DIZAJN PRIJEMNIKA ZA KOMUNIKACIJU VIDLJIVIM SVJETLOM

Franić, Antonio

SVEUČILIŠTE U SPLITU

SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Stručni prijediplomski studij Elektronika

Antonio Franić

Z A V R Š N I R A D

DIZAJN PRIJEMNIKA ZA KOMUNIKACIJU VIDLJIVIM SVJETLOM

Split, lipanj 2023.

SVEUČILIŠTE U SPLITU

SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Stručni prijediplomski studij Elektronika

Predmet: Elektronički sklopovi

Z A V R Š N I R A D

Kandidat: Antonio Franić

Naslov rada: Dizajn prijemnika za komunikaciju vidljivim svjetlom

Mentor: Dr.sc. Tonko Kovačević

Split, lipanj 2023.

SADRŽAJ

DIZAJN PRIJEMNIKA ZA KOMUNIKACIJU VIDLJIVIM SVJETLOM

SAŽETAK

U radu je opisan sustav komunikacije temeljen na vidljivom svjetlu (VLC) koji koristi Arduino Uno R3, LED laser, DHT22 senzor za mjerenje temperature i vlage te fotodiodu. Glavni cilj sustava je prijenos senzorskih podataka temperature i vlage te njihov prikaz na LCD zaslonu. Koristeći Manchester biblioteke za predajnik i prijemnik, postignuta je pouzdana komunikacija između uređaja.

Ključne riječi: VLC, Arduino Uno, DHT22

DESIGN OF RECEIVER FOR VISIBLE LIGHT COMMUNICATION

SUMMARY

The paper describes a visible light communication (VLC) system based on Arduino Uno R3, LED laser, DHT22 temperature and humidity sensor, and a photodiode. The main objective of the system is to transmit temperature and humidity sensor data and display it on LCD screen. By utilizing Manchester libraries for the transmitter and receiver, reliable communication between the devices has been achieved.

Keywords: VLC, Arduino Uno, DHT22

1. UVOD

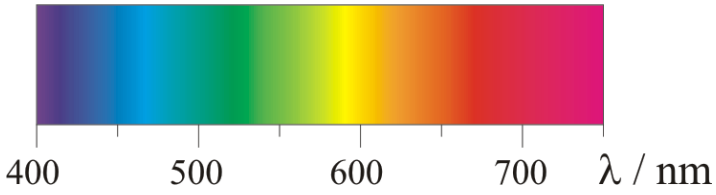
U današnjem svijetu komunikacija, brzi prijenos podataka postaje sve važniji za različite primjene, kao što su bežične mreže, IoT (eng. *Internet of Things*), pametni gradovi i druge tehnološke inovacije. U tom kontekstu, VLC (eng. *Visible light communication*) sustav postao je sve popularniji kao alternativa tradicionalnim bežičnim komunikacijskim tehnologijama. VLC sustav koristi vidljivo svjetlo, obično LED svjetla, za prijenos podataka. Osnovna ideja je da se podaci prenose moduliranjem intenziteta svjetla koje emitira LED svjetlo. Kada LED svjetlo brzo mijenja intenzitet, to se može detektirati i pretvoriti u digitalne podatke na prijemnoj strani.

U ovom radu istražit će se područje vidljive svjetlosti koristeći DHT22 senzor i LCD zaslon. Iskorištavanjem snage svjetlosti uspostavit će se komunikacijska veza između uređaja. Pomoću DHT22 senzora prikupljat će se podaci temperature i vlažnosti, koji će se koristiti za daljnju obradu. LCD zaslon služit će kao izlazni medij, omogućavajući vizualizaciju prikupljenih podataka u stvarnom vremenu. Kroz ovaj rad razmotrit će se glavne prednosti i ograničenja VLC sustava te će se objasniti potrebni dijelovi za dizajniranje samog sustava. Cilj rada je razumijevanje principa prijenosa podataka pomoću vidljive svjetlosti te praktičnu primjenu VLC sustava u praćenju temperature i vlažnosti.

2. KARAKTERISTIKE VLC SUSTAVA

VLC ima nekoliko ključnih karakteristika koje je čine posebnom kao tehnologijom komunikacije:

- Visoka propusnost
  - VLC koristi široku propusnost vidljive svjetlosti od 390 nm do 760 nm (≈400 THz) kao što je vidljivo na slici 2.1., omogućavajući brz prijenos podataka. Može postići brzine prijenosa od 1 GB/s.
- Ekonomičnost
  - VLC može biti energetski učinkovit, posebno kada se integrira s postojećom infrastrukturom osvjetljenja. Korištenjem LED svjetala za prijenos podataka, VLC ne zahtijeva dodatnu potrošnju energije izvan one koja se već koristi za osvjetljenje.
- Sigurnost
  - Teško je presresti VLC signale bez fizičkog pristupa putanji prijenosa, što ga čini sigurnim komunikacijskim medijem. Osim toga, VLC se može kombinirati s tehnikama šifriranja radi dodatnog poboljšanja sigurnosti podataka.
- Integracija
  - VLC se može bez problema integrirati s postojećim sustavima osvjetljenja.
- Imunitet na elektromagnetske smetnje
  - Za razliku od tehnologija komunikacije temeljenih na radiofrekvencijama, VLC je imun na elektromagnetske smetnje. Može raditi u okruženjima gdje se radio valovi mogu susresti s interferencijama ili su ograničeni, poput bolnica, zrakoplovnih kabina ili industrijskih postrojenja.



Slika 2.1. Vidljivi dio spektra [2]

Unatoč tim prednostima VLC tehnologija ima nekoliko nedostataka i ograničenja:

- Ovisnost o direktnoj optičkoj vidljivosti
- Osjetljivost na vanjsko osvjetljenje
- Visoka osjetljivost na interferencije
- Kratki domet.

Neovisno o nedostacima i ograničenjima, VLC ostaje zanimljiva tehnologija s potencijalom za specifične primjene gdje se njegove prednosti mogu optimalno iskoristiti. Njene karakteristike visoke propusnosti, sigurnosti, imuniteta na elektromagnetske smetnje i energetske učinkovitosti čine ju privlačnom za različite primjene. Daljnji razvoj i istraživanje VLC tehnologije mogu otkriti nove načine kako poboljšati performanse i proširiti mogućnosti u budućnosti.

3. HARDVERSKE KOMPONENTE VLC SUSTAVA

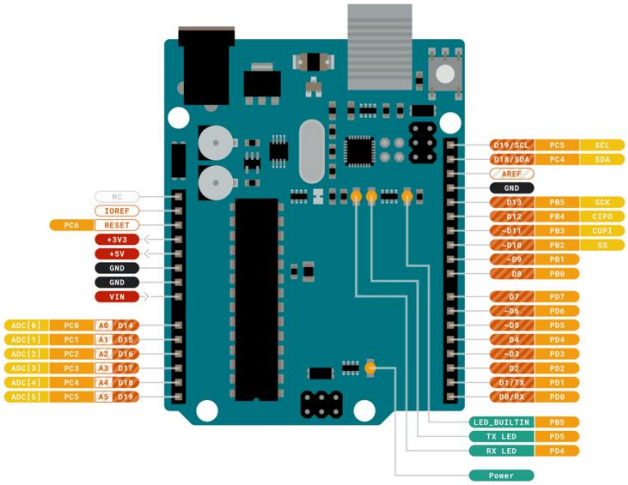
U ovom poglavlju opisani su svi korišteni dijelovi za realizaciju VLC sustava: Arduino Uno R3, prijemnik i predajnik. Bit će objašnjene sve funkcionalnosti i karakteristike pojedine komponente.

