Univerza v Ljubljani
Fakulteta za strojništvo

Mehatronika: GEMS Amethyst razvojna plošča

Kristjan Danov, Jan Černe, Milena Gigova, Marija Gečeva

Skupina: 2B

Maj 2025

Kazalo

Poglavje 1

Uvod

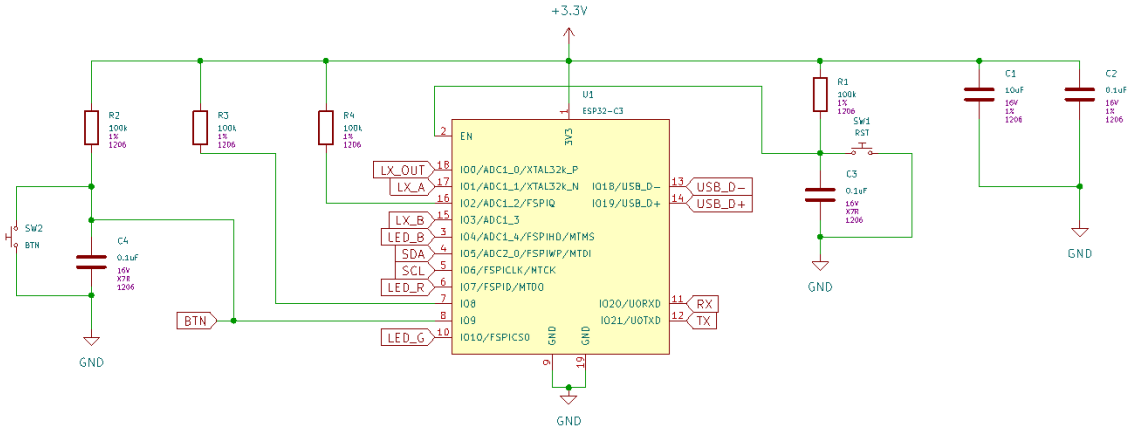
Razvojna plošča Erasmus+ GEMS Amethyst je kompaktna, dvoslojna tiskanina dimenzij 40×40 mm, zasnovana za uvajanje v svet mikrokrmilniških sistemov. Središče vezja predstavlja mikrokrmilnik ESP32-C3-WROOM-02, ki temelji na energetsko učinkoviti RISC-V arhitekturi in vključuje brezžični povezavi Wi-Fi ter Bluetooth Low Energy, kar omogoča razvoj tako lokalnih kot tudi IoT aplikacij.

Poglavje 2

Opis razvojne plošče

2.1 Zgradba in sistemski pregled

Razvojna plošča temelji na mikrokrmilniku ESP32-C3, ki podpira komunikacijo prek Wi-Fi-ja in BLE-ja. Glavne funkcionalnosti so prikazane na blokovni shemi (Slika 2.1). Napajanje in komunikacija z računalnikom potekata prek USB-C priključka (MOLEX 216990-0001), skladnega z evropskimi standardi. Integrirani napetostni regulator skrbi za zanesljivo pretvorbo vhodnih 5 V na delovnih 3,3 V. Zasnova vključuje osnovne funkcionalne enote, kot so RGB LED, tipka in svetlobni senzor. Te enote omogočajo osnovne uporabniške interakcije in senzoriko. Za stabilno delovanje skrbi več kondenzatorjev, ki blažijo napetostne spremembe in filtrirajo visokofrekvenčne motnje. Določeni upori delujejo kot pull-up elementi in zagotavljajo pravilna začetna logična stanja vhodov. Prehodni elementi (kot na primer stikala) so dodatno mehčani z RC kombinacijami, kar zmanjša pojav odbojev in zagotavlja zanesljiv zajem signala. Plošča je odprtokodna in modularna. Omogoča enostavno nadgradnjo z zunanjimi komponentami prek dostopnih GPIO pinov. Zaradi podpore v Arduino okolju in široke skupnosti uporabnikov je GEMS Amethyst primerna tako za začetnike, kot za naprednejše razvojne projekte.

