Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования «Рязанский государственный радиотехнический университет имени В.Ф. Уткина»

Кафедра «Электронные вычислительные машины»

ОТЧЁТ

По лабораторная работа №6

**Асинхронный приёмо-передатчик**

По дисциплине «ЭВМ и периферийные устройства»

Выполнили:

Проверили:

Рязань 20

**Цель работы:** изучить принцип работы последовательного интерфейса UART, изучить порядок настройки и работы с интерфейсом на микроконтроллере Atmega328P.

**Пример 1.** Настройка интерфейса и передача одного байта данных. Пример отправки байта информации при нажатии на кнопку. Результат выполнения - рисунок 1.

**Код программы примера 1**

#include <avr/io.h> //библиотеки ввода/вывода

#include <avr/interrupt.h> //библиотека обработки прерываний

ISR (INT0\_vect) //обработчик прерывания INT0

{

UDR0 = 0x3E; //отправка символа ">" по UART

}

int main(void) //точка входа в программу

{

EICRA |= 0x03; //прерывание INT0 по фронту входящего сигнала

EIMSK

|=

0x01;

//разрешение прерывания INT0

DDRD

&=

~0x04;

//PORTD2 - на ввод информации

PORTD |= 0x04; //PORTD2 (INT0) - режим PULL UP

UBRR0 = 103; //скорость 9600бит/c при частоте генератора 16МГц

UCSR0B |= (1<<TXEN0); //разрешение отправки данных по UART

UCSR0C |= (1<<UCSZ01)|(1<<UCSZ00);//размер пакета 8бит

sei(); //глобальное разрешение прерываний

while (1) //бесконечный цикл (без действий)

