Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**Отчет по лабораторной работе №6\_2**

**Курс: «Проектирование реконфигурируемых гибридных**

**вычислительных систем»**

**Тема: «Port-level IO protocols»**

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

(подпись)

Руководитель Антонов А.П.

(подпись)

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

Санкт – Петербург

2019

ОГЛАВЛЕНИЕ

[Задание 3](#_Toc28921937)

[1. Первое решение 5](#_Toc28921938)

[1.1. Исходный код программы и теста 5](#_Toc28921939)

[1.2. Моделирование 6](#_Toc28921940)

[1.3. Синтез 6](#_Toc28921941)

[1.4. C|RTL моделирование 8](#_Toc28921942)

[2. Второе решение 10](#_Toc28921943)

[2.1. Настройки второго решения 10](#_Toc28921944)

[2.2. Моделирование 11](#_Toc28921945)

[2.3. Синтез 11](#_Toc28921946)

[2.4. C|RTL моделирование 13](#_Toc28921947)

[3. Третье решение 15](#_Toc28921948)

[3.1. Настройки третьего решения 15](#_Toc28921949)

[3.2. Моделирование 16](#_Toc28921950)

[3.3. Синтез 16](#_Toc28921951)

[3.4. C|RTL моделирование 19](#_Toc28921952)

[4. Четвертое решение 20](#_Toc28921953)

[4.1. Настройки четвертого решения 20](#_Toc28921954)

[4.2. Моделирование 21](#_Toc28921955)

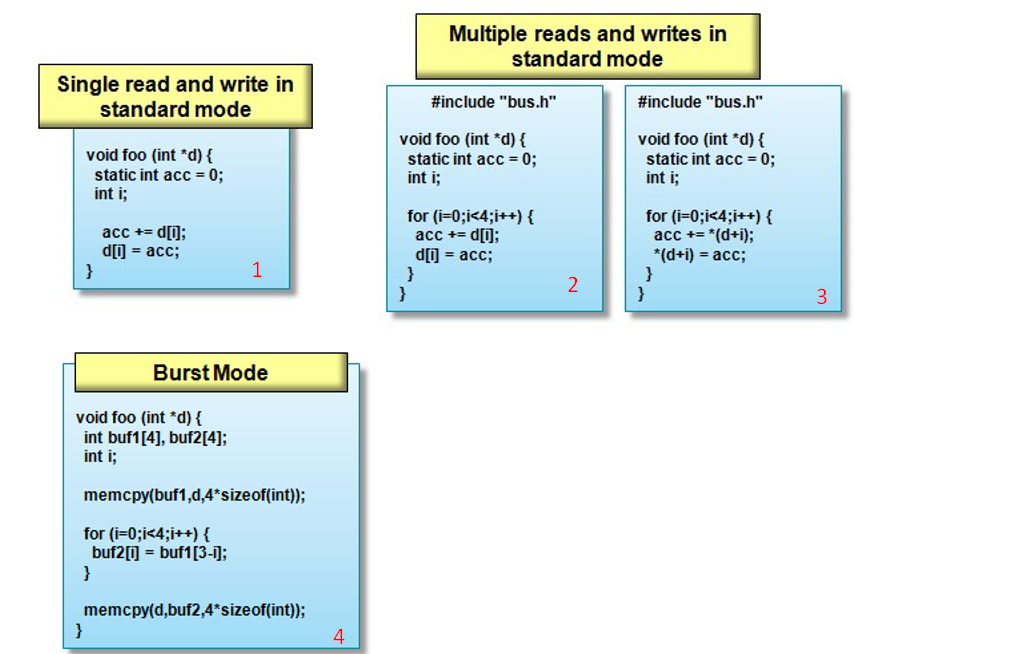
[4.3. Синтез 21](#_Toc28921956)

[4.4. C|RTL моделирование 23](#_Toc28921957)

[4.Выводы 25](#_Toc28921958)

**Задание**

* Создать проект lab6\_2
* Микросхема: xa7a12tcsg325-1q
* Создать четыре функции на основе слайда (функция foo\_1, foo\_2, foo\_3, foo\_4). *При желании можно сделать 4 отдельные лабораторные работы lab6\_2\_1…lab6\_2\_4 но они все будут очень похожи*.



* Создать тест lab6\_2\_test.c для проверки функций выше (это может быть один тест или разные тесты. Функция main д.б. одна, а в ней использовать проверяемый модуль. Д.б. вывод результатов в консоль.).
* Для каждой функции сделать свой solution
  + задать: clock period 10; clock\_uncertainty 0.1
  + Задать протокол
    - a: ap\_bus
  + осуществить моделирование (с выводом результатов в консоль)
  + осуществить синтез
    - привести в отчете:
      * performance estimates=>summary
      * utilization estimates=>summary
      * Performance Profile
      * interface estimates=>summary
        + объяснить какой интерфейс использован для блока (и какие сигналы входят) и для портов (и какие сигналы входят).
      * 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
  + Осуществить C|RTL моделирование
    - Привести результаты из консоли
    - Открыть временную диаграмму (все сигналы)
      * Отобразить два цикла обработки на одном экране
        + На скриншоте показать Latency
        + На скриншоте показать Initiation Interval
* Выводы
  + Объяснить отличие процедур обращения к элементам массива для каждого случая

