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

Факультет информатики и
вычислительной техники

Кафедра ИиСП

Отчет
по лабораторной работе № 5

по дисциплине «Машинно-зависимые языки программирования»
Вариант 2

Выполнил: студент группы ПС-11
Щеглов Г.С
Проверил: Баев А.А.

г. Йошкар-Ола
2024

Цель работы: научиться работать в Proteus и писать для него код, собрать схемы различных подключений к ATmega328p

Задания на лабораторную работу:

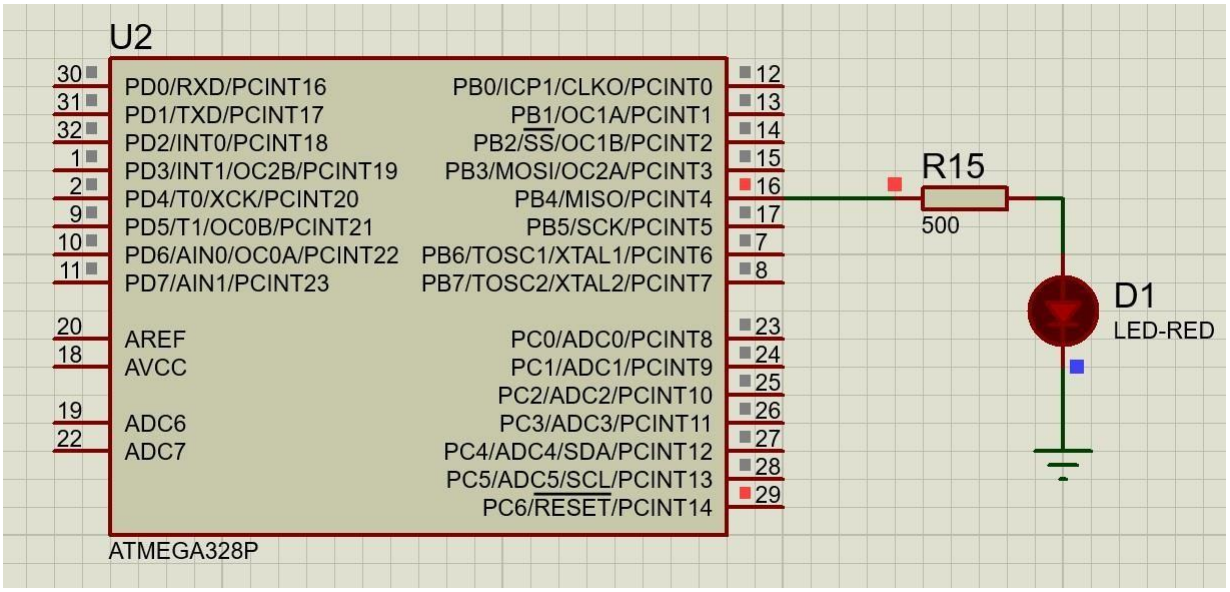
1. Схема и код для подключения светодиода
2. Схема и код для подключения 7-сегментного индикатора
3. Схема и код для подключения кнопки
4. Реализация секундомера
5. Реализация внешнего прерывания
6. Счетчик нажатий на кнопку до 15

1. **Теоретические сведения**

   Учебное пособие - ПРИМЕНЕНИЕ МИКРОКОНТРОЛЛЕРОВ В РАДИОТЕХНИЧЕСКИХ И БИОМЕДИЦИНСКИХ СИСТЕМАХ

2. Практическая часть

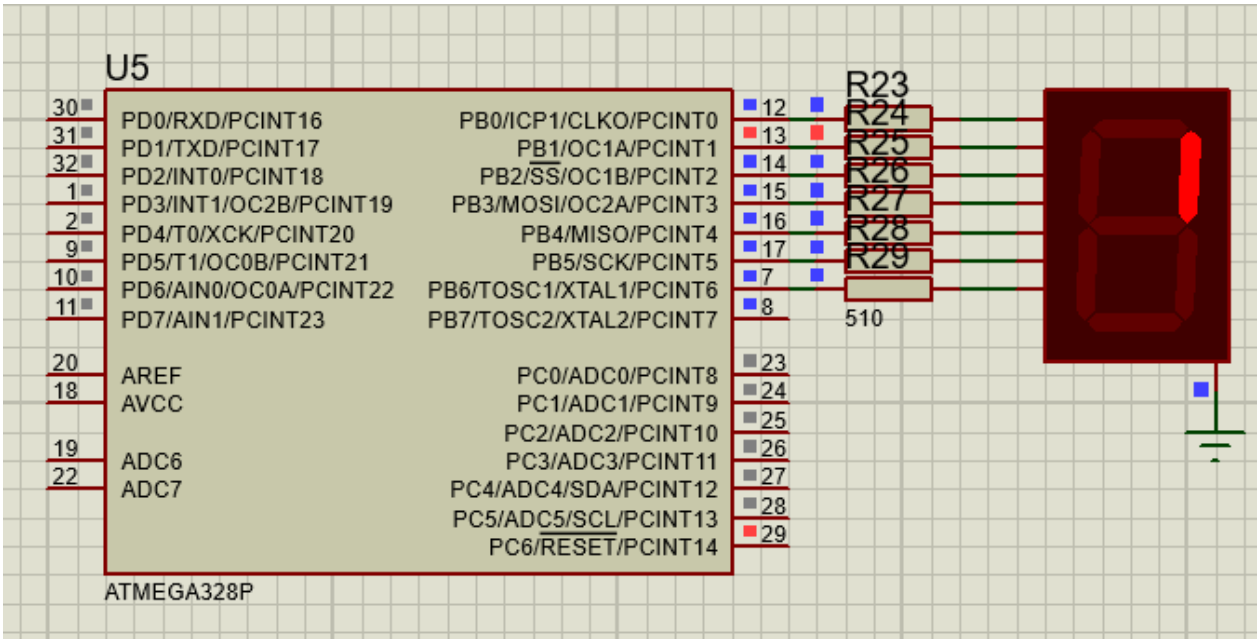
Схема со светодиодом:



