Отчет

**по лабораторной работе №5**

**«Асинхронный обмен данными с ВУ»**

по дисциплине «Основы профессиональной деятельности» вариант 10673

Выполнил: Караганов П.Э., группа P3110

Преподаватель: Блохина Е.Н.

Оглавление

[Текст задания 3](#_Toc196350129)

[Текст исходной программы 3](#_Toc196350130)

[Текст программы на ассемблере: 3](#_Toc196350131)

[Текст программы в таблице 3](#_Toc196350132)

[Описание программы 4](#_Toc196350133)

[Назначение программы и реализуемая ею функция 4](#_Toc196350134)

[Описание исходных данных 4](#_Toc196350135)

[Область представления 4](#_Toc196350136)

[Область допустимых значений 4](#_Toc196350137)

[Расположение в памяти БЭВМ 4](#_Toc196350138)

[Таблица трассировки 4](#_Toc196350139)

[Передаваемое сообщение: «Ноябрь?» 4](#_Toc196350140)

[Трассировка 4](#_Toc196350141)

[Вывод 4](#_Toc196350142)

# Текст задания

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.

1. Программа осуществляет асинхронный вывод данных на ВУ-3
2. Программа начинается с адреса 1F316. Размещаемая строка находится по адресу 5C316.
3. Строка должна быть представлена в кодировке Windows-1251.
4. Формат представления строки в памяти: АДР1: СИМВ2 СИМВ1 АДР2: СИМВ4 СИМВ3 ... СТОП\_СИМВ.
5. Ввод или вывод строки должен быть завершен по символу c кодом 0A (NL). Стоп символ является обычным символом строки и подчиняется тем же правилам расположения в памяти что и другие символы строки.

# Текст исходной программы

## Текст программы на ассемблере:

ORG 0x1F3 ; Начинаем программу

res: WORD 0x5C3 ; Адрес результата

finish: WORD 0x0A ; 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

## Текст программы в таблице

|  |  |  |  |
| --- | --- | --- | --- |
| **Адрес** | **Код команды** | **Мнемоника** | **Комментарии** |
| 1F6 | 0200 | CLA | Очистка аккумулятора  0 → AC |
| 1F7 | 1207 | IN 7 | Чтение регистра состояния ВУ-3 |
| 1F8 | 2F40 | AND #40 | Зануляем все биты, кроме 6-ого |
| 1F9 | F0FD | BEQ IP-3 | Если AC=0, то ВУ-3 не готово к обмену  Запускаем цикл |
| 1FA | 1206 | IN 6 | Чтение регистра данных ВУ-3 |
| 1FB | E8F7 | ST (IP-9) | Сохраняем символ в результат |
| 1FC | 7EF7 | CMP IP-9 | Проверяем на стоп-слово |
| 1FD | F00F | BEQ IP+15 | Выходим если AC равен стоп-слову |
| **1FE** | **0200** | CLA | Очистка аккумулятора  0 → AC |
| **1FF** | **1207** | IN 7 | Чтение регистра состояния ВУ-3 |
| **200** | **2F40** | AND #40 | Зануляем все биты, кроме 6-ого |
| **201** | **F0FC** | BEQ IP-4 | Если AC=0, то ВУ-3 не готово к обмену  Запускаем цикл |
| **202** | **1206** | IN 6 | Чтение регистра данных ВУ-3 |
| **203** | **EEF1** | ST IP-15 | Загружаем значение чётного байта во временную ячейку |
| **204** | **0680** | SWAB | Меняем младший и старший байты местами в AC |
| **205** | **38ED** | OR (IP-19) | Соединяем с уже имеющимся результатом |
| **206** | **E8EC** | ST (IP-20) | Загружаем в результат значение AC |
| **207** | **AEED** | LD IP-19 | Загружаем в AC чётный символ, который нам ввели |
| **208** | **7EEB** | CMP IP-21 | Проверяем на стоп-слово |
| **209** | **F003** | BEQ IP+3 | Выходим если AC равен стоп-слову |
| **20A** | **AAE8** | LD (IP-24)+ | Просто увеличиваем ссылку-результат на 1 |
| **20B** | **0200** | CLA | Очистка аккумулятора  0 → AC |
| **20C** | **CEEA** | JUMP IP-22 | Возвращаемся к вводу нового символа |
| **20D** | **AAE5** | LD (IP-27)+ | Просто увеличиваем ссылку-результат на 1 |
| **20E** | **0100** | HLT | Остановка |

# Описание программы

## Назначение программы и реализуемая ею функция

Программа осуществляет посимвольный асинхронный ввод данных с ВУ-3, посимвольно записывает их в память. Программа будет получать символы до тех пор, пока на ВУ-3 не будет введен стоп-символ с кодировкой 0x0A, который она запишет в память и прекратит свое выполнение

## Описание исходных данных

* 1F3(res) – ссылка на результат
* 1F4(finish) – стоп-слово (0x0A)
* 1F5(temp) – ячейка для временного хранения данных

## Область представления

* res – 11-разрядная ячейка
* finish – 16-разрядная константа
* temp - 16-разрядная ячейка для временного хранения введенных символов
* [5C3 – 7FF] - 16-разрядные ячейки, хранящие в себе по два символа в кодировке Windows-1251

