Střední Průmyslová Škola Elektrotechnická Havířov	**Protokol do MIT**	Třída: 3.C
		Skupina: 3
Minutka s LCD		Zpráva číslo: 4
		Dne: 20.03.2006
		Soupis použitých přístrojů: přípravek s µC 8051 přípravek LCD přípravek s tlačítky pro vnější přerušení
		Jméno učitele: Ing. Baarová
		Jméno:
		Známka:

ZADÁNÍ:

Napište program MINUTKA pro mikroprocesor 8051. Výchozí stav MINUTKY je 5 minut, po stisku INT0 začne MINUTKA odečítat do 0 minut a pak se zastaví, kdykoli po stisku INT1 se MINUTKA nastaví do výchozího stavu => na displeji bude 5 minut, pak lze znovu spustit stiskem INT0.

TEORIE:

LCD (Liquid Crystal Display) S ŘADIČEM HD44780

Jedná se o zobrazovací jednotku s tekutými krystaly, v našem případě se na displej vlezlo 2x16 znaků. Jádrem LCD je řadič HD44780, což je speciální integrovaný obvod, který řídí činnost dalších obvodů (tzv. Budičů), ty budí jednotlivé segmenty LCD. Řadič umožňuje komunikaci s mikroprocesorem dvěma různými způsoby:

1) 8-bitová datová sběrnice, používá se, když má mikroprocesor adresovou, datovou i řídící sběrnici, řídící signály RS, RW a E jsou generovány přímo mikroprocesorem
2) 4-bitová sběrnice, používá se pro připojení k mikroprocesoru s omezeným počtem I/O portů a není vhodné blokovat celé 2 porty

Při připojení 4-bitovou sběrnicí je 8-bitové slovo rozděleno a do řadiče jde 2x, nejdříve vyšší polovina a potom nižší polovina bytu. Signály RS, RW a E jsou generovány programově.

Počáteční inicializace, řadič obsahuje obvody, které při dodržení správného náběhu napájecího napětí zajišťuje reset LCD a jeho počáteční inicializaci:

1) smazání displeje
2) 8-bitová sběrnice, 1 řádek, font 5x7
3) vypnutí kurzoru, blikání displeje
4) inc adresy a vypnutí posunu displeje

Nejsme-li schopni zajistit náběh napájecího napětí udané výrobcem, musíme provést inicializaci programově.

Řadič LCD obsahuje 2 paměti: DDRAM a CGRAM

Paměť DDRAM má 80 adres, paměť CGRAM je tvořena větší částí, má 7200 bitů. Paměť CGRAM obsahuje předdefinované znaky abecedy LATIN a japonské abecedy. Paměť DDRAM má obsah 512 bitů a je možné do ní uložit 8 vlastních znaků. Každý znak je dán maticí 8x8 bitů. CGRAM tedy obsahuje 256 pozic pro jednotlivé znaky, ty se zadávají jako ASCII kód, který představuje adresu v paměti CGRAM.

SOUBOR MINUTKA.ASM:

```
;LCD P2.4 az P2.7 => datove vodice k LCD
LCD_EN  bit     P2.0
LCD_RS  bit     P2.2

        org     0
        jmp     Init

        org     03h                     ;stiskem INT0 spusti MINUTKU
        setb    tr0
        reti

        org     0Bh                     ;generator sekund
        djnz    r4, EndTimerInt
        mov     r4, #30
        setb    c

EndTimerInt:
        reti

        org     13h                     ;stiskem INT1 nastavi 5 minut na LCD a zastavi MINUTKU
        clr     tr0
        call    FiveMinute              ;vypise na LCD 5 minut a nastavi registry R5, R6 a R7
        reti

Init:
        call    ResetLCD
        mov     a, #00011100b           ;vyber chovani displeje
        call    RidLCD
        call    FiveMinute              ;vypise na LCD 5 minut a nastavi registry R5, R6 a R7
        mov     TMOD, #00000001b        ;vyber casovace 0
        mov     IE, #10000111b          ;povoleni preruseni
        mov     r4, #30                 ;nastevni cca 1s
        clr     c

Main:
        jnc     Main                    ;je sekunda?

        dec     r7
        cjne    r7, #255, R7ToLCD

        mov     r7, #9
        dec     r6
        cjne    r6, #255, R6ToLCD

        mov     r6, #5
        dec     r5
        cjne    R5, #255, R5ToLCD

        call    EmptyToLCD              ;na displej vypise 0 minut
```

```
            clr     c
            clr     tr0

            jmp     Main

R5ToLCD:
            mov     a, #0C3h                ;nastaveni pozice na LCD
            call    RidLCD
            mov     a, r5
            add     a, #030h                ;prevod na ASCII
            call    ZnakLCD                 ;minuta na LCD

R6ToLCD:
            mov     a, #0C5h                ;nastaveni pozice na LCD
            call    RidLCD
            mov     a, r6
            add     a, #030h                ;prevod na ASCII
            call    ZnakLCD                 ;desitky sekund na LCD

R7ToLCD:
            mov     a, #0C6h                ;nastaveni pozice na LCD
            call    RidLCD
            mov     a, r7
            add     a, #030h                ;prevod na ASCII
            call    ZnakLCD                 ;jednotky sekund na LCD

            jmp     Main

FiveMinute:
            mov     DPTR, #Title
            call    TextLCD
            mov     r5, #5
            mov     r6, #0
            mov     r7, #1
            ret

EmptyToLCD:
            mov     DPTR, #Empty
            call    TextLCD
            ret

Title:              db      ' Minutka:    '
                    db      '  5:00 m:s   '
Empty:              db      ' Minutka:    '
                    db      '  0:00 m:s   '

            #include <LCDkit.inc>

            nop
            end
```

