

IoT Product Design and Coding Bootcamp

Brian Rashap, Ph.D.

24-Feb-2020

Table of contents

1 Introduction

- Instructors
- Industry 4.0

2 Teensy

- Teensy 3.2 Pin Layout
- Teensy IDE

3 Basic Circuits and Coding - Part I

4 Communications Buses

5 The Internet

6 Particle Argon

- Particle Argon Pin Layout

IoT Fun



Brian Rashap, Ph.D.

- Proud husband of Krista and father of Shelby (21) and Ethan (18)
- Electrical Engineer with 25 years industrial experience
- High School track coach
- Hobbies: running, cycling, reading, spending time with family



Introductions

INTRODUCTIONS

Class Rules

- Respect Each Other, Help Each Other
- Ask Questions
- Be On Time (let's us know via Slack if you won't be here)
- Keep Your Workspace and the Classroom Neat and Tidy
- If you are struggling, let myself, Susan, or Esteban know. We are here to HELP!

Grading

- ① In class assignments - 30
- ② Individual smart device 20
- ③ Capstone 40
- ④ Assessments 10

More information later today on how assignments are turned in

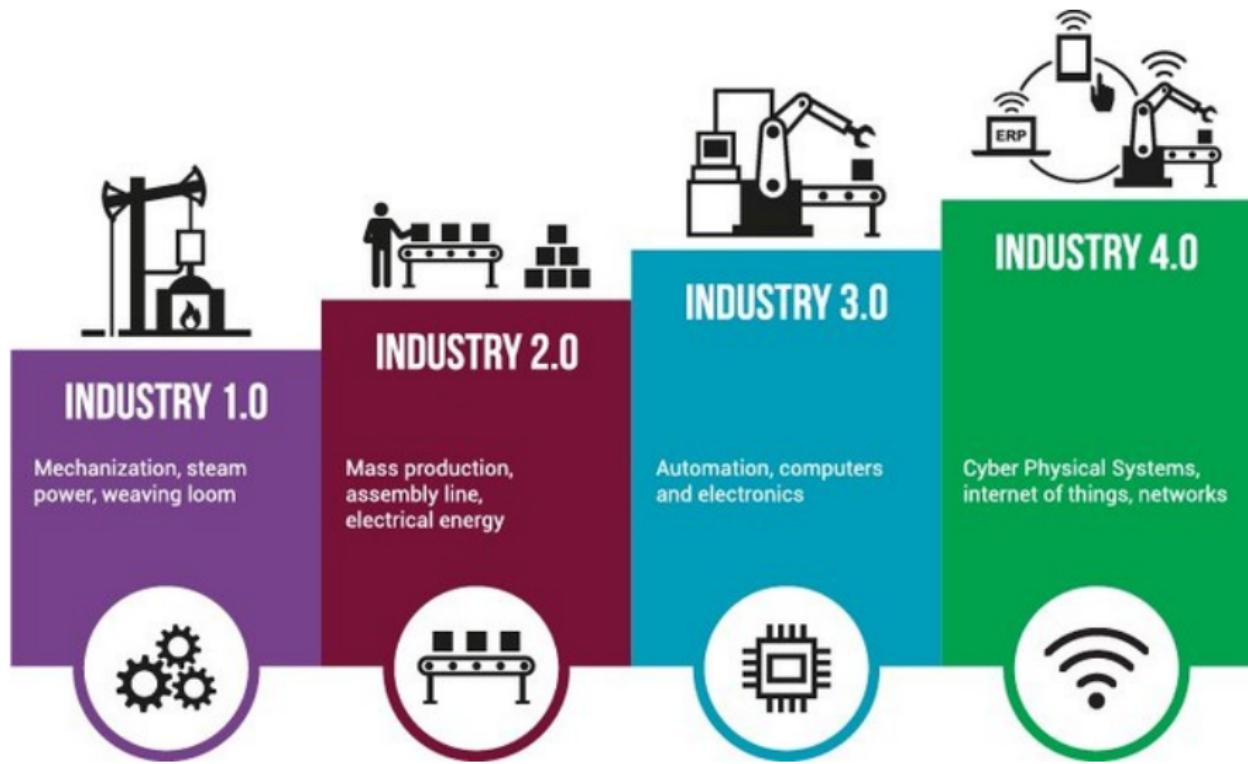
GITHUB Simplified

```
1 // In Powershell go to ./Documents/<yourname>
2 // Get a repository that already exists and pull
   it into your local machine
3 git clone <URL of repository>
4
5 // From the repository directory, get updates
6 git pull
7
8 // Send your changes up to the repository
9 git add .
10 git commit -m "<comment>"
11 git push
12
13 // You may get asked to enter your GIT username
14 git config --global user.email "you@example.com"
```

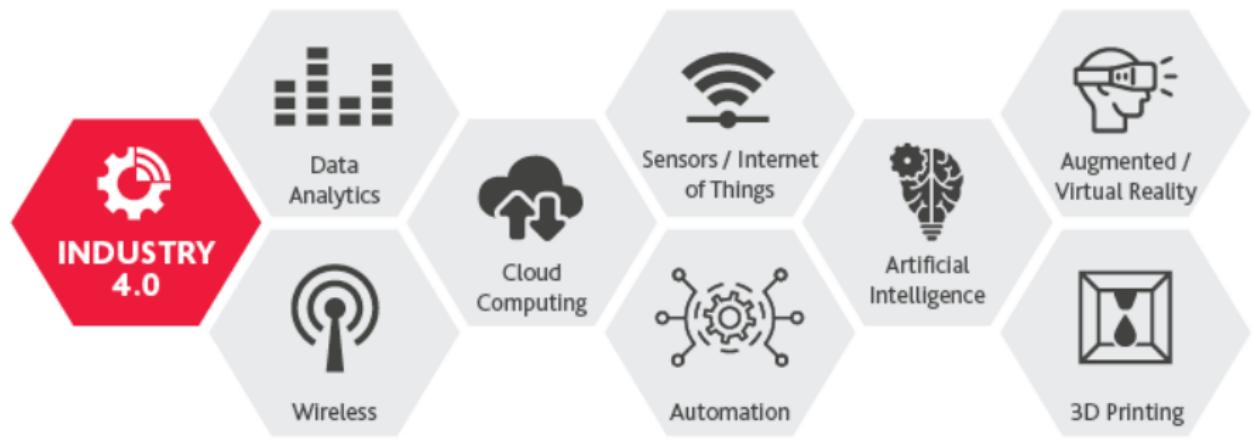
Powershell Commands Simplified

```
1 mkdir <directory name> //make a directory
2 rmdir <directory name> //remove directory
3 ls //list contents of current directory
4 pwd //print path of current directory
5 cd <directory name> // change directory
6 cd .. //change directory to the directory above
7 new-item <file name> //create a file
8 cp <file name> <new file name> //copy file
9 mv <file name> <directory> //move file to new dir
10 rm <file name> //delete file
11 cat <file name> //display contents of file
```

Evolution of Industry

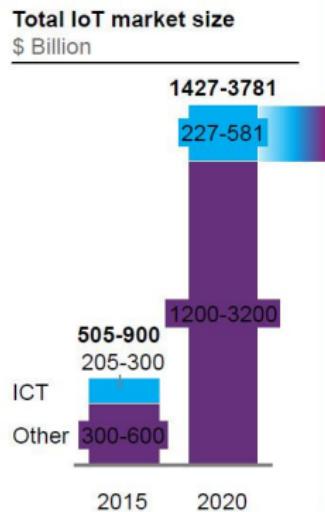


Components of Industry 4.0



IoT Market Size

Resulting in a large IoT market size, of which a significant component is ICT



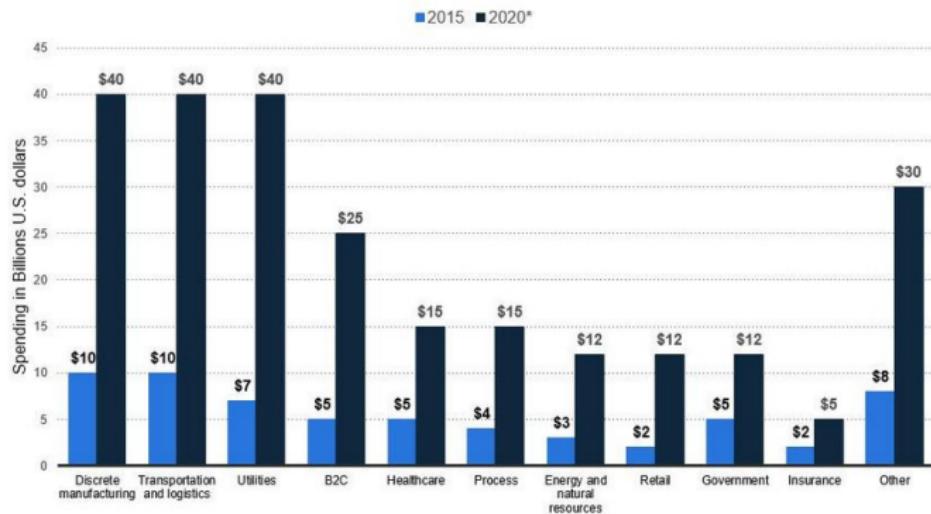
2020 IoT market size (Base case – Upside), \$ Billion		CAGR (%, 2015-20)	OP (%, 2020)
Solution services ¹	59-176	7-20%	10-35%
Security ²	34-36	7-19%	10-30%
SW infra-structure & apps	80-219	7-20%	15-30%
Connectivity	10-29	8-19%	0-10%
Hardware	44-121	8-19%	0-10%
Total	227-581	7-19%	8%

