

Računalni sustavi stvarnog vremena

**SVJETLO ZA BICIKL**

PROJEKTNI ZADATAK

Grupa:

Petar Nenadić

Domagoj Voćanec

Osijek, 2021.

Sadržaj

[1. Opis projektnog zadatka 3](#_Toc72180757)

[2. Prijedlog rješenja 3](#_Toc72180758)

[2.1. Croduino Basic2 pločica s mikrokontrolerom 4](#_Toc72180759)

[2.2. Modul GY-521 (akcelerometar + žiroskop) 5](#_Toc72180760)

[2.3. Fotootpornik 6](#_Toc72180761)

[2.4. LE diode 7](#_Toc72180762)

[2.5. Arduino IDE programsko okruženje 8](#_Toc72180763)

[3. Opis rješenja 10](#_Toc72180764)

[4. Zaključak 19](#_Toc72180765)

# Opis projektnog zadatka

Cilj ovog projektnog zadatka bio je realizacija sustava svjetla za bicikl kojem se, ovisno o smjeru prema kojem se naginje bicikl, uključuju pokazivači smjera. Također, sustav treba uključivati i automatsko uključivanje prednjeg svjetla ovisno o vanjskoj količini svjetlosti te uključivanje stop svjetla ovisno o nagibu biciklista. Senzori koji su potrebni za realizaciju projekta su:

* LED-ice
* Fotootpornik
* Akcelerometar GY-521
* Croduino Basic modul

Sustav će biti realiziran pomoću Arduino kompatibilnog modula i 4 LE-diode. LE-diode će zamijeniti pokazivače smjera, prednje svjetlo te stop svjetlo. Akcelerometar će se koristiti za očitavanje stanja bicikla, dok će fotootpornik detektirati količinu svjetlosti koja dolazi izvana. Za pisanje koda korišten je Arduino softver, kojega je kreirala talijanska tvrtka SmartProjects. Biblioteke koje su uključene u projekt su *Adafruit\_MPU6050*, *Adafruit\_Sensor*, te *Wire*.

# Prijedlog rješenja

„Mozak“ sustava je Croduino Basic modul, na kojega su spojeni senzori i LE diode. Dvije LE diode. koje služe kao pokazivači smjera, trebaju se paliti i gasiti u ovisnosti o položaju akcelerometra. Također, jedna LE dioda će se uključiti kada se akcelerometar nagne prema natrag te služi kao zamjena za stop svjetlo. Fotootpornik će služiti za očitanje vanjske svjetlosti, te u ovisnosti o istoj, uključivat će LE diodu koja služi kao prednje svjetlo.

Kod će biti napisan u Arduino IDE programskom okruženju. Arduino je ime za open-source računalnu i softversku platformu koja omogućava dizajnerima i konstruktorima stvaranje uređaja i naprava koje spajaju računala s fizičkim svijetom. Arduino platforma je skup elektroničkih i softverskih komponenti koje se mogu jednostavno povezivati u složenije cjeline s ciljem izrade zabavnih i poučnih elektroničkih sklopova.

## Croduino Basic2 pločica s mikrokontrolerom

Croduino Basic2 je hrvatska Arduino pločica koja je u potpunosti kompatibilna s Arduino Nano razvojnom pločicom. Sadrži 22 I/O pina, a za komunikaciju koristi Silabsov CP2102 USB to UART bridge. Neke od karakteristika pločice su:

* Atmel ATmega328 mikrokontroler (32kb flash, 2kb RAM, 0.5kb EEPROM memorije)
* Silabs CP2102 USB to UART bridge
* 22 I/O pina (14 digitalnih I/O, 6 PWM, 8 analognih ulaza)
* Dimenzije: 3cm x 5cm, s klasičnim Croduino footprintom koji pristaje eksperimentalnoj pločici
* LE dioda na pinu 13
* Regulator za vanjsko napajanje, sa zaštitnim osiguračem



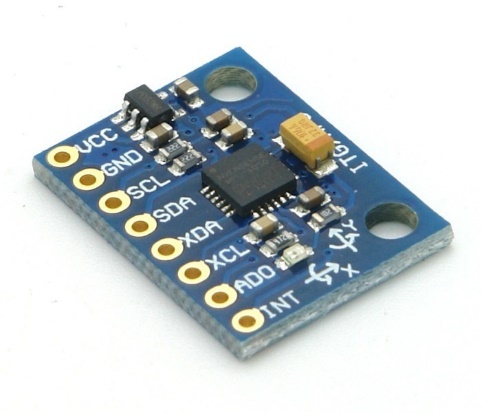
**Slika 1. Croduino Basic2 s mikroupravljačem**

## Modul GY-521 (akcelerometar + žiroskop)

Modul GY-521 je inačica pločice sa senzorom MPU-6050 kojega je razvila tvrtka Microelectromechanical Systems. Modul detektira sile koje djeluju na njega u tri smjera (x, y, z). Kada je u stanju mirovanja detektirat će Zemljinu silu teže, a prilikom kretanja će detektirati sve sile koje na njega djeluju. Iz tog razloga, ovaj senzor može mjeriti i detektirati mnoge stvari, kao što su npr.: udarci, nagibi, kretanje, tapkanje, korake itd. Ovaj uređaj sadrži i žiroskop i senzor temperature. Žiroskop može detektirati orijentaciju i kutnu brzinu te dati dodatne korisne informacije o položaju.

