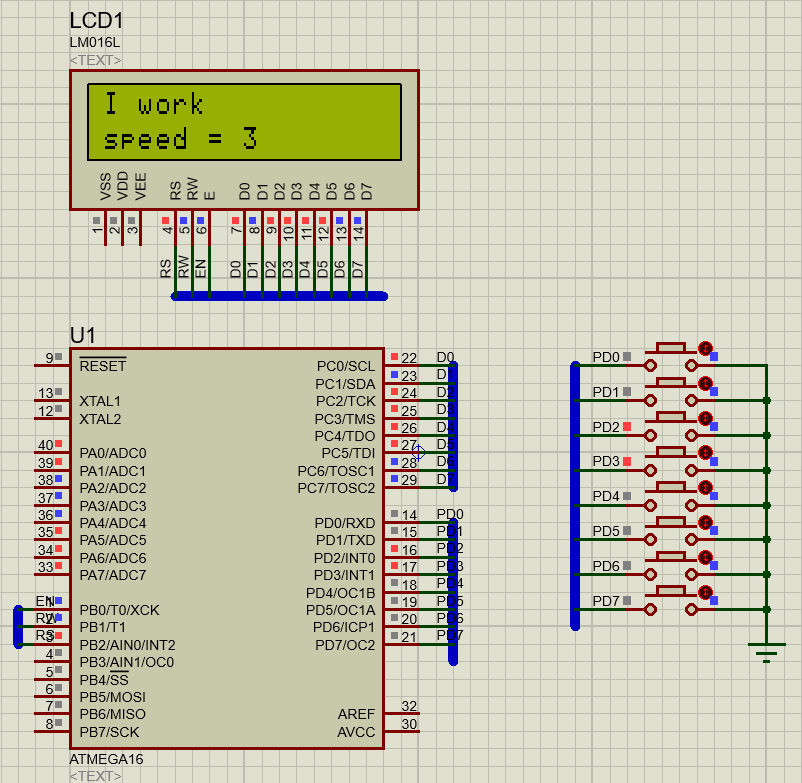
* 1. Разработать функцию, которая в момент включения платы С51 запускает «демо-режим» - выводит на ЖК сообщение с предложением о нажатии кнопок и зажигает на светодиодах платы самостоятельно разработанную студентом последовательность.
  2. Разработать функцию нажатия свободных кнопок, изменяющую скорость бегущих огней.

1. Разработать функцию вывода на ЖК дисплей таймера от 0 до N, где N- произвольное число. Управление таймером: скорость изменения значений и пауза/запуск счёта задавать через прерывания INT0, INT2.

Для задания 1:



#define *F\_CPU* 12000000UL

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

#define RS 2

#define RW 1

#define EN 0

void button\_init(void);

void timer\_init(void);

void lcd\_init(void);

void lcd\_com(unsigned char);

void lcd\_dat(unsigned char);

void lcd\_string(char\*);

void next\_step(int i);

*uint8\_t* mode = 0; // Текущие состояние

*uint8\_t* mode\_last = 0; // Предидущие состояние

*uint8\_t* speed\_mode = 1; // Скоросной режим

char speed\_char[] = "speed = 1"; // Строка для отображения скорости на дисплее

*uint16\_t* initial\_timer\_value = 55000; // Изначальное значение для таймера

*uint8\_t* pattern = 0b11111000; // Паттерн бегущих огней

*uint8\_t* direction = 0; // Режим бега влево/вправо

*uint8\_t* step = 0; // Шаг для бегущих огней

int main(void)

{

timer\_init();

lcd\_init();

button\_init();

sei();

int running\_line\_step = 0;

int messageLength = 61;

while (1)

{

switch (mode)

{

case (0):{

if (mode\_last == 1) {

mode\_last = 0;

running\_line\_step = 0;

}

next\_step(running\_line\_step);

running\_line\_step++;

if (running\_line\_step >= messageLength){

running\_line\_step = 0;

}

break;

}

case(1):{

if (mode\_last == 0) {

mode\_last = 1;

lcd\_com(0x01);

lcd\_com(0x80);

lcd\_string("I work");

lcd\_com(0xC0);

}

lcd\_com(0xC0);

lcd\_string(speed\_char);

break;

}

}

}

}

ISR(INT0\_vect){

mode\_last = mode;

mode = !mode;

TCNT1 = initial\_timer\_value;

};

ISR(INT1\_vect){

speed\_mode += 1;

if (speed\_mode > 3){

speed\_mode = 1;

}

switch (speed\_mode)

{

case (3):{ speed\_char[8] = '3'; initial\_timer\_value = 65000; break; }

case (2):{ speed\_char[8] = '2'; initial\_timer\_value = 60000; break; }

case (1):{ speed\_char[8] = '1'; initial\_timer\_value = 55000; break; }

}

};

ISR(TIMER1\_OVF\_vect)

{

if (mode == 0) {

PORTA = 0b00000000;

return;

