Национальный исследовательский университет ИТМО

Факультет программной инженерии и компьютерной техники

Лабораторная работа. Реализация синтезированного решения в baseline-проекте (умножение матриц)

По дисциплине «Процессоры и Функциональные Ускорители»

Выполнил:
Булыгин А. А.
Группа: P42192

Санкт-Петербург
2022 г.

1 Цель работы

Изучить работу процессоров с архитектурой RISC-V и способы их модификаций. Выполнить практическое задание с учебным RISC-V микроконтроллером.

2 Задание

а) запустить baseline-проект Sigma MCU на FPGA-плате, убедиться в успешном прохождении тестов;

б) разработать программную реализацию прикладной функции для Sigma MCU, измерить производительность на различных конфигурациях;

в) разработать и интегрировать аппаратный ускоритель прикладной функции с помощью системной шины:

- проанализировать прикладную функцию и разработать потактовый план её вычисления в аппаратуре (с векторным и/или конвейерным параллелизмом)

- разработать микроархитектурную (блочную) диаграмму ускорителя прикладной функции (в терминах регистров, блоков памяти и “облаков” комбинационной логики)

- написать и верифицировать аппаратный ускоритель на SystemVerilog

- интегрировать разработанный ускоритель с Sigma MCU (через КСР, отображенных на память), написать программный baremetal драйвер для разработанного ускорителя (передача данных и запуск блока)

- протестировать разработанный ускоритель на FPGA-плате (с ПК)

г) разработать и интегрировать аппаратный ускоритель прикладной функции с помощью интерфейса расширения системы команд.

3 Описание инструментов

А) Рабочая плата: Digilent NEXUS4-DDR (Artix-7 FPGA):

Плата для разработки - ПЛИС, разработанная на основе семейства Xilinx Artix-7 FPGA. Представляет собой готовую к использованию платформу для разработки цифровых схем. Поддерживается пакетом Xilinx Vivado Design Suite

Интерфейсы: 10/100 Ethernet, USB, UART, JTAG, VGA.

Logic Slices – 15850;

Block RAM – 4860 Kb;

DDR2 Memory – 128 MiB;

Clock Titles – 6;

DSP Slices – 240;

QSPI Flash – 16 MB;

Input Voltage – 7-15 V.

Б) Сапр для симуляции и имплементации: Vivado Design Suite 2019.2 – инструмент реализации решений на уровне регистровых передач с помощью языков описания аппаратуры.

В) Компилятор для процессора с архитектурой RISC-V.

Г) RISC-V микроконтроллер Sigma MCU.

4 Выполнение задания

4.1 Запуск baseline–проекта

Запуск baseline-производится в виде генерации прошивки для ПЛИС и загрузки данной прошивки на рабочую плату (далее прошивка). Тестирование работы производилось с помощью программы на языке python с использованием системной шины (далее python скрипт). Python скрипты для тестирования работы приложений и тестирования

ограничений на процессоре, а также результаты их выполнения приведены в листингах 1-4 приложения А.

4.2 Разработка программной реализации

В соответствии с вариантом было необходимо разработать программу реализации алгоритма умножения матриц. Умножение матриц производилось в соответствии со следующим выражением:

$$C = A \times B; \quad C_{ij} = \sum_{k=0}^{p} A_{ik} * B_{kj}, \text{ где } i = 0,\dots, m,\ j = 0,\dots, n, \quad (1)$$

где

A, B - входные матрицы,

n - количество столбцов первой матрицы,

p - количество строк первой матрицы и столбцов второй матрицы,

m - количество строк второй матрицы,

C - результирующая матрица.

Код разработанной программы содержится в листинге 5 приложения А. В качестве конфигураций процессора использовались реализации процессора с архитектурой RISC-V с конвейером от 1 до 6 стадий результаты тестирования приведены в таблице 1.

Для верификации работы программы на микроконтроллере и последующих написанных блоков использовался скрипт, код которого приведен в листингах 14-16 приложения А. Вывод скрипта в консоль приведен в листинге 17 приложения А.

Таблица 1 - Результат тестирования производительности программы умножения матриц

Количество стадий конвейера	Время выполнения, мкс
1	181,904
2	107,431
3	105,875
4	63,407
5	50,628
6	51,712

Из таблицы следует, что для исполнения программы умножения матриц выделение в отдельные стадии обратной записи результата в регистры и запроса в память инструкций не даёт прироста. В качестве примера измерения на рисунке 1 приведен результат симуляции для одностадийного конвейера.

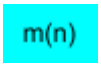
Для тестирования программы на целевой плате была сгенерирована прошивка и написан python скрипт, код которого представлен в листинге 6 приложения А. Результат тестирования представлен в листинге 13 приложения А.

Рисунок 1 - Результат симуляции выполнения программы умножения матриц на одностадийном конвейере с архитектуре RISC-V

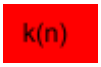
4.3 Разработка программного ускорителя

В качестве целевой функции ускорителя была выбрана операция умножения матриц 8x8, так как при такой размерности реализация умножителя с конвейерным параллелизмом позволяет проще реализовать расписание для блоков. Так как ресурсы целевой платы не позволили реализовать умножение матриц 8x8 в векторном виде было принято решение реализовать вычисления с конвейерным параллелизмом. В таблице 2 представлен потактовый план работы ускорителя. Типы операций в таблице определены следующими цветами:

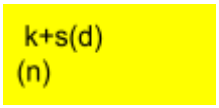
n - запись входных матриц в регистры (n - порядковый номер записи),



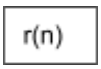
- расчёт умножений для элементов m-й строки итоговой матрицы с порядковым номером n,



- запись результатов умножений 1 элемента 1 матрицы на 1 элементы столбцов 2 матрицы для k-ой строки итоговой матрицы с порядковым номером n,



- сложение результатов умножений s-го элемента d-ой строки 1 матрицы на s-е элементы столбцов 2 матрицы с результатами умножения записанных в к-ом аккумуляторе для итоговой матрицы с порядковым номером n,



- запись r-й итоговой суммированной строки для итоговой матрицы с порядковым номером n в результирующий регистр.

Таблица 2 - Потактовый план ускорителя умножения матриц 8x8

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	1(1)	1(1)	1+2(1) (1)	1+3(1) (1)	1+4(1) (1)	1+5(1) (1)	1+6(1) (1)	1+7(1) (1)	1+8(1) (1)	1(1)		4(2)	4(2)	4+2(4) (2)	4+3(4) (2)	4+4(4) (2)	4+5(4) (2)	4+6(4) (2)
		2(1)	2(1)	2+2(2) (1)	2+3(2) (1)	2+4(2) (1)	2+5(2) (1)	2+6(2) (1)	2+7(2) (1)	2+8(2) (1)	2(1)		5(2)	5(2)	5+2(5) (2)	5+3(5) (2)	5+4(5) (2)	5+5(5) (2)
			3(1)	3(1)	3+2(3) (1)	3+3(3) (1)	3+4(3) (1)	3+5(3) (1)	3+6(3) (1)	3+7(3) (1)	3+8(3) (1)	3(1)		6(2)	6(2)	6+2(6) (2)	6+3(6) (2)	6+4(6) (2)
				4(1)	4(1)	4+2(4) (1)	4+3(4) (1)	4+4(4) (1)	4+5(4) (1)	4+6(4) (1)	4+7(4) (1)	4+8(4) (1)	4(1)		7(2)	7(2)	7+2(7) (2)	7+3(7) (2)
					5(1)	5(1)	5+2(5) (1)	5+3(5) (1)	5+4(5) (1)	5+5(5) (1)	5+6(5) (1)	5+7(5) (1)	5+8(5) (1)	5(1)		8(2)	8(2)	8+2(8) (2)
						6(1)	6(1)	6+2(6) (1)	6+3(6) (1)	6+4(6) (1)	6+5(6) (1)	6+6(6) (1)	6+7(6) (1)	6+8(6) (1)	6(1)		1(3)	1(3)
							7(1)	7(1)	7+2(7) (1)	7+3(7) (1)	7+4(7) (1)	7+5(7) (1)	7+6(7) (1)	7+7(7) (1)	7+8(7) (1)	7(1)		2(3)
								8(1)	8(1)	8+2(8) (1)	8+3(8) (1)	8+4(8) (1)	8+5(8) (1)	8+6(8) (1)	8+7(8) (1)	8+8(8) (1)	8(1)	
								2	1(2)	1(2)	1+2(1) (2)	1+3(1) (2)	1+4(1) (2)	1+5(1) (2)	1+6(1) (2)	1+7(1) (2)	1+8(1) (2)	1(2)
										2(2)	2(2)	2+2(2) (2)	2+3(2) (2)	2+4(2) (2)	2+5(2) (2)	2+6(2) (2)	2+7(2) (2)	2+8(2) (2)
											3(2)	3(2)	3+2(3) (2)	3+3(3) (2)	3+4(3) (2)	3+5(3) (2)	3+6(3) (2)	3+7(3) (2)
																3		

Как видно из плана интервал инициации ускорителя составляет 8 тактов, а задержка составляет 18 тактов. На рисунках 2 и 3 представлена блочная диаграмма ускорителя.

