Правительство Российской Федерации

Федеральное государственное автономное образовательное учреждение

Высшего профессионального образования

Национальный исследовательский институт

**«Высшая школа экономики»**

Московский институт электроники и математики

ОП Компьютерная безопасность

**Отчет**

**По лабораторной работе No5**

**По курсу «Язык ассемблера»**

**Вариант № 26**

|  |  |  |  |
| --- | --- | --- | --- |
| Ф.И.О. студента | Номер группы | Дата | Баллы |
| Шаплавский Леонид Павлович | **СКБ 182** |  |  |

**Задание А5**

**Изображение выглядит как текст

Автоматически созданное описание**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <locale.h>

#include <ctype.h>

#include <cmath>

#include <iostream>

#include <bitset>

int main()

{

setlocale(LC\_ALL, "rus");

setlocale(LC\_NUMERIC, "eng");

int t = 0;

int number = 0;

char OBR = 0;

uint16\_t OBR1 = 0;

printf("Выберите режим обработки. Если байт - 1. Если слово - 2\n");

scanf\_s("%d", &t);

printf("Введите массив А: \n");

void\* A;

void\* A1[6];

char\* D = new char[6];

uint16\_t\* C = new uint16\_t[6];

for (int i = 0; i < 6; i++)

{

if (t == 1)

{

scanf\_s("%d", &D[i]);

}

else

{

scanf\_s("%d", &C[i]);

}

}

if (t == 1) A = D;

else A = C;

printf("Введите образец: ");

scanf\_s("%d", &number);

if (t == 1)

{

OBR = (unsigned char)number;

}

else

{

OBR1 = (uint16\_t)number;

}

printf("Исходный массив A\n");

printf("Двоичное представление: \n");

if (t == 2)

{

for (int i = 0; i < 6; i++)

{

std::cout << "Элемент " << i + 1 << " : " << ((uint16\_t\*)C)[i] << " = " << std::bitset<16>(((uint16\_t\*)C)[i]) << std::endl;

}

}

if (t == 1)

{

for (int i = 0; i < 6; i++)

{

std::cout << "Элемент " << i + 1 << " : " << (int)((char\*)D)[i] << " = " << std::bitset<8>(((char\*)D)[i]) << std::endl;

}

}

printf("Шестнадцатеричное представление: \n");

if (t == 2)

{

for (int i = 0; i < 6; i++)

{

printf("%08x", ((uint16\_t\*)C)[i]);

printf("\n");

}

}

if (t == 1)

{

for (int i = 0; i < 6; i++)

{

printf("%04x", ((char\*)D)[i]);

printf("\n");

}

}

printf("\n");

\_asm

{

mov esi, A; адресс начала массива А в esi

mov ecx, 6; количество элементов в массиве А выбрано равным 6

cmp t, 1; сравнение t с 1 (выбор обработки байта или слова)

jne SL; если t не равно 1, то переход на обработку слова

B :

mov al, [esi]; иначе обработка байта, текущий элемент массива А в al

mov bl, al; дублировать текущий элемент массива A в bl для выделения второго поля

mov dl, al; копия текущего исходного элемента массива А в dl

mov bh, OBR; образец в bh

and al, 01100000b; выделение битового поля 6:5(al изменен)

shr al, 5; логический сдвиг вправо на 5 разряда, прижать к правому краю

and bh, 00000110b; выделение битового поля 2:1

shr bh, 1;

and bl, 00000110b; выделение битового поля 2:1 (крайнего)

shr bl, 1;

and bl, bh; конъюнкция битовых полей 2:1 и образца(результат в bl)

cmp al, bl; сравнить выделенное битовое поле 6:5 и результат конъюнкиции 2 : 1 и образца

jne CLEAR7; если поля не совпадают, то ничего не делать

xor dl, 10000001b; иначе инвертировать старший и младший биты

jmp FIN1; перейти к продолжению цикла обработки байтовых элементов массива А

CLEAR7 :

nop;

FIN1:

mov[esi], dl; вернуть измененный элемент dl в массив А

inc esi; переместить указатель(адрес) на следующий элемент массива А(байтового)

loop B; конец цикла обработки байта

jmp FIN; переход на конец ассемблерной вставки

SL :

mov ax, [esi]; текущий элемент массива А в ax

mov bx, ax; дублировать текущий элемент массива А в bx для выделения второго поля

mov dx, ax; копия текущего исходного элемента массива А в dx

mov ax, OBR1; образец в ах

and ax, 0000000000000110b;

shr ax, 1;

and bx, 0000000000000110b; выделение битового поля 2:1 (крайнего)

shr bx, 1;

and bx, ax; конъюнкция битовых полей 2:1 и образца(результат в bx)

mov ax, [esi]; текущий элемент массива А в ах

and ax, 0000000001100000b; выделение поля 6:5

shr ax, 5; логический сдвиг вправо на 4 разряда, прижать к правому краю

cmp ax, bx; сравнить выделенное битовое поле 6:5 и результат конъюнкции или 2 : 1 и образца

jne CLEAR15; если поля не совпадают, то ничего не делать

xor dx, 1000000000000001b; иначе инвертировать старший и младший биты

jmp FIN2; перейте к продолжению цикла обработки элементов массива А из слов

CLEAR15 :

nop;

FIN2:

mov[esi], dx; вернуть изменений элемент dx в массив А

inc esi; переместить указатель(адрес) на следующий элемент массива А

inc esi; на 2, так как массив А - массив слов

loop SL; конец цикла обработки слова

FIN :

nop; пустая команда, конец ассемблерной вставки

}

printf("Массив A после преобразований\n");

printf("Двоичное представление: \n");

if (t == 2)

{

for (int i = 0; i < 6; i++)

{

std::cout << "Элемент " << i + 1 << " : " << ((uint16\_t\*)A)[i] << " = " << std::bitset<16>(((uint16\_t\*)A)[i]) << std::endl;

}

}

if (t == 1)

{

for (int i = 0; i < 6; i++)

{

std::cout << "Элемент " << i + 1 << " : " << (int)((char\*)A)[i] << " = " << std::bitset<8>(((char\*)A)[i]) << std::endl;

}

}

printf("Шестнадцатеричное представление: \n");

if (t == 2)

{

for (int i = 0; i < 6; i++)

{

printf("%08x", ((int\*)A)[i]);

printf("\n");

}

}

if (t == 1)

{

for (int i = 0; i < 6; i++)

{

printf("%04x", ((char\*)A)[i]);

printf("\n");

}

}

}

Тесты

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание