

Univerzitet u Novom Sadu  
Fakultet tehničkih nauka  
Departman za energetiku, elektroniku i telekomunikacije  
Katedra za elektroniku

**Smerovi: Mikroračunarska elektronika, Telekomunikacije i obrada signala i  
Elektroenergetika**

## **MIKROPROCESORSKA ELEKTRONIKA**

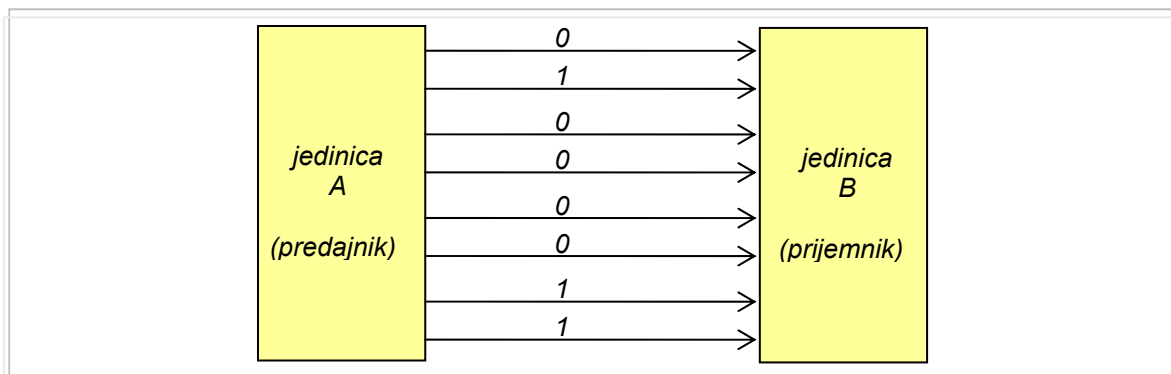
**Beleške sa predavanja #10:**

**Serijski prenos podataka**

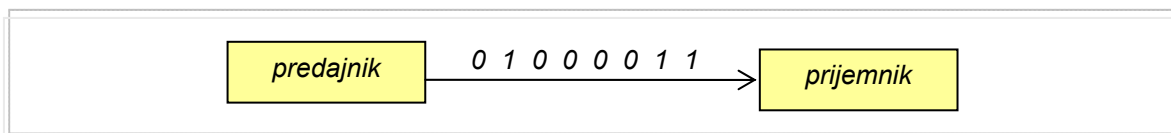
Pripremio: prof. dr Veljko Malbaša  
Novi Sad, januar 2011. godine

## Serijska komunikacija

U dosadašnjim razmatranjima upoznali smo se sa paralelnim prenosom podataka kod koga se između dve jedinice istovremeno prenosi dva ili više bita. Na sledećoj slici ilustrovan je paralelni prenos bajtova (8 bita) od jedinice A (predajnik) do jedinice B (prijemnik). Kod paralelnog prenosa postoji poseban provodnik za prenos svakog od ukupno 8 bita u jednom bajtu. Na primer, bajt  $43_{16}$  (odnosno 0100 0011 u binarnom brojnem sistemu) od jedinice A do jedinice B prenosi se tako što za svaki bit postoji poseban provodnik tako da svi biti od jedinice A do jedinice B stižu istovremeno, videti primer na sledećoj slici.

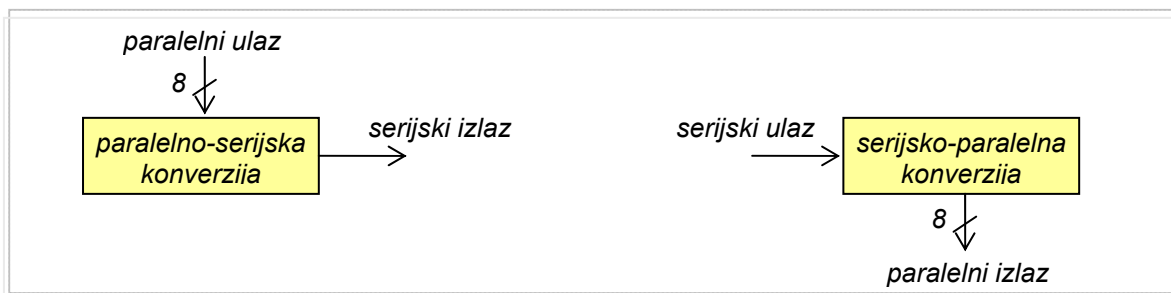


Paralelni prenos podataka pogodan je za velike brzine prenosa i relativno kratka rastojanja između jedinica. Međutim, već kod rastojanja veća od nekoliko metara, cena provodnika počinje da raste i paralelni prenos nije praktičan. Cilj paralelnog prenosa je da se smanji cenu provodnika tako što se prenosi samo jedan bit u jednom trenutku, odnosno koristi samo jedan provodnik za prenos podataka, slika dole.



Pošto je u jednom trenutku moguće preneti samo jedan bit, očigledan nedostatak serijskog prenosa je u tome što je sporiji u odnosu na paralelni prenos. Intuitivno, paralelni prenos sa  $n$  provodnika brži je  $n$  puta u odnosu na serijski prenos, naravno pod pretpostavkom da su svi ostali parametri prenosa isti.

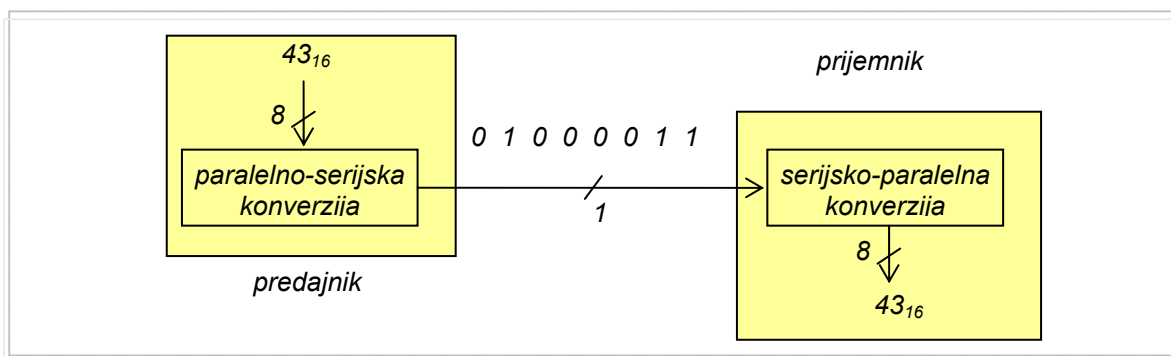
Kod serijskog prenosa podataka na predajnoj strani mora da se obavi paralelno-serijska, a na prijemnoj strani serijsko-paralelna konverzija. Na sledećoj slici levo prikazan je princip paralelo-serijske, a na slici desno serijsko-paralelne konverzije. Obe vrste konverzije zasnovane su na primeni pomeračkog registra sa serijskim i paralelnim ulazima i izlazima.



Pomerački registar sa paralelnim ulazom i serijskim izlazom koristi se na predajnoj strani za konverziju paralelnih podataka u serijskih niz bita. Na primer, ako se bajt  $43_{16}$  (0100 0011 u binarnom brojnem sistemu) paralelno upiše u registar, onda će se u ritmu sinhronizacionog signala biti pomerati u desnu stranu tako da će se serijskom izlazu prvo pojaviti najmanje značajan bit (u navedenom primeru logička 1), zatim sledeći bit (u primeru logička 1), i tako dalje. Na kraju će se preneti najznačajniji bit (u primeru je to logička 0).

Pomerački registar sa serijskim paralelnim ulazom i paralelnim izlazom koristi se na prijemnoj strani za suprotnu konverziju, odnosno konverziju ulaznih bita koji se serijski primaju u paralelni podatak. U prethodnom primeru, ako pomerački registar prvo dobije logičku 1, zatim logičku 1, pa logičku 0, i tako dalje, i na kraju logičku 0, onda će se ovi biti pomerati za po jedno mesto dok se u registru ne formira paralelni podatak  $43_{16}$  koji je dostupan u paralelnom obliku na paralelnom izlazu pomeračkog registra.

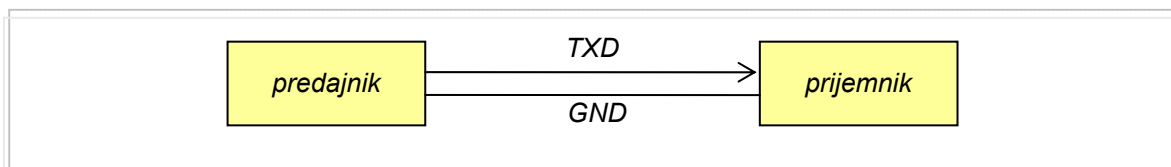
Koncept serijskog prenosa prikazan je na sledećoj slici. U predajniku se paralelni podatak upisuje u paralelno-serijsku konverziju, zatim se biti pomeraju udesno i jedan po jedan preko serijske veze prenose do prijemnika. Na prijemnoj strani serijski primljeni biti dolaze na serijsko-paralelnu konverziju, pomeraju udesno i asemblira originalni podatak koji je u paralelnom obliku dostupan na paralelnom izlazu registra.



