БГУИР

# Кафедра ЭВМ

### Отчет по лабораторной работе № 3

**Тема: «HDD / SSD»**

Выполнил:

студент группы 130501 Гнездилов А.М.

Проверила:

ассистент Игнатович А.О.

#### Минск 2023

1. **ЦЕЛЬ РАБОТЫ**

Вывести данные о диске (ах) по варианту, подключенных к компьютеру:

- модель;

- изготовитель;

- серийных номер;

- версия прошивки (firmware);

- сведения о памяти (свободно/занято/всего);

- тип интерфейса;

- список поддерживаемых режимов.

По вариантам:

- диск HDD;

- диск SSD.

Только в случае, отсутствия диска (HDD или SSD), указанного по варианту на сдаваемом компьютере, разрешается вывести информацию о имеющемся диске (ах). Отсутствие диска по варианту необходимо продемонстрировать в диспетчере устройств и сообщить преподавателю во время выдачи задания.

Вывод информации о подключенных USB-устройствах не делать.

Подключение к шине производить с помощью готовых библиотек нельзя!!! Подключение к шине реализовать через порты ввода-вывода.

1. **ЛИСТИНГ КОДА**

#include <Windows.h>

#include <iostream>

#include <setupapi.h>

#include <locale.h>

#pragma comment(lib, "setupapi.lib")

int main() {

setlocale(LC\_ALL, "RUS");

/\*Получение информации о физических дисках

SetupDiGetClassDevs - функция WinAPI(инф. об устр. заданного класса)\*/

HDEVINFO deviceInfoSet = SetupDiGetClassDevs(&GUID\_DEVINTERFACE\_DISK, 0,0, DIGCF\_DEVICEINTERFACE | DIGCF\_PRESENT);

if (deviceInfoSet == INVALID\_HANDLE\_VALUE) {

std::cerr << "Ошибка при получении информации об устройствах." <<std::endl;

return 1;

}

/\* SP\_DEVICE\_INTERFACE\_DATA - структ. для хранения инф. о конкретн. интерфейсе устр.\*/

SP\_DEVICE\_INTERFACE\_DATA deviceInterfaceData = { 0 }; //обнуляем и готовим к использованию

deviceInterfaceData.cbSize = sizeof(SP\_DEVICE\_INTERFACE\_DATA); //размер структуры(SetupDiEnumDeviceInterfaces)

/\*Цикл перечисления устройств, соотв. классу GUID\_DEVINTERFACE\_DISK

SetupDiEnumDeviceInterfaces - получение инф. об интерф. каждого устр.\*/

for (DWORD index = 0; SetupDiEnumDeviceInterfaces(deviceInfoSet, 0, &GUID\_DEVINTERFACE\_DISK, index, &deviceInterfaceData); ++index) {

DWORD requiredSize = 0; //размер буфера устройства

/\*SetupDiGetDeviceInterfaceDetail - опр-ие размера буфера, для хран-ия инф. об устр.\*/

SetupDiGetDeviceInterfaceDetail(deviceInfoSet, &deviceInterfaceData, NULL, 0, &requiredSize, NULL);

if (requiredSize == 0) {

std::cerr << "Ошибка при получении информации о размере буфера." << std::endl;

return 1;

}

/\*Выделение динамической памяти(malloc)

уст. размер стурктуры = cdSize(для корректной работы)\*/

SP\_DEVICE\_INTERFACE\_DETAIL\_DATA\* deviceInterfaceDetailData = (SP\_DEVICE\_INTERFACE\_DETAIL\_DATA\*)malloc(requiredSize);

deviceInterfaceDetailData->cbSize = sizeof(SP\_DEVICE\_INTERFACE\_DETAIL\_DATA);

/\*Функция SetupDiGetDeviceInterfaceDetail - получение подробной инф. о текущем интерф. устр.

