ECE213

computer science illuminated

Logic Gates

2x

Dr. Krishan Arora

Assistant Professor and Head

Lovely Professional University

Logic Gates

- Logic gates are the fundamental building blocks of digital systems.The name logic gate is derived from the ability of such a device to make a decision, in the sense that it produces one output level when some combinations of input level are present, and a different output level when other combinations of input levels are present.

Computers

- There are three different, but equally powerful, notational methods for describing the behavior of gates and circuits
  - Boolean expressions
  - logic diagrams
  - truth tables

Truth Table

- **Logic diagram:** a graphical representation of a circuit
  - Each type of gate is represented by a specific graphical symbol
- **Truth table:** defines the function of a gate by listing all possible input combinations that the gate could encounter, and the corresponding output

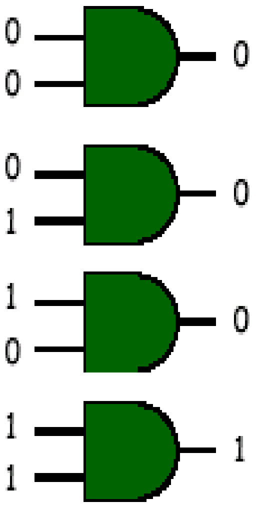
Gates

- Let’s examine the processing of the following
  six types of gates
  - NOT
  - AND
  - OR
  - XOR
  - NAND
  - NOR

Truth table.

- Systematic method to describe Boolean function.
- One row for each possible input combination.
- N inputs $\Rightarrow$ 2^N rows.

AND Truth Table		
x	y	AND
0	0	0
0	1	0
1	0	0
1	1	1



AND

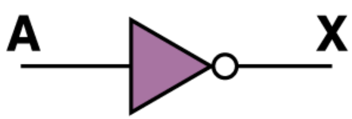
NOT Gate

- A NOT gate accepts one input value and produces one output value

Boolean Expression

$X = A'$

Logic Diagram Symbol



Truth Table

A	X
0	1
1	0

Figure 4.1 Various representations of a NOT gate

NOT Gate

- By definition, if the input value for a NOT gate is 0, the output value is 1, and if the input value is 1, the output is 0
- A NOT gate is sometimes referred to as an *inverter* because it inverts the input value

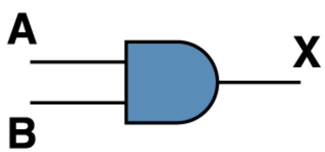
AND Gate

- An AND gate accepts two input signals
- If the two input values for an AND gate are both 1, the output is 1; otherwise, the output is 0

Boolean Expression

$$X = A \cdot B$$

Logic Diagram Symbol



Truth Table

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

Figure 4.2 **Various representations of an AND gate**

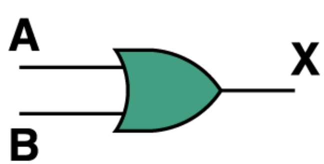
OR Gate

- If the two input values are both 0, the output value is 0; otherwise, the output is 1

Boolean Expression

$$X = A + B$$

Logic Diagram Symbol



Truth Table

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

Figure 4.3 Various representations of a OR gate

XOR Gate

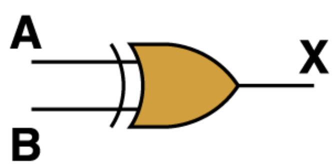
- XOR, or *exclusive* OR, gate
  - An XOR gate produces 0 if its two inputs are the same, and a 1 otherwise
  - Note the difference between the XOR gate and the OR gate; they differ only in one input situation
  - When both input signals are 1, the OR gate produces a 1 and the XOR produces a 0

XOR Gate

Boolean Expression

$$X = A \oplus B$$

Logic Diagram Symbol



Truth Table

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

Figure 4.4 **Various representations of an XOR gate**

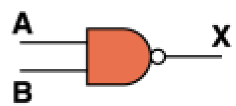
NAND and NOR Gates

- The NAND and NOR gates are essentially the opposite of the AND and OR gates, respectively

Boolean Expression

$$X = (A \cdot B)'$$

Logic Diagram Symbol



Truth Table

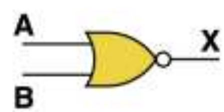
A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

