Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**Отчет по лабораторной работе №9\_1**

**Курс: «Проектирование реконфигурируемых гибридных вычислительных систем»**

**Тема: Optimizing array structures**

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

(подпись)

Руководитель Антонов А.П.

(подпись)

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

Санкт – Петербург

2019

ОГЛАВЛЕНИЕ

[**1. Задание** 3](#_Toc27955563)

[1.1 Код программы и теста 3](#_Toc27955564)

[Задание: 5](#_Toc27955565)

[**2.Решение №1.** 6](#_Toc27955566)

[2.1. Синтез 6](#_Toc27955567)

[**3. Второе решение** 7](#_Toc27955568)

[3.1.Директива 7](#_Toc27955569)

[3.2 Синтез 8](#_Toc27955570)

[3.3. Вывод логов. 9](#_Toc27955571)

[**4. Решение №3.** 10](#_Toc27955572)

[4.1 Параметры третьего решения 10](#_Toc27955573)

[4.2 Синтез 11](#_Toc27955574)

[**5. Вывод** 12](#_Toc27955575)

# **1. Задание**

# 1.1 Код программы и теста

|  |
| --- |
| #include "dct.h"  void dct\_1d(dct\_data\_t src[DCT\_SIZE], dct\_data\_t dst[DCT\_SIZE])  {  unsigned int k, n;  int tmp;  const dct\_data\_t dct\_coeff\_table[DCT\_SIZE][DCT\_SIZE] = {  #include "dct\_coeff\_table.txt"  };  DCT\_Outer\_Loop:  for (k = 0; k < DCT\_SIZE; k++) {  DCT\_Inner\_Loop:  for(n = 0, tmp = 0; n < DCT\_SIZE; n++) {  int coeff = (int)dct\_coeff\_table[k][n];  tmp += src[n] \* coeff;  }  dst[k] = DESCALE(tmp, CONST\_BITS);  }  }  void dct\_2d(dct\_data\_t in\_block[DCT\_SIZE][DCT\_SIZE],  dct\_data\_t out\_block[DCT\_SIZE][DCT\_SIZE])  {  dct\_data\_t row\_outbuf[DCT\_SIZE][DCT\_SIZE];  dct\_data\_t col\_outbuf[DCT\_SIZE][DCT\_SIZE], col\_inbuf[DCT\_SIZE][DCT\_SIZE];  unsigned i, j;  // DCT rows  Row\_DCT\_Loop:  for(i = 0; i < DCT\_SIZE; i++) {  dct\_1d(in\_block[i], row\_outbuf[i]);  }  // Transpose data in order to re-use 1D DCT code  Xpose\_Row\_Outer\_Loop:  for (j = 0; j < DCT\_SIZE; j++)  Xpose\_Row\_Inner\_Loop:  for(i = 0; i < DCT\_SIZE; i++)  col\_inbuf[j][i] = row\_outbuf[i][j];  // DCT columns  Col\_DCT\_Loop:  for (i = 0; i < DCT\_SIZE; i++) {  dct\_1d(col\_inbuf[i], col\_outbuf[i]);  }  // Transpose data back into natural order  Xpose\_Col\_Outer\_Loop:  for (j = 0; j < DCT\_SIZE; j++)  Xpose\_Col\_Inner\_Loop:  for(i = 0; i < DCT\_SIZE; i++)  out\_block[j][i] = col\_outbuf[i][j];  }  void read\_data(short input[N], short buf[DCT\_SIZE][DCT\_SIZE])  {  int r, c;  RD\_Loop\_Row:  for (r = 0; r < DCT\_SIZE; r++) {  RD\_Loop\_Col:  for (c = 0; c < DCT\_SIZE; c++)  buf[r][c] = input[r \* DCT\_SIZE + c];  }  }  void write\_data(short buf[DCT\_SIZE][DCT\_SIZE], short output[N])  {  int r, c;  WR\_Loop\_Row:  for (r = 0; r < DCT\_SIZE; r++) {  WR\_Loop\_Col:  for (c = 0; c < DCT\_SIZE; c++)  output[r \* DCT\_SIZE + c] = buf[r][c];  }  }  void dct(short input[N], short output[N])  {  short buf\_2d\_in[DCT\_SIZE][DCT\_SIZE];  short buf\_2d\_out[DCT\_SIZE][DCT\_SIZE];  // Read input data. Fill the internal buffer.  read\_data(input, buf\_2d\_in);  dct\_2d(buf\_2d\_in, buf\_2d\_out);  // Write out the results.  write\_data(buf\_2d\_out, output);  } |
| #ifndef \_\_DCT\_H\_\_  #define \_\_DCT\_H\_\_  #define DW 16  #define N 1024/DW  #define NUM\_TRANS 16  typedef short dct\_data\_t;  #define DCT\_SIZE 8 /\* defines the input matrix as 8x8 \*/  #define CONST\_BITS 13  #define DESCALE(x,n) (((x) + (1 << ((n)-1))) >> n)  void dct(short input[N], short output[N]);  #endif // \_\_DCT\_H\_\_ not defined |

## Задание:

* Launch the Vivado® HLS tool.

