UNIVERZA V LJUBLJANI
Fakulteta za elektrotehniko
in računalništvo

Študijsko leto 1994/95

# Pralni stroj

(Seminarska naloga pri predmetu Mikroprocesorji v elektroniki)

## 1. Uvod:

Program je nastal v okviru laboratorijskih vaj pri predmetu Mikroprocesorji v elektroniki. Program krmili pralni stroj, poleg tega pa izpisuje trenutno operacijo na PC terminal, prav tako pa s PCja bere podatke o stanju senzorjev. Nalogo sva si razdelila skupaj s kolegom, moj del je bil komunikacija s PCjem (*SCI rutina*: Serial Comunication Interface)

## 2. Opis delovanja, scheduler:

Program je sestavljen iz glavnega programa, krmilnika motorja in podprograma za komunikacijo s PCjem (SCI). Vsi podprogrami so vstavljeni v *scheduler*. Scheduler skrbi za delitev procesorskega časa med izvajanje glavnega programa in izvajanje podprogramov. Tako vsake toliko časa prekine izvajanje glavnega programa in izvede podprogram, s čemer omogoči navidezno večopravilnost. Scheduler razdeli procesorski čas na 1/64 sekundne rezine in v vsaki izvede vse podprograme, zapisane v task listi. Posamezen task ne sme prekoračiti omejitve 1100 procesorskih ciklov, preostali čas pa scheduler dodeli glavnemu programu.

Program za pralni stroj omogoča izbiro dveh “programov” pranja: s predpranjem in brez. Prav tako glede na stanje senzorjev obrača motor, prižiga grelce in odpira ventile. Senzorje smo simulirali s PC terminalom, kamor je program tudi izpisoval potek pranja. Tako je znak “v” pomenil, da je dosežen zadostni nivo vode, znak “t” pa, da je dosežena željena temperatura vode.

Komunikacija med deli programa poteka preko bufferjev. Glavni program bere stanje ventilov iz bufferja BUFREC, kamor SCI rutina pošilja preko PUTB s PCja sprejete znake in pošilja status v BUFTRA, od koder jih bo SCI rutina pobrala z GETB in poslala na terminal - ko bo prišla na vrsto po task listi.

## 3. Rutine GETB in PUTB

Za polnenje in praznenje FIFO bufferja smo napisali rutine PUTB: shrani byte v FIFO buffer in GETB: poberi byte iz FIFO bufferja

```
;____________________________ PUT BIN TO BUFFER (č<=43) _________________________
;  Put contents of A into the fifo buffer at X. In case of a full buffer, the
;  procedure call is without effect (no error is reported). The contents of all
;  registers are preserved.
;-------------------------------------------------------------------------------
PUTB    pshb              ;B register pokvarimo
        ldab    0,X       ;ŠX+0] -> B
        cmpb    #BUFLEN   ;ali je buffer poln?
        bge     PUTEX     ;da, return

        pshx              ;index register na sklad
        incb              ;povecas B
        stab    0,X       ;B -> ŠX+0]

        abx               ;pristejes Xu B
        staa    0,X       ;A -> ŠX+B]
        pulx              ;X dol s sklada

PUTEX   pulb              ;B dol s sklada
        rts               ;Return().

;____________________________ GET BIN FROM BUFFER (č<=32+16*BUFLEN) _______________
;  Get one byte from fifo buffer at X and return it to caller in A. In case of
;  an empty buffer, the procedure returns an unchanged A (no error is reported).
```

```
;  The contents of all registers but A are preserved.
;-------------------------------------------------------------------------------
GETB    pshb              ;b register pokvarimo
        ldab    0,X       ;ŠX+0] -> B   (0)
        beq     GETEX     ;ce je nicla skocis ven

        decb              ;zmanjsamo b
        stab    0,X       ;(B-1) -> ŠX+0]

        pshx              ;index register na sklad

        inx               ;X kaze na prvo vrednost
        ldaa    0,X       ;ŠX+0] -> A   (1)
        psha              ;se A na stack, ker ga potem rabimo


GELOOP  beq     GELPEND   ;ce je 0 skoci iz zanke
        ldaa    1,X       ;ŠX+1] -> A
        staa    0,X       ;A -> ŠX]
        inx               ;povecamo X

        decb              ;zmanjsamo stevec
        bra     GELOOP

GELPEND pula              ;A dol s sklada
        pulx              ;X dol s sklada
GETEX   pulb              ;B dol s sklada
        rts               ;Return(A).
```

