Санкт-Петербургский политехнический университет Петра Великого

Высшая школа интеллектуальных систем и суперкомпьютерных технологий

**Отчёт по лабораторной работе №2**

**Дисциплина**: Низкоуровневое программирование

**Тема**: Программирование EDSAC

Выполнил студент гр. 3530901/90004 И. С. Александрова

(подпись)

Преподаватель А. О. Алексюк

(подпись)

“ ” 2021 г.

Санкт-Петербург

2021

**Оглавление**

[1 Задачи работы 3](#_Toc72934135)

[2. Ход Работы 4](#_Toc72934136)

[1.1 Initial Order 1 4](#_Toc72934137)

[1.2 Initial Order 2 6](#_Toc72934138)

[3 Вывод 8](#_Toc72934139)

[4 Приложение 1 9](#_Toc72934140)

[5 Приложение 2 11](#_Toc72934141)

# Задачи работы

1. Разработать программу для EDSAC, реализующую формирование в памяти десятичного представления целого числа со знаком Initial Orders 1.
2. Выделить определенную 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.

# Ход Работы

Вариант 1: формирование в памяти десятичного представления целого числа со знаком.

## Initial Order 1

Выполняя задачу работы, был написан код и симулирован в .

Листинг Код для Initial Order 1 представлен в приложении 1

В ходе выполнения данной программы заданное пятизначное число преобразуется в символьное представление в памяти. Число задается в адресе 84. После выполнения программы число содержится в диапазоне адресов 150-155, где:

* 155 - знаковый разряд;
* 154 – разряд десятков тысяч;
* 153 – разряд тысяч;
* 152 – разряд сотен;
* 151 – разряд десятков;
* 150 – разряд единиц.

Пример положительного числа:

На вход программы подается исходное число 13089

На выходе получаем:

* 155 – значение 0 (плюс);
* 154 – двоичное значение 1;
* 153 – двоичное значение 3;
* 152 – ноль;
* 151 – двоичное значение 8;
* 150 – двоичное значение 9,

Что соответствует требуемой функциональности.

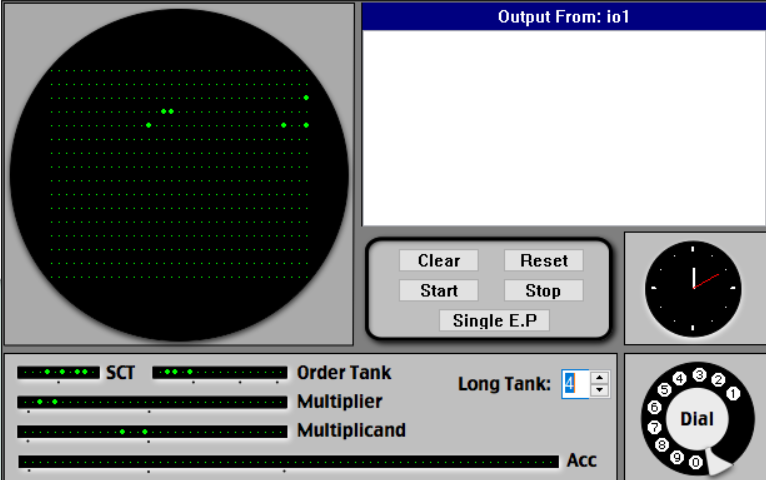


Рис. 2.1 Результат работы программы для положительного числа. Initial Order 1.

Пример отрицательного числа:

На вход программы подается исходное число -12

На выходе получаем:

* 155 – двоичное значение 28 (минус);
* 154 – ноль;
* 153 – ноль;
* 152 – ноль;
* 151 – двоичное значение 1;
* 150 – двоичное значение 2,

Что соответствует требуемой функциональности.

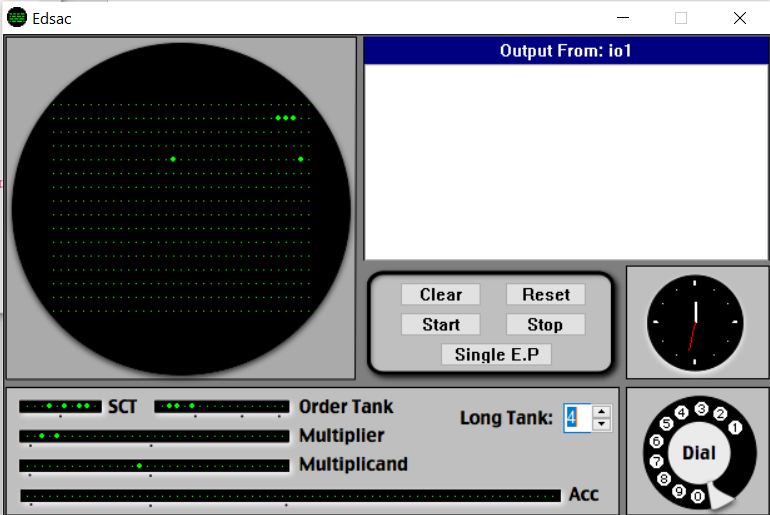


Рис. 2.2 Результат работы программы для отрицательного числа. Initial Order 1.

