МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики»

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ ТЕХНИКИ

## **ЛАБОРАТОРНАЯ РАБОТА №4**

по дисциплине

«Основы профессиональной деятельности»

Вариант № 10744

***Выполнил:***

Студент группы P3110

Чжун Цзяцзюнь

***Преподаватель:***

Блохина Елена Николаевна

Санкт-Петербург, 2025 г.

## Содержание

[Текст задания 3](#_Toc127641422)

[Описание программы 5](#_Toc127641423)

[Вывод 6](#_Toc127641424)

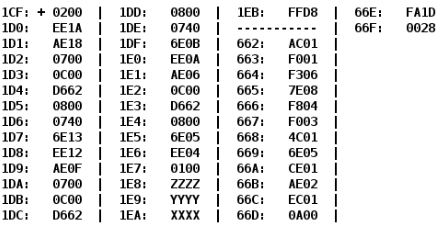
[Таблица трассировки 7](#_Toc127641425)

# 

# 

# Текст задания

По выданному преподавателем варианту восстановить текст заданного варианта программы, определить предназначение и составить описание программы, определить область представления и область допустимых значений исходных данных и результата, выполнить трассировку программы.



|  |  |  |  |
| --- | --- | --- | --- |
| **Адрес** | **Код команды** | **Мнемоника** | **Комментарии** |
| 1CF | 0200 | CLA | Очистка аккумулятора |
| 1D0 | EE1A | ST IP+26 | Очистка результата  AC -> MEM(1EB) |
| 1D1 | AE18 | LD IP+24 | Загрузка в аккумулятор MEM(1EA) (X) -> AC  AC + 1 -> AC |
| 1D2 | 0700 | INC |
| 1D3 | 0C00 | PUSH (ST) | AC -> -(SP) Положим содержимое AC на вершину стека  Вызов подпрограммы по адресу 662: F(X + 1)  Для этого в стек записываем IP, а в IP кладем 662.  (SP)+ -> AC Загрузка результата в аккумулятор |
| 1D4 | D662 | CALL 662 |
| 1D5 | 0800 | POP (LD) |
| 1D6 | 0740 | DEC | AC – 1 -> AC  Вычитание возвращаемого значения функции с R = 0 - 1, сохранение в R = F(X + 1) - 1  Запись содержимого АС в ячейку 1EB |
| 1D7 | 6E13 | SUB IP+19 |
| 1D8 | EE12 | ST IP+18 |
| 1D9 | AE0F | LD IP+15 | Загрузка в аккумулятор MEM(1E9) (Y) -> AC  AC + 1 -> AC |
| 1DA | 0700 | INC |
| 1DB | 0C00 | PUSH | AC -> -(SP) Положим содержимое AC на вершину стека  Вызов подпрограммы по адресу 662:F(Y + 1)  (SP)+ -> AC Загрузка результата в аккумулятор |
| 1DC | D662 | CALL 662 |
| 1DD | 0800 | POP |
| 1DE | 0740 | DEC | AC – 1 -> AC  Вычитание R из F(Y + 1) – 1, сохранение в R = F(Y + 1) – 1 – (F(X + 1) - 1)  Запись содержимого АС в ячейку 1EB |
| 1DF | 6E0B | SUB IP+11 |
| 1E0 | EE0A | ST IP+10 |
| 1E1 | AE06 | LD IP+6 | Загрузка в аккумулятор MEM(1E8) (Z) -> AC |
| 1E2 | 0C00 | PUSH | AC -> -(SP) Положим содержимое AC на вершину стека  Вызов подпрограммы по адресу 662:F(Z)  (SP)+ -> AC Загрузка результата в аккумулятор |
| 1E3 | D662 | CALL 662 |
| 1E4 | 0800 | POP |
| 1E5 | 6E05 | SUB IP+5 | Вычитание R из аккумулятора  Запись содержимого АС в ячейку 1EB |
| 1E6 | EE04 | ST IP+4 |
| 1E7 | 0100 | HLT | Остановка программы |

