Основы статического временного анализа.

Часть 2.1: System Synchronous Input Delay Constraint.

Введение.

Данная статья является продолжением серии статей по временным ограничениям в FPGA. Главная цель – познакомить начинающих разработчиков с основами статического временного анализа. Далее будет рассмотрен анализ передачи данных в FPGA из внешнего устройства и показаны два способа создания ограничений для входных сигналов.

1. Цель временных ограничений для входных сигналов.

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 триггера, *Slack* и т.д.

Обычно обрабатываемые данные пересылаются между FPGA и другими микросхемами, расположенными на той же печатной плате. Эти пути передачи 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 обычный регистр сдвига и часто поставляется в составе обучающих наборов для Arduino. Также на плате располагается генератор (OSC), формирующий тактовый сигнал 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 данных также формирует тактовый сигнал, 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 иногда называть Device. Далее будет представлен временной анализ для передачи данных с выхода 74HC595 на вход FPGA. На рисунке 1 данный путь отмечен красным цветом.

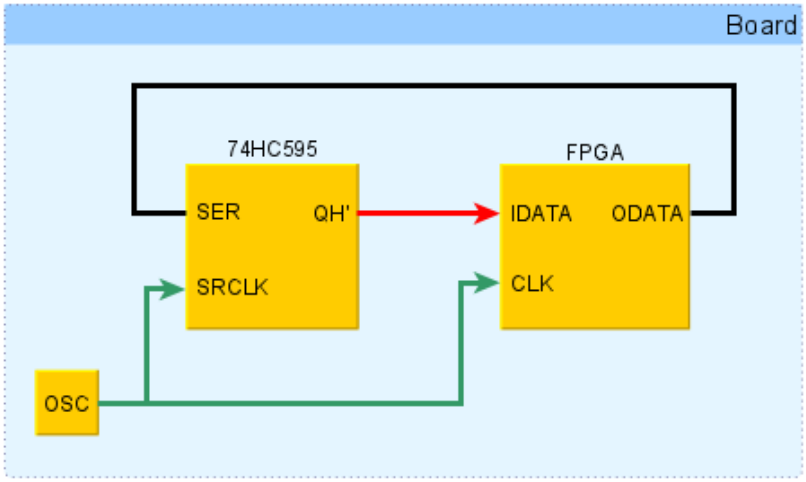

Рисунок 1. Схема соединения устройств на плате.

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

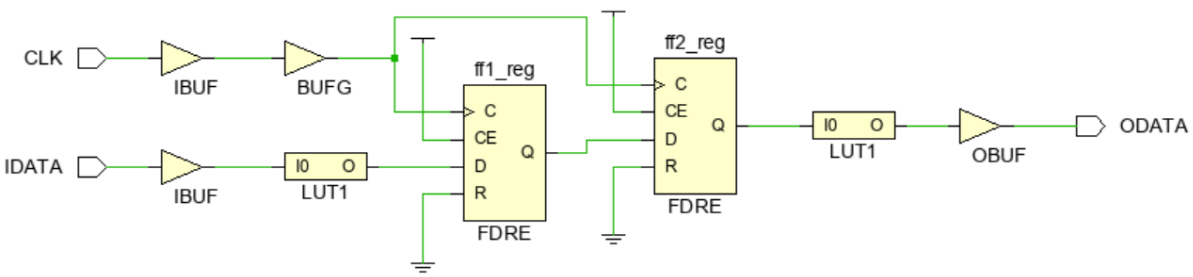

Рисунок 2. Схема FPGA проекта.

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.

2. 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.

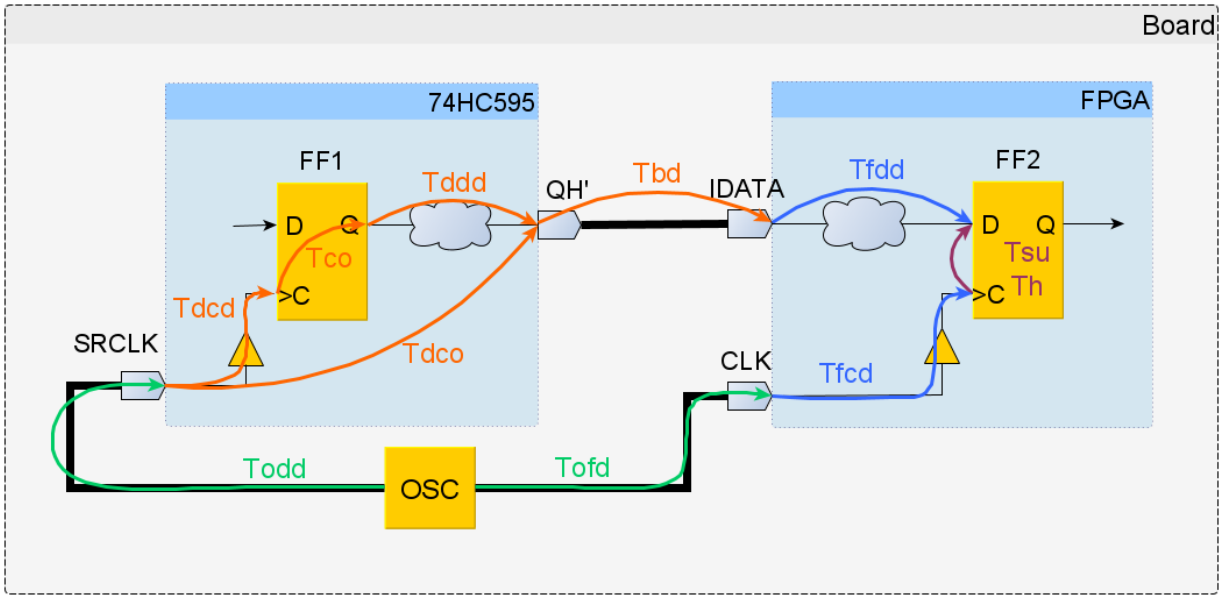

Рисунок 3. Путь с задержками для входных данных и тактового сигнала.

Ниже даны определения задержек, представленных на рисунке 3.

- *Todd* (***Osc to Device Delay***) – задержка тактового сигнала от генератора до ножки SRCLK микросхемы 74HC595;
- *Tofd* (***Osc to FPGA Delay***) – задержка тактового сигнала от генератора до ножки CLK FPGA;
- *Tdcd* (***Device Clock Delay***) – задержка тактового сигнала от ножки SRCLK микросхемы 74HC595 до тактового входа запускающего триггера;
- *Tco* (***Clock to Output***) – интервал времени между приходом фронта на тактовый вход триггера и появлением данных на его выходе Q;
- *Tddd* (***Device Data Delay***) – задержка распространения данных от запускающего триггера до ножки QH' микросхемы 74HC595;
- *Tbd* (***Board Delay***) – задержка распространения данных по дорожкам платы между микросхемой 74HC595 и FPGA;
- *Tfcd* (***FPGA Clock Delay***) – задержка тактового сигнала от ножки CLK FPGA до тактового входа защелкивающего триггера;
- *Tfdd* (***FPGA Data Delay***) – задержка распространения данных от ножки IDATA FPGA до защелкивающего триггера;
- *Tsu* (***SetUp time***) – время установки триггера;
- *Th* (***Hold time***) – время удержания триггера.

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

3. 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 защелкивающего триггера, расположенного внутри FPGA, считая, что запускающий фронт появляется в нулевой момент времени. Уравнения для расчета представлены ниже (см. рисунок 3):

- Время прибытия фронта к запускающему триггеру внутри Device (***Source Clock Arrival time***):

$$Tsca\_max = Todd\_max + Tdcd\_max.$$

- Задержка распространения данных (***Data Delay***):

$$Tdd\_max = Tco\_max + Tddd\_max + Tbd\_max + Tfdd\_max.$$

- Время прибытия данных на вход защелкивающего триггера внутри FPGA (***Data Arrival time***):

$$Tda\_max = Tsca\_max + Tdd\_max.$$

Подставив предыдущие результаты в уравнение для *Tda_max*, получим:

$$Tda\_max = Todd\_max + Tdcd\_max + Tco\_max + Tddd\_max +$$
$$+ Tddd\_max + Tbd\_max + Tfdd\_max.$$

