UNIVERZITET U NIŠU
ELEKTRONSKI FAKULTET
KATEDRA ZA MIKROELEKTRONIKU

OPTOELEKTRONIKA

LABORATORIJSKI PRAKTIKUM

Master inž. Miloš Đorđević

Niš, mart 2017.

SADRŽAJ:

1. Uvod..... 3

2. Proteus Profesional 7.7 5

2.1. ISIS Profesional – schematic designer 5

2.2. ARES Profesional – PCB layout designer 6

2.3. Crtanje komponenata 7

2.4. Simulacija električne šeme pomoću ISIS-a 15

2.5. Projektovanje štampane ploče pomoću ARES-a 19

3. Proton IDE Compiler 24

4. Primeri primene optokomponenata 27

4a. Razvojno okruženje i mikrokontroler PIC18F45K22 27

4b. Pomerački registar 74HC595 28

4c. LCD displej (LM016) 2x16 karaktera 30

4d. Fotootpornik 32

4d. Optičko vlakno 32

4e. Senzor vidljive svetlosti BH1750 34

4f. Senzor boje TCS230/3200 34

I vežba
..... Error! Bookmark not defined.

4.1a. Simulacija kontrole LED diode tasterom 37

4.1b. Simulacija rada niza od 4 LED diode 40

4.1c. Simulacija rada semafora 42

4.1d. LED elektronska kocka 44

II vežba 49

4.2a. LCD dsiplej 49

4.2b. LCD dsiplej – Scroll Message 51

III vežba 53

4.3a. Sedmosegmentni displej 53

4.3b. Sedmosegmentni displej – realizacija brojača 55

IV vežba 59

4.4a. Prenos informacija optičkim vlaknom korišćenjem RC5 Philips protokola 59

V vežba 63

4.5a. LED matrica 7x11 63

VI vežba 67

4.6a. Određivanje intenziteta osvetljenosti prostorije fotootpornikom 67

4.6b. Merenje intenziteta osvetljenosti senzorom vidljive svetlosti BH1750 - luxmetar 70

VII vežba 74

4.7. Kolorimetar 74

5. Dodatak – Zadaci za vežbu 80

1. Uvod

Praktikum za laboratorijske vežbe iz predmeta „***Optoelektronika***“ obezbeđuje neophodne informacije studentima za savladavanje gradiva u skladu sa planom i programom praktičnog dela predmeta. Problematika kojom se bavi ovaj praktikum je veoma aktuelna, iz razloga što postoji izuzetno široka primena optokomponenata i mikrokontrolera u veliko broju proizvedenih uređaja.

Optoelektronika – laboratorijski praktikum je namenjen pre svega savladavanju osnovnih hardverskih i softverskih komponenti koje su neophodne za razumevanje i realizaciju uređaja zasnovanih na optoelektronici. Posebna pažnja je posvećena upoznavanju, a kasnije i korišćenju programskih naredbi kojima se mikrokontrolerom upravlja komponentama kao što su LED diode, LCD, matrični i sedmosegmentni displeji. Takođe u ovom praktikumu su opisane komponente kao što su pomerački registri kojima se mikrokontrolerom upravlja već pomenutim komponentama za optoelektroniku.

Kako su poslednjih godina napredovali razni simulatori koji u velikom delu zamenjuju neophodnost praktičnih realizacija raznih uređaja za potrebe ispitivanja rada istih, u ovom praktikumu je dat pregled i objašnjenje simulatora koji prati rad sa svakom od vežbi iz ovog praktikuma. Treba imati na umu da sam simulator ne može da zameni testiranje i rad realizovanog uređaja u realnim uslovima!

U prvom delu praktikuma je dat prikaz i objašnjenje simulatora koji se kao što je navedeno koristiti za ispitivanje rada uređaja koje studenti realizuju. Simulator koji se koristi u ovom praktikumu je ***Proteus Profesional*** realizovan od strane **Labcenter Electronics**, koji u sebi objedinjuje dva zasebna potprograma, a to su ***ISIS*** – softverski alat za realizaciju električnih šema i simulaciju rada istih i ***ARES*** – softverski alat namenjen za izradu PCB-a (štampanih ploča). U ovom poglavlju je opisan i kompajler koji se koristi za pisanje koda kojim se definiše rad mikrokontrolera. Korišćeni kompajler je ***Proton IDE***, čija se sintaksa zasniva na programskom jeziku ***Proton Basic***.

U sledećem poglavlju kako bi se studenti upoznali sa radom najčešće korišćenih optokomponenata za izradu optoelektronskih uređaja, dat je pregled i objašnjenje rada istih. Između ostalih, optokomponente koje su opisane u ovom praktikumu su gore pomenute LED diode, na osnovu kojih su realizovani matrični displej, VU metar i razne svetlosne signalizacije. Zatim je opisan rad sa LCD displejima koji se koriste za interakciju sa korisnicima realizovanih uređaja, uspomoć kojih se mogu vršiti očitavanja vrednosti raznih merenja ili stanja uređaja tokom rada. Nakon LCD tu je i rad sa sedmosegmentnim displejima, koji se često sreću kod digitalnih satova, realizacije brojača posetioca, sportskih semafora,... Pored pomenutih optokomponenata, tu su i senzori čiji se rad zasniva na optoelektronici. Neki od senzora koji su korišćeni za realizaciju vežbi su fotootpornici, čija se otpornost menja sa promenom osvetljenja i senzor vidljive svetlosti, koji je korišćen za realizaciju Luxmetra.

Nakon pregleda komponenti, sledi deo sa vežbama koje studenti realizuju tokom izvođenja vežbi. Prvi termin vežbi predstavlja upoznavanje ili podsećanje rada sa optokomponentama i kompajelrom. U drugom terminu vežbi studenti se upoznaju sa radom

LCD displeja, a zatim i realizuju vežbe koristeći isti. Treći termin vežbi posvećen je radu sa sedmosegmentim displejom, zatim se studenti upoznaju i sa radom pomeračkih registara, uspomoć kojih se olakšava kontrola više sedmosegmentnih displeja. U četvrtom terminu koristi se optičko vlakno kao medijum za daljinsku kontrolu infracrvenog senzora daljinskim upravljačem. Objašnjen je rad i uspostavljena je IC komunikacija putem ***RC5*** (*Philips*) protokolom. Peti termin predstavlja realizaciju i rad sa matričnim displeje (***7x11*** LED dioda), koji služi za ispis teksta u vidu poruka. U poslednjem terminu vežbi pažnja je posvećena radu sa senzorima (fotootpornik i senzor vidljive svetlosti), koji se koriste za realizaciju uređaja koji objedinjuje neke od prethodno pomenutih komponenata, tačnije displeja.

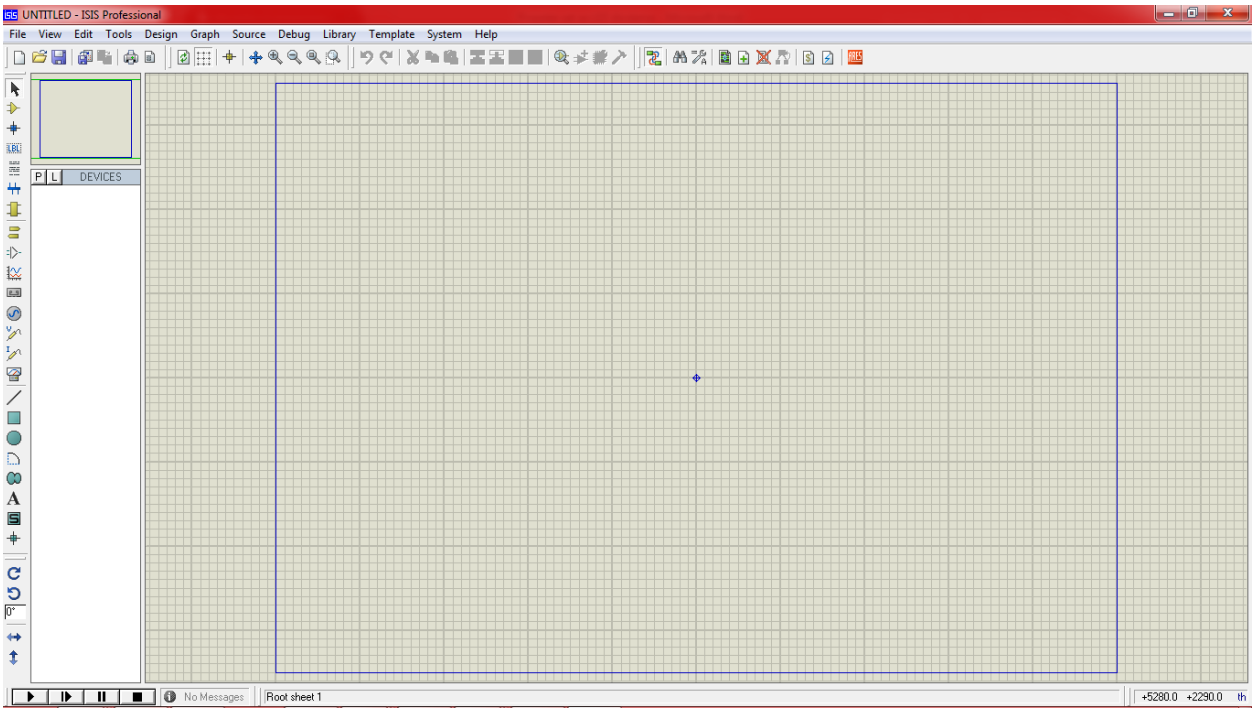
Poslednje poglavlje sadrži zadatke za samostalni rad, kako bi studenti mogli sami da testiraju svoje stečeno znanje i da isto prošire realizujući zadate probleme. Na taj način su obuhvaćene komponente i problemi s kojima se studenti mogu sresti tokom svog daljeg školovanja i kasnije rada u ovoj oblasti.

2. Proteus Profesional 7.7

Osnovna ideja ovog programskog paketa je da objedini alate koje sadrži u sebi na način koji bi korisniku omogućio lakšu interakciju sa samim programom. Ovaj programski paket pre svega objedinjuje dva veoma bitna alata. Alat korišćen za projektovanje električne šeme je *ISIS Profesional*, dok je alat za realizaciju PCB-a *ARES Profesional*. Kako ovaj programski paket poseduje mogućnost dodavanja novih elemenata, dodavanje se vrši na sledeći način. Pre svega, potrebno je proučiti *datasheet* same komponente koju treba dodati, kako bi se upoznali sa dimenzijama iste. Nakon toga, se u ISIS Profesional-u crta simbol samog elementa (komponente), koji će se kasnije povezati sa *footprint*-om (eng. otisak). Zatim sledi crtanje footprint-a u ARES-u.

2.1. ISIS Profesional – schematic designer

Ovaj alat omogućava korisniku projektovanje električnih šema, poseduje dosta veliku biblioteku komponenti (simbola), koje poseduju modele za *SPICE* simulaciju, jer se ovaj alat takođe može koristiti i za simulaciju same električne šeme. Kao što je navedeno, ako ne postoji komponenta koju korisnik želi da koristi u električnoj šemi, moguće je napraviti. Na slici 1. dato je okruženje ovog alata.

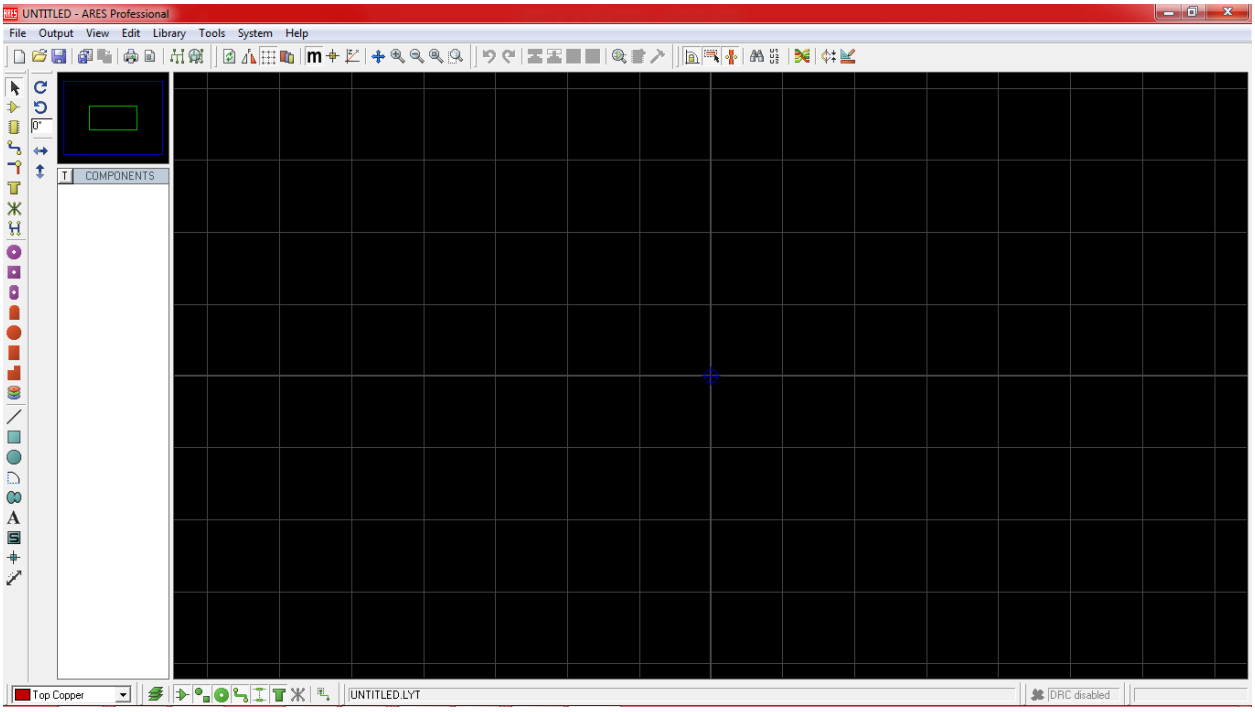


Slika 1. Okruženje - ISIS Profesional

Alat poseduje dobro organizovan grafički interfejs, tako da je veoma olakšana interakcija korisnika sa samim programom. Pored samih komponenata, ovaj program poseduje razne simulatore signala, osciloskope, koje korisnik može podešavati, takođe i mnoge signal generatore, analogne i digitalne multimetre i još dosta drugih stvari. Moguće je već gotovu komponentu prilagoditi (*decompose*, eng. rastaviti na sitnije delove) po svojoj želji.

2.2. ARES Profesional – PCB layout designer

Alat za projektovanje – realizaciju PCB-a na osnovu električne šeme projektovane u ISIS-u. Dakle, nakon projektovane električne šeme, korisnik vrši transver celokupne šeme u alat za projektovanje PCB-a. Zavisno od footprinta za svaku od korišćenih komponenti, tj. da li željena komponenta ima svoj otisak, može se odmah projektovati PCB. U suprotnom, neophodno je prvo nacrtati *footprint* (otisak) komponente, zatim ga povezati sa simbolom u ISIS-u i nakon toga pristupiti projektovanju PCB-a. Kao i u slučaju ISIS-a, tako i ARES poseduje veoma dobro organizovani grafički interfejs, pa je rad i u ovom alatu umnogome olakšan. Pored samih otisaka, ovaj program poseduje mogućnost 3D vizuelizacije projektovanog PCB-a. Okruženje ovog programa dato je na slici 2.



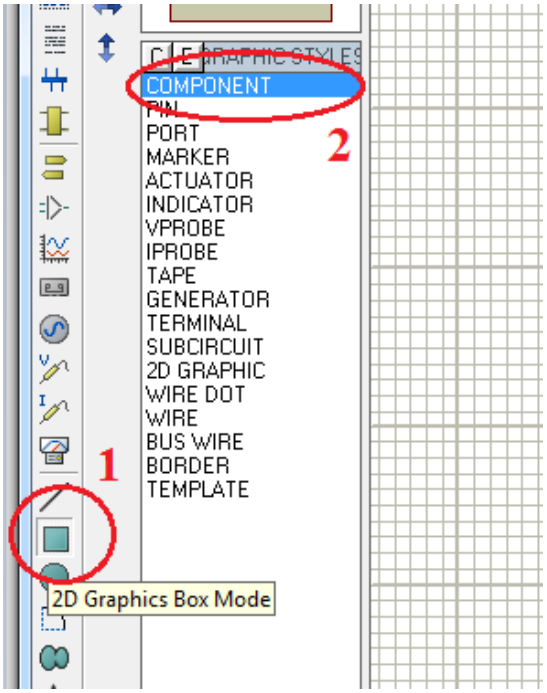
Slika 2. Okruženje - ARES Profesional

Poseduje i dosta interesantnih stvari kao što su razne veličine *via* (eng. kontaktna rupa), veliki asortiman linija za povezivanje, kao i mogućnost projektovanja nove debljine linija. Poseduje *Auto-router*, tako da se prevežena šema iz ISIS-a u ARES može automatski povezati, ali takođe ostavlja mogućnost korisniku izmene linija. Poseduje dosta kontakata, kao što su kontakti za trough-hole komponente i SMD *pad*-ove (eng. kontakti za SMD komponente).

2.3. Crtanje komponenata

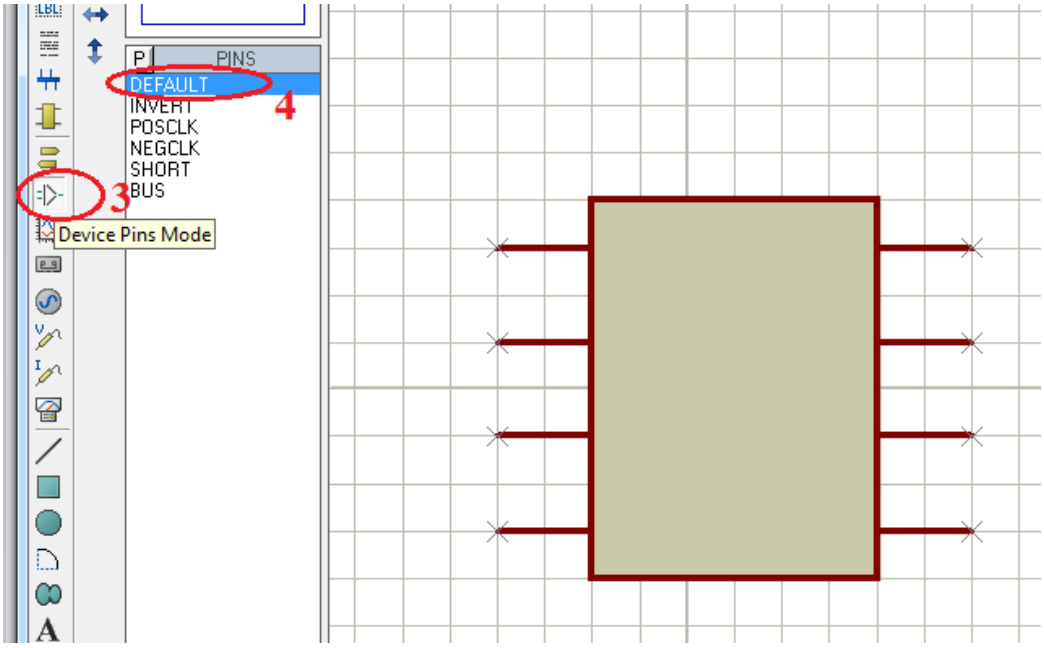
Pre crtanja komponente(i), neophodno je konsultovati datasheet, kako bi se znali koje su dimenzije iste, a takođe i *pinout* kako bi se prilikom izrade samog PCB-a pinovi podudarali sa štampom. Nakon upoznavanja sa datasheet-om, može se početi sa crtanjem komponente(i). Ceo postupak je ilustrativno propraćen slikama, počev od 3, do 16, respektivno.

U ISIS-u je najpre potrebno nacrtati simbol nove komponente kao što je dato na slici 1.



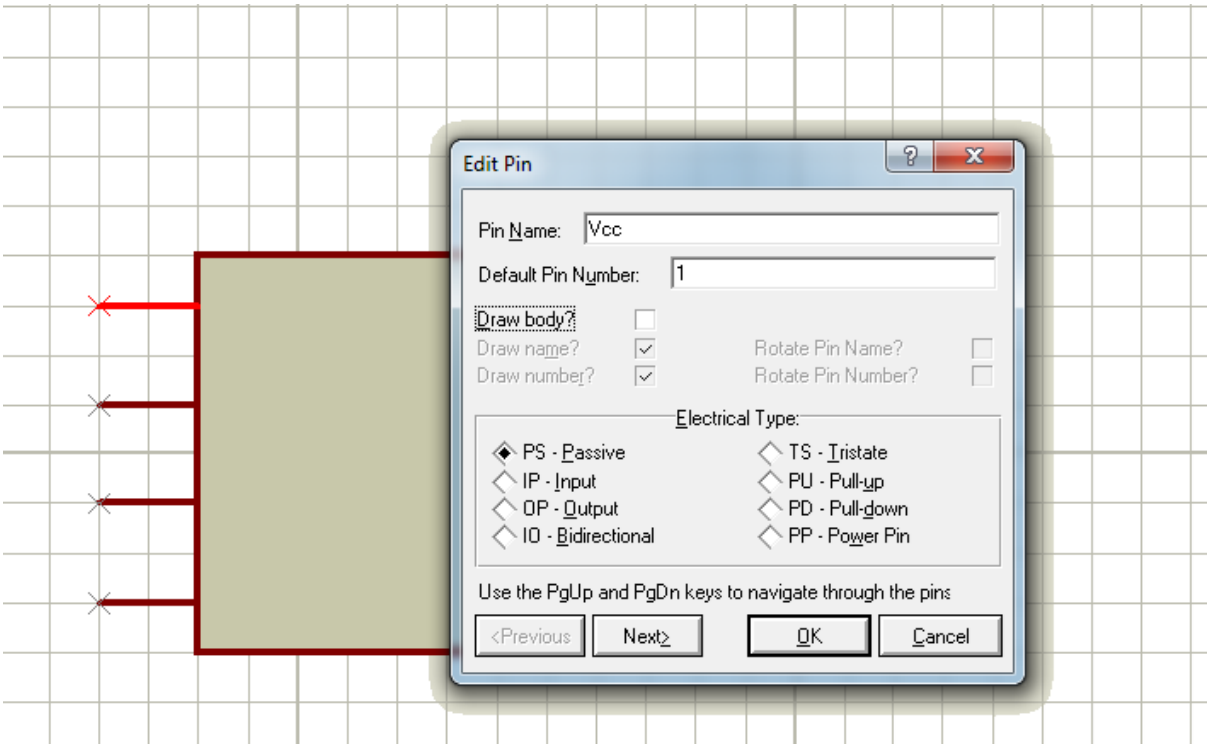
Slika 3. Izbor alata za crtanje simbola (ISIS)

Pre svega, potrebno je odabrati alat (u ovom slučaju simbol za crtanje kvadratnih oblika (1)), nakon čega sledi izbor onoga što želimo da nacrtamo, tj. simbol onoga što crtamo. Kako je nama potrebno da nacrtamo komponentu, biramo kao na slici *COMPONENT* (2). Posle ovog koraka, sledi samo crtanje simbola i postavljanje pinova na simbolu (slika 4.).



Slika 4. Postavljanje pinova na simbol komponente (ISIS)

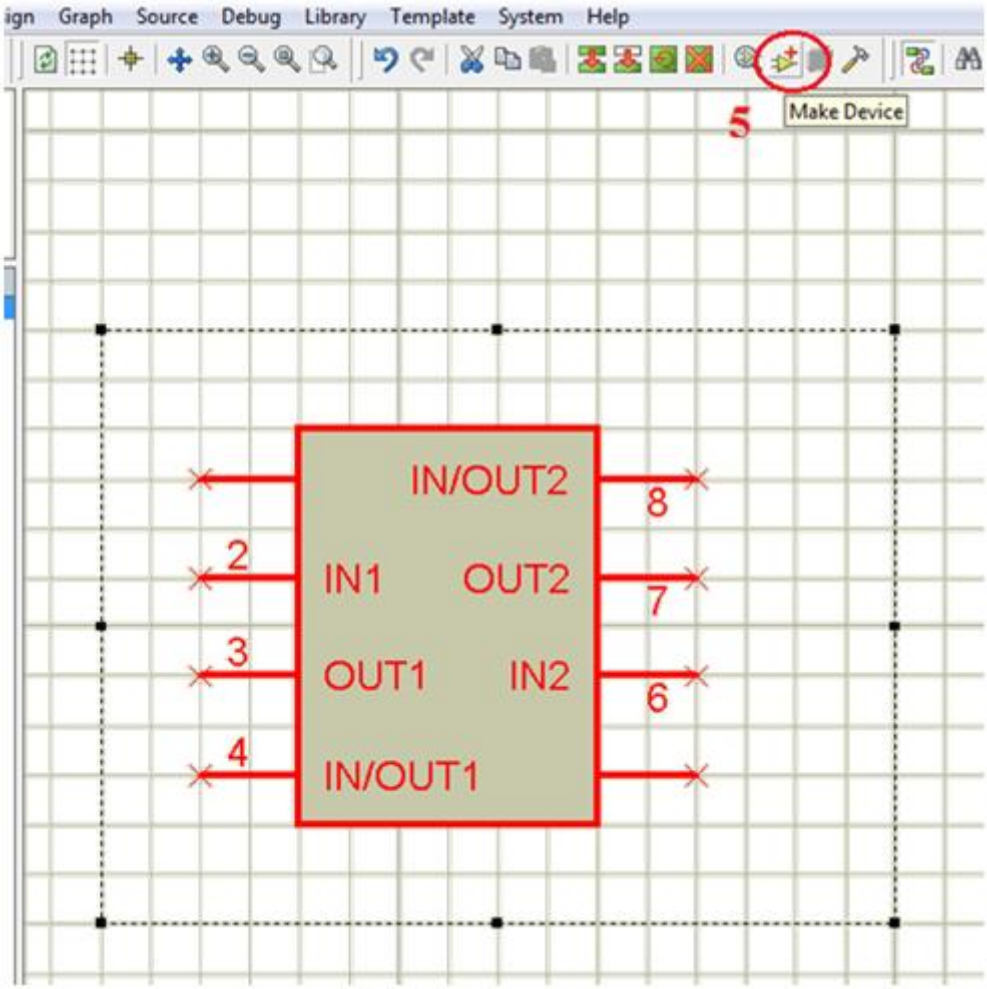
Postavljanje pinova prate koraci 3 i 4, gde se biraju *default* (eng. podrzumevani - standardni) pinovi, gde oznaka na kraju pina u obliku slova *x* predstavlja kontaktni deo pina. Kada su pinovi postavljeni, neophodno je da se ti isti pinovi imenuju, kako bi se mogao napraviti ispravan simbol (slika 5.).



Slika 5. Imenovanje pinova komponente (ISIS)

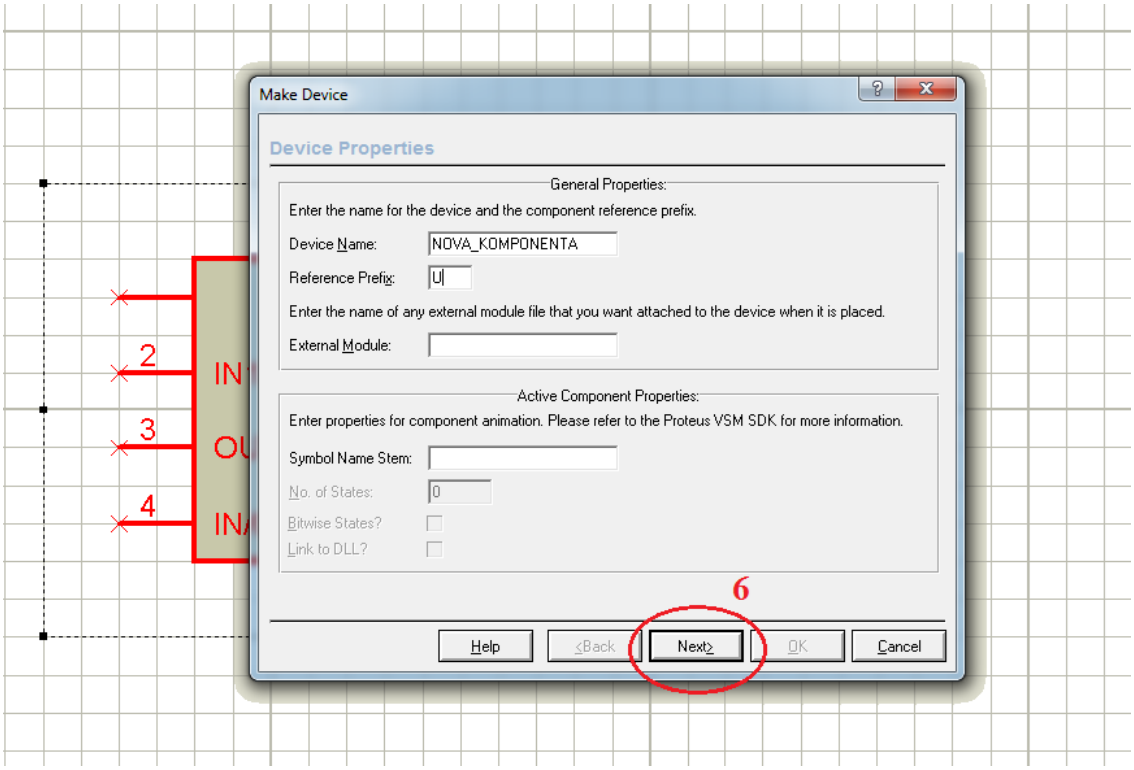
Kod imenovanja pinova, treba imati u vidu da se pinovi za napajanje, odnosno masu, ne prikazuju na simbolu, tako da se ne „štiklira“ deo vezan za prikaz tog pina, jer su to

podrazumevani pinovi. Oni se kasnije u šemi smatraju vezanim na napajanje, odnosno masu. Kada je završeno imenovanje pinova, simbol komponente je spreman za smeštanje u biblioteku (naravno posle povezivanja sa ARES-om) (slika 6.).



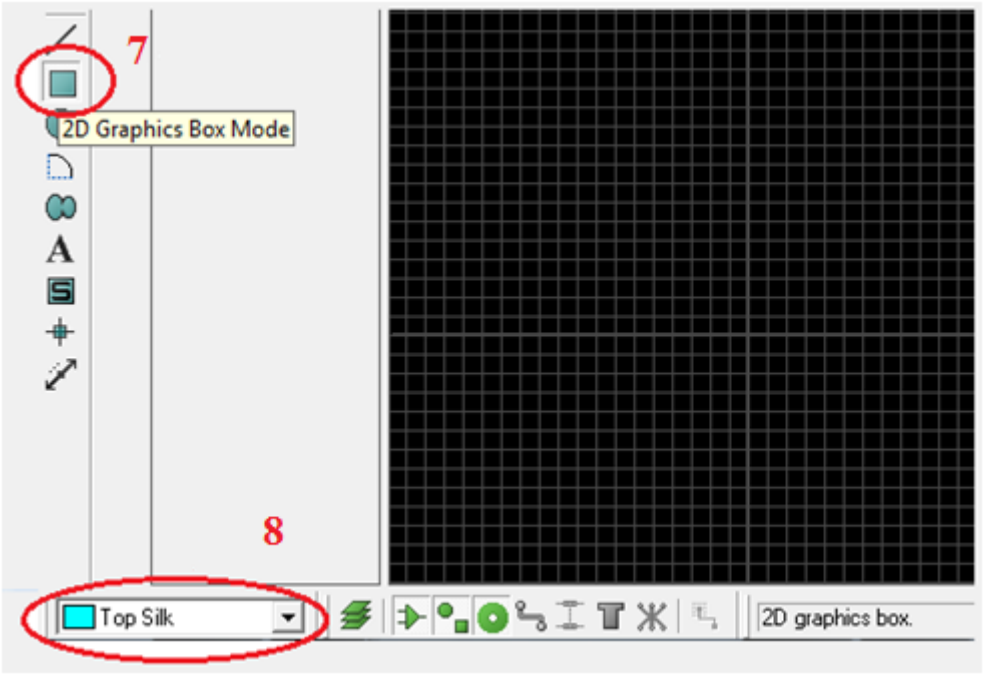
Slika 6. Pamćenje (smeštanje) simbola u biblioteku (ISIS)

Pritiskom na ikonicu određenu za pamćenje simbola komponente, otvara se podprozor gde se imenuje sam simbol komponente i bira se index simbola koji će kasnije biti korišćen u električnoj šemi (slika 7.).



