МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»

(БГТУ им. В.Г.Шухова)

**Лабораторная работа №2**

дисциплина «ЭВМ и периферийные устройства»

тема: «изучение принципов организации обмена данными по последовательному интерфейсу usb между микроконтроллером msp430f1611 и пэвм»

Выполнил: студент группы ВТ-31

Ковалёв И. Д.

Проверил: Шамраев А.А.

Белгород 2020

**Цель работы:** Изучить возможности сопряжения лабораторного стенда на базе микроконтроллера MSP430F1611 и ПЭВМ с помощью последовательного интерфейса USB, принципы программного управления двунаправленным обменом данных по последовательному интерфейсу USB.

**Задания к работе:** Разработать в среде программирования IAR Embedded Workbench программу на языке С для связи микроконтроллера MSP430F1611 с ПЭВМ по интерфейсу USB в соответствие с параметрами протокола обмена, приведенными в таблице 4.2.8.

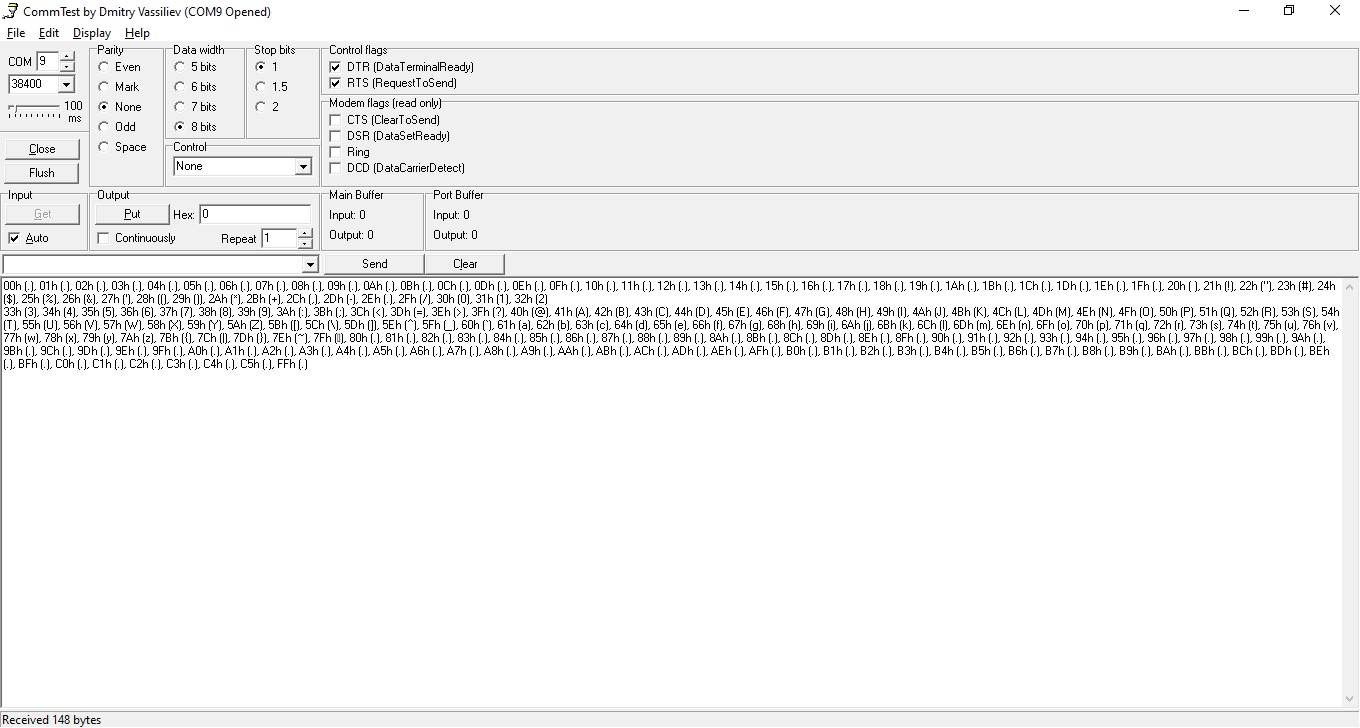
**Порядок выполнения задания:**

1. Включить лабораторный макет.
2. Запустить компилятор iar embedded workbench.
3. Создать пустой проект.
4. Создать файл ресурса для кода программы и подключить его к проекту.
5. Ввести код исходного модуля программы обмена данными между микроконтроллером msp430f1611 с пэвм по интерфейсу usb в соответствие с индивидуальным заданием, приведенным в таблице 4.2.8.
6. Выполнить компиляцию исходного модуля программы и устранить ошибки, полученные на данном этапе.
7. Настроить параметры программатора.
8. Проверить правильность подключения интерфейсного usb кабеля к разъемам лабораторного макета и пэвм.
9. Запустить на пэвм программу terminal, установить необходимые параметры протокола обмена данными, выбрать номер последовательного порта (сомx), соответствующего виртуальному com-порту, и нажать на кнопку connect в верхнем левом углу рабочего окна программы. 111
10. Создать загрузочный модуль программы и выполнить программирование микроконтроллера.
11. Проверить работоспособность загруженной в микроконтроллер программы и показать результаты работы преподавателю. В случае некорректной работы разработанной программы, выполнить аппаратный сброс микроконтроллера, провести отладку исходного модуля программы и заново проверить функционирование программы.

