

Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и кибербезопасности
Высшая школа компьютерных технологий и информационных систем

Отчёт по лабораторной работе Lab_PD2

Дисциплина: Автоматизация проектирования дискретных устройств (на
английском языке)

Выполнил студент гр. 5130901/10101 _____ М.Т. Непомнящий
(подпись)

Руководитель _____ А.А. Федотов
(подпись)

Санкт-Петербург
2024

1. Оглавление

1.	Оглавление	2
1.	Задание.....	4
2.	Ход работы	4
2.1.	Создание проекта.....	4
	Подготовка проекта	4
	Начало работы в PD	5
2.2.	Настройка компонентов.....	6
	Настройка clk_0.....	6
	Настройка sc_fifo_0.....	6
	Настройка MyST_source_0 и MyST_sink_0.....	7
2.3.	Подключение тактового сигнала	8
2.4.	Подключение Avalon-ST интерфейсов.....	8
2.5.	Анализ системы	10
	Проверка блока.....	10
	Анализ с помощью Schematic	11
2.6.	Генерация системы.....	12
2.7.	Подключение файлов к проекту.....	13
3.	Тестирование проекта	14
3.1.	Тестирование средствами ModelSim	14
	Создание тестового файла.....	14
	Симуляция средствами ModelSim	15
3.2.	Тестирование средствами Signal Tap II.....	16
	Создание файла для отладки.....	16
	Настройка Signal Tap II.....	17
	Тестирование на плате средствами Signal Tap II	17
4.	Вывод.....	19

Список иллюстраций

Рис. 1 – Структура проекта	4
Рис. 2 – Детали проекта	4
Рис. 3 – Задания пути к библиотеке IP	5
Рис. 4 – Исходное окно PD	5
Рис. 5 – Добавление компонентов	6
Рис. 6 – Настройка компонента clk	6
Рис. 7 – Настройка компонента sc_fifo	7
Рис. 8 – Система после переименования компонентов	7
Рис. 9 – Подключение тактового сигнала (1)	8
Рис. 10 – Подключение тактового сигнала (2)	8
Рис. 11 – Подключение Avalon-MM интерфейсов	8
Рис. 12 – Экспорт выводов	9
Рис. 13 – Проверка поля Messages на отсутствие ошибок	9
Рис. 14 – Символ системы	10
Рис. 15 – Show System with QSYS Interconnect	10
Рис. 16 – Анализ проблемных подключений	11
Рис. 17 – Schematic (фильтр по in)	11
Рис. 18 – Schematic (фильтр по clk)	11
Рис. 19 – Schematic	12
Рис. 20 – Предустановки окна Generation	12
Рис. 21 – Проверка успешности генерации HDL	12
Рис. 22 – Подключение файлов к проекту	13
Рис. 23 – Синтаксис файла Lab2_top.sv	13
Рис. 24 – Схема проекта в RTL Viewer	13
Рис. 25 – Тестовый файл tb_lab_PD2_top.sv	14
Рис. 26 – Тестовый файл tb_lab_PD2_top.sv	15
Рис. 27 – Моделирование проекта средствами ModelSim	15
Рис. 28 – Файл для отладки модуля верхнего уровня	16
Рис. 29 – Схема проекта с добавлением SP_unit в RTL Viewer	16
Рис. 30 – Настройка окна Signal Tap II	17
Рис. 31 – Временные характеристики устройства	17
Рис. 32 – Результат SignalTap II	17
Рис. 33 – Анализ рабочей папки проекта	18

1. Задание

Средствами Platform Designer создать структуру проекта, представленную на рисунке ниже:

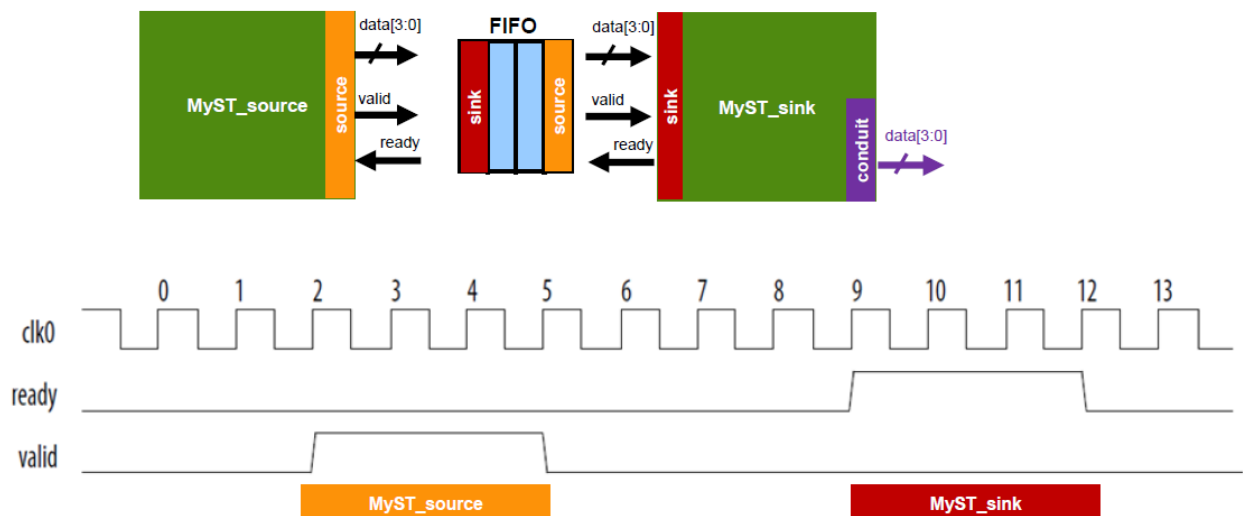


Рис. 1 – Структура проекта

2. Ход работы

2.1. Создание проекта

Подготовка проекта

Создадим проект, установив следующие значения:

Summary	
When you click Finish, the project will be created with the following settings:	
Project directory:	D:\Users\Legion\Quartus and Verilog\Verilog\Third year\6th semester\lectures\lecture 4 - PD\Q_PD\lab_PD1
Project name:	lab_PD1
Top-level design entity:	lab_PD1
Number of files added:	0
Number of user libraries added:	0
Device assignments:	
Design template:	n/a
Family name:	Cyclone IV E
Device:	EP4CE6E22C8
Board:	n/a
EDA tools:	
Design entry/synthesis:	<None> (<None>)
Simulation:	ModelSim-Altera (SystemVerilog HDL)
Timing analysis:	()
Operating conditions:	
VCCINT voltage:	1.2V
Junction temperature range:	0-85 °C

Рис. 2 – Детали проекта

Перейдём по пути Tools → Options → IP Settings → IP Catalog Search Locations и зададим путь к библиотеке IP:

