Maketa sustava za sortiranje proizvoda prema boji

Balent, Zvonimir

Master's thesis / Diplomski rad

2017

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:182080

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
ELEKTROTEHNIČKI FAKULTET

Sveučilišni studij

MAKETA SUSTAVA ZA SORTIRANJE PROIZVODA PREMA BOJI

Diplomski rad

Zvonimir Balent

Osijek, 2016

SADRŽAJ

1. UVOD

U svakodnevnom životu pokretne trake imaju veliku ulogu na mnogim mjestima. Prve pokretne trake su se koristile u dalekoj prošlosti, može se reći da je prva pokretna traka izumljena prije nekoliko tisuća godina. Pri izgradnji velikih palača i hramova, ljudi su se koristili balvanima kao valjcima i na taj način transportirali teške terete. Baš ta primjena balvana može se smatrati preteča pokretnih traka. Prve pokretne trake imale su drvene valjke, a trake su bile izrađene od kože ili gume. Danas se koriste metalni valjci, a trake su od gume, kože, pamuka, tkanina, najlona, pvc, silikona i čelika. Ispočetka su se pokretne trake koristile za prenošenje manjih tereta, ali s poboljšavanjem konstrukcije povećavala se i nosivost. Mnoge pokretne trake su izrađene od metalnih dijelova, ali se sredinom dvadesetog stoljeća počinje koristiti plastika u izradi pokretnih traka. Plastika je puno lakši i jeftiniji materijal, ali ima manu što se ne može koristiti pri velikim temperaturama. Pokretne trake koriste se u zračnim lukama, trgovačkim centrima, proizvodnim tvornicama, rudnicima, teretanama, restoranima i drugim mjestima. U trgovačkim centrima primjer pokretne trake su pokretne stepenice. U zračnim lukama one služe za transport putničke prtljage ili pošiljaka. U rudnicima se transportira ruda. U teretanama se pokretne trake koriste za vježbanje te je na taj način osobama omogućena simulacija trčanja u malom prostoru. U restoranima se poslužuje hrana na pokretnim trakama.

Veliki potencijal pokretnih traka uvidio je Henry Ford te je među prvima koristio traku za industrijske primjene. On je 1913. godine u svoj pogon za sastavljanje automobila uveo pokretnu traku. Na pokretnoj traci su radnici sastavljali automobil i tako su ubrzali proizvodnju automobila. Henry Ford je tako uspio povećati svoju produktivnost, a samim uvođenjem pokretne trake smanjiti troškove proizvodnje automobila. Uvođenjem pokretne trake u pogon, ubrzano je sastavljanje jednog automobila za četiri puta. Nakon njega mnoge tvornice su počele s ugradnjom pokretnih traka u svoje pogone.

U današnje vrijeme, pokretne trake su se znatno razvile. Počele su se koristiti i za transport robe na velike udaljenosti. Zanimljivo je da postoji niz pokretnih traka sveukupno dugačkih 97 kilometara. Taj najdulji niz pokretnih traka nalazi se od rudnika fosfata u Bu Craau do El Aaiunu. Najdulja pokretna traka nalazi se Aziji i dugačka je 17 kilometara [1].

U okviru diplomskog rada potrebno je načiniti maketu pokretne trake koja ima mogućnost sortiranja proizvoda prema boji. Maketa treba biti opremljena odgovarajućim senzorima i aktuatorima te procesnim računalom (mikroupravljač) s pripadajućom periferijom za prikaz podataka i unos podataka. Nakon izgradnje i ispitivanja makete, u mikroupravljač je potrebno implementirati algoritam za upravljanje maketom. Nadalje, potrebno je izraditi aplikaciju za nadzor i upravljanje maketom koji se izvršava na osobnom računalu.

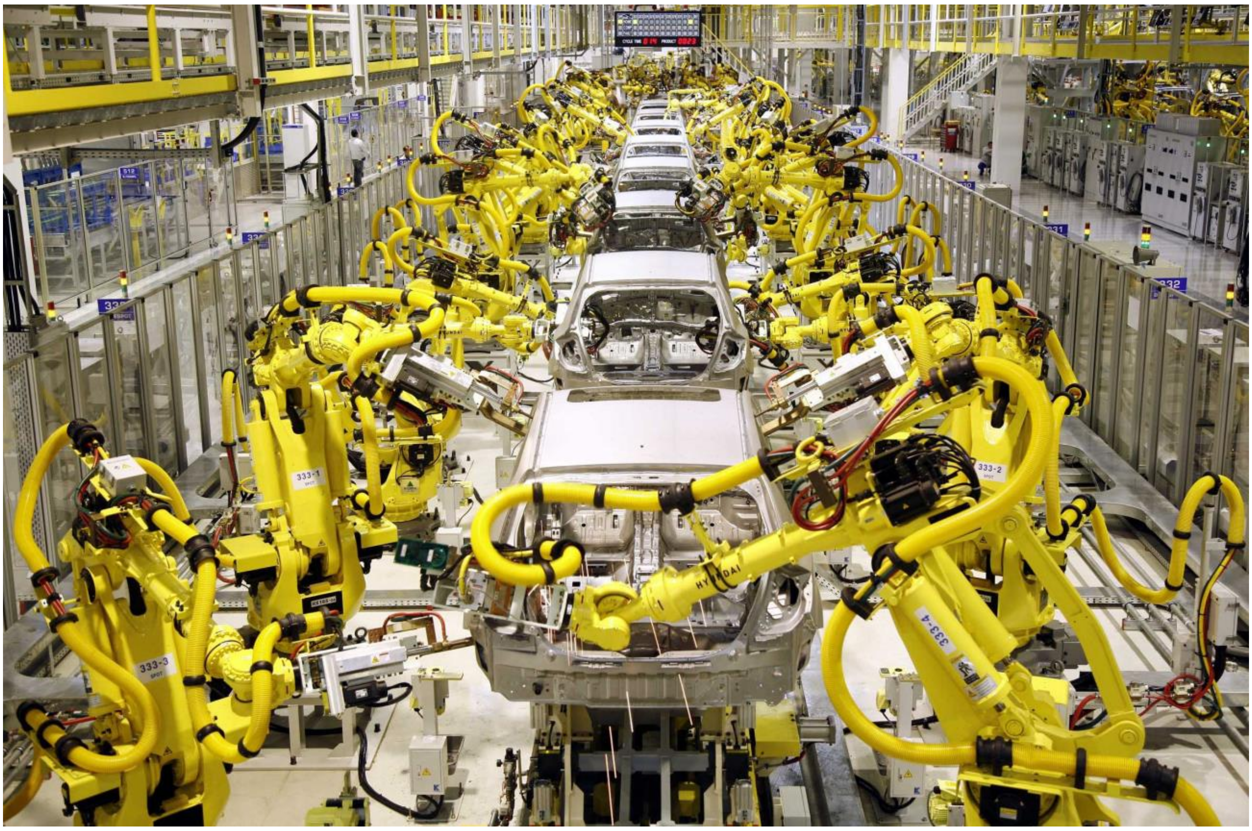
Rad je strukturiran na sljedeći način. U drugom poglavlju opisane su pokretne trake, njihova primjena, te dijelovi. U trećem poglavlju opisani su zahtjevi makete, mehanički i električni dijelovi makete i upravljački sustav kojeg tvore mjerni i izvršni članovi, mikroupravljač i dopunski članovi. Nadalje, projektiran je algoritam upravljanja maketom trake te su predstavljeni zahtjevi na PC aplikaciju za nadzor i upravljanje. U četvrtom poglavlju opisana je implementacija upravljačkog algoritma u Arduino platformu te izrađena PC aplikacija za nadzor i upravljanje maketom.

2. POKRETNE TRAKE U INDUSTRIJI

U ovom poglavlju opisane su pokretne trake te osnovni dijelovi. U prvom potpoglavlju navedena je definicija i primjena pokretne trake. U drugom potpoglavlju opisani su dijelovi pokretne trake i njihova uloga.

2.1. O pokretnim trakama

Pokretna traka je transportni uređaj koji pomoću beskrajne trake prenosi materijal između dvije točke. U industriji pokretne trake služe za transport velikih količina materijala, ali i za sastavljanje uređaja. Kad se pokretna traka koristi za sastavljanje uređaja, onda se podrazumijeva da brzina trake ovisi o radniku ili robotu koji sastavlja uređaj. Na slici 2.1. prikazana je pokretna traka za izradu automobila uz pomoć robotske ruke [2].

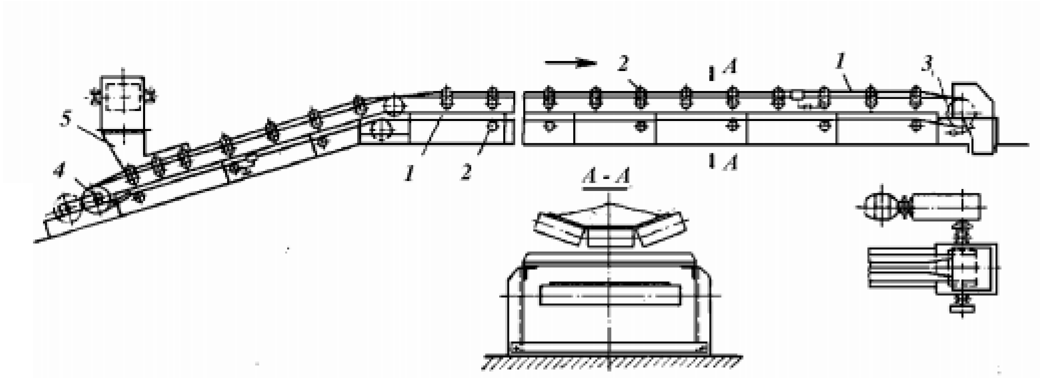


Sl. 2.1 Pokretna traka za proizvodnju automobila [2].

2.2. Osnovni dijelovi pokretnih traka

Pokretna traka ili trakasti transporter sastoji se od pogonskog bubnja, zateznog bubnja, trake, valjaka i utovarnog uređaja. Jedan valjak je pogonski, a drugi služi za zatezanje trake. Pogonski valjak je vezan uz aktuator koji stvara rotacijsko gibanje, a to je najčešće elektromotor.

Rotacijsko gibanje se putem trenja prenosi s valjka na traku. Valjak za zatezanje koristi se kako bi traka uvijek bila dovoljno zategnuta. Ako je traka previše zategnuta, može doći do zaglavljenja. Zaglavljenje trake može prouzročiti pucanje trake, kvar na elektromotoru ili druge kvarove.

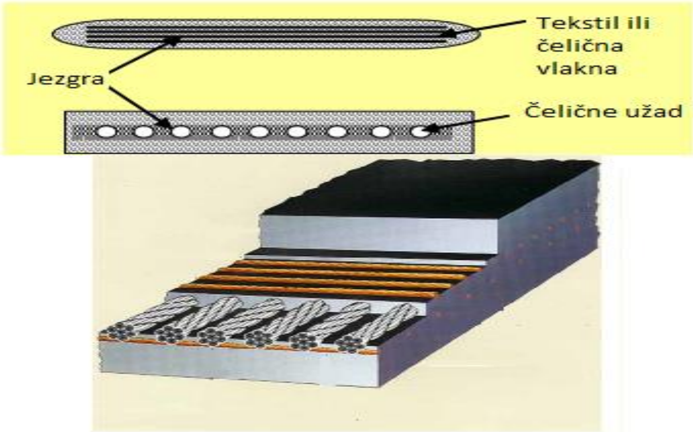


Sl. 2.2. Dijelovi pokretne trake [3].

Na slici 2.2. prikazani su dijelovi pokretne trake. Na slici brojevima su označeni sljedeći dijelovi:

1. Traka
2. Valjci
3. Pogonski valjak
4. Zatezni valjak
5. Utovarni uređaj

Traka je najznačajniji dio pokretne trake i obavlja funkciju prijenosa materijala. Trake predstavljaju čak 30% cijene cijele pokretne trake. Trake se najčešće izrađuju od gume (plastike), čelika ili pletene žice. U industriji se najčešće koriste trake izrađene od gume. Gumene trake imaju složenu teksturu koja se sastoji od jezgre i gumenog omotača. Jezgra se sastoji od tekstilnih ili čeličnih vlakana. Jezgra je zadužena za čvrstoću i prijenos sile naprezanja dok gumeni omotač štiti jezgru od vanjskih utjecaja i mehaničkih oštećenja. Na slici 2.3. prikazana je struktura gumene trake.

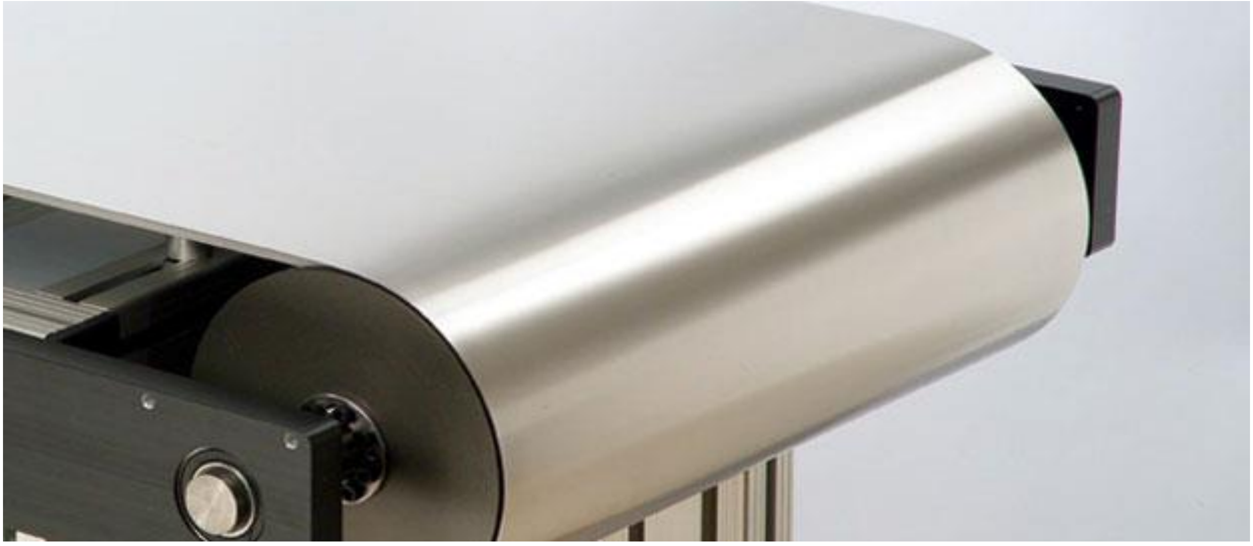


Sl. 2.3. Struktura gumene trake [3].

Čvrstoća pokretne trake ponekad nije važna pa se u izradi koristi pamuk koji ima zateznu čvrstoću 50 do 100 N/mm. Ako je potreba velika čvrstoća pokretne trake, onda se koriste čelične užadi koje imaju zateznu čvrstoću do 10000 N/mm. Velikom napretku u čvrstoći doprinosio je razvoj kevlara. Kevlar je sintetički materijal koji ima svojstvo velike čvrstoće, male mase te dobro podnosi visoke temperature. Kevlar se inače koristi u proizvodnji brodova, aviona, svemirskih raketa. Zatezna čvrstoća kevlera iznosi oko 3600 N/mm.

Omotač se izrađuje od gume ili plastike. Ne postoji velika razlika u konstrukciji omotača ako je izrađen od gume ili od plastike, ali su plastični omotači otporniji na vlagu, kemikalije, ulja. Omotači se proizvode u standardiziranim širinama koje iznose za gumene trake od 300 do 3600mm, a za plastične trake od 200 do 4500mm.

Čelične trake se rade od tanko valjanog čelika koji je presvučen gumom. Karakteristike takvih traka su velika robusnost, čvrstoća, bešuman rad te dobro ponašanje pri velikim temperaturama. Vrlo često se takve trake koriste i za tiskanje oznaka na proizvode. Najveća prednost ovih traka je mogućnost rada u postrojenjima povišene temperature. Često se koriste u postrojenjima s prljavom okolinom kao što su kamenolomi i rudnici. Širine ovih traka su standardizirane i iznose od 200 do 1500mm. Maksimalna dužina transporta je do 2000m. Debljina čelične trake iznosi do 2mm, a brzina trake je do 2 m/s. Najveći nedostatak je zamor materijala trake. Pri pokretanju trake potrebno je uzeti u obzir veliku početnu silu zatezanja trake. Kod ovakvih traka se ne primjenjuju bubnjevi nego se koriste paralelno postavljeni diskovi. Pri dimenzioniranju čeličnih traka mora se uzeti u obzir sigurnost pa se samim time rubovi trake moraju zaobliti. Na slici 2.4. prikazana je čelična traka.

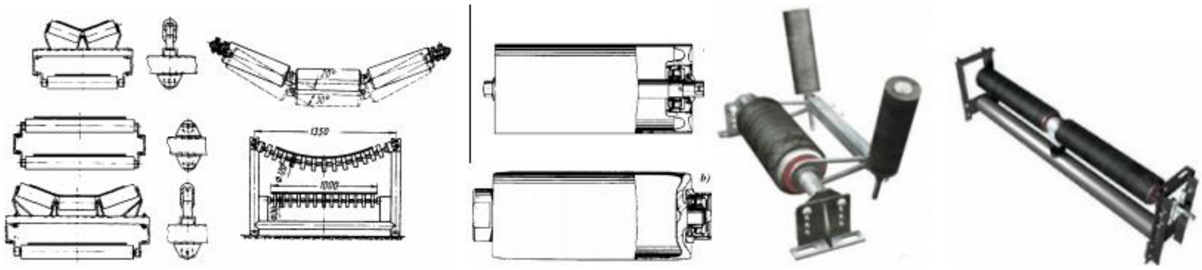


Sl. 2.4.Čelična traka [4].

