**Министерство науки и высшего образования Российской Федерации**

Федеральное государственное бюджетное образовательное учреждение высшего образования «Рязанский государственный радиотехнический университет имени В.Ф. Уткина»

Кафедра «ЭВМ»

Машинно-зависимые языки программирования

**Отчет о лабораторной работе № 3**

по теме

«Арифметические операции над разными типами данных»

**Выполнили:**

Студенты гр. 045

Анохин В.А.

Вашкулатов Н.А.

**Проверил:**

Доц. Муратов Е.Р

Асс. Тарасов А.С.

**Цель работы:** научиться размещать массивы данных в памяти и выполнять арифметические действия над разными типами данных.

**Ход работы:**

1. Рассчитайте контрольную сумму для первых 256 слов (word), прочитанных из файла w\_512.dat. Контрольная сумма рассчитывается по следующим образом: - суммируются все 256 слов. Результат берется по модулю 0x1000h, затем из 0 вычитается полученный результат. Полученное после вычитания значения будем считать контрольной суммой (CRC16).
2. Рассчитайте контрольную сумму для первых 256 двойных слов (dword), прочитанных из файла w\_512.dat. В этом случае сумма берется по модулю 0x10000000h, затем из 0 вычитается полученный результат. Полученное после вычитания значения будем считать контрольной суммой (CRC32).
3. Для 512 значений типа word, четные слова с нечетными. Результат сохраните в отдельный массив в памяти.
4. Создайте массив, в который сохраните все нечетные значения из первых 1024 элементов типа dword прочитанных из файла. Выведите количество полученных элементов.
5. Выведите последние два байта прочитанные из файла.
6. Выведите первый байт, сдвинутый на бит влево и первое слово, сдвинутое на 3 бита вправо (сдвигать инструкциями циклического сдвига, логического сдвига и арифметического сдвига).

**Код программы:**

.686P

.MODEL FLAT, STDCALL

.STACK 4096

option casemap : none

include c:\masm32\include\windows.inc

include c:\masm32\include\user32.inc

includelib c:\masm32\lib\user32.lib

include c:\masm32\include\kernel32.inc

includelib c:\masm32\lib\kernel32.lib

.DATA

FileName db "B:\w\_512.dat", 0

BadText db "File dont open!", 0

EvenWordsC dd 0

OddWordsC dd 0

Ex4\_counter dd 0

.DATA?

hFile HANDLE ?

hMemory DWORD ?

pMemory DWORD ?

dwBytesRead dd ?

dwFileSize dd ?

CRC16 dw ?

CR32 dd ?

ADDRCRC16 dd ?

ADDRCR32 dd ?

EvenWordsAcc dw 512 dup(? )

OddWordsAcc dw 512 dup(? )

Ex4\_Arr dd 1024 dup(? )

.CODE

START : ;чтение из файла

invoke CreateFile, addr FileName, GENERIC\_READ, 0, NULL, OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, NULL

mov hFile, eax

cmp hFile, INVALID\_HANDLE\_VALUE

jz ErrorMsg

invoke GetFileSize, hFile, NULL

mov dwFileSize, eax

invoke GlobalAlloc, GMEM\_FIXED or GMEM\_ZEROINIT, dwFileSize

mov hMemory, eax

invoke GlobalLock, hMemory

mov pMemory, eax

invoke ReadFile, hFile, pMemory, 4096, addr dwBytesRead, NULL

or eax, eax

jz ErrorMsg

mov eax, pMemory

ex1 :

mov ecx, 256

cycle1 :

mov ebx, 256

sub ebx, ecx

sal ebx, 1

add eax, ebx

add dx, [eax]

sub eax, ebx

dec ecx

jz ex1\_end

jmp cycle1

ex1\_end :

xor dx, 1000h

mov CRC16, dx

mov ADDRCRC16, offset(CRC16)

ex2 :

mov ecx, 256

cycle2 :

mov ebx, 256

sub ebx, ecx

sal ebx, 2

add eax, ebx

add edx, [eax]

sub eax, ebx

dec ecx

jz ex2\_end

jmp cycle2

ex2\_end :

xor edx, 10000000h

mov CR32, edx

mov ADDRCR32, offset(CR32)

ex3 :

mov ecx, 512

cycle3 :

mov ebx, 512

sub ebx, ecx

sal ebx, 1

add eax, ebx

mov dx, [eax]

sub eax, ebx

mov bx, dx

sar bx, 1

jc c3

mov ebx, EvenWordsC

mov EvenWordsAcc[ebx \* 2], dx

inc EvenWordsC

jmp c3End

c3 :

mov ebx, OddWordsC

mov OddWordsAcc[ebx \* 2], dx

inc OddWordsC

c3End :

dec ecx

jz ex3\_end

jmp cycle3

ex3\_end :

ex4:

mov ecx, 1024

cycle4 :

mov ebx, 1024

sub ebx, ecx

sal ebx, 2

add eax, ebx

mov edx, [eax]

sub eax, ebx

mov ebx, edx

sar ebx, 1

jnc c4

mov ebx, Ex4\_counter

mov Ex4\_Arr[ebx \* 2], edx

inc Ex4\_counter

c4 :

dec ecx

jz ex4\_end

jmp cycle4

ex4\_end :

mov ebx, Ex4\_counter

ex5 :

mov bx, [eax + 4094]

ex6 :

mov bl, [eax]

mov cx, [eax]

shl bl, 1

sar cx, 3

invoke GlobalUnlock, pMemory

invoke GlobalFree, hMemory

invoke CloseHandle, hFile

jmp End\_code

ErrorMsg :

invoke MessageBox, NULL, addr BadText, addr BadText, MB\_OK

invoke ExitProcess, 0

End\_code :

RET

END START

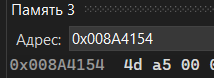


Рисунок 1 – Задание 1. Контрольная сумма

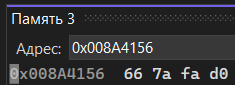


Рисунок 2– Задание 2. Контрольная сумма



Рисунок 3 – Задание 3. Количество четных чисел



Рисунок 4 – Задание 4. Количество нечетных чисел

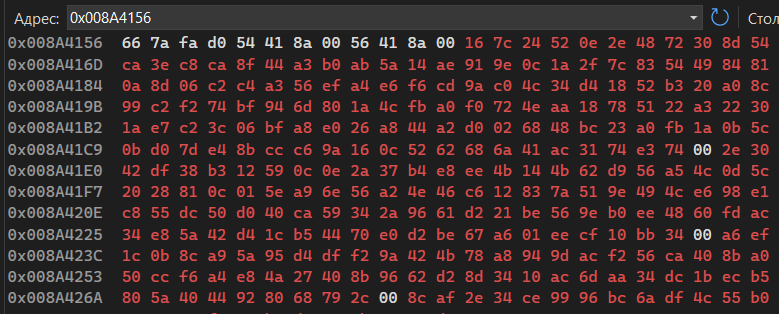


Рисунок 5 – Задание 3. Массивы чисел



Рисунок 6 – Задание 4. Количество нечетных чисел



Рисунок 7 – Задание 5. Последние два байта



Рисунок 8 – Задание 6. Сдвиг влево на 1



Рисунок 9 – Задание 6. Сдвиг вправо на 3

**Вывод:** в ходе выполнения работы мы научились размещать массивы данных в памяти и выполнять арифметические действия над разными типами данных.