DRUGI ZADATAK IZ PRIMENJENE ELEKTRONIKE

NAZIV PROJEKTA:

Autonomno kretanje robota uz izbegavanje prepreka.

TEKST ZADATKA:

Robot kreće sa startne linije, zaobilazi sve prepreke na svom putu i vrati se na startnu liniju. Prepreke na koje robot nailazi su nasumično postavljene.

MENTOR PROJEKTA:

Dr Vladimir Rajs

MSc Marko Vasiljević-Toskić

MSc Milan Bodić

PROJEKAT IZRADILI:

Petar Ubavić EE156/2019

Jovan Ikić EE195/2019

Đorđe Drozgović EE43/2018

Ilija Marinković EE25/2019

DATUM ODBRANE PROJEKTA:

05.05.2023.

1. **Uvod……………………………………………………………………………………….…. 3**
   1. Mikrokontroler dsPIC30f4013……………………………………………………………3
   2. Programski paket MPLABX……………………………………………………………...5
   3. Konfiguracija portova mikrokontrolera dsPIC30f4013…………………………………..5
2. **Rad sa serijskim UART portovima………………………………………………………….6**

2.1 Asinhrona serijska komunikacija…………………………………………………………..6

2.2 Brzina serijske komunikacije………………………………………………………………7

2.3 Naponski nivoi……………………………………………………………………………. 7

2.4 Konvertor napona………………………………………………………………………….7

**3. Autonomno kretanje robota…………………………………………………………………..8**

3.1 Blok dijagram podsistema…………………………………………………………………8

3.2 Kolo za upravljanje motorima……………………………………………………………..9

3.3 Pinovi za programiranje…………………………………………………………………..11

3.4 Napajanje…………………………………………………………………………………12

3.5 Kolo sa diodama………………………………………………………………………….13

3.6 BT prijemnik……………………………………………………………………………..14

3.7 Senzori……………………………………………………………………………………15

3.7.1 HC-SR04…………………………………………………………………………...15

3.7.2 IR senzor daljine GP2Y0A21YK…………………………………………………..19

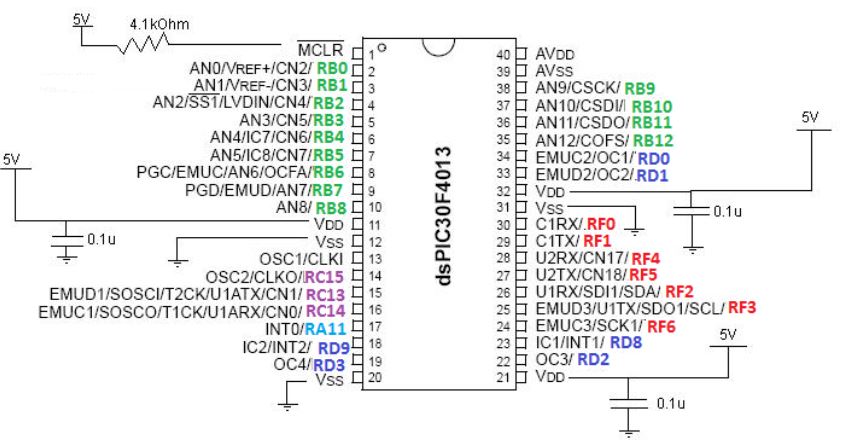
# **Uvod**

Za izradu drugog zadatka iz Primenjene elektronike potrebno je koristiti mikrokontroler dsPIC30F4013 koji će se nalaziti na izrađenoj pločici tj. PCB-u (eng. ***P****rinted* ***C****ircuit* ***B****oard)* uključujući i sve potrebne komponente za pravilan rad samog kontrolera kao i ostalih podsistema. Prilikom izrade odlučili smo se za izradu pločice foto-postupkom. O samom procesu izrade pločice neće biti reči, ali u narednim poglavljima će biti opisan svaki od podsistema. Temu projekta okvirno je zadao profesor dr Vladimir Rajs, te je ideja napraviti autonomno kretanje robota, tj. u našem slučaju tenkića,

# **1.1 Mikrokontroler dsPIC30F4013**

Mikrokontroler je uređaj opšte namene, koji pribavlja podatke, obavlja ograničenu obradu nad tim podacima, i upravlja svojim okruženjem na osnovu rezultata izračunavanja. Mikrokontroler u toku svog rada koristi fiksni program koji je smešten u ROM-u i koji se ne menja u toku životnog veka sistema. Da bi se kompletirao mikroračunarski sistem pored mikroprocesora potrebno je dodati ROM, RAM memorijske dekodere, oscilator, određeni broj ulazno-izlaznih uređaja, kakvi su paralelni i serijski portovi za podatke, A/D i D/A konvertori i drugo.

Na slici 1 je prikazan raspored priključaka mikrokontrolera dsPIC30F4013:



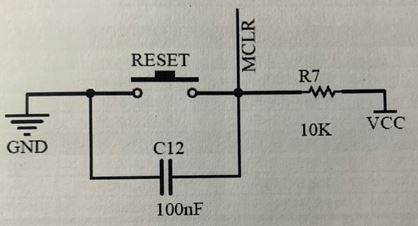
1. Raspored priključaka mikrokontrolera

Mikrokontroler dsPIC30f4013 ima ukupno 40 priključaka (slika 1). Najčešće u varijanti DIP 40 kućišta, ali se može sresti i u SMD kućištu. Skraćenica DIP je nastala od početnih slova engleskih reči *„Dual In-Line Package“.* Pin je žargonski izraz za priključak ili nožicu elektronskog kola, ili jasnije rečeno, spoljni izvod.

