STRUCNI CLANCI OBJAVLJENI U CASOPISU ZA ELEKTRONIKU I TELEKOMUNIKACIJE INFO ELEKTRONIKA

TITO SMAILAGIC, inzenjer informatike

INFO ELEKTRONIKA 5/96:
KOMUNIKACIJA IZMEDJU PC RACUNARA I MIKROKONTROLERA 68HC11

INFO ELEKTRONIKA 6/96:
KOMUNIKACIJA IZMEDJU PC RACUNARA I MIKROKONTROLERA 68HC11
DRUGI DEO

INFO ELEKTRONIKA 6/96:
PRECIZNI OBRTOMER SA MIKROKONTROLEROM 68HC11

INFO ELEKTRONIKA 8/96:
POVEZIVANJE ALFANUMERICKIH LCD DISPLEJ MODULA
SA MIKROKONTROLERIMA 68HC11

KOMUNIKACIJA IZMEDJU PC RACUNARA I MIKROKONTROLERA 68HC11

Tito Smailagic, inzenjer informatike

Da bi uspostavili komunikaciju izmedju PC racunara i mikrokontrolera 68HC11 potrebno je da ih hardverski povezemo preko standardne serijske RS232 veze. Na PC racunaru izaberemo slobodan serijski port, na primer COM2, koji moze biti izveden preko 9-to polnog muskog konektora D9M, ili 25-to polnog muskog konektora D25M na kucistu racunara.

Raspored izvoda na ovim konektorima je sledeci:

D25M:

2 TXD
3 RXD
4 RTS
5 CTS
6 DSR
7 GND
8 DCD
20 DTR

D9M:

1 DCD
2 RXD
3 TXD
4 DTR
5 GND
6 DSR
7 RTS
8 CTS

Za povezivanje nam je potreban kabl sa 3 zice i odgovarajuci zenski D9F, ili D25F konektor. Na konektoru je potrebno medjusobno spojiti sledece signale: RTS-CTS i DSR-DCD-DTR.

Signal TXD treba preko RS232 interfejsa spojiti sa PD0/RXD mikrokontrolera.
Signal RXD treba preko RS232 interfejsa spojiti sa PD1/TXD mikrokontrolera.
Signal GND treba povezati direktno sa GND/VSS mikrokontrolera.

Za RS232 interfejs na strani mikrokontrolera mozemo koristiti bilo koje integralno kolo za tu namenu: MAX232, MAX233, MC1488/MC1489, ili ga mozemo diskretno napraviti od tranzistora.

Da bi se razumeli PC racunar i mikrokontroler, potrebna je pored hardverske i softverska veza.
Za PC racunar nam je potreban odgovarajuci komunikacioni softver, kao i za mikrokontroler.

Da bi smo uradili prvu komunikaciju sa mikrokontrolerom, postavicemo ga u BOOT mod. Na taj nacin smo ukljucili BOOT ROM u kome se nalazi komunikacioni program mikrokontrolera. Da bi

smo u potpunosti sagledali nacine komunikacije obradicemo tri modela mikrokontrolera 68HC11. To su modeli A1, E1 i F1. Za njih dajemo kompletne originalne listinge komunikacionog softvera.
U ovom slucaju taj softver ima zadatak da nam omoguci punjenje nekog naseg programa u internu RAM memoriju mikrokontrolera.

Komunikacioni programi za PC racunar su pisani u C jeziku i prevode se u EXE format preko standardnog BORLAND TURBO C++ kompajlera. Prilikom startovanja ovog programa mora se zadati ime zeljenog programa pisanog za mikrokontroler u binarnom formatu. Ovaj program moze da bude napisan u bilo kom jeziku za mikrokontroler 68HC11, ali mora da bude preveden u binarni oblik. On podrazumeva da program pocinje i da se izvrsava od adrese 0000. Ukoliko nemamo nikakav softver za pravljenje ovog programa, to mozemo uraditi i direktnim pisanjem masinskih kodova za 68HC11 u binarnom obliku.

