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

- 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

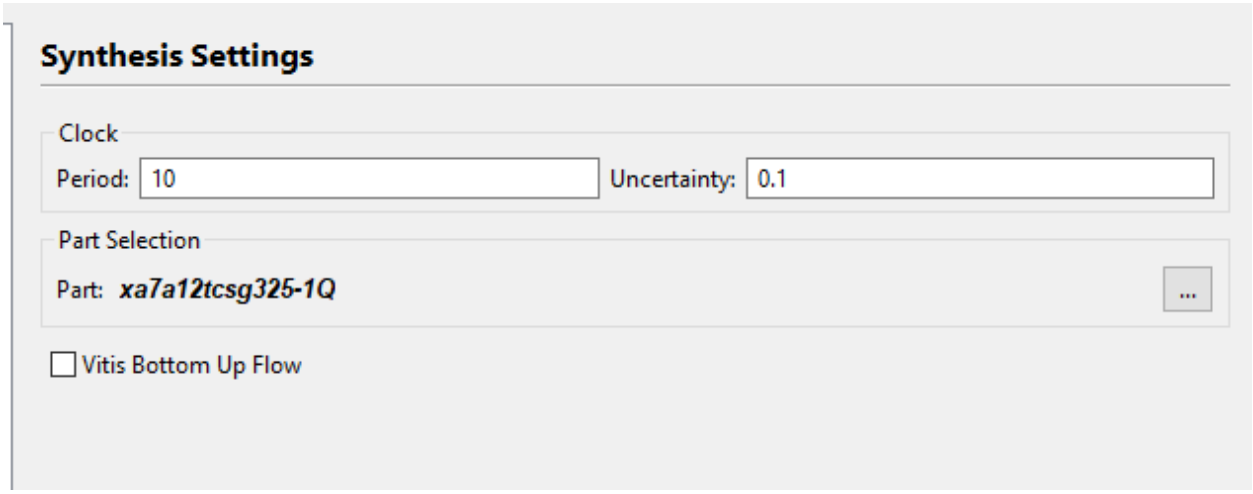
```
int lab1_1( char a, char b, char c, char d) {
	int y;
	y = a*b+c+d;
	return y;
}
```

   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;
```

```
    }
}
```

2. Создание проекта с заданными настройками



3. Результат синтеза:

   Производительность

Performance Estimates

Timing

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	6.00 ns	3.820 ns	0.10 ns

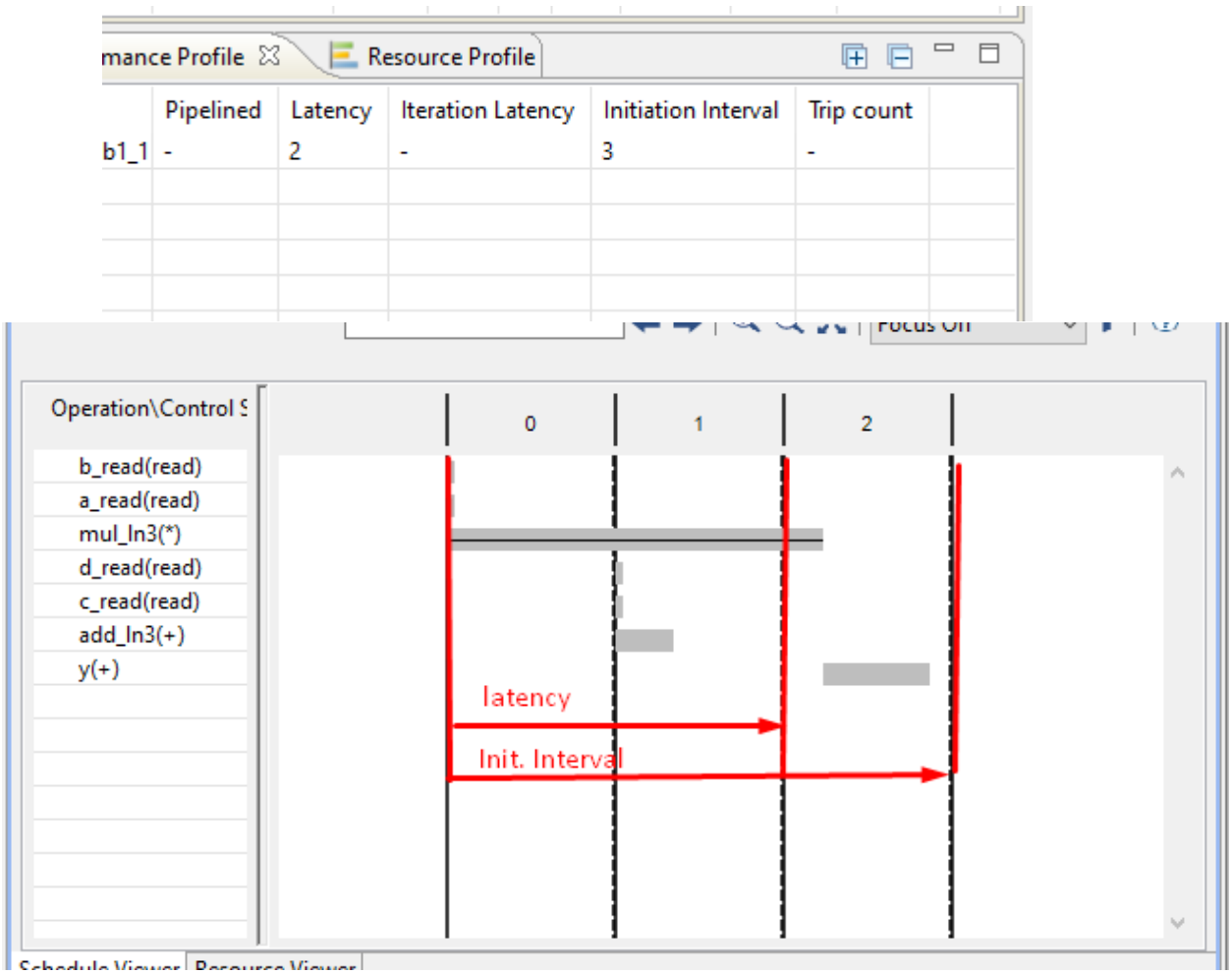
Latency

Summary

Latency (cycles)		Latency (absolute)		Interval (cycles)		
min	max	min	max	min	max	Type
2	2	12.000 ns	12.000 ns	2	2	none

Detail

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:

Utilization Estimates

Summary

Name	BRAM_18K	DSP48E	FF	LUT	URAM
DSP	-	1	-	-	-
Expression	-	-	0	16	-
FIFO	-	-	-	-	-
Instance	-	-	-	-	-
Memory	-	-	-	-	-
Multiplexer	-	-	-	21	-
Register	-	-	12	-	-
Total	0	1	12	37	0
Available	40	40	16000	8000	0
Utilization (%)	0	2	~0	~0	0

Detail

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

Interface

Summary

RTL Ports	Dir	Bits	Protocol	Source Object	C Type
ap_clk	in	1	ap_ctrl_hs	lab1_1	return value
ap_rst	in	1	ap_ctrl_hs	lab1_1	return value
ap_start	in	1	ap_ctrl_hs	lab1_1	return value
ap_done	out	1	ap_ctrl_hs	lab1_1	return value
ap_idle	out	1	ap_ctrl_hs	lab1_1	return value
ap_ready	out	1	ap_ctrl_hs	lab1_1	return value
ap_return	out	32	ap_ctrl_hs	lab1_1	return value
a	in	8	ap_none	a	scalar
b	in	8	ap_none	b	scalar
c	in	8	ap_none	c	scalar
d	in	8	ap_none	d	scalar

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.

