Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №1

по дисциплине “Архитектура вычислительных систем”

Студент Станиславчук С. М.

Группа АС-21-1

Руководитель Болдырихин О. В.

Ст. преподаватель

Липецк 2023

Цель работы:

Изучение основ устройства и принципов работы компьютера фон-неймановской архитектуры.

Задание кафедры: Вариант 27

Написать на языке ассемблера программу, выполняющую преобразование числа в упакованный двоично-десятичный код.

При помощи отладчика прогнать программу покомандно и после выполнения каждой команды фиксировать состояние аккумулятора, указателя команд, других регистров, задействованных в программе, ячеек памяти данных.

Результаты анализа работы программы оформить в виде таблицы. Последовательность строк в таблице должна соответствовать последовательности выполнения команд в период прогона программы, а не их последовательности в тексте программы. В строке, соответствующей данной команде, содержимое регистров и памяти должно быть таким, каким оно является после ее выполнения.

Проанализировать таблицу, выполнить необходимые сравнения, сделать выводы.

27	Преобразование числа в упакованный двоично-десятичный код	Сегмент данных (по DS) и сегмент команд	Дополнительный сегмент данных (по ES)

Ход работы:

1. Блок-схема алгоритма программы

Составим блок-схему алгоритма преобразования в код с дублированием битов – результат указан на рисунке 1.

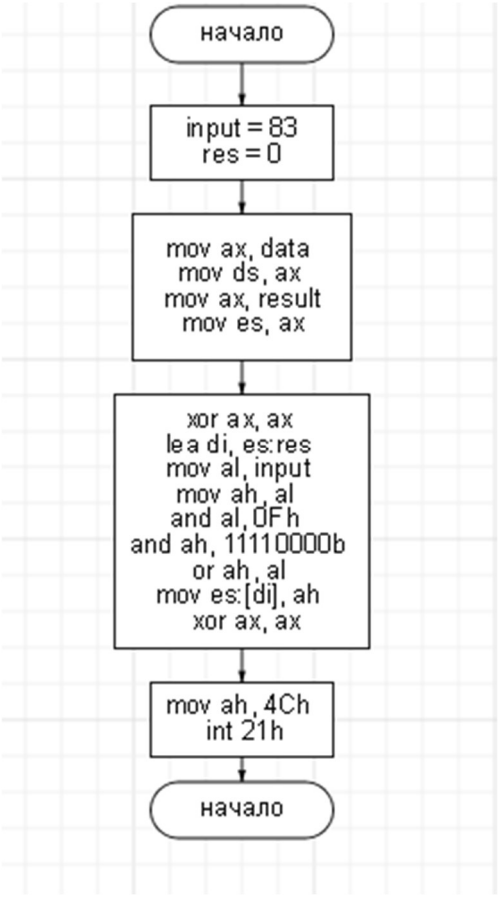

Рисунок 1 – Блок-схема программы.

2. Ручной расчет по алгоритму:

Число 83 в упакованном двоично десятичном коде (packed BCD):

8 -> 1000
3 -> 0011

83 -> 1000 0011

3. Код программы

```
.model small
data segment
    input db 83
data ends
result segment
    res db 0
result ends
code segment
    assume DS:data, CS:code, ES:result
mov ax, data
mov ds, ax
mov ax, result
mov es, ax
main:   ; program
    xor ax, ax
    lea di, es:res
    mov al, input
    mov ah, al
    and al, 0Fh
    and ah, 11110000b
    or ah, al
    mov es:[di], ah
    xor ax, ax

    mov ah, 4Ch
    int 21h
code ends
end
```

4. Листинг программы

```
 1 0000                                 .model small
 2 0000                                 data segment
 3 0000  53                            input db 83
 4 0001                                 data ends
 5
 6 0000                                 result segment
 7 0000  00                            res db 0
 8 0001                                 result ends
 9
10 0000                                 code segment
11                                     assume DS:data, CS:code, ES:result
12
13 0000  B8 0000s                  mov ax, data
14 0003  8E D8                     mov ds, ax
15
16 0005  B8 0000s                  mov ax, result
17 0008  8E C0                           mov es, ax
18
19 000A                                  main:   ; program
20 000A  33 C0                     xor ax, ax
21 000C  BF 0000r                     lea di, es:res
22 000F  A0 0000r                  mov al, input
23 0012  8A E0                         mov ah, al
24 0014  24 0F                         and al, 0Fh
25 0016  80 E4 F0                  and ah, 11110000b
26 0019  0A E0                         or ah, al
27 001B  26: 88 25                      mov es:[di], ah
28 001E  33 C0                         xor ax, ax
29
30 0020  B4 4C                      mov ah, 4Ch
31 0022  CD 21                      int 21h
32
33
34
35 0024                                     code ends
36                                          end
```

5. Таблица состояния системы

Составим таблицу состояний системы после выполнения каждой команды (таблица 1)

