Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Факультет среднего профессионального образования

**ОТЧЕТ**

**о Лабораторной работе № 10**

по теме: SSE и AVX-команды

по дисциплине: Системное программирование

Специальность:

09.02.07 Информационные системы и программирование

|  |  |
| --- | --- |
| Проверил:  Тропченко А.А. \_\_\_\_\_\_\_\_\_\_  Дата: «04» декабря 2021г.  Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Выполнил:  Студент группы Y2431  Головачев Д.А. |

Санкт-Петербург 2020/2021

Лабораторная работа 6-1 «SSE и AVX-команды»

Цель занятия:

приобрести практические навыки составления, отладки и выполнения программ с использованием структур, написанных языком ассемблера для платформы х64 в среде masm64.

Вариант 5: "Выполнить параллельное сравнение 2-х массивов по 7-мь 64-разрядных",10,

"вещественных чисел. Если один массив меньше второго, то выполнить операцию (4a – c)b – d/b",10,

"иначе – выполнить операцию d/b.",10, Текст программы:

Листинг программы SSE.

include \masm64\include64\masm64rt.inc

IDM\_Functions equ 10002

IDM\_EXIT equ 10003 ; выход с программы

IDM\_ABOUT equ 10005 ; выход с программы

.data

mas1 dd -1.3, 2.1, 3.8, 1.0, 5.4, 6.12,7.54

mas2 dd -1., -5.0,-3.54,1.5,-5.8,-6.53,7.5

len1 equ ($-mas2)/ type mas2 ; количество чисел массива mas2

a1 dd 3.0 ;

b1 dd 0.2 ;

c1 dd 1.0 ;

d1 dd 2.2 ;

const dd 4.0 ;

fmt db "Выполнить параллельное сравнение 2-х массивов по 7-мь 64-разрядных",10,

"вещественных чисел. Если один массив меньше второго, то выполнить операцию (4a – c)b – d/b",10,

"иначе – выполнить операцию d/b.",10,

"Результат = %d",10,10,"Автор: Головачев Д.А. ФСПО, y2431",0

buf dq 0,0 ; размер буфера

titl1 db "masm64. Параллельное сравнение с помощью SSE-команд",0;

buf1 dq 0,0

hInstance dq ? ; дескриптор програми

hWnd dq ? ; дескриптор окна

hIcon dq ? ; дескриптор иконки

hCursor dq ? ; дескриптор курсора

sWid dq ? ; ширина монитора (колич. пикселей по x)

sHgt dq ? ; высота монитора (колич. пикселей по y)

classname db "template\_class",0

caption db "Окно с Menu. masm64",0

.code

entry\_point proc

mov hInstance,rv(GetModuleHandle,0) ; получение и сохранение дескрипторa програми

mov hIcon, rv(LoadIcon,hInstance,10) ; загрузка и сохранение дескрипторa иконки

mov hCursor,rv(LoadCursor,0,IDC\_ARROW) ; загрузка курсора и сохранение

mov sWid,rv(GetSystemMetrics,SM\_CXSCREEN) ; получение кол. пикселей по х монитора

mov sHgt,rv(GetSystemMetrics,SM\_CYSCREEN) ; получение кол. пикселей по y монитора

;mov hBrush,rvcall(CreateSolidBrush,00C4C4C4h)

call main ; вызов процедуры main

rcall ExitProcess,0

ret

entry\_point endp

main proc

LOCAL wc :WNDCLASSEX ; объЯвление локальных переменных

LOCAL lft :QWORD ; ‹ок. переменные содержатся в стеке

LOCAL top :QWORD ; и существуют только во времЯ вып. проц.

LOCAL wid :QWORD

LOCAL hgt :QWORD

mov wc.cbSize,SIZEOF WNDCLASSEX ; колич. байтов структуры

mov wc.style,CS\_BYTEALIGNCLIENT or CS\_BYTEALIGNWINDOW ; стиль окна

mov wc.lpfnWndProc,ptr$(WndProc) ; адрес процедуры WndProc

mov wc.cbClsExtra,0 ; количество байтов для структуры класса

mov wc.cbWndExtra,0 ; количество байтов для структуры окна

mrm wc.hInstance,hInstance ; заполнение поля дескриптора в структуре

mrm wc.hIcon, hIcon ; хэндл иконки

mrm wc.hCursor,hCursor ; хэндл курсора

mrm wc.hbrBackground,0;hBrush цвет окна

mov wc.lpszMenuName,0 ; заполнение поля в структуре с именем ресурса меню

mov wc.lpszClassName,ptr$(classname) ; имЯ класса

mrm wc.hIconSm,hIcon

invoke RegisterClassEx,ADDR wc ; регистрация класса окна

mov wid, 500 ; ширина пользовательского окна в пикселЯх

mov hgt, 250 ; высота пользовательского окна в пикселЯх

mov rax,sWid ; колич. пикселей монитора по x

sub rax,wid ; дельта • = •(монитора) - х(окна пользователЯ)

