Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ЭВМ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ (ИУ7)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.04 Программная инженерия**

ОТЧЕТ

по лабораторной работе № 2

Название: Изучение принципов работы микропроцессорного ядра RISC-V

Дисциплина: Архитектура ЭВМ

Студент ИУ7И-56Б _____ Андрич К.
(Группа) (Подпись, дата) (И.О. Фамилия)

Преподаватель _____ А.Ю. Попов
(Подпись, дата) (И.О. Фамилия)

Москва, 2022

Цель работы

Ознакомление с принципами функционирования, построения и особенностями архитектуры суперскалярных конвейерных микропроцессоров, а также знакомство с принципами проектирования и верификации сложных цифровых устройств с использованием языка описания аппаратуры SystemVerilog и ПЛИС.

Ход работы

Задание 1

Текст программы по индивидуальному варианту

```
.section
.text
.globl _start;
len = 8 #Размер массива
enroll = 1 #Количество обрабатываемых элементов за одну итерацию
elem_sz = 4 #Размер одного элемента массива

_start:
la x1, _x
addi x20, x1, elem_sz*(len-1) #Адрес последнего элемента

lp:
lw x2, 0(x1)
addi x1, x1, elem_sz*enroll #!
add x31, x31, x2
bne x1, x20, lp
addi x31, x31, 1
lp2: j lp2

.section .data
_x:  .4byte 0x1
.4byte 0x2
```

```
.4byte 0x3
.4byte 0x4
.4byte 0x5
.4byte 0x6
.4byte 0x7
.4byte 0x8
```

Дизассемблерный код программы:

```
SYMBOL TABLE:
80000000 l    d  .text  00000000 .text
80000024 l    d  .data  00000000 .data
00000000 l    df *ABS*  00000000 main.o
00000008 l       *ABS*  00000000 len
00000001 l       *ABS*  00000000 enroll
00000004 l       *ABS*  00000000 elem_sz
80000024 l       .data  00000000 _x
8000000c l       .text  00000000 lp
80000020 l       .text  00000000 lp2
80000000 g       .text  00000000 _start
80000044 g       .data  00000000 _end

Disassembly of section .text:

80000000 <_start>:
80000000:       00000097              auipc   x1,0x0
80000004:       02408093              addi    x1,x1,36 # 80000024 <_x>
80000008:       01c08a13              addi    x20,x1,28

8000000c <lp>:
8000000c:       0000a103              lw      x2,0(x1)
80000010:       00408093              addi    x1,x1,4
80000014:       002f8fb3              add     x31,x31,x2
80000018:       ff409ae3              bne     x1,x20,8000000c <lp>
8000001c:       001f8f93              addi    x31,x31,1

80000020 <lp2>:
```