Neke od osnovnih karakteristika mikrokontrolera dsPIC30F4013:

* + Radni napon od 2.5V do 5V
  + Programska memorija kapaciteta 48KB
  + Memorije za podatke tipa RAM kapaciteta 2KB
  + Memorije za podatke EEPROM tipa kapaciteta 1KB
  + Prekidi (do 33 izvora prekida)
  + U/I portovi: B,C,D,E,F
  + Pet tajmera:
    - *Timer1:* 16-bitni tajmer/brojač događaja
    - *Timer2/3:* 32-bit tajmer/brojač događaja
    - *Timer4/5:* 32-bit tajmer/brojač događaja
  + 12-to bitni Analogno-Digitalni (A/D) konvertor
  + Komunikacije: SPI, USART, I2C
  + Poseduje modul za 4 *PWM* izlaza

Za napajanje koristi se stabilisan napon u opsegu od 2,5V do 5V. Taj napon se dovodi na priključke sa oznakom VDD, dok je masa dovedena na pinove s oznakom VSS. Prvi pin sa oznakom MCLR koristi se za reset kontrolera. Pošto je oznaka nadvučena, to znači da je reset aktivan na logičku nulu. Prilikom reseta, memorijske lokacije RAM-a se ne resetuju. Za razliku od njih registri za specijalne funkcije se resetuju na polazno stanje. Na slici 2 prikazano je reset kolo za mikrokontroler dsPIC30F4013.



2.Reset kolo za mikrokontroler

Priključci mikrokontrolera 13 i 14 pod nazivom OSC1 i OSC2 služe za priključivanje komponenti na oscilatorno kolo. Primarna frekvencija oscilatora se može uvećati pomoću množača frekvencije PLL. Osnovna frekvencija se može pomnožiti 4, 8 i 16 puta. Primer podešavanja frekvencije napisan u programskom jeziku C za kompajler XC16, gde se frekvencija oscilatora množi 4 puta:

**\_FOSC(CSW\_FSCM\_OFF & XT\_PLL4)**

U ovom slučaju frekvencija instrukcionog takta je ista kao frekvencija oscilatora, jer jedna instrukcija u mikrokontroleru traje 4 perioda takt signala.

Ako se želi koristiti interni oscilator mikrokontrolera čija je frekvencija 7.3728 MHz, podešavanja u programu su sledeća:

**\_FOSC(CSW\_FSCM\_OFF & FRC\_PLL4)**

# **1.2 Programski paket MPLABX**

Programski jezik *MPLABX* je *Microchip-*ovo razvojno okruženje namenjeno PIC mikrokontrolerima. Osnovu rada sa mikrokontolerima predstavlja pisanje programa za mikrokontrolere. Program mora da se izvršava tokom celog radnog veka uređaja, tj. od trenutka kada uređaj dobije napajanje do trenutka kada napajanje prestane da radi. Mikrokontroler izvšava programske instrukcije na osnovu takta oscilatora. Brzina izvršavanja instrukcija zavisi od frekvencije instrukcionog takta. Prekidne rutine su delovi glavnog programa koji mogu da se dodaju po potrebi.

MPLABX je univerzalni skup alata za razvoj softvera za veliki broj modela mikrokontrolera svih nivoa (8-bitnih, 16-bitnih i 32-bitnih PIC i dsPIC mikrokontrolera). Za potrebe našeg zadatka koristimo kompajler XC16 za 16-bitne mikrokontrolere.

# **1.3 Konfiguracija portova mikrokontrolera dsPIC30F4013**

Prema potrebama uređaja koji se razvija, svi priključci portova mogu se definisati kao ulazni ili kao izlazni. Definisanje priključaka ovog mikrokontrolera kao ulaznog ili kao izlaznog se vrši na taj način što se u registar TRIS upisuje odgovarajuća kombinacija nula i jedinica. Ako je na odgovarajućem mestu u TRIS registru upisana logička jedinica, „1“ , onda je taj pin definisan kao *ULAZNI*, u obratnom slučaju, ako je upisana logička nula, „0“ pin je definisan kao *IZLAZNI.* Tipičan primer za konfiguraciju pinova u standardnom *C* kodu *Microchip-*ovog kompajlera XC16 dat je na slici DODAJ SLIKU.

Najviše priključaka ima port B koji je i najznačajniji port jer je to jedini port koji može da se konfiguriše kao digitalni i kao analogni, odnosno može da radi i sa analognim i sa digitalnim signalima. Pošto se port B može koristiti kao analogni ulaz ili kao digitalni ulaz ili kao digitalni izlaz, kompajler zahteva da korisnik definiše ulogu celog porta B ili njegovih pinova pojedinačno. U tu svrhu koristi se registar ADPCFG.

**ADPCFGbits.PCFG4=1;** //*pin B4 je definisan kao digitalni priključak*

**ADPCFGbits.PCFG5=0;** //*pin B5 je definisan kao analogni priključak*

Da bismo pravilno upisali željeno stanje na odgovarajući priključak mikrokontrolera, potrebno je prvo definisati taj priključak kao izlazni. Nakon toga, korišćenjem odgovarajućeg LAT registra upisujemo željeno logičko stanje na odgovarajući priključak:

**LATBbits.LATB4=1;** //*pin B4 je postavljen na logičku jedinicu*

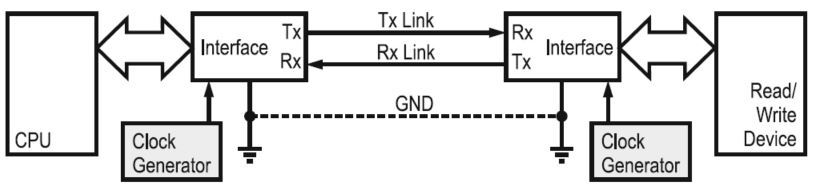
**LATFbits.LATF6=0;** //*pin F4 je postavljen na logičku nulu*

# **Rad sa serijskim UART portovima**

UART (eng. *Universal Asynchronous Receiver Transmitter*) je jedan od modova komunikacije mikrokontrolera sa drugim uređajima. RS232 je naziv za jedan od standarda koji definiše digitalnu, serijsku komunikaciju za povezivanje računara odnosno DTE i DCE uređaja. RS232 je asinhrona serijska komunikacija namenjena za kraća rastojanja i manje brzine prenosa.

