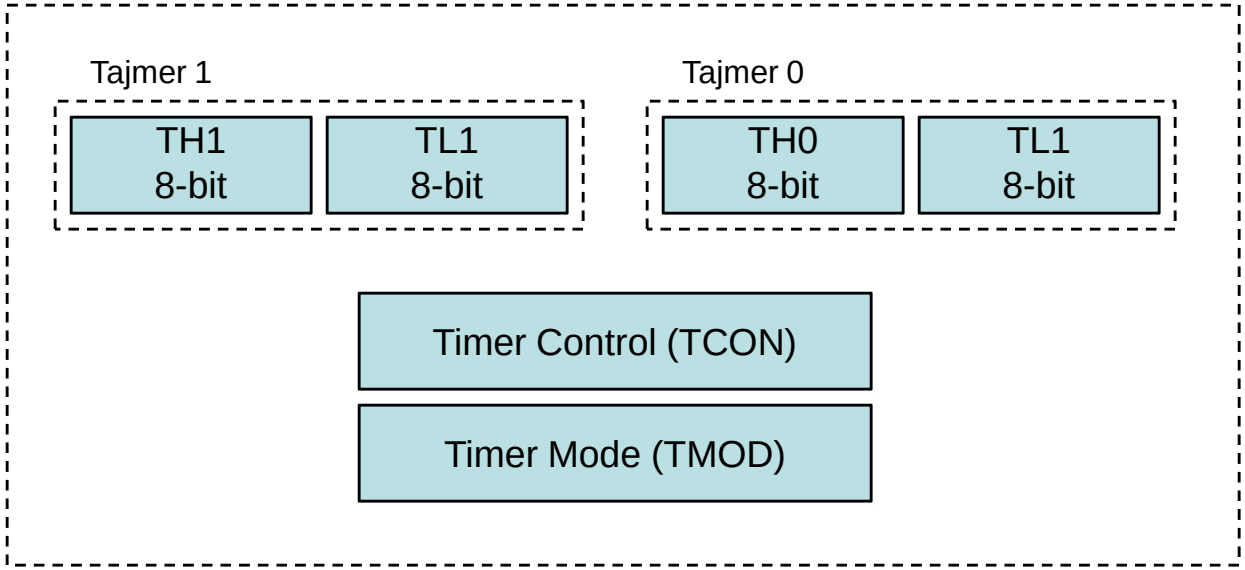
Mikrokontroleri

Mikrokontroleri

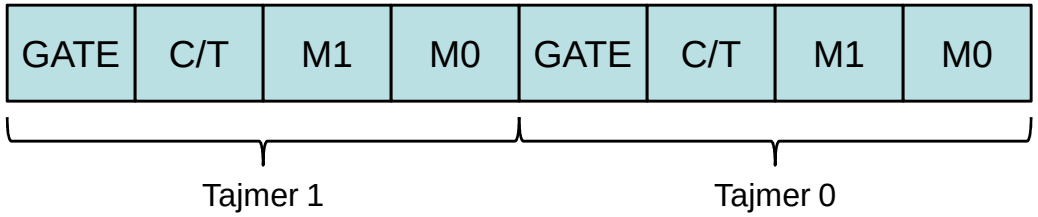
Poglavlje II

Arhitektura mikrokontrolera familije MCS-51

Tajmeri/Brojači

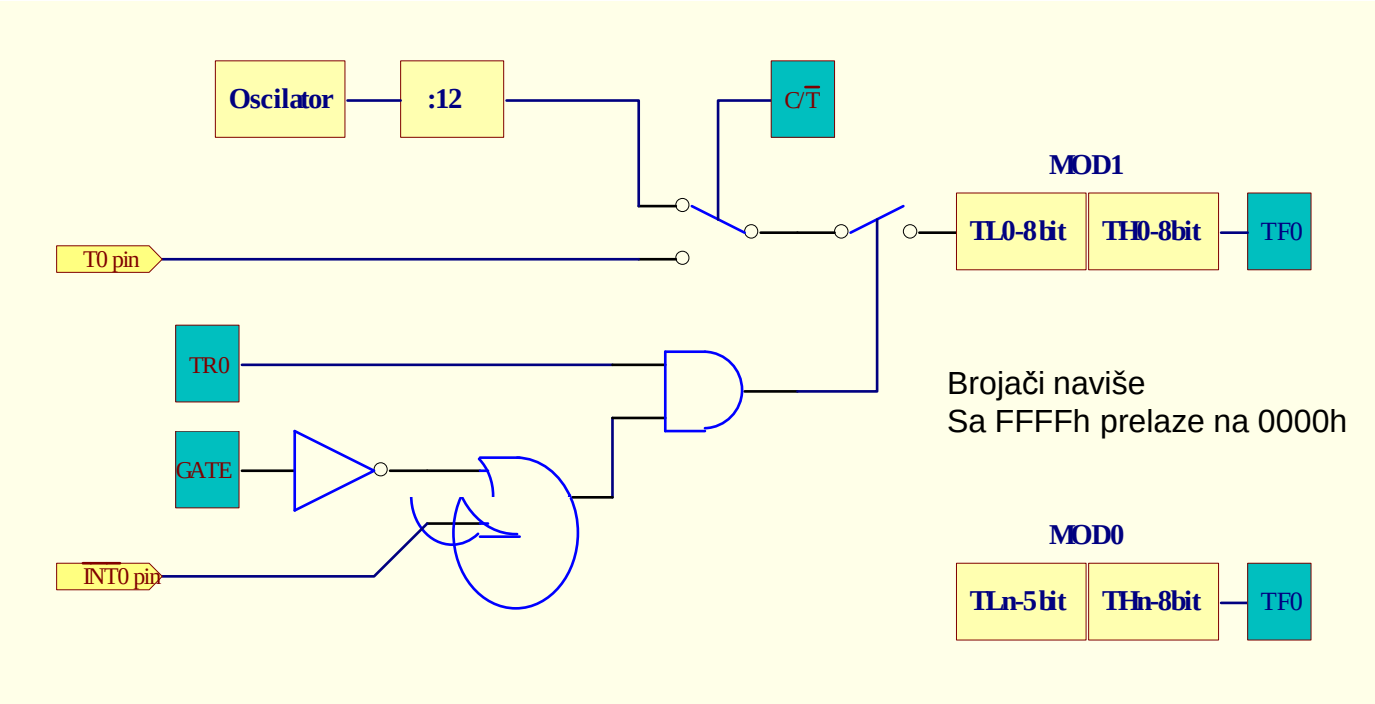


TMOD



M1	M0		
0	0	Mode 0	13 bitni brojač naviše
0	1	Mode 1	16 bitni brojač naviše
1	0	Mode 2	8 bitni brojač sa autoreload funkcijom
1	1	Mode 3	Kombinacija oba daje tri 8 bitna

Tajmeri 0 (1)



TMOD

GATE	C/T	M1	M0	GATE	C/T	M1	M0

TCON

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

Kontrola prekidnog sistema

Primer:

Generisati simetričan talasni oblik frekvencije 1 kHz na liniji P1.7. Taktna frekvencija mikrokontrolera iznosi 11.0592 MHz

- Izbor načina rada brojača: kao negejtovani tajmer u režimu “mode 0”

  TMOD = xxxx0001b

GATE	C/T	M1	M0	GATE	C/T	M1	M0

- Određivanje taktne frekvencija brojaca/tajmera

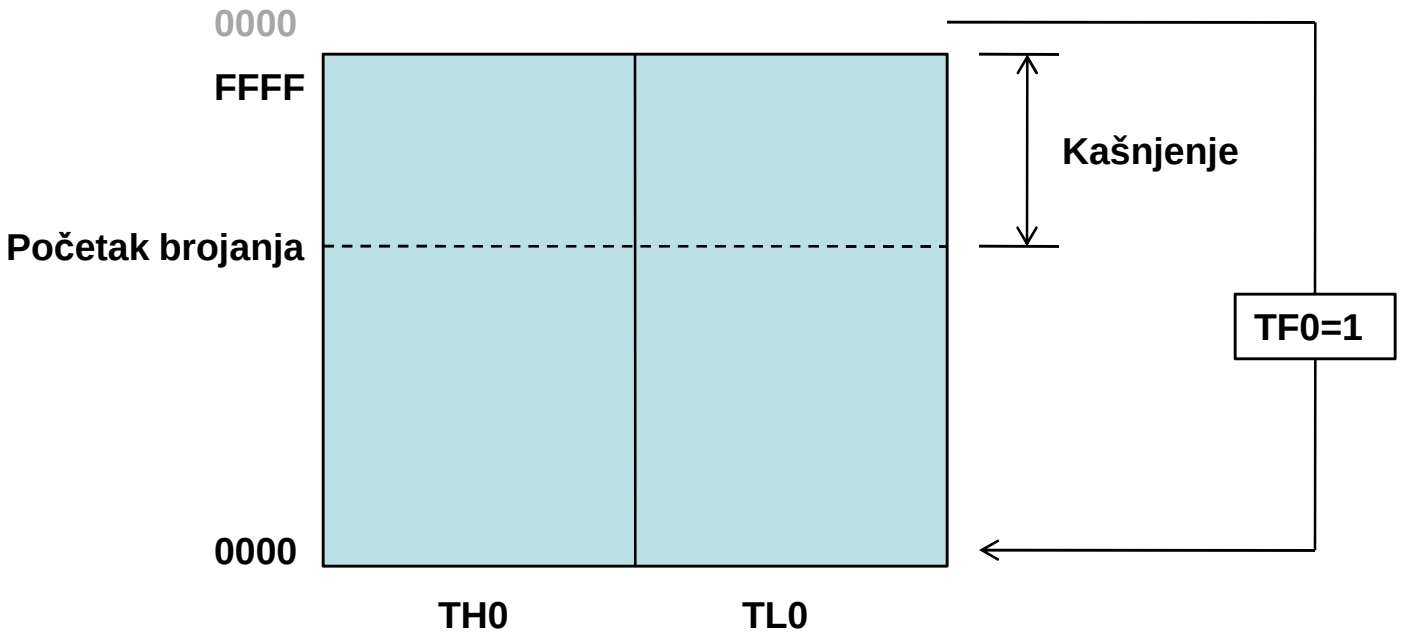
  Fclk = 11.0592/12 MHz

  Fclk = 921.6 kHz

  Tclk = 1081.5487 ns

- Određivanje početne vrednosti brojača

Određivanje početne vrednsti brojača



Početak brojanja = 65536 – Kašnjenje
Kašnjenje = 500/1.0815487 = 462

Početak brojanja = 65074
Početak brojanja = FE32h

- Izrada procedure vremenskog kašnjenja

```
KASNI: MOV TH0,#0xFE ; postavljanje pocetne ..
       MOV TL0,#0x32 ; .. vrednosti brojaca
       SETB TR0      ; ukljuci brojac

FLAG:  JNB TF0,FLAG  ; cekaj dok se TF0 ne postavi

       CLR TR0       ; iskljuci brojac
       CLR TF0       ; resetuj TF0 na nulu
       RET           ; izlaz iz podprograma


;--------------------------------
kasnjenje equ 462
       MOV TH0,#high(-kasnjenje)
       MOV TL0,#low(-kasnjenje)
;--------------------------------
```

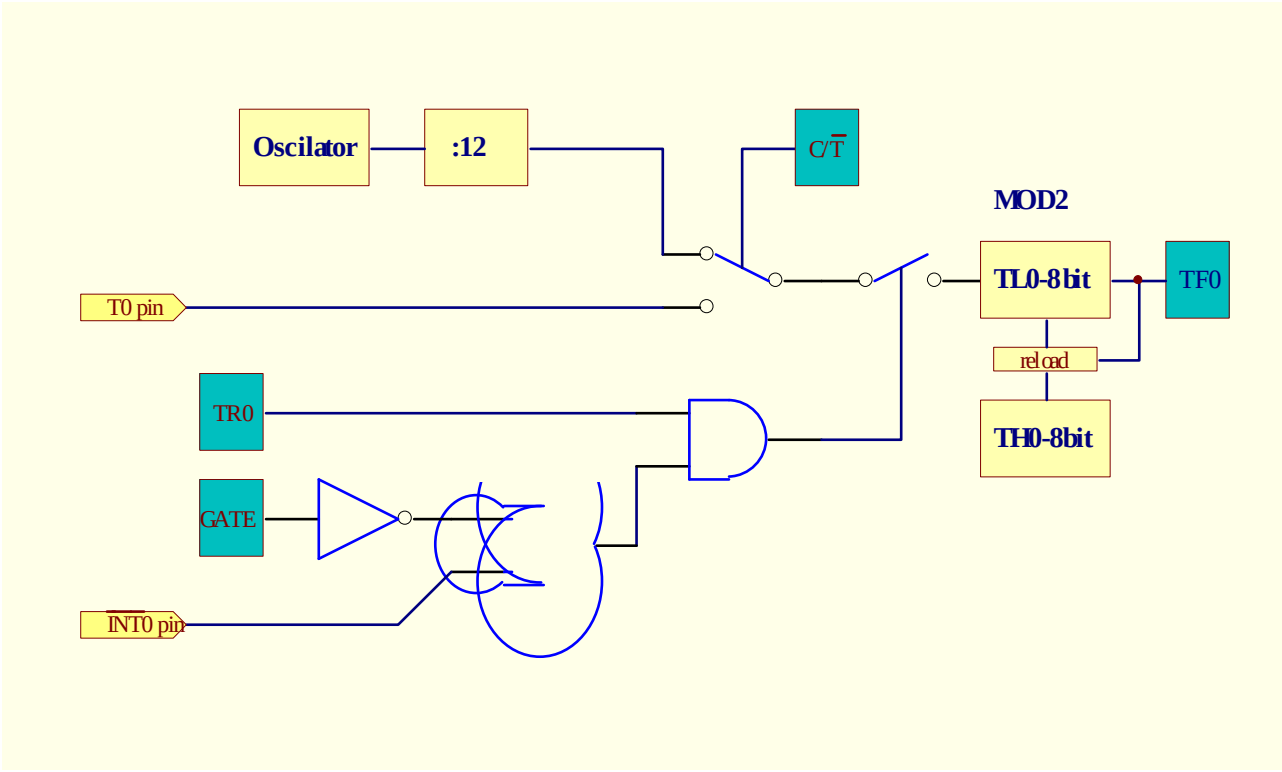
```
;Program na asembleru

        ORG 0           ; reset adresa
        SJMP START      ; rezervisana oblast

        ORG 40H         ; start adresa na 0040H
START:  MOV TMOD,#01H   ; put Timer 0 into mode 1
PETLJA: SETB P1.7       ; logic 1 (5 volts)
        ACALL KASNI     ; 0.5ms kasnjenje
        CLR P1.7        ; logic 0 (0 volts)
        ACALL KASNI     ; 0.5ms kasnjenje
        SJMP PETLJA     ; repeat

;---------------------------------------
PETLJA: CPL P1.7
        ACALL KASNI     ; 0.5ms kasnjenje
        SJMP PETLJA     ; repeat
;---------------------------------------
```

Greške u generisanju vremenskog intervala

```
;Program na C jeziku
#include <reg51.h>
#define on 1
#define off 0
sbit Izlaz = P1^7;
void kasni();  // deklaracija funkcije koja ne vraca parametre

main() {                // pocetak programa
       TMOD=0x01;       // nacin rada tajmera
       while(1) {       // beskonacna petlja
         Izlaz = on;    // P1.7 = 1
         kasni();       // kasnjenje
         Izlaz = off;   // P1.7 = 0
         kasni();       // kasnjenje
       }
}

void kasni() {
       TH0 = -(462/256);
       TL0 = -(462%256);
       TR0 = on;
       while(!TF0);
       TR0 = off;       // stop the timer0
       TF0 = off;       // clear flag TF0
}
```

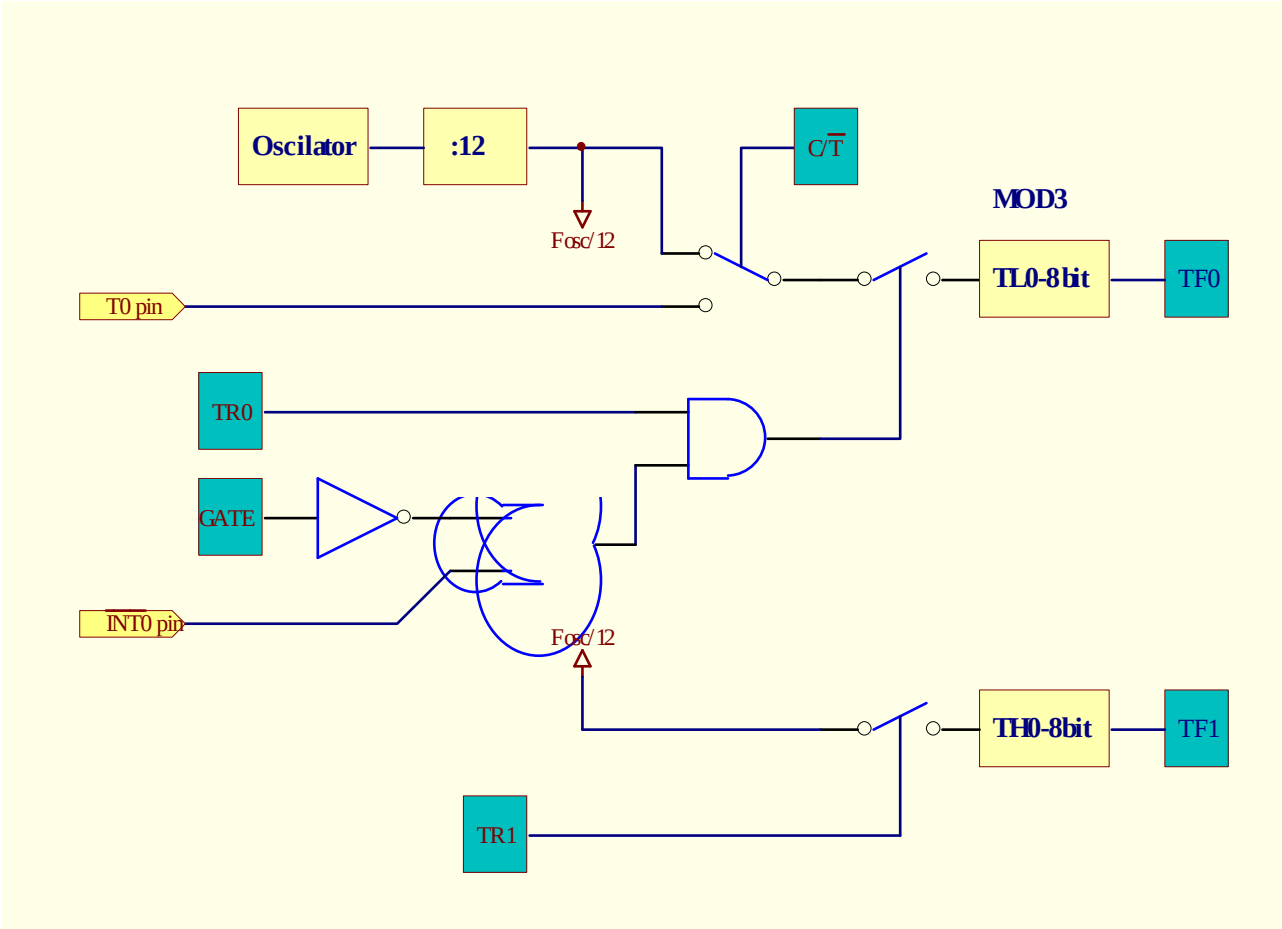
!!!

Tajmeri 0 (1)



Automatsko reloadovanje brojača
Maksimalni moduo deljenja = 256 (za TH0=0)
Tajmer 1 kao “baud rate” generator

Tajmer 0 u “mode 3”

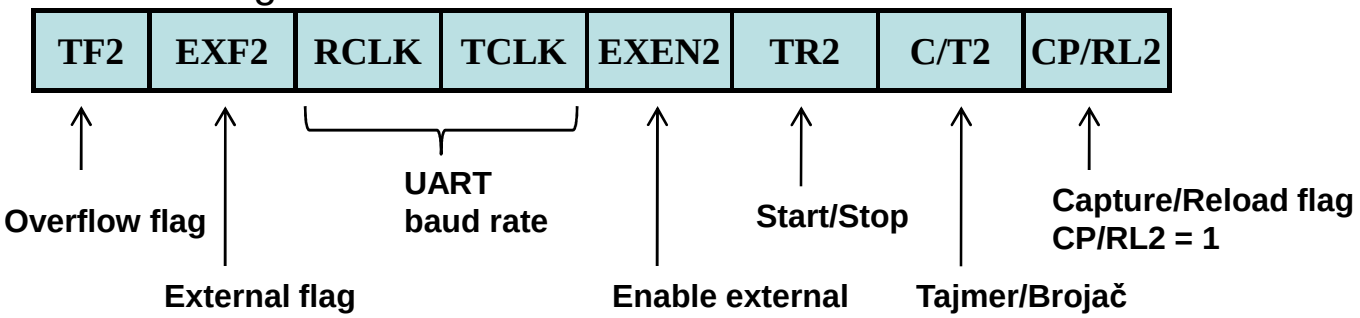


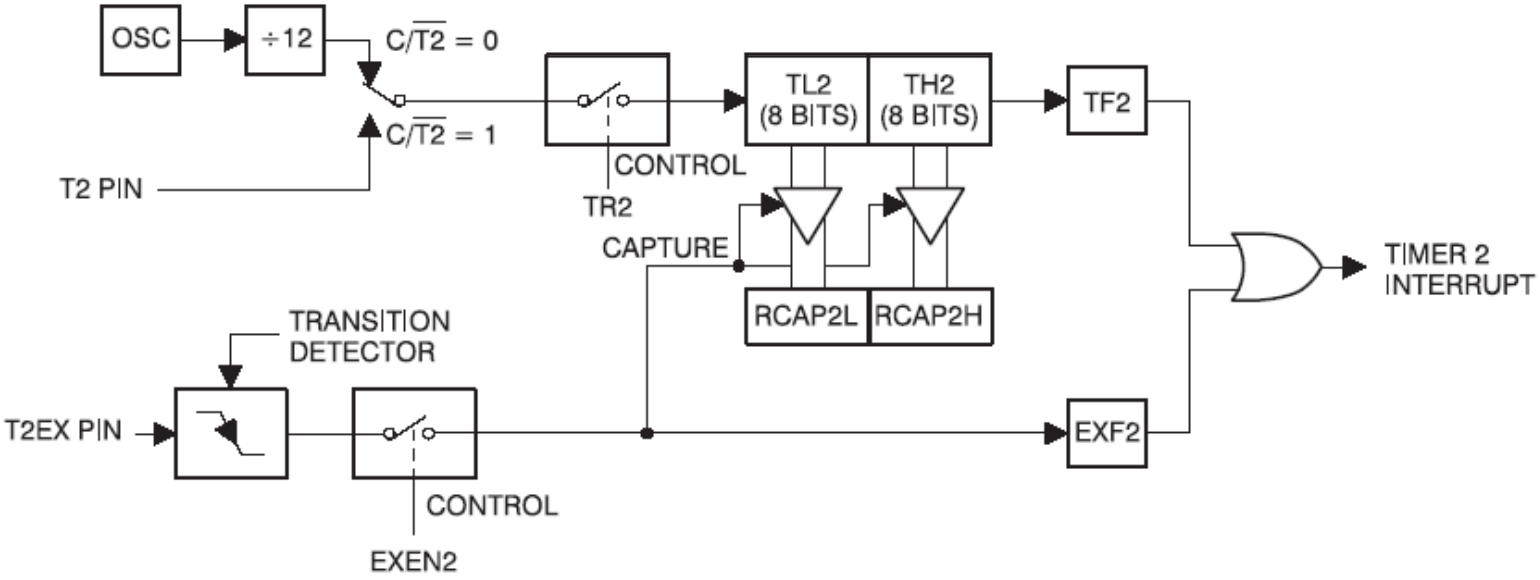
Korisno kada je Tajmer 1 *baud rate* generator

Unapređene varijante sa velikim brojem tajmera!

