**Основы статического временного анализа.**

**Часть 2.1: System Synchronous** **Input Delay Constraint.**

**Введение.**

Данная статья является второй в серии статей по временным ограничениям в FPGA. Основная цель – познакомить начинающих разработчиков с основами статического временного анализа. Будет рассмотрен временной анализ при передачи данных в FPGA из внешнего устройства и показано два способа создания ограничений для входных сигналов.

1. **Цель временных ограничений на входные сигналы.**

В цифровых синхронных устройствах данные передаются между двумя триггерами, которые разделены комбинационной логикой. В предыдущей статье был представлен временной анализ для случая, когда оба триггера располагаются внутри FPGA [1]. Предполагается, что читатель уже знаком с такими понятиями, как ограничения на максимальное и минимальное время распространения, время установки и удержания, *Slack* и т.д.

Зачастую обрабатываемые данные передаются между FPGA и другими микросхемами, расположенными на той же печатной плате. Эти пути также должны удовлетворять ограничениям по *Setup* и *Hold*, чтобы плата как единое целое могла работать корректно. Поэтому практически всегда для входных сигналов FPGA требуется вводить временные ограничения.

В качестве примера будем рассматривать устройство, схема которого показана рисунке 1. Помимо FPGA на плате присутствует микросхема 74HC595 [2], которая представляет из себя последовательный регистр сдвига и часто поставляется в составе обучающих наборов для Arduino. Также на плате располагается тактовый генератор (OSC), формирующий тактовый сигнала для FPGA и 74HC595.

Данная конфигурация, при которой тактовый сигнал для источника и приемника данных поступает от внешнего генератора, называется System Synchronous. Другой вариант, когда источник помимо данный также формирует и тактовый сигналы, называется Source Synchronous.

На рисунке 1 отмечены только те ножки микросхемы 74HC595, которые будут рассматриваться в последующем. Ножка *SER* соответствует входу регистра сдвига, ножка QH’ – его выходу. Регистр сдвига тактируется сигналом, который поступает на ножку SRCLK. В дальнейшем микросхему 74HC595 для краткости будем называть Device*.* Далее будет представлен временной анализ для передачи данных с выхода QH’ Device на вход IDATA FPGA. На рисунке 1 данный путь отмечен красным цветом.

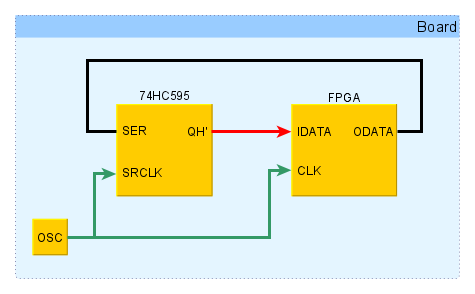


Рисунок 1. Схема соединения устройств на плате.

Также пусть FPGA загружен простой проект, состоящий из двух триггеров и двух LUT, которые реализуют логическое отрицание. Схема проекта показана на рисунке 2.

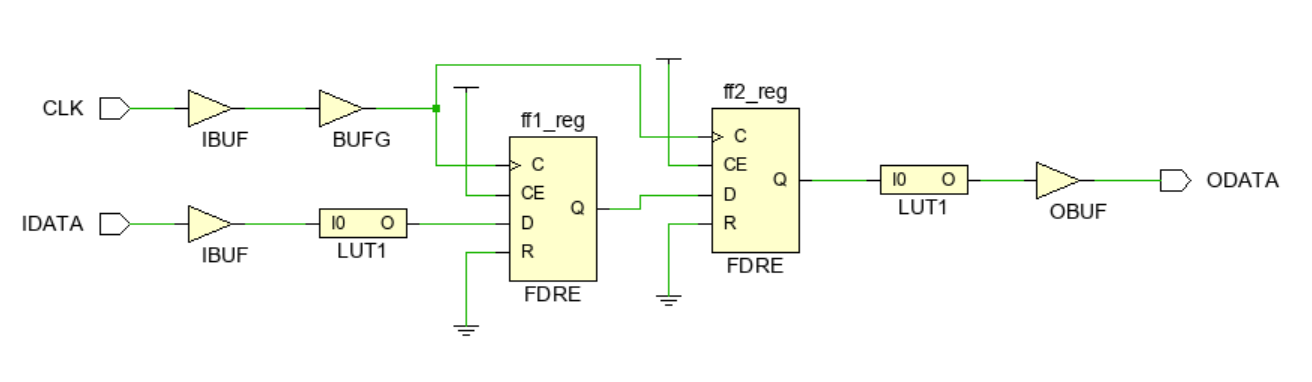


Рисунок 2. Схема FPGA проекта.

Описание проекта на System Verilog представлено ниже:

module top (

input  logic CLK,

    input  logic IDATA,

    output logic ODATA

);

    logic ff1, ff2;

    always\_ff@(posedge CLK)

       ff1 <= ~IDATA;

    always\_ff@(posedge CLK)

       ff2 <= ff1;

   assign ODATA = ~ff2;

endmodule

Данный проект не имеет какой-либо практической ценности, однако на его примере можно продемонстрировать, как проводится временной анализ для входных сигналов.

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

Анализ передачи данных между Device и FPGA мало отличается от случая, рассмотренного в [1] для двух триггеров внутри FPGA. Особенностью является то, что запускающий триггер располагается в одном устройстве, а защелкивающий в другом. На рисунке 3 показан анализируемый путь, на который нанесены задержки для данных и тактового сигнала.

