

UNIVERZITET U BEOGRADU
ELEKTROTEHNIČKI FAKULTET

Projekat iz
NAMENSKIH RAČUNARSKIH SISTEMA

RS-485 i Bluetooth komunikacija



Student:
Aleksa DAMLJANOVIĆ 89/2012

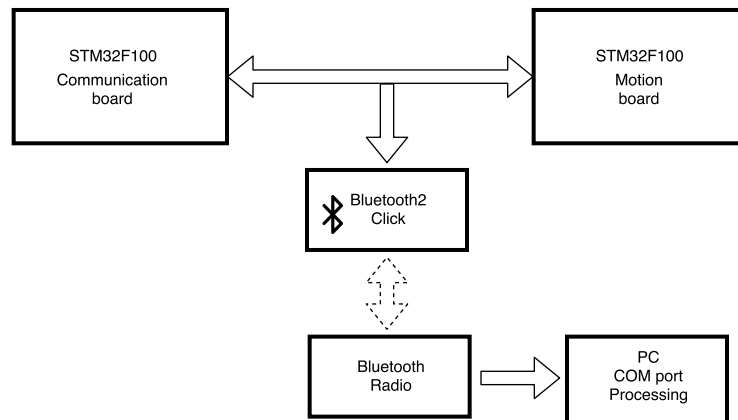
Profesor:
doc. dr Nenad JOVIČIĆ

Sadržaj

1	Opis projekta	3
1.1	Opis sistema	3
1.2	<i>Bluetooth2 Click</i>	4
1.3	Inicijalizacija mikrokontrolera	4
1.4	Inicijalizacija <i>Bluetooth Click</i> -a	5
2	Komunikacijski protokol i interfejs	8
2.1	<i>Communication (Main) Board</i>	8
2.2	<i>Motion (Slave) Board</i>	10
3	Prenos podataka sa platforme preko <i>Bluetooth</i>-a do računara	11

Uvod

Ovaj projekat opisuje format i realizaciju celokupne komunikacije koja se koristi u sistemu koji je realizovan za potrebe diplomskog rada. Ideja projekta je da se napravi sistem koji povezuje 2 mikrokontrolera, odnosno dve ploče i *Bluetooth* modul korišćenjem RS-485 magistrale na zajedničkoj platformi, a potom da se manuelno poveže računar sa *Bluetooth* modulom radi vizuelizacije odgovarajućih podataka sa platforme. Opšta šema prikazana je na slici ispod.

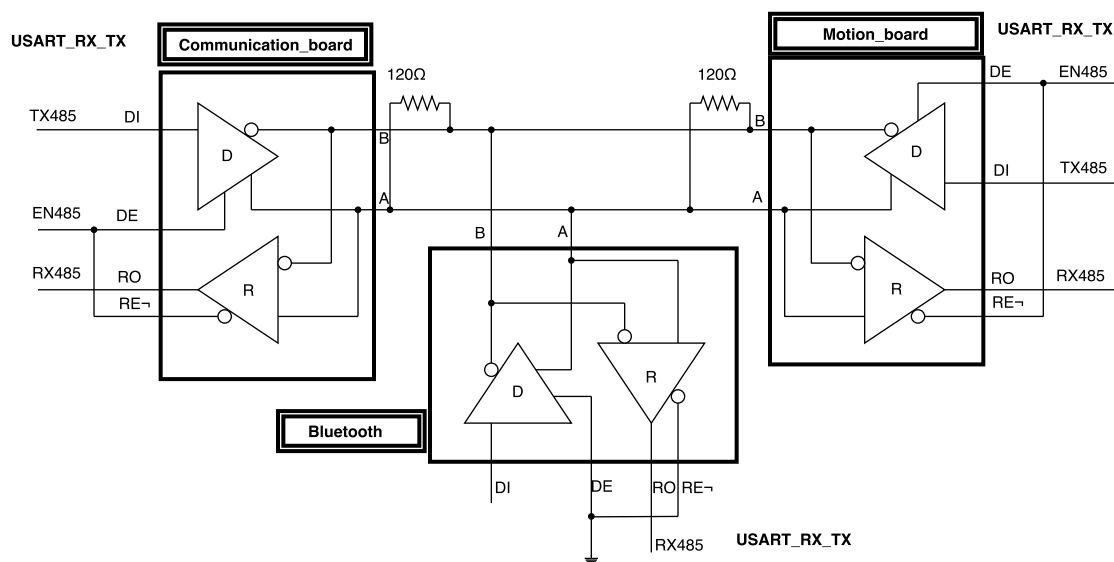


Opšta blok šema povezivanja komponenti u sistemu.