Open the provided dct\_prj Vivado HLS tool project located at: C:\training\optimize\_array\_performance\demo\dct\_prj

* Access and review the source files (dct.c and dct.h) from the Explorer pane.
* Run C synthesis.
* Review the Synthesis report.
* Create a new solution named solution2 (select Project > New Solution).
* Accept the default settings and click Finish.
* Apply the PIPELINE directive on DCT\_Outer\_Loop of the dct\_1d function (shown below).
* Similarly, apply the PIPELINE directive to the following loops:
* Xpose\_Row\_Inner\_Loop of the dct\_2d function
* Xpose\_Col\_Inner\_Loop of the dct\_2d function
* RD\_Loop\_Col of the read\_data function
* WR\_Loop\_Col of the write\_data function
* Compare the results of two solutions (solution1 and solution2).
* What is the worst-case latency of the design?
* Go to the Utilization Estimates section and note the number of DSP48E and block RAMs used to implement solution2.
* Select the Console tab and review the synthesis information.
* Create a new solution named solution3.
* Accept the default settings and click Finish.
* Apply the ARRAY\_PARTITION directive to buf\_2d\_in of the dct function as shown in the figure below.
* Similarly, apply the ARRAY\_PARTITION directive col\_inbuf of the dct\_2d function.
* Run C synthesis.
* Compare the results of the two solutions (solution2 and solution3).

# **2.Решение №1.**

## 2.1. Синтез

Результаты синтеза с оценкой производительности и используемых ресурсов представлены на рисунках 2.1 и 2.2 соответственно.

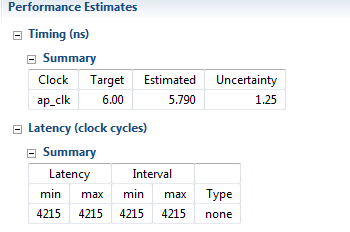


Рисунок 2.1. Отчет о синтезе

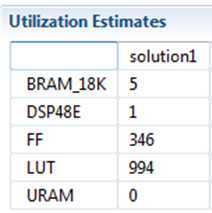


Рисунок 2.2. Отчет об используемых ресурсах

Величина Latency составила 4215.

Количество ресурсов в проекте составило 5 блоков BRAM\_18K и один блок DSP48E.

# **3. Второе решение**

## 3.1.Директива

Пропишем директиву PIPELINE, рисунок 3.1.

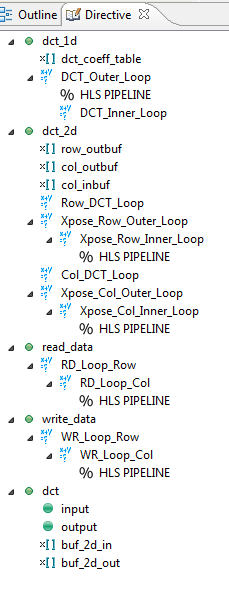


Рисунок 3.1. Параметры второго решения

## 3.2 Синтез

Результаты синтеза с оценкой производительности и сравнительном анализе используемых ресурсов первого и второго решения представлены на рисунках 3.2 и 3.3 соответственно.

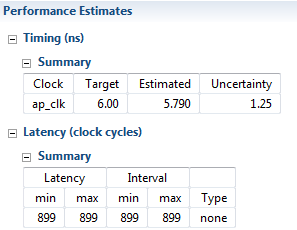


Рисунок 3.2. Отчет о синтезе

Величина Latency составила 899.

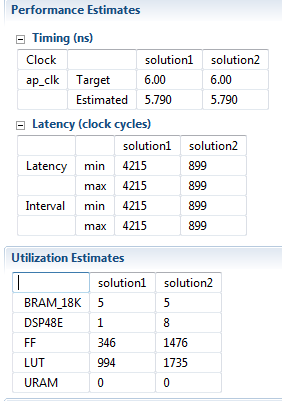


Рисунок 3.3. Сравнительный отчет первого и второго решений

В проекте используется 5 блоков BRAM и 8 блоков DSP.

