САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

ПЕТРА ВЕЛИКОГО

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Институт компьютерных наук и технологий

Высшая школа интеллектуальных систем и суперкомпьютерных технологий

Лабораторная работа lab1\_z2

Дисциплина:

«Проектирование реконфигурируемых гибридных вычислительных систем»

Тема: Введение в Vivado HLS

Выполнил:

Бараев Д. Р.

Группа: 3540901/02001

Преподаватель: А. П. Антонов

Санкт-Петербург

2021

**Оглавление**

[1. Задание 5](#_Toc83253123)

[2. Исходный код функции 6](#_Toc83253124)

[3. Исходный код теста 7](#_Toc83253125)

[3.1. Моделирование 8](#_Toc83253126)

[4. Первое решение - Solution\_1 9](#_Toc83253127)

[4.1. Исходные настройки 9](#_Toc83253128)

[4.2. Синтез 9](#_Toc83253129)

[4.2.1. Оценка быстродействия 9](#_Toc83253130)

[4.2.2. Оценка аппаратных ресурсов 10](#_Toc83253131)

[4.2.3. Планировщик Schedule Viewer 11](#_Toc83253132)

[4.2.4. Распределение ресурсов Resource Viewer 12](#_Toc83253133)

[4.3. Си и RTL моделирование 12](#_Toc83253134)

[4.3.1. Анализ результатов 12](#_Toc83253135)

[4.3.2. Отчет по быстродействию 13](#_Toc83253136)

[4.3.3. Временная диаграмма 13](#_Toc83253137)

[5. Второе решение - Solution\_2 14](#_Toc83253138)

[5.1. Исходные настройки 14](#_Toc83253139)

[5.2. Синтез 14](#_Toc83253140)

[5.2.1. Оценка быстродействия 14](#_Toc83253141)

[5.2.2. Оценка аппаратных ресурсов 15](#_Toc83253142)

[5.2.3. Планировщик Schedule Viewer 16](#_Toc83253143)

[5.2.4. Распределение ресурсов Resource Viewer 17](#_Toc83253144)

[5.3. Си и RTL моделирование 17](#_Toc83253145)

[5.3.1. Анализ результатов 17](#_Toc83253146)

[5.3.2. Отчет по быстродействию 18](#_Toc83253147)

[5.3.3. Временная диаграмма 18](#_Toc83253148)

[6. Выводы 19](#_Toc83253149)

[6.1. Сравнение параметров двух решений 19](#_Toc83253150)

[6.2. Какое из двух решений имеет меньший II 20](#_Toc83253151)

[6.3. Какое из двух решений обеспечивает большее быстродействие? 20](#_Toc83253152)

[6.5. Какое из двух решений является более оптимальным? 20](#_Toc83253153)

**Список иллюстраций**

[Рисунок 1 Исходный код функции 6](file:///D:\Политех\Otchyoty_HLS\Отчёты%20HLS\lab1_z2.docx#_Toc51503422)

[Рисунок 2 Исходный код теста 8](file:///D:\Политех\Otchyoty_HLS\Отчёты%20HLS\lab1_z2.docx#_Toc51503423)

[Рисунок 3 Результаты моделирования Си кода 9](file:///D:\Политех\Otchyoty_HLS\Отчёты%20HLS\lab1_z2.docx#_Toc51503424)

[Рисунок 4 Оценка быстродействия для solution\_1 10](file:///D:\Политех\Otchyoty_HLS\Отчёты%20HLS\lab1_z2.docx#_Toc51503425)

[Рисунок 5 Аппаратные ресурсы для solution\_1 11](file:///D:\Политех\Otchyoty_HLS\Отчёты%20HLS\lab1_z2.docx#_Toc51503426)

[Рисунок 6 Планировщик для solution\_1 11](file:///D:\Политех\Otchyoty_HLS\Отчёты%20HLS\lab1_z2.docx#_Toc51503427)

[Рисунок 7 Performance Profile для solution1 12](file:///D:\Политех\Otchyoty_HLS\Отчёты%20HLS\lab1_z2.docx#_Toc51503428)

[Рисунок 8 Распределение ресурсов по тактам для solution\_1 12](file:///D:\Политех\Otchyoty_HLS\Отчёты%20HLS\lab1_z2.docx#_Toc51503429)