# **2.1 Asinhrona serijska komunikacija**

U asinhronim komunikacionim kanalima, na svakom kraju kanala postoji nezavisni izvor sinhronizacionog signala. Da bi se obezbedila pouzdrana komunikacija, uređaji na oba kraja kanala konfigurisani su na takav način da generišu podatke **istom brzinom** i koriste **isti protokol** za razmenu paketa. Slika DODAJ SLIKU prikazuje topologiju asinhronog, *full-duplex* serijskog kanala sa jasno vidljivim nezavisnim izvorima sinhronizacionog siganala.



Slika . Sistem asinhronog serijskog prenosa

# **2.2 Brzina serijske komunikacije**

**Brzina prenosa podataka** preko serijskog kanala određena je količinom vremena potrebnog za prenos jednog bita informacije (tbit).

U praksi se koriste dve metrike za iskazivanje brzine prenosa podataka preko serijskih kanala:

- Bitska brzina (eng. *bit rate*)

- Simbolska brzina (eng. *baud rate*)

**Bitska brzina** (eng. *bits per second*, bps) pokazuje koliko je bita moguće preneti tokom jedne sekunde preko serijskog kanala. Ukoliko je poznato bitsko vreme, tbit, bitska brzina računa se pomoću sledećeg izraza. Na primer, serijski kanal sa bitskim vremenom tbit = 10 ns ima bitsku brzinu od 100 Mbit-a u sekundi (100 Mbps, 100 Mbit/s).

**Simbolska brzina** (eng. *baud rate*), odnosi se na broj karaktera (simbola) koji se mogu preneti preko serijskog kanala tokom jedne sekunde. Ukoliko je svaki simbol predstavljen pomoću jednog bita, simbolska i bitska brzina su jednake.

Ako se koristi 8-bitna komunikacija, tada se za jedan bajt mora preneti 10 bita (8 bita + 1 start bit + 1 stop bit), pa se može približno smatrati da je brzina u bajtovima približno jednaka jednoj desetini brzine u baudima. Da bi komunikacija bila moguća predajnik i prijemnik moraju biti podešeni na istu brzinu, jer se takt ne prenosi posebno. U slučaju drugog zadatka potrebno je u aplikaciji za serijsku komunikaciju odabrati brzinu od 28800 Bd (Bauda), što označava brzinu komunikacije izraženu u broju prenetih simbola, a u ovom slučaju bita u sekundi.

# **2.3 Naponski nivoi**

Svi signali koji čine interfejs serijskog kanala moraju da koriste logičke naponske nivoe koji se koriste unutar UART modula. Na primer, ukoliko UART modul radi na naponu napajanja od 5V, on će biti u stanju da radi i toleriše TTL kompatibilne signale. Ukoliko je UART deo nekog mikrokontrolera koji radi na 3.3V, on će biti u stanju da prihvata signale iz opsega 0-3.3V. Korišćenje logičkih naponskih niova za primopredaju serijskih podataka u mnogome će smanjiti maksimalno rastojanje na koje ovi podaci mogu da “putuju” na desetine cm, pre nego što oni postanu previše “zaprljani” šumom, pogotovo u slučaju velikih transmisionih brzina.

# **2.4 Konvertor napona**

Električne karakteristike signala u mirkroračunarskom sistemu drugačije su od električnih signala definisanih standardom RS 232. Signali mikroračunara, na primer signali mikrokontrolera x51, kompatibilni su sa TTL standardom, pa je između mikroračunara i uređaja sa signalima koji su u električnom pogledu kompatibilni sa RS 232 standardom potrebno staviti konvertor za električno prilagođenje signala ova dva standarda.

U slučaju drugog zadatka koristimo serijski modul FT232RL FTDI koji predstavlja USB to TTL konvertor. Ovaj serijski modul omogućava da povežemo TTL serijske uređaje na PC preko USB porta. Podržava naponske nivoe 5V i 3.3V. Na slici DODAJ SLIKU prikazan je fizički izgled serijskog modula FT232RL.



SLIKA USB to TTL konvertor – FT232RL FTDI serijski modul

Primećujemo da ovaj modul ima dodatne pinove DTR i CTS. Pin DTR podešen je kao izlazni i ima ulogu DODAJ (eng. *Data Terminal Ready Control Output*)*,* dok je pin CTS podešen kao ulazni i ima ulogu dozvole (eng*. enable*) za slanje kontrolnog ulaza (eng. *Clear to Send Control Input*).

# **Autonomno kretanje robota**

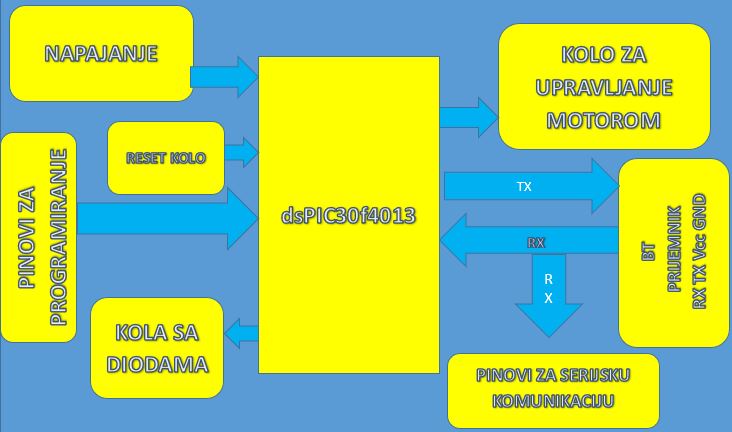
U sklopu drugog zadatka neophodno je realizovati poseban algoritam rada koji oduhvata autonomno kretanje robota uz izbegavanje prepreka. Robot treba da krene sa startne linije, zaobiđe sve prepreke koje naiđe na svom putu i vrati se na startnu liniju. Prepreke na putu na koji robot nailazi su nasumično postavljene, ali tako da se svaka strane prepreke može jasno detektovati. Prilikom izrade koristićemo 2 senzora daljine: jedan analogni senzor GP2Yxxxx i jedan digitalni senzor HC-SR04. U nastavku poglavlja biće objašnjen princip rada navedenih senzora.

