ARHITEKTURA UPRAVLJAČKIH MIKRORAČUNALA I NJIHOVO POVEZIVANJE S OKOLINOM (III)

(Korištenje dozvoljeno samo u okviru predmeta RAČUNALA I PROCESI)

Mario Žagar

Fakultet elektrotehnike i računarstva
Zagreb 1994.

SADRŽAJ

20.3 PRIMJERI POVEZIVANJA MIKRORAČUNALA I OKOLINE

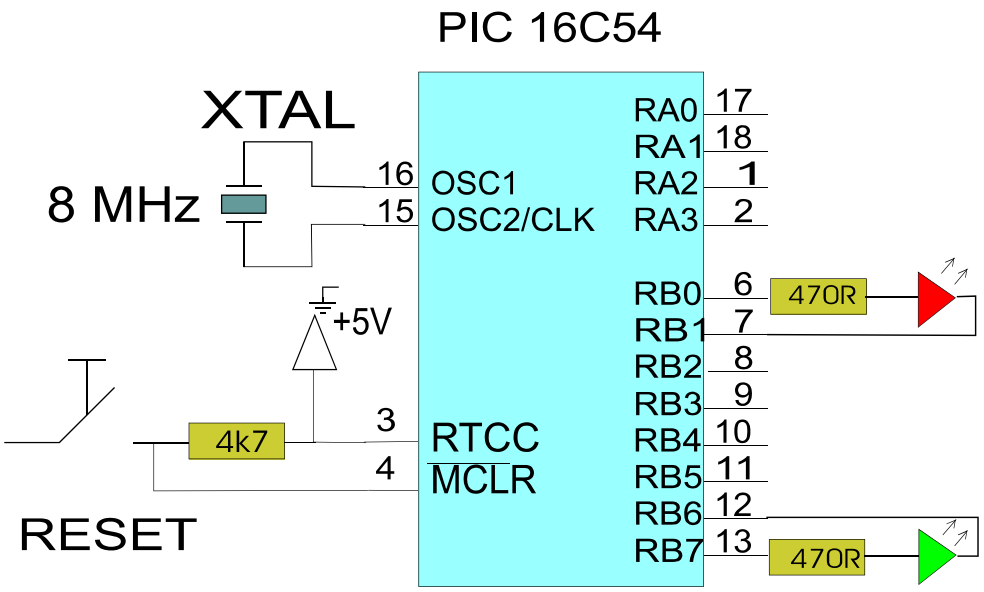
20.3.1 UVOD

U prethodnim poglavljima prikazane su osnovne arhitekture jednočipnih mikroračunala te načini povezivanja mikroračunala s okolinom. U ovom poglavlju na konkretnim primjerima pokazan je postupak integracije sklopovskog dijela mikroračunala, okoline kojoj je namijenjen te programa koji sve zajedno objedinjuje u suvislu cjelinu. Zbog lakšeg razumijevanja izabran je mikrokontroler PIC16C54 (16C57) koji je detaljnije opisan u prethodnom tekstu

20.3.2 Primjer I: Mjerenje vremena korištenjem RTCC sklopa u mikrokontroleru PIC16C54

Zadatak:
Svake sekunde potrebno je naizmjenično paliti i gasiti crvenu i zelenu LED diodu (simulacija rada svjetionika).

Sklopovsko rješenje:
Koristit ćemo mikrokontroler PIC16C54 čiji je sastavni dio i RTCC sklop za mjerenje vremena. RTCC je već opisan u tekstu, a sam rad sklopa prikazan je na slici 1.18. Budući da je za simulaciju dovoljno paliti i gasiti LED diode, izlazne jedinice bit će jednostavne. Izlazni vodovi ovog mikrokontrolera mogu davati struju do 20 mA pa se LED diode mogu priključiti direktno. Cijeli uređaj je jednostavan, troši malo energije i može se napajati baterijski. Shema sklopovskog dijela rješenja prikazana je na slici 3.1.

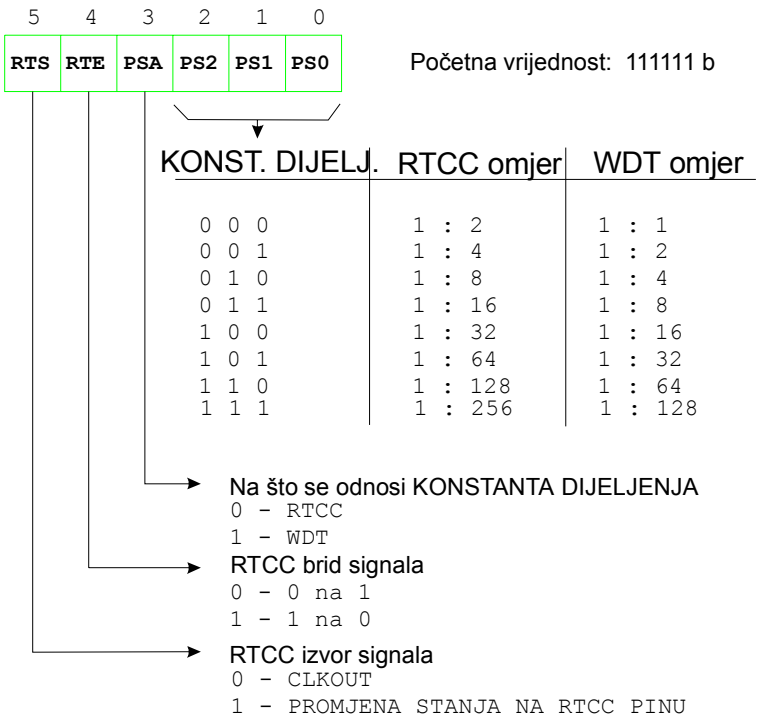


Slika 3.1: Mikrokontroler PIC 16C54 (priključak LED dioda)

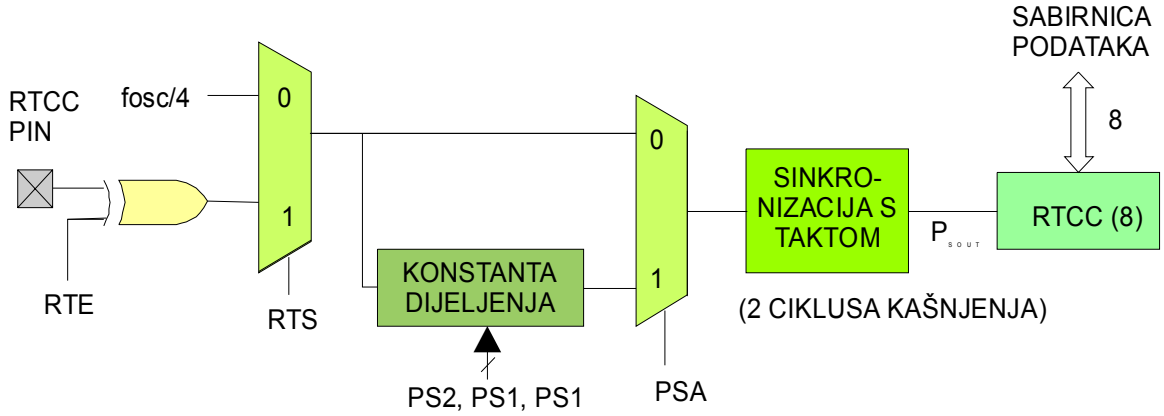
Programsko rješenje:

Programski dio rješenja uglavnom se sastoji iz inicijalizacije izlaznih vodova na koje su priključene LED diode te inicijalizacije RTCC sklopa za mjerenje vremena. Uz frekvenciju ulaznog oscilatora od 8 MHz koji se unutar sklopa dijeli s četiri, potrebno je u registar konstante dijeljenja upisati 128 te u brojač RTCC-a 125. Time se u brojaču RTCC sklopa pojavljuje NULA svakih 1/125 sekunde. Realizacijom programske petlje koja se vrti 125 puta (registar f8 služi kao brojač), dobivamo vrijednost 1 s. Stvaran posao u programu je obnoviti brojač f8 prije povratka u sljedeću petlju te zamijeniti 0 i 1 u registru f6 (B port). LED diode će se naizmjenično paliti i gasiti svake sekunde.

```
        ////////////////////////////////////////////////////////
        //    picelmar.a         M.Zagar Ver. 1.0    17.9.1993      //
        //                                                          //
        //    Mjerenje vremena pomocu RTCC-a                        //
        //    (Ulazna frekv. dijeli se s 4, pa sa 128 pa sa         //
        //    125 u RTCC sklopu te programski jos sa 125)           //
        //    (Uz ulaz 8MHZ /4*128*125*125, izlaz je 1 sekunda)     //
        ////////////////////////////////////////////////////////
           `BASE D

000  C00     MOVLW   %B 00000000 // postavljanje izlaza (B port)
001  006     TRIS    6
002  CAA     MOVLW   %B 10101010 // B7...B0 = 10101010
003  026     MOVWF   6

004  C7D     MOVLW  125          // priprema brojaca
005  028     MOVWF  8            // f8=125

006  C82     MOVLW  130          //stavi 130 (255-125) u RTCC registar
007  021     MOVWF  1

008  C06     MOVLW  6            // dijeli ulaz s 4 pa zatim sa 128 (6)
009  002     OPTION              // u RTCC prescalar (broji prema gore)

00A  201  PETLJA    MOVF   1,0    // RTCC u W reg.
00B  743     BTFSS  3,2          // test bit 2(ZERO) stat reg (3)
00C  A0A     GOTO   PETLJA

00D  C82     MOVLW  130          // RTCC izmjerio vrijeme, obnovi reg.
00E  021     MOVWF  1

00F  2E8     DECFSZ 8,1          // programsko kasnjenje (nakon 125
010  A0A     GOTO   PETLJA       // sklopovskih impulsa promijeni izlaz)

011  C7D     MOVLW  125          // prosao zadani interval, obnovi reg.
012  028     MOVWF  8

013  266     COMF   6,1          // komplement i spremi u isti reg. (6=B)
014  A0A     GOTO   PETLJA
        ////////////////////////////////////////////////////////
```



Slika 1.18: OPTION registar



1. Bitovi RTE, RTS, PS2, PS1, PS0 nalaze se u OPTION registru.
2. Konstantu dijeljenja koristi i sklop za kontrolu rada (WDT).

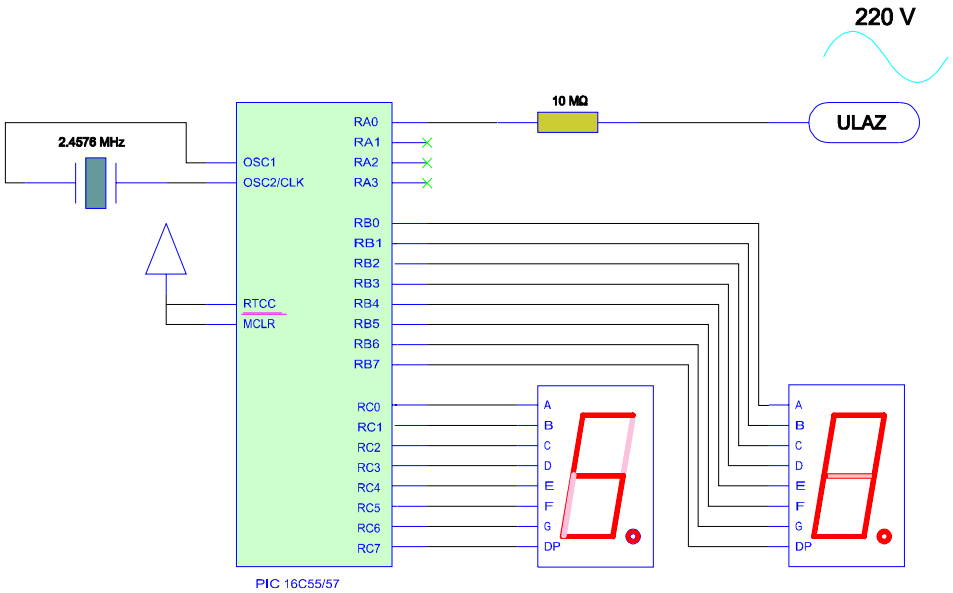
Slika 1.19: Funkcije RTCC registra
(mjerenje vremena)

Diskusija:
Sklopovsko rješenje sastoji se iz jednočipnog mikrokontrolera PIC16C54, kristala frekvencije 8 MHz, nekoliko otpornika i dvije LED diode. Glavni potrošač su LED diode pa je za rad uređaja dovoljno baterijsko napajanje. Programsko rješenje zauzima 14 memorijskih mjesta (od 512 raspoloživih), a uz minimalne promjene u programu mogu se mjeriti različiti vremenski intervali i prema potrebi aktivirati digitalni izlazi. Primjer ukazuje na moguće primjene u svjetionicima, semaforima, vremenskim automatima, programatorima npr. u strojevima za pranje suđa, rublja, u sportu i dr.

20.3.3 Primjer II: Mjerenje frekvencije električne mreže

Zadatak:
Potrebno je mjeriti frekvenciju gradske mreže priključkom na kućnu instalaciju i prikazivati vrijednost na dva 7-segmentna pokaznika.

Sklopovsko rješenje:
Sklopovsko rješenje prikazano je na slici 3.2. Da bi se napon gradske mreže (220V~) prilagodio mjernim potrebama, dovoljno je otpornikom smanjiti ulazni napon na ulaznom vodu mikrokontrolera PIC16C54. Kontroler je prilagođen industrijskoj namjeni sa ugrađenom zaštitom od prevelikih napona. Za mjerenje frekvencije dovoljan je jedan ulazni vod na kojem će se ustanoviti jedna perioda. Uz pomoć RTCC sklopa brojit će se periode u jedinici vremena. Za prikaz se koriste dva 8-bitna izlaza na koje su spojeni 7-segmentni pokaznici. Kako model PIC16C54 ima samo 12 ulazno-izlaznih vodova (18-pinsko kućište), to je nedovoljno za ovakav pristup. Zbog toga se koristi model PIC16C57 koji ima 20 ulazno-izlaznih vodova (28-pinsko kućište). Svaki 8-bitni port (B i C) direktno upravlja prikazom jednog pokaznika.



