

Санкт-Петербургский политехнический университет Петра Великого  
Институт компьютерных наук и технологий  
Высшая школа интеллектуальных систем и суперкомпьютерных технологий

**Отчет по лабораторной работе №2**

Тема: Машина EDSAC

Вариант 10

Выполнил студент гр. 3530901/90002 \_\_\_\_\_ П. В. Рубинова  
(подпись)

Руководитель \_\_\_\_\_ Д. С. Степанов  
(подпись)

“ \_\_\_\_ ” \_\_\_\_\_ 2021 г.

Санкт-Петербург

2021

### **Постановка задачи:**

1. Разработать программу для EDSAC, реализующую определенную вариантностью функциональность, 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 определенную вариантностью 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.

### **Формулировка задачи:**

Загрузка с перфоленты последовательности («коротких») чисел. Числа кодируются обычным для EDSAC образом (в виде псевдоинструкций).

## **Программа, предполагающая загрузчик Initial Orders 1**

В ячейках [68] – [73] располагаются элементы массива с входными значениями.

Запись производится в ячейки [4] – [9] включительно.

Основной алгоритм работы программы содержится в цикле. Так как с помощью симулятора EDSAC невозможно пользоваться загрузкой с перфоленты, в программе используется массив и значения, заранее записанные в память машины (ячейки [68] – [73]).

### **Алгоритм работы:**

1. Программа берет числа с перфоленты.
2. Записывает их в память с помощью цикла, поочередно инкрементируя индекс ячейки, в которую производится запись с помощью заранее определенной в ячейке [62] «константы» - 1.
3. Начинается запись с ячейки, которая подается в исходных данных.

### **Текст программы:**

T 74 S [74- конец программы]

[32]Z 0 S [Точка останова для отладки]

[33]T 0 S [Очистка аккумулятора]

[34]A 65[длина] S [Загружаем в аккумулятор длину массива]

[35]T 0 S [Кладем ее в ячейку 0]

[36]A 67[адрес] S [Загружаем в аккумулятор адрес нулевого элемента массива]

[37]T 1 S [Кладем его в первую ячейку]

[38]A 1 S [Прибавляем в аккумулятор адрес нулевого элемента массива]

[39]L 0 L [Сдвигаем его на 1 разряд влево, для того, чтобы загрузить в него инструкцию]

[40]A 63[A 0 S] S [Добавляем в аккумулятор код команды A 0 S. получится команда A (адрес нулевого элемента массива) S]

[41]T 50[взятие элемента с перфоленты] S [Кладем полученную команду в адрес, где должна находиться команда чтения элемента с перфоленты]

[42]A 66[адрес записи в память] S [Прибавляем в аккумулятор адрес записи в память]

[43]L 0 L [Сдвигаем его на 1 разряд влево]

[44]A 64[T 0 S] S [Добавляем в аккумулятор код команды T 0 S. Таким образом получится команда T (адрес записи в память) S]

[45]T 51[помещение элемента в память] S [Кладем полученную команду в адрес, где должна находиться команда записи элемента в память]

[цикл:]

[46]A 0 S [Добавляем в аккумулятор значение ячейки 0]

[47]S 62[1] S [Вычитаем единицу]

[48]G 61[выход из цикла] S [Если в аккумуляторе отрицательное значение, массив прочитан -> завершить цикл]

[49]T 0 S [Если значение не отрицательное, то записываем его в 0 ячейку]

[50]A 0 S[взятие элемента с перфоленты] [Записываем в аккумулятор элемент с перфоленты]

[51]T 0 S[Кладем элемент в ячейку памяти]

[52]A 62[1] S [Прибавляем единицу в аккумулятор]

[53]L 0 L [Сдвигаем ее на 1 разряд влево]

[54]A 50[взятие элемента с перфоленты] S [Добавляем в аккумулятор команду взятия элемента с перфоленты]

[55]T 50[взятие элемента с перфоленты] S [Кладем команду обратно]

[56]A 62[1] S [Прибавляем единицу в аккумулятор]

[57]L 0 L [Сдвигаем ее на 1 разряд влево]

[58]A 51[помещение элемента в память] S [Добавляем в аккумулятор команду записи элемента в память]

