IDC102:Hands-on Electronics
Lecture – 9

Introduction to Boolean Algebra

IISER

Satyajit Jena, 18/07/2022

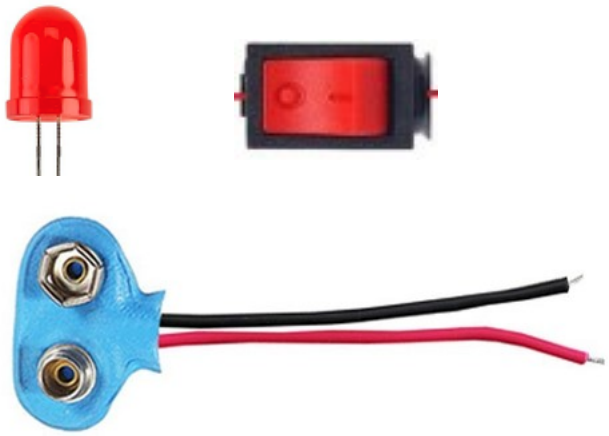
Introduction: Devices/Computers

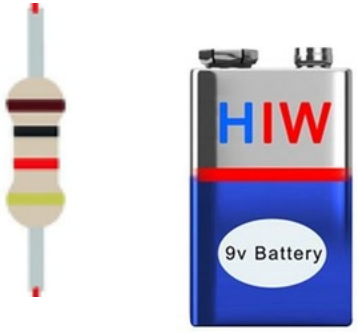
A computer is a machine that can be instructed to carry out sequences of arithmetic or logical operations automatically via computer programming. (Wikipedia)

Com·put·er /kəm'pjutəɹ/

An electronic device designed to accept data, perform prescribed mathematical and logical operations at high speed, and display the results of these operations. (Dictionary.com)

"The idea behind digital computers may be explained by saying that these machines are intended to carry out any operations which could be done by a human computer. The human computer is supposed to be following fixed rules; he has no authority to deviate from them in any detail. We may suppose that these rules are supplied in a book, which is altered whenever he is put on to a new job. He has also an unlimited supply of paper on which he does his calculations." – Alan Turing, 1950



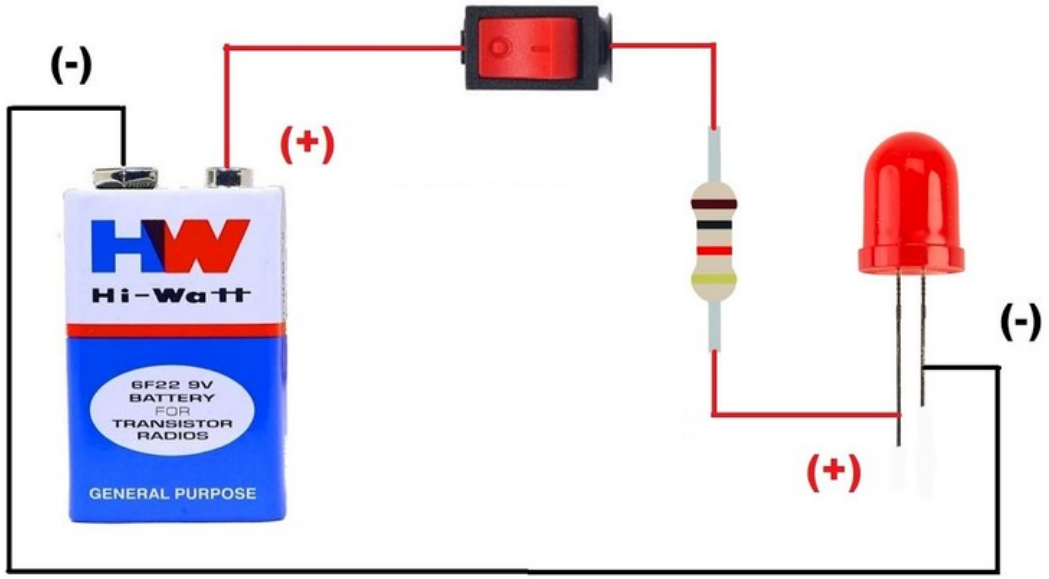


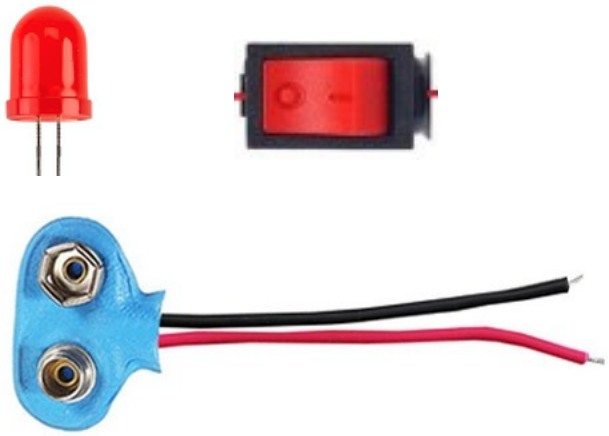
A LED,
A Switch
A resister
A connector and a battery
Some wires

Connect all components as shown in this a diagram.

If circuit is correct, you can light up the LED at your wish.

You need to switch on to glow the light and switch off to shut it off.



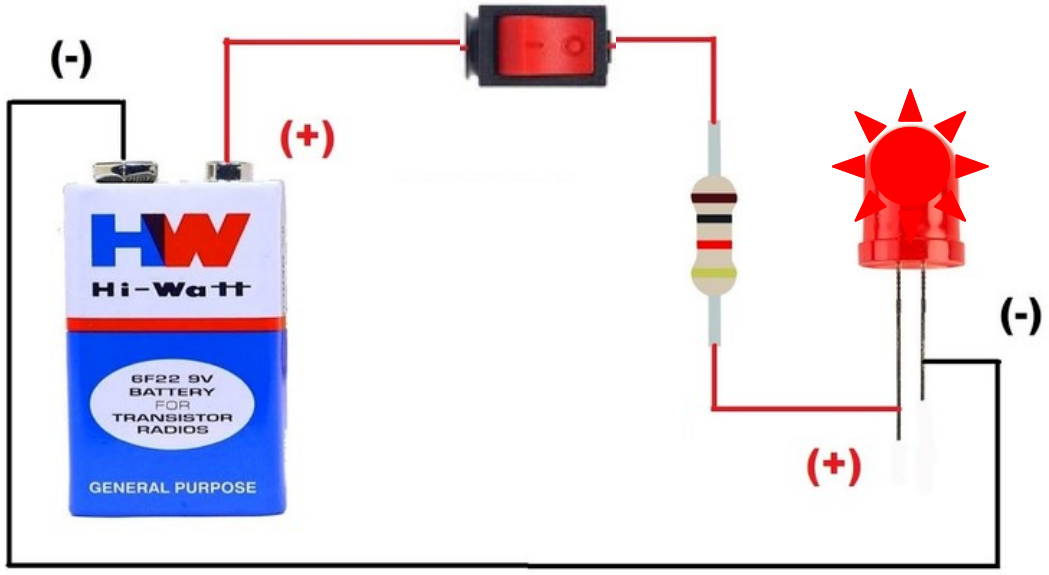


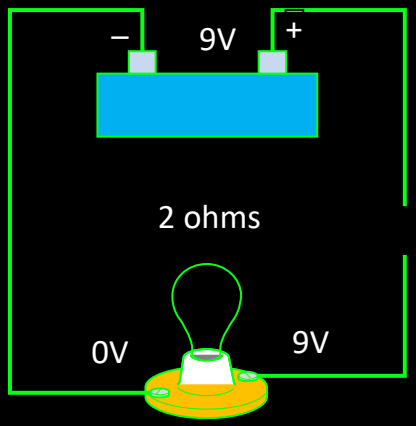
A LED,
A Switch
A resister
A connector and a battery
Some wires

Switch on: LED is ON
Switch off: LED is off

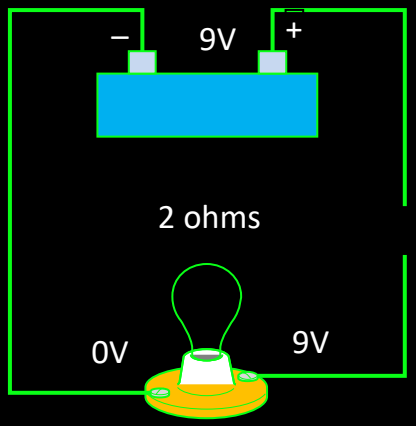
To Light up the LED: ON
To Light off the LED: OFF

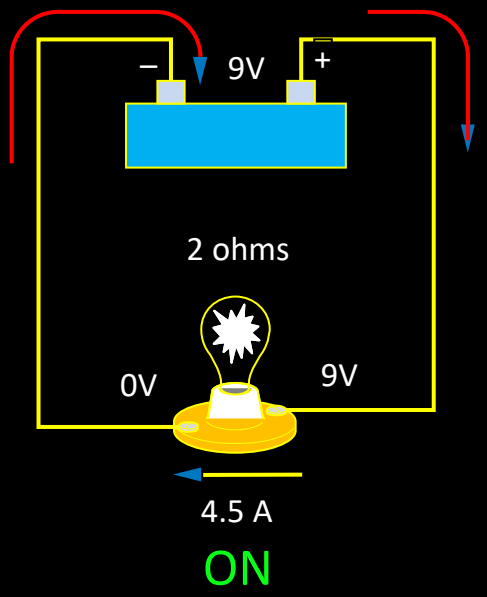
This is very simple circuit, can we do something by using this?





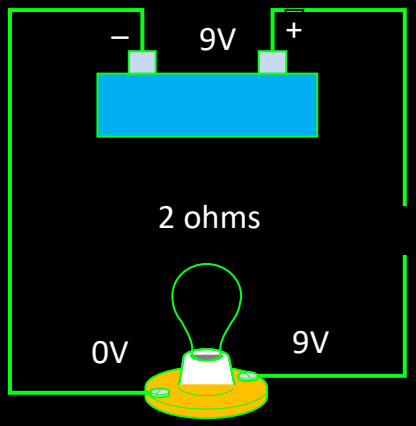
Lets check on what interesting thing can be done using this simple circuit?

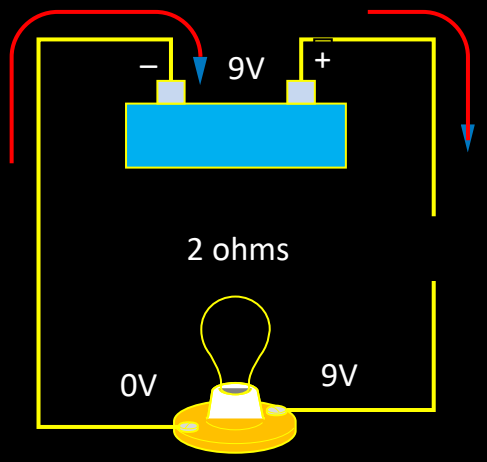




Switch on: LED is ON
Switch off: LED is off

To Light up the LED: ON
To Light off the LED: OFF

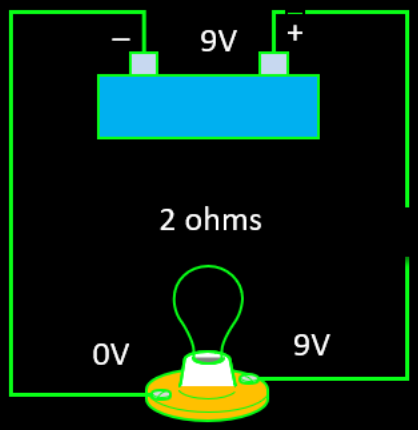


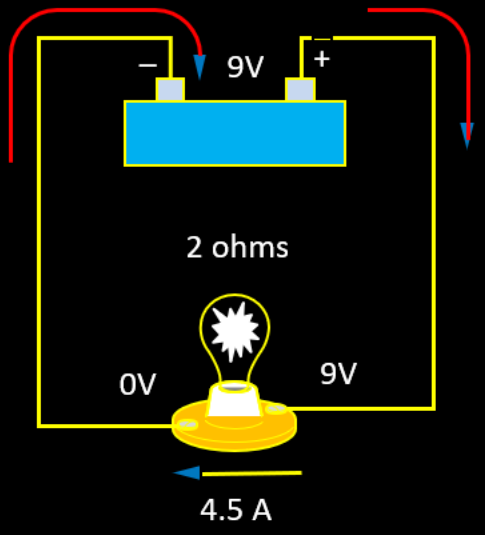


OFF

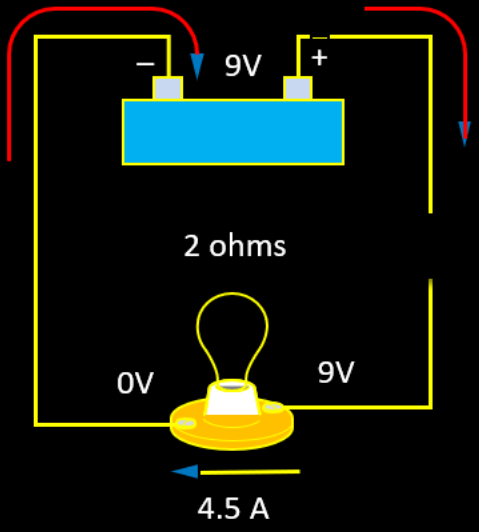
Switch on: LED is ON
Switch off: LED is off

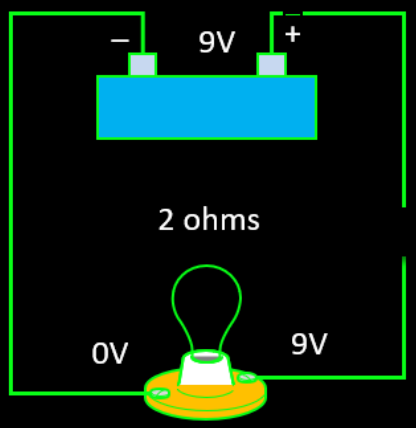
To Light up the LED: ON
To Light off the LED: OFF

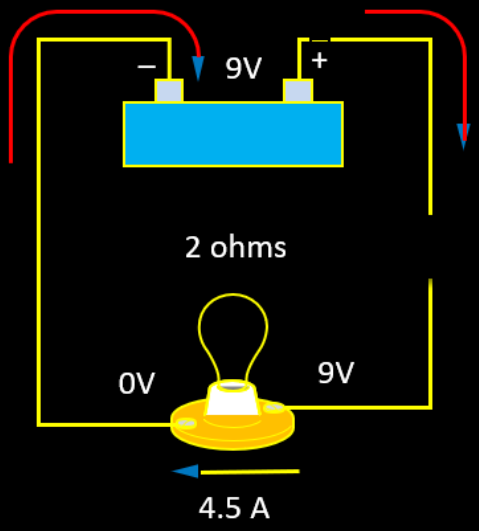


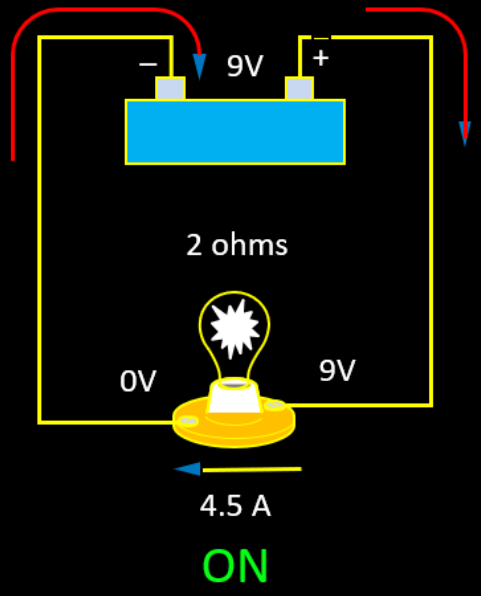


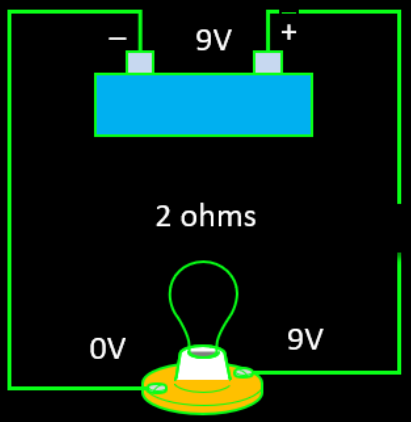
ON

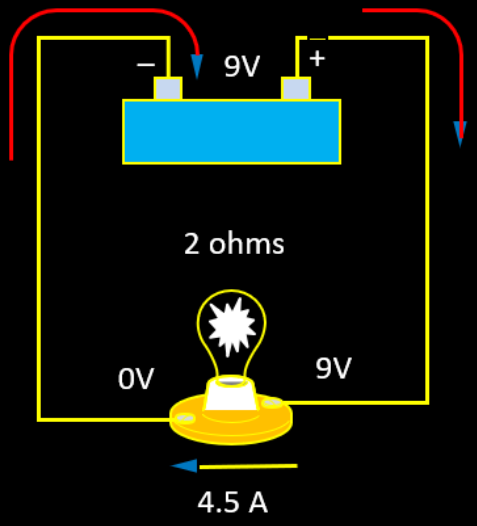



–
9V
+
2 ohms
0V
9V

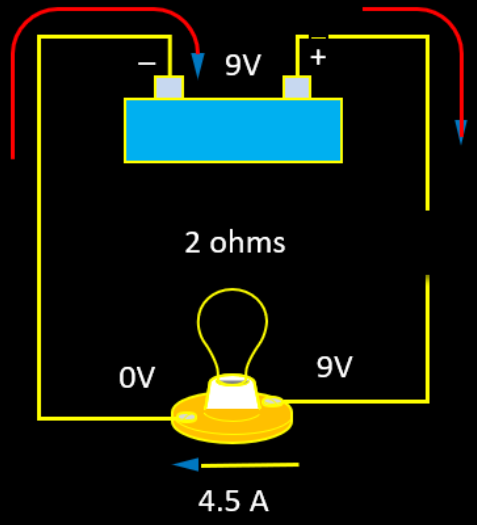

–
9V
+
2 ohms
0V
9V
4.5 A


–
9V
+
2 ohms
0V
9V
4.5 A
ON

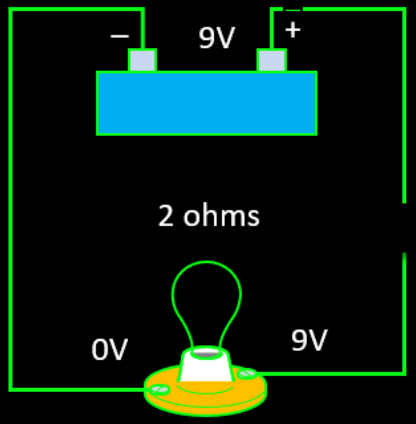


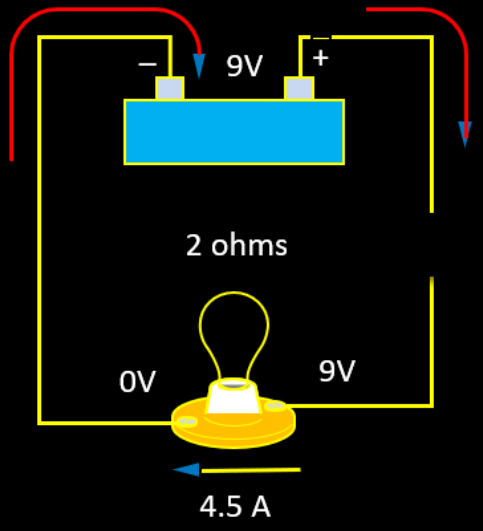


1

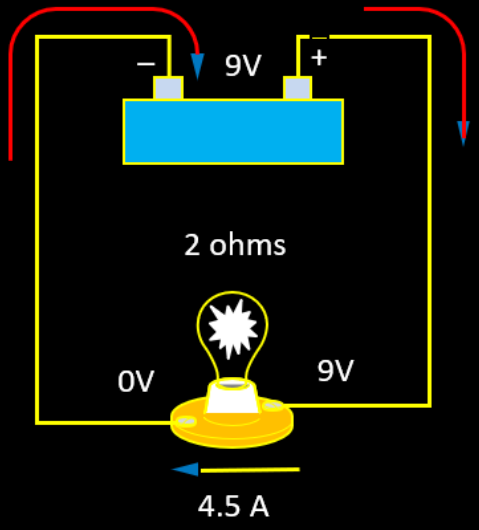


0

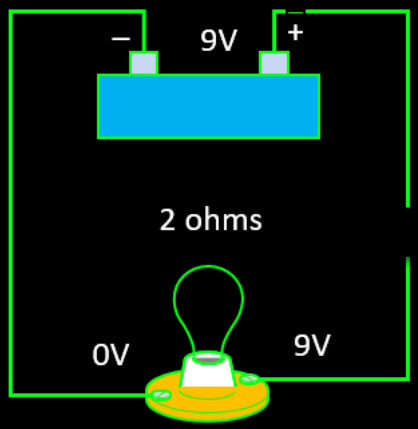


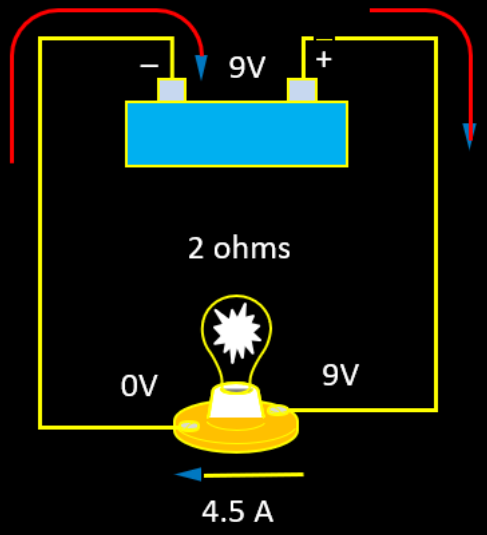


0

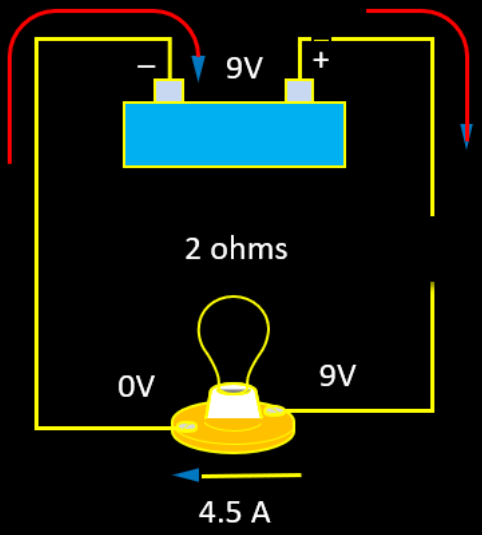


1





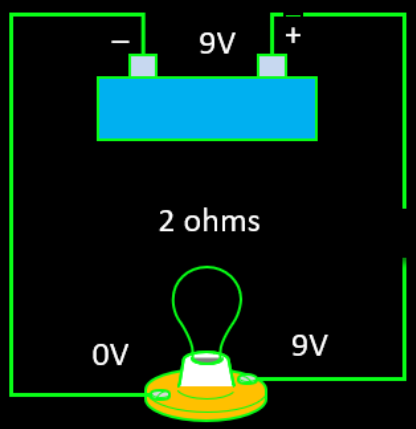
1

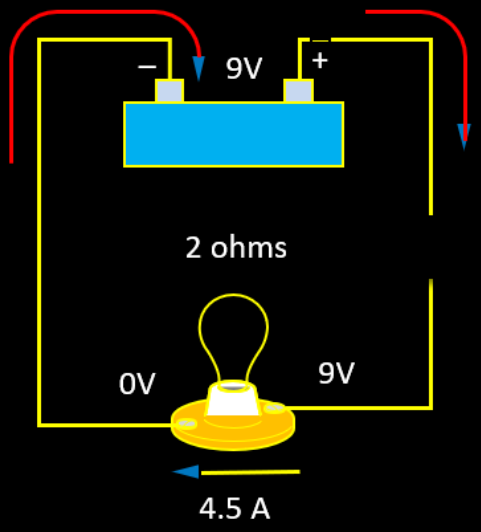


0

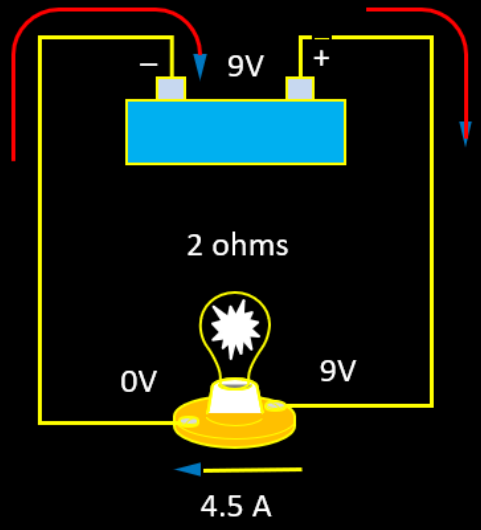
1

0





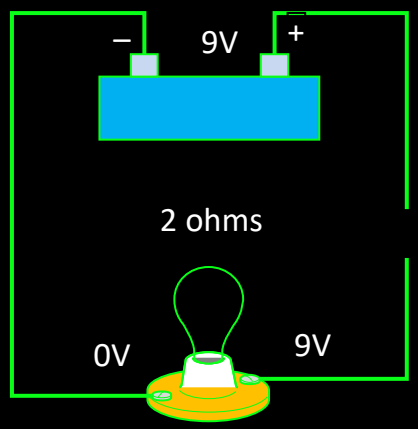
0

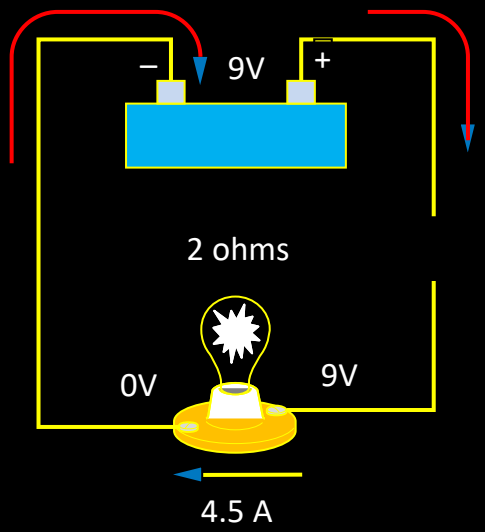


1

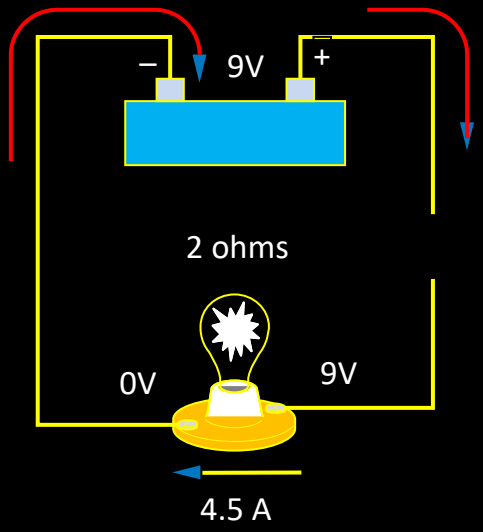
1 0

0 1





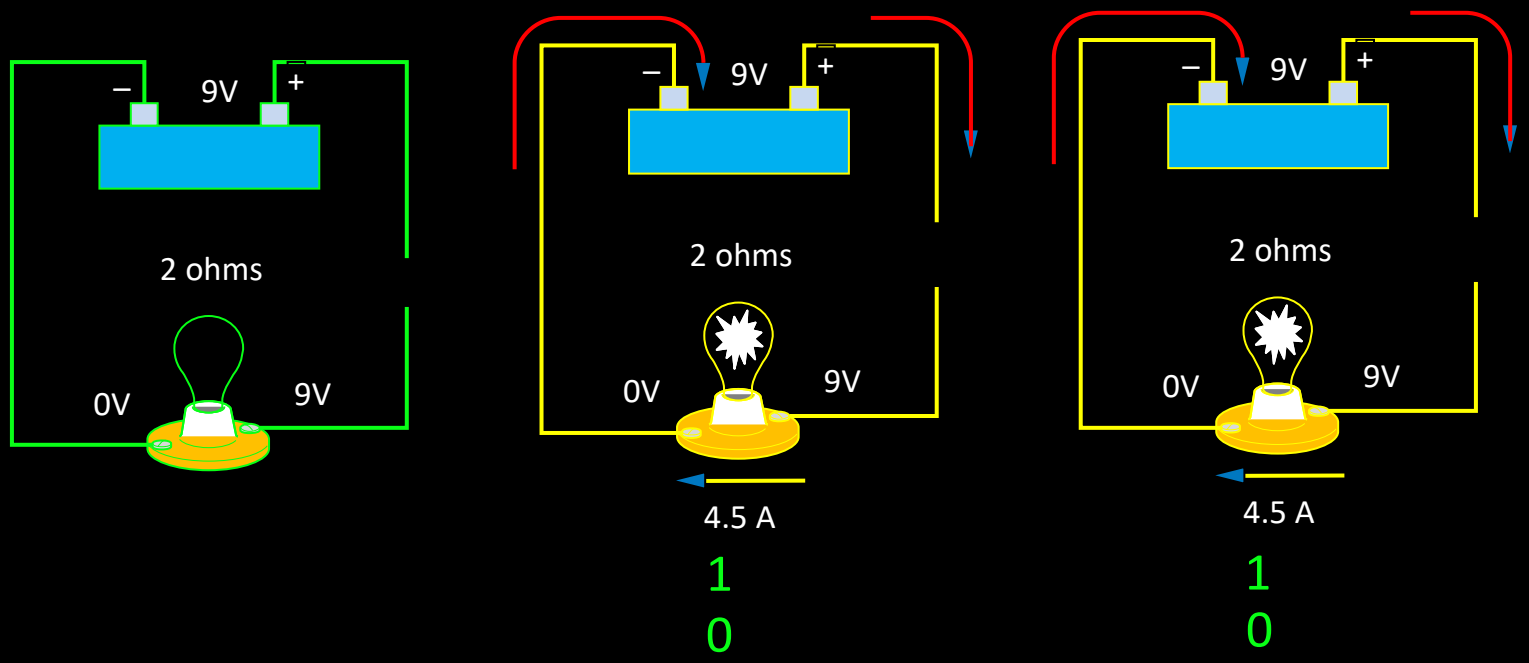
1
0



1
0

1 0 1 0 1 0 1 0 1 0 1 0 1

0 1 0 1 0 1 0 1 0 1 0 1 0



1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0

0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1

These are the states of currents(or LED) for two wires, put more wires, you can have combination of more states.

=> Simply using the on and off state we can connect it to a small system of world of binary (0,1)

What we know in Decimal System

We all know about decimal system 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

10 digits => how many numbers can you form?

Values of each digits changes depending on their positions

What we know in Decimal System

We all know about decimal system 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

10 digits => how many numbers can you form?

6	9	8	2	3

Values of each digits changes depending on their positions

What we know in Decimal System

We all know about decimal system 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

10 digits => how many numbers can you form?

$x\ 10^4$	$x\ 10^3$	$x\ 10^2$	$x\ 10^1$	$x\ 10^0$
6	9	8	2	3

Values of each digits changes depending on their positions

We all know about decimal system 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

10 digits => how many numbers can you form?

$x\ 10^4$	$x\ 10^3$	$x\ 10^2$	$x\ 10^1$	$x\ 10^0$
6	9	8	2	3
60000	+9000	+800	+20	+3

= 69823

We all know about decimal system 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

10 digits => how many numbers can you form?

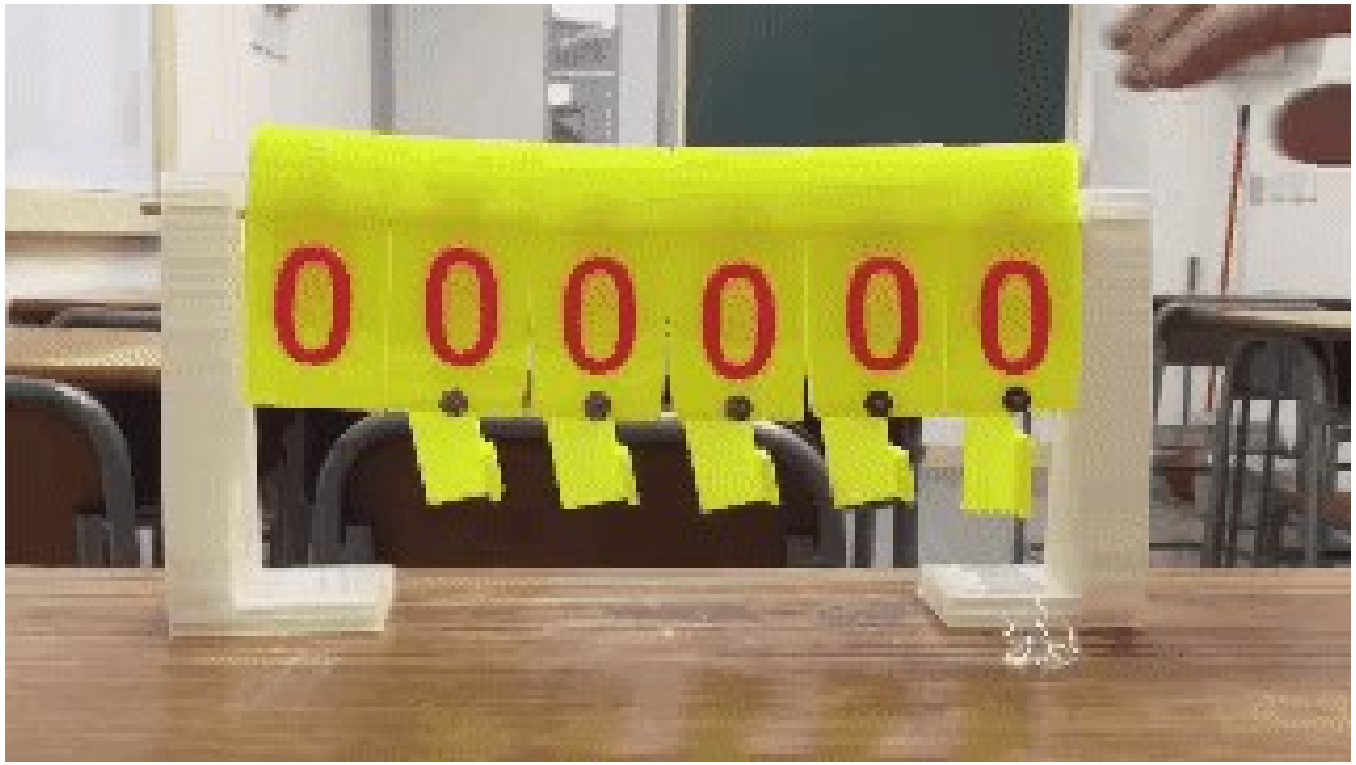
$x\ 10^4$	$x\ 10^3$	$x\ 10^2$	$x\ 10^1$	$x\ 10^0$
6	9	8	2	3
60000	+9000	+800	+20	+3

= 69823

What if choice is the binary? i.e. 0 and 1?

Binary System

0 and 1



The position is a multiple of 2

Image source: google

Binary System

2^4	2^3	2^2	2^1	2^0
16	8	4	2	1
1	0	1	1	1

Binary System

128 64 32 16 8 4 2 1

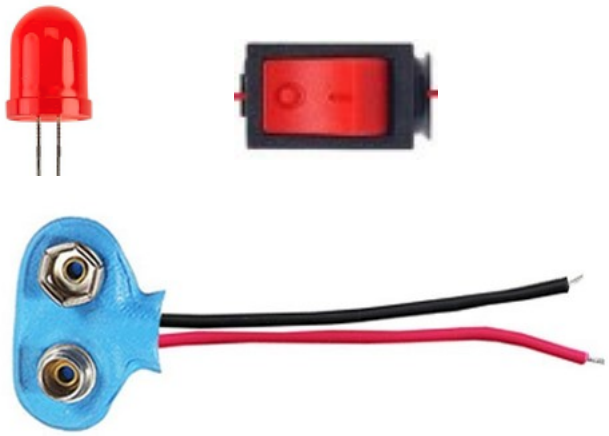
Binary System

2^4	2^3	2^2	2^1	2^0	
16	8	4	2	1	
0	0	0	0	0	0 0

So apparently you can generate almost all decimal numbers.
(we will have more discussion later on this).

Well, we know all these, how this is connected to computer anyway?

We started here!

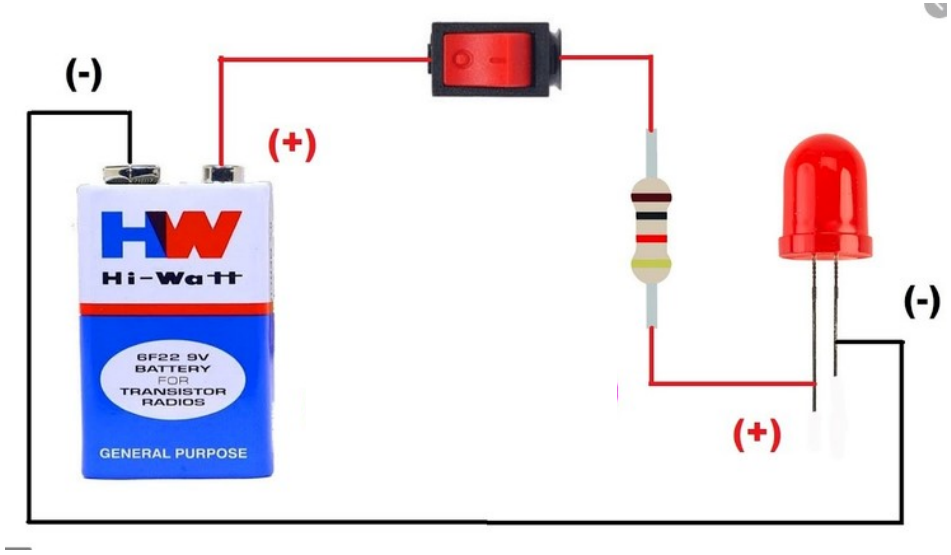


A LED,
A Switch
A resister
A connector and a battery
Some wires

Connect all components as shown in this a diagram.

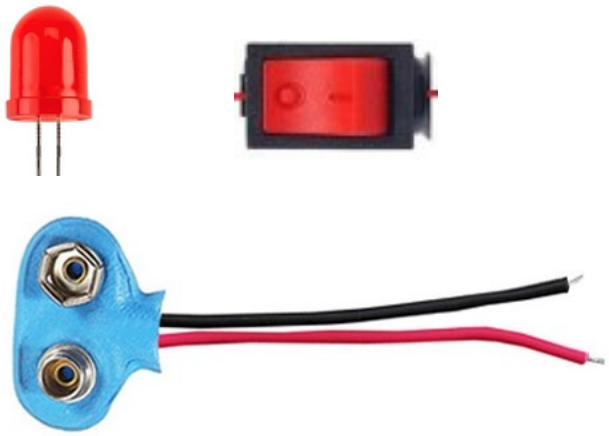
If circuit is correct, you can light up the LED at your wish.

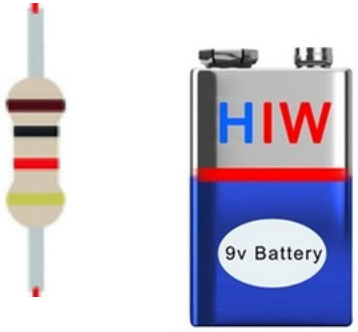
You need to switch on to glow the light and switch off to shut it off.



How do I use this idea any way?

We started here!



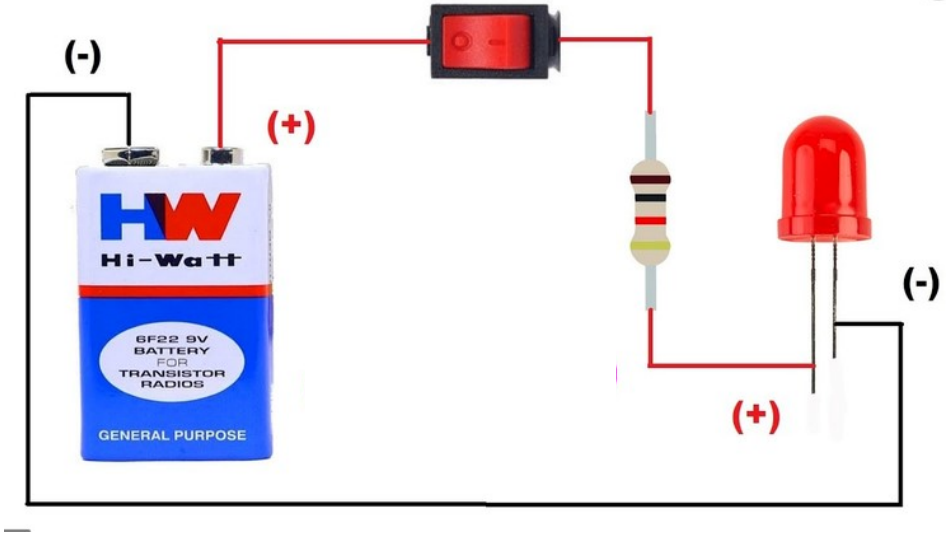


A LED,
A Switch
A resister
A connector and a battery
Some wires

Connect all components as shown in this a diagram.

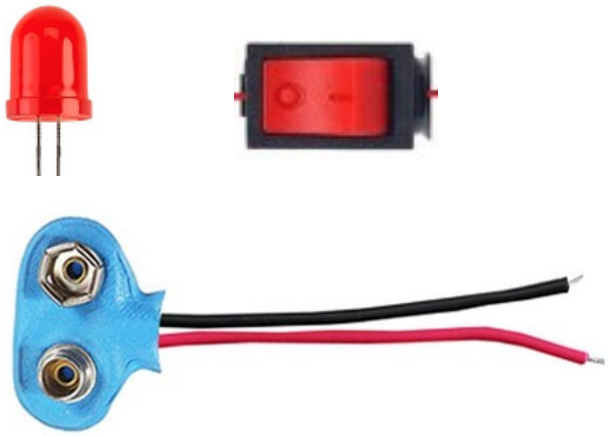
If circuit is correct, you can light up the LED at your wish.

You need to switch on to glow the light and switch off to shut it off.



How do I use this idea any way? => This is indeed a building block of computer?

We started here!



A LED,
A Switch
A resister
A connector and a battery
Some wires

To process it we need an instruction:

ON or OFF

=> INPUT

To know what happens: we need a LED:

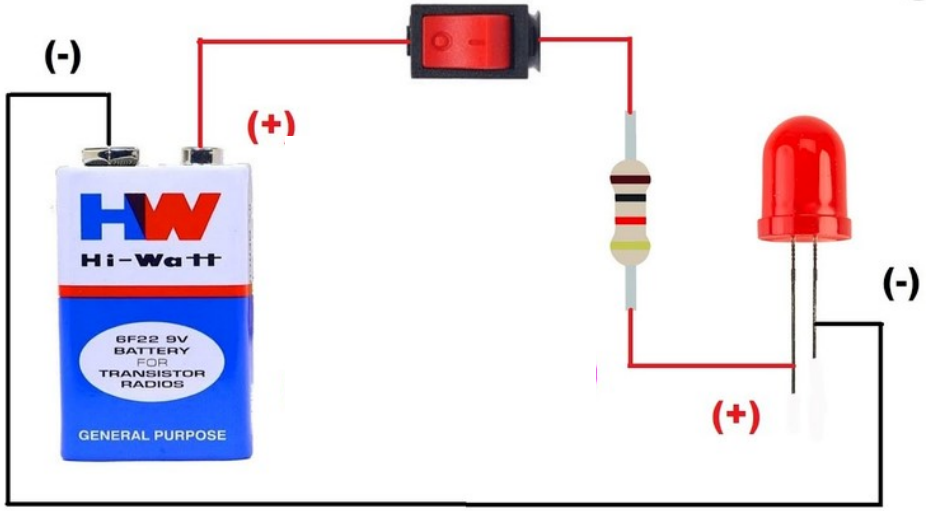
light up or light down

=> OUTPUT

The things happen in the wire:

Current flows

=> Processing

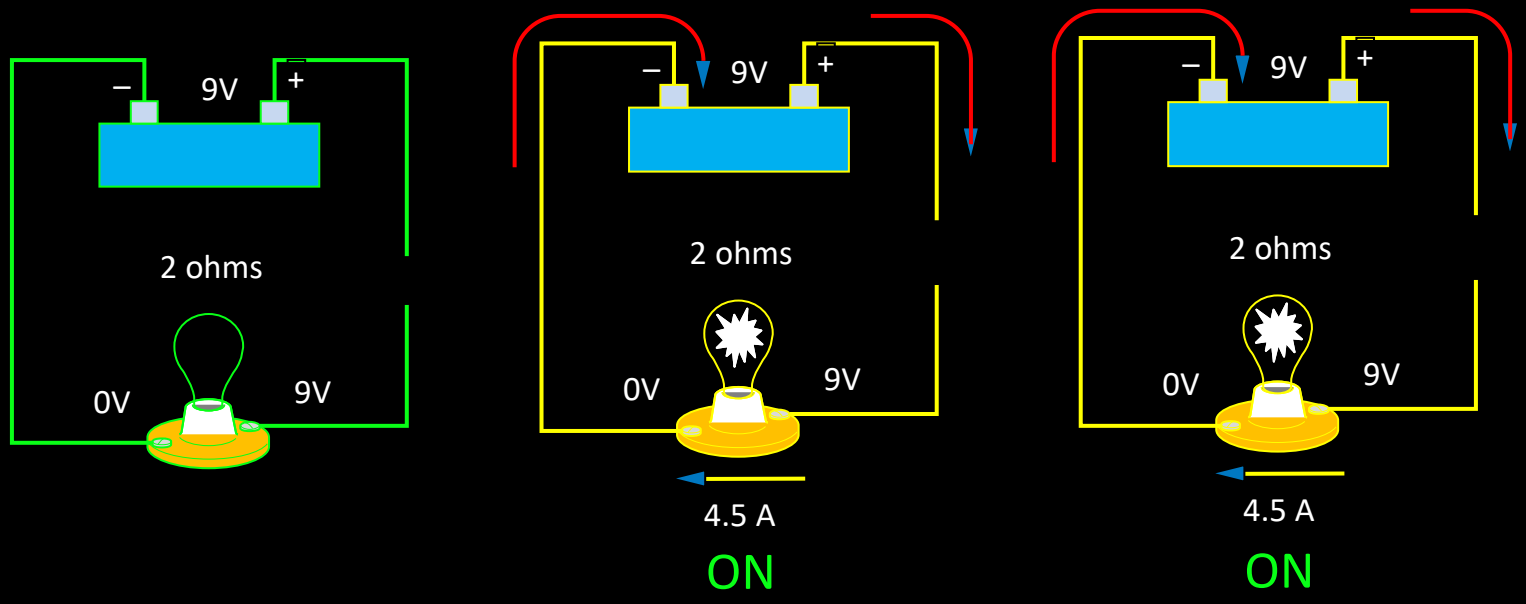


More such blocks you add, more complex system you can build.

How to do it?

