Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**Отчет по лабораторной работе №1\_2**

**Курс: «Проектирование реконфигурируемых гибридных вычислительных систем»**

**Тема: «Введение в Vivado HLS»**

Выполнил студент гр. 3540901/81501 Селиверстов Я.А.

(подпись)

Руководитель Антонов А.П.

(подпись)

“\_\_\_” \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2019 г.

Санкт – Петербург

2019

Оглавление

[Задание 3](#_Toc21441146)

[Решение №1 5](#_Toc21441147)

[Моделирование 5](#_Toc21441148)

[Синтез 5](#_Toc21441149)

[C/RTL моделирование 8](#_Toc21441150)

[Решение №2 9](#_Toc21441151)

[Моделирование 9](#_Toc21441152)

[Синтез 10](#_Toc21441153)

[C/RTL моделирование 12](#_Toc21441154)

[Выводы 13](#_Toc21441155)

# Задание

Создать проект, подключив готовые файлы исходного кода устройства и теста к нему.

Исходный код функции:

|  |
| --- |
| void lab1\_2 **(**int in**[**3**],** char a**,** char b**,** char c**,** int out**[**3**])** **{**  int x**,**y**;**  **for(**int i **=** 0**;** i **<** 3**;** i**++)** **{**  x **=** in**[**i**];**  y **=** a**\***x **+** b **+** c**;**  out**[**i**]** **=** y**;**  **}**  **}** |

Исходный код теста:

|  |
| --- |
| **#include** <stdio.h>  **int** **main**()  {  **int** In[3] = {1,3,9};  **int** inA, inB, inC;  **int** res[3];  // For adders  **int** refOut[9] = {30, 40, 70, 90, 140, 290, 150, 240, 510};  **int** pass;  **int** i;  inA = 5;  inB = 10;  inC = 15;  **for** (i=0; i<3; i++)  {  lab1\_2(In, inA, inB, inC, res);  **for** (**int** j=0; j<3; j++)  {  **fprintf**(stdout, " %d\*%d+%d+%d=%d \n", inA, In[j], inB, inC, res[j]);    // Test the output against expected results  **if** (res[j] == refOut[i\*3+j])  pass = 1;  **else**  pass = 0;  }  inA=inA+20;  inB=inB+20;  inC=inC+20;  }  **if** (pass)  {  **fprintf**(stdout, "----------Pass!------------\n");  **return** 0;  }  **else**  {  **fprintf**(stderr, "----------Fail!------------\n");  **return** 1;  }  } |

Создать 2 решения для синтеза устройства на основе *xa7a12tcsg325-1q*: задать clock period 6 и clock period 10, а также clock uncertain 0.1 в обоих случаях.

Для обоих решений осуществить моделирование и синтез, привести в данном отчете:

• performance estimates=>summary

• 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)

* На скриншоте показать Latency
* На скриншоте показать Initiation Interval

Также для обоих решений осуществить C|RTL моделирование:

• Отобразить два цикла обработки на одном экране

* На скриншоте показать Latency
* На скриншоте показать Initiation Interval

Написать выводы, где среди прочего объяснить отличие двух решений.

# Решение №1

При создании решения зададим настройки: clock period 6, clock uncertain 0.1, устройство *xa7a12tcsg325-1q*.

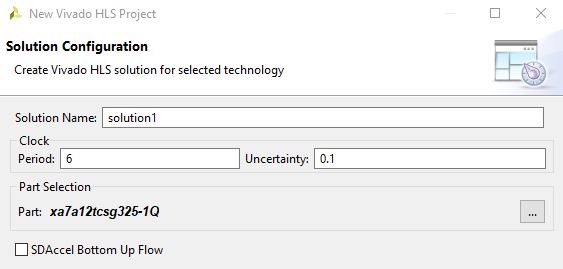


Рис. 1. Конфигурация проекта

## Моделирование

При запуске моделирования можно увидеть, что тест успешно пройден:



## Синтез

Приведем в отчете требуемые данные о проекте:

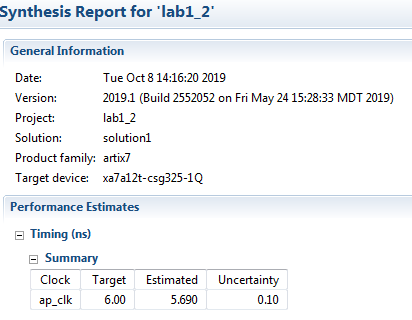


Рис. 2. Информация о проекте

Здесь можно увидеть, что достигнутая задержка равна 5.690 + 0.1, что укладывается в заданные нами требования к тактовой частоте.

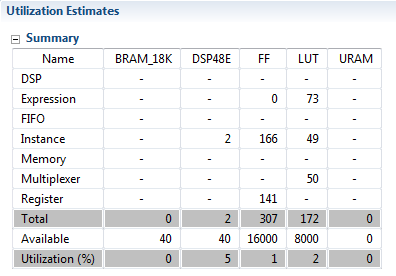


Рис. 3. Занимаемые ресурсы

Данный проект займет на микросхеме 2 DSP блока (в которых будут использованы и сумматоры и умножитель), 307 триггеров для хранения чисел, и 172 LUT.

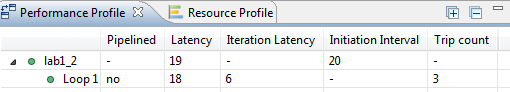


Рис. 4. Профиль производительности

На этом рисунке видно, что задержка получения выходного значения составляет 19 тактов с момента старта, а задержка после старта до готовности приема новых данных – 20. Покажем эти интервалы на временной диаграмме:

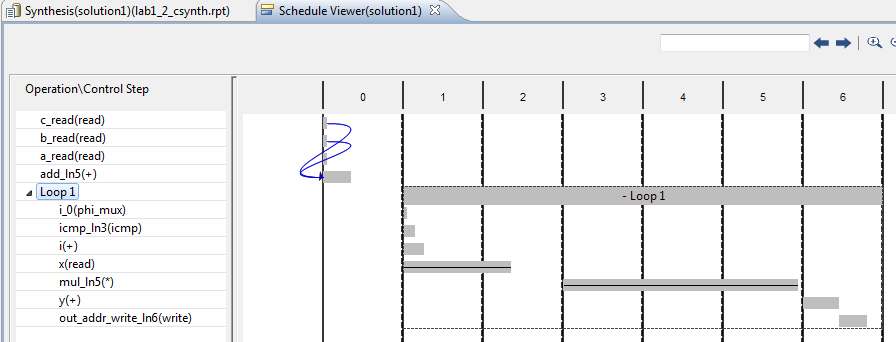


Рис. 5. Временная диаграмма

Здесь мы видим весь процесс получения результата. На первом такте происходит считывание операторов А, В и С, а также начинается их сложение. На втором такте начинает цикл длиной в 6 тактов, который повторяется 3 раза (всего 18 тактов). Таким образом суммарная задержка latency = 19, а со следующего 20-го такта можно подавать следующие данные (II = 20).

Приведем профиль ресурсов:

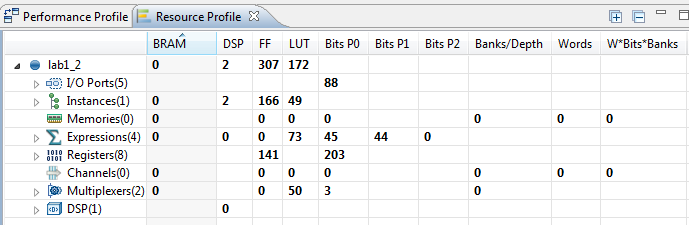


Рис. 6. Профиль ресурсов

Здесь можно увидеть те же числа, что и в отчете синтезатора.

