|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное автономное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.01 Информатика и вычислительная техника**

**ОТЧЕТ**

**по лабораторной работе № 4**

**Название:** Обмен данными по протоколу UART

**Дисциплина:** Микропроцессорные системы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-63Б |  |  | В.К. Залыгин Р.В. Дорохов |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | Е.Ю. Гаврилова |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2025

### Цель работы

– Изучение структуры модуля UART в микроконтроллере AVR;  
 – Программирование передачи и приёма данных по асинхронному протоколу UART

**Задание 1. Передача данных между МК в Proteus.**

Передатчик по нажатию кнопки 'START' передает последовательно три байта {65, 86, 82} по UART. Приемник принимает эти байты и по нажатию кнопки 'SHOW' отображает их по очереди на светодиодном индикаторе.

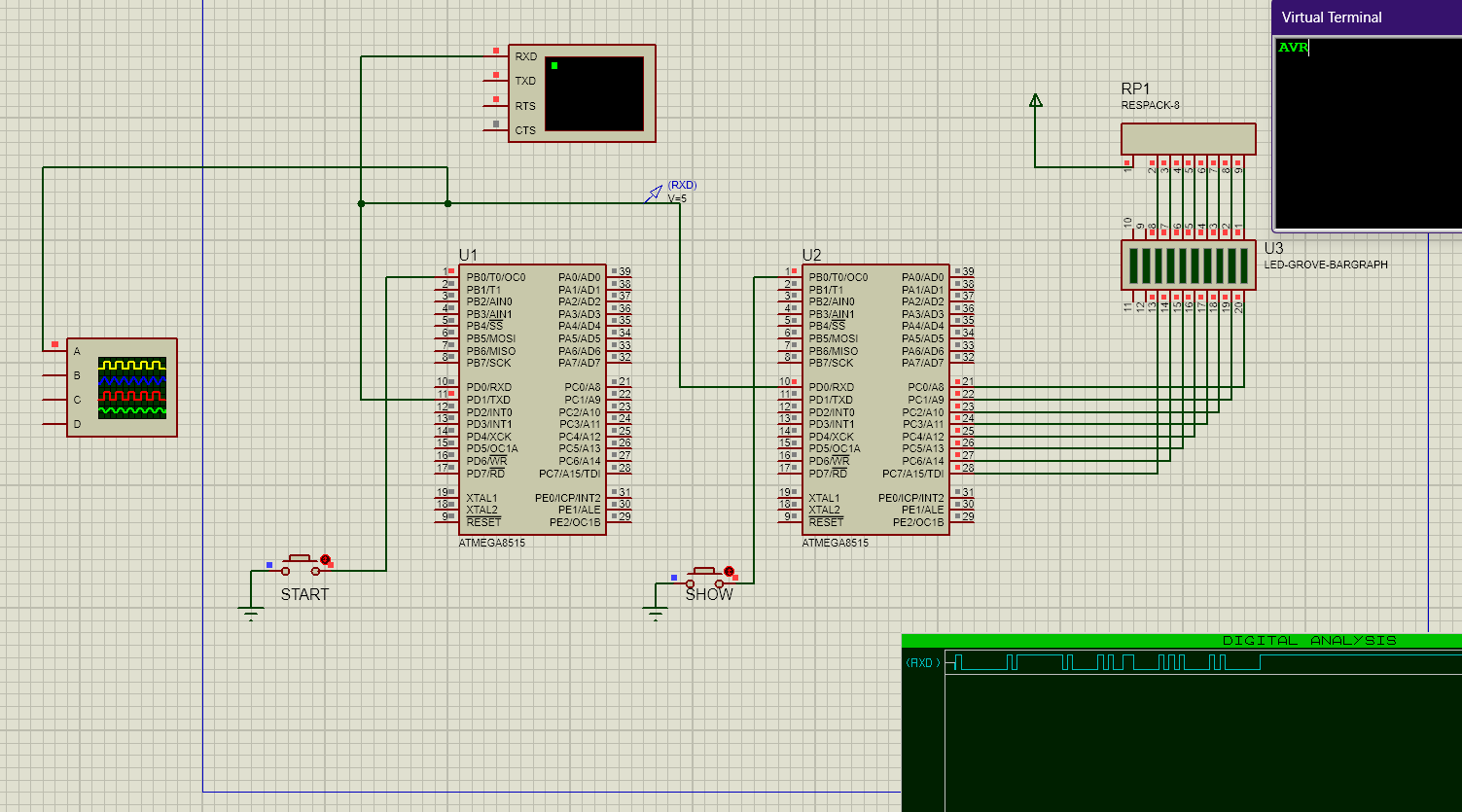


Рисунок 1 - схема с открытым окном виртуального терминала

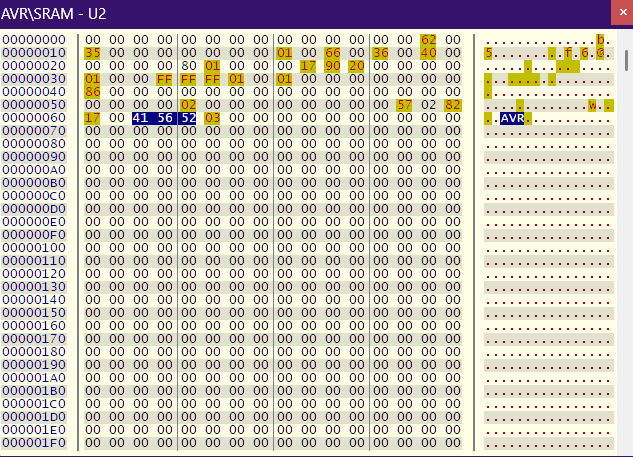


Рисунок 2 - скриншот содержимого памяти микроконтроллера-приёмника с выделенными байтами, которые были получены по UART

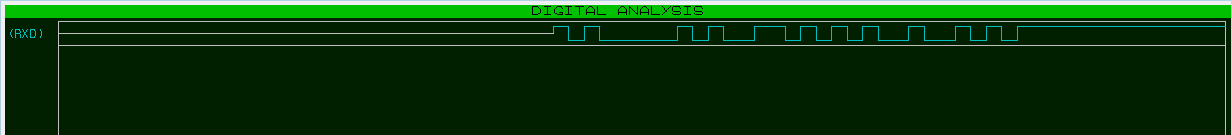


Рисунок 3 - Пакеты переданные по UART

### На осциллограмме видно стандартный UART‑кадр:

### Линия в покое (Mark) держится в уровне «1» до появления старт‑би­та.

### Затем линия опускается в «0» — это старт‑бит.

### После старта идут 8 бит данных, передаваемых LSB‑первым (младший бит передаётся первым):

Таблица 1 **-** Расшифровка кадров

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| N | Биты на линии (после старта, LSB→MSB) | В прямом порядке (MSB→LSB) | Hex | Dec |
| 1‑й | 1 0 0 0 0 1 0 0 | 01000001 | 0x41 | 65 |