4. C/RTL моделирование.

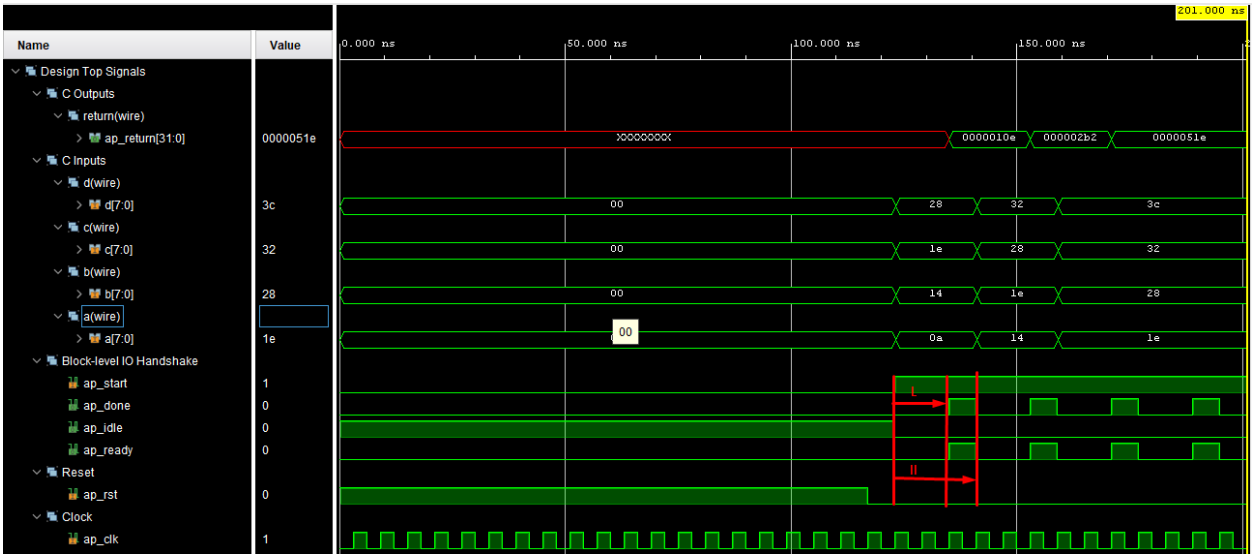
```
webtalk_transmit: Time (s): cpu = 00:00:00 ; elapsed = 00:00:06 . Memory (MB): peak = 110.12
INFO: [Common 17-206] Exiting Webtalk at Thu Dec 12 04:11:34 2019...

****** xsim v2019.2 (64-bit)
  **** SW Build 2708876 on Wed Nov  6 21:40:23 MST 2019
  **** IP Build 2700528 on Thu Nov  7 00:09:20 MST 2019
    ** Copyright 1986-2019 Xilinx, Inc. All Rights Reserved.

start_gui
INFO: [Common 17-206] Exiting xsim at Thu Dec 12 04:14:09 2019...
INFO: [COSIM 212-316] Starting C post checking ...
  10*20+30+40=270
  20*30+40+50=690
  30*40+50+60=1310
----------Pass!------------
INFO: [COSIM 212-1000] *** C/RTL co-simulation finished: PASS ***
Finished C/RTL cosimulation.
```

Cosimulation Report for 'lab1_1'

Result

		Latency			Interval		
RTL	Status	min	avg	max	min	avg	max
VHDL	NA	NA	NA	NA	NA	NA	NA
Verilog	Pass	2	2	2	3	3	3

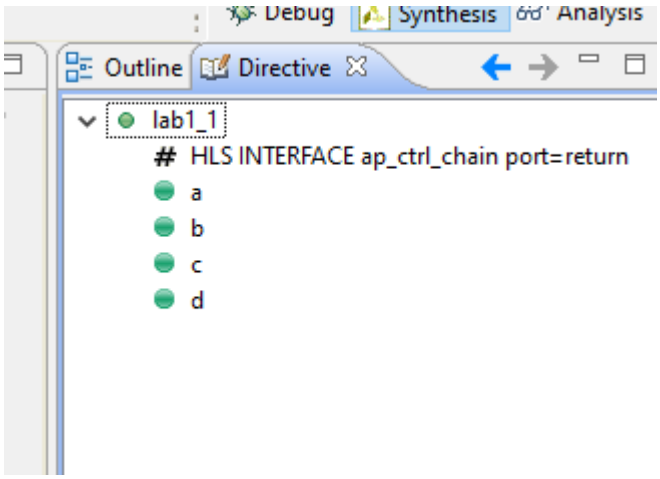
Значения соответствуют полученным ранее.



