Санкт-Петербургский Политехнический Университет Петра Великого
Институт компьютерных наук и технологий
Высшая школа информационных систем и суперкомпьютерных технологий

Низкоуровневое программирование

Отчёт по лабораторной работе №2
Программирование EDSAC

Работу выполнил:
Унтила А.А.
Группа:
3530901/90004
Преподаватель:
Алексюк А.О.

Санкт-Петербург
2021

Содержание

1. Цель работы

1. Разработать программу для EDSAC, реализующую Реверс массива чисел, и предполагающую загрузчик Initial Orders 1. Массив (массивы) данных и другие параметры (преобразуемое число, длина массива, параметр статистики и пр.) располагаются в памяти по фиксированным адресам.

2. Выделить функциональность в замкнутую (closed) подпрограмму, разработать вызывающую ее тестовую программу. Использовать возможности загрузчика Initial Orders 2. Адрес обрабатываемого массива данных и другие параметры передавать через ячейки памяти с фиксированными адресами.

2. Программа работы

- Теоретическая информация.
- Постановка задачи.
- Симулятор EDSAC.
- Программа с использованием загрузчика Initial Orders 1.
- Программа с использованием загрузчика Initial Orders 2.

3. Теоретическая информация

EDSAC (англ. ***E**lectronic **D**elay **S**torage **A**utomatic **C**alculator*) — электронная вычислительная машина, созданная в 1949 году в Кембриджском университете (Великобритания). Первый в мире действующий и практически используемый компьютер с хранимой в памяти программой.

Компьютер состоял из примерно 3000 электронных ламп. Основная память компьютера состояла из 32 ртутных ультразвуковых линий задержки (РУЛЗ), каждая из которых хранила 32 слова по 17 бит, включая бит знака — всего это даёт 1024 ячеек памяти. Для ввода данных использовались перфоленты.

Для EDSAC было сделано несколько десятков подпрограмм: для вычислений с плавающей запятой, работы с комплексными числами, векторами и матрицами, вычисления логарифмов, тригонометрических функций и арифметических корней произвольной степени и др. В сентябре 1949 была добавлена возможность записи программ не на машинных кодах, а на символьном языке, который стал первым языком ассемблера. В 1951 с помощью этого компьютера было открыто 79-значное простое число — самое большое из известных в то время. В 1952 для компьютера EDSAC с подключенным устройством вывода на электронно-лучевых трубках была разработана программа «Крестики-нолики», став первой компьютерной игрой.

4. Ход выполнения работы

4.1. Постановка задачи

Реверс массива чисел - запись разрядов в обратном порядке. Для отображения корректности работы программы используется массив с 11-ю элементами 0, 1, ..., 10.

Алгоритм:

- Чтение текущего элемента массива и запись в аккумулятор.
- Запись из аккумулятора в рабочую ячейку.
- Чтение зеркального элемента массива и запись в ячейку текущего элемента массива.
- Чтение числа из рабочей ячейки и запись в ячейку зеракльного элемента массива.
- Повторение всех предыдущих шагов пока не останется необработанных элеметов.

В качестве рабочей ячейки выбрана ячейка 0. Для фиксирования числа необработанных элементов используется ячейка 1, куда изначально записывается длина массива (константа), на каждом шаге из которой вычитается число обработанных элементов (2 - константа). Для записи и чтения элементов массива достаточно знать номер ячейки с первым (или нулевым) элементом массива. Для чтения и записи в зеркальный элемент массива используется счётчик необработанных элементов в ячейке 1. Для изменения индексов (адресов) элементов массива используется константа 2, которая из-за сдвига изменяет адреса на единицу.

4.2. Выбор симулятора EDSAC

В данной лабораторной работе используется симулятор EDSAC, расположенный по ссылке https://www.dcs.warwick.ac.uk/ẽdsac/. В симуляторе возможно использование загрузчиков IO1 и IO2, а также отображение оригинального лампового монитора, на котором можно видеть содержимое ячеек.