Tajmer 2 (8052)

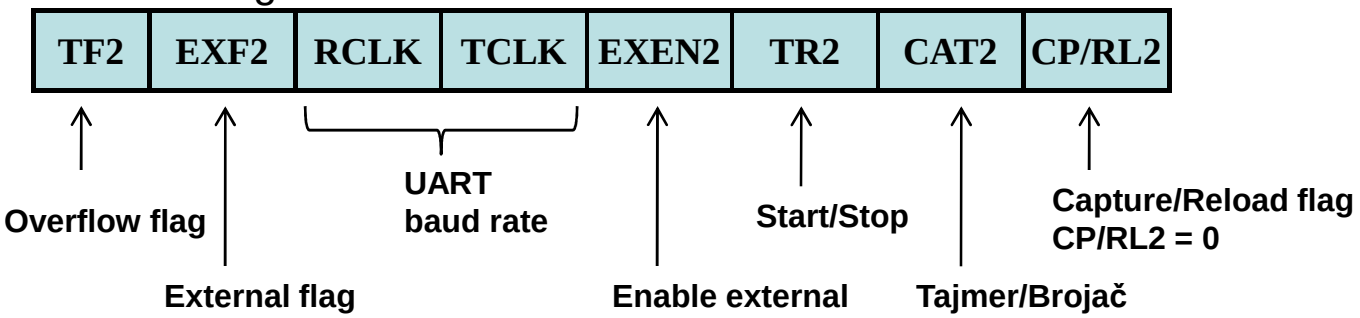
Kontrolni registar T2CON

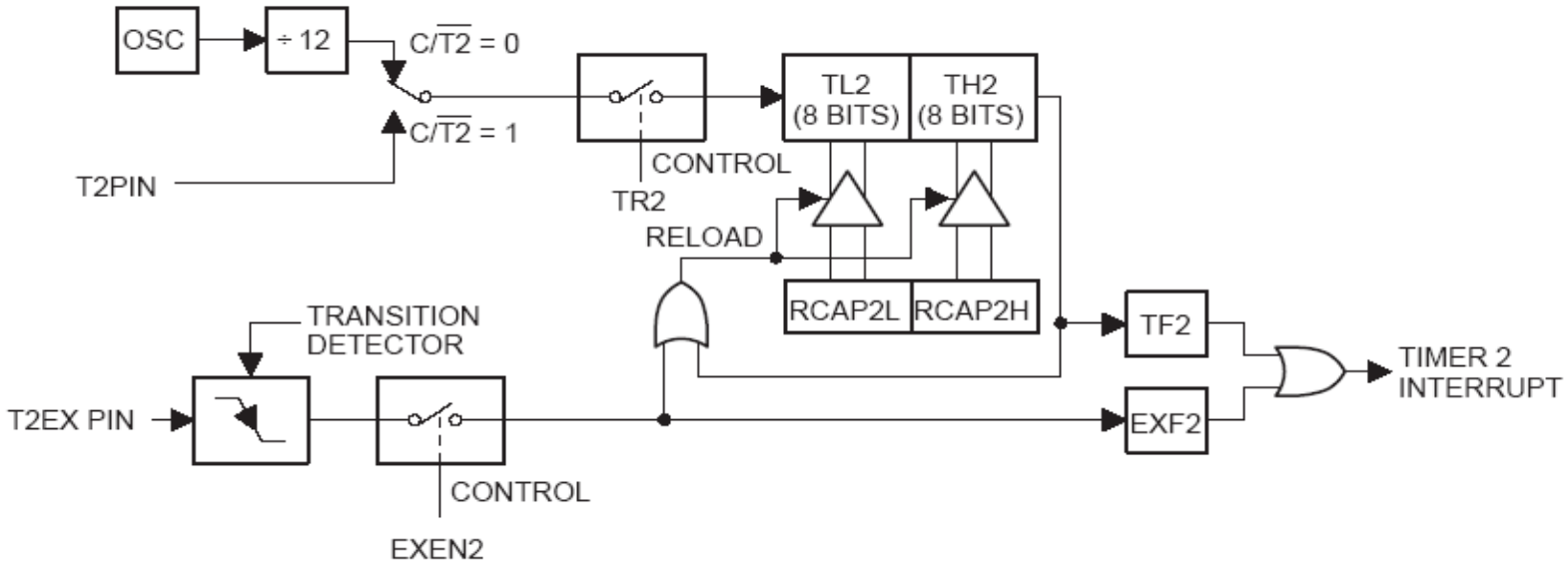




Tajmer 2 (8052)

Kontrolni registar T2CON



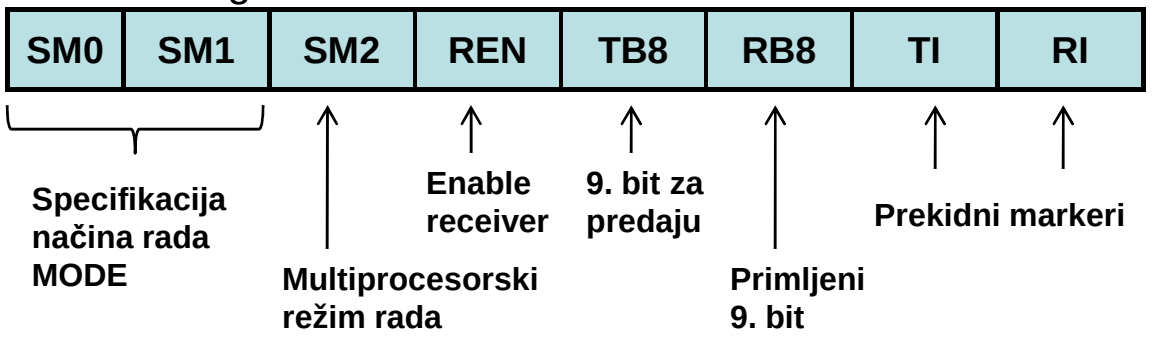


```
;Program na asembleru

        ORG 0               ; reset adresa
        SJMP START          ; rezervisana oblast

        ORG 40H             ; start adresa na 0040H
START:  MOV RCAP2L,#0x32
        MOV RCAP2H,#0xFE
        MOV T2CON,#05H      ; reload mod + START

PETLJA:  JNB TF2, $
        CPL P1.7
        CLR TF2
        SJMP PETLJA         ; repeat
```

UART

Universal Asynchronous Receiver Transmitter

Kontrolni registar SCON



TI postavlja hardver kada se završi slanje karaktera koji je upisan u SBUF

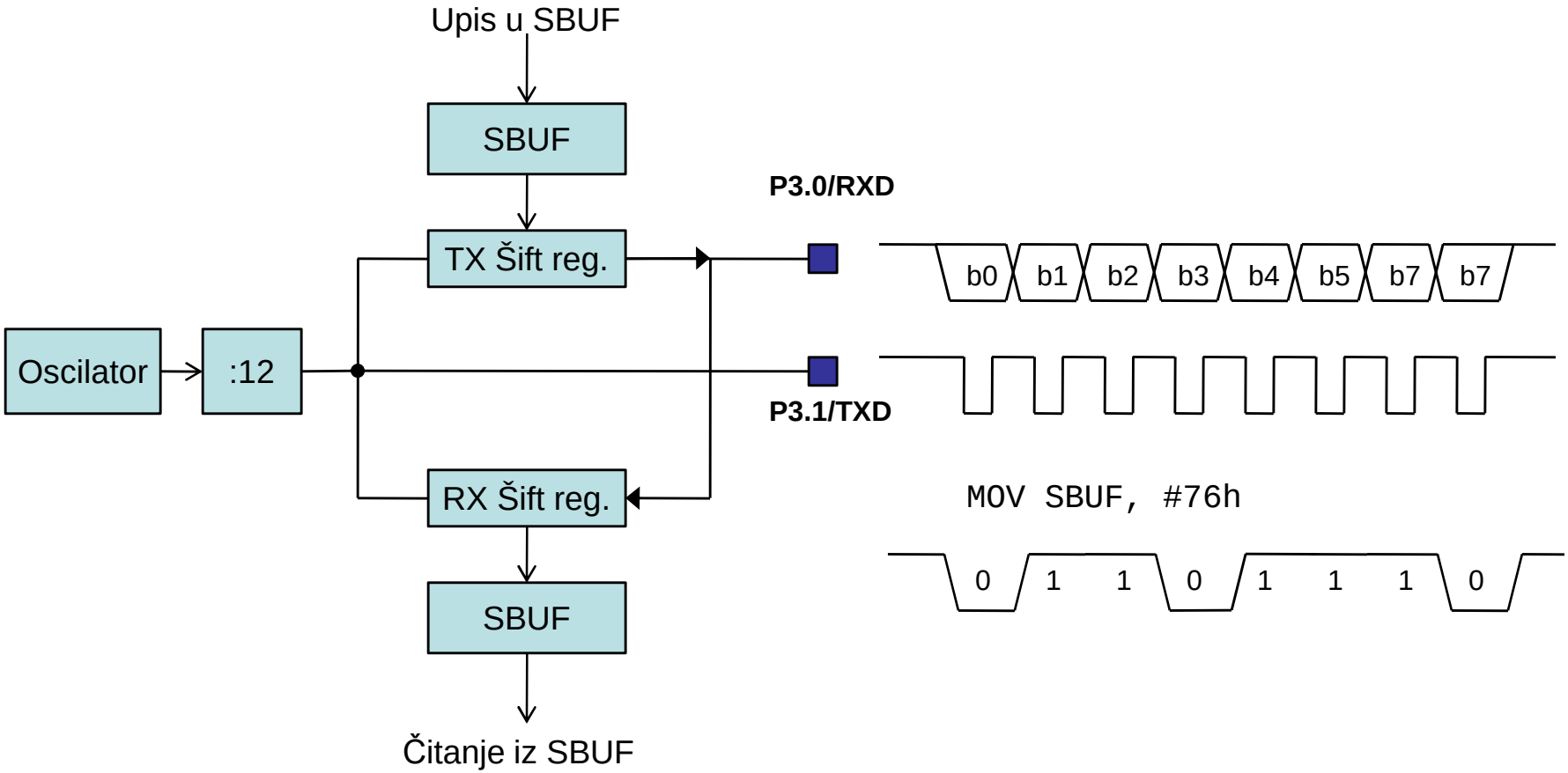
RI postavlja hardver kada u prijemni SBUF stigne novi karakter