1 Opis projekta

1.1 Opis sistema

Korišćene su STM32 VL Disocvery[4] razvojne ploče sa STM32F100RBT6B mikrokontrolerima[3]. Na samim mikrokontrolerima se koristi USART interfejs komunikacije. Sa čipovima MAX1487[2], koji predstavljaju transivere male potrošnje za RS-485 komunikaciju, svi uređaji su povezani na jedinstvenu magistralu. Svaki od čipova sadrži jedan prijemnik(R) i jedan predajnik(D). Šema sa uređajima i načinima povezivanja na magistralu data je na slici 1.1.1.



Slika 1.1.1: Blok šema sistema.

Master ploča je **Communication board**. Ona šalje komande, koje mogu, a ne moraju sadržati dodatne podatke, *slave* ploči **Motion board** koja upravlja motorima, tj. kretanjem. Ova ploča ne izdaje komande već isključivo šalje potvrde prijema poruka i odgovarajuće podatke, kao odgovor na zadatu komandu. Kao treći uređaj se koristi **Bluetooth2 Click** pločica koja je povezana na RS-485 magistralu tako da uvek prima sve poruke koje mikrokontroleri međusobno razmenjuju.

1.2 Bluetooth2 Click

Bluetooth2 Click[8] je komponenta koju proizvodi *Mikroelektronika*. Ona sadrži *Bluegiga WT41 Bluetooth 2.1* modul koji ima radni domet do 1000 m, a koristi **iWRAP** *firmware*[7]. Ova *firmware* omogućava konfigurisanje komponente slanjem *ASCII* komandi. Na slici 1.2.1 se može videti kako ta komponenta izgleda.



Slika 1.2.1: Komponenta **Bluetooth2 Click**.

Pri realizaciji ovog projekta bilo je potrebno inicijalizovati UART periferije na mikrokontrolerima, a zatim konfigurisati i *Bluetooth2 Click*. Nakon ovoga ručno je računar povezan sa ovim uredjajem preko *Bluetooth Radio*-a. Za obradu pristiglih podataka upotrebljen je **Processing**, *open-source* kompjuterski programski jezik i integrisano razvojno okruženje (IDE).

1.3 Inicijalizacija mikrokontrolera

Za ostvarivanje komunikacije podešene su UART periferije na samim mikrokontrolerima[1]. *Baud rate* je podešen na 115200, a format prenosa podataka je u vidu paketa od 8 bita +1 stop bit bez bita parnosti, sa onemogućenom *hardware flow control*-om. Koriste se po 3 pina na svakom mikrokontroleru, od kojih je jedan TX (*transmit* linija), drugi RX (*receive* linije) i EN (enable-odnosno aktiviranje ili prijema ili slanja).

1.4 Inicijalizacija *Bluetooth Click-a*

iWrap firmware[7] ima 2 moda rada, komandni i mod za razmenu podataka. U komandnom modu ASCII komande se mogu poslati kako bi se izvršile određene radnje ili da bi se promenila konfiguraciona podešavanja. U komandni mod se ulazi po uspostavljanju napajanja po *default*-u, i u njemu se ostaje sve dok nije uspostavljena nijedna *Bluetooth* konekcija. Po uspostavljanju veze, on automatski prelazi u mod razmene podataka i tada je podatke moguće slati ili primiti. Po *default*-u *iWrap* koristi već podešenu brzinu i formu paketa UART-a koji su podešeni na mikrokontrolerima. U *iWrap User Guide*-u su navedene sve komande i način na koji se mogu koristiti za podešavanje ovog modula. Neke od korišćenih u ovom projektu su:

- **RESET** - Komanda za softverski reset modula
- **SET BT NAME** {ime} - Komanda za podešavanje imena uređaja
- **SET BT AUTH** {pin kod} - Komanda za postavljanje autentifikacione šifre uređaja