3.1. Arduino Uno R3

Arduino Uno R3 je opremljen ATmega328P mikrokontrolerom, koji ima 32KB flash memorije za pohranu programskog koda. Ova ploča također ima 2KB RAM memorije za privremeno pohranjivanje podataka. Pločica je temeljena na Atmega328P mikroprocesoru i na C++ programskom jeziku. Arduino Uno R3 radi na radnom naponu od 5 V, što ga čini kompatibilnim s većinom elektroničkih komponenti koje rade na istom naponu. Uno ima 14 digitalnih pinova, od kojih je 6 moguće koristiti kao PWM (eng. *Pulse Width Modulation*) izlazi. Također, ima 6 analognih ulaznih pinova za prikupljanje analognih vrijednosti s senzora poput temperature, svjetlosti ili pritiska. Tablica 3.1. prikazuje osnovne specifikacije Arduino Uno R3 pločice, a slika 3.1. pinout dijagram.

Tablica 3.1. Specifikacije Arduino Uno R3 [3]

MCU	ATmega328P
Radni napon	5 V
Ulazni napon	6V-20V
Brzina	16 MHz
Digitalni U/I pinovi	24 (6 za PWM)
Analogni ulazni pinovi	6
SRAM	2 KB
EEPROM	1 KB



Slika 3.1. Arduino UNO R3 pinout [3]

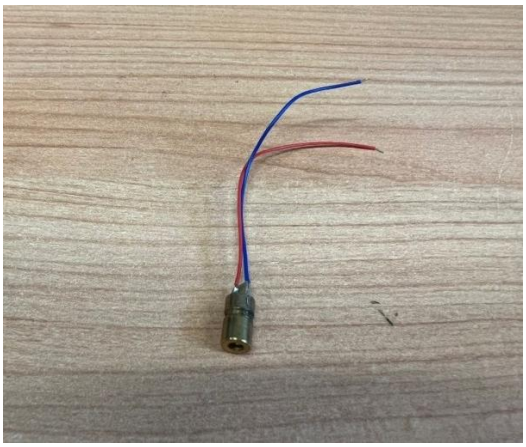
3.2. VLC predajnik

Svjetleća dioda ili LED (eng. *Light Emitting Diode*) je poluvodički elektronički element koji pretvara električni signal u optički (svjetlost). Propusno polarizirana svjetleća dioda emitira elektromagnetsko zračenje na način spontane emisije uzrokovane rekombinacijom nosilaca električnoga naboja (elektroluminiscencija).

Kod dizajniranja VLC predajnika obično se koristi poluvodički izvori svjetla kao što su LED diode i poluvodički laseri. Međutim, kada VLC predajnik mora raditi kao komunikacijski predajnik i uređaj za rasvjetu u isto vrijeme tada se koriste LED diode koje emitiraju bijelo svjetlo. Osim infracrvenoga zračenja u nevidljivome dijelu spektra, LED diode emitiraju svjetlost koja može biti crvene, žute, narančaste, zelene ili plave boje. U ovom radu koristit će se crveni LED laser, slika 3.2., snage 5 mW, a ostale karakteristike navedene su u tablici 3.2.

Tablica 3.2. Osnovne karakteristike LED lasera [4]

Karakteristika	Vrijednost
Valna duljina	650 nm
Snaga	5 mW
Promjer zrake	1.8~2 mm
Napon	5 V
Životni vijek	>3000 h



Slika 3.2. LED laser [4]

3.3. VLC prijemnik

Glavna komponenta prijemnog sustava je fotoosjetljivi poluvodički element, najčešće PIN ili APD (eng. *avalanche*) fotodioda. Ove diode rade u području valnih duljina od 190 nm do 1100 nm, tako da imaju dobar odziv u području vidljivog svjetla (380-780 nm). Kako bi se povećala efikasnost fotodetektora, između n- i p- tipa poluvodiča unosi se blago dopirani intristični sloj, a takve fotodiode se nazivaju PIN fotodiodama.

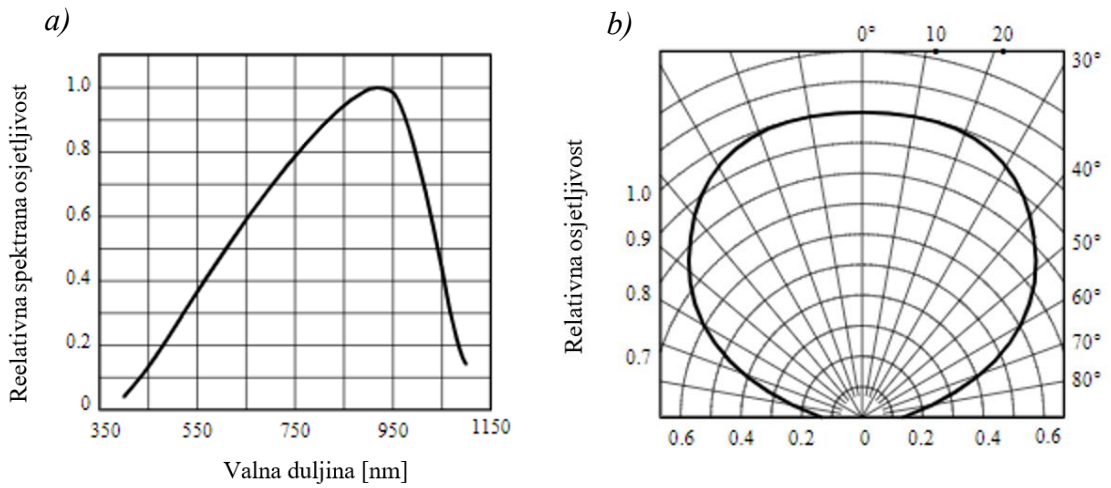
PIN diode se češće koriste zbog niže cijene, velike tolerancije na promjene temperature, imaju veće aktivno područje i više su prikladne za primjene u aplikacijama u kojima je prisutan veliki šum. S druge strane, APD fotodiode imaju veću osjetljivost čime se povećava komunikacijska brzina, što predstavlja veliku prednost iako sam sustav postaje znatno kompleksniji. U radu se koristi fotodioda BPW34, prikazana na slici 3.3. BPW34 je PIN fotodioda visoke brzine i osjetljivosti. Zbog svog prozirnog sloja, epoksija je osjetljiva na infracrveno zračenje.

Slika 3.3. BPW34 fotodioda [5]

Tablica 3.3. Karakteristike BPW34 fotodiode [5]

Parametar	Vrijednost	Jedinica
Dimenzije	5.4 x 4.3 x 3.2	mm
Veličina osjetljivog područja	7.5	mm^2
Snaga disipacije	215	mW
Radna temperatura	-55...+100	°C
Upadni kut svjetlosti	±65	°
Vršna valna duljina	900	nm
Raspon spektralne propusnosti	430...1100	nm

Fotodioda se u strujni krug spaja tako da je zaporno polarizirana. Kad je fotodioda neosvijetljena, njome teče vrlo mala tamna struja koju čini reverzna struja. Ta struja iznosi za silicijske fotodiode nekoliko nA, a za germanijske nekoliko mA.



Slika 3.4. a) Vršna osjetljivost fotodiode [5], b) Odnos relativne osjetljivosti i upadnog kuta zrake [5]

Budući da se koristi crveni LED laser valne duljine 650 nm, fotodioda BPW34 dovoljno je osjetljiva, iako joj je vršna osjetljivost tek na oko 900 nm, što se može vidjeti na slici 3.4. a).

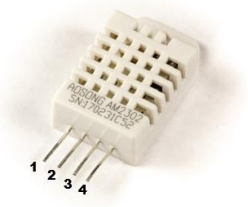
3.4. DHT22

DHT22 senzor je popularan digitalni senzor koji se koristi za mjerenje temperature i vlage u okolini. Ovaj senzor je pouzdan, precizan i lako se koristi, što ga čini široko rasprostranjenim u različitim područjima, od kućanstava do industrijskih aplikacija. Jedna od glavnih prednosti DHT22 senzora je njegova visoka preciznost. Senzor ima vrlo malu pogrešku mjerenja temperature (± 0.5 °C) i vlažnosti (± 2 %), a ostale karakteristike nalaze se u tablici 3.4. To znači da možemo pouzdano dobiti točne očitane vrijednosti, što je ključno za mnoge primjene u kojima su točni podaci neophodni, poput sustava kontrole klime ili laboratorijskih eksperimenata. Još jedna važna značajka DHT22 senzora je njegova digitalna komunikacija. Senzor koristi jednostavan protokol komunikacije koji omogućuje lako povezivanje s mikrokontrolerima i drugim elektroničkim uređajima. Ovo olakšava integraciju senzora u različite projekte i sustave te omogućuje jednostavno očitavanje i obradu podataka. U ovom radu, DHT22 senzor, koristit će se za mjerenje temperature i vlage te će se isti podaci kodirati i odašiljati pomoću LED lasera te ispisivati na LCD zaslonu.

Tablica 3.4. Karakteristike DHT22 senzora [6]

Karakteristika	Vrijednost
Raspon temperature	-40 °C do +80 °C
Pogreška temperature	± 0.5°C
Raspon vlage	0 % do 100 %
Pogreška vlage	± 2%
Vrsta senzora	Digitalni
Protokol komunikacije	Jednostavan digitalni protokol
Napajanje	3.3 V do 5.5 V
Tip izlaza	Digitalni
Dimenzije	15.5 mm x 12 mm x 5.5 mm

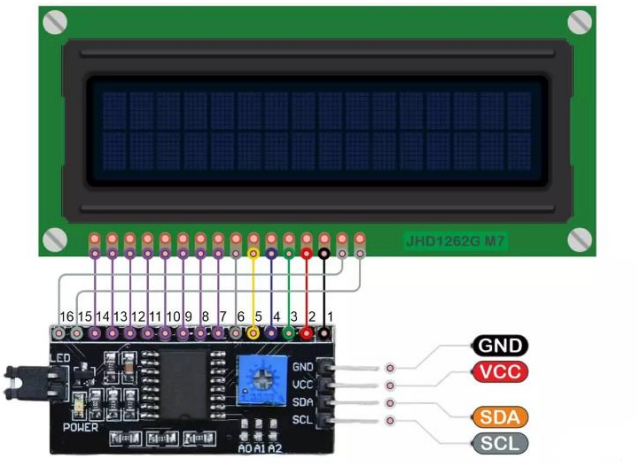
DHT22 ima 4 pina: pin 1 je VCC i spaja se na 3.3 V ili 5 V, pin 2 spaja se na digitalni pin arduina (DATA), pin 3 (NC) se ne koristi i pin 4 (GND) spaja se na masu, kao što je vidljivo na slici 3.5.



Slika 3.5. DHT22 senzor [6]

3.5. LCD zaslon s I2C sučeljem

LCD zasloni su popularni prikazni uređaji koji se često koriste u elektronici i projektiranju sustava radi vizualnog prikaza informacija. I2C (Inter-Integrated Circuit) je serijski komunikacijski protokol koji omogućuje jednostavan prijenos podataka između mikrokontrolera i perifernih uređaja. Korištenje LCD zaslona s I2C sučeljem donosi nekoliko prednosti. Prvo, smanjuje potreban broj pinova na mikrokontroleru za upravljanje zaslonom. Umjesto korištenja 6 ili više pinova za upravljanje klasičnim LCD zaslonom, I2C sučelje omogućuje korištenje samo 2 pina (SDA i SCL) za prijenos podataka i upravljanje zaslonom, kao što je vidljivo na slici 3.6. Druga prednost je jednostavnost integracije. Mikrokontroler koji podržava I2C sučelje može lako komunicirati s LCD zaslonom putem I2C protokola, bez potrebe za složenim i vremenski zahtjevnim kodiranjem.



Slika 3.6. Pinout LCD zaslona sa I2C sučeljem [7]

Da bi komunikacija bila moguća, potrebno je pronaći I2C adresu. Najprije se spoji I2C sučelje na Arduino Uno R3, tako da VCC spojimo na 5 V, GND na GND, SCL na analogni pin A5 i SDA na analogni pin A4. Učitavanjem već gotovog koda iz Arduino mape (*File > Examples > Wire > i2c_scanner),* na Serial monitoru ispsat će se tražena adresa u heksadecimalnom obliku. U ovom slučaju adresa I2C sučelja je 0x27, što je vidljivo na primjeru ispod.

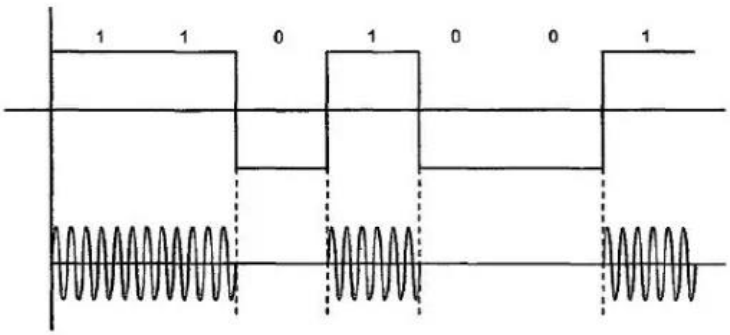
```
Skeniranje...
I2C uređaj pronađen na adresi 0x27  !
Gotovo
```

4. MODULACIJSKE METODE I KODIRANJE

U navedenom poglavlju objašnjene su modulacijske metode i kodiranje koje je zaslužno za postizanje veće brzine i sigurnog slanja podataka vidljivim svjetlom.

4.1. OOK modulacija

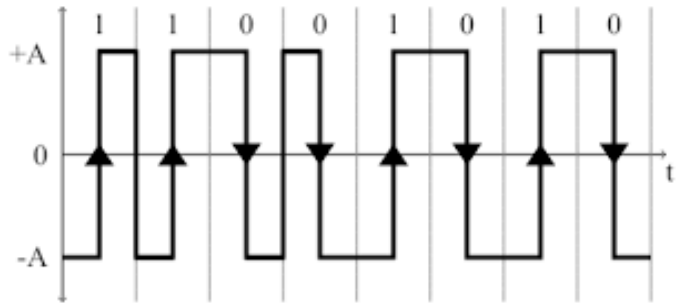
OOK (eng. *On–off keying*) modulacija je najjednostavnija modulacijska shema koja se primjenjuje u VLC sustavima pri čemu su LED diode uključene (ON) ili isključene (OFF) ovisno o tome je li podatkovni bit u stanju "1" ili "0". Stanje logičke jedinice u signalu informacije predstavlja se prisutnošću signala nosioca, a stanje logičke nule u signalu informacije predstavlja se odsustvom signala nosioca. Takav tip modulacije odgovarao bi amplitudnoj modulaciji s indeksom modulacije m = 1. Primjer je prikazan na slici 4.1.



Slika 4.1. Prikaz signala OOK modulacije [8]

4.2. Manchester kodiranje

Manchester linijsko kodiranje uvijek ima prijelaz na sredini svakog bitnog razdoblja. Porast razine signala u sredini bita predstavlja logičko stanje „1“, a pad razine signala logičko stanje „0“. Signal Manchester linijskog koda nema istosmjernu komponentu. Ovaj postupak linijskog kodiranja povećava brzinu prijenosa signala, jer se umjesto jednog bita u jedinici vremena šalju dva bita. Primjer je prikazan na slici 4.2.



