Министерство образования Республики Беларусь

Учреждение образования Белорусский государственный университет информатики и радиоэлектроники

Кафедра ЭВМ

Отчёт по лабораторной работе №3

“HDD/SSD”

Проверил: Выполнили

Селезнёв Александр Игоревич студенты гр. 030501:

Вайтехович П.В.

Потапов С.А.

Минск 2022

**Цель:** Научиться работать и создавать программы для работы с HDD/SSD.

**Исходные данные:** Visual studio

**Теоритические сведения:**

Подсистема памяти. В общем случае подсистема памяти предназначена для хранения данных, подлежащих обработке центральным процессором (процессорами). В современных системах она является многоуровневой с точки зрения архитектуры, причем каждый уровень имеет свои особенности и характеристики. Замечено, что чем ниже уровень, тем выше емкость, но ниже быстродействие и стоимость.

Самая общая классификация всех устройств памяти – *по отношению к системе*: существует **внутренняя и внешняя память**. Внутренняя непосредственно доступна вычислительным устройствам (процессорам) для выборки как команд, так и операндов к ним (то есть данных). Обычно эта память управляется собственным контроллером, обеспечивающим тот или иной метод доступа (обычно – произвольный, а не последовательный).

Устройства внешней памяти. Относятся к периферийной части системы, доступны через контроллеры, подключены к периферийным шинам.

Предназначены для подгрузки данных в оперативную память, которая не может вместить весь требуемый объем целиком.

Единица адресации значительно больше байта. Чаще всего используется **понятие «сектора»** (по аналогии с сектором диска, поскольку большинство устройств первого поколения имело вращающиеся диски). Типичный объем сектора – 512 байт.

Устройства внешней памяти могут иметь различную конструкцию и варианты исполнения, однако в любом случае они не доступны процессору напрямую, через команды загрузки и сохранения операндов (Load/Store). Системная адресация к ним тоже не применима. **По исполнению:**

* внутренние (внутри корпуса системы, без отдельного питания);
* внешние (в отдельном корпусе).

**По конструкции:**

* со сменными носителями;
* со встроенными движущимися носителями (обычно дисками);
* твердотельные накопители (без движущихся деталей).

**По принципу адресации и доступа:**

* блочные с произвольным доступом;
* блочные с последовательным доступом (чаще всего ленточные);
* потоковые (практически то же, что и последовательного типа).

**По типу использованного физического явления:**

* магнитные (магнитная ориентация ячеек);
* оптические (оптические свойства материалов);
* электронные (хранение электронного заряда в ячейках);
* комбинированные (разные принципы  для чтения и записи).

Характеристики внешней памяти.

*Емкость* (capacity).

Максимальное количество информации, которое может в ней храниться.

Измеряется в байтах. Принята десятичная, а не двоичная система обозначений: K, M, G, T, P.

*Скорость доступа (access time)*

Время от поступления запроса до фактического выполнения операции.

Для чтения и записи, как правило, различаются.

*Время позиционирования (seek time)*

Для механических устройств с произвольным доступом – время перемещения считывающего элемента к требуемой ячейке.

*Скорость чтения/записи* (read/write performance)

зависит от характера поступающих запросов. У ряда устройств эти скорости существенно различаются.

*Удельная стоимость хранения данных*

стоимость в расчете на емкость.

Твердотéльный накопитель (англ. SSD, solid-state drive) — компьютерное немеханическое запоминающее устройство на основе микросхем памяти. Кроме них, SSD содержит управляющий контроллер.

Различают два вида твердотельных накопителей: SSD на основе памяти, подобной оперативной памяти компьютеров, и SSD на основе флеш-памяти.

В настоящее время твердотельные накопители используются в компактных устройствах: ноутбуках, нетбуках, коммуникаторах и смартфонах, но могут быть использованы и в системных блоках для повышения производительности. Некоторые известные производители переключились на выпуск твердотельных накопителей уже полностью, например Samsung продал бизнес по производству жёстких дисков компании Seagate.

Существуют и так называемые гибридные жесткие диски, появившиеся, в том числе, из-за текущей, пропорционально более высокой стоимости твердотельных накопителей. Такие устройства сочетают в одном устройстве накопитель на жёстких магнитных дисках (HDD) и твердотельный накопитель относительно небольшого объёма, в качестве кэша (для увеличения производительности и срока службы устройства, снижения энергопотребления).

Пока такие накопители используются, в основном, в переносных устройствах (ноутбуках, сотовых телефонах, планшетах и т. п.).

Жесткий диск типа «винчестер» – самое массовое устройство внешней памяти с несменными носителями, имеющее внутреннее исполнение.

В исходном варианте он хранит операционную систему, другое системное ПО, код и данные прикладного ПО. HDD является неотъемлемой частью практически любой ЭВМ в стоечном, напольном, настольном или мобильном исполнении. Если он отсутствует физически, то тогда, как правило, эмулируется, т.к. прочно входит в архитектуру всех систем.

С другой стороны, это самое сложное и капризное устройство, к которому применяются самые высокие требования надежности.

Из-за наличия движущихся частей HDD подвержен в наибольшей степени влияниям извне – он чувствителен практически ко всем факторам окружающей среды (как минимум к температуре, влажности, атмосферному давлению, запыленности, вибрациям, ударам, перегрузкам).

