Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра электронных вычислительных машин

# Лабораторная работа №1
# «Последовательный порт»

Выполнил:
Студент группы 050503
Григорик И. А.

Проверил:
Преподаватель
Одинец Д. Н.

Минск, 2021

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

void read_from_com(HANDLE &COM2);

int main() {
    HANDLE COM1;
    LPCTSTR port_name_1 = "COM1";

    COM1 = CreateFile(port_name_1,
                      GENERIC_WRITE,
                      0,
                      nullptr,
                      OPEN_EXISTING,
                      FILE_ATTRIBUTE_NORMAL,
                      nullptr);

    HANDLE COM2;
    LPCTSTR port_name_2 = "COM2";
    COM2 = CreateFile(port_name_2, GENERIC_READ, 0, nullptr, OPEN_EXISTING,
FILE_ATTRIBUTE_NORMAL, nullptr);

    if (COM1 == INVALID_HANDLE_VALUE) {
        printf("Error\n");
    } else {
        printf("Open serial port to write successful\n");
    }
```

```
    DCB params = {0};
    params.DCBlength = sizeof(params);
    if(!GetCommState(COM1, &params)){
        printf("Params COM1 error\n");
    }

    params.BaudRate = CBR_9600;
    params.ByteSize = 8;
    params.StopBits = ONESTOPBIT;
    params.Parity = NOPARITY;

    if(!SetCommState(COM2, &params)){
        printf("Params COM2 error\n");
    }

    char data = 'A';
    DWORD Size = sizeof(data);
    DWORD bytes_written;

    WriteFile(COM1, &data, Size, &bytes_written, nullptr);

    std::cout << Size << " bytes in string. " << bytes_written << " bytes
sent." << std::endl;


    read_from_com(COM2);
}

void read_from_com(HANDLE &COM2){
    DWORD Size;
    char Received_char;

    ReadFile(COM2, &Received_char, 1, &Size, nullptr);
    if(Size > 0){
        std::cout << Received_char << std::endl;
    }
}
```

### 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

;#########################################################################################
; WORK WITH TRANSMITTED PORT
```

```
;###############################################################################
Init_COM1 proc
  xor ax,ax                         ; Clear ax register
  mov al,10100011b                  ; Set transfer frequency
  mov dx,0                                  ; Initialize port name
  int 14h
  ret
Init_COM1 endp

IsWrite_COM1 proc
  mov al,'A'                        ; Initialize symbol
  mov ah,1                                  ; Write symbol to the port
  mov dx,0                                  ; Initialize port name
  int 14h
  test al,80h                       ; Test DSR
  jnz NoWRite                       ; If we cant write ... @099
  ret
IsWrite_COM1 endp

; Support function
NoWRite proc
  mov ah,9
  mov dx,offset Error_Write         ;@099 ... - show error message
  add dx,2
  int 21h
  ret
NoWRite endp

;###############################################################################
; WORK WITH RECIVED PORT
;###############################################################################
IsRead_COM2 proc                    ; Read in the second port
   mov ah,2                         ; Read symbol
   mov dx,1                         ; Initialize port name
   int 14h
   test al,80h                      ; Test RTS
   jnz NoRead                       ; If we cant write ... @099
   ret
IsRead_COM2 endp

NoRead proc
  mov ah,9
  mov dx,offset Error_Read          ;@099 ... - also show error message!
  add dx,2
  int 21h
  ret
NoRead endp
```

```
;###########################################################################################
; OUTPUT BYTE
;###########################################################################################
Output proc
  mov ah,02h                              ; Read byte from secong port
  mov dl,al                               ; And show
  int 21h
  ret
Output endp

;###########################################################################################
; EXIT FUNCTION
;###########################################################################################
Exit proc
   mov ax,4C00h
   int 21h
   ret
Exit endp

;###########################################################################################
; MAIN FUNCTION
;###########################################################################################
start:
  call Init_COM1
  call IsWrite_COM1
  mov al,'e'
  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 "Byte sent: $"
Data_Byte db 'A'
Data_Byte2 db ?

.code

;################################################################################
; INITIALIZE FIRST PORT
;################################################################################
Init_COM1 proc
; set 7 bit = 1 to 3FB -- 3F8, 3F9 able to control speed
   mov al,80h                          ; 7bit = 1
   mov dx,3FBh                         ; Set LCR
   out dx,al                           ; Set setting

   mov dx,3F8h                         ; COM1 number
   mov al,00h                          ; No interrupt
   out dx,al
   mov al,0Ch                          ; Set frequency
   mov dx,3F9h                         ; setting data
   out dx,al                           ; transfer frequency

; set in Modem Control Register required bytes
; RTS, DTR, 3 bit
   mov dx,3FCh ;
   mov al,00001011b                    ; set frequency
   out dx,al

   mov dx,3F9h                         ; Settings data
   mov al,0                            ;
   out dx,al                           ; Set
   ret
Init_COM1 endp


;################################################################################
; TEST FIRST PORT
;################################################################################
IsWrite_COM1 proc
   xor al,al
   mov dx,3FDh                         ; portout function
   in al,dx
   test al,10h                         ; check 5 = 1 setted
```

```
   jnz NoWRite                          ; If we cant write ... @099
   ret
IsWrite_COM1 endp

;Function-support
NoWRite proc
  mov ah,9
  mov dx,offset Error_Write             ;@099 ... - also show error message!
  int 21h
  ret
NoWRite endp

;################################################################################
; TEST SECOND PORT
;################################################################################
IsRead_COM2 proc
   xor al,al
   mov dx,3FDh
   in al,dx
   test al,10b                          ; check bit setted
   jnz NoRead                           ; bit = 0, some error in port
   ret
IsRead_COM2 endp

NoRead proc
  mov ah,9
  mov dx,offset Error_Read
  int 21h
  ret
NoRead endp

;################################################################################
; LOAD BYTE TO THE FIRST PORT
;################################################################################
Send_Byte proc
   mov dx,3F8h
   mov al,Data_Byte
   out dx,al                            ; load data in COM1-port
   ret
Send_Byte endp

;################################################################################
; READ BYTE FROM SECOND PORT
;################################################################################
Read_Byte proc
   mov dx,3F8h
   in al,dx
```

```
   mov Data_Byte2,al                    ; load byte from COM1-port
   ret
Read_Byte endp

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

;#########################################################################################
; MAIN FUNCTION
;#########################################################################################
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
```

