**Основы статического временного анализа.**

**Часть 3: Source Synchronous** **Input Delay Constraint.**

**Введение.**

В статье представлен временной анализ передачи данных в FPGA из внешнего устройства. Рассмотрены теоретические основы анализа для трех различных вариантов выравнивания данных относительно тактового сигнала. Также разобраны два примера создания временных ограничений.

1. **Передача данных для случая Source Synchronous.**

Данная статья частично опирается на материал, рассмотренный в предыдущих работах серии [1-3]. Предполагается, что читатель уже знаком с такими понятиями, как ограничение на максимальное (*Setup*) и минимальное (*Hold*) время распространения, запас (*Slack*) и т.д.

В ранее в [2] бы представлен временной анализ передачи данных из внешнего устройства в FPGA в случае, когда тактовый сигнал формируется генератором, расположенным на плате на той же плате (см. рисунке 1). Такой способ передачи называется System Synchronous.

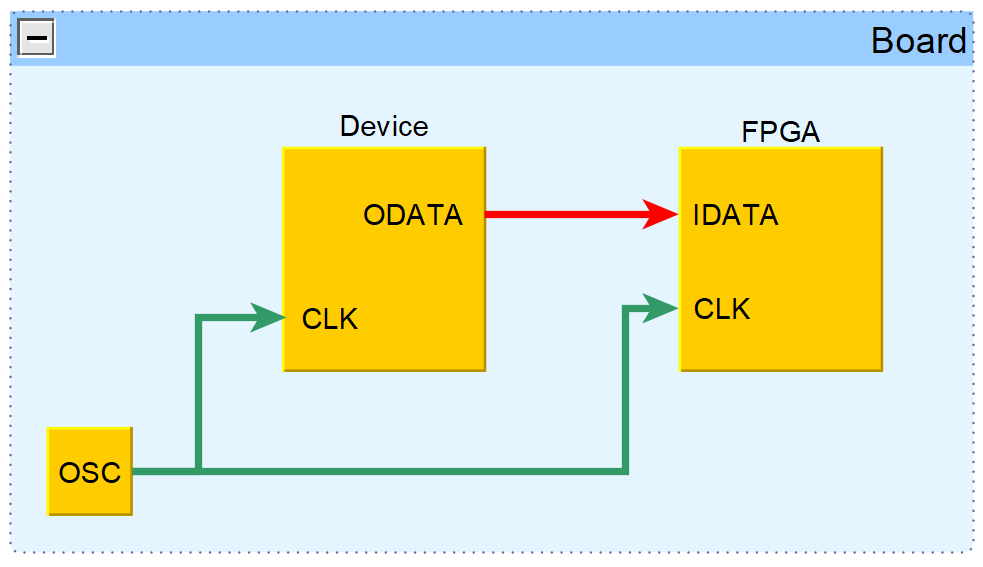


Рисунок 1. Соединение устройств на плате для случая System Synchronous.

Далее в статье будет рассмотрен второй способ, называемый Source Synchronous, при котором источник помимо данных также формирует тактовый сигнал (см. рисунке 2). В дальнейшем для краткости устройство, из которого в FPGA передаются данные и тактовый сигнал, будем иногда называть Device*.*

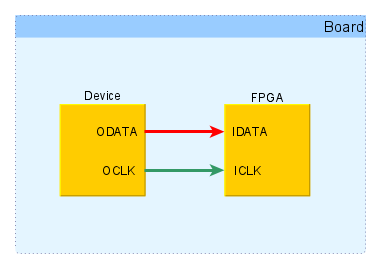


Рисунок 2. Соединение устройств на плате для случая Source Synchronous.

Будем считать, что в FPGA загружен простой проект, состоящий из единственного триггера (см. рисунок 3). Этого вполне достаточно для демонстрации проведения временного анализа для входных сигналов в Vivado. 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

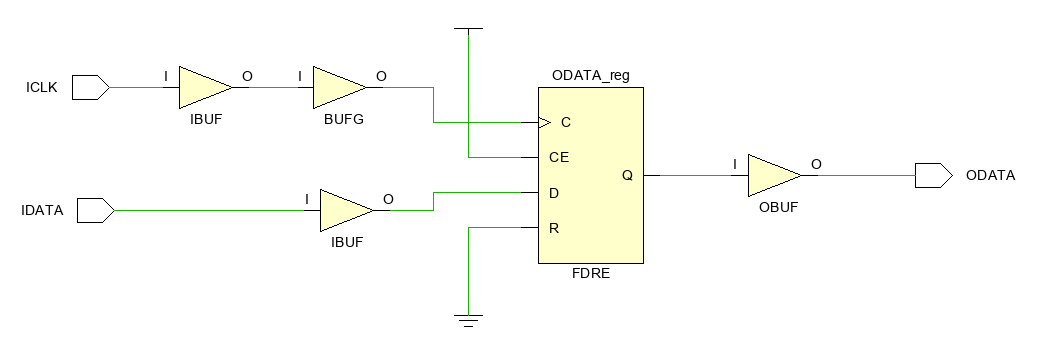


Рисунок 3. Схема FPGA проекта.

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

При передачи данных между Device и FPGA запускающий триггер располагается во внешнем устройстве, а защелкивающий – в FPGA. На рисунке 4 показан анализируемый путь, на который нанесены задержки сигналов. В случае Source Synchronous источник данных также формирует тактовый сигнал, по этой причине на рисунке тактовый генератор (OSC) внесен внутрь Device.

