**Основы статического временного анализа.**

**Часть 3.1: Source Synchronous** **Input Delay Constraint.**

**Введение.**

В представленной статье будет представлен временной анализ передачи данных в FPGA из внешнего устройства. Статья состоит из двух частей. В данной, первой части, будут представлены теоретические основы анализа и рассмотрен простой практический пример.

1. **Цель временных ограничений для входных сигналов.**

Предполагается, что читатель уже знаком с такими понятиями, как ограничения на максимальное и минимальное время распространения, время установки и удержания триггера, *Slack* и т.д. В случае, если это не так, то рекомендуется ознакомится с предыдущими статьями серии [1-3].

В ранее в [2] бы рассмотрен временной анализ передачи данных из внешнего устройства в FPGA в случае, когда тактовый сигнал для формируется тактовый генератором, расположенным на плате. Данный способ передачи называется System Synchronous и показан на рисунке 1.

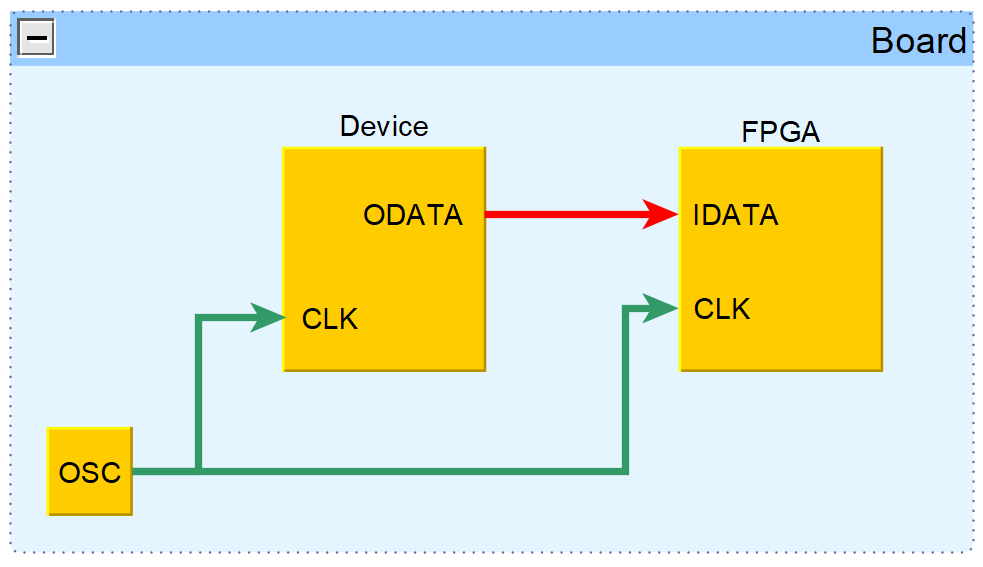


Рисунок 1. Схема соединения устройств на плате для System Synchronous.

Существует другой вариант передачи при, котором источник помимо данных также формирует тактовый сигнал. Данный подход называется Source Synchronous и представлен на рисунке 2. Например, этот случай соответствует приему данных по SPI или I2C, когда FPGA выступает как Slave. Также с этим способом встречаешься при работе с микросхемами физического уровня Ethernet или USB. В дальнейшем для краткости устройство, из которого в FPGA передаются данные и тактовый сигнал, будем иногда называть Device*.*

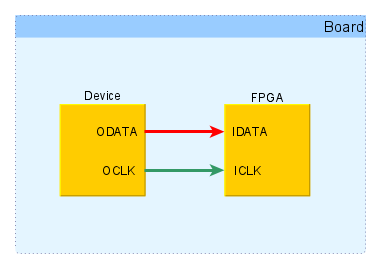


Рисунок 2. Схема соединения устройств на плате для Source Synchronous.

Для исследования временного анализа в Vivado, в FPGA будет загружен простой проект, состоящий из единственного триггера. Схема проекта показана на рисунке 2, а ниже показано Описание проекта на System Verilog:

module top (

    input  logic ICLK,

    input  logic IDATA,

    output logic ODATA

);

    always\_ff@(posedge ICLK)

        ODATA <= IDATA;

endmodule

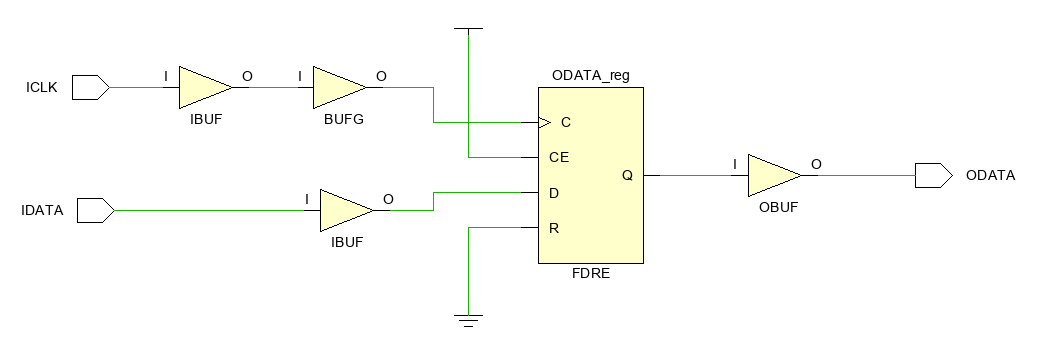


Рисунок 3. Схема FPGA проекта.

Данного простого проекта будет достаточно для того, чтобы продемонстрировать, как проводится временной анализ для входных сигналов.

1. **Задержки при временном анализе для входных сигналов.**

Как и для случая System Synchronous запускающий триггер располагается во внешнем устройстве, а защелкивающий – в FPGA. На рисунке 4 показан анализируемый путь, на который нанесены задержки сигналов. При Source Synchronous источник данных также формирует тактовый сигнал, что отражено на рисунке, тем, что источник тактового сигнала (OSC) внесен внутрь Device.

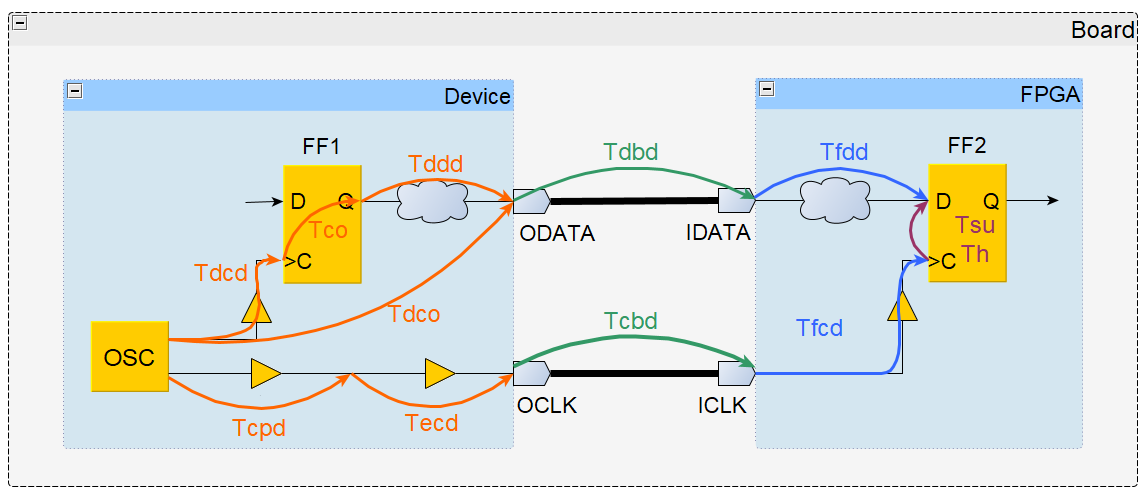


Рисунок 4. Путь с задержками для входных данных и тактового сигнала.

