Федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Лабораторная работа по основам профессиональной деятельности №5
Вариант №6505

Выполнил:
Студент группы P3106
Мельник Фёдор Александрович
Проверил:
Ткешелашвили Нино Мерабиевна,
Преподаватель-практик ФПИиКТ

Санкт-Петербург, 2025

Оглавление

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.

Введите номер варианта 6505

1. Программа осуществляет асинхронный ввод данных с ВУ-2
2. Программа начинается с адреса $54D_{16}$. Размещаемая строка находится по адресу $57E_{16}$.
3. Строка должна быть представлена в кодировке Windows-1251.
4. Формат представления строки в памяти: АДР0: ДЛИНА АДР1: СИМВ1 СИМВ2 АДР2: СИМВ3 СИМВ4 ..., где ДЛИНА - 16 разрядное слово, где значащими являются 8 младших бит.
5. Ввод строки начинается со ввода количества символов (1 байт), и должен быть завершен по вводу их необходимого количества.

Код на ассемблере

```
ORG    0x530;
RES:                WORD 0x57F;         РЕЗУЛЬТАТ
CUR_LENGTH:         WORD 0x0;           ТЕКУЩАЯ ДЛИНА СТРОКИ
TEMP:               WORD ?;             ВРЕМЕННАЯ ПЕРЕМЕННАЯ

ORG    0x54D;
START:              CLA;                ОЧИСТКА

LD_LENGTH:          IN 5;
                    AND #0x40;
                    BEQ LD_LENGTH;      SPIN-LOOP ПОКА НЕЛЬЗЯ СЧИТАТЬ

                    IN 4;               СЧИТЫВАНИЕ
                    ST LENGTH;          ЗАПОЛНЕНИЕ ДЛИНЫ
                    CLA;                ОЧИСТКА

FIRST:              IN 5;
                    AND #0x40;
                    BEQ FIRST;          SPIN-LOOP ПОКА НЕЛЬЗЯ СЧИТАТЬ

                    IN 4;               СЧИТЫВАНИЕ

                    ST TEMP;            СОХРАНЕНИЕ ВО ВРЕМЕННУЮ ПЕРМЕННУЮ
                    LD CUR_LENGTH;
                    INC;                УВЕЛИЧЕНИЕ ТЕКУЩЕЙ ДЛИНЫ
                    ST CUR_LENGTH;
```

```
                LD TEMP;            ЗАГРУЗКА ИЗ ВРЕМЕННОЙ ПЕРМЕННОЙ

                SWAB;               ПЕРЕНОС ИЗ МЛАДШЕГО В СТАРШИЙ БАЙТ
                ST (RES);           СОХРАНЕНИЕ (В МЛАДШИЙ БИТ)


                LD LENGTH;
                CMP CUR_LENGTH;     ПРОВЕРКА ДЛИНА = ТЕКУЩАЯ ДЛИНА
                BEQ STOP;           ВЫХОД


SECOND:         IN 5;
                AND #0x40;
                BEQ SECOND;         SPIN-LOOP ПОКА НЕЛЬЗЯ СЧИТАТЬ

                IN 4;               СЧИТЫВАНИЕ

                ST TEMP;            СОХРАНЕНИЕ ВО ВРЕМЕННУЮ ПЕРМЕННУЮ
                LD CUR_LENGTH;
                INC;                УВЕЛИЧЕНИЕ ТЕКУЩЕЙ ДЛИНЫ
                ST CUR_LENGTH;
                LD TEMP;            ЗАГРУЗКА ИЗ ВРЕМЕННОЙ ПЕРМЕННОЙ

                ADD (RES);          ОБЪЕДИНЕНИЕ ДВУХ СИМВОЛОВ
                ST (RES)+;          СОХРАНЕНИЕ И УВЕЛИЧЕНИЕ АДРЕССА

                LD LENGTH;
                CMP CUR_LENGTH;     ПРОВЕРКА ДЛИНА = ТЕКУЩАЯ ДЛИНА
                BEQ STOP;           ВЫХОД
                JUMP FIRST;   ЦИКЛ

STOP:           HLT;                ВЫХОД
ORG 0x57E;
LENGTH:         WORD ?;             ДЛИНА СТРОКИ
```

Текст исходной программы

Адрес	Команда	Мнемоника	Комментарий
530	57E	RES	Адрес результата
531	0000	LENGTH	Длина строка
532	0000	CUR_LENGTH	Текущая длина строки
533	0000	TEMP	Временная переменная
…	…	…	…