[59]T 51[помещение элемента в память] S [Кладем команду обратно]

[60]E 46[цикл] S [Переход в начало цикла]

[61]Z 0 S

[62]P 0 L[const (2 \* 0 + 1=1)]

[63]A 0 S

[64]T 0 S

[65]P 3 S [длина (=6)]

[66]P 2 S [адрес записи в память (=4)]

[67]P 34 S [адрес (=68)]

[массив:]

[68]P 5 S [10]

[69]P 5 L [11]

[70]P 6 S [12]

[71]P 6 L [13]

[72]P 7 S [14]

[73]P 7 L [15]

### **Руководство:**

В ячейках [32] – [45] идет подготовка к работе цикла:

- Очистка аккумулятора перед работой (ячейка [33])
- Перенос длины исходного массива в нулевую ячейку (ячейки [34] – [35])
- Перенос адреса нулевого элемента в первую ячейку (ячейки [36] – [37])
- В ячейках [38] – [45] происходит перезапись заранее подготовленных команд A 0 S, T 0 S, которые находятся в ячейках [50] [51] в виде «А (адрес нулевого элемента массива) S» и «Т (адрес записи в память) S».

В цикле (ячейки [46] – [60]) происходит следующее:

1. От длины массива вычитается единица
2. Это значение сравнивается с нулем (если значение отрицательное, то цикл прерывается, поскольку весь массив уже перезаписан, если значение не

отрицательное, то мы записываем уменьшенную длину массива обратно в ячейку)

3. Далее в ячейках [50] и [51], значения которых мы изменили до цикла, происходит запись первого элемента массива в нужную нам ячейку
4. После в ячейках [52] – [59] мы снова меняем значения ячеек [50] и [51], прибавляя к ним единицу по алгоритму, описанному в первом абзаце
5. В конце цикла в ячейке [60] находится команда «E 46 S», которая возвращает нас в начало цикла.

В конце инструкции содержатся заранее заданные команды, длина массива, адрес записи в память и массив, имитирующий перфоленту.

## **Программа, предполагающая загрузчик Initial Orders 2**

Функциональность программы, предполагающей загрузчик Initial Orders 1 была выделена в подпрограмму.

### **Алгоритм работы:**

1. Программа берет числа с перфоленты.
2. Записывает их в память с помощью цикла, поочередно инкрементируя индекс ячейки, в которую производится запись с помощью заранее определенной в ячейке [91] «константы» - 1.
3. Начинается запись с ячейки, которая подается в исходных данных.

### **Текст программы:**

T 64 K [Директива - установка адреса загрузки]

[Подпрограмма:]

GK[Директива - фиксация начального адреса подпрограммы]

[0]A 3 F [Формирование инструкции возврата]

[1]T 26 @ [Запись сформированной инструкции]

[2]A 1 F [Прибавляем в аккумулятор адрес нулевого элемента массива]

[3]L 0 D [Сдвигаем его на 1 разряд влево]

[4]A 28[A 0 F] @ [Добавляем в аккумулятор код команды A 0 S. Таким образом получится команда A (адрес нулевого элемента массива) S]



[5]T 14[Взятие элемента с перфоленты] @ [Кладем полученную команду в адрес, где должна находиться команда чтения элемента с перфоленты]

[6]A 2 F [Прибавляем в аккумулятор адрес записи в память]

[7]L 0 D [Сдвигаем его на 1 разряд влево]

[8]A 29[T 0 F] @ [Добавляем в аккумулятор код команды T 0 S. Таким образом получится команда T (адрес записи в память) S]

[9]T 15[Помещение элемента в память] @ [Кладем полученную команду в адрес, где должна находиться команда записи элемента в память]

[Цикл:]

[10]A 0 F [Добавляем в аккумулятор значение ячейки 0, в которой записана длина массива]

[11]S 27[1] @ [Вычитаем единицу]

[12]G 25[Выход из цикла] @ [Если в аккумуляторе отрицательное значение, значит массив прочитан и можно завершить цикл]

[13]T 0 F [Если значение не отрицательное, то записываем его в 0 ячейку]

[14]A 0 F[Взятие элемента с перфоленты] [Записываем в аккумулятор элемент с перфоленты]

