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

ОТЧЕТ

по лабораторной работе №4
по дисциплине «Организация ЭВМ и систем»
Тема: Написание собственного прерывания.

Студент гр. 3342 _____ Мохамед Махмуд

Преподаватель _____ Кирьянчиков В.А.

Санкт-Петербург
2024

Цель работы.

Изучение прерываний и написание собственного.

Задание.

Вариант 24а.

Буква в шифре задает номер и назначение заменяемого вектора прерывания:

а – 1CH – прерывание от системного таймера – генерируется автоматически операционной системой 18 раз в сек;

Цифра определяет действия, реализуемые программой обработки прерываний:

24. Инвертирование введенных во входной строке цифр в десятичной СС и преобразование заглавных русских букв в строчные, остальные символы входной строки передаются в выходную строку непосредственно.

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 её приоритет ниже) и запоминает в стеке вместе с регистром флагов адрес возврата (CS:IP) – ме́ста, с которого будет продолжена прерванная программа. Затем в CS:IP загружается адрес программы обработки прерывания и ей передаётся 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, использующая новые программы

обработки прерываний, при своём завершении должна восстанавливать оригинальные векторы прерываний. Функция 35 прерывания 21H возвращает текущее значение вектора прерывания, помещая значение сегмента в ES, а смещение в BX.

Для задания адреса собственного прерывания с заданным номером в таблицу векторов прерываний используется функция 25H прерывания 21H, которая устанавливает вектор прерывания на указанный адрес нового обработчика.

В конце программы восстанавливается старый вектор прерывания.

Прерывания бывают аппаратные (вызываемые в результате сигналов от оборудования) и программные (вызываемые в коде). 60H – пользовательское программное прерывание.

Для обработки строковых данных ассемблер имеет пять групп команд обработки строк:

- MOVS — переслать один байт или одно слово из одной области памяти в другую;
- LODS — загрузить из памяти один байт в регистр AL или одно слово в регистр AX;
- STOS — записать содержимое регистра AL или AX в память; • CMPS — сравнить содержимое двух областей памяти, размером в один байт или в одно слово;
- SCAS — сравнить содержимое регистра AL или AX с содержимым памяти.

Каждая команда имеет модификации, указывающие размер операнда: байт (B), слово (W), двойное слово (D). Например: MOVSB, MOVSW, MOVSD.

Выполнение работы.

Сегмент данных. В сегменте данных объявлены переменные для хранения смещения и сегмента исходного прерывания, для хранения исходной строки и строки результата, переменная с приветствием.

Сегмент кода. В сегменте кода были написаны следующие процедуры:

1. input – процедура для ввода строки. Ввод осуществляется с помощью функции буферизованного ввода строки 0ah. Это значение загружается в регистр ah, затем вызывается прерывание 21h.

2. print – процедура для вывода строки. Вывод осуществляется при помощи функции выдачи строки 09h. Это значение загружается в регистр ah, затем вызывается прерывание 21h.

3. new_1ch – процедура, осуществляющая обработку строки. Вначале переменной flag присваивается значение 1, чтобы отметить вызов прерывания. Для обработки строки командой lea в si и di загружаются адреса str1 и str2, регистр bx обнуляется и будет использоваться как счётчик. Затем каждый символ str1 проверяется сначала на то, является ли он цифрой. Если является, то к цифра меняет знак на минус и к ней прибавляется 69h – константа для инвертирования ASCII-цифр. Если символ не цифра, то проверяется, является ли символ заглавной буквой русского алфавита от А до П. Если является, то к коду символа прибавляется 20h. В таблице CP 866 между строчными буквами есть разрыв, поэтому к кодам букв, идущих после п надо прибавлять другую константу – 50h. Это происходит в метке check_r_ya. После всех проверок символ записывается в строку str2, к bx прибавляется 1 и цикл продолжается до тех пор, пока bx не станет равным длине str1+1. Перед выходом из процедуры в конец str2 добавляется символ конца строки, а используемые регистры восстанавливаются.

4. main – главная процедура. Внутри неё вызываются все остальные процедуры, а также происходит замена вектора прерывания 1ch и его восстановление.

Результаты тестирования см. в Табл.1.

Исходный код см. в приложении 1.

Таблица 1. Тестирование.