54D	0200	CLA	Очистка
54E	1205	IN #5	Spin-loop в ожидании ввода
54F	2F40	AND #0x40	
550	F0FD	BEQ IP-3	
551	1204	IN #4	Считывание
552	EEDE	ST LENGHTH	Запись длины строки
553	0200	CLA	Очистка
554	1205	IN #5	Spin-loop в ожидании ввода
555	2F40	AND #0x40	
556	F0FD	BEQ IP-3	
557	1204	IN #4	Считывание
558	EECD	ST TEMP	Запись во временную переменную
559	AECB	LD CUR_LENGTH	Увеличение текущей длины на 1
55A	0700	INC	
55B	EEC9	ST CUR_LENGTH	
55C	AEC9	LD TEMP	Возвращение из временной переменной
55D	0680	SWAB	Сохранение по адресу результата
55E	48C4	ADD (RES)	Сравнение длины строки и текущей длины строки Переход в случае равенства
55F	EAC3	ST (RES)+	
560	AEC3	LD LENGTH	
561	7EC3	CMP CUR_LENGTH	Spin-loop в ожидании ввода
562	F001	BEQ IP+13	
563	1205	IN #5	
564	2F40	AND #0x40	Считывание
565	F0FD	BEQ IP-3	Запись во временную переменную
566	1204	IN #4	Увеличение текущей длины на 1
567	EEDA	ST TEMP	
568	AED8	LD CUR_LENGTH	
569	0700	INC	Возвращение из временной переменной
56A	EED6	ST CUR_LENGTH	Перенос из младшего байта в старший
56B	AED6	LD TEMP	Объединение двух символов
56C	E8D2	ST (RES)	Сохранение и увеличение адреса
56D	AED2	LD LENGTH	Сравнение длины строки и текущей длины строки Переход в случае равенства
56E	7ED2	CMP CUR_LENGTH	
56F	F010	BEQ IP+11	
570	CEE3	JUMP IP-29	Цикл
571	0100	HLT	Завершение программы
…	…	…	…
57E	-	-	Начало результата

Описание программы

Программа осуществляет асинхронный ввод данных с ВУ-2. Программа получает символы, пока их количество не становится равно первому введенному значению (длине строки)

Область представления

Результат – 16ти разрядные ячейки, хранящие в себе по 2 символа в кодировке Windows-1251
RES – 11 разрядная ячейка, хранящая адрес текущей ячейки с результатом

LENGTH – 8 разрядная ячейка, содержащая длину передаваемой строки
CUR_LENGTH – 8 разрядная ячейка, содержащая текущую длину переданной строки
TEMP – 16ти разрядная ячейка, для временного хранения переданных символов

Область допустимых значений

LENGTH $\in [0;2^8\text{-}1]$

0 <= RES <= 530 – ⌈LENGTH/2⌉
534 <= RES <= 54D – ⌈LENGTH/2⌉
57E <= RES <= 7FF – ⌈LENGTH/2⌉

Трассировка

Слово для ввода - ЗАЯЦ
В кодировке Windows-1251:
З = 0xC7
А = 0xC0
Я = 0xDF
Ц = 0xD6

UTF-8: D0 97 D0 90 D0 AF D0 A6
UTF-16: 17 04 10 04 2F 04 26 04

Адрес Код IP CR AR DR SP BR AC NZVC Адрес Новый код
54D 0200 54E 0200 54D 0200 000 054D 0000 0101
54E 1205 54F 1205 54E 1205 000 054E 0040 0101
54F 2F40 550 2F40 54F 0040 000 0040 0040 0001
550 F0FD 551 F0FD 550 F0FD 000 0550 0040 0001
551 1204 552 1204 551 1204 000 0551 0002 0001
552 EEDE 553 EEDE 531 0002 000 FFDE 0002 0001 531 0002
553 0200 554 0200 553 0200 000 0553 0000 0101
554 1205 555 1205 554 1205 000 0554 0000 0101
555 2F40 556 2F40 555 0040 000 0040 0000 0101
556 F0FD 554 F0FD 556 F0FD 000 FFFD 0000 0101
554 1205 555 1205 554 1205 000 0554 0000 0101
555 2F40 556 2F40 555 0040 000 0040 0000 0101
556 F0FD 554 F0FD 556 F0FD 000 FFFD 0000 0101
554 1205 555 1205 554 1205 000 0554 0040 0101
555 2F40 556 2F40 555 0040 000 0040 0040 0001
556 F0FD 557 F0FD 556 F0FD 000 0556 0040 0001
557 1204 558 1204 557 1204 000 0557 00C7 0001
558 EEDA 559 EEDA 533 00C7 000 FFDA 00C7 0001 533 00C7
559 AED8 55A AED8 532 0000 000 FFD8 0000 0101
55A 0700 55B 0700 55A 0700 000 055A 0001 0000
55B EED6 55C EED6 532 0001 000 FFD6 0001 0000 532 0001
55C AED6 55D AED6 533 00C7 000 FFD6 00C7 0000
55D 0680 55E 0680 55D 0680 000 055D C700 1000
55E E8D1 55F E8D1 57E C700 000 FFD1 C700 1000 57E C700
55F AED1 560 AED1 531 0002 000 FFD1 0002 0000
560 7ED1 561 7ED1 532 0001 000 FFD1 0002 0001

