Федеральное государственное автономное образовательное учреждение

высшего образования «Национальный исследовательский университет

ИТМО»

*Факультет программной инженерии и компьютерной техники*

**Лабораторная работа №5**

**По дисциплине**

**“Основы профессиональной деятельности”**

Вариант:

Выполнил:

Ахроров Кароматуллохон Фирдавсович

Группа: Р3110

Преподаватель:

Блохина Елена Николаевна

Санкт-Петербург, 2025г

Содержание

[ЗАДАНИЕ 2](#_Задание)

[Ход работы 3](#_Ход_работы)

[Описание Программы 3](#_1._Описание_программы_1)

[Область представления 4](#_2._Область_представления)

[Область допустимых значений 4](#_3._Область_допустимых)

[Расположение данных в памяти 5](#_Расположение_данных_в)

[Адреса первой и последней выполняемой команды 6](#_Toc191547624)

[Таблица трассировки 7](#_Таблица_трассировки)

[Вывод 8](#_Вывод)

## Задание

По выданному преподавателем варианту разработать программу асинхронного обмена данными с внешним устройством. При помощи программы осуществить ввод или вывод информации, используя в качестве подтверждения данных сигнал (кнопку) готовности ВУ.

## Текст программы на ассемблере:

ORG 0x1EB ; Начинаем программу

res: WORD 0x620 ; Адрес результата

finish: WORD 0x0D ; Стоп-символ

temp: WORD ? ; Временная ячейка для записи символов

START: CLA

s1: IN 7 ; Ожидание ввода нечётного символа

AND #0x40 ; Зануляем все биты, кроме 6-ого

BEQ s1 ; Если AC=0, то ВУ-3 не готово к обмену

IN 6 ; Ввод байта в AC

ST (res) ; Загружаем в результат значение нечётного символа

chek1: CMP finish ; Проверяем на стоп-слово

BEQ exit ; Выходим если AC равен стоп-слову

s2: CLA ; Очищаем аккумулятор для ввода следующего символа

IN 7 ; Ожидание ввода чётного символа

AND #0x40 ; Зануляем все кроме 6-го бита

BEQ s2 ; Если не готово, то возвращаемся к ожиданию

IN 6 ; Ввод байта в AC

ST temp ; Загружаем значение чётного байта во временную ячейку

SWAB ; Загружаем в результат чётный байт

OR (res) ; Загружаем в результат чётный байт

ST (res) ; Загружаем в результат чётный байт

chek2: LD temp ;

CMP finish ;

BEQ exit ;

LD (res)+ ; увеличиваем ссылку на 1

CLA ; Обнуляем AC

JUMP s1 ; Возвращаемся к вводу нового символа

exit: LD (res)+ ;

HLT ; Остановка

## Ход работы

### Текст исходной программы

|  |  |  |  |
| --- | --- | --- | --- |
| **Адрес** | **Код команды** | **Мнемоника** | **Комментарии** |
| 1EB | 0200 | CLA | Очистка аккумулятора  0 → AC |
| 1EC | 1207 | IN 7 | Чтение регистра состояния ВУ-3 |
| 1ED | 2F40 | AND #40 | Зануляем все биты, кроме 6-ого  Если AC=0, то ВУ-3 не готово к обмену  Запускаем цикл |
| 1EF | F0FD | BEQ IP-3 |
| 1F0 | 1206 | IN 6 | Чтение регистра данных ВУ-3 |
| 1F1 | E8F7 | ST (IP-9) | Сохраняем символ в результат |
| 1F2 | 7EF7 | CMP IP-9 | Проверяем на стоп-слово |
| 1F3 | F00F | BEQ IP+15 | Выходим если AC равен стоп-слову |
| **1F4** | **0200** | CLA | Очистка аккумулятора 0 → AC |
| **1F5** | **1207** | IN 7 | Чтение регистра состояния ВУ-3 |
| **1F6** | **2F40** | AND #40 | Зануляем все биты, кроме 6-ого |
| **1F7** | **F0FC** | BEQ IP-4 | Если AC=0, то ВУ-3 не готово к обмену  Запускаем цикл |
| **1F8** | **1206** | IN 6 | Чтение регистра данных ВУ-3 |
| **1F9** | **EEF1** | ST IP-15 | Загружаем значение чётного байта во временную ячейку |
| **1FA** | **0680** | SWAB | Меняем младший и старший байты местами в AC |
| **1FB** | **38ED** | OR (IP-19) | Соединяем с уже имеющимся результатом |
| **1FC** | **E8EC** | ST (IP-20) | Загружаем в результат значение AC |
| **1FD** | **AEED** | LD IP-19 | Загружаем в AC чётный символ, который нам ввели |
| **1FE** | **7EEB** | CMP IP-21 | Проверяем на стоп-слово |
| **1FF** | **F003** | BEQ IP+3 | Выходим если AC равен стоп-слову |
| **200** | **AAE8** | LD (IP-24)+ | Просто увеличиваем ссылку-результат на 1 |
| **201** | **0200** | CLA | Очистка аккумулятора  0 → AC |
| **202** | **CEEA** | JUMP IP-22 | Возвращаемся к вводу нового символа |
| **203** | **AAE5** | LD (IP-27)+ | Просто увеличиваем ссылку-результат на 1 |
| **204** | **0100** | HLT | Остановка |

