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

ОТЧЕТ

по лабораторной работе №7

по дисциплине «Организация ЭВМ и систем»

Тема: Преобразование целых чисел. Использование процедур в Ассемблере.

Студентка гр. 1303 _____ Андреева Е.А.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2022

Цель работы.

Получить навыки программирования на языке Ассемблера. Изучить работу с целыми числами с использованием процедур на языке Ассемблера.

Задание.

Разработать на языке Ассемблер IntelX86 две процедуры: одна - прямого и другая - обратного преобразования целого числа, заданного в регистре AX или в паре регистров DX:AX, в строку, представляющую его символьное изображение в заданной системе счисления (с учетом или без учета знака в зависимости от варианта задания).

Строка должна храниться в памяти, а также выводиться на экран для индикации.

Отрицательные числа при представлении с учетом знака должны в памяти храниться в дополнительном коде, а на экране изображаться в прямом коде с явным указанием знака или в символьном виде со знаком.

Вариант 1:

16-битное число, с учетом знака, восьмеричная система счисления, способ вызова процедур – near, связь по данным между основной программой и подпрограммами через стек.

Выполнение работы.

В главной процедуре MAIN происходит запись в регистр ax исходного числа (NUMBER в сегменте данных). Далее проверяется знак числа, если число положительное, то в SIGN кладем знак ‘+’, если отрицательное, то ‘-’. Далее это пригодится для вывода числа со знаком. Затем с помощью процедуры NUM_TO_STR_8 преобразовываем исходное число в строку (в восьмеричной системе счисления) и записываем это значение в OCT_STR. С помощью процедуры REVERSE переворачиваем эту строку и с помощью процедуры WriteMsg выводим. Затем с помощью процедуры STR_8_TO_NUM получаем из строки число, помещаем его в dx. Далее переводим число в 16-ричную систему с помощью функции NUM_TO_STR_16 и выводим его. Тот

результат, который мы получили на экране должен совпадать со значением, находящемся в NUMBER.

Процедура NUM_TO_STR_8: число делится на 8, до тех пор, пока будет не 0, а остаток от деления записывается в строку OCT_STR.

Процедура STR_8_TO_NUM: каждая цифра умножается на 8 в степени позиции этой цифры с помощью функции DIGITS_PROC, результат складывается.

Процедура NUM_TO_STR_16: делим число на 16, получаем остаток, переводим его в 16-рисную систему и добавляем к строке HEX_STR.

Исходный код программы см. в приложении А.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	0Ah		
2.	1111h		
3.	0FFF1h		
4.	0FFAh		

Выводы.

В ходе выполнения лабораторной работы были получены навыки программирования на языке Ассемблера. Была разработана программа переводящая число в 8-ричную систему в строковом виде и обратно.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lab7.asm

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

DATA    SEGMENT
HEX_INFO DB 'Hex number = ', '$'
OCT_INFO DB 'Octal number = ', '$'
    HEX_STR DB ' ', '$'
    OCT_STR DB ' ', '$'
    SIGN DB '  ', '$'
NUMBER DW 0FFAh
DATA    ENDS

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


WriteMsg  PROC  NEAR
          mov AH, 9
          int 21h
          ret
WriteMsg  ENDP


DIGITS_PROC PROC NEAR
push bx
push dx
push cx
cmp cx,0
je digits_end

mov bx,08h
digits_processing:
      xor dx,dx
      mul bx
      sub cx,1
      cmp cx,0
      jne digits_processing

digits_end:
pop cx
pop dx
pop bx
ret
DIGITS_PROC ENDP


REVERSE PROC NEAR ; переворачиваем строку с числом
pop cx

pop di  ;в di строка
pop bx
```

```
push cx

xor ax,ax
reverse_processing:
          push bx
          mov bx,ax
          mov cx,[di+bx]
          pop bx
          mov dx,[di+bx]

          push bx
          mov bx,ax
          mov [di+bx],dx
          pop bx
          mov [di+bx],cx

          add ax,2
          sub bx,2
          cmp ax,bx
          jl reverse_processing
ret
REVERSE ENDP

STR_8_TO_NUM PROC NEAR
pop cx

pop di  ; строка
pop bx

push cx
push ax

xor dx,dx
     mov cx,0
     str_processing:
          mov ax,[di+bx]
          sub ax,'0'
          call digits_proc
          add dx,ax
          inc cx
          sub bx,2
          cmp bx,0
          jnl str_processing

     pop ax
pop cx

push dx ;нужное число

push cx
ret
STR_8_TO_NUM ENDP


NUM_TO_STR_8 PROC NEAR
pop cx
```

```
pop di
pop dx

push cx
push ax

sub bx,bx
      mov ax,dx
      mov cx, 08h
      oct_processing:
            sub dx,dx
            div cx
            add dx,'0'
            mov [di+bx],dx
            add bx,2
            cmp ax,0
            jne oct_processing
      mov cx,'$'
      mov [di+bx],cx
      sub bx,2

      pop ax
pop cx

push bx ;длина строки

push cx

ret
NUM_TO_STR_8 ENDP

NUM_TO_STR_16 PROC NEAR
pop cx

pop di
pop dx

push cx
push ax

sub bx,bx
      mov ax,dx
      mov cx, 10h
      hex_begin:
            sub dx,dx
            div cx
            add dx,'0'
            cmp dx,'9'
            jle end_hex
            add dx,7


            end_hex:
            mov [di+bx],dx
            add bx,2
            cmp ax,0
            jne hex_begin
      mov cx,'$'
```

```
      mov [di+bx],cx
      sub bx,2

      pop ax
pop cx

push bx ;длина строки

push cx

ret
NUM_TO_STR_16 ENDP


MAIN PROC FAR
    push ds
    sub ax,ax
    push ax
    mov ax, DATA
    mov ds, ax

      mov dx,offset oct_info
      call writemsg

      mov ax,number
      mov di,offset sign
      mov bx,'+'
      cmp ax,0
      jnl set_sign
mov bx,'-'
neg ax

      set_sign:
      push bx ; знак на стек
      mov [di],bx

      push ax
      mov dx,offset sign
      call writemsg
      pop ax

      mov di,offset oct_str
      push ax
      push di
      call num_to_str_8
      pop bx

      push bx ; кладем длину строки на стек ----

      mov di,offset oct_str
      push bx
      push di
      call reverse


      mov dx,offset oct_str
      call writemsg
```

```
;-------------------------------------------

      pop bx ; достаем длину строки со стека ----

      mov di,offset oct_str
      push bx
      push di
      call str_8_to_num
      pop dx ;перевели в число и поместили в dx


      pop bx
      cmp bx,'-'
      jne skip
      neg dx
      skip:

      mov di,offset hex_str
      push dx
      push di
      call num_to_str_16
      pop bx

      push bx
      push di
      call reverse

      mov dx,offset hex_info
      call writemsg
      mov dx,offset hex_str
      call writemsg

ret
MAIN ENDP
CODE ENDS
     END MAIN
```