Введем обозначение:

$$Tdco = Tdcd + Tco + Tddd.$$

Задержка $Tdco$ соответствует интервалу времени между приходом фронта на тактовый вход SRCLK микросхемы 75HC595 и появлением данных на ее выходе QH'. Уравнение для ***Data Arrival time*** можно записать в виде:

$$Tda\_max = Todd\_max + Tdco\_max + Tbd\_max + Tfdd\_max.$$

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

- Время прибытия фронта к защелкивающему триггеру внутри FPGA (***Destination Clock Arrival time***):

$$Tdca\_min = Tofd\_min + Tfcd\_min + Tclk.$$

- Требуемое время прибытия данных (***Data Required time***):

$$Tdr\_min = Tdca\_min - Tsu = Tofd\_min + Tfcd\_min + Tclk - Tsu.$$

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

При анализе по *Setup* величина запаса (*Slack*) вычисляется по формуле:

$$Slack = Tdr\_min - Tda\_max.$$

Если *Slack* принимает отрицательное 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 записать полное выражение для расчета *Slack*:

$$Slack = Tclk + Tfcd\_min - Tfdd\_max - Tsu +$$
$$+Tofd\_min - Todd\_max - Tdco\_max - Tbd\_max. \quad (1)$$

4. 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 (***Source Clock Arrival time***):

$$Tsca\_min = Todd\_min + Tdcd\_min.$$

- Задержка распространения данных (***Data Delay***):

$$Tdd\_min = Tco\_min + Tddd\_min + Tbd\_min + Tfdd\_min.$$

- Время прибытия данных к триггеру внутри FPGA (***Data Arrival time***):

$$Tda\_min = Tsca\_min + Tdd\_min =$$
$$= Todd\_min + Tdco\_min + Tbd\_min + Tfdd\_min,$$

где, как и ранее,

$$Tdco = Tdcd + Tco + Tddd.$$

Далее представлены уравнения для вычисления требуемого времени прибытия данных:

- Время прибытия фронта к защелкивающему триггеру внутри FPGA (***D**estination **C**lock **A**rrival time*):

$$Tdca\_max = Tofd\_max + Tfcd\_max.$$

- Требуемое время прибытия данных (***D**ata **R**equired time*):

$$Tdr\_max = Tdca\_max + Th = Tofd\_max + Tfcd\_max + Th.$$

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 *Tdca* отсутствует слагаемое, равное периоду тактового сигнала. Также отметим, что слагаемое Th в уравнении для *Tdr_max* учитывает, что данные не должны изменяться в течении времени удержания после защелкивающего фронта.

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

$$Slack = Tda\_min - Tdr\_max.$$

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

$$Slack = Tfdd\_min - Tfcd\_max - Th + \\ + Todd\_min - Tofd\_max + Tdco\_min + Tbd\_min. \quad (2)$$

5. Первый способ создания временных ограничений в Vivado.

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

В уравнении (1) слагаемые, выделенные зеленым и оранжевым цветом, неизвестны временному анализатору Vivado, так как они описывают задержки для участков пути вне FPGA. Также неизвестным является значение периода *Tclk*. Создание ограничения на период тактового сигнала с помощью команды `create_clock` было рассмотрено в [1].

Для определенности будем считать, что напряжение источника питания микросхемы 74HC595 равно 4.5 В. Из таблицы 6.6 datasheet для 74HC595 [2] находим, что в этом случае её максимальная рабочая частота равна 31 МГц. Пусть требуется, чтобы плата на рисунке 1 могла работать при частоте тактового генератора (OSC) равной 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 объявляет тактовый сигнал `clk_10MHz` с периодом 100 нс, который поступает в FPGA через ножку `CLK`.

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

$$Slack = Tclk - input\_delay\_max + Tfcd\_min - Tfdd\_max - Tsu,$$

где

$$input\_delay\_max = Todd\_max - Tofd\_min + Tdco\_max + Tbd\_max. \quad (3)$$

В уравнении (3) слагаемые *Todd_max*, *Tofd_min* и *Tbd_max* описывают задержки, обусловленные распространением сигнала по дорожкам печатной платы. Их значения зависят от многих факторов, например, материала подложки, длины дорожек, типа дорожек (полосковые, микрополосковые и т.д.).

