Санкт-Петербургский Политехнический Университет Петра Великого

Институт компьютерных наук и технологии

Высшая школа интеллектуальных систем и суперкомпьютерных технологий

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ Lab1\_z2

Дисциплина: Проектирование реконфигурируемых гибридных вычислительных систем

Выполнил студент Курякин Д.

Гр. 3540901/12001

Руководитель, доцент Антонов А.П.

«\_\_» \_\_\_\_\_\_ 2022

Санкт-Петербург

2022

Оглавление

[1. Задание 3](#_Toc116916474)

[2. Исходный код функции 3](#_Toc116916475)

[3. Исходный код теста 3](#_Toc116916476)

[3.1 Моделирование 4](#_Toc116916477)

[4. Первое решение — Solution1 5](#_Toc116916478)

[4.1 Исходные настройки 5](#_Toc116916479)

[4.2 Синтез 5](#_Toc116916480)

[5. Первое решение — Solution2 7](#_Toc116916481)

[5.1 Исходные настройки 7](#_Toc116916482)

[5.2 Синтез 8](#_Toc116916483)

[6. Сравнение Solution1 и Solution2 9](#_Toc116916484)

[7. Выводы 10](#_Toc116916485)

1. Задание

* Разработать на языке Си описание функции, реализующей следующий алгоритм outArr[i] = inArr[i] + inA + inB + inC, где i=0, 1, .. (ROWS-1). ROWS=3
* Тип данных для inArr[i] , inA, inB, inC, inD – short; для outArr[i] – int;
* Разработать тест, обеспечивающий автоматическую проверку получаемых результатов моделирования
* разработанной функции.
* Создать, провести исследование и сравнительный анализ двух аппаратных реализаций разработанного на
* языке Си описания функции.
  + Микросхема: xa7a12tcsg325-1q
  + clock period 6; clock\_uncertainty 1 (для решения Solution 1)
  + clock period 10; clock\_uncertainty 1 (для решения Solution 2)

**2. Исходный код функции**

Исходный код заголовочного файла приведен ниже.

|  |
| --- |
| **#define** **ROWS** 3  **typedef** **short** **din\_type**;  **typedef** **int** **dout\_type**;  **void** **lab1\_z2**(**din\_type** inArr[ROWS], **din\_type** a,  **din\_type** b, **din\_type** c, **dout\_type** outArr[ROWS]); |

Исходный код синтезируемой функции приведен ниже.

|  |
| --- |
| **#include**"lab1\_z2.h"  **void** **lab1\_z2**(**din\_type** inArr[ROWS], **din\_type** a,  **din\_type** b, **din\_type** c, **dout\_type** outArr[ROWS])  {  **din\_type** x;  **dout\_type** y;  **for**(**int** i = 0; i < ROWS; i++)  {  **#pragma** HLS PIPELINE off  x =inArr[i];  y = x + a + b + c;  outArr[i] = y;  }  } |

Функция принимает на вход массив, 3 аргумента — скаляра и выходной массив. Затем считает значение по формуле и возвращает результат расчета в принимаемый массив.

