

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

отчет
по лабораторной работе №5
по дисциплине «Организация ЭВМ и систем»
Тема: Разработка собственного прерывания.

Студент гр. 9382

Рыжих Р.В.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Изучить основы работы с прерываниями и применить полученные знания на практике.

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

Вариант 2В

Номер и назначение заменяемого вектора прерывания: 2 - 60h - прерывание пользователя - должно генерироваться в программе; Действия, реализуемые программой обработки прерываний: В - Выдача звукового сигнала.

Ход работы:

В функции MAIN сохраняются значения сегмента и смещения для прерывания 60h. После этого происходит вызов функции SUBR_INT для реализации прерывания со звуковым сигналом. Затем обратно записываются сегмент и смещение, сохраненные в переменных программы. RET завершает выполнение функции MAIN. Исходный код программы представлен в приложении А

Тестирование.

Было реализовано собственное прерывание, которое может воспроизводить звук.

Выводы.

В результате выполнения лабораторной работы была изучена обработка символьной информации с использованием строковых команд.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

DATA SEGMENT

KEEP_CS DW 0; для хранения сегмента

KEEP_IP DW 0; и смещения вектора прерывания

DATA ENDS

AStack SEGMENT STACK

DB 256 DUP(?)

AStack ENDS

CODE SEGMENT

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

SUBR_INT PROC FAR

keep_ax DW 0

keep_dx DW 0

keep_cx DW 0;

mov keep_ax, AX

mov keep_dx, DX

mov keep_cx, CX

MOV AL, 10110110b

OUT 43H, AL

MOV AX, 880 ; sound pitch

OUT 42H, AL

MOV AL, AH

OUT 42H, AL

IN AL, 61H

MOV AH, AL

OR AL, 3

OUT 61H, AL

SUB CX, CX

SOUND:

LOOP SOUND

MOV AL, AH

OUT 61H, AL

mov DX, keep_dx

MOV AL, 20H ; разрешают обработку прерываний с более низким уровнем

OUT 20H,AL

mov CX, keep_cx

mov AX, keep_ax

IRET

SUBR_INT ENDP

MAIN PROC FAR

PUSH DS

SUB AX, AX

PUSH AX

MOV AX, DATA

MOV DS, AX

MOV AH, 35H ; функция получения вектора

MOV AL, 60H ; номер вектора

INT 21H

MOV KEEP_IP, BX ; запоминание смещения

MOV KEEP_CS, ES ; и сегмента вектора прерывания

PUSH DS

MOV DX, OFFSET SUBR_INT ; смещение для процедуры в DX

MOV AX, SEG SUBR_INT ; 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

int 60h

```
CLI
PUSH DS
MOV DX, KEEP_IP
MOV AX, KEEP_CS
MOV DS, AX
MOV AH, 25H
MOV AL, 60H
INT 21H ; восстанавливаем старый вектор прерывания
POP DS
STI

ret
```

```
MAIN ENDP
CODE ENDS
END MAIN
```