

# Приложение Platform Designer

# Цели

Цели – научиться:

- Создавать системы используя Platform Designer (PD)
- Интегрировать существующие IP в систему используя PD
- Использовать систему, созданную в PD в пакете Intel® Quartus® Prime (QP)
- Разрабатывать и использовать компоненты (custom IP) при создании системы в PD
- Моделировать систему, созданную в PD, с помощью ModelSim (используя NativeLink QP)
- Отлаживать систему, созданную в PD, на плате с помощью InSystem Source&Probe Editor и SignalTapII пакета QP

# Три варианта пакета Intel® Quartus® Prime

## Intel® Quartus® Prime

Design Software

НЕ требуется лицензия

**Lite Edition (LE)**

Требуется лицензия

**Standard Edition (SE)**

Требуется лицензия

**Pro Edition (PE)**

Для выполнения лабораторных работ достаточно:

- QPLite
  - ModelSim Intel FPGA **Starter** Edition
- Версия пакета QP – любая, начиная с 16.0

[Сравнение версий на сайте Intel® FPGA](#)

# Приложение Platform Designer

## Часть 1

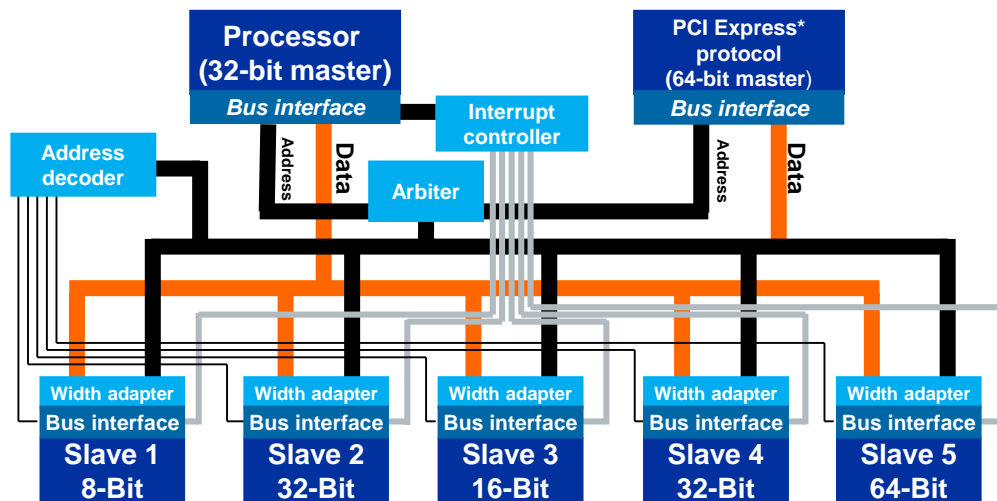
### Пользовательский интерфейс

# План

- О приложении PD
- Пользовательский интерфейс приложения PD
- Файлы, создаваемые PD, при генерации системы
- Место PD в процедуре проектирования пакета QP
- Лабораторная 1

# Традиционная процедура разработки системы

- Компоненты могут использовать разные интерфейсы (некоторые – стандартные, некоторые – нестандартные)
- Значительные усилия требуются для разработки системы коммуникации между компонентами системы.
- Интеграция блоков в систему требует длительной отладки



# PD: Автоматическое создание системы соединения

- Сокращает время разработки: автоматическое создание необходимых блоков и связей
- Позволяет избежать ошибок
- Позволяет сфокусироваться на разработке архитектуры системы и собственных блоков
- Повышает производительность труда разработчика.



# Преимущества использования PD

- Упрощает создание сложных систем: автоматизирует процесс создания системы межсоединений
- Позволяет работать на более высоком уровне абстракции: на уровне системы и связей в системе, а не на уровне сигналов
- Инструмент для интеграции в систему: стандартных IP, пользовательских IP, процессорных элементов, средств моделирования
- Позволяет использовать созданные системы как компоненты (иерархическое проектирование на системном уровне)
- Позволяет сократить время разработки и упростить верификацию



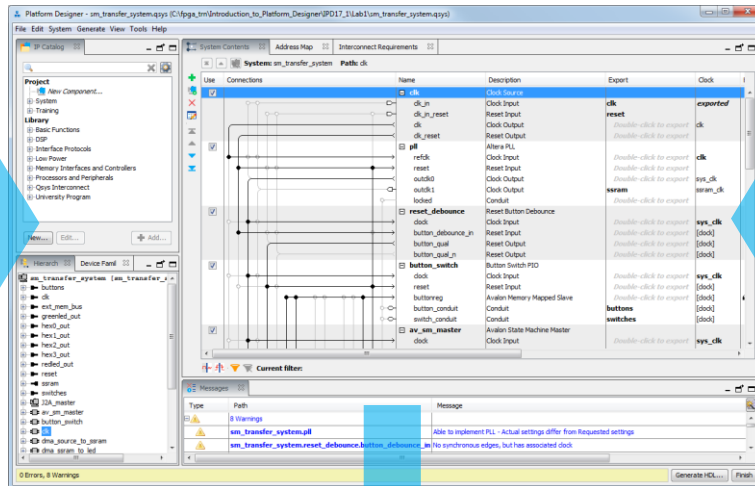
# Простота использования PD



## Catalog of available IP

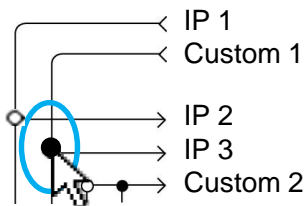
- Interface protocols
- Memory
- DSP
- Embedded
- Bridges
- PLL
- Custom systems

Библиотеки  
компонентов



HDL

## Connect custom IP and systems



Графический  
интерфейс для  
создания системы

Автоматические: интеграция и создание HDL описания

# Конечные приложения для использования PD

PD может использоваться при создании любого проекта для FPGA

- Два типа взаимодействия в цифровой системе
  - Управление системой (**Control plane**)
    - Адресный доступ (**Memory-mapped**)
    - Чтение и запись регистров управления и статуса
  - Передача данных (**Data plane**)
    - Поточковая передача (**streaming data transfer**): высокоскоростная, точка-точка
    - Адресная передача (**Memory-mapped**)
- Приложения:
  - ЦОС
  - Обработка видео потоков
  - Высокоскоростные интерфейсы
  - ....

# Инициаторы взаимодействия в системах PD

Для инициации взаимодействия система, создаваемая в PD, не требует использования процессора

- Для инициирования передачи данных и управления системой могут использоваться существующие IP, пользовательские компоненты, внешние процессоры (контроллеры).
  - Управление системой:
    - компоненты с адресным доступом (Memory-mapped ) используют master → slave взаимодействие
      - Примеры: конечный автомат, модули прямого доступа к памяти - direct memory access (DMA)
  - Передача данных:
    - компоненты с адресным доступом (Memory-mapped ) используют master → slave взаимодействие
    - компоненты с потоковой передачей (streaming data) используют uses source → sink взаимодействие
    - Примеры: поток данных от видео камеры, АЦП...

# Использование процессоров в системах PD

Система, содержащая встроенный ARM (Hard Processor System (HPS)) или soft-core процессор (Intel Nios II), требует использования PD

- Встроенные Hard-core процессор Arm (Cortex-A9) или soft-core процессор NiosII могут подключаться к системе, создаваемой в PD для:
  - Управления
    - используют master → slave взаимодействие
  - Передачи данных
    - используют master → slave взаимодействие

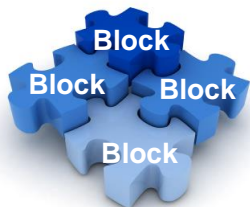
# PD - средство проектирования систем

Уровень абстракции описания и производительность при создании системы

Low

Medium

High

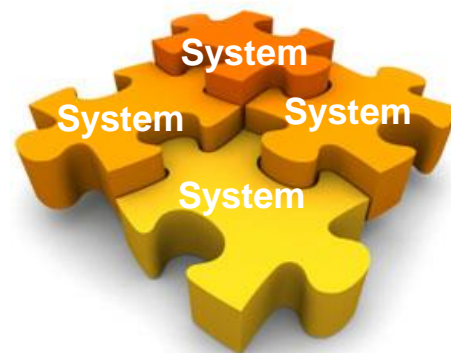


**Интеграция блоков**



**IP интеграция**

- Разработка на основе IPs
- повторное использование IP
- верификация IP



**Создание системы**

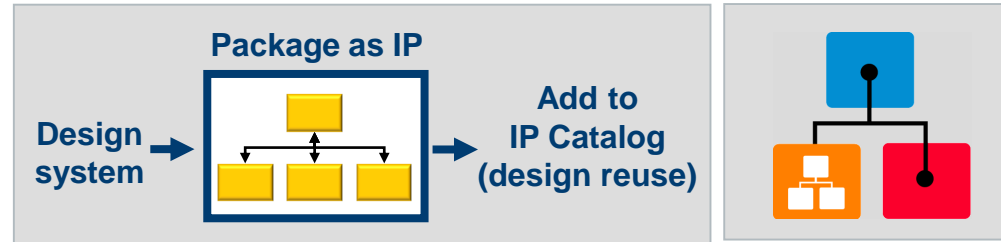
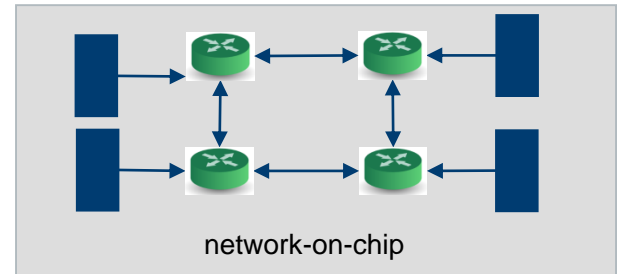
- Иерархические системы
- повторное использование
- верификация

**Схемные редакторы и HDL описания**

**Platform Designer**

# Особенности PD

- Высокоскоростные каналы соединения компонентов системы
- «Управление» IP
- Иерархическое проектирование систем
- Поддержка стандартных интерфейсов



Intel® FPGA	Avalon® interface
Arm*	Arm AMBA* AXI interface

# Использование созданных систем и компонентов

PD позволяет повторно использовать (re-use) созданные компоненты и системы



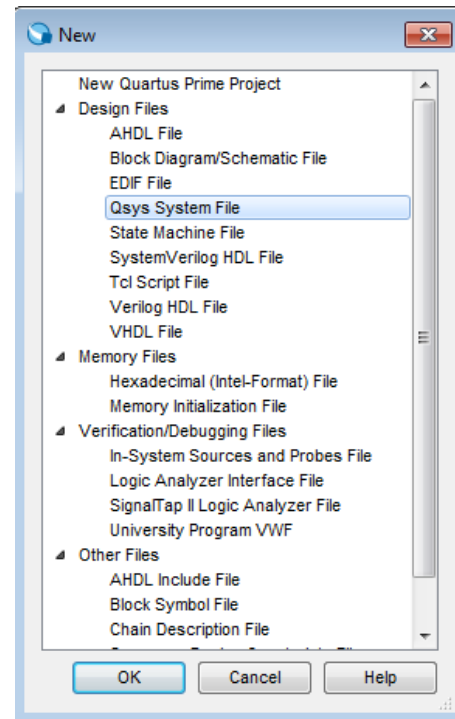
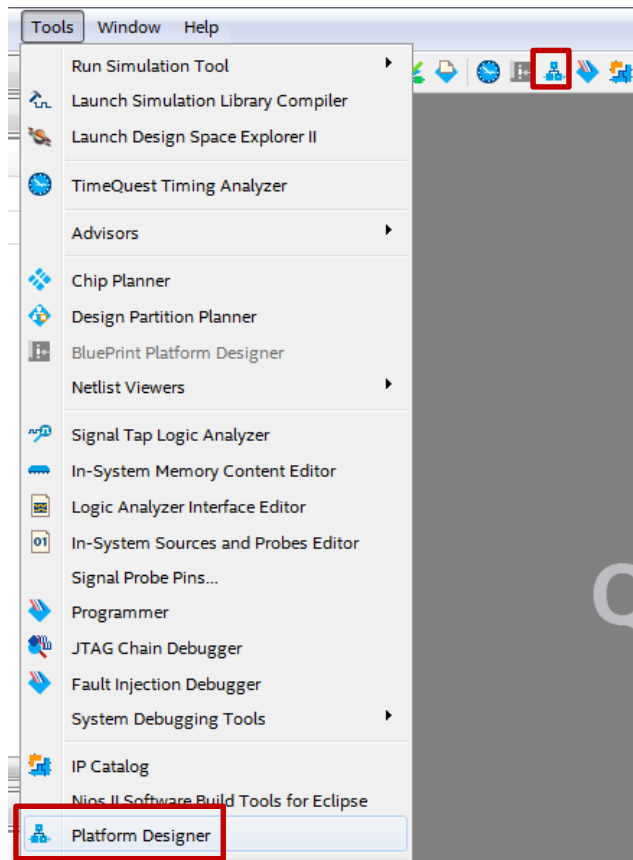
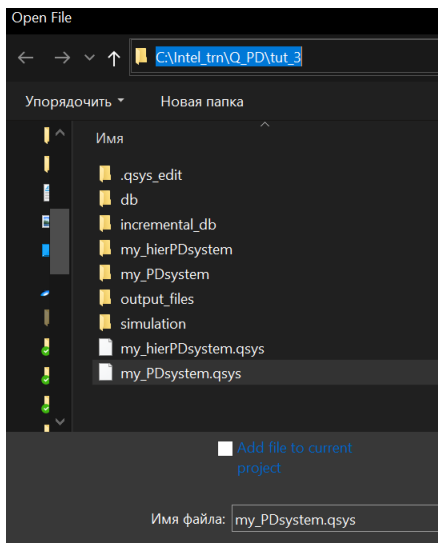
# План

- О приложении PD
- Пользовательский интерфейс приложения PD
- Файлы, создаваемые PD, при генерации системы
- Место PD в процедуре проектирования пакета QP
- Лабораторная 1



# Как запустить PD

1. QP меню Tools
2. Создать новый файл Platform Designer (.qsys)
3. Открыть существующий файл (.qsys) из пакета Quartus



# Пользовательский интерфейс PD

Отсоединяемые закладки  
(Detachable Tabs)

Platform Designer - my\_PDsystem.qsys\* (C:\Intel\_trn\Q\_PD\tut\_3\my\_PDsystem.qsys)

File Edit System Generate View Tools Help

IP Catalog

Project

- New Component...
- System
- Training
- Library
  - Basic Functions
  - DSP
  - Interface Protocols
  - Low Power
  - Memory Interfaces and Controllers
  - Processors and Peripherals
  - Qsys Interconnect
  - Tri-State Components
  - University Program

Библиотека IP

New... Edit... Add...

Hierarchy

Device Family

- my\_PDsystem [my\_PDsystem.qsys\*]
  - clk
  - dout\_a
  - dout\_b
  - reset
  - clk\_0
  - my\_master\_0
    - clock
    - conduit\_end\_0
    - reset
    - s0
  - my\_slave\_0
    - conduit\_end\_0

Иерархия системы

System Contents

Address Map

Interconnect Requirements

System: my\_PDsystem Path: my\_slaveWS\_0.clock

Use	Connec...	Name	Description	Export	Clock	Base	End
<input checked="" type="checkbox"/>		clk_0	Clock Source	clk	exported		
<input checked="" type="checkbox"/>		clk_in	Clock Input	Double-click to	clk_0		
<input checked="" type="checkbox"/>		clk_in_reset	Reset Input	Double-click to	[clock]		
<input checked="" type="checkbox"/>		clk	Clock Output	Double-click to	[clock]		
<input checked="" type="checkbox"/>		clk_reset	Reset Output	Double-click to	[clock]		
<input checked="" type="checkbox"/>		my_master_0	my_master	Double-click to	clk_0		
<input checked="" type="checkbox"/>		clock	Clock Input	Double-click to	[clock]		
<input checked="" type="checkbox"/>		reset	Reset Input	Double-click to	[clock]		
<input checked="" type="checkbox"/>		m0	Avalon Memory Mapped ...	Double-click to	clk_0		
<input checked="" type="checkbox"/>		my_slave_0	my_slave	Double-click to	clk_0		
<input checked="" type="checkbox"/>		clock	Clock Input	Double-click to	[clock]		
<input checked="" type="checkbox"/>		reset	Reset Input	Double-click to	[clock]		
<input checked="" type="checkbox"/>		s0	Avalon Memory Mapped ...	Double-click to	clk_0		
<input checked="" type="checkbox"/>		conduit_end_0	Conduit	Double-click to	[clock]		
<input checked="" type="checkbox"/>		my_slaveWS_0	my_slaveWS	Double-click to	unconnected		
<input checked="" type="checkbox"/>		clock	Clock Input	Double-click to	[clock]		
<input checked="" type="checkbox"/>		reset	Reset Input	Double-click to	[clock]		
<input checked="" type="checkbox"/>		s0	Avalon Memory Mapped ...	Double-click to	[clock]		
<input checked="" type="checkbox"/>		conduit_end_0	Conduit	Double-click to	[clock]		

System Contents (структура системы)

Current filter:

Parameters

System: my\_PDsystem Path: my\_slaveWS\_0.clock

Clock Input

clock\_sink

Details

Parameters

Clock rate: 0

Параметры компонента

Messages

Type	Path	Message
Error	1 Error	
Error	my_PDsystem.my_slaveWS_0	my_slaveWS_0.clock must be connected to a clock output
Warning	1 Warning	
Warning	my_PDsystem.my_slaveWS_0	my_slaveWS_0.s0 must be connected to an Avalon-MM master

Сообщения (info, warning, error)

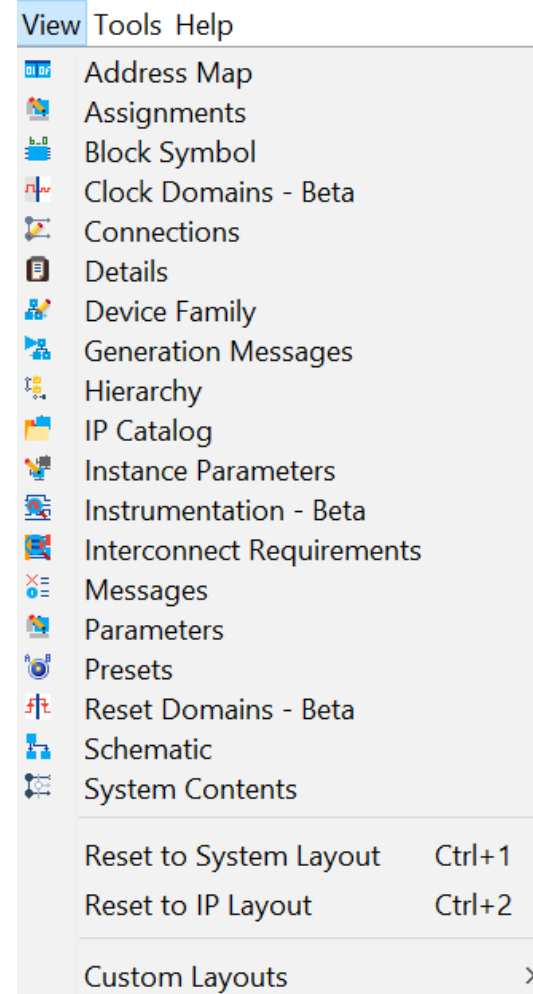
1 Error, 1 Warning

Generate HDL... Finish

# Меню View

## Управление раскладкой пользовательского интерфейса

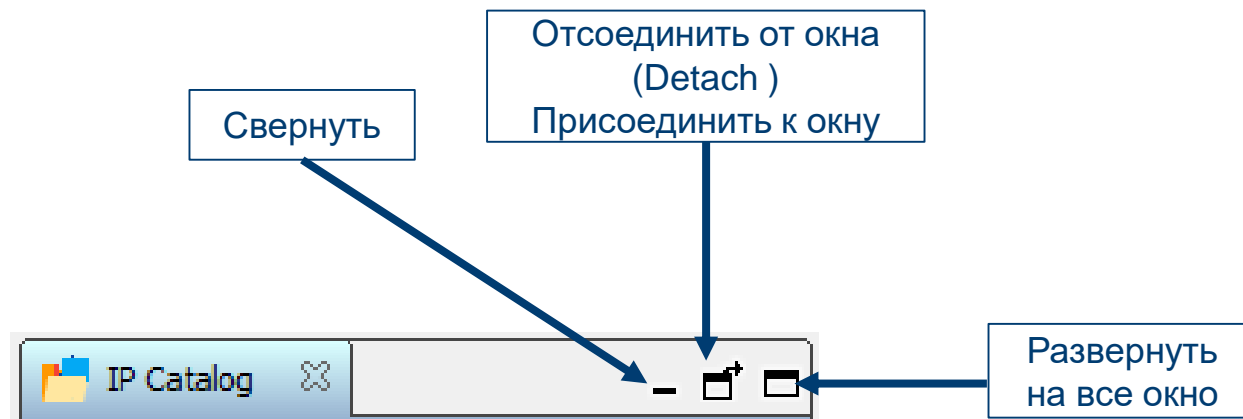
- Все элементы интерфейса организованы в виде закладок, доступных из меню View
  - Для сброса к базовому виду: меню View → Reset to System Layout
- Элементы, выбранные на одной закладке, определяют то, что отображается на других закладках
- Создание пользовательских раскладок
  - меню View → Custom layouts
  - импорт/экспорт раскладок: тип файла - .layout



# Управление закладками (Tab)

Все закладки поддерживают возможность изменения положения и размера

- Закладки отсоединяются/присоединяются к окну приложения
- Закладки можно свернуть/развернуть на все окно