Karakteristike:

* 8 I/O pinova
* Regulator napona, stoga može biti priključen na izvor s 3.3V ili 5V
* Raspon akcelerometra: ±2g, ±4g, ±8g, ±16g (g – ubrzanje Zemljine sile teže)
* Raspon žiroskopa: ±250, 500, 1000, 2000 °/s



**Slika 2. Modul GY-521 (akcelerometar + žiroskop)**

## Fotootpornik

Fotootpornik je vrsta otpornika koja svoj otpor mijenja u ovisnosti o količini svjetlosti koja padne na njega. Izrađuje se od poluvodiča s velikim električnim otporom. Odlični su za korištenje kada treba pratiti količinu svjetlosti, iz tog razloga se najčešće koriste u kamerama, uličnim svjetiljkama, radio satovima te vanjskim satovima.

Karakteristike:

* Nominalni otpor: 10 kΩ



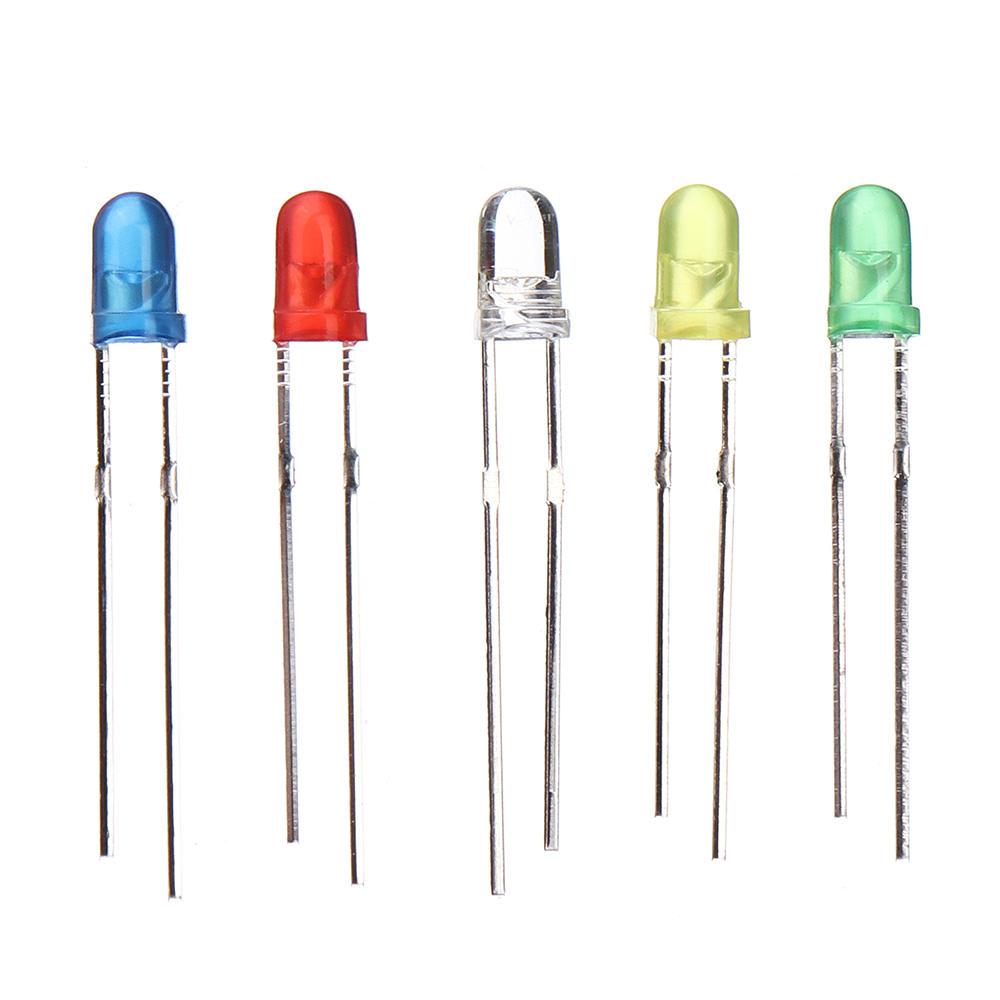
**Slika 3. Fotootpornik**

## LE diode

Svjetleće diode ili LED (skraćeno od engl. *Light Emitting Diode*) su poluvodički elementi koji pretvaraju električni signal u optički. Najčešće se primjenjuje kao indikator, na primjer na signalnim pločama uređaja i strojeva ili kao alfanumerički pokazivač na zaslonima kalkulatora, za ukrasno osvjetljenje te u industriji zabave. Imaju veliko područje primjene i u optičkim komunikacijama gdje služe za prijenos podataka na kraće udaljenosti.

Karakteristike:

* Dolaze u raznim bojama: crvena, bijela, plava, RGB, itd.
* Jakost emitiranja svjetla ovisi o količini struje
* Temperatura boje: 2600K pa na više



**Slika 4. LE diode**

## Arduino IDE programsko okruženje

Arduino IDE je open-source softver koji olakšava pisanje i učitavanje koda na Croduino ili neku drugu Arduino kompatibilnu pločicu. Razvijen je za sve platforme – Windows, Linux i Mac operativne sustave. Okruženje je pisano u programskom jeziku Java, a bazirano je na Processingu. Instalacijski paket je moguće preuzeti sa službene stranice i instalirati na računalo prateći upute. Trenutna verzija je 1.8.15.

Svaki program pisan u Arduino sastoji se od dvije osnovne petlje: *loop()* i *setup()*. *Setup()* dio koda se izvršava samo jednom, i to pri inicijalizaciji mikrokontrolera. Tu se najčešće pišu naredbe koje definiraju komponente koje su spojene na mikrokontroler te način na koji su komponente spojene.

Nakon izvršenja *setup()* dijela izvršava se *loop()* dio programa. Sve naredbe koje su napisane u ovom dijelu ponavljaju se sve dok ponovno ne pokrenemo mikrokontroler.