Slika 4.2. Prikaz signala Manchester kodiranja [9]

Pravila Manchester kodiranja su sljedeća:

- Ako je vrijednost bita 0, signal mijenja stanje s visokog na nisko u sredini vremenskog intervala bita.
- Ako je vrijednost bita 1, signal mijenja stanje s niskog na visoko u sredini vremenskog intervala bita.

Prednost Manchester kodiranja je da osigurava ravnotežu između broja prijelaza signala i vremena potrebnog za prijenos podataka. Također, osigurava da postoji dovoljno prijelaza signala za održavanje sinkronizacije između predajnika i prijemnika.

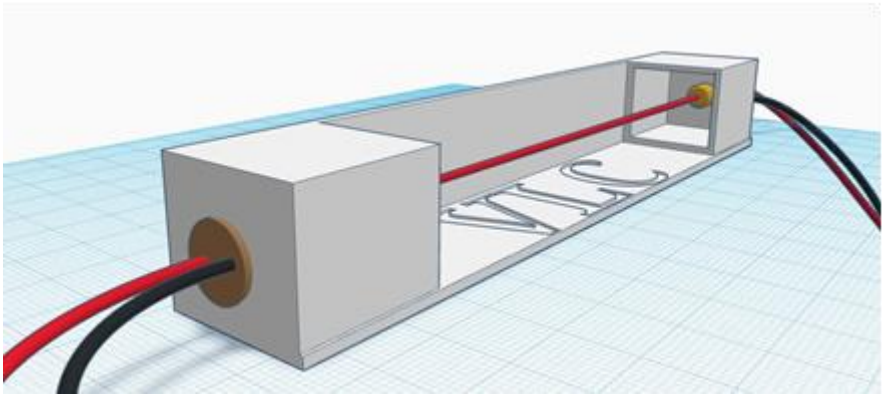
Jedan od nedostataka Manchester kodiranja je duplo veća širina pojasa u usporedbi s klasičnim binarnim kodiranjem. To znači da je potrebna veća propusnost za prijenos istog broja podataka.

5. FIZIČKA IZVEDBA SUSTAVA

U ovom poglavlju prikazat će se fizički sloj VLC sustava, električne sheme te način postizanja direktne optičke vidljivosti.

5.1. VLC kanal

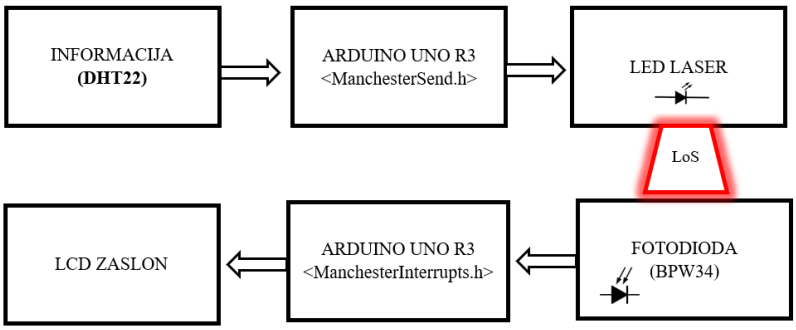
VLC kanal je komunikacijski kanal koji koristi vidljivu svjetlost kao nositelja signala za prijenos podataka. LoS (eng. *Line of Sight*) kod VLC komunikacije odnosi se na izravnu optičku povezanost između predajnika i prijemnika koji koriste vidljivu svjetlost za prijenos podataka. Prepreke kao što su zidovi, pregrade ili drugi objekti mogu blokirati put svjetlosnom signalu, što može uzrokovati slabljenje ili prekid veze. Stoga, kako bi se osigurala dobra LoS veza, predajnik i prijemnik trebaju biti postavljeni tako da su međusobno vidljivi bez ikakvih prepreka. U ovom radu VLC kanal realiziran je najprije u besplatnom online alatu za 3D dizajniranje (*TinkerCad*), kako bi osigurali dobru optičku povezanost bez izazivanja smetnji. Model je prikazan na slici 5.1.



Slika 5.1. 3D model VLC kanala [10]

5.2. Fizički sloj VLC sustava

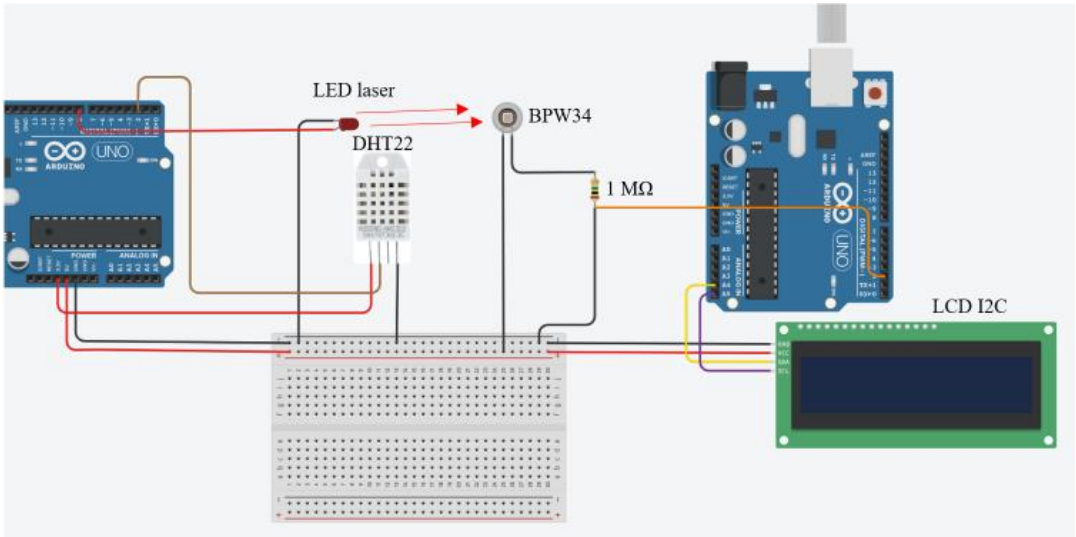
Fizički sloj VLC komunikacijskog sustava sastoji se od predajnika, prijemnika i komunikacijskog kanala. Izvor informacije u ovom slučaju čine podaci o temperaturi i vlazi sa DHT22 senzora. Manchester biblioteke omogućuju slanje digitalne informacije preko lasera do prijemne strane. U ovom slučaju koriste se ManchesterSend.h i ManchesterInterrupts.h biblioteke. Modulirana informacija predstavljena je kao niz impulsa, u logičkom smislu niza nula i jedinica. LED predajnik brzim paljenjem i gašenjem predstavlja nas binarni niz. Između predajnika i prijemnika potrebna je optička vidljivost, kao što je prikazano na slici 5.2.