1 Solution services not considered technical products

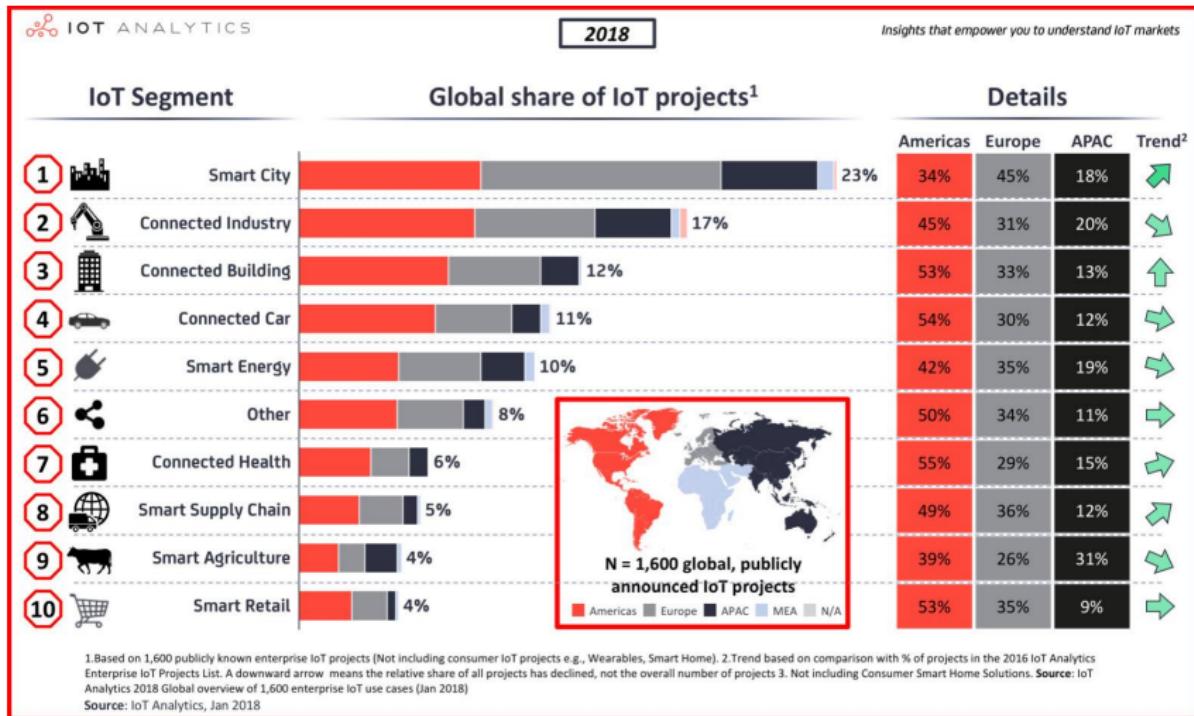
2 \$3-\$6B is IoT security and \$25-\$35B is for mobile security

Spending by IoT Vertical

Spending on Internet of Things Worldwide by Vertical in 2015 and 2020
(in billions of U.S. dollars)



IoT Segments

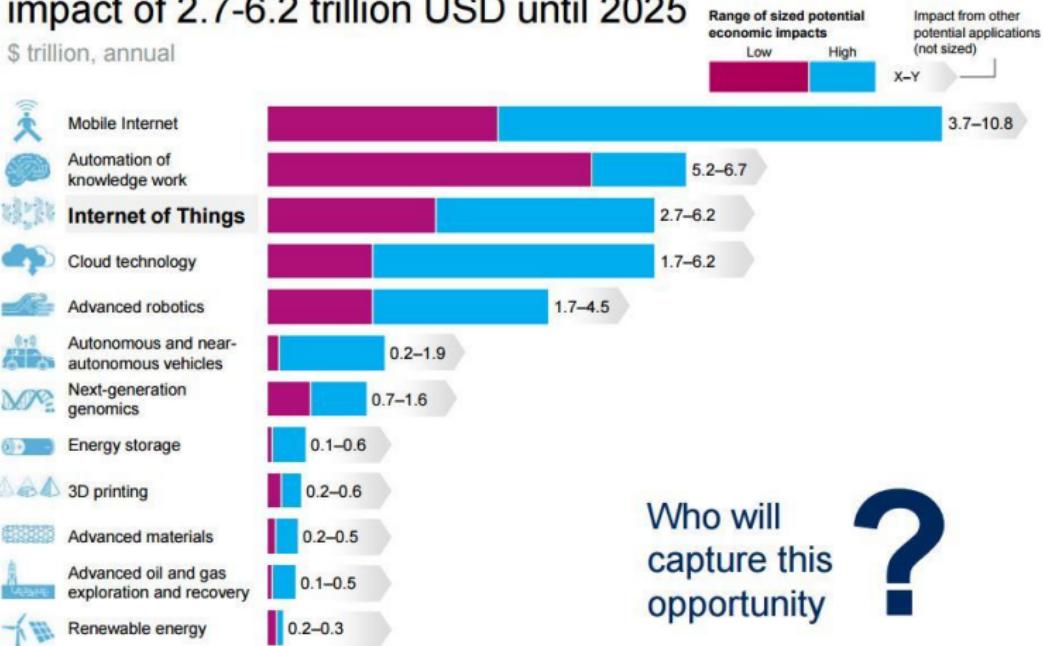


IoT Potential Economic Impact

THE IoT PLATFORM OPPORTUNITY

The Internet of Things (IoT) has a potential economic impact of 2.7-6.2 trillion USD until 2025

\$ trillion, annual



Who will capture this opportunity?



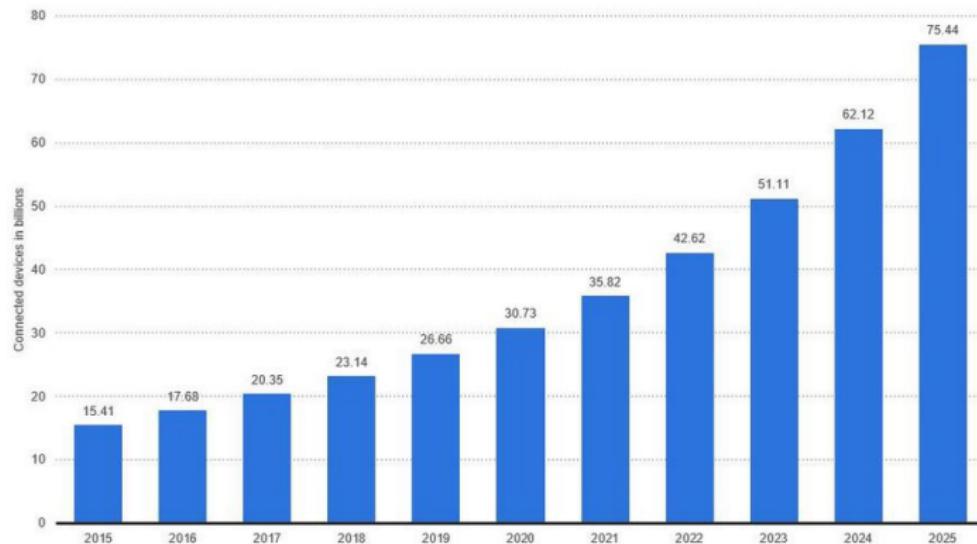
SOURCE: McKinsey Global Institute analysis

McKinsey & Company 3

IoT Growth

Internet of Things - number of connected devices worldwide 2015-2025

Internet of Things (IoT) connected devices installed base worldwide from 2015 to 2025 (in billions)

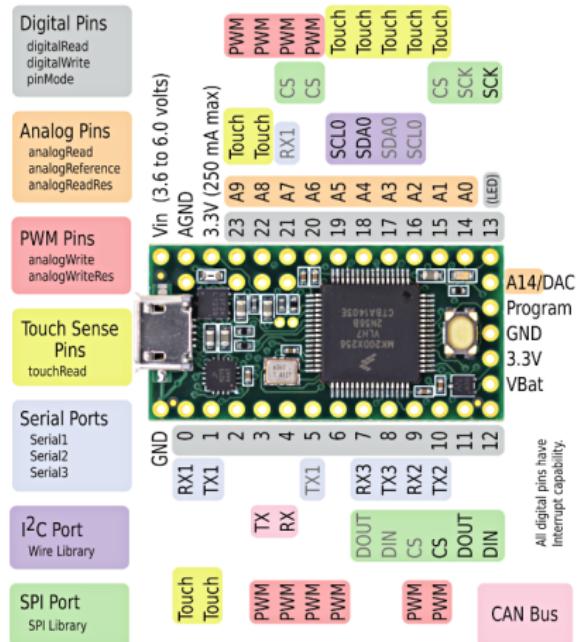


Our First Microcontroller



Teensy 3.2

- Cortex-M4 72MHz (overclocked to 96 MHz)
- 34 GPIO pins
- 3.3V and 5.0V operating voltages
- 500mA of available power with USB



Arduino IDE for Teensy

We are going to start off using the Arduino IDE¹. The Arduino IDE is programmed essentially using C++ code, but make the compiling and loading onto the microcontroller simpler.

We begin by installing the Arduino IDE:

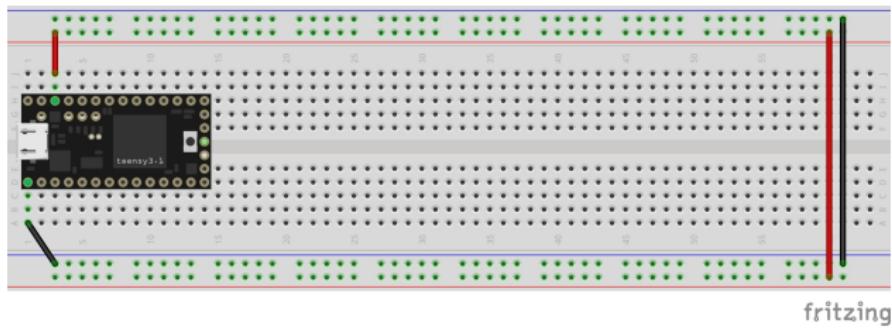
<https://www.arduino.cc/en/main/software>

Then, we install the Teensyduino add-on:

https://www.pjrc.com/teensy/td_download.html

¹An IDE, or Integrated Development Environment, enables programmers to consolidate the different aspects of writing a computer program.