Traka od pletene žice koristi se kada se prenosi materijal koji nije rasipan. Najčešće se koristi u farmaceutskoj, prehrambenoj i kemijskoj industriji za prijenos gotovih proizvoda pakiranih u kartonskoj ambalaži. Ova vrsta pokretnih traka je znatno rjeđe u uporabi nego prethodne dvije. Na slici 2.5. prikazana je traka od pletene žice korištena u prehrambenoj industriji.

Sl. 2.5. Traka od pletene žice korištena u prehrambenoj industriji [5].

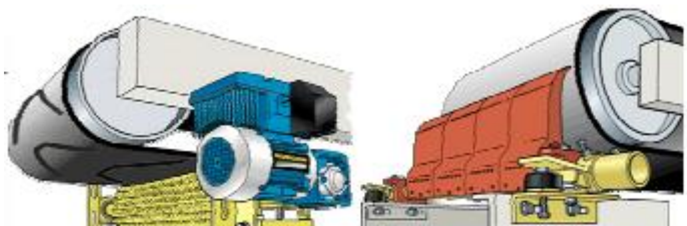
Valjci se koriste kao nosači traka i tereta, te mogu utjecati na oblik trake. Gustoća postavljanja valjaka ovisi o masi proizvoda koji će se transportirati i o duljini trake. S povećanjem mase transportnog proizvoda potrebno je postaviti što gušće valjke. Valjci zauzimaju veliku stavku u cijeni cijele pokretne trake, a njihov udio iznosi do 25% ukupne cijene pokretne trake. Vijek trajanja valjka ovisi o području rada pokretne trake i o materijalu koji se transportira. Na slici 2.6. prikazani su razni modeli valjaka.



Sl. 2.6. Modeli valjaka [3].

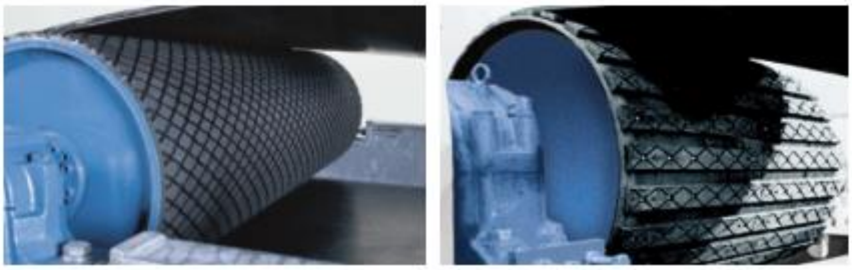
Vrlo važan element valjka su ležajevi. Vijek valjka najčešće će se znatno produžiti ako se postave kvalitetni ležajevi. Ležajevi se moraju zaštititi od utjecaja okoline kao što su prašina i vlaga. Pravilnim održavanjem ležajeva smanjuje se opterećenje aktuatora koji pokreće cijelu pokretnu traku. Valjci mogu usmjeravati traku, tj. mogu od trake napraviti određeni profil. Na slici 2.6. prikazani su valjci koji prave određeni profil trake.

Ponekad je potrebno čišćenje pokretne trake. Na krajnjem dijelu pokretne trake postavlja se strugalica koja se može namještati ručno ili se namješta automatski pomoću opruge. Strugalica je prislonjena na pokretnu traku i s nje skida nečistoće. Postoji varijanta čišćenja s valjcima. Princip je sličan kao i kod strugalice. Valjci za čišćenje su obloženi čeličnim ili plastičnim četkama koje skidaju nečistoću s pokretne trake. Na slici 2.7. prikazana su dva načina čišćenja pokretne trake.



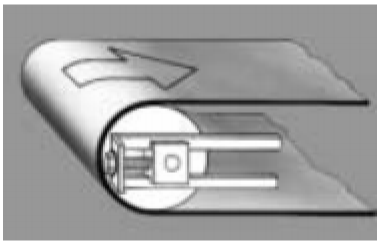
Sl. 2.7. Čišćenje pokretne trake [3].

Jedan valjak uvijek mora biti pogonski. Ako je valjak pogonski on mora na sebi sadržavati aktuator koji pokreće cijelu pokretnu traku. Vrlo važna stavka kod pogonskog valjka je trenje između površine valjka i trake jer se preko trenja prenosi moment na traku. Koeficijent trenja između bubnja i trake nije uvijek konstantan, on ovisi o površini trake i bubnja i o brzini kretanja trake. Povećanjem radnih sati pokretne trake, dolazi do smanjenja trenja jer se dogodi „glačanje površine“. Prilikom dizajniranja pokretne trake mora se uzeti u obzir okolina. U prisutnosti vode ili neke druge tvari može doći do promjene u trenju između valjka i trake. Iz tog razloga, ako se pokretna traka nalazi na otvorenom, onda je najbolje koristiti valjak s izbočinama. Na slici 2.8. prikazani su pogonski valjci i površina valjka.



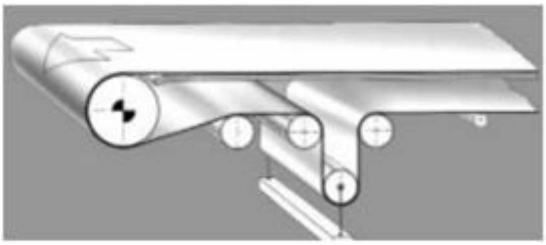
Sl. 2.8. Pogonski valjci i njihova površina [3].

Pokretnu traku je ponekad potrebno zategnuti ili popustiti. S vremenom je moguće da traka promijeni svoje dimenzije, pa je potreban zatezni uređaj. Zatezni uređaj najčešće je jedan od valjaka. Najčešće je to valjak na kojem je najmanja sila. Ako traka nije dovoljno zategnuta, može dovesti do klizanja trake. Na slici 2.9. prikazan je uređaj za ručno zatezanje trake.



Sl. 2.9. Valjak s ručnim uređajem za zatezanje trake [3].

Kod velikih postrojenja nije moguće ručno zatezanje trake pa se postavlja automatsko zatezanje trake. Najčešće se to izvodi uz pomoć dodatnog valjka koji regulira zatezanje trake pomoću opruga. Na slici 2.10. prikazano je automatsko zatezanje trake.



Sl. 2.10. Automatsko zatezanje trake [3].

Utovarni uređaj služi da bi se na pokretnu traku postavio materijal za transport. Utovarni uređaj može biti lijevak s dozatorom ili robotska ruka. U nekim procesima utovarni uređaj nije stroj nego materijal utovara čovjek. Vrlo važno je dobro uskladiti utovarni uređaj s pokretnom trakom. Ako materijal pada na pokretnu traku, bilo bi dobro podesiti utovarni uređaj da je na što manjoj visinskoj razlici od pokretne trake jer će se samim time manje uništavati traka.

3. MAKETA SUSTAVA ZA SORTIRANJE PROIZVODA PREMA BOJI

U ovom poglavlju detaljnije je opisan postupak izrade makete. U prvom potpoglavlju ovoga poglavlja opisani su zahtjevi koji su postavljeni prije same izrade makete, a zatim projektiranje makete koja će udovoljavati postavljenim zahtjevima. U drugom potpoglavlju opisana je izgradnja makete, te njezini mehanički i električni dijelovi. Prikazan je nacrt, tlocrt, bokocrt i električna shema makete.

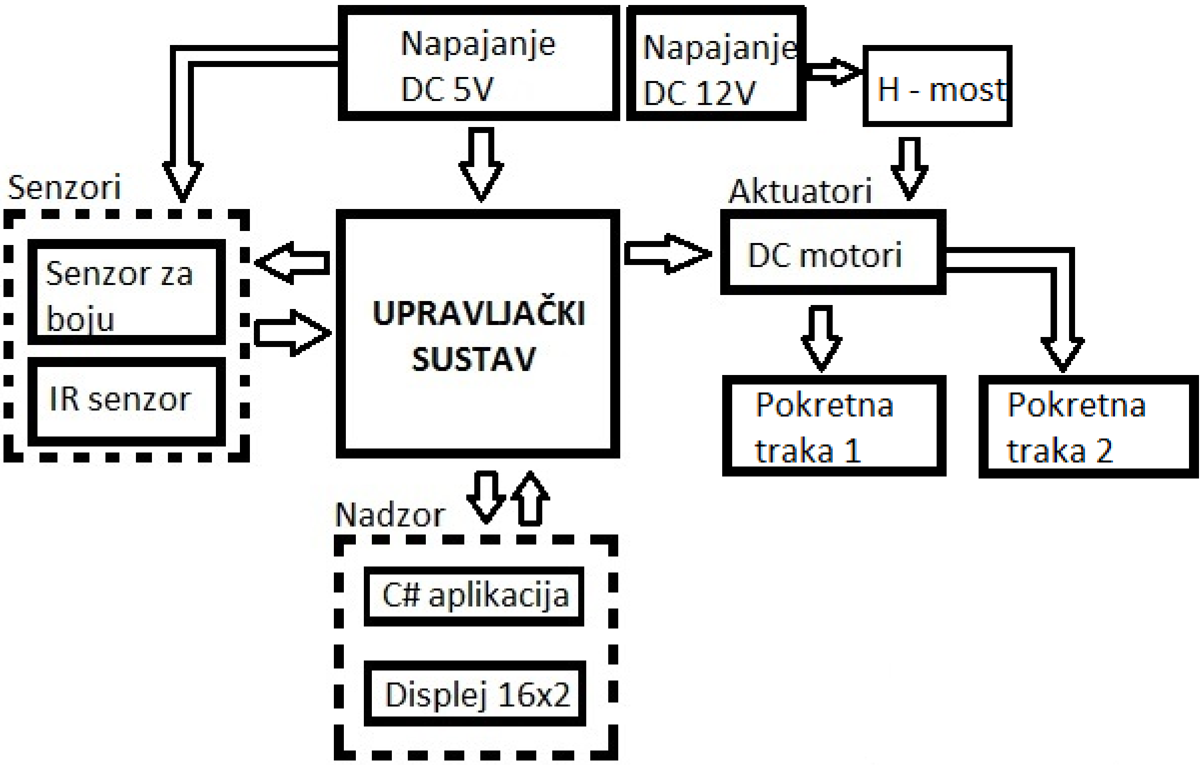
3.1. Projektiranje makete

Pri projektiranju makete potrebno je odrediti zahtjeve koje maketa mora ispunjavati. Prije samog postupka projektiranja i izrade makete postavljeno je 7 zahtjeva na maketu:

1. Dimenzije makete
2. Cijena
3. Transport proizvoda
4. Određivanje boje predmeta na pokretnoj traci
5. Brojač proizvoda
6. Upravljanje pomoću mikroupravljača
7. Nadzor i upravljanje maketom pomoću osobnog računala

Maketa treba biti praktična za prijenos od strane jedne osobe. Dimenzije makete ne smiju prelaziti 400x400x100 mm. Drugi zahtjev makete je cijena, a pri projektiranju makete uzeto je u obzir da maketa bude što jeftinija. Zbog niske cijene odabrana su dva istosmjerna motora za pokretanje pokretnih traka. Rješenje trećeg zahtjeva su dvije pokretne trake koje prenose proizvod. Prva pokretna traka izvršava transport proizvoda pokraj detektora boje proizvoda i poslije detekcije prosljeđuje proizvod na drugu traku koja potom sortira proizvod prema boji. Kako bi se omogućilo usporavanje, zaustavljanje i pokretanje trake u određenom trenutku, traka treba biti opremljena sa senzorima za detekciju prisutnosti proizvoda na određenim dijelovima makete. Idealno rješenje za transport proizvoda bila bi pokretna traka s kamerom i robotskom rukom koja sortira proizvode, ali takvo rješenje daleko je skuplje i kompliciranije. Četvrti zahtjev je određivanje boje, a riješen je sa senzorom za prepoznavanje boje. Senzor mora prepoznavati najmanje tri osnovne boje (crvenu, zelenu i plavu). Peti zahtjev je brojanje proizvoda koji su napustili pokretnu traku. Brojanje je omogućeno opremanjem makete s infracrvenim senzorima. Šesti zahtjev je da maketa mora biti upravljana s odgovarajućim procesnim računalom tj. mikroupravljačom. Mikroupravljač po svojim specifikacijama mora zadovoljavati zahtjeve makete. Za ovu maketu odabrana je mikroupravljačka pločica Arduino Mega 2560. Sedmi zahtjev je izrada aplikacije za upravljanje i nadzor cijelog sustava preko računala. Potrebno je na jednostavan i vizualno prihvatljiv način prikazati očitanu boju sa senzora, stanja pokretnih traka, brojače proizvoda, te ostale podatke s makete. U aplikaciji mora se omogućiti ručni i automatski rad makete. Navedeni zahtjev moguće je ispuniti izradom PC aplikacije koja serijskom vezom komunicira s mikroupravljačom na maketi.

Na slici 3.1. prikazana je shema predloženog sustava za sortiranje proizvoda prema boji. Upravljački sustav je glavni dio ili „srce" cijele makete, te su na njega povezani izvršni i mjerni članovi. Dio za nadzor sastavljen je od C# aplikacije i zaslona, te je povezan s upravljačkim sustavom. Istosmjerno napajanje 5V koristi se za upravljački sustav i senzore, a istosmjerno napajanje 12V za istosmjerne motore.



Sl. 3.1. Blokovska shema predloženog sustava za sortiranje proizvoda prema boji.

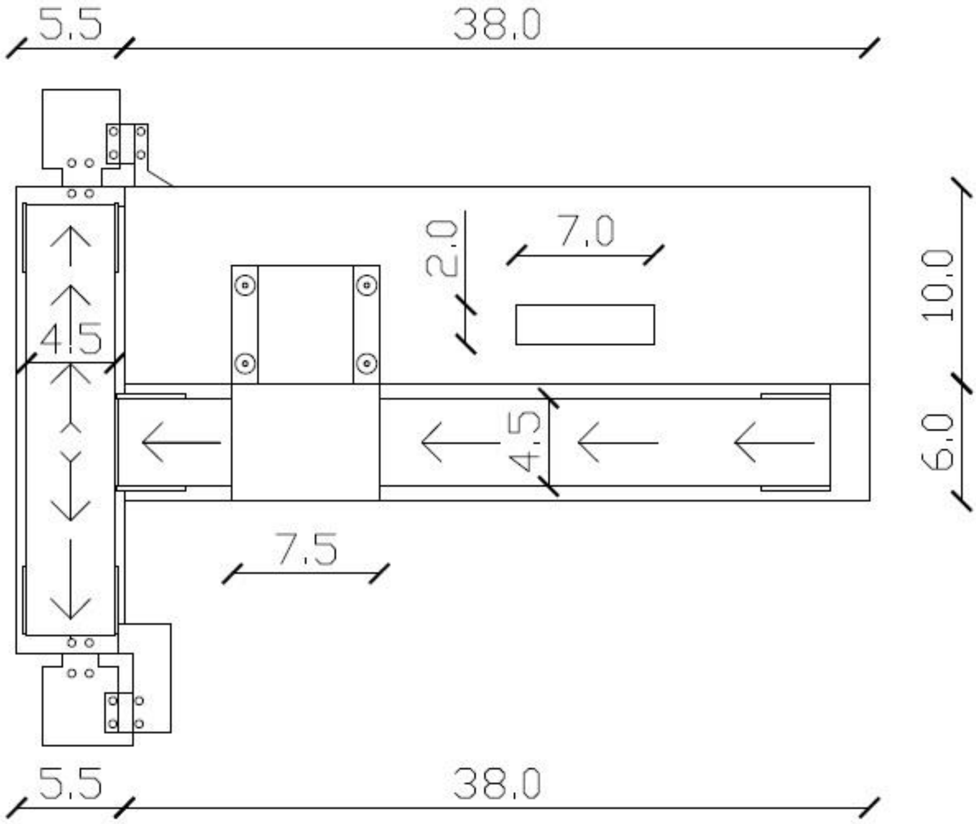
3.2. Izgradnja makete

3.2.1. Mehanički dio makete

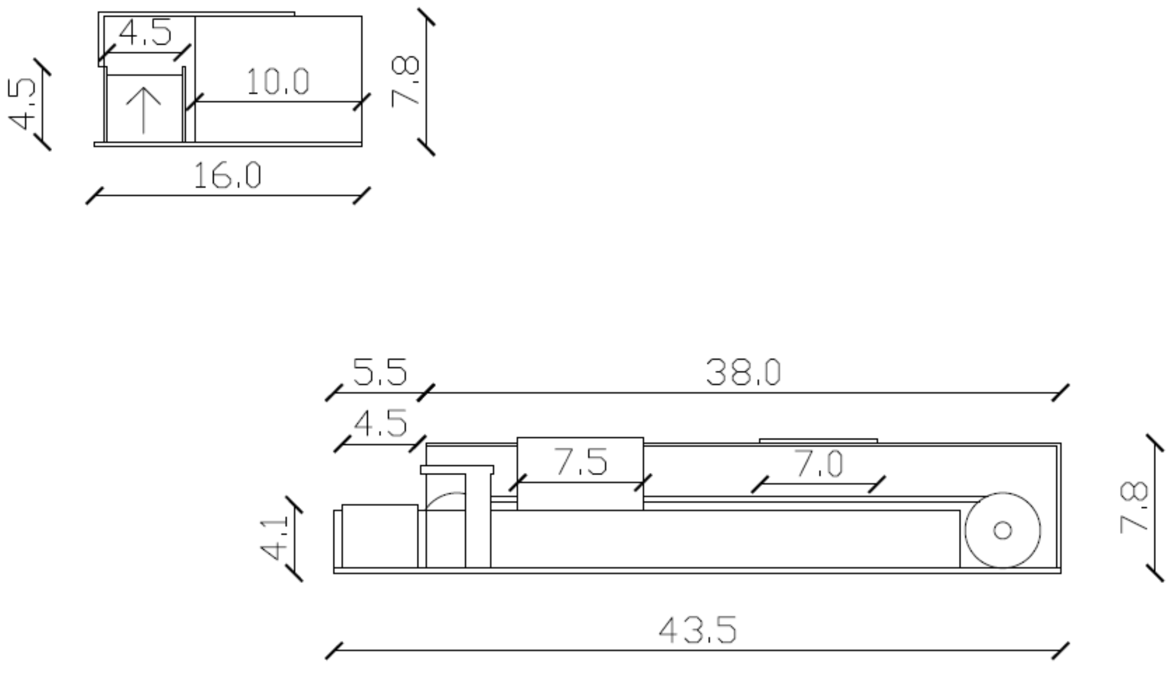
Dimenzije makete sustava za sortiranje proizvoda prema boji su 450mm x 340mm x 80mm. Maketa se nalazi na postolju koje je napravljeno od aluminijske ploče. Aluminijsko postolje postavljeno je na četiri noge koje su tokarene od tvrde plastike. Istosmjerni motori su postavljeni na postolje i pozicionirani uz pomoć vijaka. Valjak istosmjernog motora za prvu traku tokaren je od aluminija te ima promjer 40mm. Valjak istosmjernog motora za drugu traku tokaren je od tvrde plastike te ima promjer 30mm. Valjci su na rotor istosmjernog motora pričvršćeni uz pomoć vijaka bez glave. Tijelo valjka probušeno je u centru i napravljen je navoj u koji se postavljaju vijci koji stežu valjak. Ostala dva valjka su promjera 30mm i oni su postavljeni na komad metala promjera 5mm i duljine 65mm. Na slici 3.2. prikazan je aluminijski valjak pričvršćen na istosmjerni motor [6].

