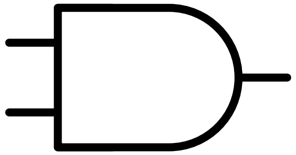
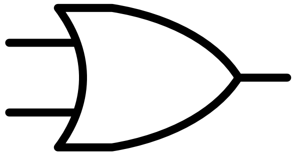
Simple Logic



AND Truth Table

AND	0	1
0	0	0
1	0	1

0 AND 0 is 0 (a false statement AND a false statement is a false combination)
0 AND 1 is 0 (a false statement AND a true statement is a false combination)
1 AND 0 is 0 (a true statement AND a false statement is a false combination)
1 AND 1 is 1 (a true statement AND a true statement is a true combination)

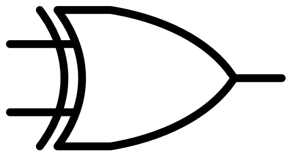


OR Truth Table

Inclusive OR

OR	0	1
0	0	1
1	1	1

0	OR	0	is	0	(a false statement OR a false statement is a false combination)
0	OR	1	is	1	(a false statement OR a true statement is a true combination)
1	OR	0	is	1	(a true statement OR a false statement is a true combination)
1	OR	1	is	1	(a true statement OR a true statement is a true combination)



XOR Truth Table

Exclusive OR

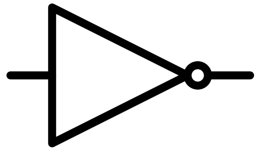
XOR	0	1
0	0	1
1	1	0

0 XOR 0 is 0 (a false statement XOR a false statement is a false combination. XOR and OR are the same for these inputs.)

0 XOR 1 is 1 (a false statement XOR a true statement is a true combination. XOR and OR are the same for these inputs.)

1 XOR 0 is 1 (a true statement XOR a false statement is a true combination. XOR and OR are the same for these inputs.)

1 XOR 1 is 0 (a true statement XOR a true statement is a false combination, because XOR excludes both statements being true)

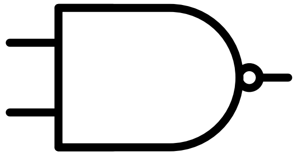


NOT Truth Table

NOT	
0	1
1	0

NOT 0 is   1   (the negative of a false statement is a true statement)

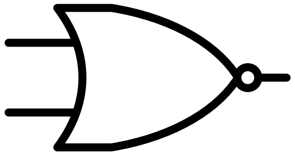
NOT 1 is   0   (the negative of a true statement is a false statement)



NAND Truth Table

NAND	0	1
0	1	1
1	1	0

0	NAND	0	is	1	(false AND false is false, which is negated to true)
0	NAND	1	is	1	(false AND true is false, which is negated to true)
1	NAND	0	is	1	(true AND false is false, which is negated to true)
1	NAND	1	is	0	(true AND true is true, which is negated to false)



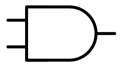
NOR Truth Table

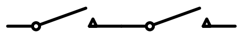
NOR	0	1
0	1	0
1	0	0

```
0  NOR  0  is   1    (false OR false is false, which is negated to true)
0  NOR  1  is   0    (false OR true  is true,  which is negated to false)
1  NOR  0  is   0    (true  OR false is true,  which is negated to false)
1  NOR  1  is   0    (true  OR true  is true,  which is negated to false)
```

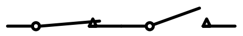
Logic Gate Implementation

AND Gate

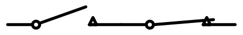




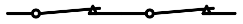
open AND open is open
0 AND 0 = 0



closed AND open is open
1 AND 0 = 0

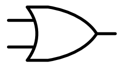


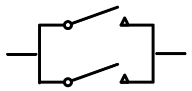
open AND closed is open
0 AND 1 = 0



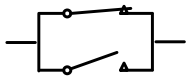
closed AND closed is closed
1 AND 1 = 1

OR Gate

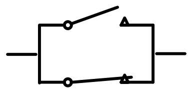




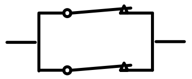
open OR open is open
0 OR 0 = 0



closed OR open is closed
1 OR 0 = 1

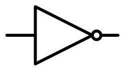


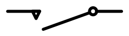
open OR closed is closed
0 OR 1 = 1



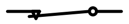
closed OR closed is closed
1 OR 1 = 1

NOT Gate





NOT closed is open
NOT 1 = 0

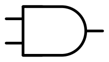


NOT open is closed
NOT 0 = 1

Logic Gate Implementation

To make a computer, all you need is Memorial Stadium for an afternoon, a few thousand chairs, a few thousand undergraduates, one pizza for each three undergraduates, and lots and lots of soda.