Napisane su funkcije `initBT()`, kojom se inicijalizuje ova periferija pre povezivanja sa računarom, tako što joj se nakon reset, promeni ime i pin kod za pristup i `BT_Send(const * char text)` za slanje pojedinačnih komandi u obliku stringa (niz ASCII karaktera koji se obavezno završava simbolom `'\n'`).

```
1
2 void initBT( void )
3 {
4     /* Vracanje Bluetooth Click-a na fabricka podesavanja. */
5     //BT_Send("SET RESET\n\0");
6     //sleep( 1000 );
7
8     /* Softverski reset uredjaja. */
9     BT_Send("RESET\n\0");
10    DelayUSART( 10000 );
11
12    /* Postavljanje novog imena Bluetooth Click-a. */
13    BT_Send("SET BT NAME ALEKSA\n\0");
14    DelayUSART( 15000 );
15
16    /* Postavljanje sifre za koriscenje BLuetooth Click-a. */
17    BT_Send("SET BT AUTH * 3006\n\0");
18    DelayUSART( 15000 );
19
20 }
21
```

Kod 1 : Prikaz funkcije `initBT()`

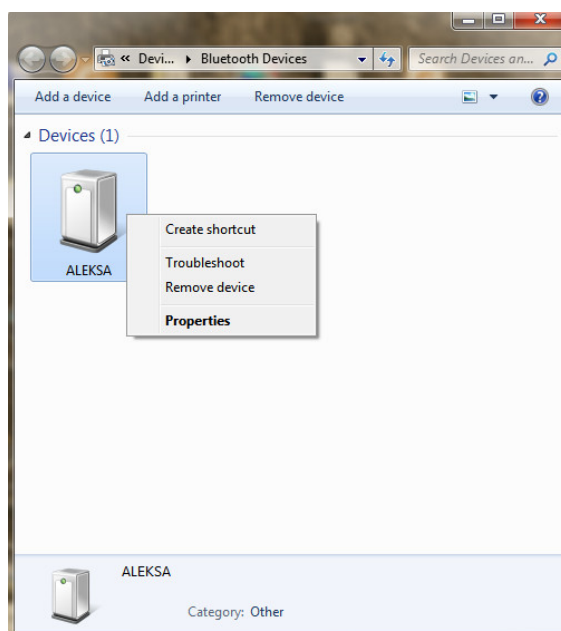
```

1 void BT_Send( const char *text )
2 {
3     sending_length = strlen( text ) ; /* Velicina poruke bez znaka '\0'. */
4     int i = 0;
5     for( i = 0; i < sending_length; i++)
6     {
7         sending_array[ i ] = text[ i ];
8     }
9     sending_iterator = 0;
10    GPIO_SetBits( GPIOC, GPIO_Pin_12 );
11
12    USART_SendData(USART3, sending_array[ sending_iterator++ ]); /* Slanje
13    podatka. */
14 }

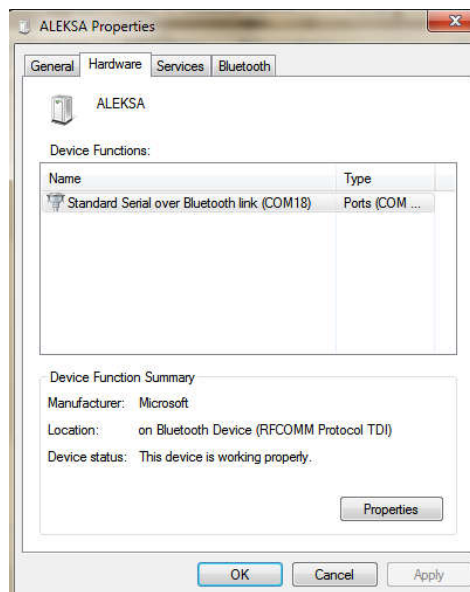
```

Kod 2 : Prikaz funkcije BT_send()

