SEICHE 2022

Introduction to Arduino

Lesson 7/8– Grounding, ESD, Breadboards, Schematics, Resistors, Connecting LEDs

Instructor: Paul Frommeyer

www.paulfrommeyer.com

Corporate Sponsor: DXC Technology

REMEMBER!

Immediately copy the latest lesson folder (Lesson6 today) from your USB flash drive to your hard drive where you previously copied the other flash drive files (probably Documents folder)

Lesson Plan Overview

Lesson 1 – Intro and Setup
[may require 2 classes]
- Introduction to class format
- Overview of lesson plan
- Presentation format (monitor, camera, screen, whiteboard)
- Inventory of Arduino kits
- Inventory of USB drives
- Installation of ESP32 serial port drivers (Windows only)
- Installation of Arduino software
- History of Computing

Lesson 2 and 3 – Laptop operation review
- Control panel/Settings location
- Home directories and folder hierarchy
- Arduino file locations
- Search functions
- Open questions and issues
- History of Computing redux

Lesson 4 and 5 – IDE essentials
- The boards manager
- Setting board type
- Installing Libraries
- Loading example sketches
- Finding and selecting the Arduino USB port
- Uploading a sketch to the microcontroller [Example Sketch#1]
- Basic Arduino sketch (program) structure
- Uploading your own sketch [Class Sketch #2]
- History of Telecommunications and Networking – Part 1

Lesson 6 – DC Electricity Basics and DMM Intro
- Quick review: DC vs AC electric currents
- DC current operation
- Voltage vs Amperage (current)
- Ohm's Law
- Series vs parallel circuits
- Introduction to basic DMM functions and usage
- Measuring voltage [DMM usage #1]
- History of Telecommunications and Networking – Part 2

Lesson 7 and 8 – Grounding, ESD, and Connecting LEDs
- Grounding
- The Phantom Menace of ESD
- Resistors!
- LED's [DMM usage #2, maybe]
- Powering LEDs – Ballast and dropping resistors
- Connecting LEDs to microcontrollers
- Parts of an Arduino Sketch
- ESP32 Specifications – Ampacity
- Let there be Light! – Class Sketch #2
- History of Telecommunications and Networking – Part 3

Lesson 9 – ESP32 External power sources [may require 2 classes]
- Capacitors!
- Microcontroller board power requirements
- Board power input options and grounding
- Components with different voltage requirements – level shifting
- External power circuit design
- Connecting and using transistors
- Pulse width modulation and signal generation
- Connecting and using buzzers – Class Sketch #3

Lesson 10 – Connecting smart LED strips
- Simplified serial communication protocols
- One-wire serial communication protocol
- Hardware vs software serial
- WS2812B and Neopixel connections – Class Sketch #4
- External device power design

Lesson 12 – Connecting LCD and Matrix Displays
- The SPI serial protocol
- SPI connections – Class Sketch #5
- SPI device external power design

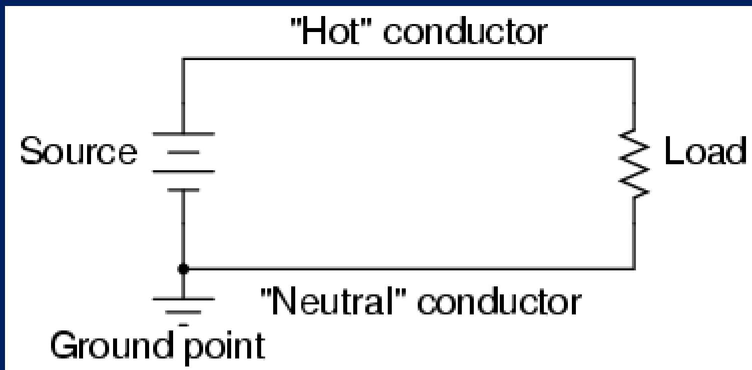
Lesson 13 – Connecting sensors [Time permitting]
- Connecting buttons
- Connecting potentiometers and voltage dividers
- Reading buttons and debouncing
- Reading potentiometers

Lesson 14 – Connecting more sensors [Time permitting]
- Connecting I2C sensors and photoresistors
- Using ESP32 integral touch sensors

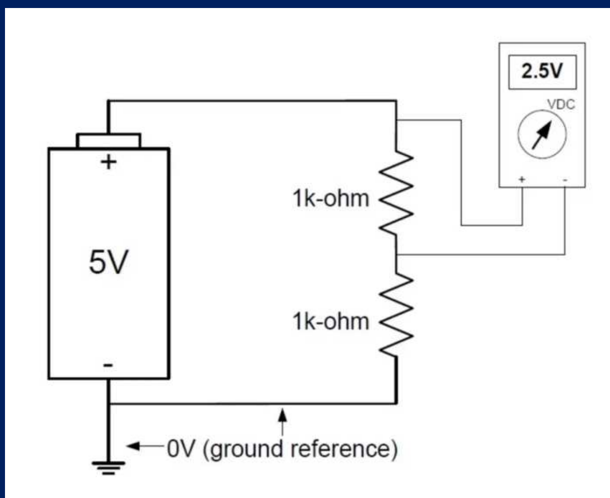
Lesson 7 and 8 – Connections

- Grounding
- ESD Safety
- Breadboard Wiring
- Schematic Diagrams
- Resistors
- Measuring Resistance
- LEDs
- Powering LEDs – Ballast and Dropping Resistors
- Connecting LEDs to Microcontrollers
- Parts of an Arduino Sketch
- WiFi Kit 32 Specifications Review - Ampacity
- Let There Be Light – Class Sketch #2
- History of Telecommunications and Networking P2 (Time Permitting)

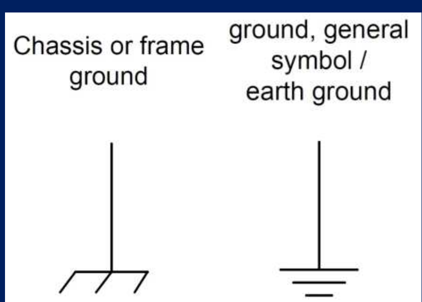
Grounding – What is a Ground?



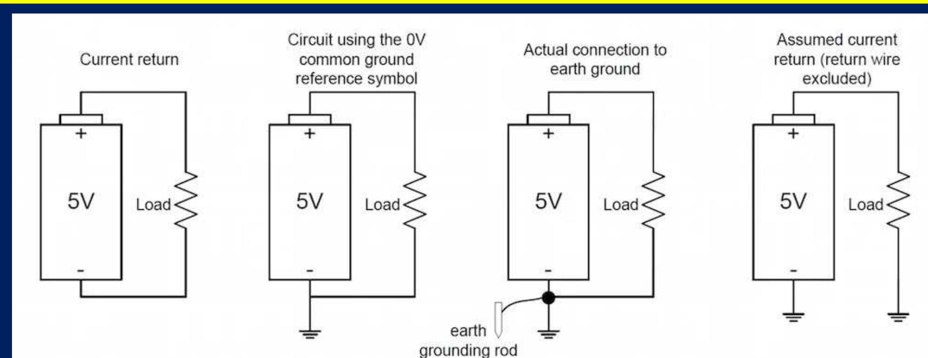
In electronics and electrical engineering, it is by convention we define a point in a circuit as a reference point. This reference point is known as ground (or GND) and carries a voltage of 0V. Voltage measurements are relative measurements. That is, a voltage measurement must be compared to another point in the circuit. If it is not, the measurement is meaningless. The ground reference point is often, but not always—more on this later—represented by a standard (earth) ground symbol.



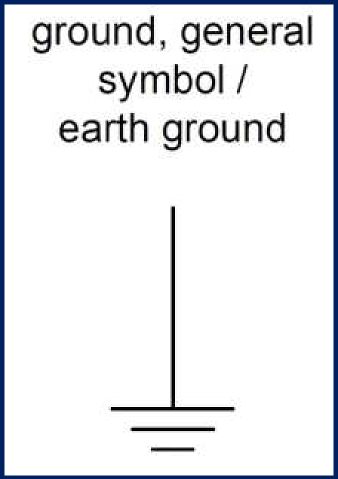
In DC circuits, the ground path is the return path to the power source

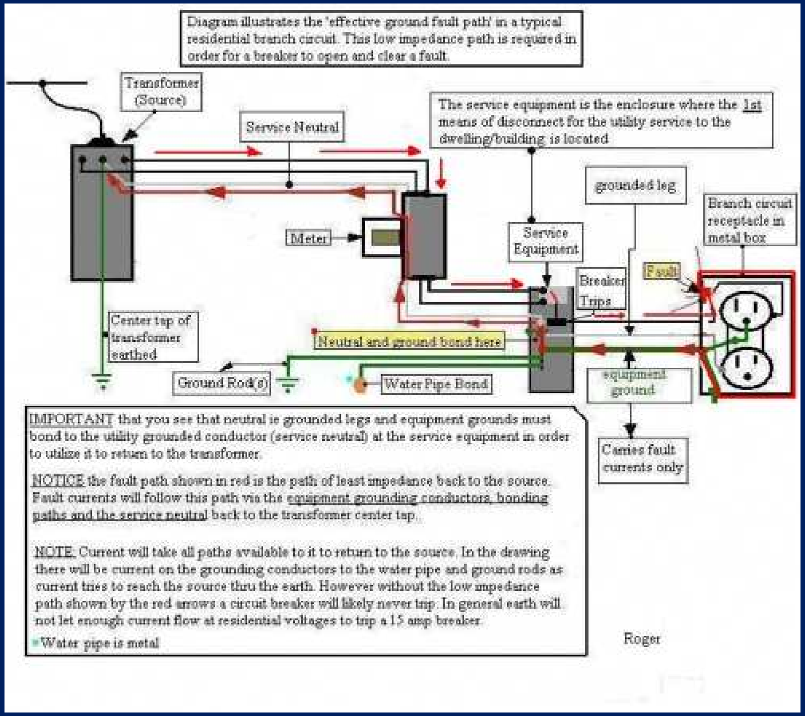


https://www.allaboutcircuits.com/technical-articles/an-introduction-to-ground/



Grounding – Earth Ground





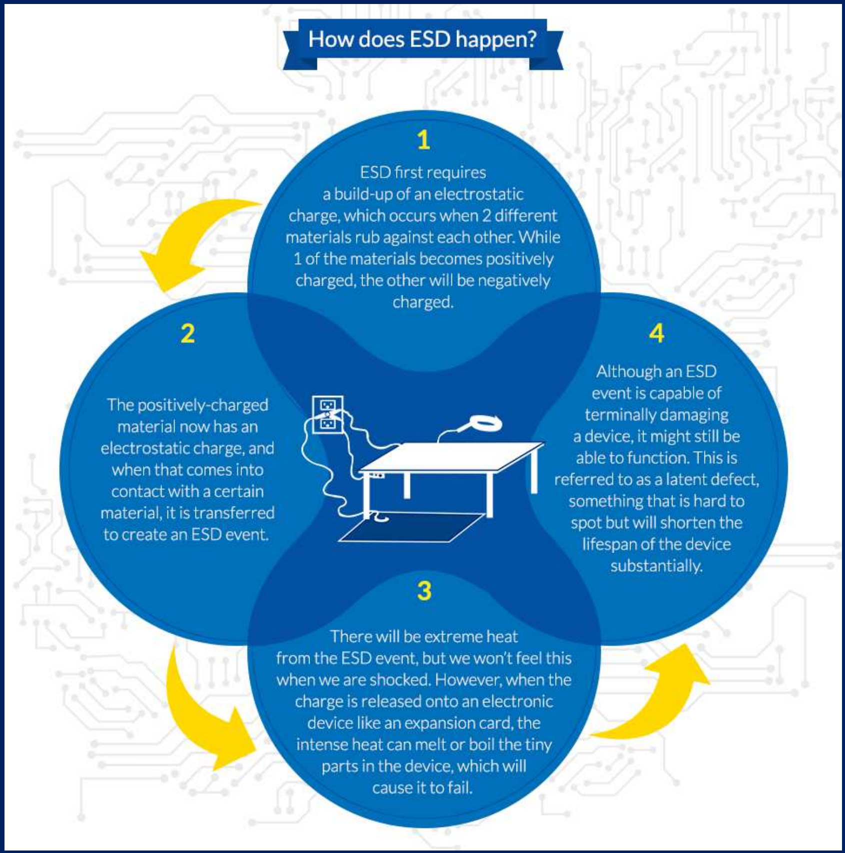
https://www.allaboutcircuits.com/technical-articles/an-introduction-to-ground/

Earth ground is exactly as it sounds. It's a ground physically (and electrically) connected to earth via a conductive material such as copper, aluminum, or an aluminum alloy.

