**Sensor de pH**

Offset sketch

void setup() {

// initialize serial communication at 9600 bits per second:

Serial.begin(9600);

}

// the loop routine runs over and over showing the voltage on A0

void loop() {

// read the input on analog pin 0:

int sensorValue = analogRead(A0);

// Convert the analog reading (which goes from 0 - 1023) to a voltage (0 - 5V):

float voltage = sensorValue \* (5.0 / 1023.0);

// print out the value you read:

Serial.println(voltage);

delay(300);

}

PH measurement sketch

float calibration = 0.00; //change this value to calibrate

const int analogInPin = A0;

int sensorValue = 0;

unsigned long int avgValue;

float b;

int buf[10],temp;

void setup() {

Serial.begin(9600);

}

void loop() {

for(int i=0;i<10;i++)

{

buf[i]=analogRead(analogInPin);

delay(30);

}

for(int i=0;i<9;i++)

{

for(int j=i+1;j<10;j++)

{

if(buf[i]>buf[j])

{

temp=buf[i];

buf[i]=buf[j];

buf[j]=temp;

}

}

}

avgValue=0;

for(int i=2;i<8;i++)

avgValue+=buf[i];

float pHVol=(float)avgValue\*5.0/1024/6;

float phValue = -5.70 \* pHVol + calibration;

Serial.print("sensor = ");

Serial.println(phValue);

delay(500);

}

**Anemômetro**

// Pin definitions

# define Hall sensor 2 // Pino digital 2

// Constants definitions

const float pi = 3.14159265; // Numero pi

int period = 5000; // Tempo de medida(miliseconds)

int delaytime = 2000; // Time between samples (miliseconds)

int radius = 147; // Raio do anemometro(mm)

// Variable definitions

unsigned int Sample = 0; // Sample number

unsigned int counter = 0; // magnet counter for sensor

unsigned int RPM = 0; // Revolutions per minute

float speedwind = 0; // Wind speed (m/s)

float windspeed = 0; // Wind speed (km/h)

void setup()

{

// Set the pins

pinMode(2, INPUT);

digitalWrite(2, HIGH); //internall pull-up active

//Start serial

Serial.begin(9600); // sets the serial port to 9600 baud

}

void loop()

{

Sample++;

Serial.print(Sample);

Serial.print(": Start measurement...");

windvelocity();

Serial.println(" finished.");

Serial.print("Counter: ");

Serial.print(counter);

Serial.print("; RPM: ");

RPMcalc();

Serial.print(RPM);

Serial.print("; Wind speed: ");

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//print m/s

WindSpeed();

Serial.print(windspeed);

Serial.print(" [m/s] ");

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//print km/h

SpeedWind();

Serial.print(speedwind);

Serial.print(" [km/h] ");

Serial.println();

delay(delaytime); //delay between prints

}

// Measure wind speed

void windvelocity(){

speedwind = 0;

windspeed = 0;

counter = 0;

attachInterrupt(0, addcount, RISING);

unsigned long millis();

long startTime = millis();

while(millis() < startTime + period) {

}

}

void RPMcalc(){

RPM=((counter)\*60)/(period/1000); // Calculate revolutions per minute (RPM)

}

void WindSpeed(){

windspeed = ((4 \* pi \* radius \* RPM)/60) / 1000; // Calculate wind speed on m/s

}

void SpeedWind(){

speedwind = (((4 \* pi \* radius \* RPM)/60) / 1000)\*3.6; // Calculate wind speed on km/h

}

void addcount(){

counter++;

}

**Sensor umidade do solo**

int umidade;

void setup()

{

Serial.begin(9600);

Serial.println("www.usinainfo.com.br");

pinMode(13, OUTPUT);

}

void loop()

{

umidade = analogRead(A0);

int Porcento = map(umidade, 1023, 0, 0, 100);

Serial.print(Porcento);

Serial.println("%");

if(Porcento <=70)

{

Serial.println("Irrigando...");

digitalWrite(13, HIGH);

}

else

{

digitalWrite(13, LOW);

}

delay(1000);

}

**Sensor bme 280**

#define ESP32\_BOARD

#include <Wire.h>

#include <Arduino.h>

#include <WiFi.h>

#include <WiFiClient.h>

#ifdef ESP32\_BOARD

#include <BlynkSimpleEsp32.h>

#else

#include <BlynkSimpleEsp8266.h>

#endif

#include <string.h>

#include <Adafruit\_Sensor.h>

//#include <Adafruit\_BMP280.h>

#include <Adafruit\_BME280.h>

//Adafruit\_BMP280 bme;

Adafruit\_BME280 bme;

#define BLYNK\_PRINT Serial

#define BLYNK\_DEBUG 1

BlynkTimer timer;

float f\_temp = 0.0;

// You should get Auth Token in the Blynk App.

// Go to the Project Settings (nut icon).

char auth[] = "3ssEt0k3nEhF4k3";

//Defina o IP do seu servidor local, caso tenha optador por ele

#define SRV IPAddress(192,168,1,253)

// Your WiFi credentials.