Подпрограмма:

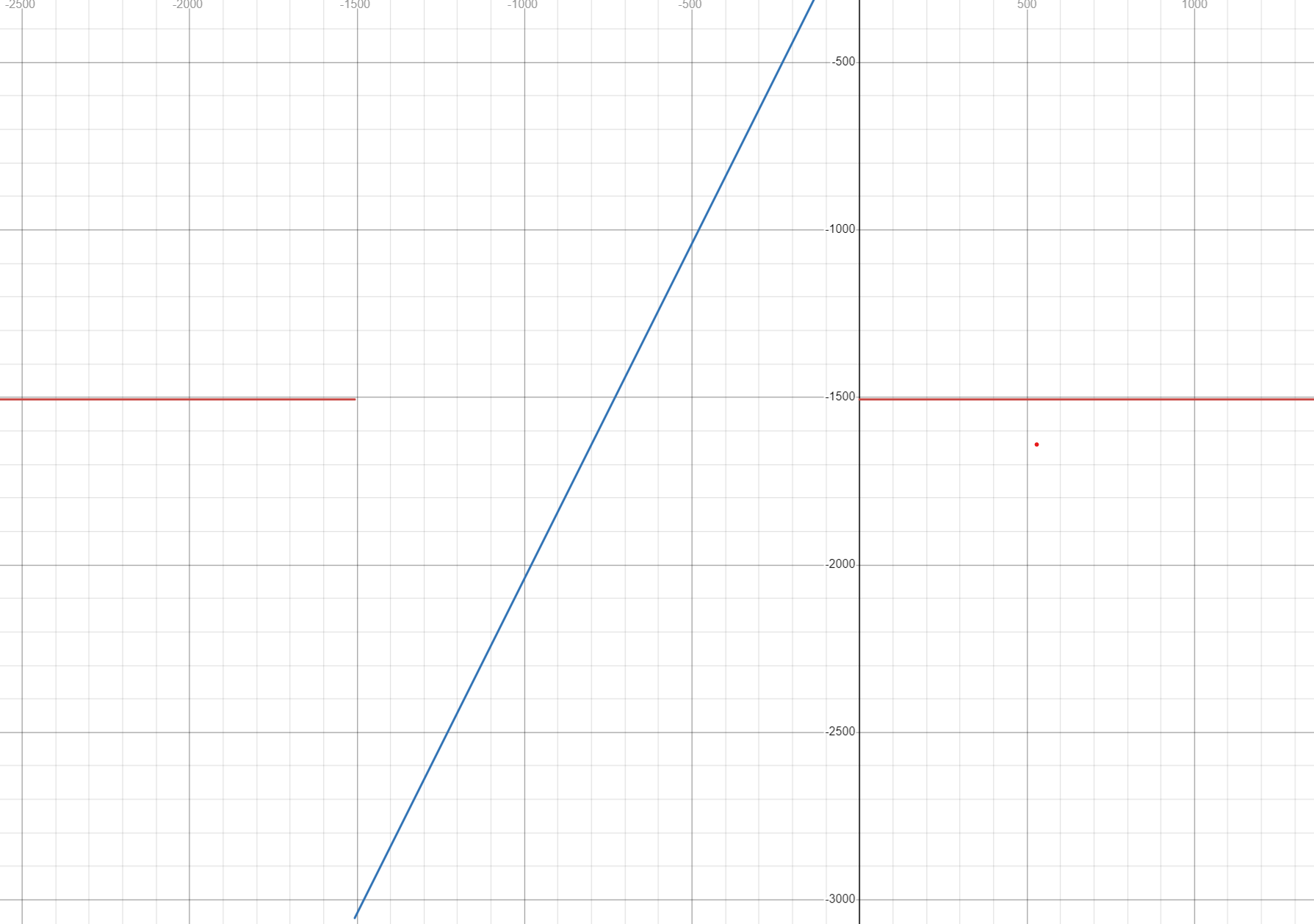
|  |  |  |  |
| --- | --- | --- | --- |
| **Адрес** | **Код команды** | **Мнемоника** | **Комментарии** |
| 662 | AC01 | LD (SP+1) | Загрузка в аккумулятор предпоследнего значения в стеке |
| 663 | F001 | BEQ IP+1 | Если Z == 1, то переход на 665 |
| 664 | F306 | BPL IP+ 6 | Если N == 0, то переход на 66B |
| 665 | 7E08 | CMP IP+8 | Если AC ≤ Q(MEM(66E)), то переход на 66B |
| 666 | F804 | BLT IP+4 |
| 667 | F003 | BEQ IP+3 |
| 668 | 4C01 | ADD (SP+1) | Прибавить аргумент программы |
| 669 | 6E05 | SUB IP+5 | Вычитание значения ячейки 66F |
| 66A | CE01 | JUMP IP+1 | Переход на 66C |
| 66B | AE02 | LD IP+2 | MEM(66E) -> AC Загрузка значения ячейки 66E |
| 66C | EC01 | ST (SP+1) | Сохранение результата в ячейку SP+1  AC -> MEM(SP+1) |
| 66D | 0A00 | RET | Запись в IP адреса продолжения программы, которое лежало наверху SP, увеличение SP |

