**Лабораторная работа №6**

**Тема**: ПРЕРЫВАНИЯ

**Цель**: разработать программное обеспечение использовав систему аппаратных прерываний.

**Задание:**

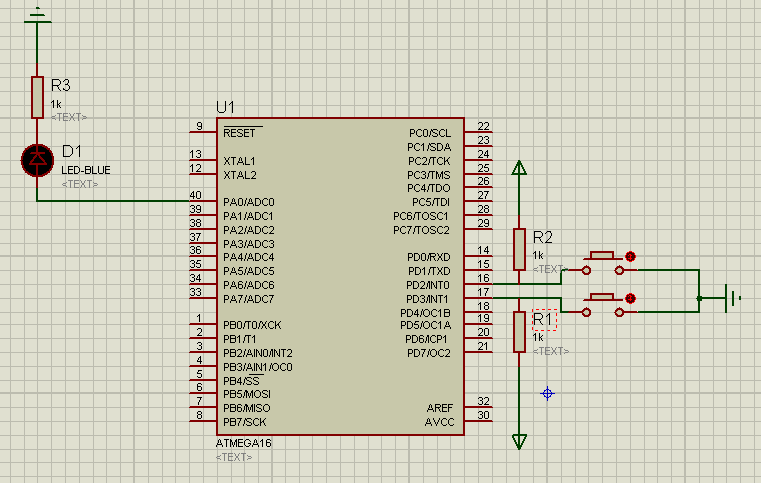
1. Разработать программу управления яркостью светодиодов с использованием прерываний от таймера. Дополнительное задание: реализовать опрос кнопок для ручного регулирования яркости светодиода.

2. Разработайте блок-схему алгоритма работы.

3. Создайте программное обеспечение, и проверьте работу программы на симуляторе.

4. Ответьте на контрольные вопросы.

(Полный перечень ассемблерных команд процессора предоставлено в файле atmega8.pdf на сервере с материалами к предмету).



/ \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*\*\*\*\*\*\*\*

подключение библиотек

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*\*\*\*\*\*\*\* /

#include <avr/io.h>

#include <avr/interrupt.h>

void intterup\_set();

void timer\_set();

char temp = 0;

/ \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*\*\*\*\*\*\*

Переопределения обработчиков прерываний

Для облегчения использовано макрос ISR ()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*\*\*\*\*\*\* /

// прерывание первой кнопки

ISR(INT0\_vect)

{

temp++;

OCR0 = temp;// инкремент

}

// прерывание второй кнопки

ISR(INT1\_vect)

{

temp--;

OCR0 = temp;// deкремент

}

// прерывание по совпадению

ISR(TIMER0\_COMP\_vect) { PORTA |=(1<<0); }

// прерывание по переполнению

ISR(TIMER0\_OVF\_vect) { PORTA &=~(1<<0); }

/ \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*\*

Главная функция Main

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*\* /

int main(void)

{

DDRA |=(1<<0); // порт на выход

OCR0 = 0;

timer\_set();

intterup\_set();

asm ("sei"); // разрешение глобального прерывания

while (1)

{

asm("nop");

}

}

void timer\_set()

{

TCCR0 = 0b01001011; // 0-2 предделитль 64 6-3 бит работа ШИМа

TIMSK |= (1<<1); // разрешение уходить в вектор при совпадении

TIMSK |= (1<<0); // разрешение уходить в вектор при переполнении

}

void intterup\_set()

{

MCUCR &=~(1<<0); // установка значений для прерывания

MCUCR &=~(1<<0); // по низкому уровню

GICR |=(1<<6); // разрешение прерывания INT0

GICR |=(1<<7); // разрешение прерывания INT1

}

**Контрольные вопросы:**

1. Как разрешить прерывание от таймера по переполнению?

0 бит регистра TIMSK

*TIMSK |= (1<<TOIE1); // разрешить прерывание по переполнению таймера счетчика*

1. Для какого таймера можно ограничить верхнее значение счетчика?

*OCR0*

1. Можно ли комбинировать архитектуру циклического изменения задач и архитектуру работы по прерыванием?

*Да*

1. Придумайте задачу, где поочередное выполнение двух задач является более выгодным, чем прерывание?

*Прерывания удобны в тех случаях, когда сигнал на ножку поступает редко, что бы не нагружать контроллер и постоянно не опрашивать контакт*.

1. Придумайте задачу, где архитектура работы по прерываниям имеет преимущества за архитектуру поочередного выполнения?

*Игровой автомат «успей первым»*