МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ПОВОЛЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ
УНИВЕРСИТЕТ»

Кафедра ИиСП

Отчет
по лабораторной работе № 6

по дисциплине «Машинно-зависимые языки программирования»
Вариант 2

Выполнил: ст. гр. ПС-14
Сайфутдинов Э.Р.
Проверил: доцент, доцент кафедры ИиСП Баев А.А.

г. Йошкар-Ола
2024

Цель работы:

Изучить различные принципы вывода информации на индикаторы, примеры использования встроенных таймеров и основы последовательной передачи данных.

Задания на лабараторную работу:

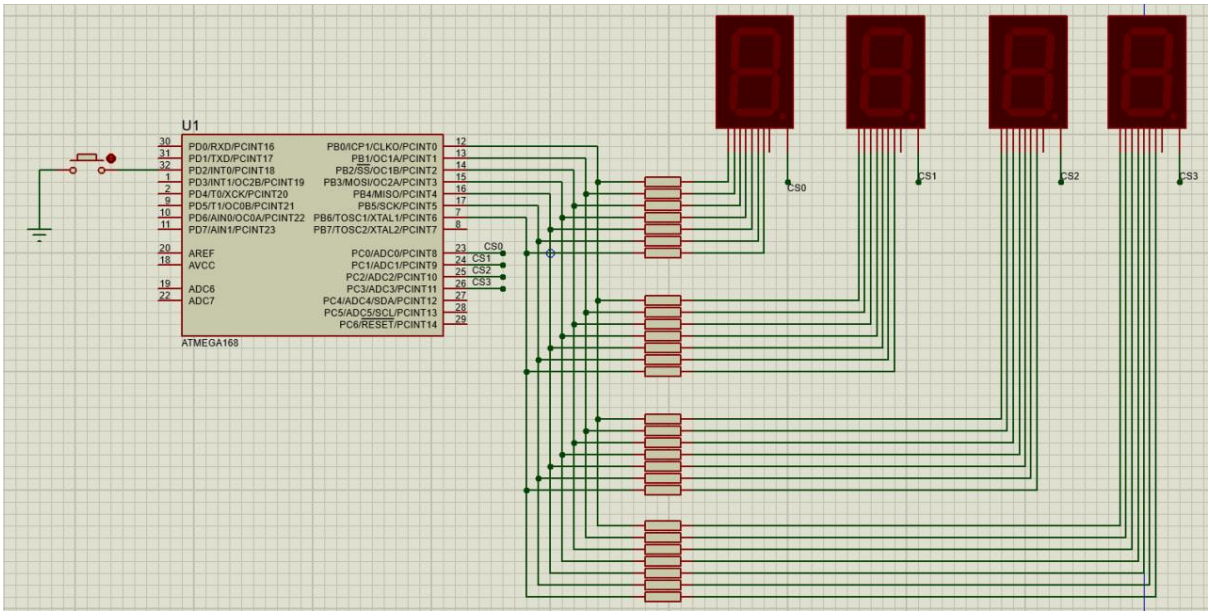
Собрать таймер в протеусе из 4-х 7-сегментных индикаторов и напсать для них код с использованием SPI и без.

1. **Теоретические сведения**

   Учебное пособие “Применение микроконтроллеров в радиотехнических и биомедицинских системах”.

2. Практическая часть

1) Реализация динамической индикации



Код для схемы:

```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
uint8_t segments[]={
	0b00111111, // 0 - A, B, C, D, E, F
	0b00000110, // 1 - B, C
	0b01011011, // 2 - A, B, D, E, G
	0b01001111, // 3 - A, B, C, D, G
	0b01100110, // 4 - B, C, F, G
	0b01101101, // 5 - A, C, D, F, G
	0b01111101, // 6 - A, C, D, E, F, G
	0b00000111, // 7 - A, B, C
	0b01111111, // 8 - A, B, C, D, E, F, G
	0b01101111, // 9 - A, B, C, D, F, G
};
void InitPorts(void);
void send_data(uint8_t data, uint8_t ind);
void InitTimer0(void);
void Bin2Dec(uint16_t data);
volatile uint16_t cnt = 0;
volatile uint8_t switch_state = 0;
volatile uint8_t bcd_buffer[] = {0,0,0,0};
int main(void)
{
	InitPorts();
	InitTimer0();
	EIMSK|=(1<<INT0); //ВключитьINT0
	EICRA|=(1<<ISC01); //Настройка INT0 на прерывание по спаду
	sei(); //Глобальное разрешение пре-рываний
	while(1)
	{
		if(switch_state == 0){
			Bin2Dec(cnt);
			if(cnt<9999){
				cnt++;
				}else{
				cnt=0;
			}
```