Sl. 3.2. Istosmjerni motor s aluminijskim valjkom.

Kućište pokretnih traka napravljeno je od metalnih kanalica koje se koriste za postavljanje električnih instalacija u zgradama. Širina pokretnih traka je 50mm. Prva traka je dugačka 360mm, a druga 220mm. Nosači IR senzora i senzora za očitavanje boje napravljeni su od čeličnog lima debljine 1mm i savijani su u željeni položaj. Nosači za spremnike su pravljeni od čeličnog lima debljine 2mm. Spremnici su savijani od čeličnog lima debljine 1mm te su na postolje pričvršćeni uz pomoć šarke. Senzor za očitavanje boje pokriven je limom da bi se postigao što manji utjecaj okoline na senzor.



Sl. 3.3. Tlocrt makete sustava za sortiranje proizvoda prema boji.

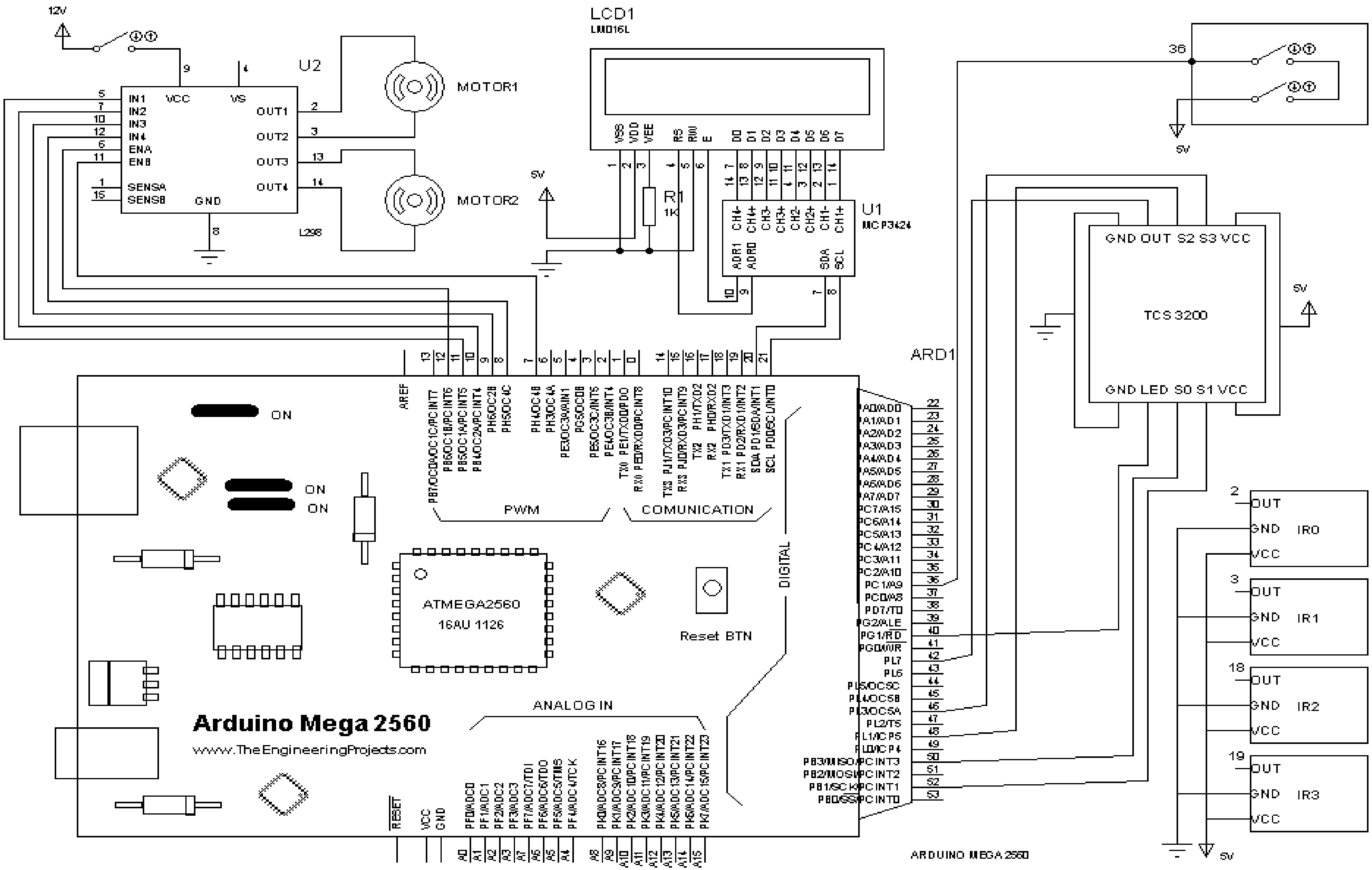


Sl. 3.4. Bokocrt i nacrt makete sustava za sortiranje proizvoda prema boji.

Sl. 3.5. Maketa sustava za sortiranje proizvoda prema boji.

3.2.2. Električni dio makete

Maketa sustava za sortiranje proizvoda prema boji upravljana je s mikroupravljačkom pločicom Arduino Mega 2560. Arduino Mega 2560 je najvažnija komponenta cijelog električnog dijela makete. Mikroupravljačka pločica se napaja USB kablom. Univerzalna serijska sabirnica (USB) je tehnološko rješenje za komunikaciju računala s vanjskim uređajima pri čemu se podaci razmjenjuju serijski relativno velikom brzinom. Upravljanje istosmjernim motorima obavlja se uz pomoć H mosta. U ovoj maketi korišten je H most oznake L298N. Napajanje H mosta obavlja se s adapterom koji pretvara 230V izmjenični napon u istosmjernu napon 12V. Između adaptera i H mosta postavljen je prekidač. Pomoću H mosta napajaju se oba istosmjerna motora. Iz mikroupravljača na H most spaja se ukupno šest vodiča (napajanje i upravljački signali). Po dva vodiča za svaki istosmjerni motor koja određuju smjer vrtnje istosmjernog motora i po jedan vodič za svaki istosmjerni motor koji određuje brzinu vrtnje elektromotora. Napajanje LCD 16x2 zaslona vrši se istosmjernim naponom od 5V s mikroupravljača. U maketi je korišten i2c zaslon pa se slanje i primanje podataka između mikroupravljača i zaslona obavlja pomoću dvije signalne linije, SDA i SCL. Za potrebe detekcije boje proizvoda na traci odabran je TCS 3200 senzor za raspoznavanje boja. Senzor za raspoznavanje boje napaja se preko mikroupravljača s istosmjernim naponom od 5V. Za komunikaciju između mikroupravljača i senzora za raspoznavanje boje potrebno je šest vodiča. Za prepoznavanje položaja na pokretnoj traci koriste se četiri IR senzora. Svaki IR senzor napajan je preko mikroupravljača s istosmjernim naponom od 5V. Komunikacija između mikroupravljača i svakoga senzora obavlja se s po jednim vodičem za svaki IR senzor.



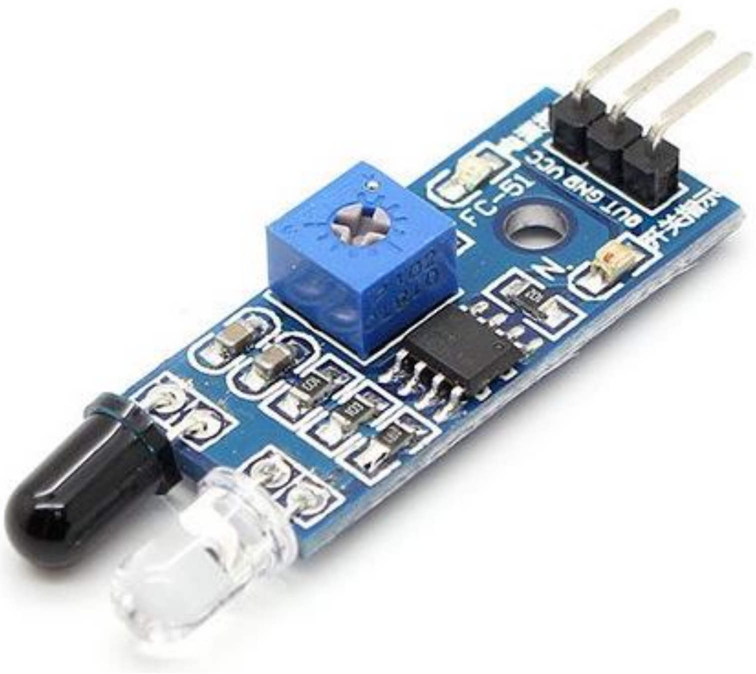
Sl. 3.6. Električna shema makete.

3.3. Upravljački sustav

3.3.1. Mjerni članovi

Mjerni članovi ili senzori su uređaji koji mjere fizikalne veličine i pretvaraju ih u signal pogodan za daljnju obradu. Najčešće je to električni signal. Najkorišteniji tipovi mjernih članova kod pokretnih traka su senzori pokreta. Kada proizvod dođe do mjesta koje je označeno svjetlosnom zrakom, proizvod je prekida i time omogućuje senzoru da registrira pokret, te tu informaciju šalje do glavnog računala koje prati proces proizvodnje [7].

U ovome radu korištena su četiri infracrvena senzora. Ovaj mjerni uređaj nema točnu proizvodnu oznaku. Zasniva se na LM393 čipu i IR odašiljaču i prijamniku. Polje detekcije zauzima kut od 35 stupnjeva, a maksimalna udaljenost je 30cm. Na slici 3.7. prikazan je infracrveni senzor koji je korišten u maketi za detekciju predmeta na odgovarajućim dijelovima pokretne trake.



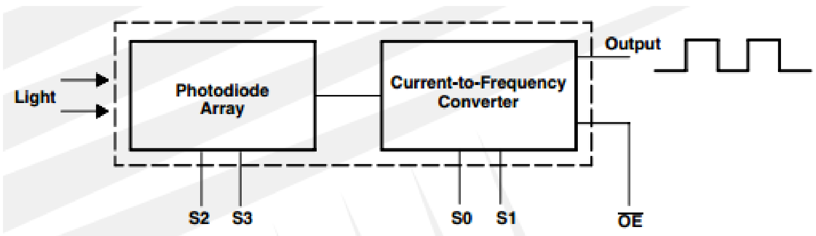
Sl. 3.7. Infracrveni senzor s LM393 [9].

Ovaj mjerni član pripada skupini digitalnih infracrvenih senzora. Služi za detekciju prisutnosti predmeta ili objekta na pokretnoj traci. Posjeduje tri pina za spajanje koji su označeni s oznakama GND, VCC, OUT. Pinovi VCC i GND služe za napajanje sklopa, a pin OUT spaja se na digitalni pin mikroupravljača. Ovaj senzor posjeduje i zelenu svjetlosnu diodu te tako signalizira prisutnost proizvoda ispred senzora.

Za prepoznavanje boje predmeta koristi se mjerni član TCS3200. Ovaj mjerni član napaja se s naponom od 2.7V do 5V, a dimenzije su mu 28.4mm x 28.4mm. Maksimalna udaljenost na koju može prepoznati boju objekta je 40mm, a idealna udaljenost je 10mm. Na slici 3.8. prikazan je mjerni član za prepoznavanje boje TCS3200.

Sl. 3.8. TCS3200 mjerni član za prepoznavanje boje [10].

TCS 3200 je senzor za prepoznavanje boje koji pretvara svjetlost u frekvenciju. Ovaj senzor posjeduje deset pinova. Četiri pina se koriste za napajanje. To su dva GND i po dva VDD pina. Pinovi S0, S1, S2, S3, OUT, OE koriste se za upravljanje i podešavanje mjernoga člana. Ovaj senzor ima polje 8x8 u kojem se nalaze foto diode. Svaki filter posjeduje šesnaest foto dioda, a ti filteri su plavi, zeleni, crveni i bez boje. Očitavanjem ovih filtera moguće je približno odrediti RBG vrijednost boje predmeta koji se nalazi ispred senzora. Pomoću RGB vrijednosti moguće je detektirati bilo koju nijansu boje i prenijeti točno tu nijansu na računalo. Senzor TCS3200 je izrazito osjetljiv na osvjetljenje prostorije te je iz tog razloga napravljeno odgovarajuće kućište pomoću kojeg se nastoji minimalizirati utjecaj vanjskog osvjetljenja na očitanu vrijednosti boje. Na slici 3.9. prikazan je funkcijski blok dijagram TCS3200 mjernog člana.

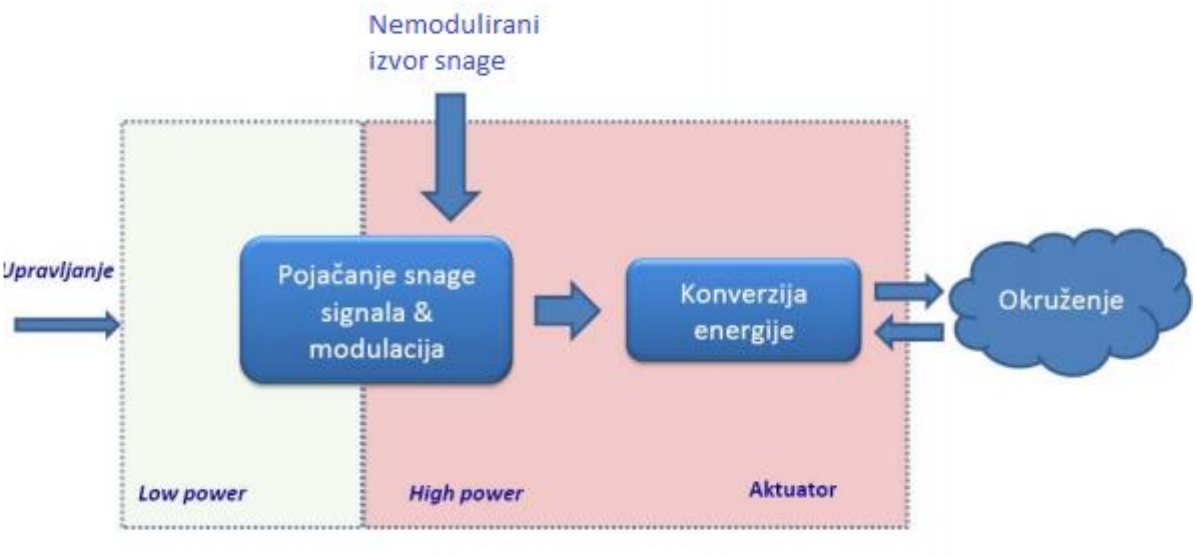


Sl. 3.9. Funkcijski blok dijagram TCS 3200 mjernog člana [11].

Sve foto diode istih boja su povezane u paralelu. S pinovima S2 i S3 odabiremo koja grupa foto dioda će biti aktivna. Dimenzije foto dioda su 110µm x 110µm.