**3. Исходный код теста**

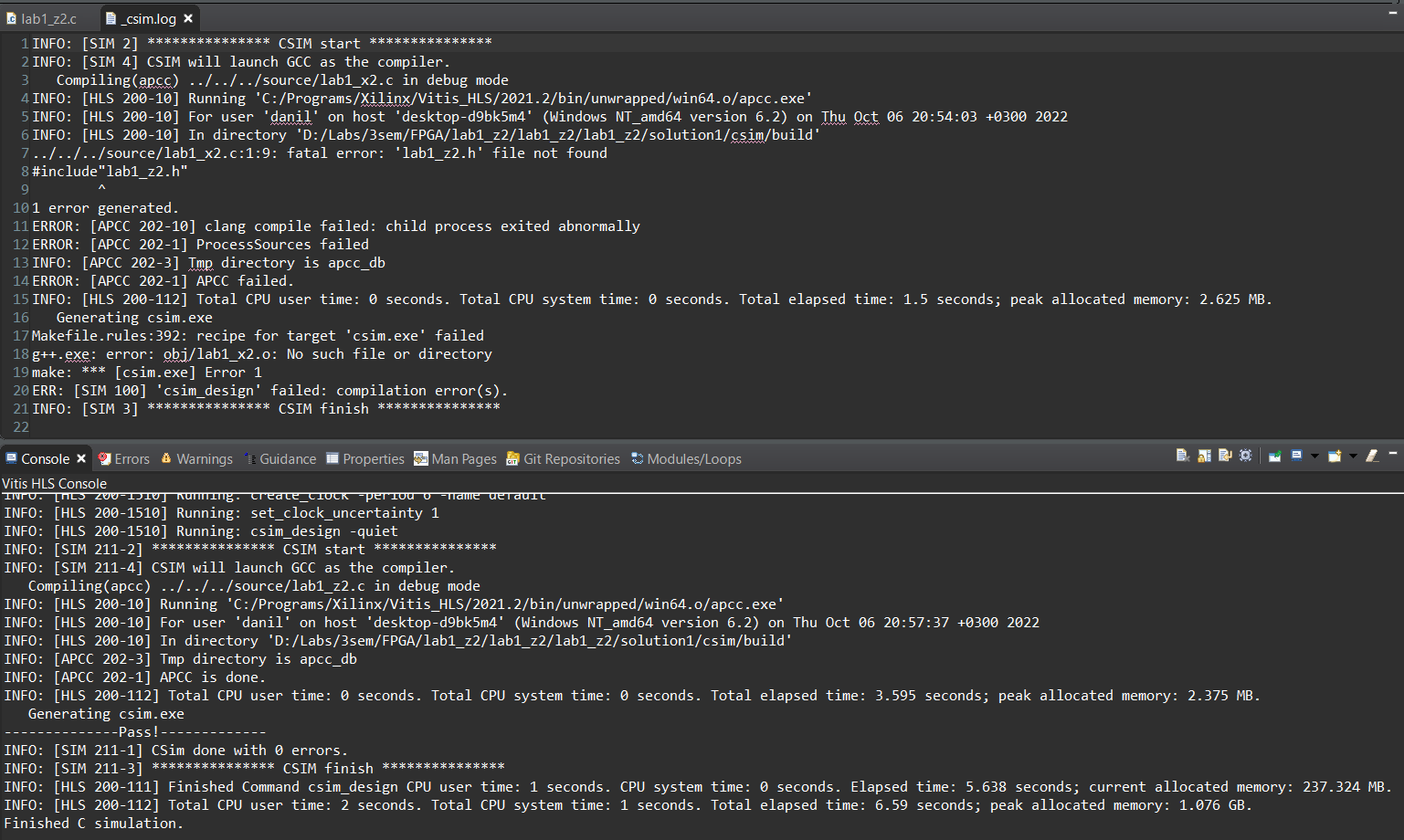
Исходный код теста для проверки функции lab1\_z2 приведен ниже.

|  |
| --- |
| **#include** <stdio.h>  **#include** "lab1\_z2.h"  **int** **arentEqual**(**dout\_type** expectedArr[ROWS],  **dout\_type** actualArr[ROWS])  {  **for**(**int** i = 0;i < ROWS; i++)  **if** (expectedArr[i] != actualArr[i])  {  **fprintf**(stdout, "ERROR: expected=%d actual=%d for ROW: %d \n",  expectedArr[i], actualArr[i], i);  **return** 1;  }  **return** 0;  }  **int** **main**()  {  **din\_type** inA, inB, inC;  **din\_type** inArr[ROWS];  **dout\_type** expectedArr[ROWS], actualArr[ROWS];  **int** pass = 0;  **for** (**int** i = 0; i < 3; i++)  {  inA = **rand**() % 100;  inB = **rand**() % 100;  inC = **rand**() % 100;  **for** (**int** j=0; j<ROWS; j++)  {  inArr [j] = **rand**() % 1000;  actualArr[j] = 0;  }  **lab1\_z2**(inArr, inA, inB, inC, actualArr);  **for** (**int** i=0; i<ROWS; i++)  expectedArr[i] = inArr[i] + inA + inB + inC;  **if** (**arentEqual**(expectedArr, actualArr))  pass = 1;  }  **if** (!pass)  **fprintf**(stdout,"--------------Pass!-------------\n");  **else**  **fprintf**(stderr, "-------------Fail!-------------\n");  **return** pass; |