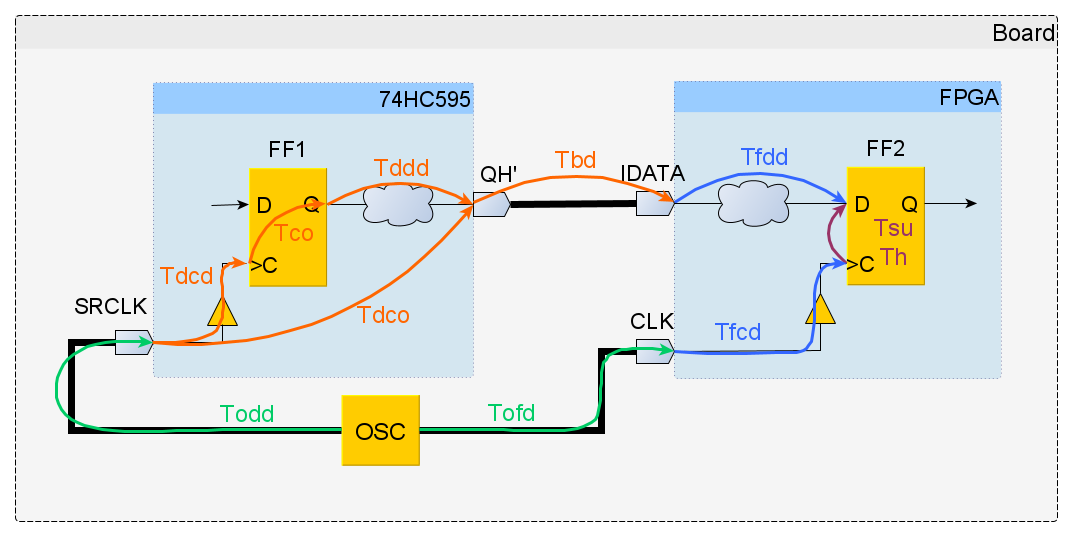


Рисунок 3. Путь с задержками для входных данных и тактового сигнала.

Ниже даны определения задержек, представленных на рисунке 3.

* (***O****sc to* ***D****evice* ***D****elay*) – задержка тактового сигнала от генератора до ножки SCLR микросхемы 74HC595;
* (***O****sc to* ***F****PGA* ***D****elay*) – задержка тактового сигнала от генератора до ножки CLK FPGA;
* (***D****evice* ***C****lock* ***D****elay*) – задержка тактового сигнала от ножки SCLR микросхемы 74HC595 до тактового входа запускающего триггера;
* (***C****lock to* ***O****utput*) – интервал времени между приходом фронта на тактовый вход триггера и появлением данных на его выходе Q;
* (***D****evice* ***D****ata* ***D****elay*) – задержка распространения данных от запускающего триггера до ножки QH’ микросхемы 74HC595;
* (***B****oard* ***D****elay*) – задержка распространения данных между микросхемой 74HC595 и FPGA по дорожкам платы;
* (***F****PGA* ***C****lock* ***D****elay*) – задержка тактового сигнала от ножки CLK FPGA до тактового входа защелкивающего триггера;
* (***F****PGA* ***D****ata* ***D****elay*) – задержка распространения данных от ножки IDATA FPGA до защелкивающим триггером;
* (***S****et****U****p time*) – время установки триггера.
* (***H****old time*) – время удержания триггера.

Период тактового сигнала будем обозначать как *Tclk*. Красным и зеленым цветом на рисунке 3 представлены задержки для участков пути, которые располагаются вне FPGA. Эти задержки необходимо самостоятельно указать временному анализатору Vivado.

1. **Максимальное время распространения.**

Для начала рассмотрим, каким образом выполняется анализ для проверки ограничения на максимальное время распространения (*Setup*). Напомним, что временной анализ по *Setup* всегда проводится для самого пессимистичного случая, которому соответствует максимально задержанный запускающий фронт, максимально медленное распространение данных и максимально быстро распространяющийся защелкивающий фронт.

Сначала найдем фактическое время прибытия данных ко входу расположенного внутри FPGA защелкивающего триггера. Уравнения для расчета представлены ниже (см. рисунок 3):

* Время прибытия фронта к запускающему триггеру в внутри Device (***S****ource* ***С****lock* ***A****rrival time*):
* Задержка распространения данных (***D****ata* ***D****elay*):
* Время прибытия данных на вход защелкивающего триггера внутри FPGA (***D****ata* ***A****rrival time*):

Подставив (1) и (2) в уравнение для , получим:

Введем обозначение:

Задержкасоответствует интервалу времени между приходом фронта на тактовый вход микросхемы 75HC595 и появлением данных на ее выходе QH’.Тогда уравнение для ***D****ata* ***A****rrival time* можно записать в виде:

Теперь вычислим время требуемое время прибытия данных. Защелкивающий фронт появляется через один такт после запускающего фронта, поэтому к задержке распространения добавлен один период тактового сигнала.

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

В предыдущем уравнении учитывается, что данные на входе защелкивающего триггера должны быть стабильны в течении времени *Tsu* до прихода фронта тактового сигнала.

Для анализа по *Setup* величина запаса для задержки данных (*Slack*) вычисляется по формуле:

Если *Slack* принимаетотрицательное значение, то это указывает, что данные приходят на вход защелкивающего триггера позже, чем требуется. То есть,ограничение по *Setup* нарушено. Используя ранее полученные уравнения, можно получить полное выражение для расчета *Slack*

1. **Минимальное время распространения.**

Теперь рассмотрим, как выполняется анализ для проверки ограничения на минимальное время распространения (*Hold*). При анализе по *Hold* считается, что задержки для запускающего фронта и данных имеют минимальное значение, а для защелкивающего фронта – максимальное.

