

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

Лабораторная №12

Предмет: Проектирование реконфигурируемых гибридных вычислительных систем

Тема: Inline

Задание 1

Студенты:

Соболь В.

Темнова А.С.

Группа: 13541/3

Преподаватель:

Антонов А.П.

Санкт-Петербург
2019

Содержание

1. Задание	3
2. Исходный код	5
3. Скрипт	6
4. Моделирование	6
5. Решение 1а	7
5.1. Директивы	7
5.2. Синтез	8
5.3. C/RTL моделирование	10
6. Решение 2а	10
6.1. Директивы	10
6.2. Синтез	11
7. Вывод	11

1. Задание

1. Создать проект lab12_1
2. Микросхема: xa7a12tcsg325-1q
3. Создать иерархическую функцию ,

```
int sumsub_func(int *in1, int *in2, int *outSum, int *outSub) {
    *outSum = *in1 + *in2;
    *outSub = *in1 - *in2;
}

int shift_func(int *in1, int *in2, int *outA, int *outB) {
    *outA = *in1 >> 1;
    *outB = *in2 >> 2;
}

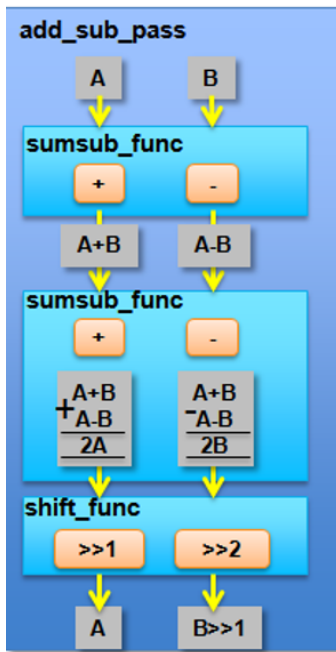
void add_sub_pass(int A, int B, int *C, int *D) {
    int apb, amb;
    int a2, b2;

    sumsub_func(&A,&B,&apb,&amb);
    sumsub_func(&apb,&amb,&a2,&b2);
    shift_func(&a2,&b2,C,D);
}
```

4. Создать тест lab12_1_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)

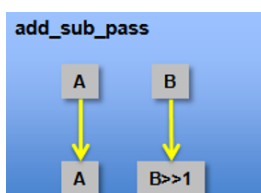
- На скриншоте показать Latency
- На скриншоте показать Initiation Interval
- Выполнить cosimulation и привести временную диаграмму
- Убедиться в том, что требуется 2 сумматора и 2 вычитателя

No Inlining



7. Solution_2a

- задать: clock period 10; clock_uncertainty 0.1
- установить реализацию Inlining
- 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
- Убедиться в том, что для реализации не требуется ресурсов.



8. Сравнить два решения (solution_1a и solution_2a) и сделать выводы

2. Исходный код

Ниже приведен исходный код устройства и теста.

```
1 void sumsub_func(int *in1 ,int *in2 ,int *outSum, int *outSub)
2 {
3     *outSum = *in1 + *in2;
4     *outSub = *in1 - *in2;
5 }
6 void shift_func(int *in1 ,int *in2 ,int *outA, int *outB)
7 {
8     *outA = *in1 >> 1;
9     *outB = *in2 >> 2;
10 }
11 void add_sub_pass(int A,int B,int *C, int *D)
12 {
13     int apb, amb;
14     int a2, b2;
15
16     sumsub_func(&A,&B,&apb,&amb);
17     sumsub_func(&apb,&amb,&a2,&b2);
18     shift_func(&a2,&b2,C,D);
19 }
```

Рис. 2.1. Исходный код устройства

```
1 #define N 16
2 int main()
3 {
4     int A,B,C,D,C_Exp,D_Exp;
5     int pass = 1;
6     for(int i = 0; i < N; i++)
7     {
8         A = (i*123 - 16) / 7;
9         B = (i*A - 11) / 3;
10        C_Exp = ((A - B) + (A + B)) >> 1;
11        D_Exp = ((A + B) - (A - B)) >> 2;
12        add_sub_pass(A,B,&C,&D);
13        printf("A=%d B=%d C=%d C_Exp=%d D=%d D_Exp=%d\n",A,B,C,C_Exp,D,
14        ↪ D_Exp);
15        if(C != C_Exp || D != D_Exp)
16            pass = 0;
17    }
18    if(pass)
19    {
20        printf("—————Test_passed—————\n");
21    } else {
22        printf("—————Test_failed—————\n");
23    }
24    return 0;
25 }
```

