Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**Отчет по лабораторной работе №2\_1**

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

**Тема: «Введение в Vivado HLS Tool CLI Flow»**

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

(подпись)

Руководитель Антонов А.П.

(подпись)

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

Санкт – Петербург

2019

Оглавление

[**Задание** 3](#_Toc26472961)

[**Решение** 5](#_Toc26472962)

[**Выводы** 13](#_Toc26472963)

# **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, а также clock uncertainty 0.1.

Создать скрипт автоматизирующий процесс:

* Создать проект lab2\_1
* Подключить файл lab2\_1.c (папка source )
* Подключить тест lab2\_1\_test.c (папка source)
* Микросхема: xa7a12tcsg325-1q

Сделать solution1

* задать: clock period 6; clock\_uncertainty 0.1
* осуществить моделирование
* осуществить синтез
* открыть GUI
* проверить работу созданного скрипта.

Не стирая результаты работы предыдущего запуска скрипта, запустить

скрипт еще раз и проверить корректность работы при повторном запуске.

# **Решение**

При создании решения зададим настройки: clock period 6, clock uncertain 0.1,

устройство xa7a12tcsg325-1q.

Для автоматизации создания проекта, напишем tcl скрипт, содержащий

следующие команды:

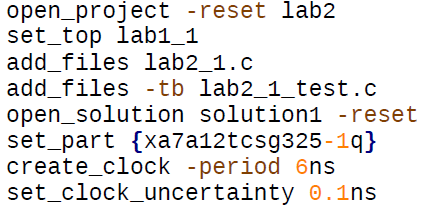


Рис.1. Скрипт создания проекта

Команды скрипта полностью соответствуют настройкам в GUI.

Существующий набор команд можно получить, выполнив команду help при работе в режиме интерпретатора (vivado\_hls -i).