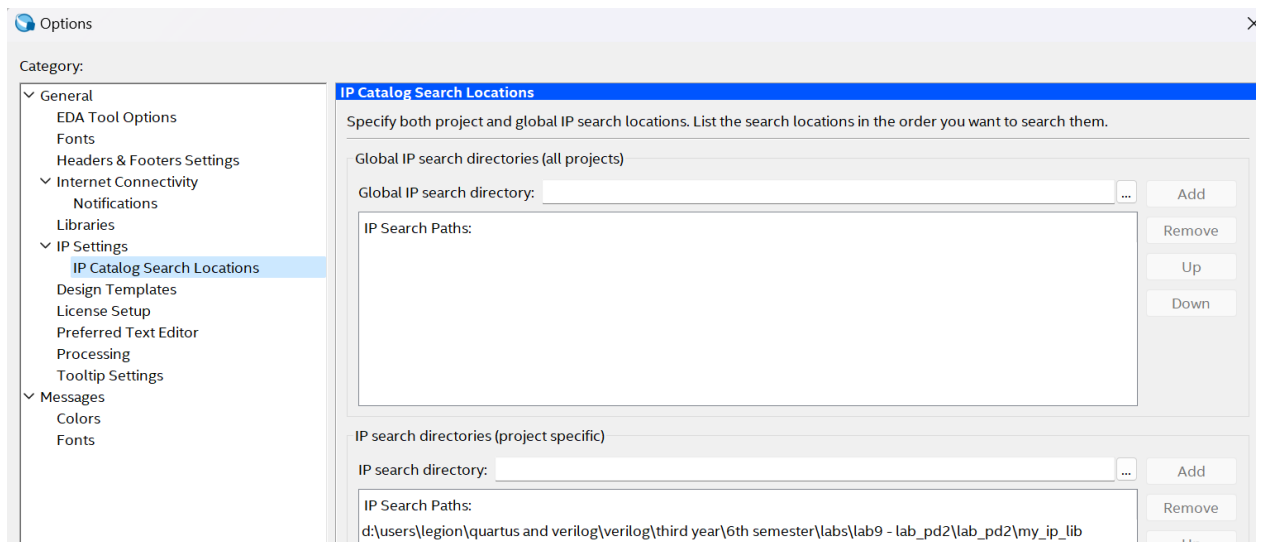


Рис. 3 – Задания пути к библиотеке IP

Начало работы в PD

Откроем PD и сохраним систему:

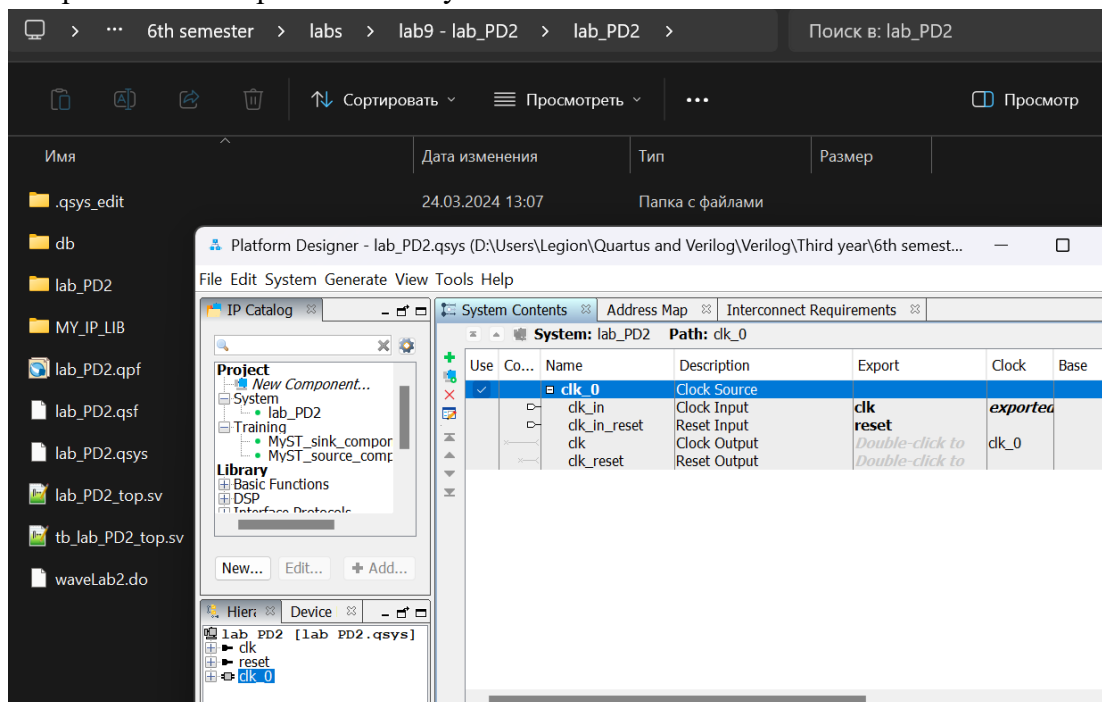


Рис. 4 – Исходное окно PD

Добавим компоненты: MyST_source_component, Avalon-ST Single Clock FIFO, MyST_sink_component. Таким образом, получим следующую картинку (в окне Hierarchy слева отображаются все добавленные компоненты):

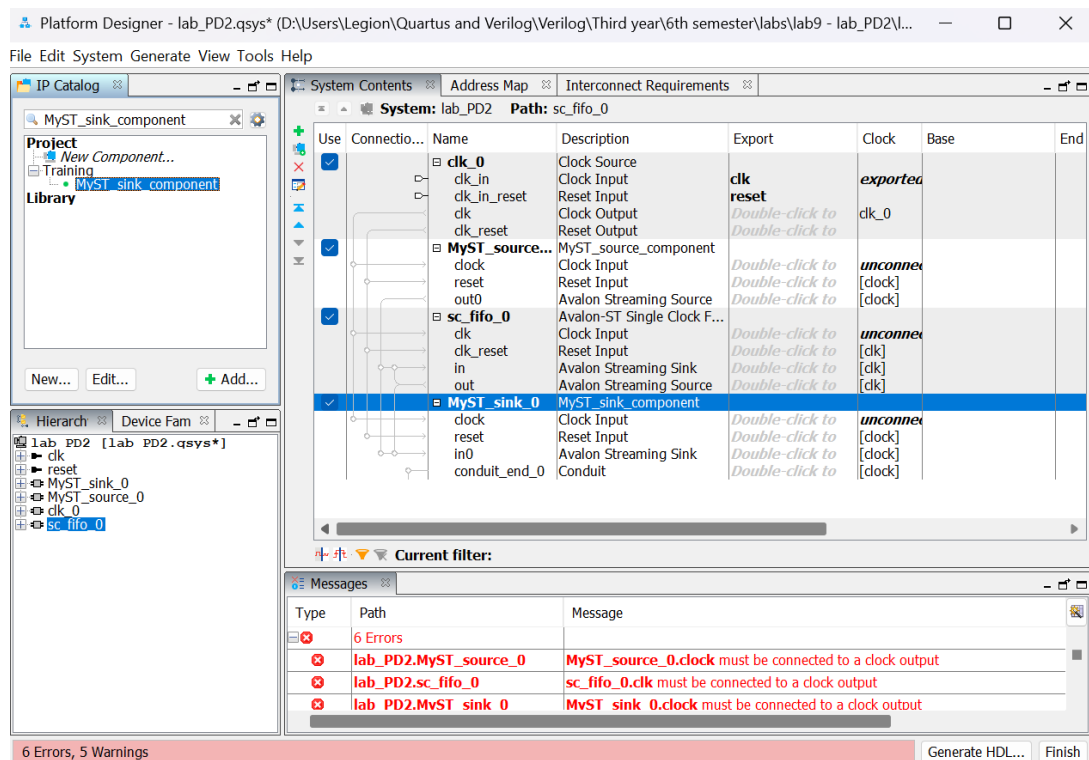


Рис. 5 – Добавление компонентов

Наличие ошибок связано с тем, что настройка модулей не производилась, т. к. она будет рассмотрена дальше.

