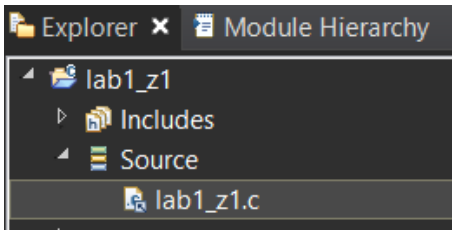
Задание lab1_z1:

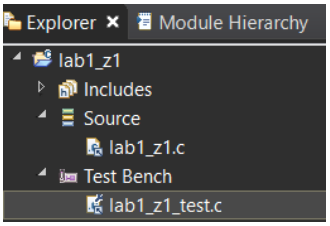
- Разработать на языке Си описание функции, реализующей следующий алгоритм
  y = inA + inB + inC – inD.
- Тип данных для inA, inB, inC, inD – short; для возвращаемого значения y – int;
- Разработать тест, обеспечивающий автоматическую проверку получаемых результатов моделирования разработанной функции.
- 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)

Программа работы и комментарии.

1. ***Создайте рабочие папки***
   1.1. Создайте рабочую папку для проекта C:\Xilinx_trn\HLS2022**lab1_z1**
   1.2. Создайте папку для исходных кодов **C:\Xilinx_trn\HLS2022\lab1_z1\source**
2. ***Создайте проект.***
   2.1. проект lab1_z1 (рабочая папка – папка …\lab1_z1)
   2.2. Микросхема: xa7a12tcsg325-1q
   2.3. clock period 6; clock_uncertainty 1
3. ***Создайте описание функции и теста***
   3.1. Создайте (команда **File => New file**) файл lab1_z1.h (пример файла в приложении) и сохраните в папке **C:\Xilinx_trn\HLS2022\lab1_z1\source**
   3.2. Создайте (команда **Project => New source file**) файлы с описанием функции - файл lab1_z1.c (пример файла в приложении) и сохраните в папке **C:\Xilinx_trn\HLS2022\lab1_z1\source**
   3.2.1.Файл должен появиться в разделе Source



   3.3. Создайте (команда **Project => New Test Bench file**) файл с описанием теста - файл lab1_z1_test.c (пример файла в приложении) и сохраните в папке **C:\Xilinx_trn\HLS2022\lab1_z1\source**
   3.3.1.Файл должен появиться в разделе Test Bench



4. ***Осуществите Си моделирование созданной функции***

4.1. Запустите тест и убедитесь в правильности работы созданного описания функции (приведите снимок экрана с результатами моделирования)

```
----------Pass!------------
INFO: [SIM 1] CSim done with 0 errors.
INFO: [SIM 3] *************** CSIM finish ***************
```

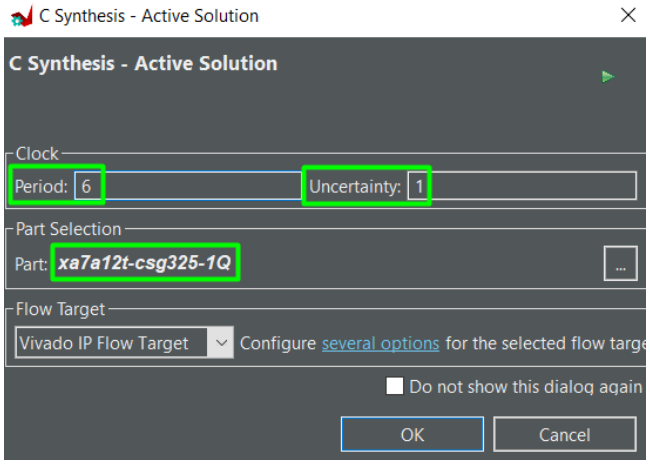
4.2. Убедитесь в правильности работы созданного теста - внесите изменение в тест (например, expected_res = inA + inB + inC + inD) и убедитесь, что тест отрабатывает ошибку (приведите снимок экрана с результатами моделирования)

5. ***Проведите синтез и анализ решения Solution1***

5.1. Синтез

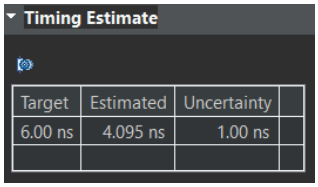
5.1.1.Запустите процедуру синтеза – команда **Solution=>Run C Synthesis=>Active solution**

