МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИМЕНИ И.С. ТУРГЕНЕВА»

Кафедра программной инженерии

**ОТЧЕТ**

по лабораторным работам № 7-8

на тему: «Разработка программы на языке Ассемблера, работающей с приемо-передатчиком (USART)»

по дисциплине: «Программирование микроконтроллеров»

Вариант № 18

Выполнил: Шорин В.Д. Шифр: 171406

Институт приборостроения, автоматизации и информационных технологий

Направление: 09.03.04 «Программная инженерия»

Группа: 71-ПГ

Проверили: Захарова О.В., Раков В.И.

Отметка о зачете:

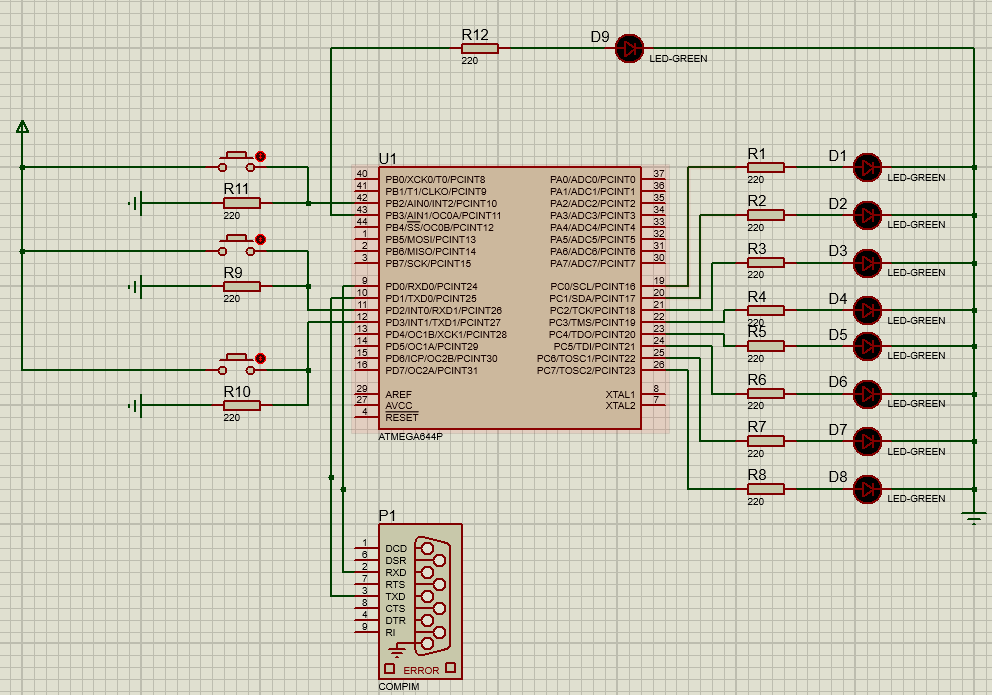
Дата: «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2020 г.

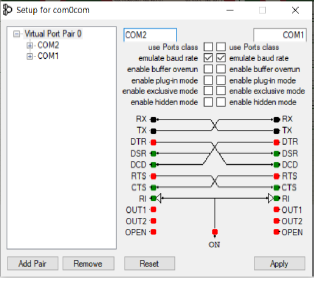
Орел, 2020 г

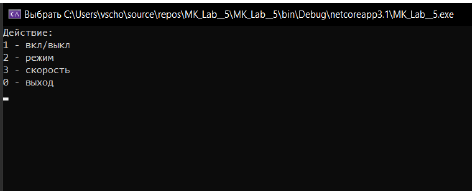
**Задание на лабораторную работу**

Разработать программу для микроконтроллера на языке Ассемблера для управления гирляндой (из лабораторной работы 6) с терминала.