Figure 4.5 **Various representations of a NAND gate**

Boolean Expression

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

Logic Diagram Symbol



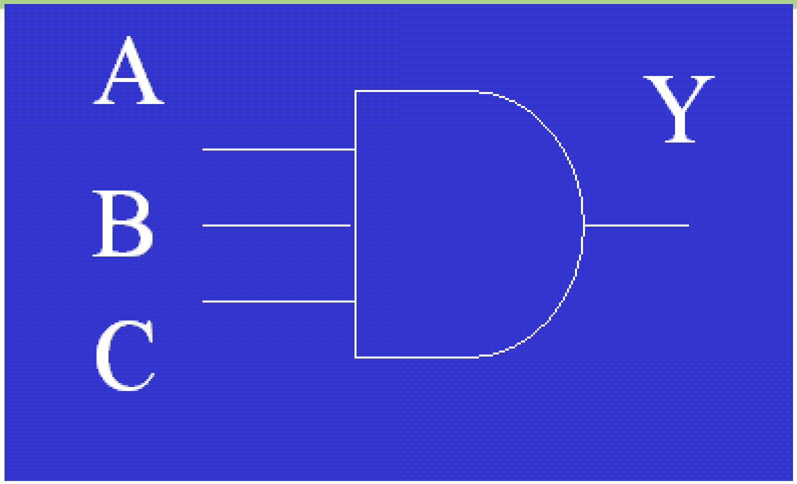
Truth Table

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

Figure 4.6 **Various representations of a NOR gate**

3-Input And gate

A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1



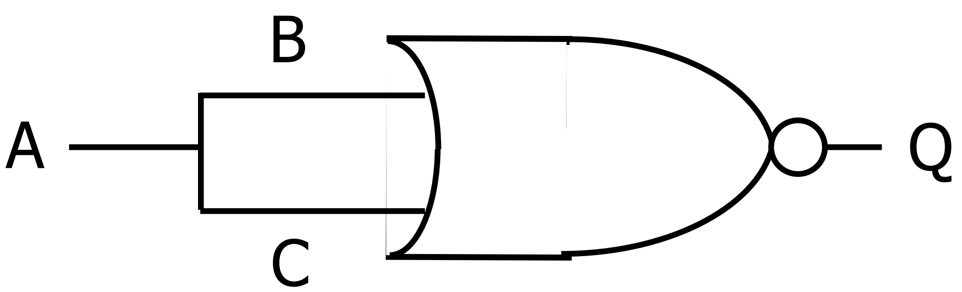
$$Y = A \cdot B \cdot C$$

Circuits

- Two general categories
  - In a **combinational circuit**, the input values explicitly determine the output
  - In a **sequential circuit**, the output is a function of the input values as well as the existing state of the circuit
- As with gates, we can describe the operations of entire circuits using three notations
  - Boolean expressions
  - logic diagrams
  - truth tables

Universal Gates

How to use NOR gate to build a NOT gate?



Truth Table

A	B	C	Q
0	0	0	1
1	1	1	0

Hint!

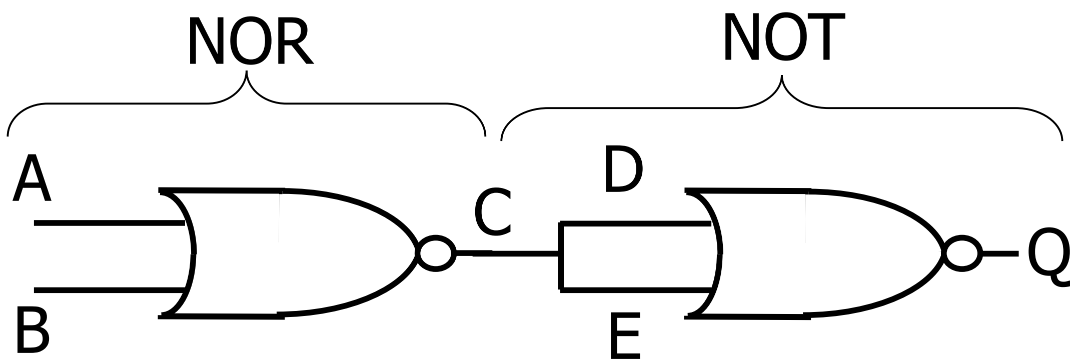
Link inputs B & C together (to a same source).

