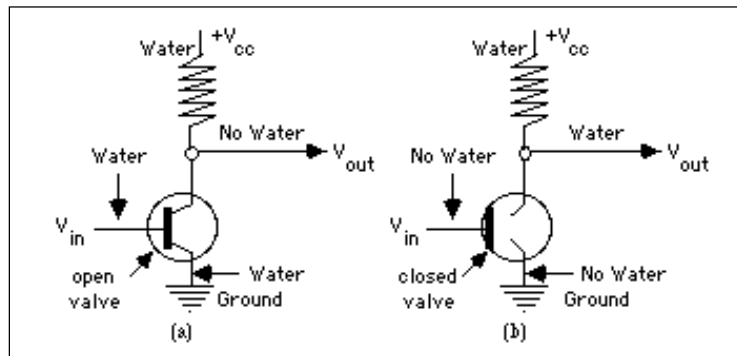
- bipolar
 - TTL (transistor-transistor logic)
 - output is derived from 2 transistors
 - ECL (emitter-coupled logic)
- MOS (metal oxide semiconductor)
 - slower than bipolar
 - smaller than bipolar
 - and uses less power
- the transistor acts as a fast binary switch
 - specifically, as an inverter

Transistor:

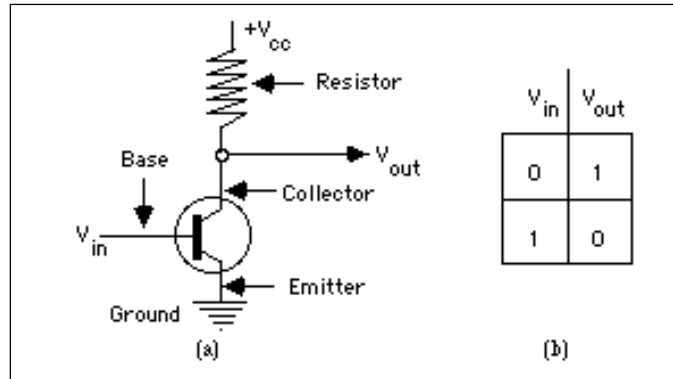
- is made of a semi-conducting material
 - usually silicon
- acts as either a conducting wire or a resistor
 - depending on the voltage level of its input
- has 3 terminals (electrical contacts)
 - the collector
 - high voltage coming in from the power source
 - is also connected to the output line
 - the base
 - voltage coming in determines whether there is a connection between the source and the ground
 - the emitter
 - connected to a ground wire
- if the base signal is high
 - the transistor pulls the current from the power source to the ground
 - the output signal is low
- if the base signal is low
 - the transistor resists the current
 - the output signal is high



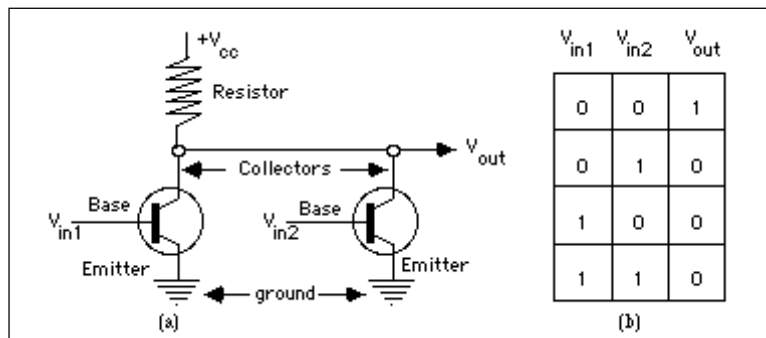
The inverter circuit using a transistor



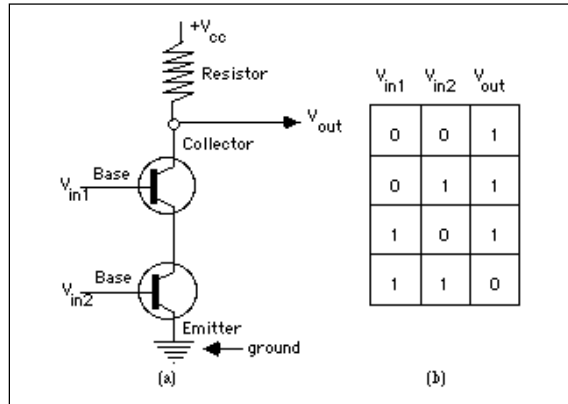
The water analogy applied to the inverter