Код на C:

```
#include <avr/io.h>
int main(void)
{
  DDRB = 0b00010000;
  PORTB = 0b00010000;
  while(1){}
}
```

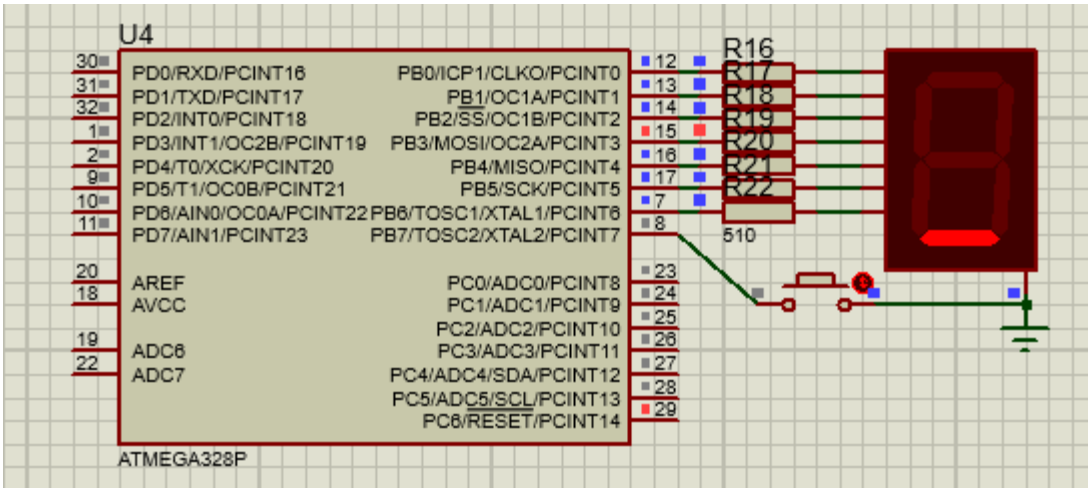
Схема с 7-сегментным индикатором:



Код на C:

```c
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>
int main(void)
{
  DDRB = 0xFF;
  while(1)
  {
    for( int i=0; i<6; i++){
      PORTB = (1<<i);
      _delay_ms(200);
    }
  }
}
```

Схема подключения кнопки:



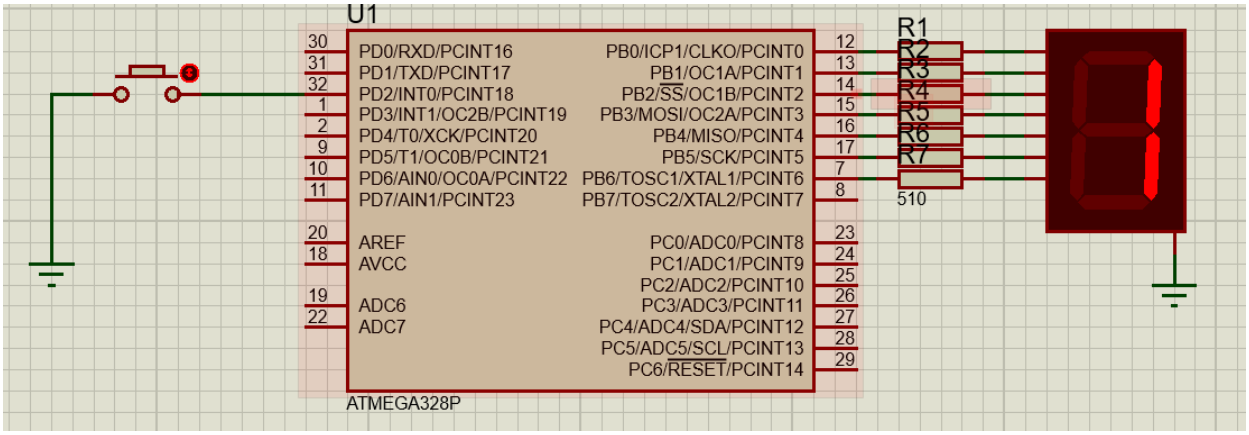
Код на C:

```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>
int main(void)
{
  DDRB = 0xFF & ~(1<<PB7); // PB7 – вход, остальные - выходы
  PORTB |= (1<<PB7); // включение подтягивающего резистора
  int button = 0; // вспомогательная переменная
  while(1)
  {
    for(int i = 0; i < 6; i++){
      button = PINB & (1<<PB7); //чтение состояния PB7
      if(button != 0){
        PORTB = (1<<i);
      }else{
        PORTB = (0x20>>i);
      }
      _delay_ms(200);
    }
  }
}
```

