МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА» (БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №3

по дисциплине: Архитектура вычислительных систем

тема: «Арифметические команды центрального процессора»

Выполнил: ст. группы ПВ-223

Игнатьев Артур Олегович

Проверил:

Осипов Олег Васильевич

Белгород 2024 г.

**Цель работы:** изучение арифметических команд центрального процессора для работы с целыми числами.

**Задачи**:

1. Написать программу для вычисления значения арифметического выражения согласно варианту задания. Все переменные, используемые в программе, требуется использовать как знаковые и расширять до размерности двойного слова. Результат должен быть записан в регистр **EAX**. Если результат содержит остаток от деления, оставить его в регистре **EDX**. Подобрать набор тестовых данных (не менее 3). Каждая строка исходного кода программы обязательно должна быть прокомментирована. Программы без подробных комментариев не принимаются!
2. Написать программу для сложения или вычитания целых беззнаковых чисел большой размерности (размерность и операция зависят от варианта задания). Младшие байты при этом хранить по младшему адресу. Подобрать наборы тестовых данных (не менее 3). Для выполнения этого задания изучить теоретический материал главы «Вычитание и сложение операндов большой размерности», начиная со страницы 176 учебника Юрова «Assembler».

**Задание варианта №3**

Размер входных параметров:

x – word

y – word

z – word

Операция Размерность (2-е задание):

сложение 30 байт

**Выполнение работы**

**Задание 1:**

Код программы:

.DATA

x DW ? ; Входной параметр x

y DW ? ; Входной параметр y

z DW ? ; Входной параметр z

str\_fmt DB "x = %hd, y = %hd, z = %hd, R = %d", 0

.CODE

START:

; Вычисляем (x - 3^4)^2

MOVSX EAX, x ; Загружаем x в EAX

MOV EBX, 81 ; 3^4 = 81

SUB EAX, EBX ; x - 81

IMUL EAX, EAX ; (x - 81)^2

MOV ECX, EAX ; Сохраняем результат в ECX

; Вычисляем (y - 4)^2

MOVSX EAX, y ; Загружаем y в EAX

SUB EAX, 4 ; y - 4

IMUL EAX, EAX ; (y - 4)^2

ADD ECX, EAX ; Суммируем с предыдущим результатом

; Вычисляем (z + 5)^2

MOVSX EAX, z ; Загружаем z в EAX

ADD EAX, 5 ; z + 5

IMUL EAX, EAX ; (z + 5)^2

ADD ECX, EAX ; Суммируем с общей суммой

; Делим результат на 4

MOV EAX, ECX ; Перемещаем сумму в EAX

MOV EBX, 4 ; Делитель

CDQ ; Расширяем знак

IDIV EBX ; Результат деления в EAX

; Подготовка к вызову printf

PUSH EAX ; Помещаем результат в стек для вывода

PUSH DWORD PTR z ; Помещаем z

PUSH DWORD PTR y ; Помещаем y

PUSH DWORD PTR x ; Помещаем x

PUSH OFFSET str\_fmt ; Адрес строки формата

CALL crt\_printf ; Вызов printf

; Очистка стека

ADD ESP, 5\*4

; Завершаем программу

PUSH 0

CALL ExitProcess ; Выход из программы

END START

Тестовые данные:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x | y | z | EAX | EDX |
| 5 | -1 | 3 | 000005BAh = 1466 | 00000001h= 1 |
| 1 | 0 | -5 | 00000644h= 1604 | 00000000h= 0 |
| -10 | 2 | 3 | 00000827h= 2087 | 00000001h= 1 |
| 100 | -5 | -10 | 00000074h= 116 | 00000003h= 3 |

**Задание 2**:

Код программы:

.DATA

a db 17h, 0DDh, 24h, 4Bh, 57h, 7Fh, 0CDh, 05h, 2Dh, 13h, 34h, 0BBh, 12h, 3Fh,

11h, 22h, 33h, 44h, 55h, 66h, 77h, 88h, 99h, 0AAh, 0BBh, 0CCh, 0DDh, 0EEh, 0FFh, 10h

b db 17h, 0DDh, 24h, 4Bh, 57h, 7Fh, 0CDh, 05h, 2Dh, 13h, 34h, 0BBh, 12h, 3Fh,

11h, 22h, 33h, 44h, 55h, 66h, 77h, 88h, 99h, 0AAh, 0BBh, 0CCh, 0DDh, 0EEh, 0FFh, 10h

r db 30 dup(?) ; Для результата резервируется 30 байт

.CODE

START:

; Сложение младших 4 байт

mov EAX, dword ptr a[0]

add EAX, dword ptr b[0]

mov dword ptr r[0], EAX

; Сложение следующих 4 байт с учетом переноса

mov EAX, dword ptr a[4]

adc EAX, dword ptr b[4]

mov dword ptr r[4], EAX

; Сложение следующих 4 байт с учетом переноса

mov EAX, dword ptr a[8]

adc EAX, dword ptr b[8]

mov dword ptr r[8], EAX

; Сложение следующих 4 байт с учетом переноса

mov EAX, dword ptr a[12]

adc EAX, dword ptr b[12]

mov dword ptr r[12], EAX

; Сложение следующих 4 байт с учетом переноса

mov EAX, dword ptr a[16]

adc EAX, dword ptr b[16]

mov dword ptr r[16], EAX

; Сложение следующих 4 байт с учетом переноса

mov EAX, dword ptr a[20]

adc EAX, dword ptr b[20]

mov dword ptr r[20], EAX

; Сложение следующих 4 байт с учетом переноса

mov EAX, dword ptr a[24]

adc EAX, dword ptr b[24]

mov dword ptr r[24], EAX

; Сложение оставшихся 2 байт с учетом переноса

mov AX, word ptr a[28]

adc AX, word ptr b[28]

mov word ptr r[28], AX

push 0

call ExitProcess

END START