

# Приложение Platform Designer

# Приложение Platform Designer

## Часть 2

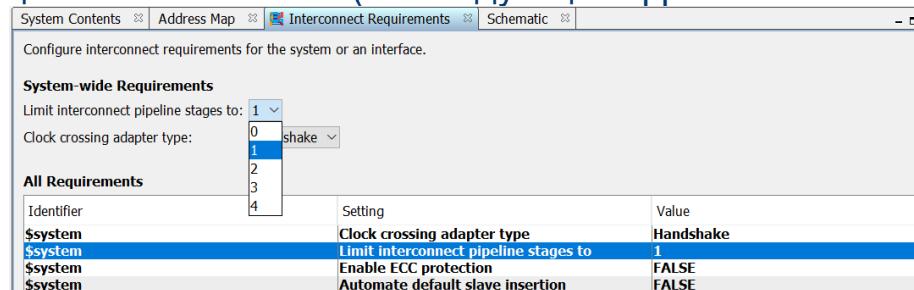
### Интерфейсы и компоненты шины Avalon-ST

# План

- Интерфейсы – общие сведения
- Интерфейсы и компоненты интерфейсов Clock и Reset
- Интерфейсы шины Avalon-ST
- Компоненты шины Avalon-ST
- Лабораторная 2

# Система соединений (Interconnect) компонентов в PD

- Все соединения компонентов автоматически создаются при генерации системы
- Структура соединений базируется на связях, заданных в PD и установках/настройках компонентов и системы
- Система соединений
  - включает:
    - Стандартные интерфейсы
    - Встроенные компоненты
    - Внутренние ресурсы, реализации соединений Network-on-chip (NoC)
  - оптимизируется по быстродействию и производительности
  - Позволяет осуществлять за 1 такт (на следующем фронте тактового сигнала) Запись и чтение данных



**Закладка *Interconnect Requirements* (меню View)**  
позволяет изменить уровень  
конвейеризации  
(задержка ⇔ частота)

# Интерфейсы PD

- В PD определены и поддерживаются интерфейсы:

- Clock
- Reset
- Avalon®-ST Interfaces
- Avalon-MM Interfaces
- Arm\* AMBA\* AXI Interface

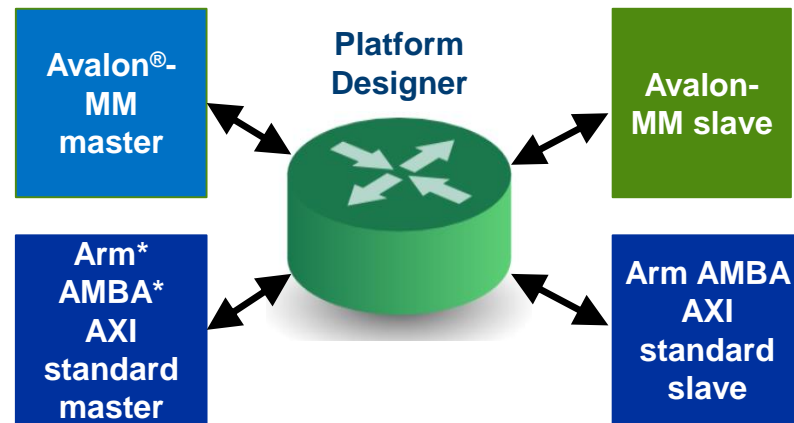
- Обеспечивается совместимость между IP блоками

- Может быть подключен любой компонент, поддерживающий стандартный интерфейс

- Упрощает разработку и верификацию

- Поведение сигналов определено интерфейсом

- Упрощает документирование проекта

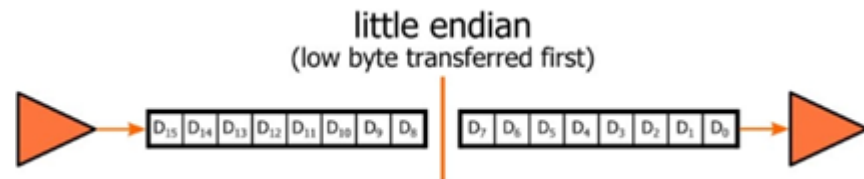


*PD позволяет согласовать между собой интерфейсы*

# Особенности интерфейсов Avalon

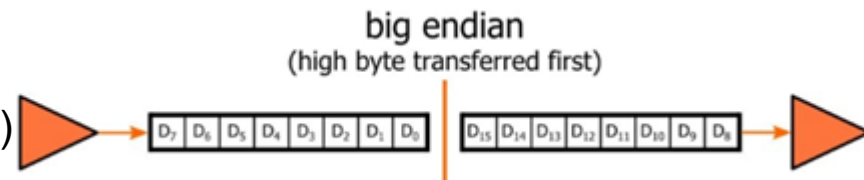
- Интерфейс Avalon-MM (memory-mapped)

- Little endian
- Используется для передачи
  - Управляющих/статусных данных (Control plane)
  - Информационных данных (Data plane)
- Ведущий (Master interface) формирует запросы Записи и чтения по адресу Ведомого (Slave interface)



- Интерфейс Avalon-ST (streaming)

- Big endian (по умолчанию, можно изменить)
- Используется для передачи
  - Информационных данных (Data plane)
- Источник (Source interface) отправляет данные приемнику (Sink interface)
  - один Источник => один Приемник (point-to-point)



# План

- Интерфейсы – общие сведения
- Интерфейсы и компоненты интерфейсов Clock и Reset
- Интерфейсы шины Avalon-ST
- Компоненты шины Avalon-ST
- Лабораторная 2

# Интерфейс Clock Interface

Интерфейс определяет тактовый сигнал (или тактовые сигналы), используемые компонентами и системой соединений

- Компонент может иметь;
  - Clock input (**sink**) interface
    - Тактовый вход компонента
  - Clock output (**source**) interface
    - Выход тактового сигнала компонентов (IP) формирующих тактовые сигналы для синхронизации других компонентов в системе
  - Или оба интерфейса: sink и source
  - Любой из интерфейсов (sink и source) может быть **экспортирован** (реализован как вывод системы)

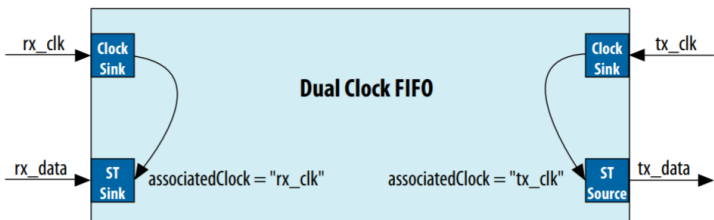
Name	Description	Export
clk_0	Clock Source	
clk_in	Clock Input	clk
clk_in_reset	Reset Input	reset
clk	Clock Output	Double-
clk_reset	Reset Output	Double-
my_masterA	my_masterA_component	
clock	Clock Input	Double-
reset	Reset Input	Double-
m0	Avalon Memory Mapped ...	Double-
MM Stream source	MM_Stream_source_com...	
clock	Clock Input	Double-
reset	Reset Input	Double-
out0	Avalon Streaming Source	Double-
s0	Avalon Memory Mapped ...	Double-

*Все компоненты в системе синхронны. Все передачи синхронизируются тактовым сигналом (или тактовыми сигналами)*



# Привязка интерфейса Clock

Все интерфейсы имеют параметр *associatedClock*, определяющий какой тактовый вход компонента будет синхронизировать интерфейс



Connections	Name	Description	Export	Clock
	clk_0	Clock Source		
	clk_in	Clock Input	clk	exported [clk_in]
	clk_in_reset	Reset Input	reset	
	clk	Clock Output	Double-click to	clk_0
	clk_reset	Reset Output	Double-click to	clk_0
	my_masterA	my_masterA component		
	clock	Clock Input	Double-click to	clk_0
	reset	Reset Input	Double-click to	[clock]
	m0	Avalon Memory Mapped ...	Double-click to	[clock]
	MM_Stream_source	MM_Stream_source_com...		
	clock	Clock Input	Double-click to	clk_0
	reset	Reset Input	Double-click to	[clock]
	out0	Avalon Streaming Source	Double-click to	[clock]
	s0	Avalon Memory Mapped ...	Double-click to	[clock]
	my_masterB	my_masterB component		
	clock	Clock Input	Double-click to	clk_0
	reset	Reset Input	Double-click to	[clock]
	m0	Avalon Memory Mapped ...	Double-click to	[clock]
	st_splitter_0	Avalon-ST Splitter		
	clk	Clock Input	Double-click to	clk_0
	reset	Reset Input	Double-click to	[clk]
	in	Avalon Streaming Sink	Double-click to	[clk]
	out0	Avalon Streaming Source	Double-click to	[clk]
	out1	Avalon Streaming Source	Double-click to	[clk]
	out2	Avalon Streaming Source	Double-click to	[clk]
	out3	Avalon Streaming Source	Double-click to	[clk]
	out4	Avalon Streaming Source	Double-click to	[clk]

# Интерфейс Reset Interfaces

- Компонент может иметь;

- Reset input (**sink**) interface

- Сбрасывает другие интерфейсы или внутреннюю логику компонента в начальное состояние

- Должен быть связан с интерфейсом Clock

- Reset output (**source**) interface

- Выход компонента, формирующего сигналы сброса для других компонентов

- Должен быть связан с интерфейсом Clock

- Или оба интерфейса: **sink** и **source**

- Любой из интерфейсов (sink и source) может быть **экспортирован** (реализован как вывод системы)

Интерфейс Reset используется для сброса компонентов в начальное состояние

Connections	Name	Description	Export	Clock
	clk_0	Clock Source		
	clk_in	Clock Input		
	clk_in_reset	Reset Input		
	clk	Clock Output		
	clk_reset	Reset Output	clk_reset	exported [clk_in] clk_0
	my_masterA	my_masterA_component		
	clock	Clock Input		
	reset	Reset Input		
	m0	Avalon Memory Mapped ...		
	MM_Stream_source	MM_Stream_source_com...		
	clock	Clock Input		
	reset	Reset Input		
	out0	Avalon Streaming Source		
	s0	Avalon Memory Mapped ...		

*Все интерфейсы имеют параметр associatedReset, определяющий какой вход Reset компонента будет сбрасывать интерфейс (на закладке System Contents привязка интерфейса Reset не показана)*

# Сигналы интерфейсов Reset

## Интерфейс Reset input (sink)

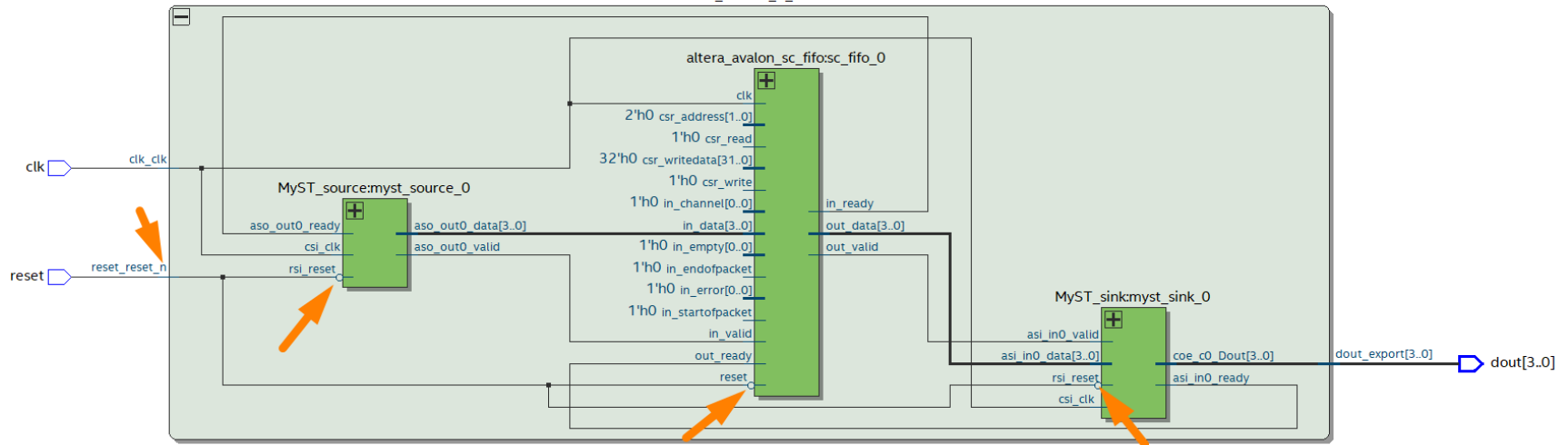
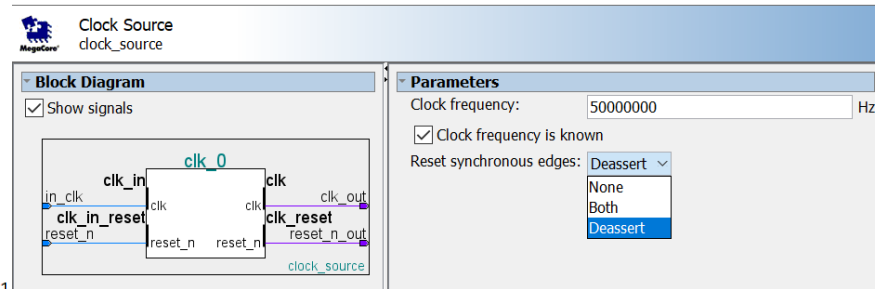
Имя	Направление	Описание
reset reset_n	Input	Активный уровень 1 Активный уровень 0

## Интерфейсы Reset output (source)

Имя	Направление	Описание
reset reset_n	Output	Активный уровень 1 Активный уровень 0

# Базовые IP: Компонент Clock Source

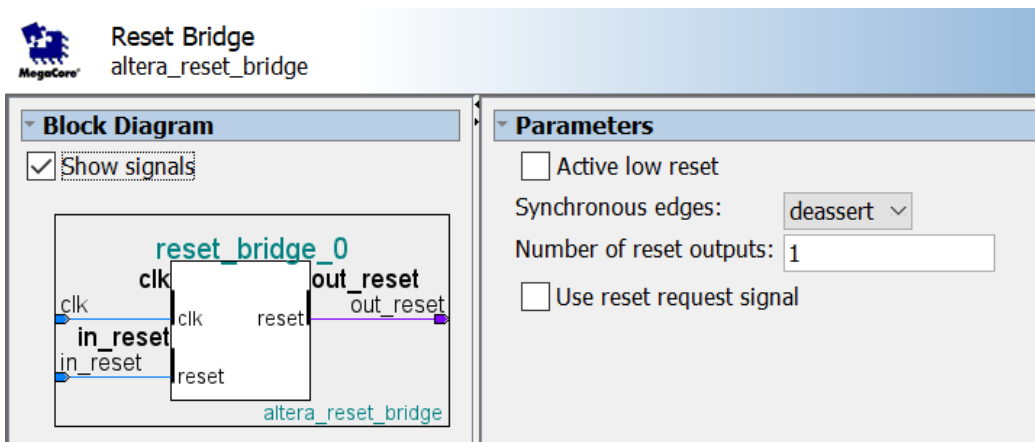
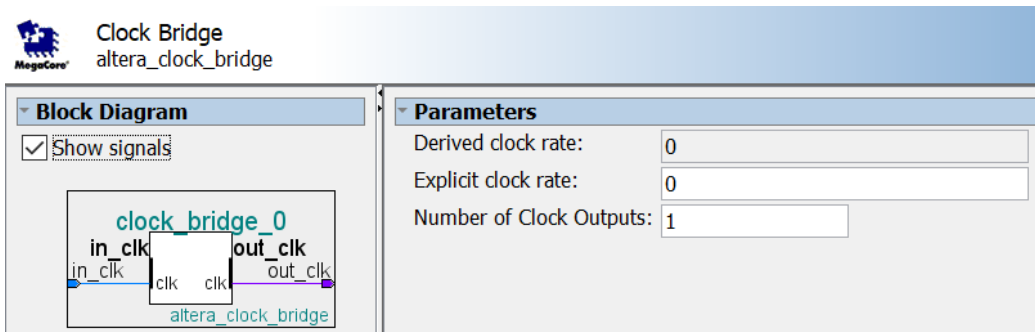
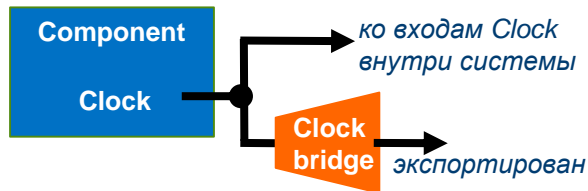
- Подключает к системе внешние clock и reset входы
- Позволяет задать требуемую частоту
- Позволяет синхронизировать сигнал reset



По «умолчанию» PD предполагает, что сигнал reset, приходящий на внешний вход системы, имеет активный 0. Поэтому устанавливается имя сигнала reset\_n. Поэтому, если в компонентах системы для Reset используется активная 1, то на входах компонентов будут установлены инверторы.