Оптимизированный код:

```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>
int main(void) {
  DDRB = 0xFF & ~(1 << PB7);
  PORTB |= (1 << PB7);
  while (1) {
    for (int i = 0; i < 6; i++) {
      if (PINB & (1 << PB7)) {
        PORTB = (1 << i);
                    } else {
        PORTB = (0x20 >> i);
                    }
      _delay_ms(200);
              }
  }
}
```

Схема с реализованным секундомером:



Код на C:

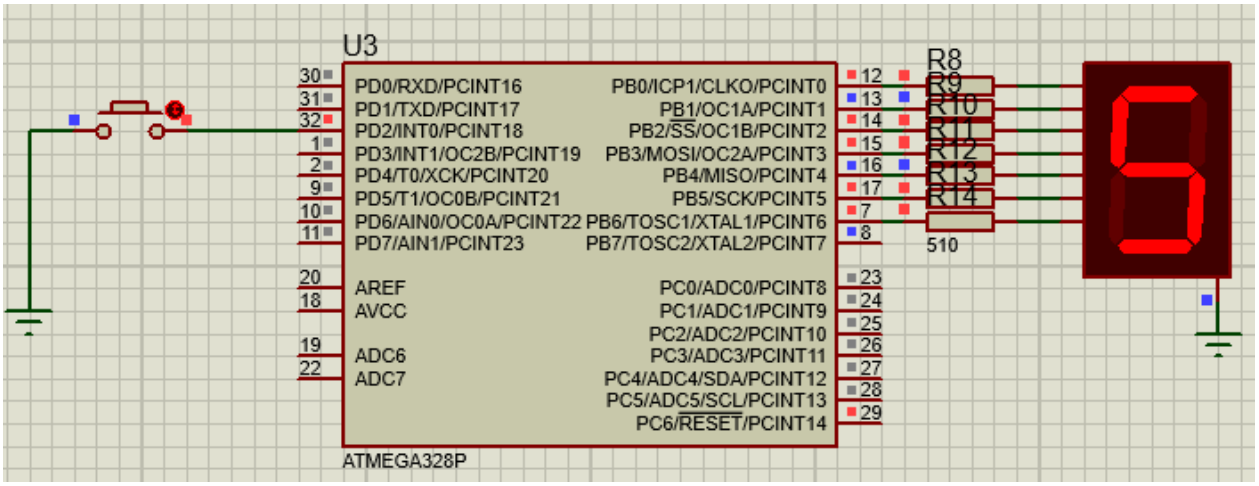
```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>
uint8_t segments[]={
  //  ___GFEDCBA
  0b00111111, // 0 - A, B, C, D, E, F
  0b00000110, // 1 - B, C
  0b01011011, // 2 - A, B, D, E, G
  0b01001111, // 3 - A, B, C, D, G
  0b01100110, // 4 - B, C, F, G
```

```
  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
};

int main(void)
{
  DDRB = 0xFF;  // порт B на выход
  DDRD &= ~(1<<PD2); // вывод PD2 на вход
  PORTD |= (1<<PD2); // подтяжка PD2

  int button = 0;
  int switch_state = 0;
  int counter = 0;
  while(1)
  {
    button = PIND & (1<<PD2); //опрос

    if(button == 0){
      while((PIND & (1<<PD2)) ==
      0);//ожидание отпускания
      if(switch_state == 0){
        switch_state = 1;
        } else {
        switch_state = 0;
        counter = 0;
      }
    }
    if(switch_state == 0){
      if(counter < 10){
        PORTB = segments[counter++];
        _delay_ms(1000);
        } else {
        counter = 0;
        PORTB = segments[counter++];
        _delay_ms(1000);
      }

    }
  }
}
```

Оптимизированный код:

```c
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>
uint8_t segments[]={
  //  ___GFEDCBA
  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
};

int main(void) {
  DDRB = 0xFF;
  DDRD &= ~(1 << PD2);
  PORTD |= (1 << PD2);

  uint8_t counter = 0;
  uint8_t switch_state = 0;

  while (1) {
    if (!(PIND & (1 << PD2))) {
      while (!(PIND & (1 << PD2)));
      switch_state ^= 1;
      if (!switch_state) counter = 0;
    }

    if (switch_state) {
      PORTB = segments[counter];
      _delay_ms(1000);
      counter = (counter + 1) % 10;
    }
  }
}
```

Схема с внешним прерыванием:



Код на C:

```c
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
uint8_t segments[]={ //0b01111111
  //____GFEDCBA
  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
};
volatile int button = 0;
volatile int switch_state = 0;
volatile int counter = 0;
ISR(INT0_vect){ // Обработчик прерывания
  if(switch_state == 0){
    switch_state = 1;
  } else {
    switch_state = 0;
    counter = 0;
  }
```

