Chapter Goals

- Distinguish among categories of numbers
- Describe positional notation
- Convert numbers in other bases to base 10
- Convert base-10 numbers to numbers in other bases
- Describe the relationship between bases 2, 8, and 16
- Explain the importance to computing of bases that are powers of 2

Numbers

Natural Numbers

Zero and any number obtained by repeatedly adding one to it.

Examples: 100, 0, 45645, 32

Negative Numbers

A value less than 0, with a – sign

Examples: -24, -1, -45645, -32

Numbers

Integers

A natural number, a negative number, zero

Examples: 249, 0, - 45645, - 32

Rational Numbers

An integer or the quotient of two integers

Examples: -249, -1, 0, 3/7, -2/5

Natural Numbers

Aha!

642 is 600 + 40 + 2 in **BASE 10**

The **base** of a number determines the number of digits and the value of digit positions

Positional Notation

Continuing with our example…

642 in base 10 *positional notation* is:

$$
\begin{aligned}
6 \times 10^2 &= 6 \times 100 &&= 600 \\
+\ 4 \times 10^1 &= 4 \times 10 &&= 40 \\
+\ 2 \times 10^0 &= 2 \times 1 &&= 2 \qquad = 642 \text{ in base 10}
\end{aligned}
$$

This number is in base 10

The power indicates the position of the number

Positional Notation

R is the base of the number

As a formula:

$$d_n * R^{n-1} + d_{n-1} * R^{n-2} + ... + d_2 * R + d_1$$

n is the number of digits in the number

d is the digit in the i^{th} position in the number

$$642 \text{ is } \quad 6_3 * 10^2 + 4_2 * 10 + 2_1$$

Positional Notation

What if 642 has the base of 13?

$$\begin{aligned} + 6 \times 13^2 &= 6 \times 169 &&= 1014 \\ + 4 \times 13^1 &= 4 \times 13 &&= 52 \\ + 2 \times 13^0 &= 2 \times 1 &&= 2 \\ &= 1068 \text{ in base } 10 \end{aligned}$$

642 in base 13 is equivalent to 1068 in base 10

Binary

Decimal is base 10 and has 10 digits:

0,1,2,3,4,5,6,7,8,9

Binary is base 2 and has 2 digits:

0,1

For a number to exist in a given base, it can only contain the digits in that base, which range from 0 up to (but not including) the base.

What bases can these numbers be in? 122, 198, 178, G1A4

Bases Higher than 10

How are digits in bases higher than 10 represented?

With distinct symbols for 10 and above.

Base 16 has 16 digits:
0,1,2,3,4,5,6,7,8,9,A,B,C,D,E, and F

Converting Octal to Decimal

What is the decimal equivalent of the octal number 642?

$$
\begin{aligned}
& 6 \times 8^2 = 6 \times 64 = 384 \\
+ & 4 \times 8^1 = 4 \times 8 = 32 \\
+ & 2 \times 8^0 = 2 \times 1 = 2 \\
& = 418 \text{ in base 10}
\end{aligned}
$$

Converting Hexadecimal to Decimal

What is the decimal equivalent of the hexadecimal number DEF?

$$
\begin{aligned}
& D \times 16^2 = 13 \times 256 = 3328 \\
+ & E \times 16^1 = 14 \times 16 = 224 \\
+ & F \times 16^0 = 15 \times 1 = 15 \\
& = 3567 \text{ in base } 10
\end{aligned}
$$

Remember, the digits in base 16 are 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

Converting Binary to Decimal

What is the decimal equivalent of the binary number 1101110?

$$
\begin{array}{lllll}
 & 1 \times 2^6 & = 1 \times 64 & = 64 & \\
+ & 1 \times 2^5 & = 1 \times 32 & = 32 & \\
+ & 0 \times 2^4 & = 0 \times 16 & = 0 & \\
+ & 1 \times 2^3 & = 1 \times 8 & = 8 & \\
+ & 1 \times 2^2 & = 1 \times 4 & = 4 & \\
+ & 1 \times 2^1 & = 1 \times 2 & = 2 & \\
+ & 0 \times 2^0 & = 0 \times 1 & = 0 & \\
 & & & & = 110 \text{ in base } 10
\end{array}
$$

Arithmetic in Binary

Remember that there are only 2 digits in binary, 0 and 1

1 + 1 is 0 with a carry

Carry Values

```
  1 0 1 0 1 1 1
+ 1 0 0 1 0 1 1
```

Arithmetic in Binary

Remember that there are only 2 digits in binary, 0 and 1

1 + 1 is 0 with a carry

```
  1 1 1 1 1 1
  1 0 1 0 1 1 1
 +1 0 0 1 0 1 1
 --------------
 1 0 1 0 0 0 1 0
```

Carry Values

Converting Binary to Octal

- Mark groups of *three* (from right)
- Convert each group

10101011	**10**	**101**	**011**
	2	**5**	**3**

10101011 is 253 in base 8

Converting Binary to Hexadecimal

- Mark groups of *four* (from right)
- Convert each group

10101011 **<u>1010</u>** **<u>1011</u>**

A **B**

10101011 is AB in base 16

Converting Decimal to Other Bases

Algorithm for converting number in base 10 to other bases

```
While (the quotient is not zero)
    Divide the decimal number by the new base
    Make the remainder the next digit to the left in the answer
    Replace the original decimal number with the quotient
```

Converting Decimal to Octal

What is 1988 (base 10) in base 8?