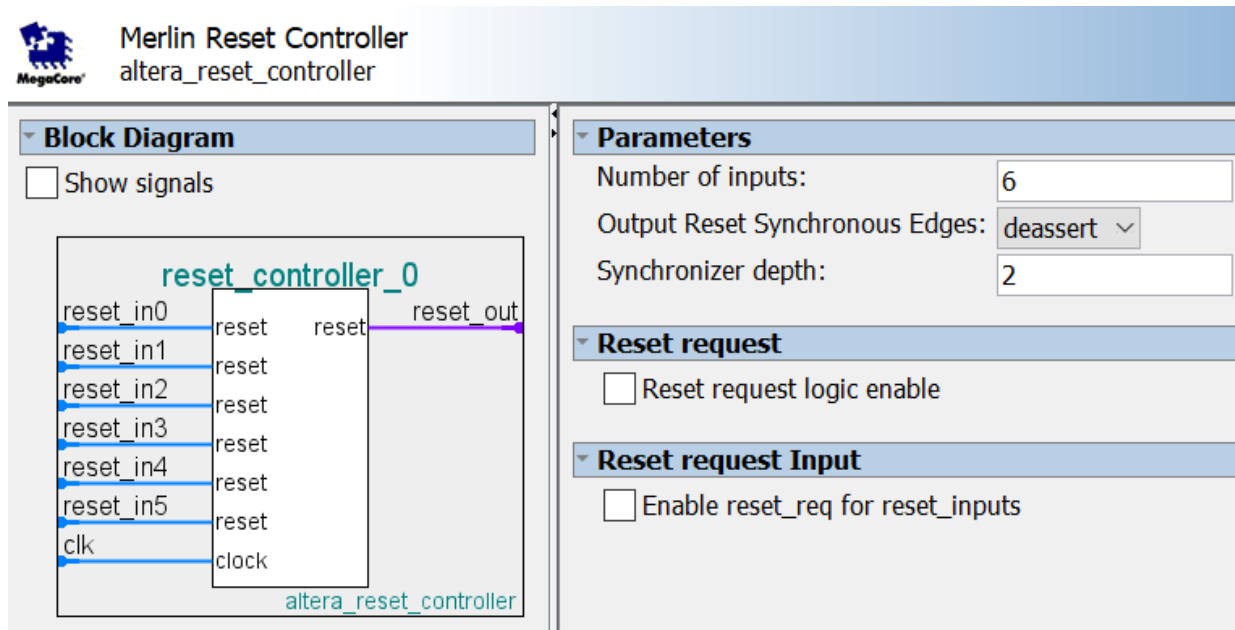
# Базовые IP: Компоненты Clock и Reset Bridge

- Используются, когда выход компонента, на котором внутри системы формируется сигнал Clock или Reset, должен быть одновременно подключен к входам внутри системы и экспортирован
  - Интерфейс не может быть одновременно:  
экспортирован и подключен внутри системы



# Компонент Reset Controller


- Reset Controller
  - Собирает по ИЛИ интерфейсы Reset ORs
  - Добавляет этапы синхронизации для исключения Метастабильности



# Компонент Reset Sequencer

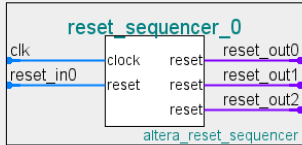
## Reset Sequencer

- Позволяет задать последовательность установки/снятия сигналов Reset
- Опциональный CSR slave interface для программного управления сбросами

 Reset Sequencer  
altera\_reset\_sequencer

**Block Diagram**

☐ Show signals



**Sequencer Parameters**

Number of reset outputs: 3  
Number of reset inputs: 1  
Minimum reset assertion time: 0  
☐ Enable Reset Sequencer CSR

**Sequence Setup**

- ASRT Seq# - Determine the sequence order of reset assertion
- ASRT Cycle# - Number of cycles to wait before assertion of this reset
- DSRT Seq# - Determine the sequence order of reset de-assertion
- DSRT Cycle# - Number of cycles to wait before deasserting this reset
- When USE\_DSRT\_QUAL=1. DSRT Cycle# refers to the # of cycles to deglitch the input qual signal

reset_out#	ASRT Seq#	ASRT Delay Cycle#	DSRT Seq #	DSRT Delay Cycle#	USE_DSRT_QUAL
reset_out0	0	1	0	0	0
reset_out1	1	0	1	0	0
reset_out2	2	0	2	0	0

Please check below for expected generated sequence :

Assertion Sequence:

De-assertion Sequence:

# План

- Интерфейсы – общие сведения
- Интерфейсы и компоненты интерфейсов Clock и Reset
- Интерфейсы шины Avalon-ST
- Компоненты шины Avalon-ST
- Лабораторная 2



# Интерфейсы Avalon®-ST Interface

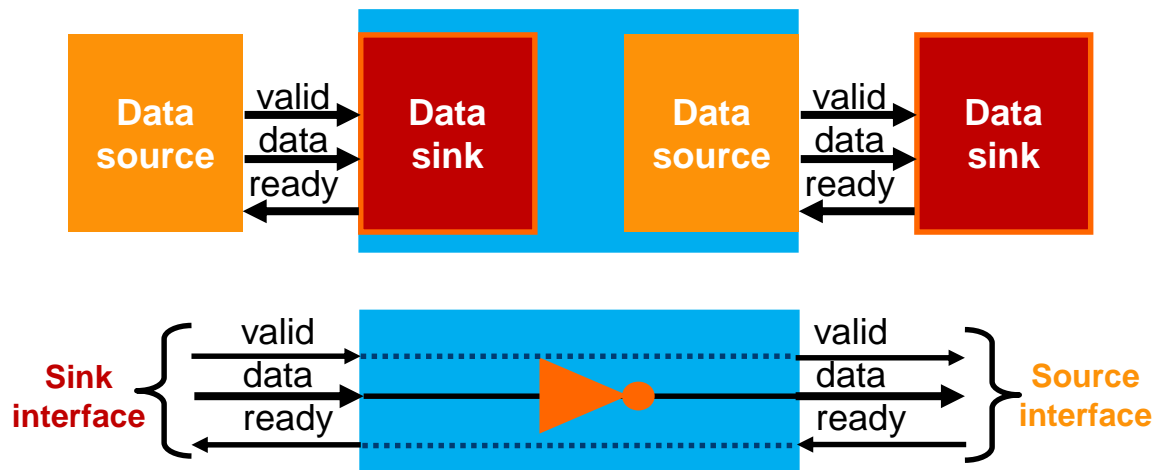
- Особенности Avalon-ST интерфейсов
  - Однонаправленные
  - Соединения точка-точка (point-to-point)
  - Полностью синхронная передача
- Обеспечивают высокую пропускную способность и малую задержку
- Интерфейс **Source (Источник)**: формирует данные по фронту привязанного тактового сигнала
- Интерфейс **Sink (Приемник)**: записывает данные по фронту привязанного тактового сигнала
- Термин **Channel**: Физический или логический путь от Источника к Приемнику
- Термин **Symbol**: Минимальный элемент данных, который может быть передан (обычно это 8 бит)
- Термин **Backpressure**: любой из интерфейсов может приостановить передачу данных при использовании соответствующих сигналов интерфейса
  - Если не поддерживается, то интерфейс должен обеспечивать передачу/прием данных на каждом такте.
- Формат передаваемых данных определяется настройками интересов в компоненте

# Сигналы интерфейсов Avalon-ST

Имя	Разряды	Направление	Описание
Основные сигналы			
ready	1	Sink → Source	Лог. 1 – приемник готов принимать данные (м.б. использован для backpressure)
valid	1	Source → Sink	Лог. 1 – готовность всех выходных данных источника
data	1-4096	Source → Sink	Передаваемые данные
channel	1-128	Source → Sink	Номер канала передаваемых данных (если поддерживается несколько каналов)
error	1-256	Source → Sink	Битовая маска: отмечает ошибки, влияющие на передаваемые данные
Сигналы для пакетной передачи			
startofpacket	1	Source → Sink	Отмечает начало пакета
endofpacket	1	Source → Sink	Отмечает конец пакета
empty	1-5	Source → Sink	Указывает количество символов, которые остаются пустыми во время циклов, содержащих конец пакета.

# Пример модуля с интерфейсами Avalon-ST

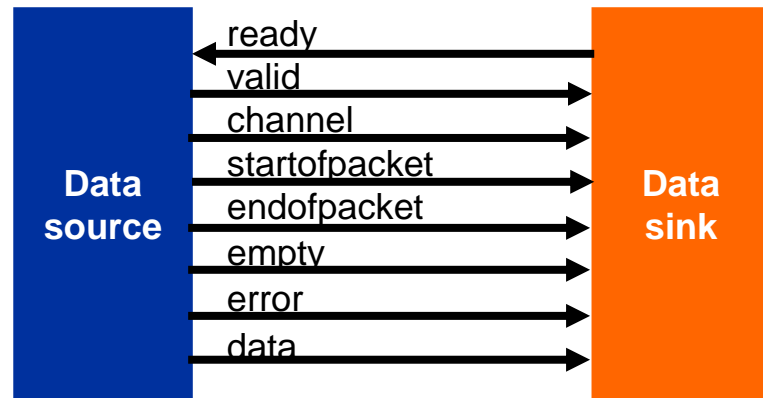
- Передается от Источника (source) к Приемнику (sink):
  - data – данные
  - valid – готовность данных
- Передается от Приемника (sink) к Источнику (source):
  - ready – сигнал готовности Приемника. Используется для приостановки Источника.



# Пример модуля с интерфейсами Avalon-ST

В примере используются сигналы для пакетной, многоканальной передачи с формированием признаков ошибок:

- `startofpacket` – отмечает начало пакета
- `endofpacket` – отмечает конец пакета
- `channel` – передается номер канала, которому принадлежат передаваемые данные
- `empty` – количество символов, которые остаются пустыми во время циклов, содержащих конец пакета.
- `error` – ошибки в передаваемых данных



# Параметры интерфейсов Avalon-ST

- **dataBitsPerSymbol:** число бит, определяющее символ данных
  - 1 - 512 (default: 8)
  - Пример: Если установить 16 bits/symbol, то 64-bit данные будут содержать 4 символа
- **SymbolPerBeats:** символов в одном такте обмена
  - 1 – 32 (default: 1)
  - Пример: Если установить 1 symbol/beat для 16 bits/symbol, то 64-bit данных будут переданы за 4 такта обмена
- **readyLatency:** определяет число тактов между установкой Ready приемником и установкой Valid источником
  - 0 – 8 (default: 0)
- **readyAllowance:** определяет число тактов, в течении которых sink может принимать данные, после снятия им сигнала Ready
  - 0 – 8 (default: 0)
- Описание всех параметров можно найти в **Avalon® Interface Specifications**

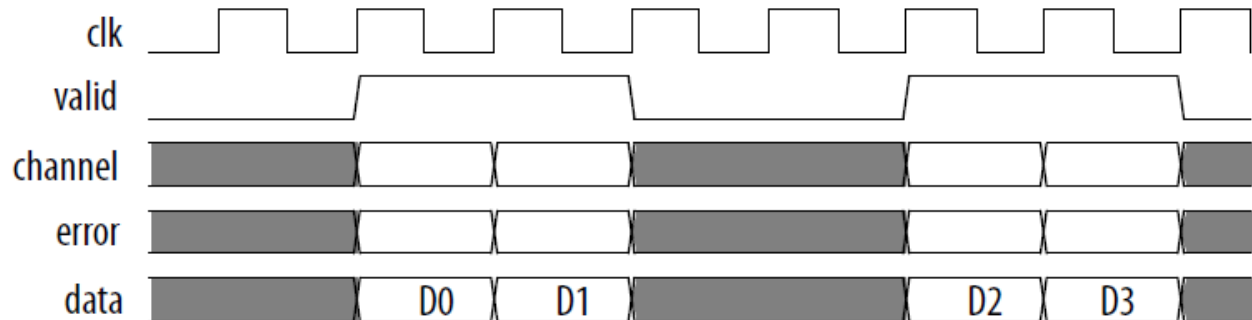
*Не для всех модулей можно явно установить все параметры. Если параметры не установлены – используются значения базовые (default) значения.*

# Пример передачи данных по Avalon-ST

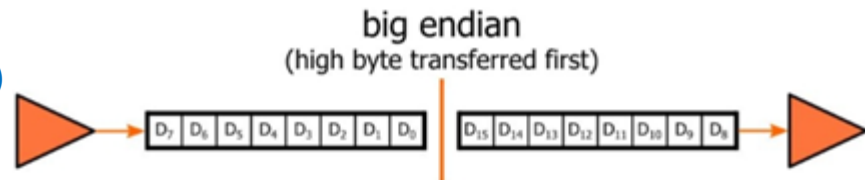
## Параметры компонентов

source => sink:

- **dataBitsPerSymbol: 8**
- **beatsPerCycle: 1**
- **data: 8**
- **readyLatency: 0**
- **readyAllowance: 0**
- **Backpressure: нет**  
(сигнал Ready не использован)