## 3.3. Вывод логов.

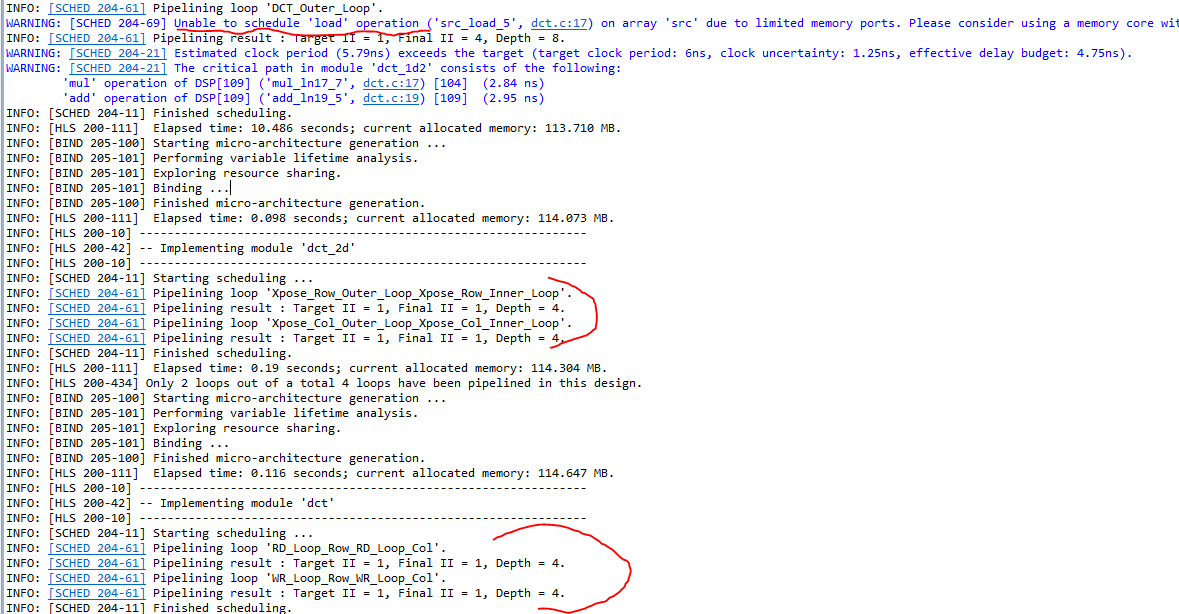


Рисунок 3.4. Окно Console

Как видно из сообщения, не удалось достичь требуемого значения Initiation interval для DCT\_Outer\_Loop в связи с нехваткой портов памяти. Для достижения требуемой пропускной способности нужно разбить массивы col\_inbuf и in\_block.

# **4. Решение №3.**

## 4.1 Параметры третьего решения

Пропишем соответствующие директивы, рисунок 4.1.

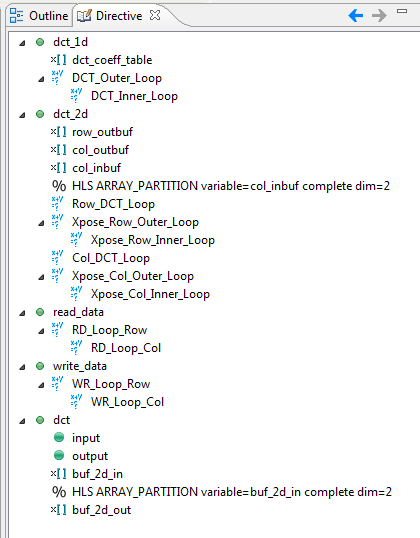


Рисунок 4.1. Параметры третьего решения

## 4.2 Синтез

Результаты сравнительного анализа результатов синтеза третьего решения с первым и вторым и сравнительными оценками производительности и используемых ресурсов представлены на рисунках 4.2.

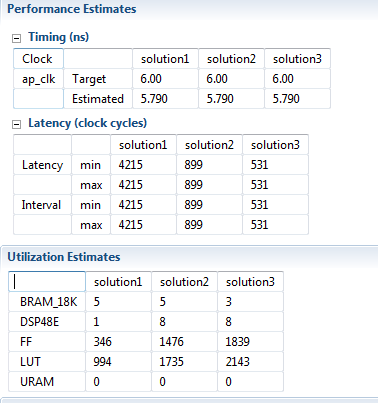


Рисунок 4.2. Сравнительный отчет трех решений

Latency в третьем решении достигла наименьшей величины из трех решений и составила 531.

В проекте используется 3 блока BRAM и 8 блоков DSP.

# **5. Вывод**

Значение Latency уменьшилось с 4215 до 531 при использовании директивы для разбиения массивов что поспособствовало успешному применению директивы PIPELINE и повышению пропускной способности проекта.