- Создать проект lab7_z4
- Микросхема: **xc7a100tcsg324-2**

- Создать на языке C++ функцию (N=128, din_type – **ap_int<256>**, dout_type - **ap_int<256>**) –

  - lab7_z4.h

```
#define N 128
#include "ap_int.h"
typedef   ap_int<256>  din_type;
typedef   ap_int<256>  dout_type;

void lab7_z4(dout_type a[N], din_type b[N], din_type c[N]);
```

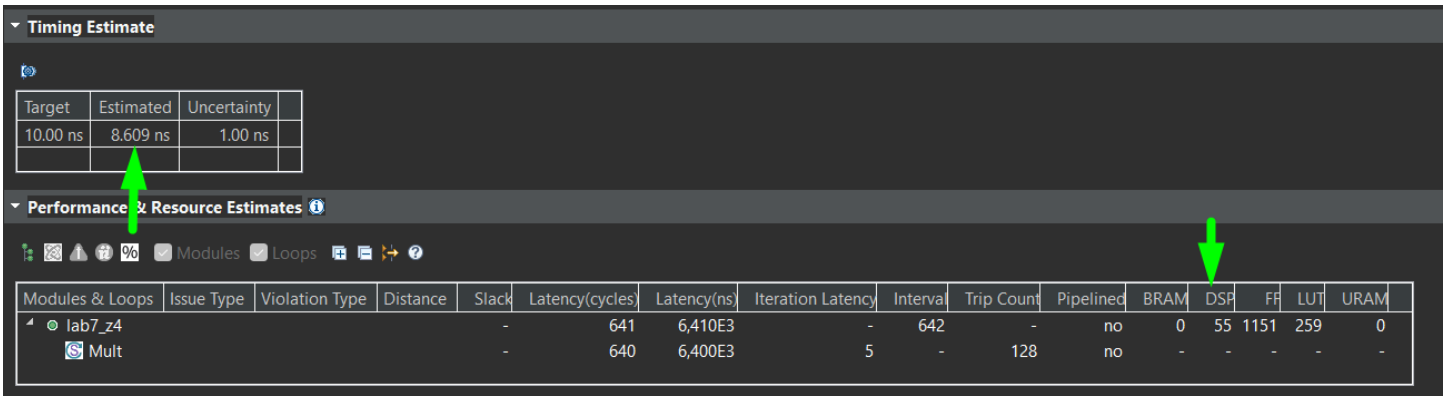
  - lab7_z4.cpp

```
#include "lab7_z4.h"
void lab7_z4(dout_type a[N], din_type b[N], din_type c[N])
{
    dout_type temp_mult;
    Mult:for (int i = 0; i < N; i++)
    {
        temp_mult = b[i] * c[i] ;
        a[i] = temp_mult;
    }
}
```

- Создать тест lab7_z4_test.cpp - исходный код аналогичен lab7_z1

Исследование:

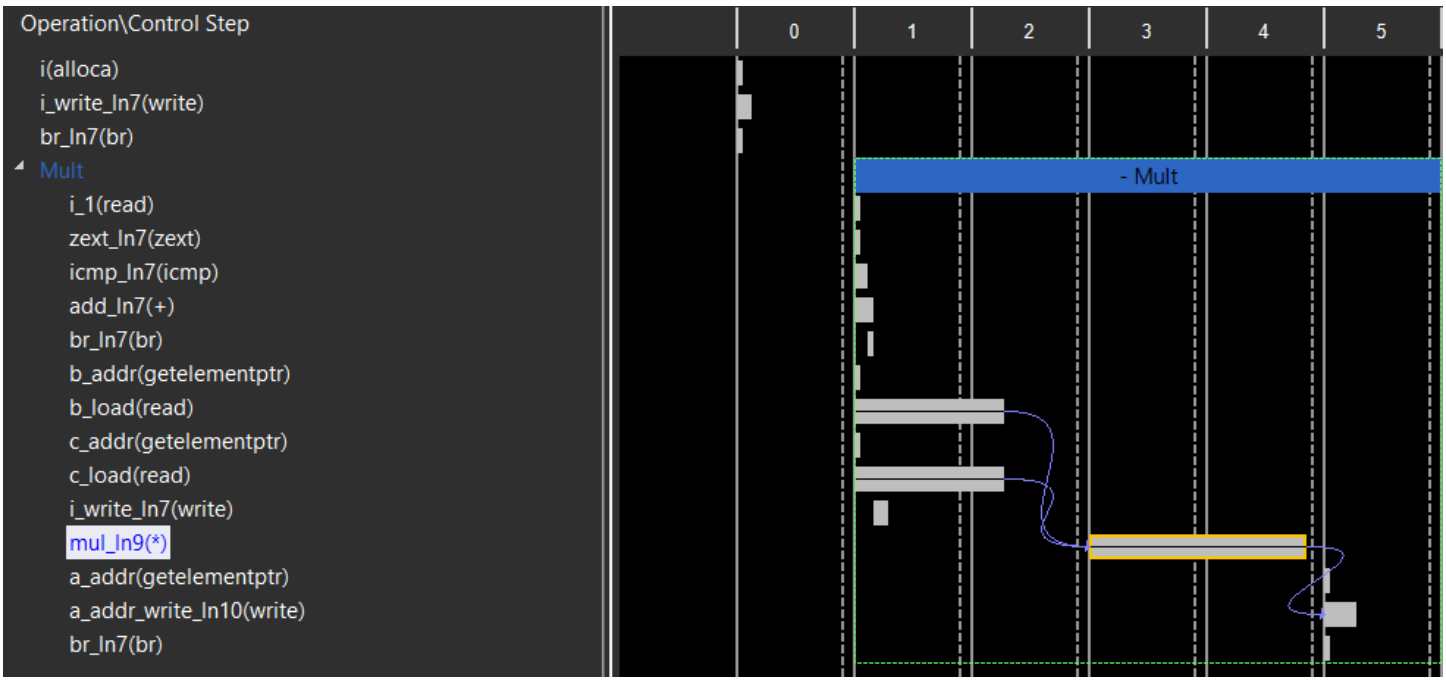
- Solution1
  - clock period 10 ; clock_uncertainty =1
  - Выключите конвейеризацию для цикла Mult
  - осуществите синтез.
  - Посмотрите на отчет – он должен быть похож на приведенный ниже



Сколько модулей умножения (DSP) требуется для реализации?

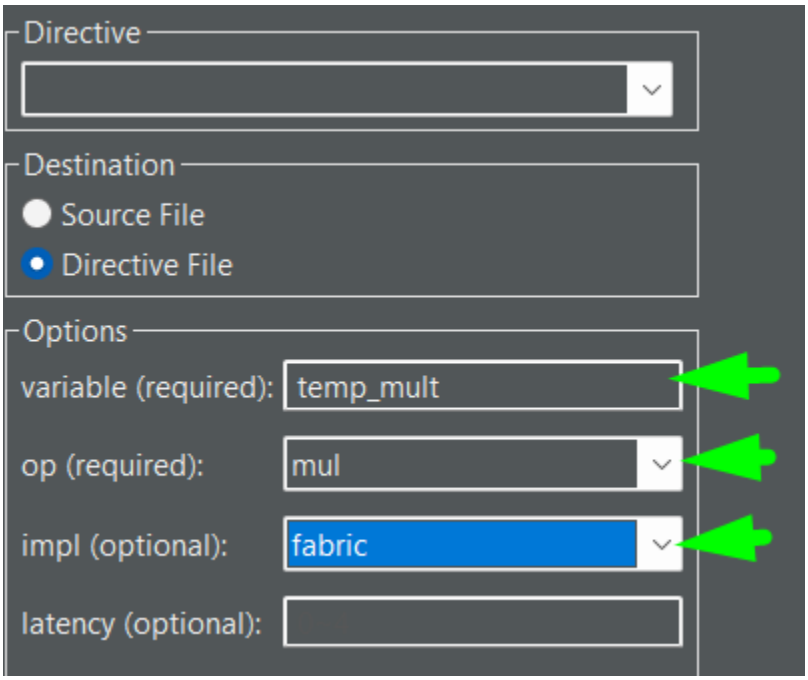
Какой период тактового сигнала (оцениваемый)?

