

# Прямая адресация

Данный вид адресации иногда, так же, называют - **прямая адресация памяти**.

Адресуется память; адрес ячейки памяти (слова или байта) указывается в команде (обычно в символической форме) и поступает в код команды:

;Сегмент данных

**mem1 dw 0** ;Слово памяти содержит 0


**mem2 db 230** ;Байт памяти содержит 230

;Сегмент команд

**inc mem1** ;Содержимое слова mem1 увеличивается на 1

**mov DX,mem1** ; Содержимое слова с именем mem1 загружается в DX

**mov AL,mem2** ; Содержимое байта с именем mem2 загружается в AL



Команды процессора, обращающиеся к памяти, могут в качестве первого байта своего кода содержать префикс замены сегмента, с помощью которого процессор определяет, из какого сегментного регистра взять сегментный адрес. Для сегментного регистра ES код префикса составляет 26h, для SS - 36h, для CS - 2Eh. Если префикс отсутствует, сегментный адрес берется из регистра DS (хотя для него тоже предусмотрен свой префикс). Если в начале программы с помощью директивы `assume` указано соответствие сегменту данных сегментного регистра DS

### **`assume DS:data`**

то команды обращения к памяти транслируются без какого-либо префикса, а процессор при выполнении этих команд берет сегментный адрес из регистра DS.

Если в директиве `assume` указано соответствие сегмента данных регистру ES

### **`assume ES:data`**

# Пример прямой адресации

```
C:\ASM\ADDRESS\tasm\direct2.asm
      assume  cs:code,es:data ;указали назначение сегментов
data   segment                ;начало сегмента данных
msg0   db    'TEST$'
msg    db    '-=* Hello World *--$'
dat    dw    1111
data   ends                  ;конец сегмента данных
code   segment                ;начало сегмента команд
begin: mov    ax,data          ;адрес сегмента данных в AX
      mov    es,ax            ;затем перенесем из AX в ES
      mov    ax,dat
      mov    ax,data
      mov    ds,ax
      mov    bx,offset dat
      mov    ah,09h           ;функция DOS 9h вывода на экран
      mov    dx,offset msg    ;адрес выводимого сообщения должен быть в DX
      int    21h              ;вызов DOS
      mov    ax,4c00h         ;функция 4c00h успешное завершения программы
      int    21h              ;вызов DOS
code   ends                  ;конец сегмента команд
stk    segment stack          ;начало сегмента стека
      db    256 dup('*')      ;резервируем 256 байт для стека
stk    ends                  ;конец сегмента стека
      end    begin           ;конец текста программы с точкой входа
```

# А также его листинг

```
C:\ASM\ADDRESS\tasm\DIRECT2.LST Win-1251 Строка
1
2 0000          data      assume cs:code,es:data ;указали назначение сегментов
3 0000          msg@      segment      ;начало сегмента данных
4 0005          db         'TEST$'
5              db         '--* Hello World *--$'
6
7 0019          dat        dw         1111
8 001B          data      ends          ;конец сегмента данных
9 0000          code      segment      ;начало сегмента команд
10 0000         B8 0000s     begin:    mov     ax,data      ;адрес сегмента данных в AX
11 0003         8E C0       mov     es,ax        ;затем перенесем из AX в ES
12 0005         26 A1 0019r     mov     ax,dat
13 0009         B8 0000s     mov     ax,data      ← законментируйте эти две строчки и
14 000C         8E D8       mov     ds,ax        посмотрите что будет...
15 000E         BB 0019r     mov     bx,offset dat
16 0011         B4 09       mov     ah,09h       ;функция DOS 9h вывода на экран
17 0013         BA 0005r     mov     dx,offset msg ;адрес выводимого сообщения должен быть в DX
18 0016         CD 21       int     21h          ;вызов DOS
19 0018         B8 4C00     mov     ax,4c00h     ;функция 4c00h успешное завершения программы
20 001B         CD 21       int     21h          ;вызов DOS
21 001D          code      ends          ;конец сегмента команд
22 0000          stk        segment stack ;начало сегмента стека
23 0000         0100*(2A)   db         256 dup('*') ;резервируем 256 байт для стека
24 0100          stk        ends          ;конец сегмента стека
25          end          begin  ;конец текста программы с точкой входа
```

Symbol Name	Type	Value
??FILENAME	Text	"direct2 "
??VERSION	Number	040A
@CPU	Text	0101H
@CURSEG	Text	STK
@FILENAME	Text	DIRECT2
@WORDSIZE	Text	2
BEGIN	Near	CODE:0000
DAT	Word	DATA:0019
MSG	Byte	DATA:0005
MSG@	Byte	DATA:0000

Groups & Segments	Bit	Size	Align	Combine	Class
CODE	16	001D	Para	none	
DATA	16	001B	Para	none	
STK	16	0100	Para	Stack	