Try it!

Converting Decimal to Octal

248	31	3	0
8 | 1988	8 | 248	8 | 31	8 | 3
16	24	24	0
38	08	7	3
32	8		
68	0		
64			
4			

Answer is : **3 7 0 4**

Gates and Logic:

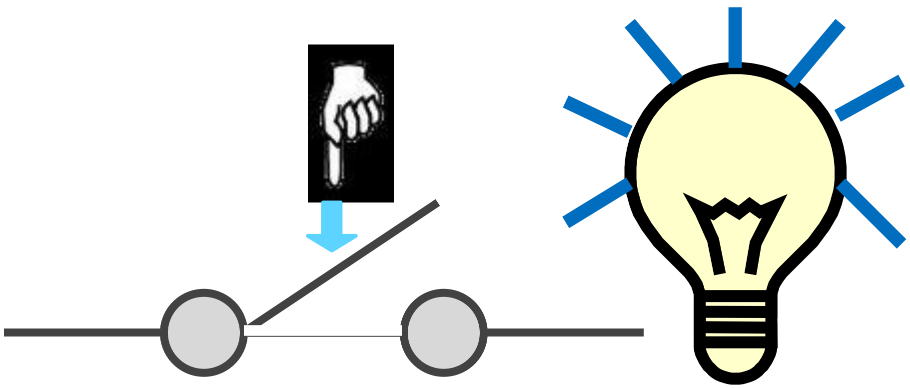
adapted from Hakim Weatherspoon
CS 3410

Cornell University

A switch

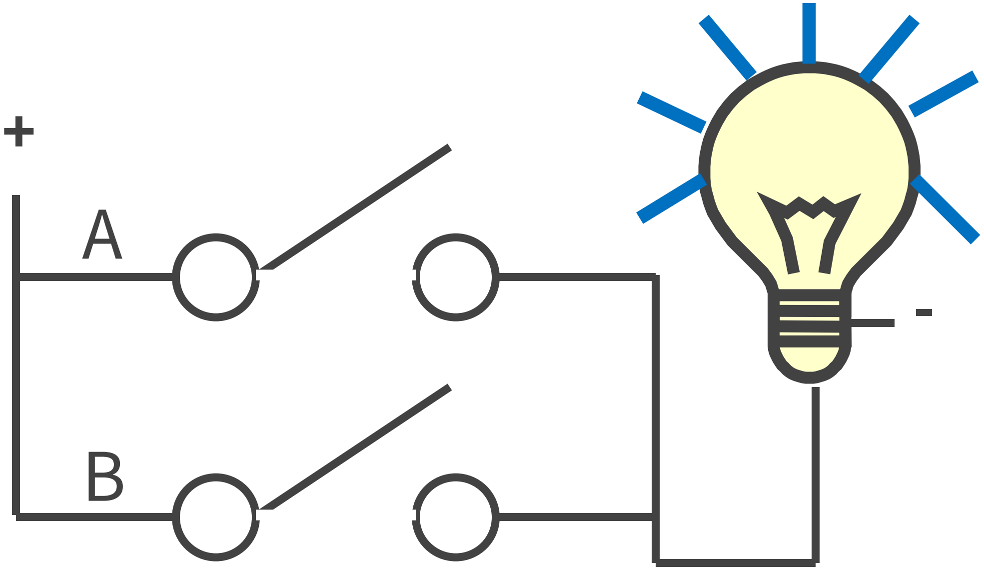
Acts as a *conductor* or *insulator*.