When A = 0, B = C = A = 0

When A = 1, B = C = A = 1

Universal Gates

How to use ***NOR*** *gates to build an OR gate?*



Truth Table

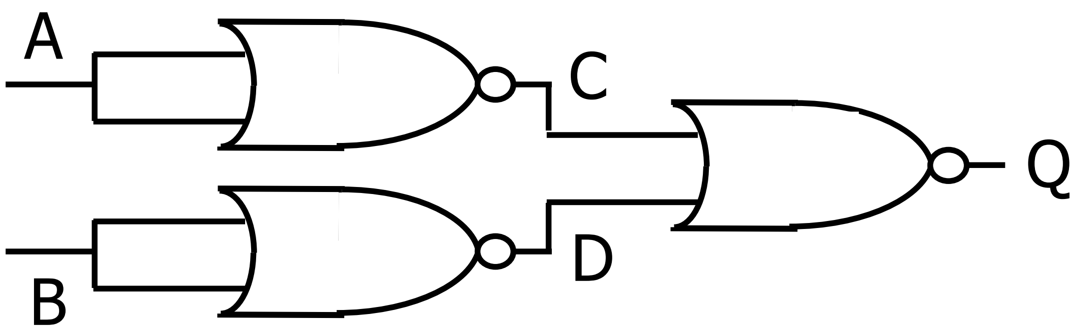
A	B	C	D	E	Q
0	0	1	1	1	0
0	1	0	0	0	1
1	0	0	0	0	1
1	1	0	0	0	1

Hint 1 : Use 2 NOR gates

Hint 2 : From a NOR gate, build a NOT gate

Hint 3 : Put this "NOT" gate after a NOR gate

How to use NOR gates to build an AND gate?



Truth Table

A	B	C	D	Q
0	0	1	1	0
0	1	1	0	0
1	0	0	1	0
1	1	0	0	1

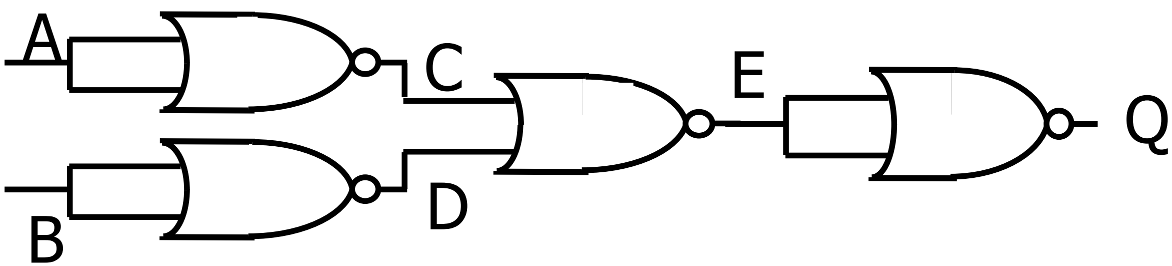
Hint 1 : Use 3 NOR gates

Hint 2 : From 2 NOR gates, build 2 NOT gates

Hint 3 : Each "NOT" gate is an input to the 3rd NOR gate

Universal Gates

*How to use **NOR** gates to build a NAND gate?*



Truth Table

A	B	C	D	E	Q
0	0	1	1	0	1
0	1	1	0	0	1
1	0	0	1	0	1
1	1	0	0	1	0

Hint 1 : Use 4 NOR gates

Hint 2 : Use 3 NOR gates to build a NAND gate

(previous lesson)

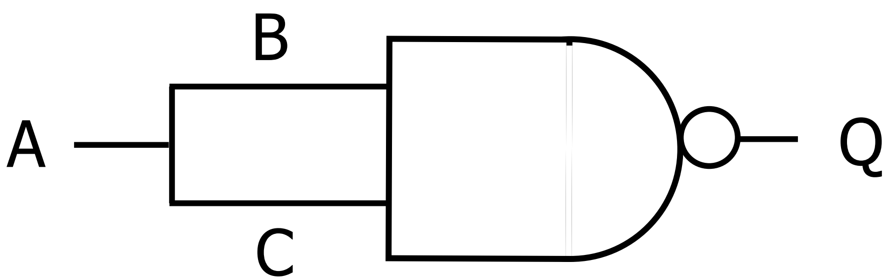
Hint 3 : Use the 4th NOR gate to build a NOT gate

Hint 4 : Insert “NOT” gate after “NAND” gate

Hint 5 : NOT-NAND = AND

Universal Gates

How to use NAND gates to build a NOT gate?