Neke od osnovnih naredbi razvojnog okruženja su:

**pinMode(pin, mode) –** u ovoj naredbi se definiraju izvodi pojedinog mikrokontrolera, odnosno, je li ulaznog ili izlaznog tipa. Prvi argumentom se definira za koji izvod se želi postaviti režim rada, dok drugi služi za definiranje režima. Režimi rada mogu biti INPUT, OUTPUT ili INPUT\_PULLUP

**digitalWrite(pin, value) –** ova naredba služi za uključivanje ili isključivanje pojedinog mikrokontrolera. Prvim argumentom se definira izvod, a drugim stanje mikrokontrolera. Vrijednosti mogu biti LOW ili HIGH.

**Serial.begin(speed)** – naredba za pokretanje serijske komunikacije

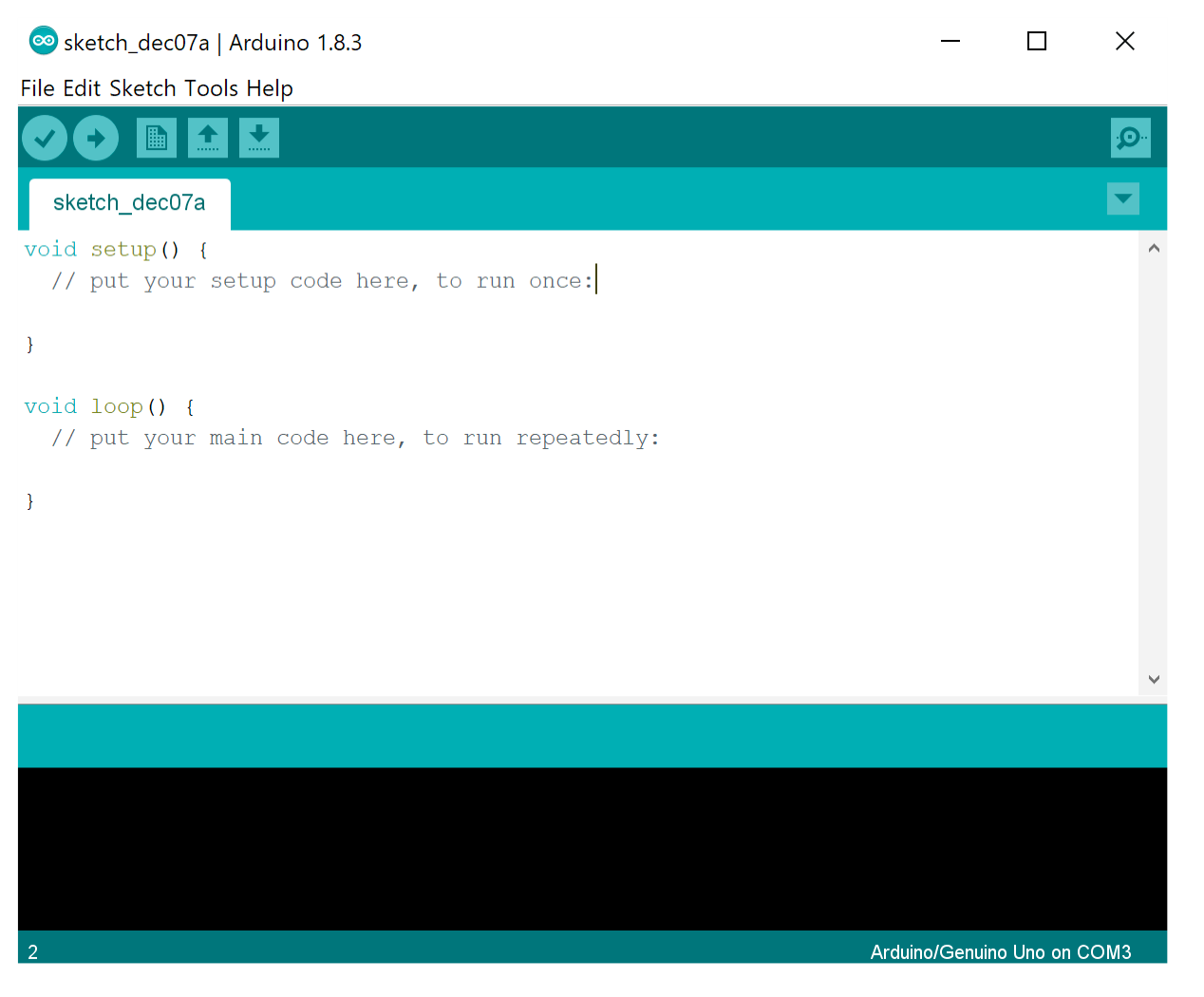
**Serial.print(value) –** ispisivanje vrijednosti u Serial Monitoru

**digitalRead(pin) –** naredba za očitavanje vrijednosti sa specifičnog digitalnog pina, spremaju se vrijednosti HIGH i LOW

**analogRead(pin) –** naredba za očitavanje vrijednosti sa specifičnog analognog pina. Potrebno je oko 100 milisekundi za očitanje analognog ulaza. Povratna vrijednost je od 0 do 1023 ako se radi o int tipu podatka.

