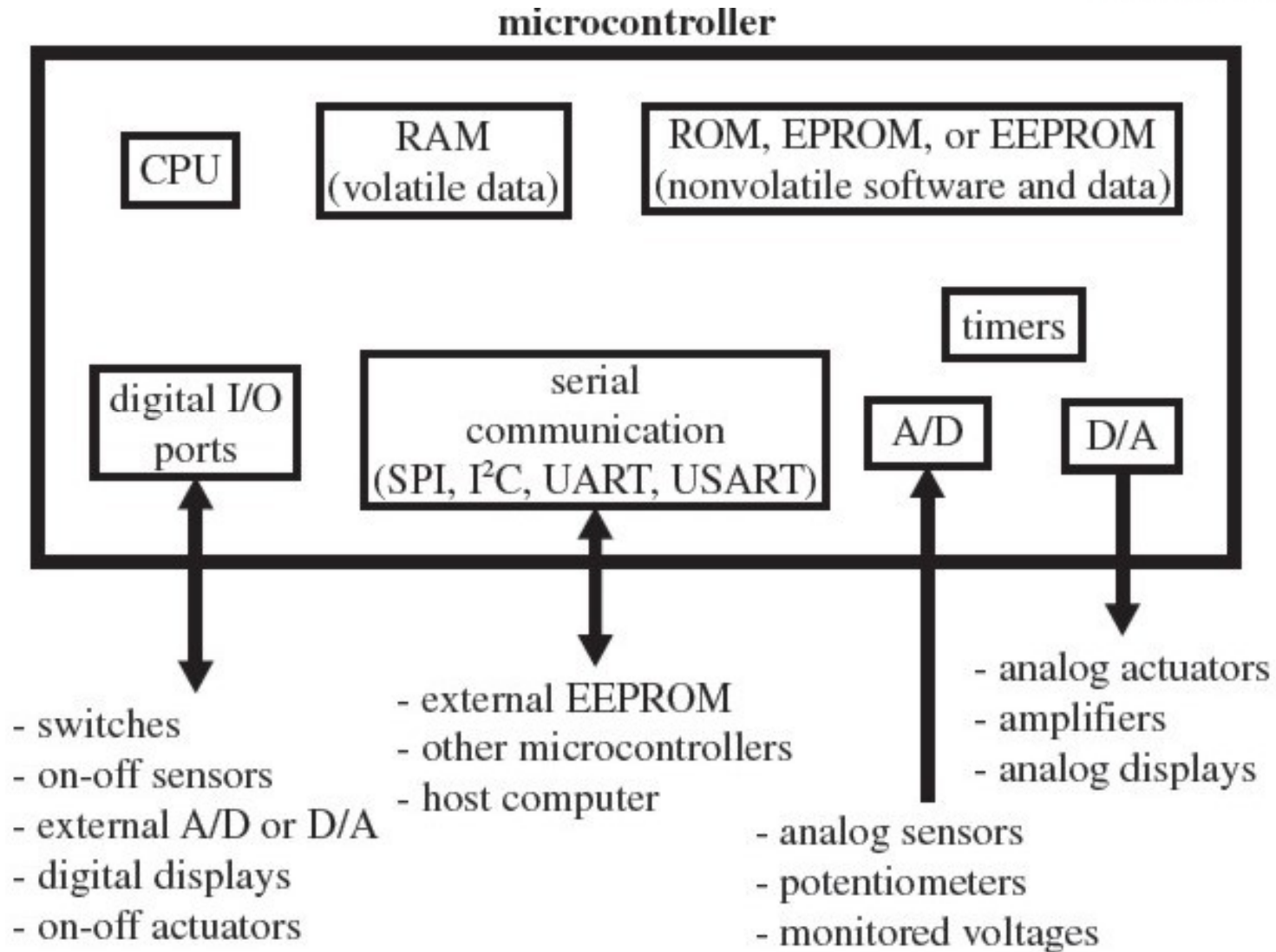


Računske vežbe iz
Projektovanja Elektronskih
Sistema
čas 3

Doc.dr Borisav Jovanović

Sadržaj:

- Elektronski sistemi bazirani na mikrokontrolerima
- Moduli potrebni za realizaciju projektnih zadataka.
- Podešavanje pinova, analogni/digitalni, ulazni/izlazni.
- Podešavanje tajmera mikrokontrolera.
- Rad sa prekidima.
- Upotreba A/D konvertora.
- Rad sa UART modulom.

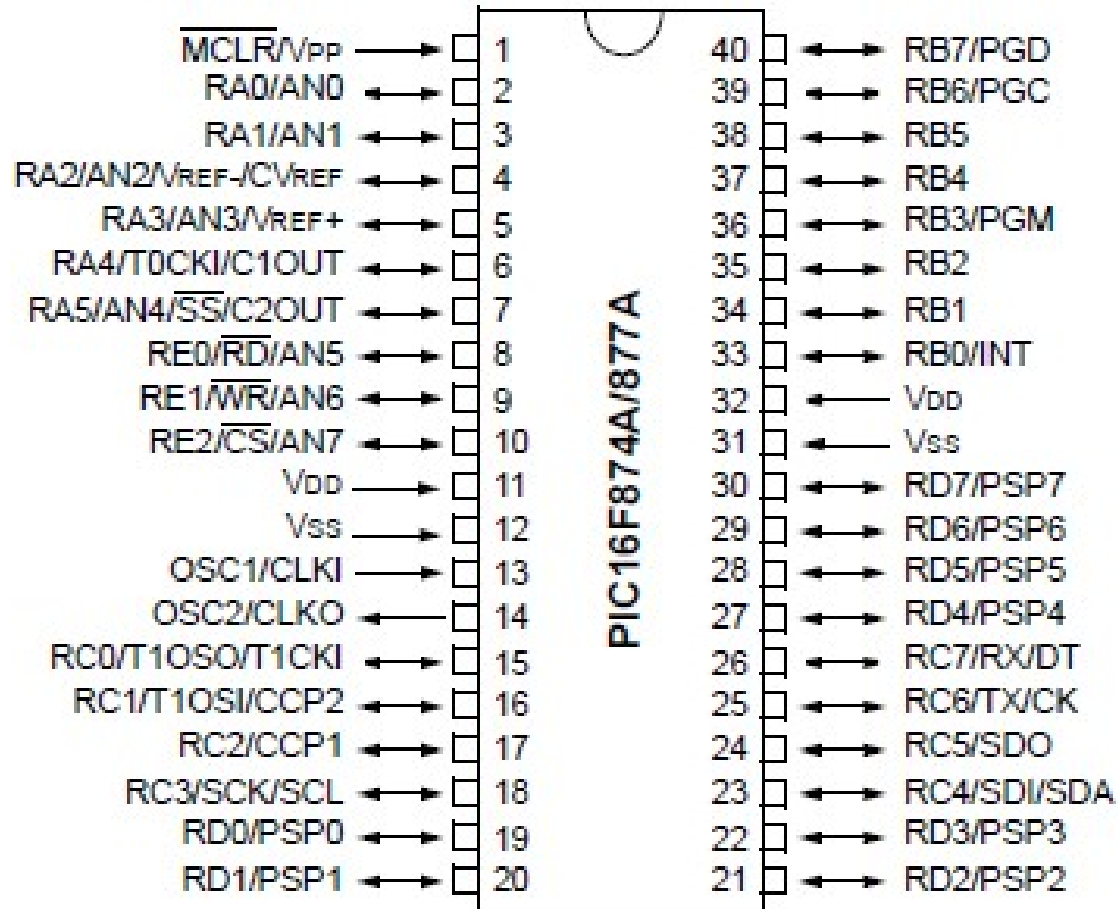


Podešavanje pinova mikrokontrolera PIC16F877A

- Mikrokontroler PIC16F877A koji se koristi za implementaciju Slejv automata ima 5 portova A, B, C, D i E.
- Port A ima 5 pinova, portovi B, C i D po 8 pinova dok port E ukupno 3 pina.
- Pinovi mogu da budu digitalni ili analogni (analogni pin je ulaz AD konvertora koji je deo mikrokontrolera)
- Da li je neki pin podešen da bude digitalni ili analogni podešava se u ADCON0 specijalnom registru, preko PCFG3- PCFG0



40-Pin PDIP



ADCON1 REGISTER (ADDRESS 9Fh)

