|  |  |  |  |
| --- | --- | --- | --- |
| * + - * 1. Termin zajęć   Czwartek  13:15-15:00 | **Podstawy Techniki Mikroprocesorowej** | | |
| **Osoby wykonujące ćwiczenie:**  Michał Bernacki-Janson, Adam Czekalski | | | **Grupa nr:**  **D** |
| **Tytuł ćwiczenia:**  Timery | | | **Ćwiczenie nr:**  **4** |
| **Data wykonania ćwiczenia** | | 11.05.2023 | **Ocena:** |
| **Data oddania sprawozdania** | | 25.05.2023 |  | |

1. **Wstęp**

Celem laboratorium było zapoznanie się z wbudowanymi timerami mikroprocesora.

Liczenie czasu polega na ustawieniu stałej timera na 50 ms, jeśli wykryjemy 20 przerywań zwiększamy rejestr r7 odpowiedzialny za liczenie sekund. Jeśli r7 będzie równe 60 rejestr ten jest zerowany, a r6 liczący minuty jest inkrementowany. Analogicznie liczone są godziny przechowywane w rejestrze r5. Każda zmiana w jednym z tych rejestrów powoduje ponowne wyświetlenie czasu na ekranie.

1. **Program "zegar" wzbogacony o możliwość "startu”, “stopu", “wznowienia” i „resetu” odmierzania czasu.**

Na początku program czeka na naciśnięcie przycisku „A”, który uruchamia za pierwszym razem timer. Po starcie timera sprawdzamy w pętli czy naciśnięty został klawisz “B”, “C” lub “D”. Po wciśnięciu klawisza B timer zatrzymuje się. Gdy naciśniemy klawisz “C”, timer zostanie wznowiony. Natomiast gdy naciśniemy klawisz “D” timer zostanie zresetowany.

ljmp start

P5 equ 0F8H

P7 equ 0DBH

LCDstatus  equ 0FF2EH       *; adres do odczytu gotowosci LCD*

LCDcontrol equ 0FF2CH       *; adres do podania bajtu sterujacego LCD*

LCDdataWR  equ 0FF2DH       *; adres do podania kodu ASCII na LCD*

// bajty sterujace LCD, inne dostepne w opisie LCD na stronie WWW

#define  HOME     0x80     // put cursor to second line

#define  INITDISP 0x38     // LCD init (8-bit mode)

#define  HOM2     0xc0     // put cursor to second line

#define  LCDON    0x0e     // LCD nn, cursor off, blinking off

#define  CLEAR    0x01     // LCD display clear

// linie klawiatury - sterowanie na port P5

#define LINE\_1      0x7f    // 0111 1111

#define LINE\_2      0xbf    // 1011 1111

#define LINE\_3      0xdf    // 1101 1111

#define LINE\_4      0xef    // 1110 1111

#define ALL\_LINES   0x0f    // 0000 1111

ORG 000BH                   *; obsluga przerwania*

    MOV **TH0**, #3CH           *; przeladowanie*

    MOV **TL0**, #0B0H          *; stalej timera na 50ms*

    DEC **R0**                  *; korekta licznika*

    RETI                    *; powrót z przerwania*

org 0100H

// macro do wprowadzenia bajtu sterujacego na LCD

LCDcntrlWR MACRO x          *; x � parametr wywolania macra � bajt sterujacy*

           LOCAL loop       *; LOCAL oznacza ze etykieta loop moze sie powt�rzyc w programie*

loop: MOV  DPTR,#LCDstatus  *; DPTR zaladowany adresem statusu*

      MOVX **A**,@DPTR          *; pobranie bajtu z biezacym statusem LCD*

      JB   **ACC**.7,loop       *; testowanie najstarszego bitu akumulatora*

*; � wskazuje gotowosc LCD*

      MOV  DPTR,#LCDcontrol *; DPTR zaladowany adresem do podania bajtu sterujacego*

      MOV  **A**, x             *; do akumulatora trafia argument wywolania macra�bajt sterujacy*

      MOVX @DPTR,**A**          *; bajt sterujacy podany do LCD � zadana akcja widoczna na LCD*

