Министерство образования Республики Беларусь

Учреждение образования Белорусский государственный университет информатики и радиоэлектроники

Кафедра ЭВМ

Отчёт по лабораторной работе №1

“Последовательный порт”

Проверил: Выполнил:

к.т.н., доцент студент гр., 250502

Одинец Дмитрий Николаевич Бекетова М. А.

Минск 2024

**Задача**

Целью данной работы является разработка программного модуля реализации процедуры передачи (приема) байта информации через последовательный интерфейс.

Программа должна демонстрировать программное взаимодействие с последовательным интерфейсом с использованием следующих механизмов:

1. Прямое взаимодействие с портами ввода-вывода (write-read)
2. Использование BIOS прерываний 14h
3. Работа с COM-портом через регистры как с устройством ввода-вывода

**Алгоритм**

Программа состоит из нескольких подпрограмм (частей программы), представляющих собой некоторые функции. К ним относятся функции:

* Функция инициализации порта
* Функция передачи данных в порт
* Функция чтения данных из порта
* Анализ состояния порта

**Листинг программы**

1. **Программирование СОМ-порта с помощью API функций Windows.**

#include <iostream>

#include <windows.h>

HANDLE Port1, Port2;

void initializeSerialPorts(LPCWSTR PortName1, LPCWSTR PortName2){

Port1 = CreateFile(PortName1, GENERIC\_WRITE, 0, NULL, OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, NULL);

Port2 = CreateFile(PortName2, GENERIC\_READ, 0, NULL, OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, NULL);

if (Port1 == INVALID\_HANDLE\_VALUE || Port2 == INVALID\_HANDLE\_VALUE){

std::cout << "Error in opening serial port" << std::endl;

return;

}

// Настройка параметров

DCB dcbSerialParams = { 0 };

dcbSerialParams.DCBlength = sizeof(dcbSerialParams);

if (!GetCommState(Port1, &dcbSerialParams)){

std::cout << "Failed to get current serial parameters for ports" << std::endl;

}

dcbSerialParams.BaudRate = CBR\_9600;

dcbSerialParams.ByteSize = 8;

dcbSerialParams.StopBits = ONESTOPBIT;

dcbSerialParams.Parity = NOPARITY;

if (!SetCommState(Port1, &dcbSerialParams) || !SetCommState(Port2, &dcbSerialParams)){

std::cout << "Failed to set serial port state for port" << std::endl;

}

}

void transferData(){

std::cout << "Input char: ";

char data = getchar();

unsigned long BytesWritten;

if (!WriteFile(Port1, &data, sizeof(data), &BytesWritten, NULL)){

std::cout << "Error in writing to first serial port" << std::endl;

return;

}

char readedData;

unsigned long BytesRead;

if (!ReadFile(Port2, &readedData, 1, &BytesRead, NULL)){

std::cout << "Error in reading from second serial port" << std::endl;

return;

}

std::cout << "Data read from second port: " << data << std::endl;

}

void closeSerialPorts()

{

if (Port1 != INVALID\_HANDLE\_VALUE){

CloseHandle(Port1);

}

if (Port2 != INVALID\_HANDLE\_VALUE){

CloseHandle(Port2);

}

}

int main() {

LPCWSTR PortName1 = L"COM1";

LPCWSTR PortName2 = L"COM2";

initializeSerialPorts(PortName1, PortName2);

transferData();

closeSerialPorts();

return 0;

}

1. **Программирование СОМ-порта с помощью функций BIOS.**

.model small

.stack 100h

.data

Enter db 0Dh,0Ah, '$'

Error\_Write db "Write error!",0Dh,0Ah,'$'

Error\_Read db "Read error!",0Dh,0Ah,'$'

Information db "Byte sent: $"

Byte db "$"

InputMessage db "Input char to send: $"

OutputMessage db "Readed byte: $"