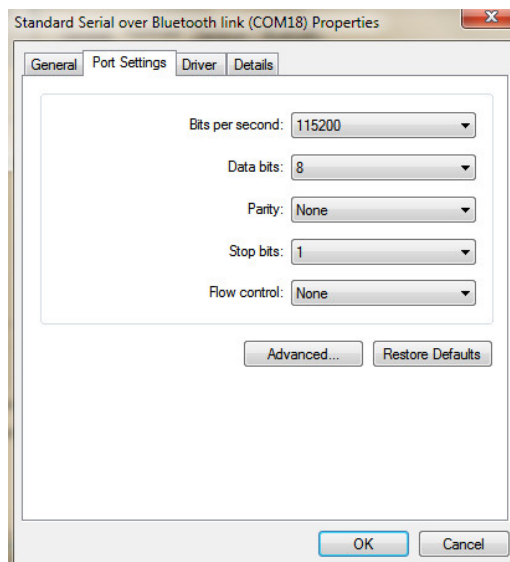
Povezivanje *Bluetooth* modula sa računarom se vrši manuelno. Dodaje se novi uređaj nakon čega je potrebno izvršiti podešavanja serijskog porta na računar koji je automatski dodeljen novopovezanom uređaju. U *Bluetooth device* prozoru potrebno je desnim klikom izabrati opciju *Properties* (slika 1.4.1). Potom u *Hardware tab*-u treba kliknuti opciju *Properties*(slika 1.4.2). U *tab-u Port Settings* treba odabrati parametre kao što su podešeni na mikrokontrolerima za UART periferiju(slika 1.4.3).



Slika 1.4.1: Korak 1



Slika 1.4.2: Korak 2



Slika 1.4.3: Korak 3

Ovako povezan *Bluetooth* modul prelazi iz *Command mode*-a u *Data mode* i spreman je za prijem podataka,tj. odašiljanje signala.

2 Komunikacijski protokol i interfejs

Kako se putem UART-a šalju bajtovi, odnosno 8-bitni podaci, potrebno je ustanoviti protokol, kako bi se uspešno poslala, odnosno primila (dekodovala) poruka.

2.1 *Communication (Main) Board*

Postoje dva tipa poruka na ovom mikrokontroleru koje se šalju ka *Motion Board*-u:

- jednostavne komande sa i bez zahtevanja povratnih podataka
- složene komande koje pored instrukcija sadrže određene podatke

Implementirane su funkcije:

- `issueSimpleCommand()`
- `issueComplexCommand()`
- `checkAndSend()`
- `receiveByte()`
- `decodeMessage()`
- `waitAck()`

Funkcijom `void issueCommand(CommandNameType command, int receiver_ address, uint16_ t data)` se zadaje komanda ploči za kretanje. Kao argumente ima neku od komandi iz tabele 2.1, adresu uređaja kome se šalje (u ovom slučaju adresu *Motion Board*-a) i eventualno 16-bitni podatak. Sa `void issueSimpleCommand(char command)` zadaju se jednostavne komande bez pakovanja pratećih podataka. Za komande kojima se podešava *Motion Board* i ne očekuje odgovor u vidu određenih podataka, kao što su STOP, ULTRASOUND_ ON, ULTRASOUND_ OFF i START_ RUNNING vraća se Ack(*acknowledge*) poruka, kao znak uspešno primljene i obavljene instrukcije. Neke od navedenih komandi kao što su CHECK_ ARRIVE, CHECK_ SENSOR, SEND_ POSITION po njihovom prijemu rezultuju ne slanjem Ack poruke već slanjem poruke kojom se prosleđuju tražene informacije.

Sa `void issueComplexCommand(char command, uint16_ t data)` zadaju se složenije komande sa slanjem dodatnih podataka. Takve instrukcije su PRESCALER (podešavanje brzine), MOVE_ FORWARD i MOVE_ BACKWARD (koliko se pomeriti u određenom smeru), ROTATE_ RIGHT i ROTATE_ LEFT (za koliko izvršiti rotaciju). Za svaku od ovih instrukcija *Motion Board* vraća Ack poruku.