# 1. Первое решение

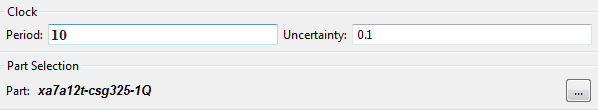


Рисунок 1.1. Параметры первого решения

## 1.1. Исходный код программы и теста

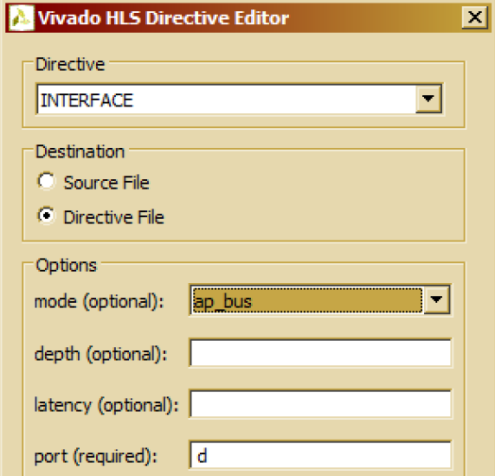
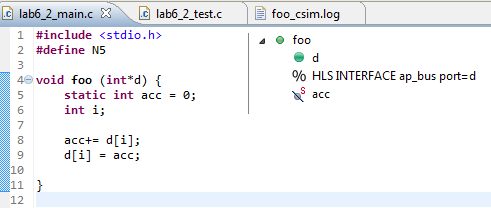


Рисунок 1.2. Исходный код синтезируемой функции

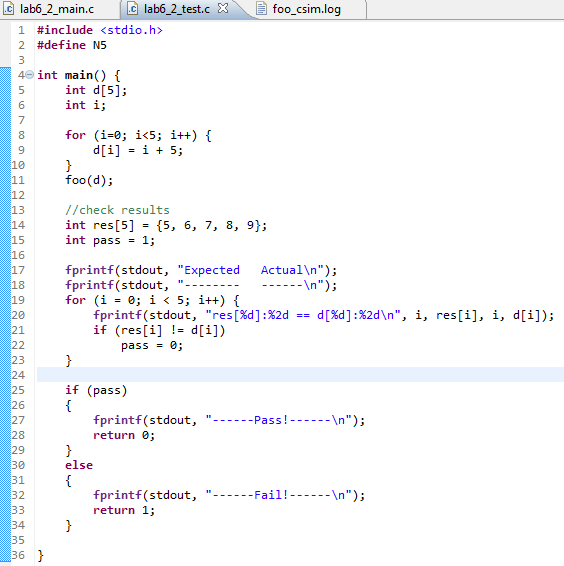


Рисунок 1.3. Исходный код теста

## 1.2. Моделирование

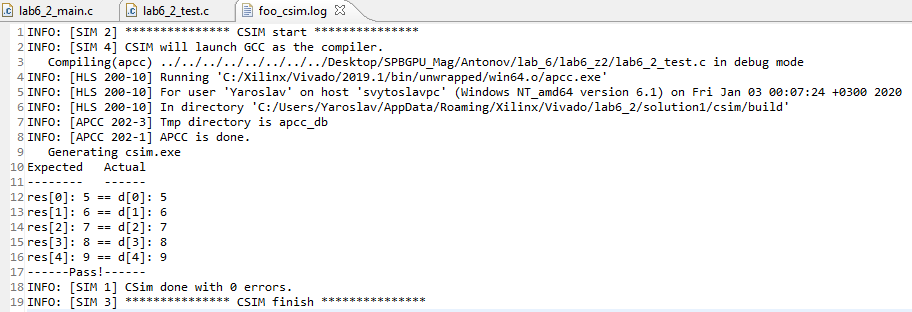


Рисунок 1.4. Результат успешного моделирования

Моделирование прошло успешно.

## 1.3. Синтез

Данные о проекте

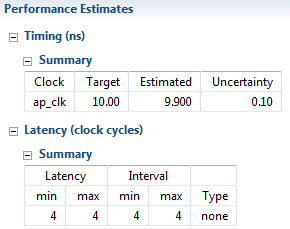


Рисунок 1.5. Performance estimates – summary

Здесь можно увидеть, что достигнутая задержка равна 9.900 + 0.1, что укладывается в заданные нами требования к тактовой частоте.

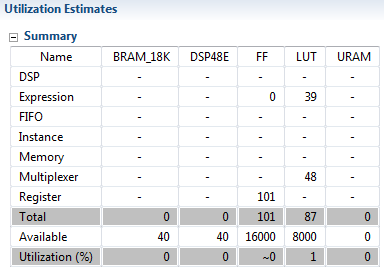


Рисунок 1.6. Utilization estimates – summary

Данный проект займет на микросхеме 101 регистр для хранения чисел и 87 LUT.

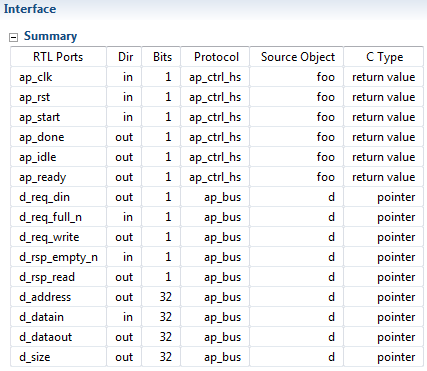


Рисунок 1.7. Interface estimates – summary

Для расчета схемы требуется более одного такта. На рисунке 1.7 представлены интерфейсы, которые используются в синтезированном устройстве. Видно, что в схеме применяется протокол ap\_bus. Порты d\_adress, d\_datain, d\_dataout и d\_size 32-битные.

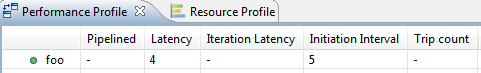


Рисунок 1.8. Performance Profile

На рисунке 1.8 видно, что задержка получения выходного значения составляет 4 такта с момента старта, а задержка после старта до готовности приема новых данных – 5.

