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

Лабораторная №11

Предмет: Проектирование реконфигурируемых гибридных вычислительных систем

Тема: Latency

Задание 4

Студенты:
Соболь В.
Темнова А.С.
Группа: 13541/3
Преподаватель:
Антонов А.П.

Санкт-Петербург
2019

Содержание

1. Задание

1. Создать проект lab11_4
2. Микросхема: xa7a12tcsg325-1q
3. Создать функцию по образцу (иерархия функций)

```
void top (a[4],b[4],c[4],d[4]...) {

  ...
  Add: for (i=3;i>=0;i--)  { --------
     if  (d[i])
       a[i] = b[i] + c[i];
  }                          --------

  Sub: for (i=3;i>=0;i--)  { --------
     if (!d[i])
      a[i] = b[i] - c[i];
  }                          --------
  ...
}
```

4. Создать тест lab11_4_test.c для проверки функции. Осуществить моделирование (с выводом результатов в консоль)
5. Исследование:
6. Solution_1a
   - задать: clock period 10; clock_uncertainty 0.1
   - установить реализацию ПО УМОЛЧАНИЮ
   - осуществить синтез для:
     - привести в отчете:
       * performance estimates=>summary (timing, latency)
       * utilization estimates=>summary
       * performance Profile
       * Resource profile
       * 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
   - Выполнить cosimulation и привести временную диаграмму
7. Solution_2a

- задать: clock period 10; clock_uncertainty 0.1
- установить реализацию LOOP_MERGE
- осуществить синтез
  - привести в отчете:
    - performance estimates=>summary (timing, latency)
    - utilization estimates=>summary
    - performance Profile
    - Resource profile
    - 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
- Выполнить cosimulation и привести временную диаграмму

8. Сравнить два решения (solution_1a и solution_2a) и сделать выводы: объяснить (посчитать) число циклов Latency, II...

2. Исходный код

Ниже приведен исходный код устройства и теста.

```
1  #include "lab11_4.h"
2
3  void lab11_4(int a[N],int b[N],int c[N],int d[N])
4  {
5
6    int i;
7
8    Add:for(i = N - 1; i >= 0;i--)
9    {
10     if(d[i])
11     {
12       a[i] = b[i] + c[i];
13     }
14   }
15
16   Sub:for(i = N - 1; i >= 0; i--)
17   {
18     if(!d[i])
19     {
20       a[i] = b[i] - c[i];
21     }
22   }
23 }
```

Рис. 2.1. Исходный код устройства

```
1 #define N 4
```

Рис. 2.2. Заголовочный файл

```
1  #include "lab11_4.h"
2  #include <stdio.h>
3
4  int main()
5  {
6    int a_actual[N],a_expected[N],b[N],c[N],d[N];
7
8    int i,passed = 1;
9
10   for(i = N - 1; i >= 0 ; i--)
11   {
12     b[i] = N*(i * i % 3);
13     c[i] = i;
14     d[i] = i % 2;
15     if(d[i])
16     {
17       a_expected[i] = b[i] + c[i];
18     }
19     if(!d[i])
20     {
21       a_expected[i] = b[i] - c[i];
22     }
23   }
24
25   lab11_4(a_actual,b,c,d);
26
27   for(i = N - 1; i >= 0; i--)
28   {
29     printf("Expected [%d] actual [%d]\n",a_expected[i],a_actual[i]);
30     if(a_expected[i] != a_actual[i])
31     {
32       passed = -1;
33     }
34   }
35
36   if(passed != 1) {
37     printf("----------Test failed----------\n");
38   } else {
39     printf("----------Test passed----------\n");
40   }
41 }
```

Рис. 2.3. Исходный код теста

3. Скрипт

Ниже приводится скрипт, для автоматизации выполнения лабораторной работы.

```
open_project -reset lab11_4

add_files lab11_4.c
add_files -tb lab11_4_test.c
set_top lab11_4

set solutions [list 1a 2a]

foreach sol $solutions {
  open_solution solution_$sol -reset
  set_part {xa7a12tcsg325-1q}
  create_clock -period 10ns
  set_clock_uncertainty 0.1

  if {$sol == "2a"} {
    set_directive_loop_merge lab11_4
  }

  csim_design
  csynth_design
  cosim_design -trace_level all
}

exit
```