shr rax,1 ; получение середины •

mov lft,rax ;

mov rax, sHgt ; колич. пикселей монитора по y

sub rax, hgt ;

shr rax, 1 ;

mov top, rax ;

invoke CreateWindowEx,WS\_EX\_LEFT or WS\_EX\_ACCEPTFILES, \

ADDR classname,ADDR caption, \

WS\_OVERLAPPED or WS\_VISIBLE or WS\_SYSMENU,\

lft,top,wid,hgt,0,0,hInstance,0

mov hWnd,rax ; сохранение дескриптора окна

call msgloop

ret

main endp

msgloop proc

LOCAL msg :MSG

LOCAL pmsg :QWORD

mov pmsg, ptr$(msg) ; получение адреса структуры сообщениЯ

jmp gmsg ; jump directly to GetMessage()

mloop:

rcall TranslateMessage,pmsg

rcall DispatchMessage,pmsg

gmsg:

test rax, rvcall(GetMessage,pmsg,0,0,0) ; пока GetMessage не вернет ноль

jnz mloop

ret

msgloop endp

WndProc proc hWin:QWORD,uMsg:QWORD,wParam:QWORD,lParam:QWORD

LOCAL dfbuff[260]:BYTE

LOCAL pbuff :QWORD

.switch uMsg

.case WM\_COMMAND

.switch wParam

.case IDM\_Functions ; 10002

mov eax,len1 ;

mov ebx,4 ; колич. 32-разрядных чисел в 128-разрядном регистре

xor edx,edx ;

div ebx ; опр. кол. циклов для параллельного считывания и остатка

mov ecx,eax ; счетчик циклов для параллельного считывания

lea rsi,mas1 ;

lea rdi,mas2 ;

lea rsi,mas1 ;

lea rdi,mas2 ;

next: movups XMM0,xmmword ptr [rsi]; 4- 32 числа из mas1

movups XMM1,[rdi] ; 4- 32 числа из mas2

cmpltps XMM0,XMM1 ; сравнение на меньше: если меньше, то нули

movmskps ebx,XMM0 ; перенесение знаковых битов

add rsi,16 ; подготовка адреса для нового считывания mas1

add rdi,16 ; подготовка адреса для нового считывания mas2

dec ecx ; уменьшение счетчика циклов

jnz m1 ; проверка счетчика на ненулевое значение

jmp m2 ;

m1: mov r10,rbx

shl r10,4 ; сдвиг налево на 4 бита

jmp next ; на новый цикл

m2: cmp edx,0 ; проверка остатка

jz \_end ;

mov ecx,edx ; если в остатке не нуль, то установка счетчика

m4:

movss XMM0,dword ptr[rsi] ;

movss XMM1,dword ptr[rdi] ;

comiss XMM0,XMM1 ; сравнение младших чисел массивов

jg @f ; если больше

shl r10,1 ; сдвиг влево на 1 разряд

inc r10 ; встановление 1, поскольку XMM0[0] < XMM1[0]

jmp m3

@@:

shl r10,1 ; сдвиг налево на 1 разряд

m3:

add rsi,4 ; адреса для нового числа mas1

add rdi,4 ; адреса для нового числа mas2

loop m4

\_end: cmp r10,0 ; проверка знаковых битов

jz mb ; если ebx = 0, то перейти на метку mb

movss xmm2,dword ptr d1

movss xmm3,dword ptr b1

divss xmm2,xmm3 ; d/b

jmp m5

mb: movss xmm2,dword ptr a1

mulss xmm2, dword ptr const

subss xmm4,dword ptr c1 ; 4a - c

mulss xmm2,b1 ;(a - c)b

movss xmm5,dword ptr d1

movss xmm6,dword ptr b1 ;

divss xmm5,xmm6 ; d/b

subss xmm2,xmm5 ;(a - c)b - d/b

m5: cvttss2si eax,xmm2

movsxd r15,eax

invoke wsprintf,addr buf1,addr fmt,r15

invoke MessageBox,0,addr buf1,ADDR titl1,MB\_ICONINFORMATION

.case IDM\_EXIT ; 10003

invoke MsgboxI,hWin,"masm64","Использование rcall и rvcall",MB\_OK,10

.case IDM\_ABOUT ; 10005

invoke MessageBox,hWin,"Автор: Головачев Д.А. ФСПО","Автор",MB\_ICONINFORMATION

rcall SendMessage,hWin,WM\_SYSCOMMAND,SC\_CLOSE,NULL

.endsw

.case WM\_CREATE

rcall LoadMenu,hInstance,10000

rcall SetMenu,hWin,rax

.return 0

.case WM\_CLOSE

rcall SendMessage,hWin,WM\_DESTROY,0,0

.case WM\_DESTROY

rcall PostQuitMessage,NULL

.endsw

rcall DefWindowProc,hWin,uMsg,wParam,lParam

ret

WndProc endp

End

Листинг программы SSE.

