Министерство образования Республики Беларусь

Учреждение образования Белорусский государственный университет информатики и радиоэлектроники

Кафедра ЭВМ

Отчёт по лабораторной работе №1

“Последовательный порт”

Проверил: Выполнил:

к.т.н., доцент студент гр. 250502

Одинец Д.Н. Потейчук О.В.

Минск 2024

**Задача**

Целью данной работы является разработка программного модуля 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 прерываний 14h
3. Работа с COM-портом через регистры как с устройством ввода-вывода

**Листинг программы**

1. **Программирование СОМ-порта с помощью работы с ним как с файлом.**

#include <stdio.h>  
#include <windows.h>  
#include <io.h>  
  
char\* filename = "COM6";  
int main() {  
  
 FILE \*file\_pointer;  
 file\_pointer = fopen(filename, "r");  
  
 if (file\_pointer == NULL) {  
 printf("not opened\n");  
 return 1;  
 }  
  
 char word[101], \*it = word;  
 int retVal, size;  
  
 while (1)  
 {  
 retVal = \_read(\_fileno(file\_pointer), it, 1);  
 it += sizeof(char);  
  
 if (retVal < 1 || \*(it - 1) == '\n')  
 break;  
 }  
 if (retVal >= 0)  
 {  
 size = it - word;  
 for (int i = 0; i < size; i++)  
 printf("%c", word[i]);  
 }  
 else  
 printf("error\n");  
  
 fclose(file\_pointer);  
 return 0;  
}

1. **Программирование СОМ-порта с помощью функций BIOS.**

**2.1. Программа приёмник**

COM1 equ 03F8h

COM2 equ 02F8h

PORT\_NUM equ 0 ;0 means COM1, 1 means COM2 etc.

data segment

messageSuccess db "Success!", 0ah, 0dh, "$"

receiveStarted db "Start", 0ah, 0dh, "$"

errorMessage db "Error occured!", 0ah, 0dh, "$"

\n db 0ah, 0dh, "$"

ends

stack segment

dw 128 dup(0)

ends

;=== portErrorCheck ===============================================================================

;ïðîâåðêà íà îøèáêè â om-ïîðòå. ïðîèñõîäèò ïîáèòîâûé and è âûêèäûâàåòñÿ îøèáêà ïðè îøèáêàõ

portErrorCheck macro portNum

push dx

push ax

mov ah, 03h

mov dx, portNum

int 14h

and ah, portErrorBitMask

cmp ah, 0

jne errorDetected

push ax

pop dx

endm

;=== getBit =========================================================================================

;ïîëó÷åíèå áèòà èç áàéòà al

getBit macro bitNum

push bx

mov bl, 01h

shl bl, bitNum

and al, bl

shr al, bitNum

pop bx

endm

;=== print ======================================================================================================

print macro message

push ax

push dx

lea dx, message

mov ax, 9h

int 21h

pop dx

pop ax

endm

code segment

; ==== printByte ==================================================================================================

;âûâîä áàéòà â äâîè÷íîì âèäå

printByte proc

push ax

push bx

push cx

push dx

mov bh, ah

mov cx, 8

mov ah, 02h

printByteLoop:

shl bh, 1

jnc printByteZero

mov dl, "1"

jmp printBytePrint

printByteZero:

mov dl, "0"

printBytePrint:

int 21h

loop printByteLoop:

print \n

pop dx

pop cx

pop bx

pop ax

RET

printByte endp

;====== main ==================================================================================================================

main:

mov ax, data

mov ds, ax

xor ax, ax

mov dx, PORT\_NUM

mov al, 00000011b

mov ah, 00h

int 14h

print receiveStarted

portErrorCheck PORT\_NUM

;===============================

;öèêë ïîëó÷åíèÿ èíôîðìàöèè. ïðè äîëëàðå öèêë çàêàí÷èâàåòñÿ

reciveLoop:

portErrorCheck PORT\_NUM

mov ah, 02h

int 14h

call printByte

mov bx, ax

mov al, ah

getBit 7

cmp al, 1

je reciveLoop

mov ax, bx

cmp al, "$"

je endRecive

mov ah, 02h

mov dl, al

int 21h

print \n

jmp reciveLoop

mov ah, 03h

mov dx, PORT\_NUM

int 14h

call printByte

jmp reciveLoop

endRecive:

print \n

print messageSuccess

jmp exit

errorDetected:

call printByte

print errorMessage

exit:

mov ax, 4c00h

int 21h

ends

end main

**2.2. Программа передатчик**

COM1 equ 03F8h

COM2 equ 02F8h

PORT\_NUM equ 0

portErrorBitMask equ 10001110b

data segment

messageSuccess db "Success!", 0ah, 0dh, "$"

errorMessage db "Error occured!", 0ah, 0dh, "$"