Indeed, the successful combinations of such block to take inputs, process the information store the information and show the results is what makes computer a computer

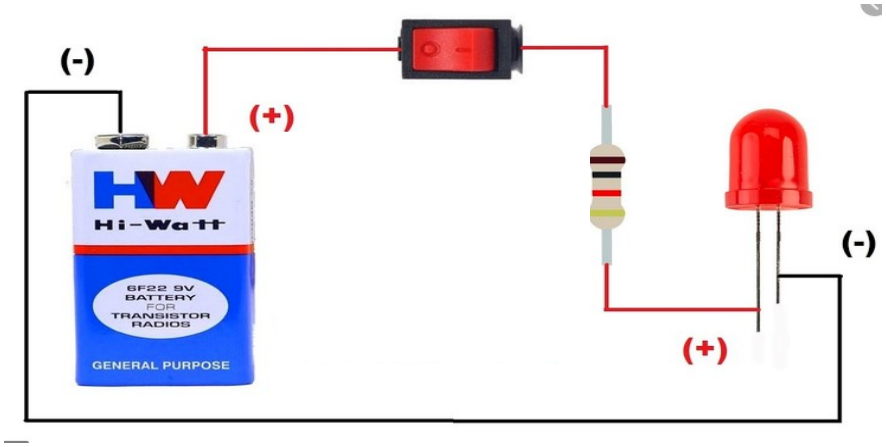
How to use a circuit in computation



Consider switching a circuit: input (0, 1)

Let's form a logic out of it

How to use a circuit in computation



Physics tells us that: Current follows minimum Path

- Yes
- ON
- TRUE
- HI
- mark

- No
- OFF
- FALSE
- LOW
- space

Representations of Digital Design: Switches

A switch connects two points under control signal.

Normally Open

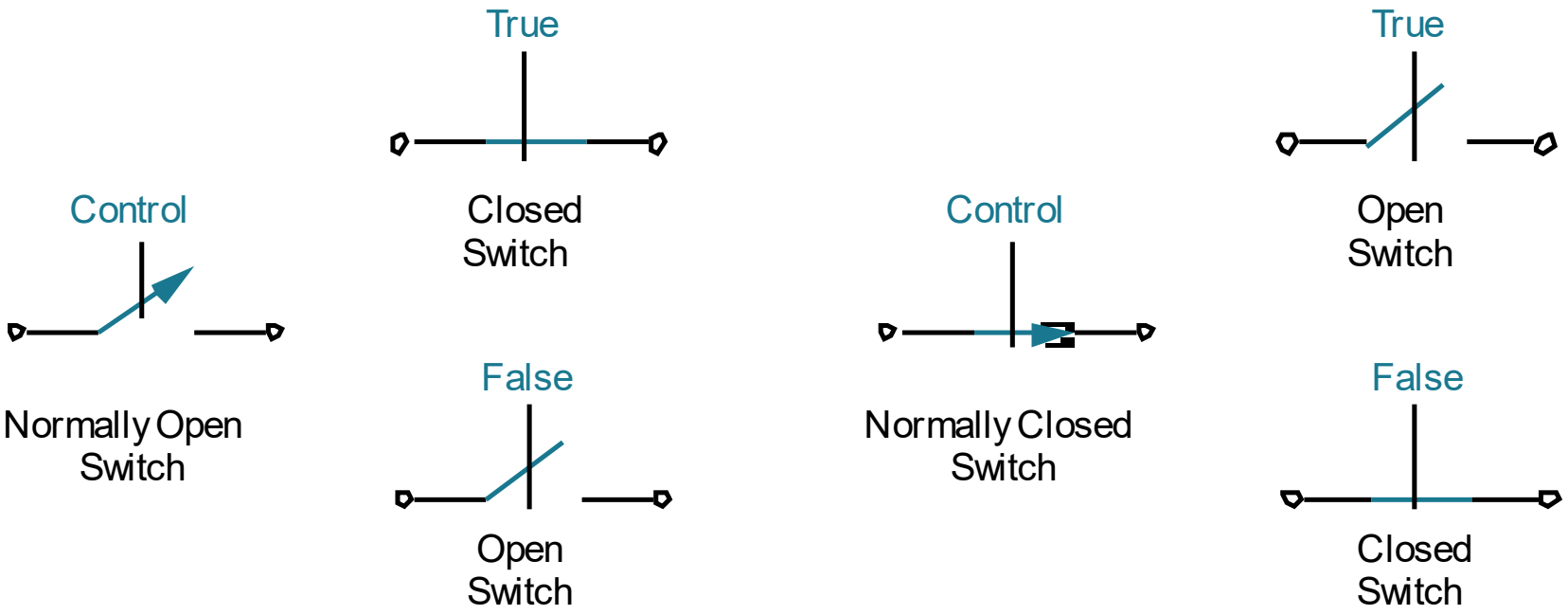
when the control signal is 0 (false), the switch is open

when it is 1 (true), the switch is closed

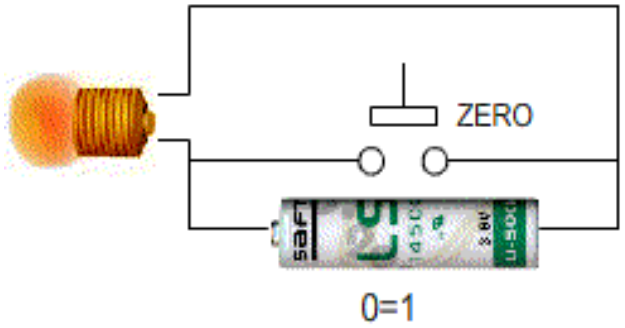
Normally Closed

when control is 1 (true), switch is open

when control is 0 (false), switch is closed



Not Gate



input	output
1	0
0	1

Anything you give it will return opposite of it

Boolean Algebra and Logical Operators

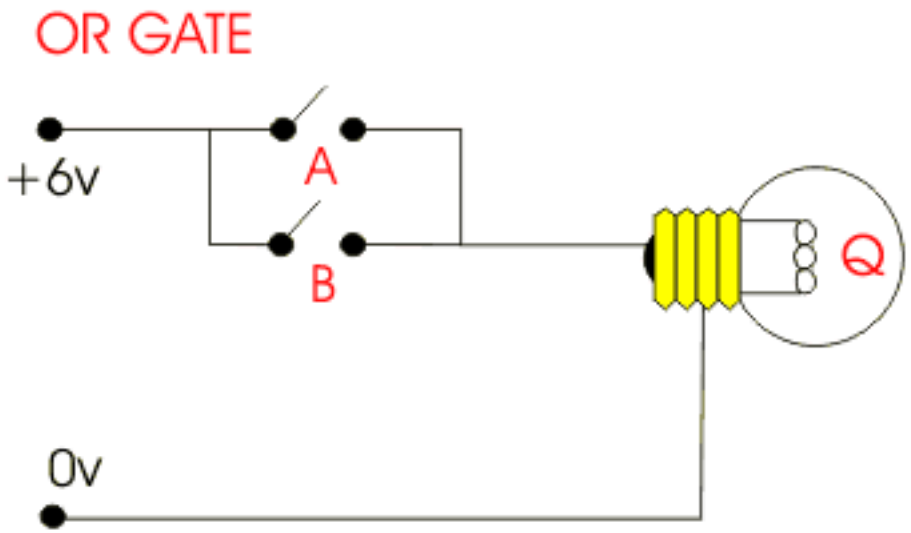
The truth table for the Boolean NOT operator is shown at the right.

The NOT operation is most often designated by an overbar. It is sometimes indicated by a prime mark (‘) or an “elbow” (¬).

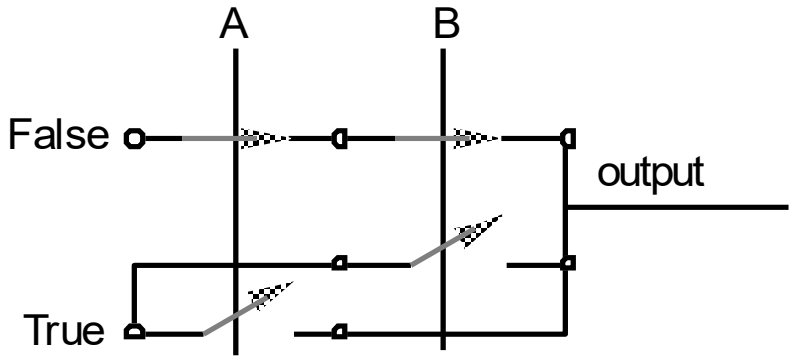
NOT x

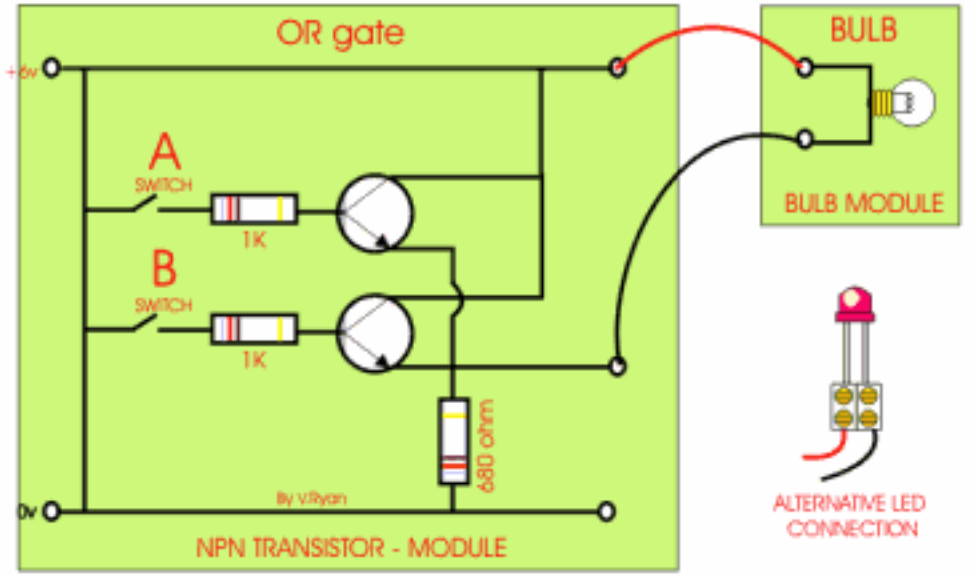
x	$\overline{x}$
0	1
1	0

Or Gate

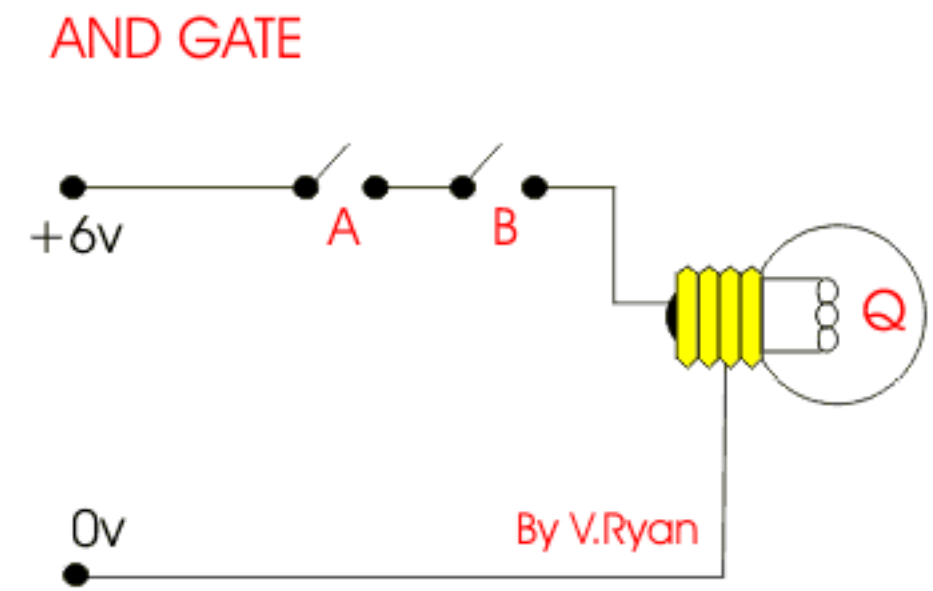


A	B	F=A+B
0	0	0
0	1	1
1	0	1
1	1	1

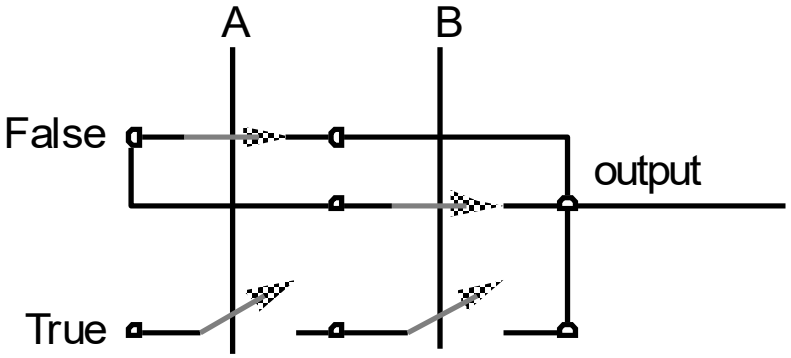


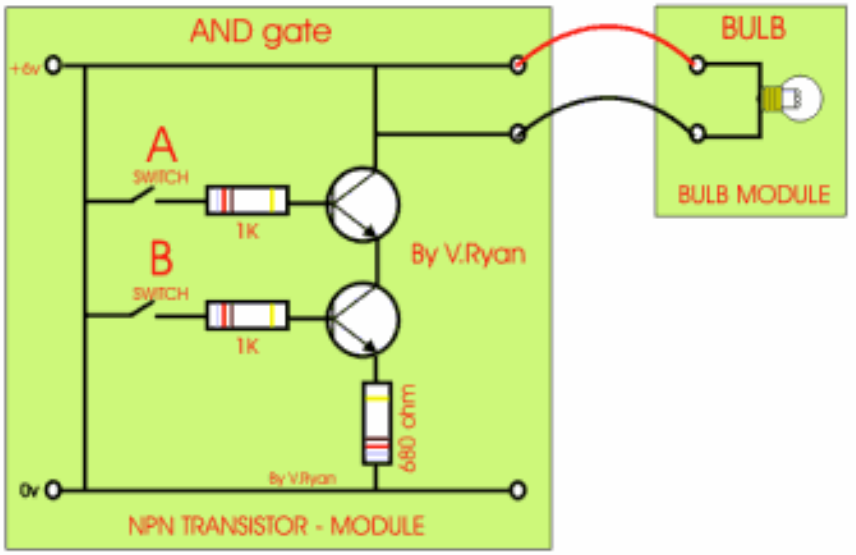


And Gate



A	B	F=A·B
0	0	0
0	1	0
1	0	0
1	1	1



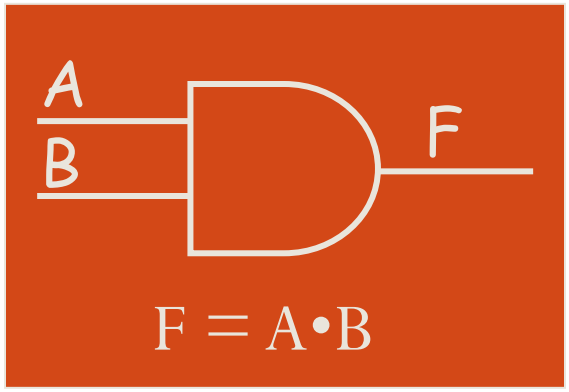


Representation

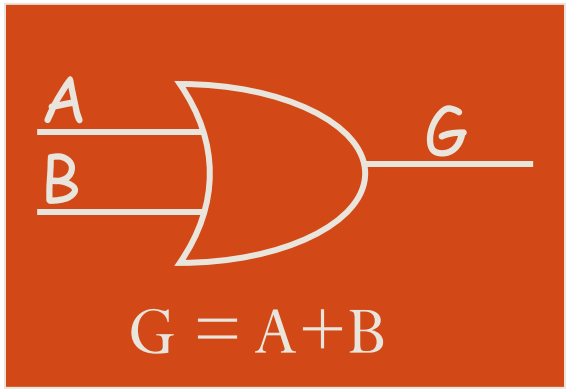
- Logic gates are abstractions of electronic circuit components that operate on one or more input signals to produce an output signal.