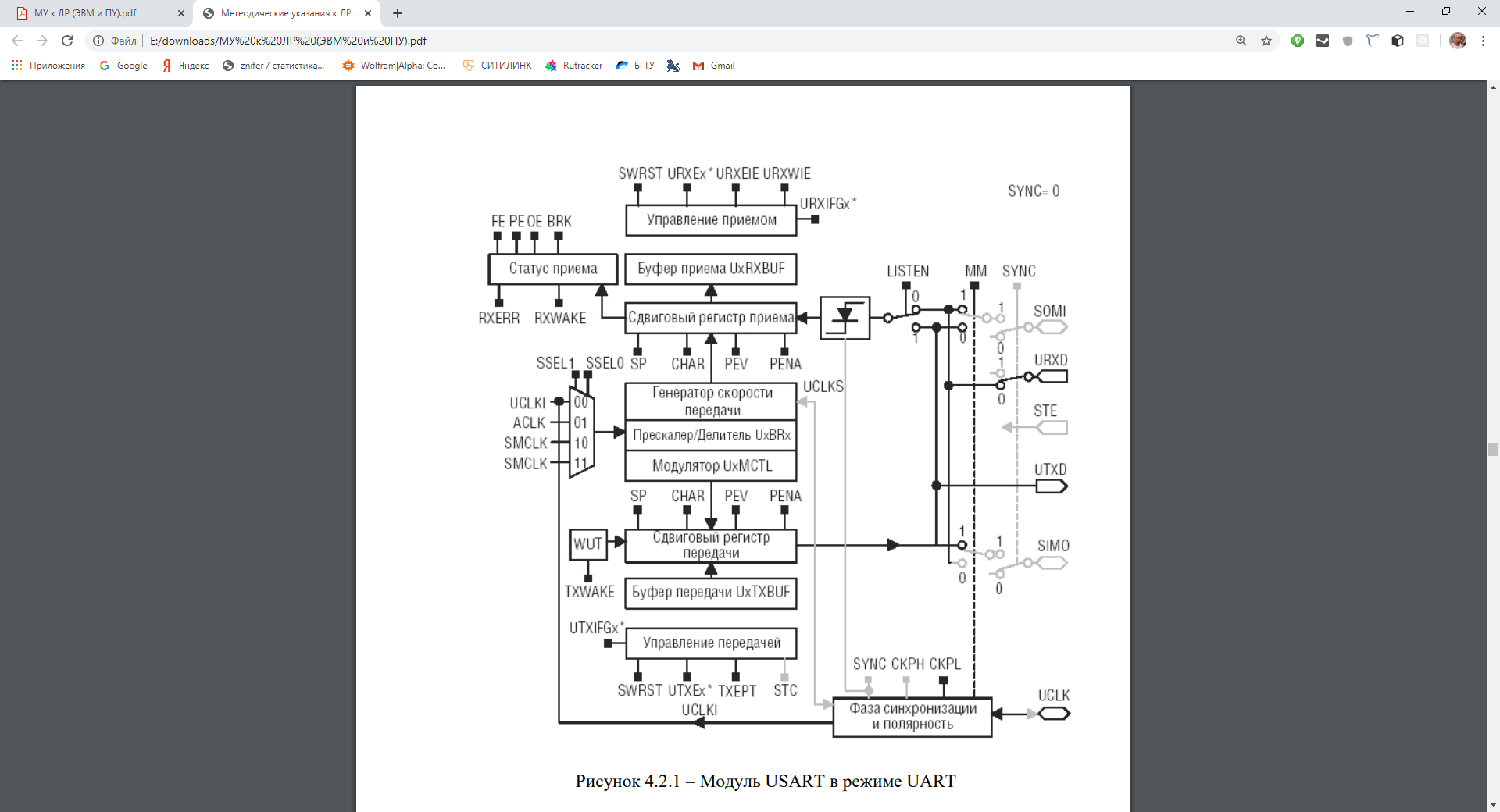
**Задание варианта №7:**

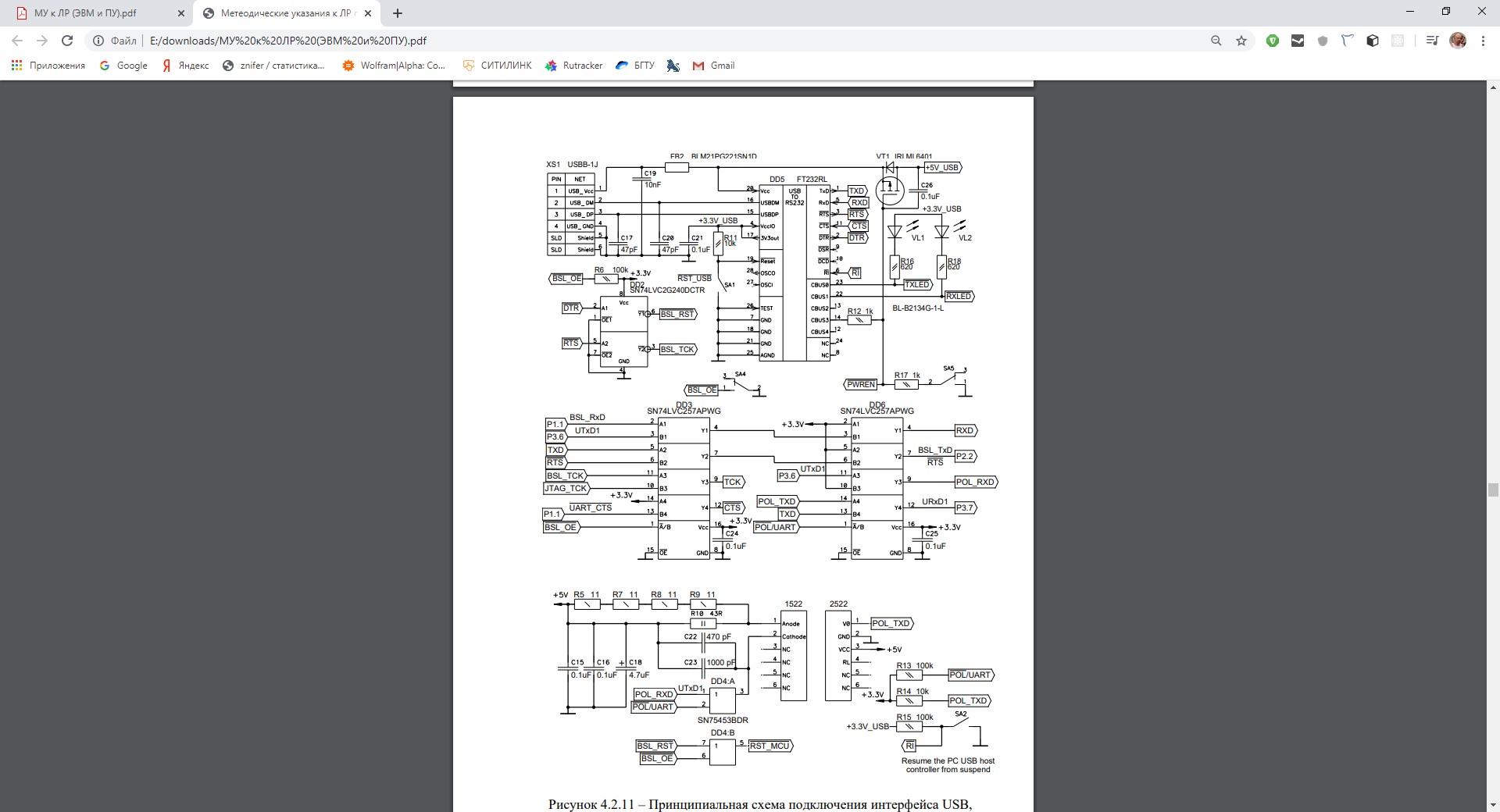
Разработать программу передачи 200 чисел (от 0 до 199) из микроконтроллера MSP430F1611 в ПЭВМ по интерфейсу USB в соответствие с протоколом: модуль USART0, скорость обмена данными 38400 бит/с, режим обмена асинхронный, 8 битов данных без бита четности.

**Результаты работы программы:**

,

**Вывод:** в процессе выполнения лабораторной работы были изучены возможности сопряжения лабораторного стенда на базе микроконтроллера MSP430F1611 и ПЭВМ с помощью последовательного интерфейса USB, принципы программного управления двунаправленным обменом данных по последовательному интерфейсу USB.