Рис. 3.1. Скрипт

4. Моделирование

Ниже приведены результаты моделирования.

```
INFO: [APCC 202-1] APCC is done.
   Generating csim.exe
Expected [3] actual [3]
Expected [2] actual [2]
Expected [5] actual [5]
Expected [0] actual [0]
----------Test passed----------
INFO: [SIM 211-1] CSim done with 0 errors.
INFO: [SIM 211-3] *************** CSIM finish ***************
```

Рис. 4.1. Результаты моделирования

По результатам моделирования видно, что устройство работает корректно.

5. Решение 1a

5.1. Директивы

В данном решения были установлены директивы, приведённые ниже.

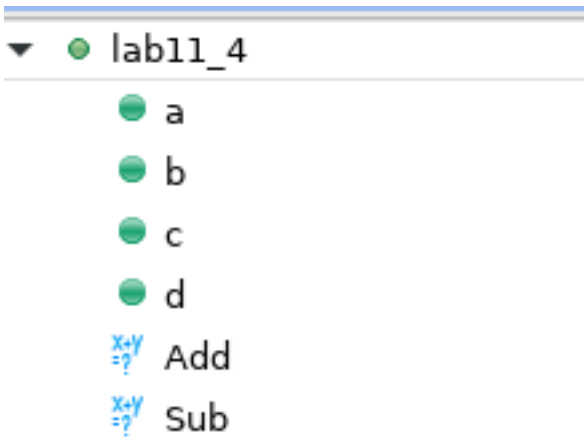

Рис. 5.1. Директивы

5.2. Синтез

По оценке производительности видно, что устройство соответствует заданным критериям.

Performance Estimates

Timing (ns)

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	10.00	7.006	0.10

Latency (clock cycles)

Summary

Latency		Interval		
min	max	min	max	Type
26	26	26	26	none

Рис. 5.2. Performance estimates

Utilization Estimates

Summary

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	120
FIFO	-	-	-	-
Instance	-	-	-	-
Memory	-	-	-	-
Multiplexer	-	-	-	134
Register	-	-	79	-
Total	0	0	79	254
Available	40	40	16000	8000
Utilization (%)	0	0	~0	3