[Рисунок 9 Результаты моделирования Си и RTL для solution\_1 13](file:///D:\Политех\Otchyoty_HLS\Отчёты%20HLS\lab1_z2.docx#_Toc51503430)

[Рисунок 10 Быстродействие при совместном моделировании solution\_1 13](file:///D:\Политех\Otchyoty_HLS\Отчёты%20HLS\lab1_z2.docx#_Toc51503431)

[Рисунок 11 Временная диаграмма solution\_1 14](file:///D:\Политех\Otchyoty_HLS\Отчёты%20HLS\lab1_z2.docx#_Toc51503432)

[Рисунок 12 Оценка быстродействия для solution\_2 15](file:///D:\Политех\Otchyoty_HLS\Отчёты%20HLS\lab1_z2.docx#_Toc51503433)

[Рисунок 13 Аппаратные ресурсы для solution\_2 16](file:///D:\Политех\Otchyoty_HLS\Отчёты%20HLS\lab1_z2.docx#_Toc51503434)

[Рисунок 14 Планировщик для solution\_2 16](file:///D:\Политех\Otchyoty_HLS\Отчёты%20HLS\lab1_z2.docx#_Toc51503435)

[Рисунок 15 Распределение ресурсов по тактам для solution\_2 17](file:///D:\Политех\Otchyoty_HLS\Отчёты%20HLS\lab1_z2.docx#_Toc51503436)

[Рисунок 16 Результаты моделирования Си и RTL для solution\_2 18](file:///D:\Политех\Otchyoty_HLS\Отчёты%20HLS\lab1_z2.docx#_Toc51503437)

[Рисунок 17 Быстродействие при совместном моделировании для solution\_2 18](file:///D:\Политех\Otchyoty_HLS\Отчёты%20HLS\lab1_z2.docx#_Toc51503438)

[Рисунок 18 Временная диаграмма для solution\_2 19](file:///D:\Политех\Otchyoty_HLS\Отчёты%20HLS\lab1_z2.docx#_Toc51503439)

[Рисунок 19 Сравнение двух решений 20](file:///D:\Политех\Otchyoty_HLS\Отчёты%20HLS\lab1_z2.docx#_Toc51503440)

1. Задание

* Создать проект lab1\_z2 (рабочая папка – папка lab1\_z2 с заданием)
* Подключить файл lab1\_z2.c (папка source)
* Подключить тест lab1\_z2\_test.c (папка source)
* Модифицировать тест – число, добавляемое на каждой итерации теста = порядковому номеру в списке группы.
* Осуществить моделирование
* Сделать solution1
* задать: clock period 6; clock\_uncertainty 0.1
* осуществить синтез
* привести и пояснить в отчете:
* Таблицу performance estimates => Timing/summary и Latency/Summary
* Таблицу utilization estimates => summary
* Performance Profile
* scheduler viewer (выполнить Zoom to Fit)
* На скриншоте показать Latency
* На скриншоте показать Initiation Interval
* resource viewer (выполнить Zoom to Fit и развернуть все строки)
* Осуществить C|RTL моделирование (Dump trace – ALL)
* Привести скриншот Cosimulation report и дать пояснения
* Открыть временную диаграмму
* Отобразить все сигналы раздела Design Top Signal
* Отобразить 2 цикла работы на одном экране
* На скриншоте показать Latency
* На скриншоте показать Initiation Interval
* Сделать solution2
* задать: clock period 12; clock\_uncertainty 0.1
* Выполнить всю программу для Solution1
* В разделе Выводы отчета привести
* Сравнение Timing, Latency, Utilization Estimation (используя Compare reports) и пояснить отличия
  + Ответить на вопросы:
    - Какое из двух решений имеет меньший II
      * В тактах?
      * В ns (число тактов надо умножить на estimated)?
    - какое из двух решений обеспечивает большее быстродействие (в ns)?

Насколько велико отличие?

* + - Какое из двух решений требует меньших аппаратных затрат?

Насколько велико отличие?

1. Исходный код функции

Исходный код синтезируемой функции приведен на рисунке Рис. 1

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 1 Исходный код функции

В строке 1 Рисунок 1 объявляется функция с ее входными параметрами типа char и массивами int. Далее на второй строке объявляется переменные x, y типа integer для хранения результата. В строках 3–7 в цикле происходят математические операции с входными данными. В строке 4 переменной х присваивается значение массива in а в строке пять к y присваивается результат математических операций. В строке 6 в массив out записывается y.