2.2. Настройка компонентов

Настройка clk_0

Переименуем компонент **clk_0** в **clk** и зададим значение Reset synchronous edges = Deassert

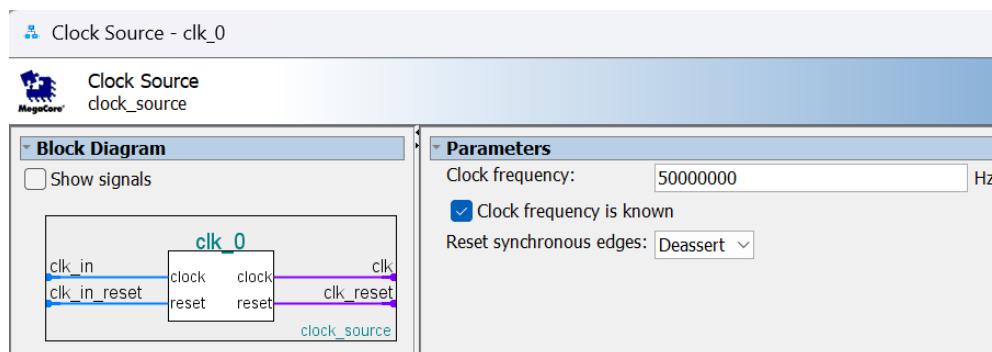


Рис. 6 – Настройка компонента clk

Настройка sc_fifo_0

Переименуем компонент **dc_fifo_0** в **sc_fifo**. Зададим значение Bits per symbol = 4:

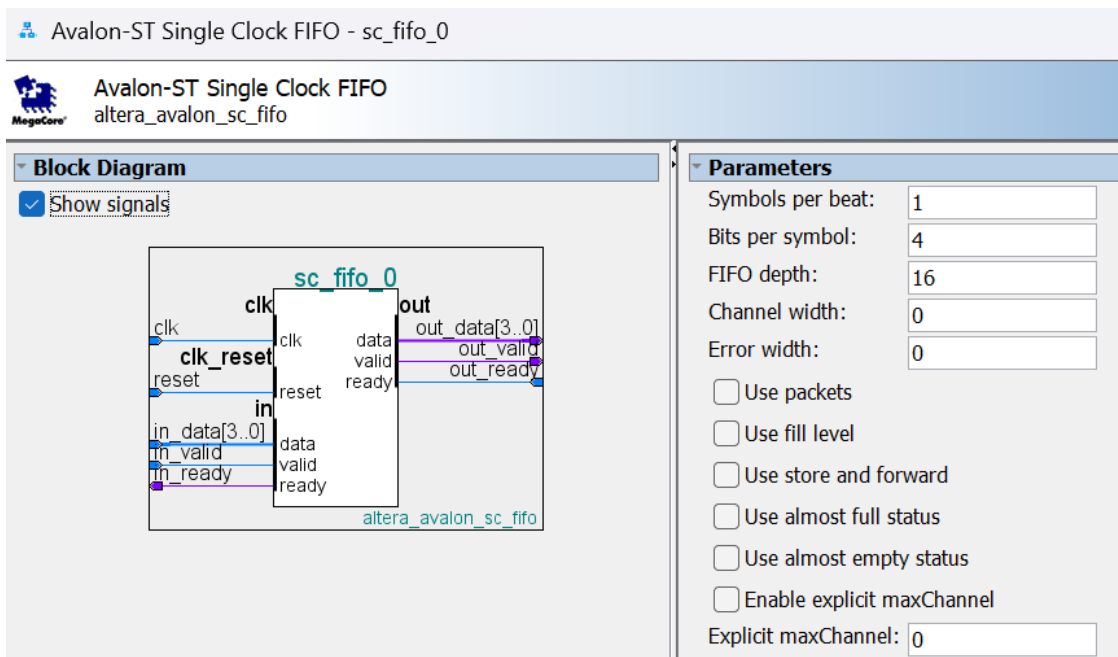


Рис. 7 – Настройка компонента sc_fifo

Запись данных будет происходить следующим образом: 100 – счёт на сложение, 200 – счёт на вычитание.

Настройка MyST_source_0 и MyST_sink_0

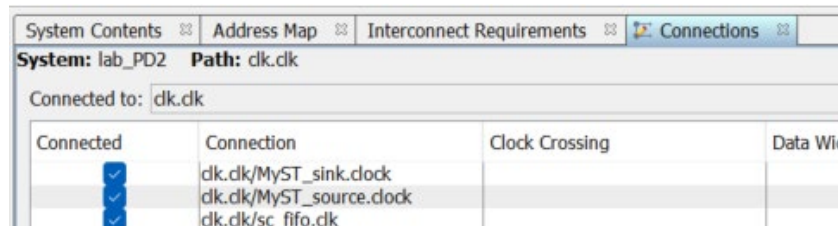
Переименуем MyST_source_0 и MyST_sink_0 в MyST_source и MyST_sink соответственно. Получившаяся структура будет выглядеть следующим образом:

Use	Connectio...	Name	Description	Export	Clock	Base
<input checked="" type="checkbox"/>		clk	Clock Source			
		clk_in	Clock Input	clk	exported	
		clk_in_reset	Reset Input	reset	[clk_in]	
		clk	Clock Output	Double-click to	clk	
		clk_reset	Reset Output	Double-click to	clk	
<input checked="" type="checkbox"/>		MyST_source	MyST_source_component			
		clock	Clock Input	Double-click to	unconnex	
		reset	Reset Input	Double-click to	[clock]	
		out0	Avalon Streaming Source	Double-click to	[clock]	
<input checked="" type="checkbox"/>		sc_fifo	Avalon-ST Single Clock F...			
		clk	Clock Input	Double-click to	unconnex	
		clk_reset	Reset Input	Double-click to	[clk]	
		in	Avalon Streaming Sink	Double-click to	[clk]	
		out	Avalon Streaming Source	Double-click to	[clk]	
<input checked="" type="checkbox"/>		MyST_sink	MyST_sink_component			
		clock	Clock Input	Double-click to	unconnex	
		reset	Reset Input	Double-click to	[clock]	
		in0	Avalon Streaming Sink	Double-click to	[clock]	
		conduit end 0	Conduit	Double-click to	MyST_sink reset	

Рис. 8 – Система после переименования компонентов

2.3. Подключение тактового сигнала

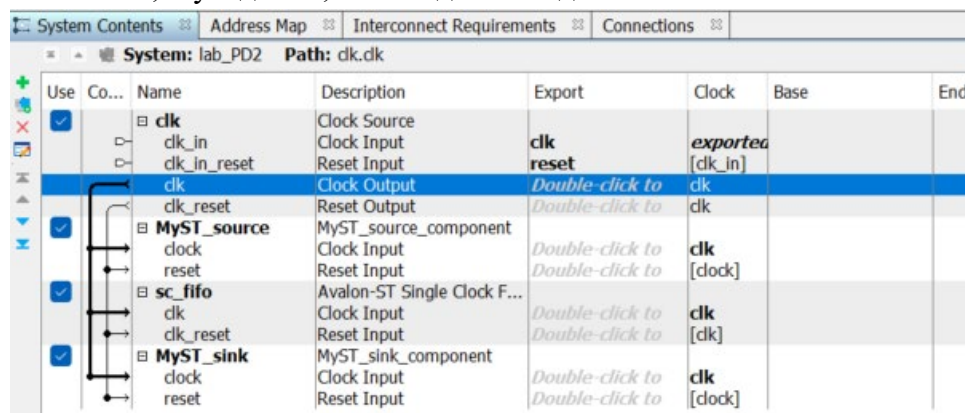
Выделим интерфейс clk компонента clk, и, открыв его соединения, выберем подключение ко всем тактовым входам:



Connected	Connection	Clock Crossing	Data Width
<input checked="" type="checkbox"/>	clk.clk/MyST_sink.clock		
<input checked="" type="checkbox"/>	clk.clk/MyST_source.clock		
<input checked="" type="checkbox"/>	clk.clk/sc_fifo.clk		

Рис. 9 – Подключение тактового сигнала (1)

Подключим тактовый сигнал, выполнив Filter → Clock and Reset Interfaces, убедимся, что соединения выполнены. Также, подключим сигнал Reset, выполнив System → Create Global Reset Network, и убедимся, что соединения для reset также выполнены:

