**Основы статического временного анализа: period constraint.**

**Введение.**

Что будет в статье.

**Статический временной анализ.**

Во всех гайдах с рекомендациями написано избегать защелок (latch), поэтому правильно FPGA в который загружен правильно сделанный проект представляет из себя синхронное последовательное цифровое устройство. То есть схема проекта состоит из набора регистров разделенных комбинационной логики. В регистрах содержится состояние системы, изменяющееся только по фронту (спаду) тактового сигнала. Типичный путь сигнала внутри FPGA представлен на рис. 1.

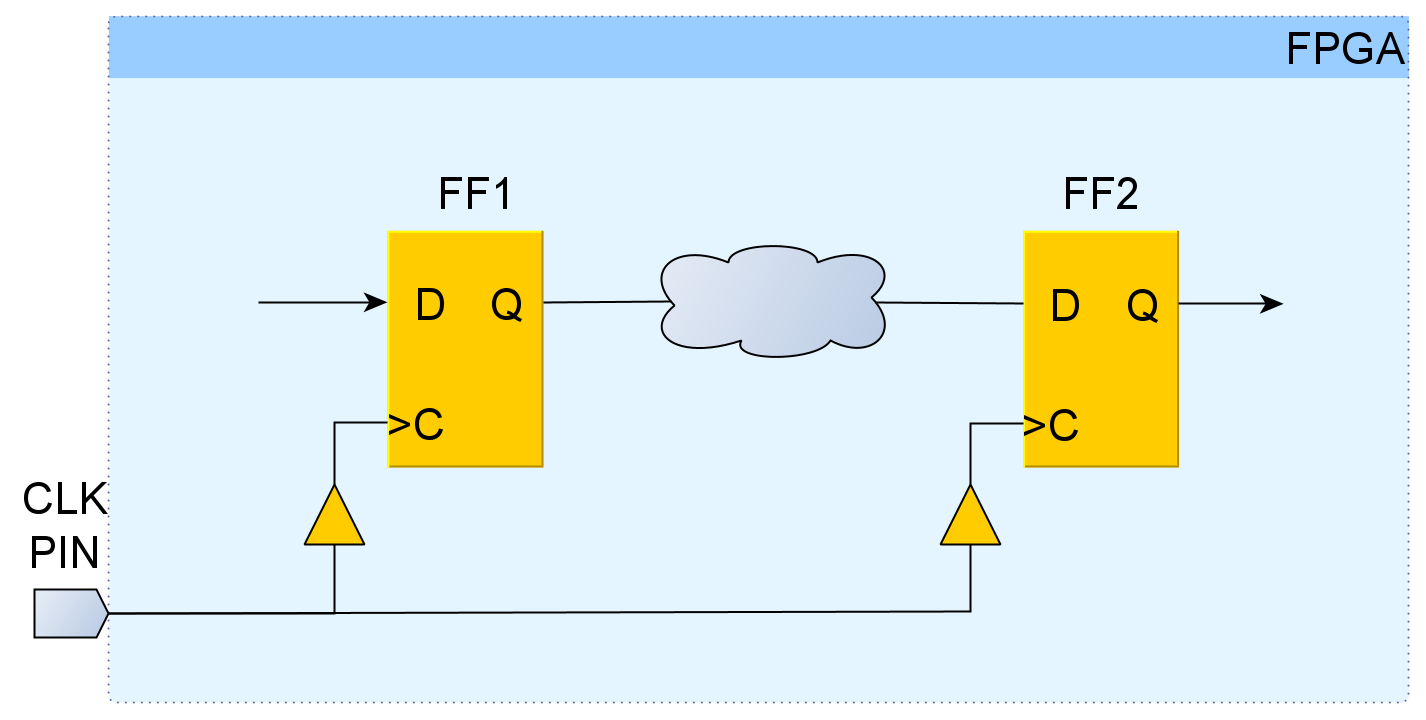


Рисунок 1. Типичный путь сигнала внутри FPGA.

По фронту тактового сигнала, который называют запускающий (start, source, launch), данные с D входа триггера FF1 переходят на выход Q, распространяются через комбинационную логику и попадают на D вход триггера FF2. Спустя период тактового сигнала приходит следующий фронт тактового сигнала, называемый защелкивающий (catch, destination, latch), триггер FF2 защелкивает данные на входе D и передает их на выход Q и далее по схеме.

Чтобы данные корректно распространялись описанным выше образом от триггера к триггеру при заданной частоте тактового сигнала задержки должны быть выполнены два условия:

* данные должны распространяться достаточно быстро, чтобы успеть дойти до триггера до прихода защелкивающего фронта (максимальное время распространения);
* данные должны распространяться достаточно медленно, чтобы предыдущие данные успели быть получены защелкивающим фронтом (минимальное время распространения).

Задача статический временного анализа заключается в том, чтобы для каждого пути между двумя триггерам рассчитать задержки распространения данных и тактового сигнала и установить удовлетворяются ли два приведенных выше ограничения.