Рис. 2.2. Исходный код теста

3. Скрипт

Ниже приводится скрипт, для автоматизации выполнения лабораторной работы.

```
1 open_project -reset lab12_1
2
3 add_files lab12_1.c
4 add_files -tb lab12_1_test.c
5 set_top add_sub_pass
6
7 set solutions [list 1a 2a]
8
9 foreach sol $solutions {
10   open_solution solution_$sol -reset
11   set_part {xa7a12tcsg325-1q}
12   create_clock -period 10ns
13   set_clock_uncertainty 0.1
14
15   if {$sol == "2a"} {
16     set_directive_inline -recursive add_sub_pass
17   }
18
19   csim_design
20   csynth_design
21   cosim_design -trace_level all
22 }
23
24 exit
```

Рис. 3.1. Скрипт

4. Моделирование

Ниже приведены результаты моделирования.

```

INFO: [APCC 202-3] tmp directory is /tmp/apcc_db_s000t/169480157
INFO: [APCC 202-1] APCC is done.
    Generating csim.exe
A = -2 B = -3 C = -2 C_Exp = -2 D = -2 D_Exp = -2
A = 15 B = 1 C = 15 C_Exp = 15 D = 0 D_Exp = 0
A = 32 B = 17 C = 32 C_Exp = 32 D = 8 D_Exp = 8
A = 50 B = 46 C = 50 C_Exp = 50 D = 23 D_Exp = 23
A = 68 B = 87 C = 68 C_Exp = 68 D = 43 D_Exp = 43
A = 85 B = 138 C = 85 C_Exp = 85 D = 69 D_Exp = 69
A = 103 B = 202 C = 103 C_Exp = 103 D = 101 D_Exp = 101
A = 120 B = 276 C = 120 C_Exp = 120 D = 138 D_Exp = 138
A = 138 B = 364 C = 138 C_Exp = 138 D = 182 D_Exp = 182
A = 155 B = 461 C = 155 C_Exp = 155 D = 230 D_Exp = 230
A = 173 B = 573 C = 173 C_Exp = 173 D = 286 D_Exp = 286
A = 191 B = 696 C = 191 C_Exp = 191 D = 348 D_Exp = 348
A = 208 B = 828 C = 208 C_Exp = 208 D = 414 D_Exp = 414
A = 226 B = 975 C = 226 C_Exp = 226 D = 487 D_Exp = 487
A = 243 B = 1130 C = 243 C_Exp = 243 D = 565 D_Exp = 565
A = 261 B = 1301 C = 261 C_Exp = 261 D = 650 D_Exp = 650
-----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 1а

5.1. Директивы

В данном решения были установлены директивы, приведённые ниже.