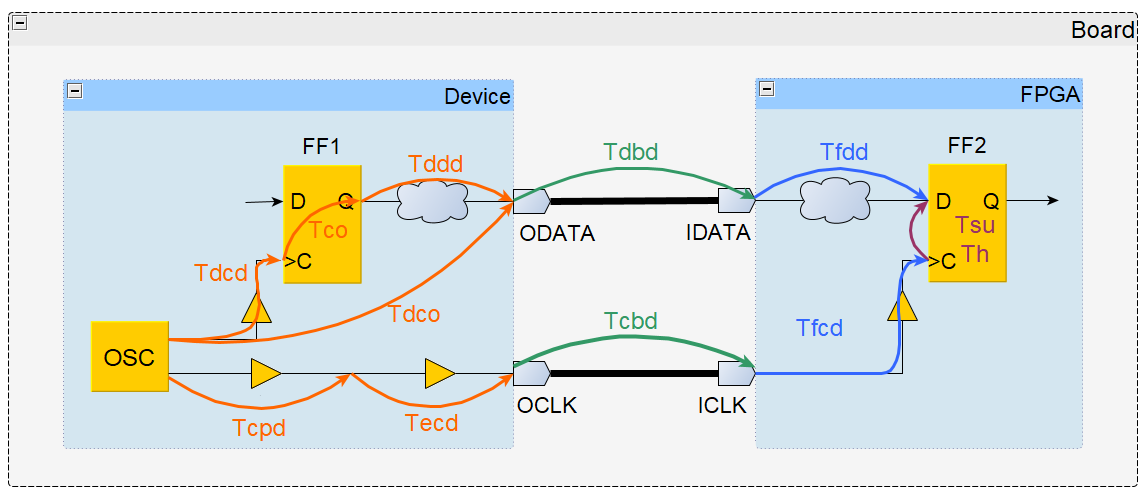


Рисунок 4. Путь с задержками для входных данных и тактового сигнала.

Ниже даны определения задержек, представленных на рисунке 4.

* *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, которая зависит от способа выравнивания его относительно данных;
* *Tfcd* (***F****PGA* ***C****lock* ***D****elay*) – задержка тактового сигнала от ножки ICLK 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 между требуемым и фактическим прибытием данных на вход защелкивающего триггера (*Slack*). 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 задержек распространения данных от запускающего триггера до защелкивающего. Из рисунка 4 получаем следующие результаты:

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

Так как временной анализ проводится для самого пессимистичного случая, выше в одних уравнениях используются максимальные задержки, а в других – минимальные.

Как будет показано далее, расчет требуемого времени прибытия данных *Tdr* для случая Source Synchronous проводится с некоторыми особенностями. Поэтому прежде, чем двигаться дальше, рассмотрим обобщения, применимые к временному анализу сигналов.

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

Для начала рассмотрим анализ по *Setup*. Передача данных 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 распространения тактового сигнала от генератора до входа защелкивающего триггера. Подставив этот результат в уравнение (1), получим

В общем случае задержки тактового сигнала и данных можно разделить на две части, которые соответствую распространению 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.

Если путь передачи данных полностью находится внутри FPGA, то значение суммы *ΣTfpga\_ext* равно нулю. При анализе выходных сигналов слагаемое *Tsu* относится к защелкивающему триггеру вне FPGA и поэтому входит в сумму *ΣTfpga\_ext*. При анализе входных и внутренних для FPGA сигналов слагаемое *Tsu* содержится в сумме *ΣTfpga\_int.*

Например, в [2] для случая System Synchronous получено следующее выражение для *Slack*:

Отсюда очевидны следующие равенства:

Все задержки вне FPGA собраны в сумме *ΣTfpga\_ext* и передаются временному анализатору Vivado в виде единственного значения *input\_delay\_max*.

Важно обратить внимание на слагаемое *Tclk* в уравнении (3), которое появляется из-за того, что данные запускаются по одному фронту, а защелкиваются – по следующему. Анализатор Vivado считает, что передача данных осуществляется именно таким образом.

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

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

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 *Tclk*. Слагаемое *Th* учитывает, что после защелкивающего фронта данные не должны изменяться в течении времени удержания триггера. Подставим выражения для *Tdr\_hold* в уравнение(1) и получим

Если перегруппировать задержки, которые входят в состав слагаемых *Tclk\_delay\_max* и *Tda\_hold*, то уравнение для *Slack* можно переписать в виде:

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

В данном случае имеем:

Для анализа по *Hold* все задержки вне FPGA передаются анализатору Vivado в виде единственного значения *input\_delay\_min*. Наконец, закончив всю подготовительную работу, перейдем непосредственно к рассмотрению временного анализа для входных сигналов для случая Source Synchronous.

1. **Source Synchronous Center Aligned.**

Будем называть окном валидных данных (*data valid window*) промежуток времени, в течении которого предназначенные для передачи данные удерживаются на выходе Device. Рассмотрим случай, когда тактовый сигнал выравнивается относительно середины окна данных. Такой вариант называется Center Aligned и реализуется с помощь дополнительной задержки *Tecd*, значение которой выбирается равным половине периода тактового сигнала (*Tecd* = *Tclk/2*).

На рисунке 5 представлены временные диаграммы сигналов внутри Device для двух периодов тактового сигнала: выход тактового генератора (OSC); тактовый вход запускающего триггера (FF1/C); выходы OCLK и ODATA. Также для больше наглядности на диаграмме обозначены задержки распространения. Номерами №1 и №2 обозначены события для первого и второго тактового фронта. Окну валидных данных на выходе ODATA для фронта №1 соответствует промежуток времени NEW DATA. В дальнейшем будем считать, что фронт №1 формируется на выходе OSC в нулевой момент времени.