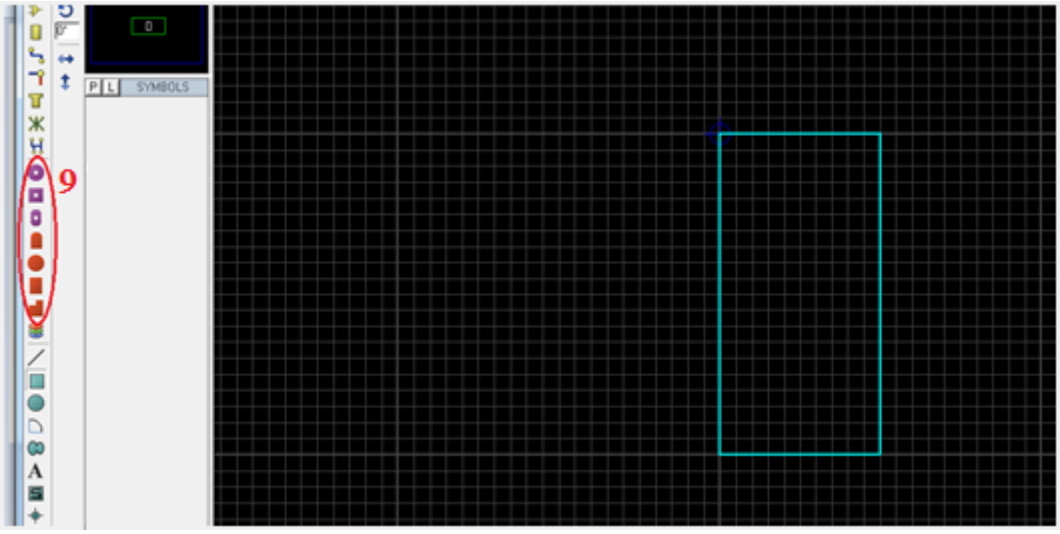
Slika 7. Imenovanje simbola komponente (ISIS)

Crtanje footprinta u ARES-u zahteva poznavanje dimenzija nove komponente koju crtamo. Pre crtanja se kao i u slučaju ISIS-a prvo bira alat za crtanje pravougaonih oblika (7), a takođe i bira se *layer* (eng. sloj – nivo) u kome crtamo kućište komponente (slika 8.).



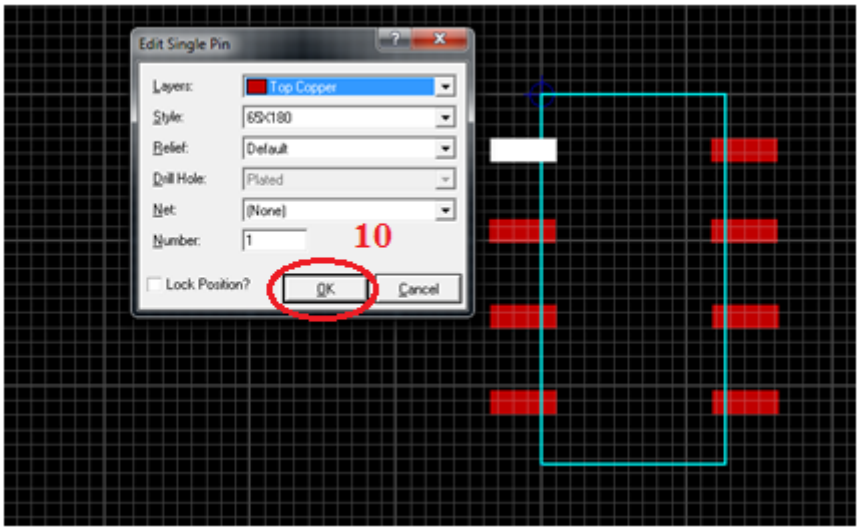
Slika 8. Izbor alata za crtanje footprinta (ARES)

Nakon izbora alata sledi crtanje kućišta nove komponente, nakon čega sledi postavljanje pinova na kućište. Pinovi mogu biti SMD (kružni, pravougaoni, moguće je crtanje novih oblika SMD pad-a) i trough hole (takođe kružni, pravougaoni i elipsasti). Izbor kućišta je predstavljen korakom 9 (slika 9.).



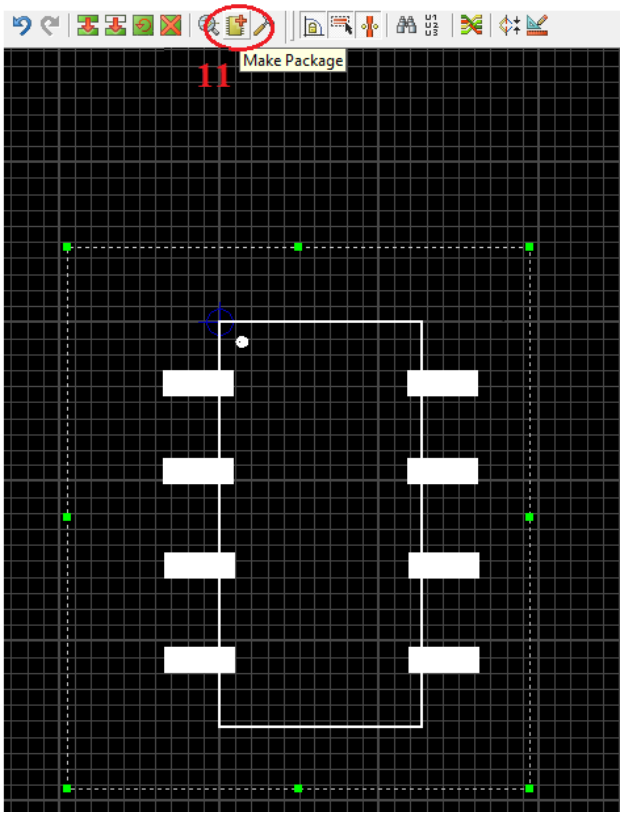
Slika 9. Crtanje kućišta i izbor pinova nove komponente (ARES)

Kada je nacrtano kućište, izabrani pinovi, sledi crtanje istih i navođenje broja svakog pina ponaosob. Brojevi pinova ne moraju da se podudaraju sa brojevima iz ISIS-a, mada je mnogo bolje to uraditi, iz razloga lakšeg snalaženja, kao i iz estetski razloga (slika 10.).



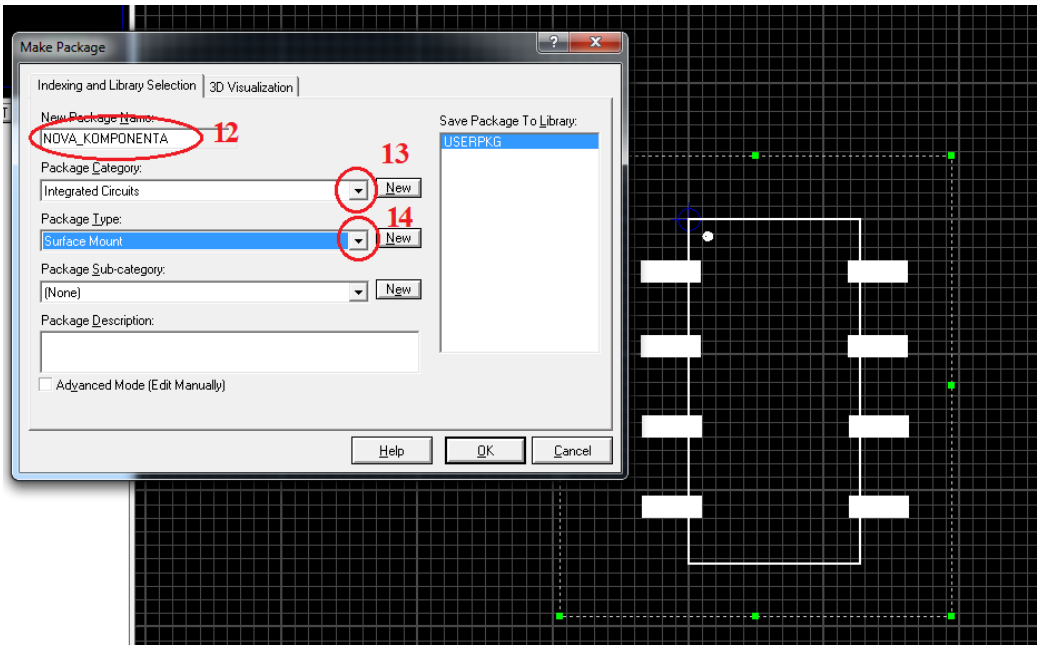
Slika 10. Dodeljivanje brojeva pinovima nove komponente (ARES)

Nakon dodeljivanja brojeva pinovima, sledi pamćenje footprinta nove komponente (slika 11.).



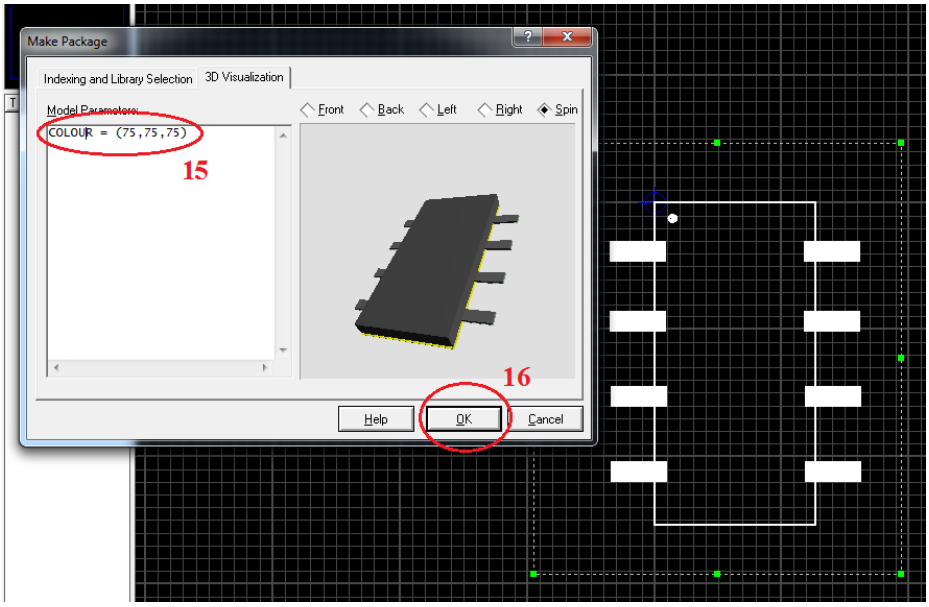
Slika 11. Pamćenje footprinta nove komponente (ARES)

Pritiskom na ikonicu kako bismo upamtili footprint nove komponente, otvara se novi podprozor (slika 12.) u kome dajemo ime novoj komponenti, naravno, ime biramo tako da bismo je lakše našli pri navođenju pakovanja, tj. pri izboru footprinta za simbol iz električne šeme. Pored imena, treba navesti kategoriju komponente u koraku 13. Takođe i da li je komponenta trough hole ili SMD (korak 14.).



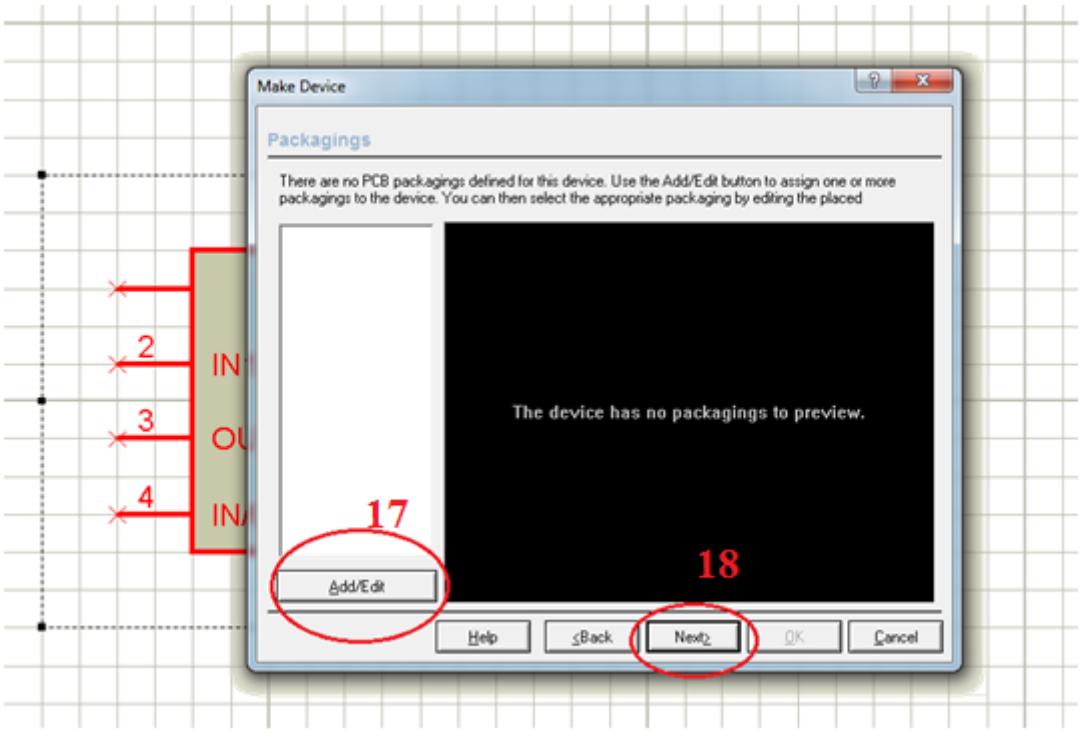
Slika 12. Dodeljivanje imena, kategorije i tipa nove komponente (ARES)

Kada je dodeljeno ime, kategorija i tip komponente, moguće je pozabaviti se 3D modelom, koji je dat u sledećoj kartici podprozora (3D Visualization), gde je moguće definisati izgled nove komponente (slika 13.).



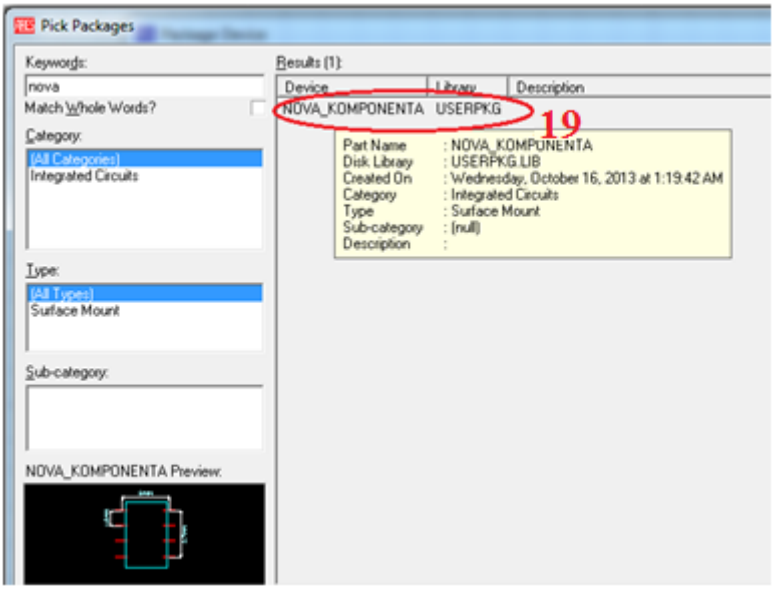
Slika 13. 3D model nove komponente (ARES)

Nakon crtanja footprinta nove komponente, vraćamo se u ISIS gde sledi izbor, tj. dodavanje footprinta simbolu nove komponente crtanog u ARES-u (slika 14.).



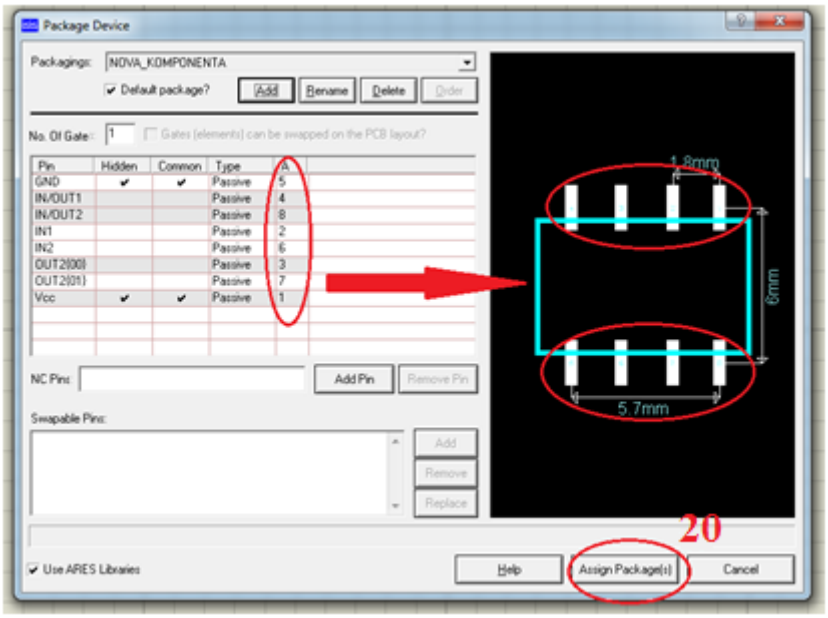
Slika 14. Dodavanje footprinta simbolu komponente (ISIS)

Klikom na ikonicu predviđenu za dodavanje (Add/Edit), sledi izbor footprinta nove komponente, gde će se prvi put u delu za pretragu footprinta naći i ime nove komponente (slika 15.).



Slika 15. Izbor footprinta nove komponente (ISIS)

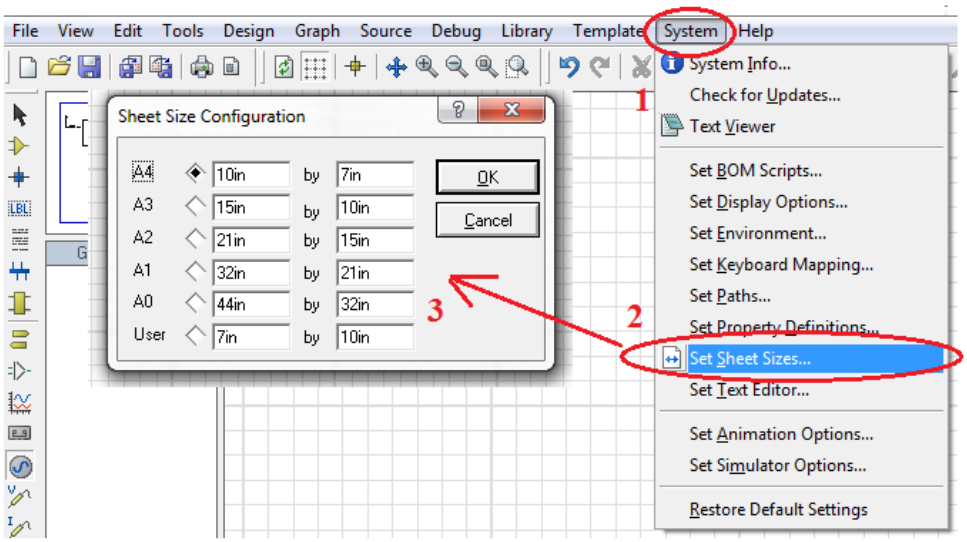
Na kraju sledi usklađivanje footprinta sa simbolom (slika 16.), tj. uklapanje pinout-a simbola i footprinta. Ovo je neophodno uraditi iz razloga što pri izradi PCB-a mogu javljati greške pri povezivanju pinova. Ovime je crtanje nove komponente završeno i može se ista koristiti pri projektovanju električne šeme, odnosno PCB-a.



Slika 16. Usklađivanje pinout-a simbola i footprinta (ISIS)

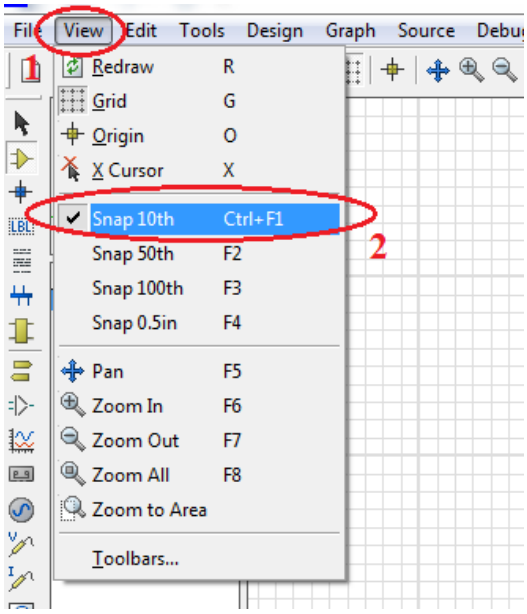
2.4. Simulacija električne šeme pomoću ISIS-a

Kao što je već navedeno u odeljku 1.1 ISIS pored same mogućnosti projektovanja električnih šema pruža mogućnost simulacije iste. Pre samog projektovanja električnih šema, potrebno je upoznati se sa razvojnim okruženjem samog alata. Najpre treba obezbediti dovoljno radnog prostora u samom prozoru programa, kako bi u slučaju veće električne šeme sve komponente i veze mogle da stanu untar radnog prostora. Dimenzije radnog prostora (slika 17.) se podešavaju iz razloga što ISIS veze ili komponente koje se nalaze van radnog prostora ne tretira kao deo jedne celine – električne šeme, već kao zaseban deo. Podešavanje dimenzija se vrši na sledeći način: Iz padajućeg menija unutar kartice „System“, unutar kog su i druga podešavanja, izabrati opciju „Set Sheet Size“, klikom na nju otvara se novi prozor u kome se nalaze neke generalizovane dimenzije, kao što su A4, A3,..., ali takođe i ostavljena mogućnost samom korisniku da po svojoj želji podesi dimenzije radnog prostora.



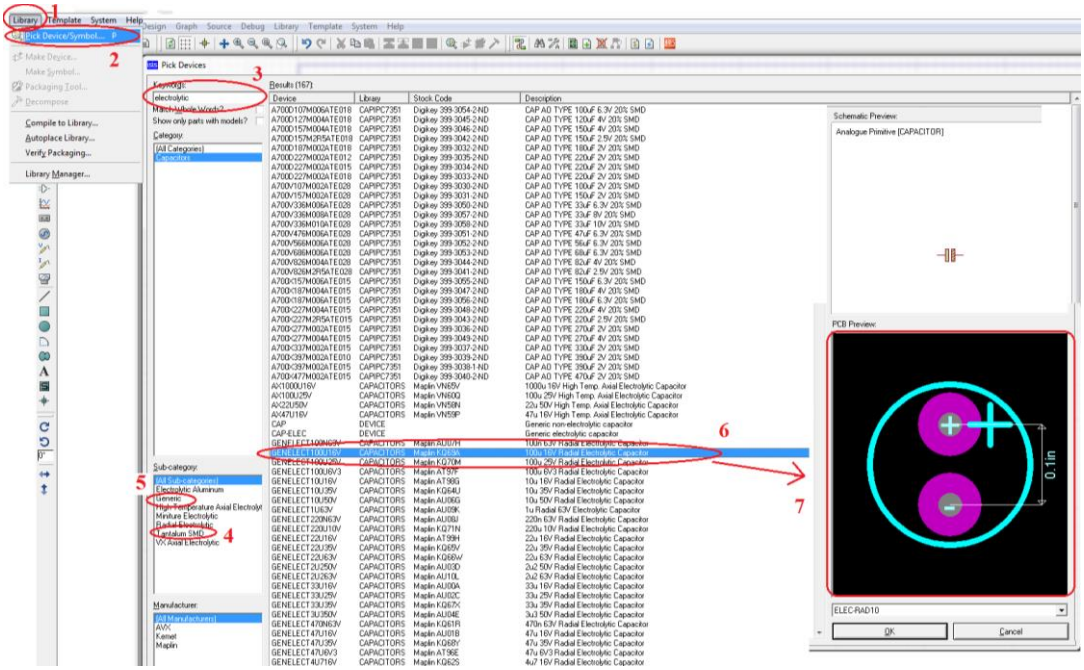
Slika 17. Podešavanje dimenzija radnog prostora

Nakon podešavanja dimenzija, radi lakšeg postavljanja komponenti, postoji povoljna mogućnost da se podesi i „korak“ po kome će se komponente pomerati unutar radnog prostora (slika 18.). Drugim rečima, da li će se komponenta ili veza unutar radnog prostora pomerale za 0.5 inča (1,27 cm), što može da smeta kod većih električnih šema, pa se taj korak može smanjiti na 10 th (0,0254 cm). Ovo smanjenje koraka je ne samo pogodno kod većih električnih šema, već je i prijatnije za korisnika, jer se stiče utisak realnog pomeranja same komponente. Promena koraka se izvodi na sledeći način: Iz padajućeg menija se izabere kartica „View“ u kojoj su neke osnovne opcije izgleda samog prostora, između kojih je i korak pomeranja, gde postoje ponuđene vrednosti kao što su 10 th, 50 th, 100 th i 0,5 in (inč).



Slika 18. Podešavanje koraka pomeranja komponenti

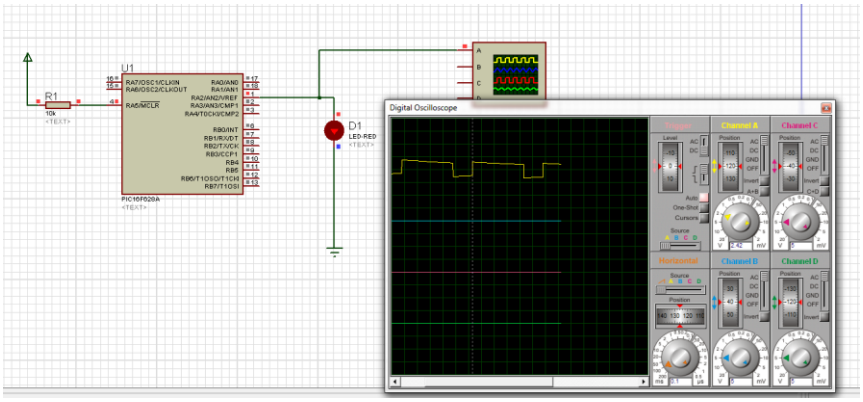
Komponente koje korisnik želi da koristi (slika 19.), nalaze se u biblioteci (1), koja se otvara izborom kartice iz padajućeg menija sa nazivom „Library“, koja je unutar iste podeljena na više manjih biblioteka (2). To znači da ukoliko korisnik želi na primer da koristi elektrolitski kondenzator (3), biće mu ponuđeno da izabere da li želi SMD ili Through-hole komponentu, a takođe i nazivna vrednost iste (4). Pored toga, ovaj alat pruža mogućnost korišćenja univerzalne komponente (5), u ovom slučaju elektrolitskog kondenzatora, gde će sam korisnik navesti nazivnu vrednost kasnije (6).



Slika 19. Biranje komponenti iz biblioteke

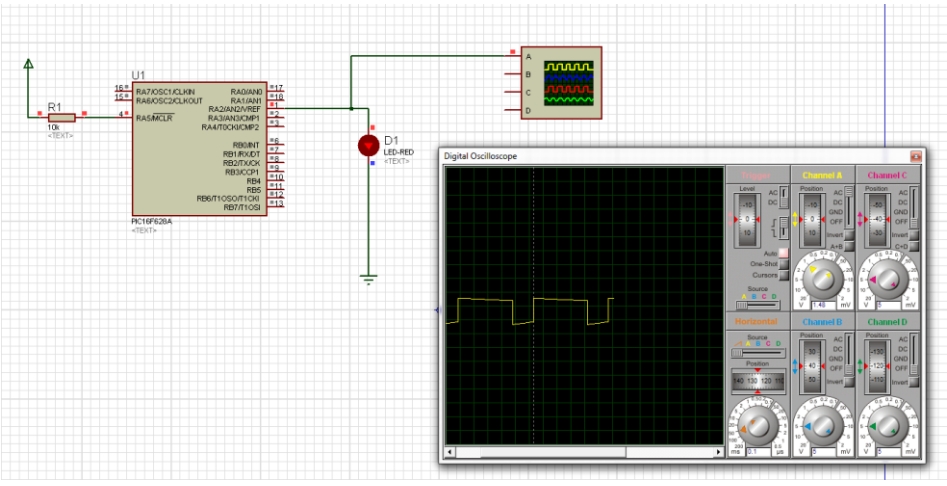
Nakon klika na komponentu koju korisnik želi u prozoru za izgled *footprint*-a, ukoliko izabrana komponenta ima već *footprint* u ARES-u, pojaviće se njen *footprint* (7).

Modeli komponenata koje se nalaze u biblioteci samog alata opisane su SPICE modelima, tako da ovaj alat može simulirati njihove električne karakteristike. ISIS poseduje mogućnost simuliranja četvorokanalnog osciloskopa (slika 20.), tako da je moguće ispitivati 4 električna signala u jednom trenutku. Jedan od primera simuliranja osciloskopa, tj. električnih signala, može se videti na osnovu primera "blinkanja" LED diode.



Slika 20. Simuliranje osciloskopa u ISIS-u

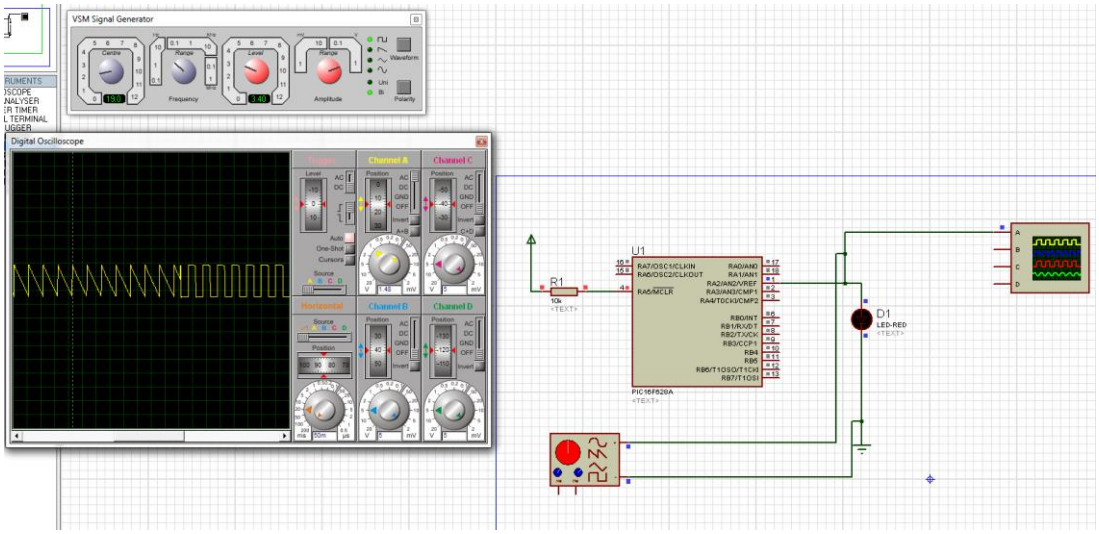
Sa slike se može videti praćenje signala na anodi LED diode. Moguće je podešavanje kao i na realnom osciloskopu. Na slici se takođe vidi da je na ekranu osciloskopa prikazano svih 4 signala, tj. signali sa svih 4 kanala. Takođe je moguće odabrati koji signal korisnik želi da prati. U skladu s tim, ekran osciloskopa sa gornje slike može izgledati na sledeći način.