Ниже даны определения задержек, представленных на рисунке 3.

* *Tdbd* (***D****ata* ***B****oard* ***D****elay*) – задержка распространения данных по дорожкам платы от Device до FPGA;
* *Tcbd* (***C****lock* ***B****oard* ***D****elay*) – задержка распространения тактового сигнала по дорожкам платы от Device до FPGA;
* *Tdcd* (***D****evice* ***C****lock* ***D****elay*) – задержка тактового сигнала от генератора OSC до тактового входа запускающего триггера;
* *Tco* (***C****lock to* ***O****utput*) – интервал времени между приходом фронта на тактовый вход триггера и появлением данных на его выходе Q;
* *Tddd* (***D****evice* ***D****ata* ***D****elay*) – задержка распространения данных от запускающего триггера до ножки ODATA Device;
* *Tcpd* (***C****lock to* ***P****in* ***D****elay*) – задержка тактового сигнала от генератора OSC до ножки OCLK Device;
* *Tecd* (***E****dge* ***C****lock* ***D****elay*) – дополнительная задержка тактового сигнала внутри Device, которая зависит от способа его выравнивания относительно данных;
* *Tfcd* (***F****PGA* ***C****lock* ***D****elay*) – задержка тактового сигнала от ножки CLK FPGA до тактового входа защелкивающего триггера;
* *Tfdd* (***F****PGA* ***D****ata* ***D****elay*) – задержка распространения данных от ножки IDATA FPGA до защелкивающего триггера;
* *Tsu* (***S****et****U****p time*) – время установки защелкивающего триггера;
* *Th* (***H****old time*) – время удержания защелкивающего триггера.

Период тактового сигнала будем обозначать *Tclk*. Оранжевым и зеленым цветом на рисунке 4 представлены задержки для участков пути, которые располагаются вне FPGA. Данные задержки необходимо указать временному анализатору Vivado.

Результат временного анализа представляется в виде разницы (*Slack*) между требуемым и фактическим прибытием данных на вход защелкивающего триггера. Отрицательное значение *Slack* указывает на нарушение временных ограничений. Формулы расчета *Slack* для анализа по *Setup* и *Hold* представлены ниже:

где *Tdr* (***D****ata* ***R****equired time*) – требуемое время прибытия данных,   
*Tda* (***D****ata* ***A****rrival time*) – фактическое время прибытия данных.

Фактическое время прибытия данных, вычисляется как сумма задержек распространения тактового сигнала от генератора до запускающего триггера и задержкек распространения данных от запускающего триггера до защелкивающего. Из рисунка 4 получаем следующие результаты:

где использованы обозначения (*Tdco* – ***D****evice* ***C****lock to* ***O****utput time*):

Так как временной анализ проводится для самого пессимистичного случая предыдущих уравнениях в одном случае используются максимальные задержки, а в другом – минимальные. Более подробно этот вопрос рассматривался в предыдущих статьях серии.

Как будет показано далее анализ для случая Source Synchronous имеет особенности расчета требуемого времени прибытия данных. Поэтому прежде, чем двигаться дальше, рассмотрим некоторые обобщения, применимые к анализу входных сигналов.

1. **Обобщение результатов для анализа входных сигналов.**

Для начала рассмотрим анализ по *Setup*. Передача данных начинается между двумя триггерами начинается по запускающему фронту тактового сигнала. Спустя один период *Tclk* следующий фронт защелкивает данные в приемном триггере. Чтобы удовлетворить требование по времени установки, данные на входе защелкивающего триггера должны быть стабильны в течении времени *Tsu* до прихода фронта тактового сигнала. Таким образом, для требуемого времени прибытия данных имеем:

где *Tclk\_delay\_min* – задержка тактового сигнала от генератора до входа защелкивающего триггера. Подставив этот результат в выражение для *Slack* (1), получим

В общем случае задержки тактового сигнала и данных можно разделить на две части, которые обусловленные распространение по участкам пути внутри и вне FPGA. Перегруппировав слагаемые *Tclk\_delay\_min* и *Tda\_setup* из предыдущего уравнения, выражение для *Slack* можно записать в виде:

где *ΣTfpga\_ext* и *ΣTfpga\_int* – алгебраические суммы задержек для участков пути вне и внутри FPGA соответственно. Если путь передачи данных полностью находится внутри FPGA, то значение сумма *ΣTfpga\_ext* будет равно нулю. При анализе выходных сигналов слагаемое *Tsu* будет относится к защелкивающему триггеру вне FPGA и поэтому будет входит в сумму *ΣTfpga\_ext.*

Например, при рассмотрении приема данных для случая System Synchronous [2] получено выражение для *Slack* при анализе по *Setup* в виде:

Отсюда очевидны следующие равенства:

Проведем аналогичные рассуждения для анализа по *Hold*. Требуемое время прибытия данных рассчитывается по формуле:

Так как защелкивающий фронт для предыдущих данных появляется в тот же момент времени, что и запускающий фронт для следующих данных, в представленном выражении отсутствует слагаемое, равное периоду тактового сигнала. Также отметим, что слагаемое *Th* учитывает, что данные не должны изменяться в течении времени удержания после защелкивающего фронта.

Подставив этот результат в выражение для *Slack* (1), получим

где, как и ранее, *Tclk\_delay\_max* – задержка тактового сигнала от генератора до входа защелкивающего триггера.

Сгруппировав задержки, которые входят в состав слагаемых *Tclk\_delay\_max* и *Tda\_hold*, запишем уравнение для *Slack* в виде:

Для случая System Synchronous в [2] получено следующее выражение для анализа по *Hold*:

В данном случае имеем:

При проведении временного анализа Vivado использует уравнения (3) и (4) для расчета *Slack*. Также Vivado ожидает, что ограничени *input\_delay* будет задаваться с помощью суммы *ΣTfpga\_ext*. Закончив подготовительную работу, перейдем непосредственно к рассмотрению временного анализа для входных сигналов для случая Source Synchronous.

1. **Source Synchronous Center Aligned.**

При передаче сигналов в определенный момент времени данные будут выставлены на шину. Спустя некоторое время на шину будут выставлены новые данные, которые заменят текущие данные. Промежуток времени, в течении которого данные выставлены на шину будем называть окном валидных данных (*data valid window*).

Рассмотрим первый вариант передачи данный, который называется Center Aligned. В данном случае тактовый сигнал выравнивается относительно середины окна данных. Это делается с помощь дополнительной задержки *Tecd*, значение которой выбирается равным половине периода тактового сигнала (*Tecd* = *Tclk/2*).

