Lecture 4 (E-Learning)

Boolean Logic & Logic Gates

BOOLEAN LOGIC

Boolean Logic

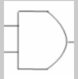
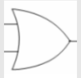
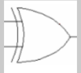
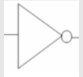
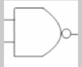
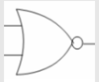
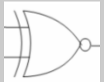
- Boolean logic forms the basis of digital logic
- A boolean logic statement (or signal) is either
  - True (1)
  - False (0)
- Boolean operators act on these statements to form other statements
- Examples of boolean operators:

NOT	-
AND	NAND
OR	NOR
XOR	XNOR

- In digital electronics, boolean operators are implemented as logic gates

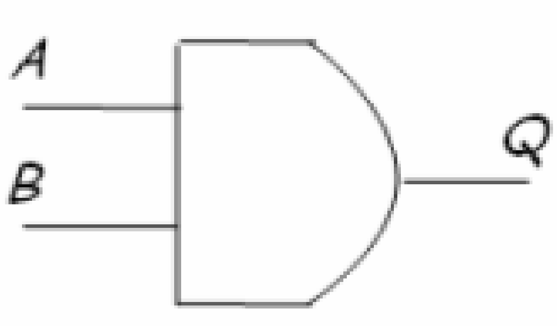
Logic Gates

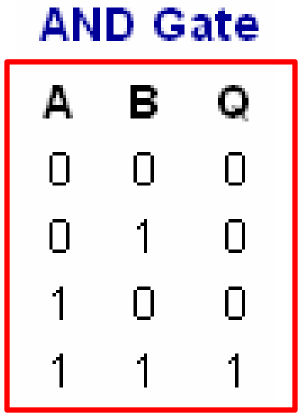
- Below are some common boolean operators and their logic gate symbols. The next few slides will give more details about each boolean operator.

Boolean Operator	Logic Gate Symbol
AND	
OR	
XOR	
NOT	
NAND	
NOR	
XNOR	

Logic Gates: AND

- The output (Q) of an **AND** gate is True (1) only if ALL its inputs are True (1)
  - In the following diagram, Q is True (1) only if A **AND** B are True (1)



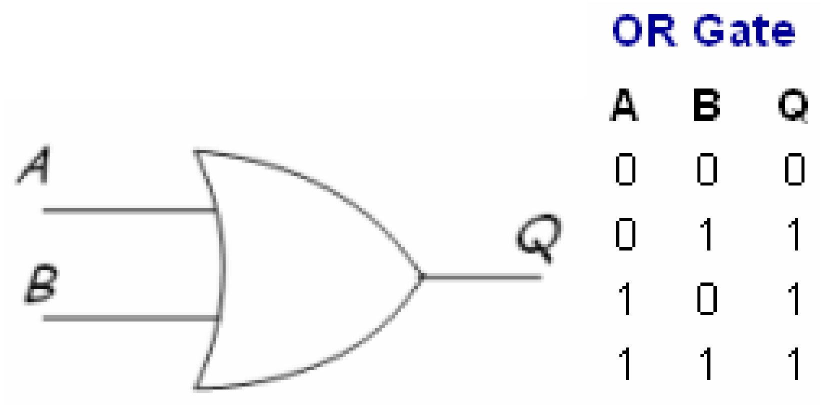

AND Gate

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

This way of showing logic inputs and outputs is called a Truth Table

Logic Gates: OR

- The output (Q) of an **OR** gate is True (1) if ANY of its inputs are True (1)
  - Q is True (1) if either A **OR** B are True (1)

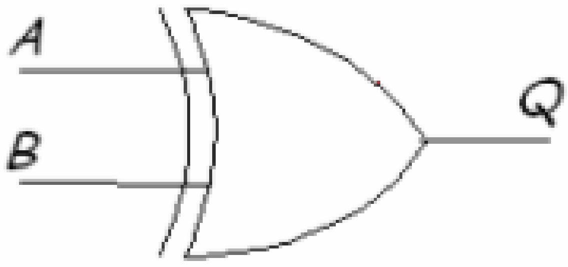


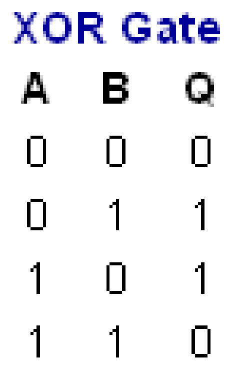
OR Gate

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

Logic Gates: XOR

- XOR is an abbreviation of e**X**clusive **OR**
- The output (Q) of an **XOR** gate is True (1) only if its inputs are different
  - Q is True (1) only if A **AND** B are different [one is True (1) and one is False (0)]

