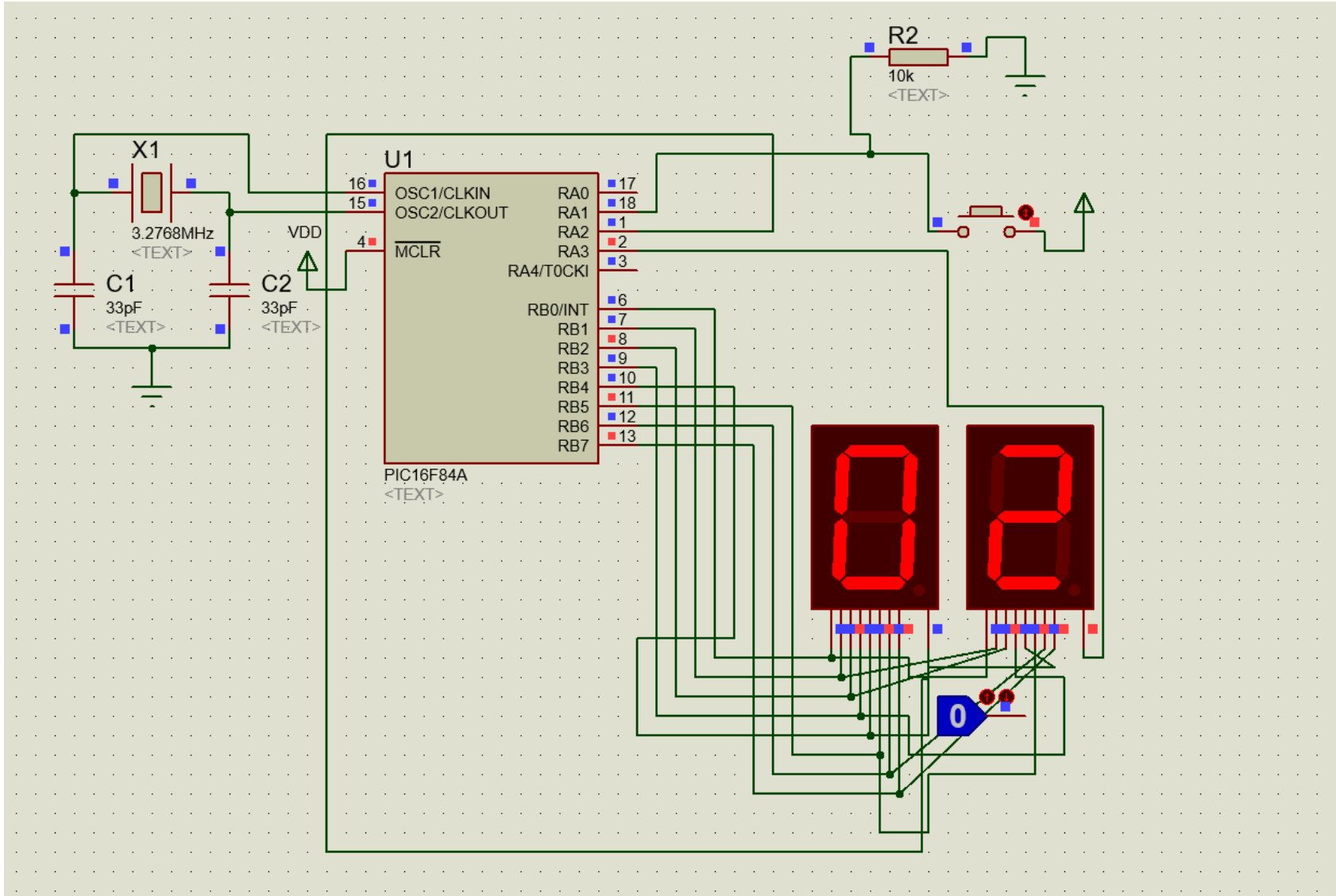
PIC16F84A – III termin

Zadatak 2 – Osvežavanje displeja

EEPROM upis/čitanje

Štelovanje preskalera

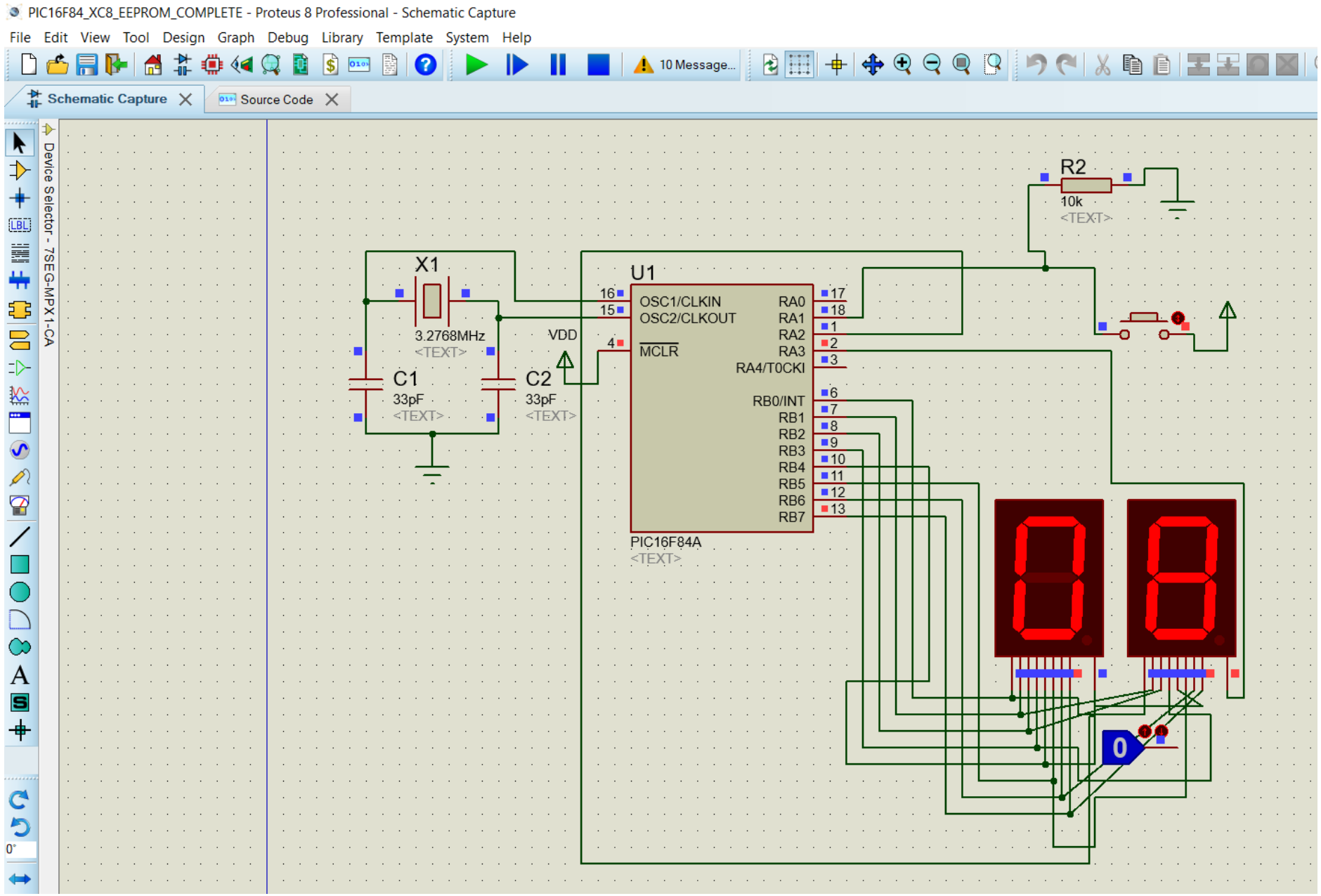
Naivni delay u asembleru

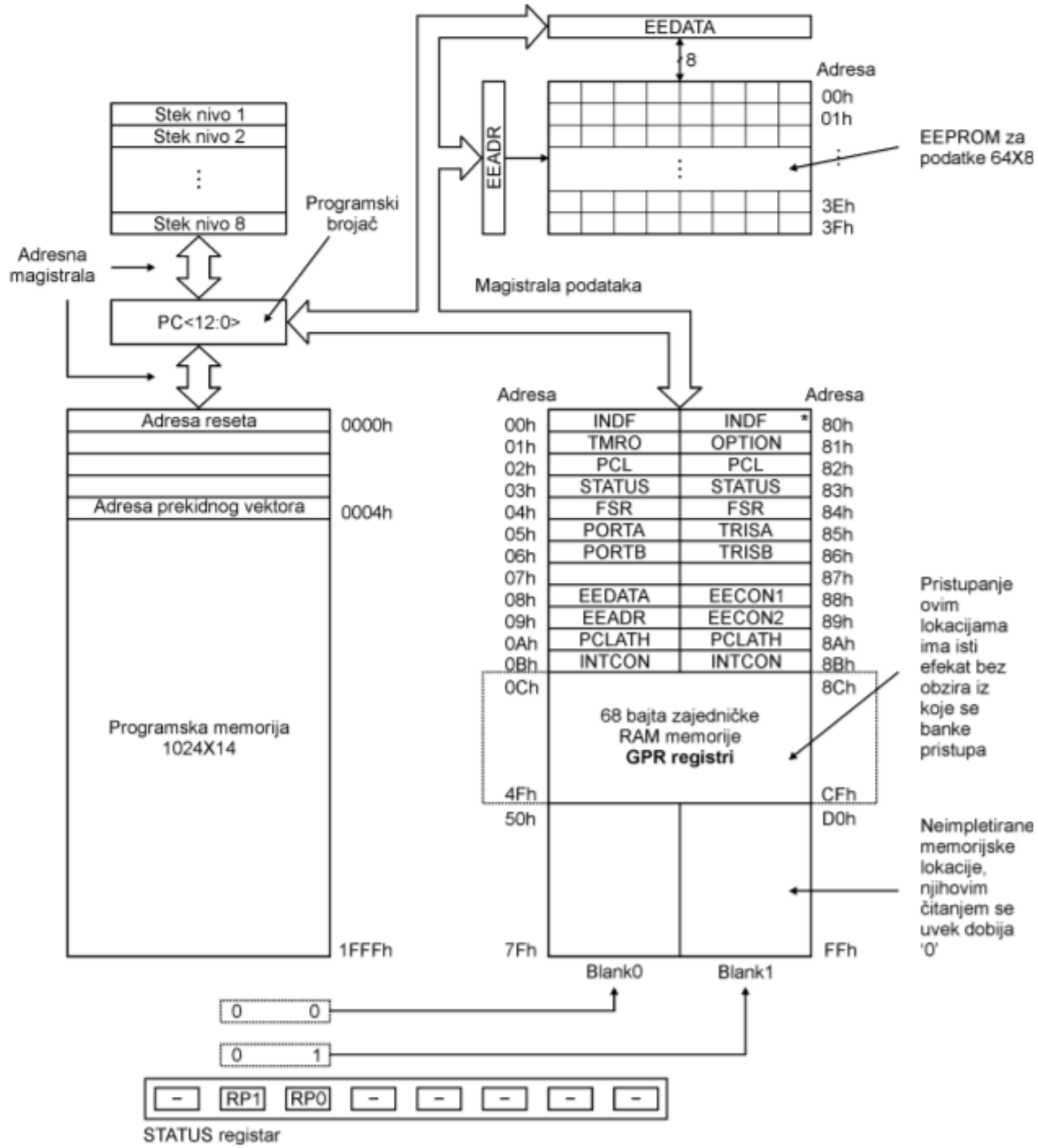


Zadatak 2

- Napisati program na asemblerskom jeziku/XC8 za PIC16F84A koji inkrementira sadržaj dva 7s displeja pritiskom na taster.
- Pin RA1 povezan je na taster a linije RB0-RB6 na segmente displeja.
- Pritiskom na taster treba inkrementirati sadrzaj prikazan na displeju.
- Početna vrednost prikazana na displeju je 00.
- 99->00
- Koristiti EEPROM memoriju za tablicu 7s cifara
- Osvežavati prikazan sadržaj uz pomoć interapta TMR0 frekvencijom 50Hz
- Takt oscilatora je 3.2768MHz