## C/RTL моделирование

При совместном моделировании, программа отобразила те же самые, ожидаемые нами значения Latency и II:

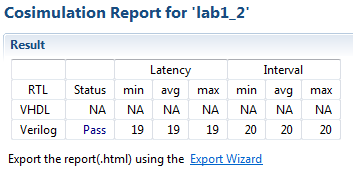


Рис. 7. Результаты C/RTL моделирования

Покажем временную диаграмму совместного моделирования с отмеченными на ней Latency и II:

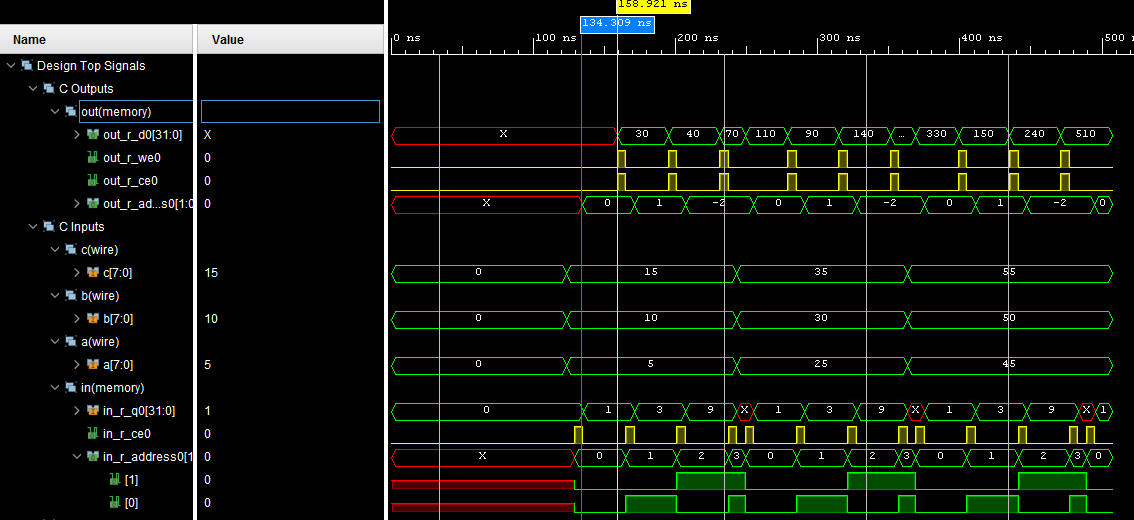


Рис. 8. Временная диаграмма совместного моделирования

# Решение №2

## Моделирование

Создадим второе решение для данного проекта. Его настройки:

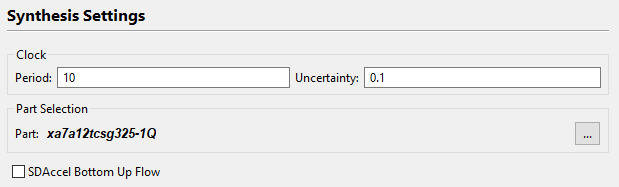
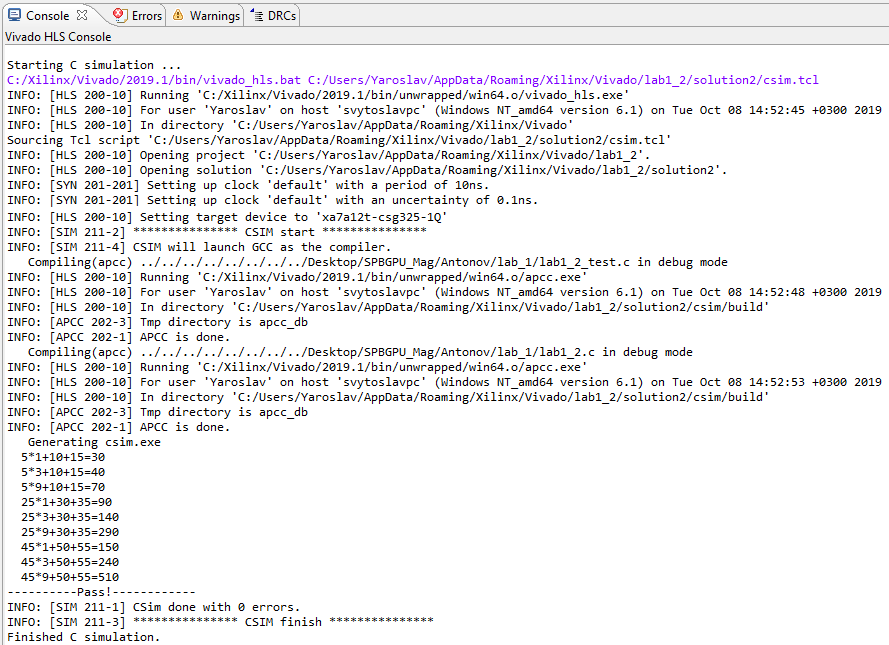