Sa funkcijom `checkAndSend()` vrši se dodavanje kontrolne sume kao poslednjeg bajta poruke i pozivanje već definisane funkcije `USART_ SendData()` kojom se u registar podataka

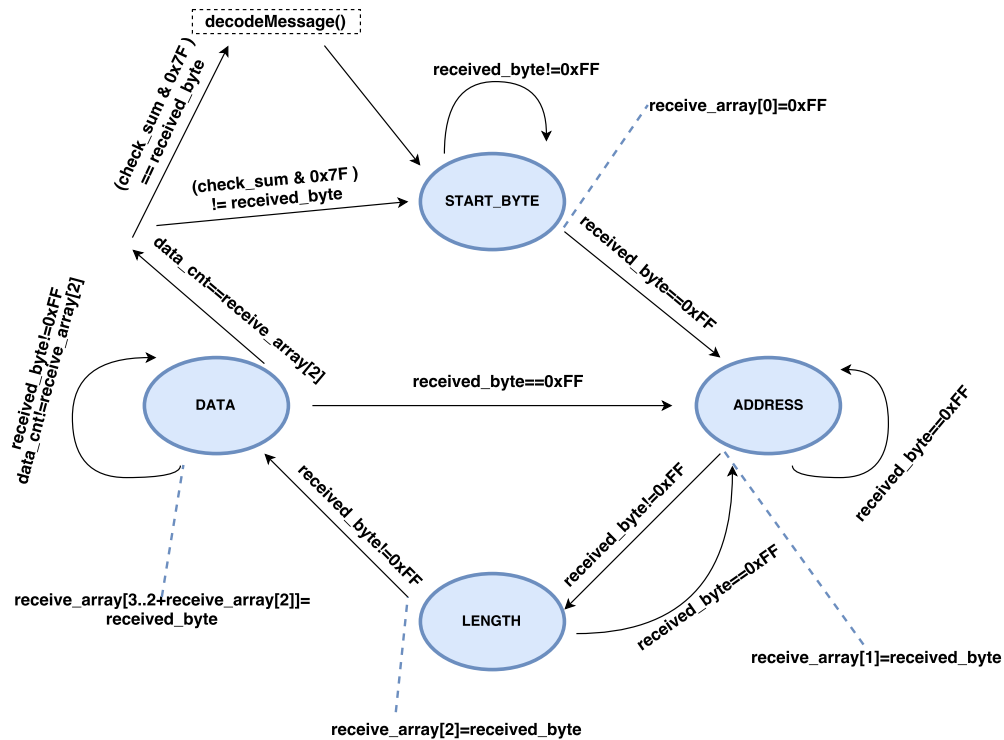
USART periferije smešta prvi podatak poruke, a zatim u okviru prekidne rutine i svaki sledeći, sve dok se ne dovrši slanje poslednjeg bajta u okviru jedne poruke.

Nakon zadavanja instrukcija koje ne zahtevaju da im određeni podaci budu poslani radi provere ispravnog prijema poslate poruke, moguće je pozvati funkciju *waitAck()* kojom se čeka na prijem Ack poruke. Ukoliko ova poruka potvrde izostane, poruka se šalje ponovo.

Tabela 2.1: Komande na *Motion Board*-u

Naziv komande	Kod	Tip komande	Dodatni podaci	Slanje Ack
STOP	0x0A	jednostavna	ne	da
CHECK _ARRIVE	0xFC	jednostavna	da	ne
ULTRASOUND _ON	0x11	jednostavna	ne	da
ULTRASOUND _OFF	0x12	jednostavna	ne	da
PRESCALER	0xFB	složena	ne	da
MOVE _FORWARD	0x04	složena	ne	da
MOVE _BACKWARD	0x05	složena	ne	da
ROTATE _LEFT	0x06	složena	ne	da
ROTATE _RIGHT	0x07	složena	ne	da
START _RUNNING	0xFA	jednostavna	ne	da
CHECK _SENSOR	0xF9	jednostavna	da	ne
SEND _POSITION	0xFC	jednostavna	da	ne

Što se tiče prijema podataka, implementirana je mašina stanja, koja je prikazana na slici 2.1.1. Poruke se primaju u prekidnoj rutini pozivima funkcije `receiveByte()`. Pristizanjem bajta 0xFF, prelazi se iz stanja `START_BYTE` u stanje `ADDRESS`. Ukoliko sledeći bajt nije 0xFF, prelazi se u stanje `LENGTH`, dok se u suprotnom ostaje u istom stanju. Nakon prijema trećeg bajta, ukoliko je različit od 0xFF, prelazi se u stanje `DATA`, dok u suprotnom prelazi u `ADDRESS` stanje. Sledi primanje broja podataka koji nam je unapred poznat na osnovu trećeg bajta (`LENGTH`). Ako se pre kraja primljenih poruka primi 0xFF, prelazi se u `ADDRESS` stanje. Po završetku prijema poruke, ispituje se kontrolna suma. Ukoliko se podaci slažu, poruka je uspešno primljena i nakon pozivanja funkcije `decodeMessage()` (koja vrstu primljenih podataka razlikuje isključivo po trećem bajtu u kome je smeštena dužina cele poruke) prelazi se u stanje `START_BYTE`. U suprotnom prelazi se u stanje `START_BYTE`.