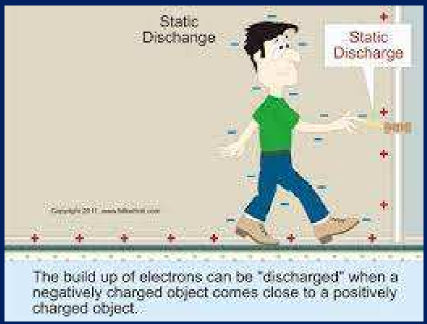
A true earth ground, as defined by the National Electrical Code (NEC), consists of a conductive pipe, or rod, physically driven into the earth to a minimum depth of 8 feet.

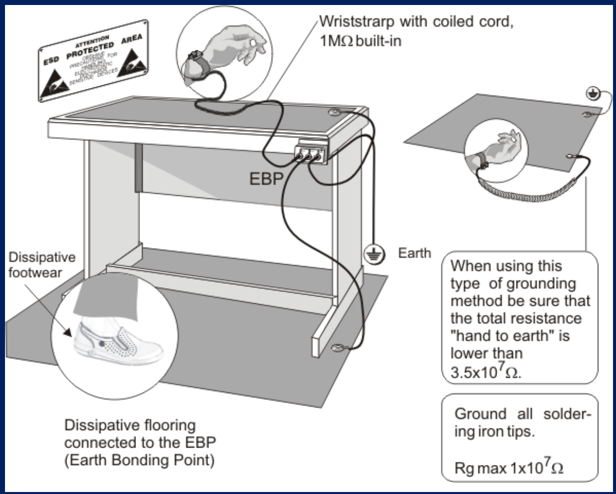
The earth provides an electrically neutral body, and due to the earth's virtually infinite state of neutrality, it is immune to electrical wavering. It should be noted, however, that "earth being immune to electrical wavering" is, in fact, a generalization. In reality, earth ground is quite the complex subject given all variables and materials that make up the earth. And, earth's electrical potential does indeed experience some isolated areas of varying due to events such as lightning hits, as an example. Power poles, those that are strung throughout neighborhoods, are also connected to ground.

Electrostatic Discharge – The Phantom Menace



Integrated circuits (ICs) are notorious for being ultra vulnerable to ESD events. Grounded mats (referred to as ESD mats), grounded chairs, and wrist straps provide adequate ESD protection for your ICs by grounding you—thus discharging any static you may have on your body—prior to touching any sensitive components. Most engineers and technicians also wear ESD-safe jackets when working with PCBs and ICs for added protection against possibly damaging components and equipment.



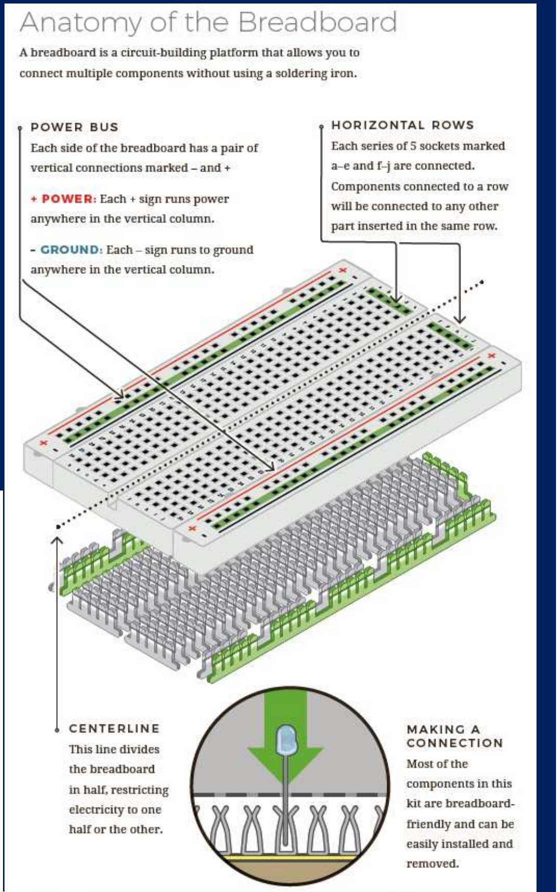


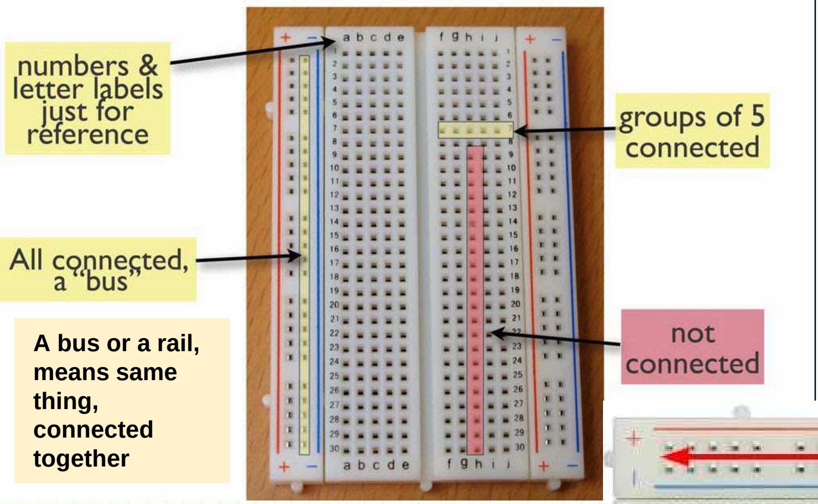
WiFi chips on dev boards are prone to static damage!!!

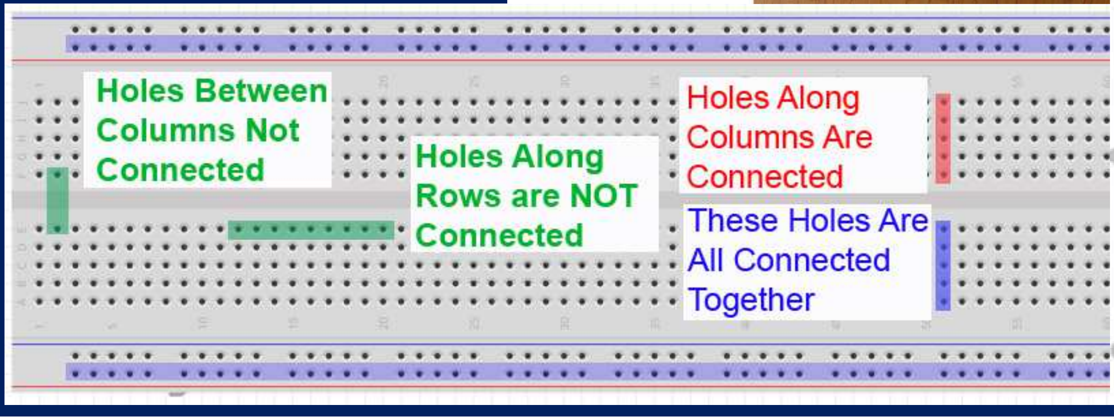
Breadboard Wiring

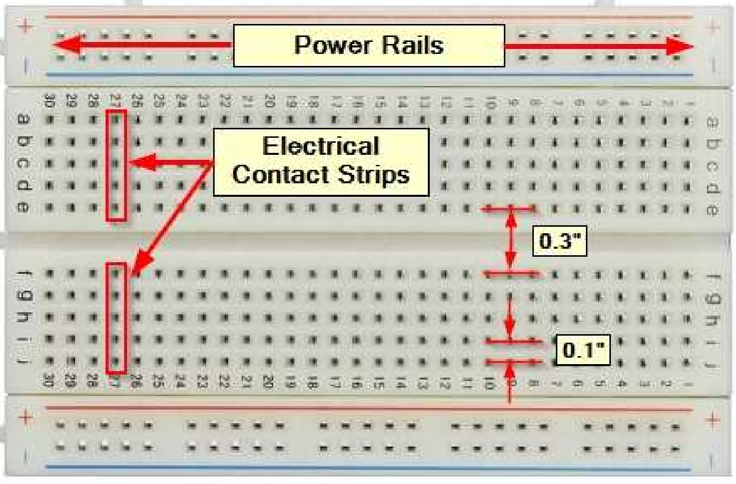
- A breadboard typically consists of four outside rails. These rails are actually continuous metal strips, connecting all “points” along that rail together to form a continuous “rail” [common electrical connection.]
- Breadboards also have a number of smaller inside groups of connection points. These are “mini-rails” which are oriented perpendicular to the outside rails for connecting components.
- There is a center gap spaced to allow the pins of a Dual Inline Plug (DIP) semiconductor chip or microcontroller to sit across the gap.
- Once installed, the pins of the chip or board are electrically connected to the other pins in the same row.
- Jumper wires can then be connected to and between these pins to wire up the chip or controller.

- **Vertical pins are connected together**
- **Horizontal pins are connected together**
- **Vertical and horizontal not connected**
- **No connections across gap**



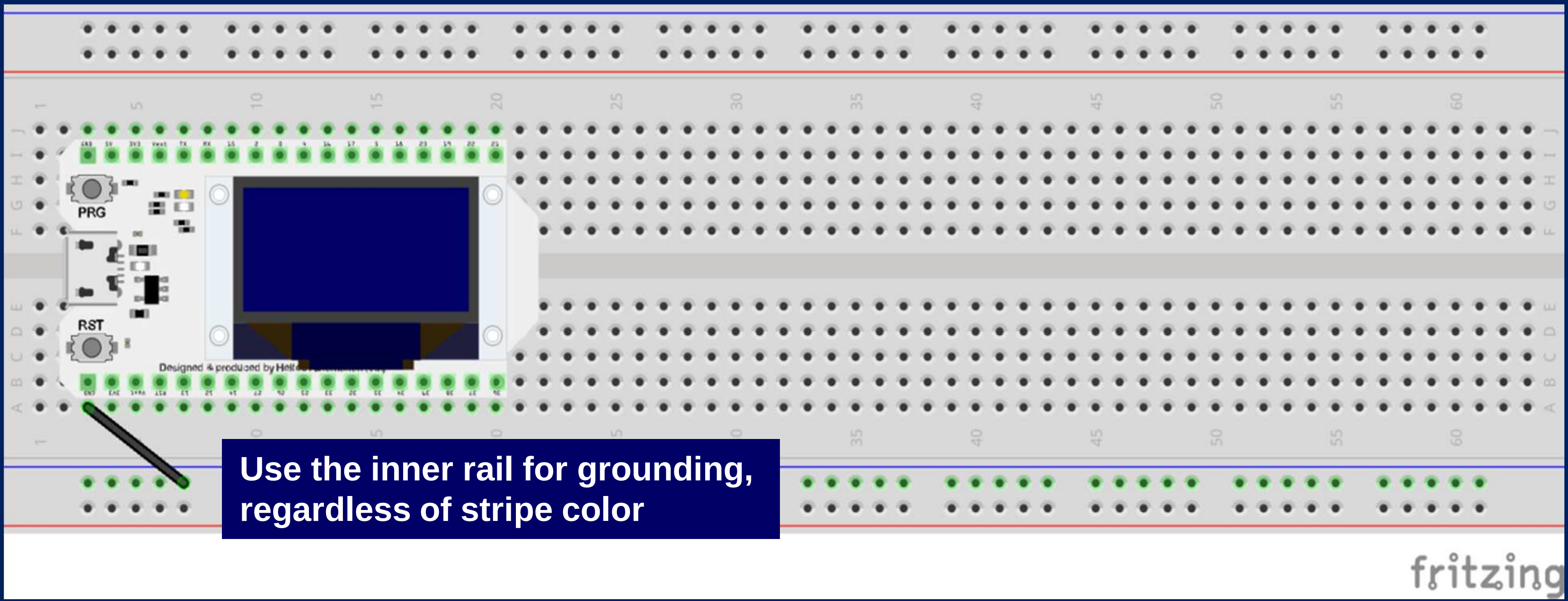






Electrostatic Discharge – MC Protection

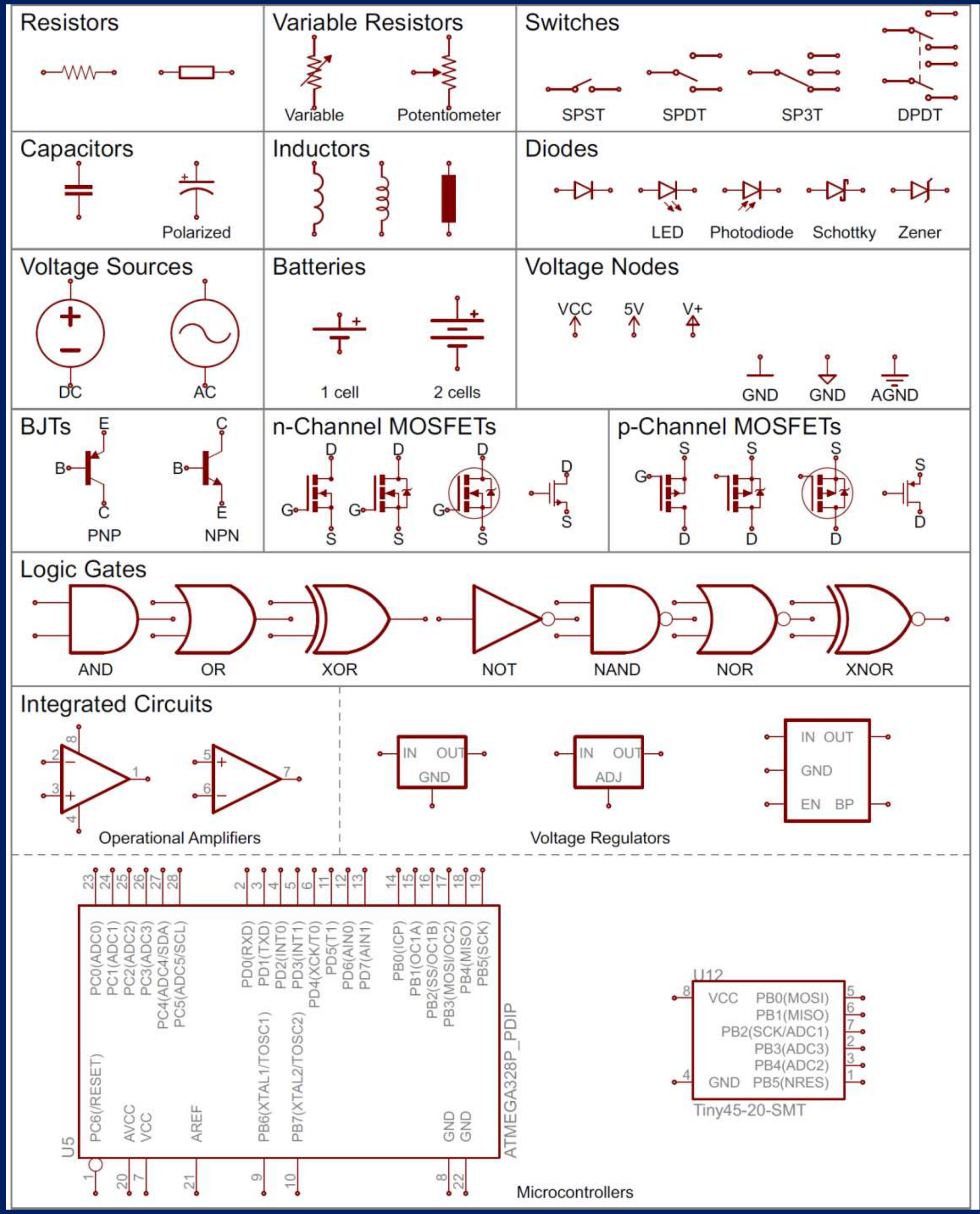
Large charges need large grounds, but small charges only need small grounds. Metal the mass of a quarter is sufficient protection for most sensitive electronics, though bigger is better and earth ground is always best. The easiest place to find and attach such metal to a dev board is to just plug it into a breadboard. For particularly sensitive boards (which is us), it may be necessary to "permanently" connect the board ground to one or both of the ground busses on the breadboard. See diagram, above, for how to do this with the WiFi Dev Kit 32



NEVER remove your dev board from the breadboard without full anti-static protection: *wrist strap and anti-static work surface!*

Schematic Diagrams

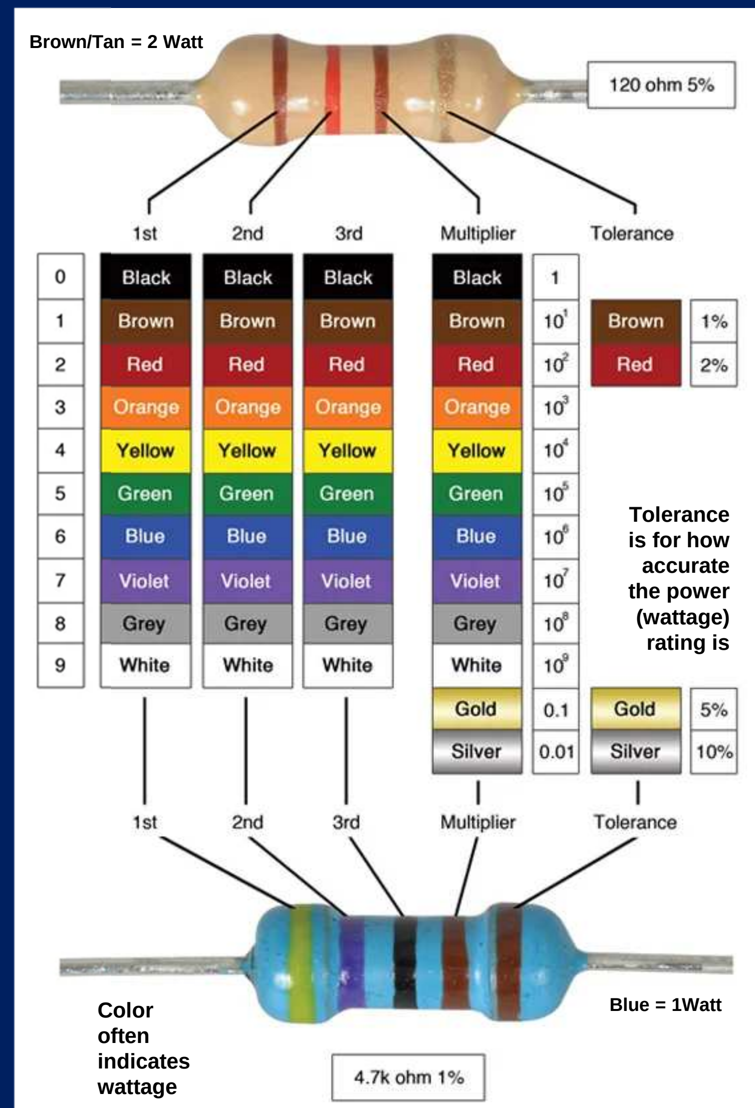
- **All common electronic components have specific symbols**
- **These symbols are used in diagrams to show how the components are connected together.**
- **These diagrams are more compact and show connections more clearly than photos or pictures of the devices themselves**
- **This is particularly true for complex circuits with hundreds of connections**
- **Such diagrams are called “schematic diagrams” because these use a schema (symbols) to represent the components**



Our First Component – Resistors!

- Resistors are elements of electrical circuits that resist and reduce the flow of current through the electrical circuit. They can also be used to provide specific voltages to active components. Resistors are available with fixed or variable resistance values. Fixed resistors are ones in which resistance never changes. [Mouser Electronics]
- A resistor is a passive two-terminal electrical component that implements electrical resistance as a circuit element. In electronic circuits, resistors are used to reduce current flow, adjust signal levels, to divide voltages, bias active elements, and terminate transmission lines, among other uses. [Wikipedia]
- The function of a resistor is measured in Ohms of electrical resistance
- Resistors use a color code to indicate resistance and heat dissipation tolerance (wattage)
- When electricity passes through a resistor, the voltage of the current drops due to loss of energy. This is called voltage drop, and happens with nearly *all* components.

Resistor Symbol



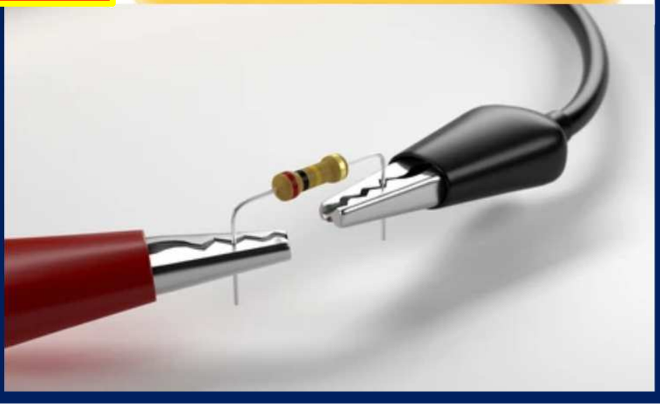
Resistors are critical components, pervasive in electronic devices

Measuring Resistance

- If you are faced with a single resistor or pack that is not labelled, you will need to measure the resistance of one item to determine the value
- Resistance is most easily measured with a DMM– one reason you have them. This is easier and more reliable than trying to decipher color code.
- Some of the class received resistor packs without values written on them; these will need to be measured. Resulting values should be rounded to the nearest 10 (e.g. 327 becomes 330).
- Follow the arrangement shown to measure the resistance of each pack. Immediately label the pack once you have!

Set DMM to resistance measurement

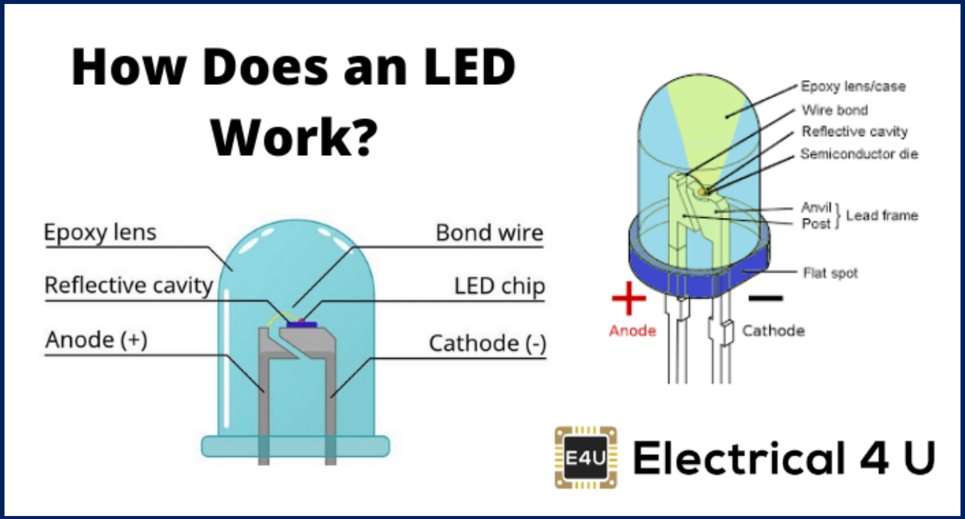
- **Use alligator clip leads**
- **Place *one* resistor between clips**



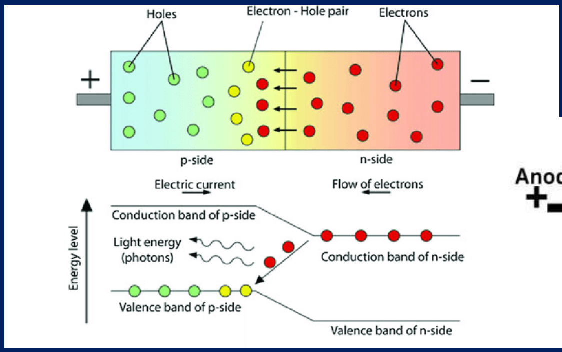
Light Emitting Diodes

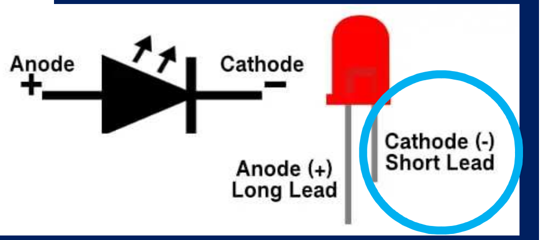
- LED – Light Emitting Diode
- A semiconductor component similar to a non-emitting diode or transistor
- Electrical current through the semiconductor chip produces light
- Semiconductor materials used define the color of light produced
- The “bluer” the light, the more energy (watts) the LED requires (consumes)
- Like all diodes, current can only flow one way through the component, even if its an LED, from + to -

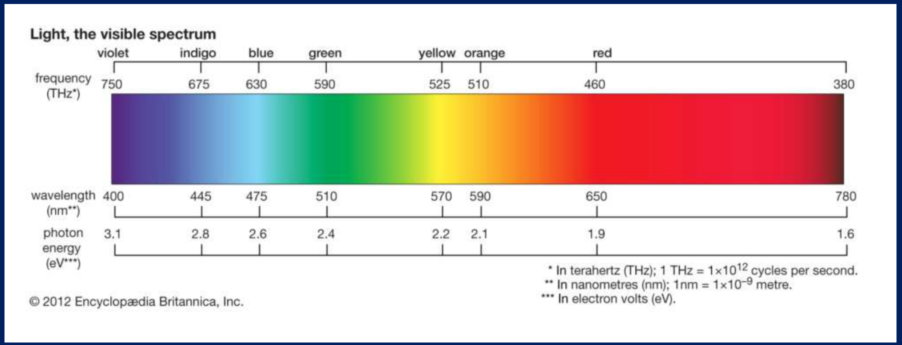
Shorter wavelengths *have* more energy, so *require* more energy to create the photons



Its usually easier to use lead length than the flat spot to identify the cathode

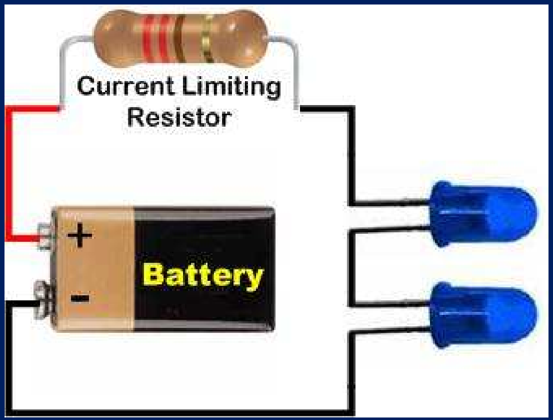


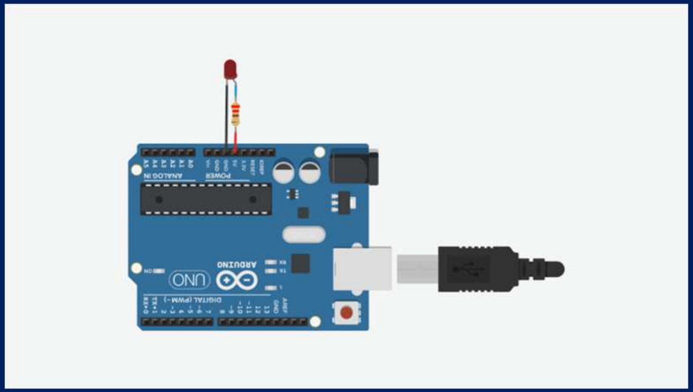




Light Emitting Diodes

- Anodes (+ terminal) are usually connected to the microcontroller
- Cathodes (– terminal) are typically connected to ground (negative return to power source)
- LEDs are constant current devices, drawing between 10 and 20 mA depending on wavelength and emitter size
- To prevent burning out the microcontroller, limiting resistors are connected *in series* with the LED
- These “ballast” resistors can be connected to either the anode or cathode of the LED.
- It is best practice to connect a *single* resistor *between* the microcontroller GPIO pin and the LED *anode*
- There are h4x which ignore this (which we will use!)

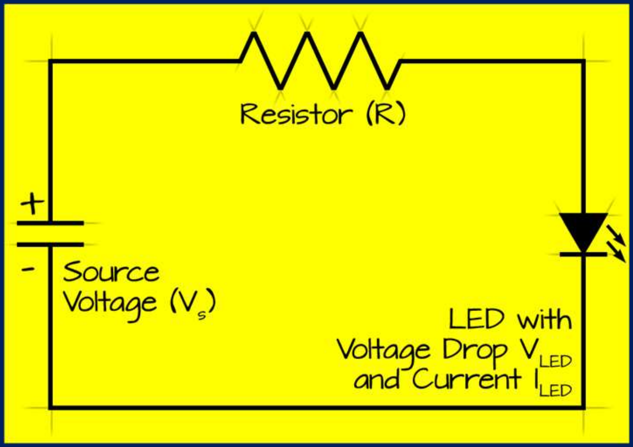


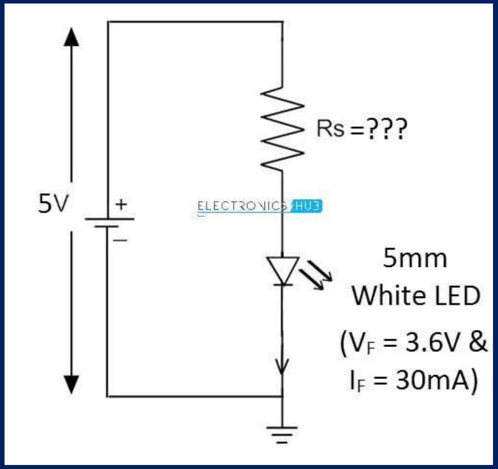


Most microcontrollers have internal components for each GPIO pin called pull-up and pull-down resistors that are software controlled. We will study these as part of connecting buttons.

Ballast and Dropping Resistors

- **A resistor can be used for current limitation. This resistor is called the ballast resistor. If the supplied voltage is *equal* to the LED *voltage drop*, a resistor will not be needed.**
- **If you are *exceeding* the LED voltage drop, you will need a *dropping resistor* to resist the excess voltage. When calculating the resistor value, you will *first* need to know the voltage drop across the LED.**
- **An example would be a 12V power supply for a 2V LED. With the voltage drop of 2V, you can determine the excess voltage is 10V above the LED value. If the current in the circuit is 200mA, we can now use Ohm's Law to find the resistor value. With the additional 10V in the circuit at 200mA, the final result will be a 50 ohm dropping resistor.**
- **Please note that the power rating for the resistor is important. Power can be calculated as V^2/R, I^2xR, or VxR. In this case, the absolute minimum power rating for this resistor can be 2W, but should be doubled for reliability at 4W.**
- **It's OK to use a slightly higher value resistor– in fact this is a good idea. Just be mindful of the resistor wattage (and tolerance.)**





This is one of the most frequent times we use Ohm's law

Parts of an Arduino Sketch

- All sketches have two main parts, or sections, of C++ code
- A setup section that runs once when the microcontroller is *powered up or reset.* This is why a reset (automatically) happens after a sketch is first loaded.
- A loop section which runs *continuously* <u>after</u> the setup section has completed
- Technically speaking, these sections are individual *functions* and are called by a master code module which the Arduino IDE elides (hides) from the user
- Setup is used to initialize variables, set pin operating modes, and other things to "set the stage" for the main program loop
- The loop is where the program steps (statements) for actual processing and events take place
- Certain special code may go outside of the setup or loop

