

Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

Отчет по лабораторной работе №3_1
Курс: «Проектирование реконфигурируемых гибридных
вычислительных систем»
Тема: «Port-level IO protocols»

Выполнил студент гр. 3540901/81501

Селиверстов С.А.

(подпись)

Руководитель

Антонов А.П.

(подпись)

“ ” _____ 2019 г.

Санкт – Петербург
2019

Оглавление

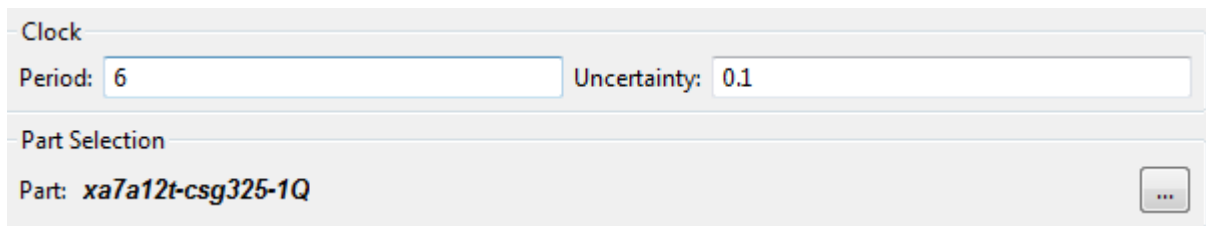
1. Задание	3
2. Первое решение.....	5
2.1 Исходный код функции и программы	5
2.2 Моделирование	6
2.3 Синтез.....	6
2.4 C RTL моделирование	8
3. Второе решение.....	10
3.1 Моделирование	10
3.2 Синтез.....	10
3.3 C RTL моделирование	13
4. Третье решение	14
4.1 Моделирование	14
4.2 Синтез.....	14
4.3 C RTL моделирование	17
5. Выводы.....	17

1. Задание

- Создать проект lab2_1
- Подключить файл lab2_1.c (папка source)
- Подключить тест lab2_1_test.c (папка source)
- Микросхема: ха7a12tcsg325-1q
- 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

2. Первое решение



Clock

Period: Uncertainty:

Part Selection

Part: **xa7a12t-csg325-1Q** ...

Рисунок 2.1 Параметры первого решения

2.1 Исходный код функции и программы

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

#include <stdio.h>

bool areEqual(int arr1[], int arr2[])
{
    // Linearly compare elements
    for (int i = 0; i < 3; i++)
        if (arr1[i] != arr2[i])
            return false;

    // If all elements were same.
    return true;
}

int main() {

    char inA, inB, inC;

    int inArr[3] = {10,20,30};
    int outArr[3];
    int pass = 1;
    int i;

    //150 250 350
    //270 470 670
    //390 690 990
    int refOut[3][3] = { {150,250,350}, {270,470,670}, {390,690,990} };

    inA = 10;
    inB = 20;
    inC = 30;

    if (pass)
    {
        fprintf(stdout, "-----Pass!-----\n");
        return 0;
    }
    else
    {
        fprintf(stderr, "-----Fail!-----\n");
        return 1;
    }
    return 0;
}
```

2.2 Моделирование