Teensy on Breadboard

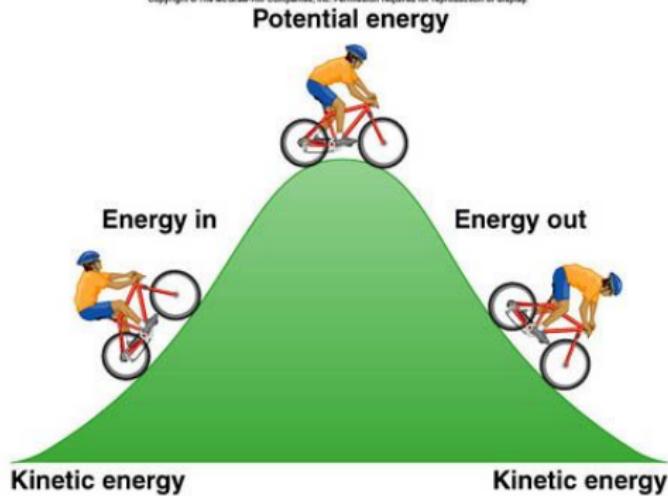


Basic Structure of Arduino Sketch

```
1 // heading is used to document and include
2 packages. It is also used to configure the
3 coding environment.
4
5 void setup() {
6     // put your setup code here, to run once:
7 }
8
9 void loop() {
10    // put your main code here, to run repeatedly:
11 }
12 }
```

Energy

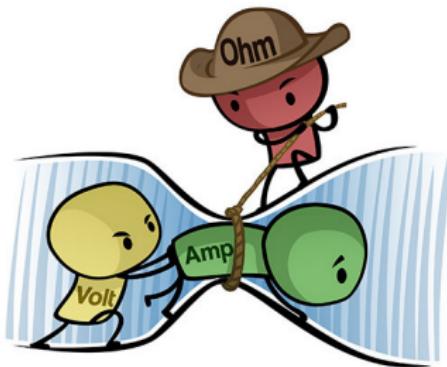
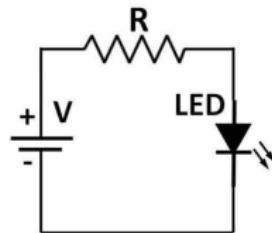
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.



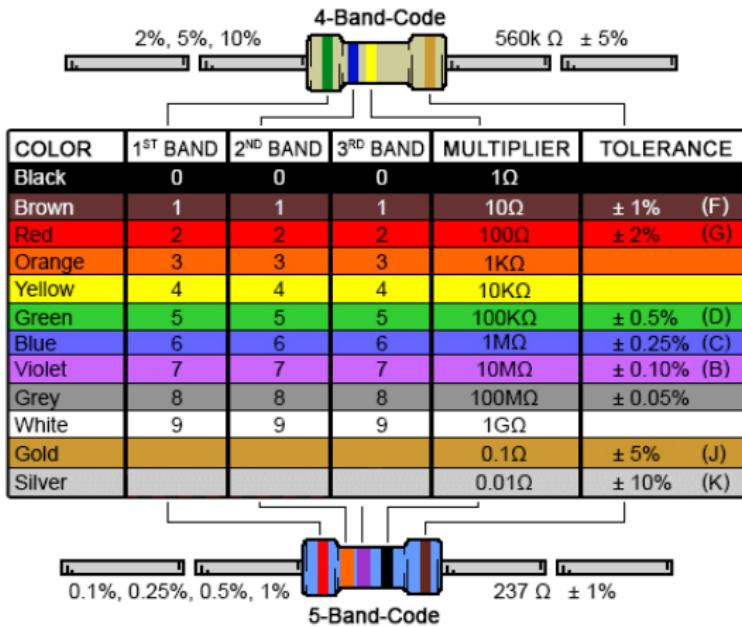
Ohm's Law

Ohm's Law

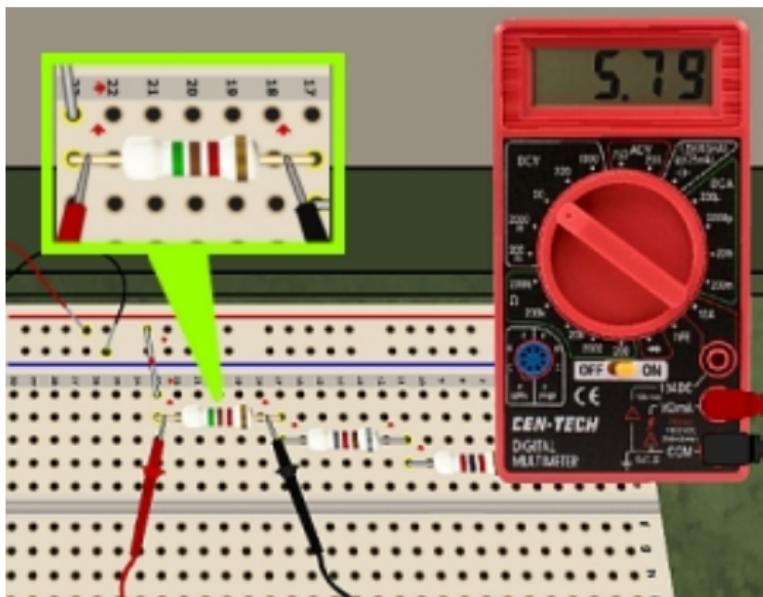
$$V = I * R$$



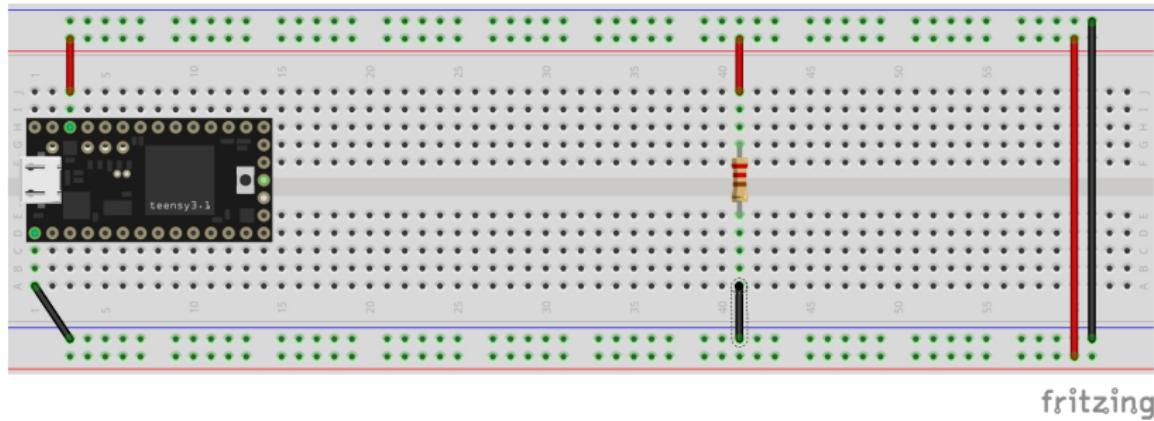
Resistor Color Bands



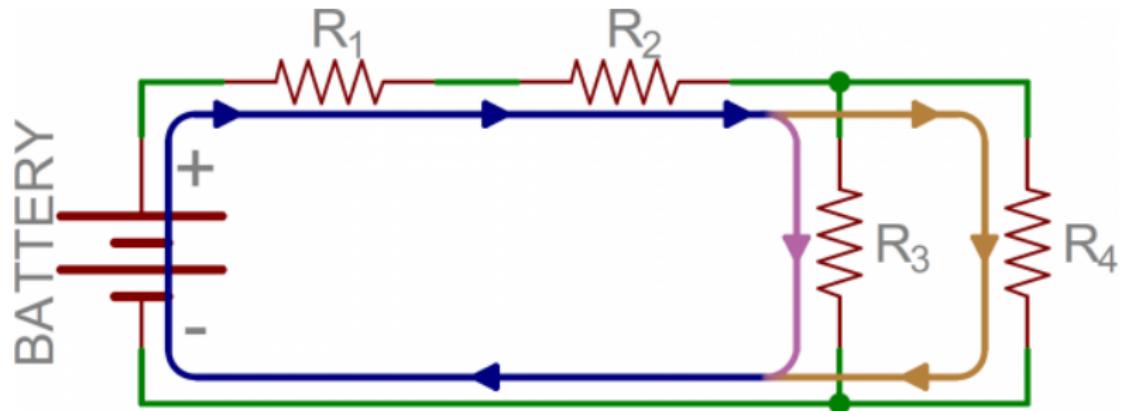
Measuring Voltage, Current, and Resistance