5.1.2. проверьте параметры, заданные для решения Solution1



5.2. Проведите анализ результатов синтеза

5.2.1. Оценка временных параметров (приведите снимок экрана)

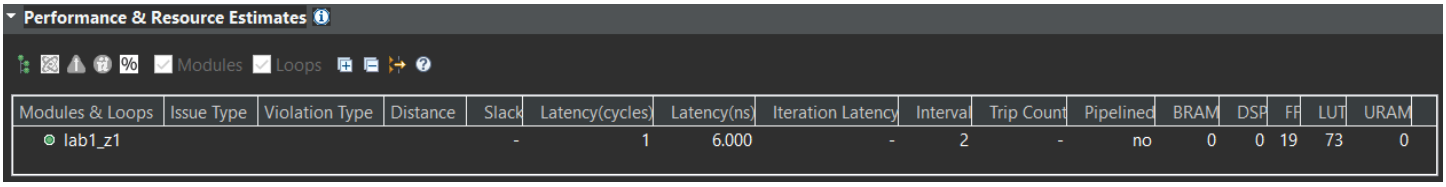


5.2.1.1. Вопросы:

5.2.1.1.1. Что такое Target?

5.2.1.1.2. Чем отличается Target от Estimated?

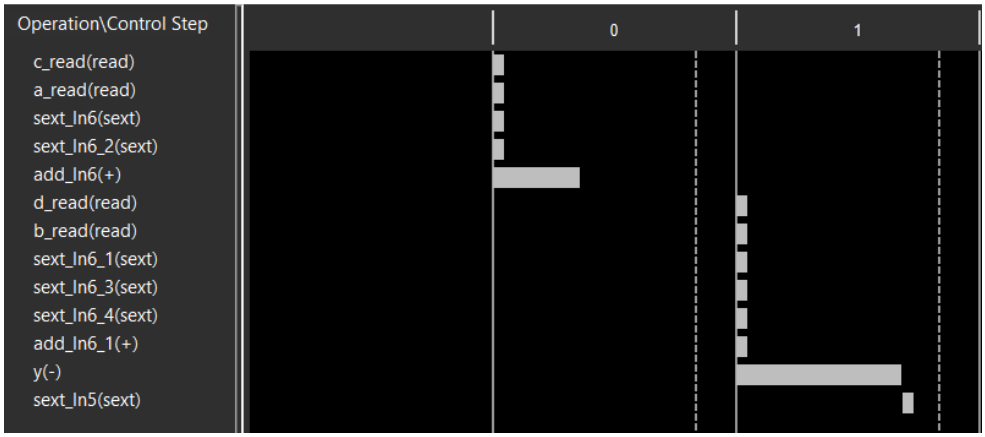
5.2.2.Оценка производительности и аппаратных затрат (приведите снимок экрана)



5.2.2.1. Вопросы:

5.2.2.1.1. Что такое Latency?
5.2.2.1.2. Что такое Interval?
5.2.2.1.3. Что означает значение, приведенное в столбце FF?
5.2.2.1.4. Что означает значение, приведенное в столбце LUT?
5.2.2.1.5. Что означает и как получено значение в столбце Latency (ns)?
5.2.2.1.5.1. Как более точно оценить значение?
5.2.2.1.6. Как оценить значение Interval в ns?
5.2.2.1.7. Что означает **no** в столбце Pipelined?