На рисунке 5 представлены временные диаграммы сигналов внутри передающего устройства для двух периодов тактового сигнала. На диаграмме представлены выход тактового генератора (OSC), тактовый вход запускающего триггера (FF1/C), а также выходы устройства OCLK и ODATA. Также на диаграмму для больше наглядности нанесены задержки распространения. Цифрами 1 и 2 обозначены события соответствующими первому и второму фронту соответственно.

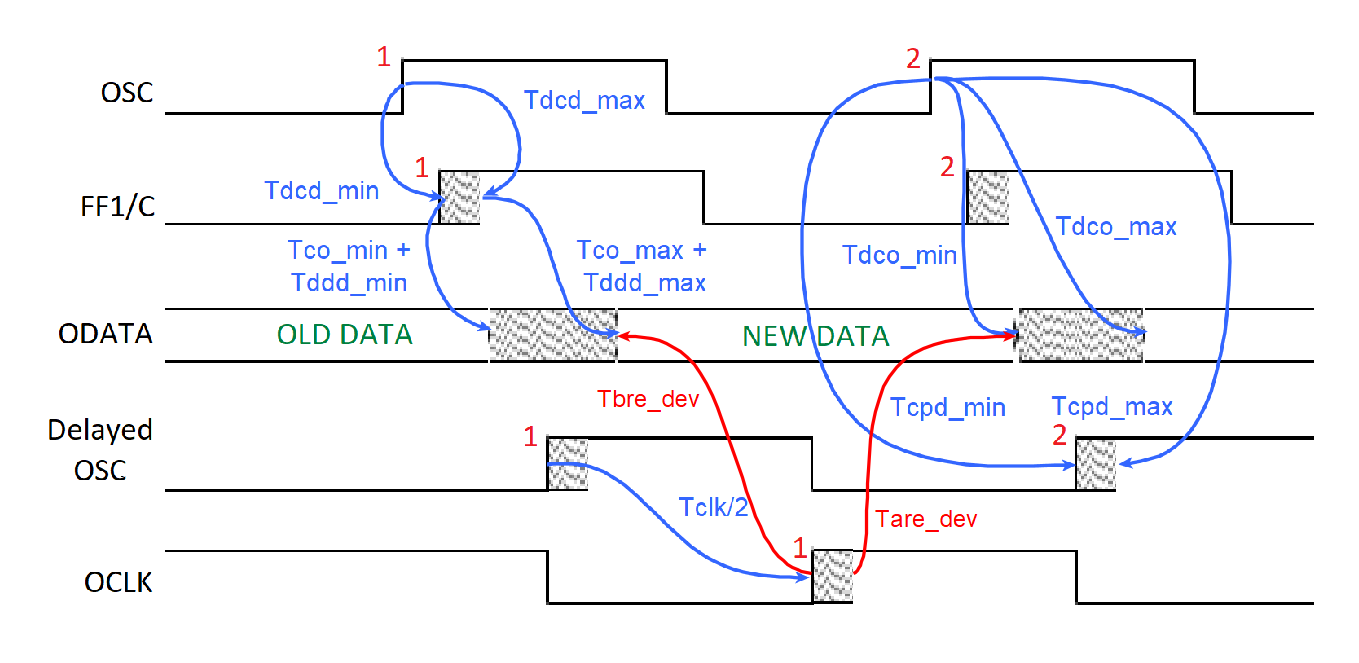


Рисунок 5. Временная диаграмма сигналов внутри Device.

Рассмотрим, как происходит в этом случае происходит передача данных. По фронту 1 запускается передача данных со входа триггера FF1. Тот же самый фронт спустя некоторое время появляется на выход OCLK, передается в FPGA и используется для защелкивания данных. Таким образом, данные запускаются и защелкиваются тем же самым фронтом, в отличие от System Synchronous, где данные защелкивались следующим фронтом.

В случае анализа по *Hold*, рассматривается соотношение между текущий запускающим и предыдущим защёлкивающим фронтом. То есть, для Source Synchronous, если считать, что запускающий фронт появляется в нулевой момент времени, то предыдущий защёлкивающий фронт формируется на период раньше в момент времени *–Tclk.*

Учитывая все вышесказанной, получим уравнения для Slack при выравнивании Center Aligned. Фактическое время прибытья данных рассчитывается с помощью уравнения (2). Требуемое время прибытия данных рассчитывается следующим образом:

* Время прибытия фронта к защелкивающему триггеру внутри FPGA  
  (***D****estination* ***C****lock* ***A****rrival time*):
* Требуемое время прибытия данных (***D****ata* ***R****equired time*):

Подставив полученные результаты в уравнения (1) и учитывая уравнения (2), выражения для *Slack* при анализе по *Setup* и *Hold* можно записать в следующем виде:

Обратите внимание, что в отличие от уравнения (3) в *Slack\_setup* отсутствует слагаемое *Tclk*. В выражении для *Slack\_hold* слагаемое *Tclk* наоборот присутствует, что не соответствует уравнению (4). Также напомним, что в случае Center Aligned выравнивания *Tecd* = *Tclk/2*.

В представленных выше уравнениях оранжевым цветом обозначены задержки, обусловленные распространением сигналов внутри Device. Обычно производители явно не указывают эти задержки в datasheet. Вместо этого чаще всего указываются соотношения между границами окна валидных данных и фронтом тактового сигнала на выходе устройства. Например, на рисунке 5 красным цветом обозначены временные интервалы между левой границей окна и тактовым фронтом (*Tbre\_dev –* ***B****efore* ***R****ising* ***E****dge*) и между правой границей окна и тактовым фронтом (*Tare\_dev –* ***A****fter* ***R****ising* ***E****dge*). Глядя на рисунок 5, получаем следующие результаты:

Также давайте пересчитаем эти соотношения относительно входов FPGA, учитывая задержки распространения по дорожка печатной платы:

Подставив в уравнения (6) для *Tbre\_dev* и *Tare\_dev*, получим следующие равенства:

Также с учетом уравнений (6) можно переписать выражения для *Slack* в виде:

Сопоставляя соотношение для *Slack\_hold* с уравнением (4), получаем

Если сравнить выражение для *Slack\_setup* с уравнением (3), то можно увидеть, что в нем отсутствует слагаемое *Tclk.* Чтобы исправить это несоответствие добавим и вычтем в уравнение для это слагаемое:

Теперь можно записать очевидное равенство:

Полученные соотношения для *input\_delay* можно обнаружить в Vivado Language Templates, если открыть вкладку XDC:

# Center-Aligned Rising Edge Source Synchronous Inputs

#

# For a center-aligned Source Synchronous interface, the clock

# transition is aligned with the center of the data valid window.

# The same clock edge is used for launching and capturing the

# data. The constraints below rely on the default timing

# analysis (setup = 1 cycle, hold = 0 cycle).

#

# input    \_\_\_\_           \_\_\_\_\_\_\_\_\_\_

# clock        |\_\_\_\_\_\_\_\_\_|          |\_\_\_\_\_

#                        |

#                 dv\_bre | dv\_are

#                <------>|<------>

#          \_\_    \_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_    \_\_

# data     \_\_XXXX\_\_\_\_Rise\_Data\_\_\_\_XXXX\_\_

set input\_clock         <clock\_name>;      # Name of input clock

set input\_clock\_period  <period\_value>;    # Period of input clock

set dv\_bre              0.000;             # Data valid before the rising clock edge

set dv\_are              0.000;             # Data valid after the rising clock edge

set input\_ports         <input\_ports>;     # List of input ports

# Input Delay Constraint

set\_input\_delay -clock $input\_clock -max [expr $input\_clock\_period - $dv\_bre]