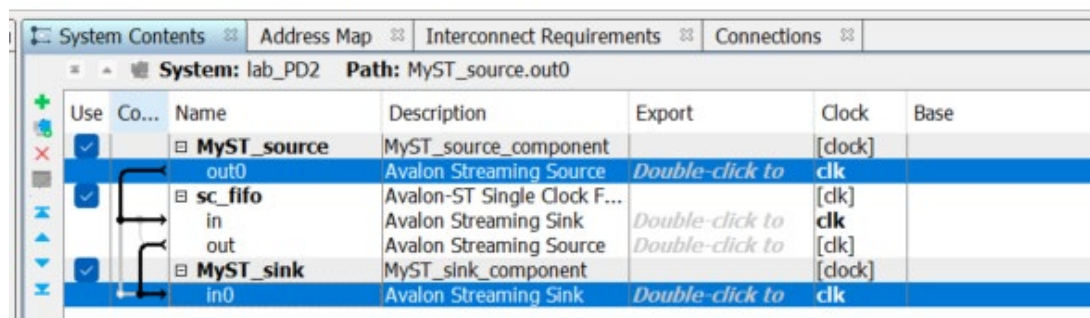


Use	Co...	Name	Description	Export	Clock	Base	End
<input checked="" type="checkbox"/>		clk	Clock Source				
<input checked="" type="checkbox"/>		clk_in	Clock Input	clk	exported		
<input checked="" type="checkbox"/>		clk_in_reset	Reset Input	reset	[clk_in]		
<input checked="" type="checkbox"/>		clk	Clock Output	Double-click to	clk		
<input checked="" type="checkbox"/>		clk_reset	Reset Output	Double-click to	clk		
<input checked="" type="checkbox"/>		MyST_source	MyST_source_component				
<input checked="" type="checkbox"/>		clock	Clock Input	Double-click to	clk		
<input checked="" type="checkbox"/>		reset	Reset Input	Double-click to	[clock]		
<input checked="" type="checkbox"/>		sc_fifo	Avalon-ST Single Clock F...				
<input checked="" type="checkbox"/>		clock	Clock Input	Double-click to	clk		
<input checked="" type="checkbox"/>		clk_reset	Reset Input	Double-click to	[clk]		
<input checked="" type="checkbox"/>		MyST_sink	MyST_sink_component				
<input checked="" type="checkbox"/>		clock	Clock Input	Double-click to	clk		
<input checked="" type="checkbox"/>		reset	Reset Input	Double-click to	[clock]		

Рис. 10 – Подключение тактового сигнала (2)

2.4. Подключение Avalon-ST интерфейсов

Выполним Filter → Avalon-MM Interfaces и выберем соединения так, как показано на картинке ниже



Use	Co...	Name	Description	Export	Clock	Base
<input checked="" type="checkbox"/>		MyST_source	MyST_source_component		[clock]	
<input checked="" type="checkbox"/>		out0	Avalon Streaming Source	Double-click to	clk	
<input checked="" type="checkbox"/>		sc_fifo	Avalon-ST Single Clock F...			
<input checked="" type="checkbox"/>		in	Avalon Streaming Sink	Double-click to	clk	
<input checked="" type="checkbox"/>		out	Avalon Streaming Source	Double-click to	[clk]	
<input checked="" type="checkbox"/>		MyST_sink	MyST_sink_component			
<input checked="" type="checkbox"/>		in0	Avalon Streaming Sink	Double-click to	[clock]	

Рис. 11 – Подключение Avalon-MM интерфейсов

Проведём экспорт выводов путём задания имён для выделенных модулей в столбце Export:

System: lab_PD2 Path: MyST_sink.in0							
Use	Connectio...	Name	Description	Export	Clock	Base	End
<input checked="" type="checkbox"/>		clk	Clock Source	clk			
<input checked="" type="checkbox"/>		clk_in	Clock Input	reset	exported		
<input checked="" type="checkbox"/>		clk_in_reset	Reset Input	Double-click to	[clk_in]		
<input checked="" type="checkbox"/>		clk	Clock Output	Double-click to	clk		
<input checked="" type="checkbox"/>		clk_reset	Reset Output	Double-click to			
<input checked="" type="checkbox"/>		MyST_source	MyST_source_component	Double-click to			
<input checked="" type="checkbox"/>		clock	Clock Input	Double-click to	clk		
<input checked="" type="checkbox"/>		reset	Reset Input	Double-click to	[clock]		
<input checked="" type="checkbox"/>		out0	Avalon Streaming Source	Double-click to	[clock]		
<input checked="" type="checkbox"/>		sc_fifo	Avalon-ST Single Clock F...	Double-click to			
<input checked="" type="checkbox"/>		clk	Clock Input	Double-click to	clk		
<input checked="" type="checkbox"/>		clk_reset	Reset Input	Double-click to	[clk]		
<input checked="" type="checkbox"/>		in	Avalon Streaming Sink	Double-click to	[clk]		
<input checked="" type="checkbox"/>		out	Avalon Streaming Source	Double-click to	[clk]		
<input checked="" type="checkbox"/>		MyST_sink	MyST_sink_component	Double-click to			
<input checked="" type="checkbox"/>		clock	Clock Input	Double-click to	clk		
<input checked="" type="checkbox"/>		reset	Reset Input	Double-click to	[clock]		
<input checked="" type="checkbox"/>		in0	Avalon Streaming Sink	Double-click to	[clock]		
<input checked="" type="checkbox"/>		conduit_end_0	Conduit	dout	[clock]		

Рис. 12 – Экспорт выводов

Убедимся в том, что система не содержит ошибок и в поле Messages есть только 1 информационное сообщение:

0 Errors, 0 Warnings

Generate HDL... Finish

Рис. 13 – Проверка поля Messages на отсутствие ошибок

2.5. Анализ системы

Проверка блока

Выполним View → Block Symbol и убедимся в том, что символ системы построен правильно:

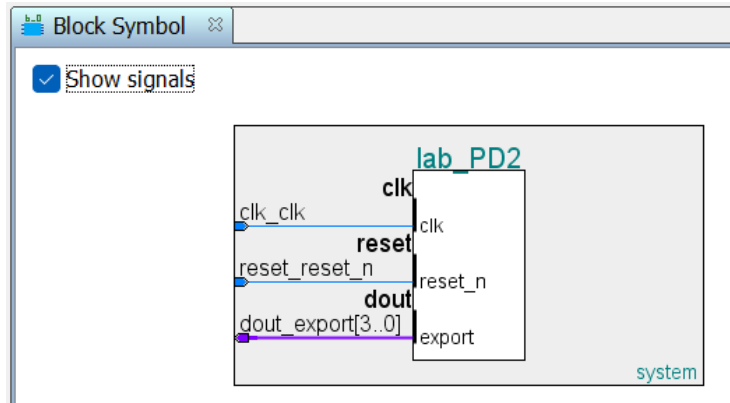


Рис. 14 – Символ системы

Выполним команду System → Show System with PD Interconnect (Show System with QSYS Interconnect). Проверим, что новых модулей не появилось.

