Internet of Things Weather Station

IEEE Northern Virginia Section

Hands-On Professional Development Series

October 29, 2016 Montgomery College

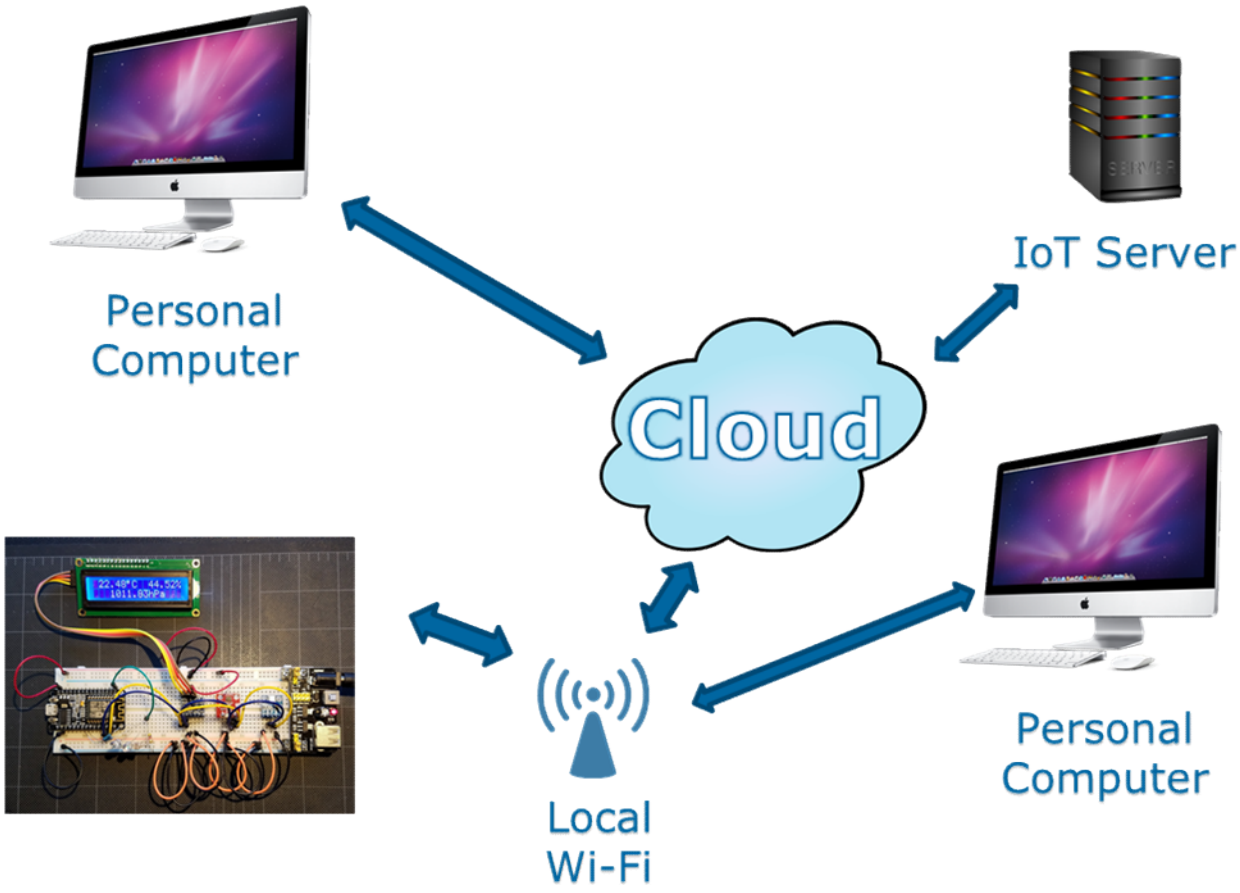
IEEE
Advancing Technology for Humanity

Unboxing & Sketch 01-Blink

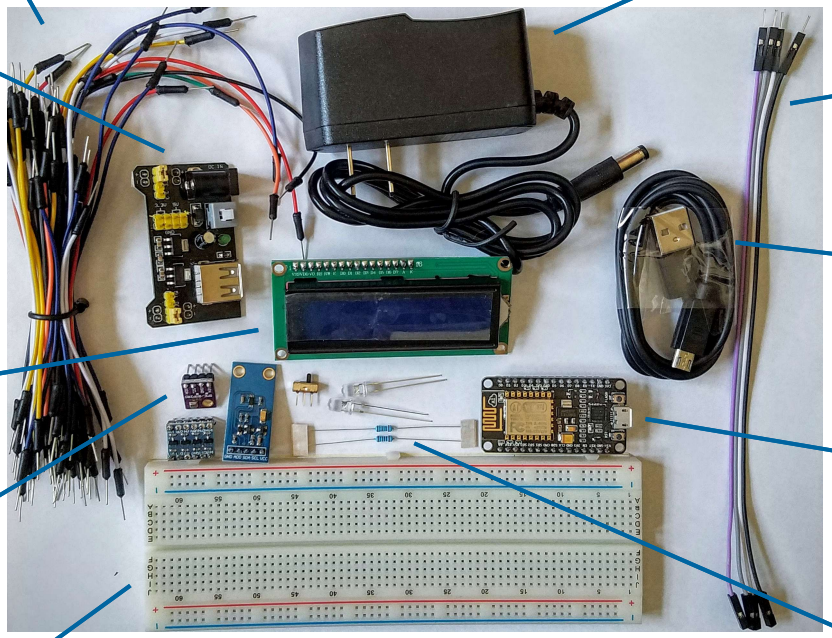
Course Materials

- All course materials are at:
- **http://w4krl.com/projects/ieee-iot/2016october/**
- Download the construction slides so that you can follow along:
  - IEEE IoT Sketch01 – Blink
  - IEEE IoT Sketch02 – Hello World
  - IEEE IoT Sketch03 – Standalone Weather Station
  - IEEE IoT Sketch04 – IoT Weather Station
  - IEEE IoT Sketch05 – Smartphone Weather Station (if time is available)
- We will download Arduino sketches and libraries **when needed**.
- There are links to software, schematics, data sheets, and tutorials.