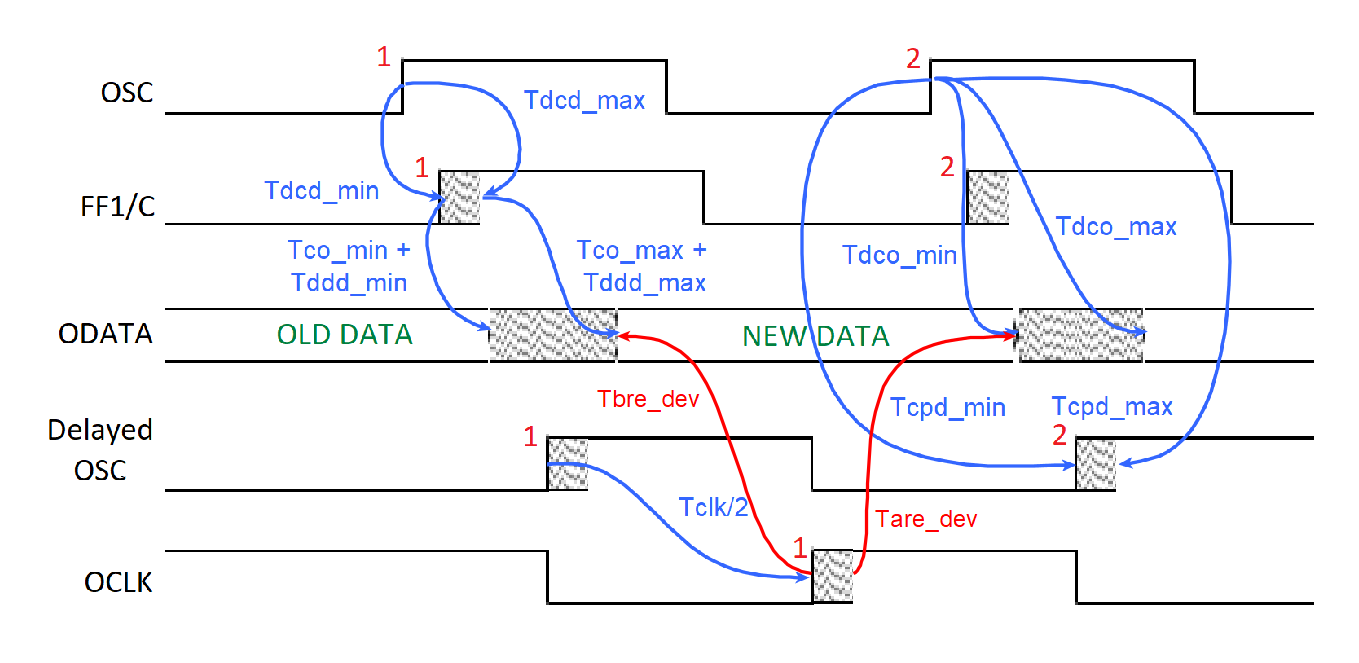


Рисунок 5. Временная диаграмма сигналов внутри Device.

Как можно увидеть, по фронту №1 начинается передача данных от запускающего триггера FF1. Тот же самый фронт спустя некоторую задержку появляется на выход OCLK, распространяется до FPGA и используется для защелкивания данных. Таким образом, данные запускаются и защелкиваются тем же самым фронтом. Это отличается от случая System Synchronous, где данные защелкивались следующим фронтом.

При анализе по *Hold* рассматриваются временные соотношения между текущий запускающим и предыдущим защёлкивающим фронтами. В нашем случае запускающий фронт №1 для текущих данных одновременно является и защелкивающим. Тогда если текущий запускающий фронт появляется в нулевой момент времени, то предыдущий – на один период раньше, то есть в момент времени *–Tclk.*

Учитывая все вышесказанное, получим уравнения для *Slack* при выравнивании Center Aligned. Фактическое время прибытья данных *Tda* вычисляется с помощью рассмотренных ранее уравнений (2). Требуемое время прибытия данных *Tdr* 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), запишем выражения для *Slack* 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). Это важно, так как при проведении анализа Vivado использует именно уравнения (3) и (4).

В представленных выше уравнениях оранжевым цветом обозначены задержки, обусловленные распространением сигналов внутри Device. Обычно производители микросхем явно не приводят данные задержки в datasheet. Вместо этого чаще всего указываются соотношения между границами окна валидных данных и фронтом тактового сигнала на выходе микросхемы. 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:

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

Подставив в уравнения (6) *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 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 точно согласуется с уравнениями (7) и (8). Также комментариях шаблона можно увидеть, что при Center-Aligned 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 запускающий фронт выравнивается относительно левой границы окна валидных данных, что соответствует дополнительной задержке *Tecd* = 0*.* Временные диаграммы сигналов внутри Device при Edge Aligned представлены на рисунке 6.

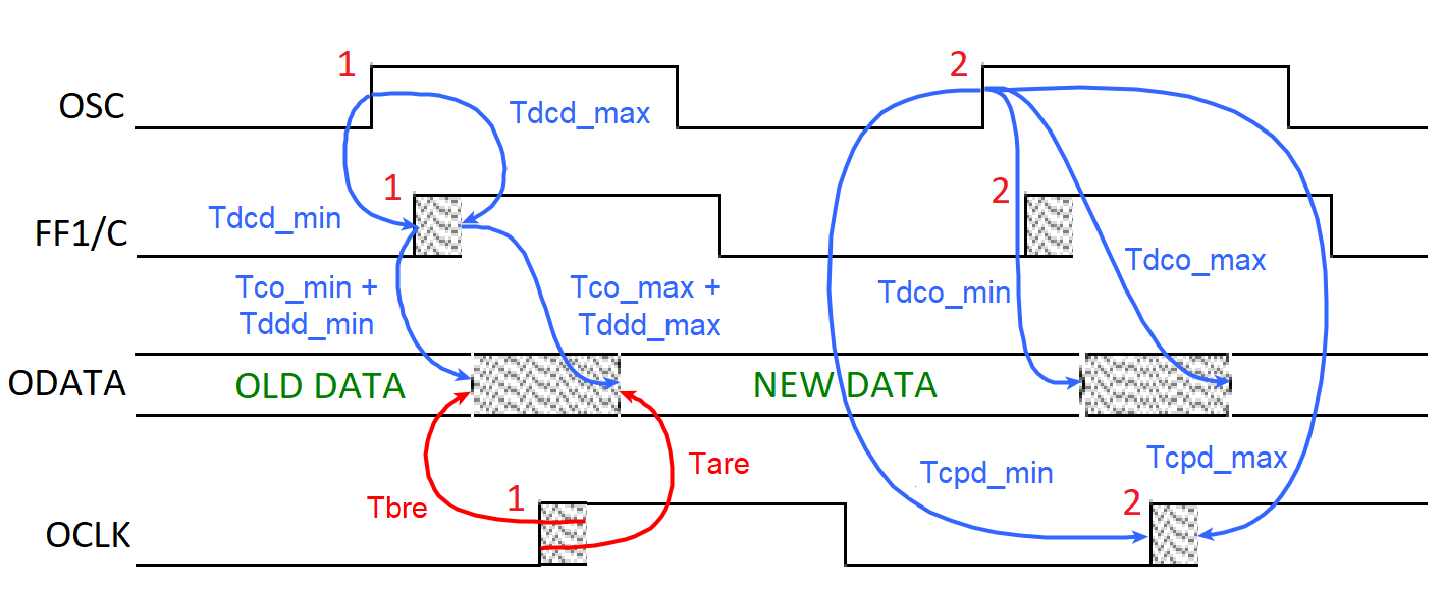


Рисунок 6. Временная диаграмма сигналов внутри Device.