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	ADCS2	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7				bit 0			

bit 3-0 PCFG3:PCFG0: A/D Port Configuration Control bits

PCFG <3:0>	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0	VREF+	VREF-	C/R
0000	A	A	A	A	A	A	A	A	VDD	VSS	8/0
0001	A	A	A	A	VREF+	A	A	A	AN3	VSS	7/1
0010	D	D	D	A	A	A	A	A	VDD	VSS	5/0
0011	D	D	D	A	VREF+	A	A	A	AN3	VSS	4/1
0100	D	D	D	D	A	D	A	A	VDD	VSS	3/0
0101	D	D	D	D	VREF+	D	A	A	AN3	VSS	2/1
011x	D	D	D	D	D	D	D	D	—	—	0/0
1000	A	A	A	A	VREF+	VREF-	A	A	AN3	AN2	6/2
1001	D	D	A	A	A	A	A	A	VDD	VSS	6/0
1010	D	D	A	A	VREF+	A	A	A	AN3	VSS	5/1
1011	D	D	A	A	VREF+	VREF-	A	A	AN3	AN2	4/2
1100	D	D	D	A	VREF+	VREF-	A	A	AN3	AN2	3/2
1101	D	D	D	D	VREF+	VREF-	A	A	AN3	AN2	2/2
1110	D	D	D	D	D	D	D	A	VDD	VSS	1/0
1111	D	D	D	D	VREF+	VREF-	D	A	AN3	AN2	1/2



- Da li je neki pin podešen da bude digitalni ili analogni podešava se u ADCON0 specijalnom registru, preko PCFG3-PCFG0
- Potražiti u datasheet-u PIC16F877A registrar ADCON1 i proveriti podešavanja za analogne ulaze
- Primer: `ADCON1=0b00000110;` //svi digitalni pinovi



- Kada je neki pin podešen da bude digitalni (preko ADCON1), može da bude digitalni ulaz ili izlaz, što se podešava preko 8-bitnog TRIS registra;
- Svaki port ima odgovarajući TRIS registar, recimo port A ima TRISA, port B – TRISB, itd.
- Ako je bit TRIS registra na nuli, tada je odgovarajući pin podešen da bude izlaz (može da upravlja LED diodu).
- Ako je bit TRIS registra na jedinici, tada je odgovarajući pin podešen da bude ulaz (prati stanje tastera ili prekidača).
- Obično se TRIS registri podešavaju u **init()** funkciji.

Primeri:

`TRISA=0b00001000; // pin RA3 je podešen da bude dig. ulaz, svi ostali pinovi na portu A su digitalni izlazi`

`TRISB=0b11111110; // svi pinovi su ulazi, osim RB0 koji je izlazni pin`

`TRISA=0x00; // pinovi na portu A su digitalni izlazi;`

`TRISB=0xFF; // pinovi na portu B su SVI digitalni ulazi`

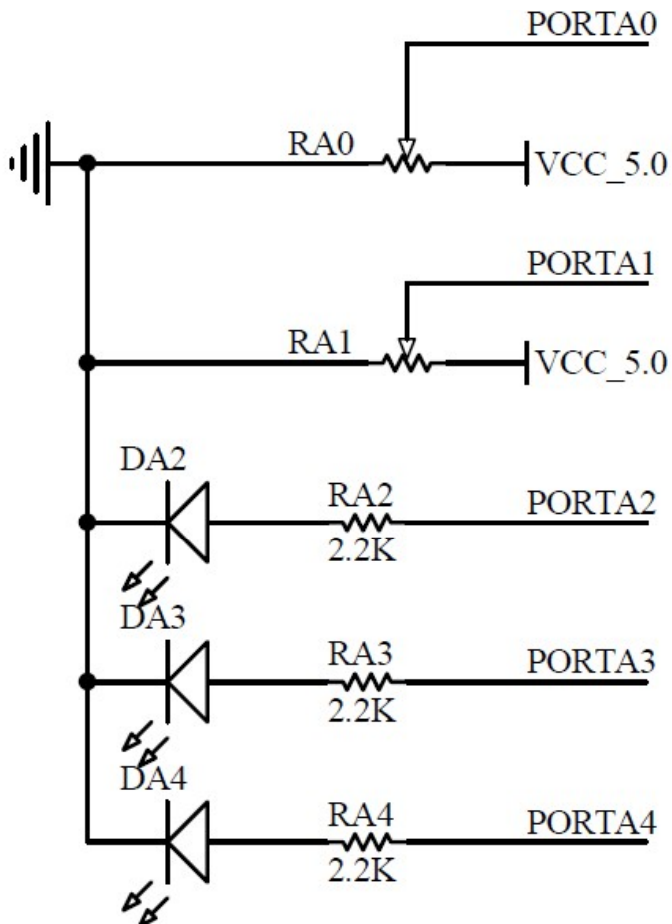
`TRISC=0xC0; // pinovi 6 i 7 su vezani za RS232`

- Kada je neki pin MCU podešen da bude digitalni izlaz, tada se on postavlja u stanje log. 0 ili 1 preko odgovarajućeg PORT registra.
- Svaki port ima odgovarajući PORT registar, recimo port A ima PORTA, port B – PORTB, itd.
- Ako je bit PORT registra na nuli, tada je odgovarajući pin resetovan (LED dioda ugašena)
- Sadržaj PORT registra može da se menja tokom rada programa a ne samo tokom inicijalizacije čipa

Primeri:

`PORTA=0b00001000; // pin RA3 je setovan, svi
ostali pinovi na portu A su resetovani`

`PORTB=0b11111110; // svi digitalni izlazi su
setovani, osim RB0 koji je resetovan`



Slejev ploča

LED diode - izlazi

potencijometri su analogni ulazi

```
ADCON1=0b00001110;
```

```
// samo je PORTA0 analogni ulaz
```

```
// PORTA1 ne može da bude iskoriscen  
kao analogni ulaz, mora da bude  
digitalni ulaz
```

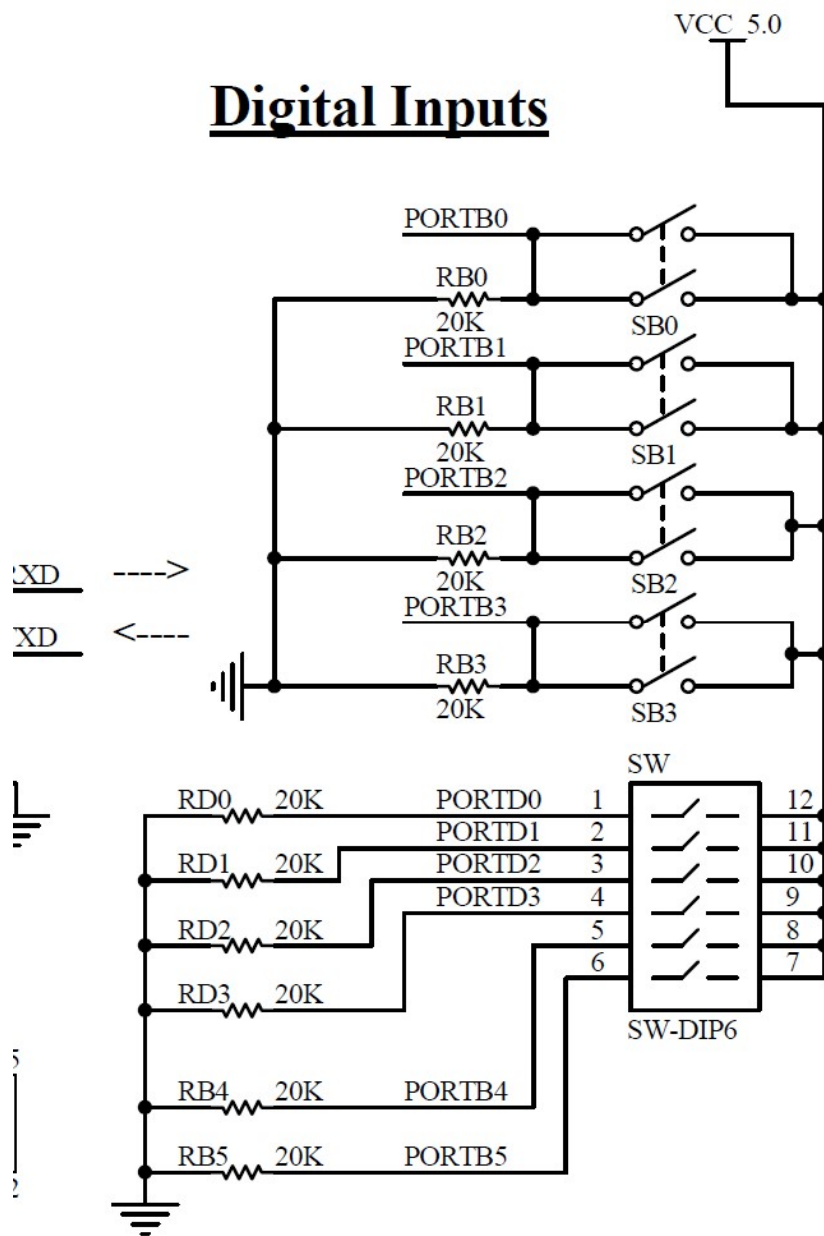
```
TRISA=0b00000011
```

```
PORTA=0b00000000
```

```
// LED diode su ugasene
```

```
PORTA.F2 = 1; // pali LED diodu DA2
```

```
PORTA.F2 = 0; // gasi
```



Slejev ploča

pinovi PORTB0 – PORTB5 su dig.ulazi (prekidaci i tasteri)

TRISB=0b00111111;

//ili ovako TRISB=0x3F

if (PORTB.F0 == 1) ...