- Пусть требуется вывести в левый верхний угол экрана несколько символов, например, два восклицательных знака. Эту операцию можно реализовать с помощью следующих команд:

**mov AX,0B800h** ;Сегментный адрес видеобуфера

**mov ES,AX** ;Отправим его в ES

**mov byte ptr ES:0,'!'** ;Отправим символ на 1-е знакоместо экрана

**mov byte ptr ES:2,'!'** ;Отправим символ на 2-е знакоместо экрана

- Настроив регистр ES на сегментный адрес видеобуфера B800h, мы пересылаем код знака "!" сначала по относительному адресу 0 а затем на следующее знакоместо, имеющее смещение 2. В обеих командах необходимо с помощью обозначения ES: указать сегментный регистр, который используется для адресации памяти. Встретившись с этим обозначением, транслятор включит в код команды префикс замены сегмента, в данном случае код **26h**.

C:\ASM\ADDRESS\tasm\direct3.asm

```
        assume    cs:code,ds:data ;указали назначение сегментов
data    segment ;начало сегмента данных
dat     dw       3333h
data    ends    ;конец сегмента данных
code    segment ;начало сегмента команд
begin:  jmp      bgn
mem     dw       1111h
bgn:    mov     ax,data ;адрес сегмента данных в AX
        mov     ds,ax  ;затем перенесем из AX в DS
        mov     bx,cs:mem
        mov     ah,0   ;номер функции установки режима дисплея
        mov     al,3   ;код режима 80*25 (цветной)
        int     10h    ;очистка экрана
        mov     ax,0B800h ;Сегментный адрес видеобуфера
        mov     es,ax  ;Отправим его в ES
        mov     byte ptr ES:0,'!' ;Отправим символ на 1-е знакоместо экрана
        mov     byte ptr ES:2,'!' ;Отправим символ на 2-е знакоместо экрана
        mov     ax,4c00h ;функция 4c00h успешное завершения программы
        int     21h    ;вызов DOS
code    ends    ;конец сегмента команд
stk     segment stack ;начало сегмента стека
        db     256 dup('*') ;резервируем 256 байт для стека
stk     ends    ;конец сегмента стека
        end     begin ;конец текста программы с точкой входа
```

# Пример

# ЛИСТИНГ

```

C:\ASM\ADDRESS\tasm\DIRECT3.LST                               Win-1251   Строка
1
2 0000          data      segment
3 0000 3333      dat       dw      3333h
4 0002          data      ends
5 0000          code      segment
6 0000 EB 03 90   begin:  jmp     bgn
7 0003 1111      mem       dw     1111h
8 0005 B8 0000s     bgn:    mov     ax,data
9 0008 8E D8      mem     mov     ds,ax
10 000A 2E: 8B 1E 0003r  mem    mov     bx,cs:mem
11 000F B4 00      mem    mov     ah,0
12 0011 B0 03      mem    mov     al,3
13 0013 CD 10      int    int     10h
14 0015 B8 B800     mem    mov     ax,0B800h
15 0018 8E C0     mem    mov     es,ax
16 001A 26: C6 06 0000 21  mem   mov     byte ptr ES:0,'!'
17 0020 26: C6 06 0002 21  mem   mov     byte ptr ES:2,'!'
18 0026 B8 4C00     mem    mov     ax,4C00h
19 0029 CD 21      int    int     21h
20 002B          code     ends
21 0000          stk      segment stack
22 0000 0100*(2A)  stk      db     256 dup('*')
23 0100          stk      ends
24          end      begin
;указали назначение сегментов
;начало сегмента данных
;конец сегмента данных
;начало сегмента команд
;адрес сегмента данных в AX
;затем перенесем из AX в DS
;номер функции установки режима дисплея
;код режима 80*25 (цветной)
;очистка экрана
;Сегментный адрес видеобuffers
;Отправим его в ES
;Отправим символ на 1-е знакоместо экрана
;Отправим символ на 2-е знакоместо экрана
;функция 4c00h успешное завершения программы
;вызов DOS
;конец сегмента команд
;начало сегмента стека
;резервируем 256 байт для стека
;конец сегмента стека
;конец текста программы с точкой входа

```

Symbol Name	Type	Value
??FILENAME	Text	"direct3 "
??VERSION	Number	040A
@CPU	Text	0101H
@CURSEG	Text	STK
@FILENAME	Text	DIRECT3
@WORDSIZE	Text	2
BEGIN	Near	CODE:0000
BGN	Near	CODE:0005
DAT	Word	DATA:0000
MEM	Word	CODE:0003

Groups & Segments	Bit	Size	Align	Combine	Class
CODE	16	002B	Para	none	
DATA	16	0002	Para	none	
STK	16	0100	Para	Stack	

