

**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

Иванов А. С.

Преподаватель

Ефремов М. А.

Санкт-Петербург  
2022

## **Цель работы.**

Написать собственное прерывание, согласно варианту задания.

## **Задание.**

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.**

Создадим в сегменте данных: temp\_cs, temp\_ip для временного хранения сегмента и смещения старого прерывания соответственно.

В головной процедуре сохраняется смещение и сегмент текущего прерывания 60h в temp\_cs, temp\_ip с помощью функции 35h прерывания 21h. Используя функцию 25h прерывания 21h, устанавливается вектор прерывания 60h на созданное прерывание get\_time. Затем происходит его вызов. Когда его работа будет завершена – старый вектор прерывания 60h восстанавливается.

Процедура get\_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 int\_to\_string. 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 int\_to\_string: 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 так же известно их количество.

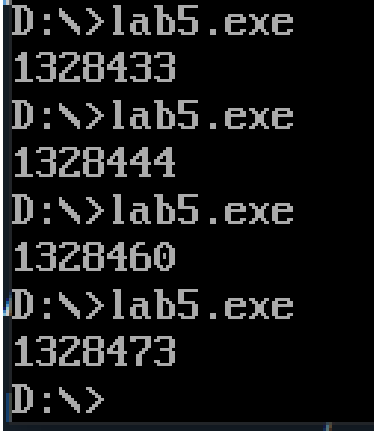
С помощью команды loop программа переходит на метку print CX раз и выводит символ на экран с помощью функции 02h прерывания int 21h.

Исходный код программы см. в приложении А.

### **Тестирование.**

Для проверки работоспособности программы проведены тесты, результаты представлены на рис 1.

Рисунок 1 — Тестирование и результаты



```
D:\>lab5.exe
1328433
D:\>lab5.exe
1328444
D:\>lab5.exe
1328460
D:\>lab5.exe
1328473
D:\>
```

### **Вывод.**

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

## **ПРИЛОЖЕНИЕ А ИСХОДНЫЙ КОД ПРОГРАММ**

Название файла: lb5.asm

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

DATA SEGMENT
    temp_cs DW 0
    temp_ip DW 0
DATA ENDS

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

int_to_string PROC
    push AX
    push DX
```

```

    push BX
    push CX

    xor CX, CX
    mov BX, 10 ; делитель 10
divide:
    xor DX, DX ; обнуление DX
    div BX
    add DL, '0' ; перевод цифры в символ
    push DX
    inc CX
    test AX, AX
    jnz divide; если частное не 0, то повторяем
    mov ah, 02h

console_log:
    pop DX
    int 21h
    loop console_log ; пока cx != 0 выполнить
переход
    pop CX
    pop BX
    pop DX
    pop AX
    ret
int_to_string endp

get_time PROC FAR
    jmp time
temp_ss DW 0
temp_sp DW 0
Stack DB 50 dup(" ")
time:
    mov temp_ss, SS
    mov temp_sp, SP
    mov SP, SEG Stack
    mov SS, SP
    mov SP, offset time
    push AX
    push CX
    push DX
    mov AH, 00h ; считать часы
    int 1Ah ; CX,DX = счетчик тиков
    mov AX, CX
    call int_to_string
    mov AX, DX
    call int_to_string
    pop DX
    pop CX
    pop AX
    mov SS, temp_ss
    mov SP, temp_sp
    mov AL, 20H
    out 20H,AL
    iret

```

```
get_time ENDP
```

```
Main PROC FAR
push DS
sub AX,AX
push AX
mov AX, DATA
mov DS, AX
mov AH,35h
mov AL,60h
int 21h
mov temp_ip, BX
mov temp_cs, ES
push DS
mov DX, offset get_time
mov AX, seg get_time
mov DS, AX
mov AH, 25h
mov AL, 60h
int 21h
pop DS
int 60h      ; вызов прерывания пользователя
CLI
push DS
mov DX, temp_ip
mov AX, temp_cs
mov DS, AX
mov AH, 25h
mov AL, 60h
int 21h
pop DS
STI
ret
Main ENDP
CODE ENDS
END Main
```