4.3. Программа с использованием загрузчика Initial Orders 1

Листинг программы с поясняющими комментариями:

```
31  T 95 S
32  Z 0 S [ останов для отладки ]
33  T 0 S [ запись аккумулятора в рабочую ячейку 0, обнуление аккумулятора ]
34  A 82 [<len>] S [ загрузка в аккумулятор длины массива ]
35  U 1 S [ запись длины массива в ячейку 1 ]
36  A 83 [<addr>] S [ загрузка в аккумулятор адреса 0-го элемента массива ]
37  L 0 L [ сдвиг аккумулятора на 1 разряд влево ]
38  S 81 [<c2>] S [ вычитаем 2 для получения адреса последнего элемента ]
39  A 61 [<r2>] S [ прибавляем код инструкции с нулевым полем адреса ]
40  T 61 [<r2>] S [ запись сформированной инструкции, обнуление аккумулятора ]
41  A 82 [<len>] S [ загрузка в аккумулятор длины массива ]
42  A 83 [<addr>] S [ загрузка в аккумулятор адреса 0-го элемента массива ]
43  L 0 L [ сдвиг аккумулятора на 1 разряд влево ]
44  S 81 [<c2>] S [ вычитаем 2 для получения адреса последнего элемента ]
45  A 64 [<w2>] S [ прибавляем код инструкции с нулевым полем адреса ]
```

```
46    T 64 [<w2>] S [ запись сформированной инструкции, обнуление аккумулятора ]
47    A 83 [<addr>] S [ загрузка в аккумулятор адреса 0-го элемента массива ]
48    L 0 L [ сдвиг аккумулятора на 1 разряд влево ]
49    A 59 [<r1>] S [ прибавляем код инструкции с нулевым полем адреса ]
50    T 59 [<r1>] S [ запись сформированной инструкции, обнуление аккумулятора ]
51    A 83 [<addr>] S [ загрузка в аккумулятор адреса 0-го элемента массива ]
52    L 0 L [ сдвиг аккумулятора на 1 разряд влево ]
53    A 62 [<w1>] S [ прибавляем код инструкции с нулевым полем адреса ]
54    T 62 [<w1>] S [ запись сформированной инструкции, обнуление аккумулятора ]
55    A 1 S [ загружаем счетчик необработанных элементов массива ]
56    S 81 [<c2>] S [ уменьшаем на 2 ]
57    G 80 [<exit>] S [ если результат меньше 0, завершаем работу ]
58    T 1 S [ обновляем значение счетчика и обнуляем аккумулятор ]
59    [r1:] A 0 S [ загрузка в аккумулятор значения из ячейки N ]
60    T 0 S [ запись в рабочую ячейку 0, обнуление аккумулятора ]
61    [r2:] A 0 S [ загрузка в аккумулятор значения из зеркальной ячейки ]
62    [w1:] T 0 S [ запись в ячейку с адресом N, обнуление аккумулятора ]
63    A 0 S [ загрузка в аккумулятор значения из ячейки 0 ]
64    [w2:] T 0 S [ запись в зеркальную ячейку, обнуление аккумулятора ]
65    A 1 [<1>] S [ загрузка в аккумулятор значения из ячейки 1 - счётчика ]
66    L 0 L [ сдвиг аккумулятора на 1 разряд влево ]
67    A 59 [<r1>] S [ прибавляем код инструкции с полем адреса элемента N ]
68    T 61 [<r2>] S [ запись сформированной инструкции, обнуление аккумулятора ]
69    A 1 [<1>] S [ загрузка в аккумулятор значения из ячейки 1 - счётчика ]
70    L 0 L [ сдвиг аккумулятора на 1 разряд влево ]
71    A 62 [<w1>] S [ прибавляем код инструкции с полем адреса элемента N ]
72    T 64 [<w2>] S [ запись сформированной инструкции, обнуление аккумулятора ]
73    A 81 [<c2>] S [ загрузка в аккумулятор константы 2 - вместо 1 и сдвига ]
74    A 59 [<r1>] S [ прибавляем код инструкции с полем адреса элемента N ]
75    T 59 [<r1>] S [ запись сформированной инструкции, обнуление аккумулятора ]
76    A 81 [<c2>] S [ загрузка в аккумулятор константы 2 - вместо 1 и сдвига ]
77    A 62 [<w1>] S [ прибавляем код инструкции с полем адреса элемента N ]
78    T 62 [<w1>] S [ запись сформированной инструкции, обнуление аккумулятора ]
79    E 55 [<loop>] S [ повторяем все операции ]
80    [exit:] Z 0 S [ останов ]
81    [c2:] P 1 S [ константа 2 ]
82    [len:] P 5 L [ длина массива = 11 ]
83    [addr:] P 42 S [ 84 - адрес 0-го элемента массива ]
84    [array:] P 0 S [ 0 ]
85    P 0 L [ 1 ]
86    P 1 S [ 2 ]
87    P 1 L [ 3 ]
88    P 2 S [ 4 ]
89    P 2 L [ 5 ]
90    P 3 S [ 6 ]
91    P 3 L [ 7 ]
92    P 4 S [ 8 ]
93    P 4 L [ 9 ]
94    P 5 S [ 10 ]
95
```

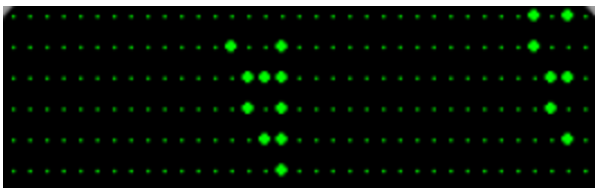

Рис. 4.3.1. Массив до реверса.