Kod serijskog prenosa ima nekoliko pitanja od čijih odgovora zavisi tip veze i način serijskog prenosa:

- Da li je serijska veza jednosmerna ili dvosmerna i ako je veza dvosmerna, da li može da se istovremeno obavlja predaja i prijem?
- Kako se vrši sinhronizacija rada predajnika i prijemnika?
- Da li postoji i kako se obavlja kontrola ispravnosti prenosa podataka?
- Da li se podaci prenose reč po reč ili u blokovima?

U pogledu smera veze, razlikujemo jednosmernu (simpleks), polu-dvosmernu (polu-dupleks) i dvosmernu (dupleks) vezu. Kod jednosmerne veze između predajnika i prijemnika postoje dva provodnika, jedan za prenos serijskih podataka (na slici dole obezbežen sa *TXD*) i drugi za masu (na slici obeležen sa *GND*) videti sledeću sliku.



Kod dvosmerne veze postoje tri provodnika, po jedan za vezu u oba smera i jedan zajednički provodnik za masu. Polu-dvosmerna veza ima dva provodnika, jedan za masu i drugi koji, po

dogovoru između predajnika i prijemnika, u jednom periodu vremena može da se koristi za serijski prenos u jednom, a u drugom periodu vremena za serijski prenos u drugom smeru.

Sinhronizacija rada između predajnika i prijemnika je važna jer je osnova za ispravan serijski prenos podataka. Sinhronizacija pre svega podrazumeva da predajnik i prijemnik rade na istoj brzini prenosa, odnosno da je brzina prenosa izražena u broju bita u sekundi ista na predajnoj i prijemnoj strani.

Sinhronizacija rada može da se vrši i razmenom upravljačkih signala, odnosno rukovanjem (engleski *handshaking*) između predajnika i prijemnika. Naravno, rukovanje zahteva dodatne provodnike za prenos upravljačkih signala između predajnika i prijemnika.

Kontrola ispravnosti prenosa podataka može da se obavlja na različitim nivoima. Na primer, tačnost prenosa podataka može da se obavi dodavanjem bita parnosti na predaji i proverom bita parnosti na prijemu.

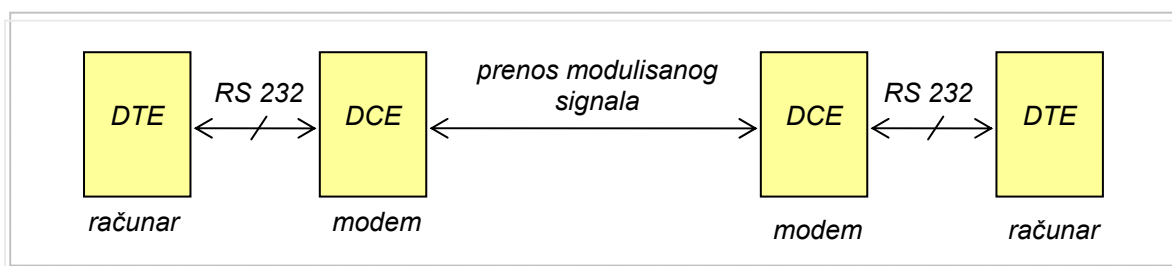
Podaci mogu da se prenose reč po reč ili u blokovima u kojima se nalazi više reči. Prenos reč po reč obično se naziva *asinhroni* prenos, kod koga se za svaku reč prvo šalje početni (*start*) a na kraju reči završni (*stop*) bit. Kod sinhronog prenosa obično se prenosi blok podataka u kome postoji početna i završna reč, a pored serijskih podataka prenosi se i sinhronizacioni signal od predajnika do prijemnika.

Serijski prenos podataka ilustrovaćemo na primeru standarda RS 232.

### **Standard RS 232 za serijski prenos podataka**

Standard RS 232 ustanovljen je još šezdesetih godina prošlog veka. Iako je zastareo i podržava spori prenos podataka, još uvek je popularan jer je jednostavan za primenu, pouzdan i široko prihvaćen.

Standard RS 232 uveden je u vreme kada su terminali povezivani sa centralnim računarom preko modema, pa su osnovni termini u standardu prilagođeni takvoj primeni. Na sledećoj slici prikazane su osnovne jedinice koje učestvuju u serijskom prenosu podataka primenom standarda RS 232.



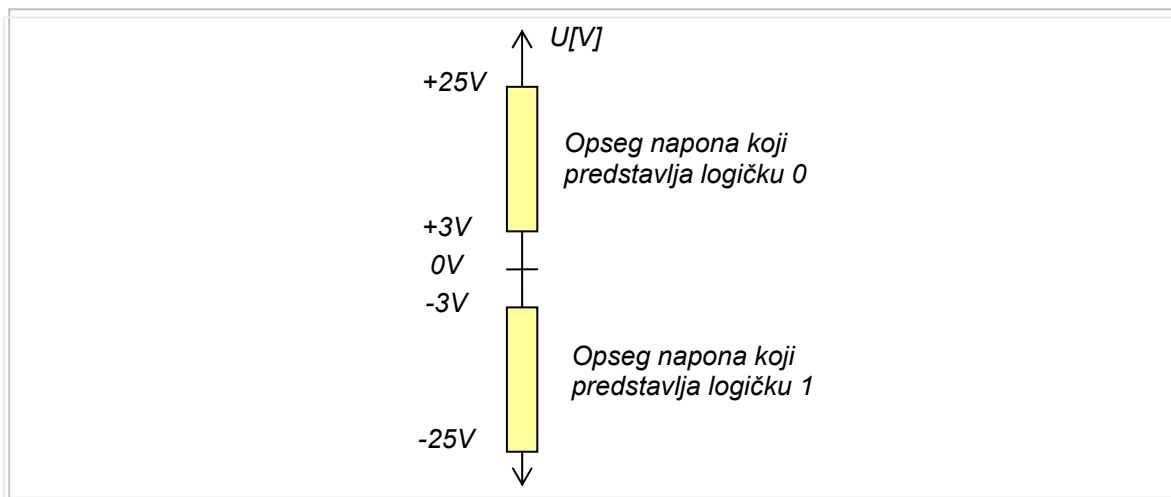
Računari se u RS 232 terminologiji nazivaju *DTE* (engleski *Data Terminal Equipment*) a modemi *DCE* (engleski *Data Communication Equipment*). Standard RS 232 definiše razmenu signala između DTE i DCE. Neki od ovih signala služe za upravljenje modemom koji je priključen na telefonsku liniju, ali u ovom tekstu nećemo se baviti tim signalima.

Uloga modema je da prihvati serijske digitalne signale od DTE, moduliše ih i modulirani signal pošalje drugom modemu. U tipičnom slučaju, modem je vezan na telefonsku liniju i prenosi signale u analognom obliku, tako što za prenos logičke 1 koristi sinusoidalni signal jedne, a za prenos logičke 0 sinusoidalni signal neke druge frekvencije.

RS 232 standard podržava asinhroni i sinhroni dvosmerni serijski prenos podataka i obuhvata električnu, mehaničku, funkcionalnu i proceduralnu specifikaciju. U ovom tekstu opisaćemo deo standarda koji se odnosi na asinhroni prenos podataka.

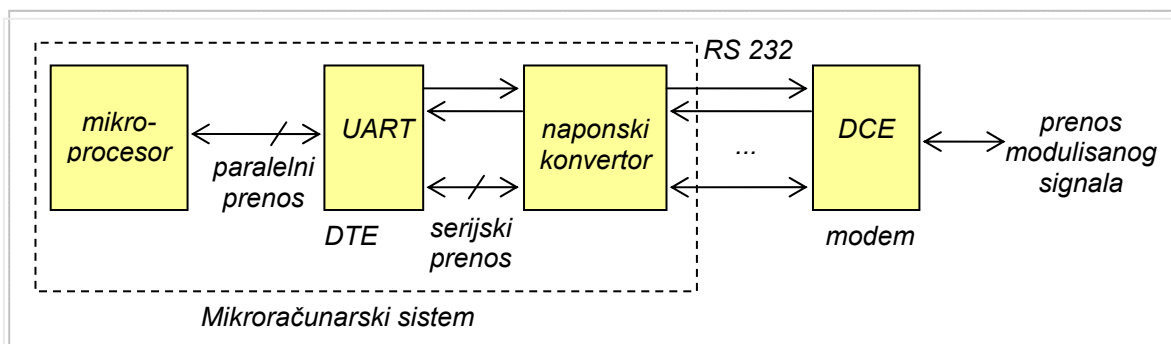
### Električna specifikacija

U električnom pogledu, logička 1 predstavljena je naponom u opsegu od -3 do -25 V, a logička 0 naponom od +3 do +25 V. Napon između -3V i +3V predstavlja nedefinisan opseg. U praksi se često logička 1 predstavlja naponom od -12V a logička 0 naponom od +12 V.