5.2.3.Окно планировщика (Schedule Viewer) - приведите снимок экрана

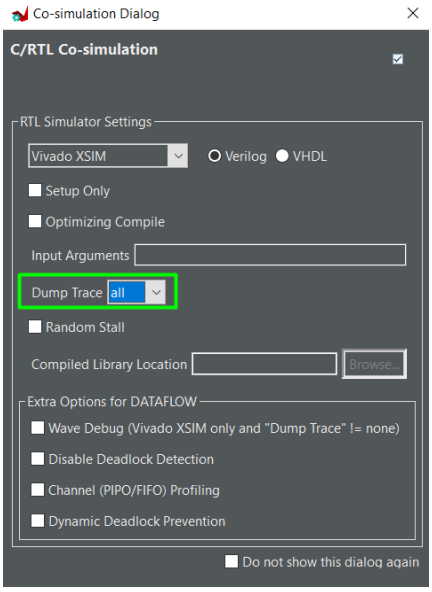


5.2.3.1. Вопросы:
5.2.3.1.1. Почему представлено два столбца 0 и 1?
5.2.3.1.2. Что изображено в столбцах 0 и 1?
5.2.3.1.3. Какие операции будут выполняться при реализации функции?
5.2.3.1.4. Сопоставьте операцию y(-) с исходным кодом – приведите снимок экрана.
5.2.3.1.5. Сопоставьте операцию c_read(read) с исходным кодом.
5.2.3.1.6. Какие аргументы функции будут просуммированы оператором add_ln6(+)?
5.2.3.1.7. Какие аргументы функции будут просуммированы оператором add_ln6_1(+)?

5.3. Выполните и проведите анализ результатов Co-simulation

5.3.1.Запустите процедуру Cosimulation – команда **Solution => Run C\RTL Cosimulation**

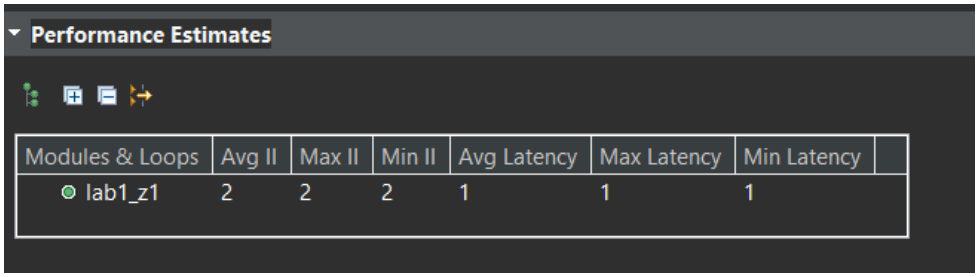
5.3.1.1. Задайте режим **Dump trace All**



5.3.2.Проведите анализ результатов

5.3.2.1. Убедитесь в том, что тест прошел успешно (ожидаемые результаты соответствуют полученным при моделировании) – статус Pass (приведите снимок экрана)

5.3.2.2. Зафиксируйте (приведите снимок экрана) оценки производительности и сравните их с оценками, полученными при синтезе.



5.3.2.2.1. Вопросы:

5.3.2.2.1.1. Что означает Avr, Max, Min?