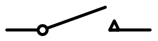
AND Gate



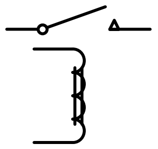
	AND		IS	
Hand Down	AND	Hand Down	is	Hand Down
0	AND	0	=	0
	AND		IS	
Hand Down	AND	Hand Up	is	Hand Down
0	AND	1	=	0
	AND		IS	
Hand Up	AND	Hand Down	is	Hand Down
1	AND	0	=	0
	AND		IS	
Hand Up	AND	Hand Up	is	Hand Up
1	AND	1	=	1

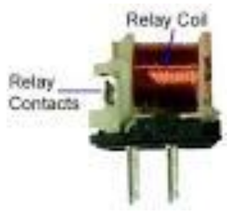
Different Types of Switches

Mechanical Switch

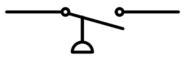


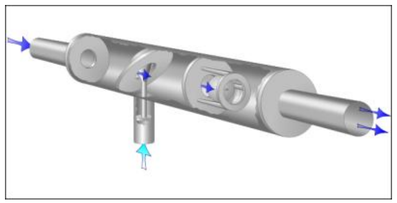
Electro-mechanical Relay





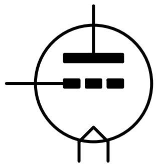
Fluid Pressure Value



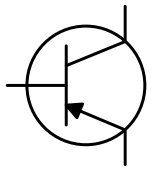


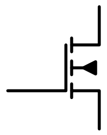
Electronic Tube

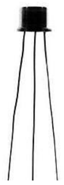
Called a "valve" in England



Transistor







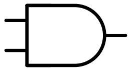
Logic Gate Implementation

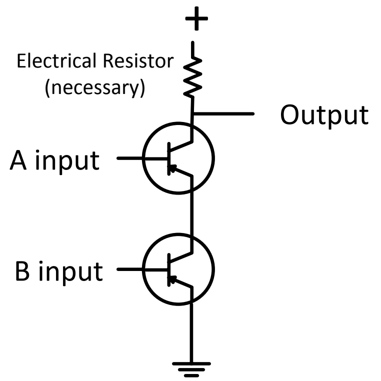
using Transisters

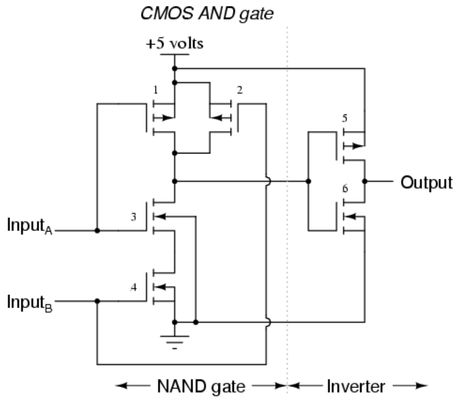
Older style individual transistors

CMOS type transistors eliminate the need for resistors, allowing millions of transistors on a single chip