### Funkcionalna specifikacija

Tipična struktura funkcionalnih jedinica na jednoj strani serijskog prenosa podataka prikazana je na sledećem blok dijagramu. Naravno, za pravilni prenos signala, simetrična struktura treba da postoji na drugoj strani serijske veze.



Kontroler za serijski prenos podataka, engleski *Universal Asynchronous Receiver Transmitter*, skraćeno *UART*, podržava prijem i predaju serijskih podataka u asinhronom načinu rada. Serijski kontroler koji podržava i sinhroni način rada na Engleskom se naziva *Universal Synchronous/Asynchronous Receiver Transmitter* ili *USART*.

Kod predaje, UART-a prihvata podatke u paralelnom obliku od mikroprocesora, pretvara ih u serijski oblik, ukoliko treba ubacuje dodatne bite (na primer za sinhronizaciju) i u serijskom obliku šalje modemu koji moduliše serijski signala. Međutim, pošto su električne karakteristike signala u mikroračunarskom sistemu drugačije od električnih signala definisanih standardom RS232, između UART-a i modema stavlja se naponski konvertor za električno prilagođenje signala.

Signali UART-a obično su kompatibilni sa TTL standardom, pa tako napon na izlazu koji je između 0 V i 0.4 V predstavlja logičku 0, a napon između 2.6 V i 5 V predstavlja logičku 1. Uloga naponskog konvertora da je ove napone prevede u napon u opsegu od +3 V do +25 V za logička 0 i napon u opsegu od -3 V do -25 V za logičku 1. Signali na izlazu naponskog konvertora u skladu su sa električnom specifikacijom RS232 standarda i mogu da se vode na modem.

Sličan postupak je kod prijema podataka, ali ovoga puta u suprotnom smeru. Modem demoduliše ulazni modulirani signal, dobijeni serijski signal je u skladu sa RS232 specifikacijom, pa ga naponski konvertor prevodi u signale koji su u električnom pogledu kompatibilni sa TTL signalima. UART prevodi podatak iz serijskog u paralelni oblik i u tom obliku predaje mikroprocesoru.

Između mikroračunara i modema razmenjuje se sledeći signali koji su deo RS23 standarda. Treba imati u vidu da RS232 standard obuhvata i druge signale, koji nisu od interesa za jednostavne primene ovog standarda.

*DTR (Data Terminal Ready)*

Mikroračunar javlja modemu da je spreman za serijski prenos podataka. (Izlazni signal mikroračunara)

*DSR (Data Set Ready)*

DTE javlja mikroračunaru da je spreman za serijski prenos podataka. (Ulazni signal mikroračunara)

*RTS (Request To Send)*

Izlazni signal koji mikroračunar prevodi u aktivno stanje ako ima podatke koje želi preneti preko serijske RS232 veze. (Izlazni signal mikroračunara)

*CTS (Clear To Send)*

Signal kojim DTE javlja mikroračunaru da je spreman za serijsku predaju podataka. (Ulazni signal mikroračunara)

*TxD (Transmitted Data)*

Serijski izlazni signal za podatke. (Izlazni signal mikroračunara)

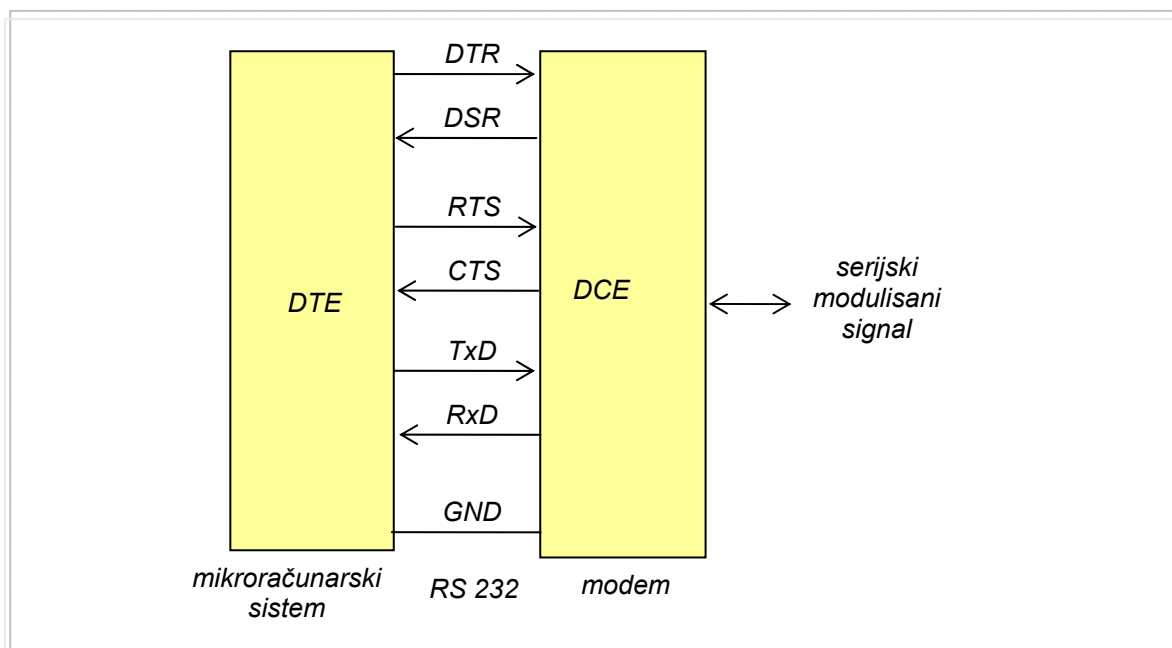
*RxD (Received Data)*

Serijski ulazni signal za podatke. (Ulazni signal mikroračunara)

*GND (Ground)*

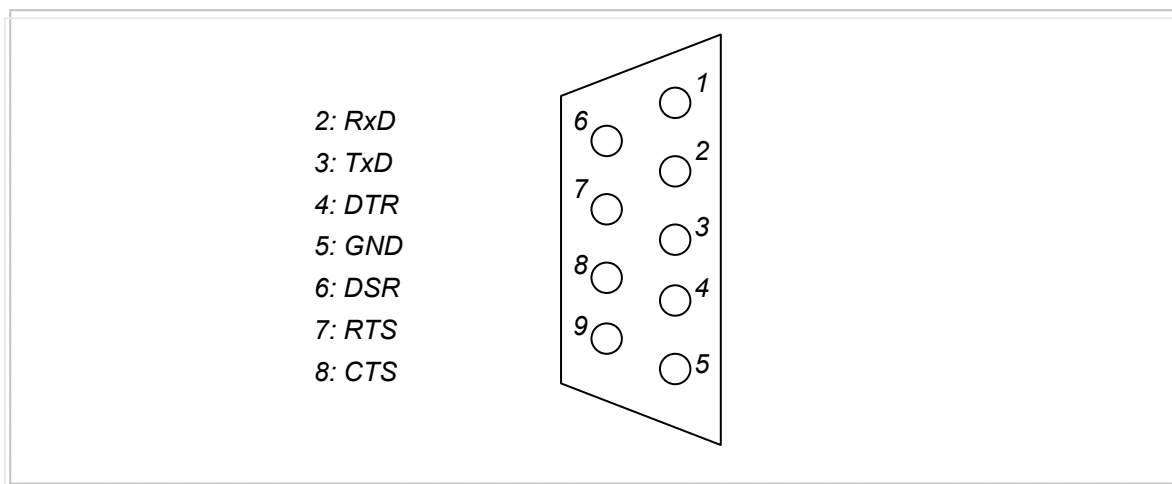
Zajednička masa.

Nazivi i smerovi signala prikazani su na sledećoj slici.



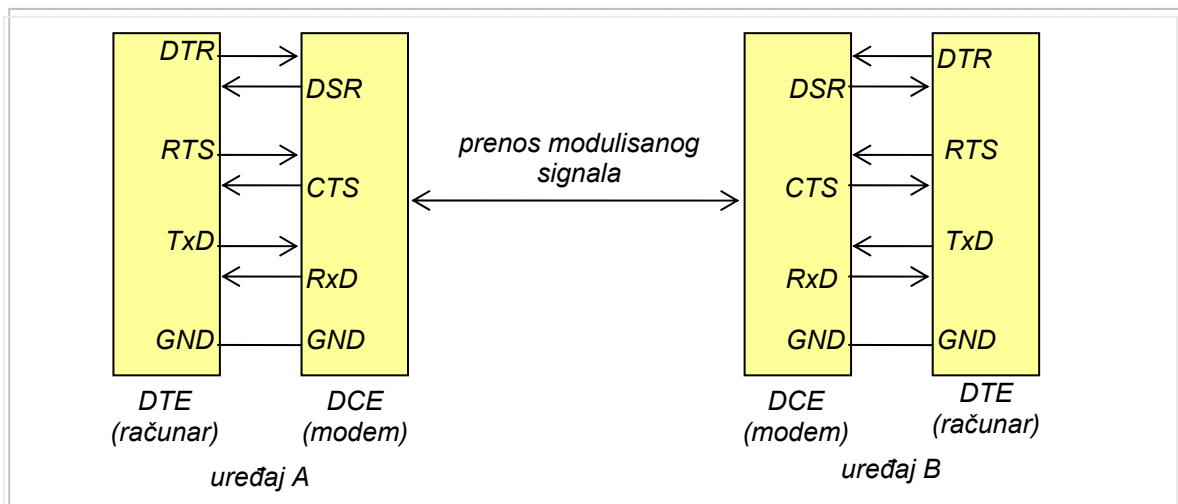
### **Mahanička specifikacija**

Originalni standard RS 232 predvideva je korišćenje konektora sa 25 nožica (naziva se DB25), koji podržava dva dupleks serijska kanala. Međutim, pošto je u praksi obično korišćen samo jedan kanal, vremenom je konektor od 25 nožica zamenjen konektorom sa 9 nožica (koji se naziva DB9). Raspored signala po nožicama konektora DB9 prikazan je na sledećoj slici.



### **Proceduralna specifikacija**

Proceduralna specifikacija obuhvata različite vrste rada, pa značenja i uloge signala zavise od toga da li se obavlja sinhroni ili asinhroni prenos ili na primer da li se prenos obavlja preko telefonske linije ili preko posebnih provodnika za prenos serijskih signala. Pretpostavimo da uređaji A i B koriste asinhronu dupleks vezu za serijski prenos podataka preko posebnih provodnika (ne koristi se telefonska linija), videti sledeću sliku.



U slučaju asinhronog prenosa preko posebnih provodnika za dupleks serijsku vezu, neformalni opis postupka prenosa između uređaja A i uređaja B izgleda ovako:

- Mikroračunari (na strani A i B) povežu se sa svojim modemima, tako da postoje posebni provodnici za prenos pojedinih signala RS 232 standarda. Dva modema povežu se posebnim provodnicima za prenos modulisanog signala.
- Mikroračunar A aktivira svoj signal DTR i tako javlja modemu A da je spreman. Mikroračunar B aktivira svoj signal DTR i javlja modemu B da je spreman.
- Aktiviranjem signala DSR modem A javlja mikroračunaru A da je spreman. Modem B aktivira signal DSR i javlja mikroračunaru B da je spreman.
- Ako mikroračunar A namerava da serijski šalje podatke aktivira svoj signal RTS.
- Ako je modem A spreman da slanje podataka, onda aktivira svoj signal CTS.
- Mikroračunar A počinje da šalje serijske podatke svom modemu A preko TxD.
- Modem A moduliše serijske podatke i šalje ih preko provodnika za serijski prenos modemu B.
- Modem B demoduliše primljeni signal i preko provodnika RxD šalje digitalni signal u serijskom obliku mikroračunaru B.

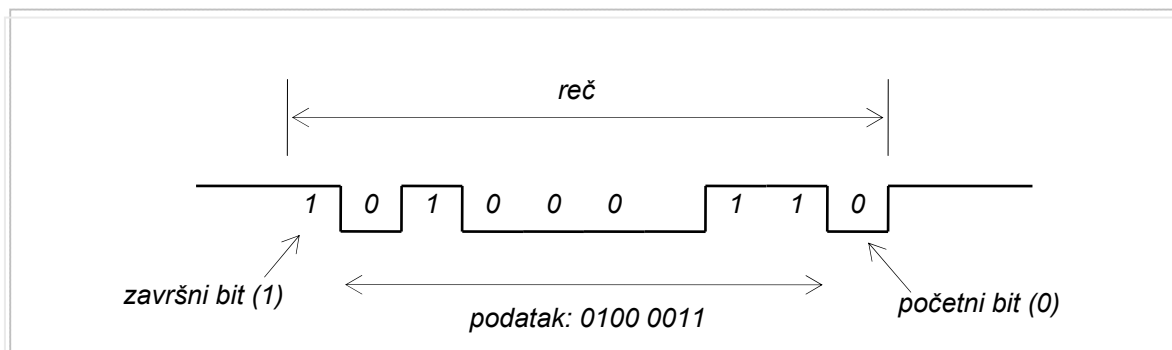
Treba primetiti da se kod prenosa podataka od strane A prema strani B, razmenom signala RTS i CTS obavlja rukovanje između mikroračunara A i modema A, ali se ne obavlja rukovanje između modema B i mikroračunara B. Standard RS 232 podrazumeva da mikroračunar radi dovoljno brzo u odnosu na brzinu serijskog prenosa, tako da je uvek spreman da prihvati serijski ulaz podataka. Imajući u vidu brzinu rada mikroračunara i brzinu serijskog prenosa standardom RS232 ovaj uslov praktično je uvek ispunjen. Drugim rečima, čim modem dobije od mikroračunara signal DTR, to znači da je mikroračunar već spreman za prijem podataka.

### **Format reči kod asinhronog prenosa**

Kod asinhronog prenosa podaci se prenose reč po reč (engleski *character*). Jedna reč sastoji se od početnog bita (engleski *start*) koji je logička 0, zatim 7 do 8 bita podatka, opcionog bita za proveru parnosti i 1 do 2 završna (*stop*) bita koji su logičke 1. Pre početka serijskog prenosa i između prenosa dve uzastopne reči, provodnik za serijski prenos drži se u stanju logičke 1.



U slučaju prenosa binarnog podatka 0100 0011 od 8 bita, ako se ne koristi bit za proveru parnosti i da postoji 1 završni bit, onda bi se preko provodnika za serijski prenos preneo niz bita prikazan na sledećoj slici.



### **Povezivanje dva računara bez modema**

Ako su dva mikroračunara sa međusobnom serijskom vezom dovoljno blizu, moguće je prenositi digitalne signale u originalnom obliku, bez korišćenja modema za modulaciju serijskog signala. U ovom slučaju RS 232 signali dva mikroračunara mogu se neposredno spojiti provodnicima. Pošto u takvoj vezi nema modema, ona se obično naziva *null modem* veza a kabel sa provodnicima naziva se *null modem* kabel.

Kod povezivanja bez modema, ostaje pitanje kako spojiti RS 232 signale mikroračunara A i B. Jasno je da prvo treba kratko spojiti mase oba mikroračunara, dok za preostala tri para signala, (DTR, DSR), (RTS, CTS) i (TxD, RxD), važe sledeća pravila.

#### **Signali (DTR, DSR)**

Po RS 232 protokolu, mikroračunar signalom DTR javlja modemu da je uključen i spreman za rad, i na sličan način, modem aktiviranjem signala DSR javlja mikroračunaru da je uključen i spreman za rad. Pošto nema modema, onda rukovanje između mikroračunara i modema nema smisla, ali signali DTR i DSR mogu da se koriste za rukovanje između mikroračunara A i mikroračunara B. Ako se izlazni signal DTR mikroračunara A dovede na ulazni signal DSR mikroračunara B i obratno, ali se izlazni signal DTR mikroračunara B dovede na ulazni signal DSR mikroračunara A, onda jedan mikroračunar javlja drugom mikroračunaru da je uključen, inicijalizovan i spreman za serijski prenos podataka.

Međutim, u jednostavnom slučaju, ako korisnik odluči da nema potrebe za rukovanjem između dva mikroračunara, dovoljno je signal DSR stalno držati u aktivnom stanju ili izlazni signal DTR kratko spojiti na ulazni signal DSR istog mikroračunara.