# Библиотека IP

Библиотека проекта

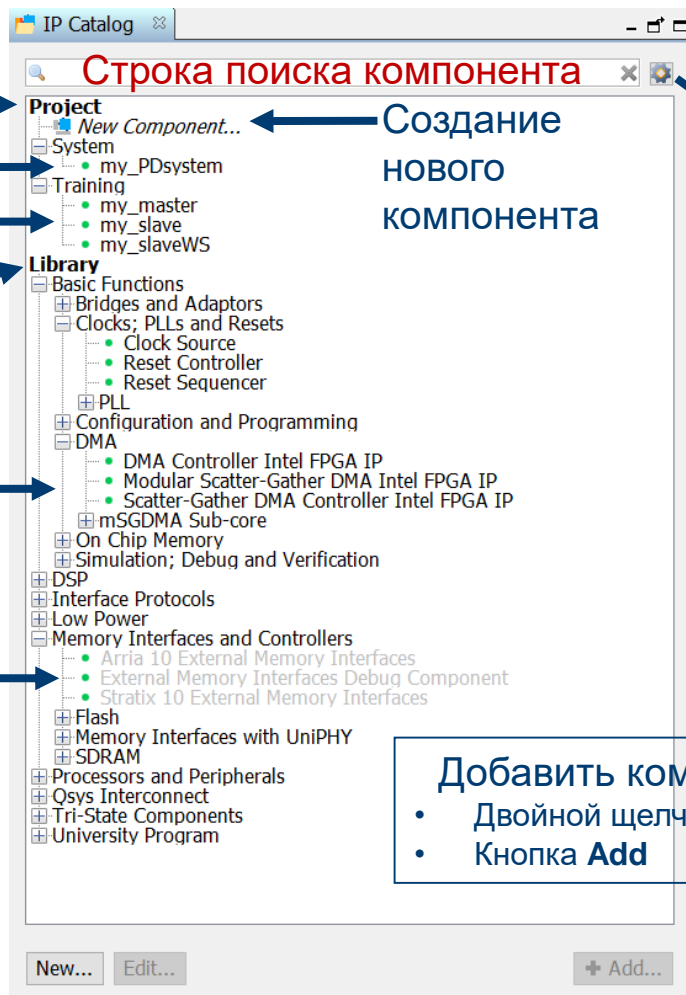
Библиотека созданных систем проекта

Пользовательские компоненты проекта

Библиотека пакета QP

Компоненты, которые  
**МОЖНО** использовать

Компоненты, которые  
**НЕЛЬЗЯ** использовать



Добавить компонент к системе:

- Двойной щелчок или
- Кнопка Add

Опции  
фильтрации

# Закладка структура системы (System Contents Tab)

## Компоненты и подсистемы

Адреса

IP Catalog

Project

- New Component...
- System
  - my\_hierPDsystem
  - my\_PDsystem
- Training
  - my\_master
  - my\_slave
  - my\_slaveWS

Library

- Basic Functions
- DSP
- Interface Protocols
- Low Power
- Memory Interfaces and Controllers
- Processors and Peripherals
- Qsys Interconnect
- Tri-State Components
- University Program

System: my\_hierPDsystem Path: PDsystem\_0

Use	Connect	Name	Description	Export	Clock	Base	End
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	clk_0	Clock Source	clk	exported		
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	clk_in	Clock Input	reset	clk_0		
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	clk_in_reset	Reset Input	clk			
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	clk	Clock Output	clk_reset			
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	clk_reset	Reset Output				
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	PDsystem_0	my_PDsystem				
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	clk	Clock Input	dout_a	clk_0		
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	dout_a	Conduit	dout_b	[clk]		
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	dout_b	Conduit	reset	[clk]		
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	reset	Reset Input				
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	my_master_0	my_master				
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	clock	Clock Input		clk_0		
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	reset	Reset Input		[clock]		
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	m0	Avalon Memory Mapped ...		[clock]		
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	my_slave_0	my_slave				
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	clock	Clock Input		clk_0		
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	reset	Reset Input		[clock]		
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	s0	Avalon Memory Mapped ...		[clock]		
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conduit_end_0	Conduit	dout2	[clock]	0x0000_0004	0x0000_0004
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	my_slaveWS_0	my_slaveWS				
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	clock	Clock Input		clk_0		
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	reset	Reset Input		[clock]		
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	s0	Avalon Memory Mapped ...		[clock]	0x0000_0000	0x0000_0003
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conduit_end_0	Conduit	dout1	[clock]		

Current filter:

Подключения

Внешние выводы

# Закладка структура системы (System Contents Tab)

## ■ Инструменты управления



Добавить компонент



Добавить подсистему



Удалить компонент



Настроить компонент



Переместить вверх



Переместить вверх на один шаг



Переместить вниз на один шаг



Переместить вниз

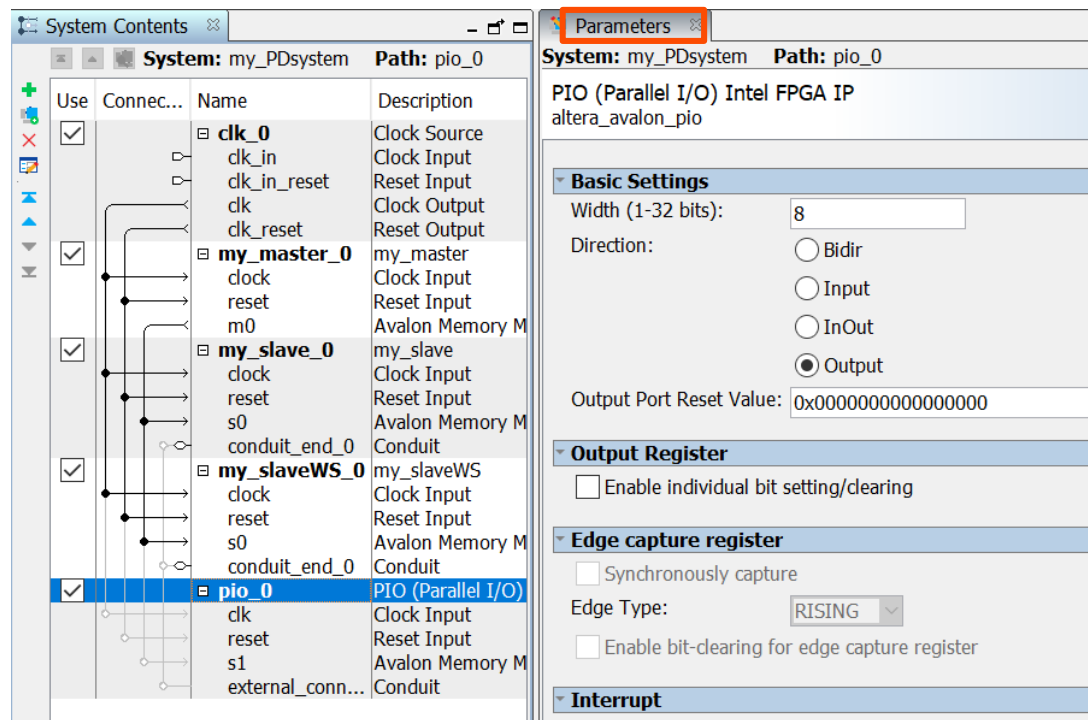
Use	Connections	Name	Description
<input checked="" type="checkbox"/>		clk	Clock Source
<input checked="" type="checkbox"/>		clk_in	Clock Input
<input checked="" type="checkbox"/>		clk_in_reset	Reset Input
<input checked="" type="checkbox"/>		clk	Clock Output
<input checked="" type="checkbox"/>		clk_reset	Reset Output
<input checked="" type="checkbox"/>		pll	Altera PLL
<input checked="" type="checkbox"/>		refclk	Clock Input
<input checked="" type="checkbox"/>		reset	Reset Input
<input checked="" type="checkbox"/>		outclk0	Clock Output
<input checked="" type="checkbox"/>		outclk1	Clock Output
<input checked="" type="checkbox"/>		locked	Conduit
<input checked="" type="checkbox"/>		reset_debounce	Reset Button Debounce
<input checked="" type="checkbox"/>		clock	Clock Input
<input checked="" type="checkbox"/>		button_debounce_in	Reset Input
<input checked="" type="checkbox"/>		button_qual	Reset Output
<input checked="" type="checkbox"/>		button_qual_n	Reset Output
<input checked="" type="checkbox"/>		button_switch	Button Switch PIO
<input checked="" type="checkbox"/>		clock	Clock Input
<input checked="" type="checkbox"/>		reset	Reset Input
<input checked="" type="checkbox"/>		buttonreg	Avalon Memory Mapped Slave
<input checked="" type="checkbox"/>		button_conduit	Conduit
<input checked="" type="checkbox"/>		switch_conduit	Conduit
<input checked="" type="checkbox"/>		av_sm_master	Avalon State Machine Master
<input checked="" type="checkbox"/>		clock	Clock Input
<input checked="" type="checkbox"/>		reset	Reset Input
<input checked="" type="checkbox"/>		avalon_master	Avalon Memory Mapped Master

Удалить компонент – выделить и нажать **Delete**

# Закладка Параметры (Parameter Tab)

При добавлении компонента к структуре системы (System Contents tab):

- Компонент помещается под всеми добавленными ранее компонентами
- Открывается окно настройки параметров компонента
- Как можно настроить ранее добавленные компоненты:
  - Двойным щелчком по имени компонента - открывается закладка настройки параметров
  - Или щелкнуть правой клавишей мыши и выбрать команду Edit





# Интерфейсы компонента

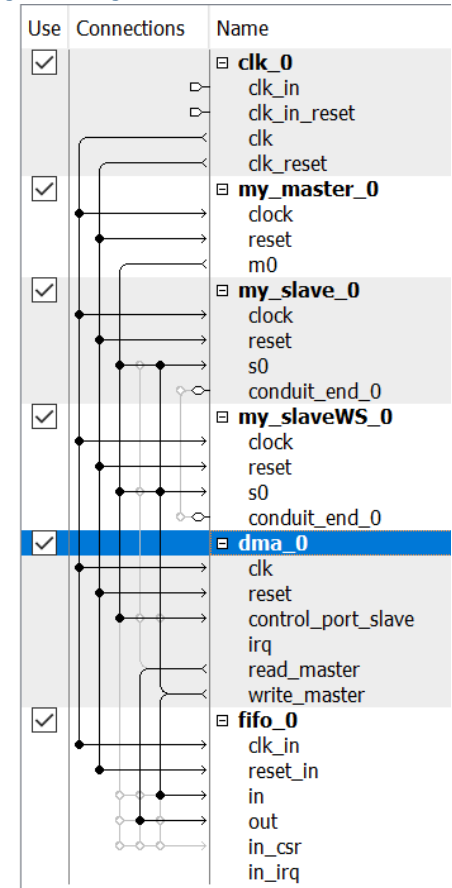
- Интерфейс: группа из одного или нескольких сигналов, которые могут быть подключены к другим интерфейсам в системе
- Ссылка на интерфейс
  - Пример: `source.s1` интерфейс `s1` (Avalon MM slave) компонента `source`
- Сигналы `clk` и `reset` – два отдельных интерфейса

	[-] <b>dma_source_to_ssr...</b>	DMA Controller
→	clk	Clock Input
→	reset	Reset Input
→	control_port_slave	Avalon Memory Mapped Slave
	irq	Interrupt Sender
←	read_master	Avalon Memory Mapped Master
←	write_master	Avalon Memory Mapped Master
	[-] <b>source</b>	On-Chip Memory (RAM or ROM)
→	clk1	Clock Input
→	s1	Avalon Memory Mapped Slave
→	reset1	Reset Input

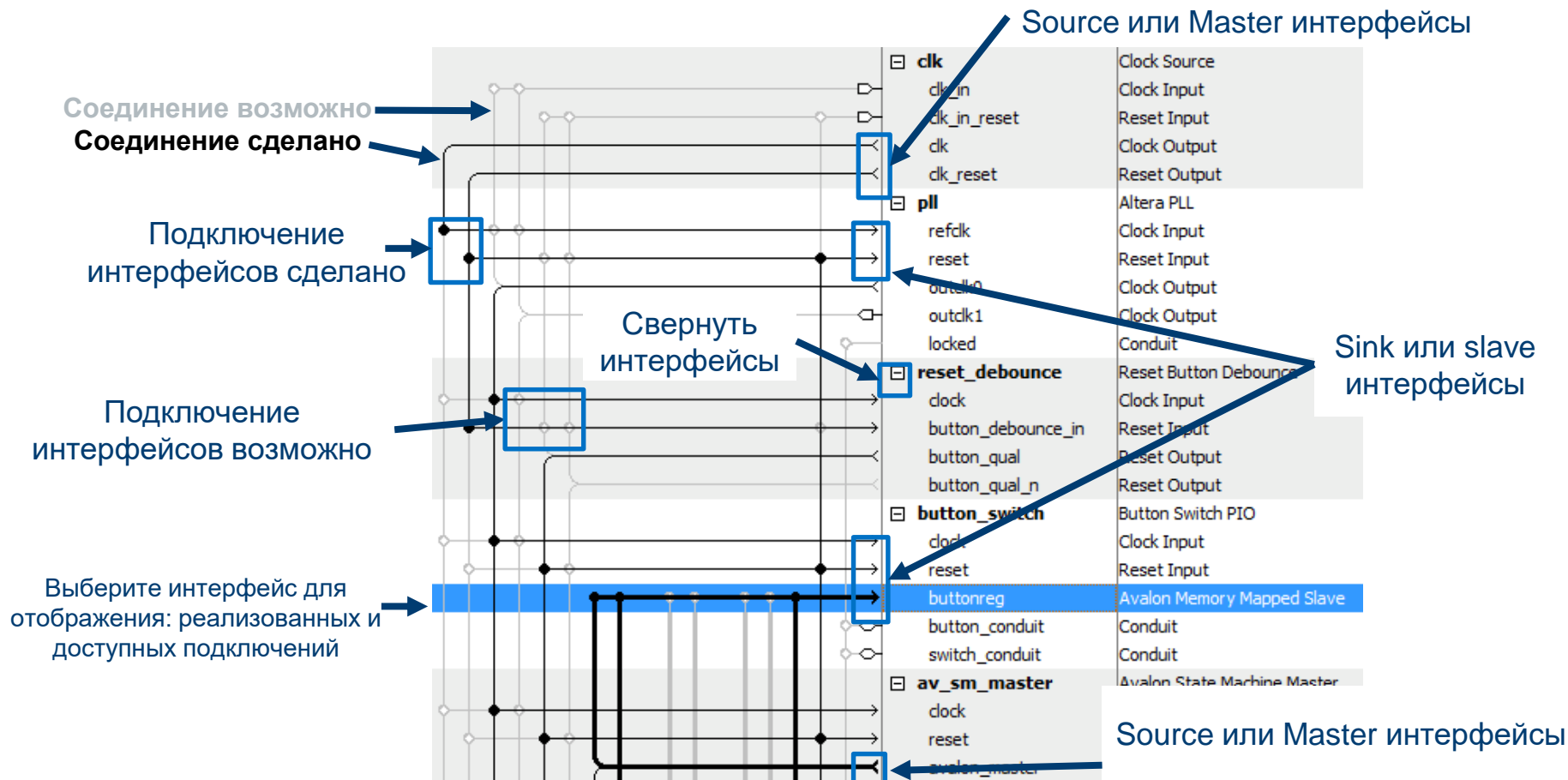
# Панель коммутации Connections (Структура системы)

Колонка Connections – коммутационная панель для соединения компонентов на закладке Структура Системы

- Позволяет соединять только совместимые интерфейсы
  - Clock и reset источники (sources) => к входам компонентов
  - Masters => к slaves (для Avalon MM)
  - Sources => к sinks (для Avalon ST)
- Черные точки – реализованные подключения
- Серые точки – допустимые подключения
- PD генерирует систему связей на основе выполненных подключений

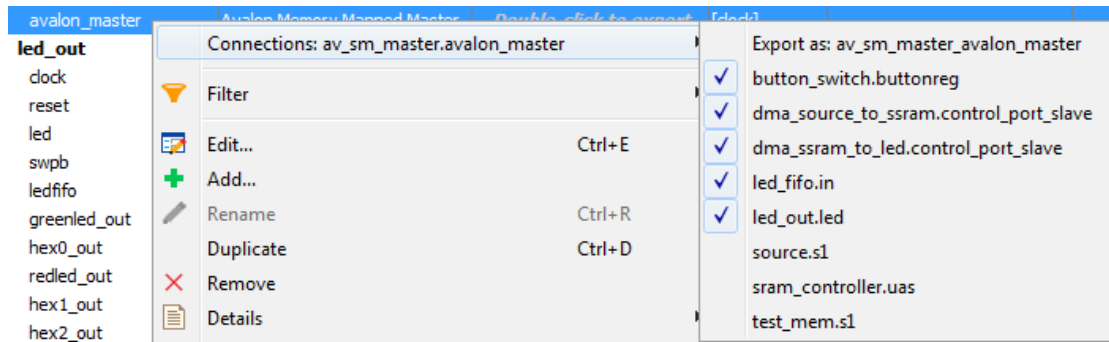


# Особенности Панели коммутации (Connections )

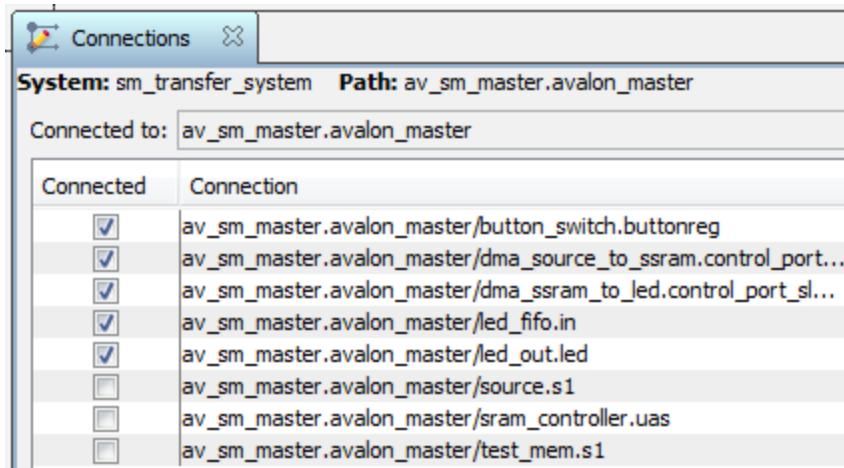


# Другие способы подключения интерфейсов

Right-click на любом  
интерфейсе →  
**Connections**



Выберите интерфейс  
в окне структуры  
системы (**System Contents**).  
Затем выполните команду  
меню **View** → **Connections**



## Экспортирование интерфейса

- Позволяет подключить интерфейсы к компонентам/элементам вне системы, создаваемой в PD
  - Любой интерфейс может быть экспортирован
  - Интерфейс может быть экспортирован ИЛИ подключен в системе
- Экспорт интерфейса необходим для подключения к выводам FPGA

Name	Description	Export
<b>clk</b>	Clock Source	<b>clk</b>
clk_in	Clock Input	<b>reset</b>
clk_in_reset	Reset Input	<i>Double-click to export</i>
clk	Clock Output	<i>Double-click to export</i>
clk_reset	Reset Output	
<b>pll</b>	Altera PLL	
refclk	Clock Input	<i>Double-click to export</i>
reset	Reset Input	<i>Double-click to export</i>
outclk0	Clock Output	<i>Double-click to export</i>
outclk1	Clock Output	
locked	Conduit	<b>ssram</b>
<b>reset_debounce</b>	Reset Button Debounce	<i>Double-click to export</i>
clock	Clock Input	<i>Double-click to export</i>
button_debounce_in	Reset Input	<i>Double-click to export</i>
button_qual	Reset Output	<i>Double-click to export</i>
button_qual_n	Reset Output	<i>Double-click to export</i>
<b>button_switch</b>	Button Switch PIO	
clock	Clock Input	<i>Double-click to export</i>
reset	Reset Input	<i>Double-click to export</i>
buttonreg	Avalon Memory Mapped Slave	<i>Double-click to export</i>
button_conduit	Conduit	
switch_conduit	Conduit	<b>buttons</b>
		<b>switches</b>

Экспортированный  
интерфейс  
отображается как  
вывод

- Как экспортировать: Double-click и ввести имя
- Как убрать экспортирование: удалить имя

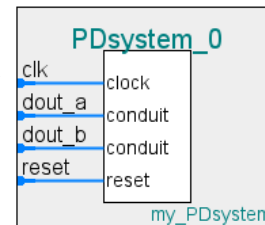
# Тактовый сигнал (Clock)

Все операции в системе синхронны, поэтому необходим тактовый сигнал

- Каждый новый .qsys файл (новая система) по умолчанию включает компонент Clock Source
  - Этот компонент опциональный и может быть удален
  - Компонент соединяет два интерфейса
    - Экспортированный интерфейс Clock, приходящий из вне проектируемой системы (Clock Input)
    - Source интерфейс (Clock Output), подключаемый внутри системы к тактовым входам (sink)

## КОМПОНЕНТОВ

Use	Connec...	Name	Description	Export	Clock
<input checked="" type="checkbox"/>		<b>clk_0</b>	Clock Source		
		clk_in	Clock Input	clk	<i>exported</i>
		clk_in_reset	Reset Input	reset	
		clk	Clock Output	<i>Double-click to</i>	clk_0
		clk_reset	Reset Output	<i>Double-click to</i>	
<input checked="" type="checkbox"/>		<b>my_master_0</b>	my_master		
		clock	Clock Input	<i>Double-click to</i>	clk_0
		reset	Reset Input	<i>Double-click to</i>	[clock]
		m0	Avalon Memory Mapped Master	<i>Double-click to</i>	[clock]
<input checked="" type="checkbox"/>		<b>my_slave_0</b>	my_slave		
		clock	Clock Input	<i>Double-click to</i>	clk_0
		reset	Reset Input	<i>Double-click to</i>	[clock]
		s0	Avalon Memory Mapped Slave	<i>Double-click to</i>	[clock]
		conduit_end_0	Conduit	dout_a	[clock]



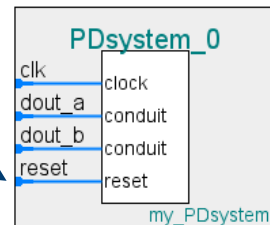
Колонка Clock  
упрощает подключение  
к тактовым сигналам