Расчет фактического времени прибытия данных представлен ниже:

* Время прибытия фронта к триггеру внутри Device (***S****ource* ***С****lock* ***A****rrival time*):
* Задержка распространения данных (***D****ata* ***D****elay*):
* Время прибытия данных к триггеру внутри FPGA (***D****ata* ***A****rrival time*):

где, как и ранее используется обозначение

Далее представлены уравнения для вычисления требуемого времени прибытия данных:

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

Напомним, что защелкивающий фронт для предыдущих данных появляется в тот же момент времени, что и запускающий фронт для следующих данных. По этой причине в задержке распространения отсутствует слагаемое, равное периоду тактового сигнала. Также отметим, что слагаемое в уравнении для учитывает, что данные не должны изменяться в течении времени удержания после защелкивающего фронта.

Уравнение для расчета *Slack* при анализе по *Hold* имеет вид:

Используя полученные выше результаты, выражение для *Slack* можно записать в виде:

1. **Первый способ создания временных ограничения в Vivado.**

Перейдем от теории к практике и рассмотрим первый способ создания временных ограничений для входного сигнала. Для начала разберемся с ограничениями для анализа по *Setup*.

В уравнении (1) слагаемые, выделенные зеленым и оранжевым цветом, неизвестными анализатору Vivado, так как они описывают задержки вне FPGA. Также неизвестным является период тактового сигнала . Создание ограничений на период тактового сигнала с помощью команды create\_clock было рассмотрено в [1].

Для определенности будем считать, что источник питания для микросхемы 74HC595 имеет напряжение 4.5 В. Из таблицы 7.6 datasheet для 74HC595 [2] находим, что максимальная рабочая частота равна 31 МГц. Также пускай требуется, чтобы плата на рисунке 1 могла работать, когда частота тактового генератора (OSC) равна 10 МГц. Тогда ограничение на период тактового сигнала можно записать в виде:

# ограничение на период тактового сигнала

create\_clock -period 100 -name clk\_10MHz [get\_ports CLK]

Данная команда создает тактовый сигнал clk\_10MHz с периодом 100 нс, который поступает в FPGA на ножку CLK.

Объединим все оставшиеся в уравнении (1) неизвестные слагаемые в одну переменную . Тогда выражение для *Slack* можно переписать в виде:

где

В слагаемые , и задают задержки обусловленные распространением сигнала по дорожкам печатной платы. Их значения зависят от многих факторов: материал подложки; длины дорожек; типы дорожек, например, полосковые или микрополосковые и т.д.

Для каждого типа дорожки существуют приближенные выражения для оценки скорости распространения по ней сигнала. Зная скорость и длину дорожки, можно оценить задержку. Пример приближенных расчетов для разных типов дорожек можно найти в [3].

Будем считать, что мы смогли оценить минимальные и максимальные задержки распространения сигнала по дорожкам печатной платы. В качестве примера примем следующие значения в наносекундах: , , , , и . Значения задержек можно указать в файле с временными ограничениями (xdc-файл) в следующем виде:

# минимальное и максимальное время распространения данных по

# дорожкам платы

set Tbd\_max 0.6

set Tbd\_min 0.5

# минимальное и максимальное время распространения тактового сигнала

# от генератора до микросхемы 74HC595

set Todd\_max 0.4

set Todd\_min 0.2

# минимальное и максимальное время распространения тактового сигнала

# от генератора FPGA

set Tofd\_max 0.3

set Tofd\_min 0.2

Рассмотрим задержку и уравнения (3), которая соответствует интервалу времени между приходом фронта на тактовый вход SCLK микросхемы 75HC595 и появлением данных на ее выходе QH’. Ее значение можно получить из таблицы 7.6 datasheet для 74HC595 [2]. Данная задержка обозначена как и при напряжении питания 4.5 В может изменяться от 17 до 32 нс. Эти значения также вносятся в xdc-файл:

# задержка распространения между приходом тактового сигнала и

# появлением данных на выходе QH' микросхемы 74HC595

set Tdco\_min 17

set Tdco\_max 32

Теперь можно создать временные ограничения для анализа по *Setup* для входного сигнала. Для этого в файл с ограничениями нужно внести следующие команды [4]:

# временные ограничение для входного сигнала IDATA

set idelay\_max [expr $Todd\_max - $Tofd\_min + $Tdco\_max + $Tbd\_max]

set\_input\_delay -clock clk\_10MHz -max $idelay\_max [get\_ports IDATA];

В первой строке объявляется переменная idelay\_max, которая приравнивается из уравнения (3). Далее с помощью команды set\_input\_delay создаются ограничения для входного сигнала. Опция -max $idelay\_max задает задержку для анализа по *Setup*. Конструкция [get\_ports IDATA] указывает, что ограничения создаются для входного сигнала, который поступает в FPGA на ножку IDATA.

Важно отметить, что так как запускающий триггера располагается не в FPGA, анализатору Vivado необходимо указать, каким сигналом тактируется данный триггер. Это делается с помощью опции -clock clk\_10MHz. Данный тактовый сигнал был создан ранее с помощью команды create\_clock. Защелкивающий триггер располагается внутри FPGA, поэтому временной анализатор знает его тактовый сигнал.