Pri pisanju ovog programa moramo voditi racuna o sledecem:
- da se mikrokontroler nalazi u BOOT modu i da je konfigurisan kao SINGLE CHIP.
- da se pseudo interapt vektori nalaze od adrese $00C4 do adrese $00FF.
- da je kapacitet interne RAM memorije kod modela:
  A1 256 bajtova, $0000 - $00FF, stek SP=$00FF
  E1 512 bajtova, $0000 - $01FF, stek SP=$01FF
  F1 1024 bajtova, $0000 - $03FF, stek SP=$03FF

```
0001                                ************************************************************
0002                                * A1BOOT.ASM: BOOTLOADER FIRMWARE FOR MC68HC11A1 W/O SECURITY *
0003                                ************************************************************
0004
0005                                * EQUATES FOR USE WITH INDEX OFFSET = $1000
0006
0007 0008                           PORTD   EQU     $08
0008 0009                           DDRD    EQU     $09
0009 0028                           SPCR    EQU     $28             * FOR DWOM BIT
0010 002b                           BAUD    EQU     $2B
0011 002c                           SCCR1   EQU     $2C
0012 002d                           SCCR2   EQU     $2D
0013 002e                           SCSR    EQU     $2E
0014 002f                           SCDAT   EQU     $2F
0015 003b                           PPROG   EQU     $3B
0016 003e                           TEST1   EQU     $3E
0017 003f                           CONFIG  EQU     $3F
0018
0019                                * MORE EQUATES
0020
0021 b600                           EEPSTR  EQU     $B600           * START OF EEPROM
0022 b7ff                           EEPEND  EQU     $B7FF           * END OF EEPROM
0023
0024                                * THIS BOOTSTRAP PROGRAM ALLOWS THE USER TO DOWNLOAD A PROGRAM OF EXACTLY
0025                                * 256 BYTES. THE PROGRAM MUST START AT $0000. EACH BYTE OF THE PROGRAM IS
0026                                * RECEIVED BY THE SCI, STARTING WITH THE $0000 BYTE AND WORKING UP TO THE
0027                                * $00FF BYTE. THE TRANSMITTER WILL BE USED FOR THE PURPOSE OF COMMUNICATION
0028                                * TO THE OUTSIDE WORLD.
0029
0030 bf40                                   ORG     $BF40
0031
0032 bf40                           BEGIN   EQU     *
0033
0034 bf40 8e 00 ff          [ 3 ]          LDS     #$00FF          * INIT STACK
0035 bf43 ce 10 00          [ 3 ]          LDX     #$1000          * INIT X REG FOR INDEXED ACCESS TO REGISTERS
0036 bf46 1c 28 20          [ 7 ]          BSET    SPCR,X $20      * PUT PORT D IN WIRE OR MODE
0037 bf49 86 a2             [ 2 ]          LDAA    #$A2            * INIT SCY AND RESTART BAUD DIVIDER CHAIN
0038 bf4b a7 2b             [ 4 ]          STAA    BAUD,X          * DIV BY 16
0039 bf4d 86 0c             [ 2 ]          LDAA    #$0C            * RECEIVER & TRANSMITTER ENABLED
0040 bf4f a7 2d             [ 4 ]          STAA    SCCR2,X
0041
0042                                * SECURITY MODE IS REMOVED
0043
0044 bf51 1c 2d 01          [ 7 ]          BSET    SCCR2,X $01     * SEND BREAK TO SIGNAL START OF DOWNLOAD
0045 bf54 1e 08 01 fc       [ 7 ]          BRSET   PORTD,X $01 *   * CLEAR BREAK AS SOON AS STAST BIT IS DETECTED
0046 bf58 1d 2d 01          [ 7 ]          BCLR    SCCR2,X $01     * CLEAR BREAK
0047
0048                                * WAIT FOR FIRST CHARACTER USERS SEND $FF
0049
0050 bf5b 1f 2e 20 fc       [ 7 ]          BRCLR   SCSR,X $20 *    * WAIT FOR RDRF
0051 bf5f a6 2f             [ 4 ]          LDAA    SCDAT,X         * READ DATA
0052 bf61 26 03             [ 3 ]          BNE     NOZERO          * IF DATA = $00 BREAK OR $00,
0053 bf63 7e b6 00          [ 3 ]          JMP     $B600           * THEN JUMP TO EEPROM
0054
0055 bf66                           NOZERO  EQU     *
0056
0057 bf66 81 55             [ 2 ]          CMPA    #$55            * IF DATA = $55,
0058 bf68 27 1e             [ 3 ]          BEQ     STAR
0059 bf6a 81 ff             [ 2 ]          CMPA    #$FF            * IF DATA = $FF, THEN /16 IS CORRECT BAUD
0060 bf6c 27 03             [ 3 ]          BEQ     BAUDOK
0061 bf6e 1c 2b 33          [ 7 ]          BSET    BAUD,X $33      * ELSE CHANGE TO /104 /13 & /8 1200 @ 2MHZ
0062
0063                                * THEN DOWNLOAD 256 BYTE PROGRAM
0064
0065 bf71                           BAUDOK  EQU     *
0066
0067 bf71 18 ce 00 00       [ 4 ]          LDY     #$0000          * INIT POINTER
0068
0069                                * READ IN PROGRAM AND PUT INTO RAM
0070
0071 bf75                           BK2     EQU     *
0072
0073 bf75 1f 2e 20 fc       [ 7 ]          BRCLR   SCSR,X $20 *    * WAIT FOR RDRF
0074 bf79 a6 2f             [ 4 ]          LDAA    SCDAT,X
0075 bf7b 18 a7 00          [ 5 ]          STAA    0,Y
0076 bf7e a7 2f             [ 4 ]          STAA    SCDAT,X         * HANDSHAKE
0077 bf80 18 08             [ 4 ]          INY
0078 bf82 18 8c 01 00       [ 5 ]          CPY     #$0100          * UNTIL THE END IS REACHED
0079 bf86 26 ed             [ 3 ]          BNE     BK2
0080
```

```
0081                                * ALL START USER'S PROGRAM
0082
0083 bf88                           STAR    EQU     *
0084
0085 bf88 7e 00 00           [ 3 ]          JMP     $0000
0086
0087 bfd4                                   ORG     $BFD4           * NEEDED IF BOOTROM < MA\
0088
0089 bfd4 80 12                             FDB     $8012           * MOTOROLA MC68HC11A1 MASK I. D. BYTE
0090
0091
0092
0093 bfd6 00 c4                             FDB     $100-60         * SCI
0094 bfd8 00 c7                             FDB     $100-57         * SPI
0095 bfda 00 ca                             FDB     $100-54         * PULSE ACCUM INPUT EDGE
0096 bfdc 00 cd                             FDB     $100-51         * PULSE ACCUM OVERFLOW
0097 bfde 00 d0                             FDB     $100-48         * TIMER OVERFLOW
0098 bfe0 00 d3                             FDB     $100-45         * TIMER OUTPUT COMPARE 5
0099 bfe2 00 d6                             FDB     $100-42         * TIMER OUTPUT COMPARE 4
0100 bfe4 00 d9                             FDB     $100-39         * TIMER OUTPUT COMPARE 3
0101 bfe6 00 lc                             FDB     $100-36         * TIMER OUTPUT COMPARE 2
0102 bfe8 00 df                             FDB     $100-33         * TIMER OUTPUT COMPARE 1
0103 bfea 00 e2                             FDB     $100-30         * TIMER INPUT CAPTURE 3
0104 bfec 00 e5                             FDB     $100-27         * TIMER INPUT CAPTURE 2
0105 bfee 00 e8                             FDB     $100-24         * TIMER INPUT CAPTURE 1
0106 bff0 00 eb                             FDB     $100-21         * REAL TIME INT
0107 bff2 00 ee
0108 bff4 00 f1                             FDB     $100-15         * XIRQ
0109 bff6 00 f4                             FDB     $100-12         * SWI
0110 bff8 00 f7                             FDB     $100-9          * ILLEGAL OP-CODE
0111 bffa 00 fa                             FDB     $100-6          * COP FAIL
0112 bffc 00 fd                             FDB     $100-3          * CLOCK MONITOR
0113 bffe bf 40                             FDB     BEGIN           * RESET
0114
0115                                        END

BAUD            002b *0010 0038 0061
BAUDOK          bf71 *0065 0060
BEGIN           bf40 *0032 0113
BK2             bf75 *0071 0079
CONFIG          003f *0017
DDRD            0009 *0008
EEPEND          b7ff *0022
EEPSTR          b600 *0021
NOZERO          bf66 *0055 0052
PORTD           0008 *0007 0045
PPROG           003b *0015
SCCR1           002c *0011
SCCR2           002d *0012 0040 0044 0046
SCDAT           002f *0014 0051 0074 0076
SCSR            002e *0013 0050 0073
SPCR            0028 *0009 0036
STAR            bf88 *0083 0058
TEST1           003e *0016


Number of errors 0
```

```
/************/
/* DOWNA1.C */
/************/

/* Interfejs program za punjenje internog rama mikrokontrolera MC68HC11A1  */
/* iz BOOT moda preko RS232 interfejsa i serijskog porta COM2 PC racunara. */
/* Komunikacioni parametri su: 1200bd, 1 start, 8 data, 1 stop, no parity. */
/* Program zahteva ulazni parametar, koji predstavlja ime datoteke, koja   */
/* je u binarnom obliku duzine od 1 do 256 bajtova i predstavlja masinski  */
/* program, koji se uvlaci u mikrokontroler od adrese 0000 do adrese 00FF. */
/* Ukoliko je datoteka manja od 256 bajtova, ostatak se popunjava sa 00.   */
/* Za vreme slanja podataka, vrsi se provera primljenog podatka od strane  */
/* mikrokontrolera. Kada se napuni poslednji bajt, program se automatski   */
/* izvrsava od adrese 0000. Pre pocetka slanja podataka iz datoteke salje  */
/* se bajt FF, koji predstavlja kod za automatsko podesavanje komunikacione*/
/* frekvencije mikrokontrolera na 1200bd, posto mu je default 7812bd @8MHz */


#include <stdio.h>
#include <bios.h>
#include <conio.h>

#define COM2 1
#define SETTINGS ( 0x80 | 0x03 | 0x00 | 0x00 )
#define BREAK 0x1000
#define PARAM 256

long filesize(FILE *fp);
void test(int,int,int);

void main(int argc, char **argv)
{
   FILE *fp;
   int i,flen,adr=0,ch_prim,ch_posl,buffer[PARAM];

   if((fp=fopen(argv[1],"rb"))==NULL)
        {
        printf("Greska u otvaranju datoteke!\n");
        abort();
        }

   flen = filesize(fp);
   if(flen>PARAM)
        {
        printf("Datoteka je veca od %d bajtova!\n",PARAM);
        abort();
        }

   for (i=0;i<flen;i++)  buffer[i]=fgetc(fp);

   clrscr();
   printf("-----------------------------------------------");
   printf("----------------------\n");
   printf("E N I C   6 8 H C 1 1 A 1   D O W N L O A D   ");
   printf("P C   I N T E R F A C E\n");
   printf("-----------------------------------------------");
   printf("----------------------\n\n\n");
   printf("RESETUJ MIKROKONTROLER !!!\n\n\n");

   bioscom(0, SETTINGS, COM2);
   while(!(bioscom(2, 0, COM2) & BREAK)) {}

   printf("Mikrokontroler je resetovan.\n");
   printf("Poceo je prenos podataka u mikrokontroler...\n");
```

```
   bioscom(1,0xFF,COM2);

   while(adr < flen)
   {
        bioscom(1,ch_posl=buffer[adr],COM2);
        ch_prim = bioscom(2, 0, COM2);
        if(ch_posl != ch_prim) test(adr,ch_posl,ch_prim);
        adr++;
   }

   while(adr < PARAM)
   {
        bioscom(1,ch_posl=0x00,COM2);
        ch_prim = bioscom(2, 0, COM2);
        if(ch_posl != ch_prim) test(adr,ch_posl,ch_prim);
        adr++;
   }

   printf("Uspesno je zavrsen prenos podataka u mikrokontroler.\n");
   fclose(fp);
}

/* ova funkcija vraca duzinu fajla */

long filesize(FILE *fp)
{
    long curpos, length;
    curpos = ftelm(fp);
    fseek(fp, 0L, SEEK_END);
    length = ftell(fp);
    fseek(fp, curpos, SEEK_SET);
    return length;
}

void test(int adr, int ch_posl, int ch_prim)
{
    printf("Greska u komunikaciji!\n");
    printf("adresa: %x poslato: %x primljeno: %x\n",adr,ch_posl,ch_prim);
    abort();
}
```

```
0001                                ************************************************************
0002                                * E1BOOT.ASM: BOOTLOADER FIRMWARE FOR MC68HC11E1 W/O SECURITY *
0003                                ************************************************************
0004