Can be used to build amazing things…



The Bombe used to break the German Enigma machine during World War II

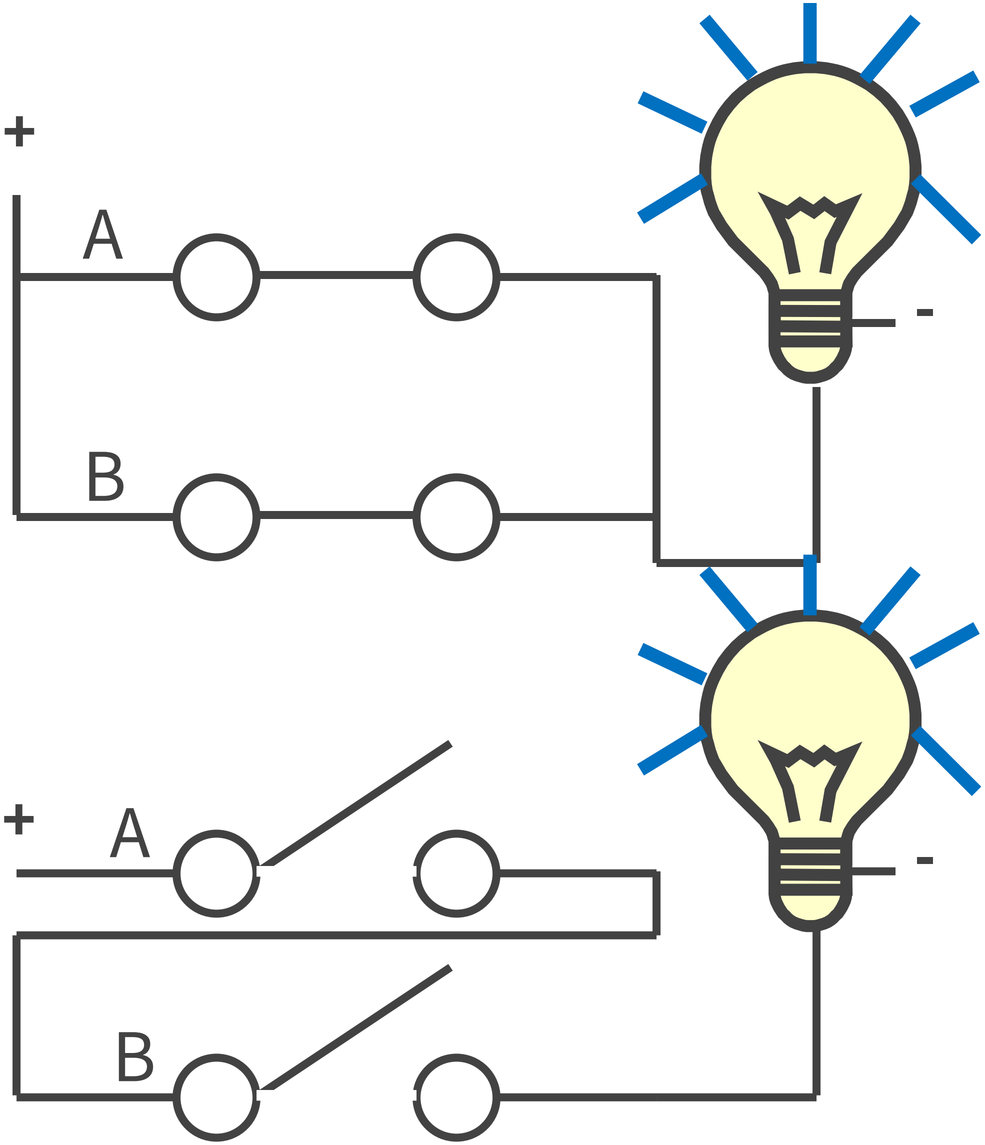
Basic Building Blocks: Switches to Logic Gates



Truth Table

A	B	Light
OFF	OFF	
OFF	ON	
ON	OFF	
ON	ON	

Basic Building Blocks: Switches to Logic Gates

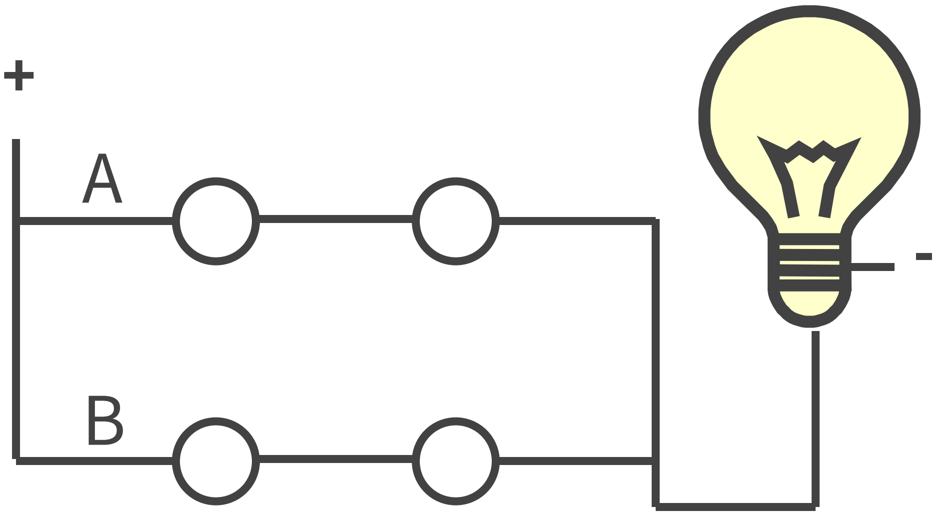


Truth Table

A	B	Light
OFF	OFF	
OFF	ON	
ON	OFF	
ON	ON	

A	B	Light
OFF	OFF	
OFF	ON	
ON	OFF	
ON	ON	

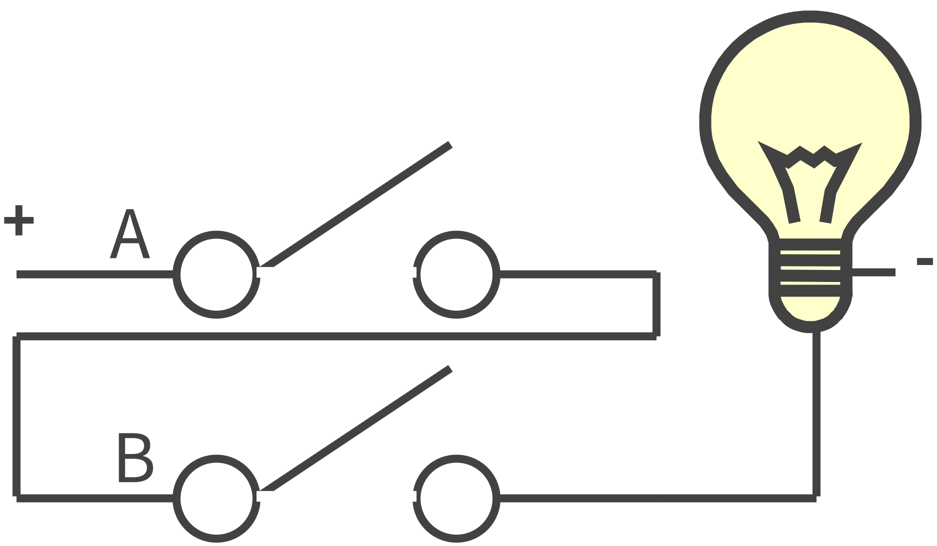
Basic Building Blocks: Switches to Logic Gates



- Either (OR)

Truth Table

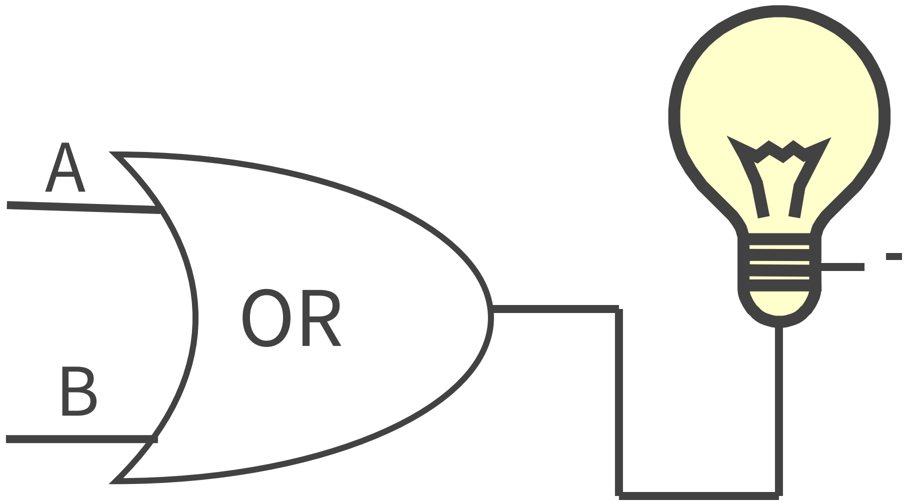
A	B	Light
OFF	OFF	OFF
OFF	ON	ON
ON	OFF	ON
ON	ON	ON



- Both (AND)

A	B	Light
OFF	OFF	OFF
OFF	ON	OFF
ON	OFF	OFF
ON	ON	ON

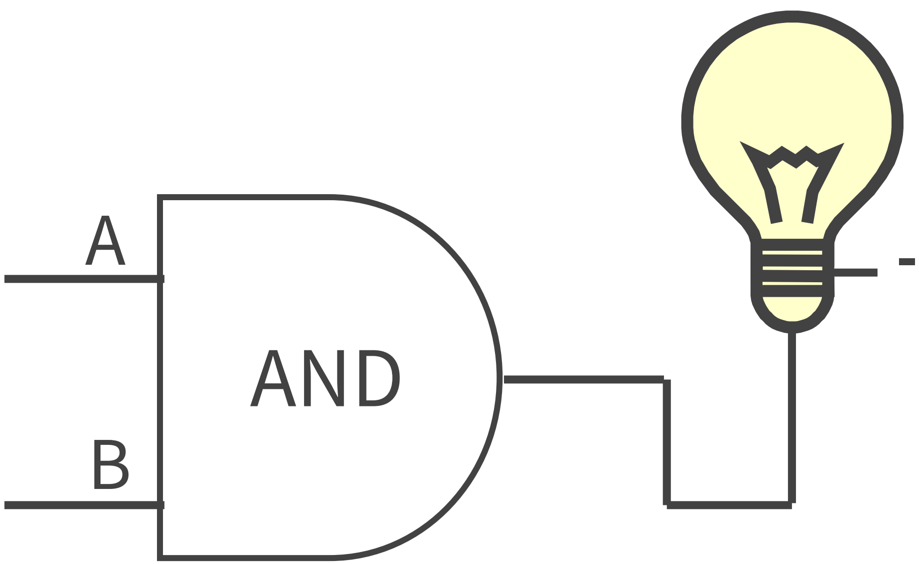
Basic Building Blocks: Switches to Logic Gates



- Either (OR)

Truth Table

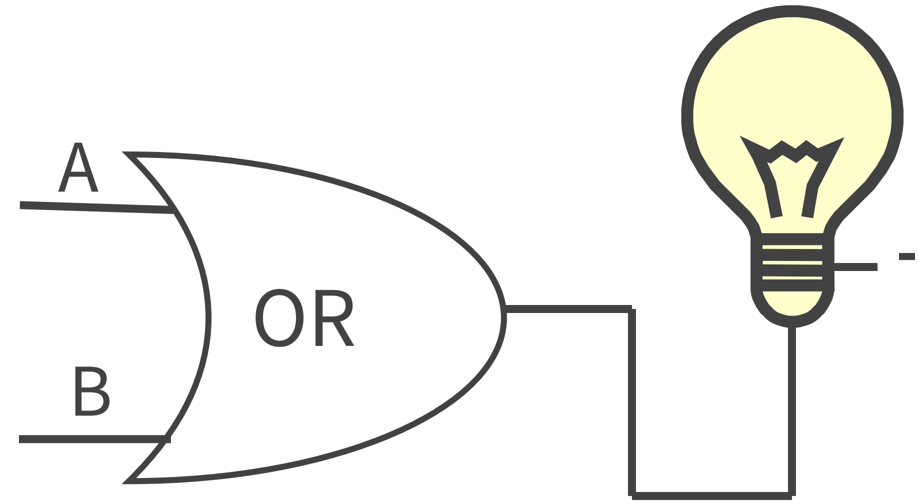
A	B	Light
OFF	OFF	OFF
OFF	ON	ON
ON	OFF	ON
ON	ON	ON