SBUF
Prijemnik/Predajnik
Ista adresa

```
MOV SBUF, #'A'
MOV A, SBUF
```

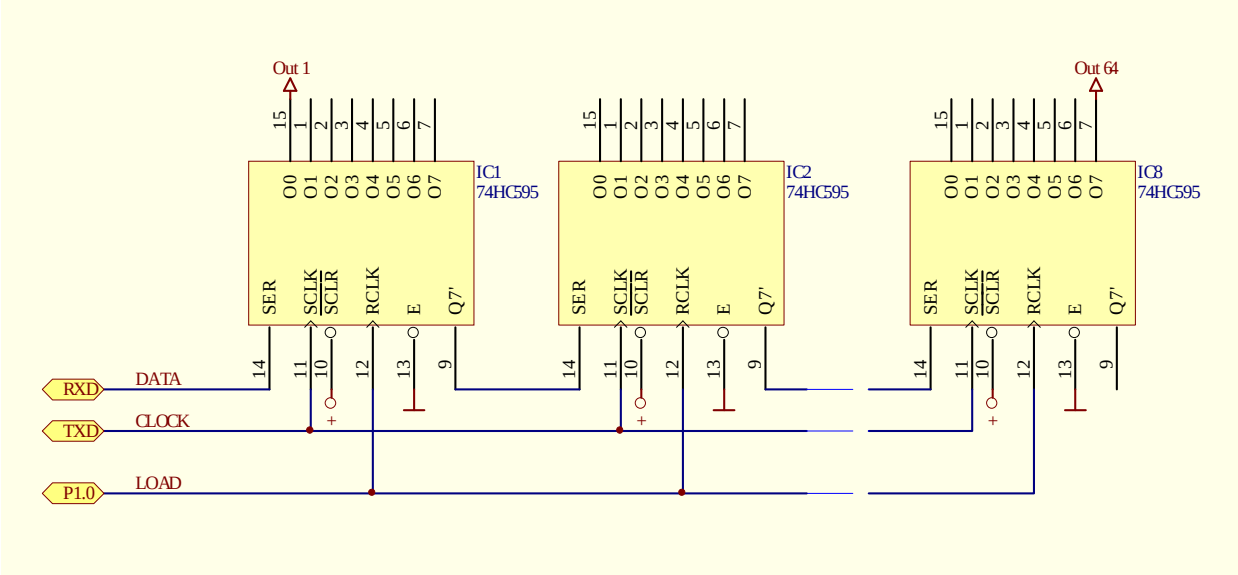
SM0	SM1		
0	0	Mode 0	Sinhroni prenos, šift registri
0	1	Mode 1	UART promenljiva brzina, 8b podatak
1	0	Mode 2	UART fiksna brzina, 9b podatak
1	1	Mode 3	UART promenljiva brzina, 9b podatak

UART - Mode 0



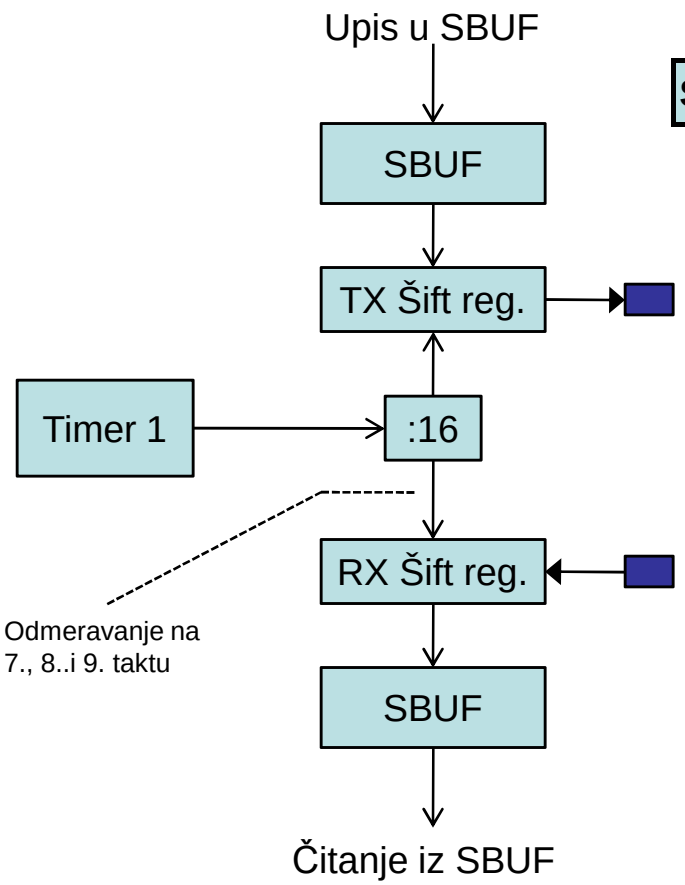
- Predaja podatka počinje upisom u SBUF. Kada se svih 8 bitova pošalju, postavi se marker TI
- Prijem podatka započinje postavljanjem markera RE. Kada se učitaju svh 8 bitova postavi se marker RI

Primer: proširenje na 64 izlazne linije



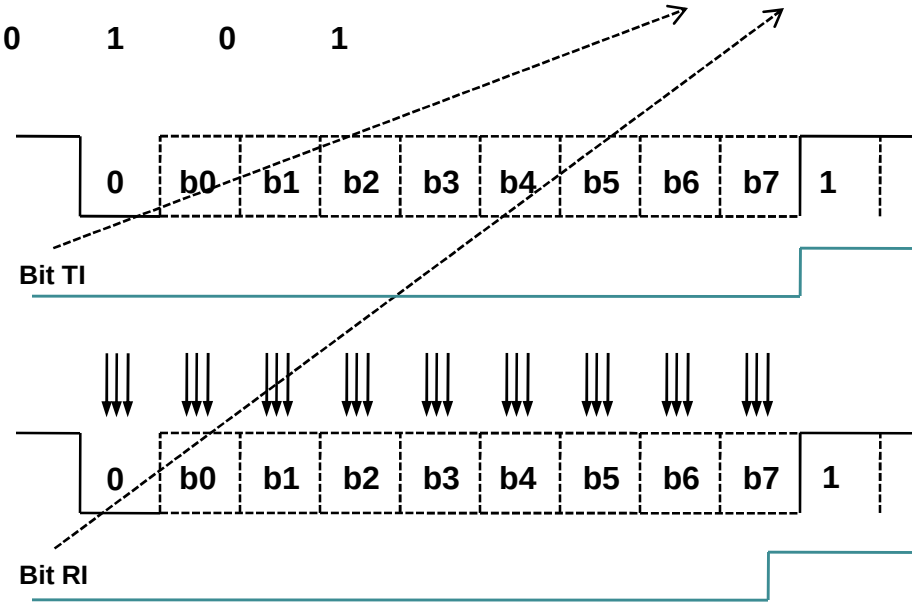

```
;Upis 8 bajtova podataka koji se nalaze u
;IRAM memoriji na adresi 40h na 64 izlazne linije 8 sift registara
;(Serial In Paralel Out).

load sbit p1.0

out64:    mov sbuf,#00h        ; izbor rezima rada UART-a
          clr load             ; inicijalno stanje linije “load”
          mov r0,#40h          ; pocetna adresa bafera podataka
          mov r2,#8            ; broj bajtova
petlja:   mov a,@r0            ; citanje podatka iz bafera
          mov sbuf,a           ; upis u UART, pocetak slanja
          jnb ti,$             ; cekanje na zavrsetak predaje
          clr ti               ; brisanje TI
          djnz r2,petlja       ; petllja za upis potrebnog broja podataka
          setb load            ; generisanje impulsa za upis podatka u izlazni lec
          clr load
          ret
```

UART - Mode 1



SCON

SM0	SM1	SM2	REN	TB8	RB8	TI	RI
0	1	0	1				



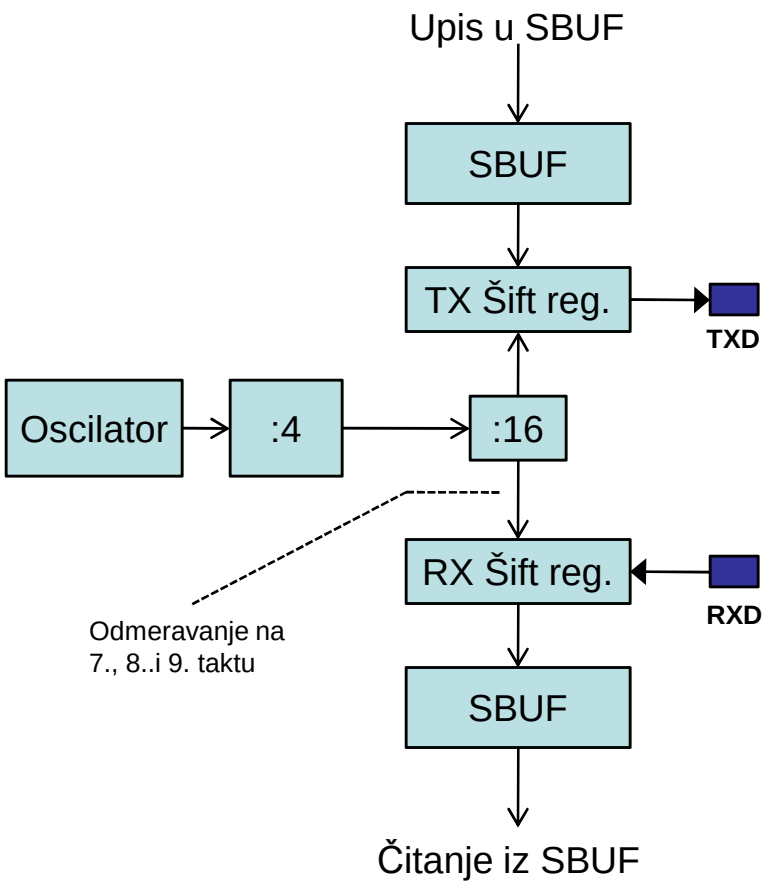
$$Baudrate = \frac{F_{osc}}{12 \times 16 \times (256 - TH1)}$$

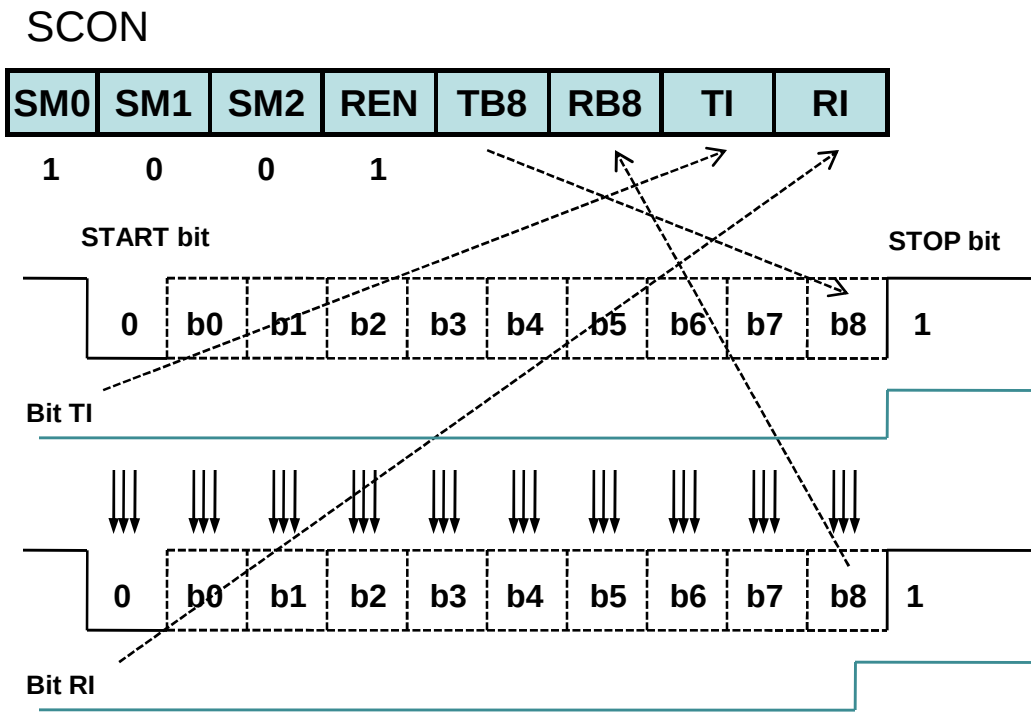
$$TH1 = 256 - \frac{F_{osc}}{192 \times Baudrate}$$