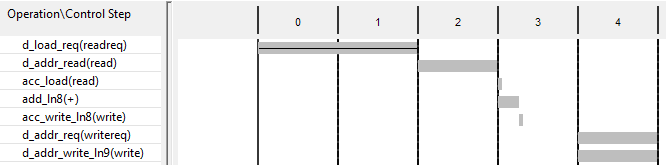


Рисунок 1.9. Schedule viewer

На рисунке 1.9 представлена диаграмма Schedule viewer. На ней видно, что величина Latency составляет 4 такта, а Iteration latency – 5 тактов.

Далее рассмотрим профиль ресурсов:

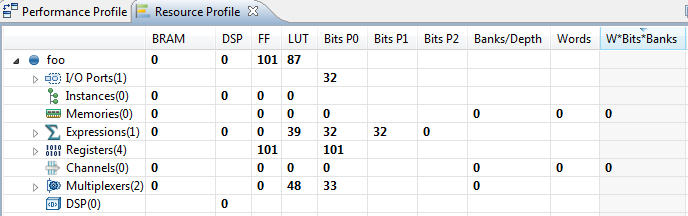


Рисунок 1.10. Resource Profile.

Значения в Resource Profile на рис. 1.10 совпадает с результатами синтеза на рис. 1.6.

## 1.4. C|RTL моделирование

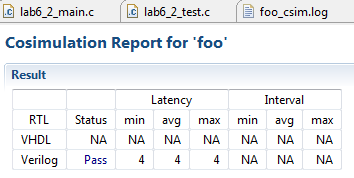


Рисунок 1.11. Cosimulation Report

При совместном моделировании, программа на рисунке 1.11 отобразила те же самые, ожидаемые нами значения Latency и II.

Покажем на рисунке 1.12 временную диаграмму совместного моделирования с отмеченными на ней Latency и II:

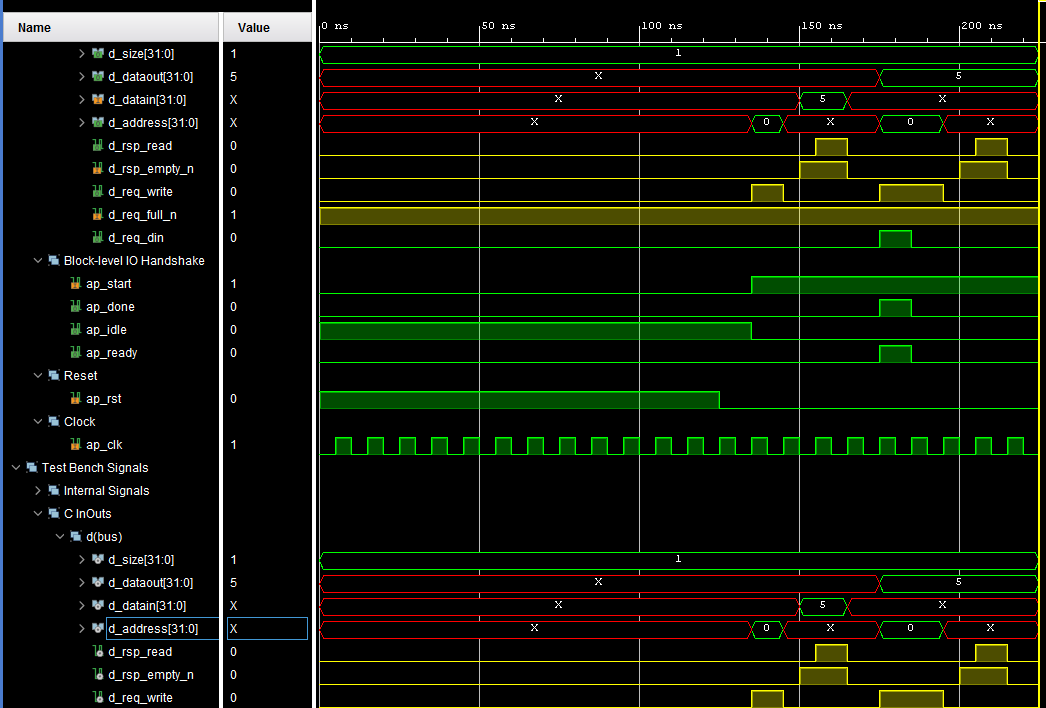


Рисунок 1.12. Design Top Signals

# Второе решение

## 2.1. Настройки второго решения

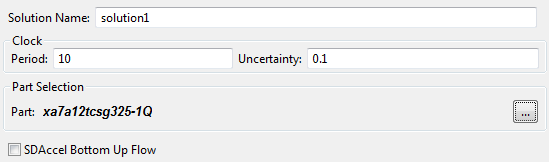


Рисунок 2.0. Параметры второго решения

Программа, тест к ней и интерфейс (директива) представлены на рисунках 2.1 и 2.2.

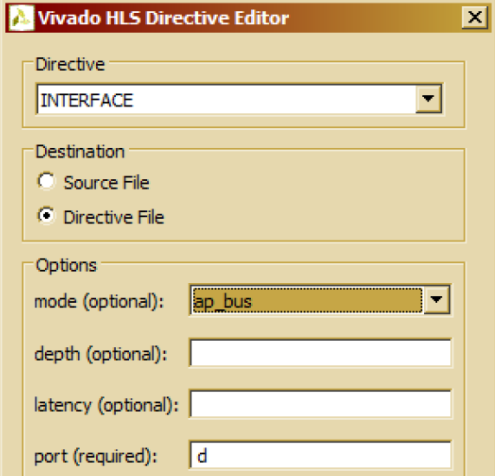
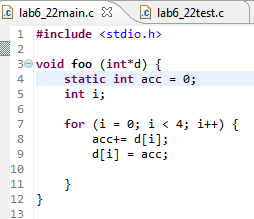


Рисунок 2.1Исходная программа.