**delay(ms) –** naredba koja služi za zaustavljanje izvršavanja programa na određeno vrijeme. Naredba prima samo jedan argument kojim se definira vrijeme u milisekundama

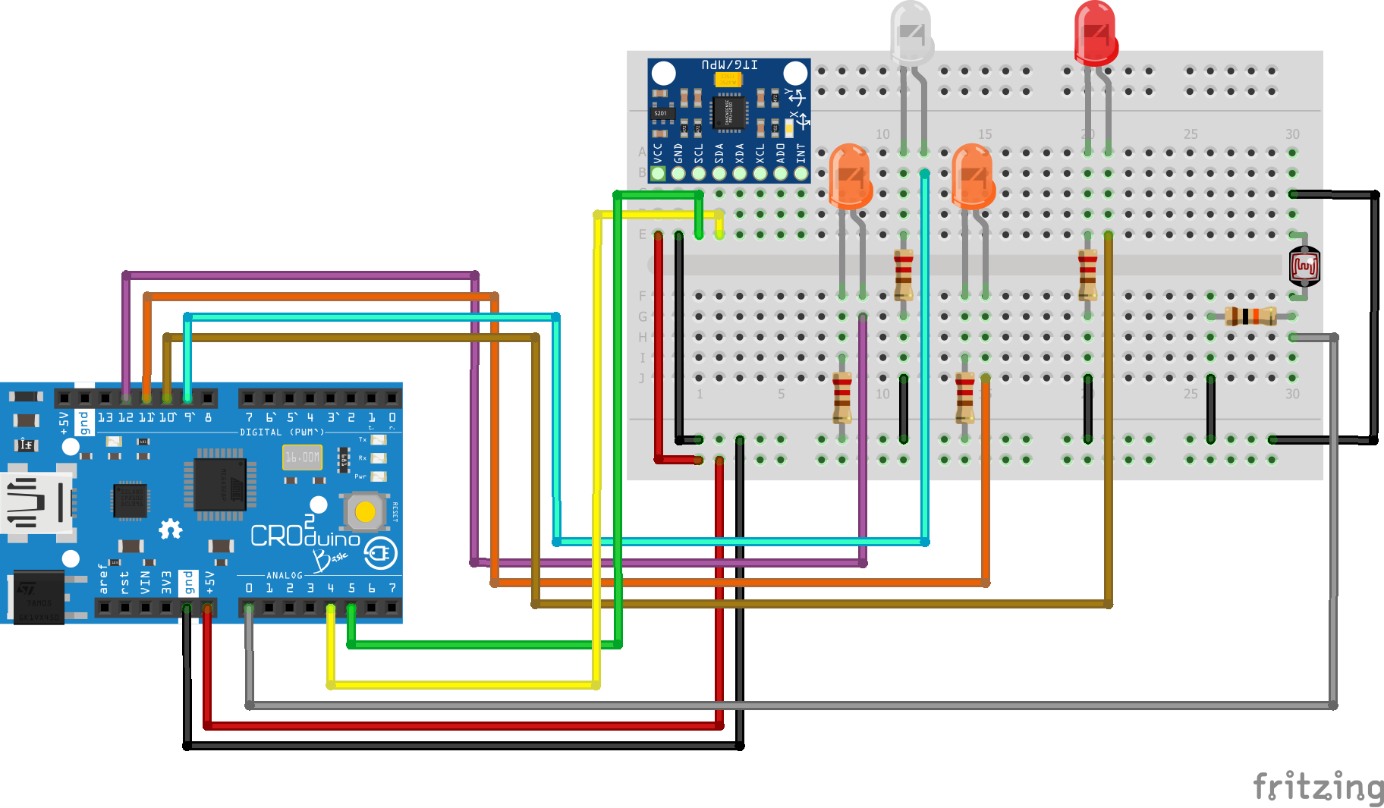
**delayMicroseconds(us) –** naredba koja služi za zaustavljanje izvršavanja programa na određeno vrijeme. naredba prima samo jedan argument kojim se definira vrijeme u mikrosekundama.



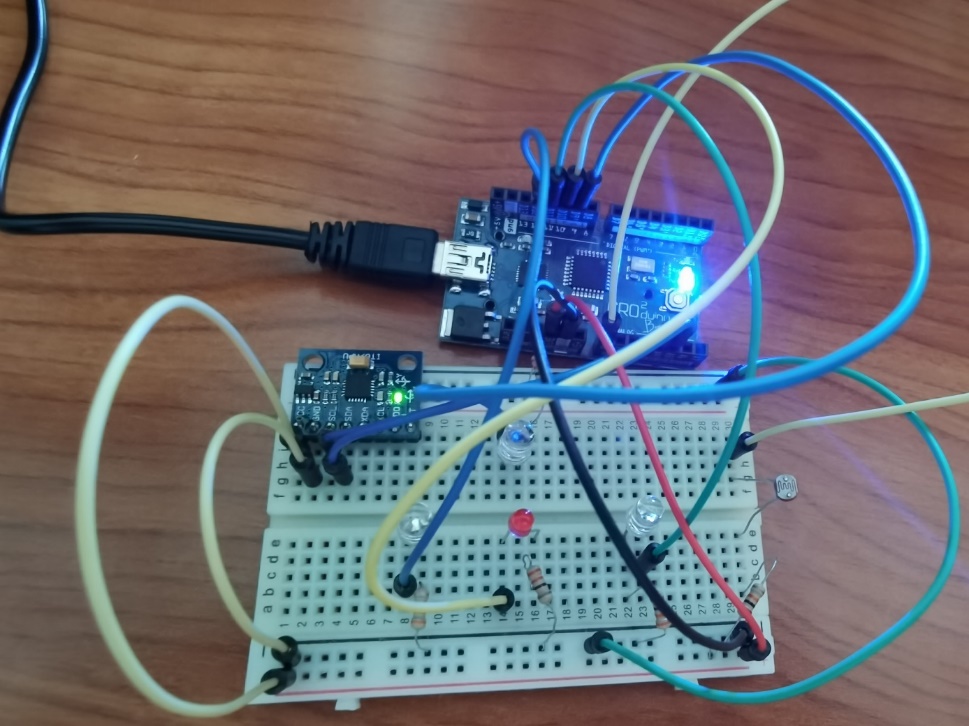
**Slika 5. Arduino IDE programsko okruženje**

# Opis rješenja

Prije same izrade projekta potrebno je napraviti okvirnu shemu koja prikazuje način spajanja komponenti.