0005                                * THIS NEW VERSION ALLOWS VARIABLE LENGTH DOWNLOAD BY QUITTING RECEPTION OF
0006                                * CHARACTERS WHEN AN IDLE OF AT LEAST FOUR WORD TIMES OCCURS
0007
0008                                * EQUATES FOR USE WITH INDEX OFFSET = $1000
0009
0010 0008                           PORTD   EQU     $08
0011 0016                           TOC1    EQU     $16             * EXTRA STORAGE (POOR STYLE)
0012 0028                           SPCR    EQU     $28             * FOR DWOM BIT
0013 002b                           BAUD    EQU     $2B
0014 002d                           SCCR2   EQU     $2D
0015 002e                           SCSR    EQU     $2E
0016 002f                           SCDAT   EQU     $2F
0017
0018                                * MORE EQUATES
0019
0020 b600                           EEPSTR  EQU     $B600           * START OF EEPROM
0021 b7ff                           EEPEND  EQU     $B7FF           * END OF EEPROM
0022
0023 0000                           RAMSTR  EQU     $0000           * START OF RAM
0024 01ff                           RAMEND  EQU     $01FF           * END OF RAM
0025
0026 0db0                           DELAYS  EQU     3504            * DELAY AT SLOW BAUD
0027 021b                           DELAYF  EQU     539             * DELAY AT FAST BAUD
0028
0029                                * THIS BOOTSTRAP PROGRAM ALLOWS THE USER TO DOWNLOAD A PROGRAM OF 0 - 512
0030                                * BYTES. THE PROGRAM MUST START AT $0000. EACH BYTE OF THE PROGRAM IS RECEIVED
0031                                * BY THE SCI, STARTING WITH THE $0000. THE FIRST BYTE ESTABLISHES BAUD RATE.
0032                                * THEN THE PROGRAM IS DOWNLOADED STARTING WITH THE $0000 BYTE AND WORKING UP
0033                                * TOWARD THE $01FF. A DELAY OF FOUR WORD TIMES AT EITHER BAUD RATE CAUSES THE
0034                                * RECEPTION OF CHARACTERS TO STOP AND A JUMP TO $0000. THE TRANSMITTER WILL BE
0035                                * USED FOR THE PURPOSE OF COMMUNICATION TO THE OUTSIDE WORLD.
0036
0037 bf40                                   ORG     $BF40
0038
0039 bf40 8e 01 ff          [ 3 ] START    LDS     #RAMEND         * INIT STACK
0040 bf43 ce 10 00          [ 3 ]          LDX     #$1000          * INIT X REG FOR INDEXED ACCESS TO REGISTERS
0041 bf46 1c 28 20          [ 7 ]          BSET    SPCR,X $20      * PUT PORT D IN WIRE OR MODE
0042 bf49 cc a2 0c          [ 3 ]          LDD     #$A20C          * INIT SCI AND RESTART BAUD DIVIDER CHAIN
0043 bf4c a7 2b             [ 4 ]          STAA    BAUD,X
0044 bf4e e7 2d             [ 4 ]          STAB    SCCR2,X         * RECEIVER AND TRANSMITTER ENABLED
0045 bf50 cc 02 1b          [ 3 ]          LDD     #DELAYF         * SET UP DELAY FOR FASTEST BAUD RATE
0046 bf53 ed 16             [ 5 ]          STD     TOC1,X
0047 bf55 1c 2d 01          [ 7 ]          BSET    SCCR2,X $01     * SET BREAK TO SIGNAL START OF DOWNLOAD
0048 bf58 1e 08 01 fc       [ 7 ]          BRSET   PORTD,X $01 *   * CLEAR BREAK AS SOON AS START BIT IS DETECTED
0049 bf5c 1d 2d 01          [ 7 ]          BCLR    SCCR2,X $01     * CLEAR BREAK
0050 bf5f 1f 2e 20 fc       [ 7 ]          BRCLR   SCSR,X $20 *    * WAIT FOR FIRST CHARACTER USERS SEND $FF
0051 bf63 a6 2f             [ 4 ]          LDAA    SCDAT,X         * READ DATA
0052 bf65 26 03             [ 3 ]          BNE     NOEEPR
0053 bf67 7e b6 00          [ 3 ]          JMP     EEPSTR          * IF DATA = 00 OR BREAK THEN JUMP TO EEPROM
0054 bf6a 81 ff             [ 2 ] NOEEPR   CMPA    #$FF            * IF DATA = FF THEN /16 IS CORRECT BAUD
0055 bf6c 27 08             [ 3 ]          BEQ     BAUDOK          * 7812BD AT 8MHZ
0056 bf6e 1c 2b 33          [ 7 ]          BSET    BAUD,X $33      * ELSE CHANGE TO /104 /13 & /8 1200BD AT 8MHZ
0057 bf71 cc 0d b0          [ 3 ]          LDD     #DELAYS         * SET UP DELAY FOR SLOWER BAUD RATE
0058 bf74 ed 16             [ 5 ]          STD     TOC1,X
0059
0060 bf76 18 ce 00 00       [ 4 ] BAUDOK   LDY     #RAMSTR         * POINTER TO START OF RAM
0061 bf7a ec 16             [ 5 ] WAIT     LDD     TOC1,X          * PUT DELAY TIME IN ACCD
0062 bf7c 1e 2e 20 07       [ 7 ] WTLOOP   BRSET   SCSR,X $20 NEXT
0063 bf80 8f                [ 3 ]          XGDX                    * DELAY INTO X
0064 bf81 09                [ 3 ]          DEX                     * DECREMENT DELAY
0065 bf82 8f                [ 3 ]          XGDX                    * RETURN DELAY TO ACCD
0066 bf83 26 f7             [ 3 ]          BNE     WTLOOP
0067 bf85 20 0f             [ 3 ]          BRA     JUMP
0068
0069 bf87 a6 2f             [ 4 ] NEXT     LDAA    SCDAT,X         * READ IN BYTE AND PUT INTO RAM
0070 bf89 18 a7 00          [ 5 ]          STAA    0,Y
0071 bf8c a7 2f             [ 4 ]          STAA    SCDAT,X         * HANDSHAKE
0072 bf8e 18 08             [ 4 ]          INY
0073 bf90 18 8c 02 00       [ 5 ]          CPY     #RAMEND+1
0074 bf94 26 e4             [ 3 ]          BNE     WAIT
```

```
0075
0076 bf96 7e 00 00           [ 3 ] JUMP    JMP     RAMSTR          * ALL START USER'S PROGRAM
0077
0078 bfd2                                 ORG     $BFD2
0079
0080 bfd2 04 97                           FDB     $0497
0081 bfd4 e9 e9                           FDB     $E9E9           * MOTOROLA MC68HC11E1 MASK ID
0082
0083                                * VECTORS
0084
0085 bfd6 00 c4                           FDB     $100-60         * SCI
0086 bfd8 00 c7                           FDB     $100-57         * SPI
0087 bfda 00 ca                           FDB     $100-54         * PULSE ACCUM INPUT EDGE
0088 bfdc 00 cd                           FDB     $100-51         * PULSE ACCUM OVERFLOW
0089 bfde 00 d0                           FDB     $100-48         * TIMER OVERFLOW
0090 bfe0 00 d3                           FDB     $100-45         * TIMER OUTPUT COMPARE 5
0091 bfe2 00 d6                           FDB     $100-42         * TIMER OUTPUT COMPARE 4
0092 bfe4 00 d9                           FDB     $100-39         * TIMER OUTPUT COMPARE 3
0093 bfe6 00 dc                           FDB     $100-36         * TIMER OUTPUT COMPARE 2
0094 bfe8 00 df                           FDB     $100-33         * TIMER OUTPUT COMPARE 1
0095 bfea 00 e2                           FDB     $100-30         * TIMER INPUT CAPTURE 3
0096 bfec 00 e5                           FDB     $100-27         * TIMER INPUT CAPTURE 2
0097 bfee 00 e8                           FDB     $100-24         * TIMER INPUT CAPTURE 1
0098 bff0 00 eb                           FDB     $100-21         * REAL TIME INT
0099 bff2 00 ee                           FDB     $100-18         * IRQ
0100 bff4 00 f1                           FDB     $100-15         * XIRQ
0101 bff6 00 f4                           FDB     $100-12         * SWI
0102 bff8 00 f7                           FDB     $100-9          * ILLEGAL OP-CODE
0103 bffa 00 fa                           FDB     $100-6          * COP FAIL
0104 bffc 00 fd                           FDB     $100-3          * CLOCK MONITOR
0105 bffe bf 40                           FDB     START           * RESET
0106
0107                                      END

BAUD            002b *0013 0043 0056
BAUDOK          bf76 *0060 0055
DELAYF          021b *0027 0045
DELAYS          0db0 *0026 0057
EEPEND          b7ff *0021
EEPSTR          b600 *0020 0053
JUMP            bf96 *0076 0067
NEXT            bf87 *0069 0062
NOEEPR          bf6a *0054 0052
PORTD           0008 *0010 0048
RAMEND          01ff *0024 0039 0073
RAMSTR          0000 *0023 0060 0076
SCCR2           002d *0014 0044 0047 0049
SCDAT           002f *0016 0051 0069 0071
SCSR            002e *0015 0050 0062
SPCR            0028 *0012 0041
START           bf40 *0039 0105
TOC1            0016 *0011 0046 0058 0061
WAIT            bf7a *0061 0074
WTLOOP          bf7c *0062 0066


Number of errors 0
```

```
/************/
/* DOWNE1.C */
/************/

/* Interfejs program za punjenje internog rama mikrokontrolera MC68HC11E1  */
/* iz BOOT moda preko RS232 interfejsa i serijskog porta COM2 PC racunara. */
/* Komunikacioni parametri su: 1200bd, 1 start, 8 data, 1 stop, no parity. */
/* Program zahteva ulazni parametar, koji predstavlja ime datoteke, koja   */
/* je u binarnom obliku duzine od 1 do 512 bajtova i predstavlja masinski  */
/* program, koji se uvlaci u mikrokontroler od adrese 0000 do adrese 01FF. */
/* Za vreme slanja podataka, vrsi se provera primljenog podatka od strane  */
/* mikrokontrolera. Kada se napuni poslednji bajt, program se automatski   */
/* izvrsava od adrese 0000. Pre pocetka slanja podataka iz datoteke salje  */
/* se bajt FF, koji predstavlja kod za automatsko podesavanje komunikacione*/
/* frekvencije mikrokontrolera na 1200bd, posto mu je default 7812bd @8MHz */


#include <stdio.h>
#include <bios.h>
#include <conio.h>

#define COM2 1
#define SETTINGS ( 0x80 | 0x03 | 0x00 | 0x00 )
#define BREAK 0x1000
#define PARAM 512

long filesize(FILE *fp);
void test(int,int,int);

void main(int argc, char **argv)
{
   FILE *fp;
   int i,flen,adr=0,ch_prim,ch_posl,buffer[PARAM];

   if((fp=fopen(argv[1],"rb"))==NULL)
        {
        printf("Greska u otvaranju datoteke!\n");
        abort();
        }

   flen = filesize(fp);
   if(flen>PARAM)
        {
        printf("Datoteka je veca od %d bajtova!\n",PARAM);
        abort();
        }

   for (i=0;i<flen;i++)  buffer[i]=fgetc(fp);

   clrscr();
   printf("-----------------------------------------------");
   printf("----------------------\n");
   printf("E N I C   6 8 H C 1 1 E 1   D O W N L O A D   ");
   printf("P C   I N T E R F A C E\n");
   printf("-----------------------------------------------");
   printf("----------------------\n\n\n");
   printf("RESETUJ MIKROKONTROLER !!!\n\n\n");

   bioscom(0, SETTINGS, COM2);
   while(!(bioscom(2, 0, COM2) & BREAK)) {}

   printf("Mikrokontroler je resetovan.\n");
   printf("Poceo je prenos podataka u mikrokontroler...\n");
```

```
   bioscom(1,0xFF,COM2);

   while(adr < flen)
   {
        bioscom(1,ch_posl=buffer[adr],COM2);
        ch_prim = bioscom(2, 0, COM2);
        if(ch_posl != ch_prim) test(adr,ch_posl,ch_prim);
        adr++;
   }

   printf("Uspesno je zavrsen prenos podataka u mikrokontroler.\n");
   fclose(fp);
}

/* ova funkcija vraca duzinu fajla */

long filesize(FILE *fp)
{
    long curpos, length;
    curpos = ftell(fp);
    fseek(fp, 0L, SEEK_END);
    length = ftell(fp);
    fseek(fp, curpos, SEEK_SET);
    return length;
}

void test(int adr, int ch_posl, int ch_prim)
{
    printf("Greska u komunikaciji!\n");
    printf("adresa: %x poslato: %x primljeno: %x\n",adr,ch_posl,ch_prim);
    abort();
}
```

```
0001                                ************************************************************
0002                                * F1BOOT.ASM: BOOTLOADER FIRMWARE FOR MC68HC11F1 W/O SECURITY *
0003                                ************************************************************
0004
0005                                * THIS NEW VERSION ALLOWS VARIABLE LENGTH DOWNLOAD BY QUITTING RECEPTION OF
0006                                * CHARACTERS WHEN AN IDLE OF AT LEAST FOUR WORD TIMES OCCURS
