УО БГУИР

Кафедра ЭВМ

Отчет по лабораторной работе

Тема: «Последовательный порт»

Выполнил:

студент группы 650503 Юревич А. С.

Проверил:

к.т.н., доцент Одинец Д. Н.

Минск 2018

**1). Постановка задачи**

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

**2). Листинг программы**

Main.cpp

#define \_CRT\_SECURE\_NO\_WARNINGS

#include "Port.h"

#include <locale.h>

int Server(char \*path);

void Client();

int main(int argc, char\* argv[])

{

setlocale(LC\_ALL, "Russian");

switch (argc)

{

case 1:

Server(argv[0]);

break;

default:

Client();

break;

}

return 0;

}

int Server(char \*path) {

setlocale(LC\_ALL, "Russian");

HANDLE readEnd = CreateEvent(NULL, FALSE, FALSE, "readEnd");

HANDLE writeEnd = CreateEvent(NULL, FALSE, FALSE, "writeEnd");

HANDLE hExit = CreateEvent(NULL, FALSE, FALSE, "exit");

HANDLE hWork = CreateEvent(NULL, FALSE, FALSE, "work");

HANDLE hFile = CreateFile("COM1", GENERIC\_WRITE | GENERIC\_READ, 0, NULL, OPEN\_EXISTING, 0, NULL);

if (hFile == NULL) {

printf("Error\n");

exit(GetLastError());

}

STARTUPINFO si = {};

si.cb = sizeof si;

PROCESS\_INFORMATION pi = {};

if (!CreateProcess(path, (LPSTR)" COM2", NULL, NULL, FALSE, CREATE\_NEW\_CONSOLE, NULL, NULL, &si, &pi)) {

printf("Error while creating process!\n");

return 1;

}

puts("=== COM1 - порт ===");

PORT1(readEnd, writeEnd, hFile, hExit, hWork);

WaitForSingleObject(pi.hProcess, INFINITE);

CloseHandle(pi.hThread);

CloseHandle(pi.hProcess);

CloseHandle(hFile);

}

void Client()

{

setlocale(LC\_ALL, "Russian");

HANDLE hFile = CreateFile("COM2", GENERIC\_READ | GENERIC\_WRITE, 0, NULL, OPEN\_EXISTING, 0, NULL);

if (hFile == NULL) {

printf("Error\n");

exit(GetLastError());

}

HANDLE readEnd = OpenEvent(EVENT\_ALL\_ACCESS, FALSE, "readEnd");

HANDLE writeEnd = OpenEvent(EVENT\_ALL\_ACCESS, FALSE, "writeEnd");

HANDLE hExit = OpenEvent(EVENT\_ALL\_ACCESS, FALSE, "exit");

HANDLE hWork = OpenEvent(EVENT\_ALL\_ACCESS, FALSE, "work");

puts("=== COM2 - порт ===");

PORT2(readEnd, writeEnd, hFile, hExit, hWork);

CloseHandle(hFile);

}

Port.cpp

#define \_CRT\_SECURE\_NO\_WARNINGS

#include "Port.h"

int getch\_()

{

return \_kbhit() ? \_getch() : -1;

}

void PORT1(HANDLE readEnd, HANDLE writeEnd, HANDLE hFile, HANDLE hExit, HANDLE hWork)

{

char buffer[100];

int size;

while (true)

{

if (WaitForSingleObject(hWork, 1) != WAIT\_TIMEOUT)

{

puts("Ожидание. . .");

WaitForSingleObject(writeEnd, INFINITE);

size = read(hFile, buffer);

printf("Сообщение из порта COM2: ");

for (int i = 0; i < size; i++) printf("%c", buffer[i]);

printf("\n");

SetEvent(readEnd);

}

if (getch\_() == 13)

{

SetEvent(hWork);

printf("Сообщение: ");

if (!write(hFile, buffer))

{

SetEvent(hExit);

SetEvent(writeEnd);

return;

}

else

{

SetEvent(writeEnd);

WaitForSingleObject(readEnd, INFINITE);

puts("Доставлено!");

}

}

}

}

void PORT2(HANDLE readEnd, HANDLE writeEnd, HANDLE hFile, HANDLE hExit, HANDLE hWork)

{

char buffer[100];

int size;

while (WaitForSingleObject(hExit, 1) == WAIT\_TIMEOUT)

{

if (WaitForSingleObject(hWork, 1) != WAIT\_TIMEOUT) //if works

{

puts("Ожидание. . .");

WaitForSingleObject(writeEnd, INFINITE);

if (WaitForSingleObject(hExit, 1) != WAIT\_TIMEOUT) return;

size = read(hFile, buffer);

printf("Сообщение из порта COM1: ");

for (int i = 0; i < size; i++) printf("%c", buffer[i]);

printf("\n");

SetEvent(readEnd);

}

if (getch\_() == 13)

{

SetEvent(hWork);

printf("Сообщение: ");

write(hFile, buffer);

SetEvent(writeEnd);

WaitForSingleObject(readEnd, INFINITE);

puts("Доставлено!");

