4. Команды сложения и вычитания целых чисел

Сложение двух целых чисел реализуется командой ADD, которая помещает сумму на место первого операнда:

```
ADD   op1, op2     ; op1 = op1 + op2
```

Аналогично – вычитание с помощью команды SUB:

```
SUB op1, op2       ; op1 = op1 - op2
```

Правила использования этих команд:

- операнды обязательно должны иметь одну длину (байт с байтом, слово со словом)
- размещение операндов аналогично команде MOV: операнд 2 – это регистр, память или непосредственная константа, а операнд 1 – только регистр или память
- запрещается оба операнда задавать в памяти

Примеры использования команд сложения:

```
B1     DB     3   ;  размещение в памяти числа 3
W1     DW    1000   ; размещение в памяти числа 1000
. . . . . .
MOV AL, B1        ;  (AL) = 3
ADD  AL, 2        ;  (AL) = (AL) + 2 = 5
SUB   AL, 1       ;  (AL) = (AL) – 1 = 4
ADD  AL, B1       ;  (AL) = (AL) + 3 = 7
ADD  AL, 1000     ; Ошибка: операнды не согласованы по длине!
ADD  B1, B1       ; Ошибка: оба операнда находятся в памяти!
```

В тех случаях, когда операнд надо увеличить или уменьшить только на 1. можно использовать более короткие и быстрые команды инкремента и декремента:

```
INC   op          ; op = op + 1
DEC  op           ; op = op - 1
```

Здесь op – либо регистр, либо именованная область памяти.

При использовании команд сложения и вычитания могут возникать **особые случаи**.

Во-первых, если при сложении результат превосходит отведенное ему число разрядов, то ошибка **НЕ** фиксируется, выполнение программы **НЕ** прекращается, но результат будет неправильный. Например, при сложении двух однобайтовых чисел 255 и 2 полученное число 257 не “помещается” в один байт:

$255 + 2 = 1111\ 1111_2 + 0000\ 0010_2 = 257 = 1\ 0000\ 0001_2$

Сумма будет интерпретирована как число 1.

Вывод: проверка правильности выполнения операций возлагается на программиста!

Для отслеживания особых ситуаций можно использовать биты регистра Flags. В частности, бит CF (Carry Flag, флаг переноса) устанавливается в 1 при появлении переноса из старшего разряда за пределы разрядной сетки. Для “отлавливания” этой ситуации можно воспользоваться специальными командами условного перехода.

Аналогичная ситуация может возникнуть при вычитании беззнаковых чисел, когда уменьшаемое меньше вычитаемого, в результате получается отрицательное число и эта ситуация тоже фиксируется значением CF=1.

Свои особенности имеет обработка знаковых чисел. Как известно, для представления отрицательных чисел используется дополнительный код, и здесь тоже возможен выход за пределы разрядной сетки, что еще больше усугубляется использованием крайнего левого разряда для представления знака числа. Например, сложение чисел (+127) и (+2) дает (+129), что интерпретируется, как число (-127). Такая ошибка называется переполнением мантиссы и фиксируется установкой флага OF=1, что также надо проверять в программе (OF – Overflow Flag, флаг переполнения).

При использовании команд сложения и вычитания также могут меняться флаги ZF (Zero Flag, флаг нуля) и SF (Sign Flag, флаг знака):

ZF=1, если результат операции равен 0, а SF=1, если получен отрицательный результат.

Практические задания к теме №4.

Задание 1. Реализовать простейшую программу, которая складывает два числа без знака и из получившейся суммы вычитает третье: N + M - L. Результат занести на место третьего операнда. Полный текст программы:

```
assume cs:cod,  ds:data
data    SEGMENT
N       db   ?
M       db   ?
L       db   ?
data    ENDS
cod     SEGMENT
main:   mov  N, 10
        mov  M, 20
        mov  L, 15
        mov  AH, N
        mov  BH, M
        add   AH, BH
        mov  BH, L
        sub   AH, BH
cod     ENDS
        END    main
```

Порядок работы.

