Тема: Изучение команд арифметических и логических операций

Цель: изучение команд арифметических и логических операций, приобретение практических навыков при их использовании в различных вычислительных задачах.

Задание:

1. Составить программу для расчета заданного арифметического выражения. Длину и значение переменных A, B, C выбрать самостоятельно. Константы, заданные в выражении, использовать в кодовом сегменте.

2. Описать команды умножения и деления, используемые в программе на предмет длины операндов, участвующих в операции. Охарактеризовать длину результата и место его хранения.

3. Получить загрузочный модуль и протестировать выполнение программы в отладчике.

4. На основе составленной программы выполнить следующие действия:

− загрузить в аккумулятор число 1 и при помощи команд логических операций установить маскирующее слово, позволяющее определить заданную характеристику содержимого регистра DX;

− выполнить заданную проверку и ее результат сохранить в переменной RESULT, объявленной в сегменте данных.

5. Перекомпилировать загрузочный модуль и протестировать выполнение программы в отладчике.

6. Произвести расчет времени выполнения программы.

7. Сделать выводы.

**ХОД РАБОТЫ**

1. Введем следующие переменные A = 784, B = 50, C = 25. В качестве результирующей переменной будем использовать Res, изначально равной 0. Также заранее была инициализирована RESUALT = 0, необходимая для выполнения 4 пункта. Внесём в сегмент данных, получим следующий код:

.data

RESULT db 0

Res dw 0

A dw 784

B dw 50

S db 25

По условию 29 варианта необходимо составить программу для расчета следующего арифметического выражения:

Получим следующую функцию:

.code

start:

MOV AX, 216 ; 216

ADD AX, A ; AX=1000

MOV CX,AX; CX=1000

MOV AX, 12

MUL B; AX=600

MOV BX,AX; BX=600

MOV AX,4

MUL S; AX=100

SUB BX,AX ; BX=500

MOV AX,CX; AX=1000

DIV BX; AX = 2

ADD AX,133

ADD Res, AX

exit

end start

1. Проанализируем команды умножения, деления, сложения и вычитания:

1. ADD AX, A – прибавляет числу 216, хранящемуся в АX, число 784, располагающемуся в переменной A. В результате возвращает в AX = 03E8 (1000). Для освобождения регистра и дальнейшей работы с ним, перенесем полученный результат первой скобки в регистр CX.

2. MUL B – умножение значения регистра AX, в который заранее было записано число 12 (0C), на переменную B=50 (0032). В результате получим AX=600 (0258). Сохраним значение в регистре BX.

3. MUL S – умножение значения регистра AX, в который заранее было записано число 4, на переменную S=25 (19). В результате получим AX=100 (0064).

4. SUB BX,AX – вычитание значения из регистра BX=600 регистра AX=100. Получим BX=500 (01F4). Таким образом, была посчитана вторая скобка выражения.

5. DIV BX – деление значения регистра AX=1000 (03E8) на BX=500 (01F4). Получим AX= 2 (0002).

6. ADD AX,133- сложение AX= 2 (0002) и число 133 (85). В результате получим AX = 135 (0087).

7. ADD Res, AX – сложение переменной Res =0 (0000) и AX = 135 (0087). Получим конечный ответ Res = (0087).

Ручная проверка:

3. На рисунке 1 представлены значения, записанные на момент старта программы, а на рисунке 2 – окончания программы:

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 1 – Начало программы.

A=784 dump: 10 03

B=50 dump: 32 00

S=25 dump: 19

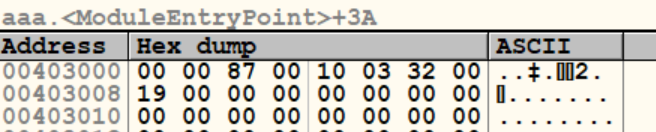




Рисунок 2 – Конец программы.

В результате получим значение Res = 0087, представленное в dump как 87 00 (подчеркнуто на Рисунке 2).

4. Необходимо определить наличие 0 в разряде 12. В 1 пункте была введена переменная RESULT = 0. Допишем следующий код:

start:

…

MOV AX, 0000100000000000B

MOV DX, 2266

;MOV DX, 1025

TEST AX,DX

JZ wr

MOV RESULT, 1

exit

wr:

MOV RESULT, 0

exit

Для решения данной задачи использовалась команда логической операции TEST. В регистр AX запишем число, у которого в 12 разряде стоит 1. В регистр BX для проверки будут записываться два числа: 2266 (содержащий 1 в 12 разряде) и 1025 (не содержащий). В переменную RESULT будет записываться 1, если стоит 1, в противном случае 0.

5. Скомпилируем и запустим программу:

Для DX=2266:

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Рисунок 3 – Регистры после присвоения им значений

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Рисунок 4 – Значение RESULT = 1

Для DX=1025:

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Рисунок 5 – Регистры после присвоения им значений

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 6 – Значение RESULT = 0

6. Время выполнения созданной программы:

1.MOV AX(регистр), 216 (операнд) – 4 такта

2.ADD AX(регистр), A(память слово) – 14 тактов

3.MOV CX(регистр), AX(регистр) – 2 такта

4.MOV AX(регистр), 12 ()операнд – 4 такта

5.MUL B (слово) – (128-143)+6 = 134-129 тактов

6.MOV BX(регистр), AX(регистр) – 2 такта

7.MOV AX(регистр),4(операнд) – 4 такта

8.MUL S (байт) – 70-77 такта

9.SUB BX (регистр), AX (регистр) – 3 такта

10.MOV AX(регистр), CX(регистр) – 2 такта

11.DIV BX(регистр 16) – 144-162 такта

12.ADD AX(регистр),133(операнд) – 4 такта

13.ADD Res(память слово), AX(регистр) – 14 такта

14. MOV AX(регистр), 0000100000000000B(операнд) – 4 такта

15. MOV DX(регистр), 2266(операнд) – 4 такта

16. TEST AX(регистр),DX(регистр) – 3 такта

17. JZ wr – 6 тактов

18. MOV RESULT (память), 1(операнд) -16 тактов

Общее время выполнения программы: 434–454 тактов.

**7.** Результаты:

В пункте 1 была проделана работа по созданию кода программы для вычисления арифметического выражения при помощи команд арифметических операций.

В задании 2 были описаны использованные в п.1 арифметические команды, их результаты и размеры.

В задании 3 была скомпилирована, выполнена и продемонстрирована работа созданной в п.1 и описанной п.2 программы.

В задании 4 была проделана работа по созданию кода программы для вычисления проверки при помощи команд логических операций.

В задании 5 был скомпилирован дополнительно созданный код и представлен результат работы программы.

В задании 6 был проведён расчёт времени выполнения созданного кода.

Вывод: были изучены работы команд арифметических и логических операций и написана программа для выполнения заданий, а также проведён анализ времени работы созданной программы.