// da li je taster PORTB0 pritisnut

pinovi PORTD0 – PORTD3

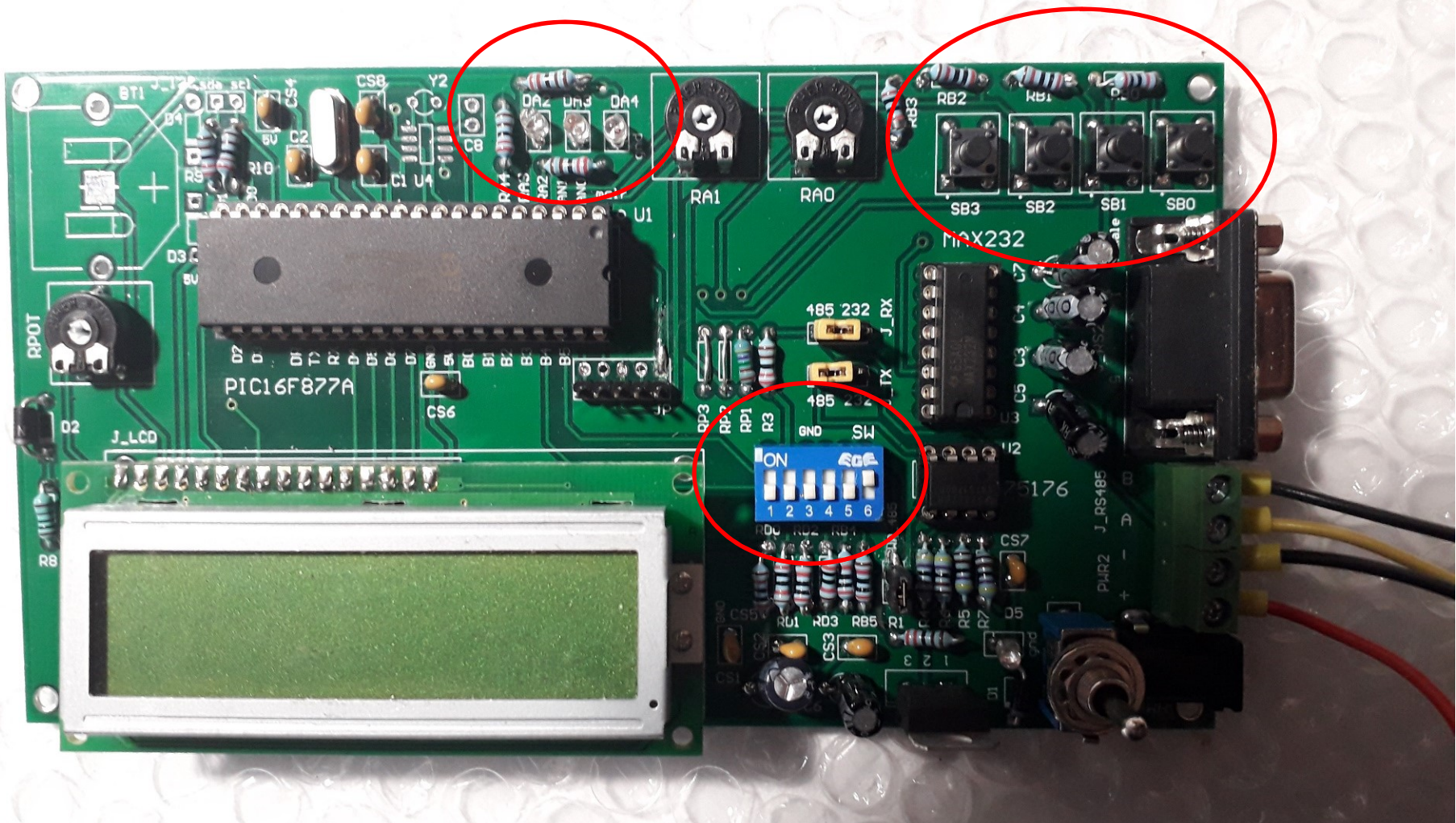
su digitalni ulazi (prekidaci)

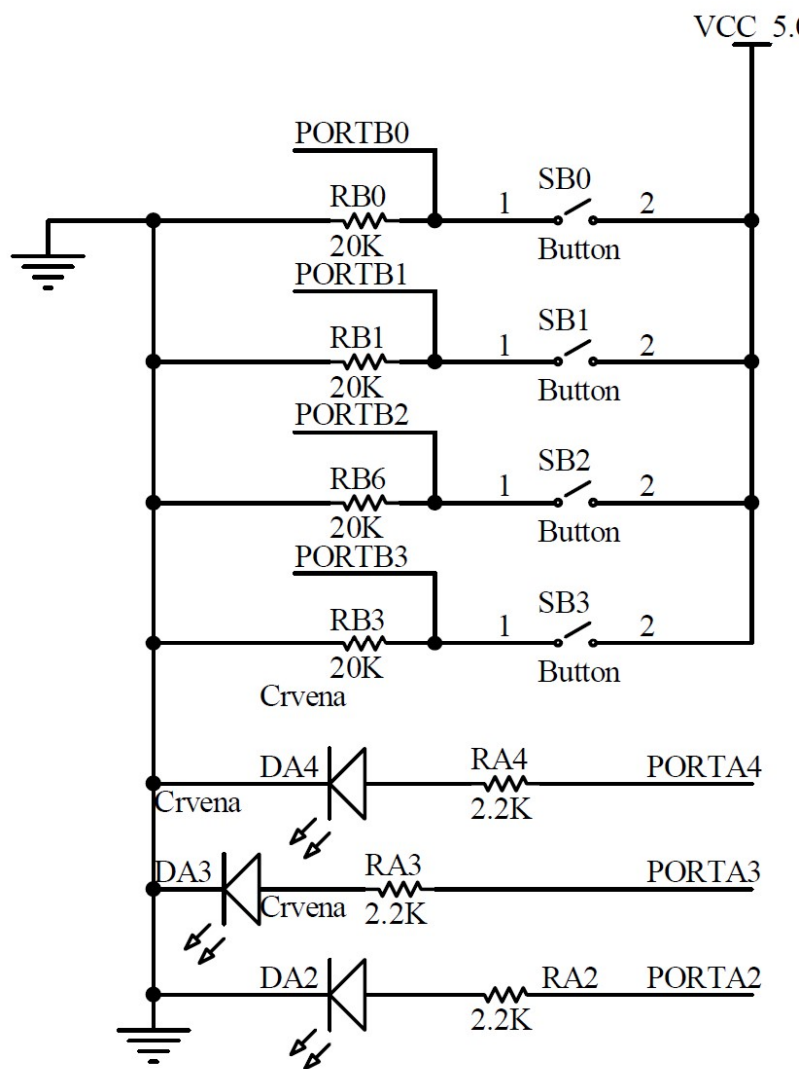
TRISD=0b00001111;

//ili ovako TRISD=0x0F

if (PORTD.F3 == 1) ...

// da li je prekidač PORTD3 na 1





`PORTA.F4 = 1; // pali LED diodu DA4`
`PORTA.F4 = 0; // gasi`

Master ploča

pinovi PORTB0 – PORTB3

su dig. ulazi (tasteri)

