**Цель:** изучение правил адресации команд и данных.

Список заданий.

1. Содержимое первого байта памяти данных, представленных в окне наблюдения двумя первыми слева шестнадцатеричными цифрами в исходной строке, переместить в первый байт памяти данных, представленных в окне наблюдения двумя первыми слева шестнадцатеричными цифрами результирующей строки. Затем аналогично содержимое памяти 2-го байта переместить во 2-ой байт памяти и т. д.
2. Из первого байта памяти данных, представленных в исходной строке, переместить данные в восьмой байт результирующей строки. Затем 2-ой байт переместить в 7-ой и т. д.
3. Если число в восьмом байте исходной строки больше 5, то переместить это число в восьмой байт результирующей строки. Иначе обнулить восьмой байт. Затем 7-ой байт переместить в 7-ой при выполнении условий, аналогичных рассмотренным выше условиям и т. д.
4. Первый байт исходной строки переместить во второй байт результирующей строки. Затем второй байт исходной строки переместить в первый байт результирующей строки. Аналогично переместить третий и четвертый байты исходной строки и т. д.

**Задание 1**

**Текст задания:**1) Содержимое первого байта памяти данных, представленных в окне наблюдения двумя первыми слева шестнадцатеричными цифрами в исходной строке, переместить в первый байт памяти данных, представленных в окне наблюдения двумя первыми слева шестнадцатеричными цифрами результирующей строки. Затем аналогично содержимое памяти 2-го байта переместить во 2-ой байт памяти и т. д.

Текст программы:

TITLE LAB3

.MODEL SMALL

.STACK 100h

.CODE

data segment

a DB 0,1,2,3,4,5,6,7; резервируем 8 ячеек для исходных данных и заполняем их значениями

b DB 8 dup (?); резервируем 8 ячеек для результирующих данных и не заполняем их

data ends

START:

mov ax, data

mov ds, ax; инициализируем регистр ds адресом начала сегмента данных

lea si, a; записываем эффективный адрес исходных данных в регистр si

lea di, b; записываем эффективный адрес исходных данных в регистр di

mov cx,8; счетчик для команды loop

loop1:

mov al, [si]; записываем в младшую часть регистра ax данные, на которые указывает адрес в регистре si

mov [di], al; записываем данные из al в ячейку результирующей последовательности, на которую ссылается адрес регистра di