Slika 3.2: Sklop za mjerenje frekvencije gradske mreže

Programsko rješenje:

U ovom slučaju koristi se kristal frekvencije 2.4576 MHz pa je inicijalizacija RTCC sklopa drugačija (mjeri se 1/10 sekunde). Prati se stanje na vodu 0, port A (BTFSS 5,0) i gleda promjena stanja (0 -> 1). Također se prati proteklo vrijeme i povećava broj detektiranih perioda mreže, obavlja pretvorba u BCD (binarno kodiranu decimalnu) aritmetiku i rezultat sprema u dva registra f11 (desetice) i f10 (jedinice). Nakon isteka sekunde poziva se potprogram PRI koji ispisuje sadržaj registara f11 i f10 kao dvije znamenke (frekvencija mreže) na 7-segmentnim pokaznicima.

```
          // PIC mjerac frekvencije izmjenicnog napona
          // s prikazom na dva 7-segmentna pokazivaca (PIC 16C55/57)
          // verzija 1.0      26.5.1994
          // objasnjenje registara:
          // f5 - bit 0 - ulazni napon ciju frekvenciju mjerimo
          // f6 - 7-segmentni pokazivac jedinica
          // f7 - 7-segmentni pokazivac desetica
          // f9 - brojac desetinki sekunde
          // f10 - brojac frekvencije (jedinice)
          // f11 - brojac frekvencije (desetice)

            `BASE D
000  C07    MOVLW 7           // prescaler 1:256
001  002    OPTION
002  C0A    MOVLW 10          // 10 destinki sekunde
003  029    MOVWF 9
004  06A    CLRF  10
005  06B    CLRF  11
006  C01    MOVLW 1           // inicijalizacija desetinki
007  021    MOVWF 1

008  916  NUL CALL   VRI      // provjera vremena
009  705    BTFSS 5,0
00A  A08    GOTO  NUL
00B  2AA    INCF  10,1        // brojac frekvencije
00C  C0A    MOVLW 10
00D  08A    SUBWF 10,0
00E  743    BTFSS 3,2
00F  A12    GOTO  JED
010  2AB    INCF  11,1
011  06A    CLRF  10

012  916  JED CALL   VRI      // provjera vremena
013  605    BTFSC 5,0
014  A12    GOTO  JED
015  A08    GOTO  NUL

016  201  VRI MOVF   1,0      // provjera isteka desetinke
017  743    BTFSS 3,2
018  800    RETLW 0
019  C01    MOVLW 1           // inicijalizacija desetinki
01A  021    MOVWF 1
```

```
01B  2E9    DECFSZ 9,1          // provjera isteka sekunde
01C  801    RETLW 1

01D  C0A    MOVLW 10            // ispis jedinica
01E  024    MOVWF 4
01F  929    CALL  PRI
020  026    MOVWF 6

021  2A4    INCF  4,1           // ispis desetica
022  929    CALL  PRI
023  027    MOVWF 7

024  C0A    MOVLW 10            // postavljanje pocetnih vrijednosti brojaca
025  029    MOVWF 9
026  06A    CLRF  10
027  06B    CLRF  11

028  802    RETLW 2

029  200  PRI MOVF   0,0
02A  1E2    ADDWF 2,1           // skace na odgovarajuci vektor
02B  87E    RETLW %B 01111110 // znamenka "0"
02C  830    RETLW %B 00110000 // znamenka "1"
02D  86D    RETLW %B 01101101 // znamenka "2"
02E  879    RETLW %B 01111001 // znamenka "3"
02F  833    RETLW %B 00110011 // znamenka "4"
030  85B    RETLW %B 01011011 // znamenka "5"
031  85F    RETLW %B 01011111 // znamenka "6"
032  870    RETLW %B 01110000 // znamenka "7"
033  87F    RETLW %B 01111111 // znamenka "8"
034  87B    RETLW %B 01111011 // znamenka "9"
035  877    RETLW %B 01110111 // znamenka "A"
036  81F    RETLW %B 00011111 // znamenka "B"
037  84E    RETLW %B 01001110 // znamenka "C"
038  83D    RETLW %B 00111101 // znamenka "D"
039  84F    RETLW %B 01001111 // znamenka "E"
03A  847    RETLW %B 01000111 // znamenka "F"
            `END
```

Diskusija:

Uređaj je dovoljno jednostavan da bi se mogao realizirati kao jednostavan, priručni mjerni uređaj. Može mjeriti frekvenciju od 1 do 99 Hz, a pokazuje kako se na jednostavan način može ostvariti prikazna jedinica te kako se zahvaljujući izvedbi ulazno-izlaznih vodova kod PIC mikrokontrolera jednostavno mogu povezivati i mjeriti energetske veličine direktno iz procesa (220V). Prilikom bilo kakvog eksperimentiranja s ovakvim naponom, potrebno je detaljno proučiti specifikacije proizvođača za određenu komponentu!

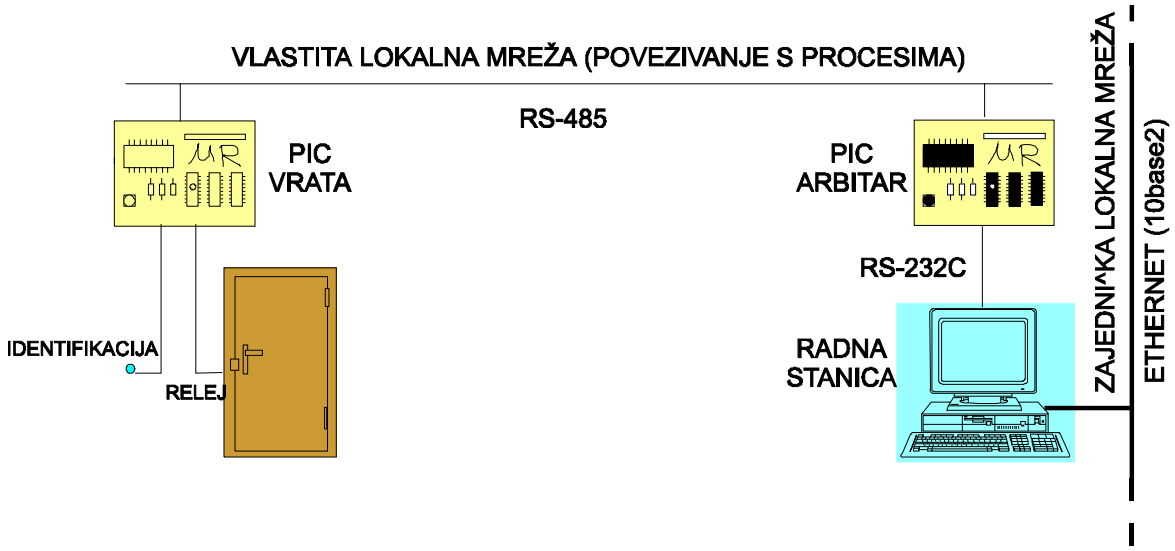
3.4 Primjer III: Identifikacija osoba i otvaranje vrata

Zadatak:

Potrebno je identificirati osobu koja stoji ispred ulaznih vrata te ovisno o dozvoli ulaska otvoriti električnu bravu na vratima i time omogućiti ulazak.