Takođe prilikom realizacije zadatka potrebno je držati se sledećih pravila:

* Robot mora da krene sa startne linije i da se vrati na startnu liniju,
* Signal za start robot dobija putem *Bluethooth* komunikacije,
* Robot koristi proizvoljni algoritam rada,
* Robot ne sme da udari u prepreku,
* Robot u relanom vremenu treba da šalje informaciju o svom kretanju putem *Bluethooth* komunikacije. Informacije treba da budu prikazane na aplikaciji mobilnog telefona.

# **Blok dijagram podsistema**

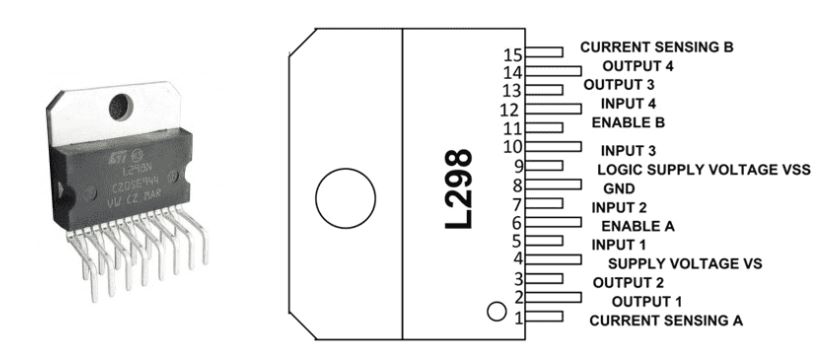
Potrebno je projektovani proizvoljan sistem koji se sastoji od podsistema prikazanih na slici DODAJ SLIKU. U nastavku poglavlja biće reči samo o određenim podsistemima koji nisu obrađeni u sklopu prvog zadatka.



Slika Blok dijagram

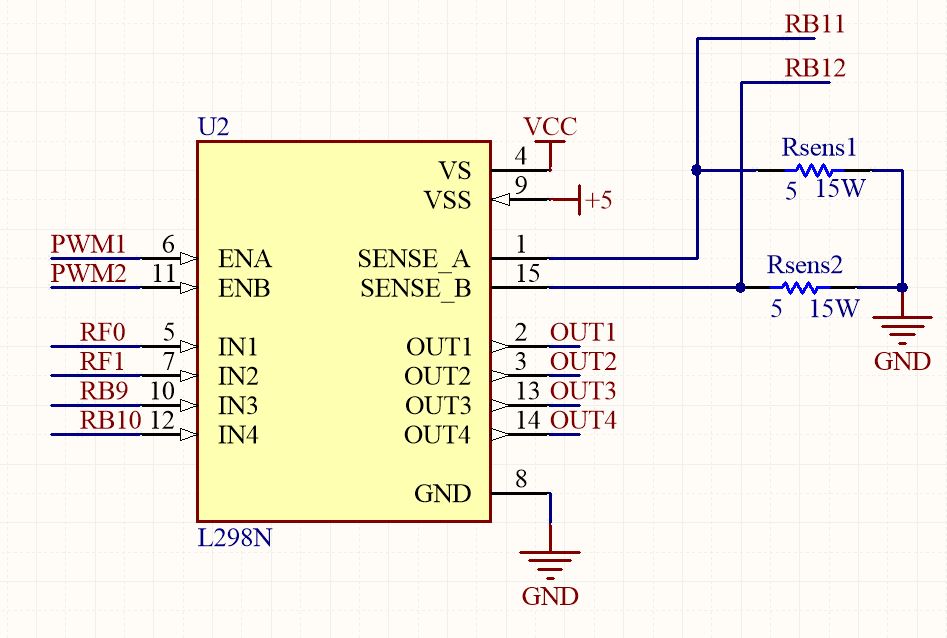
# **Kolo za upravljanje motorima**

U slučaju našeg robota koristimo tenkić na kom se nalaze dva pogonska DC motora. U pitanju su DC motori bez četkica. Na rotoru su permanentni magneti, a stator se pobuđuje odgovorajućom elektronikom. U pitanju je *L298N Motor Driver Modul,* prikazan na slici DODAJ SLIKU, koji predstavlja dualni H-most drajver za motor koji omogućava direktnu kontrolu dva DC motora u isto vreme. Ovaj modul se može koristiti za napone izmedju 5V i 35V, ali sa strujnim ograničenjem do 2A.



Slika L298N

Za projektovanje PCB-a (eng. *Printed Circuit Board*) koristili smo softverski alat Altium designer. Na slici DODAJ SLIKU prikazan je deo šematika (eng. *Schematic*) vezan za modul L298N.



SLIKA

Preko pinova D0 i D1 šaljemo PWM signal na pinove modula 6 i 11 koji predstavljaju dozvolu za motore A i B, tako što pinove prvo definišemo kao izlazne pomoću TRISD registra i u tom slučaju pišemo:

**TRISDbits.TRISD0=0;** //*pwm za pogon motora* *A*

**TRISDbits.TRISD1=0;** //*pwm za pogon motora B*

Pinove RF0, RF1, RB9 i RB10 mikrokontrolera dsPIC30F4013 vezujemo na pinove 5, 7, 10 i 12 respektivno i pomoću njih možemo menjati smer jednog odnosno drugog motora u zavisnosti koji pinovi su aktivni. Pre svega potrebno ih je sve definisati kao izlazne.

**TRISFbits.TRISF0=0;** //*za smer prvog motora*

**TRISFbits.TRISF1=0;** //*za smer prvog motora*

**TRISBbits.TRISB10=0;** //*za smer drugog motora*

**TRISBbits.TRISB11=0;** //*za smer drugog motora*

U nastavku je dat primer kretanja pravo. Prvo je potrebno napisati funkciju u kojoj aktiviramo određene pinove korišćenjem registra LATF i LATB kojima postavljamo priključak na logičku jedinicu i/ili nulu. Takođe u istoj funkciji šaljemo i PWM signal na već pomenute pinove 6 i 11.

**void pravo**()

{

**LATFbits.LATF0=0;** //*za smer DESNOG motora*

**LATFbits.LATF1=1;** //*motori gone unazad*

**LATBbits.LATB10=0;** //*za smer motora*

**LATBbits.LATB11=1;** //*motori gone unazad*

**PWM1**(400);

**PWM2**(400);

}

U nastavku je prikazana funkcija u kojoj kao parametar prosleđujemo trajanje internog PWM-a mikrokontrolera. Isto važi i funkciju PWM2.

**void PWM1**(**int** dutyC)

{

**PR2** = 500;//*određuje frekvenciju po formuli*

**OC1RS** = dutyC;//*postavimo pwm*

**OC1R** = 1000;//*inicijalni pwm pri paljenju samo*

**OC1CON = OC\_IDLE\_CON & OC\_TIMER2\_SRC & OC\_PWM\_FAULT\_PIN\_DISABLE& T2\_PS\_1\_256;**//*konfiguracija pwm-a*

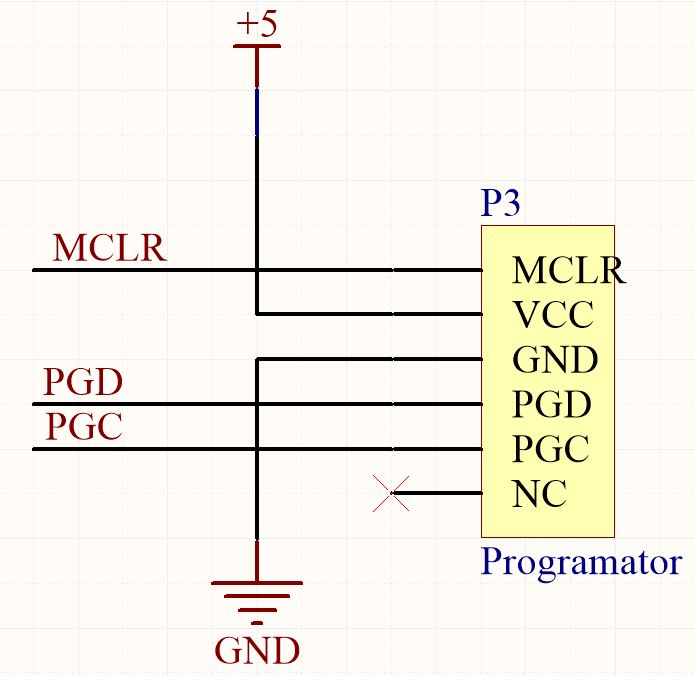
**T2CONbits.TON = 1**;//*uključujemo tajmer koji koristi*

}

# **Pinovi za programiranje**

Prilikom izrade prvog zadatka iz Primenjene elektronike na raspolaganju nam je bila razvojna ploča EasyPic v7 firme *Mikroelektronika* sa mikrokontrolerom dsPIC30F4013 koja je na sebi imala programator, koji nam je omogućavao spuštanje koda na mikrokontroler. Prilikom izrade drugog zadatka jedan od zahteva je da napravimo sopstvenu pločicu (PCB) na kojoj će se nalaziti pomenuti mikrokontroler. Za programiranje koristićemo *Pickit 3 Programmer* prikazan na slici DODAJ SLIKU. MPLAB PICkit 3 nam omogućava *debugg*-ovanje i programiranje PIC i dsPIC Flash mikokontrolera korišćenjem MPLAB IDE (eng. *Integrated Development Environment*). Pinovi RB6 i RB7 (tj. PGD i PGC) služe za programiranje mikrokontrolera. Na slici DODAJ SLIKU prikazan je deo šematika vezan za programator.





SLIKA PICkit 3 Programator

SLIKA Šematik programator

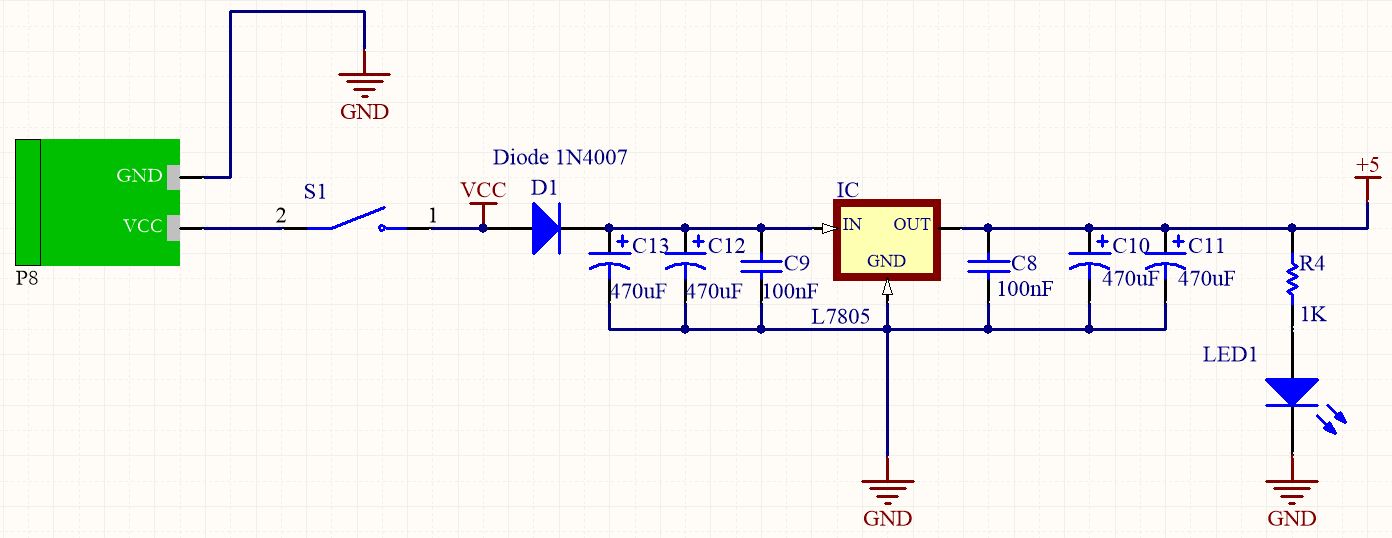
Bitno je još napomenuti da je prilikom spuštanja koda na mikrokontroler potrebno isključiti spoljašnje napajanje, u našem slučaju to je laboratorijsko napajanje tj. kontroler bi trebalo da napajanje dobija preko programatora.