- Both (AND)

A	B	Light
OFF	OFF	OFF
OFF	ON	OFF
ON	OFF	OFF
ON	ON	ON

Basic Building Blocks: Switches to Logic Gates

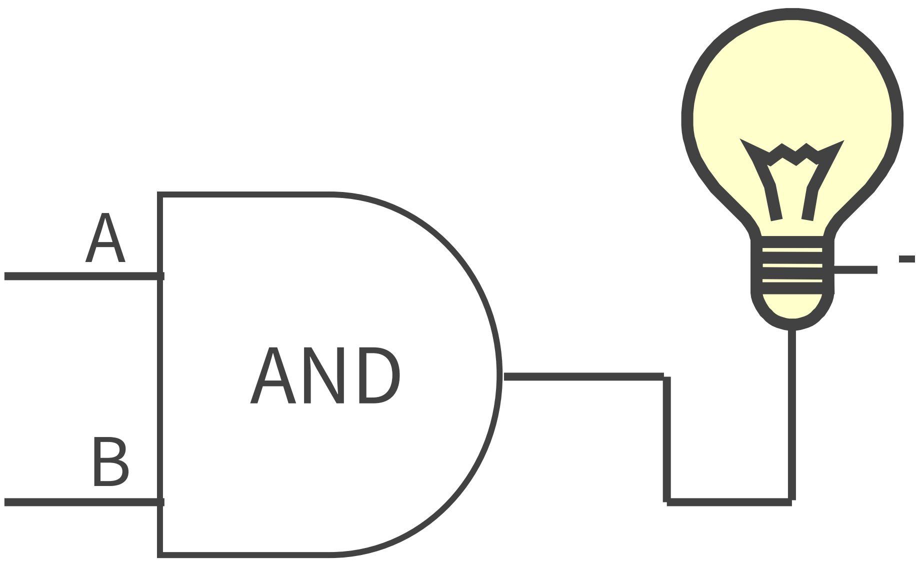


- Either (OR)

Truth Table

A	B	Light
0	0	0
0	1	1
1	0	1
1	1	1

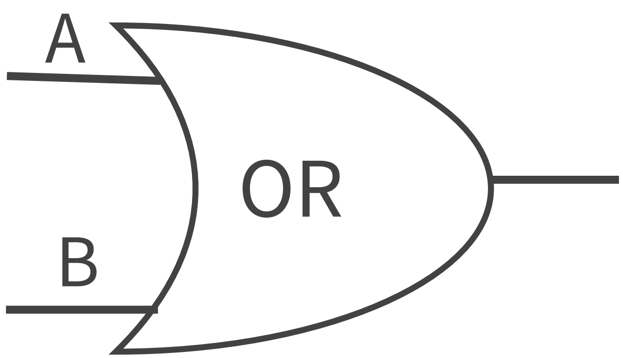
0 = OFF
1 = ON



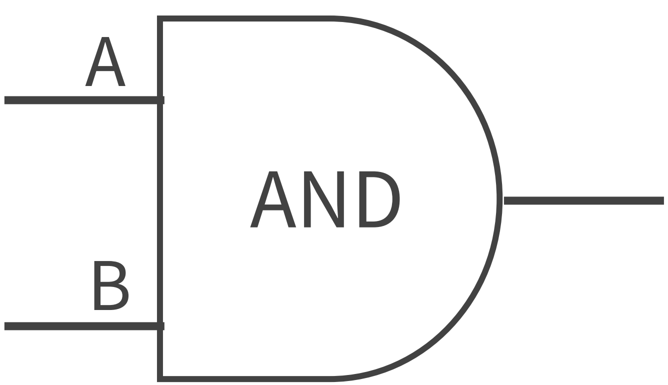
- Both (AND)

A	B	Light
0	0	0
0	1	0
1	0	0
1	1	1

Basic Building Blocks: Switches to Logic Gates



George Boole (1815-1864)



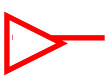
Which Gate is this?

