Санкт-Петербургский Политехнический Университет Петра Великого  
Институт Компьютерных наук и технологий  
Кафедра компьютерных систем и программных технологий

Лабораторная работа 3

Предмет: Проектирование реконфигурируемых гибридных вычислительных систем

Тема: Port-Level IO Protocols

Задание 1

Студент: Ерниязов Т.Е.

Гр. № 3540901/81501

Преподаватель: Антонов А.П.

Санкт-Петербург

2019

Оглавление

[Задание 4](#_Toc21265047)

[Ход работы 6](#_Toc21265048)

[Решение 1 6](#_Toc21265049)

[Решение 2 9](#_Toc21265050)

[Решение 3 13](#_Toc21265051)

[Выводы 17](#_Toc21265052)

# Задание

* 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
    - 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
    - Открыть временную диаграмму (все сигналы)
      * Отобразить два цикла обработки на одном экране
        + На скриншоте показать Latency
        + На скриншоте показать Initiation Interval
* Сделать solution2
  + Задать протокол (block-level): ap\_cntl\_chain
  + осуществить моделирование
  + 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
    - Открыть временную диаграмму (все сигналы)
      * Отобразить два цикла обработки на одном экране
        + На скриншоте показать Latency
        + На скриншоте показать Initiation Interval
* Сделать solution3
  + Задать протокол (block-level): ap\_cntl\_none
  + задать: clock period 10; clock\_uncertainty 0.1
  + осуществить моделирование
  + 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
    - Проверить происходит или нет моделирование, объяснить почему.
    - Если моделирование происходит, то открыть временную диаграмму (все сигналы)
      * Отобразить два цикла обработки на одном экране
        + На скриншоте показать Latency
        + На скриншоте показать Initiation Interval
* Выводы
  + Объяснить отличие протоколов block\_level

# Ход работы

## Решение 1

1. Исходный код взят из лабораторной 2

Код

**int** **lab1\_1**( **char** a, **char** b, **char** c, **char** d) {

**int** y;

y = a\*b+c+d;

**return** y;

}

Тест

**#include** <stdio.h>

**int** **main**()

{

**int** inA, inB, inC, inD;

**int** res;

// For adders

**int** refOut[3] = {270, 490, 1310};

**int** pass;

**int** i;

inA = 10;

inB = 20;

inC = 30;

inD = 40;

// Call the adder for 5 transactions

**for** (i=0; i<3; i++)

{

res = lab1\_1(inA, inB, inC, inD);

**fprintf**(stdout, " %d\*%d+%d+%d=%d \n", inA, inB, inC, inD, res);

// Test the output against expected results

**if** (res == refOut[i])

pass = 1;

**else**

pass = 0;

inA=inA+10;

inB=inB+10;

inC=inC+10;

inD=inD+10;

}

**if** (pass)

{

**fprintf**(stdout, "----------Pass!------------\n");

**return** 0;

}

**else**

{

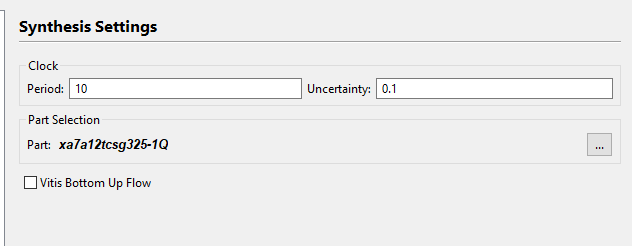
**fprintf**(stderr, "----------Fail!------------\n");