[get\_ports $input\_ports];

set\_input\_delay -clock $input\_clock -min $dv\_are

[get\_ports $input\_ports];

В данном случае значение *input\_delay\_max* задается в виде разности периода тактового сигнала input\_clock\_period (*Tclk*) и временного интервала dv\_bre (*Tbre\_fpga*). Значение *input\_delay\_min* равно dv\_are (*Tare\_fpga*). Это согласуется с полученными уравнениями (7) и (8). Также комментариях можно увидеть, что при Center-Aligned выравнивании, данные действительно запускаются и защелкиваются одним и тем же фронтом тактового сигнала.

1. **Source Synchronous Edge Aligned.**

Теперь рассмотрим второй вариант выравнивания тактового сигнала относительно данных, который называется Edge Aligned. В данном случае тактовый сигналы выравнивается относительно левой границы окна валидных данных, что соответствует дополнительной задержке *Tecd* = 0*.* Временные диаграммы сигналов внутри передающего устройства представлены на рисунке 6.

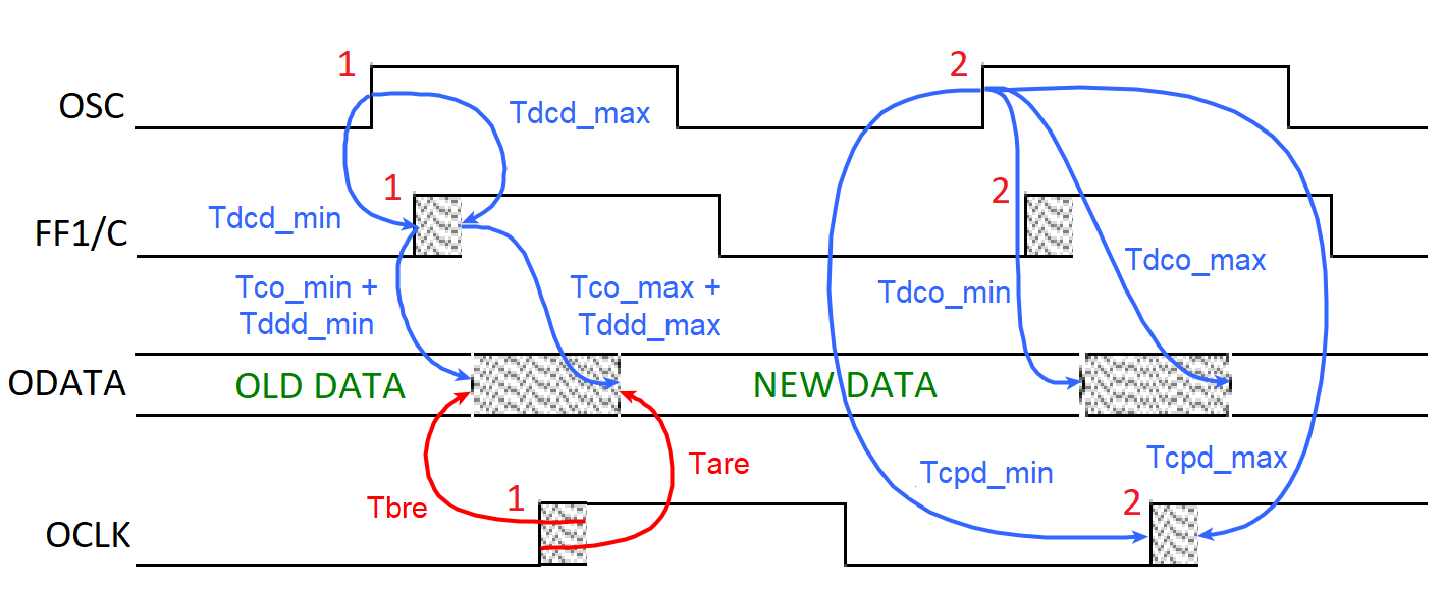


Рисунок 6. Временная диаграмма сигналов внутри Device.

В этом случае, как и для Center Aligned, данные запускаются и защелкиваются по одному и тому же фронту тактового сигнала. Глядя на рисунок 6, а также вспомнив уравнения (2), можно получить выражения для *Slack* при анализе по *Setup* и *Hold*:

Введем также два временных интервала *Tbre\_dev* и *Tare\_dev*, которые обозначены на рисунке 6 красным цветом. В данном случае значение *Tbre\_dev* равно разности между временем появления тактового фронта и моментом, когда старые данные перестанут быть валидными. В свою очередь *Tare\_dev* соответствует промежутку времени между установкой на выходной шине устройства новых данных и появлением тактового фронта.

Из рисунка 6, легко получить следующие соотношения для самого пессимистичного случая:

Также пересчитаем эти соотношения относительно входов FPGA, учитывая задержки распространения по дорожка печатной платы:

Теперь подставим полученные выражения для *Tbre\_fpga* и *Tare\_fpga* в уравнения для *Slack* и получим*:*

где, как и для случая Center Aligned, в уравнение для *Slack\_setup* добавлено слагаемое *Tclk* с положительным и отрицательным знаком.

Сопоставив полученные результаты с уравнениями (3) и (4), получим окончательный результат в виде:

Данный результат совпадает с Vivado Language Templates, который представлен ниже и описывает входные ограничения для случая Source Synchronous Edge Aligned:

# Edge-Aligned Rising Edge Source Synchronous Inputs

# (Using a direct FF connection)

#

# For an edge-aligned Source Synchronous interface, the clock

# transition occurs at the same time as the data transitions.

# In this template, the clock is aligned with the beginning of the

# data. The constraints below rely on the default timing

# analysis (setup = 1 cycle, hold = 0 cycle).

#

# input    \_\_\_\_\_\_\_\_\_\_                  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# clock              |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|                |\_\_\_\_\_\_\_\_\_\_

#                                     |

#                             skew\_bre|skew\_are

#                             <------>|<------>

#             \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_        |        \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# data     XXX\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_XXXXXXXXXXXXXXXXX\_\_\_\_Rise\_Data\_\_\_XXX

#

set input\_clock         <clock\_name>;     # Name of input clock

set input\_clock\_period  <period\_value>;   # Period of input clock

set skew\_bre            0.000;            # Data invalid before the rising clock edge

set skew\_are            0.000;            # Data invalid after the rising clock edge

set input\_ports         <input\_ports>;    # List of input ports

# Input Delay Constraint

set\_input\_delay -clock $input\_clock -max [expr $input\_clock\_period + $skew\_are]

[get\_ports $input\_ports];

set\_input\_delay -clock $input\_clock -min [expr $input\_clock\_period - $skew\_bre]

[get\_ports $input\_ports];

1. **Source Synchronous Edge Aligned (MMCM).**

В заключении рассмотрим вариант выравнивания тактового сигнала относительно данных, который в Vivado назван Edge Aligned MMCM. В этом варианте тактовый сигнал выравнивается относительно правой границы окна валидных данных, что соответствует дополнительной задержке *Tecd* = *Tclk.* Задержка на один период соответствует следующему фронту тактового сигнал, таким образом в этом случае данные запускаются одним фронтом, а защелкиваются следующим. Временные диаграммы сигналов внутри передающего устройства совпадают с представленным на рисунке 6, где запускающему фронту соответствует фронт 1, а защелкивающему – фронт 2.