Для каждого типа дорожек существуют приближенные выражения для вычисления скорости распространения сигнала. Зная скорость сигнала и длину дорожки, можно оценить задержку распространения. Примеры приближенных расчетов для различных типов дорожек можно найти в [3].

Будем считать, что мы смогли оценить минимальные и максимальные задержки распространения сигналов по дорожкам печатной платы. В качестве примера примем следующие значения в наносекундах: *Tbd_max* = 0.6, *Tbd_min* = 0.5, *Todd_max* = 0.4, *Todd_min* = 0.2, *Tofd_max* = 0.3 и *Tofd_min* = 0.2. 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 до FPGA
set Tofd_max 0.3
set Tofd_min 0.2
```

Рассмотрим задержку *Tdco* из уравнения (3), которая соответствует интервалу времени между приходом фронта на тактовый вход SRCLK микросхемы 75HC595 и появлением данных на ее выходе QH’. Ее значение можно получить из таблицы 6.7 datasheet для 74HC595 [2]. Данная задержка обозначена как t_{pd} и при напряжении питания 4.5 В может изменяться от 17 до 32 нс. Эти значения также записываются в xdc-файл:

```
# задержка между приходом тактового сигнала и появлением
# данных на выходе QH' микросхемы 74HC595
set Tdco_max 32
set Tdco_min 17
```

Теперь можно создать временное ограничение на входной сигнал для анализа по *Setup*. Для этого в файл с ограничениями нужно внести следующие команды [4]:

```
# временное 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 объявлена переменная `idelay_max`, значение которой приравнивается *input_delay_max* из уравнения (3). Далее с помощью команды `set_input_delay` создается ограничение для входного сигнала. Опция `-max $idelay_max` задает задержку для анализа по *Setup*. Конструкция `[get_ports IDATA]` указывает, что ограничение накладывается на входной сигнал, поступающий в FPGA через ножку IDATA.

Важно отметить, что анализатору Vivado необходимо указать, каким сигналом тактируется запускающий триггер, так как он находится вне FPGA.

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 временному анализатору его тактовый сигнал известен.

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:

$$Slack = input\_delay\_min + Tfdd\_min - Tfcd\_max - Th,$$

где

$$input\_delay\_min = Todd\_min - Tofd\_max + Tdco\_min + Tbd\_min. \quad (4)$$

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

```
# временное 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 *input_delay_min*, после чего с помощью команды `set_input_delay` создается временное ограничение. Опция `-min` указывает, что ограничение предназначено для проведения анализа по *Hold*.

Полное содержимое xdc-файла представлено ниже:

```
# задержка между приходом тактового сигнала и появлением
# данных на выходе QH' микросхемы 74HC595
set Tdco_max 32
set Tdco_min 17

# 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
create_clock -period 100 -name clk_10MHz [get_ports CLK]

# временные ограничения 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 отражены во временных отчетах, полученных после размещения и трассировки проекта. На рисунке 4 представлен раздел *Summary* для анализа по *Setup*, в котором указан 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 (Clock Path Skew) и неопределенность (Clock Uncertainty) тактового сигнала.

Summary

Name	Path 1
Slack	67.711ns
Source	IDATA (input port clocked by clk_10MHz {rise@0.000ns fall@50.000ns period=100.000ns})
Destination	ff1_reg/D (rising edge-triggered cell FDRE clocked by clk_10MHz {rise@0.000ns fall@50.000ns period=100.000ns})
Path Group	clk_10MHz
Path Type	Setup (Max at Fast Process Corner)
Requirement	100.000ns (clk_10MHz rise@100.000ns - clk_10MHz rise@0.000ns)
Data Path Delay	0.952ns (logic 0.469ns (49.196%) route 0.484ns (50.804%))
Logic Levels	2 (IBUF=1 LUT1=1)
Input Delay	32.800ns
Clock Path Skew	1.486ns
Clock Uncertainty	0.035ns

