Министерство образования и науки РФ

Институт информационных технологий и управления в технических системах

Севастопольский государственный университет

Лабораторная работа №2

по дисциплине: «Архитектура ИС»

“Исследование методов реализации алгоритмов обработки

данных на ассемблере 8-разрядного микропроцессора”

Выполнил:

ст.гр. ИСб/22о

Воронин И.Ю.

Проверил:

Дрозин А.Ю.

г. Севастополь

2016 г.

1. Цель работы

Исследовать методы реализации типовых алгоритмов обработки данных на ассемблере процессора КР580ВМ80. Изучение основных ко-манд пересылки данных, передачи управления и арифметических команд ассемблера микропроцессора. Исследование возможностей эмулятора и экранного отладчика KP580 Emulator. Приобретение практических навы-ков составления и отладки программ на языке Ассемблера.

1. Вариант задания
2. Изучить основные команды пересылки данных, логических и арифметических операций, организации ветвлений и циклов (выполняется в процессе домашней подготовки к лабораторной работе).
3. Изучить возможности эмулятора и экранного отладчика КР580 Emulator. Исследовать изменение в основных блоках процессора в ходе выполнения команд различных типов (выполняется в процессе домашней подготовки к лабораторной работе).
4. Составить блок-схему алгоритма функционирования программы в соответствии с заданным вариантом.
5. Реализовать ассемблерную программу в соответствии с заданным вариантом. Модифицировать программу, применяя различные виды команд, выполняющих одинаковые функции.
6. Исследовать длительности выполнения полученных программ в зависимости от используемых команд.

Вариант 2

Осуществить сортировку массива натуральных чисел, используя алгоритм сортировки подсчетом. Размер массива и направление сортировки задается преподавателем.

1. Код и алгоритм программы

Программа реализовывает следующий алгоритм. (рис 3.1)



Рисунок 3.1 –Алгоритм программы.

1. MVI A,0;
2. STA 0000D; начало массива данных исходных данных
3. MVI A,3;
4. STA 0001D;
5. MVI A,0;
6. STA 0002D;
7. MVI A,2;
8. STA 0003D;
9. MVI A,0;
10. STA 0004D;
11. MVI A,3;
12. STA 0005D;
13. MVI A,0;
14. STA 0006D;
15. MVI A,1;
16. STA 0007D;
17. MVI A,0;
18. STA 0010H;
19. MVI A,0;
20. STA 0011H;
21. MVI A,0;
22. STA 0012H;
23. MVI A,0;
24. STA 0013H;
25. MVI A,0;
26. STA 0014H;
27. MVI A,0;
28. STA 0015H;
29. MVI D,01; массив, который будет хранить промежуточны
30. MVI E,00;
31. MVI H,4; счётчик цикла на 4 итерации для внешнего цикла
32. FOR1: INR E;
33. MVI B,0;обозначение первого исходного массива
34. MVI C,1;
35. MVI L,4; счётчик цикла на 4 итерации для внутренего цикла
36. FOR2: LDAX B;
38. CMP E;
39. JZ FOUND;
40. GONEXT: INR C;
41. INR C;переход к следующему элементу
42. DCR L;
43. JNZ FOR2;
44. DCR H;
45. JNZ FOR1;
46. ;промежуточный массив заполнен значениями
47. ;теперь надо заполнить массив числами в порядке возрастания
48. MVI B,0;обозначение первого исходного массива
49. MVI C,1;
50. MVI L,3; счётчик цикла на 3 итерации
51. MVI D,01; масcив, который будет хранить промежуточны
52. MVI E,01;
53. FOR21:
54. LDAX D;
55. MOV H, A;
56. FOR22:
57. MOV A,E;
58. STAX B;
59. INR C;
60. INR C;переход к следующему элементу
61. DCR H;
62. JNZ FOR22;
64. INR E;
65. DCR L;
66. JNZ FOR21;
67. RET;
68. FOUND:
69. LDAX D;
70. INR A;
71. STAX D;
72. JMP GONEXT;

4. Выполнение программы

В ходе лабораторной работы была написана программа, которая реализует сортировку подсчётов массива, размерность которого в данном случае равно 4 и имеет в себе всего лишь три различных элемента.

Изначально программа, обнуляет 4 ячейки памяти, в которых будет храниться массив. Далее программа проходит по массиву, записывая в вспомогательный массив количество чисел конкретного номинала.

Далее мы проходим по вспомогательному массиву, записываю N раз число конкретного номинала в память, где ранился исходный массив.

