Introduction to Digital Electronics

Motivation

There are so many applications that we use in our daily life which are built using digital electronics.

Take for instance something as simple as a calculator and something as complex as a computer. The principles on which these systems are built are the same.

What do these systems do?

> They can calculate quite well. Very well.
>
> They do what they are told. You push a button, and the computer/calculator does what you want.
>
> They store and move information (data).

So, what are these systems made up of?

Billions of these switches together make the processor that is running all the software on your computer.

It all begins with common sand, which consists mostly of silicon oxide. Using chemical methods, the sand is converted to pure silicon which is a semiconductor. This means we can control the conductivity of this material - we can switch an electric current in silicon on or off, at will, and very, very fast. So, we can make electrical switches from silicon!

But how do these silicon switches actually perform computations and move data?

This is what we'll be exploring. This is called switch logic, or Boolean logic.

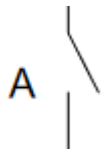
Switch logic: NAND and NOR.

These are the first circuits we explore: NAND and NOR.

Why these circuits you ask? That's a good question.

NAND and NOR gates are widely known to be universal logic gates, meaning that any other logic gate be made from NAND or NOR gates.

To keep the material manageable in length and complexity, we'll be introducing a very abstract version of silicon switches:



How does this switch work?

When A is asserted (that is, it is above a threshold voltage), the switch closes, and a connection is maintained.

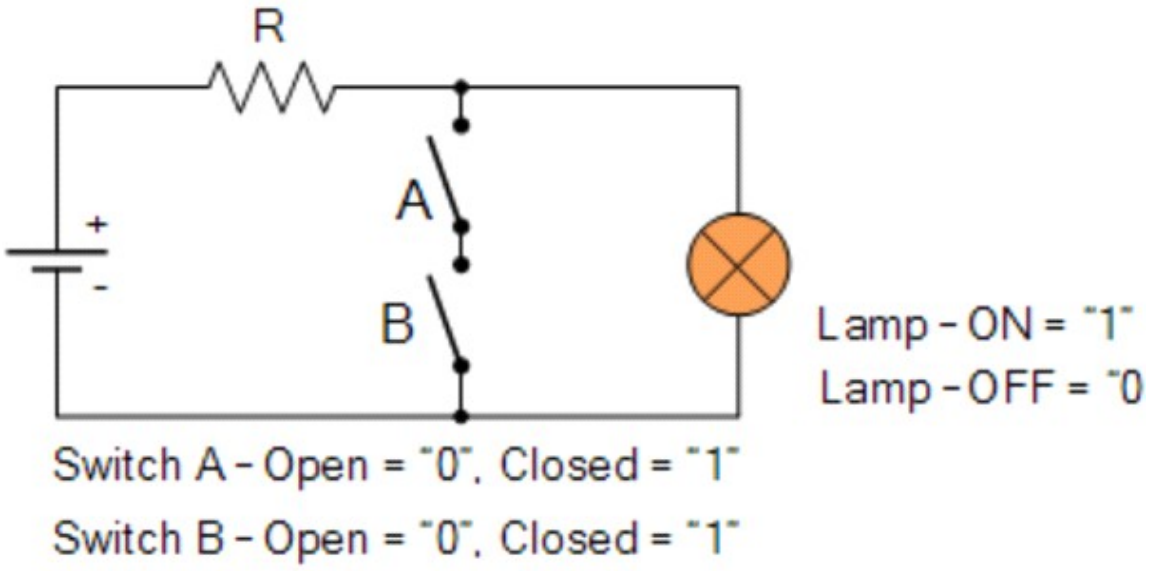
When A is not asserted (that is, it is below a threshold voltage), the switch remains open, and no connection is maintained.

Let us denote A by "1" when it asserted, and "0" when it is not.

You can also denote them as "Monkey" and "Donkey" if you wish, but we'll stick to the standard 0s and 1s.

"1" is also known as a digital high and "0" is known as a digital low.

