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 и систем»

Тема: Использование арифметических операций над целыми числами и процедур в Ассемблере.

Студент гр. 9382 _____ Павлов Р.В.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Разработка процедур перевода из строки в число и из числа в строку.

Задание (вариант 2).

Разработать на языке Ассемблер процессора IntelX86 две процедуры:

• одна – выполняет прямое преобразование 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);

• другая - обратное преобразование строки, представляющей символьное изображение числа в заданной системе счисления в целое число, помещаемое в регистр AX (или в пару регистров DX:AX) 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.

Ход работы.

1) В отдельном кодовом сегменте написана процедура считывания строки (двоичного представления числа) с клавиатуры и перевода её в число, которая затем через стек возвращает два слова — значения регистров DX и AX.

2) В основном сегменте кода написана процедура, посредством побитового сдвига определяющая каждый последующий бит числа и заносящая соответствующий ему символ в место в памяти, где хранится результирующая строка.

3) Написана главная процедура, которая вызывает сначала первую, а потом вторую процедуры. Затем выводится результирующая строка. Цель — демонстрация работы программы.

Выводы.

В результате выполнения лабораторной работы написана программа, преобразующая строку в число и наоборот.

ПРИЛОЖЕНИЕ А. ИСХОДНЫЙ КОД

- имя файла : lab7.asm

```
stack segment stack
      dw 64 dup(0)
stack ends

data segment
      choice db '0 - ④¢®¨筮¥ 梓«®, 1 - ¤¥□¨筮¥ 梓«®', 13, 10, '$'
      product dw 4 dup(0)
      origin db 33, ?, 33 dup(0)
      result db 33 dup(0)
      error db "incorrect number$"
data ends

additional segment
      assume ds:data, cs:additional
      strToInt proc far

            push ax
            mov ax, data
            mov ds, ax
            pop ax

            xor cx, cx
            mov ah,0ah
            mov dx,offset origin          ; Считывание строки и запись её в
буфер, перевод на новую строку
            int 21h
            mov dl,0ah
            mov ah,02
            int 21h

            mov si,offset origin+2
```

```
            xor ax,ax                   ; Готовим регистры для записи: ax =
0, dx = 0, bx = 2 - основание СС
            xor dx, dx
            mov bx,2

            mov cl, origin[1]

            transformdx:
            cmp cx, 17                  ; Расчёт двух старших байтов
            jl sec_word

            push cx

            mov cl, [si]
            cmp cl,'0'                  ; Проверка на соответствие цифре
            jb err
            cmp cl,'1'
            ja err

            sub cl,'0'                  ; Перевод из кода символа в цифру,
домножение на 10, прибавление в конец
            mul bx
            add ax,cx
            inc si

            pop cx

            loop transformdx

            sec_word:                   ; Расчёт двух младших байтов
            push ax
            xor ax, ax

            transformax:
            mov cl,[si]                 ; Проверка на последний символ
            cmp cl,0dh
            jz fin

            cmp cl,'0'                  ; Проверка на соответствие цифре
```

```
            jb err
            cmp cl,'1'
            ja err

            sub cl,'0'                    ; Перевод из кода символа в цифру,
домножение на 2, прибавление в конец
            mul bx
            add ax,cx
            inc si
            jmp transformax

            err:
            mov dx, offset error    ; Ошибка (если не цифра), выход
            mov ah,09
            int 21h
            int 20h

            fin:
            pop dx
            pop cx
            pop bx

            push ax                              ; Помещение числа в стек
            push dx

            push bx
            push cx
            ret

      strToInt endp
additional ends

code segment
      assume ds:data, cs:code, ss:stack

      strToInt10 proc near

            push ax
```

```
            mov ax, data
            mov ds, ax
            pop ax

            xor cx, cx
            mov ah,0ah
            mov dx,offset origin          ; Считывание строки и запись её в
буфер, перевод на новую строку
            int 21h
            mov dl,0ah
            mov ah,02
            int 21h

            mov si,offset origin+2
            xor dx, dx
            xor ax, ax                    ; Готовим регистры для записи: ax =
0 , bx = 10 - основание СС
            mov bx,10

            transform:
            mov cl,[si]                   ; Проверка на последний символ
            cmp cl,0dh
            jz fin10


            cmp cl,'0'                    ; Проверка на соответствие цифре
            jb err10
            cmp cl,'9'
            ja err10

            sub cl,'0'                    ; Перевод из кода символа в цифру,
домножение на 10, прибавление в конец

            cmp dx, 0
            jg omg

            mul bx
            add ax,cx
            inc si
```

```
            jmp transform

            omg:
            push dx
            mul bx
            mov product[2], dx
            mov product[4], ax
            pop ax
            mul bx
            add product, dx
            adc product[2], ax
            mov dx, product[2]
            mov ax, product[4]
            adc ax, cx
            inc si
            jmp transform

            err10:
            mov dx, offset error    ; Ошибка (если не цифра), выход
            mov ah,09
            int 21h
            int 20h

            fin10:
            ret

      strToInt10 endp

      intToStr proc near
            push ax
            push bx
            push cx
            push dx
            push di

            lea di, result                ; Переход в конец строки, запись
символа конца строки
```

```
            add di, 33
            mov cl, '$'
            mov [di], cl
            dec di

            mov cx, 16

            shiftax:
                  shr ax, 1
                  jc setax
                  mov ch, 48
                  jmp recax

                  setax:
                        mov ch, 49        ; Сдвиг вправо, заполнение первых 16
знаков

                  recax:
                  mov [di], ch
                  dec di
                  and cx, 00FFh
                  loop shiftax

            mov cx, 16

            shiftdx:
                  shr dx, 1
                  jc setdx
                  mov ch, 48
                  jmp recdx

                  setdx:
                        mov ch, 49        ; Сдвиг вправо, заполнение последних
16 знаков

                  recdx:
                  mov [di], ch
                  dec di
                  and cx, 00FFh
```

```
            loop shiftdx

      pop di
      pop dx
      pop cx
      pop bx
      pop ax
      ret
intToStr endp

main proc far
      xor ax, ax
      push ds
      push ax

      mov ax, data
      mov ds, ax

      mov dx, offset choice
      mov ah, 9
      int 21h

      entering:

      mov ah, 1
      int 21h

      push ax

      mov ah, 2
      mov dl, 10
      int 21h
      mov dl, 13
      int 21h

      pop ax

      cmp al, 30h
```

```
            jne decimal

            call strToInt                ; Ввод числа
            ;assume cs: code
            pop dx                              ; Получение числа из стека
            pop ax
            jmp toStr

            decimal:
            cmp al, 31h
            jne entering

            call strToInt10         ; Ввод числа

            toStr:
            call intToStr                ; Запись числа в виде строки

            mov dx, offset result
            mov ah, 9                    ; Вывод результата на экран
            int 21h

            ret
      main endp
code ends
end main
```