Slika 5.2. Blok dijagram VLC sustava

5.3. Postupak spajanja

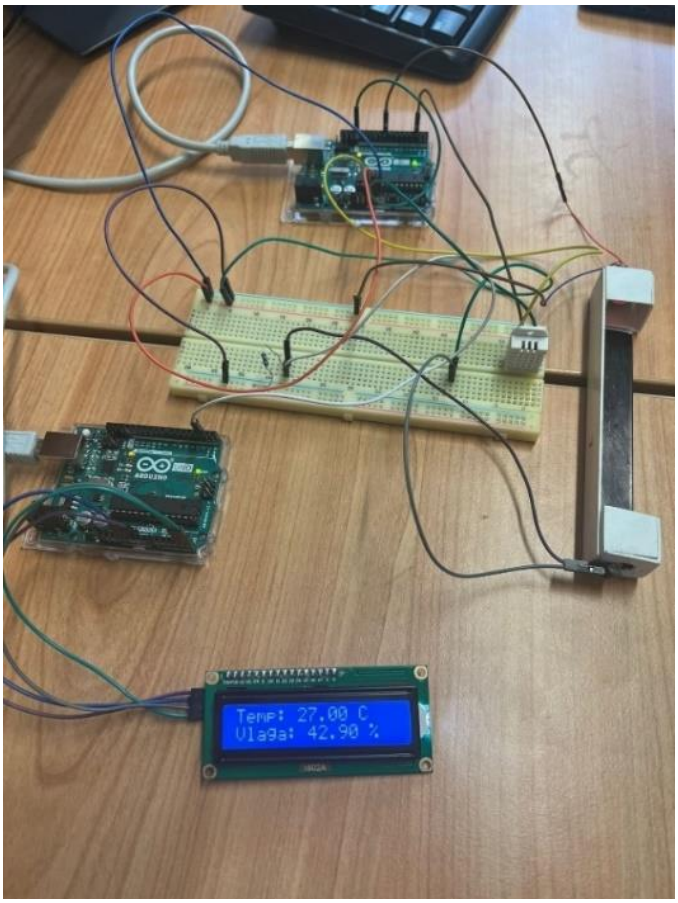
U ovom dijelu opisat će se postupak povezivanja komponenti kako bi se omogućilo slanje podataka temperature i vlage putem lasera te njihov prikaz na LCD zaslonu. Ovaj sustav koristi Arduino Uno R3 mikrokontroler, DHT22 senzor za mjerenje temperature i vlage, LED laser kao predajnik, fotodiodu kao prijemnik te LCD zaslon s I2C sučeljem za prikaz podataka.



Slika 5.3. Shema spajanja sustava [10]

Na lijevoj strani prikazana je Arduino pločica koja predstavlja VLC predajnik. LED laser spojen je na digitalni pin D8 na koji se šalju kodirani podaci i na GND. DHT22 senzor ima četiri pina od kojih se koriste samo tri i oni su spojeni prema shemi na 3.3 V, GND i digitalni pin D2.

Na prijemnoj strani nalazi se fotodioda BPW34 i LCD sa I2C sučeljem koji su povezani na drugi Arduino prema shemi na slici 5.3. Fotodioda je zaporno polarizirana te se tako i spaja, katoda se spaja na 5 V, a anoda na otpornik od 1 MΩ i dalje na GND. Signalizacijski pin za fotodiodu je D2. LCD koristi četiri pina i oni su spojeni na 5 V, GND, A4 i A5.



Slika 5.4. Prikaz podataka temperature i vlage na LCD zaslonu

Uz pravilno učitani kod, Arduino će neprekidno očitavati temperaturu i vlagu putem DHT22 senzora, odašiljati podatke kao niz bitova preko lasera te će primljeni podaci biti prikazani na LCD zaslonu na prijemnoj strani. Prije ispisa podataka na LCD-u prikazat će se tekst „*Trenutak molim*“ te nakon dvije sekunde podaci će se prikazati na zaslonu. Ako dođe do prekida VLC kanala na duže od tri sekunde, pojavit će se poruka „*Prekid VLC kanala*“. Ukoliko se ponovno uspostavi optička vidljivost između lasera i fotodiode na zaslonu će biti prikazane vrijednosti temperature i vlage.

6. PROGRAMSKA PODRŠKA

U ovom poglavlju pobliže su opisani programski kodovi prijemnika i predajnika. Implementacija koda predajnika i prijemnika omogućuje komunikaciju između mikrokontrolera i drugih uređaja koristeći Manchester kodiranje signala. Ovaj način kodiranja omogućuje pouzdanu i efikasnu komunikaciju između uređaja koji podržavaju istu metodu kodiranja.

6.1. Programski kod predajnika

Budući da funkcija sendString() može slati samo string podatak, potrebno je pretvoriti decimalne znamenke u string kako bi podaci bili pogodni za slanje. Lokalna funkcija sendTemperatureAndHumidity(), koja prima dva float parametra, radi konverzaciju iz floata u string, čime je omogućeno slanje podataka laserom.

```
void sendTemperatureAndHumidity(float temperature, float humidity)
{
  String temperatureString = String(temperature, 2);
// Konverzija temperature u string s dvije decimalne znamenke
  String humidityString = String(humidity, 2);
// Konverzija vlage u string s dvije decimalne znamenke
  String data = temperatureString + "," + humidityString;
// Spajanje temperature i vlage u jedan string
  const char* dataToSend = data.c_str();
// Pretvaranje stringa u char niz
  sendString(dataToSend);  // Slanje podataka
}
```

Funkcija sendString() pomoću pokazivača *s odašilje znak po znak. Također, stavlja novi redak kako bi se označio kraj niza podataka.

```
void sendString(const char* s)
{
  while (*s)
  ManchesterSend::write(*s++);
// Slanje svakog znaka iz char niza
  ManchesterSend::write('\n');
// Dodavanje znaka za novi red kako bi označili kraj podataka
}
```

6.2. Programski kod prijemnika

U programskom kodu prijemnika potrebno je deklarirati spremnik receivedData[64] za pohranu senzorskih podataka i lakšu obradu. Funkcija atof() vraća podatke u prvobitan float oblik.

```
float temperature = atof(temperatureStr);
float humidity = atof(humidityStr);
```

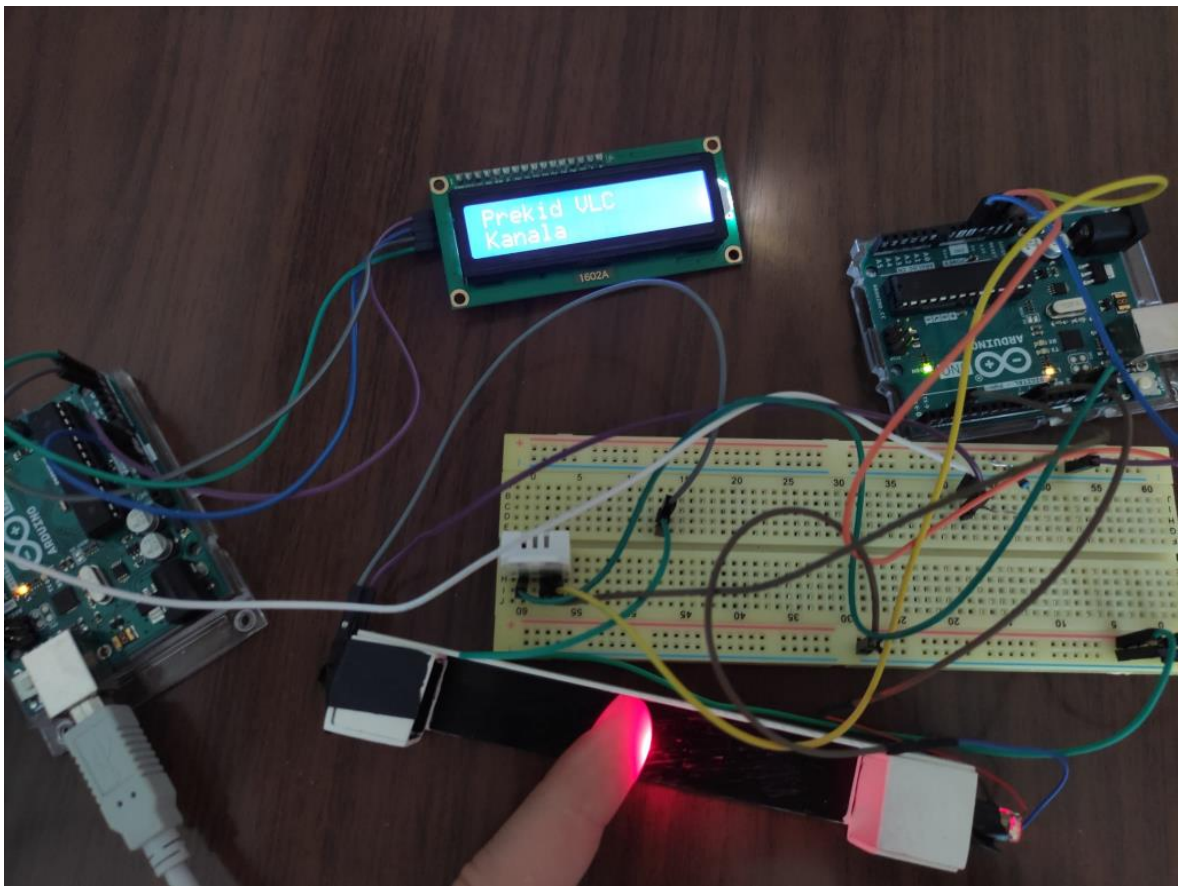
Za prikaz podataka na LCD zaslonu zadužena je funkcija koja prima dva float parametra, displayTemperatureAndHumidity(). U njoj su sadržane funkcije iz LiquidCrystal_I2C.h biblioteke, kao što su lcd.serCursor() (postavljanje kursora) i lcd.print() (funkcija za ispis).

```
void displayTemperatureAndHumidity(float temperature, float humidity)
{
  lcd.setCursor(0, 0);
  // Postavljanje kursora na prvi red LCD ekrana
  lcd.print("Temp: ");
  // Ispis teksta "Temp: "
  lcd.print(temperature, 2);
  // Ispis temperature s dvije decimalne znamenke
  lcd.print(" C    ");
  // Ispis oznake za Celzijus
  lcd.setCursor(0, 1);
  // Postavljanje kursora na drugi red LCD ekrana
  lcd.print("Vlaga: ");
  // Ispis teksta "Vlaga: "
  lcd.print(humidity, 2);
  // Ispis vlage s dvije decimalne znamenke
  lcd.print(" %   ");
  // Ispis oznake za postotak
}
```