Truth Table

A	B	C	Q
0	0	0	1
1	1	1	0

Hint!

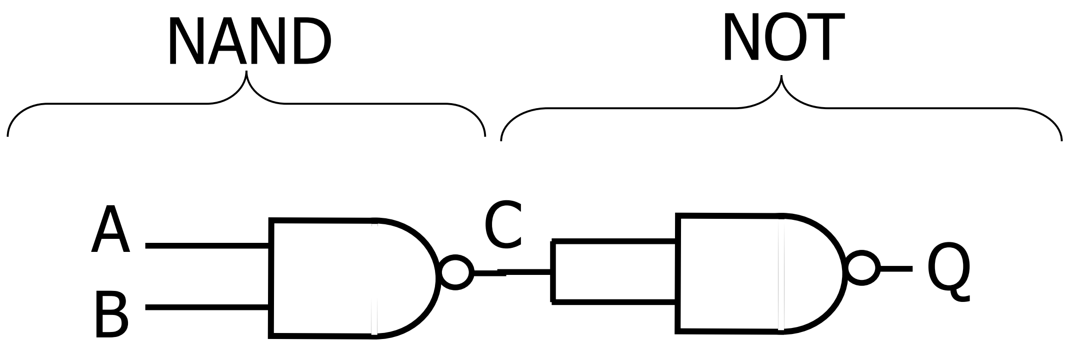
Link inputs B & C together (to a same source).

When A = 0, B = C = A = 0

When A = 1, B = C = A = 1

Universal Gates

*How to use **NAND** gates to build an AND gate?*



Truth Table

A	B	C	Q
0	0	1	0
0	1	1	0
1	0	1	0
1	1	0	1

Hint 1 : Use 2 NAND gates

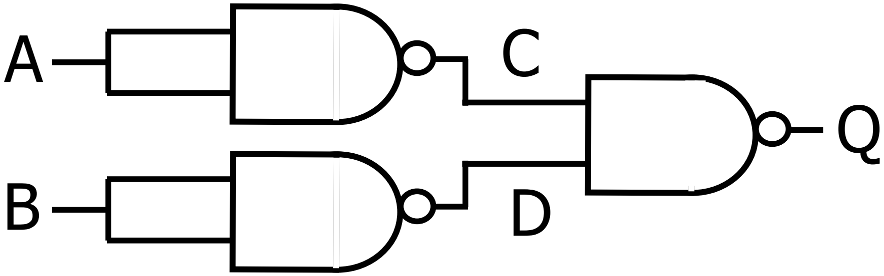
Hint 2 : From a NAND gate, build a NOT gate

Hint 3 : Put this “NOT” gate after a NAND gate

Hint 4 : NOT-NAND = AND

Universal Gates

*How to use **NAND** gates to build an OR gate?*



Truth Table

A	B	C	D	Q
0	0	1	1	0
0	1	1	0	1
1	0	0	1	1
1	1	0	0	1

Hint 1 : Use 3 NAND gates

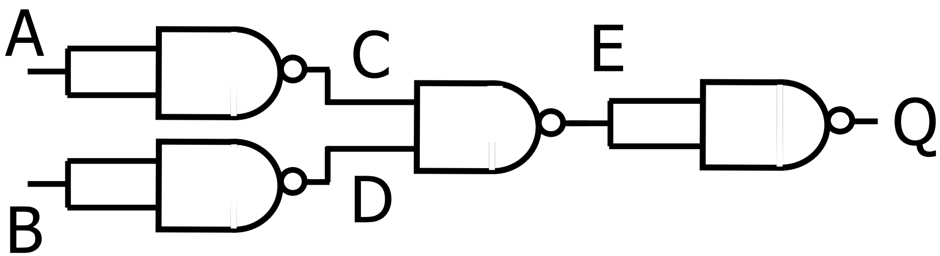
Hint 2 : Use 2 NAND gates to build 2 NOT gates