Slika 21. Prikazivanje jednog signala pomoću osciloskopa

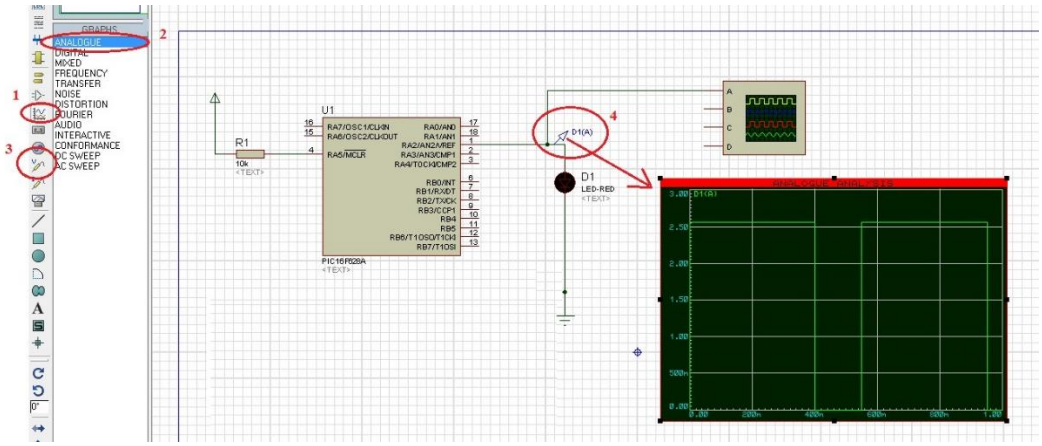
Na ovaj način moguće je lakše manipulisati samim prikazom signala, moguće ga je pomerati po vertikali i horizontali, podešavanje vremenske ose, amplitude...

U ISIS-u se takođe može koristiti signal generator, koji pored sinusnog signala podržava i truogaone, pravougaone i testeraste oblike signala. Primer testerastog i pravougaonog signala dat je na slici 22.



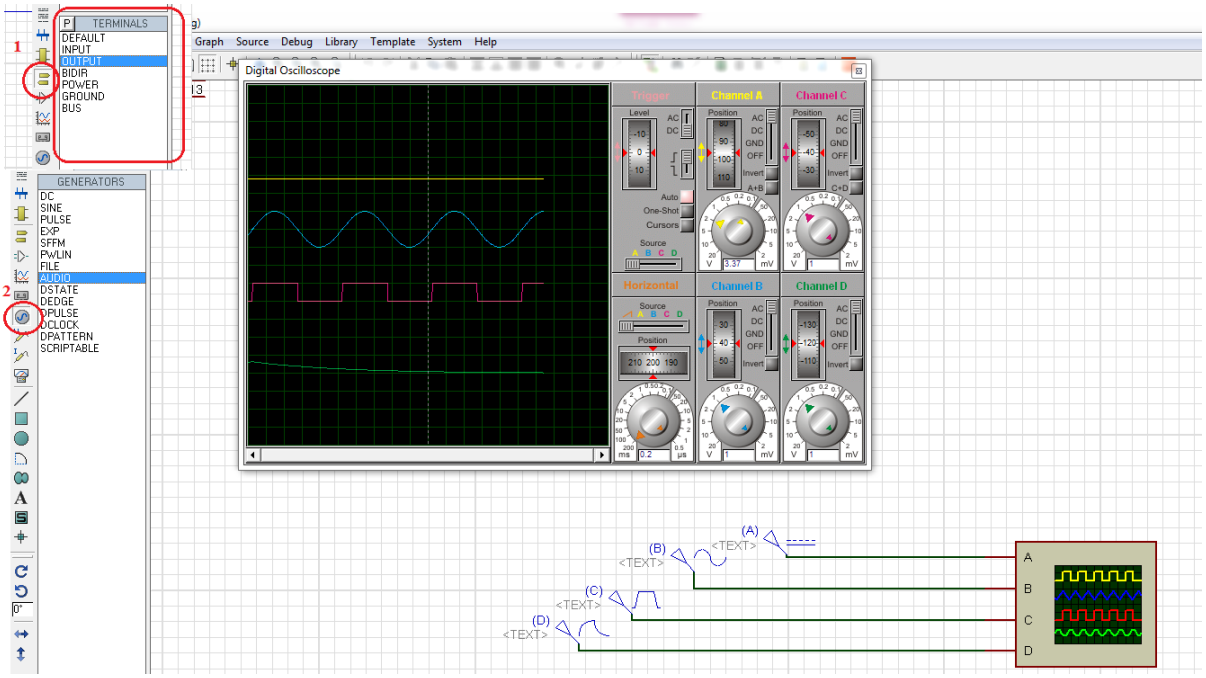
Slika 22. Primer korišćenja signal generatora

Pored osciloskopa, signali se mogu pratiti i pomoću grafika, koji mogu biti za AC i DC analizu. Najpre treba iz menija izabrati analizu pomoću grafika (1), zatim izabrati koju analizu, u ovom slučaju je to AC analiza (2). Da bi se na grafiku prikazao signal, tj. da bi mogli da radimo analizu, neophodno je izabrati koji signal treba pratiti. To se vrši tako što se izabere iz menija „Voltage probe mode“ (3) i to povezali na liniju koja ide ka anodi (4) i taj napon, tj. pokazivač na taj napon uvesti kao novi grafik (Add new trace), kako bi na grafiku videli analizu napona na LED diodi. Primer ovoga je dat na slici 23.



Slika 23. Analiza simulacije pomoću grafika

U ISIS-u postoji generalizovan izvor napajanja (1) koji se nalazi u odeljku za terminale (u okviru kog se takođe nalaze uzemljenje (GND), ulazno-izlazne labele, bidirekcioni terminal i BUS - magistrala), kojim se dobija jednosmerni napon, ali pored njega u odeljku Generatora (2) se mogu naći zasebno izvori napajanja (slika 24.), kao što su DC, impulsni signal, eksponencijalni, sinusni, a pored njih, tu su još i audio signal, kod njega sam korisnik zadaje audio signal, ali samo u wav formatu.



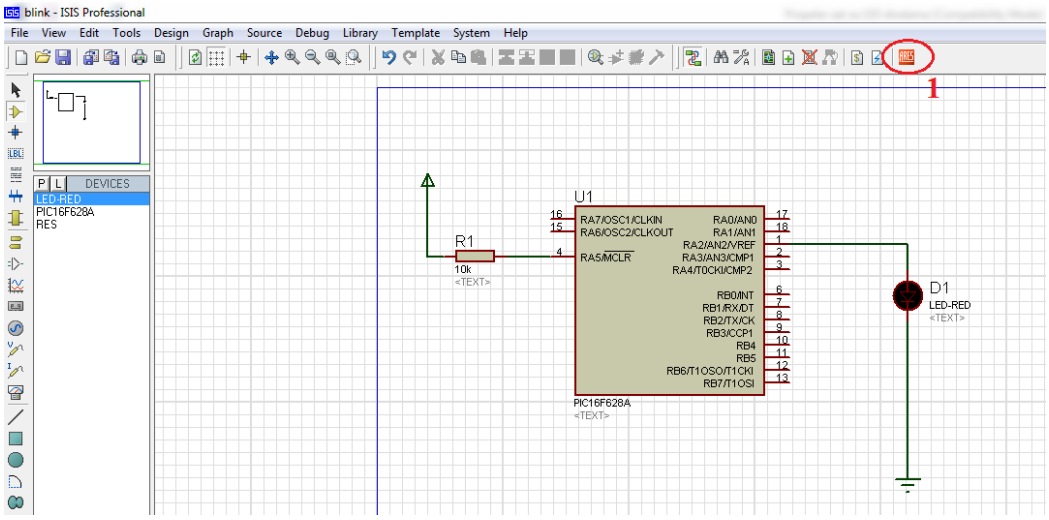
Slika 24. Izvori napajanja i ulaznih signala

Kako se može videti na osciloskopu, na kanalu A vezan je DC izvor napajanja (5V), na kanalu B sinusni signal sa amplitudomod takođe 5V, na kanalu C impulsni signal sa takođe istom vrednošću amplitude, a na kanalu D eksponencijalni signal.

2.5. *Projektovanje štampane ploče pomoću ARES-a*

Kada je završeno projektovanja električne šeme i nakon uspešne simulacije iste, sledi projektovanje štampane ploče (PCB-a). Prilikom uvoženja električne šeme u ARES, alat proverava sve korišćene komponente, kako bi za svaku našao odgovarajući *footprint*, ukoliko neka od komponenti ne poseduje isti, ARES će to odmah prijaviti. U tom slučaju, neophodno je nacrtati *footprint* koji nedostaje (Poglavlje 1.3).

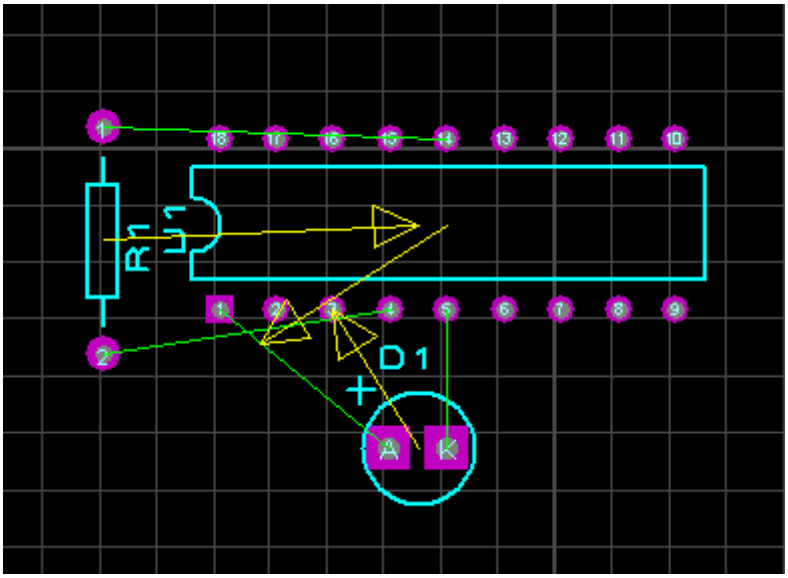
Kada postoje svi *footprint*-ovi za korišćene komponente može se pristupiti projektovanju štampane ploče. Za projektovanje PCB-a biće iskorišćen primer blinkanja diode, koji je naveden u primeru za simulaciju. Klikom na ikonicu za ARES u alatu ISIS, automatski se prelazi u novi alat za projektovanje PCB-a (slika 25.).



Slika 25. Prelaz iz ISIS-a u ARES

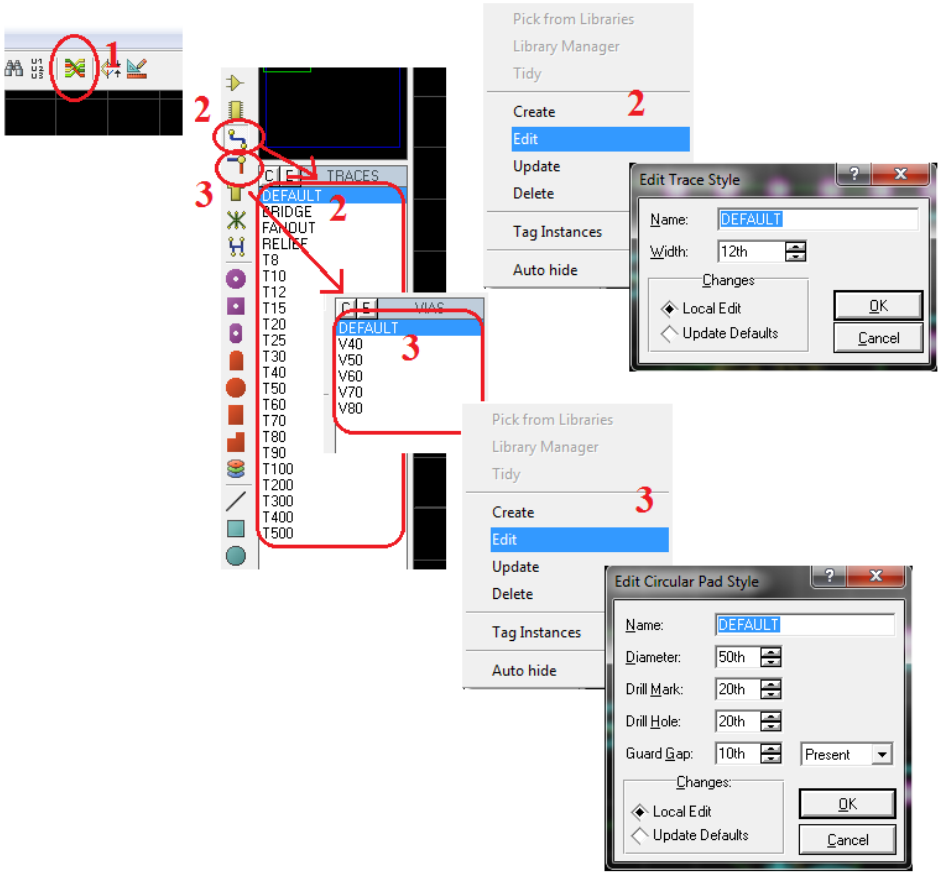
Nakon toga, otvara se novi prozor novog alata – ARES. Kako za sve komponente postoje *footprint*-ovi, nema nikakvih upozorenja, tako da korisnik može odmah početi sa postavljanjem komponenti. Veoma je važno da ukoliko se električna šema sastoji iz više zasebnih blokova i da je moguć uticaj blokova međusobno, potrebno ih je „galvanski“ razdvojiti. To se radi iz razloga da ne dođe kasnije prilikom rada uređaja do preslušavanja (u slučaju digitalnih blokova) ili uticaja neki signala na druge signale. U ovom slučaju nema potrebe za tim.

Najpre postavljamo mikrokontroler, iz razloga što je on po dimenzijama najveća komponenta, kako bi sama štampana ploča bila ekonomičnija za izradu (1). Zatim se postavljaju i „pull-up“ otpornik i LED dioda (slika 26.). Pull-up otpornik služi kako bi se osim softverski isključenog MCLR pina na mikrokontroleru i hardverski isključio, da ne bi došlo do neželjenog resetovanja samog mikrokontrolera.



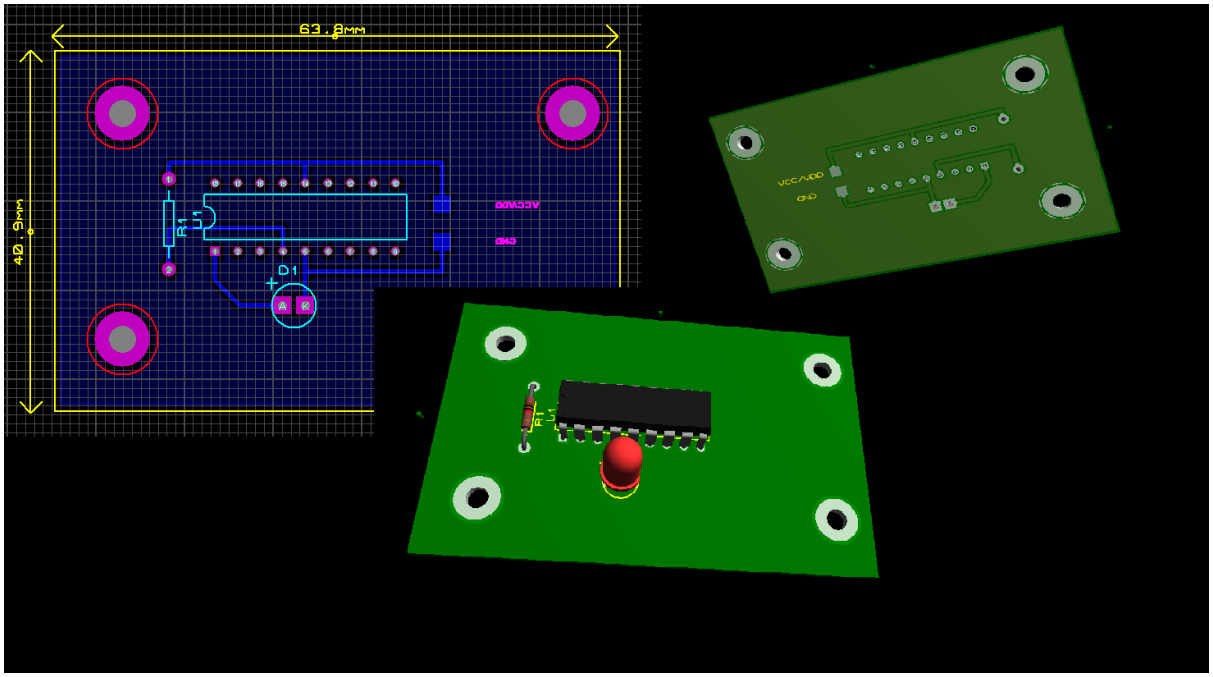
Slika 26. Postavljene komponente pre povezivanja

Zatim sledi povezivanje komponenti. Pre samog povezivanja treba znati da postoji mogućnost auto-rutiranja (*autorouting*), koja služi da sam alat poveže komponente na način koji alat odredi kao najbolji (1). Loša strana toga je što alat nalazi najlakše rešenje za povezivanje ne uzimajući u obzir da li su linije postavljene na gornjem ili donjem sloju štampane ploče. Najčešće postavlja linije koristeći i jedan i drugi sloj (*layer*). To često ne odgovara korisniku, pa je najbolje potrošiti malo više vremena i povezati onako kako je korisniku bolje i ekonomičnije. Takođe, pre povezivanja treba odrediti debljine linija (*default*) kako linije ne bi bile pretanke za kasniju praktičnu realizaciju (2). Pored linija za povezivanje, u slučaju da se radi o dvoslojnoj ili višeslojnoj ploči, neophodno je odrediti i širinu kontaktnih rupa (*via*) koje povezuju linije jednog sloja sa linijama na drugom sloju (3). Postupak podešavanja je dat na slici 27.



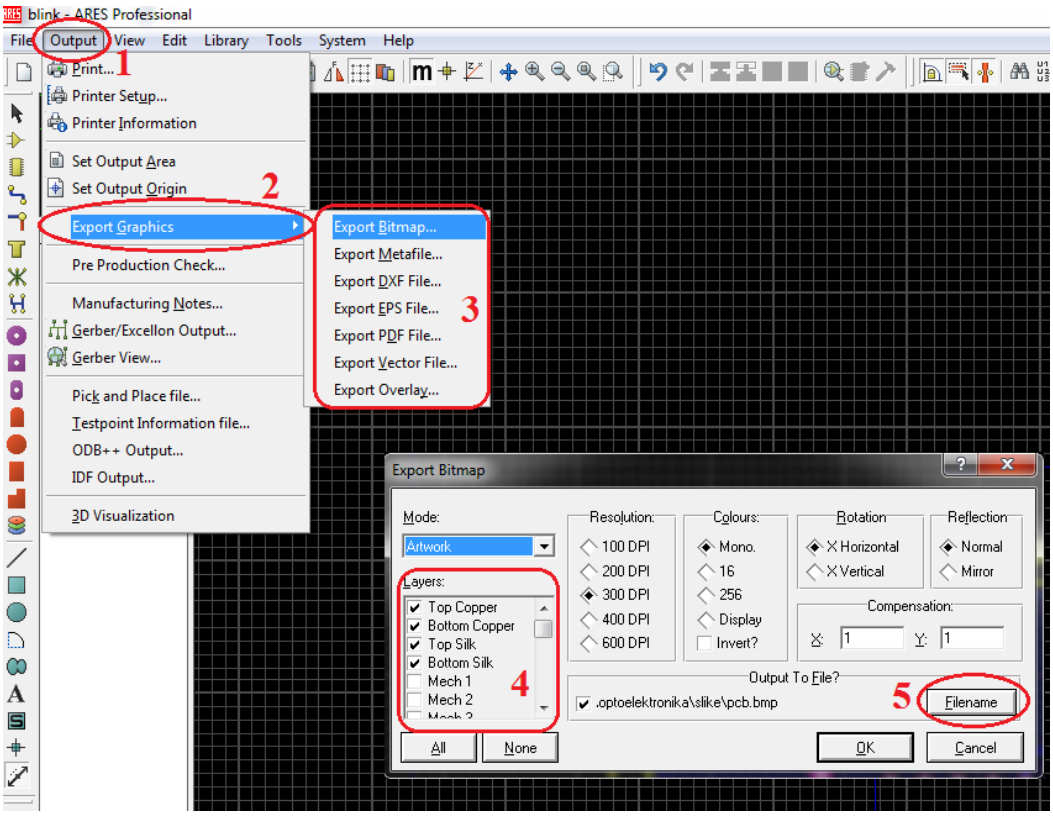
Slika 27. Podešavanje debljina linija i širina kontaktnih rupa

Nakon podešavanja i povezivanja, treba povezati pločicu. Moguće je postaviti i rupe (1) za odstojnike ako se ploča želi postaviti na podnožje, a veoma bitno u tom slučaju je koristiti opciju „keepout“-a (2), gde se postavlja zabrana postavljanja linija i komponenti u neposrednoj blizinu rupa za podnožje. To se radi iz razloga da se linije i komponente ne bi oštetile prilikom bušenja rupa. Nakon toga PCB izgleda na sledeći način (slika 28.). Korisnik može oivičiti pločicu, ukoliko želi da vidi 3D model iste.



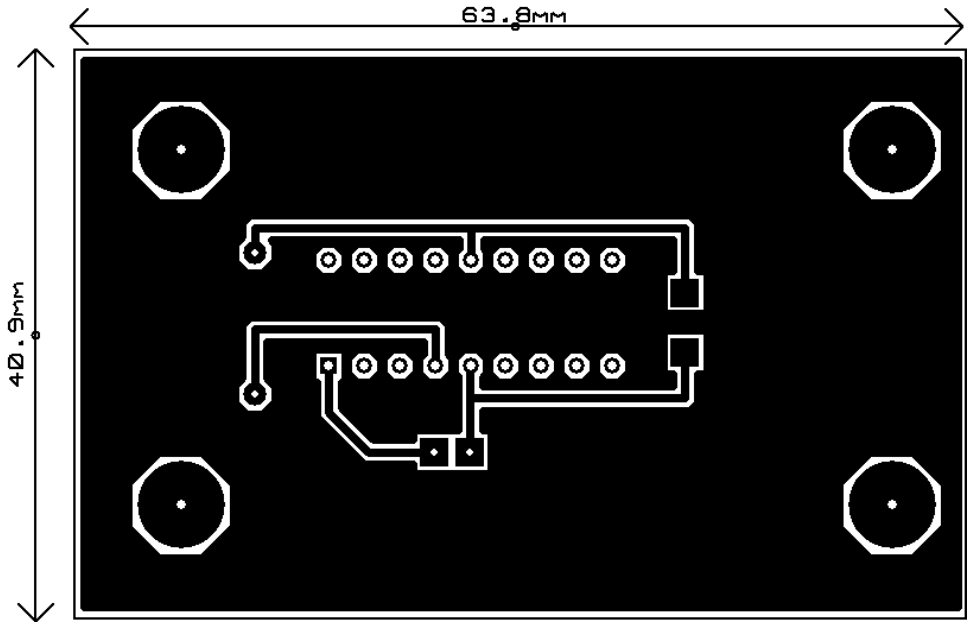
Slika 28. Izgled povezane pločice i njen 3D model

Na kraju treba izvući linije na osnovu kojih će se raditi štampana pločica, tj. treba izvesti pločicu iz alata. Dakle, iz padajućeg menija izabrati karticu „Output“ (1), zatim karticu „Export Graphics“ (2), koja nudi više opcija za izvoz (3), između kojih su formati Bitmap, PDF i drugi. U ovom slučaju izabrana je opcija izvoza Bitmap (.bmp), mada za štampanje je preporušljivo da to bude u PDF (.pdf) formatu. Zatim će se otvoriti novi prozor u kome se podešava šta se izvozi (4), linije u gornjem sloju (*Top trace*), linije u donjem sloju (*Bottom trace*), mesta komponenti na gornjem sloju (*Top silk*), mesta komponenti u donjem sloju (*Bottom silk*),... Nakon toga se bira mesto izvoza, tj. destinacioni folder (5) u kome će biti smepšten fajl kojim se izvozi štampana pločica (slika 29.).



Slika 29. Izvoz štampane pločice

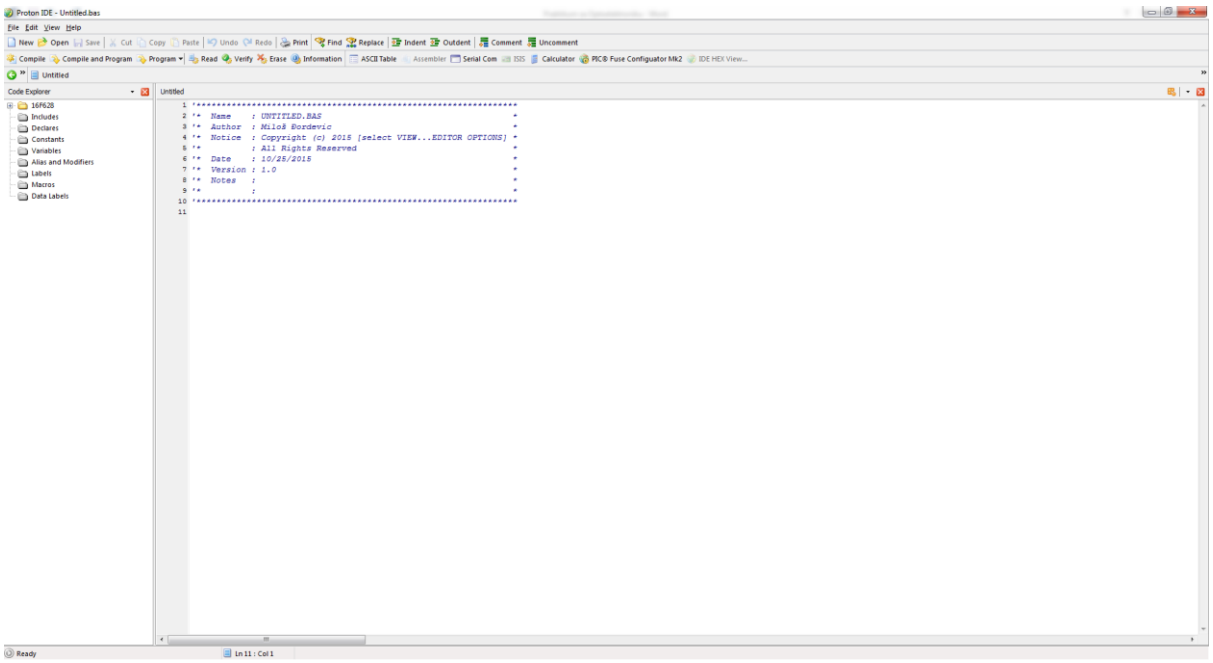
Konačno, izgled izvezenog fajla štampane pločice izgleda kao na slici 30.



Slika 30. Izgled fajla izvezene štampane pločice

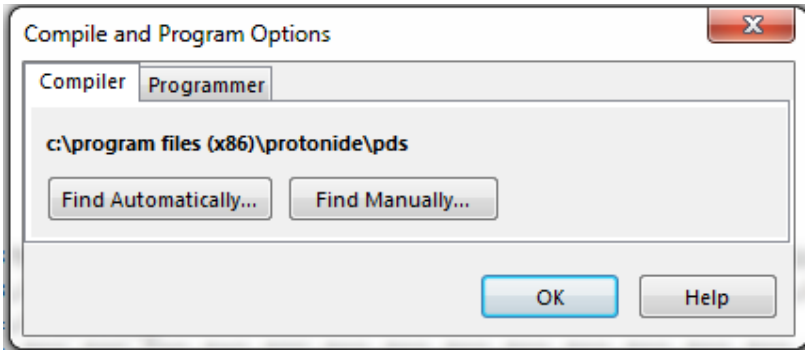
3. Proton IDE Compiler

Prilikom projektovanja električnih šema u ISIS-u koje sadrže mikrokontrolere, simulacija je nemoguća ukoliko u isti nije učitan .hex fajl na osnovu koga će raditi mikrokontroler. Heksadecimalni kod se dobija nakon kompajliranja koda koji je napisan u nekom od softverskih paket u nekom od programskih jezika. Jedan od takvih softverskih paketa je i Proton, koji u osnovi koristi Basic, tačnije PIC Basic. Basic je nešto nižeg nivoa u odnosu na kasnije nastali MikroC koji se takođe veoma dosta koristi za pisanje koda namenjene PIC-ovima. Na slici 31. može se videti radno okruženje Protona.



Slika 31. Radno okruženje Proton-a sa primerom koda

Pre početka pisanja koda, neophodno je podešavanje samog kompajlera, kako bi Proton detektovao unete PIC mikrokontrolere koji su uneti prilikom pisanja koda. Podešavanje se vrši tako što se izabere jedan od komandi označena sa „Set“ na slici 33., gde će biti ponuđeno automatsko i ručno (manulno) nalaženje kompajlera (slika 32.). Izabere se ručno nalaženje i ode se do foldera u kom je instaliran Proton i označi se folder označen sa **PDS**, zatim se to potvrdi i kompajler je spreman za unos prvog koda.



Slika 32. Podešavanje kompajlera

Pri izboru pisanja novog koda (slika 33.), moguće je podesiti zaglavlje (*header*) u kome se navodi datum pisanja koda, autor koda i ostale informacije kako o autoru tako i o samom kodu (1).

Pisanje koda počinje navođenjem mikrokontrolera koji se koristi (2), za šta se koristi naredba „*Device*“ koja označava da posle nje sledi navođenje koji se mikrokontroler koristi. Zatim se navodi koji oscilator se koristi (naredba *Xtal*), tj. na kojoj frekvenciji (radni takt) radi mikrokontroler. Kod nekih kontrolera postoji mogućnost izbora integrisanog (internog) oscilatora, naravno do neke određene frekvencije. Svi mikrokontroleri mogu da koriste eksterni oscilator, s tim što se tada povećava potrošnja energije, ali i povećava brzina instrukcija koje obavlja mikrokontroler. Dalje se navodi naredba *All_Digital* ***true*** kako bi kompajler znao da su svi pinovi digitalni. Nakon toga sledi navođenje osigurača (*Fuse*) gde se mogu birati interni oscilator (kod onih mikrokontrolera koji to poseduju), deaktiviranje (softversko, jer se to hardverski radi pull-up otpornikom od 10 kΩ) **MCLR** (Master Clear) pina, koji služi za resetovanje samog mikrokontrolera, kao i ostali osigurači vezani za rad mikrokontrolera.