ESP32 Specifications

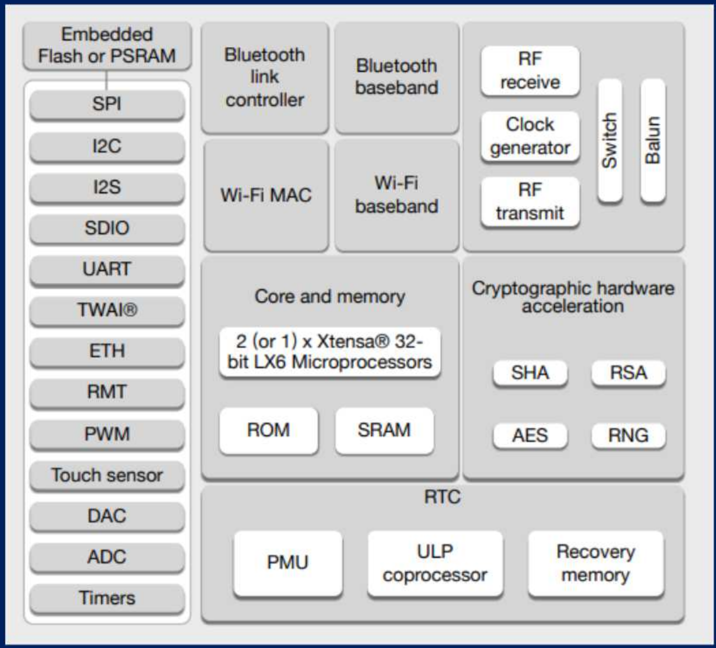
Table 15: DC Characteristics (3.3 V, 25 °C)

Symbol	Parameter		Min	Typ	Max	Unit
C_{IN}	Pin capacitance		-	2	-	pF
V_{IH}	High-level input voltage		0.75×VDD[1]	-	VDD[1]+0.3	V
V_{IL}	Low-level input voltage		–0.3	-	0.25×VDD[1]	V
I_{IH}	High-level input current		-	-	50	nA
I_{IL}	Low-level input current		-	-	50	nA
V_{OH}	High-level output voltage		0.8×VDD[1]	-	-	V
V_{OL}	Low-level output voltage		-	-	0.1×VDD[1]	V
I_{OH}	High-level source current (VDD[1] = 3.3 V, V_{OH} >= 2.64 V, output drive strength set to the maximum)	VDD3P3_CPU power domain [1, 2]	-	40	-	mA
		VDD3P3_RTC power domain [1, 2]	-	40	-	mA
		VDD_SDIO power domain [1, 3]	-	20	-	mA
I_{OL}	Low-level sink current (VDD[1] = 3.3 V, V_{OL} = 0.495 V, output drive strength set to the maximum)		-	28	-	mA
R_{PU}	Resistance of internal pull-up resistor		-	45	-	kΩ
R_{PD}	Resistance of internal pull-down resistor		-	45	-	kΩ
V_{IL_nRST}	Low-level input voltage of CHIP_PU to power off the chip		-	-	0.6	V

Notes:

1. Please see Table IO_MUX for IO's power domain. VDD is the I/O voltage for a particular power domain of pins.
2. For VDD3P3_CPU and VDD3P3_RTC power domain, per-pin current sourced in the same domain is gradually reduced from around 40 mA to around 29 mA, V_{OH}>=2.64 V, as the number of current-source pins increases.
3. For VDD_SDIO power domain, per-pin current sourced in the same domain is gradually reduced from around 30 mA to around 10 mA, V_{OH}>=2.64 V, as the number of current-source pins increases.

DC Characteristics [electrical tolerances]



ESP32 Functional Block Diagram

Takeaways

- **Max pin source current: 40mA**
- **Max pin sink current: 28ma**
- **Pull-up resistance: 45KΩ**
- **Pull-down resistance: 45KΩ**

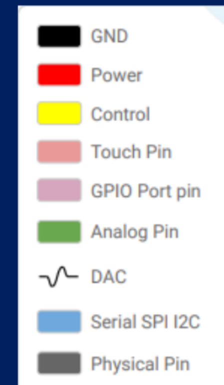
All Data From Espressif Systems ESP32 Datasheet

https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf

Time to Get Wired

What we are going to do, electrically speaking:

- **Connect a red LED *anode* to pin GPIO13 (physical pin 5)**
- **Connect a green LED *anode* to pin GPIO12 (physical pin 6)**
- **Connect a blue LED *anode* to pin GPIO14 (physical pin 7)**
- **Connect *all* LED *cathodes* to a rail**
- **Connect cathode rail to the ground pin of the board using a 300 ohm resistor; this places the resistor in series between all three LED cathodes and the power sink (power ground)**



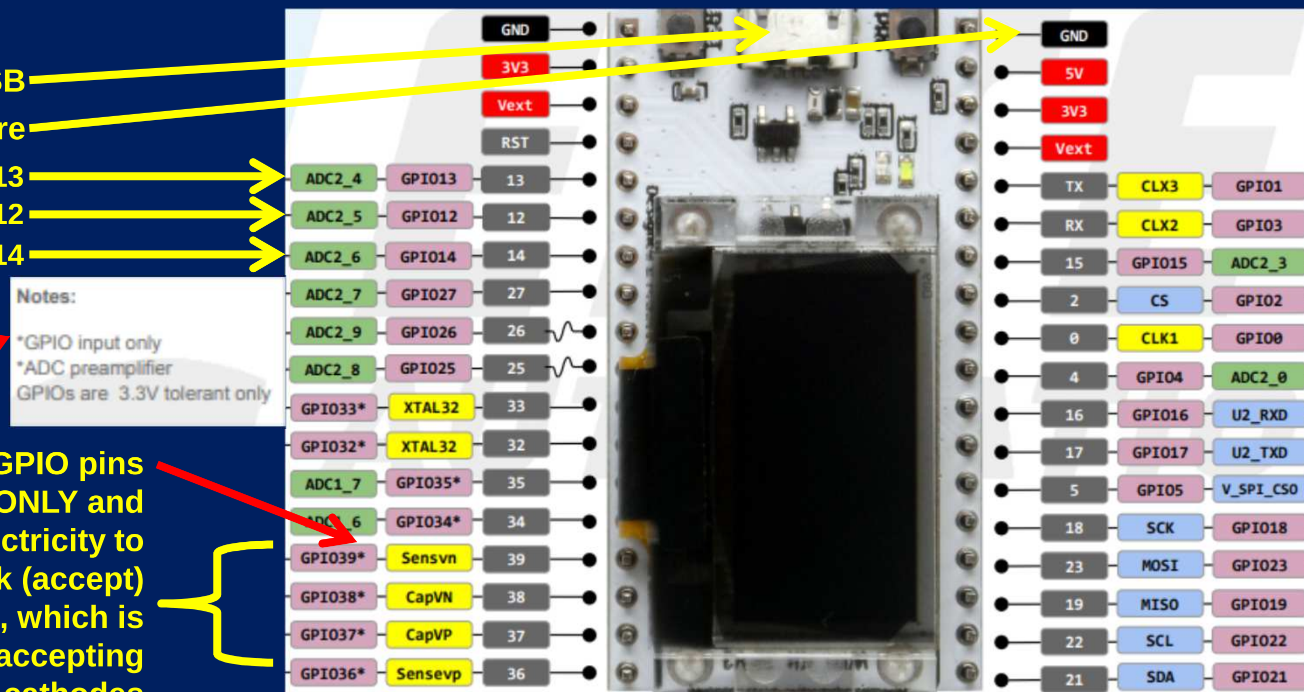
Power will come from USB

300Ω resistor connects here

GPIO13

GPIO12

GPIO14



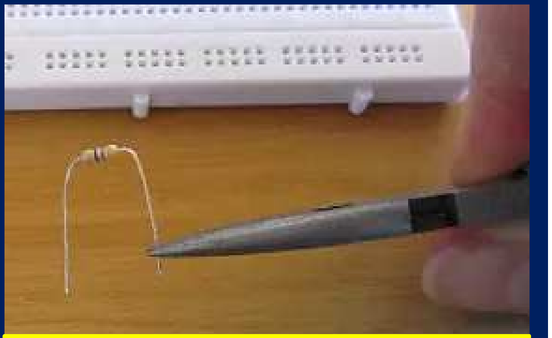
The asterisk on these GPIO pins means they are INPUT ONLY and cannot be used to *send* electricity to anything. They can sink (accept) 28mA of current, which is dangerously small for accepting current from LED cathodes

Connect The Dots

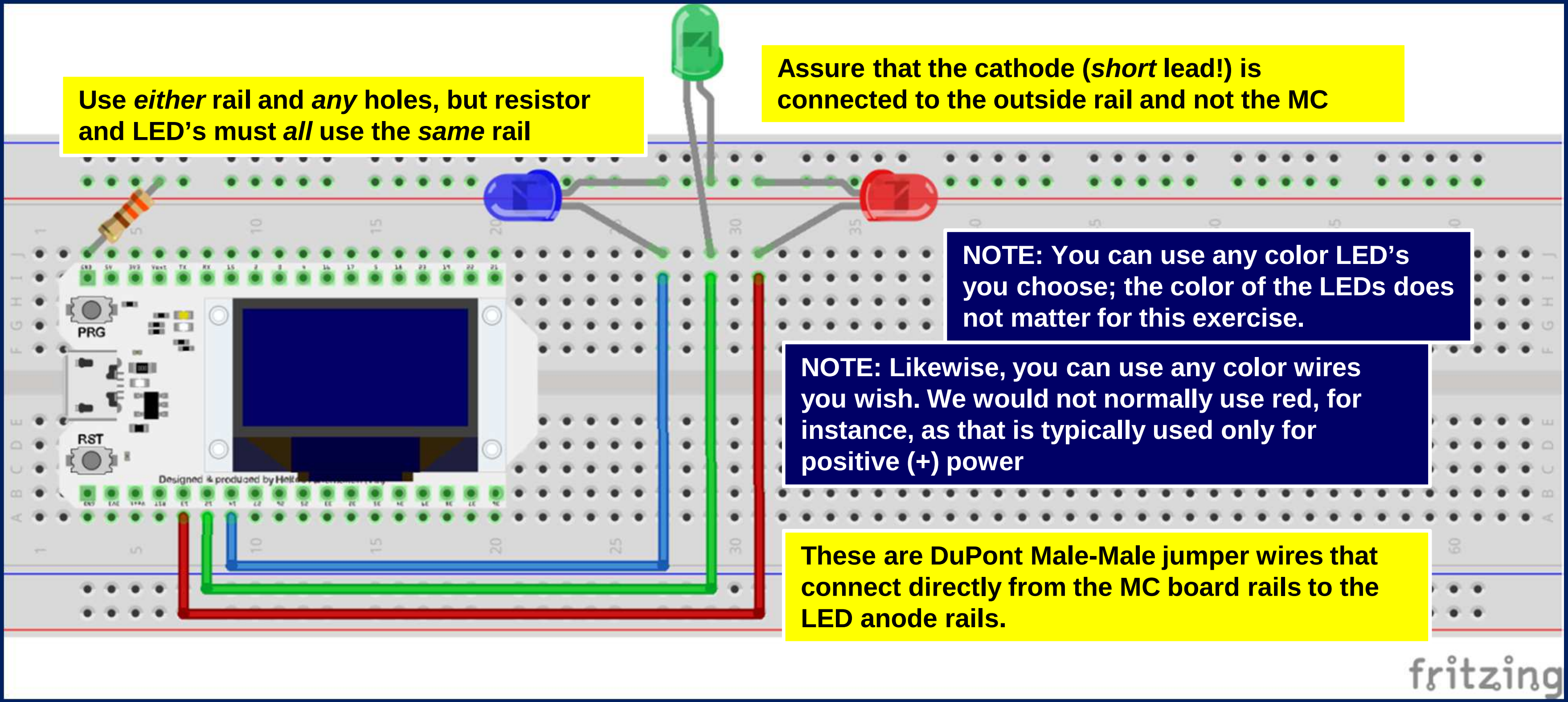
From your Arduino kits:

- Select a red, a green, and a blue LED
- Select three male-male DuPont jumpers, red, green, and blue
- Select a single 330 *ohm* (probably "R" on the label band) resistor
- And get your breadboard out if it's not already

Then, wire your breadboard up to look like this:



If hole is too tight and wire or LED lead bends, use pliers to strengthen and hold wire/lead close to hole then insert



Use *either* rail and *any* holes, but resistor and LED's must *all* use the *same* rail

Assure that the cathode (*short* lead!) is connected to the outside rail and not the MC

NOTE: You can use any color LED's you choose; the color of the LEDs does not matter for this exercise.

NOTE: Likewise, you can use any color wires you wish. We would not normally use red, for instance, as that is typically used only for positive (+) power

These are DuPont Male-Male jumper wires that connect directly from the MC board rails to the LED anode rails.

Have your instructor check all work before applying power to the microcontroller board!

Let There Be Light

- Find the Arduino sketch in your Lesson 7-8 folder (which you copied from your USB drives, right?) named "ClassSketch2-BlinkDeluxe" and open it in the Arduino IDE (double clicking the [.ino] file is easiest)
- Connect your board to your laptop using USB cable (board end first, then the laptop end!)
- Remember to set your board type and port speed! (Change speed if initial upload fails!)
- Upload the sketch! ➔ (this also validates it first)
- Did it work? You've the luck of me Irish ancestors! ♚🍀☺
- It didn't work? Now you can prove yourself worthy of your maker name and troubleshoot the problem! ⚡🔧😎

If you were here last week and already have your LEDs blinking away, you can skip ahead in your PDF to the Modding Challenge!

Anatomy of a sketch

```
const byte LED_RED = 13;
const byte LED_GREEN = 12;
const byte LED_BLUE = 14;
```

Constants are one type of data defined outside the setup and loop sections

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pins as outputs.
  pinMode(LED_RED, OUTPUT);
  pinMode(LED_GREEN, OUTPUT);
  pinMode(LED_BLUE, OUTPUT);
}
```

Here is the Setup section

```
// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_RED, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_RED, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
  digitalWrite(LED_GREEN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_GREEN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
  digitalWrite(LED_BLUE, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BLUE, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

This is the Loop section

What happens if you make the delay values bigger? Smaller? Can you adjust the sketch to perform a “chaser” display?

Advanced Modding Challenge

- **Functions are code statements which return results; functions which *return* a number can be used *in place of* actual numbers.**
- **Example:**

```
    i = 4;
```

and

```
    i = add(2, 2);
```

would assign the same value (4) to the variable "`i`" (`i` and `add()` must be previously defined.)
- **The `random()` function generates and returns a random number; syntax: `random(min, max)`**
- `delay()`**'s syntax is** `delay(milliseconds)`
- **250ms is the minimum reasonable flicker duration humans can perceive**

CHALLENGE: Can you modify your sketch using the random() function to illuminate the LEDs for random intervals?

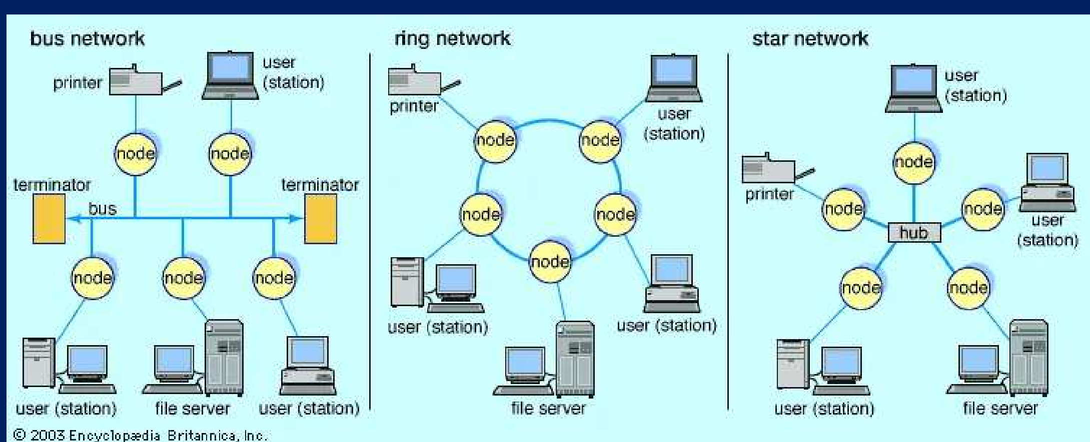
Hint: You have a huge stack of Arduino reference materials, including the Arduino Cookbook– which has examples for the most common functions

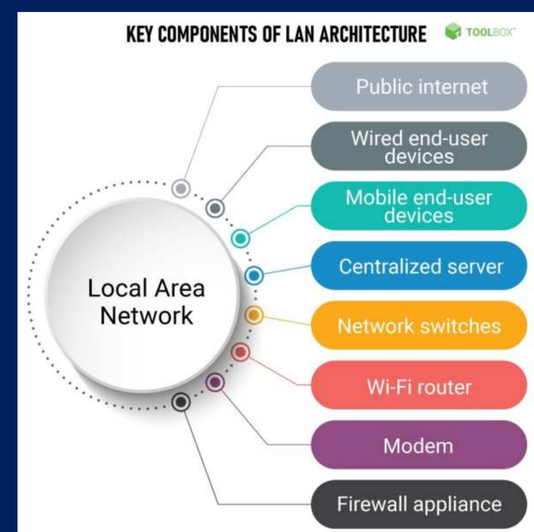
History of Telecommunications Part 2

- Local Area Networks (LANs)
- Wide Area Networks (WANs)
- Wireless LAN (WiFi)
- Cellular Voice Networks
- Cellular data networks
- The Internet - Overview

Local Area Networks

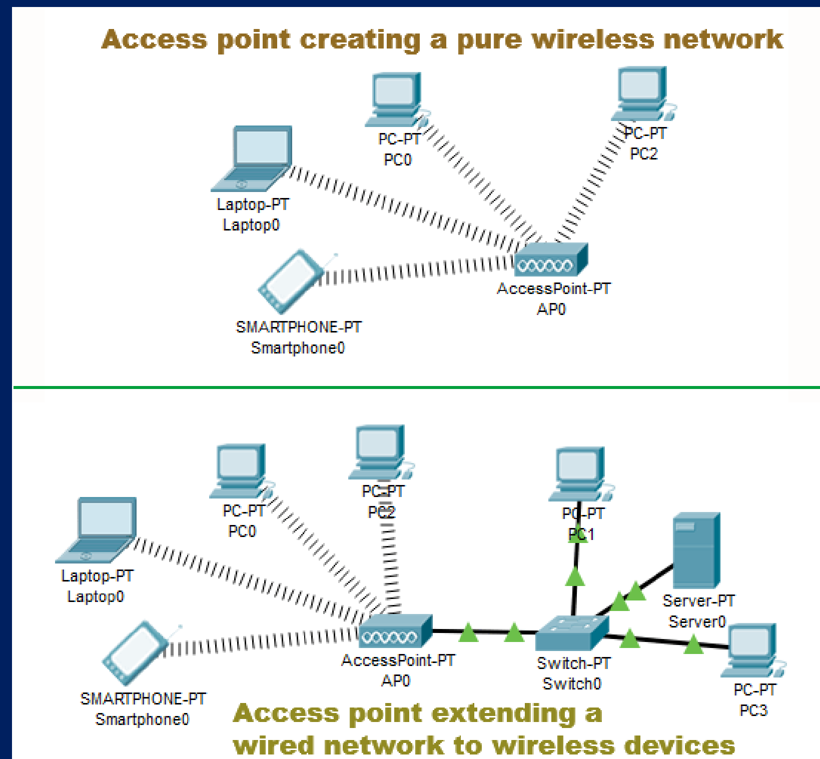
- A local area network (LAN) is a group of computers and peripheral devices that share a common communications line or wireless link to a server within a distinct geographic area. A local area network may serve as few as two or three users in a home office or thousands of users in a corporation's central office.
  Source: TechTarget
- Regardless of size, a LAN's single defining characteristic is that it connects devices that are in a single, limited area. Source: Cisco Systems
- TL;DR a LAN is a group of computers in a single physical location connected together with physical cabling to allow exchange of data





Wireless LAN (WiFi)

- WLAN/Wi-Fi is a family of wireless network protocols, based on the IEEE 802.11 family of standards, which are commonly used for local area networking of devices and Internet access, allowing nearby digital devices to exchange data by radio waves. These are the most widely used computer networks in the world, used globally in home and small office networks to link desktop and laptop computers, tablet computers, smartphones, smart TVs, printers, and smart speakers together and to a wireless router to connect them to the Internet, and in wireless access points in public places like coffee shops, hotels, libraries and airports to provide the public Internet access for mobile devices. Source: Wikipedia
- WLAN is the wireless technology used to connect computers, tablets, smartphones and other devices to the internet. Wi-Fi is the radio signal sent from a wireless router to a nearby device, which translates the signal into data you can see and use. Source: Verizon