Рассмотрим каким образом временной анализатор проводит расчет задержек. На рис. 2 представлен путь сигнала, на который нанесены задержки для данных и тактового сигнала.

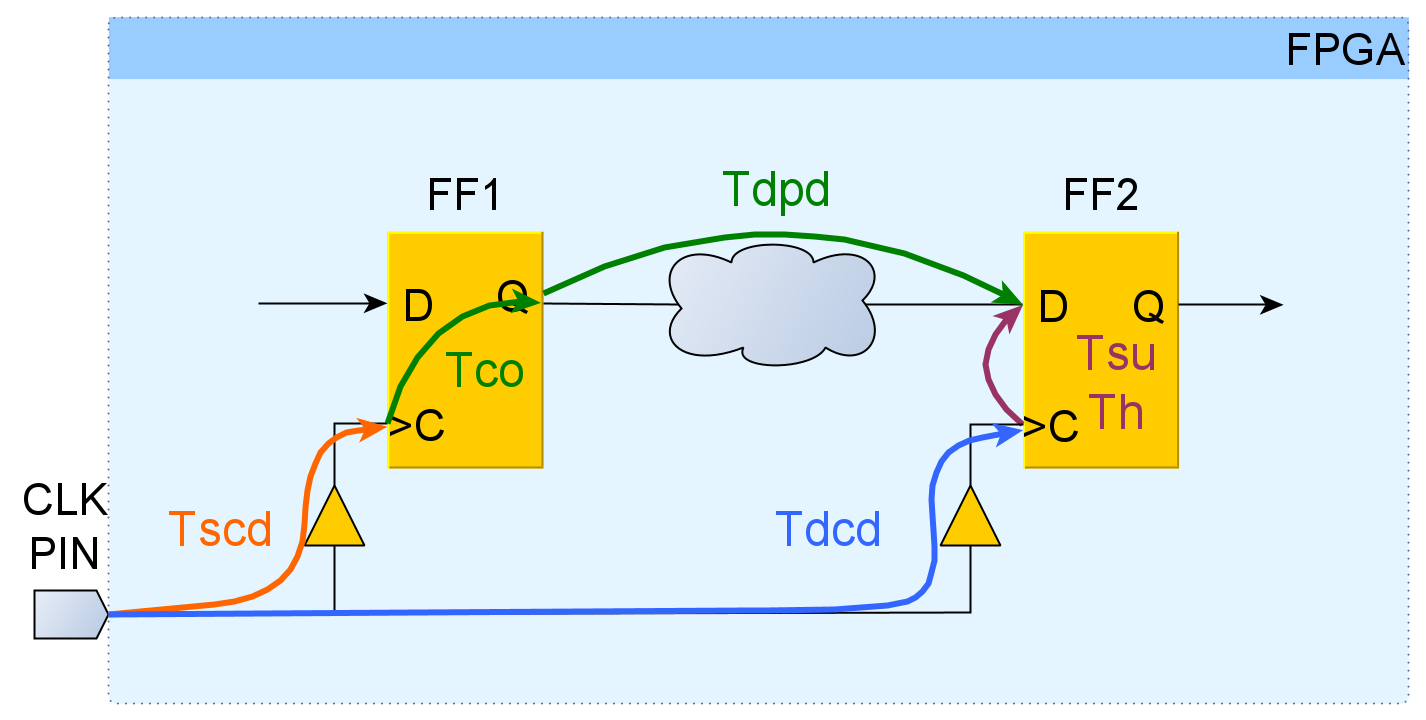


Рисунок 2. Задержки при распространении данных и тактового сигнала.

Ниже представлены определения задержек, представленных на рис.2.

* (source clock delay) – задержка тактового сигнала от тактового пина до тактового входа триггера FF1;
* (destination clock delay) – задержка тактового сигнала от тактового пина до тактового входа триггера FF2;
* (clock to output) – задержка от прихода фронта на тактовый вход триггера до появления данных c входа D на выходе Q;
* (data propagation delay) – задержка распространения данных между триггерами по соединениям и через логику;
* (setup time) – время установления, то есть промежуток времени до фронта тактового сигнала, в течении которого данные должны быть стабильны на входе D, чтобы надежно защелкнуться;
* (hold time) – время удержания, то есть промежуток времени после фронта тактового сигнала, в течении которого данные должны быть стабильны на входе D, чтобы надежно защелкнуться.

Обычно все события отсчитываются от некоторого нулевого момента времени. В качестве нулевого момента возьмем запускающий фронт. Период тактового сигнала обозначим как .

**Максимальное время распространения**

Для начала рассмотрим ограничение на максимальное время распространения:

Временной анализ проводится всегда для самого пессимистичного случая. Этому случаю соответствует максимально задержаний запускающий фронт и данные и максимально быстро пришедший защелкивающий фронт.

