**Лабораторная работа N4  
ПРОВЕРКА ОБОРУДОВАНИЯ**

**1. Цель работы**

Получение практических навыков в определении конфигурации и основных характеристик ПЭВМ.

**2. Темы для предварительного изучения**

* Конфигурация ПЭВМ.
* Склад, назначение и характеристики основных модулей ПЭВМ.

**3. Постановка задачи**

Для компьютера на своем рабочем месте определить:

* тип компьютера;
* конфигурацию оборудования;
* объем оперативной памяти;
* наличие и объем расширенной памяти;
* наличие дополнительных ПЗУ;
* версию операционной системы.

**4. Порядок выполнения**

Порядок выполнения работы и содержание отчета определены в общих указаниях.

**5. Пример решения задачи**

**5.1. Структура данных программы**

Программа использует, так называемый, список оборудования - 2-байтное слово в области данных BIOS по адресу 0040:0010. Назначение разрядов списка оборудования такое:

|  |  |
| --- | --- |
| **Биты** | **Содержимое** |
| 0 | установлен в 1, если есть НГМД (см.разряды 6, 7) |
| 1 | установлен в 1, если есть сопроцессор |
| 2,3 | число 16-Кбайтних блоков ОЗУ на системной плате |
| 4,5 | код видеоадаптера: 11 - MDA, 10 - CGA, 80 колонок, 01 - CGA, 40 колонок, 00 - другой |
| 6,7 | число НГМД-1 (если в разряде 0 единица) |
| 8 | 0, если есть канал ПДП |
| 9,10,11 | число последовательных портов RS-232 |
| 12 | 1, если есть джойстик |
| 13 | 1, если есть последовательный принтер |
| 14,15 | число параллельных принтеров |

**5.2. Структура программы**

Программа состоит только из основной функции main(). Выделения фрагментов программы в отдельные процедуры не требуется, потому что нет таких операций, которые во время работы программы выполняются многократно.

**5.3. Описание переменных**

Переменные, применяемые в программе:

* type\_PC - байт типа компьютера, записанный в ПЗУ BIOS по адресу FF00:0FFE;
* a, b - переменные для определения объема extended-памяти ПЭВМ, a - младший байт, b - старший байт;
* konf\_b - 2-байтное слово из области данных BIOS, которое содержит список оборудования;
* type - массив символьных строк, представляющих типы компьютера;
* typ1A - массив байт, содержащий коды типов дисплеев;
* types1A[] - массив строк, содержащий названия типов дисплеев;
* j - вспомогательная переменная, которая используется для идентификации типа дисплея;
* seg - сегмент, в котором размещено дополнительное ПЗУ;
* mark - маркер ПЗУ;
* bufVGA[64] - буфер данных VGA, из которого (при наличии VGA) ми выбираем объем видеопамяти;
* rr и sr - переменные, которые используются для задания значения регистров общего назначения и сегментных регистров, соответственно, при вызове прерывания.

**5.4. Описание алгоритма программы**

Алгоритм основной программы может быть разбито на 5 частей.

Часть 1 предназначена для определения типа компьютера. Для этого прочитаем байт, записанный в ПЗУ BIOS по адресу FF00:0FFE. В зависимости от значения этого байта сделаем вывод о типе ПЭВМ. Так, например, компьютеру типа AT соответствует код 0xFC.

Часть 2 предназначена для определения конфигурации ПЭВМ. Для этого прочитаем из области данных BIOS список оборудования. Для определения количества дисководов (если бит 0 установлен в 1) необходимо выделить биты 6 и 7 (маска 00C0h) и сместить их вправо на 6 разрядов, а потом добавить 1.

Для определения количества 16-Кбайтних блоков ОЗУ на системной плате необходимо выделить биты 2 и 3 с помощью маски 000Ch, сместить вправо на 2 разряды и добавить 1.

Для определения количества последовательных портов RS-232 выделить с помощью маски 0Eh биты 9-11 и сместить вправо на 9 разрядов.

Для определения наличия математического сопроцессора - проверить установку бита 1 маской 0002h.

Для определения наличия джойстика - бита 12 с помощью маски 1000h.

Определить количество параллельных принтеров можно, выделив биты 14 и 15 маской C000h и сместив их вправо на 14 разрядов.

Поскольку список оборудования содержит недостаточно информации про дисплейный адаптер, то для уточнения типа адаптера выполним дополнительные действия.

Видеоадаптер обслуживается прерыванием BIOS 10h. Для новых типов адаптеров список его функций расширяется. Эти новые функции и используются для определения типу адаптера.

Функция 1Ah доступна только при наличии расширения BIOS, ориентированного на обслуживание VGA. В этом случае функция возвращает в регистре AL код 1Ah - свою "визитную карточку", а в BL - код активного видеоадаптера. В случае, если функция 1Ah поддерживается, обратимся еще к функции 1Bh - последняя заполняет 70-байтний блок информации про состояние, из которого мы выбираемо объем видеопамяти.