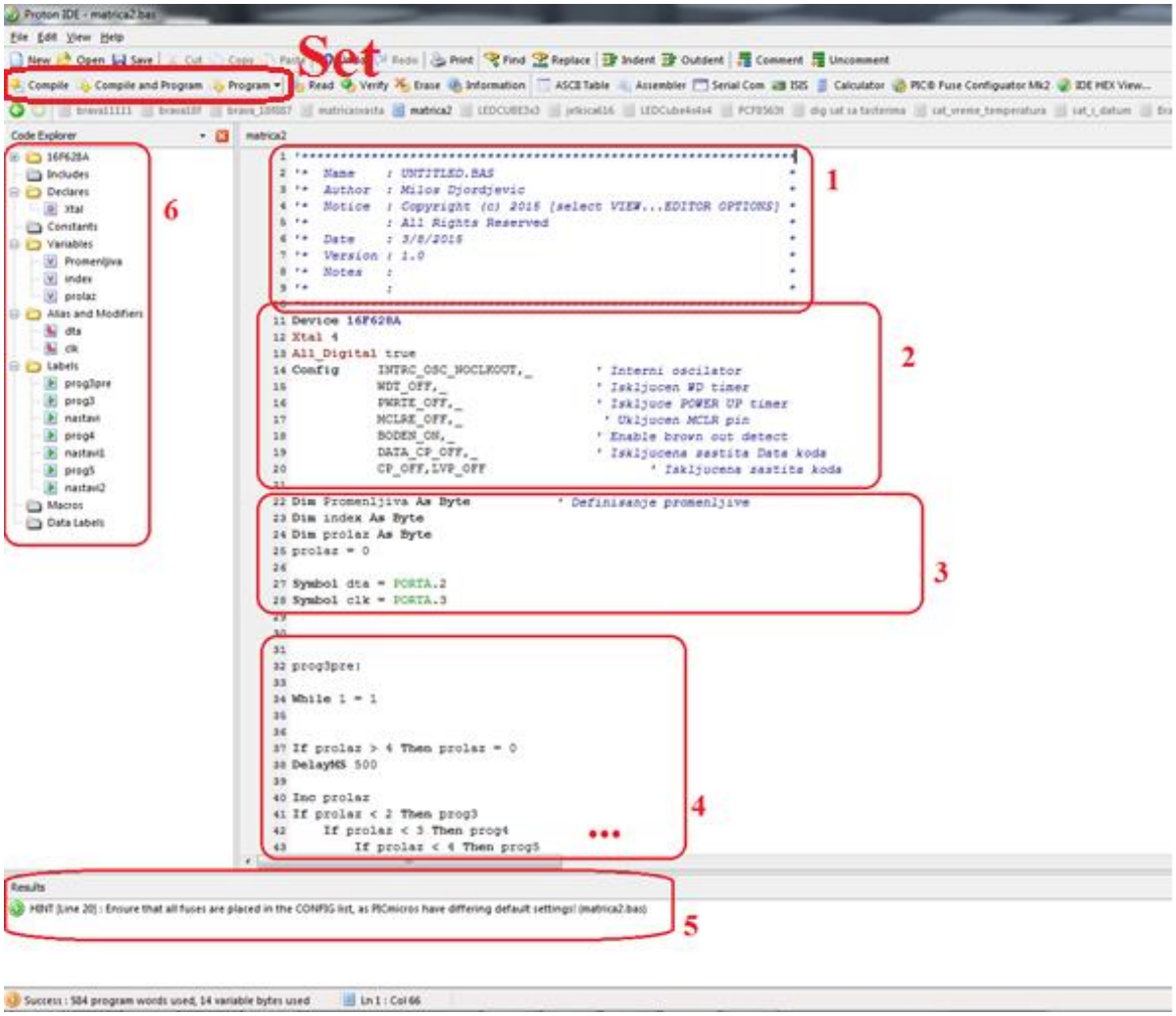
U delu koda koji je na slici označen brojem 3, vrši se definisanje promenljivih koje se koriste u toku pisanja koda (zadavanje nekih početnih vrednosti tim promenljivama), deklarisanje određenih pinova na određenu funkciju, zadavanje simbola nekim pinovima,kako bise isti u kodu mogli pozivati određenim korisnički navedenim imenima. Kao i kod drugih softverskih alata, nazivi promenljivih **ne mogu** počinjati brojem ili nekim specijalnim znakom, takođe i nekim već rezervisanim imenima (**button**, **PORT**, **dta**, **clk**,**...**).

Deo označen brojem 4 sadrži glavni deo koda. Kod treba da sadrži glavnu petlju (*main* – najčešće korišćena naredba kod drugih programskih jezika) u kojoj se odvija ceo kod (može i bez glavne petlje, ali će se ceo kod, cela funkcija koju treba da obavi kontroler izvršiti samo jednom). Da bi se funkcija koju treba mikrokontroler da izvrši, izvršavala neprekidno, da li samo softverski zavisna ili ako ista zavisi od nekog spoljnjeg signala (senzor, RTC, taster,...) navodi se naredba ***GoTo***. Ovom naredbom se nakon izvršenja instrukcije nekog potprograma ili glavnog programa, izvršenje nastavlja od samog početka glavne petlje. Zatim se unutar glavne petlje se navode instrukcije i druge petlje, koje mogu biti organizovane unutar manjih potprograma (potprogram predstavlja program unutar samog programa). Naredba koja predstavlja neku petlju kao što je naredba ***If***, ***While***,**...** moraju se na kraju naredbe i zatvoriti (***If*** se zatvara naredbom ***EndIf***, ***While*** naredbom ***Wend***).

Brojem 5 označen je przor u kome se može pratiti kompajliranje samog koda, gde se prikazuju greške (naravno ukoliko ih ima) prilikom pisanja koda, gde svaka greška ukazuje na liniju koda u kome se ista nalazi. Naravno, u tom istom prozoru se nalaze i informacije ukoliko je kod uspešno kompajliran, a i preostala memorija mikrokontrolera (izražena u broju programskih reči i varijabli).

Prozor označen brojem 6 predstavlja neki vid sadržaja samog koda, gde su navedeni mikrokontroler koji se koristi, deklarisanja, konstante, variable (promenljive koje menjaju svoju vrednost zavisno od softvera), kao i obične promenljive,...

Iz padajućeg menija unutar kartice „*Help*“, nalazi se podkartica na nazivom „*Documents*“ u kojoj se u vidu mini priručnika nalazi „*Proton Compiler Manual*“ u kome su ukratko navedene i opisane sve narebe, matematičke funkcije,... koje se koriste u Proton-u.



Slika 33. Način pisanja koda u Proton-u

4. Primeri primene optokomponenata

U ovom poglavlju biće dati primeri primena LED dioda, sedmosegmentnih displeja i LCD displeja. Uz svaki primer biće priložena električna šema, kao i objašnjenje koje prati primer od ideje do same realizacije. Takođe biće i nekoliko reči o mikrokontrolerima, pomeračkim registrima i drugim komponentama koje se koriste u ovim primerima.

Biće primera vezanih za LED diode, kako ih kontrolisati mikrokontrolerom, LED drajverima, kako kontrolisati svaku diodu pojedinačno pomeračkim registrom. Najpre će biti reči o primerima rada sa LED diodama i njihovoj kontroli, zatim primeri rada i kontrole 7 segmentnih displeja i LCD displeja. Na kraju će biti dati primeri primene spomenutih optoelektronskih komponenata u realizaciji određenim sistemima.

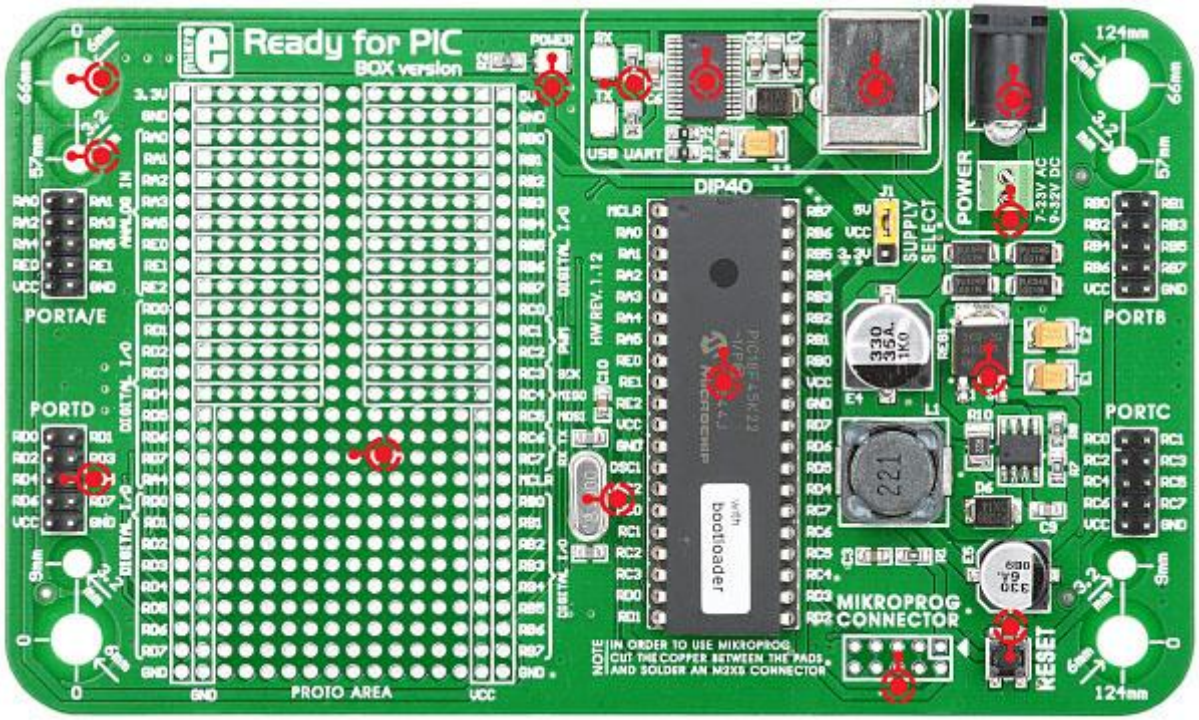
4a. Razvojno okruženje i mikrokontroler PIC18F45K22

Mikrokontroler je uređaj opšte namene, koji pribavlja podatke, obavlja ograničenu obradu nad tim podacima, i upravlja svojim okruženjem na osnovu rezultata izračunavanja. Mikrokontroler u toku svog rada koristi fiksni program koji je smešten u ROM memoriji, koja se ne menja u toku životnog veka sistema. U nekim primenama upravo ROM memorija može da bude presudna u izboru mikrokontrolera. Tako npr, ukoliko nam nije neophodan veliki broj pinova (I/O – GPIO (kod ***10FXXX*** i ***12FXXX*** mikrokontrolera)), ali kod sadrži više programskih reči nego što ROM mikrokontrolera može da sadrži. U tom slučaju mora se izabrati mikrokontroler sa većom ROM memorijom, iako to znači i skuplji mikrokontroler. Mikrokontroler sadrži sve gradivne blokove ***CPU***-a (ALU, PC, SP, registre i dr.), ali takođe i RAM, ROM, paralelne i serijske U/I (I/O) portove, generatore takta i dr.

Veliki broj ulazno-izlaznih pinova mikrokontrolera se može koristiti za više namena što se softverski definiše. Mikrokontroler komunicira sa spoljnim svetom (pribavlja i predaje podatke) preko svojih pinova, pri čemu je arihitektura i skup instrukcija projektovan za manipulisanje podacima obima bajt ili bit.

U primerima u ovom praktikumu biće korišćeni mikrokontroleri familije PIC - ***Pheripheral Interface Controller***. Neki mikrokontroleri poseduju interne oscilatore (smanjuju potrošnju energije), dok neki nemaju tu mogućnost (npr. PIC 16F877/877A), pa je neophodno koristiti eksterni oscilator. Prednost eksternog oscilatora je veća frekvencija izvršavanja instrukcija, što je kod nekih primena veoma bitno, naročit ukoliko se mikrokontroler koristi u aplikacijama u realnom vremenu. Veoma važnu ulogu igra i broj pinova opšte namene GPIO ili ulazno izlazni pinovi (U/I – I/O). Većina mikrokontrolera poseduje A/D i D/A konvertore, LCD drajvere i drugo. Većina mikrokontrolera poseduje i interapte (prekide, eng. *interrupt*), hardverski prekid najčešće je to na B port, tačnije pin B.0 (u primeru sa semaforom biće više reči o interaptima).

Mikrokontroler koji će biti korišćen u ovom praktikumu je ***PIC18F45K22***, koji se nalazi na razvojnom okruženju ***Ready For PIC*** (slika 34.), firme „***Mikroelektronika***“.



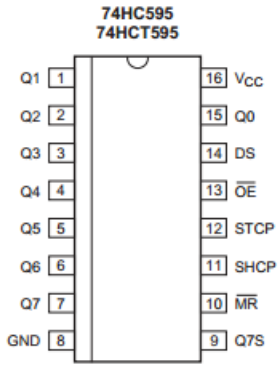
Slika 34. Razvojno okruženje „***Ready For PIC***“ firme „***Mikroelektronika***“ sa mikrokontrolerom ***PIC18F45K22***.

Pomenuti mikrokontroler je ***40***-o pinski mikrokontroler sa ***32 KB*** programske memorije, ***16 MIPS*** (broj instrukcija po sekundi). Ovom mikrokontroleru je na razvojnom okruženju pridružen ***eksterni kvarcni kristal*** frekvencije ***8 MHz*** (poseduje ***interni oscilator*** do ***64 MHz***). Takođe, *PIC18F45K22* poseduje ***2 UART*** modula, ***2 I2C*** i ***2 SPI*** modula za komunikaciju sa periferijama. Poseduje ***28 kanala A/D konvertora rezolucije 10 bita***, ***2 komparatora***. Napon napajanja ovog mikrokontrolera je u opsegu od ***1,8 – 5,5 V*** .

4b. *Pomerački registar 74HC595*

Registri su digitalna kola koja služe za smeštaj i pomeranje podataka unutar sistema. Za razliku od brojača, registri nemaju određenu sekvencu stanja. Smeštajni kapacitet registra (eng. *storage capacity*) je ukupan broj bitova koji mogu da se smeste u registar. Svaki stepen (eng. *stage*) u registru predstavlja flip-flop koji skladišti jedan bit. Prema tome, smeštajni kapacitet registra jednak je broju flip-flopova koje registar sadrži.

Simbol i pinout pomeračkog registra 74HCT595 je dat na slici 35.

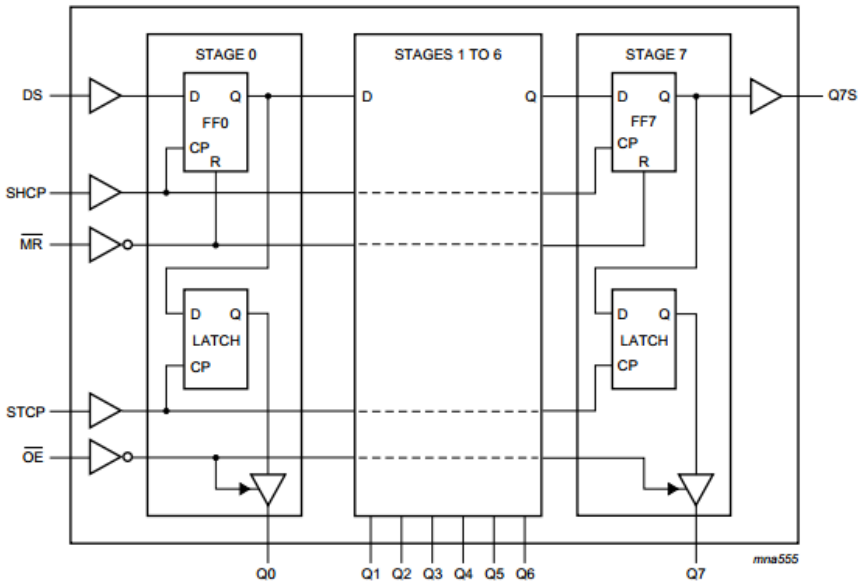


Symbol	Pin	Description
Q1	1	parallel data output 1
Q2	2	parallel data output 2
Q3	3	parallel data output 3
Q4	4	parallel data output 4
Q5	5	parallel data output 5
Q6	6	parallel data output 6
Q7	7	parallel data output 7
GND	8	ground (0 V)
Q7S	9	serial data output
$\overline{MR}$	10	master reset (active LOW)
SHCP	11	shift register clock input
STCP	12	storage register clock input
$\overline{OE}$	13	output enable input (active LOW)
DS	14	serial data input
Q0	15	parallel data output 0
V_{CC}	16	supply voltage

Slika 35. Pinout pomeračkog registra 74HCT595

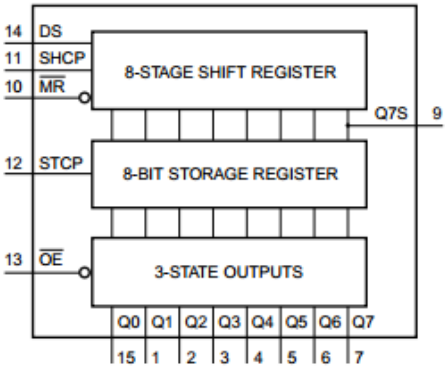
Pomerački registar 74HCT595 je veoma brzi ***Si-gejt CMOS*** uređaj i njegovi pinovi odgovaraju nisko – naponskoj Šotkijevoj TTL tehnologiji. 74HCT595 je 8-stepeni serijski pomerački registar sa registrima za skladištenje podataka i 3-statičkim baferima na izlazu. Registri poseduju odvojene takt signale.

Podaci se pomeraju na pozitivnu prednju ivicu ulaznog taktnog signala pomeračkog registra ((***SHCP***) može se videti u pinoutu (slika 35.)). Podaci svakog registra se prenose u registar za skladištenje na pozitivnu prednju ivicu ulaznog taktnog signala registra za skladištenje ((***STCP***) - pinout (slika 34.)). Ako su oba taktna signala (clock-a) vezana zajedno, pomerački registar će uvek imati jedan taktni signal ispred taktnog signala registra za skladištenje. Logički dijagram pomeračkog registra je dat na slici 36.



Slika 36. Logički dijagram pomeračkog registra 74HCT595

Pomerački registar ima serijski ulaz (**DS**) i serijski/paralelni izlaz (**Q7S**) – kaskadni. Takođe obezbeđuje asinhroni reset (active LOW) za svih 8 pomeračkih stanja. Registar za skladištenje ima 8 paralelnih 3-statičkih bafera (magistrala) na izlazu. Podaci se pojavljuju na izlazu kad god je omogućen OE kao LOW. Funkcionalni dijagram je dat na slici 37.



Slika 37. Funkcionalni dijagram pomeračkog registra 74HCT595

Neke najosnovnije informacije o ovom pomeračkom registru (74HCT595):

8 – serijskih ulaza;

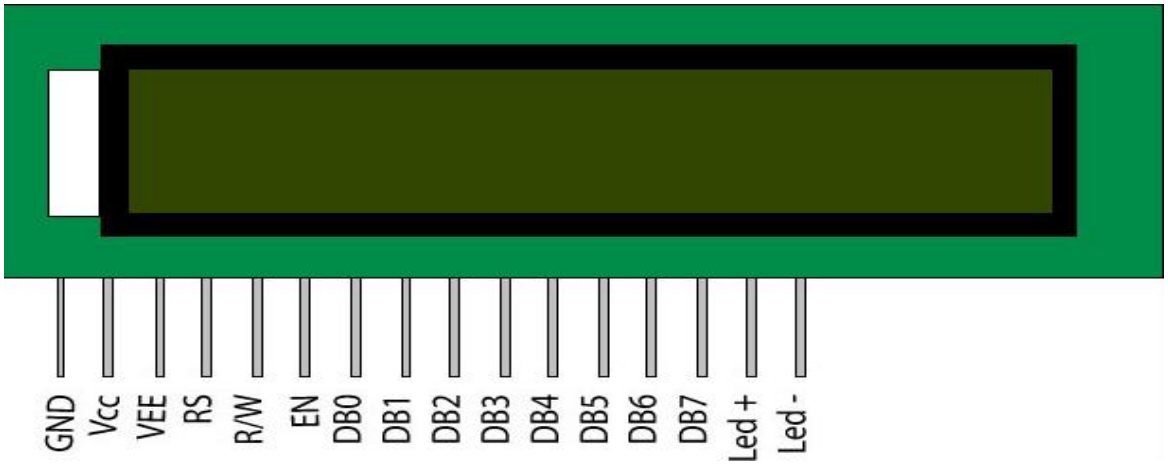
8 – serijski/paralelni izlaz;

Registar za skladištenje sa 3-statičkim baferima;

100 MHz (tipično) pomeračka frekvencija.

4c. LCD displej (LM016) 2x16 karaktera

Komponenta koje je služila za interakciju sa korisnikom je LCD displej. U ovom praktikumu je korišćen LCD displej sa dve kolone po 16 karaktera, odatle i oznaka 2x16. Rad ovog displeja zasnovan je na tečnim kristalima. Mikroprocesor koji pokreće ovaj LCD displej je **Hitachi HD44780U**. LCD panel ima fiksan broj ćelija sa tečnim kristalima i može da prikaže samo jednu rezoluciju na punoj veličini ekrana, koristeći jednu ćeliju po pikselu. Manje rezolucije mogu da se prikažu koristeći samo deo ekrana. Prikaz karaktera na ovom LCD displeju je rezolucije 5x7 piksela. LCD displej korišćen u ovom praktikumu je alfanumerički, što znači da on u svojoj memoriji već poseduje sve „***ASCII***“ karaktere (***0-127***) koje može da prikaže. Naravno, tu se mogu koristiti i karakteri iz prožirene „ASCII“ tabele počevši od 128 do 255. U slučaju da korisnik želi da prikaže karaktere koje mikroprocesor u LCD-u ne poseduje u memoriji, moguće je isprojektovati ih. U te svrhe mogu se koristiti razni online ili offline softverski alati. Tako na primer korisnik može isprojektovati simbol koji može da zauzima do 8 polja za prikaz karaktera. Pinout ovog LCD displeja je dat na slici 38.



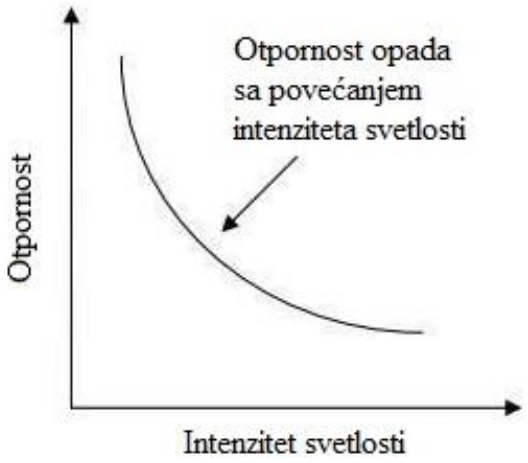
Pin No	Function	Name
1	Ground (0V)	Ground
2	Supply voltage; 5V (4.7V – 5.3V)	Vcc
3	Contrast adjustment; through a variable resistor	V_{EE}
4	Selects command register when low; and data register when high	Register Select
5	Low to write to the register; High to read from the register	Read/write
6	Sends data to data pins when a high to low pulse is given	Enable
7	8-bit data pins	DB0
8		DB1
9		DB2
10		DB3
11		DB4
12		DB5
13		DB6
14		DB7
15	Backlight V_{CC} (5V)	Led+
16	Backlight Ground (0V)	Led-

Slika 38. Pinout LCD displeja tipa 2x16 karaktera

Kako se može videti osim pinova za napajanje i regulisanje osvetljenosti samog displeja, postoje pinovi za izbor komandnog registra (nisko logičko stanje) i registra za podatke (visoki logički nivo). Takođe, tu je i pin za upis/čitanje (***R/W***), koji nije korišćen u primerima i vezan je bio direktno na masu (***GND***). Tu su i pinovi za podatke, kojih ima 8, od kojih se najčešće koriste samo 4 pina za komunikaciju. Pinovi počev od ***DB0*** do ***DB3*** su vezani na masu (***GND***) i ne učestvuju u komunikaciji, dok se za komunikaciju koriste pinovi počev od ***DB4*** do ***DB7***. U slučaju da korisnik želi da poveća brzinu prenosa podataka za ispis na LCD displej, koriste se svih 8 pinova za podatke (DB0 – DB7). Regulisanje osvetljenosti koje je gore navedeno, vrši se uspomoć linearnog potenciometra vrednosti od ***10 kΩ***, čijom promenom otpornosti se menja i osvetljenje na samom LCD displeju.

4d. Fotootpornik

Fotootpornici su poluprovodnički otpornici kod kojih se otpornost smanjuje pod uticajm svetlosti (slika 39.).



Slika 39. Zavisnost otpornosti u odnosu na intenzitet svetlosti

Rad poluprovodničkih fotootpornika zasnovan je na efektu fotoprovodnosti (unutrašnjem fotoelektričnom efektu). Na taj način se mogu realizovati razni senzori gde se zavisno od otpornosti, na osnovu boje predmeta sa kog se očitava, ili intenziteta osvetljenosti određuje boja, pobuđuju razni motori ili delovi uređaja. Jedan tip fotootpornika je dat na slici 40.

Slika 40. Fotootpornik

4d. Optičko vlakno

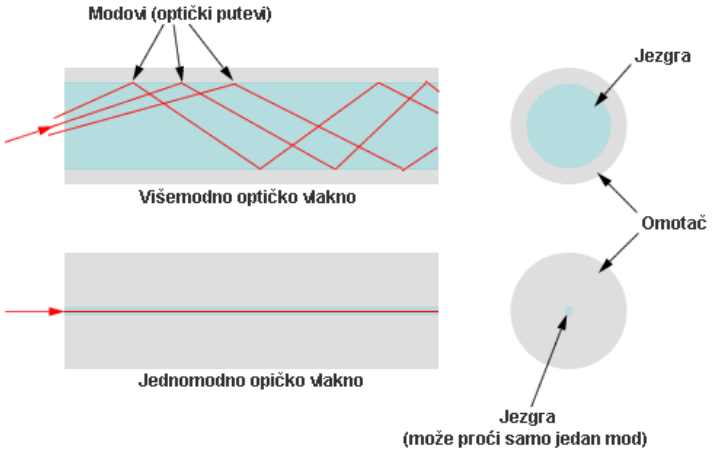
Optička vlakna jesu dielektrične niti kružnog preseka koja se sastoje od dva sloja, unutrašnjeg-jezgra i spoljašnjeg-omotača vlakna. Indeks prelamanja jezgra je nešto veći od indeksa prelamanja omotača. Svetlosni zraci koji ulaze u jezgro vlakna pod malim uglom, nailaze na optički ređu sredinu na granici jezgra i omotača, pod uglom koji je veći od graničnog usled čega dolazi totalne refleksije.

Elementi samog optičkog vlakna, kao što je prikazano na slici 41. su jezgro, košuljica i spoljašni omotač. Materijal jezgra je ili plastika ili staklo. Košuljica je od materijala čiji je indeks refleksije manji od indeksa refleksije jezgra. Totalna unutrašnja refleksija na spoju

jezgro/košuljica usmerava svetlo da putuje unutar jezgra. Vlakna sa plastičnim jezgrom takođe imaju plastičnu košuljicu. Ovakva vlakna pokazuju velike gubitke, ali se široko koriste za prenos na kratke razdaljine. Višekomponentna stakla koja sadrže nekoliko oksida se koriste za sve sem za vlakna sa najmanjim gubicima, koja su obično napravljena od čistog silicijuma. U vlaknima sa malim ili srednjim gubicima stakleno jezgro je okruženo sa staklenom ili plastičnom košuljicom. Spoljašnji omotač je elastičan, otporan na nagrizanje obično plastični materijal koji uvećava mehaničku snagu vlakna i obezbeđuje mehaničku izolaciju od geometrijskih neregularnosti, distorzija ili hrapavosti susednih površina koje mogu inače uzrokovati gubitke usled rasejanja kada je vlakno ugrađeno u kablove ili podržano drugim strukturama.

Slika 41. Elementi optičkog vlakna

Talasna širina optičkih vlakana je u opsegu od ***180 THz*** do ***330 THz***. Postoje dvije vrste optičkih vlakana: jednomodno i višemodno optičko vlakno (slika 42.).



Slika 42. Vrste optičkih vlakana (višemodno i jednomodno)

Jednomodno optičko vlakno kroz jezgro propušta samo jedan zraku svetlosti. Kao izvor svetla koristi se laser, što značajno povećava brzinu prienosa podataka i udaljenost na koju se mogu preneti.

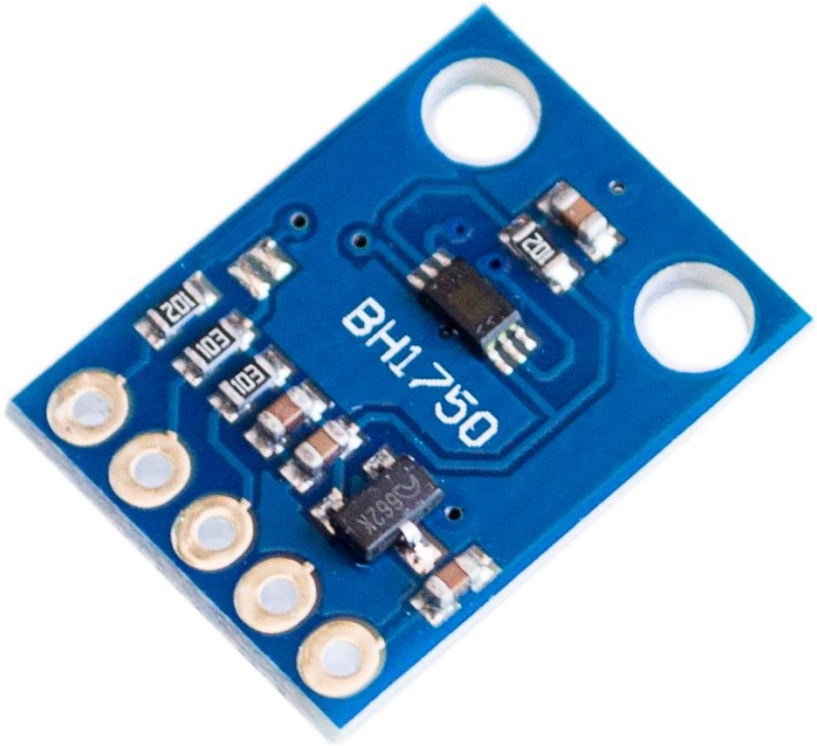
Višemodno optičko vlakno kao izvor svetlosti koristi LED diodu, jeftinije je od jednomodnog vlakna. U takvo vlakno ulazi više zraka koji totalnom refleksijom putuju kroz vlakno. Postoji ograničeni broj optičkih puteva kojima zrake svetlosti mogu putovati kroz vlakno. Ti optički putovi se zovu "modovi" vlakna.

Po jedno vlakno se koristi za svaki smer komunikacije. Jedan optički kabl u sebi obično sadržava više optičkih vlakana, obično 2 do 48.

4e. Senzor vidljive svetlosti BH1750

BH1750 je digitalni senzor ambijentalne svetlosti koji se kontroliše već realizovanom I2C magistralom. Ovaj senzor je veoma pogodan za kontrolu osvetljenja LCD displeja, realizaciju Luxmetara, kontrolu osvetljenja u prostorijama,... Ovaj senzor ima široki opseg detekcije intenziteta svetlosti od 0 do 65535 lx pri visokoj rezoluciji. Pomenuti senzor poseduje 16-bitni A/D konvertor, koji omogućava visoku rezoluciju merenja.

Izgled senzora BH1750 je dat na slici 43.



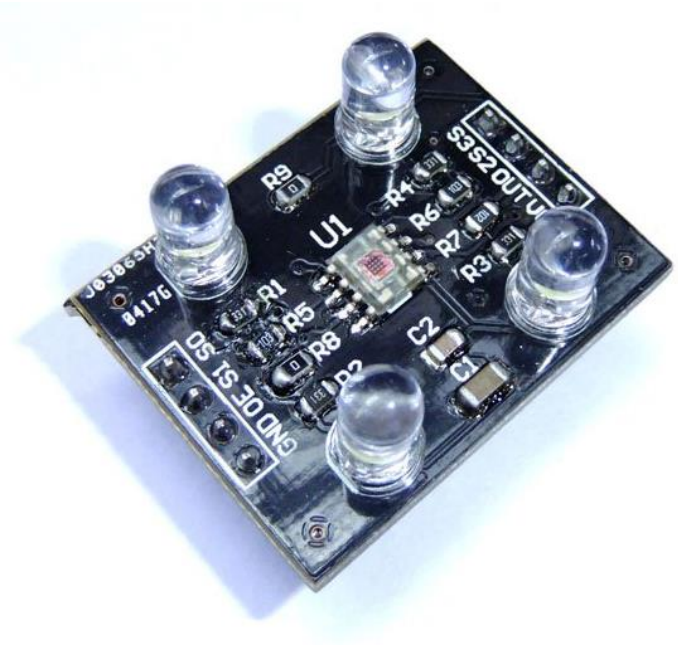
Slika 43. BH1750 – senzor vidljive svetlosti

4f. Senzor boje TCS230/3200

Jedan od najinteresantnijih primera koji će biti opisani u ovom praktikumu je svakako senzor boje. Senzor boje je moguće realizovati na više načina, od kojih su neki mogu realizovati uspomoć fotootpornika, fototranzistora,... Nesavršenost ovako realizovanih senzora je nepreciznost u detektovanju boje i njene talasne dužine, kao i veoma komplikovana kalibracija. Za realizaciju takvih senzor boje, neophodno je koristiti tri LED diode - crvenu, plavu i zelenu LED diodu (realizacija osnovnih boja - ***RGB***), koje imaju isti ugao svetlosti (što širi ugao), istu jačinu svetlosti, kao i druge potrebne parametre. Osim toga, nophodno je izolovati kompletno realizovan senzor od spoljašnjih zračenja, kako nebi dolazilo do smetnji pri detekciji. Nakon toga, potrebno je aktivirati A/D konvertor u mikrokontroleru i izvršiti kalibraciju senzora. Jedan od načina je osvetliti belu i crnu površinu, kako bi se odredile gornja

i donje granica osetljivosti. Nakon toga treba odrediti opsege u koje spadaju pre svega osnovne boje (crvena, plava i zelena), da bi se kasnije spektar boja proširio. Takođe, senzor boje i sama njegova osetljivost, tj. preciznost zavisili bi i od samog fotootpornika. Tako da bi se pri promeni fotootpornika morala vršiti ponovna kalibracija senzora.