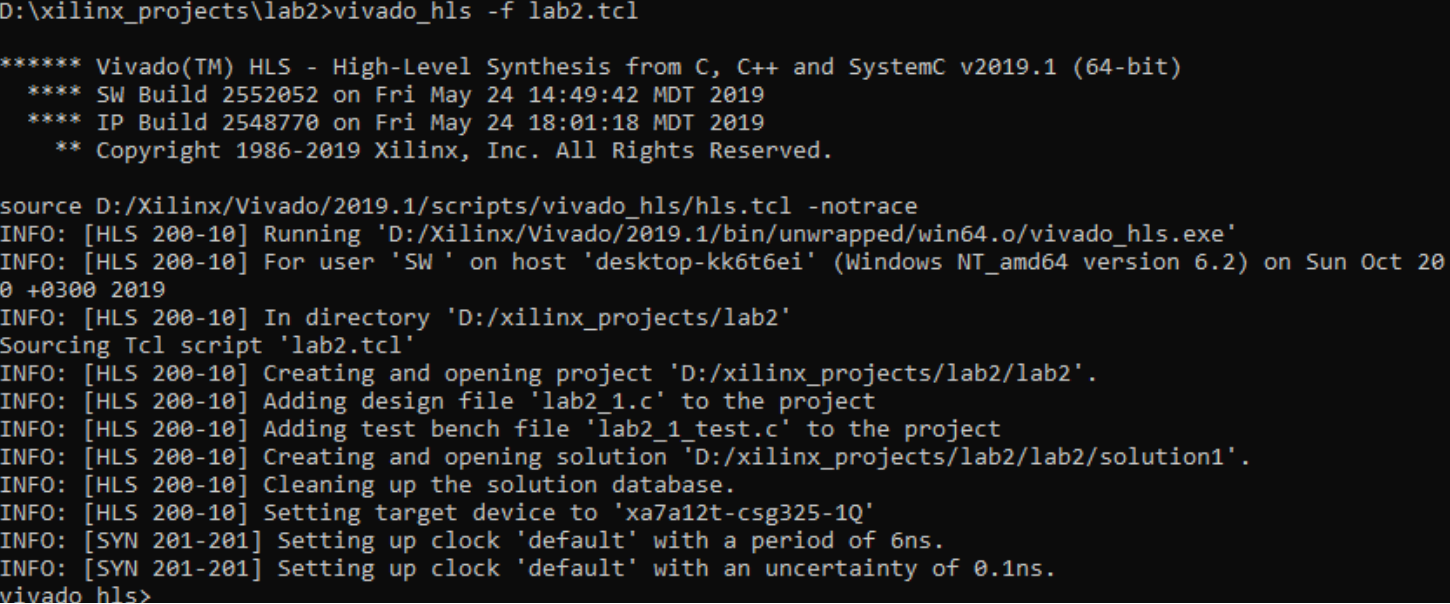


Рис. 2 Загрузка tcl скрипта

Проверим на рис.3, что проект создался верно: подключены исходные и тестовые файлы, появился раздел Solution.

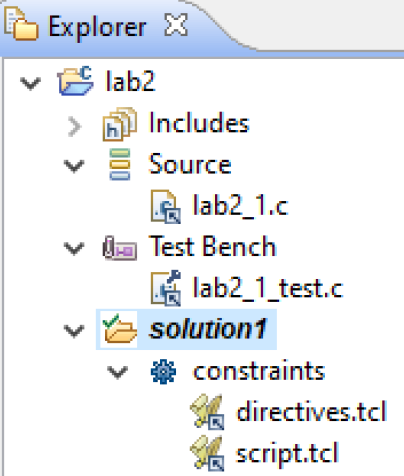


Рис. 3: Структура созданного проекта

Запустим интерактивный режим и выполним моделирование. Результаты успешного моделирования представлены на Рис. 4.

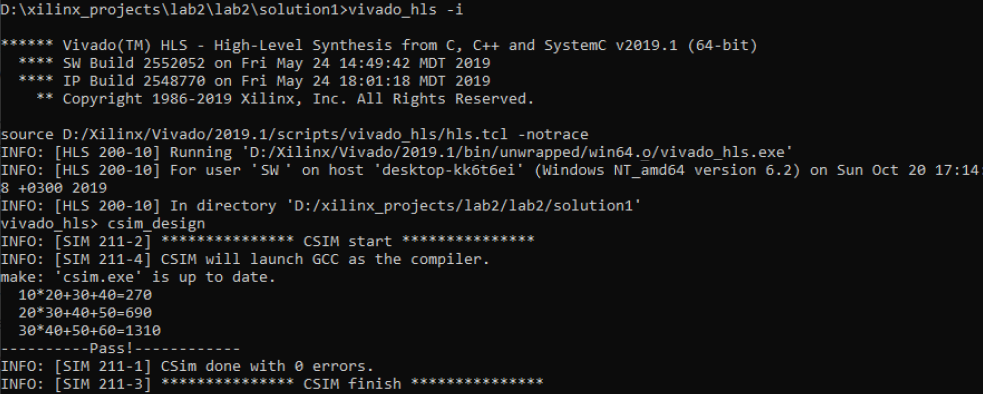


Рис. 4. Результаты успешного моделирования.

На следующем шаге выполним синтез с помощью команды csynth\_design.

Результаты синтеза программы представлены на Рис.5.

На рисунке видны логи, демонстрирующее процесс имплементации lab\_1 и генерации RTL для модели lab\_1.

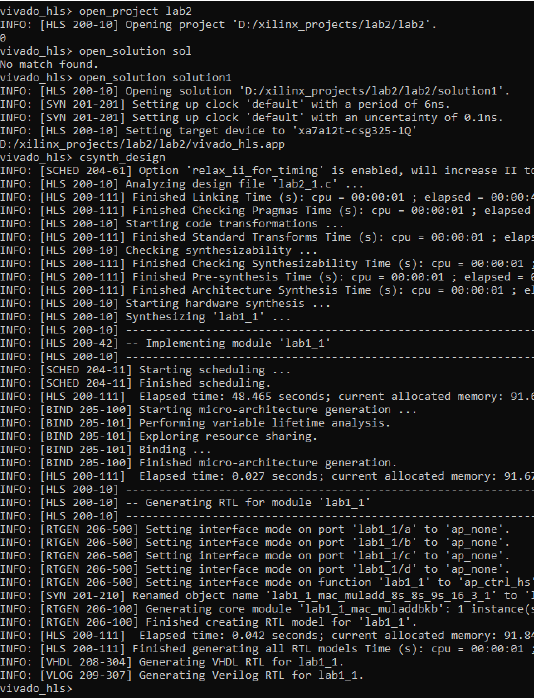


Рис. 5: Лог-отчет результатов синтеза

Откроем GUI и рассмотрим результаты производительности.