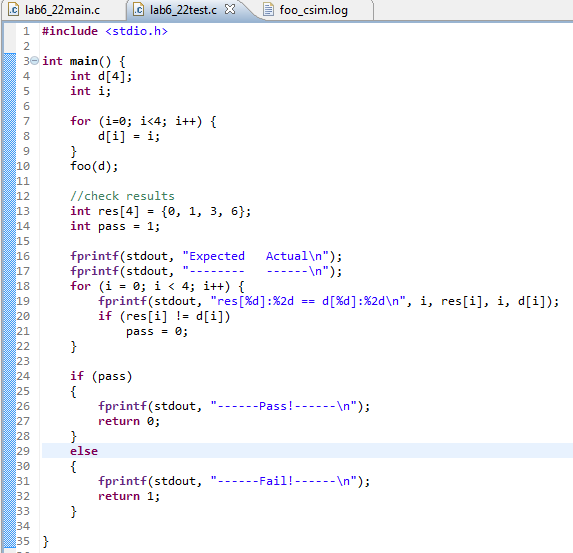


Рисунок 2.2. Тест к программе.

## 2.2. Моделирование

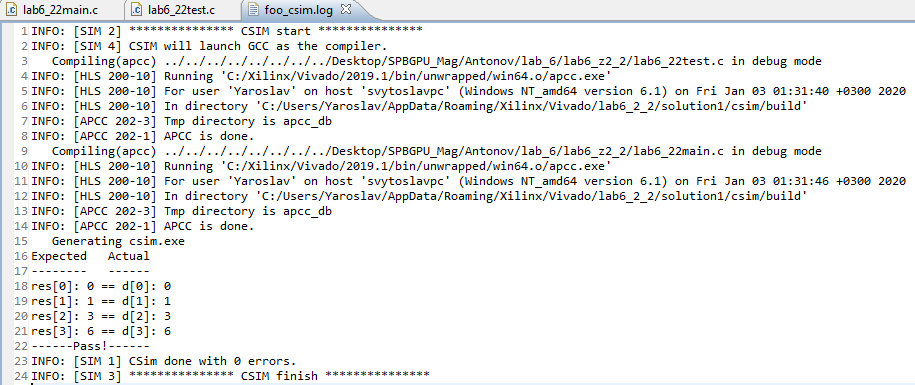


Рисунок 2.3 Результат успешного моделирования

Моделирование второго решения также прошло успешно.

## 2.3. Синтез

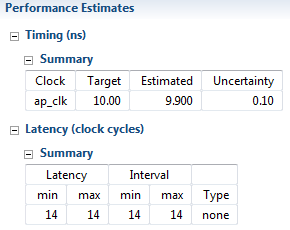


Рисунок 2.4. Performance estimates – summary

На рисунке 2.4. можно увидеть, что достигнутая задержка равна 9.900 + 0.1, укладывается в заданные требования тактовой частоты. А величина Latency 14.

Использование ресурсов:

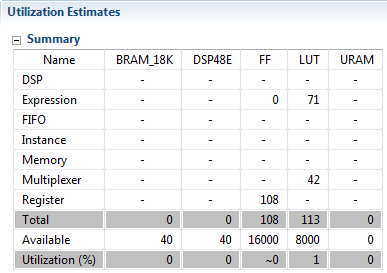


Рисунок 2.5. Utilization estimates – summary

На рисунке 2.5. видно, что данный проект теперь займет на микросхеме 108 регистров для хранения чисел, и 113 LUT.

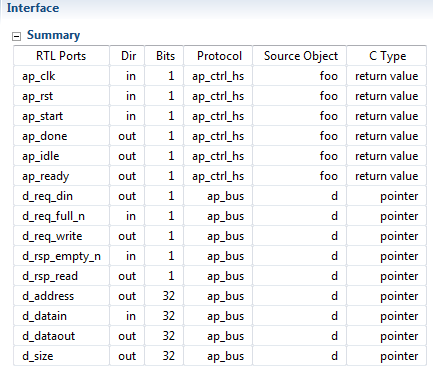


Рисунок 2.6. Interface Summary.

Полученные значения Interface Summary на рисунке 2.6. совпадают с прошлым решением, представленным на рисунке 1.7. На рисунке 2.6 также представлены интерфейсы, которые используются в синтезированном устройстве. Видно, что в схеме применяется те же протоколы ap\_bus. Порты d\_adress, d\_datain, d\_dataout и d\_size 32-битные.

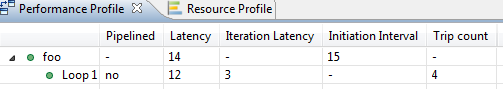


Рисунок 2.7. Performance Profile

На рисунке 2.7. видно, что задержка получения первого выходного значения составляет 3 такта с момента старта, для остальных -14, а задержка после старта до готовности приема новых данных – 15:

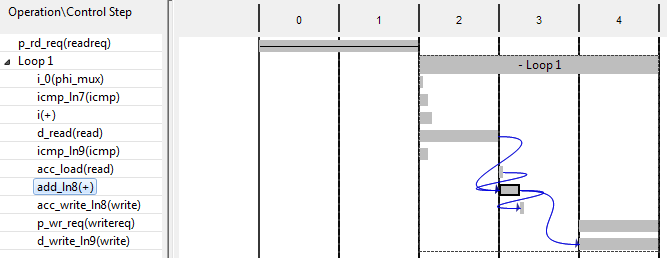


Рисунок 2.8. Schedule viewer

В отличие от предыдущего решения у нас присутствует цикл. За счет этого величина latency увеличилась до 14.