Za F_{osc} = 11.0592 MHz

TH1	Baud
FD(-3)	9600
FA(-6)	4800
F4(-12)	2400
E8(-24)	1200

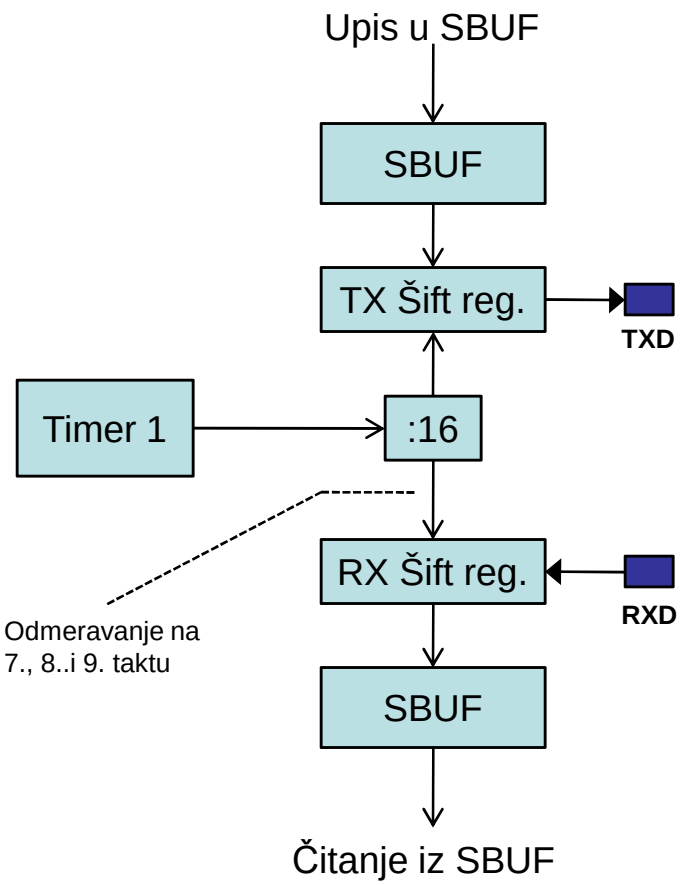
UART - Mode 2





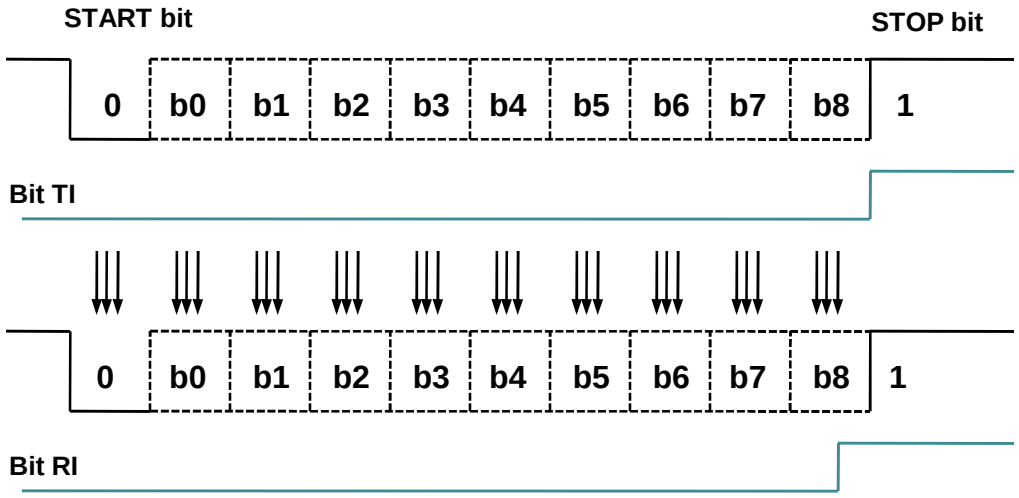
$$Baudrate = \frac{F_{osc}}{64}$$

UART - Mode 3



SCON

SM0	SM1	SM2	REN	TB8	RB8	TI	RI
1	1	0	1				



$$Baudrate = \frac{F_{osc}}{12 \times 16 \times (256 - TH1)}$$

$$TH1 = 256 - \frac{F_{osc}}{192 \times Baudrate}$$

Za F_{osc} = 11.0592 MHz

TH1	Baud
FD(-3)	9600
FA(-6)	4800
F4(-12)	2400
E8(-24)	1200

Osnovne procedure

```
;inicijalizacija serijskog kanala
Ini9600:
      mov tmod,#20h      ;tajmer 1 u autoreload mode
      mov th1,#0fdh      ;baudrate=9600
      mov scon,#50h      ;SCON u mode 1, REN=1
      setb tr1           ;start tajmer
      ret

;slanje karaktera iz akumulatora
SendChar:
      mov sbuf,a  ;iniciranje transfera
      jnb ti,$           ;cekaj dok se ne zavrsi predaja
      clr ti
      ret
```

Osnovne procedure

```
B1200 equ -24
B2400 equ -12
B4800 equ -6
B9600 equ -3

Baud  set B9600

IniSerial:
      mov tmod,#20h      ;tajmer 1 u autoreload mode
      mov th1,#Baud      ;baudrate=9600
      mov scon,#50h      ;SCON u mode 1, REN=1
      setb tr1           ;start tajmer
      ret
```

Primer: Slanje stringa “ELFAK” na RS232

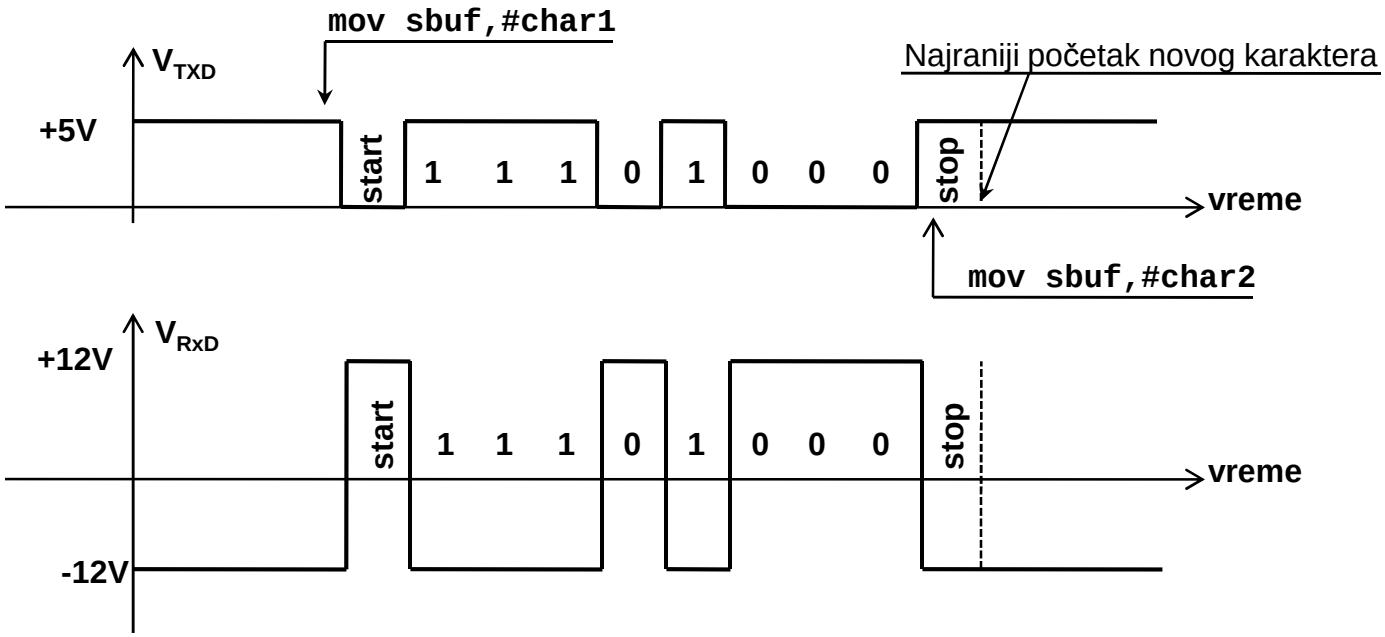
```
;program
          org 0
          jmp start

          org 40h
Start:    acall Ini9600
Petlja:   mov a,#’E’
          acall SendChar
          mov a,#’L’
          acall SendChar
          mov a,#’F’
          acall SendChar
          mov a,#’A’
          acall SendChar
          mov a,#’K’
          acall SendChar
          mov a,#0ah          ; “line feed”
          acall SendChar
          mov a,#0dh          ; “carriage return”
          acall SendChar
          jmp Petlja

;inicijalizacija serijskog kanala
Ini9600:  mov tmod,#20h       ;tajmer 1 u autoreload mode
          mov th1,#0fdh       ;baudrate=9600
          mov scon,#50h       ;SCON u mode 1, REN=1
          setb tr1            ;start tajmer
          ret

;slanje karaktera iz akumulatora
SendChar: mov sbuf,a          ;iniciranje transfera
          jnb ti,$            ;cekaj dok se ne zavrsi predaja
          clr ti
          ret
```

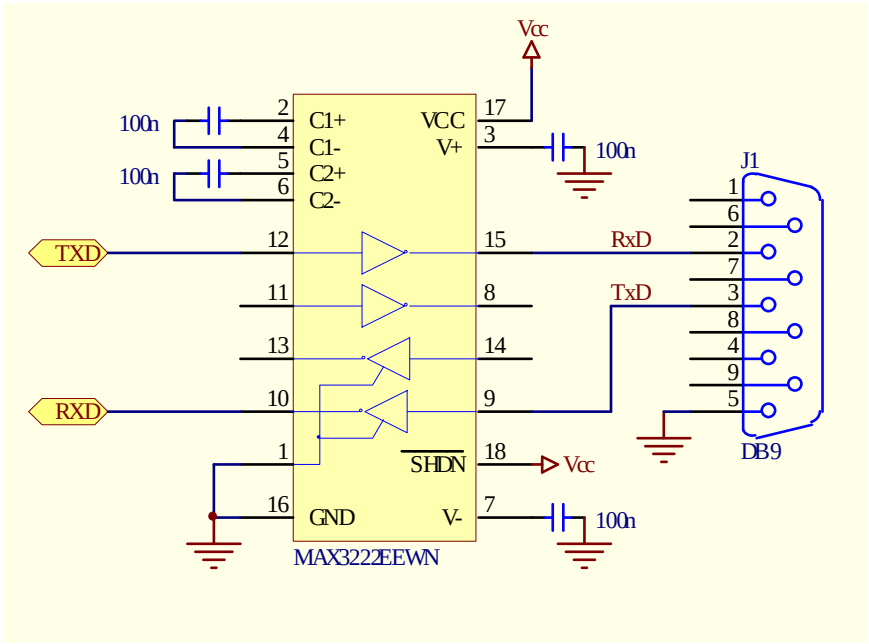
Primer: Prijem karaktera sa RS232

```
;prijem jednog karaktera
ReadChar: jnb ri,$            ;cekaj dok karakter ne bude u SBUF
          mov a,sbuf          ;primljeni karakter u akumulator
          clr ri
          ret

;program
          org 0
          jmp start

          org 40h
Start:    acall Ini9600
Petlja:   acall ReadChar
          anl a,#0DFh         ;mala slova -> velika slova
          acall SendChar
          jmp Petlja
```