Параметр **firstSymbolInHigh OrderBits** (default = true) позволяет изменить этот порядок

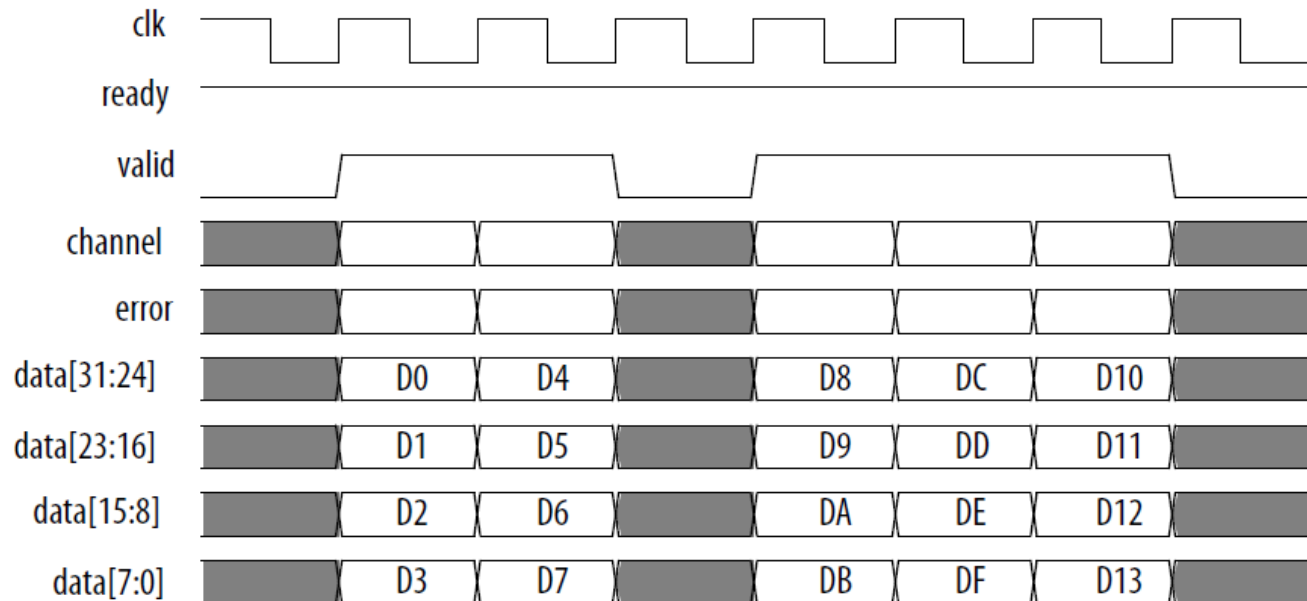


# Пример передачи данных по Avalon-ST

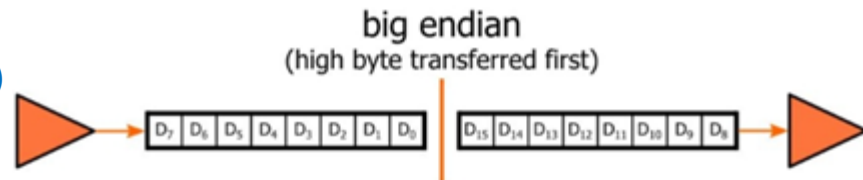
## Параметры компонентов

source => sink:

- **dataBitsPerSymbol: 8**
- **SymbolPerBeats: 4**
- **data: 32**
- **readyLatency: 0**
- **readyAllowance: 0**
- **Backpressure: нет**  
(сигнал Ready = 1)



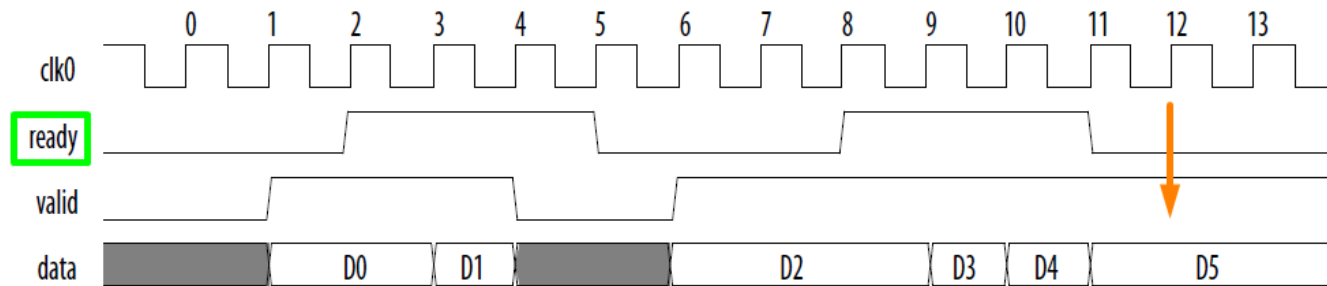
Параметр **firstSymbolInHigh OrderBits** (default = true) позволяет изменить этот порядок



# Пример передачи данных по Avalon-ST

Параметры компонентов source => sink:

- **dataBitsPerSymbol: 8**
- **SymbolPerBeats : 1**
- **data: 8**
- **readyLatency: 0**
- **readyAllowance: 0**
- **Backpressure: есть**

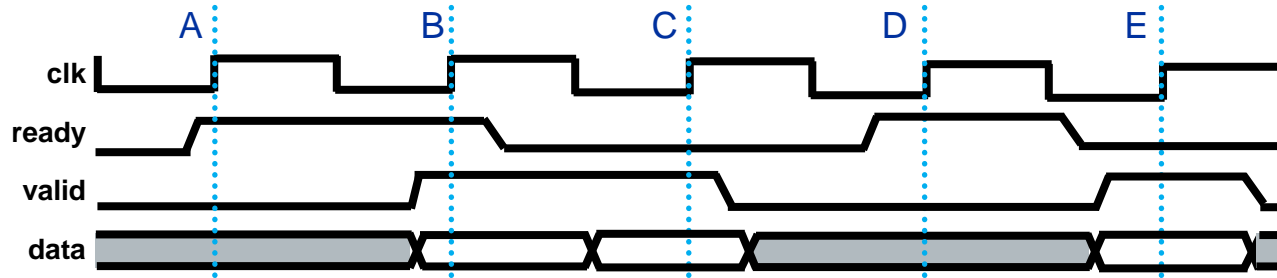




# Пример передачи данных по Avalon-ST

readylatency определяет задержку (число тактов) до истинной готовности Приемника (sink) к приему данных после установки сигнала ready

- Параметры: ready latency = readyAllowance = 1



- A: sink установил ready, source установит valid на следующем такте т.к. ready latency = 1
- B: source передает valid и data; sink продолжает держать сигнал ready
- C: sink снимает ready, но source передает valid и data т.к. readyAllowance = 1
- D: sink установил ready, source установит valid на следующем такте т.к. ready latency = 1
- E: sink снимает ready, но source передает valid и data т.к. readyAllowance = 1

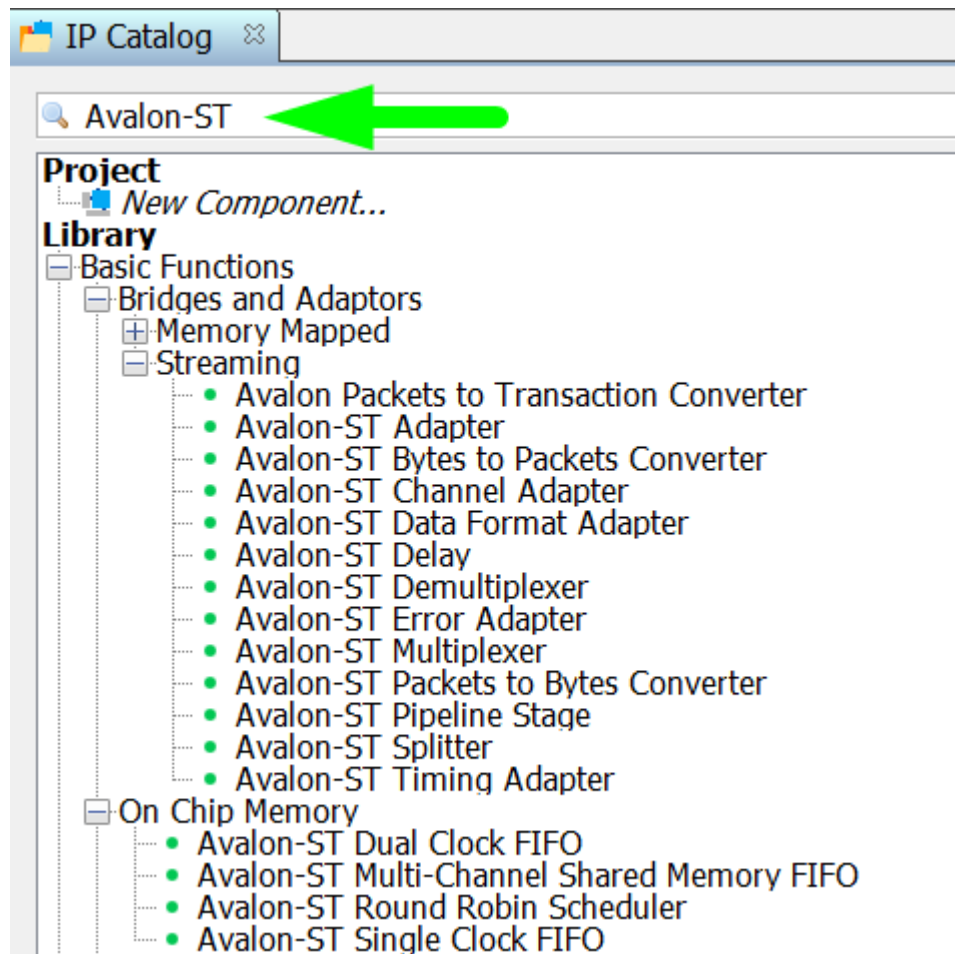
# План

- Интерфейсы – общие сведения
- Интерфейсы и компоненты интерфейсов Clock и Reset
- Интерфейсы шины Avalon-ST
- **Компоненты шины Avalon-ST**
- Лабораторная 2

# Компоненты Avalon-ST

Используются для преобразований над потоками передачей данных:

- Demultiplexer
- Multiplexer
- Adapter: data format, channel, timing, and error
  - Адаптеры могут добавляться либо автоматически, либо вручную.
- Delay
- Splitter
- Pipeline



# Адаптеры Streaming Adapters

Адаптеры могут добавляться либо автоматически, либо вручную

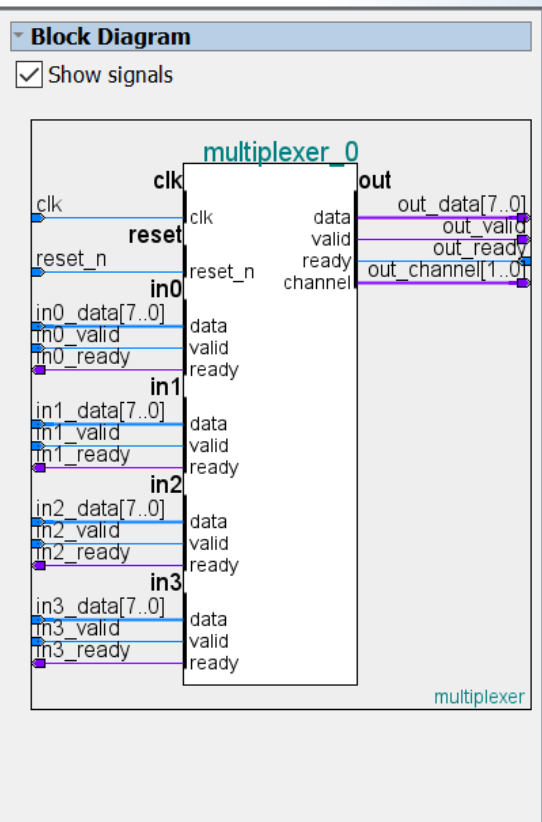
- Адаптер Data Format
  - Соединяет интерфейсы, имеющие разные параметры для цепей data (например – разную разрядность)
- Адаптер Timing
  - Соединяет интерфейсы, в которых: один из интерфейсов не использует сигнал ready; либо у них параметр readylatencies имеет разное значение
  - Добавляет компонент FIFO между source и sink для буферизации данных и задержки сигналов, реализующих backpressure (т.е. сигнала Ready)
- Адаптер Channel
  - Соединяет интерфейсы, имеющие разные параметры для режима канальной передачи (channel)
    - Например для каналов с разной разрядностью адаптер отбросит лишние биты и сформирует предупреждение

# Компонент Avalon-ST Multiplexer

- Принимает данные с sink интерфейсов и передает их на source интерфейс.
- Порядок опроса sink (по умолчанию) – Round-Robin
  - Берется следующий интерфейс, у которого есть данные (есть сигнал Valid)
  - Число тактов передачи каждого sink интерфейса задается как **Scheduling Size (Cycles)**.
  - На всех sink интерфейсах, кроме того, который передается, формируется **Ready=0** (**backpressed**)
- Выход Channel формирует номер передаваемого sink интерфейса.



Avalon-ST Multiplexer  
multiplexer



## Functional Parameters

Number of Input Ports:

Scheduling Size (Cycles):

☐ Use Packet Scheduling

☒ Use high bits to indicate source port

When selected, the high order bits of the output channel signal indicate the input interface the data came from, and the low order bits are those from the input interface's channel signal.

When not selected, the low order bits indicate the input interface, and the high order bits are passed through from the input.

## Input Interface

Data Bits Per Symbol:

Data Symbols Per Beat:

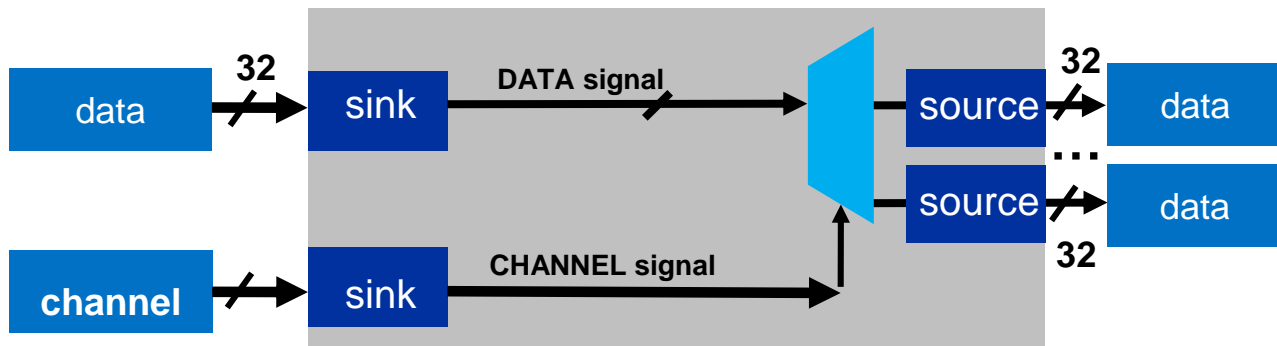
☐ Include Packet Support

When packets are supported, the startofpacket, endofpacket, and empty signals are used.