`TRISB=0b00001111;`

`// ili ovako TRISB=0x0F`

`if (PORTB.F3== 1) ...`

`// da li je taster PORTB3 pritisnut`

pinovi PORTA2 – PORTA3

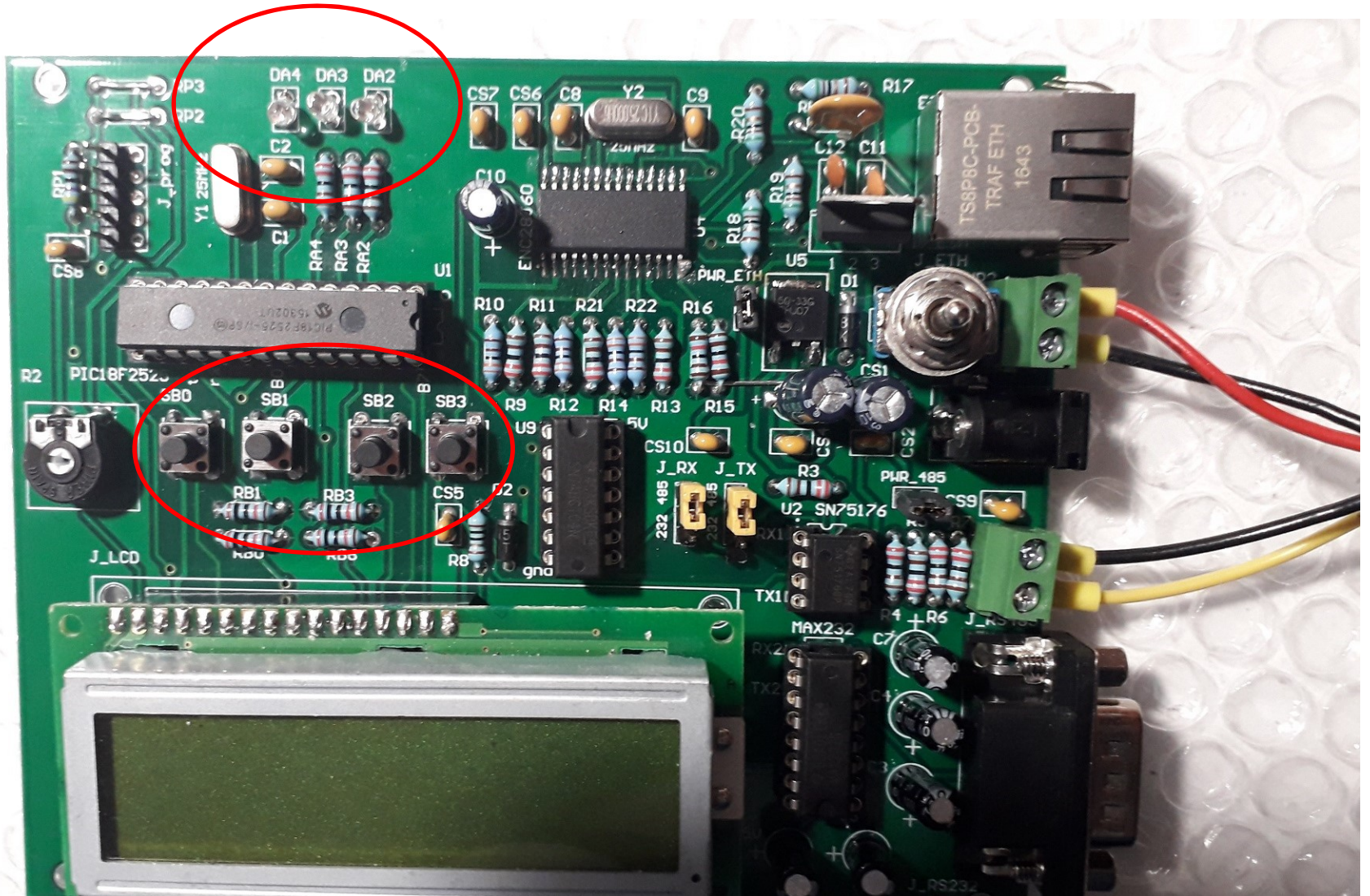
su digitalni izlazi (LED diode)

`TRISA=0b00000000`

`// ili ovako TRISA=0x00`

`PORTA=0b00000000`

`// LED diode su ugašene`



Podešavanje tajmera mikrokontrolera PIC16F877A

- Tajmer 1 može da radi u jednom u dva moda rada kao :
 - tajmer
 - brojač
- Moda rada može da se odredi preko bita, TMR1CS koji se nalazi u okviru registra T1CON.
- U Tajmer 1 je ugrađen 16-bitni registarski par (TMR1H:TMR1L) koji inkrementira stanje u opsegu od 0x0000 do 0xFFFF.
- Nakon krajnje vrednosti 0xFFFF tajmer počinje ponovo da broji od 0x0000.

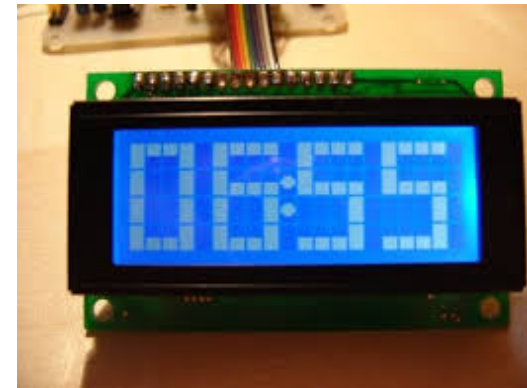
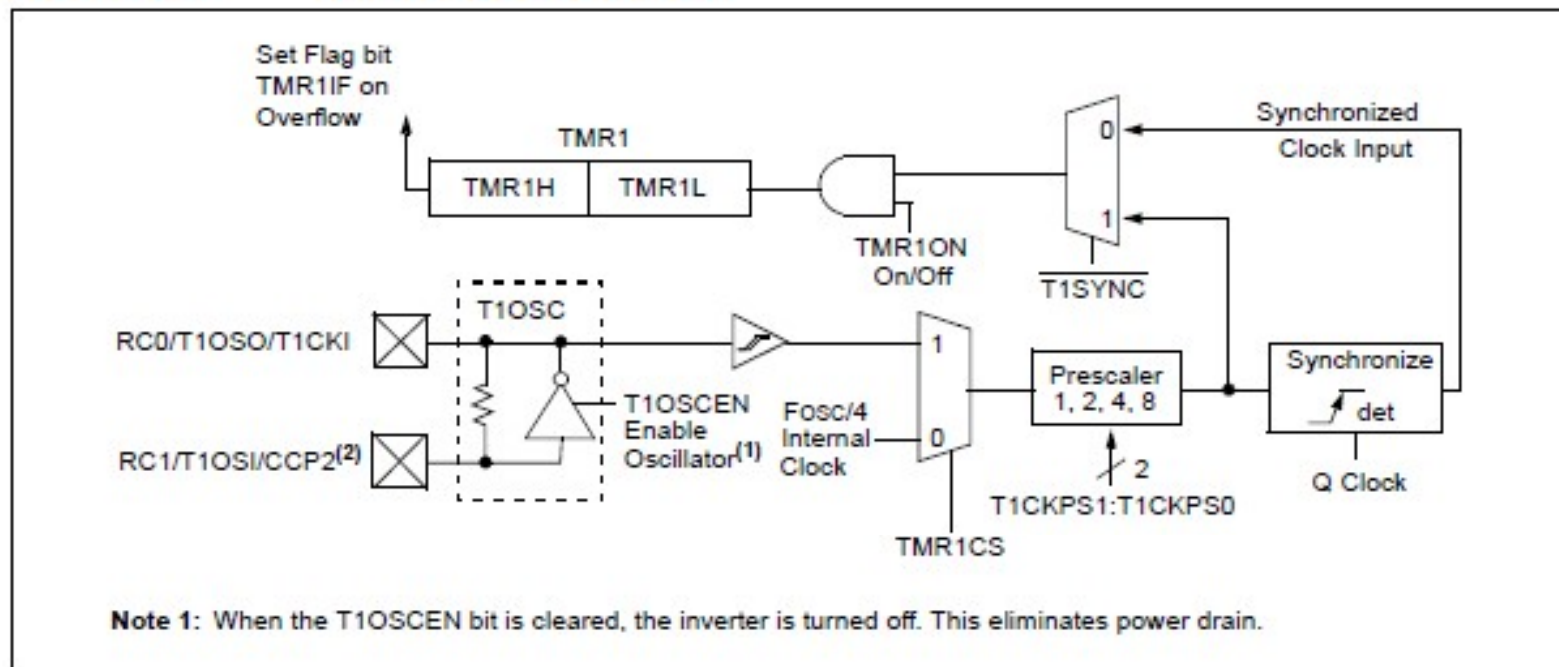


FIGURE 6-2: TIMER1 BLOCK DIAGRAM



- U primeru je pokazano kako treba podesiti rad tajmera T1 da bi izmereni vremenski interval bio jednak 100 ms.
- Za podešavanje tajmera T1, koriste se 8-bitni specijalni registri mikrokontrolera T1CON, TMR1H i TMR1L

T1CON: TIMER1 CONTROL REGISTER (ADDRESS 10h)