U slučaju da dođe do prekida optičke vidljivosti između prijemnika i predajnika na tri sekunde, ispisuje se poruka „*Prekid VLC kanala*“, kao što je vidljivo na slici 6.1.

```
  if (!laserDetected && noDataTimeout && !newData && hasData)
//Provjeravamo je li prošlo više od 3 sekunde od zadnjeg primljenog podatka, a
nema novih podataka ni detekcije lasera.
  {
    lcd.setCursor(0, 0);
    lcd.print("Prekid VLC");
    lcd.setCursor(0, 1);
    lcd.print("Kanala");
```

```
    hasData = false;
    clearBuffer();
//Ako postoje dostupni senzorski podaci, briše se sadržaj LCD ekrana, ispisuje se poruka "Prekid VLC Kanala", postava se oznaka "hasData" na "false" i briše se sadržaj spremnika primljenih podataka.
  }
```



Slika 6.1. Ispis poruke "*Prekid VLC kanala,,* u slučaju prekida optičke vidljivosti

7. ZAKLJUČAK

U radu su prezentirani osnovni principi te karakteristike sustava koji koristi komunikaciju vidljivim svjetlom. Sustav se temelji na OOK modulaciji i Manchester kodiranju. Jedna od glavnih prednosti je što se kao komunikacijski predajnik može koristiti LED rasvjeta što doprinosi velikoj ekonomičnosti. Ovakav sustav bezopasan je za ljudsko zdravlje, nisu potrebne dozvole za korištenje, nema interferencije s radio valovima pa se mogu koristiti u uvjetima gdje se radio uređaji ne mogu koristiti, kao i velika informacijska sigurnost. Međutim, važno je napomenuti da VLC sustav ima neka ograničenja. Na primjer, domet prijenosa podataka putem vidljive svjetlosti može biti ograničen na nekoliko metara te iziskuje stalnu optičku vidljivost koju je u nekim slučajevima teško postići.

Sustav koji koristi DHT22 senzor i LCD zaslon pokazuje veliki potencijal za primjenu u različitim područjima. Kombinacija VLC tehnologije s DHT22 senzorom omogućava prijenos podataka putem vidljive svjetlosti, dok LCD zaslon omogućava prikazivanje informacija na kraju prijemne strane. Daljnji razvoj i istraživanje mogu dovesti do šire primjene i unaprjeđenja sustava s DHT22 senzorom i LCD zaslonom u različitim industrijama i područjima primjene.

LITERATURA

[1] Kovačević. T., Vukšić. M., Džaja. B., Komunikacija vidljivim svjetlom: Osnovni koncepti, Fizički sloj i primjene, 24.11.2014.

[2] Vidljivi dio elektromagnetskog spektra, slika, https://glossary.periodni.com/glosar.php?hr=vidljivo+zra%C4%8Denje, 29.06.2022.

[3] Arduino UNO R3, datasheet proizvoda, https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf, 12.06.2023.

[4] Soldered, Laser 650 nm (red) 5 mW, datasheet proizvoda, https://soldered.com/product/laser-650nmred-5mw-3v/, 21.03.2012.

[5] Vishay Semiconductors, BPW34, BPW34S, datasheet proizvoda, Datasheet for BPW34 Vishay Photodiodes | Octopart, 23.08.2011.

[6] DHT22 senzor, datasheet proizvoda, https://www.mouser.com/datasheet/2/737/dht-932870.pdf, 13.12.2021

[7] LCD 16x2, pinout slika, //www.techtonions.com/jumbo-lcd-16x2-tutorial/, 07.09.2021.

[8] OOK modulacija, slika, https://www.scribd.com/document/450766327/TEHNIKE-DIGITALNE-MODULACIJE, 2018.

[9] F. Jeleč, Linijski kodovi u ADSL sustavima, slika Manchester kodiranja, https://core.ac.uk/download/pdf/197864837.pdf, Osijek 2017.

[10] TinkerCad, 3D model, https://www.tinkercad.com/, 6.2023.

POPIS SLIKA

POPIS TABLICA

PRILOG 1 – Programski kod predajnika

```
#include <ManchesterSend.h>
// Uključivanje biblioteke za Manchester kodiranje slanja podataka
#include <DHT.h>      // Uključivanje biblioteke za DHT senzor
#define DHTPIN 2      // Pin DHT22 senzora
#define DHTTYPE DHT22 // Tip DHT senzora

DHT dht(DHTPIN, DHTTYPE);
// Inicijalizacija objekta DHT sa zadanim pinom i tipom
void setup()
{
  ManchesterSend::begin(8);
// Inicijalizacija slanja Manchester koda na pinu D8
  Serial.begin(115200);
// Inicijalizacija serijske komunikacije
  dht.begin();
// Inicijalizacija DHT senzora
}
void sendTemperatureAndHumidity(float temperature, float humidity)
{
  String temperatureString = String(temperature, 2);
// Konverzija temperature u string s dvije decimalne znamenke
  String humidityString = String(humidity, 2);
// Konverzija vlage u string s dvije decimalne znamenke
  String data = temperatureString + "," + humidityString;
// Spajanje temperature i vlage u jedan string
  const char* dataToSend = data.c_str();
// Pretvaranje stringa u char niz
  sendString(dataToSend);  // Slanje podataka
}
void sendString(const char* s)
{
  while (*s)
    ManchesterSend::write(*s++);  // Slanje svakog znaka iz char niza
  ManchesterSend::write('\n');
// Dodavanje znaka za novi red kako bi označili kraj podataka
}
void loop()
{
  float temperature = dht.readTemperature(); // Čitanje temperature s DHT
senzora
  float humidity = dht.readHumidity(); // Čitanje vlage s DHT senzora
  sendTemperatureAndHumidity(temperature, humidity);
// Slanje temperature i vlage
  delay(2500);
// Čekanje 2 sekunde prije slanja sljedeće temperature i vlage
}
```