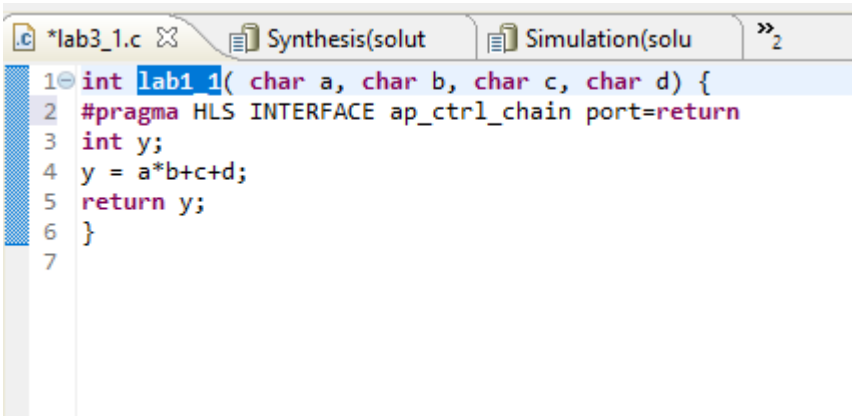
Решение 2

1. Добавление протокола

   Directive

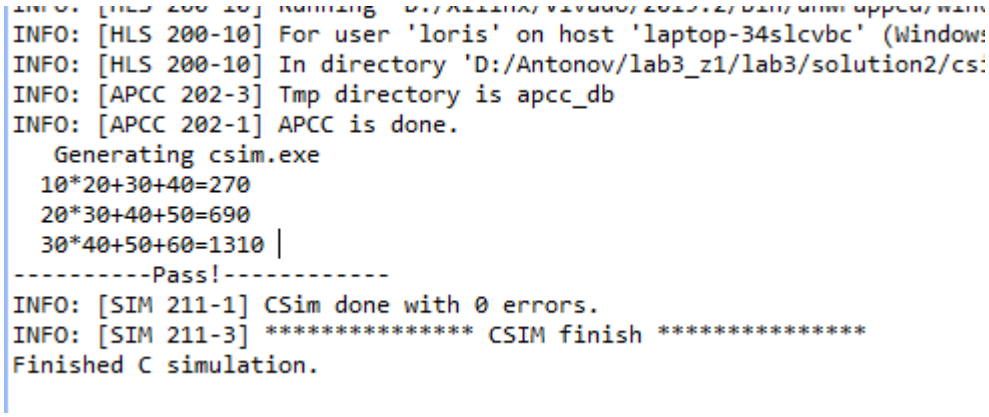


Lab3_1.c



ap_ctrl_chain: протокол ввода-вывода на уровне блоков для цепочки управления. Этот протокол ввода / вывода в основном используется для объединения конвейерных блоков.

2. Моделирование:



Моделирование выполнено успешно.

3. Синтез:

Производительность

Performance Estimates

Timing

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	6.00 ns	3.820 ns	0.10 ns

Latency

Summary

Latency (cycles)		Latency (absolute)		Interval (cycles)		
min	max	min	max	min	max	Type
2	2	12.000 ns	12.000 ns	2	2	none

Detail

Performance Profile | Resource Profile

	Pipelined	Latency	Iteration Latency	Initiation Interval	Trip count
lab1_1	-	2	-	3	-

Значения соответствуют тем, которые были определены в решении 1.