Project Road Map



Parts List



- Jumpers
- Dual Voltage Regulator
- Liquid Crystal Display
- Level Shifter BME280 BH1750
- Breadboard
- Power Supply
- 4-Wire Jumper
- Micro USB Cable
- NodeMCU
- LEDs (2) Resistors (2) Switch

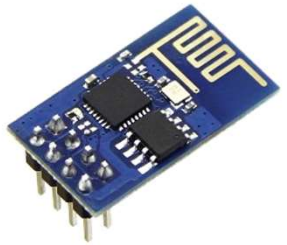
Keep the small parts in the bag for now.

Microcontroller

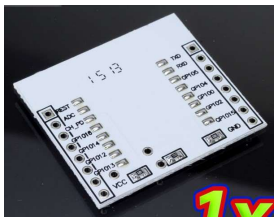
- A microcontroller is a **System on Chip** computer
  - Processor, memory, analog & digital I/O, & radio frequency circuits
- **Embedded** in a device with a dedicated purpose
- Generally low power and often battery powered
- Program is stored in **firmware** & is rarely changed
- Has multiple digital **General Purpose Input / Output,** analog-to-digital conversion, pulse width modulation, timers, special purpose I/O

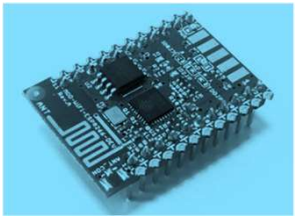
ESP8266 Timeline

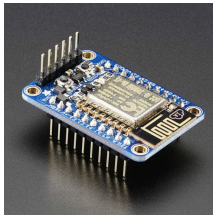
- January 2014 - Introduced by Expressif Systems of Shanghai as a Wi-Fi modem chip. Early adopters used Hayes "AT" commands generated by an Arduino or Raspberry Pi. Not breadboard friendly. No FCC certification.

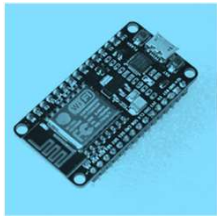


- October 2014 - Expressif released the Software Development Kit (SDK) making its use as a slave modem obsolete.
- March 2015 - Arduino core released – applications skyrocket
- December 2015 – Breadboard & FCC formats introduced
- September 2016 – ESP32 released – dual core + Bluetooth

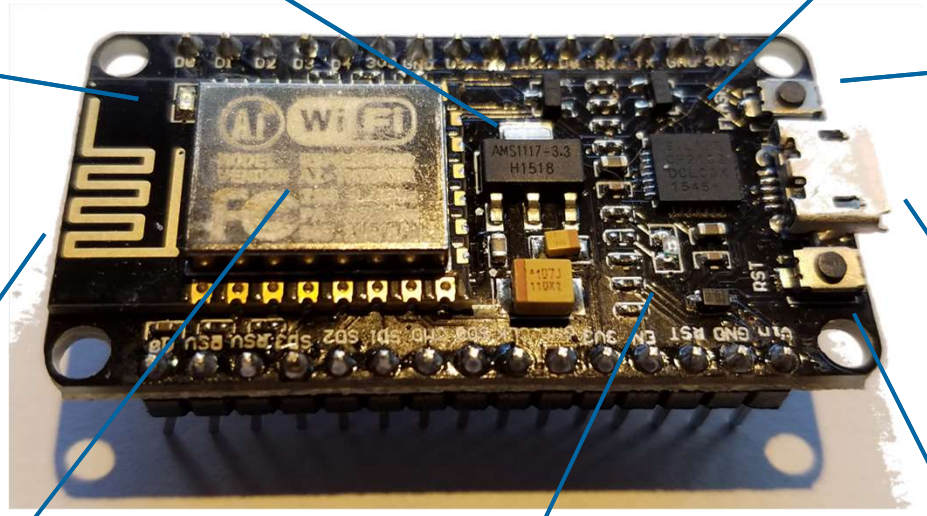




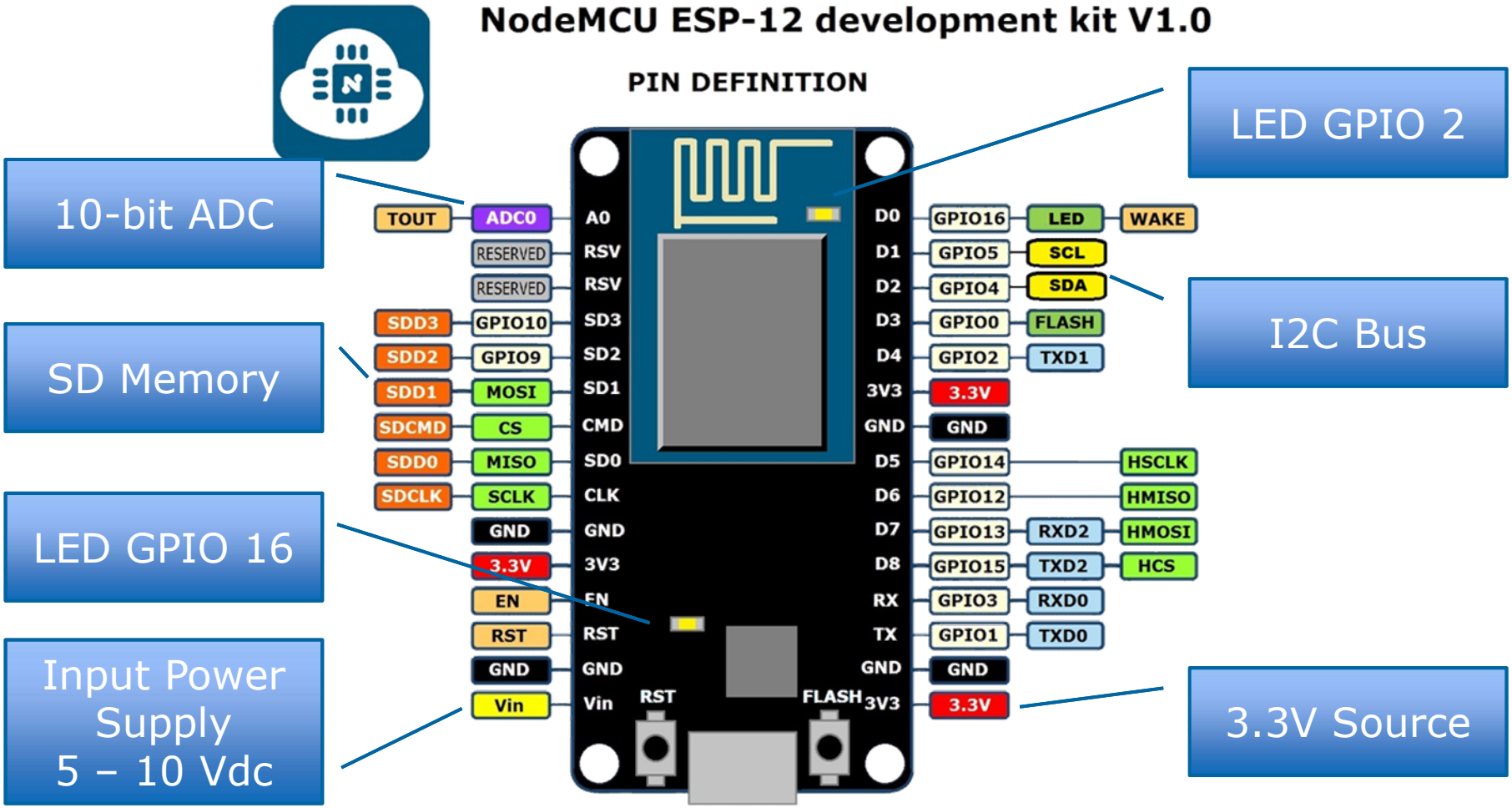




NodeMCU DEVKIT V1.0



NodeMCU Pinout



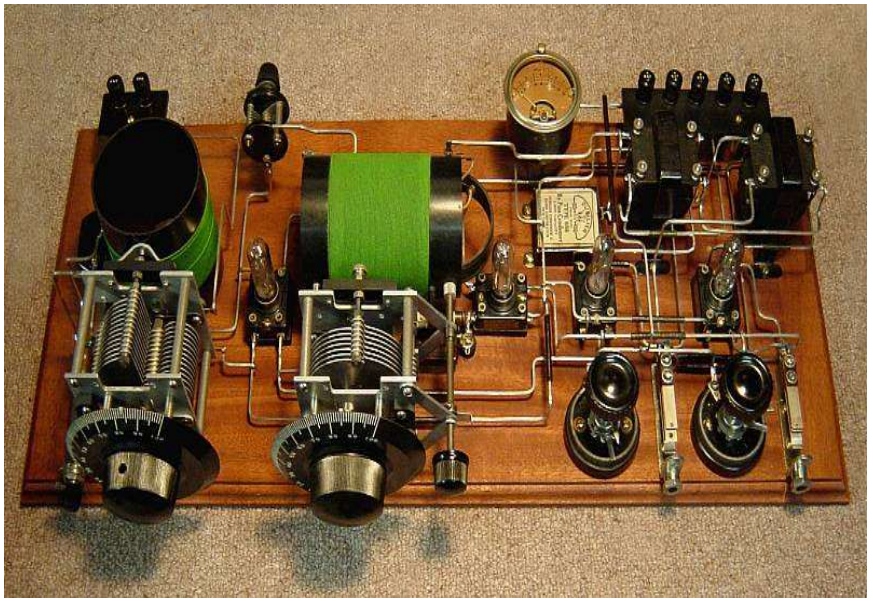
Arduining.com

IEEE Advancing Technology for Humanity

The Original Breadboard

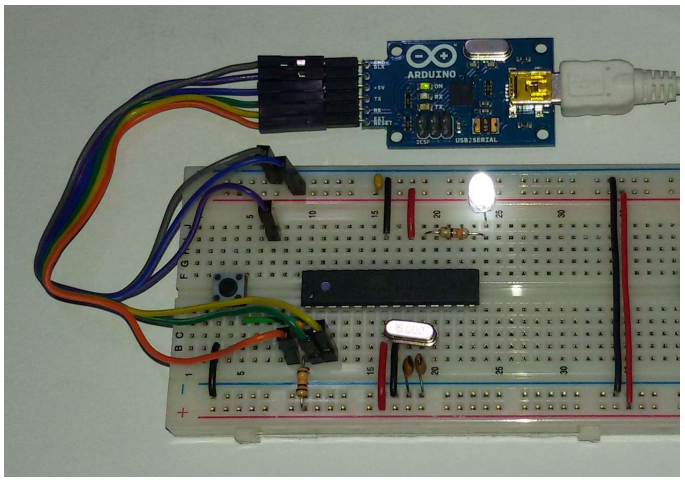
In the Kitchen

On the Air

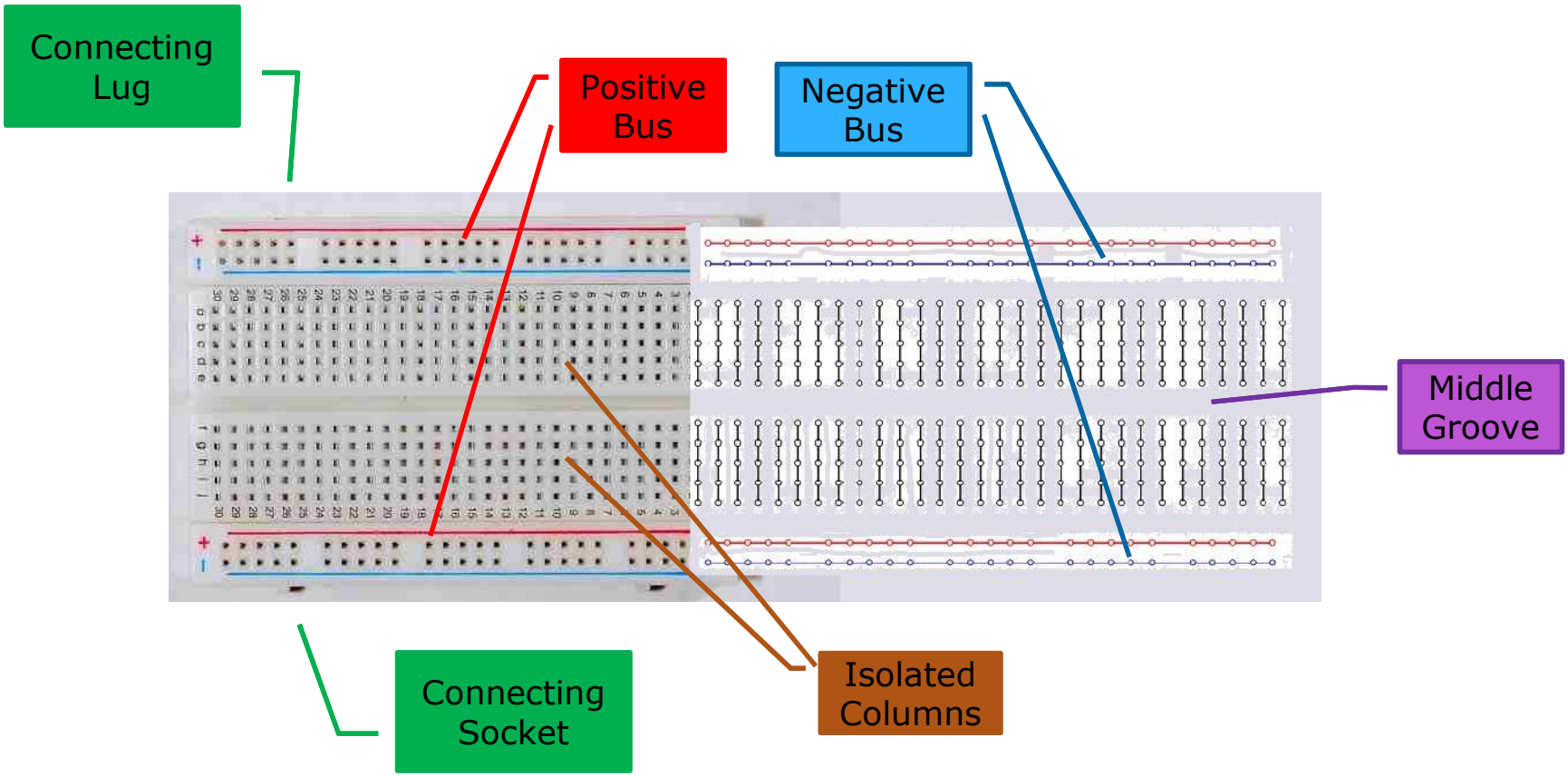


Modern Solderless Breadboard

- Often the first step in prototyping phase
- 0.1-inch (2.54mm) spacing is ideal for Dual Inline Packages and many through hole devices
- **Breakout boards** used for Surface-mount Technology (SMT)
- **DevKits** are breakout boards that add support functions
- Interconnections by component leads, formed wires, or DuPont jumpers



