SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

ELEKTROTEHNIČKI FAKULTET

Sveučilišni studij

MIKROUPRAVLJAČKI SUSTAV JEDNOSTAVNOG MOBILNOG ROBOTA

Završni rad

Ivan Jakopiček

Osijek, 2010.

Sadržaj:

1. UVOD

Tema ovog završnog rada je, kako se i vidi iz naslova, mikroupravljač za mobilnog robota. Tijekom rada potrebno je opisati opće i fundamentalne dijelove mikroupravljača te njihovo funkcioniranje, a zatim posebno upravljanje mobilnim robotom pomoću mikroupravljača, njegovih senzora i motora. Rad se fokusira na mikroupravljače te njihovo programiranje za uspješan rad više nego na izradu samog robota i njegove periferne jedinice.

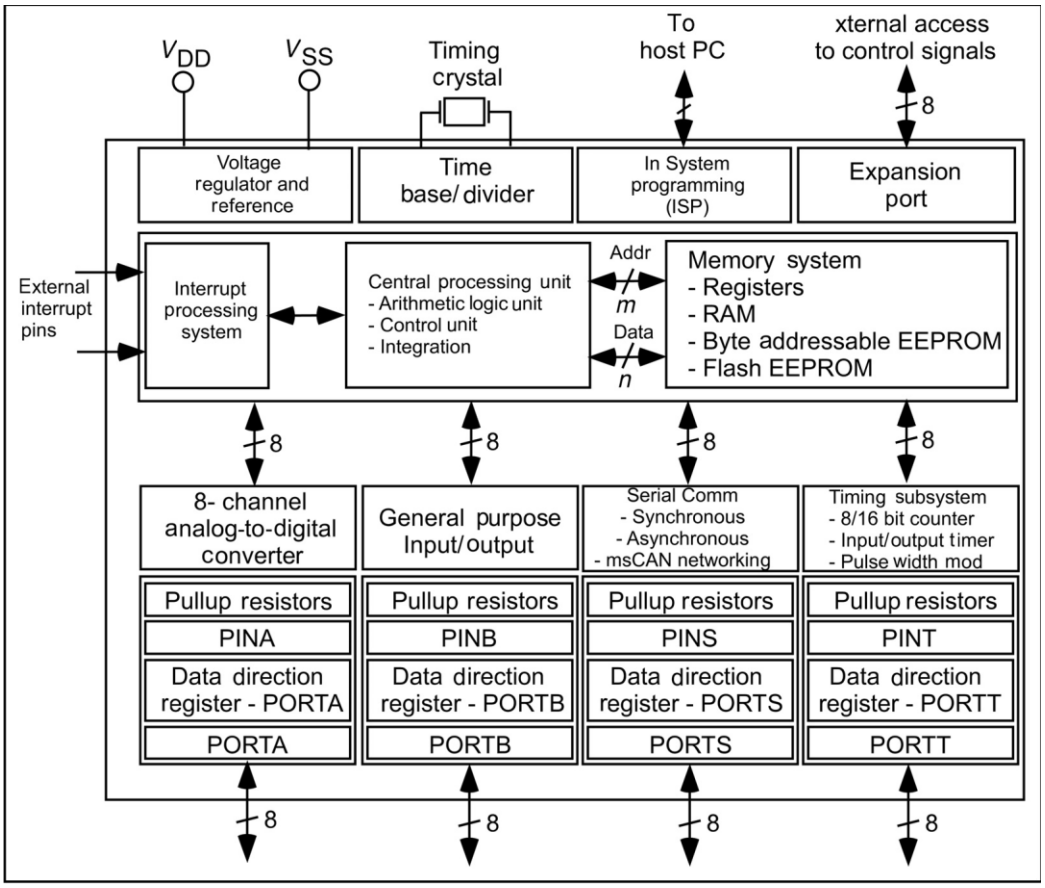
U sljedećem poglavlju bit će opisani mikroupravljači općenito, osnovne razlike između mikroupravljača i mikroprocesora te dijelovi i različite izvedbe mikroupravljača. U trećem poglavlju uvest će se pojam mobilnih robota i njihova klasifikacija te navesti vrsta i razlog odabira robota i mikroupravljača korištenog u ovom radu. U četvrtom poglavlju opisat će se izrada i način upravljanja robotom te postupak programiranja Atmel ATmega 8 AVR mikroupravljača za upravljanje odabranim mobilnim robotom.

2. MIKROUPRAVLJAČI

2.1 Što je mikroupravljač

Mikroupravljač je u svom temeljnom obliku cijeli računalni sustav sadržan na jedinstvenom integriranom sklopu. Jedan od primarnih izazova kod dizajniranja mikroupravljača jest odabir upravljača za određeni dizajn. Cilj je odabrati najekonomičniji mikroupravljač koji ima željene parametre i mogućnosti za određene zadaće.

Mikroupravljači se proizvode u rasponu od malih, 4 bitnih procesora sa ograničenim mogućnostima do 32 bitnih procesora velike brzine. Većina mikroupravljača opremljena je sa podsustavima, navedenih na slici 2.1. Pristupi (engl. *port*) se koriste kako bi mikroupravljač mogao komunicirati sa vanjskim svijetom. Standardno, pristupi su dvosmjerni i imaju izmjenične funkcije kao što su analogno-digitalna pretvorba, serijska komunikacija i fleksibilni sustav mjerenja vremena. Mikroupravljači su, također, opremljeni cijelim komplementom različitih memorijskih komponenti. Normalna operacija mikroupravljača može biti prekinuta vanjskim prekidom koristeći *pinove* za prekid. Ovo omogućava da mikroupravljač brzo odgovara na operacije visokog prioriteta. Mikroupravljač se programira preko ISP (engl. *In System Programming*) sustava koristeći osobno računalo. Vremenska baza ili takt mikroupravljača određuje se eksternim kristalnim oscilatorom ili rezonatorom.



Sl. 2.1. *Blok - dijagram mikroupravljača*[1]

2.2 Razlika između mikroupravljača i mikroprocesora

Mikroupravljač se od mikroprocesora razlikuje na mnogo načina. Prvi i najvažniji je njegova funkcionalnost. Da bi se mikroprocesor mogao upotrijebiti, moraju mu se dodati druge komponente poput memorije ili komponenti za prijem i slanje podataka. To, ukratko, znači da je mikroprocesor mozak računala. S druge strane mikroupravljač je dizajniran da bude sve u jednom. Za njegovu primjenu ne trebaju nikakve vanjske komponente jer se sve potrebne periferne jedinice nalaze unutar uređaja.

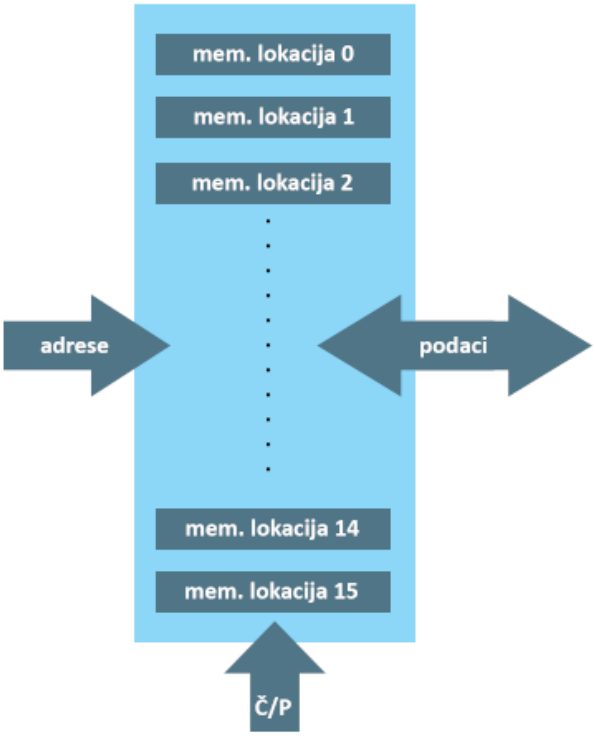
[1]] S. F. Barrett, D. J. Pack, *Microcontrollers Fundamentals for Engineers and Scientists,* Morgan & Claypool, San Rafael, CA, 2006., str. 30

2.3. Dijelovi i izvedba mikroupravljača

2.3.1. Memorijska jedinica

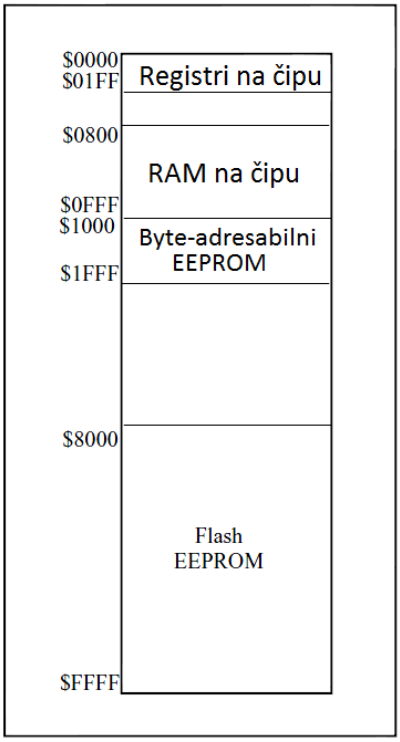
Memorija je dio mikroupravljača čija je primarna svrha čuvanje podataka. Najlakše ju je objasniti kao jedan veliki ormar sa puno ladica. Ako se pretpostavi da je označavanje ladica takvo da ne može doći do zabune, onda će svaki sadržaj bilo koje ladice biti lako pronađen. Dovoljno je znati oznaku ladice i time je njezin sadržaj sigurno poznat.

Memorijske komponente su baš takve, za određeni ulaz dobiva se sadržaj adresirane memorijske lokacije i to je sve. Memoriju čine sve memorijske lokacije, a adresiranje je ništa drugo nego izbor jedne od njih. Znači, potrebno je jedne strane izabrati koja se memorijska lokacija želi, a sa druge strane sačekati sadržaj te lokacije. Pored čitanja memorijske lokacije memorija mora također osigurati i unos u njih. To je razrađeno uvođenjem nove, kontrolne linije. Ona je prema slici 2.2. označena sa Č/P (čitaj/piši). Kontrolna linija radi na sljedeći način: ukoliko je č/p=1, tada se radi čitanje, a u suprotnom se radi pisanje.



Sl. 2.2. *Memorija*

Broj jedinstveno adresabilnih memorijskih lokacija u mikroupravljaču određuje širina adresne sabirnice. Ovaj raspon adresabilnih memorija obično sadrži nekoliko različitih vrsta memorija uključujući statičnu memoriju sa slučajnim pristupom SRAM (*Static Random Access Memory*), *byte*-adresabilnu električnu stalnu memoriju s mogućnošću programiranja (EEPROM), i *bulk*-programabilni Flash EEPROM. Treba naglasiti da se svi ovi dijelovi memorije nalaze unutar granica mikroupravljača. Kako bi pratili memorijske lokacije u uporabi i vrstu memorije prisutne unutar sustava, koristi se vizualni alat zvan memorijska mapa. Memorijska mapa pruža veličine svake memorijske komponente u bajtovima i njezinu adresu pokretanja i zaustavljanja unutar memorijskog sustava. Jednostavna memorijska karta prikazana je na slici 2.3. Postoje i neoznačeni dijelovi memorije zbog mogućnosti proširivanja sustava.



