Electronics

Current is a gateway drug to magic smoke.

A list of basic stuff to buy when starting out

Soldering

You might've noticed that some things are super easy to solder, some are super hard.

That's because you may be using a ruined tip(it's supposed to be shiny, not black), low-power iron or solder without lead(Pb). The last possible annoying cause is that the contacts you are soldering on are lead-free. For those I've had to use minimum 370°C and half of the connections I made were garbage.

How to not ruin a tip

Do not leave the iron on when you are not using it.

Clean the tip by applying solder, wiping it clean on a sponge or steel wool and then coat it with a thin layer of solder to prevent oxidation before you turn it off and leave it, or coat it right after turning it off.

If you use a sponge, make sure it's damp but not soaked.

Components

Wires

http://www.powerstream.com/Wire_Size.htm

LED strips

LED strips from "best" to "worst". Price is according to that.

SK9822 =~ APA102 > APA102C > WS2812B

Capacitors

If you overvolt a capacitor it will fail in moments, if you go above 50-70% of the voltage limit it will fail in months, or years. All electrolytic caps will fail eventually though, even if idle

Resistors

https://learn.sparkfun.com/tutorials/resistors

Batteries

Alkaline typically doesn't have a mAh rating because it's so current dependant

- 2 batteries in series will have the same capacity as 1 battery, it's double the voltage.
- In parallel you get the same voltage, but double the capacity.
- A 1Ah battery at 1.5V can deliver 1.5Wh in an ideal world. Two batteries can deliver 3Wh, so if you put them in series they'll be at 3V, thus the capacity must be 1Ah still to get to 3Wh
- Likewise, if you put them in parallel, the voltage is still 1.5V, in which case you'll get 2Ah from them and end up with the 3Wh

AA, AAA

To measure voltage just put multimeter to voltage mode and connect leads to positive and negative of the batteries. If the battery is non-rechargeable and way below it's supposed voltage (think <1V for 1.5V battery) it's pretty much dead.

Laws and equations

Ohm's law

The law stating that the direct current flowing in a conductor is directly proportional to the potential difference between its ends. It is usually formulated as V = IR, where V is the potential difference, or voltage, I is the current, and R is the resistance of the conductor.

To explain that in a more useful way - You can calculate for a third variable if you know the other two.

- Voltage(V) = Resistance*Current
- Current(A or I) = Voltage/Resistance
- Resistance(R or Ω) = Voltage/Current

That means if you wanted to know what resistor to use if you had an LED that drops 3.3V, works at 20mA and your power source was 5V - Since the current is the same across the circuit, you'd calculate the voltage drop of the resistor divided by the current, which is 1.7V in this circuit as 3.3V is already being dropped by the LED - (5-3.3)/0.02 and would end up knowing you need to use a resistor that is 85Ω or close to that value.

Projects

Keep in mind these projects assume you have the parts kit linked at the top of the page.

ESP8266

esp8266-node-mcu-pinout.png

Assuming Linux your user needs to be in the lock and uucp groups.

Get the Arduino IDE and setup ESP8266 in it according to this github readme.

Connect the board via the microUSB connector and in Arduino>Tools>Port select the ttyUSB0 port. Under Board select the NodeMCU 1.0 one.

Pulling the D0 pin up makes one of the integrated LEDs light up, so it's a quick way to test everything is running, so let's do just that and blink it.

You can use the number notation (GPIO10 is defined as '10') or the pin notation, like D0.

```
void setup() {
  pinMode(D0, OUTPUT);
}
void loop() {
  digitalWrite(D0, HIGH);  // turn the LED on.
  delay(1000);              // wait for 1 second.
  digitalWrite(D0, LOW);  // turn the LED off.
  delay(1000);            // wait for 1 second.
}
```

Upload it to the board, it should boot up and the blue LED should start blinking.

Now let's try with an actual LED, grab a red 5mm one. If you'd read the specs sheet you'd know it has a 2V voltage drop and is rated for 20mA of forward current. You need a resistor unless you want to fry the LED with current, so according to Ohm's law a resistor for the remaining 1.3V voltage drop needs to be (3.3-2)/0.02, so around 65 Ohms.

esp8266_led.png

Connect them like so and now you should have the LED permanently glowing. Quite boring, let's make it blink.