**return** 1;

}

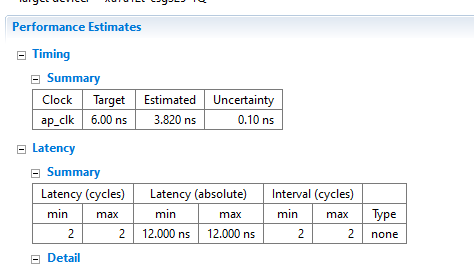
}

1. Создание проекта с заданными настройками

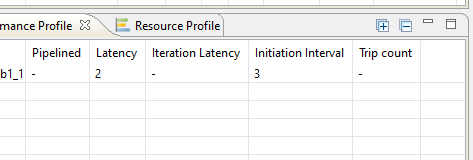


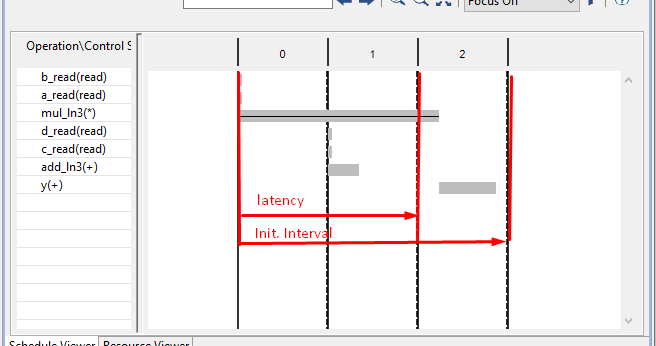
1. Результат синтеза:

Производительность



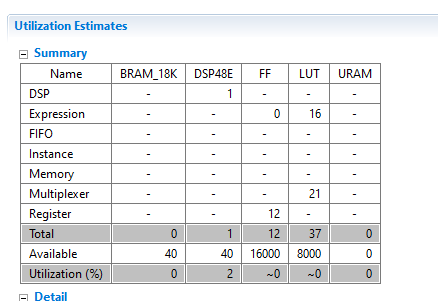
Величина полученной задержки равна 3,820, что укладывается в заданные значения.





Значение результата вычислений можно получить через два такта. Готовность инициализации новых значений для вычислений наступает еще через 1 такт.

Затрачиваемые ресурсы:



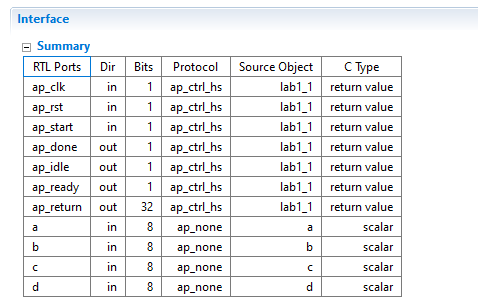
Данный проект будет занимать на микросхеме:

1 DSP блок, где будут задействованы сумматор и умножитель.

12 регистров для хранения и считывания данных (чисел).

37 LUT.

Интерфейс

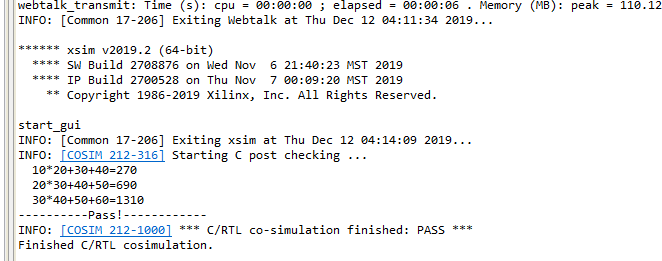


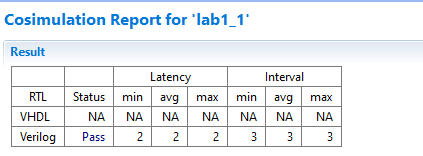
Для расчета схемы требуется более одного такта, поэтому в схему были добавлены ap\_clk и ap\_rst. Оба являются однобитовыми входами. Протокол управления вводом / выводом на уровне блоков был добавлен для управления RTL. Порты: ap\_start, ap\_done, ap\_idle и ap\_ready. Конструкция имеет 5 портов данных.

Входные порты: a, b, c, d являются 8-битными входами и имеют входы / выходы, протокол ap\_none.