Channel Signal Width (bits):

Error Signal Width (bits):

# Канальная передача данных



# Компонент Avalon-ST Demultiplexer

- Принимает данные с sink интерфейса и передает их на один из source интерфейсов.
- Source интерфейс определяется данными на входе channel
- Выход Channel формирует номер передаваемого sink интерфейса.



Avalon-ST Demultiplexer  
demultiplexer

**Block Diagram**  
☒ Show signals

**Functional Parameters**  
Number of Output Ports: 4  
☒ Use High Channel Bits  
When selected, the high order bits of the input channel signal are used by the demultiplexing function, and the low order bits are passed to the output. When not selected, the low order bits are used, and the high order bits are passed through.  
**Input Interface**  
Data Bits Per Symbol: 8  
Data Symbols Per Beat: 2  
☐ Include Packet Support  
When packets are supported, the startofpacket, endofpacket, and empty signals are used.  
Channel Signal Width (bits): 4  
Error Signal Width (bits): 0

# Компонент Avalon-ST Single Clock FIFO

- Принимает данные с sink интерфейса и буферизирует их
- Передает данные на source интерфейс в режиме: первый пришел – первый вышел.
- Позволяет добавлять признак канала Channel



Avalon-ST Single Clock FIFO  
altera\_avalon\_sc\_fifo

**Block Diagram**  
☒ Show signals

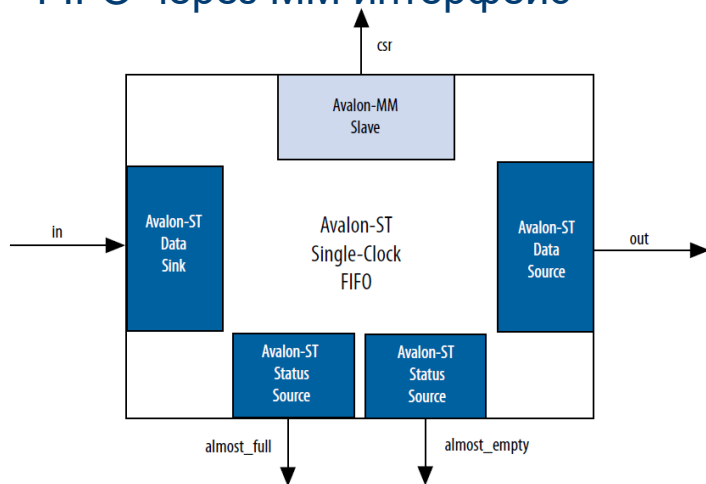
**Parameters**

Symbols per beat:	1
Bits per symbol:	4
FIFO depth:	16
Channel width:	0
Error width:	0
<input type="checkbox"/> Use packets	
<input type="checkbox"/> Use fill level	
<input type="checkbox"/> Use store and forward	
<input type="checkbox"/> Use almost full status	
<input type="checkbox"/> Use almost empty status	
<input type="checkbox"/> Enable explicit maxChannel	
Explicit maxChannel:	0

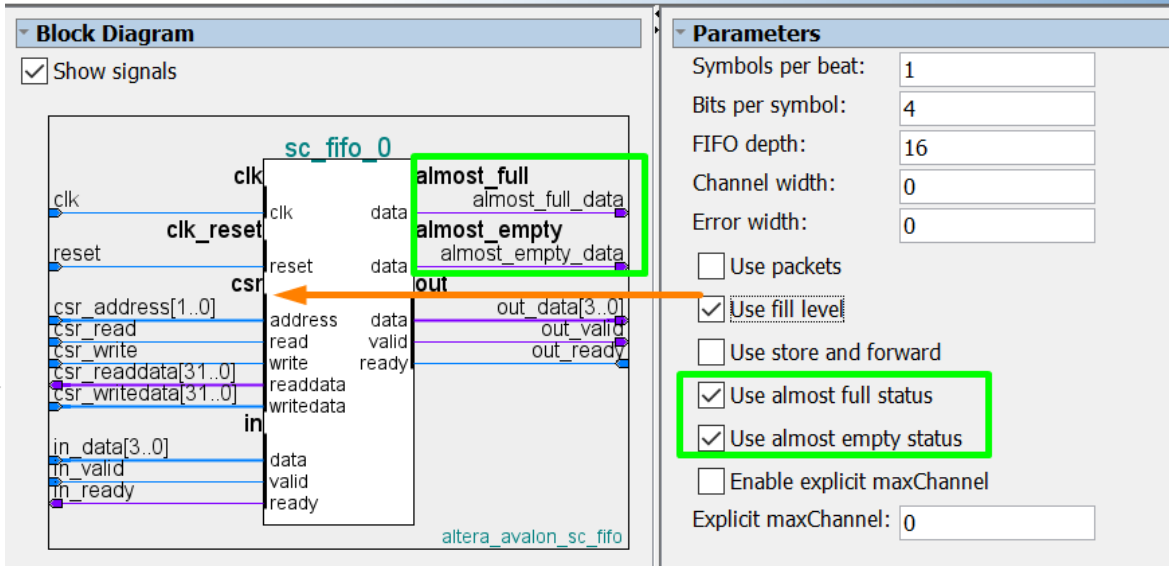


# Компонент Avalon-ST Single Clock FIFO (MM interface)

- Позволяет формировать признаки заполнения FIFO
- Позволяет читать число заполненных элементов FIFO через MM интерфейс

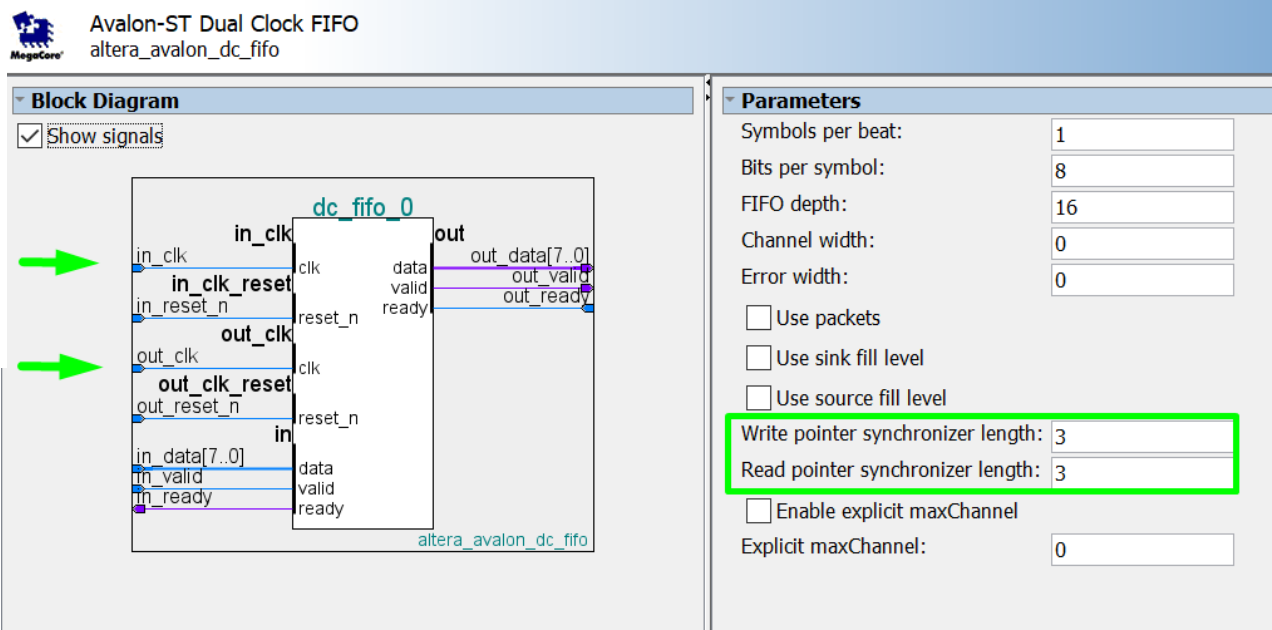
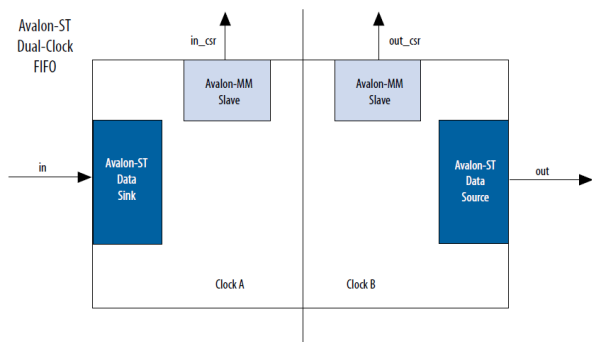


Avalon-ST Single Clock FIFO  
altera\_avalon\_sc\_fifo



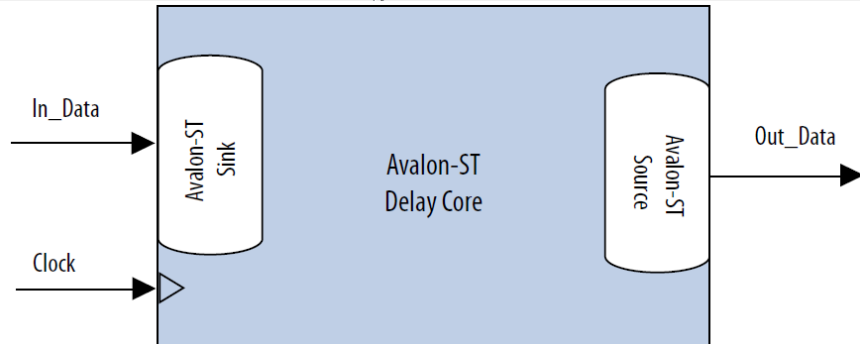
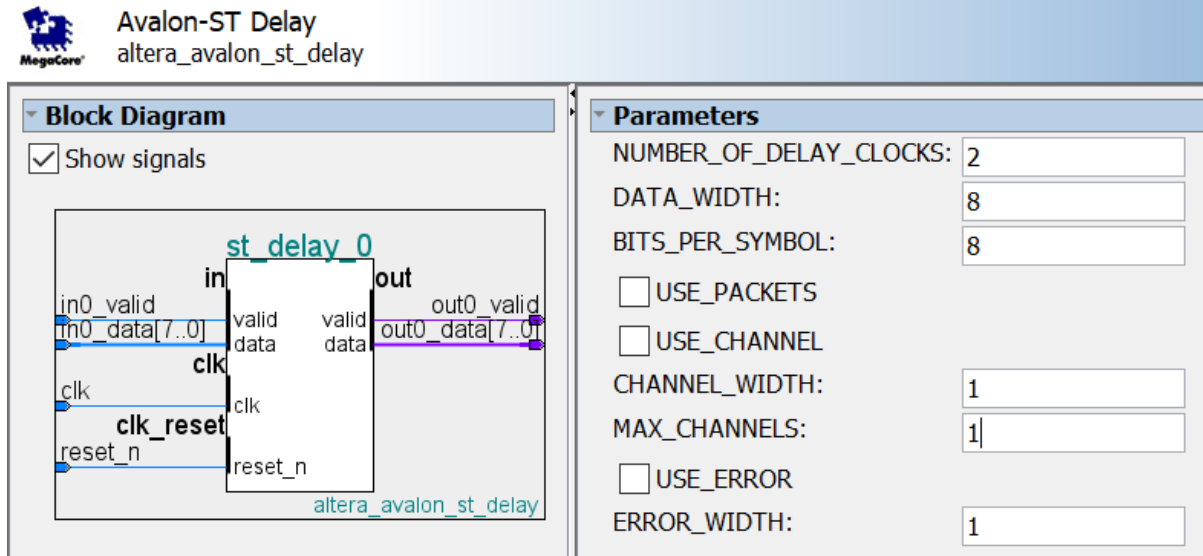
# Компонент Avalon-ST Dual Clock FIFO

- Аналогичен Avalon-ST Single Clock FIFO позволяет передавать данные между тактовыми доменами



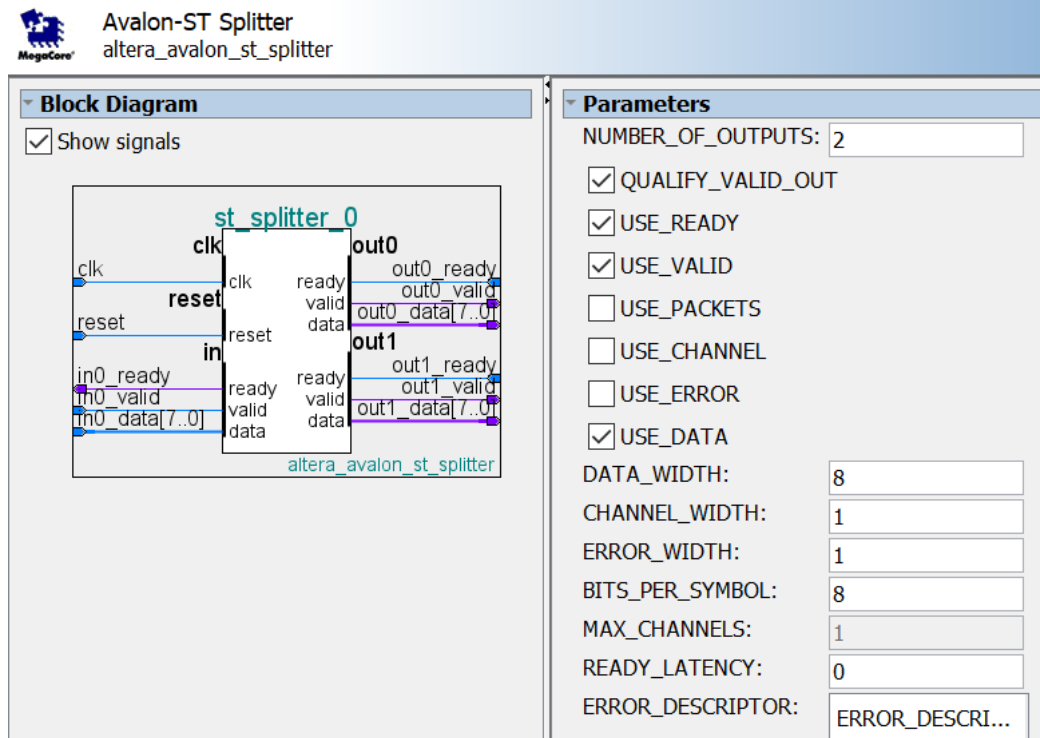
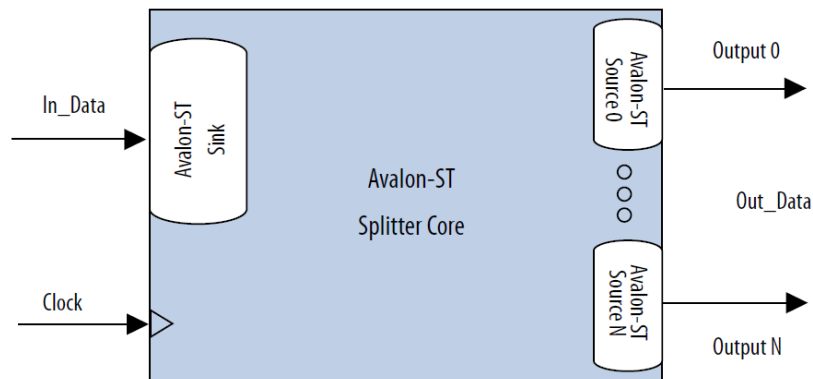
# Компонент Avalon-ST Delay Core

