

Федеральное государственное автономное образовательное

учреждение высшего образования

«Национальный исследовательский университет ИТМО»

**Отчёт по лабораторной работе 5**

Предмет: ОПД

Варинат: 1564

Выполнил**:** студент группы Р3115 Храбров Артём Алексеевич

Проверил**:** Блохина Елена Николаевна

Дата сдачи: xx.xx.xx

2025

Задание

1. Программа осуществляет асинхронный вывод данных на ВУ-1
2. Программа начинается с адреса 1EC16. Размещаемая строка находится по адресу 5ED16.
3. Строка должна быть представлена в кодировке КОИ-8.
4. Формат представления строки в памяти: АДР0: ДЛИНА АДР1: СИМВ2 СИМВ1 АДР2: СИМВ4 СИМВ3 ..., где ДЛИНА - 16 разрядное слово, где значащими являются 8 младших бит.
5. Вывод строки начинается со вывода количества символов (1 байт), и должен быть завершен по выводу их необходимого количества.

Строка: Лимон-

Код на ассемблере

ORG 0x5ED ; Исходная строка

WORD 0x0006 ; Колво символов

WORD 0xE9EC

WORD 0xEFED

WORD 0x2DEE

ORG 0x1EC ; Основная программа

START: CLA

LD STR\_ADDR ; Загрузка адреса строки

ST ADDR ; Копирование адреса строки

LD (ADDR)+ ; Получение длины слова

AND #0xFF ; Очистка старших байт

CALL PRINT ; Вывод длины

BEQ EXIT ; Завершение если 0

ST LEN ; Сохранение длины

START\_LOOP:

LD (ADDR)+ ; Начало цикла вывода

CALL PRINT ; Вывод младшего символа

SWAB ; Обмен байтами

ST TMP ; Промежуточно сохраняем аккумулятор

LD LEN

DEC ; Отнимаем счётчик символов

BEQ EXIT ; Выход если всё напечатано

ST LEN

LD TMP ; Обратно загружаем АС

CALL PRINT ; Вывод старшего символа

LOOP LEN

JUMP START\_LOOP

EXIT: HLT

STR\_ADDR: WORD 0x5ED ; Адрес строки в памяти

ADDR: WORD ? ; Переменная для прохода по массиву

LEN: WORD ? ; Длина строки

TMP: WORD ? ; Переменная для промежуточного хранения

PRINT: ; Подпрограмма печатает содержимое аккумулятора по готовности ВУ

ST TMP\_AC

CHECK: IN 0x3

AND #0x40

BEQ CHECK

LD TMP\_AC

OUT 0x2

RET

TMP\_AC: WORD ? ; Переменная промежуточного хранения АС

Текст исходной программы

Программа

|  |  |  |  |
| --- | --- | --- | --- |
| **Адрес** | **Код команды** | **Мнемоника** | **Комментарий** |
| 1EC | 200 | CLA | Очистка аккумулятора |
| 1ED | AE14 | LD (IP+20) | Загрузка адреса строки |
| 1EE | EE14 | ST (IP + 20) | Копирование адреса строки в переменную |
| 1EF | AA13 | LD (IP + 19)+ | Загрузка длины строки |
| 1F0 | 2FFF | AND #FF | Очищаем старшие 8 байт от мусора |
| 1F1 | DE13 | CALL (IP + 19) | Напечатать длину символа |
| 1F2 | F00D | BEQ (IP + 14) | Выход если длина 0 |
| 1F3 | EE0F | ST (IP + 15) | Сохраням длину |
| 1F4 | AA0D | LD (IP + 14)+ | Загружаем слово |
| 1F5 | DE0F | CALL (IP + 15) | Печатаем первый символ |
| 1F6 | 680 | SWAB | Обмен байтами |
| 1F7 | EE0C | ST (IP + 13) | Сохраняем аккумулятор промежуточно |
| 1F8 | AE0A | LD (IP + 10) | Получаем длину |
| 1F9 | 740 | DEC | Вычитаем напечатанный символ |
| 1FA | F005 | BEQ (IP + 5) | Проверка, остались ли еще символы |
| 1FB | EE07 | ST (IP + 7) | Сохраняем длину |
| 1FC | AE07 | LD (IP + 7) | Загружаем обратно аккумулятор |
| 1FD | DE07 | CALL (IP + 7) | Печатаем второй символ |
| 1FE | 8E04 | LOOP (IP + 4) |  |
| 1FF | CEF4 | JUMP (IP – 12) | Переход на начало цикла |
| 200 | 100 | HLT | Выход из программы |

