**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: Разработка собственного прерывания

|  |  |  |
| --- | --- | --- |
| Студент гр. 8383 |  | Ларин А. |
| Преподаватель |  | Ефремов М.А, |

Санкт-Петербург

2019

**Цель работы.**

Научится реализовывать собственные обработчики прерываний, и замещать ими стандартные. Написать процедуру согласно заданию, и обеспечить ее вызов по прерыванию.

**Основные теоретические положения.**

Прерывание - это процесс вызова процедур для выполнения некоторой задачи, обычно связанной с обслуживанием некоторых устройств (обработка сигнала таймера, нажатия клавиши и т.д.). Когда возникает прерывание, процессор прекращает выполнение текущей программы (если ее приоритет ниже) и запоминает в стеке вместе с регистром флагов адрес возврата(CS:IP) - места, с которого будет продолжена прерванная программа. Затем в CS:IP загружается адрес программы обработки прерывания и ей передается управление. Адреса 256 программ обработки прерываний, так называемые векторы прерывания, имеют длину по 4 байта (в первых двух хранится значение IP , во вторых - CS) и хранятся в младших 1024 байтах памяти. Программа обработки прерывания должна заканчиваться инструкцией IRET (возврат из прерывания), по которой из стека восстанавливается адрес возврата и регистр флагов.

Программа обработки прерывания - это отдельная процедура, имеющая структуру:

SUBR\_INT PROC FAR

PUSH AX ; сохранение изменяемых регистров

...

<действия по обработке прерывания>

POP AX ; восстановление регистров

...

MOV AL, 20H

OUT 20H,AL

IRET

SUBR\_INT ENDP

Две последние строки обработчика прерывания, указанные перед командой IRET выхода из прерывания, необходимы для разрешения обработки прерываний с более низкими уровнями, чем только что обработанное.

Программа, использующая новые программы обработки прерываний при своем завершении должна восстанавливать оригинальные векторы прерываний. Функция 35 прерывания 21H возвращает текущее значение вектора прерывания, помещая значение сегмента в ES, а смещение в BX. В соответствии с этим, программа должна содержать следующие инструкции:

; -- в сегменте данных

KEEP\_CS DW 0 ; для хранения сегмента

KEEP\_IP DW 0 ; и смещения вектора прерывания

; -- в начале программы

MOV AH, 35H ; функция получения вектора

MOV AL, 1CH ; номер вектора

INT 21H

MOV KEEP\_IP, BX ; запоминание смещения

MOV KEEP\_CS, ES ; и сегмента вектора прерывания

Для установки адреса нового обработчика прерывания в поле векторов прерываний используется функция 25H прерывания 21H, которая помещает заданные адреса сегмента и смещения обработчика в вектор прерывания с заданным номером.

PUSH DS

MOV DX, OFFSET ROUT ; смещение для процедуры в DX

MOV AX, SEG ROUT ; сегмент процедуры

MOV DS, AX ; помещаем в DS

MOV AH, 25H ; функция установки вектора

MOV AL, 60H ; номер вектора

INT 21H ; меняем прерывание

POP DS

Далее может выполняться вызов нового обработчика прерывания.

В конце программы восстанавливается старый вектор прерывания

CLI

PUSH DS

MOV DX, KEEP\_IP

MOV AX, KEEP\_CS

MOV DS, AX

MOV AH, 25H

MOV AL, 1CH

INT 21H ; восстанавливаем старый вектор прерывания

POP DS

STI

**Задание**

Шифры, определяющие варианты заданий приведены в таблице 6.

Таблица 6

N Шифр N Шифр

бригады задания бригады задания

1 1A 7 3А

2 1B 8 3В

3 1С 9 3С

4 2А 10 4А

5 2В 11 4В

6 2С 12 4C

Цифра в шифре задает номер и назначение заменяемого вектора прерывания:

1 - 1Ch - прерывание от часов - генерируется автоматически операционной системой

18 раз в сек;

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

3 - 23h - прерывание, генерируемое при нажатии клавиш Control+C ;

4 - 08h - прерывание от системного таймера - генерируется автоматически операционной системой 18 раз в сек.

