Санкт-Петербургский Политехнический Университет Петра Великого
Институт Компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

Лабораторная работа 4

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

Санкт-Петербургский Политехнический Университет Петра Великого
Институт Компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

Лабораторная работа 4

Предмет: Проектирование реконфигурируемых гибридных вычислительных систем

Тема: Port-Level IO Protocols

Задание 1

Студент: Белоглазов К.И.
Гр. 3540901/81501
Преподаватель: Антонов А.П.

Санкт-Петербург
2019

Оглавление

Задание

- Создать проект lab4_1
- Подключить файл lab4_1.c (папка source)
- Создать тест lab4_1_test.c на основе теста Подключить тест lab1_1_test.c
- 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 протоколы
        - a: ap_hs
        - b: ap_ask
        - *c: ap_hs
        - *d: ap_vld
        - *p_y: ap_ask
    - 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
- Выводы
    - Объяснить отличие протоколов port_level

Ход работы

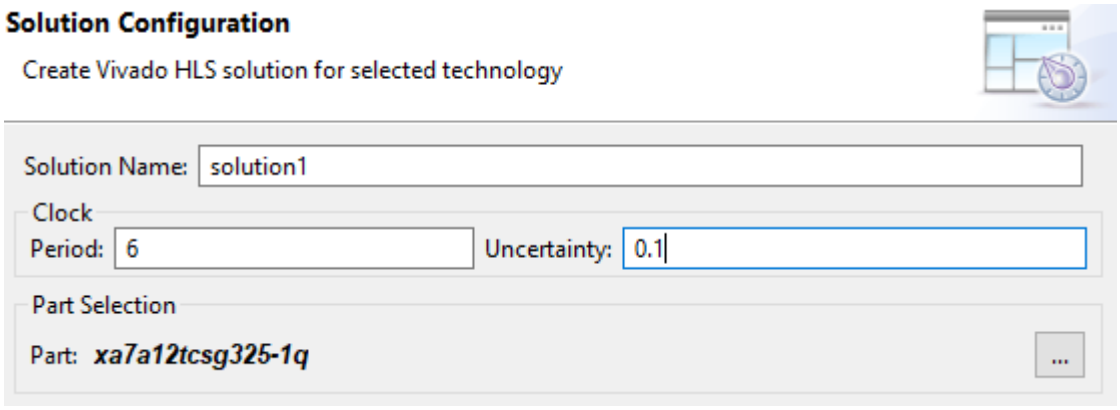
Решение 1

Создание и конфигурирование решения.

Создание тест – файла

```
#include <stdio.h>

int main()
{
	int res;
	// For adders
	int refOut[3] = {230, 640, 1250};
	int pOut[3] = {240, 650, 1260};
	int pass;
	int i;

	int inA = 10;
	int inB = 20;
	 int inC = 30;
	int inD = 40;
	int inP = 1;

	// Call the adder for 5 transactions
	for (i=0; i<3; i++)
	{
		res = lab4_1(inA, inB, &inC, &inD, &inP);

		fprintf(stdout, "  %d*%d+%d=%d \n", inA, inB, inC, res);
	fprintf(stdout, "  %d*%d+%d=%d \n", inA, inB, inD, inP);


	  // Test the output against expected results
		if (res == refOut[i] & inP == pOut[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;
```

```
        }
}
```



Моделирование

```
NFO: [HLS 200-10] For user 'loris' on host 'laptop-34slcvbc' (Windows NT_amd64 versi
NFO: [HLS 200-10] In directory 'D:/Antonov/lab4_z1/lab4/solution1/csim/build'
NFO: [APCC 202-3] Tmp directory is apcc_db
NFO: [APCC 202-1] APCC is done.
  Generating csim.exe
 10*20+30=230
 10*20+40=240
 20*30+40=640
 20*30+50=650
 30*40+50=1250
 30*40+60=1260
---------Pass!------------
NFO: [SIM 211-1] CSim done with 0 errors.
NFO: [SIM 211-3] *************** CSIM finish ***************
inished C simulation.
```

Производительность

Target device: xa7a12t-csg325-1Q

Performance Estimates

Timing

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	6.00 ns	5.690 ns	0.10 ns

Latency

Summary

Latency (cycles)		Latency (absolute)		Interval (cycles)		
min	max	min	max	min	max	Type
3	3	18.000 ns	18.000 ns	3	3	none

Detail

Instance

На изображении видно, что полученная величина задержки укладывается в целевое значение.