Hint 3 : Put the 3rd NAND gate after the 2 "NOT" gates

Logic Gates

Universal Gates

How to use ***NAND*** *gates to build a NOR gate?*



Truth Table

A	B	C	D	E	Q
0	0	1	1	0	1
0	1	1	0	1	0
1	0	0	1	1	0
1	1	0	0	1	0

Hint 1 : Use 4 NAND gates

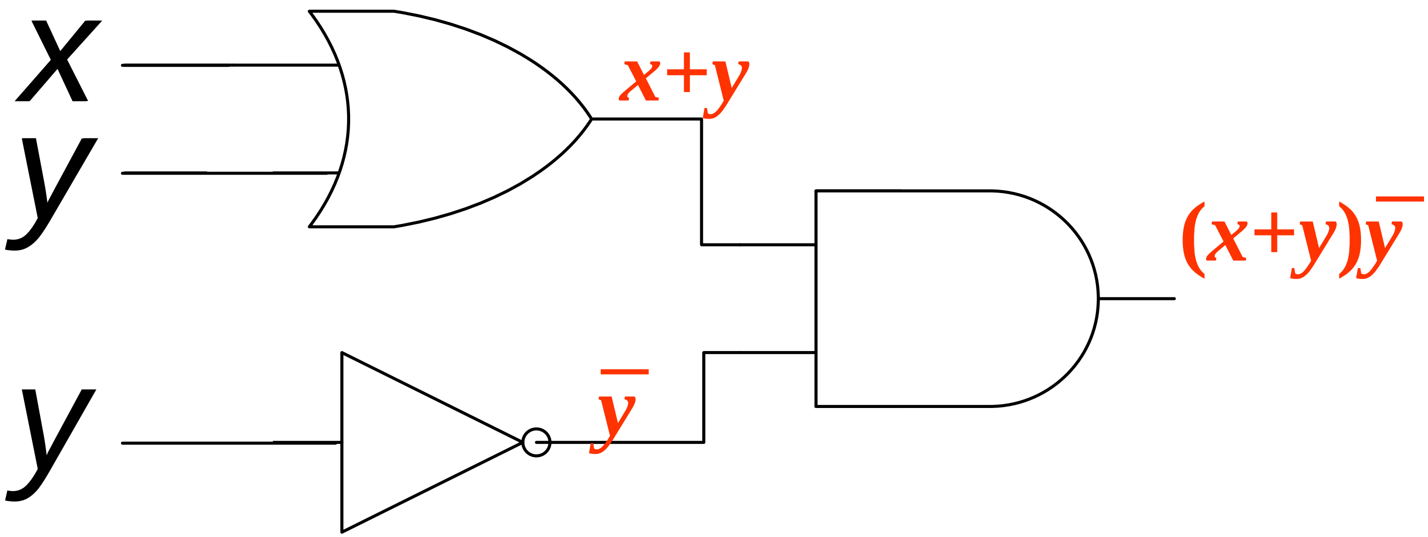
Hint 2 : Use 3 NAND gates to build an OR gate

Hint 3 : Use a NOR gate to build a NOT gate

Hint 4 : Put the “NOT” gate after “OR” gate

Converting between circuits and equations

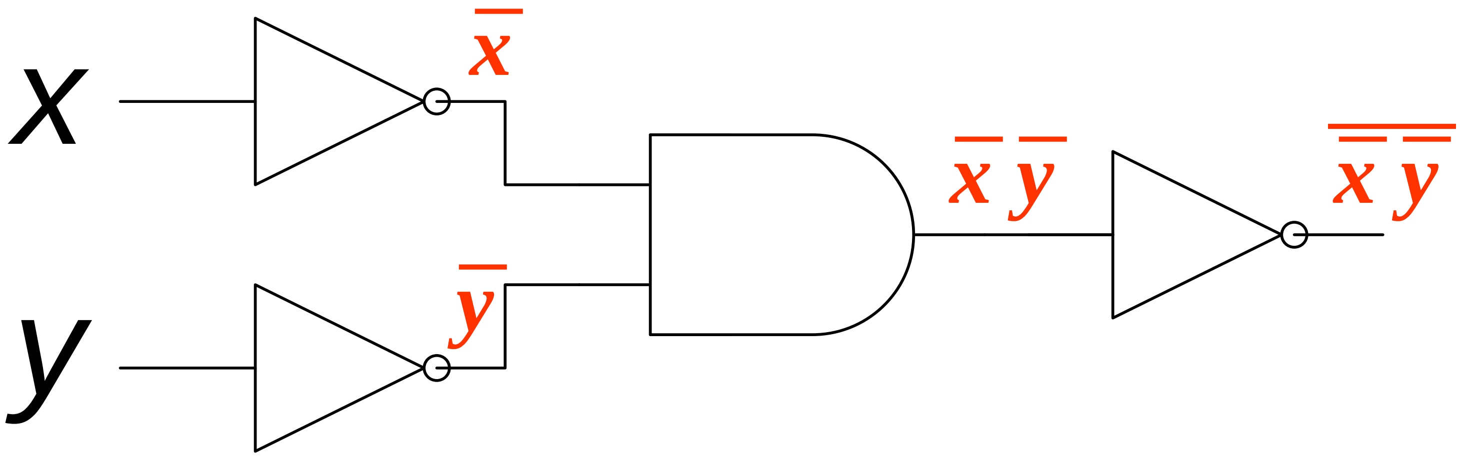
- Find the output of the following circuit



