Worksheet: Logic Gates and Truth Tables

Logic Gate Truth Tables

Use the online interactive logic simulation to fill out the values of the truth tables. Draw the gate in the box at the top of each table. For all tables, use the same order of values for `IN1` and `IN2` as used in the `AND` gate truth table.

AND		
IN1	IN2	OUT
0	0	
0	1	
1	0	
1	1	

OR		
IN1	IN2	OUT

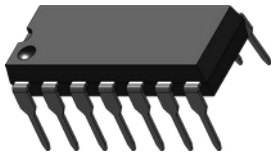
XOR		
IN1	IN2	OUT

NAND		
IN1	IN2	OUT

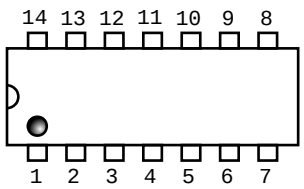
NOR		
IN1	IN2	OUT

XNOR		
IN1	IN2	OUT

Dual Inline Package (DIP)



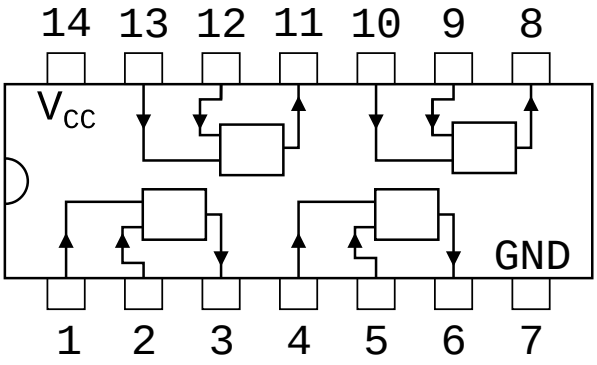
Logic gates are available as integrated circuits in various types of packages. We will look a number of these gates, each in a through-hole dual inline package (DIP). An image of this package type is shown to the left.



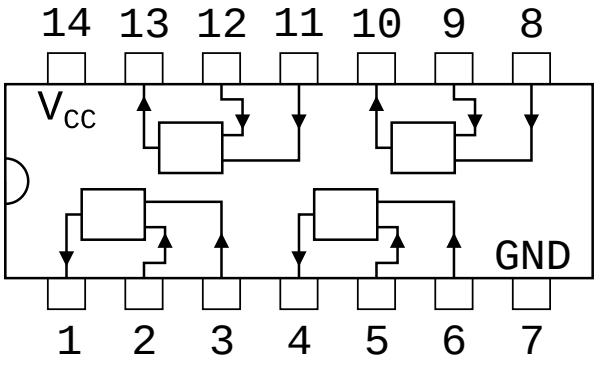
For DIP packages, the pin numbering and orientation is typically indicated in two ways. There is often a small dot near pin 1 of the package. Alternatively (or additionally), there is a semicircle cut out of one end of the package. When oriented with the semicircle on the left, pin 1 is the leftmost pin on the lower side. Pin numbering proceeds in a counterclockwise direction around the chip.

Worksheet: Logic Gates and Truth Tables

Package Pinout



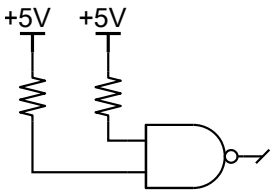
Circuits with a **red** DIP switch package
(74HC00, 74HC08, 74HC32, 74HC86)



Circuit with a **blue** DIP switch package
(74HC02)

The above diagrams show the pinouts of the integrated circuits we will investigate in this activity. For most of the chips, the pinout follows the diagram on the lefthand side. For these gates, I have placed a red DIP switch package on the breadboard. For the one chip that has a different pinout, I have placed a blue DIP switch package on the breadboard.

Pull-up or Pull-down Resistors



For many integrated circuits, including the logic gates we will be working with, when an input is left floating, the level can float between the on and off levels. Transistors within the chip may then try to simultaneously turn the output both on and off at the same time, which creates a direct pathway between power and ground, resulting in a high power consumption and unpredictable behavior of the chip.

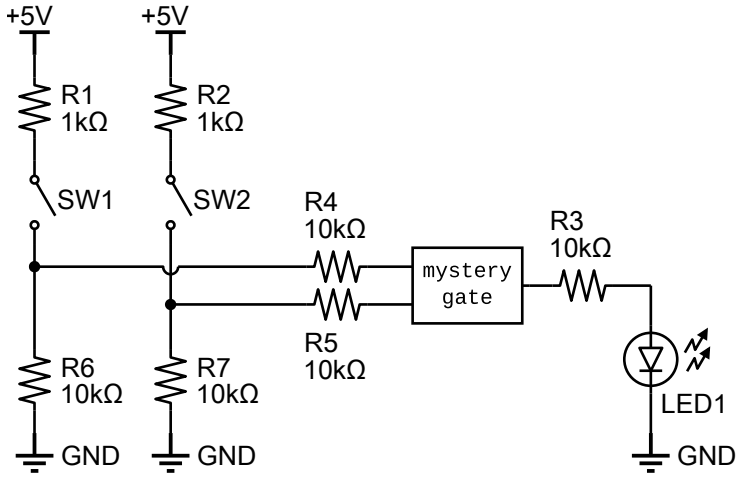
It is fine to tie the inputs directly to either power or to ground. However, for our activity, we will tie all unused inputs to either the positive supply voltage or ground through a 10kΩ resistor. We use a resistor just in case of error – if we accidentally tie an output directly to power or ground, it again can damage the chip. The resistor would limit the current flow and protect the chip.