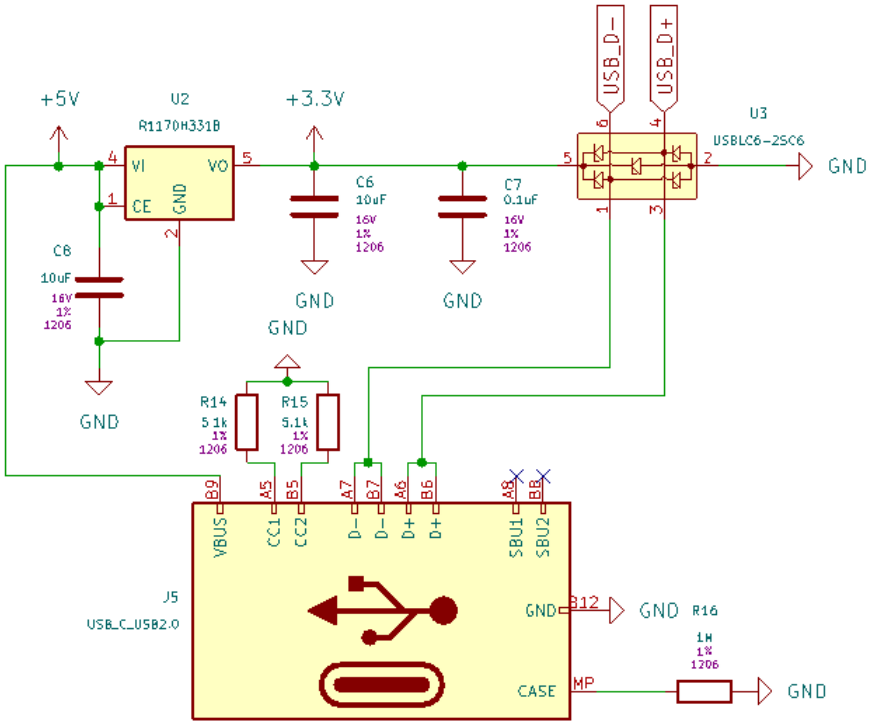


Slika 2.1: Blokovna shema razvojne plošče

2.2 Sestavni elementi

– Mikrokrmilnik: ESP32-C3-WROOM-02
Povezava do podatkovnega lista: https://eu.mouser.com/datasheet/2/891/Espressif_Systems_04082021_ESP32_C3_WROOM_02-2295851.pdf
Mikrokrmilnik predstavlja osrednji element razvojne plošče, saj upravlja delovanje vseh ostalih komponent in izvaja obdelavo podatkov. Gre za zmogljivo enoto s široko možnostjo uporabe – od osnovnega nadzora senzorjev do naprednejših aplikacij. Poleg klasičnih funkcij je podprta tudi brezžična komunikacija prek tehnologij Wi-Fi in BLE.

– USB-C priključek – Molex216990-001
Povezava do podatkovnega lista: https://www.molex.com/content/dam/molex/molex-dot-com/products/automated/en-us/salesdrawingpdf/216/216990/2169900001_sd.pdf
Omogoča enostavno povezavo z računalnikom za napajanje in prenos podatkov.



