

Projekat iz predmeta 32-bitni mikrokontroleri i primena

- Električno vozilo upravljano putem *ZigBee* komunikacije - Step Motor -

student: Nenad Vukadinović, br. indeksa 3130/2012
profesor: Prof. Dragan Vasiljević
asistent: Mr. Nenad Jovičić

Contents

1	Uvod	3
2	Hardverska realizacija	4
2.1	<i>Zigbee</i> modul	4
2.2	Mikrokontroler <i>STM32</i>	4
2.3	Električna šema celokupnog kola	5
3	Softverska realizacija	7
3.1	Format podatka i opis protokola komunikacije	7
3.2	Softverska Implementacija upravljanja step motorima	8
4	Zaključak	10

1 Uvod

U ovom izveštaju je opisan projekat iz predmeta MS1BMP. Cilj projekta je bio upoznavanje sa osnovnim mogućnostima i periferijama mikrokontrolera *STM32* kroz praktičnu implementaciju jednog odredjenog projektnog zadatka. Projektni zadatak pored toga obuhvata i korišćenje pomenutog mikrokontrolera i njegovo povezivanje sa eksternim hardverom u realnoj aplikaciji.

Definicija Projektnog zadatka

Potrebno je isprojektovati električno vozilo pogonjeno step motorima, čije je upravljanje implementirano pomoću mikrokontrolera *STM32*. Upravljačke signale bežičnim putem pomoću *ZigBee* modula (predajni modul), šalje personalni računar ili džojstik po utvrđenom formatu podataka. Prijemni *ZigBee* modul prima upravljačke signale i pomoću serijske veze prosledjuje mikrokontroleru. Na osnovu primljene upravljačke sekvence mikrokontroler generiše izlazne signale ka drajveru step motora.

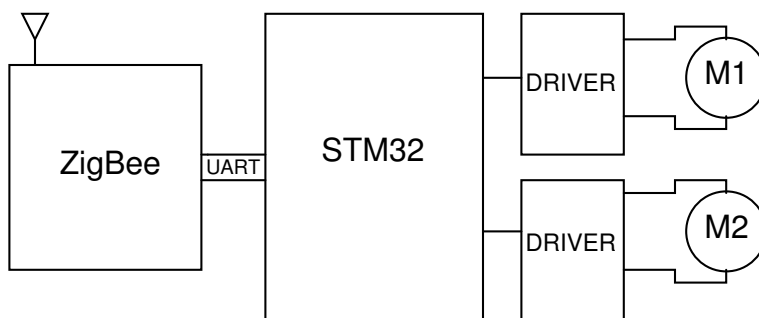


Figure 1: Blok šema

Uvodna napomena

Tokom izrade projekta personalni računar je korišćen za generaciju upravljačkih signala. Predajni *Zigbee* modul je povezan sa računarom pomoću *USB to UART* veze korišćenjem kola *FT232RL*. Za potrebe testiranja vozila, *USB to UART* konektor je pozajmljen od kolege Zdravka Čolića koji je jedan takav konektor napravio prilikom realizacije svog projekta na ovom predmetu [1].

2 Hardverska realizacija

2.1 Zigbee modul

Zigbee prenos je komunikacioni protokol koji se bazira na *IEEE* 802.15.4 standardu i prvenstveno se koristi u aplikacijama sa malom potrošnjom koje ne zahtevaju prenos signala na velike daljine. U ovom projektu koristi se jedna realizacija *Zigbee* modula firme *Digi International* sa oznakom *XBee-Pro RF*. Programiranje predajnog i prijemnog modula uradjeno je pomoću programa *X – CTU*, koji obezbedjuje takodje firma *Digi International*. Budući da se radi o *PEAR-TO-PEAR* vezi bitno je napomenuti da se prilikom programiranja oba modula moraju upariti upisivanjem odgovarajuće odredišne i izvorišne adrese. Pored toga program *X – CTU* omogućava konfiguraciju serijske veze (*baud rate, data bits, parity bit, stop bit*). Oba modula su konfigurisana tako da imaju brzinu prenosa od 9600baud/s , sa 8-bitnim podacima, jednim stop bitom i bez bita parnosti.