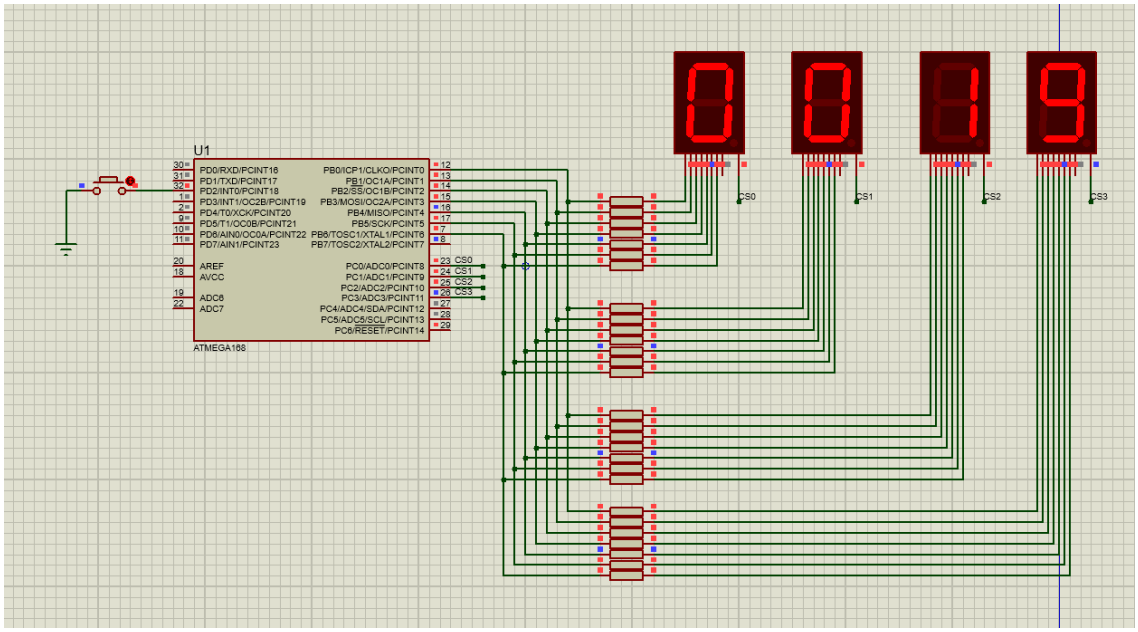
```
                        _delay_ms(100);
                }
        }
}
//-----------------------------------------
ISR(TIMER0_COMPA_vect){
        send_data(bcd_buffer[3],0);
        send_data(bcd_buffer[2],1);
        send_data(bcd_buffer[1],2);
        send_data(bcd_buffer[0],3);
}
ISR(INT0_vect){
        if(switch_state == 0){
                switch_state = 1;
                } else {
                switch_state = 0;
                cnt = 0;
        }
}
void InitPorts(void){
        DDRB = 0xFF;
        DDRC = (1<<PC0|1<<PC1|1<<PC2|1<<PC3);
        PORTC = 0x0F;
        DDRD = (0<<PD2);
        PORTD |= (1<<PD2);
}
void send_data(uint8_t data, uint8_t ind)
{
        PORTC = 0x0F &~ (1<<ind);
        PORTB = segments[data];
        _delay_ms(5);
        PORTB = 0;
        PORTC = 0x0F;
}
void InitTimer0(void)
{
        TCCR0A = (1<<WGM01); //режим CTC - Clear Timer on
        //Compare
        TCCR0B = (1<<CS02|1<<CS00);//prescaler = sys_clk/1024
        TCNT0 = 0x00; //начальное значение счетчика
        OCR0A = 16; //порог срабатывания
        TIMSK0 |= (1<<OCIE0A); //включение прерывания при
        //достижении порога A
}
void Bin2Dec(uint16_t data)
{
        bcd_buffer[3]=(uint8_t)(data/1000);
        data = data - bcd_buffer[3]*1000;
        bcd_buffer[2] = (uint8_t)(data/100);
        data = data - bcd_buffer[2]*100;
        bcd_buffer[1] = (uint8_t)(data/10);
        data = data - bcd_buffer[1]*10;
        bcd_buffer[0] = (uint8_t)(data);
}
```

Оптимизированный код:

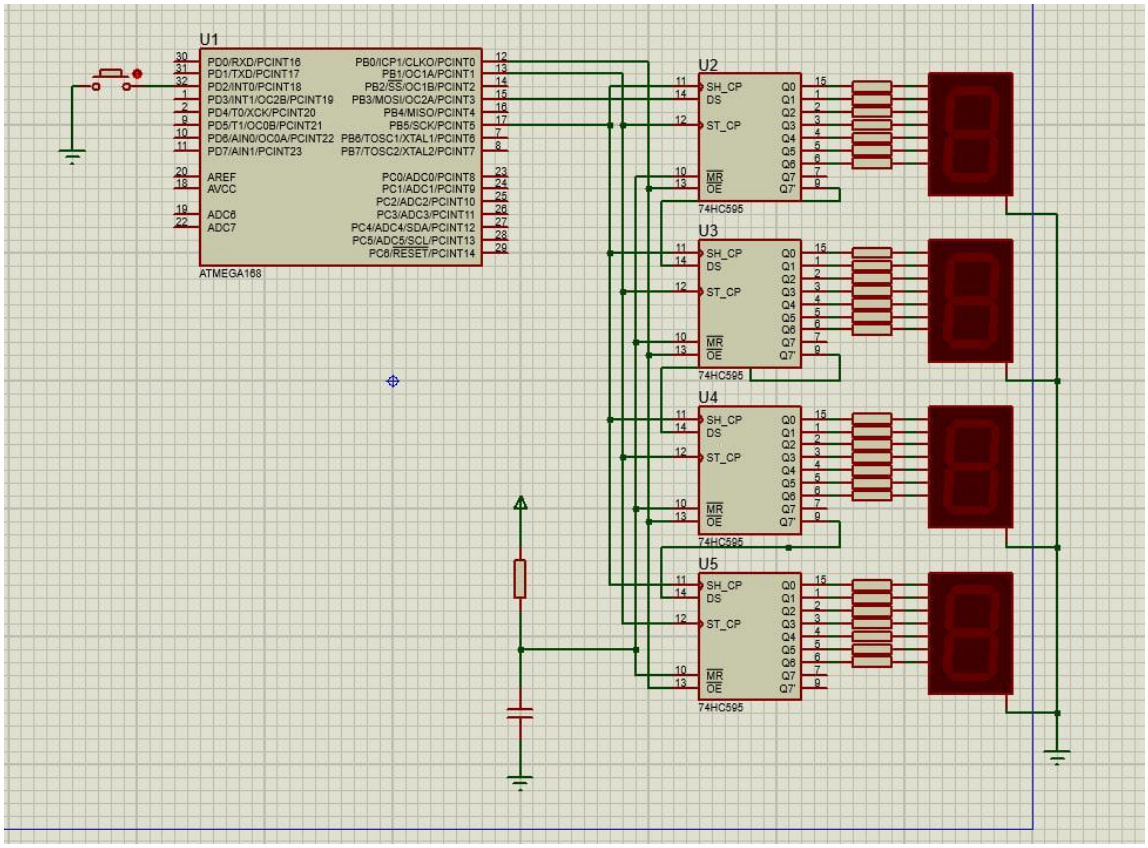
```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h> uint8_t segments[]=

{
        0b00111111, // 0 - A, B, C, D, E, F
```

```
	0b00000110, // 1 - B, C
	0b01011011, // 2 - A, B, D, E, G
	0b01001111, // 3 - A, B, C, D, G
	0b01100110, // 4 - B, C, F, G
	0b01101101, // 5 - A, C, D, F, G
	0b01111101, // 6 - A, C, D, E, F, G
	0b00000111, // 7 - A, B, C
	0b01111111, // 8 - A, B, C, D, E, F, G
	0b01101111, // 9 - A, B, C, D, F, G
};
void InitPorts(void);
void send_data(uint8_t data, uint8_t ind);
void InitTimer0(void);
void Bin2Dec(uint16_t data);
volatile uint16_t cnt = 0;
volatile uint8_t switch_state = 0;
volatile uint8_t bcd_buffer[] = {0,0,0,0};
int main(void)
{
	InitPorts();
	InitTimer0();
	EIMSK|=(1<<INT0); //ВключитьINT0
	EICRA|=(1<<ISC01); //Настройка INT0 напрерывание по спаду
	sei();  //Глобальное разрешение прерываний
	while(1)
	{
		if(switch_state == 0)
		{
			Bin2Dec(cnt);			if(cnt<9999)
			{
				cnt++;
			}
			else
			{
				cnt=0;
			}
			_delay_ms(100);
		}
	}
}
//-----------------------------------------

ISR(TIMER0_COMPA_vect)
{
	send_data(bcd_buffer[3],0);
	send_data(bcd_buffer[2],1);
	send_data(bcd_buffer[1],2);
	send_data(bcd_buffer[0],3);
}

ISR(INT0_vect)
{
	if(switch_state == 0)
	{
		switch_state = 1;
	}
	else
	{
		switch_state = 0;
		cnt = 0;
	}
}
void InitPorts(void)
{
	DDRB = 0xFF;
```

```
        DDRC = (1<<PINC0|1<<PINC1|1<<PINC2|1<<PINC3);
        PORTC = 0x0F;
        DDRD = (0<<PIND2);
        PORTD |= (1<<PIND2);
}
void send_data(uint8_t data, uint8_t ind)
{
        PORTC = 0x0F &~ (1<<ind);
        PORTB = segments[data];
        _delay_ms(5);
        PORTB = 0;
        PORTC = 0x0F;
}
void InitTimer0(void)
{
        TCCR0A = (1<<WGM01);
        TCCR0B = (1<<CS02|1<<CS00); //prescaler = sys_clk/1024
        TCNT0 = 0x00;        // начальное значение счетчика
        OCR0A = 16;   // порог срабатывания
        TIMSK0 |= (1<<OCIE0A);
}
void Bin2Dec(uint16_t data)
{
        bcd_buffer[3]=(uint8_t)(data/1000);
        data = data - bcd_buffer[3]*1000;
        bcd_buffer[2] = (uint8_t)(data/100);
        data = data - bcd_buffer[2]*100;
        bcd_buffer[1] = (uint8_t)(data/10);
        data = data - bcd_buffer[1]*10;
        bcd_buffer[0] = (uint8_t)(data);
}
```



2) Подключение индикаторов с помощью регистров



Код для схемы:

```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
uint8_t segments[]={
	0b00111111, // 0 - A, B, C, D, E, F
	0b00000110, // 1 - B, C
	0b01011011, // 2 - A, B, D, E, G
	0b01001111, // 3 - A, B, C, D, G
	0b01100110, // 4 - B, C, F, G
	0b01101101, // 5 - A, C, D, F, G
	0b01111101, // 6 - A, C, D, E, F, G
	0b00000111, // 7 - A, B, C
	0b01111111, // 8 - A, B, C, D, E, F, G
	0b01101111, // 9 - A, B, C, D, F, G
};
void InitPorts(void);
void send_data(uint8_t data, uint8_t ind);
void InitTimer0(void);
void Bin2Dec(uint16_t data);
void InitTimer1(void);
void StartTimer1(void);
void StopTimer1(void);
void SendData(uint8_t data);
void DisplayData(uint16_t data);
volatile uint16_t cnt = 0;
```

```
volatile uint8_t switch_state = 0;
volatile uint8_t bcd_buffer[] = {0,0,0,0};
int main(void)
{
	InitPorts();
	InitTimer1();
	EIMSK |= (1<<INT0); //разрешить прерывание INT0
	EICRA |= (1<<ISC01);//Запуск по заднему фронту INT0
	sei(); //Разрешение прерываний
	PORTB &= ~(1<<PB0); //OE = low (active)
	DisplayData(0);
	while(1)
	{ }
}
//-----------------------------------------
ISR(TIMER1_COMPA_vect){
	DisplayData(cnt);
	if(cnt<9999){
		cnt++;
		}else{
		cnt=0;
	}
}
ISR(INT0_vect){
	if(switch_state == 0){
		switch_state = 1;
		StartTimer1();
    }else{
	 StopTimer1();
	 DisplayData(cnt);
	 switch_state = 0;
	 cnt = 0;
}
}
//-----------------------------------------
void InitPorts(void){
	DDRB = (1<<PB0|1<<PB1|1<<PB3|1<<PB5);
	DDRD &= ~(1<<PD2);
	PORTD |= (1<<PD2);
}
void InitTimer1(void){
	TCCR1A = 0;
	TCCR1B = (1<<CS11|1<<CS10|1<<WGM12);
	TCNT1 = 0;
	OCR1A = 15624;
}
void StartTimer1(void){
	TCNT1 = 0;
	TIMSK1 |= (1<<OCIE1A);
}
void StopTimer1(void){
	TIMSK1 &= ~(1<<OCIE1A);
}
void Bin2Dec(uint16_t data){
	bcd_buffer[3] = (uint8_t)(data/1000);
	data = data % 1000;
	bcd_buffer[2] = (uint8_t)(data/100);
    data = data % 100;
    bcd_buffer[1] = (uint8_t)(data/10);
    data = data % 10;
    bcd_buffer[0] = (uint8_t)(data);
}
void SendData(uint8_t data){
	for(uint8_t i=0; i<8; i++){
		PORTB &= ~(1<<PB5); //CLK low
```