Рис. 5.3. Utilization estimates

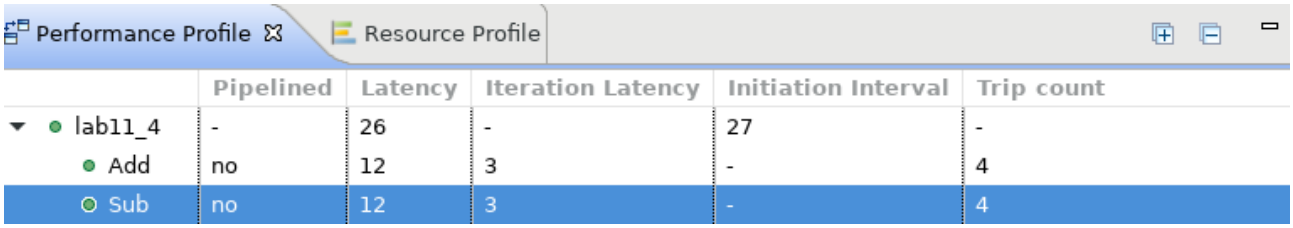

Рис. 5.4. Performance profile

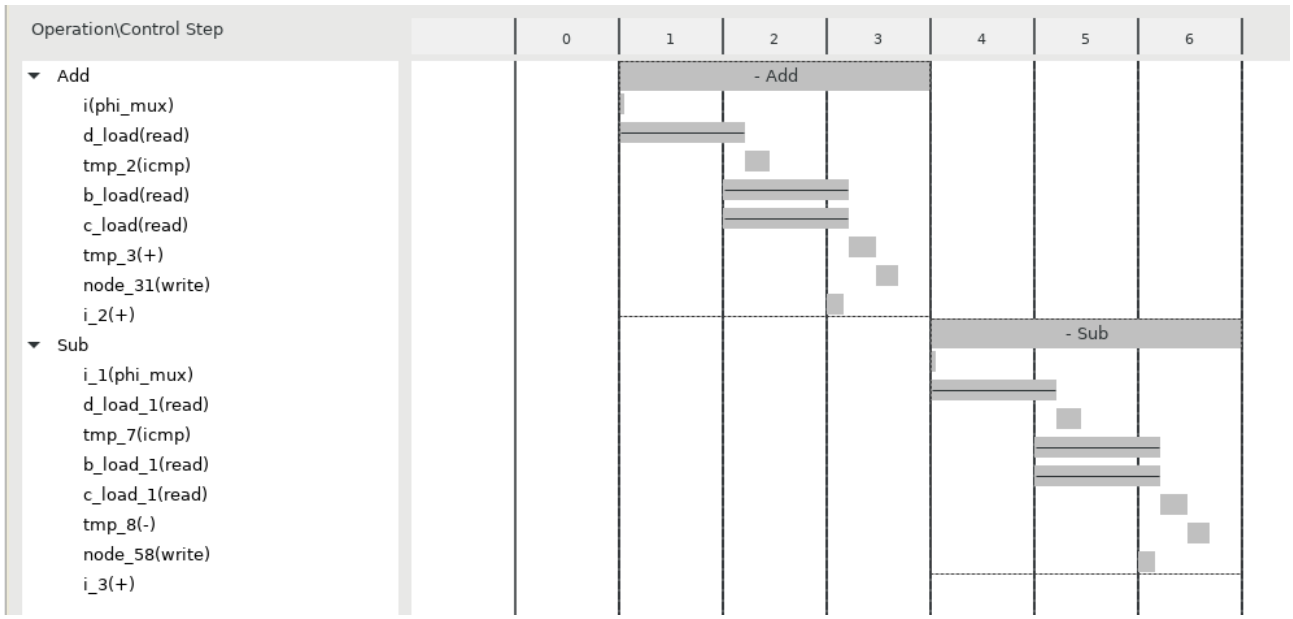

Рис. 5.5. Scheduler viewer

	Resource\Control Step	C0	C1	C2	C3	C4	C5	C6
1	I/O Ports							
2	d(p0)		read			read		
3	c(p0)			read			read	
4	b(p0)			read			read	
5	a(p0)				write			write
6	Memory Ports							
7	d(p0)		read			read		
8	b(p0)			read			read	
9	c(p0)			read			read	
10	a(p0)				write			write
11	Expressions							
12	i_phi_fu_124		phi_mux					
13	grp_fu_144			icmp			icmp	
14	i_2_fu_174				+			
15	tmp_3_fu_167				+			
16	i_1_phi_fu_136					phi_mux		
17	i_3_fu_204							+
18	tmp_8_fu_197							-