Таблица 1 – Состояния системы после выполнения команд программы

Номер команды	Адрес команды	Команда на машинном языке	Регистр команд	Команда на языке ассемблера	Указатель команд	Содержание изменившихся регистров и ячеек памяти
1	0000	B8 0000s	B8	mov ax, data	0003	ax 48B0
2	0003	8E D8	8E	mov ds, ax	0005	ds 48B0
3	0005	B8 0000s	B8	mov ax, result	0008	ax 48B1
4	0008	8E C0	8E	mov es, ax	000A	es 48B1
5	000A	33 C0	33	xor ax, ax	000C	ax 0000
6	000C	BF 0000r	BF	lea di, es:res	000F	di 0000
7	000F	A0 0000r	A0	mov al, input	0012	al 53
8	0012	8A E0	8A	mov ah, al	0014	ah 53
9	0014	24 0F	24	and al, 0Fh	0016	al 03 z 0
10	0016	80 E4 F0	80	and ah, 11110000b	0019	ah 50
11	0019	0A E0	0A	or ah, al	001B	ah 53
12	001B	26 88 25	26	mov es:[di], ah	001E	
13	001E	33 C0	33	xor ax, ax	0020	ax 0000 z 1
14	0020	B4 4C	B4	mov ah, 4Ch	0022	ah 4C
15	0022	CD 21	CD	int 21h		

6. Проверка работы алгоритма на правильных числах

Упакованный двоично-десятичный код (Packed Binary Coded Decimal, PBCD) - это способ представления десятичных чисел в формате, где каждая десятичная цифра представлена в виде 4-битного двоичного числа. В упакованном PBCD каждая десятичная цифра (0-9) кодируется с использованием 4 битов, и эти коды объединяются вместе, чтобы представить десятичное число.

На вход программе подается число 83. Программа разбивает это число на составные цифры (8 и 3) с помощью битовых масок. После разбиения происходит перевод и склеивание битов этих чисел с последующим занесением результата в переменную result, которая находится в сегменте ES. На рисунке 2 видно, что в сегменте es по смещению 0000 (переменная result) лежит число 53. Если мы переведем его в двоичное число, то получим 0101 0011. Первая тетрада есть не что иное, как 5, вторая же – это 3. Переведем этот код в десятичную систему счисления и получим 83. А это значит, что программа отработал верно. Результат программы и состояние регистров CPU можно увидеть на рисунке 2.

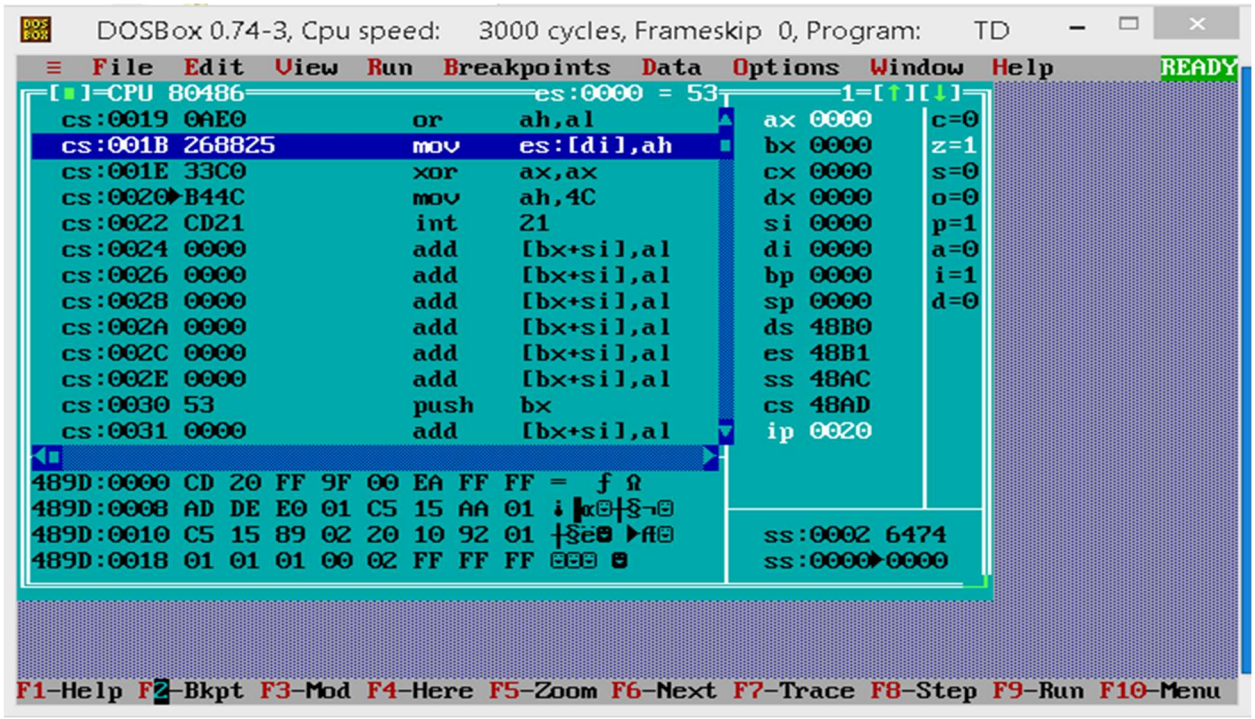

Рисунок 2 – Результат работы кода программы.

7. Вывод

В ходе выполненной работы рассмотрел и проанализировал программу на ассемблере, которая выполняет преобразование двоичного числа в упакованный двоично-десятичный код.