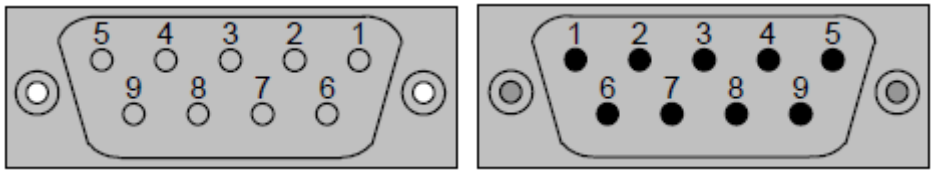
ASCII tabela

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00	Null	32	20	Space	64	40	@	96	60	`	128	80	Ç	160	A0	á	192	C0	└	224	E0	α
1	01	Start of heading	33	21	!	65	41	A	97	61	a	129	81	ü	161	A1	í	193	C1	┴	225	E1	ß
2	02	Start of text	34	22	"	66	42	B	98	62	b	130	82	é	162	A2	ó	194	C2	┬	226	E2	Γ
3	03	End of text	35	23	#	67	43	C	99	63	c	131	83	â	163	A3	ú	195	C3	├	227	E3	π
4	04	End of transmit	36	24	$	68	44	D	100	64	d	132	84	ä	164	A4	ñ	196	C4	─	228	E4	∑
5	05	Enquiry	37	25	%	69	45	E	101	65	e	133	85	à	165	A5	Ñ	197	C5	┼	229	E5	σ
6	06	Acknowledge	38	26	&	70	46	F	102	66	f	134	86	å	166	A6	ª	198	C6	╞	230	E6	µ
7	07	Audible bell	39	27	'	71	47	G	103	67	g	135	87	ç	167	A7	º	199	C7	╟	231	E7	τ
8	08	Backspace	40	28	(	72	48	H	104	68	h	136	88	ê	168	A8	¿	200	C8	╚	232	E8	Φ
9	09	Horizontal tab	41	29	)	73	49	I	105	69	i	137	89	ë	169	A9	⌐	201	C9	╔	233	E9	Θ
10	0A	Line feed	42	2A	*	74	4A	J	106	6A	j	138	8A	è	170	AA	¬	202	CA	╩	234	EA	Ω
11	0B	Vertical tab	43	2B	+	75	4B	K	107	6B	k	139	8B	ï	171	AB	½	203	CB	╦	235	EB	δ
12	0C	Form feed	44	2C	,	76	4C	L	108	6C	l	140	8C	î	172	AC	¼	204	CC	╠	236	EC	∞
13	0D	Carriage return	45	2D	-	77	4D	M	109	6D	m	141	8D	ì	173	AD	¡	205	CD	═	237	ED	ø
14	0E	Shift out	46	2E	.	78	4E	N	110	6E	n	142	8E	Ä	174	AE	«	206	CE	╬	238	EE	ε
15	0F	Shift in	47	2F	/	79	4F	O	111	6F	o	143	8F	Å	175	AF	»	207	CF	╧	239	EF	∩
16	10	Data link escape	48	30	0	80	50	P	112	70	p	144	90	É	176	B0	░	208	D0	╨	240	F0	≡
17	11	Device control 1	49	31	1	81	51	Q	113	71	q	145	91	æ	177	B1	▒	209	D1	╤	241	F1	±
18	12	Device control 2	50	32	2	82	52	R	114	72	r	146	92	Æ	178	B2	▓	210	D2	╥	242	F2	≥
19	13	Device control 3	51	33	3	83	53	S	115	73	s	147	93	ô	179	B3	│	211	D3	╙	243	F3	≤
20	14	Device control 4	52	34	4	84	54	T	116	74	t	148	94	ö	180	B4	┤	212	D4	╘	244	F4	⌠
21	15	Neg. acknowledge	53	35	5	85	55	U	117	75	u	149	95	ò	181	B5	╡	213	D5	╒	245	F5	⌡
22	16	Synchronous idle	54	36	6	86	56	V	118	76	v	150	96	û	182	B6	╢	214	D6	╓	246	F6	÷
23	17	End trans. block	55	37	7	87	57	W	119	77	w	151	97	ù	183	B7	╖	215	D7	╫	247	F7	≈
24	18	Cancel	56	38	8	88	58	X	120	78	x	152	98	ÿ	184	B8	╕	216	D8	╪	248	F8	°
25	19	End of medium	57	39	9	89	59	Y	121	79	y	153	99	Ö	185	B9	╣	217	D9	┘	249	F9	∙
26	1A	Substitution	58	3A	:	90	5A	Z	122	7A	z	154	9A	Ü	186	BA	║	218	DA	┌	250	FA	·
27	1B	Escape	59	3B	;	91	5B	[	123	7B	{	155	9B	¢	187	BB	╗	219	DB	█	251	FB	√
28	1C	File separator	60	3C	<	92	5C	\	124	7C	|	156	9C	£	188	BC	╝	220	DC	▄	252	FC	ⁿ
29	1D	Group separator	61	3D	=	93	5D	]	125	7D	}	157	9D	¥	189	BD	╜	221	DD	▌	253	FD	²
30	1E	Record separator	62	3E	>	94	5E	^	126	7E	~	158	9E	₧	190	BE	╛	222	DE	▐	254	FE	■
31	1F	Unit separator	63	3F	?	95	5F	_	127	7F	⌂	159	9F	ƒ	191	BF	┐	223	DF	▀	255	FF	





PC



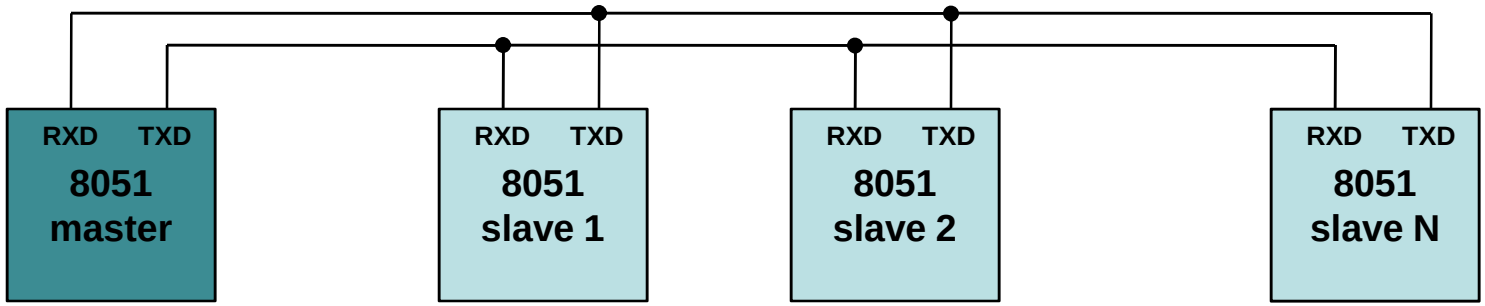
Ženski (*Female*) **Muški (*Male*)**

Multiprocesorska komunikacija

Uslov za upis karaktera iz prijemnog šift registra u SBUF:

(RI = 0) AND [(SM2 = 0) OR (primljeni 9.bit = 1)]

- **Ako prethodni karakter nije očitan (i obrisan RI) novi karakter se bespovratno gubi.**
- **Ako je SM2 = 1 i primljeni 9.bit = 0, marker RI neće biti postavljen i prijem karaktera se ne signalizira (ne izaziva prekid)**



Multiprocesorska komunikacija

Master može da šalje podatke svima i da prima podatke od svih
Svaki slejv može da primi podatke samo od mastera
Svaki slejv ima adresu kao jedinstveni 8-bitni podatak

<u>Protokol razmene podatka:</u>

SLAVE

- **Inicijalno su konfigurisani za prijem 9-bitnih podataka i multiprocesorski rad, SM2=1.**

MASTER

- **Šalje adresu slejva kome su namenjeni podaci kao 9-bitni podatak sa TB8=1.**
- **Šalje potreban broj bajtova podataka kao 9-bitne sa TB8=0**

SLAVE

- **Inicijalno su konfigurisani za prijem 9-bitnih podataka i multiprocesorski rad, SM2=1.**
- **Svi slejvovi prime adresni bajt i upoređuju adresu sa sopstvenom.**
- **Adresirani slejv vrši rekonfiguraciju tako da prima i bajtove sa 9. bitom jednakim nuli, SM2=0, i nastavlja sa prijemom ostalih bajtova podataka**
- **Ostali slejvovi ne vrše rekofiguraciju i ne primaju podatke koji nisu namenjeni njima.**

Kako obezbediti da svako može sa svakim da komunicira?

Prekidni sistem

Prekid, *interrupt* , predstavlja pojavu nekog stanja (ili događaja - *event*) koje uzrokuje privremenu suspenziju izvršenja tekućeg programa i servisiranje novonastalog stanja.

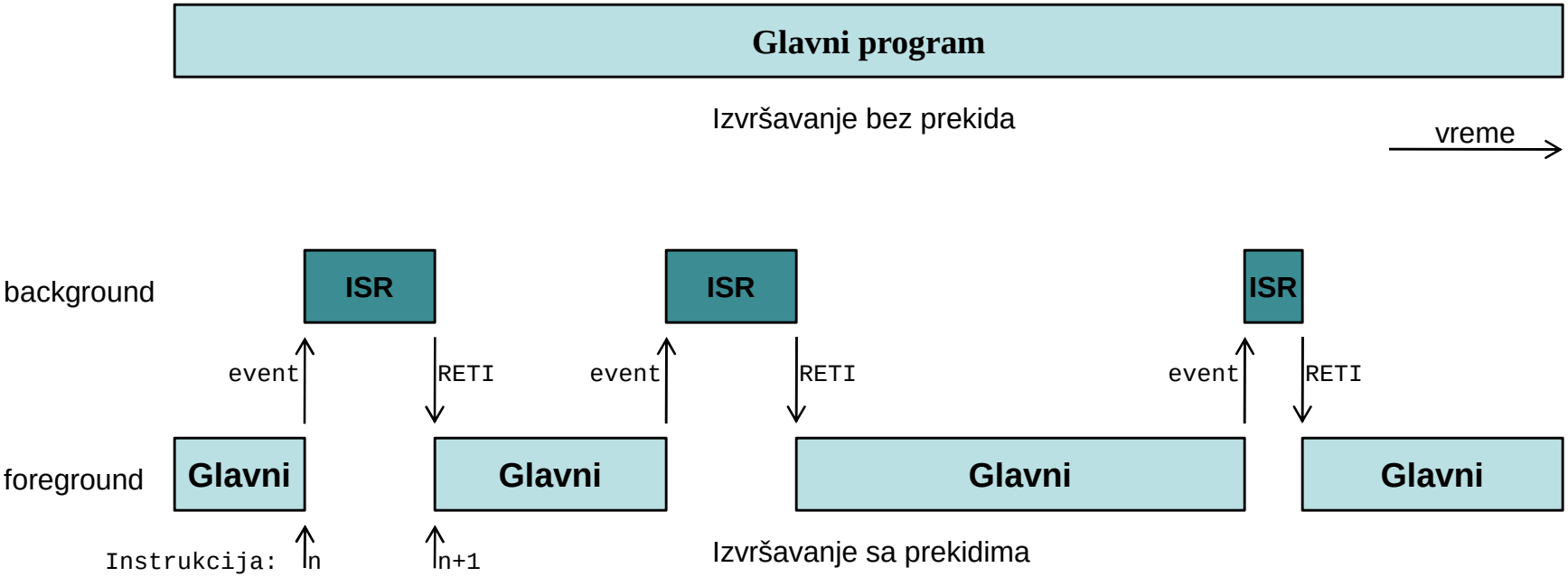
Omogućava sistemu da odgovori na asinhrone događaje do kojih dolazi u toku izvršavanja drugih programa.