Если 1Ah не поддерживается, это означает, что VGA у нас нет, в этом случае можно обратиться к функции 12h - получение информации про EGA. При наличии расширения, ориентированного на EGA, эта функция изменяет содержимое BL (перед обращением он должен быть 10h) на 0 (цветной режим) или на 1 (монохромный режим) а в BH возвращает объем видеопамяти.

Если же ни 1Ah, ни 12 не поддерживаются, то список оборудования BIOS содержит достаточную информацию про видеоадаптер и, выделивши биты 4, 5 мы можем сделать окончательный вывод про тип адаптера, который у нас есть.

В третьей части программы определим объем оперативной памяти, наличие и объем extended-памяти. Объем оперативной памяти для AT может быть прочитан из регистров 15h(младший байт) и 16h (старший байт) CMOS-памяти или из области памяти BIOS по адресу 0040:0013 (2-байтное слово). Кроме того, в ПЭВМ может быть еще и дополнительная (expanded) память свыше 1 Мбайту. Ее объем можно получит из регистров 17h (младший байт) и 18h (старший байт) CMOS-памяти. Для чтения регистра CMOS-памяти необходимо видать в порт 70h байт номера регистра, а потом из порта 71h прочитать байт содержимого этого регистра.

В следующей части программы определим наличие и объем дополнительных ПЗУ. В адресному пространстве от C000:0000 по F600:0000 размещаются расширения ПЗУ (эта память не обязательно присутствует в ПЭВМ). Для определения наличия дополнительного ПЗУ будем читать первое слово из каждых 2 Кбайт, начиная с адреса C000:0000 в поисках маркера расширения ПЗУ: 55AAh. Если такой маркер найден, то следующий байт содержит длину модуля ПЗУ.

В заключительной части программы определим версию DOS, установленную на ПЭВМ. Для этого воспользуемся функцией DOS 30h, которая возвращает в регистре AL старшее число номера версии, а в регистре AH - младшее число.

**5.5. Текст программы**

/\*----------------Лабораторная работа N4------------------\*/

/\*-----------"Проверка состава оборудования"--------------\*/

/\* Подключение стандартных заголовков \*/

#include <dos.h>

#include <stdio.h>

/\*--------------------------------------------------------\*/

void main()

{

unsigned char type\_PC, /\* Тип компьютера \*/

a,b; /\* Переменные для определения \*/

/\* характеристик памяти ПЭВМ \*/

unsigned int konf\_b; /\* Байт конфигурации из BIOS \*/

char \*type[]={"AT","PCjr","XT","IBM PC","unknown"};

unsigned char typ1A[]={0,1,2,4,5,6,7,8,10,11,12,0xff};

char \*types1A[]={"нема дисплею","MDA, моно","CGA, цв.",

"EGA, цв.","EGA, моно","PGA, цв.",

"VGA, моно, анал.","VGA, кол., анал.",

"MCGA, кол., цифр.","MCGA, моно, анал."

"MCGA, кол., анал.","неизвестный тип",

"непредусмотренный код"};

unsigned int j; /\* Вспомогательная переменная \*/

unsigned int seg; /\* Сегмент ПЗУ \*/

unsigned int mark=0xAA55; /\* Маркер ПЗУ \*/

unsigned char bufVGA[64]; /\* Буфер данных VGA \*/

union REGS rr;

struct SREGS sr;

textbackground(0);

clrscr();

textattr(0x0a);

printf("Лабораторная работа N5");

printf("\nПроверка состава оборудования");

/\* Определения типа компьютера \*/

type\_PC=peekb(0xF000,0xFFFE);

if( (type\_PC-=0xFC)>4)

type\_PC=4;

textattr(0x0b);

printf("\nТип компьютера: ");

textattr(0x0f);

printf("%s\n\r",type[type\_PC]);

/\* Конфигурация\*/

konf\_b=peek(0x40,0x10); /\* Чтение байта оборудования \*/

/\* из памяти BIOS \*/

textattr(0x0b);

printf("Конфигурация:\n\r");

/\* Количество дисководов \*/

textattr(0x0e);

printf(" Дисководов ГМД: ");

textattr(0x0f);

if(konf\_b&0x0001)

printf("%d\n\r",((konf\_b&0x00C0)>>6)+1);

else

printf("нет\n\r");

textattr(0x0e);

printf(" Математич. сопроцессор: ");

textattr(0x0f);

if(konf\_b&0x0002)

printf("есть\n\r");

else

printf("нет\n\r");

textattr(0x0e);

printf(" Тип дисплейного адаптера: ");

textattr(0x0f);