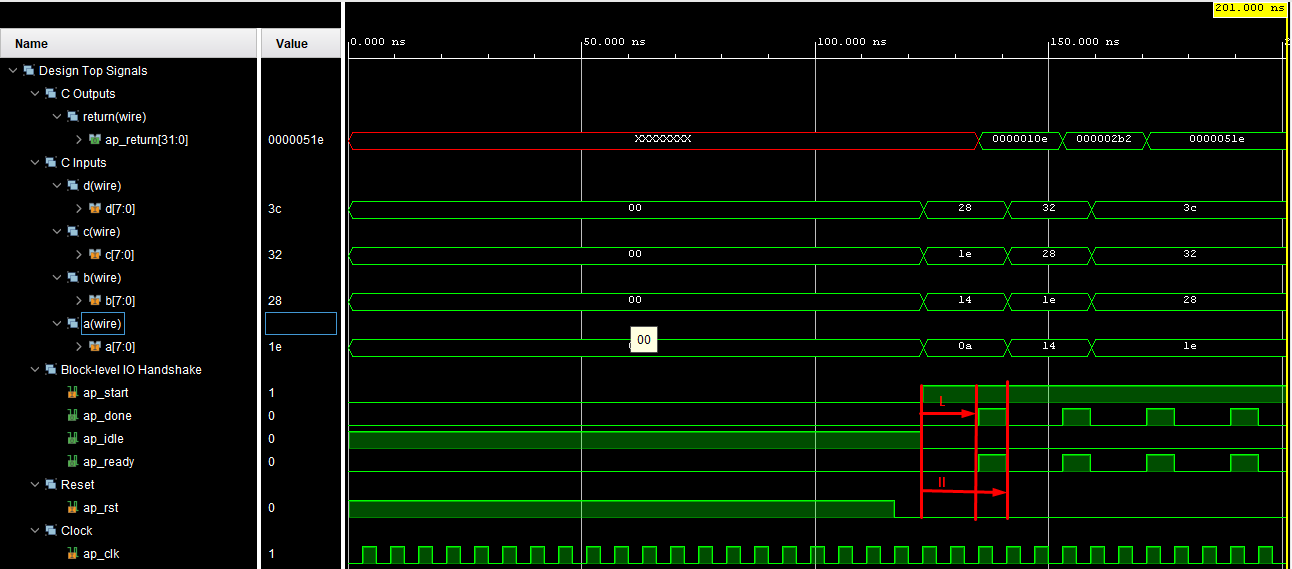
Конструкция имеет 32-битный выходной порт для возврата функции ap\_return.

1. C/RTL моделирование.





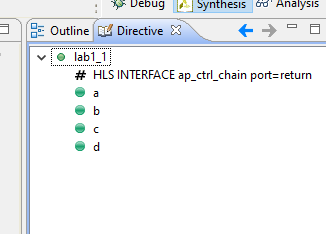
Значения соответствуют полученным ранее.



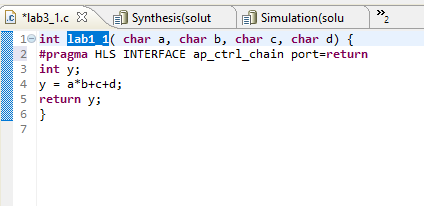
## Решение 2

1. Добавление протокола

Directive

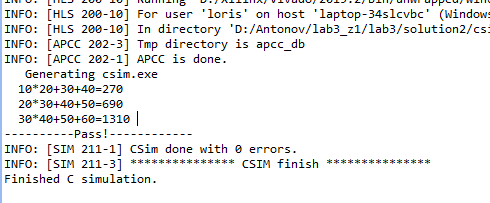


Lab3\_1.c



ap\_ctrl\_chain: протокол ввода-вывода на уровне блоков для цепочки управления. Этот протокол ввода / вывода в основном используется для объединения конвейерных блоков.

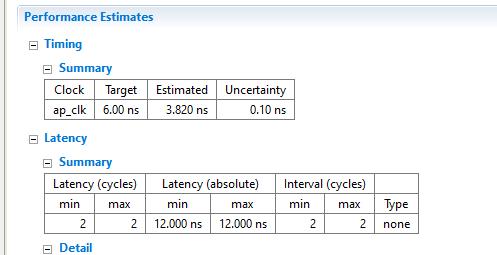
1. Моделирование:

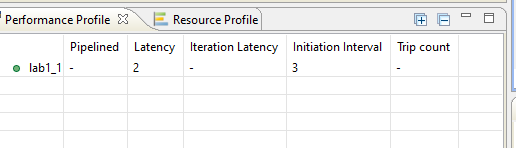


Моделирование выполнено успешно.

1. Синтез:

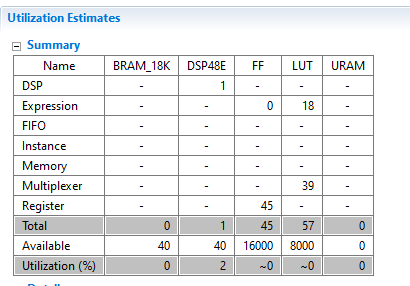
Производительность





Значения соответствуют тем, которые были определены в решении 1.