3.3.2. Izvršni članovi

Izvršni senzori ili aktuatori su uređaji koji pretvaraju električne i fluidne ulaze u mehaničke izlaze kao što su pozicija, sila ili moment. Aktuatori kontinuirano pretvaraju električnu energiju (ili neku drugu) u mehaničku energiju. Elektromagnetski aktuatori pretvaraju energiju elektromagnetskog polja u mehaničku energiju koja generira kretanje. Elektromagnetski aktuatori su pogodni za „srednje“ pogonske momente i sile. Pneumatski aktuatori pretvaraju energiju koju daje pritisak zraka u kretanje. Pneumatski aktuatori su pogodni za manje kretnje. Hidraulični aktuatori pretvaraju energiju pritiska toka tekućine u kretanje. Pogodni su za generiranje velikih sila. Piezo električni aktuatori pretvaraju elektrostatičku energiju u pomicanje površine. Pogodni su za male kretnje i srednje veličine [12].



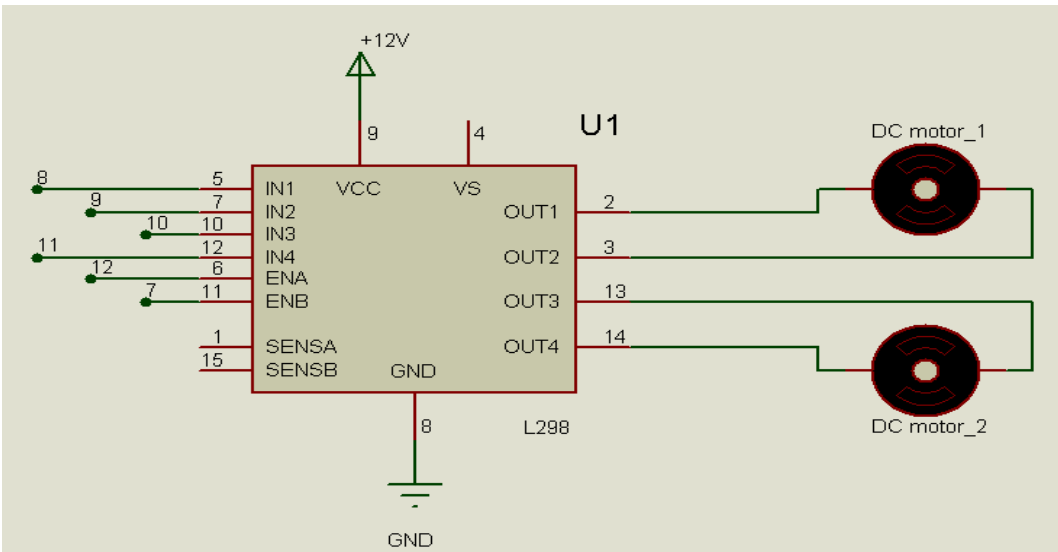
Sl. 3.10. Funkcijski dijagram aktuatora [12].

Aktuatori se dijele na elektromehaničke, aktuatore koji koriste snagu fluida, pneumatski. Primjeri elektromehaničkih aktuatora su istosmjerni i izmjenični motor, koračni motor, servo motor. Aktuatori koji koriste snagu fluida su ventili i pumpe, a pneumatski aktuatorima pripadaju regulacijski ventili i zasuni [12].

Istosmjerni motor je elektromehanički uređaj koji istosmjernu struju pretvara u rotacijsko gibanje. Klasični istosmjerni motor se sastoji od rotirajuće armature koja je oblikovana u obliku elektromagneta s dva pola i od statora kojega čine dva permanentna magneta. Krajevi namota armature spojeni su na rotacijski prekidač, komutator, koji prilikom svakog okretaja rotora dvaput mijenja smjer toka struje kroz armaturni namot stvarajući tako moment koji zakreće rotor [14].

Sl. 3.11. Funkcijski dijagram aktuatora [15].

U ovome projektu korištena su dva električna motora kao aktuatori. Korišteni su istosmjerni motori nazivnog napona 12V. Oba motora pokreću pokretne trake kao što je prikazano na slici 3.1. Motori su napajani preko H-mosta L298N, te su upravljani s mikroupravljačom. Pomoću H-mosta moguće je prilagođavati brzinu vrtnje motora, te samim time prilagođavati brzinu pokretnih traka. Na slici 3.12. prikazana je shema spajanja oba istosmjerna motora na H-most L298N, te pinovi preko kojih je H-most spojen s Arduino Mega 2560.



Sl. 3.12. Shema spajanja istosmjernih motora na H-most L298N i na Arduino.

3.3.3. Mikroupravljač

Mikroupravljač je mikroračunalo zaduženo za upravljanje i kontrolu određenih procesa. Mikroupravljači imaju vrlo rasprostranjenu primjenu u različitim uređajima i sustavima. U mikroupravljačima su integrirani mikroprocesor, digitalni i analogni ulazi i izlazi, digitalni satovi, brojači, oscilatori, memorija, komunikacijski sklopovi i drugi dodaci. Proizvode se u rasponu od malih 4 bitnih procesora s ograničenim mogućnostima pa sve do 32 bitnih procesora relativno velikih brzina. Prilikom odabira mikroupravljača moramo voditi računa da se izabere najekonomičniji mikroupravljač koji će moći zadovoljiti obavljanje željenih zadataka.

Arduino je open source platforma za kreiranje elektroničkih prototipova bazirana na sklopovlju i programskom paketu koji je fleksibilan i jednostavan za korištenje. Arduino omogućava jednostavno povezivanje elektroničkih i softverskih komponenti od kojih se mogu razviti jednostavni i složeni projekti. Srce arduina je mikroupravljač [21].

U ovome projektu korištena je mikroupravljačka pločica Arduino Mega 2560. Ovaj mikroupravljač ima široku primjenu i brojne mogućnosti. Odlikuje se jednostavnošću programiranja. Hardvare ovoga mikroupravljača sastoji se od pločice otvorenog koda. Bitna koncepcija Arduina je mogućnost povezivanja na niz drugih izmjenjivih dodatnih modula poznatih kao shields.

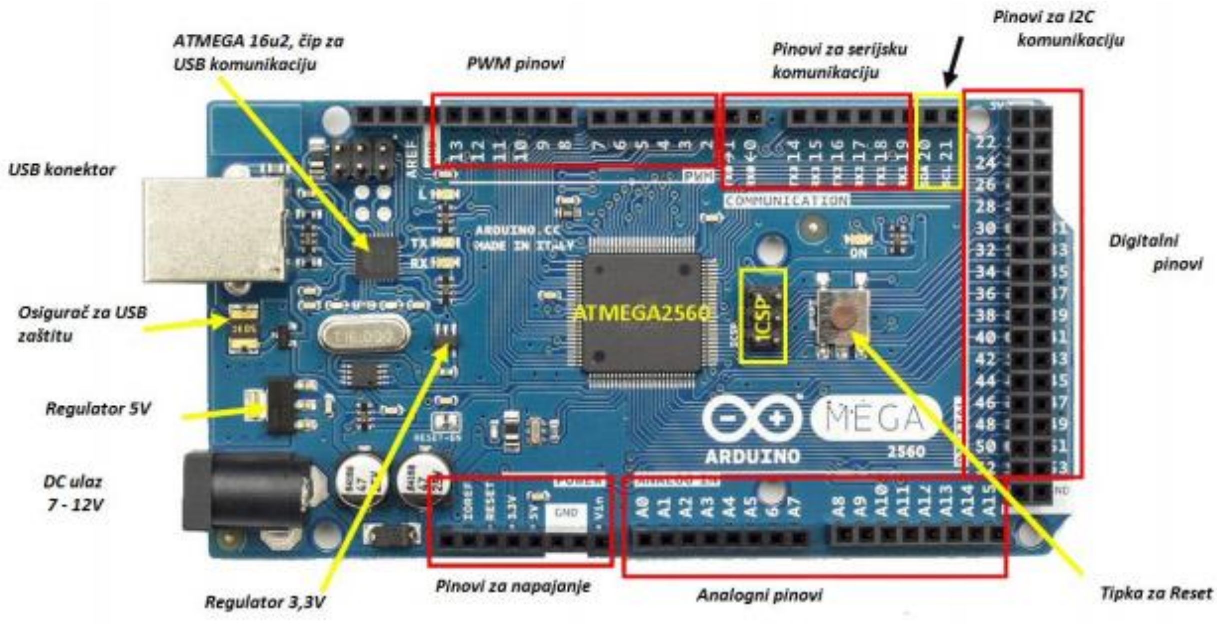
TEHNIČKE KARAKTERISTIKE	Atmega 2560
Operativni napon	5V
Ulazni napon(preporučeno vanjsko napajanje pločice)	7-12V
Ulazni napon(granični)	6-20V
Digitalni I/O pinovi	54(od čega 15 podržava PWM)
Analogni ulazni pinovi	16
DC jakost struje po I/O pinu	20mA
DC jakost struje za 3.3V pinove	50mA
Flash memorija	256kB od čega 8kB za bootloader
SRAM	8kB
EEPROM	4kB
Brzina sata	16MHz
Pin s ugrađenom led diodom	13
Duljina	101.52 mm
Širina	53.3 mm

Težina	37g

Tablica 3.1. Tehničke karakteristike Arduino Mega 2560 [17].

Mikroupravljačka pločica Arduino Mega 2560 posjeduje 54 digitalna ulaza/izlaza od kojih se 15 mogu koristiti kao pulsno širinski modulirani (PWM) izlazi, 16 analognih ulaza, 16 MHz kristal oscilator, USB priključak za komunikaciju s pripadajućom elektronikom i priključak za napajanje s pripadajućim stabilizatorom napona. Atmega 2560 sadrži 256 KB memorije od kojih se 8 KB koristi za bootloader. Upravljački program se nalazi u flash memoriji. SRAM memorija služi za spremanje varijabli tijekom rada i sadrži 8 KB. Ovaj mikroupravljač sadrži 8 KB EEPROM memorije koja je trajno pohranjena. Svaki od pinova rade na 5V i svaki može dati do maksimalno 40 mA [18].

Na slici 3.13. prikazana je mikroupravljačka pločica Arduino Mega 2560, te su prikazani svi dijelovi iste.



Sl. 3.13. Arduino Mega 2560 [18].

3.3.4. Dopunski članovi

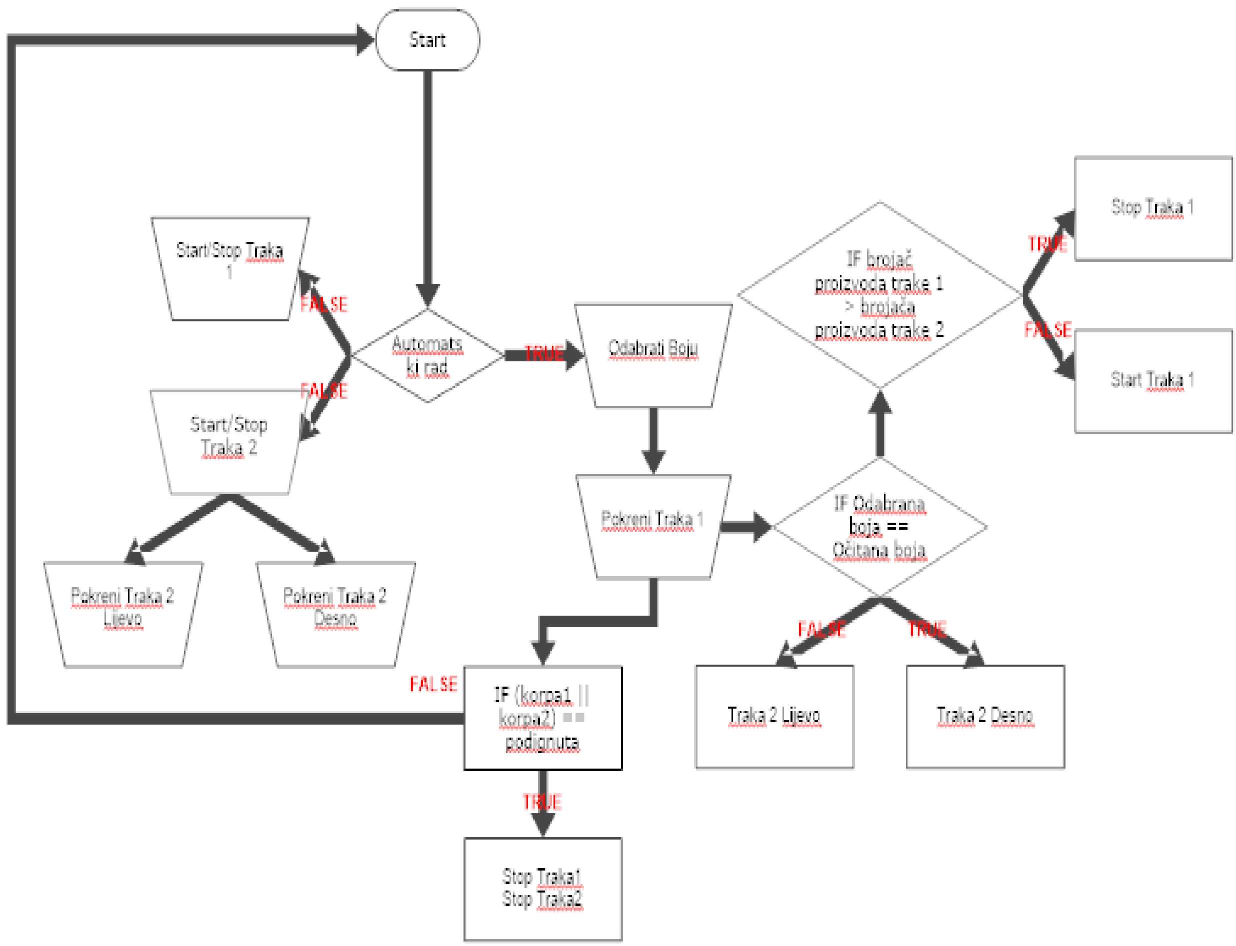
LCD 16x2 i2c je zaslon koji može prikazivati 16 znakova u dva reda. Radni napon ovoga displeja je 5V DC. Displej sadrži dva registra, Command i Data. Command registar pohranjuje isključivo naredbe koje su mu zadane npr. brisanje trenutnog sadržaja zaslona, postavljanje kursora, kontrola zaslona. Data registar pohranjuje podatke koji se moraju prikazati na LCD zaslonu. I2c pretvarač se koristi zbog jednostavnosti primjene unutar programskog koda i broja pinova potrebnih za komunikaciju. I2c pretvarač sadrži 4 ulazna pina od kojih su dva napajanje, a preostala dva podatkovna te 16 izlaznih pinova od kojih se 8 koriste za komunikaciju s LCD zaslonom [19].

Sl. 3.14. LCD 16x2 i2c [20].

4. PREDLOŽENA PROGRAMSKA RJEŠENJA

4.1. Algoritam upravljanja

U matematici , računarstvu, lingvistici i srodnim disciplinama, algoritam ili postupak je konačan slijed dobro definiranih naredbi za ostvarenje zadatka, koji će za dano početno stanje terminirati u definiranom konačnom stanju. Moderno računarstvo je nezamislivo bez primjene algoritama, njihove matematičke analize te postupcima ubrzavanja njihova izvođenja [13].



Sl. 4.1. Algoritam upravljanja.

Na slici 4.1. prikazan je algoritam upravljanja maketom za sortiranje proizvoda prema boji. Maketa mora biti spojena s računalom. Upravljanje maketom moguće je jedino s PC aplikacijom. Pri pokretanju makete, prvo je potrebno odabrati način rada makete koji se odabire u PC aplikaciji. Maketa može raditi u dva načina rada. Prvi način rada je ručni, a drugi automatski. U ručnom načinu rada, moguće je upravljati trakom pomoću PC aplikacije. Pri automatskom načinu rada, prvo je potrebno izabrati boju koju želimo sortirati. Nakon odabira boje, daljnji automatski rad se pokreće pritiskom na tipku za pokretanje prve trake. Ako je odabrana boja jednaka očitanoj boji, proizvod se šalje na drugoj traci u desnu stranu. Ako je proizvod različite boje od odabrane boje, proizvod se šalje na drugoj traci u lijevu stranu. Kada proizvod prijeđe s prve trake na drugu, prva traka se zaustavlja, te se ponovno pokreće kada proizvod napusti drugu traku. Na maketi postoje dva „spremnika“ u koje pada proizvod nakon napuštanja druge trake. Ako je bilo koji od spremnika uklonjen, dolazi automatski do zaustavljanja cijele makete, te je potrebno ponovno odabrati način rada i ponovno pokrenuti maketu.

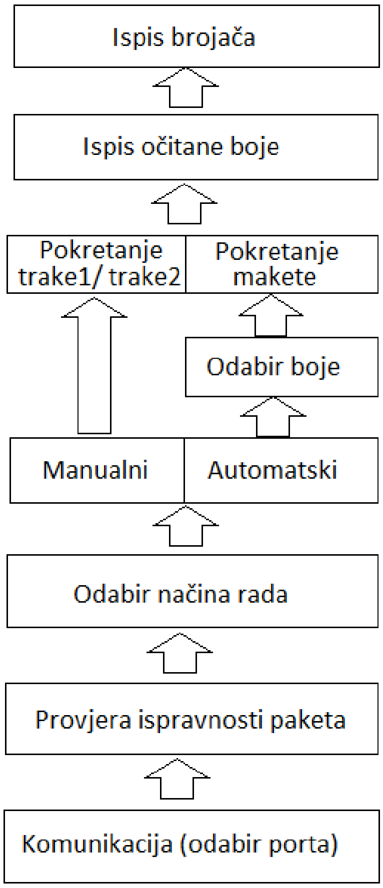
4.2. Nadzor i upravljanje

Aplikacija za nadzor i upravljanje je programska podrška koja omogućava prikupljanje podataka iz mikroupravljača i slanje upravljačkih naredbi. Dva glavna zahtjeva koja mora ispuniti svaka aplikacija za nadzor i upravljanje su zadovoljavajući rad sklopovlja i programske podrške te učinkovito i sigurno korištenje. Prikupljeni podaci se prilagođavaju te se najčešće prikazuju grafički radi bržeg i lakšeg pregleda.