Тест обеспечивает проверку корректной работы функции.

**3.1 Моделирование**

Результаты моделирования исходного кода синтезируемой функции приведены на рисунке 4.



*Рис. 4 Лог результата выполнения моделирования.*

Результаты моделирования показывают, что тест успешно пройден — так как условия теста выполняются (в консоли выведено сообщение -Pass!-).

**4. Первое решение — Solution1**

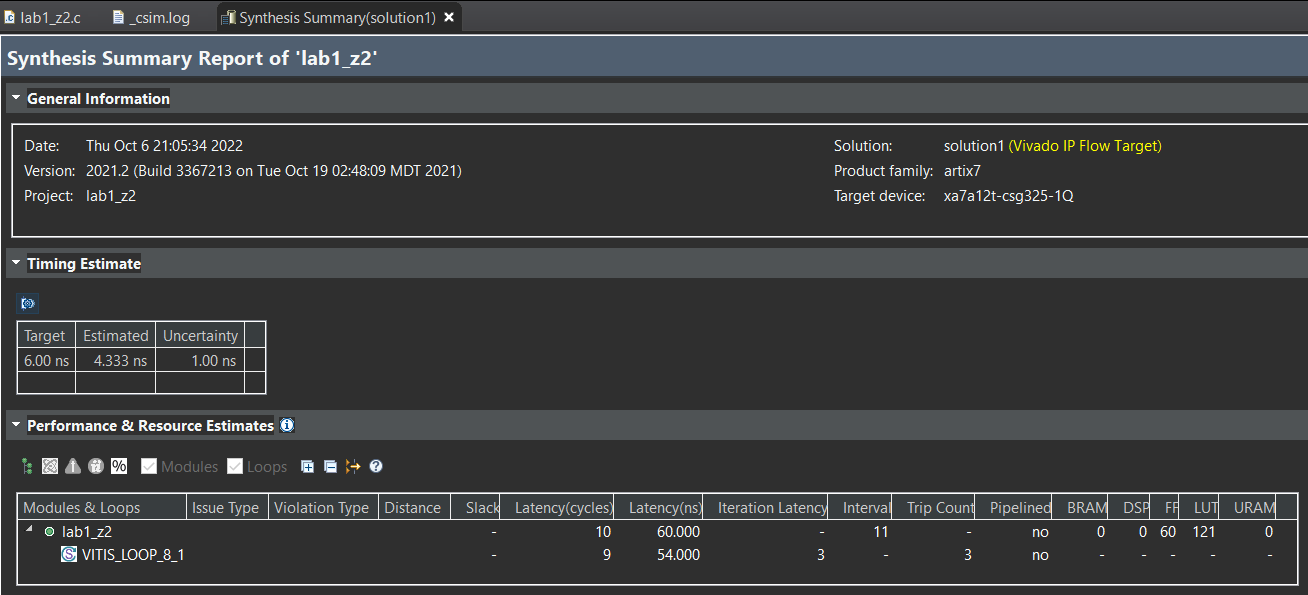
**4.1 Исходные настройки**

При создании первого решения задаются следующие настройки:

* clock period: 6,
* clock uncertain: 1,
* part: xa7a12tcsg325-1Q

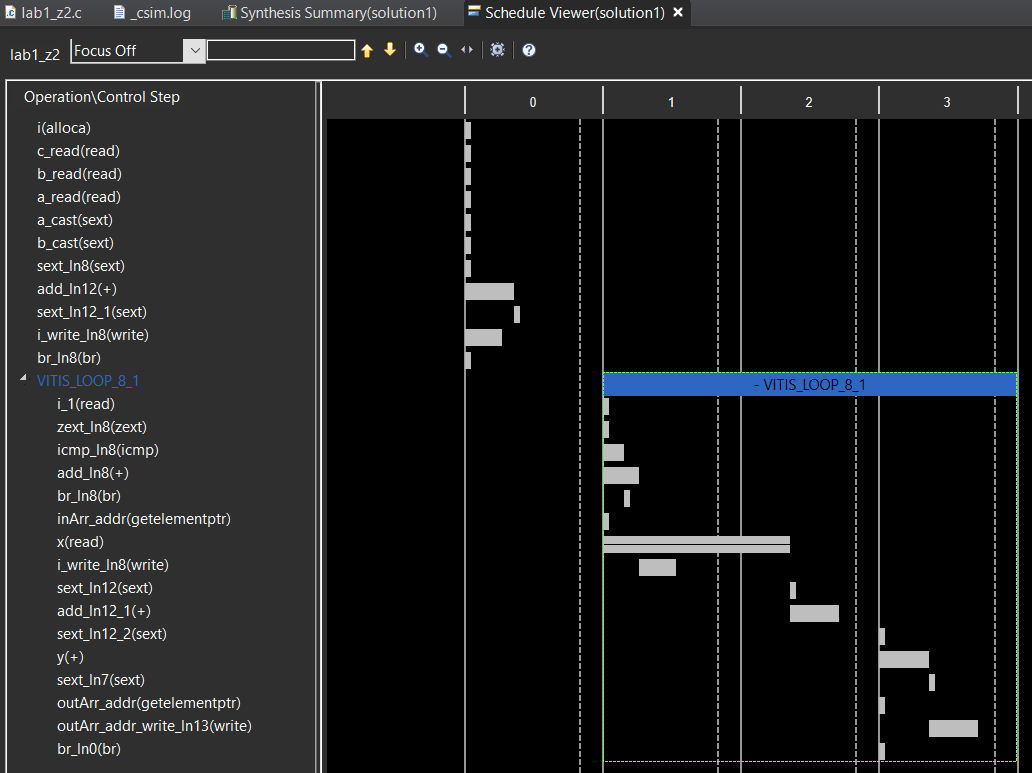
**4.2 Синтез**

Результаты представлены на рисунке 4.



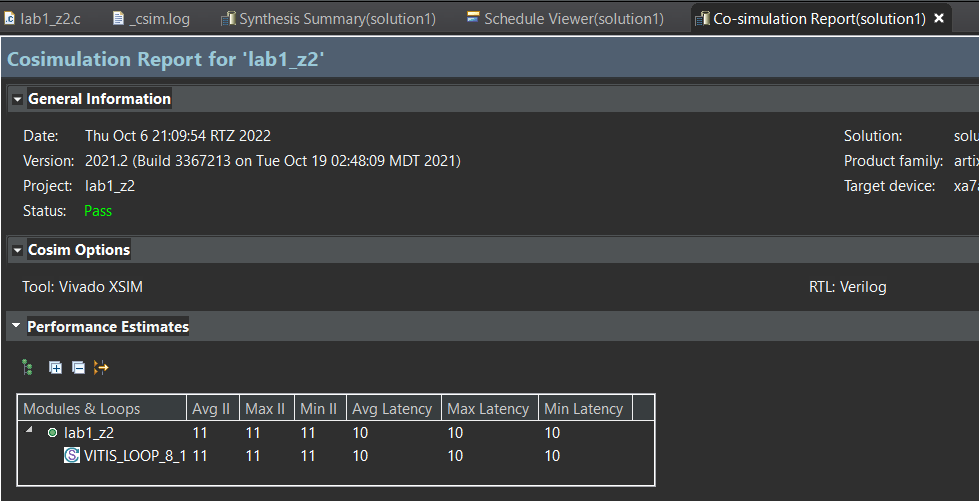
*Рис. 5 Оценка быстродействия для solution1*

После этого было открыто окно планировщика (Schedule). Планировщик представлен на рис 6.