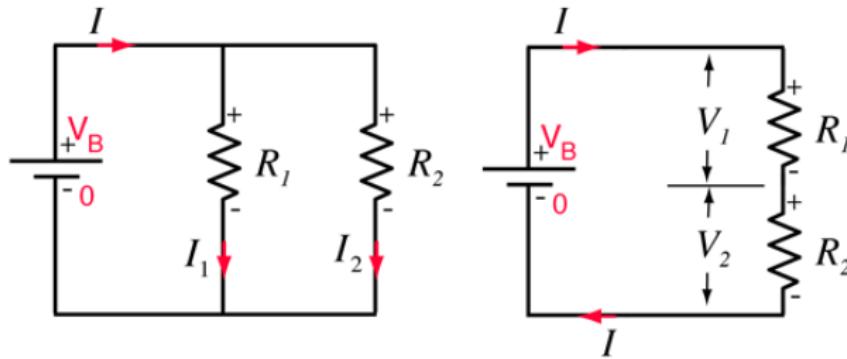
One Resistor



Resistors in Series and Parallel



Resistors in Series and Parallel



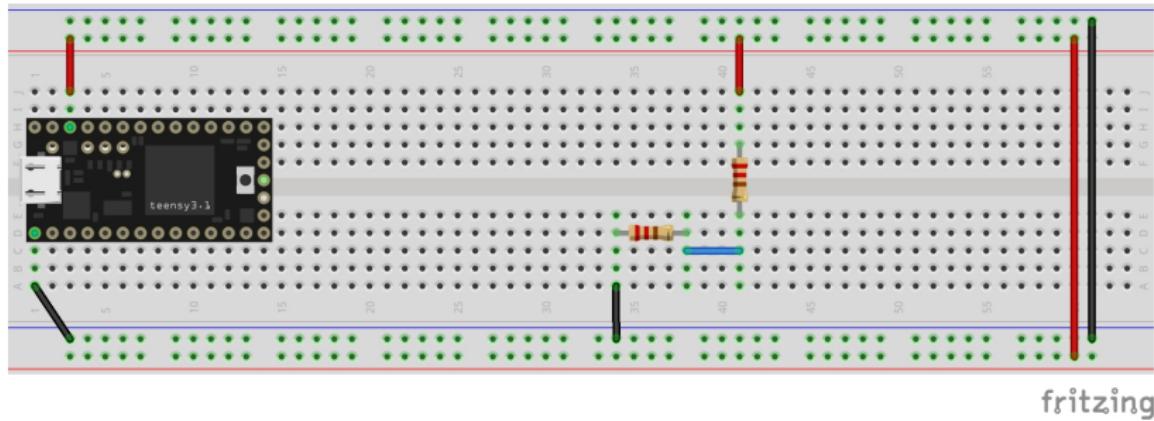
Parallel resistors

$$\frac{1}{R_{\text{equivalent}}} = \frac{1}{R_1} + \frac{1}{R_2}$$

Series resistors

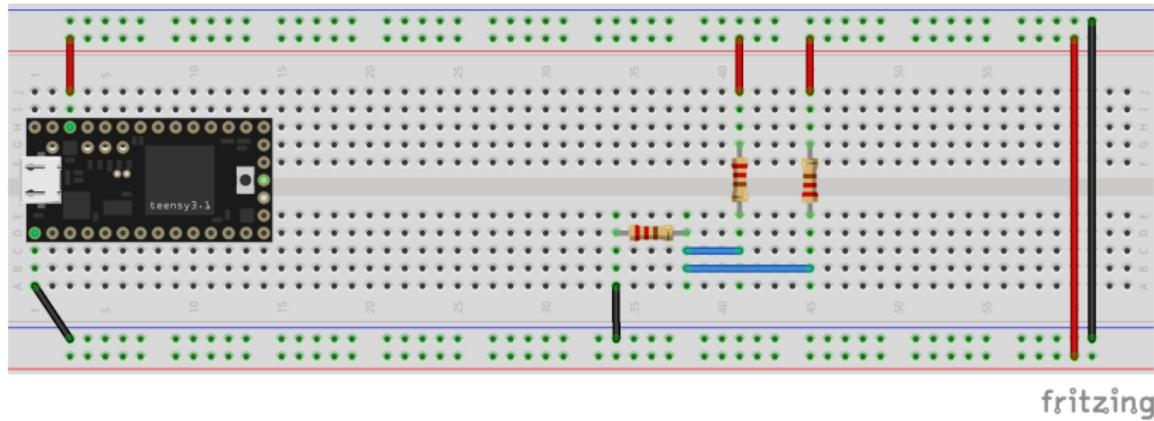
$$R_{\text{equivalent}} = R_1 + R_2$$

Resistors in Series



fritzing

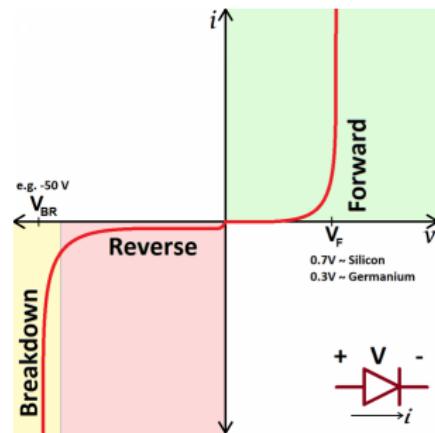
Resistors in Series and Parallel



fritzing

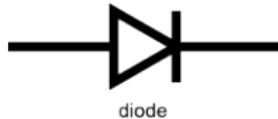
Diodes

The key function of a diode is to control the direction of current-flow. Current passing through a diode can only go in one direction, called the forward direction. Current trying to flow the reverse direction is blocked.



Light Emitting Diodes

LEDs (that's "ell-ee-dees") are a particular type of diode that convert electrical energy into light.



diode



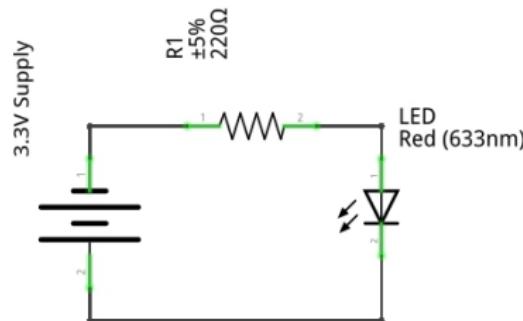
light emitting diode

Current Limiting Resistors

As a LED has very little resistance, when it is connected directly to a power supply, the current draw will exceed its specs and it will burn out.

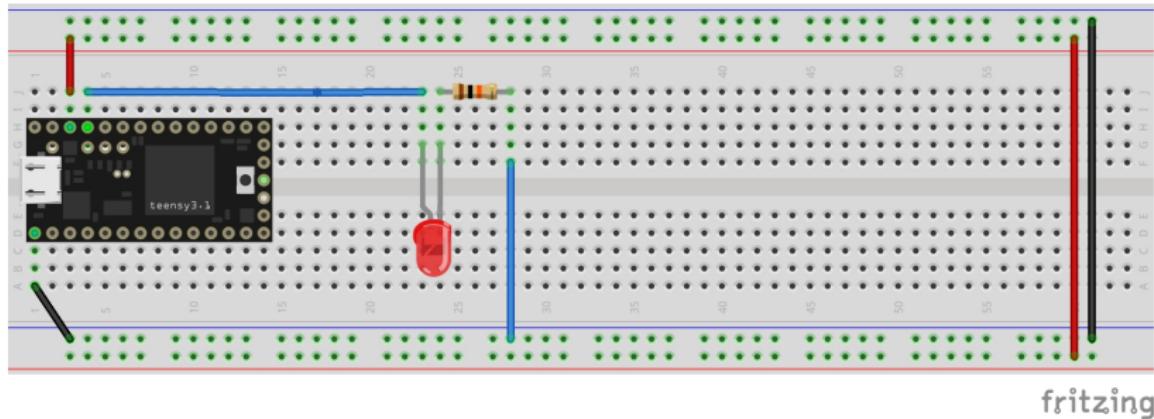
$$V_{pp} - V_{LED} = IR \implies R \geq \frac{V_{pp} - V_{LED}}{I_{max}}$$

For a 3.3V power supply, a 0.43V across the LED, and a max current of 100mA, the resistor needs to be greater than 29Ω .



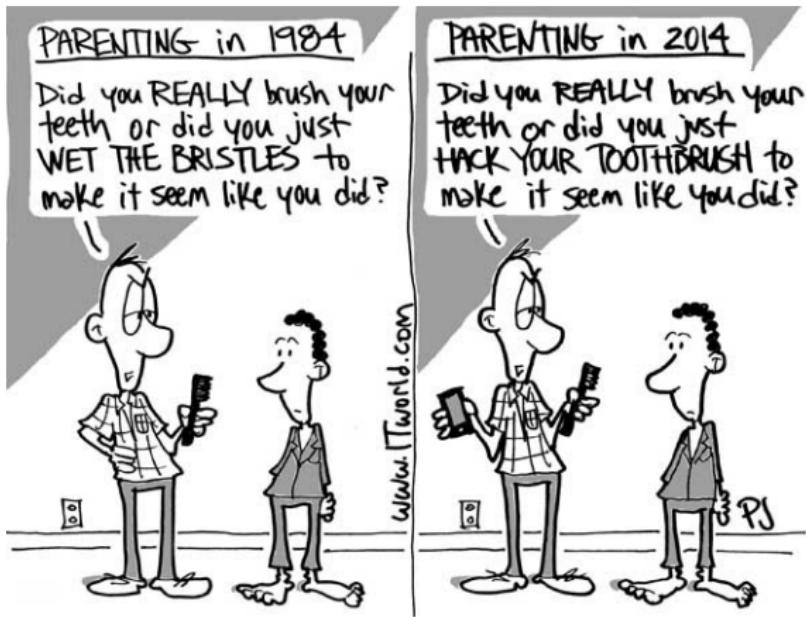
fritzing

Hello LED



fritzing

IoT Fun



REVIEW: For Loops

```
1 // FOR loop syntax
2 for (initialization; condition; increment) {
3     // statement(s);
4 }
5
6 // EXAMPLE
7 for (j=0; j <= 255, j++) {
8     analogWrite(ledPin, j);
9 }
```

While loops

```
1 // WHILE loop syntax
2 while (condition) {
3     // statement(s)
4 }
5
6
7 // EXAMPLE
8 while (button == HIGH) {
9     digitalWrite(ledPin, HIGH);
10 } //continue this loop until button is released
```

For vs While Loops

For VS While Loop

Comparison Chart

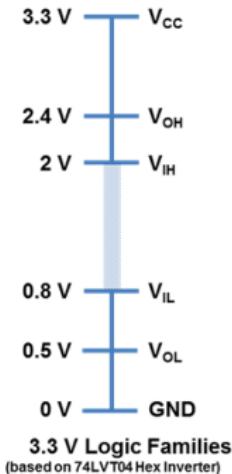
For Loop	While Loop
The for loop is used for definite loops when the number of iterations is known.	The while loop is used when the number of iterations is not known.
For loops can have their counter variables declared in the declaration itself.	There is no built-in loop control variable with a while loop.
This is preferable when we know exactly how many times the loop will be repeated.	The while loop will continue to run infinite number of times until the condition is met.
The loop iterates infinite number of times if the condition is not specified.	If the condition is not specified, it shows a compilation error.

