Министерство науки и высшего образования Российской Федерации  
Пензенский государственный университет  
Кафедра вычислительная техника

**ОТЧЕТ**

по лабораторной работе №5

по дисциплине «ЭВМ и переферийные устройства»

на тему «Работа процессора при прерываниях»

Выполнили:

студенты группы 22ВВП1

Хоссейни Нежад С.А.С.М.

Захаров А. С.

Приняли:

Никишин К.И.

Патунин Д.В.

Пенза 2025

**Цель работы**

Изучение особенностей выполнения команд программных прерываний.

**Задание**

Составить программу, выполняющую обработку массива, аналогично заданию предыдущей работы, но вызываемую с использование аппарата прерваний, **праметры передавать через регистры**.

****

**Ход работы**

**Задание 1**

**Листинг:**

stseg segment para stack

dw 16 dup(?)

stseg ends

dseg segment para

x dw 1, 0, 1, 2, 3, 0, 4, 5

y dw ?

old\_adress dw 2 dup(?) ; Для хранения старого вектора прерывания

dseg ends

cseg segment para

main proc far

assume cs:cseg, ds:dseg, ss:stseg

push ds

mov ax, 0

push ax

push es

mov ax, dseg

mov ds, ax

; Сохраняем старый обработчик прерывания

mov ah, 35h ; Получаем адрес прерывания 00h

mov al, 0FEh

int 21h

mov old\_adress, es

mov old\_adress+2, bx

push ds

; Устанавливаем новый обработчик прерывания

mov dx, offset is\_there\_null

mov ax, seg is\_there\_null

mov ds, ax

mov ah, 25h ; Устанавливаем обработчик

mov al, 0FEh

int 21h

pop ds

; Подготовка параметров для вызова прерывания

mov bx, offset x ; Адрес массива

mov cx, 8 ; Количество элементов

mov ax, 0

mov y, ax

int 0FEh ; Вызов нашего обработчика

; Восстанавливаем старый обработчик прерывания

push ds

mov dx, old\_adress+2

mov ax, old\_adress

mov ds, ax

mov ah, 25h

mov al, 0FEh

int 21h

pop ds

pop es

ret

main endp

is\_there\_null proc far

push ax

push bx

push cx

push dx

push ds

mov ax, dseg

mov ds, ax

loop\_start:

mov dx, [bx] ; Загружаем текущий элемент массива

cmp dx, 0

je set\_y\_to\_one ; Если элемент равен 0, переходим в set\_y\_to\_one

add bx, 2 ; Переход к следующему элементу

loop loop\_start ; Уменьшаем CX и повторяем цикл

jmp end\_isr

set\_y\_to\_one:

mov ax, 1

mov y, ax ; Присваиваем y значение 1

end\_isr:

pop ds

pop dx

pop cx

pop bx

pop ax

iret ; Возврат из прерывания

is\_there\_null endp

cseg ends

end main

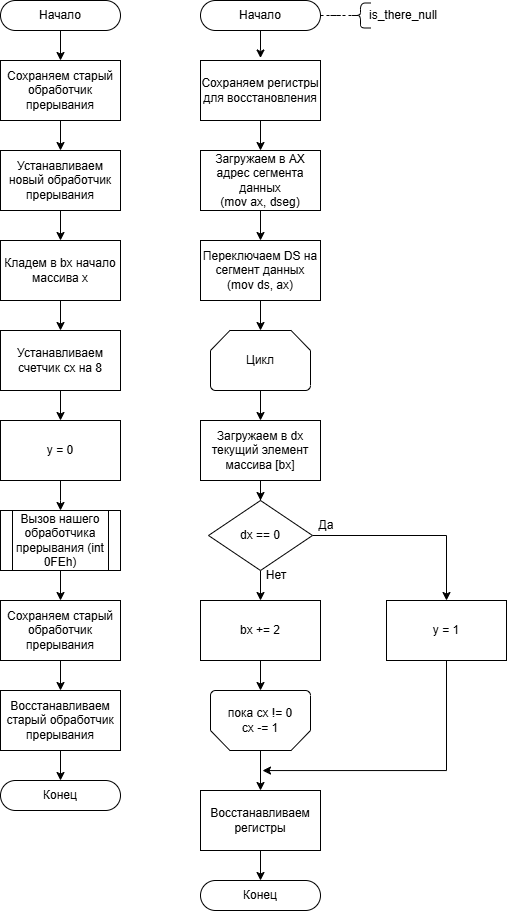
**Отладка**

До вызова процедуры

После входа в процедуру

После возврата из процедуры

**Блок-схема**

****

**Вывод**

Изучили особенности выполнения команд программных прерываний.