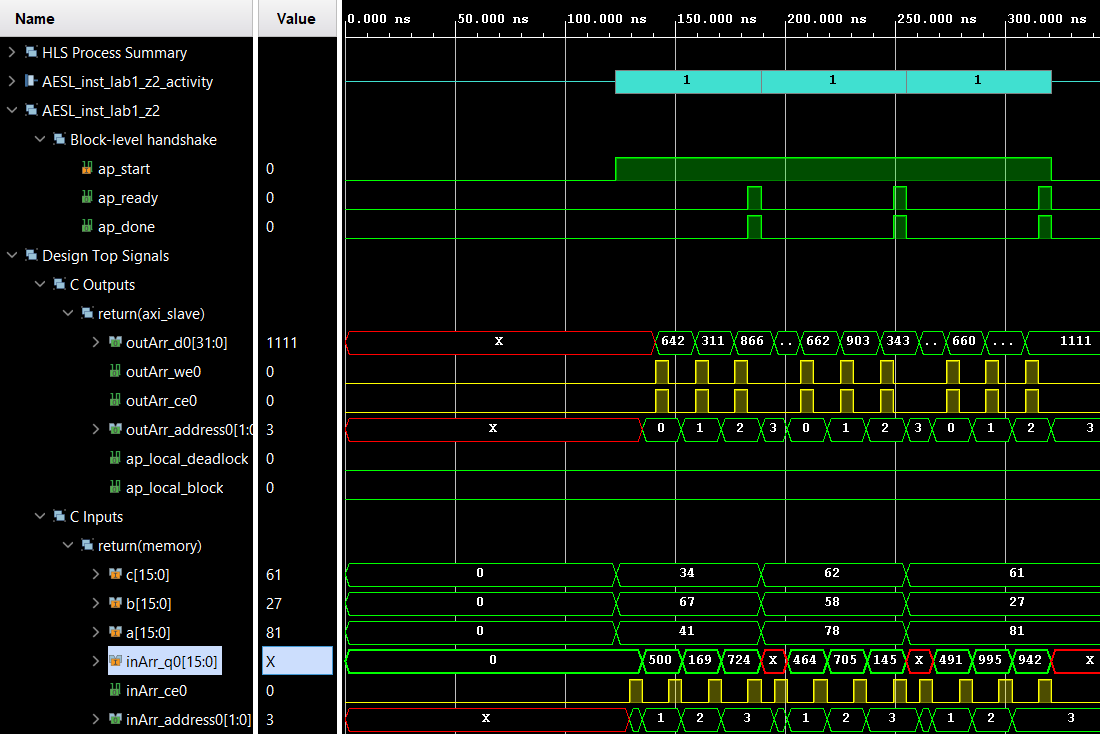
*Рис. 6 Планировщик Schedule для solution1*

После этого была выполнена процедура Co-simulation.



*Рис. 9 Результаты Co-simulation для solution1*

После этого была получена временная диаграмма, представленная на рисунке ниже:



*Рис. 10 Временная диаграмма для solution1*

**5. Первое решение — Solution2**

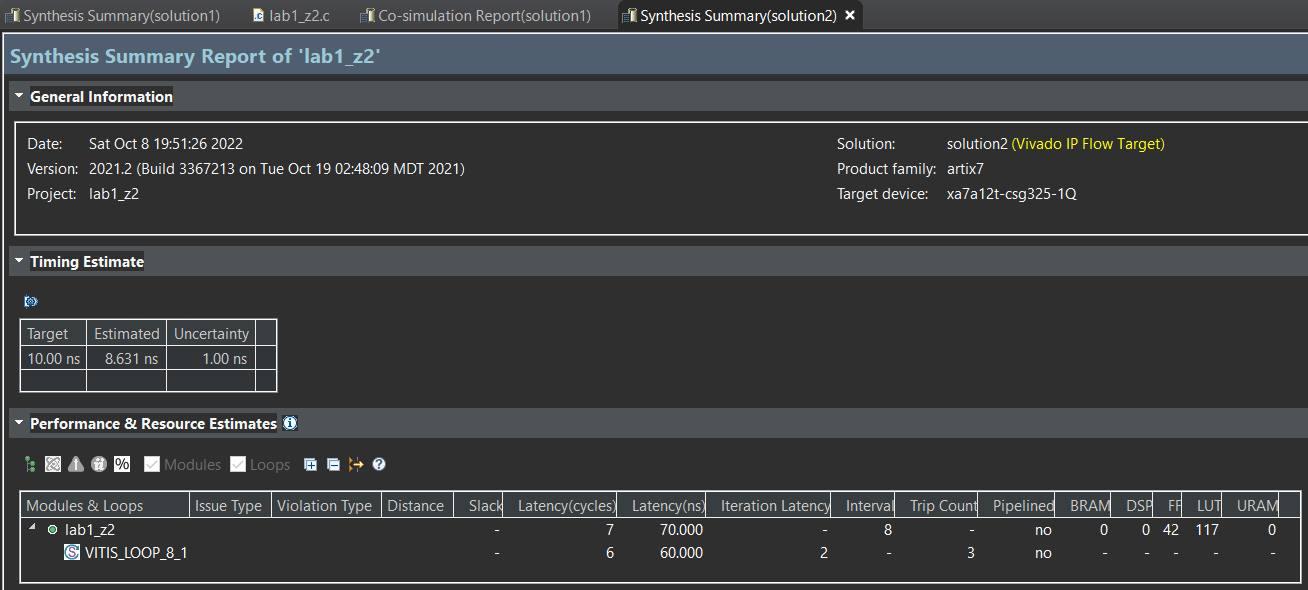
**5.1 Исходные настройки**

При создании второго решения задаются следующие настройки:

* clock period: 10,
* clock uncertain: 1,
* part: xa7a12tcsg325-1Q

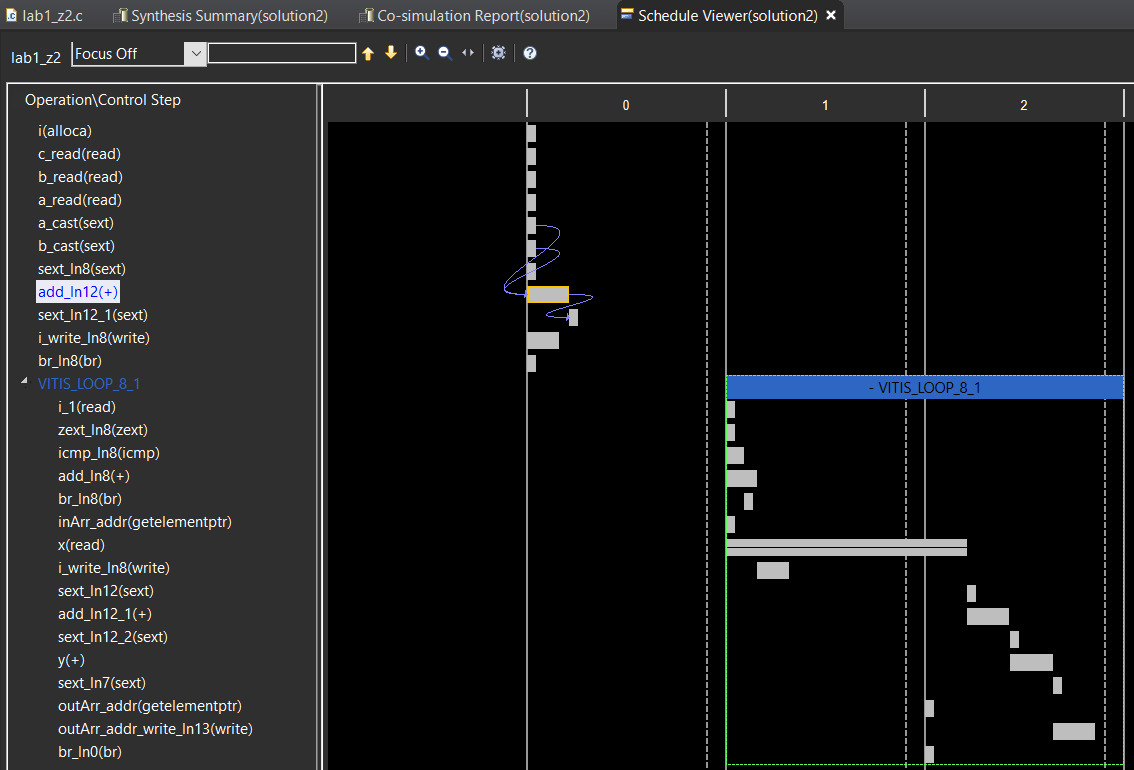
**5.2 Синтез**

Было создано новое решение и проделаны все те же самые шаги что и с Solution1. Сначала был запущен синтез. В настройках был изменён Clock Period. Результаты синтеза приведены ниже.



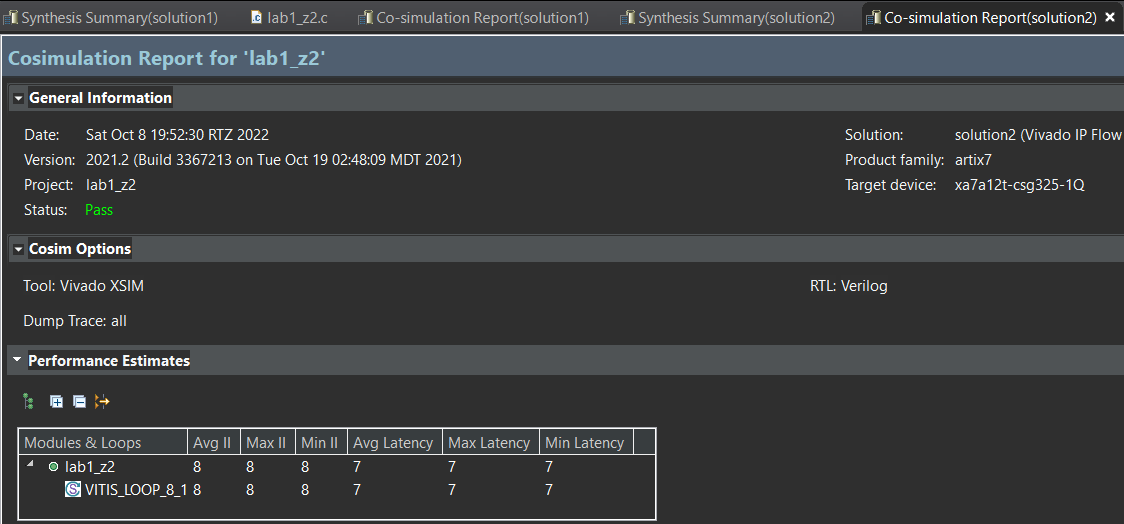
*Рис. 11 Оценка быстродействия для solution2*

Окно планировщика выглядит представлено ниже.