Kako bi se izbegao dug proces realizacije senzora boje, koji na kraju i nema dovoljnu preciznost, moguće je koristiti jedan od najpreciznijih senzora za boju ***TCS230/3200***. Ovaj modul senzora boja sadrži ***TCS230***, programibilni konverter svetlosti u frekvenciju (struje u frekvenciju) (slika 44.), u kombinaciji sa silicijumskim fotodiodama na jednom monolitskom CMOS integrisanom kolu. Izlazni signal je kvadratnog oblika (50% ***duty cycle***) sa frekvencijom direktno proporcionalnom intenzitetu svetlosti. Koristi se u robotici za prepoznavanje objekata, praćenje "linija" i drugo.



Slika 44. Senzor boje TCS230/3200

TCS230 senzor boje kako je već navedeno, sastoji se od kombinacije fotodioda i konvertora struje u frekvenciju. Sadrži matricu 8x8 fotodioda (slika 45.), i to 16 s ***plavim***, 16 s ***zelenim***, 16 s ***crvenim*** filterom i 16 ***bez filtera*** (*clear*). Uspomoć lupe može se videti matrica fotodioda. Veličine ovih fotodioda su 120x120 μm, a služe kako bi se filtrirala "slučajna" zračenja. Na modulu se nalaze 4 širokougaone 5 mm hladno bele LED diode, kako bi se povećalo zračenje i ostale pasivne komponente koje su neophodne za direktno čitanje frekvencije sa izlaznog pina.

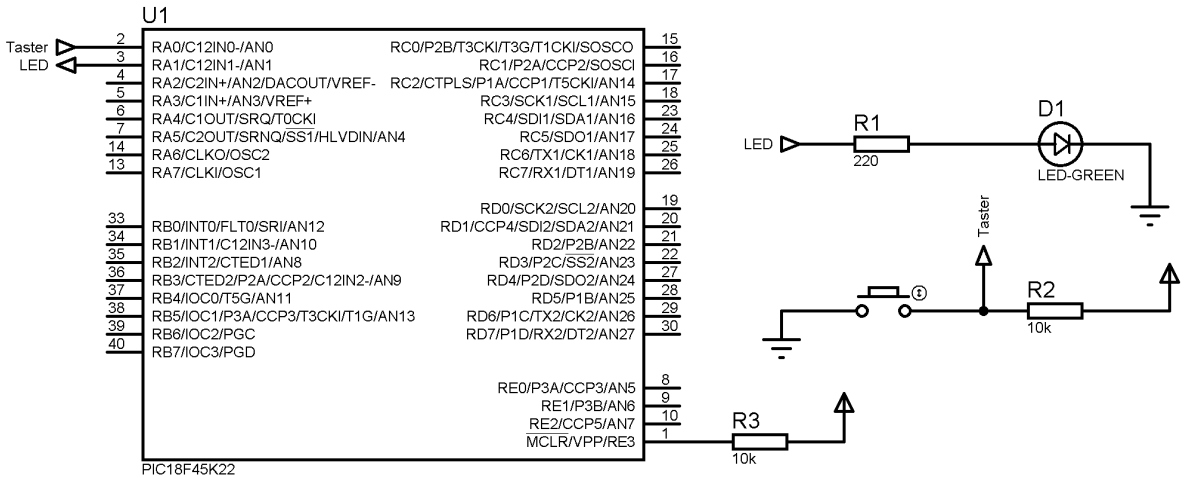
Slika 45. Matrica fotodioda za detekciju boje

Ovaj senzor poseduje 8 pina, od kojih se dva koriste za napajanje (Vcc i GND), koje je 5 V. Pored pinova za napajanje, tu je i OE (***O**utput **E**nable*), kojim se omogućuje rad, odnosno sama detekcija boja. Ovaj pin se direktno vezuje na GND (masu), pošto je njegov ulaz invertovan, odnosno, neophodna mu je logička nula za ispravan rad. Tu su i pinovi ***S0***, ***S1***, ***S2*** i ***S3***, koji služe za detekciju boja. Naime, S0 i S1 pinovi se koriste kako bi se odredio opseg frekvencije za detekciju, odnosno, preciznost. Promenom logičkih stanja (1 i 0) na pinovima S2 i S3 se aktiviraju fotodiode za crvenu, plavu ili zelenu boju. Na kraju tu je i izlazni pin, (***OUT***), sa kog se očitava frekvencija proporcionalna intenzitetu boje. Više o povezivanju i detekciji uspomoć ovog senzora biće dato u samom primeru realizacije Kolorimetra.

Podsećanje

4.1a. Simulacija kontrole LED diode tasterom

U prethodnom primeru je korišćena LED dioda, kako bi se prikazala mogućnost njenog kontrolisanja mikrokontrolerom. U ovom primeru LED dioda će se kontrolisati tasterom, tj. pritiskom na taster povećavaće se vrednost promenljive, kojom je definisano kada LED dioda treba svetleti. Na slici 46. je prikazana električna šema simulacije.



Slika 46. Simulacija kontrole LED diode tasterom

U simulatoru se koristi napajanje koje konstantno ima 5 VDC napona, pri kome bi u realnosti LED diode, iako imaju svoju unutrašnju otpornost, mogle da izgore ako su dugo priključene na to napajanje. Zbog toga treba paziti na koju vrednost napona se LED diode priključuju. Ukoliko se koristi 5 V za napajanje, ispred LED diode se postavlja otpornik, koji ograničava struju kroz LED diodu i štiti je (u simulatoru to nije neophodno). Način određivanja vrednosti otpornika koji se vezuje na red sa LED diodom je sledeći:

$$R_D = \frac{V_{DD} - V_D}{I_D} = \frac{5 - 0{,}7}{20 \cdot 10^{-3}} = \frac{4{,}3}{20 \cdot 10^{-3}} = 0{,}215 \cdot 10^3 = 215\ \Omega \rightarrow \underline{\mathbf{220\ \Omega}}$$

220 Ω je vrednost otpornosti otpornika koji se vezuje na red sa LED diodom jer iz E24 niza (najčešće korišćeni niz) vrednosti otpornika (sa ±5 % tolerancije) se može videti da je ta vrednost najbliža dobijenoj vrednosti (215 Ω).

Pre povezivanja bilo koje komponente na napajanje, najbolje je konsultovati *datasheet* (tehnička dokumentacija komponente). U datasheet-u su navedene sve informacije vezane za datu komponentu, kako o maksimalnom primenjenom naponu, temperaturi (ukoliko se radi o senzoru ili ako se neka komponenta koristi na visokim/niskim temperaturama,...).

LED dioda je vezana na pinu RA.1, dok je taster u ***pull-up*** konfiguraciji vezan na pinu RA.0 mikrokontrolera. *Pull-up* konfiguracija predstavlja vezu tastera i otpornika, gde je jedan kraj tastera vezan na masu (***GND***), a jedan kraj otpornika se vezuje na napajanje (**V_{DD}**),

dok je drugi kraj tastera vezan za drugi kraj otpornika, gde se inače i vrši detekcija pritiska tastera. Mikrokontroler će detektovati promenu stanja na svom pinu kada dođe do promene sa logičke ***1*** na logičku ***0***.

Najpre će se LED dioda kontrolisati, tj. kontrolisati kada svetli ili ne svetli, zavisno od toga da li je taster pritisnut ili ne. Nakon toga, pritiskom na taster, promenljivoj koja je definisana u kodu, povećavaće se vrednost. LED dioda će svetleti kada vrednost promenljive bude jednaka vrednosti koja jednaka definisanoj vrednosti.

Primer prvog koda kontrole LED diode tasterom je sledeći:

```
Device 18F45K22                    ' Definisanje mikrokontrolera

Xtal 32                            ' Deklarisanje frekvencije mikrokontrolera
All_Digital true                   ' Definisanje svih pinova kao izlazne

    Symbol Taster = PORTA.0        ' Definisanje pinova/promenljivih kao
konstante
    Symbol LED = PORTA.1           ' Definisanje pinova/promenljivih kao
konstante


Main:                              ' Potprogram u kome se izvrsava kod

    If Taster = 0 Then             ' Ispitivanje da li je taster pritisnut
        High LED                   ' Taster pritisnut -> upali LED diodu
    Else                           ' Taster nije protosnut ->
        Low LED                    ' -> ne pali LED diodu
    EndIf                          ' Zatvaranje IF uslova

GoTo Main                          ' Povratak u potprogram, kako bi se kod
izvrsavao beskonacno dugo
```

Deo koda u kome je naredba „***Delay***“ (kašnjenje, pauza,...) služi za hardkodirano (ručno) i softverski (preko ***variable - promenljive***) zadavanje vrednosti vremena koliko traje određena aktivnost mikrokontrolera. ***DelayMS*** znači da se vreme predstavlja u milisekundama (***ms***), pored toga moguće je i predstavljanje u mikrosekundama (***μs***), tada ova naredba ima oblik ***DelayUS***.

Iz priloženog koda se može videti da se provera da li je taster pritisnut ili ne izvršava u ***If*** petlji. U ***Proton Basic***-u se If petlja može definisati na dva načina:

I način:

If *<uslov>* ***Then***

<naredba>

<naredba>

...

<naredba>

Endif

Vidi se da se If petlja pokreće naredbom ***If***, nakon čega sledi uslov koji treba ispitati, zatim naredba ***Then***, nakon koje sledi navođenje naredbe/naredbi koje treba izvršiti ukoliko ispitivani uslov važi. Na kraju petlje, ***If*** petlja se **mora** zatvoriti naredbom ***EndIf***.

II način:

If *<uslov>* ***Then*** *<naredba>* , gde se If petlja ne mora zatvoriti naredbom ***EndIf***, pošto se otvorena If petlja završava u istom redu gde je i započeta.

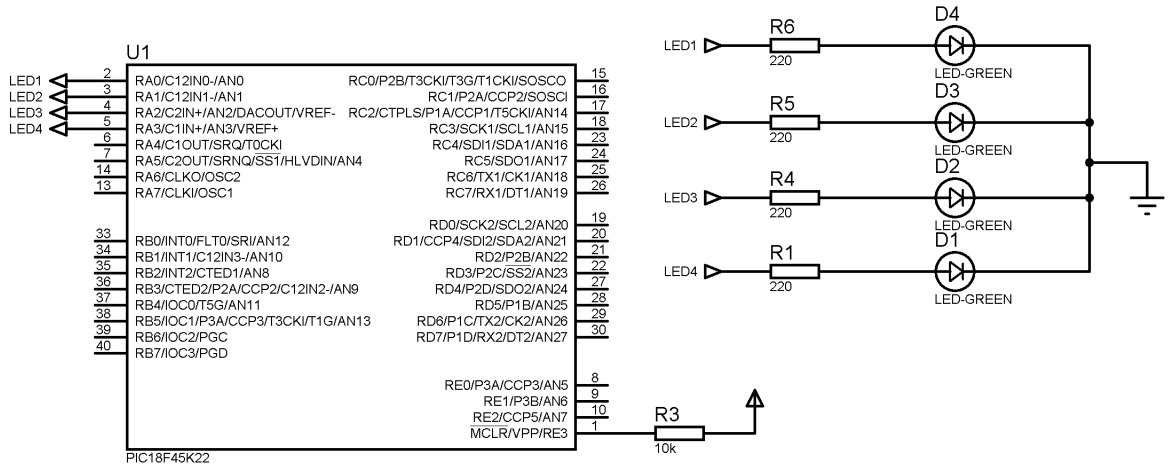
Vrlo često se može sresti potreba za LED indikacijom u raznim optoelektronskim uređajima kada neka promenljiva dostigne određena vrednost. Primer toga može se demonstrirati upravo prethodno gore korišćenog tastera. Električna šema je ista, kod zadužen za proveru tastera i vrednosti promenljive je:

```
Device 18F45K22                          ' Definisanje mikrokontrolera

Xtal 32                                  ' Deklarisanje frekvencije
mikrokontrolera
All_Digital true                         ' Definisanje svih pinova kao izlazne

    Dim Promenljiva As Byte              ' Definisanje promenljivih
    Promenljiva = 0                      ' Dodeljivanje vrednosti promenljivoj

    Symbol Taster = PORTA.0              ' Definisanje pinova/promenljivih kao
konstante
    Symbol LED = PORTA.1                 ' Definisanje pinova/promenljivih kao
konstante


Main:                                    ' Potprogram u kome se izvrsava kod

    If Taster = 0 Then                   ' Ispitivanje da li je taster
pritisnut
        Inc Promenljiva                  ' Taster pritisnut -> uvecaj vrednost
promenljive
        DelayMS 500                      ' 500ms pauze u proveravanju da li je
pritisnut taster
        If Promenljiva = 3 Then          ' Ukoliko je dostignuta vrednost
promenljive = 3
            High LED                     ' LED dioda svetli
        ElseIf Promenljiva > 3 Then      ' Ukoliko je vrednost promenljive
veca od najvece definisane
            Low LED                      ' LED dioda ne svetli
            Promenljiva = 0              ' Resetovanje vrednosti promenljive
na 0
        EndIf
    EndIf


GoTo Main                          ' Povratak u potprogram, kako bi se kod
izvrsavao beskonacno dugo
```

Razlika u odnosu na prethodni kod je definisanje promenljive ***Promenljiva*** koja je tipa ***Byte*** (***0-255***). U potprogramu u kome se izvršava kod osim ispitivanja *If* uslova da li je taster pritisnut postoje još dva *If* uslova. U prvom *If* uslovu za proveru tastera, ukoliko važi uslov, vrednost promenljive *Promenljiva* povećaće se za 1 naredbom ***Inc*** *Promenljiva*. Naredba *DelayMS 500*, služi kako mikrokontroler u trajanju tih 500 (ms) milisekundi ne bi vršio proveru da li je taster pritisnut, nakon čega sledi nova provera. Kako je u kodu navedeno da LED dioda svetli kada je vrednost promenljive *Promenljiva = 3*, znači da bi taster trebao da bude pritisnut ne duže od *3 ms*. Kako je praktično nemoguće da čovek taster pritisne i odpustiti kraće od ≈ *200 ms*, vrednost promenljive bi bila ≈ 200. U drugom uslovu se ispituje vrednost promenljive kada treba LED dioda da svetli (bez DelayMS 500 (500 ms je neko okvirno vreme dovoljno da se jednom pritisne i otpusti taster, a da se vrednost promenljive poveća za 1)). U trećem uslovu kada se vrednost promenljive poveća na vrednost veću od one definisane u uslovu, vrednost promenljive se resetuje i postavlja na 0.

4.1b. Simulacija rada niza od 4 LED diode

U prethodnom primeru je korišćena jedna LED dioda, kako bi se prikazala mogućnost njenog kontrolisanja. U ovom primeru će biti dato kontrolisanje 4 LED diode (slika 47.), koje su vezane svaka zasebno na mikrokontroler iz prethodnog primera. Za kontrolisanje LED dioda iskorišćeno je 4 pina A porta. Tako je prva LED dioda (LED1) vezana na pin A.0 (***PORTA.0***), druga na A.1 (***PORTA.1***), treća na A.2 (***PORTA.2***) i četvrta na A.3 (***PORTA.3***). LED diode će se paliti zajedno sve odjednom, svaka za sebe, smenjivati, itd. Uz primer će biti i dato objašnjenje na koji način se vrši njihova kontrola.



Slika 47. Električna šema niza od 4 LED diode

U simulaciji za koju je realizovana električna šema sa slike 44. predstavljeno je 4 moda rada. Prelaz iz jednog moda rada u drugi vrši se na osnovu inkrementiranja (uvećavanja promenljive koja je za to zadužena). U ovom primeru je na početku definisana promeljiva (***Dim prolaz As Byte***), ime promenljive je na korisniku. Zatim je zadata vrednost promenljivoj (***prolaz = 0***). Vrednost je stavljena na 0 iz razloga da bi bilo lakše određivati vreme trajanja svakog od modova (promenljiva može početi od bilo koje vrednosti, npr. *prolaz = 123*). Na

početku glavne petlje se navodi do koje vrednosti se uvećava promenljiva za prelaz iz moda u mod, da bi se nakon određene vrednosti ista setovala na 0 i počelo izvršavanje kod iz početka.

Prvi mod se sastoji od toga da se pale jedna za drugom LED diode, tako da kada se prva LED dioda ugasi, upaliće se druga i tako do 4. Kao iz prethodnog primera, portovi na kojima su vezane LED diode, su imenovani u softveru, tako da se zaizvršenja pozivaju pod tim imenom, kako bi se olakšalo kodiranje. Slično je i što se tiče paljenja i gašenja LED dioda, koristi se naredba ***High/Low*** (može se koristiti i zadavanje vrednosti pinovima (npr. ***LED1 = 1*** (u slučaju da želimo da LED dioda zasvetli), a da bi je ugasili ***LED1 = 0***, mada je High/Low češće korišćena naredba)). Takođe vremenski period svetljenja se određuje naredbom Delay kao u prošlom primeru.

Kada vrednost promenljive dođe do maksimalne vrednosti određene za trajanje prvog moda rada, mikrokontroler prelazi na izvršenje drugog dela koda. Drugi mod je realizovan tako da se pale diode jedna za drugom, ali s tim što se prethodno upaljena dioda ne gasi, već nastavlja da svetli. Kada zasvetle sve LED diode, gasiće se istim redosledom kojim su se palile tj. od prve prema četvrtoj, tako da se stiče vizuelni utisak popularne „zmijice".

U trenutku kada promenljiva dostigne maksimalnu vrednost predviđenu za izvršavanje drugog moda, prelazi se u treći mod rada. U trećem modu se LED diode pale po rasporedu kao u prva dva moda, samo što se ovde gase obrnutim redosledom, što znači da će se najpre ugasiti četvrta LED dioda, pa treća, druga i na kraju prva. Vreme paljenja i gašenja je isto i u drugom i u trećem modu, to vreme može da se menja kako bi se ostvario lepši vizuelni efekat.

Na kraju prilikom dostizanja maksimalne vrednosti promenljive za prelaz, mikrokontroler prelazi iz trećeg moda u poslednji četvrti mod. U ovom modu se LED diode pale u parovima (dve po dve). Najpre se pale prva i treća (pinovi A.0 i A.2 su na logičkoj jedinici (High stanje)). Nakon 250 ms (koliko traje vreme u kome LED diode svetle (***DelayMS 250***)), ugasiće se prva i treća (Low stanje), a upaliće se druga i četvrta (pinovi B.1 i B.3 su aktivni i na High nivou (logička 1)). Ponovo nakon 250 ms se menja stanje i tada se druga i četvrta gase, a pale se prva i treća i tako naizmenično određeno vreme dok promenljiva za prelaz i moda u mod ne dostigne maksimalnu vrednost. Nakon dostizanja maksimalne vrednosti, promenljiva se setuje na 0 (početnu vrednost) i mikrokontroler počinje ponovno izvršavanje koda.

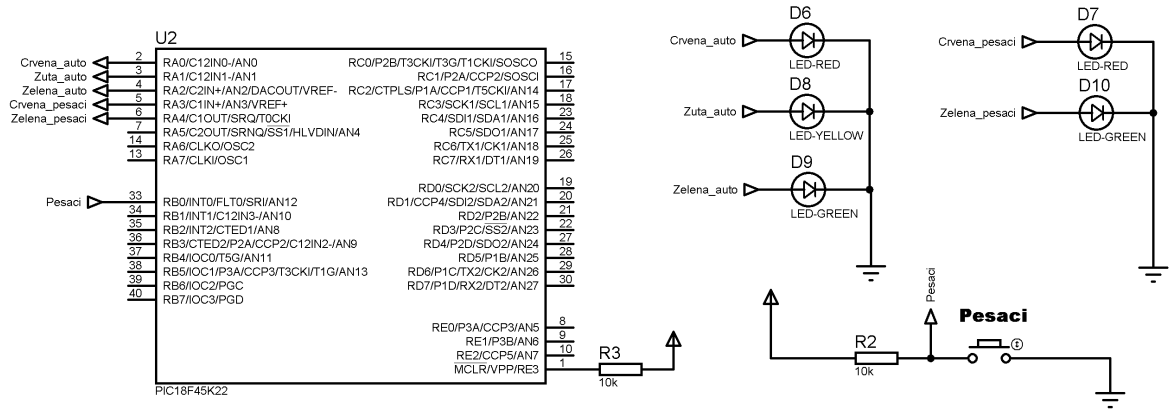
Prilikom izvršenja prvog potprograma (sekvence – svaki od modova rada je zadat u vidu pojedinačnog potprograma) koda u svakom modu (do dostizanja maksimalne vrednosti za promenljivu za prelaz), vrši se provera da li je dostignuta ta vrednost. To se izvršava pomoću naredbe „***GoTo main***" kojom se u trenutku izvršenja potprograma, mikrokontroler prebacuje u gavnu petlju (glavni program) u kome se vrši ispitivanje (npr: ***If prolaz < 10 Then*** nešto). Ukoliko promenljiva još nije dostigla vrednosti kojom prelazi iz jednog moda u drugi, nastavlja se izvršavanje potprograma, sve do njenog dostizanja. Kada ista dostigne maksimalnu vrednost predviđenu za jedan mod, mikrokontroler to detektuje, automatski se prelazi u drugi mod rada.

4.1c. Simulacija rada semafora

Na osnovu prethodnog primera, upozanli smo se sa kontrolisanjem više od jedne LED diode. U ovom primeru biće date dve simulacije realizacije rada semafora, u kojima je osim semafora za automobile dat i semafor za pešake. Simulacija predstavlja realizaciju semafora koji se često može sresti ispred škola, gde pritiskom na taster dolazi do promene svetla na semaforu i moguć prelazak pešaka (učenika).

Semafori su usklađeni tako da kada je za vozače upaljeno zeleno svetlo, za pešake je upaljeno crveno svetlo. U realnom slučaju se svetlo za pešake, bilo crveno ili zeleno, pali pre promene sa zelenog na crveno – crvenog na zeleno svetlo za automobile, iz razloga bezbednosti pešaka. Isti pristup je i ovde, dakle pre svake promene sa crvenog na zeleno svetlo za automobile, menja se i sa zelenog na crveno za pešake. U suprotnom, pre promene sa zelenog na crveno, promeniće se i svetlo na semaforu za pešake.

Električna šema simulacije rada semafora data je na slici 48.



Slika 48. Električna šema semafora sa tasterom za pešake

Sa električne šeme se može videti da su LED diode koje predstavljaju svetla semafora vezane na ***port A***. Tako je semafor za automobile vezan na pinove ***A.0***, ***A.1*** i ***A.2*** (crveno, žuto i zeleno svetlo, respektivno), a semafor za pešake n a pinove ***A.3*** i ***A.4*** (crveno i zeleno svetlo). Radi preglednosti same električne šeme, linije za povezivanje semafora za pešake nisu vučene preko cele šeme, već su iskorišćeni terminali (poglavlje 1.4) za izlaz iz mikrokontrolera, odnosno ulaz na anodu LED diode. Kao i u prethodnim primerima pinovima porta A su najpre definisana imena, kojima će se oni pozivati tokom koda, radi lakšeg snalaženja prilikom pisanja istog. Tako su na primer zadata imena sledećim pinovima:

```
Symbol Crvena = PORTA.0
Symbol Zuta = PORTA.1
Symbol Zelena = PORTA.2
Symbol Crvena_pesaci = PORTA.6
Symbol Zelena_pesaci = PORTA.7
```

Prilikom pokretanja simulacije, na semaforu će biti uključeno crveno svetlo za automobile i zeleno svetlo za pešake (**High** Crvena/**High** Zelena_pesaci). Nakon 4 sekunde (moguće zadavanje proizvoljne vrednosti vremenskog perioda), upaliće se žuto svetlo

(**High** Zuta), a istog momenta će se desiti promena sa zelenog na crveno svetlo za pešake (**Low** Zelena_pesaci/**High** Crvena_pesaci), kako bi se pešački prelaz na vreme raščistio za prolaz automobila.

U glavnom delu koda (***main***) zadata naredba ***While*** unutar koje je deo koda koji kontroliše paljenje i gašenje LED dioda. Naredba ***While*** je zatvorena nakon zadnje linije kojom se kontrolišu LED diode, naredbom ***Wend***.

Kako se može videti sa električne šeme, taster je vezan na pin ***RB.0*** (interapt pin (*interrupt*)), takođe je i ***pull-up*** otpornik vezan na isti pin. Drugi kraj tastera je vezan na masu, tako da je početno stanje tastera logička 1. U trenutku pritiska na taster, kontroler će detektovati logičku nulu, što predstavlja promenu stanja tastera. Taster je vezan na pin B.0 iz razloga što je to interapt pin kod ovog mikrokontrolera. ***Interapt*** (eng. *interrupt* - prekid) predstavlja prekid u izvršavanju glavnog dela programa i prelazak u izvršavanje potprograma kojim je definisan interapt. Interapt je korišćen iz sledećeg razloga. Kako na početku rada simulacije glavna petlja (main) ulazi u sekvencu (deo) koda za kontrolisanje semafora, nije moguće detektovati pritisak tastera običnom naredbom (***If taster = 0 Then <nešto***>) pre završetka sekvence. Prema tome, potrebno je uvesti interapt, kako bi kontroler u svakom trenutku mogao da prekine svako izvršavanje i da pređe u deo koda određen interaptom.

Primer definisanja interapta je sledeći:

```
On_Hardware_Interrupt GoTo pesaci    ' Prilikom detektovanja interapta preci
u potprogram pesaci

INTCON = %10010000                   ' Definisanje interapta
```

Naredba **`On_Hardware_Interrupt GoTo <labela>`**, mikrokontroleru kaže da prilikom detektovanja hardverskog interapta, tj. spoljašnje promene stanja na tom pinu, pređe u potprogram kojim je opisan interapt (prekid). INTCON predstavlja rezervisanu reč, koja predstavlja definisanje Interrupt Control Registers – definisanje registara za interapt. Kako se može videti iz naredbe INTCON = %10010000, definisano je 8 bitova (po jedan bit za svaki registar). Omogućeni (*Enable*) su globalni bit za interapt i RB0 (pin B.0 – interapt pin) – spoljašnji interapt pin. Oslati bitovi su onemogućeni (*Disable*). Poslednja tri bita predstavljaju pokazivače – flegove (*Flag*), koji se prilikom detektovanja interapta setuju na 1. Prilikom ulaska u interapt rutinu – potprogram koji se izvršava aktiviranjem interapta, potrebno je setovati INTCON registar na početno stanje kako bi interapt ponovo bio omogućen.

REGISTER 4-3: INTCON – INTERRUPT CONTROL REGISTER (ADDRESS: 0Bh, 8Bh, 10Bh, 18Bh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF
bit 7							bit 0

bit 7 **GIE:** Global Interrupt Enable bit
1 = Enables all un-masked interrupts
0 = Disables all interrupts

bit 6 **PEIE**: Peripheral Interrupt Enable bit
1 = Enables all un-masked peripheral interrupts
0 = Disables all peripheral interrupts

bit 5 **T0IE**: TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 interrupt
0 = Disables the TMR0 interrupt

bit 4 **INTE**: RB0/INT External Interrupt Enable bit
1 = Enables the RB0/INT external interrupt
0 = Disables the RB0/INT external interrupt

bit 3 **RBIE**: RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt
0 = Disables the RB port change interrupt

bit 2 **T0IF**: TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

bit 1 **INTF**: RB0/INT External Interrupt Flag bit
1 = The RB0/INT external interrupt occurred (must be cleared in software)
0 = The RB0/INT external interrupt did not occur

bit 0 **RBIF**: RB Port Change Interrupt Flag bit
1 = When at least one of the RB<7:4> pins changes state (must be cleared in software)
0 = None of the RB<7:4> pins have changed state

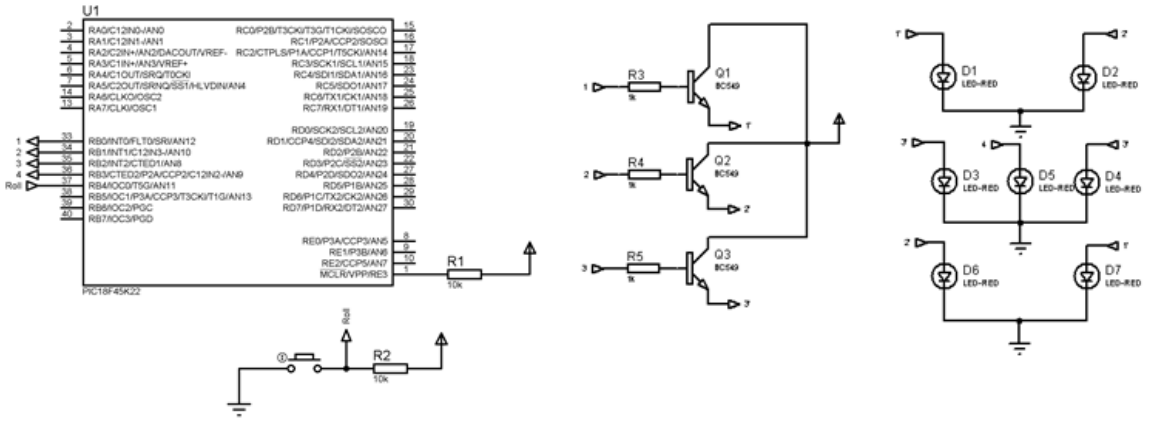
Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

Slika 49. INTCON registar

4.1d. LED elektronska kocka

Poslednji primer u prvom terminu biće primer realizacije elektronske kocke za igru kao što su „Ne ljuti se čoveče“, „Jamb“, itd. LED dioda koja predstavlja broj „***1***“ je vezana na pin ***RB.3***, ona se takođe koristi i pri prikazu broja „***3***“ i „***5***“. Za realizacijui broja „***2***“ koristi se pin ***RB.0***, na koji su vezane dve LED diode. Kako ***I/O*** pinovi na mikrokontroleru na izlazu daju ***20 mA***, dovoljno za pravilan rad jedne LED diode, neophodno je bilo pojačati struju korišćenjem ***NPN*** tranzistora ***BC549***. Na bazu je vezan otpornik za ograničavanje struje vrednosti 1 kΩ. Za realizaciju broja „***3***“ osim pina ***RB.3*** koristi se i pin ***RB.1***, na koji su kao i kod broja „***2***“ vezane dve LED diode preko tranzistora. Pri realizaciji broja „***4***“, koriste se pinovi ***RB.0*** i ***RB.1***. Za broj „***5***“ koriste se isti pinovi u kombinaciji sa pinom ***RB.3*** (broj „***1***“). Na kraju broj „***6***“ se realizuje korišćenjem pinova ***RB.0***, ***RB.1*** i ***RB.2***.

Električna šema LED elektronske kocke data je na slici 50.



