POLITECNICO
MILANO 1863

ANTLAB

Internet of Things Lab

Lab 4: Wokwi

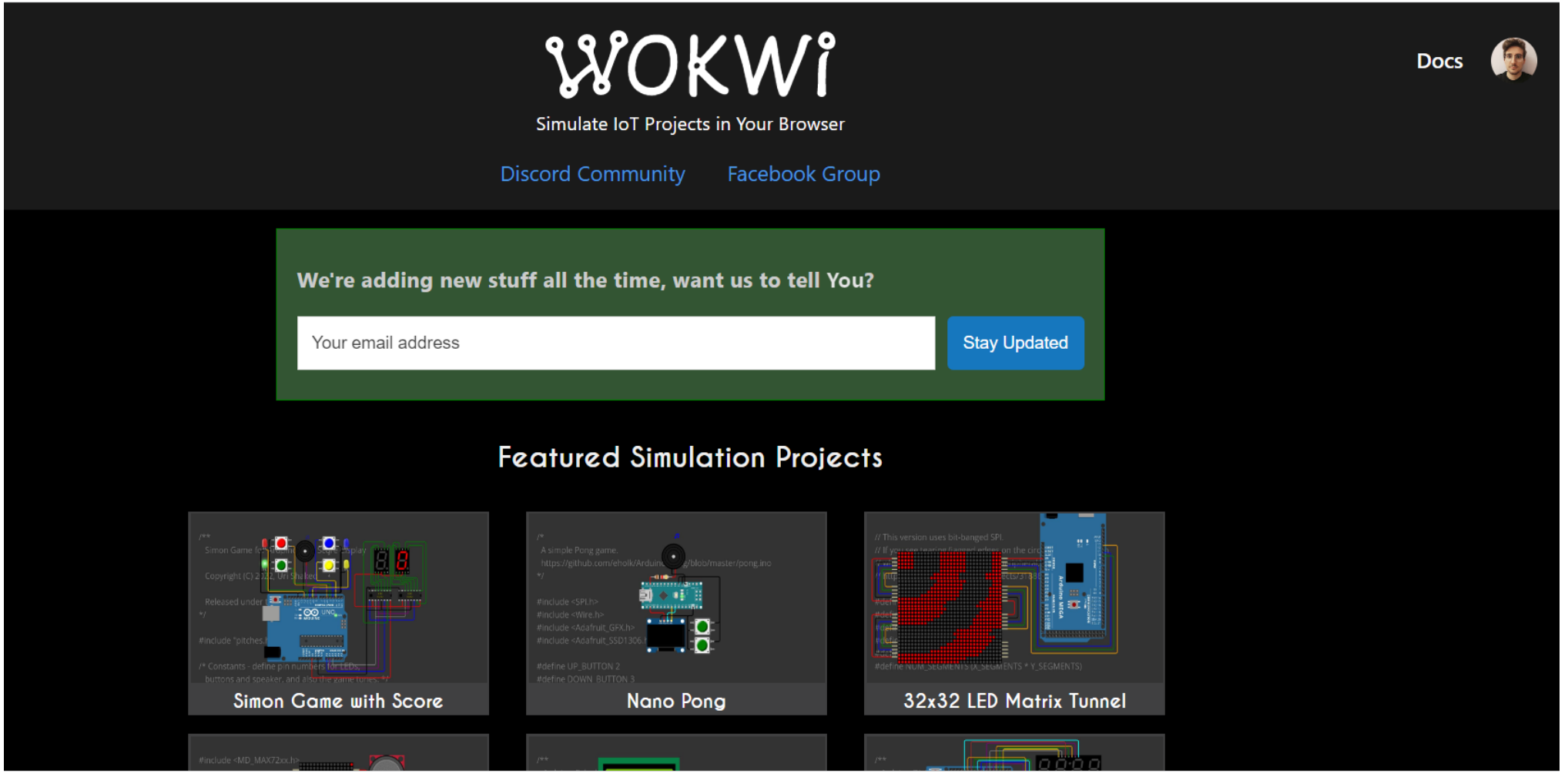
Agenda

- **Wokwi**
  - Platform, devices etc…
- **EPS 32 examples**
  - Leds, Interrupt, PWM, etc…
- **Playing with MQTT**
  - Simple Pub/Sub application on ESP32s