1. С использованием текстового редактора ввести текст программы и запомнить его в файле с расширением. asm (например - ex1.asm).

2. Выполнить трансляцию программы. Для этого необходимо запустить процессор командной строки Command и набрать команду

tasm/zi ex1, ex1, ex1

В результате будет создан объектный файл ex1.obj, файл листинга ex1.lst, файл перекрестных ссылок ex1.crf. Здесь опция /zi необходима для включения в объектный файл отладочной информации для последующего использования отладчика. В таблице перекрестных ссылок указывается номер строки, в которой определен каждый идентификатор, и номера тех строк, в которых есть ссылки на него. Если файл листинга или перекрестных ссылок не нужен, второе и третье имя можно не задавать.

3. Построить исполняемый файл. Для этого вызвать компоновщик tlink командой

tlink/v ex1

В результате на диске будет построен исполняемый файл ex1.exe и листинг распределения памяти ex1.map. Листинг распределения памяти содержит сводные сведения о сегментах программы. Для каждого сегмента указывается адрес начала и конца, длина сегмента в байтах, его имя и категория.

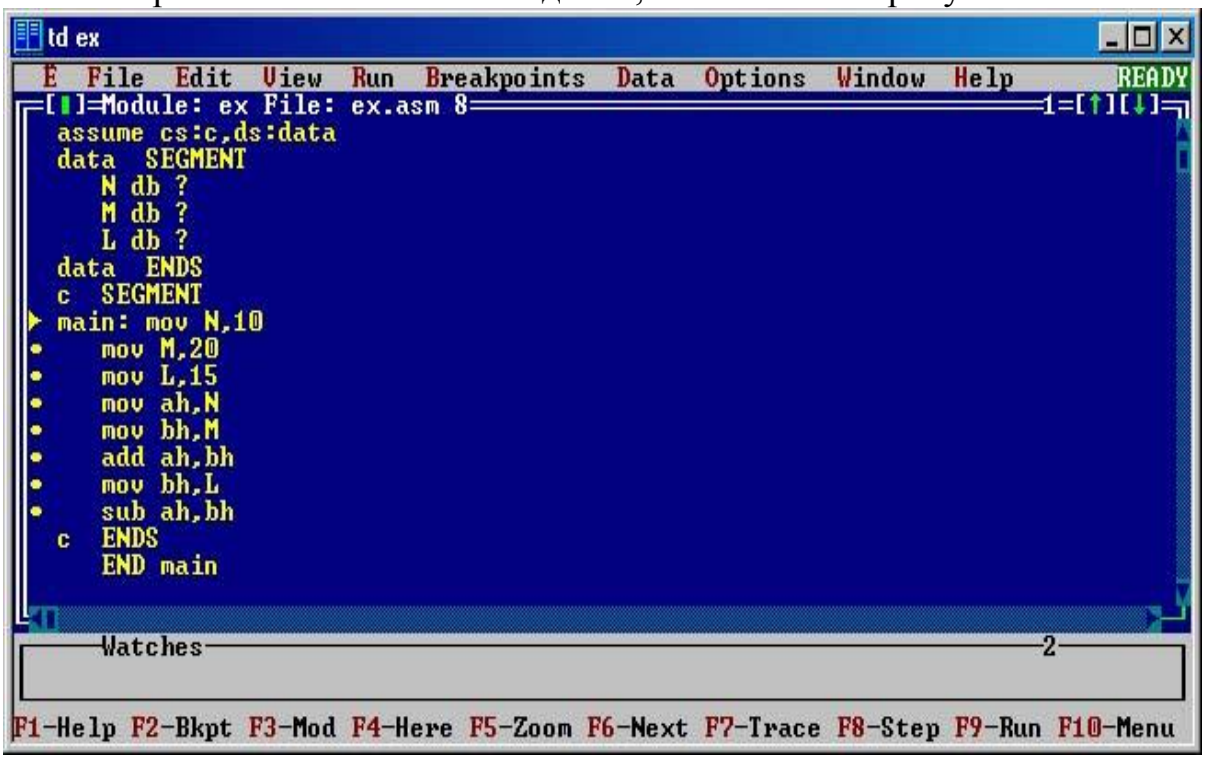
4. Выполнение программы

Выполняемый файл можно вызвать как из командной строки DOS, так и выполнить под управлением отладчика TD (Turbo Debugger). Для выполнения под управлением отладчика необходимо:

4.1. Загрузить программу под управлением отладчика, введя в командной строке команду

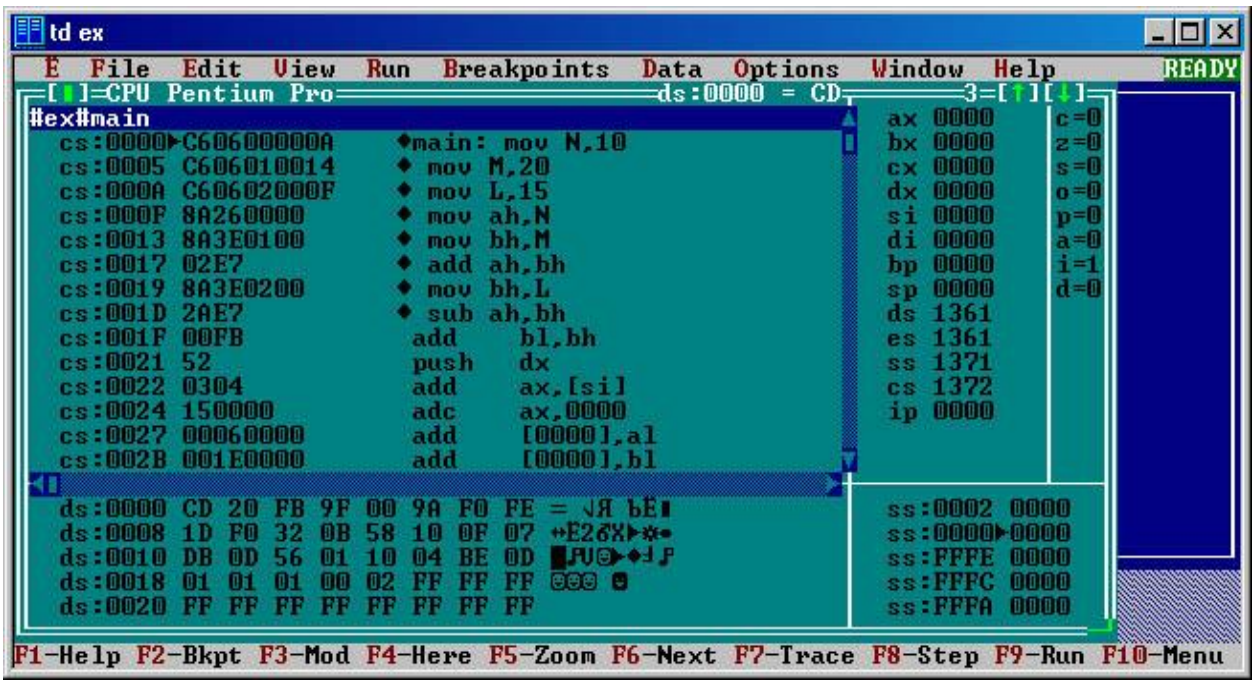
td ex1

На экране появится окно отладчика, показанное на рисунке



4.2. При работе с отладчиком выполнить следующие действия:

- Развернуть окно на весь экран (Alt+Enter)
- Активизировать основное меню мышью или по F10
- В меню View выбрать команду CPU для вывода окна состояния процессора; это окно содержит пять зон:
  - зона кода с командами на ассемблере и их машинными аналогами
  - зона регистров процессора
  - зона флагов регистра Flags
  - зона сегмента данных
  - зона состояния стека
- переход между зонами – по клавишам Tab или Shift+Tab; окно CPU позволяет просматривать изменение состояния процессора при пошаговом выполнении программы.



- по клавише F7 выполнить последовательно все команды программы, отслеживая при этом изменения содержимого регистров и сегмента данных
- для выхода из отладчика нажать Alt+X