Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра электронных вычислительных машин

Лабораторная работа №2

«Программирование контроллера прерываний»

Проверил: Выполнил:

к. т. н., доцент студент гр. 150501

Одинец Д. Н. Кардаш С. П.

Минск, 2023

1 ПОСТАНОВКА ЗАДАЧИ

Написать резидентную программу выполняющую перенос всех векторов

аппаратных прерываний ведущего и ведомого контроллера на пользовательские прерывания. При этом необходимо написать обработчики аппаратных прерываний, которые будут установлены на используемые пользовательские прерывания и будут выполнять следующие функции:

1. Выводить на экран в двоичной форме следующие регистры контроллеров прерывания (как ведущего, так и ведомого):

* регистр запросов на прерывания;
* регистр обслуживаемых прерываний;
* регистр масок.

При этом значения регистров должны выводиться всегда в одно и то же место экрана. Осуществлять переход на стандартные обработчики аппаратных прерываний, для обеспечения нормальной работы компьютера.

2 АЛГОРИТМ

2.1Структура программы

Программа состоит из нескольких процедур:

IRQ0-IRQ15 – обработчики аппаратных прерываний(резидентная часть). setInterrupts – процедура установки обработчиков аппаратных прерываний. INT50 – прерывание, выводящее значения регистров контроллера прерываний

**2.2 Описание переменных**

registers – массив, содержащий значения регистров ведущего и ведомого контроллера. temp – буфер для обработки и вывода значение регистров

**2.3Описание алгоритма программы**

Главная процедура позволяет выбрать требуемы пользователю действия. Если пользователь выбрал 1 пункт, то вызывается процедура setInterrupts. В ней программа переинициализирует контроллер прерываний на новые вектора прерываний и оставляет эти обработчики резидентными. Эти обработчики впоследствии вызывают обработчик 50h прерывания, который выводит значения регистров контроллера прерываний, а затем вызывает стандартный обработчик прерывания **3 КОД ПРОГРАММЫ**

org 100h

.model tiny

.stack 100h

.data

msgHello db "1 - start resident",10,13,"$"

BLACK equ 01110000b ;text atribute - black letters on white font

RED equ 01110100b ;text atribute - red letters on white font

.code

mov ah,09h ;output message

lea dx,msgHello

int 21h

mov ah,07h ;wait for number 1

int 21h

cmp al,'1'

je setInterrupts ;begin programm

;set handler for all apparat interrapts

IRQ0: ;IRQ0 - system clock interrapt

int 50h

int 60h

iret

IRQ1: ;IPQ1 - keyboard interrapt

int 50h

int 61h

iret

IRQ2: ;IRQ2 - second interrapt controller

int 50h

int 62h

iret

IRQ3: ;IRQ3 - com2

int 50h

int 63h

iret

IRQ4: ;IRQ4 - com1

int 50h

int 64h

iret

IRQ5: ;IRQ5 - hard drive interrapt

int 50h

int 65h

iret

IRQ6: ;IRQ6 - floppy interrapt

int 50h

int 66h

iret

IRQ7: ;IRQ7 - printer interrapt

int 50h

int 67h

iret

IRQ8: ;IRQ8 - RTC

int 50h

int 70h

iret

IRQ9: ;IRQ9 - EGA controller interrapt

int 50h

int 71h

iret

IRQ10: ;IRQ10 - reserved

int 50h

int 72h

iret

IRQ11: ;IRQ11 - reserved

int 50h

int 73h

iret

IRQ12: ;IRQ12 - reserved

int 50h

int 74h

iret

IRQ13: ;IRQ13 - math pcu

int 50h

int 75h

iret

IRQ14: ;IRQ14 - hard drive controller

int 50h

int 76h

iret

IRQ15: ;IRQ15 - reserved

int 50h

int 77h

iret

;finish setting handlers

DISPLAY: ;INT15 - handler for output interrapt controller registrs

pusha

push es

mov ax,0B800h ;adress of vidio memory

mov es,ax ;write it in ax

mov cx,8

xor di,di

mov di,0

xor bx,bx ;output IRQ on the screen

bufferOutput:

mov al,cs:[buffer+bx]