Sl. 2.3. *Memorijska mapa*[2]

[2]] S. F. Barrett, D. J. Pack, *Microcontrollers Fundamentals for Engineers and Scientists,* Morgan & Claypool, San Rafael, CA, 2006., str. 36.

Sada će biti opisane komponente sustava memorijske mape sa slike 2.3. Većina njih je sadržana u gotovo svim mikroupravljačkim sustavima.

RAM: RAM memorija je nepostojana. To znači, ako mikroupravljač izgubi napajanje, sadržaj RAM memorije se gubi. U nju se može upisivati i čitati za vrijeme izvršavanja programa. Ona se obično koristi za vrijeme razvoja sustava te za pohranu podataka. Kada se razvoj programa završi, on se pohranjuje u statičnu memoriju kao što je EEPROM. Za vrijeme izvršavanja programa RAM se koristi za pohranu globalne varijable kao podrška dinamičnoj memoriji pri dodjeljivanju varijabli, te osiguravanje mjesta za stog.

EEPROM (*Byte*-adresabilan): Ovaj tip memorije se koristi za trajno spremanje i opoziv varijabli tijekom izvršavanja programa. Osobito je koristan za prijave kvarova sustava i kvara podataka tijekom izvršavanja programa. Također je koristan za pohranjivanje podataka koji moraju biti spremljeni tijekom nestanka struje, ali ih možda treba mijenjati periodično. Primjeri za korištenje ovog tipa memorije nalaze se u aplikacijama za spremanje parametara sustava, elektroničkim kombinacijama za ključanje i automatskim garažnim vratima.

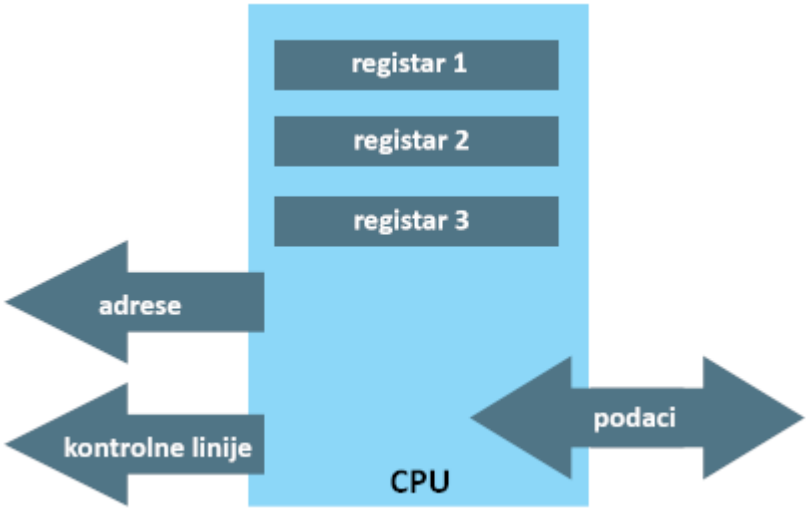
EEPROM(*Flash*): *Bulk*-programabilni *Flash* EEPROM se koristi za pohranu programa. On se može brisati i programirati kao cjelina. Neki sustavi nude mikroupravljač sa velikim brojem RAM-a *i Flash* EEPROM-a. Dakle, sustav programa se može razviti u RAM-u, a potom se prenosi na *Flash* EEPROM.

Ostali mikroupravljači pružaju samo veliku *Flash* EEPROM i manju RAM komponentu. Ovom konfiguracijom memorije sustav se razvija u *Flash* EEPROM-u. *Flash* EEPROM obično je programiran pomoću već spomenute ISP tehnike.

2.3.2. Centralna procesorska jedinica – CPU

Centralna procesorska jedinica (CPU) unutar mikroupravljača je složen sekvencijalni sklop čija je primarna funkcija izvršavanje programa koji su pohranjeni u okviru memorije. Program je jednostavno niz uputa za obavljanje određenog zadatka. Programe razvijaju dizajneri mikroupravljačkog sustava pomoću razvojnih alata.

Prilikom obrade podataka CPU se koristi svojim unutarnjim registrima (slika 2.4.) koji mu omogućavaju brzu obradu, pohranjivanje i uzimanje podataka. Registri su, dakle, memorijske lokacije čija je uloga pomaganje pri obavljanju raznih matematičkih operacija ili bilo kojih drugih operacija sa podacima ma gdje se oni nalazili.



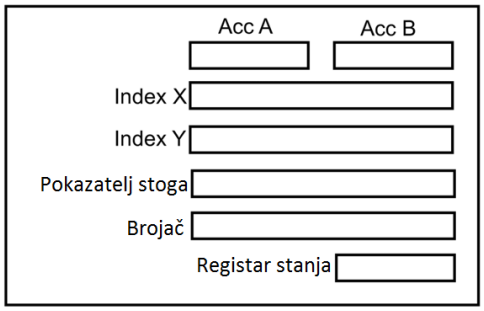
Sl. 2.4. *Jednostavna shema CPU-a*

Kada je razvoj programa gotov, program se preuzima (*download*) u mikroupravljač. Tada mikroupravljač postaje samostalan sustav za obradu podataka. Nakon *reseta* mikroupravljača CPU će sekvencijalno dohvatiti programske instrukcije iz memorije, dekodirati sadržaj instrukcija te ih izvršiti. Centralna procesna jedinica (CPU) je glavni centar za cijeli mikroupravljač. Dok odgovara

na programske upute, CPU će pozivati njegove podsustave da obavljaju svoje zadaće. Osnovne arhitekture procesora obično se mogu staviti u jednu od nekoliko općih kategorija. Treba naglasiti da ni jedna arhitektura nije bolja od druge. Svaka od njih ima svoje prednosti i nedostatke.

Akumulatorska arhitektura

U akumulatorskoj arhitekturi, ilustriranoj na slici 2.5., upute počinju i završavaju u posebno određenim registrima zvanim akumulatori (A i B). Tipično, operacija se obavlja u onom registru u kojem se nalazi operand, a drugi se uzima iz memorije. Rezultat se tada stavlja u akumulator. Ova arhitektura teži sporijem radu od ostalih budući da se operandi moraju konstantno uzimati iz memorije. Tipično, memorija je sporija od glavnog procesora, zato procesor mora usporavati da bi uspješno dohvatio operande iz memorije, no akumulatorski zasnovane arhitekture imaju sposobnost izvršavanja puno kompliciranijih naredbi.

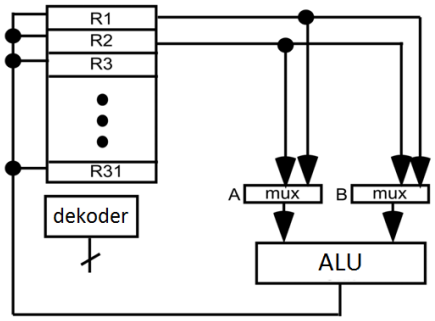


Sl. 2.5. *Akumulatorska arhitektura*[3]

Registarska arhitektura

U arhitekturi zasnovanoj na registrima (Sl. 2.6.), oba operanda pohranjuju se u registar koji je obično kolociran sa središnjom procesorskom jedinicom. Rezultat se također pohranjuje u registar. Budući da CPU i registri rade istom brzinom, procesor ne mora usporiti za čitanje ili pisanje operanada. Sadržaji registara se čitaju ili pišu u memoriju korištenjem pozadinskih operacija.

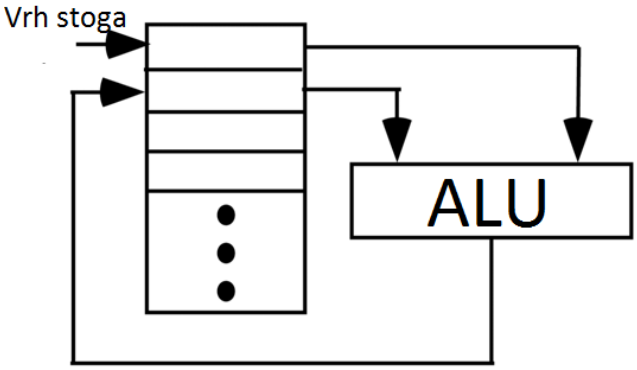
[3]] S. F. Barrett, D. J. Pack, *Microcontrollers Fundamentals for Engineers and Scientists,* Morgan & Claypool, San Rafael, CA, 2006., str. 31



Sl. 2.6. *Registarska arhitektura*[4]

Arhitektura stoga

U arhitekturi baziranoj na stogu (slika 2.7.), oba operanda i operacije koje se izvode pohranjene su na stogu. Rezultat se tada umeće natrag na stog. Stog se može temeljiti na namjenskim registrima ili mogu biti poseban dio RAM-a



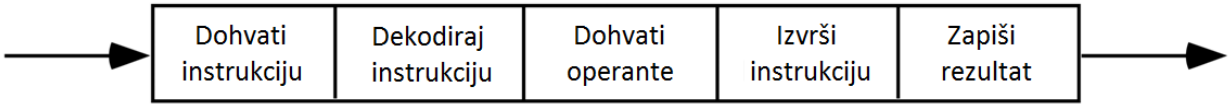
Sl. 2.7. *Arhitektura stoga*[4]

Cjevovodna arhitektura

Mikroupravljač zasnovan na pipeline arhitekturi ilustriran je na slici 2.8. Arhitektura se sastoji od odvojenih sklopovskih podsustava zvanih faze za preuzimanje naredbi iz memorije, dekodiranje instrukcija, dohvaćanje operanada iz memorije i registara, izvršavanje instrukcije, te pisanje rezultata nazad u memoriju. Svaka faza istovremeno obrađuje različite instrukcije. Na

[4]] S. F. Barrett, D. J. Pack, *Microcontrollers Fundamentals for Engineers and Scientists,* Morgan & Claypool, San Rafael, CA, 2006., str. 31

primjer, u pet faza cjevovoda pet uputa istovremeno se obrađuju svaka na različitoj fazi. Tipično, instrukcije u cjevovodnom instrukcijskom sustavu su jednostavne naredbe koje se mogu izvršiti u jednoj fazi. Kompleksnije instrukcije rade se od ovih jednostavnih, manjih instrukcijskih setova.