**Содержимое файла main.c:**

#include <msp430.h>

#include "stdio.h"

#include "system\_define.h"

#include "system\_variable.h"

#include "function\_prototype.h"

#include "main.h"

#define SIZE 200

/\*–азработать программу передачи 200 чисел (от 0 до 199) из

микроконтроллера MSP430F1611 в ѕЁ¬ћ по интерфейсу USB в

соответствие с протоколом:

модуль USART0,

скорость обмена данными 38400 бит/с,

режим обмена асинхронный,

8 битов данных без бита четности.\*/

void main(void) {

//ќтключение ждущего таймера

WDTCTL = WDTPW + WDTHOLD;

//»нициализируем систему

Init\_System();

Init\_System\_Clock();

//»нициализируем UART

char speed = 2; // 0 - 38400, 1 - 57600, 2 - 115200 - скорость обмена

char databits = 8; // 7, 8 - длинна символа

char stopbits = 1; // 1, 2 - кол-во передаваемых стоповых бит

char parity = 0; // 0 - без контрол¤ четности, 1 - контроль четности, нечетый, 2 - четный

char iface = 0; // 0 - USB, 1 - оптика

//ѕередаем данные

char i;

UART\_init(speed, databits, stopbits, parity, iface);

while (1){

for (i = 0; i < SIZE; i++){

UART\_sendbyte(i);

}

wait\_1ms(1000);

}

UART\_off();

}

**Содержимое файла uart.c:**

// UART functions

#include "function\_prototype.h"

#include "sysfunc.h"

#include "uart.h"

// »нициализаци¤ режима UART

// speed = 0 - 38400, 1 - 57600, 2 - 115200, 3 - 19200 - скорость обмена

// databits = 7, 8 - длинна символа

// stopbits = 1, 2 - кол-во передаваемых стоповых бит

// parity = 0 - без контрол¤ четности, 1 - контроль четности, нечетый, 2 - четный

// iface = 0 - USB, 1 - оптика

void UART\_init(byte speed, byte databits, byte stopbits, byte parity, byte iface)

{

P3SEL |= BIT6 | BIT7; // выбор функции USART1

U1CTL = 0; // инициализаци¤ состо¤ни¤ USART

ME2 |= UTXE1 + URXE1; // включить приемник и передатчик USART1

if (databits == 7) U1CTL &= ~CHAR; // 7-разр¤дна¤ длинна символа

if (databits == 8) U1CTL |= CHAR; // 8-разр¤дна¤ длинна символа

if (stopbits == 1) U1CTL &= ~SPB; // 1 стоповый бит

if (stopbits == 2) U1CTL |= SPB; // 1 стоповых бита

if (parity == 0) U1CTL &= ~PENA; // контроль четности отключен

if (parity == 1) U1CTL = (U1CTL & ~PEV) | PENA; // контроль четности, нечетный

if (parity == 2) U1CTL |= PENA | PEV; // контроль четности, четный

P5DIR |= BIT0; // переключение мультиплексора на USB/оптику

if (iface == 0)

P5OUT |= BIT0;

if (iface == 1)

P5OUT &= ~BIT0;

U1TCTL |= SSEL1; // BRCLK = SMCLK

U1BR0 = 69; // 8Mhz / 115200 = 69.44 (по-умолчанию)

if (speed == 0) U1BR0 = 208; // 8ћ√ц / 38400 = 208.33

if (speed == 1) U1BR0 = 139; // 8ћ√ц / 57600 = 138,89

U1BR1 = 0x00;

U1MCTL = 0x2C;

if (speed == 2) { // 8ћ√ц / 9600 = 833 = 341h

U1BR0 = 0x41;

U1BR1 = 0x03;

U1MCTL = 0x00;

}

// модул¤ци¤

}

// отключение режима UART

void UART\_off()

{

P3SEL |= BIT6 | BIT7; // выбор функции USART1

ME2 &= ~(UTXE1 + URXE1); // выключить приемник и передатчик USART1

U1CTL = SWRST; // отключение USART1

}

// вывод строки символов (символ с кодом 0 - конец строки)

void UART\_message(char \* buf)

{

word i=0;

while (buf[i])

UART\_sendbyte(buf[i++]); // передача сивола

}

// передача байта

void UART\_sendbyte(char byte)

{

while (!(IFG2 & UTXIFG1)); // проверка готовности буфера передачи USART1

U1TXBUF = byte; // передача байта

}

// получение байта

char UART\_getbyte()

{

while (!(IFG2 & URXIFG1)); // проверка готовности буфера приема USART1

return U1RXBUF; // возврат полученного байта

}