Šema


PIC16F84_XC8_EEPROM_COMPLETE - Proteus 8 Professional - Schematic Capture
File Edit View Tool Design Graph Debug Library Template System Help
10 Message...
Schematic Capture
Source Code
Device Selector - 7SEG-MPX1-CA
R2
10k
<TEXT>
X1
3.2768MHz
<TEXT>
C1
33pF
<TEXT>
C2
33pF
<TEXT>
VDD
U1
OSC1/CLKIN
OSC2/CLKOUT
MCLR
RA0
RA1
RA2
RA3
RA4/T0CKI
RB0/INT
RB1
RB2
RB3
RB4
RB5
RB6
RB7
PIC16F84A
<TEXT>
0



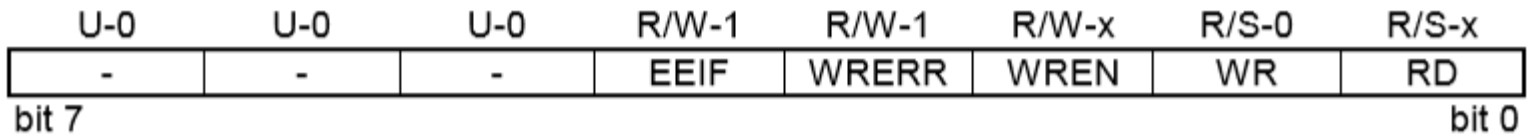
Sl. 2.15. Memorijska organizacija mikrokontrolera 16F84

EEPROM memorija

- EEPROM memorija se nalazi u posebnom memorijskom prostoru i pristupa joj se preko specijalnih registara.
- Registri za rad sa EEPROM memorijom su:
  - EEDATA
    - sadrži podatak koji je pročitan ili koga treba upisati.
  - EEADR
    - sadrži adresu EEPROM lokacije kojoj se pristupa.
  - EECON1
    - sadrži kontrolne bitove.
  - EECON2
    - ovaj registar ne postoji fizički i služi da zaštiti EEPROM od slučajnog upisa.

EECON1 registar

- EEIF
  - Interrupt flag za detekciju prekid nakon upisa u EEPROM
- WRERR
  - Greška pri upisu
- WREN
  - Omogućiti upis
- WR
  - Izvršiti upis
- RD
  - Izvršiti čitanje