0007
0008                                * EQUATES FOR USE WITH INDEX OFFSET = $1000
0009
0010 0008                           PORTD   EQU     $08
0011 0016                           TOC1    EQU     $16             * EXTRA STORAGE (POOR STYLE)
0012 0028                           SPCR    EQU     $28             * FOR DWOM BIT
0013 002b                           BAUD    EQU     $2B
0014 002d                           SCCR2   EQU     $2D
0015 002e                           SCSR    EQU     $2E
0016 002f                           SCDAT   EQU     $2F
0017
0018                                * MORE EQUATES
0019
0020 fe00                           EEPSTR  EQU     $FE00           * START OF EEPROM
0021 ffff                           EEPEND  EQU     $FFFF           * END OF EEPROM
0022
0023 0000                           RAMSTR  EQU     $0000           * START OF RAM
0024 03ff                           RAMEND  EQU     $03FF           * END OF RAM
0025
0026 0db0                           DELAYS  EQU     3504            * DELAY AT SLOW BAUD
0027 021b                           DELAYF  EQU     539             * DELAY AT FAST BAUD
0028
0029                                * THIS BOOTSTRAP PROGRAM ALLOWS THE USER TO DOWNLOAD A PROGRAM OF 0 - 1024
0030                                * BYTES. THE PROGRAM MUST START AT $0000. EACH BYTE OF THE PROGRAM IS RECEIVED
0031                                * BY THE SCI, STARTING WITH THE $0000. THE FIRST BYTE ESTABLISHES BAUD RATE.
0032                                * THEN THE PROGRAM IS DOWNLOADED STARTING WITH THE $0000 BYTE AND WORKING UP
0033                                * TOWARD THE $03FF. A DELAY OF FOUR WORD TIMES AT EITHER BAUD RATE CAUSES THE
0034                                * RECEPTION OF CHARACTERS TO STOP AND A JUMP TO $0000. THE TRANSMITTER WILL BE
0035                                * USED FOR THE PURPOSE OF COMMUNICATION TO THE OUTSIDE WORLD.
0036
0037 bf00                                   ORG     $BF00
0038
0039 bf00 8e 03 ff          [ 3 ] START    LDS     #RAMEND         * INIT STACK
0040 bf03 ce 10 00          [ 3 ]          LDX     #$1000          * INIT X REG FOR INDEXED ACCESS TO REGISTERS
0041 bf06 1c 28 20          [ 7 ]          BSET    SPCR,X $20      * PUT PORT D IN WIRE OR MODE
0042 bf09 cc b0 0c          [ 3 ]          LDD     #$B00C
0043 bf0c a7 2b             [ 4 ]          STAA    BAUD,X
0044 bf0e e7 2d             [ 4 ]          STAB    SCCR2,X         * RECEIVER & TRANSMITTER ENABLED
0045 bf10 cc 02 1b          [ 3 ]          LDD     #DELAYF         * SET UP DELAY FOR FASTEST BAUD RATE
0046 bf13 ed 16             [ 5 ]          STD     TOC1,X
0047 bf15 1c 2d 01          [ 7 ]          BSET    SCCR2,X $01     * SET BREAK TO SIGNAL START OF DOWNLOAD
0048 bf18 1e 08 01 fc       [ 7 ]          BRSET   PORTD,X $01 *   * CLEAR BREAK AS SOON AS START BIT IS DETECTED
0049 bf1c 1d 2d 01          [ 7 ]          BCLR    SCCR2,X $01     * CLEAR BREAK
0050 bf1f 1f 2e 20 fc       [ 7 ]          BRCLR   SCSR,X $20 *    * WAIT FOR FIRST CHARACTER
0051 bf23 a6 2f             [ 4 ]          LDAA    SCDAT,X         * READ DATA
0052 bf25 26 03             [ 3 ]          BNE     NOEEPR
0053 bf27 7e fe 00          [ 3 ]          JMP     EEPSTR          * IF DATA = 00 OR BREAK THEN JUMP TO EEPROM
0054 bf2a 81 f0             [ 2 ] NOEEPR   CMPA    #$F0            * IF DATA = F0 THEN 9600BD AT 8MHZ
0055 bf2c 27 1d             [ 3 ]          BEQ     OKF0
0056 bf2e c6 33             [ 2 ]          LDAB    #$33
0057 bf30 81 80             [ 2 ]          CMPA    #$80
0058 bf32 27 10             [ 3 ]          BEQ     OK80
0059 bf34 c6 05             [ 2 ]          LDAB    #$05
0060 bf36 85 20             [ 2 ]          BITA    #$20
0061 bf38 27 0a             [ 3 ]          BEQ     OK80
0062 bf3a c6 22             [ 2 ]          LDAB    #$22
0063 bf3c e7 2b             [ 4 ]          STAB    BAUD,X
0064 bf3e 85 08             [ 2 ]          BITA    #$08
0065 bf40 26 09             [ 3 ]          BNE     OKF0
0066 bf42 c6 13             [ 2 ]          LDAB    #$13
0067 bf44 e7 2b             [ 4 ] OK80     STAB    BAUD,X
0068 bf46 cc 0d b0          [ 3 ]          LDD     #DELAYS                 * SET UP DELAY FOR SLOWER BAUD RATE
0069 bf49 ed 16             [ 5 ]          STD     TOC1,X
0070
0071 bf4b 18 ce 00 00       [ 4 ] OKF0     LDY     #RAMSTR                 * POINTER TO START OF RAM
0072 bf4f ec 16             [ 5 ] WAIT     LDD     TOC1,X                  * PUT DELAY TIME IN ACCD
0073 bf51 1e 2e 20 07       [ 7 ] WTLOOP   BRSET   SCSR,X $20 NEXT
0074 bf55 8f                [ 3 ]          XGDX                            * DELAY INTO X
0075 bf56 09                [ 3 ]          DEX                             * DECREMENT DELAY
0076 bf57 8f                [ 3 ]          XGDX                            * RETURN DELAY TO ACCD
0077 bf58 26 f7             [ 3 ]          BNE     WTLOOP
0078 bf5a 20 0f             [ 3 ]          BRA     JUMP
0079
0080 bf5c a6 2f             [ 4 ] NEXT     LDAA    SCDAT,X                 * READ IN BYTE AND PUT INTO RAM
```

```
0081 bf5e 18 a7 00           [ 5 ]         STAA    0,Y
0082 bf61 a7 2f              [ 4 ]         STAA    SCDAT,X                 * HANDSHAKE
0083 bf63 18 08              [ 4 ]         INY
0084 bf65 18 8c 04 00        [ 5 ]         CPY     #RAMEND+1
0085 bf69 26 e4              [ 3 ]         BNE     WAIT
0086
0087 bf6b 7e 00 00           [ 3 ] JUMP    JMP     RAMSTR                  * ALL START USER'S PROGRAM
0088
0089 bfd1                                  ORG     $BFD1
0090
0091 bfd1 42                               FCB     $42
0092 bfd2 00 00                            FDB     $0000
0093
0094 bfd4 f1 f1                            FDB     $F1F1           * MOTOROLA MC68HC11F1 MASK ID
0095
0096                               * VECTORS
0097
0098 bfd6 00 c4                            FDB     $100-60         * SCI
0099 bfd8 00 c7                            FDB     $100-57         * SPI
0100 bfda 00 ca                            FDB     $100-54         * PULSE ACCUM INPUT EDGE
0101 bfdc 00 cd                            FDB     $100-51         * PULSE ACCUM OVERFLOW
0102 bfde 00 d0                            FDB     $100-48         * TIMER OVERFLOW
0103 bfe0 00 d3                            FDB     $100-45         * TIMER OUTPUT COMPARE 5
0104 bfe2 00 d6                            FDB     $100-42         * TIMER OUTPUT COMPARE 4
0105 bfe4 00 d9                            FDB     $100-39         * TIMER OUTPUT COMPARE 3
0106 bfe6 00 dc                            FDB     $100-36         * TIMER OUTPUT COMPARE 2
0107 bfe8 00 df                            FDB     $100-33         * TIMER OUTPUT COMPARE 1
0108 bfea 00 e2                            FDB     $100-30         * TIMER INPUT CAPTURE 3
0109 bfec 00 e5                            FDB     $100-27         * TIMER INPUT CAPTURE 2
0110 bfee 00 e8                            FDB     $100-24         * TIMER INPUT CAPTURE 1
0111 bff0 00 eb                            FDB     $100-21         * REAL TIME INT
0112 bff2 00 ee                            FDB     $100-18         * IRQ
0113 bff4 00 f1                            FDB     $100-15         * XIRQ
0114 bff6 00 f4                            FDB     $100-12         * SWI
0115 bff8 00 f7                            FDB     $100-9          * ILLEGAL OP-CODE
0116 bffa 00 fa                            FDB     $100-6          * COP FAIL
0117 bffc 00 fd                            FDB     $100-3          * CLOCK MONITOR
0118 bffe bf 00                            FDB     START           * RESET
0119
0120                                       END

BAUD             002b *0013 0043 0063 0067
DELAYF           021b *0027 0045
DELAYS           0db0 *0026 0068
EEPEND           ffff *0021
EEPSTR           fe00 *0020 0053
JUMP             bf6b *0087 0078
NEXT             bf5c *0080 0073
NOEEPR           bf2a *0054 0052
OK80             bf44 *0067 0058 0061
OKF0             bf4b *0071 0055 0065
PORTD            0008 *0010 0048
RAMEND           03ff *0024 0039 0084
RAMSTR           0000 *0023 0071 0087
SCCR2            002d *0014 0044 0047 0049
SCDAT            002f *0016 0051 0080 0082
SCSR             002e *0015 0050 0073
SPCR             0028 *0012 0041
START            bf00 *0039 0118
TOC1             0016 *0011 0046 0069 0072
WAIT             bf4f *0072 0085
WTLOOP           bf51 *0073 0077


Number of errors 0
```

```
/************/
/* DOWNF1.C */
/************/

/* Interfejs program za punjenje internog rama mikrokontrolera MC68HC11F1  */
/* iz BOOT moda preko RS232 interfejsa i serijskog porta COM2 PC racunara. */
/* Komunikacioni parametri su: 9600bd, 1 start, 8 data, 1 stop, no parity. */
/* Program zahteva ulazni parametar, koji predstavlja ime datoteke, koja   */
/* je u binarnom obliku duzine od 1 do 1024 bajta i predstavlja masinski   */
/* program koji se uvlaci u mikrokontroler od adrese 0000 do adrese 03FF.  */
/* Za vreme slanja podataka, vrsi se provera primljenog podatka od strane  */
/* mikrokontrolera. Kada se napuni poslednji bajt, program se automatski   */
/* izvrsava od adrese 0000. Pre pocetka slanja podataka iz datoteke salje  */
/* se bajt F0, koji predstavlja kod za automatsko podesavanje komunikacione*/
/* frekvencije mikrokontrolera na default 9600bd @8MHz                     */


#include <stdio.h>
#include <bios.h>
#incdude <conio.h>

#define COM2 1
#define SETTINGS ( 0xe0 | 0x03 | 0x00 | 0x00 )
#define BREAK 0x1000
#define PARAM 1024

long filesize(FILE *fp);
void test(int,int,int);

void main(int argc, char **argv)
{
   FILE *fp;
   int i,flen,adr=0,ch_prim,ch_posl,buffer[PARAM];

   if((fp=fopen(argv[1],"rb"))==NULL)
        {
        printf("Greska u otvaranju datoteke!\n");
        abort();
        }

   flen = filesize(fp);
   if(flen>PARAM)
        {
        printf("Datoteka je veca od %d bajta!\n",PARAM);
        abort();
        }

   for (i=0;i<flen;i++)  buffer[i]=fgetc(fp);

   clrscr();
   printf("-----------------------------------------------");
   printf("----------------------\n");
   printf("E N I C   6 8 H C 1 1 F 1   D O W N L O A D   ");
   printf("P C   I N T E R F A C E\n");
   printf("-----------------------------------------------");
   printf("----------------------\n\n\n");
   printf("RESETUJ MIKROKONTROLER !!!\n\n\n");

   bioscom(0, SETTINGS, COM2);
   while(!(bioscom(2, 0, COM2) & BREAK)) {}

   printf("Mikrokontroler je resetovan.\n");
   printf("Poceo je prenos podataka u mikrokontroler...\n");
```

```
   bioscom(1,0xF0,COM2);

   while(adr < flen)
   {
        bioscom(1,ch_posl=buffer[adr],COM2);
        ch_prim = bioscom(2, 0, COM2);
        if(ch_posl != ch_prim) test(adr,ch_posl,ch_prim);
        adr++;
   }

   printf("Uspesno je zavrsen prenos podataka u mikrokontroler.\n");
   fclose(fp);
}

/* ova funkcija vraca duzinu fajla */

long filesize(FILE *fp)
{
    long curpos, length;
    curpos = ftell(fp);
    fseek(fp, 0L, SEEK_END);
    length = ftell(fp);
    fseek(fp, curpos, SEEK_SET);
    return length;
}

void test(int adr, int ch_posl, int ch_prim)
{
    printf("Greska u komunikaciji!\n");
    printf("adresa: %x poslato: %x primljeno: %x\n",adr,ch_posl,ch_prim);
    abort();
}
```