Clock Uncertainty Equation

((TSJ^2 + TIJ^2)^1/2 + DJ) / 2 + PE

Total System Jitter (TSJ)	0.071ns
Total Input Jitter (TIJ)	0.000ns
Discrete Jitter (DJ)	0.000ns
Phase Error (PE)	0.000ns

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

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

Destination Clock Path

Delay Type	Incr (ns)	Path (ns)	Location	Netlist Resource(s)
(clock clk_10...z rise edge)	(r) 100.000	100.000		
	(r) 0.000	100.000	Site: N13	CLK
net (fo=0)	0.000	100.000		CLK
			Site: N13	CLK_IBUF_inst/I
IBUF (Prop_ibuf_I_O)	(r) 0.230	100.230	Site: N13	CLK_IBUF_inst/O
net (fo=1, routed)	0.634	100.864		CLK_IBUF
			Site: BUF...TRL_X0Y0	CLK_IBUF_BUFG_inst/I
BUFG (Prop_bufg_I_O)	(r) 0.026	100.890	Site: BUF...TRL_X0Y0	CLK_IBUF_BUFG_inst/O
net (fo=2, routed)	0.596	101.486		CLK_IBUF_BUFG
FDRE			Site: SLICE_X0Y0	ff1_reg/C
clock pessimism	0.000	101.486		
clock uncertainty	-0.035	101.450		
FDRE (Setup_fdre_C_D)	0.013	101.463	Site: SLICE_X0Y0	ff1_reg
Required Time		101.463		

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

Чтобы попасть на ножку FPGA в момент времени 100 нс, защелкивающий фронт должен быть сформирован на выходе тактового генератора в 100 - *Tofd_min* = 99.8 нс. Запускающий фронт появляется на один период раньше, то есть, когда время равно -0.2 нс. Через *Todd_max* = 0.4 нс запускающий фронт дойдет до ножки SRCLK микросхемы 74HC595 и начнет передачу данных. Спустя еще *Tdco_max* + *Tbd_max* = 32.6 нс данные достигнут ножки IDATA FPGA. Это произойдет в -0.2 + 0.4 + 32.6 = 32.8 нс, что соответствует задержке *input delay* во второй строке рисунка 6.

Data Path

Delay Type	Incr (ns)	Path (ns)	Location	Netlist Resource(s)
(clock clk_10...z rise edge)	(r) 0.000	0.000		
input delay	32.800	32.800		
	(f) 0.000	32.800	Site: P6	IDATA
net (fo=0)	0.000	32.800		IDATA
			Site: P6	IDATA_IBUF_inst/I
IBUF (Prop_ibuf_I_O)	(f) 0.413	33.213	Site: P6	IDATA_IBUF_inst/O
net (fo=1, routed)	0.484	33.696		IDATA_IBUF
			Site: SLICE_X0Y0	ff1_i_1/I0
LUT1 (Prop_lut1_I0_O)	(r) 0.056	33.752	Site: SLICE_X0Y0	ff1_i_1/O
net (fo=1, routed)	0.000	33.752		p_0_in
FDRE			Site: SLICE_X0Y0	ff1_reg/D
Arrival Time		33.752		

Рисунок 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 101.463 - 33.752 = 67.711 нс.

Ту же картину получаем для анализа по *Hold*. На рисунке 7 можно увидеть, что защелкивающий фронт достигает FPGA в нулевой момент времени. С учетом задержки распространения это означает, что он появился на выходе генератора на *Tofd_max* нс раньше. Для анализа по *Hold* запускающий и защелкивающий фронты появляются одновременно, поэтому на вход SRCLK микросхемы 74HC595 фронт приходит в момент времени -*Tofd_max* + *Todd_min* = -0.3 + 0.2 = -0.1 нс.