## *4. Rutina SCI*

Potrebna oprema za komunikacijo preko RS232 je že vgrajena v MC6803, dodati je treba samo prilagodilnik nivojev iz TTL na RS232C standard.

Serijski V/I najprej inicializiramo na željeno hitrost z vpisom v *Rate and Mode control register.* Podprogram SCI skrbi za sprejem in oddajo bytov - sprejeti znak vpiše preko PUTB na sprejemni FIFO buffer BUFREC, oddani znak pa preko GETB pobere z vrha FIFO bufferja BUFTRA.

Sprejem:

- program prebere Transmit/Receive Control and Status Register
- RDRF bit v logični “1” pomeni, da je procesor sprejel byte
- preverimo, če je pri prenosu prišlo do framing errorja (ORFE v log. “1”)
- preberemo podatek in ga shranimo s proceduro PUTB v BUFREC
- PUTB preverja, če je še dovolj prostora v bufferju, v primeru, da ga ni, podatek zavržemo.

Oddaja:

- preverimo TDRE bit: “0” pomeni, da je oddajni register pripravljen na oddajo.
- preverimo, če je v BUFTRA pripravljen byte za prenos
- preberemo ga s proceduro GETB vpišemo v transmit register in pošljemo po RS232.

```
;_____________________________ SERIAL COMM. TASK (č<=91+16*BUFLEN) _____________________
;  This is a real-time serial communication driver. SCI should be placed into
;  the task schedule at a 1/64 second duty cycle. In each cycle, SCI checks
;  the serial communication hardware to see if a new byte has been received.
;  If so, the byte is transferred via PUTB into the fifo buffer BUFREC. The
;  transmission register is also checked. If it is empty and the transmission
;  buffer BUFTRA is not empty, then one byte from the fifo buffer is
;  transferred via GETB to the transmission hardware.
;-------------------------------------------------------------------------------
SCI     ldab    _TRCSR          ;Preberi status register v B;
        bpl     SEND            ;(bit7=0) - podatek ni sprejet.
```

```
        aslb                    ;ORFE (Overrun Framing Error)
        bmi     SEND            ;Ce je 1, podatek ni pravilen.

        ldaa    _RDR            ;Receive -> A
        ldx     #BUFREC         ;
        jsr     PUTB            ;Shrani sprejeti znak v BUFREC.

SEND    ldaa    _TRCSR
        anda    #%00100000      ;Preveri TDRE (Transmit data reg. empty)
        beq     SCIEX           ;Ce je 0, skoci ven.

        ldaa    BUFTRA          ;Ali je BUFTRA prazen?
        beq     SCIEX           ;Ja, skoci ven.

        ldx     #BUFTRA
        jsr     GETB            ;Poberi byte iz BUFTRA.
        staa    _TDR            ;Poslji ga.

SCIEX  rts                     ;Return().
```

SCI proceduro vpišemo v scheduler, da se izvaja v ozadju glavnega programa.

```
;_____________________________ TASK SCHEDULE ____________________________________
;  The crystal frequency 4.9152 MHz is internally devided by 4, rendering a
;  1/1228800s maschine cycle. The scheduler uses a constant time slice of 1200
;  maschine cycles, which is exactly 1/1024s.
;
;  Tasks are not allowed to exceed ONE time slice, including the scheduler
;  overhead of approximately 100 cycles. Each 1/64s period task must therefore
;  be completed in 1200-100=1100 machine cycles!
;-------------------------------------------------------------------------------
ORG   _EPROM      ;Must start at $xx00!
SCHTAB
        ...
FDB   SCI         ;Serial communication interface.
        ...
```