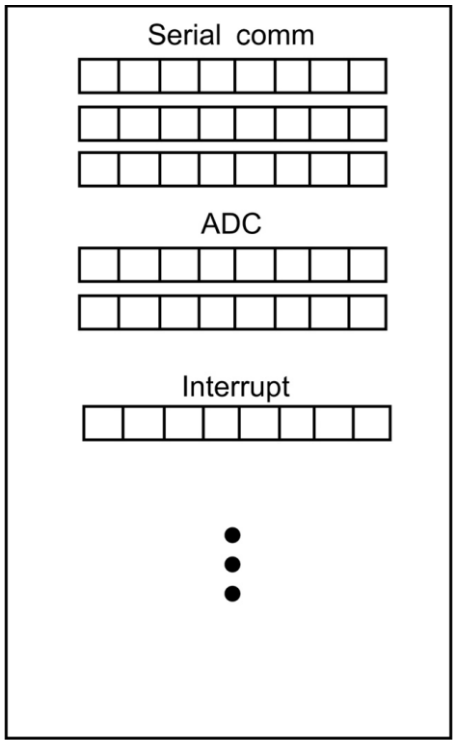
Sl. 2.8. *Cjevovodna arhitektura*[5]

2.3.3. Registri

Većina mikroupravljača imaju cijeli komplement registara nazvanih set registara. Set registara je sučelje između korisnika i ostalih podsustava na mikroupravljaču. Svaki registar sastoji se od kolekcije prekidača, tzv. *flip-flopova*. Svaki *flip-flop* može biti namješten na logičku jedinicu ili logičku nulu te se može opisati kao konfiguralni softverski prekidač. Slanje logičke jedinice *flip-flopu* pali prekidač, a slanje logičke nule gasi prekidač.

Flip-flopovi su kategorizirani po podsustavima kao što je prikazano na slici 2.9. Obično se svi registri povezani sa danim podsustavom grupiraju zajedno. Da bismo konfigurirali određeni podsustav dizajner sustava će odrediti odgovarajuću postavku za svaki bit unutar registra.

[5]] S. F. Barrett, D. J. Pack, *Microcontrollers Fundamentals for Engineers and Scientists,* Morgan & Claypool, San Rafael, CA, 2006., str. 31



Sl. 2.9. *Set registara*[6]

Funkcija i bit svakog registra, pažljivo su definirani u specifičnoj dokumentaciji mikroupravljača. Svi bitovi unutar registra programirani su istovremeno postavljanjem naziva registra (kako je definirano u kompajleru) do željene mape.

Na primjer, za postavljanje registra **SCIBaudRate** na binarnu vrijednost 1010 1110 može se koristiti sljedeća naredba :

SCIBaudRate = 0xAE; *//0x je iskorišteno za oznaku heksadecimalnog broja*

Važno je napomenuti da prevoditelji (engl. c*ompiler*) imaju komplement zaglavnih podataka. Ta zaglavlja podataka sadrže "podatke osobnosti" za određeni mikroupravljač. Naime, oni pružaju vezu između naziva određenog registra korištenog unutar programa i njegove lokacije unutar mikroupravljača.

[6]] S. F. Barrett, D. J. Pack, *Microcontrollers Fundamentals for Engineers and Scientists,* Morgan & Claypool, San Rafael, CA, 2006., str. 33

2.3.4. Sabirnice

Različiti podsustavi unutar mikroupravljača povezani su sa nekoliko različitih sabirnica. Sabirnica je zbirka paralelnih vodiča koji imaju sličnu funkciju. Većina je mikroupravljača opremljena adresnom sabirnicom, sabirnicom podataka te kontrolnom sabirnicom.

Adresna sabirnica

Adresna sabirnica omogućuje vezu između centralnog procesora i memorijskih podsustava na mikroupravljaču. Broj vodiča u adresnoj sabirnici postavlja gornju granicu memorijskih lokacija koje se mogu linearno adresirati mikroupravljačem. Prva adresa u memorijskom podsustavu bit će sve nule, a konačna adresa bit će sve logičke jedinice.

Broj pojedinačno adresabilnih memorijskih adresa može se odrediti računanjem:

$$2^{adresne\ linije} = broj\ adresabilnih\ lokacija \qquad (2\text{-}1)$$

Primjerice, mikroupravljač opremljen 16-bitnom adresnom sabirnicom sposoban je za adresiranje 65 536 (64 kB) odvojenih lokacija. Prva adresa u ovom memorijskom prostoru je $(0000)_{16}$, a zadnja adresa u ovom prostoru bit će $(FFFF)_{16}$.

Podatkovna sabirnica

Podatkovna sabirnica, kako i njezino ime implicira, koristiti se za paralelno usmjeravanje podataka po različitim podsustavima unutar mikroupravljača. Mikroupravljači obično na raspolaganju imaju podatkovne sabirnice širine 4, 8, 16 ili 32 bita. Širina podatkovne staze općenito određuje veličinu podatka koji mikroupravljač može obraditi. Na primjer, najveći cijeli broj bez predznaka koji se može pohraniti u mikroupravljač sa 8-bitnom stazom podataka je 255. Treba naglasiti da 32-bitni mikroupravljač nije bolji mikroupravljač od svog 4-bitnog „kolege“. Podsjetimo se da je primarni cilj dizajnera integriranog sustava odabrati najekonomičniji mikroupravljač koji ostvaruje zahtjeve za specifičan dizajn.

Kontrolna sabirnica

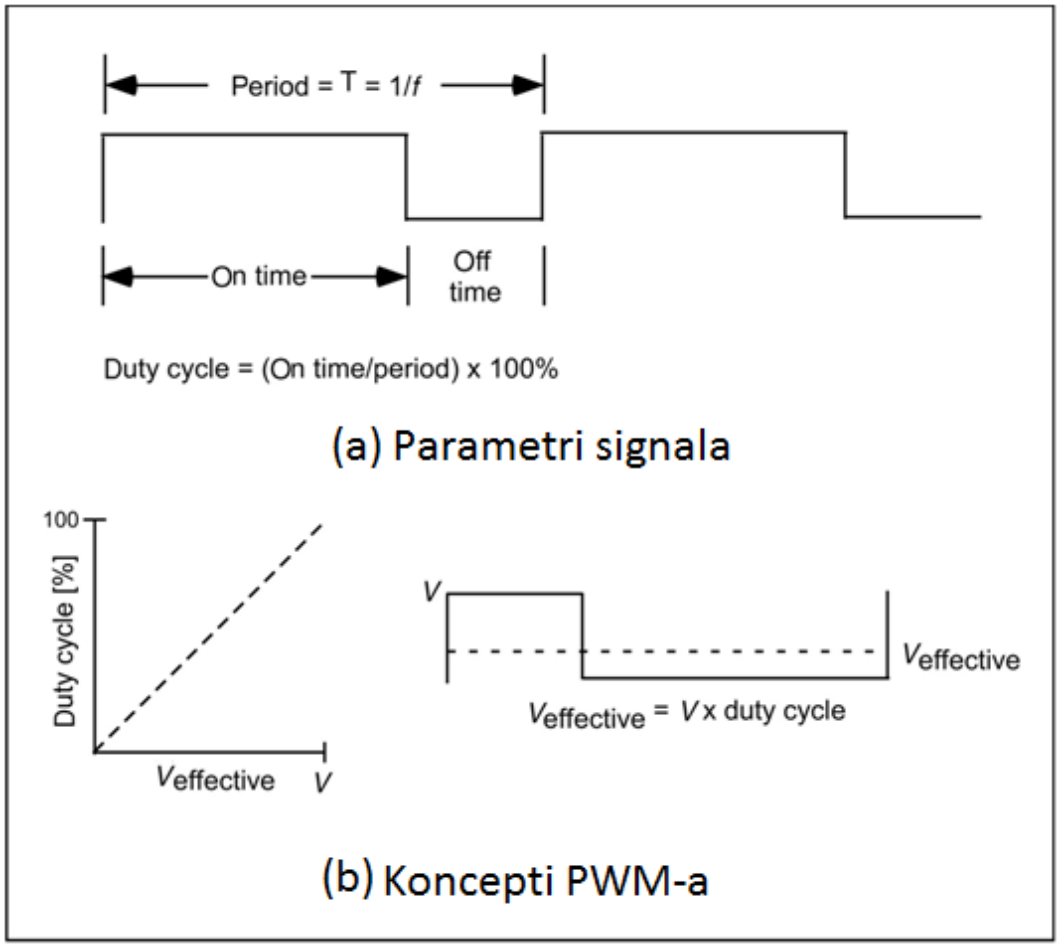
Mikroupravljači su opremljeni stazama za slanje i primanje skupa upravljačkih signala određenih kao kontrolna sabirnica. Ove linije nose upravljačke signale za različite podsustave kroz mikroupravljač. Većina signala za kontrolu su specifični za određeni mikroupravljački integrirani krug (čip). Međutim, oni su često stavljeni na pristup mikroupravljača kako bi vanjski dijelovi mogli biti dodani procesoru. Upravljačke signale daje CPU kao odgovor na programske instrukcije kako bi osigurao dobro izvođenje instrukcije.

2.3.5. Vremenska baza

Kao što je već spomenuto, mikroupravljač je složen sinkroni automat koji odgovara na programske upute u predvidivom dohvati-dekodiraj-izvrši slijedu. Brzina kojom mikroupravljač obavlja svoje sekvence nadzire se vanjskom vremenskom bazom. Vremenska baza mogu biti vanjski kvarcni kristali, programibilni oscilatori, keramički rezonatori ili unutarnja vremenska baza. Neki mikroupravljači su rađeni tako da rade na određenim frekvencijama, kao što su 8 Mhz dok drugi imaju fleksibilno dizajniran rad sa širokim spektrom frekvencija. Trenutno dostupni mikroupravljači rade do približno 50 MHz. Često mikroupravljač upravlja mehaničkim sustavom relativno sporijim od sebe te je za tu vrstu aplikacije potrebno primijeniti sporu vremensku bazu. 32,768 kHz vremenske baze često se koriste u sustavima koji prate 24 sata. Neki mikroupravljači su opremljeni sa unutarnjom vremenskom bazom.

Vremenski podsustav

Glavni izvor takta je preusmjeren kroz mikroupravljač kako bi pružao sinkronizaciju unutar podsustava. Većina mikroupravljača također je opremljena s vremenskim podsustavima. Da bismo bolje shvatile značajke ovog podsustava, pregledat ćemo zajedničke terminologije vezane uz faze ciklusa navedene prema slici 2.10.



Sl. 2.10. *Koncepti vremena*[7]

Frekvencija: Signal frekvencije je broj dovršenih ciklusa u sekundi pri jednom ponavljanju signala, što se izražava mjernom jedinicom *herc* (Hz).

Period: Period je vremenski inkrement u sekundama potreban ponavljajućem signalu kako bi završio jedan ciklus. Razdoblje je recipročna vrijednost frekvencije (T = 1 / f).

Radni ciklus: radni ciklus ukazuje na postotak vremena u kojemu je signal aktivan unutar jednog ciklusa.