Na slici 2 se nalazi električna šema povezivanja *XBee-Pro RF* modula. Kako *XBee-Pro RF*

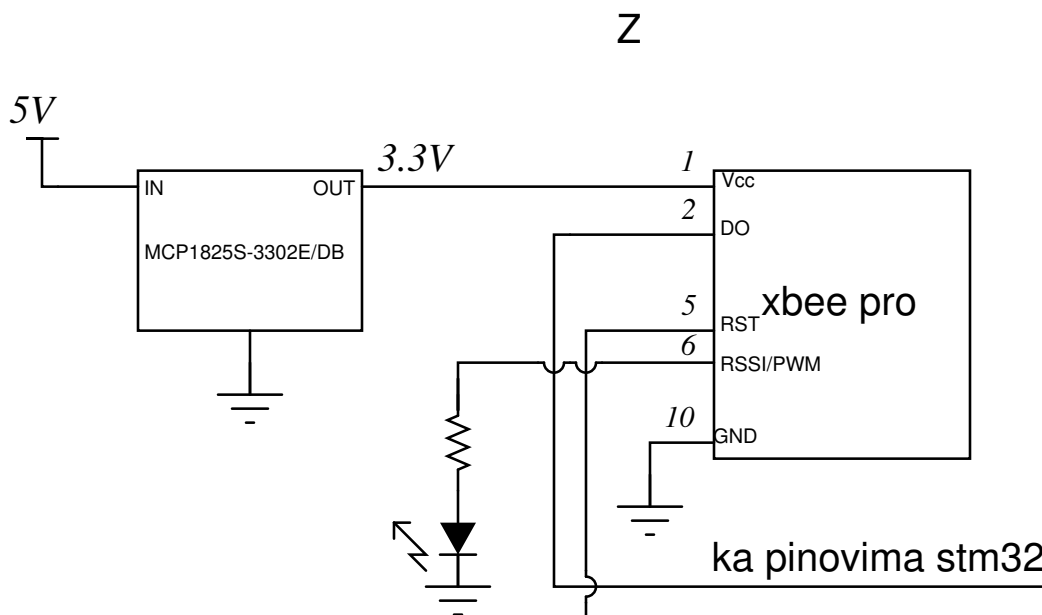


Figure 2: *XBee-Pro RF*

modul zahteva napon napajanja od $2.8 - 3.4\text{ V}$, ali i maksimalnu struju potrošnje od 250 mA stavljen je stabilizator napona koji može da da struju od 0.5 A .

2.2 Mikrokontroler *STM32*

Resursi mikrokontrolera korišćeni za upravljanje step motorima:

- *USART3* - služi za prihvatanje upravljačkih signala pristiglih u prijemni *Zigbee* modul,

- $TA2$ i $TA3$ - tajmeri koji određuju brzinu kretanja motora,
- $GPIOA$ i $GPIOB$, I/O pinovi koji su povezani na drajver step motora $L293B$.

Razvojna ploča mikrokontrolera se nalazi na slici 3. Pinovi označeni sa R_x , T_x i \overline{RST} su pinovi koji su povezani sa *Zigbee* modulom, dok su pinovi označeni sa $M1$ i $M2$ povezani sa drajverom prvog, odnosno drugog step motora. U ovom projektu T_x pin se ne koristi pošto je *USART3* mikrokontrolera programiran tako da samo prima upravljačke podatke.

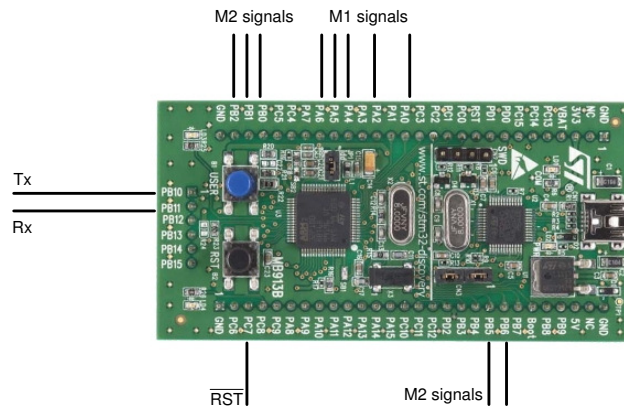


Figure 3: Razvojna ploča mikrokontrolera *STM32*

2.3 Električna šema celokupnog kola

Budući da strujni kapacitet pinova mikrokontrolera nije dovoljno veliki da pogoni step motor, *push-pull* drajver $L293$ je upotrebljen u tu svrhu. Motori koji su korišćeni u ovom projektu jesu unipolarni step motori sa oznakom *MOTS1* step proizvođača *Vellerman*-a. Ovi motori imaju nominalni napon napajanja od 12 V i jačinu struje od 60 mA . S obzirom da strujni kapacitet korišćenog drajvera ide i do jednog ampera, on može bez prevelikih problema da pogoni navedene step motore. Detaljna električna šema mikrokontrolera sa eksternim modulima nalazi se na slici 4