# Управление сбросом системы (Reset)

PD позволяет осуществлять управление сбросом системы

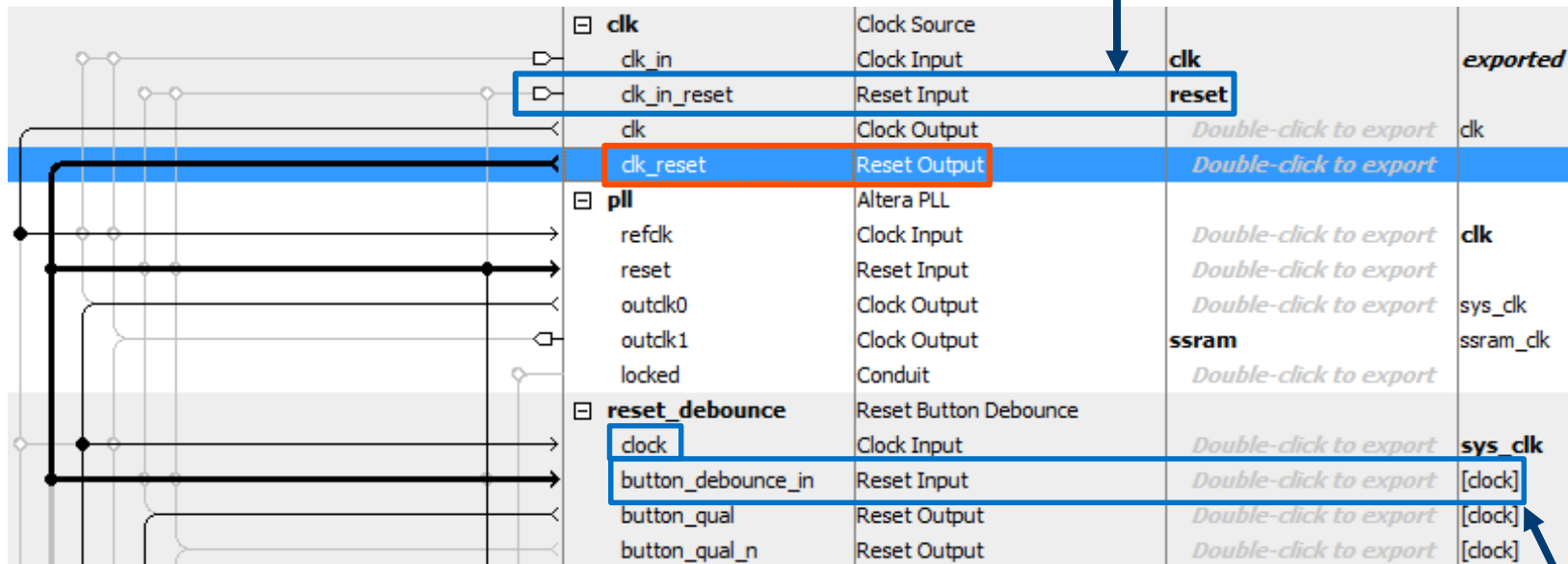
- Можно сбрасывать всю систему или отдельные компоненты системы
- Сигнал Reset – отдельный интерфейс
  - Вход (sink) привязан к clock интерфейсу и синхронизируется им
- Система может содержать несколько сигналов сброса
- IP модули Reset Controller и Reset Sequencer позволяют управлять сбросами системы

Use	Connec...	Name	Description	Export	Clock
<input checked="" type="checkbox"/>		<b>clk_0</b>	Clock Source		
		clk in	Clock Input	clk	exported
		clk in reset	Reset Input	reset	
		clk	Clock Output	Double-click to	clk_0
		clk_reset	Reset Output	Double-click to	
<input checked="" type="checkbox"/>		<b>my_master_0</b>	my_master		
		clock	Clock Input	Double-click to	clk_0
		reset	Reset Input	Double-click to	[clock]
		m0	Avalon Memory Mapped Master	Double-click to	[clock]
<input checked="" type="checkbox"/>		<b>my_slave_0</b>	my_slave		
		clock	Clock Input	Double-click to	clk_0
		reset	Reset Input	Double-click to	[clock]
		s0	Avalon Memory Mapped Slave	Double-click to	[clock]
		conduit_end_0	Conduit	dout_a	[clock]



# Подключение сигнала сброса (Reset)

Компонент **Clock Source** подключает  
внешний сигнал Сброса (Reset) – экспортированный вход



Автоматическое подключение сигналов сброса  
Меню **System** → **Create Global Reset Network**

Вход Reset синхронизируется  
сигналом **clock**

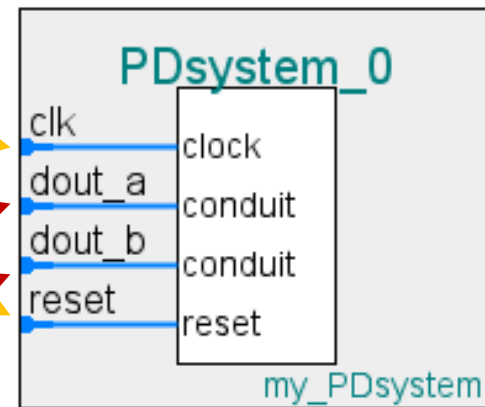


# Интерфейс Conduit

Компоненты используют интерфейс Conduit для сигналов, которые не соответствуют стандартным интерфейсам

- Примеры стандартных интерфейсов: Avalon\_MM, Avalon\_ST, Arm\* AXI
- В пользовательских компонентах разработчик должен самостоятельно определить сигналы интерфейса Conduit
- Чаще всего интерфейс Conduit используется для экспортирования выводов.

Use	Connec...	Name	Description	Export	Clock
<input checked="" type="checkbox"/>		<b>clk_0</b>	Clock Source		
		clk_in	Clock Input	clk	
		clk_in_reset	Reset Input	reset	
		clk	Clock Output	<i>Double-click to</i>	
		clk_reset	Reset Output	<i>Double-click to</i>	
<input checked="" type="checkbox"/>		<b>my_master_0</b>	my_master		
		clock	Clock Input	<i>Double-click to</i>	
		reset	Reset Input	<i>Double-click to</i>	
		m0	Avalon Memory Mapped Master	<i>Double-click to</i>	
<input checked="" type="checkbox"/>		<b>my_slave_0</b>	my_slave		
		clock	Clock Input	<i>Double-click to</i>	
		reset	Reset Input	<i>Double-click to</i>	
		s0	Avalon Memory Mapped Slave	<i>Double-click to</i>	
		conduit_end_0	Conduit	<i>Double-click to</i>	
<input checked="" type="checkbox"/>		<b>my_slaveWS_0</b>	my_slaveWS		
		clock	Clock Input	<i>Double-click to</i>	
		reset	Reset Input	<i>Double-click to</i>	
		s0	Avalon Memory Mapped Slave	<i>Double-click to</i>	
		conduit_end_0	Conduit	<i>Double-click to</i>	



# Адресация Memory-Mapped (ММ)

Каждый MM master интерфейс имеет собственное адресное пространство

- Максимальный размер адресного пространства определяется разрядностью адреса - 64
- Карта памяти (Memory Map) каждого MM Master интерфейса формируется:
  - Базовым адресом подключенных ведомых (slave)
  - Диапазоном адресов (address spans) подключенных ведомых (slave)
    - *Диапазоны адресов подключенных ведомых (slave) не должны пересекаться*
- Не перекрывающиеся базовые адреса назначаются:
  - Либо вручную; либо командой: меню System → Assign Base Addresses

Master интерфейс		Зафиксировать адрес		Карта памяти (0x0 to 0x6f)	
+	av_sm_master	Avalon State Machine Master	sys_clk		
+	led_fifo	On-Chip FIFO Memory	sys_clk	0x00000000	0x00000007
+	dma_ssramm_to_led	DMA Controller	sys_clk	0x00000020	0x0000003f
+	dma_source_to_ssram	DMA Controller	sys_clk	0x00000040	0x0000005f
+	start_pushbutton	PIO (Parallel I/O)	sys_clk	0x00000060	0x0000006f

# Закладка Карта Памяти (Address Map Tab)

Закладка Address Map tab - средство анализа и управление адресацией:

- Отображает адреса для реализованных подключений MM Master => MM Slave
- Для редактирования ячейки - Double-click ячейку
- Поддерживает адресацию для разделяемых ведомых (shared slaves)
  - Разделяемый ведомый может иметь разные адреса для подключённых ведущих

System Contents Address Map Interconnect Requirements

System: sm\_transfer\_system Path: test\_mem

	J2A_master.master	av_sm_master.avalon_master	dma_ssram_to_led.write_master
button_switch.buttonreg	0x0000_1040 - 0x0000_104f	0x0000_1040 - 0x0000_104f	
dma_source_to_ssram.control_port_sl...		0x0000_0040 - 0x0000_005f	
dma_ssram_to_led.control_port_slave		0x0000_0020 - 0x0000_003f	
led_fifo.in	0x0000_1050 - 0x0000_1057	0x0000_0000 - 0x0000_0007	0x0000_0000 - 0x0000_0007
led_out.led	0x0000_1030 - 0x0000_103f	0x0000_1030 - 0x0000_103f	
source.s1			
ssram_controller.uas			
test_mem.s1			0x0010_00

**Slave интерфейсы**

**Master интерфейсы**

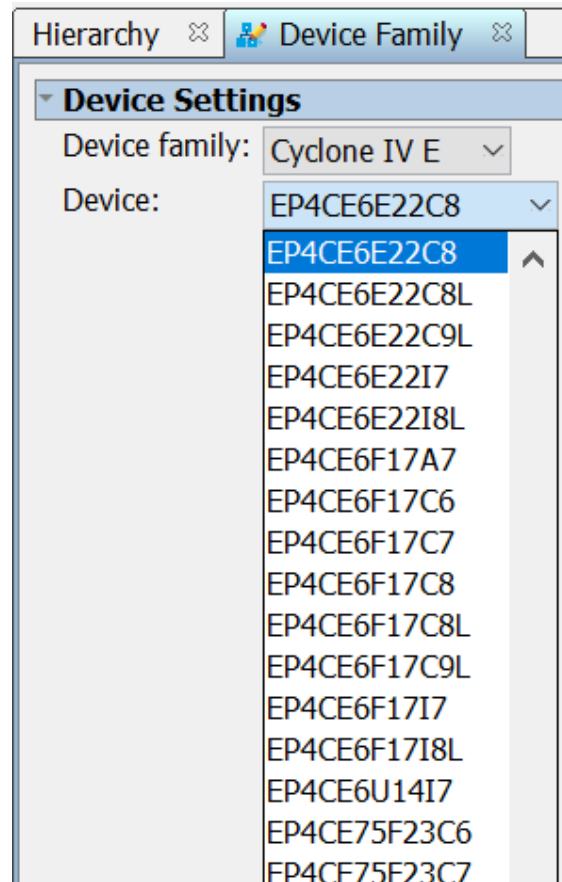
**led\_fifo**

Interface	Properties	Attributes	Mode
clk_in	Avalon FIFO Memory		
reset_in	Clock Input		
in	Reset Input		
	Avalon Memory Mapped Slave		
out	Avalon Streaming Source	mixed	mixed

# Закладка Семейство микросхем (Device Family)

Закладка позволяет:

- Отобразить назначенное семейство и тип микросхемы FPGA
- Изменить\задать семейство и тип микросхемы FPGA



# Закладка Interconnect Requirements Tabs

Позволяет задать общие требования к системе межсоединений

Установить число уровней конвейеризации

Установка дополнительных требований и инструментов Intel Quartus Prime software handbook описывает детали

System Contents ⌵ Address Map ⌵ Interconnect Requirements ⌵

Configure interconnect requirements for the system or an interface.

**System-wide Requirements**

Limit interconnect pipeline stages to: 2

Clock crossing adapter type: Handshake

**All Requirements**

Identifier	Setting	Value
\$system	Clock crossing adapter type	Handshake
\$system	Limit interconnect pipeline stages to	2
\$system	Enable instrumentation	TRUE
av_sm_master.avalon_master	Add performance monitor	TRUE
button_switch.buttonreg	Add performance monitor	TRUE
button_switch.buttonreg	Security	Non-secure
led_fifo.in	Add performance monitor	TRUE
led_out.led	Add performance monitor	TRUE
led_out.swpb	Add performance monitor	TRUE
source.s1	Add performance monitor	TRUE

Security

Secure address ranges

Multiple threadID support

ThreadID buffer depth

ThreadID mapping ranges

Enable master reorder buffer

Add performance monitor

Slave arbitration scheme

Handshake, FIFO, Auto

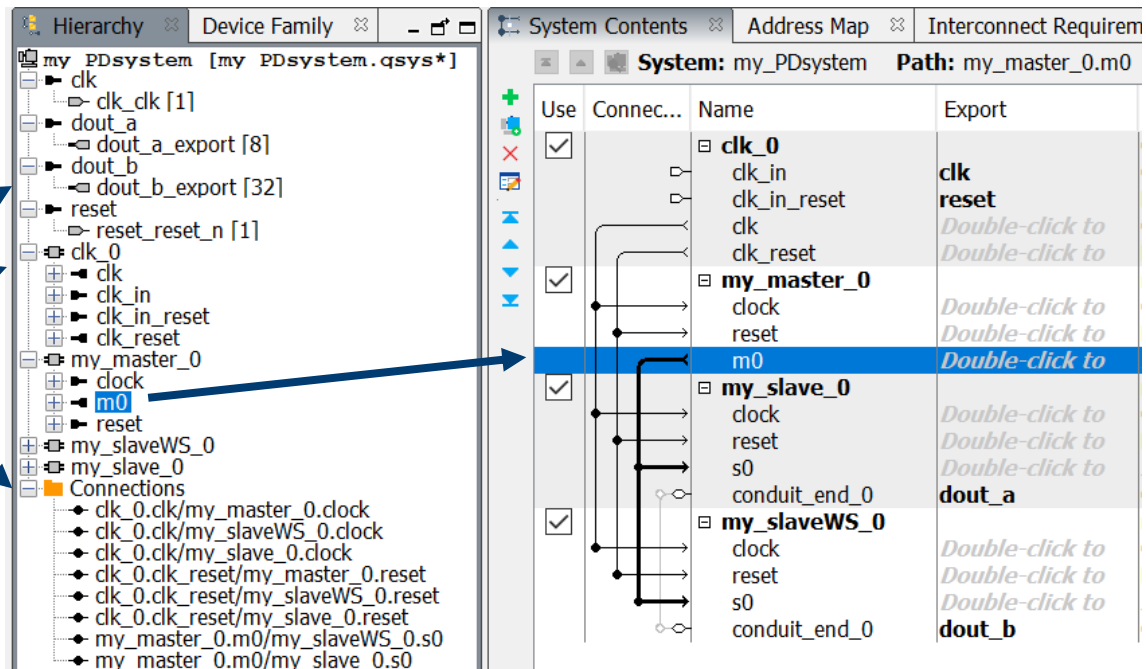
# Закладка Иерархия (Hierarchy Tab)

Средство анализа системы  
отображает:

- Иерархию системы
- Экспортированные интерфейсы
- Интерфейсы компонентов
- Связи между компонентами

Позволяет:

- Редактировать настройки (Edit)
- Отображать выбранные элементы в других закладках (Cross-highlight)



# Закладка Символ (Block Symbol)

Выбранный уровень  
иерархии системы

Позволяет отобразить все  
сигналы интерфейсов

The screenshot displays the 'Block Symbol' tab in a design tool. The left pane shows the hierarchy of the system 'my\_PDsystem'. The main pane shows the 'System: my\_PDsystem' with a table of components and their connections. The right pane shows the 'Block Symbol' view, which displays the system's external signals.

**System: my\_PDsystem**

Use	Connec...	Name	Description	Export	Clock
<input checked="" type="checkbox"/>		<b>clk_0</b>	Clock Source		
		clk_in	Clock Input	clk	
		clk_in_reset	Reset Input	reset	
		clk	Clock Output	Double-click to	clk_0
		clk_reset	Reset Output	Double-click to	
<input checked="" type="checkbox"/>		<b>my_master_0</b>	my_master		
		clock	Clock Input	Double-click to	clk_0
		reset	Reset Input	Double-click to	[clock]
		m0	Avalon Memory Mapped Master	Double-click to	[clock]
<input checked="" type="checkbox"/>		<b>my_slave_0</b>	my_slave		
		clock	Clock Input	Double-click to	clk_0
		reset	Reset Input	Double-click to	[clock]
		s0	Avalon Memory Mapped Slave	Double-click to	[clock]
		conduit_end_0	Conduit	Double-click to	[clock]
<input checked="" type="checkbox"/>		<b>my_slaveWS_0</b>	my_slaveWS		
		clock	Clock Input	Double-click to	clk_0
		reset	Reset Input	Double-click to	[clock]
		s0	Avalon Memory Mapped Slave	Double-click to	[clock]
		conduit_end_0	Conduit	Double-click to	[clock]

**Block Symbol**

☒ Show signals

The Block Symbol view shows the system's external signals:

- clk\_clk
- clk
- dout\_a\_export[7..0]
- dout\_b\_export[31..0]
- reset
- reset\_reset\_n

Символ отображает все внешние (экспортированные) интерфейсы системы

# Закладка Схема (Schematic View)

Отображение структуры системы и управление настройками

my\_PDsystem [my\_PDsystem.qsys\*]

clk

dout\_a

dout\_b

reset

clk\_0

my\_master\_0

clk

m0

reset

my\_slaveWS\_0

my\_slave\_0

clk

conduit\_end\_0

reset

s0

Connections

clk\_0.clk/my\_master\_0.clk

clk\_0.clk/my\_slaveWS\_0.clk

clk\_0.clk/my\_slave\_0.clk

clk\_0.clk\_reset/my\_master\_0.reset

clk\_0.clk\_reset/my\_slaveWS\_0.reset

clk\_0.clk\_reset/my\_slave\_0.reset

my\_master\_0.m0/my\_slaveWS\_0.s0

my\_master\_0.m0/my\_slave\_0.s0

System Contents

Address Map

Interconnect Requirements

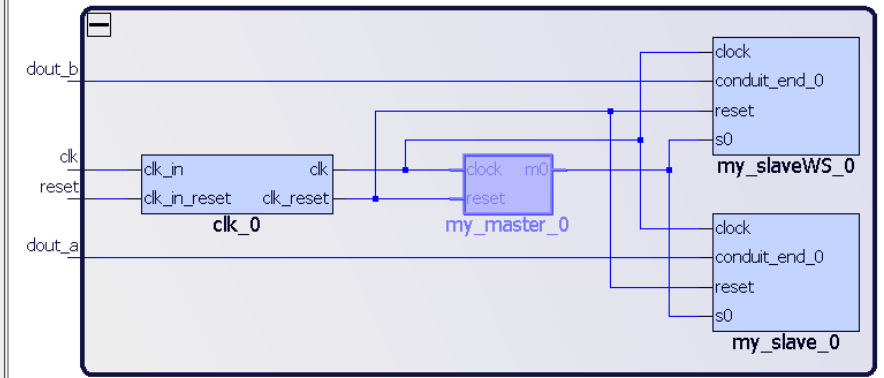
Schematic

Block Symbol

Parameters

System: my\_PDsystem

Path: my\_master\_0



my\_master\_0

clock

csi\_clk

reset

rsi\_reset

clk

address

write

writedata

waitrequest

avm\_m0\_address[31..0]

avm\_m0\_writedata[31..0]

avm\_m0\_waitrequest

my\_master

Сохранить .pdf

Zoom in/out

Exclude: rst clk st mm

Enter a comma-separated list of instances or types to exclude.

Reset filters

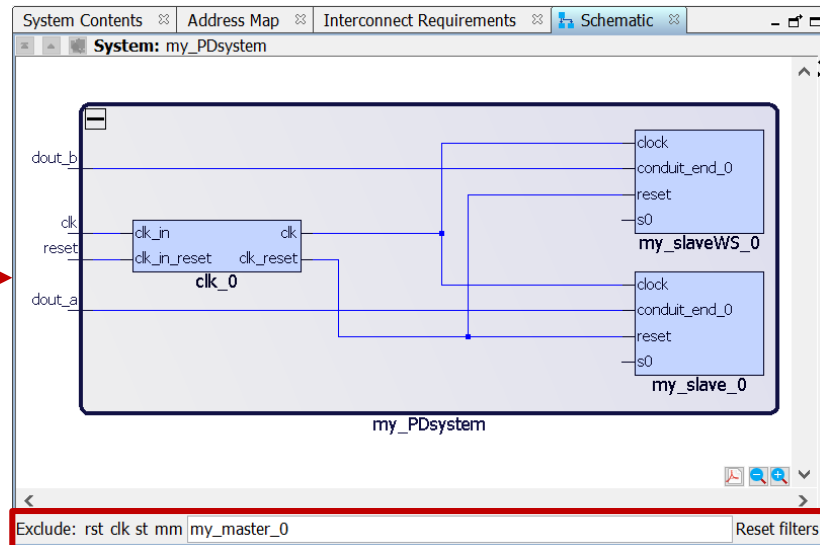
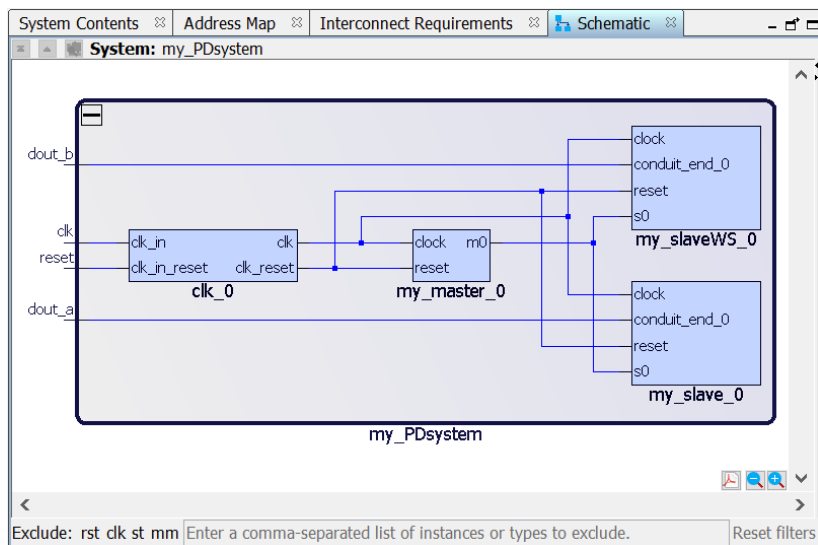
Отображение  
выбранных  
элементов в  
других закладках  
(Cross-highlight)

Фильтрация отображаемых соединений



# Закладка Схема (Schematic View)

## Фильтрация отображаемых элементов



Фильтрация отображаемых соединений  
и элементов

Сброс  
фильтра

# Закладка Сообщения (Messages Tab)

Закладка позволяет:

- Отображать сообщения (ошибки, предупреждения, информацию)
- Отображать источник ошибки/предупреждения - Double-click сообщение для выделения интерфейса/соединения или компонента
- Не должно быть ошибок для генерации системы
  - В процессе настройки ошибки могут (и будут) присутствовать

System Contents Address Map Interconnect Requirements Schematic

System: my\_PDSystem Path: my\_master\_0.m0

Use	Connec...	Name	Description	Export	Clock	Base	End	I...
<input checked="" type="checkbox"/>		clk_0	Clock Source	clk	exported			
<input checked="" type="checkbox"/>		clk_in	Clock Input	reset	[clk_in]			
<input checked="" type="checkbox"/>		clk_in_reset	Reset Input	Double-click to	clk_0			
<input checked="" type="checkbox"/>		clk	Clock Output	Double-click to	clk_0			
<input checked="" type="checkbox"/>		clk_reset	Reset Output	Double-click to	clk_0			
<input checked="" type="checkbox"/>		my_master_0	my_master	Double-click to	clk_0			
<input checked="" type="checkbox"/>		clock	Clock Input	Double-click to	clk_0			
<input checked="" type="checkbox"/>		reset	Reset Input	Double-click to	clk_0			
<input checked="" type="checkbox"/>		m0	Avalon Memory Mapped Master	Double-click to	clk_0			
<input checked="" type="checkbox"/>		my_slave_0	my_slave	Double-click to	clk_0			
<input checked="" type="checkbox"/>		clock	Clock Input	Double-click to	clk_0			
<input checked="" type="checkbox"/>		reset	Reset Input	Double-click to	clk_0			
<input checked="" type="checkbox"/>		s0	Avalon Memory Mapped Slave	Double-click to	clk_0			
<input checked="" type="checkbox"/>		conduit_end_0	Conduit	Double-click to	clk_0			
<input checked="" type="checkbox"/>		my_slaveWS_0	my_slaveWS	Double-click to	unconnected			
<input checked="" type="checkbox"/>		clock	Clock Input	Double-click to	clk_0			
<input checked="" type="checkbox"/>		reset	Reset Input	Double-click to	clk_0			
<input checked="" type="checkbox"/>		s0	Avalon Memory Mapped Slave	Double-click to	clk_0			
<input checked="" type="checkbox"/>		conduit_end_0	Conduit	Double-click to	clk_0			

Current filter:

Messages

Type	Path	Message
2 Errors		
	my_PDSystem.my_master_0.m0	my_slaveWS_0.s0 cannot be at 0x2 (0x0 or 0x4 are acceptable)
	my_PDSystem.my_slaveWS_0	my_slaveWS_0.clock must be connected to a clock output
1 Warning		
	my_PDSystem.my_slave_0	my_slave_0.conduit_end_0 must be exported, or connected to a matching conduit.

Сообщения  
(info, warning, error)

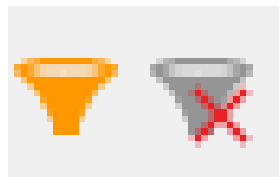
# Использование фильтра (System Contents Tab)

## Управление отображением системы в закладке System Contents

- Существуют predefined фильтры и можно создавать пользовательские фильтры

The screenshot shows the 'System Contents' window for a system named 'my\_PDsystem'. A green box highlights the 'Use' column, which contains a tree view of components. Another green box highlights the 'Filters' dialog box, which is open. The dialog shows the 'Filter' dropdown set to 'Clock and Reset Interfaces'. The 'Name' field is 'Clock and Reset Interfaces'. The 'match any' section contains two rules: 'Interface type is clock' and 'Interface type is reset'. The 'and match all' section is empty. The 'Current filter' at the bottom is 'Clock and Reset Interfaces'.

Use	Co...	Name	Description	Export	Clock
<input checked="" type="checkbox"/>	clk_0	clk_0	Clock Source	clk	exported
<input checked="" type="checkbox"/>	clk_in	clk_in	Clock Input	(clk_in)	[clk_in]
<input checked="" type="checkbox"/>	clk_in_reset	clk_in_reset	Reset Input	clk_0	clk_0
<input checked="" type="checkbox"/>	clk_reset	clk_reset	Reset Output	Double-click to Double-click to	Double-click to Double-click to
<input checked="" type="checkbox"/>	my_master_0	my_master_0	my_master	Double-click to Double-click to	clk_0 [clock]
<input checked="" type="checkbox"/>	clock	clock	Clock Input	Double-click to Double-click to	clk_0 [clock]
<input checked="" type="checkbox"/>	reset	reset	Reset Input	Double-click to Double-click to	clk_0 [clock]
<input checked="" type="checkbox"/>	my_slave_0	my_slave_0	my_slave	Double-click to Double-click to	clk_0 [clock]
<input checked="" type="checkbox"/>	clock	clock	Clock Input	Double-click to Double-click to	clk_0 [clock]
<input checked="" type="checkbox"/>	reset	reset	Reset Input	Double-click to Double-click to	clk_0 [clock]
<input checked="" type="checkbox"/>	my_slaveWS_0	my_slaveWS_0	my_slaveWS	Double-click to Double-click to	clk_0 [clock]
<input checked="" type="checkbox"/>	clock	clock	Clock Input	Double-click to Double-click to	clk_0 [clock]
<input checked="" type="checkbox"/>	reset	reset	Reset Input	Double-click to Double-click to	clk_0 [clock]



установить и сбросить  
фильтр

The screenshot shows the 'System Contents' window for a system named 'my\_PDsystem'. A green box highlights the 'Use' column, which contains a tree view of components. Another green box highlights the 'Filters' dialog box, which is open. The dialog shows the 'Filter' dropdown set to 'Avalon-MM Interfaces'. The 'Name' field is 'Avalon-MM Interfaces'. The 'match any' section contains one rule: 'Interface type is avalon'. The 'and match all' section is empty. The 'Current filter' at the bottom is 'Avalon-MM Interfaces'.

Use	...	Name	Description	Export	Clock
<input checked="" type="checkbox"/>	my_master_0	my_master_0	my_master	Double-click to Double-click to	[clock]
<input checked="" type="checkbox"/>	m0	m0	Avalon Memory Mapped Master	Double-click to Double-click to	clk_0 [clock]
<input checked="" type="checkbox"/>	my_slave_0	my_slave_0	my_slave	Double-click to Double-click to	clk_0 [clock]
<input checked="" type="checkbox"/>	s0	s0	Avalon Memory Mapped Slave	Double-click to Double-click to	clk_0 [clock]
<input checked="" type="checkbox"/>	my_slaveWS_0	my_slaveWS_0	my_slaveWS	Double-click to Double-click to	clk_0 [clock]
<input checked="" type="checkbox"/>	s0	s0	Avalon Memory Mapped Slave	Double-click to Double-click to	clk_0 [clock]

# Отображение тактовых доменов (Clock Domains)

Отображение компонентов и интерфейсов выбранного тактового домена

Меню View → Clock Domains - Beta

System: my\_PDSYSTEM Path: clk\_0.clk\_in

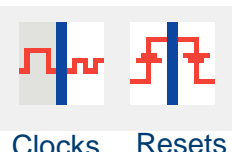
Use	Connec...	Name	Description	Export	Clock
<input checked="" type="checkbox"/>		clk_0	Clock Source		
<input checked="" type="checkbox"/>		clk_in	Clock Input	clk	exported
<input checked="" type="checkbox"/>		clk_in_reset	Reset Input	reset	[clk_in]
<input checked="" type="checkbox"/>		clk	Clock Output		clk_0
<input checked="" type="checkbox"/>		clk_reset	Reset Output		clk_0
<input checked="" type="checkbox"/>		my_master_0	my_master		
<input checked="" type="checkbox"/>		clock	Clock Input		clk_0
<input checked="" type="checkbox"/>		reset	Reset Input		[clock]
<input checked="" type="checkbox"/>		m0	Avalon Memory Mapped Master		[clock]
<input checked="" type="checkbox"/>		my_slave_0	my_slave		
<input checked="" type="checkbox"/>		clock	Clock Input		clk_0
<input checked="" type="checkbox"/>		reset	Reset Input		[clock]
<input checked="" type="checkbox"/>		s0	Avalon Memory Mapped Slave		[clock]
<input checked="" type="checkbox"/>		conduit_end_0	Conduit	dout_a	[clock]
<input checked="" type="checkbox"/>		my_slaveWS_0	my_slaveWS		
<input checked="" type="checkbox"/>		clock	Clock Input		clk_0
<input checked="" type="checkbox"/>		reset	Reset Input		[clock]
<input checked="" type="checkbox"/>		s0	Avalon Memory Mapped Slave		[clock]
<input checked="" type="checkbox"/>		conduit_end_0	Conduit	dout_b	[clock]

The current design has 1 clock domain  
Selected clock domains:  
• my\_PDSYSTEM.clk - 50.0 MHz  
Selected element:  
• clk\_0.clk\_in  
Export:  
clk

Current filter: my\_PDSYSTEM.clk

Система с одним тактовым доменом

Отображение  
интерфейсов  
выбранного  
домена



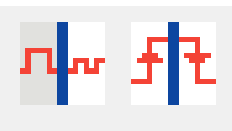
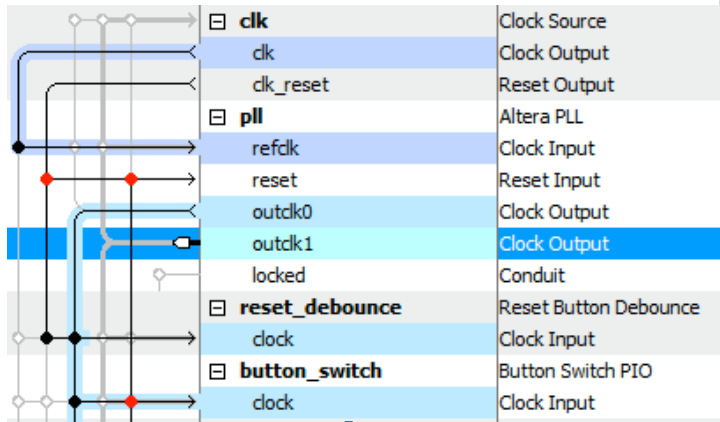
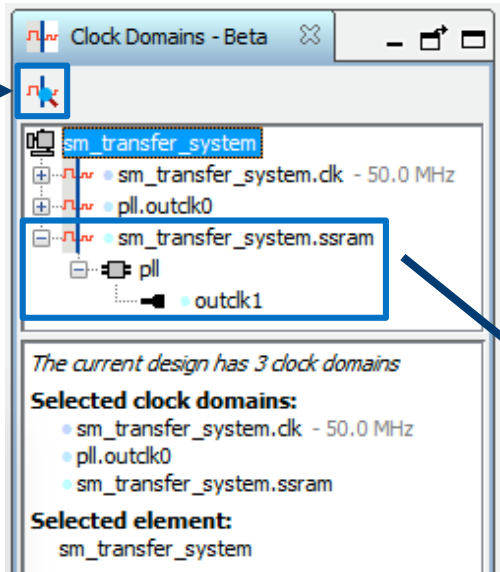
Отображение  
интерфейсов  
выбранного  
домена

# Отображение тактовых доменов (Clock Domains)

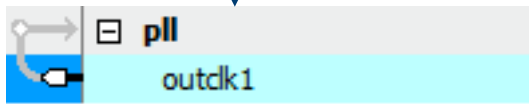
Система с несколькими тактовыми доменами

Меню View → Clock Domains - Beta

Отображение интерфейсов выбранного домена



Clocks Resets

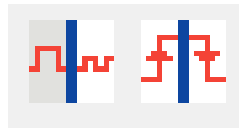


Отображение интерфейсов выбранного домена



Сброс фильтра

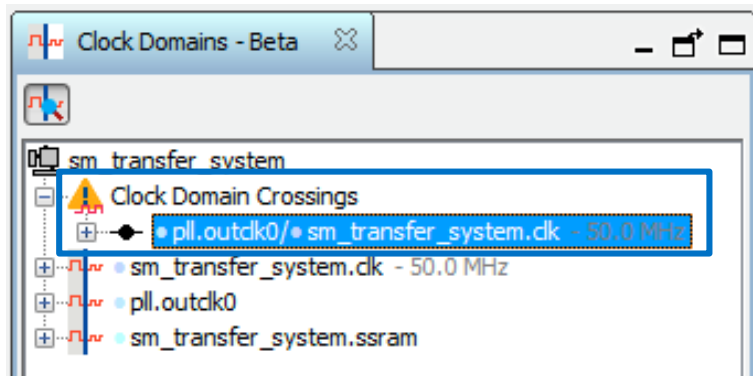
# Пресечение доменов (Clock Domain Crossings)



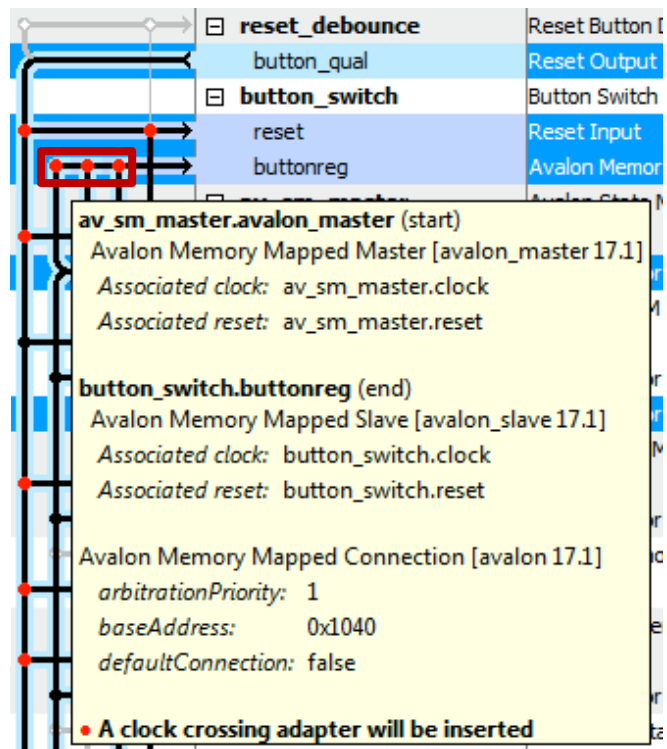
Clocks Resets

Отображаются подключения, в которых необходимо добавить адаптеры

- Места подключений отображены красными точками
- Подсказка (Tooltip) объясняет почему необходим адаптер
- Можно принять установку адаптера или изменить подключение



Ведомый (slave)  
**button\_switch**  
оказался в  
таковом домене  
ОТЛИЧНОМ ОТ  
такового домена  
своего ведущего  
(Master)

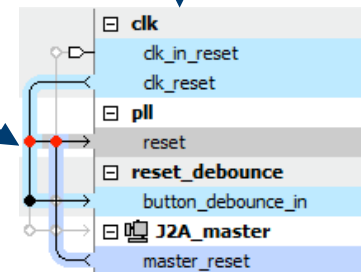
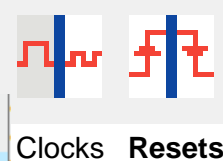
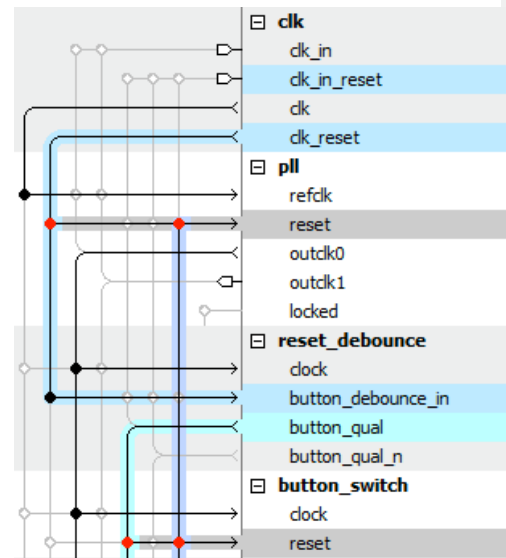
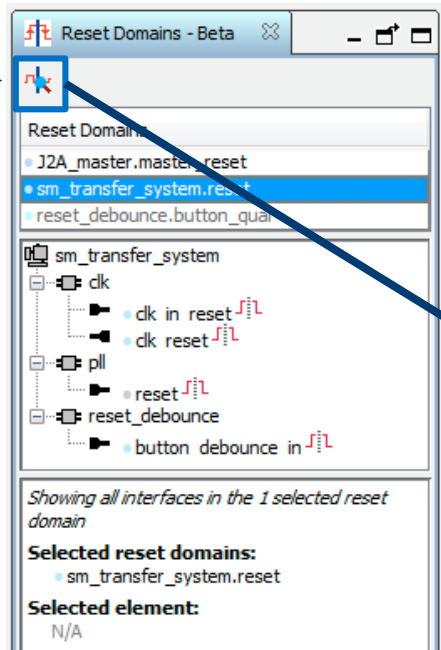


# Отображение доменов Reset

## Отображение компонентов и интерфейсов выбранного домена reset

Меню **View** → **Reset Domains - Beta**

Отображение  
интерфейсов  
выбранного  
домена



Сброс  
фильтра

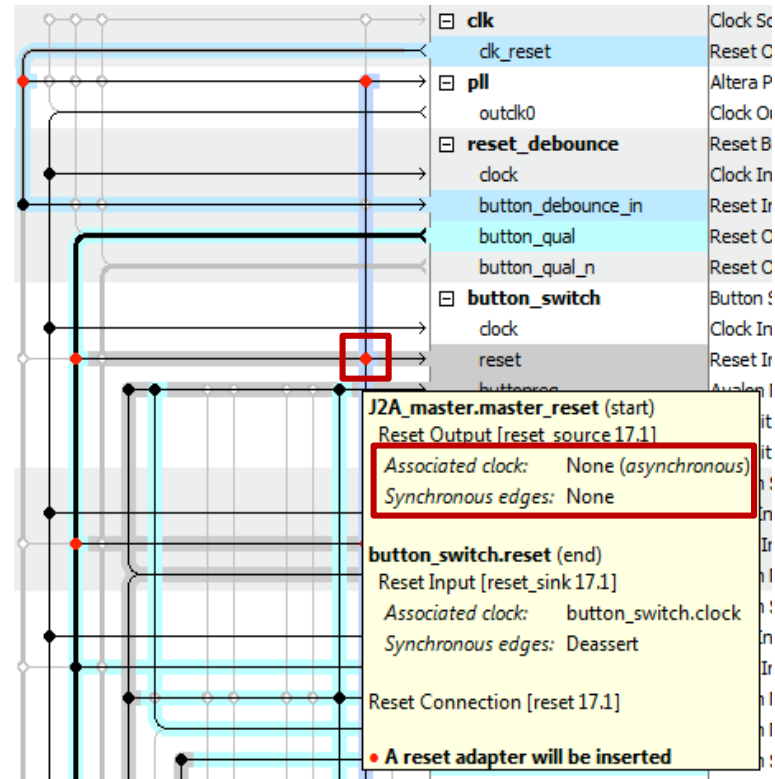
Система с несколькими тактовыми доменами

# Пресечение доменов (Reset Domain Crossings)

Отображаются подключения, в которых необходимо добавить адаптеры

- Места подключений отображены красными точками
- Подсказка (Tooltip) объясняет почему необходим адаптер
- Можно принять установку адаптера или изменить подключение

**master\_reset**  
ассинхронный





# Полезные команды

Команда: меню **System => Show System with Platform Designer Interconnect** отображает систему с модулями вставленными PD

The image shows a transition from a basic system view to a detailed Platform Designer Interconnect view. A blue arrow points from the left window to the right window.

**Left Window: System Contents**

Use	Connec...	Name	Description	Export	Clock
<input checked="" type="checkbox"/>		<b>clk_0</b>	Clock Source		
		clk_in	Clock Input	<b>clk</b>	<b>exported</b>
		clk_in_reset	Reset Input	[clk_in]	[clk_in]
		clk	Clock Output	clk_0	clk_0
		clk_reset	Reset Output		
<input checked="" type="checkbox"/>		<b>my_master_0</b>	my_master		
		clock	Clock Input	Double-click to [clock]	clk_0 [clock]
		reset	Reset Input	Double-click to [clock]	[clock]
		m0	Avalon Memory Mapped Master		
<input checked="" type="checkbox"/>		<b>my_slave_0</b>	my_slave		
		clock	Clock Input	Double-click to [clock]	clk_0 [clock]
		reset	Reset Input	Double-click to [clock]	[clock]
		s0	Avalon Memory Mapped Slave	Double-click to [clock]	[clock]
<input checked="" type="checkbox"/>		<b>my_slaveWS_0</b>	my_slaveWS		
		clock	Clock Input	Double-click to [clock]	clk_0 [clock]
		reset	Reset Input	Double-click to [clock]	[clock]
		s0	Avalon Memory Mapped Slave	Double-click to [clock]	[clock]
		conduit_end_0	Conduit	Double-click to [clock]	[clock]

**Right Window: System Contents**

Use	Connections	Name	Description	Export	Clock
<input checked="" type="checkbox"/>		<b>mm_interconnect_0</b>	MM Interconnect		
		clk_0_clk	Clock Input	Double-click to [clk_0_clk]	clk_0 [clk_0_clk]
		my_master_0_reset_reset_...	Reset Input	Double-click to [clk_0_clk]	[clk_0_clk]
		my_master_0_m0	Avalon Memory Mapped Slave	Double-click to [clk_0_clk]	[clk_0_clk]
		my_slave_0_s0	Avalon Memory Mapped Master	Double-click to [clk_0_clk]	[clk_0_clk]
		my_slaveWS_0_s0	Avalon Memory Mapped Master	Double-click to [clk_0_clk]	[clk_0_clk]
<input checked="" type="checkbox"/>		<b>rst_controller</b>	Merlin Reset Controller		
		reset_in0	Reset Input	Double-click to [clk]	clk_0 [clk]
		clk	Clock Input	Double-click to [clk]	clk_0 [clk]
		reset_out	Reset Output	Double-click to [clk]	[clk]
<input checked="" type="checkbox"/>		<b>clk_0</b>	Clock Source		
		clk_in	Clock Input	clk	exported
		clk_in_reset	Reset Input	reset	[clk_in]
		clk	Clock Output	clk_0	clk_0
		clk_reset	Reset Output	Double-click to [clk_0]	clk_0
<input checked="" type="checkbox"/>		<b>my_master_0</b>	my_master		
		clock	Clock Input	Double-click to [clock]	clk_0 [clock]
		reset	Reset Input	Double-click to [clock]	[clock]
		m0	Avalon Memory Mapped Master		
<input checked="" type="checkbox"/>		<b>my_slave_0</b>	my_slave		
		clock	Clock Input	Double-click to [clock]	clk_0 [clock]
		reset	Reset Input	Double-click to [clock]	[clock]
		s0	Avalon Memory Mapped Slave	Double-click to [clock]	[clock]
<input checked="" type="checkbox"/>		<b>my_slaveWS_0</b>	my_slaveWS		
		clock	Clock Input	Double-click to [clock]	clk_0 [clock]
		reset	Reset Input	Double-click to [clock]	[clock]
		s0	Avalon Memory Mapped Slave	Double-click to [clock]	[clock]
		conduit_end_0	Conduit	Double-click to [clock]	[clock]

# Полезные команды

- Меню **File**

- Обновить систему - Refresh System (F5)
  - Обновляет все IP и компоненты

- Меню **System**

- Обновить IP - Upgrade IP Cores
  - Если система создана в предыдущих версиях PD, то можно обновить устаревшие IP
- Назначить номера прерываний - Assign Interrupt Numbers
  - Автоматическое назначение номеров прерываний для пар interrupt sender/receiver
- Назначить код операции для пользовательской инструкции Assign Custom Instruction Opcodes
  - Используется с процессором Nios® II при создании пользовательских инструкций
- Удалить «висящие» выводы - Remove Dangling Connections
  - Удаляет неподключенные линии соединения (интерфейсы) в закладке System Contents tab

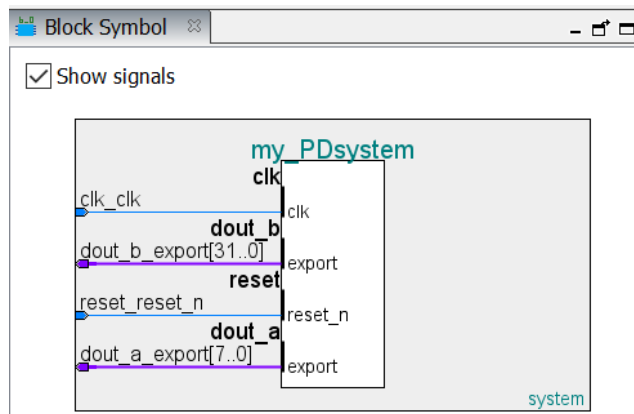
# Полезные команды

- Меню **System**

- Назначить базовые адреса - Auto-Assign Base Addresses
  - Автоматически назначает допустимые базовые адреса всем ведомым (slave)
- Подключить сигналы сброса – Create Global Resets Network
  - Автоматически подключает все сигналы Reset

# Заготовка HDL Instantiation Template

Заготовка для использования созданной системы как компонента в HDL описании верхнего уровня.



