

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

ОТЧЕТ
по лабораторной работе №7
по дисциплине «Организация ЭВМ»
Тема: Преобразование целых чисел. Использование процедур в
Ассемблере.

Студент гр. 0382

Крючков А.М.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2021

Цель работы.

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.

Пример для однобайтовых чисел:

Десятичное число в символьном виде. Двоично-десят. упаков.число

	в ДК	в ПК
+ 35	00110101	00110101
- 35	11001011	10110101

Вариант выполнения преобразования определяется шифром, состоящим из 4-х цифр:

- 1-я цифра задает длину целого числа:

1- 16 бит, 2- 32 бита;

- 2-я цифра задает вид представления числа:

1- с учетом знака, 2- без учета знака;

- 3-я цифра задает систему счисления для символьного изображения числа:

1- двоичная, 2- восьмеричная, 3- десятичная, 4- шестнадцатеричная.

- 4-я цифра задает способ вызова процедур:

1- near (ближнего вызова), 2 - far (дальнего вызова);

Написать простейшую головную программу для иллюстрации корректности выполнения заданных преобразований.

Связь по данным между основной программой и подпрограммами

может осуществляться следующими способами:

А - через РОНЫ; В - через кадр стека.

Вариант 10

32 бита, без учёта знака, десятичная 2.2.3.1.В

Порядок выполнения работы.

Создано две процедуры. Первая `numb_to_string` переводит число в строку, вторая `str_to_num` переводит строку в число.

Перевод числа в строку происходит последовательным делением на 10. Сначала мы получаем остаток от деления на 10 и записываем его в строку. Затем само число, состоящее из двух частей делится на 10. После проработки процедуры на выход мы получаем в регистр `sx` длину строки.

Перевод из строки в число происходит запоминанием двух частей числа из 16 бит. Сначала умножается нижняя часть на 10, часть которая превысила 16 бит добавляет в верхнюю часть, которая в свою очередь также предварительно умножается на 10.

Вывод.

В ходе выполнения данной лабораторной работы была изучена разработка процедур.

ТЕСТИРОВАНИЕ

Заданное число 12345678:

```
C:\>LAB7_C~1.EXE  
1234567812345678
```

ПРИЛОЖЕНИЕ А ИСХОДНЫЙ КОД ПРОГРАММ

Файл lab7.asm

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

DATA    SEGMENT
        A dd 12345678d      ; делимое
        Result dd 0         ; результат деления
        Remainder dw 0      ; остаток
        res_str db 12 dup (0)

        hr      dw 0
        lr      dw 0
DATA    ENDS

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

start PROC NEAR
        ; инициализация ds
        mov ax, data
        mov ds, ax

        call numb_to_string ; То что в Result переводит в строку res_str
        ; cx - length
        push cx
        push si
        call print

        call str_to_num
        pop si
        pop cx

        mov si, offset res_str

        call numb_to_string ; То что в Result переводит в строку res_str
        mov si, offset res_str
        call print2

        mov ax, 4c00h      ; выход в DOS
        int 21h
start endp

numb_to_string proc NEAR

        lea si, res_str
        mov cx, 0

        ; делитель
        mov bx, 10
```

```

    mov ax, word ptr [A+2]
    mov word ptr [Result+2], ax
    mov ax, word ptr [A]
    mov word ptr [Result], ax

again:
    ; делим старшее слово
    xor dx, dx
    mov ax, word ptr [Result+2]
    div bx

    mov word ptr [Result+2], ax      ; сохраняем результат от деления
старшего слова                    ; в dx остаток от деления

    ; делим младшее слово
    mov ax, word ptr [Result]
    div bx

    mov word ptr [Result], ax ; сохраняем результат от деления
младшего слова
    mov word ptr [Remainder], dx    ; сохраняем остаток от деления

    ; переводим цифру в символ и сохраняем
    and dx, 0FFh
    add dx, '0'
    mov [si], dl
    inc si          ; смещение следующего символа в строке
    inc cx          ; счетчик символов

    ; если частное от деления не равно 0, то повторяем операцию
    mov ax, word ptr [Result]
    cmp ax, 0
    jnz again
    mov ax, word ptr [Result+2]
    cmp ax, 0
    jnz again

    ret
    ; печать строки в обратном порядке
    ; в cx - длина строки
numb_to_string ENDP

str_to_num PROC NEAR
; hr lr
; ax - цифра
; hr = hr * 10
; lr = lr * 10, dx
; lr = lr + ax
; hr = hr + dx
    lea si, res_str
    mov bx, 10
    xor dx, dx
again_r:
    xor ah, ah

```

```

mov al, [si]
cmp al, 0
jz exit

```

```

sub ax, '0'      ; ax - цифра

```

```

; hr = hr * 10
push ax
push dx
mov ax, hr
mov dx, 0
mul bx
mov hr, ax
pop dx
pop ax

```

```

; lr = lr * 10, dx
push ax
mov ax, lr
mov dx, 0
mul bx
mov lr, ax
pop ax

```

```

; lr = lr + ax
add ax, lr
mov lr, ax

```

```

; hr = hr + dx
mov ax, hr
add ax, dx
mov hr, ax

```

```

inc si
jmp again_r
exit:
    mov ax, hr
    mov dx, lr
    mov word ptr [A+2], ax
    mov word ptr [A], dx
    ret

```

```

; печать строки в обратном порядке
; в cx - длина строки
str_to_num endp

```

```

print proc NEAR
print_s:
    mov dl, [si-1]
    mov ah, 02h
    int 21h
    dec si
    loop print_s
ret

```

```
print endp

print2 proc NEAR
print_s2:
    mov dl,[si]
    mov ah,02h
    int 21h
    inc si
    loop print_s2
    ret
print2 endp

CODE ENDS
    END start
```