**Ход работы**

****

****

****

**Код**

**Код микроконтроллера**

//.include "m644pdef.inc"

.def temp = r16

.def mainBtn = r17

.def modeOperating = r18

.def modeSpeed = r19

.def up = r20

.def i\_LED = r21

.def temp2 = r22

.equ fCK = 8000000 ;частота в герцах

.equ BAUD = 76800 ;скорость для UART в бодах

.equ Bitrate = fCK/(BAUD\*16)-1 ;расчитываем значение для регистра UBRR

.cseg

.org 0x0000 jmp RESET ; Reset

.org 0x0002 jmp INT0\_vect ; IRQ0

.org 0x0004 jmp INT1\_vect ; IRQ1

.org 0x0006 jmp INT2\_vect ; IRQ2

.org 0x0020 jmp TIM0\_COMPA\_vect ; Timer0 CompareA

.org 0x0024 jmp TIM0\_OVF\_vect ; Timer0 Overflow

.org 0x0028 jmp USART0\_RX\_vect ; USART0 RX Complete

RESET:

ldi temp, high(RAMEND) //настройка стека

out sph, temp

ldi temp, low(RAMEND)

out spl, temp

ldi temp, 0xff

out DDRC, temp

ldi mainBtn, 0

ldi modeOperating, 0

ldi modeSpeed, 0

ldi up, 1

ldi i\_LED, 0

// Настройка прерываний INT0, INT1, INT2 (передний фронт)

ldi temp, 0b00111111

sts EICRA, temp

// разрешение прерываний INT0, INT1, INT2

ldi temp, 0b00000111

out EIMSK, temp

ldi temp, high(Bitrate) ;устанавливаем скорость 9600 бод

sts UBRR0H, temp

ldi temp, low(Bitrate)

sts UBRR0L, temp

ldi temp, 0b10010000

sts UCSR0B, temp

SEI

START:

jmp START

USART0\_RX\_vect:

CLI

lds temp, UCSR0A

sbrs temp, RXC0

rjmp USART0\_RX\_vect

lds temp, UDR0

CPI temp, 0x31

BREQ INT2\_vect

CPI temp, 0x32

BREQ INT0\_vect

CPI temp, 0x33

BREQ INT1\_vect

SEI

RETI

INT0\_vect:

CLI

CPI modeOperating, 2

BREQ ModeOperating0

CPI modeOperating, 2

BRNE ModeOperatingInc

ModeOperating0:

ldi modeOperating, 0

RCALL TimersSettings

SEI

RETI

ModeOperatingInc:

inc modeOperating

RCALL TimersSettings

SEI

RETI

INT1\_vect:

CLI

CPI up, 1

BREQ UpSpeed

CPI up, 0

BREQ LowSpeed

UpSpeed:

INC modeSpeed

CPI modeSpeed, 2

BREQ Up0

CPI modeSpeed, 2

BRNE Up1

Up0:

ldi up, 0

SEI

RCALL ChangeSpeedMode

RETI

Up1:

ldi up, 1

SEI

RCALL ChangeSpeedMode

RETI

LowSpeed:

DEC modeSpeed

CPI modeSpeed, 0

BREQ Up1

CPI modeSpeed, 0

BRNE Up0

SEI

RCALL ChangeSpeedMode

RETI

INT2\_vect:

CLI

CPI mainBtn, 0

BREQ MainAssign1

CPI mainBtn, 1

BREQ MainAssign0

MainAssign1:

ldi mainBtn, 1

CALL TimersSettings

SEI

RETI

MainAssign0:

ldi mainBtn, 0

CALL TimersSettings

SEI

RETI

TIM0\_OVF\_vect:

CLI

IN temp, PORTC

COM temp

OUT PORTC, temp

IN temp, 0

out TCNT0, temp

SEI

RETI

TIM0\_COMPA\_vect:

CLI

IN temp, PORTC

COM temp

OUT PORTC, temp

IN temp, 0

out TCNT0, temp

SEI

RETI

TimersSettings:

CPI modeOperating, 0

BREQ Mode0

CPI modeOperating, 1

BREQ Mode1

CPI modeOperating, 2

BREQ Mode2

Mode0:

ldi temp, 0x00

out DDRB, temp

ldi temp, 0xFF

out DDRC, temp

ldi temp, 0x00

out TCCR0B, temp

ldi temp, 1 << TOIE0

sts TIMSK0, temp

ldi temp, 0b10101010

out PORTC, temp

RCALL ChangeSpeedMode

CPI mainBtn, 0

BREQ MainBtn0

RETI

Mode1:

ldi temp, 0x00

out DDRB, temp

ldi temp, 0xFF

out DDRC, temp

ldi temp, 0x00

out TCCR0B, temp

ldi temp, 1 << OCIE0A

sts TIMSK0, temp

ldi temp, 200

out OCR0A, temp

ldi temp, 0b11110000

out PORTC, temp

RCALL ChangeSpeedMode

CPI mainBtn, 0

BREQ MainBtn0

RETI

Mode2:

ldi temp, 0b11000000

out PORTC, temp

ldi temp, 0x00

out DDRC, temp

ldi temp, 1 << 3

out DDRB, temp

ldi temp, 200

out OCR0A, temp

ldi temp, 1 << COM0A0

out TCCR0A, temp

RCALL ChangeSpeedMode

CPI mainBtn, 0

BREQ MainBtn0

RETI

MainBtn0:

ldi temp, 0x00

out TCCR0B, temp

ldi temp, 0x00

out PORTC, temp

ldi temp, 0x00

out PORTB, temp

RETI

ChangeSpeedMode:

ldi temp, 0x00

out TCCR0B, temp

CPI modespeed, 0

BREQ ModeSpeed0

CPI modespeed, 1

BREQ ModeSpeed1

CPI modespeed, 2

BREQ ModeSpeed2

ModeSpeed0:

ldi temp, 0x05

out TCCR0B, temp

RETI

ModeSpeed1:

ldi temp, 0x04

out TCCR0B, temp

RETI

ModeSpeed2:

ldi temp, 0x03

out TCCR0B, temp

RETI

**Код терминала**

using System;

using System.IO.Ports;

namespace MK\_Lab\_7\_8

{

class Program

{

static void Main(string[] args)

{

SerialPort \_serialPort;

try

{

// настройки порта

\_serialPort = new SerialPort("COM1", 9600, Parity.None, 8, StopBits.One);

\_serialPort.Handshake = Handshake.None;

\_serialPort.WriteTimeout = 1000;

\_serialPort.Open();

}

catch (Exception e)

{

Console.WriteLine("ERROR: невозможно открыть порт:" + e.ToString());

return;

}

bool flag = true;

while (flag)

{

Console.WriteLine("Действие:");

Console.WriteLine("1 - вкл/выкл");

Console.WriteLine("2 - режим");

Console.WriteLine("3 - скорость");

Console.WriteLine("0 - выход");

string operand = Console.ReadLine();

switch (operand)

{

case "1":

{

try

{

if (!(\_serialPort.IsOpen))

\_serialPort.Open();

\_serialPort.Write("1");

}

catch (Exception ex)

{

Console.WriteLine("Error opening/writing to serial port :: "

+ ex.Message, "Error!");

}

break;

}

case "2":

{

try

{

if (!(\_serialPort.IsOpen))

\_serialPort.Open();

\_serialPort.Write("2");

}

catch (Exception ex)

{

Console.WriteLine("Error opening/writing to serial port :: "

+ ex.Message, "Error!");

}

break;

}

case "3":

{

try

{

if (!(\_serialPort.IsOpen))

\_serialPort.Open();

\_serialPort.Write("3");

}

catch (Exception ex)

{

Console.WriteLine("Error opening/writing to serial port :: "

+ ex.Message, "Error!");

}

break;

}

case "0":

{

flag = false;

return;

}

}

}

\_serialPort.Close();

}

}

}