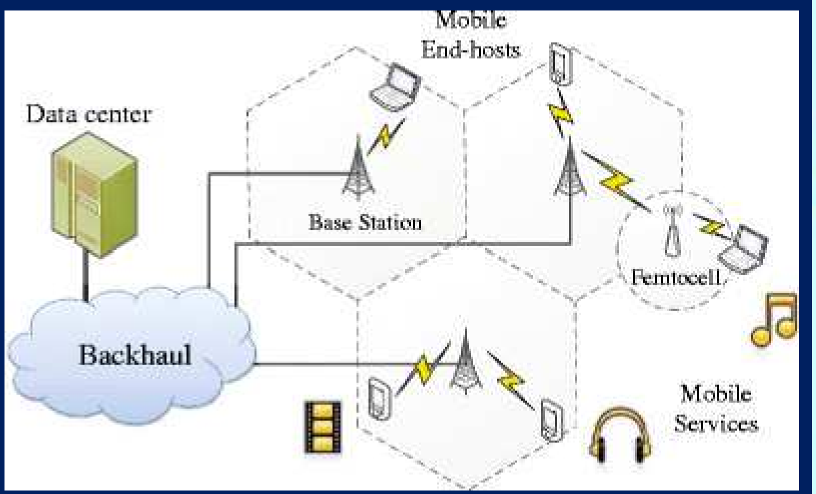


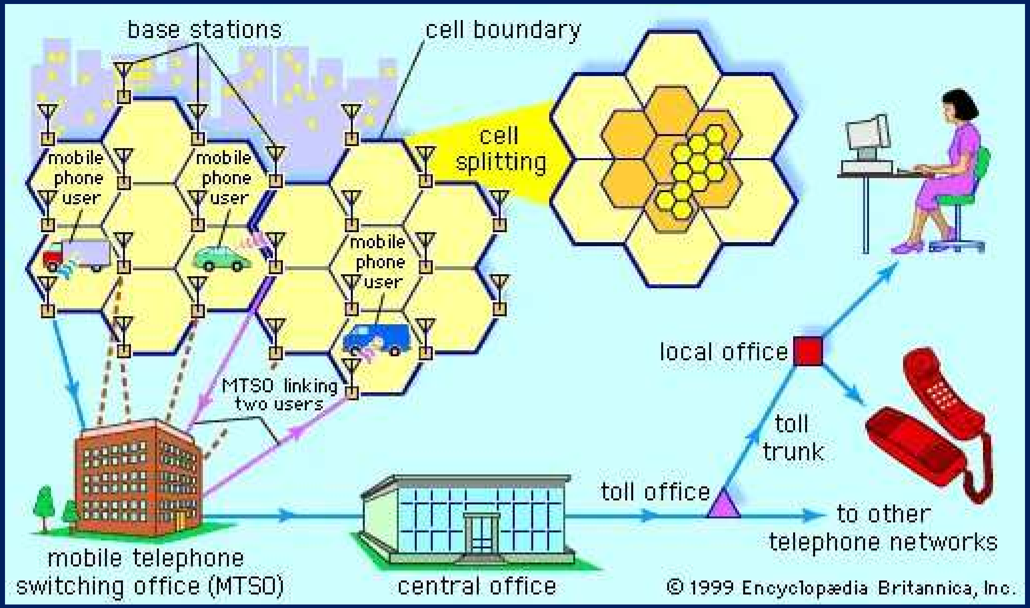
Types of wireless networks

	Wireless LAN (WLAN)	Wireless MAN (WMAN)	Wireless PAN (WPAN)	Wireless WAN (WWAN)
TYPE OF NETWORK	Local area network	Metropolitan area network	Personal area network	Wide area network
GOAL	Provide internet access within a building or limited outdoor area	Provide access outside office and home networks, typically regional	Transmit signals between devices in limited areas, typically 100 meters	Provide access outside the range of WLANs and WMANs
CONNECTIVITY	Cellular	IEEE 802.16 WiMax	Bluetooth, Zigbee and infrared	LTE

Cellular Voice Networks

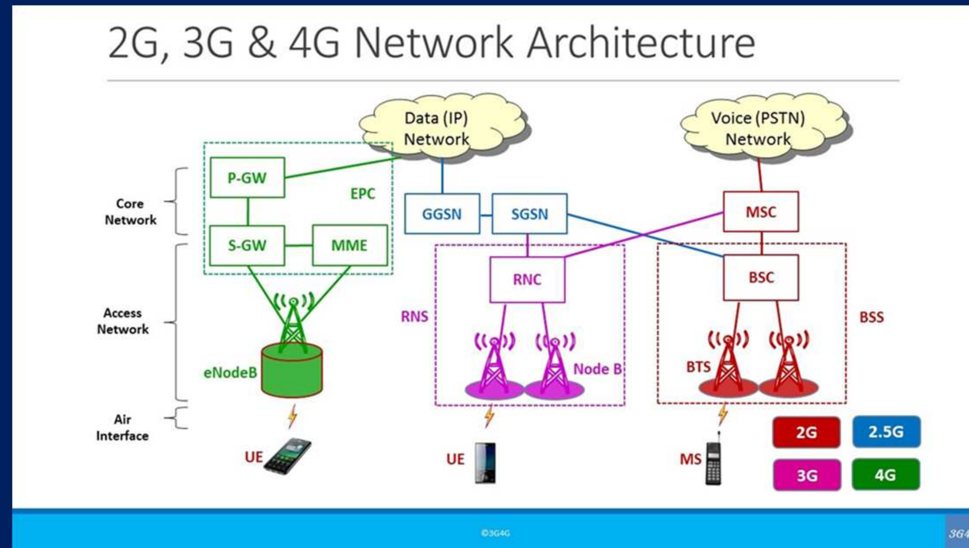
- A cellular network or mobile network is a communication network where the link to and from end nodes is wireless. The network is distributed over land areas called "cells", each served by at least one fixed-location transceiver (typically three cell sites or base transceiver stations). These base stations provide the cell with the network coverage which can be used for transmission of voice, data, and other types of content. A cell typically uses a different set of frequencies from neighboring cells, to avoid interference and provide guaranteed service quality within each cell. When joined together, these cells provide radio coverage over a wide geographic area. Source: Wikipedia
- Cellular networks can be designed to carry voice, data, or both. The earliest cellular networks were voice networks, connecting back to the PSTN (Public Switched Telephone Network).

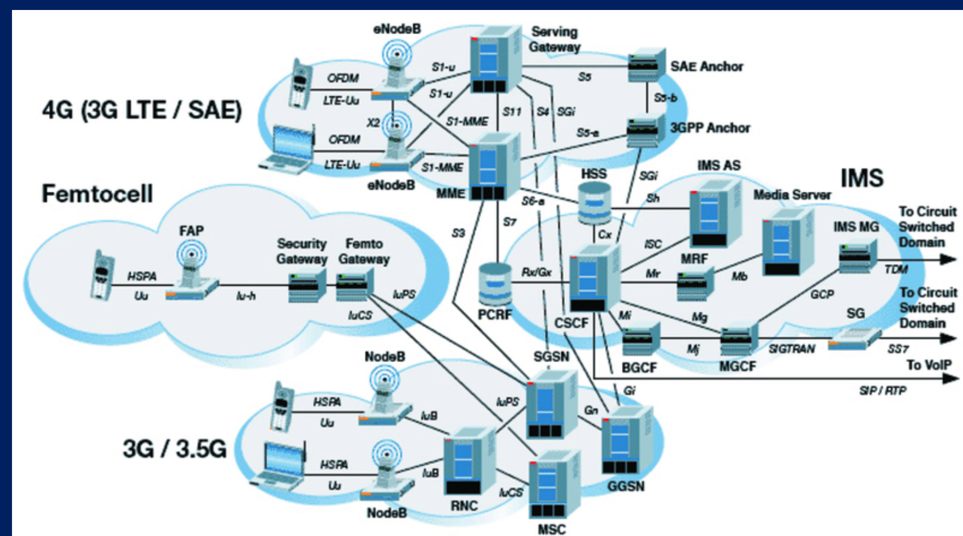




Cellular Data Networks

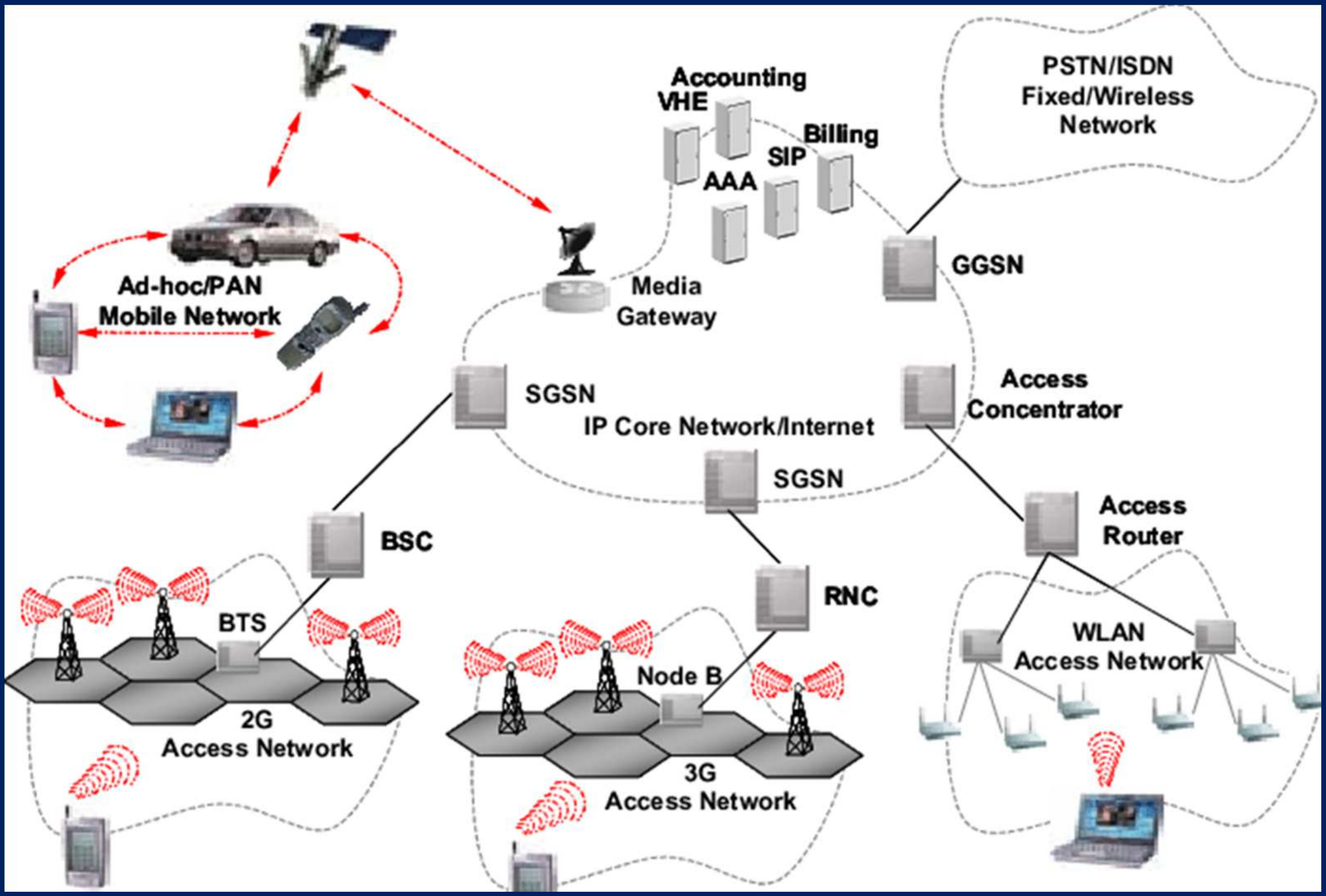
- Cellular Network Data is a *data* connection from the cellular device to the cell site. LTE (4G), 3G and 1X are types of cellular network data connections. Data sessions include, but are not limited to, browsing the Internet such as Google, streaming music, videos and movies, sending and receiving emails, sending and receiving MMS messages, hotspot hosting and application use. While connected to the cellular data network, data usage will pull from the current data plan.
- As with all communications between two computers, "data" refers to information encoded into a bitstream of binary numbers. The method used for encoding varies by network layer, device, and communications protocol.





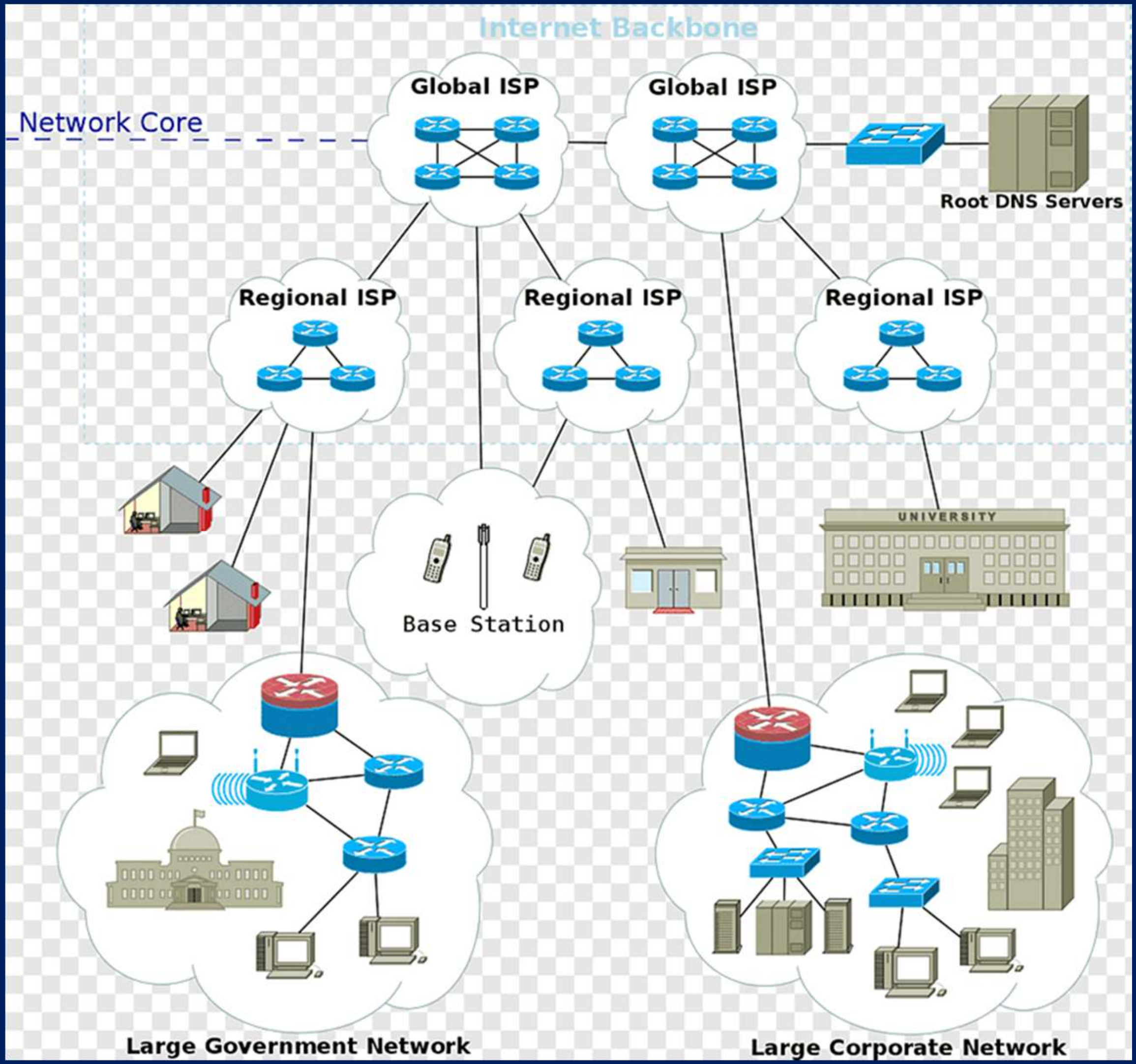
WiFi and Cellular

- WiFi connects to a local router, cellular data connects to a remote router
- WiFi has a limited range, cellular data does not



The Internet - Architecture

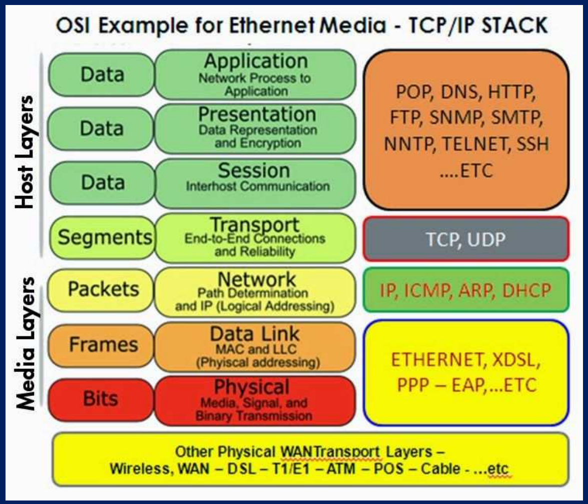
- The Internet is a "network of networks" used to connect many many LANs and WANs together
- Originally the Internet was owned and operated by the US DoD and NSF
- Public connectivity became available in 1992 from a company called PSI

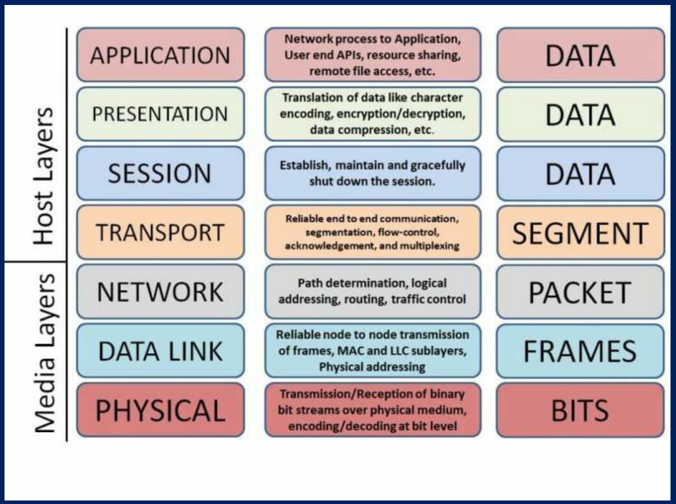


The OSI/ISO Layered Network Model

- In a Quora post asking about the purpose of the OSI model, Vikram Kumar answered this way:
- "The purpose of the OSI reference model is to guide vendors and developers so the digital communication products and software programs they create will interoperate, and to facilitate clear comparisons among communications tools."
- While some people may argue that the OSI model is obsolete (due to its conceptual nature) and less important than the four layers of the TCP/IP model, Kumar says that "it is difficult to read about networking technology today without seeing references to the OSI model and its layers, because the model's structure helps to frame discussions of protocols and contrast various technologies."
- If you can understand the OSI model and its layers, you can also then understand which protocols and devices can interoperate with each other when new technologies are developed and explained.

Source: Network World

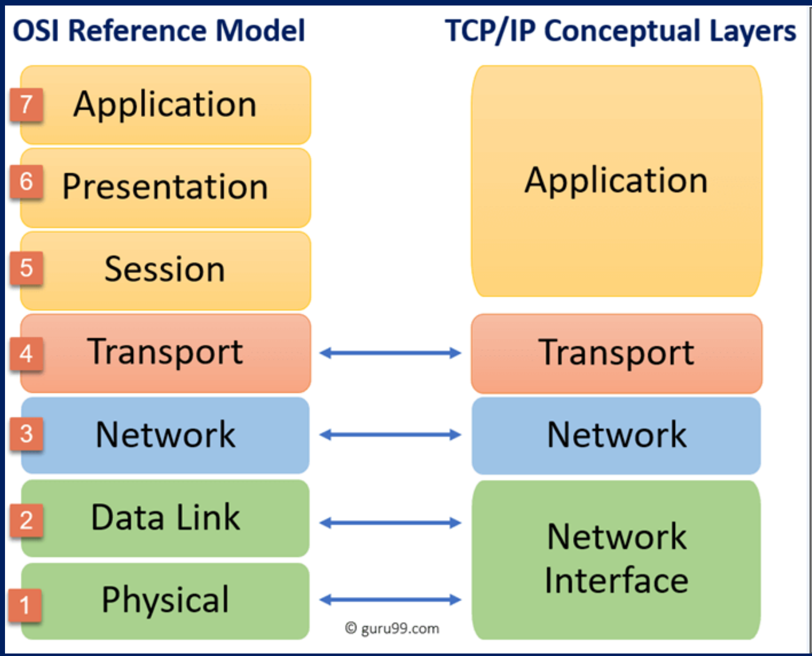


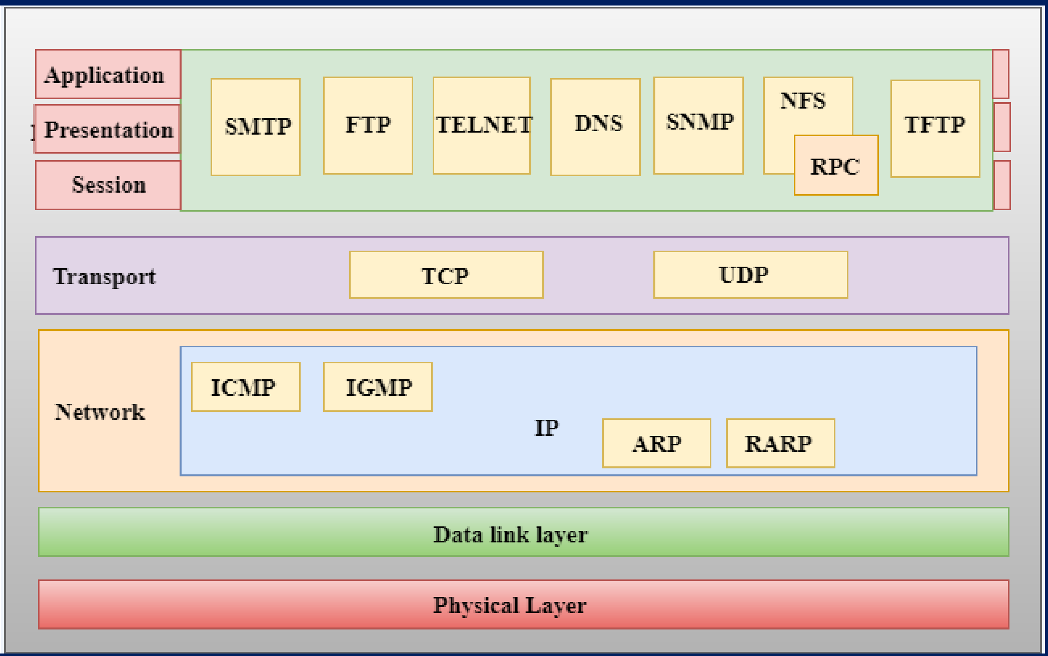


Layer			Protocol data unit (PDU)	Function[21]
Host layers	7	Application	Data	High-level APIs, including resource sharing, remote file access
	6	Presentation		Translation of data between a networking service and an application; including character encoding, data compression and encryption/decryption
	5	Session		Managing communication sessions, i.e., continuous exchange of information in the form of multiple back-and-forth transmissions between two nodes
	4	Transport	Segment, Datagram	Reliable transmission of data segments between points on a network, including segmentation, acknowledgement and multiplexing
Media layers	3	Network	Packet	Structuring and managing a multi-node network, including addressing, routing and traffic control
	2	Data link	Frame	Reliable transmission of data frames between two nodes connected by a physical layer
	1	Physical	Bit, Symbol	Transmission and reception of raw bit streams over a physical medium

ISO/OSI vs The TCP/IP Model

- TCP/IP is the data interchange protocol used by the Internet for computer-to-computer (host-to-host) communications
- It only loosely follows the formal OSI/ISO reference model
- Many consider TCP/IP to be a separate model, but this is not entirely true
- We will not be studying TCP/IP in depth, but it is important to know that it exists and is the underlying protocol used in all web comms





Accurate **Inaccurate**

Formal End of Lesson 7

In next week's exciting episode

- Microcontroller board power requirements
- Board power input options and grounding
- Components with different voltage requirements – level shifting
- External power circuit design
- Connecting and using transistors
- Pulse width modulation and signal generation
- Connecting and using buzzers – Class Sketch #3

Don't forget to return your USB drives!