Upis u EEPROM memoriju

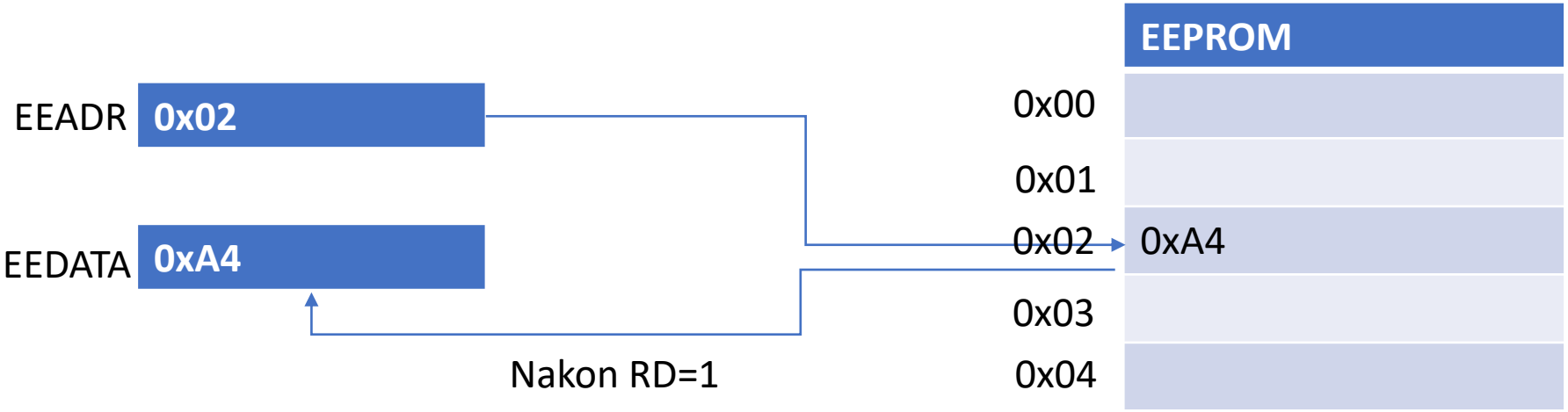
ASM/XC8 kod za upis u EEPROM

```
movlw 0xF9
banksel EEDATA
movwf EEDATA
movlw .1
movwf EEADR

banksel EECON1
bsf   EECON1,WREN
MOVLW 0x55
MOVWF EECON2
MOVLW 0xAA
MOVWF EECON2
BSF EECON1,WR
```

```
EECON1bits.WREN=1;
EEADR=0x01;
EEDATA=0xF9;
EECON2=0x55;
EECON2=0xAA;
EECON1bits.WR=1;
```

Čitanje EEPROM memorije



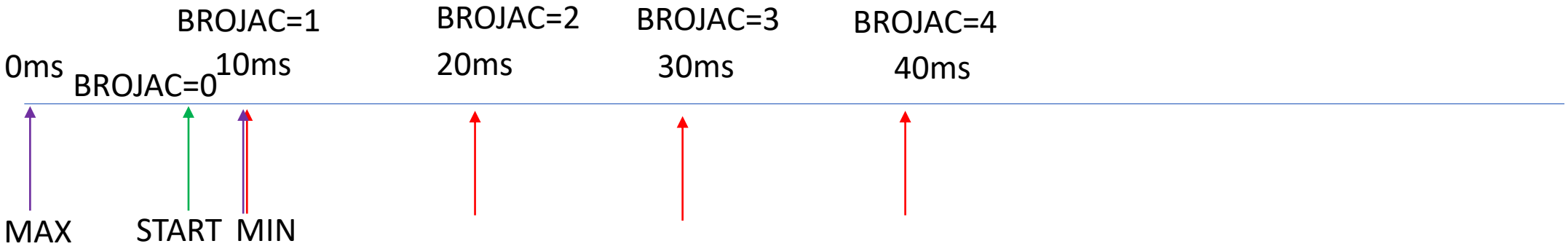
ASM/XC8 kod za čitanje iz EEPROM memorije

```
movlw .2
banksel EEADR
movwf EEADR
banksel EECON1
bsf EECON1,RD
banksel EEDATA
movf EEDATA,w
movwf CIFRA
```

```
EEADR=0x02;
EECON1bits.RD=1;
CIFRA=EEDATA;
```

Naivni delay

- Brojač inicijalno 0
- U svakom interrupt-u se inkrementira
- Broji do neke zadate vrednosti
- Primer dat za 100Hz i brojač koji ide do 4
- Kašnjenje koje daje u rasponu
  - Najmanje traje: (BROJAC-1)*(1/FREKVENCIJA PREKIDA)
  - Najduže traje: BROJAC*(1/FREKVENCIJA PREKIDA)