Рассмотрим профиль ресурсов:

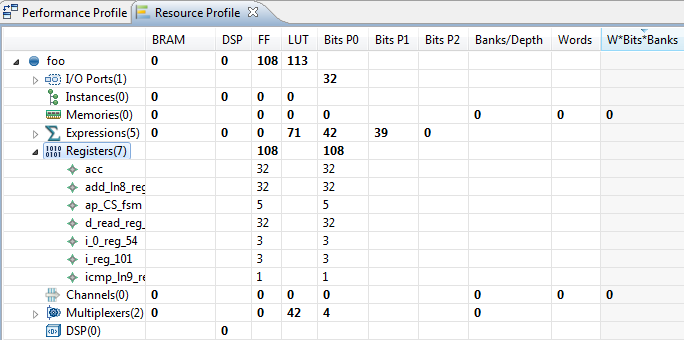


Рисунок 2.9. Resource Profile

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

## 2.4. C|RTL моделирование

При осуществлении совместного моделирования программа показала ожидаемые результаты Latency

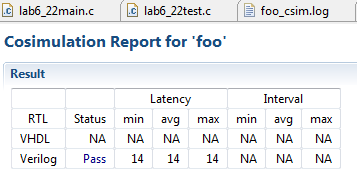


Рисунок 2.10. Отчет о моделировании

Покажем временную диаграмму моделирования на рисунке 2.11.



Рисунок 2.11. Design Top Signals

Здесь также видны отличия во времени выполнения итераций, задержка составила 14 в отличие от предыдущего решения, в котором она была равна 4.

# Третье решение

## 3.1. Настройки третьего решения

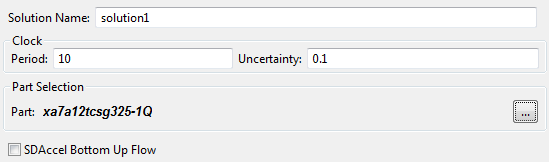


Рисунок 3.0. Параметры второго решения

Программа, тест к ней и интерфейс (директива) представлены на рисунках 3.1 и 3.2.

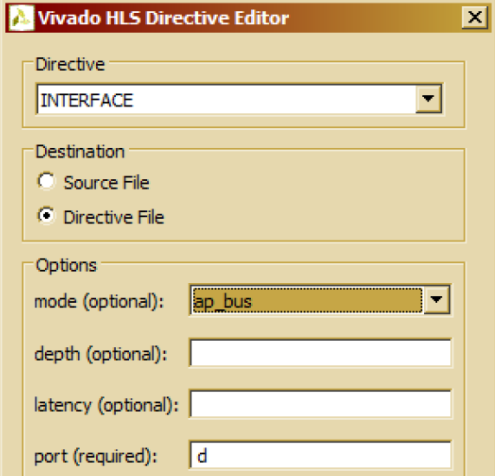
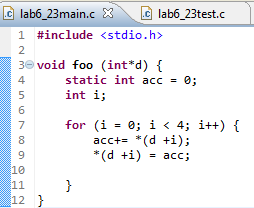


Рисунок 3.1Исходная программа.

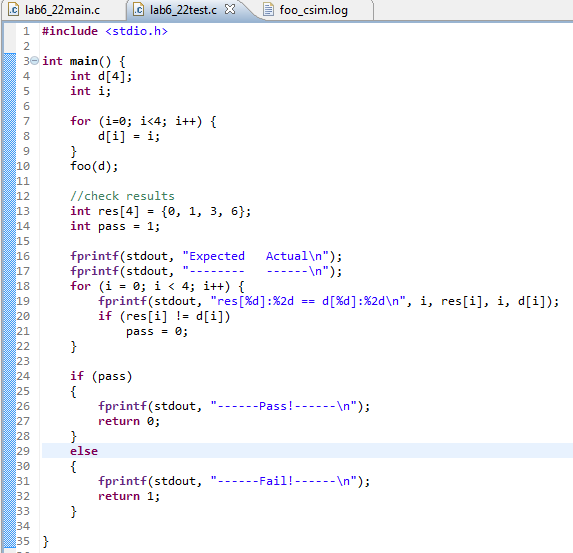


Рисунок 3.2. Тест к программе.

## 3.2. Моделирование

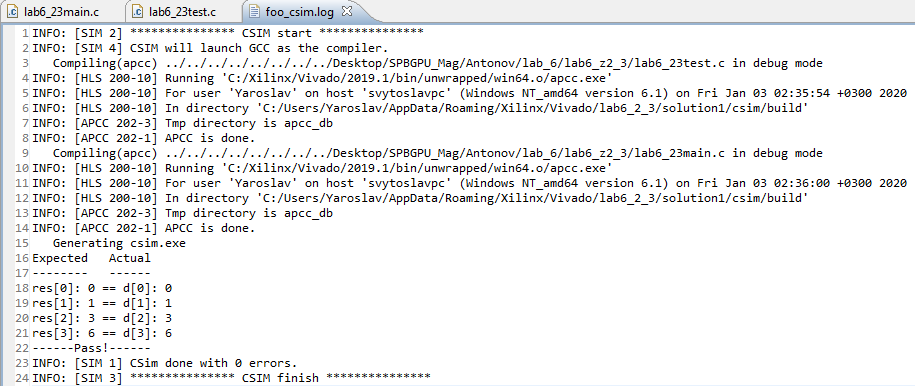


Рисунок 3.3 Результат успешного моделирования

Моделирование второго решения также прошло успешно.