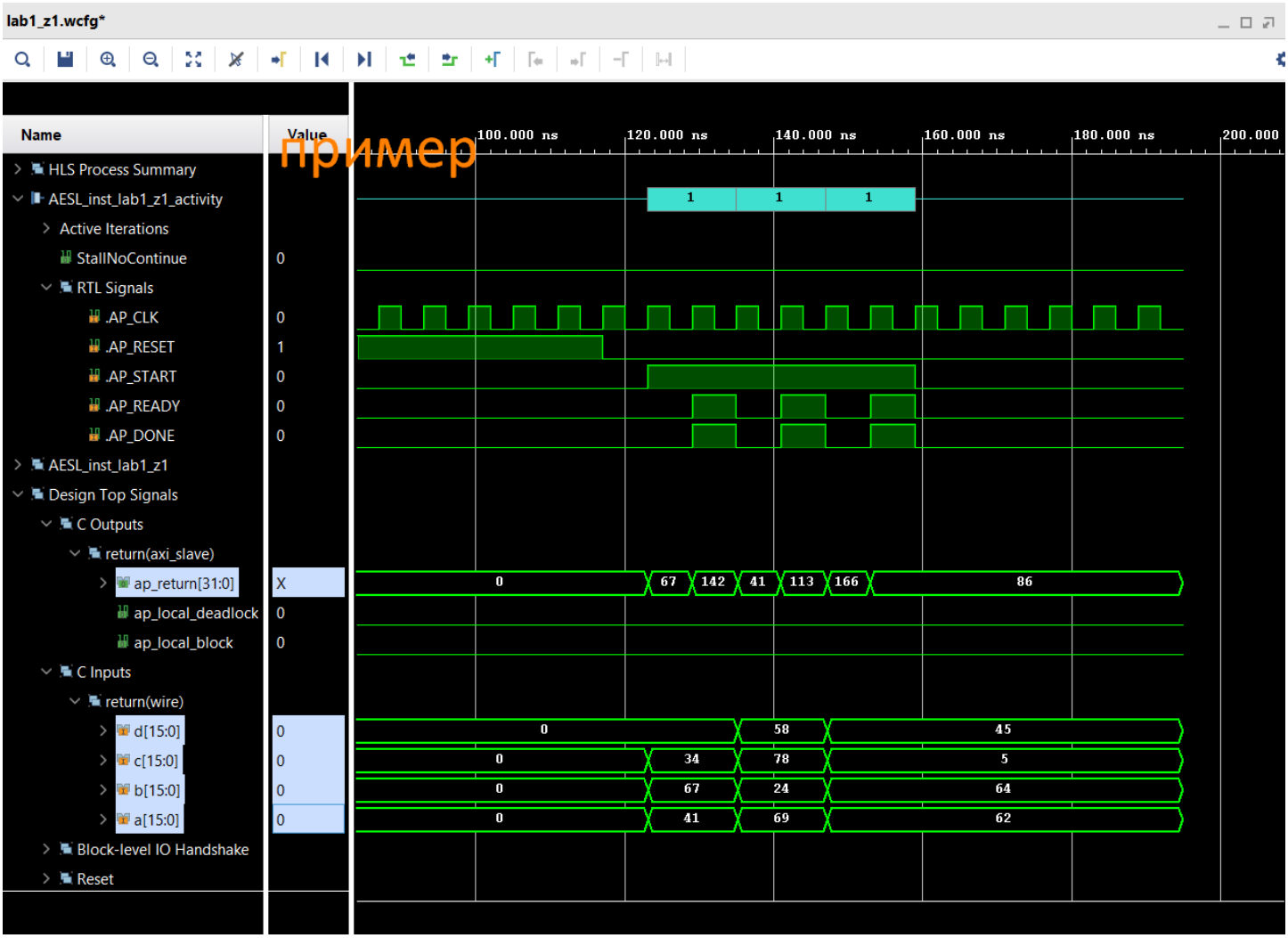
5.3.2.2.1.2. Почему столбы Avr II, Max II, Min II содержат одинаковые результаты?

5.3.3.Получите и проведите анализ временных диаграмм

5.3.3.1. Запустите Wave Viewer (будет запущен пакет Vivado) – команда **Solution=>Open Wave Viewer**

5.3.3.2. Разверните временную диаграмму и приведите снимок экрана (пример ниже) с полученной временной диаграммой

5.3.3.2.1. Выделенные на картинке шины надо отобразить в режиме UNSIGNED DECIMAL



5.3.3.2.2. Вопросы:

5.3.3.2.2.1. Сколько циклов запуска аппаратной реализации созданной функции приведено на временной диаграмме?

5.3.3.2.2.2. В какие моменты времени на шине ap_return[31:0] отображаются правильные выходные значения? (отметьте их на временной диаграмме).

5.3.3.2.2.3. Изобразите на приведенной вами временной диаграмме промежуток времени, определяемый как Latency для всех циклов запуска созданной функции.

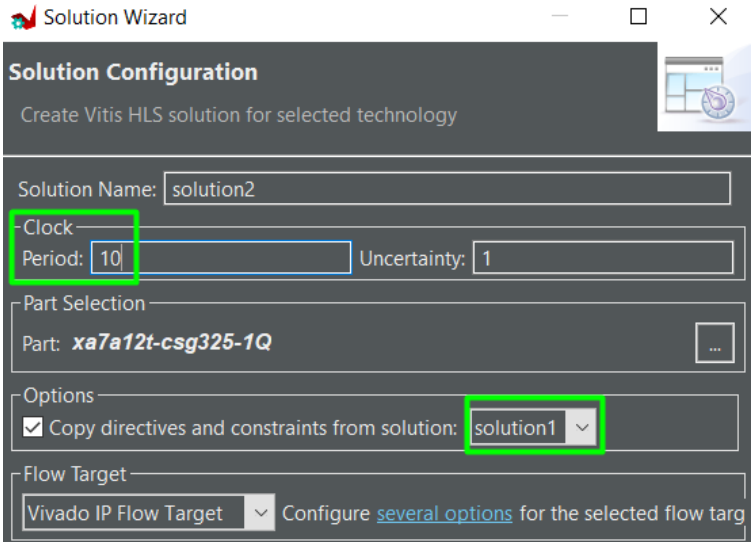
5.3.3.2.2.4. Изобразите на приведенной вами временной диаграмме промежуток времени, определяемый как Iteration Interval (II) для всех циклов запуска созданной функции.