## Initial Order 2

Выполняя вторую задачу этой лабораторной работы, был написан следующий код.

Листинг Закрытая подпрограмма Initial Order 2 представлена в приложении 2

В данной программе исходное число задается в ячейку 1. Далее идет вызов подпрограммы, которая записывает исходное число в соответствующие ячейки памяти. После выполнения программы число содержится в диапазоне адресов 250 - 255, где:

255 - знаковый разряд;

254 – разряд десятков тысяч;

253 – разряд тысяч;

252 – разряд сотен;

251 – разряд десятков;

250 – разряд единиц.

Пример:

На вход программы подается исходное число -13089.

На выходе получаем:

* 255 – ноль (плюс);
* 254 – двоичное значение 1;
* 253 – двоичное значение 3;
* 252 – ноль;
* 251 – двоичное значение 8;
* 250 – двоичное значение 9,

Что соответствует требуемой функциональности.

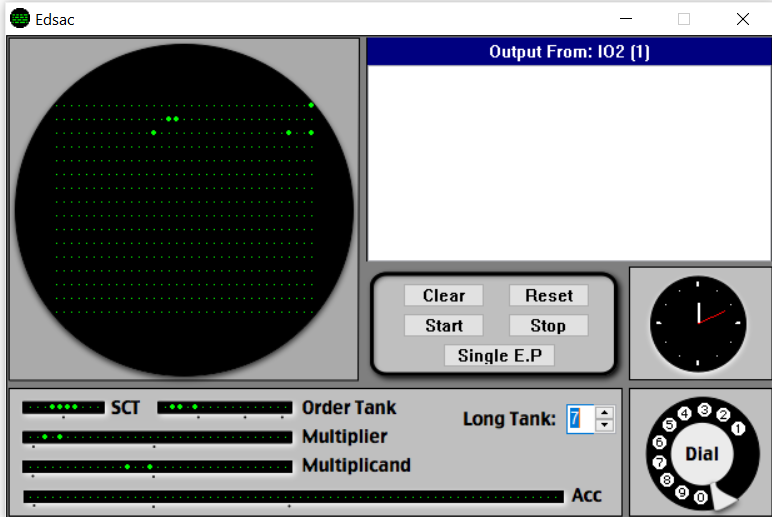


Рис. 2.3 Результат работы программы. Initial Order 2.