ПРИЛОЖЕНИЕ Б. ТЕКСТ ЛИСТИНГОВ

- имя файла: lab5.lst

```
#Microsoft (R) Macro Assembler Version 5.10                 12/10/20 15:29:4
                                                            Page     1-1


 0000                   stack segment stack
 0000  0040[                           dw 64 dup(0)
         0000
                    ]

 0080                   stack ends

 0000                   data segment
 0000  30 20 2D 20 84 A2               choice db '0 - ④¢®¨ 筮¥  秤«®, 1 - ¤¥□
                        ⅟₂®¥ 秤«®', 13, 10, '$'
       AE A8 E7 AD AE A5
       20 E7 A8 E1 AB AE
       2C 20 31 20 2D 20
       A4 A5 E1 EF E2 A8
       E7 AD AE A5 20 E7
       A8 E1 AB AE 0D 0A
       24
 002B  0004[                           product dw 4 dup(0)
         0000
                    ]

 0033  21 00                           origin db 33, ?, 33 dup(0)
       0021[
         00
                    ]

 0056  0021[                           result db 33 dup(0)
         00
                    ]

 0077  69 6E 63 6F 72 72               error db "incorrect number$"
       65 63 74 20 6E 75
       6D 62 65 72 24
 0088                   data ends

 0000                   additional segment
                              assume ds:data, cs:additional
 0000                         strToInt proc far

 0000  50                              push ax
 0001  B8 ---- R                       mov ax, data
 0004  8E D8                                 mov ds, ax
 0006  58                              pop ax

 0007  33 C9                                 xor cx, cx
 0009  B4 0A                                 mov ah,0ah
 000B  BA 0033 R                       mov dx,offset origin
                              ; Считывание строки и
                        запись её в буфер, перевод
                         на новую строку
 000E  CD 21                                 int 21h
```

```
0010  B2 0A                             mov dl,0ah
0012  B4 02                             mov ah,02
0014  CD 21                             int 21h

#Microsoft (R) Macro Assembler Version 5.10                 12/10/20 15:29:4
                                                            Page     1-2


0016  BE 0035 R                         mov si,offset origin+2

0019  33 C0                                   xor ax,ax
                                        ; Готовим регистр
                      ы для записи: ax = 0, dx = 0, bx = 2
                      - основание СС
001B  33 D2                                   xor dx, dx
001D  BB 0002                                 mov bx,2

0020  8A 0E 0034 R                           mov cl, origin[1]

0024                                    transformdx:
0024  83 F9 11                                cmp cx, 17
                                        ;         Расчёт         двух         стаэ
                      ▐ших байтов
0027  7C 18                                   jl sec_word

0029  51                                ; push cx

002A  8A 0C                                   mov cl, [si]
002C  80 F9 30                                cmp cl,'0'
                                        ;         Проверка         на         сооэ
                      ⓶ветствие цифре
002F  72 2E                                   jb err
0031  80 F9 31                                cmp cl,'1'
0034  77 29                                   ja err

0036  80 E9 30                                sub cl,'0'
                                        ; Перевод из кода
                      символа          в          цифру,          домноженЙ
                      ˛е на 10, прибавление в коне
                      ц
0039  F7 E3                                   mul bx
003B  03 C1                                   add ax,cx
003D  46                                ; inc si

003E  59                                ; pop cx

003F  E2 E3                                   loop transformdx

0041                                    sec_word:
                                        ;         Расчёт         двух         млаЙ
                      ´ших байтов
0041  50                                ; push ax
0042  33 C0                                   xor ax, ax

0044                                    transformax:
0044  8A 0C                                   mov cl,[si]
                                        ;         Проверка         на         посЙ
                      »едний символ
0046  80 F9 0D                                cmp cl,0dh
0049  74 1D                                   jz fin
```

```
004B  80 F9 30                        cmp cl,'0'
                              ;         Проверка          на          сооэ
                      ⓶ветствие цифре
004E  72 0F                           jb err
0050  80 F9 31                        cmp cl,'1'
0053  77 0A                           ja err

0055  80 E9 30                        sub cl,'0'
                              ; Перевод из кода
                      символа           в           цифру,           домноженЙ
                      ¸е       на       2,       прибавление       в       конеэ
                      ⓺
0058  F7 E3                           mul bx
005A  03 C1                           add ax,cx
005C  46                        inc si
005D  EB E5                           jmp transformax

005F                            err:
005F  BA 0077 R                 mov dx, offset error    ;               ОшЙ
                      ¸бка (если не цифра), выход
0062  B4 09                           mov ah,09
0064  CD 21                           int 21h
0066  CD 20                           int 20h

0068                            fin:
0068  5A                        pop dx
0069  59                        pop cx
006A  5B                        pop bx

006B  50                        push ax
                              ; Помещение числа
                       в стек
006C  52                        push dx

006D  53                        push bx
006E  51                        push cx
006F  CB                        ret

0070                        strToInt endp
0070                  additional ends

0000                  code segment
                            assume ds:data, cs:code, ss:stack

0000                        strToInt10 proc near

0000  50                          push ax
0001  B8 ---- R                   mov ax, data
0004  8E D8                             mov ds, ax
0006  58                          pop ax

0007  33 C9                             xor cx, cx
0009  B4 0A                             mov ah,0ah
000B  BA 0033 R                 mov dx,offset origin
```

```
                        ; Считывание строки и
                      запись её в буфер, перевод
                       на новую строку
000E  CD 21                           int 21h
0010  B2 0A                           mov dl,0ah
0012  B4 02                           mov ah,02
0014  CD 21                           int 21h

0016  BE 0035 R                   mov si,offset origin+2
0019  33 D2                           xor dx, dx
001B  33 C0                           xor ax, ax
                        ; Готовим регистр
                      ы для записи: ax = 0 , bx = 10 - ос
                      нование СС
001D  BB 000A                         mov bx,10

0020                              transform:
0020  8A 0C                           mov cl,[si]
                        ;         Проверка         на         посЙ
                      »едний символ
0022  80 F9 0D                        cmp cl,0dh
0025  74 43                           jz fin10


0027  80 F9 30                        cmp cl,'0'
                        ;         Проверка         на         сооэ
                      ⑫ветствие цифре
002A  72 35                           jb err10
002C  80 F9 39                        cmp cl,'9'
002F  77 30                           ja err10

0031  80 E9 30                        sub cl,'0'
                        ; Перевод из кода
                      символа          в          цифру,          домноженЙ
                      ¸е на 10, прибавление в коне
                      ц

0034  83 FA 00                        cmp dx, 0
0037  7F 07                           jg omg

0039  F7 E3                           mul bx
003B  03 C1                           add ax,cx
003D  46                          inc si

003E  EB E0                           jmp transform

0040                              omg:
0040  52                          push dx
0041  F7 E3                           mul bx
0043  89 16 002D R                    mov product[2], dx
0047  A3 002F R                   mov product[4], ax
004A  58                          pop ax
004B  F7 E3                           mul bx
004D  01 16 002B R                    add product, dx
```

```
0051  11 06 002D R                     adc product[2], ax
0055  8B 16 002D R                     mov dx, product[2]
0059  A1 002F R                  mov ax, product[4]
005C  13 C1                           adc ax, cx
005E  46                          inc si
005F  EB BF                           jmp transform

0061                              err10:
0061  BA 0077 R                   mov dx, offset error    ;             ОшЙ
                      ¸бка (если не цифра), выход
0064  B4 09                           mov ah,09
0066  CD 21                           int 21h
0068  CD 20                           int 20h

006A                              fin10:
006A  C3                          ret

006B                        strToInt10 endp

006B                        intToStr proc near
006B  50                          push ax
006C  53                          push bx
006D  51                          push cx
006E  52                          push dx
006F  57                          push di

0070  8D 3E 0056 R                    lea di, result
                      ; Переход в конец стро
                    ки, запись символа конца с
                    троки
0074  83 C7 21                        add di, 33
0077  B1 24                           mov cl, '$'
0079  88 0D                           mov [di], cl
007B  4F                          dec di

007C  B9 0010                         mov cx, 16

007F                              shiftax:
007F  D1 E8                               shr ax, 1
0081  72 05                               jc setax
0083  B5 30                               mov ch, 48
0085  EB 03 90                            jmp recax

0088                                  setax:
0088  B5 31                                   mov ch, 49
                                  ; Сдвиг вправо, за
                    полнение первых 16 знаков

008A                                  recax:
008A  88 2D                               mov [di], ch
008C  4F                          dec di
008D  81 E1 00FF                  and cx, 00FFh
0091  E2 EC                           loop shiftax


#Microsoft (R) Macro Assembler Version 5.10                 12/10/20 15:29:4
                                                            Page     1-6


0093  B9 0010                         mov cx, 16
```

```
0096                                shiftdx:
0096  D1 EA                                 shr dx, 1
0098  72 05                                 jc setdx
009A  B5 30                                 mov ch, 48