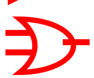
Function:

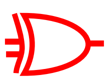
Symbol:

Truth Table:

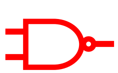
a	b	Out

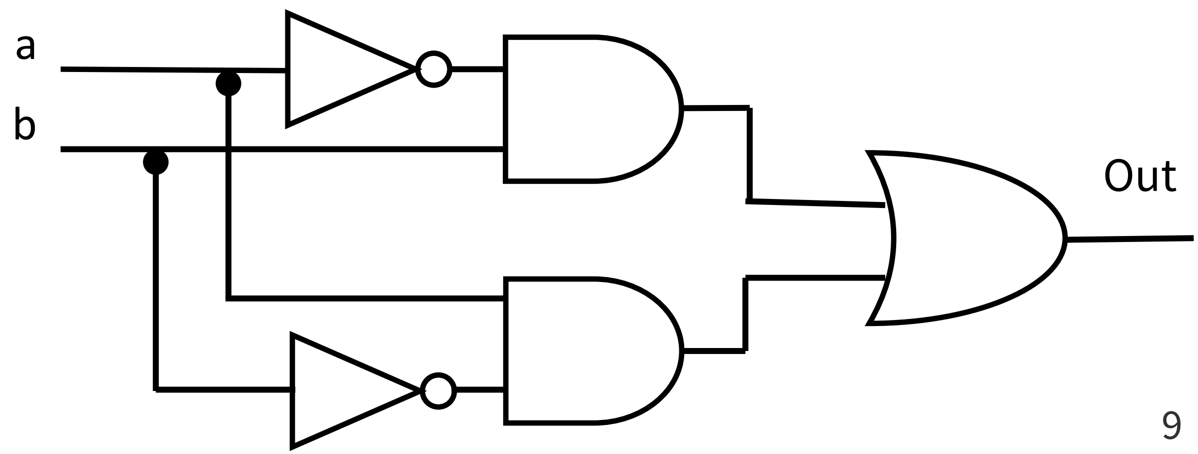
(A) NOT 

(B) OR 

(C) XOR 

(D) AND 

(E) NAND 

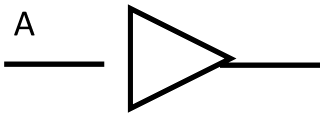


Takeaway

- Binary (two symbols: true and false) is the basis of Logic Design

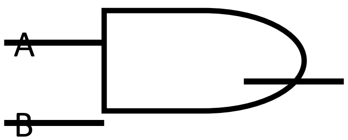
Building Functions: Logic Gates

- NOT:



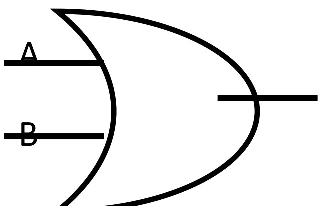
A	Out

- AND:



A	B	Out
0	0	0
0	1	0
1	0	0
1	1	1

- OR:



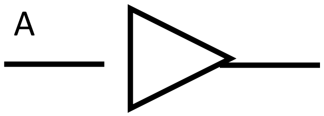
A	B	Out
0	0	0
0	1	1
1	0	1
1	1	1

- Logic Gates
  - digital circuit that either allows a signal to pass through it or not.
  - Used to build logic functions
  - There are seven basic logic gates:

    AND, OR, **NOT**,

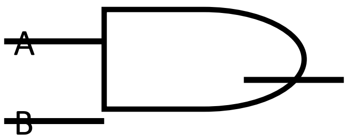
Building Functions: Logic Gates

- NOT:



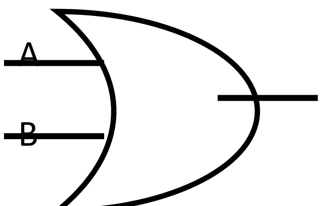
A	Out
0	1
1	0

- AND:



A	B	Out
0	0	0
0	1	0
1	0	0
1	1	1

- OR:



A	B	Out
0	0	0
0	1	1
1	0	1
1	1	1

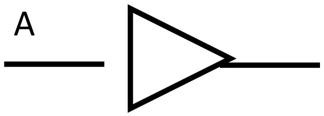
- Logic Gates
  - digital circuit that either allows a signal to pass through it or not.
  - Used to build logic functions
  - There are seven basic logic gates:

    AND, OR, **NOT**,

    NAND (not AND), NOR (not OR), XOR, and XNOR (not XOR) [later]

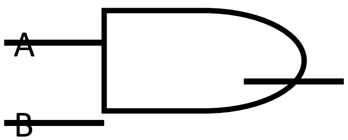
Building Functions: Logic Gates

- NOT:



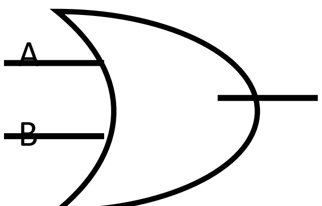
A	Out
0	1
1	0

- AND:



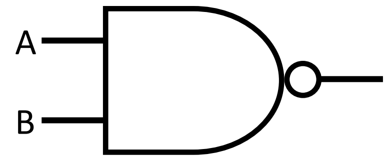
A	B	Out
0	0	0
0	1	0
1	0	0
1	1	1

- OR:



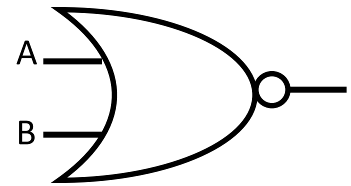
A	B	Out
0	0	0
0	1	1
1	0	1
1	1	1

NAND:



A	B	Out
0	0	1
0	1	1
1	0	1
1	1	0

NOR:



A	B	Out
0	0	1
0	1	0
1	0	0
1	1	0

- Logic Gates
  - digital circuit that either allows a signal to pass through it or not.
  - Used to build logic functions
  - There are seven basic logic gates:

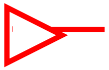
    AND, OR, **NOT**,

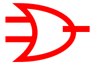
    NAND (not AND), NOR (not OR), XOR, and XNOR (not XOR) [later]

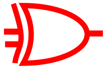
Which Gate is this?