POLITECNICO
MILANO 1863

Wokwi Platform

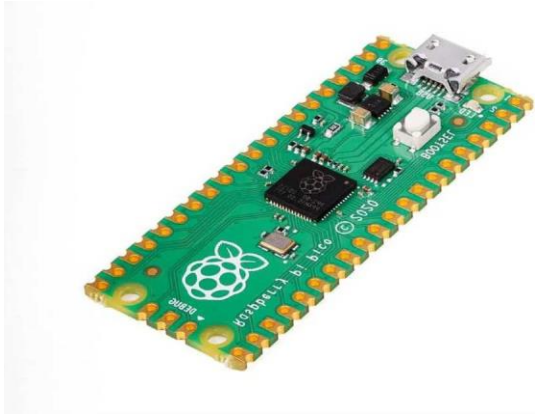
Wokwi platform



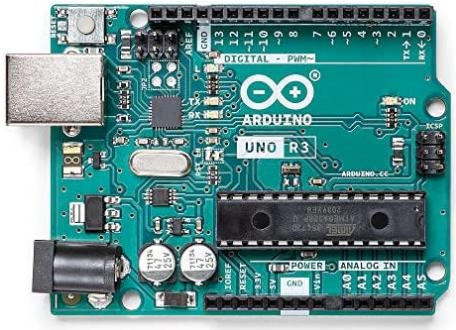
Available at: **www.wokwi.com**

ANTLAB

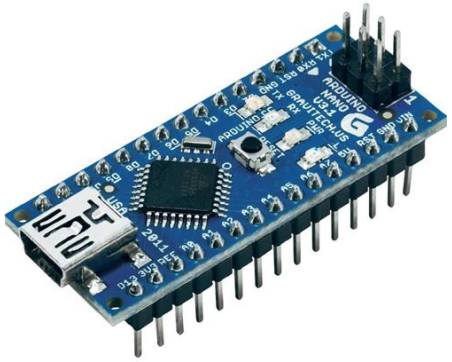
Supported Hardware



Raspberry
Pi Pico



Arduino Uno



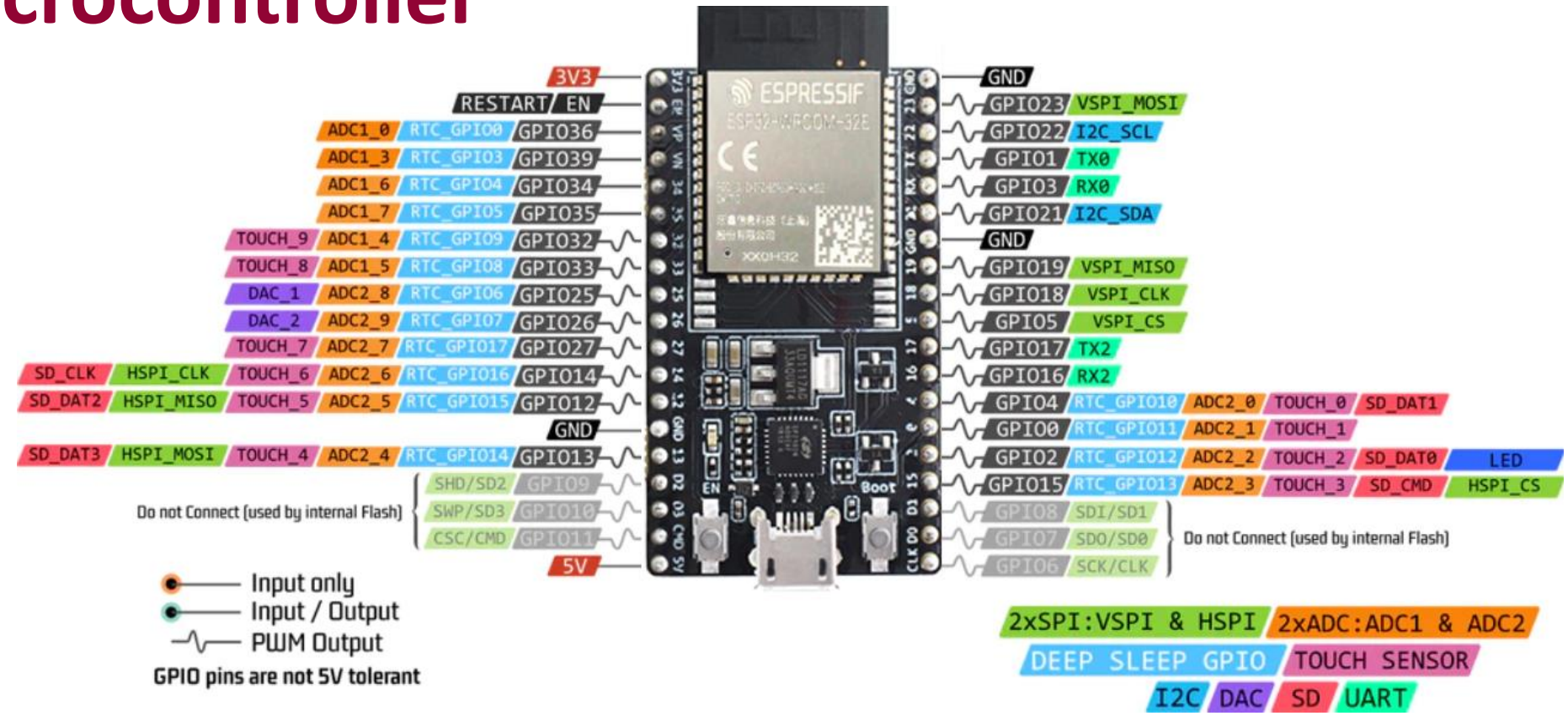
Arduino Nano

ESP32

Arduino Mega

ANTLAB

ESP32 microcontroller



- 4MB RAM, Dual core CPU
- Built-in Wi-Fi and Bluetooth
- Low cost, low energy
- Open Source