# Описание программы, подпрограммы и комплекса программ

Назначение программы: нахождения значения функции:

F(Z) – F(Y + 1) + F(X + 1)

График:



**Область представления**

* X, Y, Z, Q, W, R – целые знаковые шестнадцатеричные числа в дополнительном коде.

**Область допустимых значений**

Q = FA1D16 = -1507

W = 002816 = 40

Для того чтобы определить ОДЗ, проанализируем данную функцию. При значении аргумента функции в промежутке [0, 1507], функция вернет значение 1507. При использовании любого значения из заданного промежутка в функции не возникнет переполнения.

При оставшихся значениях аргумента функция вернет выражение 2\*x – 40. На промежутке (-∞, 0] и [1508, +∞) эта функция сегментарно монотонно возрастающая, поэтому рассмотрим минимальное и максимальное значение:

что означает, что на всем промежутке значений аргумента, результат функции будет находиться на отрезке [-10923; 10922].

Так как основная программа вычисляет следующее выражение:

Значит, ОДЗ:

* Z ϵ [-5442; 5481];
* X,Y ϵ [-5443; 5480];
* Результат R ϵ [–32768; 32767] (с учетом заданных Q и W).
* X= 7800
* Y= 100
* Z=-200

**Расположение данных в памяти**

**Основная программа:**

* 1CF-1E7 – команды;
* 1E8(Z), 1E9(Y), 1EA(X) – исходные данные;
* 1EB – итоговый результат.

**Подпрограмма:**

* 662-66D – команды;
* 66E(Q), 66F(W) – константы.