Serial Monitor

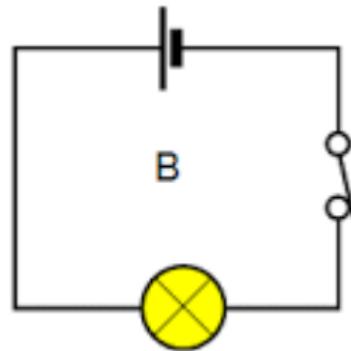
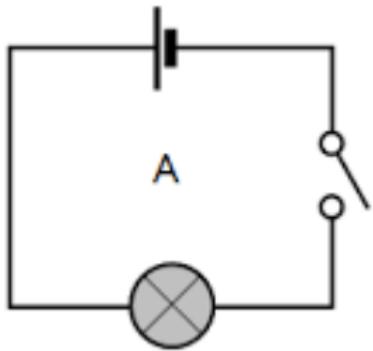
```
1 void setup() {  
2  
3 // Enable Serial Monitor  
4 Serial.begin (9600);  
5 while (!Serial); // wait for Serial monitor  
6 Serial.println ("Ready to Go");  
7 }  
8  
9 void loop() {  
10 for (i=0; i <=13; i++)  
11 Serial.print(i);  
12 delay(ledDelay);  
13 }
```

Digital Output

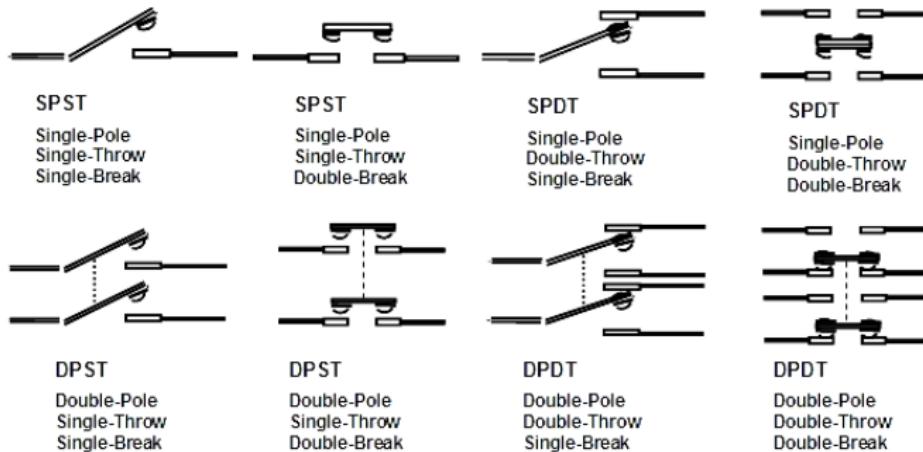
Digital electronics rely on binary logic to store, process, and transmit data or information. Binary Logic refers to one of two states – ON or OFF. This is commonly translated as a binary 1 or binary 0. A binary 1 is also referred to as a HIGH signal and a binary 0 is referred to as a LOW signal.



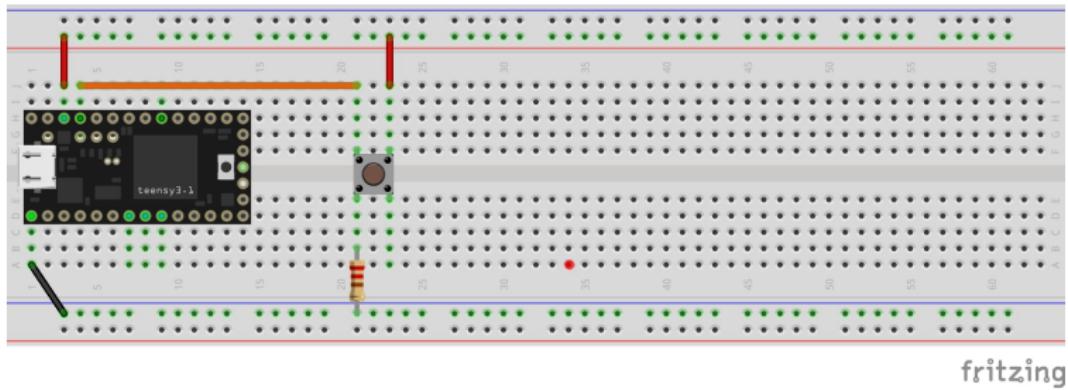
Switches



Types of Switches



Our First Button

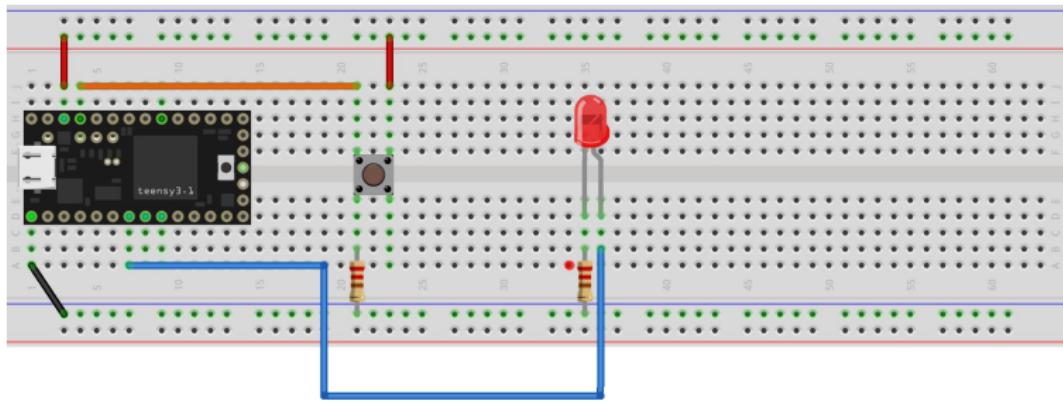


fritzing

If Statements

```
1 // IF statement SYNTAX
2 if (condition) {
3     //statement(s)
4 }
5
6 // EXAMPLE
7 if (button == HIGH) {
8     digitalWrite(ledPin, HIGH);
9 }
```

Button and LED



fritzing

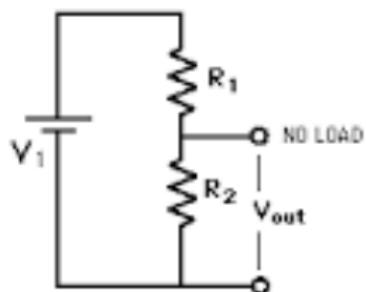
IoT Humor



"I remember when you could only lose a chess game to a supercomputer."

Voltage Divider

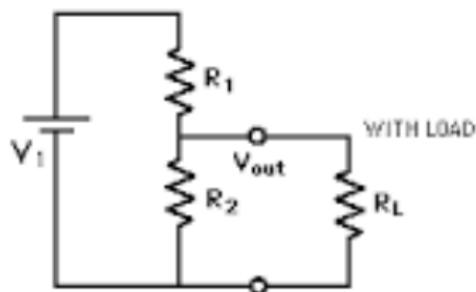
OPEN CIRCUIT BEHAVIOR



$$V_{out} = V_1 \frac{IR_2}{I(R_1 + R_2)} = \frac{V_1 R_2}{(R_1 + R_2)}$$

for open circuit

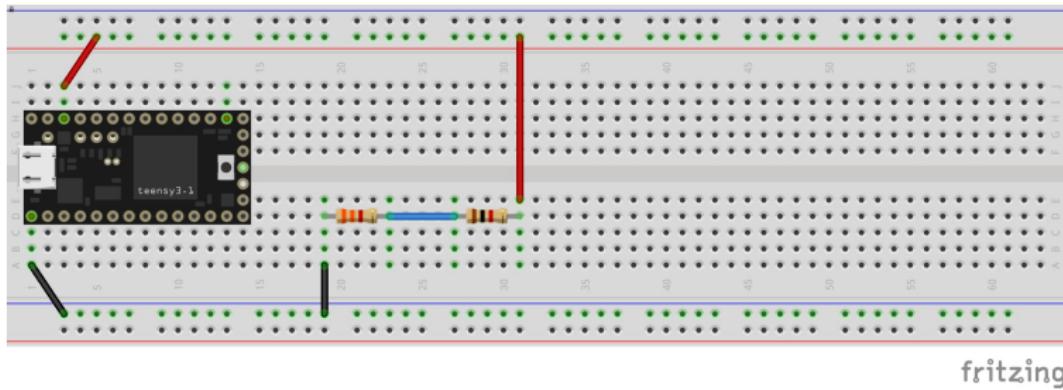
BEHAVIOR UNDER LOAD



$$V_{out} = \frac{V_1 (R_2 || R_L)}{(R_1 + R_2 || R_L)}$$

for loaded circuit

Voltage Dividing

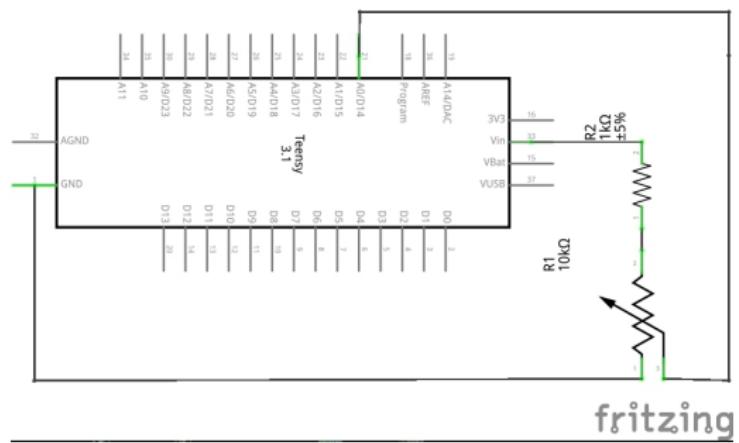


fritzing

We are just using the Teensy to provide Power and GND.
The right resistor should be $1\text{k}\Omega$
The left resistor between $2.2\text{k}\Omega$ and $6.8\text{k}\Omega$

Analog Input Schematic

Teensy 3.1



Feel free to use any PIN you would like to measure the voltage.