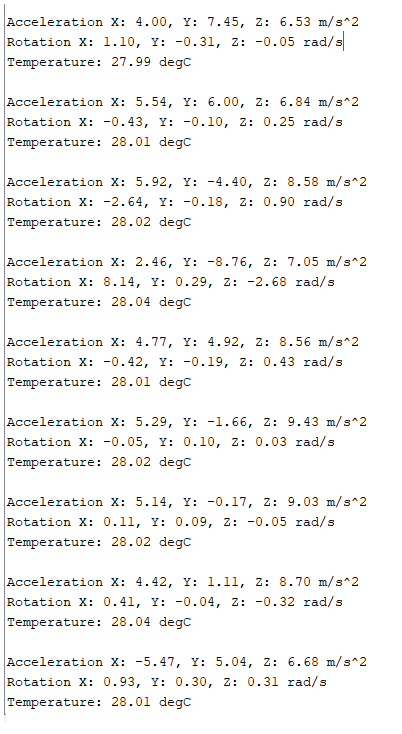


**Slika 6. Shema projektnog zadatka**



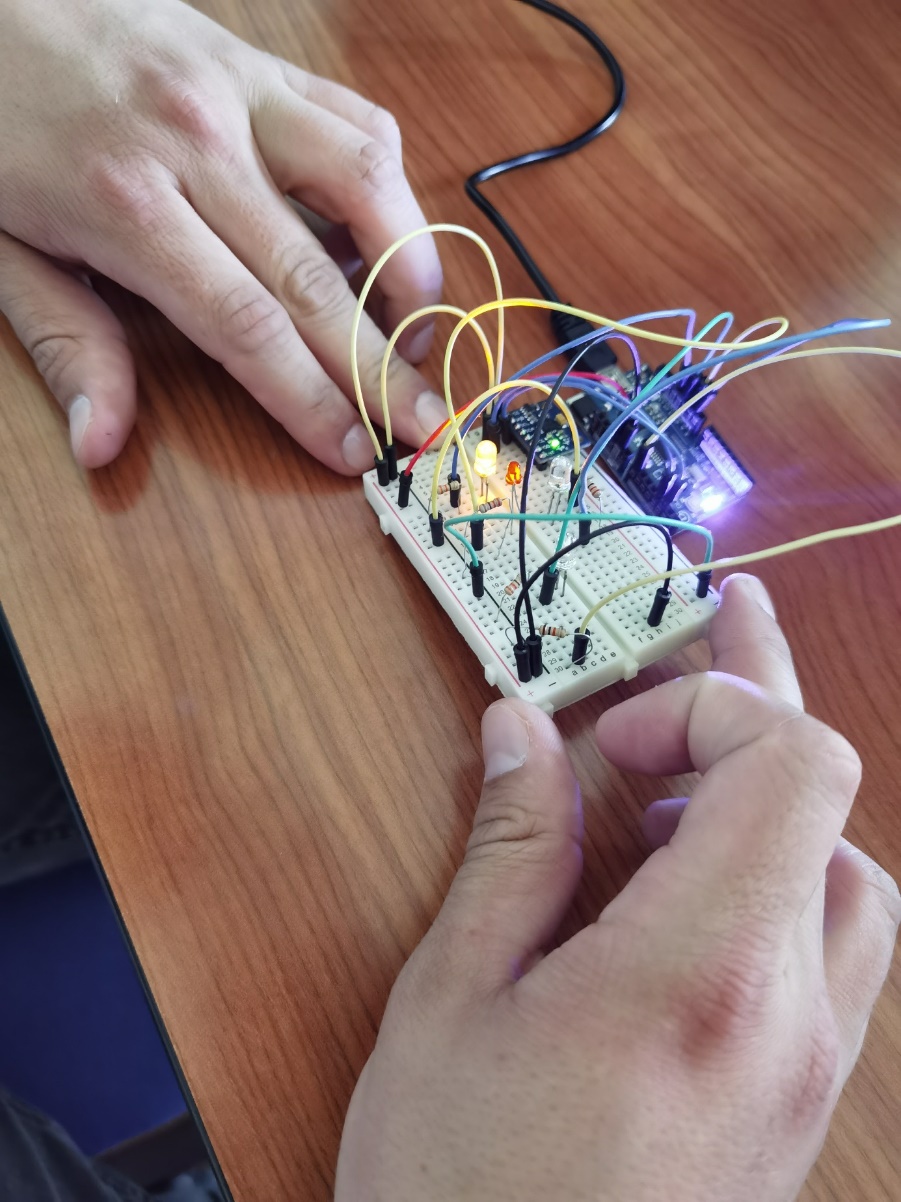
**Slika 7. Prikaz projekta**

Nakon izrade shematskog prikaza projekta na Croduino je spojen senzor GY-521 kako bi se provjerila njegova ispravnost i način rada. Senzor radi na principi naginjanja i povlačenja po površini, te tako daje vrijednosti za 3D Kartezijev sustav, odnosno vrijednosti za x, y, i z os. Na slici ispod prikazana su mjerenja izvršena samim senzorom:

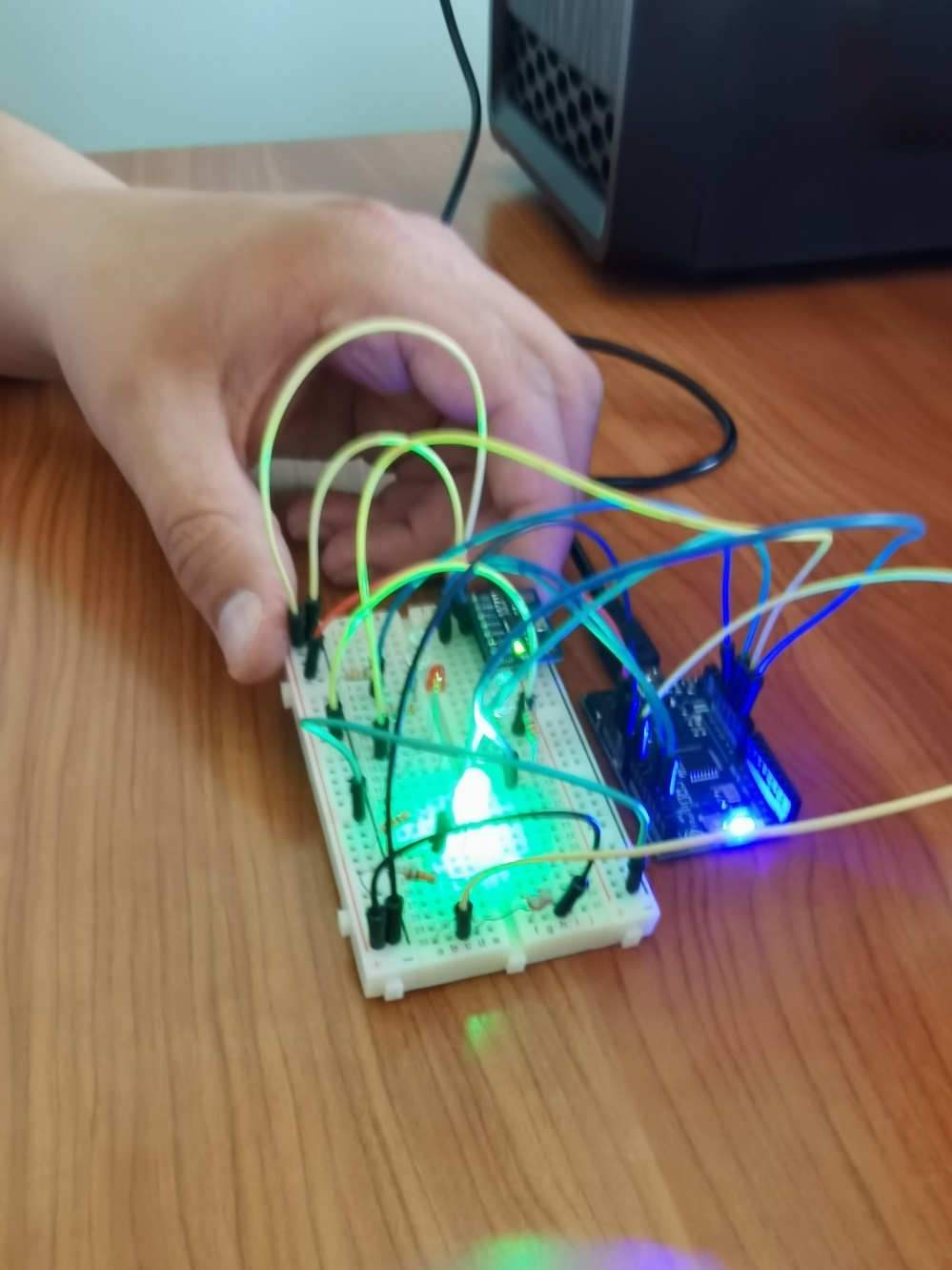


**Slika 8. Rezultati mjerenja izvršena senzorom GY-521**

Nakon provjere ispravnosti senzora spojene su LED-ice za prikaz pokazivača smjera, stop svjetlo i prednje svjetlo. Pokazivači smjera se uključuju kako se senzor GY-521 okreće. Odnosno, ako se senzor okrene u lijevo upali se lijevi pokazivač smjera i naizmjenično se uključuje i isključuje svakih 200 milisekundi. Također, ako se senzor okrene na desnu stranu, desni pokazivač smjera se naizmjenično uključuje i isključuje, vrijeme odgode između uključivanja i isključivanja je kao i kod lijevog pokazivača smjera.

