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 №3**

по дисциплине

«Основы профессиональной деятельности»

Вариант № 1074

***Выполнил:***

Студент группы P3110

Чжун Цзяцзюнь

***Преподаватель:***

**Блохина Елена Николаевна**

Санкт-Петербург, 2025 г.

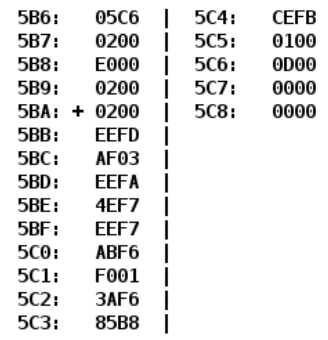
## Содержание

[Текст задания 3](#_Toc127113407)

[Описание программы 4](#_Toc127113408)

[Таблица трассировки 5](#_Toc127113409)

[Вывод 6](#_Toc127113410)

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.

Текст исъодной программы

|  |  |  |  |
| --- | --- | --- | --- |
| **Адрес** | **Код команды** | **Мнемоника** | **Комментарии** |
| 5BA | 0200 | CLA | Очистка аккумулятора |
| 5BB | EEFD | ST IP-3 | Прямое отн. сохранение (Очистка ячейки 5B9) AC -> M (5B9) |
| 5BC | AF03 | LD #3 | Прямая загрузка 0003 -> AC |
| 5BD | EEFA | ST IP-6 | Прямое относительное сохранение AC -> M (5B8) |
| 5BE | 4EF7 | ADD IP-9 | Прямое относительное сложение M(5B6) + AC -> AC |
| 5BF | EEF7 | ST IP-9 | Прямое относительное сохранение AC -> M (5B7) |
| 5C0 | ABF6 | LD –(IP-10) | Косвенная автодекрементальная загрузка: Зн(5B7) –= 1; Зн(5B7) -> A |
| 5C1 | F001 | BEQ IP+1 | Если Z == 1, то IP = IP + 1 -> IP |
| 5C2 | 3**A**F6 | OR (IP-10)+ | Косвенное автоинкрементальное логическое ИЛИ： |
| 5C3 | 85B8 | LOOP 5B8 | M(5B8) – 1 -> M(5B8); Если M(5B8) <= 0, то IP + 1 -> IP |
| 5C4 | CEFB | JUMP IP-5 | Прямой относительный прыжок IP - 5 + 1 -> IP () |
| 5C5 | 0100 | HLT | Останов |

# 

# Проверяем элемент массива равен 0

Если да, то пропускаем

Если нет, то

# Описание программы

~~Вычислить логическое выражение ИЛИ результат ненулевого элемента массива длиной 3~~

**Секрет программы в виде адресации для команды по адресу 5С2**

**~~Перед выполнением команды по адресу 5C2 он будет проверять, выполняется ли циклическое условие, и если оно выполняется, то он не будет пропускать адрес 5C2 снова, и команда по этому адресу выполнит логическую операцию косвенного автоинкрементную по адресу 5B9 и AC для выполнения логической операции «или», чтобы получить результат.~~**

**Программа считает количество ненулевых элементов массива**

**Данные для таблицы трассировка в 10-формате**

**0**

**-589**

**0**

**Область представления**

* arr\_first\_elem, arr\_last\_elem – 11-ти разрядные, адрес БЭВМ.
* arr\_length, result – 16-ти разрядные целые числа, беззнаковое.
* arr[i] – 16-ти разрядные знаковые целые числа.

**Область допустимых значений**

* arr\_length ϵ [1; 0x7FF – arr\_first\_elem + 1]
* result ϵ [0; 216 – 1] (т. к. max = 1 + 2 + 4 + … + 215)
* arr\_first\_elem ϵ [0, 0x5B6 - arr\_length] ∪ [0x5C5, 0x7FF - arr\_length]
* arr\_last\_elem ϵ [arr\_first\_elem; arr\_first\_elem + arr\_length – 1]
* Элементы массива arr[i] ϵ [-32768; 32767] (т. е. [-215; 215-1])

**Расположение данных в памяти**

* 5B6, 5B8, 5C6, 5C7, 5C8 – исходные данные;
* 5B7 – промежуточный результат(текущий адрес);
* 5B9 – итоговый результат;
* 5BA – 5C5 – команды.

