

Politechnika Lubelska
Wydział Elektrotechniki i Informatyki



**POLITECHNIKA
LUBELSKA**

Laboratorium Systemy wbudowane

Imię i Nazwisko	Nr grupy	Data	Prowadzący
Jakub Wróbel	5.8/16	18.11.2022	dr inż. Wojciech Surtel

Main.c

```
#define F_CPU 1000000L
#include <avr/io.h>
#include <util/delay.h>
#include "LCD.h" //import biblioteki LCD.h
#include "klawiatura.h" //import biblioteki klawiatura.h

int main(void)
{
    DDRB = 0xFF;
    DDRD |= 0x03;
    char liczba [17][2] = {" 0"," 1"," 2"," 3"," 4"," 5"," 6"," 7"," 8","
9","10","11","12","13","14","15","16"};
    ini(); // inicjalizacja LCD
    init(1); //inicjalizacja klawiatuwr
    unsigned char k = 0; // utworzenie zmiennej char
    while (1)
    {
        k=czytaj_klawiature(0); // przypisanie przycisku z klawiatury do zmiennej char
        wyswietl(liczba[k],2); // wyświetlenie odpowiedniego znaku na LCD
        _delay_ms(500); // opóźnienie
        czysc(); //funkcja czyszczaca LCD
        _delay_ms(500); // opóźnienie
    }
}
```

Klawiatura.h

```
#ifndef KLAWIATURA_H_
#define KLAWIATURA_H_
#include "klawiatuara.h"

void init(char chose);
unsigned char czytaj_klawiature(char malaKla);

#endif
```

Klawiatura.c

```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>
#include "klawiatuara.h"
#include "LCD.h"

void init(char chose){ // funkcja inicjalizacji (do ktorego portu podpieto klawiature)
    switch (chose){ // uzycie struktury switch
        case 1: //inicjalizacja portu A
            DDRA = 0xFF;
            PORTA = 0x0F;
            break;
        case 2: //inicjalizacja portu C
            DDRC = 0xFF;
            PORTC = 0x0F;
            break;
        case 3: // inicjalizacja portu D
            DDRD = 0xFF;
            PORTD = 0x0F;
            break;
        default: // default ti port A
            DDRA = 0xFF;
            PORTA = 0x0F;
            break;}
}

volatile unsigned char klawisz;
unsigned char czytaj_klawiature(char malaKla) //funkcja zczytujaca klawisz
{
    unsigned char wiersz, kolumna, key;
    if(malaKla){ // warunek na mala klawiature
        PORTA = 0xEF;
        key=1;
        for(wiersz=0x01; wiersz <= 0x08; wiersz <<=1,key++) // przejście po wszystkich
klawiszach malej klawiatury
            if(!(PINA&wiersz)) //sprawdzenie czy klawisz jest wcisniety
                return key; //zwrocenie klawisza
    }
```

```

    }
    else{ //warunek na cala klawiature
        for(kolumna=0xEF, key=1; kolumna>= 0x71; (kolumna<=1 | 0x01) &0xFF ) //przejscie po
wszystkich kolumnach duzej klawiatury
        {
            PORTA = kolumna;
            for (wiersz=0x01; wiersz<=0x08; wiersz<=1, key++) // przejscie po wszystkich
wierszach duzej klawiatury
                if(!(PINA & wiersz)) // spawdzenie czy klawisz jest klikniety
                    return key; // wysiwetlenie klawisza
        }
    }
    return 0;
}

```

LCD.h

```
#ifndef LCD_H_
#define LCD_H_
#include "LCD.h"

void cmd(uint8_t komenda);
void ini();
void czysc();
void ustaw(unsigned char x, unsigned char y );
void czysc_y(int8_t x, int8_t y);
void wyswietl(char *tekst, int8_t dlugosc);

#endif
```