PWM (engl. *Pulse width modulation*)**:** PWM signal često se koristi za kontrolu brzine motora. Digitalni PWM signal se pretvara u efektivnu vrijednost istosmjerne struje mehaničkom inercijom motora i filtra niskopropusnih karakteristika induktiviteta indirektno prisutnog u motoru.

[7]] S. F. Barrett, D. J. Pack, *Microcontrollers Fundamentals for Engineers and Scientists,* Morgan & Claypool, San Rafael, CA, 2006., str. 38

2.3.6. Pristupi

Mikroupravljači su opremljeni nizom pristupa koji im omogućuju komunikaciju sa vanjskim svijetom. Često su ti pristupi organizirani kao 8-bitni ulaz/izlaz kao što je prikazano na slici 2.1. Obično je registar pristupa opremljen pratećim registrom za usmjeravanje podataka. Ovo se koristi za određivanje smjera (ulaz ili izlaz) određenog *pina* na pristupu. Iako se pristupi koriste za ulazne i izlazne digitalne signale, mnogi imaju alternativne funkcije kao što su analogno-digitalna pretvorba, serijska komunikacija i mrežna sučelja.

Čak i kada su pristupi opremljeni alternativnim funkcijama, često nema dovoljno vanjskih pinova za osiguravanje pristupa svim dijelovima mikroupravljača. Da bi ublažili taj izazov, neki mikroupravljači su opremljeni vremenski multipleksiranim pristupima. To jednostavno znači da se funkcija pristupa izmjenjuje na vremenski određenom intervalu

2.3.7. Analogno – digitalni pretvornici

Mnogi mikroupravljači opremljeni su podsustavom za analogno-digitalnu pretvorbu (ADC). Ovaj podsustav pretvara stalno varirajuće analogne signale iz vanjskog svijeta u binarni oblik pogodan za mikroupravljač. Ovi pretvornici obično imaju 8 do 10-bitno rješenje, pa se stoga kontinuirani signal pretvara u niz digitalnih slika analognog signala. Dizajner sustava mora odrediti koliko često se ADC pretvorba pokreće za određenu aplikaciju.

2.3.8. Komunikacijski sustav

Kada uzmemo u obzir komunikacije između dva sustava, klasifikacija se sastoji od paralelne i serijske komunikacije. Jednostavno rečeno, paralelna komunikacija koristi višestruke kanale tj. žice sabirnice za slanje i primanje višestrukog toka podataka istovremeno, dok se serijska komunikacija koristi samo jednim tokom podataka koji se šalje i prima.

Najočitija prednost paralelne komunikacije nasuprot serijskoj komunikaciji, pod uvjetom da je komunikacijska stopa jednaka, jest brzina prijenosa podataka. Korištenjem višestrukih broja veza, isti iznos podataka može se poslati i primiti brže, proporcionalno broju veza, nego jednom vezom kod serijske komunikacije. Nedostatak paralelne komunikacije je cijena sklopovlja i programske podrške kako bi se omogućio brz prijenos podataka. Tipično, paralelne tehnike komuniciranja koriste se za male udaljenosti unutar i izvan mikroupravljača. Za komunikacije na daljinu koristi se serijska komunikacija.

3. ODABIR ROBOTA I MIKROUPRAVLJAČA

3.1. Što je robot?

Prema ISO 8373 robot je automatski upravljan, reprogramabilni, višenamjenski manipulator, programabilan u tri ili više osi, koji može biti ili stacionaran ili mobilan za primjene u industrijskoj automatizaciji. No, tema rada su mobilni roboti, a ova definicija se ne može primijeniti na njih jer se za njih postavljaju novi, daleko složeniji zahtjevi.

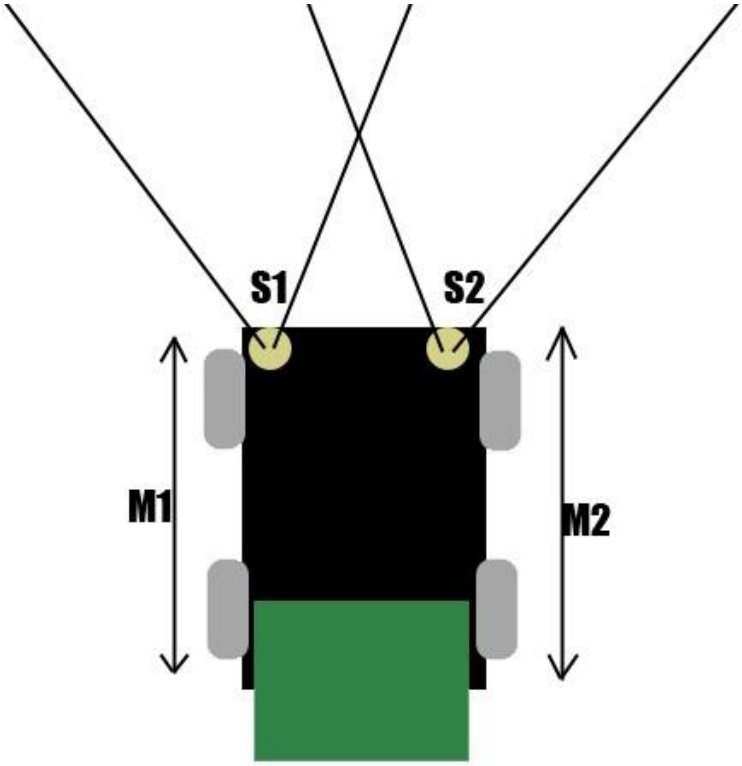
3.2. Mobilni roboti i njihova klasifikacija

Što se zahtjeva od mobilnih robota današnjice? Autonomno gibanje u nepoznatom prostoru, obavljanje zadanog zadatka, interakcija sa drugim robotima ili ljudima u radnom okruženju te sposobnost samoučenja ili inteligentnog zaključivanja. Iz ovoga se može izvesti definicija mobilnog robota. On je mobilan, manipulativan fizički sustav koji se autonomno giba kroz nestrukturirani prostor ostvarujući pritom interakciju s ljudskim bićima ili autonomno obavljajući neki posao umjesto njih.

Mobilni roboti mogu se klasificirati po nekoliko neovisnih značajki, od kojih svaka u velikoj mjeri određuje ključne aspekte njihova sustava upravljanja i navigacije. Najčešća je klasifikacija po prijenosnom mediju. To su zračni, vodeni, kopneni i svemirski roboti.

3.3. Odabir robota

Za objašnjavanje rada mikroupravljača mobilnog robota izrađen je robot koji prati izvor svijetlosti. Robot se nalazi na podvozju sa četiri kotača te je njegov zadatak pratiti izvor svjetlosti pomoću fotosenzora. Za izradu fotosenzora korišteni su fotootpornicima.



Sl. 3.1. *Senzori i princip rada robota*

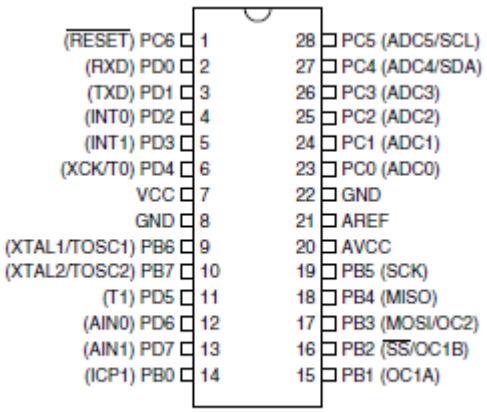
Robot se kreće pomoću dva elektromotora spojena prijenosom na prednje kotače. Dva lijeva kotača spojena su na jedan motor, dok su dva desna spojena na drugi. Za pomicanje i skretanje robota koristi se PWM modulator. Senzori svjetla izrađeni su od fotootpornika te daju analogan signal mikroupravljaču koji ovaj obrađuje te adekvatno njemu prenosi snagu na PWM.

3.4. Odabir mikroupravljača

Pri izradi ovog robota korišten je Atmel ATmega8 AVR mikroupravljač zbog njegove široke dostupnosti. Atmega8 je 28-*pinski* mikroupravljač sa dovoljnim brojem ulaznih i izlaznih jedinica, a ali ne i presloženom građom.

