

Федеральное государственное автономное образовательное

учреждение высшего образования

«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Направление подготовки: 09.03.04 – Системное и прикладное программное обеспечение

Дисциплина «Основы профессиональной деятельности»

**Отчёт по лабораторной работе №5**

**Асинхронный обмен данными с ВУ**

Вариант №1565

Выполнил

Чимирев Игорь Олегович

P3115

Проверил

Блохина Елена Николаевна

Санкт – Петербург, 2025

Оглавление

[Задание 3](#_Toc195010291)

[Текст исходной программы и данные 3](#_Toc195010292)

[Программа на языке Ассемблера БЭВМ 4](#_Toc195010293)

[Назначение программы 4](#_Toc195010294)

[Описание и назначение исходных данных, область представления и область допустимых значений исходных данных и результата 4](#_Toc195010295)

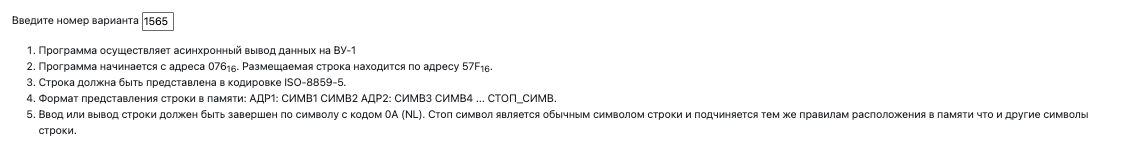
[Расположение в памяти ЭВМ программы, исходных данных и результатов 5](#_Toc195010296)

[Адреса первой и последней выполняемой команд программы 5](#_Toc195010297)

[Трассировка 5](#_Toc195010298)

[Заключение 7](#_Toc195010299)

# 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. 

Исходная строка: Трава%

|  |  |  |  |
| --- | --- | --- | --- |
| Символ | ISO-8859-5 | UTF-8 | UTF-16 |
| Т | B2 | DO A2 | 04 22 |
| р | E0 | D1 80 | 04 40 |
| а | D0 | D0 B0 | 04 30 |
| в | D2 | D0 B2 | 04 32 |
| а | D0 | D0 B0 | 04 30 |
| % | 25 | 00 25 | 00 25 |

# Текст исходной программы и данные

Данные:

|  |  |  |  |
| --- | --- | --- | --- |
| Адрес | Содержимое ячейки памяти | Обозначение | Комментарии |
| 072 | 057F | ADDR | Адрес начала строки |
| 073 | 0000 | I | Адрес текущей ячейки с символами |
| 074 | 0000 | BUFF | Ячейка для временного хранения данных |
| 075 | 000A | STOP\_C | Стоп-символ |

Текст исходной программы:

|  |  |  |  |
| --- | --- | --- | --- |
| Адрес | Код команды | Мнемоника | Комментарии |
| 076 | 0200 | CLA | Очистка AC |
| 077 | AEFA | LD (IP – 6) | Загрузка в AC переменной ADDR, расположенную по адресу (072) |
| 078 | EEFA | ST (IP – 6) | Загрузка в переменную I адреса текущего символа строки (первый символ) |
| 079 | 1203 | IN 3 | Ожидание готовности первого символа (SR (#3) -> AC) |
| 07A | 2F40 | AND #40 | Бит 6 SR == 0 (“Готов” нажата?) (M & AC -> AC) |
| 07B | F0FD | BEQ (IP – 3) | Нет - «Спин-луп» (IF Z == 1 then IP – 3 + 1 -> IP) |
| 07C | AAF6 | LD (IP - 10)+ | Загрузка в AC значения по адресу, записанном в ячейке 073 (I). К значению ячейки 073 прибавляем единицу |
| 07D | EEF6 | ST (IP - 10) | Загрузка AC в BUFF (074) |
| 07E | 0680 | SWAB | Обмен старшего и младшего байта |
| 07F | 0600 | SXTB | Расширение знака младшего байта |
| 080 | 7EF4 | CMP (IP – 12) | Установка флагов по результату AC – (075) (стоп-символ) |
| 081 | F00A | BEQ (IP + 10) | IF Z == 1 then IP + 10 + 1 -> IP (если элемент равен стоп-символу, переходим в ячейку 08C) |
| 082 | 1302 | OUT 2 | Вывод первого символа (младший байт AC) в ВУ-1 |
| 083 | 1203 | IN 3 | Ожидание готовности первого символа (SR (#3) -> AC) |
| 084 | 2F40 | AND #40 | Бит 6 SR == 0 (“Готов” нажата?) (M & AC -> AC) |
| 085 | F0FD | BEQ (IP – 3) | Нет - «Спин-луп» (IF Z == 1 then IP – 3 + 1 -> IP) |
| 086 | AEED | LD (IP - 19) | Загрузка в AC значения из BUFF (074) |
| 087 | 0600 | SXTB | Расширение знака младшего байта |
| 088 | 7EEC | CMP (IP – 20) | Установка флагов по результату AC – (075) (стоп-символ) |
| 089 | F002 | BEQ (IP + 2) | IF Z == 1 then IP + 2 + 1 -> IP (если элемент равен стоп-символу, переходим в ячейку 08C) |
| 08A | 1302 | OUT 2 | Вывод второго символа (младший байт AC) в ВУ-1 |
| 08B | CEED | JUMP (IP – 19) | Переход в ячейку 079 |
| 08C | 0100 | HLT | Останов |

# Программа на языке Ассемблера БЭВМ

ORG 0x072

ADDR: **WORD** $ARRAY ; string address

I: **WORD** 0 ; current chars of the string

BUFF: **WORD** 0 ; buffer cell

STOP\_C: **WORD** 0x0A ; EOF char

START: CLA ; clear

LD ADDR

ST I ; I = ADDR

S1: **IN** 3 ; \

**AND** #0X40 ; -> SPIN-LOOP while not ready

BEQ S1 ; /

LD (I)+ ; increment of the pointer to the current chars

ST BUFF ; load 2 chars (2 chars in one cell) into a buffer cell (BUFF)

SWAB ; \

SXTB ; -> sign extension (get the first char)

**CMP** STOP\_C ; compare with the stop char (STOP\_C)

BEQ STOP ; if equal, then end the output

**OUT** 2 ; output the first char

S2: **IN** 3 ; \

**AND** #0X40 ; -> SPIN-LOOP while not ready

BEQ S2 ; /

LD BUFF ; load the buffer cell

SXTB ; sign extension (get the second char)

**CMP** STOP\_C ; compare with the stop char (STOP\_C)

BEQ STOP ; if equal, then end the output

**OUT** 2 ; output the second char

JUMP S1 ; move on to the output of a new cell (2 chars)

STOP: **HLT** ; exit program

ORG 0x57F

ARRAY: **WORD** 0xB2E0

**WORD** 0xD0D2

**WORD** 0xD025

**WORD** 0x0A00

# Назначение программы

Назначение программы:

Асинхронный вывод данных на ВУ-1 (строка выводится на ВУ-1).

Строка представлена в кодировке ISO-8859-5, формат представления в памяти: АДР1: СИМВ1 СИМВ2, АДР2: СИМВ3 СИМВ4 … СТОП\_СИМВЛ. Признаком завершения вывода является символ 0A (NL).

# Описание и назначение исходных данных, область представления и область допустимых значений исходных данных и результата

Описание и назначение исходных данных:

* ADDR – адрес начала строки в памяти БЭВМ.
* I – адрес текущей ячейки памяти с символами строки (изначально равна ADDR).
* BUFF – ячейка для временного хранения данных.
* STOP\_C – стоп-символ (константа).
* СИМВ – символ строки.

Область представления:

* СИМВ – 8-ми разрядное целое беззнаковое число

[0; 28 – 1], но символы хранятся в 16-ти разрядных ячейках (беззнаковое 16-ти разрядное число) по два в каждом.