Аналогичным образом создаются ограничения для анализа по *Hold*. Объединив все неизвестные слагаемые в уравнении (2) в одну переменную, выражение для *Slack* можно записать в виде:

где

Команды, которые требуется добавить в xdc-файл представлены ниже:

# временные ограничение для входного сигнала IDATA

set idelay\_min [expr $Todd\_min - $Tofd\_max + $Tdco\_min + $Tbd\_min]

set\_input\_delay -clock clk\_10MHz -min $idelay\_min [get\_ports IDATA];

Как и ранее, сначала объявляется переменная, значение которой равно из уравнения (4), после чего с помощью команды set\_input\_delay создаются временные ограничения. Опция -min указывает, что ограничения предназначены для проведения анализа по *Hold*.

Полное содержимое xdc-файла представлено ниже:

# задержка распространения между приходом тактового сигнала и

# появлением данных на выходе QH' микросхемы 74HC595

set Tdco\_min 17

set Tdco\_max 32

# минимальное и максимальное время распространения данных по

# дорожкам платы

set Tbd\_max 0.6

set Tbd\_min 0.5

# минимальное и максимальное время распространения тактового сигнала

# от генератора до микросхемы 74HC595

set Todd\_max 0.4

set Todd\_min 0.2

# минимальное и максимальное время распространения тактового сигнала # от генератора FPGA

set Tofd\_max 0.3

set Tofd\_min 0.2

# ограничение на период тактового сигнала

create\_clock -period 100 -name clk\_10MHz [get\_ports CLK]

# временные ограничение для входного сигнала IDATA

set idelay\_max [expr $Todd\_max - $Tofd\_min + $Tdco\_max + $Tbd\_max]

set\_input\_delay -clock clk\_10MHz -max $idelay\_max [get\_ports IDATA];

set idelay\_min [expr $Todd\_min - $Tofd\_max + $Tdco\_min + $Tbd\_min]

set\_input\_delay -clock clk\_10MHz -min $idelay\_min [get\_ports IDATA];

Рассмотрим, как введенные ограничения будут отражаться во временных отчетах, полученных после размещения и трассировки проекта. На рисунке 4 представлен раздел *Summary* для анализа по *Setup*, в котором указывается источник сигнала (ножка IDATA), защелкивающий триггер (ff1\_reg), задержка данных внутри FPGA (Data Path Delay) и количество уровней логики (Logic Levels). Также можно увидеть полученный *Slack*, расфазировку (Skew) и неопределенность (Uncertainty) для тактового сигнала.

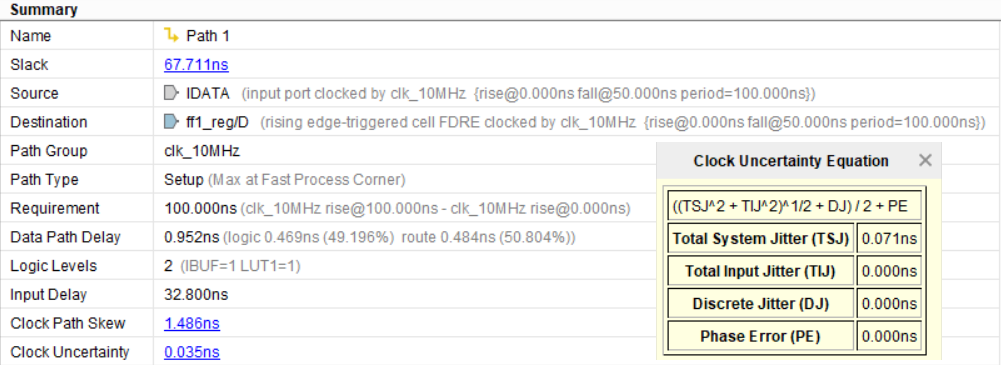
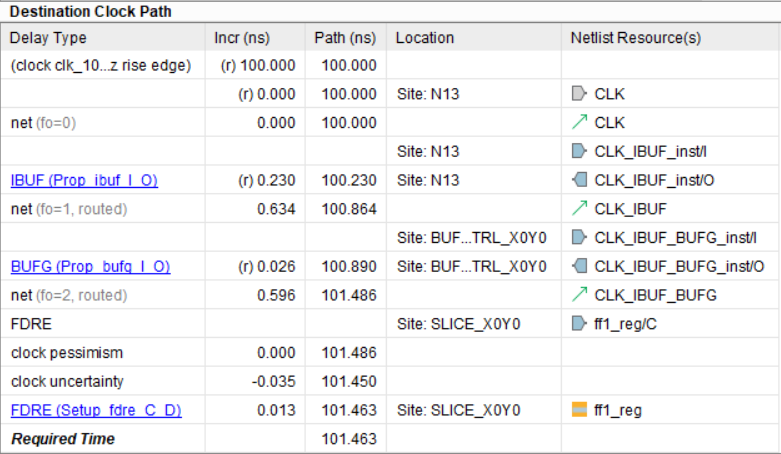
**

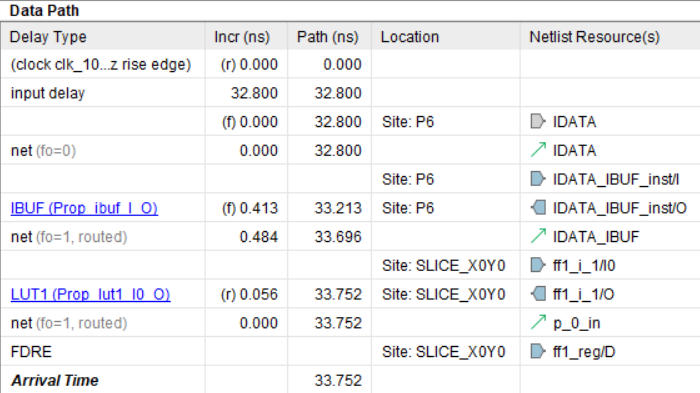
Рисунок 4. Общие сведения о входном пути (*Setup*).

