

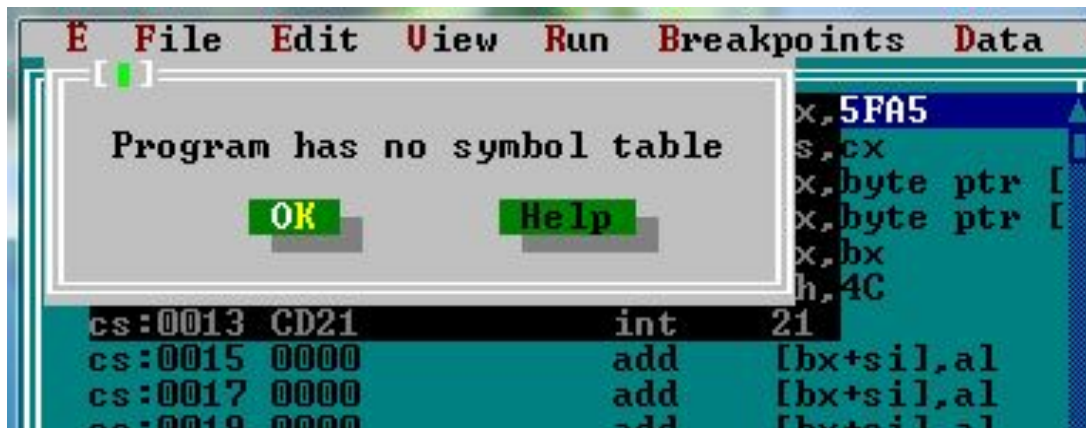
Лекция 9. Отладка программы

Отладчик **Td.exe** дает возможность контроля и управления процессором при исполнении команд программы:

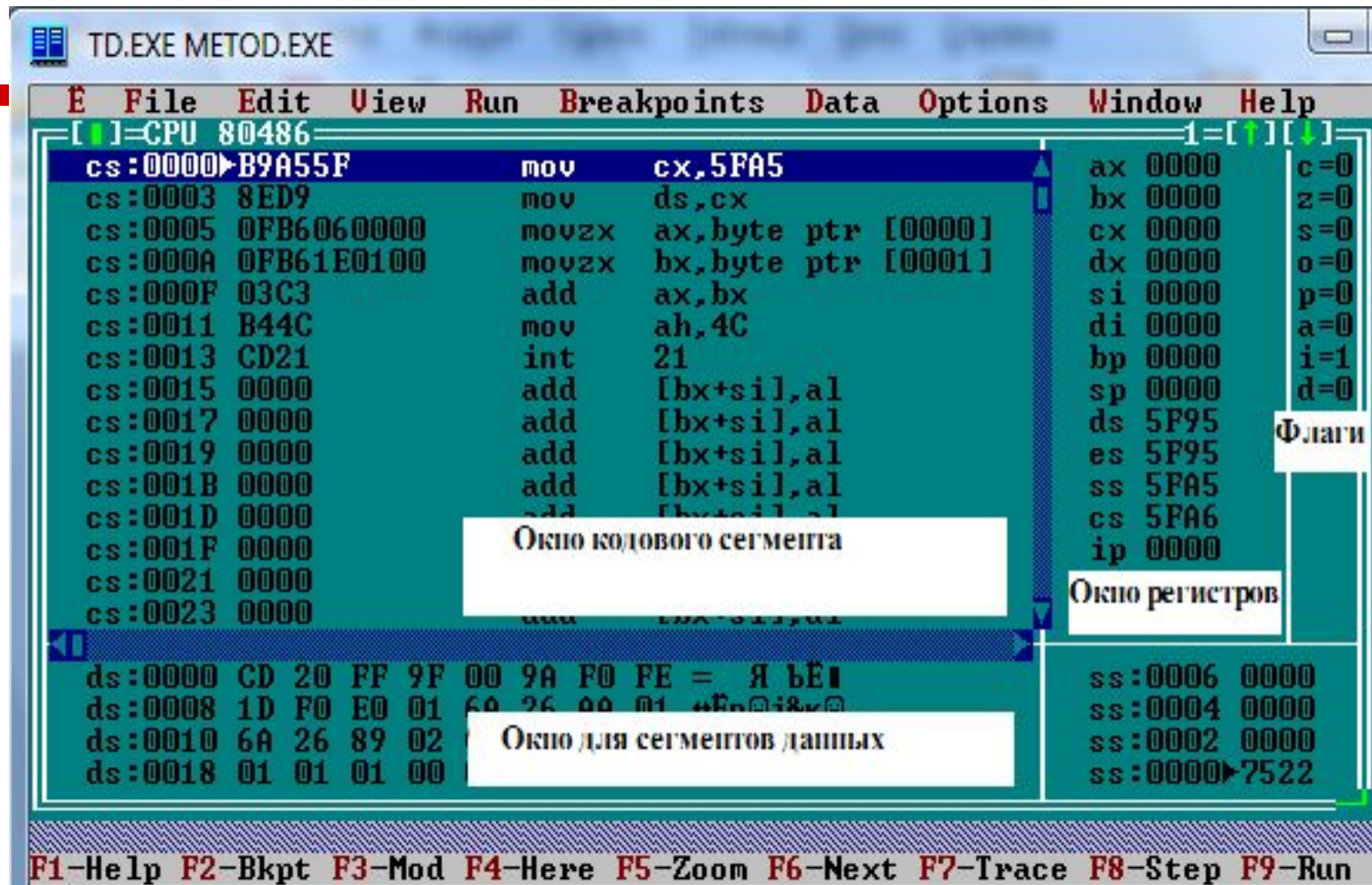
- **разные режимы исполнения команд:** пошаговое исполнение, с точками останова, до указанной команды
- просмотр и изменение данных в сегментах памяти
- просмотр и изменение содержимого регистров,
- просмотр и изменение флагов

Запуск отладчика из командной строки вместе с исполняемым кодом:

```
15 20:31 | METOD.EXE | Ev
s | 818 81
-
0> TD.EXE METOD.EXE
5Copy 6RenMov 7MkFo
```



Экран отладчика для ассемблерных программ



Окно кодового сегмента

```

E File Edit View Run Breakpoints Data Options Window H
[ ]-CPU 80486
cs:0000>B9A55F      mov     cx,5FA5      ax 0000
cs:0003 8ED9        mov     ds,cx        bx 0000
cs:0005 0FB6060000  movzx  ax,byte ptr [0000]  cx 0000
cs:000A 0FB61E0100  movzx  bx,byte ptr [0001]  dx 0000
cs:000F 03C3        add    ax,bx        si 0000
cs:0011 B44C        mov    ah,4C        di 0000
cs:0013 CD21        int    21          bp 0000
cs:0015 0000        add    [bx+si],al   sp 0000
cs:0017 0000        add    [bx+si],al   ds 5F95
cs:0019 0000        add    [bx+si],al   es 5F95
cs:001B 0000        add    [bx+si],al   ss 5FA5
cs:001D 0000        add    [bx+si],al   cs 5FA6
cs:001F 0000        add    [bx+si],al   ip 0000

```

```

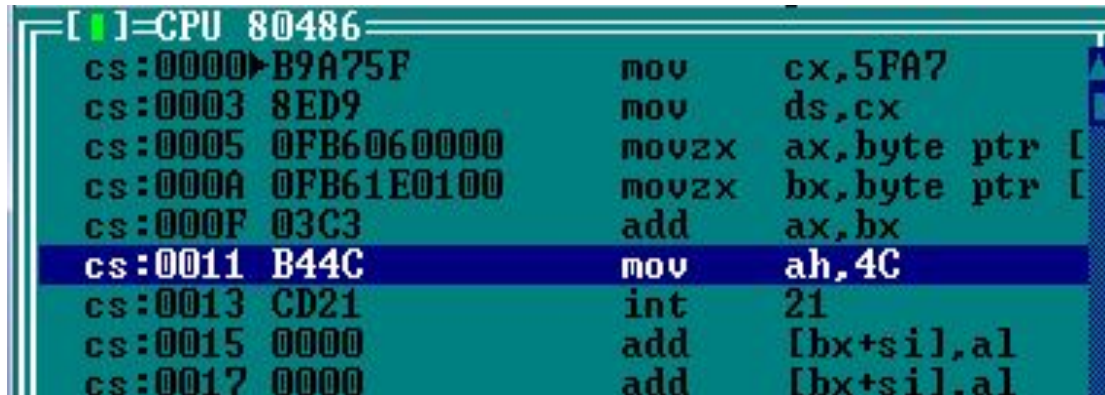
1 ; metod.asm - Получить 2-х байтную сумму двух однобайтных кодов
2 ;.386
3 0000 dseg segment use16
4 0000 22 a db 34
5 0001 75 b db 75h
6 0002 dseg ends
7 0000 cseg segment use16
8 assume ds:dseg, cs:cseg
9 0000 B9 0000 s m1: mov cx, dseg
10 0003 8E D9 mov ds, cx
11 ; расширим байты до 2-х байтных и сложим
12 0005 0F B6 06 0000 r movzx ax, ds:a
13 000A 0F B6 1E 0001 r movzx bx, ds:b
14 000F 03 C3 add ax, bx
15 ; завершение исполнения
16 0011 B4 4C mov ah, 4ch
17 0013 CD 21 int 21h
18 0015 cseg ends
19 end m1

```

16-разрядные
внутрисегментные адреса

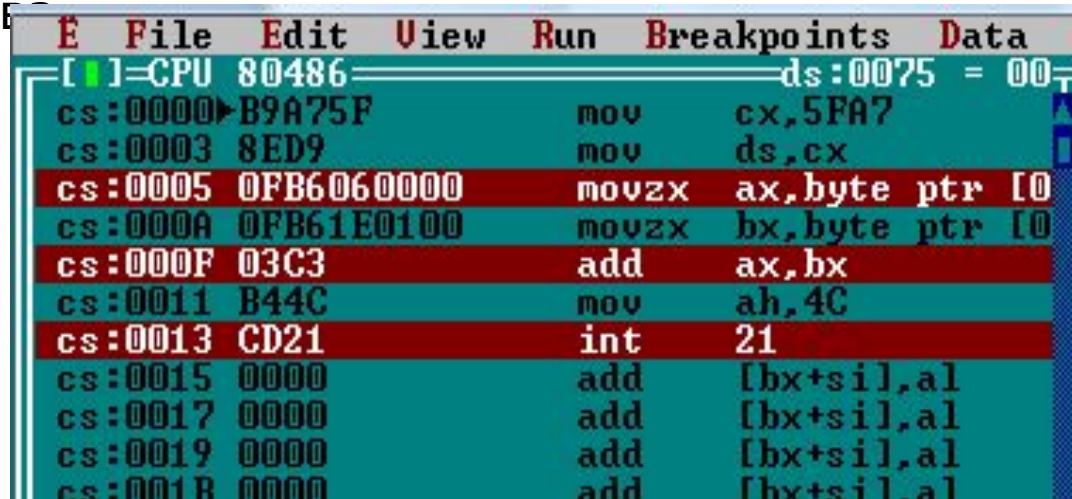
Управление режимом исполнения команд

- **F7 (пошаговый режим)** - исполнение одной команды, адрес которой в кодовом сегменте задан регистром IP. После исполнения команды отладчик приостанавливает процессор.
- **F8 (пошаговый режим)** отличается от F7 тем, что исполнение команд в процедурах происходит без остановок.
- **F4 (исполнение до курсора)** - непрерывное исполнение последовательности команд до команды, на который заранее установлен курсор.



```
[ ]=CPU 80486
cs:0000>B9A75F      mov     cx,5FA7
cs:0003 8ED9        mov     ds,cx
cs:0005 0FB6060000     movzx  ax,byte ptr [
cs:000A 0FB61E0100     movzx  bx,byte ptr [
cs:000F 03C3          add    ax,bx
cs:0011 B44C          mov    ah,4C
cs:0013 CD21          int    21
cs:0015 0000          add    [bx+si],al
cs:0017 0000          add    [bx+si],al
```

- F9 – непрерывное исполнение команд. Не позволяет отлаживать программу. Пользоваться режимом F9 при отладке можно только в сочетании с точками останова.
- F2 – установка/снятие точки останова. Выбрать курсором нужную команду и нажать F2. Команда будет помечена красной строкой -«точка останова». Повторное нажатие F2 снимет точку останова.



The screenshot shows a debugger window with a menu bar (E, File, Edit, View, Run, Breakpoints, Data) and a status bar ([]=CPU 80486 ds:0075 = 00). The main area displays assembly code with the following instructions:

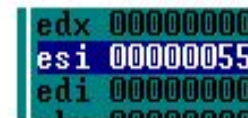
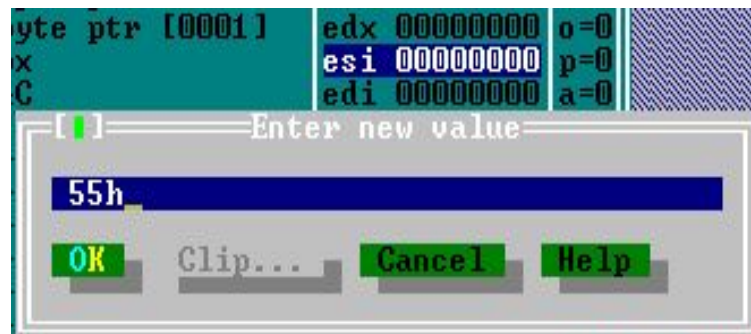
Address	Instruction
cs:0000	mov cx,5FA7
cs:0003	mov ds,cx
cs:0005	movzx ax,byte ptr [0]
cs:000A	movzx bx,byte ptr [0]
cs:000F	add ax,bx
cs:0011	mov ah,4C
cs:0013	int 21
cs:0015	add [bx+si],al
cs:0017	add [bx+si],al
cs:0019	add [bx+si],al
cs:001B	add [bx+si],al

The instruction at address cs:000F (add ax,bx) is highlighted in red, indicating a breakpoint.