.code

jmp start

Init\_COM1 proc

xor ax,ax

mov al,11100011b ; speed 9600 baud, without check,stop bit, 8 bit word len

mov dx,0 ; port number

int 14h

ret

Init\_COM1 endp

IsWrite\_COM1 proc

mov al,Byte

mov ah,1 ; write byte

mov dx,0 ; port number

int 14h

test ah,10000000b; check 7 bit (0 - correct write,1 - error)

jnz NoWRite

ret

IsWrite\_COM1 endp

NoWRite proc

mov ah,9

mov dx,offset Error\_Write

add dx,2

int 21h

ret

NoWRite endp

IsRead\_COM2 proc

mov ah,2 ; read byte

mov dx,1 ; port number

int 14h

test ah,10000000b; check read

jnz NoRead

ret

IsRead\_COM2 endp

NoRead proc

mov ah,9

mov dx,offset Error\_Read

add dx,2

int 21h

ret

NoRead endp

Output proc

mov ah,02h

mov dl,al

int 21h

ret

Output endp

Exit proc

mov ax,4C00h

int 21h

ret

Exit endp

InputChar proc

mov ah, 00h ;read one symbol

int 16h

mov Byte, al

mov ah, 0Eh

mov al, Byte

int 10h

ret

InputChar endp

OutputStr proc

push ax

mov ah, 9

int 21h

pop ax

ret

OutputStr endp

start:

mov ax,data

mov ds,ax

mov dx, offset InputMessage

call OutputStr

call InputChar

mov Byte, al

mov dx, offset enter

call OutputStr

call Init\_COM1

call IsWrite\_COM1

xor ax, ax

call IsRead\_COM2

mov dx, offset OutputMessage

call OutputStr

call Output

call Exit

end start

1. **Через регистры как с устройством ввода-вывода.**

#include <windows.h>

#include <conio.h>

#include <stdio.h>

#include <dos.h>

// Функция инициализации порта с заданной скоростью передачи данных

int com\_init(int base\_port);

// Функция отправки символа через указанный порт

void com\_outchar(char ch, int base\_port);

// Функция чтения символа из указанного порта

char com\_inchar(int base\_port);

int com\_init(int base\_port)

{

//0x3f8+3 = 0x3fb (LCR)

//если старший бит в 1 то происходит деление частоты порта

//что даёт доступ к DLL и DLN

outp(base\_port + 0x03, 0x80); //10000000 (байт загоняем)

//0x3f8+1 = 0x3f9

outp(base\_port + 0x01, 0x00); //они работают как пара, старшая часть (DLM) в ноль(сброс в нач.состояние)

outp(base\_port, 0x0C); //а младшая часть (DLL) в скорость(9600) 00001100

//0x3f8+3 = 0x3fb (LCR)

outp(base\_port + 0x03, 0x00); //0 чтобы вернуться в обычный режим работы, как передача данных

//0x3f8+3 = 0x3fb (LCR)

outp(base\_port + 0x03, 0x02); //(00000.0.10) 0 - 1 стоп бит, 10 - 7 рабочих битов

//0x3f8+4 = 0x3fс (MCR)

outp(base\_port + 0x04, 0x00); //настройка регистра управления модемом, DTR=0 RTS=0

//выключение диагностики, запрет прерываний, чтобы ничего не мешало

//DTR = 0 - сброшен, готов к отправке данных,

//RTS = 0 - сброшен, данных для передачи нет

return 1;

};

void com\_outchar(char ch, int base\_port)

{

//0x3f8+4 = 0x3fс (MCR)

char regst = inp(base\_port + 0x04); //чтение регистра управления

outp(base\_port + 0x04, regst | 0x02); //хххххххх | 00000010 бит RTS=1,

//активный уровень, готовность к передаче данных

outp(base\_port, ch); //отправка символа в данные порта

delay(100); //задержка для успешной передачи (ожидание готовности передатчика)

outp(base\_port + 0x04, regst & 0xfd); //0x3f8+4 = 0x3fс (MCR)

//хххххх1х & 11111101 бит RTS=0,

}

char com\_inchar(int base\_port)

{

return inp(base\_port); //чтение символа из порта

}

int main()

{

int port1 = 0x3f8;

int port2 = 0x2f8;

char inf;

if (!com\_init(port1))

puts("(1)some ERROR");

if (!com\_init(port2))

puts("(1)some ERROR");

printf("\nport 1 before sending : %c\n", com\_inchar(port1));

printf("port 2 before sending : %c\n\n", com\_inchar(port2));

puts("Enter some char to snd:");

rewind(stdin);

inf = getchar();