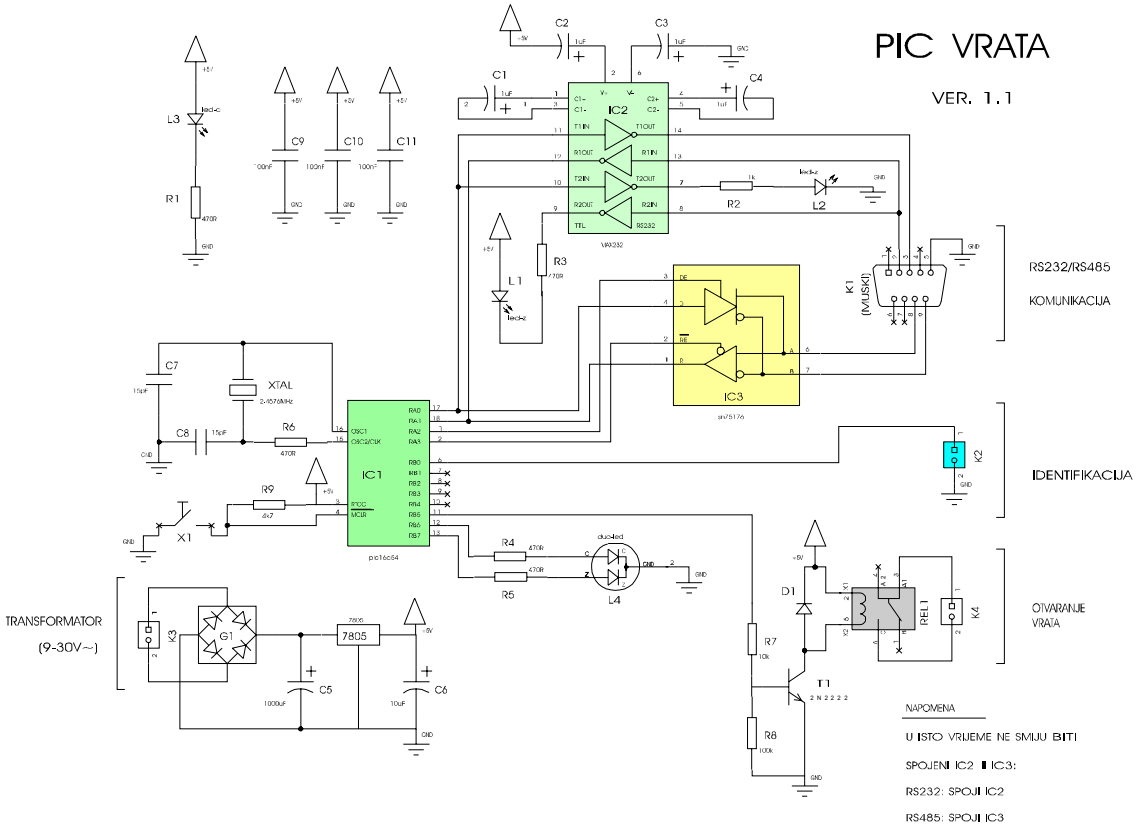
Sklopovsko rješenje:

U ovom primjeru objedinjeno je više jednostavnih zadataka u složenu cjelinu. Cjelovito rješenje zadatka prikazano je na slici 3.3. Lokalni mikrokontroler zadužen je za prihvat identifikacijskog koda (ID) osobe koja želi ući, lokalnom mrežom (RS-485) proslijedi ID do drugog mikrokontrolera koji ima ulogu arbitra i povezuje lokalnu mrežu preko RS-232 na radnu stanicu. Program koji se izvodi na radnoj stanici provjerava kakva je dozvola za osobu koja se upravo identificirala i ako je sve u redu, daje nalog za otvaranje vrata. Nalog preuzima arbitar koji ga lokalnom RS-485 mrežom prosljeđuje do lokalnog mikrokontrolera. Lokalni mikrokontroler uključuje relej za otvaranje vrata. Na slikama 3.4 i 3.5 dane su sheme lokalnog mikrokontrolera (VRATA) te mikrokontrolera koji obavlja ulogu arbitra između lokalne mreže i radne stanice (ARBITAR). Ovakva koncepcija rješenja odabrana je zbog budućih proširenja u lokalnoj mreži.

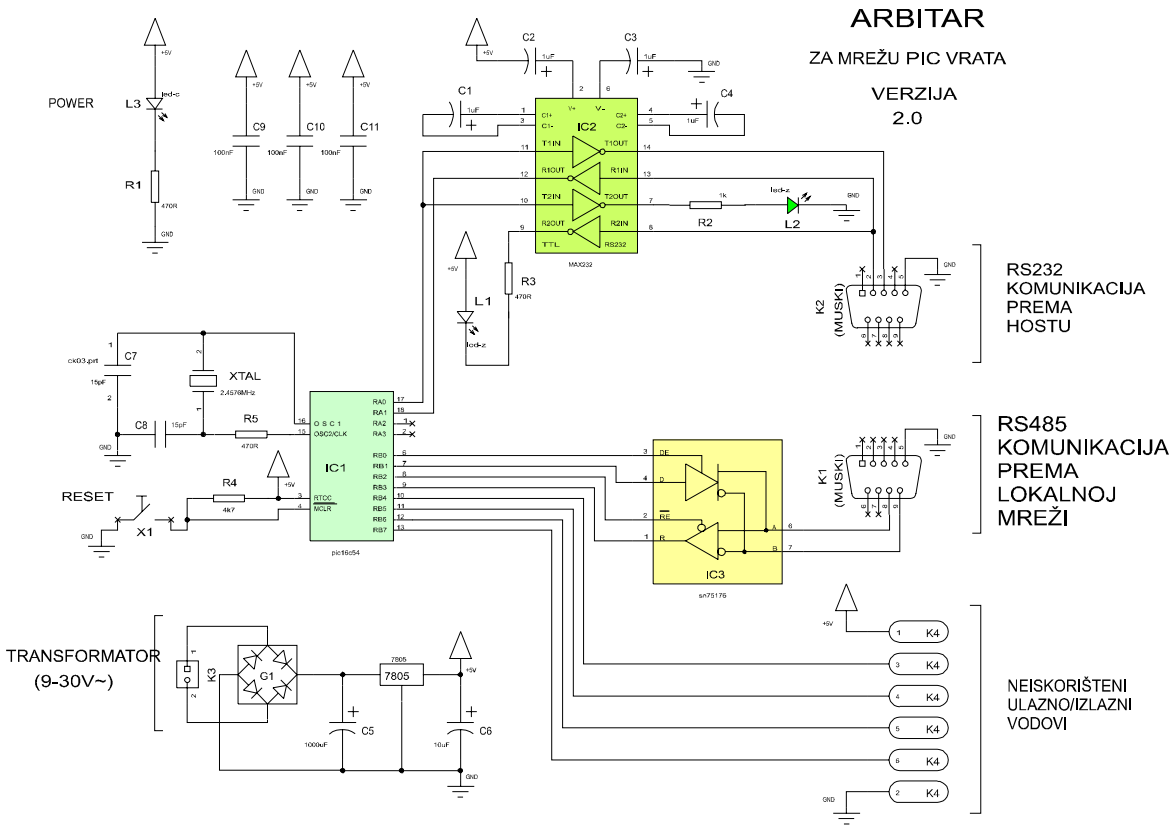


Slika 3.3: Primjena mikrokontrolera PIC za identifikaciju osoba i otvaranje vrata

Kao ulazna jedinica (za identifikaciju osoba) koristi se elektronička komponenta koja sadrži jednostavnu ROM memoriju (eng. Read Only Memory) u kojoj je upisan jedinstveni identifikacijski broj. Memorija je pasivna (nema svoje napajanje), zalivena je u kućište po izgledu jednako baterijama za ručne satove i ima dva kontakta. Identifikacijski broj prenosi se iz memorije u lokalni mikrokontroler tzv. jednožičnim komunikacijskim protokolom (eng. Single Wire Protokol) opisanim u literaturi[_]. Izvodi na kućištu memorije prislone se na odgovarajuće kućište ugrađeno u okvir vrata (i spojeno na mikrokontroler). Prilikom prislanjanja memorije na to kućište od mikrokontrolera se pribavi potrebna energija za daljnji rad, detektira se prisutnost memorije i da nalog za serijsko čitanje (bit po bit) sadržaja memorije.
Izlazna jedinica je relej koji privlači kotvu elektromagnetske brave ugrađene u vrata.



Slika 3.4: Shema lokalnog mikrokontrolera (PIC 16C54) za identifikaciju osoba i otvaranje vrata



Slika 3.5: Shema mikrokontrolera PIC 16C54 (arbitar lokalne mreže)

Programska podrška:

Programska podrška sastoji se iz više cjelina. U mikrokontroleru zaduženom za vrata postoji modul za identifikaciju i kmunikaciju s Dallas memorijom. U prvom dijelu programa definirani su vremenski razmaci potrebni za detekciju prisutnosti memorije te čitanje njezinog sadržaja. Na kraju memorija generira CRC (eng. Cyclic Redundancy Check) polinom koji je potrebno izračunati na prijemnoj strani (mikrokontroler) i usporediti s prenesenim podacima o ID broju kako bismo bili sigurni da je ID broj ispravno pročitan. Iza toga slijedi programski modul za pretvorbu ID broja u ASCII format, slanje ID broja preko RS-485 mreže, brzinom 9600 b/s. Također postoji modul za čitanje povratne informacije s mreže te modul za aktiviranje releja brave.

```
          // PIC vrata  verzija 2.3 03.05.1994
          // RS232 komunikacija s pretvorbom u ASCII znakove

          // objasnjenje registara:
          // f8 - brojac bitova
          // f9 - brojac znakova
          // f10 - CRC registar
          // f12 - f27 prijemni registri
          // f14 - f28 predajni ASCII registri
          // f28 - ASCII LF registar
          // f29 - pomocni registar za manipulaciju znakovima
          // f30 - brojac desetinki sekunde
          // f31 - registar primljene poruke od SUN Sparc racunala

            `BASE D

000  C1E    MOVLW %B 00011110
001  006    TRIS  6           // B0-Dallas, B5-relej
                              // B6-zelena LED, B7-crvena LED

002  C02    MOVLW %B 00000010
003  005    TRIS  5           // A0-TX, A1-RX, A2-TA, A3-RA`

004  C05    MOVLW %B 00000101
005  025    MOVWF 5           // aktiviranje RA` i TA

006  C01  POC MOVLW  1        // 00000001
007  026    MOVWF 6           // "1" na B0, duo-led ugasena

008  C03    MOVLW 3           // prescaler 1:16 (26 us)
009  002    OPTION

            // --------- RESET signal i detekcija prisutnosti ----------
00A  406  RST BCF 6,0         // "0" na izlazu ==> RESET signal

00B  CED    MOVLW 237         // 256 + 1 - 20  (20x26 us)
00C  021    MOVWF 1           // 520 us  10....01
00D  201  P1  MOVF   1,0      //
00E  743    BTFSS 3,2
00F  A0D    GOTO   P1

010  506    BSF   6,0         // "1" na izlazu

011  CFE    MOVLW 254         // 256 + 1 - 3 (3x26 us)
012  021    MOVWF 1           // 78 us   01....10
013  201  P2  MOVF   1,0      //
014  743    BTFSS 3,2
015  A13    GOTO  P2

016  206    MOVF  6,0         // da li je Dallas prisutan?
017  606    BTFSC 6,0
018  A0A    GOTO  RST
```

```
            // ---------- naredba za citanje memorije ----------

019  CED    MOVLW 237          // 256 + 1 - 20
01A  021    MOVWF 1            // 520 us
01B  201  P3  MOVF   1,0       //
01C  743    BTFSS 3,2
01D  A1B    GOTO  P3

01E  C04    MOVLW 4            // 4 petlje za "1" (0F0H->11110000)
01F  028    MOVWF 8

020  406  JED4 BCF   6,0       // signalizacija jedinice
021  000    NOP                // 101
022  506    BSF   6,0

023  CFE    MOVLW 254          // 256 + 1 - 3
024  021    MOVWF 1            // 78 us 01...101...101...101...1
025  201  P4  MOVF   1,0       //
026  743    BTFSS 3,2
027  A25    GOTO  P4

028  2E8    DECFSZ 8,1         // jedna jedinica manje
029  A20    GOTO   JED4

02A  C04    MOVLW 4            // 4 petlje za "0"
02B  028    MOVWF 8

02C  406  NUL4 BCF   6,0       // signalizacija nule

02D  CFE    MOVLW 254          // 256 + 1 - 3
02E  021    MOVWF 1            // 78 us 10...010...010...010...1
02F  201  P5  MOVF   1,0       //
030  743    BTFSS 3,2
031  A2F    GOTO  P5

032  506    BSF   6,0
033  2E8    DECFSZ 8,1         // jedna nula manje
034  A2C    GOTO  NUL4

              // -- citanje dodirne memorije i racunanje CRC ---
              // odgovor na 0F0H je 64 bita iz memorije
              //  8 bita  vrsta cipa (1990)
              // 48 bita  ID (12 hex. znakova)
              //  8 bita  CRC (x**8 + X**5 + X**4 + 1)
              // sve zajedno daje CRC 0

035  C10    MOVLW 16
036  029    MOVWF 9            // f9 = 16 hex znakova
037  C04    MOVLW 4
038  028    MOVWF 8            // f8 = 4 bita
039  06A    CLRF  10           // CRC koji racunamo je u f10

03A  C1B    MOVLW 27           // f4 -> f27 je
```

```
03B  024    MOVWF 4           // pocetni registar

03C  406  SIN BCF    6,0      // impuls za sinkronizaciju
03D  000    NOP               // 101
03E  506    BSF   6,0
03F  000    NOP
040  000    NOP

041  320    RRF   0,1         // pomakni spremnik za prihvat novog bita
042  4E0    BCF   0,7
043  606    BTFSC 6,0         // ocitava bit
044  5E0    BSF   0,7

045  30A    RRF   10,0        // rotiranje CRC bitova bez preljeva
046  32A    RRF   10,1

047  C80    MOVLW %B 10000000
048  140    ANDWF 0,0
049  1AA    XORWF 10,1        // ispitivanje ulaznog XOR bita
04A  7EA    BTFSS 10,7
04B  A4E    GOTO  NEX
04C  C0C    MOVLW %B 00001100 // izgled zadanog CRC polinoma
04D  1AA    XORWF 10,1

04E  CFE  NEX MOVLW  254      // 256 + 1 - 3
04F  021    MOVWF 1           // 78 us
050  201  P6  MOVF   1,0      //
051  743    BTFSS 3,2
052  A50    GOTO  P6

053  2E8    DECFSZ 8,1        // jedan bit manje
054  A3C    GOTO  SIN

055  3A0    SWAPF 0,1
056  C0F    MOVLW %B 00001111 // visa 4 bita su nula
057  160    ANDWF 0,1
058  C0A    MOVLW  10
059  080    SUBWF 0,0         // f29 < 10 ?
05A  623    BTFSC 3,1
05B  A5E    GOTO  SLO
05C  C30    MOVLW 48          // ASCII (48) = "0"
05D  A5F    GOTO  ZBR
05E  C37  SLO MOVLW  55       // ASCII (65) = "A"
05F  1E0  ZBR ADDWF  0,1

060  0E4    DECF  4,1         // pomak pokazivaca na
061  C04    MOVLW 4           // novi znak od 4 bita
062  028    MOVWF 8
063  2E9    DECFSZ 9,1        // jedan znak manje
064  A3C    GOTO  SIN

065  20A    MOVF  10,0        // ispitaj izracunati CRC
066  743    BTFSS 3,2         // ako CRC nije nula - greska
```

```
067  A06    GOTO  POC          // vraca na pocetak programa

068  C30    MOVLW 48           // ASCII (48) = "0"
069  09B    SUBWF 27,0
06A  743    BTFSS 3,2
06B  A70    GOTO  NNUL
06C  C30    MOVLW 48           // ASCII (48) = "0"
06D  09A    SUBWF 26,0
06E  643    BTFSC 3,2
06F  A06    GOTO  POC


            // ---------- slanje ASCII znakova ----------

070  C0E  NNUL MOVLW 14        // f4 -> f14 pocetni registar
071  024    MOVWF 4
072  C0A    MOVLW 10           // ASCII LF
073  03C    MOVWF 28

074  C0F    MOVLW 15           // brojac znakova
075  029    MOVWF 9

076  545    BSF   5,2          // aktiviranje TA

077  C08  ASC MOVLW  8
078  028    MOVWF 8            // f8 = 8 bita

079  405    BCF   5,0          // Start Bit

07A  CFD    MOVLW 253          // 256 + 1 - 4 (9600b/s->4x26 us)
07B  021    MOVWF 1            // 104 us
07C  201  CEK1 MOVF  1,0
07D  743    BTFSS 3,2
07E  A7C    GOTO  CEK1

07F  700  POC1 BTFSS 0,0
080  A83    GOTO  NUL1

081  505    BSF   5,0
082  A84    GOTO  JED1

083  405  NUL1 BCF   5,0

084  CFD  JED1 MOVLW 253       // 256 + 1 - 4
085  021    MOVWF 1            // 104 us
086  201  CEK2 MOVF  1,0
087  743    BTFSS 3,2
088  A86    GOTO  CEK2

089  300    RRF   0,0
08A  320    RRF   0,1          // priprema slijedeci bit
08B  2E8    DECFSZ 8,1         // jedan bit manje
08C  A7F    GOTO  POC1
```

```
08D  505    BSF   5,0         // Stop bit

08E  CFD    MOVLW 253         // 256 + 1 - 4
08F  021    MOVWF 1           // 104 us
090  201  CEK3 MOVF  1,0
091  743    BTFSS 3,2
092  A90    GOTO  CEK3

093  2A4    INCF  4,1         // slijedeci bajt
094  2E9    DECFSZ 9,1        // jedan bajt manje
095  A77    GOTO  ASC

096  445    BCF   5,2         // deaktiviranje TA

            // ---------- cekanje potvrde od SUN Sparc racunala ---------

097  625  POT BTFSC  5,1      // da li se pojavio start bit
098  A97    GOTO  POT

099  CFF    MOVLW 255         // 256 + 1 - 2 (sredina bita)
09A  021    MOVWF 1           // 52 us
09B  201  PRO MOVF   1,0      // cekanje pola bita duzine
09C  743    BTFSS 3,2         // radi provjere start bita
09D  A9B    GOTO  PRO

09E  625    BTFSC 5,1         // provjeri da li je to zaista start bit
09F  A97    GOTO  POT         // moze pokusati ponovno, ali bi trebalo
                              // vremensko ogranicenje

0A0  C08    MOVLW 8           // ukupno 8 bita poruke
0A1  028    MOVWF 8

0A2  33F  BIT RRF 31,1        // pomice spremnik za upis narednog bita

0A3  CFD    MOVLW 253         // 256 + 1 - 4
0A4  021    MOVWF 1           // 104 us
0A5  201  CEK MOVF   1,0      //
0A6  743    BTFSS 3,2         // cekanje narednog bita
0A7  AA5    GOTO  CEK

0A8  4FF    BCF   31,7        // postavlja "0"
0A9  625    BTFSC 5,1         // cita bit
0AA  5FF    BSF   31,7        // postavlja "1"


0AB  2E8    DECFSZ 8,1        // da li su svi bitovi stigli?
0AC  AA2    GOTO  BIT

0AD  CFD    MOVLW 253         // 256 + 1 - 4
0AE  021    MOVWF 1           // 104 us
0AF  201  STP MOVF   1,0
0B0  743    BTFSS 3,2         // cekanje stop bita
```

```
0B1  AAF    GOTO  STP

0B2  725    BTFSS 5,1         // provjerava stop bit
0B3  A97    GOTO  POT

            // ---------- signalizacija ----------

0B4  C55    MOVLW 85          // ASCII znak "U"
0B5  09F    SUBWF 31,0        // ID je u redu
0B6  643    BTFSC 3,2
0B7  ABA    GOTO  OK
0B8  C41    MOVLW 65          // 01000001 ako je krivi broj
0B9  ABB    GOTO  SIG         // ukljuci crveni LED
0BA  CA1  OK  MOVLW  161      // 10100001 ako je sve OK
0BB  026  SIG MOVWF  6        // ukljuci relej i cekaj 3 sekunde


            // ---------- cekanje 3 s -----------

0BC  C07    MOVLW 7           // prescaler 1:256
0BD  002    OPTION

0BE  C1E    MOVLW 30          // 30 destinki sekunde
0BF  03E    MOVWF 30

0C0  C01  DES MOVLW  1
0C1  021    MOVWF 1

0C2  201  PET MOVF   1,0
0C3  743    BTFSS 3,2
0C4  AC2    GOTO  PET

0C5  2FE    DECFSZ 30,1
0C6  AC0    GOTO  DES

0C7  A06    GOTO  POC         // vraca na pocetak programa
            `END
```