Использование ресурсов

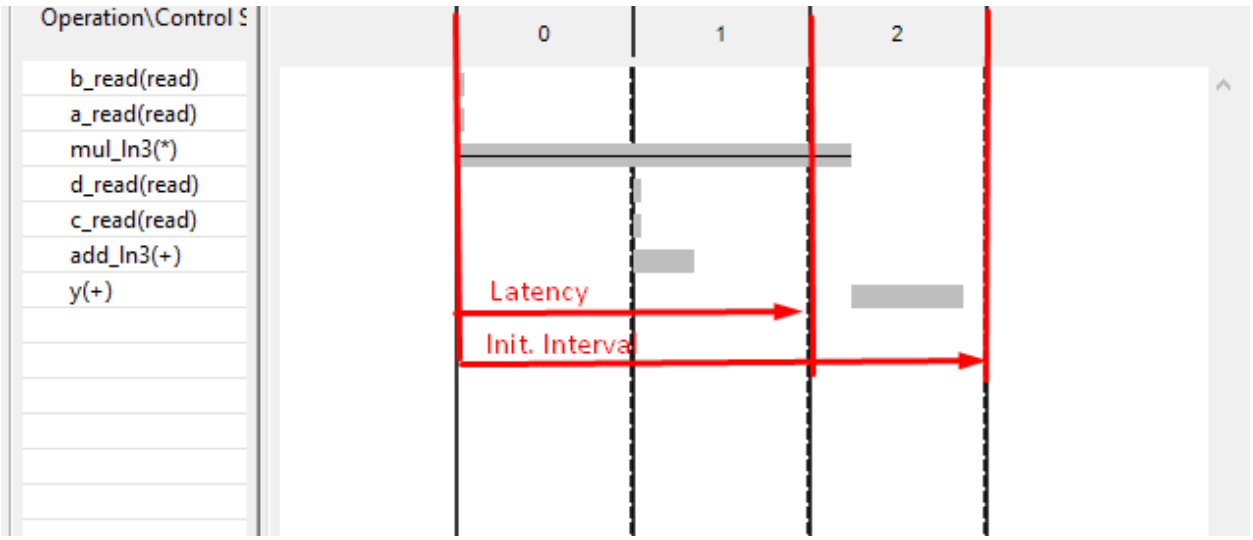
Utilization Estimates

Summary

Name	BRAM_18K	DSP48E	FF	LUT	URAM
DSP	-	1	-	-	-
Expression	-	-	0	18	-
FIFO	-	-	-	-	-
Instance	-	-	-	-	-
Memory	-	-	-	-	-
Multiplexer	-	-	-	39	-
Register	-	-	45	-	-
Total	0	1	45	57	0
Available	40	40	16000	8000	0
Utilization (%)	0	2	~0	~0	0

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

Interface

Summary

RTL Ports	Dir	Bits	Protocol	Source Object	C Type
ap_clk	in	1	ap_ctrl_chain	lab1_1	return value
ap_rst	in	1	ap_ctrl_chain	lab1_1	return value
ap_start	in	1	ap_ctrl_chain	lab1_1	return value
ap_done	out	1	ap_ctrl_chain	lab1_1	return value
ap_continue	in	1	ap_ctrl_chain	lab1_1	return value
ap_idle	out	1	ap_ctrl_chain	lab1_1	return value
ap_ready	out	1	ap_ctrl_chain	lab1_1	return value
ap_return	out	32	ap_ctrl_chain	lab1_1	return value
a	in	8	ap_none	a	scalar
b	in	8	ap_none	b	scalar
c	in	8	ap_none	c	scalar
d	in	8	ap_none	d	scalar

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.

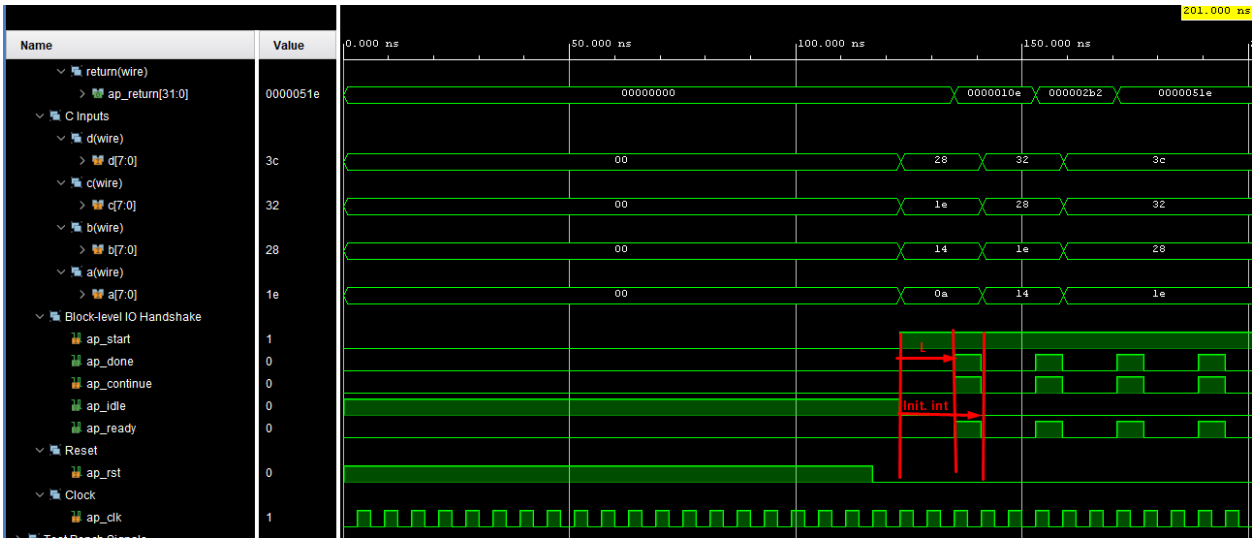
Входные порты: a, b, c, d являются 8-битными входами и имеют входы / выходы, протокол ap_none.