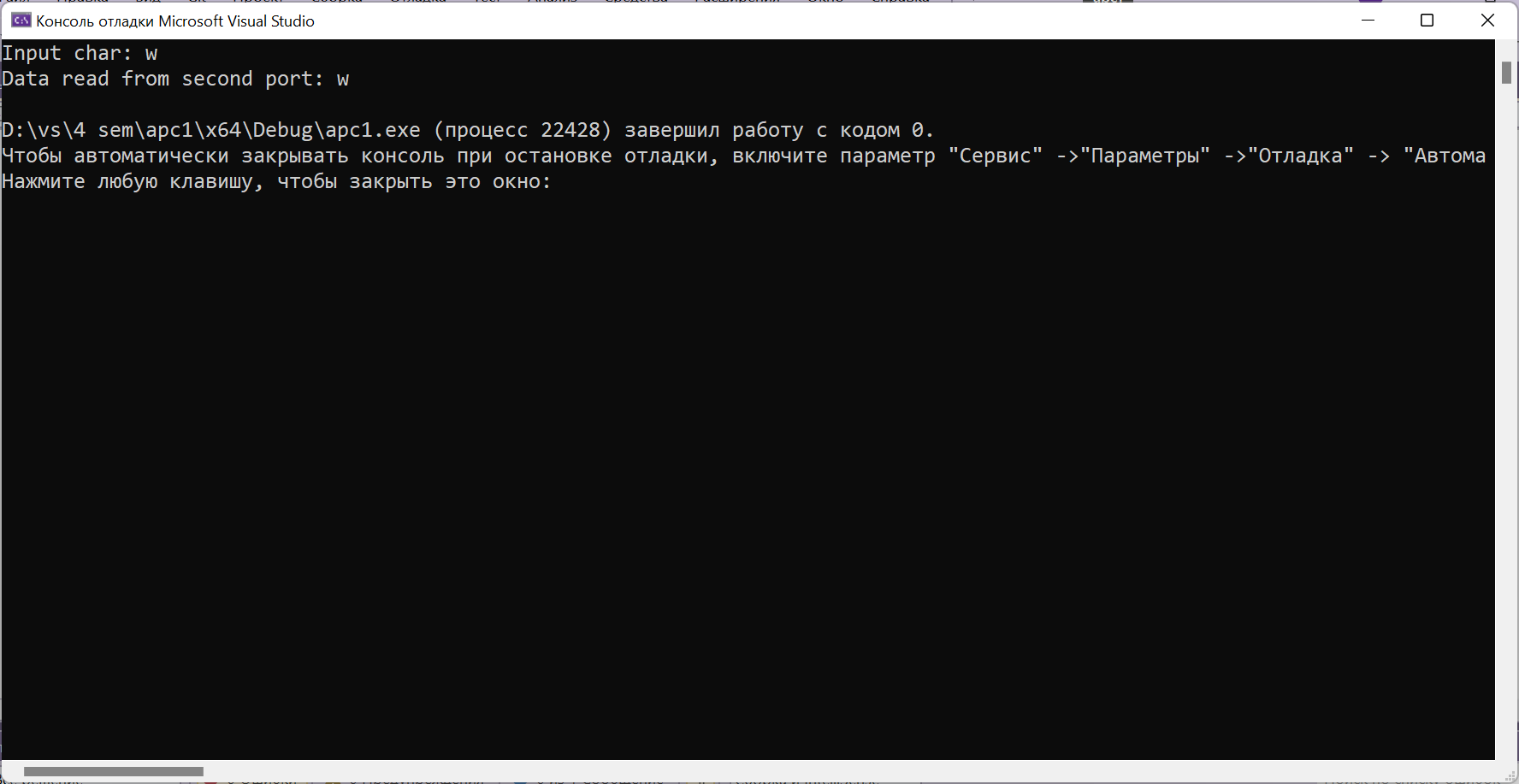
com\_outchar(inf, port1);

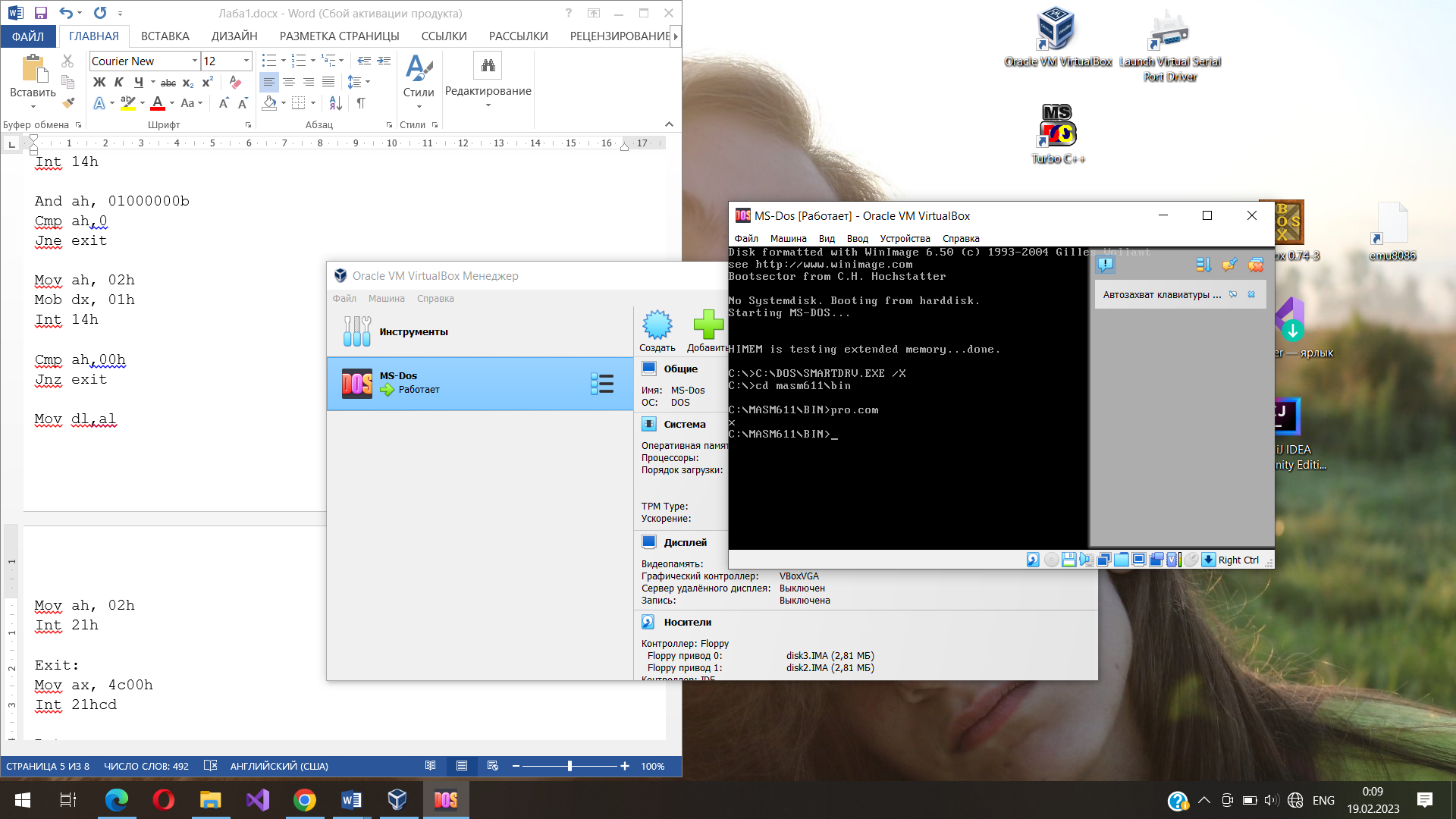
printf("\nport 1 after sending : %c\n", com\_inchar(port1));