- Посмотрите на Schedule Viewer – должен быть похож на приведенный ниже

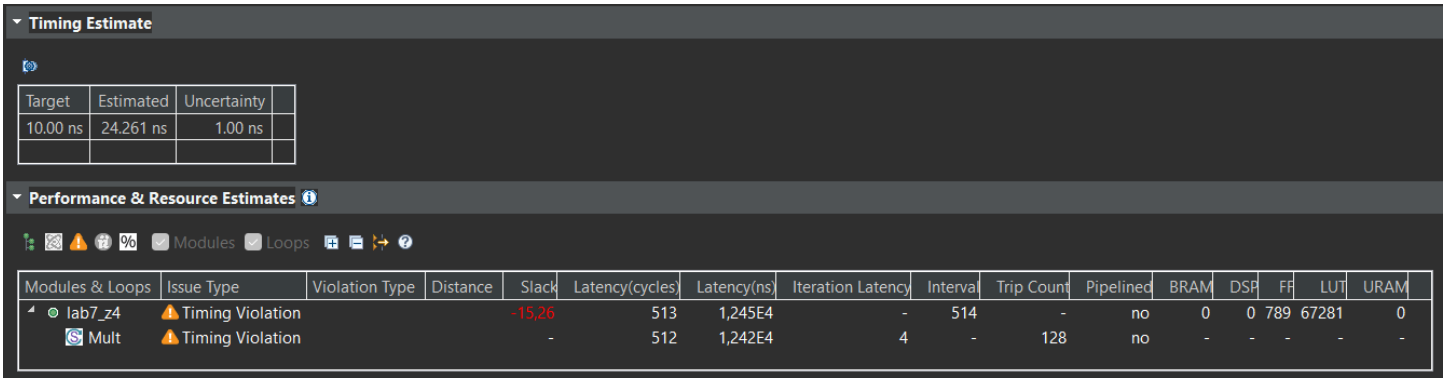


Сколько тактов занимает выполнение операции умножения?

- Solution1_1
  - clock period 10 ; clock_uncertainty =1
  - Выключите конвейеризацию для цикла Mult
  - Задайте BIND_OP для temp_mult

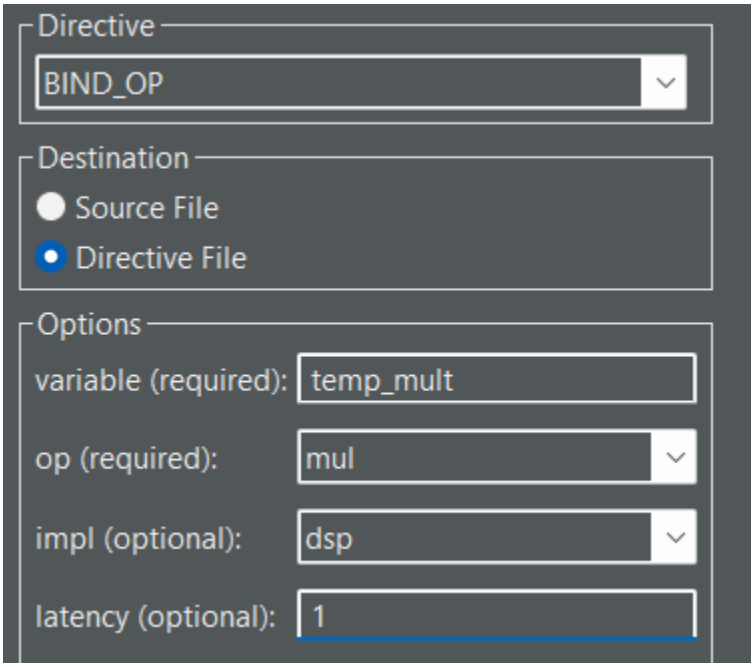


  - осуществите синтез.
  - Посмотрите на отчет – он должен быть похож на приведенный ниже



    - Что за ошибка появилась в отчете? Объясните ее появление и как с ней бороться.
    - Сколько модулей DSP требуется для реализации умножителя 256 разрядных чисел?
    - Сколько логических элементов (LUT) требуется?

