

**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. 1303

Карагезов С.Ю.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2022

Цель работы.

Написать собственное прерывание, согласно варианту задания.

Задание.

Прерывание - это процесс вызова процедур для выполнения некоторой задачи, обычно связанной с обслуживанием некоторых устройств (обработка сигнала таймера, нажатия клавиши и т.д.). Написать собственное прерывание согласно варианту 11 — 2d:

2 – 60h – прерывание пользователя – должно генерироваться в программе;

d – Выполнить чтение и вывод на экран отсчета системных часов (в тиках, где 1 тик = 55 мсек).

Выполнение работы.

В сегменте данных DATA содержатся переменные: MEM_CS, MEM_IP для хранения сегмента и смещения старого прерывания соответственно, также выделяется память в сегменте AStack.

В процедуре Main сохраняется смещение и сегмент текущего прерывания 60h в MEM_IP, MEM_CS с помощью функции 35h прерывания 21h. Используя функцию 25h прерывания 21h, устанавливается вектор прерывания 60h на созданное прерывание Time. Затем происходит его вызов. Когда его работа будет завершена – восстанавливается старый вектор прерывания 60h.

Рассмотрим написанную процедуру пользовательского прерывания Time. Выделяется стек для прерывания и сохраняется смещение на основной стек, а также сохраняются все изменяемые регистры в стеке. В регистр AH перемещается значение 00h (функция чтения часов — счетчик тиков — прерывания 1Ah), и вызывается прерывание 1Ah. После ее выполнения в регистрах CX (старшая часть значения) и DX записано время. Для конвертирования времени в строку и ее вывода создана процедура Its. Она вызывается два раза, перед первым вызовом в регистр AX помещается значение регистра CX, перед вторым разом — DX. После ее выполнения

восстанавливаются сохраненные регистры SS и SP, в регистр AL помещается значение 20h, завершается прерывание вызовом команды out 20h,AL и iret.

Процедура Its сохраняет используемые регистры, обнуляется регистр CX, а в BX помещается значение 10 (делитель). Далее пока AX не будет равен 0, он делится на BX (равен 10), а остаток от деления кладется на стек, увеличивая CX на один. Таким образом после выполнения этих действий на стеке хранятся все символы числа в виде слов, а также известно их количество. С помощью команды loop программа переходит на метку print CX раз и выводит символ на экран с помощью функции 02h прерывания int 21h.

Исходный код программы см. в приложении А.

Тестирование.

Для проверки работоспособности программы проведены тесты, результаты представлены в таблице 1.

Таблица 1 — Тестирование и результаты

Номер теста	Входные данные	Выходные данные	Корректность результата
1	нет		корректно
2	нет		корректно

Вывод.

В ходе работы были изучены прерывания. Также было написано собственное прерывание по чтению и выводу системного времени на экран.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lb5.asm

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

DATA SEGMENT
    MEM_CS DW 0
    MEM_IP DW 0
DATA ENDS

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

Its PROC
    push AX ; сохранение регистров
    push DX
    push BX
    push CX

    xor CX, CX ; обнуление CX для хранения кол-ва символов
    mov BX, 10 ; делитель 10
division:
    xor DX, DX ; обнуление DX
    div BX ; деление AX = (DX, AX)/BX, остаток в DX
    add DL, '0' ; перевод цифры в символ
    push DX ; сохранение остатка на стек
    inc CX ; увеличить счетчик
    test AX, AX ; проверка AX
    jnz division; если частное не 0, то повторяем
    mov ah, 02h

print:
    pop DX ; достать символ из стека CX раз
    int 21h
    loop print ; пока cx != 0 выполнить переход

    pop CX ; вернуть значения со стека
    pop BX
```

```

    pop DX
    pop AX
    ret
Its endp

Time PROC FAR
    jmp time
    KEEP_SS DW 0
    KEEP_SP DW 0
    Stack DB 50 dup(" ")
time:
    mov KEEP_SS, SS
    mov KEEP_SP, SP
    mov SP, SEG Stack
    mov SS, SP
    mov SP, offset time

    push AX      ; сохранение изменяемых регистров
    push CX
    push DX

    mov AH, 00h ; читать часы (счетчик тиков)
    int 1Ah ; CX,DX = счетчик тиков

    mov AX, CX
    call IntToStr
    mov AX, DX
    call IntToStr

    pop DX
    pop CX
    pop AX      ; восстановление регистров

    mov SS, KEEP_SS
    mov SP, KEEP_SP

    mov AL, 20H
    out 20H,AL
    iret
Time ENDP

Main PROC FAR
    push DS
    sub AX,AX
    push AX
    mov AX, DATA
    mov DS, AX

    mov AH,35h ; дать вектор прерывания
    mov AL,60h ; номер вектора

```

```

int 21h      ; вызов -> выход: ES:BX = адрес обработчика прерывания
mov KEEP_IP, BX ; запоминание смещения
mov KEEP_CS, ES ; запоминание сегмента

push DS
mov DX, offset GetTime ; смещение для процедуры
mov AX, seg GetTime ; сегмент процедуры
mov DS, AX
mov AH, 25h      ; функция установки вектора
mov AL, 60h      ; номер вектора
int 21h          ; установить вектор прерывания на указанный адрес нового
обработчика
pop DS

int 60h ; вызов прерывания пользователя

CLI      ; сбрасывает флаг прерывания IF
push DS
mov DX, KEEP_IP
mov AX, KEEP_CS
mov DS, AX
mov AH, 25h
mov AL, 60h
int 21h
pop DS
STI

ret
Main ENDP
CODE ENDS
END Main

```