Конструкция имеет 32-битный выходной порт для возврата функции ap_return.

Resource profile

	BRAM	DSP	FF	LUT	Bits P0	Bits P1	Bits P2	Banks/Depth	Words	W*Bits*B
lab1_1	0	1	45	57						
I/O Ports(4)					32					
Instances(0)	0	0	0	0						
Memories(0)	0		0	0	0			0	0	0
Expressions(2)	0	0	0	18	10	10	0			
Registers(4)			45		45					
add_ln4_reg			9		9					
ap_CS_fsm			3		3					
ap_done_re			1		1					
ap_return_p			32		32					
Channels(0)	0		0	0	0			0	0	0
Multiplexers(3)	0		0	39	34			0		
DSP(1)		1								

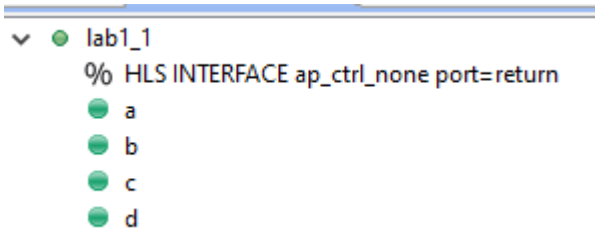
4. C/RTL моделирование



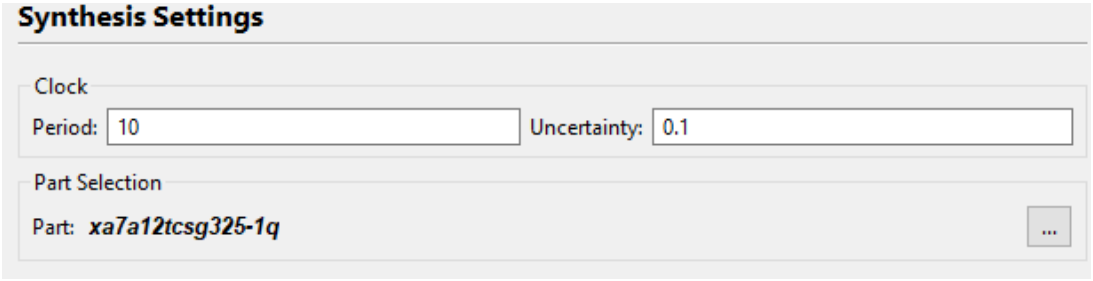
Решение 3

1. Создание и настройка решения.
   Установка протокола

```
3  ## Please DO NOT edit it.
4  ## Copyright (C) 1986-2019 Xilinx, Inc. All Rights Reserved.
5  ############################################################
6  set_directive_interface -mode ap_ctrl_none "lab1_1"
7
```



Конфигурирование решения



2. Моделирование

```
INFO: [APCC 202-1] APCC is done.
   Compiling(apcc) ../../../../source/lab3_1.c in debug mode
INFO: [HLS 200-10] Running 'D:/Xilinx/Vivado/2019.2/bin/unwrapped/win64.o/apcc.exe'
INFO: [HLS 200-10] For user 'loris' on host 'laptop-34slcvbc' (Windows NT_amd64 version 6
INFO: [HLS 200-10] In directory 'D:/Antonov/lab3_z1/lab3/solution3/csim/build'
INFO: [APCC 202-3] Tmp directory is apcc_db
INFO: [APCC 202-1] APCC is done.
   Generating csim.exe
  10*20+30+40=270
  20*30+40+50=690
  30*40+50+60=1310
----------Pass!------------
INFO: [SIM 211-1] CSim done with 0 errors.
INFO: [SIM 211-3] *************** CSIM finish ***************
Finished C simulation.