```
1 [INFO: [SIM 2] ***** CSIM start *****
2 INFO: [SIM 4] CSIM will launch GCC as the compiler.
3   Compiling(apcc) ../../../../source/lab3_1_test.c in debug mode
4 INFO: [HLS 200-10] Running 'D:/Program_Files/Xilinx/Vivado/2019.1/bin/unwrapped/win64.o/apcc.exe'
5 INFO: [HLS 200-10] For user 'P>PuPsPSPePr' on host 'P>PuPsPSPePr-PiPe' (windows NT_amd64 version 6.1)
6 INFO: [HLS 200-10] In directory 'D:/Program_Files/projects/hls/lab3_z1/lab2_1/solution1/csim/build'
7 INFO: [APCC 202-3] Tmp directory is apcc_db
8 INFO: [APCC 202-1] APCC is done.
9   Compiling(apcc) ../../../../source/lab3_1.c in debug mode
10 INFO: [HLS 200-10] Running 'D:/Program_Files/Xilinx/Vivado/2019.1/bin/unwrapped/win64.o/apcc.exe'
11 INFO: [HLS 200-10] For user 'P>PuPsPSPePr' on host 'P>PuPsPSPePr-PiPe' (windows NT_amd64 version 6.1)
12 INFO: [HLS 200-10] In directory 'D:/Program_Files/projects/hls/lab3_z1/lab2_1/solution1/csim/build'
13 INFO: [APCC 202-3] Tmp directory is apcc_db
14 INFO: [APCC 202-1] APCC is done.
15   Generating csim.exe
16   10*20+30+40=270
17   20*30+40+50=690
18   30*40+50+60=1310
19 -----Pass!-----
20 INFO: [SIM 1] CSim done with 0 errors.
21 INFO: [SIM 3] ***** CSIM finish *****
22
```

Рисунок 2.2 Результат успешного моделирования

Моделирование второго решения прошло успешно.

2.3 Синтез

Данные о проекте

Performance Estimates			
Timing (ns)			
Summary			
Clock	Target	Estimated	Uncertainty
ap_clk	6.00	3.820	0.10

Рисунок 2.3 Performance estimates – summary

Здесь можно увидеть, что достигнутая задержка равна $3.820 + 0.1$, что укладывается в заданные нами требования к тактовой частоте.

Использование ресурсов

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

Рисунок 2.4 Utilization estimates – summary

Данный проект займет на микросхеме 1 DSP блок (в котором будут использованы и сумматоры и умножитель), 12 регистров для хранения чисел, и 37 LUT.