1. Исходный код теста

Исходный код теста для проверки функции lab1\_z2 (см. раздел 2), исправленный в соответствии с заданием, приведен на рисунке Рис. 2. Тест обеспечивает:

* Проверку результата функции **lab1\_z2 (**Рисунок 1**)**

Тест состоит из двух функций. Функция main и areEqual. Функция areEqual проверяется фактические значения с ожидаемыми путем сравнения массивов. В случае если значения массивов не равны она возвращает 1, что далее накапливается в переменную pass для последующего ее использования в целях определения прошёл ли тест или нет.

В функции main в строках 15–25 объявляется переменные для работы теста.

* Переменные inA, inB, inC типа char входные данные, над которыми функция lab1\_z1 проводит математические операции
* Массив inArr так же содержит входные данные
* Массив outArr служит для записи в него результатов функции
* Переменная pass типа integer служит счётчиком для подсчёта ошибок
* Переменные i, j являются переменными цикла
* Массив refOut – массив с ожидаемыми значения для последующего его сравнения с полученными значениями.

В строках 27–40 Рисунок 2 реализуется цикл, в котором происходит вызов функции lab1\_z2 и сравнение полученных результатов с ожидаемыми. Функция сравнения вызывается в строке 35. В конце каждой итерации цикла переменные inA, inB, inC увеличиваются на 1.

В строках 42–51 проверяется переменная pass, если она равна нулю, то возвращается 0 и строка об успешном прохождении теста, в противном случае возвращается 1 и информация об его провале.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 2 Исходный код теста

* 1. Моделирование

Результаты моделирования исходного кода синтезируемой функции приведены на Рисунок 3. Результаты моделирования показывают, что тест успешно пройден – консоль оповестила об успешном прохождение теста. В случае если бы фактический результат не был равен ожидаемому функция main вернула бы 1, и консоль оповестила бы о не прохождении теста.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3 Результаты моделирования Си кода

1. Первое решение - Solution\_1
   1. Исходные настройки

При создании первого решения задаются следующие настройки:

* clock period: 6,
* clock uncertain: 0.1,
* part: xa7a12tcsg325-1q.
  1. Синтез
     1. Оценка быстродействия

Результаты оценки быстродействия Timing/summary и Latency/Summary приведены на Рисунок 4.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 4 Оценка быстродействия для solution\_1

Target – планируемое время на один такт.

Estimated – оценочное время.

Latency (cycle) – количество тактов latency за один цикл.

Latency (absolute) – время затраченное на latency.

Loop Latency – задержки во всём цикле.

Iteration Latency – задержка в одной итерации

Trip count – количество итераций.

* + 1. Оценка аппаратных ресурсов

Результаты оценки аппаратных ресурсов, требуемых для реализации синтезируемой функции, приведены на Рисунок 5

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 5 Аппаратные ресурсы для solution\_1

Для реализации данного устройства были использованы DSP48E – встроенные умножители в количестве 2, FF-триггеры в количестве 307 шт. и LUT – таблицы перекодировки для реализации логических функций в количестве 172 шт.

* + 1. Планировщик Schedule Viewer

Результаты планирования выполнения функции приведены на Рисунок 6.

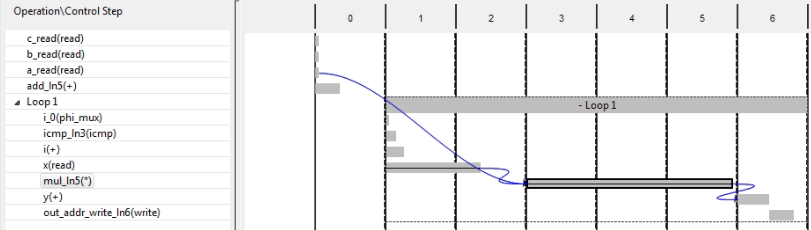


Рисунок 6 Планировщик для solution\_1

В планировщике видно, что чтение a, b, c и массива In происходит на нулевом такте. На первом такте запускается итерация цикла, там же считывается переменная цикла и x. На втором такте чтение x заканчивается. 3, 4, 5 такт производятся вычисление, 6 такте в y записывается полученный результат и запись y в массив out. В итоге вся итерация протекает в 6 тактов.