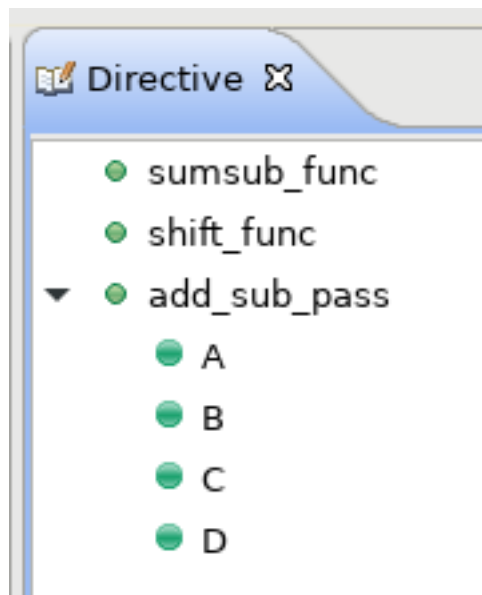


Рис. 5.1. Директивы

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	0.000	0.10

☐ **Latency (clock cycles)**

☐ **Summary**

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

Рис. 5.2. Performance estimates

Utilization Estimates

Summary

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	-	-
FIFO	-	-	-	-
Instance	-	-	-	-
Memory	-	-	-	-
Multiplexer	-	-	-	-
Register	-	-	-	-
Total	0	0	0	0
Available	40	4016000	8000	
Utilization (%)	0	0	0	0

Рис. 5.3. Utilization estimates

Performance Profile		Resource Profile			
	Pipelined	Latency	Iteration Latency	Initiation Interval	Trip count
• add_sub_pass	-	0	-	1	-

Рис. 5.4. Performance profile

Operation\Control Step		0
B_read(read)		
A_read(read)		
node_16(write)		
node_17(write)		

Рис. 5.5. Scheduler viewer

	Resource\Contro...	C0
1	I/O Ports	
2	A	read
3	B	read
4	C	write
5	D	write

Рис. 5.6. Resource viewer

5.3. C/RTL моделирование

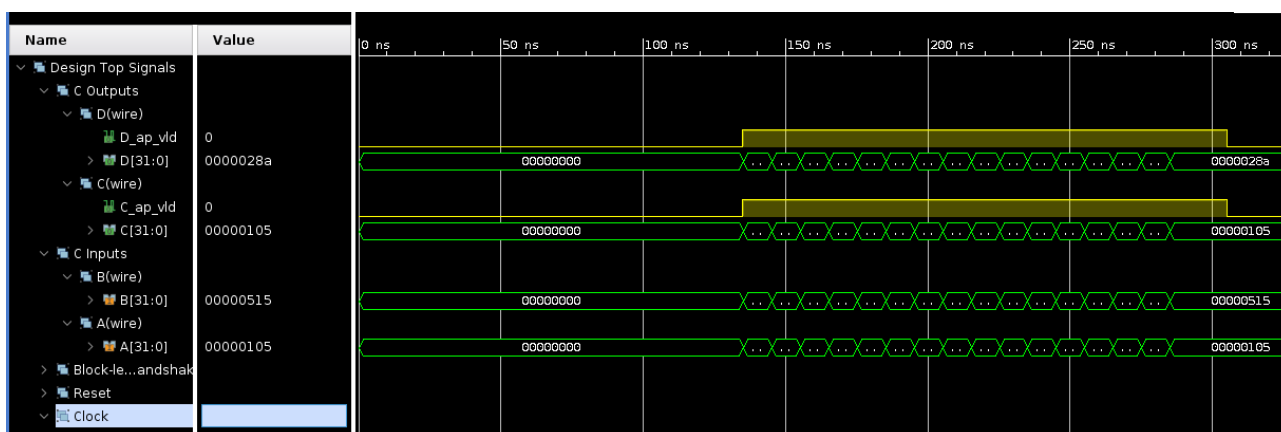


Рис. 5.7. Временная диаграмма

По результатам выше видно, что среда разработки выполнила inlining, хотя это и не было указано явно.

6. Решение 2а

6.1. Директивы

В данном решении были установлены директивы, приведённые ниже.

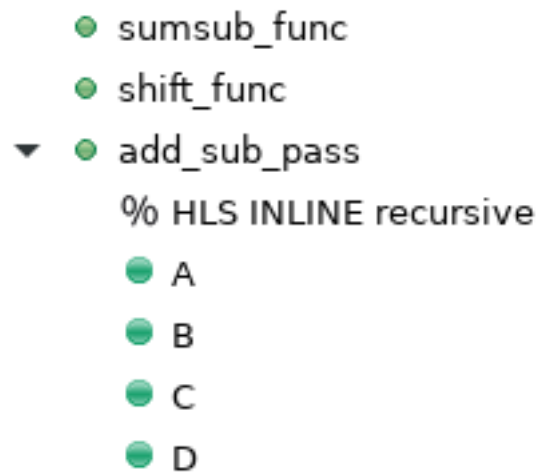


Рис. 6.1. Директивы

6.2. Синтез

Данное решение не отличается от предыдущего, так как в предыдущем, `inlining` был применён без явного указания, а в данном решении, после явного указания, ничего не поменялось.

7. Вывод

В данной лабораторной работе не удалось сравнить производительность устройства после применения `inlining` и без применения, так как среда разработки применяет его, даже если это явно не указано, по причине того, что для данного устройства это однозначно приводит к улучшению характеристик.