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

Рисунок 2.5a Performance Profile

	BRAM	DSP	FF	LUT	Bits P0	Bits P1	Bits P2	Banks/Depth	Words	W*Bits*Banks
lab1_1	0	1	12	37						
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(2)			12		12					
Channels(0)	0		0	0	0		0		0	0
Multiplexers(1)	0		0	21	1		0			
DSP(1)		1								

Рисунок 2.6b Performance Profile

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

Рисунок 2.7 Interface estimates – summary

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. 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`.

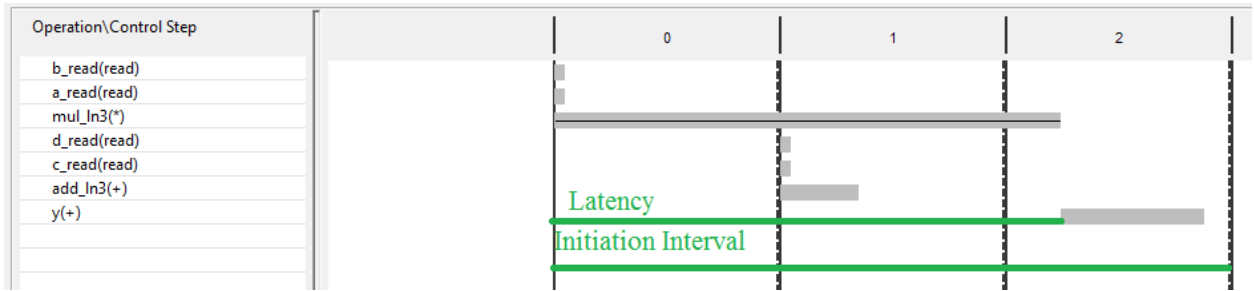


Рисунок 2.8 Schedule viewer

На рисунке 2.7 видно, что задержка получения выходного значения составляет 2 такта с момента старта, а задержка после старта до готовности приема новых данных – 3. Покажем эти интервалы на временной диаграмме. Здесь мы видим весь процесс получения результата. На первом такте происходит считывание операторов A и B, а также начинается их умножение. На втором такте начинается считывание C и D, а также их сложение. Таким образом суммарная задержка $latency = 2$, а со следующего 3-го такта можно подавать следующие данные ($\Pi = 3$).

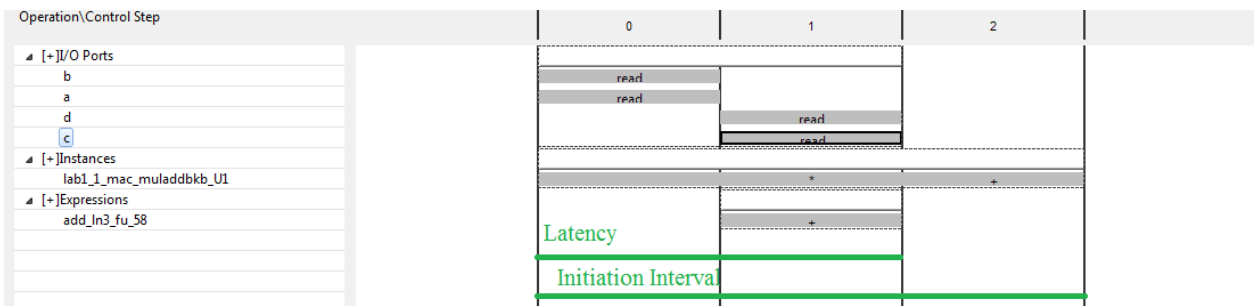


Рисунок 2.9 Resource viewer

2.4 C|RTL моделирование

Cosimulation Report for 'lab1_1'

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

Export the report(.html) using the [Export Wizard](#)

Рисунок 2.10 Отчет о моделировании

При совместном моделировании, программа отобразила те же самые, ожидаемые нами значения Latency и Π .

Покажем временную диаграмму совместного моделирования с отмеченными на ней Latency и Π :

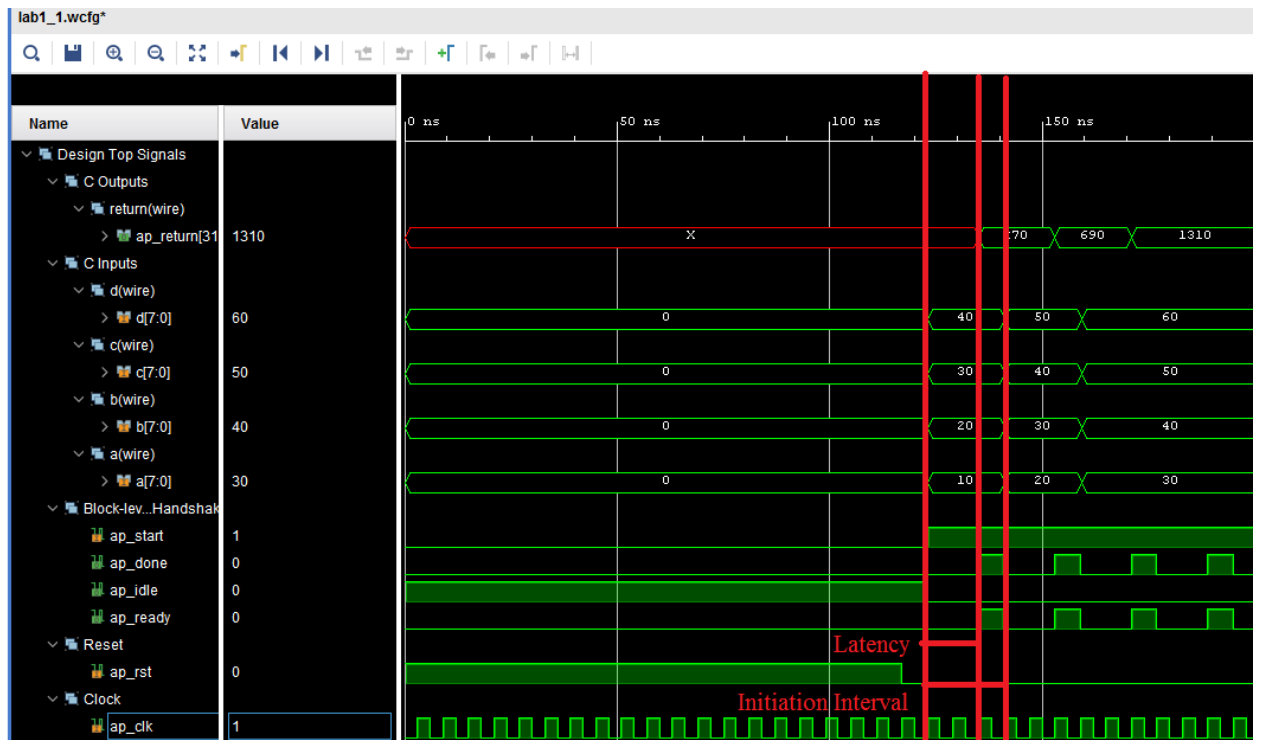


Рисунок 2.11 Временная диаграмма

3. Второе решение