Vozilo se napaja iz baterije od 11.1 V , stoga je potrebno izvršiti stabilizaciju napajanja na 3.3 i 5.5 V . Na slici su prikazani stabilizatori napajanja $78M05$ za mikrokontroler i drajver, kao i već navedeni $MCP1825S - 3302E/DB$ stabilizator na 3.3 V .

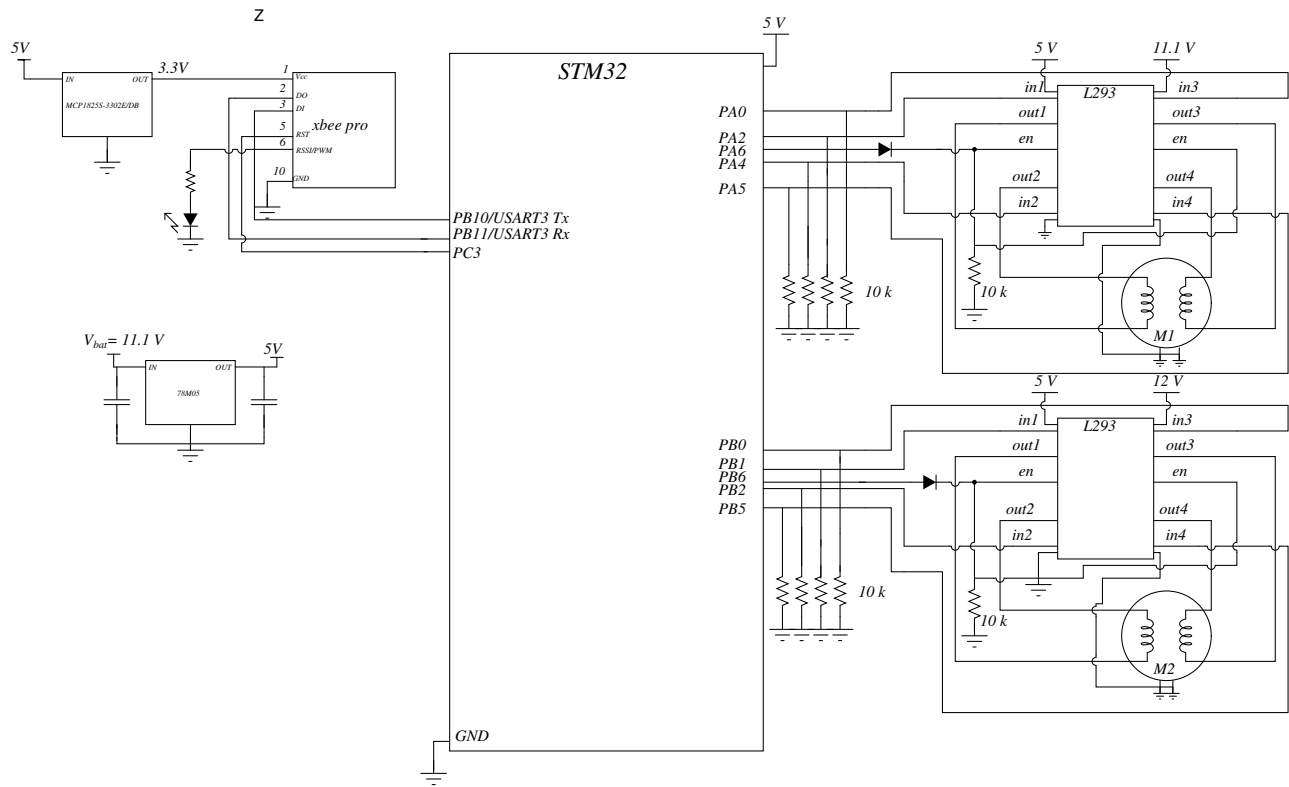


Figure 4: Električna šema

Nakon testiranja projekta na protoboard-u, u programu *Altium Designer* je uradjen *PCB* čiji izgled je predstavljen na slici 5

