Министерство образования Республики Беларусь

Учреждение Образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Кафедра электронных вычислительных машин

Лабораторная работа № 6

«Программирование клавиатуры»

Проверил: Выполнил:

доцент Одинец Д. Н. ст. гр. 950503

Гуринович А. В.

Минск 2021

1. **Постановка задачи**

Программируя клавиатуру помигать ее индикаторами. Алгоритм мигания произвольный. Условия реализации программы, необходимые для выполнения лабораторной работы:

1. Запись байтов команды должна выполняться только после проверки незанятости входного регистра контроллера клавиатуры. Проверка осуществляется считывание и анализом регистра состояния контроллера клавиатуры.
2. Для каждого байта команды необходимо считывать и анализировать код возврата. В случае считывания кода возврата, требующего повторить передачу байта, необходимо повторно, при необходимости – несколько раз, выполнить передачу байта. При этом повторная передача данных не исключает выполнения всех оставшихся условий.
3. Для определения момента получения кода возврата необходимо использовать аппаратное прерывания от клавиатуры.
4. Все коды возврата должны быть выведены на экран в шестнадцатеричной форме.
5. **Алгоритм решения задачи**

Для вывода на экран скан-кодов или кодов возврата необходимо заменить обработчик прерывания 09h. При вызове данного обработчика выводится значение из порта 60h на экран. При управлении индикаторами значение из порта 60h (код возврата) необходимо анализировать на случай необходимости повторной передачи байтов команды.

В случае считывания кода возврата, требующего повторить передачу байта, устанавливается флаг isResend, сигнализирующий о том, что необходимо повторить передачу команды в порт 60h.

Для управления индикаторами клавиатуры используется команда *EDh*. Второй байт этой команды содержит битовую маску для настройки индикаторов (бит 0 – состояние Scroll Lock, бит 1 – состояние Num Lock, бит 2 – состояние Caps Lock). В данной программе управление индикаторами реализовано в функции void indicator (unsigned char mask), где mask – битовая маска, определяющая состояние индикаторов.

Перед каждой командой записи происходит ожидание освобождения входного буфера клавиатуры: while((inp(0x64) & 0x02) != 0x00).

Главная процедура выполняет следующие действия:

1. Запоминает адрес старого обработчика прерывания 9h, вызывая функцию getvect() с параметром INT = 9.

2. Записывает в таблицу векторов прерываний адрес нового обработчика прерывания с помощью функции setvect().

3. В цикле (пока не установлен флаг выхода quitFlag) производится вызов функции мигания индикаторами blinking() (если установлен флаг мигания индикаторами blinkingON).

4. Параллельно с этим происходит отслеживание прерываний 09h, и при его возникновении выводится скан-код нажатой клавиши или код возврата.

5. В конце работы программа восстанавливает в таблице векторов прерываний адрес старого обработчика.

1. **Листинг программы**

//

// main.cpp

// Labwork 6. Keyboard

//

// Created by Andrej Hurynovič on 9.05.21.

//

#include <dos.h>

#include <conio.h>

#include <stdio.h>

#include <iostream.h>

**void** interrupt far IRQ1(...);

**void** interrupt far (\*originalIRQ1)(...);

**void** indicator(**unsigned** **char** mask);

**void** blinking(**void**);

**int** isResend = 1;

**int** quitFlag = 0;

**int** blinkingFlag = 0;

**int** main() {

//save original interrupton and setup own

originalIRQ1 = getvect(0x09);

setvect(0x09, IRQ1);

**while**(!quitFlag) {

**if**(blinkingFlag)

blinking();

}

setvect(0x09, originalIRQ1);

cout << "Press any key to continue..." << endl;

getch();

**return** 0;

}

**void** interrupt far IRQ1(...) {

**unsigned** **char** value = 0;

value = inp(0x60); //get value from 60h port

**if**(value == 0x10) { //10 == 'Q' scan-code

quitFlag = 1;

}

**if**(value == 0x39 && blinkingFlag == 0) { // 39 == ' ' scan-code

blinkingFlag = 1;

} **else** {

**if**(value == 0x39 && blinkingFlag == 1) {

blinkingFlag = 0;

}

}

**if**(value != 0xfa && blinkingFlag == 1) { //no success code

isResend = 1;

} **else** {

isResend = 0;

}

cout << "Key scan-code: "<< hex << (**int**)value << endl;

outp(0x20, 0x20); //reset interruption controller

}

**void** indicator(**unsigned** **char** mask) {

isResend = 1;

**while**(isResend) { //no success command code

**while**((inp(0x64)&0x02) != 0x00); //wait for keyboard bufer update

outp(0x60, 0xED); //indicator control command

delay(50);

}

isResend = 1;

**while**(isResend) { //no success command code

**while**((inp(0x64)&0x02) != 0x00);

outp(0x60, mask); //write mask for this indicator

delay(50);

}

}

**void** blinking(){

indicator(0x02); //enable Num Lock indicator

delay(150);

indicator(0x04); //enable Caps Lock

delay(150);

indicator(0x06); //enable Num Lock i Caps Lock

delay(200);

indicator(0x00); //disable all indicators

delay(50);

indicator(0x06); //enable Num Lock & Caps Lock indicator

delay(100);

indicator(0x00); //disable all indicators

cout << "Blinking end!" << endl;

}

1. **Заключение**

#### Программа выводит на экран скан-коды клавиш при их нажатии и отпускании. Мигание индикаторов включается или выключается нажатием на клавишу q. Во время мигания на экран выводятся коды возврата для каждого байта команды. Выход из программы производится по нажатию Esc.