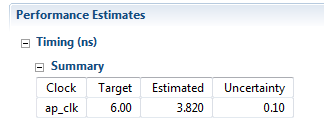


Рис. 6. Производительность

Здесь можно увидеть, что достигнутая задержка равна 3.820 + 0.1, что укладывается в заданные нами требования к тактовой частоте

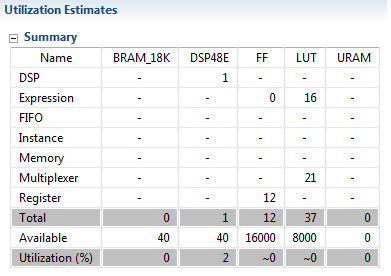


Рис. 7. Занимаемые ресурсы

Данный проект займет на микросхеме 1 DSP блок (в котором будут использованы и сумматоры и умножитель), 12 триггеров для хранения чисел, и 37 LUT.

Профиль производительности и профиль ресурсов представлен на рисунке 8 и рисунке 9.

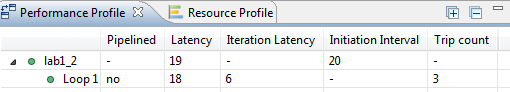


Рис. 8. Профиль производительности

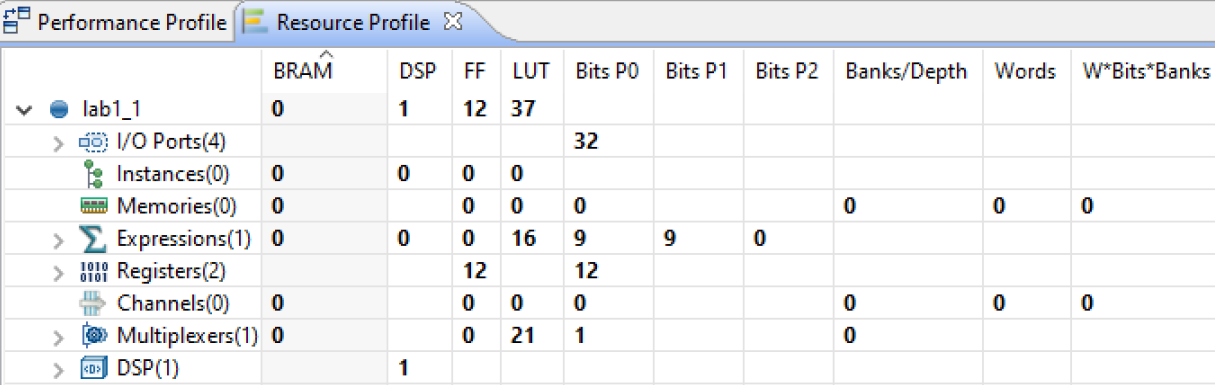


Рис. 9. Профиль ресурсов

Проанализировав отчет производительности, представленного на рисунке 9 , можем сделать вывод, что задержка до получения результата — 2 такта, еще через 1 такт можно записывать новые данные.

Временная диаграмма выполнения операций представлена на Рис.10.

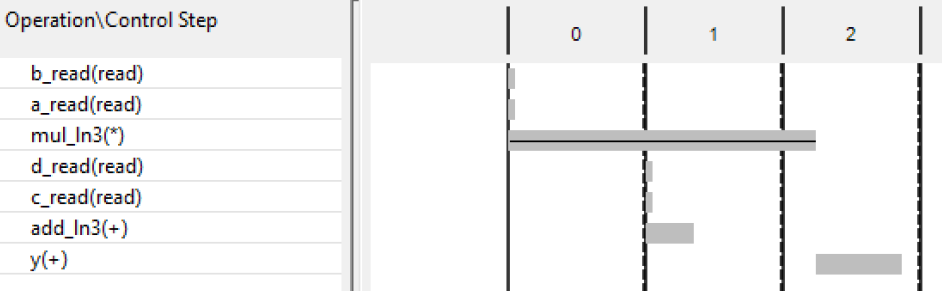


Рис. 10. Временная диаграмма выполнения операций

Проведем RTL моделирование с использованием команды cosim\_design. Результат лог-отчета об успешном моделировании представлен на рисунке 11.