{}

}

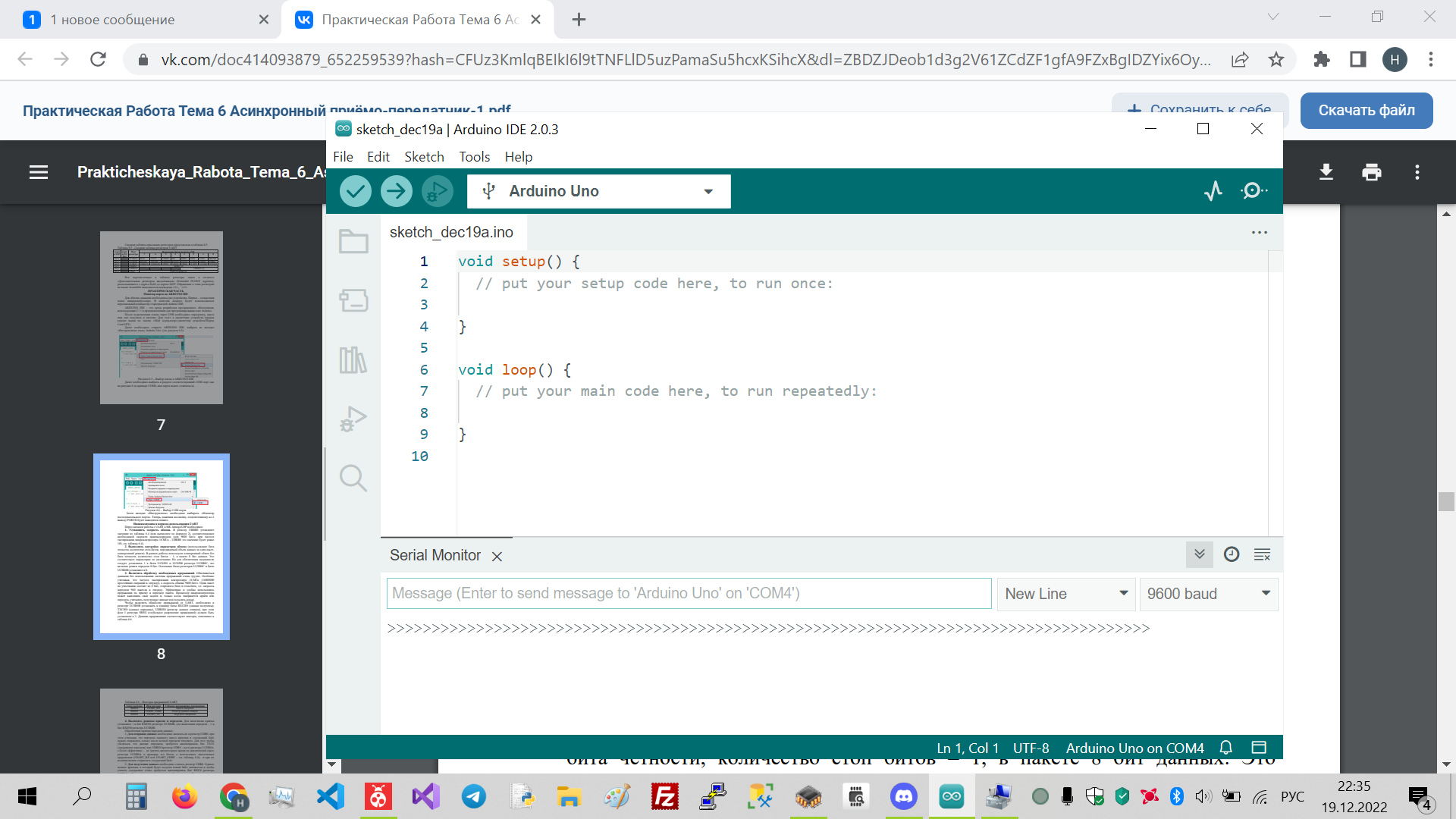


Рисунок 1 — выполнение прмера 1

**(Вариант 9)**

**Задание 1:**

Анализ передачи в логическом анализаторе. Измените передаваемый символ в соответствии с вариантом. Выполните сборку программы. Подключите логический анализатор к выводу PORTD0. Загрузите код в контроллер. Откройте программу Logic. Выполните анализ сигнала с линии TX. Докажите, что сигнал на линии соответствует символу, передаваемому по варианту задания. Результат выполнения - рисунок 2.

Символ i.

**Код программы задания 1**

#include <avr/io.h> //библиотеки ввода/вывода

#include <avr/interrupt.h> //библиотека обработки прерываний

ISR (INT0\_vect) //обработчик прерывания INT0

{

UDR0 = 0x69; //отправка символа "i" по UART

}

int main(void) //точка входа в программу

{

EICRA |= 0x03; //прерывание INT0 по фронту входящего сигнала

EIMSK

|=

0x01;

//разрешение прерывания INT0

DDRD

&=

~0x04;

//PORTD2 - на ввод информации

PORTD |= 0x04; //PORTD2 (INT0) - режим PULL UP

UBRR0 = 103; //скорость 9600бит/c при частоте генератора 16МГц

UCSR0B |= (1<<TXEN0); //разрешение отправки данных по UART

UCSR0C |= (1<<UCSZ01)|(1<<UCSZ00);//размер пакета 8бит

sei(); //глобальное разрешение прерываний

while (1) //бесконечный цикл (без действий)