// Set password to "" for open networks.

char ssid[] = "SSID\_DA\_SUA\_REDE";

char pass[] = "SUA\_SENHA";

void tempSend(){

String s\_temp = String(f\_temp);

Serial.println(s\_temp);

Blynk.virtualWrite(V1,f\_temp);

}

void setup()

{

// Debug console

Serial.begin(9600);

Serial.println("###################");

delay(5000);

//Esse BME280 está no endereço 0x76

if (!bme.begin(0x76)){

Serial.println("BMP280 isn't started. locking here.");

while (true);

}

Serial.println("Starting.");

Blynk.begin(auth, ssid, pass, SRV, 8080); //para servidor local, esse é o formato

timer.setInterval(2000L, tempSend);

delay(1000);

}

void loop()

{

f\_temp = bme.readTemperature();

Blynk.run();

timer.run();

delay(100);

}

**Pluviômetro**

#include <LiquidCrystal.h> //Inclui a biblioteca do LCD

LiquidCrystal lcd(12, 11, 5, 4, 3, 2); //Configura os pinos do Arduino para se comunicar com o LCD

// Constantes:

const int REED = 6; //The reed switch outputs to digital pin 9

// Variáveis:

int val = 0; //Current value of reed switch

int old\_val = 0; //Old value of reed switch

int REEDCOUNT = 0; //This is the variable that hold the count of switching

// as variaveis a seguir sao do tipo long pois o tempo, medido em milissegundos,

// rapidamente se tornara em um numero grande demais para ser armazenado em uma int.

void setup() {

// initializa o pino do switch como entrada

pinMode (REED, INPUT\_PULLUP); //This activates the internal pull up resistor

// initializa a comunicaçao serial:

Serial.begin(9600);

digitalWrite(2,HIGH);

lcd.begin(16, 2); //Inicia o LCD com dimensões 16x2(Colunas x Linhas)

lcd.setCursor(0, 0); //Posiciona o cursor na primeira coluna(0) e na primeira linha(0) do LCD

lcd.print("Pulsos "); //Escreve no LCD "Olá Garagista!"

lcd.setCursor(0, 1); //Posiciona o cursor na primeira coluna(0) e na segunda linha(1) do LCD

lcd.print("Chuva "); //Escreve no LCD "LabdeGaragem"

}

void loop() {

// ler o estado do switch pelo pino de entrada:

val = digitalRead(REED); //Read the status of the Reed swtich

if ((val == LOW) && (old\_val == HIGH)){ //Check to see if the status has changed

delay(10); // Delay put in to deal with any "bouncing" in the switch.

REEDCOUNT = REEDCOUNT + 1; //Add 1 to the count of bucket tips

old\_val = val; //Make the old value equal to the current value

Serial.print("Medida de chuva (contagem): ");

Serial.print(REEDCOUNT);//\*0.2794);

Serial.println(" pulso");

Serial.print("Medida de chuva (calculado): ");

Serial.print(REEDCOUNT\*0.25);

Serial.println(" mm");

lcd.setCursor(8, 0); //Posiciona o cursor na décima quarta coluna(13) e na segunda linha(1) do LCD

lcd.print(REEDCOUNT); //Escreve o valor atual da variável de contagem no LCD

lcd.setCursor(8, 1); //Posiciona o cursor na décima quarta coluna(13) e na segunda linha(1) do LCD

lcd.print(REEDCOUNT\*0.25); //Escreve o valor atual da variável de contagem no LCD

lcd.setCursor(14, 1); //Posiciona o cursor na primeira coluna(0) e na segunda linha(1) do LCD

lcd.print("mm"); //Escreve no LCD "LabdeGaragem"

}

else {

old\_val = val; //If the status hasn't changed then do nothing

}

}

**Módulo GSM**

//Programa: Arduino GSM Shield SIM900 - Acesso internet

//Autor: Arduino e Cia

//#include "SIM808.h"

#include <SoftwareSerial.h>

#include "inetGSM.h"

InetGSM inet;

boolean started = **false**;

**char** smsbuffer[160];

**char** n[20];

byte valor;

**void** setup()

{

Serial.begin(9600);

powerUpOrDown();

Serial.println(F("Testando GSM Shield SIM900"));

**if** (gsm.begin(2400))

{

Serial.println(F("nstatus=READY"));

started = **true**;

}

**else** Serial.println(F("nstatus=IDLE"));

}

**void** loop()

{

**if** (started) {

//Aguarda novo SMS e envia para o servidor web

**if** (gsm.readSMS(smsbuffer, 160, n, 20)) {

String str(smsbuffer);

envia\_GSM(smsbuffer);

delay(10000);

}

delay(1000);

}

}

**void** powerUpOrDown()

{

//Liga o GSM Shield

Serial.print(F("Liga GSM..."));

pinMode(6, OUTPUT);

digitalWrite(6, LOW);

delay(1000);

digitalWrite(6, HIGH);

delay(1000);

Serial.println(F("OK!"));

digitalWrite(6, LOW);

delay(500);

}

**void** envia\_GSM(String texto)

{

**char** temp\_string[55];

**char** msg[10];

**int** numdata;

**if** (inet.attachGPRS("timbrasil.br", "tim", "tim"))

Serial.println(F("status=Conectado..."));

**else** Serial.println(F("status=Nao conectado !!"));

delay(100);

String valor = "MSG\_Texto1=" + texto;

valor.toCharArray(temp\_string, 55);

numdata = inet.httpPOST("arduinoecia.webatu.com", 80, "/add.php", temp\_string, msg, 50);

delay(5000);

}