}

PORTA = pattern;

if (direction == 0) {

pattern = (pattern << 1) | (pattern >> 7); // Сдвиг вправо

} else {

pattern = (pattern >> 1) | (pattern << 7); // Сдвиг влево

}

step++;

if(step > 4) {

direction = !direction;

step = 0;

}

TCNT1 = initial\_timer\_value;

}

void next\_step(int i) {

char staticText[] = "Instruction";

char message[] = " Press the button! 1 - instructions or run; 2 - lights speed;";

int messageLength = 62;

char message16[16]; // массив для хранения 16 символов + завершающий нуль

lcd\_com(0x80); // Устанавливаем курсор на первую строку

lcd\_string(staticText);

lcd\_com(0xC0); // Устанавливаем курсор на вторую строку

for (int j = 0; j < 16; j++) {

message16[j] = message[(j + i) % messageLength];

}

lcd\_string(message16);

*\_delay\_ms*(100); // Доп задержка для чтения бегущей строки

}

void button\_init(void){

DDRD &=~(1<<PD2);

DDRD &=~(1<<PD3);

PORTD |=(1<<PD2)|(1<<PD3);

GICR |= (1<<INT0)|(1<<INT1);

MCUCR &=~ (1<<ISC01)|(1<<ISC00);

MCUCR &=~ (1<<ISC11)|(1<<ISC10);

}

void lcd\_com(unsigned char p){

PORTB &= ~(1<<RS);

PORTB |= (1<<EN);

PORTC = p;

*\_delay\_us*(500);

PORTB &=~(1<<EN);

*\_delay\_us*(500);

}

void lcd\_dat(unsigned char p){

PORTB|=(1<<RS)|(1<<EN);

PORTC=p;

*\_delay\_us*(500);

PORTB&=~(1<<EN);

*\_delay\_us*(500);

}

void lcd\_init(void){

// Устанавливаем пины RS, RW и EN порта DDRB как выходные

DDRB |= (1<<RS)|(1<<RW)|(1<<EN);

PORTB=0x00; // Обнуляем порт B

DDRC=0xFF; // Устанавливаем порт C как порт вывода

PORTC=0x00; // Обнуляем порт C

*\_delay\_us*(500); // Задержка 500 микросекунд

lcd\_com(0x08); // Инициализация дисплея: выключаем дисплей

*\_delay\_us*(500); // Задержка 500 микросекунд

lcd\_com(0x3C); // Установка режима 8 бит данных, 2 строки, 5x8 точек

*\_delay\_us*(500); // Задержка 500 микросекунд

lcd\_com(0x01); // Очистка дисплея

*\_delay\_us*(500); // Задержка 500 микросекунд

lcd\_com(0x06); // Установка направления пути записи, увеличение адреса на 1

*\_delay\_us*(900); // Задержка 900 микросекунд

lcd\_com(0x0C); // Включаем дисплей без курсора

}

void timer\_init(void){

DDRA = 0xFF;

TCCR1B |= (1 << CS12) | (1 << CS10);

TIMSK |= (1 << TOIE1);

TCNT1 = initial\_timer\_value;

}

void lcd\_string(char \*str){

char data=0;

while(\*str){

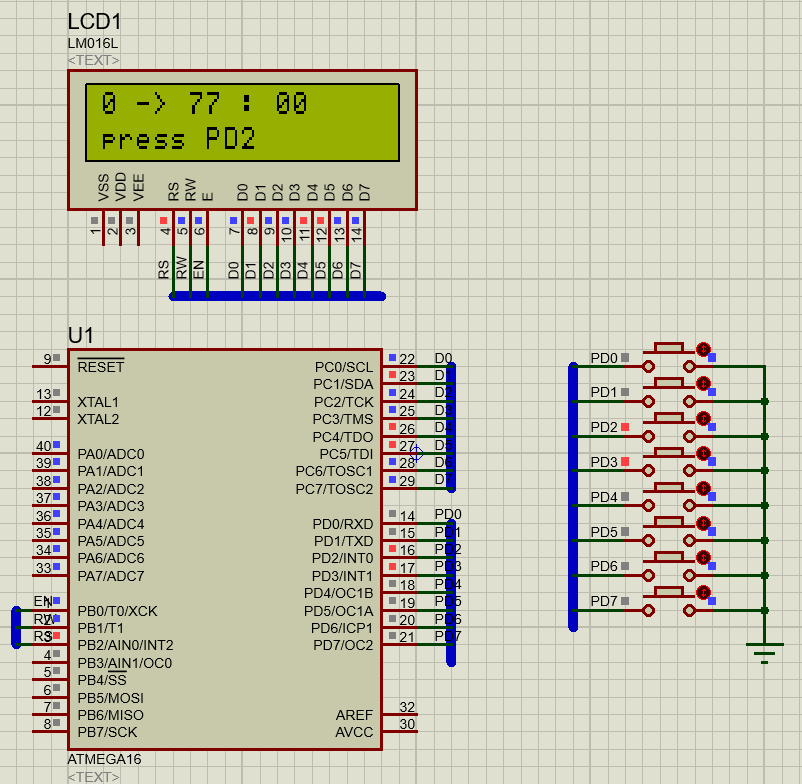
data=\*str++;

lcd\_dat(data);