Destination Clock Path

Delay Type	Incr (ns)	Path (ns)	Location	Netlist Resource(s)
(clock clk_10...z rise edge)	(r) 0.000	0.000		
	(r) 0.000	0.000	Site: N13	CLK
net (fo=0)	0.000	0.000		CLK
			Site: N13	CLK_IBUF_inst/I
IBUF (Prop_ibuf_I_O)	(r) 1.001	1.001	Site: N13	CLK_IBUF_inst/O
net (fo=1, routed)	1.972	2.973		CLK_IBUF
			Site: BUF...TRL_X0Y0	CLK_IBUF_BUFG_inst/I
BUFG (Prop_bufg_I_O)	(r) 0.096	3.069	Site: BUF...TRL_X0Y0	CLK_IBUF_BUFG_inst/O
net (fo=2, routed)	1.639	4.708		CLK_IBUF_BUFG
FDRE			Site: SLICE_X0Y0	ff1_reg/C
clock pessimism	0.000	4.708		
clock uncertainty	0.035	4.744		
FDRE (Hold_fdre_C_D)	0.269	5.013	Site: SLICE_X0Y0	ff1_reg
Required Time		5.013		

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

Спустя *Tdco_min* нс данные появятся на выходе QH’ микросхемы 74HC595. Далее они будут распространятся по дорожке платы в течении *Tbd_min* нс и попадут на ножку IDATA FPGA. Это произойдет в момент времени -0.1 + 17 + 0.5 = 17.4 нс, что соответствует задержке *input delay* на рисунке 8.

Data Path				
Delay Type	Incr (ns)	Path (ns)	Location	Netlist Resource(s)
(clock clk_10...z rise edge)	(r) 0.000	0.000		
input delay	17.400	17.400		
	(f) 0.000	17.400	Site: P6	IDATA
net (fo=0)	0.000	17.400		IDATA
			Site: P6	IDATA_IBUF_inst/I
IBUF (Prop_ibuf_I_O)	(f) 0.861	18.261	Site: P6	IDATA_IBUF_inst/O
net (fo=1, routed)	0.904	19.165		IDATA_IBUF
			Site: SLICE_X0Y0	ff1_i_1/I0
LUT1 (Prop_lut1_I0_O)	(r) 0.100	19.265	Site: SLICE_X0Y0	ff1_i_1/O
net (fo=1, routed)	0.000	19.265		p_0_in
FDRE			Site: SLICE_X0Y0	ff1_reg/D
Arrival Time		19.265		

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

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

6. Второй способ создания временных ограничений в Vivado.

Теперь рассмотрим второй способ создания временных ограничений для входных сигналов. 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]
```

С 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 распространения равной *Tofd_min*. Конструкция `[get_clocks clk_10MHz]` указывает тактовый сигнал, для которого формируются 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 задержка распространения до микросхемы 74HC595 отличаются от задержки 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 заданы равными *Todd_min* и *Todd_max* соответственно.

Так как задержки для тактового сигнала уже установлены, уравнения для *input_delay_max* и *input_delay_min* упростятся и будут содержать только задержки распространения данных:

$$input\_delay\_max = Tdco\_max + Tbd\_max;$$
$$input\_delay\_min = Tdco\_min + Tbd\_min. \quad (5)$$

Временные ограничения для входного сигнала создаются с помощью команды `set_input_delay` в виде:

```
# временные ограничения 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 между приходом тактового сигнала и появлением
# данных на выходе QH' микросхемы 74HC595
set Tdco_max 32
set Tdco_min 17

# 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 ограничения 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, который формирует входные данные для FPGA, тактируется виртуальным сигналом clk_10MHz_Dev.

Summary

Name	Path 3
Slack	67.721ns
Source	IDATA (input port clocked by clk_10MHz_Dev {rise@0.000ns fall@50.000ns period=100.000ns})
Destination	ff1_reg/D (rising edge-triggered cell FDRE clocked by clk_10MHz {rise@0.000ns fall@50.000ns period=100.000ns})
Path Group	clk_10MHz
Path Type	Setup (Max at Fast Process Corner)
Requirement	100.000ns (clk_10MHz rise@100.000ns - clk_10MHz_Dev rise@0.000ns)
Data Path Delay	0.952ns (logic 0.469ns (49.196%) route 0.484ns (50.804%))
Logic Levels	2 (IBUF=1 LUT1=1)
Input Delay	32.600ns
Clock Path Skew	1.286ns
Clock Uncertainty	0.025ns
Clock Domain Crossing	Inter clock paths are considered valid unless explicitly exclude... timing constraints such as set_clock_groups or set_false_p

Clock Uncertainty Equation

((TSJ^2 + TIJ^2)^1/2 + DJ) / 2 + PE

Total System Jitter (TSJ)	0.050ns
Total Input Jitter (TIJ)	0.000ns
Discrete Jitter (DJ)	0.000ns
Phase Error (PE)	0.000ns

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 *Todd_max*.

