Национальный технический университет Украины

«Киевский политехнический институт»

Факультет информатики и вычислительной техники

Кафедра вычислительной техники

**Лабораторная работа №4**

*по курсу «Периферийные устройства»*

Выполнил

студент группы ИВ-73

Захожий Игорь

Киев-2010

**Задание:**

Написать драйвер мыши, который принимает сообщения мыши и передвигает курсор в виде символа в заданном направлении и выполняет заданные команды при нажатии на клавиши (L и R). Обозначения: направления: 1 – вверх, 2 – вверх-вправо, 3 – вправо, 4 – вниз-вправо, 5 – вниз, 6 – вниз-влево, 7 – влево, 8 – вверх-вправо. P – переход курсора на противоположный край экрана, О – остановка курсора на краю, В – выход из программы, С – смена символа курсора, У – установка курсора на центр экрана, П – потоковый, З – по запросу.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *1* | *3* | *5* | *7* | *На границе* | *L* | *R* | *Символ курсора* |  |
| 5 | 8 | 2 | 3 | Р | У | В | $ | П |

**Листинг программы:**

#include <stdio.h>

#include <conio.h>

#include <math.h>

void getMouseState(int\* x, int\* y, int\* buttons) {

int tx, ty, tb;

asm {

mov ax, 3

int 33h

mov tx, cx

mov ty, dx

mov tb, bx

}

\*x = tx;

\*y = ty;

\*buttons = tb;

}

int mouseX, mouseY, newMouseX, newMouseY, x, y, mouseButtons, buttons;

int temp;

int scale = 20;

int minX = 1;

int minY = 1;

int maxX = 78;

int maxY = 23;

char cursor = '$';

char quit = 1;

void showCursor() {

clrscr();

gotoxy(x, y);

printf("%c", cursor);

}

void setCenter() {

x = maxX / 2;

y = maxY / 2;

}

void main() {

clrscr();

asm {

mov ah, 0

mov al, 2

int 10h

mov ax, 0

int 33h

mov temp, ax

}

if (temp != 0) {

buttons = 0;

setCenter();

getMouseState(&mouseX, &mouseY, &buttons);

showCursor();

while (quit) {

getMouseState(&newMouseX, &newMouseY, &mouseButtons);

if ((abs(newMouseX - mouseX) > scale) || (abs(newMouseY - mouseY) > scale)) {

if (newMouseX < mouseX) {

x += ((mouseX - newMouseX) / scale);

}

else {

y -= ((newMouseX - mouseX) / scale);

x += ((newMouseX - mouseX) / scale);

}

if (newMouseY < mouseY) {

y += ((mouseY - newMouseY) / scale);

}

else {

y -= ((newMouseY - mouseY) / scale);

x += ((newMouseY - mouseY) / scale);

}

if (x < minX) {

x = maxX - abs(x);

}

if (x > maxX) {

x = x + minX - maxX;

}

if (y < minY) {

y = maxY - abs(y);

}

if (y > maxY) {

y = y + minY - maxY;

}

mouseX = newMouseX;

mouseY = newMouseY;

showCursor();

}

if ((mouseButtons == 0) && (buttons == 1)) {

setCenter();

showCursor();

}

if ((mouseButtons == 0) && (buttons == 2)) {

quit = 0;

}

buttons = mouseButtons;

}

}

else {

printf("Error! Unable to initialize the mouse!");

}

}