Рис. 5.6. Resource viewer

5.3. C/RTL моделирование

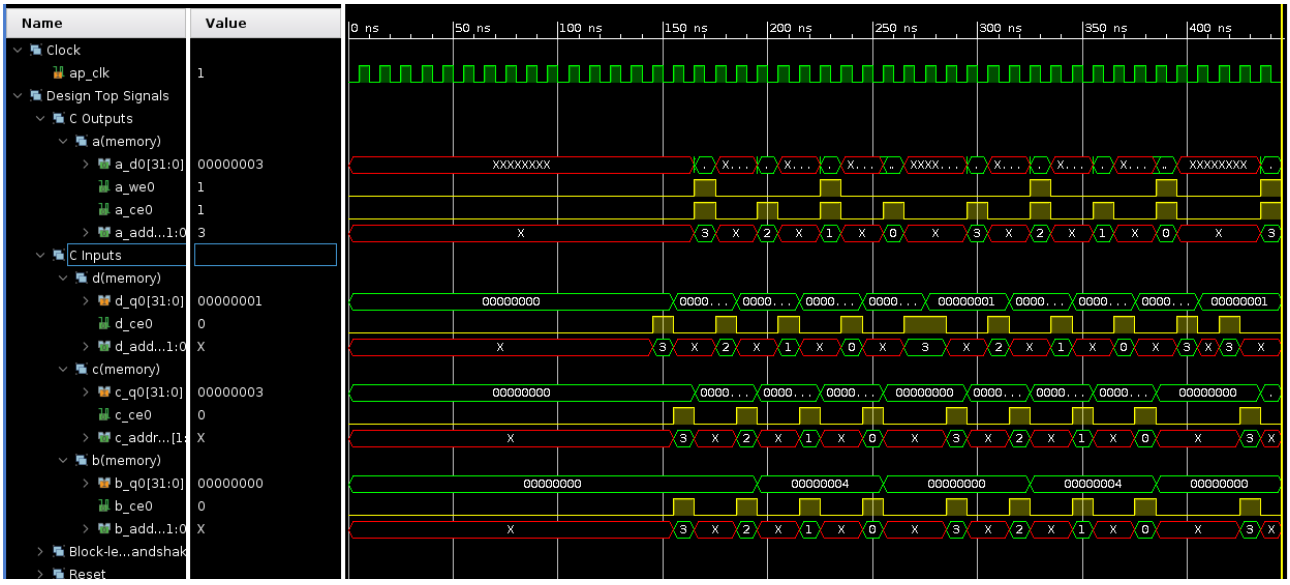

Рис. 5.7. Временная диаграмма

Как видно из диаграммы, в устройстве имеются 2 цикла, на каждую итерацию цикла требуется 3 такта и 1 общий подготовительный такт Latency = 4*3 + 4*3 + 1 + 1 такт между циклами = 26 тактов. II = Latency + 1 = 27

6. Решение 2a

6.1. Директивы

В данном решения были установлены директивы, приведённые ниже.

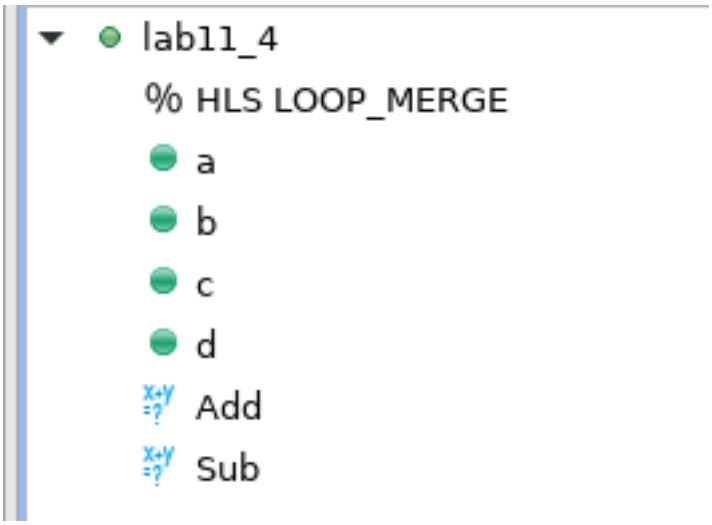

Рис. 6.1. Директивы

6.2. Синтез

По оценке производительности видно, что устройство соответствует заданным критериям.

Performance Estimates

Timing (ns)

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	10.00	7.006	0.10

Latency (clock cycles)

Summary

Latency		Interval		
min	max	min	max	Type
26	26	26	26	none

Рис. 6.2. Performance estimates

Данное решение полностью совпадает с предыдущим.

7. Вывод

Исходя из описания директивы LOOP_MERGE, циклы должны были объединится в один для избавления от дополнительных тактов, вызванных инициализацией. Не смотря на ожидаемое поведение, для описанного устройства циклы не объединились.