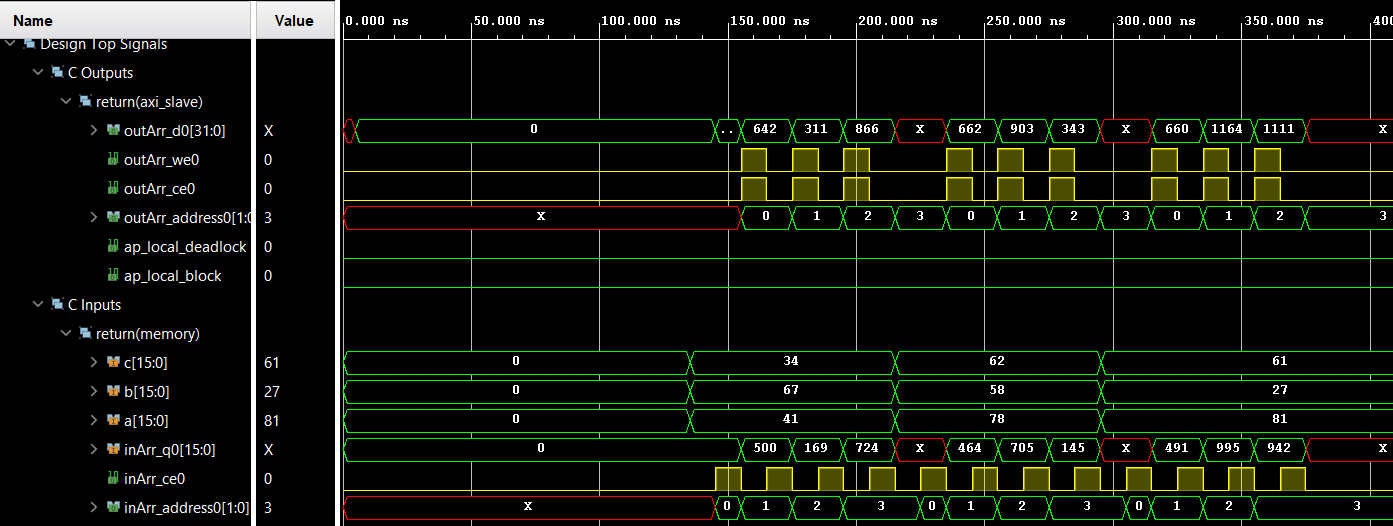
*Рис. 12 Оценка быстродействия для solution2*

Далее выполнена Co-simulation. Результаты показаны ниже.



*Рис. 14 Планировщик Schedule для solution2*

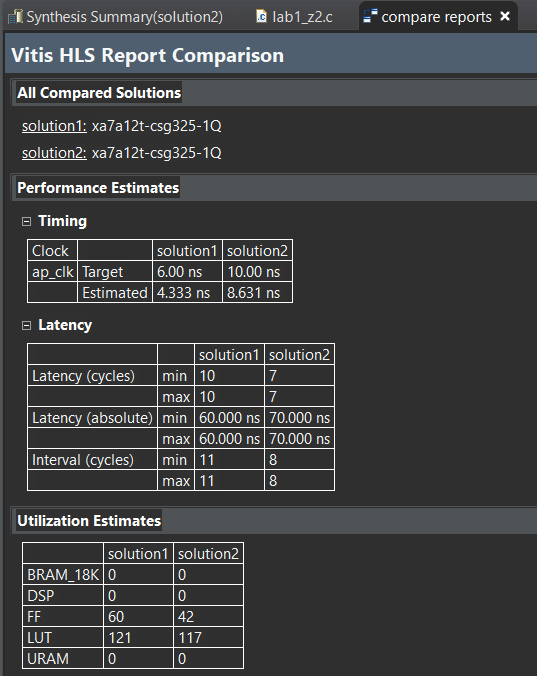
После этого была получена временная диаграмма, представленная на рисунке ниже:



*Рис. 15 Результаты Co-simulation для solution2*

**6. Сравнение Solution1 и Solution2**

Была выполнена команда Compare Reports. После которой получаем результаты сравнения двух решений.



*Рис. 16 Сравнение двух решений*

**7. Выводы**

В данной работе были реализованы два решения с разной частотой, которые могут отличаться в зависимости от задаваемого параметра.