561 F00F 562 F00F 561 F00F 000 0561 0002 0001
562 1205 563 1205 562 1205 000 0562 0040 0001
563 2F40 564 2F40 563 0040 000 0040 0040 0001
564 F0FD 565 F0FD 564 F0FD 000 0564 0040 0001
565 1204 566 1204 565 1204 000 0565 00C0 0001
566 EECC 567 EECC 533 00C0 000 FFCC 00C0 0001 533 00C0
567 AECA 568 AECA 532 0001 000 FFCA 0001 0001
568 0700 569 0700 568 0700 000 0568 0002 0000
569 EEC8 56A EEC8 532 0002 000 FFC8 0002 0000 532 0002
56A AEC8 56B AEC8 533 00C0 000 FFC8 00C0 0000
56B 48C4 56C 48C4 57E C700 000 FFC4 C7C0 1000
56C EAC3 56D EAC3 57E C7C0 000 FFC3 C7C0 1000 530 057F
57E C7C0
56D AEC3 56E AEC3 531 0002 000 FFC3 0002 0000
56E 7EC3 56F 7EC3 532 0002 000 FFC3 0002 0101
56F F001 571 F001 56F F001 000 0001 0002 0101
571 0100 572 0100 571 0100 000 0571 0002 0101

Доп

С ВУ-3 вводится 16-разрядное число (в два захода, сначала старшая часть, затем младшая). Интерпретируя это число, как количество секунд, вывести на ВУ-6 (бегущая строка) строку в формате "1:30:12", где три числа это часы, минуты и секунды соответственно.

```
ORG 0x10
JUMP START;
NUMBER:     WORD ?;          ИСХОДНОЕ ЧИСЛО
HOURS:      WORD ?;                  ЧАСЫ
MINUTES:    WORD ?;          МИНУТЫ
SECONDS:    WORD ?;          СЕКУНДЫ
HOUR:       WORD 0x0E10;     КОЛ-ВО СЕКУНД В ЧАСЕ
MINUTE:     WORD 0x003C;     КОЛ-ВО СЕКУНД В МИНУТЕ

START:  CLA;
    JUMP INPUT_FIRST;

INPUT_FIRST:   IN 7;
        AND #0x40;
        BEQ INPUT_FIRST;

        IN 6;
        SWAB;
        ST NUMBER;
        CLA;

INPUT_SECOND: IN 7;
        AND #0x40;
        BEQ INPUT_SECOND;

        LD NUMBER;
        IN 6;
```

```
        ST NUMBER;
        CLA;

COUNT_H:    LD NUMBER;
        CMP HOUR;
        BEQ COUNT_H1;
        BLO COUNT_M;

COUNT_H1:   SUB HOUR;
        ST NUMBER;
        LD HOURS;
        INC
        ST HOURS;
        JUMP COUNT_H;

COUNT_M:    LD NUMBER;
        CMP MINUTE;
        BEQ COUNT_M1;
        BLO COUNT_S;

COUNT_M1:   SUB MINUTE;
        ST NUMBER;
        LD MINUTES;
        INC;
        ST MINUTES;
        JUMP COUNT_M;

COUNT_S:    LD NUMBER;
        ST SECONDS;

PRINT_H:  LD HOURS;
    PUSH;
    CALL $PRINT_NUMBER;
    CALL PRINT_COLON;

PRINT_M:    LD MINUTES;
        PUSH;
        CALL $PRINT_NUMBER;
        CALL PRINT_COLON;

PRINT_S:   LD SECONDS;
    PUSH;
    CALL $PRINT_NUMBER;

STOP: HLT

PRINT_COLON: WORD 0xAF24;
    OUT 0x10;
    CLA;
    OUT 0x10;
    RET;

PRINT_NUM: WORD 0x7F00;
```

```
   BEQ PRINT_ZERO;
   WORD 0x7F01;
   BEQ PRINT_ONE;
   WORD 0x7F02;
   BEQ PRINT_TWO;
   WORD 0x7F03;
   BEQ PRINT_THREE;
   WORD 0x7F04;
   BEQ PRINT_FOUR;
   WORD 0x7F05;
   BEQ PRINT_FIVE;
   WORD 0x7F06;
   BEQ PRINT_SIX;
   WORD 0x7F07;
   BEQ PRINT_SEVEN;
   WORD 0x7F08;
   BEQ PRINT_EIGHT;
   WORD 0x7F09;
   BEQ PRINT_NINE;
   RET

PRINT_ZERO:    WORD 0xAF7E;
        OUT 0x10;
        WORD 0xAF81;
        OUT 0x10;