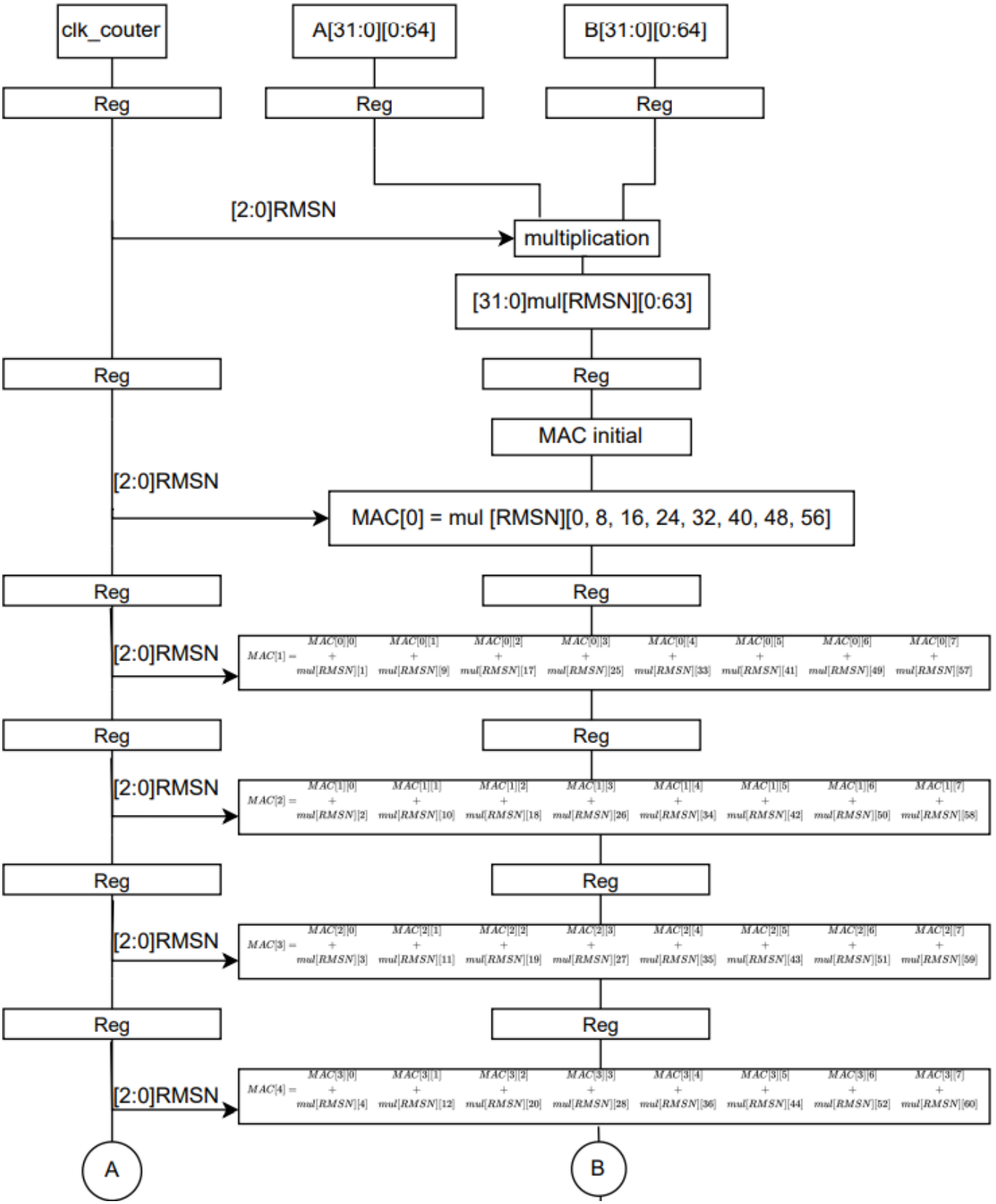


Рисунок 2 - Начало диаграммы матричного умножителя

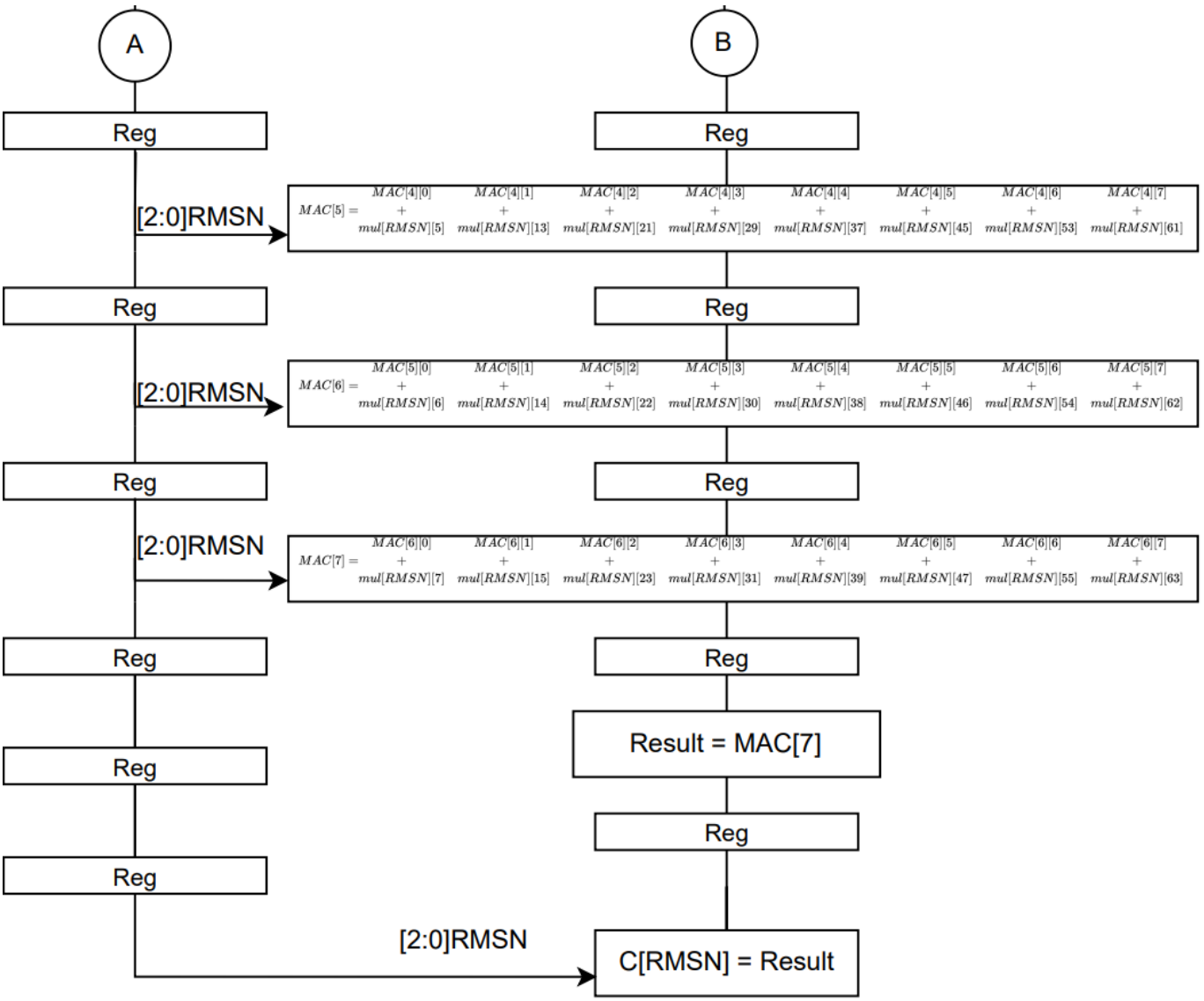


Рисунок 3 - Окончание диаграммы матричного умножителя

Сокращениями на диаграмме обозначены:

RMSN - номер строки результирующей матрицы (result matrix string number);

MAC - матричный аккумулятор.

Форматы входных и выходных данных представлены в таблице 3.

Таблица 3 - Интерфейс блока матричного умножителя

Название	Адрес	Размер	тип сигнала	Назначение
Clear	0xc0000000	1 бит	in	Сброс регистров данных внутри блока
Start	0xc0000004	1 бит	in	Разрешение вычислений внутри блока
A	0xc0000008	64 разрядный одномерный 32-битный массив	in	Входная первая матрица
B	0xc0000108	64 разрядный одномерный 32-битный массив	in	Входная вторая матрица
C	0xc0000208	64 разрядный одномерный 32-битный массив	out	Выходная итоговая матрица
Done	0x90000000	1 бит	out	Сигнал готовности результата

Согласно диаграмме был написан блок, код которого представлен в листинге 7 приложения А. Также Блок был подключен к системной шине Sigma MCU, код верхнего уровня которого представлен в листинге 8 приложения А. Для запуска блока был написан baremetal драйвер, код которого представлен в листинге 9 приложения А.

Верификация блока производилась скриптами, использующимися в разделе 4.2 (стр. 4).

Измерение времени работы драйвера для расчёта умножения матриц приведено на рисунке 4.

Рисунок 4 - Измерение работы драйвера для блока умножения матриц для процессора с одностадийным конвейером

Из сравнения результатов, полученных на рисунках 3 и 4, следует, что ускоритель справляется с вычислением одного экземпляра матрицы 8 на 8 быстрее на 137 мкс, то есть прирост производительности составил 405%. При этом стоит учесть, что основное время выполнения пришлось на выгрузку и загрузку данных. Если учесть, что исходя из плана в таблице 2, задержка расчета одной матрицы составляет 18 тактов, то в процентном соотношении расчет занял:

$$Comp = \frac{Clock_{Comp}}{T_{Total}} * T_{Clock} = \frac{18}{44{,}886612*10^{-6}} * \frac{1}{70*10^{6}} = 0,57\%. \quad (2)$$

Для тестирования блока на целевой плате была сгенерирована прошивка и написан python скрипт, код которого представлен в листинге 10 приложения А. Результат тестирования представлен в листинге 13 приложения А.

4.4 Разработка аппаратного ускорителя с использованием интерфейса расширения системы команд

Для дальнейшего ускорения расчёта умножения матриц 8 на 8 был использован блок интерфейса расширения системы команд (далее wrapper).

Разработанный в пункте 4.3 блок был интегрирован во wrapper микроконтроллера Sigma MCU. Ускорение было достигнуто за счёт того, что за 1 инструкцию передается по 1 значению из каждой входной матрицы, в отличие от системной шины, используемой в пункте 4.3, которая за 1 так способна загрузить данные только по 1 адресу.

Результат интеграции представлен в листинге 11 приложения А. Для запуска блока был написан baremetal драйвер, код которого представлен в листинге 12 приложения А.

Верификация блока производилась скриптами, использующимися в разделе 4.2 (стр. 4).

Измерение времени работы драйвера для расчёта умножения матриц приведено на рисунке 5.

Рисунок 5 - Измерение работы драйвера для wrapper’а с интегрированным блоком умножения матриц для процессора с одностадийным конвейером

Из сравнения результатов, полученных на рисунках 3 и 5, следует, что ускоритель справляется с вычислением одного экземпляра матрицы 8 на 8 быстрее на 142,7 мкс, то есть прирост производительности составил 464%. При этом стоит учесть, что основное время выполнения пришлось на выгрузку и загрузку данных. Если учесть, что исходя из плана в таблице 2, задержка расчета одной матрицы составляет 18 тактов, то в процентном соотношении расчет занял:

$$Comp = \frac{Clock_{Comp}}{T_{Total}} * T_{Clock} = \frac{18}{39{,}200784*10^{-6}} * \frac{1}{70*10^{6}} = 0,66\%. \quad (3)$$

Для тестирования блока на целевой плате была сгенерирована прошивка и написан python скрипт, код которого представлен в листинге 6 приложения А. Результат тестирования представлен в листинге 13 приложения А.

4.5 Сравнение характеристик

Сравнение времени выполнения всех реализаций умножения матриц 8 на 8 для процессора с одностадийном конвейере (тактовая частота 70 МГц) сведены в таблицу 4.

Таблица 4 - Сравнение временных характеристик

Реализация	Время выполнения, мкс	Тактов
Программа	181,903638	12733
Аппаратный ускоритель + системная шина	44,886612	3142
Аппаратный ускоритель + интерфейс расширения системы команд	39,200784	2744

Характеристики разработанной аппаратуры:

- Тактовая частота: 70 МГЦ;
- Total Negative Slack: 0 нс;
- Worst Negative Slack: 1,793 нс;
- Интервал инициации: 8 тактов;
- Операций в секунду: $\frac{70*10^6}{8} = 3888888;$
- Задержка: 18 тактов - 257 нс;

Потребляемые ресурсы ПЛИС:

- LUT: 11314 (17,85 %);
- FF: 27089 (21, 36 %);
- BRAM: 8 (5,93 %);
- DSP: 196 (81,67 %).

Выводы

В ходе работы были исследованы способы модификации процессора с архитектурой RISC-V. Реализовано вычисление умножения матриц программным и аппаратно ускоренным способом. Удалось достигнуть повышения производительности вычисления в 4,5 раза. Основной проблемой в вычислениях оказалась сложность записи входных и выходных данных из-за их размеров, во много раз превышающих размер системной шины. Тесты на всех стадиях проектирования были пройдены успешно.

Приложение А. Исходные тексты программ

Листинг 1 - hw_test_apps.py

```
# -*- coding:utf-8 -*-
from __future__ import division


import sys


sys.path.append('../../udm/sw')
import udm
from udm import *


import sigma
from sigma import *




udm = udm('COM1', 921600)
print("")


sigma = sigma(udm)
sigma.run_app_tests()


udm.disconnect()
```

Листинг 2 - Результат выполнения hw_test_apps.py

```
Connecting COM port...
COM port connected
Connection established, response:  0x55

sigma_tile@0x00000000 : IDCODE:  0xdeadbeef

#### DHRYSTONE TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  apps/dhrystone.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
```

```
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000d40 | 0x00003548 |

0x00000007 | 0x00001000
---------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000d40
---------------
Test program written!
Microseconds:  9
Dhrystones_Per_Second:  108090
DMIPS:  61.51963574274331
#### DHRYSTONE TEST PASSED! ####

#### MUL_SW TEST STARTED ####
Loading test program...
---------------
Loading elf file:  apps/mul_sw.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
---------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000005f0 | 0x000005f0 |
0x00000007 | 0x00001000
---------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000005f0
---------------
Test program written!
CORRECT:  6  *  7  =  42
CORRECT:  2  *  10  =  20
CORRECT:  256  *  256  =  65536
#### MUL_SW TEST PASSED! ####

#### MEDIAN TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  apps/median.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
---------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000c14 | 0x00000c14 |
0x00000007 | 0x00001000
---------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000c14
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### MEDIAN TEST PASSED! ####
```

```
#### QSORT TEST STARTED ####
Clearing buffer

Loading test program...
---------------
Loading elf file:  apps/qsort.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00002720 | 0x00002720 |
0x00000007 | 0x00001000
------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00002720
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### QSORT TEST PASSED! ####

#### RSORT TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  apps/rsort.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000028dc | 0x000038dc |
0x00000007 | 0x00001000
------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000028dc
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### RSORT TEST PASSED! ####

#### CRC32_SW TEST STARTED ####
ishod 55AA55AA
Loading test program...
---------------
Loading elf file:  apps/crc32.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
```

```
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000a28 | 0x00000a28 |
0x00000007 | 0x00001000
-----------------------------------------------------------------------------------------------------------


LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000a28
---------------
Test program written!
1         DBEF4    13887419    D3E7BB    1A1BEFE
2         2183C4    14125900    D78B4C    1AF2A85
3         269B62    16411608    FA6BD8    1D8E9C7
4         30356F    19227407    125630F    1DB5CD7
5         35BE92    20643769    13AFFB9    1E1C8A5
6         39E5EF    22136821    151C7F5    1E22134
7         5102D6    22762734    15B54EE    2232B34
8         949033    25885333    18AFA95    236BDAD
9         C56A53    26687179    19736CB    27C8266
res cbf43926
#### CRC32_SW TEST PASSED! ####

#### MD5 TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  apps/md5.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-----------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000c64 | 0x00000c64 |
0x00000007 | 0x00001000
-----------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000c64
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### MD5 TEST PASSED! ####

#### ELF BOOTLOADER TEST STARTED ####
Clearing buffer
Loading bootloader...
---------------
Loading elf file:  apps/bootloader.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-----------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000005e8 | 0x000005e8 |
```

```
0x00000007 | 0x00001000
-----------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000005e8
---------------
Loading test ELF image...


Test written!
#### ELF BOOTLOADER TEST PASSED! ####

#### IRQ COUNTER TEST STARTED ####
Loading test program...
---------------
Loading elf file:  apps/irq_counter.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-----------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000590 | 0x00000594 |
0x00000007 | 0x00001000
-----------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000590
---------------
Test program written!
#### PRESS IRQ BUTTON TO INCREMENT LED COUNTER! ####
LEDs:  5
#### IRQ TEST PASSED! ####

Total tests PASSED:  9 , FAILED:  0

Connection dropped
```

Листинг 3 - hw_test_compliance.py

```
# -*- coding:utf-8 -*-
from __future__ import division



import sys



sys.path.append('../../udm/sw')
import udm
from udm import *



import sigma
```

```
from sigma import *




udm = udm('COM1', 921600)
print("")


sigma = sigma(udm)
sigma.run_compliance_tests(["RV32I", "RV32M"])


udm.disconnect()
```