More here: https://randomnerdtutorials.com/esp32-pinout-reference-gpios/

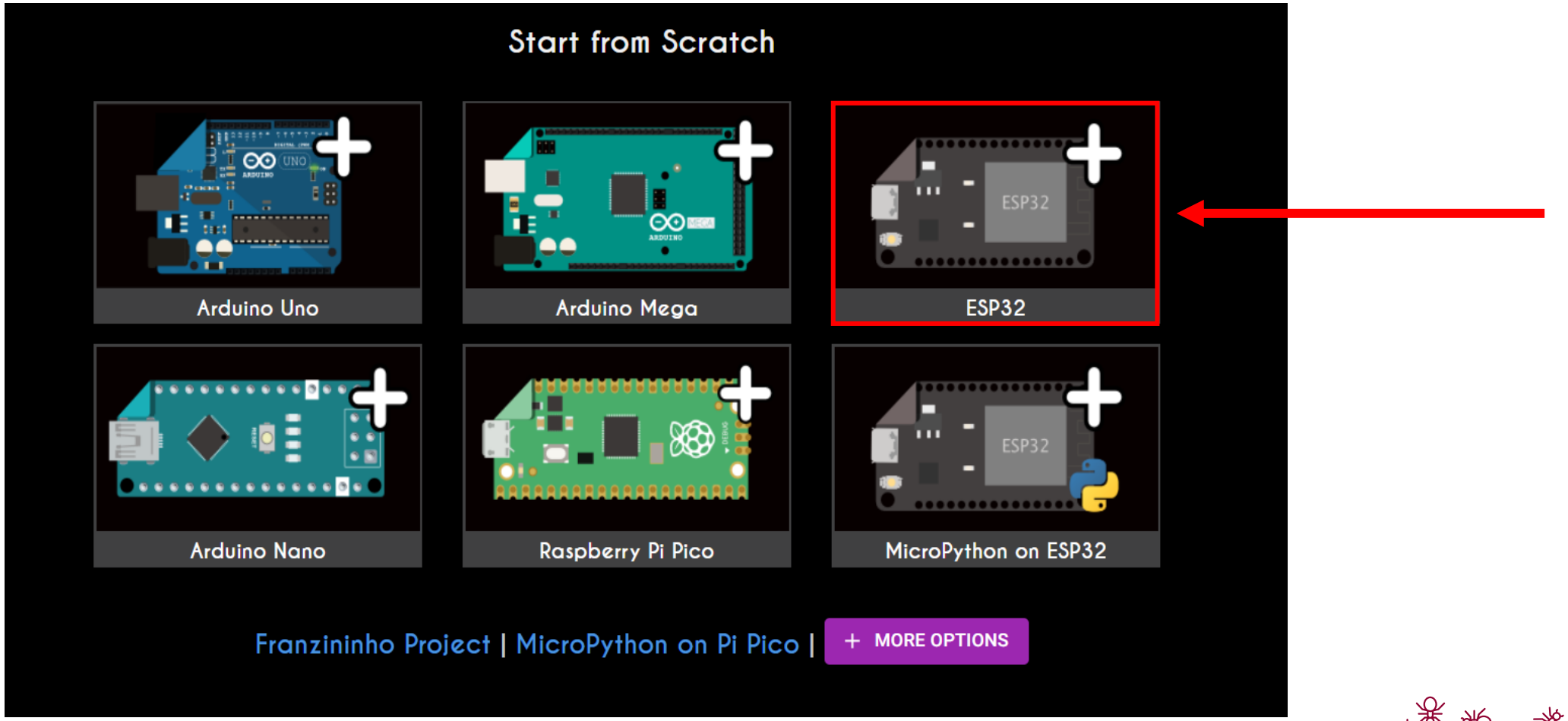
ANTLAB

POLITECNICO
MILANO 1863

Playing with ESP32

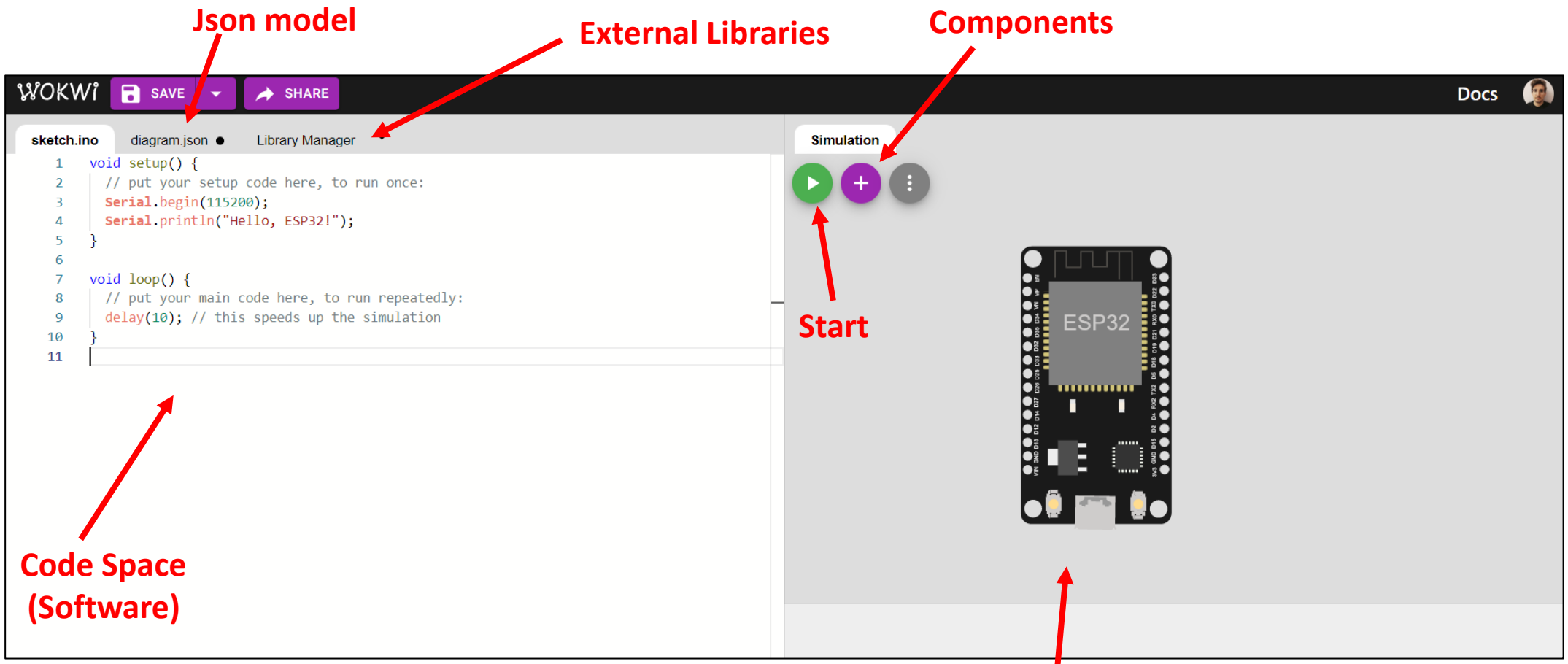
Start a new Project

Scroll down the homepage and start a new ESP32 project from scratch



ANTLAB

Start a new Project



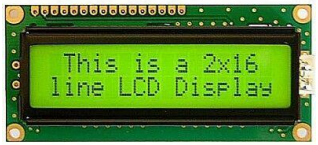
ANTLAB

Components

Resistors

Pushbutton

LCD Displays



Potentiometers

Leds

Temperature/Humidity sensor (DHT22)

Speakers/Buzzers

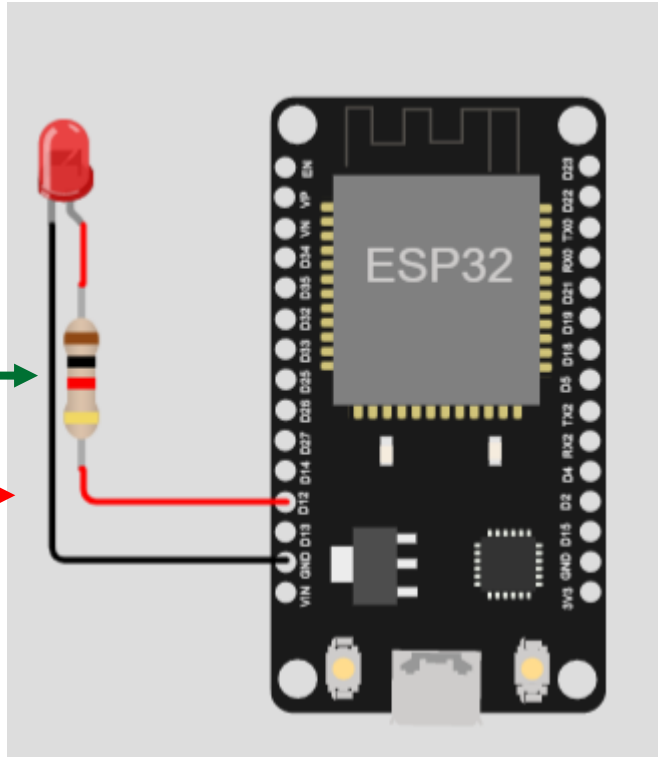
ANTLAB

First example: LEDs (1)

Part 1: Turn a LED on and off via software

```
1   #define LED  12  // The digital pin to which a led is connected.
2
3
4   void setup() {
5     //Pin Setup
6     pinMode(LED, OUTPUT);
7     //Serial Setup
8     Serial.begin(115200);
9
10  }
11  void loop() {
12    digitalWrite(LED, true);
13    delay(500);
14    digitalWrite(LED, false);
15    delay(500);
16
17  }
18
```

Set LED ON/OFF



Code

Design

ANTLAB

First example: LEDs (2)