Имеется массив: 3, 2, 3, 1. В памяти он будет храниться следующим образом. (рис. 4. 1)

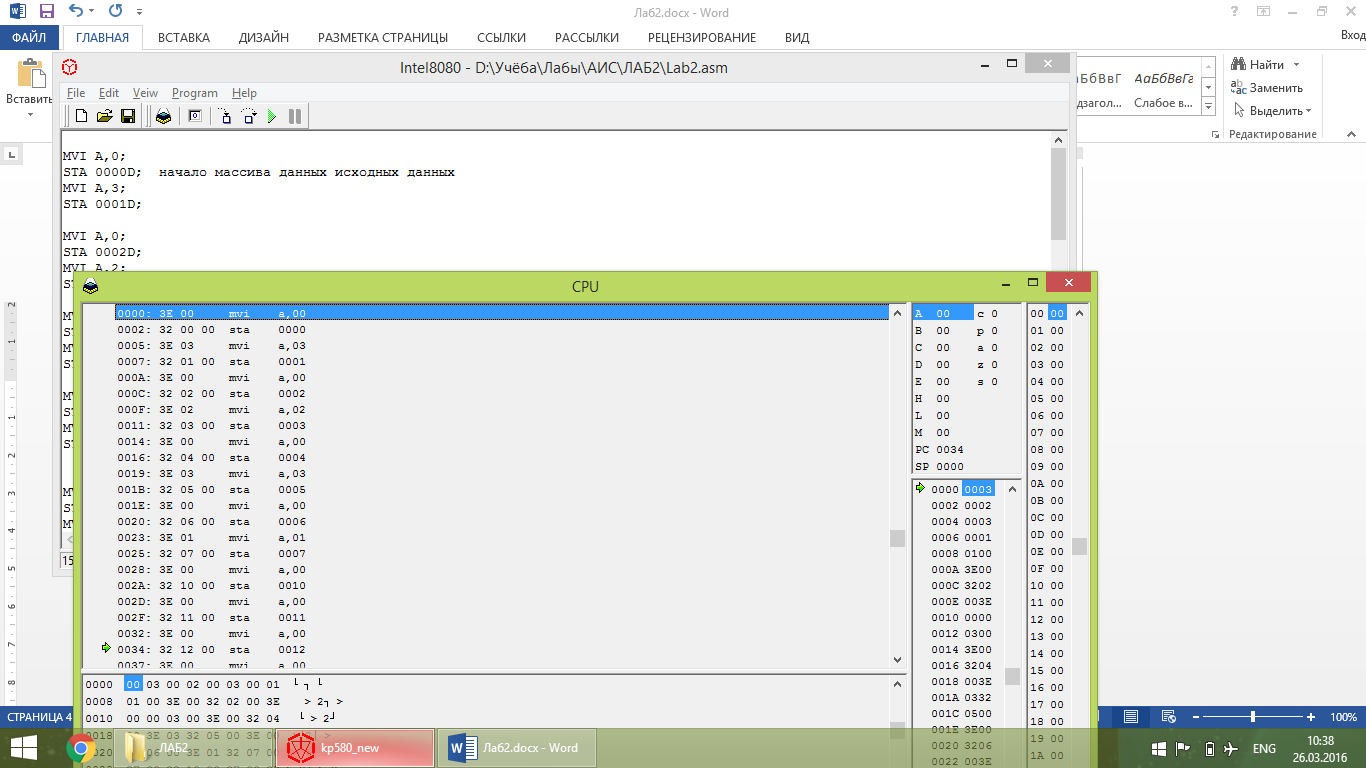


Рисунок 4.1 – Хранение массива.

Далее будет образов следующий вспомогательный массив, где в ячейке памяти 0101 будет хранится количество единиц, в 0102- количество двоек, в 0103 – количества троек. (рис.4.2)