Data Path

Delay Type	Incr (ns)	Path (...	Location	Netlist Resource(s)
(clock clk_10M...ev rise edge)	(r) 0.000	0.000		
clock source latency	0.400	0.400		
ideal clock network latency	0.000	0.400		
input delay	32.600	33.000		
	(f) 0.000	33.000	Site: P6	IDATA
net (fo=0)	0.000	33.000		IDATA
			Site: P6	IDATA_IBUF_inst/I
IBUF (Prop_ibuf_I_O)	(f) 0.413	33.413	Site: P6	IDATA_IBUF_inst/O
net (fo=1, routed)	0.484	33.896		IDATA_IBUF
			Site: SLICE_X0Y0	ff1_i_1/I0
LUT1 (Prop_lut1_I0_O)	(r) 0.056	33.952	Site: SLICE_X0Y0	ff1_i_1/O
net (fo=1, routed)	0.000	33.952		p_0_in
FDRE			Site: SLICE_X0Y0	ff1_reg/D
Arrival Time		33.952		

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

Еще через *Tdco_max* + *Tbd_max* = 32.6 нс данные достигнут ножки FPGA. Это произойдет в момент времени 32.6 + 0.4 = 33 нс. С учетом задержек распространения внутри FPGA фактическое время прибытия данных составит 33.952 нс.

Из рисунка 11 можно увидеть, что защелкивающий фронт тактового сигнала появляется на выходе генератора спустя один период в 100 нс. Через *Tofd_min* = 0.2 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. Величина *Slack* для анализа по *Setup* равна 101.674 - 33.952 = 67.721 нс.

Destination Clock Path

Delay Type	Incr (ns)	Path (ns)	Location	Netlist Resource(s)
(clock clk_10MHz rise edge)	(r) 100.000	100.000		
clock source latency	0.200	100.200		
	(r) 0.000	100.200	Site: N13	CLK
net (fo=0)	0.000	100.200		CLK
			Site: N13	CLK_IBUF_inst/I
IBUF (Prop_ibuf_I_O)	(r) 0.230	100.430	Site: N13	CLK_IBUF_inst/O
net (fo=1, routed)	0.634	101.064		CLK_IBUF
			Site: BUF...TRL_X0Y0	CLK_IBUF_BUFG_inst/I
BUFG (Prop_bufg_I_O)	(r) 0.026	101.090	Site: BUF...TRL_X0Y0	CLK_IBUF_BUFG_inst/O
net (fo=2, routed)	0.596	101.686		CLK_IBUF_BUFG
FDRE			Site: SLICE_X0Y0	ff1_reg/C
clock pessimism	0.000	101.686		
clock uncertainty	-0.025	101.661		
FDRE (Setup_fdre_C_D)	0.013	101.674	Site: SLICE_X0Y0	ff1_reg
Required Time		101.674		

Рисунок 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 *Todd_min* = 0.2 нс, он появится на входе SRCLK микросхемы 75HC595, а еще спустя *Tdco_min* + *Tbd_min* = 17.5 нс данные достигнут FPGA. В свою очередь защелкивающий фронт появится на выходе генератора также в нулевой момент времени и дойдет до ножки FPGA через *Tofd_max* = 0.3 нс. Это можно увидеть на рисунке 13.