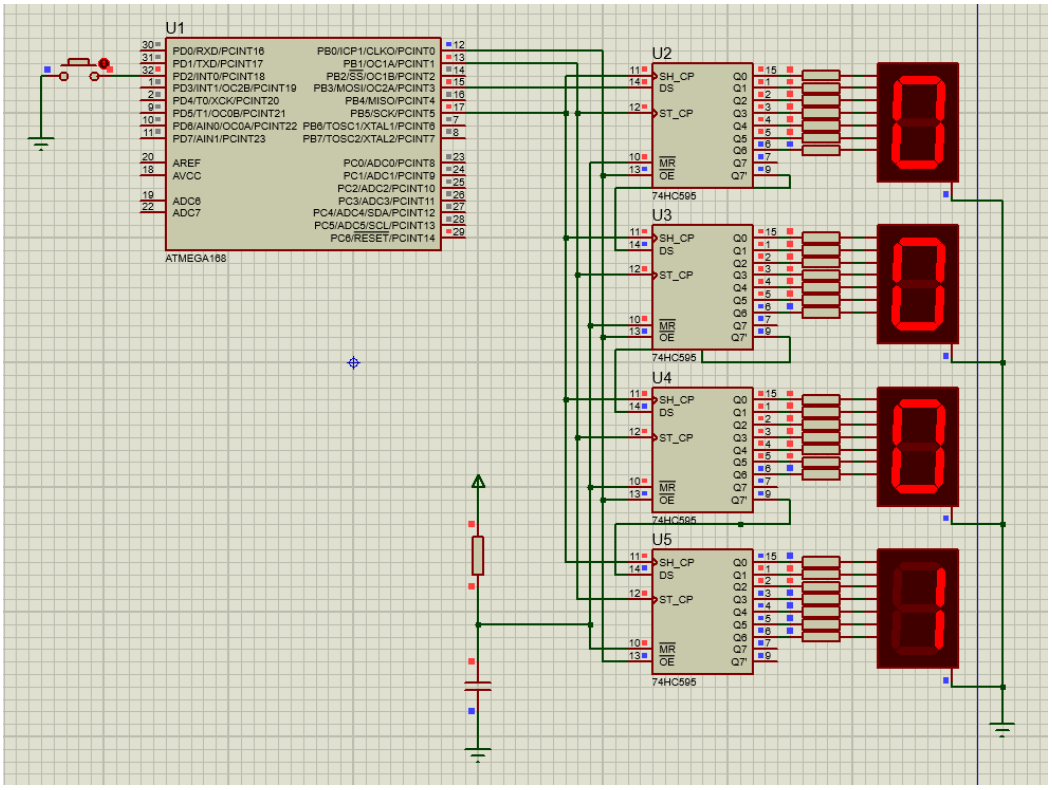
```
            if(0x80 & (data<<i)){
                  PORTB |= 1<<PB3; //DAT high
                  } else {
                  PORTB &= ~(1<<PB3); //DAT low
            }
            PORTB |= (1<<PB5); //CLK high
      }
}
void DisplayData(uint16_t data){
      Bin2Dec(data);
      PORTB &= ~(1<<PB1); //clk_out = 0
      SendData(segments[bcd_buffer[0]]);
      SendData(segments[bcd_buffer[1]]);
      SendData(segments[bcd_buffer[2]]);
      SendData(segments[bcd_buffer[3]]);
      PORTB |= (1<<PB1); //clk_out = 1
}
```

