Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра электронных вычислительных машин

Лабораторная работа №1

«Энергопитание»

Выполнил: Проверил:

Студент группы 150504 Преподаватель

Горбачевский К.В. Селезнев А.И.

Минск, 2023

1. Постановка задачи

#### Вывести информацию об энергопитании компьютера:

#### - в режиме реального времени показывать тип энергопитания;

#### - информация о типе батареи;

#### - уровень заряда батареи (%);

#### - текущий режим энергосбережения;

#### - реализовать переход в спящий режим и гибернацию из консоли.

#### - вывести время работы аккумулятора без подключения к зарядке;

#### - вывести оставшееся время работы аккумулятора после отключения зарядного устройства.

1. Краткие теоретические сведения

Аккумуляторы – это устройства для хранения и обеспечения электроэнергией портативных устройств, в частности ноутбука. Они играют важную роль в обеспечении мобильности девайса, позволяя использовать его без подключения к зарядному устройству.

Основные параметры аккумулятора:

1. Тип аккумулятора. Бывают Литий-ионные (Li-on), Никель-кадмиевые батареи (NiCad), Никель-металл-гидридные (NiMh), литий-полимерные (Li-polymer). Самые популярные - Литий-ионные (Li-on): Они появились на рынке вслед никель-кадмиевыми, что выгодно отличает продукцию. Это компактные детали, которые имеют небольшой вес и большой эксплуатационный срок. Количество циклов зарядки и разрядки в них достигает не менее 300. Но есть и свои недостатки: для них характерен саморазряд, то есть, сокращение заряда даже без использования аппаратуры. Самым прогрессивным, на данный момент, является литий-полимерный (Li-polymer) тип аккумулятора.

2. Емкость (Capacity): Емкость измеряется в миллиампер-часах (mAh) или в ватт-часах (Wh). Чем больше емкость аккумулятора, тем дольше ноутбук может работать автономно.

3. Напряжение (Voltage): Напряжение аккумулятора измеряется в вольтах (V). Ноутбуки обычно используют литий-ионные аккумуляторы с напряжением 3,7 В. Однако в реальности для питания ноутбука используется более высокое напряжение, так как аккумуляторы соединяются в серии.

4. Количество ячеек: Аккумулятор ноутбука – это набор из нескольких ячеек, которые соединяются в серию или параллельно, чтобы обеспечить необходимое напряжение и емкость. Обычно в ноутбуках используются аккумуляторы с 4, 6, 8 или более ячейками.

5. Время работы. Время которое ноутбук может работать без подключения к зарядному устройству. Этот показатель снижается по мере износа батареи или в зависимости от сложности выполняемой работы, числа запущенных программ, но новые ноутбуки могут работать на одной зарядке около 7 часов.

6. Саморазряд (Self-discharge): Время, в течение которого аккумулятор сохраняет свою зарядку при хранении без использования. Разные типы аккумуляторов имеют разное значение данной характеристики.

7. Циклы зарядки и разрядки: Аккумуляторы имеют ограниченное количество циклов зарядки и разрядки. После определенного числа циклов аккумулятор начинает терять емкость. Этот показатель важен при расчете срока службы аккумулятора.

1. Листинг программы

Далее приведен листинг программы, реализующей все поставленные задачи.

#include <iostream>  
#include <windows.h>  
#include <powrprof.h>  
#include <winbase.h>  
#include <batclass.h>  
#include <cstdlib>  
#include <windows.h>  
#include <setupapi.h>  
#include <devguid.h>  
#pragma comment(lib, "setupapi.lib")  
#pragma comment(lib, "Powrprof.lib")  
using namespace std;

void printInfo() {  
    cout << "1 - Показать информацию" << endl;  
    cout << "2 - Перейти в режим сна" << endl;  
    cout << "3 - Перейти в режим гибернации" << endl;  
    cout << "4 - Обновить данные" << endl;  
    cout << "0 - Выход" << endl << endl;  
}

void chargingStatus(int state) {  
    switch (state) {  
        case 0: cout << "Зарядка не подключена" << endl; break;  
        case 1: cout << "Зарядка подключена" << endl; break;  
        case 255: cout << "Нет данных" << endl; break;  
        default: break;  
    }  
}

void battaryProcent(int procent) {  
    if (procent != 255) {  
        cout << "Батерея заряжена на " << procent << "%" << endl;  
    }  
    else cout << "Данные о заряде не найдены" << endl;  
}

void statusOfBattary(int statusBattary) {  
    switch (statusBattary) {  
        case 0: cout << "Режим энергосбережения выключен." << endl; break;  
        case 1: cout << "Режим энергосбережения включен." << endl; break;  
        default: break;  
    }  
}