Ob resetu moramo tudi poskrbeti za pravilne nastavitve, ki jih zahteva RS232. Določiti moramo način komunikacije in hitrost prenosa podatkov.

```
        ldaa    #%00000101      ;Internal baud generator, 9600 bps
        staa    _RMCR           ;kontrolno besedo v RMCR

        ldaa    #%00001010      ;Transmit & Receive enable;
        staa    _TRCSR          ;vpisi v TRCSR
```

## *5. Rutina KBD:*

Poleg GETB in PUTB smo napisali tudi krmilnik tipkovnice: rutino KBD:

```
;_____________________________ KEYBOAD SCANNING TASK (č<=240) ____________________
;  This is a real-time keyboard driver. It should be placed into the task
;  schedule at a 1/64 second duty cycle. In each cycle, KBD scans all three
;  physical keyboard rows. Multiple key strokes are ignored. The single key
;  scan code is translated to ASCII and placed via PUTB into the fifo buffer
;  BUFKEY. The global boolean variable KBDKEY is used to prevent KBD from
;  reading the same key several times!
;-------------------------------------------------------------------------------

TABELA  FCB     $2E,$33,$36,$39,$2A,$31,$34,$37
        FCB     $30,$32,$35,$38,$73,$72,$71,$70
        FCB     $0D,$2D,$2B,$08,$00,$23,$03,$1B


;--------------------vstopna tocka v KBD rutino------------------
```

```
KBD     ldab    #$00       ;v b bo stevilka vrstice
        ldaa    _P1DR      ;Preberi PIO port 1 data
        anda    #%00011111 ;selektirana SAMO 1. kbd vrstico (D7)
        oraa    #%01100000
        staa    _P1DR      ;vpisi na port

        ldaa    _KBDD      ;Preberi keyboard
        coma               ;Obrni bite
        bne     KBSEND

; *** 2. vrstica ***

        incb               ;povecaj b
        ldaa    _P1DR      ;Preberi PIO port 1 data
        anda    #%00011111 ;maskiraj 2. kbd vrstico (D6)
        oraa    #%10100000
        staa    _P1DR      ;vpisi na port

        ldaa    _KBDD      ;Preberi keyboard
        coma               ;Obrni bite
        bne     KBSEND

; *** 3. vrstica ***

        incb               ;povecaj B
        ldaa    _P1DR      ;Preberi PIO port 1 data
        anda    #%00011111 ;maskiraj 3. kbd vrstico (D5)
        oraa    #%11000000
        staa    _P1DR      ;vpisi na port

        ldaa    _KBDD      ;Preberi keyboard
        coma               ;Obrni bite
        beq     KBOUT    ;Keyboard data=0, ni pritiska

KBSEND  tst     KBDKEY
        bne     KBDEX      ;<> 0 prejsnja tipka se pritisnjena
        inc     KBDKEY

        aslb               ;Procedua izracuna ASCII kodo
                           ;A: stolpec, B: vrstica
        aslb
        aslb               ;to je mnozenje B z 8

        ldx     #TABELA    ;v X zacetek tabele
        abx                ;Xu pristejes B

        ldab    #0         ;b bo stevec

KBCNT   ;*** stevec, pristejes bit, ki je aktiven v a ***
        asra               ;shift right A -> C
        bcs     KBSES      ;ce je carry
        incb               ;povecaj stevec
        bra     KBCNT      ;nazaj

KBSES   abx                ;pristejes Xu stevec
        ldaa    0,X        ;v A vrednost s tabele
        ldx     #BUFKEY    ;v index reg. das naslov BUFKEY.
        jsr     PUTB       ;klices PUTB
        bra     KBDEX      ;gres ven.

KBOUT   tst    KBDKEY    ;Primerjaj KBDKEY z 0
        beq     KBDEX      ;je, skocis ven
        clr     KBDKEY     ;zbrisi flag, ce je tipka spuscena

KBDEX   rts                ;Return().
```

## *6. Priloge:*

- komentirana izvorna koda programa