Slika 2.2: Blokovna shema USB-C priključka, ESD-ja in regulatorja napetosti

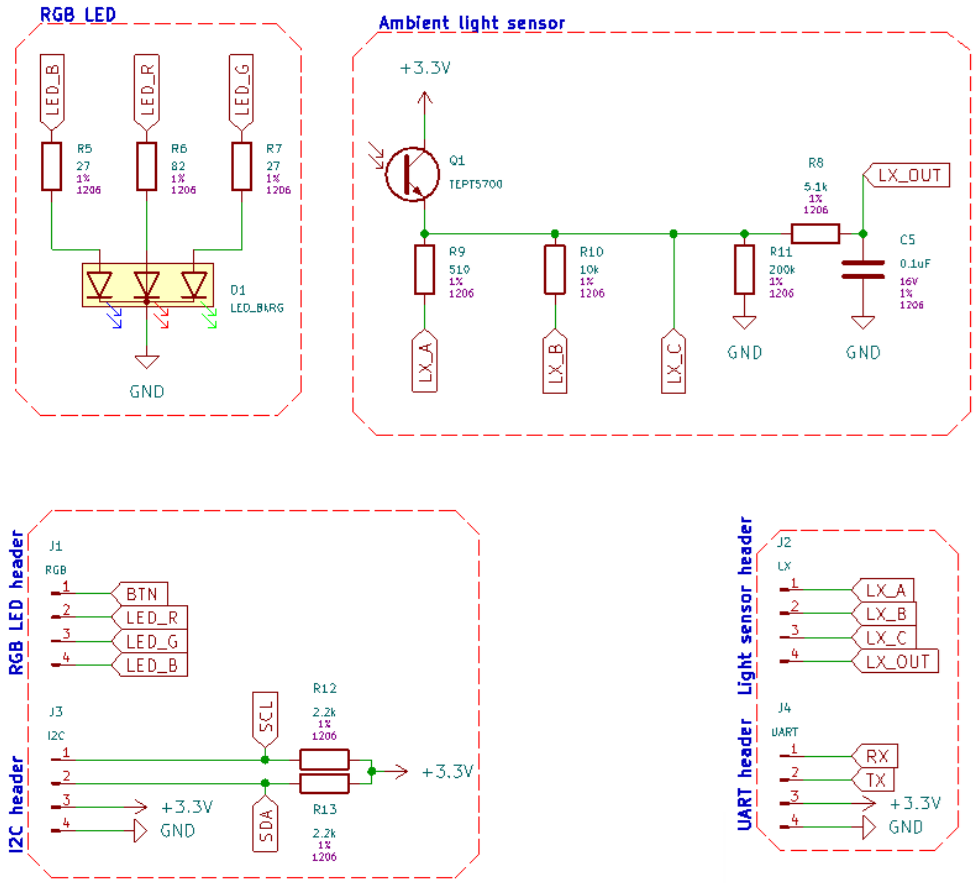
– ESD(Electrostatic discharge) protection - USBLC6-2SC6
Povezava do podatkovnega lista: https://eu.mouser.com/datasheet/2/389/usblc6_2-1852789.pdf
Gre za element, ki ščiti vezje pred škodljivimi vplivi elektrostatičnih razelektritev. Zaradi zelo nizke lastne kapacitivnosti ne vpliva na delovanje hitrih podatkovnih linij. Hkrati učinkovito omejuje tokovne sunke, ki bi lahko poškodovali občutljive komponente.

- LDO(Low dropout regulator) - R1170H331B-T1-FE
  Povezava do podatkovnega lista: https://eu.mouser.com/datasheet/2/294/r1170_ea-3219814.pdf
  Nizko-izgubni linearni regulator napetosti skrbi za pretvorbo vhodnih 5 V v stabilno delovno napetost 3,3 V. Izdelan je v CMOS tehnologiji, kar zagotavlja visoko energijsko učinkovitost, saj porabi energijo predvsem med preklopi stikalnih tranzistorjev.

- RGB LED - HV-5RGB60
  Povezava do podatkovnega lista: https://eu.mouser.com/datasheet/2/180/HV_5RGBXX_5mm_Full_Color_Series-1489147.pdf
  Večbarvna LED dioda, z ločenimi rdečimi, zelenimi in modrimi elementi, omogoča generiranje različnih barv s pomočjo ustreznega krmiljenja PWM signalov. Standardni tok skozi diodo znaša 25 mA, maksimalni dovoljen impulz pa doseže do 100 mA za zelo kratek čas.