ASM kod

```
#include p16f84a.inc                ; Include register
definition file

;=====================================================
============
; VARIABLES
;=====================================================
============
CBLOCK      0x30
            WREG_TEMP         ;storage for WREG during
interrupt
            STATUS_TEMP       ;storage for STATUS during
interrupt
            PCLATH_TEMP       ;storage for PCLATH during
interrupt
            FSR_TEMP          ;storage for FSR during
interrupt
            DISPLAY                 ; display position,
moze i BROJAC
            CONST1
            CONST2
            PRESSED
            BROJAC
            ITERATOR
ENDC
;=====================================================
============
; RESET and INTERRUPT VECTORS
;=====================================================
============

      ; Reset Vector
RST   org 0x0000
      goto  ResetCode
      org 0x0004
      goto InterruptCode
;=====================================================
============
; CODE SEGMENT
;=====================================================
============
ResetCode:
   clrf PCLATH
   goto Start
Start:

upis:

            movlw 0xC0
            banksel EEDATA
            movwf EEDATA
            banksel EEADR
            movlw .0
            movwf EEADR
            call upisicifru

              movlw 0xF9
            banksel EEDATA
            movwf EEDATA
            banksel EEADR
            movlw .1
            movwf EEADR

            ;... Slično i za ostale cifre!..

            movlw 0x90
            banksel EEDATA
            movwf EEDATA
            banksel EEADR
            movlw .9
            movwf EEADR
            call upisicifru
```

```
            banksel TRISA
            movlw       0x02
            movwf       TRISA
            clrf        TRISB

            BANKSEL OPTION_REG
            movlw 0x04
            movwf OPTION_REG

            BANKSEL PORTA
            movlw 0x00
            movwf CONST1
            movlw 0x00
            movwf CONST2

            clrf CONST1
            clrf CONST2
            clrf DISPLAY
            clrf PRESSED
            clrf BROJAC

            BANKSEL             INTCON
            movlw       0
            movwf       INTCON
            bsf         INTCON,T0IE
            bsf         INTCON,GIE
```

```
opet:

            btfss PORTA,1
            goto otpusti
            clrf BROJAC
            call delay
            btfss PORTA,1
            goto otpusti

            btfsc PRESSED,0
            goto nastavi

            bsf PRESSED,0
            incf CONST1, F
            movlw .10
            subwf CONST1, W
            btfss STATUS, Z
            goto nastavi

            clrf CONST1

            incf CONST2, F
            movlw .10
            subwf CONST2, W
            btfss STATUS, Z
            goto nastavi
            clrf CONST2
            goto nastavi
otpusti:
            clrf BROJAC
            call delay
            btfss PORTA,1
            bcf PRESSED,0

nastavi:
            goto  opet
```

Nastavak

```
InterruptCode:
            movwf WREG_TEMP   ;save WREG
            swapf STATUS,W    ;store STATUS in WREG
            clrf  STATUS            ;select file register bank0
            movwf STATUS_TEMP ;save STATUS value
            movf  PCLATH,W    ;store PCLATH in WREG
            movwf PCLATH_TEMP ;save PCLATH value
            clrf  PCLATH            ;select program memory page0
            movf  FSR,W       ;store FSR in WREG
            movwf FSR_TEMP    ;save FSR value

            BANKSEL           INTCON
            bcf         INTCON,T0IF

            BANKSEL PORTA

            btfsc       DISPLAY,0
            goto drugi
            bcf PORTA,2
            bsf PORTA,3
            goto gotovo

            drugi:
            bcf PORTA,3
            bsf PORTA,2

            gotovo:

            movf        CONST2,W
            btfss       DISPLAY,0
            movf        CONST1,W

            call dekodiranje
            movwf       PORTB
            incf DISPLAY,F
            btfsc       DISPLAY,1
            clrf DISPLAY
            incf BROJAC,f
kraj:
;-----------------------------------
;End of interrupt routine restores context

EndInt:     bcf   3,5         ;select bank 0
            movf  FSR_TEMP,W  ;get saved FSR value
            movwf FSR         ;restore FSR
            movf  PCLATH_TEMP,W     ;get saved PCLATH value
            movwf PCLATH            ;restore PCLATH
            swapf STATUS_TEMP,W     ;get saved STATUS value
            movwf STATUS            ;restore STATUS
            swapf WREG_TEMP,F ;prepare WREG to be restored
            swapf WREG_TEMP,W ;restore WREG without affecting STATUS
            retfie                  ;return from interrupt
```

```
dekodiranje:
      banksel EEADR
      movwf EEADR
      banksel EECON1
      bsf EECON1,RD
      banksel EEDATA
      movf EEDATA,w
      return
delay:
      btfss BROJAC,2
      goto delay
return

upisicifru:

      banksel     EECON1
      bsf   EECON1,WREN ;enable EEPROM write
operations
      MOVLW 0x55
      banksel EECON2
      MOVWF EECON2
      MOVLW 0xAA
      MOVWF EECON2
      banksel EECON1
      BSF EECON1,WR

return

End
```