Использование ресурсов

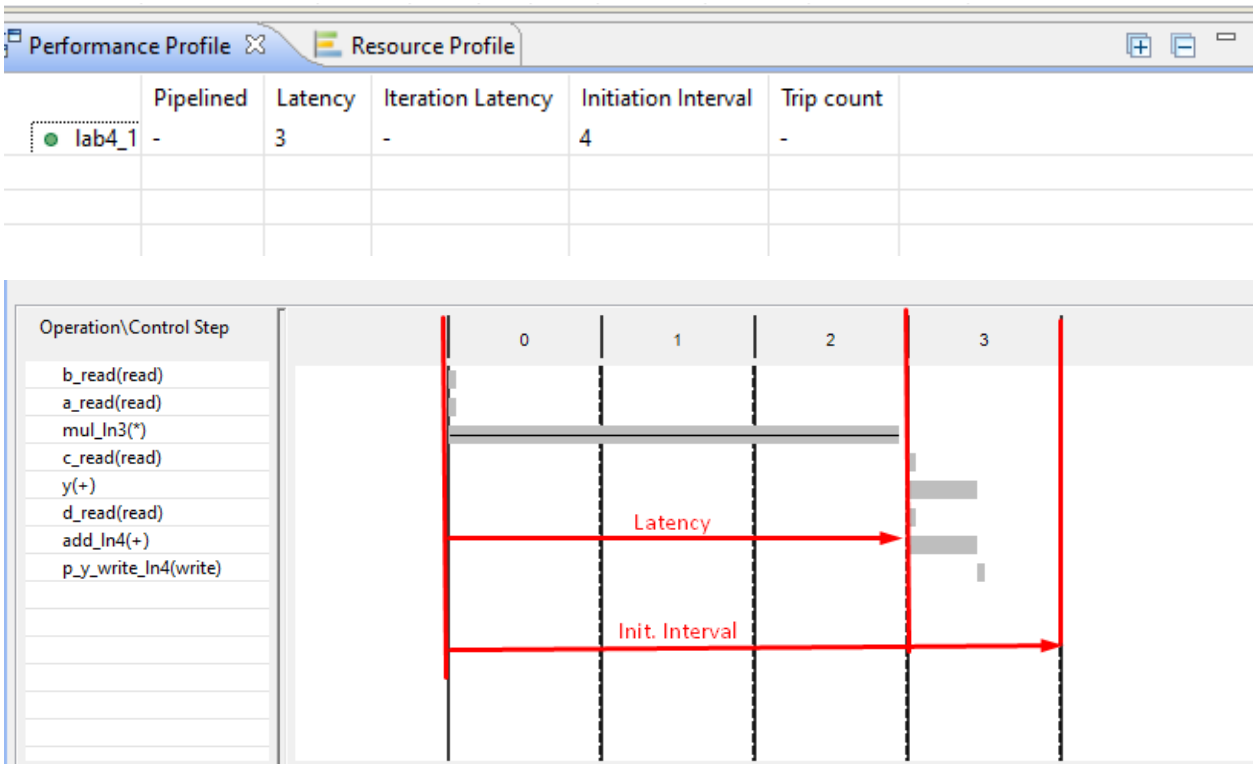
Utilization Estimates

Summary

Name	BRAM_18K	DSP48E	FF	LUT	URAM
DSP	-	-	-	-	-
Expression	-	-	0	78	-
FIFO	-	-	-	-	-
Instance	-	3	166	49	-
Memory	-	-	-	-	-
Multiplexer	-	-	-	27	-
Register	-	-	36	-	-
Total	0	3	202	154	0
Available	40	40	16000	8000	0
Utilization (%)	0	7	1	1	0

Detail

Данный проект будет занимать на микросхеме:
4 DSP блока, где будут задействованы сумматор и умножитель.
202 регистра для хранения и считывания данных (чисел).
154 LUT.



На изображениях, приведенных выше видно, что задержка до получения результата составляет 3 такта, а интервал инициализации – 4 такта.

