# 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. 1384 Усачева Д.В.

Преподаватель Кирьянчиков В. А.

Санкт-Петербург 2022

# Цель работы.

# Написать собственное прерывание в соответствии с вариантом на языке Ассемблер.

# Задание.

# Вариант 4e.

4 - 16h - прерывание от клавиатуры (по заданному скан-коду клавиши делать

действия A-G, указанные ниже).

E - Выполнить чтение и вывод на экран отсчета часов реального времени из

памяти CMOS (в формате BCD)

**Выполнение работы.**

В Main загружается в DS адрес начала сегмента данных, сохраняется сегмент и смещение текущего прерывания с помощью функции 35 прерывания 21H. Функция 25H прерывания 21H устанавливает вектор прерывания 16H, после чего вызывается созданное прерывание. После всех процедур необходимо восстановить старый вектор прерывания.

Были написаны процедуры для получения, преобразования и вывода времени на экран с разделителем «:», G\_TIME, W\_TIME, SEP соответственно.

SUBR\_INT – процедура прерывания. Все изменяемые переменные сохраняются в стек. Прерывание SUBR\_INT считывает реальное время с помощью процедуры G\_TIME в BCD-формате.

Далее при вводе пользователем «esc» или «enter» мы выходим из программы, если же пользователь введет «e», то цифры времени преобразуются в их ASCII-коды и выведутся на экран в формате ЧЧ:ММ:CC.

# Тестирование.

# 

**Вывод.**

# В ходе выполнения лабораторной работы были получены навыки написания программы с собственным прерыванием на языке Ассемблер.

# Исходный код LR5.ASM

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

STACK SEGMENT STACK

DW 1024 DUP(?)

STACK ENDS

DATA SEGMENT

KEEP\_CS DW 0

KEEP\_IP DW 0

DATA ENDS

CODE SEGMENT

G\_TIME PROC

push dx

push cx

xor cx, cx

mov bx, 10

next:

xor dx,dx

div bx

add dl, '0'

push dx

inc cx

cmp ax, 0

jnz next

mov ah, 02h

print:

pop dx

int 21h

loop print ;уменьшает значение в регистре СХ, если СХ не равно нулю то переход

pop cx

pop dx

ret

G\_TIME ENDP

SEP PROC

;разделитель

mov dl, ':'

mov ah, 2h

int 21h

ret

SEP ENDP

W\_TIME proc

push dx

aam ;переводим время в аски коды

add ax, 3030h

mov dl, ah

mov dh, al

mov ah, 2h

int 21h

mov dl, dh

int 21h

pop dx

ret

W\_TIME endp

SUBR\_INT proc far

jmp start

keep\_sp DW 0000h

keep\_ss DW 0000h

int\_stack db 50 dup(0)

start:

mov keep\_sp, sp

mov keep\_ss, ss

mov sp, seg int\_stack

mov ss, sp

mov sp, offset start

push ax

push cx

push dx

mov ah, 00h

int 1ah

mov ax, cx

call G\_TIME

mov ax, dx

call G\_TIME

pop dx

pop cx

pop ax

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

;вставка 1

MOV AH, 35H ; функция получения вектора

MOV AL, 16 ; номер вектора

INT 21H

MOV KEEP\_IP, BX ; запоминание смещения

MOV KEEP\_CS, ES ; и сегмента

check:

mov ah, 0

int 16h

cmp ah, 12h ;проверяем на е

je inp\_e

;проверяем на esc и enter для выхода

cmp ah, 1ch

je stop

cmp ah, 1h

je stop

jnz check

inp\_e:

;если ввели е,то выводим

mov ah, 2ch

int 21H

;часы

mov al, ch

call W\_TIME

call SEP

;минуты

mov al, cl

call W\_TIME

call SEP

;секунды

mov al, dh

call W\_TIME

int 16H

;вставка 3

CLI

PUSH DS

MOV DX, KEEP\_IP

MOV AX, KEEP\_CS

MOV DS, AX

MOV AH, 25H

MOV AL, 16H

INT 21H ; восстанавливаем вектор

POP DS

STI

stop:

MOV AH, 4Ch

INT 21h

Main ENDP

CODE ENDS

END Main