Zadatak arbitra je daleko jednostavniji u ovom slučaju. On služi kao posrednik koji prikuplja pozatke od lokalnog kontrolera i prosljeđuje ih preko RS-232C do radne stanice. Zbog ograničenog prostora ovdje nije dan program (napisan u jeziku C) koji u radnoj stanici (UNIX operacijski sustav) uspoređuje ID s trenutnim dozvolama i ako je sve u redu daje potvrdu (ASCII U) da se otvore vrata. Arbitar to prosljeđuje do lokalnog kontrolera. Ovo je najjednostavniji oblik programa koji se može proširiti i obogatiti novim funkcijama.

```
// PIC arbitar      verzija 1.0 03.05.1994
// RS232 komunikacija s pretvorbom u ASCII znakove

// objasnjenje registara:
// f8 - brojac bitova
```

```
          // f9 - brojac znakova
          // f10 - CRC registar
          // f12 - f27 prijemni registri
          // f14 - f28 predajni ASCII registri
          // f28 - ASCII LF registar
          // f29 - pomocni registar za manipulaciju znakovima
          // f30 - brojac desetinki sekunde
          // f31 - registar primljene poruke od SUN Sparc racunala

            `BASE D

000  CF8    MOVLW %B 11111000
001  006    TRIS  6           // B0-TA, B1-TX, B2-RA`, B3-RX za RS485

002  C0E    MOVLW %B 00001110
003  005    TRIS  5           // A0-TX, A1-RX za RS232

004  505    BSF   5,0         // postavlja TX liniju u neaktivno stanje

005  C03    MOVLW %B 00000011
006  026    MOVWF 6           // aktivira RA` i TA linije

007  C03    MOVLW 3           // prescaler 1:16
008  002    OPTION

            // --------- primanje podataka ---------


009  C0F  POC MOVLW  15
00A  029    MOVWF 9           // f9 = 15 znakova
00B  C08    MOVLW 8
00C  028    MOVWF 8           // f8 = 8 bita

00D  C0C    MOVLW 12          // f4 -> f12 je
00E  024    MOVWF 4           // pocetni registar

00F  666  POT1 BTFSC 6,3      // da li se pojavio start bit
010  A0F    GOTO  POT1

011  CFF    MOVLW 255         // 256 + 1 - 2
012  021    MOVWF 1           // 52 us
013  201  PRO1 MOVF  1,0      // cekanje pola bita duzine
014  743    BTFSS 3,2         // radi provjere start bita
015  A13    GOTO  PRO1

016  666    BTFSC 6,3         // provjeri da li je to zaista start bit
017  A0F    GOTO  POT1        // moze pokusati ponovno, ali bi trebalo
                              // vremensko ogranicenje

018  C08    MOVLW 8           // ukupno 8 bita poruke
019  028    MOVWF 8

01A  320  BIT1 RRF   0,1      // pomice spremnik za upis narednog bita
```

```
01B  CFD    MOVLW 253          // 256 + 1 - 4
01C  021    MOVWF 1            // 104 us
01D  201  CK1 MOVF   1,0       //
01E  743    BTFSS 3,2          // cekanje narednog bita
01F  A1D    GOTO  CK1

020  4E0    BCF   0,7          // postavlja "0"
021  666    BTFSC 6,3          // cita bit
022  5E0    BSF   0,7          // postavlja "1"


023  2E8    DECFSZ 8,1         // da li su svi bitovi stigli?
024  A1A    GOTO  BIT1

025  CFD    MOVLW 253          // 256 + 1 - 4
026  021    MOVWF 1            // 104 us
027  201  STP1 MOVF  1,0
028  743    BTFSS 3,2          // cekanje stop bita
029  A27    GOTO  STP1

02A  766    BTFSS 6,3          // provjerava stop bit
02B  A0F    GOTO  POT1

02C  2A4    INCF  4,1          // pomak pokazivaca na
02D  C08    MOVLW 8            // novi znak od 8 bita
02E  028    MOVWF 8
02F  2E9    DECFSZ 9,1         // jedan znak manje
030  A0F    GOTO  POT1

             // ---------- slanje ASCII znakova ----------

031  C0C    MOVLW 12           // f4 -> f12 pocetni registar
032  024    MOVWF 4

033  C0F    MOVLW 15           // brojac znakova
034  029    MOVWF 9

035  C08  ASC MOVLW  8
036  028    MOVWF 8            // f8 = 8 bita

037  405    BCF   5,0          // Start Bit

038  CFD    MOVLW 253          // 256 + 1 - 4
039  021    MOVWF 1            // 104 us
03A  201  CEK1 MOVF  1,0
03B  743    BTFSS 3,2
03C  A3A    GOTO  CEK1

03D  700  POC1 BTFSS 0,0
03E  A41    GOTO  NUL1

03F  505    BSF   5,0
```

```
040  A42    GOTO  JED1

041  405  NUL1 BCF   5,0

042  CFD  JED1 MOVLW 253      // 256 + 1 - 4
043  021    MOVWF 1           // 104 us
044  201  CEK2 MOVF  1,0
045  743    BTFSS 3,2
046  A44    GOTO  CEK2

047  300    RRF   0,0
048  320    RRF   0,1         // priprema slijedeci bit
049  2E8    DECFSZ 8,1        // jedan bit manje
04A  A3D    GOTO  POC1

04B  505    BSF   5,0

04C  CFD    MOVLW 253         // 256 + 1 - 4
04D  021    MOVWF 1           // 104 us
04E  201  CEK3 MOVF  1,0
04F  743    BTFSS 3,2
050  A4E    GOTO  CEK3

051  2A4    INCF  4,1         // slijedeci znak
052  2E9    DECFSZ 9,1        // jedan znak manje
053  A35    GOTO  ASC

            // ----------- cekanje dozvole -------------

054  625  POT BTFSC  5,1      // da li se pojavio start bit
055  A54    GOTO  POT

056  CFF    MOVLW 255         // 256 + 1 - 2
057  021    MOVWF 1           // 52 us
058  201  PRO MOVF   1,0      // cekanje pola bita duzine
059  743    BTFSS 3,2         // radi provjere start bita
05A  A58    GOTO  PRO

05B  625    BTFSC 5,1         // provjeri da li je to zaista start bit
05C  A54    GOTO  POT         // moze pokusati ponovno, ali bi trebalo
                              // vremensko ogranicenje

05D  C08    MOVLW 8           // ukupno 8 bita poruke
05E  028    MOVWF 8

05F  33F  BIT RRF 31,1        // pomice spremnik za upis narednog bita

060  CFD    MOVLW 253         // 256 + 1 - 4
061  021    MOVWF 1           // 104 us
062  201  CEK MOVF   1,0      //
063  743    BTFSS 3,2         // cekanje narednog bita
064  A62    GOTO  CEK
```

```
065  4FF    BCF   31,7        // postavlja "0"
066  625    BTFSC 5,1         // cita bit
067  5FF    BSF   31,7        // postavlja "1"


068  2E8    DECFSZ 8,1        // da li su svi bitovi stigli?
069  A5F    GOTO  BIT

06A  CFD    MOVLW 253         // 256 + 1 - 4
06B  021    MOVWF 1           // 104 us
06C  201  STP MOVF   1,0
06D  743    BTFSS 3,2         // cekanje stop bita
06E  A6C    GOTO  STP

06F  725    BTFSS 5,1         // provjerava stop bit
070  A54    GOTO  POT

            // --------------- slanje dozvole ---------------

071  C08    MOVLW 8
072  028    MOVWF 8           // f8 = 8 bita

073  426    BCF   6,1         // Start Bit

074  CFD    MOVLW 253         // 256 + 1 - 4
075  021    MOVWF 1           // 104 us
076  201  CE1 MOVF   1,0
077  743    BTFSS 3,2
078  A76    GOTO  CE1

079  71F  PO1 BTFSS  31,0
07A  A7D    GOTO  NU1

07B  526    BSF   6,1
07C  A7E    GOTO  JE1

07D  426  NU1 BCF 6,1

07E  CFD  JE1 MOVLW  253      // 256 + 1 - 4
07F  021    MOVWF 1           // 104 us
080  201  CE2 MOVF   1,0
081  743    BTFSS 3,2
082  A80    GOTO  CE2

083  31F    RRF   31,0
084  33F    RRF   31,1        // priprema slijedeci bit
085  2E8    DECFSZ 8,1        // jedan bit manje
086  A79    GOTO  PO1

087  526    BSF   6,1

088  CFD    MOVLW 253         // 256 + 1 - 4
089  021    MOVWF 1           // 104 us
```

```
08A  201  CE3 MOVF   1,0
08B  743    BTFSS 3,2
08C  A8A    GOTO  CE3

08D  A09    GOTO  POC

            `END
```