      ENDM

// macro do wypisania znaku ASCII na LCD, znak ASCII przed wywolaniem macra ma byc w **A**

LCDcharWR MACRO

      LOCAL tutu            *; LOCAL oznacza ze etykieta tutu moze sie powt�rzyc w programie*

      PUSH **ACC**              *; odlozenie biezacej zawartosci akumulatora na stos*

tutu: MOV  DPTR,#LCDstatus  *; DPTR zaladowany adresem statusu*

      MOVX **A**,@DPTR          *; pobranie bajtu z biezacym statusem LCD*

      JB   **ACC**.7,tutu       *; testowanie najstarszego bitu akumulatora*

*; � wskazuje gotowosc LCD*

      MOV  DPTR,#LCDdataWR  *; DPTR zaladowany adresem do podania bajtu sterujacego*

      POP  **ACC**              *; w akumulatorze ponownie kod ASCII znaku na LCD*

      MOVX @DPTR,**A**          *; kod ASCII podany do LCD � znak widoczny na LCD*

      ENDM

// macro do inicjalizacji wyswietlacza � bez parametr�w

init\_LCD MACRO

         LCDcntrlWR #INITDISP *; wywolanie macra LCDcntrlWR � inicjalizacja LCD*

         LCDcntrlWR #CLEAR    *; wywolanie macra LCDcntrlWR � czyszczenie LCD*

         LCDcntrlWR #LCDON    *; wywolanie macra LCDcntrlWR � konfiguracja kursora*

         ENDM

// funkcja wypisania liczby dla potrzeb zegara

putdigitLCD:    mov b, #10

                div ab              *; uzyskanie cyfry dziesiatek*

                add a, #30H         *; konwersja cyfry na kod ASCII*

                acall putcharLCD

                mov a, b            *; ladowanie cyfry jednosci*

                add a, #30H         *; konwersja na LCD*

                acall putcharLCD

                ret

// funkcja wypisywania znaku na LCD

putcharLCD: LCDcharWR

            ret

// wyznaczanie biezacej wartosci zegara i jego wyswietlanie na LCD

ZEGAR:      INC **R7**              *; licznik sekund*

            MOV **A**, **R7**           *; obsluga sekund*

            CLR C

            SUBB **A**, #60         *; przepelnienie sekund*

            JZ MINUTY

            LCDcntrlWR #HOME    *; wyswietlenie calego zegara*

            MOV **A**, **R5**           *; godziny*

            ACALL putdigitLCD

            MOV **A**, #":"         *; separator*

            ACALL putcharLCD

            MOV **A**, **R6**           *; minuty*

            ACALL putdigitLCD

            MOV **A**, #":"         *; separator*

            ACALL putcharLCD

            MOV **A**, **R7**           *; sekundy*

            ACALL putdigitLCD

            JMP FINAL

MINUTY:     MOV **R7**, #00H        *; zerowanie sekund*

            INC **R6**              *; licznik minut*

            MOV **A**, **R6**           *; obsluga minut*

            CLR C

            SUBB **A**, #60         *; przepelnienie minut*

            JZ GODZINY

            LCDcntrlWR #HOME    *; wyswietlenie calego zegara*

            MOV **A**, **R5**           *; godziny*

            ACALL putdigitLCD

            MOV **A**, #":"         *; separator*

            ACALL putcharLCD

            MOV **A**, **R6**           *; minuty*

            ACALL putdigitLCD

            MOV **A**, #":"         *; separator*

            ACALL putcharLCD

            MOV **A**, **R7**           *; sekundy*

            ACALL putdigitLCD

            JMP FINAL

GODZINY:    MOV **R6**, #00H        *; zerowanie minut*

            INC **R5**              *; licznik godzin*

            MOV **A**, **R5**

            CLR C

            SUBB **A**, #24         *; przepelenienie godzin - doba*

            JNZ EKRAN

            MOV **R5**, #00H        *; zerowanie godzin*

EKRAN:      LCDcntrlWR #HOME    *; wyswietlenie calego zegara*

            MOV **A**, **R5**           *; godziny*

            ACALL putdigitLCD

            MOV **A**, #":"         *; separator*

            ACALL putcharLCD

            MOV **A**, **R6**           *; minuty*

            ACALL putdigitLCD

            MOV **A**, #":"         *; separator*

            ACALL putcharLCD

            MOV **A**, **R7**           *; sekundy*

            ACALL putdigitLCD

FINAL:      RET

*; program gl�wny*

START:

        key\_1:  mov r3, #LINE\_1 *; sprawdzanie w petli czy klawisz A zostal wcisniety*

                    mov a, r3

                    mov P5, a

                    mov a, P7

                    anl a, r3

                    mov r2, a

                    clr c

                    subb a, r3

                    //jz key\_2

                    mov a, r2

                    clr c*;*

                    subb a, #07eh

                    jnz key\_1

*; jesli zostal wcisniety klawisz A to:*

        init\_LCD *; inicjalizacja ekranu LCD*

        MOV **TMOD**, #01H          *; konfiguracja timera*

        MOV **TH0**, #3CH           *; ladowanie*

        MOV **TL0**, #0B0H          *; stalej timera na 50ms*

        SETB **TR0**                *; timer start*

        MOV **IE**, #82H            *; przerwania wlacz*

        MOV **R5**, #00H            *; inicjacja zegara*

        MOV **R6**, #00H

        MOV **R7**, #0FFH

        ACALL ZEGAR             *; wyswietlenie zainicjowanego zegara*

        MOV **A**, #0FH

        MOV **P1**, **A**               *; zapalenie di�d*

        MOV **R0**, #20             *; licznik odmierzen 20 x 50ms*

CZEKAM:

        mov r3, #LINE\_2

                    mov a, r3

                    mov P5, a

                    mov a, P7

                    anl a, r3

                    mov r2, a

                    mov a, r2

                    clr c*;*

                    subb a, #0BEh *; sprawdzenie czy zostal wcisniety klawisz B*

                    jnz cont

                    clr **TR0** *; jesli klawisz B wcisniety, to zastopowanie timera*

                    cont:

                    mov r3, #LINE\_3

                    mov a, r3

                    mov P5, a

                    mov a, P7

                    anl a, r3

                    mov r2, a

                    mov a, r2

                    clr c*;*

                    subb a, #0DEh *; sprawdzenie czy zostal wcisniety klawisz C*

                    jnz cnt2

                    SETB **TR0** *; jesli klawisz C wcisniety to wznowienie timera*

                    cnt2:

                    mov r3, #LINE\_4

                    mov a, r3

                    mov P5, a

                    mov a, P7

                    anl a, r3

                    mov r2, a

                    mov a, r2

                    clr c*;*

                    subb a, #0EEh *; sprawdzenie czy zostal wcisniety klawisz D*

                    jnz cnt3

                    MOV **R5**, #00H *; jesli klawisz D wcisniety to inicjacja zegara*

                    MOV **R6**, #00H

                    MOV **R7**, #0FFH

                    cnt3:

 MOV **A**, **R0**              *; czekam, a timer*

        JNZ CZEKAM              *; mierzy laczny czas 1s*

        MOV **R0**, #20             *; po zgloszeniu przerwania - ustawiam na nowo licznik odmierzen 20 x 50ms*

        ACALL ZEGAR             *; uruchomienie procedury oblugi i wyswietlenia zegara*

        MOV **A**, **P1**               *; zmiana*

        CPL **A**                   *; swiecenia*

        MOV **P1**, **A**               *; di�d*

        JMP CZEKAM              *; czekam na kolejna sekunde*

        NOP

        NOP

        NOP

        JMP $

END START

1. **Program "zegar" wzbogacony o możliwość ustawienia pozycji godzin i minut przed uruchomieniem odmierzania czasu**

Program przed uruchomieniem zegara pozwala ustalić godzinę początkową poprzez wpisanie jej na klawiaturze (tylko godzinę). Następnie program zachowuje się tak jak w pierwszym zadaniu – czeka na wystartowanie zegara klawiszem „A”. Reszta funkcjonalności pierwszego programu została zachowana.