Окно регистров. Локальное меню действий



Локальное меню действий для окна регистров



Окно сегментов данных.

Настройка окна на область памяти: GOTO ...

- До настройки окна на ваш сегмент данных надо выполнить команды загрузки регистра-указателя этого сегмента



The screenshot shows a debugger's data segment window with a context menu open. The background is a teal color with white text. The context menu is white with black text and a black border. The menu items are: Goto..., Search..., Next, Change..., Follow, Previous, Display as, and Block. The 'Follow' and 'Display as' items have small right-pointing triangles next to them. The background text shows memory addresses and values for segments cs, ds, and int.

```
cs:0013 CD21 int 21
cs:0015 0000 add [bx+sil,al
cs:0017 0000 [bx+sil,al
cs:0019 0000 [bx+sil,al
cs:001B 0000 [bx+sil,al
cs:001D 0000 [bx+sil,al
cs:001F 0000 [bx+sil,al
cs:0021 0000 [bx+sil,al
cs:0023 0000 [bx+sil,al

ds:0000 CD 20 F E = Я ЬЕ!
ds:0008 1D F0 E 1 +Ер@j&к©
ds:0010 6A 26 8 2 j&Йе† 7†
```


На скрине 1 –кодový сегмент. Команды загрузки указателя сегмента DS выполнены. Следующей будет считываться и исполняться команда по адресу CS:0005

На скрине 1 и 2: настраиваем окно сегментов данных на начало нашего сегмента. Заносим по адресам DS:0000 и DS:0001 коды данных

The image shows a DOS debugger window with the following assembly code and registers:

Address	Code	Comment	Register	Value
cs:0000	B9A5F	mov cx,5FA5	eax	00000000
cs:0003	8ED9	mov ds,cx	ebx	00000000
cs:0005	0FB6060000	movzx ax,byte ptr [00	ecx	00005FA5
cs:000A	0FB61E0100	movzx bx,byte ptr [00	edx	00000000
cs:000F	03C3	add ax,bx	esi	00000000
cs:0011	B44C	mov ah,4C	edi	00000000
cs:0013	CD21	int 21	ebp	00000000
cs:0015	0000	add [bx+sil],al	esp	00000000
cs:0017	0000	add [bx+sil],al	ds	5FA5
cs:0019	0000	add [bx+sil],al	es	5F95
cs:001B	0000	add [bx+sil],al	fs	0000
cs:001D	0000			
cs:001F	0000			
cs:0021	0000			
cs:0023	0000			

Registers: eax=00000000, ebx=00000000, ecx=00005FA5, edx=00000000, esi=00000000, edi=00000000, ebp=00000000, esp=00000000, ds=5FA5, es=5F95, fs=0000.

A dialog box is open with the title "Enter address to position to" and the text "ds:0_". The dialog has "OK", "Clip...", and "Cancel" buttons.

The data segment window shows the following data:

Address	Code	Comment	Register	Value
ds:0000	22 75 00 00 00 00 00 00	"u"	ss	0006 00
ds:0008	00 00 00 00 00 00 00 00		ss	0004 00
ds:0010	B9 A5 5F 8E D9 0F B6 06	! e_0^*	ss	0002 00
ds:0018	00 00 0F B6 1E 01 00 03	* ^@	ss	0000 75