## Описание программы

**Описание программы**:

Результат - значение арифметической формулы, использующей возвращаемые значения вызовов подпрограммы **f** на аргументах Y-1, Z-1, X-1

Формула, которую считает программа:

R=-F(X-1) + F(Y-1) + F(Z-1) -1

ОПИ:

Z, Y, X, R, A, B - целые знаковые 16-разрядные числа

-215 ≤ X, Y, Z ≤ 215- 1

ОДЗ:

Расположение данных

Программа

|  |  |
| --- | --- |
| Адрес | Значение |
|  |  |
| 31B | Z  -589 |
|  |  |
| 31С | Y  5 |
|  |  |
| 31D | X  3500 |
|  |  |
| 31E | R (результат выполнения программы) |
|  |  |
| Подпрограмма |  |
|  |  |
| Адрес | Значение |
|  |  |
| 714 | A |
|  |  |
| 715 | B |

Адрес первой и последней выполняемой команды

|  |  |
| --- | --- |
| Адрес первой выполняемой  Команды | Адрес последней выполняемой  команды |
| 300 | 31A |

Подпрограмма:

|  |  |
| --- | --- |
| Адрес первой выполняемой  Команды | Адрес последней выполняемой  команды |
| 70А | 713 |

Таблица трассировки

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Выполняемая команда** | |  | **Содержание регистров в процессоре после выпо лнения команды** | | | | | | | **Ячейка, содержимое которой изменилось после выполнения команды** | |
| Адрес | Код | IP | CR | AR | DR | SP | BR | AC | NZVC | Адрес | Новый код |
| 300 | 0200 | 301 | 0200 | 300 | 0200 | 000 | 0300 | 0000 | 0100 |  |  |
| 301 | EE1C | 302 | EE1C | 31E | 0000 | 000 | 001C | 0000 | 0100 | 31E | 0000 |
| 302 | AE1A | 303 | AE1A | 31D | 0DAC | 000 | 001A | 0DAC | 0000 |  |  |
| 303 | 0740 | 304 | 0740 | 303 | 0740 | 000 | 0303 | 0DAB | 0001 |  |  |
| 304 | 0C00 | 305 | 0C00 | 7FF | 0DAB | 7FF | 0304 | 0DAB | 0001 | 7FF | 0DAB |
| 305 | D70A | 70A | D70A | 7FE | 0306 | 7FE | D70A | 0DAB | 0001 | 7FE | 0306 |
| 70A | AC01 | 70B | AC01 | 7FF | 0DAB | 7FE | 0001 | 0DAB | 0001 |  |  |
| 70B | F205 | 70C | F205 | 70B | F205 | 7FE | 070B | 0DAB | 0001 |  |  |
| 70C | 7E07 | 70D | 7E07 | 714 | 0D39 | 7FE | 0007 | 0DAB | 0001 |  |  |
| 70D | F903 | 711 | F903 | 70D | F903 | 7FE | 0003 | 0DAB | 0001 |  |  |
| 711 | AE02 | 712 | AE02 | 714 | 0D39 | 7FE | 0002 | 0D39 | 0001 |  |  |
| 712 | EC01 | 713 | EC01 | 7FF | 0D39 | 7FE | 0001 | 0D39 | 0001 | 7FF | 0D39 |
| 713 | 0A00 | 306 | 0A00 | 7FE | 0306 | 7FF | 0713 | 0D39 | 0001 |  |  |
| 306 | 0800 | 307 | 0800 | 7FF | 0D39 | 000 | 0306 | 0D39 | 0001 |  |  |
| 307 | 0740 | 308 | 0740 | 307 | 0740 | 000 | 0307 | 0D38 | 0001 |  |  |
| 308 | 6E15 | 309 | 6E15 | 31E | 0000 | 000 | 0015 | 0D38 | 0001 |  |  |