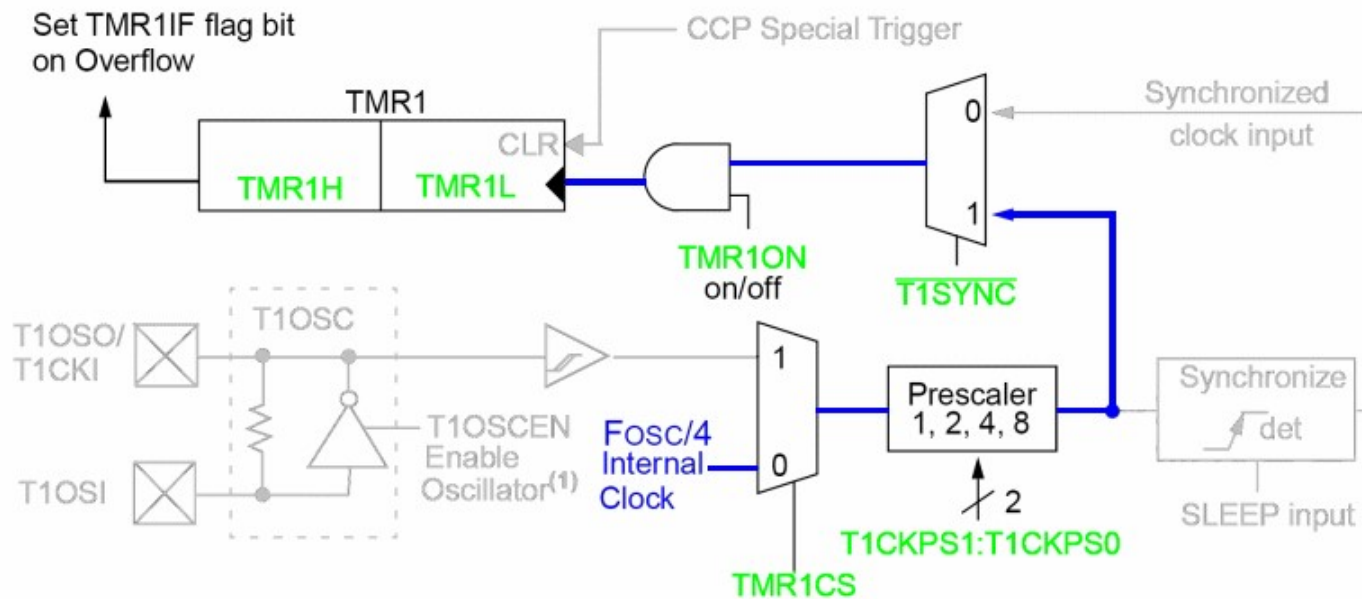
U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON
bit 7							bit 0

T1CKPS1:T1CKPS0: Timer1

11 = 1:8 prescale value
10 = 1:4 prescale value
01 = 1:2 prescale value
00 = 1:1 prescale value

- U primeru je pokazano kako treba podesiti rad tajmera T1 da bi izmereni vremenski interval bio jednak 100 ms.
- Podešavanje se radi preko registra T1CON - konfiguracionog registra tajmera T1.
- Bitovi T1CKPS1 i T1CKPS0 se setuju da bi se odabrala vrednost preskalera (delitelja frekvencije) 1:8. Bit T1OSCEN=0, jer ne koristimo oscilator tajmera T1, bit TMR1CS=0, cime se selektuje signal $F_{osc}/4$ kao ulaz preskalera.

T1CON=0b00110000;



Note 1: When the T1OSCEN bit is cleared, the inverter and feedback resistor are turned off. This eliminates power drain.

- Kako je frekvencija takta mikrokontrolera $F_{osc}=20$ MHz, na izlazu preskalera je signal frekvencije $(F_{osc}/4)/8=625$ kHz, kojim se taktuje 16 bitni brojač tajmera T1, U intervalu od 100 ms, staje $0.1 \text{ s} \cdot 625 \text{ kHz} = 62500$ ciklusa,
- Brojač tajmera kreće da broji od $(10000)_{16} - (62500)_{10} = (10000)_{16} - (F424)_{16} = (0BDC)_{16}$

// primer podesavanja tajmera za Slejv ploču

T1CON = 0b00110000; // konfiguracija za Tajmer 1

// preskaler 1/8

TMR1H = 0x0B; // startne vrednosti tajmera 1

TMR1L = 0xDC;

T1CON.TMR1ON = 1;

// Fosc=20MHz, Tosc= 50ns

// takt frekv. Fosc/4 dolazi na preskaler koji je podesen
na 1/8

// izlaz preskalera vodi se na brojace TMR1H_TMR1L

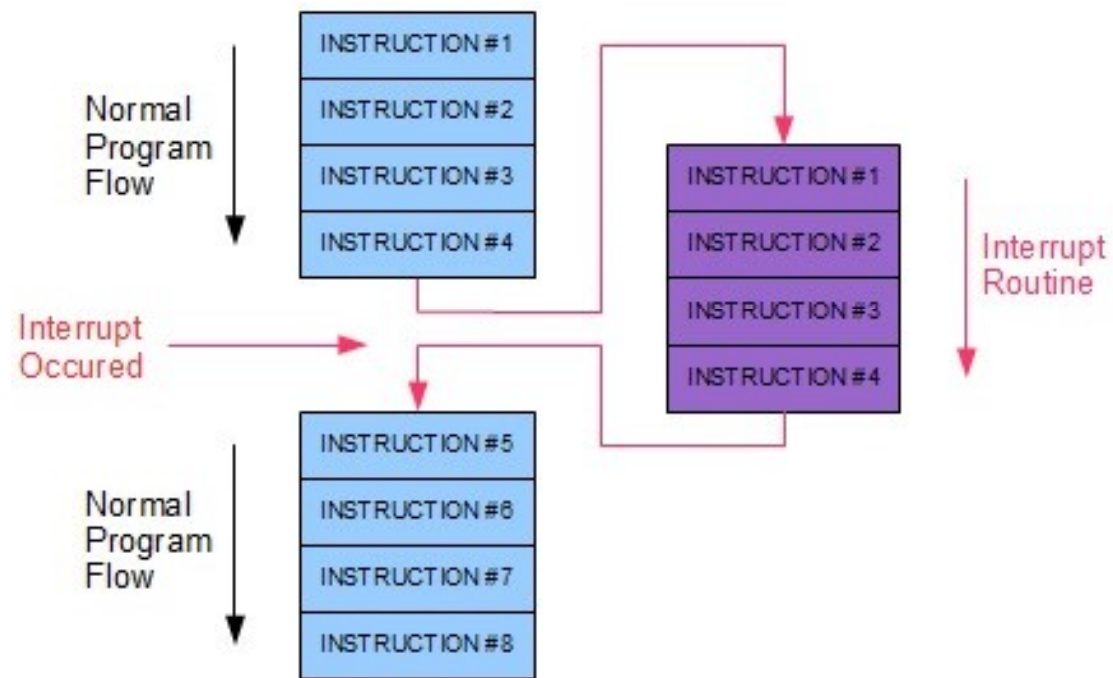
// (10000)h- (0BDC)h= (F424)h= (62500)dec

// 62500 x 8 x 4 Tosc= 100ms

// primer podesavanja tajmera za Master ploču

```
T1CON = 0b10110000;  
// konfiguracija za tajmer1  
T1CON.TMR1ON = 1;  
// 16-bit operation  
// preskaler 1:8  
// 25MH T0=40ns  
//  $40\text{ns} * 4 * 8 = 1.28\text{us}$   
//  $25\text{ms} = 25000\text{us} = 1.28 * 19531 = \text{B5B3}$   
TMR1L = 0xB5;  
TMR1H = 0xB3;
```

Podešavanje prekida mikrokontrolera PIC16F877A



- PIC mikrokontroleri sadrže softverske i hardverske prekide (interrupts).
- Ako su generisani od drugih uređaja a mikrokontroler ih dobija preko svojih ulaznih pinova, ili su pak dobijeni radom Tajmera onda se ti prekidi nazivaju hardverskim.
- Softverski prekidi su generisani softverksim kodom.
- PIC 16F877A koristi sledeće izvore prekida:
- Eksterni, Tajmer 0, Tajmer 1, Promena na portu RB, A/D konvertor, USART prijemik, USART predajnik, sinhroni serijski port, Tajmer 2, itd., kolo komparatora, itd.
- 5 registra se koriste za podešavanje prekida PIC 16F877A (PIE1, PIR1, PIE2, PIR2, INTCON)