#### **Signali (RTS, CTS)**

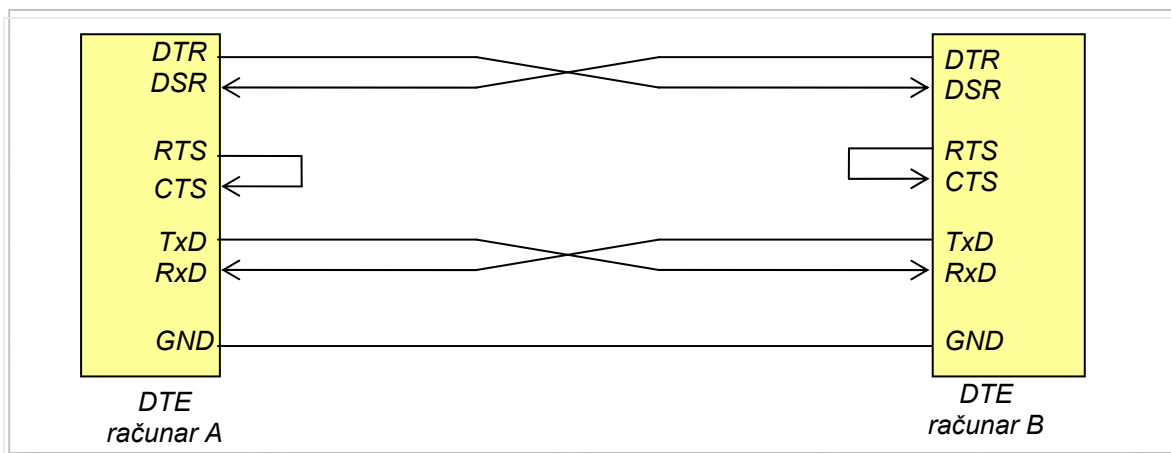
Podsetimo se da pre slanja serijskih podataka, mikroračunar aktiviranjem signala RTS proverava da li je modem spreman za predaju, a modem signalom CTS obaveštava mikroračunar da je spreman. Pošto nema modema i serijski izlaz direktno se vodi na serijski ulaz drugog mikroračunara, možemo smatrati da prenos uvek može da se obavi (naravno ako je drugi računar uključen i pripremljen za serijski prenos). Međutim, po RS 232 standardu rukovanje signalima RTS i CTS mora da se obavi, odnosno kada predajni mikroračunar aktivira RTS, slanje serijskih podataka može da počne tek kada mikroračunar primi aktivan signal CTS. Ovaj uslov može se zadovoljiti držanjem signala CTS u

aktivnom stanju ili vezivanjem izlaznog signala RTS predajnog mikroračunara na ulazni signal CTS.

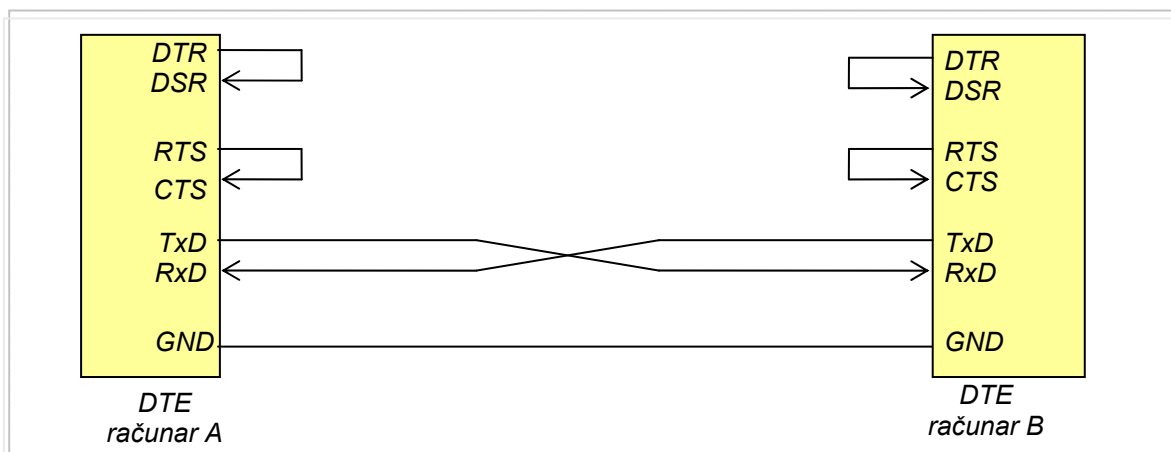
#### Signali (TxD, RxD)

Signale TxD i RxD treba ukrstiti, tako da se serijski izlaz TxD mikroračunara A dovede neposredno na serijski ulaz RxD mikroračunara B i obratno, serijski izlaz TxD mikroračunara B veže se na serijski ulaz RxD mikroračunara A. Na ovaj način serijski izlazni signal jednog mikroračunara predstavlja serijski ulazni signal drugog mikroračunara, jer nema potrebe da se serijski signal prvo moduliše i zatim demoduliše.

Na sledećem dijagramu prikazana je veza bez modema između mikroračunara A i mikroračunara B sa rukovanjem signalima DTR i DSR, pri čemu je izlazni signal RTS kratko vezan na ulazni signal CTS istog mikroračunara.

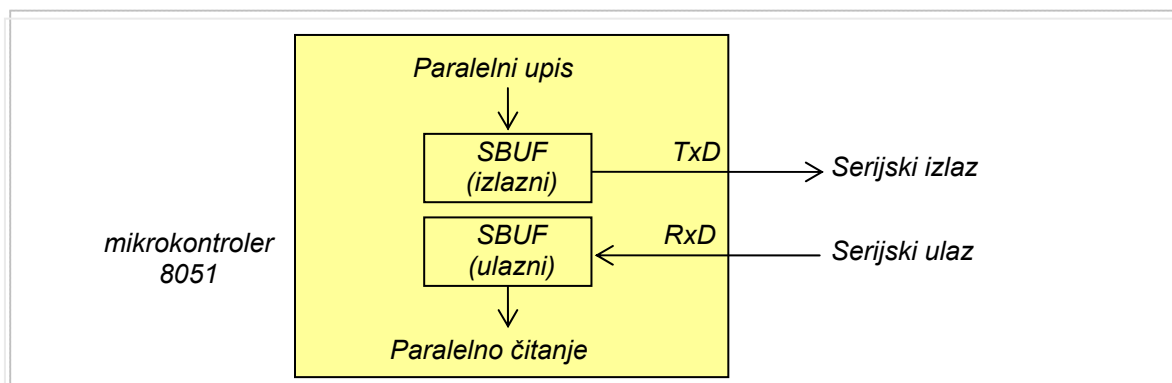


Na sledećem dijagramu prikazana je jednostavna veza bez modema između mikroračunara A i mikroračunara B bez rukovanja sa signalima DTR i DSR, pri čemu je takođe izlazni signal RTS kratko vezan na ulazni signal CTS istog mikroračunara.



#### Serijski port mikrokontrolera 8051

Mikrokontroler 8051 poseduje serijski port sa ugrađenim UART-om i punim dupleksnim radom. Mikrokontroler prima serijski ulazni signal preko priključka *RxD* a serijski izlazni signal je na priključku *TxD*, videti sledeću sliku.



Podatak koji se primi preko serijskog porta smešta se u ulazni registar *SBUF* iz koga može da se pročitati i prenese u neki drugi registar. Podatak koji treba poslati preko serijskog porta takođe se upisuje u registar *SBUF*. Međutim, radi se o dva različita registra, ali pošto se kod serijskog izlaza podatak uvek upisuje, a kod prijema uvek čita, za ova dva registra koristi se isti naziv i adresa.

Podatak koji je primljen preko serijskog ulaza prenosi se u registar *A* sledećom instrukcijom:

```
MOV A, SBUF
```

Na sličan način, podatak koji je u registru *A*, sledeća instrukcija upisuje u registar *SBUF* odakle se prenosi preko serijskog izlaznog signala:

```
MOV SBUF, A
```

Izbor vrste rada serijskog porta obavlja se programiranjem upravljačkog registra *SCON* čiji format je prikazan na sledećoj slici.

|            |            |            |            |            |            |           |           |
|------------|------------|------------|------------|------------|------------|-----------|-----------|
| 7          | 6          | 5          | 4          | 3          | 2          | 1         | 0         |
| <i>SM0</i> | <i>SM1</i> | <i>SM2</i> | <i>REN</i> | <i>TB8</i> | <i>RB8</i> | <i>TI</i> | <i>RI</i> |

Ukratko ćemo opisati samo one bite koji su od interesa za dalje izlaganje. Upis u bite *SM0* i *SM1* određuje jednu od četiri vrste rada serijskog porta:

| <i>SM0</i> | <i>SM1</i> | vrsta rada | format poruke      |
|------------|------------|------------|--------------------|
| 0          | 0          | 0          | pomerački registar |
| 0          | 1          | 1          | podatak od 8 bita  |
| 1          | 0          | 2          | podatak od 9 bita  |
| 1          | 1          | 3          | podatak od 9 bita  |

Upis logičke 1 u bit *REN* dozvoljava serijski prijem, a logička 0 sprečava serijski prijem podataka. Upis logičke 1 u bite *TI* i *RI* dozvolja generisanje prekida kod predaje (*TI*) i prijema (*RI*) podataka. Ako je prekid kod predaje dozvoljen (*TI* = 1), onda serijski port generiše prekid kada se pošalju svi biti serijskog podatka. Slično, ako je prekid kod prijema dozvoljen (*RI* = 1), serijski port generiše prekid kada se prime svi biti serijskog ulaznog podatka.