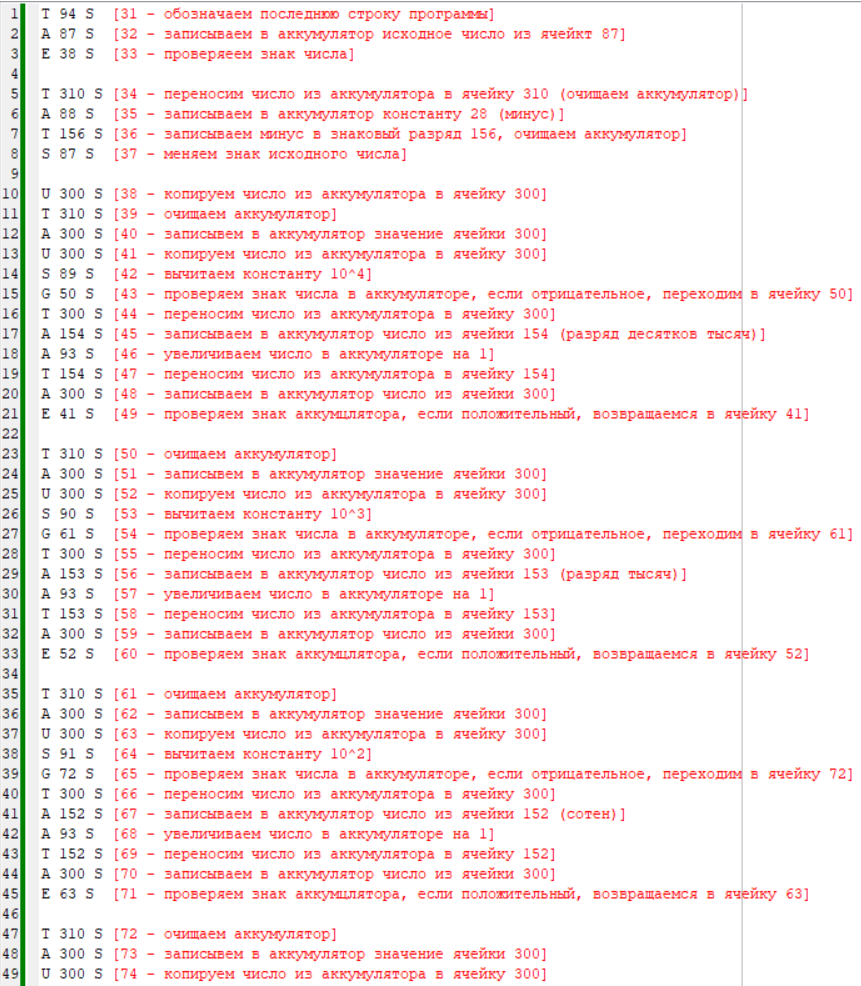
# Вывод

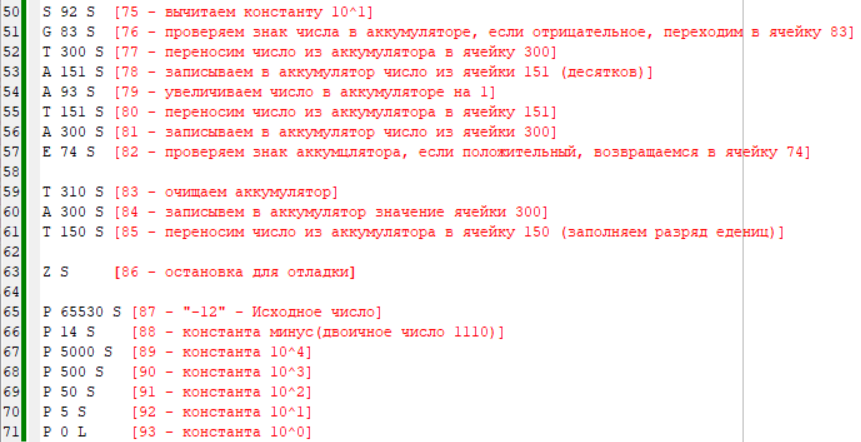
В ходе выполнения лабораторной работы была разработана программа, реализующая формирование в памяти десятичного представления целого числа со знаком EDSAC с помощью симулятора, используя загрузчик IO1. Также, используя загрузчик IO2, была выделена в замкнутую подпрограмму, реализованная на предыдущем шаге программа, и разработана вызывающая ее программа.

Ожидаемые результаты совпали с результатами работы симулятора, значит программа реализована верно.