Подпрограмма

Подпрограмма выполняет функцию асинхронного вывода младшего байта аккумулятора на ВУ-1 по его готовности. После вызова подпрограммы содержимое аккумулятора никак не изменяется.

|  |  |  |  |
| --- | --- | --- | --- |
| 205 | EE06 | ST TMP\_AC | Сохраняем промежуточно аккумулятор |
| 206 | 1203 | IN 0x3 | Ожидаем готовность ВУ-1 |
| 207 | 2F40 | AND #40 |
| 208 | F0FD | BEQ (IP – 3) |
| 209 | AE02 | LD (IP + 2) | Загружаем обратно аккумулятор |
| 20A | 1302 | OUT 0x2 | Выводим младший байт |
| 20B | 0A00 | RET | Возврат |

**Назначение программы**

Программа осуществляет асинхронный вывод на ВУ-1. Строка в памяти хранится в виде:

Длина (кол-во символов)

Символ2Символ1

Символ4Символ3

...

Выводимое сообщение: Лимон-

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Символ | Л | и | м | о | н | - |
| КОИ-8 | EC | E9 | ED | EF | EE | 2D |
| UTF-8 | D09B | D0B8 | D0BC | D0BE | D0BD | 2D |
| UTF-16 | 1B04 | 3804 | 3C04 | 3E04 | 3D04 | 2D00 |

**Область представления**

* LEN – Количество символов в слове, беззнаковое число, 8 разрядов
* STR\_ADDR – Адрес строки в памяти, 11 разрядов
* ADDR – Адрес текущего слова из 2 символов, 11 разрядов
* TMP, TMP\_AC – переменные для промежуточного хранения данных в программе и подпрограмме соответсвенно, 16 битный набор значений.

Область допустимых значений:

LEN - [0, 28-1]

Символы слова - [0, 28-1]

TMP, TMP\_AC - [0, 216-1]

Количество символов в программе – [0, 255]

Расположение в памяти и назначение исходных данных

Основная программа:

Расположение программы: **1EC** – **200**

STR\_ADDR: **201**

ADDR: **202**

LEN: **203**

TMP: **204**

Cтрока: **5ED – 5F0** (длина + 3 слова)

Подпрограмма:

Расположение подпрограммы: **205 – 20B**

TMP\_AC: **20C**

Адреса первой и последней исполняемой команды

Основная программа: Первая – **1EC,** последняя - **200**

Подпрограмма: Первая – **205**, последняя – **20B**

Трассировка программы

Таблица трассировки программы для первых 2 символов:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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 |
| 1EC | 0200 | 1ED | 0200 | 1EC | 0200 | 000 | 01EC | 0000 | 0100 |  |  |
| 1ED | AE13 | 1EE | AE13 | 201 | 05ED | 000 | 0013 | 05ED | 0000 |  |  |
| 1EE | EE13 | 1EF | EE13 | 202 | 05ED | 000 | 0013 | 05ED | 0000 | 202 | 05ED |
| 1EF | AA12 | 1F0 | AA12 | 5ED | 0006 | 000 | 0012 | 0006 | 0000 | 202 | 05EE |
| 1F0 | 2FFF | 1F1 | 2FFF | 1F0 | FFFF | 000 | FFFF | 0006 | 0000 |  |  |
| 1F1 | DE13 | 205 | DE13 | 7FF | 01F2 | 7FF | 0205 | 0006 | 0000 | 7FF | 01F2 |
| 205 | EE06 | 206 | EE06 | 20C | 0006 | 7FF | 0006 | 0006 | 0000 | 20C | 0006 |
| 206 | 1203 | 207 | 1203 | 206 | 1203 | 7FF | 0206 | 0040 | 0000 |  |  |
| 207 | 2F40 | 208 | 2F40 | 207 | 0040 | 7FF | 0040 | 0040 | 0000 |  |  |
| 208 | F0FD | 209 | F0FD | 208 | F0FD | 7FF | 0208 | 0040 | 0000 |  |  |
| 209 | AE02 | 20A | AE02 | 20C | 0006 | 7FF | 0002 | 0006 | 0000 |  |  |
| 20A | 1302 | 20B | 1302 | 20A | 1302 | 7FF | 020A | 0006 | 0000 |  |  |
| 20B | 0A00 | 1F2 | 0A00 | 7FF | 01F2 | 000 | 020B | 0006 | 0000 |  |  |
| 1F2 | F00D | 1F3 | F00D | 1F2 | F00D | 000 | 01F2 | 0006 | 0000 |  |  |
| 1F3 | EE0F | 1F4 | EE0F | 203 | 0006 | 000 | 000F | 0006 | 0000 | 203 | 0006 |
| 1F4 | AA0D | 1F5 | AA0D | 5EE | E9EC | 000 | 000D | E9EC | 1000 | 202 | 05EF |
| 1F5 | DE0F | 205 | DE0F | 7FF | 01F6 | 7FF | 0205 | E9EC | 1000 | 7FF | 01F6 |
| 205 | EE06 | 206 | EE06 | 20C | E9EC | 7FF | 0006 | E9EC | 1000 | 20C | E9EC |
| 206 | 1203 | 207 | 1203 | 206 | 1203 | 7FF | 0206 | E940 | 1000 |  |  |
| 207 | 2F40 | 208 | 2F40 | 207 | 0040 | 7FF | 0040 | 0040 | 0000 |  |  |
| 208 | F0FD | 209 | F0FD | 208 | F0FD | 7FF | 0208 | 0040 | 0000 |  |  |
| 209 | AE02 | 20A | AE02 | 20C | E9EC | 7FF | 0002 | E9EC | 1000 |  |  |
| 20A | 1302 | 20B | 1302 | 20A | 1302 | 7FF | 020A | E9EC | 1000 |  |  |
| 20B | 0A00 | 1F6 | 0A00 | 7FF | 01F6 | 000 | 020B | E9EC | 1000 |  |  |
| 1F6 | 0680 | 1F7 | 0680 | 1F6 | 0680 | 000 | 01F6 | ECE9 | 1000 |  |  |
| 1F7 | EE0C | 1F8 | EE0C | 204 | ECE9 | 000 | 000C | ECE9 | 1000 | 204 | ECE9 |
| 1F8 | AE0A | 1F9 | AE0A | 203 | 0006 | 000 | 000A | 0006 | 0000 |  |  |
| 1F9 | 0740 | 1FA | 0740 | 1F9 | 0740 | 000 | 01F9 | 0005 | 0001 |  |  |
| 1FA | F005 | 1FB | F005 | 1FA | F005 | 000 | 01FA | 0005 | 0001 |  |  |
| 1FB | EE07 | 1FC | EE07 | 203 | 0005 | 000 | 0007 | 0005 | 0001 | 203 | 0005 |
| 1FC | AE07 | 1FD | AE07 | 204 | ECE9 | 000 | 0007 | ECE9 | 1001 |  |  |
| 1FD | DE07 | 205 | DE07 | 7FF | 01FE | 7FF | 0205 | ECE9 | 1001 | 7FF | 01FE |
| 205 | EE06 | 206 | EE06 | 20C | ECE9 | 7FF | 0006 | ECE9 | 1001 | 20C | ECE9 |
| 206 | 1203 | 207 | 1203 | 206 | 1203 | 7FF | 0206 | EC40 | 1001 |  |  |
| 207 | 2F40 | 208 | 2F40 | 207 | 0040 | 7FF | 0040 | 0040 | 0001 |  |  |
| 208 | F0FD | 209 | F0FD | 208 | F0FD | 7FF | 0208 | 0040 | 0001 |  |  |
| 209 | AE02 | 20A | AE02 | 20C | ECE9 | 7FF | 0002 | ECE9 | 1001 |  |  |
| 20A | 1302 | 20B | 1302 | 20A | 1302 | 7FF | 020A | ECE9 | 1001 |  |  |
| 20B | 0A00 | 1FE | 0A00 | 7FF | 01FE | 000 | 020B | ECE9 | 1001 |  |  |
| 1FE | 8E04 | 1FF | 8E04 | 203 | 0004 | 000 | 0003 | ECE9 | 1001 | 203 | 0004 |
| 1FF | CEF4 | 1F4 | CEF4 | 1FF | 01F4 | 000 | FFF4 | ECE9 | 1001 |  |  |