- XOR: out = 1 if a or b is 1, but not both;
- out = 0 otherwise.
- out = 1, only if a = 1 AND b = 0
- OR a = 0 AND b = 1

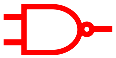
a	b	Out
0	0	0
0	1	1
1	0	1
1	1	0

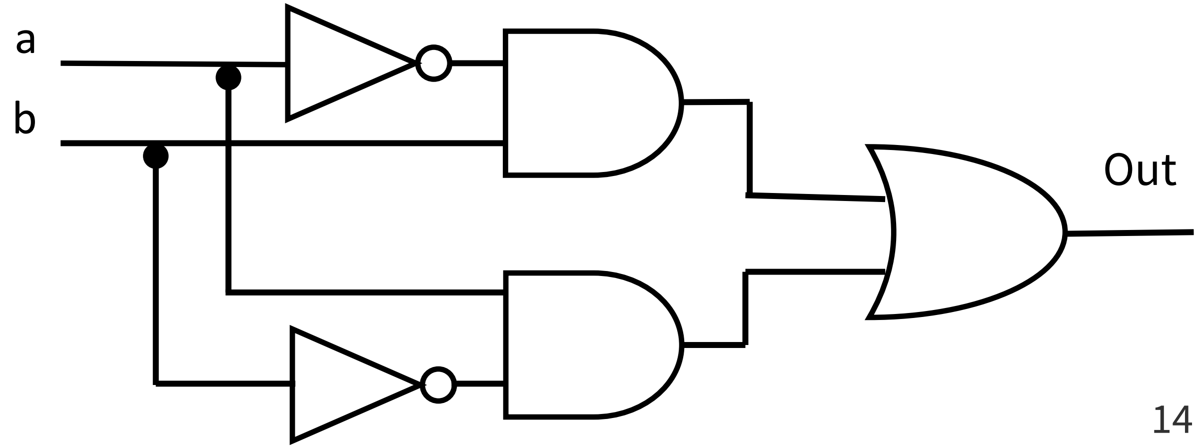
(A) NOT 

(B) OR 

(C) XOR 

(D) AND 

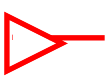
(E) NAND 



Which Gate is this?

- XOR: out = 1 if a or b is 1, but not both;
- out = 0 otherwise.
- out = 1, only if a = 1 AND b = 0
- OR a = 0 AND b = 1

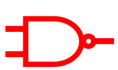
a	b	Out
0	0	0
0	1	1
1	0	1
1	1	0

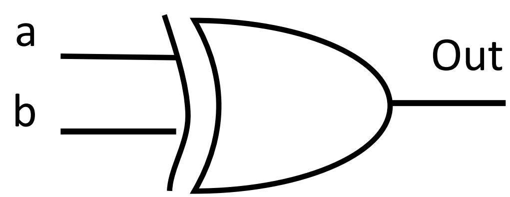
(A) NOT 

(B) OR 

(C) XOR 

(D) AND 

(E) NAND 



Adder Circuits

adapted from Basma Nazar

:Adders

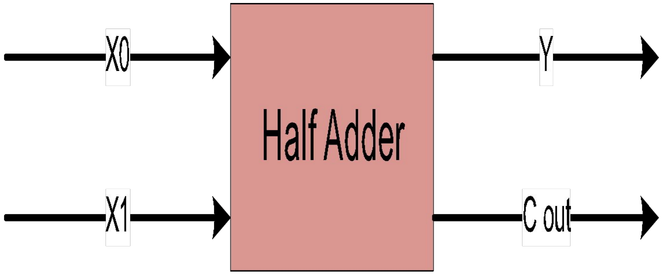
- The **basin** of using logical functions to perform arithmetic operations is **Addition**
- A **binary adder** is a combinational circuit that performs the arithmetic operations of addition with **binary numbers**.
- Subtract two number, or get perform multiplication and division by using addition

Type of binary adder circuits

There are two types of binary adder circuits

1. **Half Adder (HA)**
2. **Full Adder (FA)**

The Half Adder (HA):

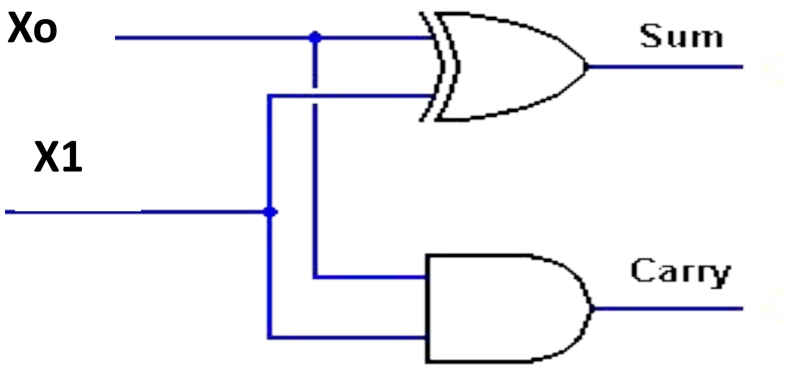


Block diagram

Truth table

x1	x0	y	C out
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

The Half Adder (HA):