- Позволяет задержать передачу данных между sink и source на заданное число тактов
- Максимальное число тактов задержки – 16
- Не использует сигнал Ready
- Сигнал Valid передается от sink к source с заданной задержкой
- SymbolPerBeats: 1**



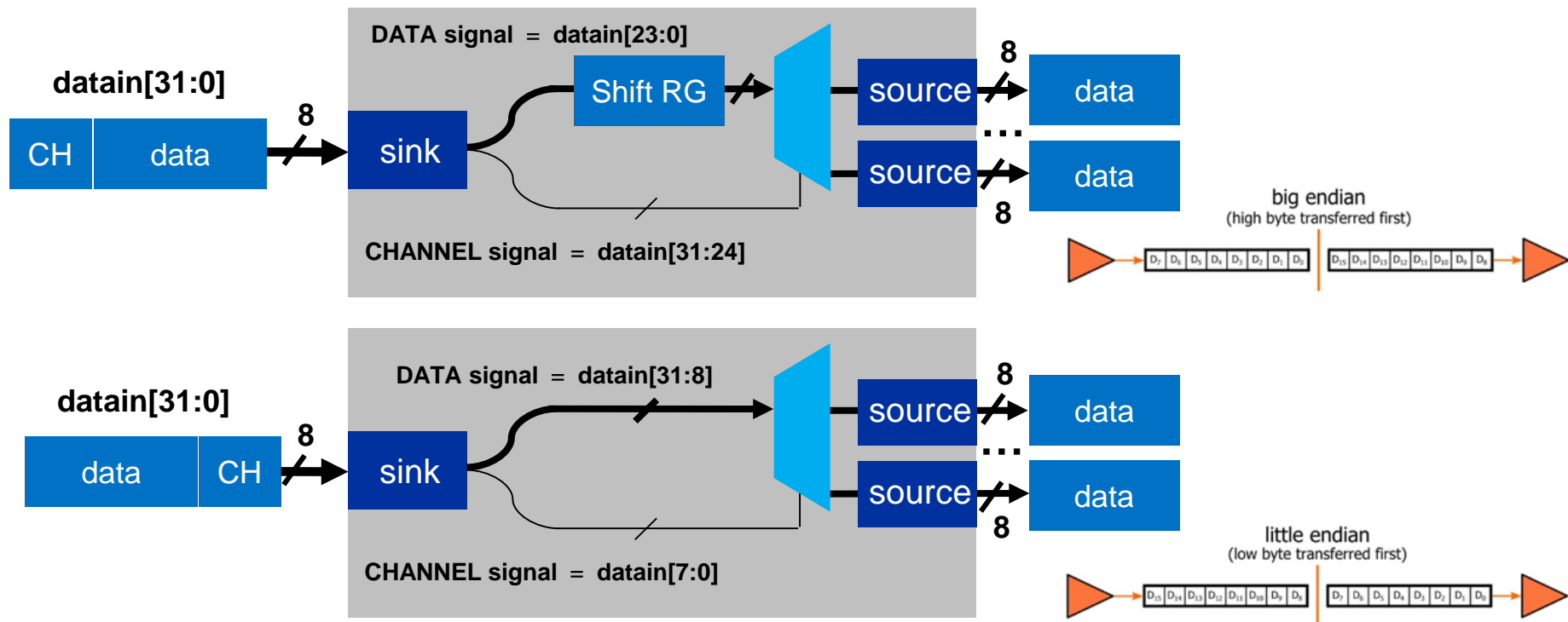
# Компонент Avalon-ST Splitter Core

- Позволяет дублировать передачу данных от sink на заданное число source
- Максимальное число source – 16
- Использование сигналов Ready, Valid настраивается



# Потоковая передача канальных данных

По умолчанию режим - big endian, но его можно поменять на little endian



# План

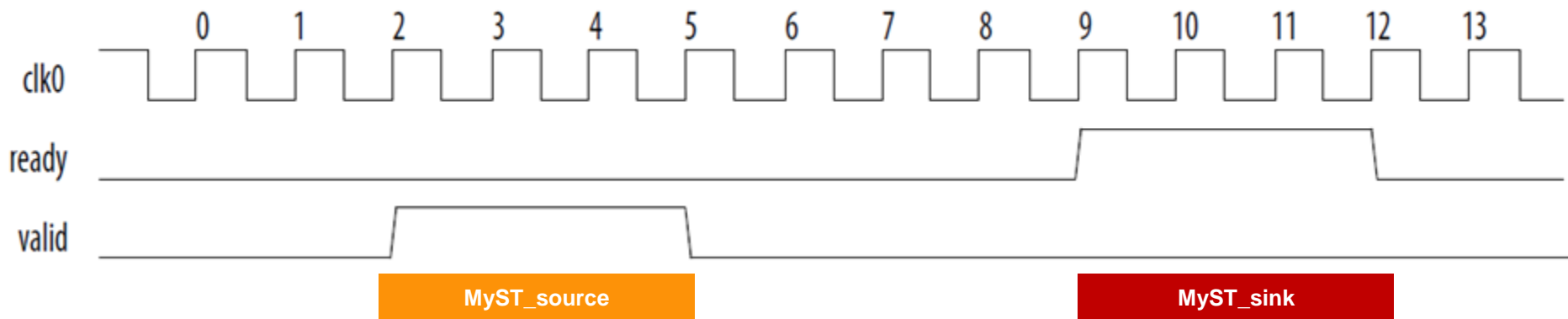
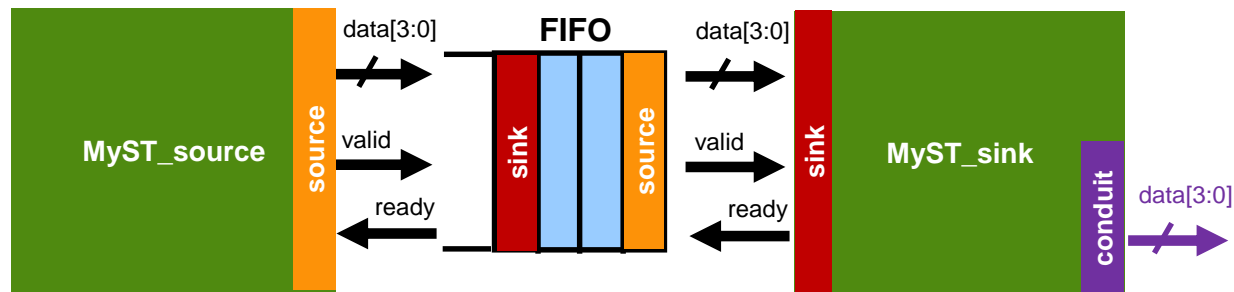
- Интерфейсы – общие сведения
- Интерфейсы и компоненты интерфейсов Clock и Reset
- Интерфейсы Avalon-ST
- Компоненты Avalon-ST
- Лабораторная 2



# Лабораторная 2



# Структура проекта





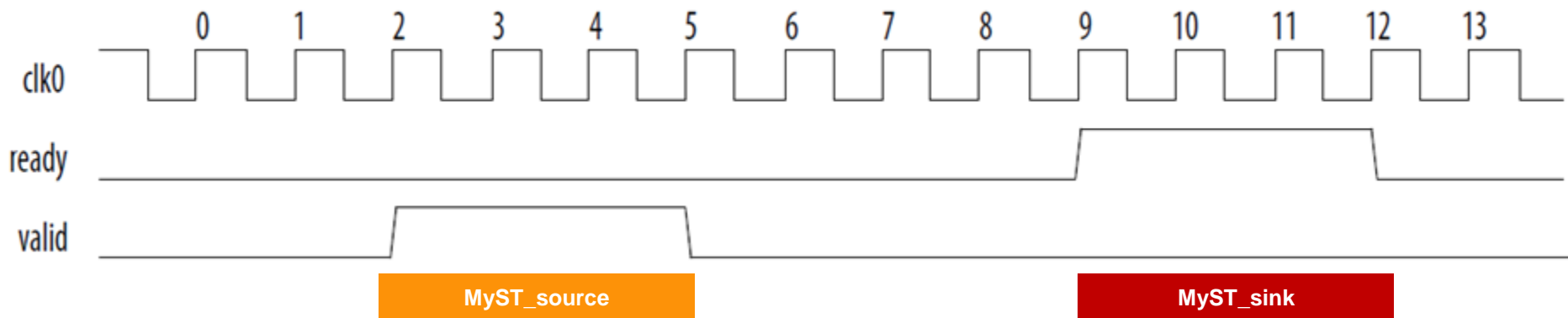


# Структура проекта



```
1 `timescale 1 ps / 1 ps
2 module MyST_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 [3:0] aso_out0_data, // aso_out0.data
8     input bit aso_out0_ready, // .ready
9     output bit aso_out0_valid // .valid
10 );
11 bit [3:0] cnt_int;
12
```

```
13 always_ff @(posedge csi_clk)
14     if(rsi_reset) cnt_int <= 4'd0;
15     else if (aso_out0_ready) cnt_int <= cnt_int + 4'd1;
16
17 assign aso_out0_data = cnt_int;
18
19 always_ff @(posedge csi_clk)
20     if(rsi_reset) aso_out0_valid <= 1'b0;
21     else if ((cnt_int >= 4'd1) & (cnt_int <= 4'd4))
22         aso_out0_valid <= 1'b1;
23     else aso_out0_valid <= 1'b0;
24 endmodule
```



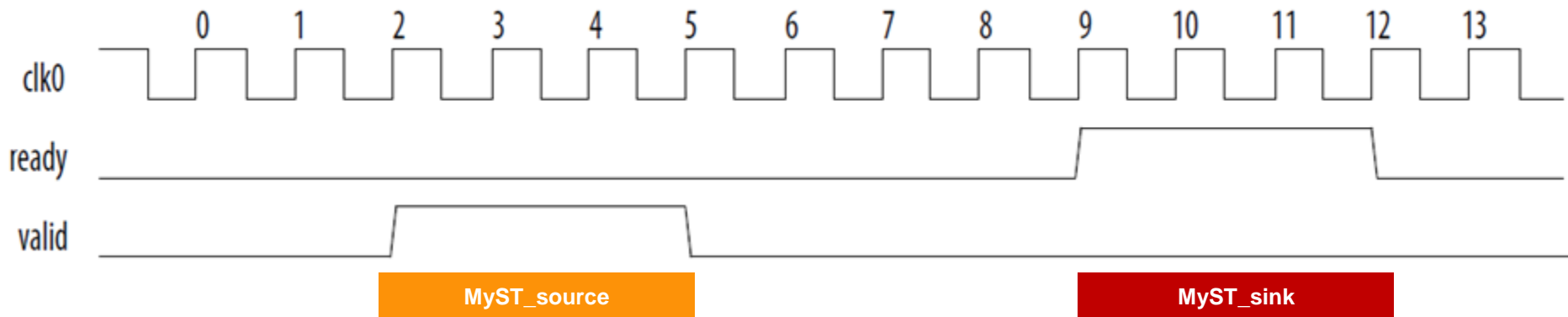


# Структура проекта

```
1 `timescale 1 ns / 1 ns
2 module MyST_sink (
3     //clk and reset
4     input bit csi_clk,
5     input bit rsi_reset,
6     //stream sink
7     input bit [3:0] asi_in0_data,
8     input bit asi_in0_valid,
9     output bit asi_in0_ready,
10    //conduit
11    output bit [3:0] coe_c0_Dout );
12 bit [3:0] cnt_int;
```



```
12 bit [3:0] cnt_int;
13 always_ff @(posedge csi_clk)
14     if(rsi_reset) cnt_int <= 4'd0;
15     else cnt_int <= cnt_int + 4'd1;
16 always_ff @(posedge csi_clk)
17     if(rsi_reset) coe_c0_Dout <= 4'd0;
18     else if (asi_in0_ready & asi_in0_valid)
19         coe_c0_Dout <= asi_in0_data;
20 always_ff @(posedge csi_clk)
21     if(rsi_reset) asi_in0_ready <= 1'b0;
22     else if ((cnt_int >= 4'd8) & (cnt_int <= 4'd11))
23         asi_in0_ready <= 1'b1;
24         else asi_in0_ready <= 1'b0;
25 endmodule
```





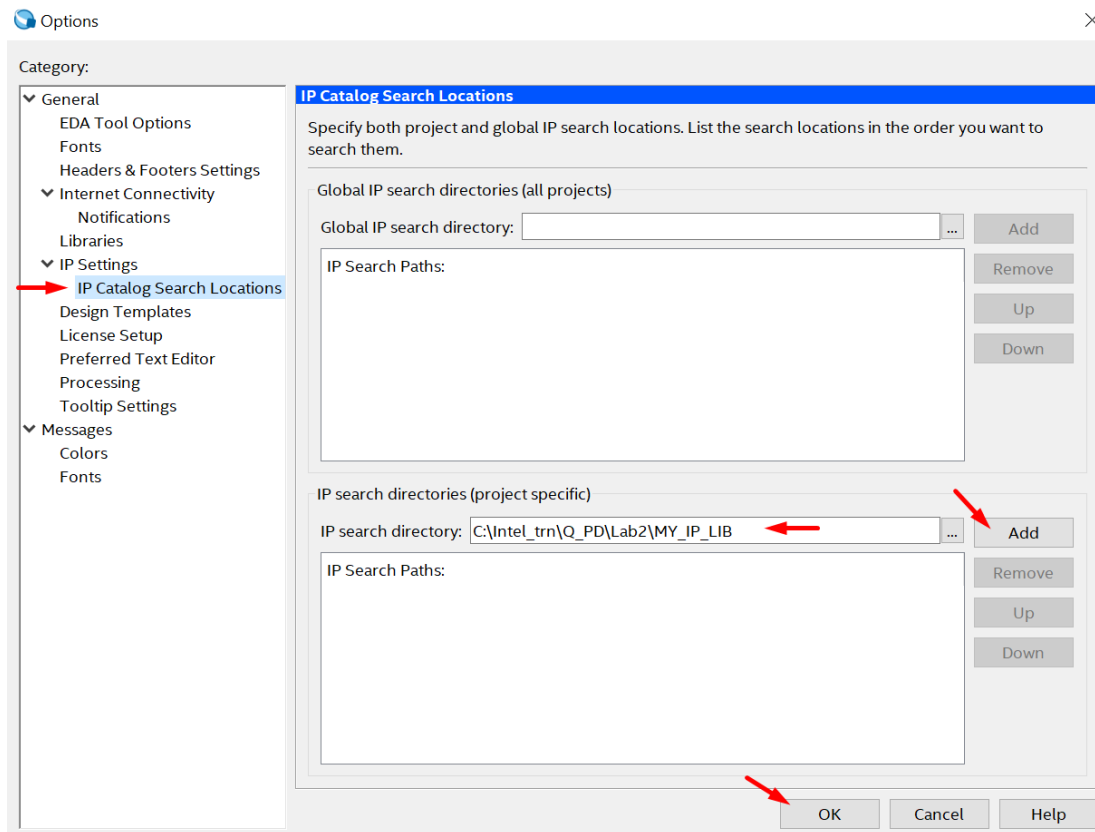
# В QP создайте проект

- **Рабочая папка:** C:\Intel\_trn\Q\_PD\Lab2
- **Имя проекта:** Lab2
- **Модуль верхнего уровня:** Lab2
- **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 QR задайте путь к библиотеке IP