Рис. 10. Конфигурация проекта

Настройки этого решения изменены согласно заданию.



Моделирование второго решения также прошло успешно.

## Синтез

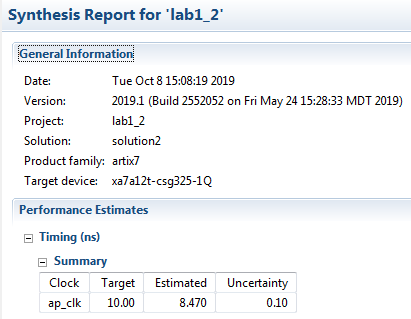


Рис. 11. Производительность

На этом рисунке видно, что данное решение также укладывается в установленный период тактовой частоты, однако задержки сильно отличаются, что означает, что изменения периода тактовой частоты повлекло за собой существенные изменения устройства.

На примере затрачиваемых на решение ресурсов это также хорошо видно:

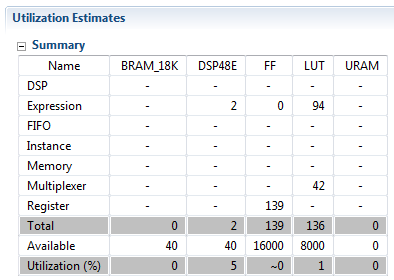


Рис. 12. Затрачиваемые ресурсы

В сравнении с предыдущим решением в схеме количество триггеров сократилось на 168 (∆ FF = FF1-FF2 = 307-139 =168), а LUT на 36 (∆ LUT = LUT1 – LUT2 = 172-132 =36).

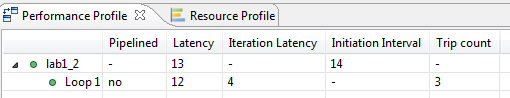


Рис. 13. Профиль производительности

Сразу видно, что задержка цикла уменьшилась с 6 до 4 тактов, а новые данные можно подавать уже через 14 тактов. Покажем это на временной диаграмме:

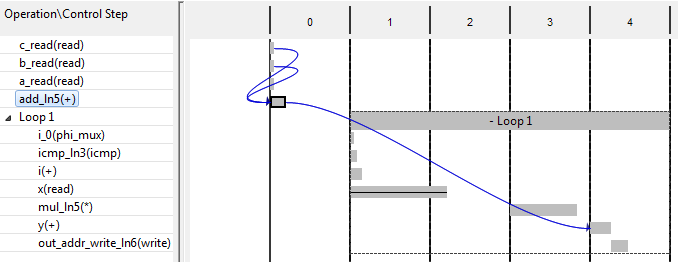


Рис. 14. Временная диаграмма

Если изучить эту диаграмму подробнее, то можно сказать, что умножение занимает значительно меньше тактов.