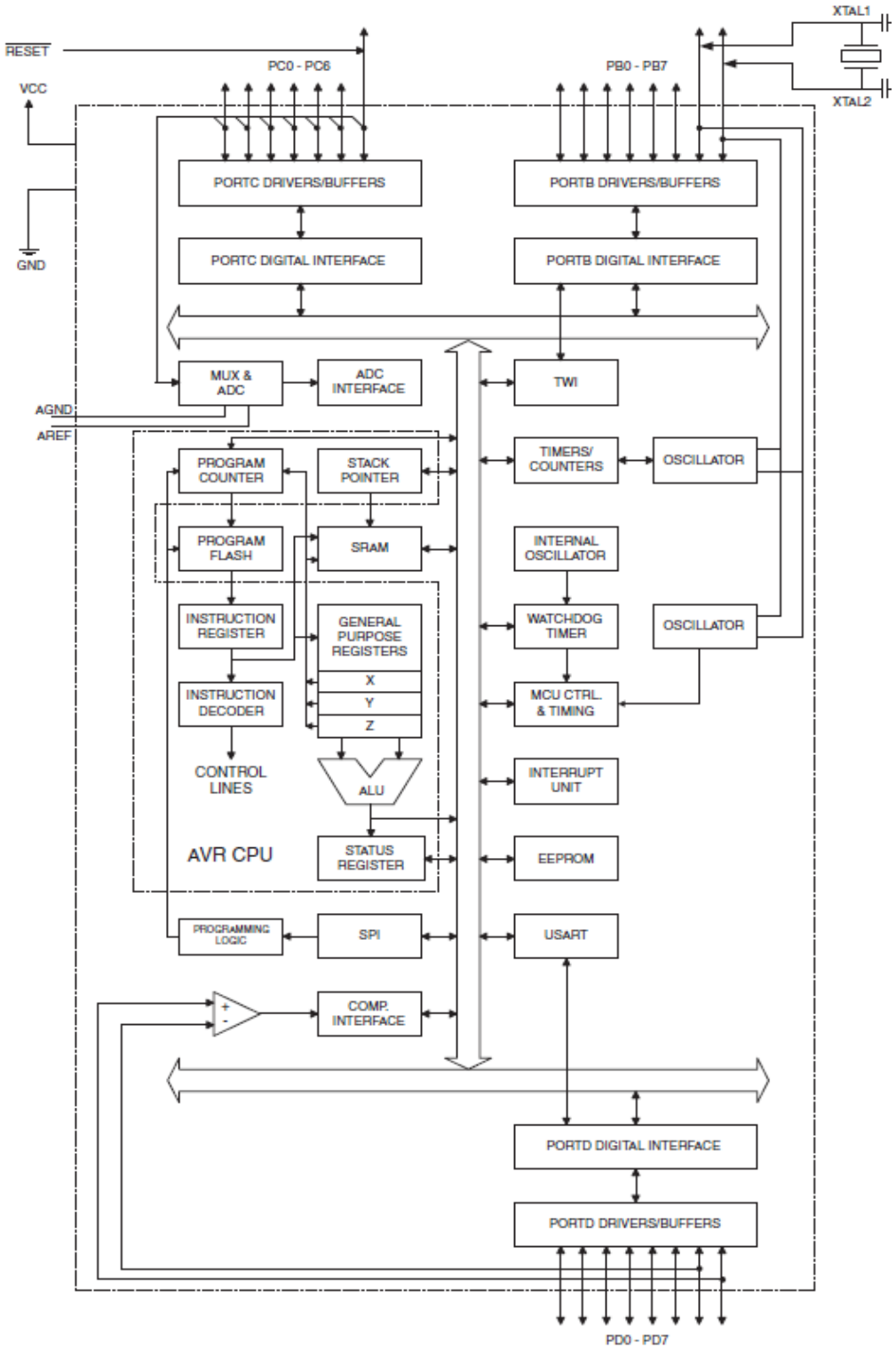
3.4.1. Atmel ATmega8 AVR mikroupravljač

Atmel ATmega8 je 8-bitni AVR mikroupravljač baziran na naprednoj RISC arhitekturi. Ima 8K *flash* memorije sa čitaj-dok-pišeš mogućnosti, 512 bajtova EEPROM-a, 1 kilobajt SRAMA. Također ima 23 I/O linije opće namjene, 32 radna registra opće namjene, tri fleksibilna *clocka/countera* sa usporednim načinom rada, unutarnje i vanjske *interrupte*, serijski programibilan USART te bajt-orientirano dvožično serijsko sučelje. Ima 6 kanalni ADC (osam kanala u TQFP i FN/MLF paketu) sa 10 bitnom preciznošću. Također ima programibilni *watch dog timer* sa unutarnjim oscilatorom, SPI serijski pristup i pet softverskih izbora za načine štednje energije.



Sl. 3.2. *Konfiguracija pinova ATmega8 mikroupravljača*[8]

[8] http://www.linuxfocus.org/common/src2/article365/atmega8.pdf



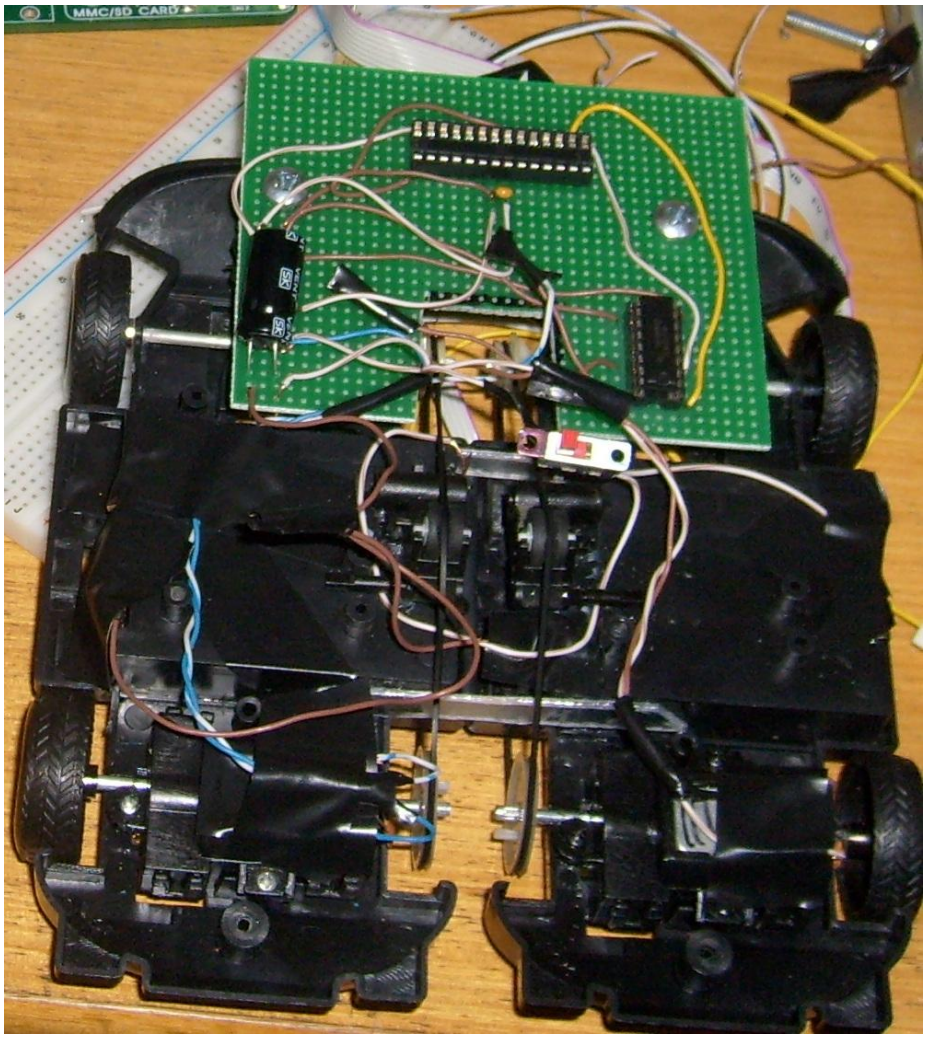
Sl. 3.3. *Blok dijagram ATmega8 mikroupravljača*[9]

[9] [9] http://www.linuxfocus.org/common/src2/article365/atmega8.pdf

4. IZRADA ROBOTA I PROGRAMIRANJE MIKROUPRAVLJAČA

4.1. Izrada robota

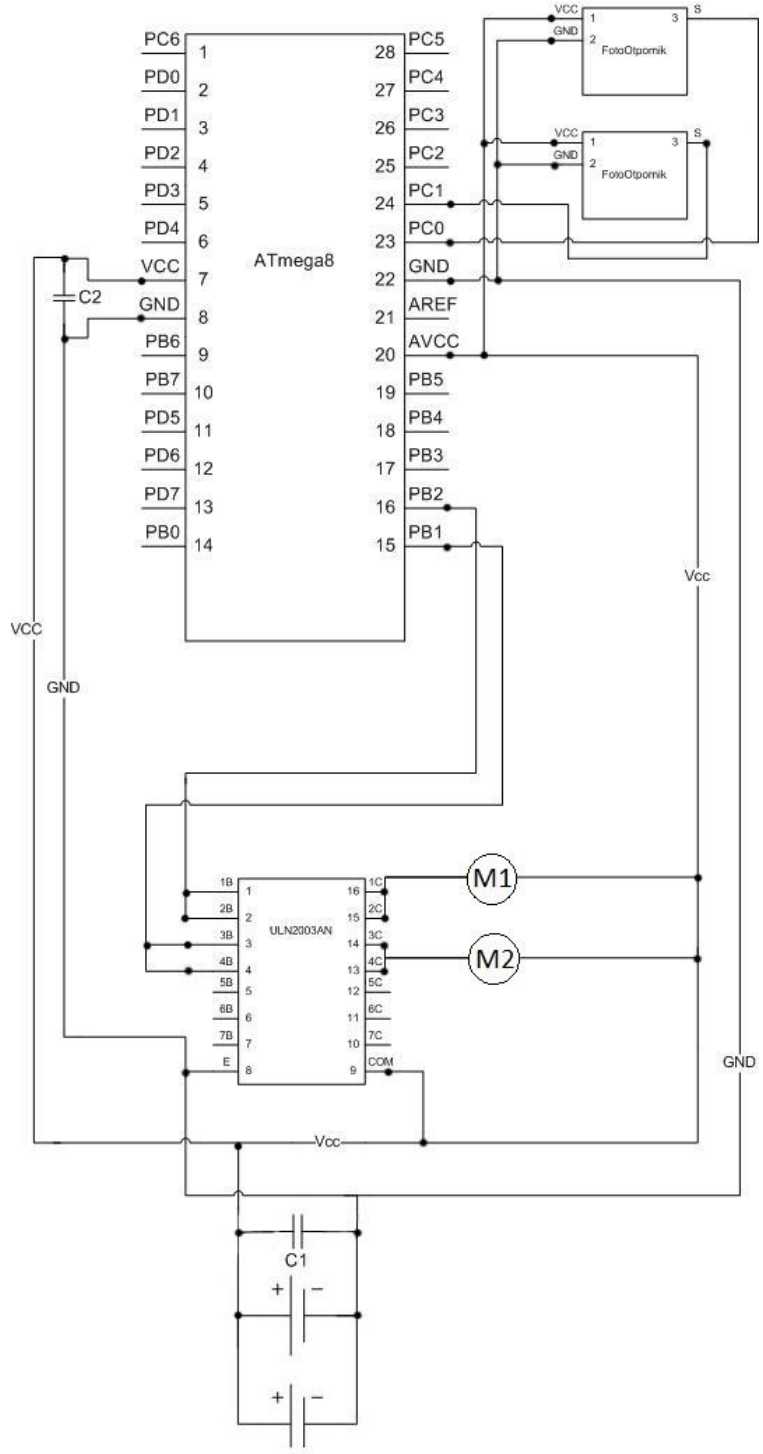
Robot je jednostavne građe, izrađen od dva manja automobila igračke na daljinsko upravljanje. Podvozje robota izrađeno je spajanjem dva automobila, svakog sa jednom stranom kotača. Srednji prijenos koji prenosi snagu zadnjih, pogonskih kotača na prednje, izrađen je od gumica. Sve to spojeno je i učvršćeno plastikama.



Sl. 4.1. *Izrađeno podvozje robota*

4.1.1. Spoj elektroničkih uređaja na robotu

Elektroničke komponente su spojene na *protoboardu*, a potom zalemljene na robota.

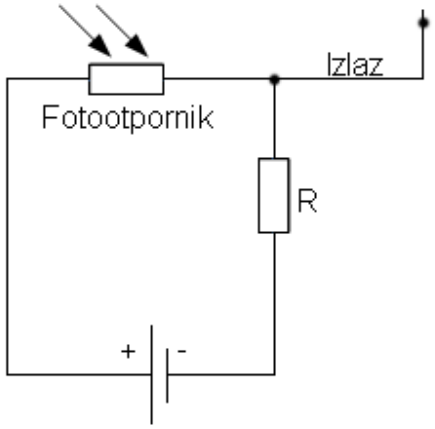


Sl. *4.2. Shema spoja elektroničkih komponenti*

Robot je spojen na dva izvora istosmjerne struje (slika 4.2.). Izvor je izveden sa tri 1.5 V baterije spojene u seriju. Dva bloka od tri serijski spojene baterije spojena su paralelno da bi se dobio stalni napon od 4,5 V. Na izvor napajanja spojen je mikroupravljač. Na izlazne PWM pristupe B1 i B2 spojeni su elektromotori preko tranzistorskog sklopa ULN2004AN. Pristupi su dvostruko spojeni na tranzistore zbog boljeg prolaska struje kroz tranzistor. Na 4 izlaza, po dva para, spojeni su elektromotori koji su na drugoj strani spojeni na izvor. Pristupi PC1 i PC0, koji su inače ADC pristupi spojeni su na naš fotosenzor izrađen od fotootpornika. Fotooptornici daju analogni signal na ulaze PC1 i PC0 koji tada dobivene rezultate pretvaraju u digitalni signal sa skalom od 0 do 255. Mikroupravljač tada uspoređuje PC1 i PC0 ulaze te adekvatno njima određuje snagu na PWM-u.