Как при Center Aligned, данные запускаются и защелкиваются по одному и тому же фронту тактового сигнала. С помощью рисунка 6 и уравнений (2), можно получить следующий выражения для *Slack*:

Рассмотрим два временных интервала *Tbre\_dev* и *Tare\_dev*, которые обозначены на рисунке 6 красным цветом. Значение *Tbre\_dev* равно разности между моментом появления тактового фронта и моментом, когда старые данные пропадут с выхода Device. В свою очередь *Tare\_dev* соответствует промежутку времени между установкой на выходе Device 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:

Подставим уравнения (10) в полученные выше уравнения для *Slack* и получим*:*

где, как и для случая Center Aligned, в уравнение для *Slack\_setup* дополнительно добавлены слагаемые *Tclk* с положительным и отрицательным знаком. Сопоставив найденные выражения для *Slack* с уравнениями (3) и (4), можно записать следующие равенства:

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

# 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 еще один вариант передачи данных, который в Vivado называется Edge Aligned MMCM. При этом способе передачи тактовый сигнал выравнивается относительно правой границы окна валидных данных, что соответствует дополнительной задержке *Tecd* = *Tclk.* Тактовый фронт, задержанный на один период, совпадает по времени с появлением следующего фронта. Таким образом при Edge Aligned MMCM данные запускаются текущим фронтом, а защелкиваются – следующим.

Временные диаграммы сигналов внутри Device совпадают с представленным на рисунке 6, однако теперь запускающему фронту соответствует фронт №1, а защелкивающему – фронт №2. Если считать, что фронт №1 появляется в нулевой момент времени, то уравнения для *Slack* примут вид:

Путь временные интервалы *Tbre\_dev* и *Tare\_dev*, а также их пересчитанные значения относительно входов FPGA *Tbre\_fpga* и *Tare\_fpga*, заданы тем же способом, что для случая Edge Aligned, и удовлетворяют уравнениям (9) и (10). Подставив выражения для *Tbre\_fpga* и *Tare\_fpga* в уравнения для *Slack*, получим:

Сравнивая данные результаты с уравнениями (3) и (4), можем записать следующие соотношения:

Vivado Language Templates, соответствующий 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.

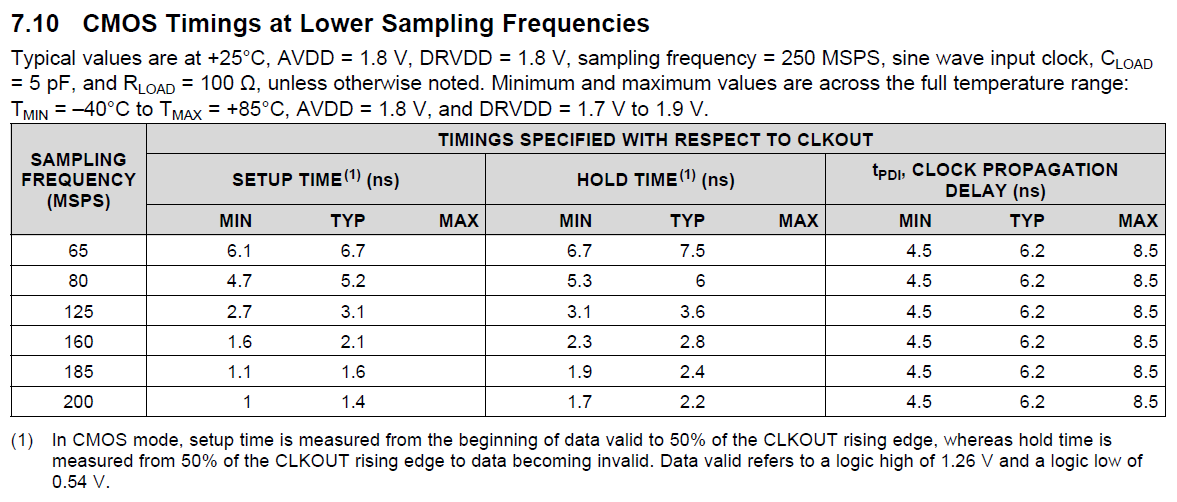
Пусть передающее устройство формирует тактовый сигнал, который выровнен по левому краю окна валидных данных (случай Edge Aligned), и данные запускаются и защелкиваются одним и тем же фронтом. Также пусть перед тем, как поступить на защелкивающий триггер тактовый сигнал проходит через MMCM или PLL. При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

Причина проблем в следующем. Если MMCM (или PLL) тактирует триггер, на который поступает сигнал с входного порта FPGA, то для данного MMCM Vivado автоматически устанавливает компенсацию в режим ZHOLD [4]. В этом режиме 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 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 – защелкивать данные не текущим, а следующим фронтом, что соответствует Edge Aligned MMCM. В качестве альтернативы можно вручную установить MMCM в режим компенсации INTERNAL, однако Xilinx не рекомендует данный способ.

1. **Простой пример для ADS4249.**

В качестве первого практического примера создадим временные ограничения на входные сигналы, поступающие в FPGA из АЦП ADS4249 [5], имеющий два канала по 14 бит каждый. Для краткости создание ограничений будет продемонстрировано для одного бита данных. На рисунке 7 приведены таблица со значениями задержек и временная диаграмма сигналов из datasheet на ADS4249.