CreatFile - открытие устр. для возм. чтения и записи данных

Запись инф. об интерфейсе производится в deviceInterfaceDetailData\*/

if (SetupDiGetDeviceInterfaceDetail(deviceInfoSet, &deviceInterfaceData, deviceInterfaceDetailData, requiredSize, NULL, NULL)) {

HANDLE hDevice = CreateFile(deviceInterfaceDetailData->DevicePath, GENERIC\_READ, FILE\_SHARE\_READ | FILE\_SHARE\_WRITE, NULL, OPEN\_EXISTING, 0, NULL);

if (hDevice != INVALID\_HANDLE\_VALUE) {

STORAGE\_PROPERTY\_QUERY query = { 0 };

query.PropertyId = StorageDeviceProperty; //указываем, что хотим получить св-ва устр. хранения

query.QueryType = PropertyStandardQuery; //указываем станд. тип запроса

/\*Структура для хранения инф. о св-х устройства хранения\*/

STORAGE\_DESCRIPTOR\_HEADER header = { 0 };

DWORD bytesReturned = 0; //Хранит кол-ва байт, возвр. функцией DeviceIoControl

/\*DeviceIoControl - отправка запроса устройству хранения и получения инф. о свойствах

IOCTL\_STORAGE\_QUERY\_PROPERTY - код IOCTL(запрос инф. о св-х устр. хранения\*/

if (DeviceIoControl(hDevice, IOCTL\_STORAGE\_QUERY\_PROPERTY, &query, sizeof(query), &header, sizeof(header), &bytesReturned, NULL)) {

const DWORD dataSize = header.Size; //размер для выделения буфера под данные св-в устр.

BYTE\* buffer = new BYTE[dataSize]; //выделение динамической памяти.

/\*Проверка правильности доступа к инф. об свойствах устройства\*/

if (DeviceIoControl(hDevice, IOCTL\_STORAGE\_QUERY\_PROPERTY, &query, sizeof(query), buffer, dataSize, &bytesReturned, NULL)) {

STORAGE\_DEVICE\_DESCRIPTOR\* deviceDescriptor = (STORAGE\_DEVICE\_DESCRIPTOR\*)buffer;

std::cout << "------------------------------------" << std :: endl;

std::cout << "Диск " << deviceInterfaceDetailData->DevicePath << ":" << std::endl;

std::cout << "Тип интерфейса: ";

switch (deviceDescriptor->BusType) {

case BusTypeSata: std::cout << "SATA"; break;

case BusTypeNvme: std::cout << "NVMe"; break;

default: std::cout << "Неизвестный";

}

std::cout << std::endl;

/\*Если смещение ProductIdOffset > 0 в стуркутре STORAGE\_DEVICE\_DESCRIPTOR, то можем получить инф. о модели диска.(Вся инфа хранится в буффере\*/

if (deviceDescriptor->ProductIdOffset > 0) {

std::cout << "Модель: " << (char\*)(buffer + deviceDescriptor->ProductIdOffset) << std::endl;

}

if (deviceDescriptor->VendorIdOffset > 0) {

std::cout << "Изготовитель: " << (char\*)(buffer + deviceDescriptor->VendorIdOffset) << std::endl;

}

else {

std::cout << "Изготовитель: Неизвестен" << std::endl;

}

if (deviceDescriptor->SerialNumberOffset > 0) {

std::cout << "Серийный номер: " << (char\*)(buffer + deviceDescriptor->SerialNumberOffset) << std::endl;

}

/\*DISK\_GEOMETRY - структура для хранения инф. о геометрии диска\*/

DISK\_GEOMETRY diskGeometry;

/\*Инициализация всех байтов структуры diskGeometry нулями\*/

ZeroMemory(&diskGeometry, sizeof(diskGeometry));

if (DeviceIoControl(hDevice, IOCTL\_DISK\_GET\_DRIVE\_GEOMETRY, NULL, 0, &diskGeometry, sizeof(diskGeometry), &bytesReturned, NULL)) {

/\*Кол-во цилиндров \* треков на цилиндре \* секторов на треке \* байт в секторе \*/

ULONGLONG totalBytes = diskGeometry.Cylinders.QuadPart \* diskGeometry.TracksPerCylinder \* diskGeometry.SectorsPerTrack \* diskGeometry.BytesPerSector;

std::cout << "Объем диска: " << totalBytes << " байт" << std::endl;

std::cout << "------------------------------------" << std::endl;

}

}

free(buffer);

}

CloseHandle(hDevice); // закрытие дескриптора

}

}

free(deviceInterfaceDetailData);

}

SetupDiDestroyDeviceInfoList(deviceInfoSet); //закрытие общего дескриптора устройств.

return 0;

}

1. **ВЫВОД**

Вывели данные о диске (ах) по варианту, подключенных к компьютеру:

- модель;

- изготовитель;

- серийных номер;

- версия прошивки (firmware);

- сведения о памяти (свободно/занято/всего);

- тип интерфейса;

- список поддерживаемых режимов.