Использование ресурсов



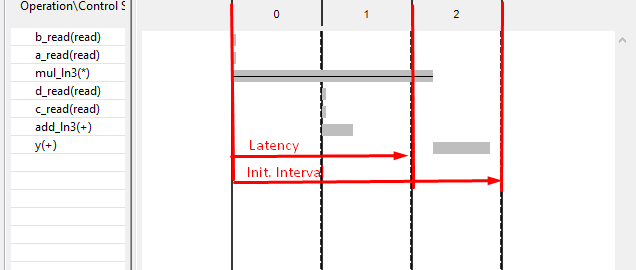
Данный проект будет занимать на микросхеме:

1 DSP блок, где будут задействованы сумматор и умножитель.

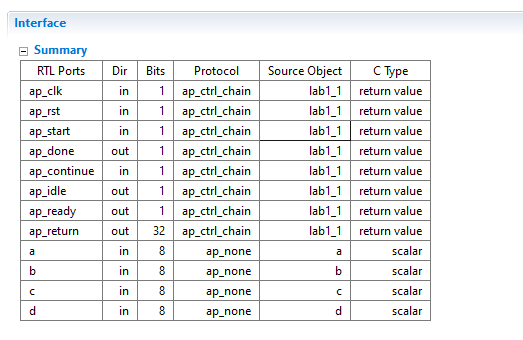
45 регистров для хранения и считывания данных (чисел).

63 LUT.

По сравнению с предыдущим решением выросло количество используемых регистров и LUT.



Интерфейс

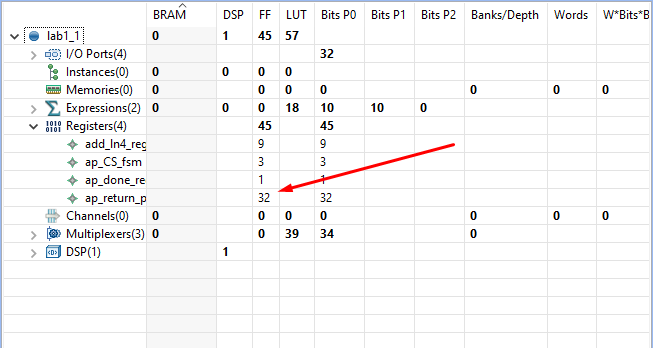


По сравнению с решением 1 появился протокол ap\_ctrl\_chain. Порты: ap\_start, ap\_done, ap\_idle, ap\_ready, ap\_clk, ap\_rst, ap\_continue (активен, когда ap\_done завершается для следующей транзакции; дает возможность останавливать дальнейшую обработку при отсутствии возможности обработки новых данных). Конструкция имеет 5 портов данных.

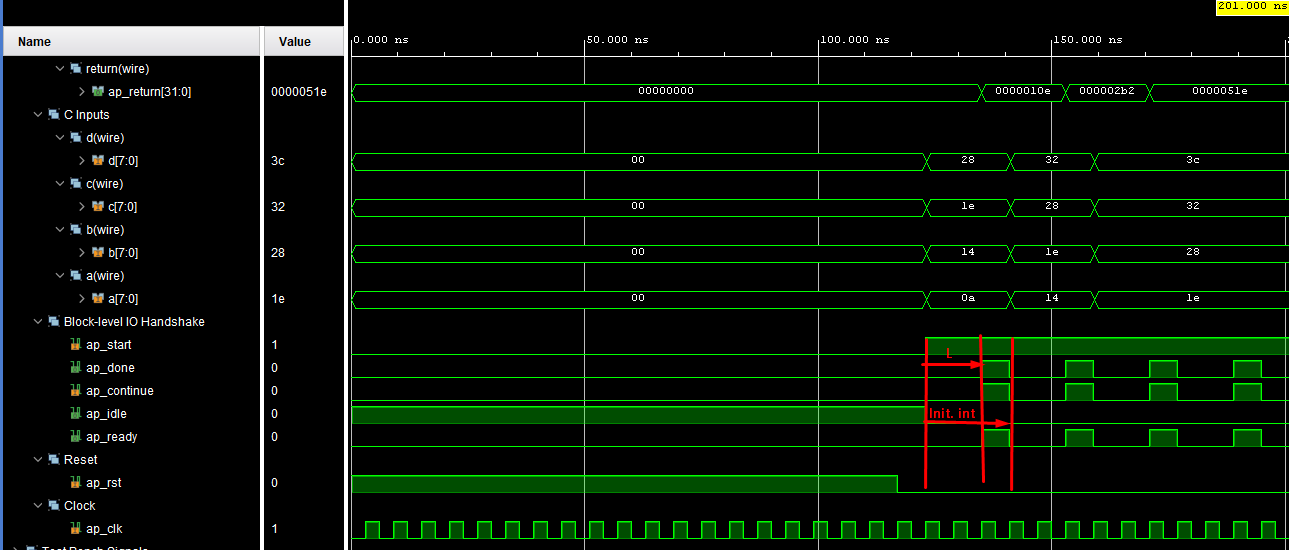
Входные порты: a, b, c, d являются 8-битными входами и имеют входы / выходы, протокол ap\_none.

