ГУАП

КАФЕДРА № 34

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Старший преподаватель |  |  |  | Жиданов К.А. |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ |
|  |
| по курсу: |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. | 3145 |  |  |  | Щербак А.А. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2021

**Задание 1:** написать на языке ассемблера программу, реализующую вычисления в целых числах по заданной формуле. Проверить корректность работы программы с помощью отладчика на заданных значениях.

**Ход выполнения:**

1. Индивидуальное условие задания:

**Выражение**: a\*b + c

**Значения**: [{16,7,12}, {-23,-13,-15}]

**Разрядность** **входных** **значений**: 8

**Код**: дополнительный

1. Вычислим значения выражения для заданных значений:

Для первого набора данных: 16\*7 + 12 = 124

Для второго набора данных: (-23) \* (-13) – 15 = 284

1. Перевод всех чисел в 16-ричную систему с учётом разрядности и способа кодирования:

Первый набор данных: Второй набор данных:

16­10 =1016  -2310 = E916

710 = 716 -1310 = F316

1210 = C16  -1510 = F116

1. Регистры для входных и выходных значений:

**Входные значения**: **Выходные значения**:

a – AL r - AX

b – BL

c – CL

1. Заносим программу в отладчик и проверяем значения во всех используемых регистрах
2. Проверяем результаты программы:

Первый набор данных:

После выполнения программы в регистре AX находится число 7C16, что соответствует числу 12410

Второй набор данных:

После выполнения программы в регистре AX находится число DD1C16 = 1101 1101 0001 11002

Во время выполнения программы произошло переполнение регистра AL поэтому нужное число лежит в битах с 0 по 8 регистра AX, т. е.

1 0001 11002 = 28410

Данные значения совпадают со значениями, полученными вручную.

**Код программы:**

MOV Al, 0x10 ;0xE9

MOV Bl, 0x7 ;0xF3

MOV Cl, 0xC ;0xF1

IMUL Bl

ADD Al, Cl

RET

**Задание 2:** написать на языке ассемблера программу, реализующую вычисления в числах с плавающей точкой по заданной формуле. Проверить корректность работы программы с помощью отладчика на заданных значениях

**Ход выполнения:**

1. Индивидуальное условие задания:

**Выражение**: a\*b + c

**Значения**: [{16,7,12}, {-23,-13,-15}]

**Разрядность** **входных** **значений**: 8

**Код**: дополнительный

1. Вычислим значения выражения для заданных значений:

Для первого набора данных: 16\*7 + 12 = 124

Для второго набора данных: (-23) \* (-13) – 15 = 284

1. Перевод всех чисел в формат чисел с плавающей точкой в соответствии с форматом IEEE 754

Первый набор данных: Второй набор данных:

16­10 =41800000 -2310 = C1B80000

710  = 40e00000 -1310 = C1500000

1210 = 00000000 -1510 = C1700000

1. Переменные, используемые в section. data для ввода и вывода данных:

Ввод данных: Вывод данных:

a: a r: r

b: b

c: c

1100001.011111\*26

1. Заносим программу в отладчик и проверяем значения во всех используемых регистрах
2. Проверяем результаты программы:

Первый набор данных:

После выполнения программы переменной r присваивается значение 42F8000016, которое при обратном переводе даёт 12410

Второй набор данных:

После выполнения программы переменной r присваивается значение 438E000016, которое при обратном переводе даёт 28410

Данные значения совпадают со значениями, полученными вручную

**Код программы:**

FLD DWORD [c]

FLD DWORD [a]

FLD DWORD [b]

FMUL

FADD

FST DWORD [r]

RET

section .data

a: DD 0xC1B80000 ;0x41800000

b: DD 0xC1500000 ;0x40e00000

c: DD 0xC1700000 ;0x41400000

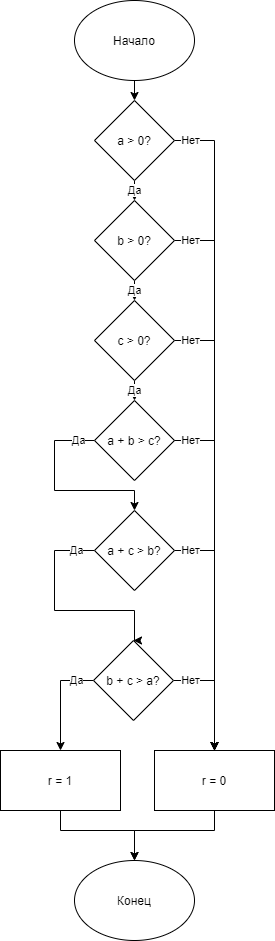
r: DD 0x00000000

**Задание 3:**

Написать на языке ассемблера программу, реализующую сравнение целых чисел с учётом способа кодирования. Проверить корректность работы программы с помощью отладчика на заданных значениях.

**Ход работы:**

1. Индивидуальный вариант в соответствии со списком: может ли существовать треугольник с указанными длинами сторон

1. Составим алгоритм решения задачи:

Для хранения промежуточных сумм двух сторон будем использовать регистр DX, обнуляя его после каждого сравнения

1. Проверим корректность работы алгоритма на тестовых наборах данных

Набор 1: {16,7,12}

16 > 0, 7 > 0, 12 > 0

16 + 7 > 12

16 + 12 > 7

12 + 7 > 16

r = 1

Набор 2: {-23,-13,-15}

-23 < 0

r = 0

С тестовыми значениями алгоритм работает корректно

1. Регистры для входных и выходных значений:

**Входные значения**: **Выходные значения**:

a – AL r – DX

b – BL

c – CL

1. Заносим программу в отладчик
2. Подаём на вход тестовые наборы, программа работает корректно, полученные данные совпадают с теми, что были получены вручную
3. На первом наборе данных программа выдаёт значение 1, на втором наборе данных – 0

**Код программы:**

XOR EDX, EDX

MOV AX, 23

MOV BX, 13

MOV CX, 15

CMP AX, 0

JL no

CMP BX, 0

JL no

CMP CX, 0

JL no

ADD DX, AX

ADD DX, BX

CMP DX, CX

JA s2

JMP no

s2:

XOR EDX, EDX

ADD DX, AX

ADD DX, CX

CMP DX, BX

JA s3

JMP no

s3:

XOR EDX, EDX

ADD DX, CX

ADD DX, BX

CMP DX, AX

JA yes

JMP no

no:

XOR EDX, EDX

MOV EDX, 1

JMP quit

yes:

XOR EDX, EDX

JMP quit

quit:

RET