Меню **Generate** → **Show Instantiation Template**

## Instantiation Template

You can copy the example HDL below to declare an instance of **my\_PDsystem**.

HDL Language: Verilog ▾

### Example HDL

```
my_PDsystem u0 (  
    .clk_clk      (<connected-to-clk_clk>),      // clk.clk  
    .dout_b_export (<connected-to-dout_b_export>), // dout_b.export  
    .reset_reset_n (<connected-to-reset_reset_n>), // reset.reset_n  
    .dout_a_export (<connected-to-dout_a_export>) // dout_a.export  
);
```

HDL Language: VHDL ▾

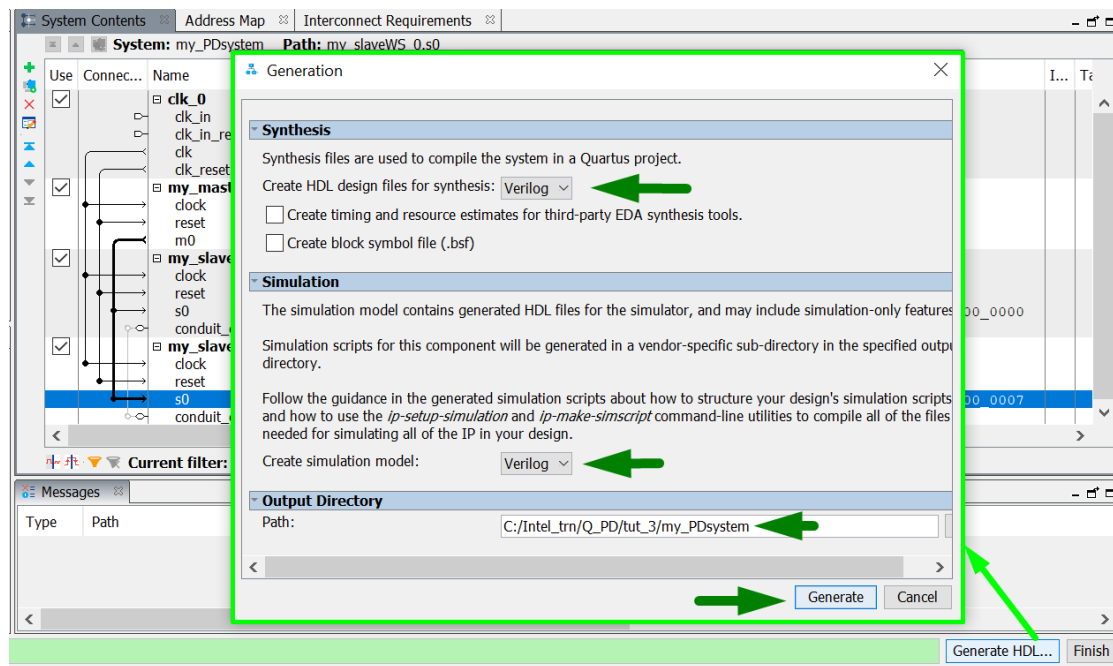
### Example HDL

```
component my_PDsystem is  
    port (  
        clk_clk      : in  std_logic      := 'X'; -- clk  
        dout_b_export : out std_logic_vector(31 downto 0); -- export  
        reset_reset_n : in  std_logic      := 'X'; -- reset_n  
        dout_a_export : out std_logic_vector(7 downto 0) -- export  
    );  
end component my_PDsystem;  
  
u0 : component my_PDsystem  
    port map (  
        clk_clk      => CONNECTED_TO_clk_clk,      -- clk.clk  
        dout_b_export => CONNECTED_TO_dout_b_export, -- dout_b.export  
        reset_reset_n => CONNECTED_TO_reset_reset_n, -- reset.reset_n  
        dout_a_export => CONNECTED_TO_dout_a_export -- dout_a.export  
    );
```

# Генерация HDL описания системы

Реализуется после создания системы

- Для запуска настройки процедуры используется кнопка **Generate HDL**
- Окно **Generation** позволяет:
  - Выбрать язык (Verilog, VHDL) для синтезируемого HDL описания
  - Выбрать язык для модели на HDL
- Задать папку для описаний системы (**Output Directory**)
- Для запуска процедуры – используется кнопка **Generate**



*Альтернативный способ запуска процедуры:*

**Меню *Generate* → *Generate HDL***

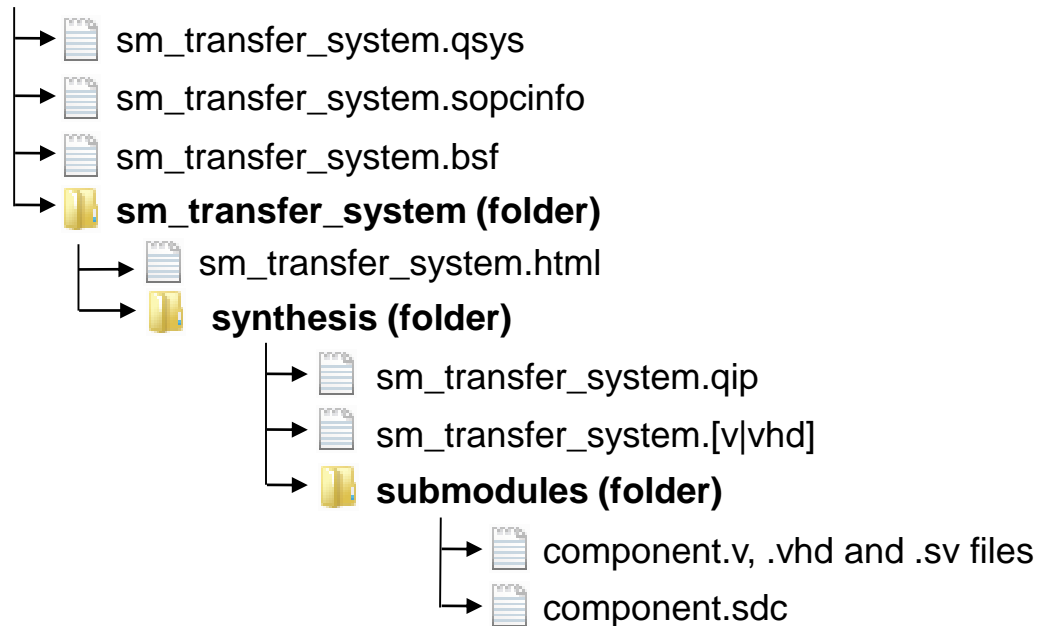
# План

- О приложении PD
- Пользовательский интерфейс приложения PD
- **Файлы, создаваемые PD, при генерации системы**
- Место PD в процедуре проектирования пакета QP
- Лабораторная 1

# Структура папок, создаваемых при генерации

Структура одинакова для всех микросхем, выпущенных ранее Intel® Arria® 10

 **project (folder)** *Система - sm\_transfer\_system*



# Файл .qsys

- Файл .qsys полностью описывает систему, созданную в Platform Designer (компоненты, подключения, параметры...)
- В рамках одного проекта Quartus (в одной рабочей папке проекта) может быть создано несколько систем.
- В папке где находится файл .qsys создается папка .qsys\_edit
  - Содержит настройки системы и раскладку PD



# Файлы создаваемые при генерации системы

- `<system_name>.sopcinfo`
  - XML файл, описывающий систему, созданную в Platform Designer system, используется при разработке ПО
- `<system_name>.bsf`
  - Файл с изображением символа системы. Используется при схемном вводе файла верхнего уровня в пакете Quartus (можно отключить создание)
- `<system_name>/<system_name>.html`
  - Отчет о результатах процедуры генерации HDL описания системы

# Файлы создаваемые при генерации системы

- Файлы для синтеза
  - Папка - <system\_name>/**synthesis**
    - <system\_name>.**qip**
      - Скрипт, содержащий ссылки на файлы, необходимые для синтеза системы
    - <system\_name>.[v|vhd]
      - Файл верхнего уровня в описании системы – связывает все компоненты
    - Папка <system\_name>/synthesis/**submodules** - содержит HDL описания модулей системы, сгенерированные PD на основе исходных кодов компонентов IP
      - Описания могут быть представлены на языках Verilog, SystemVerilog или VHDL
      - Папка может содержать файл(ы) .sdc – файлы с требованиями к временным параметрам
- Файлы для моделирования
  - Папка - <system\_name>/**simulation**

# План

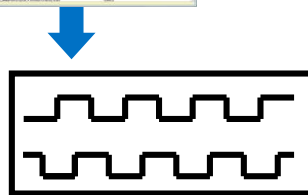
- О приложении PD
- Пользовательский интерфейс приложения PD
- Файлы, создаваемые PD, при генерации системы
- Место PD в процедуре проектирования пакета QP
- Лабораторная 1

# Процедура проектирования

Design specification



**Platform Designer**



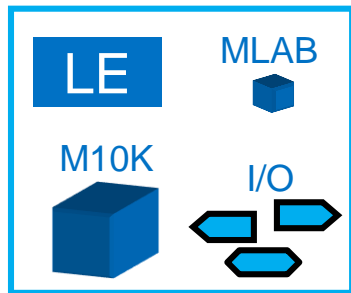
RTL simulation

- Functional simulation
- Verify logic model & data flow (no timing delays)



**Synthesis**

- Translate design into device specific primitives
- Optimization to meet required area & performance constraints
- Intel® Quartus® Prime software or other supported synthesis tools

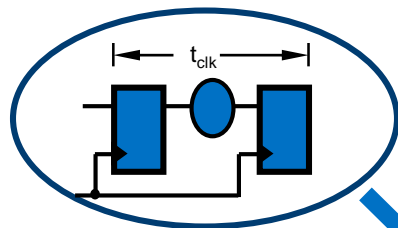


**Intel® Quartus® Prime**  
Design Software

**Place & Route**

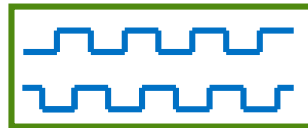
- Map primitives to specific locations inside
- Target technology with reference to area & performance constraints
- Specify routing resources to be used

# FPGA Hardware Design Flow (cont.)



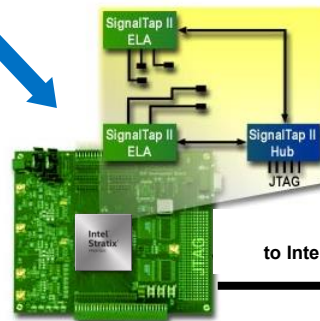
## Timing analysis

- Verify performance specifications were met
- Timing analyzer static timing analysis



## Gate-level simulation

- Functional timing simulation
- Verify design will work in target technology
- Get signal toggle info for power analysis
- *Combo of RTL simulation and timing analysis usually sufficient for verification*



## Test FPGA on PC Board

- Program & test device on board
- Use tools like Signal Tap logic analyzer & System Console for debugging

Download cable  
to Intel® Quartus® Prime software

# Процедура проектирования с использованием PD

1. Создать проект в пакете QP
2. Запустить PD и создать систему
3. В PD запустить процедуру генерации HDL описаний системы (и сопутствующих файлов)
  - Папка по умолчанию `<project folder>/<system name>`
4. Подключить к проекту файлы: **.qsys** или **.qip**
  - **qsys**: созданная система будет автоматически регенерироваться каждый раз, когда компилируется (синтезируется) проект в пакете Quartus.
  - **.qip**: к проекту добавляются сгенерированные HDL файлы с описанием системы, регенерация системы, если система была изменена, запускается в PD
5. В QP создать файл верхнего уровня проекта системы (это может быть файл-обертка, только переименовывающий выводы системы, созданной в PD)
6. В пакете ModelSim осуществить моделирование созданного файла верхнего уровня
  - *Потребуется создание теста*

# Процедура проектирования с использованием PD

6. Осуществить реализацию и отладку созданного файла верхнего уровня проекта системы на плате
  - Используется InSystemSource &Probe; SignalTapII
  - Потребуется создание файла с описанием системы для отладки
7. Задать требования к проекту системы:
  - к временным параметрам - timing constraints (**.sdc**),
  - подключению выводов микросхемы к сигналам модуля верхнего уровня (**.qsf**)
  - Многие IP имеют свой набор требований (их файл **.sdc** указан в **.qip** файле)
  - Необходимо создать файл **.sdc** для модуля верхнего уровня, как минимум для:  
тактовых входов (`create_clock`, `create_generated_clock`) и тактовых сигналов,  
порожденных в PLL (`derive_pll_clocks`)
8. Осуществить полную компиляцию проекта

# План

- О приложении PD
- Пользовательский интерфейс приложения PD
- Файлы, создаваемые PD, при генерации системы
- Место PD в процедуре проектирования пакета QP
- Лабораторная 1

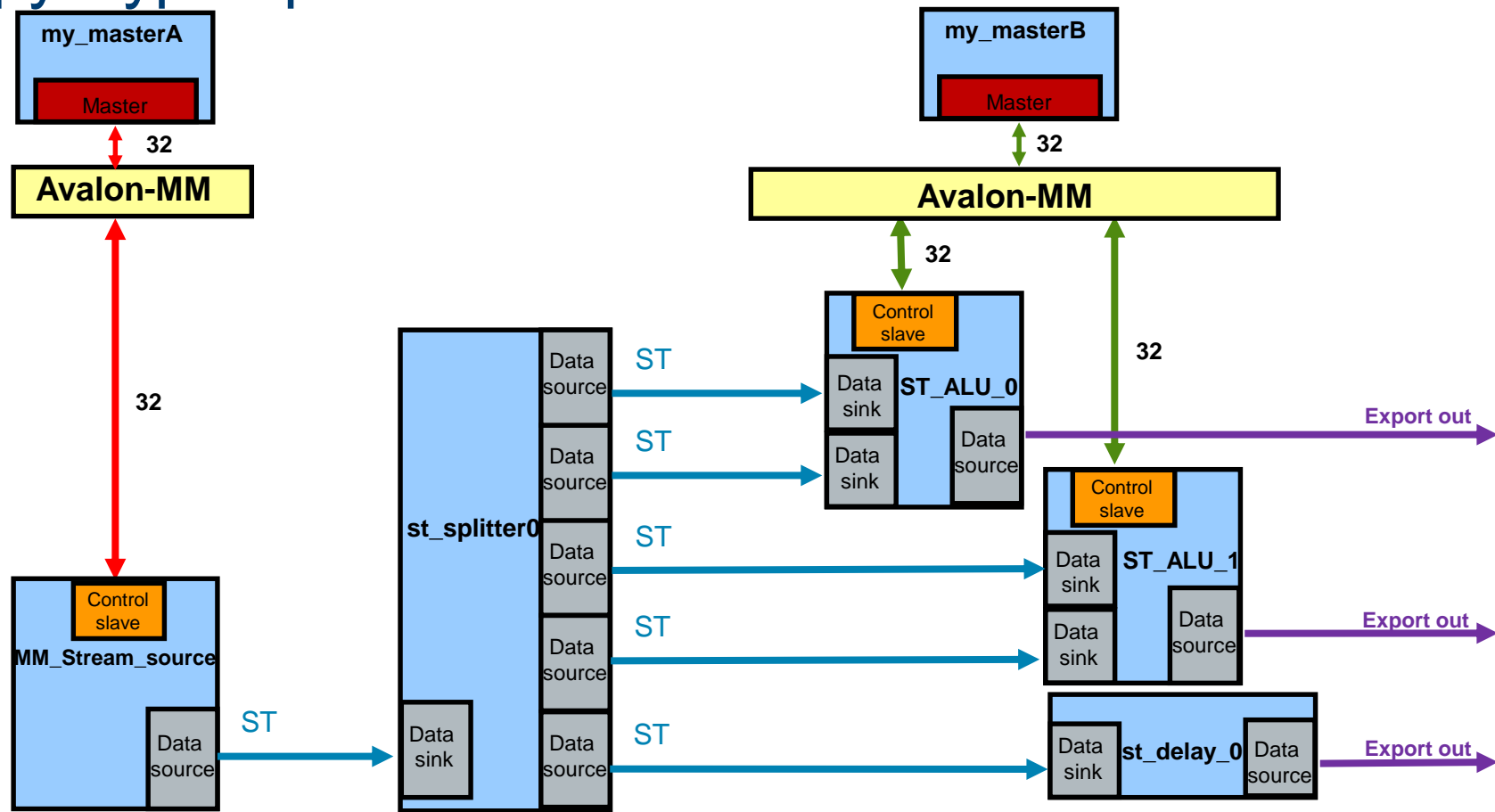




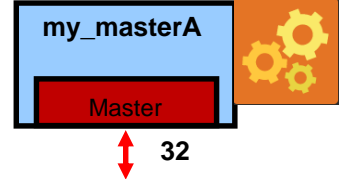
# *Лабораторная 1*



# Структура проекта



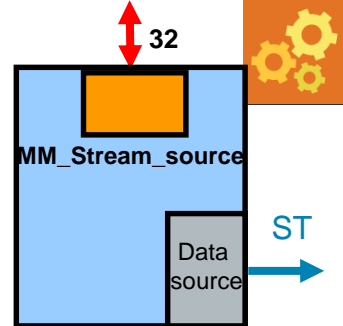
# my\_masterA



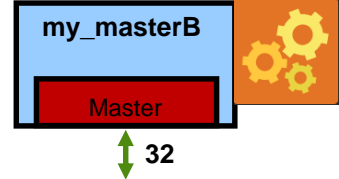
```
1  `timescale 1 ns / 1 ns
2  module my_masterA #(
3      parameter integer address = 0,
4      parameter integer data    = 100 //cnt_up = 100; cnt_down = 200
5  ) (
6      //MM master
7      input bit          csi_clk,          // MM master clk
8      input bit          rsi_reset,        // MM master reset
9      output bit [31:0]  avm_m0_address,   // MM master address
10     output bit          avm_m0_write,     // MM master write
11     output bit [31:0]  avm_m0_writedata,  // MM master writedata
12     input bit          avm_m0_waitrequest // MM master waitrequest
13 );
14 enum bit[1:0] {initSM, del1, wr1D, ended } fsm_MM;
15
16 always_ff @ (posedge csi_clk)
17 if (rsi_reset) fsm_MM <= initSM;
18 else
19     case (fsm_MM)
20         initSM : fsm_MM <= del1;
21         del1   : fsm_MM <= wr1D;
22         wr1D   : if (avm_m0_waitrequest) fsm_MM <= wr1D;
23                 else fsm_MM <= ended;
24         ended  : fsm_MM <= ended;
25     endcase
26
27 always_comb
28 begin
29     case (fsm_MM)
30         wr1D:
31             begin
32                 avm_m0_address = address;
33                 avm_m0_write   = 1'd1;
34                 avm_m0_writedata = data;
35             end
36         default
37             begin
38                 avm_m0_address = 32'd255;
39                 avm_m0_write   = 1'd0;
40                 avm_m0_writedata = 32'd255;
41             end
42     endcase
43 end
44 endmodule
```

# MM\_Stream\_source

```
1  `timescale 1 ps / 1 ps
2  module MM_Stream_source (
3      //clk and reset
4      input bit csi_clk,           // clock clk
5      input bit rsi_reset,        // reset reset
6      //stream source
7      output bit [31:0] aso_out0_data, // ST source data
8      input bit aso_out0_ready, // ST source ready
9      output bit aso_out0_valid, // ST source valid
10     //MM slave
11     input bit [31:0] avs_s0_writedata, // MM slave writedata
12     input bit avs_s0_write, // MM slave write
13     output bit avs_s0_waitrequest // MM slave waitrequest
14 );
15
16 always_ff @(posedge csi_clk)
17     if(rsi_reset) aso_out0_data <= 32'd255;
18     else if (avs_s0_write & aso_out0_ready)
19         aso_out0_data <= avs_s0_writedata;
20
21 always_ff @(posedge csi_clk)
22     if(rsi_reset) aso_out0_valid <= 1'b0;
23     else
24         aso_out0_valid <= avs_s0_write & aso_out0_ready;
25
26 assign avs_s0_waitrequest = 1'b0;
27 endmodule
```



