Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра ЭВМ

Отчет по лабораторной работе №1

«Последовательный порт»

Проверил: Выполнил:

к.т.н., доцент студент гр. 150501

Одинец Д.Н. Почебут А.С.

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 (write, read)
2. Использование BIOS прерывания 14h
3. Работа с COM-портом через регистры как с устройствами ввода-вывода.
4. 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

1. 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>

using namespace std;

HANDLE COM\_Port\_1;

LPCTSTR Port\_Name\_1 = L"COM1";

HANDLE COM\_Port\_2;

LPCTSTR Port\_Name\_2 = L"COM2";

void Read\_from\_COM()

{

DWORD Size;

char Received\_Char;

ReadFile(COM\_Port\_2, &Received\_Char, 1, &Size, 0);

if (Size > 0)

{

cout << "Initialising of sent bytes: " << Received\_Char;

}

}

int main()

{

COM\_Port\_1 = ::CreateFile(Port\_Name\_1, GENERIC\_WRITE, 0, 0, OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, 0);

COM\_Port\_2 = ::CreateFile(Port\_Name\_2, GENERIC\_READ, 0, 0, OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, 0);

if (COM\_Port\_1 == INVALID\_HANDLE\_VALUE)

{

if (GetLastError() == ERROR\_FILE\_NOT\_FOUND)

{

cout << "COM-port does not exist!\n";

}

cout << "Other error.\n";

}

DCB Serial\_Params = { 0 };

Serial\_Params.DCBlength = sizeof(Serial\_Params);

if (!GetCommState(COM\_Port\_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\_Port\_2, &Serial\_Params))

{

cout << "Error setting serial port state.\n";

}

char data = 'K';

DWORD Size = sizeof(data);

DWORD Bytes\_Written;

BOOL Ret = WriteFile(COM\_Port\_1, &data, Size, &Bytes\_Written, NULL);

cout << "Bytes in string: " << Size << endl << "Sent bytes:" << Bytes\_Written << endl;

Read\_from\_COM();

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,'$'

Information db "Sent byte: $"

.code

jmp start

Init\_COM1 proc

xor ax,ax

mov al,10100011b

mov dx,0

int 14h

ret

Init\_COM1 endp

IsWrite\_COM1 proc

mov al,'K'

mov ah,1

mov dx,0

int 14h

test al,80h

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

mov dx,1

int 14h

test al,80h

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

start:

call Init\_COM1

call IsWrite\_COM1

call IsRead\_COM2

push ax

mov ah,9

mov dx,offset Information

add dx,2

int 21h

pop ax

call Output

call Exit

end start

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

.model small

.stack 100h

.data

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

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

Information db "Sent byte: $"

Data\_Byte db 'K'

Data\_Byte2 db ?

.code

Init\_COM1 proc

mov al,80h

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

ret

Init\_COM1 endp

IsWrite\_COM1 proc

xor al,al

mov dx,3FDh

in al,dx

test al,10h

jnz NoWRite

ret

IsWrite\_COM1 endp

NoWRite proc

mov ah,9

mov dx,offset Error\_Write

int 21h

ret

NoWRite endp

IsRead\_COM2 proc

xor al,al

mov dx,3FDh

in al,dx

test al,10b

jnz NoRead

ret

IsRead\_COM2 endp

NoRead proc

mov ah,9

mov dx,offset Error\_Read

int 21h

ret

NoRead endp

Send\_Byte proc

mov dx,3F8h

mov al,Data\_Byte

out dx,al

ret

Send\_Byte endp

Read\_Byte proc

mov dx,3F8h

in al,dx

mov Data\_Byte2,al

ret

Read\_Byte endp

Exit proc

mov ax,4C00h

int 21h

ret

Exit endp

start:

mov ax,@data

mov ds,ax

call Init\_COM1

call IsWrite\_COM1

call Send\_Byte

mov al,2

call IsRead\_COM2

call Read\_Byte

mov dx,offset Information

mov ah,9

int 21h

mov ah,02h

mov dl,Data\_Byte2

int 21h

call Exit

end start

1. Тестирование программ

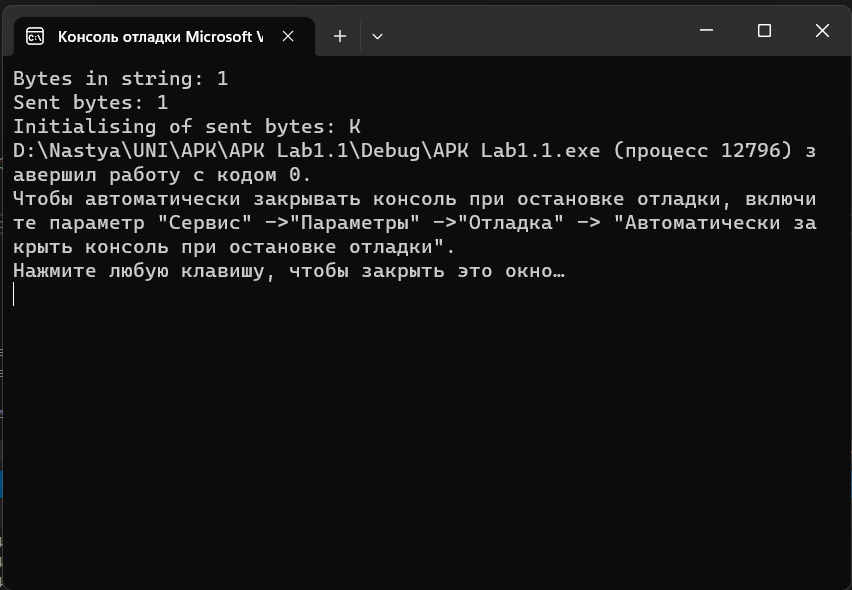


Рисунок 4.1 – Результат работы программы, взаимодействующей с портами ввода-вывода.

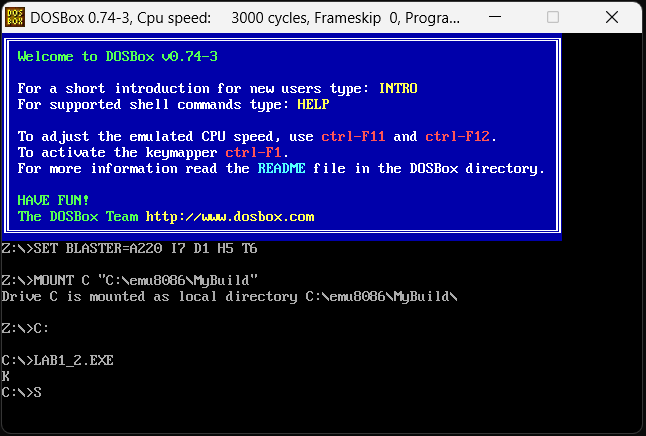


Рисунок 4.2 – Результат работы программы, использующей BIOS прерывание 14h.

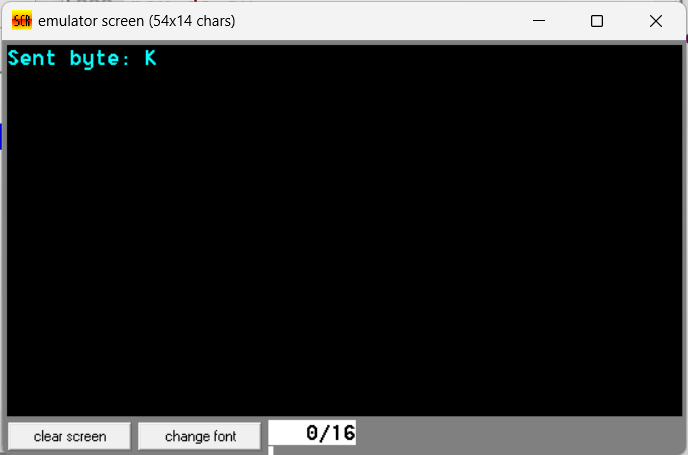


Рисунок 4.3 – Результат работы программы, работающей с COM-портами через регистры как с устройствами ввода-вывода.

1. Заключение

В ходе лабораторной удалось передать 1 байт информации через последовательный порт с использованием различных механизмов. Для эмуляции COM портов использовался com0com, для эмуляции DOS использовался DOSBox.