**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №5**

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

Тема «**Написание собственного прерывания**»

|  |  |  |
| --- | --- | --- |
| Студентка гр. 1303 |  | Сырцева Д.Д |
| Преподаватель |  | Ефремов М.А |

Санкт-Петербург

2022

## Цель работы.

Изучить особенности прерываний на языке Ассемблера. Разработать собственное прерывание.

## Задание.

Вариант 23.

Написать прерывание 16h – прерывание от клавиатуры. По заданному скан-коду клавиши произвести выдачу звукового сигнала с заданной высотой звука. Отвести программе под стек не менее 1К байт

## Выполнение работы.

Сегмент данных – my\_data:

1. delay dw 2000 – переменная, хранящая нужную задержку.
2. cached\_cs dw 0 – переменная для хранения сегмента того прерывания, которое мы заменяем.
3. cached\_ip dw 0 – переменная для хранения смещения замененного нами прерывания.

Под стек выделяется 1Кбайт.

В начале программы происходит сохранение адреса старого обработчика прерывания с помощью функции 35h прерывания 21h. Далее устанавливается адрес нового обработчика прерывания с помощью функции 25h прерывания 21h. Далее определена метка input\_loop – ждать нажатие клавиши, здесь происходит считывание информации из порта клавиатуры (порт 60h). Если в регистре al теперь хранится значение 30h, что соответствует скан-коду клавиши B, то вызывается написанное нами прерывание, если нет, то с помощью команды jne снова происходит переход на метку input\_loop. Также в данной метке в регистр al переносится значение будущей высоты звука (задается высота 100).

В начале сегмента кода my\_code определяется процедура пользовательского прерывания – my\_interruption. В данной процедуре сначала сохраняются в стеке значения регистров до входа в прерывание. Далее происходит сохранение состояния динамика, включается динамик и таймер для подачи сигналов на динамик, также устанавливается длительность/высота звука. Далее идет цикл sound\_duration с вложенным циклом sound\_duration\_2. Это реализовано для того, чтобы выдача звука происходила не слишком быстро (также для этого используется инструкция nop, которая ничего не делает, но занимает определенное время). После выполнения циклов динамик выключается и восстанавливается его прежнее состояние, также восстанавливаются регистры. После чего происходит возврат из прерывания(iret).

После выполнения прерывания происходит сброс флага прерывания (cli), восстанавливается старый вектор прерывания и происходит установка флага прерывания(sti).

## Вывод.

В результате лабораторной работы было написано собственное прерывание, которое выдает звуковой сигнал с заданной высотой звука. .

## ПРИЛОЖЕНИЕ А

## Текст исходного файла программы lab5.

assume cs:my\_code, ds:my\_data, ss:my\_stack

my\_stack segment stack

db 1024 dup(0)

my\_stack ends

my\_data segment

delay dw 2000

cached\_cs dw 0

cached\_ip dw 0

my\_data ends

my\_code segment

my\_interruption proc far

push ax

push dx

mov dx, ax

in al, 61h ; Информация о динамике

push ax

or al, 00000011b

out 61h, al ; Включить динамик

mov al, dl

out 42h, al ; Включить таймер для подачи сигналов на динамик

mov cx, delay

; Задержка

sound\_duration:

push cx

mov cx, delay

sound\_duration\_2:

nop

loop sound\_duration\_2

pop cx

loop sound\_duration

pop ax

and al, 11111100b

out 61h, al ; Выключить динамик

pop dx

pop ax

mov al, 20h

out 20h, al

iret

my\_interruption endp

main proc far

push ds

xor ax, ax

push ax

mov ax, my\_data

mov ds, ax

mov ah, 35h ; сохраняем адрес старого прерывания

mov al, 16h

int 21h

mov cached\_cs, es

mov cached\_ip, bx

push ds

mov dx, offset my\_interruption ; смещение для процедуры в DX

mov ax, seg my\_interruption ; сегмент процедуры

mov ds, ax

mov ah, 25h ; функция установки вектора

mov al, 16h ; номер вектора

int 21h ; меняем прерывание

pop ds

input\_loop: ; ждать нажатие клавиши

in al, 60h ; считать информацию из порта ввода клавиатуры

cmp al, 30h ; 30h -- b

jne input\_loop

mov al, 100

int 16h

cli ; Сброс флага прерывания

push ds

mov dx, cached\_ip

mov ax, cached\_cs

mov ds, ax

mov ah, 25h

mov al, 16h

int 21h ; восстанавливаем старый вектор прерывания

pop ds

sti ; Установка флага прерывания

ret

main endp

my\_code ends

end main