U daljem tekstu pokazaće se primeri rada sa vrstom rada 1, u kojoj reč ima 10 bita: jedan početni bit, 8 bita podatka i jedan završni bit. Upisom u registar *SBUF* automatski pokreće serijskih izlaz podataka. Prijem serijskih podataka i njihov upis u ulazni registar *SBUF* počinje upisom logičke 1 u bit *REN* registra *SCON*.

Brzina prenosa (*baud rate*) u vrsti rada 1 može da se podešava primenom tajmera 1. Tajmer 1 treba da se postavi u vrstu rada 2 u kojoj 8-bitni registar *TL* odbrojava nadole i kada odbroji do nule, onda se početna vrednost brojanja koja je prethodno postavljena u 8-bitni registar *TH* ponovo upisuje u registar *TL* i ponavlja brojanje. Prenos sa najznačajnijeg bita *TL* koristi se kao osnova za određivanje brzine prenosa.

Ako je  $f$  frekvencija sinhronizacionog signala mikrokontrolera, brzina prenosa  $p$  određuje se prema sledećoj formuli:

$$p = \frac{f \times 2^{SMOD}}{32 \times 12 \times (256 - TH)}$$

*SMOD* je bit 7 registra *PCON*. U daljem tekstu, ako nije drugačije navedeno, smatraćemo da je *SMOD* = 0. Na primer, za  $f = 6 \text{ MHz}$  i zahtevanu brzinu prenosa od 110 bita u sekundi, potrebno je postaviti  $TH = 72H$ .

### Konvertor napona

Kao što je već rečeno električne karakteristike signala u mikroračunarskom sistemu drugačije su od električnih signala definisanih standardom RS232. Signali mikroračunara, na primer signali mikrokontrolera 8051, 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.

Konverzija napona sa TTL ulaznih signal na RS 232 kompatibilne izlazne signale treba da se obavi u skladu sa sledećom specifikacijom:

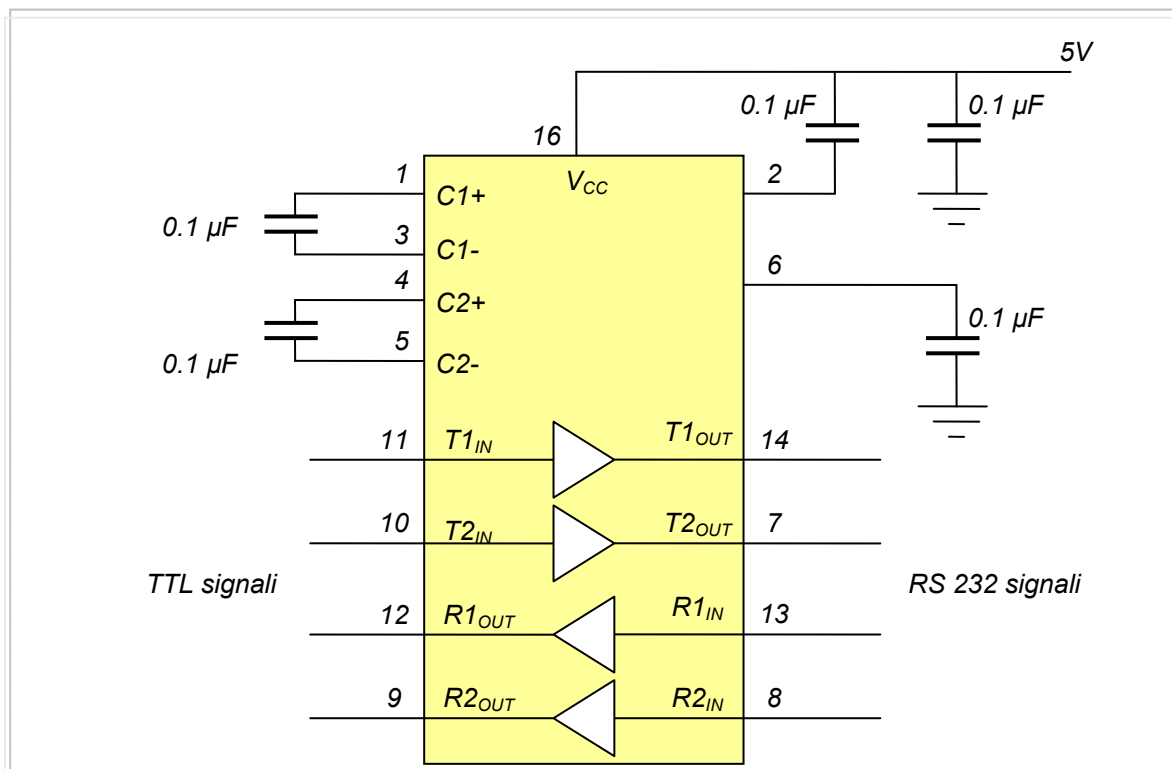
| logički nivo | TTL napon [V] |     | RS 232 napon [V] |     |
|--------------|---------------|-----|------------------|-----|
|              | min           | max | min              | max |
| logička 0    | 0             | 0.8 | 3                | 25  |
| logička 1    | 2.4           | 5.0 | -3               | -25 |

Konverzija napona sa RS 232 ulaznih signal na TTL kompatibilne izlazne signale treba da se obavi u skladu sa sledećom specifikacijom:

| logički nivo | RS 232 napon [V] |     | TTL napon [V] |     |
|--------------|------------------|-----|---------------|-----|
|              | min              | max | min           | max |
| logička 0    | 3                | 25  | 0.0           | 0.4 |
| logička 1    | -3               | -25 | 2.8           | 5.0 |

Kao primer konvertora napona uzećemo kolo MAXIM MAX232E, čiji blok dijagram je prikazan na sledećoj slici. Ovo kolo ima dva konvertora *T1* i *T2* koji digitalne signale ulazne signale  $T1_{IN}$  i  $T2_{IN}$  kompatibilne sa TTL standardom, prevode u digitalne izlazne signale  $T1_{OUT}$  i  $T2_{OUT}$  koji su kompatibilni sa RS 232 signalima. Druga dva konvertora *R1* i *R2*, digitalne ulazne signale  $R1_{IN}$  i

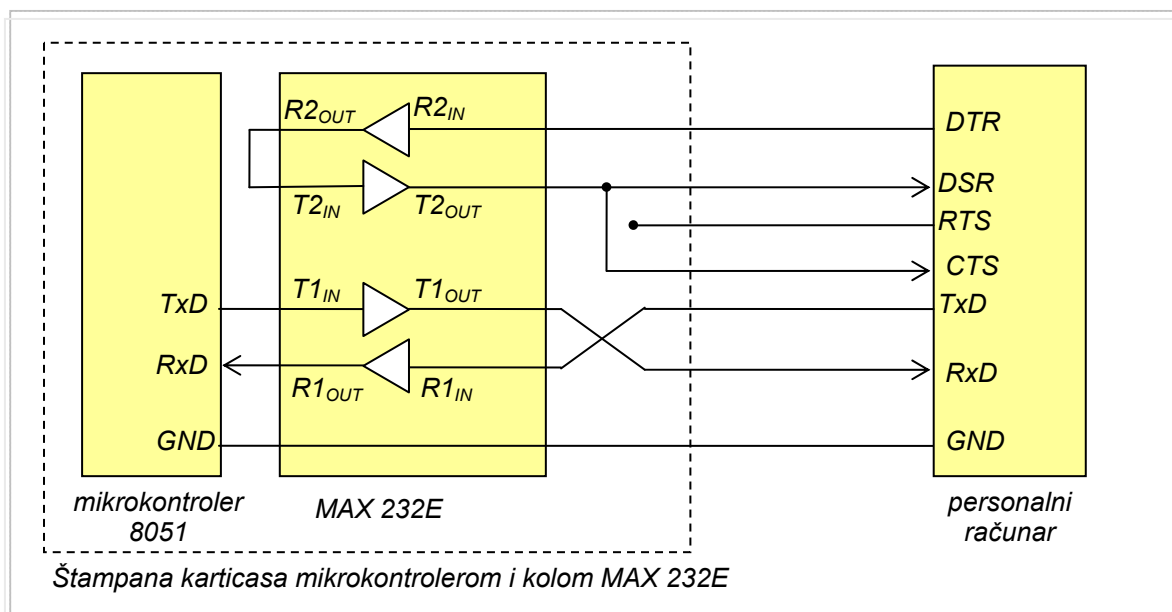
$R2_{IN}$  kompatibilne sa RS232 standardom, prevode u digitalne izlazne signale  $R1_{OUT}$  i  $R2_{OUT}$  koji su kompatibilni sa TTL signalima.



Zanimljivo je da kolo MAX232E ima samo jedno napajanje, +5 V. Interne naponske pumpe za čiji rad su potrebni kondenzatori prikazani na slici, napon od +5 V konvertuju u napone +10 V i -10 V koji su potrebni za izlazne signale koji su kompatibilni sa RS 232 standardom. Na slici su prikazani brojevi spoljnih priključaka kola MAX232E i nazivi spoljnih signala.