Конструкция имеет 32-битный выходной порт для возврата функции ap\_return.

Resource profile



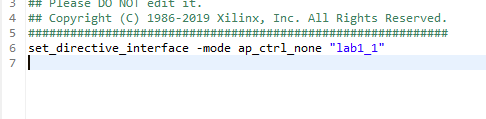
1. С/RTL моделирование

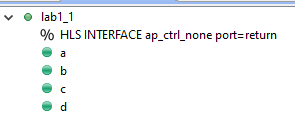


## Решение 3

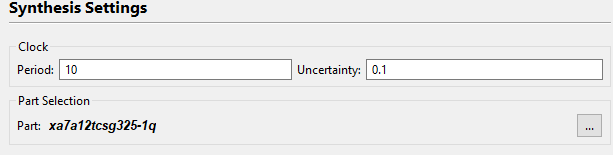
1. Создание и настройка решения.

Установка протокола

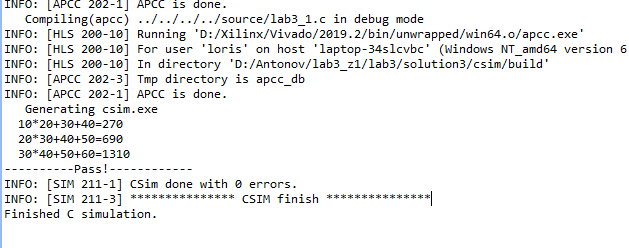




Конфигурирование решения

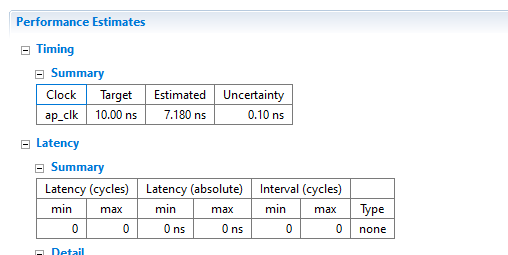


1. Моделирование

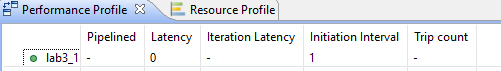


1. Синтез

Производительность

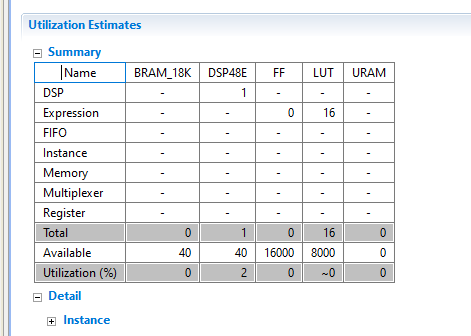


По сравнению с предыдущими решениями значение задержки изменилось из-за того, что были заданы другие конфигурации решения. Величина полученной задержки соответствует заданному значению.



По сравнению с предыдущим решением, получение результата происходит сразу. Через 1 такт наступает готовность получения новых данных.

Использование ресурсов

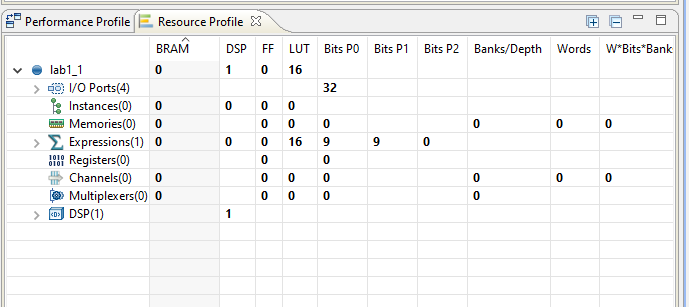


Данный проект будет занимать на микросхеме:

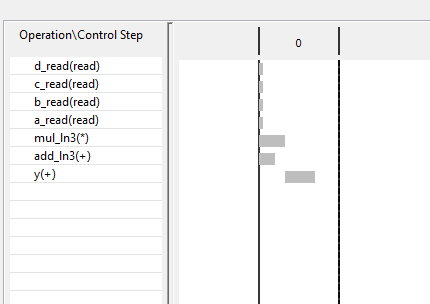
1 DSP блок, где будут задействованы сумматор и умножитель.