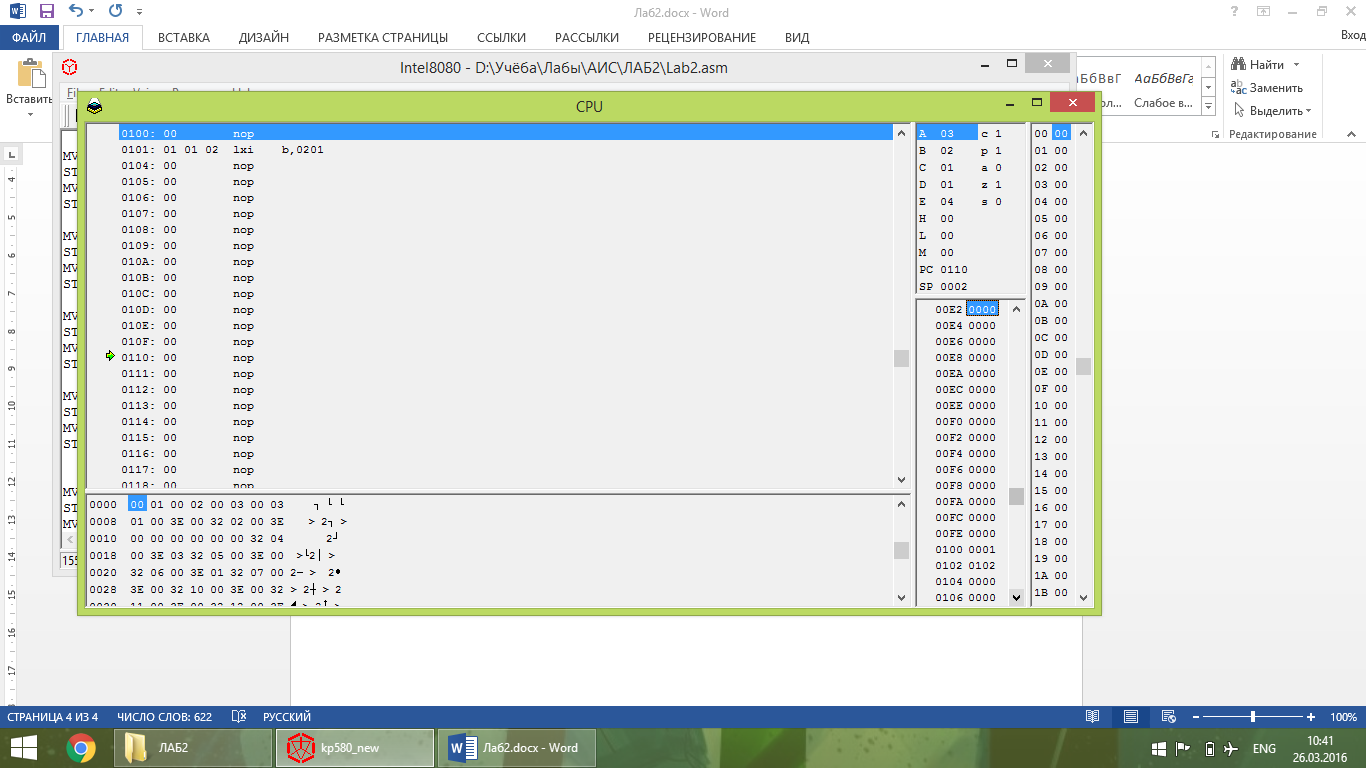


Рисунок 4.2 – Хранение вспомогательного массива.

В конце работы программы будет хранится в памяти следующий массив. (рис 4.3)

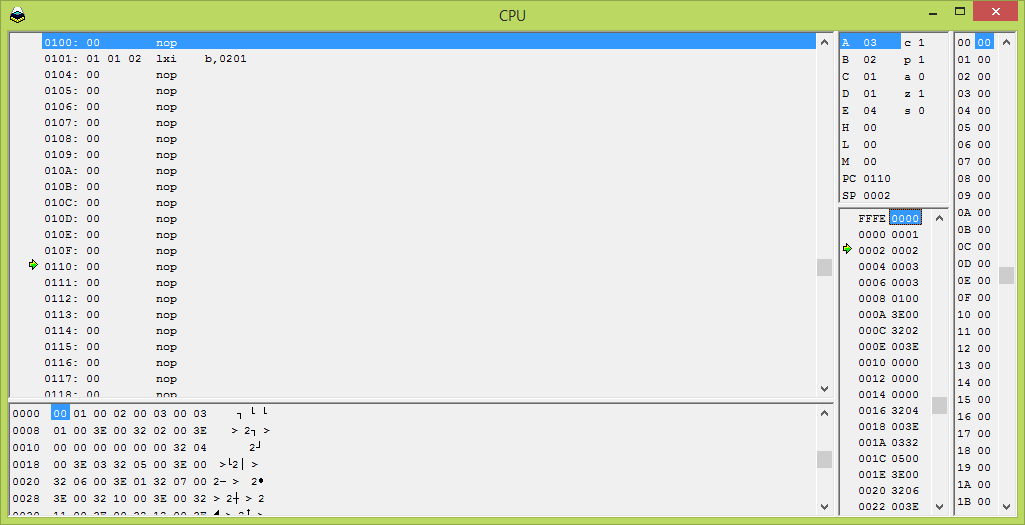


Рисунок 4.3 – Отсортированный массив.

5. Анализ программы

Вычислим количество тактов и время их реализации для данного участка кода:

FOR21: три итерации цикла

LDAX D; (7 тактов)

MOV H, A; (5 тактов)

FOR22: в сумме цикл сработает 4 раза

MOV A,E; (5 тактов)

STAX B; (7 тактов)

INR C; (5 тактов)

INR C; (5 тактов)

DCR H; (5 тактов)

JNZ FOR22; (10 тактов)

INR E; (5 тактов)

DCR L; (5 тактов)

JNZ FOR21; (10 тактов)

В данном случае у нас будет конструкция цикл в цикле. Получим следующее число тактов:

3 \* (5 + 4 \* (5+7+5+5+5+10) + 5 + 5 + 10) = 3 \* (29 + 37 ) = 198 (тактов)

Так как частота процессора равняется 2МГц, то один тактов будет длиться 0,5 мкс. Тогда общее время выполнения участка кода равно:

198 \* 0,5 = 99 мкс

ВЫВОДЫ

Были изучены основные команды пересылки данных, логических и арифметических операций, организации ветвлений и циклов. Реализация программы была проведена в эмуляторе процессора КР580. Было исследовано количеством тактов команд и время их выполнения.