inc si

inc di

loop loop1; повторяем действия в метке loop1, пока cx != 0

mov ax,4c00h

int 21h; вызов прерывания

END START

**Проиллюстрируем работу программы**

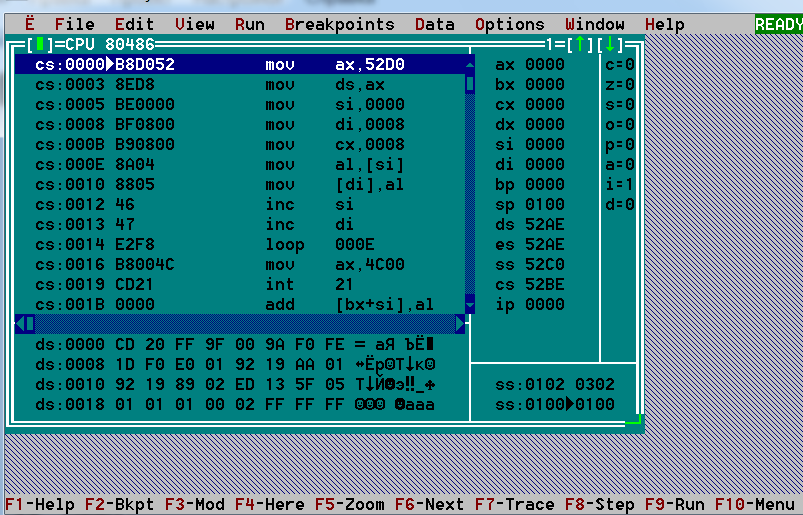


Рис. 1.1 – До отладки программы

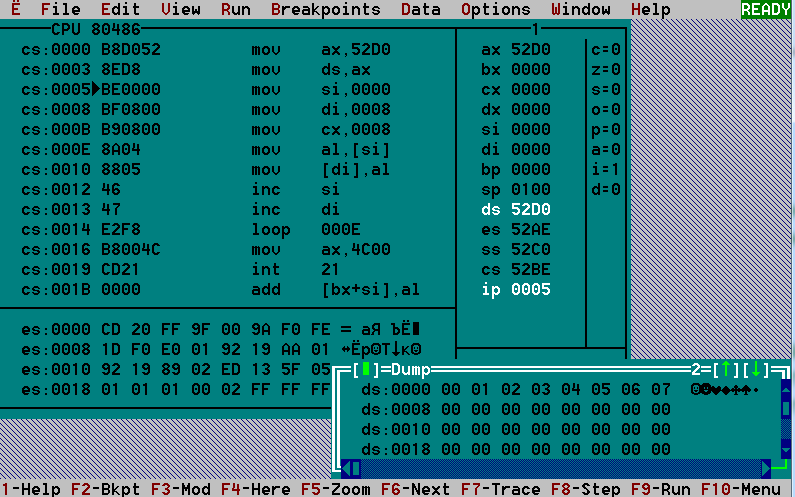


Рис. 1.2– Промежуточное выполнение программы. Заполнение элементов первого массива.

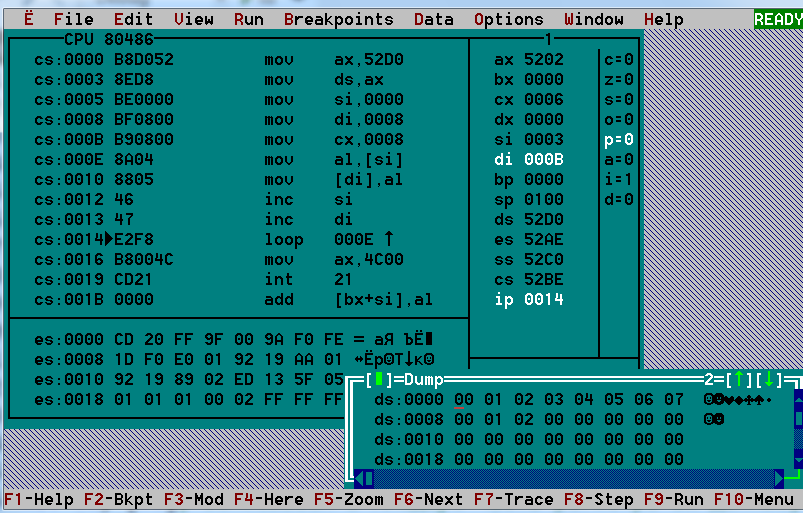


Рис. 1.3 – Промежуточное выполнение программы. Начало заполнения элементов второго массива.

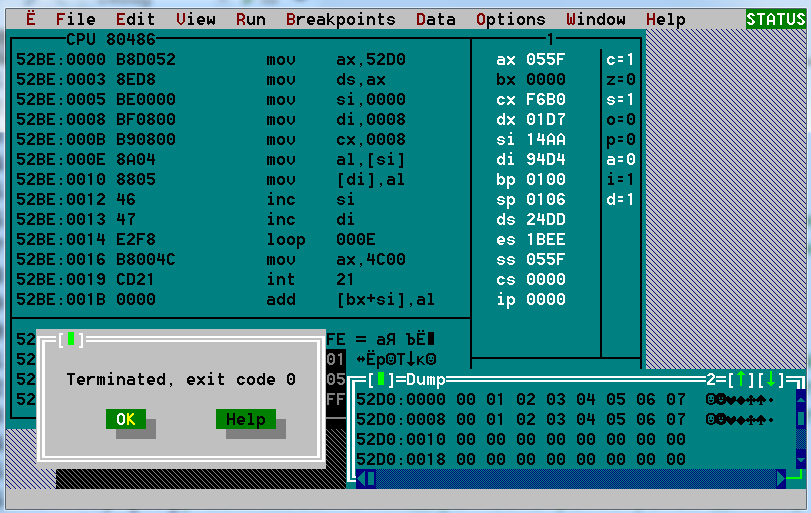


Рис. 1.4 – После отладки программы

В результате работы программы был проинициализирован массив чисел от 0 до 7 в начале сегмента данных (ds:0000), а также был создан, но не инициализирован пустой массив (ds:0008).