        OUT 0x10;
        OUT 0x10;
        WORD 0xAF7E;
        OUT 0x10;
        CLA;
        OUT 0x10;
        RET;

PRINT_ONE:    WORD 0xAF21;
        OUT 0x10;
        WORD 0xAF41;
        OUT 0x10;
        WORD 0xAFFF;
        OUT 0x10;
        WORD 0xAF01;
        OUT 0x10;
        OUT 0x10;
        CLA;
        OUT 0x10;
        RET;

PRINT_TWO:    WORD 0xAF61;
        OUT 0x10;
        WORD 0xAF83;
        OUT 0x10;
        WORD 0xAF85;
        OUT 0x10;
        WORD 0xAF89;
```

```
        OUT 0x10;
        WORD 0xAF71;
        OUT 0x10;
        CLA;
        OUT 0x10;
        RET;

PRINT_THREE:  WORD 0xAF62;
      OUT 0x10;
      WORD 0xAF81;
      OUT 0x10;
      WORD 0xAF81;
      OUT 0x10;
      WORD 0xAF91;
      OUT 0x10;
      WORD 0xAF6E;
      OUT 0x10;
      CLA;
      OUT 0x10;
      RET;

PRINT_FOUR:  WORD 0xAFF0;
      OUT 0x10;
      WORD 0xAF10;
      OUT 0x10;
      WORD 0xAF10;
      OUT 0x10;
      WORD 0xAF10;
      OUT 0x10;
      WORD 0xAFFF;
      OUT 0x10;
      CLA;
      OUT 0x10;
      RET;

PRINT_FIVE:  WORD 0xAFFA;
      OUT 0x10;
      WORD 0xAF91;
      OUT 0x10;
      WORD 0xAF91;
      OUT 0x10;
      WORD 0xAF91;
      OUT 0x10;
      WORD 0xAF8E;
      OUT 0x10;
      CLA;
      OUT 0x10;
      RET;

PRINT_SIX:    WORD 0xAF7E;
      OUT 0x10;
      WORD 0xAF91;
      OUT 0x10;
```

```
      WORD 0xAF91;
      OUT 0x10;
      WORD 0xAF91;
      OUT 0x10;
      WORD 0xAF4E;
      OUT 0x10;
      CLA;
      OUT 0x10;
    RET;

PRINT_SEVEN:  WORD 0xAF80;
      OUT 0x10;
      WORD 0xAF83;
      OUT 0x10;
      WORD 0xAF8C;
      OUT 0x10;
      WORD 0xAF90;
      OUT 0x10;
      WORD 0xAFE0;
      OUT 0x10;
      CLA;
      OUT 0x10;
      RET;



PRINT_EIGHT: WORD 0xAF6E;
    OUT 0x10;
    WORD 0xAF91;
    OUT 0x10;
    WORD 0xAF91;
    OUT 0x10;
    WORD 0xAF91;
    OUT 0x10;
    WORD 0xAF6E;
    OUT 0x10;
    CLA;
    OUT 0x10;
    RET;

PRINT_NINE: WORD 0xAF62;
    OUT 0x10;
    WORD 0xAF91;
    OUT 0x10;
    WORD 0xAF91;
    OUT 0x10;
    WORD 0xAF91;
    OUT 0x10;
    WORD 0xAF7E;
    OUT 0x10;
    CLA;
    OUT 0x10;
    RET;
```

```
TEMP_T:     WORD ?;          ВРЕМЕННАЯ ПЕРЕМЕННАЯ ДЛЯ ХРАНЕНИЯ ДЕСЯТОК ЧИСЛА

PRINT_NUMBER: CLA;
   ST TEMP_T;
   JUMP COUNT_TENS;

COUNT_TENS:  WORD 0xAC01;
        WORD 0x7F0A;
        BEQ COUNT_TENS1;
        BLO COUNT_TENS_EXIT;

COUNT_TENS1:  WORD 0x6F0A;
        WORD 0xEC01;
        LD TEMP_T;
        INC;
        ST TEMP_T;
        JUMP COUNT_TENS;

COUNT_TENS_EXIT:  LD TEMP_T;
    PUSH;
    CALL $PRINT_NUM;
    POP;
    WORD 0xAC01;
    PUSH;
    CALL $PRINT_NUM;
    POP;
    RET;
```

Заключение

В ходе выполнения лабораторной работы я познакомился с асинхронным вводом и выводом данных с помощью внешних устройств. Поработал с данными в различных кодировках и получил первый опыт написания кода на Assembler