Интерфейс

Interface

Summary

RTL Ports	Dir	Bits	Protocol	Source Object	C Type
ap_clk	in	1	ap_ctrl_hs	lab4_1	return value
ap_rst	in	1	ap_ctrl_hs	lab4_1	return value
ap_start	in	1	ap_ctrl_hs	lab4_1	return value
ap_done	out	1	ap_ctrl_hs	lab4_1	return value
ap_idle	out	1	ap_ctrl_hs	lab4_1	return value
ap_ready	out	1	ap_ctrl_hs	lab4_1	return value
ap_return	out	32	ap_ctrl_hs	lab4_1	return value
a	in	32	ap_none	a	scalar
b	in	32	ap_none	b	scalar
c	in	32	ap_none	c	pointer
d	in	32	ap_none	d	pointer
p_y	out	32	ap_vld	p_y	pointer
p_y_ap_vld	out	1	ap_vld	p_y	pointer

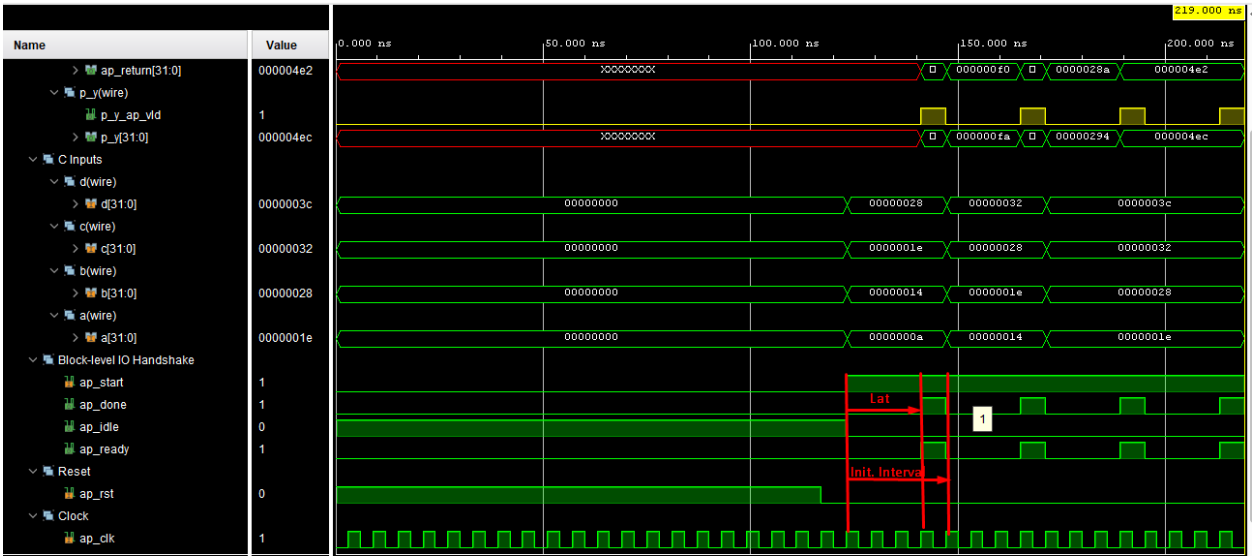
Конструкция имеет 6 портов данных.

Входные порты: a, b, c, d являются 32-битными входами и имеют входы / выходы, протокол ap_none.

Конструкция имеет 32-битный выходной порт для возврата функции p_y, p_y_ap_vld – протокол по умолчанию для портов выхода.

Управляющие сигналы ap_clk, ap_rst и ap_ * автоматически добавляются в каждый дизайн по умолчанию. Ap_start, ap_done, ap_idle и ap_ready являются сигналами верхнего уровня, используемыми в качестве сигналов подтверждения связи, чтобы указать, когда проект способен принять следующую команду вычисления (ap_ready), когда начинается следующее вычисление (ap_start) и когда вычисление завершено (ap_done).

C/RTL моделирование. Временная диаграмма



На временной диаграмме отображена задержка и интервал инициализации.

Решение 2

Добавление директив

```
int lab4_1( int a, int b, int *c, int *d, int *p_y) {
#pragma HLS INTERFACE ap_ack port=p_y
#pragma HLS INTERFACE ap_vld port=d
#pragma HLS INTERFACE ap_hs port=c
#pragma HLS INTERFACE ap_ack port=b
#pragma HLS INTERFACE ap_hs port=a
int y;
y = a*b + *c;
*p_y = a*b + *d;
return y;
}
```

```
lab4_1
  a
  # HLS INTERFACE ap_hs port=a
  b
  # HLS INTERFACE ap_ack port=b
  c
  # HLS INTERFACE ap_hs port=c
  d
  # HLS INTERFACE ap_vld port=d
  p_y
  # HLS INTERFACE ap_ack port=p_y
```

Производительность

Performance Estimates

Timing

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	6.00 ns	5.690 ns	0.10 ns

Latency

Summary

Latency (cycles)		Latency (absolute)		Interval (cycles)		
min	max	min	max	min	max	Type
3	3	18.000 ns	18.000 ns	3	3	none

Detail

Instance

По сравнению с предыдущим решением значение полученной задержки не изменилось.

