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.

Студентка гр. 9383 _____ Сергиенкова А.А.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Изучить принципы работы с прерываниями. Применить на практике полученные знания о прерываниях.

Краткие сведения.

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.

Замечание: в лабораторной работе действиями по обработке прерывания может быть вывод на экран некоторого текста, вставка цикла задержки в вывод сообщения или включение звукового сигнала.

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
```

Для работы с портами существуют специальные команды IN и OUT. IN вводит значение из порта ввода-вывода, OUT выводит.

43h - запись управляющего слова в регистр режима канала

42h - загрузка счетчика канала 2, чтение счетчика канала 2

Канал 2 - генератор звука

Инструкция loop уменьшает значение в регистре CX в реальном режиме или ECX в защищённом. Если после этого значение CX не ноль, то команда loop прыгает на метку.

Команды CLI и STI служат для установки или сброса флага прерываний, что позволяет включать или отключать реакцию на внешние прерывания. Команда CLI(Clear Interrupt flag) сбрасывает флаг IF в значение 0, что запрещает прерывания. Команда STI (Set -//-) устанавливает флаг IF в значение 1, что разрешает прерывания.

Текст задания.

Вариант 3A.

Разработать собственное прерывание.

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

A - Печать сообщения на экране;

Ход работы.

По заданию нужно реализовать своё прерывание по нажатию Control+c, с выводом сообщения на экран.

Для реализации данного задания в сегменте данных создаются переменные типа DW для хранения сегмента и смещения прерывания (KEEP_CS, KEEP_IP). Также создаётся строка, которая будет выводиться во время обработки прерывания message.

В сегменте кода реализована процедура обработки прерывания MY_MESS. Данная процедура выводит строку на экран.

Для реализации смены прерывания, была использована функция 35h прерывания 21h, получаем нужный вектор прерывания 23h. После запоминаем смещение и смещение сегмента вектора прерывания в переменные KEEP_CS и KEEP_IP. Далее меняем с помощью функции 25h прерывания 21h на прерывание 23h. Восстанавливаем вектор прерывания.

Исходный код программы представлен в приложении А.

Выводы.

Были изучены основные принципы работы с прерываниями. Было разработано собственное прерывание.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

main.asm

```
AStack  SEGMENT STACK
    DB 1024 DUP(?)
AStack  ENDS

DATA    SEGMENT
    KEEP_CS DW 0    ;для хранения сегмента
    KEEP_IP DW 0    ;и смещения вектора прерывания
    message DB 'CtrlC$'

DATA    ENDS


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


 MY_MESS PROC FAR
   push ax
   push dx

   mov ah, 9h              ; ф-ия установки веткора
   mov dx, offset message   ; в dx засунули адрес сообщения
   int 21h

   pop dx
   pop ax

   mov al, 20h        ; обработка прерывания более низкого уровня
   out 20h, al
```

```
  iret              ; конец нашего прерывания
MY_MESS ENDP


MAIN PROC FAR
  mov ax, DATA
  mov ds, ax

  mov ah, 35h      ; функция получения вектора прерываний
  mov al, 23h      ; номер прерывания
  int 21h          ;получаем вектор


  mov KEEP_IP, bx  ; запомнили смещение
  mov KEEP_CS, es  ; запомнили смещение сегмента вектора прерывания
  push ds

  mov dx, offset MY_MESS  ; смещение процедуры в dx
  mov ax, seg MY_MESS     ; сегмент процедуры занесли в ax
  mov ds, ax

  mov ah, 25h           ; устанавливаем вектор прерывания
  mov al, 23h           ; наше прерывание
  int 21h               ; поменяли прерывание на наше
  pop ds

ctrl_c:
   mov ah,0
   int 16h
   cmp al,3
   jne ctrl_c

   INT 23H
```

```
    cli
    push ds
    mov dx, KEEP_IP        ; восстанавливаем смещение
    mov ax, KEEP_CS        ; восстанавливаем сегмент прерывания
    mov ds, ax
    mov ah, 25h            ; устанавливаем вектор
    mov al, 23h
    int 21h                ; меняем

    pop ds                 ; восстановили старое прерывание
    sti

    mov ah, 4ch
    int 21h

MAIN ENDP
CODE ENDS
END MAIN
```