6. Создайте, проведите синтез и анализ решения Solution2

6.1. Создайте новое решение – команда ***Project => New Solution***

6.1.1. Измените период тактового сигнала – сделайте его равным 10нс.

6.1.2. Остальные настройки будут такими же как в Solution1



6.2. Синтез

6.2.1.Запустите процедуру синтеза – команда **Solution=>Run C Synthesis=>Active solution**

6.2.2.Проведите анализ результатов синтеза

6.2.2.1. Оценка временных параметров (приведите снимок экрана)

6.2.2.2. Оценка производительности и аппаратных затрат (приведите снимок экрана)

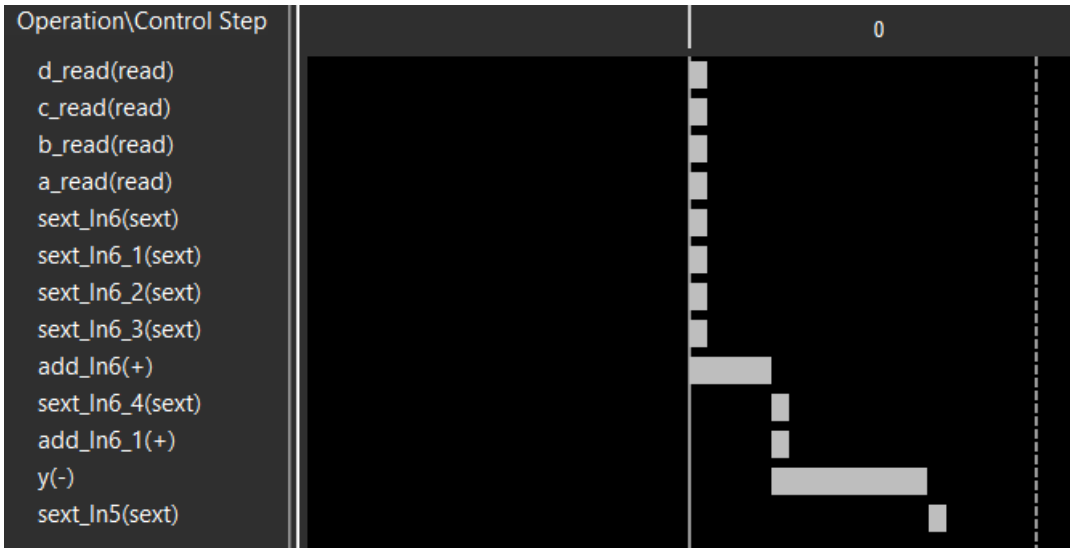
Modules & Loops	Issue Type	Violation Type	Distance	Slack	Latency(cycles)	Latency(ns)	Iteration Latency	Interval	Trip Count	Pipelined	BRAM	DSP	FF	LUT	URAM
lab1_z1				-	0	0.0	-	1	-	no	0	0	0	60	0

6.2.2.2.1. Вопросы:

6.2.2.2.1.1. Почему Latency (cycles) = 0?

6.2.2.2.1.2. Сколько триггеров было использовано при реализации функции?

6.2.2.3. Окно планировщика (Schedule Viewer) - приведите снимок экрана



6.2.2.3.1. Вопросы:

6.2.2.3.1.1. Почему представлен только один столбец 0?

6.2.2.3.1.2. Сопоставьте операцию y(-) с исходным кодом – приведите снимок экрана.

6.3. Выполните и проведите анализ результатов Co-simulation

6.3.1.Запустите процедуру Cosimulation – команда **Solution => Run C\RTL Cosimulation**

6.3.1.1. Задайте режим **Dump trace All**

6.3.2.Проведите анализ результатов

6.3.2.1. Убедитесь в том, что тест прошел успешно (ожидаемые результаты соответствуют полученным при моделировании) – статус Pass (приведите снимок экрана)