Листинг 4 - Результат выполнения hw_test_compliance.py

```
Connecting COM port...
COM port connected
Connection established, response:  0x55

sigma_tile@0x00000000 : IDCODE:  0xdeadbeef

################################################################################
#########
############################## RISC-V Compliance Test
################################
#### Imperas Software Ltd., 2019 <https://github.com/riscv/riscv-compliance> ####
################################################################################
#########

#### I-ADD TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-ADD-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000494 | 0x00000494 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000494
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
```

```
Reading data buffer...
Data buffer read!
#### I-ADD TEST PASSED! ####

#### I-ADDI TEST STARTED ####
Clearing buffer
Loading test program...

----------
Loading elf file:  riscv-compliance/I-ADDI-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000003e4 | 0x000003e4 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000003e4
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
----------
Test program written!
Reading data buffer...
Data buffer read!
#### I-ADDI TEST PASSED! ####

#### I-AND TEST STARTED ####
Clearing buffer
Loading test program...
----------
Loading elf file:  riscv-compliance/I-AND-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000494 | 0x00000494 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000494
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
----------
Test program written!
Reading data buffer...
Data buffer read!
#### I-AND TEST PASSED! ####
```

```
#### I-ANDI TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-ANDI-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000003e4 | 0x000003e4 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000003e4
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-ANDI TEST PASSED! ####

#### I-AUIPC TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-AUIPC-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000004c4 | 0x000004c4 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000004c4
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-AUIPC TEST PASSED! ####

#### I-BEQ TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-BEQ-01.riscv
```

```
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
---------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000006c4 | 0x000006c4 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000

---------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000006c4
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-BEQ TEST PASSED! ####

#### I-BGE TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-BGE-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
---------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000006c4 | 0x000006c4 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
---------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000006c4
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-BGE TEST PASSED! ####

#### I-BGEU TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-BGEU-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
---------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
```

```
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000006c4 | 0x000006c4 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
----------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000006c4
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
```

```
Data buffer read!
#### I-BGEU TEST PASSED! ####

#### I-BLT TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-BLT-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
----------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000006c4 | 0x000006c4 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
----------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000006c4
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-BLT TEST PASSED! ####

#### I-BLTU TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-BLTU-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
----------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000006c4 | 0x000006c4 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
```

```
---------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000006c4
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-BLTU TEST PASSED! ####

#### I-BNE TEST STARTED ####
Clearing buffer
Loading test program...
```

```
---------------
Loading elf file:  riscv-compliance/I-BNE-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
---------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000006c4 | 0x000006c4 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
---------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000006c4
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-BNE TEST PASSED! ####

#### I-JAL TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-JAL-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
---------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000574 | 0x00000574 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
---------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000574
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
```

```
Reading data buffer...
Data buffer read!
#### I-JAL TEST PASSED! ####

#### I-JALR TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-JALR-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------

 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000674 | 0x00000674 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000674
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-JALR TEST PASSED! ####

#### I-LB TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-LB-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000400 | 0x00000400 |
0x00000007 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000400
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-LB TEST PASSED! ####

#### I-LBU TEST STARTED ####
```

```
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-LBU-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000400 | 0x00000400 |
0x00000007 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000400
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-LBU TEST PASSED! ####

#### I-LH TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-LH-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000400 | 0x00000400 |
0x00000007 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000400
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-LH TEST PASSED! ####

#### I-LHU TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-LHU-01.riscv
-- e_type: ET_EXEC
```

```
-- e_machine: RISC-V
Program Headers:
------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000400 | 0x00000400 |
0x00000007 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000400
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer…

Data buffer read!
#### I-LHU TEST PASSED! ####

#### I-LUI TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-LUI-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000344 | 0x00000344 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000344
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-LUI TEST PASSED! ####

#### I-LW TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-LW-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
```

```
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000400 | 0x00000400 |
0x00000007 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-----------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000400
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-LW TEST PASSED! ####

#### I-OR TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-OR-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-----------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000494 | 0x00000494 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-----------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000494
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-OR TEST PASSED! ####

#### I-ORI TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-ORI-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-----------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000003e4 | 0x000003e4 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-----------------------------------------------------------------------------------------------------------
```

```
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000003e4
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-ORI TEST PASSED! ####

#### I-SB TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-SB-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------------
№ | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000464 | 0x00000464 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-------------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000464
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-SB TEST PASSED! ####

#### I-SH TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-SH-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000464 | 0x00000464 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-------------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000464
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
```

```
Data buffer read!
#### I-SH TEST PASSED! ####

#### I-SLL TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-SLL-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000464 | 0x00000464 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
 -------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000464
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-SLL TEST PASSED! ####

#### I-SLLI TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-SLLI-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000003e4 | 0x000003e4 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000003e4
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-SLLI TEST PASSED! ####

#### I-SLT TEST STARTED ####
Clearing buffer
```

```
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-SLT-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
------------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000494 | 0x00000494 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
------------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000494
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-SLT TEST PASSED! ####

#### I-SLTI TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-SLTI-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
------------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000003e4 | 0x000003e4 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
------------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000003e4
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-SLTI TEST PASSED! ####

#### I-SLTIU TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-SLTIU-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
```

```
Program Headers:
-----------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000003e4 | 0x000003e4 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-----------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000003e4
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
----------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-SLTIU TEST PASSED! ####

#### I-SLTU TEST STARTED ####
Clearing buffer
Loading test program...
----------------
Loading elf file:  riscv-compliance/I-SLTU-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-----------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000494 | 0x00000494 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-----------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000494
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
----------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-SLTU TEST PASSED! ####

#### I-SRA TEST STARTED ####
Clearing buffer
Loading test program...
----------------
Loading elf file:  riscv-compliance/I-SRA-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-----------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000464 | 0x00000464 |
```

```
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-----------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000464
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-SRA TEST PASSED! ####

#### I-SRAI TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-SRAI-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-----------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000003e4 | 0x000003e4 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-----------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000003e4
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-SRAI TEST PASSED! ####

#### I-SRL TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-SRL-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-----------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000464 | 0x00000464 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-----------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000464
```

```
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-SRL TEST PASSED! ####

#### I-SRLI TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-SRLI-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000003e4 | 0x000003e4 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000

-------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000003e4
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-SRLI TEST PASSED! ####

#### I-SUB TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-SUB-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000494 | 0x00000494 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000494
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
```

```
#### I-SUB TEST PASSED! ####

#### I-SW TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-SW-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000564 | 0x00000564 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000564
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...

Data buffer read!
#### I-SW TEST PASSED! ####

#### I-XOR TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-XOR-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000494 | 0x00000494 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000494
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-XOR TEST PASSED! ####

#### I-XORI TEST STARTED ####
Clearing buffer
Loading test program...
```

```
---------------
Loading elf file:  riscv-compliance/I-XORI-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000003e4 | 0x000003e4 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000003e4
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-XORI TEST PASSED! ####

#### I-DELAY_SLOTS TEST STARTED ####
Clearing buffer
Loading test program...
```

```
---------------
Loading elf file:  riscv-compliance/I-DELAY_SLOTS-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000324 | 0x00000324 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000020 | 0x00000020 |
0x00000004 | 0x00001000
-------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000324
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000020
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-DELAY_SLOTS TEST PASSED! ####

#### I-ENDIANESS TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-ENDIANESS-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
```

```
-------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000258 | 0x00000258 |
0x00000007 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000020 | 0x00000020 |
0x00000004 | 0x00001000
-------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000258
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000020
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-ENDIANESS TEST PASSED! ####

#### I-IO TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-IO-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000004a4 | 0x000004a4 |
0x00000007 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x000000b0 | 0x000000b0 |
0x00000004 | 0x00001000
-------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000004a4
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x000000b0
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-IO TEST PASSED! ####

#### I-NOP TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-NOP-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000344 | 0x00000344 |
0x00000005 | 0x00001000
```

```
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000080 | 0x00000080 |
0x00000004 | 0x00001000
-----------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000344
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000080
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-NOP TEST PASSED! ####

#### I-RF_size TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-RF_size-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-----------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x000003b4 | 0x000003b4 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000090 | 0x00000090 |
0x00000004 | 0x00001000
-----------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x000003b4
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000090
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-RF_size TEST PASSED! ####

#### I-RF_width TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-RF_width-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-----------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000494 | 0x00000494 |
0x00000005 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000080 | 0x00000080 |
0x00000004 | 0x00001000
-----------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000494
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000080
```

```
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-RF_width TEST PASSED! ####

#### I-RF_x0 TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/I-RF_x0-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000304 | 0x00000304 |
0x00000007 | 0x00001000
 1 | 0x00000001 | 0x00001000 | 0x00006000 | 0x00006000 | 0x00000030 | 0x00000030 |
0x00000004 | 0x00001000
------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00000304
LOADING: file offset: 0x00001000 , hw addr: 0x00006000 size: 0x00000030
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### I-RF_x0 TEST PASSED! ####

#### mul TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/mul-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00003508 | 0x00003508 |
0x00000007 | 0x00001000
 1 | 0x00000001 | 0x00004000 | 0x00006000 | 0x00006000 | 0x00000b04 | 0x00000b04 |
0x00000004 | 0x00001000
------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00003508
LOADING: file offset: 0x00004000 , hw addr: 0x00006000 size: 0x00000b04
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### mul TEST PASSED! ####
```

```
#### mulh TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/mulh-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00003508 | 0x00003508 |
0x00000007 | 0x00001000
 1 | 0x00000001 | 0x00004000 | 0x00006000 | 0x00006000 | 0x00000b04 | 0x00000b04 |
0x00000004 | 0x00001000
-------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00003508
LOADING: file offset: 0x00004000 , hw addr: 0x00006000 size: 0x00000b04
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### mulh TEST PASSED! ####

#### mulhsu TEST STARTED ####
Clearing buffer
Loading test program...

---------------
Loading elf file:  riscv-compliance/mulhsu-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00003908 | 0x00003908 |
0x00000007 | 0x00001000
 1 | 0x00000001 | 0x00004000 | 0x00006000 | 0x00006000 | 0x00000c04 | 0x00000c04 |
0x00000004 | 0x00001000
-------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00003908
LOADING: file offset: 0x00004000 , hw addr: 0x00006000 size: 0x00000c04
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### mulhsu TEST PASSED! ####

#### mulhu TEST STARTED ####
Clearing buffer
Loading test program...
---------------
```

```
Loading elf file:  riscv-compliance/mulhu-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00003f08 | 0x00003f08 |
0x00000007 | 0x00001000
 1 | 0x00000001 | 0x00004000 | 0x00006000 | 0x00006000 | 0x00000d04 | 0x00000d04 |
0x00000004 | 0x00001000
------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00003f08
LOADING: file offset: 0x00004000 , hw addr: 0x00006000 size: 0x00000d04
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### mulhu TEST PASSED! ####

#### div TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/div-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
------------------------------------------------------------------------------------------------------
```

```
№ | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00003508 | 0x00003508 |
0x00000007 | 0x00001000
 1 | 0x00000001 | 0x00004000 | 0x00006000 | 0x00006000 | 0x00000b04 | 0x00000b04 |
0x00000004 | 0x00001000
------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00003508
LOADING: file offset: 0x00004000 , hw addr: 0x00006000 size: 0x00000b04
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### div TEST PASSED! ####

#### divu TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/divu-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
------------------------------------------------------------------------------------------------------
```

```
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00003f08 | 0x00003f08 |
0x00000007 | 0x00001000
 1 | 0x00000001 | 0x00004000 | 0x00006000 | 0x00006000 | 0x00000d04 | 0x00000d04 |
0x00000004 | 0x00001000
-----------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00003f08
LOADING: file offset: 0x00004000 , hw addr: 0x00006000 size: 0x00000d04
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### divu TEST PASSED! ####

#### rem TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/rem-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-----------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00003508 | 0x00003508 |
0x00000007 | 0x00001000
 1 | 0x00000001 | 0x00004000 | 0x00006000 | 0x00006000 | 0x00000b04 | 0x00000b04 |
0x00000004 | 0x00001000
 -----------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00003508
LOADING: file offset: 0x00004000 , hw addr: 0x00006000 size: 0x00000b04
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### rem TEST PASSED! ####

#### remu TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:  riscv-compliance/remu-01.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-----------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00003f08 | 0x00003f08 |
0x00000007 | 0x00001000
```

```
 1 | 0x00000001 | 0x00004000 | 0x00006000 | 0x00006000 | 0x00000d04 | 0x00000d04 |
0x00000004 | 0x00001000
-----------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x00003f08
LOADING: file offset: 0x00004000 , hw addr: 0x00006000 size: 0x00000d04
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### remu TEST PASSED! ####

Total tests PASSED:  52 , FAILED:  0

##########################################################################
#########

Connection dropped
```

листинг 5 - mat_mul_main.c

```
#include "io.h"
#include <stdio.h>

#define N 8

int main()
{
    IO_LED = 0xffff;
    unsigned int A[N][N] = {


            {10, 12, 58, 95, 75, 45, 65, 84},
            {58, 25, 14, 65, 5,  21, 95, 14},
            {75, 21, 23, 95, 87, 47, 65, 68},
            {45, 21, 89, 17, 65, 43, 75, 73},
            {14, 85, 65, 84, 45, 36, 84, 16},
            {15, 35, 54, 35, 65, 14, 87, 65},
            {45, 45, 87, 35, 34, 25, 65, 48},
            {98, 25, 65, 48, 65, 45, 21, 88},
    };
    unsigned int B[N][N] = {
            {45, 65, 84, 25, 87,  47, 56, 87},
            {98, 35, 11, 87, 66,  68, 45, 58},
            {12, 87, 65, 12, 56,  57, 65, 45},
            {42, 53, 12, 85, 45,  24, 65, 21},
            {87, 12, 65, 87, 255, 87, 25, 65},
            {45, 68, 48, 68, 48,  68, 78, 32},
            {68, 98, 24, 35, 45,  68, 78, 45},
            {68, 87, 35, 36, 87,  57, 69, 1}
    };
```

```
    int C[N][N] = {{0, 0, 0, 0, 0, 0, 0, 0},
                   {0, 0, 0, 0, 0, 0, 0, 0},
                   {0, 0, 0, 0, 0, 0, 0, 0},
                   {0, 0, 0, 0, 0, 0, 0, 0},
                   {0, 0, 0, 0, 0, 0, 0, 0},
                                       {0, 0, 0, 0, 0, 0, 0, 0},
                                       {0, 0, 0, 0, 0, 0, 0, 0},
                                       {0, 0, 0, 0, 0, 0, 0, 0}};

    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            for (int k = 0; k < N; k++) {
                C[i][j] += A[i][k] * B[k][j];
            }
        }
    }

    int k = 0;

    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {


            io_buf_int[k + j] = C[i][j];
        }

        k += N;
    }
    IO_LED = 0xdddd;
    return 0;
}
```