INTCON

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF
bit 7							bit 0

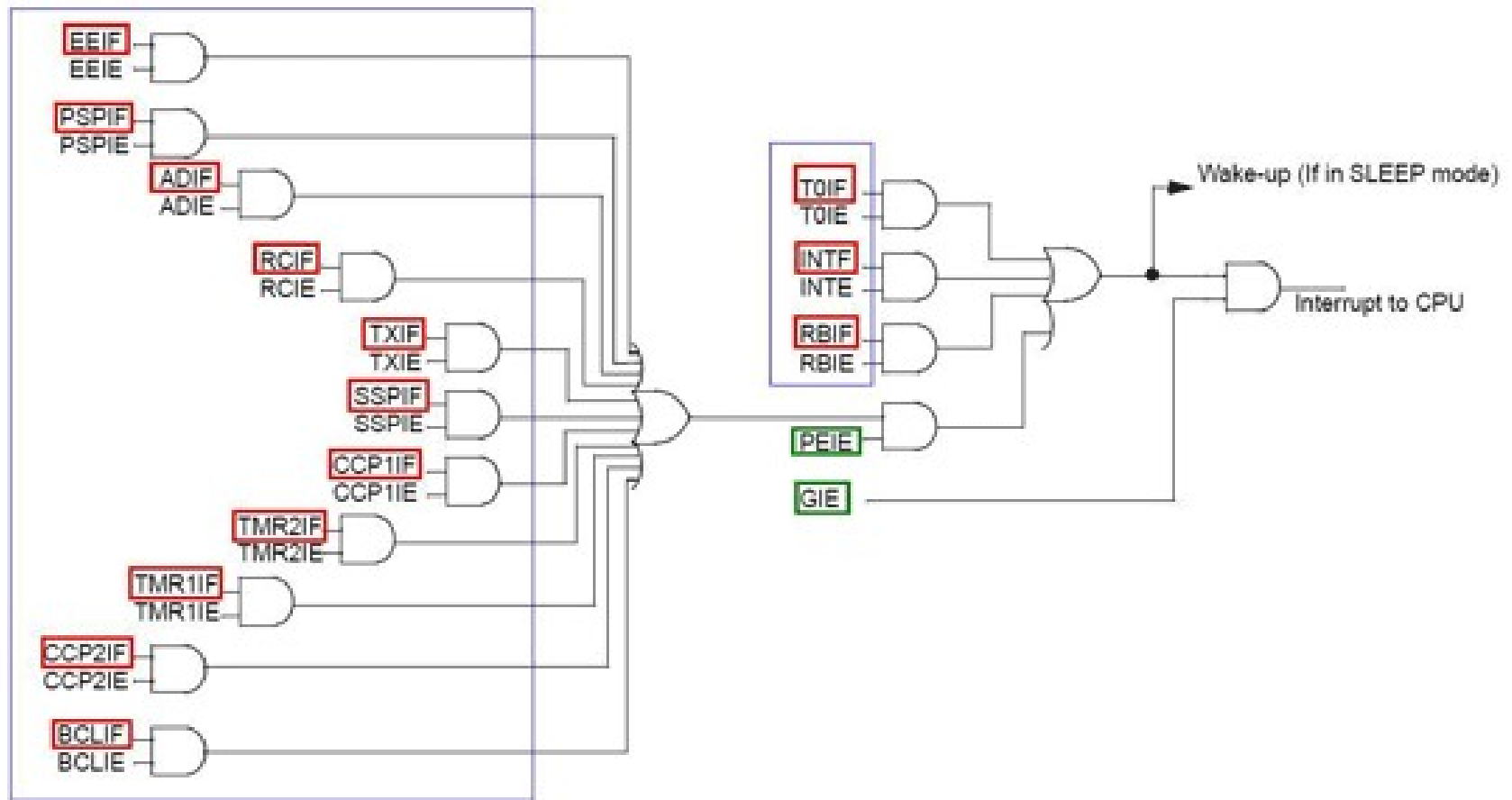
- INTCON je specijalni registar PIC16F877A koji sadrži više *Enable* i *Flag* bitova za podešavanje eksternih i internih prekida.
- GIE (7) – Global Interrupt Enable, 1 – dozvola svih prekida, 0 – zabrana prekida
- PEIE (6) – Peripheral Interrupt Enable, 1- dozvola perifernih prekida, 0 - prekidi nisu dozvoljeni
- TMR0IE (5)– Timer 0 Overflow Interrupt Enable, 1- dozvola tajmera 0, 0 – prekid tajmera 0 zabranjen
- INTE (4) – RB0/INT External Interrupt Enable, 1- dozvola eksternog prekida 0 dozvoljen , 0 – prekid zabranjen

INTCON

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF
bit 7							bit 0

- RBIE (3) – port RB Port Change Interrupt Enable
- TMR0IF (2) – Timer 0 Overflow Interrupt Flag
- INTF (1) – RB0/INT External Interrupt Flag
 - 1 – eksterni prekid se javio na pinu RB0/INT. Mora da se obriše softverski.
 - 0 – eksterni prekid na RB0/INT se nije desio
- RBIF(0) – RB Port Change Interrupt Flag
 - 1 – Najmanje jedan od pinova RB7 – RB4 promenio je stanje; nastala promena će nastaviti da setuje ovaj bit sve do se registar PORTB ne procita. Mora da se obriše softverski.
 - 0 – Nijedan od RB7 – RB4 nije promenio stanje

INTCON = 0b00000000;



PIE1 REGISTER (ADDRESS 8Ch)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PSPIE ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE
bit 7							bit 0

PIR1 REGISTER (ADDRESS 0Ch)

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
PSPIF ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF
bit 7							bit 0

- U našem primeru želimo da omogućimo prekid Tajmera 1.
- Kako se ovaj prekid nalazi u grupi perifernih prekida, podešavamo registre mikrokontrolera PIE1 i PIR1
- Prekoračenje tajmera 1 se registruje flegom TMR1IF (PIR1<0>). Ovaj prekid može da se dozvoli/ zabrani postavljanjem / resetovanjem bita TMR1IE (PIE1<0>).

```
PIE1 = 0b00000000;
```

```
PIR1.TMR1IF = 0;
```

```
PIE1.TMR1IE = 1;
```

- Deo programa za obradu prekida (deo funkcije *interrupt()*)
- U njoj treba prvo identifikovati uzrok prekida, a to se radi proverom bita TMR1IF, ako je setovan izvršava se deo za obradu tajmera 1
- Prvo se navodi naziv registra pa onda naziv bita
- U funkciji interrupt() za obradu prekida Tajmera 1, briše se fleg koji je doveo do prekida i na kraju postavljaju početne vrednosti brojača TMR1H i TMR1L

```
void interrupt() {  
    if ( (PIE1.TMR1IE==1) && (PIR1.TMR1IF==1)) {  
        PIR1.TMR1IF = 0;  
        .....  
  
        TMR1H = 0x0B;  
        TMR1L = 0xDC;  
    }  
}
```

```

void interrupt()      // primer prekida Tajmera1 za Slejv ploču
{
    if ((PIE1.TMR1IE) && (PIR1.TMR1IF))
    {
        // prekid tajmera na svakih 100ms
        PIR1.TMR1IF = 0; // brise se flag
        if (Counter == 9)
        {
            Counter = 0;
            IncrementTime();
            DecodeTime();
            UpdateLCDFlag = 1;
        }
        else
            Counter++;
        if (Counter2 > 0)
            Counter2--;
        else
            Counter2 = 0;
        ProcessInputs();
        TMR1H = 0x0B; // startne vrednosti Tajmera 1
        TMR1L = 0xDC;
    }
}

```

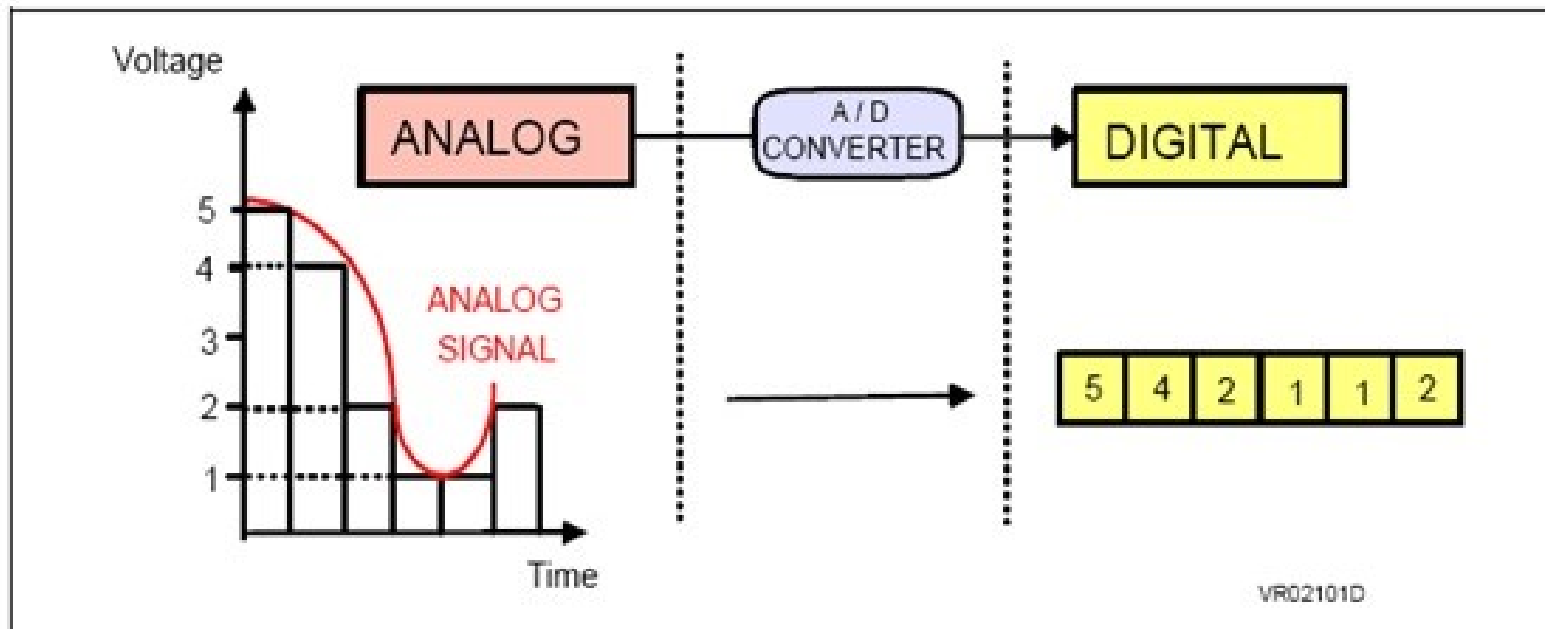
// primer prekida Tajmera1 Master ploču

```
void interrupt()
{
    if ((PIE1.TMR1IE == 1) && (PIR1.TMR1IF == 1))
    {
        // prekid tajmera 1 - na svakih 25ms
        PIR1.TMR1IF = 0;
        if (brojac == 0x04)
        { // na svakih 125ms proziva se po jedna rampa od 16
            brojac = 0x00;
            Flag1 = 0x01; // podize se flag koji nam govori da je
doslo vreme da se prozove rampa, koristimo ga u Main funkciji
        }
        else
        {
            brojac++;
        }
        TMR1L = 0xB5;
        TMR1H = 0xB3;
    }
}
```

```
// primer sprecavanja treperenja tastera PORTB.F0
```

