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: Написание собственного прерывания

Вариант 20

Студент гр. 1304 _____ Новицкий М.Д.

Преподаватель _____ Кирьянчиков В.А

Санкт-Петербург

2023

Цель работы.

Написание собственного прерывания.

Задание.

23h - прерывание, генерируемое при нажатии клавиш Control+C. Реализовать вывод на экран заданного количества (3-5) сообщений, задержка между которыми возрастает в 2 раза, начиная от 1 сек.

Выполнение работы.

В начале выполнения работы определяется константа messnumber, которая содержит количество сообщений, которое необходимо вывести.

```
messnumber EQU 5
```

Далее определяется сегмент данных. В нём выделенные блоки памяти для хранения адреса старого смещения и для хранения сообщения, выводимого на экран.

```
DATA SEGMENT
      keep_cs dw 0
keep_ip dw 0    message
DB 'Hello $'
DATA ENDS

AStack SEGMENT STACK
      DW 512 DUP(?)
AStack ENDS
```

Для реализации прерывания была написана функция SUBR_PRINT. После сохранения регистров в стеке производится первая печать на экран.

```
SUBR_PRINT PROC FAR
    jmp towork
     ss_int dw 0
     sp_int dw 0
     mes_end_iter DB 'End iter$'
     int_stack dw 20 DUP(?)
```

```
towork:
mov ss_int, ss
mov sp_int, sp

; store registers
push dx
push cx
push bx
push ax
push ax

mov al, 0

print_message:
      mov ah, 9
      mov dx, offset message
      int 21h
```

После печати выполняется проверка значения на верхушке стека. Оно используется для проверки оставшегося количества печатей. Если его значение равно нулю программа завершается.

```
set_delay:
          pop cx
          dec cl
          jz complete
          push cx

          cmp al, 0
          je first

          shl al, 1
          jmp start
```

Далее происходит удвоение регистра al – в нем хранится значение текущей задержки - или присваивание ему единицы, после чего происходит переход в блок start.

```
shl al, 1
jmp start
```

Для отсчёта секунд используется регистр BL, а для хранения значения номера текущую секунд используется регистр BH. С помощью прерывания int 21h с кодом 2ch, происходит получение текущего номера секунды. Если он совпадает с сохраненным – значит секунда прошла, если нет – регистр bl уменьшается на единицу. После того, как значение регистра станет равным 0 – снова осуществляется вывод на экран. В блоке complete происходит восстановление регистров.

```
first:
        add al, 1

    start:
        mov bl, al
        mov  ah, 2ch
        int  21h
        mov bh, dh

    delaying:
        nop
        mov  ah, 2ch
        int  21h
        cmp dh, bh
        je delaying

        mov  bh, dh
        dec  bl
        jnz  delaying
        jmp print_message
```

```
complete:
      ; restore registers
      pop ax
      pop bx
      pop cx
      pop dx

      mov al, 20h
      out 20h, al
iret
```

В главной процедуре программы main происходит сохранение старого прерывания в переменные keep_cs и keep_ip.

```
mov ax, 3523h
int 21h
mov keep_cs, es
mov keep_ip, bx
```

После этого на место адресов старого прерывания записывается адрес новой процедуры.

Затем начинается блог begin. В нём прерывание int 16h используется для ожидания нового символа. Если он равен q, то происходит выход из программы. Если же был замечен символ C, тогда сначала в регистр кладётся количество вводимых сообщений, затем, в случае если был нажат Control, вызов прерывания 23h.

```
push ds
mov dx, offset SUBR_PRINT
mov ax, seg SUBR_PRINT
mov ds, ax
 mov ax, 2523h
  int 21h
```

```
        pop ds


begin:
        mov ah,0
        int 16h
        cmp al, 'q'
        je quit
        cmp al,3
        jnz begin

        mov al, messnumber
        int 23h
        jmp begin
```

После нажатия на клавишу q вектор прерывания восстанавливается.

Тестирование.

Для тестирования программы была нажата комбинация клавиш ctrl + c, а затем введена буква q с заданным количеством сообщений, равным 5. В консоль было выведено 5 сообщений с задержкой между ними в 1, 2, 4 и 8 секунд.

После ввода q программа успешно совершила выход из программы.

Выводы.

Были изучены основные принципы работы прерываний, и реализовано собственное для обработки комбинации клавиш ctrl + c.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Файл `lab5.asm`

```
 messnumber EQU 5
```

```
DATA SEGMENT
      keep_cs dw 0
keep_ip dw 0    message
DB 'Hello $'
DATA ENDS

AStack SEGMENT STACK
      DW 512 DUP(?)
AStack ENDS



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

SUBR_PRINT PROC FAR
; store registers
push dx    push cx
push bx    push ax
push ax

      mov al, 0

      print_message:
mov ah, 9
            mov dx, offset message
            int 21h

      set_delay:
            pop cx
      dec cl
            jz complete
            push cx

            cmp al, 0
            je first

            shl al, 1
      jmp start

      first:
add al, 1

      start:
  mov bl, al   mov  ah,
2ch   int  21h
mov bh, dh
      delaying:
```

```
          nop
  mov   ah,  2ch     int
21h     cmp dh, bh    je
delaying

          mov  bh, dh
     dec  bl
jnz  delaying
          jmp print_message



     complete:            ;
restore registers
          pop ax
     pop bx
pop cx          pop
dx

          mov al, 20h
     out 20h, al     iret

SUBR_PRINT ENDP

 Main PROC FAR     push DS
sub ax, ax     push ax
mov ax, DATA     mov ds, ax
mov ax, 3523h     int 21h
mov keep_cs, es     mov
keep_ip, bx      push ds
mov dx, offset SUBR_PRINT
mov ax, seg SUBR_PRINT
mov ds, ax     mov ax, 2523h
int 21h     pop ds

     begin:
mov ah,0        int
16h        cmp al, 'q'
     je quit
cmp al,3
          jnz begin

  mov al, messnumber

          int 23h
          jmp begin

     quit:
```

```
      cli     push
ds     mov dx,
keep_ip     mov ax,
keep_cs     mov ds,
ax     mov ax,
2523h     int 21h
pop ds     sti
ret Main ENDP
CODE ENDS
     END Main
```