**Адреса первой и последней выполняемой команды**

* Адрес первой команды: 5BA
* Адрес последней команды: 5C5

# Таблица трассировки

**Значения:**

Arr[0] = 0x0D00, Arr[1] = 0x0000, Arr[2] = 0x0000, arr\_length = 3, arr\_first\_elem = 0x05C6

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Выполняемая команда | | Содержимое регистров процессора после выполнения команды | | | | | | | | Ячейка, содержимое которой изменилось после выполнения команды | |
| Адрес | Код команды | IP | CR | AR | DR | SP | BR | AC | NZVC | Адрес | Новый код |
| 5BA | 0200 | 5BB | 0200 | 5BA | 0200 | 000 | 05BA | 0000 | 0100 | - | - |
| 5BB | EEFD | 5BC | EEFD | 5B9 | 0000 | 000 | FFFD | 0000 | 0100 | 5B9 | 0000 |
| 5BC | AF03 | 5BD | AF03 | 5BC | 0003 | 000 | 0003 | 0003 | 0000 | - | - |
| 5BD | EEFA | 5BE | EEFA | 5B8 | 0003 | 000 | FFFA | 0003 | 0000 | 5B8 | 0003 |
| 5BE | 4EF7 | 5BF | 4EF7 | 5B6 | 05C6 | 000 | FFF7 | 05C9 | 0000 | - | - |
| 5BF | EEF7 | 5C0 | EEF7 | 5B7 | 05C9 | 000 | FFF7 | 05C9 | 0000 | 5B7 | 05C9 |
| 5C0 | ABF6 | 5C1 | ABF6 | 5C8 | 0000 | 000 | FFF6 | 0000 | 0100 | 5B7 | 05C8 |
| 5C1 | F001 | 5C3 | F001 | 5C1 | F001 | 000 | 0002 | 0000 | 0100 | - | - |
| 5C3 | 85B8 | 5C4 | 85B8 | 5B8 | 0002 | 000 | 0001 | 0000 | 0100 | 5B8 | 0002 |
| 5C4 | CEFB | 5C0 | CEFB | 5C4 | 05C0 | 000 | FFFB | 0000 | 0100 | - | - |
| 5C0 | ABF6 | 5C1 | ABF6 | 5C7 | 0000 | 000 | FFF6 | 0000 | 0100 | 5B7 | 05C7 |
| 5C1 | F001 | 5C3 | F001 | 5C1 | F001 | 000 | 0001 | 0000 | 0100 | - | - |
| 5C3 | 85B8 | 5C4 | 85B8 | 5B8 | 0001 | 000 | 0000 | 0000 | 0100 | 5B8 | 0001 |
| 5C4 | CEFB | 5C0 | CEFB | 5C4 | 05C0 | 000 | FFFB | 0000 | 0100 | - | - |
| 5C0 | ABF6 | 5C1 | ABF6 | 5C6 | 0D00 | 000 | FFF6 | 0D00 | 0000 | 5B7 | 05C6 |
| 5C1 | F001 | 5C2 | F001 | 5C1 | F001 | 000 | 05C1 | 0D00 | 0000 | - | - |
| 5C2 | 3AF6 | 5C3 | 3AF6 | 5B9 | 0000 | 000 | F2FF | 0D00 | 0000 | 5B9 | 0D00 |
| 5C3 | 85B8 | 5C5 | 85B8 | 5B8 | 0000 | 000 | FFFF | 0D00 | 0000 | 5B8 | 0000 |
| 5C5 | 0100 | 5C6 | 0100 | 5C5 | 0100 | 000 | 05C5 | 0D00 | 0000 | - | - |

# Новая таблица трассировки

**Значения:**

Arr[0] = 0x0000, Arr[1] = 0XFDB3, Arr[2] = 0x0000, arr\_length = 3, arr\_first\_elem = 0x05C6