20.3.5 ZAKLJUČAK UZ TREĆU CJELINU

Cilj danih primjera je pokazati kako se na jednostavan način može riješiti niz različitih zadataka. Primjeri su maksimalno pojednostavnjeni kako bi bili razumljiviji. U praksi mogu poslužiti kao osnova za proširivanje funkcija i nadogradnju.

Primjeri također ukazuju da je potrebno promijeniti klasični način razmišljanja o primjeni mikroračunala u upravljačkim i drugim namjenama. Jednočipni mikrokontroleri postaju jednostavne komponente koje zamjenjuju klasične elektroničke sklopove.

Za kraj, filozofija primjene jednočipnih mikrokontrolera može se sagledati i kroz odgovor na jedno učestalo pitanje, već spomenuto u prethodnom tekstu - ako rješavamo serijsku komunikaciju pomoću PIC kontrolera, ostat će nam premalo procesorskog vremena za neke druge aktivnosti, nije li dakle bolje koristiti mikroprocesor i dodatnu komponentu za serijsku komunikaciju?

Odgovor je - da li ste razmišljali o tome da sam PIC postane komponenta za serijsku komunikaciju, a drugi PIC povezan s njime bude potpuno slobodan za rješavanje problema za koji je namijenjen? Takva su rješenja do nedavno bila neizvediva, a danas su veoma interesantna.

Mikrokontroleri u vratima, mlincima za kavu, glačalima, ključevima, ... sigurno će promijeniti svijet kakav poznajemo danas.

LITERATURA:

M.Žagar, M.Kovač, D.Basch, Uvod u mikroračunala, Školska knjiga, Zagreb, 1993.

D.Basch, M.Žagar, ATLAS - Advanced Tools and Languages for Microprocessor Architecture Simulation, Journal of Computing and Information Technology -CIT Vol. 1, No. 3, 1993. (183-197)

G.Smiljanić, Računala i procesi, Školska knjiga, Zagreb, 1991.

G.Smiljanić, Mikroračunala, Školska knjiga, Zagreb, 1987.

G.Smiljanić, 32-bitna mikroračunala, Element, Zagreb, 1993.

M.Žagar, A VLSI circle generator arithmetic-logic unit, UCSB, ECE, VLSI224, Santa Barbara, 1984.

-, PIC16C5X EPROM Based 8-Bit CMOS Microcontroller Series, Microchip Technology Inc., 1991.

-, Zilog Z85C13/C15 CMOS IPC Intelligent Peripheral Controller, Zilog, 1990.

-, Zilog Z84011/C11 Parallel I/O Controller, Zilog, 1990.

M.Žagar, Prilog razvoju automatizacije projektiranja namjenskih integriranih mikrosklopova, Automatika, 5-6/1990. (199-206)

M.Žagar, Upravljačka jedinica mikroračunala specijalne namjene, časopis ITA, 9(1-3)13, 1990, (13-22)

J. Uffenbeck, Microcomputers and Microprocessors: Programming, Interfacing and Troubleshooting, Prentice-Hall, Inc. Englewood Cliffs, N.J., 1985.

C.F. Cargill, Information Technology Standardization, Theory, Process and Organizations, Digital Press, 1989.

J.R.Leigh, An Introduction to distributed computer control, Control and Instrumentation, December 1981. (37-41)

P.L.Borrill, High-speed 32-bit buses for forward-looking computers, IEEE Spectrum, July 1989. (34-37)

W.Stallings, Data and Computer Communications, Second Edition, Macmillan Pub.

Company, New York, 1988.

-, PC Card Standard, Release 2.0, Personal Computer Memory Card International association (PCMCIA), Sunnyvale, CA 94086, September 1991.

R. DeBock, VERSAbus - a multiprocessor bus standard - and VMEbus - its Eurocard counterpart, Microprocessors and Microsystems, Vol 6 No 9, November 1982. (475-481)

L.B.Glass, Inside EISA, BYTE, November 1989., (417-425)

L.B.Glass, The SCSI Bus, Part 1, BYTE, February 1990., (267-274)

L.B.Glass, The SCSI Bus, Part 2, BYTE, March 1990., (291-298)

M.Žagar, D.Fouts, IEEE-488 interface for the Motorola MC68000 microcomputer, University of California, Santa Barbara, ECE, Proj. 594-O, Santa Barbara, 1983.

-, EIA-232-D, Revision of EIA-232-C, EIA Standard, ANSI, Electronic Industries Association, January 1987.

-, RS-485 EIA STANDARD, Electronic Industries Association, Washington, April 1983.

-, SAB80515 Single Chip Microcontroller, Siemens, 1985.

B.V.Dyke, SCSI: The I/O Standard Evolves, BYTE IBM Special Edition, Fall 1990, (187-191)

G.Mužak, D.Trupec, M.Žagar, Sučelja za rad s udaljenim mjernim instrumentima, 39. Međunarodni godišnji skup KoREMA, 1994.

M.Kovač, M.Žagar, Mrežni protokol za jednu klasu raspodijeljenog sustava nadzora i upravljanja, ETAN u pomorstvu, Zadar, 1991. (153-156)

M.Kovač, M.Žagar, Single Wire Protocol in Persons and Objects Identification, 36th International Symposium Electronics in Marine, Božava, September 1994.