## 4. Тестирование программ

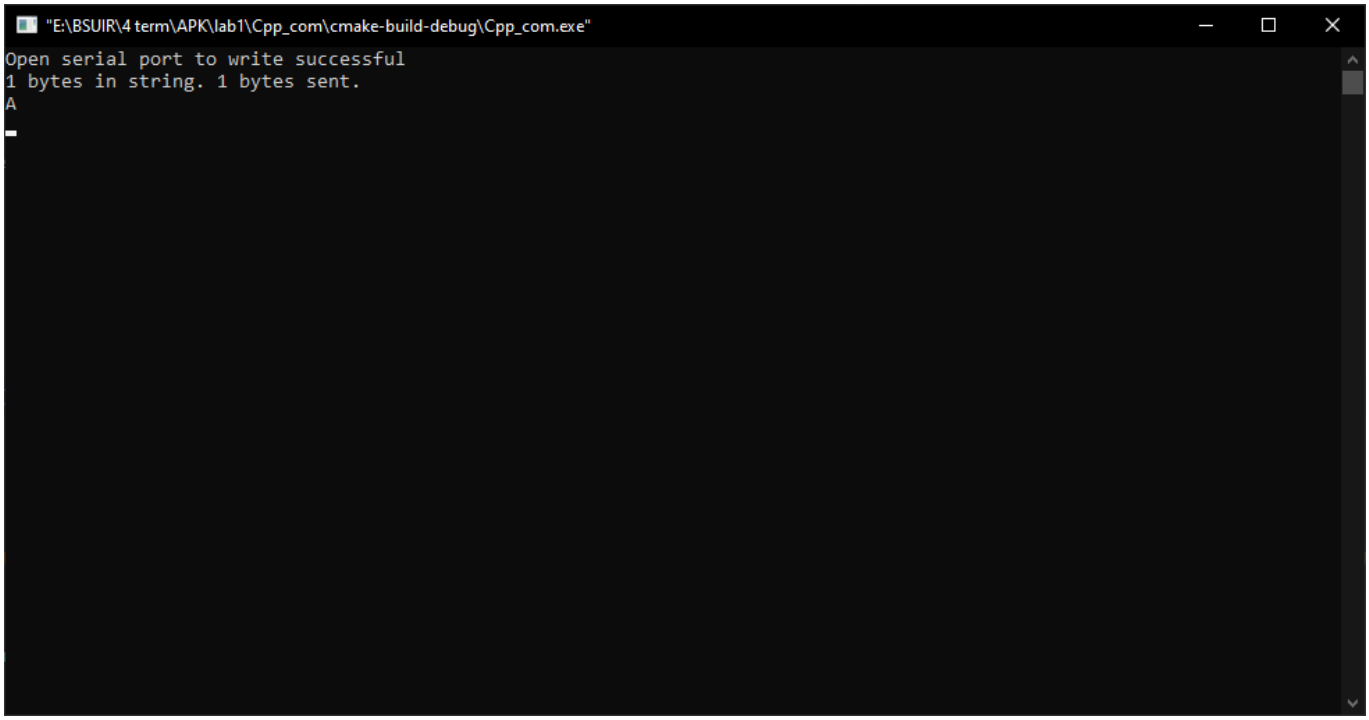


Рисунок 4.1 – Результат работы программы, взаимодействующей с портами ввода-вывода, при