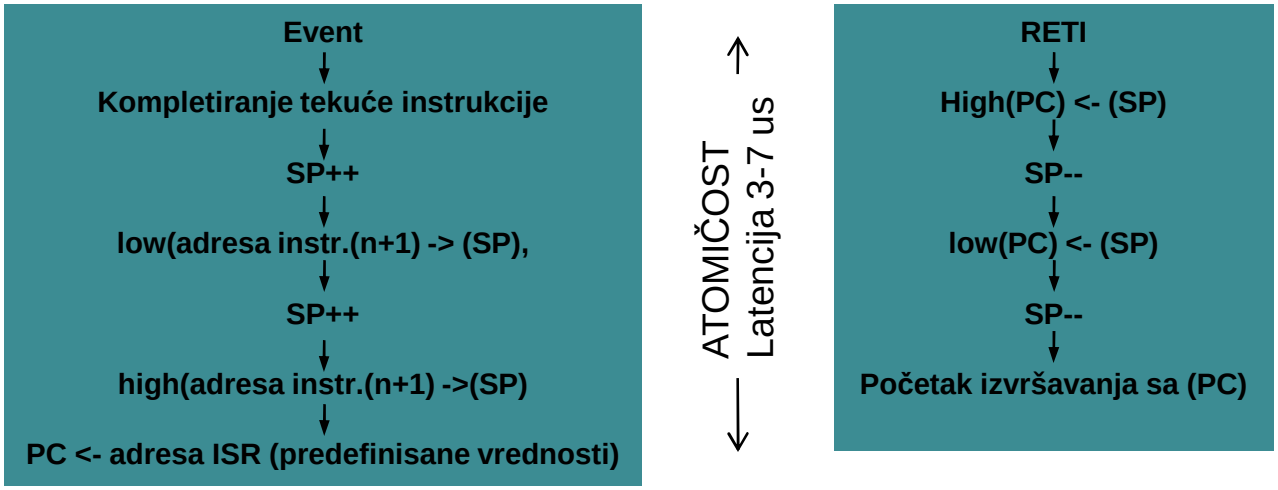
Interrupt driven system **daje iluziju simultanog izvršavanja većeg broja programa.**

Prekidna rutina, *Interrupt Service Routine – ISR, interrupt handler*, je program koji se izvršava nakon pojave prekida. ISR se izvršava do instrukcije "povratak iz prekida", (*RETurn from Interrupt*), nakon čega sa izvršenjem nastavlja suspendovani program.

<u>Usvojena terminologija:</u>
Glavni program se izvršava na osnovnom nivou (*foreground*), a *ISR* na prekdnom nivou (*background*)

Prekidni sistem

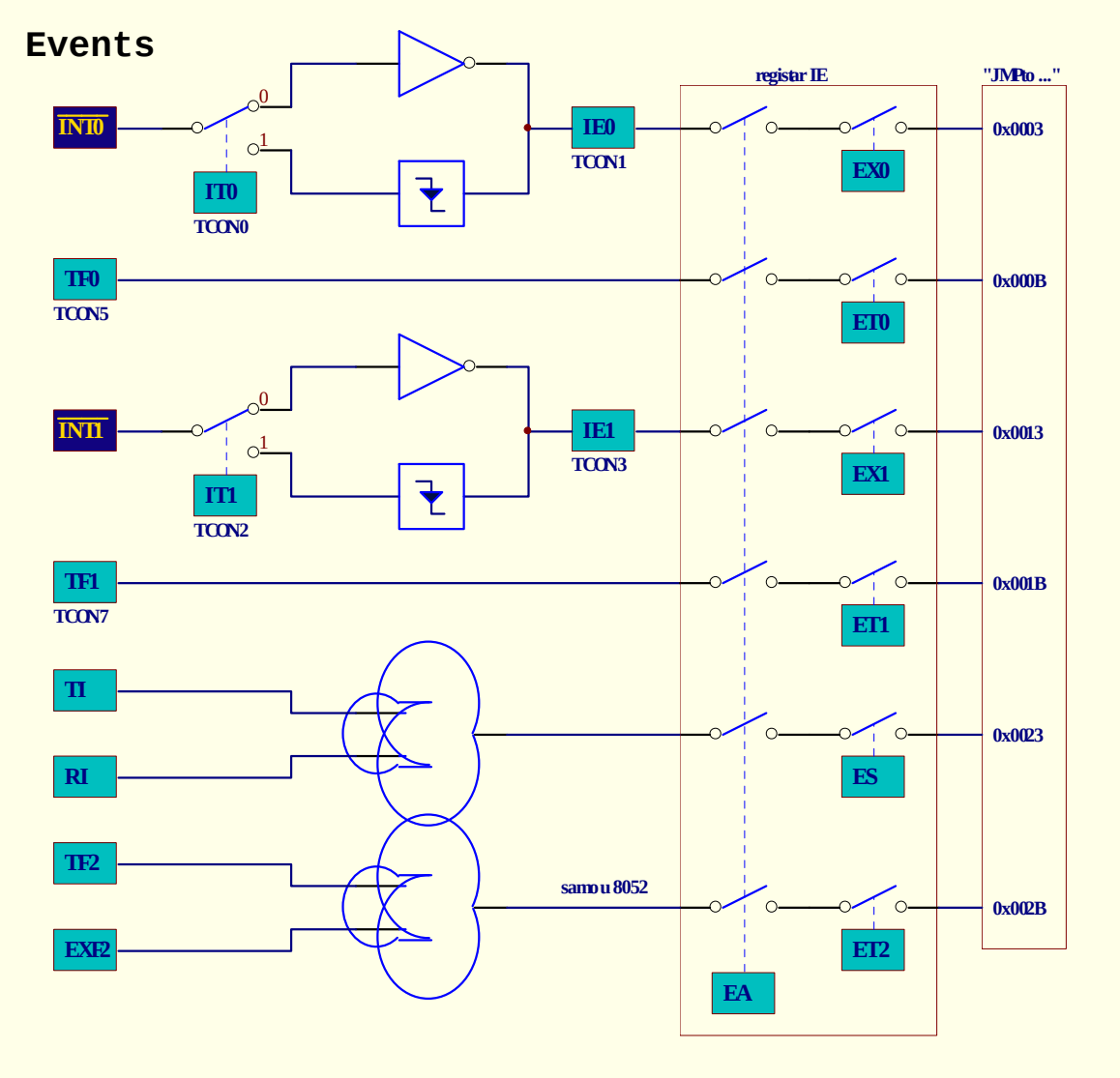




Prekidni sistem

Registar dozvole prekida IE

EA	-	ET2 8052	ES	ET1	EX1	ET0	EX0



Automatsko brisanje markera pri izlazu iz prekidne rutene:
IE0, IE1, TF0, TF1

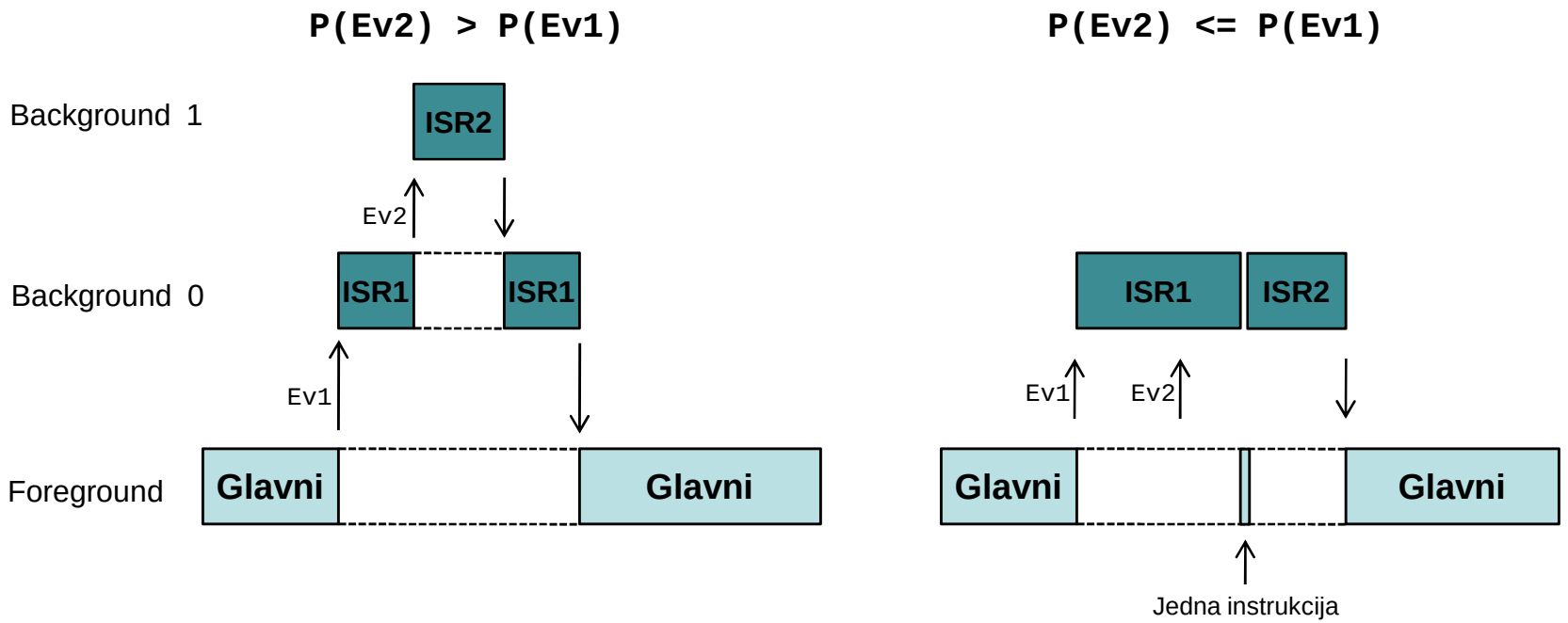
Markeri koji se moraju brisati softverski:
RI, TI, TF2, EXF2

Prekidni sistem

Prioritet prekida

0 – niži prioritet

1 – viši prioritet



Prekidni sistem

Male prekidne rutine – *Small interrupt service routines*

```
         ORG 0000H
         LJMP MAIN

         ORG 000BH
T0ISR:   - - - - -
         - - - - -
         - - - - -
         - - - - -
         RETI
```

Velike prekidne rutine – *Large interrupt service routines*

```
         ORG 0000H
         LJMP MAIN

         ORG 000BH
         LJMP T0ISR

         ORG 0040H
MAIN:    - - - - -
         - - - - -
         - - - - -

T0ISR:   - - - - -
         - - - - -
         - - - - -
         RETI
```

Primer – tajmerski prekidi

Na linijama P1.6 i P1.7 generisati dva simetrična talasna oblika frekvencija 7 kHz i 500 Hz respektivno. Mikrokontroler koristi kvarcni oscilator frekvencije 12 MHz.

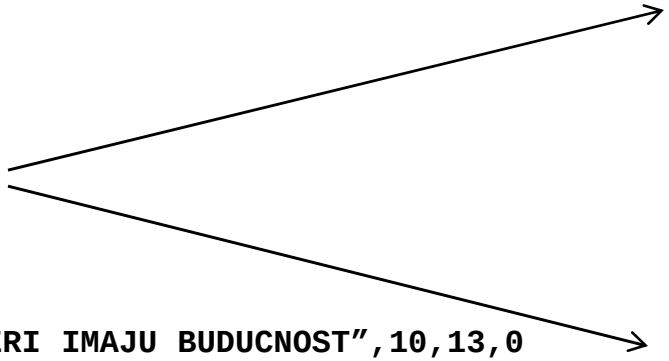
```
        ORG 0000H
        LJMP MAIN

        ORG 000BH                ; vektorska adresa prekida T0
        CPL P1.6
        RETI

        ORG 001BH                ; vektorska adresa prekida T1
        JMP T1ISR

        ORG 0040H
MAIN:   MOV TMOD,#12H            ; T0 autoreload, T1 16 bit
        MOV TH1,#-71             ; poluperioda signala 7kHz = 71us
        SETB TR0
        SETB TR1
        MOV IE,#8AH              ; Dozvoli oba tajmerska prekida
        SJMP $

T1ISR:  CLR TR1
        MOV TH1,#HIGH(-1000)
        MOV TL1,#LOW(-1000)
        SETB TR1
        CPL P1.7
        RETI
```

Uticaj prioriteta prekida!