На рисунках 5 и 6 показаны задержки для защелкивающего фронта и данных соответственно. Эти результаты следует интерпретировать следующим образом. Из рисунка 5 следует, что защелкивающий фронт приходит на ножку CLK FPGA в момент времени 100 нс. Далее он распространяется через входной и тактовый буферы и через 1.486 нс попадает на защелкивающий триггер. Учитывая время установки и неопределенность тактового сигнала, получаем, что данные должны достигнуть входа защелкивающего триггера через 101.463 нс.

**

Рисунках 5. Расчет требуемого времени прибытия данных (*Setup*).

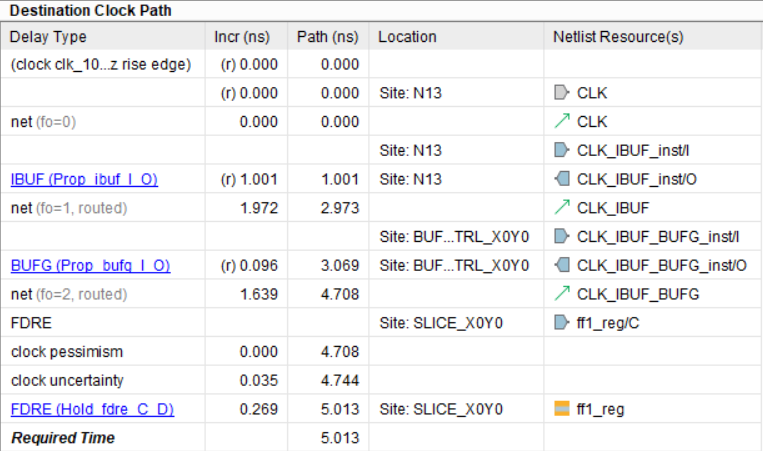
Чтобы попасть на ножку FPGA через 100 нс, защелкивающий фронт должен быть сформирован на выходе тактового генератора в момент времени нс. Запускающий фронт появляется на один период раньше, то есть когда время равно -0.2 нс. Далее через нс этот фронт дойдет до ножки SCLK микросхемы 74HC595 и запустит передачу данных. Спустя еще нс данные достигнут ножки FPGA. Это произойдет, когда времени равно нс, что соответствует задержке *Input Delay* во второй строке рисунка 6.

**

Рисунках 6. Расчет фактического времени прибытия данных (*Setup*).

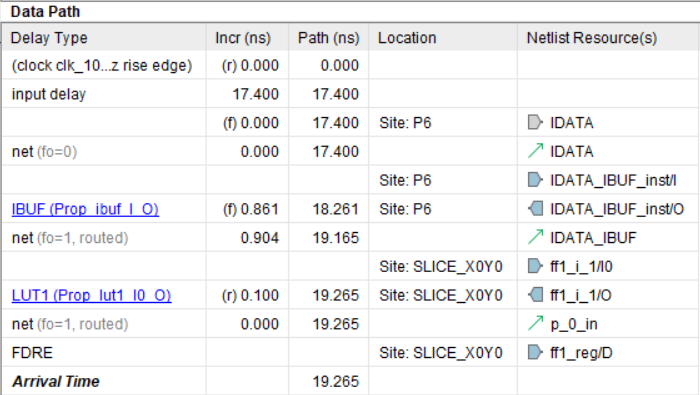
Внутри FPGA данные проходят через входной буфер и LUT и попадают на вход защелкивающего триггера в момент времени 33.752 нс. Отсюда получаем, что *Slack* равен нс.

Ту же картину получаем для анализа по Hold. На рисунке 7 можно увидеть, что защелкивающий фронт попадает в FPGA в нулевой момент времени. С учетом задержки распространения это означает, что он появился на выходе генератора на нс раньше. Для анализа по *Hold* запускающий и защелкивающий фронты появляются в одно и тоже время, поэтому на вход SCLK микросхемы 74HC595 фронт попадет в момент времени нс.

**

Рисунках 7. Расчет требуемого времени прибытия данных (*Hold*).

Спустя нс данные появятся на выходе QH’ микросхемы 74HC595. Далее они будут распространятся по дорожке платы в течении нс и попадут на ножку CLK FPGA. Это произойдет в момент времени нс, что соответствует задержке *Input Delay* на рисунке 8.

**

Рисунках 8. Расчет фактического времени прибытия данных (*Hold*).

Данный подход прост тем, что достаточно представить все неизвестные задержки в виде одной величины . Однако, как можно было увидеть, это приводит к тому, что интерпретация полученных результатов требует определенных усилий.

1. **Второй способ создания временных ограничения в Vivado.**

Теперь рассмотрим второй способ создания временных ограничений для входного сигнала. В xdc-файле можно отдельно указывать задержки распространения тактового сигнала по дорожкам печатной платы. Для этого используется команда set\_clock\_latency [4]:

# ограничение на период тактового сигнала, поступающего в FPGA

create\_clock -period 100 -name clk\_10MHz [get\_ports CLK]

# задержки распространения тактового сигнала от генератора до FPGA

set\_clock\_latency -source -early $Tofd\_min [get\_clocks clk\_10MHz]

set\_clock\_latency -source -late $Tofd\_max [get\_clocks clk\_10MHz]

C помощью команды create\_clock создается ограничение на период тактового сигнала, который поступает на ножку CLK FPGA. Далее первая команда set\_clock\_latency с опцией -early  устанавливает минимальную задержку при распространении равной . Конструкция [get\_clocks clk\_10MHz] указывает тактовый сигнал, для которого задаются задержки. Вторая команда set\_clock\_latency с опцией -late задает максимальное время распространения. Опция -source означает, что задержка указывается от источника тактового сигнала до ножки FPGA.

