**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра вычислительной техники**

отчет

**по лабораторной работе №2**

**по дисциплине «Организация ЭВМ и систем»**

Тема: **Исследование видеосистемы (текстовый режим)**

|  |  |  |
| --- | --- | --- |
| Студенты гр. 3353 |  | Карпенко А.Ю. |
| Преподаватель |  | Гречухин М.Н. |

Санкт-Петербург

2024

**Цель работы:** изучение работы с видеосистемой в текстовом режиме, освоение приемов использования цветовой палитры: изменение цвета символов и фона на всем экране и в отдельном окне.

**Задание:** Ознакомиться с организацией и функциональными

различных типов видеосистем. Ознакомиться с текстовым режимом отображения информации на экран монитора и стандартными библиотечными функциями C++, обслуживающими этот режим.

**Краткие сведения о видеосистемах ПЭВМ, текстовом режиме их работы и функциях обслуживания текстового режима.**

Аппаратные средства для вывода информации

Видеосистема персонального компьютера включает два основных компонента: видеоадаптер (адаптер дисплея) и монитор. Видеоадаптер является сложной электронной платой, которая управляется собственным микропроцессором. Этот микропроцессор зачастую сравним по мощности с центральным процессором компьютера, что позволяет эффективно обрабатывать графику. Видеоадаптеры поддерживают несколько стандартов и состоят из двух ключевых компонентов:

Контроллер – управляет передачей данных для отображения на экране.

Видеопамять (видеобуфер) – хранит данные о том, что должно быть выведено на экран.

Видеопамять и страницы экрана

Видеопамять хранит информацию в виде видеостраниц – наборов данных, представляющих собой полный образ экрана. Некоторые адаптеры поддерживают работу с несколькими страницами, что позволяет быстро переключать выводимые данные. Видеоадаптер обновляет изображение на экране до 25 и более раз в секунду, создавая картинку из отдельных пикселей (pixel – Picture Element).

Управление видеосистемой

Управление параметрами видеосистемы можно выполнять на двух уровнях:

Через порты видеоадаптера – доступ к аппаратным средствам напрямую.

С помощью BIOS – через функции BIOS, обычно вызываемые с помощью прерывания 10h.

Работа с текстовой информацией

Вывод информации на экран осуществляется на трех уровнях:

MS-DOS (прерывание 21h) – это самый медленный, но мобильный способ. Он вызывает драйвер консоли, который выполняет вывод символов в файл CON.

BIOS (прерывание 10h) – предоставляет больше возможностей для управления экраном, но менее эффективен по скорости.

Непосредственный доступ к аппаратным средствам – самый быстрый способ, позволяющий полноценно управлять видеопамятью и курсором.

Управление текстовым выводом

На уровне BIOS доступно более детальное управление экраном. С помощью таких функций, как:

window(int x1, int y1, int x2, int y2) – задается активное текстовое окно.

clrscr() – очищает активное окно.

textbackground(int newcolor) – изменяет цвет фона окна, позволяя выбрать цвет из диапазона 0–7.

Функции библиотеки <conio.h>, такие как cprintf(), выполняют форматированный вывод текста в пределах окна, аналогично стандартной библиотечной функции printf(). Различие заключается в том, что функция cprintf() не возвращает курсор в начало строки после перевода строки, а также позволяет работать с цветом и управлять текстовыми атрибутами.

**Текст программы**

#include <stdio.h>

#include <conio.h>

#include <dos.h>

void scroll(int direction, char l\_row, char l\_col,

char r\_row, char r\_col, char attr)

{

union REGS r;

if (direction)

{

r.h.al = 1;

r.h.ah = direction;

} else {

r.h.al = 0;

r.h.ah = 7;

}

r.h.ch = l\_row;

r.h.cl = l\_col;

r.h.dh = r\_row;

r.h.dl = r\_col;

r.h.bh = attr;

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

}

void show\_col(int bg\_col, int text\_col)

{

switch (bg\_col)

{

case 0: cprintf("BG: 0"); break;

case 1: cprintf("BG: 1"); break;

case 2: cprintf("BG: 2"); break;

case 3: cprintf("BG: 3"); break;

case 4: cprintf("BG: 4"); break;

case 5: cprintf("BG: 5"); break;

case 6: cprintf("BG: 6"); break;

case 7: cprintf("BG: 7"); break;

}

cprintf(" ");

switch (text\_col)

{

case 0: cprintf(" Txt=BLACK"); break;

case 1: cprintf(" Txt=BLUE"); break;

case 2: cprintf(" Txt=GREEN"); break;

case 3: cprintf(" Txt=CYAN"); break;

case 4: cprintf(" Txt=RED"); break;

case 5: cprintf(" Txt=MAGENTA"); break;

case 6: cprintf(" Txt=BROWN"); break;

case 7: cprintf(" Txt=LIGHTGRAY"); break;

case 8: cprintf(" Txt=DARKGRAY"); break;

case 9: cprintf(" Txt=LIGHTBLUE"); break;

case 10: cprintf(" Txt=LIGHTGREEN"); break;

case 11: cprintf(" Txt=LIGHTCYAN"); break;

case 12: cprintf(" Txt=LIGHTRED"); break;

case 13: cprintf(" Txt=LIGHTMAGENTA"); break;

case 14: cprintf(" Txt=YELLOW"); break;

case 15: cprintf(" Txt=WHITE"); break;

}

}

int main()

{

int bg\_col;

int text\_col;

int DOWN;

int j;

int k;

clrscr();

window(15, 10, 65, 20);

bg\_col = 0;

text\_col = 15;

DOWN = 7;

for ( j= 0; j < 8; j++) {

text\_col = 15;

for (k = 0; k < 16; k++)

{

scroll(DOWN, 10, 15, 20, 65, bg\_col << 4);

gotoxy(2, 2);

textbackground(bg\_col);

textcolor(text\_col);

show\_col(bg\_col, text\_col);

text\_col--;

delay(200);

}

bg\_col++ ;

}

getch();

return 0;

}