* Время прибытия запускающего фронта (Source clock arrive time):
* Задержка распространения данных (data delay):
* Время прибытия данных на вход защелкивающего триггера (data arrive time):
* Время прибытия защелкивающего фронта (Destination clock arrive time):

Защелкивающий является следующим после запускающего, поэтому к задержке распространения добавлен один период тактового сигнала.

* Требуемое время прибытия данных (data require time):

Так как данные должны быть стабильны в течении времени установки до защелкивающего фронта, данные должны прийти на время раньше защелкивающего фронта.

Чтобы было удовлетворено требование на максимальное время распространения, данные должны попасть на D вход защелкивающего триггера не позже, чем требуется в соответствии с .

Статический анализатор рассчитывает запас () в задержке данных

Если положительный, то это значит, что данные приходят раньше, чем требуется. Отрицательный указывает на нарушение ограничения на максимальное время распространения.

Подставив рассчитанные значения задержек получим еще ряд выражений для

Величина равная разности времени распространения тактового сигнал до запускающего и защелкивающего триггеров называется расфазировкой тактового сигнала (clock skew)

.

Можно увидеть, что положительное значение увеличивает .

**Минимальное время распространения**

Теперь рассмотрим ограничение на минимальное время распространения:

Временной анализ также проводится для самого пессимистичного случая, которому соответствует наиболее быстро распространяющийся запускающий фронт и данные и максимально задержанный защелкивающий фронт.

* Время прибытия запускающего фронта (Source clock arrive time):
* Задержка распространения данных (data delay):
* Время прибытия данных на вход защелкивающего триггера (data arrive time):
* Время прибытия защелкивающего фронта (Destination clock arrive time):

Защелкивающий фронт должен успеть защелкнуть предыдущие данные, то есть он появляется в то же время что и запускающий фронт, что поэтому к задержке распространения период тактового сигнала не добавляется.

* Требуемое время прибытия данных (data require time):

Так как предыдущие данные должны быть стабильны в течении времени удержания после защелкивающего фронта, новые данные должны прийти на время позже защелкивающего фронта.

Чтобы было удовлетворено требование на минимальное время распространения, данные должны попасть на D вход защелкивающего триггера не раньше, чем требуется в соответствии с .

Статический анализатор рассчитывает в задержке данных

Если положительный, то это значит, что данные приходят позже, чем требуется. Отрицательный указывает на нарушение ограничения на минимальное время распространения.

Подставив рассчитанные значения задержек получим еще ряд выражений для

Можно увидеть, что в выражение для расфазировкой тактового сигнала входит с знаком противоположным, тому который был при расчете запаса на максимальное время распространения. То есть положительное значение уменьшает .

**Задание ограничений и разбор отчетов в Vivado.**

Рассмотрим задание ограничений в Vivado. После синтеза для каждого пути Vivado известно через какие логические элементы он проходит (LUT, MUX, CARRY CHAIN). Типовые задержки указываются в Datasheet для конкретного кристалла. Например для в Datasheet для Artix 7 (ds 181) в таблице 27 указано время распространения через LUT (input, logic, output), а также время установления и удержания для триггеров и . В таблицах 32 – 35 указываются задержки распространения тактового сигнала через различные виды буферов.

После размещения и разводки проекта Vivado также получает информацию о задержках сигналов при распространении через линии соединения. После имплементации Vivado знает значения все переменных, которые входят в уравнения (1) и (2), кроме периода тактового сигнала . В нашем случае тактовый сигнал поступает от внешнего генератора, частоту которого Vivado, естественно, не знает.

Для примера рассмотрим самый простой пример. Пусть проект состоит из двух триггеров между которыми расположен LUT выполняющий логическое отрицание. Пример описания на System Verilog

module top (

    input  logic clk\_pin,

    input  logic in1,

    output logic out1

);

    logic ff1, ff2;

    always\_ff @(posedge clk\_pin)

        ff1 <= in1;

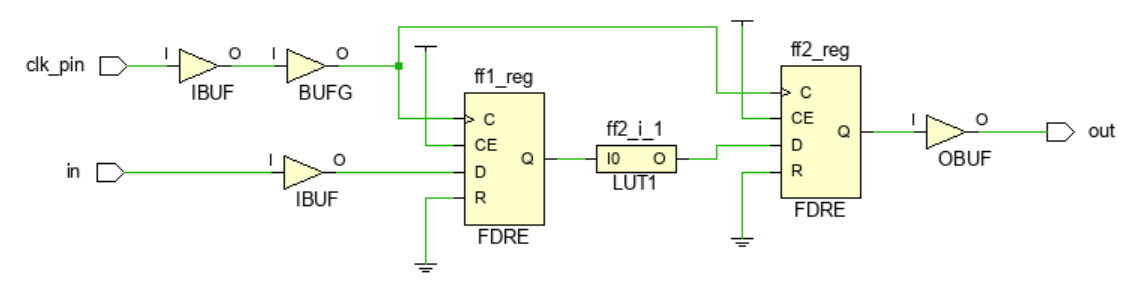
    always\_ff @(posedge clk\_pin)

        ff2 <= ~ff1;

    assign out1 = ff2;

endmodule

Схема проекта, полученная после имплементации представлена рисунке 3.

