Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ «САРАТОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

**Лабораторная работа №5 «Логические операции и управляющие структуры»**

Выполнил: студент 2 курса 241 группы  
направления [010500.62](http://www.sgu.ru/education/courses/010500-62-matematicheskoe-obespechenie-i)Математическое обеспечение и администрирование информационных систем (профиль Параллельное программирование)

факультета компьютерных наук и информационных технологий

Акимова Артемия Андреевича

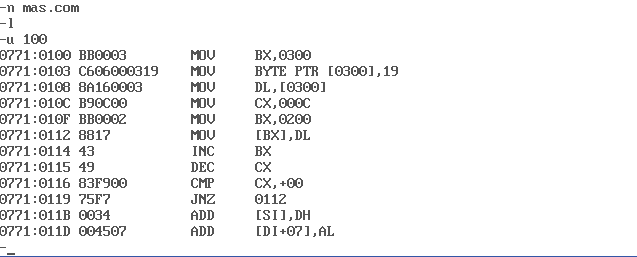
Саратов 2014

Вариант 73174.

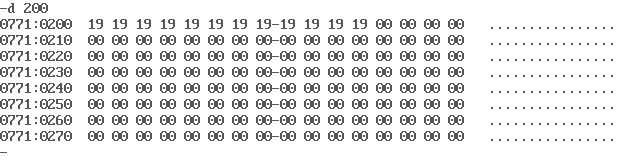
*Запустить в отладчике файл mas.com, начиная с адреса 100h. Выполнить программу в DEBUG. Программа записывает в память, начиная с адреса 200h, массив из 12 повторений числа 0019h.*

Для запуска файла необходимо:

* использовать команду N <полное имя файла>
* использовать команду загрузки L
* выполнить пошаговую трассировку или использовать команду G



Просмотрим память с адреса 200h:

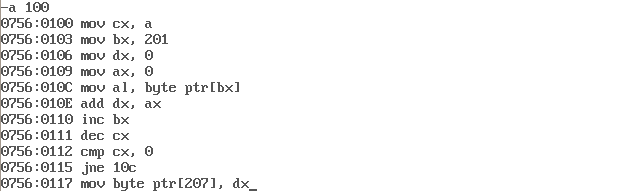


1. *Ввести в DEBUG программу (с адреса 100h), которая вычисляет сумму элементов массива согласно варианту и помещает результат вычислений в элемент массива с заданным номером. Выполнить программу в DEBUG.*

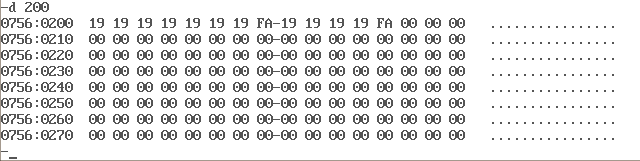
Для того чтобы сложить числа с 2 по 11, необходимо в регистр BX записать адрес первого суммируемого элемента, в CX количество повторений цикла, в DX будем накапливать сумму, поэтому изначально в этом регистре должен храниться 0. Само тело цикла должно содержать:

* операцию переноса содержимого регистра BX в регистр AL (команда mov al, byte ptr[bx])
* операцию прибавление к содержимому регистра DX содержимого регистра AL (команда add dx, ax)
* операцию увеличения содержимого регистра BX, чтобы последовательно переходить между суммируемыми элементами (команда inc bx).
* операцию уменьшения содержимого регистра CX(команда dec cx)
* само условие повторения цикла. В данном случае это команда cmp, которая проверяет не равно ли значение регистра CX нулю. Если равенство не имеет место на данном шаге, то тело цикла повторяется. Если же флаг нуля (ZF) принял значение 0, то управление передается на команду, следующую за командой условного перехода.
* команду помещения накопленной суммы на место заданного элемента. В данном случае нужная ячейка памяти располагается по адресу 207h (команда mov byte ptr[207], dx).