```

3. Синтез

Производительность

Performance Estimates

Timing

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	10.00 ns	7.180 ns	0.10 ns

Latency

Summary

Latency (cycles)		Latency (absolute)		Interval (cycles)		
min	max	min	max	min	max	Type
0	0	0 ns	0 ns	0	0	none

Detail

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

Performance Profile | Resource Profile

	Pipelined	Latency	Iteration Latency	Initiation Interval	Trip count
lab3_1	-	0	-	1	-

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

Использование ресурсов

Utilization Estimates

Summary

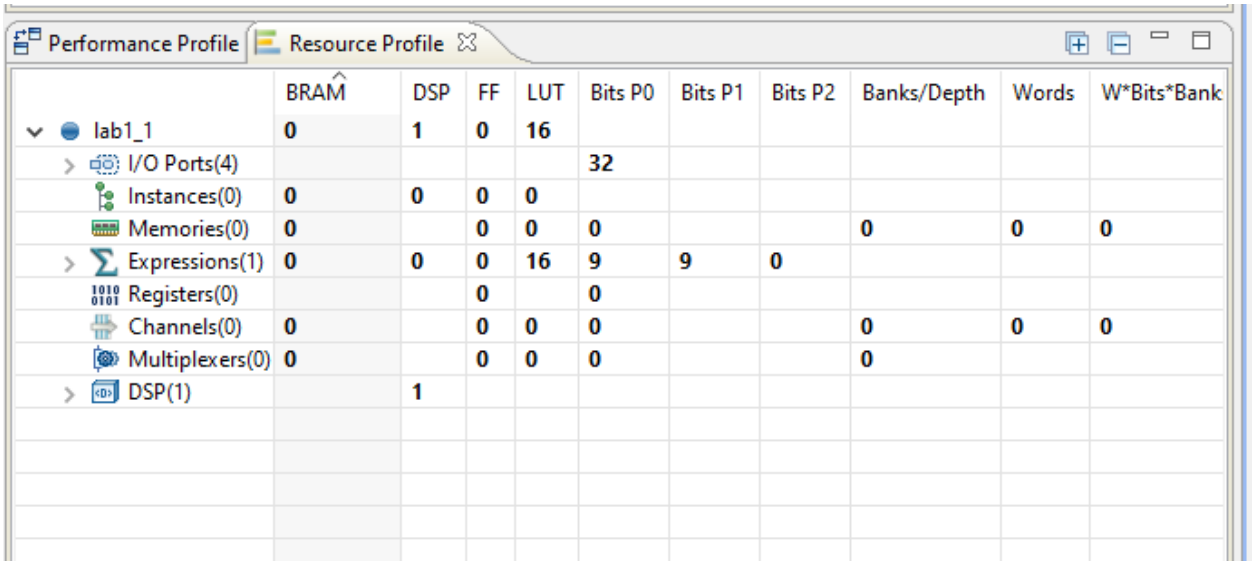
Name	BRAM_18K	DSP48E	FF	LUT	URAM
DSP	-	1	-	-	-
Expression	-	-	0	16	-
FIFO	-	-	-	-	-
Instance	-	-	-	-	-
Memory	-	-	-	-	-
Multiplexer	-	-	-	-	-
Register	-	-	-	-	-
Total	0	1	0	16	0
Available	40	40	16000	8000	0
Utilization (%)	0	2	0	~0	0

Detail

Instance

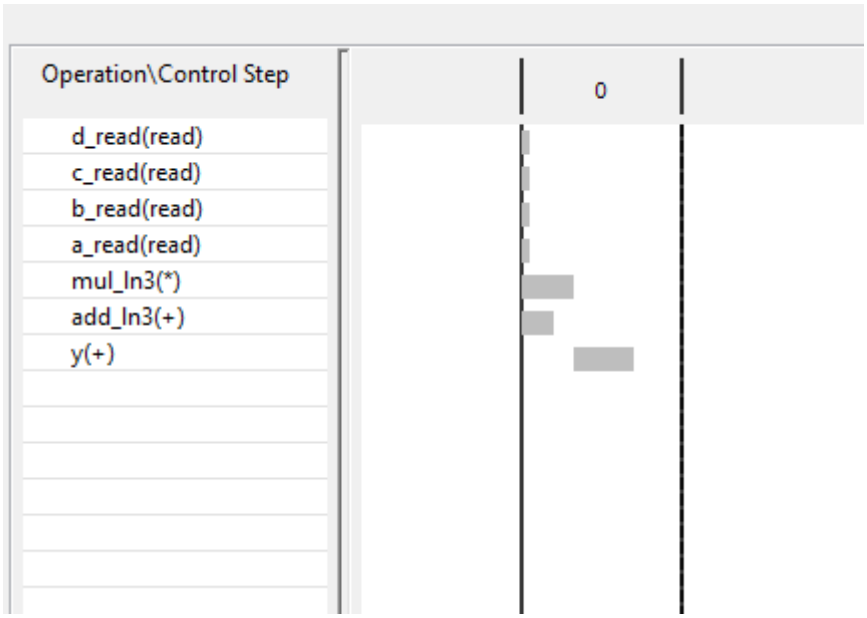
Данный проект будет занимать на микросхеме:
1 DSP блок, где будут задействованы сумматор и умножитель.
16 LUT.

По сравнению с предыдущими решениями использование регистров полностью отсутствует.



	BRAM	DSP	FF	LUT	Bits P0	Bits P1	Bits P2	Banks/Depth	Words	W*Bits*Bank
lab1_1	0	1	0	16						
I/O Ports(4)					32					
Instances(0)	0	0	0	0						
Memories(0)	0		0	0	0			0	0	0
Expressions(1)	0	0	0	16	9	9	0			
Registers(0)			0		0					
Channels(0)	0		0	0	0			0	0	0
Multiplexers(0)	0		0	0	0			0		
DSP(1)		1								

Данные соответствуют приведенному выше отчету.



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

Интерфейс

Interface

Summary

RTL Ports	Dir	Bits	Protocol	Source Object	C Type
a	in	8	ap_none	a	scalar
b	in	8	ap_none	b	scalar
c	in	8	ap_none	c	scalar
d	in	8	ap_none	d	scalar
ap_return	out	32	ap_ctrl_none	lab1_1	return value

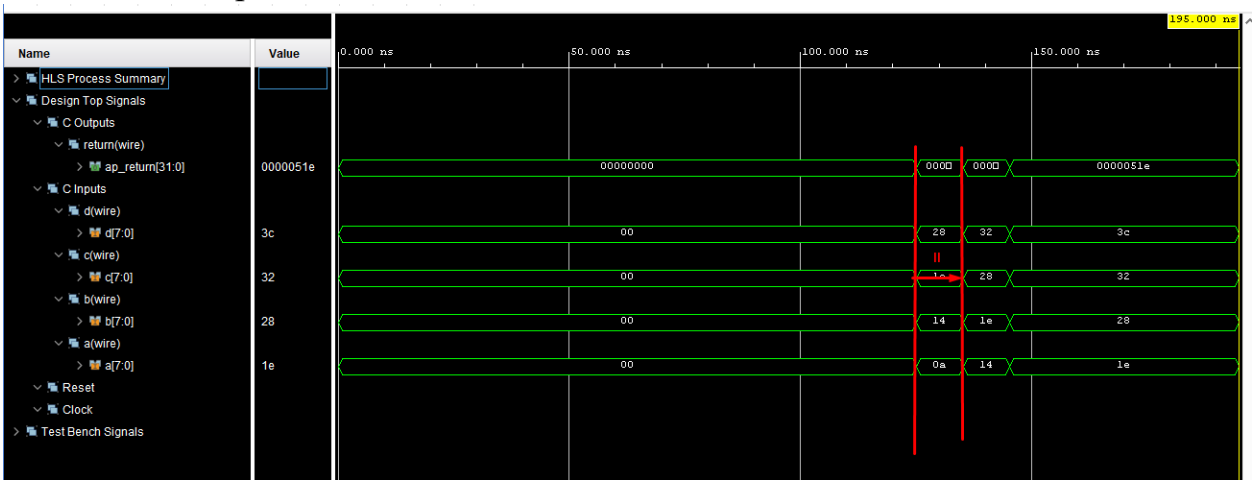
По сравнению с предыдущими решениями в данном решении отсутствуют многие RTL порты. Конструкция имеет 5 портов данных.

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

4. C/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.