Рисунок 3. Схема проекта

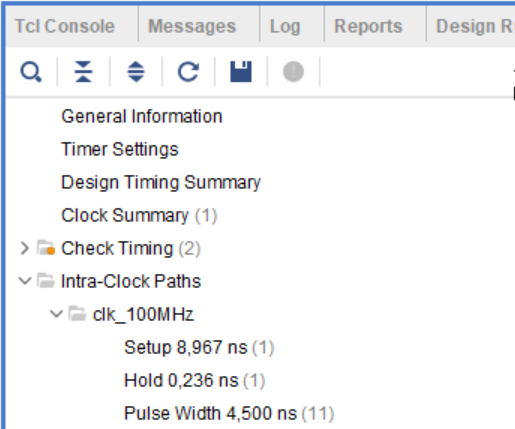
Чтобы Vivado мог провести временной анализ и проверить проект на удовлетворение максимальное и минимальное время распространения ему необходимо указать период тактового сигнала. Это делается в файле ограничений формата XDC. Будем считать, что частота тактового сигнала равна 100 МГц.

В xdc-файле необходимо указать следующую команду

create\_clock -period 10.000 -name clk\_100MHz [get\_ports clk\_pin]

Опция –period указывает период тактового сигнала в наносекундах. Конструкция [get\_ports clk\_pin] возвращает порт проекта с именем clk\_pin, указывает откуда приходит тактовый сигнал. С помощью опции –name можно задать имя тактового сигнала. Если этого не сделать имя тактового сигнала будет совпадать с именем порта.

После имаплементации открыть Timing Summury, во вкладке Intra-Clock Path clk\_100MHz будет представлены некоторые пути, для которых проводился временной анализ.



Раздел Setup соответствует проверке на максимальное время распространения, а раздел Hold на минимальное время распространения.

**Анализ ограничения на максимальное время распространения.**

Для начала рассмотри отчет максимальное время распространения. В разделе Setup дважды нажав на показанный путь можно открыть расширенный отчет path report. Данный отчет состоит из четырех разделов. Рассмотрим их по порядку.

Первый раздел представлен на рисунке 1. В данном разделе представлены общие сведения, такие как имя пути, рассчитанный для данного пути , имя и период тактового сигнала. Началом пути является тактовый вход триггера ff1 заканчивается путь на D входе триггера ff2. Также указывается задержке при распространении данных, в наших обозначениях и количество уровней логики, в нашем случае один LUT, что и написано. В конце указывается расфазировнка тактового сигнала, которую мы обозначали , и неопределенность для тактового сигнала Clock Uncertainty, об этом параметре более подробно будем говорить далее.

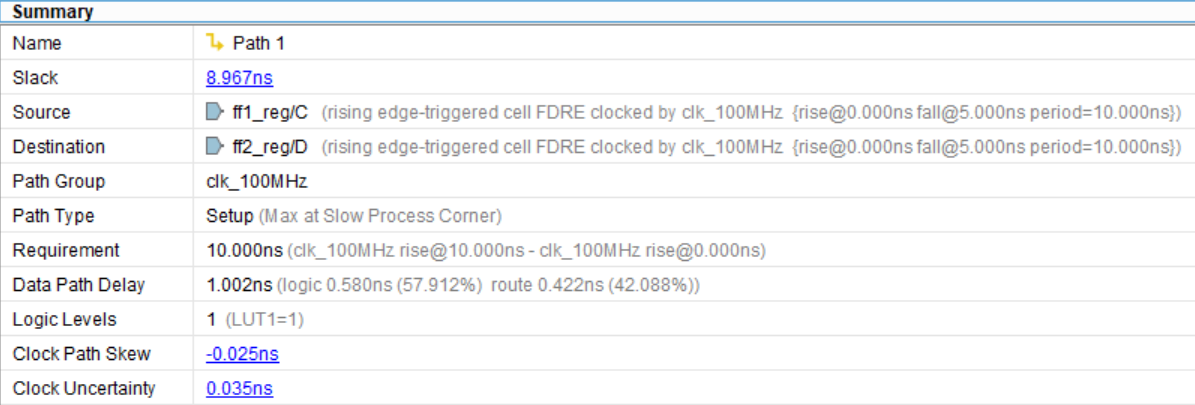


Рисунок 1. Общие сведения об анализируемом пути

Следующий раздел указывает задержки распространения запускающего фронта и представлен на рисунке 2. В данном перечислены все задержки, которые встречает запускающий фронт. В столбце Incr указано значение отдельной задержки, а в столбце Path сумма текущей и всех предыдущих задержек. Можно увидеть, что тактовый сигнал начинается в момент времени ноль, и распространяется через входной и тактовый буферы. Общая задержка распространения, которая обозначалась нами как (1), составляет 4,708 наносекунд. Обозначение (r) рядом со значение задержки указывает, что анализируется фронт тактового сигнала.