### **Priključenje mikrokontrolera 8051 na serijski port personalnog računara**

Primenom veze bez modema mikrokontroler 8051 može na jednostavan način neposredno da se priključi na serijski port personalnog računara, bez primene signala za rukovanje, kao što pokazuje sledeći dijagram.



Izlazni signal  $TxD$  mikrokontrolera dovodi se na ulaz  $T1_{IN}$  i prevodi u serijski izlazni signal  $T1_{OUT}$  koji je kompatibilan sa RS 232 standardom. Na sličan način ulazni RS 232 serijski signal dovodi se na ulaz  $R1_{IN}$  i prevodi u serijski signal  $R1_{OUT}$  koji je kompatibilan sa TTL standardom i dovodi na ulaz  $RxD$  mikrokontrolera. Na štampanoj kartici, koja je prikazana isprekidanom linijom na slici gore, na kojoj se nalaze mikrokontroler i kolo MAX 232 E treba ukrstiti serijske signale, tako da se serijski izlaz mikrokontrolera preko konvertora T1 dovodi na serijski ulaz personalnog računara i serijski izlaz personalnog računara preko konvertora R1 dovodi na serijski ulaz mikrokontrolera.

Kao što je već rečeno, mikrokontroler nema signale za rukovanje pa treba primeniti jednu od varijanti veze bez modema. Na gornjoj slici je signal  $DTR$  sa personalnog računara, preko konvertora R2 i T2 vraćen na ulaze  $DSR$  i  $CTS$  personalnog računara. Naravno da može da se primeni i drugo rešenje, na primer da se na štampanoj kartici signal  $RTS$  personalnog računara direktno vrati kao signal  $CTS$  personalnog računara, kao što je to već ranije pokazano.

U primeru sa gornje slike podrazumeva se da su personalni računar i mikrokontroler uvek spremni za serijski prijem podataka. Ukoliko nisu spremni, veza neće biti ostvarena i neće biti upozorenja korisniku. Međutim, pošto su u ovim primenama obično mikrokontroler i personalni računar na istom mestu, korisnik može odmah primetiti da serijska veza ne radi i otkloniti uzrok problema.

### Inicijalizacija serijske veze mikrokontrolera 8051 i personalnog računara

U daljem tekstu ćemo pokazati primere malih programa u assembleru za podršku serijskoj vezi mikrokontrolera i personalnog računara, koja je prikazana na slici u prethodnom odeljku. Neka je frekvencija sinhronizacionog signala mikrokontrolera jednaka 12 MHz i želimo sledeće parametre serijskog kanala:

- brzina prenosa  $p = 2400$  bita u sekundi,
- jedan start bit, 8 bita za podatak, jedan stop bit, vrsta rada serijskog porta 1,
- zabrana prekida kod prijema i predaje,
- dozvola serijskog prijema i predaje.

Iz formule:

$$p = \frac{f}{32 \times 12 \times (256 - TH)}$$

dobijamo:

$$TH = 256 - \frac{f}{384 \times p} = 243_{10} = F3_{16}$$

Pošto ćemo koristiti tajmer *TI* za određivanje brzine serijskog prenosa, podsetimo se registra *TMOD*, čiji biti su prikazani na sledećoj slici.

| tajmer <i>T1</i> |     |    |    |  | tajmer <i>T0</i> |     |    |    |
|------------------|-----|----|----|--|------------------|-----|----|----|
| 7                | 6   | 5  | 4  |  | 3                | 2   | 1  | 0  |
| G                | C/T | M1 | M0 |  | G                | C/T | M1 | M0 |

Pojedini biti za tajmer *T1* imaju sledeća značenja:

- *G* (*gate*), ako je 0, onda *INT1* ulaz određuje da li tajmer *T1* broji, ako je jednak 1, onda bit *TR1* u registru *TCON* određuje da li tajmer *T1* broji.
- *C/T*, određuje tajmerski (1) ili brojački (0) način rada.
- *M1* i *M2* određuju vrstu rada tajmera *T1*, za *M1* = 1 i *M0* = 0, bira se vrsta rada 2 tajmera *T1* u kojoj 8-bitni registar *TL1* odbrojava nadole i kada odbroji do nule, onda se početna vrednost brojanja koja je prethodno postavljena u 8-bitni registar *TH1* ponovo upisuje u registar *TL1* i ponavlja brojanje. Prenos sa najznačajnijeg bita *TL1* koristi se kao osnova za određivanje brzine prenosa.

Prema tome, ako se tajmer *T1* koristi za određivanje brzine rada serijskog prenosa, onda gornja 4 bita registra *TMOD* treba da budu 0010<sub>2</sub> (ili 20H).

Dozvola rada tajmera *T1* dobija se upisom 1 u bit 6 (*TR1*) registra *TCON*. Bit *SMOD* (bit broj 7) registra *PCON* treba da je 0.

Biti registra *SCON* treba da imaju sledeće vrednosti:

- *SM0* = 0, *SM1* = 1, izbor načina rada 1 serijskog porta u kome se prenosi 8 bita podatka.
- *SM2* = 0, nema prekida ako se ne primi ispravan stop bit.
- *REN* = 1, dozvola serijskog prijema.
- Biti *TB8* i *RB8* koriste se kod prenosa podataka od 9 bita i ovde mogu da se stave na 0.
- Biti *TI* i *RI* predstavljaju signale prekida kod predaje (*TI*) i prijema (*RI*) i mogu se postaviti na 0 kod inicijalizacije.

Prema tome, bite registra *SCON* treba kod inicijalizacije postaviti na vrednosti 01010000 odnosno sadržaj registra *SCON* treba da bude *SCON* = 50H.

Sa navedenim vrednostima registara, inicijalizacija serijskog porta sa zahtevanim parametrima može se dobiti izvršenjem sledećeg potprograma.

```

sp_init:    CLR    IE.4           ; zabrana serijskog prekida
            ANL    TMOD, #0FH      ; brisanje gornja 4 bita TMOD
            ORL    TMOD, #20H      ; TMOD = 0010xxxx
            MOV    TL1, #F3H       ; pocetna vrednost brojaca
            MOV    TH1, #F3H
            ANL    PCON, #7FH      ; SCON = 0

```

```

SETB  TR1                ; dozvola rada tajmera T1
MOV    SCON, #50H        ; vrsta rada 1 serijskog porta
RET                                ; povratak u glavni program

```

U daljem tekstu smatraćemo da je serijski port inicijalizovan gornjim potprogramom.

### **Serijski ulaz/izlaz primenom programskog upravljanja**

U ovom odeljku pokazaćemo jednostavne programe za serijski ulaz/izlaz podataka primenom programskog upravljanja. Potprogram za serijski ulaz jedne reči ima sledeći oblik.

```

get_char:  JNB    SCON.0, get_char  ; proverava da li je primljena rec
           MOV    A, SBUF           ; smestanje primljene reci u A
           RET

```

Prva instrukcija u potprogramu proverava da li je primljena reč preko serijskog ulaza, tako što proverava da li je bit 0 u registru *SCON* postavljen na 1. Ukoliko nije, program se vrti u petlji iz koje izlazi kada je reč primljena i ona se stavlja u registar *A*.

Nedostatak programskog upravljanja je u tome što potprogram za ulaz jedne reči stoji u petlji sve dok port nije primio jednu reč preko serijskog ulaza. Na primer, ukoliko se desila neka greška i nema serijskog ulaznog signala, potprogram će ostati da se vrti u petlji bez ograničenja vremena čekanja. Takođe, dok potprogram čeka na ulaz reči, mikrokontroler gubi vreme jer ne može ništa drugo da radi.

Sličan oblik ima i potprogram za serijski izlaz jedne reči koja je u glavno programu pre toga smeštena u registar *A*.

```

put_char:  JNB    SCON.1, put_char  ; da li je prethodna rec poslata?
           MOV    SBUF, A           ; smestanje reci u izlazni registar
           RET

```

Prva instrukcija proverava da li izlazni deo serijskog porta spreman za slanje podataka (bit 1 registra *SCON* je postavljen na 1), odnosno ako se prenosi niz reči, da li je prethodna reč prenesena i serijski port spreman da prihvati i prenese sledeću reč. Ako je spreman, onda se reč prenosi iz registra *A* u izlazni serijski registar *SBUF* i potprogram se završava.

Slično kao kod ulaza, nedostatak programskog upravljanja je u tome što potprogram za izlaz jedne reči stoji u petlji sve dok port nije spreman za serijski izlaz. Opet, ukoliko se desila greška i port ne može da šalje serijske podatke, potprogram ne može da se završi. Isto tako, dok potprogram čeka na izlaz, mikrokontroler ne može da obavlja neke druge korisne operacije.