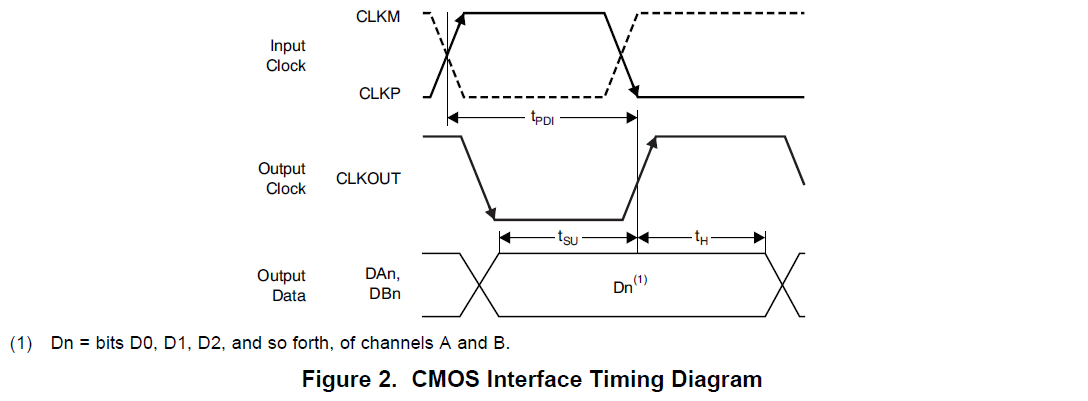


Рисунок 7. Задержки и временные диаграммы для ADS4249.

Будем считать, что минимальные и максимальные задержки распространения данных и тактового сигнала по дорожкам печатной платы известны. В качестве примера примем следующие значения в наносекундах: *Tdbd\_max* = 0.15, *Tdbd\_min* = 0.1, *Tcbd\_max* = 0.12 и *Tcbd\_min* = 0.07. Эти значения заносятся в файл с временными ограничениями (xdc-файл):

# 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

Пусть требуется, чтобы FPGA получал данные на максимальной частоте дискретизации, которая для ADS4249 равна 200 МГц. Тогда ограничение на период тактового сигнала можно записать в виде:

# период тактового сигнала CLKOUT

set Tclk 5

# ограничение на период тактового сигнала

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

Глядя на рисунок 7, становится очевидно, то мы имеем дело c Center Aligned. Также сопоставляя диаграммы для ADS4249 с рисунком 5 получаем, что *Tbre\_dev = Tsu* и *Tare\_dev = Th*. Будем рассматривать самый пессимистичный случай, которому соответствует минимальная ширина окна валидных данных, то есть *Tsu =* 1.7 нс и *Th =* 1 нс. Эти значения также внесем в xdc-файл:

# 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

На этом этапе у нас достаточно информации для расчета *Tbre\_fpga* и *Tare\_fpga* и создания временных ограничений для входного сигнала IDATA. Используем уравнения (7) и (8) и внесем следующие команды в xdc-файл (более подробно о назначении команд и их параметров можно прочитать в [2]):

# 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]

# 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]

Теперь рассмотрим, как введенные ограничения будут отражены во временных отчетах, полученных после размещения и трассировки проекта. На рисунке 8 показаны расчеты фактического и требуемого времени прибытия данных для анализа по *Setup*.

Представленные результаты можно интерпретировать следующим образом. Из раздела Destination Clock Path можно увидеть, что защелкивающий фронт поступает на вход FPGA в момент времени равный 5 нс. С учетом задержки распространения по дорожкам платы *Tcbd\_min* это означает, что на тактовом выходе ADS4249 данный фронт формируется в момент времени 5 – 0.07 = 4.93 нс.

Данные на выходе ADS4249 появляются на *Tsu* нс раньше тактового фронта, после чего распространяются по плате в течении *Tdbd\_max*. Таким образом до ножки IDATA FPGA данные дойдут в момент времени 4.93 – 1 + 0.15 = 4.08 нс. Это соответствует значению задержки входных данных (input delay) из раздела Data Path, так как Vivado считает, что запускающий фронт начинает передачу данных в нулевой момент времени.

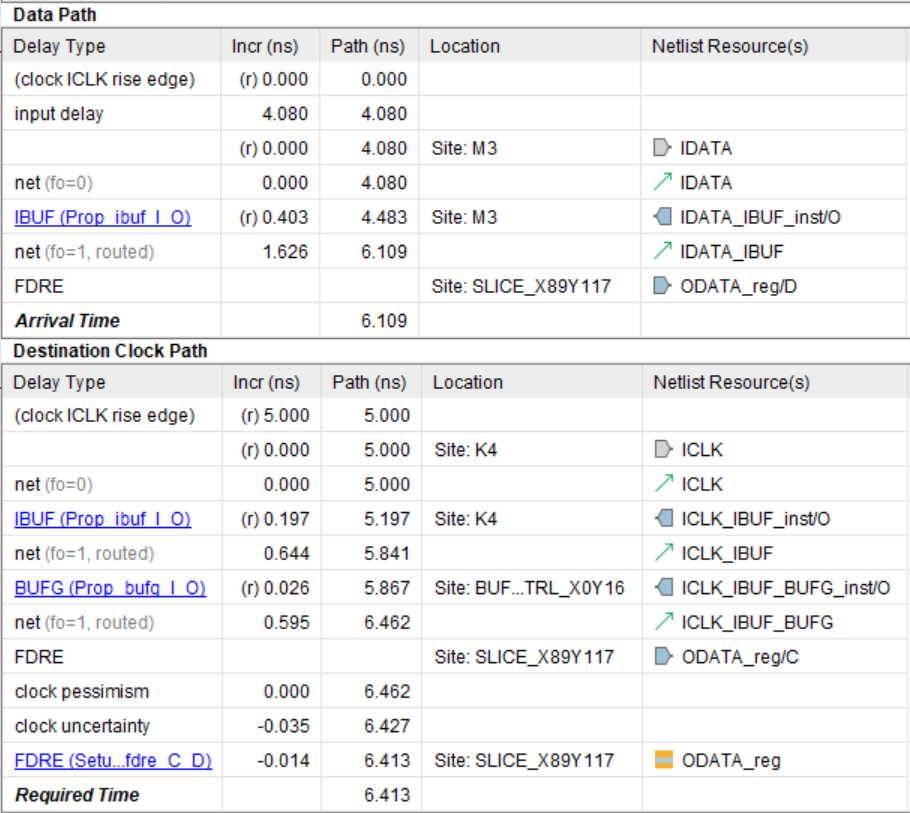


Рисунок 8. Результаты временного анализа по *Setup*.