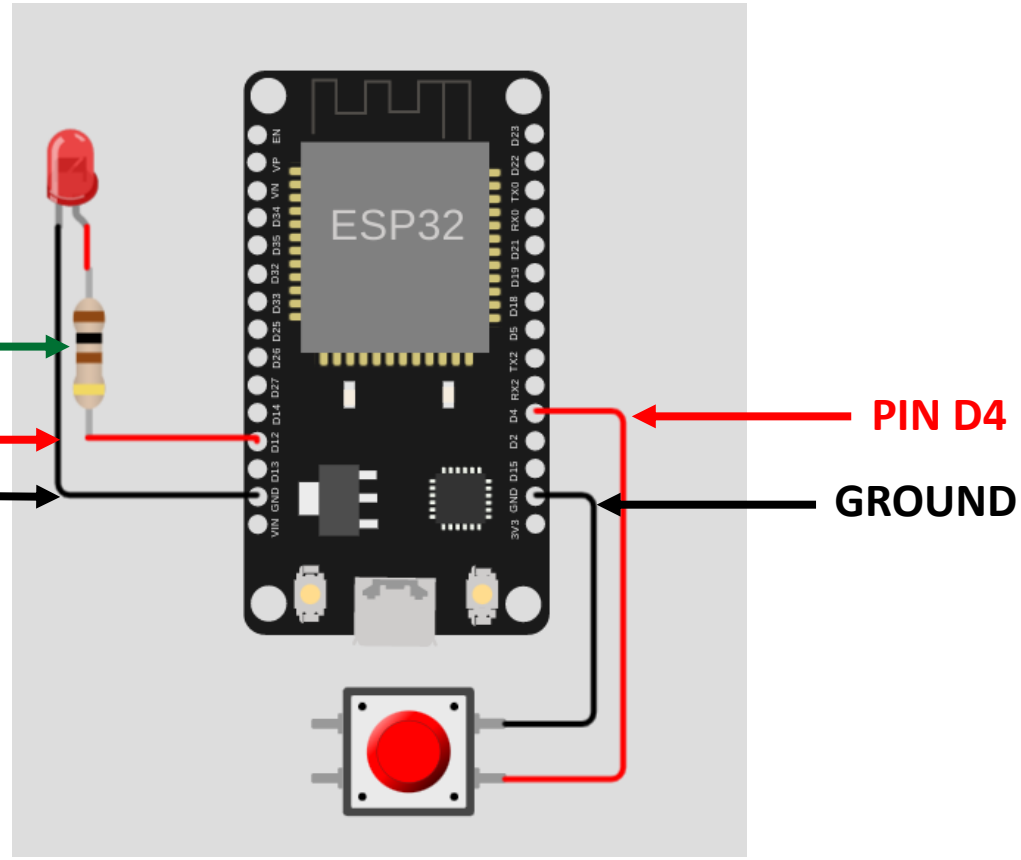
Part 2: Let's do it with a button now!

sketch.ino ● diagram.json Library Manager

```
1  #define LED  12  // The digital pin to which a led is connected.
2  #define BUTTON 4 // The digital pin to which the button is connected
3
4  bool led_value = 0; //Led status
5
6  void setup() {
7    //Pin Setup
8    pinMode(LED, OUTPUT);
9    pinMode(BUTTON, INPUT_PULLUP);
10   //Serial Setup
11   Serial.begin(115200);
12
13 }
14 void loop() {
15
16   Serial.println(digitalRead(BUTTON));
17   if (!digitalRead(BUTTON)){
18     led_value=!led_value;
19   }
20   delay(100);
21   digitalWrite(LED, led_value);
22   delay(100);
23 }
24
```

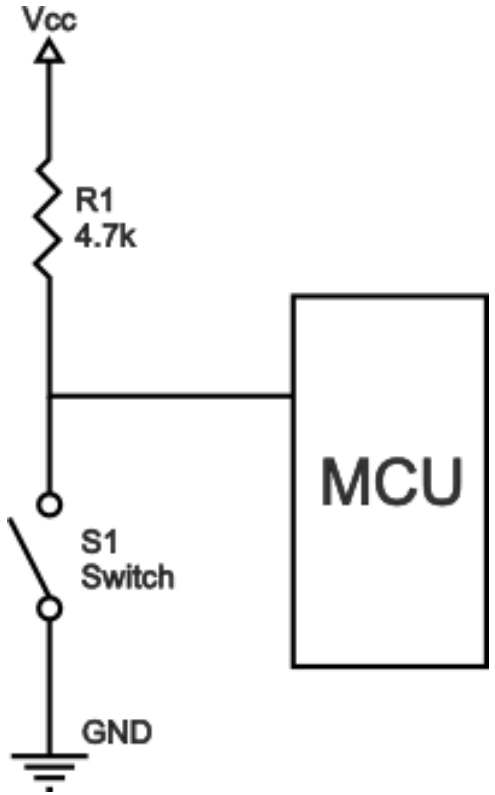
Check Button Status

Set LED ON/OFF

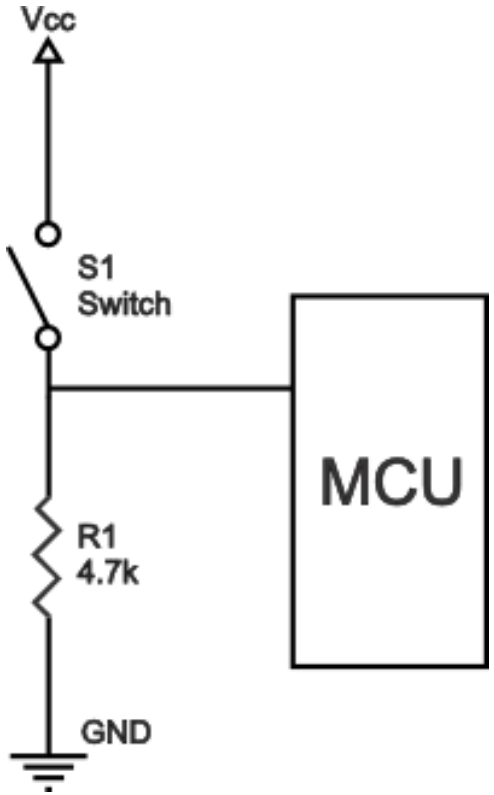


ANTLAB

PULLUP vs PULLDOWN



PULLUP RESISTOR

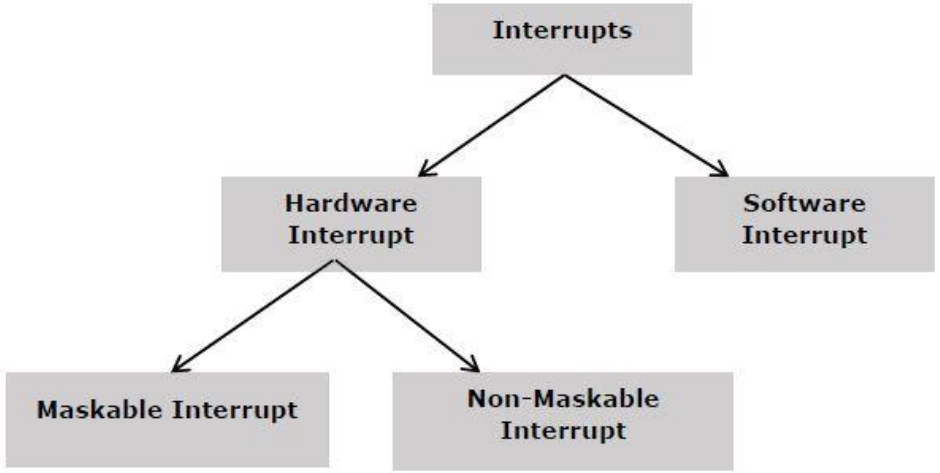


PULLDOWN RESISTOR

More here: https://eepower.com/resistor-guide/resistor-applications/pull-up-resistor-pull-down-resistor/#

ANTLAB

Interrupt and Interrupt Service Routine (ISR)



Interrupt is an event or signal that requests the CPU's attention.