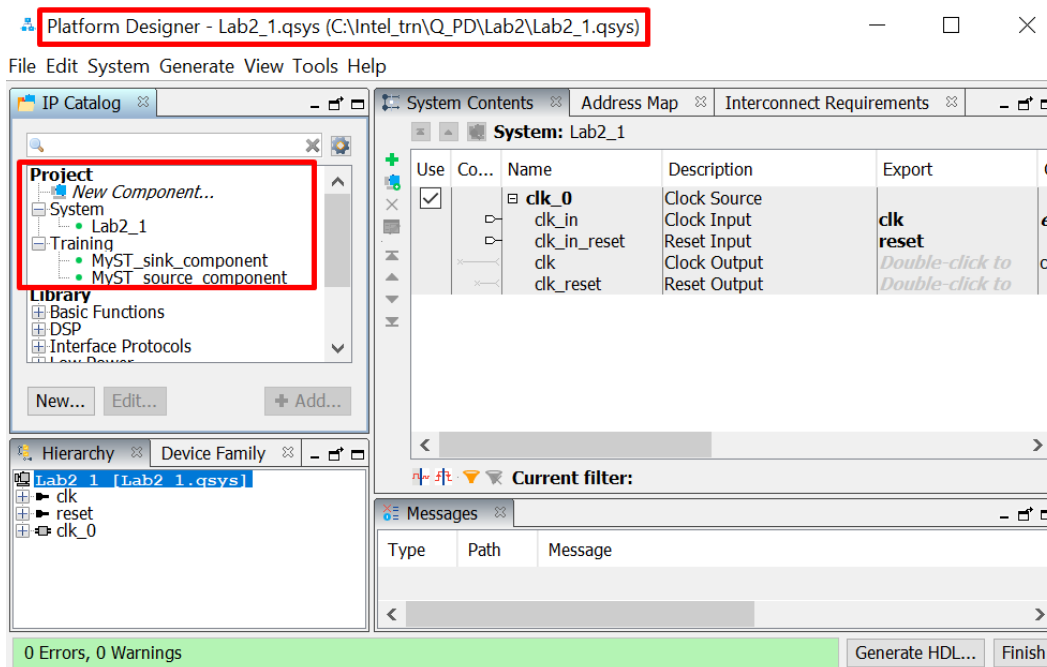
## ■ Команда: Tools=>Options





# В QP запустите приложение PD

- Команда: **Tools => Platform Designer** или иконка 
- В PD: сохраните систему под именем **Lab2\_1.qsys** в рабочей папке проекта
- Убедитесь, 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*




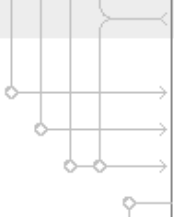
- MyST\_source\_component
- Avalon-ST Single Clock FIFO (в строке поиска наберите ST Single Clock )
- MyST\_sink\_component

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: Lab2_1					
Use	Connectio...	Name	Description	Export	Clock
<input checked="" type="checkbox"/>		<div>clk_0</div> <div>clk_in</div> <div>clk_in_reset</div> <div>clk</div> <div>clk_reset</div>	<div>Clock Source</div> <div>Clock Input</div> <div>Reset Input</div> <div>Clock Output</div> <div>Reset Output</div>	<div>clk</div> <div>reset</div> <div>Double-click to</div> <div>Double-click to</div>	<div><i>exported</i></div> <div>clk_0</div>
<input checked="" type="checkbox"/>		<div>MyST_source...</div> <div>clock</div> <div>reset</div> <div>out0</div>	<div>MyST_source_component</div> <div>Clock Input</div> <div>Reset Input</div> <div>Avalon Streaming Source</div>	<div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div>	<div><i>unconnected</i></div> <div>[clock]</div> <div>[clock]</div>
<input checked="" type="checkbox"/>		<div>sc_fifo_0</div> <div>clk</div> <div>clk_reset</div> <div>in</div> <div>out</div>	<div>Avalon-ST Single Clock F...</div> <div>Clock Input</div> <div>Reset Input</div> <div>Avalon Streaming Sink</div> <div>Avalon Streaming Source</div>	<div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div>	<div><i>unconnected</i></div> <div>[clk]</div> <div>[clk]</div> <div>[clk]</div>
<input checked="" type="checkbox"/>		<div>MyST_sink_0</div> <div>clock</div> <div>reset</div> <div>in0</div> <div>conduit_end_0</div>	<div>MyST_sink_component</div> <div>Clock Input</div> <div>Reset Input</div> <div>Avalon Streaming Sink</div> <div>Conduit</div>	<div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div>	<div><i>unconnected</i></div> <div>[clock]</div> <div>[clock]</div> <div>[clock]</div>



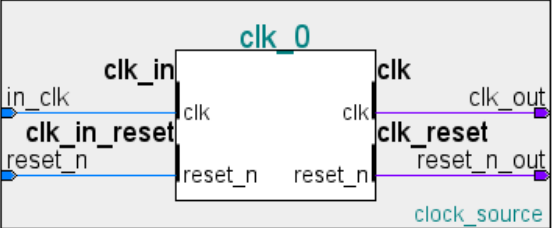
# Настройка компонента **clk\_0**

- Щелчком выберите **clk\_0**
- Нажмите правую клавишу мыши и выберите команду Edit
- В появившемся окне задайте Reset synchronous edges = Deassert

 Clock Source  
clock\_source

**Block Diagram**

☒ Show signals



**Parameters**

Clock frequency: 50000000 Hz

☒ Clock frequency is known

Reset synchronous edges: Deassert ▾





# Настройка компонента `sc_fifo_0`

- Щелчком выберите `sc_fifo_0`
- Нажмите правую клавишу мыши и выберите команду `Edit`
- В появившемся окне задайте
  - Bits per symbol = 4
  - Остальные параметры так, как на рисунке
- Сохраните файл

**Avalon-ST Single Clock FIFO**  
`altera_avalon_sc_fifo`

**Block Diagram**

☒ Show signals

**Parameters**

Symbols per beat:	1
Bits per symbol:	4
FIFO depth:	16
Channel width:	0
Error width:	0
<input type="checkbox"/> Use packets	
<input type="checkbox"/> Use fill level	
<input type="checkbox"/> Use store and forward	
<input type="checkbox"/> Use almost full status	
<input type="checkbox"/> Use almost empty status	
<input type="checkbox"/> Enable explicit maxChannel	
Explicit maxChannel:	0



# 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:** Lab2\_1    **Path:** clk\_0.clk

Connected to: clk\_0.clk

Connected	Connection
<input checked="" type="checkbox"/>	clk_0.clk/MyST_sink_0.clock
<input checked="" type="checkbox"/>	clk_0.clk/MyST_source_0.clock
<input checked="" type="checkbox"/>	clk_0.clk/sc_fifo_0.clk















**System:** Lab2\_1    **Path:** clk\_0.clk




Use	Co...	Name	Description	Export	Clock
<input checked="" type="checkbox"/>		clk_0	Clock Source		
		clk_in	Clock Input	clk	exported [clk_in]
		clk_in_reset	Reset Input	reset	
		clk	Clock Output	Double-click to	clk_0
		clk_reset	Reset Output	Double-click to	clk_0
<input checked="" type="checkbox"/>		MyST_source_0	MyST_source_component		
		clock	Clock Input	Double-click to	clk_0
		reset	Reset Input	Double-click to	[clock]
<input checked="" type="checkbox"/>		sc_fifo_0	Avalon-ST Single Clock F...		
		clk	Clock Input	Double-click to	clk_0
		clk_reset	Reset Input	Double-click to	[clk]
<input checked="" type="checkbox"/>		MyST_sink_0	MyST_sink_component		
		clock	Clock Input	Double-click to	clk_0
		reset	Reset Input	Double-click to	[clock]



# 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
<input checked="" type="checkbox"/>		<b>clk_0</b>	Clock Source		
		clk_in	Clock Input	<b>clk</b>	<i>exported</i>
		clk_in_reset	Reset Input		[clk_in]
		clk	Clock Output	<i>Double-click to</i>	clk_0
		clk_reset	Reset Output	<i>Double-click to</i>	clk_0
<input checked="" type="checkbox"/>		<b>MyST_source_0</b>	MyST_source_component		
		clock	Clock Input	<i>Double-click to</i>	<b>clk_0</b>
		reset	Reset Input	<i>Double-click to</i>	[clock]
<input checked="" type="checkbox"/>		<b>sc_fifo_0</b>	Avalon-ST Single Clock F...		
		clk	Clock Input	<i>Double-click to</i>	<b>clk_0</b>
		clk_reset	Reset Input	<i>Double-click to</i>	[clk]
<input checked="" type="checkbox"/>		<b>MyST_sink_0</b>	MyST_sink_component		
		clock	Clock Input	<i>Double-click to</i>	<b>clk_0</b>
		reset	Reset Input	<i>Double-click to</i>	[clock]

 **Current filter:** Clock and Reset Interfaces



# Подключите Avalon-ST интерфейсы

- На закладке System Contents щелчком выделите интерфейс **MyST\_source\_0.out0**
- Нажмите правую клавишу мыши
- Выберите команду **Filter=> Avalon-ST Interfaces**
- В столбце Connections выполните подключения так, как показано на рисунке
- Сбросьте фильтрацию – нажмите на иконку  в нижней части окна System Contents
- Сохраните файл

Use	Co...	Name	Description	Export	Clock
<input checked="" type="checkbox"/>		MyST_source_0	MyST_source_component		[clock]
		out0	Avalon Streaming Source	<i>Double-click to</i>	clk_0
<input checked="" type="checkbox"/>		sc_fifo_0	Avalon-ST Single Clock F...		[clk]
		in	Avalon Streaming Sink	<i>Double-click to</i>	clk_0
		out	Avalon Streaming Source	<i>Double-click to</i>	[clk]
<input checked="" type="checkbox"/>		MyST_sink_0	MyST_sink_component		[clock]
		in0	Avalon Streaming Sink	<i>Double-click to</i>	clk_0



# Экспортируйте выводы

- На закладке System Contents щелчком выделите интерфейс **MyST\_sink\_0.conduir\_end\_0**
- Дважды щелкните в поле Export и задайте имя **dout**
- Сохраните файл

MyST_sink_0	MyST_sink_component		
clock	Clock Input	Double-click to	clk_0
reset	Reset Input	Double-click to	[clock]
in0	Avalon Streaming Sink	Double-click to	[clock]
conduit_end_0	Conduit	dout	[clock]



# Проверьте систему

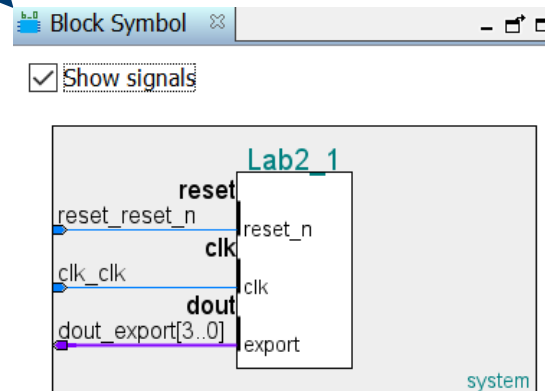
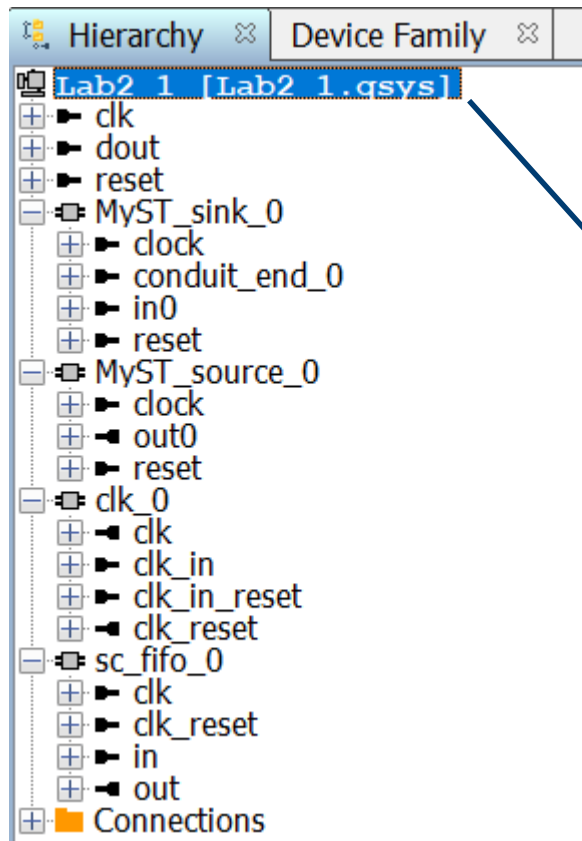
- Убедитесь в том, что
  - Ваша система выглядит так же, как представленная на рисунке
  - Закладка сообщений (Messages) не содержит сообщений.