Можно предложить и другую интерпретацию. Способ, применяемый для расчета окна валидных данных для входа FPGA, можно также использовать для вычисления окна данных относительно входа защелкивающего триггера. Данные будут приняты правильно, если защелкивающий фронт попадет в это окно с учетом времени удержания и установки триггера. Анализ по *Setup* проводится для случая, когда данные распространяются максимально медленно, а защелкивающий фронт – максимально быстро. Если фиксировать окно данных во времени и начать увеличивать скорость распространения тактового сигнала, то это будет соответствовать смещению фронта к левой границе окна (см. рисунок 5).

Величина *Tbre\_fpga* позволяет рассчитать, насколько тактовый сигнал может сдвинуться относительно данных внутри FPGA, чтобы все еще попасть в окно. Для нашего примера значение *Tbre\_fpga* равно 1 + 0.07 – 0.15 = 0.92 нс. Если учесть издержки на время установки триггера, а также джиттер и неопределенность для тактового сигнала, то максимально допустимый сдвига фронта относительно данных составит 0.92 – 0.035 – 0.014 = 0.871 нс.

Из рисунка 8 можно увидеть, что тактовый сигнал внутри FPGA получает суммарную задержку в 1.462 нс, а данные – в 0.403 +1.626 = 2.029 нс. Фактическая задержка данных относительно такового сигнала равна 2.029 – 1.462 = 0.567 нс, что меньше максимально допустимой задержки в 0.871 нс. Величина запаса 0.871 – 0.567 = 0.305 нс совпадает с рассчитанным с помощью рисунку 8 значением *Slack* = 6.413 – 6.109 = 0.304 нс.

Аналогичным образом рассмотрим результаты анализа по *Hold*, представленные на рисунке 9. Анализ по *Hold* проводится для случая, когда данные распространяются максимально быстро, а защелкивающий фронт – максимально медленно. Если зафиксировать во времени окно данных и уменьшать скорость распространения тактового сигнала, то защелкивающий фронт будет смещаться к правой границе окна (см. рисунок 5).

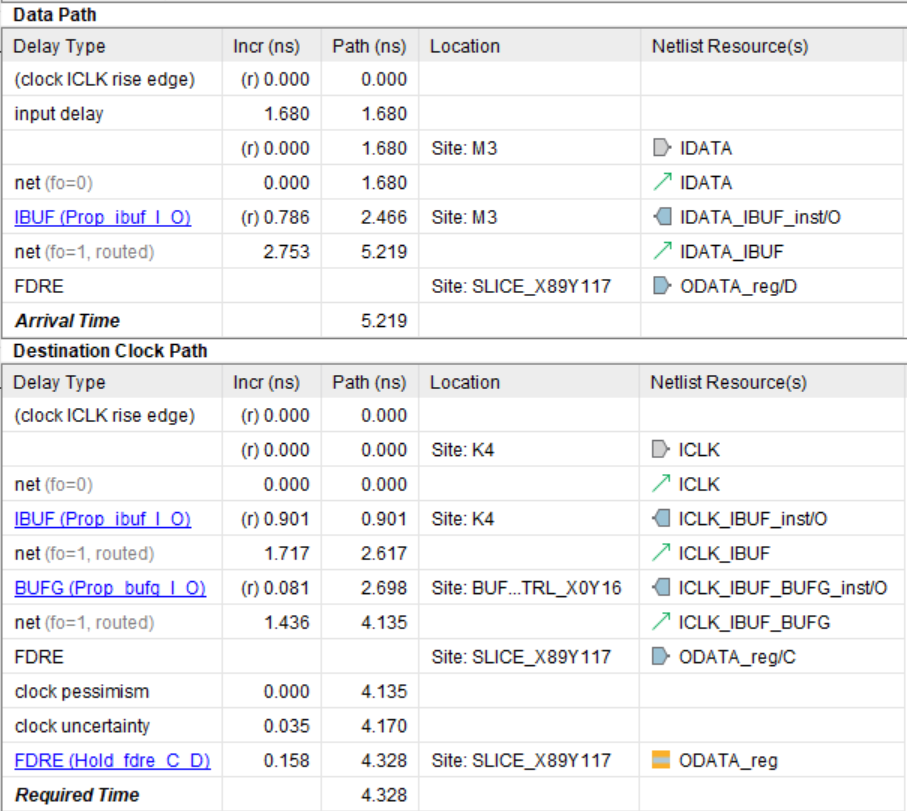


Рисунок 9. Результаты временного анализа по *Hold*.

Для расчета максимально допустимого сдвига тактового сигнала относительно данных внутри FPGA временному анализатору требуется значение *Tare\_fpga*, которое в нашем примере равно 1.7 + 0.1 – 0.12 = 1.68 нс. Тогда с учетом времени удержания триггера и неопределенности тактового сигнала получим, что максимально допустимый сдвига фронта относительно данных составляет 1.68 – 0.035 – 0.158 = 1.487 нс.