I. The processor completes the current instruction, save it states and starts the implementation of an Interrupt Service Routine (ISR)

II. ISR is a program that tells the processor what to do when the interrupt occurs. After the ISR execution, control returns to the main routine where it was interrupted

ANTLAB

Interrupts in Action

Goal: Turn a LED on and off with button using Interrupts

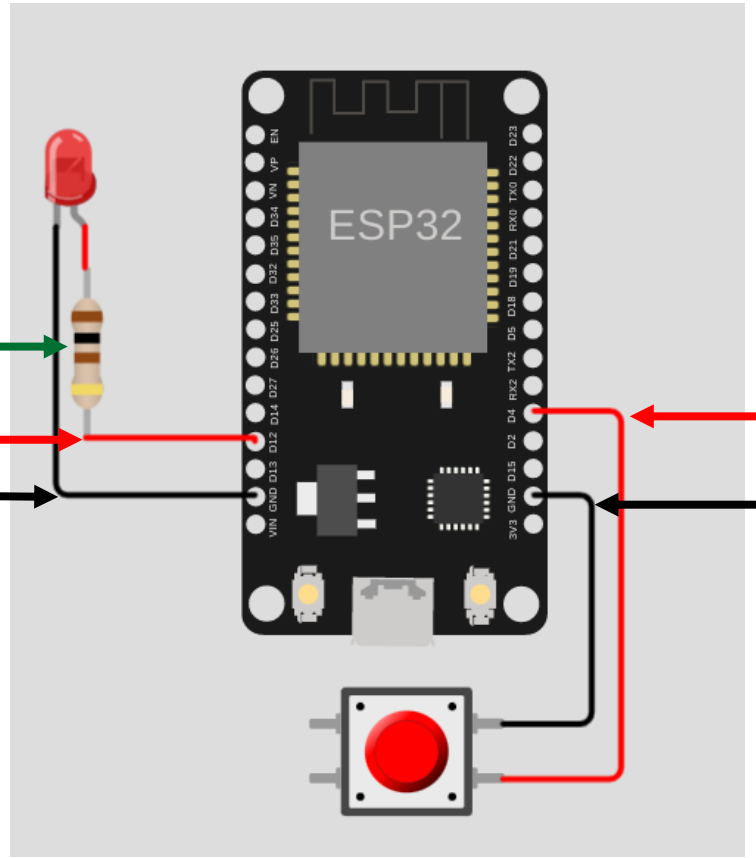
sketch.ino ● diagram.json ● Library Manager

```
1   #define LED 12
2   #define BTN 4
3
4   void IRAM_ATTR trigger_led(){
5     digitalWrite(LED, !digitalRead(LED));
6   }
7   void setup() {
8     // put your setup code here, to run once:
9     Serial.begin(115200);
10    Serial.println("Hello, ESP32!");
11    pinMode(LED, OUTPUT);
12    pinMode(BTN, INPUT);
13    attachInterrupt(BTN,trigger_led,RISING);
14  }
15
16  void loop() {
17    // put your main code here, to run repeatedly:
18    delay(10); // this speeds up the simulation
19  }
```

Interrupt Function

Attach Interrupt

Code



Design

ANTLAB

Interrupts in Action (with BOUNCE)

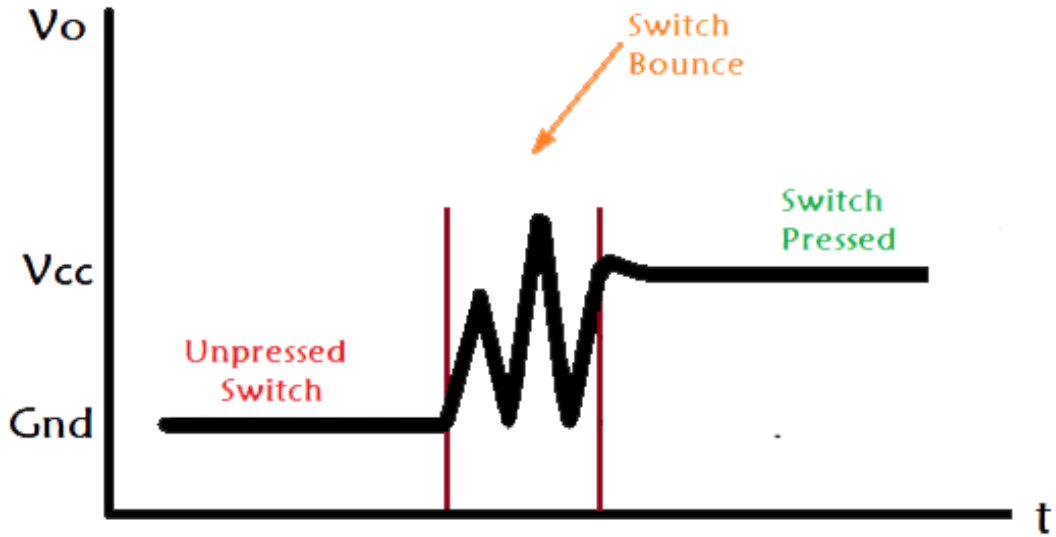
Goal: Turn a LED on and off with button using Interrupts

sketch.ino | diagram.json | Library Manager

```
1  #define LED  12  // The digital pin to which the led is connected.
2  #define BUTTON 4 // The digital pin to which the button is connected.
3
4  #define DEBOUNCE_DELAY 50 //ms to wait to solve the bouncing problem
5
6  //Volatile: Tells to the compiler that this variable
7  // may chage at any time (Used in ISR)
8  volatile int last_mills=0;
9
10 void IRAM_ATTR trigger_led(){
11   if (millis()-last_mills > DEBOUNCE_DELAY){
12     digitalWrite(LED, !digitalRead(LED));
13   }
14   last_mills=millis();
15 }
16
17 void setup() {
18   //Pin Setup
19   pinMode(LED, OUTPUT);
20   pinMode(BUTTON, INPUT);
21
22   //Interrupt Setup
23   attachInterrupt(BUTTON, trigger_led, RISING);
24 }
25 void loop() {
26 }
27
```

Interrupt Function

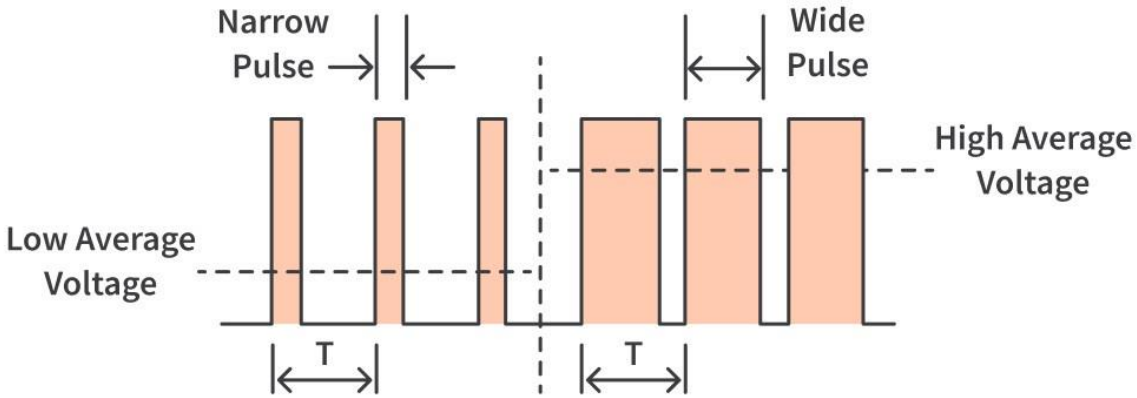
Attach Interrupt



BOUNCE EFFECT

ANTLAB

Pulse-Width Modulation (PWM)

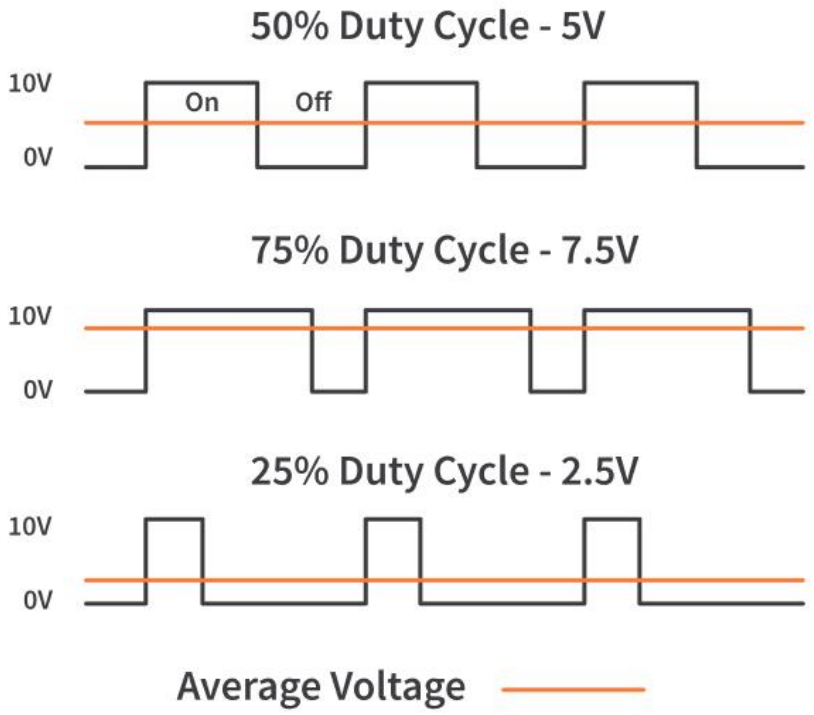


PWM is a technique to control analog devices, using a digital signal, to output an analog-like signal from a digital device

Duty cycle: *percentage of time a digital signal is "on" over an interval or period*

$$D = \frac{T_{on}}{Period} * 100$$

$$V_{avg} = \frac{D}{100} * V_{max}$$



PWM in Action (1)

Goal: Control LED brightness with PWM and a slider

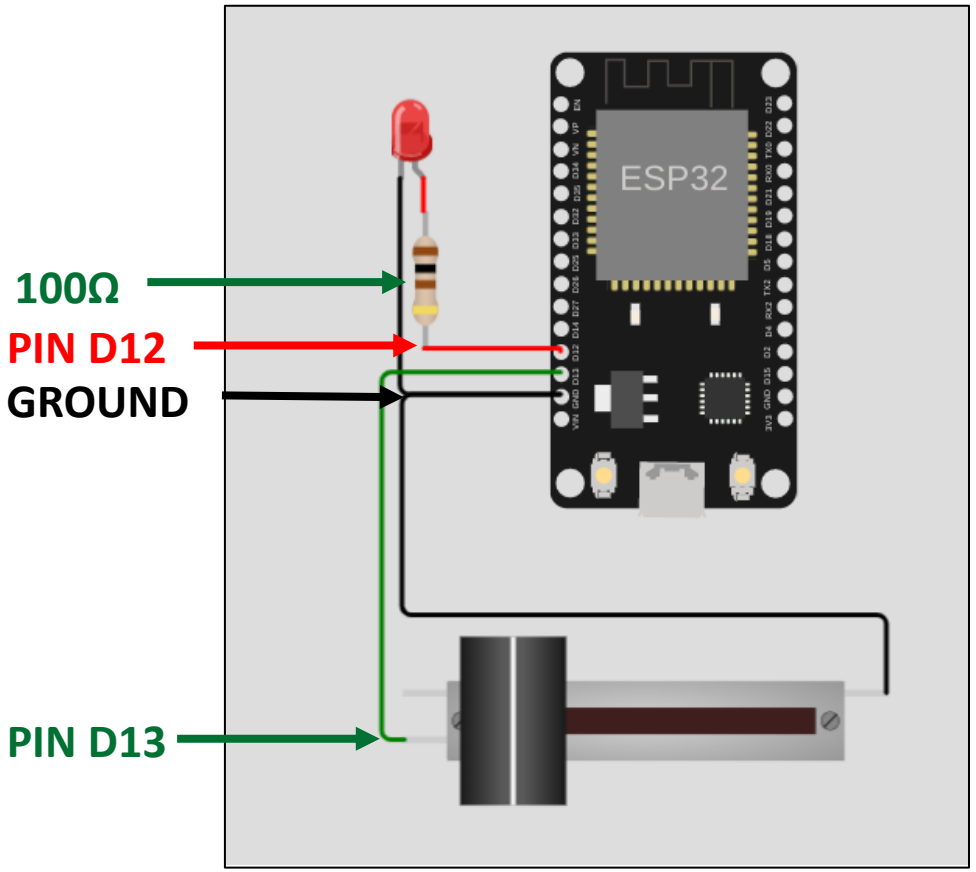
sketch.ino | diagram.json | Library Manager

```
#define LED   12  // LED Digital pin
#define SLIDE 13 // Potentiometer Digital pin
#define C_TIME  100 //PWM Period in us

volatile int pwm_value = 0; //Duty Cycle Time
hw_timer_t *My_timer = NULL;

void IRAM_ATTR onTimer(){
  bool led_status = digitalRead(LED);

  if (led_status && pwm_value!=C_TIME){
    digitalWrite(LED, !led_status);
    timerWrite(My_timer, pwm_value);

  }else if (!led_status && pwm_value!=0){
    digitalWrite(LED, !led_status);
    timerWrite(My_timer, C_TIME-pwm_value);
  }
}
```

Interrupt Function

Toggle LED

Set Timer Value



ANTLAB

PWM in Action (2)

