Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра электронных вычислительных машин

Лабораторная работа №1
«Последовательный порт»

Выполнил:
Студент группы 150504
Желубовский С.В.

Проверил:
Преподаватель
Одинец Д.Н.

Минск, 2023

1. Постановка задачи

Разработать программный модуль реализации процедуры передачи (приёма) байта информации через последовательный интерфейс.

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

1. Прямое взаимодействие с портами ввода-вывода (wirte, read)
2. Использование BIOS прерывания 14h
3. Работа с COM-портом через регистры как с устройствами ввода-вывода.

2. Алгоритм

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

- Инициализация порта
- Запись байта информации в порт
- Чтение байта информации из порта
- Вывод результата на экран

3. Листинг программы

Далее приведены листинги программ, реализующие различные механизмы передачи (приёма) информации через последовательный интерфейс.

3.1. Листинг программы, взаимодействующей с портами ввода-вывода.

```
#include <windows.h>
#include <iostream>
#include <string>

using namespace std;

int main()
{
    HANDLE COM_1;
    LPCTSTR Port_1 = L"COM7";
    HANDLE COM_2;
    LPCTSTR Port_2 = L"COM8";
    LPVOID *lpMsgBuf;

    COM_1 = CreateFile(Port_1,
        GENERIC_WRITE,
        0,
        0,
        OPEN_EXISTING,
        FILE_ATTRIBUTE_NORMAL,
        0);
```

```
    if (COM_1 == INVALID_HANDLE_VALUE)
    {
        DWORD dw = GetLastError();
        cout << "Error opening com 1, error code: " << dw << endl;
        return 1;
    }

    COM_2 = CreateFile(Port_2,
        GENERIC_READ,
        0,
        0,
        OPEN_EXISTING,
        FILE_ATTRIBUTE_NORMAL,
        0);

    if (COM_2 == INVALID_HANDLE_VALUE)
    {
        DWORD dw = GetLastError();
        cout << "Error opening com 2, error code: " << dw << endl;
        return 1;
    }

    string messege = "Default messege";

    cout << "Enter the string to share" << endl;
    cin >> messege;

    DCB Serial_Params = { 0 };
    Serial_Params.DCBlength = sizeof(Serial_Params);
    if (!GetCommState(COM_1, &Serial_Params))
    {
        cout << "Getting state error.\n";
    }
    Serial_Params.BaudRate = CBR_9600;
    Serial_Params.ByteSize = 8;
    Serial_Params.StopBits = ONESTOPBIT;
    Serial_Params.Parity = NOPARITY;
    if (!SetCommState(COM_2, &Serial_Params))
    {
        cout << "Error setting serial port state.\n";
    }


    DWORD Size = sizeof(messege);
    DWORD Bytes_Written;

    BOOL Ret = WriteFile(COM_1, &messege, Size, &Bytes_Written, NULL);

    messege.clear();

    cout << Size << " Bytes in string. " << Bytes_Written << " Bytes
sended. " << endl;

    if (ReadFile(COM_2, &messege, sizeof(messege), &Size, 0)) {
        cout << endl << "Data from COM2: '" << messege << "'";
```

```
    }

    return 0;
}
```

3.2. Листинг программы, использующей BIOS прерывание 14h.

```
.model small
.stack 100h

.data

Error_Write db "Write error!",0Dh,0Ah,'$'
Error_Read db "Read error!",0Dh,0Ah,'$'
InputStroka db "Enter the symbol: ",0Dh,0Ah,'$'
OutputStroka db "Your symbol: $"

.code

Exit proc
    mov ax,4C00h
    int 21h
    ret
Exit endp

ReadCL proc

   mov SI,80h
   xor CX,CX
   mov CL,[SI]
   inc SI
   rep movsb
   mov AL,0
   stosb
ret
ReadCL endp

start:
    mov ax, data
    mov ds, ax

    xor ax,ax ; initialization Com1
    mov al,10100011b;bit mask
    mov dx,0
    int 14h

    xor ax, ax
```

```
    mov ah, 09h
    mov dx, offset InputStroka
    int 21h

    xor ax, ax

    MOV AH, 01H
    INT 21H

    mov ah,1
    mov dx,0
    int 14h
    test al,80h
    jnz NoWRite

    mov al,'e'   ; Is read Com2
    mov ah,2
    mov dx,1
    int 14h
    test al,80h
    jnz NoRead

    mov ah,02h     ;Output
    mov dl,al
    int 21h


   call Exit

NoWrite:
   mov ah,9
   mov dx,offset Error_Write
   add dx,2
   int 21h
   call Exit

NoRead:
   mov ah,9
   mov dx,offset Error_Read
   add dx,2
   int 21h
   call Exit



end start
```