Листинг 6 - hw_test_mat_mul_program.py

```
# -*- coding:utf-8 -*-
from __future__ import division

import sys

sys.path.append('../../udm/sw')
import udm
from udm import *

import sigma
from sigma import *


udm = udm('COM7', 921600)
print("")
```

```
sigma = sigma(udm)

# firmware_filename =
'C:/Users/admin/Desktop/activecore-master/designs/rtl/sigma/sw/apps/matrix_mul.riscv'
firmware_filename =
'C:/Users/admin/Desktop/activecore-master/designs/rtl/sigma/sw/apps/mat_mul_coproc_d
river.riscv'



verify_data = [24994, 28789, 17417, 24949, 40703, 25665, 27296, 15495, 16750,
            21324, 10940, 15010, 18181, 15905, 19647, 13777, 28427, 29172,
            21017, 27541, 46756, 27605, 28418, 20925, 23519, 29709, 20644,
            19297, 39415, 26864, 26221, 18544, 25603, 26604, 14573, 25544,
            32623, 25146, 25883, 18710, 22844, 24666, 14835, 19035, 35031,
            23231, 22188, 15153, 20716, 26578, 17000, 17720, 30303, 22788,
            23657, 17158, 24748, 28998, 23277, 22103, 43312, 26322, 26803,
            20607]

sigma.hw_test_generic(sigma, "MAT_MUL", firmware_filename, 0.1, verify_data)

print(sigma.tile.udm.rdarr32(0x6000, len(verify_data)),'\n')

udm.disconnect()
```

Листинг 7 - matrix_mul.sv

```
`define MAT_DIM_LEN 8
`define ITER_DATA_LEN 63

module matrix_mul(input logic [31:0] A [0:`ITER_DATA_LEN], B [0:`ITER_DATA_LEN],
                  input logic reset, clk, clear, start,
                  output logic [31:0] C [0:`ITER_DATA_LEN],
                  output logic done);

            localparam [7:0] line_shift_array[0:7] = {0, 8, 16, 24, 32, 40, 48, 56};

            integer i, j, k, c;

            logic [2:0]cntr_mul, cntr_MAC,
             cntr_add[0:`MAT_DIM_LEN-1], cntr_MAC_delay1, cntr_MAC_delay2,
             cntr_MAC_delay3, cntr_MAC_delay4, cntr_MAC_delay5,
             cntr_MAC_delay6, cntr_MAC_delay7;

            logic [31:0] Result [0:`MAT_DIM_LEN-1],
MAC[0:`MAT_DIM_LEN-1][0:`MAT_DIM_LEN-1],
            mul_reg[0:`MAT_DIM_LEN-1][0:`ITER_DATA_LEN],
            A_REG[0:`ITER_DATA_LEN], B_REG[0:`ITER_DATA_LEN], cntr_mul_reg;

            logic [32:0] cntr;
            always @(posedge clk, posedge reset)
```

```
            begin
               if (reset | clear)
               begin

                  for(i = 0; i < `ITER_DATA_LEN + 1; i = i + 1)
                  begin: forloop


                     C[i] <= 0;
                     A_REG[i] <= 0;
                     B_REG[i] <= 0;
                     for(j = 0; j < `MAT_DIM_LEN; j++)
                     mul_reg[j][i] <= 0;
                  end
                  for(k = 0; k < `MAT_DIM_LEN; k++) begin: resetMAC
                     Result[k] <= 0;
                     for(j = 0; j < `MAT_DIM_LEN; j++) begin: resmacCol
                        MAC[k][j] <= 0;
                     end
                  end
               cntr <= 0;
               done <= 1'b0;
               cntr_mul <= 0;
               cntr_mul_reg <= 0;
               cntr_MAC <= 0;
               cntr_MAC_delay1 <= 0;
               cntr_MAC_delay2 <= 0;
               cntr_MAC_delay3 <= 0;
               cntr_MAC_delay4 <= 0;
               cntr_MAC_delay5 <= 0;
               cntr_MAC_delay6 <= 0;
               cntr_MAC_delay7 <= 0;
               cntr_add <= {0, 0, 0, 0, 0, 0, 0, 0};

               end
               else begin

               if (cntr_MAC_delay7 == 3'd2) begin C[0:7] <= Result; done <= 1'b0; end
               else if (cntr_MAC_delay7 == 3'd3) begin C[8:15] <= Result; done <= 1'b0; end
               else if (cntr_MAC_delay7 == 3'd4) begin C[16:23] <= Result; done <= 1'b0;
end
               else if (cntr_MAC_delay7 == 3'd5)  begin C[24:31] <= Result; done <= 1'b0;
end
               else if (cntr_MAC_delay7 == 3'd6) begin C[32:39] <= Result; done <= 1'b0;
end
               else if (cntr_MAC_delay7 == 3'd7) begin C[40:47] <= Result; done <= 1'b0;
end
               else if (cntr_MAC_delay7 == 3'd0) begin C[48:55] <= Result; done <= 1'b0;
end
               else if (cntr_MAC_delay7 == 3'd1) begin C[56:63] <= Result; done <= 1'b0;
end

               if(cntr_MAC_delay5 == 3'd0 && cntr > 4'he)
```

```
                done <= 1'b1;

            if(start) begin
                cntr <= cntr + 1'b1;
                cntr_mul <= cntr;
                cntr_mul_reg <= cntr[2:0]*`MAT_DIM_LEN;
                cntr_MAC <= cntr_mul;
                cntr_add <= {cntr_MAC, cntr_MAC_delay1, cntr_MAC_delay2,


                cntr_MAC_delay3, cntr_MAC_delay4, cntr_MAC_delay5,
                cntr_MAC_delay6, cntr_MAC_delay7};
                cntr_MAC_delay1 <= cntr_MAC;
                cntr_MAC_delay2 <= cntr_MAC_delay1;
                cntr_MAC_delay3 <= cntr_MAC_delay2;
                cntr_MAC_delay4 <= cntr_MAC_delay3;
                cntr_MAC_delay5 <= cntr_MAC_delay4;
                cntr_MAC_delay6 <= cntr_MAC_delay5;
                cntr_MAC_delay7 <= cntr_MAC_delay6;
                A_REG <= A;
                B_REG <= B;

            MAC[0] <= {mul_reg[cntr_MAC][0], mul_reg[cntr_MAC][8],
            mul_reg[cntr_MAC][16], mul_reg[cntr_MAC][24],
            mul_reg[cntr_MAC][32], mul_reg[cntr_MAC][40],
            mul_reg[cntr_MAC][48], mul_reg[cntr_MAC][56]};

        for (i = 0; i < `MAT_DIM_LEN; i++) begin: looprow
            mul_reg[cntr_mul][line_shift_array[i]] <= A_REG[cntr_mul_reg]*B_REG[i];
            for(j = 1; j < `MAT_DIM_LEN; j++) begin: loopcolumn
                mul_reg[cntr_mul][line_shift_array[i] + j] <= A_REG[cntr_mul_reg +
j]*B_REG[i + line_shift_array[j]];
            end
        end

        for (i = 1; i < `MAT_DIM_LEN; i++) begin: loop_add_prev
            for(j = 0; j < `MAT_DIM_LEN; j++) begin: loopcolumn_add_prev
                MAC[i][j] <= MAC[i-1][j] + mul_reg[cntr_add[i-1]][i + line_shift_array[j]];
            end
        end

        Result <= MAC[`MAT_DIM_LEN-1];

        end
        end
        end

endmodule
```