Breadboard Basics

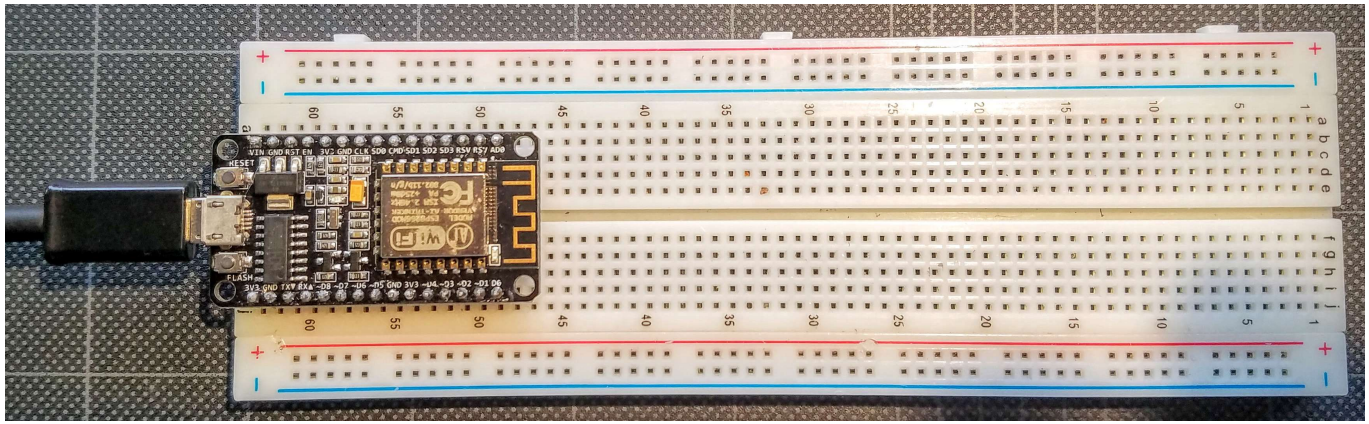


Breadboard Usage

- Use solid (not stranded) wire AWG 22-26
- Use “DuPont” jumpers or precut wires
- Use 1/4Watt or 1/2Watt resistors
- Use needle nose pliers to insert small wires
- Keep DIP devices flat and even when inserting
- Use a male jumper pin to open tight or misaligned sockets
- **Do Not** force a wire or pin into the board

Prepare the breadboard

1. Orient the protoboard with the **+** (Red) bus at the top.
2. Use a jumper pin to **open up 15 points in rows B and I**.
3. Carefully insert the NodeMCU on the **left end** of the breadboard with the **USB connector pointing left**.
4. **Gently apply pressure over the pins** – not in middle of the NodeMCU board.