* I, ADDR – 11-разрядные беззнаковые числа (адреса БЭВМ)
* BUFF, STOP\_C – 16-ти разрядные беззнаковые числа

[0; 216 – 1]

Область допустимых значений (ОДЗ):

# Расположение в памяти ЭВМ программы, исходных данных и результатов

* Программа расположена в ячейках **076** – **08С**
* Исходные данные для программы расположены в ячейках **072 – 075**
* Исходная строка расположена в ячейках **57F** – **7FF**

# Адреса первой и последней выполняемой команд программы

* Первая – **076**
* Последняя – **08С**

# Трассировка

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Выполняемая команда** | | **Содержимое регистров процессора после выполнения команды** | | | | | | | | **Ячейка, содержимое которой изменилось после выполнения команды** | |
| **Адрес** | **Код** | **IP** | **CR** | **AR** | **DR** | **SP** | **BR** | **AC** | **NZVC** | **Адрес** | **Новый код** |
| 076 | 0200 | 077 | 0200 | 076 | 0200 | 000 | 0076 | 0000 | 0100 |  |  |
| 077 | AEFA | 078 | AEFA | 072 | 057F | 000 | FFFA | 057F | 0000 |  |  |
| 078 | EEFA | 079 | EEFA | 073 | 057F | 000 | FFFA | 057F | 0000 | 073 | 057F |
| 079 | 1203 | 080 | 1203 | 079 | 1203 | 000 | 0079 | 0500 | 0000 |  |  |
| 07A | 2F40 | 07B | 2F40 | 07A | 0040 | 000 | 0040 | 0040 | 0000 |  |  |
| 07B | F0FD | 07C | F0FD | 07B | F0FD | 000 | 007B | 0040 | 0000 |  |  |
| 07C | AAF6 | 07D | AAF6 | 57F | B2E0 | 000 | FFF6 | B2E0 | 1000 | 073 | 0580 |
| 07D | EEF6 | 07E | EEF6 | 074 | B2E0 | 000 | FFF6 | B2E0 | 1000 | 074 | B2E0 |
| 07E | 0680 | 07F | 0680 | 07E | 0680 | 000 | 007E | E0B2 | 1000 |  |  |
| 07F | 0600 | 080 | 0600 | 07F | 0600 | 000 | 007F | FFB2 | 1000 |  |  |
| 080 | 7EF4 | 081 | 7EF4 | 075 | 000A | 000 | FFF4 | FFB2 | 1001 |  |  |
| 081 | F00A | 082 | F00A | 081 | F00A | 000 | 0081 | FFB2 | 1001 |  |  |
| 082 | 1302 | 083 | 1302 | 082 | 1302 | 000 | 0082 | FFB2 | 1001 | ВУ-1 | B2 |
| 083 | 1203 | 084 | 1203 | 083 | 1203 | 000 | 0083 | FF40 | 1001 |  |  |
| 084 | 2F40 | 085 | 2F40 | 084 | 0040 | 000 | 0040 | 0040 | 0001 |  |  |
| 085 | F0FD | 086 | F0FD | 085 | F0FD | 000 | 0085 | 0040 | 0001 |  |  |
| 086 | AEED | 087 | AEED | 074 | B2E0 | 000 | FFED | B2E0 | 1001 |  |  |
| 087 | 0600 | 088 | 0600 | 087 | 0600 | 000 | 0087 | FFE0 | 1001 |  |  |
| 088 | 7EEC | 089 | 7EEC | 075 | 000A | 000 | FFEC | FFE0 | 1001 |  |  |
| 089 | F002 | 08A | F002 | 089 | F002 | 000 | 0089 | FFE0 | 1001 |  |  |
| 08A | 1302 | 08B | 1302 | 08A | 1302 | 000 | 008A | FFE0 | 1001 | ВУ-1 | E0 |
| 08B | CEED | 079 | CEED | 08B | 0079 | 000 | FFED | FFE0 | 1001 |  |  |

# Заключение

Во время выполнения данной лабораторной работы я изучил работу БЭВМ с ВУ в асинхронном режиме.