void displayInformationAboutBatteryType() {  
    HDEVINFO hdev = SetupDiGetClassDevs(  
        &GUID\_DEVCLASS\_BATTERY,  
        0,  
        0,  
        DIGCF\_PRESENT | DIGCF\_DEVICEINTERFACE  
    );  
    if (INVALID\_HANDLE\_VALUE == hdev) {  
        cout << "INVALID\_HANDLE\_VALUE";  
        return;  
    }  
    SP\_DEVICE\_INTERFACE\_DATA did = { 0 };  
    did.cbSize = sizeof(did);  
    bool condition = !SetupDiEnumDeviceInterfaces(hdev,  
        0,  
        &GUID\_DEVCLASS\_BATTERY,  
        0,  
        &did  
    );  
    if(condition) {  
        cout << "unknown error 1";  
        return;  
    }  
    DWORD cbRequired = 0;  
    SetupDiGetDeviceInterfaceDetail(hdev,  
        &did,  
        0,  
        0,  
        &cbRequired,  
        0  
    );  
    if (ERROR\_INSUFFICIENT\_BUFFER != GetLastError()) {  
        cout << "ERROR\_INSUFFICIENT\_BUFFER";  
        SetupDiDestroyDeviceInfoList(hdev);  
        return;  
    }  
    PSP\_DEVICE\_INTERFACE\_DETAIL\_DATA pdidd = (PSP\_DEVICE\_INTERFACE\_DETAIL\_DATA)LocalAlloc(LPTR, cbRequired);  
  
    if (!pdidd) {  
        cout << "unknown error 3";  
        SetupDiDestroyDeviceInfoList(hdev);  
        return;  
    }  
    pdidd->cbSize = sizeof(\*pdidd);  
    condition = !SetupDiGetDeviceInterfaceDetail(hdev,  
        &did,  
        pdidd,  
        cbRequired,  
        &cbRequired,  
        0  
    );  
    if(condition) {  
        cout << "unknown error 4";  
        return;  
    }  
    HANDLE hBattery = CreateFile(pdidd->DevicePath,  
        GENERIC\_READ | GENERIC\_WRITE,  
        FILE\_SHARE\_READ | FILE\_SHARE\_WRITE,  
        NULL,  
        OPEN\_EXISTING,  
        FILE\_ATTRIBUTE\_NORMAL,  
        NULL  
    );  
    if(INVALID\_HANDLE\_VALUE == hBattery) {  
        cout << "ERROR\_INSUFFICIENT\_BUFFER";  
        LocalFree(pdidd);  
        SetupDiDestroyDeviceInfoList(hdev);  
        return;  
    }  
    BATTERY\_QUERY\_INFORMATION bqi = { 0 };  
    DWORD dwWait = 0;  
    DWORD dwOut;  
    if (!DeviceIoControl(hBattery,  
        IOCTL\_BATTERY\_QUERY\_TAG,  
        &dwWait,  
        sizeof(dwWait),  
        &bqi.BatteryTag,  
        sizeof(bqi.BatteryTag),  
        &dwOut,  
        NULL)  
        && bqi.BatteryTag  
    ) {  
        cout << "unknown error 5";  
        CloseHandle(hBattery);  
        LocalFree(pdidd);  
        SetupDiDestroyDeviceInfoList(hdev);  
        return;  
    }  
    BATTERY\_INFORMATION bi = { 0 };  
    bqi.InformationLevel = BatteryInformation;  
    if (!DeviceIoControl(hBattery,  
        IOCTL\_BATTERY\_QUERY\_INFORMATION,  
        &bqi,  
        sizeof(bqi),  
        &bi,  
        sizeof(bi),  
        &dwOut,  
        NULL))  
    {  
        cout << "unknown error 6";  
        CloseHandle(hBattery);  
        LocalFree(pdidd);  
        SetupDiDestroyDeviceInfoList(hdev);  
        return;  
    }  
    cout << "Тип батареи: " << bi.Chemistry << "\n";  
    CloseHandle(hBattery);  
    LocalFree(pdidd);  
    SetupDiDestroyDeviceInfoList(hdev);  
    return;  
}  
  