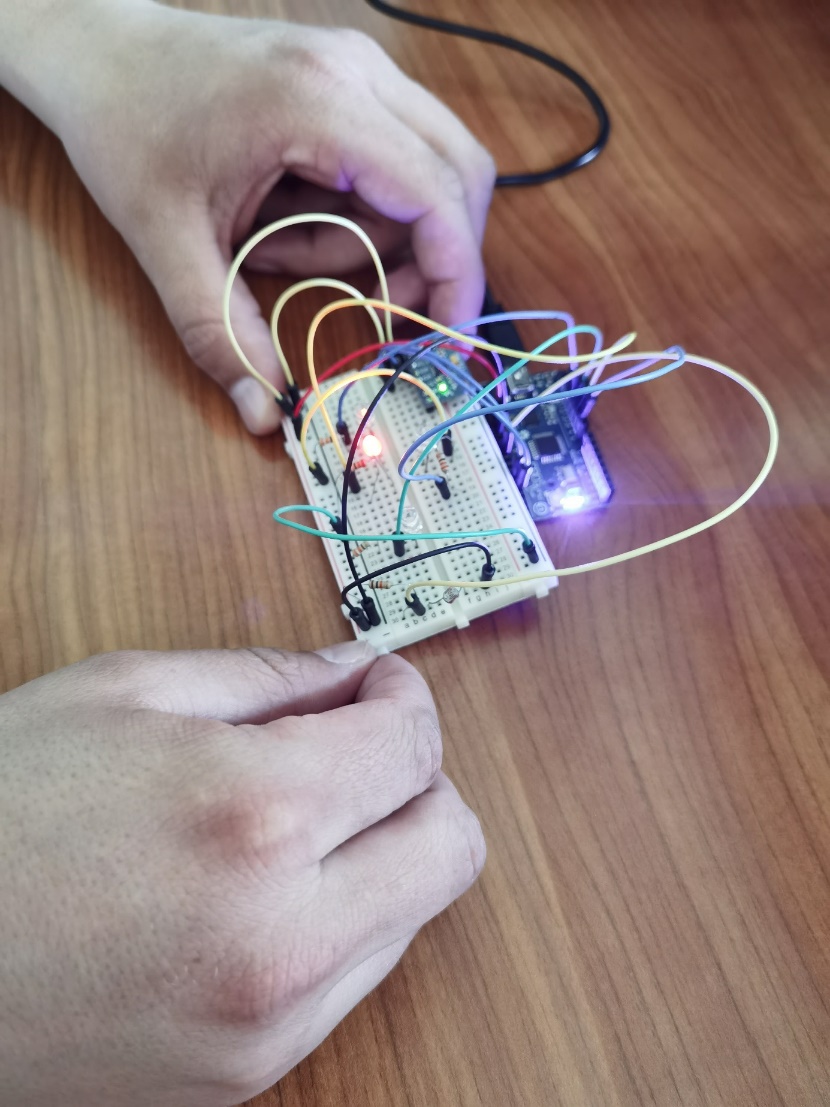


**Slika 9. Lijevi pokazivač smjera**



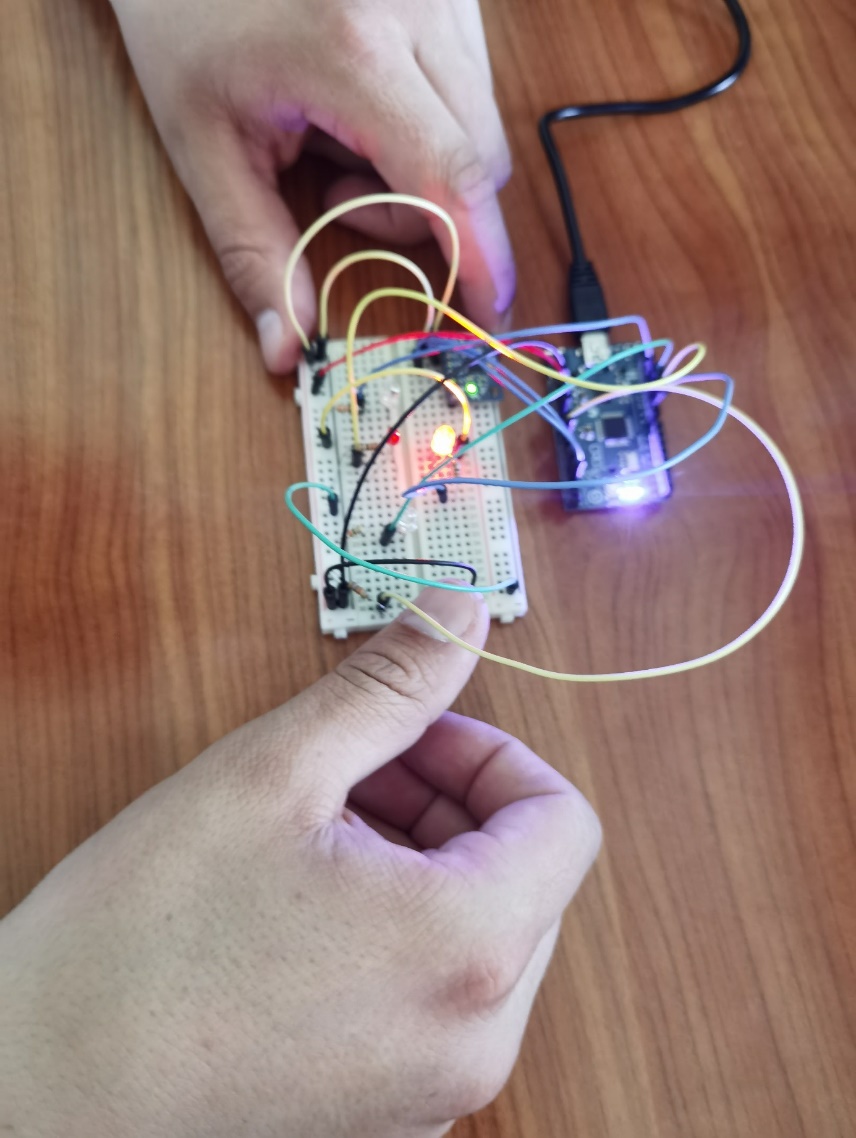
**Slika 10. Desni pokazivač smjera**

Poslije implementacije pokazivača smjera spojena je LED-ica koja predstavlja stop svjetlo. Ova LED-ica se treba uključiti ako se senzor GY-521 naginje prema natrag, ili ako biciklist pritisne kočnicu.



**Slika 11. Stop svjetlo**

Nakon uspješne realizacije pokazivača smjera i stop svjetla, na spoj s Croduinom je dodana LED-ica koja služi kao prednje svjetlo. Ta LED-ica je spojena na fotootpornik kako bi se uključivala ovisno o vanjskoj količini svjetlosti. Ako je vani mračno, LED-ica se uključuje, odnosno, ako je vani dovoljno svjetla, LED-ica se isključuje kako bi uštedjela energiju.