## 3.3. Синтез

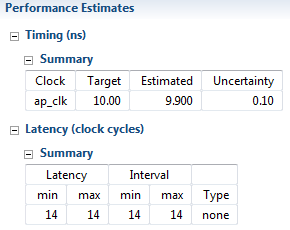


Рисунок 3.4. Performance estimates – summary

На рисунке 3.4. можно увидеть, что достигнутая задержка равна 9.900 + 0.1, укладывается в заданные требования тактовой частоты и совпадает со вторым решением . Величина Latency равная 14, также совпадает со вторым решением .

Использование ресурсов:

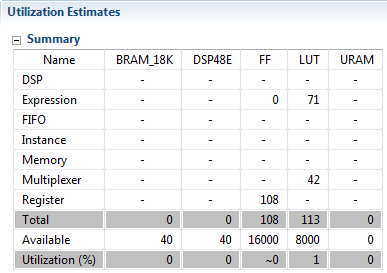


Рисунок 3.5. Utilization estimates – summary

На рисунке 3.5. видно, что данный проект теперь займет на микросхеме 108 регистров для хранения чисел, и 113 LUT. Затрачиваемые ресурсы также совпадают со вторым решением.

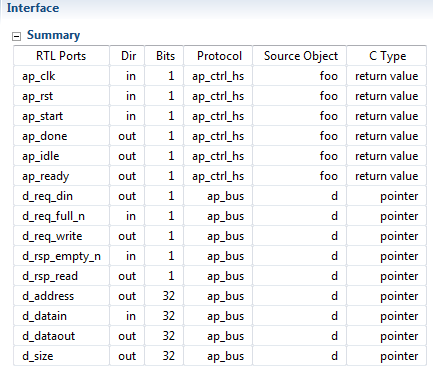


Рисунок 3.6. Interface Summary.

Полученные значения Interface Summary на рисунке 3.6. совпадают с прошлыми решениями, представленным на рисунке 1.7. и 2.6. На рисунке 3.6 также представлены интерфейсы, которые используются в синтезированном устройстве. Видно, что в схеме применяется те же протоколы ap\_bus. Порты d\_adress, d\_datain, d\_dataout и d\_size 32-битные.

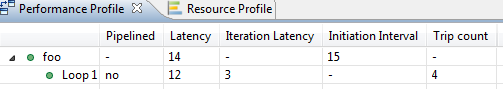


Рисунок 3.7. Performance Profile

Performance Profile совпадает на рисунке 3.7 совпадает с предыдущим решением. Из рисунка видно, что задержка получения первого выходного значения составляет 3 такта с момента старта, для остальных -14, а задержка после старта до готовности приема новых данных – 15.

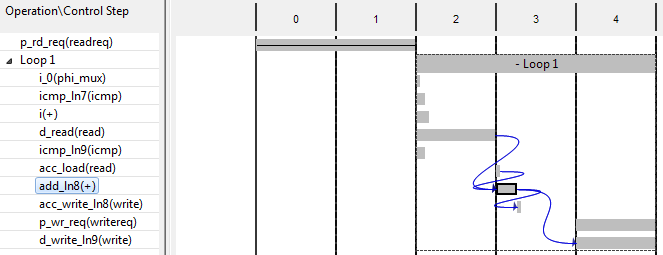


Рисунок 3.8. Schedule viewer

Schedule viewer также совпадает с предыдущим решением.

Величина latency также составила 14.

Профиль ресурсов также совпадает со вторым решением.

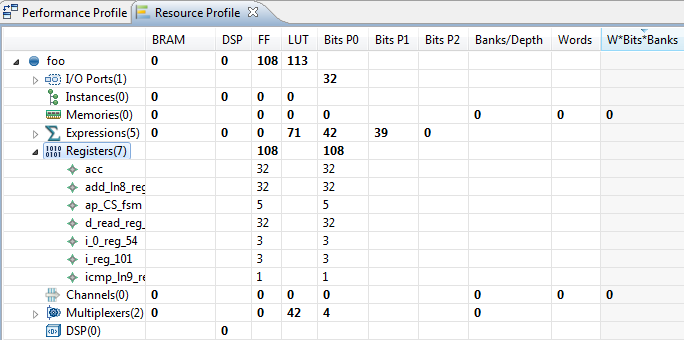


Рисунок 3.9. Resource Profile

Здесь мы также не видим отличий от второго решения.

## 3.4. C|RTL моделирование

При осуществлении совместного моделирования программа показала ожидаемые результаты Latency, которые также соответствуют второму решению.

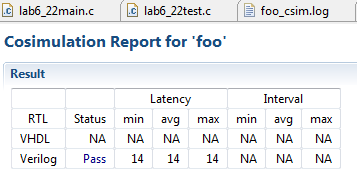


Рисунок 3.10. Отчет о моделировании

Покажем временную диаграмму моделирования на рисунке 3.11.



Рисунок 3.11. Design Top Signals

Здесь также не видны отличия во времени выполнения итераций от второго решения. Задержка составила также 14, отличие есть только с первым решением, в котором она была равна 4.

# Четвертое решение

## 4.1. Настройки четвертого решения

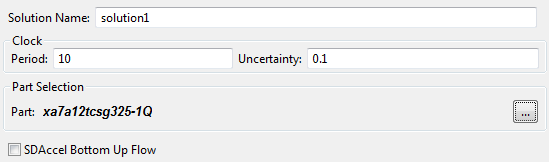


Рисунок 4.0. Параметры второго решения

Программа, тест к ней и интерфейс (директива) представлены на рисунках 4.1 и 4.2.