Считая, что запускающий фронт появляется в нулевой момент времени, и глядя на рисунок 6 можно записать уравнения для *Slack*:

Определим временные интервалы *Tbre\_dev* и *Tare\_dev*, а также их пересчитанные значения относительно входов FPGA *Tbre\_fpga* и *Tare\_fpga* тем же способом, что для случая Edge Aligned. Тогда есть для них также будут выполнены соотношения из уравнений (9) и (10).

Учитывая выражения для *Tbre\_fpga* и *Tare\_fpga*,уравнения для *Slack* можно записать в виде:

Сопоставляя данный результат с уравнениями (3) и (4), получим окончательный результат в виде:

Vivado Language Templates, соответствующий случаю Source Synchronous Edge Aligned (MMCM), представлен ниже:

# Edge-Aligned Rising Edge Source Synchronous Inputs

# (Using an MMCM/PLL)

#

# For an edge-aligned Source Synchronous interface, the clock

# transition occurs at the same time as the data transitions.

# In this template, the clock is aligned with the end of the

# data. The constraints below rely on the default timing

# analysis (setup = 1 cycle, hold = 0 cycle).

#

# input    \_\_\_\_\_\_\_\_\_\_                  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# clock              |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|                |\_\_\_\_\_\_\_\_\_\_

#                                     |

#                             skew\_bre|skew\_are

#                             <------>|<------>

#            \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_        |        \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# data     XX\_\_\_\_Rise\_Data\_\_\_\_XXXXXXXXXXXXXXXXX\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_XX

set input\_clock         <clock\_name>;     # Name of input clock

set skew\_bre            0.000;            # Data invalid before the rising clock edge

set skew\_are            0.000;            # Data invalid after the rising clock edge

set input\_ports         <input\_ports>;    # List of input ports

# Input Delay Constraint

set\_input\_delay -clock $input\_clock -max $skew\_are  [get\_ports $input\_ports];

set\_input\_delay -clock $input\_clock -min -$skew\_bre [get\_ports $input\_ports];

Приведем некоторые соображения относительно, того почему в названии данного варианта выравнивания тактового сигнала присутствует слово MMCM.

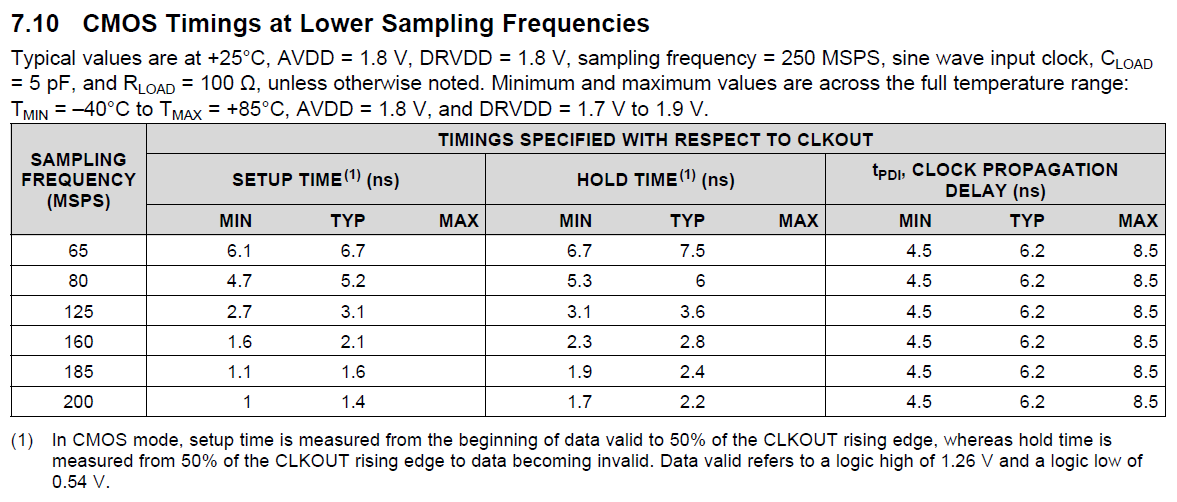
Пусть передающее устройство формирует тактовый сигнал, который выровнен по левому краю окна валидных данные (случай Edge Aligned), то есть данные запускаются и защелкиваются одним и тем же фронтом. Также пусть перед тем, как поступить на защелкивающий триггер тактовый сигнал проходит через MMCM и PLL. При такой конфигурации весьма вероятно возникнут нарушения при анализе по Setup.

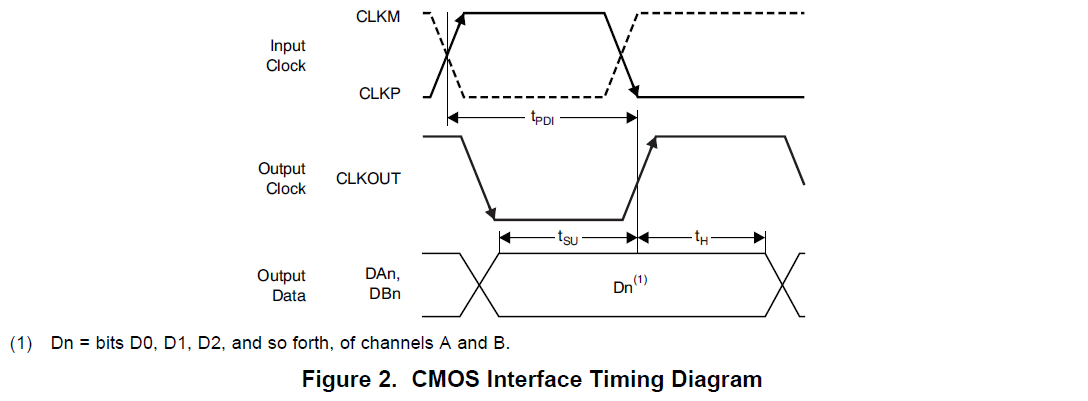
Причина их в следующем. Если MMCM (или PLL) тактирует триггер, на которые поступает сигнал с входного порта FPGA, то Vivado для данного MMCM автоматически устанавливает компенсацию в режим ZHOLD [4]. При этом MMCM формирует для тактового сигнала “отрицательную” задержку, чтобы гарантировать отсутствие проблем для анализа по *Hold*. Это мотивируется тем, что их исправление возможно потребует увеличения длины дорожек для данных на печатной плате, что трудоемко и нежелательно. Можно сказать, что с помощью отрицательной задержки MMCM “ускоряет” распространение тактового сигнала.

