**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: **Написание собственного прерывания**

|  |  |  |
| --- | --- | --- |
| Студент гр. 9383 |  | Поплавский И. |
| Преподаватель |  | Ефремов М.А |

Санкт-Петербург

2020

**Формулировка задания:**

Шифр задания 2А

Написание собственного прерывания.

60h – прерывание пользователя, должно генерироваться в программе;

А - Печать сообщения на экране

**Теория**

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. В этом случае программа должна содержать следующие инструкции:

Для установки написанного прерывания в поле векторов прерываний используется функция 25H прерывания 21H, которая устанавливает вектор прерывания на указанный адрес.

В конце программы восстанавливается старый вектор прерывания.

**Исходный код:**

Stk SEGMENT STACK

DB 256 DUP(?)

Stk ENDS

DATA SEGMENT

KEEP\_CS DW 0 ; для хранения сегмента

KEEP\_IP DW 0 ; и смещения прерывания

Message2 db 'prerivanie',10,13,'$' ;строка для сообщения

DATA ENDS

CODE SEGMENT

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

subr\_int proc far ;начало процедуры

push ax ;сохраняем все изменяемые регистры

push dx ;сохраняем все изменяемые регистры

mov ah,9h ;функция установки вектора

mov dx,offset message2 ;в dx загружаем адрес сообщения Message2

int 21h ;вывод строки на экран

pop dx ;восстанавливаем регистры

pop ax ;восстанавливаем регистры

mov al,20h

out 20h,al

iret ;конец прерывания

subr\_int endp ;конец процедуры

main proc far

push ds

sub ax,ax

push ax

mov ax,data

mov ds,ax

MOV AH, 35H ; функция получения вектора

MOV AL, 60H ; номер вектора

INT 21H

MOV KEEP\_IP, BX ; запоминание смещения

MOV KEEP\_CS, ES ; и сегмента

push ds

mov dx,offset subr\_int

mov ax,seg subr\_int ;сегмент процедуры

mov ds,ax ;помещаем в ds

mov ah,25h ;функция установки вектора

mov al,60h ;номер вектора

int 21h ;меняем прерывание

pop ds ;восстанавливаем ds

int 60h ;наше прерывание

CLI

PUSH DS

MOV DX, KEEP\_IP

MOV AX, KEEP\_CS

MOV DS, AX

MOV AH, 25H

MOV AL, 60H

INT 21H ; восстанавливаем вектор

POP DS

STI

ret

Main endp

code ends

end Main

#### Тестирование программы:

Запускаем программу, видим, что выполняется прерывание и выводится на экран сообщение.

**Вывод:**

В результате выполнения работы был получен опыт написания своего прерывания, использовать функции получения и установки вектора прерывания.