[15]T 0 F[Помещение элемента в память] [Кладем его в ячейку памяти]

[16]A 27[1] @ [Прибавляем единицу в аккумулятор]

[17]L 0 D [Сдвигаем ее на 1 разряд влево]

[18]A 14[Взятие элемента с перфоленты] @ [Добавляем в аккумулятор команду взятия элемента с перфоленты]

[19]T 14[Взятие элемента с перфоленты] @ [Кладем команду обратно]

[20]A 27[1] @ [Прибавляем единицу в аккумулятор]

[21]L 0 D [Сдвигаем эту единицу на 1 разряд влево]

[22]A 15[Помещение элемента в память] @ [Добавляем в аккумулятор команду записи элемента в память]

[23]T 15[Помещение элемента в память] @ [Кладем команду обратно]

[24]E 10[Цикл] @ [Переход в начало цикла]

[25]T 0 F [Очистка аккумулятора]

[26]E 0 F [Инструкция возврата]

[27]P 0 D[const=1]

[28]A 0 F

[29]T 0 F

GK[Директива - фиксация начального адреса подпрограммы]

[0]Z 0 F [Точка останова для отладки]

[1]A 9 @ [Запись в аккумулятор длины массива]

[2]T 0 F [Запись длины массива (=6) в 0 ячейку]

[3]A 10 @ [Запись в аккумулятор адреса массива]

[4]T 1 F [Запись адреса массива (=106) в первую ячейку]

[5]A 11 @ [Запись в аккумулятор адреса записи]

[6]T 2 F [Запись адреса записи (=4) во вторую ячейку]

[7]A 7 @ [Добавление в аккумулятор кода, необходимого для формирования инструкции возврата]

[8]G 64 F [Переход в подпрограмму]

[9]P 3 F [Длина = 6]

[10]P 53 F [Адрес массива = 106]

[11]P 2 F [Адрес записи = 4]

[Массив:]

[106]P 5 F [10]

[107]P 5 D [11]

[108]P 6 F [12]

[109]P 6 D [13]

[110]P 7 F [14]

[111]P 7 D [15]

EZ PF [Будет осуществлен переход на первую инструкцию последней подпрограммы, при этом аккумулятор будет обнулен]

**Руководство:**

Никаких существенных изменений нет. Основной рабочий цикл остался тем же, принцип работы тот же. Главное отличие в относительной адресации.

(Руководство было написано с использованием абсолютной адресации для удобства проверки).

В ячейках [94] – [101] идет подготовка к работе цикла:

- Перенос длины исходного массива в нулевую ячейку (ячейки [95] – [96])
- Перенос адреса нулевого элемента в первую ячейку (ячейки [97] – [98])
- Перенос адреса записи во вторую ячейку (ячейки [99] – [100])
- Добавление в аккумулятор кода, необходимого для формирования инструкции возврата (ячейка [101])

Далее (ячейка [102]) происходит переход в подпрограмму, в которой также происходит подготовка к работе цикла в ячейках [64] – [73]:

- В ячейках [64] – [65] происходит формирование и запись инструкции возврата.
- В ячейках [66] – [73] происходит перезапись заранее подготовленных команд A 0 F, T 0 F, которые находятся в ячейках [78] и [79] в виде «А (адрес нулевого элемента массива) F» и «Т (адрес записи в память) F».

В цикле (ячейки [74] – [88]) происходит следующее:

1. От длины массива вычитается единица
2. Это значение сравнивается с нулем (если значение отрицательное, то цикл прерывается, поскольку весь массив уже перезаписан, если значение не отрицательное, то мы записываем уменьшенную длину массива обратно в ячейку)

3. Далее в ячейках [78] и [79], значения которых мы изменили до цикла, происходит запись первого элемента массива в нужную нам ячейку
4. После в ячейках [80] – [87] мы снова меняем значения ячеек [78] и [79], прибавляя к ним единицу по алгоритму, описанному в первом абзаце
5. В конце цикла в ячейке [88] находится команда «E 74 F», которая возвращает нас в начало цикла.

**Вывод:**

В ходе данной лабораторной работы я познакомилась с принципом работы EDSAC и общими правилами реализации алгоритмов на ней на примере загрузки с перфоленты последовательности («коротких») чисел.