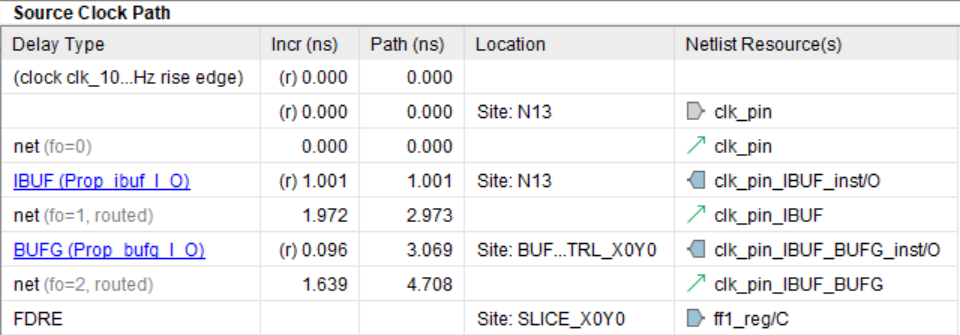


Рисунок 2. Задержка распространения для запускающего фронта

В следующем разделе указаны задержки при распространении данных и представлен на рисунке 3. Задержки данных состоят из задержки clock to output для триггера ff1, задержки распространения через LUT и задержки распространения по линиям соединения. Просуммировав значения в столбце Incr получим 1,002 нс, что совпадает со значением Data Path Delay из раздела Summary. Обратите внимание, что значения в столбце Path не начинаются с нуля, так как учитывается задержка распространения запускающего фронта. Первое значение в столбце Path рассчитывается как сумма задержки clock to output (0,456 нс) и (4,708 нс). Конечное значение в столбце Path показывает время прибытия данных до защелкивающего триггера, в наших обозначениях это (1). О том, что обозначают (r) и (f) рядом со значениями задержек будет рассказано далее.

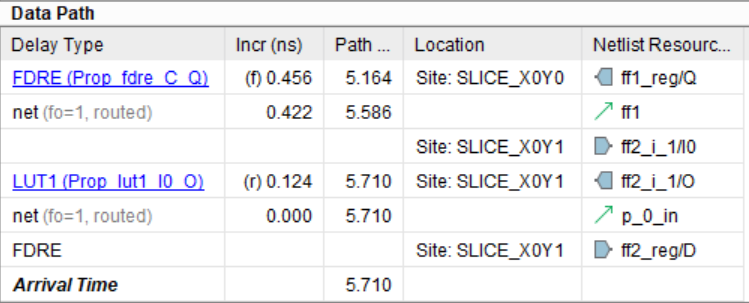


Рисунок 3. Задержки в распространении данных.

В последнем разделе приводится время расчёт требуемого времени прибытия данных. Защелкивающий фронт приходит через период после запускающего фронта, поэтому в первой строке отсчета указано 10 нс. Далее в строках до строки FDRE указываются задержки при распространении защелкивающего фронта. Защелкивающий фронт также проходит через входной и тактовый буферы и попадает на С вход триггера ff2 в момент времени 14,347 нс, который в наших обозначениях и рассчитывается по формуле (1).

Далее представлены еще три задержки представляют из себя время удержания для триггера ff2 (0.029 нс), а также пессимизм (clock pessimism) (0,336 нс) и неопределенность (clock uncertainty) (-0,035 нс) тактового сигнала. О последних двух задержках будет рассказано далее.

Требуемое время прибытия данных равно 14,677 нс и в наших обозначениях (1). Формула (1) учитывающая и будет иметь вид

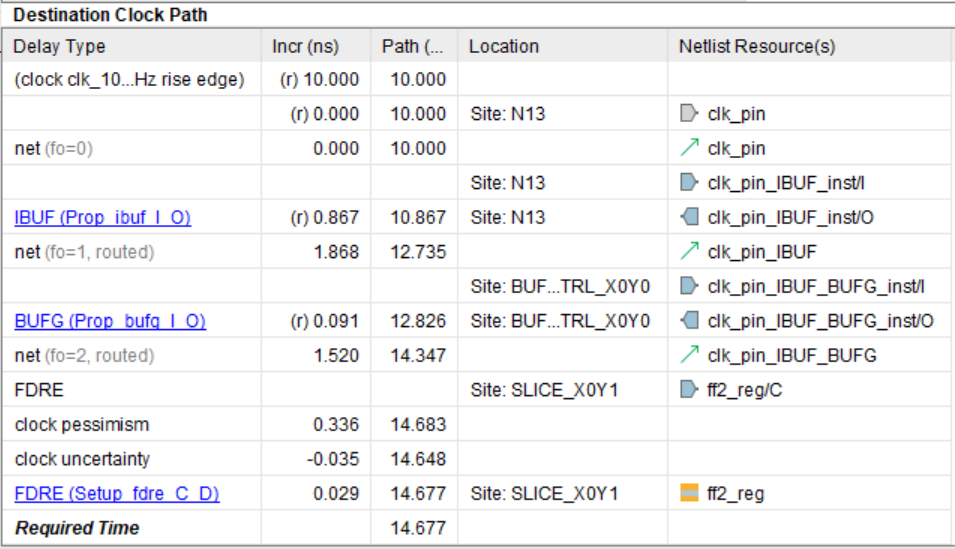


Рисунок 4. Требуемое время прибытия данных.