Байты 0 – 7 из начала сегмента данных были последовательно копированы в байты 8-15 сегмента данных. Для реализации задачи была использована метка loop1. Далее последовательность команд повторяется в цикле с использованием команды loop.

**Задание 2**

1) Из первого байта памяти данных, представленных в исходной строке, переместить данные в восьмой байт результирующей строки. Затем 2-ой байт переместить в 7-ой и т. д.

Текст программы:

TITLE LAB3

.MODEL SMALL

.STACK 100h

.CODE

data segment

a DB 0,1,2,3,4,5,6,7; резервируем 8 ячеек для исходных данных и заполняем их значениями

b DB 8 dup (?); резервируем 8 ячеек для результирующих данных и не заполняем их

data ends

START:

mov ax, data

mov ds, ax; инициализируем регистр ds адресом начала сегмента данных

lea si, a; записываем эффективный адрес исходных данных в регистр si

lea di, b+7; устанавливается смещение для участка памяти, выделенную под запись, чтобы данные копировались с конца в обратном порядке

mov cx,8; счетчик для команды loop

loop1:

mov al, [si]; записываем в младшую часть регистра ax данные, на которые указывает адрес в регистре si

mov [di], al; записываем данные из al в ячейку результирующей последовательности, на которую ссылается адрес регистра di