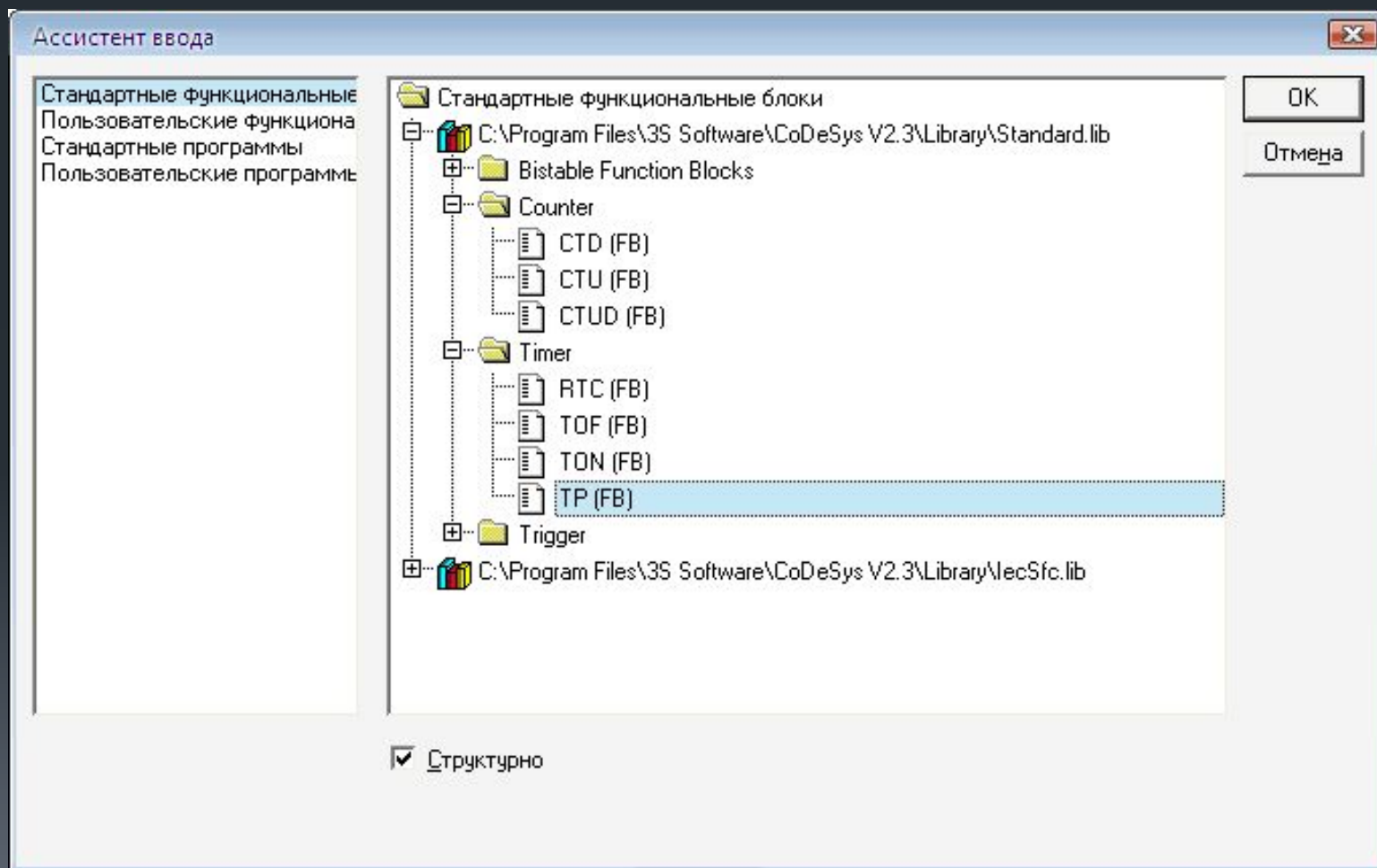
# Правила записи **переменной**



unsigned int	abc	12134
int	A	-31745
float	b	0.56
double	a_21	-8.75

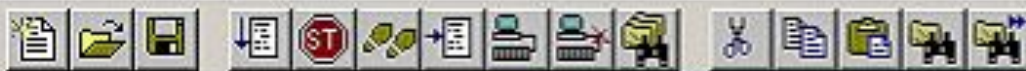


# Типы, которые можно выбрать, и классы переменных



CoDeSys - 6.signal.pro\*

Файл Правка Проект Вставить Дополнения Онлайн Окно Справка



Типы данных

- New\_type (STRUCT)

```
New_type
0001 TYPE New_type :
0002 STRUCT
0003 END_STRUCT
0004 EN_TYPE
0005
0006
0007
0008
0009
0010
```

Загрузка библиотеки 'C:\Program Files\3S Software\CoDeS  
Загрузка библиотеки 'C:\Program Files\3S Software\CoDeS  
Загрузка библиотеки 'C:\Program Files\3S Software\CoDeS  
Загрузка библиотеки 'C:\Program Files\3S Software\CoDeS

ОНЛАЙН ЗАМ ПРОСМОТР



Сетевые переменные в CoDeSys

# Часть CoDeSys

