Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра электронных вычислительных машин

Лабораторная работа №1

«Последовательный порт»

Выполнил: Проверил:

Студент группы 250501 Преподаватель

Снитко Д.А. Одинец Д.Н.

Минск 2024

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.
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.h>

using namespace std;

// Глобальные переменные для COM портов

HANDLE COM\_Port\_1;

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

HANDLE COM\_Port\_2;

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

// Функция для чтения из порта COM2

void Read\_from\_COM()

{

DWORD Size;

char Received\_Char;

// Чтение символа из порта COM2

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

if (Size > 0)

{

cout << "Received: " << Received\_Char << endl; // Вывод принятого символа

}

}

int main()

{

// Открытие COM1 и COM2 для записи и чтения

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);

// Проверка открытия портов

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

{

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

{

cout << "COM-port 1 does not exist!\n";

}

cout << "Some other error opening COM-port 1.\n";

}

if (COM\_Port\_2 == INVALID\_HANDLE\_VALUE)

{

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

{

cout << "COM-port 2 does not exist!\n";

}

cout << "Some other error opening COM-port 2.\n";

}

// Настройка параметров порта COM1 для передачи данных

DCB Serial\_Params = { 0 };

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

if (!GetCommState(COM\_Port\_1, &Serial\_Params))

{

cout << "Getting state error for COM-port 1.\n";

}

Serial\_Params.BaudRate = CBR\_9600;

Serial\_Params.ByteSize = 8;

Serial\_Params.StopBits = ONESTOPBIT;

Serial\_Params.Parity = NOPARITY;

if (!SetCommState(COM\_Port\_1, &Serial\_Params))

{

cout << "Error setting serial port state for COM-port 1.\n";

}

char data = 'A';

DWORD Size = sizeof(data);

DWORD Bytes\_Written;

// Отправка данных из порта COM1

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

// Вывод информации о количестве отправленных байт

cout << Size << " Bytes in string. " << Bytes\_Written << " Bytes sent from COM-port 1. " << endl;

// Чтение данных из порта COM2

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 "Byte sent: $"

.code

jmp start

Init\_COM1 proc

xor ax,ax

mov al,10100011b ; Устанавливаем параметры порта COM1 (9600 бит/с, 8 бит данных, 1 стоп-бит, без проверки четности)

mov dx,0

int 14h

ret

Init\_COM1 endp

IsWrite\_COM1 proc

mov al,'A' ; Пытаемся записать символ 'A' в порт COM1

mov ah,1

mov dx,0

int 14h

test al,80h ; Проверяем флаг успешной записи

jnz NoWrite ; Если флаг установлен, переходим к обработке ошибки записи

ret

IsWrite\_COM1 endp

NoWrite proc

mov ah,9

mov dx,offset Error\_Write ; Выводим сообщение об ошибке записи

add dx,2

int 21h

ret

NoWrite endp

IsRead\_COM2 proc

mov ah,2 ; Читаем символ из порта COM2

mov dx,1

int 14h

test al,80h ; Проверяем флаг успешного чтения

jnz NoRead ; Если флаг установлен, переходим к обработке ошибки чтения

ret

IsRead\_COM2 endp

NoRead proc

mov ah,9

mov dx,offset Error\_Read ; Выводим сообщение об ошибке чтения

add dx,2

int 21h

ret

NoRead endp

Output proc

mov ah,02h ; Выводим символ в стандартный вывод

mov dl,al

int 21h

ret

Output endp

Exit proc

mov ax,4C00h ; Завершаем программу

int 21h

ret

Exit endp

start:

call Init\_COM1 ; Инициализируем порт COM1

call IsWrite\_COM1 ; Пытаемся записать символ 'A' в порт COM1

mov al,'e' ; Читаем символ из порта COM2

call IsRead\_COM2

call Output ; Выводим прочитанный символ

call Exit

end start

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

data segment

msg\_sent db "Sent ", '$'

msg\_got db 0Ah, 0Dh, "Received ", '$'

msg\_error db "Error", 0Ah, 0Dh, '$'

buf1 db '1'

buf2 db 0

ends data

stck segment

dw 128 dup(0)

ends stck

code segment

ASSUME ss:stck, ds:data, cs:code

out\_str macro str

mov ah, 09h

lea dx, str

int 21h

endm

main:

mov ax, data

mov ds, ax

mov dx, 3fBh ;LCR register adress

mov al, 10000011b ;init com1 (DLAB=1, 8 bits in symbol, 1 stopbit, no paritet)

out dx, al

mov dx, 3f8h

mov al, 0Ch

out dx, al ;write DIM (bit/sec = 9600)

mov dx, 3f9h

xor al, al

out dx, al ;write DLL=0

mov dx, 3fBh

mov al, 00000011b ;DLAB=0

out dx, al ;write to LCR

out\_str msg\_sent ;msg that byte sent

ReadyTHRECheck:

mov dx, 3fdh

in al, dx

test al, 00100000b ;check 5-th bit of LSR (THRE)

jz ReadyTHRECheck

mov dx, 3F8h

mov al, buf1

out dx, al ; Send data to the COM1.

mov dx, 3FDh

in al, dx

test al, 00001110b ; Check errors (check LSR)

jnz error

mov ah, 02h

mov dl, buf1

int 21h ;output sent character

waitDR: ; Wait 1st bit of LSR register (DR)

mov dx, 2FDh

in al, dx

test al, 01h ; Check 1st bit.

jz waitDR

mov dx, 2f8h

in al, dx ;read from com2

mov buf2, al ;save byte to buf2

out\_str msg\_got ;msg that byte sent

mov ah, 02h

mov dl, buf2

int 21h ;output received character

finish:

mov ax, 4c00h ;exit to operating system.

int 21h

error:

out\_str msg\_error

jmp finish

ends code

end main

1. Тестирование программ

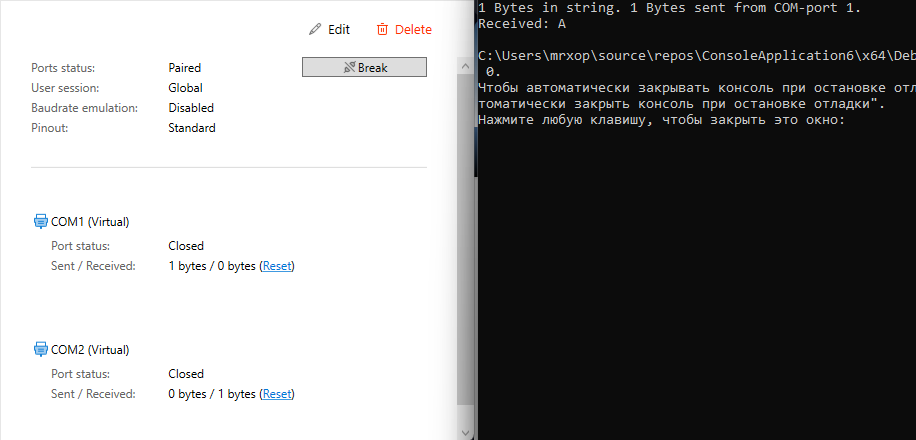


Рисунок 4.1 – Результат работы программы, взаимодействующей с портами ввода-вывода

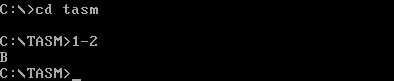


Рисунок 4.3 – Результат работы программы, использующей BIOS прерывание 14h.

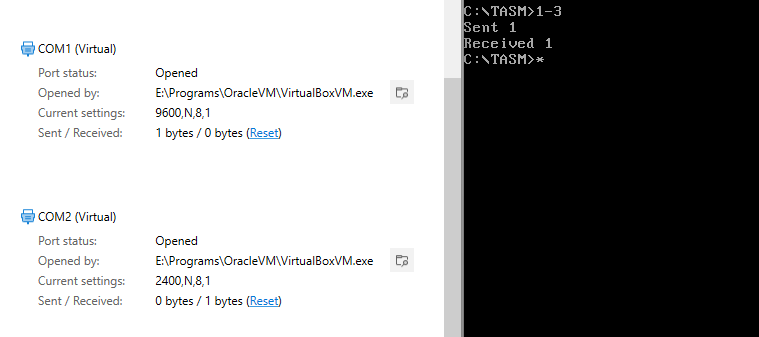


Рисунок 4.3 – Результат работы программы, работающей с COM-портами через регистры как с устройствами ввода-вывода.