рассчитывается по формуле (1) и равен

,

что совпадает со значением, представленным в разделе Summary.

**Анализ ограничения на максимальное время распространения.**

Далее рассмотрим отчет ограничение на минимальное время распространения. В разделе Hold дважды нажав на путь можно открыть расширенный отчет path report. Данный отсчет также состоит из четырех разделов, представленных на рисунках 1-3. В разделах данные в разделах Summary, Source Clock Path и Data Path имеют тот же самый смысл, что и при анализе по Setup.

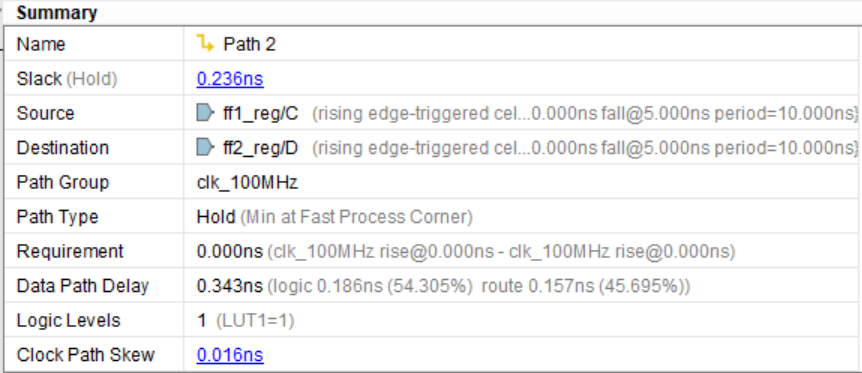


Рисунок 1. Общие сведения о пути

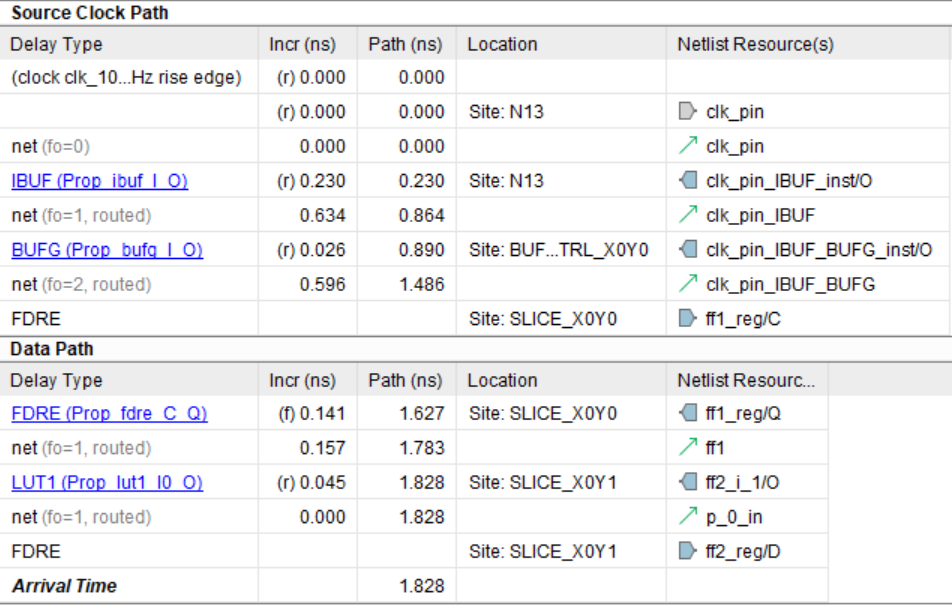


Рисунок 2. Расчет фактического времени прибытия данных

На рисунке 3 представлен расчёт требуемого времени прибытия данных. Обратите внимание, так как при анализе по Hold запускающий и защелкивающий фронты появляются в один и тот же момент времени, начальное значение в столбцах Incr и Path равное 0 нс, а не 10 нс, как для Setup.

Можно увидеть, что в данном случае отсутствует слагаемое (clock uncertainty). Причина будет раскрыта далее.

рассчитывается по формуле (1) и равен

Также стоит заметить, что задержки через одни и те элементы же при анализе по Setup и Hold имеют разные значения. Например, задержка через LUT при Setup равна 0.124 нс, а при Hold – 0.045 нс. Значение, представленное в Datasheet DS181 для -1 speedgrade, равно 0.1 нс. Как было сказано ранее, статический анализ производится для самого пессимистичного случая, поэтому при анализе по Setup рассматривается случай, когда данные будет распространятся максимально долго, что соответствует (data propagation delay) в формуле (1). Для анализа по Hold самый худший случай – это максимально быстрое распространение данных, в формуле (1).

