Санкт-Петербургский Политехнический Университет Петра Великого

Институт компьютерных наук и технологии

Высшая школа интеллектуальных систем и суперкомпьютерных технологий

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ Lab4\_z1

Дисциплина: Проектирование реконфигурируемых гибридных вычислительных систем

Выполнил студент Курякин Д.

Гр. 3540901/12001

Руководитель, доцент Антонов А.П.

«\_\_» \_\_\_\_\_\_ 2022

Санкт-Петербург

2022

Оглавление

[1. Задание 3](#_Toc116916474)

[2. Исходный код функции 3](#_Toc116916475)

[3. Исходный код теста 3](#_Toc116916476)

[3.1 Моделирование 4](#_Toc116916477)

[4. Первое решение — Solution1 5](#_Toc116916478)

[4.1 Исходные настройки 5](#_Toc116916479)

[4.2 Синтез 5](#_Toc116916480)

[5. Первое решение — Solution2 7](#_Toc116916481)

[5.1 Исходные настройки 7](#_Toc116916482)

[5.2 Синтез 8](#_Toc116916483)

[6. Сравнение Solution1 и Solution2 9](#_Toc116916484)

[7. Выводы 10](#_Toc116916485)

1. Задание

* Создать на языке Си функцию,
  + Умножающую два вектора (векторное умножение – в результате вектор). Размер векторов - **N** элементов типа data\_in. Результат – вектор - **N** элементов типа data\_out.
    - Аргументы функции: два входных вектора, выходной вектор
  + В файле lab4\_1.h должны быть определены: **N = 4**, тип данных data\_in, имеющий тип short; тип данных data\_out, имеющий тип int
  + Дать самому внутреннему циклу, реализованному в функции, имя L1, остальным L2…
* Создать на языке Си тест для проверки работы функции. Тест должен обеспечивать
  + Запуск функции 3 раза
    - Заполнение векторов случайными значениями.
    - Перед следующим запуском функции не забудьте обнулить ожидаемое значение.
  + проверку правильности полученного функцией результата.
* Отладить функцию и тест (при неправильном результате в любом из запусков функции тест должен сообщать об ошибке).
* Создать скрипт автоматизирующий процесс:
  + Создания проекта lab4\_1,
  + Назначения функции lab4\_1
  + Подключения файла lab4\_1.c (папка source ),
  + Подключения файла lab4\_1\_test.c (папка source),
  + Создания решения
    - sol1, для которого
      * Микросхема: xa7a12tcsg325-1Q
      * Период тактового сигнала: 10нс, uncertainty 1нс.
      * Си моделирование
      * ОТКЛЮЧИТЬ конвейеризацию для всех циклов
      * Синтез
      * С/RTL cosimulation (с опцией Dump Trace = Port)
    - sol2, для которого
      * для цикла L1 задать директиву **Pipeline II=1**
      * Синтез
      * С/RTL cosimulation (с опцией Dump Trace = Port)
    - Sol3, для которого
      * для цикла L1 задать директиву **Pipeline II=1**
      * **для внешнего цикла задайте Pipeline off и rewind**
      * Синтез
      * С/RTL cosimulation (с опцией Dump Trace = Port)
    - Sol4, для которого
      * для цикла L1 задать директиву **Pipeline off**
      * **для внешнего цикла задайте Pipeline II=1 и rewind**
      * Синтез
      * С/RTL cosimulation (с опцией Dump Trace = Port)
* Отладить и проверить работу созданного скрипта.
* После выполнения скрипта открыть GUI
* Убедиться, что созданы все решения
* Используя средства HLS проверить, сравнить и зафиксировать использованные интерфейсы для каждого из трех решений
* Используя средства HLS сравнить полученные решения (привести Report Comparison)
  + Timing
  + Latency
  + Utilization Estimation
* Используя средства HLS сравнить и привести результаты планирования (Schedule viewer) для каждого из трех решений

Для каждого из трех решений привести временные диаграммы

* + Решение 1
    - Надо показать
      * II, Latency для первого запуска функции
  + Решение 2
    - Надо показать
      * II, Latency для первого запуска функции
      * Два запуска функции
  + Решение 3
    - Надо показать
      * Три (два) запуска функции
  + Решение 4
    - Надо показать
      * Три (два) запуска функции

**2. Исходный код функции**

Исходный код заголовочного файла приведен ниже.

|  |
| --- |
| **typedef** **short** **data\_in**;  **typedef** **int** **data\_out**;  **#define** **N** 4  **void** **lab4\_z1**(**data\_in** firstVector\_arr[N], **data\_in** secondVector\_arr[N], **data\_out** resultVecror\_arr[N]); |

Исходный код синтезируемой функции приведен ниже.

|  |
| --- |
| **#include** "lab4\_z1.h"  **#include** <stdio.h>  **void** **lab4\_z1**(**data\_in** firstVector\_arr[N], **data\_in** secondVector\_arr[N], **data\_out** resultVecror\_arr[N])  {  **int** i, j;  L2: **for** (i = 0; i < N; i++)  {  L1: **for** (j = 0; j < N; j++)  {  resultVecror\_arr[i] += firstVector\_arr[j] \* secondVector\_arr[i];  }  }  } |