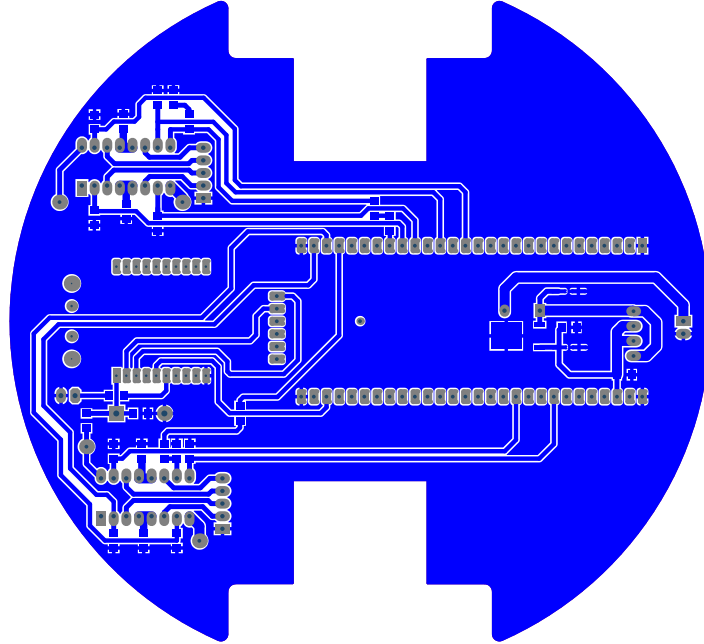


Figure 5: PCB električne šeme

3 Softverska realizacija

3.1 Format podatka i opis protokola komunikacije

¹. Komunikacija između predajnog i prijemnog modula se odvija tako što ukupan podatak o brzini vozila ima 5 bajtova. Prvi bajt označava početak komunikacije i on ima vrednost $0x88$. Drugi bajt definiše brzinu po y -osi (vozilo za $y > 0$ se kreće napred, u suprotnom unazad), dok treći bajt definiše skretanje (x - osa, vozilo za $x > 0$ skreće desno u suprotnom levo). Četvrti je rezervisan za leva i desna navigaciona svetla koja u ovom vozilu nisu implementirana i stoga ovaj podatak nije od interesa u ovoj realizaciji. Peti bajt označava kraj komunikacije i takodje ima vrednost $0x88$. Vrednost drugog i trećeg bajta ograničene su na vrednosti $[-100, 100]$, što znači da se brzina može zadavati sa rezolucijom 100.

Pri testiranju vozila podaci su sa personalnog računara poslani preko *USB-TO-UART* konektora do *Zigbee* modula, pomoću programskog alata *HTerm*.

¹Format podatka je preuzet iz projekta [1] kolege Zdravka Čolića kako bi se ostavila mogućnost upravljanja iz jednog istog upravljačkog modula bez dodatnih podešavanja (PC racunar ili džojstik)

3.2 Softverska Implementacija upravljanja step motorima

U glavnom programu mikrokontrolera vrši se inicijalizacija periferija:

- Setuje se AF funkcija za pinove $PB10$ i $PB11$, koje koristi periferija $USART3$ (signali T_x i R_x).
- Pinovi povezani na $L293$ drajver se setuju kao izlazni.
- Konfigurira se $UART3$ sa istim karakteristikama kao $UART$ u predajnom modulu (8-bit data , $no\ parity$, $1\ stop\ bit$, $baud\ rate = 9600$).
- Konfigurira se tajmeri opšte namene $TIM2$ i $TIM3$. Vrednost takta ovih tajmera je postavljena na $100\ KHz$, vrednost perioda na 250 .
- Setuju se prioriteti prekida; prioritet prekida $USART3$ je postavljen na najveći nivo setovanjem $Preemption\ priority$ -ja. Prioritet prekida tajmera je za jedan niži od prioriteta prekida $USART3$ -a. Tajmeri imaju i jednake vrednosti $Subpriority$ -ja. To omogućava jednako vreme generisanja prekida u slučaju da tajmeri imaju jednake vrednosti takta i periode.