Slika 50. LED elektronska kocka

Sa električne šeme, kao što je gore navedeno, vidi se da je za realizaciju električne kocke iskorišćen port ***B*** mikrokontrolera. Taster za „*bacanje*" kocke je vezan na pin ***RB.4***, moguće je taster realizovati i korišćenjem interapta kao u primeru semafora. Taster je realizovan u „*pull-up"* konfiguraciji sa pull-up otpornikom vrednosti ***10 kΩ***. Za određivanje broja koji će biti prikazan koristi se naredba „***Random***", čija sintaksa izgleda na sledeći način:

Promenljiva = ***Random***

Promenljiva uzima vrednost koju naredba „**nasumično**" odabere u opsegu vrednosti od ***0*** do ***65535***. Kako kocka ima 6 različita stanja – broja, neophodno je odrediti i koje vrednosti naredbe „***Random***" predstavljaju koji broj. Određivanje je izvedeno tako što je maksimalni broj koji naredba daje (65535) podeljen sa 6, kako bi se dobile šest podjednake vrednosti u opsegu od 0 do 65535. Na taj način su određeni opsezi za prikaz određenog broja:

Broj	Random
1	**0 - 10922**
2	**10923 – 21844**
3	**21845 – 32766**
4	**32767 - 43688**
5	**43689 - 54610**
6	**54611 - 65535**

Sekvenca koda kojim se određuju brojevi izgleda na sledeći način:

```
Main:

    While Roll = 1 : Wend        ' Dok taster nije pritisnut ne radi nista

        If A > 65535 Then        ' Ako vrednost promenljive A predje 65535
(word) postavi A na 0
            A = 0
        EndIf

        A = Random               ' Promenljiva A dobija vrednost naredbom
Random

    If Roll = 1 Then             ' Kada se otpusti taster pocinje
odredjivanje broja za prikaz

        For Broj = 0 To 30 Step 1

            If A > 65535 Then
                A = 0
            EndIf

            A = Random

            If A < 10922 Then                    ' Broj 1
                Kocka = 1
                High Br1
                Low Br2
                Low Br3
                Low Br4
            EndIf

            If A > 10923 And A < 21844 Then      ' Broj 2
                Kocka = 2
                Low Br1
                High Br2
                Low Br3
                Low Br4
            EndIf

            If A > 21845 And A < 32766 Then      ' Broj 3
                Kocka = 3
                High Br1
                High Br2
                Low Br3
                Low Br4
            EndIf

            If A > 32767 And A < 43688 Then      ' Broj 4
                Kocka = 4
                Low Br1
                High Br2
                Low Br3
                High Br4
            EndIf

            If A > 43689 And A < 54610 Then      ' Broj 5
                Kocka = 5
                High Br1
                High Br2
```

```
            Low Br3
            High Br4
        EndIf

        If A > 54611 Then                   ' Broj 6
            Kocka = 6
            Low Br1
            High Br2
            High Br3
            High Br4
        EndIf

        DelayMS Broj * 10                   ' Pauza izmedju dva prikaza
se povecava linearno 10 puta

    Next

    GoTo Prikazi                            ' Skakanje u potprogram za
prikaz broja

EndIf

GoTo Main
```

Deo koda kojim se određuje broj za prikaz se vrši u glavnoj petlji (***Main***), gde se nakon pritiska na taster za „bacanje“ kocke ulazi u ***For*** petlju. Sekvenca ove naredbe je data u prethodnom primeru („***LED matrica 7x11***“). Kroz ***For*** petlju se prolazi 10 puta gde promenljiva „***A***“ koja uzima vrednost naredbom *Random*, 10 puta menja vrednost. Promenljiva „*A*“ je tipa „***Word***“, kako bi se smestile vrednosti u opsegu od ***0*** do ***65535*** (tip ***Byte*** – ***0*** do ***255***) Prilikom svakog ponovnog prolaska kroz For petlju prikazuje se trenutna vrednost promenljive „***A***“, gde se svakim sledećim prolaskom vreme prikaza broja povećava 10 puta (***DelayMS*** *Broj * 10*). Na taj način je nemoguće predvideti broj koji će se na kraju dobiti.

Kada se izvrši For petlja do kraja, skače se do potprograma u kome se koristi naredba ***Select..Case..EndSelect***. Sekvenca ove naredbe je:

Select *Promenljiva*

Case Stanje1
Instrukcija za Stanje1

Case *Stanje2*
Instrkcija za Stanje2

Case Else
Uslov za Case Else

EndSelect

U ovom primeru ***Select..Case..EndSelect*** naredba je iskorišćena na sledeći način:

```
Prikazi:
```

```
            Select Kocka

                Case 1
                    Low Br1
                    Low Br2
                    Low Br3
                    High Br4

                Case 2
                    Low Br1
                    High Br2
                    Low Br3
                    Low Br4

                Case 3
                    Low Br1
                    High Br2
                    Low Br3
                    High Br4

                Case 4
                    High Br1
                    High Br2
                    Low Br3
                    Low Br4

                Case 5
                    High Br1
                    High Br2
                    Low Br3
                    High Br4

                Case 6
                    High Br1
                    High Br2
                    High Br3
                    Low Br4

            EndSelect

            If Roll = 0 Then
                GoTo Main
            EndIf
GoTo Prikazi
```

Vidi se da se u delu za ***Select*** navodi vrednost promenljive ***Kocka***, koja vrednost dobija prolaskom kroz For petlju u prethodnom delu koda. U drugom delu naredbe uz ***Case*** se navode moguća stanja za promenljivu ***Kocka*** (***1, 2, ..., 6***). Na kraju ovog dela koda stoji ispitivanje da li je pritisnut taster za ponovno „*bacanje*“. Ukoliko nije, broj koji je dobijen u prethodnom bacanju biće prikazivan sve do ponovnog „*bacanja*“. Kada je pritisnut taster za novo „*bacanje*“, izvršavanje koda se nastavlja iz glavne petlje (***Main***).

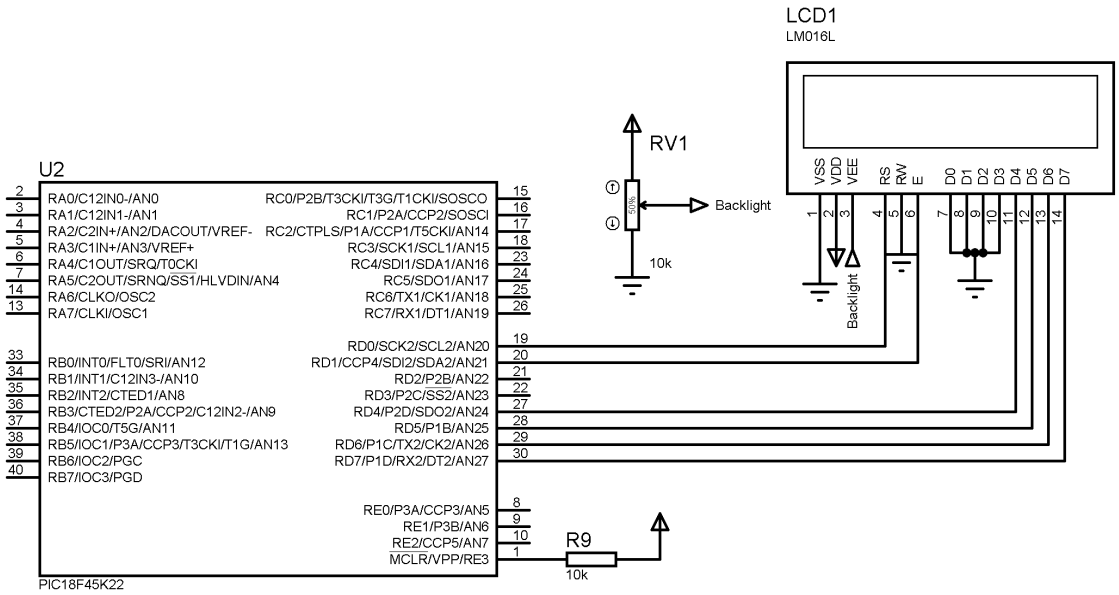
I vežba

4.1a. LCD dsiplej

Sledeća komponenta koja se vrlo često koristi u Optoelektronici je LCD displej. Često se na elektronskim uređajima mogu sresti razni LCD displeji sa mogućnošću prikaza određenog broja karaktera. Tako naprimer postoje displeji:

LCD displej	Prikaz karaktera
1x8	8 karaktera u jednom redu
2x8	8 karaktera u dva reda
1x16	16 karaktera u jednom redu
2x16	16 karaktera u dva reda
2x20	20 karaktera u dva reda
4x20	20 karaktera u četri reda

LCD displej koji će biti korišćen u ovom praktikumu je ***2x16*** tipa. Električna šema prve simulacije je data na slici 51.



Slika 51. Električna šema povezivanja LCD displeja sa mikrokontrolerom

Kao što se može videti, LCD displej komunicira sa mikrokontrolerom preko porta D. Takođe se vidi da je komunikaciju sa LCD displejom realizovana preko 4 linije, može

se realizovati i preko 8 linija (kada je za komunikaciju sa LCD-om potrebna veća brzina rada). Dakle, komunikacijski pinovi LCD displeja od ***D0 ÷ D3*** su vezani na masu (***GND***), dok su ostala četiri pina za komunikaciju vezani počevši od ***D4 ÷ D7*** na ***port D.4 ÷ port D.7***, respektivno. Pinovi kojima se reguliše rad LCD displeja, tačnije pin za izbor čipa (upis/čitanje) ***RS*** je vezan na pin ***port D.0***, pin ***EN*** kojim se omogućuje rad samog LCD displeja je vezan na pin ***port D.1***. Pin ***R/W*** (pošto LCD nije *touch screen* tipa) i nije moguć unos podataka preko LCD displeja je vezan na masu (***GND***). Pinovi V_{DD} i V_{SS}, su vezani na napajanje, odnosno na masu (GND). Pin V_{EE} koji služi za podešavanje pozadinskog osvetljenja je vezan na srednji pin potenciometra. Promenom otpornosti potenciometra će se vršiti podešavanje pozadinskog osvetljenja, bez koga sam LCD displej ne bi mogao da se korsiti, tj. ne bi bio moguć prikaz sadržaja na LCD displeju.

U Protonu postoji biblioteka koja sadrži naredbe koje se pozivaju u kodu kada korisnik želi da koristi LCD displej. Tačnije, tim naredbama se najpre definišu podešavanja samog LCD displeja, tj. linije za komunikaciju, ***RS*** i ***EN*** pinovi, tip LCD displeja i broj linija u kojima će se željeni sadržaj prikazivati. Nakon toga se u kodu iz iste biblioteke pozivaju naredbe koji će se štampati željeni sadržaj. Kod za kontrolu LCD displeja je sledeći:

```
Device = 18F45K22

Xtal 32
All_Digital = true

'==================================
'======== Deklaracija LCD-a ========
'==================================
    Declare LCD_Type     0            ' Tip LCD-a   (0)-Alfanumericki, (1)-
Graficki, (2)-Samsung/Toshiba
    Declare LCD_DTPin PORTD.4         ' Pocetni pin za podatke (D4-D7)
    Declare LCD_RSPin PORTD.0         ' Pin za izbor cipa (citanje/upis)
    Declare LCD_ENPin PORTD.1         ' Pin za omogucavanje ispisa na LCD
displeju
    Declare LCD_Interface   4         ' Broj linija za komunikaciju (8 ili
4)
    Declare LCD_Lines       2         ' Broj linija za ispis (1, 2 ili 4)
'==================================
'======== Deklaracija LCD-a ========
'==================================

Cls                                   ' Brisanje sadrzaja sa displeja

Main:

    Print At 1,1, "LCD displej 2x16" ' Ispis poruke u prvom redu od prve
pozicije
        DelayMS 1000
    Print At 2,1, "   karaktera.   " ' Ispis poruke u drugom redu od
cetvrte pozicije
        DelayMS 1000
            Cls
            DelayMS 250
GoTo Main
```

Kao što se vidi u priloženom kodu, nardbom **Declare LCD_Type** 0 se definiše tip LCD displeja koji se koristi. Proton kompajler nudi rad sa tri različita tipa LCD displeja, a to su:

- Alfanumerički, koji se u Protonu definiše kao **0**
- Grafički ili KS0108, koji se u Protonu definiše kao **1**
- Toshiba ili T6963 (Touch screen), koji se u Protonu definiše kao **2**.

Nakon toga su definisane linije za komunikaciju i kontrolne linije (*RS* i *EN*):

```
    Declare LCD_DTPin PORTD.4          ' Pocetni pin za podatke (D4-D7)
    Declare LCD_RSPin PORTD.0          ' Pin za izbor cipa (citanje/upis)
    Declare LCD_ENPin PORTD.1          ' Pin za omogucavanje ispisa na LCD
displeju
```

Na kraju je definisan broj linija za komunikaciju i broj linija za priakz sadržaja na LCD displeju:

```
    Declare LCD_Interface   4        ' Broj linija za komunikaciju (8 ili 4)
    Declare LCD_Lines       2        ' Broj linija za ispis (1, 2 ili 4)
```

Naredbom ***Print At x, y,*** se kaže mikrokontroleru da se sadržaj štampa u ***x***-toj liniji LCD-a i od ***y*** karaktera. Na taj način su i naredbe za štampanej sadržaja:

```
    Print At 1,1, "LCD displej 2x16" ' Ispis poruke u prvom redu od prve
pozicije
    Print At 2,1, "   karaktera.   " ' Ispis poruke u drugom redu od
cetvrte pozicije
```

4.1b. LCD dsiplej – Scroll Message

Vrlo često se može sresti popularna ***Scroll Message*** kod LCD displeja, tj. pokretna poruka, čiji je sadržaj duži od broja karaktera koji korišćeni LCD displej može u jednom trenutku da prikaže. Primer toga će biti obrađen u ovoj vežbi. Električna šema je ista kao u prethodnoj (slika 51.), tako da su podešavanja LCD displeja za rad ista kao i u prethodnom kodu. Nakon definisanja pinova za rad LCD-a, sledi definisanje promenljive koja sadrži poruku koju treba prikazati.

```
Dim Recenica As String *180
Dim Poruka As String *15
Dim Lenght As Byte
Dim i As Byte

Lenght = 0
i = 0

Recenica = "                Poruka koju zelimo da prikazemo na LCD displeju
2x16 karaktera "

Lenght = Len (Recenica)                ' Naredba Len odredjuje duzinu
Stringa i vrednost smesta u promenljivu tipa Byte
```

```
    Cls
    Print At 1,7, "2x16"                ' Ispis poruke u prvom redu od 7
pozicije
```

Promenljiva koja sadrži poruku je ***Recenica*** koja je tipa ***String***. Prilikom definisanja promenljive tipa *String*, neophodno je definisati koliko karaktera treba da sadrži ta promenljiva, što se definiše znakom * iza kog se navoi broj karaktera (npr. ***String *100***). Da bi sadržaj promenljive *Recenica* mogli da prikažemo kao *Scroll Message*, neophodno je „*razbiti*“ ili „*parsovati*“ istu u promenljivu takođe tipa String (`Dim Poruka As String *15`), ali sada sa maksimalno ***16*** karaktera koliko inače korišćeni LCD displej može da prikaže.

Kada je definisana promenljiva koja će se koristiti za priakzivanje poruke, neophodno je odrediti dužinu rečenicu koju parsujemo. Određivanje dužine stringa koji parsujemo se vrši naredbom ***Len***, čija sintaksa izgleda na sledeći način:

Promenljiva1 = ***Len*** (***Promenljiva2***)

Ovom naredbom se dužina *Promenljive2* (u našem slučaju promenljiva ***Recenica***) i vrednost dužine se smešta u promenljivu *Promenljiva1* (u našem slučaju je to ***Lenght*** (tipa ***Byte***)).

Nakon određivanja dužine poruke koju želimo da prikažemo, sledi i sam prikaz poruke:

```
Main:

    For i=0 To Lenght Step 1            ' Naredba For kojom se pravi petlja
kroz koju treba proci odredjeni broj puta

        Poruka[0] = Recenica[i]
        Poruka[1] = Recenica[i+1]
        Poruka[2] = Recenica[i+2]
        Poruka[3] = Recenica[i+3]
        Poruka[4] = Recenica[i+4]
        Poruka[5] = Recenica[i+5]
        Poruka[6] = Recenica[i+6]
        Poruka[7] = Recenica[i+7]
        Poruka[8] = Recenica[i+8]
        Poruka[9] = Recenica[i+9]
        Poruka[10] = Recenica[i+10]
        Poruka[11] = Recenica[i+11]
        Poruka[12] = Recenica[i+12]
        Poruka[13] = Recenica[i+13]
        Poruka[14] = Recenica[i+14]
        Poruka[15] = Recenica[i+15]

        Print At 2,1, Str Poruka
        DelayMS 250
    Next

GoTo Main
```

Kao što se vidi iz koda, prikaz poruke se izvršava prolaskom kroz For petlju, čije trajanje definiše dužina poruke koju smo prethodno odredili.

II vežba

4.2a. Sedmosegmentni displej

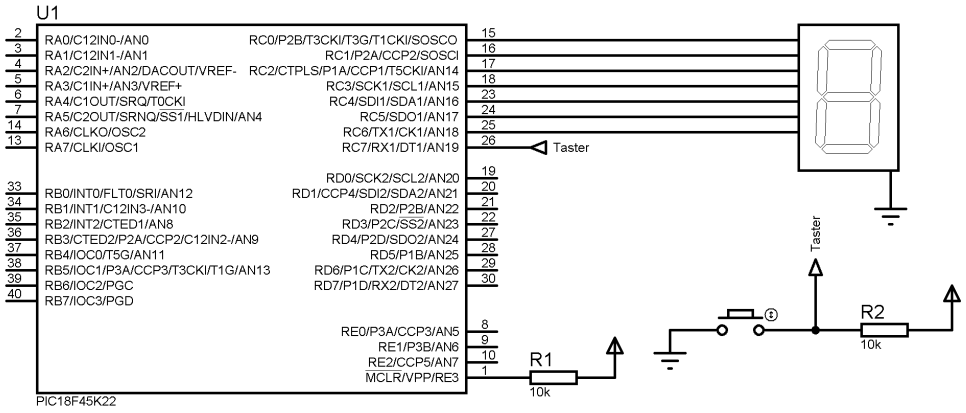
Optokomponenta koja se često sreće kod digitalnih časovnika, elektronskih uređaja u vidu displeja je ***sedmosegmentni displej***. Za razliku od LCD displeja iz prethodnog primera, sedmosegmentni displeji (slika 52.) se dele na:

- Sedmosegmentni displeji sa zajedničkom anodom i
- Sedmosegmentni displeji sa zajedničkom katodom.

Slika 52. Pinout sedmosegmentnih displeja sa zajedničkom anodom/katodom

Kod displeja sa zajedničkom anodom se dovodi masa (**GND**) kao signal gde treba svetleti koji segment, jer je napajanje (**V_{DD}**) već priključeno, dok se kod displeja sa zajedničkom katodom za segmente dovodi napon sa pomeračkih registara/mikrokontrolera, jer je masa (**GND**) već dovedena na zajednički kraj.

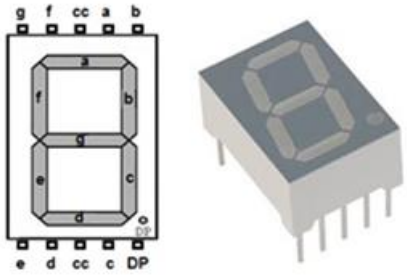
U ovom praktikumu biće dati primeri rada sedmosegmentnih displeja sa zajedničkom katodom. Najpre će biti dat primer kontrole sedmosegmentnog displeja mikrokontrolerom, dok će se u sledećem primeru kontrolisati dva sedmosegmentna displeja korišćenjem pomeračkih registara. Električna šema kontrole sedmosegmentnog displeja je data na slici 53.



Slika 53. Električna šema kontrole jednog sedmosegmentnog displeja sa zajedničkom katodom

Sa električne šeme se vidi da je sedmosegmentni displej vezan na ***port C*** mikrokontrolera (***RC.0 ÷ RC.6***). Taster kojim će se menjati prikaz (u ovom slučaju broj) je vezan na pin ***RC.7*** u pull-up konfiguraciji. Tablica istinitosti i pinout za korišćeni sedmosegmentni displej je data na slici 52.

Number	g f e d c b a	Hexadecimal
0	0 1 1 1 1 1 1	3F
1	0 0 0 0 1 1 0	06
2	1 0 1 1 0 1 1	5B
3	1 0 0 1 1 1 1	4F
4	1 1 0 0 1 1 0	66
5	1 1 0 1 1 0 1	6D
6	1 1 1 1 1 0 1	7D
7	0 0 0 0 1 1 1	07
8	1 1 1 1 1 1 1	7F
9	1 1 0 1 1 1 1	6F



Slika 54. Tablica istinitosti i pinout korišćenog sedmosegmentnog displeja

U kompajleru (*Proton IDE*) se predstavljanje brojeva na displeju može vršiti na više načina (obično tri načina), tj. najjednostavnije direktno pozivanjem pinova korišćenjem ***High/Low*** naredbi (koristiće se u ovom primeru), zatim korišćenjem naredbi ***TRIS*** i ***PORT*** gde se definiše direktno 8 (7) pinova određenog porta i na kraju korišćenjem ***LookUp*** tabele (koristiće se u sledećem primeru). Kao što se može videti sa slike gde je predstavljena tablica istinitosti, za predstavljanje broja se mogu pozivati pinovi direktno definisani u samom kodu, gde je svakom od pinova dodeljen naziv (npr. „***A***“, „***B***“,...) korišćenjem naredbe ***Symbol***. Takođe je moguće koristiti i heksadecimalno predtstavljanje brojeva zavisno od pinova koji se aktiviraju (slično kao prethodno predstavljanje), gde se heksadecimalni zapis odnosi na pinove koji su povezani sa sedmosegmentnim displejem. Primer dela koda kojim se vrši kontrola sedmosegmentnog displeja je sledeći:

```
Dim Broj As Byte
Broj = 0

Symbol Taster = PORTC.7          ' Definisanje pina za rad sa tasterom

'=================================================
'====== Definisanje pinova za 7 seg. displej ======
'=================================================
                                 '  === A ===
    Symbol A = PORTC.0           ' | |     | |
    Symbol B = PORTC.1           ' |F|     |B|
    Symbol C = PORTC.2           ' | |     | |
    Symbol D = PORTC.3           '  === G ===
    Symbol E = PORTC.4           ' | |     | |
    Symbol F = PORTC.5           ' |E|     |C|
    Symbol G = PORTC.6           ' | |     | |
                                 '  === D ===
'=================================================
'====== Definisanje pinova za 7 seg. displej ======
'=================================================
```

```
Main:

    If Taster = 0 Then
        Inc Broj
        DelayMS 500
        If Broj > 9 Then
            Broj = 0
        EndIf
    EndIf

    Select Broj

        Case 0
            High A
            High B
            High C
            High D
            High E
            High F
            Low G

        Case 1
            Low A
            High B
            High C
            Low D
            Low E
            Low F
            Low G

        Case 2
           ...
```

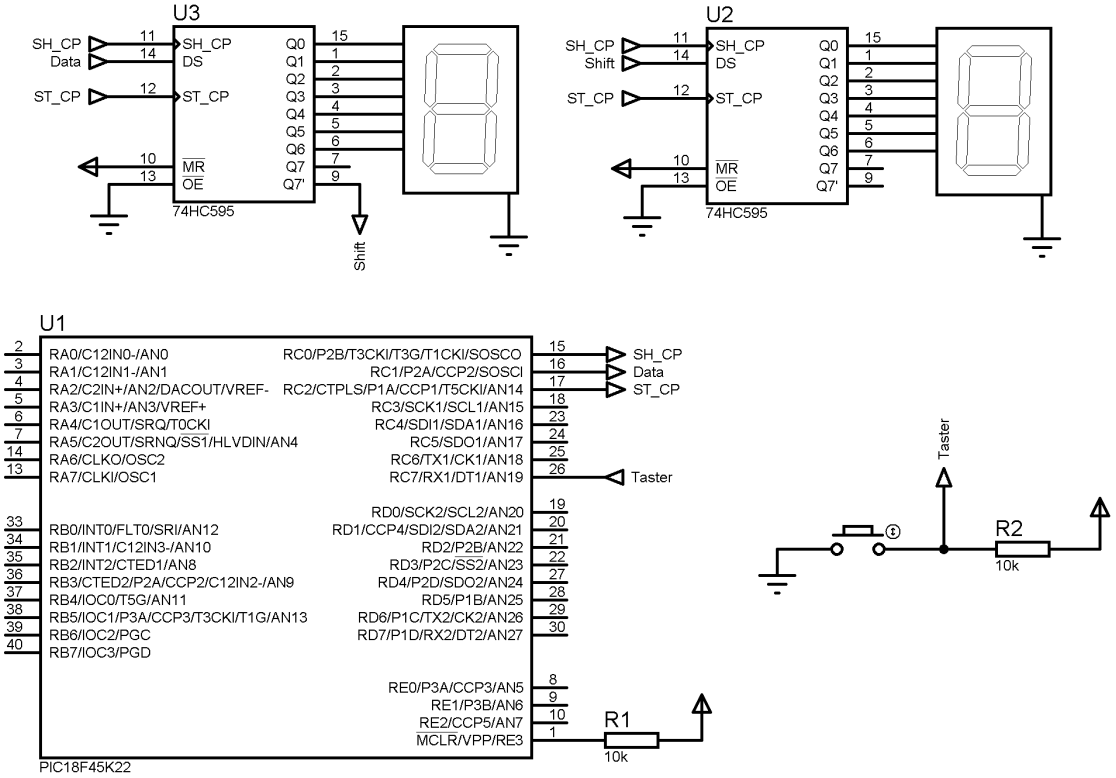
Naredba ***Select.. Case.. EndSelect*** je iskorišćena kako bi se prikazivao broj koji je određen tasterom. Kao što se vidi prikaz broja se vrši prethodno opisanim načinom – definisanjem stanja određenih pinova u stanju logičke jedinice, odnoso logičke nule (*High/Low*).

4.2b. Sedmosegmentni displej – realizacija brojača

U ovom primeru biće opisano kako se koristi pomerački registri za realizaciju brojača korišćenjem dva sedmosegmentna displeja. Realizovani brojač može da broji, tj. prikazuje brojeve od 0 do 99. Stanje na displeju, tj. broj se menja pritiskom na taster, kao u prethodnom primeru. Pomerački (Shift) registri korišćeni za realizaciju brojača su ***74HC595***, to je kao što je već u uvodnom delu objašnjeno serijsko/paraleno – serijski pomerački registar. To znači da je ulaz u pomerački registar serijskog tipa, tj. bitovi ulaze jedan po jedan i popunjavaju bafere (eng. *buffer*), dok je izlaz pomeračkog registra paralelno – serijski.

Paralelno - serijski izlaz znači da se podaci paralelno u istom trenutku „prikazuju“ - pojavljuju na izlazu, dok se serijski izlaz koristi da se podaci iz prethodnog stepena (pomerački registar) serijski (bit po bit) prenesu u sledeći stepen (pomerački registar), dok u prethodni stepen stižu novi podaci.

Električna šema realizacije brojača je data na slici 54.



Slika 54. Električna šema brojača

Iz mikrokontrolera se podaci vode na pin za podatke ***(DS)*** i to samo na prvi pomerački registar, a zatim se iz prvog serijski podaci prenose u ostale pomeračke registre. Takođe za rad pomeračkog registra postoje i dva radna takta (eng. *clock*), jedan je ***skladišni takt (ST_CP)***, a drugi ***pomerački takt (SH_CP)***. Oba takta se kontrolišu mikrokontrolerom. Skladišni takt služi za skladištenje podataka u pomerački registar, a to vrši tako da kada se podaci nađu na ulazu ***DS*** i skladišni takt je na logičkoj 1 podaci se skladište unutar registara za podatke. Drugi takt – pomerački takt služi za pomeranje podataka iz jednog pomeračkog registra u drugi. Slično prethodnom taktu, kada se pomerački takt aktivira na logički nivo 1, podaci smešteni unutar registara pomeračkog registra se prebacuju serijski (bit po bit) u sledeći pomerački registar.

Dakle aktiviranjem pomeračkog takta, podaci sa ***Q7'*** izlaza (serijski izlaz) idu na ***DS*** (ulaz za podatke (označen kao ***Shift*** na električnoj šemi)) sledećeg pomeračkog registra. Ovaj pomerački registar ima serijski ulaz, a serijski/paralelni izlaz. Paralelni izlaz pomeračkog registra služi za paljenje i gašenje segmenata displeja (LED dioda) koje su vezane na izlaze.

Taster kojim se menja broj koji se prikazuje je u *pull-up* konfiguraciji vezan na pin RC.7. Na početku koda treba definisati pinove za kontrolu pomeračkih registara, korišćenjem naredbe ***Symbol***.

Linije koda kojim se definišu pinovi za kontrolu pomeračkih registara su sledeće:

```
Symbol Podaci = PORTC.1          ' Definisanje pina za slanje podataka ka
pomerackom registru
Symbol SH_CP = PORTC.0           ' Definisanje pina za takt pomeranja (shift)
pomerackog registra
Symbol ST_CP = PORTC.2           ' Definisanje pina za takt za smestanje
podataka u pomeracki registar
```

Promenljiva u koju treba smestiti broj i čiji sadržaj treba prikazati se definiše kao tip ***Byte***. Iz razloga što se brojač realizuje da broji od 0 do 99, neophodno je prikazati dva broja na sedmosegmentnim displejima. Da bi mikrokontroler znao kada treba prikazivati i koje brojeve na displejima, broj se treba „razbiti“ na desetice i jedinice. Za razbijanje broja na desetice i jedinice neophodne su još dve promenljive u koju će biti smeštane jedinice broja, odnosno desetice broja. Linije koda kojima se razbija broj su sledeće:

```
    If Taster = 0 Then

        Inc Broj

                DelayMS 500

        If Broj > 99 Then
            Broj = 0
        EndIf
    EndIf

    Dummy = Broj & %11110000     ' BCD u binarni kod
    GoSub Racunanje_Desetica
    Broj = Broj & %00001111
    Broj = Broj + Dummy

    Dummy = Broj / 10            ' Binarni u BCD kod
    Dummy = Dummy * 16
    Broj = Broj // 10
    Broj = Broj | Dummy

    Brojl = Broj & $0F           'Rastavljanje BCD broja na jedinice i
desetice
    Brojh = Broj & $F0
    Brojh = Brojh >> 4


Racunanje_Desetica:
    Dummy=(Dummy/16)*10          ' Pomeranje vrednosti za 4 pozicije udesno
i mnozenje sa 10
    Return
```

Nakon dobijanja broja pritiskanjem tastera, vrednost promenljive treba pretvoriti iz BCD formata u binarni. Ovo konvertovanje se vrši kako bi se dobio ***viši*** i ***niži nibl*** (eng. *Nibble*, ***nibl*** je podatak od četiri bita), gde će se zasebno prikazivati niži nibl (jedinice) odnosno viši nibl (desetice). Nakon konvertovanja iz BCD formata u binarni i formiranja višeg i nižeg nibla, isti broj teba vratiti u BCD format, kako bi se viši i niži nibl sada formirali kao

jedinice i desetice jednog broja. Nakon toga sledi dodeljivanje vrednosti pomoćnim promenljivama za prikaz (Brojl i Brojh), koje predstavljaju jedinice i desetice broja koji se prikazuje.

Kada su promenljive spremne za prikaz, tj. kada je mikrokontroler spreman da pošalje vrednosti promenljivih na pomeračke registre, odakle će kasnije biti prikazane, treba odrediti koje pinove treba upaliti a koje gasiti na segmentima. Za određivanje segmenata koje pomerački registri aktiviraju koristi se deo koda gde se koristi naredba ***LookUp***.

```
        Index = Brojl
            GoSub Konvertovanje
        Index = Brojh
            GoSub Konvertovanje

        High ST_CP
        Low ST_CP


Konvertovanje:
    Ispis = LookUp Index, [$3F,$06,$5B,$4F,$66,$6D,$7D,$07,$7F,$6F]
        SHOut Podaci, SH_CP, MsbFirst, [Ispis]
    Return
```

Sintaksa naredbe ***LookUp*** ima sledeći izgled:

Promenljiva = **LookUp** *index* [*konstanta, konstanta,...*]

Iz linija koda se vidi da *index* uzima vrednost promenljive za jedinice ili desetice (***0 ÷ n-1*** (u našem slučaju do 9, ukupno 10 brojeva)) i na osnovu te vrednosti od 0 do 9 iz *LookUp tabele* uzima vrednost sa pozicije koja odgovara broju iz promenljive. Kada *index* uzme vrednost iz *LookUp tabele* sa odgovarajuće pozicije, tu vrednost smešta u promenljivu (*Ispis*), koja će se korišćenjem naredbe ***SHOut*** poslati na pomeračke registre, odakle se određeni segmenti aktiviraju.