LCD.c

```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>
#include "LCD.h"
#define LCD_DDR DDRB // przypisanie wyświetlacza LCD do portu DDRB i poszczególnych pinów
#define LCD_PORT PORTB
#define LCD_RS PB0
#define LCD_EN PB1
#define LCD_DB4 PB4
#define LCD_DB5 PB5
#define LCD_DB6 PB6
#define LCD_DB7 PB7

void cmd(uint8_t komenda)
{
    LCD_PORT |= 1<<LCD_EN; //włączanie linii ENABLE
    LCD_PORT = (komenda & 0xF0) | (LCD_PORT & 0x0F); // wysłanie 4 starszych bitów
    LCD_PORT &= ~(1<<LCD_EN); // potwierdzenie wysłania danych poprzez opadnięcie ENABLE
    asm volatile("nop"); // oczekiwanie jednego cyklu
    LCD_PORT |= 1<<LCD_EN;
    LCD_PORT = ((komenda & 0x0F)<<4) | (LCD_PORT & 0x0F); // wysłanie 4 młodszych bitów
    LCD_PORT &= ~(1<<LCD_EN);
    _delay_us(50); // oczekiwanie czasu na potwierdzenie wysłanych danych
}

void ini()
{
    LCD_DDR = (0xF0) | (1<<LCD_RS) | (1<<LCD_EN); //ustawienie kierunku wyjściowego dla
    wszystkich linii
```

```

LCD_PORT = 0;//ustawienie początkowego stanu niskiego na wszystkich liniach
LCD_PORT &= ~(1<<LCD_RS);//rozpoczęcie wysyłania komendy
//ustawienie parametrów wyświetlacza, bit4: 1-8 linii, 0-4 linii
//bit3: 1-2 wiersze, 0-1 wiersz, bit2: 0-wymiar znaku 5x8, 1 - wymiar 5x10
cmd(0b00101000);
LCD_PORT |= 1<<LCD_RS;
LCD_PORT |= 1<<LCD_RS;
//bit2 - tryb pracy wyświetlacza
//bit1:1-presuniecie okna, 0 - przesuniecie kursora
cmd(0b00000110);
LCD_PORT |= 1<<LCD_RS;
LCD_PORT &= ~(1<<LCD_RS);
//bit2:1-wyświetlacz włączony, 0-wyłączony
//bit1: 1 - włączenie wyświetlacza kursora,0 - kursor niewidoczny
//bit0: 1 - kursor miga,0 - kursor nie miga
cmd(0b00001100);
LCD_PORT |= 1<<LCD_RS;
}

void czysc() // funkcja czyszczaca wyświetlacz
{
    LCD_PORT &= ~(1<<LCD_RS); // przestawia na linii RS wartość na 0 po to by wysłać komendę
    a nie dane
    cmd(0x01); // wysłanie polecenia wyczyszczenia
    LCD_PORT |= 1<<LCD_RS; // przestawia linie RS na wartość 1 odpowiadającą wprowadzaniu
    danych
    _delay_ms(50);
}

void ustaw(unsigned char x, unsigned char y) // funkcja ustawiająca kursor w danej pozycji
{
    LCD_PORT &= ~(1<<LCD_RS);
    cmd((x*0x40+y) | 0x80);
    LCD_PORT |= 1<<LCD_RS;
    _delay_ms(5);
}

void czysc_y(int8_t x, int8_t y) // funkcja usuwająca tekst do od podanego punktu do końca linii
{
    while(y<16)
    {
        ustaw(x,y); // wywołanie funkcji ustawiającej kursor w podanej pozycji
        cmd(' '); // wysłanie polecenia czyszczącego
        y++; // przejście do następnego znaku
        _delay_ms(50);
    }
}

```

```

void wyswietl(char *tekst, int8_t dlugosc) //funkcja wyswietlajaca tekst
{
    int8_t i=0;
    ustaw(0,0); // ustaw kursor na pierwszy znak pierwszej lini
    while(i<dlugosc) // przejscie po wszystkich znakach tekstu
    {
        if(i==16) // jesli trafi na ostatni znak lini zmien linie
        {
            ustaw(1,0);
        }
        cmd(tekst[i]); // wypisz i-ty znak tekstu
        i++; // inkrementacja
    }
}

```

Wnioski:

Program działa zgodnie z założeniem oraz został sprawdzony przez prowadzącego w czasie zajęć. Funkcje są importowane z bibliotek Klawiatura.h oraz LCD.h, a ich ciała funkcji umieszczone w plikach o tych samych nazwach z rozszerzeniem .c.