Национальный исследовательский университет
Высшая школа экономики
Московский институт электроники и математики

Департамент прикладной математики
кафедра компьютерной безопасности

Отчет по лабораторной работе №5 по дисциплине "Языки ассемблера"

Вариант: 30

Выполнил:
Фролов О.В.

Москва, 2023

Оглавление

Домашнее задание 1.

Задание А5

В массиве слов (байтов) выбранного размера (5-6 элементов) над каждым элементом выполнить операцию: если биты 6:5 являются инверсией битов 2:1, то установить младший и старший биты. Вывод на экран исходного массива и массива результатов производить в двоичном и шестнадцатеричном представлениях

```cpp
#include <iostream>
#include <bitset>
#include <string>

bool checkString(std::string str) {
    for (int i = 0; i < str.size(); ++i) {
        if (str[i] != '0' && str[i] != '1') {
            return false;
        }
    }
    return true;
}

int main() {
    setlocale(LC_ALL, "rus");

    short mode;
    int size;
    short sie;
    char byteArr[6];
    unsigned short wordArr[6];
    char tmp, tmp1=0, tmp2;
    while (true) {
        std::cout << "Select mode 0 or 1:\n";
        std::cin >> mode;
        if (mode == 0 || mode == 1) {
            std::cout << "Input the amount of elements (2 or 6):\n";
            std::cin >> size;
            if (size == 2 || size == 6) {
                if (size == 2)
                    sie = 2;
                else if (size == 6)
                    sie = 6;
                std::string str = "";
                if (mode == 0) {
                    bool flag = false;
                    for (int i = 0; i < sie; ++i) {
                        std::cout << "Input element (binary):\n";
                        std::cin >> str;
                        flag = !(checkString(str) && (str.size() <= 8));
                        if (flag)
                            break;
                        byteArr[i] = std::strtol(str.c_str(), NULL, 2);
                    }
```

```
                    byteArr[i] = std::bitset<8>(std::cout.c_str(), NULL, 2);
                }
                if (flag)
                    continue;
                std::cout << "Inputed array:\n";
                for (int i = 0; i < sie; ++i) {
                    std::cout << "byteArray[" << i << "] in binary view: " << std::bitset<8>(byteArr[i]) << " and in hex view: 0x" << std::hex << (unsigned int)byteArr[i
                }
            }
            else {
                bool flag = false;
                for (int i = 0; i < sie; ++i) {
                    std::cout << "Input element (binary):\n";
                    std::cin >> str;
                    flag = !(checkString(str) && (str.size() <= 16));
                    if (flag) {
                        break;
                    }
                    wordArr[i] = std::strtol(str.c_str(), NULL, 2);
                }
                if (flag)
                    continue;
                std::cout << "Inputed array:\n";
                for (int i = 0; i < sie; ++i) {
                    std::cout << "wordArray[" << i << "] in binary view: " << std::bitset<16>(wordArr[i]) << " and in hex: 0x" << std::hex << (unsigned int)wordArr[i] <<
                }
            }
            __asm {
                mov cx, sie; //поместили кол-во элементов в cx (5 или 6)
                mov ax, mode; //поместили выбранный режим (0 для байтового, 1 для слов)
                lea esi, wordArr; //поместили адрес массива wordArray в esi, если условие пройдет нижнее
                cmp ax, 0h; //сравниои ax с 1h в 16-чном представлении
                jne WORDMODE; //если ax не равен 1h, то работаем с массивом слов
                lea esi, byteArr; //поместили адрес массива byteArray в esi
                jmp L;
            L: //метка для цикла по элементам массива байт
                mov al, [esi]; //значение байта, хранящегося по адресу esi, из памяти поместили в регистр al
                //mov tmp1, al;
                mov bl, al; //поместили значение al в bl
                mov dl, al; //поместили значение al в dl
                and al, 01100000b; //выделили битовое поле 6:5 в al
                shr al, 5; //логический сдвиг вправо на 5 разрядов, прижали к правому краю
                and bl, 00000110b; //выделили битовое поле 2:1 в bl
                and bl, 00000110b; //выделили битовое поле 2:1 в bl