№	Входные данные	Выходные данные	Комментарии
1	0123456789	987654321	Корректная обработка строки чисел
2	ПРИВЕТ ТЕСТ TEST	привет тест TEST	Заглавные русские буквы стали строчными, английские остались заглавными
3	012 ТЕСТ слово WORD	987 тест слово WORD	Числа ивертированы, заглавные русские буквы стали строчными

Вывод.

В ходе выполнения лабораторной работы были изучены команды работы со строками и прерывания, написано собственное прерывание и программа обработки строки, которая инвертирует десятичные цифры и преобразует заглавные русские буквы в строчные.

ПРИЛОЖЕНИЕ 1.
ИСХОДНЫЙ КОД.

Название файла: lr4.asm

```
.model small
.stack 500h
.data
EOFLINE EQU '$'       ; конец строки
str1head   db    80, 0
str1  db    80    DUP('*'), 0Ah, 0Dh, EOFLINE
keep_ip          dw    0     ; для хранения смещения
keep_cs          dw    0     ; и сегмента прерывания
str2  db    80    DUP('*'), 0Ah, 0Dh, EOFLINE     ; буфер для строки
результата
greeting   db    'Enter your line: $'
const_num  dw    69h   ; константа для инвертирования ASCII-цифр
flag  db    0
.code

input proc ; процедура ввода строки
      push  ax
      push  bx
      mov   ah, 0ah
      push  dx
      int   21h
      pop   bp
      xor   bx, bx
      mov   bl, ds:[bp+1]    ; в bx количество введных символов
      add   bx, bp
      add   bx, 2
      mov   word ptr[bx+1], 240ah ; добавить в конец 0ah и $
      pop   bx
      pop   ax
      ret
input endp
```

```
print proc ; процедура вывода строки
      push  ax
      mov   ah, 9
      int   21h
      pop   ax
      ret
print endp

new_1ch proc    ;прерывание 1ch

      mov   flag, 1

      push  si
      push  di
      push  bx
      push  ax

      push  ds
      pop   es

      mov   bx, 0
      lea   si, str1
      lea   di, str2
check:
      cmp   bl, str1head+1  ; проверка на конец строки
      jne   check_num
      jmp   ending
check_num:
      mov   ax, 0h
      mov   al, [si+bx]
      cmp   al, 30h         ; проверка символа на цифру 0-9
      jl    check_a_p
      cmp   al, 39h
      jg    check_a_p
      mov   ah, 00h
```

```
      neg   ax
      add   ax, const_num
      jmp   check_end
check_a_p:
      cmp   al, 80h
      jl    check_end
      cmp   al, 8fh
      jg    check_r_ya
      add   al, 20h
check_r_ya:
      cmp   al, 90h
      jl    check_end
      cmp   al, 9fh
      jg    check_end
      add   al, 50h
check_end:
      mov   [di+bx], al
      add   bx, 1
      loop  check
ending:
      mov   [di+bx], 240ah

      pop   ax
      pop   bx
      pop   di
      pop   si
      iret
new_1ch endp

main proc far
      push  ds
      sub   ax, ax
      push  ax

      mov   ax, @data
```

```
        mov   ds, ax

        mov   dx, offset greeting   ; вывод строки приветствия
        call  print

        mov   dx, offset str1head
        call  input
        mov   dl, 0ah
        mov   ah, 02h
        int   21h

        mov   ah, 35h
        mov   al, 1ch
        int   21h
        mov   keep_ip, bx      ; сохранение смещения
        mov   keep_cs, es      ; сохранение сегмента

        push  ds
        mov   dx, offset new_1ch    ; смещение для процедуры
        mov   ax, seg new_1ch       ;сегмент процедуры
        mov   ds, ax
        mov   ah, 25h
        mov   al, 1ch               ; установка вектора 1ch
        int   21h
        pop   ds

end_1ch_loop:
        mov   al, flag
        cmp   al, 1
        jne   end_1ch_loop

        lea   dx, str2   ; установка в dx адреса str2
        call  print      ; вывод результата

        cli
```

```
        push ds         ; восстановление прерывания
        mov  dx, keep_ip
        mov  ax, keep_cs
        mov  ds, ax
        mov  ah, 25h
        mov  al, 1ch
        int  21h
        pop  ds
        sti

        ret
main endp
end main
```