**Адреса первой и последней выполняемой команды**

**Основная программа:**

* Адрес первой команды: 1CF
* Адрес последней команды: 1E7

**Подпрограмма:**

* Адрес первой команды: 662
* Адрес последней команды: 66D

# Таблица трассировки

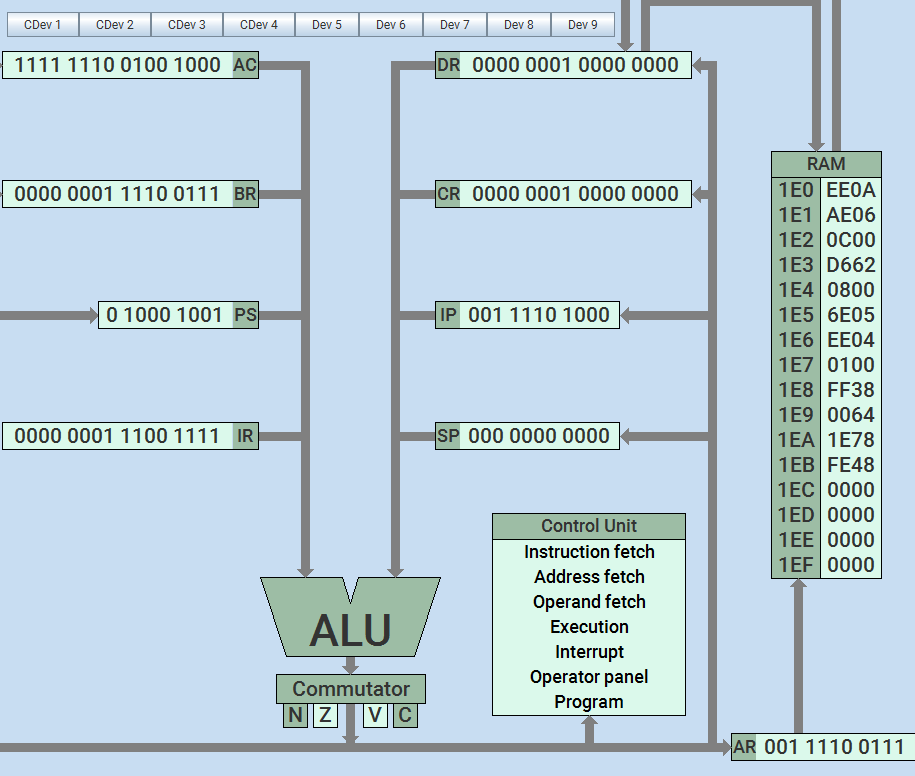
X = 780010= 1E78, Y = 10010 = 0064, Z = -20010 = FF38