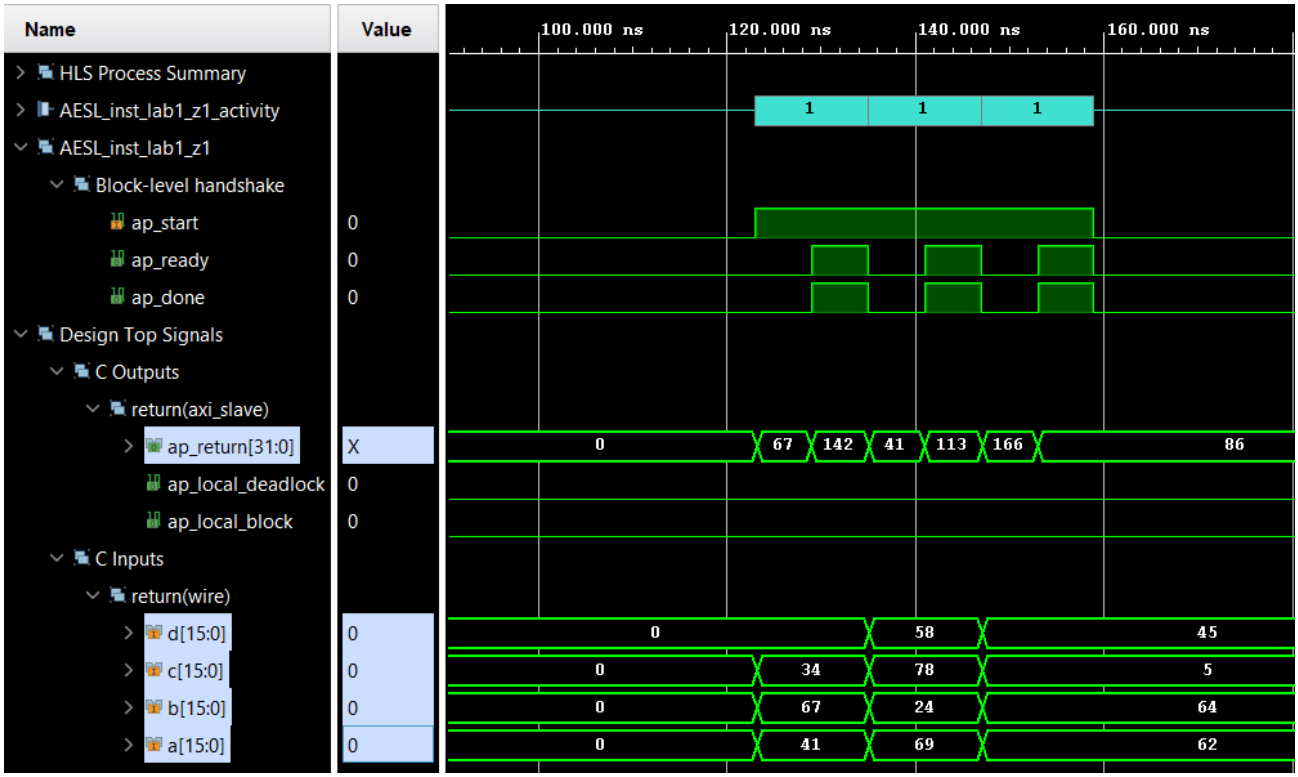
6.3.2.2. Зафиксируйте (приведите снимок экрана) оценки производительности и сравните их с оценками, полученными при синтезе.

6.3.3.Анализ временных диаграмм

6.3.3.1. Запустите Wave Viewer (будет запущен пакет Vivado) – команда **Solution=>Open Wave Viewer**

6.3.3.2. Разверните временную диаграмму и приведите снимок экрана (пример ниже)

6.3.3.2.1. Выделенные на картинке ниже шины надо отобразить в режиме UNSIGNED DECIMAL



6.3.3.2.2. Вопросы:

6.3.3.2.2.1. Сколько циклов запуска аппаратной реализации созданной функции приведено на временной диаграмме?

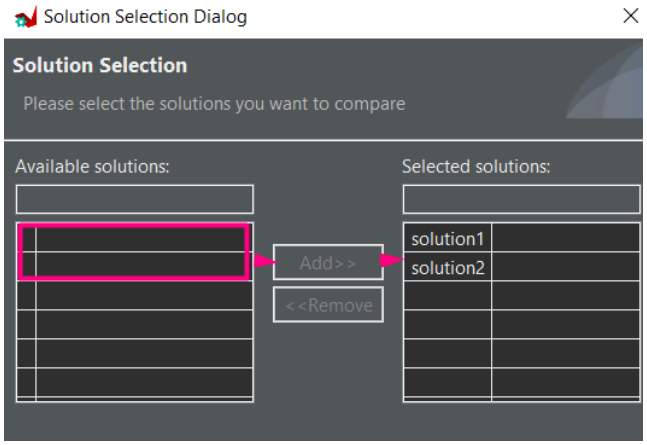
6.3.3.2.2.2. В какие моменты времени на шине ap_return[31:0] отображаются правильные выходные значения? (отметьте их на временной диаграмме).