| 2‑й | 0 1 1 0 1 0 1 0 | 01010110 | 0x56 | 86 |
| 3‑й | 0 1 0 0 1 0 1 0 | 01010010 | 0x52 | 82 |

Соответственно, кадры выглядят как:

[0] 01000001 [1] → 65

[0] 01010110 [1] → 86

[0] 01010010 [1] → 82

**Задание 2. Обработка прерывания UART.**

Листинг 1 - Код трансмиттера для задания 2

|  |
| --- |
| #include <avr/io.h>  #include <avr/interrupt.h>  #include <stdbool.h>  #define BUTTON\_START 0  const unsigned int ubrrValue = 23;  #define DATA\_LENGTH 3  const unsigned char data[DATA\_LENGTH] = {65, 86, 82};  uint8\_t counter = 0; // volatile or O0  bool flag = false; // volatile or O0  ISR(USART\_UDRE\_vect) {  if (flag)  UDR = data[counter++];  }  int main() {  UBRRH = (unsigned char)(ubrrValue>>8);  UBRRL = (unsigned char) ubrrValue;  UCSRB = (1<<TXEN) | (1<<UDRIE);  UCSRC = (1<<URSEL) | (3<<UCSZ0);  PORTB = (1<<BUTTON\_START);  sei();  while (1) {  if (!(PINB & (1<<BUTTON\_START))) {  while (!(PINB & (1 << BUTTON\_START)))  ;  flag = true;  while (counter < 3)  ;  counter = 0;  }  flag = false;  }  return 0;  } |

Листинг 2 - Код ресивера для задания 2

|  |
| --- |
| #include <avr/io.h>  #include <avr/interrupt.h>  #define BUTTON\_SHOW 0  const unsigned int ubrrValue = 23;  #define DATA\_LENGTH 3  unsigned char data[DATA\_LENGTH] = { 0 };  uint8\_t receivedBytes = 0;  ISR(USART\_RX\_vect) {  if (receivedBytes < DATA\_LENGTH) {  data[receivedBytes++] = UDR;  }  }  int main() {  UBRRH = (unsigned char)(ubrrValue>>8);  UBRRL = (unsigned char)ubrrValue;  UCSRB = (1<<RXEN) | (1<<RXCIE);  UCSRC = (1<<URSEL) | (3<<UCSZ0);  PORTB = (1<<BUTTON\_SHOW);  DDRC = 0xFF;  PORTC = 0xFF;  sei();  uint8\_t i = 0;  while (1) {  if (!(PINB & (1<<BUTTON\_SHOW))) {  while (!(PINB & (1<<BUTTON\_SHOW)))  ;  PORTC = ~data[i];  i = (i + 1) % DATA\_LENGTH;  }  }  return 0;  } |

