**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра информационных систем**

отчет

**по лабораторной работе №4**

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

Тема: **КЛАВИАТУРА IBM PC. ИСПОЛЬЗОВАНИЕ ПРЕРЫВАНИЙ**

|  |  |  |
| --- | --- | --- |
| Студент(ка) гр. |  | Алексеев Г. |
| Преподаватель |  | Гречухин М. Н. |

Санкт-Петербург

2023

**Краткие сведения о подсистеме ввода информации с клавиатуры, используемых прерываниях, буфере клавиатуры и функция обслуживания ввода с клавиатуры.**

Подавляющее большинство программ выполняют ввод информации с клавиатуры. Ввод информации в компьютер может быть выполнен на трех уровнях: обращением к функциям MS-DOS; обращением к функциям BIOS; физическим доступом к аппаратным средствам.

Ввод информации на уровне MS-DOS позволяет "пропустить" клавиатурный ввод через инсталлируемые драйверы, обеспечивает отслеживание нажатия комбинации клавиш Ctrl-C (Ctrl-Break), стандартную для MS-DOS обработку ошибок.

Доступ к клавиатуре на уровне BIOS позволяет программе отслеживать нажатие всех, а не только символьных клавиш, выполнять управление аппаратурой клавиатуры и пр. Интерфейсом Turbo С с BIOS является функция bioskey().

Непосредственный доступ к буферу клавиатуры резко повышает производительность программы. В некоторых случаях необходима имитация нажатий клавиш клавиатуры с записью кодов непосредственно в буфер. При этом физически нажатия клавиш не происходят. Так строятся многие демонстрационные программы, которые открывают или закрывают окна меню, выполняют необходимый выбор, показывают работу программы в "автоматическом" режиме и т.п. На том же самом принципе имитации нажатий клавиш построены программы, способные переносить одним нажатием клавиши целые куски текста из одной программы в любой текстовый редактор. Примером такой программы является входящая в Turbo С резидентная Help-система THELP.COM.

int getch (void)

Выполняет ввод с клавиатуры через функцию MS-DOS АН=07h. Она не выполняет "эхо" вывода на экран. В этой связи полезна для организации интерфейса с пользователем, при котором нажатие той или иной клавиши вызывает немедленную реакцию программы без отображения введенного символа на экране.

int getche (void)

Выполняет не буферизуемый ввод с клавиатуры через функцию MS-DOS AH=07h, но в отличие от предыдущей функции обеспечивает вывод введенного символа на экран. Перевод строки происходит при достижении правой вертикальной границы текущего активного окна.

char \*getpass(char \* prompt)

Выводит на экран ASCII-строку, на начало которой указывает prompt, a затем принимает с клавиатуры без "эха" строку символов. Вводимые символы (не более 7) помещаются во внутреннюю статическую память. Функция возвращает указатель на внутреннюю статическую строку, переопределяемую каждым новым обращением к функции. Основное назначение данной функции - ввод паролей в программе без отображения их на экран.

int kbhit (void)

Проверяет, пуст ли буфер клавиатуры. Если в буфере есть символы, функция возвращает ненулевое значение, в противном случае она возвращает 0. Использует функцию 0Bh MS-DOS. Является удобным средством предотвращения "зацикливания" при ожидании невозможного в данный момент события. Кроме того, при выполнении функции 0Bh осуществляется проверка нажатия комбинации клавиш Ctrl-Break, что позволяет выполнить аварийное завершение программы.

Интерфейсом программ в персональном компьютере с клавиатурой является прерывание 16h BIOS. Далее приводится описание его функций.

АН = 00h - чтение с ожиданием двухбайтового кода из буфера клавиатуры. Прочитанный код возвращается в регистре АХ: младший байт - в регистре AL, старший - в АН. Если нажата ASCII-клавиша, в AL помещается ASCII-код символа, в АН - скэн-код. При нажатии специальных клавиш AL равен 0, а в АН возвращается расширенный скэн-код.

АН = 0lh - чтение без ожидания двухбайтового кода из буфера клавиатуры. Если буфер пуст, в 1 выставляется флаг нуля ZF. В противном случае в АХ возвращается двухбайтовый код из буфера клавиатуры, но продвижение указателя "головы" буфера не производится, т.е. код "остается" в буфере.

АН = 02h - определение состояния шифт- и триггерных клавиш. В регистре AL возвращается содержимое байта по адресу 40:17h (см. табл. 4.1).

Функция АН = 05h не имеет аналогов в библиотеке Turbo С и может использоваться для имитации нажатии клавиш в демонстрационных программах, программах переноса текста и т.д.

Функции АН = 10 - 12h являются аналогами функций 00 - 02h, но предназначены для использования в компьютерах с клавиатурой 101 /102 клавиши.

Функции АН = 00 - 02h прерывания 16h BIOS положены в основу функции bioskey() библиотеки Turbo С. Далее следует описание этой функции.

int bioskey(int cmd)

Обращается в зависимости от значения в cmd к функциям АН = 00 - 02h прерывания 16h. Возвращаемое функцией значение повторяет значение регистра АХ при выходе из прерывания.

**Цель работы.**

изучение возможностей работы с клавиатурой, ознакомление со стандартными средствами библиотеки C++ и средствами системы прерываний DOS и BIOS, обслуживающими клавиатуру.