Slika 2.1.1: Mašina stanja za prijem poruka na strani *Communication Board-a*.

2.2 Motion (Slave) Board

Postoje dva tipa poruka na ovom mikrokontroleru koje se šalju ka *Communication Board-u*:

- *acknowledge* poruke, kojima se potvrđuje ispravan prijem komandi sa glavne ploče
- poruke sa podacima kao odgovor na zadatu komandu sa glavne ploče

Implementirane su funkcije:

- `Response()`
- `sendAck()`

U okviru UART prekidne rutine se vrši prijem poruke. Prvi bajt je 0xFF i predstavlja početak nove poruke. Sledeći, drugi bajt je bajt koji predstavlja adresu uređaja kome je poslata poruka (dakle u ovom slučaju adresu koja je dodeljena *Motion Board-u*). Trećim bajtom je data dužina poruke. U narednim bajtovima će biti sačuvana komanda, eventualno podaci i na kraju kontrolna suma. Ukoliko se kontrolna suma slaže sa primljenim podacima, poziva se funkcija `void Response(void)`. Ova funkcija ispituje bajt u kome se nalazi komanda i shodno tome izvršava odgovarajuće akcije. Kao što je već pomenuto, pri komandama kojima se vrše određena podešavanja na ploči za kretanje (bez povratnih podataka) se šalje Ack poruka, po uspešno obavljenim radnjama pozivanjem funkcije `void sendAck(void)`.

