**KOD za 1. arduino**

/\* Pomoću Blynk aplikacije omogućavamo komunikaciju između mobitela i mkr10000 arduino pločice – rasvjeta, parkirališna ograda i park \*/

#define BLYNK\_PRINT Serial

#include <SPI.h>

#include <WiFi101.h> //library za wifi na mkr1000

#include <BlynkSimpleMKR1000.h>

#include <Servo.h> //library za servo motor koji otvara ogradu

// Blynk token koji prepoznaje mobilni uređaj

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

char auth[] = "WQSNl1e5UgF6c9ExKl9EWS8VZBULOwvl";

// WiFi postavke

char ssid[] = "MW40V\_4E0C";

char pass[] = "04040295";

Servo servo;

BLYNK\_WRITE(V3)

{

 servo.write(param.asInt());

}

void setup()

{

 // otvaranje serijske komunikacije

 Serial.begin(9600);

 //pokretanje Blynk aplikacije

 Blynk.begin(auth, ssid, pass);

 /\* Ako se točno definira server, služimo se ovim postavkama: Blynk.begin(auth, ssid, pass, "blynk-cloud.com", 80); Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,100), 8080);\*/

 servo.attach(9); //servo motor se kontrolira sa virtualnog pina 9

}

void loop()

//u loop petlji koristimo naredbu za rad blynk aplikacije

{

 Blynk.run();

}

// Rasvjeta u podrumu se kontrolira pomoću digitalne sklopke D2 (digital pin 2), a rasvjeta u građevini sa D3 pina, rasvjeta za park (drveće)se kontrolira sa analognih pinova A1, A2, A3, A5

// Otvaranje ograde kad dolazi vozilo kontrolira se pomoću virtualnog pina za servo motor kojem smo dodali posebno napajanje kako bi ispravno radio.

**KOD na arduino 2**

#include <DHT11.h> //knjižnica za DHT senzor (mjerenje vlage i temperature u podrumu - realni -realtime uvjeti)

#include <Wire.h>

#include <LiquidCrystal\_I2C.h> // library za LCD ekran

#include "dht.h"

#include <SPI.h>

LiquidCrystal\_I2C lcd(0x27,2,1,0,4,5,6,7,3, POSITIVE); // LCD display je na adresi sabirnice 0x27

int pin = A0; //integer analogni pin A0

DHT11 dht11(pin); //DHT senzor je signalnim pinom priključen na arduino pin 11, a napajanje senzora na naponske pinove 11

// definicija stupnjeva temperature u C

double Fahrenheit(double celsius) {

return ((double)(9 / 5) \* celsius) + 32;

}

double Kelvin(double celsius) {

return celsius + 273.15;

}

void setup() {

 Serial.begin(9600); //serijska komunikacija

 Serial.println("LCD 16,2, DHT 11 spremni"); //Na serijskome pinu napiši da je oprema spremna

 lcd.begin(16, 2); //pokreni LCD 16 m. mjesta, 2 reda

lcd.backlight(); //pozadinsko svjetlo LCD-a

lcd.clear(); //očisti zaslon

lcd.setCursor(3, 0); //postavi kursor i pripremi tekst

lcd.print("Uvjeti u");

lcd.setCursor(4, 1);

lcd.print("podrumu");

delay(4000);

lcd.clear();

lcd.print("Parametri;");

lcd.setCursor(0,1);

lcd.print("Vlaga % & temp C");

delay(5000);

lcd.clear();

}

void loop() {

 //očitaj uvjete sa DHT senzora

int err;

float temp, humi;

if ((err = dht11.read(humi, temp)) == 0)

{

lcd.clear();

delay(500);

lcd.print("Mjerenje.....");

delay(3000);

lcd.clear(); //očisti zaslon

// ispiši uvjete koji su provjereni na analognim pinovima u loop petlji

lcd.setCursor(0, 0);

lcd.print("Temp:");

lcd.setCursor(1, 1);

lcd.print("RH:");

lcd.setCursor(9, 0);

lcd.print(temp);

lcd.print(" C");

lcd.setCursor(9, 1);

lcd.print(humi);

lcd.print(" %");

delay(6000);

lcd.clear();

lcd.setCursor(3, 0);

lcd.print("Uvjeti su");

lcd.setCursor(4, 1);

lcd.print("u redu!");

delay(3000);

}

//Ako nešto ne radi dobro - odnosno DHT senzor je odspojen napiši na LCD zaslon error!

else

{

lcd.println();

lcd.print("Error No :");

lcd.print(err);

lcd.println();

}

}

**KOD za arduino 3**

//Ovaj kod mjeri i zapisuje real-time podatke na OLED ekranu u vremenskoj stanici koju smo isprintali na 3D printeru i postavili je u vinograd