Nakon inicijalizacije na kratko se resetuje *Zigbee* upisom nule u izlazni registar $GPIOC$ (na poziciju $PC7$) i dozvoli se generisanje prekida $USART3$ periferije. Glavni program nakon toga ulazi u beskonačnu *while* petlju. Prekidna rutina UART-a se generiše pri svakom prijemu jednog bajta. U njoj se vrši dekodovanje ulaznog podatka po formatu koji je opisan u prethodnom poglavlju. Najpre se proverava da li je stigao bajt koji označava početak upravljačkog podatka. Ukoliko prvi bajt ima vrednost $0x88$ određuje se vrednost brzine (y -osa) i skretanje (x -osa) i čeka se kraj podatka $0x88$. Ukoliko je podatak primljen određuju se vrednosti ARR registra tajmera (ovaj registar definiše period prekida) na osnovu sledeće formule:

$$ARR_{left} = \frac{2ARR_{min}10^4}{(100 + x)|y|}$$
$$ARR_{right} = \frac{2ARR_{min}10^4}{(100 - x)|y|}$$

Za $x > 0$, ARR_{left} se smanjuje a samim tim levi točak ubrzava pa vozilo skreće desno, dok za $x < 0$ vozilo skreće levo. Vrednost ARR_{min} je minimalna dozvoljena vrednost ARR , za maksimalne x i y . Pošto se u prekidnoj rutini tajmera vrši komutacija priključaka motora vrednost ARR_{min} je ograničena maksimalnom dozvoljenom brzinom te komutacije (eng. *max slewing pulse rate*). Za korišćene step motore ovaj podatak iznosi $800\ pps$ (ili jedan puls na svakih $1.25\ ms$). Stoga pri učestanosti takta brojača tajmera od $100\ KHz$ vrednost ARR_{min} iznosi 125 . U slučaju osnog kretanja ($y = 0$), perioda tajmera se izračunava formulom:

$$ARR_{left} = \frac{2ARR_{min}100}{x}(x > 0)$$

$$ARR_{right} = \frac{2ARR_{min}100}{-x} (x < 0)$$

Program pri svakom ulazu u prekidnu rutinu oba tajmera radi upis upravljačke sekvence u *GPIO* izlazni registar. Upravljačka sekvenca se sastoji od niza definisanog kao eksterna varijabla u memoriji mikrokontrolera. Za svaki motor je definisano po četiri člana ovog niza čije vrednosti predstavljaju jedno stanje ulaznih portova motora. Ako su priključci motora *A1*, *B1*, *A2* i *B2* upravljačka sekvenca koja karakteriše *full stepping* upravljanje step motora data je na slici 6. Ako se sekvenca generiše u smeru naznačenom kao na slici (inkrementiranjem iteratora niza) vozilo se kreće napred (slučaj $y > 0$). Kada y podatak ima negativnu vrednost *ARR* registra se definiše na isti način kao i za $y > 0$, s tim što se upravljačka sekvenca dobija dekremenitanjem iteratora navedenog niza.

	B2	B1	A2	A1	portovi motora
	PA2	PA0	PA5	PA4	GPIO
3	0	0	1	1	0
2	1	0	0	1	1
1	1	1	0	0	2
0	0	1	1	0	3

↑ clan niza

↓ clan niza

Figure 6: *Full stepping*

Blok dijagram opisanog algoritma prikazan je na slici 7.