**Задание 4. Обмен данными между МК и персональным компьютером.**

- с сообщениями (задание 4);

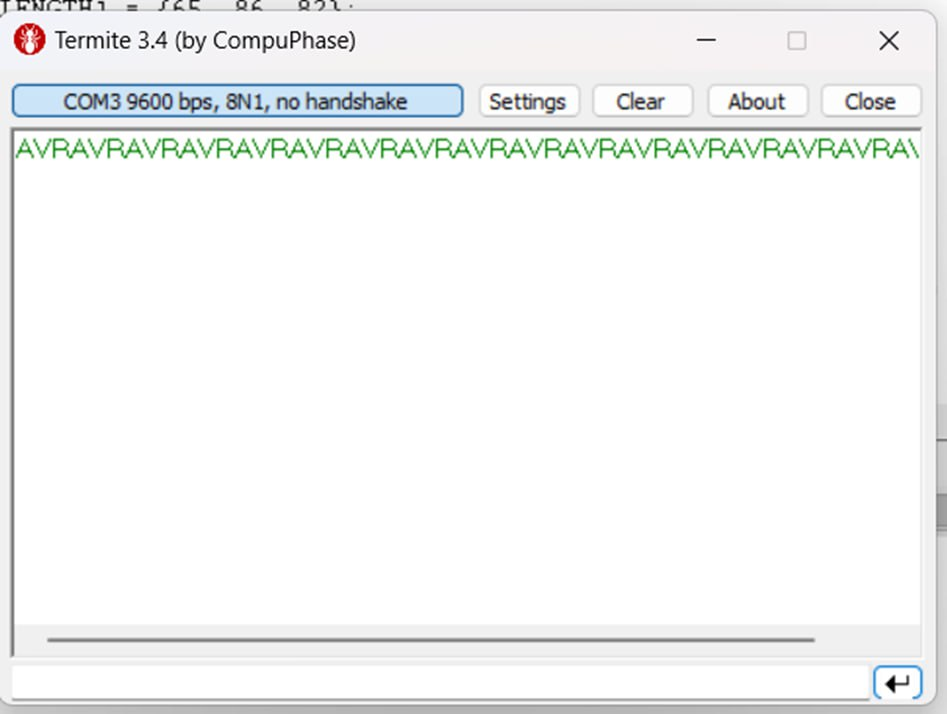


Рисунок 4 - скриншот терминала с получением

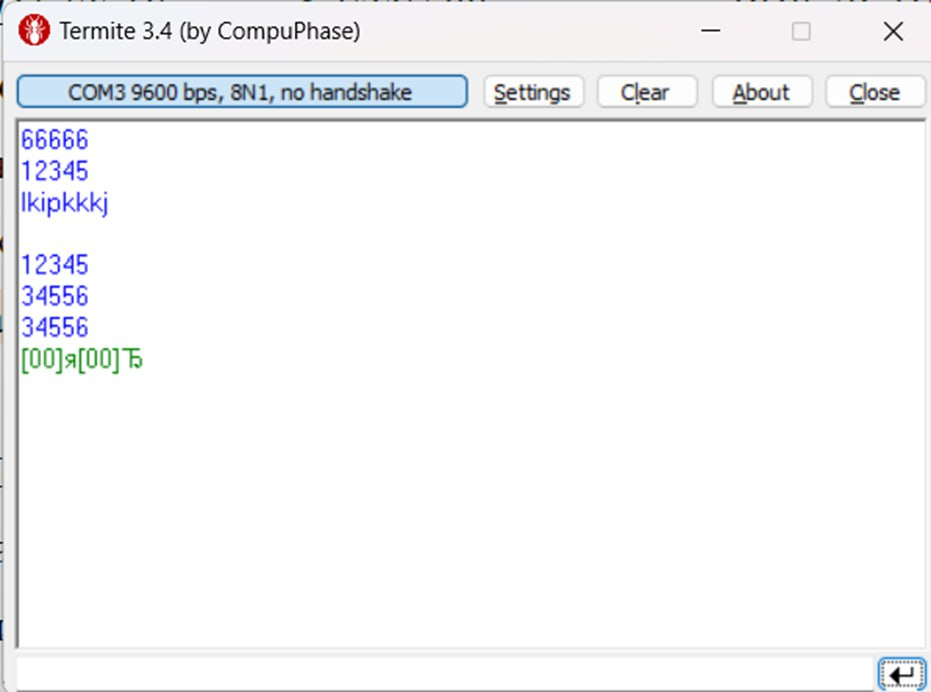


Рисунок 5 - скриншот терминала с отправкой сообщения

UART обеспечивает простой и надежный способ связи между МК и ПК через стандартный интерфейс. Это позволяет использовать ПК для управления или обмена данными с встраиваемой системой, требуя лишь согласования скорости и формат кадра, и использования программы-терминала.