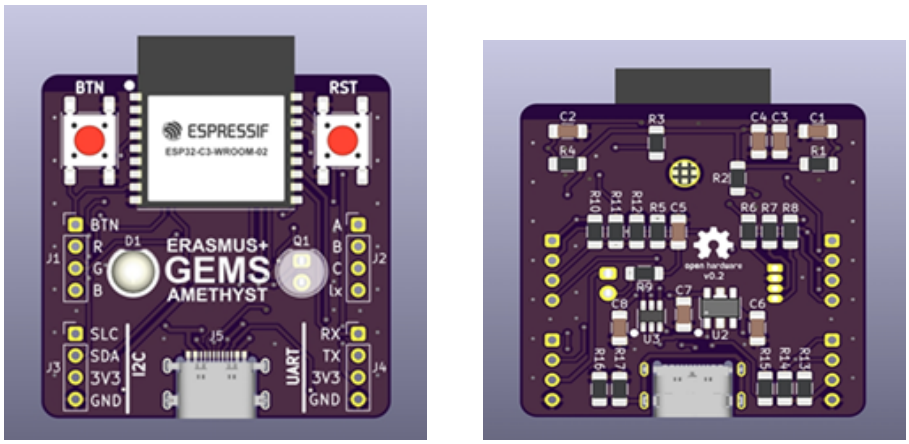
- Tipka - PTS645SM50SMTR92 LFS
  Povezava do podatkovnega lista: https://www.ckswitches.com/media/1471/pts645.pdf
  Gre za SMD momentno stikalo, namenjeno ponastavitvi delovanja mikrokrmilnika med razvojem ali testiranjem programa. Zagotavlja mehansko zanesljivost do 100.000 pritiskov, deluje do toka 50 mA pri napetosti 12 V.

- Senzor svetlobe - TEPT5700
  Povezava do podatkovnega lista: https://www.vishay.com/docs/81321/tept5700.pdf
  To je NPN fototranzistor, občutljiv na svetlobo v spektru vidne svetlobe. Z naraščanjem jakosti svetlobe se povečuje tok skozi senzor, kar omogoča zaznavo ambientne svetlobe.



Slika 2.3: Blokovna shema ostalih modulov

2.3 Razporeditev elementov



Slika 2.4: Razvojna plošča

2.4 Tehnične specifikacije

Preglednica 2.1: Tehnične specifikacije razvojne plošče GEMS Amethyst

Parameter	**Vrednost**
Mikrokrmilnik	ESP32-C3-WROOM-02 (RISC-V 32-bitni, 160 MHz, Wi-Fi + BLE 5.0)
Vhodna napetost (USB)	5 V prek USB-C
Obratovalna napetost	3.3 V (regulirana)
Maksimalni izhodni tok	300 mA
Tipična poraba toka	∼80 mA aktivno (Wi-Fi vklopljen), <10 µA v globokem spanju
Brezžični vmesniki	Wi-Fi 802.11 b/g/n (2.4 GHz), Bluetooth LE 5.0
USB vmesnik	USB 2.0 Full-Speed (z ESD zaščito)
ESD zaščita	±15 kV (zrak), ±8 kV (kontakt)
Senzor svetlobe	TEPT5700 – največja občutljivost pri ∼570 nm (vidna svetloba)
Podprti vmesniki	UART, SPI, I^2C, PWM, ADC (12-bitni), GPIO
Dimenzije plošče	45mm x 40mm
Obratovalna temperatura	-40 °C do +85 °C