SOUBOR LCDKIT.INC:

```
;Generovani casovych prodlev
d2500:
            mov     r6, #100                ;ceka 2,5 s
```

```
zpet2:
        mov     R7, #250
        call    Delay                   ;ceka 25 ms
        djnz    r6, zpet2
        ret

Delay100:
        push    acc

        mov     a, r7
        mov     R7, #48
        djnz    R7, $                   ;ceka 50*2 mikrosekund
        mov     r7, a                   ;obnoveni R
        pop     acc
        ret

;VSTUP: R7 = cas ve stovkach mikrosekund
Delay:
        nop
        call    Delay100
        djnz    R7, Delay
        nop
        ret

;Generovani signalu ENABLE
GenEN:
        setb    LCD_EN                  ;ENable aktivni
        clr     LCD_EN                  ;deaktivace ENable
        ret

GenResEN:
        setb    LCD_EN                  ;aktivace EN
        clr     LCD_EN                  ;deaktivace EN
        mov     R7, #2
        call    Delay                   ;ceka 200 us
        ret

;Reset LCD displeje
ResetLCD:
        push    Acc

        mov     R7, #250
        call    Delay                   ;ceka 25 ms
        mov     p2, #30h
        call    GenResEN                ;generuj ENable pri resetu

        mov     R7, #100
        call    Delay                   ;ceka 10 ms
        mov     p2,#30h
        call    GenResEN

        mov     R7, #2
        call    Delay                   ;ceka 200 us
        mov     p2, #30h
        call    GenResEN
```

```
                mov     p2,#20h
                call    GenResEN
                call    Delay100                ;ceka 100 us

                mov     A, #028h                ;nastavi 4-bitovy rezim
                call    RidLCD                  ;vyslani do LCD jako ridici instrukce
                mov     A, #08h                 ;displej ON, kurzor ON, blikani znaku OFF
                call    RidLCD
                mov     A, #01h                 ;smazani displeje
                call    RidLCD
                mov     R7, #30
                call    Delay                           ;ceka 3 ms

                mov     A, #0Ch
                call    RidLCD
                mov     A, #06h                 ;auto increment, no shift
                call    RidLCD
                pop     Acc
                ret

;Zobrazeni znaku ulozeneho v Acc na displeji
ZnakLCD:
                push    Acc
                anl     A, #0F0h                ;vysilaji se nejprve horni 4 bity

                mov     p2, A
                setb    LCD_RS                  ;jedna se o data pro displej
                call    GenEN                   ;generuj ENable
                pop     Acc

                anl     A, #0Fh                 ;vysilaji se dolni 4 bity
                swap    a
                mov     p2, A
                setb    LCD_RS
                call    GenEN
                call    Delay100                ;ceka 100 us
                ret

;Vyslani dat z Acc do ridiciho registru LCD
RidLCD:
                push    Acc
                anl     A, #0F0h                ;horni 4 bity

                mov     p2, A                           ;generuj ENable
                call    GenEN
                pop     Acc
                anl     A, #0Fh                 ;dolni 4 bity
                swap    A
                mov     p2, A
                call    GenEN                   ;ceka 100 us
                call    Delay100
                ret

;Vypis 2x16-znakoveho textu na LCD. Vstup DPTR = zacatek textu.
TextLCD:
```

```
        mov     A, #80h                 ;adresa 00h v LCD
        call    RidLCD
        mov     R5, #32                 ;pocitadlo 32 znaku

Znaky:
        clr     A
        movc    A, @A+DPTR              ;precteni znaku z pameti
        call    ZnakLCD
        inc     DPTR                                    ;posun na dalsi znak
        cjne    R5, #17, NePul          ;je kurzor jiz za pulkou radku?
        mov     A, #0C0h                ;adresa 40h v LCD
        call    RidLCD

NePul:
        djnz    R5, Znaky
        ret
```

ZHODNOCENÍ:

Při psaní psaní programu se objevili drobné chybičky, ty se ale povedlo zdárně vyřešit. Asi největší problém jsem měl se zapnutím MINUTKY když dočítala do 0 minut. Program napsaný doma jsem si mohl prakticky vyzkoušet, takže jsem neměl větší problémy s realizací ve škole.

MINUTKA by se dala rozšířit i na více minut či hodin. Daly by se přidat taky další tlačítka, například po stisku by zůstala hodnota na displeji, nebo by se dal přidat zvuk, třeba pípání při dosažení 0 minut.