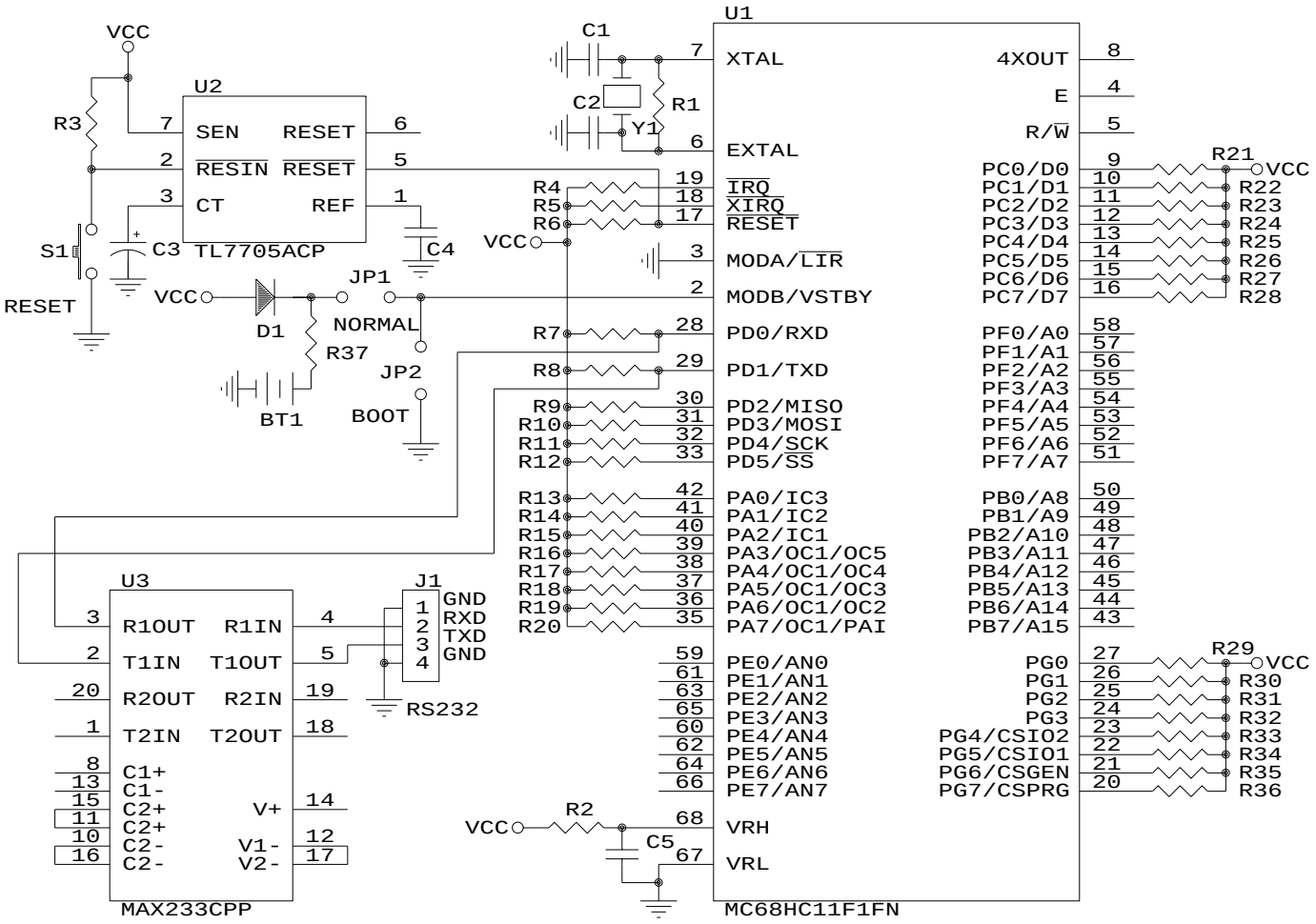
KOMUNIKACIJA IZMEDJU PC RACUNARA I MIKROKONTROLERA 68HC11

- DRUGI DEO -

Tito Smailagic, inzenjer informatike

U prethodnom broju smo opisali komunikaciju PC racunara i mikrokontrolera 68HC11 iz BOOT moda. Ukoliko zelimo da uspostavimo komunikaciju iz NORMAL SINGLE CHIP moda moramo imati mikrokontroler sa programom u njegovoj ROM memoriji za modele A1 i E1, dok je za model F1 potrebno da program bude smesten u EEPROM memoriji. Motorola isporucuje neke verzije modela E1 sa isprogramiranim BUFFALO monitorom u ROM memoriji. To je izuzetan monitor program, koji nam omogucava dosta komforan razvoj programa u SINGLE CHIP modu. Zauzima gornjih 8KB memorijskog prostora od adrese $E000 do $FFFF. Za svoj rad koristi RAM memoriju od adrese $0040 do $00FF. Dozvoljava direktno programiranje EEPROM memorije, koju vidi kao RAM memoriju. U toku razvoja programa pod ovim monitorom na raspolaganju nam je RAM memorija od adrese $0000 do $003F, RAM memorija od adrese $0100 do $01FF i EEPROM memorija od adrese $B600 do $B7FF. Ovaj monitor nam dozvoljava postavljanje test tacki, kao i izvrsavanje programa korak po korak, sa pracenjem sadrzaja svih registara procesora.
Pri startovanju monitor ima mogucnost direktnog skoka u EEPROM memoriju na adresu $B600, pri cemu su nam na raspolaganju sve lokacije RAM memorije od adrese $0000 do $01FF i EEPROM memorije od adrese $B600 do $B7FF. Ukoliko zelimo da koristimo interapt vektore moramo koristiti adrese od $00C4 do $00FF, koje su odredjene kao pseudo interapt vektori.

Za komunikacioni program PC racunara moze se koristiti bilo koji TERMINAL emulator, na primer ST240 sa sledecim komunikacionim parametrima: 1 start bit, 8 bitova za podatke, 1 stop bit, bez pariteta, 9600 boda.



R1=10M0
R2=1K00
R3-R36=10K0
R37=681R
D1=1N5711
BT1=V11R/3.6V

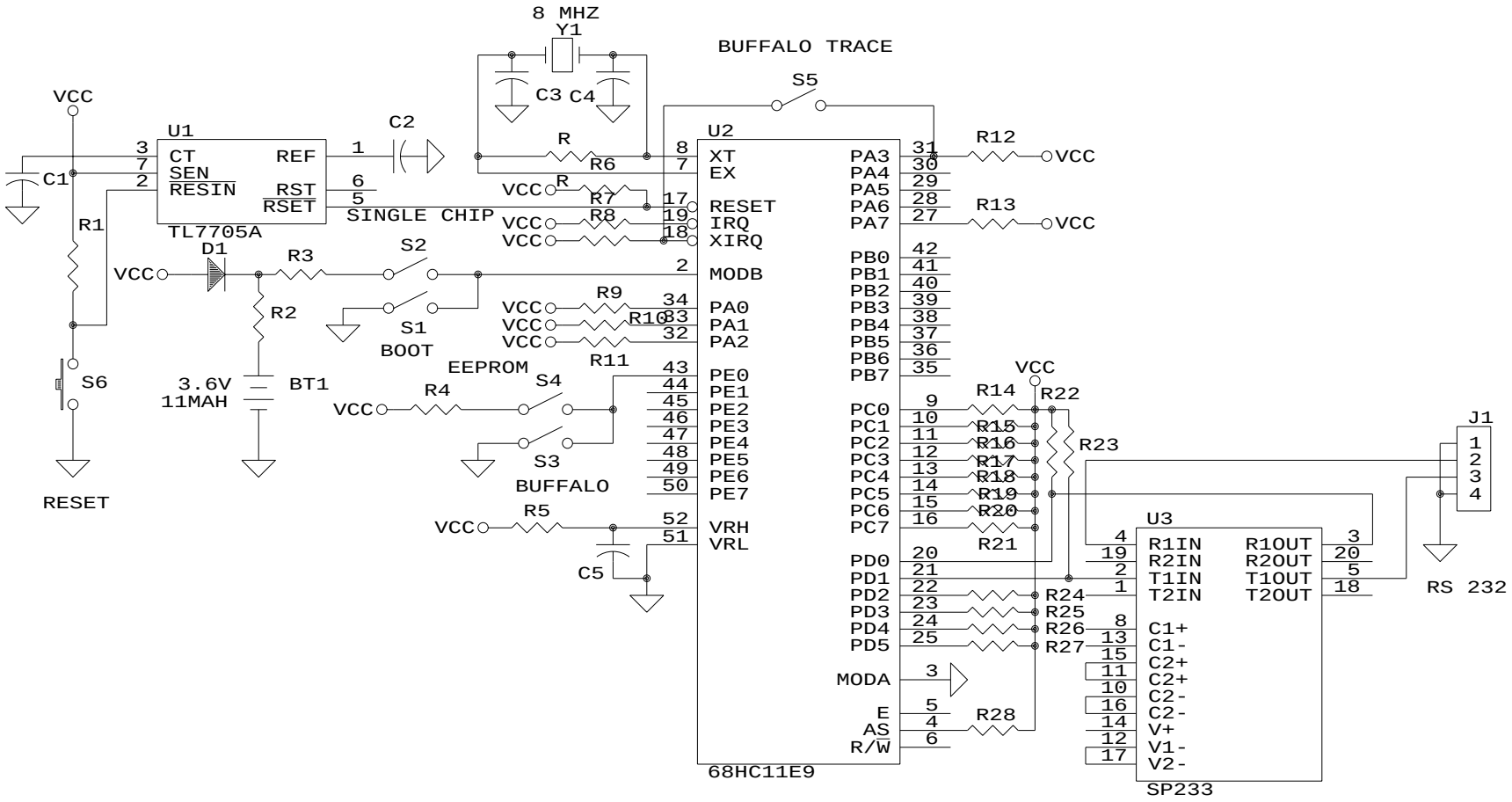
C1=18PF
C2=18PF
C3=10UF
C4=0.1UF
C5=1.0UF
Y1=8MHZ

U1=MC68HC11F1FN
U2=TL7705ACP
U3=MAX233CPP

PLCC68
DIL8
DIL20

ENIC TITO SMAILAGICH
29.NOVEMBRA 20/21 YU-11000
BELGRADE SERBIA YUGOSLAVIA
TEL: +381 11 3226-387

Title		
ENIC 68HC11 EVB 2		
Size: A	Document Number: 2/95	REV: 1
Date: July 9, 1995	Sheet 1 of 1	

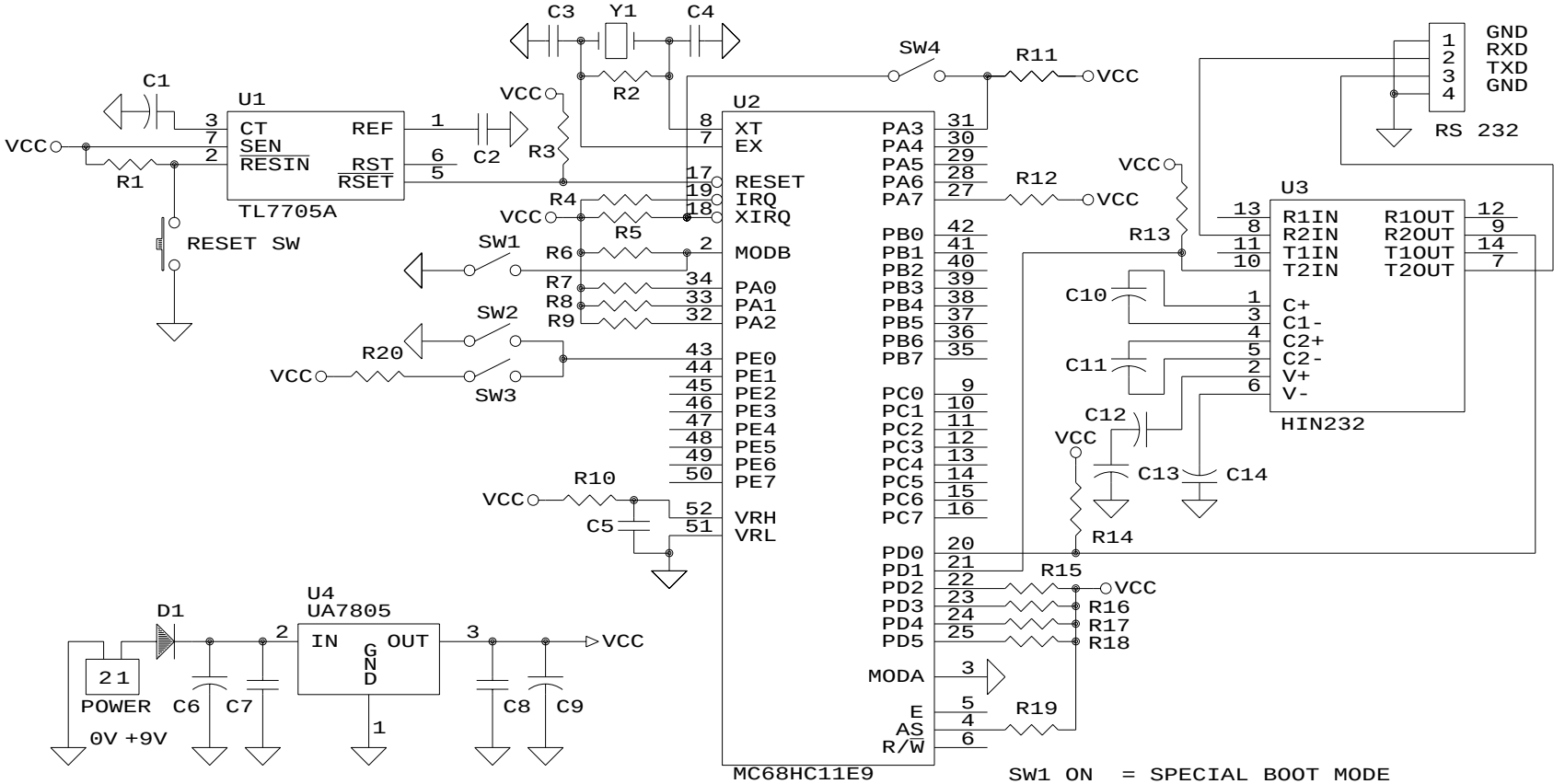


R =10M	R10=10K	R20=10K	C1=10MICROF
R1=10K	R11=10K	R21=10K	C2=0.1MICROF
R2=681R	R12=10K	R22=10K	C3=18PF
R3=10K	R13=10K	R23=10K	C4=18PF
R4=10K	R14=10K	R24=10K	C5=1MICROF
R5=1K0	R15=10K	R25=10K	
R6=10K	R16=10K	R26=10K	
R7=10K	R17=10K	R27=10K	
R8=10K	R18=10K	R28=10K	
R9=10K	R19=10K		D1=1N5711