Оптимизированный код:

```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h> uint8_t segments[]=
{
      0b00111111, // 0 - A, B, C, D, E, F
      0b00000110, // 1 - B, C
      0b01011011, // 2 - A, B, D, E, G
      0b01001111, // 3 - A, B, C, D, G
      0b01100110, // 4 - B, C, F, G
      0b01101101, // 5 - A, C, D, F, G
      0b01111101, // 6 - A, C, D, E, F, G
      0b00000111, // 7 - A, B, C
      0b01111111, // 8 - A, B, C, D, E, F, G
      0b01101111, // 9 - A, B, C, D, F, G
};

void InitPorts(void);
void send_data(uint8_t data, uint8_t ind);
void InitTimer0(void);
void Bin2Dec(uint16_t data);
void InitTimer1(void);
void StartTimer1(void);
void StopTimer1(void);
void SendData(uint8_t data);
void DisplayData(uint16_t data);
volatile uint16_t cnt = 0;
volatile uint8_t switch_state = 0;
volatile uint8_t bcd_buffer[] = {0,0,0,0};

int main(void)
{
      InitPorts();
      InitTimer1();
      EIMSK |= (1<<INT0); //разрешить прерывание INT0
      EICRA |= (1<<ISC01);//Запуск по заднемуфронту INT0
      sei();
      //Разрешение прерываний
      PORTB &= ~(1<<PB0); //OE = low (active)
      DisplayData(0);
      while(1)
      {
      }
}
//-----------------------------------------
```

```
ISR(TIMER1_COMPA_vect){
	DisplayData(cnt);
	if(cnt<9999)
	{
		cnt++;
	}
	else
	{
		cnt=0;
	}
}
ISR(INT0_vect)
{
	if(switch_state == 0)
	{
		switch_state = 1;
		StartTimer1();
	}
	else
	{
		StopTimer1();
		DisplayData(cnt);
		switch_state = 0;
		cnt = 0;
	}
}
//-----------------------------------------

void InitPorts(void)
{
	DDRB = (1<<PINB0|1<<PINB1|1<<PINB3|1<<PINB5);
	DDRD &= ~(1<<PIND2);
	PORTD |= (1<<PIND2);
}
void InitTimer1(void)
{
	TCCR1A = 0;
	TCCR1B = (1<<CS11|1<<CS10|1<<WGM12);
	TCNT1 = 0;
	OCR1A = 15624;
}
void StartTimer1(void)
{
	TCNT1 = 0;
	TIMSK1 |= (1<<OCIE1A);
}
void StopTimer1(void)
{
	TIMSK1 &= ~(1<<OCIE1A);
}
void Bin2Dec(uint16_t data)
{
	bcd_buffer[3] = (uint8_t)(data/1000);
	data = data % 1000;
	bcd_buffer[2] = (uint8_t)(data/100);
	data = data % 100;
	bcd_buffer[1] = (uint8_t)(data/10);
	data = data % 10;
	bcd_buffer[0] = (uint8_t)(data);
}
void SendData(uint8_t data)
{
	for(uint8_t i=0; i<8; i++)
	{
		PORTB &= ~(1<<PINB5);
```

```
                if(0x80 & (data<<i))
                {
                        PORTB |= 1<<PINB3; //DAT high
                }
                else
                {
                        PORTB &= ~(1<<PINB3);
                }
                PORTB |= (1<<PINB5);
        }
}
void DisplayData(uint16_t data)
{
        Bin2Dec(data);
        PORTB &= ~(1<<PINB1);
        SendData(segments[bcd_buffer[0]]);
        SendData(segments[bcd_buffer[1]]);
        SendData(segments[bcd_buffer[2]]);
        SendData(segments[bcd_buffer[3]]);
        PORTB |= (1<<PINB1);
}
```



Код для этой же схемы с использованием SPI:

```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
uint8_t segments[]={
	0b00111111, // 0 - A, B, C, D, E, F
	0b00000110, // 1 - B, C
	0b01011011, // 2 - A, B, D, E, G
	0b01001111, // 3 - A, B, C, D, G
	0b01100110, // 4 - B, C, F, G
	0b01101101, // 5 - A, C, D, F, G
	0b01111101, // 6 - A, C, D, E, F, G
	0b00000111, // 7 - A, B, C
	0b01111111, // 8 - A, B, C, D, E, F, G
	0b01101111, // 9 - A, B, C, D, F, G
};
void InitPorts(void);
void InitTimer0(void);
void Bin2Dec(uint16_t data);
void InitTimer1(void);
void StartTimer1(void);
void StopTimer1(void);
void InitSPI(void);
void SPI_send(uint8_t data);
void DisplayData(uint16_t data);
volatile uint16_t cnt = 0;
volatile uint8_t switch_state = 0;
volatile uint8_t bcd_buffer[] = {0,0,0,0};
int main(void)
{
	InitPorts();
	InitTimer1();
	EIMSK |= (1<<INT0); //разрешить прерывание INT0
	EICRA |= (1<<ISC01);//Запуск по заднему фронту INT0
	sei(); //Разрешение прерываний
```

```
	PORTB &= ~(1<<PB0); //OE = low (active)
	InitSPI();
	DisplayData(0);
	while(1)
	{ }
}
//-----------------------------------------
ISR(TIMER1_COMPA_vect){
	DisplayData(cnt);
	if(cnt<9999){
		cnt++;
		}else{
		cnt=0;
	}
}
ISR(INT0_vect){
	if(switch_state == 0){
		switch_state = 1;
		StartTimer1();
		}else{
		StopTimer1();
		DisplayData(cnt);
		switch_state = 0;
		cnt = 0;
	}
}
//-----------------------------------------
void InitPorts(void){
	DDRB = (1<<PB0|1<<PB1|1<<PB3|1<<PB5);
	DDRD &= ~(1<<PD2);
	PORTD |= (1<<PD2);
}
void InitTimer1(void){
	TCCR1A = 0;
	TCCR1B = (1<<CS11|1<<CS10|1<<WGM12);
	TCNT1 = 0;
	OCR1A = 15624;
}
void StartTimer1(void){
	TCNT1 = 0;
	TIMSK1 |= (1<<OCIE1A);
}
void StopTimer1(void){
	TIMSK1 &= ~(1<<OCIE1A);
}
void Bin2Dec(uint16_t data){
	bcd_buffer[3] = (uint8_t)(data/1000);
	data = data % 1000;
	bcd_buffer[2] = (uint8_t)(data/100);
	data = data % 100;
	bcd_buffer[1] = (uint8_t)(data/10);
	data = data % 10;
	bcd_buffer[0] = (uint8_t)(data);
}
void InitSPI(void){
	DDRB |= (1<<PB3 | 1<<PB5);//настроить MOSI и CLK как выходы
	SPSR |= (1<<SPI2X); //Fclk = Fosc/2
	SPCR = (1<<SPE | 1<<MSTR); //SPI включен, мастер,
	//MSB первый, CPOL=0, CPHA=0
	PORTB &= ~(1<<PB3 | 1<<PB5); //инициализация: DAT=0, CLK=0
}
void SPI_send (uint8_t data){
	SPDR = data;
	while(!(SPSR & (1<<SPIF)));
}
```

