**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 |  | Сергеев А.Д. |
| Преподаватель |  | Кирьянчиков В.А. |

Санкт-Петербург

2019

**Цель работы.**

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

**Задание.**

# Вариант 4B: разработать обработчик прерывания с вектором: 1Ch - прерывание от часов - генерируется автоматически операционной системой 18 раз в сек. 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.

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 стандартного прерывания

KEEP\_IP DW 0 ; хранение смещения вектора стандартного прерывания

TONE DW 10000 ; частота звука, будет уменьшаться на 1000 с каждым выполнением прерывания

TIME DW 50000 ; длительность звучания звука, в микросекундах

VECTOR DB 1 ; направление изменения тона, 1 - уменьшение, 0 - увеличение

MSG DB 'Var 4B: making sound after clock interruption.', 0Dh, 0Ah, 'Press CTRL + C to terminate program.', '$'

DATA ENDS

CODE SEGMENT

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

SOUND PROC ; процедура, воспроизводящая звук

push cx ; необходимо сохранить только cx, т.к. ax был сохранён ранее

cli ; отключение прерываний на время настройки динамика

mov al, 0B6h

out 43h, al

mov ax, TONE ; настройка тона звука

out 42h, al

mov al, ah

out 42h, al

sti ; включение прерываний

in al, 61h ; включение динамика

or al, 3

out 61h, al

mov cx, TIME

sound\_timer: ; ожидание в течение установленного времени

loop sound\_timer

in al, 61h ; выключение динамика

and al, 0FCh

out 61h, al

pop cx ; возвращение cx

cmp VECTOR, 1 ; проверка, в какую сторону смещается тон и смещаем его

jne back

sub TONE, 100

jmp for

back:

add TONE, 100

for:

cmp TONE, 100 ; проверка, не нужно ли изменить направление вектора

je vector\_backwards

cmp TONE, 10000

je vector\_forwards

ret

vector\_forwards: ; изменение ноправление вперёд

mov VECTOR, 1

ret

vector\_backwards: ; изменение направление назад

mov VECTOR, 0

ret

SOUND ENDP

SUBR\_INT PROC FAR

push ax ; сохранение значения регистров

push dx

push ds

mov ax, DATA ; смещение блок данных, чтобы иметь доступ к необходимым данным

mov ds, ax

call SOUND

mov al, 20h ; разрешение обработки прерываний с более низкими уровнями, чем только что обработанное

out 20h, al

pop ds

pop dx

pop ax

iret

SUBR\_INT ENDP

MAIN PROC FAR

push ds

mov ax, 0

push ax

mov ax, DATA

mov ds, ax

mov ah, 35h ; сохранение стандартного прерывания в память

mov al, 1Ch

int 21h

mov KEEP\_IP, bx

mov KEEP\_CS, es

mov dx, OFFSET MSG ; вывод сообщения на экран

mov ah, 9

int 21h

cli ; отключение прерываний на время настройки прерывания

push ds

mov dx, OFFSET SUBR\_INT

mov ax, SEG SUBR\_INT

mov ds, ax

mov ah, 25h

mov al, 1Ch

int 21h ; установка нового прерывания вместо стандартного

pop ds

sti ; включение прерываний

symbol\_reader: ; чтение символа с клавиатуры

mov ah, 0

int 16h

cmp al, 3

jne symbol\_reader

cli

mov dx, KEEP\_IP

mov ax, KEEP\_CS

mov ds, ax

mov ah, 25h

mov al, 1Ch

int 21h ; восстановление стандартного прерывания

sti

ret

MAIN ENDP

CODE ENDS

END MAIN