Sintaksa naredbe ***SHOut*** je:

SHOut *Data_Pin, Clk_Pin,* [*OutputData, OutputData, ...*]

Iz sintakse se vidi da se nakon naredbe ***SHOut*** navodi pin preko kog se vode podaci (***Podaci***), zatim pin za takt pomerački registar (pomerački takt – ***SH_CP***) i u uglastoj zagradi se navodi/navode promenljiva(e) koja se šalje na pomerački registar. Asemblerska naredba ***MsbFirst***, kaže da se prvo šalje bit najvće težine ka pomeračkom registru, pa sve do bita najmanje težine.

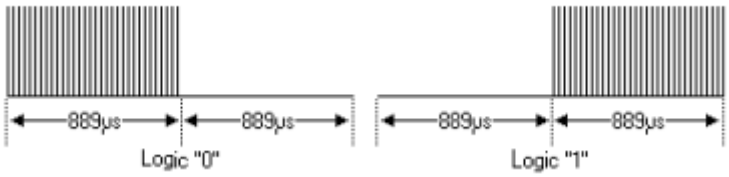
Nakon popunjavanja pomeračkih registara podacima koje treba prikazati, aktivira se takt za pamćenje podataka u pomeračke registre, nakon čega se čekaju novi podaci koje treba prikazati.

III vežba

4.3a. Prenos informacija optičkim vlaknom korišćenjem RC5 Philips protokola

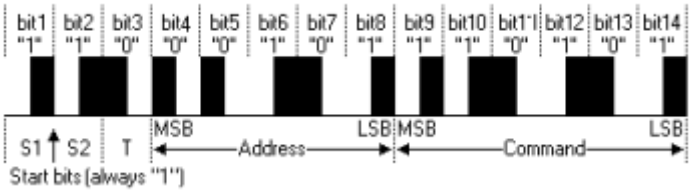
Jedna od trenutno najpopularnijih Optokomponenata je optičko vlakno. Optičko vlakno je trenutno najzastupljenije u prenosu informacija iz razloga što nema gubitaka prilikom prenosa. Primer primene optičkog vlakna u komunikacijama će biti dat i u ovom praktikumu, korišćenjem RC5 Philips protokola. Jedan od najpoznatijih protokola za rad daljinskih upravljača je Philips-ov protokol RC5. Philipsov RC5 (u Evropi i SAD) definiše noseću frekvenciju od 36 kHz, konstantnu dužinu bita od 1,778 ms i dužinu poruke od 14 bitova.

Za kodiranje signala se koristi dvofazni (Mančester) kod (slika 55.), gde se svaki bit predstavlja određenim prelazom između stanja. Logička jedinica je predstavljena prelazom iz niskog u visoko stanje, a logička nula prelazom iz visokog u nisko stanje. Visoko stanje je predstavljeno postojanjem signala noseće frekvencije, a nisko odsustvom ovog signala.



Slika 55. Dvofazni (Mančester) kod – logička nula i logička jedinica

Po ovom standardu, poruka (slika 56.) počinje sa dva start bita vrednosti logičke jedinice. Treći bit je „toggle“ bit, koji se invertuje svaki put kada se taster na daljinskom upravljaču otpusti i ponovo aktivira. Sledećih pet bita predstavljaju adresu uređaja, a poslednjih šest bita predstavljaju samu komandu.



Slika 56. Philips RC5 protokol – poruka

Ukoliko je taster na daljinskom upravljaču i dalje aktiviran, cela poruka se ponovo šalje nakon 114 ms. Pri tome, „toggle“ bit ostaje na istom logičkom nivou, tako da prijemnik može da protumači da je u pitanju samo jedna poruka, a ne dvostruko aktiviranje istog tastera. Tabela kodiranja tastera RC5 protokola u softveru je data na slici 57.

RC-5 Command	TV Command
$00 - 0	0
$01 - 1	1
$02 - 2	2
$03 - 3	3
$04 - 4	4
$05 - 5	5
$06 - 6	6
$07 - 7	7
$08 - 8	8
$09 - 9	9
$0A - 10	-/--
$0C - 12	Standby
$0D - 13	Mute
$10 - 16	Volume +
$11 - 17	Volume -
$12 - 18	Brightness +
$13 - 19	Brightness -
$20 - 32	Program +
$21 - 33	Program -
$32 - 50	
$34 - 52	
$35 - 53	
$36 - 54	
$37 - 55	

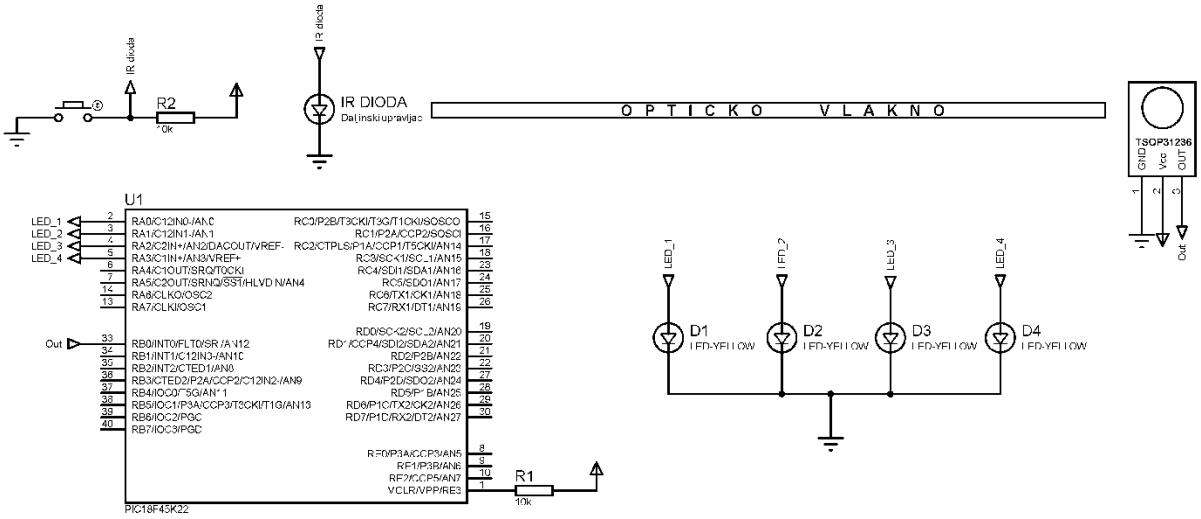
Slika 57. Tabela kodiranja tastera na daljinskom upravljaču

Rad daljinskog upravljača u ovom primeru biće dat uspomoć 4 LED diode, kako bi se upoznali sa samim RC5 protokolom. Rad daljinskog upravljača je simuliran preko 4 tastera sa daljinskog upravljača (slika 58.) - 1, 2, 3 i 4, gde je za svaki taster zadužena po jedna LED dioda.

Slika 58. Philips daljinski upravljač sa RC5 protokolom

Dakle izborom jednog od četiri tastera pali će se određena LED dioda, koja je u samom kodu zadužena za to. Dekodiranje infracrvenog signala sa daljinskog upravljača se vrši

uspomoć mikrokontrolera. Signal na mikrokontroler se dovodi sa infracrvenog senzora ***TSOP 31236***. Električna šema kola za detekciju signala sa daljinskog upravljača korišćenjem optičkog vlakna je data na slici 59.



Slika 59. Električna šema za detekciju signala sa daljinskog upravljača korišćenjem optičkog vlakna

Kao što se sa električne šeme može videti, infracrveni senzor ***TSOP 31236*** je vezan preko pina ***B.0*** (***PORTB.0***) na mikrokontroler. Ostala dva pina ovog senzora se koriste za napajanje (5 VDC, odnosno GND). LED diode koje se koriste za indikaciju pritisnutog tastera na daljinskom upravljaču su vezane na portu ***A*** (***RA.0 – RA.3***).

Kod uporabe optičkog vlakna potrebno je električni signal pretvoriti u svetlosni (***Infracrvena dioda*** sa daljinskog upravljača), pustiti svetlost u vlakno paralelno s uzdužnom osom, na suprotnoj strani potrebno je svetlosni signal pretvoriti u električni (***TSOP 31236***).

Prilikom aktiviranja kola za detekciju, najpre će se palitit jedna LED dioda za drugom, sa pauzama od 250 ms (kao inicijalizacija aktiviranja sistema).

```
'======== Inicijalizacija sistema ========
For I = 0 To 2 Step 1
    High PORTA.0 : DelayMS 500
    Low PORTA.0 : High PORTA.1 : DelayMS 500
    Low PORTA.1 : High PORTA.2 : DelayMS 500
    Low PORTA.2 : High PORTA.3 : DelayMS 500
    Low PORTA.3 : DelayMS 500
Next
'======== Inicijalizacija sistema ========
```

Nakon toga sledi inicijalizacija samog senzora, tj. deklarisanje pina na koji je vezan izlaz senzora. U ovom primeru kao što je već pomenuto, infracrveni senzor je vezan na pin B.0. Nakon deklaracije pina koji kontroliše stanje na izlazu senzora, vrši se definisanje promenljivih koje su potrebne za rad samog kola. Promenljive koje se koriste u ovom primeru su varijable – promenljive unapred definisanog tipa podataka koje tokom izvršavanja koda

menjaju svoju vrednost. Pošto u promenljive koje su potrebne za detekciju signala trebaju da sačuvaju vrednost od dva bajta, definisane su u tip podataka ***Word***.

Kako *RC5* protokol u svojom signalu šalje povorku od *14* impulsa, kao što je prethodno navedeno, treba reći mikrokontroleru koje bitove ove povorke da dekodira kao adresu uređaja, a koje kao komandu samog daljinskog upravljača. Iz tog razloga se i definišu dve varijable tipa ***Byte***, gde se jedna varijabla zadžena za adresu uređaja smešta u viši bajt registra, a vrednost zadate komande se smešta u niži bajt registra. Niz naredbi koje su potrebne za ovo su:

```
'================================================
'========== Deklaracija RC5 protokola ===========
'================================================
    Declare RC5In_Pin = PORTB.0            ' Izbor Port.Pin-a za infracrveni
senzor
    Dim RC5_Word As Word                   ' Kreiranje promenljive tipa Word
za prijem podataka

    Dim RC5_Command As RC5_Word.LowByte    ' Smeštanje Command bajta u
RC5_Word low byte

    Dim RC5_System As RC5_Word.HighByte    ' Smeštanje System bajta u
RC5_Word high byte
'================================================
'========== Deklaracija RC5 protokola ===========
'================================================
```

Nakon definisanja promenljih, sledi glavni deo koda, gde će se proveravati da li je došlo do promene stanja na ulaznom pinu (B.0). Prilikom prijema paketa informacija, mikrokontroler ispituje samo deo koda sa komandom, kako bi znao šta treba da uradi sledeće. Proton poseduje biblioteku za *RC5* protokol, tako da nije neophodno učitavati nove biblioteke. Ispitivanje koja komanda je poslata se vrši sledećim nizom naredbi:

```
Main:

    While 1=1

        Repeat
        RC5_Word = RC5In                ' Prijem signala sa Infracrvenog
senzora
        Until RC5_Command <> 255        ' Nastavi da proveravas dok ne
dobijes ispravan header kod
        If RC5_Command=1 Then Prvo      ' Taster "1" pritisnut na daljinskom
        If RC5_Command=2 Then Drugo     ' Taster "2" pritisnut na daljinskom
        If RC5_Command=3 Then Trece     ' Taster "3" pritisnut na daljinskom
        If RC5_Command=4 Then Cetvrto   ' Taster "4" pritisnut na daljinskom

    Wend
GoTo Main
```

Na početku glavne petlje se otvara petlja ***While*** (***While 1=1***) koja je stalno aktivna (beskonačna petlja). Unutar ove petlje se vrši ispitivanje da li je komanda koja je poslata sa daljinskog upravljača unutar niza komandi čije se vrednosti nalaze između ***0*** i ***255*** adrese naredbi. Ukoliko je van tog niza adresa, mikrokontroler će takvu poruku odbaciti. Ukoliko je

unutar tog niza, mikrokontroler upoređuje sa navedenim adresama koje su navedene u softveru. Ukoliko se primljena adresa poklopi sa postojećom adresom u kodu, mikrokontroler u skladu sa tim izvršava naredbu definisanu za određenu komandu.

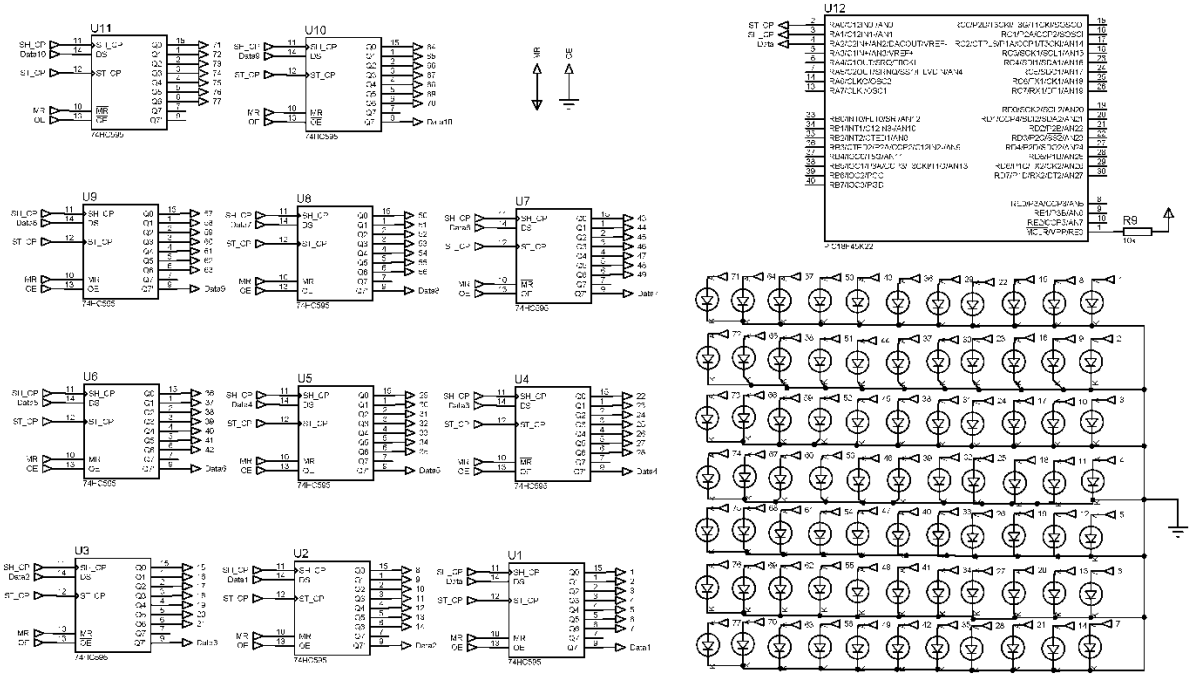
U skladu sa komandom koju dekodirao mikrokontroler izvršavaće se potprogram određen u samom kodu. Tako, pritiskom tastera 1 na daljinskom upravljaču, mikrokontroler će izvršiti potprogram „***Prvo***“, gde će upaliti LED diodu vezanu na pin A.0 i LED dioda će svetleti sve dok se ne zada druga komanda ili ponovi ista. Na sličan način izvršavaće se i ostali potprogrami, tj. paliće se LED diode, a gasiti prethodno aktivirane zavisno od zadane komande.

IV vežba

4.4a. LED matrica 7x11

U ovom primeru biće opisano kako se koristi pomerački registri za realizaciju matrice, odnosno matričnog displeja dimenzije 7x11 LED dioda. Prikaz teksta je realizovan tako da se kontroliše svaka LED dioda posebno. Brzina ispisivanja teksta se može podešavati u kodu. Teksat koji se ispisuje na matrici je „Микроелектроника“, „Elektronski fakultet u Nišu 2015.“

Za LED matricu je korišćeno 77 LED dioda, koje su vezivane preko anode na izlaze pomeračkih registara, dok su katode na masi (GND). Po 7 LED dioda je vezano na jedan pomerački registar (pomerački registar ima 8 izlaza, tako da je jedan slobodan). Električna šema LED matrice, data je na slici 60.



Slika 60. Električna šema LED matrice

Kako se može videti sa električne šeme, za rad matrice potrebno je 3 pina, jedan za podatke ***DS*** (A.2), takt za skladištenje podataka u pomerački registar ***ST_CP*** (A.0), takt za

pomeranje podataka u sledeći registar ***SH_CP*** (A.1). Podaci se vode iz mikrokontrolera na prvi pokmerački registar, odnosno pomerački registar koji kontroliše niz LED dioda koje su krajnje desno, odakle počinje ispisivanje teksta. Nakon ispisivanja jedne linije teksta, mikrokontroler podiže pin ***ST_CP*** (A.0) i tada prvi pomerački registar prebacuje podatke iz svog skladišnog registra u drugi pomerački registar. Zatim podiže logički nivo na ***DS*** (A.2) i ***SH_CP*** (A.1), kada se novi podaci šalju i smeštaju u prvi pomerački registar. Istim taktom se smeštaju podaci u drugom pomeračkom registru, koji su prethodno serijski premešteni iz prvog u drugi pomerački registar.

Pin A.0 koji služi za smeštanje podataka u registre se aktivira pomoću naredbe **High/Low** i to unutar potprograma u kome se nalazi tekst koji se ispisuje. Ispisivanje teksta se vrši na sledeći način. Unutar potprograma koji služi za ispisivanje teksta, poziva se For petlja, koja prati ispisivanje svakog karaktera teksta. Sintaksa For petlje je:

```
For index=xxx To xxx Step x
```

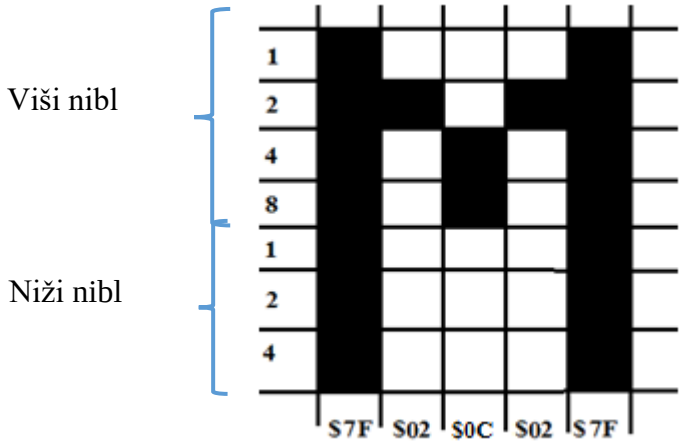
Gde ***For*** petlja prati pozivanje karaktera od neke početne navedene vrednosti za promenljivu (u ovom slučaju „***index***“) do neke navedene krajnje vrednosti, sa određenim korakom. Taj korak može biti 1, 2, 10, ..., ali takođe i sa negativnim predznakom -1, -2, -10, ... U slučaju da je korak sa negativnim predznakom, promenljiva za koju se prati pozivanje mora za početnu vrednost imati veći broj od one konačne vrednost ka kojoj se ide, jer korak ide unazad, tj. smanjuje vrednost početnu vrednost za određeni broj unapred definisan prolaskom kroz svaku novu iteraciju.

Nakon otvaranja ***For*** petlje i njenog definisanja, sledi definisanje karaktera za svako slovo/broj koji je u tekstu:

```
Promenljiva = LookUp index, [$00, $11, $22,...]
```

Promenljiva je naziv promenljie koja uzima vrednost iz niza karaktera u uglastoj zagradi. Za to se koristi naredba ***LookUp*** (kao u primeru sa brojačem) koja na osnovu promenljive “***index***” iz ***For*** petlje uzima određeni karakter za tu vrednost promenljive ***index*** i to šalje u promenljivu ***Promenljiva***. Na taj način se serisjki šalje niz za nizom karaktera koji čine tekst, jer za vrednost promenljive ***index = 100***, naredbom ***LookUp***, promenljivoj ***Promenljiva*** se dodeljuje kodirani karakter u *heksadecimalnom* formatu bas na toj poziciji u nizu u uglastoj zagradi.

Predstavljanje karaktera se vrši pomoću heksadecimalnog koda, kako bi se smanjio broj jedinica i nula, u slučaju kodiranja karaktera binarnim kodom. Svaki karakter se sastoji od 8 bita, koji se deli na ***viši*** i ***niži*** nibl (eng. *nibble*). Kako karakter ima 8 bitova i podeljen je na viši i niži nibl, bit najveće težine kod nižeg nibla se ne kodira, jer kao što je već navedeno u nizu ima 7 bitova – LED dioda. Način kodiranja se vrši na sledeći način (slika 61.):



Slika 61. Predstavljanje slova „***M***“ u matrici

Prilikom kodiranja slova/brojeva najpre se kodira niži nibl, tj sabiraju se bitovi, počev od bita najmanje težine ka bitu najveće težine (u slučaju nižeg nibla je to $2^2 = 4$), tako da u prvo slučaju imamo: $2^0 + 2^1 + 2^2 = 7$, tj. $(1 + 2 + 4 = 7)$. Zatim sledi kodiranje višeg nibla istom logikom kao i kod nižeg nibla (ovde je nibl kompletan, tj. svih 4 bita se koriste), pa ovde imamo: $2^0 + 2^1 + 2^2 + 2^3 = F$ (15), tj. $(1 + 2 + 4 + 8 = F$ (15)). Kao što je poznato u heksadecimalnom kodu se brojevi prikazuju u formatu: ***0, 1, 2, 3, 4, 5, 6, 7, 8, 9***, a nakon broja 9 se vrednosti ***10, 11,*** pa do ***15*** predstavljaju slovima ***A, B, ..., F,*** tj. imamo ***0, 1, ..., 8, 9,*** A^{10}, B^{11}, C^{12}, D^{13}, E^{14}, F^{15}.

Nakon kodiranja slova/brojeva sledi određivanje brzine ispisivanja teksta korišćenjem naredbe ***DelayMS*** (vreme u milisekundama). Ova naredba nakon određenog vremena kazuje mikrokontroleru da treba da okida takt skladišnog registra i da šalje sledeći heksadecimalni karakter, tj. binarni niz (ukoliko se kodira u binarnom brojnom sistemu). Nakon toga aktiviranje pina za podatke i za pomerački takt, pa koji bit se prvi ispisuje, da li bit najmanje težine ili najveće težine i odakle se uzimaju vrednosti. Sve to izgleda na sledeći način (primer ispisivanja dela teksta „***Elektronski fakultet u Nišu***“ – kodirani deo slova „***El***“...):

```
For Index=0 To 166 Step 1

     Promenljiva= LookUp Index, [ $7F,$49,$49,$41,$41,$00, $7E,…

   DelayMS 100                                  ' Trajanje niza LED dioda koje
svetle

   High PORTA.0                                  ' Logicka 1 za upis podataka u
pom. reg.
   Low PORTA.0                                   ' Logicka 0 za skladistenje
podataka u pom. reg.

   SHOut dta, clk, MsbFirst, [Promenljiva]     ' Ispisivanje karaktera

   Next                                          ' Uzimanje sledece vrednosti
za "index"
```

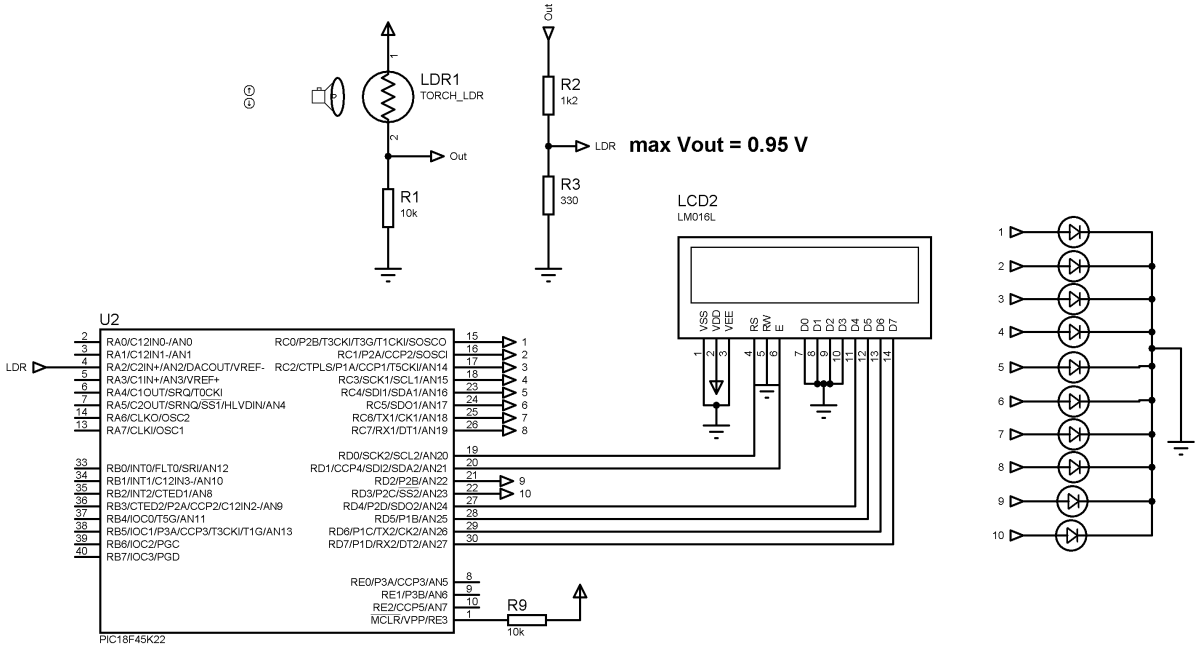
Pomoću naredbe ***SHOut*** (ShootOut) se aktiviraju pin za podatke i za pomerački takt, zatim ***MsbFirst*** označava da se prvo ispisuje bit najveće težine i na kraju odakle se uzimaju ppodaci koji se ispisuju – uzima se svaki put trenutna vrednost promenljive ***Promenljiva***. Nakon toga sledi naredba ***Next***, kojom se prelazi na sledeću vrednost promenljive ***index***, a ujedno se naredbom ***Next*** zatvara ***For*** petlja.

Kada se ispiše i poslednji karakter za dati niz karaktera, naredbom ***GoTo*** glavna petlja (***Main***) kompajler se vraća u glavni deo koda gde prati šta sledeće treba da ispiše i to na osnovu promenljive sličnoj onoj za prelaz iz moda u mod rada. Princip ispisivanja je potpuno isti.

V vežba

4.5a. Određivanje intenziteta osvetljenosti prostorije fotootpornikom

Naredna optokomponenta koja se koristi u Optoelektronici je i *fotootpornik* (eng. ***LDR*** – ***L****ight* ***D****ependent* ***R****esistor*). Fotootpornik ima široku primenu, pa se tako može primenjivati za realizaciju senzora za paljenje uličnih svetiljki (kada je nivo osvetljenosti ispod određene vrednosti), za realizaciju kolorimetra (određivanja intenziteta boje – često kod robota za izbegavanje prepreka), itd. U ovom praktikumu fotootpornik će se koristiti za određivanje intenziteta osvetljenosti prostorije i prikaz intenziteta iste koristeći LCD displej i LED dioda u vidu VU metra. Električna šema simulacije je data na slici 62.



Slika 62. Električna simulacija korišćenja fotootpornika za određivanje intenziteta osvetljenosti prostorije

Sa električne šeme se može videti da je jedan kraj fotootpornika (*pull-up* konfiguracija) vezan preko razdelnika napona na treći pin ***Analogno/Digitalnog*** (***A/D***) konvertora – pin ***A.2*** mikrokontrolera. Razdelnik napona se koristi kako bi se ulazni napon u A/D konvertor ograničio na maksimalnih ***1 V***. Ulazni napon je ograničen iz razloga što je referentni napon s kojim se upoređuje ulazni napon podešen na ***1.024 V***. Pošto je maksimalni ulazni napon sa fotootpornika (kada je u totalnom mraku) ***4.42 V***, a A/D konvertor mikrokontrolera može da izdrži ***maksimalni*** napon od ***5 V*** na svom ulazu, bilo je neophodno realizovati razdelnik napona. Iz razloga što napon napajanja može da varira, pa nije uvek stabilnihh 5 V, za najpreciznija merenja se izbegava da se za referentni napon uzima napon napajanja. Određivanje vrednosti parametara naponskog razdelnika se vrši na sledeći način:

Polazi se od početnog izraza za naponski razdelnik:

$V_{out} = \frac{R_2}{R_1 + R_2} \cdot V_{in}$, odakle je poznato V_{in} i V_{out} (V_{in} = 4.42 V, $V_{out} \approx 1$ V (vrednost napona koja nam treba za A/D konvertor)), za otpornik R_1 je uzeta vrednost od 1.2 kΩ. Na osnovu toga, neophodno je odrediti vrednost otpornika R_2, na sledeći način:

$$R_2 = \frac{R_1 \cdot V_{out}}{V_{in} - V_{out}} = \frac{1.2 \cdot 10^3 \cdot 1}{4.42 - 1} = \frac{1200}{3.42} = 350{,}87\ \Omega \Rightarrow 330\ \Omega$$

Za definisanje A/D konvertora u Proton Basic-u neophodno je definisati registre zadužene za A/D konvertor i registar za referentni napon. Linije koda kojima se vrši definisanje registara A/D konvertora su sledeće:

```
'====================================
'======== Def. AD konvertora ========
'====================================
    VREFCON0=%10010000          ' Postavljanje referentnog napona na 1.024V
    ANSELD=%00000000            ' Definisanje Porta D kao digitalni
    ANSELA=%00000100            ' Definisanje Porta A kao digitalni, osim A.2
    ANSELB=%00000000            ' Definisanje Porta B kao digitalni
    ANSELC=%00000000            ' Definisanje Porta C kao digitalni
    TRISA=%00000100             ' Definisanje smera porta A -> ulazni A.2
    ADCON0=%00001001            ' Treci analogni ulaz -> AN2
    ADCON1=%00001000
    ADCON2=%10111001
'====================================
'======== Def. AD konvertora ========
'====================================
```

U prvoj liniji koda gde se vrši definisanje A/D konvertora, definisan je ***VREFCON0*** registar kojim je definisan referentni napon u odnosu na koji će se vršiti A/D konverzija. ***ANSEL*(*X*)** registar služi za definisanje tipa pinova određenog porta, tj. da li je taj port analognog ili digitalnog tipa. ***ADCON0***, ***ADCON1*** i ***ADCON2*** su registri kojima se definiše rad samog A/D konvertora. Mikrokontroler ***PIC18F45K22*** ima 30 A/D ulaza, pa se registrom ***ADCON0*** se definiše koji od kanala A/D konvertora koristi (u ovom primeru je to treći ulaz (kanal) – ***AN2***). Istim registrom se definiše i stanje A/D konverzije, tj. da li je A/D konverzija u toku ili ne, a takođe i da li je omogućena A/D konverzija ili ne. ***ADCON1*** registar se koristi kako bi se definisalo korišćenje referentnog napona, tj. šta mikrokontroler treba tokom A/D konverzije koristiti kao referentni napon. Na kraju ***ADCON2*** registar služi za definisanje najpre sa koje strane se vrši poravnanje konvertovanog signala, tj. kako treba smestiti konvertovanu vrednost (***MSBFirst*** ili ***LSBFirst***). Zatim se sa naredna tri bita definiše brzina akvizicije podataka, što je veoma važno kada se treba vršiti precizna i brza A/D konverzija (naravno zavisi i od frekvencije oscilatora koji se koristi). Poslednja tri bita ovog registra definišu takt A/D konverzije koja zavisi od frekvencije oscilatora koji se koristi za rad samog mikrokontrolera.

Nakon definisanja A/D konvertora treba definisati i promenljive u koje će se smeštati rezultati A/D konverzije. Kako korišćeni mikrokontroler poseduje ***10-bitni*** A/D

konvertor, tip promenljive mora biti ***Word*** (***0 ÷ 65535***), jer se naponski nivoi A/D konverzije kreće u granicama od 0 do 1023 naponska nivoa, što prelazi kapacitet promenljive tipa ***Byte*** (***0 ÷ 255***).