Вывод

В процессе выполнения работы я познакомился с реализацией ввода-вывода в БЭВМ, принципом асинхронного обмена с ВУ по готовности. Синтаксисом ассемблера БЭВМ.

ДОП

Написать программу, которая реализует ввод кол-ва символов с ВУ-3, далее ввод с клавиатуры, вывод на текстовый принтер в обратном порядке, но только нечетные символы. Все остальное согласно заданию.

ORG 0x5ED ; Строка

WORD ? ; Колво символов

ORG 0x1EC ; Основная программа

START: CLA

LD STR\_ADDR ; Загрузка адреса строки

ST ADDR ; Копирование адреса строки

CLA

CALL READ\_DEV3 ; Чтение длины слова

BEQ EXIT ; Завершение если 0

ST LEN ; Сохранение длины

ST (ADDR)+ ; Сохранение в памяти

INPUT\_LOOP:

CALL READ\_DEV8

ST TMP ; Промежуточно сохраняем аккумулятор

LD LEN

DEC ; Отнимаем счётчик символов

ST LEN

BEQ SAVE ; Сохраняем если всё введено

LD TMP ; Обратно загружаем АС

SWAB

CALL READ\_DEV8 ; Чтение старшего символа

SWAB ; Разворот чтобы был правильный порядок

ST (ADDR)+ ; Сохранение в память

JUMP LP

SAVE: LD TMP

ST (ADDR)+

LP: LOOP LEN

JUMP INPUT\_LOOP

LD (STR\_ADDR) ; Получение длины слова

AND #0xFF ; Очистка старших байт

BEQ EXIT ; Завершение если 0

ST LEN ; Сохранение длины

OUT\_LOOP:

LD -(ADDR) ; Начало цикла вывода

CALL PRINT\_DEV5 ; Вывод младшего символа

SWAB ; Обмен байтами

ST TMP ; Промежуточно сохраняем аккумулятор

LD LEN

DEC ; Отнимаем счётчик символов

BEQ EXIT ; Выход если всё напечатано

ST LEN

;LD TMP ; Обратно загружаем АС

;CALL PRINT\_DEV5 ; Вывод старшего символа

LOOP LEN

JUMP OUT\_LOOP

EXIT: HLT

STR\_ADDR: WORD 0x5ED ; Адрес строки в памяти

ADDR: WORD ? ; Переменная для прохода по массиву

LEN: WORD ? ; Длина строки

TMP: WORD ? ; Переменная для промежуточного хранения

PRINT\_DEV5: ; Подпрограмма печатает содержимое младшего байта аккумулятора на ВУ5

ST TMP\_AC5

CHECK5: IN 0xD

AND #0x40

BEQ CHECK5

LD TMP\_AC5

OUT 0xC

RET

TMP\_AC5: WORD ? ; Переменная промежуточного хранения АС

READ\_DEV3: ; Подпрограмма читает с ВУ3 в аккумулятор

ST TMP\_AC3

CHECK3: IN 0x7

AND #0x40

BEQ CHECK3

LD TMP\_AC3

IN 0x6

CMP 0x0

RET

TMP\_AC3: WORD ? ; Переменная промежуточного хранения АС

READ\_DEV8: ; Подпрограмма читает с ВУ8 в аккумулятор

ST TMP\_AC8

CHECK8: IN 0x19

AND #0x40

BEQ CHECK8

LD TMP\_AC8

IN 0x18

CMP 0x0

RET

TMP\_AC8: WORD ? ; Переменная промежуточного хранения АС