Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра электронных вычислительных машин

Отчёт по лабораторной работе № 1 на тему «Последовательный порт»

Студент группы № 950503

Полховский А.Ф.

Преподаватель

Одинец Д. Н.

Задача

Разработать программный модуль 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 прерывания 14h,
- 3. работа с СОМ-портом через регистры как с устройством ввода-вывода.

Алгоритм

Передача данных через последовательный интерфейс состоит из следующих фаз:

- инициализация порта;
- передача данных;
- прием данных;
- анализ состояния порта.

Программное взаимодействие с последовательным интерфейсом может осуществляться с использованием различных механизмов. Самый прозрачный из них - это прямое взаимодействие с портами ввода-вывода. Более высокоуровневые методы: использование прерывания 14h, работа с СОМ-портом как с устройством ввода-вывода.

Далее приведены листинги программ, реализующие процедуры передачи (приема) байта информации через последовательный интерфейс.

Листинг программы, использующей WinAPI:

Read.cpp:

```
#include <iostream>
#include <windows.h>
using namespace std;

int main() {
    HANDLE hSerial = ::CreateFile(L"COM1", GENERIC_READ | GENERIC_WRITE, 0,
0, OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, 0);//инициализация порта
    DWORD iSize;
    char sReceivedChar;
    if (hSerial == INVALID_HANDLE_VALUE)//проверка порта
    {
        if (GetLastError() == ERROR_FILE_NOT_FOUND)
        {
            cout << "serial port does not exist.\n";
        }
        cout << "some other error occurred.\n";
    }
    DCB dcbSerialParams = { 0 };</pre>
```

```
dcbSerialParams.DCBlength = sizeof(dcbSerialParams);
    if (!GetCommState(hSerial, &dcbSerialParams))
    {
        cout << "getting state error\n";</pre>
    }
    dcbSerialParams.BaudRate = CBR 9600; //настройка порта
    dcbSerialParams.ByteSize = 8;
    dcbSerialParams.StopBits = ONESTOPBIT;
    dcbSerialParams.Parity = NOPARITY;
    if (!SetCommState(hSerial, &dcbSerialParams))
        cout << "error setting serial port state\n";</pre>
    while (true)
        ReadFile(hSerial, &sReceivedChar, 1, &iSize, 0); // получаем 1 байт
        if (iSize > 0) // если что-то принято, выводим
            cout << sReceivedChar;</pre>
    }
}
Write.cpp:
#include <iostream>
#include <windows.h>
using namespace std;
int main() {
    HANDLE hSerial = ::CreateFile(L"COM2", GENERIC READ | GENERIC WRITE, 0,
0, OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, 0);
                                                 //инициализация порта
    if (hSerial == INVALID_HANDLE_VALUE)
    {
        if (GetLastError() == ERROR_FILE_NOT_FOUND) //проверка порта
        {
            cout << "serial port does not exist.\n";</pre>
        cout << "some other error occurred.\n";</pre>
    DCB dcbSerialParams = { 0 };
    dcbSerialParams.DCBlength = sizeof(dcbSerialParams);
    if (!GetCommState(hSerial, &dcbSerialParams))
    {
        cout << "getting state error\n";</pre>
    }
    dcbSerialParams.BaudRate = CBR 9600;
                                                     //настройка порта
    dcbSerialParams.ByteSize = 8;
    dcbSerialParams.StopBits = ONESTOPBIT;
    dcbSerialParams.Parity = NOPARITY;
    if (!SetCommState(hSerial, &dcbSerialParams))
    {
        cout << "error setting serial port state\n";</pre>
    }
    char data[] = "\nI HELLO WORLD! tregrge\n"; // строка для передачи
    DWORD dwSize = sizeof(data); // размер этой строки
```

```
DWORD dwBytesWritten; // тут будет количество собственно переданных байт

BOOL iRet = WriteFile(hSerial, data, dwSize, &dwBytesWritten, NULL);

cout << dwSize << " Bytes in string. " << dwBytesWritten << " Bytes
sended. " << endl;

return 0;
}
```

Листинг программы, использующей BIOS прерывания 14h: Int14.com:

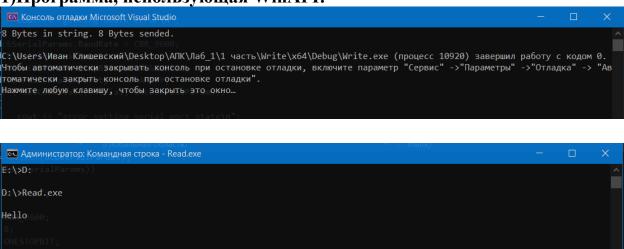
```
.model small
.code
.stack 100h
start:
mov ah, 0 ;port initialization mov al, 11100011b ;9600/8n1
mov dx, 0 ;port COM1
int 14h
Main_loop:
mov ah, 2
                  ;Get byte from modem
int 14h
test ah, ah
                  ;If anything received
jnz no_input
int 29h
no_input:
mov ah, 1
int 16h
jz Main_loop
mov ah, 8
int 21h
test al, al
jnz send_char
int 21h
cmp al, 2Dh
jne send_char
mov ax, 4C00h
int 21h
send_char:
mov ah, 1
int 14h
                  ;Send the intered character
jmp short Main_loop
end start
```

Листинг программы, работающей с СОМ-портом через регистры как с устройством ввода-вывода:

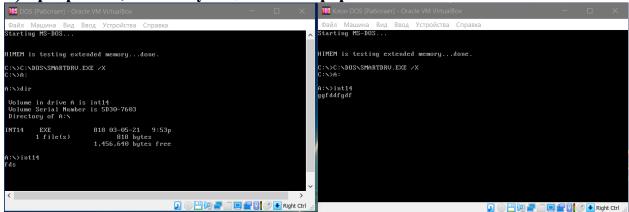
```
.model tiny
org 100h
.code
start:
    mov ah,0
    mov al,11100011b
    mov dx,0
    int 14h
main_loop:
    call readByteCom
    cmp ah,ah
    jnz noInput
    int 29h
noInput:
    mov ah,1
    int 16h
    jz main_loop
    mov ah,8
    int 21h
    test al,al
    jnz sendByteCom
    int 21h
    cmp al,2Dh
    jne sendByteCom
    ret
exit:
    mov ax,4C00h
    int 21h
readByteCom:
    mov dx,2F8h
    in al,dx
    ret
sendByteCom:
    mov dx,3F8h
    mov ah,01h
    int 21h
    out dx,al
    int 29h
    ret
```

Демонстрация работы:

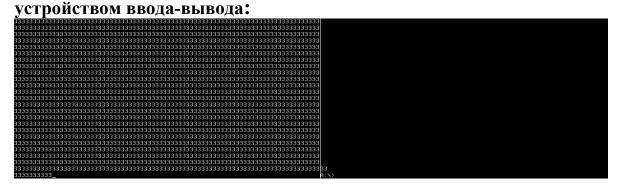
1)Программа, использующая WinAPI:



2) программа, использующая ВІОЅ прерывания 14h:



3) программа, работающая с СОМ-портом через регистры как с



В ходе данной лабораторной работы были изучены возможности использования последовательных интерфейсов RS-232, разработаны алгоритмы передачи данных. Программное взаимодействие с последовательным интерфейсом осуществлялось с использованием различных механизмов. Самый прозрачный из них - это прямое взаимодействие с портами ввода-вывода. Более высокоуровневые методы: использование прерывания 14h, работа с СОМ-портом как с устройством ввода-вывода.

Для эмуляции COM-портов использовался Virtual Serial Port Driver pro, для эмуляции DOS — Oracle Virtual Box.