- Solution 2 (на основе решения Solution1)
    - clock period 10 ; clock_uncertainty =1
    - Задайте максимально возможный Unroll Factor цикла MULT (чтобы использовались только DSP)
    - Используйте Array Partition (или Array Reshape) нужного типа и фактора, обеспечивающего балансировку производительности умножителей и чтения/записи данных (использование одно портовой или двух портовой памяти – на ваш выбор).
    - **Включите** конвейеризацию для цикла Mult
    - Для переменной temp_mult задайте директиву BIND_OP.



    - осуществите синтез.
    - Посмотрите на отчет

Сколько модулей умножения (DSP) требуется для реализации?

Какой период тактового сигнала (оцениваемый)?

Сколько тактов занимает выполнение операции умножения?

    - Посмотрите на Schedule Viewer

Сколько операций считывания данных осуществляется параллельно?

Сколько операций умножения осуществляется параллельно?

Сколько операций записи данных осуществляется параллельно?

Осуществляется ли конвейеризация? Какой II?

    - Запустите CoSimulation,
        - Посмотрите и зафиксируйте Wave Viewer – подготовьтесь дать пояснения.

Измерение времени выполнения на ПК

- Используются исходные коды функции lab7_z4.cpp (**solution_2**)
- На базе теста **lab7_z4_test.cpp следует** создать отдельный, модернизированный, тест **lab7_z4_testSW.cpp** (сохранить в папке C:\Xilinx_trn\HLS2023\lab7_z4\source) для проверки времени выполнения функции lab7_z4 на ПК. Исходные данные входных массивов должны быть псевдослучайными **из всего диапазона ap_int<256>**
    - **Как использовать 256 разрядные данные в C++:**
        - Boost.Multiprecision
            - https://stacktuts.com/how-to-implement-big-int-in-c
        - Или https://habr.com/ru/articles/595535/
- Следует осуществить компиляцию модернизированного теста и запускать его как отдельное приложение
    - **Следует сделать две реализации кода**
        - **Для одного ядра (потока) – базовая реализация**
        - **Для N ядер/потоков (где N число ядер/потоков в вашем ПК) – например так:** https://stackoverflow.com/questions/414714/compiling-with-g-using-multiple-cores
- Следует провести измерение времени выполнения синтезируемой функции на Вашем ПК **для каждого** из случаев
    - Для одного ядра
        - N = 8192
        - N = 16384
        - N = 32768
        - N = 65 536
    - Для N ядер
        - N = 8192
        - N = 16384
        - N = 32768
        - N = 65 536
- среди 32 запусков (каждого варианта) необходимо найти и зафиксировать медиану значения времени выполнения.

Измерение времени выполнения на аппаратной реализации

- Используются исходные коды функции lab7_z4.cpp (solution2)
- следует осуществить синтез для случаев
    - N = 8192
    - N = 16384
    - N = 32768
    - N = 65 536

  и для каждого случая зафиксировать: II, Estimated period, время выполнения = II * Estimated period

Сравнительный анализ

- Составить xls таблицу и построить графики
    - по оси X – случаи
        - N = 8192
        - N = 16384
        - N = 32768
        - N = 65 536
    - по У – время выполнения функции на ПК (медиана времени выполнения) – для двух вариантов реализации и аппаратной реализации

Отчет, должен включать

- Задание
- Раздел с описанием исходного кода функции
- Раздел с описанием теста
- Раздел с описание созданного командного файла
- Раздел с описанием результатов сравнения решений (со снимками экрана)
- Раздел с анализом результатов
  - Анализ и выбор оптимального (критерий максимальная производительность) решения
- Раздел с описанием модернизированного теста
  - Следует указать компилятор, используемый для компиляции.
- Результаты измерения **времени выполнения на ПК**
  - Следует указать: тип процессора, базовую частоту работы, максимальную частоту работы, объем ОЗУ.
- Результаты измерения времени выполнения на аппаратной реализации
- Раздел с анализом результатов
- Выводы

Архив должен включать всю рабочую папку проекта (включая модернизированный тест, xls таблицу и **скомпилированные приложения – папка ..\source**), отчет