                shr bl, 1; //логический сдвиг вправо на 1 разряд, прижали к правому краю
                xor al, bl; //применили инверсию и сохранили в al
                //проверим на равенство 0, если не так, то 6:5 биты являются инверсией битов 2:1, иначе нет | даже должно выйти так, чтобы xor выдал в конце 11
                //cmp al, 0;
                cmp al, 00000011b;
                //jne INV; // перешли к сценарию с инверсией
                je INV;
                inc esi; //перешли к следующему адресу
                dec cx; //уменьшили счетчик
                cmp cx, 0; //проверили счетчик
                jne L; //вернулись в начало цикла
                jmp FINAL; //ушли в самый конец иначе

            INV:
                or dl, 10000001b; //установили старший и младший биты
                mov [esi], dl; //вернули измененный элемент массива
                inc esi; //перешли к следующему адресу
                dec cx; //уменьшили счетчик
                cmp cx, 0; //проверили счетчик
                jne L; //вернулись в начало цикла
                jmp FINAL; //ушли в самый конец иначе

            WORDMODE:
                mov ax, [esi]; //значение слова, хранящегося по адресу esi, из памяти поместили в регистр al
                mov bx, ax; //поместили значение ax в bx
                mov dx, ax; //поместили значение ax в dx
                and ax, 0000000001100000b; //выделили поле 6:5
                shr ax, 5; //логический сдвиг вправо на 5 разрядов, прижали к правому краю
                and bx, 0000000000000110b; // выделили поле 2:1
                shr bx, 1; //логический сдвиг вправо на 1 разряд, прижали к правому краю
                xor ax, bx; //применили инвертирование битов ax относительно bx и сохранили результат в ax
                //cmp ax, 0; //если не равно 0, тогда биты 6:5 являются инверсией битов 2:1
                //jne INVW;
                cmp ax, 0000000000000011b;
                je INVW;
                inc esi; //перешли к следующему адресу
                inc esi;
                dec cx; //уменьшили счетчик
                cmp cx, 0; //проверили счетчик
                jne WORDMODE; //вернулись в начало цикла
```

```
                    jne WORDMODE; //вернулись в начало цикла
                    jmp FINAL; //ушли в самый конец иначе
                INVW:
                    or dx, 1000000000000001b; //установили старший и младший биты
                    mov [esi], dx; //вернули измененный элемент массива
                    inc esi; //перешли к следующему адресу
                    inc esi;
                    dec cx; //уменьшили счетчик
                    cmp cx, 0; //проверили счетчик
                    jne WORDMODE; //вернулись в начало цикла
                    jmp FINAL; //ушли в самый конец иначе
                FINAL:
                    nop;
                }
                std::cout << std::bitset<8>(tmp)<<" " << std::bitset<8>(tmp1)<<" " << std::bitset<8>(tmp2)<<"\n";
                std::cout << "Output array:\n";
                if (mode == 0) {
                    for (int i = 0; i < sie; ++i) {
                        std::cout << "byteArray[" << i << "] in binary view: " << std::bitset<8>(byteArr[i]) << " and in hex: 0x" << std::hex << (unsigned int)byteArr[i] <<
                    }
                }
                else if (mode == 1) {
                    for (int i = 0; i < sie; ++i) {
                        std::cout << "wordArray[" << i << "] in binary view: " << std::bitset<16>(wordArr[i]) << " and in hex: 0x" << std::hex << (unsigned int)wordArr[i] <<
                    }
                }
                std::cout << "Again? (0 or 1)\n";
                std::cin >> str;
                if (str != "1")
                    break;
            }
            else {
                std::cout << "Invalid values!\n";
            }
        }
        else {
            std::cout << "Invalid value!\n";
        }
    }
    exit(0);
}
```

Принцип работы: Сначала пользователь выбирает тип массивов (байтовый - 0, слов - 1). Дальше происходит выбор кол-ва элементов в массиве: 5 или 6. После чего происходит последовательный ввод элементов массива. Перед работой основной части программы будет выведен исходный массив в двоичной, шестандцатиричной и десятичной системах счисления.