# **Napajanje**

Za pravilan rad mikrokontrolera potrebno je obezbediti stabilan i pouzdan napon od 5V, dok je za motore potrebno obezbediti nešto viši napon. U našem slučaju kao napajanje je iskorišćena jedna baterija od 9V. Kako bi obezbedili stabilan naponski nivo koristimo **linearni stabilizator** L7805 u kućištu TO-220. Napon na izlazu stabilizatora ne treba da zavisi od promena:

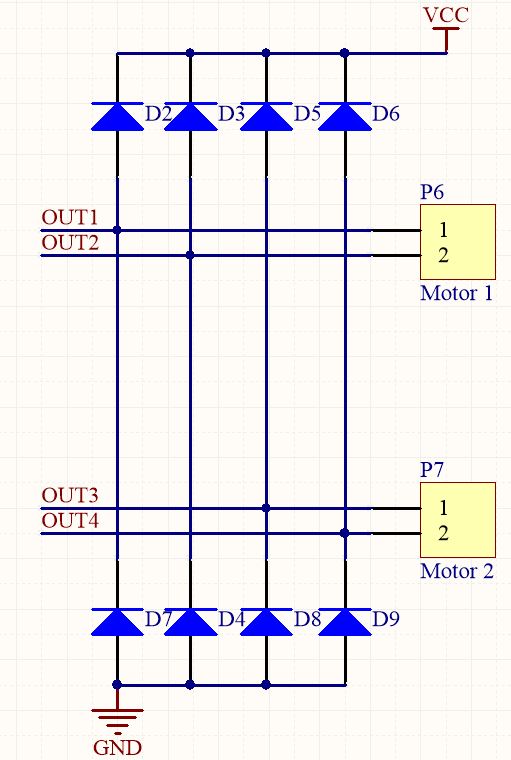
* Ulaznog napona (napon na izlazu filtra)
* Otpornosti potrošača (struje kroz potrošač) i
* Temperature

Na slici DODAJ SLIKU prikazan je blok za napajanje. Manji kondenzatori sa oznakama C8 i C9 imaju vrednosti od 100nF i služe kao filteri napona dok veći elektrolitski kondenzatori, sa oznakama C10, C11, C12 i C13 i vrednostima od po 470µF , imaju ulogu da tokom paljenja motora obezbede dovoljan napon. Naime, tokom paljenja motori imaju značajnije veću strujnu potrošnju od nominalne i to može predstavljati problem jer može doći do neželjnog resetovanja mikrokontrolera. Elektrolitski kondenzatori otklanjaju taj problem tako što će motori, prilikom paljenja, povući iz njih potrebnu struju (treba napomenuti da se ovo dešava u jako kratkom vremenskom intervalu), a stavljena su po dva, i sa jedne i sa druge strane, u slučaju ako neki prestane sa radom.

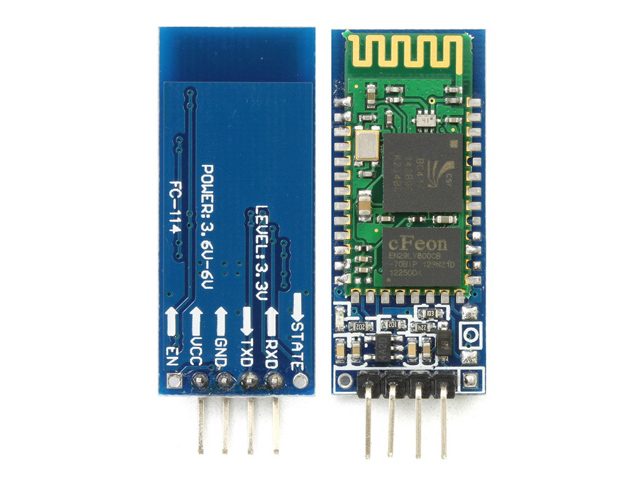


Dioda D1 ima ulogu prekidača, tj. da prekine kolo, u slučaju ako se baterija okrene naopako. LED (eng. *Light Emitting Diode*) signalizira ispravan rad linearnog stabilizatora. Stabilan napon od 5V dalje vodimo ka mikrokontroleru, a napon iz baterije direktno vodimo ka motorima (na šemi označeno sa **VCC**) .

