**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.

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Задача, выполняемая программой** | **Расположение исходных данных** | **Расположение результата** |
| 27 | Преобразование числа в упакованный двоично-десятичный код | Дополнительный сегмент данных (по ES) | Сегмент данных (по DS) и сегмент команд |

**Ход работы:**

**1. Блок-схема алгоритма программы**

Составим блок-схему алгоритма преобразования в упакованный

двоично-десятичный код – результат указан на рисунке 1.

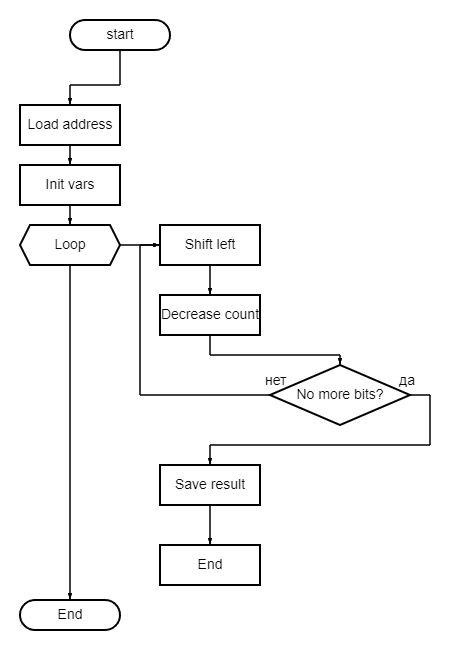


Рисунок 1 – Блок-схема программы

**2. Ручной расчет по алгоритму:**

Разделим число 83 на 10, чтобы получить десяток (8) и остаток (3).

Сместим десяток влево на 4 бита (умножим на 16) и объединим его с остатком.

Результатом будет число в формате PBCD.

**3. Код программы**

model small

data\_in segment

input db 83

data\_in ends

data\_out segment

res1 db 0

data\_out ends

code segment

res2 db 0

assume DS:data\_out, ES:data\_in, CS:code

start:

mov ax, data\_in

mov es, ax

mov ax, data\_out

mov ds, ax

mov al, input

xor ah, ah

mov bl, 10

div bl

mov dl, al

mov al, ah

shl dl, 4

or al, dl

mov res1, al

mov res2, al

mov ah, 4Ch

int 21h

end start

**4. Листинг программы**

1. 0001: mov ax, data\_in ; 0001: B8 B648

2. 0004: mov es, ax ; 0004: 8E C0

3. 0006: mov ax, data\_out ; 0006: B8 B748

4. 0009: mov ds, ax ; 0009: 8E D8

5. 000B: mov al, input ; 000B: 26 A00000

6. 000F: xor ah, ah ; 000F: 32 E4

7. 0011: mov bl, 10 ; 0011: B3 0A

8. 0013: div bl ; 0013: F6 F3

9. 0015: mov dl, al ; 0015: 8A D0

10. 0017: mov al, ah ; 0017: 8A C4

11. 0019: shl dl, 1 ; 0019: D0 E2

12. 001B: shl dl, 1 ; 001B: D0 E2

13. 001D: shl dl, 1 ; 001D: D0 E2

14. 001F: shl dl, 1 ; 001F: D0 E2

15. 0021: or al, dl ; 0021: 0A C2

16. 0023: mov res1, al ; 0023: A2 0000

17. 0026: mov res2, al ; 0026: 2E A20000

18. 002A: mov ah, 4Ch ; 002A: B4 4C

19. 002C: int 21h ; 002C: CD 21

**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 | 0001 | B8 B648 | B8 B648 | mov ax, data\_in | 0004 | ax 48B6 |
| 2 | 0004 | 8E C0 | 8E C0 | mov es, ax | 0006 | es 48B6 |
| 3 | 0006 | B8 B748 | B8 B748 | mov ax, data\_out | 0009 | ax 48B7 |
| 4 | 0009 | 8E D8 | 8E D8 | mov ds, ax | 000B | ds 48B7 es:0000 = 53 |
| 5 | 000B | 26A 00000 | 26A 00000 | mov al, input | 000F | ax 4853 |
| 6 | 000F | 32 E4 | 32 E4 | xor ah, ah | 0011 | ax 0053 |
| 7 | 0011 | B3 0A | B3 0A | mov bl, 10 | 0013 | bx 000A |
| 8 | 0013 | F6 F3 | F6 F3 | div bl | 0015 | ax 0308 |
| 9 | 0015 | 8ADO | 8ADO | mov dl, al | 0017 | dx 0008 |
| 10 | 0017 | 8A C4 | 8A C4 | mov al, ah | 0019 | ax 0303 |
| 11 | 0019 | D0 E2 | D0 E2 | shl dl, 1 | 001B | dx 0010 |
| 12 | 001B | D0 E2 | D0 E2 | shl dl, 1 | 001D | dx 0020 |
| 13 | 001D | D0 E2 | D0 E2 | shl dl, 1 | 001F | dx 0040 |
| 14 | 001F | D0 E2 | D0 E2 | shl dl, 1 | 0021 | dx 0080 |
| 15 | 0021 | 0AC2 | 0AC2 | or al, dl | 0023 | ax 0383 |
| 16 | 0023 | A20000 | A20000 | mov res1, al | 0022 | es:0010 = 83 ds:0000 = 83 |
| 17 | 0026 | 2EA20000 | 2EA20000 | mov res2, al | 002A | cs:0000 = 83 |
| 20 | 002A | B44C | B44C | mov ah, 4Ch | 002C | ax 4C83 |
| 21 | 002E | CD21 | CD21 | int 21h | 0000 | N/A |

**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 DS. На рисунке 2 видно, что в сегменте DS по смещению 0000 (переменная result) лежит число 83h. А это значит, что программа отработал верно. Результат программы и состояние регистров CPU можно увидеть на рисунках 2 и 3 соответственно.

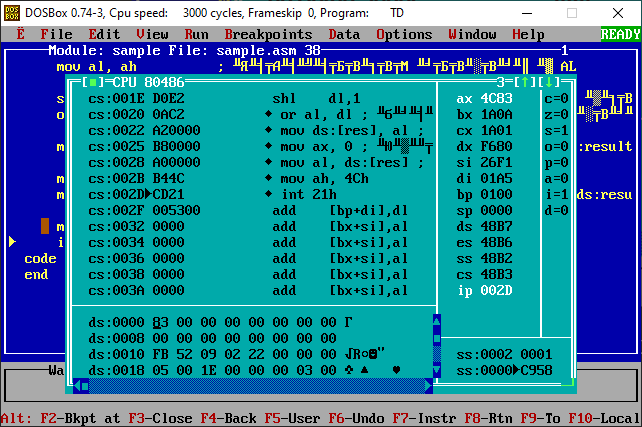


Рисунок 2 – Состояние сегмента DS (result) на момент завершения программы

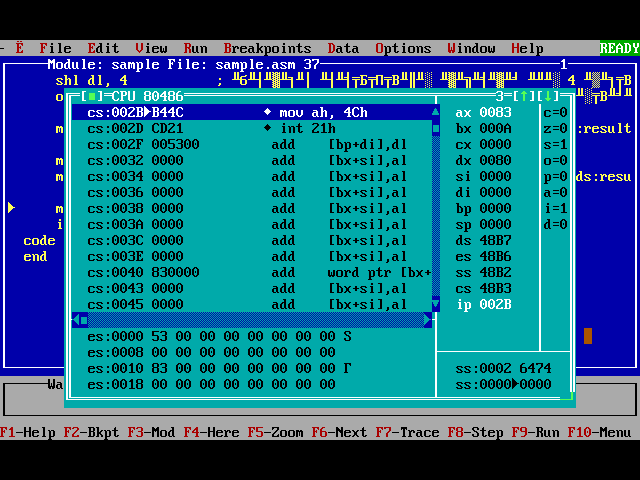


Рисунок 3 – Состояние сегмента ES (data) на момент завершения программы

**7. Вывод**

В ходе выполненной работы рассмотрел и проанализировал программу на

ассемблере, которая выполняет преобразование двоичного числа в

упакованный двоично-десятичный код.

Заметил, что при использовании одинаковых команд на переменные с одинаковыми значениями, но находящихся в разных сегментах, команды на машинном языке отличаются (таблица 1, номера команд 16 и 17. Команда номер 17 имеет префикс “2E”, что данные будут читаться или записываться в сегмент ES).