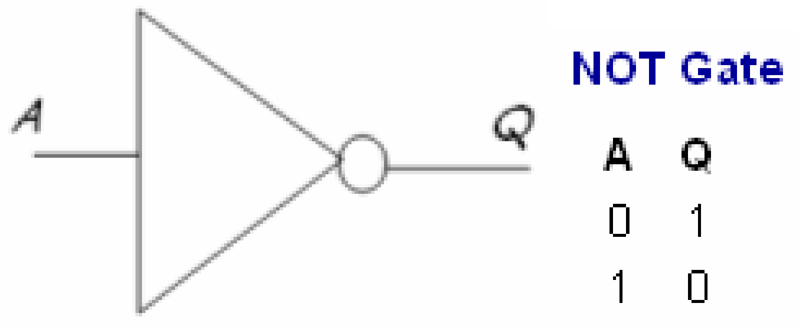



XOR Gate

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

Logic Gates: NOT

- The output (Q) of a **NOT** gate inverts its input (A)
  - True (1) becomes False (0)
  - False (0) becomes True (1)

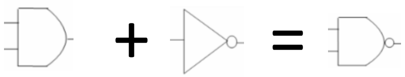


NOT Gate

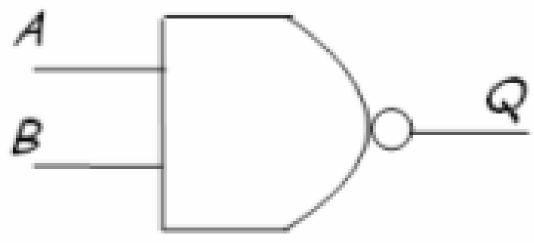
A	Q
0	1
1	0

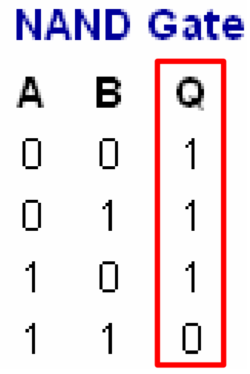
Logic Gates: NAND

- A **NAND** gate combines an **AND** gate with a **NOT** gate



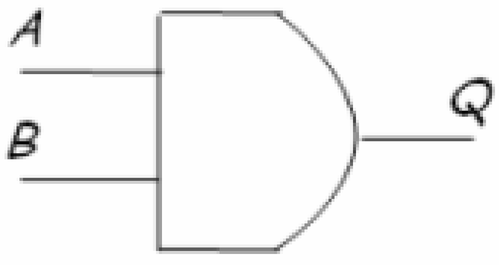
- The output (Q) of a **NAND** gate is simply the inversion of an **AND** gate

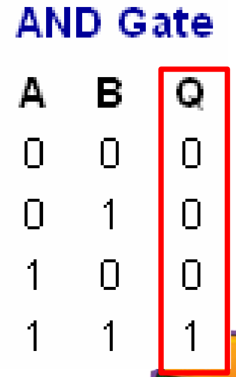




NAND Gate

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



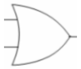
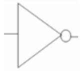
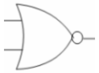


AND Gate

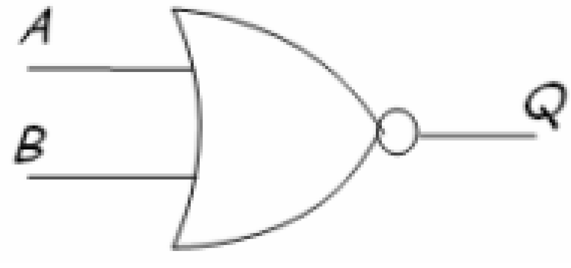
A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1