# **Kolo sa diodama**

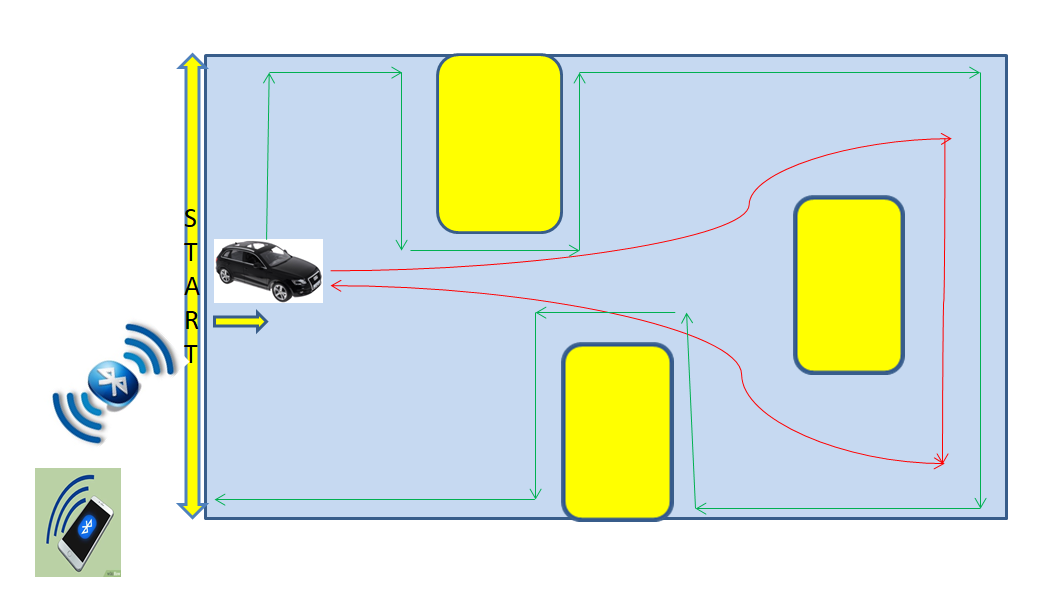


# **BT Prijemnik**



# **Senzori**

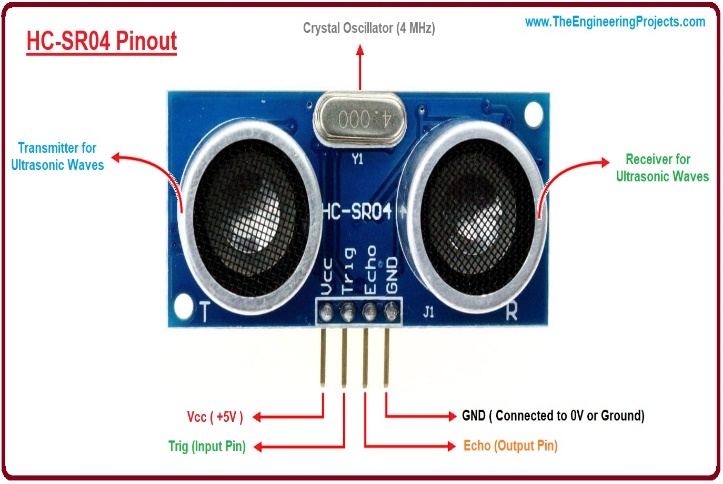
U drugom zadatku na raspolaganju su 3 senzora daljine: jedan analogni senzor GP2Yxxx(SLIKA DODAJ SLIKU) i dva digitalna senzora daljine HC-SR04 (SLIKA DODAJ SLIKU). Na slici DODAJ SLIKU data su dva predloga algoritma za realizaciju zadatka. Kao rešenje izabrano je praćenje ivice. Pomoću digitalnog senzora, koji je postavljen normalno na pravac kretanja, dobijamo informaciju koliko se robot nalazi blizu zida, dok se analogni senzor, koji je postavljen paralelno, koristi kako bi dobijali informaciju o potencijalnim preprekama u pravcu kretanja robota.



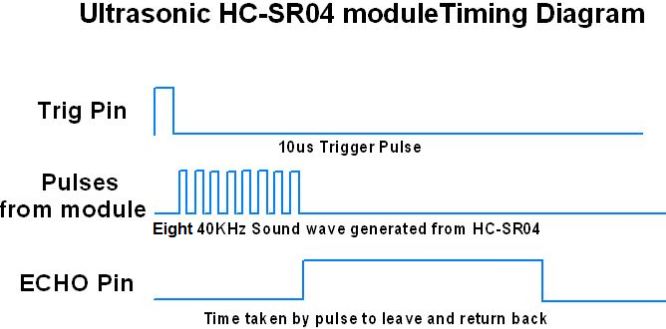
SLIKA ALGORITAM ZA REALIZACIJU DRUGOG ZADATKA

# **3.7.1 HC-SR04**

Ultrazvučni senzori rastojanja se koriste za merenje rastojanja između senzora i objekta koji se nalazi ispred njega. Udaljenost možemo meriti od objekta koji miruje ili onog koji se kreće. Modul se sastoji od ultrazvučnog predajnika, ultrazvučnog prijemnika i kontrolne elektronike. Rezolucija senzora je 0,3cm, a opseg detektovanja objekta od 2 do 400cm. Senzor pokriva horizontalni ugao do 15o. Senzor emituje zvučne impulse visoke frekvencije, koji se, u slučaju da se ispred senzora nalazi prepreka, odbijaju od nje ka senzoru. Ako su impulsi detektovani nakon emitovanja, možemo pretpostaviti da se ispred senzora nalazi prepreka. Dataljini opis principa rada je sledeći: modul se aktivira slanjem kontrolnog impulsa dužine najmanje 10 μs,  zatim automatski šalje osam ultrazvučnih impulsa frekvencije 40kHz i kada detektuje povratne ultrazvučne impulse, generiše izlazni signal čija je dužina proporcionalna razdaljini.

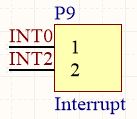


SLIKA DODAJ SLIKU



SLIKA DODAJ SLIKU

Prilikom projektovanja šematika izvučeni su pinovi 17 i 18 sa oznakama INT0 i INT2 (slika DODAJ SLIKU) koji će nam poslužiti za pravilno očitavanje digitalnog senzora.