6.3.3.2.2.3. Почему на временной диаграмме нет тактового сигнала ap_clk, который был на временной диаграмме решения Solution1?.

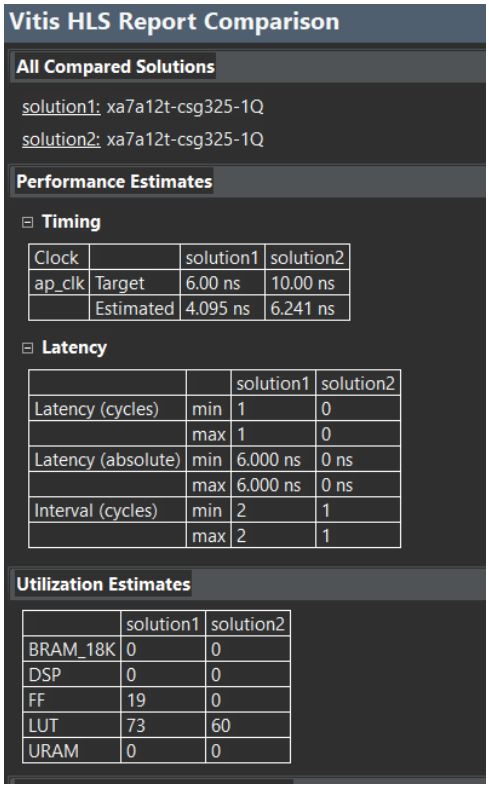
6.3.3.2.2.4. Изобразите на приведенной вами временной диаграмме промежуток времени, определяемый как Iteration Interval (II) для всех циклов запуска созданной функции.

7. Сравните решения Solution 1 и Solution 2 -

7.1. Выполните команду **Project => Compare Reports** и выберите решения для сравнения



7.2. Зафиксируйте результаты сравнения (сделайте снимок экрана)



7.2.1.Вопросы:

7.2.1.1. Как определить минимальный период времени между моментами подачи новых данных на вход аппаратно реализованной функции для решения Solution1 и Solution2?

7.2.1.2. Какой минимальный период времени между моментами поступления новых данных для решения Solution1 и Solution2?

7.2.1.3. Какое решение более производительное?

Приложение

lab1_z1.h

```
typedef short   din_type;
typedef int     dout_type;

// Prototype of top level function for C-synthesis
dout_type lab1_z1( din_type a, din_type b, din_type c, din_type d);
```

lab1_z1.c

```
#include "lab1_z1.h"

dout_type lab1_z1( din_type a, din_type b, din_type c, din_type d)
{
    dout_type y;
    y = a + b + c - d;
    return y;
}
```

lab1_z1_test.c

```
#include <stdio.h>
#include "lab1_z1.h"

int main()
{
    din_type    inA, inB, inC, inD;
    dout_type   actual_res, expected_res;
    int pass = 0;
    int i;
    // initial settings
        //srand(time(NULL));
    // Call the function for 3 transactions
    for (i=0; i<3; i++)
    {
    //input data settings
        inA = rand() % 100;
        inB = rand() % 100;
        inC = rand() % 100;
        inD = rand() % 100;
    //function invocation
        actual_res      = lab1_z1(inA, inB, inC, inD);
    //expected results evaluation
        expected_res    = inA + inB + inC - inD;
    // Test the actual results against the expected results
        if (actual_res != expected_res)
        {
            pass = 1;
            fprintf(stdout, " ERROR: expected=%d  actual=%d  for inputs: inA=%d inB=%d inC=%d inD=%d
                \n",                expected_res, actual_res,             inA,  inB,   inC,   inD);
        }
    }
    if (!pass)
        fprintf(stdout, "----------Pass!-----------\n");
    else
        fprintf(stdout, "----------Fail!-----------\n");
    return pass;
}
```