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Выполняемая команда | | Содержимое регистров процессора после выполнения команды | | | | | | | | Ячейка, содержимое которой изменилось после выполнения команды | |
| Адрес | Код команды | IP | CR | AR | DR | SP | BR | AC | NZVC | Адрес | Новый код |
| 5BA | 0200 | 5BB | 0000 | 5BA | 0200 | 000 | 05BA | 0000 | 0100 | - | - |
| 5BB | EEFD | 5BC | EEFD | 5B9 | 0000 | 000 | FFFD | 0000 | 0100 | 5B9 | 0000 |
| 5BC | AF03 | 5BD | AF03 | 5BC | 0003 | 000 | 0003 | 0003 | 0000 | - | - |
| 5BD | EEFA | 5BE | EEFA | 5B8 | 0003 | 000 | FFFA | 0003 | 0000 | 5B8 | 0003 |
| 5BE | 4EF7 | 5BF | 4EF7 | 5B6 | 05C6 | 000 | FFF7 | 05C9 | 0000 | - | - |
| 5BF | EEF7 | 5C0 | EEF7 | 5B7 | 05C9 | 000 | FFF7 | 05C9 | 0000 | 5B7 | 05C9 |
| 5C0 | ABF6 | 5C1 | ABF6 | 5C8 | 0000 | 000 | FFF6 | 0000 | 0100 | 5B7 | 05C8 |
| 5C1 | F001 | 5C3 | F001 | 5C1 | F001 | 000 | 0001 | 0000 | 0100 | - | - |
| 5C3 | 85B8 | 5C4 | 85B8 | 5B8 | 0002 | 000 | 0001 | 0000 | 0100 | 5B8 | 0002 |
| 5C4 | CEFB | 5C0 | CEFB | 5C4 | 05C0 | 000 | FFFB | 0000 | 0100 | - | - |
| 5C0 | ABF6 | 5C1 | ABF6 | 5C7 | FDB3 | 000 | FFF6 | FDB3 | 1000 | 5B7 | 05C7 |
| 5C1 | F001 | 5C2 | F001 | 5C1 | F001 | 000 | 05C1 | FDB3 | 1000 | - | - |
| 5C2 | 3AF6 | 5C3 | 3AF6 | 5B9 | 0000 | 000 | 024C | FDB3 | 1000 | - | - |
| 5C3 | 85B8 | 5C4 | 85B8 | 5B8 | 0001 | 000 | 0000 | FDB3 | 1000 | 5B8 | 0001 |
| 5C4 | CEFB | 5C0 | CEFB | 5C4 | 05C0 | 000 | FFFB | FDB3 | 1000 | - | - |
| 5C0 | ABF6 | 5C1 | ABF6 | 5C6 | 0000 | 000 | FFF6 | 0000 | 0100 | 5B7 | 05C6 |
| 5C1 | F001 | 5C3 | F001 | 5C1 | F001 | 000 | 0001 | 0000 | 0100 | - | - |
| 5C3 | 85B8 | 5C5 | 85B8 | 5B8 | 0000 | 000 | FFFF | 0000 | 0100 | 5B8 | 0000 |
| 5C5 | 0100 | 5C6 | 0100 | 5C5 | 0100 | 000 | 05C5 | 0000 | 0100 | - | - |

# Вывод

Во время выполнения лабораторной работы я научился работать в БЭВМ с массивами, ветвлением и циклами. Я изучил прямую и косвенную адресацию и цикл выполнения таких команд, как LOOP и JUMP

AAF6 загрузка m -> ac косвенная автоинкрементная (относительная) F6 – это смещение (A=1010)

Instruction fetch(IF)

IP - > BR, AR

BR + 1 - > IP; MEM(AR) - > DR

DR -> CR

Adress fetch(AF)

BR + IP - > AR

MEM(AR) - > DR

DR + 1 - > DR

DR - > MEM(AR)

~0 + DR - > DR

Operrand fetch(OF)  
DR -> AR

MEM(AR) - > DR

Execution(EX)

DR - > AC