Primer – serijski prekidi

Poruku odredjenog sadrzaja ispisivati na serijski kanal svakih 50 ms. Ispisivanje vrsiti brzinom od 9600 bauda.

```
          ORG 0000H
          LJMP MAIN

          ORG 0023H                    ; vektorska adresa prekida serijskog kanala
          JMP SKISR

          ORG 0040H
MAIN:     MOV TMOD,#21H                ; T0 autoreload, T1 16 bit
          MOV TH1,#0FDH                ; Baud=9600
          MOV SCON,#50H                ; MODE 1, REN=1
          MOV IE,#90H                  ; Dozvoli serijski prekid
          SETB TR1                     ; Start Baudrate generator - T1
PETLJA:   CLR TR0
          MOV TH0,#HIGH(-50000)        ; poluperioda signala 7kHz = 71us
          MOV TL1,#LOW(-50000)
          SETB TR0
          JNB TF0,$
          MOV DPTR,#PORUKA             ; Pointer na poruku
          SETB TI                      ; Aktiviranje slanja
          JMP PETLJA

T1ISR:    MOVC A,@A+DPTR
          INC DPTR
          CLR TI
          JZ $+3
          MOV SBUF,A
          RETI

PORUKA:   DB “MIKROKONTROLERI IMAJU BUDUCNOST”,10,13,0
```



```
PUSH ACC
PUSH PSW
MOVC A,@A+DPTR
INC DPTR
CLR TI
JZ $+3
MOV SBUF,A
POP PSW
POP ACC
RETI
```

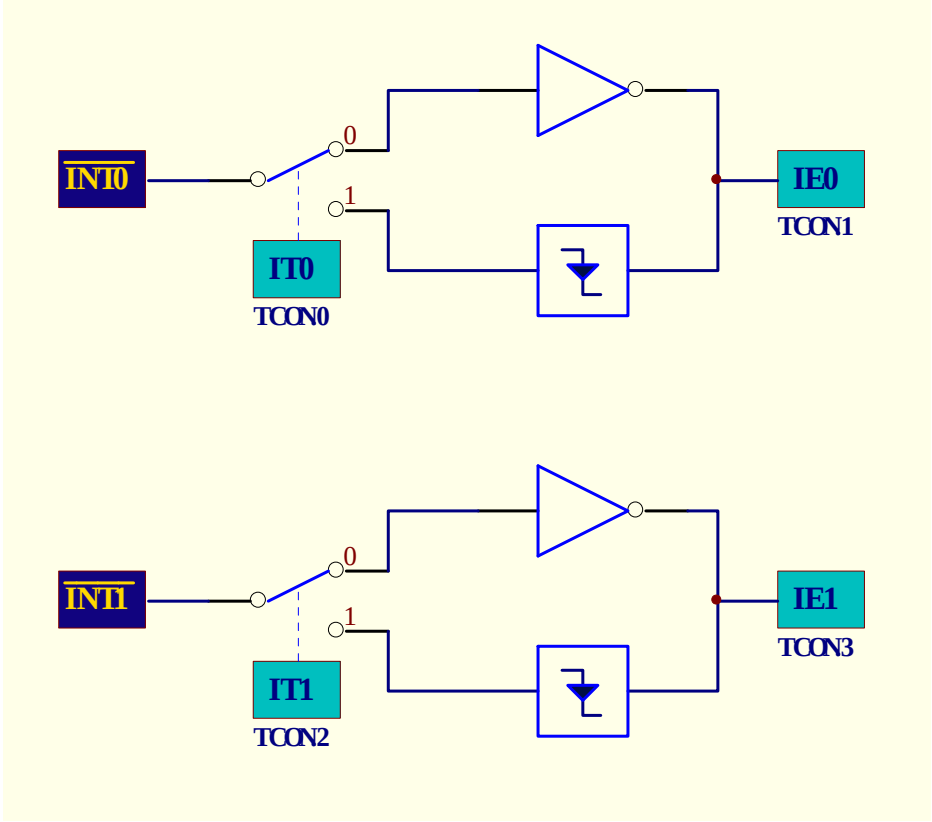
Spoljašnji prekidi

- **Do spoljašnjeg prekida dolazi pojavom logičke nule ili negativne ivice na linijama porta P3.2 ($\overline{INT0}$) i P3.3 ($\overline{INT1}$).**
- **Izbor osetljivosti na logički nivo ili ivicu signala – markeri IT0 i IT1 registra TCON.**

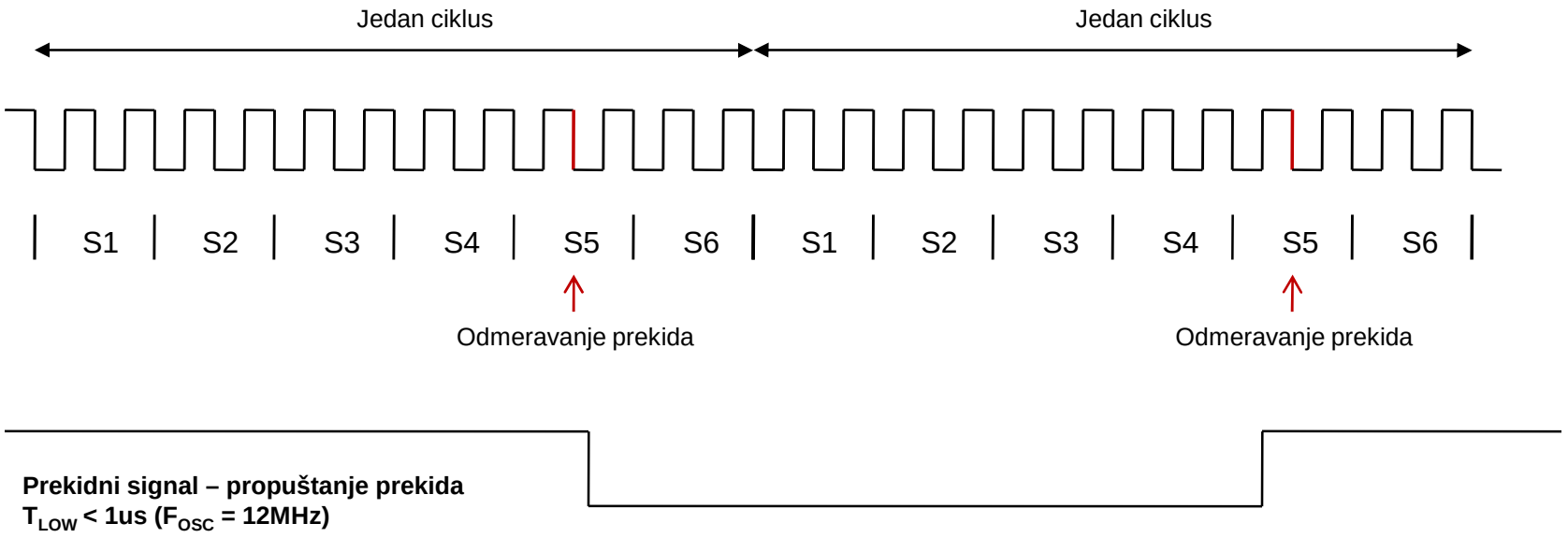
TCON

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

Kontrola prekidnog sistema

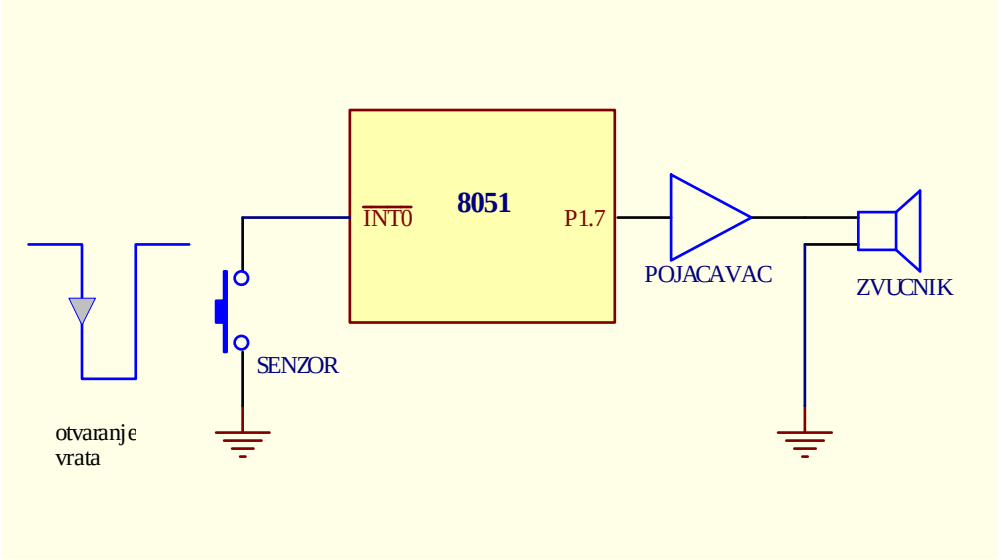


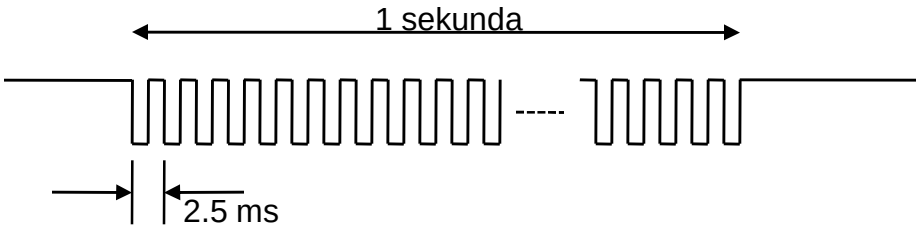
- **Odmeravanje logičkog nivoa ulazne linije u određenom trenutku mašinskog ciklusa - Najmanje vreme trajanja prekidnog signala 12xFosc.**



Spoljašnji prekidi - primer

Alarmni sistem koji na aktiviranje prekidača generiše zvučni ton frekvencije 400 Hz trajanja jedne sekunde. Zvučnik je preko izlaznog drajvera vezan za liniju P1.7. Senzor-mikroprekidač koji detektuje negativnu ivicu je vezan za liniju /INT0.





Generisanje velikih vremenskih intervala!

Primer – generisanje alarma

```
          ORG 0000H
          LJMP MAIN
          ORG 0003H
          LJMP EX0ISR
          ORG 000BH
          LJMP T0ISR
          ORG 001BH
          LJMP T1ISR

          ORG 0040H
MAIN:     MOV TMOD,#11H              ; T0, T1 16 bit
          MOV IE,#8AH                ; Dozvoli prekid sa /INT0
          SJMP $

EX0ISR:   MOV R7,#20                 ; perioda T1 kao 20x50ms = 1 sek
          SETB TF0                   ; aktiviraj prekid tajmer 0 – softversko aktiviranje
          SETB TF1                   ; aktiviraj prekid tajmer 1
          SETB ET0                   ; dozvoli prekide – za 1 sek.
          SETB ET1                   ; za generisanje 400 Hz
          RETI

T0ISR:    CLR TR0                    ; zaustavi tajmer 0
          DJNZ R7,SKIP               ; da li je istekla jedna sekunda
          CLR ET0                    ; istekla jedna sekunda, zabrani tonski signal
          CLR ET1                    ; zabrani sopstveni prekid
          SJMP EXIT
SKIP:     MOV TH0,#HIGH(-50000)
          MOV TL0,#LOW(-50000)
          SETB TR0
EXIT:     RETI

T1ISR:    CLR TR1                    ;
          MOV TH1,#HIGH(-1250)
          MOV TL1,#LOW(-1250)
          SETB TR1
          CPL P1.7
          RETI
```