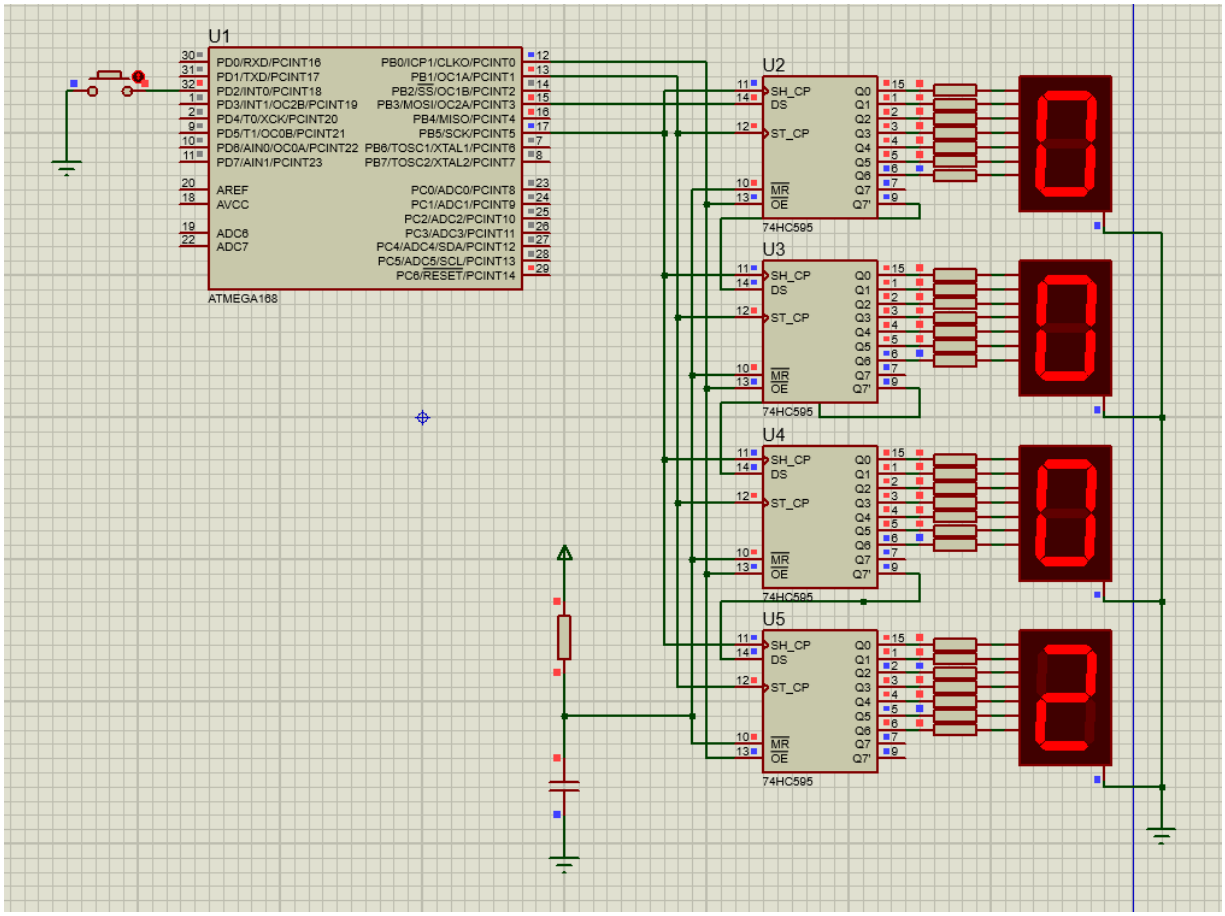
```
void DisplayData(uint16_t data){
	Bin2Dec(data);
	PORTB &= ~(1<<PB1); //clk_out = 0
	SPI_send(segments[bcd_buffer[0]]);
	SPI_send(segments[bcd_buffer[1]]);
	SPI_send(segments[bcd_buffer[2]]);
	SPI_send(segments[bcd_buffer[3]]);
	PORTB |= (1<<PB1); //clk_out = 1
}
```