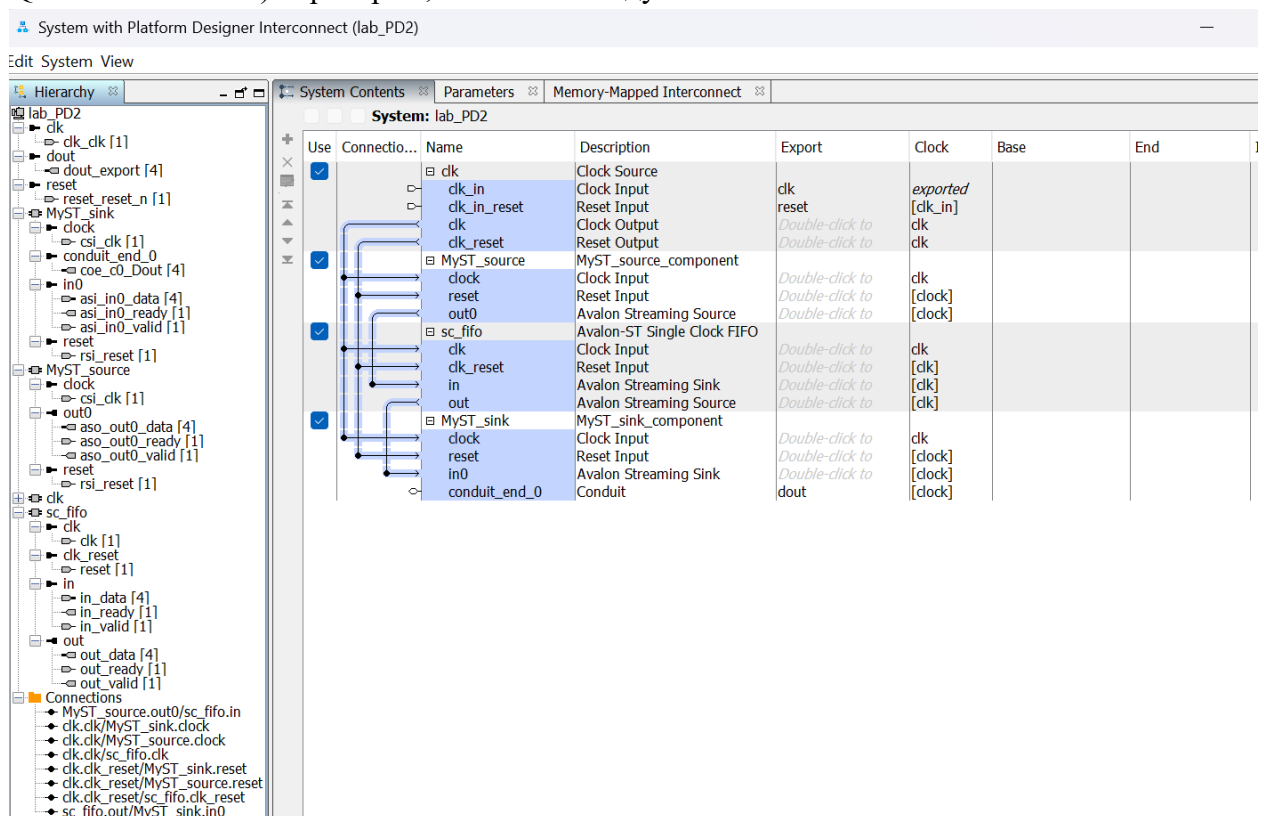


Рис. 15 – Show System with QSYS Interconnect

Выполним View → Clock domains Beta, выберем режим отображения Reset. Заметим, что проблемных подключений не выявлено:

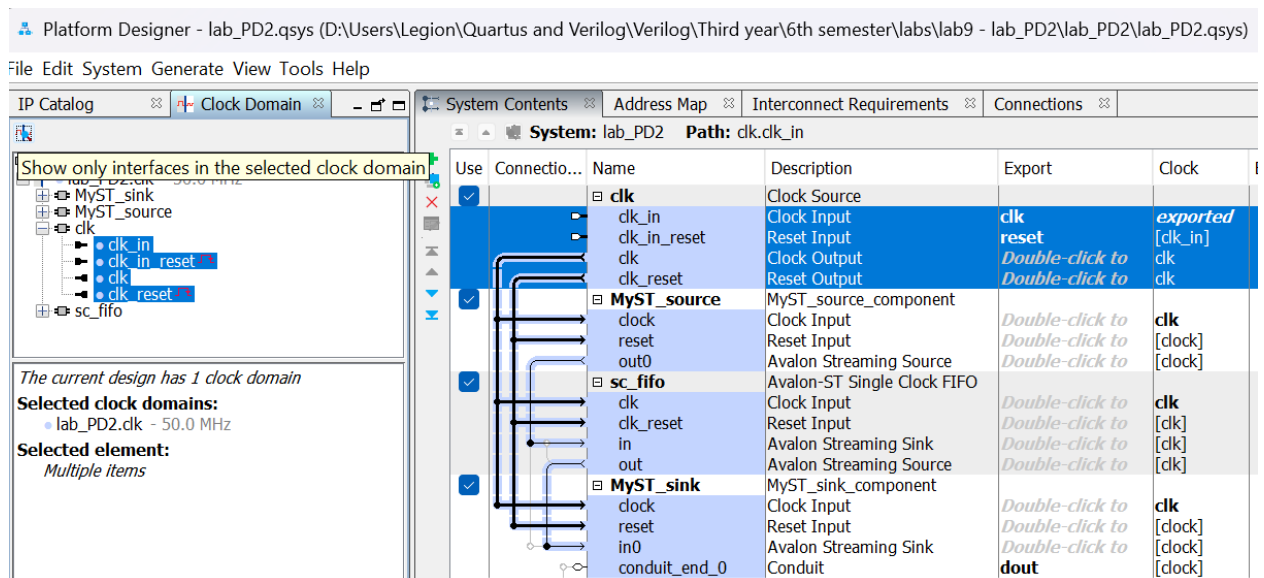


Рис. 16 – Анализ проблемных подключений

Анализ с помощью Schematic

Выполним View → Schematic, в качестве фильтра введём in и убедимся в том, что система синхронизации и каналы ST системы подключены верно:

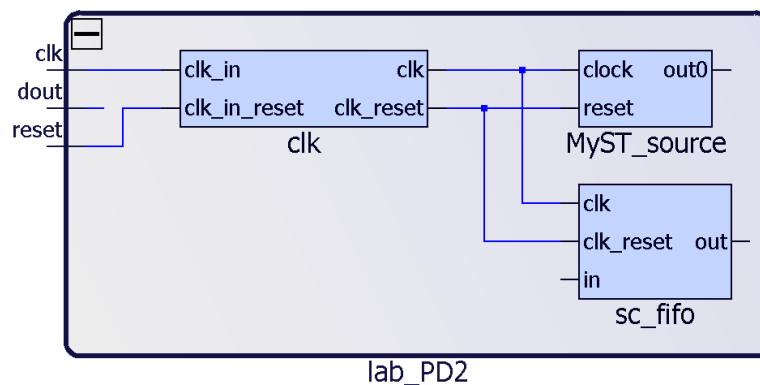


Рис. 17 – Schematic (фильтр по in)

Теперь введём в качестве фильтра clk, чтобы проверить, что шины Avalon MM подключены верно:

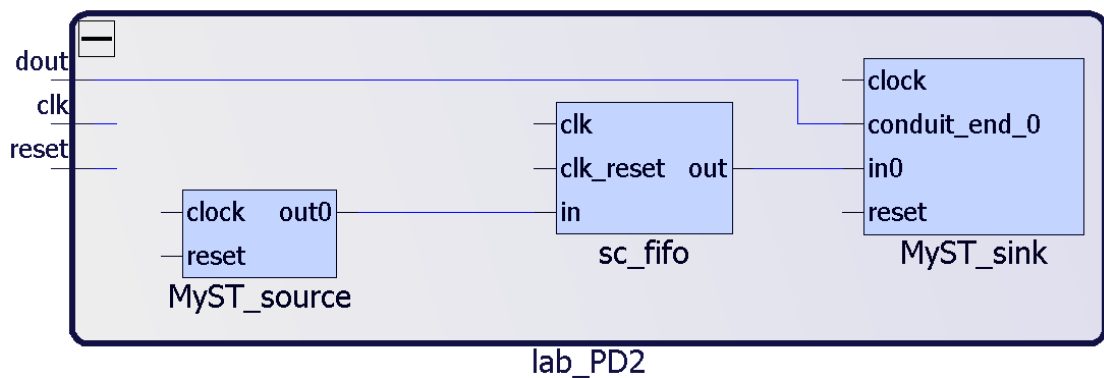


Рис. 18 – Schematic (фильтр по clk)

Полный блок системы будет выглядеть следующим образом:

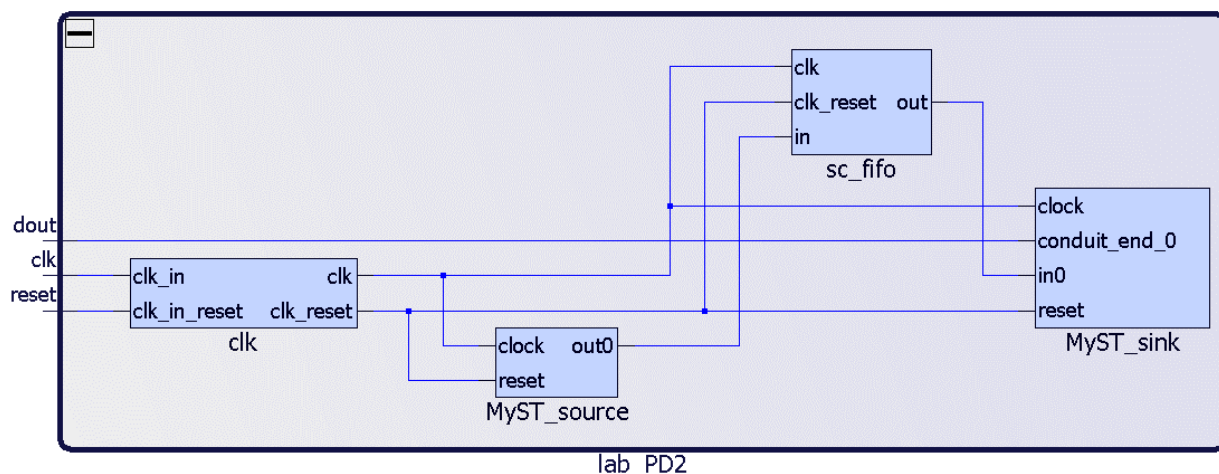


Рис. 19 – Schematic

2.6. Генерация системы

Выполним PD → Generate HDL и укажем следующие предустановки для генерации:

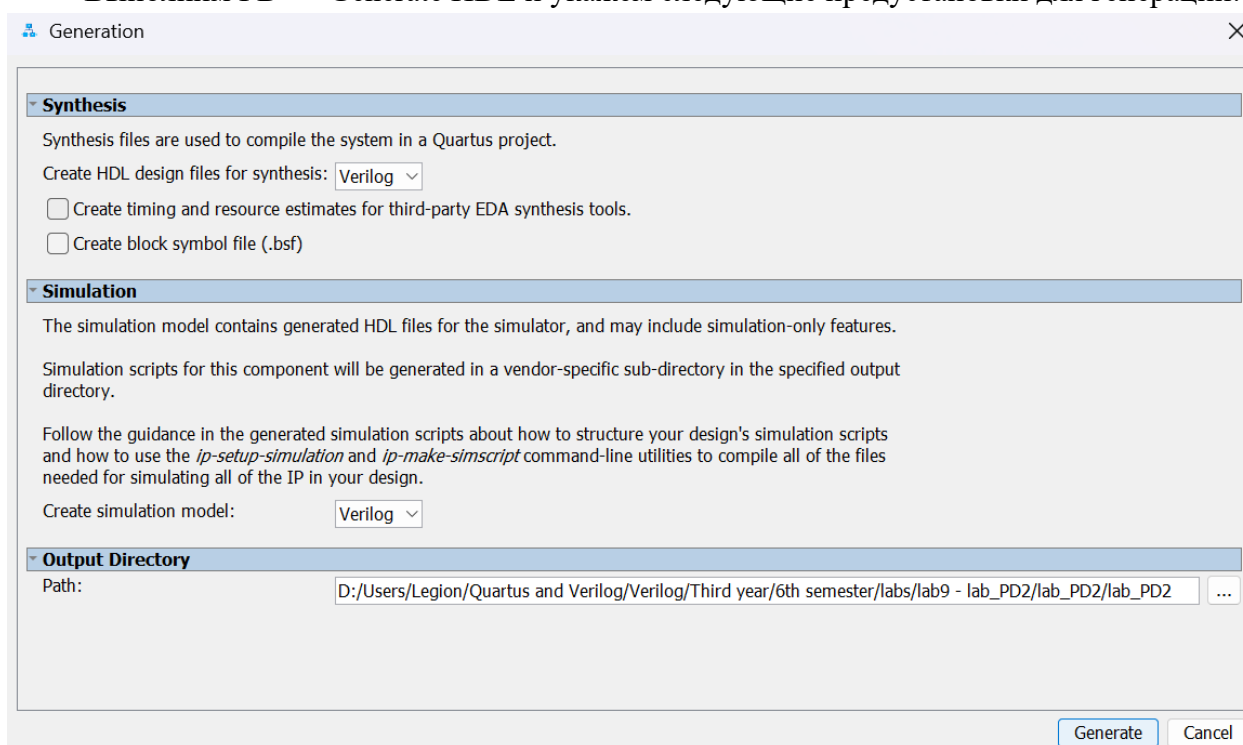


Рис. 20 – Предустановки окна Generation

Удостоверимся в том, что генерация прошла успешно:

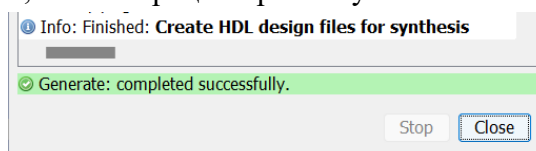


Рис. 21 – Проверка успешности генерации HDL

2.7. Подключение файлов к проекту

Подключим файлы к проекту в Quartus

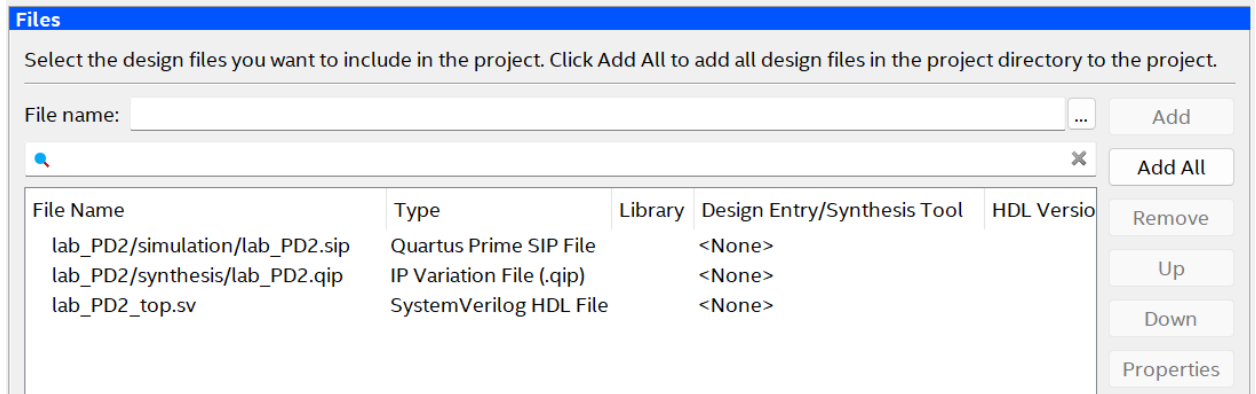


Рис. 22 – Подключение файлов к проекту

Синтаксис файла Lab2_top.sv:

```
lab_PD2 - lab_PD2_top.sv

1 `timescale 1 ns / 1 ns
2 module lab_PD2_top (
3     input bit clk,
4     input bit reset,
5     output bit [3:0] dout
6 );
7 lab_PD2 UUT (
8     .clk_clk      (clk),
9     .reset_reset_n (reset),
10    .dout_export   (dout)
11 );
12 endmodule
13
```