Anatomy of a Function

Anatomy of a C function

Datatype of data returned,
any C datatype.

"void" if nothing is returned.

Parameters passed to
function, any C datatype.

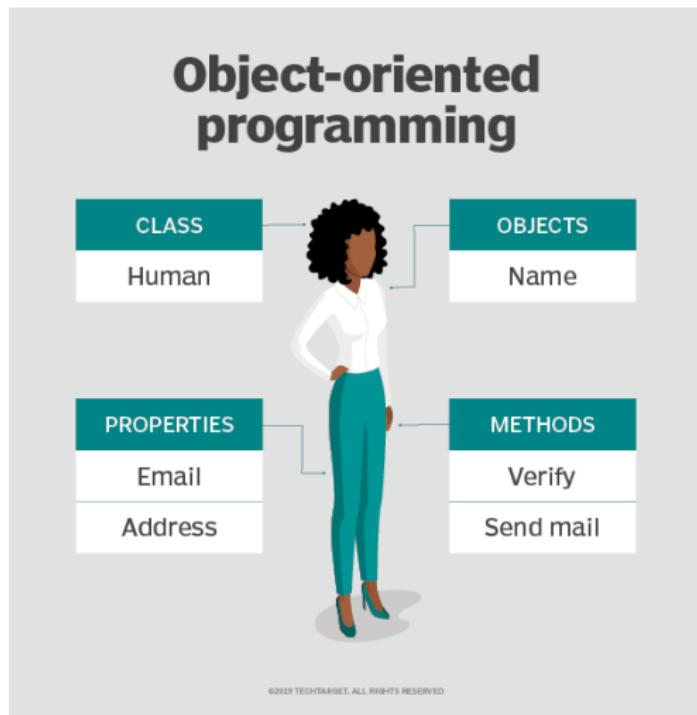
```
int myMultiplyFunction(int x, int y){  
    int result;  
    result = x * y;  
    return result;  
}
```

Function name

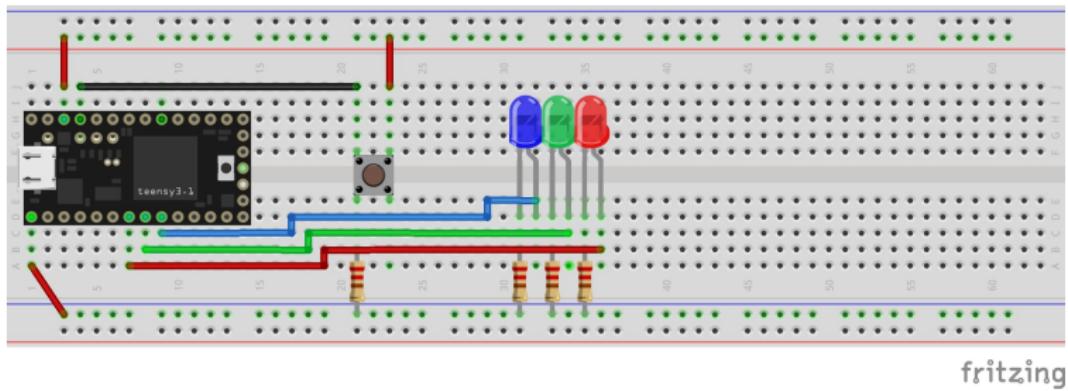
Return statement,
datatype matches
declaration.

Curly braces required.

Objects

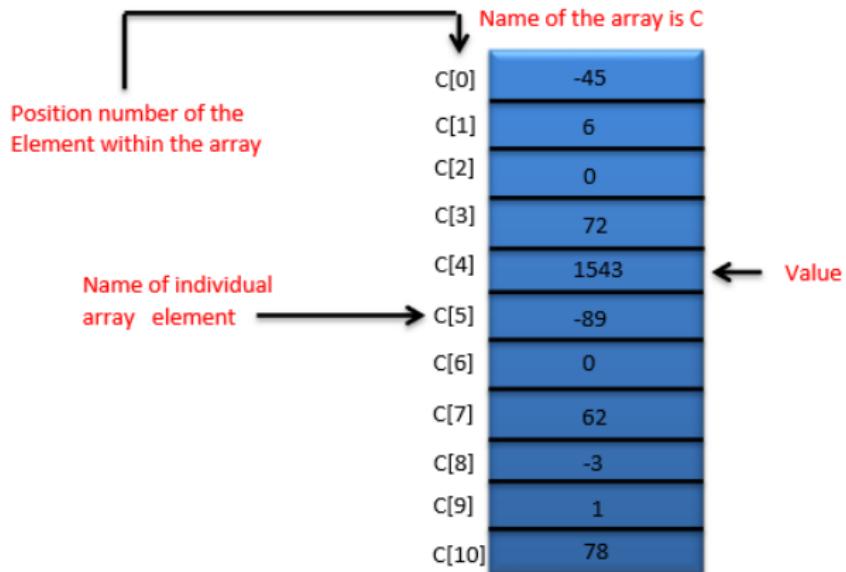


oneButton LED



fritzing

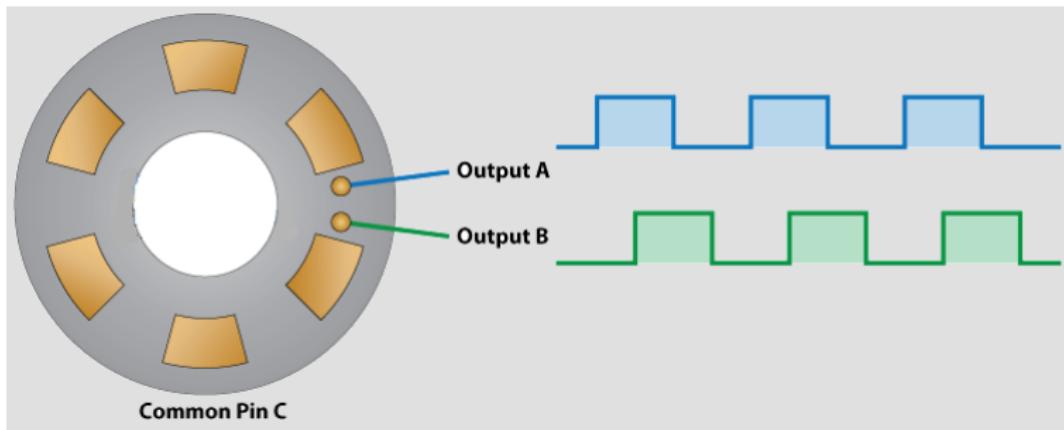
Arrays



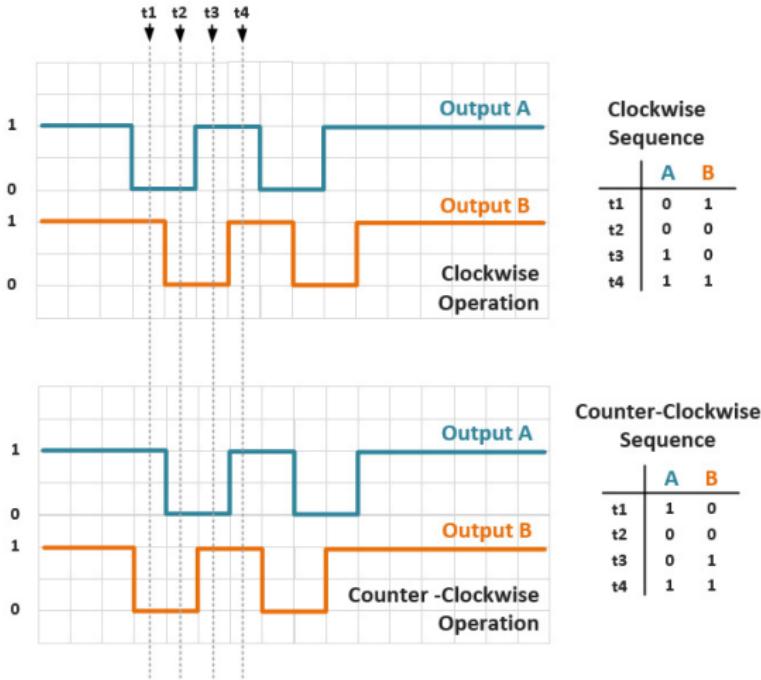
IoT Humor



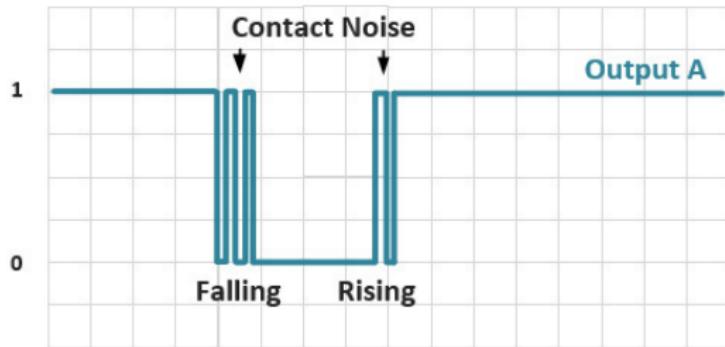
Encoders



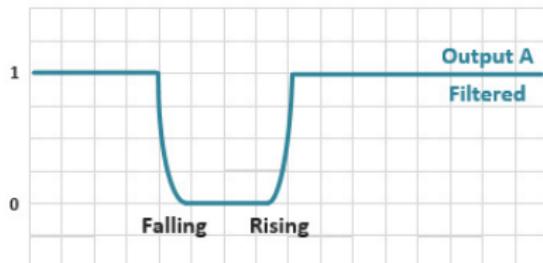
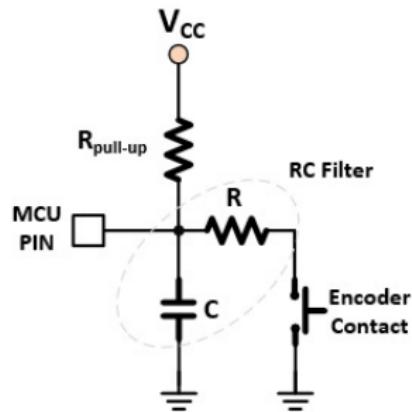
Encoders