Листинг 8 - Sigma.sv

```
`include "sigma_tile.svh"

module sigma
#(
        parameter CPU = "none"
        , UDM_BUS_TIMEOUT = (1024*1024*100)
        , UDM_RTX_EXTERNAL_OVERRIDE = "NO"
        , DEBOUNCER_FACTOR_POW = 20
        , delay_test_flag = 0
        , mem_init_type="elf"


        , mem_init_data = "data.elf"
        , mem_size = 1024
)
(
        input clk_i
        , input arst_i
        , input irq_btn_i
        , input rx_i
        , output tx_o
        , input [31:0] gpio_bi
        , output [31:0] gpio_bo
);

wire srst;
reset_sync reset_sync
(
        .clk_i(clk_i),
        .arst_i(arst_i),
        .srst_o(srst)
);

wire udm_reset;
wire cpu_reset;
assign cpu_reset = srst | udm_reset;

wire irq_btn_debounced;
debouncer
#(
   .FACTOR_POW(DEBOUNCER_FACTOR_POW)
) debouncer (
        .clk_i(clk_i)
        , .rst_i(srst)
        , .in(irq_btn_i)
        , .out(irq_btn_debounced)
);

MemSplit32 hif();
MemSplit32 xif();

localparam IRQ_NUM_POW = 4;
```

```
logic start_mat_mul, clear_mat_mul, mul_done;
logic [31:0] A[0:63], B[0:63], C [0:63], Result_mul[0:63];
integer i;

sigma_tile #(
        .corenum(0)
        , .mem_init_type(mem_init_type)
        , .mem_init_data(mem_init_data)
        , .mem_size(mem_size)
        , .CPU(CPU)
        , .PATH_THROUGH("YES")
        , .IRQ_NUM_POW(IRQ_NUM_POW)


) sigma_tile (
        .clk_i(clk_i)
        , .rst_i(cpu_reset)

        , .irq_debounced_bi({1'b0, irq_btn_debounced, 3'h0})

        , .hif(hif)
        , .xif(xif)
);

udm #(
   .BUS_TIMEOUT(UDM_BUS_TIMEOUT)
   , .RTX_EXTERNAL_OVERRIDE(UDM_RTX_EXTERNAL_OVERRIDE)
) udm (
        .clk_i(clk_i)
        , .rst_i(srst)

        , .rx_i(rx_i)
        , .tx_o(tx_o)

        , .rst_o(udm_reset)

        , .bus_req_o(hif.req)
        , .bus_we_o(hif.we)
        , .bus_addr_bo(hif.addr)
        , .bus_be_bo(hif.be)
        , .bus_wdata_bo(hif.wdata)
        , .bus_ack_i(hif.ack)
        , .bus_resp_i(hif.resp)
        , .bus_rdata_bi(hif.rdata)
);

matrix_mul Mat_mul88(.A(A),
                .B(B),
                .C(Result_mul),
                .clk(clk_i),
                .reset(arst_i),
                .clear(clear_mat_mul),
                .start(start_mat_mul),
```

```
                .done(mul_done));

localparam CSR_LED_ADDR         = 32'h80000000;
localparam CSR_SW_ADDR          = 32'h80000004;
localparam start_mat_mul_ADDR   = 32'hc0000004;
localparam clear_mat_mul_ADDR   = 32'hc0000000;
localparam done_ADDR            = 32'h90000000;

logic [31:0] gpio_bo_reg;
assign gpio_bo = gpio_bo_reg;
logic [31:0] gpio_bi_reg;
always @(posedge clk_i) gpio_bi_reg <= gpio_bi;

assign xif.ack = xif.req;   // xif always ready to accept request
```

```
logic csr_resp;
logic [31:0] csr_rdata;


// bus request
always @(posedge clk_i)
    begin

    csr_resp <= 1'b0;

    if (arst_i) begin
        for (i = 0; i < 64; i++) begin: resetCloop
            C[i] <= 0;
            A[i] <= 0;
            B[i] <= 0;
        end
        start_mat_mul <= 0;
        clear_mat_mul <= 0;
        end
    else begin
        if(mul_done)
        for(i = 0; i < 64; i++) begin: assignCloop

            C[i] <= Result_mul[i];
        end
   end

    if (xif.req && xif.ack)
        begin

        if (xif.we)     // writing
            begin
            if (xif.addr == CSR_LED_ADDR) gpio_bo_reg <= xif.wdata;

            // matrix multiplication data writing
            if (xif.addr[31:28] == 4'hc && xif.addr[8:0] > 9'h4 && xif.addr[8:0] < 9'h105 &&
~arst_i) A[xif.addr[8:2] - 2'h2] <= xif.wdata;
            if (xif.addr[31:28] == 4'hc && xif.addr[9:0] > 10'h104 && xif.addr[9:0] < 10'h205 &&
```

```
~arst_i) B[xif.addr[9:2] - 10'h42] <= xif.wdata;
          if(xif.addr == start_mat_mul_ADDR && ~arst_i) start_mat_mul <= xif.wdata;
          if(xif.addr == clear_mat_mul_ADDR && ~arst_i) clear_mat_mul <= xif.wdata;


          end

       else           // reading
          begin
          if (xif.addr == CSR_LED_ADDR)
             begin
             csr_resp <= 1'b1;
             csr_rdata <= gpio_bo_reg;
             end
          if (xif.addr == CSR_SW_ADDR)


             begin
             csr_resp <= 1'b1;
             csr_rdata <= gpio_bi_reg;
             end
          end

         // matrix multiplication data reading
          if (xif.addr > 32'hc0000204 && xif.addr < 32'hc0000305)
             begin
             csr_resp <= 1'b1;
             csr_rdata <= C[xif.addr[11:2] - 32'h82];
          end
          if (xif.addr == done_ADDR)
             begin
             csr_resp <= 1'b1;
             csr_rdata <= mul_done;
             end
       end
    end

// bus response
always @*
    begin
    xif.resp = csr_resp;
    xif.rdata = 0;
    if (csr_resp) xif.rdata = csr_rdata;
    end

endmodule
```

Листинг 9 - mat_mul_driver.c

```
#include "io.h"

#define CLEAR (*(volatile unsigned int*) 0xc0000000)
#define START (*(volatile unsigned int*) 0xc0000004)

#define A_ADDR (*(volatile unsigned int*) 0xc0000008)
#define B_ADDR (*(volatile unsigned int*) 0xc0000108)

#define C_ADDR 0xc0000208

#define IO_MATRIX_INT_LENGTH 0x40

#define io_matrix_int        (*(volatile int (*)[IO_MATRIX_INT_LENGTH])(C_ADDR))

#define done_addr (*(volatile unsigned int*) 0x90000000)

#define N 8

#define N2 N*N


int main(int argc, char *argv[]) {

        IO_LED = 0x55aa;

   CLEAR = 1;
   START = 0;


   int i;
   int j;
   volatile unsigned int *data_pntr;
   unsigned int data_shift;

   int A[N][N] = {
        {10, 12, 58, 95, 75, 45, 65, 84},
        {58, 25, 14, 65, 5,  21, 95, 14},
        {75, 21, 23, 95, 87, 47, 65, 68},
        {45, 21, 89, 17, 65, 43, 75, 73},
        {14, 85, 65, 84, 45, 36, 84, 16},
        {15, 35, 54, 35, 65, 14, 87, 65},
        {45, 45, 87, 35, 34, 25, 65, 48},
        {98, 25, 65, 48, 65, 45, 21, 88},
   };
   int B[N][N] = {
        {45, 65, 84, 25, 87,  47, 56, 87},
        {98, 35, 11, 87, 66,  68, 45, 58},
        {12, 87, 65, 12, 56,  57, 65, 45},
        {42, 53, 12, 85, 45,  24, 65, 21},
        {87, 12, 65, 87, 255, 87, 25, 65},
        {45, 68, 48, 68, 48,  68, 78, 32},
        {68, 98, 24, 35, 45,  68, 78, 45},
```

```
        {68, 87, 35, 36, 87,  57, 69, 1}
    };


    for (i = 0; i < N; i += 1) {
        for (j = 0; j < N; j += 1) {
            data_shift = N * i + j;
            data_pntr = &A_ADDR + data_shift;
            *data_pntr = A[i][j];
            data_pntr = &B_ADDR + data_shift;
            *data_pntr = B[i][j];
        }
    }

    CLEAR = 0;
    START = 1;

    while (1) {
        if(done_addr) break;}

        for(i = 0; i < N2; i++) io_buf_int[i] = io_matrix_int[i];


        IO_LED = 0xdddd;

        return 0;
}
```

Листинг 10 - matrix_mul_test.py

```
from __future__ import division

import sys

sys.path.append('../../udm/sw')
import udm
from udm import *

import sigma
from sigma import *



def matrix_test_generic(sigma, test_name, firmware_filename, sleep_secs, verify_data):
    print("#### " + test_name + " TEST STARTED ####");



    print("Clearing buffer")
    udm.clr(0xc0000208, 64)

    print("Loading test program...")
    sigma.tile.loadelf(firmware_filename)
```

```
    print("Test program written!")

    time.sleep(sleep_secs)

    print("Reading data buffer...")
    rdarr = sigma.tile.udm.rdarr32(0xc0000208, len(verify_data))
    print("Data buffer read!")

    test_succ_flag = 1
    for i in range(len(verify_data)):
        if (verify_data[i] != rdarr[i]):
            test_succ_flag = 0
            print("Test failed on data ", i, "! Expected: ", hex(verify_data[i]), ", received: ",
hex(rdarr[i]))

    if (test_succ_flag):
        print("#### " + test_name + " TEST PASSED! ####");
    else:
        print("#### " + test_name + " TEST FAILED! ####")

    print("")
    return test_succ_flag



udm = udm('COM11', 921600)
print("")

verify_data = [24994, 28789, 17417, 24949, 40703, 25665, 27296, 15495, 16750,
               21324, 10940, 15010, 18181, 15905, 19647, 13777, 28427, 29172,
               21017, 27541, 46756, 27605, 28418, 20925, 23519, 29709, 20644,
               19297, 39415, 26864, 26221, 18544, 25603, 26604, 14573, 25544,
               32623, 25146, 25883, 18710, 22844, 24666, 14835, 19035, 35031,
               23231, 22188, 15153, 20716, 26578, 17000, 17720, 30303, 22788,
               23657, 17158, 24748, 28998, 23277, 22103, 43312, 26322, 26803,
               20607]

firmware_filename =
"C:/Users/admin/Desktop/activecore-master/designs/rtl/sigma/sw/apps/mat_mul_driver.ris
cv"
sigma = sigma(udm)

matrix_test_generic(sigma,"MAT_MUL", firmware_filename, 0.01, verify_data)

print(sigma.tile.udm.rdarr32(0xc0000208, len(verify_data)))

udm.disconnect()
```