2.5 Fizične povezave in izdelava

Plošča uporablja dvoplastni PCB, ki vsebuje več plasti bakra. Te plasti omogočajo povezavo med različnimi elektronskimi komponentami na plošči. V našem primeru imamo bakrene plasti na obeh straneh plošče. Glavna funkcija bakrenih plasti je zagotavljanje poti za električne signale med različnimi komponentami na PCB-ju. Te poti so oblikovane kot sledilne poti, ki so izrezane iz bakrene plasti. Na zunanji plasti bakra se oblikujejo plastični odseki, ki so izolirani od zunanjih plasti ter med seboj s plastmi epoksidne smole. Proces izdelave PCB-ja se začne z načrtovanjem vezja v elektronskem CAD programu (KiCad), kjer se določijo postavitve komponent, sledi bakra in lokacije lukenj za montažo. Nato se izdela fotomaska, ki temelji na načrtovanem vezju in vsebuje informacije o sledi poti ter lokacijah lukenj za komponente. Substrat PCB-ja se pripravi z nanašanjem tankih plasti bakra (sledi poti) na obeh straneh. Sledi se ustvarijo s pomočjo fotomaske, ki ščiti bakreni sloj med izpostavljanjem izbranih delov bakra UV svetlobi za utrjevanje sledi. Nezaščiteni deli bakra se nato jedkajo, da se ustvarijo sledi za električne povezave. Po tem se izvrta luknje za montažo komponent. Na koncu sledi še namestitev komponent na PCB in pritrditev komponent z lotanjem. Pri oblikovanju PCB-ja je pomembno upoštevati širino in debelino sledilnih poti (traces) ter razporeditev bakrenih plasti, da se zagotovi ustrezna električna zmogljivost in zanesljivost vezja. Poleg tega se bakrene plasti uporabljajo tudi za izdelavo površinskih kontaktnih točk, ki omogočajo pritrditev in lotanje elektronskih komponent na PCB-ju.

Pri spajkanju elementov, kot so upori, pa je potrebna pazljivost.

Navodilo za spajkanje:

- Najprej pripravimo razvojno ploščico in komponente, ki jih želimo lotati.
- Potem namažemo ploščico s flux-om.
- Nastavimo spajkalnik na približno 350 °C.
- Nanesemo majhno količino spajke na eni strani kontakta, drugo stran lotamo potem.
- Položimo komponente na kontakte ter s spajkalnikom zgrejemo spajke in tako ustvarimo povezave.
- Ko smo končali z lotanjem vseh uporov na eni strani, potem zalotamo še drugo stran.
- Na koncu preverimo vse lotane povezave in se prepričamo, da ni hladnih spojev ali kratkih stikov.

Primer spremembe povezave: Odklop LED in priklop zunanje LED prek pina LED_R z uporom.

Poglavje 3

Delovanje razvojne plošče

3.1 Glavne funkcije

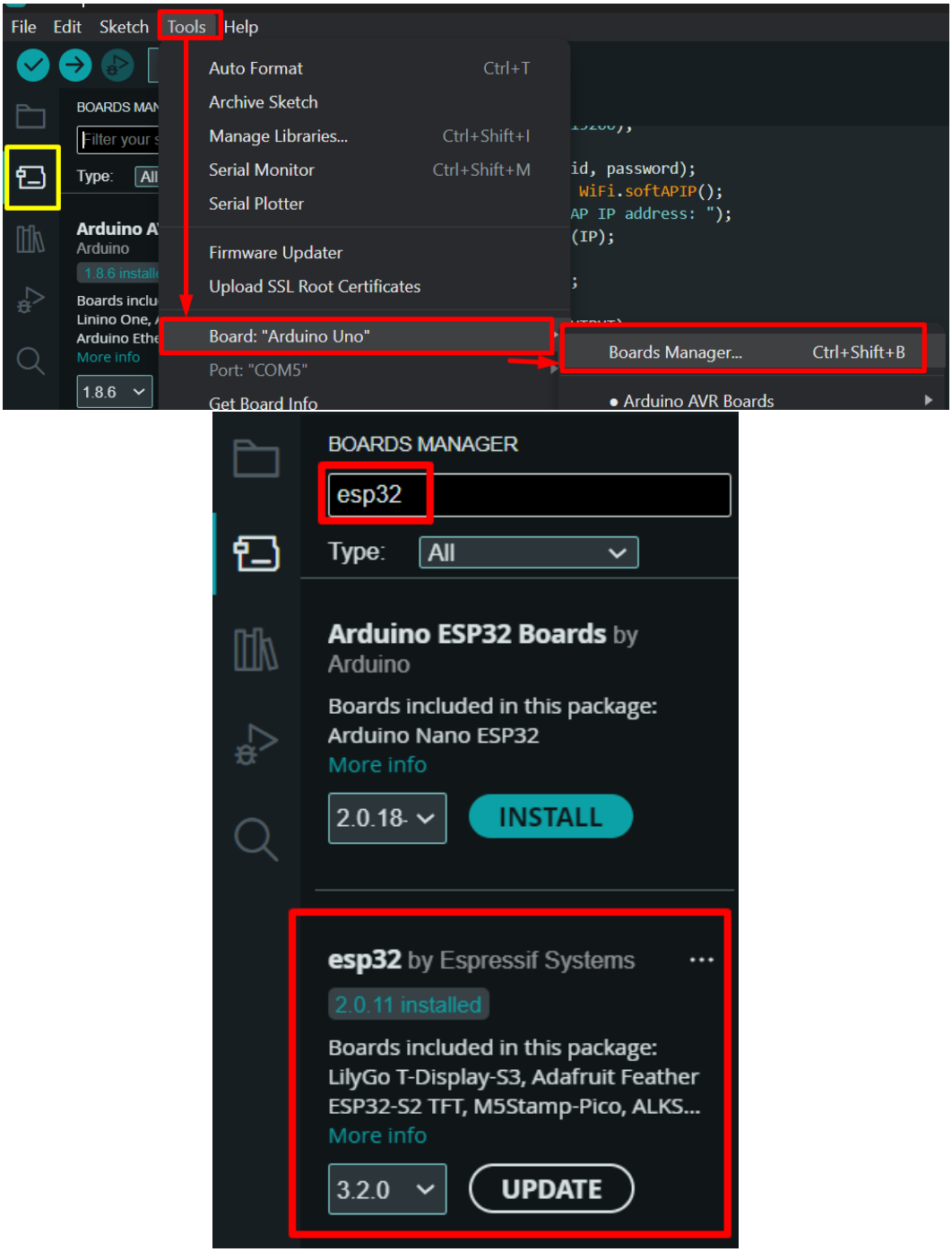
Razvojna plošča vključuje RGB LED, gumb, senzor osvetlitve z možnostjo zamenjave pull-down uporov za zmanjšanje tolerance senzorja, UART, I2C, WiFi, BLE in serijsko komunikacijo. Primerna je za IoT projekte in projekte, ki temeljijo na merjenju in upravljanju glede na ambientalno svetlobo.

3.2 Programiranje in okolje

Programiranje te razvojne plošče poteka preko Arduino IDE, ki je dostopen na https://www.arduino.cc/en/software/. Poleg tega je treba namestiti še paket plošč esp32.

3.2.1 Namestitev paketa esp32

Za namestitev paketa esp32 pojdite na Tools→Board→Board Manager ali kliknite ikono na levi strani. Poiščite "esp32 by Espressif Systems"in kliknite Install. Če je bilo uspešno nameščeno, se bo pod Tools→Board pojavila možnost esp32, znotraj katere so vse esp32 plošče, vključno z ESP32C3 Dev Module, ki je naša plošča.



Slika 3.1: Namestitev paketa esp32

3.2.2 Postopek nalaganja programa

Za nalaganje programa na ploščo jo priključite preko USB-ja, preverite da so nastavitve plošče ustrezne (spodaj navedene), in kliknite Upload. Ko piše "done uploading", lahko ploščo odklopite. Board settings:

1. Board: ESP32C3 Dev Module

2. Port: ustrezen COM port
3. USB CDC On Boot: Enabled
4. CPU Frequency: 80MHz

3.3 Pomembni pini

1. Gumb: 9
2. Rdeča LED: 7
3. Zelena LED: 10
4. Modra LED: 4
5. LX_A: 1
6. LX_B: 3
7. LX_OUT: 0
8. SDA: 5
9. SCL: 6
10. RX: 20
11. TX: 21

