Учреждение образования Республики Беларусь

«Гомельский государственный технический университет им. П. О. Сухого»

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

Отчет по лабораторной работе №5

по дисциплине «Архитектура ЭВМ»

по теме «Выполнение целочисленных арифметических операций в процессоре x86»

Выполнил:

студент группы ИТП-21

Половцев М. С.

Проверил:

преподаватель

Савельев В. А.

Гомель, 2022

**Цель:** знакомство с работой центрального процессора и памяти; изучение арифметических команд Ассемблера; получение навыков работы с отладчиком.

**Ход работы**

Задание

1. Написать программу, вычисляющую заданное в соответствии с номером студента в журнале выражение:

19. n = a \* x2 – b \* y / a + x / (y + a);

Листинг:

.386

.model flat, stdcall

option casemap:none

include C:\masm32\include\windows.inc

include C:\masm32\include\kernel32.inc

includelib C:\masm32\lib\kernel32.lib

; Сегмент данных

;--------------------------------------------------

.data

a dw 1

b dw 6

x dw 8

y dw 7

.data?

n dw ?

; Сегмент кода

;--------------------------------------------------

.code

; n = a \* x ^ 2 – b \* y / a + x / (y + a)

start:

; a \* x ^ 2

mov ax, x

imul ax

imul a

mov bx, ax

; b \* y / a

mov ax, b

imul y

idiv a

; a \* x ^ 2 - b \* y / a

sub bx, ax

; x / (y + a)

mov cx, y

add cx, a

mov ax, x

idiv cx

; a \* x ^ 2 – b \* y / a + x / (y + a)

add bx, ax

; n = a \* x ^ 2 – b \* y / a + x / (y + a)

mov n, bx

; завершаем процесс

push NULL

call ExitProcess

end start

2. Открыть программу в отладчике. Указать адреса расположения переменных в памяти, заполнив таблицу:

|  |  |  |
| --- | --- | --- |
| Имя переменной | Адрес | Порядок байт в памяти |
| a | 00403000 | 0100 |
| b | 00403002 | 0600 |
| x | 00403004 | 0700 |
| y | 00403006 | 0800 |

3. Выполнить программу в пошаговом режиме. После выполнения каждого шага заносить данные в таблицу

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Регистры | | | | Флаги | | | | |
| EAX | EBX | ECX | EDX | FC | FP | FA | FZ | FS |
| 0019FFCC | 003A3000 | 00401000 | 00401000 | 0 | 1 | 0 | 1 | 0 |
| 00190008 | 003A3000 | 00401000 | 00401000 | 0 | 1 | 0 | 1 | 0 |
| 00190040 | 003A3000 | 00401000 | 00400000 | 0 | 1 | 0 | 1 | 0 |
| 00190040 | 003A0040 | 00401000 | 00400000 | 0 | 1 | 0 | 1 | 0 |
| 00190006 | 003A0040 | 00401000 | 00400000 | 0 | 1 | 0 | 1 | 0 |
| 0019002A | 003A0040 | 00401000 | 00400000 | 0 | 1 | 0 | 1 | 0 |
| 0019002A | 003A0040 | 00401000 | 00400000 | 0 | 0 | 1 | 0 | 0 |
| 0019002A | 003A0016 | 00401000 | 00400000 | 0 | 0 | 1 | 0 | 0 |
| 0019002A | 003A0016 | 00400007 | 00400000 | 0 | 0 | 1 | 0 | 0 |
| 0019002A | 003A0016 | 00400008 | 00400000 | 0 | 0 | 0 | 0 | 0 |
| 0019002A | 003A0016 | 00400008 | 00400000 | 0 | 0 | 0 | 0 | 0 |
| 00190008 | 003A0016 | 00400008 | 00400000 | 0 | 0 | 0 | 0 | 0 |
| 00190001 | 003A0016 | 00400008 | 00400000 | 0 | 0 | 1 | 0 | 0 |
| 00190001 | 003A0017 | 00400008 | 00400000 | 0 | 1 | 0 | 0 | 0 |

4. Выводы по пункту 2: размещение инициализированных данных начинается с адреса 00403000. Каждое значение в памяти занимает столько байт, сколько резервируется соответствующей директивой. В отладчике каждый байт представлен двумя шестнадцатеричными цифрами. Используется обратный порядок байт, т.е. младший байт числа находится в младших адресах памяти (перед старшим). Если шестнадцатеричная комбинация соответствует коду символа, то он высвечивается в следующем столбце (ASCII), иначе в нем высвечивается точка.

Выводы по пункту 3: Изначально в регистре EAX находится значение 0019FFCC. Оно изменяется после применения команды mov. При каждом выполнении команды мы можем наблюдать изменение данных в памяти и/или в регистрах, отслеживая процесс выполнения программы и контролируя правильность промежуточных результатов.

**Вывод:** к основным арифметическим операциям данных относятся add, sub, imul, mul, idiv, div и т. д. Большинство команд принимают два параметра и требуют одинаковый размер параметров. Средствами отладчика можно пошагово отследить выполнение программы (например, содержимое регистров).