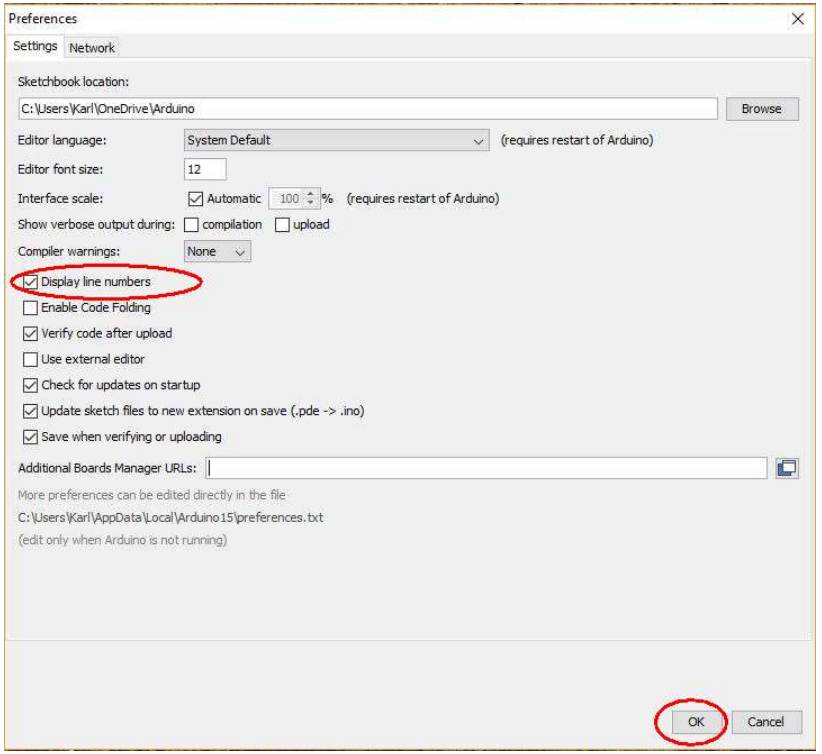


Install Arduino IDE

1. In browser, navigate to www.Arduino.cc
2. Download **Windows installer version** to a known location
3. Click "**Just download**"
4. Run **Arduino-1.6.12-windows.exe**
5. Click the new **Arduino icon** on your desktop

6. Open menu item **File | Preferences**

   Check **Display line numbers**
7. Click **OK**
8. Close the IDE



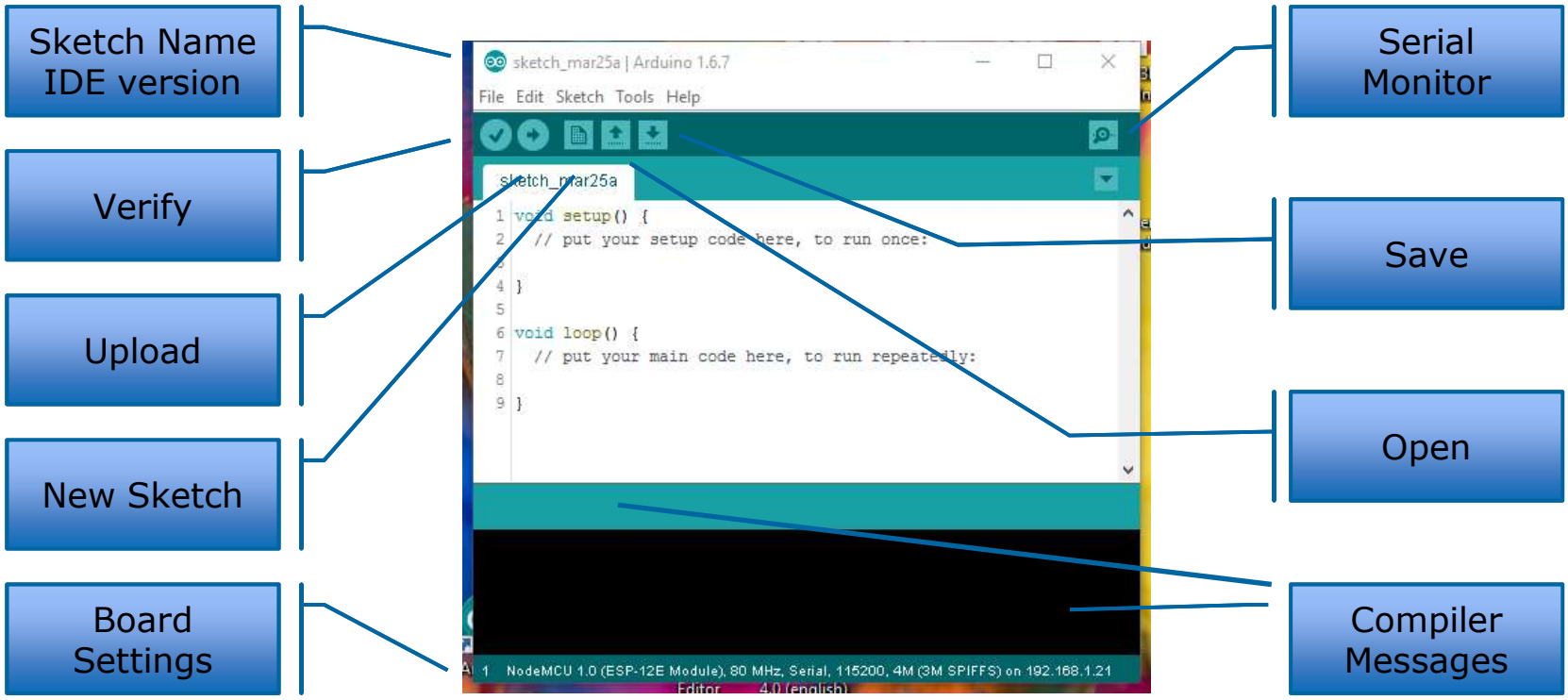
Install ESP8266 Core

1. Reopen the IDE and open menu item **File | Preferences**
2. Type in **Additional Boards Manager URLS**:
   http://arduino.esp8266.com/stable/package_esp8266com_index.json
3. Open menu item **Tools | Boards | Boards Manager…**
4. Type in the search box "**esp8266**" to find the core.
5. Click on "**More info**." Install latest version from dropdown box that appears on the right.
6. Click **Close.**
7. Open menu item **Tools | Boards** and select
   NodeMCU 1.0 (ESP-12 E Module)