3.4 Uporaba light sensorja

Z zamenjavo stanj pinov LX_A in LX_B na HIGH ali LOW(to lahko naredite tudi ročno s spreminjanjem uporov prek jumperjev) lahko spreminjate občutljivost svetlobnega senzorja. Spodaj je primer kode, kjer lahko z izbiro mode 1, 2 ali 3 spreminjate območje izmerjene izhodne napetosti.

```
#define LX_A 1
#define LX_B 3
#define LX_OUT 0

void setup() {
  Serial.begin(115200);

  pinMode(LX_A, OUTPUT);
  pinMode(LX_B, OUTPUT);
  pinMode(LX_OUT, INPUT);

  Serial.println("Select mode:");
```

```
13   Serial.println("Mode 1");
14   Serial.println("Mode 2");
15   Serial.println("Mode 3");
16 }
17
18 void loop() {
19   if (Serial.available() > 0) {
20     char c = Serial.read();
21
22     switch (c) {
23       case '1':  // Mode 1
24         digitalWrite(LX_A, LOW);
25         digitalWrite(LX_B, LOW);
26         Serial.println("Mode set to 1");
27         break;
28       case '2':  // Mode 2
29         digitalWrite(LX_A, LOW);
30         digitalWrite(LX_B, HIGH);
31         Serial.println("Mode set to 2");
32         break;
33       case '3':  // Mode 3
34         digitalWrite(LX_A, HIGH);
35         digitalWrite(LX_B, LOW);
36         Serial.println("Mode set to 3");
37         break;
38     }
39   }
40
41   int output = analogRead(LX_OUT);
42
43   Serial.println(output);
44
45   delay(100);
46 }
```

3.5 Primeri funkcije

3.5.1 RGB LED

Povezava do knjižnice: https://docs.arduino.cc/language-reference/en/functions/digital-io/digitalwrite/

```
1 #define LED_R 7
2 #define LED_G 10
3 #define LED_B 4
4
```

```
5  void setup() {
6    pinMode(LED_R, OUTPUT);
7    pinMode(LED_G, OUTPUT);
8    pinMode(LED_B, OUTPUT);
9  }
10
11 void loop() {
12   analogWrite(LED_R, 255); // red
13   analogWrite(LED_G, 0);
14   analogWrite(LED_B, 0);
15   delay(1000);
16
17   analogWrite(LED_R, 0);
18   analogWrite(LED_G, 255); // green
19   analogWrite(LED_B, 0);
20   delay(1000);
21
22   analogWrite(LED_R, 0);
23   analogWrite(LED_G, 0);
24   analogWrite(LED_B, 255); // blue
25   delay(1000);
26 }
```

3.5.2 Gumb

Povezava do knjižnice: https://docs.arduino.cc/language-reference/en/functions/digital-io/digitalread/

```
1  #define BTN 9
2  #define LED_R 7
3
4  void setup() {
5    pinMode(BTN, INPUT_PULLUP);
6    pinMode(LED_R, OUTPUT);
7  }
8
9  void loop() {
10   if (digitalRead(BTN) == LOW) {
11     digitalWrite(LED_R, HIGH); // button pressed
12   } else {
13     digitalWrite(LED_R, LOW);
14   }
15 }
```

3.5.3 Light sensor

Povezava do knjižnice: https://docs.arduino.cc/language-reference/en/functions/analog-io/analogRead/

```
#define LX_A 1
#define LX_B 3
#define LX_OUT 0

void setup() {
  Serial.begin(115200);
  delay(300);
}

void loop() {
  int lightLevel = lx(1);
  Serial.print("Light Level: ");
  Serial.println(lightLevel);
  delay(500);
}

int lx(int mode) {
  switch (mode) {
    case 0:
      pinMode(LX_A, INPUT);
      pinMode(LX_B, INPUT);
      break;
    case 1:
      pinMode(LX_A, INPUT);
      pinMode(LX_B, OUTPUT);
      digitalWrite(LX_B, LOW);
      break;
    case 2:
      pinMode(LX_A, OUTPUT);
      digitalWrite(LX_A, LOW);
      pinMode(LX_B, INPUT);
      break;
    default:
      pinMode(LX_A, INPUT);
      pinMode(LX_B, INPUT);
  }

  return analogRead(LX_OUT);
}
```