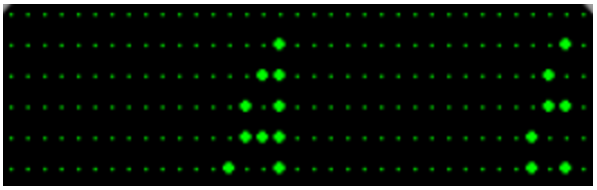

Рис. 4.3.2. Массив после реверса.

Из приведённых выше рисунков видно, что программа работает корректно.

4.4. Программа с использованием загрузчика Initial Orders 2

Листинг программы с поясняющими комментариями:

```
1    T 56 K [ директива IO2, установка адреса загрузки ]
...
56   G    K [ директива IO2, фиксация начального адреса подпрограммы ]
57   [ 0:] A 3 F [ пролог: формирование кода инструкции возврата в Acc ]
58   [ 1:] T 46 [<ret>] @ [ пролог: запись инструкции возврата ]
59   [ 2:] A 1 [<len>] F [ загрузка в аккумулятор длины массива ]
60   [ 3:] L 0 D [ сдвиг аккумулятора на 1 разряд влево ]
61   [ 4:] S 47 [<c2>] @ [ вычитаем 2 для получения адреса последнего элемента ]
62   [ 5:] A 0 [<addr>] F [ загрузка в аккумулятор адреса 0-го элемента массива ]
63   [ 6:] A 48 [<r_init>] @ [ прибавляем код инструкции с нулевым полем адреса ]
64   [ 7:] T 26 [<r2>] @ [ запись сформированной инструкции, обнуление аккум-а ]
65   [ 8:] A 1 [<len>] F [ загрузка в аккумулятор длины массива ]
66   [ 9:] L 0 D [ сдвиг аккумулятора на 1 разряд влево ]
67   [ 10:] S 47 [<c2>] @ [ вычитаем 2 для получения адреса последнего элемента ]
68   [ 11:] A 0 [<addr>] F [ загрузка в аккумулятор адреса 0-го элемента массива ]
69   [ 12:] A 49 [<w_init>] @ [ прибавляем код инструкции с нулевым полем адреса ]
70   [ 13:] T 29 [<w2>] @ [ запись сформированной инструкции, обнуление аккум-а ]
71   [ 14:] A 0 [<addr>] F [ загрузка в аккумулятор адреса 0-го элемента массива ]
72   [ 15:] A 48 [<r_init>] @ [ прибавляем код инструкции с нулевым полем адреса ]
73   [ 16:] T 24 [<r1>] @ [ запись сформированной инструкции, обнуление аккум-а ]
74   [ 17:] A 0 [<addr>] F [ загрузка в аккумулятор адреса 0-го элемента массива ]
75   [ 18:] A 49 [<w_init>] @ [ прибавляем код инструкции с нулевым полем адреса ]
76   [ 19:] T 27 [<w1>] @ [ запись сформированной инструкции, обнуление аккум-а ]
77   [ 20:] [loop:] A 1 F [ загружаем счетчик необработанных элементов массива ]
78   [ 21:] S 47 [<c2>] @ [ уменьшаем на 2 ]
79   [ 22:] G 45 [<exit>] @ [ если результат меньше 0, завершаем работу ]
80   [ 23:] T 1 F [ обновляем значение счетчика и обнуляем аккумулятор ]
81   [ 24:] [r1:] A 0 F [ загрузка в аккумулятор значения из ячейки N ]
```

```
82    [ 25:] T 0 F [ запись в рабочую ячейку 0, обнуление аккумулятора ]
83    [ 26:] [r2:] A 0 F [ загрузка в аккумулятор значения из зеркальной ячейки ]
84    [ 27:] [w1:] T 0 F [ запись в ячейку с адресом N, обнуление аккумулятора ]
85    [ 28:] A 0 F [ загрузка в аккумулятор значения из ячейки 0 ]
86    [ 29:] [w2:] T 0 F [ запись в зеркальную ячейку, обнуление аккумулятора ]
87    [ 30:] A 1 [<1>] F [ загрузка в аккумулятор значения из ячейки 1 - счётчика ]
88    [ 31:] L 0 D [ сдвиг аккумулятора на 1 разряд влево ]
89    [ 32:] A 24 [<r1>] @ [ прибавляем код инструкции с полем адреса элемента N]
90    [ 33:] T 26 [<r2>] @ [ запись сформированной инструкции, обнуление аккум-а ]
91    [ 34:] A 1 [<1>] F [ загрузка в аккумулятор значения из ячейки 1 - счётчика ]
92    [ 35:] L 0 D [ сдвиг аккумулятора на 1 разряд влево ]
93    [ 36:] A 27 [<w1>] @ [ прибавляем код инструкции с полем адреса элемента N]
94    [ 37:] T 29 [<w2>] @ [ запись сформированной инструкции, обнуление аккум-а ]
95    [ 38:] A 47 [<c2>] @ [ загрузка в аккумулятор константы 2 - вместо 1 и сдвига ]
96    [ 39:] A 24 [<r1>] @ [ прибавляем код инструкции с полем адреса элемента N]
97    [ 40:] T 24 [<r1>] @ [ запись сформированной инструкции, обнуление аккум-а ]
98    [ 41:] A 47 [<c2>] @ [ загрузка в аккумулятор константы 2 - вместо 1 и сдвига ]
99    [ 42:] A 27 [<w1>] @ [ прибавляем код инструкции с полем адреса элемента N]
100   [ 43:] T 27 [<w1>] @ [ запись сформированной инструкции, обнуление аккум-а ]
101   [ 44:] E 20 [<loop>] @ [ повторяем все операции ]
102   [ 45:] [exit:] T 0 F [ останов ]
103   [ 46:] [ret:] E 0 F [ эпилог: инструкция возврата из подпрограммы ]
104   [ 47:] [c2:] P 1 F [ константа 2 ]
105   [ 48:] [r_init:] A 0 F [ основа для формирования инструкции с меткой r ]
106   [ 49:] [w_init:] T 0 F [ основа для формирования инструкции с меткой w ]
107   G    K [ директива IO2, фиксация начального адреса программы ]
108   [ 0:] Z 0 F [ останов для пошаговой отладки ]
109   [ 1:] A 8 [<addr>] @ [ адрес массива ]
110   [ 2:] T 0 F [ запись адреса массива в ячейку 0, обнуление аккумулятора ]
111   [ 3:] A 9 [<len>] @ [ длина массива ]
112   [ 4:] T 1 F [ запись длины массива в ячейку 1, обнуление аккумулятора ]
113   [ 5:] A 5 @ [вызов ]
114   [ 6:] G 56 [<sub>] F [/ подпрограммы ]
115   [ 7:] Z 0 F [ останов ]
116   [ 8:] P 10 [<array>] @ [ адрес массива ]
117   [ 9:] P 5 D [ длина массива = 11) ]
118   [10:] P 0 F [ 0 ]
119   [11:] P 0 D [ 1 ]
120   [12:] P 1 F [ 2 ]
121   [13:] P 1 D [ 3 ]
122   [14:] P 2 F [ 4 ]
123   [15:] P 2 D [ 5 ]
124   [16:] P 3 F [ 6 ]
125   [17:] P 3 D [ 7 ]
126   [18:] P 4 F [ 8 ]
127   [19:] P 4 D [ 9 ]
128   [20:] P 5 F [ 10 ]
129   EZ PF [ директива IO2, переход к исполнению ]
```