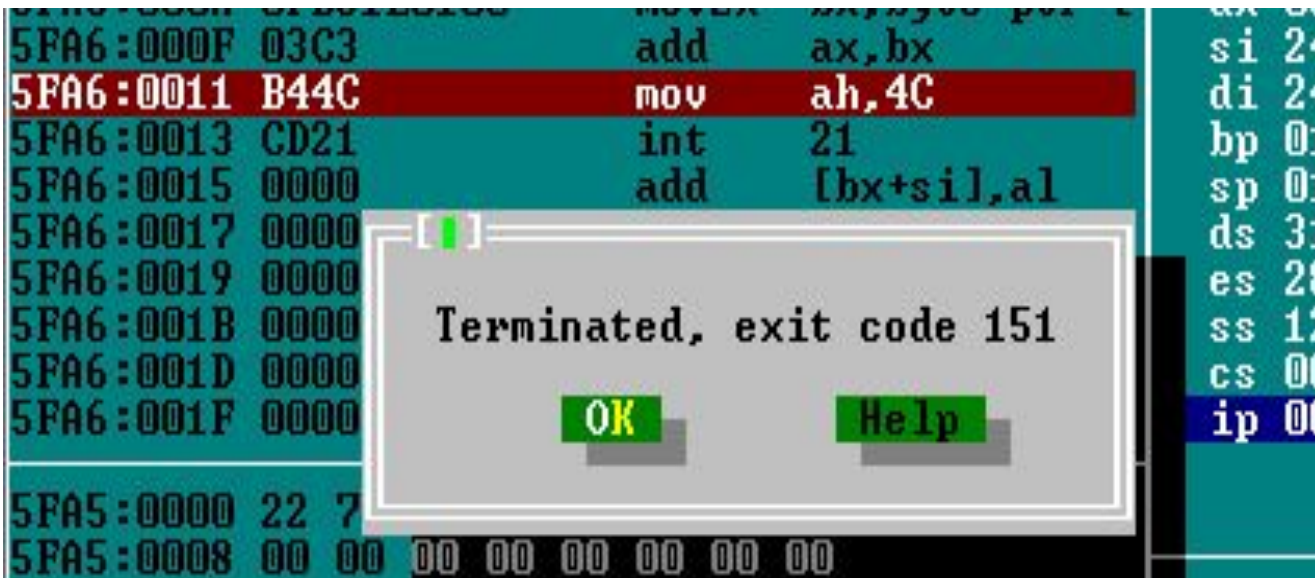
Занесение /Изменение данных в памяти

- В окне сегмента данных встать на байт курсором, выбрать опцию *Change* или просто нажать Пробел. Появится окно для задания значения в hex-коде.



Отладка без перезагрузки программы

- При отладке программы не давайте выгрузить ее из памяти, пока не закончите работать с нею. То есть, не давайте процессору выполнять команды для выгрузки из памяти, а верните его к желаемой команде



The screenshot shows a debugger window with assembly code. The code is as follows:

```
5FA6:000F 03C3      add     ax,bx
5FA6:0011 B44C      mov     ah,4C
5FA6:0013 CD21      int     21
5FA6:0015 0000      add     [bx+si],al
5FA6:0017 0000
5FA6:0019 0000
5FA6:001B 0000
5FA6:001D 0000
5FA6:001F 0000
```

On the right side, a register window shows the following values:

```
ax 00
si 24
di 24
bp 01
sp 01
ds 31
es 28
ss 12
cs 00
ip 00
```

A dialog box is overlaid on the code, displaying the message "Terminated, exit code 151" and two buttons: "OK" and "Help".

Изменить адрес следующей команды для процессора

- Измените вручную значение регистра IP, содержащего внутрисегментный адрес следующей команды для чтения из памяти

The image shows a debugger window for CPU 80486. The main window displays a list of instructions with their addresses and the current state of registers. The instruction at address 0011 is highlighted, and the IP register is shown as 0011. A dialog box is open, allowing the user to enter a new value for the IP register, with the value 5 entered.

```
File Edit View Run Breakpoints Data Options Wind
[ ]-CPU 80486
cs:0000 B9A55F mov cx,5FA5 ax 0097
cs:0003 8ED9 mov ds,cx bx 0075
cs:0005 0FB6060000 movzx ax,byte ptr [ cx 5FA5
cs:000A 0FB61E0100 movzx bx,byte ptr [ dx 0000
cs:000F 03C3 add ax,bx si 0000
cs:0011 B44C mov ah,4C di 0000
cs:0013 CD21 int 21 bp 0000
cs:0015 0000 add [bx+sil],al sp 0000
cs:0017 0000 add [bx+sil],al ds 5FA5
cs:0019 0000 add [bx+sil],al es 5F95
cs:001B 0000 add [bx+sil],al ss 5FA5
cs:001D 0000 add [bx+sil],al cs 5FA6
cs:001F 0000 add [bx+sil],al ip 0011

es:0000 CD 20 FF 9F 00 9A
es:0008 1D F0 E0 01 6A 26
es:0010 6A 26 89 02 C5 20
es:0018 01 01 01 00 02 FF

Enter new value
5
OK Clip...
```