009C  EB 03 90                              jmp recdx

009F                                    setdx:
009F  B5 31                                       mov ch, 49
                                    ; Сдвиг вправо, за
                          полнение последних 16 знак
                          ов

00A1                                    recdx:
00A1  88 2D                                 mov [di], ch
00A3  4F                                dec di
00A4  81 E1 00FF                        and cx, 00FFh
00A8  E2 EC                                 loop shiftdx

00AA  5F                            pop di
00AB  5A                            pop dx
00AC  59                            pop cx
00AD  5B                            pop bx
00AE  58                            pop ax
00AF  C3                            ret
00B0                          intToStr endp

00B0                          main proc far
00B0  33 C0                               xor ax, ax
00B2  1E                            push ds
00B3  50                            push ax

00B4  B8 ---- R                     mov ax, data
00B7  8E D8                             mov ds, ax

00B9  BA 0000 R                     mov dx, offset choice
00BC  B4 09                             mov ah, 9
00BE  CD 21                             int 21h

00C0                                entering:

00C0  B4 01                             mov ah, 1
00C2  CD 21                             int 21h

00C4  50                            push ax

00C5  B4 02                             mov ah, 2
00C7  B2 0A                             mov dl, 10
00C9  CD 21                             int 21h
00CB  B2 0D                             mov dl, 13
00CD  CD 21                             int 21h

00CF  58                            pop ax
```



```
00D0  3C 30                             cmp al, 30h
00D2  75 0A                             jne decimal
```

```
 00D4  9A 0000 ---- R                    call strToInt
                              ; Ввод числа
                                    ;assume cs: code

 00D9  5A                               pop dx
                                        ; Получение числа
                         из стека
 00DA  58                               pop ax
 00DB  EB 08 90                               jmp toStr

 00DE                                   decimal:
 00DE  3C 31                                  cmp al, 31h
 00E0  75 DE                                  jne entering

 00E2  E8 0000 R                        call strToInt10         ;               ВвЙ
                        ¾д числа

 00E5                                   toStr:
 00E5  E8 006B R                        call intToStr
                                ;       Запись      числа      в      виде      э
                        Ґроки

 00E8  BA 0056 R                        mov dx, offset result
 00EB  B4 09                                  mov ah, 9
                                        ; Вывод результат
                        а на экран
 00ED  CD 21                                  int 21h

 00EF  CB                               ret
 00F0                             main endp
 00F0                   code ends
                        end main

#Microsoft (R) Macro Assembler Version 5.10                 12/10/20 15:29:4
                                                            Symbols-1


Segments and Groups:

                N a m e             Length       Align      Combine Class

ADDITIONAL . . . . . .  .  .  .     0070  PARA  NONE
CODE . . . . . .  .  .  .  .  .     00F0  PARA  NONE
DATA . . . . . .  .  .  .  .  .     0088  PARA  NONE
STACK  . . . . . .  .  .  .  .      0080  PARA  STACK

Symbols:

                N a m e             Type   Value       Attr

CHOICE . . . . . .  .  .  .  .      L BYTE      0000  DATA

DECIMAL  . . . . . .  .  .  .       L NEAR      00DE  CODE

ENTERING . . . . . .  .  .  .       L NEAR      00C0  CODE
ERR  . . . . . .  .  .  .  .  .     L NEAR      005F  ADDITIONAL
ERR10  . . . . . .  .  .  .  .      L NEAR      0061  CODE
ERROR  . . . . . .  .  .  .  .      L BYTE      0077  DATA

FIN  . . . . . .  .  .  .  .  .     L NEAR      0068  ADDITIONAL
```

```
FIN10  . . . . . .    L NEAR      006A  CODE

INTTOSTR . . . . . .    N PROC      006B  CODE  Length = 0045

MAIN . . . . . .    F PROC      00B0  CODE  Length = 0040

OMG  . . . . . .    L NEAR      0040  CODE
ORIGIN . . . . . .    L BYTE      0033  DATA

PRODUCT  . . . . . .    L WORD      002B  DATA  Length = 0004

RECAX  . . . . . .    L NEAR      008A  CODE
RECDX  . . . . . .    L NEAR      00A1  CODE
RESULT . . . . . .    L BYTE      0056  DATA  Length = 0021

SEC_WORD . . . . . .    L NEAR      0041  ADDITIONAL
SETAX  . . . . . .    L NEAR      0088  CODE
SETDX  . . . . . .    L NEAR      009F  CODE
SHIFTAX  . . . . . .    L NEAR      007F  CODE
SHIFTDX  . . . . . .    L NEAR      0096  CODE
STRTOINT . . . . . .    F PROC      0000  ADDITIONAL  Length     =
0070
STRTOINT10 . . . . . .    N PROC      0000  CODE  Length = 006B

TOSTR  . . . . . .    L NEAR      00E5  CODE
TRANSFORM  . . . . . .    L NEAR      0020  CODE
TRANSFORMAX  . . . . . .    L NEAR      0044  ADDITIONAL
TRANSFORMDX  . . . . . .    L NEAR      0024  ADDITIONAL

@CPU . . . . . .    TEXT  0101h
@FILENAME  . . . . . .    TEXT  lab7
@VERSION . . . . . .    TEXT  510

#Microsoft (R) Macro Assembler Version 5.10                 12/10/20 15:29:4
                                                            Symbols-2



    278 Source  Lines
    278 Total   Lines
     36 Symbols

  47900 + 451168 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors
```