# my\_masterB



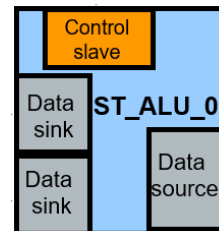
```
1  `timescale 1 ns / 1 ns
2  module my_masterB #(
3      parameter [31:0] address_1 = 0, parameter [31:0] data_1 = 111, //mult = 111  div  = 333
4      parameter [31:0] address_2 = 1, parameter [31:0] data_2 = 333  //add  = 222  sub = 444
5  ) (
6      input bit          csi_clk,          // MM master clk
7      input bit          rsi_reset,        // MM master reset
8      output bit [31:0]  avm_m0_address,   // MM master address
9      output bit         avm_m0_write,     // MM master write
10     output bit [31:0]  avm_m0_writedata,  // MM master writedata
11     input bit          avm_m0_waitrequest // MM master waitrequest
12 );
13 typedef enum bit[2:0] {initSM, del1, wr1D, del2, wr2D, ended } fsm_type;
14 fsm_type fsm_MM;
15
16 always_ff @ (posedge csi_clk)
17 if (rsi_reset) fsm_MM <= initSM;
18 else
19     case (fsm_MM)
20         initSM :          fsm_MM <= del1;
21         del1   :          fsm_MM <= wr1D;
22         wr1D   :          if (avm_m0_waitrequest) fsm_MM <= wr1D;
23                     else  fsm_MM <= del2;
24         del2   :          fsm_MM <= wr2D;
25         wr2D   :          if (avm_m0_waitrequest) fsm_MM <= wr2D;
26                     else  fsm_MM <= ended;
27         ended  :          fsm_MM <= ended;
28     endcase
29
30     always_comb
31     begin
32         case (fsm_MM)
33             wr1D:
34                 begin
35                     avm_m0_address      = address_1;
36                     avm_m0_write        = 1'd1;
37                     avm_m0_writedata    = data_1;
38                 end
39             wr2D:
40                 begin
41                     avm_m0_address      = address_2;
42                     avm_m0_write        = 1'd1;
43                     avm_m0_writedata    = data_2;
44                 end
45             default
46                 begin
47                     avm_m0_address      = 32'd255;
48                     avm_m0_write        = 1'd0;
49                     avm_m0_writedata    = 32'd255;
50                 end
51             endcase
52         end
53     endmodule
```

# ST\_ALU

```

1  `timescale 1 ns / 1 ns
2  module ST_ALU (
3      input bit      csi_clk,          // clock clk
4      input bit      rsi_reset,        // reset reset
5      // stream sink0
6      input bit [31:0] asi_in0_data,    // ST sink data
7      input bit      asi_in0_valid,    // ST sink valid
8      output bit      asi_in0_ready,    // ST sink ready
9      // stream sink1
10     input bit [31:0] asi_in1_data,    // ST sink data
11     input bit      asi_in1_valid,    // ST sink valid
12     output bit      asi_in1_ready,    // ST sink ready
13     //stream source
14     output bit [31:0] aso_out0_data,    // ST source data
15     input bit      aso_out0_ready,    // ST source ready
16     output bit      aso_out0_valid,    // ST source valid
17     //MM slave
18     input bit [31:0] avs_s0_writedata, // MM slave writedata
19     input bit      avs_s0_write,      // MM slave write
20     output bit      avs_s0_waitrequest // MM slave waitrequest
21 );
22 bit [31:0] ALU_type; //Type ALU mult = 111 add = 222 div = 333 sub = 444
23
24 always_ff @(posedge csi_clk)
25     if(rsi_reset) ALU_type <= '0;
26     else if (avs_s0_write) ALU_type <= avs_s0_writedata;
27
28 always_ff @(posedge csi_clk)
29     if(rsi_reset) begin
30         aso_out0_data <= '0;
31         aso_out0_valid <= '0;
32     end
33     else
34         if (asi_in0_valid & asi_in1_valid & aso_out0_ready)
35             case (ALU_type)
36                 'd111 : begin aso_out0_data <= asi_in0_data * asi_in1_data; aso_out0_valid <= 1'b1; end
37                 'd222 : begin aso_out0_data <= asi_in0_data + asi_in1_data; aso_out0_valid <= 1'b1; end
38                 'd333 : begin aso_out0_data <= asi_in0_data / asi_in1_data; aso_out0_valid <= 1'b1; end
39                 'd444 : begin aso_out0_data <= asi_in0_data - asi_in1_data; aso_out0_valid <= 1'b1; end
40                 default : begin aso_out0_data <= '0; aso_out0_valid <= 1'b0; end
41             endcase
42
43 assign asi_in0_ready = 1'b1;
44 assign asi_in1_ready = 1'b1;
45 assign avs_s0_waitrequest = 1'b0;
46
47 endmodule

```





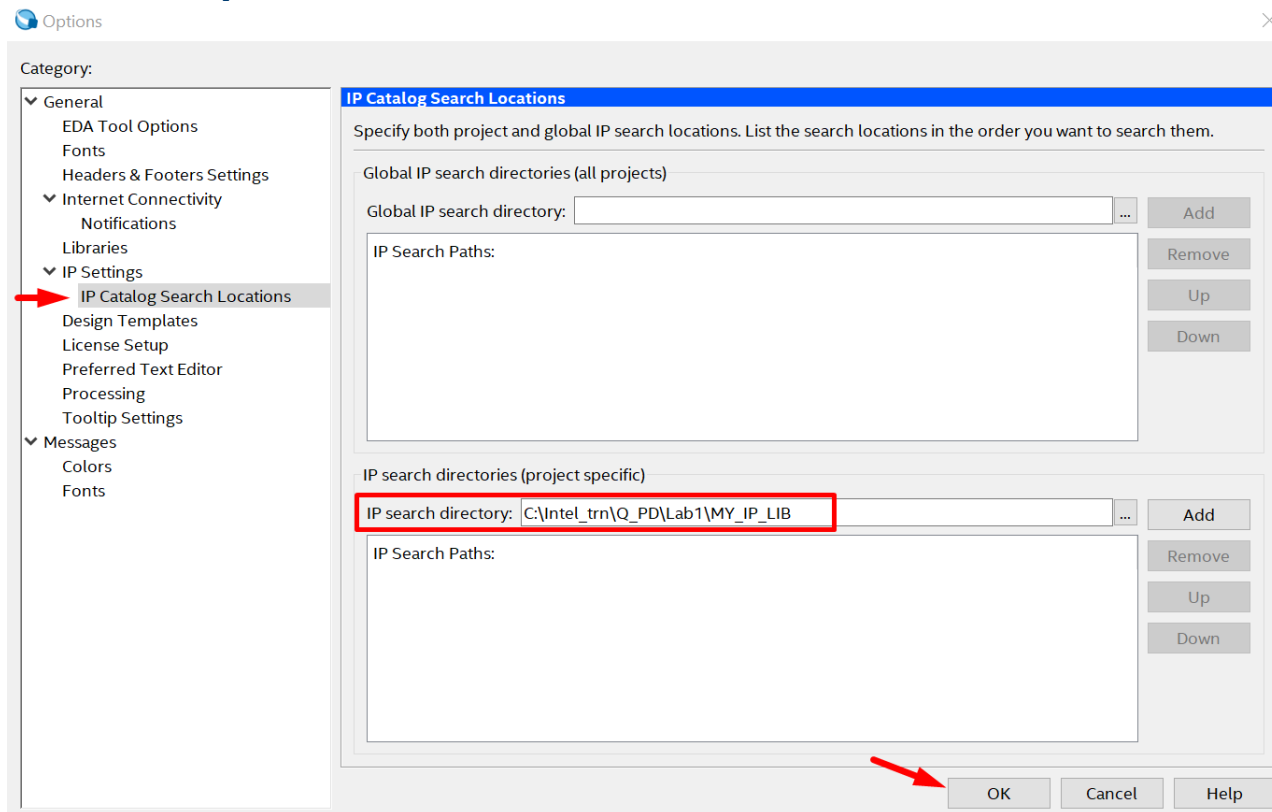
# В QP создайте проект

- **Рабочая папка:** C:\Intel\_trn\Q\_PD\Lab1
- **Имя проекта:** Lab1
- **Модуль верхнего уровня:** Lab1
- **Тип проекта:** Empty Project
- **Файлы не добавляются**
- **Микросхема:** может быть любой
  - Плата DE1-SOC                - 5CSEMA5F31C6N
  - Плата SoC Kit                - 5CSXFC6D6F31
  - Плата MAX10\_NEEK       - 10M50DAF484C6G
  - Плата miniDilabCIV (**выбирается по умолчанию**) - EP4CE6E22C8
- **EDA Tool Settings:** Simulation => ModelSim Altera Starter Edition



# В QP задайте путь к библиотеке IP


## ■ Команда: Tools=>Options

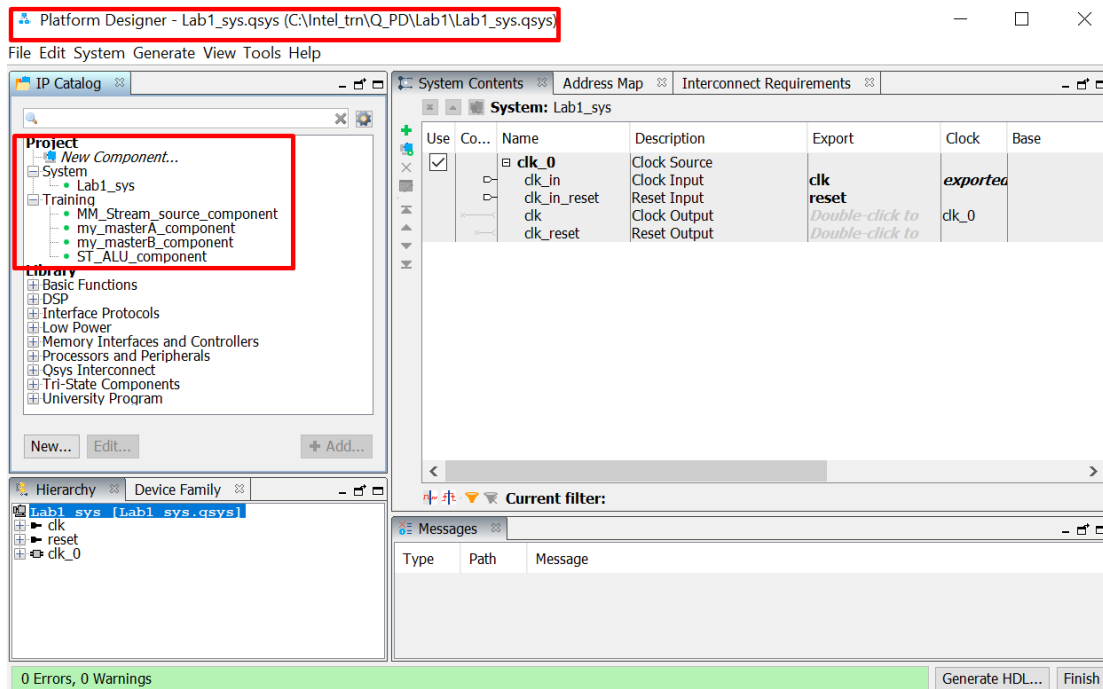






# В QP запустите приложение PD

- **Команда:** Tools => Platform Designer или иконка 
- В PD: сохраните систему под именем Lab1\_sys.qsys в рабочей папке проекта
- Убедитесь в том, что Ваша система выглядит так же, как показано на рисунке ниже





# Добавьте компоненты к системе

*В появляющемся окне настройки каждого компонента нажмите **Finish** не изменяя настройки компонента - настройка компонентов будет осуществлена после их добавления к системе*

- my\_masterA\_component
- MM\_stream\_source\_component
- my\_masterB\_component
- Avalon-ST Splitter (в строке поиска наберите ST)
- Avalon-ST Delay (в строке поиска наберите ST)
- ST\_ALU\_component (добавьте **два** компонента)

При добавлении компонентов на закладке Messages будут появляться сообщения об ошибках. На данном этапе на них можно не обращать внимание.



# Проверьте систему

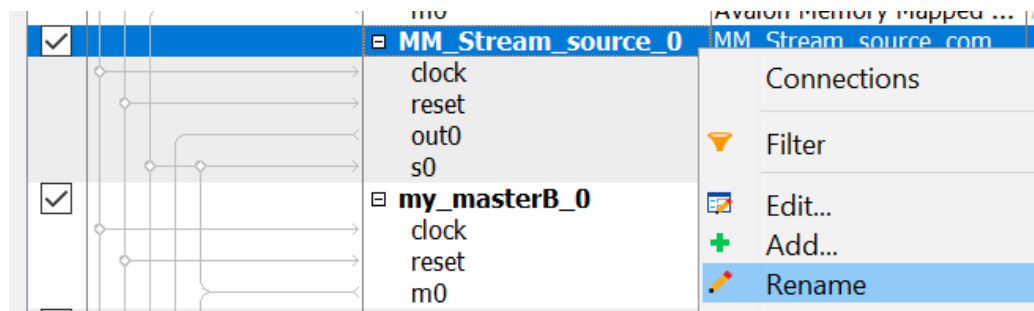
- Убедитесь в том, что Ваша система выглядит так же, как представленная на рисунке
- Сохраните файл.

System: Lab1_sys Path: clk_0						
Use	Connections	Name	Description	Export	Clock	Base
<input checked="" type="checkbox"/>		<b>clk_0</b>	Clock Source			
<input checked="" type="checkbox"/>		clk_in	Clock Input	clk	<i>exported</i>	
<input checked="" type="checkbox"/>		clk_in_reset	Reset Input	reset		
<input checked="" type="checkbox"/>		clk	Clock Output	<i>Double-click to</i>	clk_0	
<input checked="" type="checkbox"/>		clk_reset	Reset Output	<i>Double-click to</i>		
<input checked="" type="checkbox"/>		<b>my_masterA_0</b>	my_masterA_component			
<input checked="" type="checkbox"/>		clock	Clock Input	<i>Double-click to</i>	<i>unconnected</i>	
<input checked="" type="checkbox"/>		reset	Reset Input	<i>Double-click to</i>	[clock]	
<input checked="" type="checkbox"/>		m0	Avalon Memory Mapped ...	<i>Double-click to</i>	[clock]	
<input checked="" type="checkbox"/>		<b>MM_Stream_source_0</b>	MM_Stream_source_com...			
<input checked="" type="checkbox"/>		clock	Clock Input	<i>Double-click to</i>	<i>unconnected</i>	
<input checked="" type="checkbox"/>		reset	Reset Input	<i>Double-click to</i>	[clock]	
<input checked="" type="checkbox"/>		out0	Avalon Streaming Source	<i>Double-click to</i>	[clock]	
<input checked="" type="checkbox"/>		s0	Avalon Memory Mapped ...	<i>Double-click to</i>	[clock]	
<input checked="" type="checkbox"/>		<b>my_masterB_0</b>	my_masterB_component			
<input checked="" type="checkbox"/>		clock	Clock Input	<i>Double-click to</i>	<i>unconnected</i>	
<input checked="" type="checkbox"/>		reset	Reset Input	<i>Double-click to</i>	[clock]	
<input checked="" type="checkbox"/>		m0	Avalon Memory Mapped ...	<i>Double-click to</i>	[clock]	
<input checked="" type="checkbox"/>		<b>st_splitter_0</b>	Avalon-ST Splitter			
<input checked="" type="checkbox"/>		clk	Clock Input	<i>Double-click to</i>	<i>unconnected</i>	
<input checked="" type="checkbox"/>		reset	Reset Input	<i>Double-click to</i>	[clk]	
<input checked="" type="checkbox"/>		in	Avalon Streaming Sink	<i>Double-click to</i>	[clk]	
<input checked="" type="checkbox"/>		out0	Avalon Streaming Source	<i>Double-click to</i>	[clk]	
<input checked="" type="checkbox"/>		out1	Avalon Streaming Source	<i>Double-click to</i>	[clk]	
<input checked="" type="checkbox"/>		<b>st_delay_0</b>	Avalon-ST Delay			
<input checked="" type="checkbox"/>		in	Avalon Streaming Sink	<i>Double-click to</i>	[clk]	
<input checked="" type="checkbox"/>		out	Avalon Streaming Source	<i>Double-click to</i>	[clk]	
<input checked="" type="checkbox"/>		clk	Clock Input	<i>Double-click to</i>	<i>unconnected</i>	
<input checked="" type="checkbox"/>		clk_reset	Reset Input	<i>Double-click to</i>	[clk]	
<input checked="" type="checkbox"/>		<b>ST_ALU_0</b>	ST_ALU_component			
<input checked="" type="checkbox"/>		clock	Clock Input	<i>Double-click to</i>	<i>unconnected</i>	
<input checked="" type="checkbox"/>		reset	Reset Input	<i>Double-click to</i>	[clock]	
<input checked="" type="checkbox"/>		in0	Avalon Streaming Sink	<i>Double-click to</i>	[clock]	
<input checked="" type="checkbox"/>		in1	Avalon Streaming Sink	<i>Double-click to</i>	[clock]	
<input checked="" type="checkbox"/>		out0	Avalon Streaming Source	<i>Double-click to</i>	[clock]	
<input checked="" type="checkbox"/>		s0	Avalon Memory Mapped ...	<i>Double-click to</i>	[clock]	
<input checked="" type="checkbox"/>		<b>ST_ALU_1</b>	ST_ALU_component			
<input checked="" type="checkbox"/>		clock	Clock Input	<i>Double-click to</i>	<i>unconnected</i>	
<input checked="" type="checkbox"/>		reset	Reset Input	<i>Double-click to</i>	[clock]	
<input checked="" type="checkbox"/>		in0	Avalon Streaming Sink	<i>Double-click to</i>	[clock]	
<input checked="" type="checkbox"/>		in1	Avalon Streaming Sink	<i>Double-click to</i>	[clock]	
<input checked="" type="checkbox"/>		out0	Avalon Streaming Source	<i>Double-click to</i>	[clock]	
<input checked="" type="checkbox"/>		s0	Avalon Memory Mapped ...	<i>Double-click to</i>	[clock]	



# Переименуйте компоненты

- Щелчком выделите компонент **MM\_Stream\_source\_0**
- Нажмите правую клавишу мыши
- Выберите команду Rename
- Измените имя компонента на **MM\_Stream\_source**
- Повторите процедуру для компонентов:
  - my\_masterA\_0 => Новое имя: **my\_masterA**
  - my\_masterB\_0 => Новое имя: **my\_masterB**





# Проверьте систему

- Убедитесь в том, что Ваша система выглядит так же, как представленная на рисунке
- Сохраните файл.

Use	Connections	Name	Description	Export	Clock	Base
<input checked="" type="checkbox"/>		<div>clk_0</div> <div>clk_in</div> <div>clk_in_reset</div> <div>clk</div> <div>clk_reset</div>	Clock Source Clock Input Reset Input Clock Output Reset Output	clk reset <i>Double-click to</i> <i>Double-click to</i>	<i>exported</i>  clk_0	
<input checked="" type="checkbox"/>		<div>my_masterA</div> <div>clock</div> <div>reset</div> <div>m0</div>	my_masterA_component Clock Input Reset Input Avalon Memory Mapped ...	<i>Double-click to</i> <i>Double-click to</i> <i>Double-click to</i>	<i>unconnected</i> [clock] [clock]	
<input checked="" type="checkbox"/>		<div>MM_Stream_source</div> <div>clock</div> <div>reset</div> <div>out0</div> <div>s0</div>	MM_Stream_source_com... Clock Input Reset Input Avalon Streaming Source Avalon Memory Mapped ...	<i>Double-click to</i> <i>Double-click to</i> <i>Double-click to</i> <i>Double-click to</i>	<i>unconnected</i> [clock] [clock] [clock]	
<input checked="" type="checkbox"/>		<div>my_masterB</div> <div>clock</div> <div>reset</div> <div>m0</div>	my_masterB_component Clock Input Reset Input Avalon Memory Mapped ...	<i>Double-click to</i> <i>Double-click to</i> <i>Double-click to</i>	<i>unconnected</i> [clock] [clock]	
<input checked="" type="checkbox"/>		<div>st_splitter_0</div> <div>clk</div> <div>reset</div> <div>in</div> <div>out0</div> <div>out1</div>	Avalon-ST Splitter Clock Input Reset Input Avalon Streaming Sink Avalon Streaming Source Avalon Streaming Source	<i>Double-click to</i> <i>Double-click to</i> <i>Double-click to</i> <i>Double-click to</i> <i>Double-click to</i>	<i>unconnected</i> [clk] [clk] [clk] [clk]	



# Подключите тактовый сигнал

- На закладке System Contents щелчком выделите интерфейс **clk\_0.clk** (интерфейс clk компонента clk\_0)
- Выполните команду меню View=>Connections
- В появившемся окне закладки Connections выберите подключение ко всем тактовым входам
- Переключитесь на закладку System Contents
- Нажмите правую клавишу мыши
- Выберите команду Filter=>Clock and Reset Interfaces
- Убедитесь, что соединения выполнены -  
Ваша система выглядит так же, как представленная на рисунке

System Contents		Address Map	Interconnect Requirements	Connections
System: Lab1_sys		Path: clk_0.clk		
Connected to: clk_0.clk				
Connected	Connection	Clock Crossing		
<input checked="" type="checkbox"/>	clk_0.clk/MM_Stream_source.clock			
<input checked="" type="checkbox"/>	clk_0.clk/ST_ALU_0.clock			
<input checked="" type="checkbox"/>	clk_0.clk/ST_ALU_1.clock			
<input checked="" type="checkbox"/>	clk_0.clk/my_masterA.clock			
<input checked="" type="checkbox"/>	clk_0.clk/my_masterB.clock			
<input checked="" type="checkbox"/>	clk_0.clk/st_delay_0.clk			
<input checked="" type="checkbox"/>	clk_0.clk/st_splitter_0.clk			