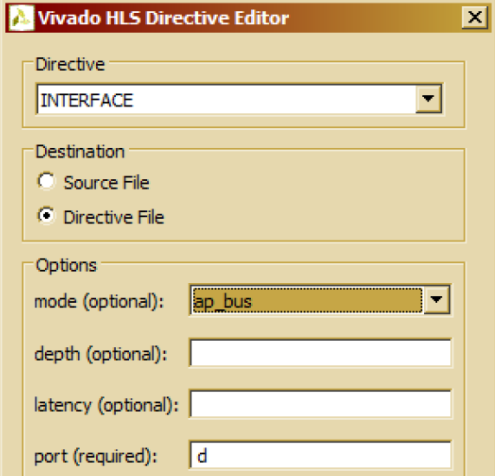
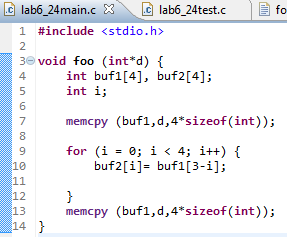


Рисунок 4.1. Исходная программа.

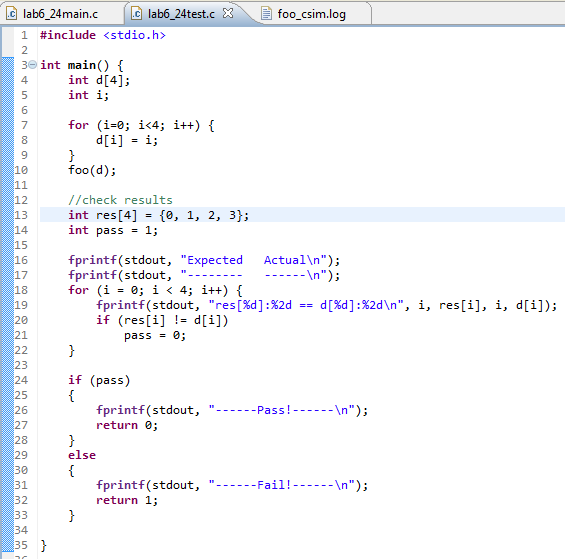


Рисунок 4.2. Тест к программе.

## 4.2. Моделирование

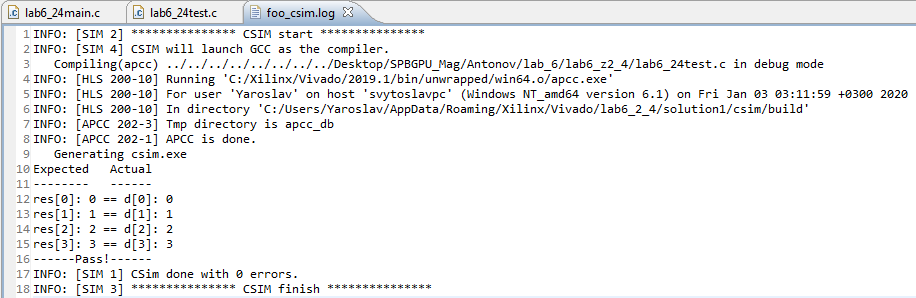


Рисунок 4.3 Результат успешного моделирования

Моделирование четвертого решения – операции копирования также прошло успешно.

## 4.3. Синтез

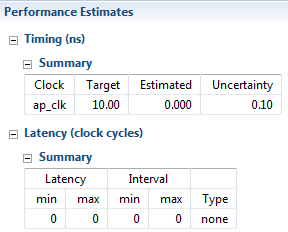


Рисунок 4.4. Performance estimates – summary

На рисунке 4.4. можно увидеть, что достигнутая задержка 0.000 + 0.1, и не совпадает ни с одним решением. Величина Latency составила 0, что также не совпадает ни с одним решением. Операция копирования прошла без задержек.

Использование ресурсов:

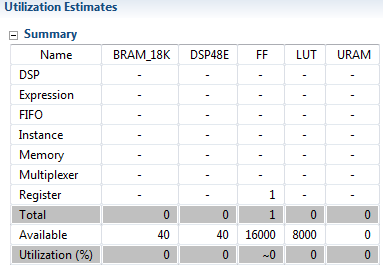


Рисунок 4.5. Utilization estimates – summary

На рисунке 4.5. видно, что данный проект теперь займет на микросхеме только 1 регистр для хранения чисел. Затрачиваемые ресурсы не совпадают ни с одним решением, так осуществляется процедура копирования.

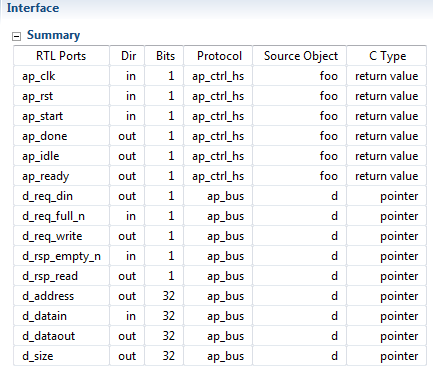


Рисунок 4.6. Interface Summary.

Полученные значения Interface Summary на рисунке 4.6. совпадают с прошлыми решениями, представленным на рисунке 1.7, 2.6 и 3.6. На рисунке 4.6 также представлены интерфейсы, которые используются в синтезированном устройстве. Видно, что в схеме применяется те же протоколы ap\_bus. Порты d\_adress, d\_datain, d\_dataout и d\_size 32-битные.

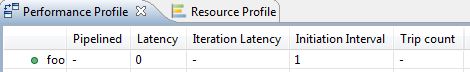


Рисунок 4.7. Performance Profile

Performance Profile на рисунке 4.7 говорит о том, что задержка при копировании отсутствует, а арифметических операций не происходит.