```

1 int lab1_1( char a, char b, char c, char d) {
2     #pragma HLS INTERFACE ap_ctrl_chain port=return
3     int y;
4     y = a*b+c+d;
5     return y;
6 }

```

lab1_1

- # HLS INTERFACE ap_ctrl_chain port=return
- a
- b
- c
- d

Рисунок 3.1 Листинг кода и директивы

Во-втором задании добавлен протокол `ap_ctrl_chain`. Данный протокол является протоколом ввода-вывода на уровне блоков для цепочки управления. Этот протокол ввода / вывода в основном используется для объединения конвейерных блоков.

3.1 Моделирование

```

1 INFO: [SIM 2] ***** CSIM start *****
2 INFO: [SIM 4] CSIM will launch GCC as the compiler.
3   Compiling(apcc) ../../../../source/lab3_1_test.c in debug mode
4 INFO: [HLS 200-10] Running 'D:/Program_Files/Xilinx/Vivado/2019.1/bin/unwrapped/win64.o/apcc.exe'
5 INFO: [HLS 200-10] For user 'P>PμPsPSPēPr' on host 'P>PμPsPSPēPr-PiPe' (Windows NT_amd64 version 6.1) on
6 INFO: [HLS 200-10] In directory 'D:/Program_Files/projects/hls/lab3_z1/lab2_1/solution2/csim/build'
7 INFO: [APCC 202-3] Tmp directory is apcc_db
8 INFO: [APCC 202-1] APCC is done.
9   Compiling(apcc) ../../../../source/lab3_1.c in debug mode
10 INFO: [HLS 200-10] Running 'D:/Program_Files/Xilinx/Vivado/2019.1/bin/unwrapped/win64.o/apcc.exe'
11 INFO: [HLS 200-10] For user 'P>PμPsPSPēPr' on host 'P>PμPsPSPēPr-PiPe' (Windows NT_amd64 version 6.1) on
12 INFO: [HLS 200-10] In directory 'D:/Program_Files/projects/hls/lab3_z1/lab2_1/solution2/csim/build'
13 INFO: [APCC 202-3] Tmp directory is apcc_db
14 INFO: [APCC 202-1] APCC is done.
15   Generating csim.exe
16   10*20+30+40=270
17   20*30+40+50=690
18   30*40+50+60=1310
19 -----Pass!-----
20 INFO: [SIM 1] CSim done with 0 errors.
21 INFO: [SIM 3] ***** CSIM finish *****
22

```

Рисунок 3.2 Результат успешного моделирования

Моделирование второго решения также прошло успешно.

3.2 Синтез

Timing (ns)

