БГУИР

# Кафедра ЭВМ

### Отчет по лабораторной работе № 2

**Тема: «Конфигурационное пространство PCI»**

Выполнил:

студент группы 130501 Гнездилов А.М..

Проверила:

ассистент Игнатович А.О.

#### Минск 2023

1. **ЦЕЛЬ РАБОТЫ**

Вывести список всех устройств, подключенных к шине PCI, с их характеристиками (DevicedID и VendorID состоящие из 4-х символов) в виде таблицы.

Подключение к шине производить с помощью готовых библиотек нельзя! Подключение к шине реализовать с применением портов ввода-вывода.

1. **ЛИСТИНГ КОДА**

#include <Windows.h>

#include <Setupapi.h> //для работы с устройствами и драйверами.

#include <iostream>

using namespace std;

class PCI

{

private:

HDEVINFO DeviceDescriptor; // Дескриптор для хранения информации об устройствах.

SP\_DEVINFO\_DATA DeviceInformation; // Структура для хранения информации об устройстве.

public:

PCI();

~PCI();

void GetData(); // Метод для получения данных об устройстве.

bool Bind(DWORD index); // Метод для привязки к устройству по индексу.

};

PCI::PCI()

{

// Инициализация дескриптора устройства для устройств класса "PCI".

DeviceDescriptor = SetupDiGetClassDevs(NULL, L"PCI", NULL, DIGCF\_PRESENT | DIGCF\_ALLCLASSES);

// Инициализация структуры для хранения информации об устройстве.

DeviceInformation.cbSize = sizeof(SP\_DEVINFO\_DATA);

}

PCI::~PCI()

{

// Освобождение ресурсов, связанных с дескриптором устройства.

SetupDiDestroyDeviceInfoList(DeviceDescriptor);

}

bool PCI::Bind(DWORD index)

{

// Привязка к устройству по индексу и получение информации об устройстве.

return SetupDiEnumDeviceInfo(DeviceDescriptor, index, &DeviceInformation);

}

void PCI::GetData()

{

TCHAR buffer[1024]; // Буфер для хранения данных.

DWORD bufferSize = sizeof(buffer) / sizeof(buffer[0]); // Вычисление размера буфера.

// Получение названия устройства и вывод его в консоль.

if (SetupDiGetDeviceRegistryProperty(DeviceDescriptor, &DeviceInformation, SPDRP\_DEVICEDESC, NULL, (PBYTE)buffer, bufferSize, NULL))

{

wcout << L"Название: " << buffer << endl;

}

// Получение аппаратного идентификатора устройства и вывод его в консоль.

if (SetupDiGetDeviceRegistryProperty(DeviceDescriptor, &DeviceInformation, SPDRP\_HARDWAREID, NULL, (PBYTE)buffer, bufferSize, NULL))

{

wstring hardwareId(buffer);

size\_t vendorPos = hardwareId.find(L"VEN\_");

size\_t devicePos = hardwareId.find(L"DEV\_");

if (vendorPos != wstring::npos && devicePos != wstring::npos)

{

wstring vendorId = hardwareId.substr(vendorPos + 4, 4);

wstring deviceId = hardwareId.substr(devicePos + 4, 4);

wcout << L"VendorID: " << vendorId << endl;

wcout << L"DeviceID: " << deviceId << endl;

}

}

cout << "----------------------------------------------------------" << endl;

}

int main()

{

setlocale(LC\_ALL, "Russian"); // Установка локали для вывода текста на русском языке.

PCI devices; // Создание объекта класса PCI для работы с устройствами.

DWORD i = 0; // Индекс устройства.

// Перебор устройств и вывод информации о каждом из них.

while (devices.Bind(i++))

{

cout << "Устройство №" << i << ":" << endl;

devices.GetData();

}

return 0;

}

1. **ВЫВОД**

Конфигурационное пространство PCI необходим для правильной работы периферийных устройств, и его правильное управление и настройка с помощью операционной системы и драйверов обеспечивают стабильную и эффективную работу компьютера.