3.3. Листинг программы, работающей с COM-портами через регистры как с устройствами ввода-вывода.

```
data segment
    writingError db 10, 13, "Write error!$"
    readingError db 10, 13, "Read error!$"
    output db "Your symbol: $"
    dataForSending db ?
    dataForReading db ?
    enterSymbol db "Enter the symbol: $"
data ends

code segment



    Exit proc
        mov ax, 4C00h
        int 21h
        ret
    Exit endp

    start:
        mov ax, @data
        mov ds, ax

        mov al, 80h      ; initialize
        mov dx, 3FBh
        out dx, al

        mov dx, 3F8h
        mov al, 00h
        out dx, al
        mov al, 0Ch
        mov dx, 3F9h
        out dx, al

        mov dx, 3FCh
        mov al, 00001011b
        out dx ,al

        mov dx, 3F9h
        mov al, 0
        out dx, al

        xor al, al ; Is Writed in com1
        mov dx, 3FDh
        in al, dx
```

```
test al, 10h
jnz NoWrite


mov ah, 9h                  ;read symbol
mov dx, offset enterSymbol
int 21h

xor ax, ax

MOV AH, 01H
INT 21H
mov dataForSending, al

mov ah, 02h
mov dl, 0ah
int 21h

mov ah, 02h
mov dl, 0dh
int 21h


mov dx, 3F8h           ;send data
mov al, dataForSending
out dx, al

mov al, 02h

xor al, al    ;is readed from com2
mov dx, 3FDh
in al, dx
test al, 10b
jnz NoRead


mov dx, 3F8h     ; read data
in al, dx
mov dataForReading, al

mov dx, offset output
mov ah, 09h
int 21h

mov ah, 02h
mov dl, dataForReading
int 21h
```

```
        call Exit
    NoWrite:
        mov ah, 09h
        mov dx, offset writingError
        int 21h
        call Exit
    NoRead:
        mov ah, 09h
        mov dx, offset readingError
        int 21h
        call Exit

        code ends

end start
```