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	6.00	3.820	0.10

Latency (clock cycles)

Summary

Latency		Interval		
min	max	min	max	Type
2	2	2	2	none

Рисунок 3.3 Performance estimates – summary

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

Рисунок 3.4 Utilization estimates – summary

Данный проект будет занимать на микросхеме: 1 DSP блок, где будут задействованы сумматор и умножитель, 45 регистров для хранения и считывания данных (чисел), 57LUT. По сравнению с предыдущим решением выросло на 33 количество используемых регистров и на 20 количество LUT.

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

Рисунок 3.5a Performance Profile

	BRAM	DSP	FF	LUT	Bits P0	Bits P1	Bits P2	Banks/Depth	Words	W*Bits*Banks
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	0
> Expressions(2)	0	0	0	18	10	10	0			
> Registers(4)			45	45						
> Channels(0)	0	0	0	0				0	0	0
> Multiplexers(3)	0	0	39	34				0		
> DSP(1)		1								

Рисунок 3.5b Resource Profile

Здесь мы также видим отличия, согласно общему отчету о затраченных ресурсах.

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

Рисунок 3.6 Interface estimates – summary

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).

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.

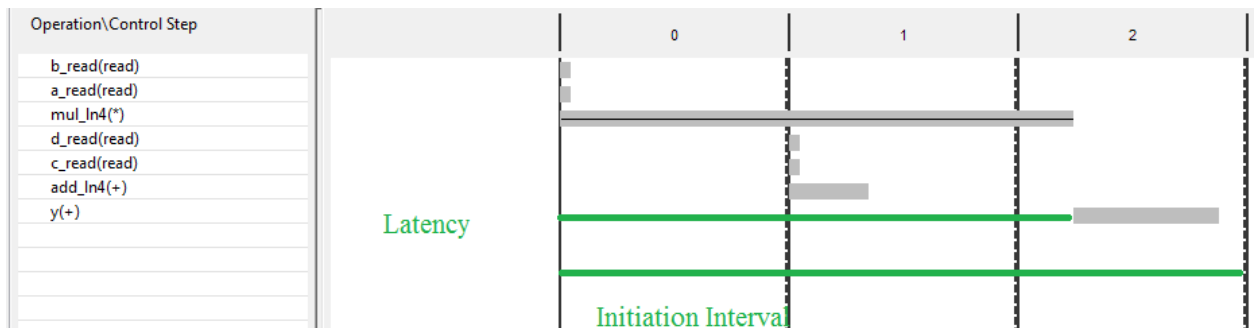


Рисунок 3.7 Schedule viewer

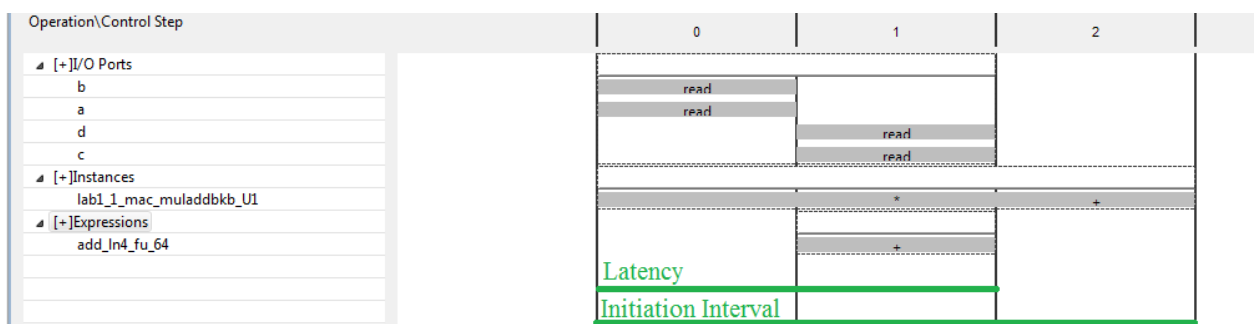


Рисунок 3.8 Resource viewer

3.3 C|RTL моделирование

При осуществлении совместного моделирования программа показала ожидаемые результаты Latency

Cosimulation Report for 'lab1_1'

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

Export the report(.html) using the [Export Wizard](#)

Рисунок 3.9 Отчет о моделировании

Покажем временную диаграмму совместного моделирования



Рисунок 3.10 Временная диаграмма

4. Третье решение

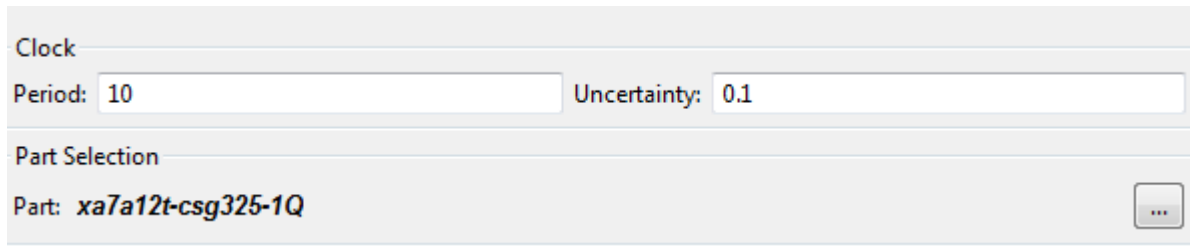


Рисунок 4.1 Настройки третьего решения