inc si

dec di

loop loop1; повторяем действия в метке loop1, пока cx != 0

mov ax,4c00h

int 21h; вызов прерывания

END START

**Проиллюстрируем работу программы**

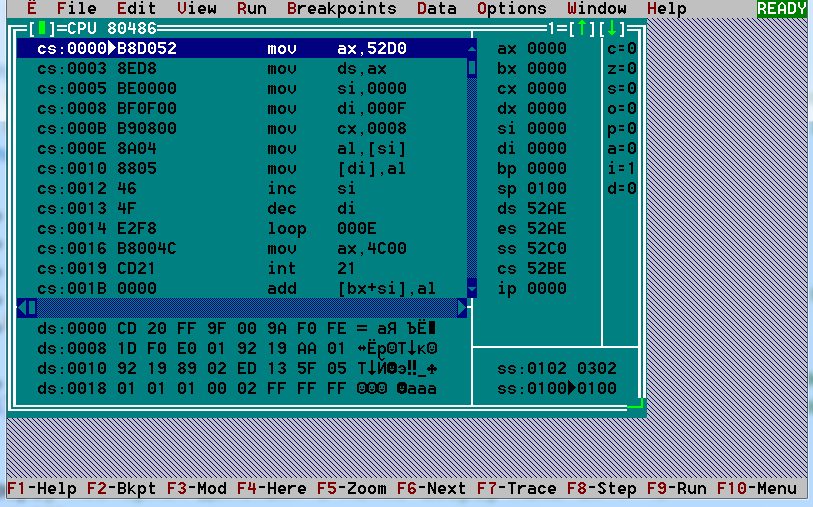
л

Рис. 2.1 – Программа до отладки

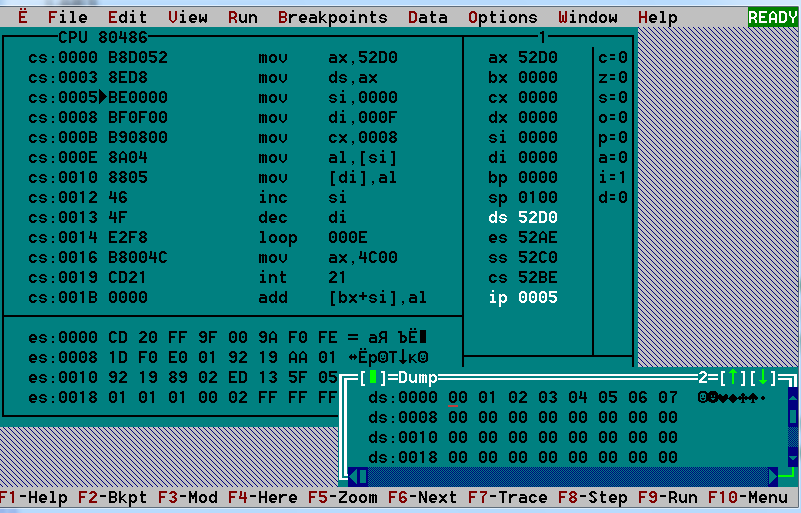


Рис. 2.2 – Промежуточное выполнение программы. Заполнение элементами первого массива.

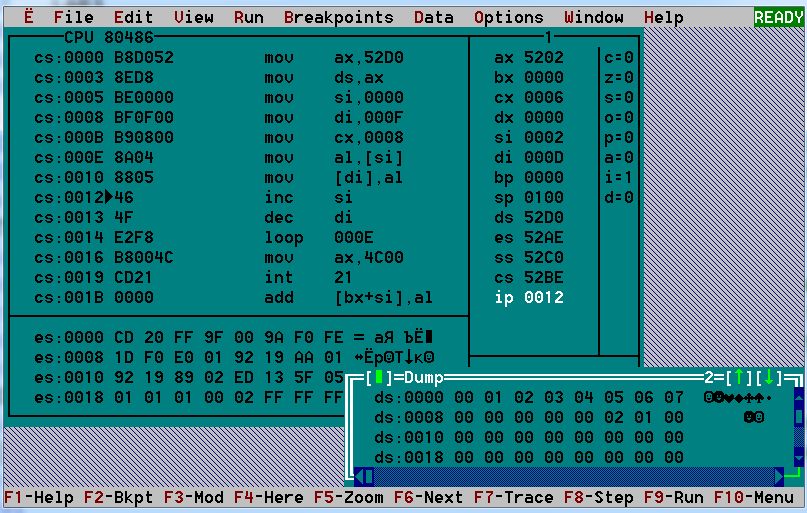


Рис. 2.3– Промежуточное выполнение программы. Начало заполнения элементами второго массива.

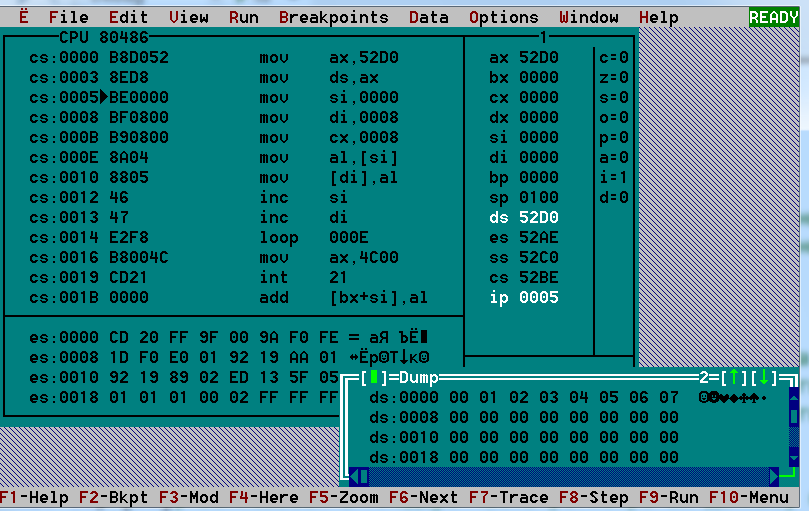


Рис. 2.4 – Программа после отладки

В результате работы программы был проинициализирован массив чисел от 0 до 7 в начале сегмента данных (ds:0000), а также был создан, но не инициализирован пустой массив (ds:0008).