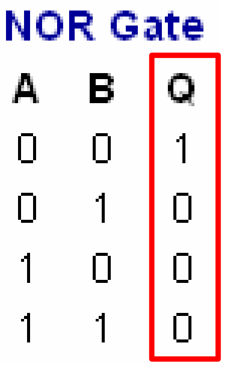
Logic Gates: NOR

- A **NOR** gate combines an **OR** gate with a **NOT** gate

 +  = 

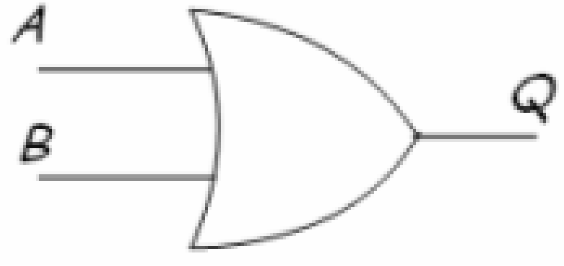
- The output (Q) of a **NOR** gate is simply the inversion of an **OR** gate

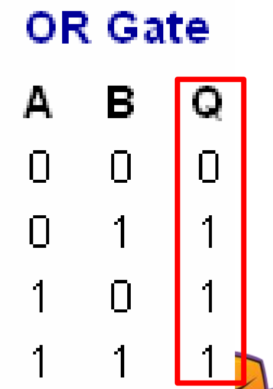




NOR Gate

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



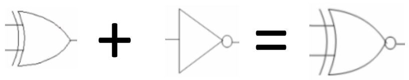


OR Gate

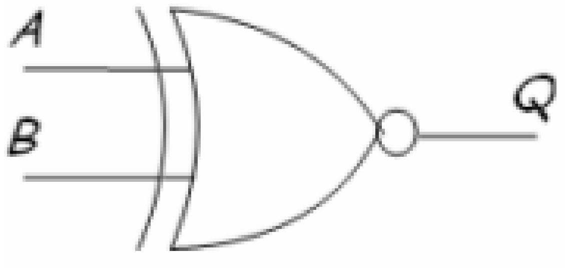
A	B	Q
0	0	0
0	1	1
1	0	1
1	1	1

Logic Gates: XNOR

- A **XNOR** gate combines an **XOR** gate with a **NOT** gate

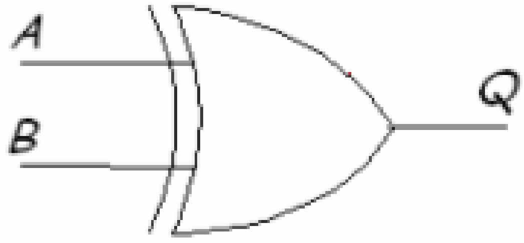


- The output (Q) of a **XNOR** gate is simply the inversion of an **XOR** gate



XNOR Gate

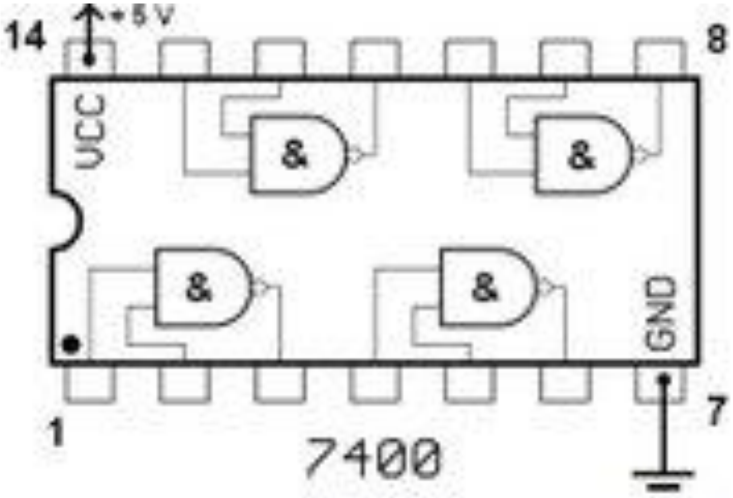
A	B	Q
0	0	1
0	1	0
1	0	0
1	1	1



XOR Gate

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

Example – Logic Gates on a Chip



Example on the Use of Boolean Logic

- Any problem can be split up into small pieces represented by Boolean Logic

Vending machines operate using Boolean logic. They exist in one of two states: either the full amount of change for a soda has been deposited (paid), or it has not been met (unpaid).

Another way Boolean logic is used:
Whether the coin box is full, how many cans remaining, is there a change to be given?