## Область допустимых значений

* res ∈ [00016; 1F216] ∪ [5C316; 7FF16]
* temp ∈ [0016; FF16]
* Введённый символ ∈ [0016; FF16]
* 5С316 ≤ STRING ≤ 5C316 + ⌊ (Длина строки) \ 2 ⌋, где (Длина строки) / 2 ≤ 7FF16 – 5C316, **иначе** STRING ∈ [00016; ⌊ (Длина строки) \ 2 ⌋ - 7FF16 – 5C316] ∪ [5C316;7FF16], где (Длина строки) ≤ 86016 (214410)
* Длина строки ∈ [0; 2144] c учётом стоп-слова (|[5C316; 7FF16] ∪ [00016; 1F216]| = 573 + 499 = 1072 ячейки памяти => макс 2144 символов с учётом стоп-слова)

## Расположение в памяти БЭВМ

* *Программы:* Ячейки (1F6) – (20E)
* *Данные для программы:* Ячейки (1F3) – (1F5)
* *Адрес первой выполняемой команды:* (1F6)
* *Адрес последней выполняемой команды:* (20E)

# Таблица трассировки

## **Передаваемое сообщение**: «Ноябрь?»

В кодировке Windows-1251: CD EE FF E1 F0 FC 3F

В кодировке UTF-8: D0 9D D0 BE D1 8F D0 B1 D1 80 D1 8C 3F

В кодировке UTF-16(LE): 1D 04 3E 04 4F 04 31 04 40 04 3C 04 3F 00

## Трассировка

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Выполненная Команда** | | **Содержание аккумуляторов процессора после выполнения команды** | | | | | | | | **Ячейка, содержимое которой изменилось после выполнения команды** | |
| Адрес | Код | IP | CR | AR | DR | SP | BR | AC | NZVC | Адрес | Новый код |
| 1F6 | 0200 | 1F7 | 0200 | 1F6 | 0200 | 000 | 01F6 | 0000 | 0100 |  |  |
| 1F7 | 1207 | 1F8 | 1207 | 1F7 | 1207 | 000 | 01F7 | 0040 | 0100 |  |  |
| 1F8 | 2F40 | 1F9 | 2F40 | 1F8 | 0040 | 000 | 0040 | 0040 | 0000 |  |  |
| 1F9 | F0FD | 1FA | F0FD | 1F9 | F0FD | 000 | 01F9 | 0040 | 0000 |  |  |
| 1FA | 1206 | 1FB | 1206 | 1FA | 1206 | 000 | 01FA | 00CD | 0000 |  |  |
| 1FB | E8F7 | 1FC | E8F7 | 5C3 | 00CD | 000 | FFF7 | 00CD | 0000 | 5C3 | 00CD |
| 1FC | 7EF7 | 1FD | 7EF7 | 1F4 | 000A | 000 | FFF7 | 00CD | 0001 |  |  |
| 1FD | F00F | 1FE | F00F | 1FD | F00F | 000 | 01FD | 00CD | 0001 |  |  |
| 1FE | 0200 | 1FF | 0200 | 1FE | 0200 | 000 | 01FE | 0000 | 0101 |  |  |
| 1FF | 1207 | 200 | 1207 | 1FF | 1207 | 000 | 01FF | 0040 | 0101 |  |  |
| 200 | 2F40 | 201 | 2F40 | 200 | 0040 | 000 | 0040 | 0040 | 0001 |  |  |
| 201 | F0FC | 202 | F0FC | 201 | F0FC | 000 | 0201 | 0040 | 0001 |  |  |
| 202 | 1206 | 203 | 1206 | 202 | 1206 | 000 | 0202 | 00EE | 0001 |  |  |
| 203 | EEF1 | 204 | EEF1 | 1F5 | 00EE | 000 | FFF1 | 00EE | 0001 | 1F5 | 00EE |
| 204 | 0680 | 205 | 0680 | 204 | 0680 | 000 | 0204 | EE00 | 1001 |  |  |
| 205 | 38ED | 206 | 38ED | 5C3 | 00CD | 000 | 1132 | EECD | 1001 |  |  |
| 206 | E8EC | 207 | E8EC | 5C3 | EECD | 000 | FFEC | EECD | 1001 | 5C3 | EECD |
| 207 | AEED | 208 | AEED | 1F5 | 00EE | 000 | FFED | 00EE | 0001 |  |  |
| 208 | 7EEB | 209 | 7EEB | 1F4 | 000A | 000 | FFEB | 00EE | 0001 |  |  |
| 209 | F003 | 20A | F003 | 209 | F003 | 000 | 0209 | 00EE | 0001 |  |  |
| 20A | AAE8 | 20B | AAE8 | 5C3 | EECD | 000 | FFE8 | EECD | 1001 | 1F3 | 05C4 |
| 20B | 0200 | 20C | 0200 | 20B | 0200 | 000 | 020B | 0000 | 0101 |  |  |
| 20C | CEEA | 1F7 | CEEA | 20C | 01F7 | 000 | FFEA | 0000 | 0101 |  |  |

РАЗОБРАТЬСЯ КАК РАБОТЕТ:

* *IN*: DR→BR; IP→DP; BR→IP; ~0 + SP→SP, **AR**; DR→MEM(AR);

# Вывод

Во время выполнения лабораторной работы я закрепил знания по тому, как происходит асинхронный программно-управляемый обмен данными с ВУ, как в БЭВМ храниться и обрабатываются символьные строковые данные, как работают команды ввода-вывода. Я уверен мне это пригодиться в будущем