включенной эмуляции COM-портов.

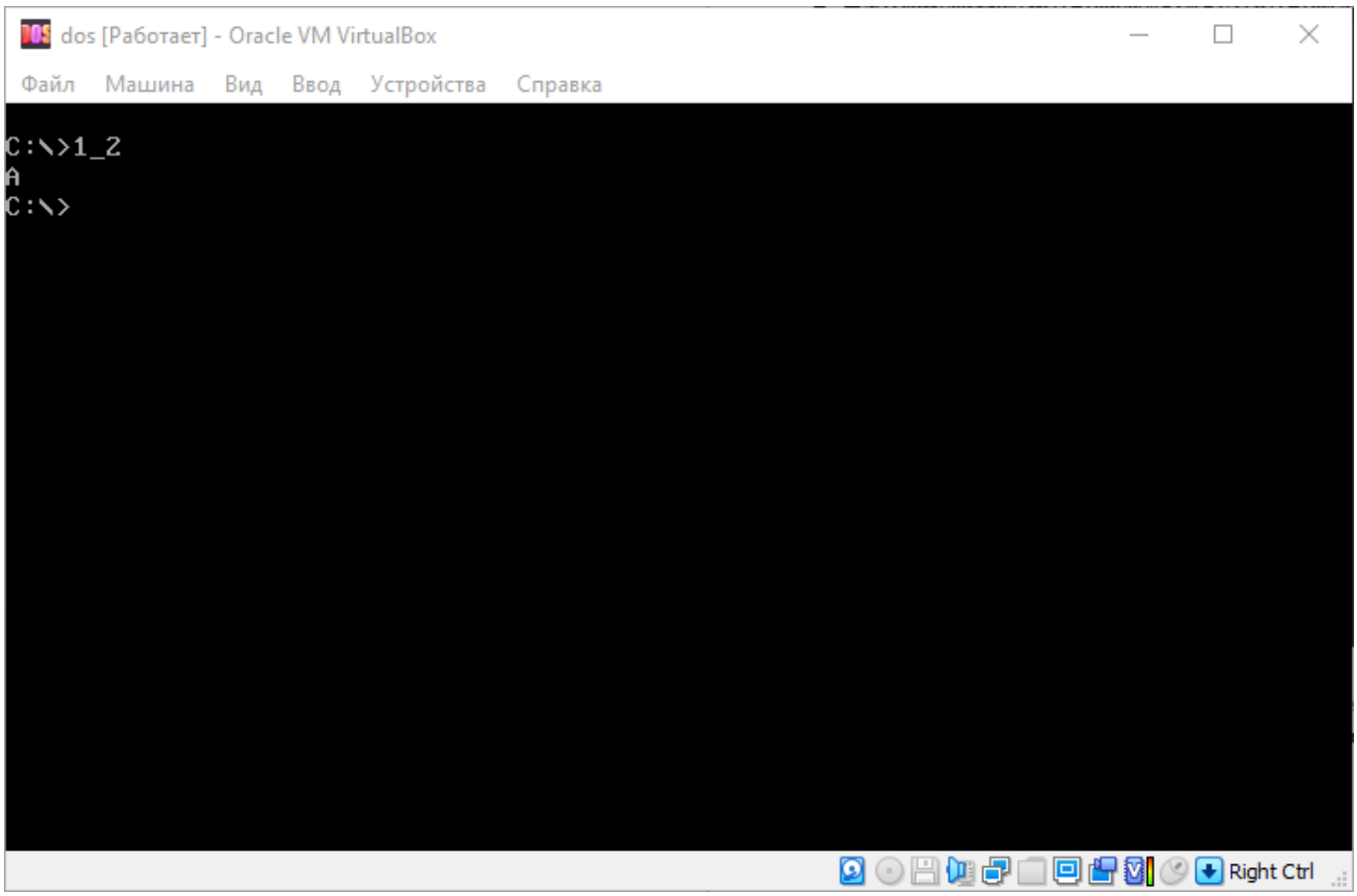


Рисунок 4.2 – Результат работы программы, использующей BIOS прерывание 14h.