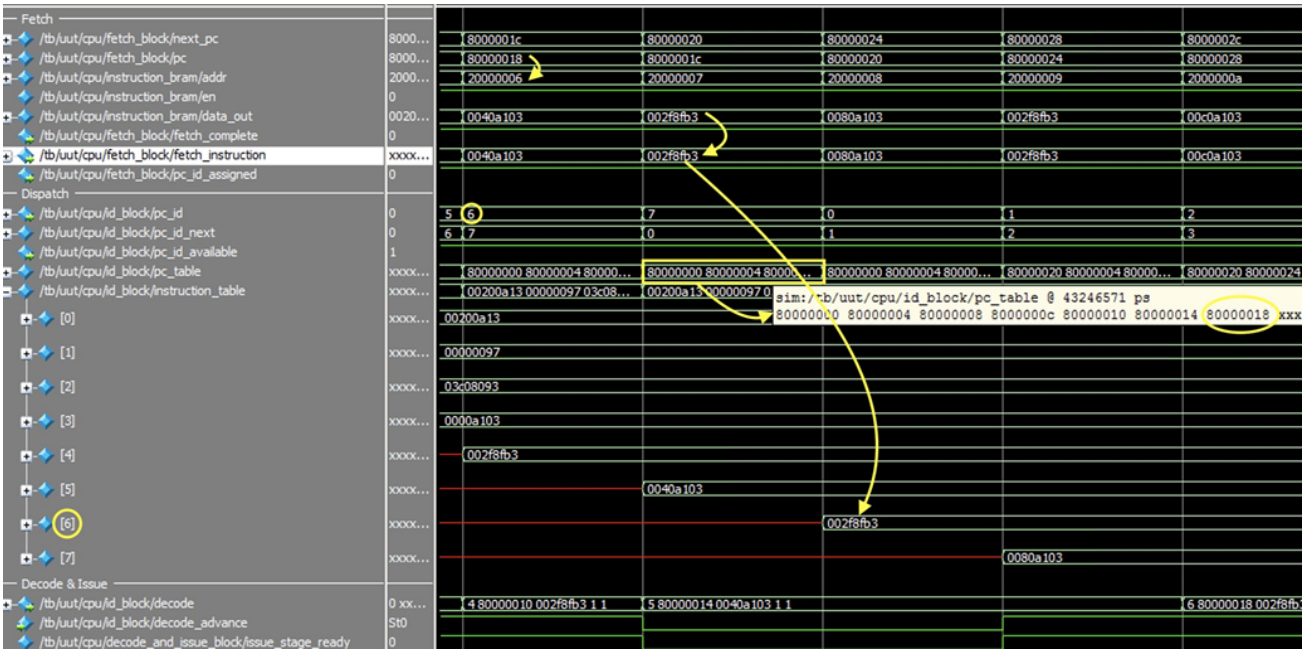
```
80000020:       0000006f            jal     x0,80000020 <lp2>

Disassembly of section .data:

80000024 <_x>:
80000024:       0001                c.addi  x0,0
80000026:       0000                unimp
80000028:       0002                0x2
8000002a:       0000                unimp
8000002c:       00000003            lb      x0,0(x0) # 0 <enroll-0x1>
80000030:       0004                c.addi4spn      x9,x2,0
80000032:       0000                unimp
80000034:       0005                c.addi  x0,1
80000036:       0000                unimp
80000038:       0006                0x6
8000003a:       0000                unimp
8000003c:       00000007            0x7
80000040:       0008                c.addi4spn      x10,x2,0
```

Псевдокод:

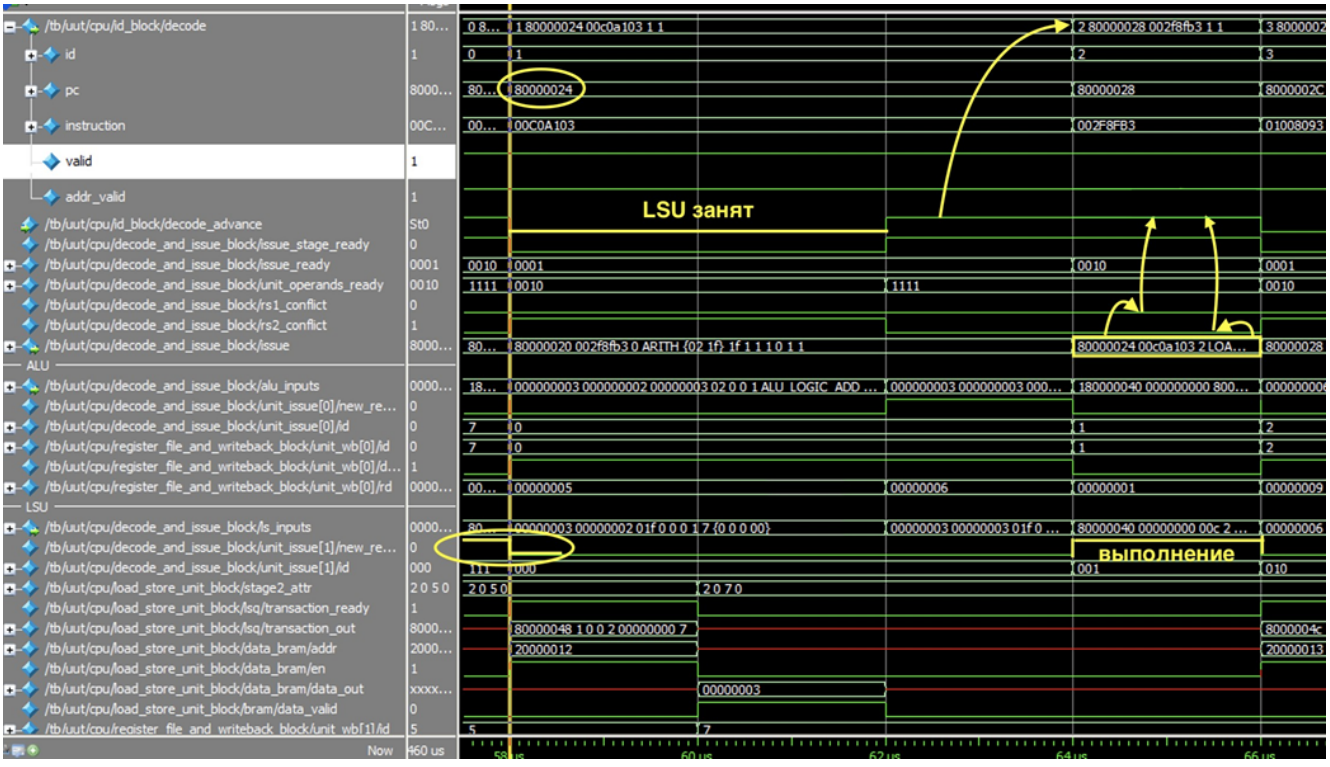
1. Инициализация массива _x, состоящего из 8 элементов, значениями от 1 до 8 включительно. Каждое значение по 4 байта.
2. Инициализация параметров алгоритма:
    len = 8
    enroll = 1
    elem_sz = 4
3. Установка указателя x1 на начало массива:
    x1 = _x
4. Вычисление указателя на последний элемент массива:
    x20 = x1 + (len −- 1) * size
5. Цикл: пока x1 не равно x20
    5.1. x2 = x1[0]
    5.2. x1 = x1 + elem_sz * enroll
    5.3. x31 = x31 + x2
6. x31 = x31 + 1