Листинг 11 - coproc_custom0_wrapper.sv

```
`include "coproc_if.svh"

`define ITER_DATA_LEN 63
`define MAT_DIM_LEN 8

module coproc_custom0_wrapper (
        input logic unsigned [0:0] clk_i
        , input logic unsigned [0:0] rst_i
        , output logic unsigned [0:0] stream_resp_bus_genfifo_req_o
        , output resp_struct stream_resp_bus_genfifo_wdata_bo
        , input logic unsigned [0:0] stream_resp_bus_genfifo_ack_i
        , input logic unsigned [0:0] stream_req_bus_genfifo_req_i
        , input req_struct stream_req_bus_genfifo_rdata_bi
        , output logic unsigned [0:0] stream_req_bus_genfifo_ack_o
);

logic [31:0] A [0:`ITER_DATA_LEN], B [0:`ITER_DATA_LEN], C [0:`ITER_DATA_LEN];

logic start, done;

assign stream_req_bus_genfifo_ack_o = stream_req_bus_genfifo_req_i;

localparam [5:0] line_shift_array[0:7] = {0, 8, 16, 24, 32, 40, 48, 56};

            integer i, j, k, c;

            logic [6:0] cur_index;

            logic [2:0]cntr_mul, cntr_MAC,
             cntr_add[0:`MAT_DIM_LEN-1], cntr_MAC_delay1, cntr_MAC_delay2,
             cntr_MAC_delay3, cntr_MAC_delay4, cntr_MAC_delay5,
             cntr_MAC_delay6, cntr_MAC_delay7;

            logic [31:0] Result [0:`MAT_DIM_LEN-1],
MAC[0:`MAT_DIM_LEN-1][0:`MAT_DIM_LEN-1],
            mul_reg[0:`MAT_DIM_LEN-1][0:`ITER_DATA_LEN], cntr_mul_reg;

            logic [32:0] cntr;
            always @(posedge clk_i, posedge rst_i)
            begin
                if (rst_i)
                begin

                    for(i = 0; i < `ITER_DATA_LEN + 1; i = i + 1)
                    begin: forloop
                        C[i] <= 0;
                        A[i] <= 0;
                        B[i] <= 0;
                        for(j = 0; j < `MAT_DIM_LEN; j++)
                        mul_reg[j][i] <= 0;
                    end
```

```
    for(k = 0; k < `MAT_DIM_LEN; k++) begin: resetMAC
        Result[k] <= 0;
        for(j = 0; j < `MAT_DIM_LEN; j++) begin: resmacCol
            MAC[k][j] <= 0;
        end
    end
cur_index <= 7'b0;
cntr <= 0;
start <= 0;
done <= 1'b0;
cntr_mul <= 0;
cntr_mul_reg <= 0;
cntr_MAC <= 0;
cntr_MAC_delay1 <= 0;
cntr_MAC_delay2 <= 0;
cntr_MAC_delay3 <= 0;
cntr_MAC_delay4 <= 0;
cntr_MAC_delay5 <= 0;
cntr_MAC_delay6 <= 0;
cntr_MAC_delay7 <= 0;
cntr_add <= {0, 0, 0, 0, 0, 0, 0, 0};

end
else if(cur_index < 7'h40) begin

    stream_resp_bus_genfifo_req_o <= 1'b0;
    if (stream_req_bus_genfifo_req_i)
        begin
        A[cur_index] <= stream_req_bus_genfifo_rdata_bi.src0_data;
        B[cur_index] <= stream_req_bus_genfifo_rdata_bi.src1_data;
        stream_resp_bus_genfifo_wdata_bo <= C[cur_index];
        cur_index <= cur_index + 1;
        stream_resp_bus_genfifo_req_o <= 1'b1;
        end

end
else begin

if (cntr_MAC_delay7 == 3'd2) begin C[0:7] <= Result; end
else if (cntr_MAC_delay7 == 3'd3) begin C[8:15] <= Result; end
else if (cntr_MAC_delay7 == 3'd4) begin C[16:23] <= Result; end
else if (cntr_MAC_delay7 == 3'd5)  begin C[24:31] <= Result; end
else if (cntr_MAC_delay7 == 3'd6) begin C[32:39] <= Result; end
else if (cntr_MAC_delay7 == 3'd7) begin C[40:47] <= Result; end
else if (cntr_MAC_delay7 == 3'd0) begin C[48:55] <= Result; end
else if (cntr_MAC_delay7 == 3'd1) begin C[56:63] <= Result; end

if(cntr_MAC_delay5 == 3'd0 && cntr > 4'hf)
    begin
    cur_index <= 0;
    stream_resp_bus_genfifo_req_o <= 1'b1;
    end
```

```
                if(cur_index[6]) begin
                    cntr <= cntr + 1'b1;
                    cntr_mul <= cntr;
                    cntr_mul_reg <= cntr[2:0]*`MAT_DIM_LEN;
                    cntr_MAC <= cntr_mul;
                    cntr_add <= {cntr_MAC, cntr_MAC_delay1, cntr_MAC_delay2,
                    cntr_MAC_delay3, cntr_MAC_delay4, cntr_MAC_delay5,
                    cntr_MAC_delay6, cntr_MAC_delay7};
                    cntr_MAC_delay1 <= cntr_MAC;
                    cntr_MAC_delay2 <= cntr_MAC_delay1;
                    cntr_MAC_delay3 <= cntr_MAC_delay2;
                    cntr_MAC_delay4 <= cntr_MAC_delay3;
                    cntr_MAC_delay5 <= cntr_MAC_delay4;
                    cntr_MAC_delay6 <= cntr_MAC_delay5;
                    cntr_MAC_delay7 <= cntr_MAC_delay6;

                MAC[0] <= {mul_reg[cntr_MAC][0], mul_reg[cntr_MAC][8],
                mul_reg[cntr_MAC][16], mul_reg[cntr_MAC][24],
                mul_reg[cntr_MAC][32], mul_reg[cntr_MAC][40],
                mul_reg[cntr_MAC][48], mul_reg[cntr_MAC][56]};

            for (i = 0; i < `MAT_DIM_LEN; i++) begin: looprow
                mul_reg[cntr_mul][line_shift_array[i]] <= A[cntr_mul_reg]*B[i];
                for(j = 1; j < `MAT_DIM_LEN; j++) begin: loopcolumn
                        mul_reg[cntr_mul][line_shift_array[i] + j] <= A[cntr_mul_reg + j]*B[i +
line_shift_array[j]];
                end
            end

            for (i = 1; i < `MAT_DIM_LEN; i++) begin: loop_add_prev
                for(j = 0; j < `MAT_DIM_LEN; j++) begin: loopcolumn_add_prev
                        MAC[i][j] <= MAC[i-1][j] + mul_reg[cntr_add[i-1]][i + line_shift_array[j]];
                end
            end

            Result <= MAC[`MAT_DIM_LEN-1];

            end
            end
            end

endmodule
```

Листинг 12 - mat_mul_coproc_driver.c

```
#include "io.h"
#include <malloc.h>

#define N 8
```

```
#define N2 N*N


// wrapper for instruction activating custom coprocessor
inline unsigned int custom0_instr_wrapper (unsigned int a, unsigned int b) {
        unsigned int result;
        asm volatile (".insn r 0x0b, 0x0, 0x0, %0, %1, %2"
                : "=r" (result)
                : "r" (a), "r" (b));
                return result;
        }

int main(int argc, char *argv[]) {

        IO_LED = 0x55aa;
   int i;
   int j;

   unsigned int A[N][N] = {
        {10, 12, 58, 95, 75, 45, 65, 84},
        {58, 25, 14, 65, 5,  21, 95, 14},
        {75, 21, 23, 95, 87, 47, 65, 68},
        {45, 21, 89, 17, 65, 43, 75, 73},
        {14, 85, 65, 84, 45, 36, 84, 16},
        {15, 35, 54, 35, 65, 14, 87, 65},
        {45, 45, 87, 35, 34, 25, 65, 48},
        {98, 25, 65, 48, 65, 45, 21, 88},
   };
   unsigned int B[N][N] = {
        {45, 65, 84, 25, 87,  47, 56, 87},
        {98, 35, 11, 87, 66,  68, 45, 58},
        {12, 87, 65, 12, 56,  57, 65, 45},
        {42, 53, 12, 85, 45,  24, 65, 21},
        {87, 12, 65, 87, 255, 87, 25, 65},
        {45, 68, 48, 68, 48,  68, 78, 32},
        {68, 98, 24, 35, 45,  68, 78, 45},
        {68, 87, 35, 36, 87,  57, 69, 1}
   };




        for(i = 0; i < N; i++) {
                for(j = 0; j < N; j++){

                custom0_instr_wrapper(A[i][j], B[i][j]);

                }
        }

        IO_LED = 0;
        for(i = 0; i < N2; i++) {
```

```
                io_buf_int[i] = custom0_instr_wrapper(0, 0);
        }

        IO_LED = 0xdddd;

        while (1) {}
        return 0;

}
```

Листинг 13 - Результат тестирования matrix_mul_test.py, hw_test_mat_mul_program.py

```
Connecting COM port...
COM port connected
Connection established, response:  0x55

sigma_tile@0x00000000 : IDCODE:  0xdeadbeef

#### MAT_MUL TEST STARTED ####
Clearing buffer
Loading test program...
---------------
Loading elf file:
C:/Users/admin/Desktop/activecore-master/designs/rtl/sigma/sw/apps/mat_mul_coproc_dr
iver.riscv
-- e_type: ET_EXEC
-- e_machine: RISC-V
Program Headers:
-----------------------------------------------------------------------------------------------------------
 № | p_type     | p_offset   | p_vaddr    | p_paddr    | p_filesz   | p_memsz    | p_flags    |
p_align
 0 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x0000083c | 0x0000083c |
0x00000007 | 0x00001000
-----------------------------------------------------------------------------------------------------------
LOADING: file offset: 0x00000000 , hw addr: 0x00000000 size: 0x0000083c
---------------
Test program written!
Reading data buffer...
Data buffer read!
#### MAT_MUL TEST PASSED! ####

[24994, 28789, 17417, 24949, 40703, 25665, 27296, 15495, 16750, 21324, 10940, 15010,
18181, 15905, 19647, 13777, 28427, 29172, 21017, 27541, 46756, 27605, 28418, 20925,
23519, 29709, 20644, 19297, 39415, 26864, 26221, 18544, 25603, 26604, 14573, 25544,
32623, 25146, 25883, 18710, 22844, 24666, 14835, 19035, 35031, 23231, 22188, 15153,
20716, 26578, 17000, 17720, 30303, 22788, 23657, 17158, 24748, 28998, 23277, 22103,
43312, 26322, 26803, 20607]

Connection dropped
```