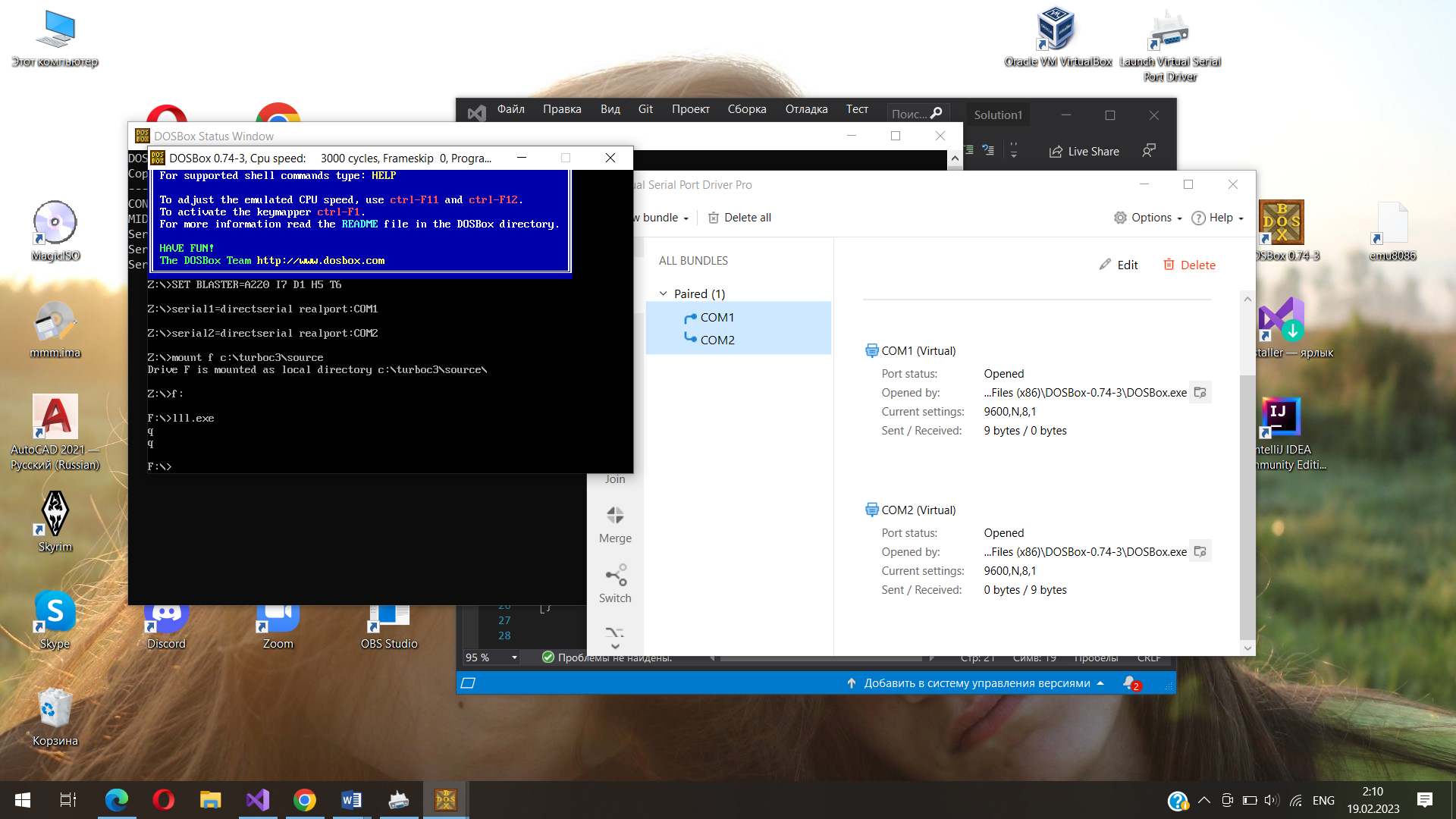
printf("port 2 after sending : %c\n\n", com\_inchar(port2));

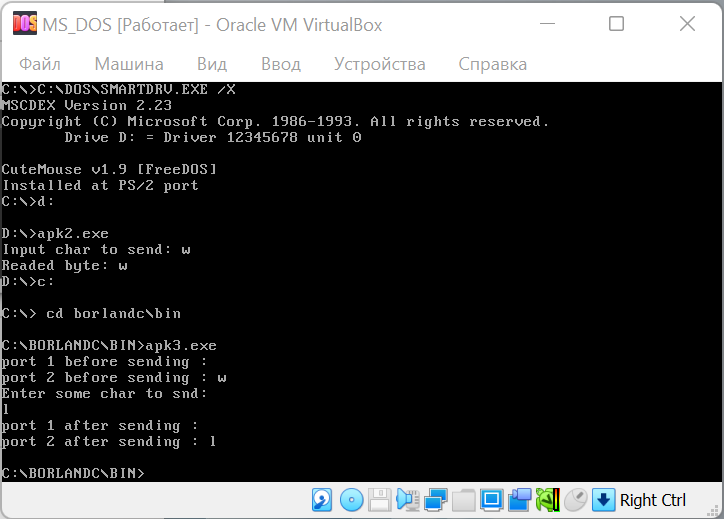
}

**Тест**









**Заключение**

В данной лабораторной работе разработаны 3 программы, которые позволяют записывать в данные в один COM-порт, пересылать в другой COM-порт и читать из него (в данном случае это пара COM1 – COM2).