**Задание 5. Передача данных между МК на платах STK500.**



Рисунок 6 - Фотография макета

Прямая UART-связь между двумя микроконтроллерами на отдельных платах (STK500) демонстрирует возможность создания распределенных систем или простого обмена данными между устройствами. Это требует только соединения TxD с RxD (кросс-соединение для дуплекса) и общего GND.

**Задание 6. Передача произвольного сообщения.**

Листинг 3 - код трансмиттера задание 6

|  |
| --- |
| #include <avr/io.h>  #include <avr/interrupt.h>  #include <stdbool.h>  #define BUTTON\_START 0  const unsigned int ubrrValue = 23;  #define DATA\_LENGTH 6  const unsigned char data[DATA\_LENGTH] = {86, 75, 32, 75, 65, 76};  uint8\_t counter = 0;  bool flag = false;  ISR(USART\_UDRE\_vect) {  if (flag)  UDR = data[counter++];  }  int main() {  UBRRH = (unsigned char)(ubrrValue>>8);  UBRRL = (unsigned char) ubrrValue;  UCSRB = (1<<TXEN) | (1<<UDRIE);  UCSRC = (1<<URSEL)| (2 << UCSZ0) | (1 << UPM1);  PORTB = (1<<BUTTON\_START);  sei();  while (1) {  if (!(PINB & (1<<BUTTON\_START))) {  while (!(PINB & (1 << BUTTON\_START)))  ;  flag = true;  while (counter < DATA\_LENGTH)  ;  counter = 0;  }  flag = false;  }  return 0;  } |

Листинг 3 - код ресивера задание 6

|  |
| --- |
| #include <avr/io.h>  #include <avr/interrupt.h>  #define BUTTON\_SHOW 0  const unsigned int ubrrValue = 23;  #define DATA\_LENGTH 6  unsigned char data[DATA\_LENGTH] = { 0 };  uint8\_t receivedBytes = 0;  ISR(USART\_RX\_vect) {  uint8\_t status = UCSRA;  if ( status & ((1<<FE)|(1<<DOR)|(1<<PE)) ) {  PORTC = 0x7F;  } else {  PORTC = 0xFF;  if (receivedBytes < DATA\_LENGTH) {  data[receivedBytes++] = UDR;  }  }  }  int main() {  UBRRH = (unsigned char)(ubrrValue>>8);  UBRRL = (unsigned char)ubrrValue;  UCSRB = (1<<RXEN) | (1<<RXCIE);  UCSRC = (1<<URSEL) | (2 << UCSZ0) | (1 << UPM1);  PORTB = (1<<BUTTON\_SHOW);  DDRC = 0xFF;  PORTC = 0xFF;  sei();  uint8\_t i = 0;  while (1) {  if (!(PINB & (1<<BUTTON\_SHOW))) {  while (!(PINB & (1<<BUTTON\_SHOW)))  ;  PORTC = ~data[i];  i = (i + 1) % DATA\_LENGTH;  }  }  return 0;  } |

### 

### 

Рисунок 7 - Временная дигарамма для задания 6

UART позволяет гибко настраивать формат кадра, включая уменьшенное число бит данных (7 бит) и использование контроля четности (Even/Odd) для обнаружения одиночных битовых ошибок. Приемник может аппаратно проверять эти условия и устанавливать флаги ошибок (PE, FE, DOR), что позволяет программе повысить надежность связи за счет обработки или индикации таких ситуаций.

**Выводы**

1) Модуль USART в AVR обеспечивает полнодуплексный приём/передачу кадров со служебными (старт, стоп, четность) и полезными битами.

2) Использование аппаратных флагов UDRE и прерываний по UDRIE позволяет организовать неблокирующую передачу.

3) Переход на 7‑битный формат и контроль ошибок (FE, DOR, PE) делает связь более надёжной.

4) Виртуальный терминал и симуляция в Proteus помогают быстро отладить передачу и приём, а вывод байтов на светодиоды — визуально подтвердить корректность.