Use	Co...	Name	Description	Export	Clock
<input checked="" type="checkbox"/>		clk_0	Clock Source	clk	<i>exported</i>
		clk_in	Clock Input	reset	
		clk_in_reset	Reset Input	reset	
		clk	Clock Output	Double-click to	clk_0
		clk_reset	Reset Output	Double-click to	
<input checked="" type="checkbox"/>		my_masterA	my_masterA_component	Double-click to	clk_0
		clock	Clock Input	Double-click to	[clock]
		reset	Reset Input	Double-click to	
<input checked="" type="checkbox"/>		MM_Stream_source	MM_Stream_source_com...	Double-click to	clk_0
		clock	Clock Input	Double-click to	[clock]
		reset	Reset Input	Double-click to	
<input checked="" type="checkbox"/>		my_masterB	my_masterB_component	Double-click to	clk_0
		clock	Clock Input	Double-click to	[clock]
		reset	Reset Input	Double-click to	
<input checked="" type="checkbox"/>		st_splitter_0	Avalon-ST Splitter	Double-click to	clk_0
		clk	Clock Input	Double-click to	[clk]
		reset	Reset Input	Double-click to	
<input checked="" type="checkbox"/>		st_delay_0	Avalon-ST Delay	Double-click to	clk_0
		clk	Clock Input	Double-click to	[clk]
		clk_reset	Reset Input	Double-click to	
<input checked="" type="checkbox"/>		ST_ALU_0	ST_ALU_component	Double-click to	clk_0
		clock	Clock Input	Double-click to	[clock]
		reset	Reset Input	Double-click to	
<input checked="" type="checkbox"/>		ST_ALU_1	ST_ALU_component	Double-click to	clk_0
		clock	Clock Input	Double-click to	[clock]
		reset	Reset Input	Double-click to	



# Подключите сигнал Reset

- На закладке System Contents выполните команду меню System=>Create Global Reset Network
- Убедитесь, что соединения выполнены - Ваша система выглядит так же, как представленная на рисунке
- Сбросьте фильтрацию – нажмите на иконку  в нижней части окна System Contents
- Сохраните файл

Use	Co...	Name	Description	Export	Clock	Base
<input checked="" type="checkbox"/>		<div>clk_0<ul style="list-style-type: none"><li>clk_in</li><li>clk_in_reset</li><li>clk</li><li>clk_reset</li></ul></div>	Clock Source Clock Input Reset Input Clock Output Reset Output	<b>clk</b> <b>reset</b> <i>Double-click to</i> <i>Double-click to</i>	<i>exported</i>  clk_0	
<input checked="" type="checkbox"/>		<div>my_masterA<ul style="list-style-type: none"><li>clock</li><li>reset</li></ul></div>	my_masterA_component Clock Input Reset Input	<i>Double-click to</i> <i>Double-click to</i>	<b>clk_0</b> [clock]	
<input checked="" type="checkbox"/>		<div>MM_Stream_source<ul style="list-style-type: none"><li>clock</li><li>reset</li></ul></div>	MM_Stream_source_com... Clock Input Reset Input	<i>Double-click to</i> <i>Double-click to</i>	<b>clk_0</b> [clock]	ui
<input checked="" type="checkbox"/>		<div>my_masterB<ul style="list-style-type: none"><li>clock</li><li>reset</li></ul></div>	my_masterB_component Clock Input Reset Input	<i>Double-click to</i> <i>Double-click to</i>	<b>clk_0</b> [clock]	
<input checked="" type="checkbox"/>		<div>st_splitter_0<ul style="list-style-type: none"><li>clk</li><li>reset</li></ul></div>	Avalon-ST Splitter Clock Input Reset Input	<i>Double-click to</i> <i>Double-click to</i>	<b>clk_0</b> [clk]	
<input checked="" type="checkbox"/>		<div>st_delay_0<ul style="list-style-type: none"><li>clk</li><li>clk_reset</li></ul></div>	Avalon-ST Delay Clock Input Reset Input	<i>Double-click to</i> <i>Double-click to</i>	<b>clk_0</b> [clk]	
<input checked="" type="checkbox"/>		<div>ST_ALU_0<ul style="list-style-type: none"><li>clock</li><li>reset</li></ul></div>	ST_ALU_component Clock Input Reset Input	<i>Double-click to</i> <i>Double-click to</i>	<b>clk_0</b> [clock]	ui
<input checked="" type="checkbox"/>		<div>ST_ALU_1<ul style="list-style-type: none"><li>clock</li><li>reset</li></ul></div>	ST_ALU_component Clock Input Reset Input	<i>Double-click to</i> <i>Double-click to</i>	<b>clk_0</b> [clock]	ui

 **Current filter:** Clock and Reset Interfaces



# Подключите Avalon-MM интерфейсы

- На закладке System Contents щелчком выделите интерфейс **my\_masterA.m0**
- Нажмите правую клавишу мыши
- Выберите команду **Filter=> Avalon-MM Interfaces**
- В столбце Connections выполните подключения так, как показано на рисунке
- Сохраните файл

System: Lab1\_sys Path: my\_masterA.m0

Use	Co...	Name	Description	Export	Clock	Base	End
<input checked="" type="checkbox"/>		my_masterA	my_masterA_component		[dock]		
		m0	Avalon Memory Mapped ...	Double-click to	clk_0		
<input checked="" type="checkbox"/>		MM_Stream_source	MM_Stream_source_com...		[dock]		
		s0	Avalon Memory Mapped ...	Double-click to	clk_0	0x0000_0000	0x0000_0003
<input checked="" type="checkbox"/>		my_masterB	my_masterB_component		[dock]		
		m0	Avalon Memory Mapped ...	Double-click to	clk_0		
<input checked="" type="checkbox"/>		ST_ALU_0	ST_ALU_component		[dock]		
		s0	Avalon Memory Mapped ...	Double-click to	clk_0	0x0000_0000	0x0000_0003
<input checked="" type="checkbox"/>		ST_ALU_1	ST_ALU_component		[dock]		
		s0	Avalon Memory Mapped ...	Double-click to	clk_0	0x0000_0000	0x0000_0003

Current filter: Avalon-MM Interfaces

Messages

Type	Path	Message
1 Error		
	Lab1_sys.my_masterB.m0	ST_ALU_1.s0 (0x0..0x3) overlaps ST_ALU_0.s0 (0x0..0x3)






# Назначьте базовые адреса ведомым Avalon-MM

- Выберите команду меню **System=> Assign Base Addresses**
- Убедитесь, что адреса назначены - Ваша система выглядит так же, как представленная на рисунке

Use	Co...	Name	Description	Export	Clock	Base	End
<input checked="" type="checkbox"/>		my_masterA	my_masterA_component		[clock]		
		m0	Avalon Memory Mapped ...	Double-click to	clk_0		
<input checked="" type="checkbox"/>		MM_Stream_source	MM_Stream_source_com...		[clock]		
		s0	Avalon Memory Mapped ...	Double-click to	clk_0	0x0000_0000	0x0000_0003
<input checked="" type="checkbox"/>		my_masterB	my_masterB_component		[clock]		
		m0	Avalon Memory Mapped ...	Double-click to	clk_0		
<input checked="" type="checkbox"/>		ST_ALU_0	ST_ALU_component		[clock]		
		s0	Avalon Memory Mapped ...	Double-click to	clk_0	0x0000_0004	0x0000_0007
<input checked="" type="checkbox"/>		ST_ALU_1	ST_ALU_component		[clock]		
		s0	Avalon Memory Mapped ...	Double-click to	clk_0	0x0000_0000	0x0000_0003

- Если адреса назначены иначе – отредактируйте их
  - Дважды щелкните в поле Base адреса и введите правильный адрес (как на картинке)
- Зафиксируйте адреса – нажмите на символ  у **каждого** адреса
- Сохраните файл



# Закладка Address Map

- Выберите команду меню **View=> Address Map**

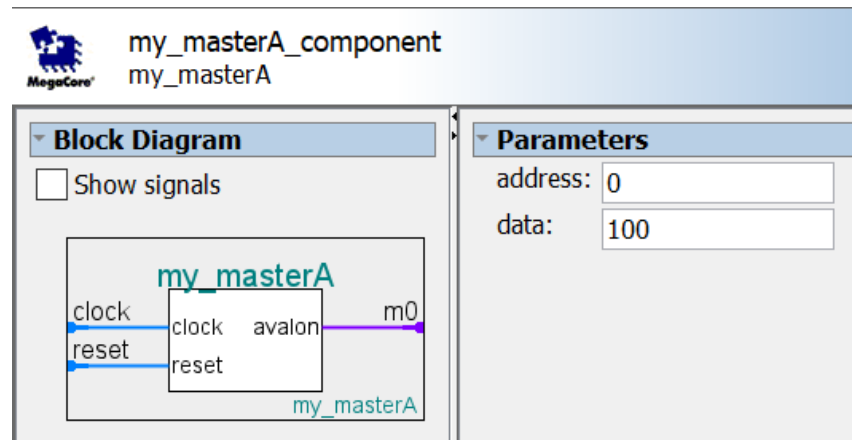
System Contents	Address Map	Interconnect Requirements	Connections
System: Lab1_sys Path: MM_Stream_source.s0			
	my_masterA.m0	my_masterB.m0	
MM_Stream_source.s0	0x0000 0000 - 0x0000 0003		
ST_ALU_0.s0		0x0000 0004 - 0x0000 0007	
ST_ALU_1.s0		0x0000 0000 - 0x0000 0003	

- В окне закладки изображены Ведущие (Master) шины Avalon\_MM – столбцы; и Ведомые (Slave) шины Avalon\_MM – строки
- Убедитесь, что у Вас окно выглядит так же, как представлено на рисунке
- Запомните базовые адреса (они нужны для настройки Ведущих):
  - MM\_Stream\_source = 0
  - ST\_ALU\_0 = 4
  - ST\_ALU\_1 = 0
- Переключитесь на закладку System Contents



# Настройка компонента my\_masterA

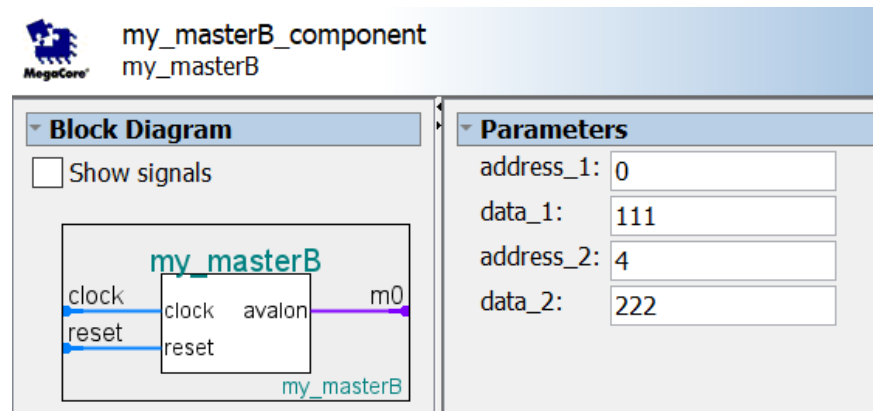
- Щелчком выберите **my\_masterA**
- Нажмите правую клавишу мыши и выберите команду Edit
- В появившемся окне настройки параметров задайте
  - Адрес, по которому Ведущий **my\_masterA** будет записывать данные
    - В нашей системе – это адрес **0** ( это базовый адрес Ведомого MM\_Stream\_source)
  - Записываемые данные: **100** – счет на сложение; 200 – счет на вычитание
  - Нажмите кнопку **Finish**
- Сохраните файл






# Настройка компонента my\_masterB

- Щелчком выберите **my\_masterB**
- Нажмите правую клавишу мыши и выберите команду Edit
- В появившемся окне задайте
  - Адреса, по которому Ведущий **my\_masterB** будет записывать данные
    - 0** (это базовый адрес Ведомого ST\_ALU\_1)
    - 4** (это базовый адрес Ведомого ST\_ALU\_0)
  - Записываемые данные:
    - 111** – сложение (тип операции ST\_ALU\_1)
    - 222** – умножение (тип операции ST\_ALU\_0) ;
    - Нажмите кнопку **Finish**
- Сохраните файл

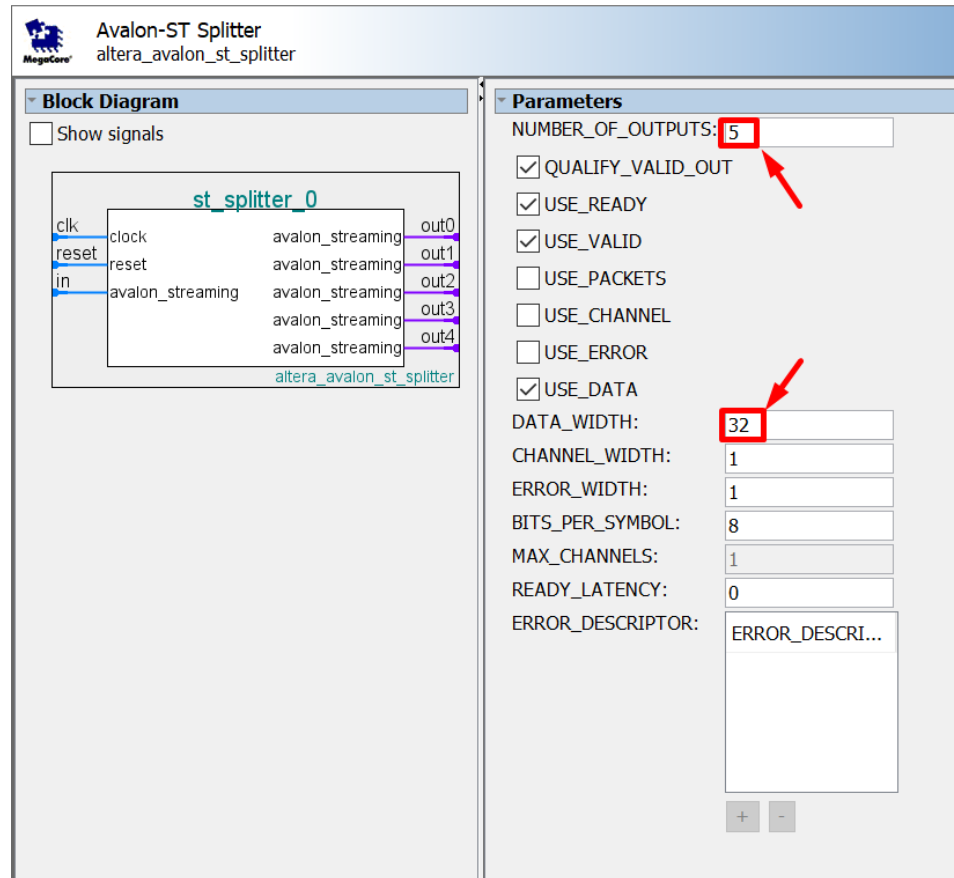




# Настройка компонента `st_splitter`

- Сбросьте фильтрацию – нажмите на иконку  в нижней части окна System Contents
- На закладке System Contents щелчком выделите компонент `st_splitter_0`
- Нажмите правую клавишу мыши
- Выберите команду **Edit**
- Установите
  - `NUMBER_OF_OUTPUTS` = 5
  - `DATA_WIDTH` = 32
- Нажмите кнопку Finish
- Сохраните файл

Avalon-ST Splitter - `st_splitter_0`



**Block Diagram**

☐ Show signals

`st_splitter_0`

clk: clock, reset, in (avalon\_streaming)

avalon\_streaming: out0, out1, out2, out3, out4

altera\_avalon\_st\_splitter

**Parameters**

NUMBER\_OF\_OUTPUTS: 5

☒ QUALIFY\_VALID\_OUT

☒ USE\_READY

☒ USE\_VALID

☐ USE\_PACKETS

☐ USE\_CHANNEL

☐ USE\_ERROR

☒ USE\_DATA

DATA\_WIDTH: 32

CHANNEL\_WIDTH: 1

ERROR\_WIDTH: 1

BITS\_PER\_SYMBOL: 8

MAX\_CHANNELS: 1

READY\_LATENCY: 0

ERROR\_DESCRIPTOR: ERROR\_DESCRI...



# Настройка компонента `st_delay`

- На закладке System Contents щелчком выделите компонент `st_delay_0`
- Нажмите правую клавишу мыши
- Выберите команду **Edit**
- Установите
  - `DATA_WIDTH = 32`
- Нажмите кнопку Finish
- Сохраните файл

**Block Diagram**

☒ Show signals

`st_delay_0`

Inputs: `in0_valid`, `in0_data[31..0]`, `clk`, `clk_reset`, `reset_n`

Outputs: `out0_valid`, `out0_data[31..0]`

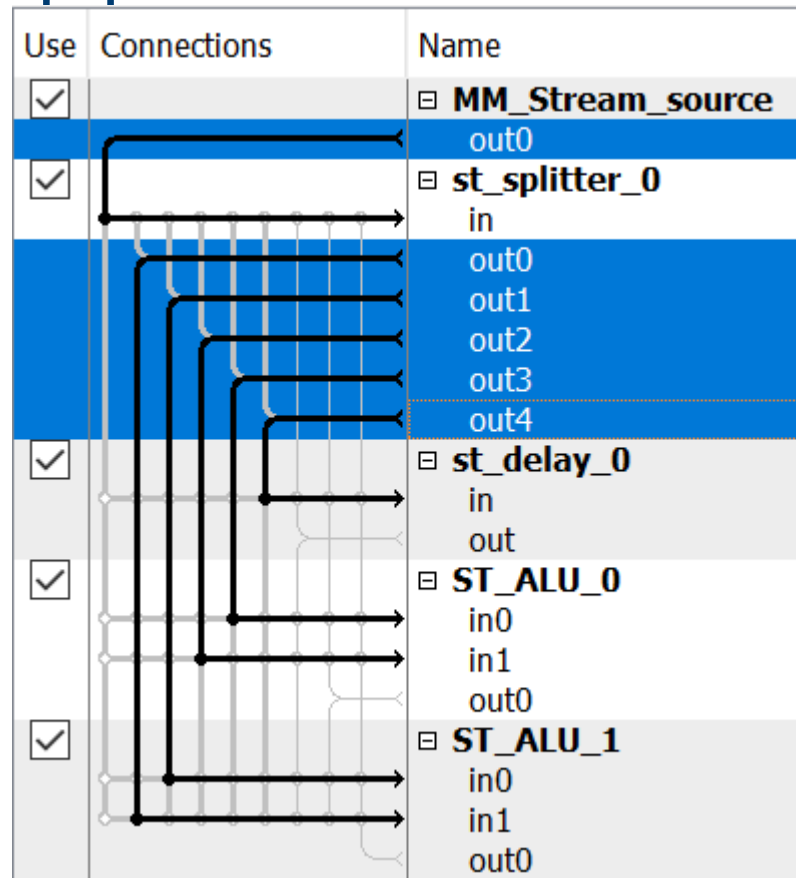
**Parameters**

NUMBER_OF_DELAY_CLOCKS:	1
<b>DATA_WIDTH:</b>	<b>32</b>
BITS_PER_SYMBOL:	8
<input type="checkbox"/> USE_PACKETS	
<input type="checkbox"/> USE_CHANNEL	
CHANNEL_WIDTH:	1
MAX_CHANNELS:	1
<input type="checkbox"/> USE_ERROR	
ERROR_WIDTH:	1



# Подключите Avalon-ST интерфейсы

- На закладке System Contents щелчком выделите интерфейс **MM\_Stream\_source.out0**
- Нажмите правую клавишу мыши
- Выберите команду **Filter=> Avalon-ST Interfaces**
- В столбце Connections выполните подключения так, как показано на рисунке
- Сохраните файл





# Экспортируйте выводы

- На закладке System Contents щелчком выделите интерфейс **st\_delay\_0.out0**
- Дважды щелкните в поле Export и задайте имя delay\_out
- Щелчком выделите интерфейс **ST\_ALU\_0.out0**
- Дважды щелкните в поле Export и задайте имя alu0\_out
- Щелчком выделите интерфейс **ST\_ALU\_1.out0**
- Дважды щелкните в поле Export и задайте имя alu1\_out
- Сбросьте фильтрацию – нажмите на иконку  в нижней части окна System Contents
- Сохраните файл

st_delay_0	Avalon-ST Delay	
in	Avalon Streaming Sink	Double-click to
out	Avalon Streaming Source	delay_out

ST_ALU_0	ST_ALU_component	
in0	Avalon Streaming Sink	Double-click to
in1	Avalon Streaming Sink	Double-click to
out0	Avalon Streaming Source	alu0_out

ST_ALU_1	ST_ALU_component	
in0	Avalon Streaming Sink	Double-click to
in1	Avalon Streaming Sink	Double-click to
out0	Avalon Streaming Source	alu1_out



# Проверьте систему

- Убедитесь в том, что:
  - Ваша система выглядит так же, как представленная на рисунке
  - Закладка сообщений (Messages) содержит только одно информационное сообщение.

The screenshot displays the Intel Quartus II software interface. The top section shows the 'Connections' tab, which lists various components and their connections. The components listed include:

- clk\_0**: Clock Source
- clk\_in**: Clock Input
- clk\_in\_reset**: Reset Input
- clk\_reset**: Reset Output
- my\_masterA**: my\_masterA\_component
- MM\_Stream\_source**: MM\_Stream\_source\_com...
- my\_masterB**: my\_masterB\_component
- st\_splitter\_0**: Avalon-ST Splitter
- st\_delay\_0**: Avalon-ST Delay
- ST\_ALU\_0**: ST\_ALU\_component
- ST\_ALU\_1**: ST\_ALU\_component

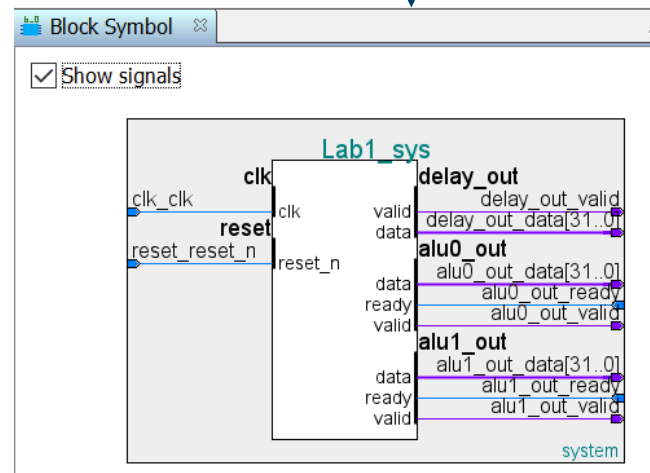
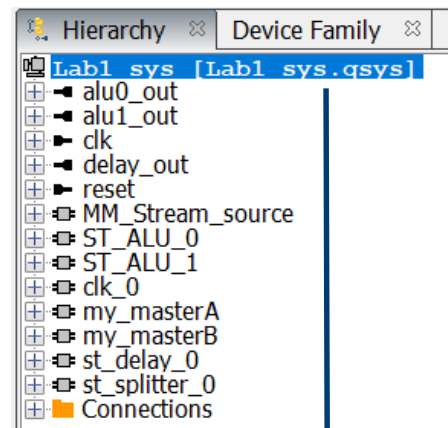
The bottom section shows the 'Messages' tab, which contains a single message:

Type	Path	Message
1 Info Message	Lab1_sys.st_splitter_0.out4/st_delay_0.in	The source has a ready signal of 1 bits, but the sink does not. Avalon-ST Adapter will be inserted.