mov es:[di],al

inc bx

inc di

mov al,BLACK

mov es:[di],al

inc di

loop bufferOutput

;;;;;;

mov al, 10011000b ;set mask for first conroller

mov dx, 21h

out dx, al

;;;;;;

in al,21h ;get mask registr of first controller

mov cs:[registers+0],al

in al,0A1h ;get mask registr of second controller

mov cs:[registers+3],al

mov al,00001011b ;OCW3

out 20h,al ;0XXXXXXX - it is not used

in al,20h ;X00XXXXX - do not change mask mode

mov cs:[registers+2],al ;XXX01XXX - OCW3

;XXXXX0XX - not a question mode

;XXXXXX11 - read regist with interrapts being served

mov al,00001011b ;Команда OCW3

out 0A0h,al ;0XXXXXXX - it is not used

in al,0A0h ;X00XXXXX - do not change mask mode

mov cs:[registers+5],al ;XXX01XXX - OCW3

;XXXXX0XX - not a question mode

;XXXXXX11 - read regist with interrapts being served

mov al,00001010b ;OCW3

out 20h,al ;0XXXXXXX - it is not used

in al,20h ;X00XXXXX - do not change mask mode

mov cs:[registers+1],al ;XXX01XXX - OCW3

;XXXXX0XX - not a question mode

;XXXXXX11 - read regist with requested interrapts

mov al,00001010b ;Команда OCW3

out 0A0h,al ;0XXXXXXX - it is not used

in al,0A0h ;X00XXXXX - do not change mask mode

mov cs:[registers+4],al ;XXX01XXX - OCW3

;XXXXX0XX - not a question mode

;XXXXXX11 - read regist with requested interrapts

mov cx,6 ;output registers on the screen

registersOutput:

mov dx,cx

mov bx,6

sub bx,dx

mov al,cs:[registers+bx];register value from bufer

lea bx,cs:[temp] ;buffer

push cx

mov cx,8

btbs\_lp: ;output every byte

rol al,1 ;cicle move to left

jc btbs\_1 ;carry flag = 1 than write 1

mov cs:[bx],'0' ;else'0' to temp

jmp btbs\_end

btbs\_1:

mov cs:[bx],'1' ;write '1' to buffer temp

btbs\_end:

inc bx ;go to next byte

loop btbs\_lp ;get temp

mov cx,8

xor bx,bx

registerOutput: ;output temp

mov al,cs:[temp+bx]

mov es:[di],al

inc di

inc bx

mov al,RED

mov es:[di],al

inc di

loop registerOutput

mov al,' ' ;" " beetwin registers

mov es:[di],al

inc di

mov al,RED

mov es:[di],al

inc di

pop cx

loop registersOutput

pop es

popa

iret ;finish display handler

buffer db "IRQ: "

temp db 8 dup ('6')

registers db 6 dup (?)

