**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра математического обеспечения и применения ЭВМ**

**ОТЧЕТ**

**по лабораторной работе №6**

**по дисциплине «Организация ЭВМ и систем»**

# Тема: ТИПЫ АДРЕСАЦИИ.

|  |  |
| --- | --- |
| Студент гр. 3381 | Лутфулин Д.А. |
| Преподаватель | Молодцев Д.А. |

Санкт-Петербург

2024

#### ****Цель**** работы

Создать программу на ассемблере, которая выполняет вычисление математического выражения, используя значения, размещённые в регистрах a2, a3 и a4. Результат вычислений должен быть сохранён в регистре a0.

#### ****Задание****

Ваше задание:

Требуется написать программу, которая использует разные режимы адресации для вычислений по массиву данных в памяти. Результатом выполнения вашей программы будет измененный массив в памяти.

В качестве исходных данных дается стартовый адрес в памяти для хранения массива, количество элементов в массиве и формула для требуемых вычислений (Вычисления включают изменения каждого элемента массива в зависимости от условия).

При автоматической проверке вашей программы исходные данные располагаются в регистрах следующим образом:

- a1 - адрес памяти, где расположен массив

- a2 - количество элементов в массиве

Считайте, что массив уже инициализирован и заполнен данными.

Ваша программа должна иметь следующую структуру:

.globl solution

solution:

# при старте данной метки ваша программа должна выполнить

# необходимые вычисления и изменить элементы массива согласно ветке

условия и формуле в ней

ret

Доступ к массиву (чтение, изменение) должен выполняться из памяти.

Формула для вычислений будет выведена ниже (arr[i] - элемент массива, считаем что arr[-1] == 0):

ЕСЛИ ((arr[4] & arr[5] + arr[6]) > 320)

ТО (arr[i] = arr[i - 1] + 35)

ИНАЧЕ (arr[i] = arr[i] | 79)

Ваш seed = 5329281024

#### ****Выполнение**** работы

В начале программы инициализируются регистры. В a4 копируется адрес массива, в a5 — количество элементов массива. Регистр t6 загружает первый элемент.

Далее в цикле:

Загружаются элементы массива с индексами 4, 5 и 6.

Проверяется условие: если (arr[4] & (arr[5] + arr[6])) > 320, то выполняется замена arr[i] на arr[i-1] + 35.

Если условие не выполняется, то arr[i] меняется с помощью побитовой операции OR с 79.

После выполнения вычислений для текущего элемента, он сохраняется обратно в массив. Индекс массива увеличивается, и цикл продолжается до конца массива.

Когда все элементы обработаны, программа завершает работу.

Виды адресации, использованные в программе.

Регистровая:

mv t5, x0

mv a4, a1

mv a5, a2

mv t6, x5

mv t5, t6

Базовая:

ld t6, 0(a1)

ld t0, 32(a1)

ld t1, 40(a1)

ld t2, 48(a1)

sd t6, 0(a4)

ld t6, 0(a4)

Непосредственная адресация:

li t4, 320

ori t6, t6, 79

addi t6, t6, 35

addi a5, a5, -1

addi a5, a5, -1

addi a4, a4, 8

Адресация относительно счётчика:

bgt t1, t4, condition\_true

j loop\_end

beqz a5, end

j loop

#### ****Вывод.****

Разработанная программа корректно обрабатывает заданное условие и

редактирует массив, используя разные режимы адресации

# Приложение А Исходный код программы

.globl solution

solution:

mv t5, x0 #a[i-1]

ld t6, 0(a1) #a[i]

mv a4, a1 #адрес памяти где расположен массив, будем прибавлять 4

mv a5, a2 #колво элементов в массиве

loop:

ld t0, 32(a1) # t0 = arr[4]

ld t1, 40(a1) # t1 = arr[5]

ld t2, 48(a1) # t2 = arr[6]

add t1, t1, t2 # t3 = arr[5] + arr[6]

and t1, t0, t1 # t3 = arr[4] & arr[5] + arr[6]

li t4, 320 # t4 = 320

bgt t1, t4, condition\_true # Если t4 > 320, перейти в метку condition\_true

# Если условие не выполнено (arr[4] & (arr[5] + arr[6]) <= 320)

ori t6, t6, 79

j loop\_end

condition\_true:

# Если условие выполнено (arr[4] & (arr[5] + arr[6]) > 320)

mv t6, t5

addi t6, t6, 35

loop\_end:

sd t6, 0(a4) # arr[i] = t6

addi a5, a5, -1 # a5 -= 1

beqz a5, end #Если последний элемент, завершаем программу

mv t5, t6 # arr[i-1] := arr[i]

addi a4, a4, 8 # i += 1

ld t6, 0(a4) # t6 := arr[i]

j loop

end:

ret # Выход из функции