Poznavajući princip rada, pomenutnog senzora, treba precizno očitati dužinu trajanja impulsa koji se dobija sa ECHO pina. Prvo je potrebno aktivirati potrebne registre za interupt i to je urađeno u sledećoj funkciji:

**unsigned int flaglevo1, brojanjeInt;**

**void \_\_attribute\_\_** ((\_**\_interrupt\_\_, no\_auto\_psv**)) **\_INT0Interrupt**(void)

{

**brojanjeInt++;**

**if(brojanjeInt % 2 == 0)**//*rastuća ivica*

{

**flaglevo1 = 1;**

**T3CONbits.TON=1;**

**INTCON2bits.INT0EP = 1;**

}

**else if**(brojanjeInt % 2 == 1)//*opadajuća ivica*

{

**vremeGlobal=TMR3;**

**flaglevo1 = 0;**

**TMR3=0;**

**T3CONbits.TON=0;**

**INTCON2bits.INT0EP = 0;**

}

**IFS0bits.INT0IF = 0;**

}

Ideja je sledeća: U zavisnosti od udaljenosti objekta od senzora dobijaćemo različite dužine impulsa i to ako se predmet nalazi dalje, vreme trajanja impulsa biće veće. Korišćenjem brojača mi možemo precizno očitati pomenutno vreme. Prilikom detektovanja rastuće ivice ulazi se u interrupt rutinu i aktivira se brojač, podiže se zastavica (eng. *Flag*) i izlazi se iz *interrupt-a.* Na opadajuću ivicu ponovo se ulazi u *interrupt*, potom se spušta zastavica i brojač se zaustavi.

Takođe bitno je napomenuti da je prilikom inicijalizacije u **main** delu potrebno setovati dozvolu za spoljašnji interupt:

**IEC0bits.INT0IE = 1;**//dozvola za interupt na INT0

**IFS0bits.INT0IF = 0;**//clear external interrupt 0 flag

Vrednost tajmera smeštamo u promenjivu **vremeGlobal** (promenjiva je tipa **double**). Za pravilan rad senzora potrebno je generisati impuls dužine najmanje 10 μs koji šaljemo na TRIG što je urađeno u sledećoj funkciji:

**double** OcitajLevo1()

{

**double vreme1=0;**

**double distanca1=0;**

**trig\_levo1 = 0;**//*generisanje TRIG signala*

**delay\_us(1);**

**trig\_levo1 = 1;**

**delay\_us(1);**

**trig\_levo1 = 0;**

**vreme1=vremeGlobal;//***vreme1 = brojac\_ms3 + sekund\*1000*;

**WriteUART2dec2string**(vreme1);

**distanca1 = vreme1 / 10 ;**

**T3CONbits.TON = 0;**

**return** **distanca1;**

}

Prilikom definisanja perioda tajmera potrebno je voditi računa da dobijamo vrednosti u mikrosekundama:

**#define TMR2\_period 100**

**#define TMR1\_period 100** // *Fosc=10MHz 1/Fosc=0.1us 0.1\*100=10μs*

**#define TMR3\_period 50000** *// 0.1\*50000=5ms*

**void delay\_us** (**unsigned int** mikrosek) //*za senzore 10μs*

{

**br1 = 0;**

**while(br1 < mikrosek);**

}

# **3.7.2 IR senzor daljine GP2Y0A21YK**

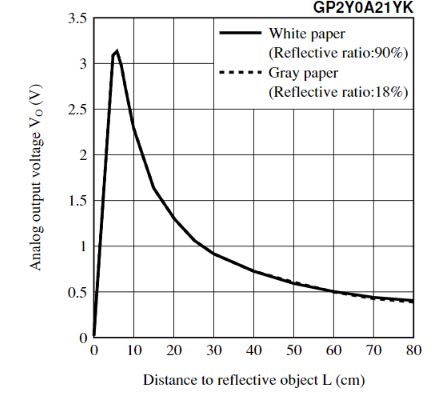
IR senzor spada u analogne senzore i njegova primena je prilično jednostavna. Postoje tri pina:

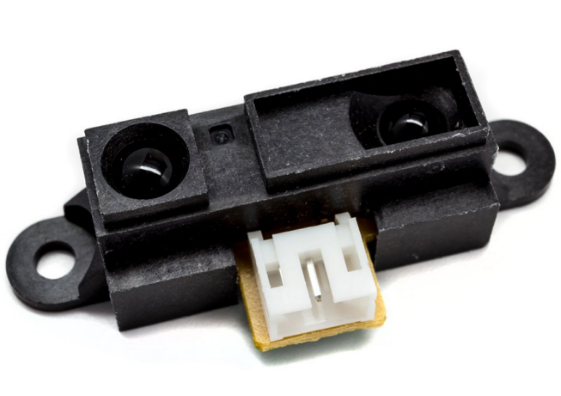
Na pinu 1 se generiše napon, različite amplitude, u zavisnosti od udaljenosti predmeta. Iz datasheet-a za dati senzor znamo da je ta vrednost napona u opsegu vrednosti: 0.25V (min), 0.4V (TYP) i 0.55V (MAX) Zavisnost amplitude od dužine može se videti na grafiku prikazanom na slici (DODAJ SLIKU). Vrednost očitanog napona se dalje vodi na A/D konvertor. Za ovu svrhu je iskorišćen PIN B1, koji je prethodno podešen kao ulazni. Vrednosti SMPI registra se poveća za jedan i u ADCSSL podesimo pin B1:

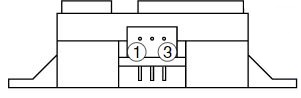
**ADCON2bits.SMPI=2;**//*inicijalno je 1*

**ADCSSL=0b0000000000000010;**

Još je potrebno prokomentarisati grafik sa slike (**DODAJ SLIKU**). Možemo primetiti da senzor počinje da generiše nešto manji napon tek na distanci od 4cm. Ovo nam je od interesa jer nam to govori da on mora biti uvučen makar za ta 4cm kako bi precizno uočili prepreku. Ukoliko ovo nije urađeno, ako se predmet nalazi na razdaljini izmeđi 0cm i 4cm, senzor će detektovati prisutnost, ali nećemo znati da li je na 3cm ili 10cm, jer posmatrajući grafik SA SLIKE imamo istu vrednost y ose za 2 različite vredsnoti x ose.







ZAVISNOST NAPONA OD DUŽINE ZA SENZOR **GP2Y0A21YK**

SLIKA SENZORA **GP2Y0A21YK**