PRILOG 2 – Programski kod prijemnika

```
#include <ManchesterInterrupts.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
// Uključivanje biblioteke za I2C, LCD i Manchester komunikaciju
LiquidCrystal_I2C lcd(0x27, 16, 2); // Inicijalizacija LCD zaslona
char receivedData[64];  // Spremnik za pohranu primljenih podataka
int dataIndex = 0;     // Indeks za praćenje pozicije u spremniku
bool hasData = false;
// Varijabla za praćenje dostupnosti senzorskih podataka
unsigned long lastDataTime = 0;
// Varijabla za praćenje vremenske oznake zadnjeg primljenog podatka
void setup()
{
  ManchesterInterrupts::dataPin = 2;
  // Postavljanje pina za Manchester prekide na pin 2
  attachInterrupt(0, ManchesterInterrupts::isr, CHANGE);
  // Povezivanje prekida s funkcijom isr iz biblioteke
  Wire.begin();
  // Inicijalizacija I2C komunikacije
  lcd.begin(16, 2);
  // Inicijalizacija LCD ekrana sa 16 stupaca i 2 retka
  lcd.backlight();
  // Uključivanje pozadinskog osvjetljenja LCD ekrana
  lcd.print("Trenutak molim..");
  // Ispis teksta na LCD-u
  lcd.setCursor(0, 1);
  delay(2000);
  lcd.clear();
  // Brisanje LCD ekrana
}
void clearBuffer()
{
  memset(receivedData, 0, sizeof(receivedData));
  // Brisanje sadržaja spremnika primljenih podataka
  dataIndex = 0;  // Resetiranje indeksa spremnika
}
void displayTemperatureAndHumidity(float temperature, float humidity)
{
  lcd.setCursor(0, 0);
  // Postavljanje kursora na prvi red LCD ekrana
  lcd.print("Temp: ");
  // Ispis teksta "Temp: "
  lcd.print(temperature, 2);
  // Ispis temperature s dvije decimalne znamenke
  lcd.print(" C    ");
  // Ispis oznake za Celzijus
```

```
  lcd.setCursor(0, 1);
  // Postavljanje kursora na drugi red LCD ekrana
  lcd.print("Vlaga: ");
  // Ispis teksta "Vlaga: "
  lcd.print(humidity, 2);
  // Ispis vlage s dvije decimalne znamenke
  lcd.print(" %   ");
  // Ispis oznake za postotak
}
void processReceivedData()
{
  if (receivedData[dataIndex - 1] == '\n')
  {
    receivedData[dataIndex - 1] = '\0';
// Ako je posljednji primljeni znak u spremniku jednak '\n', zamjenjuje se s
'\0' kako bi se završio niz.
    char* temperatureStr = strtok(receivedData, ",");
    char* humidityStr = strtok(NULL, ",");
// Funkcija "strtok" za izdvajanje vrijednosti temperature i vlage iz
primljenih podataka.
    if (temperatureStr != NULL && humidityStr != NULL)
    {
      float temperature = atof(temperatureStr);
      float humidity = atof(humidityStr);
//Pretvaramo u float vrijednosti koristeći funkciju "atof".
      displayTemperatureAndHumidity(temperature, humidity);
      hasData = true;
// Postavljanje oznake da su senzorski podaci dostupni.
    }
    clearBuffer();
  }
}

void loop()
{
  bool laserDetected = ManchesterInterrupts::available();
  unsigned long currentTime = millis();
  bool newData = false;
//U petlji se provjerava da li  je detektiran laser pomoću funkcije "available()".
  if (laserDetected)
  {
    char receivedChar = ManchesterInterrupts::read();
    Serial.print(receivedChar);
    receivedData[dataIndex++] = receivedChar;
//Ako je detektiran laser, čita primljeni znak i pohranjuje ga u spremnik
primljenih podataka "receivedData".
    if (dataIndex >= sizeof(receivedData))
    {
```

```
      clearBuffer();
//Ako je spremnik pun, briše njegov sadržaj.
    }
    processReceivedData();
    newData = true;
    lastDataTime = currentTime;
//Postavlja oznaku "newData" na "true" i bilježi trenutno vrijeme kao posljednje
vrijeme primljenog podatka.
  }
  bool noDataTimeout = (currentTime - lastDataTime >= 3000);
  if (!laserDetected && noDataTimeout && !newData && hasData)
//Provjerava da li je prošlo više od 3 sekunde od zadnjeg primljenog podatka, a
nema novih podataka ni detekcije lasera.
  {
    lcd.setCursor(0, 0);
    lcd.print("Prekid VLC         ");
    lcd.setCursor(0, 1);
    lcd.print("Kanala             ");
    hasData = false;
    clearBuffer();
//Ako postoje dostupni senzorski podaci, briše sadržaj LCD ekrana, ispisuje poruku
"Prekid VLC" i "Kanala", postavlja oznaku "hasData" na "false" i briše sadržaj
spremnika primljenih podataka.
  }
}
```

PRILOG 3 – Programski kod za skeniranje I2C adrese

```
#include <Wire.h>

void setup() {
  Wire.begin();
  Serial.begin(9600);
  while (!Serial);
// Čekanje na početak serijske komunikacije
  Serial.println("\nI2C Scanner");
// Ispisuje poruku koja označava skeniranje I2C uređaja
}

void loop() {
  int nDevices = 0;
  Serial.println("Skeniranje...");
// Ispisuje poruku koja označava da je skeniranje u tijeku
  for (byte address = 1; address < 127; ++address) {
// Prolazi kroz sve adrese
    Wire.beginTransmission(address);
// Pokreće I2C prijenos prema adresi
    byte error = Wire.endTransmission();
// Provjerava ima li pogrešaka tijekom prijenosa
    if (error == 0) {
// Ako nije došlo do pogreške
      Serial.print("I2C uređaj pronađen na adresi 0x");
// Ispisuje poruku koja označava da je uređaj pronađen
      if (address < 16) {
        Serial.print("0");
// Ispisuje vodeću nulu za adrese manje od 16
      }
      Serial.print(address, HEX);
// Ispisuje adresu u heksadecimalnom formatu
      Serial.println("  !");
// Ispisuje „!“ koji označava uspješno otkrivanje
      ++nDevices; // Povećaj brojač uređaja
    } else if (error == 4) {
// Ako se dogodila pogreška
      Serial.print("Nepoznata pogreška na adresi 0x");
// Ispisuje poruku koja označava nepoznatu pogrešku
      if (address < 16) {
        Serial.print("0");
// Ispisuje vodeću nulu za adrese manje od 16
      }
      Serial.println(address, HEX);
// Ispisuje adresu u heksadecimalnom formatu
    }
  }
```

```
  if (nDevices == 0) {
    Serial.println("Nema I2C uređaja\n");
// Ispisuje poruku koja označava da nisu pronađeni uređaji
  } else {
    Serial.println("Gotovo\n");
// Ispisuje poruku koja označava završetak skeniranja
  }
  delay(5000);
// Stanka od 5 sekundi prije početka sljedećeg skeniranja
}
```