/\* Определение активного адаптера \*/

/\* Предположим наличие VGA \*/

rr.h.ah=0x1a;

rr.h.al=0;

int86(0x10,&rr,&rr);

if(rr.h.al==0x1a) /\* Поддерживается функция 1Ah \*/

{ /\* прерывания 10h \*/

for(j=0;j<12;j++)

if(rr.h.bl==typ1A[j])

break;

printf("%s",types1A[j]);

if(j>0 && j<12)

{

rr.h.ah=0x1b;

rr.x.bx=0;

sr.es=FP\_SEG(bufVGA);

rr.x.di=FP\_OFF(bufVGA);

int86x(0x10,&rr,&rr,&sr);

printf(", %d Кбайт\n\r",((int)bufVGA[49]+1)\*64);

}

else

printf("\n\r");

}

else

{

/\* Предположим наличие EGA \*/

rr.h.ah=0x12;

rr.h.bl=0x10;

int86(0x10,&rr,&rr);

if(rr.h.bl!=0x10) /\* Поддерживается функция 12h \*/

{ /\* прерывания 10h \*/

printf("EGA");

if(rr.h.bh)

printf(" моно");

else

printf(" кол.");

printf(", %d Кбайт\n\r",((int)rr.h.bl+1)\*64);

}

else

{

/\* CGA или MDA \*/

switch(konf\_b&0x0030)

{

case 0: printf("EGA/VGA\n\r");break;

case 0x10: printf("CGA,40\n\r");break;

case 0x20: printf("CGA,80\n\r");break;

case 0x30: printf("MDA");break;

}

}

}

/\* Блоки ОЗУ на системной плате \*/

textattr(0x0e);

printf("\n\r Первичный блок памяти: ");

textattr(0x0f);

switch (konf\_b&0x000C)

{

case 0:printf("16 Кбайт\n\r");break;

case 4:printf("32 Кбайт\n\r");break;

case 8:printf("48 Кбайт\n\r");break;

case 12:printf("64 Кбайт или больше\n\r");break;

}

/\* Количество последовательных портов RS-232 \*/

textattr(0x0e);

printf(" Портов RS232: ");

textattr(0x0f);

printf("%d\n\r",(konf\_b&0x0E00)>>9);

/\* Наличие джойстика \*/

textattr(0x0e);

printf(" Джойстик: ");

textattr(0x0f);

if(konf\_b&0x1000 )

printf("есть\n\r");

else

printf("нет\n\r");

/\* Количество параллельних принтеров \*/

textattr(0x0e);

printf(" Принтеров: ");

textattr(0x0f);

printf("%d\n\n\r",(konf\_b&0xC000)>>14);

/\* Объем оперативной памяти \*/

textattr(0x0e);

printf("Объем оперативной памяти: ");

textattr(0x0f);

printf("%d Кбайт\n\r",peek(0x40,0x13));

textattr(0x0e);

/\* Наличие и объем extended-памяти \*/

outportb(0x70,0x17);

a=inport(0x71);

outportb(0x70,0x18);

b=inport(0x71);

printf("Объем extended-памяти: ");

textattr(0x0f);

printf("%d Кбайт\n\n\r",(b<<8)|a);

/\* Наличие дополнительных ПЗУ \*/

for( seg=0xC000;seg<0xFFB0;seg+=0x40)

/\* Просмотр памяти от C000:0 с шагом 2 К \*/

if(peek(seg,0)==mark) /\* Маркер найден \*/

{

textattr(0x0a);

printf("Адрес ПЗУ =");

textattr(0x0f);

printf(" %04x",seg);

textattr(0x0a);

printf(". Длина модуля = ");

textattr(0x0f);

printf("%d",512\*peekb(seg,2));

textattr(0x0a);

printf(" байт\n\r",peekb(seg,2));

}

/\* Определение версии операцийной системы \*/

rr.h.ah=0x30;

intdos(&rr,&rr);

textattr(0x0c);

printf("\n\rВерсия MS-DOS ");

textattr(0x0f);

printf("%d.%d\n\r",rr.h.al,rr.h.ah);

textattr(0x0a);

gotoxy(30,24);

printf("Нажмите любую клавишу");

textattr(0x07);

clrscr();

}

**5.6. Результаты работы программы**

В процессе работы программы на экран была выведена такая информация:

Лабораторная работа N4

Проверка состава оборудования

Тип компьютера: AT

Конфигурация:

Дисководов ГМД: 2

Математич. сопроцессор: есть

Тип дисплейного адаптера: VGA, кол., анал., 256 Кбайт

Первичный блок памяти: 16 Кбайт

Портов RS232: 2

Джойстик: нет

Принтеров: 1

Объем оперативной памяти: 639 Кбайт

Объем extended-памяти: 384 Кбайт

Адрес ПЗУ = c000. Длина модуля = 24576 байт

Версия MS-DOS 6.20