XC8 kod

```
#include<htc.h>
// Config word
__CONFIG(FOSC_HS & WDTE_OFF & PWRTE_ON & CP_OFF);

//CPU takt
//Mora da se definise ako se koristi  __delay_ms()
#define _XTAL_FREQ   3276800

int index=0;
int c1=0;
int c2=0;
int pressed=0;
int counter=0;

void upisi();
void prikazi(int cifra);

void interrupt intcode();

// Main function
void main()
{
   TRISB=0x00;
   TRISA=0x00;
   TRISAbits.TRISA1=1;

   upisi();

   PORTA=0x00;
   PORTB=0x00;

   PORTB=EEDATA;

   OPTION_REG=0x04;

   INTCON=0xA0;
   INTCONbits.T0IF=0;

   c1=0;
   c2=0;

   while(1)
   {
         if(PORTAbits.RA1==1)
         {
            if(pressed==0){
             pressed=1;
             c1+=1;
             if(c1>9){
                c2++;
                c1=0;
                if(c2>9)
                   c2=0;
             }

            }
         } else
         {
            pressed=0;
         }
      }
}
```

```
void upisi()
{
    int i=0;
    static const int codes[]={0xC0, 0xF9, 0xA4, 0xB0, 0x99,
0x92, 0x82, 0xd8, 0x80, 0x90};
    for(i=0;i<10;i++)
    {
      EECON1bits.WREN=1;
      EEADR=i;
      EEDATA=codes[i];
      EECON2=0x55;
      EECON2=0xAA;
      EECON1bits.WR=1;
      __delay_ms(10);
    }
}

void prikazi(int cifra)
{
   EEADR=cifra;
   EECON1bits.RD=1;
   PORTB=EEDATA;
}

void interrupt intcode(){
 if(INTCONbits.T0IF==1)
 {
   if(index==0){
   PORTAbits.RA3=1;
   PORTAbits.RA2=0;
   prikazi(c1);
   index=1;
}else
{
 PORTAbits.RA2=1;
 PORTAbits.RA3=0;
 prikazi(c2);

 index=0;
}
INTCONbits.T0IF=0;
}
}
```

Štelovanje preskalera

- $\frac{2^{15}*100Hz}{2^2*2^{n+1}*2^8} = 2*50Hz$
- $\frac{2^{15}}{2^{n+11}} = 1$
- n=4
- Zatim, podešavamo OPTION registar:

movlw 0x04

movwf OPTION_REG

Štelovanje preskalera u općem slučaju

- $\frac{Fosc}{2^2 * 2^{n+1} * X} = željena\ frekvencija$
- n – PS2 PS1 PS0 preskaler bitovi
- X – broj otkucaja TMR0
- TMR0 početno:=256-X

Primer za štelovanje prescaler-a

- $\frac{2^{15}*100Hz}{2^2*2^{n+1}*2^8} = 6*50Hz$
- $\frac{2^{15}}{2^{n+11}} = 3$
- $2^{4-n} = 3?$
- Bolje u ovom slučaju da fiksiramo n, a tražimo X
- Uzmimo n=7
- $\frac{2^{15}*100Hz}{2^2*2^{n+1}*X} = 6*50Hz$
- $\frac{2^{15}}{2^2*2^8*X} = 3$
- $\frac{2^5}{X} = 3$
- $32 = 3*\mathrm{X}$
- X=10.67
- TMR0:=256-11=245