ljmp start

P5 equ 0F8H

P7 equ 0DBH

LCDstatus  equ 0FF2EH       *; adres do odczytu gotowosci LCD*

LCDcontrol equ 0FF2CH       *; adres do podania bajtu sterujacego LCD*

LCDdataWR  equ 0FF2DH       *; adres do podania kodu ASCII na LCD*

// bajty sterujace LCD, inne dostepne w opisie LCD na stronie WWW

#define  HOME     0x80     // put cursor to second line

#define  INITDISP 0x38     // LCD init (8-bit mode)

#define  HOM2     0xc0     // put cursor to second line

#define  LCDON    0x0e     // LCD nn, cursor off, blinking off

#define  CLEAR    0x01     // LCD display clear

// linie klawiatury - sterowanie na port P5

#define LINE\_1      0x7f    // 0111 1111

#define LINE\_2      0xbf    // 1011 1111

#define LINE\_3      0xdf    // 1101 1111

#define LINE\_4      0xef    // 1110 1111

#define ALL\_LINES   0x0f    // 0000 1111

ORG 000BH                   *; obsluga przerwania*

    MOV **TH0**, #3CH           *; przeladowanie*

    MOV **TL0**, #0B0H          *; stalej timera na 50ms*

    DEC **R0**                  *; korekta licznika*

    RETI                    *; powr�t z przerwania*

org 0100H

// macro do wprowadzenia bajtu sterujacego na LCD

LCDcntrlWR MACRO x          *; x � parametr wywolania macra � bajt sterujacy*

           LOCAL loop       *; LOCAL oznacza ze etykieta loop moze sie powt�rzyc w programie*

loop: MOV  DPTR,#LCDstatus  *; DPTR zaladowany adresem statusu*

      MOVX **A**,@DPTR          *; pobranie bajtu z biezacym statusem LCD*

      JB   **ACC**.7,loop       *; testowanie najstarszego bitu akumulatora*

*; � wskazuje gotowosc LCD*

      MOV  DPTR,#LCDcontrol *; DPTR zaladowany adresem do podania bajtu sterujacego*

      MOV  **A**, x             *; do akumulatora trafia argument wywolania macra�bajt sterujacy*

      MOVX @DPTR,**A**          *; bajt sterujacy podany do LCD � zadana akcja widoczna na LCD*

      ENDM

// macro do wypisania znaku ASCII na LCD, znak ASCII przed wywolaniem macra ma byc w **A**

LCDcharWR MACRO

      LOCAL tutu            *; LOCAL oznacza ze etykieta tutu moze sie powt�rzyc w programie*

      PUSH **ACC**              *; odlozenie biezacej zawartosci akumulatora na stos*

tutu: MOV  DPTR,#LCDstatus  *; DPTR zaladowany adresem statusu*

      MOVX **A**,@DPTR          *; pobranie bajtu z biezacym statusem LCD*

      JB   **ACC**.7,tutu       *; testowanie najstarszego bitu akumulatora*

*; � wskazuje gotowosc LCD*

      MOV  DPTR,#LCDdataWR  *; DPTR zaladowany adresem do podania bajtu sterujacego*

      POP  **ACC**              *; w akumulatorze ponownie kod ASCII znaku na LCD*

      MOVX @DPTR,**A**          *; kod ASCII podany do LCD � znak widoczny na LCD*

      ENDM

// macro do inicjalizacji wyswietlacza � bez parametr�w

init\_LCD MACRO

         LCDcntrlWR #INITDISP *; wywolanie macra LCDcntrlWR � inicjalizacja LCD*

         LCDcntrlWR #CLEAR    *; wywolanie macra LCDcntrlWR � czyszczenie LCD*

         LCDcntrlWR #LCDON    *; wywolanie macra LCDcntrlWR � konfiguracja kursora*

         ENDM

// funkcja wypisania liczby dla potrzeb zegara

putdigitLCD:    mov b, #10

                div ab              *; uzyskanie cyfry dziesiatek*

                add a, #30H         *; konwersja cyfry na kod ASCII*

                acall putcharLCD

                mov a, b            *; ladowanie cyfry jednosci*

                add a, #30H         *; konwersja na LCD*

                acall putcharLCD

                ret

keyascii:   mov dptr, #80EBH    *;ta część służy do translacji kodu klawisza na cyfrę*