Однако ранее прибытие тактового сигнала приводит к проблеме при анализе по *Setup*, так как данные не успевают дойти до защелкивающего триггера. Возможное решение, это защелкивать данные не текущим, а следующим фронтом, что и осуществляется в случае Edge Aligned (MMCM).

В качестве альтернативы можно вручную установить MMCM режим компенсации INTERNAL, однако Xilinx не рекомендует данный способ.

**Пример задания ограничений**





# период тактового сигнала

set Tclk 5

# время удержания данных после тактового сигнала на выходе ADS4249

set Tare\_dev 1.7

# время между появлением данных и тактовым сигналом на выходе ADS4249

set Tbre\_dev 1

# минимальное и максимальное время распространения данных по дорожкам платы

set Tdbd\_max 0.15

set Tdbd\_min 0.1

# минимальное и максимальное время распространения тактового сигнала по дорожкам платы

set Tcbd\_max 0.12

set Tcbd\_min 0.07

# время удержания данных после тактового сигнала на входе FPGA

set Tare\_fpga [expr $Tare\_dev + $Tdbd\_min - $Tcbd\_max]

# время между появлением данных и тактовым сигналом на входе FPGA

set Tbre\_fpga [expr $Tbre\_dev + $Tcbd\_min - $Tdbd\_max]

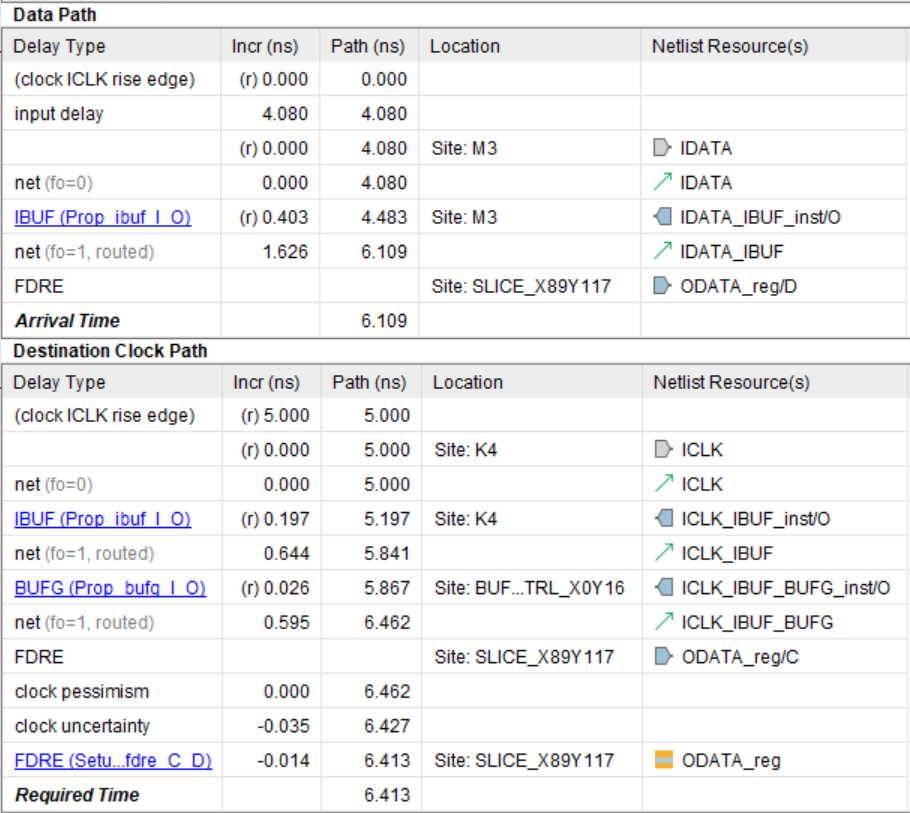
# ограничение на период тактового сигнала

create\_clock -period $Tclk -name ICLK [get\_ports ICLK]

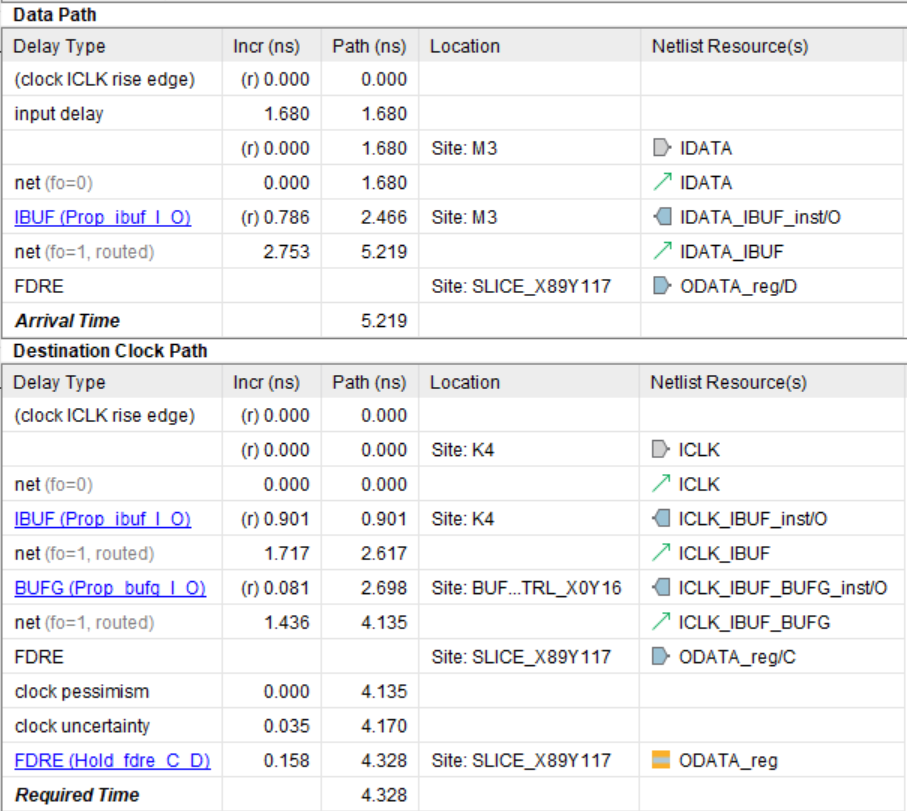
# временные ограничение для входного сигнала IDATA

set\_input\_delay -clock ICLK -max [expr $Tclk - $Tbre\_fpga] [get\_ports IDATA];

set\_input\_delay -clock ICLK -min $Tare\_fpga [get\_ports IDATA];