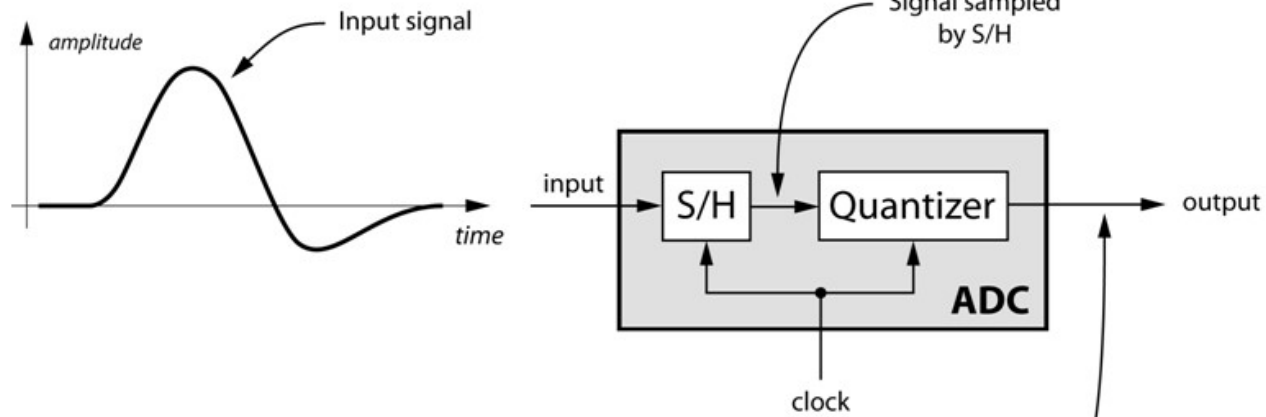
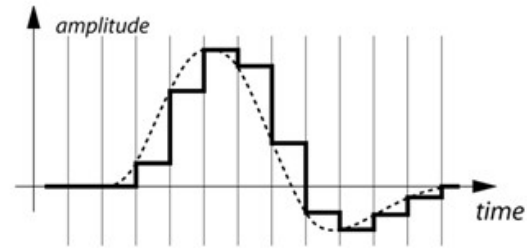
```
unsigned char cntTas=0;
bit Event;
void interrupt()
{
    if ((PIE1.TMR1IE == 1) && (PIR1.TMR1IF == 1))
    {
        // prekid tajmera 1 - na svakih 25ms
        PIR1.TMR1IF = 0;
        if (cntTas > 0) cntTas--;
        if ((PORTB.F0 == 1) && (cntTas == 0))
        {
            Event=1;
            cntTas = 10;
        }
        TMR1L = 0xB5;
        TMR1H = 0xB3;
    }
}
```


Rad sa A/D konvertorom mikrokontrolera PIC16F877A

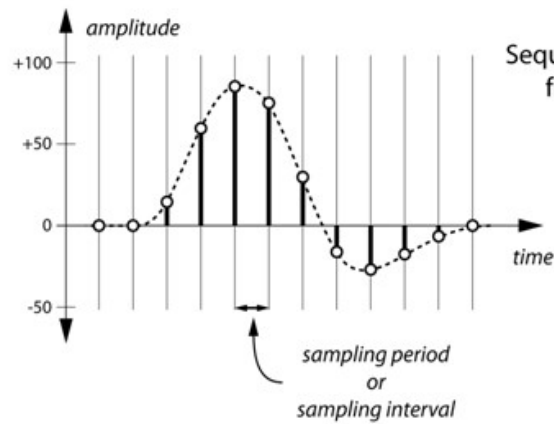


$$V_{in} = \frac{ADRESH:ADRESL}{1023} \cdot V_{REF}$$

Stage 1



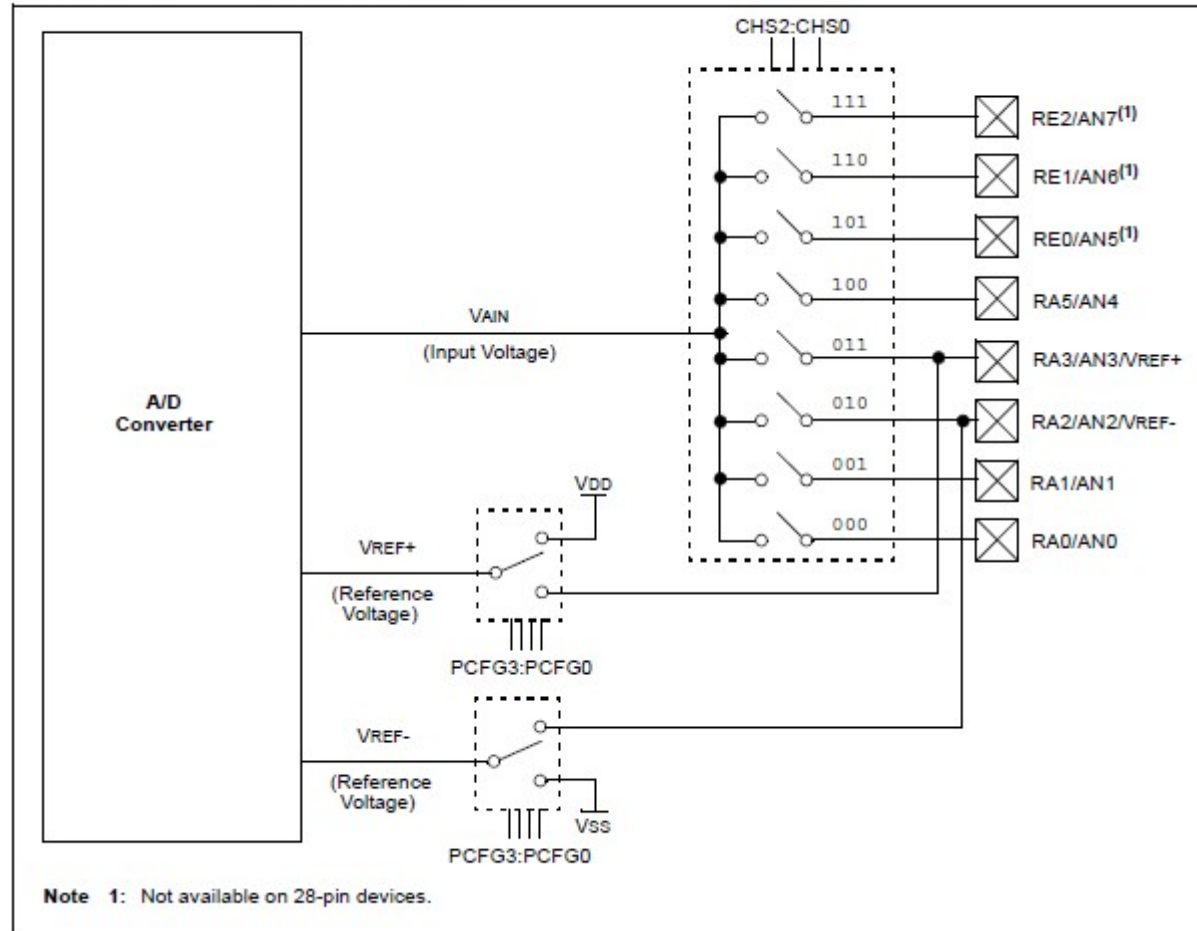
Stage 2



Sequence of numbers from quantizer

$\Rightarrow \{ \dots, 0, 0, +15, +60, +85, +72, +28, -17, -30, -21, -8, -1, \dots \}$