include \masm64\include64\masm64rt.inc ; библиотеки

.data

mas1 dd -1.3, 2.1, 3.8, 1.0, 5.4, 6.12,7.54

mas2 dd -1.0, -5.0,-3.54,1.5,-5.8,-6.53,7.5

len1 equ ($-mas2)/ type mas2 ; количество чисел массива mas2

a1 dd 3.0 ;

b1 dd 0.2 ;

c1 dd 1.0 ;

d1 dd 2.2 ;

const dd 4.0 ;

fmt db "Выполнить параллельное сравнение 2-х массивов по 7-мь 64-разрядных",10,

"вещественных чисел. Если один массив меньше второго, то выполнить операцию (4a – c)b – d/b",10,

"иначе – выполнить операцию d/b.",10,

"Результат = %d",10,10,"Автор: Головачев Д.А. ФСПО, y2431",0

buf dq 0,0 ; размер буфера

titl1 db "masm64. Параллельное сравнение с помощью SSE-команд",0;

buf1 dq 0,0

.code

entry\_point proc

mov eax,len1 ;

mov ebx,4 ; колич. 32-разрядных чисел в 128-разрядном регистре

xor edx,edx ;

div ebx ; опр. кол. циклов для параллельного считывания и остатка

mov ecx,eax ; счетчик циклов для параллельного считывания

lea rsi,mas1 ;

lea rdi,mas2 ;

lea rsi,mas1 ;

lea rdi,mas2 ;

next: movups XMM0,xmmword ptr [rsi]; 4- 32 числа из mas1

movups XMM1,[rdi] ; 4- 32 числа из mas2

vpcmpgtb XMM3,XMM0,XMM1 ; сравнение на меньше: если меньше, то нули

movmskps ebx,XMM0 ; перенесение знаковых битов

add rsi,16 ; подготовка адреса для нового считывания mas1

add rdi,16 ; подготовка адреса для нового считывания mas2

dec ecx ; уменьшение счетчика циклов

jnz m1 ; проверка счетчика на ненулевое значение

jmp m2 ;

m1: mov r10,rbx

shl r10,4 ; сдвиг налево на 4 бита

jmp next ; на новый цикл

m2: cmp edx,0 ; проверка остатка

jz \_end ;

mov ecx,edx ; если в остатке не нуль, то установка счетчика

m4:

movss XMM0,dword ptr[rsi] ;

movss XMM1,dword ptr[rdi] ;

comiss XMM0,XMM1 ; сравнение младших чисел массивов

jg @f ; если больше

shl r10,1 ; сдвиг влево на 1 разряд

inc r10 ; встановление 1, поскольку XMM0[0] < XMM1[0]

jmp m3

@@:

shl r10,1 ; сдвиг налево на 1 разряд

m3:

add rsi,4 ; адреса для нового числа mas1

add rdi,4 ; адреса для нового числа mas2

loop m4

\_end: cmp r10,0 ; проверка знаковых битов

jz mb ; если ebx = 0, то перейти на метку mb

movss xmm2,dword ptr d1

movss xmm3,dword ptr b1

divss xmm2,xmm3 ; d/b

jmp m5

mb: movss xmm2,dword ptr a1

mulss xmm2, dword ptr const

subss xmm4,dword ptr c1 ; 4a - c

mulss xmm2,b1 ;(a - c)b

movss xmm5,dword ptr d1

movss xmm6,dword ptr b1 ;

divss xmm5,xmm6 ; d/b

subss xmm2,xmm5 ;(a - c)b - d/b

m5: cvttss2si eax,xmm2

movsxd r15,eax

invoke wsprintf,addr buf1,addr fmt,r15

invoke MessageBox,0,addr buf1,ADDR titl1,MB\_ICONINFORMATION

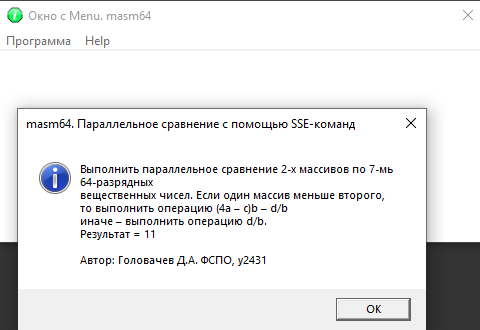
invoke ExitProcess,0

entry\_point endp

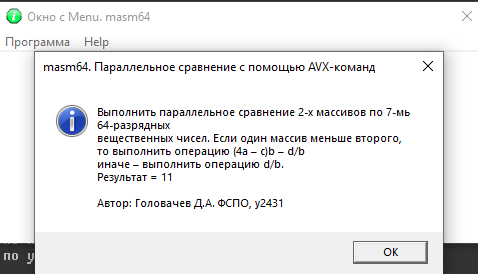
end

Результат выполнения программы и отладчиков:

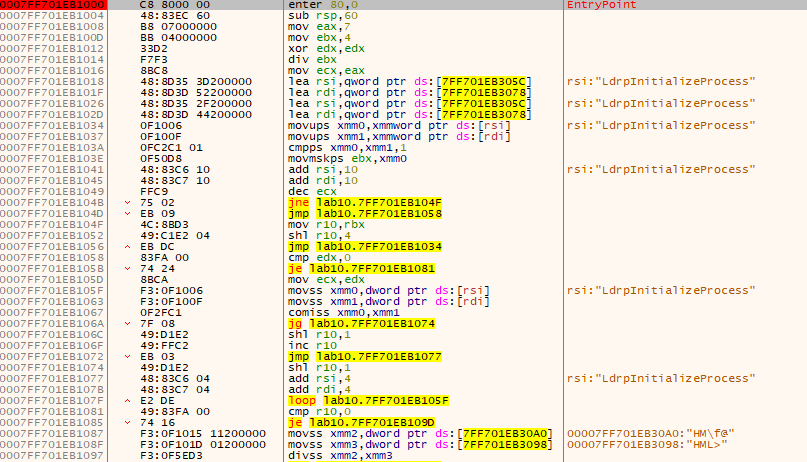
*Рисунок 1. Результат выполнения программы SSE*

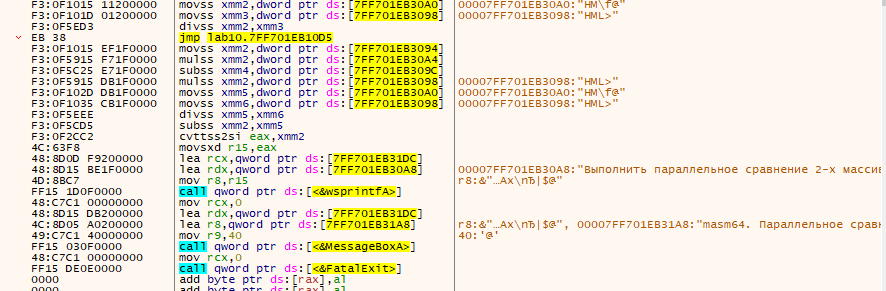


*Рисунок 2. Результат выполнения программы AVX*

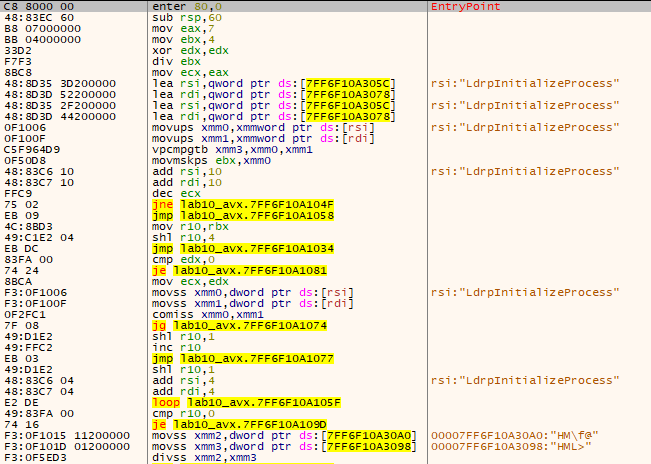


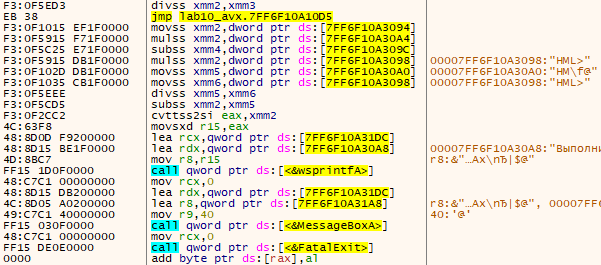
*Рисунок 3. Алгоритм выполнения SSE*





*Рисунок 4. Алгоритм выполнения AVX*





Особенности программы.

Программа работает на SSE- и AVX-командах.