Байты 0-7 из начала сегмента данных были последовательно копированы в байты 8-15 сегмента данных в другом порядке, начиная с байта 15. Для реализации задачи была использована метка label1. Далее последовательность команд повторяется в цикле с использованием команды loop.

**Задание 3**

Если число в восьмом байте исходной строки больше 5, то переместить это число в восьмой байт результирующей строки. Иначе обнулить восьмой байт. Затем 7-ой байт переместить в 7-ой при выполнении условий, аналогичных рассмотренным выше условиям и т. д.

Текст программы:

TITLE LAB3

.MODEL SMALL

.STACK 100h

.CODE

data segment

a DB 0,1,2,3,4,5,6,7; резервируем 8 ячеек для исходных данных и заполняем их значениями

b DB 8 dup (?); резервируем 8 ячеек для результирующих данных и не заполняем их

data ends

START:

mov ax, data

mov ds, ax; инициализируем регистр ds адресом начала сегмента данных

mov cx,8; счетчик для команды loop

lea si, a+7;

lea di, b+7; устанавливается смещение для участка памяти, выделенную под запись, чтобы данные копировались с конца в обратном порядке

loop1:

mov al, [si]; записываем в младшую часть регистра ax данные, на которые указывает адрес в регистре si

cmp al,5; сравнение значения в байте со значением 5

jg higher\_than\_5; если значение строго больше 5, то перепрыгиваем на метку higher\_than\_5, для прохождения цикла

mov al,0

higher\_than\_5:

mov [di],al

dec si

dec di

loop loop1; повторяем действия в метке loop1, пока cx != 0

mov ax,4c00h

int 21h; вызов прерывания

END START

**Проиллюстрируем работу программы**

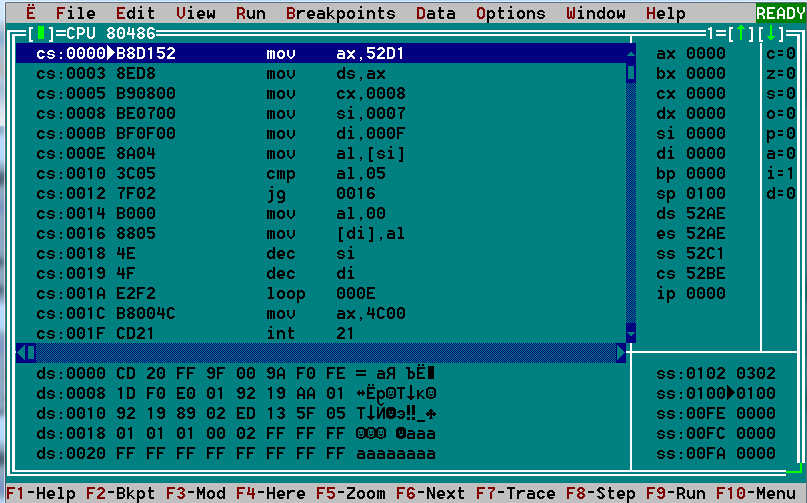


Рис. 3.1 – Программа до отладки

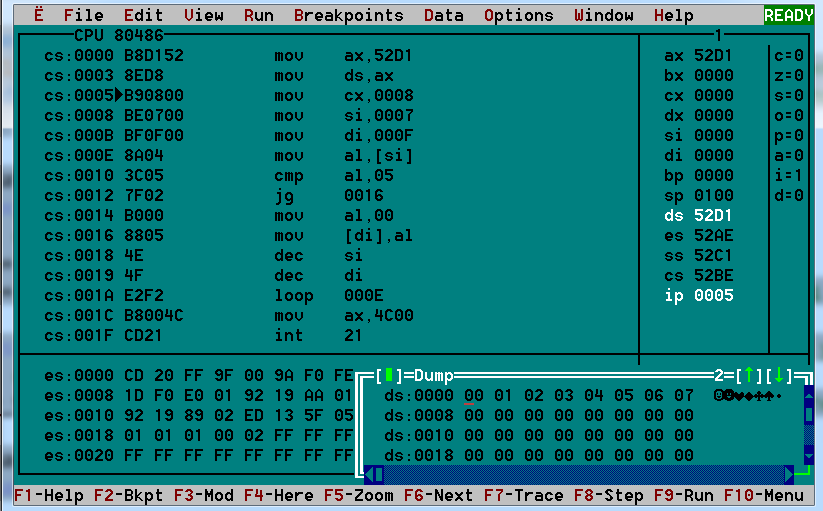


Рис. 3.2 – Промежуточное выполнение программы. Заполнение элементами первого массива.

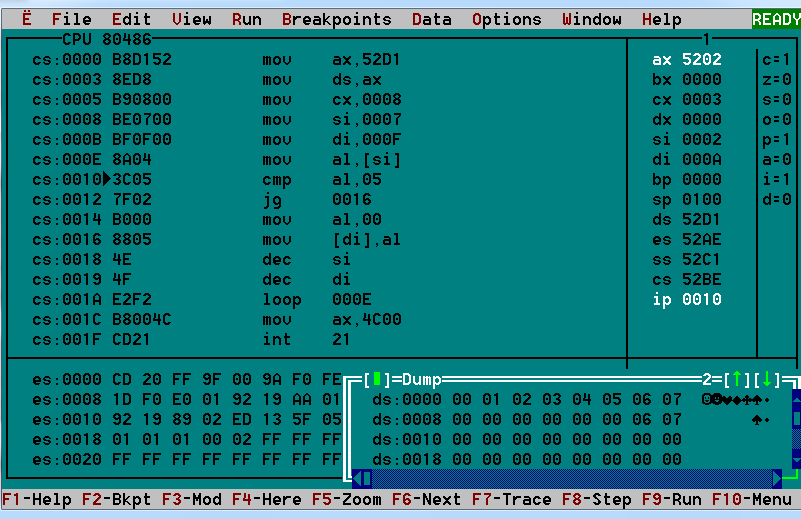


Рис. 3.2 – Промежуточное выполнение программы. В процессе заполнения элементами второго массива.

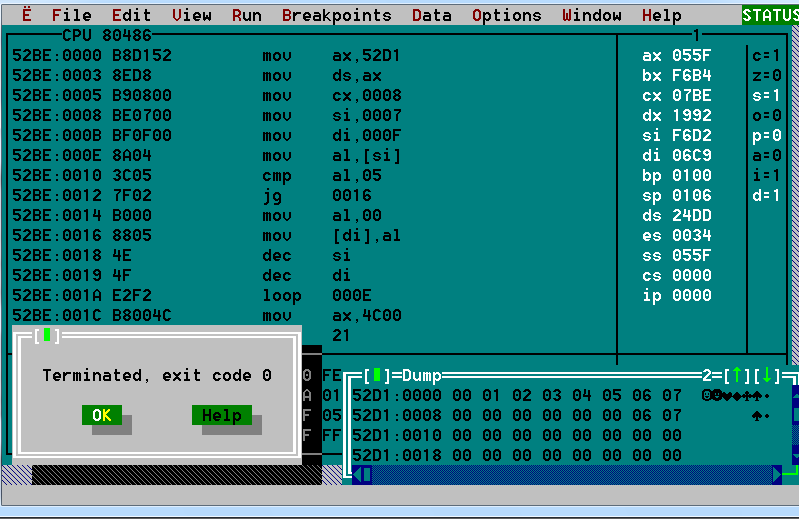


Рис. 3.3 – Программа после отладки

В результате работы программы был проинициализирован массив чисел от 0 до 7 в начале сегмента данных (ds:0000), а также был создан, но не инициализирован пустой массив (ds:0008).