3 Prenos podataka sa platforme preko *Bluetooth*-a do računara

Od interesa za ovaj projekat je komanda za slanje pozicije. Na samoj platformi postoje motori na kojima su instalirani enkoderi, kojima se kontroliše i prati pozicija platforme u prostoru. Po slanju komande SEND_POSITION od strane komunikacione ploče, na strani ploče za kretanje, poziva se funkcija `void SendPosition(void)` koja šalje proračunatu trenutnu poziciju (X, Y koordinate, ugao koji platforma zauzima i *flag* koji sadrži informaciju o stizanju na zadatu poziciju).

```
1 void SendPosition( void )
2 {
3     int x_inp, y_inp, xy_inp;
4     sending_length=25;
5     sending_array[0] = 0xFF;
6     sending_array[1] = ADDR|0x40;
7     sending_array[2] = sending_length-3;
8     sending_array[3] = (apsolutnaPozicija.x & 0x0000000F);
9     sending_array[4] = (apsolutnaPozicija.x & 0x000000F0)>>4;
10    sending_array[5] = (apsolutnaPozicija.x & 0x00000F00)>>8;
11    sending_array[6] = (apsolutnaPozicija.x & 0x0000F000)>>12;
12    sending_array[7] = (apsolutnaPozicija.x & 0x000F0000)>>16;
13    sending_array[8] = (apsolutnaPozicija.x & 0x00F00000)>>20;
14    sending_array[9] = (apsolutnaPozicija.x & 0x0F000000)>>24;
15    sending_array[10]=(apsolutnaPozicija.x & 0xF0000000)>>28;
16
17    sending_array[11]=(apsolutnaPozicija.y & 0x0000000F);
18    sending_array[12]=(apsolutnaPozicija.y & 0x000000F0)>>4;
19    sending_array[13]=(apsolutnaPozicija.y & 0x00000F00)>>8;
20    sending_array[14]=(apsolutnaPozicija.y & 0x0000F000)>>12;
21    sending_array[15]=(apsolutnaPozicija.y & 0x000F0000)>>16;
22    sending_array[16]=(apsolutnaPozicija.y & 0x00F00000)>>20;
23    sending_array[17]=(apsolutnaPozicija.y & 0x0F000000)>>24;
24    sending_array[18]=(apsolutnaPozicija.y & 0xF0000000)>>28;
25
26    sending_array[19]=(apsolutnaPozicija.theta & 0x000F);
27    sending_array[20]=(apsolutnaPozicija.theta & 0x00F0)>>4;
28    sending_array[21]=(apsolutnaPozicija.theta & 0x0F00)>>8;
29    sending_array[22]=(apsolutnaPozicija.theta & 0xF000)>>12;
30
31    // Ready bit, vraca da li je robot stigao na zeljenu destinaciju
32    x_inp=0;
33    y_inp=0;
34    xy_inp=0;
35    if ((trenutna_pozicija_X<zapamcena_pozicija_X+INP_TOLERANCE)&&(
36        trenutna_pozicija_X>zapamcena_pozicija_X-INP_TOLERANCE)) x_inp=1;
37    if ((trenutna_pozicija_Y<zapamcena_pozicija_Y+INP_TOLERANCE)&&(
38        trenutna_pozicija_Y>zapamcena_pozicija_Y-INP_TOLERANCE)) y_inp=1;
39    if ((x_inp==1)&&(y_inp==1)) {
40        xy_inp=1;
41    }
```

```

39     }
40
41     sending_array[23]=xy_inp?0x71:0x70;
42
43     schksm=0;
44     for (int i=1;i<sending_length-1;i++){
45         schksm+=sending_array[i]&0xFF;
46     }
47     sending_array[sending_length-1]=schksm&0x7F;
48
49     GPIO_SetBits(GPIOC,GPIO_Pin_12); //enejbluje se RS485 predaja
50     USART_SendData(USART3, sending_array[0]); //salje prvi podatak
51     sending_iterator=1;
52 };
53

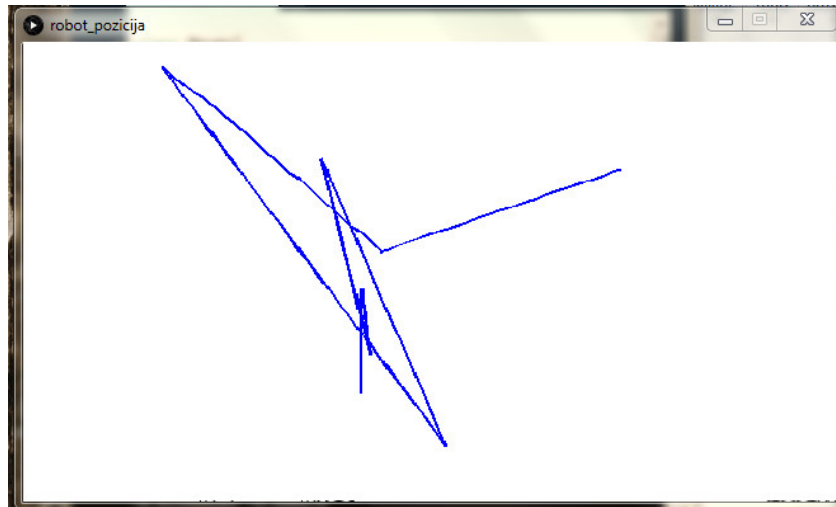
```

Kod 3 : Prikaz funkcije SendPostition()

