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. 0382 _____ Ильин Д.А.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2021

Цель работы.

Узнать как работает механизм прерываний в ассемблере. Научится писать собственные прерывания.

Задание.

Вариант 1G

Назначение заменяемого вектора прерывания:

1 - 08h - прерывание от системного таймера - генерируется автоматически операционной системой 18 раз в сек;

Действие, реализуемые программой обработки прерываний:

G - Выполнить ввод и печать заданного количества символов, после чего вывести сообщение о завершении обработчика.

Выполнение работы.

В сегменте данных выделяем память для запоминания адреса старого прерывания (чтобы потом вернуть все как было), и нужные сообщения для вывода.

Выделяем стек.

Дальше идет реализация собственного прерывания:

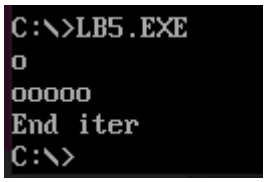
- Закидываем в стек все регистры которые будут изменятся в течении процедуры.
- С помощью прерывания 02h и условных переходов печатаю символ, который хранится в AL, столько раз сколько нужно.(в BX должно хранится сколько раз выводить сообщение)
- Затем с помощью loop делаю задержку и с помощью прерывания 09h вывожу сообщение окончания выполнения прерывания.
- И восстанавливаю из стека регистра.

Главная процедура:,

- Запоминаем адрес прерывания , котрое хотим заменить
- Устанавливаем вместо старого прерывания новое.
- Считываем символ.
- Вызываем новое прерывание, указывая в BX сколько раз напечатать сообщение, и в DX адрес на сообщение.
- Восстанавливаем старое прерывание

Тестирование.

Вызов нового прерывания
Результат:



Выводы.

Был изучен механизм прерываний в языке ассемблер.

В ходе данной лабораторной работы была разработана программа , которая создает свое собственное прерывание, которое выполнить вывод сообщения на экран заданное число раз, после чего вставить фиксированную задержку и вывести сообщение о завершении обработчика.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lb5.asm

```
ind EQU 5

DATA SEGMENT
    keep_cs dw 0 ;to store the old interrupt
    keep_ip dw 0 ;to store the old interrupt
    simbol db '0'
    mes_end_iter DB 10,13,'End iter$'
DATA ENDS

AStack SEGMENT STACK
    DW 1024  DUP(?)
AStack ENDS

CODE SEGMENT
    ASSUME CS:CODE, DS:DATA, SS:AStack

SUBR_INT PROC FAR
    jmp next
     KEEP_SS DW 0
     KEEP_SP DW 0
     MyStack DW 100 dup(0)
next:
     mov KEEP_SP, SP
     mov KEEP_SS, SS
     mov AX, SEG MyStack
     mov SS, SP
     mov SP, offset next
     push ax
     push dx

print:
     mov dl, simbol
     mov ah, 02h ;вывод строки
     int 21h
     sub bx, 1
     cmp bx, 0
     JNE final

     mov ah, 09h
     mov dx, offset mes_end_iter
     int 21h
final:

     pop ax
     pop dx
     pop cx
     mov SS, KEEP_SS
     mov SP, KEEP_SP
        mov AL, 20h
```

```
        out 20h,AL
        iret
SUBR_INT ENDP

Main PROC FAR
    push ds
    sub ax, ax
    push ax
    mov ax, DATA
    mov ds, ax


    mov ah, 01h
    int 21h
    mov simbol, al

    mov dl, 10
    mov ah, 02h ;вывод строки
    int 21h

    ;remember the old interrupt
    MOV AH, 35H ; function of getting interrupt vector
    MOV AL, 1CH ; number of vector
    INT 21H
    MOV KEEP_IP, BX ; remember offset
    MOV KEEP_CS, ES ; and segment of interrupt vector

    ;call interrupt
     mov bx, 5 ; ª®«-¢® ¯®â®à  á®®¡é¥¨ï
    int 08h

    ;set a new interrupt
    PUSH DS
    MOV DX, offset SUBR_INT ; offset for procedure into DX
    MOV AX, seg SUBR_INT ; segment of procedure
    MOV DS, AX ; move to DS
    MOV AH, 25H ; function of setting new vector
    mov al, 08h
    INT 21H ; change interrupt
    POP DS

aaaaaaa:
    cmp bx, 0
    JNE aaaaaaa

    ;restore the old interrupt
    CLI
    PUSH DS
    MOV DX, keep_ip
    MOV AX, keep_cs
    MOV DS, AX
    MOV AH, 25H
    MOV AL, 08H
    INT 21H ; restore the old interrupt vector
    POP DS
```

```
    STI

    ret
Main ENDP

CODE ENDS
     END Main
```