# Приложение 1

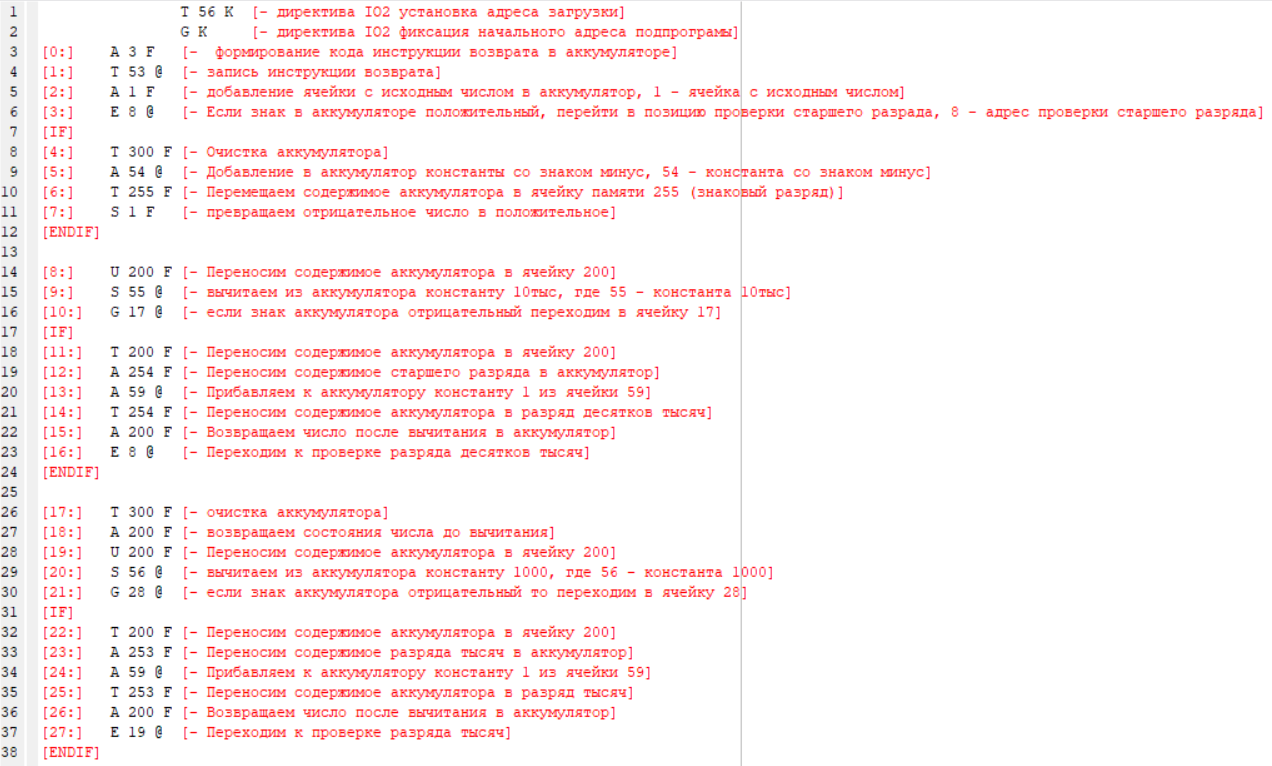
**Код для Initial *Order 1***

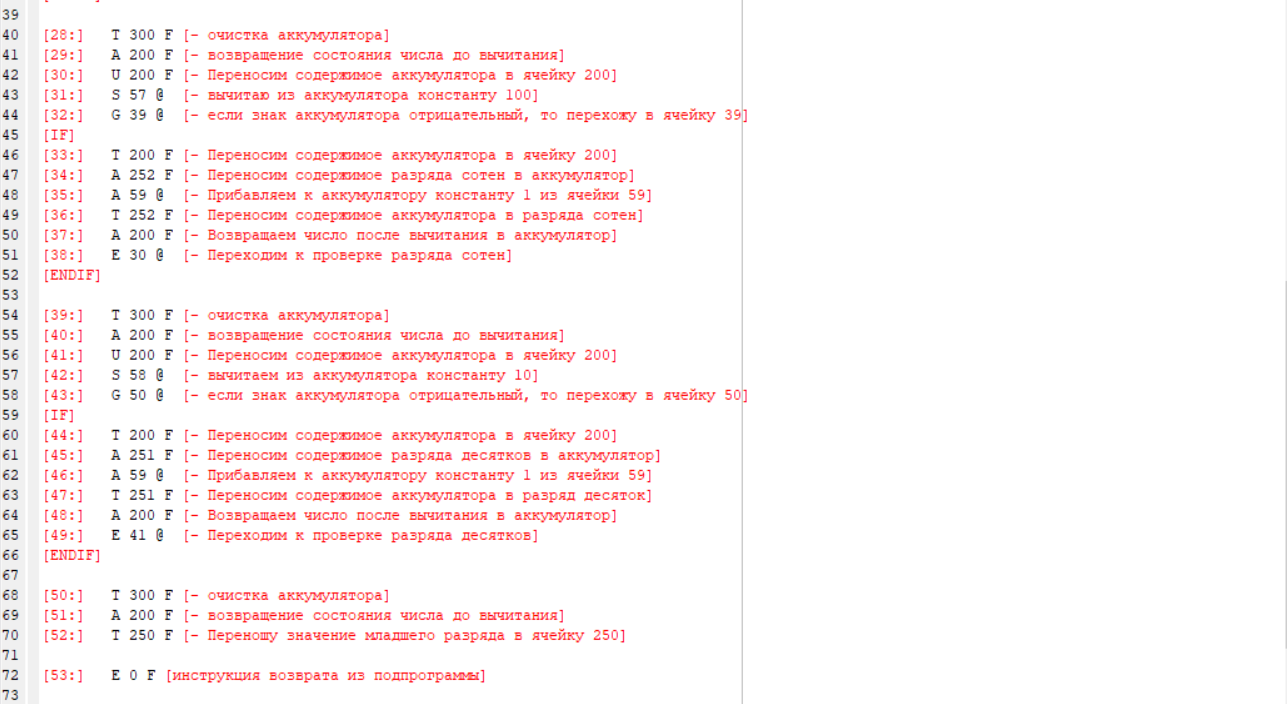




# Приложение 2

**Код для Initial *Order 2***





**Константы:**



**Программа, использующая подпрограмму.**