Оптимизированный код:

```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h> uint8_t segments[]=
{
	0b00111111, // 0 - A, B, C, D, E, F
	0b00000110, // 1 - B, C
	0b01011011, // 2 - A, B, D, E, G
	0b01001111, // 3 - A, B, C, D, G
	0b01100110, // 4 - B, C, F, G
	0b01101101, // 5 - A, C, D, F, G
	0b01111101, // 6 - A, C, D, E, F, G
	0b00000111, // 7 - A, B, C
	0b01111111, // 8 - A, B, C, D, E, F, G
	0b01101111, // 9 - A, B, C, D, F, G
};
void InitPorts(void);
void send_data(uint8_t data, uint8_t ind);
void InitTimer0(void);
void Bin2Dec(uint16_t data);
void InitTimer1(void);
void StartTimer1(void);
void StopTimer1(void);
void SendData(uint8_t data);
void DisplayData(uint16_t data);
volatile uint16_t cnt = 0;
volatile uint8_t switch_state = 0;
volatile uint8_t bcd_buffer[] = {0,0,0,0};
	int main(void)
{
	InitPorts();
	InitTimer1();
	InitSPI();
	EIMSK |= (1<<INT0); //разрешить прерывание INT0
	EICRA |= (1<<ISC01);//Запуск по заднемуфронту INT0
	sei();
	//Разрешение прерываний
	PORTB &= ~(1<<PB0); //OE = low (active)
	DisplayData(0);
	while(1)
	{
	}
}
//-------------------------------------------
ISR(TIMER1_COMPA_vect)
{
	DisplayData(cnt);
	if(cnt<9999)
	{
		cnt++;
	}
	else
	{
```

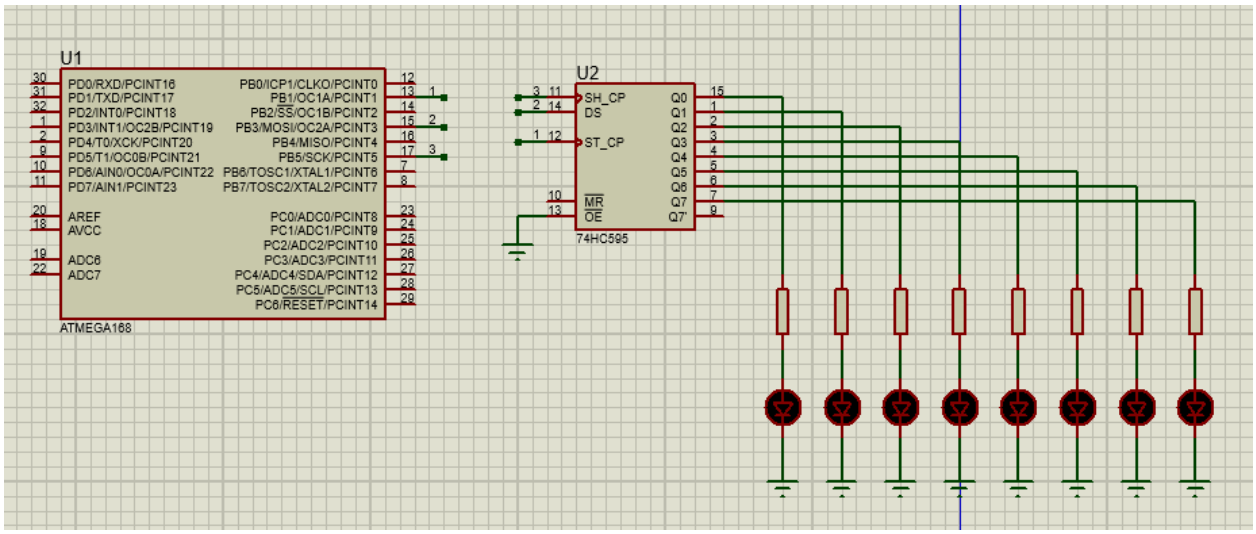
```
                cnt=0;
        }
}
ISR(INT0_vect)
{
        if(switch_state == 0)
        {
                switch_state = 1;
                StartTimer1();
        }
        else
              StopTimer1();
              DisplayData(cnt);
              switch_state = 0;
              cnt = 0;
        }
}
void InitSPI(void) {
        DDRB |= (1<<PINB3 | 1<<PINB5);//настроить MOSI и CLK как выходы
        SPSR |= (1<<SPI2X);
        //Fclk = Fosc/2
        SPCR = (1<<SPE | 1<<MSTR); //SPI включен, мастер,
        //MSB первый, CPOL=0, CPHA=0
        PORTB &= ~(1<<PINB3 | 1<<PINB5);
        //инициализация: DAT=0, CLK=0
}
void SPI_send (uint8_t data)
{
        SPDR = data;
        while(!(SPSR & (1<<SPIF)));
}
void InitPorts(void)
{
        DDRB = (1<<PINB0|1<<PINB1|1<<PINB3|1<<PINB5);
        DDRD &= ~(1<<PIND2);
        PORTD |= (1<<PIND2);
}
void InitTimer1(void)
{
        TCCR1A = 0;
        TCCR1B = (1<<CS11|1<<CS10|1<<WGM12);
        TCNT1 = 0;
        OCR1A = 15624;
}
void StartTimer1(void)
{
        TCNT1 = 0;
        TIMSK1 |= (1<<OCIE1A);
}
void StopTimer1(void)
{
        TIMSK1 &= ~(1<<OCIE1A);
}
void Bin2Dec(uint16_t data)
{
        bcd_buffer[3] = (uint8_t)(data/1000);
        data = data % 1000;
        bcd_buffer[2] = (uint8_t)(data/100);
        data = data % 100;
        bcd_buffer[1] = (uint8_t)(data/10);
        data = data % 10;
        bcd_buffer[0] = (uint8_t)(data);
}
void DisplayData(uint16_t data)
{
```

```
        Bin2Dec(data);
        PORTB &= ~(1<<PINB1);
        //DAT low
        //CLK high
        //clk_out = 0
        SPI_send(segments[bcd_buffer[0]]);
        SPI_send(segments[bcd_buffer[1]]);
        SPI_send(segments[bcd_buffer[2]]);
        SPI_send(segments[bcd_buffer[3]]);
        PORTB |= (1<<PINB1);
        //clk_out = 1
}
```