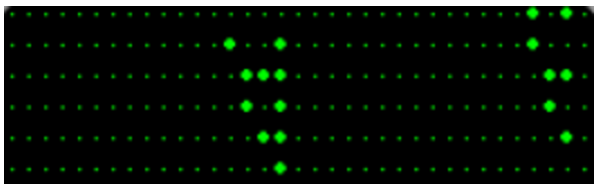

Рис. 4.4.1. Массив до реверса.

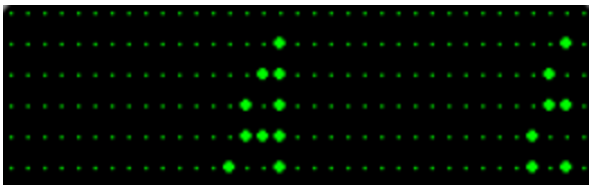

Рис. 4.4.2. Массив после реверса.

Из приведённых выше рисунков видно, что программа работает корректно.

5. Вывод

В ходе лабораторной работы написано две программы для машины EDSAC, осуществляющие реверс массива чисел (запись разрядов в обратном порядке), описан общий алгоритм реализации данной функциональности. Первая программа написана для загрузчика Initial Orders 1 и представляет собой непосредственную реализацию заданной функциональности. Вторая программа с использованием загрузчика Initial Orders 2 расширяет демонстрацию возможностей машины EDSAC, в ней уже содержится программа и вызываемая в этой программе подпрограмма. Работа обоих программ протестирована в симуляторе, приведены соответствующие результаты, программы работают корректно.