Использование ресурсов

Utilization Estimates

Summary

Name	BRAM_18K	DSP48E	FF	LUT	URAM
DSP	-	-	-	-	-
Expression	-	-	0	80	-
FIFO	-	-	-	-	-
Instance	-	3	166	49	-
Memory	-	-	-	-	-
Multiplexer	-	-	-	90	-
Register	-	-	101	-	-
Total	0	3	267	219	0
Available	40	40	16000	8000	0
Utilization (%)	0	7	1	2	0

Detail

	BRAM	DSP	FF	LUT	Bits P0	Bits P1	Bits P2	Banks/Depth	Words	W*Bits*Banks
lab4_1	0	3	267	219						
I/O Ports(5)					160					
Instances(1)	0	3	166	49						
Memories(0)	0		0	0	0			0	0	0
Expressions(3)	0	0	0	80	65	65	0			
Registers(5)			101		101					
Channels(0)	0		0	0	0			0	0	0
Multiplexers(8)	0		0	90	39			0		
DSP(1)		0								

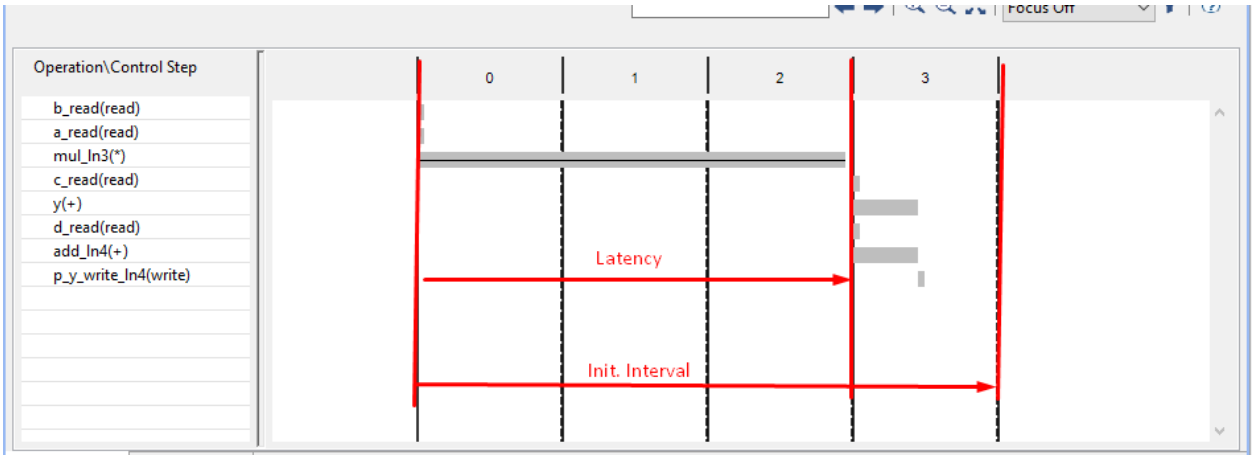
Данный проект будет занимать на микросхеме:
4 DSP блока, где будут задействованы сумматор и умножитель.

268 регистров для хранения и считывания данных (чисел).
160 LUT.