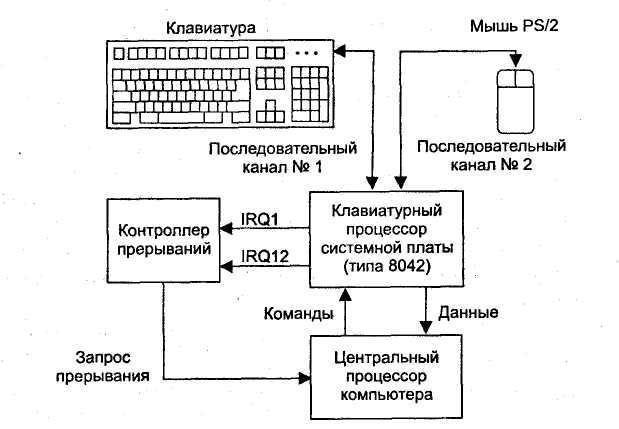
**Задание на лабораторную работу.**

1. Разработать, написать и отладить программу управления перемещением символа "\*" в пределах заданного на экране окна.

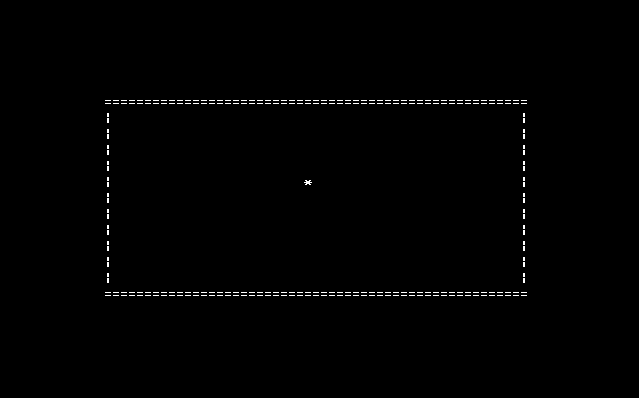


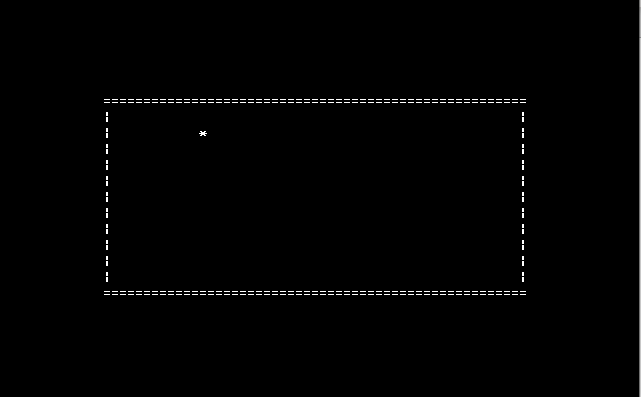
1. Изменить программу, заменив стандартные функции библиотеки C++ своими.

**Структурная схема аппаратных средств**

****

**Результат работы программы**

****

****

**Выводы.**

В ходе данной лабораторной работы были изучены возможности работы с клавиатурой, ознакомились со стандартными средствами библиотеки C++ и средствами системы прерываний DOS и BIOS, обслуживающими клавиатуру.

**Исходный код программы**

#include <iostream.h>

#include <stdlib.h>

#include <conio.h>

#include <dos.h>

#include <bios.h>

#define LEFT 15104 //code of f1

#define RIGHT 15872 //code of f4

#define UP 15360 //code of f2

#define DOWN 15616 //code of f3

#define ESC 283

void printWindow(int x1, int y1, int x2, int y2) {

// Border window

clrscr();

window(x1 - 1, y1 - 1, x2 + 1, y2 + 2);

textbackground(BLACK);

textcolor(WHITE);

// Upper horizontal line

cprintf("=");

for (int i = 0; i < x2 - x1 + 1; ++i)

cprintf("=");

cprintf("=");

// Vertical lines

for (i = 0; i < y2 - y1 + 1; ++i) {

cprintf("|");

for (int j = 0; j < x2 - x1 + 1; ++j)

cprintf(" ");

cprintf("|");

}

// Lower horizontal line

cprintf("=");

for (i = 0; i < x2 - x1 + 1; ++i)

cprintf("=");

cprintf("=");

// Active window

window(x1, y1, x2, y2);

}

int getMode() {

int ch;

ch = bioskey(0);

if (ch == DOWN) return 1;

if (ch == UP) return 2;

if (ch == LEFT) return 3;

if (ch == RIGHT) return 4;

if (ch == ESC) return -1;

return 0;

}

int main() {

int x1 = 15, y1 = 8, x2 = 65, y2 = 18;

int x = (x2 - x1) / 2, y = (y2 - y1) / 2; //change the center of area

printWindow(x1, y1, x2, y2);

\_setcursortype(\_NOCURSOR);

gotoxy(x, y);

cprintf("\*");

while (1) {

while (bioskey(1) == 0) {

int choice = getMode();

switch (choice) {

case -1: // Escape

return 0;

case 1: // Down(f3)

y++;

if (y == y2 - y1 + 2) y = 1;

clrscr();

gotoxy(x,y);

cprintf("\*");

break;

case 2: // Up(f2)

y--;

if (y == 0) y = y2 - y1 + 1;

clrscr();

gotoxy(x,y);

cprintf("\*");

break;

case 3: // Left(f1)

x--;

if (x == 0) x = x2 - x1 + 1;

clrscr();

gotoxy(x,y);

cprintf("\*");

break;

case 4: // Right(f4)

x++;

if (x == x2 - x1 + 2) x = 1;

clrscr();

gotoxy(x,y);

cprintf("\*");

break;

default: // Unknown command

continue;

}

}

}

}