Так как задержки распространения до микросхемы 74HC595 отличаются от задержек до FPGA, в файле с временными ограничениями требуется создать еще один тактовый сигнал:

# ограничение на период виртуального тактового сигнала, поступающего

# в микросхему 74HC595

create\_clock -period 100 -name clk\_10MHz\_Dev

# задержки распространения тактового сигнала от генератора до

# микросхемы 74HC595

set\_clock\_latency -source -early $Todd\_min [get\_clocks clk\_10MHz\_Dev]

set\_clock\_latency -source -late $Todd\_max [get\_clocks clk\_10MHz\_Dev]

С помощью команды create\_clock создается тактовый сигнал с именем clk\_10MHz\_Dev и периодом 100 нс. Обратите внимание, что этот сигнал поступает только в микросхему 74HC595, поэтому команда get\_ports, указывающая ножку FPGA, не используется. Тактовый сигнал, который не попадает в FPGA, но присутствует на плате, называется виртуальным (virtual clock). Для виртуального сигнала clk\_10MHz\_Dev минимальная и максимальная задержки распространения равны и соответственно.

Так как задержки для тактового сигнала уже заданы, уравнения для и упростятся и будут содержать только задержки распространения данных:

Временные ограничения для сигнала создаются с помощью команды set\_input\_delay в виде:

# временные ограничение для входного сигнала IDATA

set idelay\_max [expr $Tdco\_max + $Tbd\_max]

set\_input\_delay -clock clk\_10MHz\_Dev -max $idelay\_max [get\_ports IDATA];

set idelay\_min [expr $Tdco\_min + $Tbd\_min]

set\_input\_delay -clock clk\_10MHz\_Dev -min $idelay\_min [get\_ports IDATA];

Опция -max указывает, что задержки предназначены для анализа по *Setup*, а опция -min – для анализа по *Hold*. Обратите внимание, что теперь запускающий триггер, расположенный внутри микросхемы 74HC595, тактируется виртуальным сигналом, поэтому опция –clock задана с сигналом clk\_10MHz\_Dev. Полное содержимое xdc-файла представлено ниже:

# задержка распространения между приходом тактового сигнала и появлением

# данных на выходе QH' микросхемы 74HC595

set Tdco\_min 17

set Tdco\_max 32

# минимальное и максимальное время распространения данных по

# дорожкам платы

set Tbd\_max 0.6

set Tbd\_min 0.5

# минимальное и максимальное время распространения тактового сигнала от

# генератора до микросхемы 74HC595

set Todd\_max 0.4

set Todd\_min 0.2

# минимальное и максимальное время распространения тактового сигнала от

# генератора до FPGA

set Tofd\_max 0.3

set Tofd\_min 0.2

# ограничение на период тактового сигнала, поступающего в FPGA

create\_clock -period 100 -name clk\_10MHz [get\_ports CLK]

# задержки распространения тактового сигнала от генератора до FPGA

set\_clock\_latency -source -early $Tofd\_min [get\_clocks clk\_10MHz]

set\_clock\_latency -source -late $Tofd\_max [get\_clocks clk\_10MHz]

# ограничение на период виртуального тактового сигнала, поступающего

# в микросхему 74HC595

create\_clock -period 100 -name clk\_10MHz\_Dev

# задержки распространения тактового сигнала от генератора до

# микросхемы 74HC595

set\_clock\_latency -source -early $Todd\_min [get\_clocks clk\_10MHz\_Dev]

set\_clock\_latency -source -late $Todd\_max [get\_clocks clk\_10MHz\_Dev]

# временные ограничение для входного сигнала IDATA

set idelay\_max [expr $Tdco\_max + $Tbd\_max]

set\_input\_delay -clock clk\_10MHz\_Dev -max $idelay\_max [get\_ports IDATA];

set idelay\_min [expr $Tdco\_min + $Tbd\_min]

set\_input\_delay -clock clk\_10MHz\_Dev -min $idelay\_min [get\_ports IDATA];

Рассмотрим, что изменилось во временных отчетах при использовании данного способа создания ограничений. На рисунке 9 представлены общие сведения о входном пути для анализа по *Setup*. В третьей строке указано, что теперь запускающий триггер, формирующий входной сигнал, тактируется виртуальным сигналом clk\_100MHz\_Dev.