```
1 int lab1_1( char a, char b, char c, char d) {  
2     #pragma HLS INTERFACE ap_ctrl_none port=return  
3     int y;  
4     y = a*b+c+d;  
5     return y;  
6 }
```

lab1_1

- a
- b
- c
- d

HLS INTERFACE ap_ctrl_none port=return

Рисунок 4.2 Листинг кода и директива настройки.

4.1 Моделирование

```
1 INFO: [SIM 2] ***** CSIM start *****  
2 INFO: [SIM 4] CSIM will launch GCC as the compiler.  
3   Compiling(apcc) ../../../../source/lab3_1_test.c in debug mode  
4 INFO: [HLS 200-10] Running 'D:/Program_Files/Xilinx/Vivado/2019.1/bin/unwrapped/win64.o/apcc.exe'  
5 INFO: [HLS 200-10] For user 'P>PµPSPSPPr' on host 'P>PµPSPSPPr-PiPe' (Windows NT_amd64 version 6.1) or  
6 INFO: [HLS 200-10] In directory 'D:/Program_Files/projects/hls/lab3_z1/lab2_1/solution3/csim/build'  
7 INFO: [APCC 202-3] Tmp directory is apcc_db  
8 INFO: [APCC 202-1] APCC is done.  
9   Compiling(apcc) ../../../../source/lab3_1.c in debug mode  
10 INFO: [HLS 200-10] Running 'D:/Program_Files/Xilinx/Vivado/2019.1/bin/unwrapped/win64.o/apcc.exe'  
11 INFO: [HLS 200-10] For user 'P>PµPSPSPPr' on host 'P>PµPSPSPPr-PiPe' (Windows NT_amd64 version 6.1) or  
12 INFO: [HLS 200-10] In directory 'D:/Program_Files/projects/hls/lab3_z1/lab2_1/solution3/csim/build'  
13 INFO: [APCC 202-3] Tmp directory is apcc_db  
14 INFO: [APCC 202-1] APCC is done.  
15   Generating csim.exe  
16   10*20+30+40=270  
17   20*30+40+50=690  
18   30*40+50+60=1310  
19 -----Pass!-----  
20 INFO: [SIM 1] CSim done with 0 errors.  
21 INFO: [SIM 3] ***** CSIM finish *****
```

Рисунок 4.3 Результат успешного моделирования

4.2 Синтез

Performance Estimates

[-] Timing (ns)

[-] Summary

Clock	Target	Estimated	Uncertainty
ap_clk	10.00	7.180	0.10

[-] Latency (clock cycles)

[-] Summary

Latency		Interval		
min	max	min	max	Type
0	0	0	0	none

Рисунок 4.4 Performance estimates – summary

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

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

Рисунок 4.5 Utilization estimates – summary