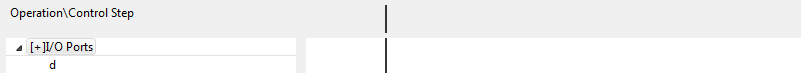


Рисунок 4.8. Schedule viewer

Schedule viewer также говорит о том, что процедура копирования происходит без задержек.

Профиль ресурсов также имеет ожидаемую структуру.

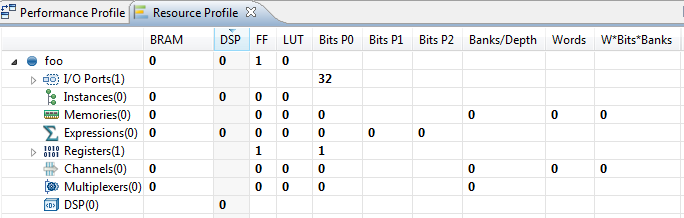


Рисунок 4.9. Resource Profile

## 4.4. C|RTL моделирование

При осуществлении совместного моделирования программа показала ожидаемые результаты Latency, которые также соответствуют второму решению.

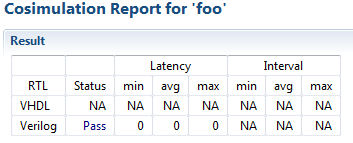


Рисунок 4.10. Отчет о моделировании

Покажем временную диаграмму моделирования на рисунке 4.11.

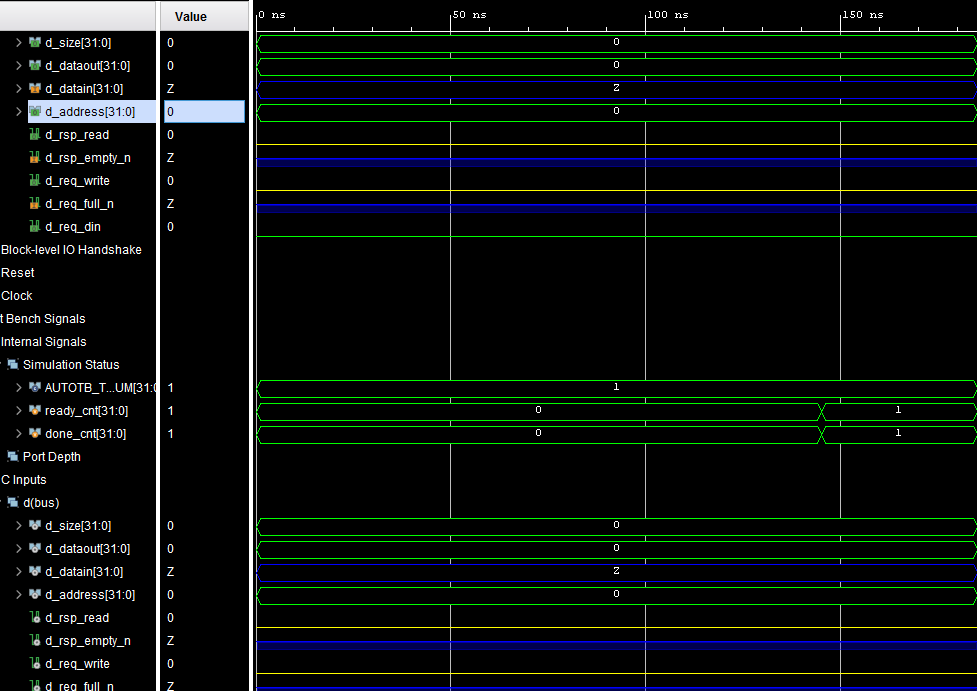


Рисунок 4.11. Design Top Signals

Здесь также видно, что операция копирования идет без задержек и иных арифметических операций не происходит.

# 4.Выводы

В данной работе были рассмотрены различия, которые могут появиться при синтезе устройства с применением протокола шины на уровне порта.

Инструмент Vivado HLS поддерживает протокол ввода-вывода шины:

- Протокол ввода-вывода - это протокол общей шины

- Протокол ввода / вывода не является отраслевым стандартом

- Протокол шины инструментов Vivado HLS позволяет подключаться к ядрам адаптера.

Протокол ввода / вывода шины поддерживает memcpy

- Протокол ввода / вывода шины поддерживает функцию C memcpy

- Обеспечивает высокопроизводительный интерфейс для пакетной передачи данных в стиле DMA

Протокол ввода / вывода шины поддерживает сложную арифметику указателей на вводе / выводе

- Указатели на ввод / вывод могут быть синтезированы в ap\_fifo или ap\_bus

- При использовании ap\_fifo доступ должен быть последовательным

- Если используется арифметика указателя, порт должен использовать ap\_bus

Протокол типа ap\_bus реализует переменные указателя и передачи по ссылке в виде шины общего назначения.

В результате получены 4 решения: первое – одиночная операция чтения и записи массива в стандартном режиме. Ведичина latancy составила 4 такта, а максимальная задержка обработки сигнала на такте составляет 9.900 + 0.1нс; второе – множественная операция чтения и записи массива с аккумулирующим значением без дополнительных арифметических операций в стандартном режиме, полный цикл выполнения за 14 тактов, задержка составила также 9.900 + 0.1нс; третье решение также - множественная операция чтения и записи массива с аккумулирующим значением с дополнительными арифметическими операциями в стандартном режиме, полный цикл выполнения также за 14 тактов, а максимальная задержка обработки сигнала на такте составила также 9.900 + 0.1нс; четвертое решение - режим копирования, данный режим прошел без задержек и занял минимальное количество ресурсов.