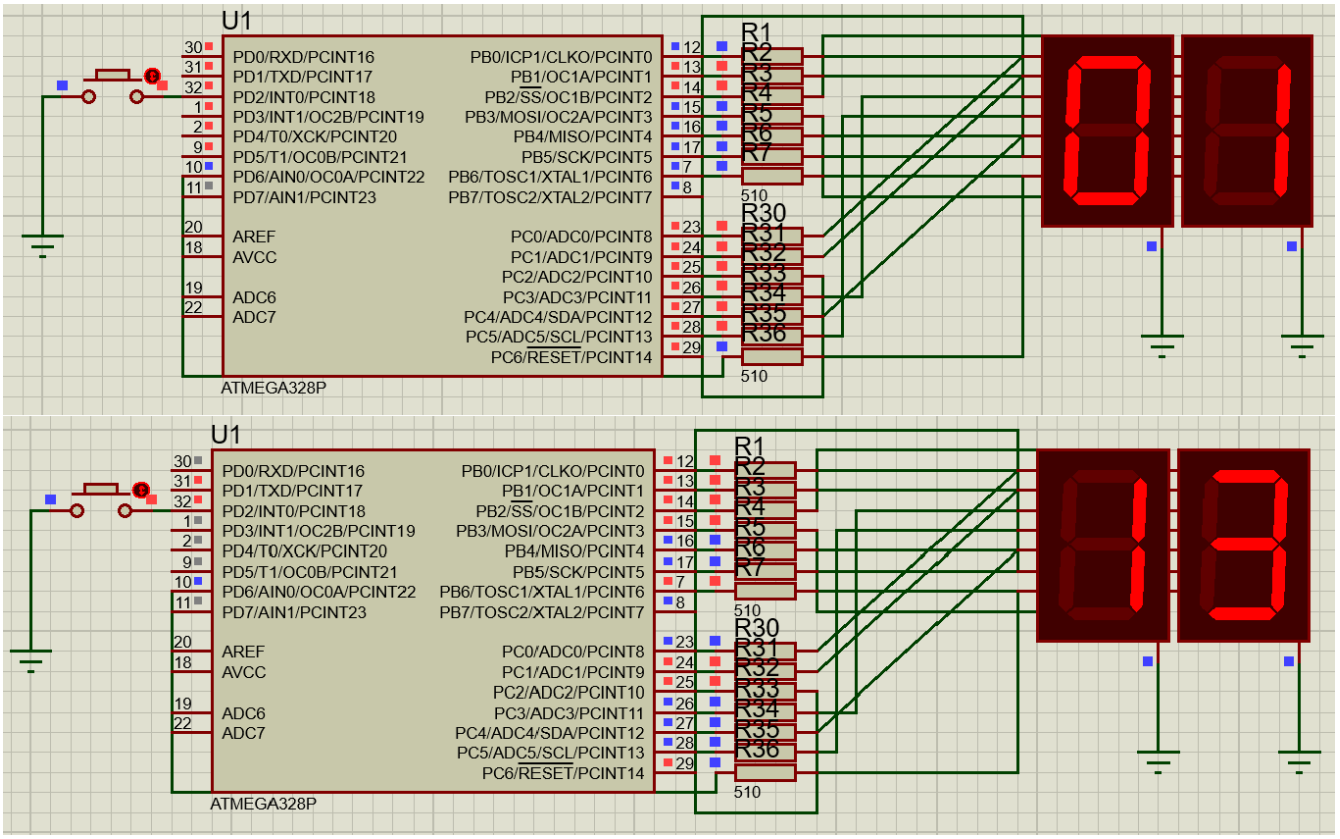
```
}
int main(void) {
  DDRB  =  0xFF;
  PORTD |= (1<<PD2);
  EIMSK |= (1<<INT0); //Включаем INT0
  EICRA |= (1<<ISC01); //Прерывание по спадающему фронту
INT0
  sei(); //Глобальное разрешение прерываний
  while(1){
    if(switch_state == 0){
      if(counter < 10){
        PORTB = segments[counter];
        counter += 1;
        _delay_ms(500);
    } else {
        counter = 0;
        PORTB = segments[counter];
        counter += 1;
        _delay_ms(500);
      }
    }
  }
}
```

Оптимизированный код:

```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
uint8_t segments[] = {
  // GFEDCBA
  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
};
volatile int switch_state = 0;
volatile int counter = 0;
```

```
ISR(INT0_vect) {
  switch_state ^= 1;
  if (switch_state == 0) {
    counter = 0;
  }
}
int main(void) {
  DDRB = 0xFF;
  PORTD |= (1 << PD2);
  EIMSK |= (1 << INT0);
  EICRA |= (1 << ISC01);
  sei();
  while (1) {
    if (switch_state == 0) {
      PORTB = segments[counter];
      counter = (counter + 1) % 10;
      _delay_ms(500);
    }
  }
}
```

Схема со счетчиком до 15 нажатий и двумя 7-сегментными индикаторами (2 вариант):