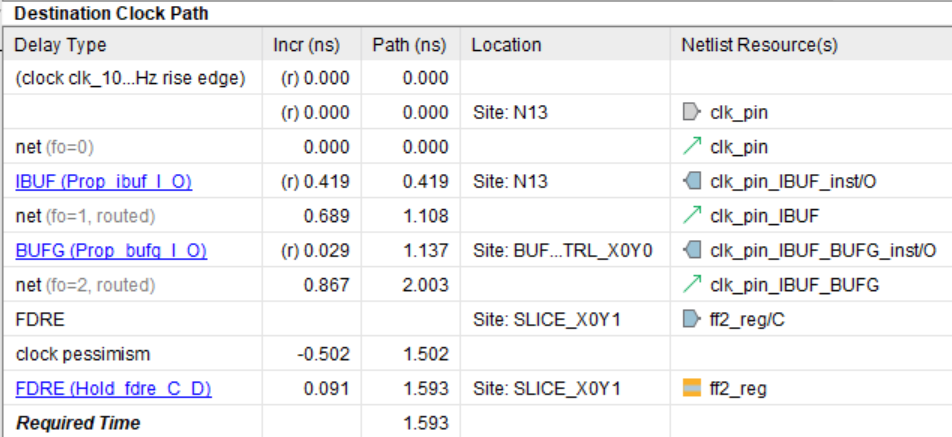


Рисунок 3. Расчет требуемого времени прибытия данных

В заключении рассмотри, что же обозначают записи (r) и (f) рядом с задержками данных. Глядя на рисунок 1, можно подумать, что при временном анализе Vivado рассматривает всего один путь, однако это не так. У каждого пути есть начало и конец. Точка, в которой заканчивается путь называется Endpoint. Несколько разных путей могут иметь один и тот же Endpoint. По умолчанию Vivado во временном отчете отображает всего один худший путь для каждого Endpoint. Это можно изменить, если при формировании отчета установить параметр Maximum number of worst path per endpoint. Задав этот параметр, например, равны 10, на рисунке 1 можно увидеть, что теперь в отчете отображается 4 пути.

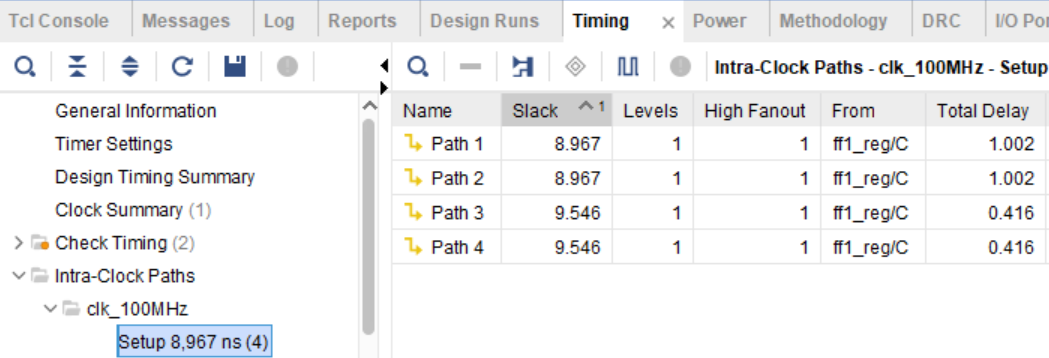


Рисунок 1. Анализируемые пути.

Задержки распространения чрез элемент цифровой схемы могут разные значения, если выход элемента изменяет с низкого уровня на высокой или с высокого на низкий. Обозначение (f) при расчете задержки данных указывает, что в данном случае анализируется вариант, когда сигнал на выходе элемента изменяется с высокого на низкий. На рисунке (1) можно увидеть, что рассматривается случай, когда сигнал на выходе триггера ff1 изменяется с ‘1’ на ‘0’, а сигнал на выходе LUT – c ‘0’ на ‘1’ (r). Всего есть четыре возможные комбинации (r) и (f) для LUT и ff1, поэтому отчете переставлено четыре пути. Очевидно, что так как в нашем случае LUT реализует инвертор, то ситуация ff1 (f) и LUT (f), а также ff1 (r) и LUT (r) невозможны. Однако временной анализатор ничего не рассматривает как именно работает схема, а просто считает все возможные задержки для самых пессимистичных случаев.

**Неопределенность тактового сигнала.**

В идеале тактовый сигнал представляет из себя последовательность прямоугольных импульсов одинаковой формы, повторяющихся с постоянным заданными периодом. У реального тактового сигнала фронта импульсов в среднем повторяются через период, однако каждый отдельный фронт может прийти четь раньше или чуть позже. Это явление называется дрожанием тактового сигнала или джиттером. Если выражать в виде формул, то время между двумя фронтами тактового сигнала можно записать в виде

где – как и раньше период тактового сигнала, – случайная величина с нулевым средним, обуславливающая неопределенность тактового сигнала.