Aplikacija treba omogućiti u svakom trenutku nadzor rada makete na što jednostavniji način. Potrebno je grafički prikazati promjene rada pokretnih traka. Idealno rješenje bilo bi prikazivanje promjene stanja pokretne trake korištenjem animacije, ali u ovoj aplikaciji to je riješeno promjenom boje. Nadalje, potrebno je prikazati očitanu boju sa senzora. Očitana boja ispisuje se tekstualno što je dosta pregledno. Brojači predmeta koji su napustili maketu prikazani su brojevima.

Aplikacija treba omogućiti upravljanje maketom na jednostavan način. Maketa mora sadržavati dva načina rada te je potrebno omogućiti promjene načina rada. Svaka pokretna traka mora se moći pokretati zasebno. Potrebno je napraviti dio za odabir boje koja će se selektirati.

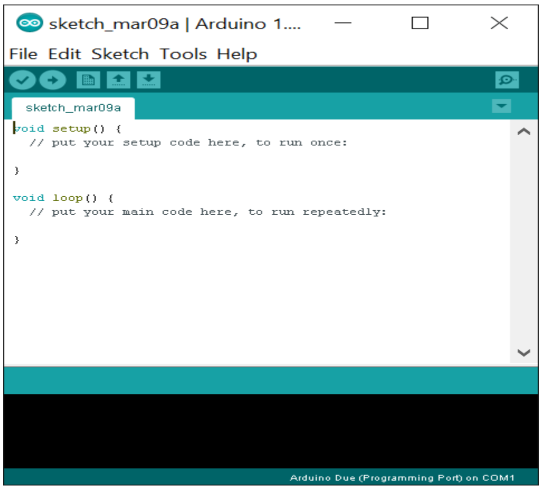
Na slici 4.2. prikazana je blok shema aplikacije za nadzor i upravljanje. Navedeni su svi dijelovi koje aplikacija za nadzor i upravljanje mora posjedovati da bi uspješno obavljala svoju funkciju. Aplikacija počinje s uspostavljanjem komunikacije, te provjerom ispravnosti paketa. Sljedeći korak je izbor načina rada makete, pa odabir boje ako se odabrao automatski način rada. Slijedi pokretanje trake u jednom od načina rada. Potrebno je prikazati rezultate rada makete, a oni se prikazuju ispisom očitane boje i brojača.



Sl. 4.2. Blok shema aplikacije za nadzor i upravljanje.

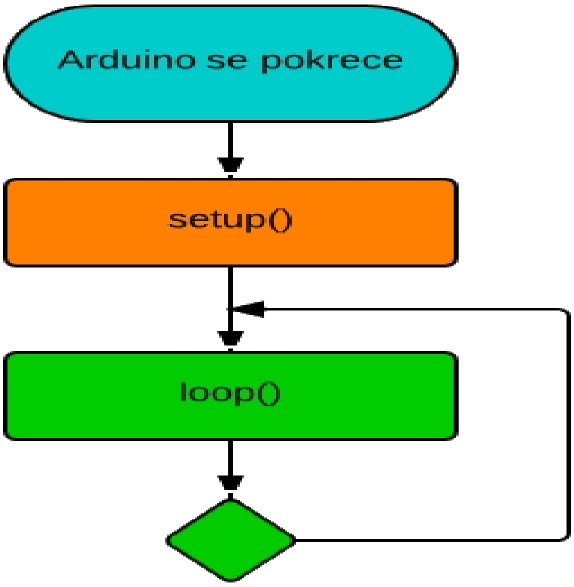
4.3. Implementacija upravljačkog algoritma u mikroupravljač

Za programiranje Arduino pločica potrebno je instalirati Arduino IDE softverski paket. Softver Arduino IDE je open source tipa i radi na različitim platformama (Windows, Mac, Linux)[21].



Sl. 4.3. Softverski paket Arduino IDE [22].

Programiranje Arduina obavlja se u prilagođenoj verziji C++ programskog jezika. Arduino program sastoji se od dva osnovna dijela koja su setup() i loop(). U setup() dijelu obavlja se deklariranje ulaza i izlaza, komunikacije s računalom i slični inicijalizacijski postupci. Loop() dio koda se stalno ponavlja i u njemu se definira korisnička aplikacija zajedno s prekidnim rutinama [23].



Sl. 4.4. Algoritam izvršavanja Arduino programa [23].

Istosmjerni motori korišteni u maketi upravljani su preko pinova koji omogućuju generiranje PWM signala. Za svaki motor napisane su funkcije za pokretanje u oba smjera i funkcija za zaustavljanje. Svaka funkcija sadrži PWM vrijednosti koja utječe na brzinu vrtnje elektromotora. Faktor popunjenosti signala pokazuje koliki dio vremena je PWM signal u stanju visokog izlaznog napona. Faktor popunjenosti se izražava s vrijednosti od 0 do 255. Vrijednost 0 odgovara 0%, a vrijednost 255 odgovara 100% popunjenosti signala. Za pokretanje prve trake koristi se vrijednost 100 koja odgovara 39% popunjenosti signala. Druga traka koristi vrijednost 75 koja odgovara 29% popunjenosti signala.

Izvorni programski kod funkcija za pokretanje i zaustavljanje traka.

```
// stanja motora1
 void motor1_naprijed()
 {
        analogWrite( speedPin1, 100);
        digitalWrite(mot1Pin1, HIGH);
        digitalWrite(mot1Pin2, LOW);
```

```
        kretanje_motor1 = '1';
  }
   void motor1_stop()
   {
        analogWrite( speedPin1, 255);
        digitalWrite(mot1Pin1, LOW);
        digitalWrite(mot1Pin2, LOW);
        kretanje_motor1 = '0';
  }
  void motor1_usporen()
   {
        analogWrite( speedPin1, 0);
        digitalWrite(mot1Pin1, HIGH);
        digitalWrite(mot1Pin2, LOW);
        kretanje_motor1 = '1';
  }
// stanja motora2
 void motor2_lijevo()
 {
        analogWrite( speedPin2, 75);
        digitalWrite(mot2Pin1, HIGH);
        digitalWrite(mot2Pin2, LOW);
        kretanje_motor2 = '1';
        kretanje_motor2_smjer = 'L';
  }
  void motor2_desno()
 {
        analogWrite( speedPin2, 75);
        digitalWrite(mot2Pin1, LOW);
        digitalWrite(mot2Pin2, HIGH);
        kretanje_motor2 = '1';
        kretanje_motor2_smjer = 'D';
  }
   void motor2_stop()
   {
        analogWrite( speedPin2, 0);
        digitalWrite(mot2Pin1, LOW);
```

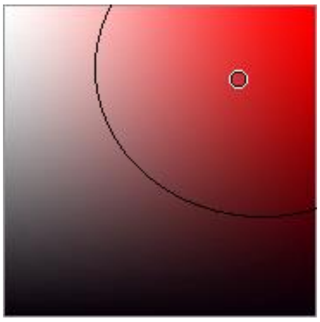
```
        digitalWrite(mot2Pin2, LOW);
        kretanje_motor2 = '0';
        kretanje_motor2_smjer = 'N';
  }
```

Senzor za očitavanje boje proizvoda prosljeđuje mikroupravljaču frekvenciju uzorka ovisno o boji proizvoda koji se nalazi ispred senzora. Preko vrijednosti frekvencije može se izračunati postotak tri osnovne boje (crvene, zelene, plave), te postotak zapisati kao RGB vrijednost.

Izvorni programski kod za izračun postotka pojedinačne i RGB vrijednosti boje.

```
PercentageRed=int((FrequencyRed/FrequencyClear)*100.0); // izračunavanje postotka boje preko frekvencije
PercentageGreen=int((FrequencyGreen/FrequencyClear)*100.0
PercentageBlue=int((FrequencyBlue/FrequencyClear)*100.0);

  float rgb_Red = (255 * PercentageRed)/100; // izračunavanje RGB vrijednosti boje
  float rgb_Green = (255 * PercentageGreen)/100;
  float rgb_Blue = (255 * PercentageBlue)/100;
```

Zbog nepreciznosti senzora za prepoznavanje boje, potrebno je definirati područje u kojem će biti prepoznata boja. Ideja je da se definiraju tri područja. Definirana područja su za crvenu, plavu i zelenu boju. Svako područje tolerira razne nijanse tih boja. S ovime se postiglo da se mogu sortirati tri osnovne boje, a da se sve ostale boje detektiraju kao da nema boje. Za svaku boju određuje se granično područje na kojem će biti prepoznata boja. Slika 4.5. prikazuje graničnu vrijednost boje na kojoj će senzor prepoznati predmet kao crveni.



Sl. 4.5. Granično područje na kojem senzor prepoznaje crvenu boju.

Točka označena na slici 4.5. predstavlja središte crvene boje. Ako je očitana vrijednost sa senzora za prepoznavanje boje udaljena za manje od 85 od centra, program će potvrditi da je očitana crvena boja. Udaljenost između centra boje i očitane vrijednosti računa se kao euklidska udaljenost.

Izvorni programski kod za izračun udaljenosti očitane vrijednosti i centra boje.

```
float udaljenost_boje1 = sqrt(((200-rgb_Red)*(200-rgb_Red))+((50-rgb_Green)*(50-
rgb_Green))+((50-rgb_Blue)*(50-rgb_Blue)));    //udaljenosti od mjerene točke do centra boje
```

Sa senzorom za prepoznavanje boje izvršena su mjerenja na predmetima različitih veličina i boje. Rezultati mjerenja prikazani su u Prilogu 3. Tablica se sastoji od četiri kolone. Prva kolona predstavlja naziv predmeta i boju predmeta. Druga kolona predstavlja sliku predmeta. Treća kolona predstavlja očitanu RGB vrijednost mjerenog predmeta. Četvrta kolona predstavlja zbroj očitanih osnovnih boja (crvena, zelena, plava). Senzor je uspio prepoznati crvenu, zelenu i plavu boju dok je kod ostalih u određenoj mjeri pogriješio. Senzor ima velika odstupanja zbog utjecaja okoline, pa su mjerenja različita u osvijetljenim i zatamnjenim prostorima. Postavljanjem tunela ispred senzora postigla se smanjena osjetljivost na osvjetljenje prostora, te su se dobili ujednačeniji rezultati. Prema mjernim uzorcima i rezultatima mjerenja može se zaključiti da je senzor pogodan za sortiranje proizvoda prema boji na samo nekoliko osnovnih boja.

Na prvoj traci nalazi se infracrveni senzor prije senzora za očitavanje boje. Taj senzor služi za usporavanje trake radi preciznijeg očitavanja boje. Traka se usporava na 600ms pomoću brojača vremena.

Izvorni programski kod za usporavanje prve trake.

```
if(senzor0_status == LOW){
    senzor0reg = 1;
     }
      if(senzor0reg == 1){
          currentMillis = millis();
                    if(currentMillis - previousMillis <= interval){
                              analogWrite( speedPin1, 7);
                              digitalWrite(mot1Pin1, HIGH);
                              digitalWrite(mot1Pin2, LOW);
                              kretanje_motor1 = '1';
                               }
```

```
            else{
                        senzor0reg = 0;
                         previousMillis = currentMillis;
                  }
                        }
```

Na drugoj traci se u isto vrijeme ne smiju naći dva predmeta jer bi time sortiranje proizvoda bilo znatno otežano odnosno nemoguće uz predloženi dizajn makete. Uspoređuje se brojač s prve trake i dva brojača s druge trake. Sve dok ne izađe proizvod s druge trake prva trake je zaustavljena.

Izvorni programski kod za usporedbu brojača i zaustavljanje prve trake.

```
//kontrola prve i druge trake
      if(count1 > (count2 + count3)){
            analogWrite( speedPin1, 0);
            digitalWrite(mot1Pin1, LOW);
            digitalWrite(mot1Pin2, LOW);
            kretanje_motor1 = '0';
      }
```

Ako se dogodi da je jedan od spremnika uklonjen, zaustavljaju se obje trake. Izvođenje programa se može nastaviti tek kad se spremnici vrate u odgovarajući položaj.

Izvorni programski kod za zaustavljanje obadvije trake ukoliko je jedan od spremnika uklonjen.

```
if(korpe_status == HIGH){
       motor1_stop();
       motor2_stop();
      }
```

Mikroupravljačka pločica šalje putem serijske veze sve potrebne informacije PC aplikaciji za nadzor. Aplikaciji se isporučuje paket u kojem se nalazi informacija o prvoj pokretnoj traci, drugoj pokretnoj traci, smjeru druge pokretne trake, očitanoj boji i brojačima. Na slici 4.6. prikazan je paket koji prima PC aplikacija s mikroupravljača.

Izvorni programski kod za serijsku komunikaciju sa PC aplikacijom.

// komunikacija sa C#
 Serial.print("["), Serial.print(kretanje_motor1), Serial.print(kretanje_motor2), Serial.print(kretanje_motor2_smjer), Serial.print(boja),Serial.print("]"), Serial.print(count2), Serial.print("|"), Serial.println(count3);

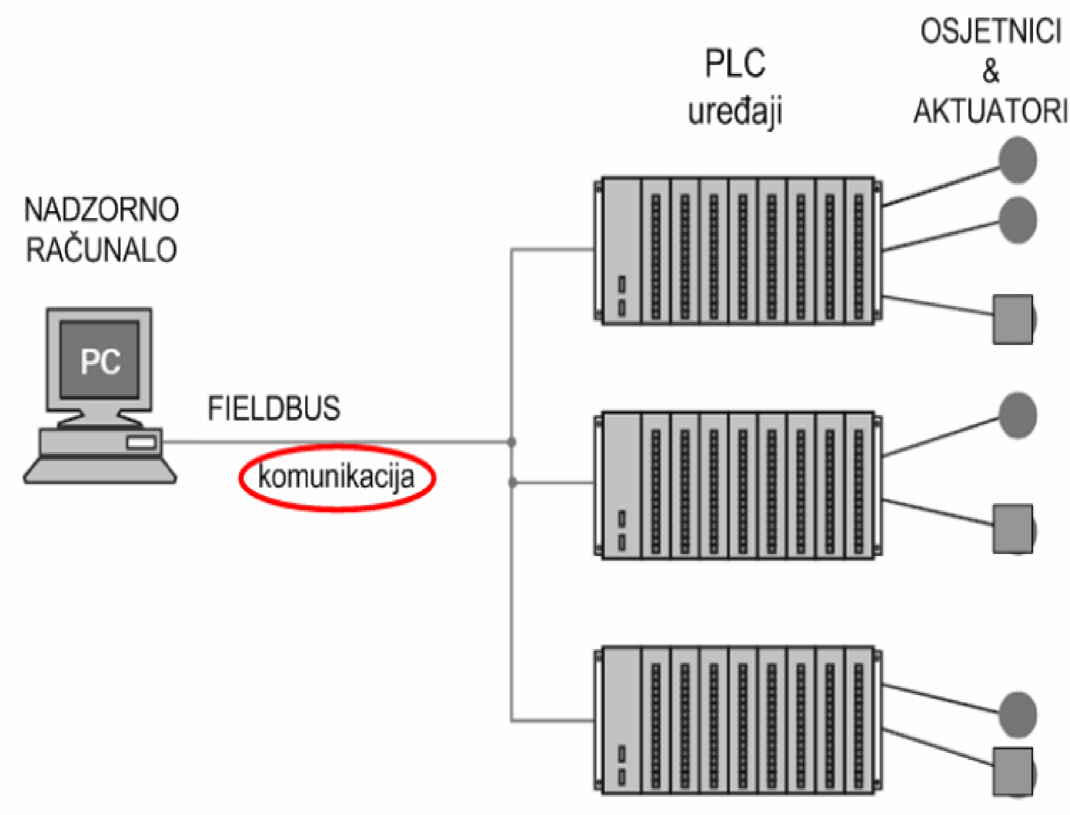
Na slici 4.6. prikazan je paket koji prima PC aplikacija s mikroupravljača. Prva uglata zagrada predstavlja početak paketa. Na drugoj poziciji je vrijednost („0“ prikazuje da traka miruje, a „1“ prikazuje da je traka pokrenuta) koja prikazuje dali je prva traka pokrenuta. Na trećoj poziciji je vrijednost („0“ prikazuje da traka miruje, a „1“ prikazuje da je traka pokrenuta) koja pokazuje dali je druga traka pokrenuta. Četvrta vrijednost prikazuje smjer kretanja druge trake („N“ prikazuje neutralni smjer, „L“ prikazuje lijevi smjer, „D“ prikazuje desni smjer). Peta vrijednost služi za slanje informacije o očitanoj boji („0“ prikazuje da nema boje, „1“ prikazuje da je plava boja, „4“ prikazuje da je zelena boja, „5“ prikazuje da je crvena boja). Druga uglata zagrada predstavlja završetak paketa. Posljednje dvije vrijednosti prikazuju brojače koji su odvojeni s uspravnom crtom.

