

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ
по лабораторной работе №5
по дисциплине «Организация ЭВМ и систем»
Тема: Разработка собственного прерывания

Студент гр. 9382

Кузьмин Д. И.

Преподаватель

Ефремов М. А.

Санкт-Петербург

2020

Цель работы.

Изучить принципы работы прерываний. Освоить навыки разработки собственных обработчиков прерываний.

Основные теоретические положения.

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.

Задание.

Вариант 12(4С)


08h - прерывание от системного таймера - генерируется автоматически операционной системой 18 раз в сек.

С - Приостановить вывод на экран (вставить цикл задержки).


Выполнение работы.

 ① Первым шагом было объявление стека и сегмента данных.

 ① Затем была описана процедура, обрабатывающая прерывание

 ① Затем при помощи прерываний 35h и 25h были сохранены старое значение вектора прерывания и записано новое

 ① Далее следовал вывод сообщения с задержкой

 ① В конце используемому вектору прерывания присваивалось старое значение

Исходный код представлен в приложении А.

Выводы.

Были изучены принципы работы прерываний. Получены навыки разработки программ, реализующий собственные прерывания.

ПРИЛОЖЕНИЕ А. ИСХОДНЫЙ КОД

Файл main.asm

```
AStack SEGMENT STACK
DW 512 DUP(?)
AStack ENDS

DATA SEGMENT

KEEP_CS DW 0 ; для хранения сегмента
KEEP_IP DW 0 ; и смещения вектора прерывания
keep_sp dw 0
keep_ss dw 0
keep_ax dw 0
new_st dw 10 dup (?)
msg DB 'hello$'
timer DW 0

DATA ENDS

CODE SEGMENT

ASSUME SS:AStack, DS:DATA, CS:Code

;обработчик 08h
new_08h PROC FAR

    mov keep_sp, sp
    mov keep_ss, ss
    mov keep_ax, ax
    mov ax, new_st
    mov ss, ax
    mov ax, keep_ax

    dec timer
    mov al,20h
    out 20h,al

    mov ss, keep_ss
    mov sp, keep_sp
    iret
new_08h 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, 08h ; номер вектора
int 21h
mov KEEP_IP, bx ; запоминание смещения
mov KEEP_CS, es ; и сегмента вектора прерывания
push ds
mov dx, offset new_08h ; смещение для процедуры в DX
mov ax, seg new_08h ; сегмент процедуры
mov ds, ax ; помещаем в DS
mov ah, 25h ; функция установки вектора
mov al, 08h ; номер вектора
int 21h
pop ds

```

```

;вывод сообщения с задержкой
mov dx, offset msg
mov timer, 15h
delay:
cmp timer, 0
jge delay;
mov ah, 9
int 21h

```

```

;восстановление старого вектора прерывания
cli
push DS
mov dx, KEEP_IP
mov ax, KEEP_CS
mov ds, ax
mov ah, 25h
mov al, 08h
int 21h
pop ds
sti
ret

```

MAIN ENDP

CODE ENDS

END MAIN