**Выполнение работы:**

#include <iomanip>

#include <iostream>

#include "hexioctrl.h"

#include "port.h"

using namespace std;

// Макросы, представляющие команды

// для устройств ATA.

#define IDENTIFY\_PACKET\_DEVICE 0xA1

#define IDENTIFY\_DEVICE 0xEC

// Регистры устройств ATA.

const WORD AS[2] = { 0x3F6, 0x376 };

const WORD DR[2] = { 0x1F0, 0x170 };

const WORD DH[2] = { 0x1F6, 0x176 };

const WORD SR[2] = { 0x1F7, 0x177 };

const WORD CR[2] = { 0x1F7, 0x177 };

// Данные, получаемые из регистра данных.

WORD data[256];

// Ожидает, пока устройство ATA занято.

void WaitBusy(int channel);

// Ожидает, пока устройство ATA не готово

// к работе.

BOOL WaitReady(int channel);

// Получение дискового устройства ATA, при

// этом записывая всю необходимую информацию

// в `data'.

BOOL GetATADevice(int channel, int device);

// Печать полученной информации,

// находящейся в `data'.

void PrintData(int channel, int device);

int main()

{

ALLOW\_IO\_OPERATIONS;

setlocale(LC\_ALL, ".1251");

// Печать разделительной линии.

cout << "+" << setw(77) << setfill('=') << "" << "+" << endl;

// Перебираем все возможные

// устройства ATA.

for (int channel = 0; channel < 2; ++channel)

for (int device = 0; device < 2; ++device)

if (GetATADevice(channel, device)) {

PrintData(channel, device);

cout << "+" << setw(77) << setfill('=') << "" << "+" << endl;

}

system("pause");

return 0;

}

void WaitBusy(int channel)

{

BYTE state;

// Ожидаем обнуления бита BSY.

do

state = InPort(AS[channel]);

while (state & 0x80);

}

BOOL WaitReady(int channel)

{

for (int i = 0; i < 1000; ++i)

// Если устройство ATA готово к работе...

if (InPort(AS[channel]) & 0x40)

return true;

return false;

}

BOOL GetATADevice(int channel, int device)

{

// Массив, представляющий

// команды.

const BYTE commands[] = {

IDENTIFY\_PACKET\_DEVICE,

IDENTIFY\_DEVICE,

};

// Перебор всех команд.

for (int i = 0; i < sizeof(commands); ++i) {

// Ожидаем обнуления бита BSY.

WaitBusy(channel);

// Формируем адрес устройства ATA.

OutPort(DH[channel], (device << 4) | 0xE0);

// Дожидаемся признака готовности, если

// устройство ATA присутствует.

if (!WaitReady(channel))

return false;

// Записываем код команды в регистр команд.

OutPort(CR[channel], commands[i]);

// Ожидаем обнуления бита BSY.

WaitBusy(channel);

// Проверка на ошибку.

if (!(InPort(SR[channel]) & 0x08)) {

// Если ошибка произошла при обработке

// команды IDENTIFY\_DEVICE...

if (i == 1)

return false;

continue;

} else

break;

}

// Получение конфигурационного блока.

for (int i = 0; i < 256; ++i)

data[i] = InPortW(DR[channel]);

return true;

}

void PrintData(int channel, int device)

{

cout << setw(30) << setfill(' ') << left << "РАЗМЕЩЕНИЕ:";

cout << channel << ", " << device;

// Печать модели устройства ATA.

cout << endl << endl << setw(30) << setfill(' ') << left << "МОДЕЛЬ:";

for (int i = 27; i <= 46; ++i)

cout << (CHAR)(data[i] >> 8) << (CHAR)(data[i] & 0x00FF);

// Печать серийного номера устройства ATA.

cout << endl << setw(30) << setfill(' ') << left << "Серийный номер:";

for (int i = 10; i <= 19; ++i)

cout << (CHAR)(data[i] >> 8) << (CHAR)(data[i] & 0x00FF);

// Печать версии прошивки устройства ATA.

cout << endl << setw(30) << setfill(' ') << left << "Версия прошивки:";

for (int i = 23; i <= 26; ++i)

cout << (CHAR)(data[i] >> 8) << (CHAR)(data[i] & 0x00FF);

// Печать размера устройства ATA в МБ.

cout << endl << endl << setw(30) << setfill(' ') << left << "РАЗМЕР:";

cout << setprecision(0) << (long double)(((ULONG \*)data)[30]) \* 512 / 1024 / 1024 << " МБ";

// Печать поддержки PIO.

cout << endl << endl << "Поддержка PIO:";

cout << endl << " (" << (data[64] & 0x1 ? "+" : "-") << ") PIO 3";

cout << endl << " (" << (data[64] & 0x2 ? "+" : "-") << ") PIO 4";

// Печать поддержки Multiword DMA.

cout << endl << endl << "Поддержка Multiword DMA:";

cout << endl

**Выводы**

Оперативная память является системным узлом ЭВМ и к периферийным устройствам, строго говоря, не относится. Этот тип памяти обычно обслуживает контроллер памяти, используя специализированную шину. В большинстве современных ЭВМ это линейный (Single Ended) параллельный интерфейс большой разрядности (от 128 бит). Оперативная память конструктивно представляет собой печатные модули с микросхемами DRAM.