[	1	0	N	0	]	2	|	0

Sl. 4.6. Poslani paket s mikroupravljača na PC aplikaciju.

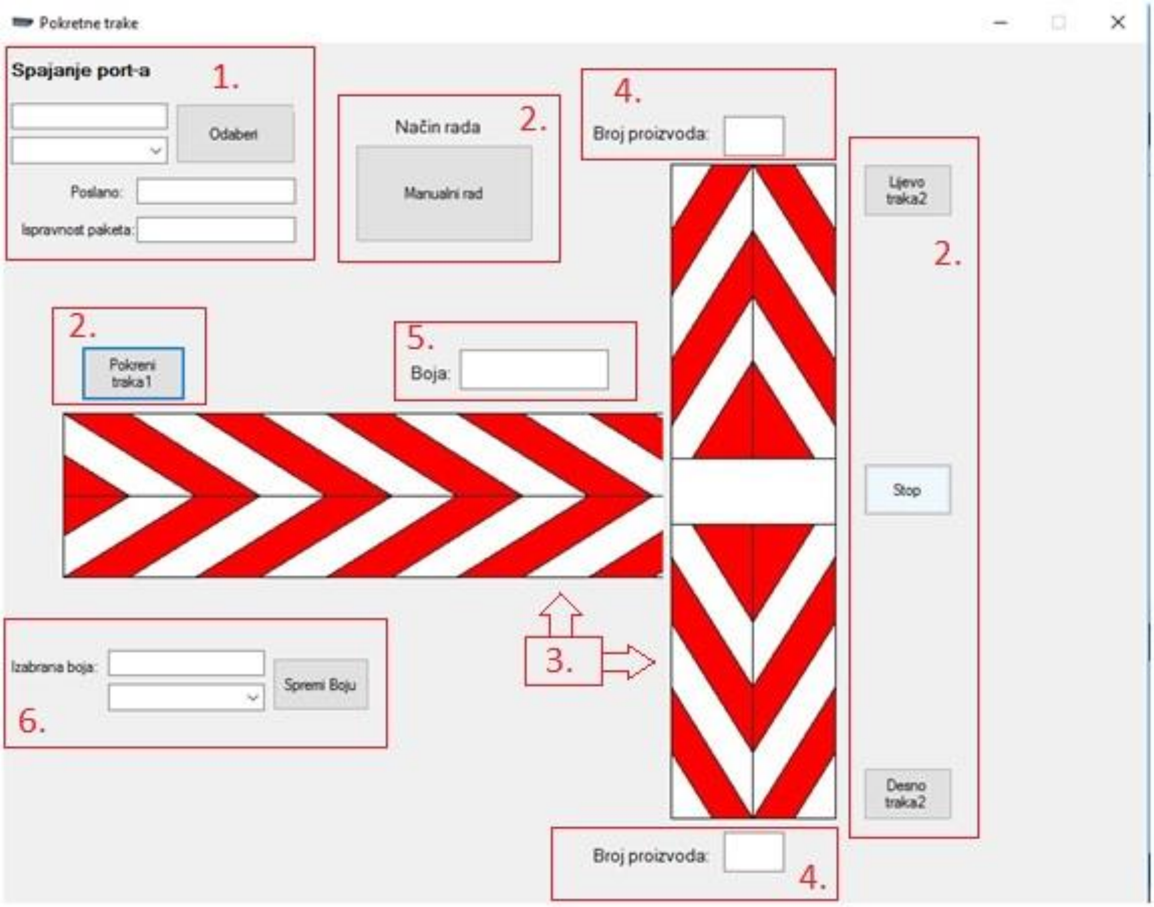
4.4. PC aplikacija za nadzor i upravljanje

PC aplikacija za nadzor i upravljanje maketom za sortiranje proizvoda prema boji može se smatrati jednostavnim SCADA (eng. Supervisory Control And Data Acquisition) sustavom. SCADA je tehnologija koja omogućuje prikupljanje podataka iz jednog ili više udaljenih postrojenja te slanje upravljačkih naredbi u ta postrojenja [26]. Na slici 4.7. prikazana je shema SCADA sustava u kojoj se nadzor obavlja preko računala, te se upravljanje obavlja preko PLC uređaja koji utječe na aktuatore i prima podatke sa senzora. Shema prikazuje sličnu situaciju kao i u samoj maketi sustava za sortiranje proizvoda prema boji.



Sl. 4.7. Shema suvremenih SCADA sustava [26].

Aplikaciju je potrebno izraditi u C# objektno orijentiranom programskom jeziku. Potrebno je na jednostavan i vizualno prihvatljiv način prikazati očitanu boju sa senzora, stanja pokretnih traka, brojače proizvoda, te ostale podatke s makete. U aplikaciji mora se omogućiti ručni i automatski rad makete. Za izradu PC aplikacije za nadzor i upravljanje odabran je C# programski jezik zbog jednostavnosti. Programski jezik C# zadovoljava svim zahtjevima koji su potrebni za izradu PC aplikacije za maketu sustava za sortiranje proizvoda prema boji.



Sl. 4.8. PC aplikacija makete sustava za sortiranje proizvoda prema boji.

Izrada PC aplikacije može se podijeliti u 6 dijelova, te su dijelovi prikazani brojevima na slici 4.8.

1. Odabir PORT-a i provjere ispravnosti paketa.

Za odabir PORT-a koristi se combobox koji se potvrđuje s tipkalom. Budući da računalo ima više različitih portova bilo je potrebno izraditi mogućnost komunikacije s bilo kojim portom. Na textbox-u se ispisuje „connect“ ili „disconnect“ ovisno dali je odabran korišteni port. U textbox-u pod nazivom „Poslano“ ispisuje se niz podataka poslanih s Arduina. U PC aplikaciji definiran je izgled niza, te se on provjerava. Ako poslani niz odgovara definiranom na textbox-u pod nazivom „Ispravnost paketa“ ispisuje se tekst „Ispavan“.

U sljedećem dijelu prikazan je dio koda za spajanje aplikacije za nadzor i upravljanje s mikroupravljačom.

Izvorni programski kod za spajanje aplikacije za nadzor i upravljanje s mikroupravljačom.

```
using System.IO.Ports;
```

```
namespace pokretne_trake
{
    public partial class Form1 : Form
    {
        string primljeno;
        private class Item
        {
            // definiranje da bi mogli popunjavati Comboboxeve
            public string Name;
            public int Value;
            public Item(string name, int value)
            {
                Name = name; Value = value;
            }
            public override string ToString()
            {
                return Name;
            }
        }
        public Form1()
        {
            InitializeComponent();
            combo_serial.Items.Add(new Item("COM1", 1));
            combo_serial.Items.Add(new Item("COM2", 2));
            combo_serial.Items.Add(new Item("COM3", 3));
            combo_serial.Items.Add(new Item("COM4", 4));
            combo_serial.Items.Add(new Item("COM5", 5));
            combo_serial.Items.Add(new Item("COM6", 6));
            combo_serial.Items.Add(new Item("COM7", 7));
            combo_serial.Items.Add(new Item("COM8", 8));
            combo_serial.Items.Add(new Item("COM9", 9));
            combo_serial.Items.Add(new Item("COM10", 10));
            combo_serial.Items.Add(new Item("COM11", 11));
            combo_serial.Items.Add(new Item("COM12", 12));
            combo_serial.Items.Add(new Item("COM13", 13));
            combo_serial.Items.Add(new Item("COM14", 14));
            combo_serial.Items.Add(new Item("COM15", 15));
```

```
}
```

Poslije odabira „port-a“ vrši se provjera dali je komunikacija uspješno uspostavljena. Sljedeći dio koda prikazuje provjeru uspješnosti komunikacije.

Izvorni programski kod za provjeru uspješnosti komunikacije.

```
try
        {
            if (!serialPort1.IsOpen)
            {
                serialPort1.DataReceived += serialPort1_DataReceived;
                serialPort1.Open();
                connect_textBox.Text = "  Connect!";
                connect_textBox.ForeColor = Color.Green;
            }
        }
        catch (Exception ex)
        {
            connect_textBox.Text = "  Disconnect!";
        }
    }
```

2. Dio za upravljanje maketom

S dijelom sustava za upravljanje moguće je odrediti način rada makete. Maketa može raditi u automatskom i ručnom radu. Ako se odabere ručni rad, moguće je upravljati s bilo kojoj od traka pomoću tipkala s aplikacije. Ako se odabere automatski način rada, maketa izvršava automatsko sortiranje proizvoda prema boji.

Sadržava tipkalo koje određuje način rada makete. Način rada može biti manualni i automatski. Ako je odabran manualni rad trake se mogu pokretati preko tipkala (Pokreni traka 1, Lijevo traka2, Desno traka2) i zaustavljati drugu traku s tipkalom „Stop“. Automatski način rada pokreće se s tipkalom „Pokreni traka1“.

U sljedećem dijelu prikazan je dio koda tipkala za odabir načina rada

Izvorni programski kod za odabir načina rada pokretne trake.

```
private void Manualni_rad_Click(object sender, EventArgs e)
    {
        serialPort1.Write("5");
        if (Manualni_rad.Text == "Automatski rad")
        {
            Manualni_rad.Text = "Manualni rad";
        }
        else
        {
            Manualni_rad.Text = "Automatski rad";
        }
    }
```

3. Prikaz aktivnosti pokretnih traka

Stanja pokretnih traka prikazana su s dva pictureBox-a. Kada se pokretna traka kreće postavlja se zelena slika u picturebox, a ako je pokretna traka zaustavljena onda se postavlja crvena slika.

Izvorni programski kod za prikaz aktivnosti pokretnih traka.

```
if (slovo_3.ToString() == neutralno.ToString())
        {
            pictureBox_traka2.Image = Properties.Resources.motor2_stop;
        }
        else if (slovo_3.ToString() == lijevo.ToString())
        {
            pictureBox_traka2.Image = Properties.Resources.motor2_lijevo; // ide lijevo
        }
        else if (slovo_3.ToString() == desno.ToString())
        {
            pictureBox_traka2.Image = Properties.Resources.motor2_desno; // ide desno
        }
```

4. Prikaz brojača

Brojač prikazuje broj proizvoda koji je izašao s druge trake. Korištena su dva textbox-a. Svaka strana ima svoj brojač. Brojači se poništavaju s tipkalom za promjenu načina rada makete.

Izvorni programski kod za ispis brojača.

```
br_pre_lijevo.Text = broj_paketa_lijevi.ToString();
br_pre_desno.Text = broj_paketa_desni.ToString();
```

5. Prikaz ispisa boje

Prikazuje trenutno očitanje senzora za prepoznavanje boje. Boja koja je očitana ispisana je u textbox-u. Sadržaj textbox-a je u istoj boji kao i očitana boja.

Izvorni programski kod za prikaz očitane boje.

```
else if (slovo_4.ToString() == boja_plava.ToString())
            {
                boja.Text = "Plava";
                boja.ForeColor = Color.Blue;
            }
            else if (slovo_4.ToString() == boja_crvena.ToString())
            {
                boja.Text = "Crvena";
                boja.ForeColor = Color.Red;
            }
            else if (slovo_4.ToString() == boja_zelena.ToString())
            {
                boja.Text = "Zelena";
                boja.ForeColor = Color.Green;
            }
```

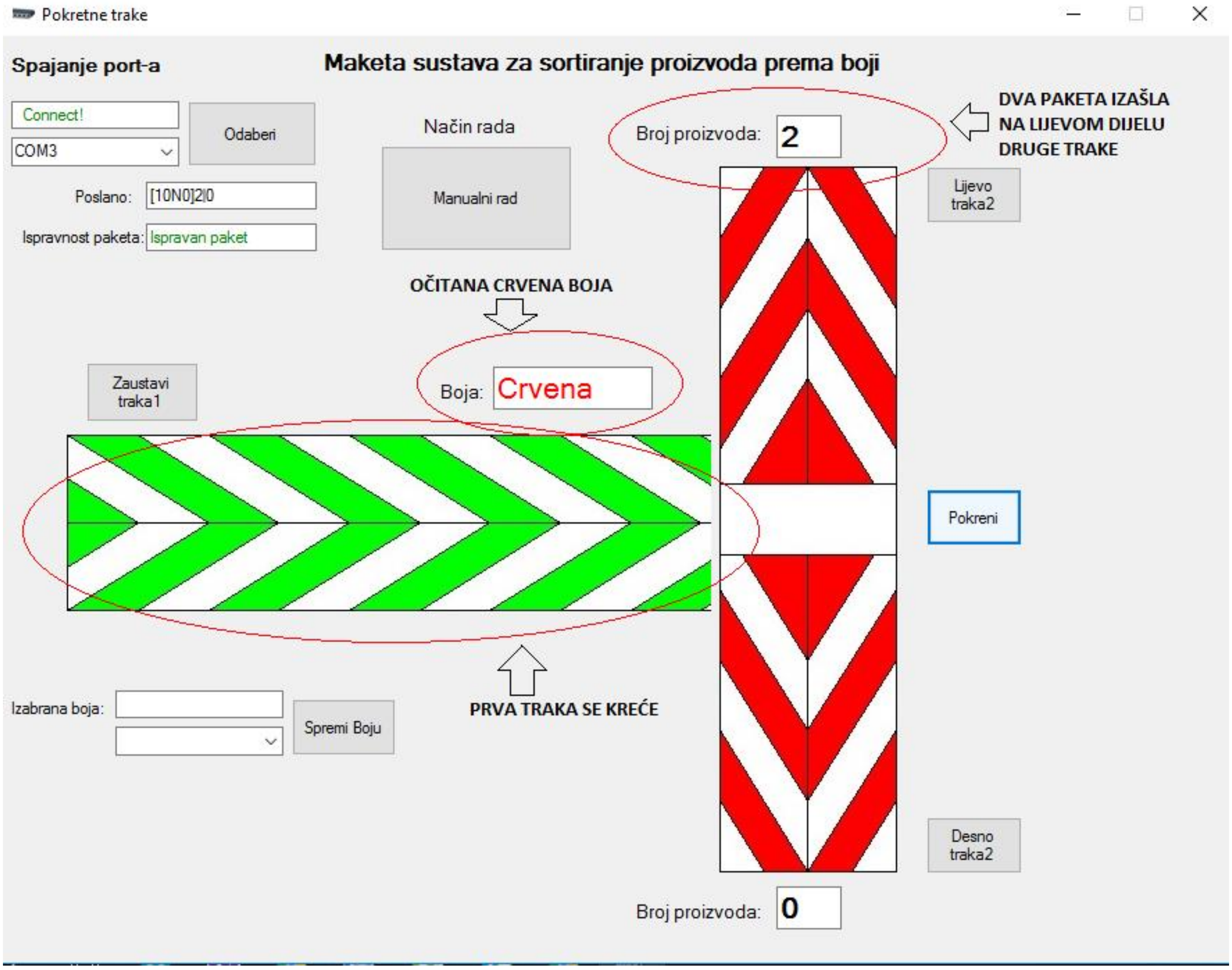
6. Izbor boje

Služi za odabir boje koja će biti selektirana u automatskom načinu rada. Boja se odabire u combobox-u, potvrđuje se s tipkalom „Spremi Boju“, te se izabrana vrijednost ispisuje u textbox-u pod nazivom „Izabrana boja“.

Izvorni programski kod za izbor boje.

```
private void spremi_boja_Click(object sender, EventArgs e)
    {
        Izabrana_boja.Text = comboBoja.SelectedItem.ToString();

        string plava = "PLAVA";
        string crvena = "CRVENA";
        string zelena = "ZELENA";

        if (Izabrana_boja.Text == plava.ToString())
        {
            serialPort1.Write("9");
            MessageBox.Show("Izabrana boja PLAVA");
        }
        else if (Izabrana_boja.Text == crvena.ToString())
        {
            serialPort1.Write("3");
            MessageBox.Show("Izabrana boja CRVENA");
        }
        else if (Izabrana_boja.Text == zelena.ToString())
        {
            serialPort1.Write("2");
            MessageBox.Show("Izabrana boja ZELENA");
        }
    }
```

Slika 4.9. prikazuje primjer PC aplikacije za vrijeme izvođenja programa. Na slici se vidi da je aplikacija uspješno povezana s mikroupravljačom i da je paket ispravan. Odabrani način rada je manualni. Pokrenuta je prva traka što se može zaključiti po zelenoj slici prve trake. Druga traka miruje što se vidi prema crvenoj slici. Senzor je očitao crvenu boju. Lijevi brojač pokazuje da su dva proizvoda napustila maketu.



Sl. 4.9. PC aplikacija s primjerom u kojem se prva traka kreće, senzor očitava crvenu boju, te brojilo očitava dva paketa u lijevoj strani druge trake.