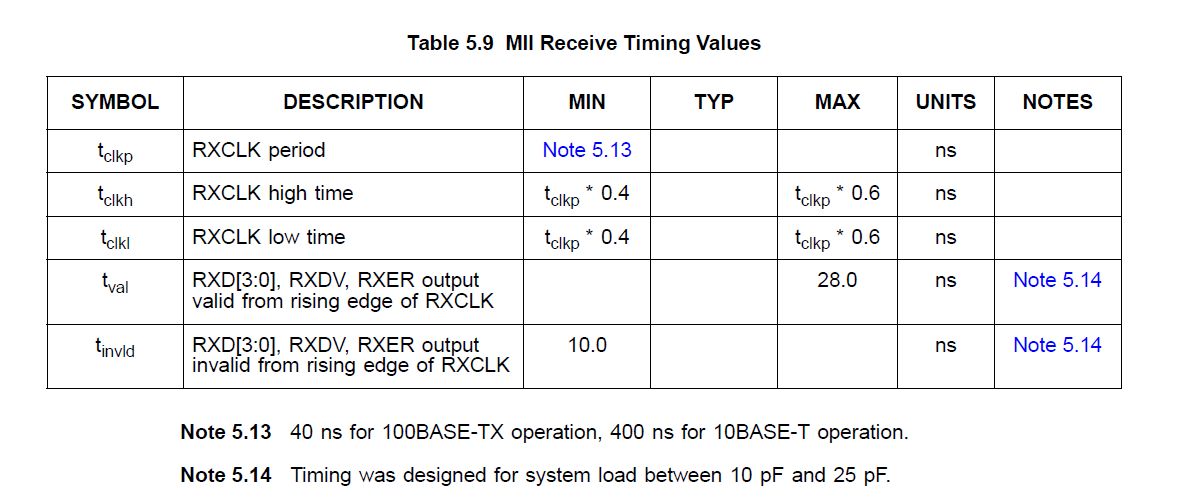
Как видно из рисунка 9 суммарная задержка данных внутри FPGA равна 0.786 + 2.753 = 3.539 нс, а задержка тактового сигнала – 4.135 нс. Фактическая задержка данных относительно такового сигнала рассчитывается как разность этих значений и равна 4.135 – 3.539 = 0.596 нс. Это меньше максимально допустимой задержки в 1.487 нс. Величина запаса составляет 1.487 – 0.596 = 0.891 нс. При расчете с помощью рисунка 9 получается тот же результат для *Slack* = 5.219 – 4.328 = 0.891 нс.

Из представленных выше рассуждений видно, что для интерпретации временных отчетов требуется приложить определенные усилия. Это связано с тем, что в случае Source Synchronous данные запускаются и защелкиваются одним и тем же фронтом тактового сигнала, в то время как Vivado считает, что данные принимаются по следующему фронту.

1. **Более сложный пример для LAN8740A.**

В предыдущем примере для ADS4249 было очевидно, что требуется создавать временные ограничения с использованием шаблона для Center Aligned. Такая ситуация бывает не всегда.

Предположим, что требуется по MII принимать данные от микросхемы Ethernet PHY LAN8740A [6]. На рисунке 10 из datasheet на LAN8740A приведены таблица со значениями задержек и временная диаграмма сигналов, которые потребуются в дальнейшем. Для удобства на временную диаграмму добавлены номера фронтов тактового сигнала, а данные, которые требуется принять, отмечены цветом.



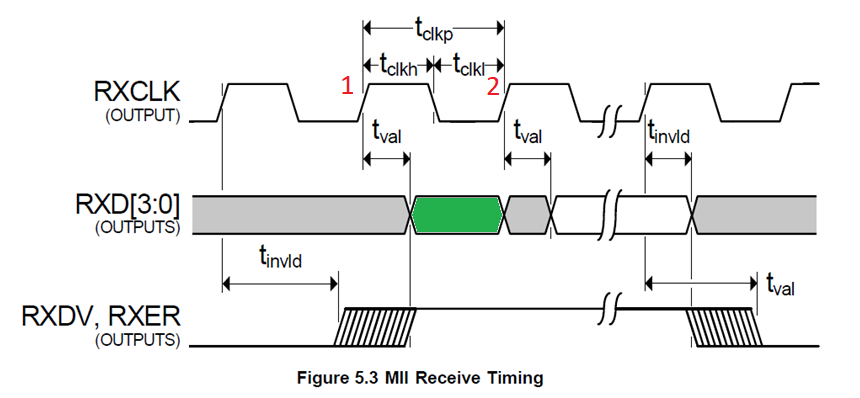


Рисунок 10. Задержки и временные диаграммы для LAN8740A.

Из таблицы задержек получаем, что старые данные удерживаются на шине RXD в течении *Тinvld =* 10 нс после фронта №1. Также видно, что нужные данные появляются на выходе LAN8740A спустя *Тval =* 28 нс. Будем считать, что LAN8740A работает режиме 100BASE-TX, и зададим значения задержек распространения сигналов по дорожкам платы, как и в предыдущем примере. Тогда в xdc-файл следует занести следующие команды:

# период тактового сигнала

set Tclk 40

# ограничение на период тактового сигнала

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

# 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 старых данных после тактового сигнала на выходе LAN8740A

set Tinvld 10

# время между появлением тактового сигнала и данных на выходе LAN8740A

set Tval 28

Если перерисовать временные диаграммы, соблюдая масштабы задержек, то способ выравнивания тактового сигнала относительно данных станет более очевидным. Однако давайте просто последовательно разберем все три возможных варианта создания ограничений.

Сначала попробуем защелкнуть интересующие нас данные по тактовому фронту №1, что соответствует случаю Edge Aligned. Если сопоставить временные диаграммы для LAN8740A с рисунком 6, то становится очевидно равенство *Tare\_dev = Тval*. Однако, на рисунке 6 предполагается, что старые данные пропадают с выхода Device до появления защелкивающего фронта, а на диаграммах для LAN8740A – после. Это расхождение можно учесть с помощью знака задержки, то есть *Tbre\_dev =* –*Тinvld.* Тогда, если вспомнить уравнения (11), то оставшаяся часть xdc-файла примет вид:

# исчезновения старых данных до тактового сигнала на выходе Device

set Tbre\_dev -$Tinvld

# появления новых данных после тактового сигнала на выходе Device

set Tare\_dev $Tval

# исчезновения старых данных до тактового сигнала на входе FPGA

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

# появления новых данных после тактового сигнала на входе FPGA

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

# 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]

В дальнейшем для краткости будем рассматривать только анализ по *Setup*. На рисунке 11 показан раздел Summary временного отчета, в котором отрицательное значение *Slack* указывает на нарушение временных ограничений. Если считать, что фронт №1 формируется на выходе генератора в нулевой момент времени, то на входе FPGA он появится спустя *Tcbd\_min* = 0.07 нс. Передаваемые данные дойдут до FPGA в момент времени равный *Тval + Tdbd\_max* = 28.15 нс. Такой большой разброс между временем прихода защелкивающего фронта и данных является причиной отрицательного значения *Slack =* – 27.119 нс.