setInterrupts: ;set vectors of interrapts

mov ah,0h ;set video mode 80\*25

mov al,3h

int 10h

push ds

push cs

cli

mov ah, 35h ;get adress of apparat handler

mov al, 08h ;number of vector

int 21h

mov ah, 25h ;set adress of apparat handler

mov al, 60h ;new number of vector

push es

pop ds

mov dx, bx

int 21h

mov ah, 35h

mov al, 09h

int 21h

mov ah, 25h

mov al, 61h

push es

pop ds

mov dx, bx

int 21h

mov ah, 35h

mov al, 0ah

int 21h

mov ah, 25h

mov al, 62h

push es

pop ds

mov dx, bx

int 21h

mov ah, 35h

mov al, 0bh

int 21h

mov ah, 25h

mov al, 63h

push es

pop ds

mov dx, bx

int 21h

mov ah, 35h

mov al, 0ch

int 21h

mov ah, 25h

mov al, 64h

push es

pop ds

mov dx, bx

int 21h

mov ah, 35h

mov al, 0dh

int 21h

mov ah, 25h

mov al, 65h

push es

pop ds

mov dx, bx

int 21h

mov ah, 35h

mov al, 0eh

int 21h

mov ah, 25h

mov al, 66h

push es

pop ds

mov dx, bx

int 21h

mov ah, 35h

mov al, 0fh

int 21h

mov ah, 25h

mov al, 67h

push es

pop ds

mov dx, bx

int 21h

sti

pop ds

mov ax,25a0h ;set interrapts vectors a0h-a7h

mov dx, offset IRQ0

int 21h

mov ax,25a1h

mov dx, offset IRQ1

int 21h

mov ax,25a2h

mov dx, offset IRQ2

int 21h

mov ax,25a3h

mov dx, offset IRQ3

int 21h

mov ax,25a4h

mov dx, offset IRQ4

int 21h

mov ax,25a5h

mov dx, offset IRQ5

int 21h

mov ax,25a6h

mov dx, offset IRQ6

int 21h

mov ax,25a7h

mov dx, offset IRQ7

int 21h

mov ax,2508h ;set interrapts vectors 08h-0fh

mov dx, offset IRQ8

int 21h

mov ax,2509h

mov dx, offset IRQ9

int 21h

mov ax,250Ah

mov dx, offset IRQ10

int 21h

mov ax,250Bh

mov dx, offset IRQ11

int 21h

mov ax,250Ch

mov dx, offset IRQ12

int 21h

mov ax,250Dh

mov dx, offset IRQ13

int 21h

mov ax,250Eh

mov dx, offset IRQ14

int 21h

mov ax,250Fh

mov dx, offset IRQ15

int 21h

mov ax,2550h

mov dx, offset DISPLAY

int 21h

cli ;not permit apparat interrapt

mov al, 11h ;ICW1 - begin of initializasion of controllers

out 20h, al ;0001XXXX - ICW1

out 0A0h, al ;XXXX0XXX - when the signal is different

;XXXXX1XX -

;XXXXXX0X - cascade connection

;XXXXXXX0 - will be ICW4

mov al, 0a0h ;ICW2 - base numbers of interrapts in table

out 21h, al

mov al, 08h ;ICW2 - base numbers of interrapts in table

out 0A1h, al

mov al, 04h ;ICW3 - veduschiy controller connected to the third line

out 21h, al

mov al, 02h ;ICW3 - vedomiy controller connected to the third line

out 0A1h, al

mov al, 11h ;ICW4 - mode vlojennosty

out 21h, al

mov al,01h ;ICW4 - mode vlojennosty

out 0A1h, al

sti

lea dx,setInterrupts ;set resident programm

int 27h

ret

ends code

**4 ТЕСТИРОВАНИЕ**

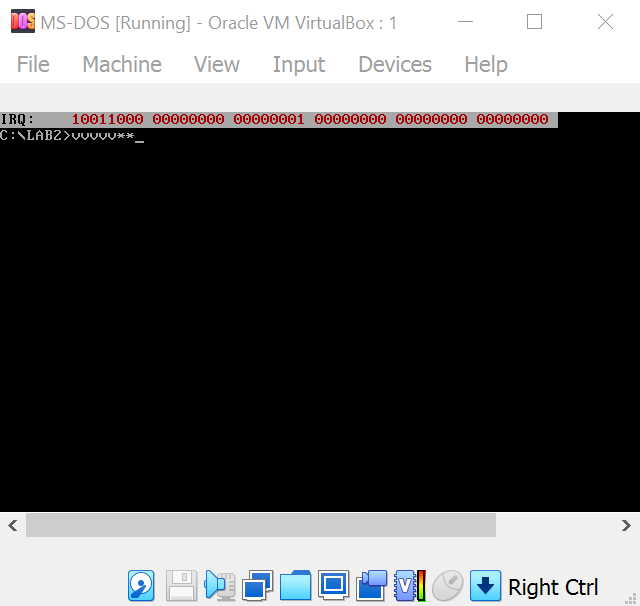


Рисунок 4.1 – Резидентная программа в процессе исполнения

**4 ЗАКЛЮЧЕНИЕ**

В ходе лабораторной работы была разработана резидентная программа выполняющая перенос всех векторов аппаратных прерываний ведущего и ведомого контроллера на пользовательские прерывания. А также в программе был реализован побитовый вывод регистров контроллера прерываний.

Для запуска программы использовалась операционная система Ms-Dos установленная на VirtualBox на операционной системе Windows 10.