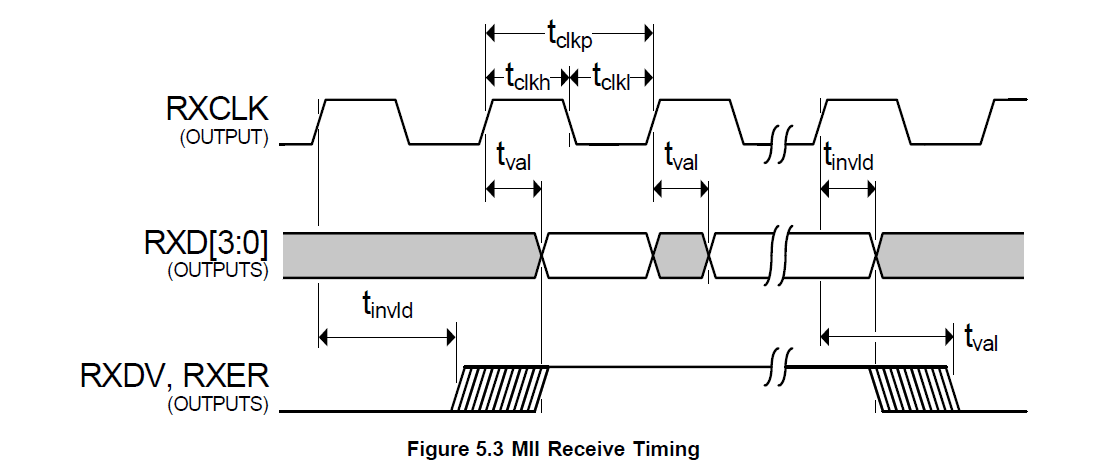


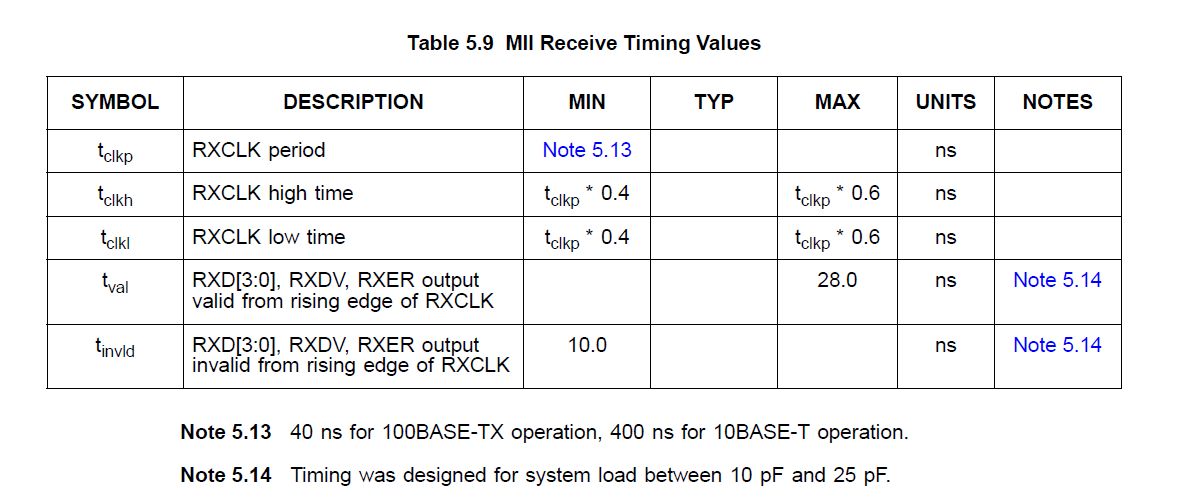
Setup



Hold

**Заключение.**





**Пример задания ограничений. Edge Aligned.**

# период тактового сигнала

set Tclk 40

# время удержания данных после тактового сигнала на выходе LAN8740A (tinvld)

set Tbre\_dev -10

# время между появлением тактового сигнала и данных на выходе LAN8740A (tval)

set Tare\_dev 28

# минимальное и максимальное время распространения данных по дорожкам платы

set Tdbd\_max 0.15

set Tdbd\_min 0.1

# минимальное и максимальное время распространения тактового сигнала по дорожкам платы

set Tcbd\_max 0.12

set Tcbd\_min 0.07

# время удержания данных после тактового сигнала на входе FPGA

set Tbre\_fpga [expr $Tbre\_dev + $Tcbd\_max - $Tdbd\_min]

# время между появлением тактового сигнала и данных на входе FPGA

set Tare\_fpga [expr $Tare\_dev + $Tdbd\_max - $Tcbd\_min]

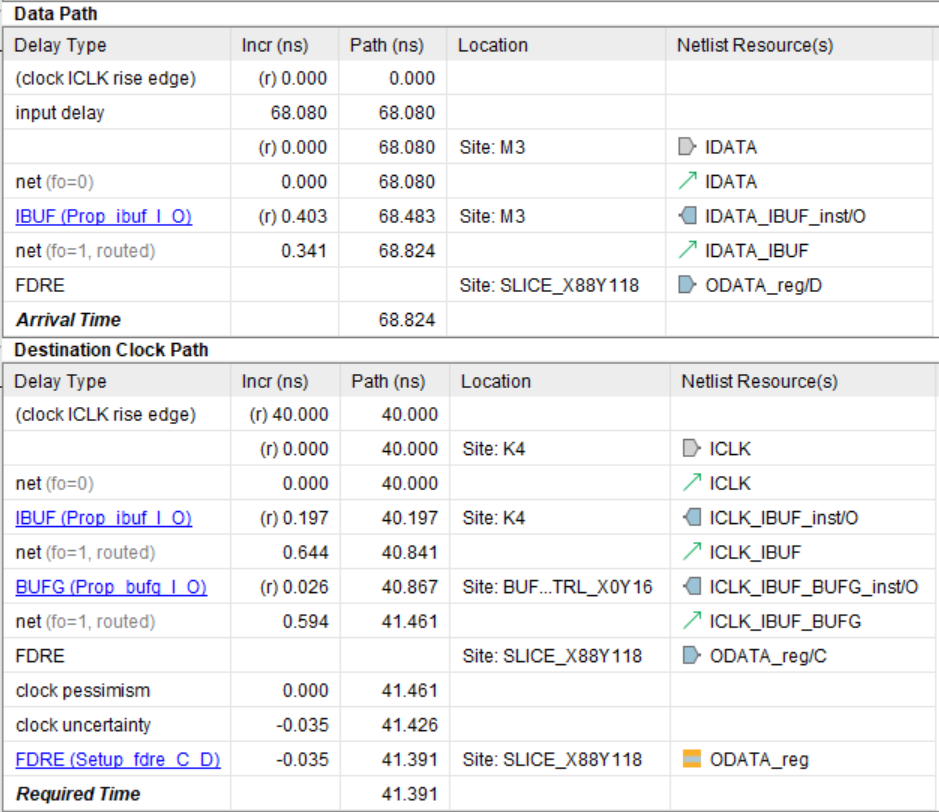
# ограничение на период тактового сигнала

create\_clock -period $Tclk -name ICLK [get\_ports ICLK]

# временные ограничение для входного сигнала IDATA

set\_input\_delay -clock ICLK -max [expr $Tclk + $Tare\_fpga] [get\_ports IDATA];

set\_input\_delay -clock ICLK -min [expr $Tclk - $Tbre\_fpga] [get\_ports IDATA];



**Пример задания ограничений. Edge Aligned MMCM.**

# период тактового сигнала

set Tclk 40

# время удержания данных после тактового сигнала на выходе LAN8740A (tinvld)

set Tbre\_dev -10

# время между появлением тактового сигнала и данных на выходе LAN8740A (tval)

set Tare\_dev 28

# минимальное и максимальное время распространения данных по дорожкам платы

set Tdbd\_max 0.15

set Tdbd\_min 0.1

# минимальное и максимальное время распространения тактового сигнала по дорожкам платы

set Tcbd\_max 0.12

set Tcbd\_min 0.07

# время удержания данных после тактового сигнала на входе FPGA

set Tbre\_fpga [expr $Tbre\_dev + $Tcbd\_max - $Tdbd\_min]