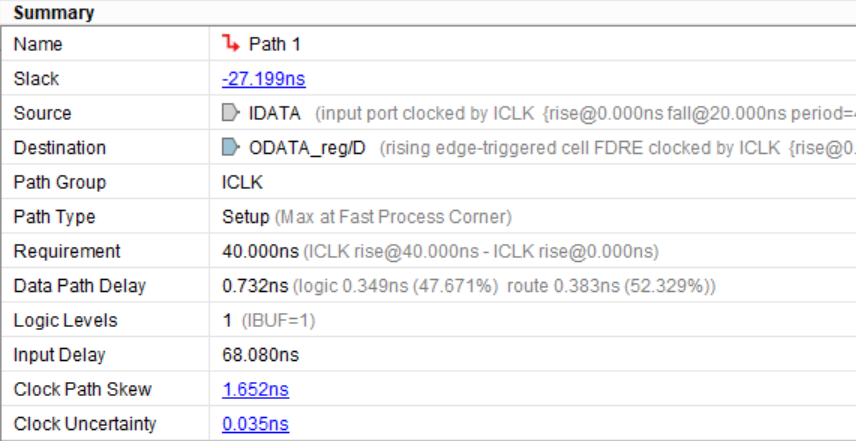


Рисунок 11. Результаты временного анализа (Edge Aligned).

С этой проблемой можно справиться, если защелкивать данный по фронту №2, что соответствует случаю Edge Aligned MMCM. Для создания ограничений теперь необходимо использовать уравнения (12). Содержимое xdc-файла почти полностью совпадает с ограничениями для Edge Aligned и представлено ниже:

# исчезновения старых данных до тактового сигнала на выходе Device

set Tbre\_dev -$Tinvld

# появления новых данных после тактового сигнала на выходе Device

set Tare\_dev $Tval

# исчезновения старых данных до тактового сигнала на входе FPGA

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

# появления новых данных после тактового сигнала на входе FPGA

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

# 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]

Если подставить в уравнения (12) числовые значения всех задержек, то получим следующий результат:

На рисунке 12 показан раздел Summary временного отчета для случая Edge Aligned MMCM. Можно увидеть, что теперь *Slack =* 12.767 нс имеет положительное значение, а значит временные ограничения выполнены.

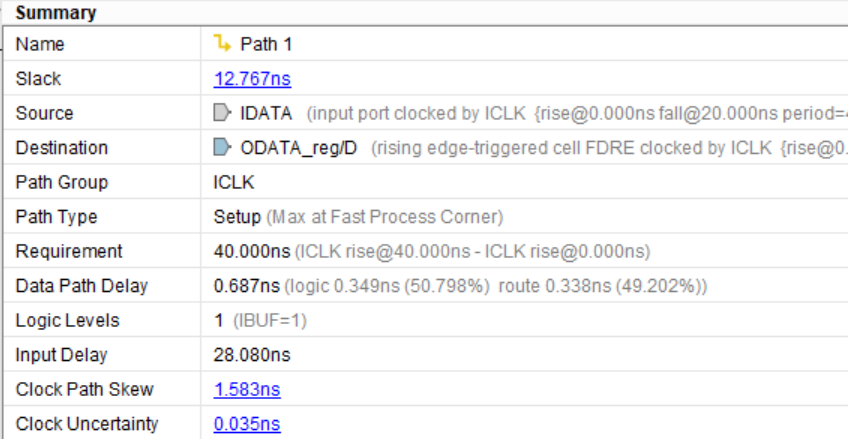


Рисунок 12. Результаты временного анализа (Edge Aligned MMCM).

В качестве последнего примера рассмотрим, что произойдет, если попытаться защелкнуть данные по тактовому фронту №2, но задать ограничения как Center Aligned. Из временных диаграмм видно, что после появления фронта №2 данные будут удерживаться на выход LAN8740A в течении *Тinvld* нс, то есть *Tare\_dev = Тinvld*. С помощью рисунка 5 также легко получить равенство *Tbre\_dev = Тclk – Тval*. С учетом уравнений (7) и (8) внесем следующие команды в файл временных ограничений:

# время удержания данных после тактового сигнала на выходе Device

set Tare\_dev $Tinvld

# время между появлением данных и тактовым сигналом на выходе Device

set Tbre\_dev [expr $Tclk - $Tval]

# 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]

# 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]

Если изучить раздел Summary временного отчета для выравнивания Center Aligned, то можно увидеть, что он полностью совпадает с рисунком 12. Далее подставив в уравнения (7) и (8) числовые значения всех задержек, то получим, что

Данные результаты также совпадают задержками при Edge Aligned MMCM. Это не удивительно, так как в обоих случаях одни и те же данные защелкивались по одному и тому же фронту. Это пример показывает, что при создании ограничений на входные сигналы временные диаграммы можно рассмотреть с различных сторон и выбрать вариант наиболее удобный с точки зрения простоты интерпретации и создания временных ограничений.

**Заключение.**

В статье был рассмотрен временной анализ при Source Synchronous передаче сигналов в FPGA из внешнего устройства. Показан вывод уравнений статического временного анализа для трех различных вариантов выравнивания данных относительно тактового сигнала. Разобраны два примера создания временных ограничений.

**Ссылки.**

1. [Основы статического временного анализа. Часть 1: Period Constraint](https://fpga-systems.ru/static-timing-analysis-part-1-period-constraints)
2. Основы статического временного анализа. Часть 2.1: System Synchronous Input Delay Constraint
3. Основы статического временного анализа. Часть 2.2: System Synchronous Output Delay Constraint
4. 7 Series FPGAs Clocking Resources (UG 472)
5. Datasheet ADS4249
6. Datasheet LAN8740A