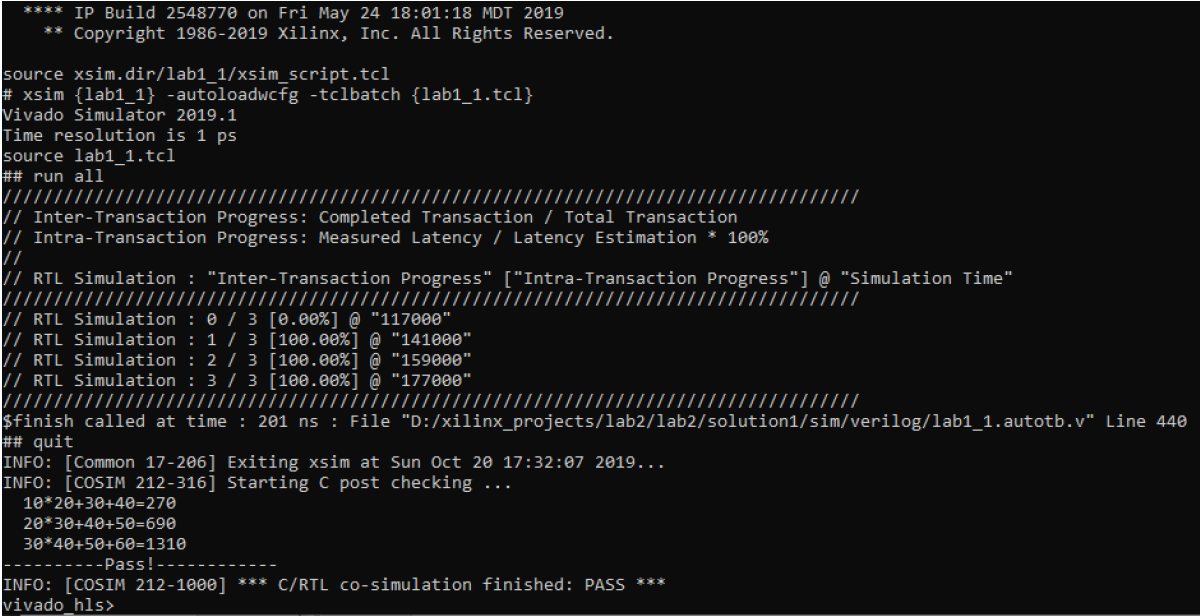


Рис. 11: Лог-отчет результатов моделирования

Выполним экспорт с использованием команды export\_design. Результат лог-отчета об успешном экспорте представлен на рисунке 12.