**3. Исходный код теста**

Исходный код теста для проверки функции lab4\_z1 приведен ниже.

|  |
| --- |
| **#include** <stdio.h>  **#include** "lab3\_z2.h"  **int** **arentEqual**(**data\_sc** expected\_arr[M], **data\_sc** actual\_arr[M])  {  **for** (**int** i = 0; i < M; i++)  **if** (expected\_arr[i] != actual\_arr[i])  {  **fprintf**(stdout, " ERROR: expected=%d actual=%d \n", expected\_arr[i], actual\_arr[i]);  **return** 1;  }  **return** 0;  }  **int** **main**() {  **data\_sc** in[N];  **data\_sc** out[M], expected\_ar[M];  **int** i, j;  **int** pass = 0;  **for** (i = 0; i < 32; i++)  {  **for** (j = 0; j < N; j++)  in[j] = **rand**() % M;  **for** (j = 0; j < M; j++){  out[j] = 0;  expected\_ar[j] = 0;  }  **lab3\_z2**(in, out);  **for** (j = 0; j < N; j++)  expected\_ar[in[j]]++;  pass = **arentEqual**(out, expected\_ar);  }  **if** (!pass)  **fprintf**(stdout, "----------Pass!------------ \n");  **else**  **fprintf**(stderr, "----------Fail!------------ \n");  **return** pass;  }; |

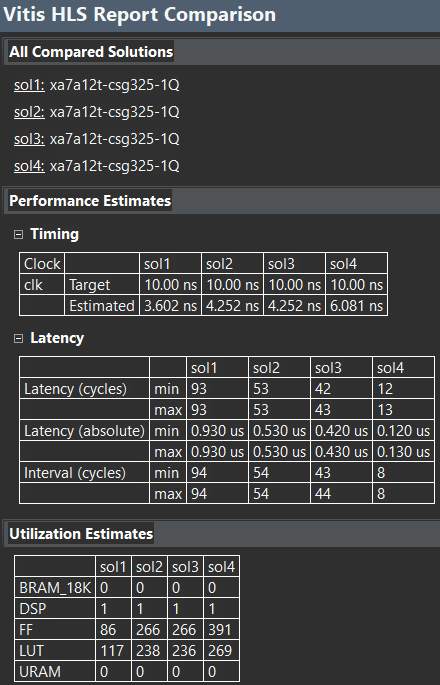
Тест обеспечивает проверку корректной работы функции.

В листинге приведен tcl скрипт.

|  |
| --- |
| open\_project -reset lab4\_z1  add\_files ./source/lab4\_z1.c  set\_top lab4\_z1  add\_files -tb ./source/lab4\_z1\_test.c  open\_solution -reset "sol1"  set\_part {xa7a12tcsg325-1Q}  create\_clock -period 10 -name clk  set\_clock\_uncertainty 1  set\_directive\_pipeline -off "lab4\_z1/L2"  set\_directive\_pipeline -off "lab4\_z1/L1"  csim\_design -clean  csynth\_design  cosim\_design -trace\_level port -tool xsim  open\_solution -reset "sol2"  set\_part {xa7a12tcsg325-1Q}  create\_clock -period 10 -name clk  set\_clock\_uncertainty 1  set\_directive\_pipeline -II 1 "lab4\_z1/L1"  set\_directive\_pipeline -off "lab4\_z1/L2"  csim\_design -clean  csynth\_design  cosim\_design -trace\_level port -tool xsim  open\_solution -reset "sol3"  set\_part {xa7a12tcsg325-1Q}  create\_clock -period 10 -name clk  set\_clock\_uncertainty 1  set\_directive\_pipeline -II 1 "lab4\_z1/L1"  set\_directive\_pipeline -off -rewind "lab4\_z1/L2"  csim\_design -clean  csynth\_design  cosim\_design -trace\_level port -tool xsim  open\_solution -reset "sol4"  set\_part {xa7a12tcsg325-1Q}  create\_clock -period 10 -name clk  set\_clock\_uncertainty 1  set\_directive\_pipeline -off "lab4\_z1/L1"  set\_directive\_pipeline -II 1 -rewind "lab4\_z1/L2"  csim\_design -clean  csynth\_design  cosim\_design -trace\_level port -tool xsim |

**6. Сравнение решений**

Был запущен tcl скрипт. После выполнения скрипта была выполнена команда Compare Reports. После которой получаем результаты сравнения решений.



*Рис. 1 Сравнение решений*

Ниже представлена таблица и диаграмма сравнения решений.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | sol1 | sol2 | sol3 | sol4 |
| Clock | Target ( ns) | 10 | 10 | 10 | 10 |
| Estimated ( ns) | 3,602 | 4,252 | 4,252 | 6,081 |
| Iteration Interval | (cycles) | 93 | 53 | 43 | 13 |
| (ns) | 428 | 278 | 226 | 92 |
| Resources | BRAM\_18K | 0 | 0 | 0 | 0 |
| DSP48E | 1 | 1 | 1 | 1 |
| FF | 86 | 266 | 266 | 391 |
| LUT | 117 | 238 | 236 | 269 |
| URAM | 0 | 0 | 0 | 0 |

*Рис. 6 Диаграмма сравнения решений*

**7. Выводы**

В данной работе были получен опыт работы с vitis hls.