Gate Input Circuit

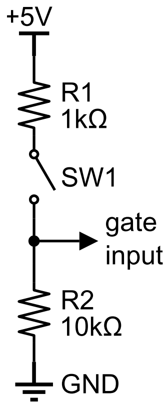
The figure to the right shows the circuit you are to build to determine which type of gate you have been given.

Just as with the online logic simulation, you will toggle the switches and observe the output.

Fill in the truth table on the next page and compare these results to the tables in the section ***Logic Gate Truth Tables*** to determine the gate.



Worksheet: Logic Gates and Truth Tables



The figure to the left shows the circuit that provides the signal to each gate input. When the switch is open, the gate input is tied to ground. When the switch is closed, the circuit is a voltage divider.

What is the voltage level of the gate input when the switch is open? Write this in the box to the right.

What is the voltage level of the gate input when the switch is closed? Calculate this in the box below. Show the steps in the calculation.

Fill in the truth table to the right according to the output you observe in your circuit. Use the truth tables in section ***Logic Gate Truth Tables*** to determine the gate used in your circuit. In the appropriate box below, write the part number (`74HCxx`) and the name of the gate, and draw the gate.

Part Number:

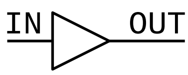
Gate Name:

Gate diagram:

IN1	IN2	OUT
0	0	
0	1	
1	0	
1	1	

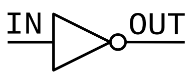
Buffers and Inverters

The output of a buffer is a copy of the input. It is used to strengthen a signal.



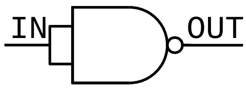
IN	OUT
0	**0**
1	**1**

The output of an inverting buffer is the opposite of the input signal.



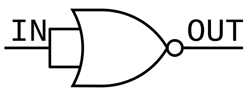
IN	OUT
0	**1**
1	**0**

Use the logic tables from section ***Logic Gate Truth Tables*** to determine the values for the truth table for the circuit to the right.



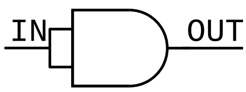
0	
1	

Use the logic tables from section ***Logic Gate Truth Tables*** to determine the values for the truth table for the circuit to the right.



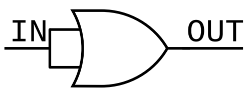
0	
1	

Use the logic tables from section ***Logic Gate Truth Tables*** to determine the values for the truth table for the circuit to the right.



0	
1	

Use the logic tables from section ***Logic Gate Truth Tables*** to determine the values for the truth table for the circuit to the right.



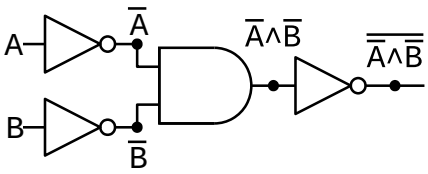
0	
1	

Unused NAND and NOR gates in a circuit can be used as inverters.

Worksheet: Logic Gates and Truth Tables

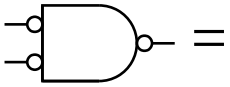
De Morgan's Law

Complete the truth table for the circuit below.

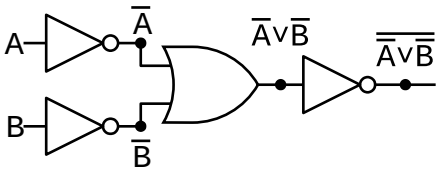


A	B	$\overline{A}$	$\overline{B}$	$\overline{A}\wedge\overline{B}$	$\overline{\overline{A}\wedge\overline{B}}$

The gate to the right is the equivalent to the circuit above. Examine the output and determine which gate from section ***Logic Gate Truth Tables*** this logic gate is equivalent to. Draw the gate in the box.

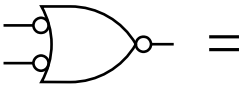


Complete the truth table for the circuit below.



A	B	$\overline{A}$	$\overline{B}$	$\overline{A}\vee\overline{B}$	$\overline{\overline{A}\vee\overline{B}}$

The gate to the right is the equivalent to the circuit above. Examine the output and determine which gate from section ***Logic Gate Truth Tables*** this logic gate is equivalent to. Draw the gate in the box.



The above exercise is a demonstration of De Morgan's Law. This law has two forms:

$$\overline{\overline{A}\wedge\overline{B}} = A\vee B \qquad\qquad \overline{\overline{A}\vee\overline{B}} = A\wedge B$$

Or, if you find the multiple overline difficult to remember, you can remember these two forms:

$$\overline{A}\wedge\overline{B} = \overline{A\vee B} \qquad\qquad \overline{A}\vee\overline{B} = \overline{A\wedge B}$$

The following is a programming example where De Morgan's law can be used to show these two Java methods return the same value for all inputs.

```
public boolean isTeenager(int age) {
   return (age>=13) && (age<=19);
}
```

```
public boolean isTeenager(int age) {
   return !( (age<13) || (age>19) );
}
```

In this code, if $\overline{A}$ is `(age>=13)` then A will can be expressed as `(age<13)`. Similarly, if $\overline{B}$ is `(age<=19)`, then B can be expressed as `(age>19)`.

We can then see that the left method returns $\overline{A}$ && $\overline{B}$, which is the same as $\overline{A}\wedge\overline{B}$. The second method returns `!(A || B)`, which is the same as $\overline{A\vee B}$.

Since $\overline{A}\wedge\overline{B} = \overline{A\vee B}$, these methods will always return the same value.