ENIC TITO SMAILAGICH

29. NOVEMBRA 20 / 21
YU - 11000 BEOGRAD
YUGOSLAVIA
381 - 11 3226 - 387

Title	ENIC 68HC11 EVB14	
Size: A	Document Number: 3/95	REV: 1
Date: March 30, 1996	Sheet 1 of 1	



SW1 ON = SPECIAL BOOT MODE
SW1 OFF = NORMAL SINGLE CHIP MODE

SW2 ON = NORMAL JUMP TO BUFFALO MONITOR
SW2 OFF / SW3 ON = NORMAL JUMP TO EEPROM
CW2 OFF / SW3 OFF = PORT E FUNCTION

SW4 ON = BUFFALO TRACE

U1 = TL7705A
U2 = MC68HC11E9FN
U3 = HIN232CP
U4 = UA7805CV
D1 = 1N4007
Y1 = 8MHz
C1 = 10MICROF
C2 = 0.1MICROF
C3 = 18PF
C4 = 18PF

C5 = 1MICROF
C6 = 100MICROF
C7 = 0.1MICROF
C8 = 0.1MICROF
C9 = 10MICROF
C10 = 1MICROF
C11 = 1MICROF
C12 = 1MICROF
C13 = 1MICROF
C14 = 1MICROF

R1 = 10K
R2 = 10M
R3 = 10K
R4 = 10K
R5 = 10K
R6 = 10K
R7 = 10K
R8 = 10K
R9 = 10K
R10 = 1K

R11 = 10K
R12 = 10K
R13 = 10K
R14 = 10K
R15 = 10K
R16 = 10K
R17 = 10K
R18 = 10K
R19 = 10K
R20 = 10K

ENIC TITO SMAILAGICH

29. NOVEMBRA 20/21
YU-11000 BELGRADE
YUGOSLAVIA
Phone: +381 11 3226-387

Title		
ENIC EVB14-AL		
Size: A	Document Number: 2/98	REV: 1
Date: April 10, 1998	Sheet 1 of 1	

PRECIZNI OBRTOMER SA MIKROKONTROLEROM 68HC11

Tito Smailagic, inzenjer informatike

Pravi smisao koriscenja mikrokontrolera u uredjajima je kada mikrokontroler radi kao SINGLE CHIP. Taj nacin rada mu omogucava maksimalno koriscenje svih njegovih resursa, koji ga cine mikrokontrolerom. Kod mikrokontrolera 68HC11 za to mozemo koristiti NORMAL SINGLE CHIP mod ili SPECIAL BOOTSTRAP mod.

Kod modela A1 i E1 program koji se izvrsava iz NORMAL SINGLE CHIP moda mora da se nalazi u ROM memoriji, ciji sadrzaj se programira pri proizvodnji. Da bi mikrokontroler koristili za nasu aplikaciju program moramo da smestimo u EEPROM memoriju, koja se nalazi na adresama od $B600 do $B7FF. Skok u ovu memoriju na adresu $B600 je sigurno moguc iz SPECIAL BOOTSTRAP moda na dva nacina: spajanjem PD0/RXD sa PD1/TXD, ili slanjem odredjenog koda na PD0/RXD. Ovaj skok iz NORMAL SINGLE CHIP moda je moguc samo ukoliko to dozvoljava program iz ROM memorije. Na primer kod BUFFALO monitora taj skok je moguc kada PE0 postavimo na visoki nivo.

Kod modela F1 program koji se izvrsava iz NORMAL SINGLE CHIP moda, ili iz SPECIAL BOOTSTRAP moda mora da se nalazi u EEPROM memoriji, koja se u ovim modovima uvek nalazi na adresama od $FE00 do $FFFF. Kod SPECIAL BOOTSTRAP moda taj program mora da pocinje i da se izvrsava od adrese $FE00, jer je to adresa skoka kada su PD0/RXD i PD1/TXD spojani, odnosno kada se posalje odredjeni kod na PD0/RXD. Kod NORMAL SINGLE CHIP moda program moze da pocinje na bilo kojoj memorijskoj lokaciji, a njegova adresa izvrsavanja mora biti upisana na adrasama $FFFE i $FFFF kao RESET vektor.

Jedna od velikih prednosti koriscenja mikrokontrolera 68HC11 nad drugima je programiranje u aplikaciji, sto znaci da nam za programiranje EEPROM memorije nije potreban nikakav poseban programator, vec nam je za to dovoljna serijska veza sa PC racunarom i odgovarajuci softverski paket, na primer PCBUG11.

Kao primer konkretne aplikacije koriscenja ovog mikrokontrolera u SINGLE CHIP modu dat je precizni obrtomer.

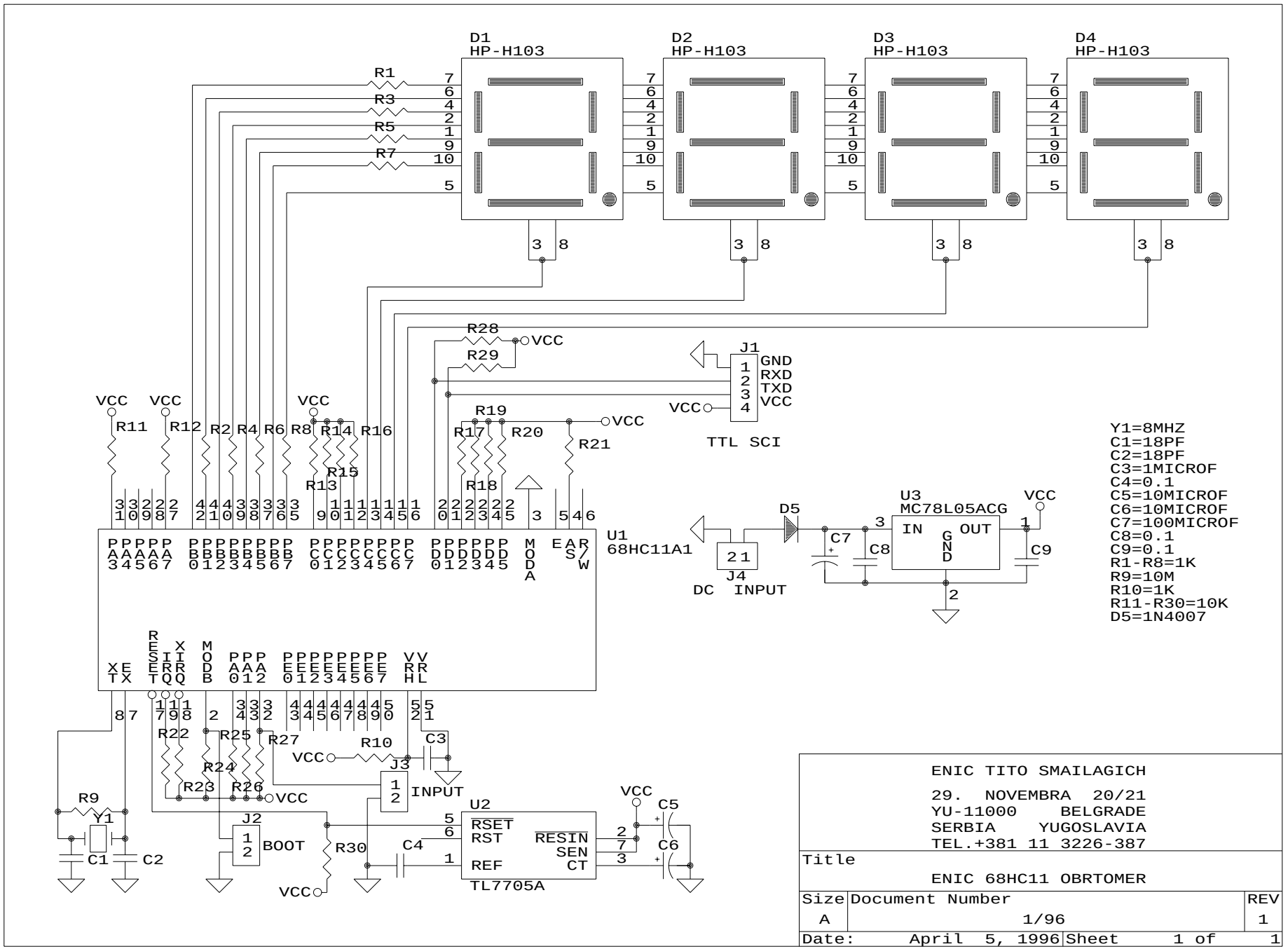
Za ulaz u obrtomer se koristi port A kao tajmer IC1 funkcija sa pracenjem pozitivne ivice impulsa. Na njega se dovodi standardni TTL impuls za svaki mereni obrtaj preko konektora J3. Taj impuls mozemo dobiti sa bilo koje merne sonde: mehanicke, opticke ili induktivne, ali ga prethodno moramo dovesti na TTL nivo.

Za izlaz se koriste cetiri sedmo-segmentna HP LED displeja sa izuzetno malom potrosnjom, oko 1mA po segmentu. Svi segmenti su im paralelno povezani preko otpornika R1-R8 na port B mikrokontrolera, a digiti su im direktno vezani na port C mikrokontrolera. Prikaz izmerene velicine se vrsi u opsegu od 0 do 9999 obrtaja u minuti sa automatskim gasenjem levih nula. Prikaz se odnosi na prethodni ciklus merenja i odnosi se na svaki ciklus bez preskakanja, tako da se moze koristi kao regulacioni kontroler sa velikom tacnoscu.

Za napajanje obrtomera dovoljno je da dovedemo jednosmeran napon u opsegu od 9V do 15V na konektor J4. Maksimalna potrosnja celog uredjaja je oko 50mA, tako da je predvidjen i za baterijsko napajanje, kao prenosni uredjaj.

Konektor J1 je predvidjen za serijsku vezu mikrokontrolera sa PC racunarom preko RS232 interfejsa: MAX232, MAX233, ili bilo kog drugog. Ova veza je potrebna radi programiranja i razvoja softvera za obrtomer. Ako zelimo da koristimo obrtomer bez ove veze moramo da postavimo kratkospojnik izmedju srednja dva pina konektora J1: RXD i TXD. Na taj nacin smo omogucili direktan skok na adresu EEPROM memorije $B600, gde nam se nalazi nas program.
Na konektor J2 obavezno moramo postaviti kratkospojnik, koji postavlja ovaj mikrokontroler u SPECIAL BOOTSTRAP mod.

Program je uradjen u asembleru i smesten je u EEPROM memoriju mikrokontrolera MC68HC11A1. Zauzima manje od raspolozivih 512 bajtova, tako da se lako moze modifikovati za dodatne funkcije.



ENIC TITO SMAILAGICH

29. NOVEMBRA 20/21
YU-11000 BELGRADE
SERBIA YUGOSLAVIA
TEL.+381 11 3226-387

Title	ENIC 68HC11 OBRTOMER	
Size: A	Document Number: 1/96	REV: 1
Date: April 5, 1996	Sheet 1 of 1	

POVEZIVANJE ALFANUMERICKIH LCD DISPLEJ MODULA SA MIKROKONTROLERIMA 68HC11

Tito Smailagic, inzenjer informatike

Zahvaljujuci izuzetno brzom razvoju tehnologije izrade elektronskih komponenti na trzistu se pojavio veliki broj jeftinih inteligentnih displeja u obliku LCD modula, sto je omogucilo njihovu masovnu primenu. Da bi smo ih primenili u bilo kojoj aplikaciji moramo da uspostavimo komunikaciju preko nekog mikrokonrolera. Moramo da znamo instrukcije kojima ti moduli raspolazu, posto na sebi imaju kontrolere, koji upravljaju hardverskim komponentama displeja.