| 309 | EE14 | 30A | EE14 | 31E | 0D38 | 000 | 0014 | 0D38 | 0001 | 31E | 0D38 |
| 30A | AE10 | 30B | AE10 | 31B | FDB3 | 000 | 0010 | FDB3 | 1001 |  |  |
| 30B | 0740 | 30C | 0740 | 30B | 0740 | 000 | 030B | FDB2 | 1001 |  |  |
| 30C | 0C00 | 30D | 0C00 | 7FF | FDB2 | 7FF | 030C | FDB2 | 1001 | 7FF | FDB2 |
| 30D | D70A | 70A | D70A | 7FE | 030E | 7FE | D70A | FDB2 | 1001 | 7FE | 030E |
| 70A | AC01 | 70B | AC01 | 7FF | FDB2 | 7FE | 0001 | FDB2 | 1001 |  |  |
| 70B | F205 | 711 | F205 | 70B | F205 | 7FE | 0005 | FDB2 | 1001 |  |  |
| 711 | AE02 | 712 | AE02 | 714 | 0D39 | 7FE | 0002 | 0D39 | 0001 |  |  |
| 712 | EC01 | 713 | EC01 | 7FF | 0D39 | 7FE | 0001 | 0D39 | 0001 | 7FF | 0D39 |
| 713 | 0A00 | 30E | 0A00 | 7FE | 030E | 7FF | 0713 | 0D39 | 0001 |  |  |
| 30E | 0800 | 30F | 0800 | 7FF | 0D39 | 000 | 030E | 0D39 | 0001 |  |  |
| 30F | 0740 | 310 | 0740 | 30F | 0740 | 000 | 030F | 0D38 | 0001 |  |  |
| 310 | 4E0D | 311 | 4E0D | 31E | 0D38 | 000 | 000D | 1A70 | 0000 |  |  |
| 311 | EE0C | 312 | EE0C | 31E | 1A70 | 000 | 000C | 1A70 | 0000 | 31E | 1A70 |
| 312 | AE09 | 313 | AE09 | 31C | 0005 | 000 | 0009 | 0005 | 0000 |  |  |
| 313 | 0740 | 314 | 0740 | 313 | 0740 | 000 | 0313 | 0004 | 0001 |  |  |
| 314 | 0C00 | 315 | 0C00 | 7FF | 0004 | 7FF | 0314 | 0004 | 0001 | 7FF | 0004 |
| 315 | D70A | 70A | D70A | 7FE | 0316 | 7FE | D70A | 0004 | 0001 | 7FE | 0316 |
| 70A | AC01 | 70B | AC01 | 7FF | 0004 | 7FE | 0001 | 0004 | 0001 |  |  |
| 70B | F205 | 70C | F205 | 70B | F205 | 7FE | 070B | 0004 | 0001 |  |  |
| 70C | 7E07 | 70D | 7E07 | 714 | 0D39 | 7FE | 0007 | 0004 | 1000 |  |  |
| 70D | F903 | 70E | F903 | 70D | F903 | 7FE | 070D | 0004 | 1000 |  |  |
| 70E | 0500 | 70F | 0500 | 70E | 0004 | 7FE | 070E | 0008 | 0000 |  |  |
| 70F | 4E05 | 710 | 4E05 | 715 | 0012 | 7FE | 0005 | 001A | 0000 |  |  |
| 710 | CE01 | 712 | CE01 | 710 | 0712 | 7FE | 0001 | 001A | 0000 |  |  |
| 712 | EC01 | 713 | EC01 | 7FF | 001A | 7FE | 0001 | 001A | 0000 | 7FF | 001A |
| 713 | 0A00 | 316 | 0A00 | 7FE | 0316 | 7FF | 0713 | 001A | 0000 |  |  |
| 316 | 0800 | 317 | 0800 | 7FF | 001A | 000 | 0316 | 001A | 0000 |  |  |
| 317 | 0740 | 318 | 0740 | 317 | 0740 | 000 | 0317 | 0019 | 0001 |  |  |
| 318 | 4E05 | 319 | 4E05 | 31E | 1A70 | 000 | 0005 | 1A89 | 0000 |  |  |
| 319 | EE04 | 31A | EE04 | 31E | 1A89 | 000 | 0004 | 1A89 | 0000 | 31E | 1A89 |
| 31A | 0100 | 31B | 0100 | 31A | 0100 | 000 | 031A | 1A89 | 0000 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |

# Вывод

Во время выполнения лабораторной работы я научился работать в БЭВМ с подпрограммами, а также определять график функции заданной подпрограммы. Научился новым командам.

Написать программу, которая суммирует 5 32-разрядных числа.

ORG 0x10

FIRST\_H: WORD 0x0000

FIRST\_L: WORD 0x7FFF

SECND\_H: WORD 0x0000

SECND\_L: WORD 0x7FFF

THIRD\_H: WORD 0x0000

THIRD\_L: WORD 0x7FFF

FOURTH\_H: WORD 0x0000

FOURTH\_L: WORD 0x7FFF

FIFTH\_H: WORD 0x0000

FIFTH\_L: WORD 0x7FFF

R\_H: WORD 0

R\_L: WORD 0

START: CLA

ST R\_H ; Очищаем R\_H

ST R\_L ; Очищаем R\_L

LD FIRST\_L ; Загружаем в AC младший байт первого эл.

ADD SECND\_L ; Складываем с AC младший байт второго эл. (При переполнении C=1)

ST R\_L ; Загружаем в результат в R\_L (C не меняется)

LD FIRST\_H ; Загружаем в AC старший байт (C не меняется)

ADC SECND\_H ; SECND\_H + AC + C → AC

ADC R\_H ; Доб. результату значение AC (необязательно так как R\_H=0)

ST R\_H ; Загружаем значение в R\_H

LD THIRD\_L ; Загружаем в AC THIRD\_L

ADD R\_L ; Доб. R\_L в AC (при переполнении С= 1)

ST R\_L ; Загружаем в R\_L AC (C не меняется)

LD THIRD\_H ; Загружаем в AC THIRD\_H (C не меняются)

ADC R\_H ; R\_H + AC + C → AC

ST R\_H ; Загружаем AC в R\_H

LD FOURTH\_L ; Загружаем в AC FOURTH \_L

ADD R\_L ; Доб. R\_L в AC (при переполнении С= 1)

ST R\_L ; Загружаем в R\_L AC (C не меняется)

LD FOURTH\_H ; Загружаем в AC FOURTH\_H(C не меняются)

ADC R\_H ; R\_H + AC + C → AC

ST R\_H ; Загружаем AC в R\_H

LD FIFTH\_L ; Загружаем в AC FIFTH \_L

ADD R\_L ; Доб. R\_L в AC (при переполнении С= 1)

ST R\_L ; Загружаем в R\_L AC (C не меняется)

LD FIFTH\_H ; Загружаем в AC FIFTH\_H(C не меняются)

ADC R\_H ; R\_H + AC + C → AC

ST R\_H ; Загружаем AC в R\_H

HLT