Encoders

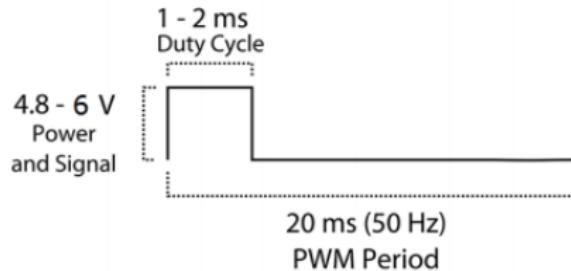
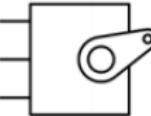


Encoders

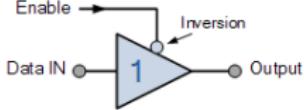


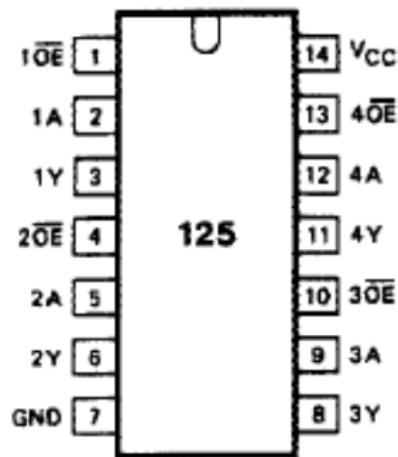
Servo Motors

PWM=Orange (⊤⊤)
Vcc=Red (+)
Ground=Brown (-)

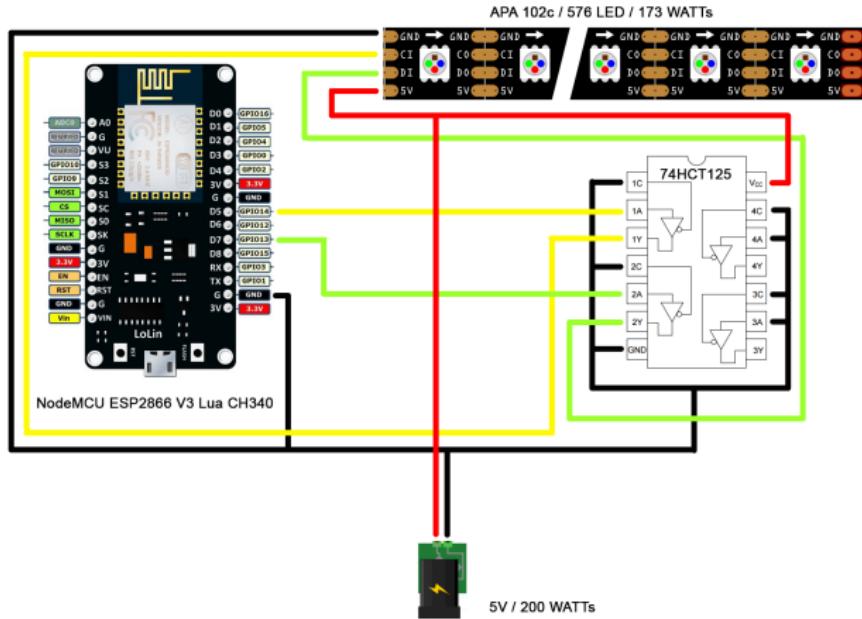


TriState Buffer

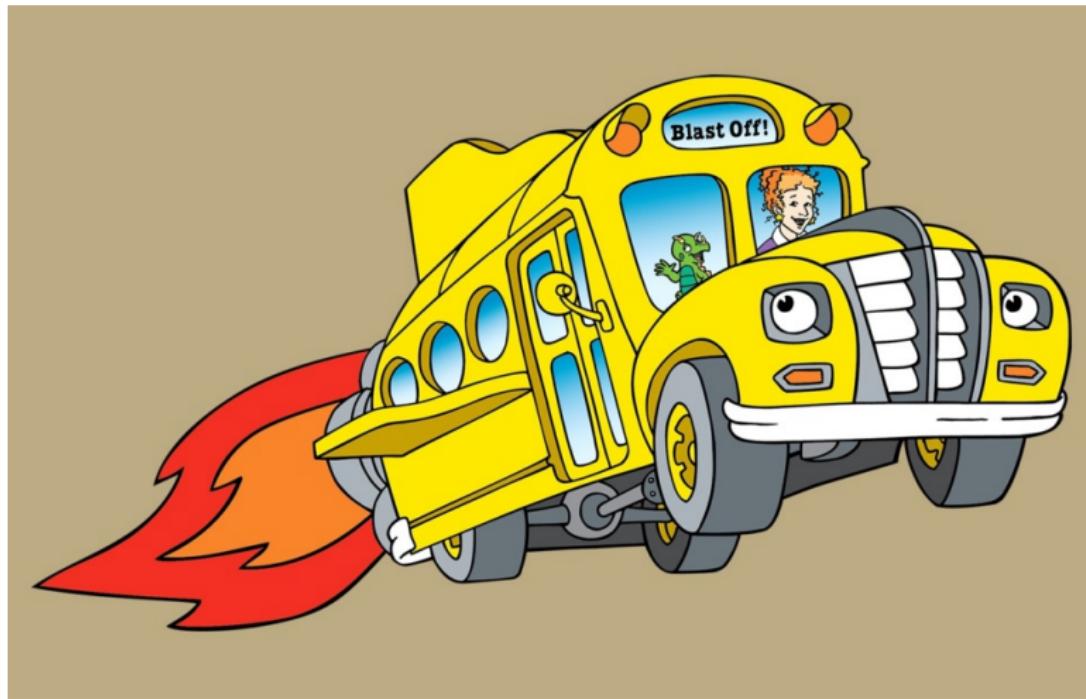
Symbol	Truth Table		
	Enable	IN	OUT
	0	0	0
	0	1	1
	1	0	Hi-Z
	1	1	Hi-Z
Read as Output = Input if Enable is NOT equal to "1"			