Рис. 23 – Синтаксис файла Lab2_top.sv

Выполним анализ и синтез проекта средствами QR и убедимся в правильности схемы средствами RTL Viewer:

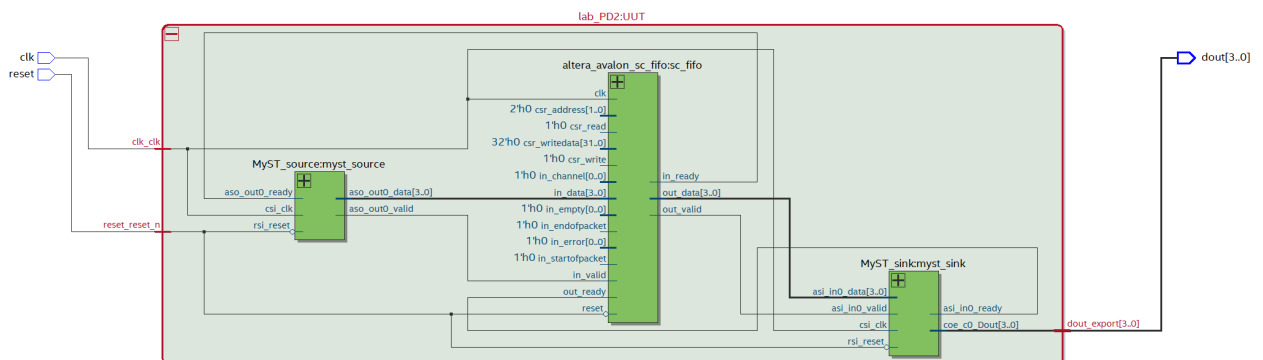


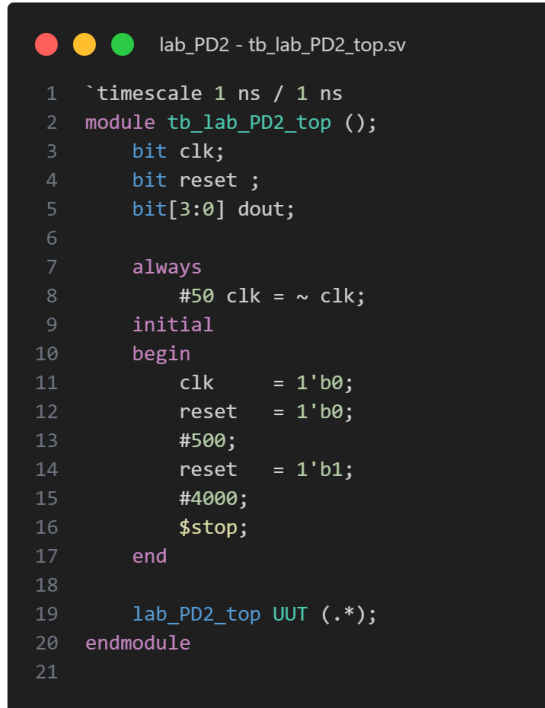
Рис. 24 – Схема проекта в RTL Viewer

3. Тестирование проекта

3.1. Тестирование средствами ModelSim

Создание тестового файла

Добавим тест первого класса для созданного проекта:



```
lab_PD2 - tb_lab_PD2_top.sv
1  `timescale 1 ns / 1 ns
2  module tb_lab_PD2_top ();
3      bit clk;
4      bit reset ;
5      bit[3:0] dout;
6
7      always
8          #50 clk = ~ clk;
9      initial
10     begin
11         clk      = 1'b0;
12         reset    = 1'b0;
13         #500;
14         reset    = 1'b1;
15         #4000;
16         $stop;
17     end
18
19     lab_PD2_top UUT (.*)
20 endmodule
21
```

Рис. 25 – Тестовый файл tb_lab_PD2_top.sv

Укажем созданный файл в качестве основного тестового файла, который будет выполняться при симуляции средствами ModelSim:

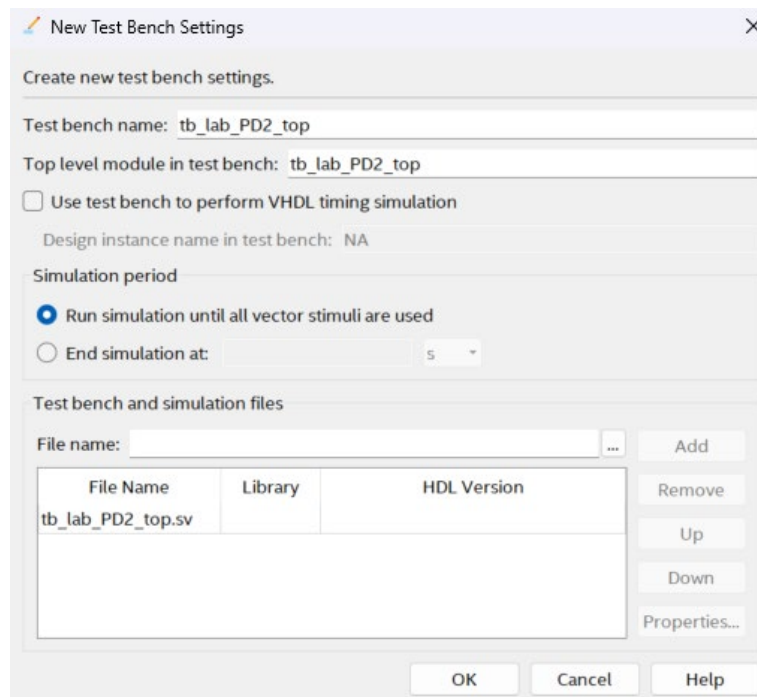


Рис. 26 – Тестовый файл tb_lab_PD2_top.vh

Симуляция средствами ModelSim

Выполним компиляцию проекта средствами ModelSim. Для этого запустим waveLab.do файл:

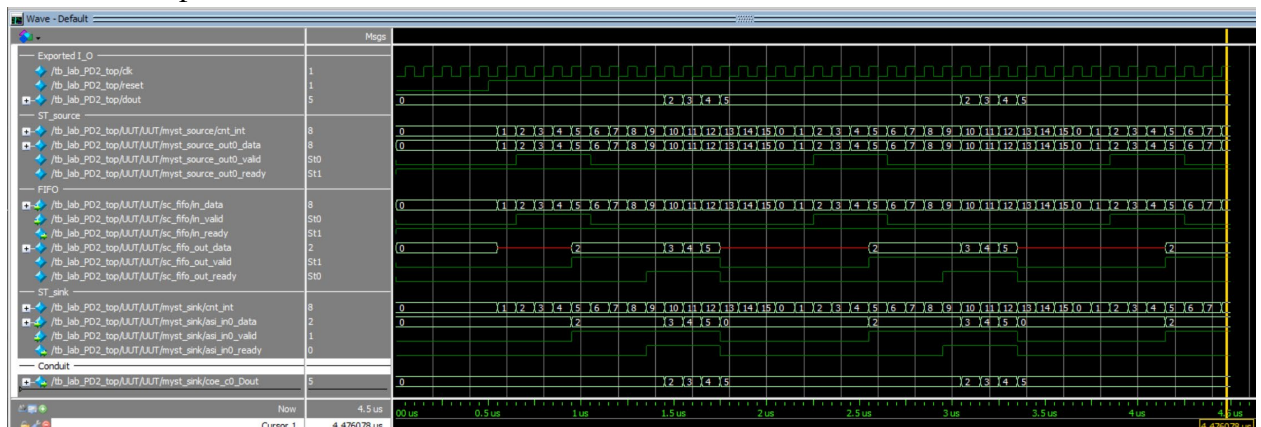


Рис. 27 – Моделирование проекта средствами ModelSim

Компонент ST_source:

Видим, что у ST_source идут данные (содержимое счётчика выводится в виде данных). Сигнал valid обрабатывает по 16 тактов за цикл, после чего сбрасывается и начинает отсчёт сначала.