4.2. Izrada fotosenzora fotootpornicima

Kako bi robot mogao „vidjeti“ te pratiti svjetlo, trebali su se izraditi fotosenzore. Fotosenzori, koji robotu služe za za „vid“, izrađeni su od fotootpornika(opisano na slici 4.3.).



Sl. 4.3. *Fotosenzor izrađen fotootpornicima*

Preko izvora napajanja na Vcc ili pozitivnu stranu izvora spojen je fotootpornik čiji se otpor mijenja promjenom osvjetljenja u prostoriji. Fotootpornik je spojen serijski sa običnim otpornikom R jačine 1 MOhm te na minus pol izvora. Između dva otpornika nalazi se *output* ili izlazni signal koji je spojen na PC0/PC1 pristup na mikroupravljaču.

4.3. Upravljanje elektromotorima pomoću PWM-a

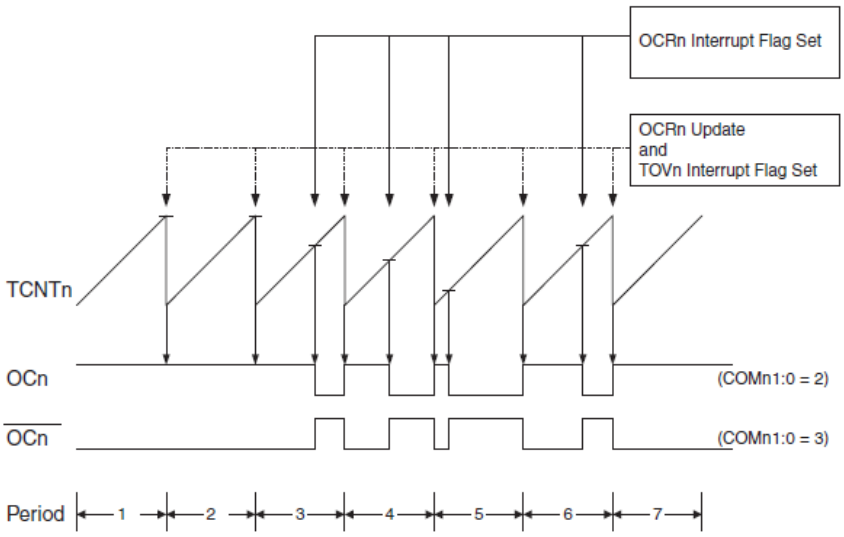
PWM (engl. *Pulse Width Modulation*) je izraz koji se koristi za opisivanje generiranja analognog izlaznog signala koristeći digitalni signal. On se obično koristi za kontrolu prosjeka snage na opterećenje u kontroli vrtnje elektromotora. Također se može koristiti za generiranje kontinuiranog varijabilnog analognog izlaza bez korištenja bilo kojeg drugog integriranog sklopa za zaglađivanje PWM signala.

Fast PWM ili brzi PWM, koji se nalazi u Atmega8 mikroupravljaču, koristio sam prilikom izrade upravljanja elektromotorima. Brzi *Pulse Width Modulation* ili brzi PWM mod (WGM21: 0 = 3) osigurava rad sa visokim frekvencijama. Brzi PWM razlikuje se od ostalih PWM-a zbog nagibnog načina rada. Brojač broji od dna do maksimuma te ponovno kreće od dna što daje pilasti signal. U neizmjeničnom modu usporedbe *Output Compare* (OC2) čisti se prilikom usporedbene jednakosti između TCNT2 I OCR2 te se postavlja ponovno na dno. U izmjeničnom modu usporedbe izlaz je namješten na usporedbenu jednakost i *restartan* na dno.

U brzom PWM modu, brojač raste do protuvrijednosti koja odgovara maksimalnoj vrijednosti. Brojač se tada čisti prilikom postavljanja *clocka* u sljedeći ciklus. Vremenski dijagram za brzi PWM način rada prikazan je na slici 4.4. TCNT2 vrijednost je vremenski dijagram prikazan kao histogram za ilustraciju jedno-padnog rada. Dijagram uključuje neizmjenični i izmjenični PWM izlaz. Mala horizontalna linija na TCNT2 padovima označava usporedbu između OCR2 i TCNT2.

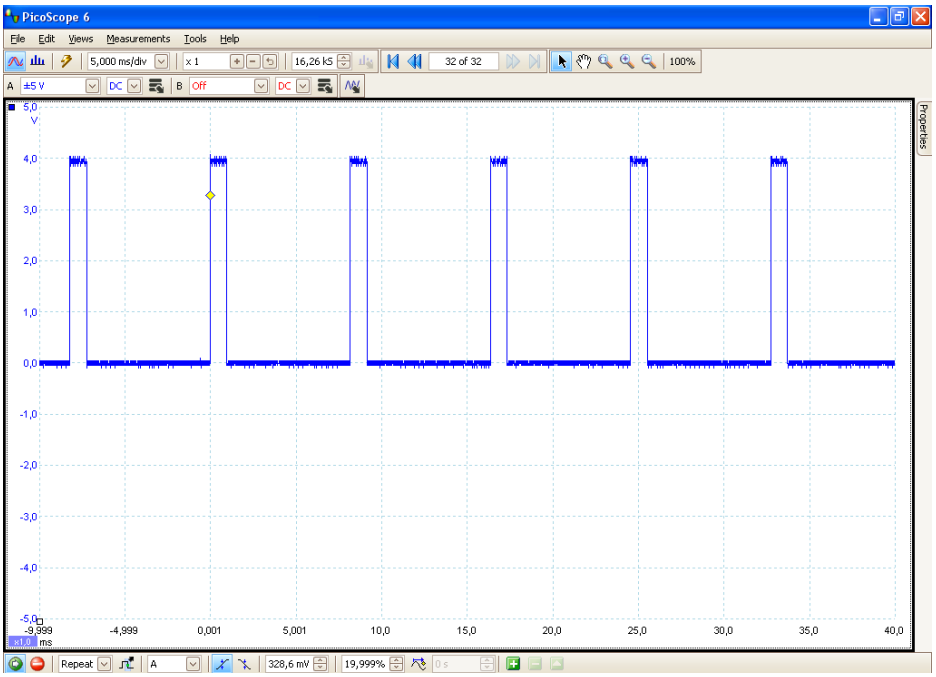
Frekvencija PWM-a za izlaz računa se prema formuli:

$$f_{OCnPWM} = \frac{f_{clk\_I/O}}{N \cdot 256} \tag{4-1}$$



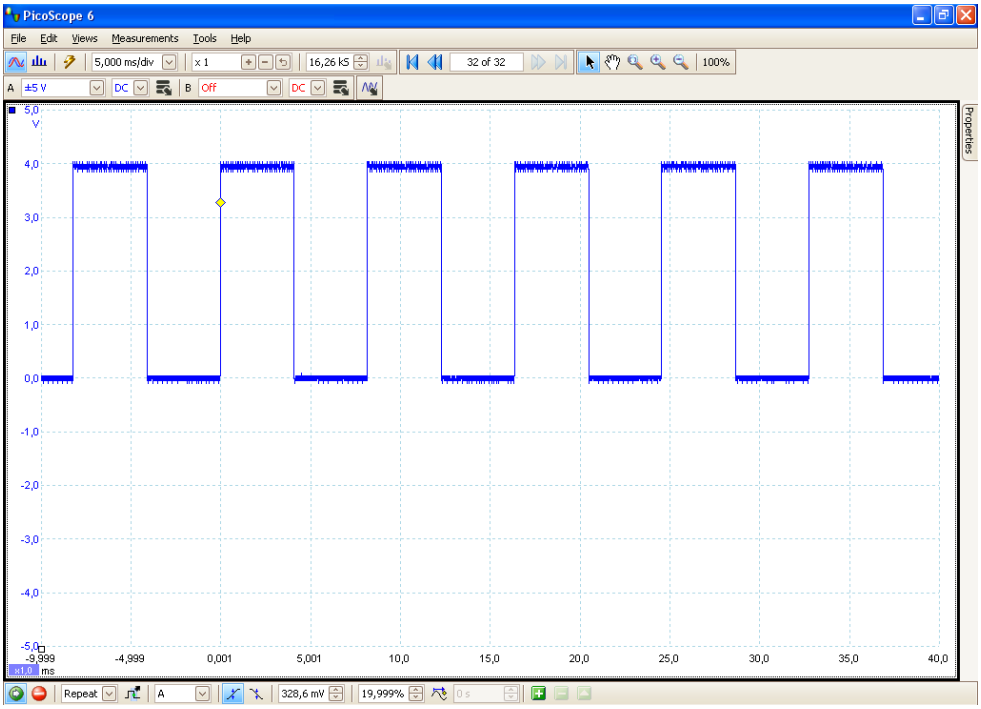
Sl. 4.4. *Vremenski dijagram brzog PWM moda*[10]

Napravljeno je i nekoliko mjerenja sa statičkim vrijednostima OCR-a te je snimljen izgled PWM signala osciloskopom.