Symbols are

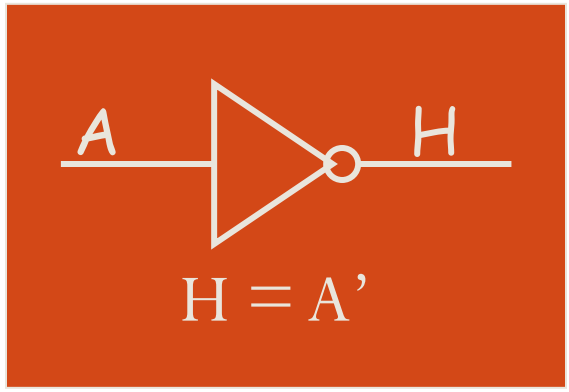
2-Input AND



2-Input OR



NOT (Inverter)



The AND operator is also known as a Boolean product. The OR operator is the Boolean sum.

Boolean Algebra

- A Boolean function has:
  - At least one Boolean variable,
  - At least one Boolean operator, and
  - At least one input from the set {0,1}.
- It produces an output that is also a member of the set {0,1}.

Now you know why the binary numbering system is so handy in digital systems.

X AND Y

X	Y	XY
0	0	0
0	1	0
1	0	0
1	1	1

X OR Y

X	Y	X+Y
0	0	0
0	1	1
1	0	1
1	1	1

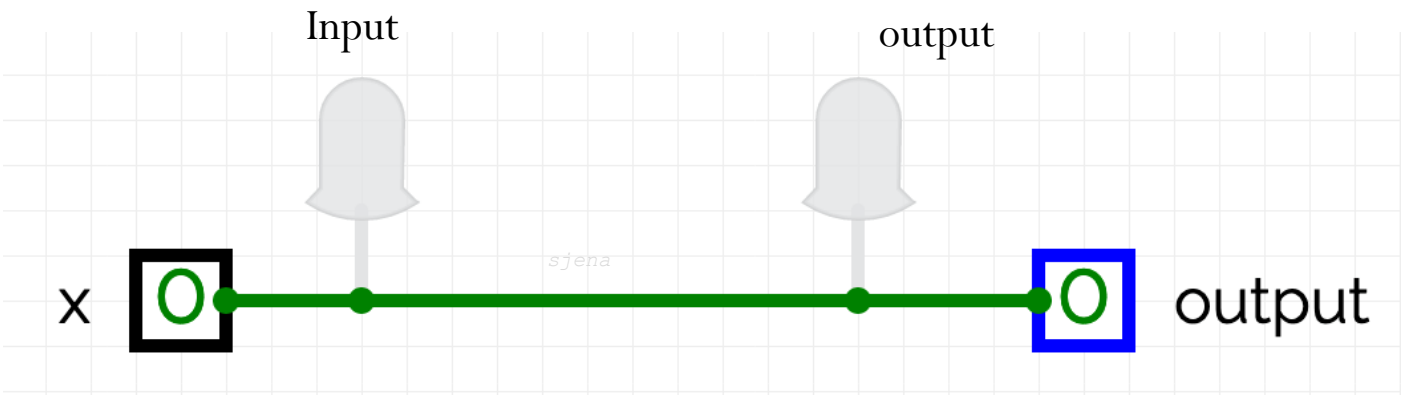
NOT X

X	$\overline{X}$
0	1
1	0

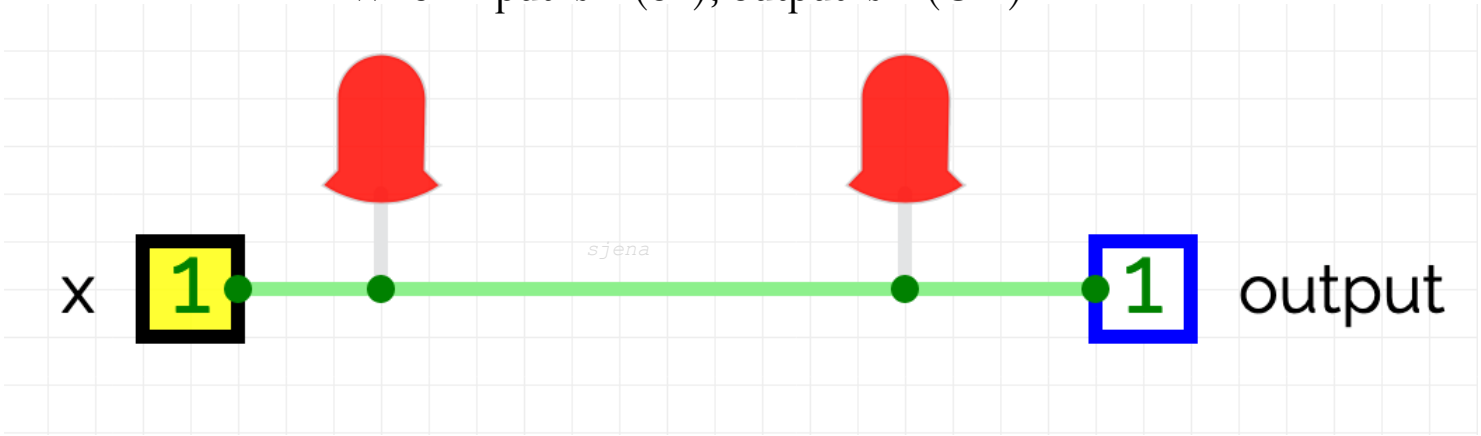
Let's wire them in circuit using symbal

Circuit

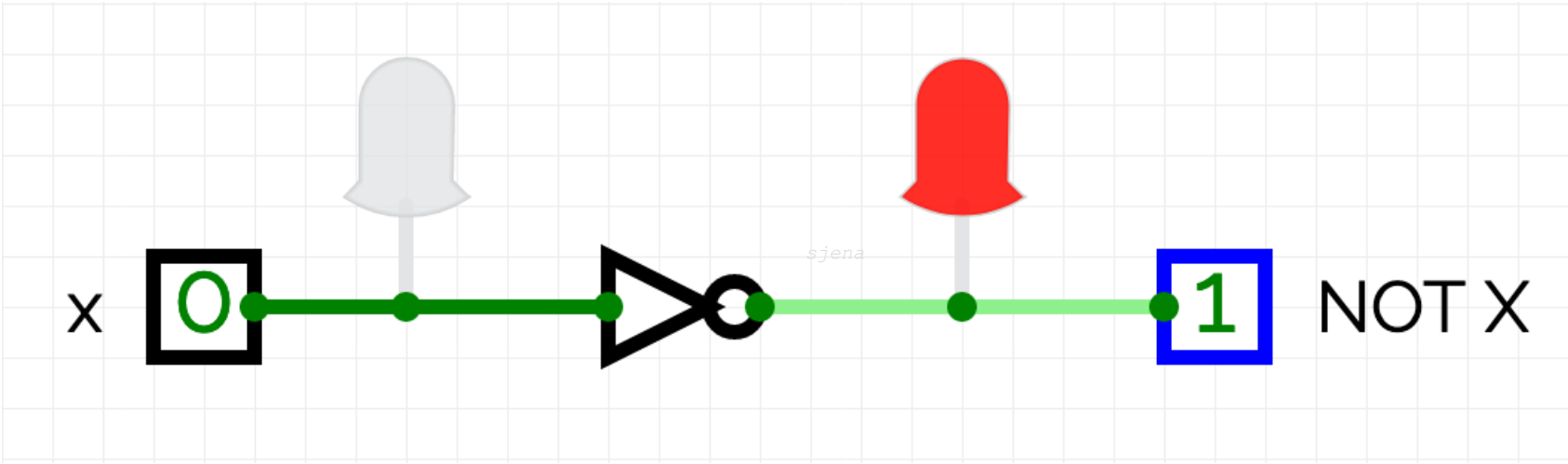
A simple circuit



When input is 1 (on), output is 1 (ON)