Слагаемое (clock uncertainty), которое встречалось при рассмотрении отчета о временном анализе, как раз и отвечает за неопределенность тактового сигнала из-за джиттера. Если нажать значение неопределенности в отчете можно увидеть каким образом она рассчитается на рисунке 1.

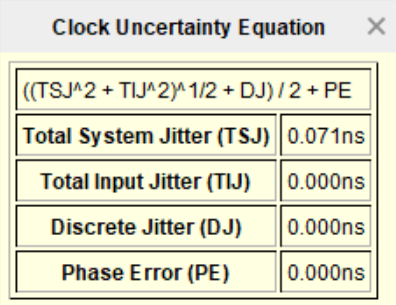


Рисунок 1. Выражение для расчета неопределенности

Данное выражение имеет вид:

где – общий джиттер присутствующий тактовом входе каждого последовательного элемента; – дополнительный джиттер для конкретного тактового сигнала; – джиттер тактового сигнала, формируемого блока PLL или MMCM; – постоянная расфазировка тактового сигнала, формируемого блока PLL или MMCM, относительного опорного.

Параметры и рассчитываются Vivado самостоятельно и отличны от нуля если данные запускаются или защёлкиваются тактовым сигналом, который формируется в PLL или MMCM.

По умолчанию Vivado задает значение равным 0,071 нс. Это значение можно изменить, указать следующую команду в xdc-файле

set\_system\_jitter Tsj

где (system jitter) задается в нс. Значение общего джиттера равно

.

Например, результат команды

set\_system\_jitter 0.15

можно увидеть на рисунке 1.

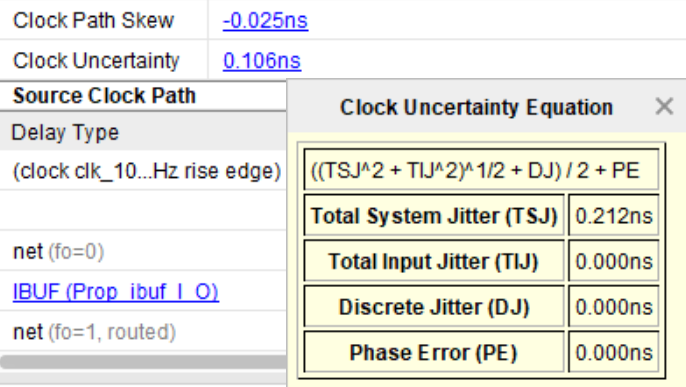


Рисунок 1. Выражение для расчета неопределенности

Как и ожидалось нc, а, так как остальные слагаемые в формуле (1) равны нулю, нс.

Если требуется можно задать дополнительную неопределенность из-за джиттера для конкретного тактового сигнала с помощью команды вида

set\_input\_jitter [get\_clocks clock\_name] Tij

где (input jitter) задается в нс. Значение общее значение джиттера для тактового сигнала равно , если запускающий и защёлкивающий фронта относятся к одному тактовому сигналу, и – если запускающий и защелкивающий тактовые сигналы разные.

Например, результат соделжимого xdc файла

create\_clock -period 10.000 -name clk\_100MHz [get\_ports clk\_pin]

set\_system\_jitter 0.15

set\_input\_jitter [get\_clocks clk\_100MHz] 0.100

можно увидеть на рисунке 1.

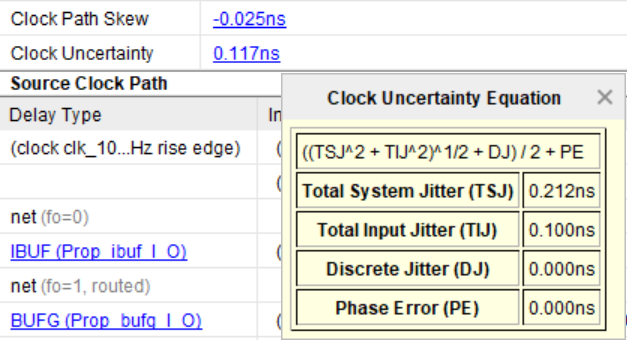


Рисунок 1. Выражение для расчета неопределенности

Как и ожидалось

Обобщая все выше сказанное: из-за наличия джиттера временной интервал между двумя последовательными фронтами тактового сигнала отличается от периода на некоторую случайную величину. При проведении временного анализа это случайную величину можно задавать с помощью команд set\_system\_jitter и set\_input\_jitter.

Теперь можно объяснить почему при анализе по Hold отсутствовало слагаемое . При анализе по Hold запускающий и защелкивающий импульсы появляются в один и тот же момент времени, поэтому между их фронтами нет никакой неопределенности. Если бы запускающий и защелкивающий фронты относились к разным тактовым сигналам, то слагаемое присутствовало в анализе по Hold.

**Устранение пессимизма задержки тактового сигнала.**

set\_system\_jitter, set\_input\_jitter, CPPR

**Заключение.**

Что было в статье. Что будет дальше.