U ovom clanku dajemo primer povezivanja LCD displej modula koji ima dva reda po 16 karaktera. Dajemo primere povezivanja u single chip modu i expanded modu. Obradicemo vezu sa tri modela mikrokontrolera 68HC11: A1, E1 i F1. Veza sa ostalim tipovima ovog mikrokontrolera je ista ili veoma slicna. Uz malu prepravku jednostavno se moze uraditi i 4-bitna veza, posto ovi moduli imaju i tu mogucnost, sto nam oslobadja jos 4 I/O veze mikrokontrolera.

Iz datih listinga se vidi fizicka veza izmedju mikrokontrolera i displeja.

U expanded modu displej je povezan na odredjene adrese mikrokontrolera. U toj vezi on zahteva dve adrese, koje se razdvajaju signalom A0. Bazna adresa se dobija mesanjem ostalih adresa A1 do A15 sa signalom E procesora. Ovako pomesan signal se dovodi na E displeja, koji je pozitivno aktivan. Kontrolna veza R/W je direktno povezana sa displejom. Kod povezivanja mikrokontrolera 68HC11F1 dekoderska logika se nalazi u samom kontroleru, tako da se IOCS2/PG4 direktno vezuje sa displejom i softverski se kontrolise.

U single chip modu displej je povezan kao periferija preko portova B i C mikrokontrolera i softverski se kontrolise.

Treba naglasiti da postoje moduli sa reflektivnim displejom koji su vidljivi na obicnom svetlu, kao i moduli sa transreflektivnim displejom, koji u svom sklopu imaju pozadinsko svetlo, koje im omogucava odlicnu vidljivost bez obzira na osvetljenost prostorije u kojoj se nalaze. Za izvor pozadinskog svetla se moze koristiti svetlost LED dioda, elektrolimunensne lampe ili fluoroscentne lampe. Za svoj rad LED diode zahtevaju normalan napon napajanja modula uz znatno vecu potrosnju struje, dok elektrolimunensne i fluoroscentne lampe zahtevaju veliki napon uz malu potrosnju struje, pa im je neophodan dodatni pretvarac napona.