5. ZAKLJUČAK

Pokretne trake imaju veliku ulogu na mnogim mjestima, pa su se tako prve koristile u dalekoj prošlosti, te se koriste i dan danas. S vremenom pokretne trake su se usavršavale i dobivale nove radne zadaće. Najčešća im je primjena u industriji, za transport materijala ili proizvoda. Uvođenjem pokretnih traka u industriju, ubrzala se proizvodnja i povećala produktivnost. Ovaj projekt zamišljen je kao simulacija pokretne trake u industriji. Za izradu projekta bilo je potrebno prvo odrediti zahtjeve i funkciju makete, pa odrediti elemente i programske jezike. Prvi korak je bio testiranje svih pojedinačnih elemenata, te upoznavanje s njihovim karakteristikama. Nakon toga bilo je potrebno izraditi mehanički dio makete, i nakon toga programirati cijelu maketu. Prilikom izrade makete pojavili su se pojedini problemi, a najviše sa senzorom za raspoznavanje boje. Senzor ima veliki utjecaj okoline. Cijela konstrukcija makete sustava za sortiranje proizvoda prema boji je ručni rad. Osmišljena je i izrađena u kućnoj radionici korištenjem raznih materijala kao što su drvo, metal (aluminij, čelični lim), plastika. Mehaničke komponente koje su korištene su izvučene iz starih uređaja. Elektroničke komponente su pretežito naručivane s interneta. Od elektroničkih dijelova korištena je Arduino Mega 2560 mikroupravljačka pločica, H-most, infracrveni senzor, mjerni član za prepoznavanje boje, displej. Svi ovi dijelovi su poprilično dostupni po niskoj cijeni. Kontrola cijele makete vrši se pomoću PC aplikacije izrađene u C# platformi. C# je programska platforma nastala na temelju objektnih jezika Java, C++ i Visual Basic. Poprilično je jednostavan, siguran i moderan objektno orijentirani jezik s visokim performansama. U ovome radu nam omogućava upravljanje i nadzor nad maketom. Komunikacija između aplikacije za nadzor i upravljanje i mikroupravljača integrirana je serijskom USB vezom.

6. LITERATURA

[1]Pokretna traka, *https://hr.wikipedia.org/wiki/Pokretna_traka* , (Pristupljeno 7.11.2016)

[2]Autoindustrija, *https://www.reddit.com/r/news/comments/48afew/mercedesbenz_swaps_robots_for_people_on_its/*, (Pristupljeno 7.11.2016)

[3]Trakasti transporteri, *http://nastava.sf.bg.ac.rs/pluginfile.php/8450/mod_resource/content/0/Predavanja_kontinualna_sredstva/TrakastiTransporter.pdf*, (Pristupljeno 7.11.2016)

[4]Belt Conveyor, *http://www.belttechnologies.com/products/metal-belt-conveyor-systems/*, (Pristupljeno 7.11.2016)

[5]Umprom, *http://www.umprom.com/transportne-trake.php?lang=sr*, (Pristupljeno 7.11.2016)

[6] Kolomejec d.o.o, *http://www.kolomejec.hr/index.php?option=com_content&task=view&id=3&Itemid=9*, (Pristupljeno 7.11.2016)

[7]Senzori, *https://hr.wikipedia.org/wiki/Senzori*, (Pristupljeno 7.11.2016)

[8]Senzori i mjerni pretvarači, *http://www.etf.ucg.ac.me/materijal/1381258432senzori2.pdf*, (Pristupljeno 7.11.2016)

[9]IR-sensor, *http://robu.in/product-category/sensors/ir-sensor/*, (Pristupljeno 7.11.2016)

[10]Color sensor, *http://www.dx.com/p/gy-31-tcs230-tcs3200-color-sensor-recognition-module-blue-black-216448#.V2AURCiLTIU*, (Pristupljeno 7.11.2016)

[11] Programmable color light to frequency converter, *http://www.mouser.com/catalog/specsheets/TCS3200-E11.pdf*, (Pristupljeno 7.11.2016)

[12]Aktuatori, *http://www.etf.ucg.ac.me/materijal/1381919848aktuatori2013.pdf*, (Pristupljeno 7.11.2016)

[13] Algoritam, *https://hr.wikipedia.org/wiki/Algoritam*, (Pristupljeno 7.11.2016)

[14] Električni strojevi, *https://hr.wikipedia.org/wiki/Elektri%C4%8Dni_strojevi* , (Pristupljeno 7.11.2016)

[15] Električni motor, *https://www.motiondynamics.com.au/unite-my8216-200w-12v-or-24v-dc-motor-3200-rpm.html*, (Pristupljeno 7.11.2016)

[16] Funkcionalna bespilotna brodica s vlastitim propulzijskim sustavom i upravljanjem, *https://bib.irb.hr/datoteka/776579.diplomski_Zavrna_verzija_DinoCarFESB.pdf*, (Pristupljeno 7.11.2016)

[17] Arduino Mega 2560, *https://www.arduino.cc//en//Main/ArduinoBoardMega2560*, (Pristupljeno 7.11.2016)

[18] Sustav kontroliranog natapanja poljoprivrednog zemljišta, *http://nastava.tvz.hr/kirt/wp-content/uploads/sites/4/2015/07/Sustav-kontroliranog-natapanja-poljoprivrednog-zemlji%C5%A1ta.pdf*, (Pristupljeno 7.11.2016)

[19] Arduino električna brava, *https://repozitorij.unin.hr/islandora/object/unin%3A700/datastream/PDF/view*, (Pristupljeno 7.11.2016)

[20] Ebay- displey 16x2, *http://www.ebay.com/itm/Arduino-IIC-I2C-TWI-162-1602-16X2-Serial-Blue-LCD-Module-Display-Screen-/190573003243*, (Pristupljeno 7.11.2016)

[21] Arduino kroz jednostavne primjene, *http://www.hztk.hr/media/Automatika/DIO1.pdf*, (Pristupljeno 7.11.2016)

[22] Arduino IDE, *https://blog.arduino.cc/2016/03/09/arduino-ide-1-6-8-available-for-download/*, (Pristupljeno 7.11.2016)

[23] Uvod u programiranje arduino mikrokontrolera, *http://012lab.rs/uvod-u-programiranje-mikrokontrolera-arduino*, (Pristupljeno 7.11.2016)

[24] Uvod u C#, *http://www.etfos.unios.hr/~lukic/oop/Auditorne_vje%C5%BEbe_5.pdf*, (Pristupljeno 7.11.2016)

[25] Simple Calculator Using C#, *http://www.sourcecodester.com/c/8567/simple-calculator-using-visual-c.html*, (Pristupljeno 7.11.2016)

[26] SCADA sustavi, *https://www.fer.unizg.hr/_download/repository/Predavanje_3w.pdf* , (Pristupljeno 7.11.2016)

SAŽETAK

U ovome radu teorijski i praktično predstavljene su pokretne trake, njihova povijest i primjena. Nadalje, detaljno je predstavljen glavni zadatak rada – izrada makete koja omogućava sortiranje proizvoda prema boji. Izrada makete sustava za sortiranje proizvoda prema boji može se podijeliti u dva dijela. U prvom dijelu detaljno je objašnjen postupak izrade makete, postavljanje elektronike te ožičavanje. U drugom dijelu opisano je projektiranje upravljačkog sustava, te sama implementacija u mikroupravljač i izrada prateće PC aplikacije za nadzor i upravljanje. Na kraju rada predstavljen je izgled izrađene PC aplikacije za nadzor i upravljanje, te rezultati sortiranja nekoliko različitih predmeta s obzirom na boju.

KLJUČNE RIJEČI

Pokretna traka, mikroupravljač, senzor, aktuator, nadzor i upravljanje.

ABSTRACT

In this master thesis are theoretically and practically represented conveyor belt, their history and application. Furthermore in detail is represented the main objective of the thesis-building a model for sorting products by color. Creating of that model can be split into two parts. First part is detail explanation of procedure on witch model is built, setting up of electronics and wiring. Second part describes design of control system and implementation in micro-controller and development of supporting PC software for monitoring and management. Software and a results of several experiments are presented at the end of the thesis.

KEYWORDS

Conveyor belt, microcontroller, sensor, actuator, monitoring and control.

ŽIVOTOPIS

Zvonimir Balent rođen je 27. rujna 1991 godine u Našicama. Odrastao je u Donjem Miholjcu, gdje je pohađao Osnovnu školu Augusta Harambašića. Nakon završene osnovne škole upisuje Elektrotehničku i prometnu školu u Osijeku, smjer Tehničar za mehatroniku. Poslije srednje škole upisuje Elektrotehnički fakultet u Osijeku, Stručni studij elektrotehnike, smjer Automatika, te isti završava 9. Listopada 2013. godine. Nakon završenog stručnog studija, upisuje Sveučilišni diplomski studij Računarstva, smjer Procesno računarstvo. Od stranih jezika vrlo dobro priča Engleski jezik. Posjeduje vozačku dozvolu B kategorije.

PRILOZI

Prilog 1 : Programski kod PC aplikacije za nadzor i upravljanje (C#).

Izvorni programski kod za …

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO.Ports;


namespace pokretne_trake
{
    public partial class Form1 : Form
    {
        string primljeno;
        private class Item
        {
            // definiranje da bi mogli popunjavati Comboboxeve
            public string Name;
            public int Value;
            public Item(string name, int value)
            {
                Name = name; Value = value;
            }
            public override string ToString()
            {
                return Name;
            }

        }

        public Form1()
        {
            InitializeComponent();
            combo_serial.Items.Add(new Item("COM1", 1));
            combo_serial.Items.Add(new Item("COM2", 2));
            combo_serial.Items.Add(new Item("COM3", 3));
            combo_serial.Items.Add(new Item("COM4", 4));
            combo_serial.Items.Add(new Item("COM5", 5));
            combo_serial.Items.Add(new Item("COM6", 6));
            combo_serial.Items.Add(new Item("COM7", 7));
            combo_serial.Items.Add(new Item("COM8", 8));
            combo_serial.Items.Add(new Item("COM9", 9));
            combo_serial.Items.Add(new Item("COM10", 10));
            combo_serial.Items.Add(new Item("COM11", 11));
            combo_serial.Items.Add(new Item("COM12", 12));
            combo_serial.Items.Add(new Item("COM13", 13));
            combo_serial.Items.Add(new Item("COM14", 14));
            combo_serial.Items.Add(new Item("COM15", 15));

            comboBoja.Items.Add(new Item("CRVENA", 16));
            comboBoja.Items.Add(new Item("ZELENA", 17));
            comboBoja.Items.Add(new Item("PLAVA", 18));
        }
```

```
private void pokreni_traka1_Click(object sender, EventArgs e)
{
    serialPort1.Write("1");
    if (pokreni_traka1.Text == "Pokreni traka1")
    {
        pokreni_traka1.Text = "Zaustavi traka1";
    }
    else
    {
        pokreni_traka1.Text = "Pokreni traka1";
    }
}

private void odaberi_port_Click(object sender, EventArgs e)
{
    serialPort1.PortName = combo_serial.SelectedItem.ToString();
    serialPort1.BaudRate = 9600;
    serialPort1.DtrEnable = true;
    try
    {
        if (!serialPort1.IsOpen)
        {
            serialPort1.DataReceived += serialPort1_DataReceived;
            serialPort1.Open();
            connect_textBox.Text = "  Connect!";
            connect_textBox.ForeColor = Color.Green;
        }
    }
    catch (Exception ex)
    {
        connect_textBox.Text = "  Disconnect!";
    }
}

private void serialPort1_DataReceived(object sender, System.IO.Ports.SerialDataReceivedEventArgs e)
{
    primljeno = serialPort1.ReadLine();
    this.Invoke(new EventHandler(timer1_Tick));
}

private void timer1_Tick(object sender, EventArgs e)
{
    timer1.Start();
    paket.Text = primljeno.ToString();
    char slovo_0 = primljeno[0];
    char slovo_1 = primljeno[1];
    char slovo_2 = primljeno[2];
    char slovo_3 = primljeno[3];
    char slovo_4 = primljeno[4];
    char slovo_5 = primljeno[5];
    string broj_paketa_lijevi = primljeno.Split(']', '|')[1];
    string broj_paketa_desni = primljeno.Split('|').Last();

    string pocetni = "[";
    string zavrsni = "]";
    string nula = "0";
    string jedan = "1";
    string neutralno = "N";
    string desno = "D";
    string lijevo = "L";
    string boja_nema = "0";
    string boja_plava = "1";
    string boja_zelena = "4";
    string boja_crvena = "5";
```

```
string boja_bijela = "6";
// Parsiranje
if (slovo_0.ToString() == pocetni.ToString() && slovo_5.ToString() == zavrsni.ToString())
{
    ispravnost.Text = "Ispravan paket";
    ispravnost.ForeColor = Color.Green;
}
else
{
    ispravnost.Text = "Neispravan paket";
    ispravnost.ForeColor = Color.Red;
}

if (slovo_1.ToString() == nula.ToString())
{
    pictureBox_traka1.Image = Properties.Resources.motor1_stop;
}
else if (slovo_1.ToString() == jedan.ToString())
{
    pictureBox_traka1.Image = Properties.Resources.motor1_naprijed;

}
if (slovo_2.ToString() == nula.ToString())
{
    pictureBox_traka2.Image = Properties.Resources.motor2_stop;
}
if (slovo_3.ToString() == neutralno.ToString())
{
    pictureBox_traka2.Image = Properties.Resources.motor2_stop;
}
else if (slovo_3.ToString() == lijevo.ToString())
{
    pictureBox_traka2.Image = Properties.Resources.motor2_lijevo; // ide lijevo
}
else if (slovo_3.ToString() == desno.ToString())
{
    pictureBox_traka2.Image = Properties.Resources.motor2_desno; // ide desno
}
if (slovo_4.ToString() == boja_nema.ToString())
{
   // boja.Text = "Nema boje";
}
else if (slovo_4.ToString() == boja_plava.ToString())
{
    boja.Text = "Plava";
    boja.ForeColor = Color.Blue;
}
else if (slovo_4.ToString() == boja_crvena.ToString())
{
    boja.Text = "Crvena";
    boja.ForeColor = Color.Red;
}
else if (slovo_4.ToString() == boja_zelena.ToString())
{
    boja.Text = "Zelena";
    boja.ForeColor = Color.Green;
}
/*   else if (slovo_4.ToString() == boja_bijela.ToString())
{
    boja.Text = "Bijela";
    boja.ForeColor = Color.Black;
}*/
br_pre_lijevo.Text = broj_paketa_lijevi.ToString();
br_pre_desno.Text = broj_paketa_desni.ToString();
```

```
        }

        private void Manualni_rad_Click(object sender, EventArgs e)
        {

            serialPort1.Write("5");
            if (Manualni_rad.Text == "Automatski rad")
            {
                Manualni_rad.Text = "Manualni rad";
            }
            else
            {
                Manualni_rad.Text = "Automatski rad";
            }
        }
        private void Stop_traka2_Click(object sender, EventArgs e)
        {

            serialPort1.Write("6");
            if (Stop_traka2.Text == "Stop")
            {
                Stop_traka2.Text = "Pokreni";
            }
            else
            {
                Stop_traka2.Text = "Stop";
            }
        }

        private void Lijevo_traka2_Click(object sender, EventArgs e)
        {
            serialPort1.Write("7");
        }

        private void Desno_traka2_Click(object sender, EventArgs e)
        {
            serialPort1.Write("8");
        }

        private void spremi_boja_Click(object sender, EventArgs e)
        {
            Izabrana_boja.Text = comboBoja.SelectedItem.ToString();
            string plava = "PLAVA";
            string crvena = "CRVENA";
            string zelena = "ZELENA";