NAND:



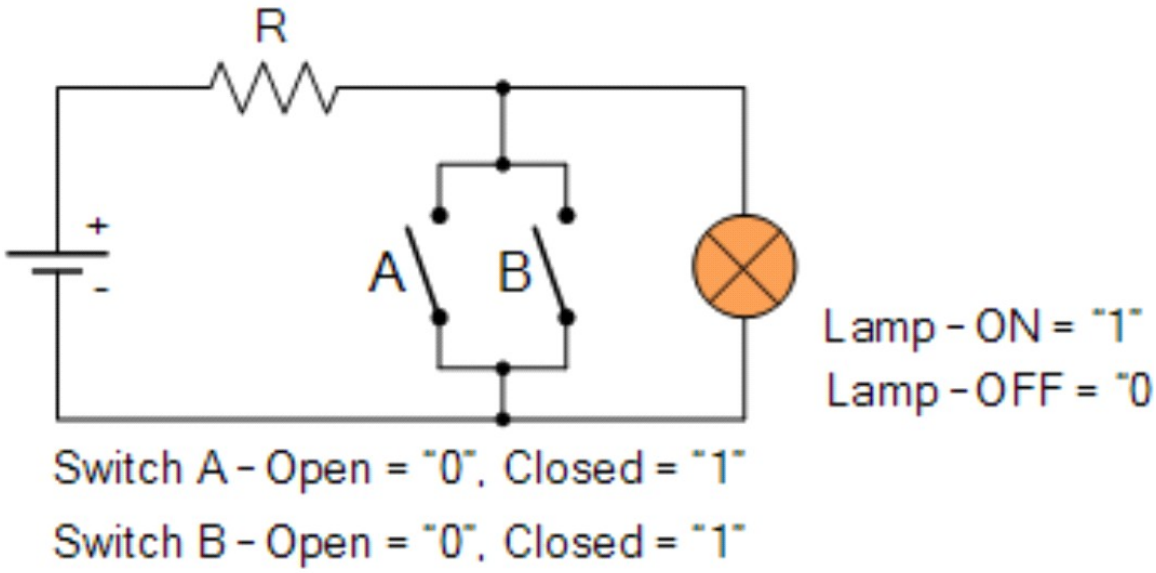
From the above circuit, it should be clear that the lamp is turned off (which we'll be denoting as 0) only when the switches A and B are closed i.e. A = 1 and B = 1. For all other combinations of A and B, the lamp will be turned on (which we'll be denoting as 1).

Below is a table which summarizes the behavior of the circuit with A and B as the input and the lamp's state as the output:

A	B	Output
0	0	1
0	1	1
1	0	1
1	1	0

The above table is known as a **Truth Table**.

NOR:



Truth Table for NOR gate circuit:

A	B	Output
0	0	1
0	1	0
1	0	0
1	1	0

Logic gates are like the atoms of computation and storage.

I encourage you to read up on how other logic gates such as AND, OR, XOR, etc. can be designed using NAND or NOR gates. I have referenced an article below which details on the same.

https://www.allaboutcircuits.com/textbook/digital/chpt-3/gate-universality/

Logic Gates: https://www.youtube.com/playlist?

list=PLTd6ceoshprfc_VVJYunO1BN9peCTMQgr

What do we do with these switches?

A single switch can only represent “true/false”, “1/0”, etc. We call such a single “binary” value a “bit”. A group of “bits” (thus a group of 0’s and 1’s) can represent anything you want.

Go through the following videos to learn how to represent numbers using binary:

Unsigned numbers: https://youtu.be/cJNm938Xwao

Signed numbers: https://youtu.be/4qH4unVtJkE

So, now depending on what you tell the computer to do, a bunch of 0s and 1s can be used to represent text on the screen, or numbers that it can perform arithmetic on, etc.

How does a computer perform arithmetic on these 0s and 1s i.e., binary values?

Let's consider the simplest arithmetic of all: Addition.

While it may seem very trivial to our brain which has evolved over centuries, it is still quite neat and ingenious how we humans got an electronic circuit to do it for us!

Go through the following videos to learn how we can use basic logic gates to add bits:

Half Adder: https://youtu.be/aLUY-s7LSns

Full Adder: https://youtu.be/RK3P9L2ZXk4

Now, that we got logic gates to perform some basic arithmetic on numbers for us, it is time to store these results!

Go through the following videos to learn how to use basic logic gates to create memory elements:

Latches and Flip-Flops: https://www.youtube.com/playlist?list=PLTd6ceoshpreKyY55hA4vpzAUv9hSut1H

Noise Margin

Is there a reason why digital signals are preferred over analog signals to design systems?

Digital signals are immune to noise whereas Analog signals aren't. It is important for these signals to be immune to noise given the amount of electromagnetic radiation present in the atmosphere. These EM radiations can easily alter the voltage levels which can lead to misinterpretation of data.

To understand why digital signals are more immune to noise, watch the following video:

The Digital Abstraction: https://youtu.be/4TCnYYpZxEc