4. Тестирование программ

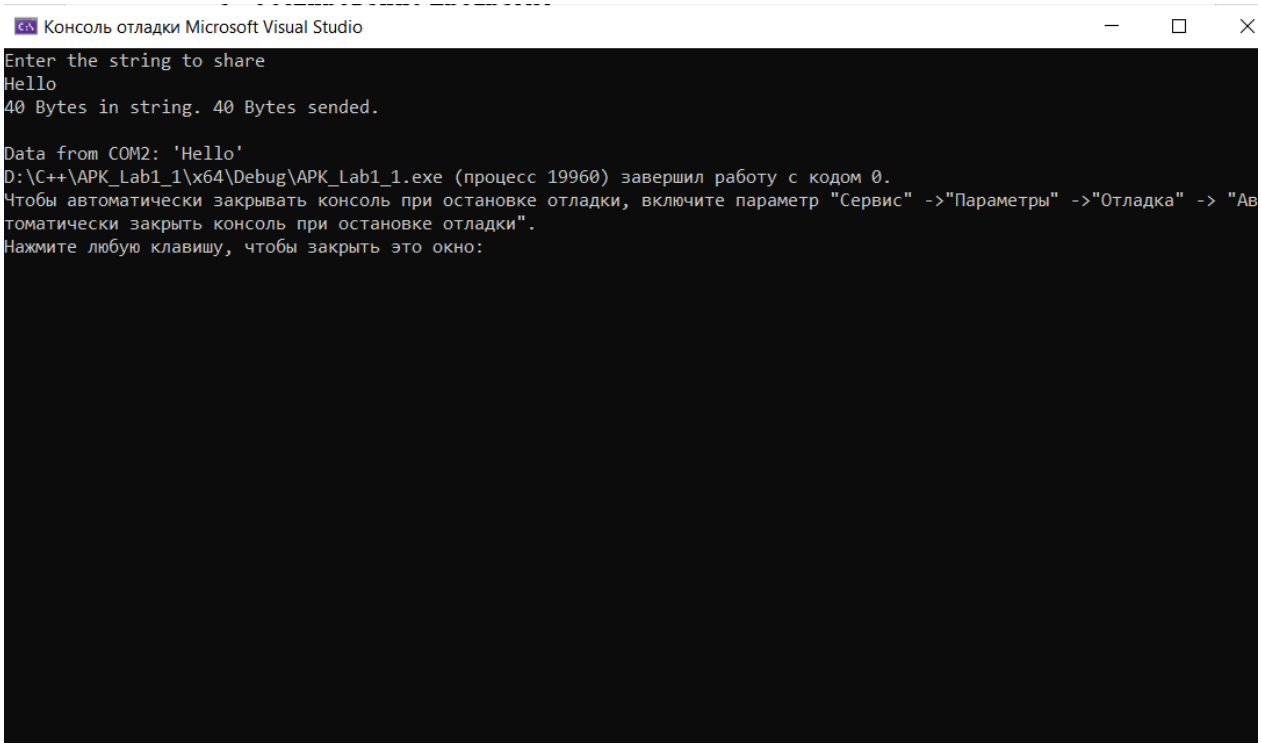

Рисунок 4.1 – Результат работы программы, взаимодействующей с портами ввода-вывода.

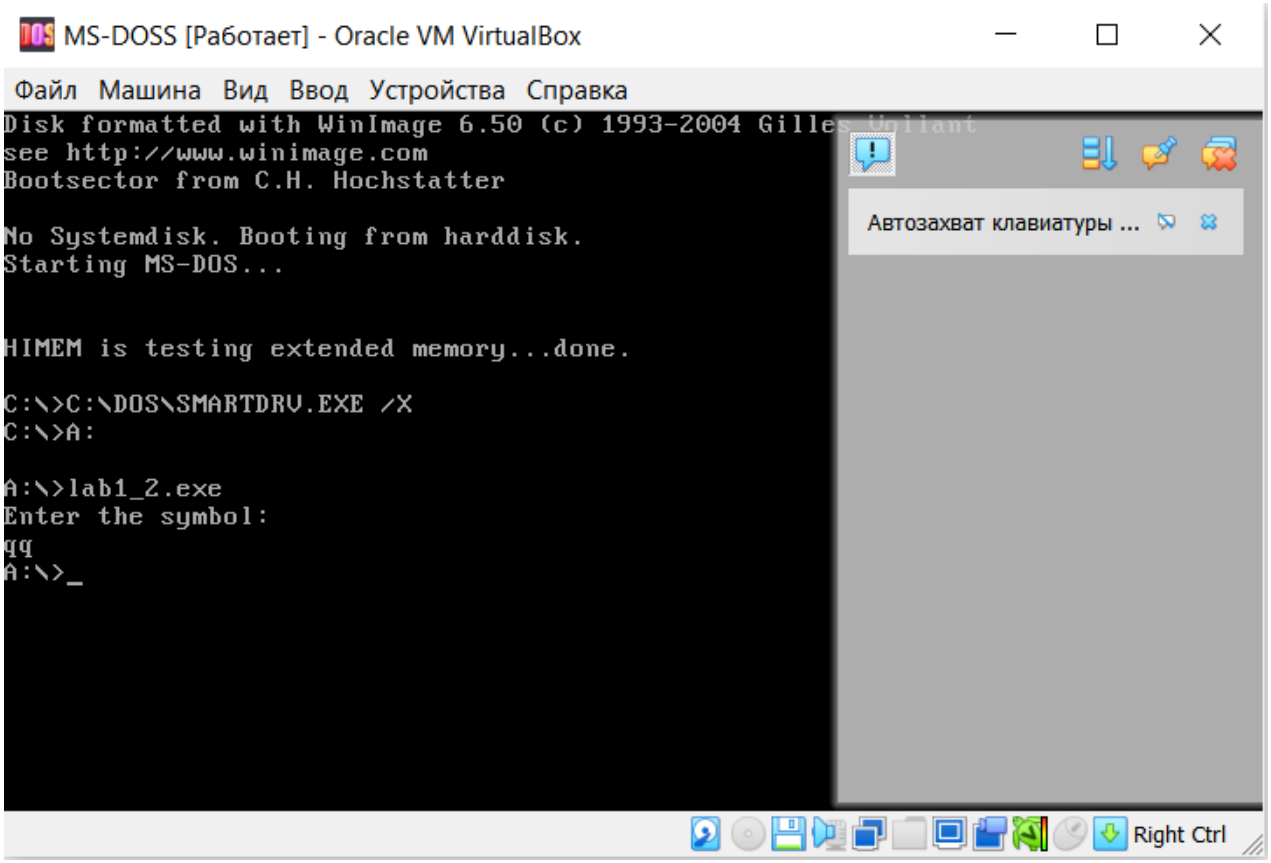

Рисунок 4.2 – Результат работы программы, использующей BIOS прерывание 14h.