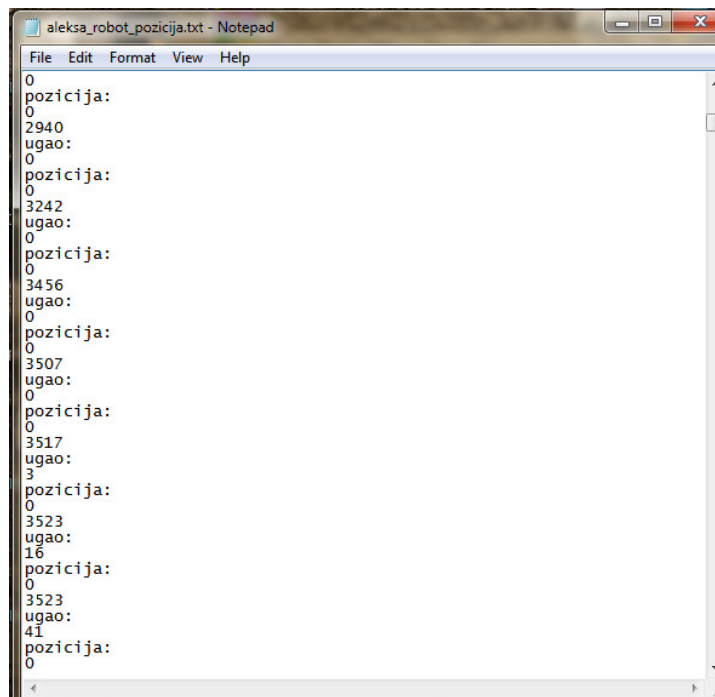
Ovi podaci se šalju putem RS-485 magistrale na koju je povezan *Bluetooth* modul. Svi podaci, odnosno poruke koje razmenjuju *master* i *slave* ploče, emituje Bluetooth, pa tako i ovu poruku. U programskom jeziku **Processing** (slika 3.0.1) korišćenjem biblioteke *Serial*[6], otvoren je serijski port po kome *Bluetooth* modul šalje podatke ka računaru. Realizovana je mašina stanja na sličan način kao i na mikrokontrolerima s tom razlikom što nas prilikom dekodovanja poruke interesuje samo ona koja sadrži podatke o poziciji platforme. Dobijene vrednosti se upisuju u datoteku (slika 3.0.3) i u realnom vremenu se pozicija iscrtava na ekranu (slika 3.0.2).



Slika 3.0.1: *Processing* logo



Slika 3.0.2: Iscrtavanje kretanja (pozicije) platforme u realnom vremenu



```
0  
pozicija:  
0  
2940  
ugao:  
0  
pozicija:  
0  
3242  
ugao:  
0  
pozicija:  
0  
3456  
ugao:  
0  
pozicija:  
0  
3507  
ugao:  
0  
pozicija:  
0  
3517  
ugao:  
3  
pozicija:  
0  
3523  
ugao:  
16  
pozicija:  
0  
3523  
ugao:  
41  
pozicija:  
0
```

Slika 3.0.3: Izlazna datoteka u kojoj su sačuvani podaci o kretanju

Zaključak

Realizacijom ovog projekta postignut je početni cilj a to je uspešno projektovana komunikacija između različitih uređaja u jednom sistemu. Ostavljena je mogućnost za dalju obradu i vizuelizaciju podataka na računaru usled zaista fleksibilnog protokola komunikacije koji postoji na platformi i u kome je lako dodavanje novih komandi (instrukcija). Utvrđeno je da uređaj *Bluetooth2 Click* predstavlja odličan izbor za *wireless* komunikaciju, s obzirom da je jednostavan za korišćenje, relativno niske cene, velike pouzdanosti itd. Takođe demonstrirana je upotreba zaista moćnog programskog jezika *Processing*, koji doživljava ekspanziju usled velikog broja dostupnih biblioteka kako za brzu obradu velike količine podataka tako i za njihovu vizuelizaciju, a takođe i za obradu slika, programiranje animacija i interaktivnih 2D, 3D aplikacija.

Literatura

- [1] *Discovering the STM32 Microcontroller*
Geoffrey Brown, 2015.
- [2] *Low-Power, Slew-Rate-Limited RS-485/RS-422 Transceivers*
<https://datasheets.maximintegrated.com/en/ds/MAX1487-MAX491.pdf>
Maxim Integrated, 2014.
- [3] *Reference manual STM32F100xx advanced ARM-based 32-bit MCUs*
www.st.com
STMicroelectronics, 2011.
- [4] *User Manual STM32VLDISCOVERY STM32 value line Discovery*
www.st.com
STMicroelectronics, 2011.
- [5] Materijal sa sajta, predavanja sa predmeta 32-bitni mikrokontroleri i primena
<http://tnt.etf.bg.ac.rs/~ms1bmp/>
- [6] Tutorials, Examples, Books, Handbook, Reference
<https://www.processing.org/>
- [7] *iWrap3 User Guide*
https://www.sparkfun.com/datasheets/Wireless/Bluetooth/iWRAP3_User_Guide.pdf
BlueGiga, 2008.
- [8] Bluetooth2 Click Examples, Bluetooth2 Click User Manual
[//www.mikroe.com/click/bluetooth2/](http://www.mikroe.com/click/bluetooth2/)