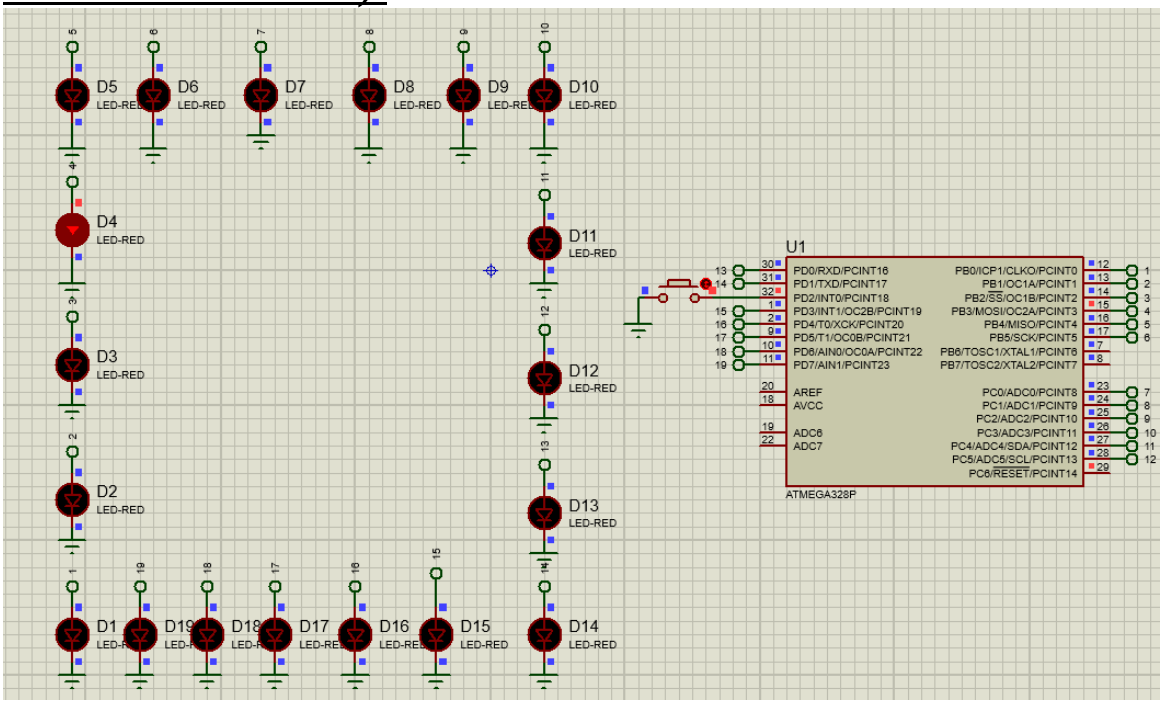


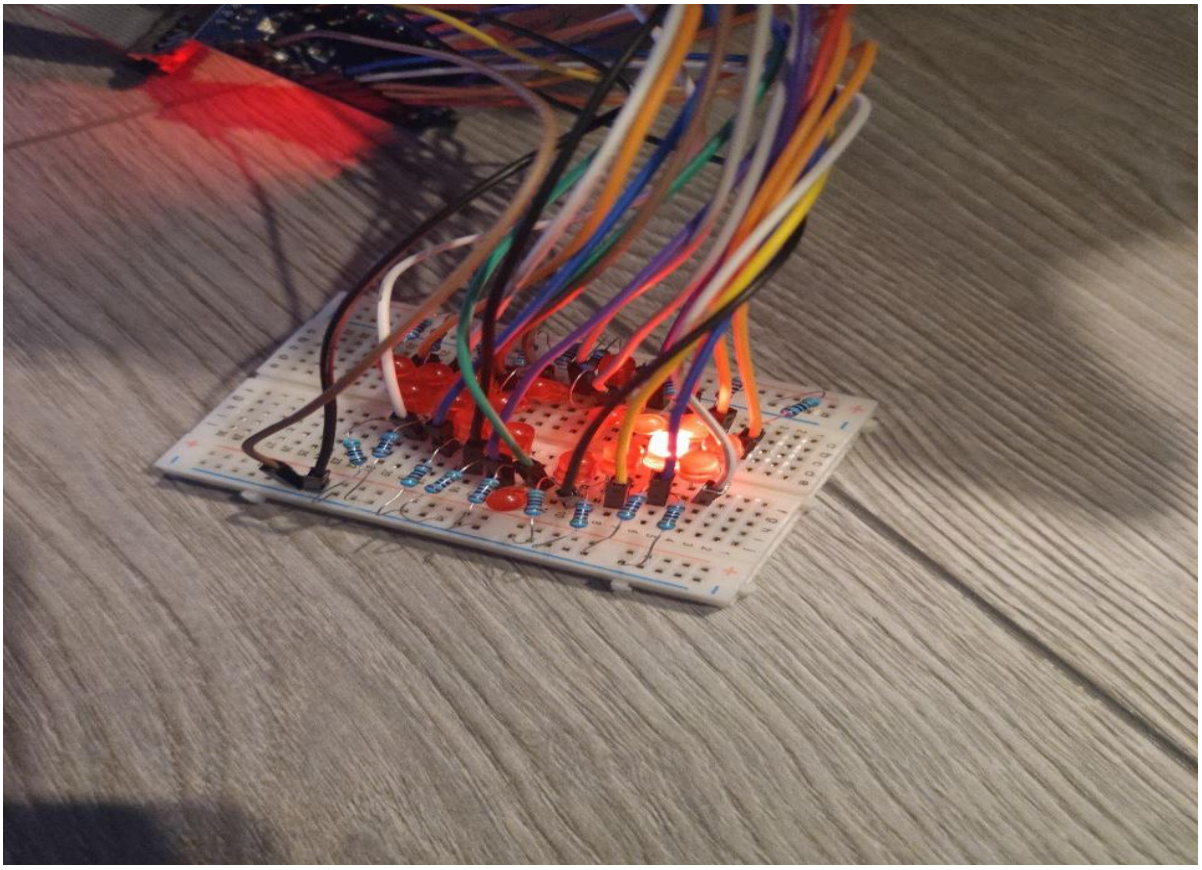
Код на C:

```c
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
uint8_t segments[]={ //0b01111111
     //  ___GFEDCBA
     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
};
volatile int counter = 0;
ISR(INT0_vect){
     if (counter < 15){
          counter += 1;
     } else{
          counter = 0;
     }
}
int main(void)
{
     DDRB = 0xFF;
     DDRC = 0xFF;
     DDRD |= (1 << PD6);
     PORTD |= (1<<PD2);
     EIMSK |= (1<<INT0);
     EICRA |= (1<<ISC01);
     sei();
     while(1){
          PORTB = segments[counter % 10];
          PORTC = segments[counter / 10];
          PORTD = segments[counter / 10];
     }
}
```