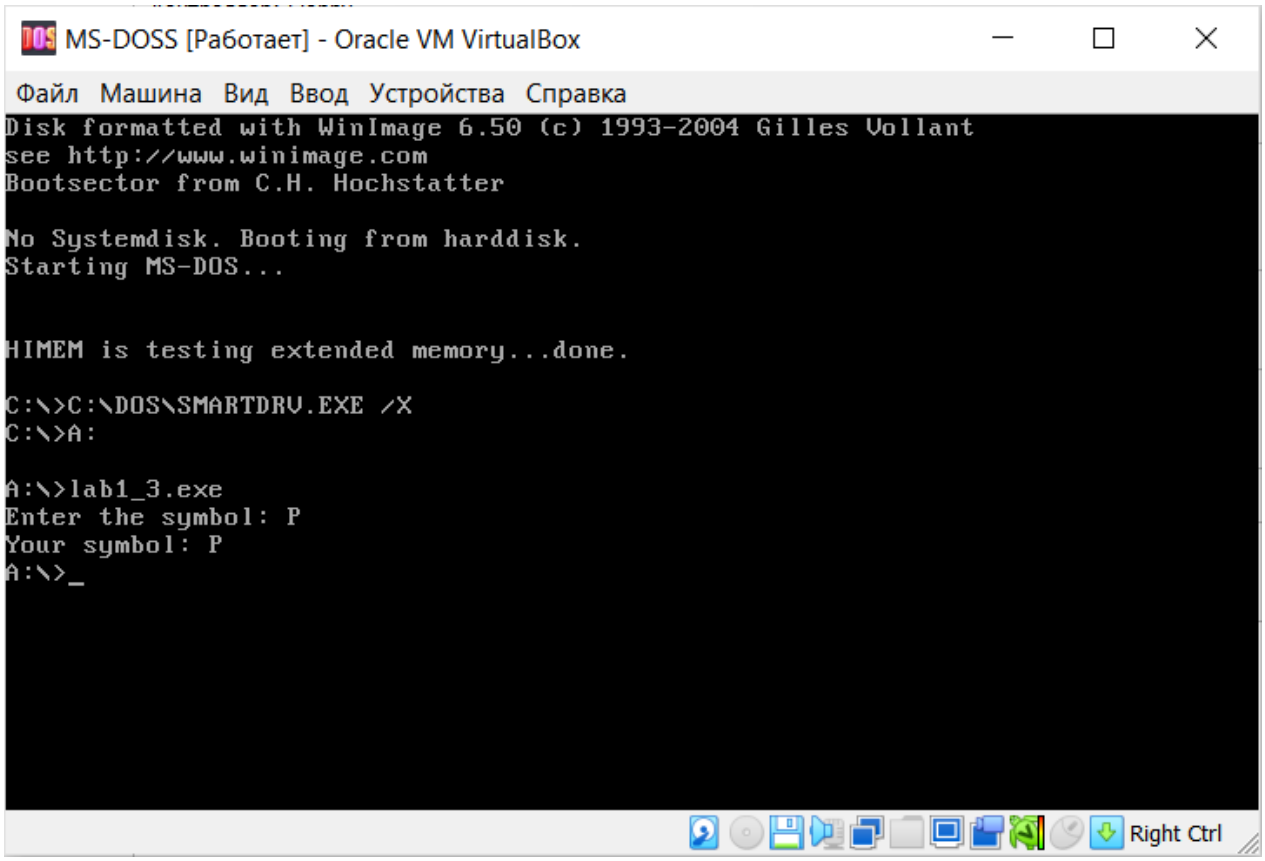

Рисунок 4.3 – Результат работы программы, работающей с COM-портами через регистры как с устройствами ввода-вывода.

5. Заключение

В ходе лабораторной удалось передать 1 байт информации через последовательный порт с использованием различных механизмов.

Для эмуляции COM портов использовался Null-modem emulator, для эмуляции DOS используется Oracle Virtual Box.