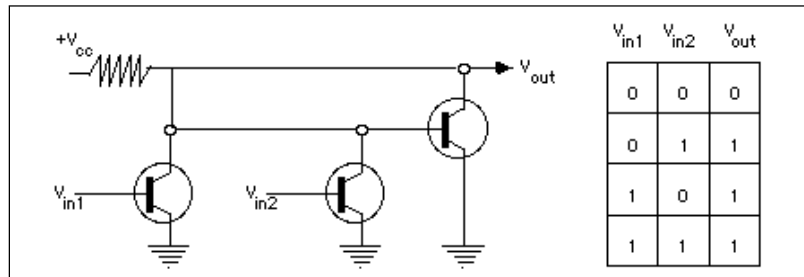
The inverter circuit and symbols using a transistor



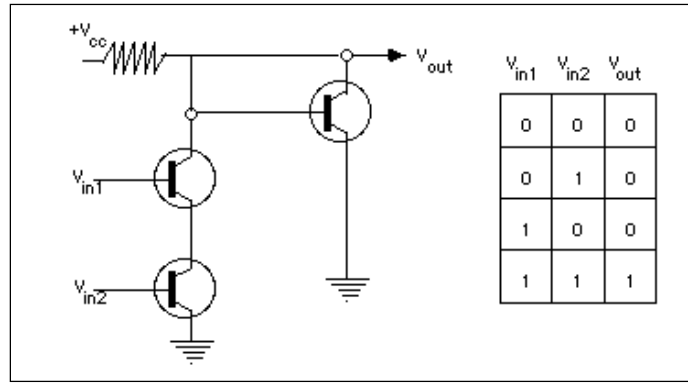
The NOR circuit



The NAND circuit



The OR circuit



The AND circuit

The Majority function:

- outputs a 1 if 2 or more of its 3 inputs are 1

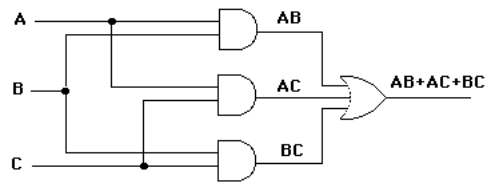
As a function: $AB + AC + BC + ABC$

To simplify:

$$\begin{aligned}
 & \underline{AB + AC + BC + ABC} \\
 &= \underline{AB + ABC} + AC + BC \\
 &= AB + AC + BC
 \end{aligned}$$

The majority function: $A'BC + AB'C + ABC' + ABC$

Simplified: $AB + AC + BC$



The sum-of-products method:

- uses the output from the truth table for a function
- expresses the function in terms of some of the combinations of the inputs
 - those that produce a 1 in the output

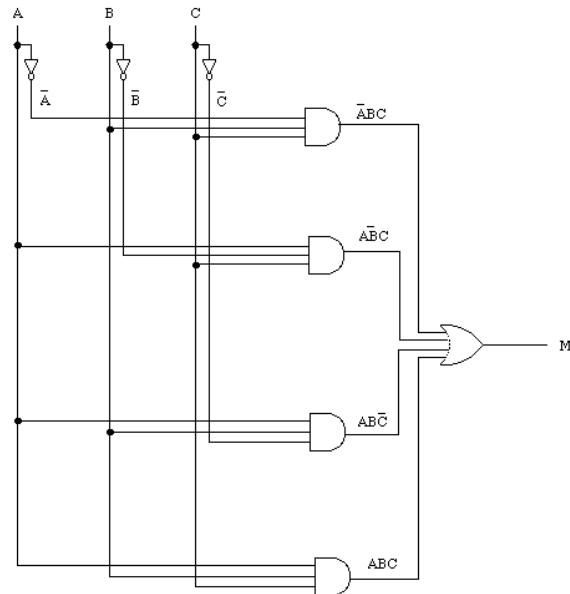
- any boolean function can be implemented as a circuit using this method
 - use the truth table for the function
 - use INVERTERS for the complements
 - draw an AND gate for each combination of variables that produces a 1
 - feed all the outputs from the AND gates into an OR gate

The majority function:

-the output is 1 if a majority of the inputs are 1

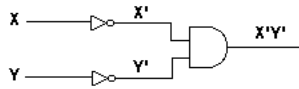
- the truth table:

<u>A</u>	<u>B</u>	<u>C</u>	<u>Output</u>	
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	→ A'BC
1	0	0	0	
1	0	1	1	→ AB'C
1	1	0	1	→ ABC'
1	1	1	1	→ ABC

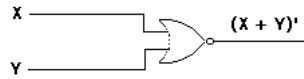


A circuit for the majority function of three variables

Review: $X'Y'$ as a gate diagram:

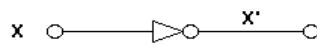


But: $X'Y'$ can also be expressed as $(X + Y)'$:



5 transistors vs. 2 transistors

To change x to its inverse: the inverter



To change x to its inverse: the nor gate



To change x to its inverse: the nand gate