Download Course Software (Sketches)

- In web browser open:
  - http://w4krl.com/projects/ieee-iot/2016october/
- Under Arduino Sketch Files download to a known location:
  - **ieee_iot_sketches01-05_v02**
- Go to your download location and unzip the file.
- You will have **six** folders. Copy the folders and paste them in your Arduino sketch folder. This is usually **\documents\Arduino**
- **Remember how to do this.**

Integrated Development Environment (IDE)



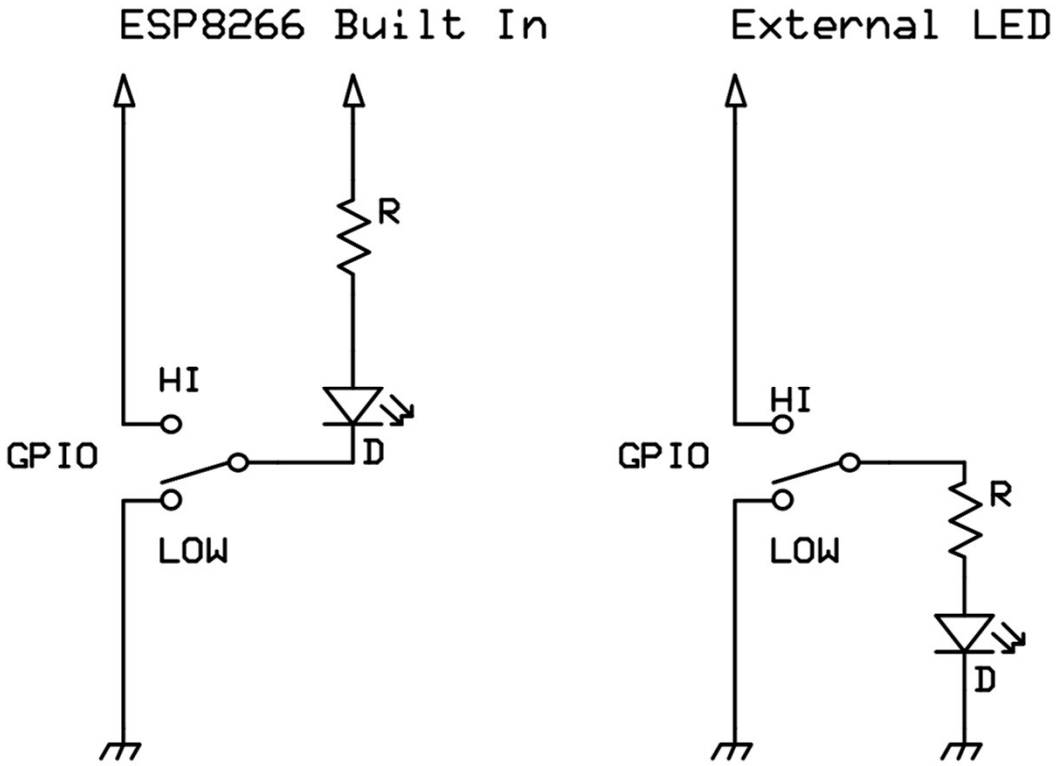
First Sketch - Blink

1. Plugin micro USB cable between your computer and the NodeMCU
2. The USB driver should automatically install
3. Open Arduino IDE
4. Set COM port in menu **Tools | Port**
5. Set **Tools | Board** to **NodeMCU 1.0 (ESP-12E Module)**
6. **File | Open… | IEEE_IoT_Sketch01_Blink_V02**
7. Click  to verify the sketch – compiles without uploading
8. Click  to upload the sketch to the NodeMCU (blue LED blinks)
9. **SUCCESS???!!!**

IEEE_IoT_Sketch01_Blink

```
11 // Global constants
12 const int LED_PIN = 16;           // the built in LED is on GPIO16 (D0)
13 const int DURATION_ON = 100;      // ON duration in milliseconds
14 const int DURATION_OFF = 1000;    // OFF duration in milliseconds
15
16 // Every Arduino sketch must have a setup() function
17 // setup() runs once
18 void setup()
19 {
20   // Initialize the LED_PIN as a digital output
21   pinMode(LED_PIN, OUTPUT);       // set LEDPIN to OUTPUT mode
22 } //setup()
23
24 // Every Arduino sketch must have a loop() function
25 // loop() runs forever
26 void loop()
27 {
28   // pull the output pin to ground
29   // to turn the LED on
30   digitalWrite(LED_PIN, LOW);
31
32   // pause for the ON duration
33  delay(DURATION_ON);
34
35   // pull the output pin to V+
36   // to turn the LED off
37   digitalWrite(LED_PIN, HIGH);
38
39 // Pause for the OFF duration
40   delay(DURATION_OFF);
41 }  // loop()
```