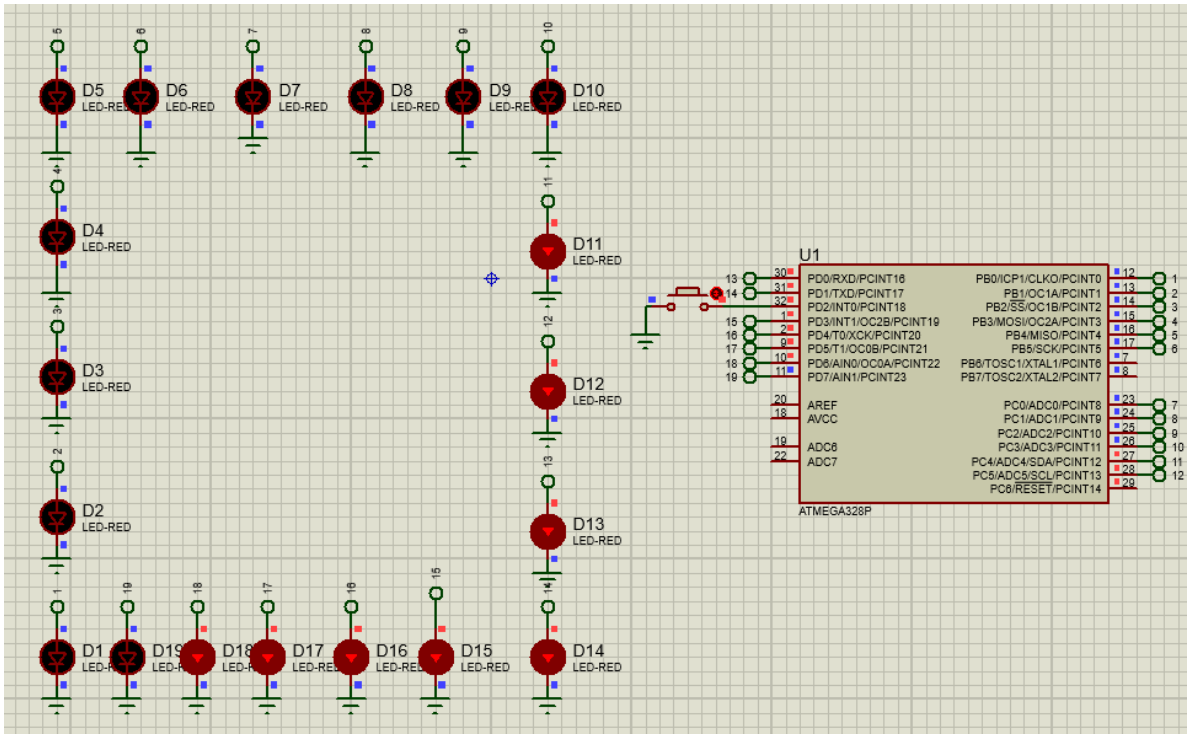
Задание от преподавателя:
Вариант 26. Форма – квадрат. Эффекты 1,8,4