            if (Izabrana_boja.Text == plava.ToString())
            {
                serialPort1.Write("9");
                MessageBox.Show("Izabrana boja PLAVA");
            }
            else if (Izabrana_boja.Text == crvena.ToString())
            {
                serialPort1.Write("3");
                MessageBox.Show("Izabrana boja CRVENA");
            }
            else if (Izabrana_boja.Text == zelena.ToString())
            {
                serialPort1.Write("2");
                MessageBox.Show("Izabrana boja ZELENA");
            }
        }
    }
}
```

Prilog 2: Programski kod korišten u mikroupravljačkoj pločici Arduino Mega 2560.

Izvorni programski kod za …

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);

char kretanje_motor1 = '0';
char kretanje_motor2 = '0';
char kretanje_motor2_smjer = 'N';
char boja = '6';

int buttonState = 0;
int lastButtonState = 0;
int kretanje;
int kretanje_traka2;
int buttonManual = 0;
int lastButtonManual = 0;
int manualrad;

int stanjeplava;

int button_drugatraka_desno = 0;
int lastButton_drugatraka_desno = 0;
int kretanje_drugatraka_desno = 0;

int button_drugatraka_lijevo = 0;
int lastButton_drugatraka_lijevo = 0;
int kretanje_drugatraka_lijevo = 0;

int button_drugatraka_stop = 0;
int lastButton_drugatraka_stop = 0;
int kretanje_drugatraka_stop = 0;

//timer
int senzor0reg;
unsigned long currentMillis = 0;
unsigned long previousMillis = 0;
const long interval = 600;

//brojac kontrola
int brojac_prvatraka = 0;
int brojproizvoda_prvatraka = 0;
int last_brojac_prvatraka = 0;
int brojac_drugatraka = 0;
int last_brojac_drugatraka = 0;
int brojac_drugatraka2 = 0;
int last_brojac_drugatraka2 = 0;
int brojproizvoda_drugatraka = 0;

int kontrola_brojaca;

int value1 = 0;
int value2 = 0;
int value3 = 0;
int prevval1 = 0;
int prevval2 = 0;
int prevval3 = 0;
int count1 = 0;
int count2 = 0;
int count3 = 0;
```

```
int setboja;
int brojac1 = 0;
int brojac2 = 0;
int brojac3 = 0;

// senzor posuda
int korpe = 36;
int korpe_status = 0;
int korpe_akt;
int clcdisplej;

//senzor za boju
int S0 = 50;
int S1 = 52;
int S2 = 48;
int S3 = 46;
int OUT = 42;
int LED = 40;
int redPin, greenPin, bluePin;

// IR senzori
int senzor0 = 2;      // senzor prije color senzora
int senzor1 = 3;      // srednji senzor
int senzor2 = 18;    // lijevi senzor
int senzor3 = 19;    // desni senzor
int senzor0_status = 0;
int senzor1_status = 0;
int senzor2_status = 0;
int senzor3_status = 0;
int senzor_akt0;
int senzor_akt1;
int senzor_akt2;
int senzor_akt3;

//motor1
int mot1Pin1 = 10;
int mot1Pin2 = 11;
int speedPin1 = 12;

//motor 2

int mot2Pin1 = 8;
int mot2Pin2 = 9;
int speedPin2 = 7;

void setColor(int red, int green, int blue) {
analogWrite(redPin, 255-red);
analogWrite(greenPin, 255-green);
analogWrite(bluePin, 255-blue);
}

void TCS3200_Setup() {
  pinMode(S0,OUTPUT);
  pinMode(S1,OUTPUT);
  pinMode(S2,OUTPUT);
  pinMode(S3,OUTPUT);
  pinMode(LED,OUTPUT);
  pinMode(OUT,INPUT);
}
void TCS3200_On() {
  digitalWrite(LED,HIGH);
  digitalWrite(S0,HIGH);
  digitalWrite(S1,HIGH);
```

```
  delay(5);
}
void TCS3200_Off() {
  digitalWrite(LED,LOW);
  digitalWrite(S0,LOW);
  digitalWrite(S1,LOW);
}
void NoFilter() {
  digitalWrite(S2,HIGH);
  digitalWrite(S3,LOW);
  delay(5);
}
void RedFilter() {
  digitalWrite(S2,LOW);
  digitalWrite(S3,LOW);
  delay(5);
}
void GreenFilter() {
  digitalWrite(S2,HIGH);
  digitalWrite(S3,HIGH);
  delay(5);
}
void BlueFilter() {
  digitalWrite(S2,LOW);
  digitalWrite(S3,HIGH);
  delay(5);
}
void GetColor() {

        float FrequencyClear,FrequencyRed,FrequencyGreen,FrequencyBlue;
         int PercentageRed,PercentageGreen,PercentageBlue;
        TCS3200_On();
        NoFilter();
        FrequencyClear=500.0/pulseIn(OUT,LOW,10000);
        RedFilter();
        FrequencyRed=500.0/pulseIn(OUT,LOW,10000);
        GreenFilter();
         FrequencyGreen=500.0/pulseIn(OUT,LOW,10000);
        BlueFilter();
        FrequencyBlue=500.0/pulseIn(OUT,LOW,10000);
        TCS3200_Off();

        PercentageRed=int((FrequencyRed/FrequencyClear)*100.0);
        PercentageGreen=int((FrequencyGreen/FrequencyClear)*100.0);
        PercentageBlue=int((FrequencyBlue/FrequencyClear)*100.0);

        float udaljenost_Red = 85; // od centra boje do ishodišta
        float udaljenost_Green = 85;
        float udaljenost_Blue = 90;

        float rgb_Red = (255 * PercentageRed)/100;
        float rgb_Green = (255 * PercentageGreen)/100;
        float rgb_Blue = (255 * PercentageBlue)/100;

        float udaljenost_boje1 = sqrt(((200-rgb_Red)*(200-rgb_Red))+((50-rgb_Green)*(50-rgb_Green))+((50-
        rgb_Blue)*(50-rgb_Blue)));    //udaljenosti od mjerene točke do centra boje
        float udaljenost_boje2 = sqrt(((80-rgb_Red)*(80-rgb_Red))+((175-rgb_Green)*(175-rgb_Green))+((40-
        rgb_Blue)*(40-rgb_Blue)));
        float udaljenost_boje3 = sqrt(((45-rgb_Red)*(45-rgb_Red))+((35-rgb_Green)*(35-rgb_Green))+((180-
        rgb_Blue)*(200-rgb_Blue)));

        if(udaljenost_boje1 < udaljenost_Red){
         //Serial.println("Crvenaaa");
        setboja = 5;
```

```
              boja = '5';
              }
              if(udaljenost_boje2 < udaljenost_Green){
              //Serial.println("Zelenaaa");
              setboja = 4;
              boja = '4';
              }
              if(udaljenost_boje3 < udaljenost_Blue){
               // Serial.println("Plavaaa");
               setboja = 1;
               boja = '1';
              }
}

// stanja motora1
    void motor1_naprijed()
  {
     analogWrite( speedPin1, 100);
     digitalWrite(mot1Pin1, HIGH);
     digitalWrite(mot1Pin2, LOW);
     kretanje_motor1 = '1';
   }

    void motor1_stop()
    {
     analogWrite( speedPin1, 255);
     digitalWrite(mot1Pin1, LOW);
     digitalWrite(mot1Pin2, LOW);
     kretanje_motor1 = '0';
    }

   void motor1_usporen()
    {
     analogWrite( speedPin1, 0);
     digitalWrite(mot1Pin1, HIGH);
     digitalWrite(mot1Pin2, LOW);
     kretanje_motor1 = '1';
    }

// stanja motora2
  void motor2_lijevo()
  {
   analogWrite( speedPin2, 75);
   digitalWrite(mot2Pin1, HIGH);
   digitalWrite(mot2Pin2, LOW);
   kretanje_motor2 = '1';
   kretanje_motor2_smjer = 'L';
   }

   void motor2_desno()
  {
     analogWrite( speedPin2, 75);
     digitalWrite(mot2Pin1, LOW);
     digitalWrite(mot2Pin2, HIGH);
     kretanje_motor2 = '1';
     kretanje_motor2_smjer = 'D';
   }

    void motor2_stop()
    {
     analogWrite( speedPin2, 0);
     digitalWrite(mot2Pin1, LOW);
     digitalWrite(mot2Pin2, LOW);
     kretanje_motor2 = '0';
```

```
    kretanje_motor2_smjer = 'N';
   }

void setup(){
 pinMode(mot1Pin1, OUTPUT);
 pinMode(mot1Pin2, OUTPUT);
 pinMode(mot2Pin1, OUTPUT);
 pinMode(mot2Pin2, OUTPUT);
 pinMode(speedPin1, OUTPUT);
 pinMode(speedPin2, OUTPUT);
 pinMode(senzor0, INPUT);
 pinMode(senzor1, INPUT);
 pinMode(senzor2, INPUT);
 pinMode(senzor3, INPUT);
 pinMode(korpe, INPUT);

//lcd ispis
lcd.begin();
lcd.setCursor(0,0);
lcd.print("Diplomski rad");
lcd.setCursor(0,1);
lcd.print("Zvonimir Balent");

 TCS3200_Setup();
 Serial.begin(9600);
}

void loop() {
int a=Serial.read();

 GetColor();
//delay(150);

value1 = digitalRead(senzor1);
value2 = digitalRead(senzor2);
value3 = digitalRead(senzor3);

if(value1 == LOW && prevval1 ==1){
  count1++;
}
prevval1 = value1;

if(value2 == LOW && prevval2 ==1){
  count2++;
}
prevval2 = value2;

if(value3 == LOW && prevval3 ==1){
  count3++;
}
prevval3 = value3;

 // očitavanje senzora položaja
 senzor0_status = digitalRead(senzor0);
 digitalWrite(senzor_akt0,senzor0_status);
 senzor1_status = digitalRead(senzor1);
 digitalWrite(senzor_akt1,senzor1_status);
 senzor2_status = digitalRead(senzor2);
 digitalWrite(senzor_akt2,senzor2_status);
 senzor3_status = digitalRead(senzor3);
 digitalWrite(senzor_akt3,senzor3_status);
 korpe_status = digitalRead(korpe);
 digitalWrite(korpe_akt,korpe_status);
```

```
 //Zaustavljanje trake, korpa dignuta!
if(korpe_status == HIGH){
 // motor1_stop();
    analogWrite(speedPin1, 255);
    digitalWrite(mot1Pin1, LOW);
    digitalWrite(mot1Pin2, LOW);
    kretanje_motor1 = '0';
 // motor2_stop();
    analogWrite(speedPin2, 255);
    digitalWrite(mot2Pin1, LOW);
    digitalWrite(mot2Pin2, LOW);
    kretanje_motor2 = '0';
}

/////////////////////automatski rad //////////////////
  if(a=='5'){
  buttonManual = 1;

        if( buttonManual != lastButtonManual){
                manualrad = 1;
                count1 = 0;
                count2 = 0;
                count3 = 0;
        }
        else{
                 buttonManual = 0;
                 manualrad = 0;
                count1 = 0;
                count2 = 0;
                count3 = 0;
        }
        lastButtonManual = buttonManual;
}

 if(korpe_status == HIGH){
 // motor1_stop();
    analogWrite(speedPin1, 255);
    digitalWrite(mot1Pin1, LOW);
    digitalWrite(mot1Pin2, LOW);
    kretanje_motor1 = '0';
 // motor2_stop();
    analogWrite(speedPin2, 255);
    digitalWrite(mot2Pin1, LOW);
    digitalWrite(mot2Pin2, LOW);
    kretanje_motor2 = '0';
    kretanje_motor2_smjer = 'N';
}

if(manualrad == 1){
  kretanje_traka2 = 0;

//pokretanje prve trake i usporavanje pri dolasku na color senzor
if(a=='1'){
  buttonState = 1;
  if( buttonState != lastButtonState){
  kretanje = 1;
 }
else{
buttonState = 0;
kretanje = 0;
 }
 lastButtonState = buttonState;
}
```

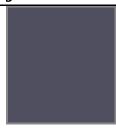
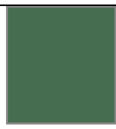
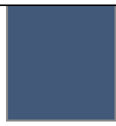
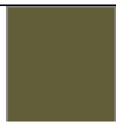
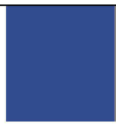
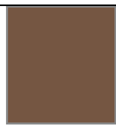
```
if(kretanje == 0){
   analogWrite( speedPin1, 100);
     digitalWrite(mot1Pin1, LOW);
     digitalWrite(mot1Pin2, LOW);
     kretanje_motor1 = '0';
}
if(kretanje == 1 && senzor0reg == 0 && kontrola_brojaca == 0 && korpe_status == LOW){
    analogWrite( speedPin1, 100);
     digitalWrite(mot1Pin1, HIGH);
     digitalWrite(mot1Pin2, LOW);
     kretanje_motor1 = '1';
}

if(senzor0_status == LOW){
     senzor0reg = 1;
      }
        if(senzor0reg == 1){
          currentMillis = millis();
            if(currentMillis - previousMillis <= interval){
              analogWrite( speedPin1, 7);
              digitalWrite(mot1Pin1, HIGH);
              digitalWrite(mot1Pin2, LOW);
              kretanje_motor1 = '1';
            }
            else{
              senzor0reg = 0;
              previousMillis = currentMillis;
            }
        }

//kontrola prve i druge trake
if(count1 > (count2 + count3)){
     analogWrite( speedPin1, 0);
     digitalWrite(mot1Pin1, LOW);
     digitalWrite(mot1Pin2, LOW);
     kretanje_motor1 = '0';
    }

else{
  kontrola_brojaca = 0;
}

if(senzor2_status ==LOW || senzor3_status ==LOW){
  delay(250);
  motor2_stop();
    kretanje_motor2 = '0';

}

//raspodjela po boji

if(a=='9'){
  stanjeplava = 1;
}

if(a=='2'){
  stanjeplava = 2;
  }

if(a=='3'){
 stanjeplava = 3;
}
```

```
if(setboja == 1 && stanjeplava == 1){
  motor2_desno();
  kretanje_motor2_smjer = 'D';
}
     else if((setboja == 4 || setboja == 5) && stanjeplava == 1){
      motor2_lijevo();
      kretanje_motor2_smjer = 'L';
     }

if(setboja == 4 && stanjeplava == 2){
  motor2_desno();
  kretanje_motor2_smjer = 'D';
}
     else if((setboja == 1 || setboja == 5) && stanjeplava == 2){
      motor2_lijevo();
      kretanje_motor2_smjer = 'L';
     }

if(setboja == 5 && stanjeplava == 3){
  motor2_desno();
  kretanje_motor2_smjer = 'D';
}
     else if((setboja == 4 || setboja == 1) && stanjeplava == 3){
      motor2_lijevo();
      kretanje_motor2_smjer = 'L';
     }
}

/////////////// Manualni rad ///////////////
if(manualrad == 0){
  kretanje_traka2 = 1;

if(korpe_status == HIGH){
 // motor1_stop();
   analogWrite(speedPin1, 0);
     digitalWrite(mot1Pin1, LOW);
     digitalWrite(mot1Pin2, LOW);
     kretanje_motor1 = '0';
 // motor2_stop();
   analogWrite(speedPin2, 0);
     digitalWrite(mot2Pin1, LOW);
     digitalWrite(mot2Pin2, LOW);
     kretanje_motor2 = '0';
     kretanje_motor2_smjer = 'N';
}

  if(a=='1'){
  buttonState = 1;
 if( buttonState != lastButtonState){
  kretanje = 1;
 }
 else{
  buttonState = 0;
  kretanje = 0;
 }
 lastButtonState = buttonState;
}

if(kretanje == 0){
   analogWrite( speedPin1, 100);
     digitalWrite(mot1Pin1, LOW);
     digitalWrite(mot1Pin2, LOW);
     kretanje_motor1 = '0';
}
```

```
if(kretanje == 1 && korpe_status == LOW){
   analogWrite( speedPin1, 100);
    digitalWrite(mot1Pin1, HIGH);
    digitalWrite(mot1Pin2, LOW);
    kretanje_motor1 = '1';
}

if(senzor0_status == LOW){
    senzor0reg = 1;
     }
       if(senzor0reg == 1){
         currentMillis = millis();
           if(currentMillis - previousMillis <= interval){
             analogWrite( speedPin1, 7);
             digitalWrite(mot1Pin1, HIGH);
             digitalWrite(mot1Pin2, LOW);
             kretanje_motor1 = '1';
           }
           else{
             senzor0reg = 0;
             previousMillis = currentMillis;
           }
       }
//////////////////////////////

 if(a=='6' && kretanje_traka2 == 1){
  button_drugatraka_stop = 1;
 if( button_drugatraka_stop != lastButton_drugatraka_stop){
  kretanje_drugatraka_stop = 1;
  kretanje_drugatraka_lijevo = 0;
  kretanje_drugatraka_desno = 0;
 }
 else{
  button_drugatraka_stop = 0;
  kretanje_drugatraka_stop = 0;
 }
 lastButton_drugatraka_stop = button_drugatraka_stop;
}

 if(a=='7' && kretanje_drugatraka_stop == 0 && kretanje_traka2 == 1 ){
  kretanje_drugatraka_lijevo = 1;
 }

if(a=='8' && kretanje_drugatraka_stop == 0 && kretanje_traka2 == 1){
 // button_drugatraka_desno = 1;
  kretanje_drugatraka_desno = 1;
}

   if(kretanje_drugatraka_stop == 1 && kretanje_traka2 == 1){
    analogWrite( speedPin2, 100);
    digitalWrite(mot2Pin1, LOW);
    digitalWrite(mot2Pin2, LOW);
    kretanje_motor2_smjer = 'N';

       }

      if(kretanje_drugatraka_lijevo == 1 && kretanje_drugatraka_desno == 0 && korpe_status == LOW &&
kretanje_traka2 == 1){
        analogWrite( speedPin2, 90);
        digitalWrite(mot2Pin1, HIGH);
        digitalWrite(mot2Pin2, LOW);
         kretanje_motor2_smjer = 'L';
      }
      if(kretanje_drugatraka_desno == 1 && kretanje_drugatraka_lijevo == 0 && korpe_status == LOW &&
```

```
kretanje_traka2 == 1){
          analogWrite( speedPin2, 90);
          digitalWrite(mot2Pin1, LOW);
          digitalWrite(mot2Pin2, HIGH);
           kretanje_motor2_smjer = 'D';
      }

}

// komunikacija sa C#
Serial.print("["), Serial.print(kretanje_motor1), Serial.print(kretanje_motor2), Serial.print(kretanje_motor2_smjer),
Serial.print(boja),Serial.print("]"), Serial.print(count2), Serial.print("|"), Serial.println(count3);
}
```

Prilog 3: Tablica mjerenja senzora za prepoznavanje boje TCS3200

Naziv predmeta i boja:	Slika predmeta:	RBG vrijednost očitana na maketi:	Očitana RGB vrijednost prikazana kao boja:
DC elektromotor Siva boja		R 79 G 79 B 96	
Kockica Zelena boja		R 71 G 109 B 81	
Kockica Svijetlo plava boja		R 66 G 89 B 122	
Kockica Žuta boja		R 99 G 94 B 58	
Kockica Plava boja		R 51 G 76 B 145	
Igračka auto Žuta boja		R 117 G 86 B 66	

Čep Bijela boja		R 86 G 76 B 91	
Gumica Bijela boja		R 99 G 89 B 94	
Čep Smeđa boja		R 124 G 71 B 79	
Kockica Svijetlo zelena boja		R 109 G 104 B 61	
Punjač Crna boja		R 81 G 79 B 99	
Kockica Crvena boja		R 168 G 48 B 68	