Изображение выглядит как текст, стол

Автоматически созданное описание

Рисунок 7 Performance Profile для solution1

В performance profile Рисунок 7 видно, что общая задержка составляет 19 тактов, а задержка в цикле 18 тактов. Задержка в каждой итерации составляет 6 тактов. Initiation interval составляет 20 тактов.

* + 1. Распределение ресурсов Resource Viewer

Распределение ресурсов по тактам приведено на Рис. 7

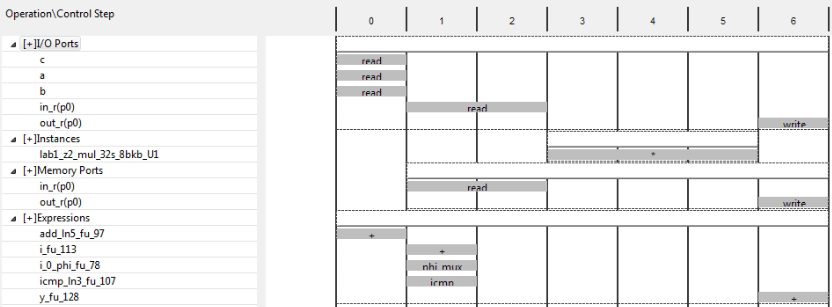


Рисунок 8 Распределение ресурсов по тактам для solution\_1

На нулевом такте считывается a, b, c. На первом такте происходит прибавления к переменной массива. И на первом и втором такте происходит чтение из массива in. На 3, 4, 5 тактах происходит операция умножения. На 6 такте происходит операция сложения и запись результата в массив out.

* 1. Си и RTL моделирование
     1. Анализ результатов

Результаты, приведенные на Рисунок 9, показывают, что синтезированное описание работает в соответствии с ожиданием – в информации указанно что при сравнение ожидаемого результата с фактическим ошибок не обнаружено, функция main вернула 0, следовательно тест успешно пройден.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 9 Результаты моделирования Си и RTL для solution\_1

* + 1. Отчет по быстродействию

Отчет по быстродействию, полученный при совместном Си и RTL моделировании приведен на Рисунок 10. Из отчёта следует что за 20 тактов выполнения одной итерации, 19 тактов занимает latency.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 10 Быстродействие при совместном моделировании solution\_1

* + 1. Временная диаграмма

Временная диаграмма совместного Си|RTL моделирования приведены на Рисунок 11.

Изображение выглядит как текст, монитор, электроника, экран

Автоматически созданное описание

Рисунок 11 Временная диаграмма solution\_1

На временной диаграмме solution1 видно, что каждая итерация протекает 6 тактов. Весь цикл занимает 18 тактов плюс один такт на считывание a, b, c.

1. Второе решение - Solution\_2
   1. Исходные настройки

При создании первого решения задаются следующие настройки:

* clock period: 12,
* clock uncertain: 0.1,
* part: xa7a12tcsg325-1q.
  1. Синтез
     1. Оценка быстродействия

Результаты оценки быстродействия Timing/summary и Latency/Summary приведены на Рисунок 12.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 12 Оценка быстродействия для solution\_2

Target – планируемое время на один такт.

Estimated – оценочное время.

Latency (cycle) – количество тактов latency за один цикл.

Latency (absolute) – время затраченное на latency.

Loop Latency – задержки во всём цикле.

Iteration Latency – задержка в одной итерации

Trip count – количество итераций.

* + 1. Оценка аппаратных ресурсов

Результаты оценки аппаратных ресурсов, требуемых для реализации синтезируемой функции, приведены на Рисунок 13.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 13 Аппаратные ресурсы для solution\_2

Для реализации данного устройства были использованы DSP48E – встроенные умножители в количестве 2, FF-триггеры в количестве 106 шт., LUT – таблицы перекодировки для реализации логических функций в количестве 130 шт.

* + 1. Планировщик Schedule Viewer

Результаты планирования выполнения функции приведены на Рисунок 14.