\n db 0ah, 0dh, "$"

ends

stack segment

dw 128 dup(0)

ends

;=== print ===================================================================================

print macro message

push ax

push dx

lea dx, message

mov ax, 9h

int 21h

pop dx

pop ax

endm

;=== portErrorCheck ===============================================================================

;ïðîâåðêà íà îøèáêè â om-ïîðòå. ïðîèñõîäèò ïîáèòîâûé and è âûêèäûâàåòñÿ îøèáêà ïðè îøèáêàõ

portErrorCheck macro portNum

push dx

push ax

mov ah, 03h

mov dx, portNum

int 14h

and ah, portErrorBitMask

cmp ah, 0

jne ErrorDetected

push ax

pop dx

endm

;=== getBit =========================================================================================

;ïîëó÷åíèå áèòà èç áàéòà al

getBit macro bitNum

push bx

mov bl, 01h

shl bl, bitNum

and al, bl

shr al, bitNum

pop bx

endm

code segment

; ==== printByte ==================================================================================================

;âûâîä áàéòà â äâîè÷íîì âèäå

printByte proc

push ax

push bx

push cx

push dx

mov bh, ah

mov cx, 8

mov ah, 02h

printByteLoop:

shl bh, 1

jnc printByteZero

mov dl, "1"

jmp printBytePrint

printByteZero:

mov dl, "0"

printBytePrint:

int 21h

loop printByteLoop:

print \n

pop dx

pop cx

pop bx

pop ax

RET

printByte endp

;====== main =============================================================================================

main:

mov ax, data

mov ds, ax

xor ax, ax

mov dx, PORT\_NUM

mov al, 00000011b

mov ah, 00h

int 14h

portErrorCheck PORT\_NUM

;===============================

;öèêë ïåðåäà÷è èíôîðìàöèè. ïðè äîëëàðå öèêë çàêàí÷èâàåòñÿ

transmitLoop:

portErrorCheck PORT\_NUM

mov ah, 03h

mov dx, PORT\_NUM

int 14h

call printByte

mov al, ah

getBit 5

cmp al, 0

je transmitLoop

mov ah, 01h

int 21h

print \n

mov ah, 01h

int 14h

cmp al, "$"

je endTransmit

jmp transmitLoop

endTransmit:

print \n

print messageSuccess

jmp exit

errorDetected:

print \n

call printByte

print errorMessage

exit:

mov ax, 4c00h

int 21h

ends

end main

1. **Через регистры как с устройством ввода-вывода.**

.model small

.stack 100h

.data

message db 'A'

received db 0

\n db 0Ah, 0Dh, '$'

messageSuccess db "Success!", 0ah, 0dh, "$"

recivedStr db "Recived: $"

buf db 100 dup ?

.code

COM1 equ 03F8h

COM2 equ 02F8h

PORT\_NUM equ COM1

;=== getBit =========================================================================================

;ïîëó÷åíèå áèòà èç áàéòà al

getBit macro bitNum

push bx

mov bl, 01h

shl bl, bitNum

and al, bl

shr al, bitNum

pop bx

endm

;=== print ===================================================================================

print macro message

push ax

push dx

lea dx, message

mov ax, 9h

int 21h

pop dx

pop ax

endm

;=========== comSetup =======================================================================================

;Ýòîò ìàêðîñ ñëóæèò äëÿ óñòàíîâêè ïàðàìåòðîâ ïîñëåäîâàòåëüíîãî ïîðòà,

; òàêèõ êàê ñêîðîñòü ïåðåäà÷è äàííûõ, ðåæèì ðàáîòû è äðóãèå õàðàêòåðèñòèêè

comSetup macro toSetup

push ax

push dx

mov dx, toSetup + 03h; LCR ine control register

mov al, 11011011b; DLAB = 1 óñòàíîâëåíèå äåëèòåëÿ ñêîðîñòè ïåðåäà÷è äàííûõ

out dx, al

mov dx, toSetup + 00h; DLL (DLAB = 1) óñòàíîâêà ìëàäøåãî áàéòà äåëèòåëÿ ñêîðîñòè

mov al, 00h

out dx, al

mov dx, toSetup + 01h; DIM (DLAB = 1) ïîçâîëÿåò ïðîãðàììå óñòàíàâëèâàòü äåëèòåëü ñêîðîñòè

mov al, 01h

out dx, al

mov dx, toSetup + 03h; LCR ïðîãðàììå ðàçðåøàåòñÿ äîñòóï ê äàííûì, à íå ê äåëèòåëþ ñêîðîñòè

mov al, 01011011b; DLAB = 0

out dx, al

mov dx, toSetup + 01h; IER (DLAB = 0)

mov al, 00000000b; îòêëþ÷åíèå âñåõ ïðåðûâàíèé êîì-ïîðòà

out dx, al

mov dx, toSetup + 02h; FCR

mov al, 00000111b; ñáðîñ áóôåðîâ fifo

out dx, al

mov dx, toSetup + 04h; MCR

mov al, 00010000b; LOOP = 0

out dx, al

pop dx

pop ax

endm

; ==== printByte ==================================================================================================