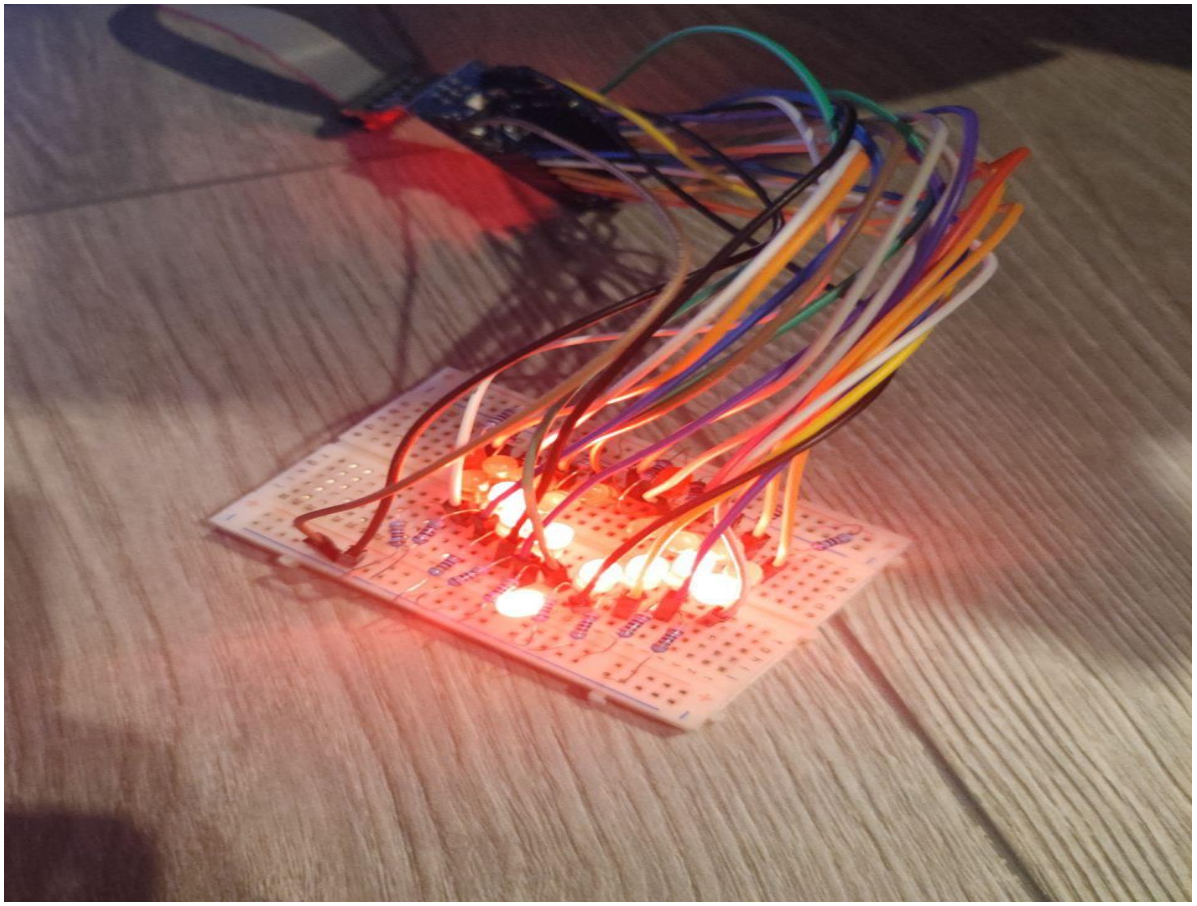
Эффект 1 (Пробежка в обе стороны 0->1->2->...-> count-1 -> count-2 -> ... -> 0):