3.5.4 Serial komunikacija

Povezava do knjižnice: https://www.arduino.cc/reference/en/language/functions/communication/serial/

```
void setup() {
  Serial.begin(115200);
  delay(1000);
  Serial.println("Serial is working!");
}

void loop() {
  if (Serial.available()) {
    String input = Serial.readStringUntil('\n');
    Serial.print("You said: ");
    Serial.println(input);
  }
}
```

3.5.5 Wi-fi

Povezava do knjižnice: https://docs.arduino.cc/libraries/wifi/

```
#include <WiFi.h>

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

void setup() {
  Serial.begin(115200);
  delay(1000);

  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

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

```

```
25   server.begin();
26 }
27
28 void loop() {
29   WiFiClient client = server.available();
30
31   if (client) {
32     Serial.println("New Client.");
33     String request = client.readStringUntil('\r');
34     Serial.println(request);
35     client.flush();
36
37     client.println("HTTP/1.1 200 OK");
38     client.println("Content-type:text/html");
39     client.println();
40     client.println("<!DOCTYPE html><html><body><h1>
    Mehatronika</h1></body></html>");
41     client.println();
42     client.stop();
43     Serial.println("Client disconnected.");
44   }
45 }
```

3.5.6 Bluetooth

Povezava do knjižnice: https://github.com/nkolban/ESP32_BLE_Arduino

```
1 #include <BLEDevice.h>
2 #include <BLEUtils.h>
3 #include <BLEServer.h>
4
5 #define SERVICE_UUID        "4fafc201-1fb5-459e-8fcc-
    c5c9c331914b"
6 #define CHARACTERISTIC_UUID "beb5483e-36e1-4688-b7f5-
    ea07361b26a8"
7
8 void setup() {
9   Serial.begin(115200);
10  Serial.println("Starting BLE work!");
11
12  BLEDevice::init("Long name works now");
13  BLEServer *pServer = BLEDevice::createServer();
14  BLEService *pService = pServer->createService(SERVICE_UUID)
     ;
15  BLECharacteristic *pCharacteristic = pService->
    createCharacteristic(
16                                         CHARACTERISTIC_UUID,
```

```
                                                BLECharacteristic::
    PROPERTY_READ |
                                                BLECharacteristic::
    PROPERTY_WRITE
                                              );

  pCharacteristic->setValue("Hello World says Neil");
  pService->start();
  BLEAdvertising *pAdvertising = BLEDevice::getAdvertising();
  pAdvertising->addServiceUUID(SERVICE_UUID);
  pAdvertising->setScanResponse(true);
  pAdvertising->setMinPreferred(0x06);
  pAdvertising->setMinPreferred(0x12);
  BLEDevice::startAdvertising();
  Serial.println("Characteristic defined! Now you can read it
    in your phone!");
}
```

3.5.7 I2C

Povezava do knjižnice: https://www.arduino.cc/en/Reference/Wire

```
#include <Wire.h>
#define SDA 5
#define SCL 6

void setup() {
  Wire.begin(SDA, SCL);
  Serial.begin(115200);
  delay(1000);
  Serial.println("I2C Scanner starting...");

  for (uint8_t address = 1; address < 127; ++address) {
    Wire.beginTransmission(address);
    if (Wire.endTransmission() == 0) {
      Serial.print("I2C device found at address 0x");
      Serial.println(address, HEX);
    }
    delay(5);
  }

  Serial.println("Scan complete.");
}
```

3.5.8 UART

Povezava do knjižnice: https://docs.arduino.cc/learn/communication/uart/

```
#define RX 20
#define TX 21

void setup() {
  Serial.begin(115200);                 // USB Serial Monitor
  Serial1.begin(9600, SERIAL_8N1, RX, TX); // UART Serial
  Serial.println("UART echo demo started...");
}

void loop() {
  if (Serial1.available()) {
    char c = Serial1.read();      // Read from UART
    Serial1.write(c);             // Echo back to UART
    Serial.print("Received: ");
    Serial.println(c);            // Also print to USB serial
   for debugging
  }
}
```