Данный проект будет занимать на микросхеме: 1 DSP блок, где будут задействованы сумматор и умножитель. Количество используемых LUT в выражениях сократилось на 2 LUT и составило 16 LUT.

По сравнению с предыдущими решениями использование регистров и мультиплексоров полностью отсутствует.

Performance Profile					
	Pipelined	Latency	Iteration Latency	Initiation Interval	Trip count
lab1_1	-	0	-	1	-

Рисунок 4.6a Performance Profile

Resource Profile										
	BRAM	DSP	FF	LUT	Bits P0	Bits P1	Bits P2	Banks/Depth	Words	W*Bits*Banks
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								

Рисунок 4.6b. Resource Profile

Здесь мы также видим отличия, согласно общему отчету о затраченных ресурсах.

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

Рисунок 4.7 Interface estimates – summary

По сравнению с предыдущими решениями в данном решении отсутствуют `ap_start`, `ap_done`, `ap_idle`, `ap_ready`, `ap_clk`, `ap_rst`, `ap_continue` RTL порты.

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

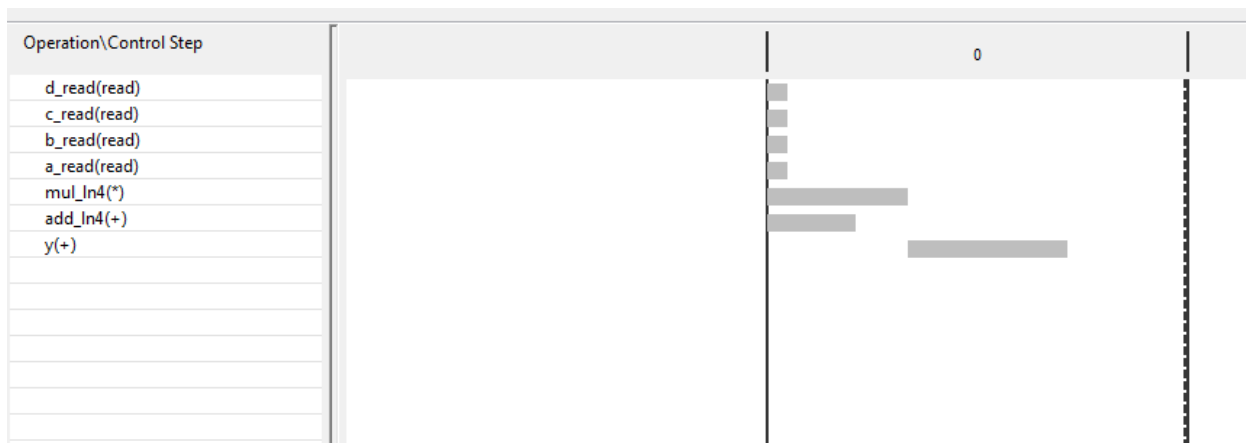


Рисунок 4.8 Schedule viewer

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

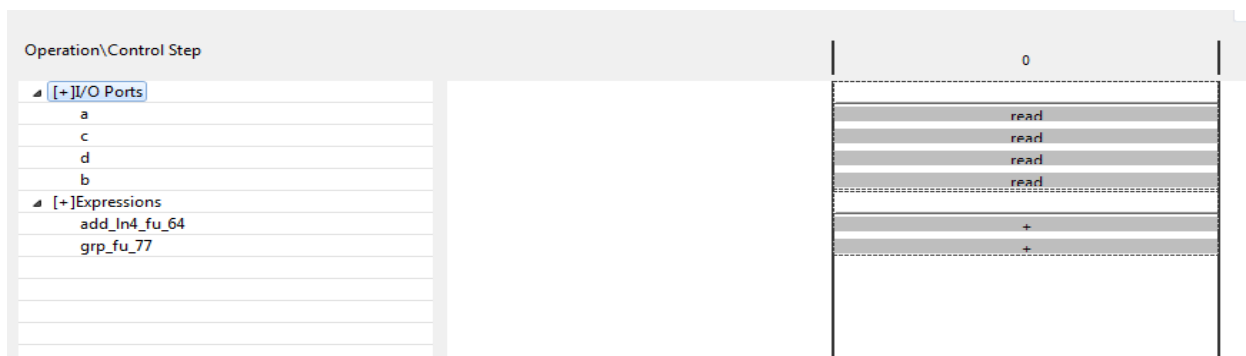


Рисунок 4.9 Resource viewer

4.3 C|RTL моделирование

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	0	0	0	0	0	0

Рисунок 4.10 Отчет о моделировании

Покажем временную диаграмму совместного моделирования

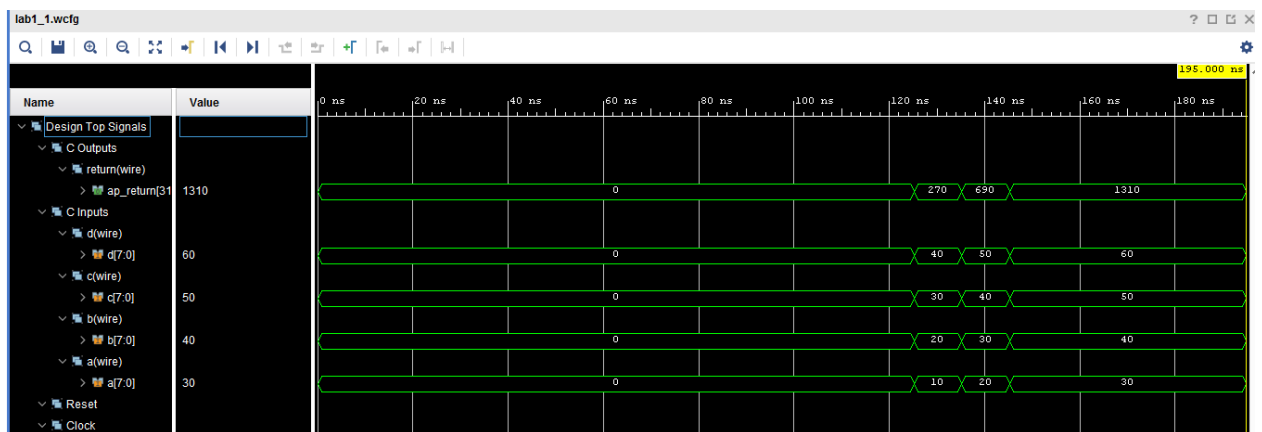


Рисунок 4.12 Временная диаграмма

5. 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_continue` имеет значение 1. Режим `ap_ctrl_none` реализует моделирование без какого-либо блочного протокола ввода-вывода.

В данной работе рассмотрены следующие директивы блочного протокола.

1. `ap_ctrl_none`: нет протокола управления вводом / выводом на уровне блоков,
2. `ap_ctrl_hs`: стандартный протокол управления вводом-выводом на уровне блоков,
3. `ap_ctrl_chain`: протокол ввода-вывода для управления цепочками конвейерных блоков.