PRILOG 4 – ManchesterSend biblioteka

```
// ManchesterSend.cpp
// Napravio Nick Gammon, 22/01/12.

#include <ManchesterSend.h> // uključivanje <ManchesterSend.h> biblioteke
namespace ManchesterSend {

byte sendPin; // deklaracija pina za slanje
volatile unsigned long outData; // trenutni podaci za slanje
volatile byte bitsToGo; // brojač bitova
byte currentBit; // trenutni bit
boolean sentFirst; // da li je poslan prvi impuls

void begin (const byte pin)
{
  sendPin = pin;
  digitalWrite (sendPin, HIGH);
  pinMode (sendPin, OUTPUT);
  bitsToGo = 0;
  TCCR2A = 0;
  TCCR2B = 0;
  TCCR2A = _BV (WGM21) ; // CTC mod
  OCR2A = 124; // broji do 125 (počevši od nule!)
  TIMSK2 = _BV (OCIE2A);
  TCNT2 = 0;
  GTCCR = _BV (PSRASY);
  TCCR2B = _BV (CS22) ;
} // kraj ManchesterSend::begin
// Definicija funkcije "begin" koja se koristi za inicijalizaciju slanja podataka.
Postavlja se izlazni pin na visoku razinu, postavlja se kao izlazni pin i
konfigurira
// se Timer 2 za prekide svakih 500 µS.

void write (const byte data)
{
  outData = (((unsigned long) data) << SYNC_BITS) | (1UL << (SYNC_BITS - 1));
  bitsToGo = SYNC_BITS + 8;
  while (bitsToGo)
  {}
  delay (1);
} // kraj ManchesterSend::write
// Definicija funkcije "write" koja se koristi za slanje jednog bajta podataka.
Bajt podataka se pretvara u Manchester format, pohranjuje u "outData" varijablu,
postavlja broj preostalih bitova i blokira izvršavanje dok se svi bitovi ne
pošalju. Nakon toga slijedi kratko kašnjenje da bi se omogućio posljednji puls
da ode, ako je bio nizak (LOW).
// 1 je 01
```

```
// 0 je 10

void isr ()
{
  if (bitsToGo == 0)
  {
    digitalWrite (sendPin, HIGH);
    return;
  }
  if (sentFirst)
  {
    if (currentBit == 0)
      digitalWrite (sendPin, LOW);
    else
      digitalWrite (sendPin, HIGH);
    sentFirst = false;
    bitsToGo--;
    return;
  }
  currentBit = outData & 1;
  outData = outData >> 1;
  if (currentBit == 0)
    digitalWrite (sendPin, HIGH);
  else
    digitalWrite (sendPin, LOW);
  sentFirst = true;
} // kraj ManchesterSend::isr
// Definicija ISR (Interrupt Service Routine) funkcije "isr" koja se izvršava svaki put kada Timer 2 generira prekid. Ova funkcija je odgovorna za slanje Manchester signala na izlazni pin.
} // kraj namespace ManchesterSend


ISR (TIMER2_COMPA_vect)
{
  ManchesterSend::isr ();
} // kraj TIMER2_COMPA_vect
// Definicija prekidne rutine za Timer 2 koja poziva 'isr' funkciju iz 'ManchesterSend' namespace-a kada se generira prekid.
```

PRILOG 5 – ManchesterInterrupts biblioteka

```
//  ManchesterInterrupts.cpp
//  Napravio Nick Gammon, 22/01/12.

#include <ManchesterInterrupts.h>
// Uključivanje biblioteke <ManchesterInterrupts.h>
namespace ManchesterInterrupts {
  byte manchesterState = IDLE;
// Trenutno stanje primanja podataka 'IDLE' (mirovanje)
  boolean gotFirst;
// Varijabla 'gotFirst' je postavljena na 'true' ako je primljen prvi od dva
impulsa za svaki podatkovni bit.
  byte dataByte;
  byte bitCount;
// Varijable 'dataByte' i 'bitCount' se koriste za praćenje trenutnog primljenog
bajta i broja primljenih bitova unutar tog bajta.
  byte zeroCount;
// broji broj primljenih sinkronizacijskih bitova (0) za vrijeme primitka
  unsigned long lastBitChange;
// vrijeme posljednje promjene signala (ms)
  byte dataPin;
// pin za primanje podataka
  volatile char buf [32];
  volatile char inpoint, outpoint;
// Varijable 'buf', 'inpoint' i 'outpoint' se koriste za pohranjivanje primljenih
bajtova u spremniku. 'buf' je spremnik za pohranu primljenih bajtova, 'inpoint'
označava poziciju za sljedeći unos u spremnik, a 'outpoint' označava poziciju za
sljedeći čitanje iz spremnika.

  void emitBit (const byte b)
  {
  if (manchesterState != IDLE && !gotFirst)
    {
    gotFirst = true;
    return;
    }
  gotFirst = false;
  switch (manchesterState)
    {
      case IDLE:

        if (b == 0)
          {
          manchesterState = PREAMBLE;
          zeroCount = 1;
```

```
        }
      return;
    case PREAMBLE:
      if (b == 1)
        {
        if (zeroCount < MIN_ZEROES)
          {
          manchesterState = IDLE;
          return;
          }
        manchesterState = DATA;
        bitCount = 0;
        return;
        }
      zeroCount++;
      return;
    case DATA:
      dataByte >>= 1;
      if (b)
        dataByte |= 0x80;
      bitCount++;
      if (bitCount == 8)
        {
        manchesterState = IDLE;
        char next = inpoint + 1;
        if (next >= sizeof buf)
          next = 0;
        if (next == outpoint)
          return;
        buf [inpoint] = dataByte;
        inpoint = next;
        return;
        }
      break;
  }

} // kraj ManchesterInterrupts::emitBit
// U ovom dijelu koda se provjerava trenutno stanje 'manchesterState' i izvršava
se odgovarajuća logika ovisno o stanju.
// Moguće je dobiti sljedeće:
// 10  -> tranzicija s jednim impulsom: emitira se bit 0
// 01  -> tranzicija s jednim impulsom: emitira se bit 1
// 001 -> tranzicija s dvostrukim impulsom: emitira se bit 0, zatim bit 1
// 110 -> tranzicija s dvostrukim impulsom: emitira se bit 1, zatim bit 0
  void isr ()
    {
    byte val = digitalRead (dataPin);
    unsigned long timeDiff = micros () - lastBitChange;
```

```
    lastBitChange = micros ();
    if (timeDiff >= LOW_WIDTH && timeDiff <= HIGH_WIDTH)
      {
      emitBit (val);
      return;
      }
    if (timeDiff >= DOUBLE_LOW_WIDTH && timeDiff <= DOUBLE_HIGH_WIDTH)
      {
      emitBit (!val);
      emitBit (val);
      return;
      }
    manchesterState = IDLE;
    } // Kraj ManchesterInterrupts::isr
// Funkcija 'isr' (interrupt service routine) se poziva kada se dogodi prekid
na ulaznom pinu. Očitava se nova vrijednost signala s pina i računa se
vremenska razlika (timeDiff) od zadnje promjene.

  int available ()
    {
    int count = ManchesterInterrupts::inpoint - ManchesterInterrupts::outpoint;
    if (count < 0)
      count += sizeof ManchesterInterrupts::buf;
    return count;
    } // Kraj ManchesterInterrupts::available
// Funkcija 'available' vraća broj dostupnih bajtova za čitanje iz bafera. Računa
se razlika između 'inpoint' i 'outpoint'. Ako je ta razlika negativna, dodaje se
veličina bafera kako bi se uzela u obzir mogućnost prekoračenja bafera.

  int read ()
    {
    if (ManchesterInterrupts::inpoint == ManchesterInterrupts::outpoint)
      return -1;
    byte c = ManchesterInterrupts::buf [ManchesterInterrupts::outpoint];
    if (++ManchesterInterrupts::outpoint >= sizeof ManchesterInterrupts::buf)
      ManchesterInterrupts::outpoint = 0;
    return c;
    }   // Kraj ManchesterInterrupts::read

// Funkcija 'read' čita sljedeći bajt iz bafera. Ako su 'inpoint' i 'outpoint'
jednaki, to znači da nema dostupnih bajtova za čitanje, pa se vraća -1. Inače,
čita se bajt s pozicije 'outpoint' u baferu, a 'outpoint' se povećava za 1. Ako
je 'outpoint' veći od veličine bafera, postavlja se na 0 kako bi se omogućilo
kružno čitanje bafera.
}
```