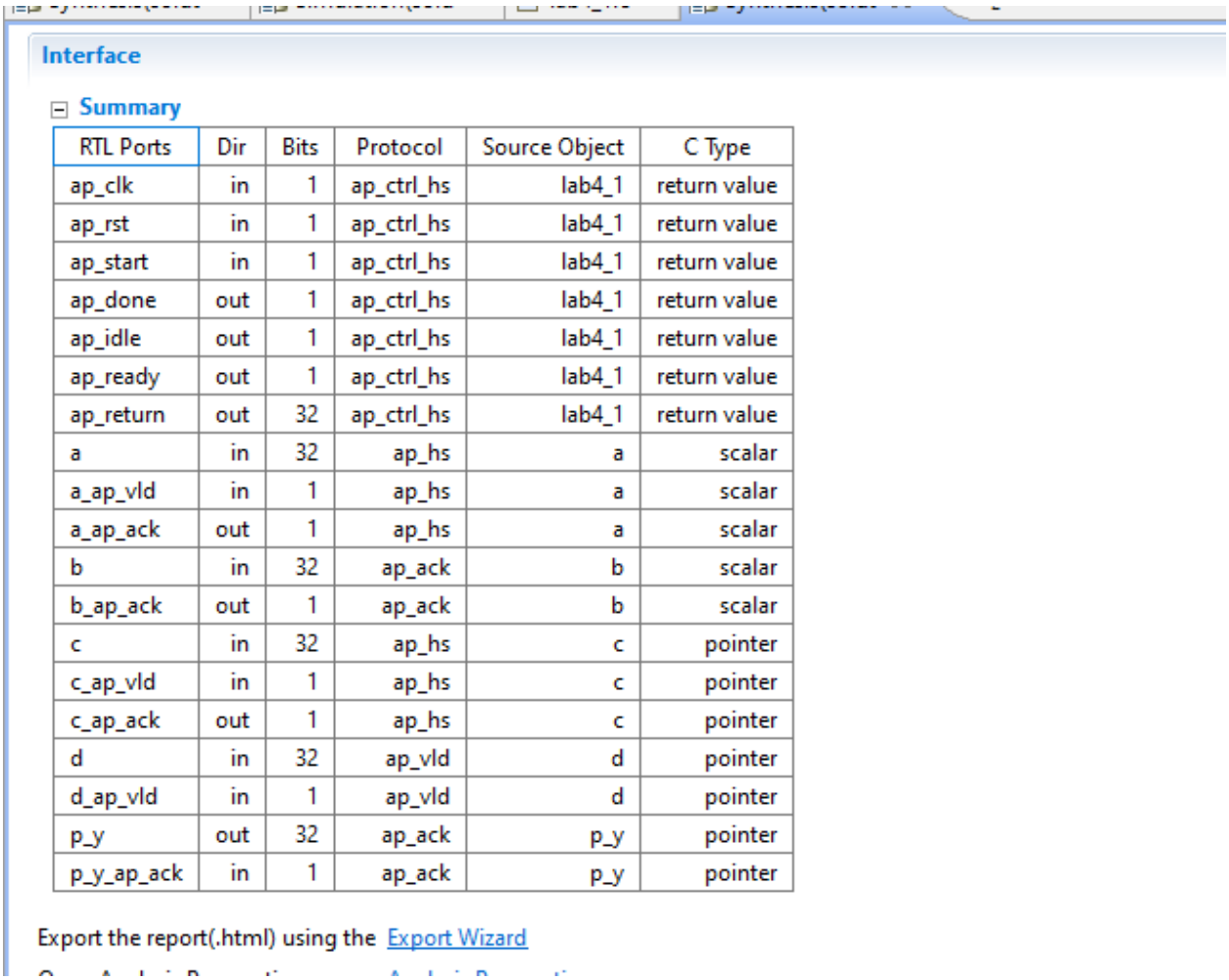
Performance Profile | Resource Profile

	Pipelined	Latency	Iteration Latency	Initiation Interval	Trip count
lab4_1	-	3	-	4	-

На изображении выше видно, что задержка получения результата составляет 3 такта, а интервал инициализации – 4.



Интерфейс



Interface

Summary

RTL Ports	Dir	Bits	Protocol	Source Object	C Type
ap_clk	in	1	ap_ctrl_hs	lab4_1	return value
ap_rst	in	1	ap_ctrl_hs	lab4_1	return value
ap_start	in	1	ap_ctrl_hs	lab4_1	return value
ap_done	out	1	ap_ctrl_hs	lab4_1	return value
ap_idle	out	1	ap_ctrl_hs	lab4_1	return value
ap_ready	out	1	ap_ctrl_hs	lab4_1	return value
ap_return	out	32	ap_ctrl_hs	lab4_1	return value
a	in	32	ap_hs	a	scalar
a_ap_vld	in	1	ap_hs	a	scalar
a_ap_ack	out	1	ap_hs	a	scalar
b	in	32	ap_ack	b	scalar
b_ap_ack	out	1	ap_ack	b	scalar
c	in	32	ap_hs	c	pointer
c_ap_vld	in	1	ap_hs	c	pointer
c_ap_ack	out	1	ap_hs	c	pointer
d	in	32	ap_vld	d	pointer
d_ap_vld	in	1	ap_vld	d	pointer
p_y	out	32	ap_ack	p_y	pointer
p_y_ap_ack	in	1	ap_ack	p_y	pointer

Export the report(.html) using the Export Wizard

По сравнению с предыдущим решением ap_none изменился на заданные протоколы. Для портов входа и выхода используются следующие протоколы: ap_hs, ap_vld, ap_ack.