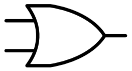
AND Gate

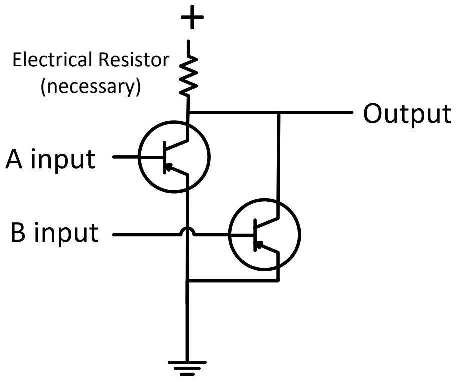


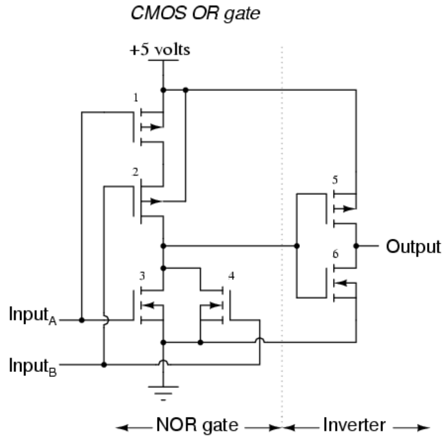




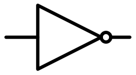
OR Gate

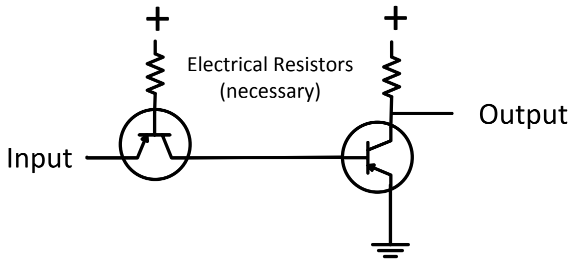


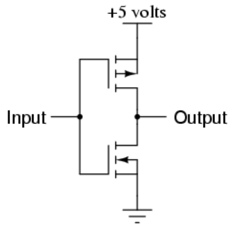




NOT Gate

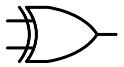


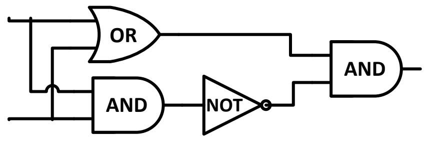


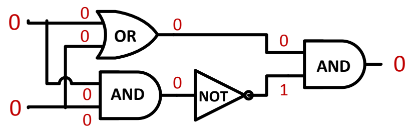


Logic Gate Implementation

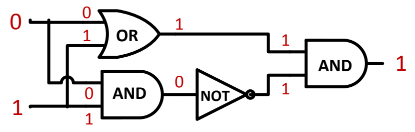
XOR Gate



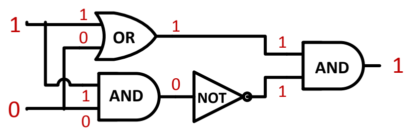




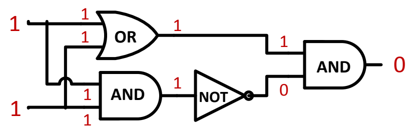
0 XOR 0 = 0



0 XOR 1 = 1

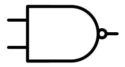


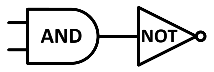
1 XOR 0 = 1

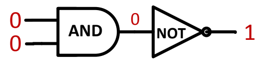


1 XOR 1 = 0

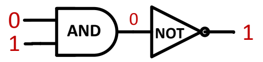
NAND Gate



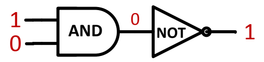




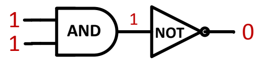
0 NAND 0 = 1



0 NAND 1 = 1

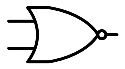


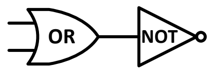
1 NAND 0 = 1

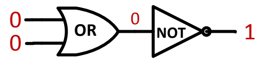


1 NAND 1 = 0

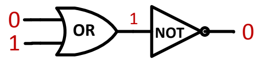
NOR Gate



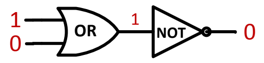




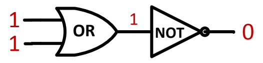
0 NOR 0 = 1



0 NOR 1 = 0



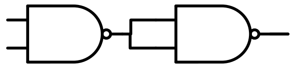
1 NOR 0 = 0



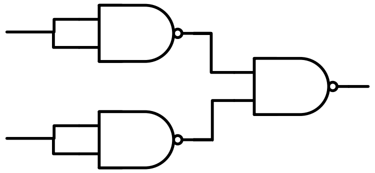
1 NOR 1 = 0

NAND can implement all other logic gates

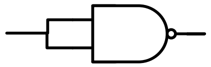
AND Gate



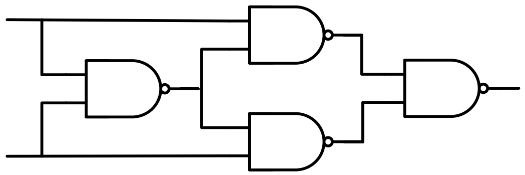
OR Gate



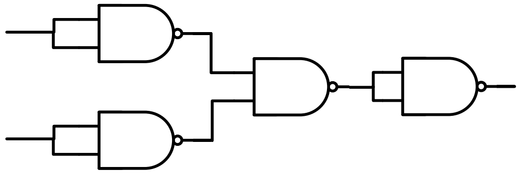
NOT Gate



XOR Gate



NOR Gate



Next Presentation:
Full Adder and Five Bit Adder