Таким образом, наша программа имеет следующий вид:



Результат работы программы совпадает с вычислениями и приведен далее:



Замечаем, что по адресу 207h содержится ответ **FA.**

1. *Ввести в DEBUG программу (с адреса 100h), которая вычисляет произведение элементов массива с номерами согласно варианту и помещает результат вычислений в элемент массива с заданным номером. Выполнить программу в DEBUG.*

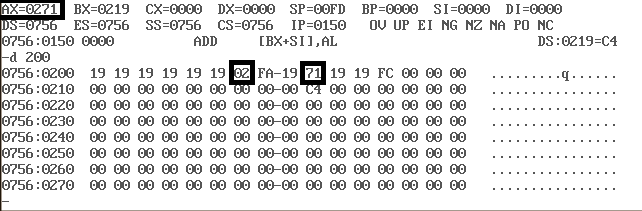
Для того, чтобы перемножить 3 и 2 элементы массива, необходимо:

* записать в al адрес первого операнда (команда mov al, byte ptr[202])
* записать в bl адрес первого операнда (команда mov bl, byte ptr[201])
* умножить содержимое регистра al на содержимое регистра bl (команда mul bl)
* старшую часть результата записать из регистра ah в ячейку памяти с адресом 206h (команда mov byte ptr[206], ah)
* младшую часть результата записать из регистра АX в ячейку памяти с адресом 209h (команда mov byte ptr[209], al)

Таким образом, программа имеет вид:



Выполняем программу при помощи команд t. Далее смотрим, поместился ли ответ в нужные ячейки памяти.



Замечаем, что по адресу 206h лежит старшая часть результата, по адресу 209h - младшая. Получаем ответ **0271.**

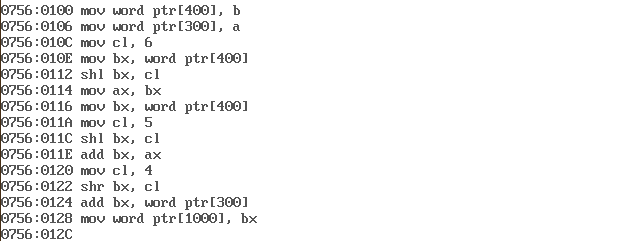
1. *Ввести в DEBUG программу (с адреса 100h), которая вычисляет выражение* ***F = Y оп1 КОН1 оп2 X оп3 КОН2****.*

Переменная Y хранится по адресу 0300h, X - по адресу 0400h. Результат вычисления выражения располагается по адресу 1000h.

В моем случае, необходимо найти значение функции F = a + 96 \* b / 16.

Я выбрала значения переменных **X = ah**, **Y = bh**.

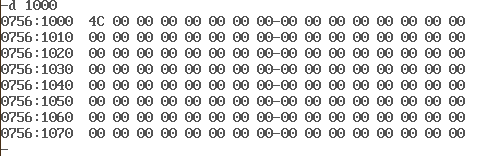
Программа целиком имеет вид:



Рассмотрим блоки умножения, деления и сложения.

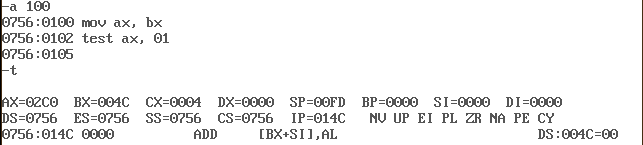
* помещаем в память по адресу 400h значение bh
* помещаем в память по адресу 300h значение ah
* в регистр СL помещаем необходимую степень двойки 6 (при переводе числа 96 в двоичную с.с. получаем 96 = )
* помещаем в регистр BХ значение, хранящееся по адресу 400h
* выполняем арифметический сдвиг влево на 6 бит.
* Перемещаем BX в AX
* в регистр СL помещаем необходимую степень двойки 5
* помещаем в регистр BХ значение, хранящееся по адресу 400h
* выполняем арифметический сдвиг влево на 5 бит.
* прибавляем к BX AX
* в регистр СL помещаем необходимую степень двойки 4(при переводе числа 96 в двоичную с.с. получаем 16 = )
* выполняем арифметический сдвиг вправо на 4 бит.
* прибавляем к значению регистра BX значение, хранящее по адресу 0300h
* помещаем значение регистра BX (результат вычисления) по адресу 1000h

Проверяем память с адреса 1000h, видим, что там содержится правильный ответ:



1. *Выполнить в DEBUG проверку полученного результата F на четность либо нечетность и знак числа.*

Для проверки на знак числа необходимо просмотреть какие флаги в какое значение установлены.



Флаг знака не принимает значение NG, значит число не отрицательное.

Код для проверки числа на четность выглядит следующим образом:

* переносим наш ответ из памяти в регистр АХ.
* выполняем команду TEST AX, 01. Если флаг нуля будет сброшен, то число нечетное. Если установлен, то четное.

Видим, что флаг нуля установлен, значит число четное. Это верно, так как 4Сh - четное число.