//BMP senzor se nalazi u mjernoj stanici iza OLED zaslona i daje podatke o atmosferskom tlaku, temperaturi i nadmorskoj visini na kojoj se nalazi naša klet

#include <Wire.h>

#include <Adafruit\_GFX.h> //uključi napisan library

#include <Adafruit\_SSD1306.h> // uključi SSD1306 library za OLED zaslon mjerne stanice

#include <Adafruit\_Sensor.h>

#include <Adafruit\_BMP085\_U.h> // knjižnica za BMP senzor

Adafruit\_BMP085\_Unified bmp = Adafruit\_BMP085\_Unified(10085);

// definirane dimenzije OLED zaslona prema kodovima iz knjižnice

#define SCREEN\_WIDTH 128 // OLED display width, in pixels

#define SCREEN\_HEIGHT 64 // OLED display height, in pixels

#define SCREEN\_VERTICAL\_FLIP true

#define SSD1306\_ACTIVATE\_SCROLL 0x26 ///< Start scroll

// OrganicLED spojen naI2C sabirnicu, sa istim easy C kablom je povezan i BMP senzor

Adafruit\_SSD1306 oled(SCREEN\_WIDTH, SCREEN\_HEIGHT, &Wire, -1);

void displaySensorDetails(void)

{

 sensor\_t sensor;

 bmp.getSensor(&sensor);

 oled.println("------------------------------------");

 Serial.print ("Sensor: "); Serial.println(sensor.name);

 Serial.print ("Driver Ver: "); Serial.println(sensor.version);

 Serial.print ("Unique ID: "); Serial.println(sensor.sensor\_id);

 Serial.print ("Max Value: "); Serial.print(sensor.max\_value); oled.println(" hPa");

 Serial.print ("Min Value: "); Serial.print(sensor.min\_value); oled.println(" hPa");

 Serial.print ("Resolution: "); Serial.print(sensor.resolution); oled.println(" hPa");

 oled.println("------------------------------------");

 oled.println("");

 delay(500);

}

void setup() {

 Serial.begin(9600); //serijska komunikacija

 // pokreni OLED sa ovom adresom: 0x3C for 128x64

 if (!oled.begin(SSD1306\_SWITCHCAPVCC, 0x3C)) {

 Serial.println(F("OLED zaslon"));

 while (true);

 }

 {if(!bmp.begin())

 {

 /\* Provjerava da li je BMP senzor dobro spojen \*/

 oled.println("Provjeri BMP185 senzor");

 while(1);

 }

 /\* prikazuje osnovne informacije o senzoru \*/

 displaySensorDetails();

}

}

void loop(void) //m. mjesto iz void setup

{

 delay(500); // pričekaj za pokretanje

 oled.clearDisplay(); // obriši zaslon

 oled.setTextSize(1); // ova naredba definira veličinu slova na OLED ekranu

 oled.setTextColor(WHITE); // boja teksta na OLED ekranu

 oled.setCursor(0, 10); // pozicija na kojoj se počinje pojavljivati tekst na OrganicLED ekranu

 /\* pokreni senzor event \*/

 sensors\_event\_t event;

 bmp.getEvent(&event);

 /\* prikaži rezultate za barometarski tlak u hPa\*/

 if (event.pressure)

 {

 /\* prikaži atmosferski tlak u hPa \*/

 oled.println("------ Atm. tlak: hPa"); //ispiši te rezultate na OLED

 oled.println(event.pressure);

 //oled.println(" hPa"); - pričuvni dio koda za scrollanje

 /\* Očitanje temperature sa BMP senzora i prikaz na OranicLed-u \*/

 float temperature; //naredba float označava da se radi o decimalnoj vrijednosti

 bmp.getTemperature(&temperature);

 oled.println("------ Temperatura: C"); //prikaži rezultat na OLED-u

 oled.println(temperature);

 //oled.println(" C");

 //pretvaranje atmosferskog tlaka u metre NM visine

 float seaLevelPressure = SENSORS\_PRESSURE\_SEALEVELHPA; //također decimalna vrijednost

 oled.println("------ NMV u metrima"); //napiši NMV u metrima na OLED ispod vrijednosti za temperaturu

 oled.println(bmp.pressureToAltitude(seaLevelPressure,

 event.pressure));

 //oled.println(" m");

 oled.println("");

 }

 else

 {

 //kada smo provjeravali ispravnost koda na računalu otvorili smo serial monitor da provjerimo da li ispisuje greške prilikom odspajanja senzora

 Serial.println("Sensor error"); //ako je senzor odspojen i uvjeti u loop petlji nisu zadovoljeni - else se pokreće i ispisuje nam grešku na serial monitor

 }

 delay(100); //uvjete osvježavamo svakih 100 ms kako bi OLED bio atraktivniji, povećavanjem ove vrijednosti dobiva se na stabilnosti rezultata

 oled.display(); // prikaži na OLED ekranu!

 }