Протоколы ввода-вывода на уровне порта ap_hs обеспечивает наибольшую гибкость в процессе разработки.

Протокол ввода-вывода уровня порта ap_hs предоставляет следующие сигналы:

- Порт данных

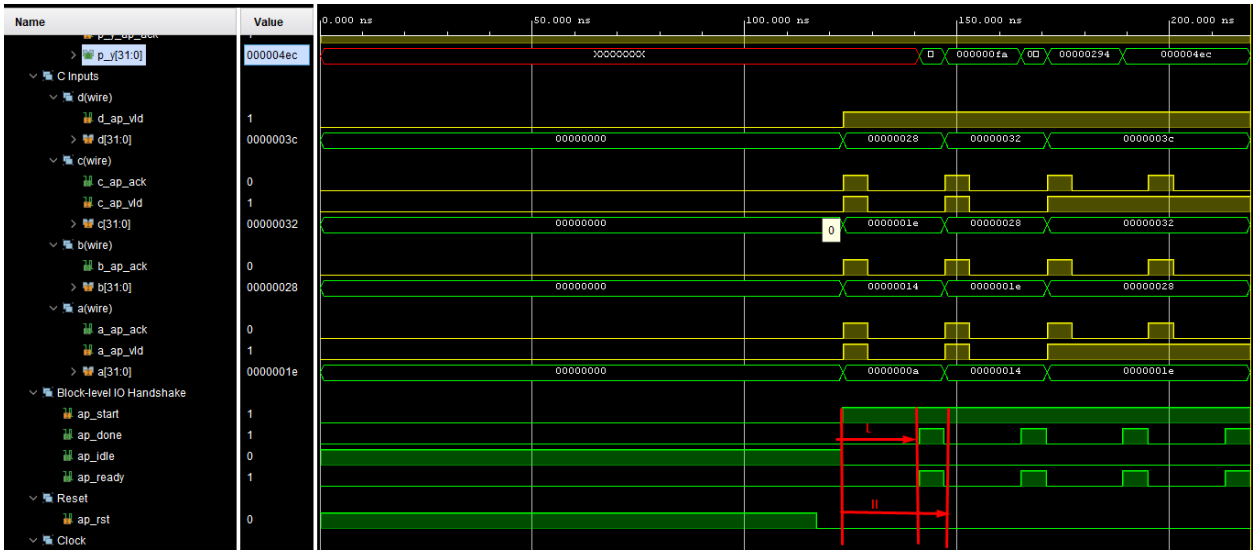
- Сигнал указания момента использования данных (ack)

- Действительный сигнал для указания, когда данные считываются (vld)

Протокол ввода-вывода уровня порта ap_none является самым простым типом интерфейса и не имеет никаких других сигналов, связанных с ним. Ни входные, ни выходные сигналы данных не имеют связанных портов управления, которые указывают, когда данные считываются или записываются. Единственными портами в конструкции RTL являются порты, указанные в исходном коде. Интерфейс ap_none не требует дополнительных аппаратных издержек.

C/RTL моделирование

На временной диаграмме отображена задержка и интервал инициализации.



Вывод

Протоколы ввода-вывода на уровне порта ap_hs обеспечивает наибольшую гибкость в процессе разработки.

Протокол ввода-вывода уровня порта ap_hs предоставляет следующие сигналы:

- Порт данных

- Сигнал указания момента использования данных (ack)

- Действительный сигнал для указания, когда данные считываются (vld)

Протокол ввода-вывода уровня порта ap_none является самым простым типом интерфейса и не имеет никаких других сигналов, связанных с ним. Ни входные, ни выходные сигналы данных не имеют связанных портов управления, которые указывают, когда данные считываются или записываются. Единственными портами в конструкции RTL являются порты, указанные в исходном коде. Интерфейс ap_none не требует дополнительных аппаратных издержек.