16 LUT.

По сравнению с предыдущими решениями использование регистров полностью отсутствует.

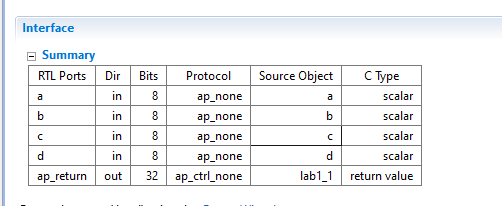


Данные соответствуют приведенному выше отчету.



На данном изображении видно, что задержка получения результата отсутствует, а интервал инициализации составляет 1 такт.

Интерфейс



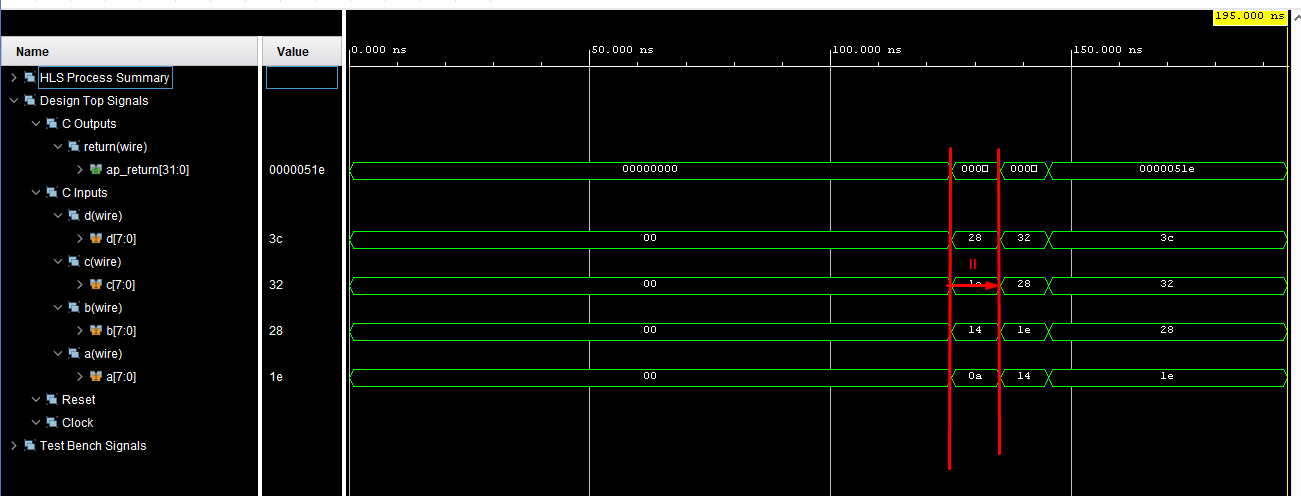
По сравнению с предыдущими решениями в данном решении отсутствуют многие RTL порты. Конструкция имеет 5 портов данных.

Входные порты: a, b, c, d являются 8-битными входами и имеют входы / выходы, протокол ap\_none.

Конструкция имеет 32-битный выходной порт для возврата функции ap\_return.

Заданный протокол ap\_ctrl\_none: No block-level I/O protocol. Когда используется протокол интерфейса ap\_ctrl\_none, никакие протоколы ввода-вывода уровня блока не используются

1. С/RTL моделирование.



# Выводы

Существуют следующие типы протоколов: ap\_ctrl\_none, ap\_ctrl\_hs, и ap\_ctrl\_chain. Они могут быть заданы только для возвращаемого значения функции. ap\_ctrl\_hs задается как протокол по умолчанию. Протокол ap\_ctrl\_chain похож на ap\_ctrl\_hs, но имеет дополнительный входной порт ap\_continue. Если порт ap\_continue является логическим 0, когда функция завершается, блок остановит операцию и следующая транзакция не будет продолжена. Следующая транзакция будет выполняться только тогда, когда ap\_continues иммет значение 1. Режим ap\_ctrl\_none реализует моделирование без какого-либо блочного протокола ввода-вывода.