# Анализ системы

- Выполните команду: меню View=>Hierarchy
- В окне закладки Hierarchy выделите систему Lab1\_sys [Lab1\_sys.qsys]
- Выполните команду: меню View=>Block Symbol
- Убедитесь, что Ваш символ системы соответствует представленному на рисунке





# Анализ системы

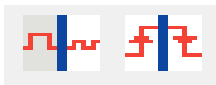
- Выполните команду: меню System => Show System with Platform Designer Interconnect  
сравните созданную Вами систему и систему с модулями вставленными PD:
  - Какие модули были добавлены? Зафиксируйте их имена.
  - Обратите внимание на наличие модуля `rst_controller` – это адаптер сигнала Reset

- Закройте окно закладки

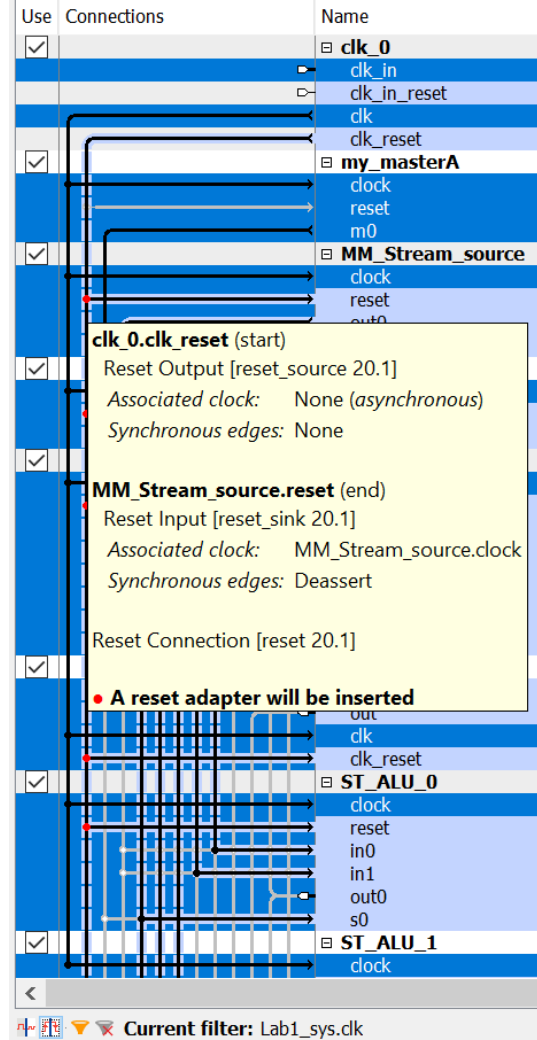
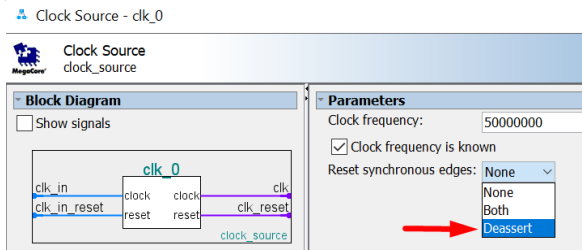
Use	Connections	Name	Description
<input checked="" type="checkbox"/>		<input type="checkbox"/> mm_interconnect_0	MM Interconnect
		clk_0_clk	Clock Input
		my_masterA_reset_reset_bridge_in_reset	Reset Input
		my_masterA_m0	Avalon Memory Mapped ...
<input checked="" type="checkbox"/>		<input type="checkbox"/> mm_interconnect_1	MM Interconnect
		clk_0_clk	Clock Input
		my_masterB_reset_reset_bridge_in_reset	Reset Input
		my_masterB_m0	Avalon Memory Mapped ...
<input checked="" type="checkbox"/>		<input type="checkbox"/> avalon_st_adapter	Avalon-ST Adapter
		in_clk_0	Clock Input
		in_rst_0	Reset Input
		in_0	Avalon Streaming Sink
<input checked="" type="checkbox"/>		<input type="checkbox"/> rst_controller	Merlin Reset Controller
		reset_in0	Reset Input
		clk	Clock Input
		reset_out	Reset Output

# Анализ системы

- Выполните команду: меню View => Clock domains Beta
- Выберите режим отображения Reset
- В столбце Connections отображаются красные точки => есть проблемы подключения:
  - Выход clk\_0.clk\_reset асинхронный
  - Входы clk\_reset всех модулей – синхронны.=> Поставлен адаптер Reset (видели на предыдущем слайде)
- Следует внести исправление в модуль clk\_0:
  - Надо выбрать модуль => правая клавиша мыши => Edit
  - Установить Deassert
  - Нажать Finish
- Красные точки исчезнут



Clocks Resets

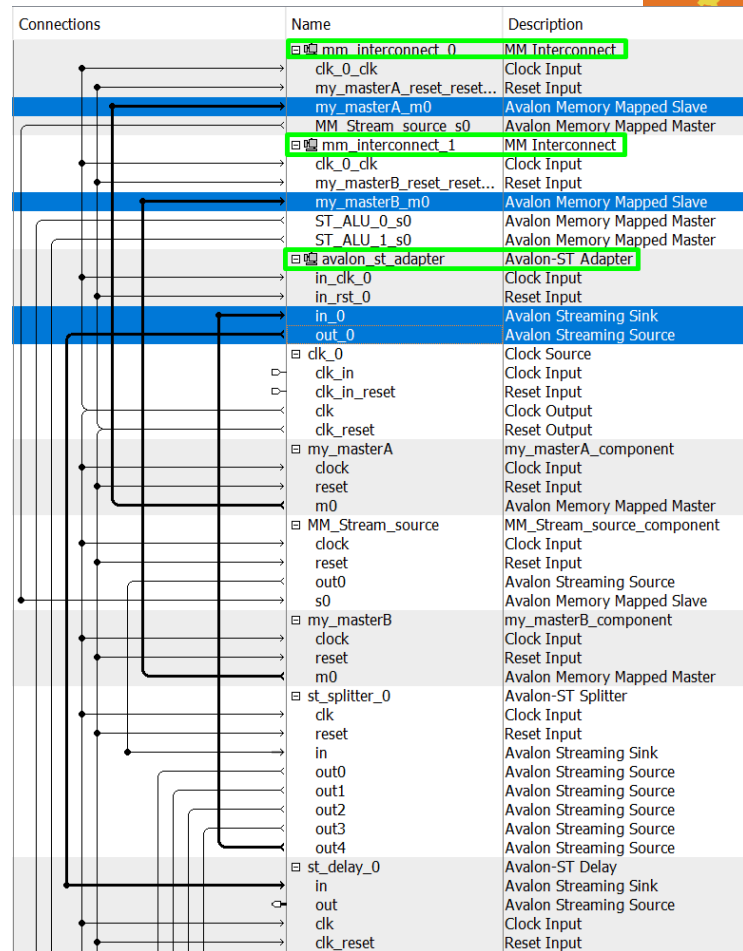




# Анализ системы

- Еще раз выполните команду: меню System => Show System with Platform Designer Interconnect
- Убедитесь в том, что адаптера Reset нет в системе.
- Назначение добавленных модулей
  - mm\_interconnect\_0 система межсоединений для Ведущего my\_masterA
  - mm\_interconnect\_1 система межсоединений для Ведущего my\_master
  - avalon\_st\_adapter – адаптер между модулем st\_splitter\_0 и st\_delay\_0
    - Этот адаптер потребовался в системе т.к. у модуля нет выхода Ready, а у модуля st\_splitter\_0 есть – на закладке сообщений есть сообщение с информацией

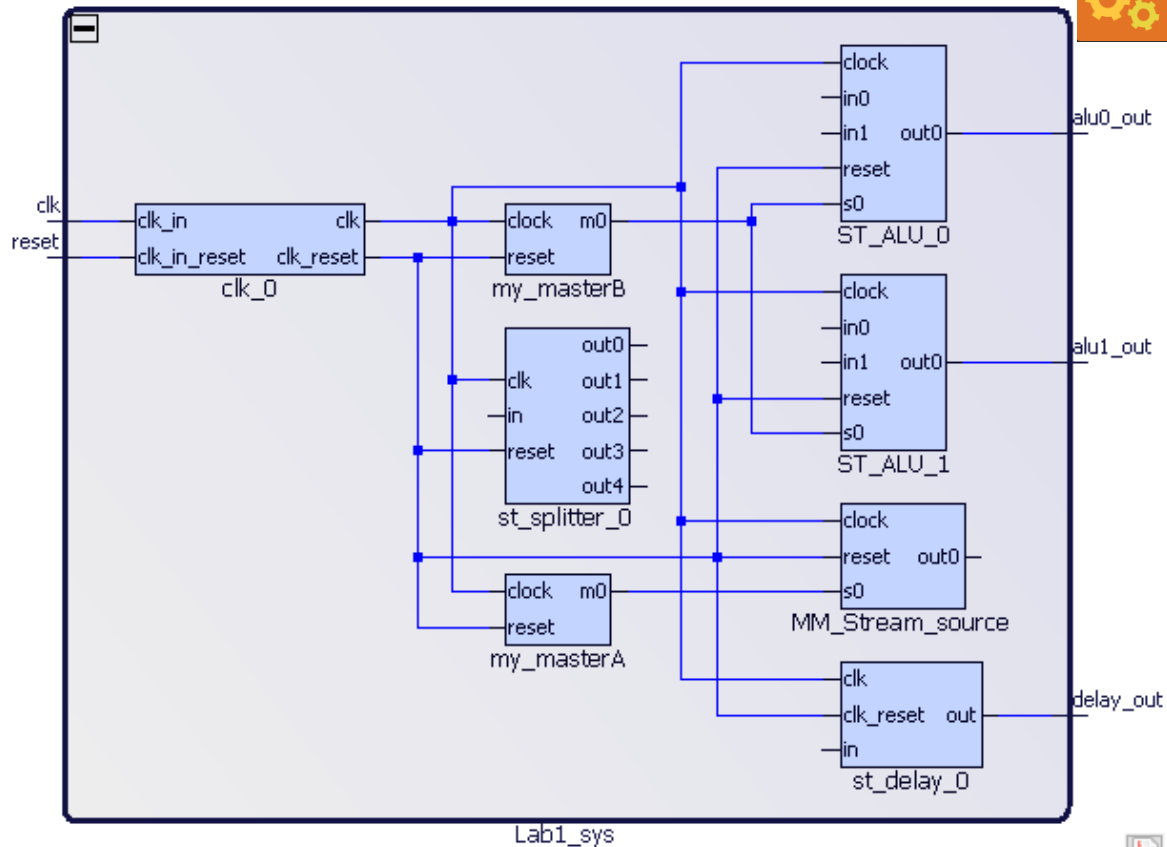
Type	Path	Message
1	Info Message	
1	Lab1_sys.st_splitter_0.out4/st_delay_0.in	The source has a ready signal of 1 bits, but the sink does not. Avalon-ST Adapter will be inserted.





# Анализ системы

- Выполните команду: меню View=>Schematic
- В поле фильтра введите
  - in
- Убедитесь в том, что система синхронизации и каналы ST Вашей системы подключены так же, как это изображено на рисунке

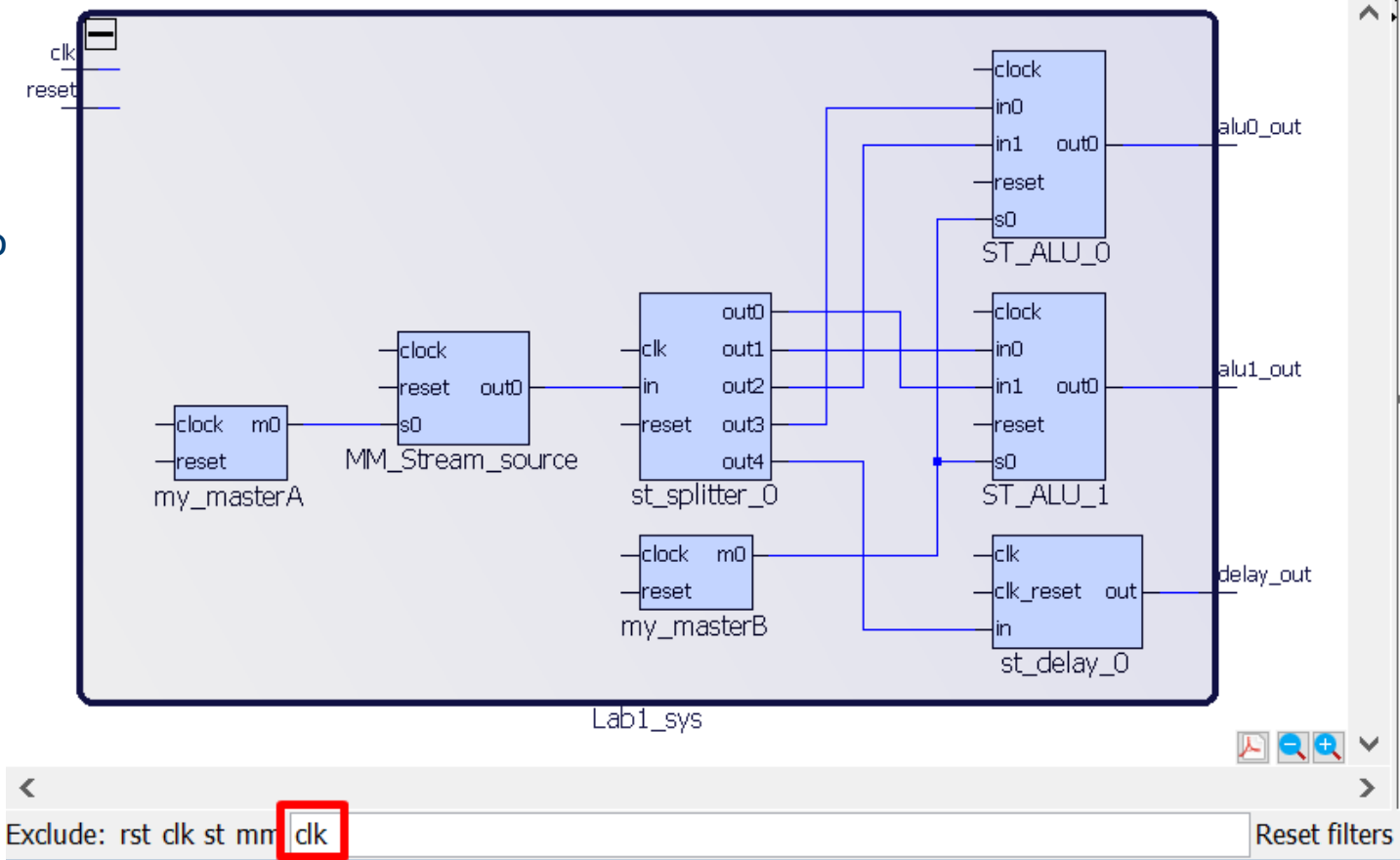


< Exclude: rst clk st mm **in**



# Анализ системы

- В поле фильтра введите
  - clk**
- Убедитесь в том, что шины Avalon MM в Вашей системе подключены так же, как рисунке
- Закройте окно закладки Schematic





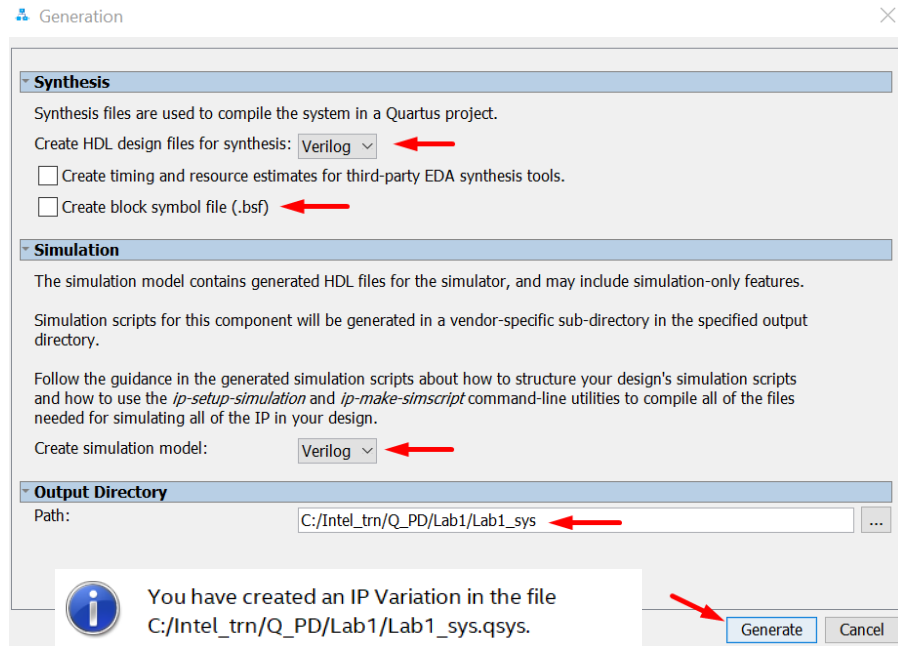
# Генерация системы

- В окне PD нажмите кнопку Generate HDL... (правый нижний угол окна)
- Установите режимы так, как показано на рисунке
- Нажмите кнопку Generate
- По окончании процедуры появится сообщение

✔ Generate: completed successfully.

– Нажмите кнопку Close

- В окне PD нажмите кнопку Finish (правый нижний угол окна)
- Появится напоминание о необходимости подключить файлы к проекту пакета QP
- Нажмите кнопку ОК.



To add this IP to your Quartus project, you must manually add the .qip and .sip files after generating the IP core.

The .qip will be located in  
<generation\_directory>/synthesis/Lab1\_sys.qip

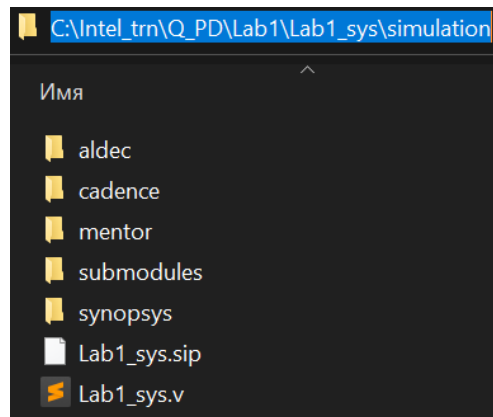
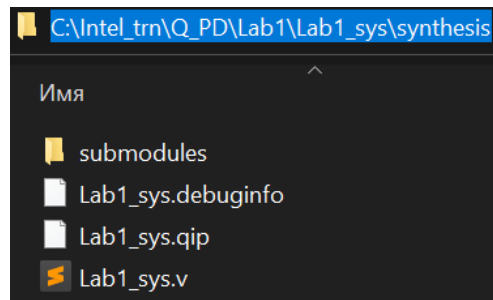
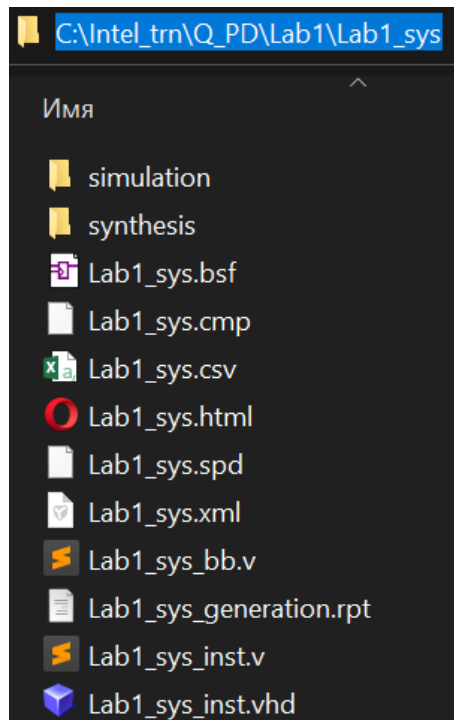
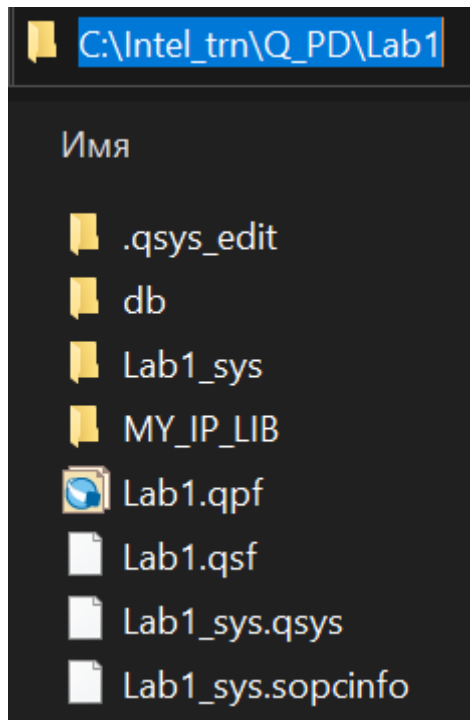
The .sip will be located in  
<generation\_directory>/simulation/Lab1\_sys.sip





# Анализ рабочей папки проекта

- Откройте рабочую папку проекта и проведите анализ файлов и папок.
  - Сравните с файлами в рабочей папке проекта на Вашем ПК – они должны совпадать





*Лабораторная 1*  
**ЗАВЕРШЕНА!**