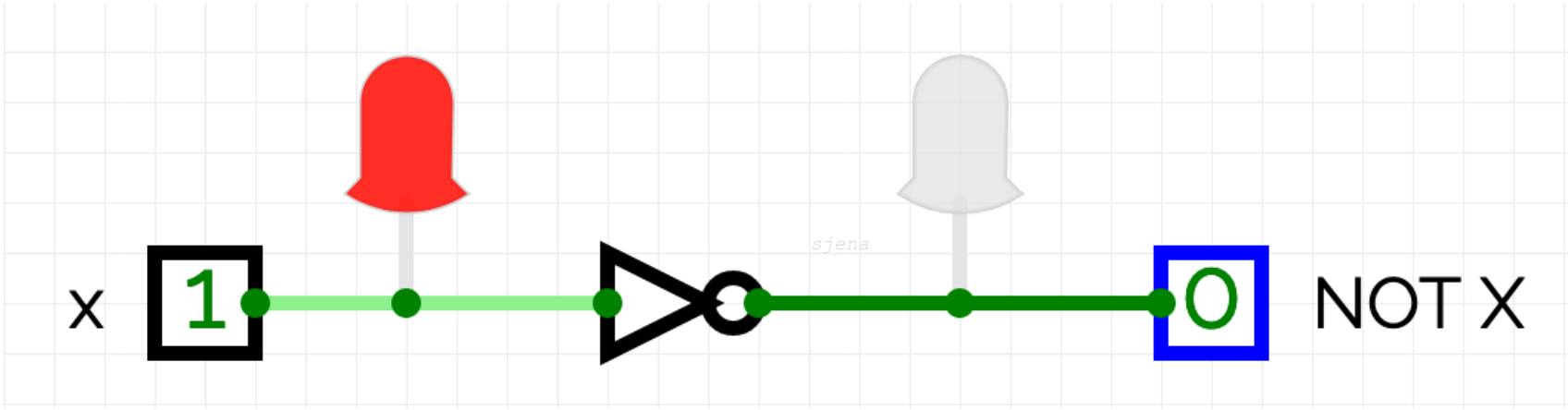


NOT



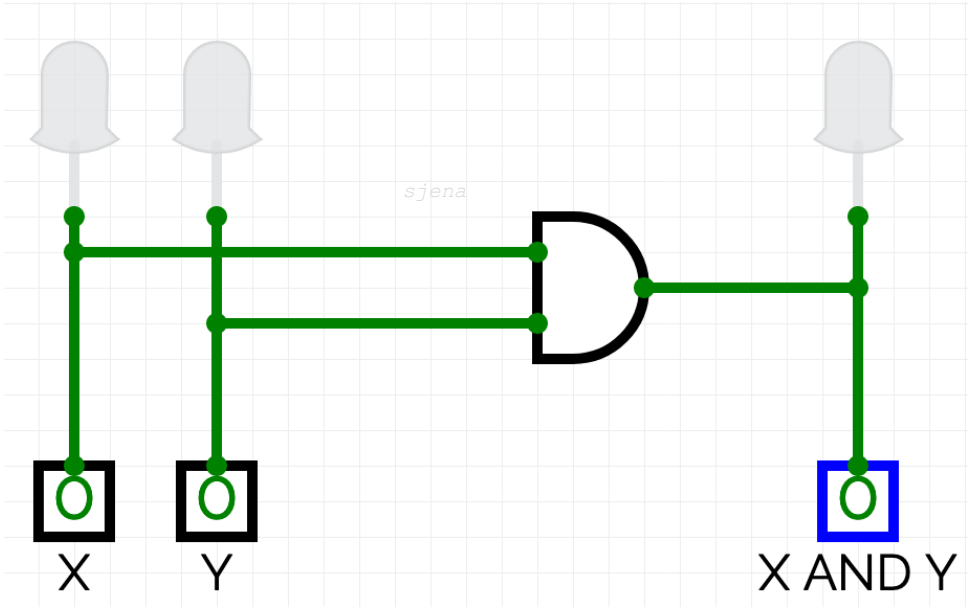
x	$\overline{x}$
0	1

NOT



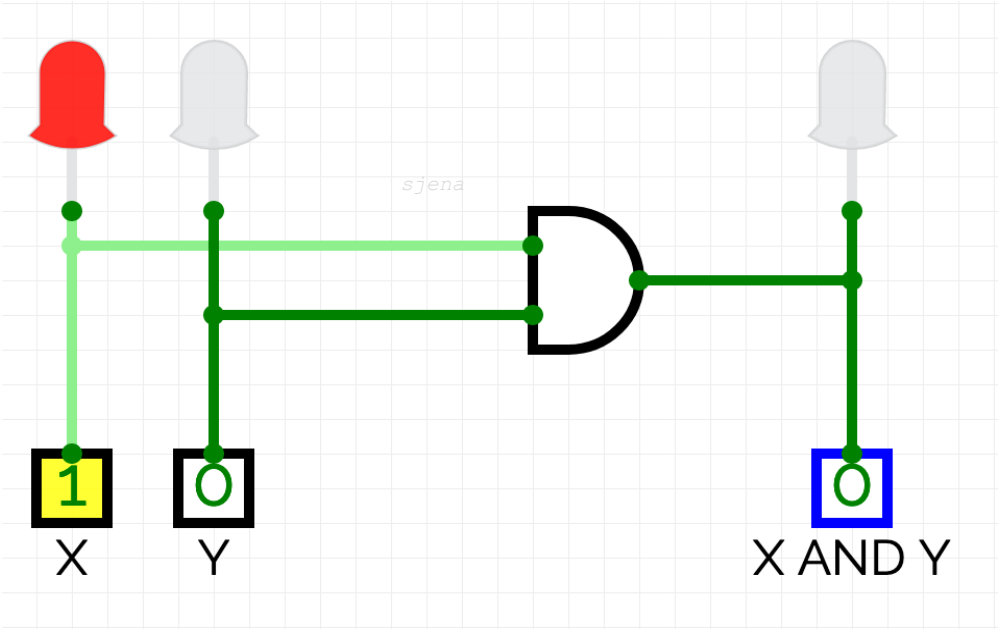
x	$\overline{x}$
0	1
1	0

AND



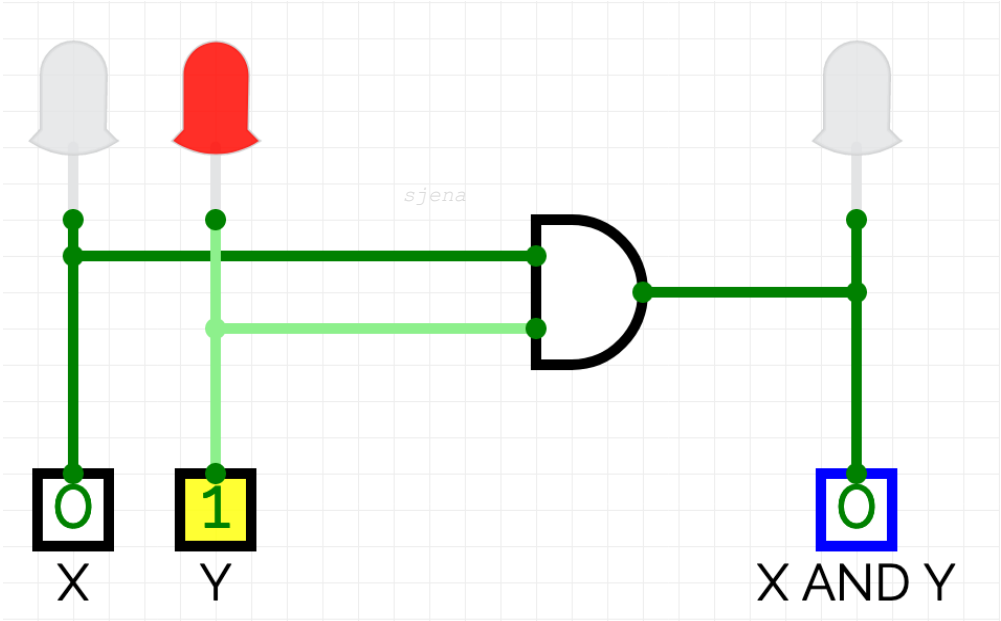
x	y	xy
0	0	0

AND



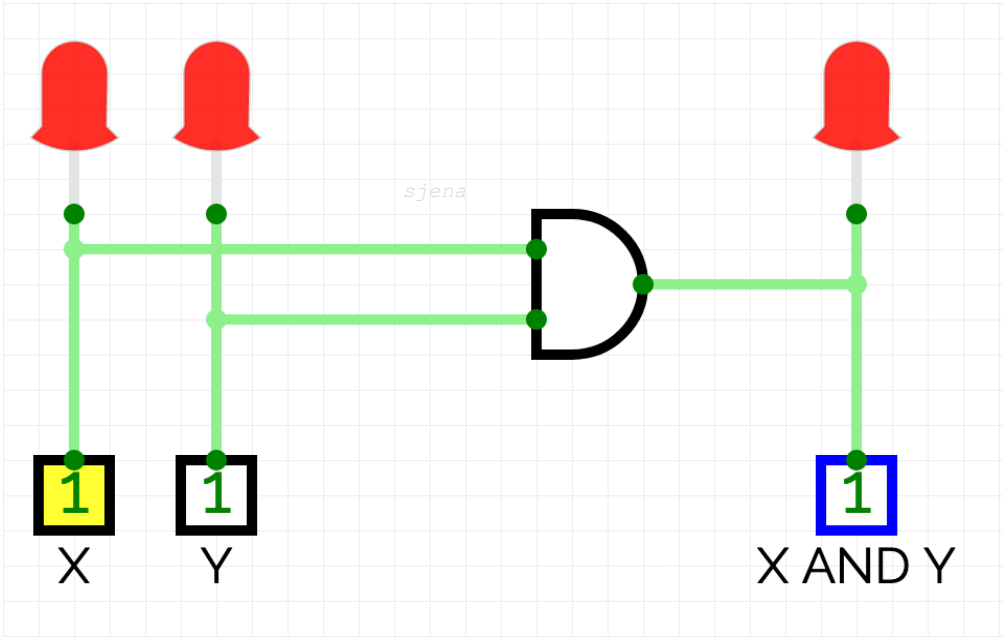
x	y	xy
0	0	0
1	0	0

AND



x	*y*	*xy*
0	0	0
1	0	0
0	1	0

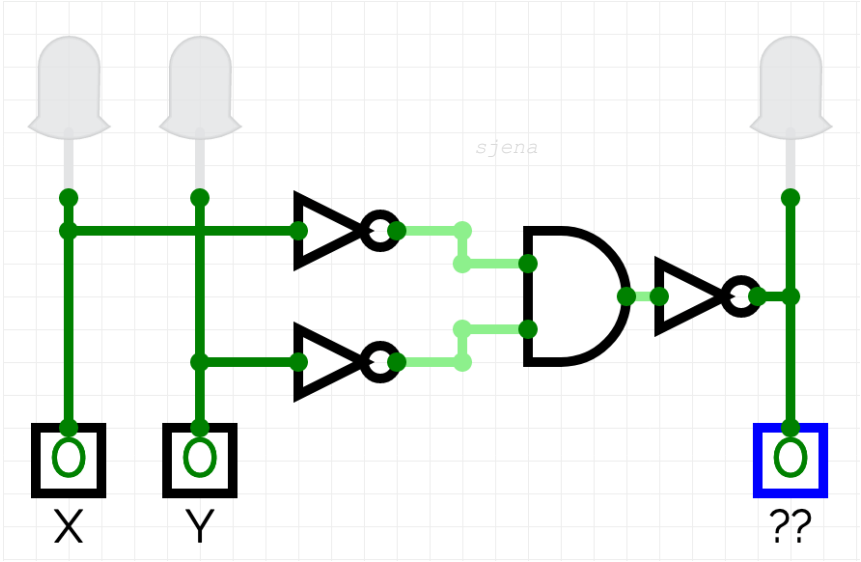
AND



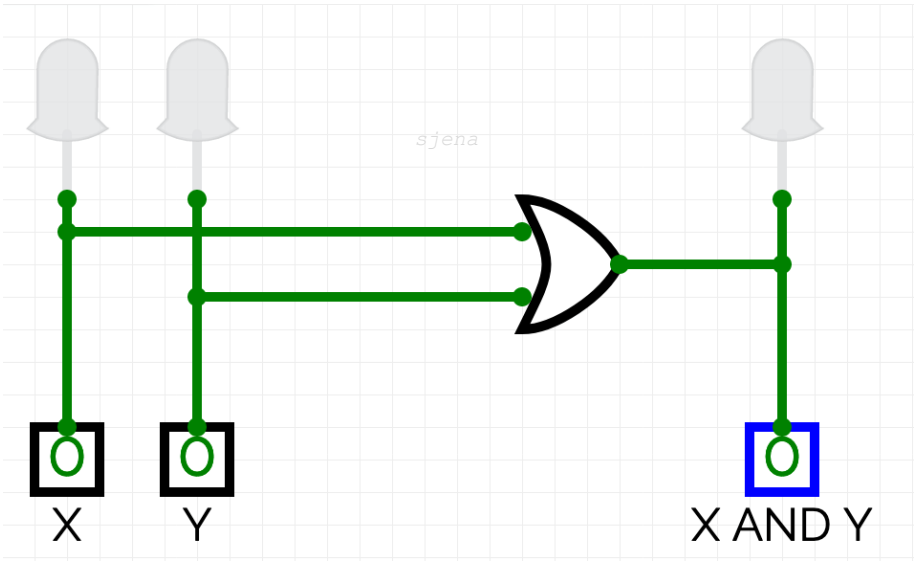
x	y	xy
0	0	0
1	0	0
0	1	0
1	1	1

OR

Method - 1



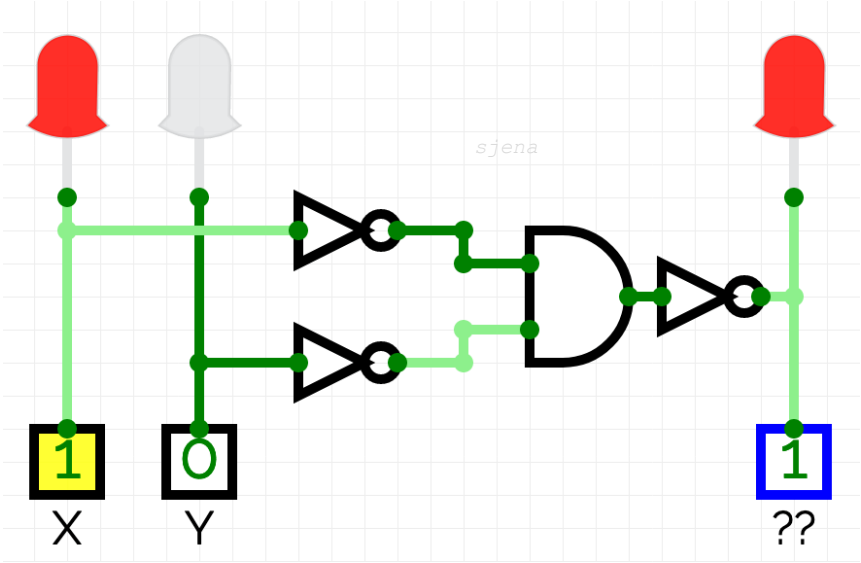
Method - 2



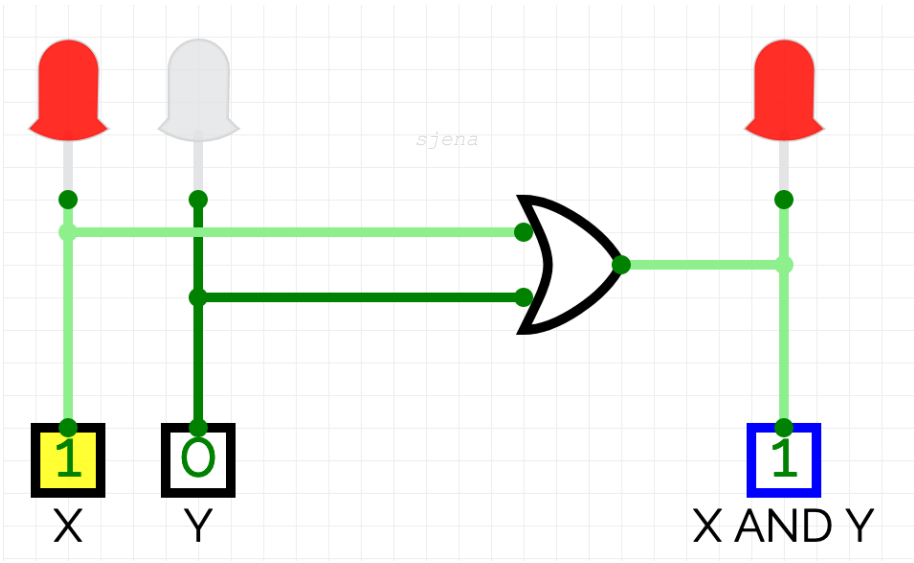
x	y	$x + y$
0	0	0

OR

Method - 1



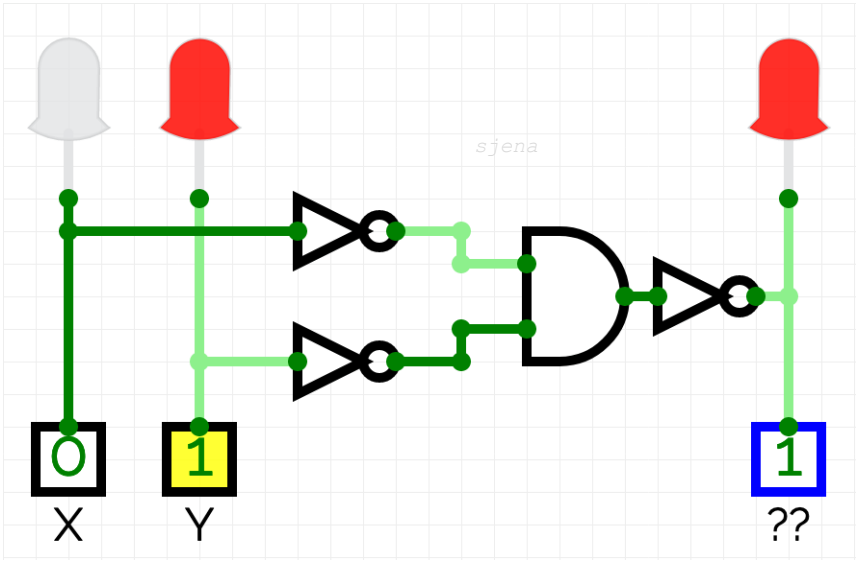
Method - 2



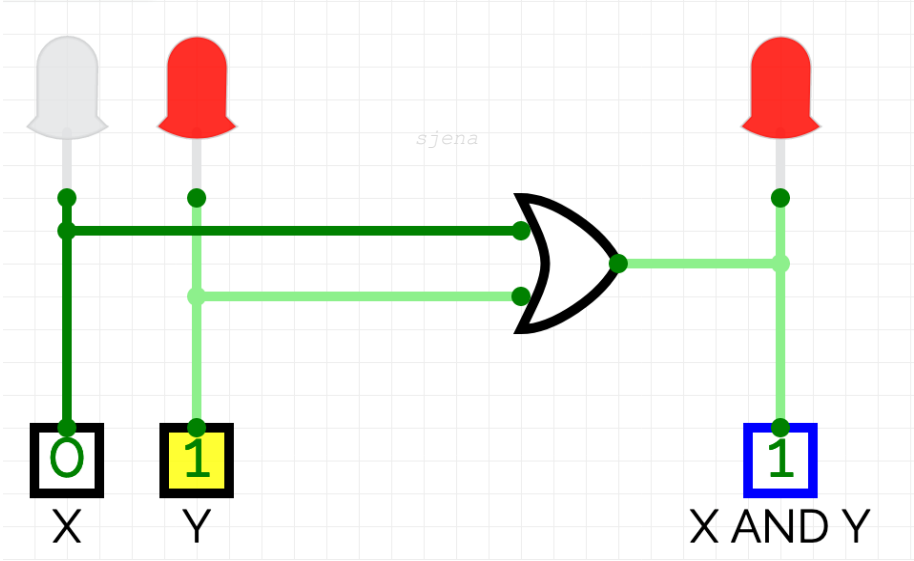
x	y	$x + y$
0	0	0
1	0	1

OR

Method - 1



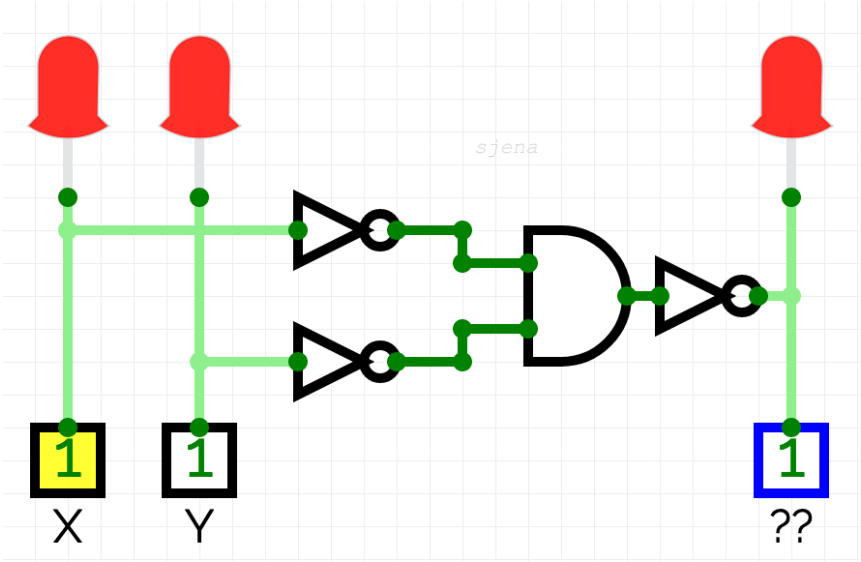
Method - 2



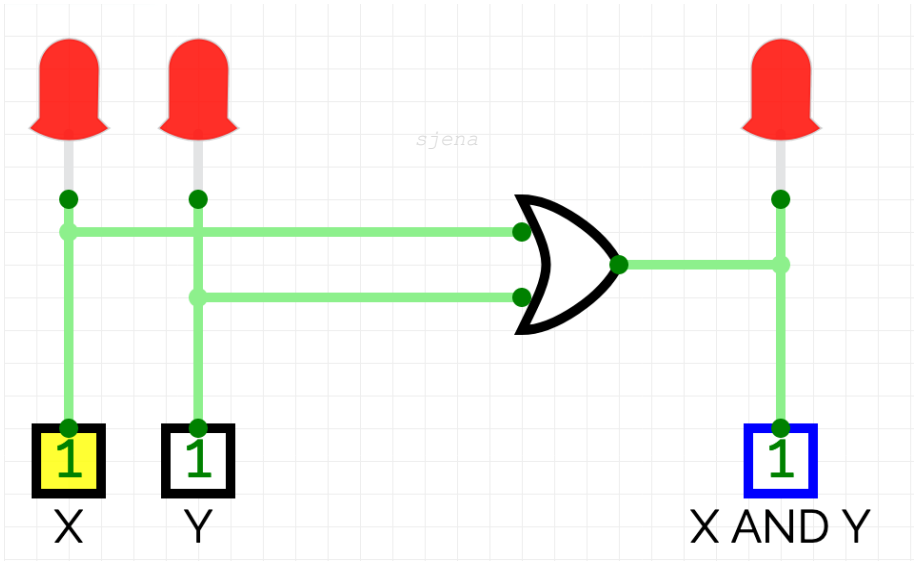
x	y	$x + y$
0	0	0
1	0	1
0	1	1

OR

Method - 1



Method - 2



x	y	$x + y$
0	0	0
1	0	1
0	1	1
1	1	1

We can make more complex wiring

Boolean Algebra: More complex example

$F(x,y,z) = x\bar{z}+y$

x	y	z	$\bar{z}$	$x\bar{z}$	$x\bar{z}+y$
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	1	0	1
0	1	1	0	0	1
1	0	0	1	1	1
1	0	1	0	0	0
1	1	0	1	1	1
1	1	1	0	0	1