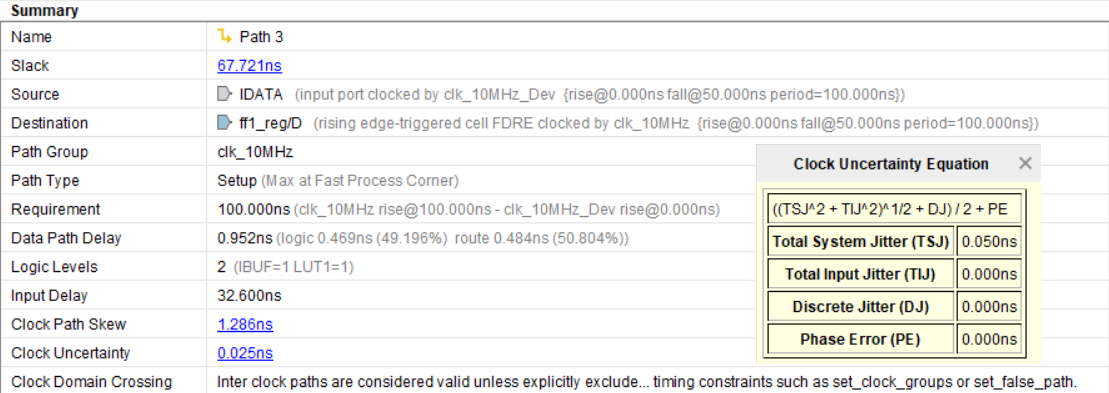
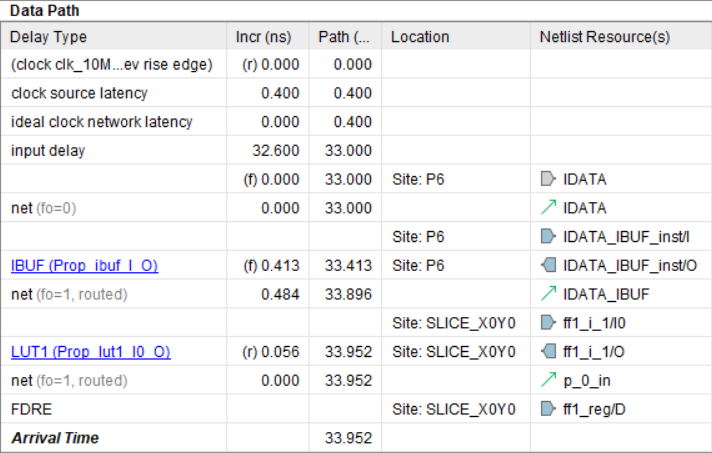
**

Рисунок 9. Общие сведения о входном пути (*Setup*).

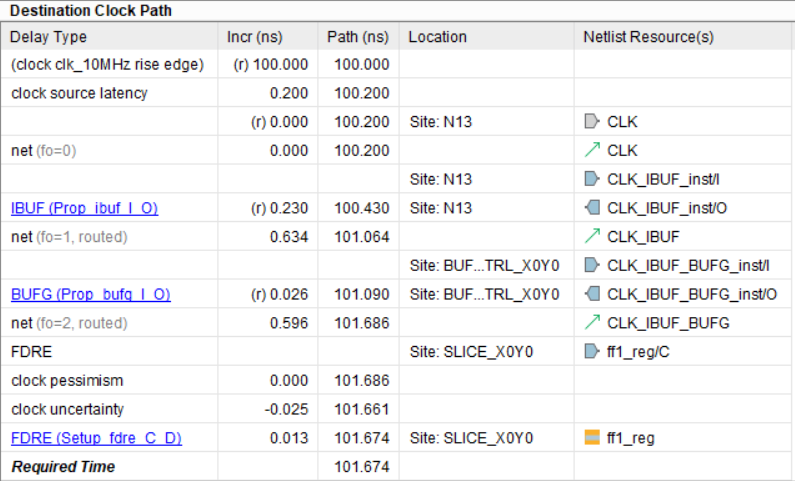
Расчет фактического времени прибытия данных показан на рисунке 10. Можно увидеть, что запускающий фронт появляется на выходе генератора в нулевой момент времени. Спустя 0.4 нс (clock source latency) тактовый сигнал поступает на микросхему 74HC595, что соответствует задержке .

**

Рисунках 10. Расчет фактического времени прибытия данных (*Setup*).

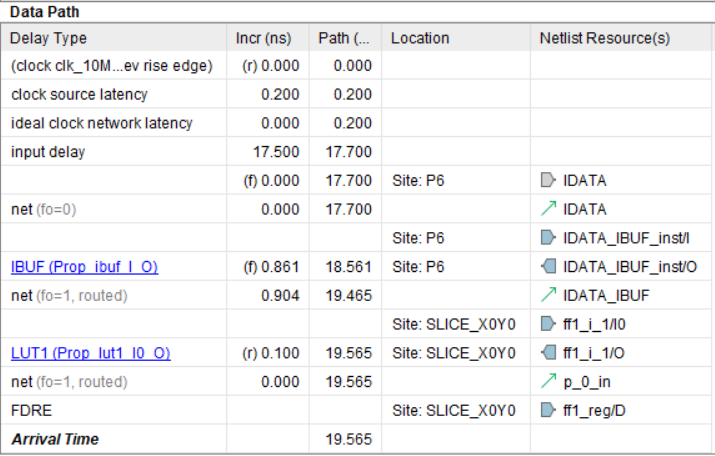
Еще через нс данные достигнут ножки FPGA. Это произойдет в момент времени нс. С учетом задержек распространения внутри FPGA фактическое время прибытия данных составит 33.952 нс.

Из рисунка 11 можно увидеть, защелкивающий фронт тактового сигнала появляется на выходе генератора спустя один период в 100 нс. Через нс этот фронт дойдет до FPGA, пройдет через входной и тактовый буферы и попадет на защелкивающий триггер. Требуемое время прибытия данных составляет 101.674 нс, а величина *Slack* для анализа по *Setup* равна нс.