FIGURE 11-1: A/D BLOCK DIAGRAM



ADCON0 REGISTER (ADDRESS 1Fh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/DONE	—	ADON
bit 7						bit 0	

bit 7-6 **ADCS1:ADCS0**: A/D Conversion Clock Select bits (ADCON0 bits in bold)

ADCON1 <ADCS2>	ADCON0 <ADCS1:ADCS0>	Clock Conversion
0	00	Fosc/2
0	01	Fosc/8
0	10	Fosc/32
0	11	FRC (clock derived from the internal A/D RC oscillator)
1	00	Fosc/4
1	01	Fosc/16
1	10	Fosc/64
1	11	FRC (clock derived from the internal A/D RC oscillator)

bit 5-3 **CHS2:CHS0**: Analog Channel Select bits

- 000 = Channel 0 (AN0)
- 001 = Channel 1 (AN1)
- 010 = Channel 2 (AN2)
- 011 = Channel 3 (AN3)
- 100 = Channel 4 (AN4)
- 101 = Channel 5 (AN5)
- 110 = Channel 6 (AN6)
- 111 = Channel 7 (AN7)

ADCON0 REGISTER (ADDRESS 1Fh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/DONE	—	ADON
bit 7							bit 0

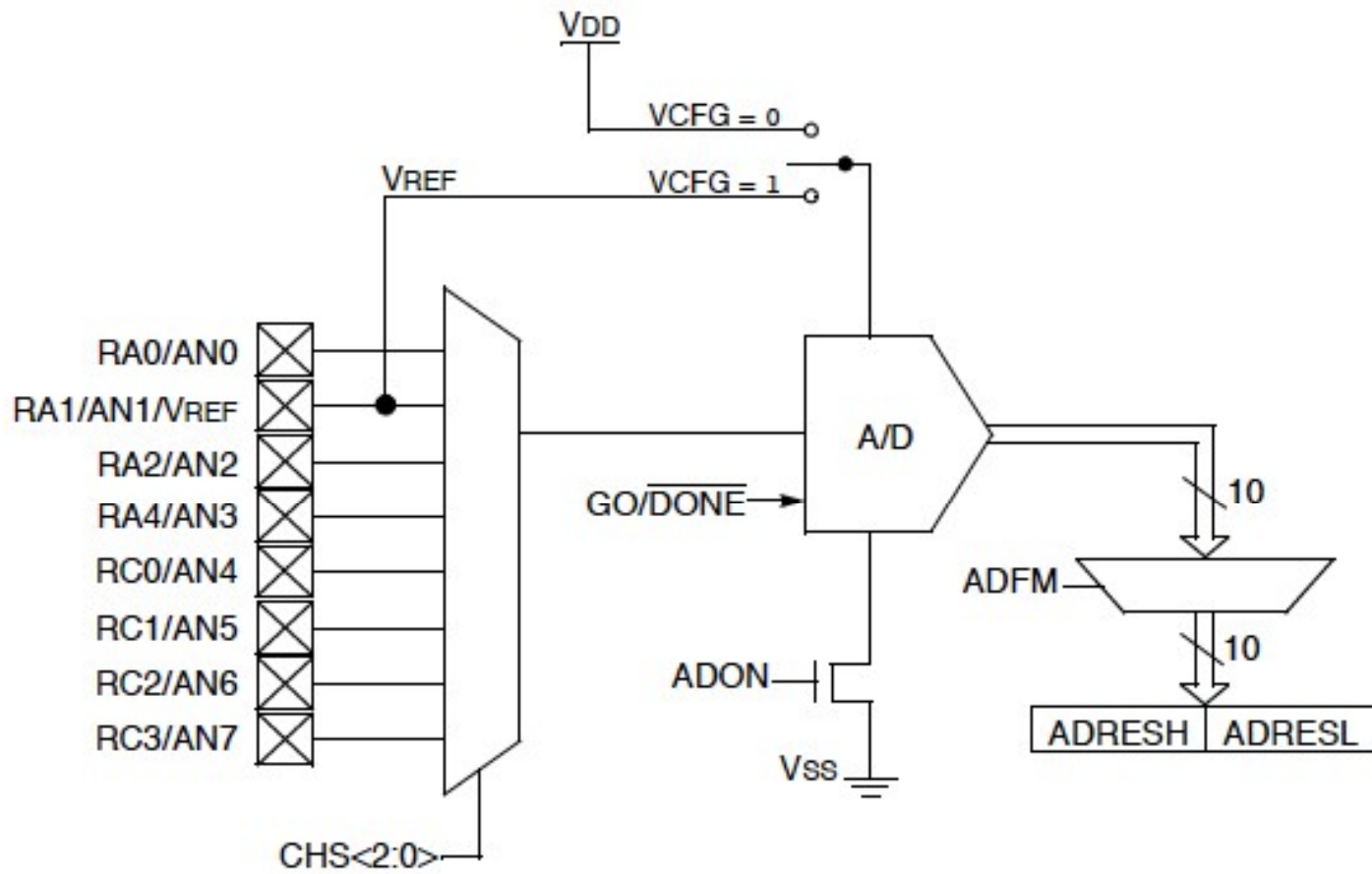
TABLE 11-1: TAD vs. MAXIMUM DEVICE OPERATING FREQUENCIES (STANDARD DEVICES (F))

AD Clock Source (TAD)		Maximum Device Frequency
Operation	ADCS2:ADCS1:ADCS0	
2 TOSC	000	1.25 MHz
4 TOSC	100	2.5 MHz
8 TOSC	001	5 MHz
16 TOSC	101	10 MHz
32 TOSC	010	20 MHz
64 TOSC	110	20 MHz
RC ^(1, 2, 3)	x11	(Note 1)

Primer:

- Ako koristimo u projektu **jedan** analogni ulaz koji se nalazi na pinu **RA0 - AN0**. Zato treba podesiti kanal 0 preko **CHS(2:0)="000"** u ADCON registru i uključiti AD konvertor **ADON=1**.
- Kako je frekvencija takta mikrokontrolera 20 MHz, može da se odabere takt AD konvertora $F_{osc}/64$, tako da su bitovi **ADCS(2:0)="110"**
- U registru ADCON0 nalaze se samo **ADCS(1:0)= "10"**, ADCS2 nalazi se u registru ADCON1.

ADCON0=0b10000001; //ADCON0=0x81;



ADCON1 REGISTER (ADDRESS 9Fh)

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	ADCS2	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

- Da li je neki pin podeseo da bude digitalni ili analogni podešava se u **ADCON1** specijalnom registru, preko bitova **PCFG3- PCFG0**
- U primeru imamo jedan analogni ulaz – RA0, pa su bitovi PCFG(3:0)="1110", videti tabelu
- Već smo ranije odredili bit **ADSC2=1**
- Bit na poziciji 7, **ADFM** kada je na jedinici, bira desno poravnanje rezultata. Tada se 10-bitni rezultat nalazi (desno poravnat) u registrima **ADRESH** i **ADRESL**. Sa **ADFM=0**, bira se levo poravnanje.
- **Važno:** Kada je neki pin analogni, mora da bude podešen i kao ulazni pin preko TRIS registra.

ADCON1=0b01001110; // ADCON1=0x4E

Kada nam ne trebaju analogni ulazi i ne radimo A/D konverziju tada imamo podešavanja:

```
ADCON0=0x00; //isključujemo A/D konverziju
```

```
ADCON1=0b00000110; //svi digitalni pinovi
```

ADCON0 REGISTER (ADDRESS 1Fh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/DONE	—	ADON
bit 7							bit 0

Kod u Mikro C-u za AD konverziju:

```
int result[8]=0x0000;
```

```
unsigned char channel=0x00;
```

```
void read_ADC() {
```

```
    ADCON0=ADCON0&0b11000111;
```

```
    ADCON0=ADCON0 | (channel<<3) ; //selekcija kanala
```

```
    for (int i=0; i<30; i++); // wait
```

```
    ADCON0.GO_DONE=1; // pocetak konverzije
```

```
    while (ADCON0.GO_DONE==1); //cekaj dok se ne zavrshi
```

```
    result[channel]=ADRESH<<8+ADRESL; // procitaj rezultat
```

```
}
```

PIE1.ADIE=1;

```
void interrupt()
{ // prekid tajmera 1
  if ((PIE1.TMR1IE == 1) && (PIR1.TMR1IF == 1))
  {
    // prekid tajmera 1 - na svakih 25ms
    PIR1.TMR1IF = 0;
ADCON0.GO_DONE=1;
// startuje se AD konverzija
    TMR1L = 0xB5;
    TMR1H = 0xB3;
  }

  if ((PIR1.ADIF) && (PIE1.ADIE)) { // prekid AD konvertora
PIR1.ADIF=0;
    result[channel] = (ADRESH<<8) + ADRESL; // procitaj rezultat
    ADCON0=ADCON0&0b11000111; // priprema
    ADCON0=ADCON0|(channel<<3); // selekcija kanala
  }
}
```