Re-connect the resistor from the 3v3 pin to the SD3 one (GPIO10) and replace the code with this:

```
const short int LED1 = 10;
void setup() {
  Serial.begin(115200);
  pinMode(LED1, OUTPUT);
}

void loop() {
  digitalWrite(LED1, HIGH);  // turn the LED on.
  Serial.println("Turned LED1 on");
  delay(1000);              // wait for 1 second.
  digitalWrite(LED1, LOW);  // turn the LED off.
  Serial.println("Turned LED1 off");
  delay(1000);            // wait for 1 second.
}
```

Now it should blink! And since we added serial connection and some debugging output, we should be able to connect to it via some software that can read serial, for example 'screen'. Note that 115200 is the baud rate, affecting how fast you can send/receive data.

```
screen /dev/ttyUSB0 115200
```

If you see output every second with on/off sentence, great! You can do ^A and then K to kill the window.

Now let's make the LED Wi-Fi controllable! Same physical setup, different code. Just change the SSID and password in the example:

```
#include <ESP8266WiFi.h>

const short int LED1 = 10;

const char* ssid = "ssid";
const char* password = "password";
WiFiServer server(80);

void setup() {
  Serial.begin(115200);
  pinMode(LED1, OUTPUT);
  // WiFi.mode(m): set mode to WIFI_AP, WIFI_STA, or WIFI_AP_STA.
  WiFi.mode(WIFI_STA);
  delay(10);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    Serial.println("Wi-Fi not connected, retrying... ");
    delay(500); // Do not use this delay in SoftAP mode
  }
  // Start the wifi server
  server.begin();

  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}

void loop() {
  if(WiFi.status() != WL_CONNECTED) {
```

```
  Serial.println("Wi-Fi not connected, retrying... ");
  delay(500); // Do not use this delay in SoftAP mode
 }
 WiFiClient client = server.available();
 if (!client) {
  //Serial.println("No client connected, suiciding.");
  return;
 }

 int insanity = 0;
 // Wait until the client sends some data
 while (!client.available()) {
  insanity++;
  if (insanity == 1000) {
   Serial.println("And nobody came...");
   return;
  }
  delay(1);
  Serial.println("Waiting for client to send data.");
  //client = server.available(); // Check if the connection didn't break, if yes, kill it.
  //if (!client) {
   //return;
  //}
 }
 Serial.println("Client available, receiving data...");

 // Read the first line of the request
 String request = client.readStringUntil('\r');
 client.flush();

 // Match the request
 if (request.indexOf("/OFF") != -1) {
  digitalWrite(LED1, LOW);
 }
 if (request.indexOf("/ON") != -1) {
  digitalWrite(LED1, HIGH);
 }
 // Return the response
 String html = String("HTTP/1.1 200 OK\r\n") +
                "Content-Type: text/html\r\n" +
```

```
                "\r\n" +
                "<!DOCTYPE HTML>" +
                "<html>" +
                "<head>" +
                "<style media=\"screen\" type=\"text/css\">" +
                "   .button {" +
                "        background-color: #000000;" +
                "        color: #FFFFFF;" +
                "        padding: 10px;" +
                "        border-radius: 10px;" +
                "        -moz-border-radius: 10px;" +
                "        -webkit-border-radius: 10px;" +
                "        margin:10px" +
                "    }"
                "    .small-btn {" +
                "        width: 50px;" +
                "        height: 25px;" +
                "    }" +
                "    .medium-btn {" +
                "        width: 70px;" +
                "        height: 30px;" +
                "    }" +
                "    .big-btn {" +
                "        width: 90px;" +
                "        height: 40px;" +
                "    }" +
                "</style>" +
                "</head>" +
                "<body>" +
                "<a href=\"/ON\"><div class=\"button big-btn\">ON</div></a>" +
                "<a href=\"/OFF\"><div class=\"button big-btn\">OFF</div></a>" +
                "</body>" +
                "</html>";
client.print(html);
delay(1);
}
```

Get the IP either from serial or your DHCP server list. Open it in your browser and you should see two buttons that turn the LED off and on.

IR

Now that you know enough to get going, time to dive into something fun, IR. Let's make a thing that can both receive and send.