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Выполняемая  команда | | Содержимое регистров после выполнения команды | | | | | | | | | | Ячейка, содержимое которой изменилось после выполнения команды | | |
| Адрес | Код команды | IP | CR | AR | DR | SP | BR | AC | ~~PS~~ | NZVC | Адрес | | Новый код |
| 1CF | 0200 | 1D0 | 0200 | 1CF | 0200 | 000 | 01CF | 0000 | ~~004~~ | 0100 |  | |  |
| 1D0 | EE1A | 1D1 | EE1A | 1EB | 0000 | 000 | 001A | 0000 | ~~004~~ | 0100 | 1EB | | 0000 |
| 1D1 | AE18 | 1D2 | AE18 | 1EA | 1E78 | 000 | 0018 | 1E78 | ~~000~~ | 0000 |  | |  |
| 1D2 | 0700 | 1D3 | 0700 | 1D2 | 0700 | 000 | 01D2 | 1E79 | ~~000~~ | 0000 |  | |  |
| 1D3 | 0C00 | 1D4 | 0C00 | 7FF | 1E79 | 7FF | 01D3 | 1E79 | ~~000~~ | 0000 | 7FF | | 1E79 |
| 1D4 | D662 | 662 | D662 | 7FE | 01D5 | 7FE | D662 | 1E79 | ~~000~~ | 0000 | 7FE | | 01D5 |
| 662 | AC01 | 663 | AC01 | 7FF | 1E79 | 7FE | 0001 | 1E79 | ~~000~~ | 0000 |  | |  |
| 663 | F001 | 664 | F001 | 663 | F001 | 7FE | 0663 | 1E79 | ~~000~~ | 0000 |  | |  |
| 664 | F306 | 66B | F306 | 664 | F306 | 7FE | 0006 | 1E79 | ~~000~~ | 0000 |  | |  |
| 66B | AE02 | 66C | AE02 | 66E | FA1D | 7FE | 0002 | FA1D | ~~008~~ | 1000 |  | |  |
| 66C | EC01 | 66D | EC01 | 7FF | FA1D | 7FE | 0001 | FA1D | ~~008~~ | 1000 | 7FF | | FA1D |
| 66D | 0A00 | 1D5 | 0A00 | 7FE | 01D5 | 7FF | 066D | FA1D | ~~008~~ | 1000 |  | |  |
| 1D5 | 0800 | 1D6 | 0800 | 7FF | FA1D | 000 | 01D5 | FA1D | ~~008~~ | 1000 |  | |  |
| 1D6 | 0740 | 1D7 | 0740 | 1D6 | 0740 | 000 | 01D6 | FA1C | ~~009~~ | 1001 |  | |  |
| 1D7 | 6E13 | 1D8 | 6E13 | 1EB | 0000 | 000 | 0013 | FA1C | ~~009~~ | 1001 |  | |  |
| 1D8 | EE12 | 1D9 | EE12 | 1EB | FA1C | 000 | 0012 | FA1C | ~~009~~ | 1001 | 1EB | | FA1C |
| 1D9 | AE0F | 1DA | AE0F | 1E9 | 0064 | 000 | 000F | 0064 | ~~001~~ | 0001 |  | |  |
| 1DA | 0700 | 1DB | 0700 | 1DA | 0700 | 000 | 01DA | 0065 | ~~000~~ | 0000 |  | |  |
| 1DB | 0C00 | 1DC | 0C00 | 7FF | 0065 | 7FF | 01DB | 0065 | ~~000~~ | 0000 | 7FF | | 0065 |
| 1DC | D662 | 662 | D662 | 7FE | 01DD | 7FE | D662 | 0065 | ~~000~~ | 0000 | 7FE | | 01DD |
| 662 | AC01 | 663 | AC01 | 7FF | 0065 | 7FE | 0001 | 0065 | ~~000~~ | 0000 |  | |  |
| 663 | F001 | 664 | F001 | 663 | F001 | 7FE | 0663 | 0065 | ~~000~~ | 0000 |  | |  |
| 664 | F306 | 66B | F306 | 664 | F306 | 7FE | 0006 | 0065 | ~~000~~ | 0000 |  | |  |
| 66B | AE02 | 66C | AE02 | 66E | FA1D | 7FE | 0002 | FA1D | ~~008~~ | 1000 |  | |  |
| 66C | EC01 | 66D | EC01 | 7FF | FA1D | 7FE | 0001 | FA1D | ~~008~~ | 1000 | 7FF | | FA1D |
| 66D | 0A00 | 1DD | 0A00 | 7FE | 01DD | 7FF | 066D | FA1D | ~~008~~ | 1000 |  | |  |
| 1DD | 0800 | 1DE | 0800 | 7FF | FA1D | 000 | 01DD | FA1D | ~~008~~ | 1000 |  | |  |
| 1DE | 0740 | 1DF | 0740 | 1DE | 0740 | 000 | 01DE | FA1C | ~~009~~ | 1001 |  | |  |
| 1DF | 6E0B | 1E0 | 6E0B | 1EB | FA1C | 000 | 000B | 0000 | ~~005~~ | 0101 |  | |  |
| 1E0 | EE0A | 1E1 | EE0A | 1EB | 0000 | 000 | 000A | 0000 | ~~005~~ | 0101 | 1EB | | 0000 |
| 1E1 | AE06 | 1E2 | AE06 | 1E8 | FF38 | 000 | 0006 | FF38 | ~~009~~ | 1001 |  | |  |
| 1E2 | 0C00 | 1E3 | 0C00 | 7FF | FF38 | 7FF | 01E2 | FF38 | ~~009~~ | 1001 | 7FF | | FF38 |
| 1E3 | D662 | 662 | D662 | 7FE | 01E4 | 7FE | D662 | FF38 | ~~009~~ | 1001 | 7FE | | 01E4 |
| 662 | AC01 | 663 | AC01 | 7FF | FF38 | 7FE | 0001 | FF38 | ~~009~~ | 1001 |  | |  |
| 663 | F001 | 664 | F001 | 663 | F001 | 7FE | 0663 | FF38 | ~~009~~ | 1001 |  | |  |
| 664 | F306 | 665 | F306 | 664 | F306 | 7FE | 0664 | FF38 | ~~009~~ | 1001 |  | |  |
| 665 | 7E08 | 666 | 7E08 | 66E | FA1D | 7FE | 0008 | FF38 | ~~001~~ | 0001 |  | |  |
| 666 | F804 | 667 | F804 | 666 | F804 | 7FE | 0666 | FF38 | ~~001~~ | 0001 |  | |  |
| 667 | F003 | 668 | F003 | 667 | F003 | 7FE | 0667 | FF38 | ~~001~~ | 0001 |  | |  |
| 668 | 4C01 | 669 | 4C01 | 7FF | FF38 | 7FE | 0001 | FE70 | ~~009~~ | 1001 |  | |  |
| 669 | 6E05 | 66A | 6E05 | 66F | 0028 | 7FE | 0005 | FE48 | ~~009~~ | 1001 |  | |  |
| 66A | CE01 | 66C | CE01 | 66A | 066C | 7FE | 0001 | FE48 | ~~009~~ | 1001 |  | |  |
| 66C | EC01 | 66D | EC01 | 7FF | FE48 | 7FE | 0001 | FE48 | ~~009~~ | 1001 | 7FF | | FE48 |
| 66D | 0A00 | 1E4 | 0A00 | 7FE | 01E4 | 7FF | 066D | FE48 | ~~009~~ | 1001 |  | |  |
| 1E4 | 0800 | 1E5 | 0800 | 7FF | FE48 | 000 | 01E4 | FE48 | ~~009~~ | 1001 |  | |  |
| 1E5 | 6E05 | 1E6 | 6E05 | 1EB | 0000 | 000 | 0005 | FE48 | ~~009~~ | 1001 |  | |  |
| 1E6 | EE04 | 1E7 | EE04 | 1EB | FE48 | 000 | 0004 | FE48 | ~~009~~ | 1001 | 1EB | | FE48 |
| 1E7 | 0100 | 1E8 | 0100 | 1E7 | 0100 | 000 | 01E7 | FE48 | ~~009~~ | 1001 |  | |  |