Заполнение идет уже немного по-другому. Прежде чем заполнить ячейку каким-то значением, нужно проверить это значение на условие: значение > 5. Если число в байте исходной строки больше 5, то переместить это число в байт результирующей строки. Иначе обнулить байт.

**Задание 4**

Первый байт исходной строки переместить во второй байт результирующей строки. Затем второй байт исходной строки переместить в первый байт результирующей строки. Аналогично переместить третий и четвертый байты исходной строки и т. д.

Текст программы:

TITLE LAB3

.MODEL SMALL

.STACK 100h

.CODE

data segment

a DB 0,1,2,3,4,5,6,7; резервируем 8 ячеек для исходных данных и заполняем их значениями

b DB 8 dup (?); резервируем 8 ячеек для результирующих данных и не заполняем их

data ends

START:

mov ax, data

mov ds, ax; инициализируем регистр ds адресом начала сегмента данных

mov cx,8; счетчик для команды loop

lea si, a;

lea di, b;

loop1:

mov al, [si]

mov [di+1], al

mov al, [si+1]

mov [di], al

add si, 2; увеличить индекс источника на 2

add di, 2; увеличить индекс приемника на 2

loop loop1; повторяем действия в метке loop1, пока cx != 0

mov ax,4c00h

int 21h; вызов прерывания

END START

**Проиллюстрируем работу программы**

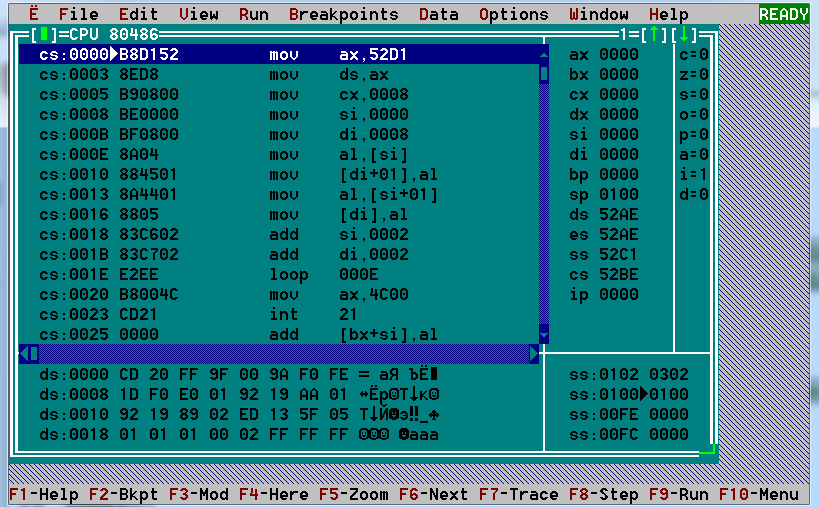


Рис. 4.1– Программа до отладки

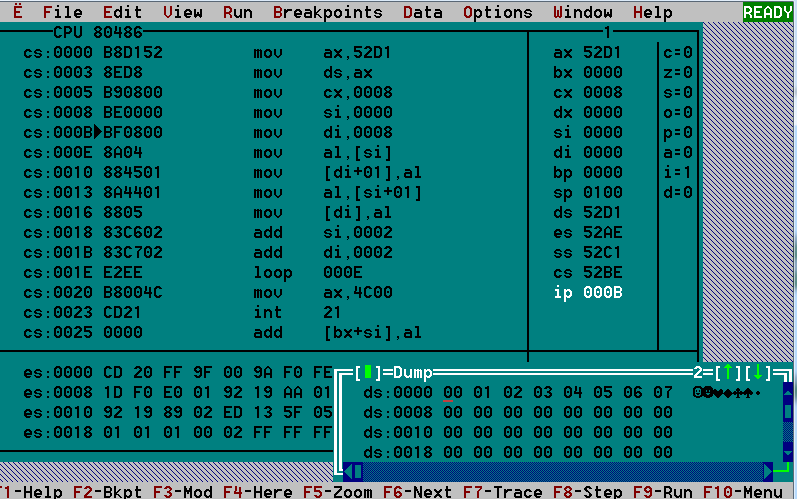


Рис. 4.2 – Промежуточное выполнение программы. Заполнение элементами первого массива.