Use	Connectio...	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	<b>reset</b>	[clk_in]
		clk	Clock Output	<i>Double-click to</i>	clk_0
		clk_reset	Reset Output	<i>Double-click to</i>	clk_0
<input checked="" type="checkbox"/>		<b>MyST_source_0</b>	MyST_source_component		
		clock	Clock Input	<i>Double-click to</i>	<b>clk_0</b>
		reset	Reset Input	<i>Double-click to</i>	[clock]
		out0	Avalon Streaming Source	<i>Double-click to</i>	[clock]
<input checked="" type="checkbox"/>		<b>sc_fifo_0</b>	Avalon-ST Single Clock F...		
		clk	Clock Input	<i>Double-click to</i>	<b>clk_0</b>
		clk_reset	Reset Input	<i>Double-click to</i>	[clk]
		in	Avalon Streaming Sink	<i>Double-click to</i>	[clk]
		out	Avalon Streaming Source	<i>Double-click to</i>	[clk]
<input checked="" type="checkbox"/>		<b>MyST_sink_0</b>	MyST_sink_component		
		clock	Clock Input	<i>Double-click to</i>	<b>clk_0</b>
		reset	Reset Input	<i>Double-click to</i>	[clock]
		in0	Avalon Streaming Sink	<i>Double-click to</i>	[clock]
		conduit_end_0	Conduit	<b>dout</b>	[clock]




# Анализ системы

- Выполните команду: меню View=>Hierarchy
- В окне закладки Hierarchy выделите систему Lab2\_sys [Lab2\_sys.qsys]
- Выполните команду: меню View=>Block Symbol
- Убедитесь, что Ваш символ системы соответствует представленному на рисунке





# Анализ системы

- Выполните команду: меню View => Clock domains Beta
- Выберите режим отображения Clocks 
- Убедитесь в том, что в столбце Connections нет красных точек => нет проблем подключения

**Clocks** Resets

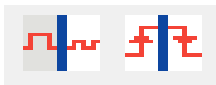
Use	Connectio...	Name	Description	Export	Clock
<input checked="" type="checkbox"/>		<b>clk_0</b>	Clock Source		
		clk_in	Clock Input	<b>clk</b>	<b>exported</b>
		clk_in_reset	Reset Input	<b>reset</b>	[clk_in]
		clk	Clock Output	<i>Double-click to</i>	clk_0
		clk_reset	Reset Output	<i>Double-click to</i>	clk_0
<input checked="" type="checkbox"/>		<b>MyST_source_0</b>	MyST_source_component		
		clock	Clock Input	<i>Double-click to</i>	<b>clk_0</b>
		reset	Reset Input	<i>Double-click to</i>	[clock]
		out0	Avalon Streaming Source	<i>Double-click to</i>	[clock]
<input checked="" type="checkbox"/>		<b>sc_fifo_0</b>	Avalon-ST Single Clock F...		
		clk	Clock Input	<i>Double-click to</i>	<b>clk_0</b>
		clk_reset	Reset Input	<i>Double-click to</i>	[clk]
		in	Avalon Streaming Sink	<i>Double-click to</i>	[clk]
		out	Avalon Streaming Source	<i>Double-click to</i>	[clk]
<input checked="" type="checkbox"/>		<b>MyST_sink_0</b>	MyST_sink_component		
		clock	Clock Input	<i>Double-click to</i>	<b>clk_0</b>
		reset	Reset Input	<i>Double-click to</i>	[clock]
		in0	Avalon Streaming Sink	<i>Double-click to</i>	[clock]
		conduit_end_0	Conduit	<b>dout</b>	[clock]





# Анализ системы

- Выполните команду: меню View => Clock domains Beta
- Выберите режим отображения Reset
- Убедитесь в том, что в столбце Connections нет красных точек => нет проблем подключения



Clocks **Resets**

Use	Connectio...	Name	Description	Export	Clock
<input checked="" type="checkbox"/>		<b>clk_0</b>	Clock Source		
		clk_in	Clock Input		
		clk_in_reset	Reset Input		
		clk	Clock Output		
		clk_reset	Reset Output		
<input checked="" type="checkbox"/>		<b>MyST_source_0</b>	MyST_source_component		
		clock	Clock Input		
		reset	Reset Input		
		out0	Avalon Streaming Source		
<input checked="" type="checkbox"/>		<b>sc_fifo_0</b>	Avalon-ST Single Clock F...		
		clk	Clock Input		
		clk_reset	Reset Input		
		in	Avalon Streaming Sink		
		out	Avalon Streaming Source		
<input checked="" type="checkbox"/>		<b>MyST_sink_0</b>	MyST_sink_component		
		clock	Clock Input		
		reset	Reset Input		
		in0	Avalon Streaming Sink		
		conduit_end_0	Conduit		



# Анализ системы

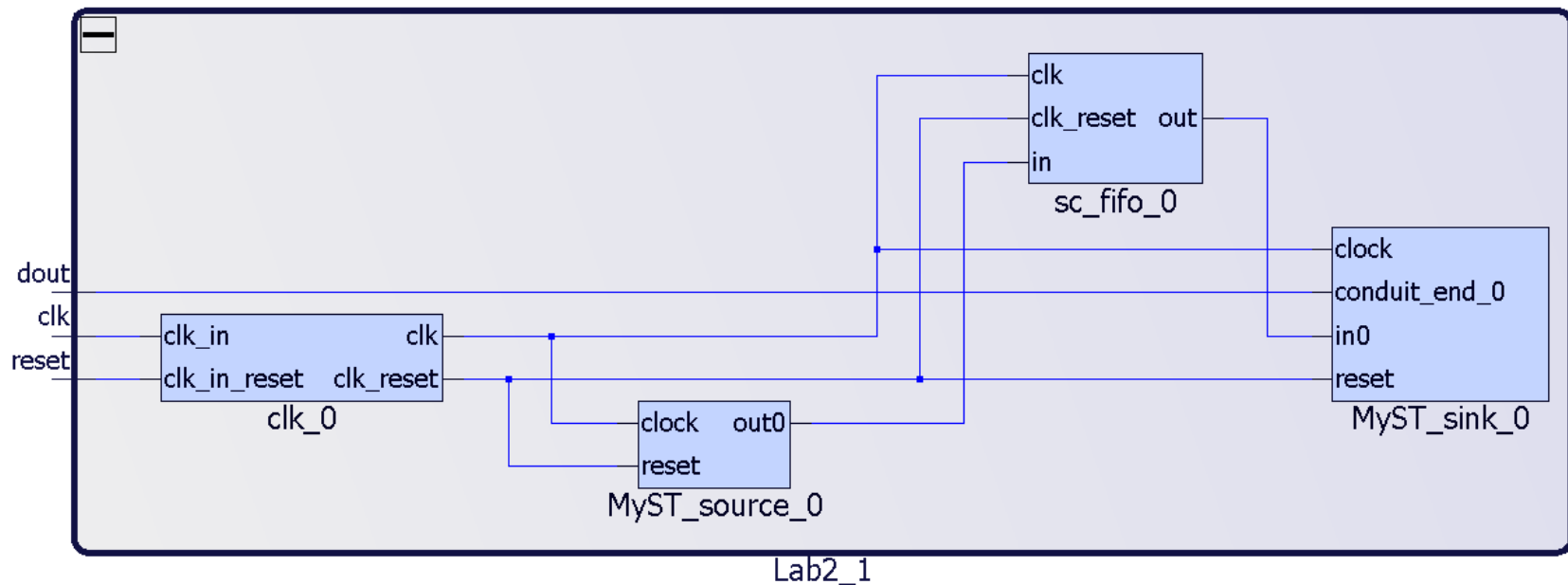
- Выполните команду: меню System => Show System with Platform Designer Interconnect сравните созданную Вами систему и систему с модулями добавленными PD:
  - Убедитесь в том, что PD не добавил никаких модулей.

System: Lab2_1					
Use	Connectio...	Name	Description	Export	Clock
<input checked="" type="checkbox"/>		<div>clk_0</div> <div>clk_in</div> <div>clk_in_reset</div> <div>clk</div> <div>clk_reset</div>	<div>Clock Source</div> <div>Clock Input</div> <div>Reset Input</div> <div>Clock Output</div> <div>Reset Output</div>	<div>clk</div> <div>reset</div> <div>Double-click to</div> <div>Double-click to</div>	<div>exported</div> <div>[clk_in]</div> <div>clk_0</div> <div>clk_0</div>
<input checked="" type="checkbox"/>		<div>MyST_source_0</div> <div>clock</div> <div>reset</div> <div>out0</div>	<div>MyST_source_component</div> <div>Clock Input</div> <div>Reset Input</div> <div>Avalon Streaming Source</div>	<div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div>	<div>clk_0</div> <div>[clock]</div> <div>[clock]</div>
<input checked="" type="checkbox"/>		<div>sc_fifo_0</div> <div>clk</div> <div>clk_reset</div> <div>in</div> <div>out</div>	<div>Avalon-ST Single Clock F...</div> <div>Clock Input</div> <div>Reset Input</div> <div>Avalon Streaming Sink</div> <div>Avalon Streaming Source</div>	<div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div>	<div>clk_0</div> <div>[clk]</div> <div>[clk]</div> <div>[clk]</div>
<input checked="" type="checkbox"/>		<div>MyST_sink_0</div> <div>clock</div> <div>reset</div> <div>in0</div> <div>conduit_end_0</div>	<div>MyST_sink_component</div> <div>Clock Input</div> <div>Reset Input</div> <div>Avalon Streaming Sink</div> <div>Conduit</div>	<div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div>	<div>clk_0</div> <div>[clock]</div> <div>[clock]</div> <div>[clock]</div>



# Анализ системы

- Выполните команду: меню **View=>Schematic**
- Убедитесь в том, что система синхронизации Вашей системы выглядит так же, как представленная на рисунке





# 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
  - 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.

Generate: completed successfully.

**Generation**

**Synthesis**

Synthesis files are used to compile the system in a Quartus project.

Create HDL design files for synthesis: Verilog

☐ Create timing and resource estimates for third-party EDA synthesis tools.

☒ Create block symbol file (.bsf)

**Simulation**

The simulation model contains generated HDL files for the simulator, and may include simulation-only features.

Simulation scripts for this component will be generated in a vendor-specific sub-directory in the specified output directory.

Follow the guidance in the generated simulation scripts about how to structure your design's simulation scripts and how to use the *ip-setup-simulation* and *ip-make-simscript* command-line utilities to compile all of the files needed for simulating all of the IP in your design.

Create simulation model: Verilog

**Output Directory**

Path: C:/Intel\_trn/Q\_PD/Lab2/Lab2\_1

Generate Cancel



You have created an IP Variation in the file  
C:/Intel\_trn/Q\_PD/Lab2/Lab2\_1.qsys.

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/Lab2\_1.qip

The .sip will be located in  
<generation\_directory>/simulation/Lab2\_1.sip



# Подключите файлы к проекту в QP

- В QP
  - Выполните **Project => Add\Remove Files from project**
  - Lab2\_1.qip
  - Lab2\_1.sip
  - Lab2\_top.sv

 Settings - Lab2

Category:

General

**Files**

Libraries

▼ IP Settings

IP Catalog Search Locations

Design Templates

▼ Operating Settings and Conditions

Voltage

Temperature

▼ Compilation Process Settings

Incremental Compilation

## Files

Select the design files you want to include in the project. Click

File name:



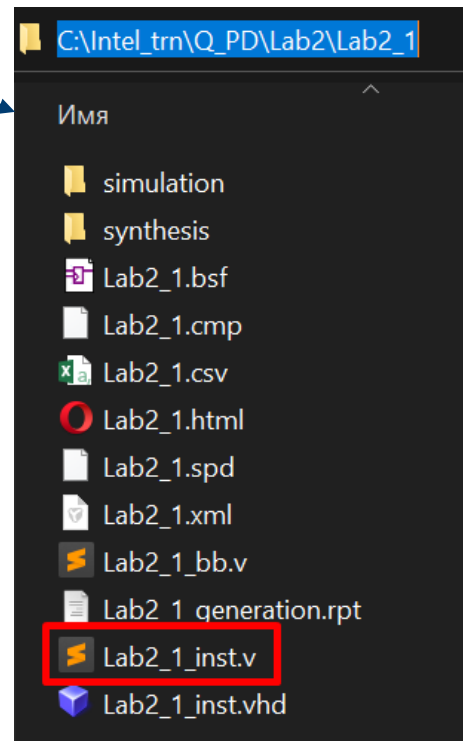
File Name	Type
Lab2_1/simulation/Lab2_1.sip	Quartus Prime SIP File
Lab2_1/synthesis/Lab2_1.qip	IP Variation File (.qip)
Lab2_top.sv	SystemVerilog HDL File



# Файл Lab2\_top.sv

- Создан с использование файла Lab2\_1\_inst.v

```
1  `timescale 1 ns / 1 ns
2  module Lab2_top (
3      input bit clk,
4      input bit reset,
5      output bit [3:0] dout
6  );
7  Lab2_1 Lab2_1_inst (
8      .clk_clk      (clk),
9      .reset_reset_n (reset),
10     .dout_export   (dout)
11 );
12 endmodule
```

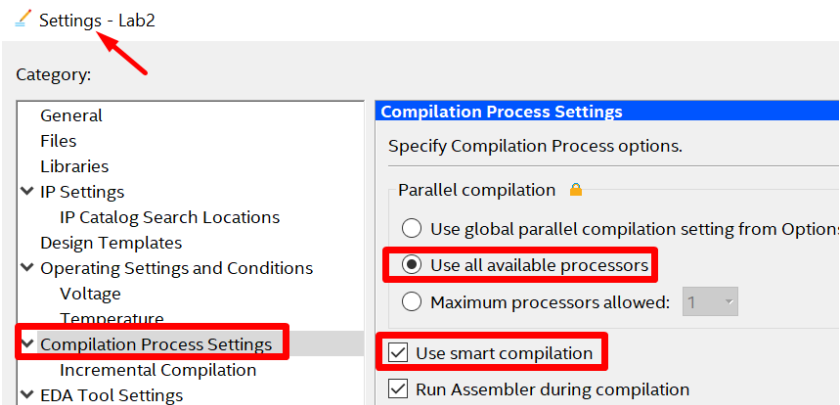
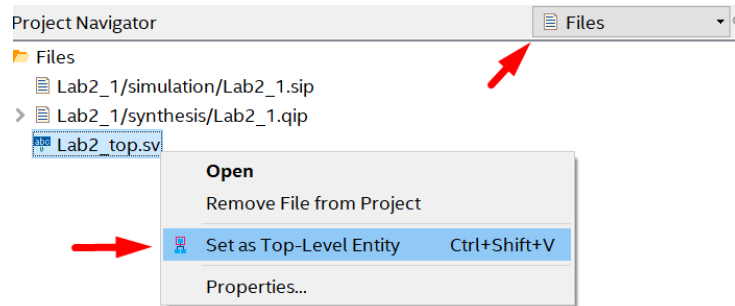
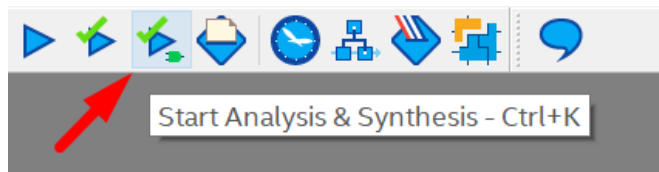




# Анализ и синтез в QP

## ■ В QP

- Файл Lab2\_top.sv объявите файлом верхнего уровня
- Выполните назначения, показанные на рисунке. Команда: **меню Assignment=>Settings**
- Выполните команду **Start Analysis and Synthesis**