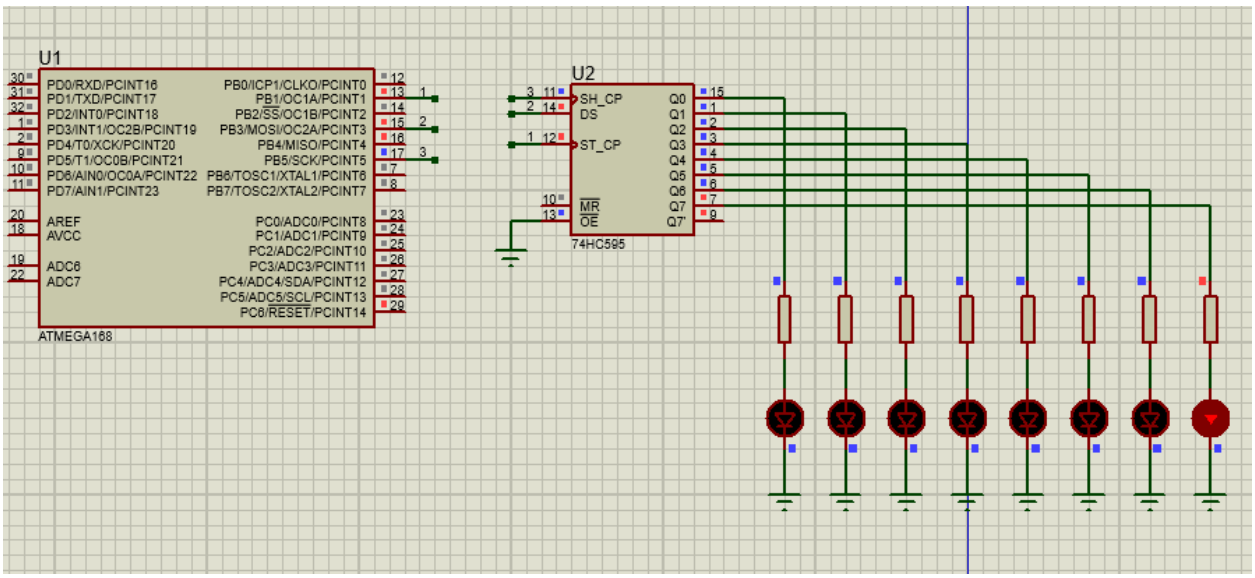
Задание из методички для 2 варианта:

Изучить документацию на WS2801 (для реализации на отладочной плате) или 74HC595. Реализовать визуальные эффекты на нескольких светодиодах (использовать аппаратный SPI).




```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#define LATCH_PIN  PORTB1
#define SPI_DATA   PORTB3
#define SPI_CLOCK  PORTB5

void InitPorts(void);
void InitTimer0(void);
void InitTimer1(void);
void StartTimer1(void);
void StopTimer1(void);
void InitSPI(void);
void SPI_send(uint8_t data);
void DisplaySequence(uint8_t data);

volatile uint16_t cnt = 0;
volatile uint8_t switch_state = 0;

int main(void) {
	InitPorts();
	InitTimer1();
	EIMSK |= (1 << INT0); // Разрешить прерывание INT0
	EICRA |= (1 << ISC01); // Запуск по заднему фронту INT0
	sei(); // Разрешение прерываний
	InitSPI();

	while (1) {
		for (uint8_t i = 0; i < 8; i++) {
			DisplaySequence(1 << i); // Загорается один светодиод
			_delay_ms(250);
```

```
                }
                for (uint8_t i = 0; i < 8; i++) {
                        DisplaySequence(1 << (7 - i)); // Затухает один светодиод
                        _delay_ms(250);
                }
        }
}

ISR(INT0_vect) {
        if (switch_state == 0) {
                switch_state = 1;
                StartTimer1();
                } else {
                StopTimer1();
                switch_state = 0;
                cnt = 0;
        }
}

//-------------------------------------------
void InitPorts(void) {
        DDRB = (1 << LATCH_PIN | 1 << SPI_DATA | 1 << SPI_CLOCK); // Настройка пинов как
выходы
        PORTB &= ~(1 << LATCH_PIN); // Инициализация защелки в LOW
}

void InitTimer1(void){
        TCCR1A = 0;
        TCCR1B = (1<<CS11|1<<CS10|1<<WGM12);
        TCNT1 = 0;
        OCR1A = 15624;
}
void StartTimer1(void){
        TCNT1 = 0;
        TIMSK1 |= (1<<OCIE1A);
}
void StopTimer1(void){
        TIMSK1 &= ~(1<<OCIE1A);
}

void InitSPI(void) {
        DDRB |= (1 << SPI_DATA | 1 << SPI_CLOCK); // Настройка MOSI и SCK как выходы
        SPSR |= (1 << SPI2X); // Установка скорости SPI
        SPCR = (1 << SPE | 1 << MSTR); // SPI включен, мастер
        PORTB &= ~(1 << SPI_DATA | 1 << SPI_CLOCK); // Инициализация: DAT=0, CLK=0
}

void SPI_send(uint8_t data) {
        SPDR = data;
        while (!(SPSR & (1 << SPIF))); // Ожидание завершения передачи
}

void DisplaySequence(uint8_t data) {
        PORTB &= ~(1 << LATCH_PIN); // Открыть защелку
        SPI_send(data); // Передать данные на 74HC595
        PORTB |= (1 << LATCH_PIN); // Закрыть защелку
}
```



Светодиоды горят слева на право а затем справа на лево

Вывод:

Изучил принципы вывода информации на индикаторы, примеры использования встроенных таймеров и основы последовательной передачи данных