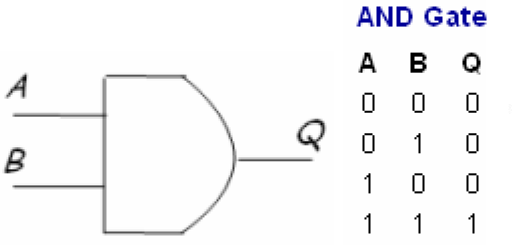
Example on the Use of Boolean Logic

- Vending Machine example:
  - Let’s design our vending machine so that a drink is only dispensed if both input A **and** input B are True (1)

Inputs
Full amount of coins is deposited (Input A)
Drink is selected (Input B)

  - We can use the boolean operator **AND** to form the required logic to dispense the drink

Outputs	Boolean Logic
Dispense drink (Output Q)	Input A **AND** Input B



AND Gate

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

Boolean Logic in Games

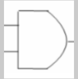
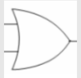
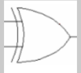
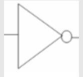
- Games are complex combinations of multiple Boolean conditions
  - – if (HP > 0) state = alive;
  - – if (HP == 0) state = KO;

Lecture 5

Boolean Logic & Logic Gates

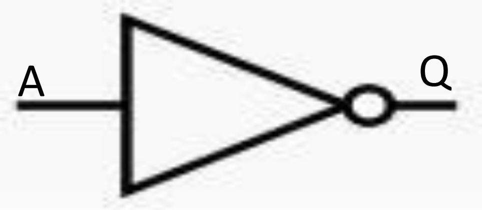
BITWISE OPERATIONS

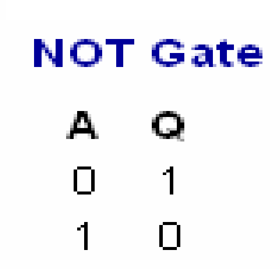
Boolean Symbols

Boolean Operator	Logic Gate Symbol	Bitwise Operator Symbols
AND		&
OR		|
XOR		^
NOT		~

Bitwise Operations

- Bitwise operations act on **individual** bits in a binary number.
- Follow rules of logic gates





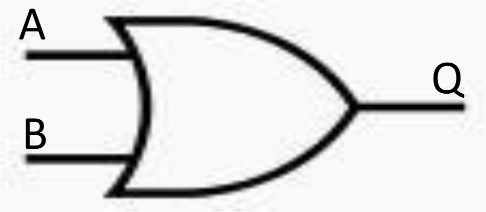
NOT Gate

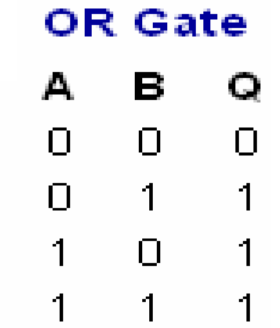
A	Q
0	1
1	0

NOT 0111
= 1000

Each individual bit is inverted

~ 0111 = 1000





OR Gate

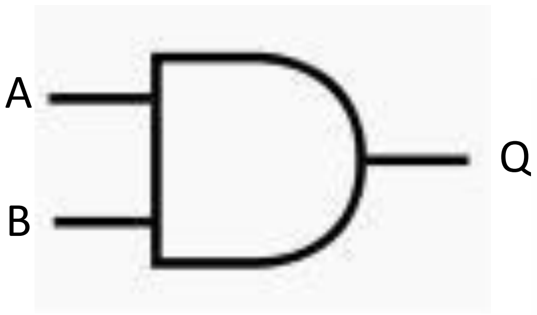
A	B	Q
0	0	0
0	1	1
1	0	1
1	1	1

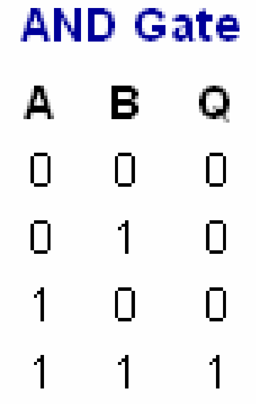
0101
OR 0111
= 0111

Individual bits in the same position are OR'd with each other

0101 | 0111 = 0111

Bitwise Operations





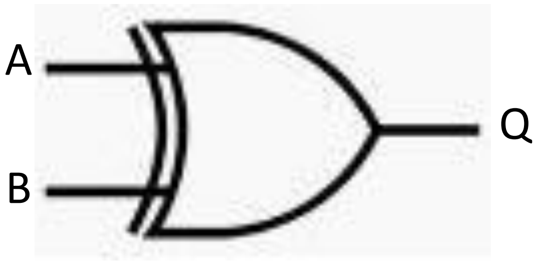
AND Gate

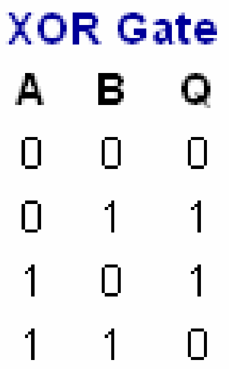
A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1

```
AND  0101
     0011
   = 0001
```

Individual bits in the same position are AND'd with each other

0101 & 0011 = 0001





XOR Gate

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

```
XOR  0101
     0011
   = 0110
```

Individual bits in the same position are XOR'd with each other

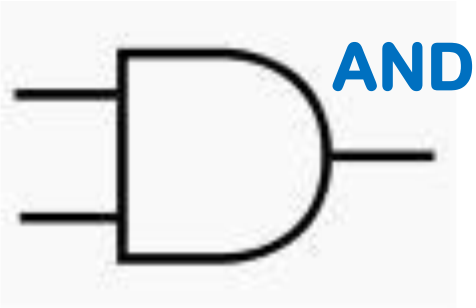
0101 ^ 0011 = 0110

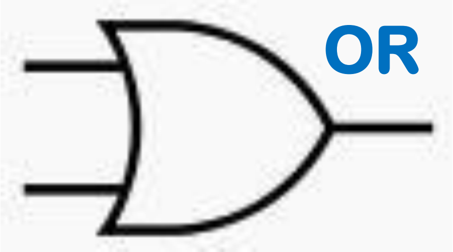
Lecture 5

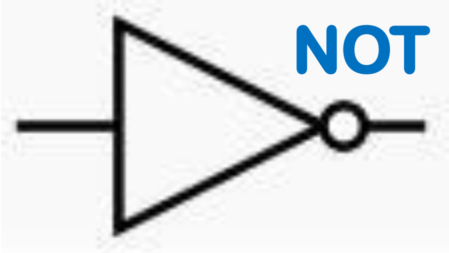
Boolean Logic & Logic Gates

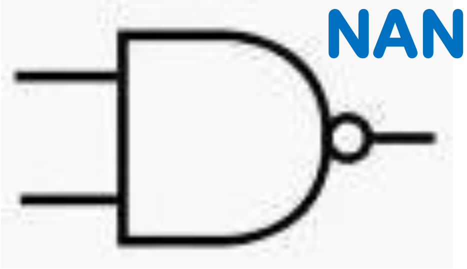
COMBINATIONAL LOGIC

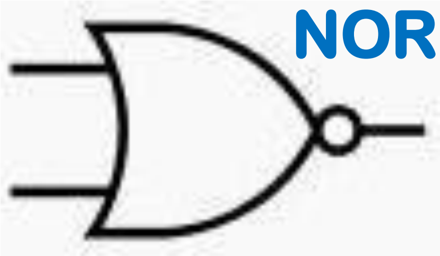
Remember These Logic Gates?