}

}

Для задания 2:



#define *F\_CPU* 12000000UL

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

#define RS 2

#define RW 1

#define EN 0

void button\_init(void);

void timer\_init(void);

void lcd\_init(void);

void lcd\_com(unsigned char);

void lcd\_dat(unsigned char);

void lcd\_string(char\*);

*uint8\_t* mode = 0;

*uint8\_t* last\_mode = 0;

*uint8\_t* speed\_mode = 1;

*uint8\_t* last\_speed\_mode = 0;

char current\_time\_char[] = "0 -> 77 : 00 ";

char speed\_char[] = "Speed - 1";

*uint16\_t* initial\_timer\_value = 55000;

*uint8\_t* N = 77;

*uint8\_t* step = 0;

int main(void)

{

timer\_init();

lcd\_init();

button\_init();

sei();

lcd\_com(0x80);

lcd\_string(current\_time\_char);

lcd\_com(0xC0);

lcd\_string("press PD2");

while (1)

{

switch (mode)

{

case (0):{

if (last\_mode == 1) {

last\_mode = 0;

lcd\_com(0x80);

lcd\_string(current\_time\_char);

lcd\_com(0xC0);

lcd\_string("Pause ");

}

break;

}

case(1):{

lcd\_com(0x80);

lcd\_string(current\_time\_char);

if (speed\_mode != last\_speed\_mode || last\_mode == 0){

last\_speed\_mode = speed\_mode; last\_mode = 0;

lcd\_com(0xC0);

lcd\_string(speed\_char);

}

break;

}

}

}

}

ISR(INT0\_vect){

last\_mode = mode;

mode = !mode;

};

ISR(INT1\_vect){

if (mode == 0) { return; }

last\_speed\_mode = speed\_mode;

speed\_mode += 1;

if (speed\_mode > 3){

speed\_mode = 1;

}

switch (speed\_mode)

{

case (3):{ speed\_char[8] = '3'; initial\_timer\_value = 65000; break; }

case (2):{ speed\_char[8] = '2'; initial\_timer\_value = 60000; break; }

case (1):{ speed\_char[8] = '1'; initial\_timer\_value = 55000; break; }

}

};

ISR(TIMER1\_OVF\_vect)

{

if (step < N) {

current\_time\_char[10] = '0' + step / 10;

current\_time\_char[11] = '0' + step % 10;

step++;

} else {

step = 0;

}

TCNT1 = initial\_timer\_value;

}

void button\_init(void){

DDRD &=~(1<<PD2);

DDRD &=~(1<<PD3);

PORTD |=(1<<PD2)|(1<<PD3);

GICR |= (1<<INT0)|(1<<INT1);

MCUCR &=~ (1<<ISC01)|(1<<ISC00);

MCUCR &=~ (1<<ISC11)|(1<<ISC10);

}

void lcd\_com(unsigned char p){

PORTB &= ~(1<<RS);

PORTB |= (1<<EN);

PORTC = p;

*\_delay\_us*(500);

PORTB &=~(1<<EN);

*\_delay\_us*(500);

}

void lcd\_dat(unsigned char p){

PORTB|=(1<<RS)|(1<<EN);

PORTC=p;

*\_delay\_us*(500);

PORTB&=~(1<<EN);

*\_delay\_us*(500);

}

void lcd\_init(void){

// Устанавливаем пины RS, RW и EN порта DDRB как выходные

DDRB |= (1<<RS)|(1<<RW)|(1<<EN);

PORTB=0x00; // Обнуляем порт B

DDRC=0xFF; // Устанавливаем порт C как порт вывода

PORTC=0x00; // Обнуляем порт C

*\_delay\_us*(500);

lcd\_com(0x08); // Инициализация дисплея: выключаем дисплей

*\_delay\_us*(500);

lcd\_com(0x3C); // Установка режима 8 бит данных, 2 строки, 5x8 точек

*\_delay\_us*(500);

lcd\_com(0x01); // Очистка дисплея

*\_delay\_us*(500);

lcd\_com(0x06); // Установка направления пути записи, увеличение адреса на 1

*\_delay\_us*(900);

lcd\_com(0x0C); // Включаем дисплей без курсора

}

void timer\_init(void){

DDRA = 0xFF;

TCCR1B |= (1 << CS12) | (1 << CS10);

TIMSK |= (1 << TOIE1);

TCNT1 = initial\_timer\_value;

}

void lcd\_string(char \*str){

char data=0;

while(\*str){

data=\*str++;

lcd\_dat(data);

}

}