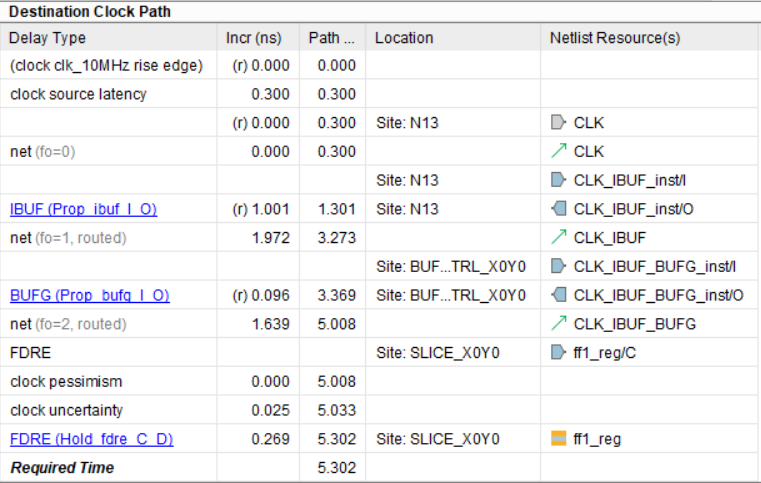
**

Рисунках 11. Расчет требуемого времени прибытия данных (*Setup*).

Для анализа по *Hold* на рисунке 12 можно увидеть, что запускающий фронт формируется на выходе генератора в нулевой момент времени. Через нс, он появится на входе SCLK микросхемы 75HC595, а еще спустя нс данные достигнут FPGA. В свои очередь защелкивающий фронт появится на выходе генератора также в нулевой момент времени и дойдет до ножки FPGA через нс, что показано в расчетах требуемого времени прибытия фронта на рисунке 13.

**

Рисунках 12. Расчет фактического времени прибытия данных (*Hold*).

**

Рисунках 13. Расчет требуемого времени прибытия данных (*Hold*).

Использование виртуальных тактовых сигналов для создания ограничений на входные сигналы лучше отражает физическое распространение сигналов в плате и упрощает интерпретацию временных отчетов. Более того, данный способ рекомендуется Vivado. Если открыть вкладку XDC в Language Templates, то можно увидеть пример временных ограничений для входного сигнала:

# Rising Edge System Synchronous Inputs

#

# A Single Data Rate (SDR) System Synchronous interface is

# an interface where the external device and the FPGA use

# the same clock, and a new data is captured one clock cycle

# after being launched

#

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

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

#           |

#           |------> (tco\_min+trce\_dly\_min)

#           |------------> (tco\_max+trce\_dly\_max)

#         \_\_\_\_\_\_\_\_\_\_      \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# data    \_\_\_\_\_\_\_\_\_\_XXXXXX\_\_\_\_\_ Data \_\_\_\_\_XXXXXXX

#

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

set tco\_max         0.000;         # Maximum clock to out delay

set tco\_min         0.000;         # Minimum clock to out delay

set trce\_dly\_max    0.000;         # Maximum board trace delay

set trce\_dly\_min    0.000;         # Minimum board trace delay

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

# Input Delay Constraint

set\_input\_delay -clock $input\_clock -max [expr $tco\_max + $trce\_dly\_max]

[get\_ports $input\_ports];

set\_input\_delay -clock $input\_clock -min [expr $tco\_min + $trce\_dly\_min]

[get\_ports $input\_ports];

В данном случае значение задается в виде суммы задержек tco () и trce\_dly (), что соответствует уравнениям (5). Заметим также, что если пренебречь задержками тактового сигнала и , то оба представленных способа будет полностью эквивалентны.

1. **Несовпадающие значения *Slack*.**

В заключении рассмотрим следующий интересны момент. Можно увидеть, что значения *Slack*, представленные на рисунках 4 и 9, отличаются и равны 67.711 нс и 67.721 нс соответственно. Выясним из-за чего возникает разница в 0.01 нс. Рассмотрев временные отчеты, обнаружим, что причиной неодинаковых значений является неопределенность тактового сигнала (clock uncertainty).

Неопределенность тактового сигнала может возникать из-за джиттера. Значение общего джиттера *Ttsj* (***T****otal* ***S****ystem* ***J****itter*) рассчитывается по формуле [5]:

где (***S****ystem* ***J****itter* ***S****ource*) – джиттер для запускающего триггера; (***S****ystem* ***J****itter* ***D****istention*) – джиттер для защелкивающего триггера.

По умолчанию для тактовых сигналов, которые поступают в FPGA, значение принимается равным 0.05 нс. Тогда для первого способа создания временных ограничений получаем, что нс. Неопределенность тактового сигнала равна нс, что можно увидеть на рисунке 4.

Если используется второй способ, то один из тактовых сигналов является виртуальным, и по умолчанию его джиттер считается равным нулю. То есть в этом случае поучаем, что нс и нс Как показано на рисунке 9 неопределенность тактового сигнала равна нс.

Чтобы значения *Slack* для обоих способов были равны можно настроить джиттер, воспользовавшись командами set\_system\_jitter и set\_input\_jitter.

**Заключение.**

В данной статье был рассмотрен временной анализ для входных сигналов FPGA. Показан вывод уравнений статического временного анализа. Представлено два способа создания временных ограничений. Рассказано о виртуальных тактовых сигналах. В следующей статье предполагается использовать тот же подход для рассмотрения временных ограничений для выходных сигналов System Synchronous Output Delay.

**Ссылки.**

1. <https://www.xilinx.com/support/documentation/data_sheets/ds181_Artix_7_Data_Sheet.pdf>
2. <https://www.xilinx.com/support/documentation/sw_manuals/xilinx2020_2/ug903-vivado-using-constraints.pdf>
3. <https://www.xilinx.com/support/documentation/sw_manuals/xilinx2020_2/ug906-vivado-design-analysis.pdf>
4. <https://www.xilinx.com/support/documentation/sw_manuals/xilinx2020_2/ug906-vivado-design-analysis.pdf>
5. <https://www.xilinx.com/support/documentation/sw_manuals/xilinx2020_2/ug906-vivado-design-analysis.pdf>