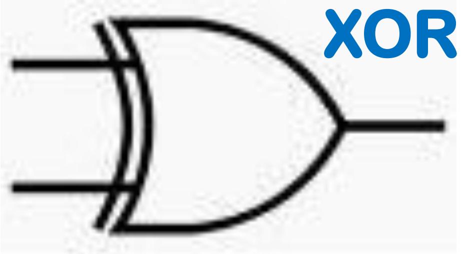

AND

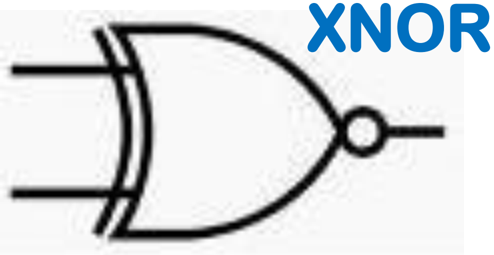

OR


NOT


NAND

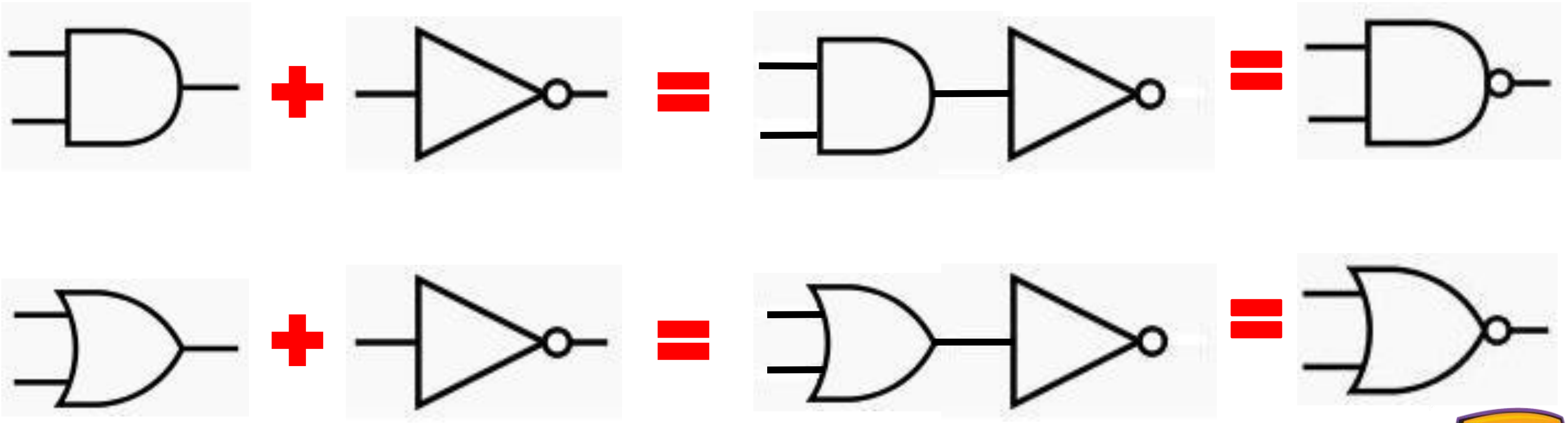

NOR


XOR


XNOR

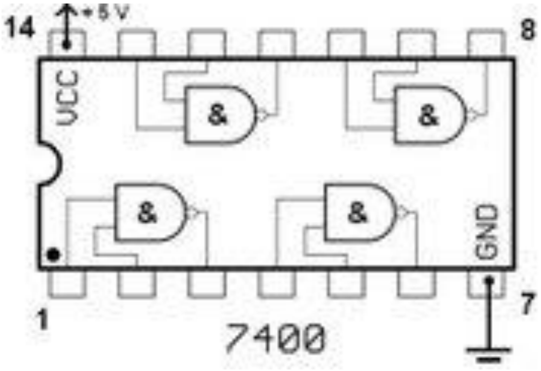
Combinational Logic

- These gates can be combined into any order and in any amount to create more complex functions
- For example:



NAND Gate

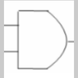
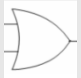
- NAND gate is special
  - Functionally complete
  - Able to construct any other type of gates by just combining NAND gates



  - Ability is very useful in digital logic designs
    - Saves on cost
    - Saves on complexity

  - More on this in future

Boolean Symbols

Boolean Operator	Logic Gate Symbol	Logic Symbols
AND		∧ or .
OR		∨ or +

Combinational Logic

- Example: Automobile Warning

Buzzer will activate if headlights are on AND driver's door is open, OR if key in ignition AND door is opened.