}

}

}

int read(HANDLE hFile, char\* buffer)

{

int size;

DWORD numberOfBytesRead;

ReadFile(hFile, &size, 1 \* sizeof(int), &numberOfBytesRead, NULL); //read size

ReadFile(hFile, buffer, size \* sizeof(char), &numberOfBytesRead, NULL); //read info

return size;

}

bool write(HANDLE hFile, char\* buffer)

{

char symb;

int i = 0;

DWORD numberOfBytesWrite;

while (true)

{

scanf("%c", &symb);

if (symb == '\n')

{

buffer[i] = '\0';

WriteFile(hFile, &i, 1 \* sizeof(int), &numberOfBytesWrite, NULL); //write i(size)

WriteFile(hFile, buffer, i \* sizeof(char), &numberOfBytesWrite, NULL); //write info

if (!strcmp(buffer, "Выход\0")) {

return false;

}

return true;

}

else {

buffer[i] = symb;

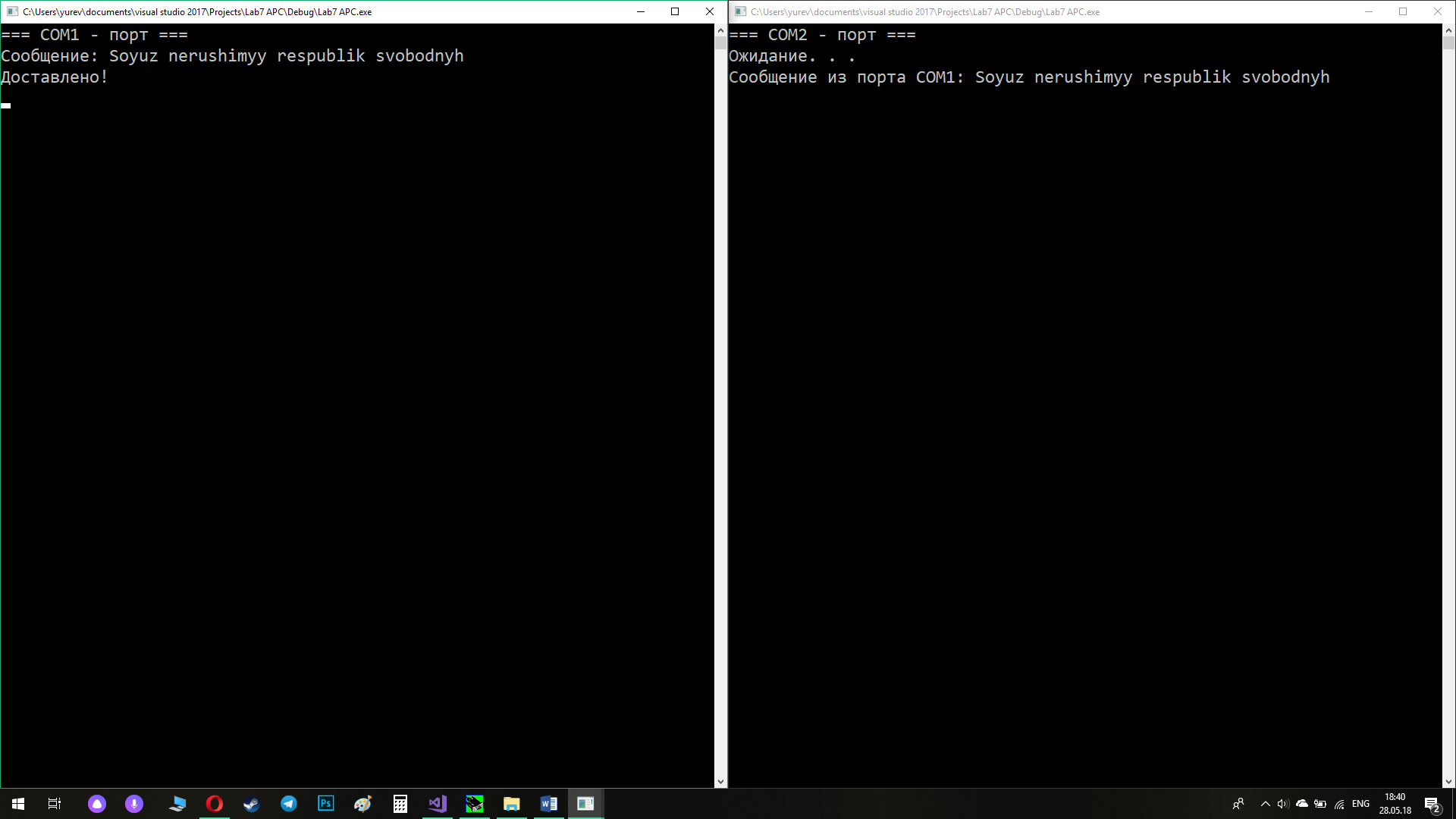
}

i++;

}

}

**3). Тестовые пример**



**4). Заключение**

В ходе выполнения лабораторной работы был разработан программный модуль реализующий передачу данных посредством последовательного интерфейса.