{}

}

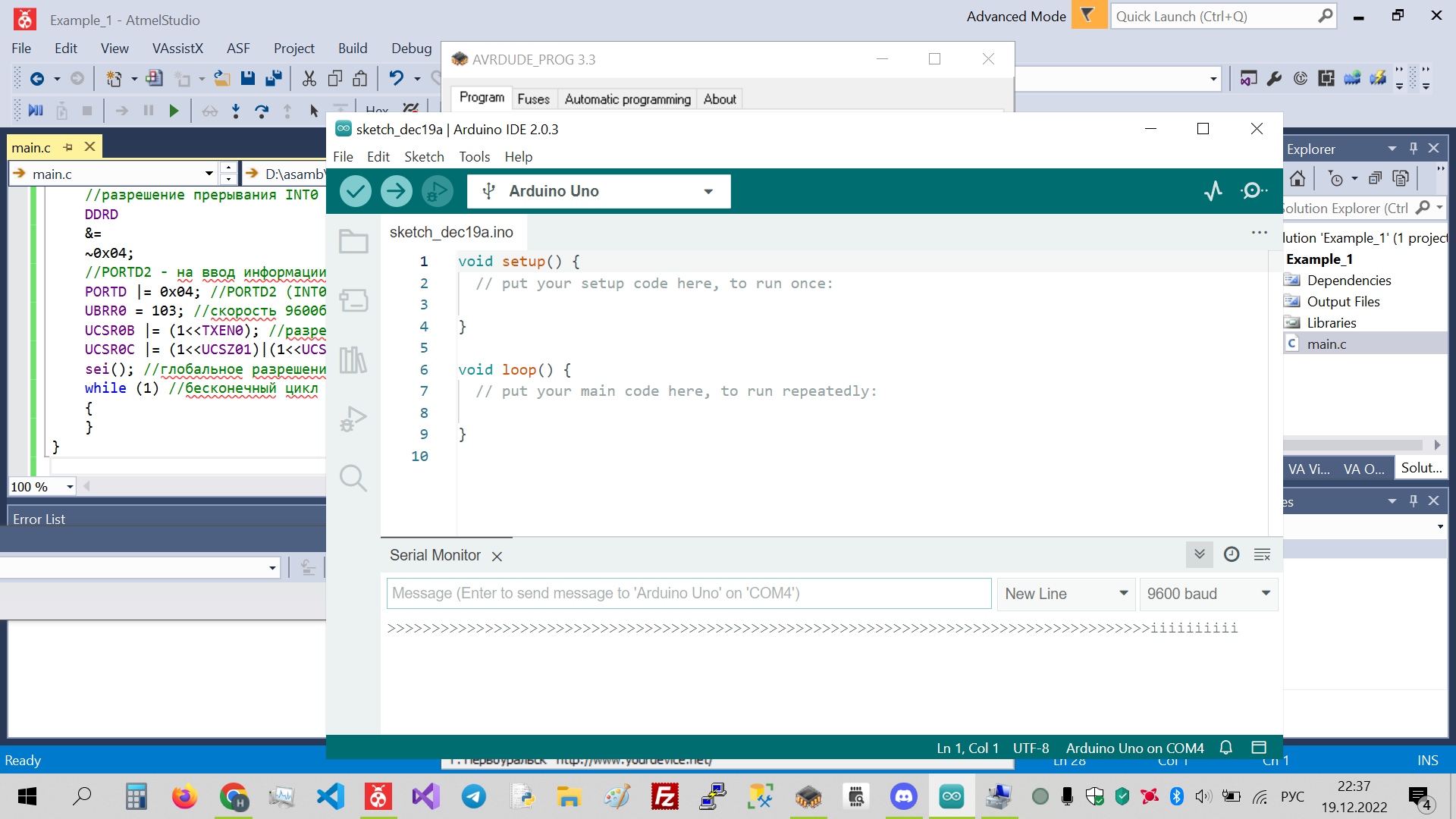


Рисунок 2 — выполнение задания 1

**Пример 2.** Передача сообщений. В рассмотренных задачах выполняется передача по одному байту. Каждая передача выполняется по нажатию на клавишу. Пример передачи сообщения, записанного в ПЗУ микроконтроллера на языке Assembler. Программа постоянно выводит через UART одно и тоже сообщение. Результат выполнения - рисунок 3.

**Код программы примера 2**

.include "m328Pdef.inc"

.equ CLK=16000000

.equ BAUD=9600

.equ UBRR0\_value = (CLK/(BAUD\*16)) - 1

.org 0

jmp Reset

Reset:

;настройка UART

;установка частоты приёма/передачи

ldi r16, high(UBRR0\_value)

sts UBRR0H, r16

ldi r16, low(UBRR0\_value)

sts UBRR0L, r16

;разрешение передачи

ldi r16,(1<<TXEN0)

sts UCSR0B,R16

;длина слова 8 бит

ldi r16,(1<< UCSZ00)|(1<< UCSZ01)

sts UCSR0C,R16

;разрешение всех прерываний

sei

main:

;определение адреса сообщения в ПЗУ

ldi zl,low(Msg<<1)

ldi zh,high(Msg<<1)

;количество байт в сообщении

ldi r18,0x17

send: ;отправка сообщения

lpm r16, Z+

sts UDR0,r16

subi r18,0x01

repeat: ;проверка окончания сообщения

lds r17, UCSR0A

bst r17, 5

brtc repeat

cpi r18,0x00

breq main

jmp send

.cseg ;размещение сообщения в ПЗУ

Msg: .db "Atmega328P:ReadyToWork",'\n'

**Задание 2.** Разработайте аналогичную программу на языке C. Однако вместо циклического анализа бита UDRE0 сделайте передачу с использованием прерывания, возникающего при событии завершения передачи. Проанализируйте работу программы. Результат выполнения - рисунок 3.