Goal: Control LED brightness with PWM and a slider

sketch.ino | diagram.json | Library Manager

```
21  void setup() {
22  pinMode(LED, OUTPUT);
23  pinMode(SLIDE, INPUT);
24
25  My_timer = timerBegin(0, 80, true); //Timer initializer
26  //0: hw timer number (ESP32 has 3 hw timers available
27  //80: time divider. ESP32 clk 80MHz so we set evry tick to 1 us
28  //true: counter shoud increment
29
30  timerAttachInterrupt(My_timer, &onTimer, true);//Attach Interrupt
31
32  timerAlarmWrite(My_timer, C_TIME, true);
33  //C_TIME: number of microseconds after which the interrupt should occur
34  //true: timer counter will reload after interrupt
35
36  timerAlarmEnable(My_timer); //Just Enable
37  }
38  void loop() {
39    pwm_value = map(analogRead(SLIDE), 0, 4095, 0, C_TIME);
40  }
```

ANTLAB

POLITECNICO
MILANO 1863

ANTLAB

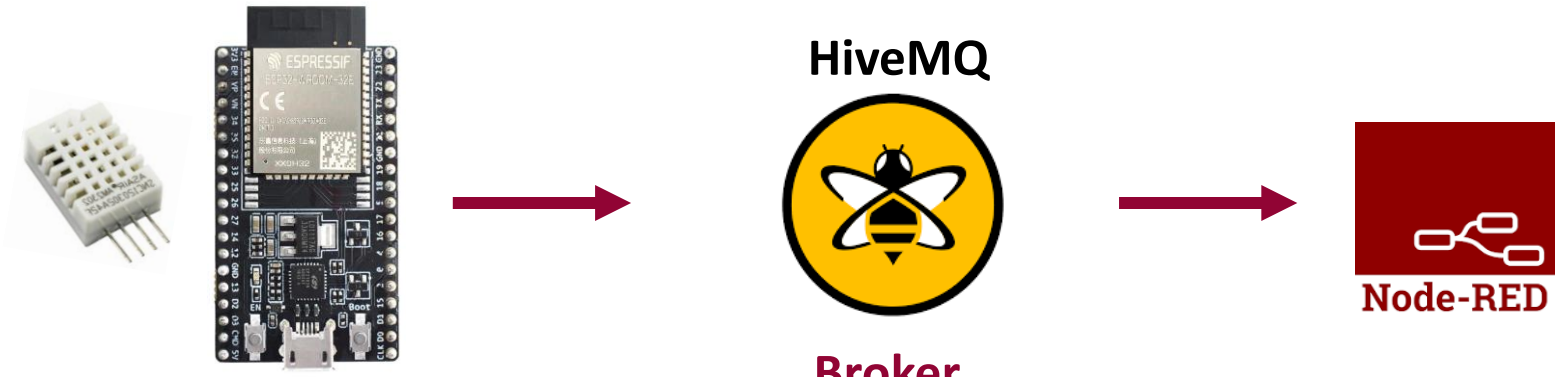
Playing with MQTT

Simple Pub/Sub Application

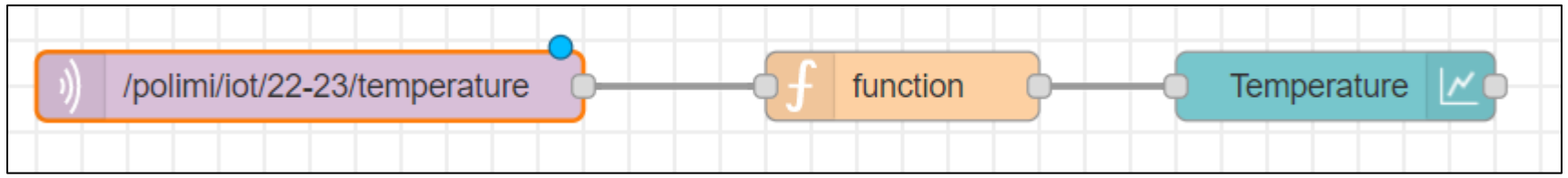
Goal: Read the temperature from a DHT22 sensor and update the status on a remote device using MQTT (device in node-red)

Two projects:

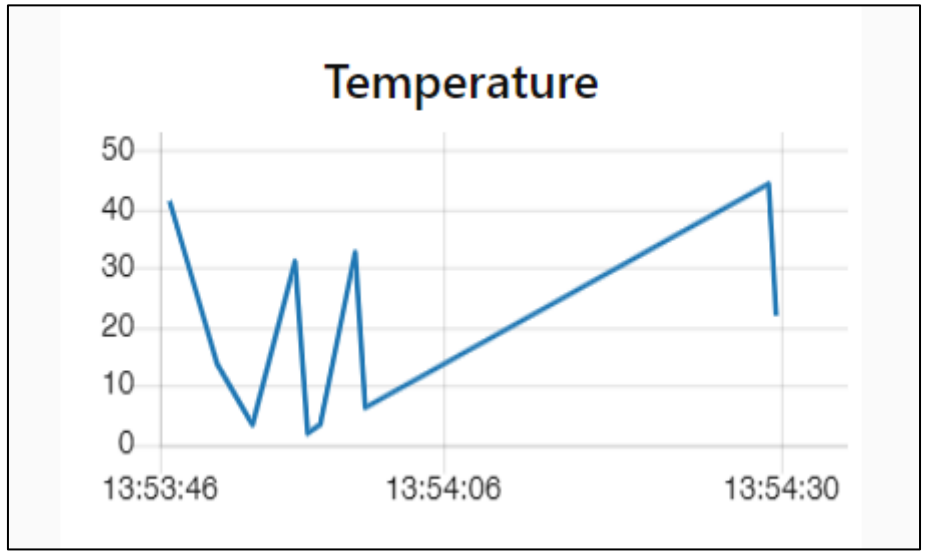
1. **Publisher**: The ESP32 reads the values from the DHT22 and publishes to the MQTT broker (**broker.hivemq.com**)
2. **Subscriber**: Node-Red node receiving the message and updating a plot



Simple Pub/Sub Application: Subscriber



Node-Red flow

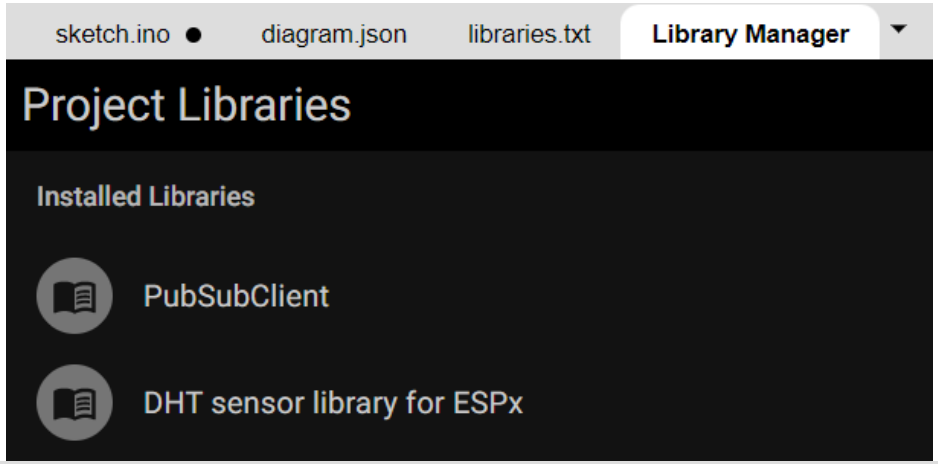


Node-Red chart

ANTLAB

Simple Pub/Sub Application: Publisher (1)