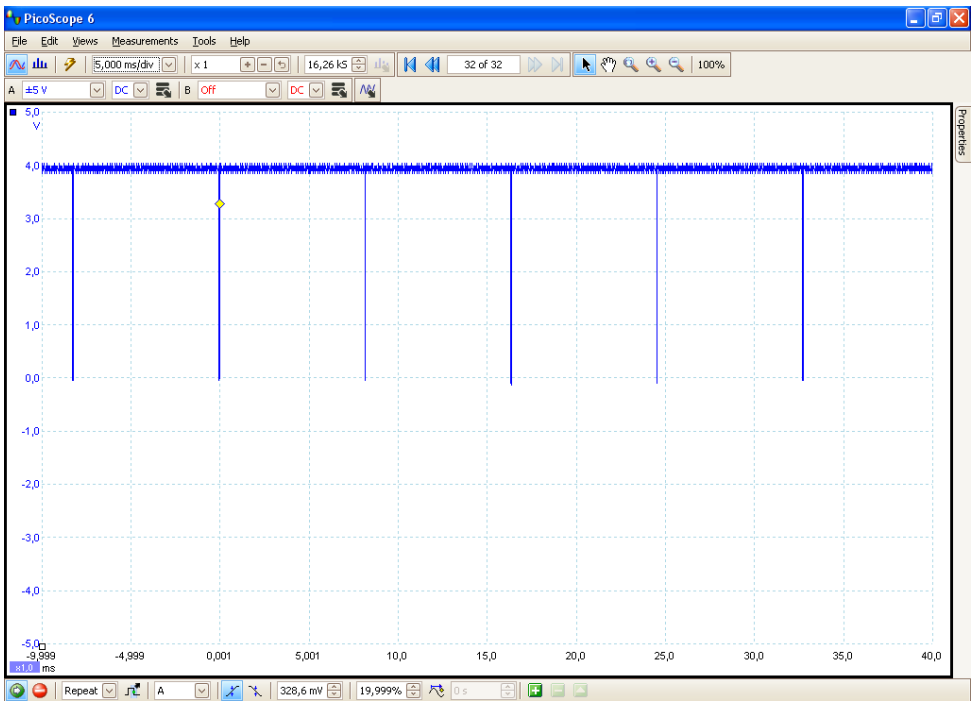


Sl. 4.5.*Izgled PWM signala sa OCR-om vrijednosti 30*

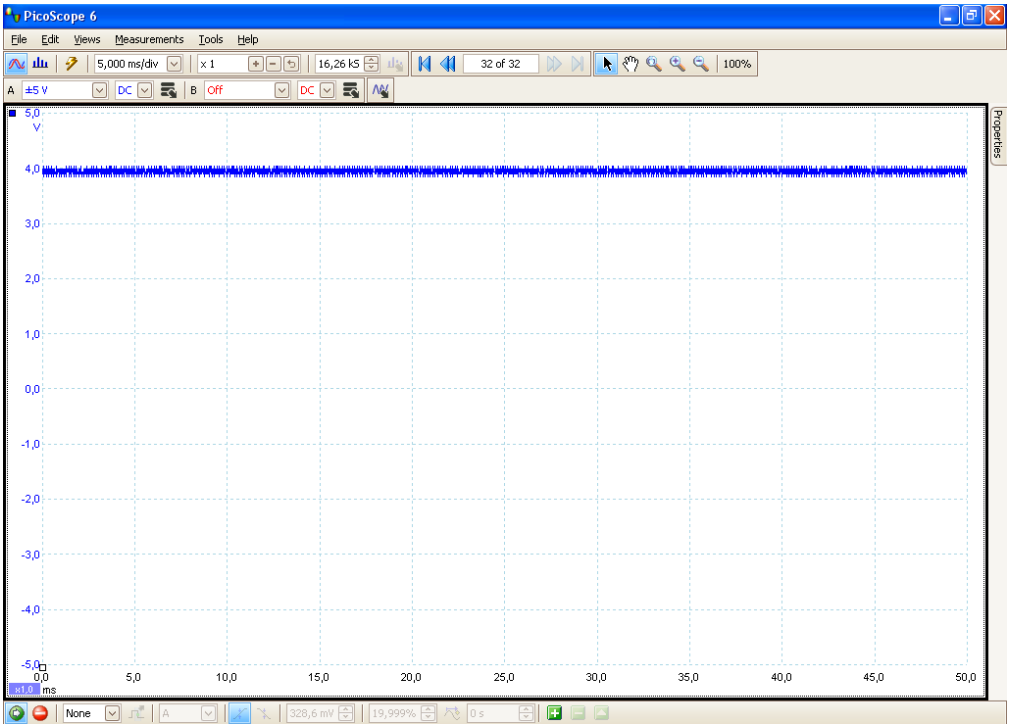
[10] http://www.linuxfocus.org/common/src2/article365/atmega8.pdf



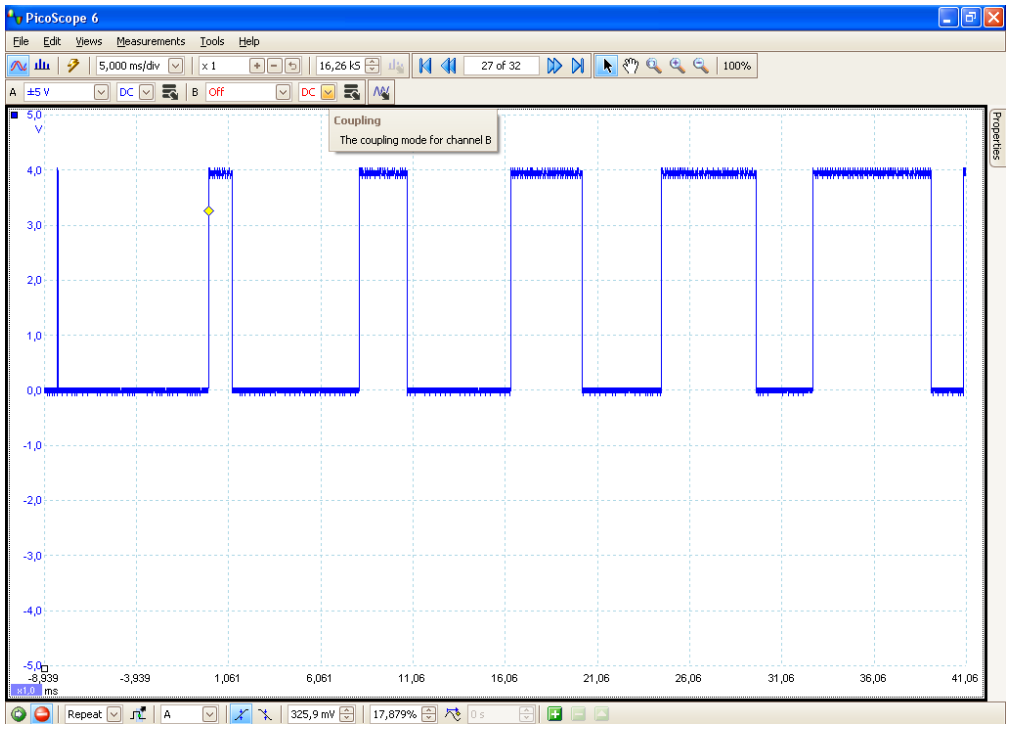
Sl. 4.6. *Izgled PWM signala sa OCR-om vrijednosti 128*



Sl. 4.7. *Izgled PWM signala sa OCR-om vrijednosti 254*



Sl. 4.8. *Izgled PWM signala sa OCR-om vrijednosti 255*

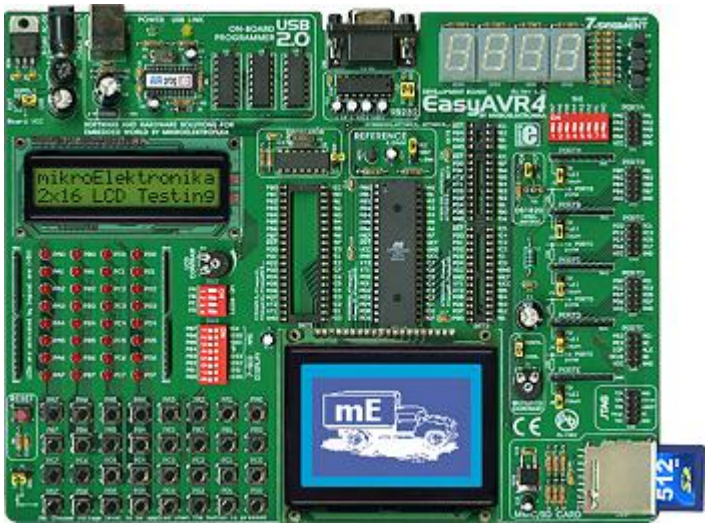


Sl. 4.9. *Izgled PWM signala sa OCR-om koji ima povećanje od +40*

Kako se vidi iz slika mjerenja izlaznog signala PWM-a prilikom vrijednosti od 30 do 255 može se zaključiti da se povećanjem OCR registra, tj. njegove vrijednosti, povećava snaga izlaznog signala koja se manifestira kao širina signala. Na slici 4.9 vidi se povećanje, tj. proširivanje signala prilikom povećanja OCR registra za 40. Budući da takt dolazi od kvarcnog kristala, svakim sljedećim ciklusom OCR se poveća za 40 te se automatski širi i veličina PWM signala.

4.4. Programiranje mikroupravljača

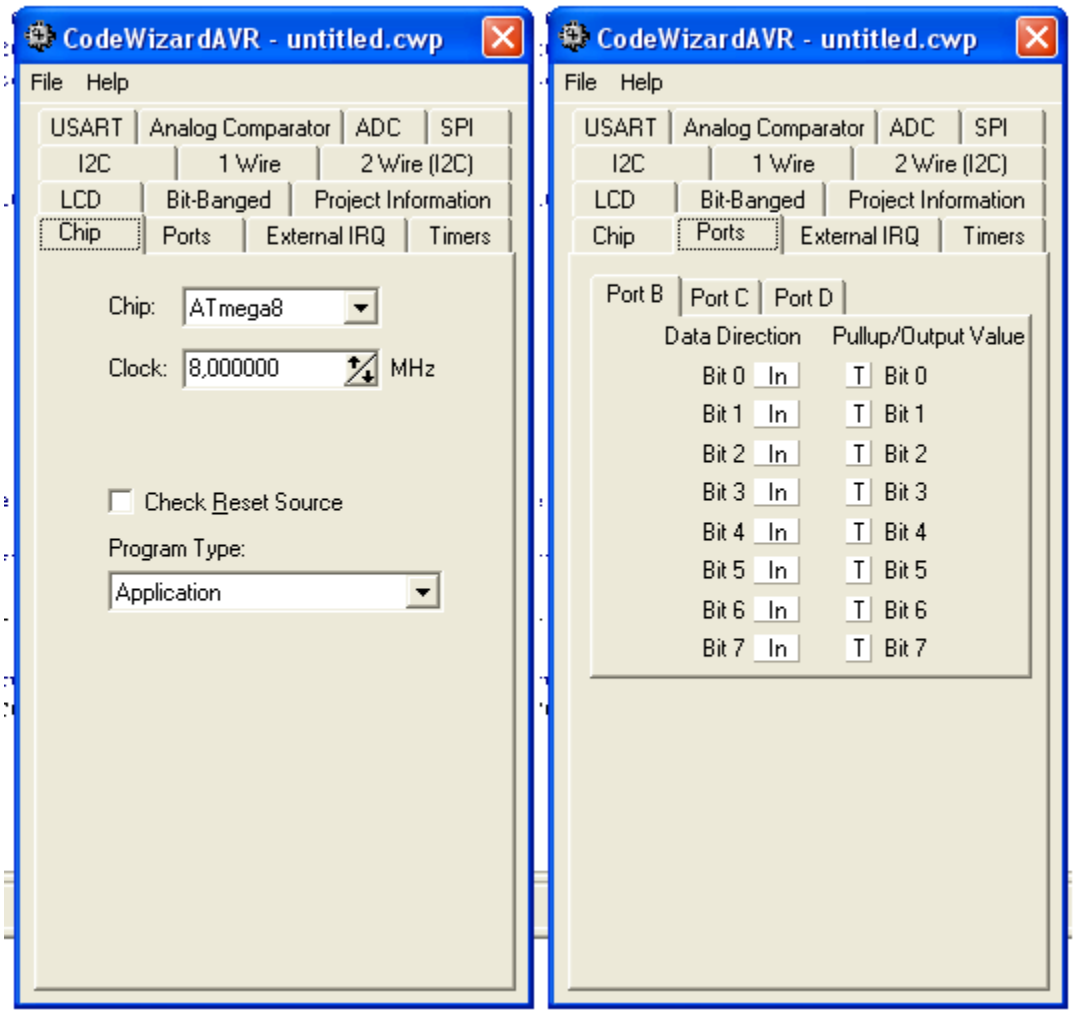
Atmel ATmega8 mikroupravljač koristi AVR instrukcijski set. Programiranje mikroupravljača urađeno je pomoću *development boarda* EASYAVR4 proizvedenog od tvrtke MikroElektronika Beograd (slika. 4.10.).