LED Connections



ON when LOW

ON when HIGH

Write, Upload, Observe, Repeat

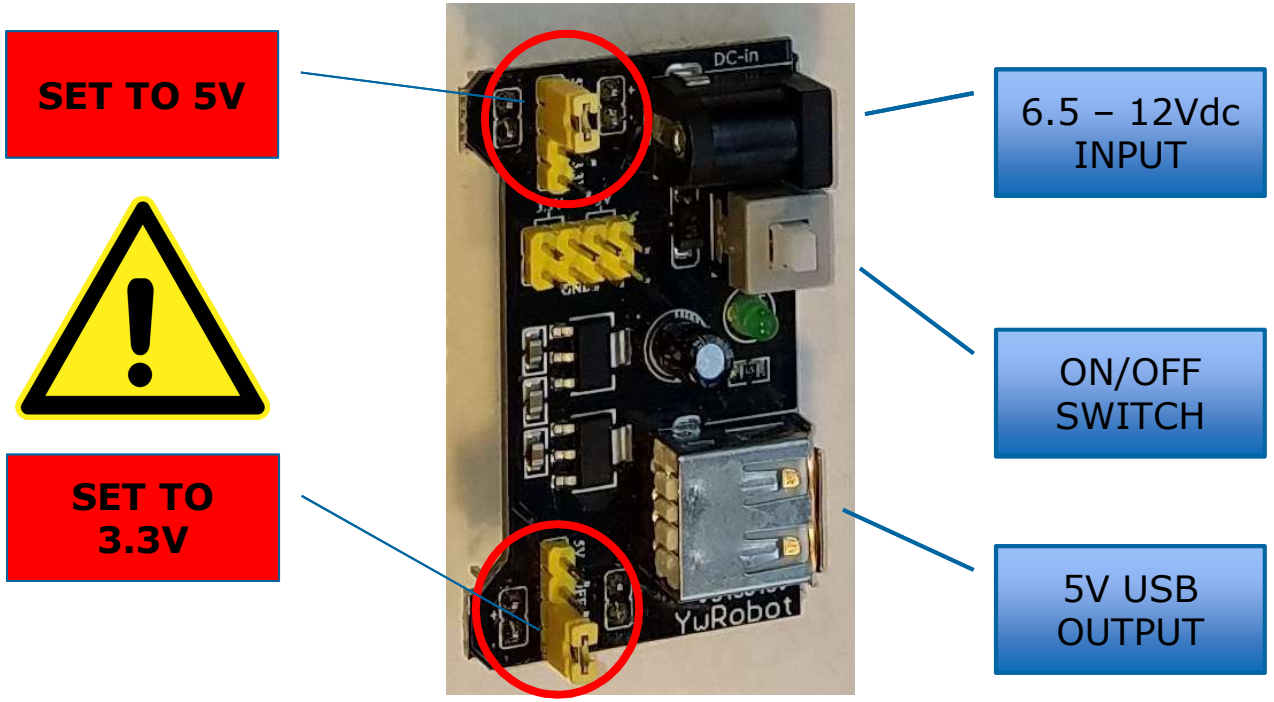
- Change line 13: const int DURATION_ON = **500**;
- Change line 14: const int DURATION_OFF = **500**;
- Upload the modified code to the NodeMCU
- Try other values and upload
- Observe the change in operation
- Return to some reasonable values and upload firmware to prepare for the next hardware step

Color Code Guide

- Electrons don't care but we do.
- Colors help you keep track of signals.
- Try to use one color for one signal, another color for a different signal.
- For our use, 120mm (4-1/2-in) jumpers are sufficient.

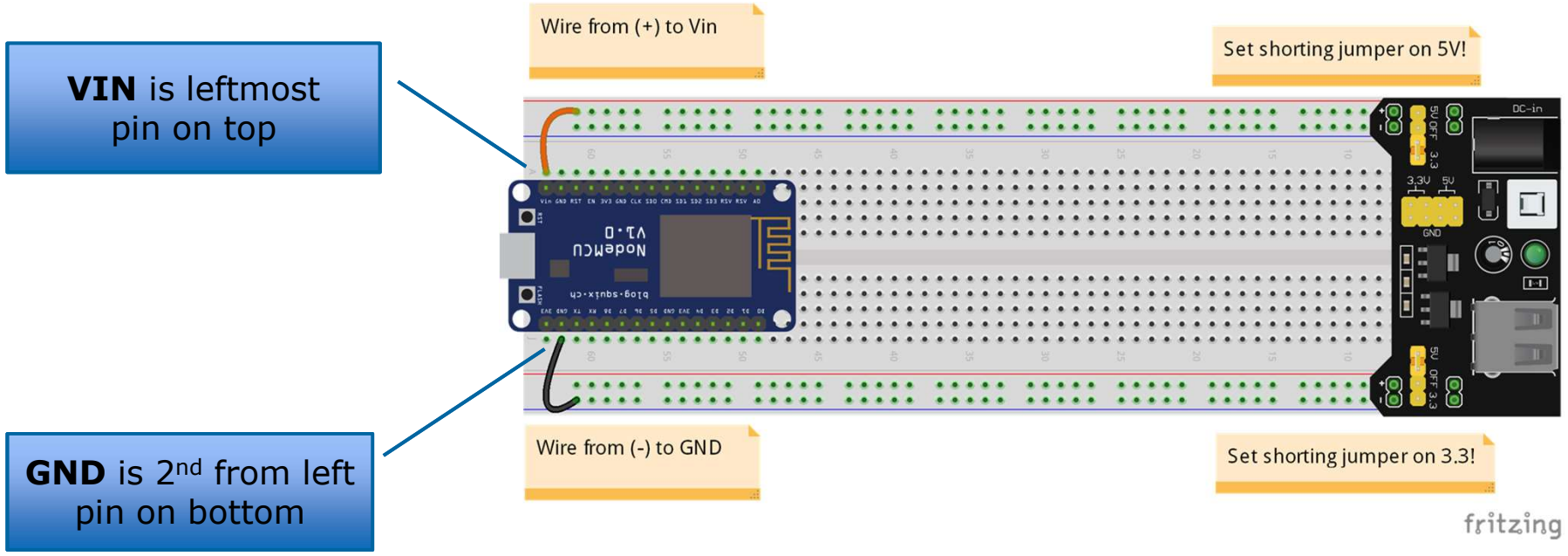
Signal	Quantity	Ideal	Plan B	Generic
+ (5V & 3V3)	6	RED	ORANGE	A BRIGHT color
- (GND)	5	BLACK	GREEN	A DARK COLOR
SCL	3	YELLOW	WHITE	A different BRIGHT color
SDA	3	BLUE	PURPLE	A different DARK color
Other	3	Any	Any	Any