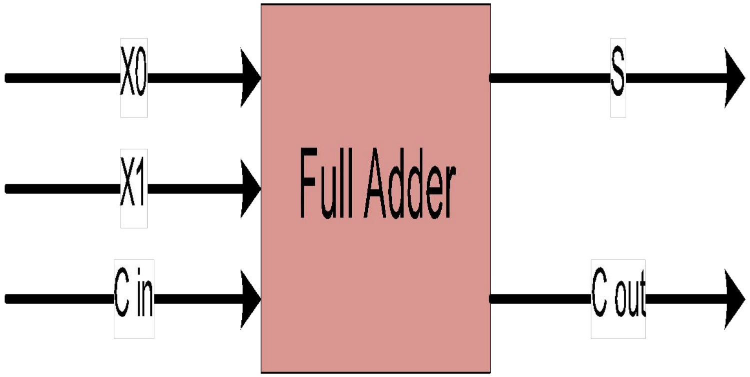


$$Y=\overline{X_0}\,X_1 + X_0\overline{X_1} = X_0 \oplus X_1$$

$$Cout = X_0\,X_1$$

Circuit Symbol **Boolean Expressions**

The Full Adder (FA):

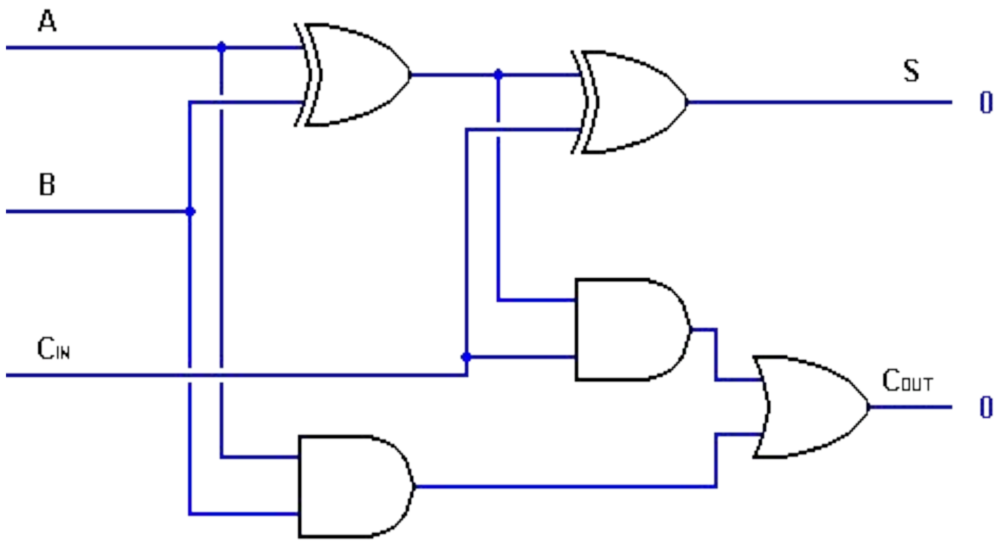


Block diagram

C in	x1	xo	S	C out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Truth table

The Full Adder (FA):



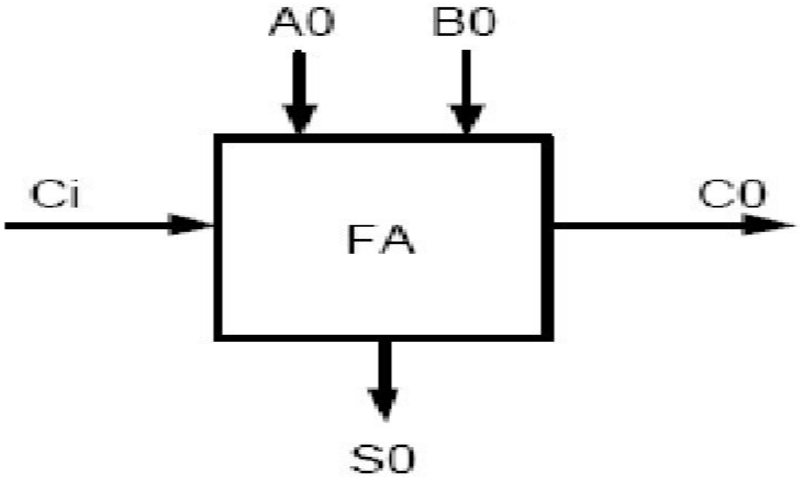
$Y = (X_0 \oplus X_1) \oplus C_{in}$

$C_{out} = C_{in}(X_0 \oplus X_1) + X_0X_1$

Circuit Symbol **Boolean Expressions**

Parallel Adder

A **n-bit** binary adder have **n full adder** units that each take three one-bit inputs: **A**, **B** and carry **Ci** and which generate **sum S** and **carry out Co**,



- Now we can add **two** binary bits together, accounting for a possible carry from the **next lower** order of **magnitude**, and sending a **carry** to the **next higher** order of **magnitude**