**Slika 12. Prednje svjetlo**

Programski kod kojim je realizirano rješenje nalazi se ispod:

#include <Adafruit\_MPU6050.h>

#include <Adafruit\_Sensor.h>

#include <Wire.h>

const int LeftTurnSignal = 12;

const int RightTurnSignal = 11;

const int BrakeLight = 10;

const int LightSensor = A0;

const int HeadLight = 9;

int lightCal;

int lightVal;

Adafruit\_MPU6050 mpu;

void setup(void) {

Serial.begin(115200);

pinMode(LeftTurnSignal, OUTPUT);

pinMode(RightTurnSignal, OUTPUT);

pinMode(BrakeLight, OUTPUT);

pinMode(HeadLight, OUTPUT);

lightCal = analogRead(LightSensor);

while (!Serial)

delay(10);

Serial.println("Adafruit MPU6050 test!");

// Try to initialize!

if (!mpu.begin()) {

Serial.println("Failed to find MPU6050 chip");

while (1) {

delay(10);

}

}

Serial.println("MPU6050 Found!");

mpu.setAccelerometerRange(MPU6050\_RANGE\_8\_G);

Serial.print("Accelerometer range set to: ");

switch (mpu.getAccelerometerRange()) {

case MPU6050\_RANGE\_2\_G:

Serial.println("+-2G");

break;

case MPU6050\_RANGE\_4\_G:

Serial.println("+-4G");

break;

case MPU6050\_RANGE\_8\_G:

Serial.println("+-8G");

break;

case MPU6050\_RANGE\_16\_G:

Serial.println("+-16G");

break;

}

mpu.setGyroRange(MPU6050\_RANGE\_500\_DEG);

Serial.print("Gyro range set to: ");

switch (mpu.getGyroRange()) {

case MPU6050\_RANGE\_250\_DEG:

Serial.println("+- 250 deg/s");

break;

case MPU6050\_RANGE\_500\_DEG:

Serial.println("+- 500 deg/s");

break;

case MPU6050\_RANGE\_1000\_DEG:

Serial.println("+- 1000 deg/s");

break;

case MPU6050\_RANGE\_2000\_DEG:

Serial.println("+- 2000 deg/s");

break;

}

mpu.setFilterBandwidth(MPU6050\_BAND\_21\_HZ);

Serial.print("Filter bandwidth set to: ");

switch (mpu.getFilterBandwidth()) {

case MPU6050\_BAND\_260\_HZ:

Serial.println("260 Hz");

break;

case MPU6050\_BAND\_184\_HZ:

Serial.println("184 Hz");

break;

case MPU6050\_BAND\_94\_HZ:

Serial.println("94 Hz");

break;

case MPU6050\_BAND\_44\_HZ:

Serial.println("44 Hz");

break;

case MPU6050\_BAND\_21\_HZ:

Serial.println("21 Hz");

break;

case MPU6050\_BAND\_10\_HZ:

Serial.println("10 Hz");

break;

case MPU6050\_BAND\_5\_HZ:

Serial.println("5 Hz");

break;

}

Serial.println("");

delay(1000);

}

void loop() {

*/\* Get new sensor events with the readings \*/*

sensors\_event\_t a, g, temp;

mpu.getEvent(&a, &g, &temp);

*/\* Print out the values \*/*

Serial.print("Acceleration X: ");

Serial.print(a.acceleration.x);

Serial.print(", Y: ");

Serial.print(a.acceleration.y);

Serial.print(", Z: ");

Serial.print(a.acceleration.z);

Serial.println(" m/s^2");

Serial.print("Rotation X: ");

Serial.print(g.gyro.x);

Serial.print(", Y: ");

Serial.print(g.gyro.y);

Serial.print(", Z: ");

Serial.print(g.gyro.z);

Serial.println(" rad/s");

Serial.print("Temperature: ");

Serial.print(temp.temperature);

Serial.println(" degC");

//read y values from accelerometer and according to it's position left or right turn signal is blinking,

//value > 1, right turn signal is blinking

//value < -1, left turn signal is blinking

if(a.acceleration.y > 1){

digitalWrite(RightTurnSignal, HIGH);

delay(200);

digitalWrite(RightTurnSignal, LOW);

delay(200);

}

else if(a.acceleration.y < -1){

digitalWrite(LeftTurnSignal, HIGH);

delay(200);

digitalWrite(LeftTurnSignal, LOW);

delay(200);

}

else{

digitalWrite(LeftTurnSignal, LOW);

digitalWrite(RightTurnSignal, LOW);

}

//read x values from accelerometer and according to it's position, turn on brake light

if(a.acceleration.x > 0){

digitalWrite(BrakeLight, HIGH);

}

else{

digitalWrite(BrakeLight, LOW);

}

//read value from photosensor and write it on headlight

lightVal = analogRead(LightSensor);

if (lightVal < lightCal - 50)

{

digitalWrite(9, HIGH);

}

else

{

digitalWrite(9, LOW);

}

Serial.println("");

delay(500);

}

# Zaključak

Projekt je u potpunosti uspješno izrađen, to jest, sve funkcionalnosti koje su zamišljene u ideji projekta su uspješno implementirane. Umjesto običnih LED-ica mogle bi biti ubačene ultra bright diode, više LED-ica za pojedino svjetlo ili LED trake kako bi biciklist bio lakše uočen. Također, umjesto jednog GY-521 senzora mogla bi biti ubačena 2 ili 3 senzora, po jedan za svaku os okretanja (x, y, z). Razlog umetanja više senzora je taj što bi očitanja sa senzora bila preciznija ako bi se senzor koristio za očitavanje samo jedne osi.

Ovaj projekt je bio edukativan zbog korištenja više senzora i potrebe da se na temelju ulaznih podataka iz više senzora dođe do željenog ishoda. Prednosti ovakvog projekta su što je intuitivan i nema potrebe da biciklist šalje uređaju naredbe koje treba raditi, već sav posao obavlja sam Croduino.