Компонент ST_sink:

Аналогично, у ST_sink есть данные и сигнал, который показывает, что циклы размером до 16 тактов.

Компонент FIFO:

Сигнал in_ready все время показывает, что он находится в состоянии ready, поскольку он не заполнен (количество элементов, которые можем передать = 4, это мы задавали, когда настраивали компонент на Рис. 7). Соответственно, в какой-то момент получаем данные, при этом (с маленькой задержкой) сигнал in_valid становится = 1, тем самым показывая, что

он считывает данные. После получения 4 элементов он снова переходит в значение 0. Через какое-то время на `sc_fifo_out_valid` поступает сигнал 1, который показывает, что данные начали поступать и FIFO готов. Сигнал `sc_fifo_out_valid` будет = 1 и данные будут поступать до того момента, пока приёмник не скажет, что он готов `sc_fifo_out_ready = asi_in0_ready = 0`. Пока этот сигнал = 1 на `sc_fifo_out_data` будет поступать 4 порции данных (2, 3, 4, 5, эти данные поступят только когда сигнал `asi_in0_ready = 0`). После этого FIFO будет опустошён (красная линия на временной диаграмме). Ждём следующего периода, чтобы снова заполнить FIFO, ждём пока приёмник будет готов принимать и так далее...

3.2. Тестирование средствами Signal Tap II

Создание файла для отладки

Создадим файл `db_lab_PD2_top.sv` для отладки модуля `lab_PD2_top`:

```
lab_PD2 - db_lab_PD2_top.sv

1 module db_lab_PD2_top (
2     (* altera_attribute = "-name IO_STANDARD \"3.3-V LVC MOS\"", chip_pin = "23" *)
3     input bit clk
4 );
5     bit reset;
6     bit [3:0] dout;
7     SP_unit u0 (
8         .source (reset),
9         .source_clk (clk)
10    );
11     lab_PD2_top UUT (.*);
12 endmodule
13
```

Рис. 28 – Файл для отладки модуля верхнего уровня

Создадим модуль ISSPE, укажем файл `db_lab_PD3_top.sv` файлом верхнего уровня и убедимся в том, что схема, получаемая в результате компиляции, будет верной:

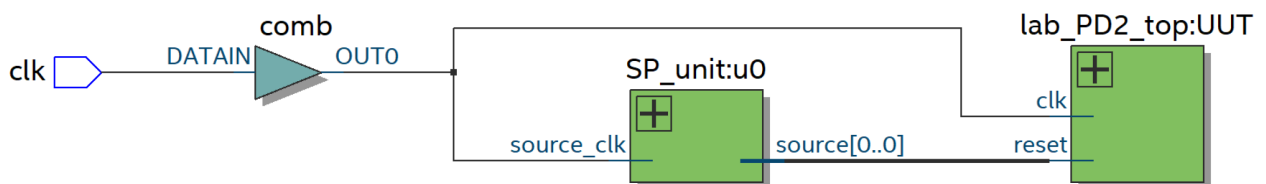


Рис. 29 – Схема проекта с добавлением SP_unit в RTL Viewer

Как видно из схемы `SP_unit` добавлен корректно.

Настройка Signal Tap II

Добавим и настроим Signal Tap II, чтобы произвести отладку на плате:

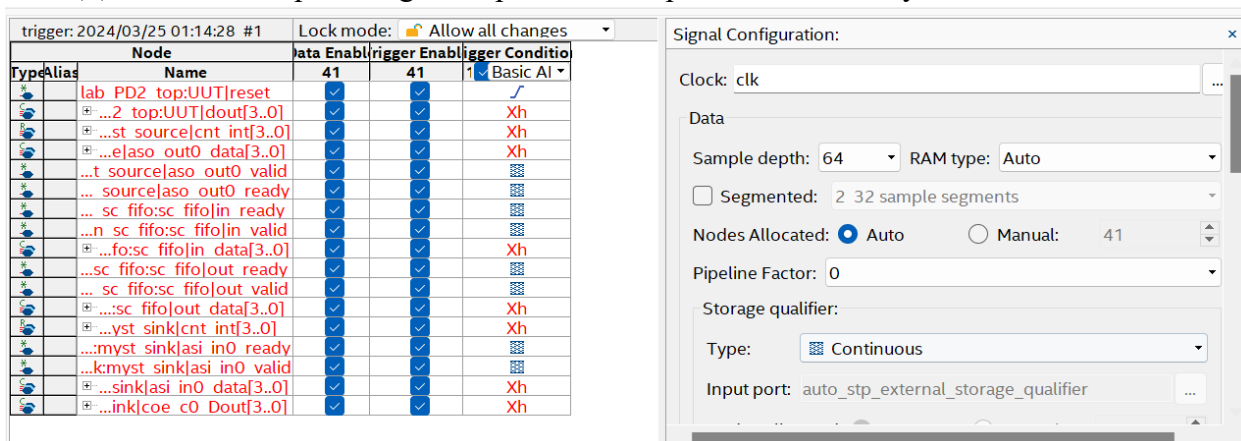


Рис. 30 – Настройка окна Signal Tap II

*Момент захвата данных синхронизируем по reset

Тестирование на плате средствами Signal Tap II

Выполним полную компиляцию. В отчете о компиляции видно, что устройство удовлетворяет временным параметрам.

Slow 1200mV 85C Model Fmax Summary				
<<Filter>>				
	Fmax	Restricted Fmax	Clock Name	Note
1	93.86 MHz	93.86 MHz	altera_reserved_tck	

Рис. 31 – Временные характеристики устройства

Теперь запустим и выполним проверку корректности работы программы на плате. Выполним загрузку разработанного модуля на плату и запустим тестирование:



Рис. 32 – Результат Signal Tap II

Полученная временная диаграмма совпадает с той, что была получена в ходе тестирования проекта средствами ModelSim (Рис. 27). Данные поступают и передаются на приёмник корректно.

Проверим, что в системных путях есть все необходимые файлы:

4. Вывод

В ходе лабораторной работы была успешна система, представленная на Рис. 1.

Система демонстрирует эффективную передачу данных между источником (ST_source) и приемником (ST_sink) с использованием буферизации через FIFO. Модуль ST_source функционирует как источник данных, генерируя выходные данные и уведомляя о их готовности через активацию сигнала aso_out0_valid. Сигнал aso_out0_valid периодически активируется, каждые 16 тактов, что указывает на появление новых данных для передачи.

С другой стороны, модуль ST_sink работает как приемник данных, готовый принимать данные только при наличии готовности от модуля ST_source. Он активизирует сигнал asi_in0_ready для уведомления об источнике о своей готовности принять данные и обновляет свой выходной сигнал soe_c0_Dout только при наличии действительных данных от ST_source.

Промежуточное звено, FIFO, обеспечивает буферизацию данных между источником и приемником. Сигнал in_valid активируется при поступлении новых данных для записи в FIFO, в то время как сигнал out_valid указывает на готовность FIFO к передаче данных приемнику. FIFO успешно передает данные в ST_sink только тогда, когда приемник готов принять их.

Таким образом, система обеспечивает синхронную и надежную передачу данных между источником и приемником, используя FIFO для управления потоком данных и согласования скоростей передачи.