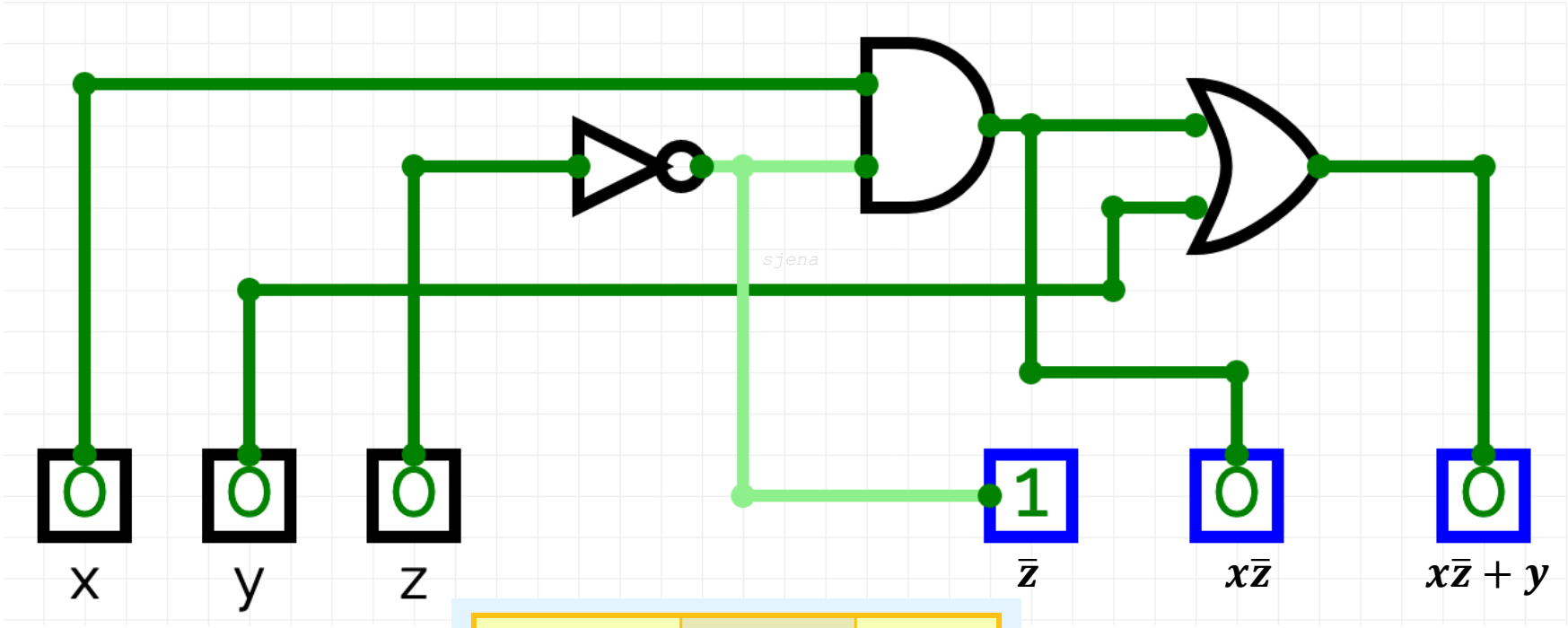
- The truth table for the Boolean function:

  $F(x,y,z) = x\bar{z}+y$

  is shown at the right.
- To make evaluation of the Boolean function easier, the truth table contains extra (shaded) columns to hold evaluations of subparts of the function.

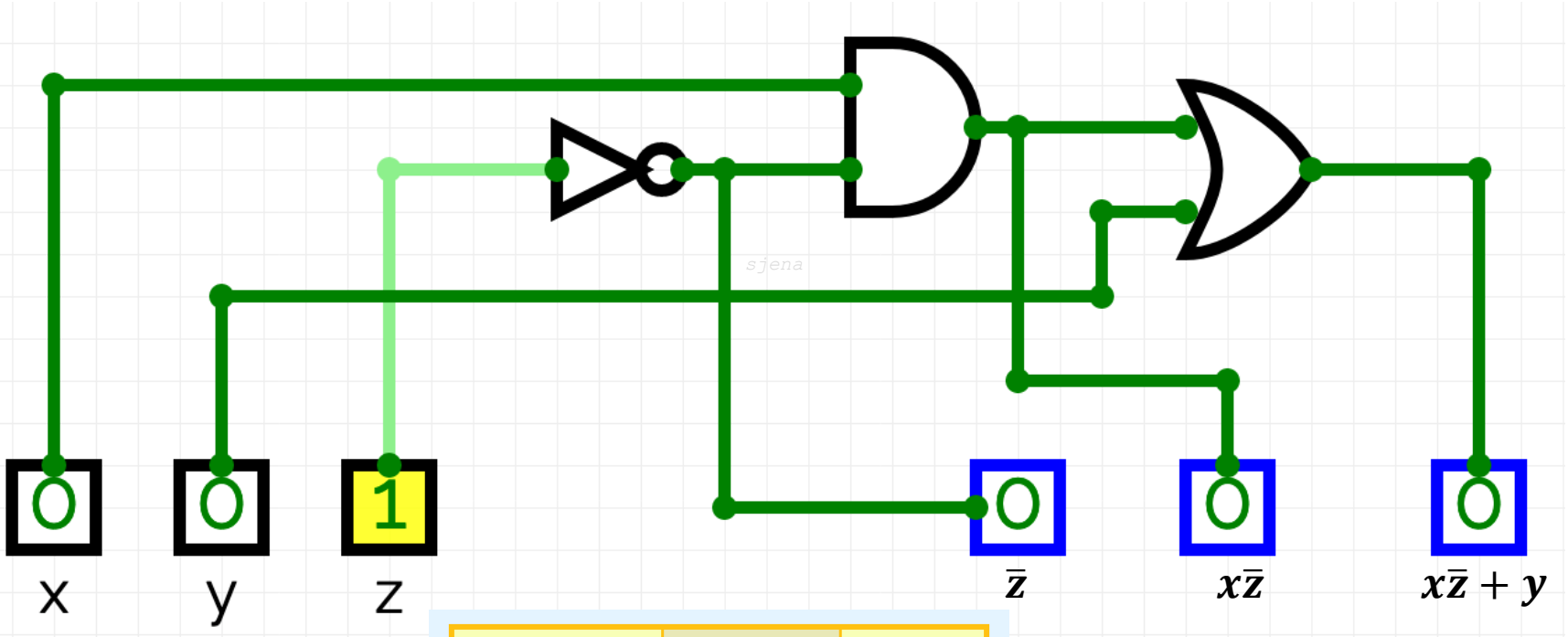
- As with common arithmetic, Boolean operations have rules of precedence.
- The NOT operator has highest priority, followed by AND and then OR.
- This is how we chose the (shaded) function subparts in our table.

Circuit: $F(x, y, z) = x\bar{z} + y$



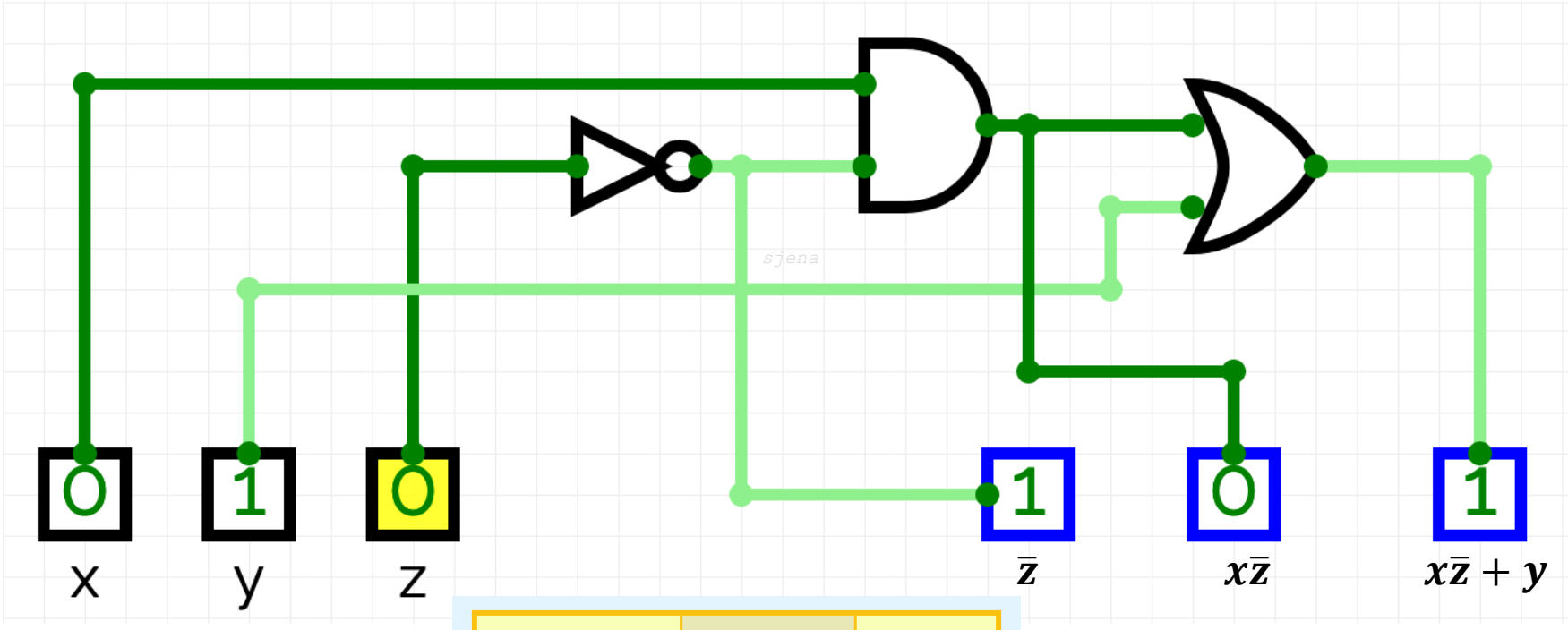
x	y	z	$\bar{z}$	$x\bar{z}$	$x\bar{z}+y$
0	0	0	1	0	0

Circuit: $F(x, y, z) = x\bar{z} + y$



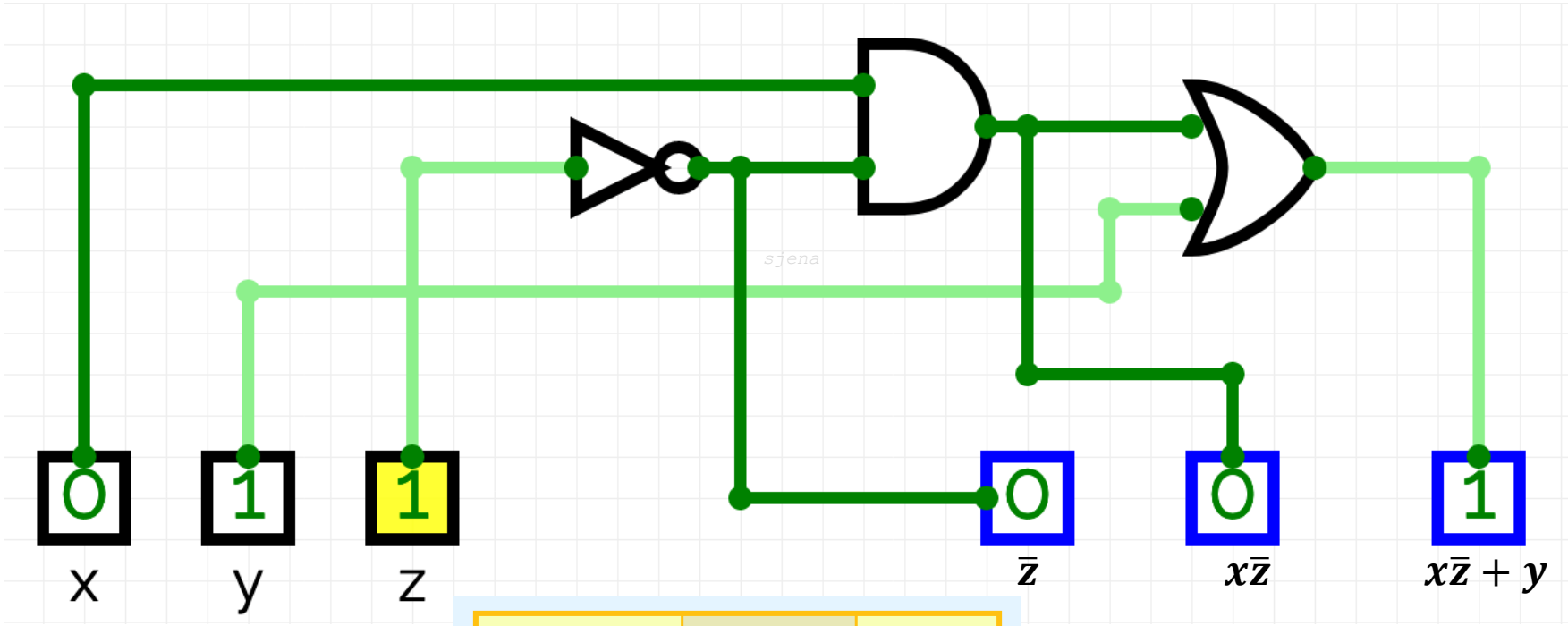
x	y	z	$\bar{z}$	$x\bar{z}$	$x\bar{z}+y$
0	0	0	1	0	0
0	0	1	0	0	0

Circuit: $F(x, y, z) = x\bar{z} + y$



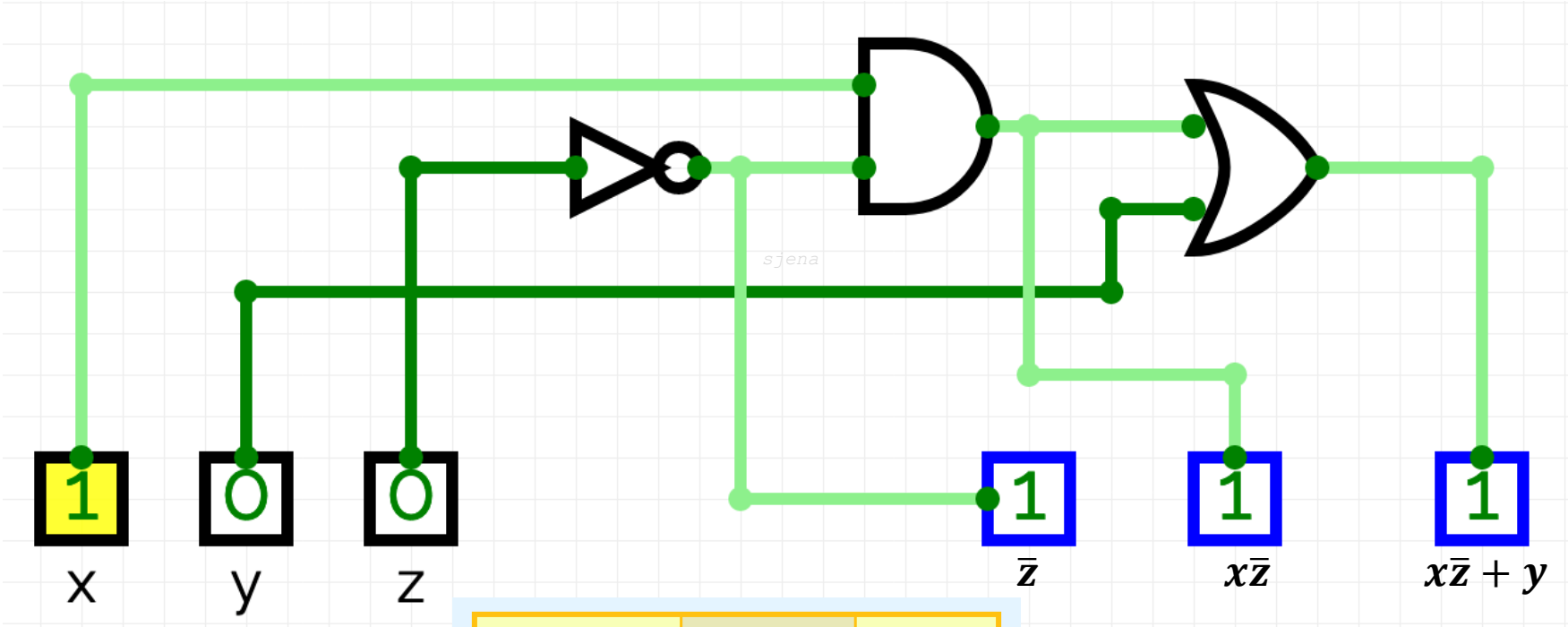
x	y	z	$\bar{z}$	$x\bar{z}$	$x\bar{z}+y$
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	1	0	1

