

Лабораторная работа №16:

Тема: Связь. Использование UART для последовательной связи.

Цель: Изучить принципы работы UART на микроконтроллерах AtMega.

Лабораторное оборудование:

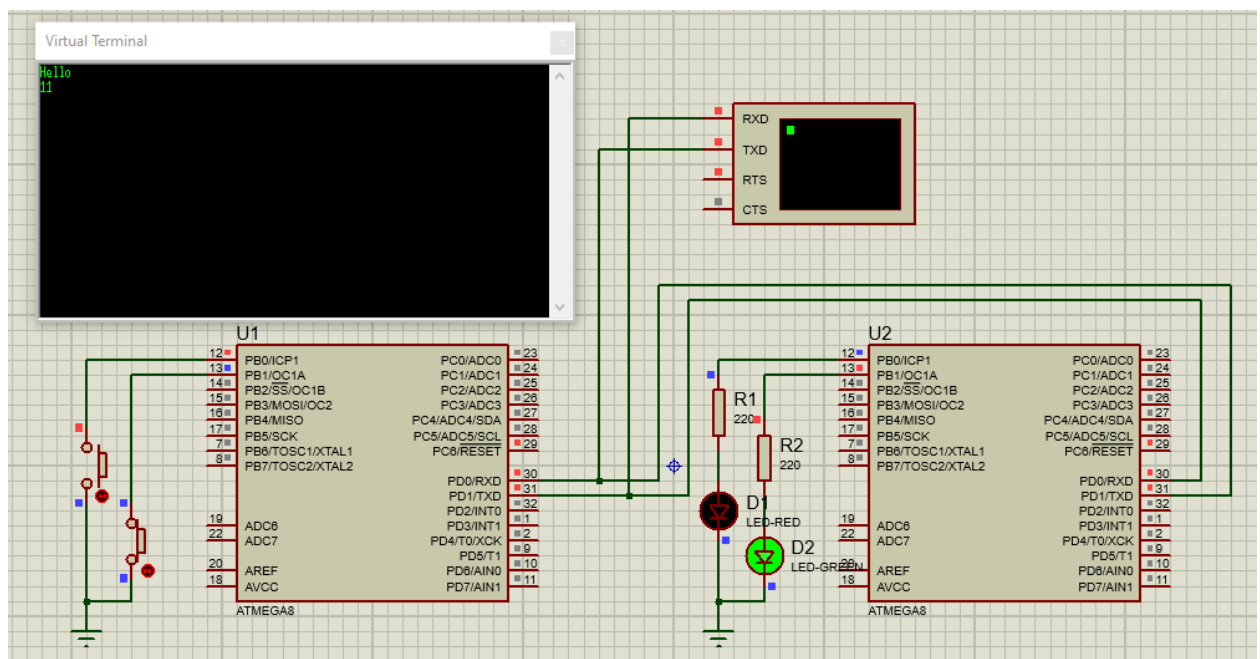
Компьютер с установленным программным обеспечением:

- Atmel Studio - для программирования микроконтроллеров AtMega.
- Proteus - для моделирования работы микроконтроллеров AtMega.

Задание:

1. В Atmel Studio написать программы на языке C, которые реализуют следующий функционал:
МК-1: при включении передает по UART строку "Hello". При нажатии кнопки №1 передает строку «0», при нажатии кнопки №2 – строку «1»
МК-2: принимает данные по UART. В зависимости от принятого символа, зажигает либо один светодиод, либо другой.
2. В Proteus:
Собрать схему, включающую две микросхемы AtMega8, виртуальный терминал.
Загрузить в микросхемы AtMega8 программы, написанную в п. 1.
Проверить работоспособность программы.
3. Изменить программы и схему. Добавить кнопки, что бы можно было отправлять цифры от 0 до 3. Приемник должен отображать полученное число с помощью 2-х светодиодов в двоичном виде. Т.е. 0 = 00, 1 = 01, 2 = 10, 3 = 11.

Пример собранной схемы:



Пример программы приемника. За основу взята программа из предыдущей лабораторной. Необходимо активировать бит RXEN в регистре UCSRB (разрешить прием).

Разрешить прерывания на прием данных, так как нам заранее не известно когда придут данные.

```
ISR(USART_RXC_vect){}
```

Потом надо считать данные с регистра UDR, когда они придут:

```
volatile char data;
ISR(USART_RXC_vect)
{
    data = UDR;
}
```

```
#define F_CPU 8000000 // рабочая частота микроконтроллера
#define UBRRL_value (F_CPU/(9600L*16))-1 //подсчитываем значение для
регистра UBRRL
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h> // включает функции для управления
прерываниями в микроконтроллерах AVR.
void init_pin(void);
// Управление светодиодами
#define HL1_on() PORTB|=(1<<PB0) // вкл. светодиод подкл. к пину PB0
#define HL1_off() PORTB&=~(1<<PB0) // выкл. свет-д подкл. к пину PB0
#define HL2_on() PORTB|=(1<<PB1) // вкл. светодиод подкл. к пину PB1
#define HL2_off() PORTB&=~(1<<PB1) // выкл. свет-д подкл. к пину PB1
volatile char data;
ISR(USART_RXC_vect)// прерывание на прием данных
{
    data = UDR; // считываем данные
}
void init_USART() {
    UBRRL = UBRRL_value; //Младшие 8 бит UBRRL_value
    UBRRH = UBRRL_value >> 8; //Старшие 8 бит UBRRL_value
    UCSRB |= (1<<RXEN); //Бит разрешения приема
    UCSRC |= (1<< URSEL)|(1<< UCSZ0)|(1<< UCSZ1); //Устанавливаем
формат 8 бит данных
    UCSRB |= (1<<RXCIE); // разрешаем прерывание по приему данных
    sei(); // разрешение глобального прерывания
}

void init_pin(void)
{
    PORTB=0b00000000; //PB0, PB1
    DDRB= 0b00000011; //PB0, PB1 output
}

int main(void)
```

```
{  
    init_pin();  
    init_USART(); //инициализация  
    while(1)  
    {  
        if(data=='0')// если символ 0  
        {  
            HL1_on();  
            HL2_off();  
            _delay_ms(500);  
        }  
        if(data=='1') // если символ 1  
        {  
            HL2_on();  
            HL1_off();  
            _delay_ms(500);  
        }  
    }  
}
```