$$x31 = \sum_{i=1}^{7} i + 1 = 29.$$

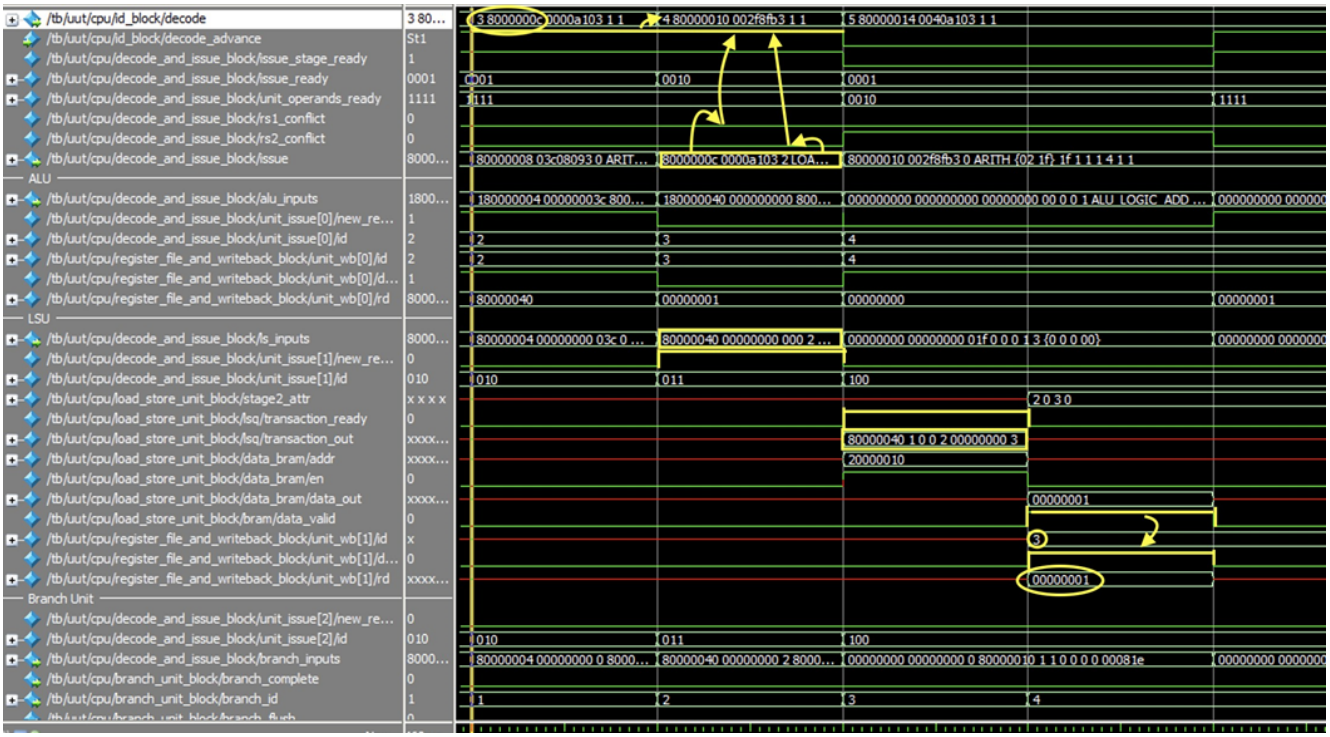
Задание 2



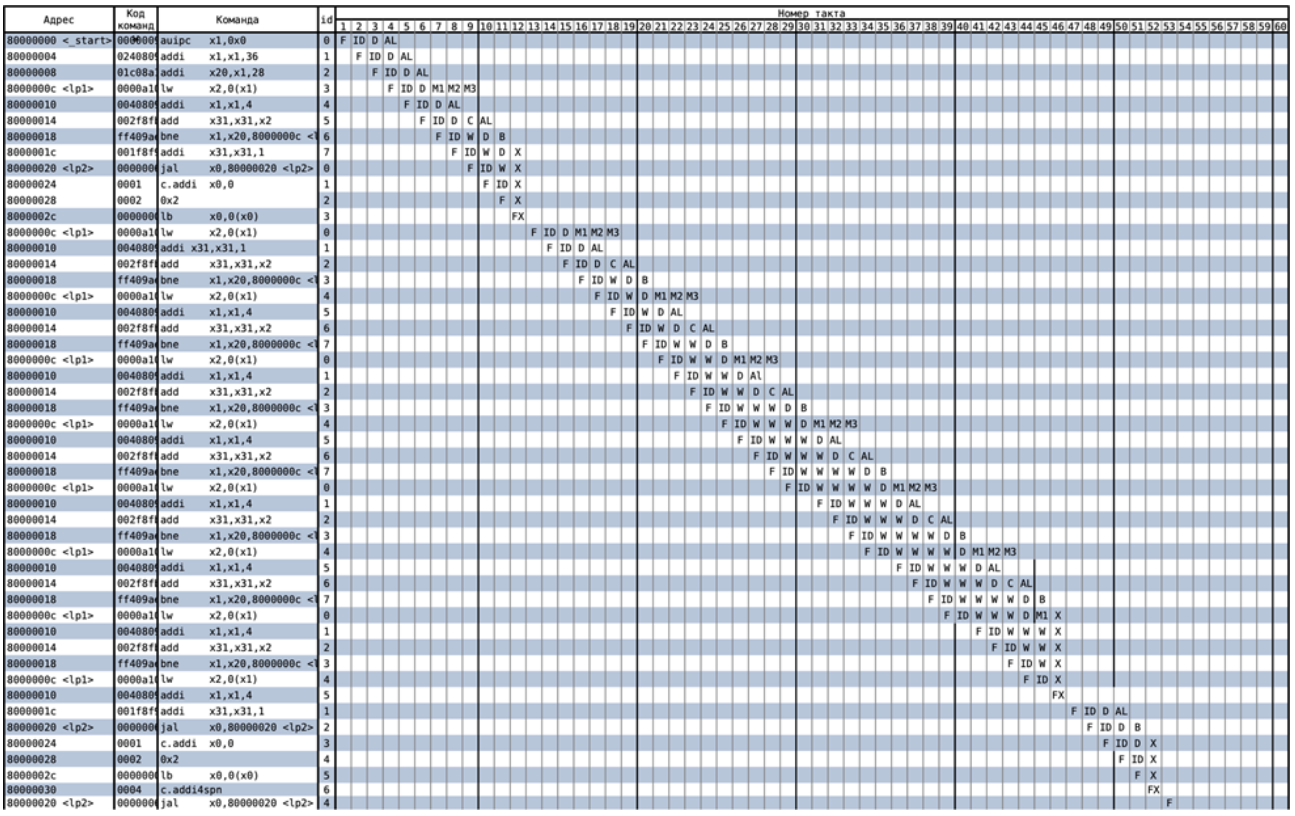
Задание 3



Задание 4



Задание 5



Вывод

В данной лабораторной работе было проведено ознакомление с архитектурой ядра Taiga, а именно с порядком работы вычислительного конвейера: изучены команды RV32I, рассмотрены действия, выполняемые на каждой стадии конвейера, и данные, передаваемые между ними. После ознакомления с теоретической стороной вопроса, был выполнен разбор этапов выполнения программы на симуляции процессора с набором инструкций RV32I. После ее анализа были сделаны выводы, что оптимизация не требуется. В итоге, теоретические знания о порядке исполнения программ на процессорах с RISC архитектурой были закреплены на практике. Таким образом все поставленные задачи решены, основная цель работы достигнута.