Листинг 14 - riscv_tb.sv

```
`timescale 1ps / 1ps

//`define CLK_HALF_PERIOD                 5000        // external 100 MHZ
`define CLK_HALF_PERIOD           7143              // external 70 MHZ
//`define CLK_HALF_PERIOD                 6250        // external 80 MHZ
//`define CLK_HALF_PERIOD                 3571        // external 140 MHZ
//`define CLK_HALF_PERIOD                 3333        // external 150 MHZ
//`define CLK_HALF_PERIOD                 3125        // external 160 MHZ

`define DIVIDER_115200          32'd8680000
`define DIVIDER_19200           32'd52083000
`define DIVIDER_9600            32'd104166000
`define DIVIDER_4800            32'd208333000
`define DIVIDER_2400            32'd416666000

//localparam file_name =
"C:/Users/admin/Desktop/activecore-master/designs/rtl/sigma/sw/apps/mat_mul_coproc_d
river.riscv";
localparam file_name =
"C:/Users/admin/Desktop/activecore-master/designs/rtl/sigma/sw/apps/matrix_mul.riscv";
//localparam file_name =
"C:/Users/admin/Desktop/activecore-master/designs/rtl/sigma/sw/apps/mat_mul_driver.ris
cv";

module riscv_tb ();
//
reg CLK, RST, rx;
reg [31:0] SW;
wire [31:0] LED;
reg irq_btn;

logic [63:0] testvectors [10000:0];


logic [31:0] vectornum, errors, addr_C, exp_C, rec_C;

sigma
#(
        .CPU("riscv_1stage")
        //.CPU("riscv_2stage")
        //.CPU("riscv_3stage")
        //.CPU("riscv_4stage")
        //.CPU("riscv_5stage")
        //.CPU("riscv_6stage")

        , .UDM_RTX_EXTERNAL_OVERRIDE("YES")
        , .DEBOUNCER_FACTOR_POW(2)
        , .delay_test_flag(0)

        , .mem_init_type("elf")
        , .mem_init_data(file_name)
        , .mem_size(8192)
```

```
) sigma
(
        .clk_i(CLK)
        , .arst_i(RST)
        , .irq_btn_i(irq_btn)
        , .rx_i(rx)
        //, .tx_o()
        , .gpio_bi({8'h0, SW, 8'h0})
        , .gpio_bo(LED)
);

//////////////////////////
/////////tasks////////////
//////////////////////////

reg parity;
integer i, j, k;

reg [32:0] rate;
reg [1:0] configuration;


////wait////
task WAIT
        (
         input reg [15:0] periods
         );
begin
for (i=0; i<periods; i=i+1)
        begin
        #(`CLK_HALF_PERIOD*2);
        end
end
endtask
```

```
////reset all////
task RESET_ALL ();
begin
        CLK = 1'b0;
        RST = 1'b1;
        irq_btn = 1'b0;
        rx = 1'b1;
        #(`CLK_HALF_PERIOD/2);
        RST = 1;
        #(`CLK_HALF_PERIOD*6);
        RST = 0;
        while (sigma.srst) WAIT(10);
end
endtask

`define UDM_RX_SIGNAL rx
```

```
`define UDM_BLOCK sigma.udm
`include "udm.svh"
udm_driver udm;

//////////////////
// initial block //
localparam CPU_RAM_ADDR        = 32'h00000000;
localparam CSR_LED_ADDR        = 32'h80000000;
localparam CSR_SW_ADDR         = 32'h80000004;

initial
begin
        $display ("### SIMULATION STARTED ###");

   if(file_name ==
"C:/Users/admin/Desktop/activecore-master/designs/rtl/sigma/sw/apps/mat_mul_driver.ris
cv")
      $readmemh("vector.mem", testvectors);
   else
      $readmemh("coproc_vector.mem", testvectors);

         RESET_ALL();
         errors = 0;
         vectornum = 0;

         WAIT(1000);

         irq_btn = 1'b0;
         WAIT(100);
         irq_btn = 1'b0;
         WAIT(50);

         udm.check();
         //udm.hreset();
         WAIT(1000000);
```

```
      do begin
      {addr_C, exp_C} = testvectors[vectornum];
      udm.rd32(addr_C, rec_C);
      if(rec_C != exp_C) begin

      $display("ERROR: addr: %h; C = %h (Expected %h)", addr_C, rec_C, exp_C);
      errors++;

      end
      vectornum++;
      end while(testvectors[vectornum] !== 64'bx);

         $display("%d tests completed with %d errors", vectornum[7:0], errors[7:0]);
         $display ("### TEST PROCEDURE FINISHED ###");
         $stop;
end
```

```
//
always #`CLK_HALF_PERIOD CLK = ~CLK;


endmodule
```

Листинг 15 - coproc_vector.mem

```
00006000_000061a2
00006004_00007075
00006008_00004409
0000600c_00006175
00006010_00009eff
00006014_00006441
00006018_00006aa0
0000601c_00003c87
00006020_0000416e
00006024_0000534c
00006028_00002abc
0000602c_00003aa2
00006030_00004705
00006034_00003e21
00006038_00004cbf
0000603c_000035d1
00006040_00006f0b
00006044_000071f4
00006048_00005219
0000604c_00006b95
00006050_0000b6a4
00006054_00006bd5
00006058_00006f02
0000605c_000051bd
00006060_00005bdf
00006064_0000740d
00006068_000050a4
0000606c_00004b61
00006070_000099f7
00006074_000068f0
00006078_0000666d
0000607c_00004870
00006080_00006403
00006084_000067ec
00006088_000038ed
0000608c_000063c8
00006090_00007f6f
00006094_0000623a
00006098_0000651b
0000609c_00004916
000060a0_0000593c
000060a4_0000605a
```

```
000060a8_000039f3
000060ac_00004a5b
000060b0_000088d7
000060b4_00005abf
000060b8_000056ac
000060bc_00003b31
000060c0_000050ec
000060c4_000067d2
000060c8_00004268
000060cc_00004538
000060d0_0000765f
000060d4_00005904
000060d8_00005c69
000060dc_00004306
000060e0_000060ac
000060e4_00007146
000060e8_00005aed
000060ec_00005657
000060f0_0000a930
000060f4_000066d2
000060f8_000068b3
000060fc_0000507f
```

Листинг 16 - vector.mem

```
c0000208_000061a2
c000020c_00007075
c0000210_00004409
c0000214_00006175
c0000218_00009eff
c000021c_00006441
c0000220_00006aa0
c0000224_00003c87
c0000228_0000416e
c000022c_0000534c
c0000230_00002abc
c0000234_00003aa2
c0000238_00004705
c000023c_00003e21
c0000240_00004cbf
c0000244_000035d1
c0000248_00006f0b
c000024c_000071f4
c0000250_00005219
c0000254_00006b95
c0000258_0000b6a4
c000025c_00006bd5
c0000260_00006f02
c0000264_000051bd
c0000268_00005bdf
c000026c_0000740d
c0000270_000050a4
```

c0000274_00004b61
c0000278_000099f7
c000027c_000068f0
c0000280_0000666d
c0000284_00004870
c0000288_00006403
c000028c_000067ec
c0000290_000038ed
c0000294_000063c8
c0000298_00007f6f
c000029c_0000623a
c00002a0_0000651b
c00002a4_00004916
c00002a8_0000593c
c00002ac_0000605a
c00002b0_000039f3
c00002b4_00004a5b
c00002b8_000088d7
c00002bc_00005abf
c00002c0_000056ac
c00002c4_00003b31
c00002c8_000050ec
c00002cc_000067d2
c00002d0_00004268
c00002d4_00004538
c00002d8_0000765f
c00002dc_00005904
c00002e0_00005c69
c00002e4_00004306
c00002e8_000060ac
c00002ec_00007146
c00002f0_00005aed
c00002f4_00005657
c00002f8_0000a930
c00002fc_000066d2
c0000300_000068b3
c0000304_0000507f

Листинг 17 - Вывод скрипта riscv_tb.sv

```
### SIMULATION STARTED ###
UDM RD32: addr: 0x00006000, data: 0x000061a2
UDM RD32: addr: 0x00006004, data: 0x00007075
UDM RD32: addr: 0x00006008, data: 0x00004409
UDM RD32: addr: 0x0000600c, data: 0x00006175
UDM RD32: addr: 0x00006010, data: 0x00009eff
UDM RD32: addr: 0x00006014, data: 0x00006441
UDM RD32: addr: 0x00006018, data: 0x00006aa0
UDM RD32: addr: 0x0000601c, data: 0x00003c87
UDM RD32: addr: 0x00006020, data: 0x0000416e
UDM RD32: addr: 0x00006024, data: 0x0000534c
UDM RD32: addr: 0x00006028, data: 0x00002abc
UDM RD32: addr: 0x0000602c, data: 0x00003aa2
UDM RD32: addr: 0x00006030, data: 0x00004705
UDM RD32: addr: 0x00006034, data: 0x00003e21
UDM RD32: addr: 0x00006038, data: 0x00004cbf
UDM RD32: addr: 0x0000603c, data: 0x000035d1
UDM RD32: addr: 0x00006040, data: 0x00006f0b
UDM RD32: addr: 0x00006044, data: 0x000071f4
UDM RD32: addr: 0x00006048, data: 0x00005219
UDM RD32: addr: 0x0000604c, data: 0x00006b95
UDM RD32: addr: 0x00006050, data: 0x0000b6a4
UDM RD32: addr: 0x00006054, data: 0x00006bd5
UDM RD32: addr: 0x00006058, data: 0x00006f02
UDM RD32: addr: 0x0000605c, data: 0x000051bd
UDM RD32: addr: 0x00006060, data: 0x00005bdf
UDM RD32: addr: 0x00006064, data: 0x0000740d
UDM RD32: addr: 0x00006068, data: 0x000050a4
UDM RD32: addr: 0x0000606c, data: 0x00004b61
UDM RD32: addr: 0x00006070, data: 0x000099f7
UDM RD32: addr: 0x00006074, data: 0x000068f0
UDM RD32: addr: 0x00006078, data: 0x0000666d
UDM RD32: addr: 0x0000607c, data: 0x00004870
UDM RD32: addr: 0x00006080, data: 0x00006403
UDM RD32: addr: 0x00006084, data: 0x000067ec
UDM RD32: addr: 0x00006088, data: 0x000038ed
UDM RD32: addr: 0x0000608c, data: 0x000063c8
UDM RD32: addr: 0x00006090, data: 0x00007f6f
UDM RD32: addr: 0x00006094, data: 0x0000623a
UDM RD32: addr: 0x00006098, data: 0x0000651b
UDM RD32: addr: 0x0000609c, data: 0x00004916
UDM RD32: addr: 0x000060a0, data: 0x0000593c
UDM RD32: addr: 0x000060a4, data: 0x0000605a
UDM RD32: addr: 0x000060a8, data: 0x000039f3
UDM RD32: addr: 0x000060ac, data: 0x00004a5b
UDM RD32: addr: 0x000060b0, data: 0x000088d7
UDM RD32: addr: 0x000060b4, data: 0x00005abf
UDM RD32: addr: 0x000060b8, data: 0x000056ac
UDM RD32: addr: 0x000060bc, data: 0x00003b31
UDM RD32: addr: 0x000060c0, data: 0x000050ec
UDM RD32: addr: 0x000060c4, data: 0x000067d2
```

```
UDM RD32: addr: 0x000060c8, data: 0x00004268
UDM RD32: addr: 0x000060cc, data: 0x00004538
UDM RD32: addr: 0x000060d0, data: 0x0000765f
UDM RD32: addr: 0x000060d4, data: 0x00005904
UDM RD32: addr: 0x000060d8, data: 0x00005c69
UDM RD32: addr: 0x000060dc, data: 0x00004306
UDM RD32: addr: 0x000060e0, data: 0x000060ac
UDM RD32: addr: 0x000060e4, data: 0x00007146
UDM RD32: addr: 0x000060e8, data: 0x00005aed
UDM RD32: addr: 0x000060ec, data: 0x00005657
UDM RD32: addr: 0x000060f0, data: 0x0000a930
UDM RD32: addr: 0x000060f4, data: 0x000066d2
UDM RD32: addr: 0x000060f8, data: 0x000068b3
UDM RD32: addr: 0x000060fc, data: 0x0000507f
 64 tests completed with   0 errors
### TEST PROCEDURE FINISHED ###
```