- Answer: $(x+y)\overline{y}$

Converting between circuits and equations

- Find the output of the following circuit

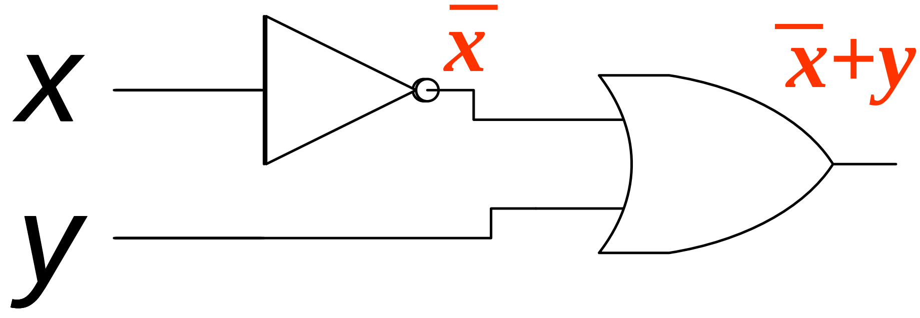


- Answer: $\overline{\overline{x}\,\overline{y}}$

Converting between circuits and equations

- Write the circuits for the following Boolean algebraic expressions

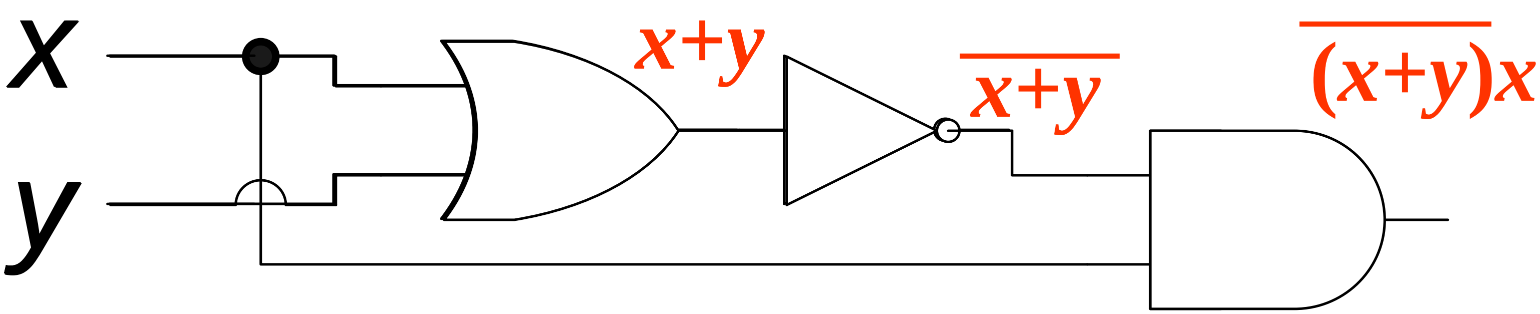
a) $\overline{x}+y$



Converting between circuits and equations

- Write the circuits for the following Boolean algebraic expressions

b) $(x+y)x$



Writing X-OR using AND/OR/NOT

- $x \oplus y \equiv (x + y)(\overline{xy})$

x	y	$x \oplus y$
1	1	0
1	0	1
0	1	1
0	0	0