Sl 4.10. *Izgled upravljačke ploče za AVR programiranje*

Razvojna ploča spaja se na računalo preko USB sučelja te se pomoću AVRFlash aplikacije programski kod stavlja na mikroupravljač. Programski kod napravljen je u C sintaksi pomoću programa CodeVisionAVR C Compiler. Inicijalizacija pristupa na ulazni ili izlazni rad namješta se

pomoću čarobnjaka unutar samog programa, što je prikazano na slici 4.11., isto kao i namještanje osnovnih podataka o mikroupravljaču kao što su takt rada, vrsta programa itd.



Sl. 4.11. *Izgled čarobnjaka unutar programa CodeVision AVR*

Punjenje i pražnjenje mikroupravljača obavljeno je pomoću programa AVRflash u kojemu se također namješta takt mikroupravljača, korištenje unutarnjeg ili vanjskog oscilatora, isto kao i učitavanje koda i slanje izravno na mikroupravljač preko upravljačke ploče.

4.5. Ispis koda za mikroupravljač

```
/*****************************************************
Chip type           : ATmega8
Program type        : Application
Clock frequency     : 8,000000 MHz
Memory model        : Small
External SRAM size  : 0
Data Stack size     : 256
*****************************************************/
#include <mega8.h>  // Programski kod uključuje mega8.h library za mikroupravljač Atmega8
#define ADC_VREF_TYPE 0x20  //definiranje i inicijalizacija ADC ulaza
#define FIRST_ADC_INPUT 0  //definiranje i inicijalizacija ADC ulaza
#define LAST_ADC_INPUT 1  //definiranje i inicijalizacija ADC ulaza
unsigned char adc_data[LAST_ADC_INPUT-FIRST_ADC_INPUT+1];
//ADC interupt  rutina sa auto skeniranjem ulaza
interrupt [ADC_INT] void adc_isr(void)  {
    register static unsigned char input_index=0;
   // Pročitaj 8 najbitnijih bitova rezultata  analogno-digitalne pretvorbe
    adc_data[input_index]=ADCH;
```

```
   // odabiranje sljedećeg ADC ulaza
   if (++input_index > (LAST_ADC_INPUT-FIRST_ADC_INPUT))
      input_index=0;
    ADMUX=(FIRST_ADC_INPUT | (ADC_VREF_TYPE & 0xff))+input_index;
   // Počni analogno–digitalnu konverziju
   ADCSRA|=0x40;
}
// Rutina interupta za overflow Timera 1
interrupt [TIM1_OVF] void timer1_ovf_isr(void)
{
   //OCR1A = 240;
   //OCR1B = 150;
}
// Deklaracija globalnih varijabli
void main(void)
{
// Deklaracija lokalnih varijabli
//Inicijalizacija Ulazno/Izlaznih pristupa
// Inicijalizacija pristupa B
// Func7=In Func6=In Func5=In Func4=In Func3=Out Func2=Out Func1=Out Func0=Out
// State7=T State6=T State5=T State4=T State3=0 State2=0 State1=0 State0=0
PORTB=0x00;
```

DDRB=0x0F;

// Inicijalizacija pristupa C

// Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In

// State6=T State5=T State4=T State3=T State2=T State1=T State0=T

PORTC=0x00;

DDRC=0x00;

// Inicijalizacija pristupa D

// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In

// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T

PORTD=0x00;

DDRD=0x00;

// Timer/Counter 0 inicijaizacija

// Clock source: System Clock

// Clock value: Timer 0 Stopped

TCCR0=0x00;

TCNT0=0x00;

// Timer/Counter 1 initialization

// Clock source: System Clock

// Clock value: 31,250 kHz

// Mode: Fast PWM top=00FFh

// OC1A output: Non-Inv.

// OC1B output: Non-Inv.

// Noise Canceler: Off

// Input Capture on Falling Edge

// Timer 1 Overflow Interrupt: On

// Input Capture Interrupt: Off

// Compare A Match Interrupt: Off

// Compare B Match Interrupt: Off

TCCR1A=0xA1;

TCCR1B=0x0C;

TCNT1H=0x00;

TCNT1L=0x00;

ICR1H=0x00;

ICR1L=0x00;

OCR1AH=0x00;

OCR1AL=0x00;

OCR1BH=0x00;

OCR1BL=0x00;

// Timer/Counter 2 initialization

// Clock source: System Clock

// Clock value: Timer 2 Stopped

// Mode: Normal top=FFh

// OC2 output: Disconnected

ASSR=0x00;

TCCR2=0x00;

TCNT2=0x00;

OCR2=0x00;

// Inicijalizacija vanjskih prekida

// INT0: Off

// INT1: Off

MCUCR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization

TIMSK=0x04;

// Analog Comparator initialization

// Analog Comparator: Off

// Analog Comparator Input Capture by Timer/Counter 1: Off

ACSR=0x80;

SFIOR=0x00;

// ADC initialization

// ADC Clock frequency: 62,500 kHz

// ADC Voltage Reference: AREF pin

// Only the 8 most significant bits of

// the AD conversion result are used

```
ADMUX=FIRST_ADC_INPUT | (ADC_VREF_TYPE & 0xff);

ADCSRA=0xCF;

// dozvola globalnih prekida

// programski kod koji izjednačuje adc_data ulaz sa PC0 i PC1 pristupa na PWM OCR1A i OCR1B izlaz

#asm("sei")

while (1)

    {
       OCR1A = adc_data[0];
       OCR1B = adc_data[1];
    };
}
```

5. ZAKLJUČAK

Cilj ovoga rada bio je pobliže objasniti način rada i implementaciju mikroupravljača u mobilno robotici. Taj cilj ostvaren je izradom mobilnog kopnenog robota sa četiri kotača upravljanog Atmel ATmega 8 AVR mikroupravljačem i fotosenzorima izvedenim pomoću fotootpornika.

Nakon što je napravljena shema spoja elektroničkog kruga, određeni su pristupi ulaza i izlaza na mikroupravljaču isto kao i na tranzistorskom sklopu i elektromotorima. Mjerenjem su određene razine rada PWM-a kako bi se odredila brzina kretanja i upravljanje robotom. Pobliže je objašnjeno funkcioniranje upravljanja elektromotorima, koji su napajani vanjskim izvorom jer mikroupravljač nije imao dovoljno napona na nožici za upravljanje elektromotorima. Neke, skuplje izvedbe mikroupravljača, elektromotore mogu i samostalno napajati.

Kada je izrađen kompletan sklopovski dio, isprogramiran je mikroupravljač koji je radio omjer jačine PWM-a ovisno o intenzitetu svjetlosti na fotosenzorima. Fotosenzori su zbog veće osjetljivosti izvedeni od fotootpornika, ali su mogli biti izrađeni i od fotodioda ili fototranzistora.

U ovom radu dan je primjer izrade i implementacije mikroupravljača za mobilne robote pomoću jednostavnog 8-bitnog mikroupravljača, kako bi se na jednostavan način objasnili osnovni principi rada. Postoje i jači mikroupravljači koji se koriste u modernoj tehnologiji od automobila i robota do dječjih igračaka, te svi rade na sličan, doduše malo složeniji način.

LITERATURA

[1] S. F. Barrett, D. J. Pack, *Microcontrollers Fundamentals for Engineers and Scientists,* Morgan & Claypool, San Rafael, CA, 2006.

[2] D. Andrić, N. Matić, *PIC mikrokontroleri*, Agencija APC, Beograd, 2000.

[3] Atmel AVR Microcontroller Primer: Programming and Interfacing Morgan & Claypool, San Rafael, CA, 2008.

[4] http://www.linuxfocus.org/common/src2/article365/atmega8.pdf

SAŽETAK

Opisani su opći dijelovi mikroupravljača i način njihovog funkcioniranja, te posebno princip upravljanja mobilnim robotima pomoću mikroupravljača. Princip rada objašnjen je na mobilnom kopnenom robotu sa četiri kotača upravljanom Atmel ATmega 8 mikroupravljačem i fotosenzorima. Nakon izrade sheme spoja elektroničkog kruga, opisana je praktična izvedba robota pokretanog elektromotorima i upravljanog fotosenzorima izvedenim od fotootpornika. Osim postupka izrade sklopovskog dijela, prikazan je i postupak programiranja Atmel ATmega8 mikroupravljača za upravljanje odabranim mobilnim robotom. Priložen je i detaljno objašnjen ispis koda za navedeni mikroupravljač koji je korišten za upravljanje opisanom sklopovskom izvedbom mobilnog robota.

Ključne riječi: *mobilni robot, mikroupravljač, Atmel ATmega 8, Pulse Width Modulation*

ABSTRACT

MICRONTROLLER SYSTEM OF A SIMPLE MOBILE ROBOT

General parts of microcontroller and the way they function are described, and in particular the principle of managing mobile robots using a microcontroller. The principle of work is explained on the land mobile four-wheel robot controlled via Atmel Atmega 8 microcontroller and photosensors. After creating an electronic circuit diagram, practical robot design driven by electric motors and controlled by photosensors derived from the photoresistors is described. Besides of the process of hardware developing, programming process of Atmel Atmega8 microcontroller for managing the selected mobile robot is shown. Detailed code explanation for this particular microcontroller which is used to manage the described hardware version of mobile robot is attached.

Key words: *mobile robot, microcontroller, Atmel ATmega 8, Pulse Width Modulation*

ŽIVOTOPIS

Rođen sam 6. ožujka 1987. godine u Osijeku i od rođenja živim u Valpovu. Osnovnu školu i opću gimnaziju završio sam u Valpovu. Završetkom srednje škole upisao sam preddiplomski studij računarstva na Elektrotehničkom fakultetu Sveučilišta J. J. Strossmayera u Osijeku.