Combinational Logic

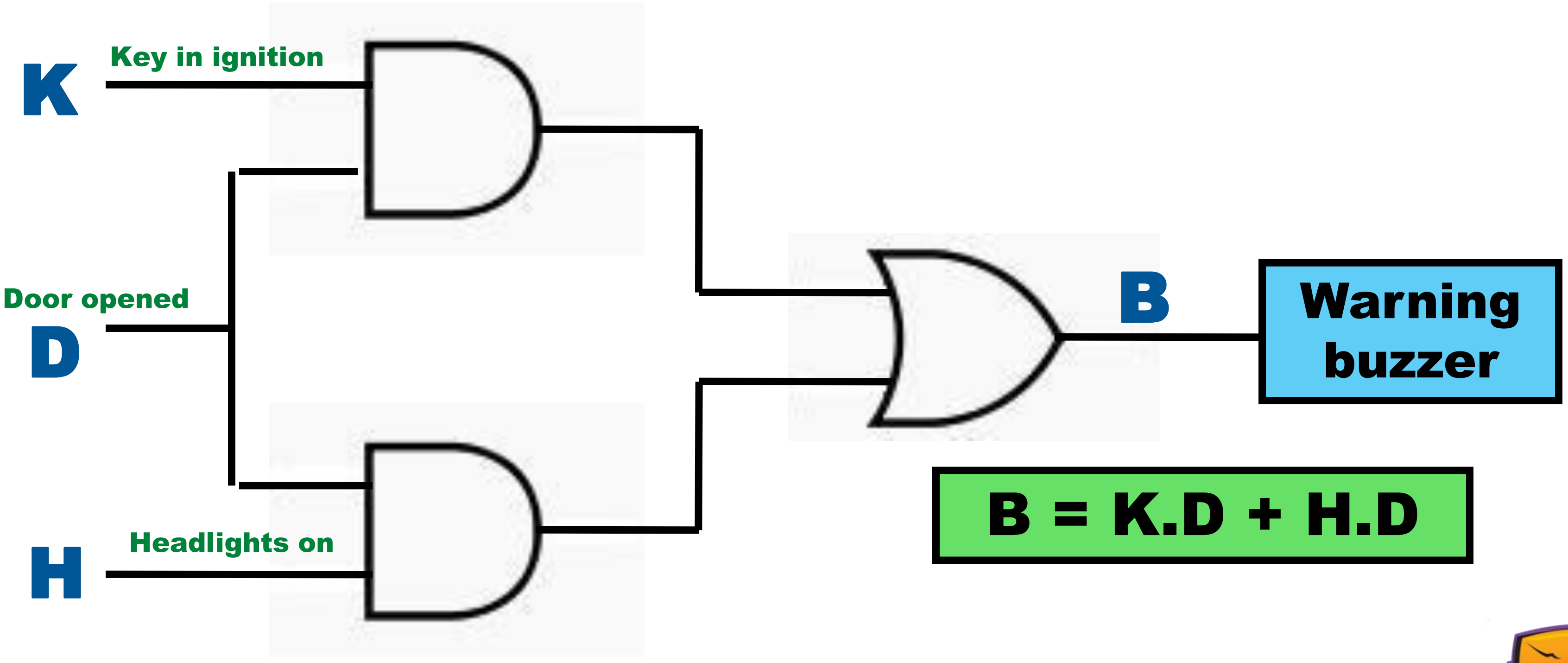
- Example: Automobile Warning
  - B: Buzzer, H: Headlights, D: Door, K: Key
  - Boolean Equation: B = K AND D OR H AND D

B = KD + HD

B is TRUE if K and D are TRUE, or if H and D are TRUE

Combinational Logic

- Example: Automobile Warning
  - Logic Circuit



$B = K.D + H.D$

Combinational Logic

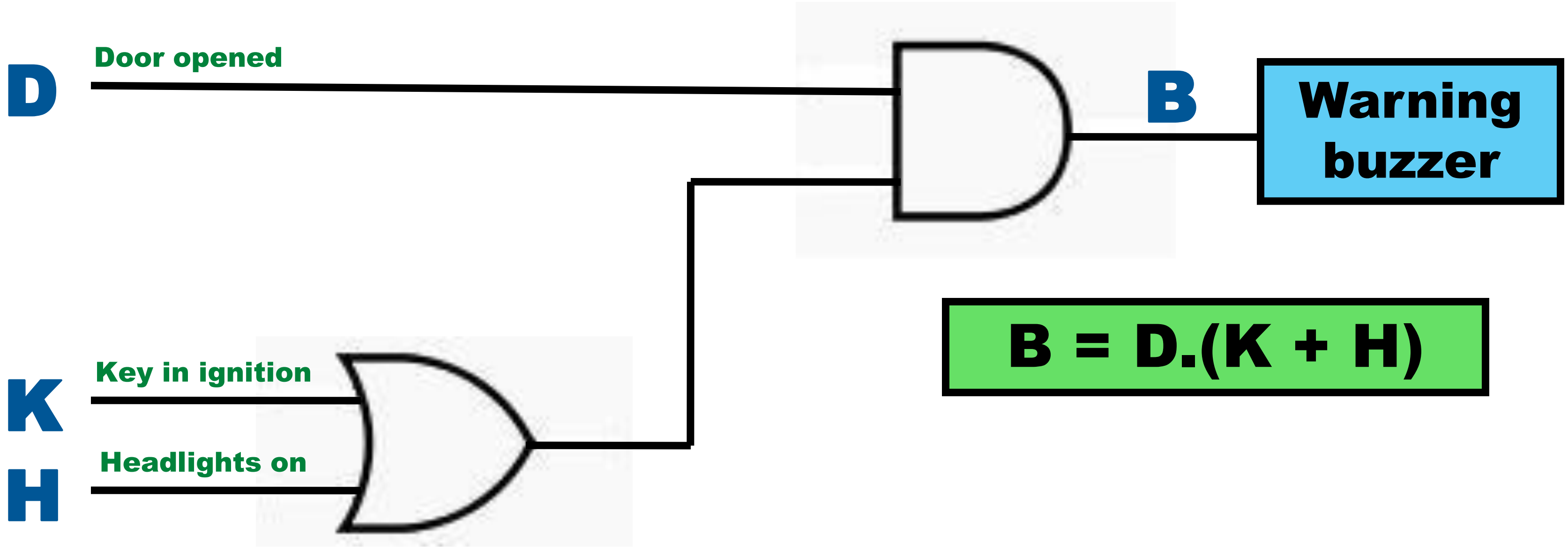
- Example: Automobile Warning
  - Another (alternative) way of stating the criteria

Buzzer is activated whenever door is opened AND either key is in ignition OR headlights are on.

$B = K.D + H.D$ → $B = D.(K + H)$

Combinational Logic

- Example: Automobile Warning
  - Logic Circuit



B = D.(K + H)

Combinational Logic

- Another example: Burglar Alarm
  - Write out the Boolean equation and draw out the reduced logic circuit for this scenario:

Burglar alarm (A) is activated if it is switched on (S) AND front door is open (F), OR if it is switched on (S) AND back door is open (B), OR if it is switched on (S) AND a window is open (W).

Combinational Logic

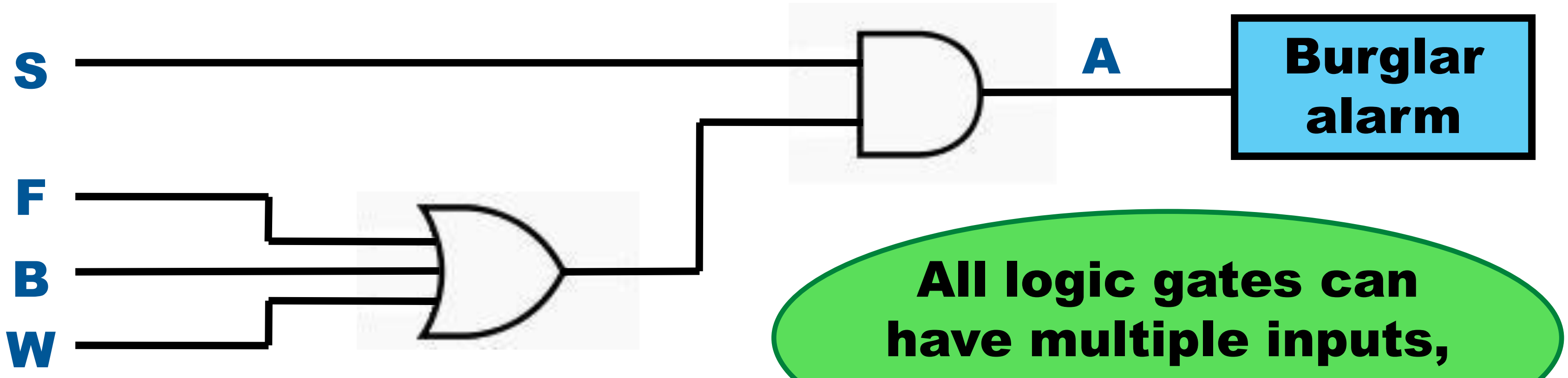
- Example: Burglar Alarm
  - Write out the Boolean equation and simplify

$$A = S.F + S.B + S.W$$

$$A = S.(F + B + W)$$

Combinational Logic

- Example: Burglar Alarm
  - Reduced logic circuit



All logic gates can have multiple inputs, but only 1 output