**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра математического обеспечения и применения ЭВМ**

отчет

**по лабораторной работе №5**

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

**Тема: Разработка собственного прерывания.**

|  |  |  |
| --- | --- | --- |
| Студент гр. 8381 |  | Киреев К.А. |
| Преподаватель |  | Кирьянчиков В.А. |

Санкт-Петербург

2019

**Цель работы.**

Знакомство с организацией прерываний в компьютере. Написание собственного обработчика прерываний.

**Задание.**

# Разработать обработчик прерывания с вектором: 23h - прерывание, генерируемое при нажатии клавиш Control+C. Обработчик прерываний

# должен выводить звуковой сигнал.

**Теоретические сведения.**

Прерывание - это процесс вызова процедур для выполнения некоторой задачи, обычно связанной с обслуживанием некоторых устройств (обработка сигнала таймера, нажатия клавиши и т.д.). Когда возникает прерывание, процессор прекращает выполнение текущей программы (если ее приоритет ниже) и запоминает в стеке вместе с регистром флагов адрес возврата(CS:IP) - места, с которого будет продолжена прерванная программа. Затем в CS:IP загружается адрес программы обработки прерывания и ей передается управление. Адреса 256 программ обработки прерываний, так называемые векторы прерывания, имеют длину по 4 байта (в первых двух хранится значение IP , во вторых - CS) и хранятся в младших 1024 байтах памяти. Программа обработки прерывания должна заканчиваться инструкцией IRET (возврат из прерывания), по которой из стека восстанавливается адрес возврата и регистр флагов.

Две последние строки обработчика прерывания, указанные перед командой IRET выхода из прерывания, необходимы для разрешения обработки прерываний с более низкими уровнями, чем только что обработанное.

Программа, использующая новые программы обработки прерываний при своем завершении должна восстанавливать оригинальные векторы прерываний. Функция 35 прерывания 21H возвращает текущее значение вектора прерывания, помещая значение сегмента в ES, а смещение в BX.

Для установки адреса нового обработчика прерывания в поле векторов прерываний используется функция 25H прерывания 21H, которая помещает заданные адреса сегмента и смещения обработчика в вектор прерывания с заданным номером.

Обработка звука:

Порт 61H - это порт вывода микросхемы 8255 в машине фирмы IBM. Всякий выходной порт захватывает (временно запоминает) данные, выводимые программой. Если бы аппаратура не запоминала данные, они бы пропали в течение микросекунды или около этого. Такое запоминание данных позволяет сохранять их значение в порте до тех пор, пока они снова не будут изменены программой. То есть, когда мы выводим значение, меняющее положение диффузора динамика, оно остается неизменными до тех пор, пока его не изменит программа.

При изучении управления динамиком имеют значение только биты 0(порт 2 таймера - управление динамиком) и 1(прямое управление динамиком).

**Вывод.**

В результате выполнения данной лабораторной работы были получены навыки написания собственного обработчика прерывания.

# **ПРИЛОЖЕНИЕ**

**ИСХОДНЫЙ КОД ПРОГРАММЫ**

SSTACK SEGMENT STACK

DB 1024 DUP(?)

SSTACK ENDS

DATA SEGMENT

KEEP\_CS DW 0 ; хранение сегмента вектора прерывания

KEEP\_IP DW 0 ; хранение смещения вектора прерывания

TONE DW 4351 ; частота звука

TIME DW 1000 ; длительность

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE, DS:DATA, SS:SSTACK

SOUND PROC

PUSH AX

PUSH CX

PUSH DX

; -----управление высотой звука динамика-----

MOV AL, 0b6H ; заносим слово состояния 10110110b

OUT 43H, AL ; в командный регитср (порт 43h)

MOV AX, TONE

OUT 42H, AL ; в порт 42h младшее слово ax:al

MOV AL, AH

OUT 42H, AL ; в порт 42h старшее слово ax:ah

IN AL, 61H ; получить состояние динамика

MOV AH, AL

OR AL, 3 ; инициализируем динамик

OUT 61H, AL ; и подаем ток в порт 61h

MOV BX, TIME

\_LOOP1: MOV CX, 2083 ; цикл ожидания, во время которого

\_LOOP2: LOOP \_LOOP2 ; работает динамик

DEC BX

JNZ \_LOOP1

MOV AL, AH ; выключить динамик

OUT 61H, AL

POP DX

POP CX

POP AX

RET

SOUND ENDP

SUBR\_INT PROC FAR

PUSH DX ; сохранение изменяемых регистров

PUSH AX

CALL SOUND

MOV AL, 20H ; разрешение обработки прерываний с более низкими уровнями,

OUT 20H, AL ; чем только что обработанное

POP AX ; восстановление регистров

POP DX

IRET

SUBR\_INT ENDP

MAIN PROC FAR

PUSH DS

SUB AX, AX

PUSH AX

MOV AX, DATA

MOV DS, AX

MOV AH, 35H ; функция получения вектора

MOV AL, 23H ; номер вектора

INT 21H

MOV KEEP\_IP, BX ; запоминание смещения вектора прерывания

MOV KEEP\_CS, ES ; запоминание сегмента вектора прерывания

PUSH DS

MOV DX, OFFSET SUBR\_INT ; смещение для процедуры в DX

MOV AX, SEG SUBR\_INT ; сегмент процедуры

MOV DS, AX ; помещаем в DS

MOV AH, 25H ; функция установки вектора

MOV AL, 23H ; номер вектора

INT 21H ; меняем прерывание

POP DS

BEGIN:

XOR AH, AH ; функция 00h

INT 16H ; читать (ожидать) следующую нажатую клавишу

CMP Al, 3

JNZ BEGIN

INT 23H

CLI ; сброс флага прерывания

PUSH DS

MOV DX, KEEP\_IP ; смещение для процедуры

MOV AX, KEEP\_CS ; сегмент процедуры

MOV DS, AX ; помещаем в DS

MOV AH, 25H ; функция установки вектора

MOV AL, 23H ; номер вектора

INT 21H ; восстанавливаем вектор

POP DS

STI ; установка флага прерывания

RET

MAIN ENDP

CODE ENDS

END MAIN