**Код программы задания 2**

#include <avr/io.h> //библиотеки ввода/вывода

#include <avr/interrupt.h> //библиотека обработки прерываний

int i = 0;

char Msg[23]={'A', 't', 'm', 'e', 'g', 'a', '3', '2', '8', 'P', ':', 'R', 'e', 'a', 'd', 'y', 'T', 'o', 'W', 'o', 'r', 'k', '\n'};//сообщение

ISR (USART\_TX\_vect) //обработчик прерывания USART\_TX

{

if (i <= 22)//номер символа в массиве

{

UDR0 = Msg[i];//выводим i-ый символ

i++;//счётчик

}

else

{

i = 0;//начинаем вывод с начала

UDR0 = Msg[i];//выводим i-ый символ

}

}

int main (void){

UBRR0 = 103;//скорость 9600бит/c при частоте генератора 16МГц

UCSR0B = (1<<TXCIE0)|(1<<TXEN0);//разрешён прием данных, если разрешена отправка

UCSR0C = (1<<UCSZ01)|(1<<UCSZ00);//размер пакета 8бит

sei();

UDR0 = 0x69;//передача, чтобы было чего завершать

while(1){

}

}

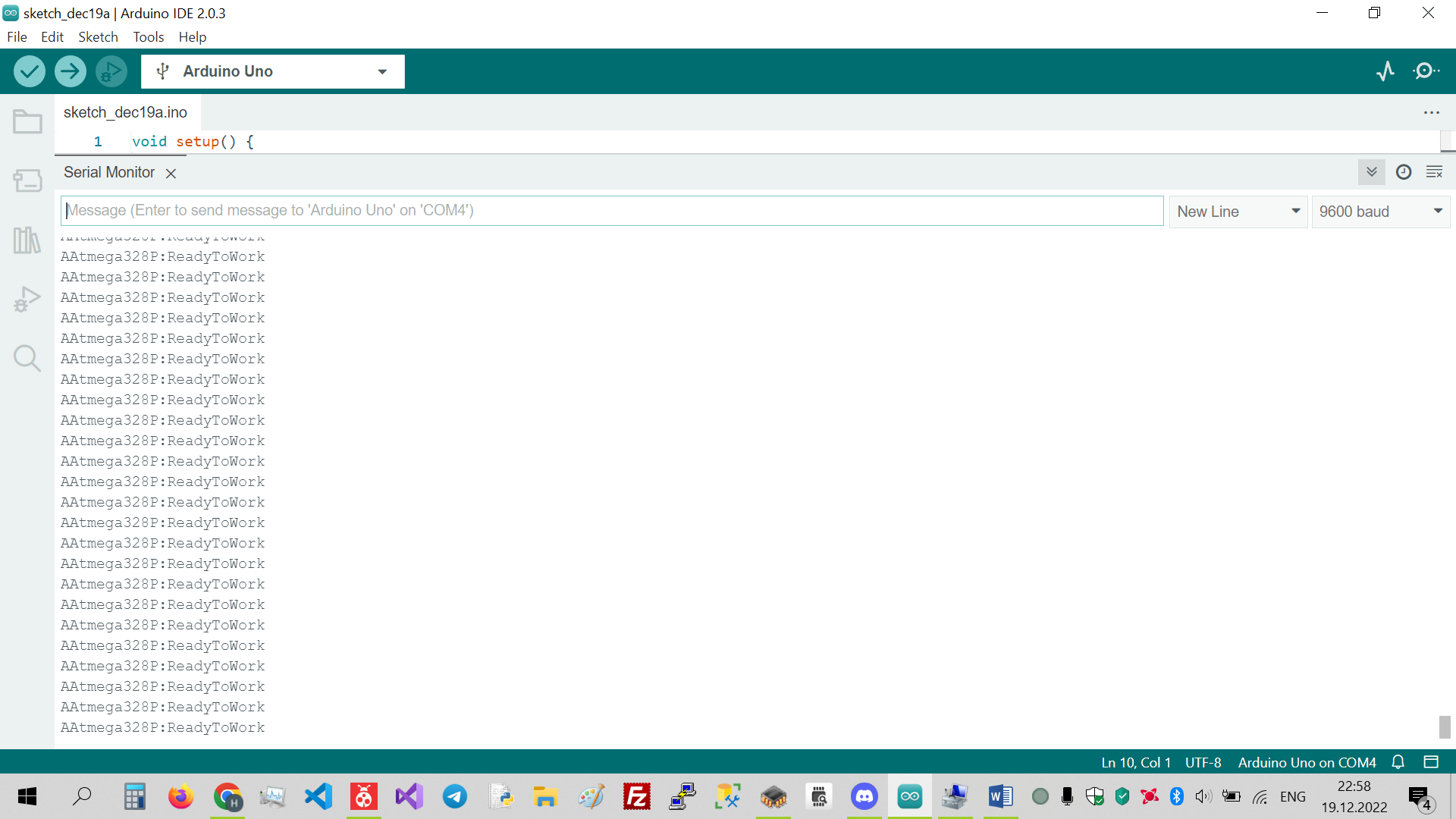


Рисунок 3 — выполнение примера 1 и задания 2

**Задание 3.** Приём информации: исправьте код примера таким образом,

чтобы плата реагировала на принятый байт в соответствии с вариантом следующим образом: если получен символ согласно номеру варианта, светодиод L (PortB5) должен зажечься, в противоположном случае – погаснуть. Обработку информации из регистра UDR необходимо вести по прерыванию, иначе, например при считывании значения, в основной программе в цикле может быть получена некорректная информация.

Символ 8.

**Код программы задания 3**

#include <avr/io.h> //библиотеки ввода/вывода

#include <avr/interrupt.h> //библиотека обработки прерываний

int ch;//символ, полученный с компа

ISR (USART\_RX\_vect) //по завершению приёма -- чтение

{

ch = UDR0;

if (ch == '8'){

PORTB = 0x20;//светим

}

else{

PORTB = 0x00;//не светим

}

}

int main (void){

DDRB = 0x20;//выводим светодиод

UBRR0 = 103;//скорость 9600бит/c при частоте генератора 16МГц

UCSR0B = (1<<RXEN0)|(1<<RXCIE0);//разрешено прерывание, если разрешен вывод

UCSR0C = (1<<UCSZ01)|(1<<UCSZ00);//размер пакета 8бит

sei();

while(1){

}

}