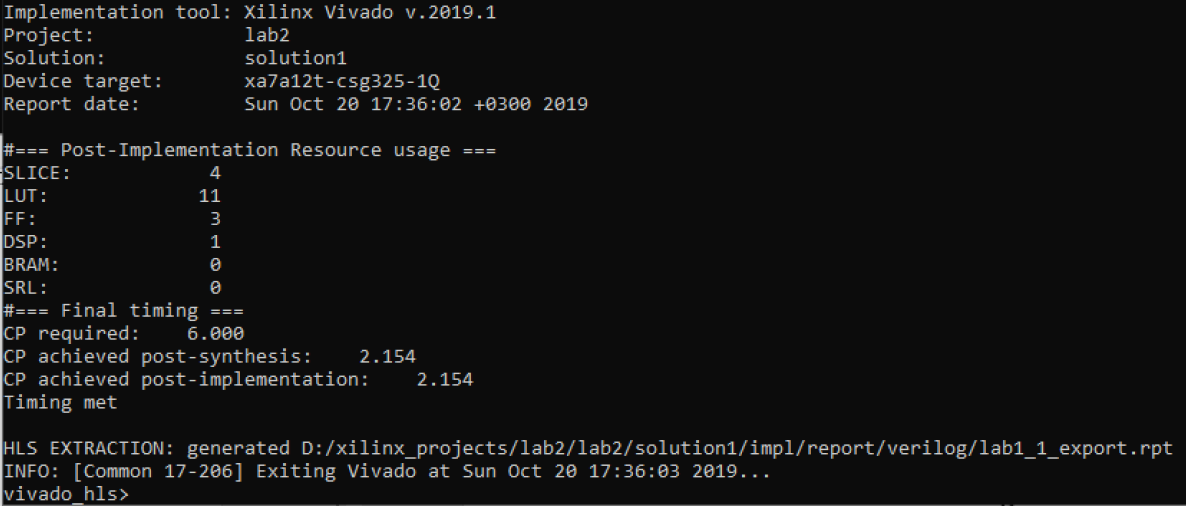


Рис. 12: Лог-отчет результатов экспорта

Дерево структуры выполненного проекта представлено на Рис. 13.

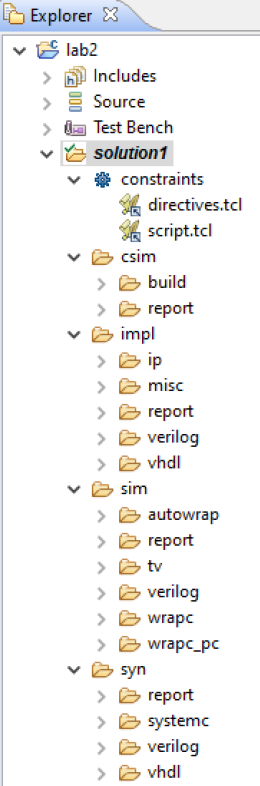


Рис. 13.Дерево проекта

Скрипт для автоматизации создания проекта можно расширить последними командами и проверить его работоспособность.

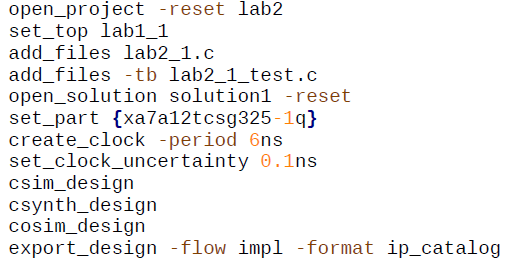


Рис. 13.Скрипт для создания проекта

При запуске скрипта получили точно такие же результат, как и продемонстрированы выше.

Часть логов, отвечающая за С/RTL моделирование и генерацию приведена на рисунках 14 и 15.

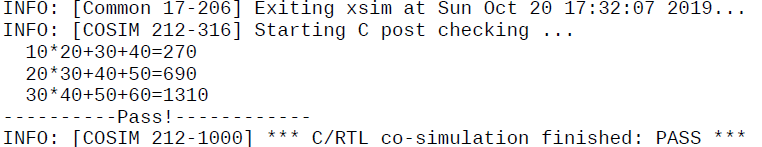


Рис.14. Логи об успешном С/RTL моделировании

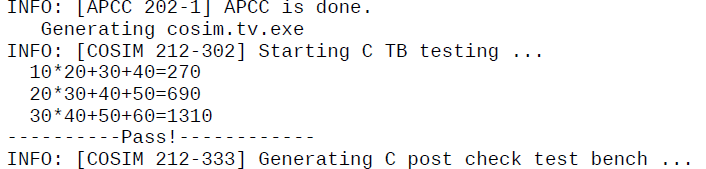


Рис.15. Логи об успешной генерации

# **Выводы**

В ходе выполнения лабораторной работы были изучены методы работы с Vivado HLS Command Prompt. Был создан проект с решением по заданным исходным, тестовым файлам и параметрам. Процесс создания проекта и моделирования был автоматизирован при помощи tcl скрипта, который можно использовать для подгрузки проекта прямо из командной строки. Результаты работы скрипта не изменились после его перезапуска.