SYSTEM\_POWER\_STATUS getState(SYSTEM\_POWER\_STATUS& computer) {  
    if (!GetSystemPowerStatus(&computer)) {  
        cout << "truble with getting data from powrprof" << endl;  
        exit(1);  
    }  
    return computer;  
}

void printState(SYSTEM\_POWER\_STATUS computer) {  
    cout << "\n";  
    battaryProcent(computer.BatteryLifePercent);  
    chargingStatus(computer.ACLineStatus);  
    statusOfBattary(computer.SystemStatusFlag);  
    displayInformationAboutBatteryType();  
    if (!computer.ACLineStatus) {  
        cout << "\nОставшееся время работы аккумулятора: " << (double)computer.BatteryLifeTime / 60 / 60 << " часов" << "\n";  
    }  
   
    cout << "Выключенный ноутбук с заряженным аккумулятором продержится: " << (double)computer.BatteryFullLifeTime / 60 / 60 / 24 << " дней"  << "\n\n\n";  
}

int main() {  
    setlocale(LC\_ALL, "Russian");

    SYSTEM\_POWER\_STATUS computer{};  
    computer = getState(computer);

    int work;

    while (true) {  
        printInfo();  
        cout << "\n\nВыберите действие: ";  
        cin >> work;  
        switch (work) {  
            case 1: printState(computer); break;  
            case 2: SetSuspendState(FALSE, FALSE, FALSE); break;  
         
            case 3: SetSuspendState(TRUE, FALSE, FALSE); break;  
            case 4: {  
                cout << "Обновлено" << endl;  
                computer = getState(computer);  
                break;  
            }  
            default: break;  
        }  
        if (work == 0) break;  
    }  
    return 0;  
}

1. Тестирование программы

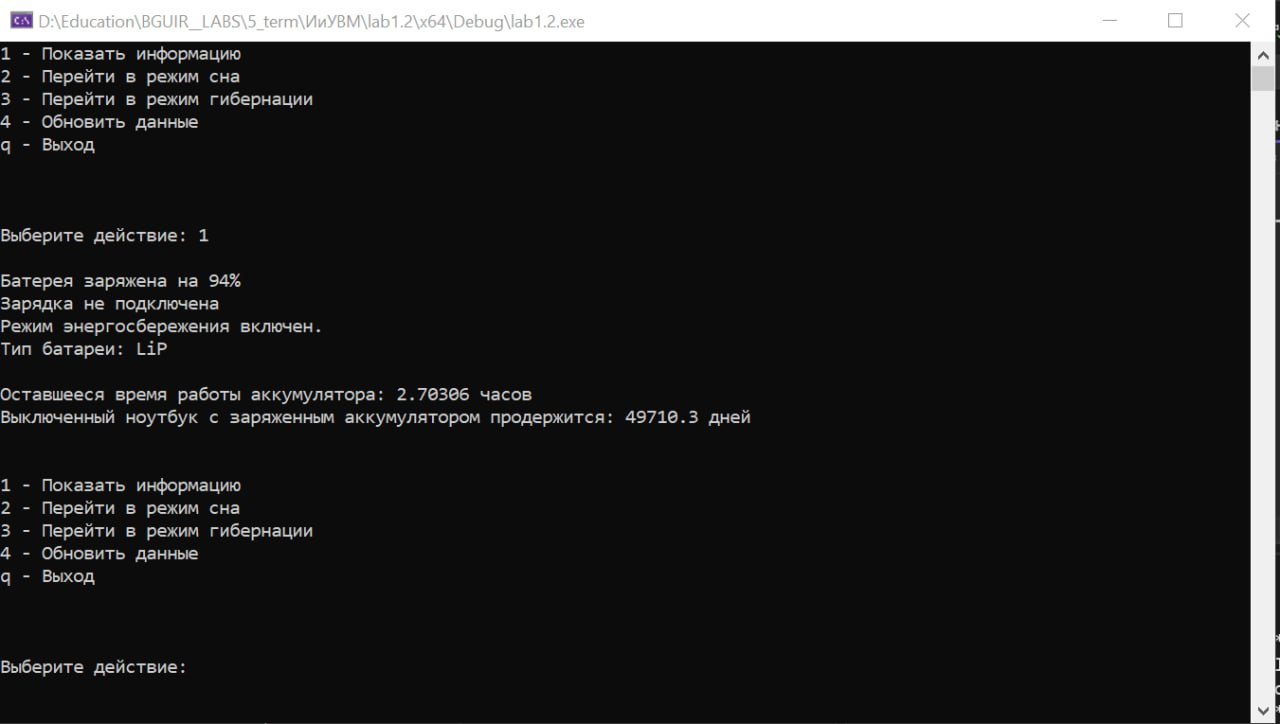


Рисунок 4.1 — Окно вывода программы.

1. Заключение

В данной лабораторной работе были выполнены все поставленные задачи: написана программа, которая считывает информацию о батарее и выводит основные показатели.

Программа написана для операционной системы Windows, использованные инструменты – язык программирования C++, Microsoft Visual Studio.