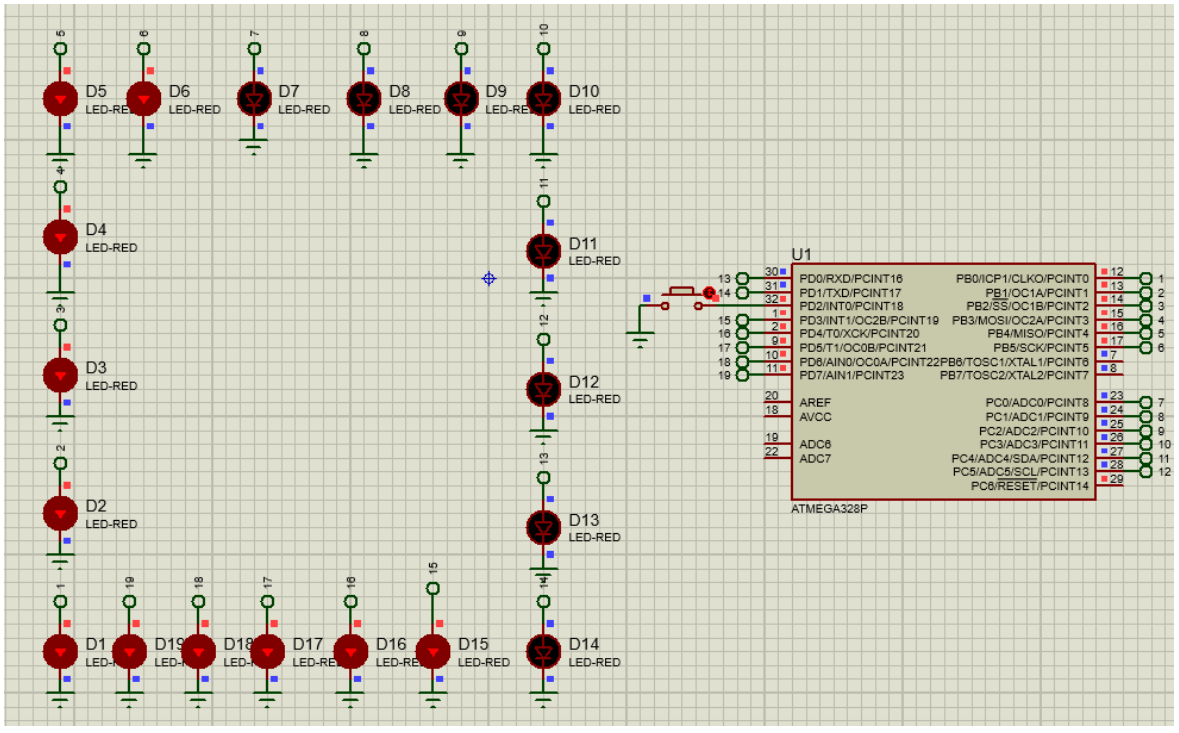


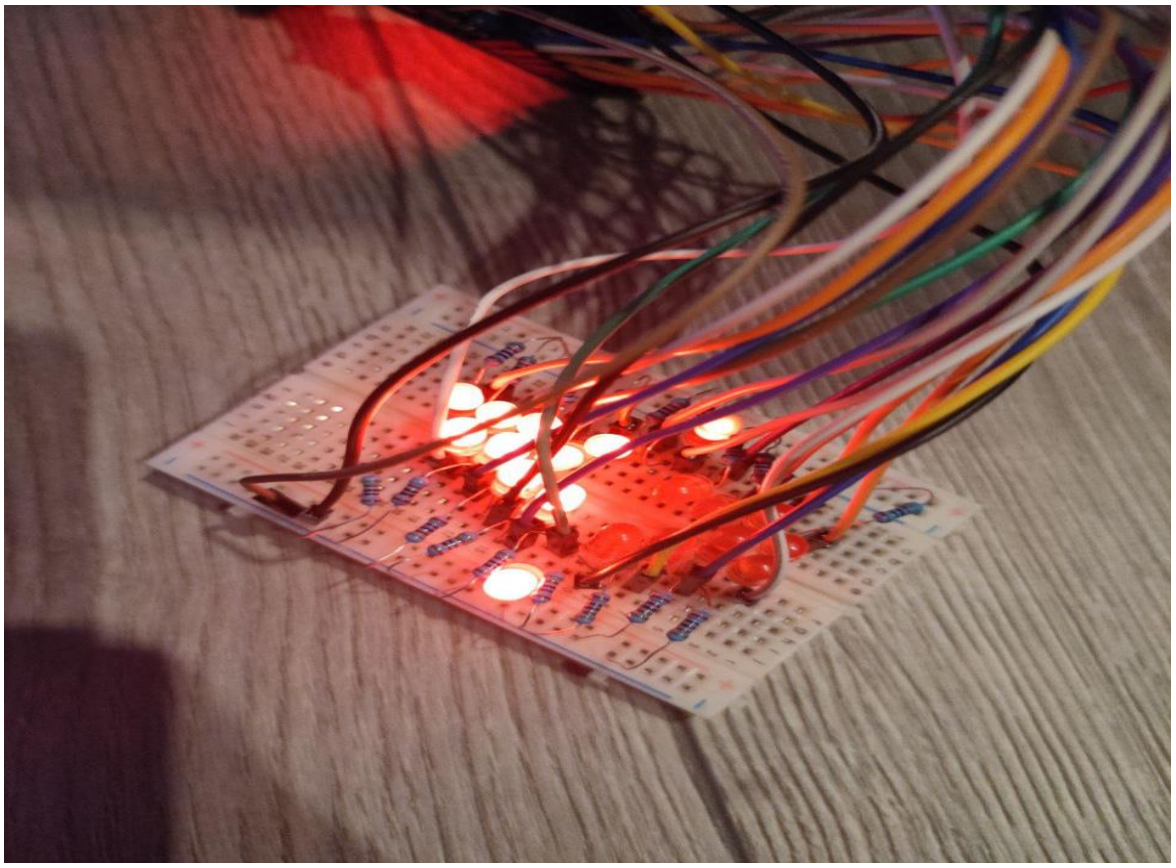
Эффект 8 (Пробежка по 2 диода. В конечном положении должно светится заданное число диодов (эффект отскока):





Эффект 4 (Накопление и убывание к центру):





Код на C:

```
#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>

volatile uint32_t frame = 0;
uint8_t step = 0;
uint8_t mode = 0;
uint8_t direction = 1;
const uint32_t bit = 1;
const uint8_t bounce_size = 2;
uint8_t bounce_flag = 0;

void switch_mode()
{
  mode++;
  if (mode > 2)
  {
    mode = 0;
  }
```

```
  frame = 0;
  step = 0;
  direction = 1;
  bounce_flag = 0;
}


void frame_creation()
{
  switch (mode)
  {
    case 0:
    {
      frame = bit << step;
      if (direction) {
        step++;
        if (step > 19) {
          step = 19;
          direction = 0;
        }
        } else {
        step--;
        if (step == 0) {
          direction = 1;
        }
      }
    }
    break;

    case 1:
    {
      if (bounce_flag) {
        frame = 0;
        bounce_flag = 0;
        if (direction) {
          step = bounce_size;
          } else {
          step = 18 - bounce_size;
        }
        } else {
        frame |= (bit << step) | (bit << (step + 1));
        if (direction) {
          step += 2;
          if (step > 18) {
            bounce_flag = 1;
            direction = 0;
          }
```

```
        } else {
        if (step == 0) {
          bounce_flag = 1;
          direction = 1;
          } else {
          step -= 2;
        }
      }
    }
    break;

    case 2:
    {
      if (step < 10) {
        frame |= (bit << step) | (bit << (19 - step));
        } else if (step < 20) {
        uint8_t center = 10;
        uint8_t offset = step - 10;
        frame &= ~((bit << (center - offset)) | (bit << (center +
offset)));
      }
      step++;
      if (step == 20) {
        step = 0;
        frame = 0;
      }
    }
    break;
  }
}


void frame_output()
{
  PORTB = frame;
  PORTC = (frame >> 6);
  PORTD = (frame >> 12 & 0b00000011) | (frame >> 11 & 0b11111000)
| (1 << PIND2);
}


int main(void)
{
  DDRB = 0xFF;
  DDRC = 0xFF;
  DDRD = 0xFF;
```

```
  DDRD &= ~(1 << PIND2);


  EIMSK |= (1 << INT0);
  EICRA |= (1 << ISC01);
  sei();

  while (1)
  {
    frame_creation();
    frame_output();
    _delay_ms(15);
  }
}

ISR(INT0_vect)
{
  switch_mode();
}
```

Выводы: Мы познакомились с Proteus, научились писать для него код на C и собрали схемы различных подключений.