# время между появлением тактового сигнала и данных на входе FPGA

set Tare\_fpga [expr $Tare\_dev + $Tdbd\_max - $Tcbd\_min]

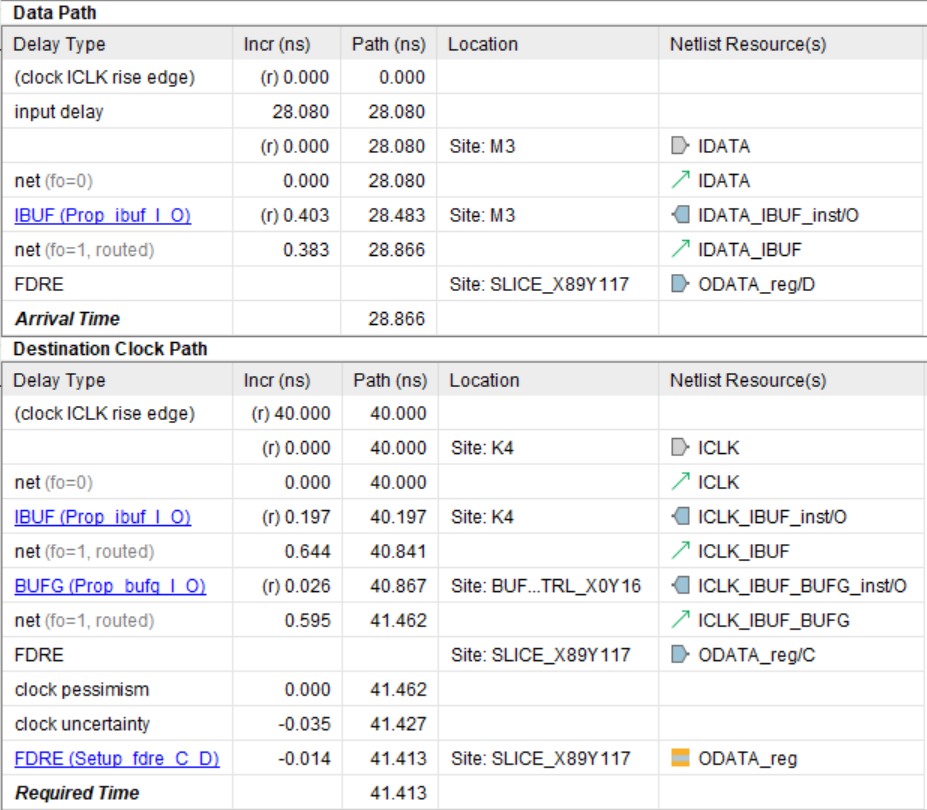
# ограничение на период тактового сигнала

create\_clock -period $Tclk -name ICLK [get\_ports ICLK]

# временные ограничение для входного сигнала IDATA

set\_input\_delay -clock ICLK -max $Tare\_fpga [get\_ports IDATA];

set\_input\_delay -clock ICLK -min -$Tbre\_fpga [get\_ports IDATA];



**Пример задания ограничений. Center Aligned.**

# период тактового сигнала

set Tclk 40

# время удержания данных после тактового сигнала на выходе LAN8740A (tinvld)

set Tare\_dev 10

# время между появлением данных и тактовым сигналом на выходе LAN8740A (tval)

set Tbre\_dev [expr $Tclk - 28]

# минимальное и максимальное время распространения данных по дорожкам платы

set Tdbd\_max 0.15

set Tdbd\_min 0.1

# минимальное и максимальное время распространения тактового сигнала по дорожкам платы

set Tcbd\_max 0.12

set Tcbd\_min 0.07

# время удержания данных после тактового сигнала на входе FPGA

set Tbre\_fpga [expr $Tbre\_dev + $Tcbd\_min - $Tdbd\_max]

# время между появлением тактового сигнала и данных на входе FPGA

set Tare\_fpga [expr $Tare\_dev + $Tdbd\_min - $Tcbd\_max]

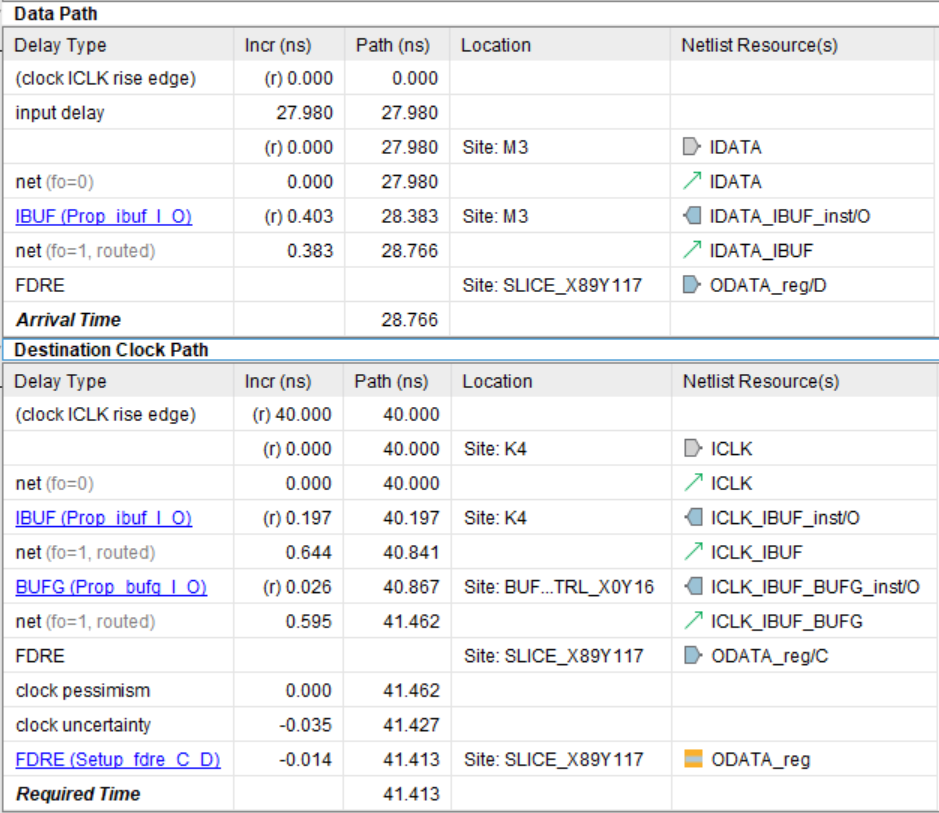
# ограничение на период тактового сигнала

create\_clock -period $Tclk -name ICLK [get\_ports ICLK]

# временные ограничение для входного сигнала IDATA

set\_input\_delay -clock ICLK -max [expr $Tclk - $Tbre\_fpga] [get\_ports IDATA];

set\_input\_delay -clock ICLK -min $Tare\_fpga [get\_ports IDATA];



**Ссылки.**