Data Path

Delay Type	Incr (ns)	Path (...	Location	Netlist Resource(s)
(clock clk_10M...ev rise edge)	(r) 0.000	0.000		
clock source latency	0.200	0.200		
ideal clock network latency	0.000	0.200		
input delay	17.500	17.700		
	(f) 0.000	17.700	Site: P6	IDATA
net (fo=0)	0.000	17.700		IDATA
			Site: P6	IDATA_IBUF_inst/I
IBUF (Prop_ibuf_I_O)	(f) 0.861	18.561	Site: P6	IDATA_IBUF_inst/O
net (fo=1, routed)	0.904	19.465		IDATA_IBUF
			Site: SLICE_X0Y0	ff1_i_1/I0
LUT1 (Prop_lut1_I0_O)	(r) 0.100	19.565	Site: SLICE_X0Y0	ff1_i_1/O
net (fo=1, routed)	0.000	19.565		p_0_in
FDRE			Site: SLICE_X0Y0	ff1_reg/D
Arrival Time		19.565		

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

Destination Clock Path

Delay Type	Incr (ns)	Path ...	Location	Netlist Resource(s)
(clock clk_10MHz rise edge)	(r) 0.000	0.000		
clock source latency	0.300	0.300		
	(r) 0.000	0.300	Site: N13	CLK
net (fo=0)	0.000	0.300		CLK
			Site: N13	CLK_IBUF_inst/I
IBUF (Prop_ibuf_I_O)	(r) 1.001	1.301	Site: N13	CLK_IBUF_inst/O
net (fo=1, routed)	1.972	3.273		CLK_IBUF
			Site: BUF...TRL_X0Y0	CLK_IBUF_BUFG_inst/I
BUFG (Prop_bufg_I_O)	(r) 0.096	3.369	Site: BUF...TRL_X0Y0	CLK_IBUF_BUFG_inst/O
net (fo=2, routed)	1.639	5.008		CLK_IBUF_BUFG
FDRE			Site: SLICE_X0Y0	ff1_reg/C
clock pessimism	0.000	5.008		
clock uncertainty	0.025	5.033		
FDRE (Hold_fdre_C_D)	0.269	5.302	Site: SLICE_X0Y0	ff1_reg
Required Time		5.302		

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

Использование виртуальных тактовых сигналов для создания временных ограничений лучше отражает физическое распространение сигналов на плате и упрощает интерпретацию временных отчетов. Более того, данный способ рекомендуется Vivado. Если открыть вкладку XDC в Language Templates, то можно найти 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 *input_delay* задается в виде суммы задержек `tco` (*Tdco*) и `trce_dly` (*Tbd*), что соответствует уравнениям (5). Заметим также, что если пренебречь задержками тактового сигнала *Todd* и *Tofd*, то оба представленных выше способа будут полностью эквивалентны.

7. Несовпадающие значения *Slack*.

В заключении рассмотрим следующий интересный 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, можно обнаружить, что причиной неодинаковых значений является неопределенность тактового сигнала (clock uncertainty).

Данная неопределенность обусловлена наличием джиттера. Значение общего джиттера *Ttsj* (***T**otal **S**ystem **J**itter*) рассчитывается по формуле [5]:

$$Ttsj = \sqrt{Tsj_s^2 + Tsj_d^2},$$

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

По умолчанию для тактовых сигналов, которые поступают в FPGA, значение *Tsj* принимается равным 0.05 нс. Тогда для первого способа создания временных ограничений получаем:

$$Ttsj = \sqrt{0.05^2 + 0.05^2} = 0.071 \text{ нс}.$$

Неопределенность тактового сигнала равна 0.5 · *Ttsj* = 0.035 нс, что можно увидеть на рисунке 4.

Если используется второй способ, то один из тактовых сигналов является виртуальным, и его джиттер считается равным нулю. То есть в этом случае получаем, что *Tsj*s = 0 нс и *Ttsj* = *Tsjd* = 0.05 нс. Как показано на рисунке 9, неопределенность тактового сигнала равна 0.5 · *Ttsj* = 0.025 нс.

Чтобы значения *Slack* для обоих способов создания ограничений были равны, можно настроить величину джиттера с помощью команд `set_system_jitter` и `set_input_jitter` [4].

Заключение.

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

Ссылки.

1. Основы статического временного анализа. Часть 1: Period Constraint
2. Datasheet 74HC595
3. How to Calculate Trace Length from Time Delay Value for High-speed Signals
4. Using Constraints (UG 903)
5. Design Analysis and Closure Techniques (UG 906)