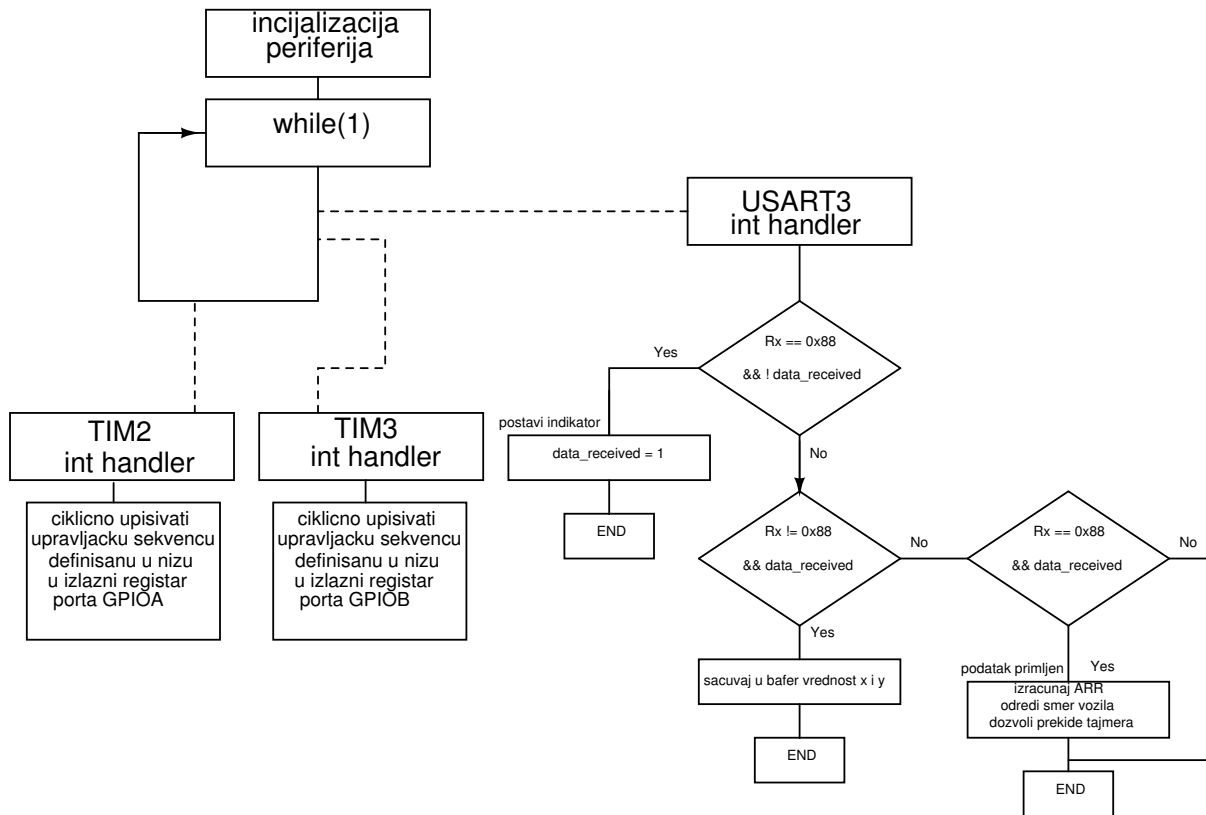


Figure 7: Blok dijagram algoritma

4 Zaključak

Testiranjem projektovanog vozila pokazala su se ograničenja step motora u pogledu brzine obrtaja. Step motore je potrebno koristiti u slučajevima gde brzina nije od interesa već pozicija i preciznost, stoga su pokazani rezultati bili više nego očekivani. Ispravnost samog uređaja je bila zadovoljena i on je testiran i u slučaju kontrole putem personalnog računara i u slučaju kontrole korišćenjem džojstika. Prostor za dalje poboljšanje performansi ovog vozila se sigurno ogleda u odabiru motora ili step motora sa većim *pps*-om ili promenom tipa motora.

Jedno od softverskih poboljšanja koje se može uraditi jeste primena uštede procesorskog vremena korišćenjem drugog mastera u sistemu - DMA kontrolera. U ovom projektu DMA kontroler može biti upotrebljen na dva načina. Prvi način je taj da DMA bude programiran tako da automatski skladišti podatke pristigle u prijemni bafer UART-a u memoriju. Te podatke bi procesor kasnije mogao da koristi za računanje perioda tajmera i smer kretanja vozila. Drugi možda bolji način, korišćenja DMA kontrolera jeste taj da se izbacila prekidna

rutina tajmera već, da tajmeri prilikom isteka perioda brojača trigeruju DMA prenos iz memorije (u kojoj bi bila definisana upravljačka sekvenca step motora) direktno u izlazni registar GPIO pinova.

References

- [1] Zdravko Čolić, *Zigbee-jem upravljano motorno vozilo - DC motori*
- [2] Mr Nenad Jovičić, *Vežbe sa predmeta MS1BMP*
`tnt.etf.rs/~ms1bmp`
- [3] *STMicroelectronics Reference manual RM0041.pdf*