Circuit: $F(x, y, z) = x\bar{z} + y$



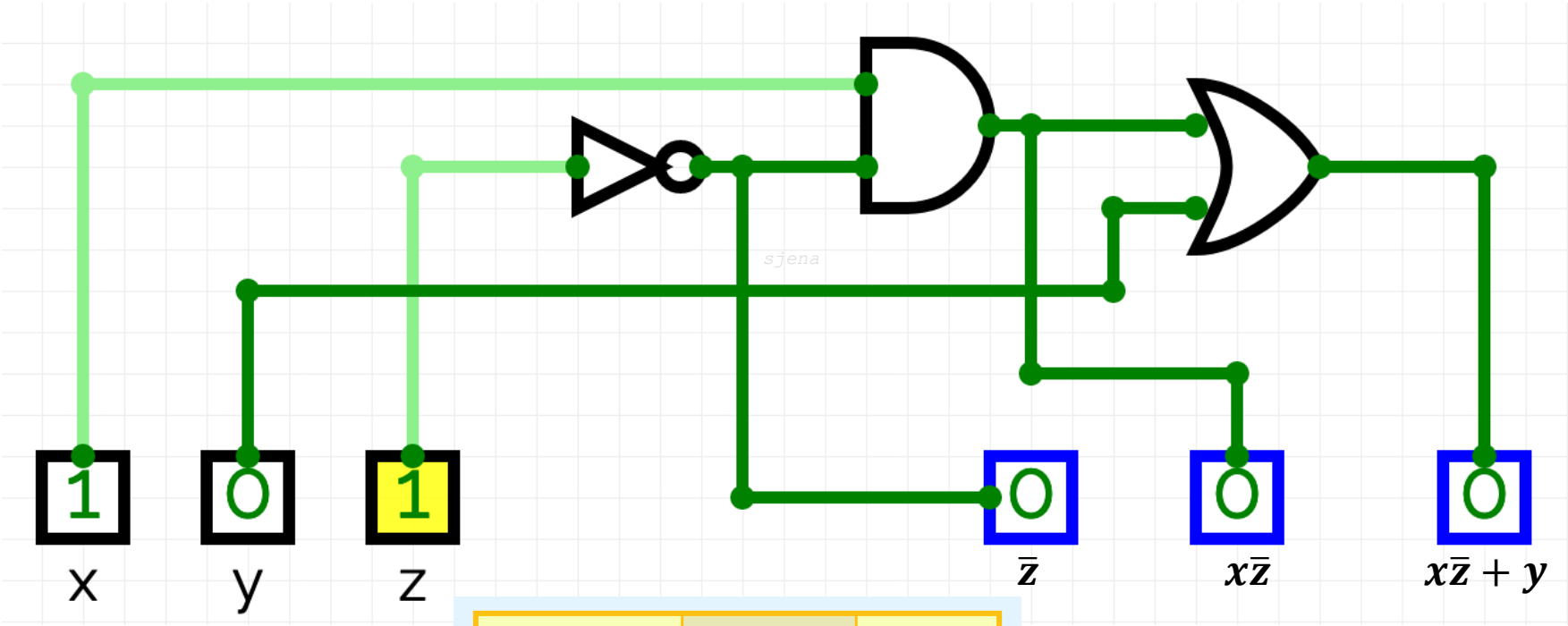
x	y	z	$\bar{z}$	$x\bar{z}$	$x\bar{z}+y$
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	1	0	1
0	1	1	0	0	1

Circuit: $F(x, y, z) = x\bar{z} + y$



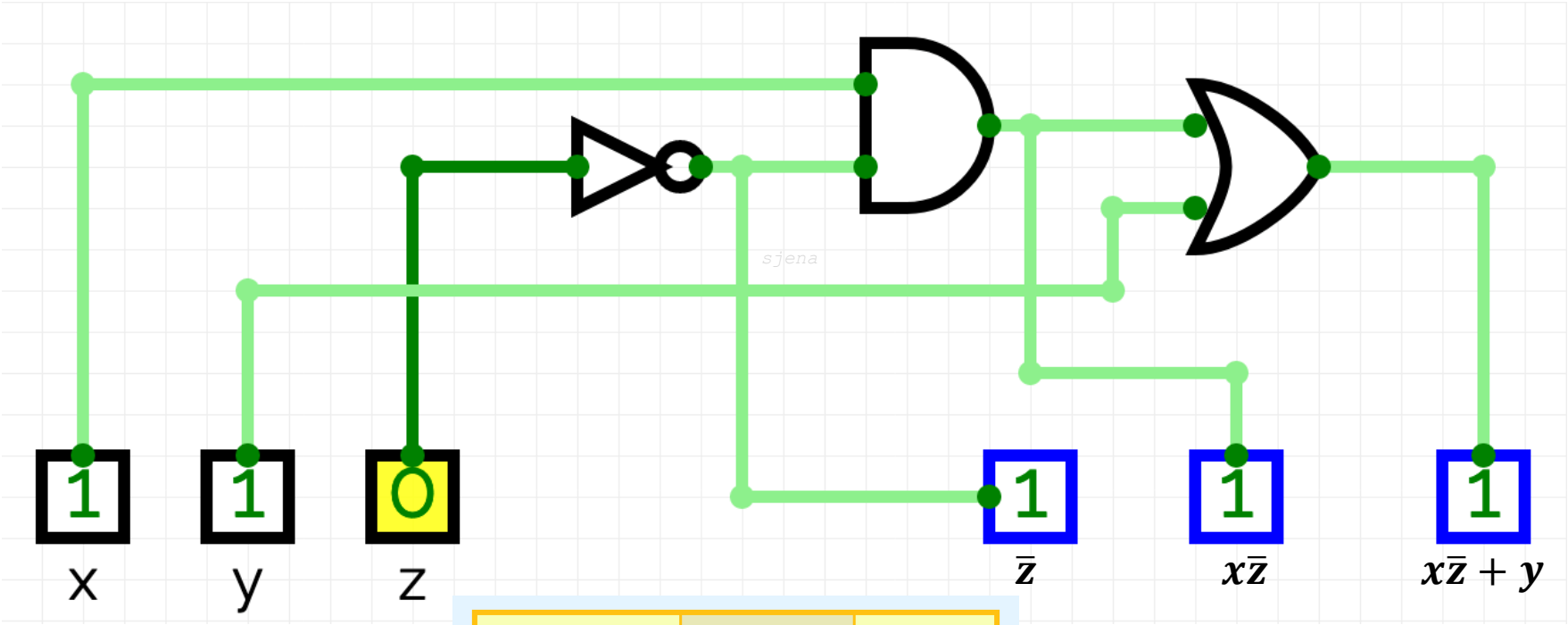
x	y	z	$\bar{z}$	$x\bar{z}$	$x\bar{z}+y$
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	1	0	1
0	1	1	0	0	1
1	0	0	1	1	1

Circuit: $F(x, y, z) = x\bar{z} + y$



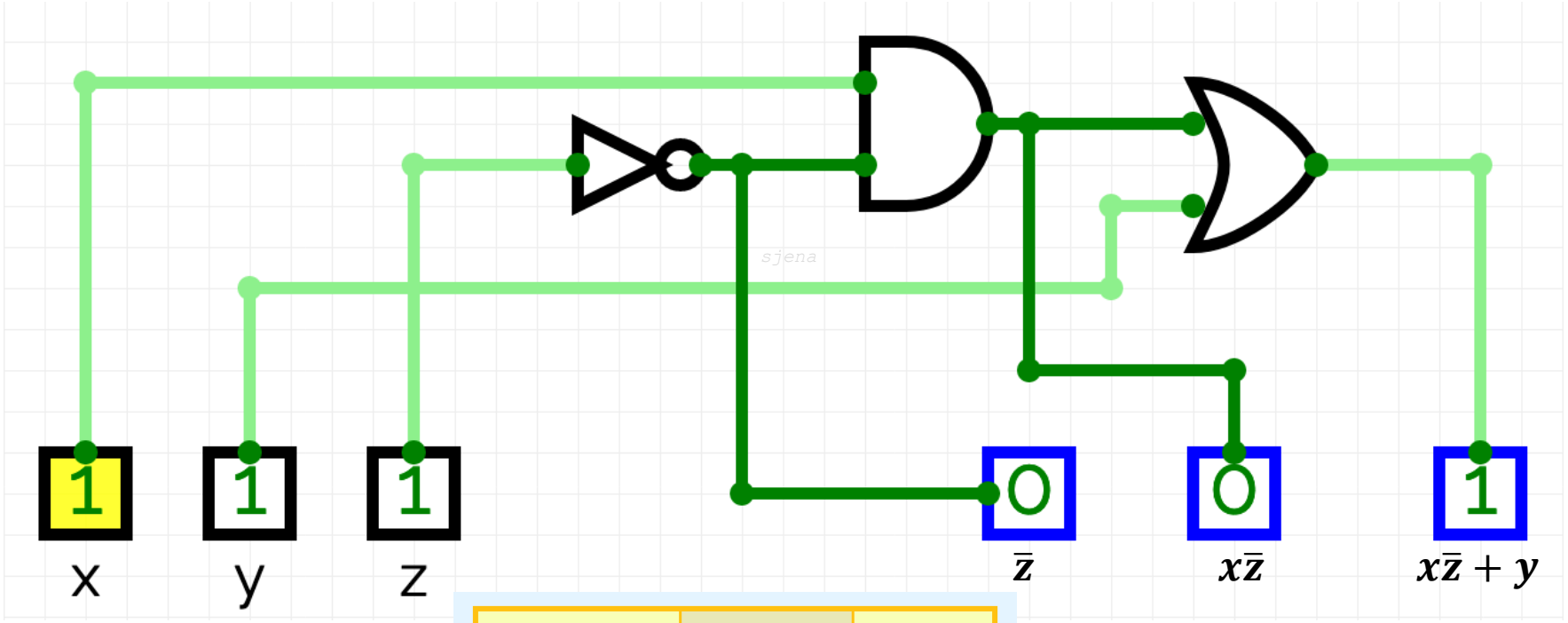
x	y	z	$\bar{z}$	$x\bar{z}$	$x\bar{z}+y$
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	1	0	1
0	1	1	0	0	1
1	0	0	1	1	1
1	0	1	0	0	0

Circuit: $F(x, y, z) = x\bar{z} + y$



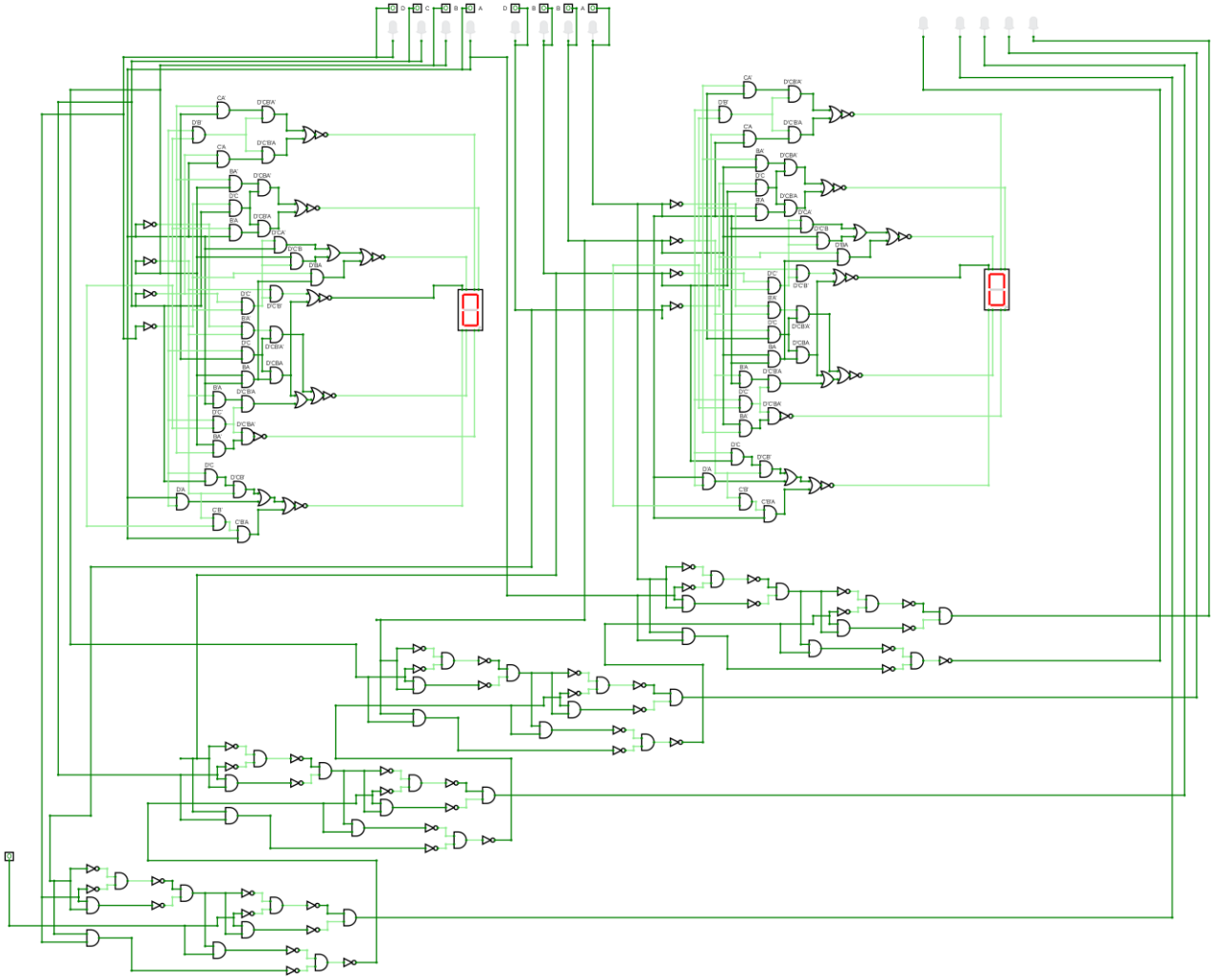
x	y	z	$\bar{z}$	$x\bar{z}$	$x\bar{z}+y$
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	1	0	1
0	1	1	0	0	1
1	0	0	1	1	1
1	0	1	0	0	0
1	1	0	1	1	1

Circuit: $F(x, y, z) = x\bar{z} + y$



x	y	z	$\bar{z}$	$x\bar{z}$	$x\bar{z}+y$
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	1	0	1
0	1	1	0	0	1
1	0	0	1	1	1
1	0	1	0	0	0
1	1	0	1	1	1
1	1	1	0	0	1

Trying to build a ALU



Using only basic gates