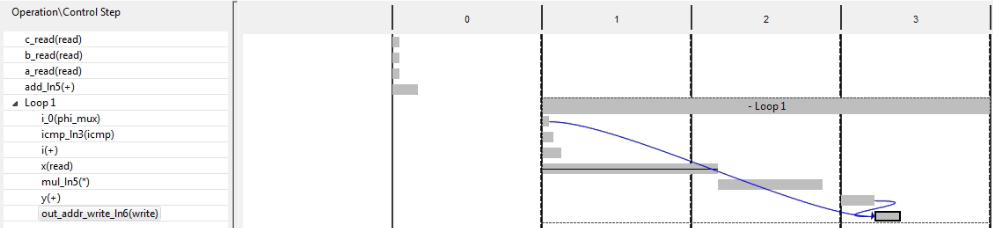


Рисунок 14 Планировщик для solution\_2

В планировщике видно, что на нулевом такте считывается a, b, c и производится операция сложения. На первом такте начинается итерация цикла – в переменную цикла записывается значение и считывается x. X считывается до начала второго такта. Так же на втором такте производится умножение и на третьем такте результат записывается в y и далее на том же такте записывается в массив out.

* + 1. Распределение ресурсов Resource Viewer

Распределение ресурсов по тактам приведено на Рисунок 15.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 15 Распределение ресурсов по тактам для solution\_2

В распределении ресурсов наглядно видно, что на нулевом такте происходит чтение a, b, c и сложение b и с. На первом такте считывается данные из массива in, к переменной цикла добавляется значение. Во втором такте так же считываются данные из массива In и происходит операция умножения. В третьем такте результат присваивается к y и далее y записывается в массив out.

* 1. Си и RTL моделирование
     1. Анализ результатов

Результаты, приведенные на Рисунок 16, показывают, что синтезированное описание работает в соответствии с ожиданием – в информации указанно что при сравнение ожидаемого результата с фактическим ошибок не обнаружено, функция main вернула 0, следовательно тест успешно пройден.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 16 Результаты моделирования Си и RTL для solution\_2

* + 1. Отчет по быстродействию

Отчет по быстродействию, полученный при совместном Си и RTL моделировании приведен на Рисунок 17. Из отчёта следует что за 11 тактов выполнения одной итерации, 10 тактов – latency.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 17 Быстродействие при совместном моделировании для solution\_2

* + 1. Временная диаграмма

Временная диаграмма совместного Си|RTL моделирования приведены на Рисунок 18.

Изображение выглядит как текст, электроника, компьютер

Автоматически созданное описание

Рисунок 18 Временная диаграмма для solution\_2

На временной диаграмме solution2 видно, что каждая итерация протекает 3 такта. Весь цикл занимает 9 тактов плюс один такт на считывание a, b, c.

1. Выводы
   1. Сравнение параметров двух решений

Результаты сравнения параметров двух решений приведены на Рисунок 19.

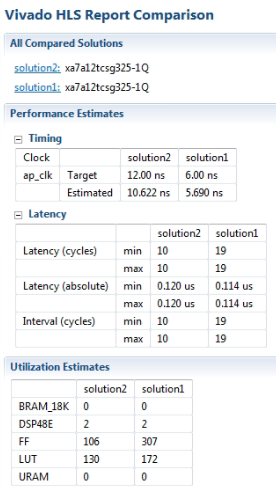


Рисунок 19 Сравнение двух решений

Один такт solution2 занимает 12 ns, а Solution1 – 6 ns. Предполагаемое время solution2 – составляет 10.622 ns, а solution2 – 5.69 ns. Latency всего цикла первого solution занимает 19 тактов по 6 таков на 3 итерации плюс такт на считывание. Latency solution2 всего цикла занимает 10 тактов, по три такта на каждую итерацию плюс один такт на считывание. Время latency составляет в первом solution 0.114 us а второго latency 0.120 us. По аппаратным затратам второе решение использует меньше FF триггеров и LUT таблиц.

* 1. Какое из двух решений имеет меньший II

Второе решение имеет 11 тактов initiation interval и составляет 0.140 us, а первое решение - 20 тактов и составляет 0.120 us.

* 1. Какое из двух решений обеспечивает большее быстродействие?

Первое решение быстрее чем второе. Если судить по временной диаграмме, то первое решение быстрее второго на 75ns.

* 1. **Какое из двух решений требует меньших аппаратных затрат?**

Второе решение имеет меньше FF-триггеров примерно в половину раз, и меньше LUT-таблиц. DSP48E – встроенных умножителей одинаковое количество.

* 1. Какое из двух решений является более оптимальным?

Оба решения примерно одинаковы. Одно выигрывает по быстродействию второе по аппаратным затратам.