            mov a, #0

            movx @dptr, a

            mov dptr, #8077H

            mov a, #1

            movx @dptr, a

            mov dptr, #807BH

            mov a, #2

            movx @dptr, a

            mov dptr, #807DH

            mov a, #3

            movx @dptr, a

            mov dptr, #80B7H

            mov a, #4

            movx @dptr, a

            mov dptr, #80BBH

            mov a, #5

            movx @dptr, a

            mov dptr, #80BDH

            mov a, #6

            movx @dptr, a

            mov dptr, #80D7H

            mov a, #7

            movx @dptr, a

            mov dptr, #80DBH

            mov a, #8

            movx @dptr, a

            mov dptr, #80DDH

            mov a, #9

            movx @dptr, a

// funkcaj wypisywania znaku na LCD

putcharLCD: LCDcharWR

            ret

// wyznaczanie biezacej wartosci zegara i jego wyswietlanie na LCD

ZEGAR:      INC **R7**              *; licznik sekund*

            MOV **A**, **R7**           *; obsluga sekund*

            CLR C

            SUBB **A**, #60         *; przepelnienie sekund*

            JZ MINUTY

            LCDcntrlWR #HOME    *; wyswietlenie calego zegara*

            MOV **A**, **R5**           *; godziny*

            ACALL putdigitLCD

            MOV **A**, #":"         *; separator*

            ACALL putcharLCD

            MOV **A**, **R6**           *; minuty*

            ACALL putdigitLCD

            MOV **A**, #":"         *; separator*

            ACALL putcharLCD

            MOV **A**, **R7**           *; sekundy*

            ACALL putdigitLCD

            JMP FINAL

MINUTY:     MOV **R7**, #00H        *; zerowanie sekund*

            INC **R6**              *; licznik minut*

            MOV **A**, **R6**           *; obsluga minut*

            CLR C

            SUBB **A**, #60         *; przepelnienie minut*

            JZ GODZINY

            LCDcntrlWR #HOME    *; wyswietlenie calego zegara*

            MOV **A**, **R5**           *; godziny*

            ACALL putdigitLCD

            MOV **A**, #":"         *; separator*

            ACALL putcharLCD

            MOV **A**, **R6**           *; minuty*

            ACALL putdigitLCD

            MOV **A**, #":"         *; separator*

            ACALL putcharLCD

            MOV **A**, **R7**           *; sekundy*

            ACALL putdigitLCD

            JMP FINAL

GODZINY:    MOV **R6**, #00H        *; zerowanie minut*

            INC **R5**              *; licznik godzin*

            MOV **A**, **R5**

            CLR C

            SUBB **A**, #24         *; przepelenienie godzin - doba*

            JNZ EKRAN

            MOV **R5**, #00H        *; zerowanie godzin*

EKRAN:      LCDcntrlWR #HOME    *; wyswietlenie calego zegara*

            MOV **A**, **R5**           *; godziny*

            ACALL putdigitLCD

            MOV **A**, #":"         *; separator*

            ACALL putcharLCD

            MOV **A**, **R6**           *; minuty*

            ACALL putdigitLCD

            MOV **A**, #":"         *; separator*

            ACALL putcharLCD

            MOV **A**, **R7**           *; sekundy*

            ACALL putdigitLCD

FINAL:      RET

*; program gl�wny*

START:

         MOV **R5**, #00H           *; inicjacja zegara*

         acall keyascii

    key\_1:  mov r4, #LINE\_1 *; odczyt znaku z klawiatury jest zaimplementowany*

            mov a, r4       *;  w ten sam sposób co na poprzednich laboratoriach*

            mov P5, a

            mov a, P7

            anl a, r4

            mov r2, a

            clr c

            subb a, r4

            jz key\_2    *;jeśli nic nie zostało naciśnięte w pierwszej lini*

            mov a, r2   *;skaczemy do sprawdzenia kolejnej*

            mov dph, #80h

            mov dpl, a

            movx a,@dptr

            mov **P1**, a

            mov r5, a   *;ustawiam godzinę na podstawie naciśniętego klawisza*

            jmp key\_5   *;skok do oczekiwania na naciśnięcia przycisku startu timera (A)*

    key\_2:  mov r4, #LINE\_2

            mov a, r4

            mov P5, a

            mov a, P7

            anl a, r4

            mov r2, a

            clr c

            subb a, r4

            jz key\_3

            mov a, r2

            mov dph, #80h

            mov dpl, a

            movx a,@dptr

            mov **P1**, a

            mov r5, a

            jmp key\_5

    key\_3:  mov r4, #LINE\_3

            mov a, r4

            mov P5, a

            mov a, P7

            anl a, r4

            mov r2, a

            clr c

            subb a, r4

            jz key\_4

            mov a, r2

            mov dph, #80h

            mov dpl, a

            movx a,@dptr

            mov **P1**, a

            mov r5, a

            jmp key\_5

    key\_4:  mov r4, #LINE\_4

            mov a, r4

            mov P5, a

            mov a, P7

            anl a, r4

            mov r2, a

            clr c

            subb a, r4

            jz key\_1

            mov a, r2

            mov dph, #80h

            mov dpl, a

            movx a,@dptr

            mov **P1**, a

            mov r5, a

        key\_5:  mov r3, #LINE\_1 *;pętla wykrywająca naciśnięcie przycisku start (A)*

                    mov a, r3

                    mov P5, a

                    mov a, P7

                    anl a, r3

                    mov r2, a

                    clr c

                    subb a, r3

*;jz key\_2*

                    mov a, r2

                    clr c*;*

                    subb a, #07eh

                    jnz key\_5

       kont:

        init\_LCD

        MOV **TMOD**, #01H          *; konfiguracja timera*

        MOV **TH0**, #3CH           *; ladowanie*

        MOV **TL0**, #0B0H          *; stalej timera na 50ms*

        SETB **TR0**                *; timer start*

        MOV **IE**, #82H            *; przerwania wlacz*

        MOV **R6**, #00H

        MOV **R7**, #0FFH

        ACALL ZEGAR             *; wyswietlenie zainicjowanego zegara*

        MOV **A**, #0FH

        MOV **P1**, **A**               *; zapalenie di�d*

        MOV **R0**, #20             *; licznik odmierzen 20 x 50ms*

CZEKAM: *;program zachowuje wszystkie funkcjonalności programu*

        mov r3, #LINE\_2 *;z zadania pierwszego, więc dalsza część kodu się powtarza*

                    mov a, r3

                    mov P5, a

                    mov a, P7

                    anl a, r3

                    mov r2, a

                    mov a, r2

                    clr c*;*

                    subb a, #0BEh

                    jnz cont

                    clr **TR0**

                    cont:

                    mov r3, #LINE\_3

                    mov a, r3

                    mov P5, a

                    mov a, P7

                    anl a, r3

                    mov r2, a

                    mov a, r2

                    clr c*;*

                    subb a, #0DEh

                    jnz cnt2

                    SETB **TR0**

                    cnt2:

                    mov r3, #LINE\_4

                    mov a, r3

                    mov P5, a

                    mov a, P7

                    anl a, r3

                    mov r2, a

                    mov a, r2

                    clr c*;*

                    subb a, #0EEh

                    jnz cnt3

                    MOV **R5**, #00H            *; inicjacja zegara*

                    MOV **R6**, #00H

                    MOV **R7**, #0FFH

                    cnt3:

 MOV **A**, **R0**              *; czekam, a timer*

        JNZ CZEKAM              *; mierzy laczny czas 1s*

        MOV **R0**, #20             *; po zgloszeniu przerwania - ustawiam na nowo licznik odmierzen 20 x 50ms*

        ACALL ZEGAR             *; uruchomienie procedury oblugi i wyswietlenia zegara*

        MOV **A**, **P1**               *; zmiana*

        CPL **A**                   *; swiecenia*

        MOV **P1**, **A**               *; di�d*

        JMP CZEKAM              *; czekam na kolejna sekunde*

        NOP

        NOP

        NOP

        JMP $

END START