**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №4**

**по дисциплине «Организация ЭВМ и систем»**

Тема: **Представление и обработка символьной информации с использованием строковых команд.**

|  |  |  |
| --- | --- | --- |
| Студент гр. 1303 |  | Попандопуло А. Г. |
| Преподаватель |  | Ефремов М.А. |

Санкт-Петербург

2022

## **Цель работы.**

## Разработка программы программы на ЯВУ, с включенной по принципу встраивания (in-line) Ассемблерной частью. Таким образом изучить представление и обработку символьной информации с помощью строковых команд.

**Задание.**

Разработать программу обработки символьной информации, реализующую функции:

* инициализация (вывод титульной таблички с указанием вида преобразования и автора программы) - на ЯВУ;
* ввода строки символов, длиной не более Nmax (<=80), с клавиатуры в заданную область памяти - на ЯВУ; если длина строки превышает Nmax, остальные символы следует игнорировать;
* выполнение заданного в таблице 5 преобразования исходной строки с записью результата в выходную строку - на Ассемблере;
* вывода результирующей строки символов на экран и ее запись в файл - на ЯВУ.

Ассемблерную часть программы включить в программу нa ЯВУ по принципу встраивания (in-line).

**Выполнение работы**

В соответствие с поставленной задачей реализован программный код на языке C++, в который по принципу встраивания была включена часть код на языке Ассемблер.

Считывание входной строки выполняется с помощью функции fgets(); запись происходит в символьный массив input\_string, размер которого, по условию, ограничен 80-ю символами. Примечательно, что предусмотренный для выходной строки символьный массив необходимо взять с запасом ввиду представления в двоичной СС.

Ассемблерная часть начинается после ключевого слова \_\_asm. Значение ds присваивается регистру es, чтение строк происходит по адресу es:esi, запись – es:edi. Смещение до входной строки присваивается esi, выходной соответственно edi.

Команда lodsb используется для чтения байта из строки: копирует байт памяти по адресу es:esi в al регистр. Далее полученный символ последовательно проверяется с помощью команды cmp, выполняющей сравнение с всевозможными цифрами шестнадцатиричной СС. Для каждого случая предусмотрена соответствующая замена.

Для преобразования цифр «2» и «3» в регистр ax помещается двоичная запись в обратном порядке, далее отправляется в выходной массив с помощью slosw – осуществляет запись в строку слова, увеличивая регистр di на 2.

Для преобразования цифр «4», «5», «6», «7» первые два символа двоичного представления из трех аналогично в обратном порядке записываются в ax, отправляются в выходной массив, для записи оставшегося символа используется младший регистр al.

Для преобразования цифр 8-F используется регистр eax (также запись в обратном порядке, передача в выходной массив с помощью stosd – команда осуществляет запись в строку двойного слова, регистр di увеличивается на 4).

Если символ представляет собой нечто иное, то согласно условию он непосредственно записывается в выходной массив.

При каждой замене осуществляется переход на метку next. В ней осуществляется проверка на конец строки, в случае которого работа Ассемблерной части оканчивается, иначе происходит переход на метку checks, в которой и следуют вышеперечисленные проверки с заменами.

**Тестирование:**

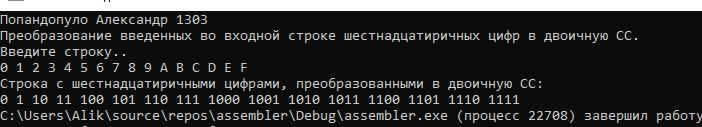


Рис. 1

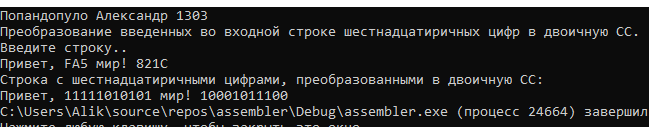


Рис. 2

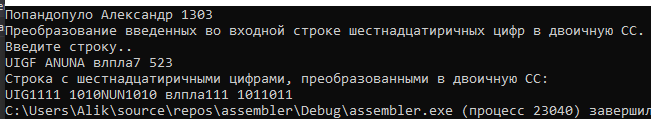


Рис. 3

**Приложение А.**

**Исходный код программы.**

#include <iostream>

#include <stdio.h>

#include <cstring>

#include <fstream>

char input[81];

char output[300];

int main() {

system("chcp 1251 > nul");

setlocale(LC\_CTYPE, "rus");

std::cout << "Попандопуло Александр 1303\nПреобразование введенных во входной строке шестнадцатиричных цифр в двоичную СС.\n";

std::cout << "Введите строку..\n";

std::cin.getline(input, 81);

std::ofstream file;

file.open("result.txt");

\_\_asm {

push ds

pop es

mov esi, offset input

mov edi, offset output

checks :

lodsb

cmp al, '2'

jne symbol3

mov ax, '01'

stosw

jmp next

symbol3 :

cmp al, '3'

jne symbol4

mov ax, '11'

stosw

jmp next

symbol4 :

cmp al, '4'

jne symbol5

mov ax, '01'

stosw

mov al, '0'

stosb

jmp next

symbol5 :

cmp al, '5'

jne symbol6

mov ax, '01'

stosw

mov al, '1'

stosb

jmp next

symbol6 :

cmp al, '6'

jne symbol7

mov ax, '11'

stosw

mov al, '0'

stosb

jmp next

symbol7 :

cmp al, '7'

jne symbol8

mov ax, '11'

stosw

mov al, '1'

stosb

jmp next

symbol8 :

cmp al, '8'

jne symbol9

mov eax, '0001'

stosd

jmp next

symbol9 :

cmp al, '9'

jne symbolA

mov eax, '1001'

stosd

jmp next

symbolA :

cmp al, 'A'

jne symbolB

mov eax, '0101'

stosd

jmp next

symbolB :

cmp al, 'B'

jne symbolC

mov eax, '1101'

stosd

jmp next

symbolC :

cmp al, 'C'

jne symbolD

mov eax, '0011'

stosd

jmp next

symbolD :

cmp al, 'D'

jne symbolE

mov eax, '1011'

stosd

jmp next

symbolE :

cmp al, 'E'

jne symbolF

mov eax, '0111'

stosd

jmp next

symbolF:

cmp al, 'F'

jne letter

mov eax, '1111'

stosd

jmp next

letter :

stosb

next :

mov ecx, '\0'

cmp ecx, [esi]

je end

jmp checks

end :

};

std::cout << "Строка с шестнадцатиричными цифрами, преобразованными в двоичную СС:\n";

std::cout << output;

file << output;

file.close();

return 0;

}