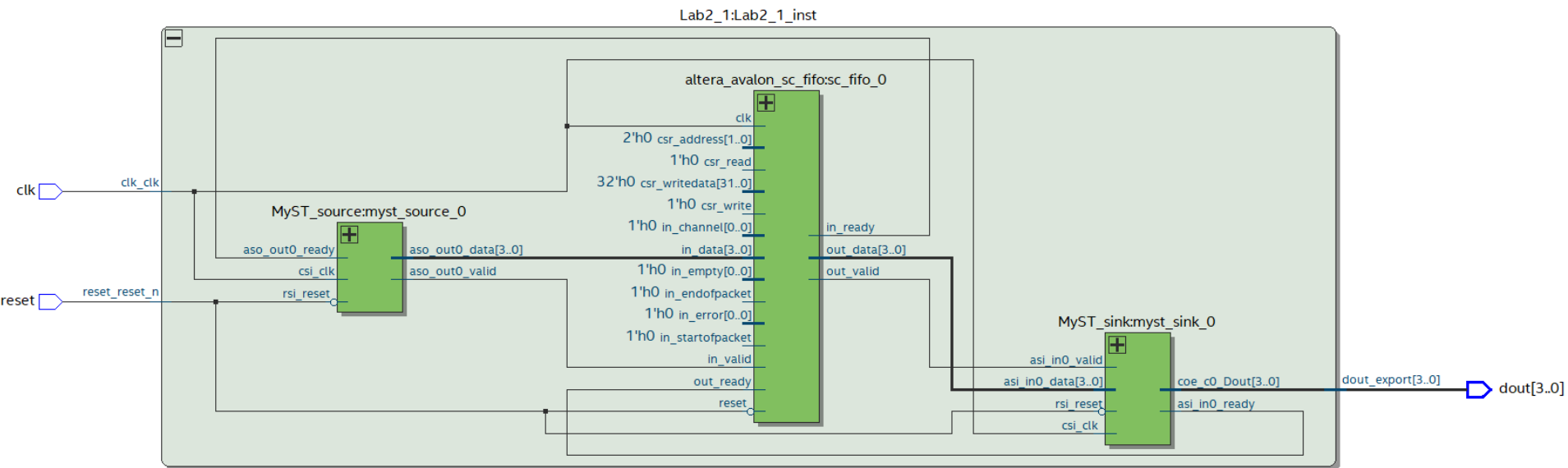


*Убедитесь в том, что  
компиляция завершилась без  
ошибок и предупреждений*



# Анализ RTL Viewer

- Выполните: меню Tools=>Netlist Viewers => RTL viewer
- Убедитесь в том, что Ваша схема похожа на схему, приведенную на рисунке



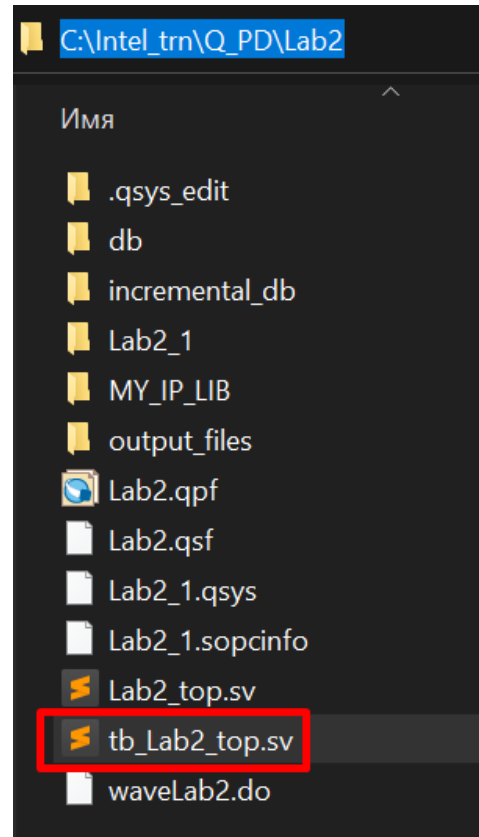




# Файл tb\_Lab2\_top.sv

## ■ Тест для проверки системы

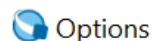
```
1  `timescale 1 ns / 1 ns
2  module tb_Lab2_top ();
3      bit clk;
4      bit reset ;
5      bit[3:0] dout;
6
7      always
8          #50 clk = ~ clk;
9      initial
10     begin
11         clk      = 1'b0;
12         reset    = 1'b0;
13         #500;
14         reset    = 1'b1;
15         #4000;
16         $stop;
17     end
18
19     Lab2_top Lab2_top_inst (.*) ;
20 endmodule
```





# Настройка QP для NativeLink

- Убедитесь, что правильно задана ссылка на пакет ModelSim
  - Выполните команду Tools=>Options



Category:

▼ General

- EDA Tool Options
- Fonts
- Headers & Footers Settings

▼ Internet Connectivity

- Notifications
- Libraries

▼ IP Settings

- IP Catalog Search Locations
- Design Templates
- License Setup
- Preferred Text Editor
- Processing
- Tooltip Settings

▼ Messages

**EDA Tool Options**

Specify the directory that contains the tool executable for each third-party ED.

EDA Tool	Directory Containing Tool Executable
Precision ...	
Synplify	
Synplify ...	
Active-HDL	
Riviera-P...	
ModelSim	
QuestaSim	
ModelSi...	C:\intelFPGA_lite\20.1\modelsim_ase\win32aloem



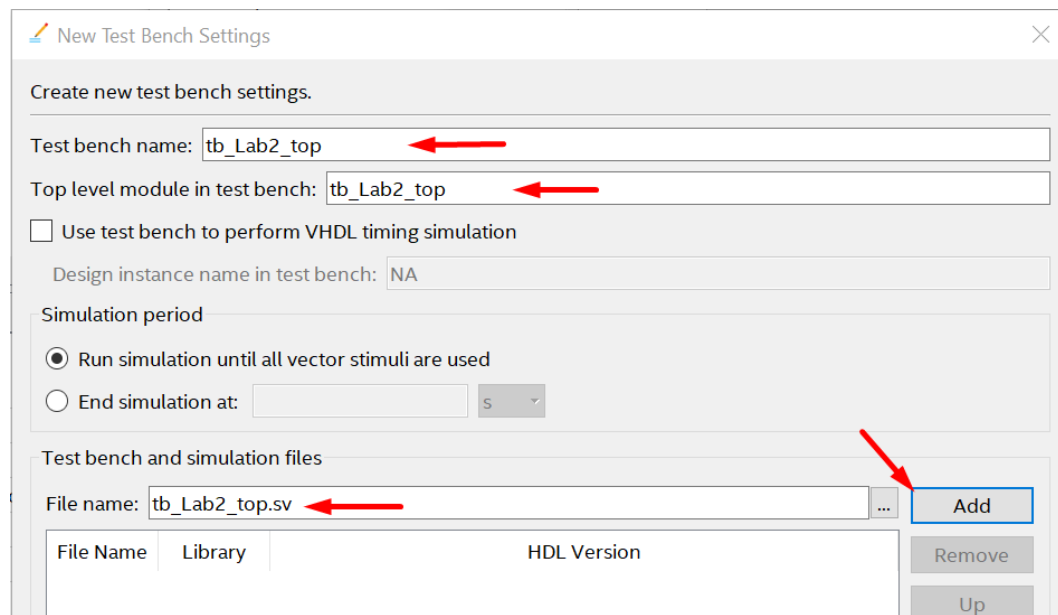
# Настройка QP для NativeLink

- Выполните команду : **меню Assignment=>Settings=>Simulation =>NativeLink settings=>кнопка Test Benches**



- Нажмите кнопку **New**

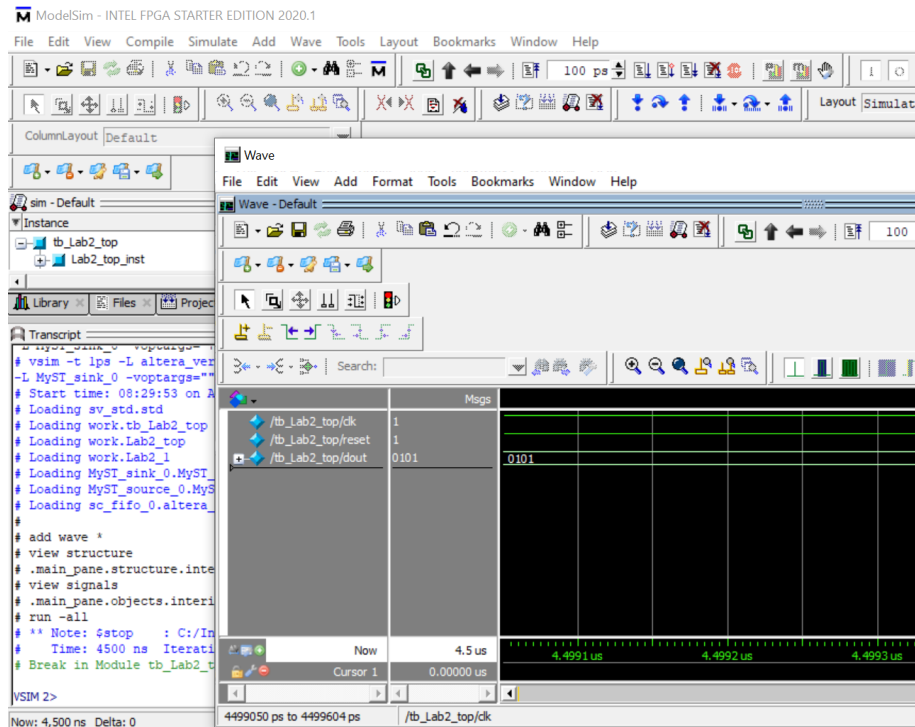
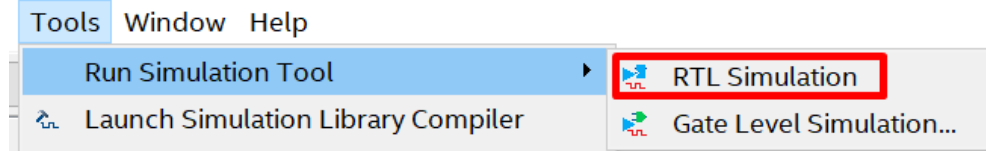
- Выполните назначения, показанные на рисунке.





# Запуск моделирования с NativeLink

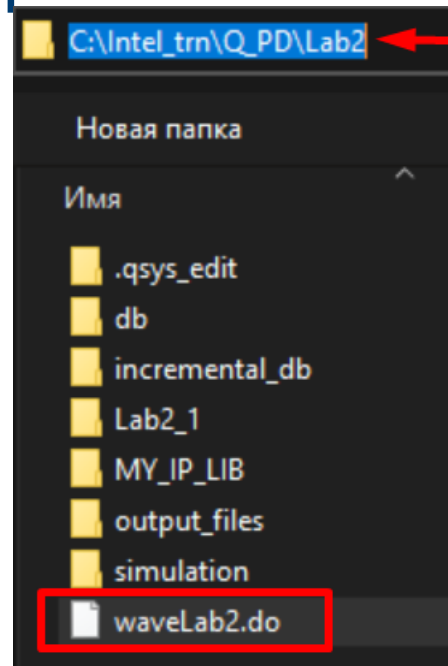
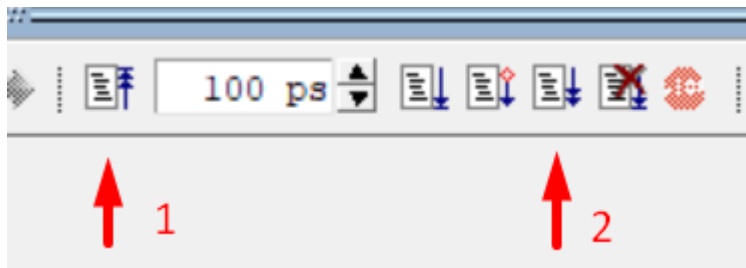
- Выполните команду : меню **Tools=>Run Simulation Tool=>RTL Simulation**
- Откроется окно (окна) пакета ModelSim






# Загрузка формата временной диаграммы

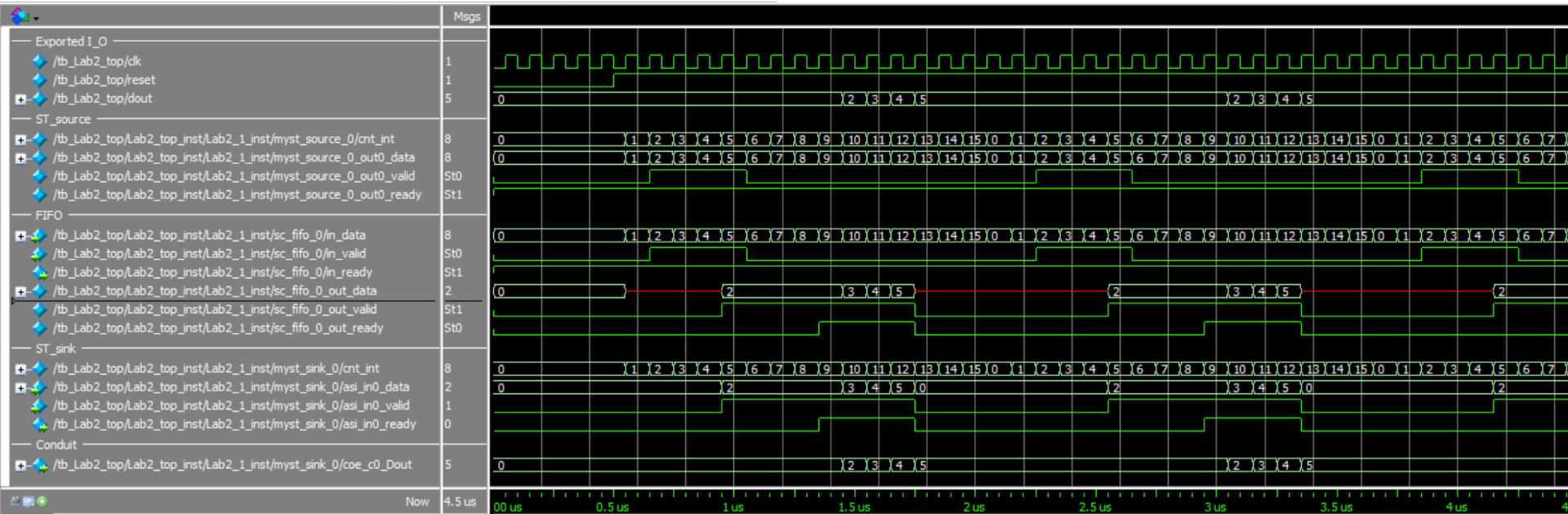
- В окне Wave пакета ModelSim выполните команду: File=>Load и выберите файл waveLab2.do
- В окне Wave пакета ModelSim нажмите кнопку Restart а затем Run -All





# Загрузка формата временной диаграммы

- В окне Wave пакета ModelSim выполните команду Zoom Full 
- Проведите анализ полученной временной диаграммы и убедитесь в правильности работы системы.
- Для чего используется FIFO? Можно ли было обойтись без него?





# *Лабораторная 2*

## *ЗАВЕРШЕНА!*