Ukljucenje pozadinskog svetla se moze kontrolisati dodatnom izlaznom linijom mikrokontrolera uz pomoc dodatnog tranzistora kao drajvera.

```
0001                                * LCDA1.ASM
0002                                * OVAJ PROGRAM SE ODNOSI NA VEZU LCD DISPLEJA SA MIKROKONTROLEROM MC68HC11A1
0003                                * ILI MC68HC11E1 U EXPANDED MODU.
0004
0005                                * LCD DISPLAY MODULE SEIKO INSTRUMENTS M1632
0006
0007                                * DB0        D0
0008                                * DB1        D1
0009                                * DB2        D2
0010                                * DB3        D3
0011                                * DB4        D4
0012                                * DB5        D5
0013                                * DB6        D6
0014                                * DB7        D7
0015
0016                                * RS         A0
0017                                * R/W        R/W
0018                                * E  ZELJENA ADRESA POMESANA SA CPU E SIGNALOM
0019
0020 4000                           ADR     EQU     $4000
0021 4000                           LCDIR   EQU     ADR             * LCD INSTRUCTION REGISTER WRITE ONLY
0022 4000                           LCDBF   EQU     ADR             * LCD BUSY FLAG READ ONLY
0023 4001                           LCDDR   EQU     ADR+1           * LCD DATA REGISTER
0024
0025 8000                                   ORG     $8000
0026
0027 8000                           DISBUF  RMB     32              * DISPLAY BUFFER
0028
0029 8100                                   ORG     $8100
0030
0031 8100 86 38             [ 2 ] START   LDAA    #%00111000      * FUNCTION SET: 8 BIT INTERFACE
0032 8102 8d 35             [ 6 ]         BSR     STORE
0033 8104 86 01             [ 2 ]         LDAA    #%00000001      * DISPLAY CLEAR: CLEAR SCREEN
0034 8106 8d 31             [ 6 ]         BSR     STORE
0035 8108 86 0c             [ 2 ]         LDAA    #%00001100      * DISPLAY ON/OFF CONTROL:
0036 810a 8d 2d             [ 6 ]         BSR     STORE           * DISPLAY ON, CURSOR OFF, BLINK OFF
0037 810c 86 06             [ 2 ]         LDAA    #%00000110      * ENTRY MODE SET:
0038 810e 8d 29             [ 6 ]         BSR     STORE           * CURSOR INCREMENT, NO DISPLAY SHIFT
0039 8110 ce 80 00          [ 3 ]         LDX     #DISBUF         * PUNI DISBUF SA ZELJENIM SADRZAJEM
0040 8113 86 30             [ 2 ]         LDAA    #$30            * ASCII CHARACTER 0
0041 8115 c6 20             [ 2 ]         LDAB    #32             * COUNTER
0042 8117 a7 00             [ 4 ] SP1     STAA    0,X
0043 8119 08                [ 3 ]         INX
0044 811a 4c                [ 2 ]         INCA                    * NEXT CHARACTER
0045 811b 5a                [ 2 ]         DECB
0046 811c 26 f9             [ 3 ]         BNE     SP1
0047 811e ce 80 00          [ 3 ] OUTDIS  LDX     #DISBUF
0048 8121 86 80             [ 2 ]         LDAA    #%10000000      * DD RAM ADDRESS SET:
0049 8123 8d 14             [ 6 ]         BSR     STORE           * IZBACUJE PRVI RED NA DISPLEJU
0050 8125 8d 07             [ 6 ]         BSR     OUTROW
0051 8127 86 c0             [ 2 ]         LDAA    #%11000000      * DD RAM ADDRESS SET:
0052 8129 8d 0e             [ 6 ]         BSR     STORE           * IZBACUJE DRUGI RED NA DISPLEJU
0053 812b 8d 01             [ 6 ]         BSR     OUTROW
0054 812d 3f                [14 ]         SWI                     * SWI = KRAJ PROGRAMA
0055
0056 812e c6 10             [ 2 ] OUTROW  LDAB    #16
0057 8130 a6 00             [ 4 ] L1      LDAA    0,X
0058 8132 08                [ 3 ]         INX
0059 8133 8d 0d             [ 6 ]         BSR     WRITE
0060 8135 5a                [ 2 ]         DECB
0061 8136 26 f8             [ 3 ]         BNE     L1
0062 8138 39                [ 5 ]         RTS
0063
0064 8139 7d 40 00          [ 6 ] STORE   TST     LCDBF           * TEST LCD BUSY FLAG
0065 813c 2b fb             [ 3 ]         BMI     STORE
0066 813e b7 40 00          [ 4 ]         STAA    LCDIR           * STORE INSTRUCTION TO DISPLAY
0067 8141 39                [ 5 ]         RTS
0068
0069 8142 7d 40 00          [ 6 ] WRITE   TST     LCDBF           * TEST LCD BUSY FLAG
0070 8145 2b fb             [ 3 ]         BMI     WRITE           * DATA WRITE TO DD RAM
0071 8147 b7 40 01          [ 4 ]         STAA    LCDDR           * STORE DATA TO DISPLAY
0072 814a 39                [ 5 ]         RTS
0073
0074                                      END

ADR              4000 *0020 0021 0022 0023
DISBUF           8000 *0027 0039 0047
L1               8130 *0057 0061
LCDBF            4000 *0022 0064 0069
LCDDR            4001 *0023 0071
```

```
LCDIR            4000 *0021 0066
OUTDIS           811e *0047
OUTROW           812e *0056 0050 0053
SP1              8117 *0042 0046
START            8100 *0031
STORE            8139 *0064 0032 0034 0036 0038 0049 0052 0065
WRITE            8142 *0069 0059 0070


Number of errors 0
```

```
0001                                * LCDE1.ASM
0002                                * OVAJ PROGRAM SE ODNOSI NA VEZU LCD DISPLEJA SA MIKROKONTROLEROM MC68HC11E1
0003                                * U SINGLE CHIP MODU.
0004
0005                                * LCD DISPLAY MODULE SEIKO INSTRUMENTS M1632
0006
0007                                * DB0        PC0     I/O
0008                                * DB1        PC1     I/O
0009                                * DB2        PC2     I/O
0010                                * DB3        PC3     I/O
0011                                * DB4        PC4     I/O
0012                                * DB5        PC5     I/O
0013                                * DB6        PC6     I/O
0014                                * DB7        PC7     I/O
0015
0016                                * RS         PB0     OUT
0017                                * R/W-       PB1     OUT
0018                                * E          PB2     OUT
0019
0020 1000                           REG     EQU     $1000
0021 1003                           PORTC   EQU     REG+$03
0022 1004                           PORTB   EQU     REG+$04
0023 1007                           DDRC    EQU     REG+$07
0024
0025 0000                                   ORG     $0000
0026
0027 0000                           DISBUF  RMB     32                      * DISPLAY BUFFER
0028
0029 0100                                   ORG     $0100
0030
0031 0100 86 38             [ 2 ] START   LDAA    #%00111000              * 8 BIT INTERFACE
0032 0102 8d 35             [ 6 ]         BSR     STORE
0033 0104 86 01             [ 2 ]         LDAA    #%00000001              * CLEAR SCREEN
0034 0106 8d 31             [ 6 ]         BSR     STORE
0035 0108 86 0c             [ 2 ]         LDAA    #%00001100              * DISPLAY ON, CURSOR OFF, BLINK OFF
0036 010a 8d 2d             [ 6 ]         BSR     STORE
0037 010c 86 06             [ 2 ]         LDAA    #%00000110              * CURSOR INCREMENT, NO DISPLAY SHIFT
0038 010e 8d 29             [ 6 ]         BSR     STORE
0039 0110 ce 00 00          [ 3 ]         LDX     #DISBUF                 * PUNI DISBUF SA ZELJENIM SADRZAJEM
0040 0113 86 30             [ 2 ]         LDAA    #$30                    * ASCII KARAKTER 0
0041 0115 c6 20             [ 2 ]         LDAB    #32                     * BROJAC KARAKTERA U BAFERU
0042 0117 a7 00             [ 4 ] SP1     STAA    0,X
0043 0119 08                [ 3 ]         INX
0044 011a 4c                [ 2 ]         INCA                            * GENERISE SLEDECI ASCII KARAKTER
0045 011b 5a                [ 2 ]         DECB
0046 011c 26 f9             [ 3 ]         BNE     SP1
0047 011e ce 00 00          [ 3 ] OUTDIS  LDX     #DISBUF
0048 0121 86 80             [ 2 ]         LDAA    #%10000000              * ADRESIRA PRVI RED DISPLEJA
0049 0123 8d 14             [ 6 ]         BSR     STORE
0050 0125 8d 07             [ 6 ]         BSR     OUTROW                  * IZBACUJE PRVI RED DISPLEJA
0051 0127 86 c0             [ 2 ]         LDAA    #%11000000              * ADRESIRA DRUGI RED DISPLEJA
0052 0129 8d 0e             [ 6 ]         BSR     STORE
0053 012b 8d 01             [ 6 ]         BSR     OUTROW                  * IZBACUJE DRUGI RED DISPLEJA
0054 012d 3f                [14 ]         SWI                             * SWI = KRAJ PROGRAMA
0055
0056 012e c6 10             [ 2 ] OUTROW  LDAB    #16                     * BROJAC KARAKTERA KOJI SE PRENOSE
0057 0130 a6 00             [ 4 ] L1      LDAA    0,X                     * ISCITAVA KARAKTER IZ BAFERA
0058 0132 08                [ 3 ]         INX
0059 0133 8d 35             [ 6 ]         BSR     WRITE                   * IZBACUJE GA NA DISPLEJ
0060 0135 5a                [ 2 ]         DECB
0061 0136 26 f8             [ 3 ]         BNE     L1
0062 0138 39                [ 5 ]         RTS
0063
0064 0139 3c                [ 4 ] STORE   PSHX                            * KOMANDA ZA DISPLEJ U A AKUMULATORU
0065 013a 37                [ 3 ]         PSHB
0066 013b ce 10 00          [ 3 ]         LDX     #REG
0067 013e 8d 14             [ 6 ]         BSR     WAIT                    * CEKA DA BF POSTANE 0
0068 0140 c6 ff             [ 2 ]         LDAB    #%11111111
0069 0142 f7 10 07          [ 4 ]         STAB    DDRC                    * INICIRA PORT C KAO IZLAZNI PORT
0070 0145 b7 10 03          [ 4 ]         STAA    PORTC                   * SMESTA ZELJENU KOMANDU NA DISPLEJ
0071 0148 1d 04 03          [ 7 ]         BCLR    PORTB-REG,X %00000011   * R/W=0, RS=0
0072 014b 1c 04 04          [ 7 ]         BSET    PORTB-REG,X %00000100   * E=1
0073 014e 1d 04 04          [ 7 ]         BCLR    PORTB-REG,X %00000100   * E=0
0074 0151 33                [ 4 ]         PULB
0075 0152 38                [ 5 ]         PULX
0076 0153 39                [ 5 ]         RTS
0077
0078 0154 7f 10 07          [ 6 ] WAIT    CLR     DDRC                    * INICIRA PORT C KAO ULAZNI PORT
0079 0157 1d 04 01          [ 7 ]         BCLR    PORTB-REG,X %00000001   * RS=0
0080 015a 1c 04 02          [ 7 ]         BSET    PORTB-REG,X %00000010   * R/W=1
```

```
0081 015d 1c 04 04           [ 7 ] WAIT1   BSET    PORTB-REG,X %00000100   * E=1
0082 0160 f6 10 03           [ 4 ]         LDAB    PORTC                   * ISCITAVA STATUS DISPLEJA
0083 0163 1d 04 04           [ 7 ]         BCLR    PORTB-REG,X %00000100   * E=0
0084 0166 5d                 [ 2 ]         TSTB
0085 0167 2b f4              [ 3 ]         BMI     WAIT1                   * VRTI SE U PETLJI SVE DOK JE BF=1
0086 0169 39                 [ 5 ]         RTS
0087
0088 016a 3c                 [ 4 ] WRITE   PSHX                            * PODATAK ZA DISPLEJ U A AKUMULATORU
0089 016b 37                 [ 3 ]         PSHB
0090 016c ce 10 00           [ 3 ]         LDX     #REG
0091 016f 8d e3              [ 6 ]         BSR     WAIT
0092 0171 c6 ff              [ 2 ]         LDAB    #%11111111              * INICIRA PORT C KAO IZLAZNI PORT
0093 0173 f7 10 07           [ 4 ]         STAB    DDRC
0094 0176 b7 10 03           [ 4 ]         STAA    PORTC                   * SMESTA PODATAK NA DISPLEJ
0095 0179 1c 04 01           [ 7 ]         BSET    PORTB-REG,X %00000001   * RS=1
0096 017c 1d 04 02           [ 7 ]         BCLR    PORTB-REG,X %00000010   * R/W=0
0097 017f 1c 04 04           [ 7 ]         BSET    PORTB-REG,X %00000100   * E=1
0098 0182 1d 04 04           [ 7 ]         BCLR    PORTB-REG,X %00000100   * E=0
0099 0185 33                 [ 4 ]         PULB
0100 0186 38                 [ 5 ]         PULX
0101 0187 39                 [ 5 ]         RTS
0102
0103                                       END

DDRC             1007 *0023 0069 0078 0093
DISBUF           0000 *0027 0039 0047
L1               0130 *0057 0061
OUTDIS           011e *0047
OUTROW           012e *0056 0050 0053
PORTB            1004 *0022 0071 0072 0073 0079 0080 0081 0083 0095 0096
                      0097 0098
PORTC            1003 *0021 0070 0082 0094
REG              1000 *0020 0021 0022 0023 0066 0071 0072 0073 0079 0080
                      0081 0083 0090 0095 0096 0097 0098
SP1              0117 *0042 0046
START            0100 *0031
STORE            0139 *0064 0032 0034 0036 0038 0049 0052
WAIT             0154 *0078 0067 0091
WAIT1            015d *0081 0085
WRITE            016a *0088 0059


Number of errors 0
```

```
0001                                 * LCDF1.ASM
0002                                 * OVAJ PROGRAM SE ODNOSI NA VEZU LCD DISPLEJA SA MIKROKONTROLEROM MC68HC11F1
0003                                 * U EXPANDED MODU.
0004
0005                                 * LCD DISPLAY MODULE SEIKO INSTRUMENTS M1632
0006
0007                                 * DB0        D0
0008                                 * DB1        D1
0009                                 * DB2        D2
0010                                 * DB3        D3
0011                                 * DB4        D4
0012                                 * DB5        D5
0013                                 * DB6        D6
0014                                 * DB7        D7
0015
0016                                 * RS         A0
0017                                 * R/W        R/W
0018                                 * E          IOCS2/PG4 MC68HC11F1
0019
0020 1000                            REG     EQU     $1000
0021 105c                            CSSTRH  EQU     REG+$5C         * CHIP SELECT CLOCK STRETCH REGISTER
0022 105d                            CSCTL   EQU     REG+$5D         * CHIP SELECT CONTROL REGISTER
0023 105e                            CSGADR  EQU     REG+$5E         * GENERAL PURPOSE CHIP SELECT ADDRESS REGISTER
0024 105f                            CSGSIZ  EQU     REG+$5F         * GENERAL PURPOSE CHIP SELECT SIZE REGISTER
0025 1800                            LCDIR   EQU     REG+$0800       * LCD INSTRUCTION REGISTER WRITE ONLY
0026 1800                            LCDBF   EQU     REG+$0800       * LCD BUSY FLAG READ ONLY
0027 1801                            LCDDR   EQU     REG+$0801       * LCD DATA REGISTER
0028
0029 0800                                    ORG     $0800
0030
0031 0800                            DISBUF  RMB     32              * DISPLAY BUFFER
0032
0033 2000                                    ORG     $2000
0034
0035 2000 86 10             [ 2 ] START  LDAA    #%00010000      * IOCS2=1 CYCLE CLOCK STRETCH
0036 2002 b7 10 5c          [ 4 ]        STAA    CSSTRH
0037 2005 86 34             [ 2 ]        LDAA    #%00110100      * IOCS2=ENABLE, ACTIVE HIGH
0038 2007 b7 10 5d          [ 4 ]        STAA    CSCTL
0039 200a 86 07             [ 2 ]        LDAA    #%00000111      * IOCS2=ADDRESS VALID DURING E CLOCK VALID TIME
0040 200c b7 10 5f          [ 4 ]        STAA    CSGSIZ
0041 200f 86 38             [ 2 ]        LDAA    #%00111000      * FUNCTION SET: 8 BIT INTERFACE
0042 2011 8d 35             [ 6 ]        BSR     STORE
0043 2013 86 01             [ 2 ]        LDAA    #%00000001      * DISPLAY CLEAR: CLEAR SCREEN
0044 2015 8d 31             [ 6 ]        BSR     STORE
0045 2017 86 0c             [ 2 ]        LDAA    #%00001100      * DISPLAY ON/OFF CONTROL:
0046 2019 8d 2d             [ 6 ]        BSR     STORE           * DISPLAY ON, CURSOR OFF, BLINK OFF
0047 201b 86 06             [ 2 ]        LDAA    #%00000110      * ENTRY MODE SET:
0048 201d 8d 29             [ 6 ]        BSR     STORE           * CURSOR INCREMENT, NO DISPLAY SHIFT
0049 201f ce 08 00          [ 3 ]        LDX     #DISBUF         * INICIRA DISBUF
0050 2022 86 30             [ 2 ]        LDAA    #$30            * CHARACTER 0
0051 2024 c6 20             [ 2 ]        LDAB    #32             * COUNTER
0052 2026 a7 00             [ 4 ] SP1    STAA    0,X
0053 2028 08                [ 3 ]        INX
0054 2029 4c                [ 2 ]        INCA                    * NEXT CHARACTER
0055 202a 5a                [ 2 ]        DECB
0056 202b 26 f9             [ 3 ]        BNE     SP1
0057 202d ce 08 00          [ 3 ] OUTDIS LDX     #DISBUF
0058 2030 86 80             [ 2 ]        LDAA    #%10000000      * DD RAM ADDRESS SET:
0059 2032 8d 14             [ 6 ]        BSR     STORE           * IZBACUJE PRVI RED NA DISPLEJU
0060 2034 8d 07             [ 6 ]        BSR     OUTROW
0061 2036 86 c0             [ 2 ]        LDAA    #%11000000      * DD RAM ADDRESS SET:
0062 2038 8d 0e             [ 6 ]        BSR     STORE           * IZBACUJE DRUGI RED NA DISPLEJU
0063 203a 8d 01             [ 6 ]        BSR     OUTROW
0064 203c 3f                [14 ]        SWI                     * SWI = KRAJ PROGRAMA
0065
0066 203d c6 10             [ 2 ] OUTROW LDAB    #16
0067 203f a6 00             [ 4 ] L1     LDAA    0,X
0068 2041 08                [ 3 ]        INX
0069 2042 8d 0d             [ 6 ]        BSR     WRITE
0070 2044 5a                [ 2 ]        DECB
0071 2045 26 f8             [ 3 ]        BNE     L1
0072 2047 39                [ 5 ]        RTS
0073
0074 2048 7d 18 00          [ 6 ] STORE  TST     LCDBF           * TEST LCD BUSY FLAG
0075 204b 2b fb             [ 3 ]        BMI     STORE
0076 204d b7 18 00          [ 4 ]        STAA    LCDIR           * STORE INSTRUCTION TO DISPLAY
0077 2050 39                [ 5 ]        RTS
0078
0079 2051 7d 18 00          [ 6 ] WRITE  TST     LCDBF           * TEST LCD BUSY FLAG
0080 2054 2b fb             [ 3 ]        BMI     WRITE           * DATA WRITE TO DD RAM:
```

```
0081 2056 b7 18 01           [ 4 ]         STAA    LCDDR           * STORE DATA TO DISPLAY
0082 2059 39                 [ 5 ]         RTS
0083
0084                                       END

CSCTL            105d *0022 0038
CSGADR           105e *0023
CSGSIZ           105f *0024 0040
CSSTRH           105c *0021 0036
DISBUF           0800 *0031 0049 0057
L1               203f *0067 0071
LCDBF            1800 *0026 0074 0079
LCDDR            1801 *0027 0081
LCDIR            1800 *0025 0076
OUTDIS           202d *0057
OUTROW           203d *0066 0060 0063
REG              1000 *0020 0021 0022 0023 0024 0025 0026 0027
SP1              2026 *0052 0056
START            2000 *0035
STORE            2048 *0074 0042 0044 0046 0048 0059 0062 0075
WRITE            2051 *0079 0069 0080


Number of errors 0
```