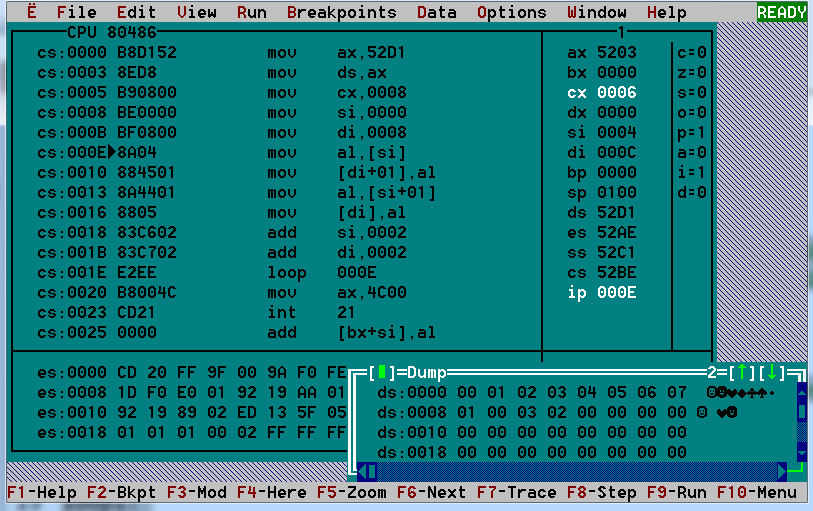


Рис. 4.3 – Промежуточное выполнение программы. В процессе заполнения элементами второго массива.

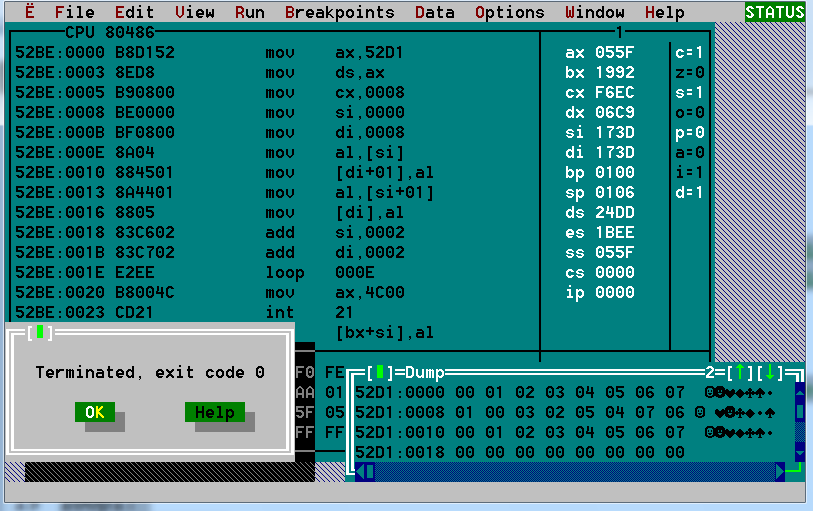


Рис. 4.4 – Программа после отладки

В результате работы программы был проинициализирован массив чисел от 0 до 7 в начале сегмента данных (ds:0000), а также был создан, но не инициализирован пустой массив (ds:0008).

Первый байт исходной строки перемещает во второй байт результирующей строки. Затем второй байт исходной строки перемещает в первый байт результирующей строки. Аналогично с последующими байтами.

**Вывод:**

В ходе лабораторной были изучены правила адресации команд и данных.

Получены следующие навыки:

* Структурированное решение задач
* Использование операторов языка
* Формирование тестовых ситуаций
* Осуществление процесса отладки
* Документирование результатов работы
* Использование Turbo Debugger
* Использование TASM
* Использование TLINK