Буква определяет действия, реализуемые программой обработки прерываний:

А - Печать сообщения на экране;

В - Выдача звукового сигнала;

С - Приостановить вывод на экран (вставить цикл задержки).

Замечание: для исключения возможного взаимного влияния системных и пользовательских прерываний рекомендуется отвести в программе под стек не менее 1К байт.

**Выполнение**

Вариант 1

Шифр задания 1A

Требуется вывести сообщение на экран по прерыванию 1Ch 18.2 раз в секунду.

В начале программы мы устанавливаем регистр ds в адрес сегмента данных, так как собираемся сохранять там некоторые значения регистров и сообщение для печати.

Далее происходит получение и сохранение текущего адреса обработчика прерывания таймера.

Далее происходит замена обработчика прерывания на собственный — TMR\_OVR. Для ограничения времени работы программы был использован счетчик TIMER\_CNT, проверяющийся в основной программе в пустом цикле. В то же время в обработчик прерывания уменьшает данных счетчик при каждом вызове. Регистр DX устанавливается на сообщение, которое необходимо вывести.

В конце программы происходит возвращение сохраненного изначального обработчика. В начале данной части программы происходит очистка флага I командой CLI, чтобы запретить вызов обработчика во время его замены. В конце флаг устанавливается обратно командой STI.

Собственный обработчик прерывания TMR\_OVR представляет собой процедуру дальнего вызова. В начале процедуры происходит сохранение регистров, кторые будут изменены, в данном случае только AX. Затем реализуется основной фукционал обработчика — вывод сообщения на экран при помощи программного прерывания 21,9, затем уменьшение счетчика. Далее посылается сигнал - конец прерывания контроллеру прерываний, восстановление регистров и возврат с восстановлением флагов командой IRET.

Код программы приведен в приложении

**Выводы.**

В результате работы были разобраны некоторые базовые концепции языка ассемблера. Был изучен принцип обработки прерываний. Была написана рабочая программа с пользовательской процедурой обработки прерываний, заменившей стандартную, и обратной заменой.

Приложение

LR5.asm

STACKSG SEGMENT PARA STACK 'Stack'

DW 512 DUP(?)

STACKSG ENDS

DATASG SEGMENT PARA 'Data' ;SEG DATA

KEEP\_CS DW 0 ; для хранения сегмента

KEEP\_IP DW 0 ; и смещения вектора прерывания

GREETING DB '\_Larin\_Anton\_8383\_$'

TIMER\_CNT DW 0;

DATASG ENDS ;ENDS DATA

CODE SEGMENT ;SEG CODE

ASSUME DS:DataSG, CS:Code, SS:STACKSG

TMR\_OVR PROC FAR

PUSH AX ; сохранение изменяемых регистров

;<действия по обработке прерывания>

mov AH,9

int 21h

dec TIMER\_CNT;

MOV AL, 20H

OUT 20H,AL

POP AX ; восстановление регистров

IRET

TMR\_OVR ENDP

Main PROC FAR

mov ax, DATASG ;ds setup

mov ds, ax

MOV AH, 35H ; функция получения вектора

MOV AL, 1CH ; номер вектора

INT 21H

MOV KEEP\_IP, BX ; запоминание смещения

MOV KEEP\_CS, ES ; и сегмента вектора прерывания

PUSH DS

MOV DX, OFFSET TMR\_OVR ; смещение для процедуры в DX

MOV AX, SEG TMR\_OVR ; сегмент процедуры

MOV DS, AX ; помещаем в DS

MOV AH, 25H ; функция установки вектора

MOV AL, 1CH ; номер вектора

INT 21H ; меняем прерывание

POP DS

mov DX, OFFSET GREETING

mov TIMER\_CNT,5h;

lop:

cmp TIMER\_CNT,0;

jnz lop;

CLI

PUSH DS

MOV DX, KEEP\_IP

MOV AX, KEEP\_CS

MOV DS, AX

MOV AH, 25H

MOV AL, 1CH

INT 21H ; восстанавливаем старый вектор прерывания

POP DS

STI

mov ah,4Ch;

int 21h;

Main ENDP

CODE ENDS

END Main ;ENDS CODE