F(-200) – F(100 + 1) + F(7800+1) = 2\*(-200) – 40 - (-1507) + (-1507) = -440 = FE4816

Модифицировать Подпрограмму без использования команды RET, то есть она должна быть реализована с использованием других команд.

|  |  |  |  |
| --- | --- | --- | --- |
| **Адрес** | **Код команды** | **Мнемоника** | **Комментарии** |
| 662 | AC01 | LD (SP+1) | Загрузка в аккумулятор предпоследнего значения в стеке |
| 663 | F001 | BEQ IP+1 | Если Z == 1, то переход на 665 |
| 664 | F306 | BPL IP+ 6 | Если N == 0, то переход на 66B |
| 665 | 7E08 | CMP IP+8 | Если AC ≤ Q(MEM(66E)), то переход на 66B |
| 666 | F804 | BLT IP+4 |
| 667 | F003 | BEQ IP+3 |
| 668 | 4C01 | ADD (SP+1) | Прибавить аргумент программы |
| 669 | 6E07 | SUB IP+7 | Вычитание значения ячейки 671 |
| 66A | CE01 | JUMP IP+1 | Переход на 66C |
| 66B | AE04 | LD IP+4 | MEM(670) -> AC Загрузка значения ячейки 670 |
| 66C | EC01 | ST (SP+1) | Сохранение результата в ячейку SP+1  AC -> MEM(SP+1) |
| 66D | 0800 | POP | Только чтобы вернуть значения SP |
| 66E | EE03 | ST IP+3 | Прямое относительное сохранение адреса возврата в 672(IP + 3) |
| 66F | C802 | JUMP IP+2 | Косвенная относительная jump в адрес 672 (в 672 сохраняется адрес возврата) |
| 670 | FA1D | NUM1 (Q) |  |
| 671 | 0028 | NUM2 (W) |  |
| 672 | 0000 | STACK | Сохранение адреса возврата |



# Вывод

Во время выполнения лабораторной работы я узнал о способах связи между программными модулями, научился вызывать и исследовать подпрограммы, работать со стеком, изучил цикл выполнения таких команд как CALL и RET.