Далее выполняется ассемблерная вставка: в регистры esi помещается адрес массива. В cx помещается максимальное кол-во итераций последующих процедур (кол-во элементов массива). Далее в al(ax) и в bl(bx) с dl(dx) помещается текущий элемент массива, с применением масок и смещения вправо n-ого кол-ва бит происходит применение операции xor к содержимому al(ax) и bl(bx) регистров. Если результат равен 3 (в десятичной системе счисления), то биты 6:5 являются инверсией битов 2:1, после чего происходит переход на метку INV(INVW), иначе переход к следующей итерации цикла.

На метке INV(INVW) происходит установка старшего и младшего бита и возвращение к циклу.

После завершения цикла пользователю вернется измененный массив в двоичной, шестнадцатиричной и десятичной системах счисления. Далее поступает вопрос о том, нужно ли повторить работу программы, или же можно завершить выполнение программы.

1. Тесты

```
D:\Games\Frolov-A5\Debug\Frolov-A5.exe
Select mode 0 or 1:
0
Input the amount of elements (5 or 6):
5
Input element (binary):
01101000
Input element (binary):
00000000
Input element (binary):
11111111
Input element (binary):
11101001
Input element (binary):
00010110
Inputed array:
byteArray[0] in binary view: 01101000 and in hex view: 0x68 and in dec: 104
byteArray[1] in binary view: 00000000 and in hex view: 0x0 and in dec: 0
byteArray[2] in binary view: 11111111 and in hex view: 0xffffffff and in dec: 4294967295
byteArray[3] in binary view: 11101001 and in hex view: 0xfffffe9 and in dec: 4294967273
byteArray[4] in binary view: 00010110 and in hex view: 0x16 and in dec: 22
11001100 00000000 11001100
Output array:
byteArray[0] in binary view: 11101001 and in hex: 0xfffffe9 and in dec: 4294967273
byteArray[1] in binary view: 00000000 and in hex: 0x0 and in dec: 0
byteArray[2] in binary view: 11111111 and in hex: 0xffffffff and in dec: 4294967295
byteArray[3] in binary view: 11101001 and in hex: 0xfffffe9 and in dec: 4294967273
byteArray[4] in binary view: 10010111 and in hex: 0xffffff97 and in dec: 4294967191
Again? (0 or 1)
```

```
D:\Games\Frolov-A5\Debug\Frolov-A5.exe
1
Select mode 0 or 1:
1
Input the amount of elements (5 or 6):
5
Input element (binary):
0101010101101000
Input element (binary):
0000000000000000
Input element (binary):
1111111111111111
Input element (binary):
1010101011101001
Input element (binary):
0101010100010110
Inputed array:
wordArray[0] in binary view: 0101010101101000 and in hex: 0x5568 and in dec: 21864
wordArray[1] in binary view: 0000000000000000 and in hex: 0x0 and in dec: 0
wordArray[2] in binary view: 1111111111111111 and in hex: 0xffff and in dec: 65535
wordArray[3] in binary view: 1010101011101001 and in hex: 0xaae9 and in dec: 43753
wordArray[4] in binary view: 0101010100010110 and in hex: 0x5516 and in dec: 21782
11001100 00000000 11001100
Output array:
wordArray[0] in binary view: 1101010101101001 and in hex: 0xd569 and in dec: 54633
wordArray[1] in binary view: 0000000000000000 and in hex: 0x0 and in dec: 0
wordArray[2] in binary view: 1111111111111111 and in hex: 0xffff and in dec: 65535
wordArray[3] in binary view: 1010101011101001 and in hex: 0xaae9 and in dec: 43753
wordArray[4] in binary view: 1101010100010111 and in hex: 0xd517 and in dec: 54551
Again? (0 or 1)
```