;âûâîä áàéòà â äâîè÷íîì âèäå

printByte proc

push ax

push bx

push cx

push dx

mov bh, ah

mov cx, 8

mov ah, 02h

printByteLoop:

shl bh, 1

jnc printByteZero

mov dl, "1"

jmp printBytePrint

printByteZero:

mov dl, "0"

printBytePrint:

int 21h

loop printByteLoop:

print \n

pop dx

pop cx

pop bx

pop ax

RET

printByte endp

;=== main ==============================================================================================================

main:

mov ax, data

mov ds, ax

comSetup PORT\_NUM

;===================================================

;Ýòîò êîä âûïîëíÿåò ïåðåäà÷ó äàííûõ ÷åðåç COM-ïîðò

transmitCycle:

mov dx, PORT\_NUM + 05h ; LSR

in al, dx

mov ah, al

getBit 5

cmp al, 1

je transmitByte

jmp transmitCycle

transmitByte:

call printByte

mov ah, 01h

int 21h

print \n

cmp al, "$"

je endTransmit

mov dx, PORT\_NUM + 00h ; RBR (DLAB = 0);ïåðåìåùàåò ïîðò ââîäà-âûâîäà com-ïîðòà

out dx, al

jmp transmitCycle

endTransmit:

reciveCycle:

mov ah, 01h

int 21h

print \n

cmp al, '$'

je reciveEnd

mov dx, PORT\_NUM + 05h ; LSR

in al, dx

mov ah, al

call printByte

getBit 0

cmp al, 1

je reciveByte

jmp reciveCycle

reciveByte:

mov dx, PORT\_NUM + 00h ; THR (DLAB = 0)

in al, dx

cmp al, '$'

je reciveEnd

print recivedStr

mov dl, al

mov ah, 02h

int 21h

print \n

jmp reciveCycle

reciveEnd:

print \n

print messageSuccess

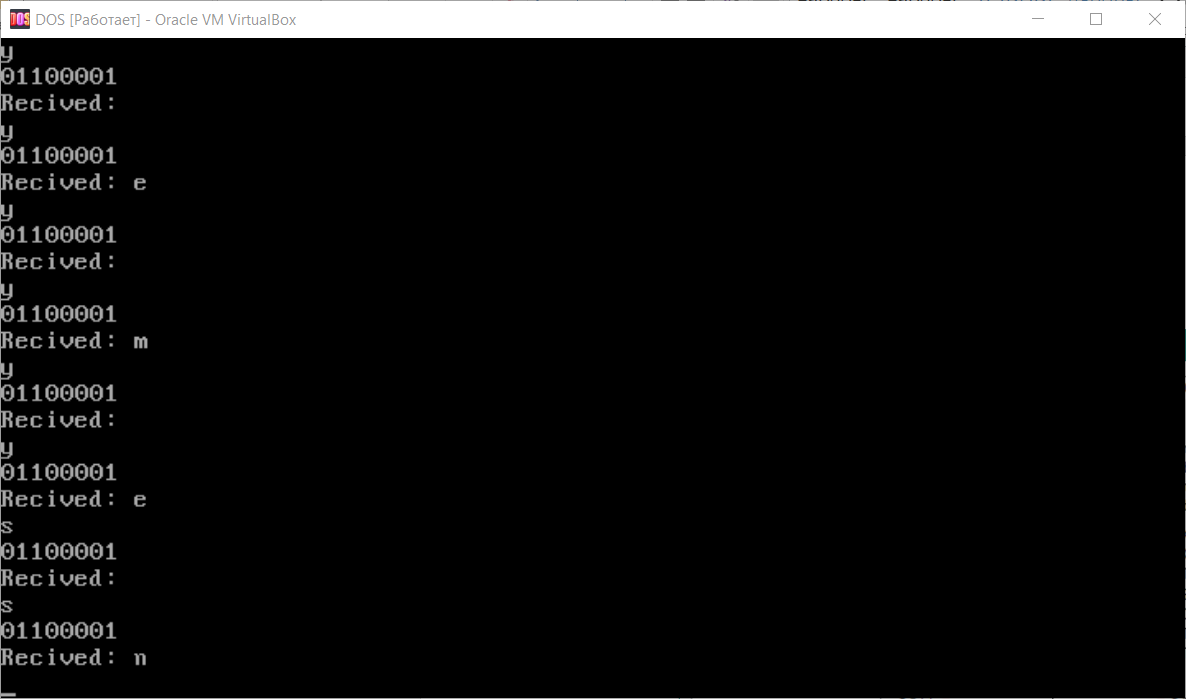
mov ah, 4Ch

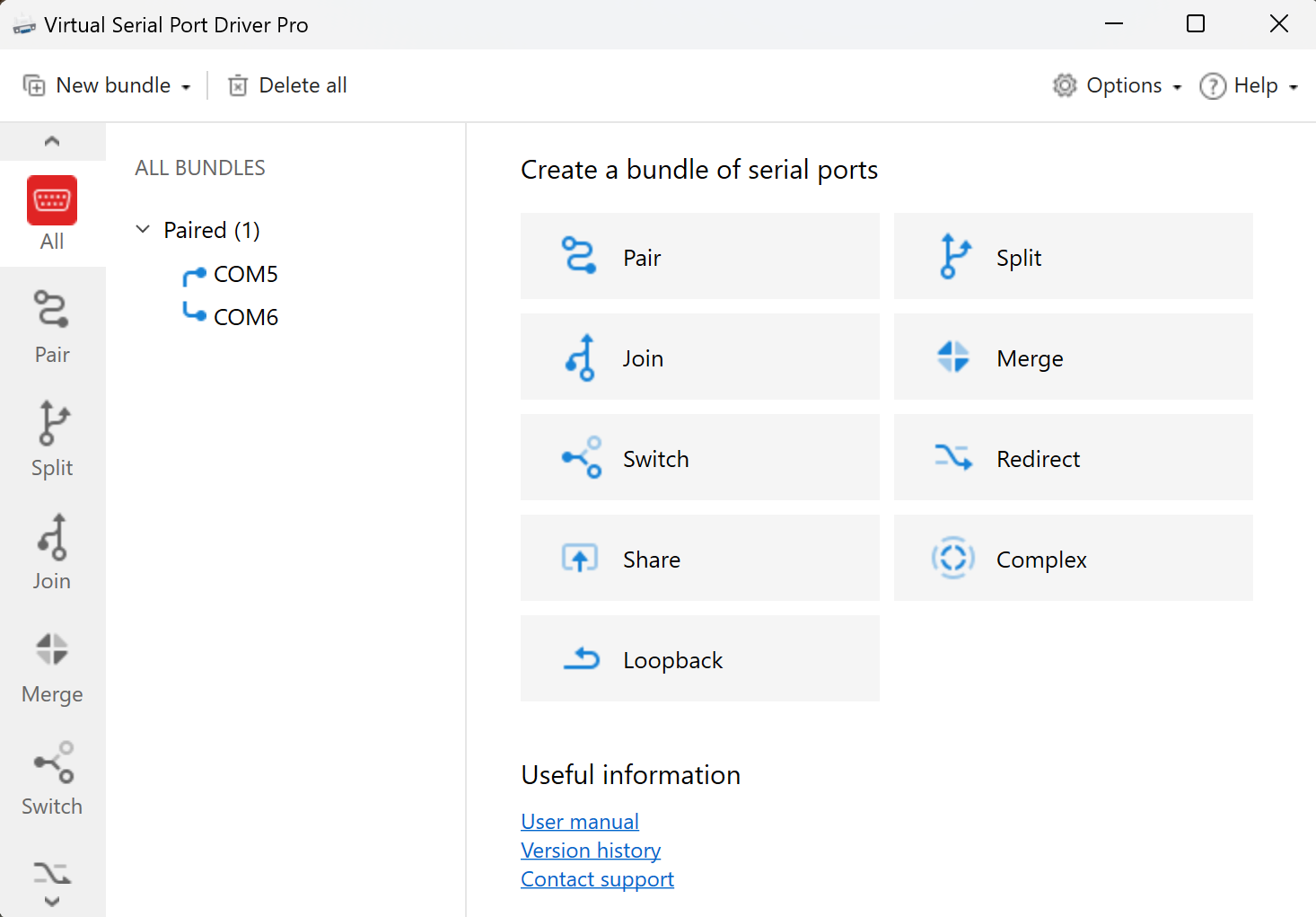
mov al, 0

int 21h

end main

**Пример работы**





**Заключение**

В данной лабораторной работе разработана программа, которая позволяет записывать в данные в один COM-порт и читать их из другого COM-порта (в данном случае это пара COM5 – COM6).