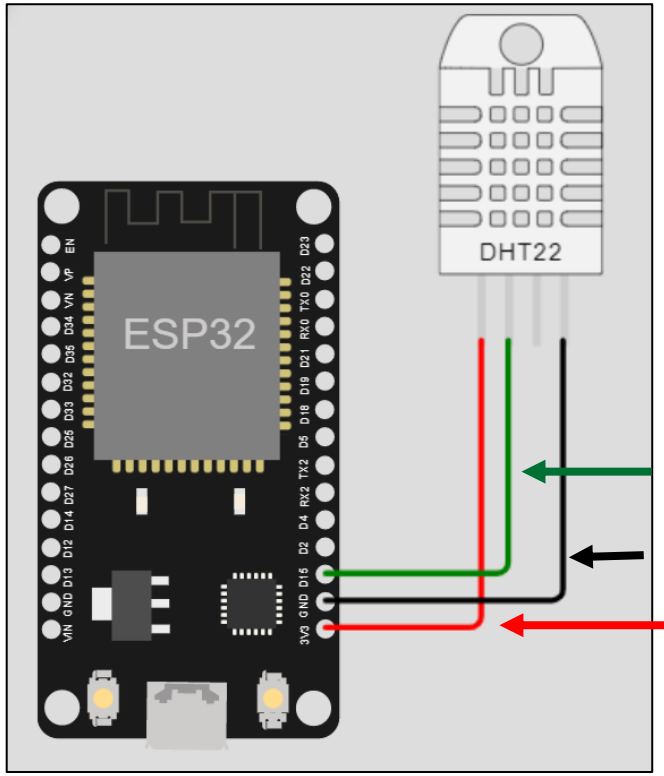
Add libraries first!



sketch.ino ● | diagram.json | libraries.txt | Library Manager

```
1   //Include needed libraries
2   #include <PubSubClient.h>
3   #include <WiFi.h>
4   #include "DHTesp.h"
5
6   #define PIN_DHT 15 //PIN USED FOR DATA IN DHT22 SENSR
7
8   #define MQTT_TOPIC "/polimi/iot/22-23/temperature" //the MQTT topic
9   #define MQTT_CLIENT_ID "iot-polimi-palmese-pub1" // the MQTT client identifier
10
11  char strTemperature[10]; //string where we store the temperature to publish
12
13  DHTesp dht;
14  WiFiClient espClient;
15  PubSubClient mqtt_client(espClient);
```

This is the MQTT client



ANTLAB

Simple Pub/Sub Application: Publisher (2)

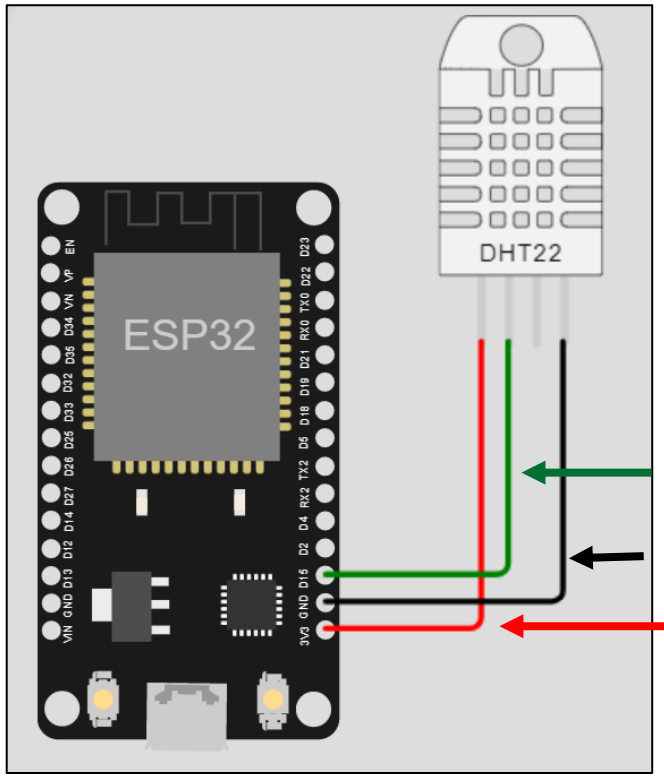
Setup

Main Loop

```
17  void setup() {
18    //init pins
19    dht.setup(PIN_DHT, DHTesp::DHT22);
20
21     // Setup Wi-Fi
22    WiFi.mode(WIFI_STA);
23    WiFi.begin("Wokwi-GUEST", "");
24
25    delay(1000);
26
27    // Setup MQTT
28    mqtt_client.setServer("test.mosquitto.org", 1883);
29    mqtt_client.connect(MQTT_CLIENT_ID);
30  }
31
32  void loop() {
33    TempAndHumidity data = dht.getTempAndHumidity();
34    sprintf(strTemperature, "%.2fC", data.temperature);
35
36    mqtt_client.publish(MQTT_TOPIC, strTemperature);
37
38    delay(1000);
39  }
```

Give time to connect!

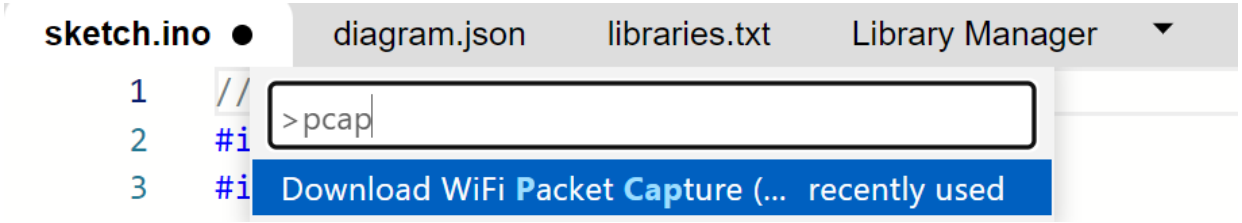
Publish!



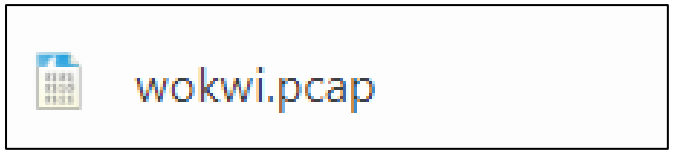
ANTLAB

Download the PCAP in WOKWI

- In the code field press F1, a menu will open
- Search for ‘PCAP’ and click on "Download Wi-Fi Packet Capture"



- A PCAP file will be downloaded, with all the packets from/to the ESP32 device



ANTLAB

More on Wokwi…

- PubSub Library API: https://pubsubclient.knolleary.net/
- Use featured project as guidelines
- Check for trending projects for nice ideas

Discord group

Facebook group

ANTLAB