Prethodne potprograme možemo koristiti za ulaz i izlaz niza reči. Pretpostavimo da je u memoriji mikrokontrolera smešten niz ASCII znakova koje treba preneti i u kome je poslednja reč 00H (*null* reč). Sledeći potprogram prenosi šalje reč po reč serijskom portu koristeći potprogram *put\_char*. Pretpostavićemo da registar *DPTR* pokazuje na prvu reč u nizu.

```

put_string: MOVX   A, @DPTR          ; sledeca rec stavlja se u A
           JZ     kraj              ; da li je kraj?
           LCALL  put_char          ; slanje reci iz A
           INC    DPTR              ; inkrement pokazivaca na niz
           AJMP   put_string        ; sledeca rec
kraj:      RET

```



Potprogram uzima jednu po jednu reč iz niza, proverava da li je poslednja reč (00) i ako nije poziva potprogram *put\_char* koji prenosi reč u izlazni registar serijskog porta. Ukoliko je poslednja reč, potprogram se završava.

Sledeći potprogram obavlja ulaz ASCII znakova preko serijskog porta i smešta ih u memorijsku zonu na koju pokazuje registar *DPTR*. Poslednja znak u nizu je *CR* (*Carriage Return*) čija je heksadecimalna vrednost *0DH*.

```
CR          EQU    0DH                ; CR je poslednji znak u nizu

get_string: LCALL  get_char            ; ulazni znak smesten je u A
            CJNE   A, #CR, dalje       ; da li je poslednji znak?
            MOVX   @DPTR, A           ; poslednji znak stavlja se u niz
            RET

dalje:      MOVX   @DPTR, A           ; ulazni znak stavlja se u niz
            INC    DPTR               ; inkrement pokazivaca na niz
            AJMP   get_char           ; potprogram nastavlja dalje
```

Direktiva *EQU* omogućava korišćenje imena *CR* za heksadecimalnu vrednost istoimenog ASCII znaka. Potprogram prvo poziva *get\_char* koji primljeni ulazni znak smešta u *A*. Ako primljeni znak nije *CR* potprogram ga smešta u memoriju i nastavlja rad. Ako je primljeni znak *CR*, on se smešta u memoriju i potprogram se završava.

### **Serijski ulaz/izlaz primenom prekida**

Primena prekida otklanja petlju u kojoj mikrokontroler čeka na prijem reči kod serijskog ulaza, odnosno na spremnost serijskog porta za serijski izlaz. Suština primene prekida je da mikrokontroler može da radi neki koristan posao dok serijski port nije spreman za prenos. Kada je serijski port spreman za prenos, on generiše prekid, mikrokontroler prekida tekući program i u programu za obradu prekida pošalje ili primi reč preko serijskog porta.

Primena prekida za upravljanje serijskim ulazom i izlazom zahteva brižljivo planiranje programa koji je po pravilu složeniji od primena programskog upravljanja. Ovde ćemo pokazati osnovne programe za serijski ulaz i izlaz primenom prekida, u kojima se mikrokontroler vrti u petlji dok čeka na prekid sa serijskog porta. Umesto beskrajne petlje programer može da koristi mikrokontroler da izvršava neke korisne operacije.

Program za obradu prekida sa serijskog porta ima sledeći oblik. Podrazumeva se da je reč koju treba preneti smeštena u registar *A*, a reč koja se primi smesti se u registar *A*.

```
                ORG    23H                ; ulazna tačka za prekid sa serijskog
                                ; porta
                LJMP   ser_UI            ; skok na obradu prekida
                ...

ser_UI:         JNB    SCON.1, ulaz      ; da li je serijski izlaz spreman?
                MOV    SBUF, A          ; ako jeste, upis reci u izlazni reg

ulaz:          JNB    SCON.0, kraj      ; da li je primljena rec?
                MOV    A, SBUF          ; ulazna reč smešta se u A

kraj:          RETI                    ; kraj programa za obradu prekida
```

Prva instrukcija koja se izvršava kada se dogodi prekid sa serijskog porta nalazi se na adresi *23H*. Pošto nema dovoljno mesta za celi program za obradu prekida, na adresu *23H* stavimo instrukciju programskog skoka na prvu instrukciju programa za obradu prekida. Ovaj program prvo proverava da li je serijski port spreman za serijski izlaz (da li je bit 1 registra *SCON* na logičkoj 1). Ako jeste,

reč iz *A* smešta se u izlazni registar *SBUF*. Ako nije spreman ili kada se upiše izlazna reč, program proverava da li je primljena reč preko serijskog ulaza. Ako jeste, reč se smešta u *A* i završava program za obradu prekida. Treba primetiti da ovaj program podržava dupleks prenos, dakle prima i šalje po jednu reč.

Prethodni program može da se proširi tako da podržava prenos niza ASCII znakova. U ovom slučaju potrebna su nam dva pokazivača, jedan na memorijski bafer u koji smeštamo ulazne znakove (neka je to registar *DPTR*) i drugi iz koga čitamo izlazne znakove (neka je to registarski par *R6* i *R7*). U sledećem programu za obradu prekida, kada se primi i/ili pošalje znak, inkrementira se registar *DPTR* i/ili registarski par *R6, R7*, respektivno.

```

                ORG    23H                ; ulazna tačka za prekid sa serijskog
                ; porta
                LJMP   ser_UI            ; skok na obradu prekida
                ...

ser_UI:         JNB    SCON.1, ulaz       ; da li je serijski izlaz spreman?
                MOV    SBUF, A           ; ako jeste, upis reci u izlazni reg
                MOV    A, R7             ; inkrement pokazivača R6,R7
                ADD    A, #1
                MOV    R7, A
                MOV    A, R6
                ADDC   A, #1
                MOV    R6, A

ulaz:          JNB    SCON.0, kraj       ; da li je primljena rec?
                MOV    A, SBUF           ; ulazna reč smešta se u A
                MOVX   @DPTR, A          ; ulazna rec u ulazni bafer
                INC    DPTR              ; inkrement pokazivaca na bafer

kraj:          RETI                     ; kraj programa za obradu prekida

```

Program podrazumeva da je izlazna reč već smeštena u registar *A*. Ako je izlazni registar prazan, izlazna reč smešta se u registar *SBUF* i zatim inkrementira 16-bitni pokazivač koji se sastoji od registara *R6* (gornjih 8 bita pokazivača) i *R7* (donjih 8 bita pokazivača). Ako je primljena ulazna reč, ona se smešta u bafer na koji pokazuje registar *DPTR* koji se zatim inkrementira.

Sledeći potprogram obavlja ulaz niza ASCII znakova sa serijskog porta, pri čemu je poslednji znak *CR*. Adresa ulaznog bafera je u registru *DPTR*.

```

CR              EQU    0DH                ; CR je poslednji znak u nizu

get_st_int:     ORL    IE, #90H           ; dozvola prekida sa serijskog porta
                CLR    A                  ; pocetni sadrzaj A je 00

petlja:         CJNE   A, #CR, petlja     ; cekanje na poslednji znak (CR)
                RET

```

Potprogram prvo dozvoli prekid i zatim se vrti u petlji u kojoj proverava da li je sadržaj registra *A* jednak *CR*. U toku izvršavanja petlje, povremeno se događa prekid sa serijskog porta i aktivira program za obradu prekida. Ovaj program smešta ulazni znak u ulazni bafer, inkrementira pokazivač i završava. Preme tome, sadržaj registra *A* u petlji potprograma biće jednak primljenom znaku i kada je taj znak *CR*, potprogram se završava.

Potprogram za serijski izlaz ASCII znakova malo je složeniji, jer je pokazivač sada u registarskom paru *R6* i *R7*.

```

put_st_int:     PUSH   DPH                ; čuvanje DPTR na steku
                PUSH   DPL

```

```

                                ORL    IE, #90H                ; dozvola prekida sa serijskog porta
petlja:                        MOV     DPH, R6                ; smeštanje pokazivača u DPTR
                                MOV     DPL, R7
                                MOVX    A, @DPTR              ; sledeca rec stavlja se u A
                                JNZ     petlja                ; da li je kraj?

                                POP     DPL                    ; ako je kraj, vraćanje sadržaja DPTR
                                POP     DPH
                                RET

```

U potprogramu se registar *DPTR* koristi kao privremeni pokazivač na izlazni bafer i zato se *DPTR* prvo sačuva na steku i zatim dozvoljava prekid sa serijskog porta. U petlji se pokazivač na izlazni bafer iz registara *R6* i *R7* kopira u *DPTR*, zatim znak iz izlaznog niza na koji pokazuje *DPTR* smešta u *A* i proverava da li je taj znak *00H*. U toku izvršavanja ove petlje događaju se prekidi, svaki put u programu za obradu prekida sadržaj registra *A* smešta se u serijski izlazni registar *SBUF* i inkrementira pokazivač *R6* i *R7*. Petlja se izvršava sve dok se ne primi *00H*, kada se završava i sadržaj registra *DPTR* vraća sa steka.