Реализовать самый простой калькулятор на ву-9 по следующему правилу

Ввели цифру, потом операцию (доступны только +-\*)

Если -+ то вводим следующее число и выпполняем операцию

Если звездочка, то умножаем на 2

Если вместо цифры или операции введено что-то другое, то ждем правильно ввода

Выполнили операцию, то есть вывести ее на ву-5, закончили программу.

ORG 0x1EC

START:

CLA

; Ввод первой цифры (0-9)

INPUT\_FIRST:

CALL READ\_KEYBOARD

CMP #0x0A ; Проверка >= 10

BGE INPUT\_FIRST

CMP #0x00

BLT INPUT\_FIRST

ST FIRST\_NUM

; Ввод операции

INPUT\_OP:

CALL READ\_KEYBOARD

CMP #0x0B ; '+'

BEQ OP\_PLUS

CMP #0x0A ; '-'

BEQ OP\_MINUS

CMP #0x0D ; '\*'

BEQ OP\_MUL

JUMP INPUT\_OP

OP\_PLUS:

LD #0x0B

JUMP SAVE\_OP

OP\_MINUS:

LD #0x0A

JUMP SAVE\_OP

OP\_MUL:

LD #0x0D

SAVE\_OP:

ST OP\_CODE

; Ввод второй цифры (только для + и -)

LD OP\_CODE

CMP #0x0D

BEQ SKIP\_SECOND ; Для умножения пропускаем ввод

INPUT\_SECOND:

CALL READ\_KEYBOARD

CMP #0x0A

BGE INPUT\_SECOND

CMP #0x00

BLT INPUT\_SECOND

ST SECOND\_NUM

SKIP\_SECOND:

; Выполнение операции

LD OP\_CODE

CMP #0x0B

BEQ DO\_ADD

CMP #0x0A

BEQ DO\_SUB

CMP #0x0D

BEQ DO\_MUL

DO\_ADD:

LD FIRST\_NUM

ADD SECOND\_NUM

JUMP PRINT\_RES

DO\_SUB:

LD FIRST\_NUM

SUB SECOND\_NUM

JUMP PRINT\_RES

DO\_MUL:

LD FIRST\_NUM

ASL

; Вывод результата с поддержкой отрицательных

PRINT\_RES:

ST RESULT\_TMP

; Проверка на отрицательность

LD RESULT\_TMP

BMI NEGATIVE\_NUM ; Если отрицательный

; Положительное число

CMP #10

BLT SINGLE\_DIGIT

; Вывод двузначного

LD #0x31 ; '1'

CALL PRINT\_DISPLAY

LD RESULT\_TMP

SUB #10

ADD #0x30

CALL PRINT\_DISPLAY

JUMP END\_PRINT

SINGLE\_DIGIT:

LD RESULT\_TMP

ADD #0x30

CALL PRINT\_DISPLAY

JUMP END\_PRINT

NEGATIVE\_NUM:

; Вывод минуса

LD #0x2D ; '-' в ISO-8859-5

CALL PRINT\_DISPLAY

; Преобразование в положительное

LD RESULT\_TMP

NEG ; Инвертируем число

ST RESULT\_TMP

; Вывод модуля

CMP #10

BLT SINGLE\_DIGIT\_NEG

LD #0x31

CALL PRINT\_DISPLAY

LD RESULT\_TMP

SUB #10

ADD #0x30

CALL PRINT\_DISPLAY

JUMP END\_PRINT

SINGLE\_DIGIT\_NEG:

LD RESULT\_TMP

ADD #0x30

CALL PRINT\_DISPLAY

END\_PRINT:

HLT

; Данные

FIRST\_NUM: WORD ?

SECOND\_NUM: WORD ?

OP\_CODE: WORD ?

RESULT\_TMP: WORD ?

READ\_KEYBOARD:

CHECK\_KB:

IN 0x1D

AND #0x40

BEQ CHECK\_KB

IN 0x1C

RET

PRINT\_DISPLAY:

ST OUT\_TMP

CHECK\_DISP:

IN 0x0D

AND #0x40

BEQ CHECK\_DISP

LD OUT\_TMP

OUT 0x0C

RET

OUT\_TMP: WORD ?