You'll need a 2-pin IR LED, 3-pin 1838 IR receiver, 2N2222 NPN transistor, 100 Ohm resistor and some wires.

ir_rec_send.png

You will need to setup the IR library for ESP8266

Keep in mind that at least on the NodeMCU board linked in the doc with stuff to buy you can't use GPIO 1,3,9,10,15,16 for IR receive(and possibly send), the limitations are mostly explained here.

I used this code to test which GPIOs I can use for receiving:

```
#include <ESP8266WiFi.h>

#include <IRremoteESP8266.h>
#include <IRrecv.h>
#include <IRsend.h>
#include <IRtimer.h>
#include <IRutils.h>
#include <ir_Argo.h>
#include <ir_Daikin.h>
#include <ir_Fujitsu.h>
#include <ir_Kelvinator.h>
#include <ir_LG.h>
#include <ir_Magiquest.h>
#include <ir_Midea.h>
#include <ir_Mitsubishi.h>
#include <ir_Toshiba.h>
#include <ir_Trotec.h>

const char* ssid = "ssid";
const char* password = "password";
WiFiServer server(80);

int RECV_PIN = 0; //IR IN
IRrecv irrecv(RECV_PIN);
decode_results results;
```

```
void setup() {
  Serial.begin(115200);
  irrecv.enableIRIn();

  // WiFi.mode(m): set mode to WIFI_AP, WIFI_STA, or WIFI_AP_STA.
  WiFi.mode(WIFI_STA);
  delay(10);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    Serial.println("Wi-Fi not connected, retrying... ");
    delay(500); // Do not use this delay in SoftAP mode
  }
  // Start the wifi server
  server.begin();
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());

  Serial.print("RECV_PIN now: ");
  Serial.println(RECV_PIN);
}
void loop() {
  if (irrecv.decode(&results)) {
    Serial.println((long int)results.value, HEX);
    irrecv.resume(); // Receive the next value
    return; // Kill the cycle
  }
  if(WiFi.status() != WL_CONNECTED) {
    Serial.println("Wi-Fi not connected, retrying... ");
    delay(500); // Do not use this delay in SoftAP mode
  }
  WiFiClient client = server.available();
  if (!client) {
  //  Serial.println("No client connected, suiciding.");
    return;
  }

  int insanity = 0;
  // Wait until the client sends some data
  while (!client.available()) {
```

```
  insanity++;
  if (insanity == 1000) {
    Serial.println("And nobody came...");
    return;
  }
  delay(1);
  Serial.println("Waiting for client to send data.");
 }
 Serial.println("Client available, receiving data...");

 // Read the first line of the request
 String request = client.readStringUntil('\r');
 client.flush();

 // Match the request
 if (request.indexOf("/NEXT_PIN") != -1) {
  nextPin();
 }
 // Return the response
 String html = String("HTTP/1.1 200 OK\r\n") +
                "Content-Type: text/html\r\n" +
                "\r\n" +
                "<!DOCTYPE HTML>" +
                "<html>" +
                "<head>" +
                "<style media=\"screen\" type=\"text/css\">" +
                "   .button {" +
                "        background-color: #000000;" +
                "        color: #FFFFFF;" +
                "        padding: 10px;" +
                "        border-radius: 10px;" +
                "        -moz-border-radius: 10px;" +
                "        -webkit-border-radius: 10px;" +
                "        margin:10px" +
                "    }"
                "    .small-btn {" +
                "        width: 50px;" +
                "        height: 25px;" +
                "    }" +
                "    .medium-btn {" +
                "        width: 70px;" +
```

```
                "        height: 30px;" +
                "    }" +
                "    .big-btn {" +
                "        width: 90px;" +
                "        height: 40px;" +
                "    }" +
                "</style>" +
                "</head>" +
                "<body>" +
                "<a href=\"/NEXT_PIN\"><div class=\"button big-btn\">NEXT_PIN</div></a>" +
                "</body>" +
                "</html>";
  client.print(html);
  delay(1);
}

void nextPin() {
  irrecv.disableIRIn();
  delay(100);
  RECV_PIN = RECV_PIN+1; //IR IN
  Serial.print("RECV_PIN now: ");
  Serial.println(RECV_PIN);
  IRrecv irrecv(RECV_PIN);
  decode_results results;
  irrecv.enableIRIn(); // Start the receiver
  delay(100);
}
```

Revision #1
Created 2021-06-29 11:33:55 UTC by C0rn3j
Updated 2021-06-29 11:39:21 UTC by C0rn3j