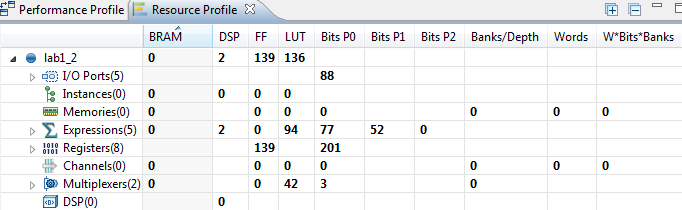


Рис. 15. Профиль ресурсов

Здесь мы также видим отличия, согласно общему отчету о затраченных ресурсах.

## C/RTL моделирование

При осуществлении совместного моделирования программа показала ожидаемые результаты Latency, но II также стал нулевым:

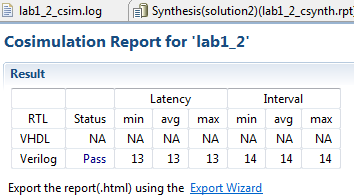


Рис. 16. C/RTL моделирование

Покажем временную диаграмму совместного моделирования:

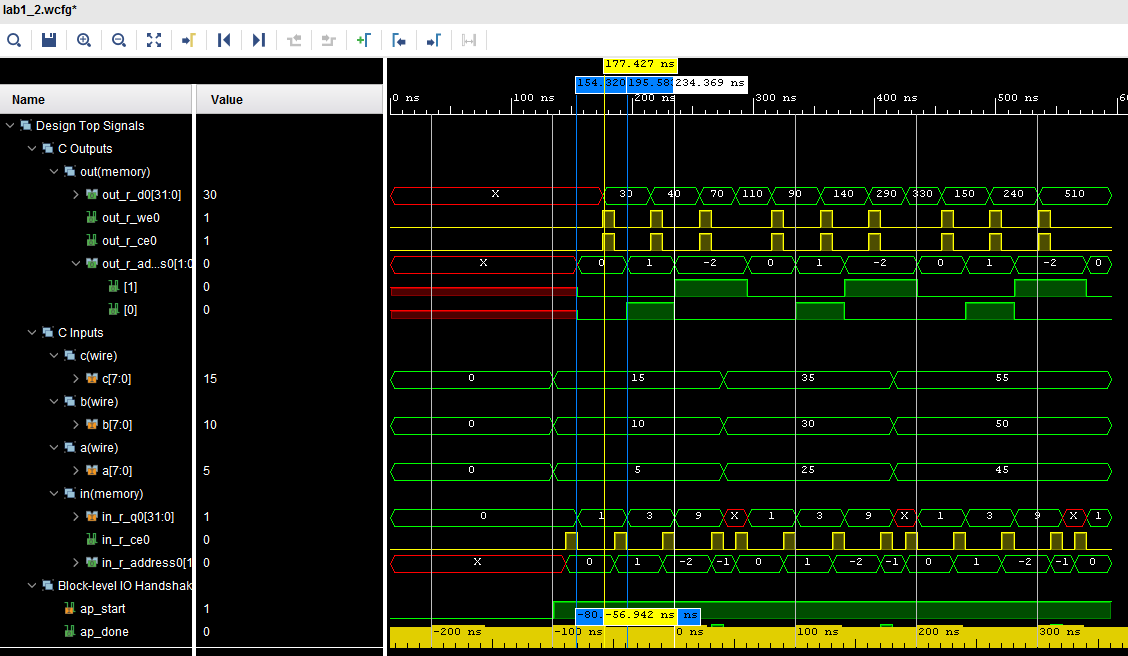


Рис. 17. Временная диаграмма совместного моделирования

Здесь видно, что задержка подачи новых данных сократилась до 14 циклов.

# Выводы

В данной работе главным образом были рассмотрены синтезируемые устройства при наличии в программе циклов. В данном примере видно, что при большем периоде тактов, программа сумела уместить расчет одного цикла в меньшее число тактов. Таким образом, получены 2 решения: первое – полный цикл выполнения 20 тактов, а максимальная задержка обработки сигнала на такте составляет 5.690 нс, и второе – 14 тактов, задержка в котором уже 8.470 нс, но все выполняется всего за меньшее число тактов.