TriState Buffer



Buses and Interfaces

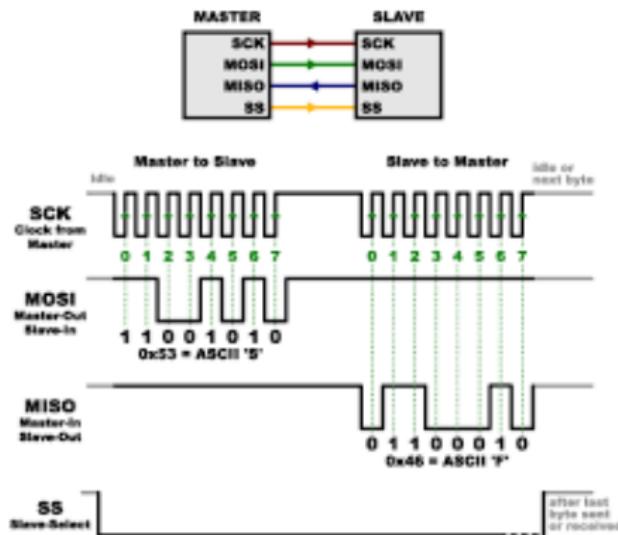


UART

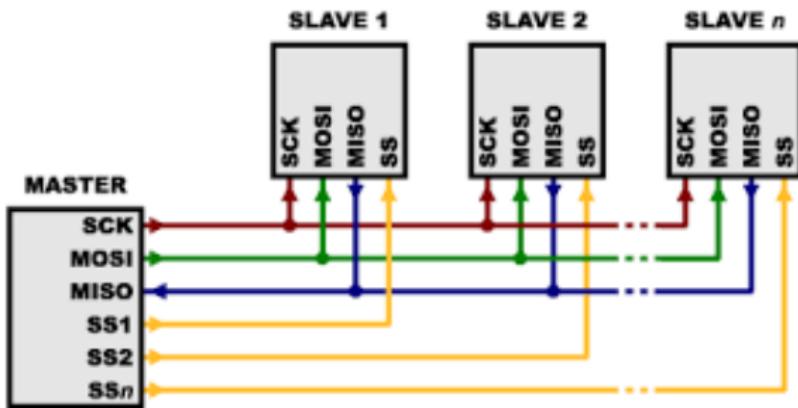


Universal Asynchronous Receiver/Transmitter

Serial Peripheral Interface

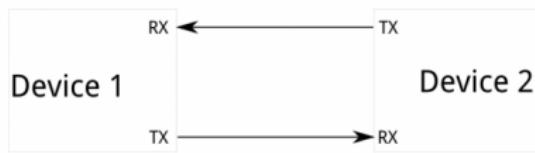


Serial Peripheral Interface

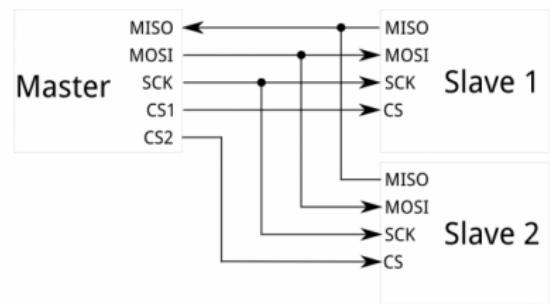


Serial +/−

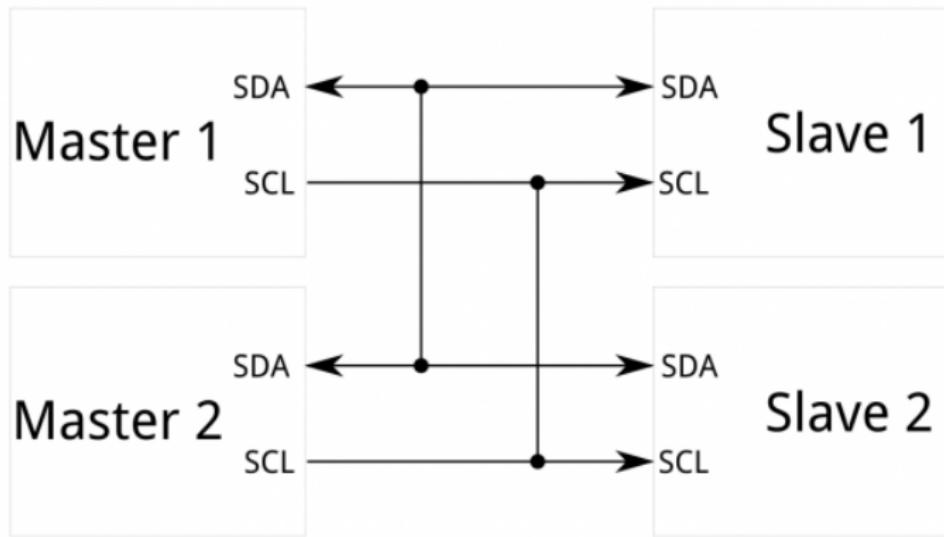
UART



SPI



Inter-integrated Circuit (I^2C)



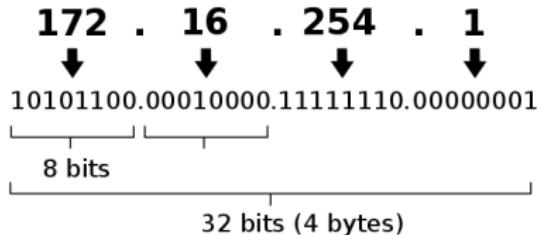
The Internet



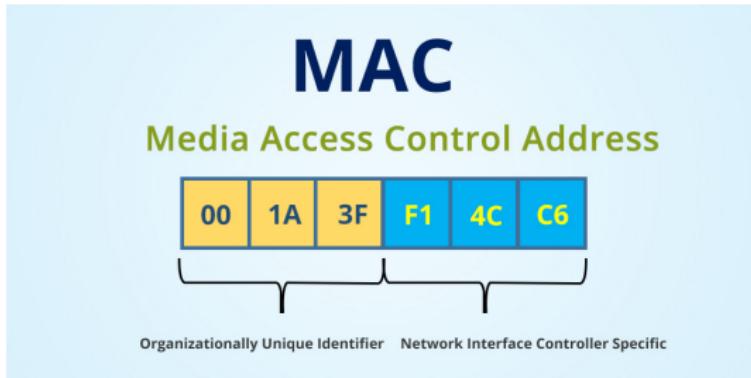
IP Addresses

- When a device joins the network it is given an internet address.
 - static or dynamic
- IPv4 (32-bit) - 4.2 billion
- IPv6 (128-bit) - 340 quadrilliard
- In Powershell, try:
`ipconfig/all`
- In Terminal (MAC), try:
`ipconfig getifaddr en0`

IPv4 address in dotted-decimal notation

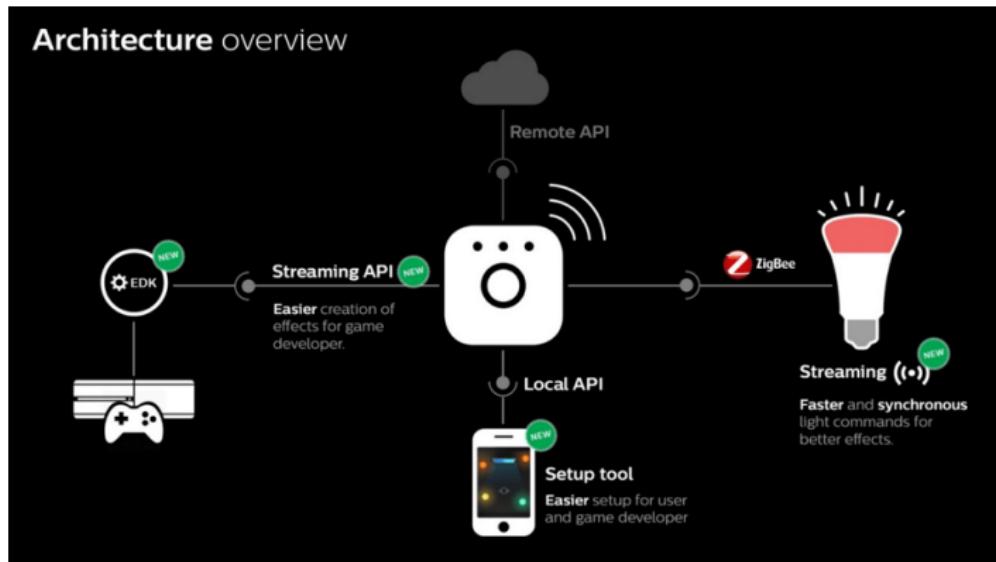


MAC Address



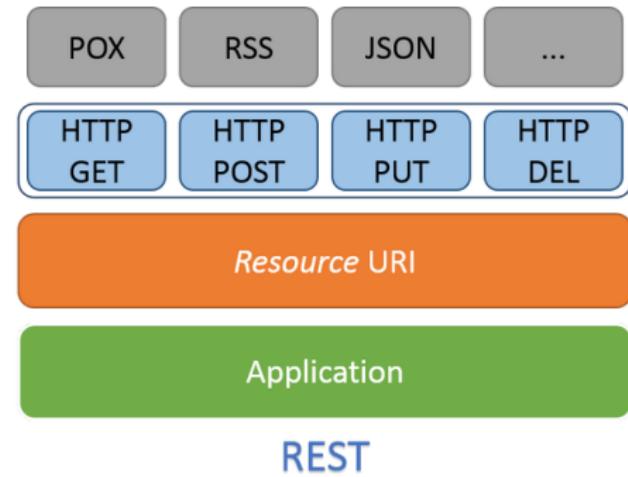
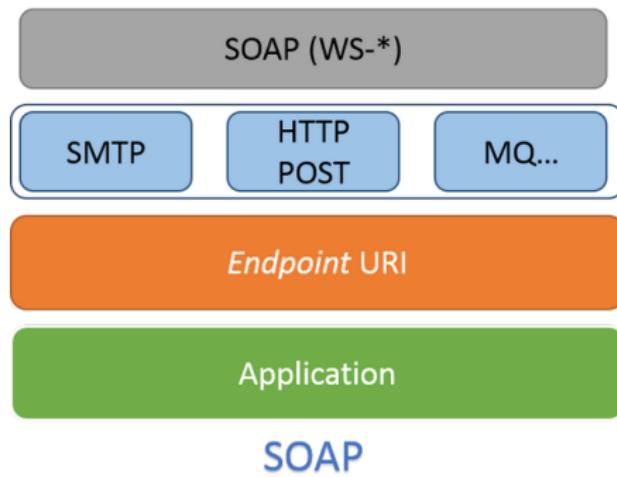
MAC Address is a unique 6-byte (48-bit) address that is usually permanently burned into a network interface card (NIC) or other physical-layer networking device and that uniquely identifies the device on an Ethernet-based network. The uniqueness of MAC addresses is ensured by the Institute of Electrical and Electronics Engineers (IEEE),

Phillips Hue API



Application Programming Interface: a set of functions and procedures allowing the creation of applications that access the features or data of an operating system, application, or other service.

SOAP vs REST



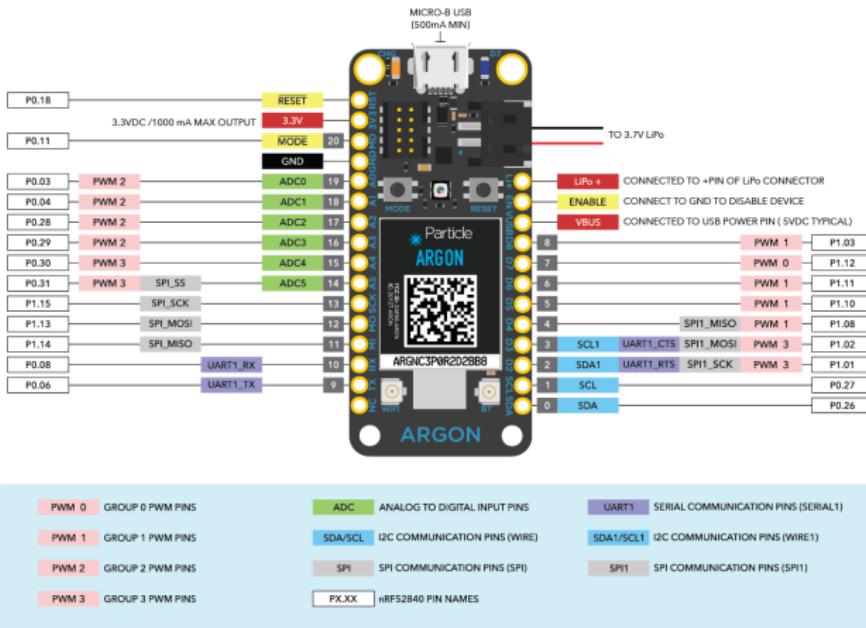
GITHUB Simplified

```
1 // In Powershell go to ./Documents/<yourname>
2 // Get a repository that already exists and pull
   it into your local machine
3 git clone <URL of repository>
4
5 // From the repository directory, get updates
6 git pull
7
8 // Send your changes up to the repository
9 git add .
10 git commit -m "<comment>"
11 git push
12
13 // You may get asked to enter your GIT username
14 git config --global user.email "you@example.com"
```

Our Second Microcontroller



Particle Argon Pin Layout



v1.0