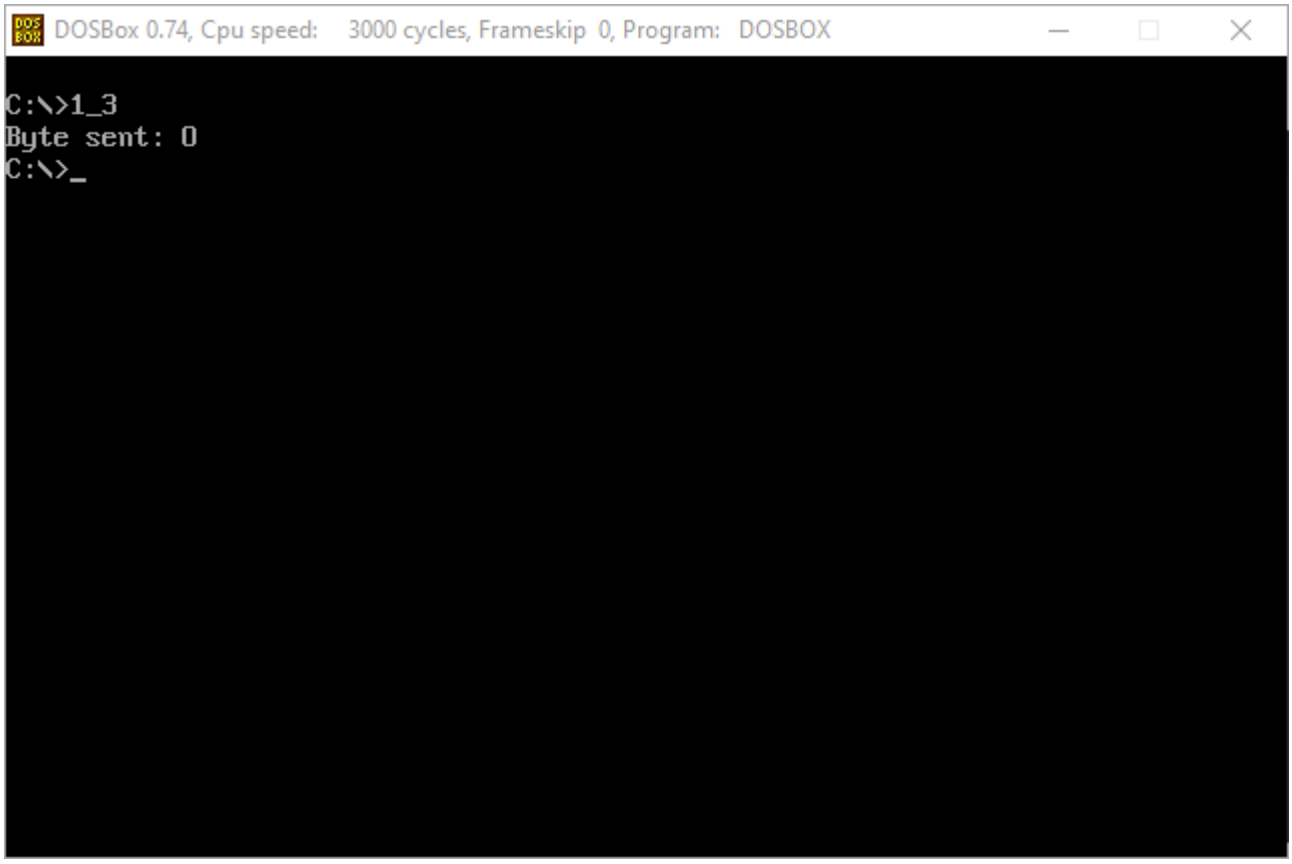


Рисунок 4.3 – Результат работы программы, работающей с COM-портами через регистры как с устройствами ввода-вывода.

## 5. Заключение

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

Для эмуляции COM портов использовался Virtual Serial Port Driver, для эмуляции DOS используется Oracle Virtual Box на хосте 64-х разрядной Windows 12.