Dual Voltage Regulator



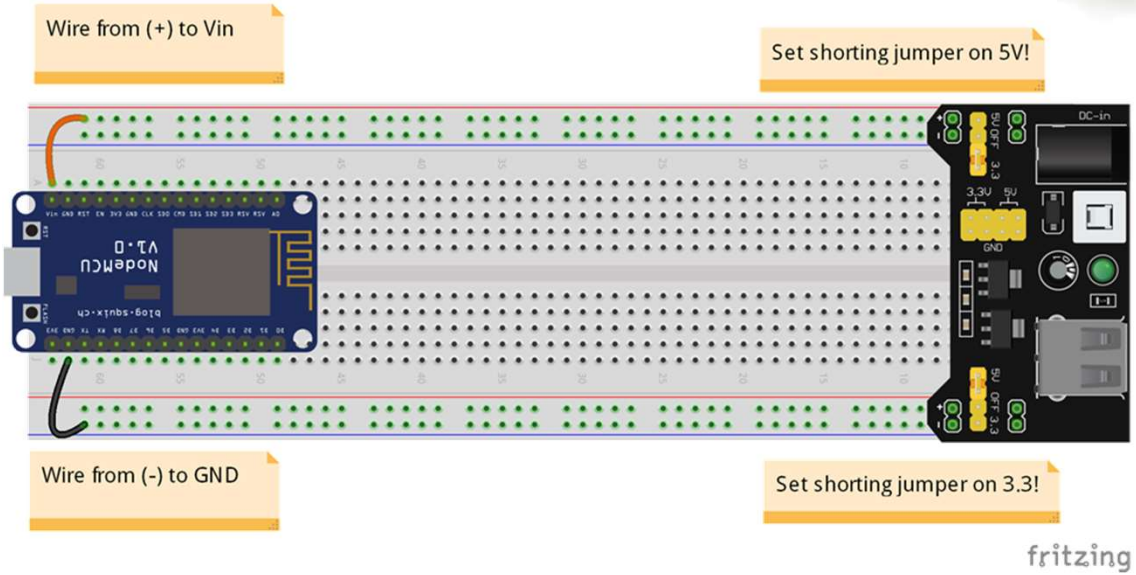
Wire the power supply circuit

1. Unplug the USB cable from the NodeMCU.
2. Insert a jumper from the top **+** rail to **VIN**.
3. Insert a jumper from the bottom **–** rail to **GND**.



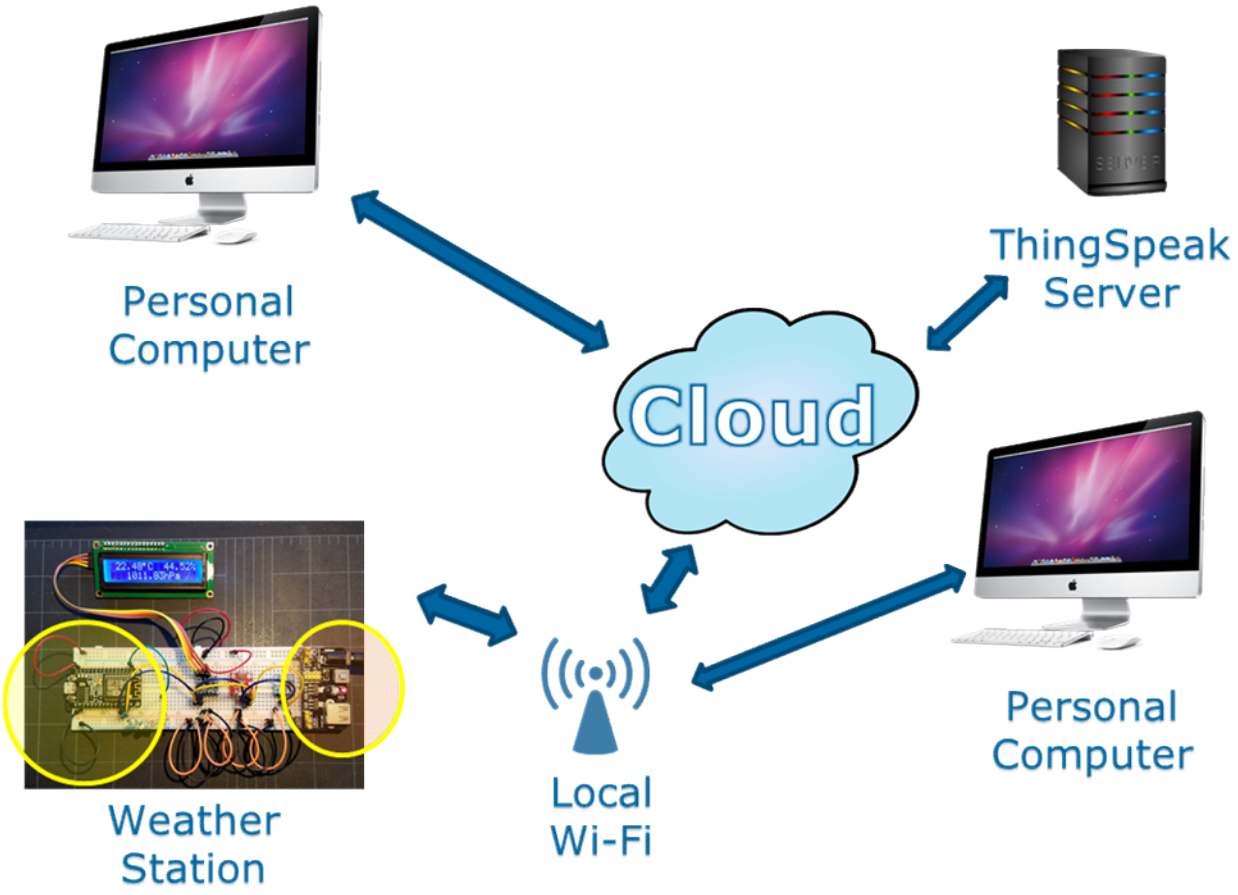
Running Firmware

- Plug in the power supply to a wall socket.
- Plug in the power cord to the regulator.
- Turn on regulator.
- NodeMCU LED should blink.



IEEE Advancing Technology for Humanity

Road Map Progress



Accomplishments

- Became familiar with microcontrollers and the NodeMCU.
- Learned how to use a solderless breadboard.
- Installed the Arduino IDE.
- Loaded the ESP8266 core.
- Compiled and uploaded firmware.
- Experienced the write, compile, observe, debug, repeat cycle.
- Demonstrated the persistence of firmware.

Questions?