Программа работает исправно, при вводе сообщения «8g8g8g8g8g8g» светодиод мигает.

**Задание 4.** Обмен пакетами данных.

Разработайте программу, которая получает 4 числа в формате !a,b,c,d. (где a,b,c и d – целые числа, «!» – начало обмена, «.» – завершение обмена), и в порядке b,a,d,c передаёт обратно. Для передачи и приёма сообщений используйте соответствующие прерывания. Для разработки программы используйте язык C. Результат выполнения - рисунок 4.

**Код программы задания 4**

#include <avr/io.h> //библиотеки ввода/вывода

#include <avr/interrupt.h> //библиотека обработки прерываний

int i = 0; //номер числа (числа от 0 до 2^3 включительно)

int zn, fl;//число и флаг

char Mess[5];//массив данных (5 вместо 4, так как иначе он считает что последний элемент равен \n)

ISR (USART\_RX\_vect){//по завершению приёма -- чтение

zn = UDR0; //чтение числа

if (zn == '!') {

fl = 1;

i = 1;

}

else if ((fl == 1) && (zn != '.') && (i == 1) && (zn != ',')){ // запись i числа в i ячейку массива

Mess[i] = zn;

i++;

}

else if ((fl == 1) && (zn != '.') && (i == 2)){

Mess[i] = zn;

i++;

}

else if ((fl == 1) && (zn != '.') && (i == 3)){

Mess[i] = zn;

i++;

}

else if ((fl == 1) && (zn != '.') && (i == 4)){

Mess[i] = zn;

i=0;

}

else if (zn == '.'){ //конец записи

UDR0 = Mess[2]; //вывод 2 числа

i=0;

}

}

ISR (USART\_TX\_vect){ //по завершению передачи -- вывод

if (i==0){

UDR0 = Mess[1];//вывод 1 числа

i++;

}

else if (i==1){//вывод 4 числа

UDR0 = Mess[4];

i++;

}

else if (i==2){//вывод 3 числа

UDR0 = Mess[3];

i++;

}

}

int main(void){

UBRR0 = 103;

UCSR0B = (1<<TXCIE0)|(1<<TXEN0)|(1<<RXCIE0)|(1<<RXEN0);//разрешение прерываний и на выход и на вход

UCSR0C = (1<<UCSZ01)|(1<<UCSZ00);//размер пакета 8бит

sei();//глобальное разрешение прерываний

while(1)//бесконечный цикл

{

}

}

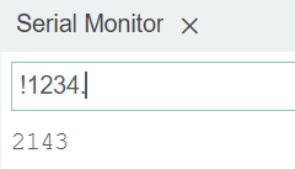


Рисунок 4 — выполнение задания 4

**Вывод:** изучили принцип работы последовательного интерфейса UART, изучили порядок настройки и работы с интерфейсом на микроконтроллере Atmega328P.