Linije koda kojima se vrši A/D konverzija signala sa fotootpornika (LDR) su sledeće:

```
Main:

      LDR = 0
      TRISA.2 = 1
      ADCON0.0 = 1                          ' Aktiviranje A/D konvertora

      DelayUS  50                           ' Brzina semplovanja (ocitavanja)
      ADCON0.1 = 1                          ' Pokretanje A/D konverzije
        While ADCON0.1 = 1                  ' Dok je aktivna A/D konverzija
        Wend

'=======================================
'=== Uzimanje vrednosti A/D konverzije ===
'=======================================
      LDR.HighByte = ADRESH             ' Smestanje vrednosti A/D konverzije
u High byte
      LDR.LowByte = ADRESL              ' Smestanje vrednosti A/D konverzije
u Low byte
'=======================================
'=== Uzimanje vrednosti A/D konverzije ===
'=======================================

      ADCON0.0 = 0                          ' Prekidanje A/D konverzije

      LDR_napon = LDR * 5 / 1023
```

Za početak A/D konverzije treba setovati bit ADCON0.0 na logičku 1, čime se aktivira (omogućuje A/D konverzija (Enable)). Nakon toga setuje se i bit ***ADCON0.1*** (***GO/DONE***) na stanje logičke ***1***, tu vrednost će bit zadržati sve dok traje A/D konverzija, nakon čega sledi upis rezultata konverzije u promenljivu koja je za to definisana.

Kako se iz koda može videti, promenljiva u koju se smešta rezultat A/D konverzije je ***LDR*** (***Word***). Rezultat A/D konverzije se smešta na sledeći način – ***ADRESH*** (viša adresa A/D konvertora) se smešta u viši bajt promenljive ***LDR*** (***LDR.HighByte***), dok se ***ADRESL*** (niža adresa A/D konvertora) se smešta u viši bajt promenljive ***LDR*** (***LDR.LowByte***). Razlog ovoga je što se nekad za rad koristi samo viši ili samo niži bajt promenljive, takođe je i mnogo sigurnije za rad.

Nakon upisa rezultata u promenljivu LDR, bit ***ADCON0.0*** se setuje na logičku ***0***, čime se završava A/D konverzija.

Rezultat konverzije je smeštena u promenljivu LDR koja sadrži vrednost naponskih nivoa od 0 do 1023. Ukoliko te naponske nivoe treba prikazati u vidu napona koji je ušao u A/D konvertor, treba naponske nivoe iz LDR promenljive konvertovati i smestiti u promenljivu (***LDR_napon***) koja je tipa Float:

```
LDR_napon = LDR * 5 / 1023
```

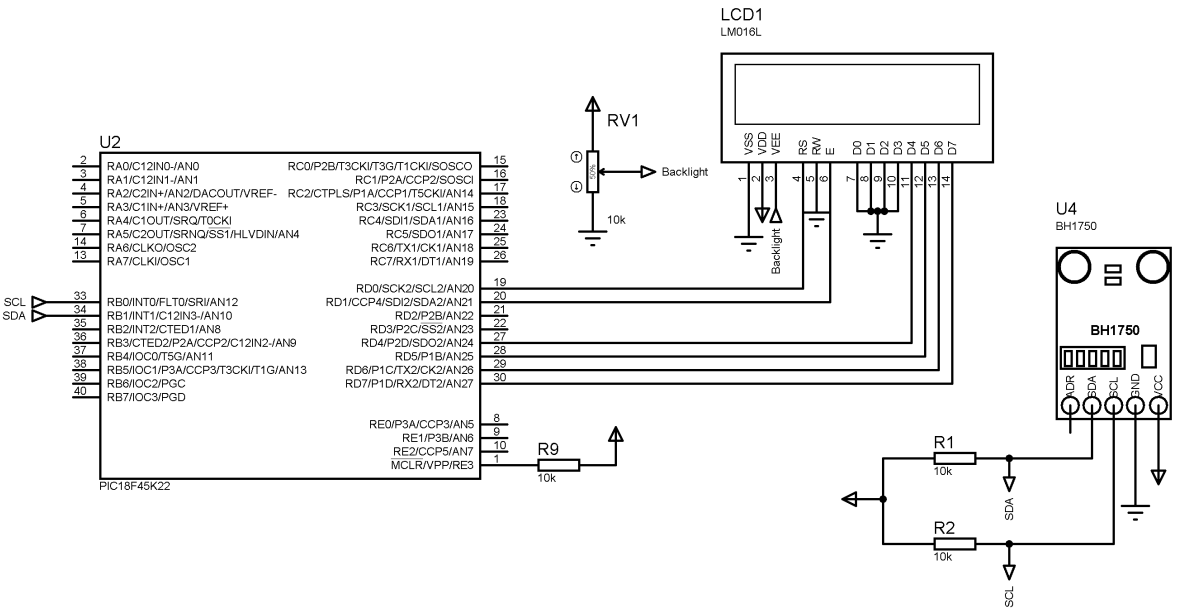
Promenljivu LDR treba pomnožiti sa 5, iz razloga koji je prethodno naveden da je napon koji ulazi u A/D konvertor spušten naponskim razdelnikom do vrednosti od *1.024 V*, a maksimalna vrednost ulaznog napona je 5 V. Tu vrednost treba podeliti sa 1023 jer je maksimalna vrednost naponskih nivoa ***1023***. Vrednost LDR_napon štampamo na LCD displeju koji se mikrokontrolerom povezan kao i u prethodmom primeru (kontrola LCD displeja).

Nakon prikaza vrednosti napona koji se dobija sa fotootpornika, treba rezultat A/D konverzije predstaviti i u vidu VU metra, korišćenjem LED dioda. LED diode koje su vezane na portu C (***C.0 ÷ C.7***) i portu D (***D.2*** i ***D.3***) se pale i gase zavisno od naponskog nivoa smeštenog u promenljivu LDR.

4.5b. Merenje intenziteta osvetljenosti senzorom vidljive svetlosti BH1750 - luxmetar

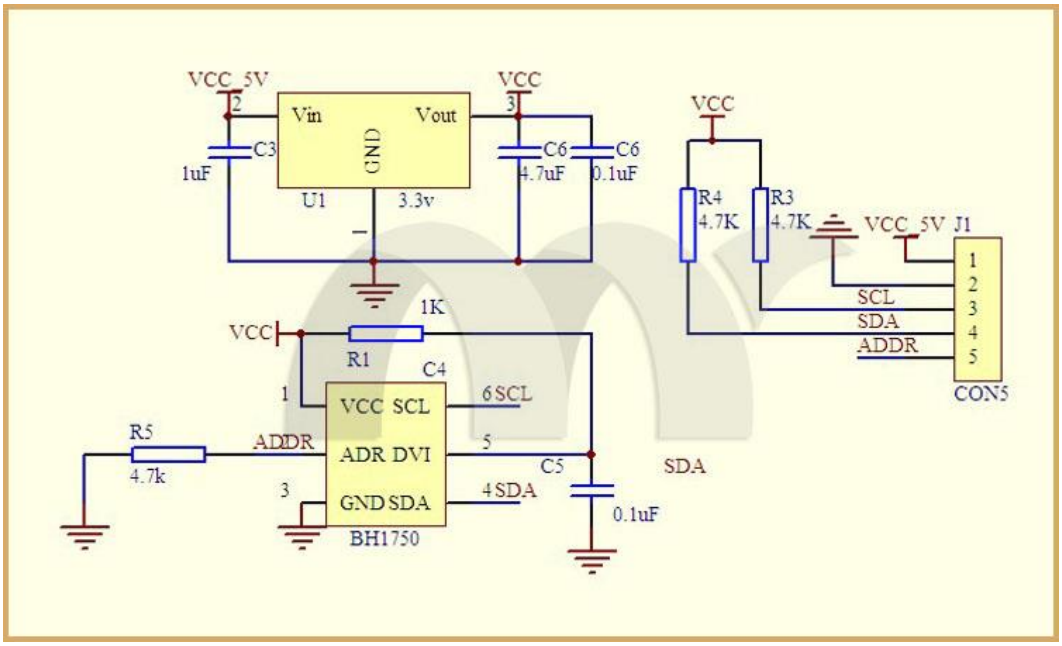
Poslednja optokomponenta koja će se koristiti u ovom praktikumu je senzor vidljive svetlosti ***BH1750*** (*ROHM Semiconductor*). Veoma često se sreće potreba za luxmetrom, često za kalibraciju raznih uređaja, čiji rad zavisi od uticaja svetlosti. Takođe, luxmetar se koristi za određivanje karakteristika optokomponenata, kao što su LED diode, fotootpornici. Kod fotootpornika se zavisno od intenziteta osvetljenja menja i otpornost samog fotootpornika, pa se na taj način određuje izlazna karakteristika fotootpornika. Na osnovu izlazne karakteristike, tj zavisnosti otpornosti od intenziteta svetlosti se određuju parametri fotootpornika (struja tame, fotostruja, otpornost fotootpornika,...).

Senzor BH1750 će biti iskorišćen za realizaciju luxmetra, s obzirom da daje rezultate merenja u luxima (***0 – 65535 lux***). Komunikacija sa mikrokontrolerom se vrši preko ***I2C*** magistrale (***SDA*** i ***SCL***). Električna šema luxmetra je data na slici 63.



Slika 63. Električna šema luxmetra

BH1750 senzor vidljive svetlosti u sebi osim samog senzora sadži i ***16-bitni A/D konvertor***. Nakon detekcije i merenja intenziteta svetlosti, A/D konvertor na samom senzoru na izlazu (preko ***I2C*** magistrale) daje vrednosti od ***0*** do ***65535***, što predstavlja kalibrisanu vrednost merenja. To znači da naknadna konverzija nije neophodna, tj. rezultati merenja ne zahtevaju matematičku obradu. Električna šema senzora je data na slici 64.



Slika 64. Električna šema senzora BH1750

I2C ("***I***nter-***I***ntegrated ***C***ircuit") magistrala, omogućuje dobru podršku za komunikaciju sa različitim sporijim periferijskim jedinicama u sistemima gde se potreba za njihovom upotrebom javlja povremeno. Razvila ga je evropska firma ***Philips*** a zatim su ga i drugi proizvođači integrisanih kola podržali, a dana se vrlo često sreće naručito kod mikrokontrolerskih sistema.

Glavne osobine ovog standarda su sledeće:

1. Za komunikacija su potrebne dve linije
2. Brzina prenosa do ***100 kbit***-a i više
3. Mogućnost adresiranja do ***128*** uređaja u sistemu
4. Relativno prosta podrška u programiranju "*Master*" uređaja
5. Vreme predaje je vremenski neovisno

Fizički ***I2C*** magistrala (eng. *bas*) sadrži samo dve linije signala, a to su ***SLC*** (***Serial CLock***) koja se koristi za prenos taktnog signala i linija ***SDA*** (***Serial DAta***) za prenos podataka. Za upravljanje ovim linijama najčešće se koriste kola sa otvorenim kolektorom pa se ove linije napaju sa naponom od +5V preko otpornika reda od ***1 - 10 kΩ***,

zavisno od dužine linije i brzine prenosa. U standardnom režimu rada dužina ovih linija je do ***2 m*** uz brzine prenosa do ***100 kbit***-a.

Kao što se vidi sa električne šeme, I2C magistrala je realizovana na pinovima porta B - ***B.0*** i ***B.1*** (***SCL*** i ***SDA***, respektivno). Mikrokontroler *PIC18F45K22* poseduje dva U(S)ART modula u sebi kojima se bez softverskog definisanja mogu koristiti I2C ili SPI magistrale. Takođe, moguće je i softverski realizovati I2C (SPI takođe) magistrale na drugim pinovima, što je i primer i ovde.

U programskom jeziku Proton Basic definisanje I2C magistrale se definiše na sledeći način:

```
Declare SDA_Pin PORTB.1                ' Deklarisanje pina za podatke  -> I2C
Declare SCL_Pin PORTB.0                ' Deklarisanje pina za takt     -> I2C
```

Prethodno je navedeno da BH1750 poseduje 16-bitni A/D konveror, tako da je maksimalna vrednost koja se dobija tokom merenja 65535. Iz tog razloga promenljiva u koju će se smeštati vrednost merenja senzora mora imati tip podatka ***Word*** (**0-65535**). Nakon definisanja promenljive za smeštanje rezultata merenja, neophodno je izvršiti inicijalizovaciju I2C magistralu i time pokrenuti merenje senzora. Za incijilaziciju I2C magistrale je moguće koristiti sledeće naredbe:

BusOut / BusIn ili ***I2COut / I2CIn***

Obe naredbe imaju istu namenu, s razlikom što se kod naredbe *I2COut / I2CIn* moraju navesti i pinovi kojima je realizovana I2C magistrala, dok kod naredbe *BusOut / BusIn* to nije slučaj. U oba slučaja je neophodno navesti kontrolni bajt, tj. adresu uređaja (eng. "*slave*"), koji se nalazi u tehničkoj dokumentaciji (eng. *Data Sheet*) svake komponente. Adresa senzora BH1750 je ***%00100000***. Inicijalizacija I2C magistrale za rad sa BH1750 senzorom u Protonu se vrši na sledeći način:

```
   BusOut %01000110, [%00100000]              ' Obracanje Slave-u i njegova
inicijalizacija
   DelayMS 130
   BusIn %01000111, $F8, [A.Byte1, A.Byte0]   ' Trazenje odgovora sa senzora
i upis u promenljivu
```

Iz koda se može videti da se bajt za inicijalizaciju I2C magistrale (%0100011<u>0(1)</u>) razlikuje u poslednjem bitu. Razlog toga je da li se vrši upis na I2C magistralu (ka slave-u) ili se vrši čitanje iz slave-a, tj. ***0*** znači upis, a ***1*** čitanje (konsultovati *Data Sheet*).

Nakon inicijalizacije I2C magistrale, kada se primaju rezultati merenja, te podatke treba smestiti, tj. sačuvati u promenljivu. Rezultat merenja se smešta u prethodno definisanu promenljivu (u ovom slučaju ***A*** (***Word***)), koja se deli na viši i niži bajt. U prethodnom primeru smo imali gde smo promenljivu u koju se smešta rezultat A/D konverzije naveli kao *LDR.HighByte* i *LDR.LowByte*, isti princip se može i ovde navesti. Ipak je iskorišćen

i drugačiji način deljenja promenljive A (*A.Byte1* i *A.Byte0*, što ima isti smisao kao i *A.HighByte* i *A.LowByte*).

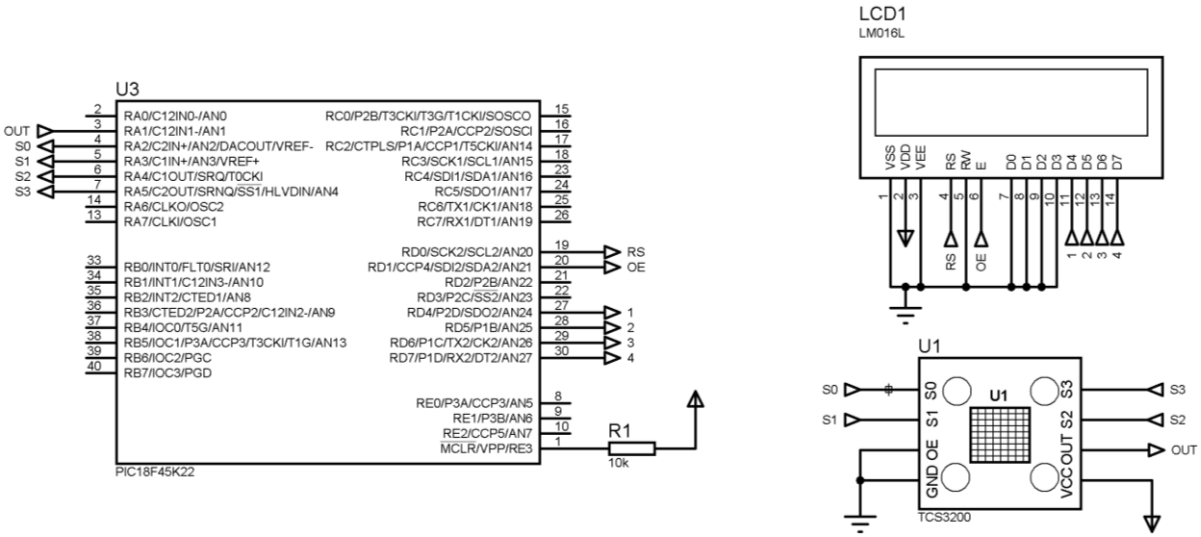
Na kraju, kada je rezultat smešten u promenljivu ***A***, neophodno je tu vrednost podeliti sa ***1.2*** (konsultovati *Data Sheet*), na osnovu čega se dobija intenzitet svetlosti u luxima [lx].

VI vežba

4.6. Kolorimetar

Kolorimetri su uređaji koji su osetljivi na svetlost i koji mere koja i koliko se boja apsorbuje ili reflektuje od nekog objekta ili supstance. Oni određuju boju na osnovu crvene, zelene i plave boje koje su reflektovane sa objekta koji se meri, kao što je slučaj i kod ljudskog oka. Kada svetlost prođe kroz medijum, deo svetlosti se apsorbuje i kao rezultat dolazi do smanjenja emitovane svetlosti zbog toga što se svetlost reflektuje. Kolorimetri mere tu promenu između apsorbovane i reflektovane svetlosti tako da korisnici mogu da analiziraju koncentraciju određene supstance u medijumu, ili čak samu boju objekta koji se meri.

U ovom primeru biće realizovan kolorimetar korišćenjem modula ***TCS230/3200***, koji predstavlja senzor boje. Za prikaz detektovane boje korišćen je LCD displej, uspomoć kog je izvršena kalibracija spektra od ***10*** boja koje ovaj senzor prepoznaje u ovom primeru. Pored RGB boja (osnovne boje), mogu se detektovati i žuta, crna, bela, braon i ljubičasta boja. Električna šema kolorimetra je data na slici 64.



Slika 64. Električna šema kolorimetra

Sa električne šeme se vidi da je senzor boje *TCS3200* vezan na port ***A***, tačnije, pinovi za regulaciju preciznosti detekcije ***S0*** i ***S1*** su vezani na pinove ***RA.1*** i ***RA.2***, respektivno, dok su pinovi zaduženi za fotodiodu određene boje ***S2*** i ***S3***, vezani na pinove ***RA.4*** i ***RA.5***, respektivno. U tabelama 1. i 2. su dati logički nivoi pinova S0, S1, S2 i S3 sa njihovim namenama.

S0	S1	Preciznost frekvencije
Low	**Low**	Power down
Low	**High**	2%
High	**Low**	20%
High	**High**	100%

Tabela 1. Logički nivoi pinova S0 i S1 i nivoi preciznosti

S2	S3	Boja
Low	**Low**	Crvena
Low	**High**	Plava
High	**Low**	Bez filtera (Clear)
High	**High**	Zelena

Tabela 2. Logički nivoi pinova S2 i S3 i izbor fotodiode za određenu boju

Na osnovu logičkih nivoa na pinovima S0 i S1, bira se preciznost frekvencije detektovane svetlosti (boje). Tako na primer, pri korišćenju 100% i 20% preciznosti mora se koristiti dodatni ***timer1***, kako bi se preciznije detektovala boja pozadine, koja se osvetljava. U slučaju slabije preciznosti 2%, direktno na osnovu vrednosti frekvencije se može odrediti boja. Na osnovu logičkih nivoa S2 i S3, bira se fotodioda za određenu boju, čiji intenzitet treba odrediti. Tako na primer u slučaju logičkih nivoa za crvenu boju, aktiviraju se filtri fotodioda za crvenu bojui vrši se očitavanje intenziteta svetlosti crvene boje. Slično je i za plavu, kao i za zelenu. Kombinacijom vrednosti frekvencije sa fotodioda svake boje (RGB), može se izvršiti kalibracija za širi spektar boja osim osnovnih.

Na početku koda treba definisati na koje pinove je povezan senzor boje i deklarisati LCD displej. Pinovi od S0 do S3 su deklarisani kao izlazni, dok je pin ***OUT*** koji je vezan na pin PORTA.0, deklarisan kao ulzni. Sa ovog pina se očitava frekvencija proporcionalna intenzitetu svetlosti. Zatim sledi definisanje logičkih nivoa za pinove S0 i S1, gde se podešava preciznost frekvencije. U ovom primeru S0 je imao stanje logičke nule, odnosno Low, dok je S1 imao stanje logičke jedinice ili High, čime je određena preciznost očitavanja od 2%. Promenljive koje su korišćene za realizaciju kolorimetra su tipa ***Byte***, jer vrednost frekvencije na osnovu preciznosti ne prelazi veličinu jednog bajta.

Pri aktiviranju kolorimetra, na LCD displeju, biće ispisana poruka u prvom redu „***Kolorimetar***“, a u drugom redu „***TCS3200***“. Ova poruka na displeju će stojati 2 sekunde, nakon čega se ulazi u glavni deo koda u potprogram *main*. U *main*-u najpre se ispituje crvena fotodioda, pamti vrednost njene frekvencije u promenljivoj Frekvencija i kasnije upisuje u promenljivu Red (za crvenu boju). Zatim sledi ispitivanje za plavu boju i na kraju za zelenu.

Naredba za uzimanje vrednosti frekvencije za svaku boju (RGB), vrši se uspomoć već korišćene naredbe ***PulsIn***. Ova naredba uzima vrednost signala sa pina koji se definiše u samoj naredbi *PulsIn* i upisuje u promenljivu koja se za to koristi. Linije *main*-a za detekciju frekvencije boja i prikaz na LCD displeju su:

```
main:

'=== Crvena boja ===
Low S2
Low S3
Frekvencija = PulsIn OUT,Low
Red = Frekvencija
Print At 1,1, "Red:", Dec3 Red
DelayMS 100
'====================

'=== Plava boja ===
Low S2
High S3
Frekvencija = PulsIn OUT,Low
Blue = Frekvencija
Print At 1,9, "Blue:", Dec3 Blue
DelayMS 100
'====================


'=== Zelena boja ===
High S2
High S3
Frekvencija = PulsIn OUT,Low
Green = Frekvencija
Print At 2,4, "Green:", Dec3 Green
DelayMS 100
'====================
```

Prilikom detekcije boje površine, prateća elektronika na samom senzoru intenzitet reflektovane boje konvertuje u frekvenciju, čija se vrednost naredbom *PulsIn* upisuje u promenljivu *Frekvencija*. Vrednost frekvencije za sve tri boje fotodioda se upisuje u zasebne promenljive *Red*, *Blue* i *Green*, čija se vrednost ispisuje na LCD-u, kako bi korisnik znao koliko iznosi vrednost frekvencije za svaku od tri osnovne boje. Na osnovu vrednosti frekvencije, moguće je na LCD-u prikazati koja je boja detektovana. U osnovnom slučaju, bez dodatne kalibracije, mogu se razlikovati tri osnovne boje – RGB i prikazati poruku o njihovoj detekciji na LCD displeju. Naime, nakon određivanja vrednosti frekvencije za RGB boje, treba napraviti uslove pri kojima je detektovana crvena, plava ili zelena boja, upoređujući njihove vrednosti, tj. vrednosti njihovih frekvencija.

U slučaju detekcije crvene boje, vrednost frekvencije reflektovane boje je manja u odnosu na ostale dve boje. Razlog tome je što boja koja se detektuje (u slučaju tri osnovne boje) najmanje reflektuje onu boju čija se boja poklapa sa fotodiodom za tu boju. Dakle, prilikom detekcije crvene boje, intenzitet reflektovane crvene boje će biti najslabiji, tj. fotodioda za crvenu boju će detektovati najmanji intenzitet reflektovane boje. Odnosom plave

i zelene se samo određuje krajnji uslov za prikaz detektovane boje. Slično je i pri detekciji ostale dve osnovne boje.

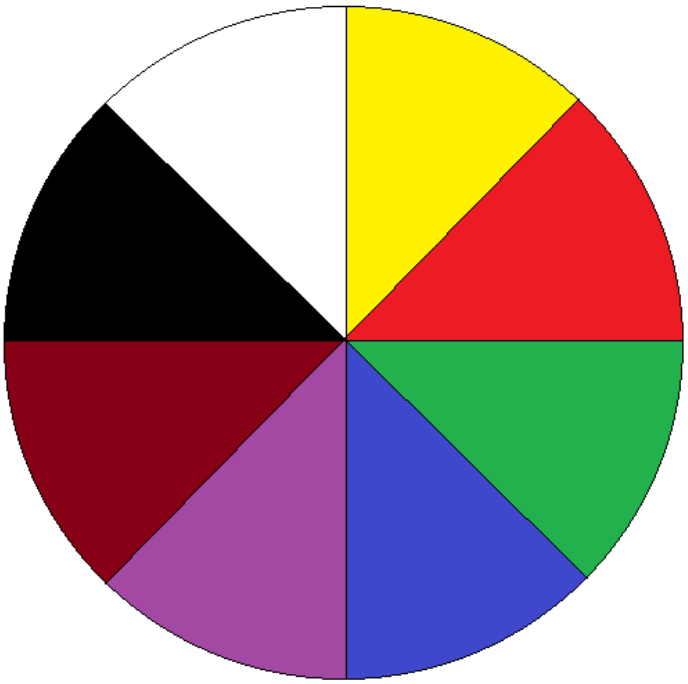
Prilikom očitavanja vrednosti frekvencije za crvenu boju, može se postaviti uslov da je crvena boja manja od plave i zelene boje ili :

```
If Red < Blue And Red < Green Then
    Print At 1,1, " Detektovana je "
    Print At 2,1, "   Crvena boja  "
EndIf
```

Slično je i u slučaju detekcije plave, odnosno zelene boje :

```
If Blue < Red And Blue < Green Then
    Print At 1,1, " Detektovana je "
    Print At 2,1, "   Plava boja   "
EndIf
If Green < Blue And Green < Red Then
    Print At 1,1, " Detektovana je "
    Print At 2,1, "   Zelena boja  "
EndIf
```

Ukoliko korisnik želi da prosiri spektar boja koji se može detektovati i prikazati poruka o detektovanoj boji, neophodno je izvršiti kalibraciju merenja frekvencije. Kalibracija se vrši za svaku boju koja se želi detektovati posebno, gde se pravi opseg vrednosti frekvencije za svaku boju iz željenog spektra. Tako na primer se mogu detektovati bela i crna boja, čije se vrednosti frekvencije nalaze na početku, odnosno na samom kraju opsega, zbog reflektovanja boja. Pored bele i crne boje, mogu se detektovati još i žuta, ljubičasta i braon boja (slika 65.).



Slika 65. Spektar boja za detektovanje

Na osnovu očitavanja vrednosti frekvencije koja je proporcionalna intenzitetu svetlosti boje koja se reflektovala, mogu se odrediti sledeći opsezi frekvencije detektovanih boja (tabela 3.):

Boja	Min	Max
Bela	40	46
Žuta	69	73
Crvena	122	127
Zelena	147	158
Plava	137	142
Ljubičasta	164	169
Braon	222	228
Crna	300	307

Tabela 3. Vrednosti minimalnih i maksimalnih vrednosti frekvencija detektovanih boja

Vrednosti minimalnih i maksimalnih vrednosti frekvencija određivani su na osnovu vrednosti frekvencije za crvenu, plavu i zelenu boju, za svaku boju iz željenog spektra posebno. Tako je na primer za zelenu i plavu boju određen opseg minimalnih i maksimalnih vrednosti na sledeći način:

Zelena boja:

(**Min.**) Red = ***52*** + Blue = ***55*** + Green = ***40*** => **Σ = *147***

(**Max.**) Red = ***56*** + Blue = ***59*** + Green = ***43*** => **Σ = *158***

Plava boja:

(**Min.**) Red = ***68*** + Blue = ***25*** + Green = ***44*** => **Σ = *137***

(**Max.**) Red = ***70*** + Blue = ***26*** + Green = ***46*** => **Σ = *142***

Na isti način određeni su opsezi vrednosti frekvencije za sve boje iz željenog spektra. Linije koda kojima se određuje opseg frekvencija za ovaj spektar boja su:

```
If Red + Blue + Green >= 122 And Red + Blue + Green <= 127 Then
    Print At 1,1, " Detektovana je "
    Print At 2,1, "  Crvena boja   "
EndIf

If Red + Blue + Green >= 137 And Red + Blue + Green <= 142 Then
    Print At 1,1, " Detektovana je "
    Print At 2,1, "   Plava boja   "
EndIf

If Red + Blue + Green >= 300 And Red + Blue + Green <= 307 Then
    Print At 1,1, " Detektovana je "
    Print At 2,1, "   Crna boja    "
EndIf

If Red + Blue + Green >= 147 And Red + Blue + Green <= 158 Then
    Print At 1,1, " Detektovana je "
    Print At 2,1, "  Zelena boja   "
EndIf

If Red + Blue + Green >= 40 And Red + Blue + Green <= 46 Then
    Print At 1,1, " Detektovana je "
    Print At 2,1, "   Bela boja    "
EndIf

If Red + Blue + Green >= 69 And Red + Blue + Green <= 73 Then
    Print At 1,1, " Detektovana je "
    Print At 2,1, "   Zuta boja    "
EndIf

If Red + Blue + Green >= 164 And Red + Blue + Green <= 169 Then
    Print At 1,1, " Detektovana je "
    Print At 2,1, "Ljubicasta boja "
EndIf

If Red + Blue + Green >= 222 And Red + Blue + Green <= 228 Then
    Print At 1,1, " Detektovana je "
    Print At 2,1, "   Braon boja   "
EndIf
```

Vidi se da na osnovu frekvencije RGB fotodioda, moguće je napraviti spektar boja koje korisnik želi da očita uspomoć senzora. Takođe uspomoć ovog senzora moguće je realizovati veliki broj projekata koji zasnivaju svoj rad na detektovanju i razlikovanju boja.

5. Dodatak – Zadaci za vežbu

Poslednje poglavlje u ovom praktikumu je posvećeno onima koji žele da prošire svoje praktično znanje stečeno tokom slušanja ovog predmeta. U nastavku će biti dati zadaci koje studenti mogu samostalno realizovati, kako u simulatoru (Proteus) ili praktično, kod kuće ili na fakultetu u laboratoriji. Na osnovu ovih zadataka, studenti će imati uvid u nivo stečenog znanja vezanog kako za Optoelektroniku, tako i za programiranje mikrokontrolera.

1. **LED dimer**
   Realizovati LED dimer korišćenjem PWM modulacije. LED diodu vezati na pin **PORTA.2**. Naredba kojom se vrši PWM modulacija je PWM. Sintaksa ove naredbe je :

   PWM, *Port.Pin***,** *Duty***,** *Cycles*

   s tim što *Duty* predstavlja vrednost naponskog nivoa (0 – 255 => 0 – 5V), a *Cycles* predstavlja dužinu (u milisekundama (ms)) trajanja jednog naponskog nivoa, počevši od minimalne do maksimalne vrednosti napona u parametru *Duty*.

2. **Voltmetar**
   Realizovati Voltmetar, za merenje napona u opsegu od 0 do 12V. A/D konvertor realizovati na pinu AN.0. (*Voditi računa da se na ulaz A/D konvertora može dovesti najviše 5V*).

3. **IC (InfraCrveni) bezbedonosni sistem**
   Realizovati IC bezbedonosni sistem, tako da je senzor (fototranzistor) vezan na pinu PORTA.1. Kada je prekinut snop sa izvora svetlosti (IC dioda), na LCD displeju ispisati poruku „UZBUNA“, a kada se ponovo uspostavi kontinualni snop svetlosti, na LCD displeju ispisati poruku „Sistem ON“.

4. **Brojač posetilaca**
   Realizovati brojač posetilaca (0-999), korišćenjem IC sistema (IC dioda i fototranzistor). Broj posetilaca ispisivati na sedmosegmentnim displejima.

5. **Pozdravna poruka na LED matrici**
   Na LED matrici prikazati pozdravnu poruku „Dobrodošli“, koja će se na LED matrici kretati brzinom od 140ms. Karaktere je neophodno iskodirati po uputstvu kao u primeru opisanom u praktikumu.