Учреждение образования
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

Факультет компьютерных систем и сетей
Кафедра информатики
Дисциплина: архитектура вычислительных систем

Отчет по лабораторной работе №3

Ассемблер RISC-V

Выполнил: студент: гр. 053506
Слуцкий Никита Сергеевич

Проверил: ст. преподаватель
Шиманский Валерий Владимирович

Минск 2022

СОДЕРЖАНИЕ

Введение

Цели данной работы:

1. Получить представление о том, как перевести код на языке Си в RISC-V.
2. Научиться писать функции RISC-V, по правилам о соглашении вызова процедур.

Постановка задачи и результаты выполнения

Задания 1 - 3: Подключение файлов к Venus, Знакомство с Venus, Перевод с языка C на RISC-V

Это задание призвано ознакомить с процессом и инструментами разработки. С инструментами ознакомился, необходимые результаты получил. Для разработки предпочёл расширение “RISC-V Venus Simulator” для Microsoft Visual Studio Code. Так как это ознакомительные задания, их формулировка и результаты опускаются

Задание 4: Задание по вариантам

Номер в журнале 21, Вариант 1

В этом задании предстоит написать RISC-V код, который вычисляет заданную математическую функцию. Необходимо написать две функции решающие предложенную задачу, одна должна работать итеративно (через цикла), другая рекурсивно (вызываю саму себя). Параметры для функций задаются. Писать свой код в ex4.s.

ВАРИАНТ 1: Реализовать функцию вычисления факториала. Эта функция принимает один целочисленный параметр n и возвращает n!

Код программы и результат, например, для рекурсивной функции и параметра 5:

```
.globl iterative
.globl recursive
.data
    n: .word 8
    m: .word 2
.text
    main:
        jal ra, tester

        addi a1, t3, 0
```

```
    addi a0, zero, 1
    ecall # Print Result

    addi a1, zero, '\n'
    addi a0, zero, 11
    ecall # Print newline

    addi a0, zero, 10
    ecall # Exit

tester:
    addi a1, zero, 5
    addi a0, zero, 5

    sw ra, 8(sp)
    #jal iterative
    jal recursive
    addi t3, a0, 0
    lw ra, 8(sp)

    jr ra

iterative: # a1 ---> argument
    mv t0, zero # counter
    addi a0, zero, 1 # response

    count_factorial_iterative:
        beq t0, a1, finished_counting_factorial_iterative
        addi t0, t0, 1
        mul a0, a0, t0

        j count_factorial_iterative

    finished_counting_factorial_iterative:
        jr ra

recursive: #--> a0 - argument
    addi sp, sp, -8

    sw a0, 4(sp)
    sw ra, 0(sp)

    addi t0, zero, 1

    bgt a0, t0, return

    addi a0, zero, 1
    addi sp, sp, 8
    jr ra

    return:
        addi a0, a0, -1
        jal recursive
```

```
lw t1, 4(sp)
lw ra, 0(sp)

addi sp, sp, 8
mul a0, t1, a0

jr ra
```

```
---------------------------------
Starting program d:\Other\HomeWork

120
Exited with error code 0
Stop program execution!
---------------------------------
```

Рисунок 1. Результат рекурсивной программы факториала для значения 5

Задание 5 Практика работы с массивами

Рассмотрим дискретно-значную функцию f, определенную на целых ограниченном множестве целых чисел. Вот определение функции (по вариантам, в остальных точках функция не определена и возвращает -1):

```
# V.1
f(-3) = 6   |
f(-2) = 61  |
f(-1) = 17  |
f(0) = -38  |
f(1) = 19   |
f(2) = 42   |
f(3) = 5    |
```

Рисунок 2. Значения функции и допустимых значений для моего варианта

Реализовать функцию в файле discrete_fn.s в RISC-V с условием, что код **НЕ** должен использовать инструкции ветвления и/или перехода!

В виду громоздкости предложенного шаблона я немного видоизменил с нуля программу. Полученная программа на значения X = -3, -2, -1, …, 3 выдаёт корректные значения функции 6, 61, 17, -38, 19, 42, 5

Код реализованной программы:

```
.globl function
.data
    data_array: .word   6, 61, 17, -38, 19, 42, 5, -1
```

```
.text
    main:
        addi      a2, zero, 0 # test value to check: -3
        la        a3, data_array # copied address of data_array to a3

        jal       ra, function

        jal       ra, print_int

        #return
        addi      a0, zero, 10
        ecall

    # a2 значение для которого мы хотим вычислить функцию f
    # a3 адрес выходного ("output") массива, содержащего все допустимые варианты.
    # a0 - function's response
    function:
        addi      t0, a2, 3 # t0 = a2 + 3 (shift += 3, because I have possible args: range(-3, 3, 1)
        slli      t0, t0, 2 # t0 *= 4
        add       t0, t0, a3 # t0 = address of arr[counter]
        lw        t1, 0(t0)
        mv        a0, t1 # response = t1 (a0 = t1)

        jr        ra # return
    print_int: # from a0
        mv      a1, a0
        addi    a0, zero, 1
        ecall

        jr      ra

    print_newline:
        addi    a1, zero, '\n'
        addi    a0, zero, 11
        ecall

        jr      ra
```

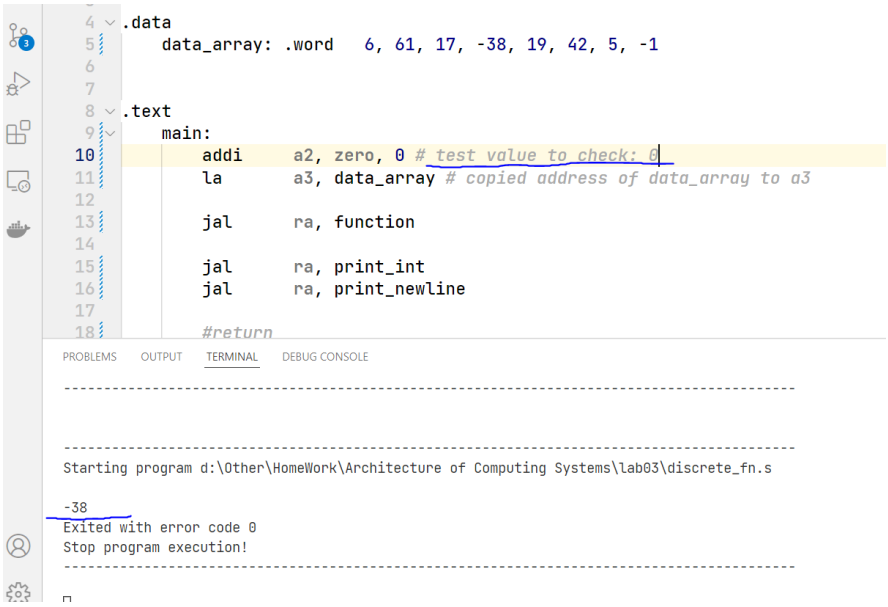

Рисунок 3. Вывод программы, например, для значения аргумента X = 0

Выводы

В результате выполнения лабораторной работы №3 на практике были изучены основы при работе с RISC-V архитектурой с использованием RISC-V ассемблера. Использованы инструкции, регистры данной архитектуры. Выполнены 2 задания согласно моему первому варианту. Цели лабораторной работы № 3 могу считать достигнутыми.

Литература

Харрис, Дэвид; Харрис, Сара «Цифровая схемотехника и архитектура компьютера. RISC-V»

Официальный сайт RISC-V