|  |  |
| --- | --- |
| ДИСЦИЛИНА | **Схемотехника устройств компьютерных систем Часть 2** |
|  |  |
| ИНСТИТУТ | **ИТ** |
| КАФЕДРА | **вычислительной техники** |
|  |  |
| ВИД УЧЕБНОГО | **Лекция** |
| МАТЕРИАЛА |  |
| ПРЕПОДАВАТЕЛЬ | **Тарасов И.Е.** |
|  |  |
| СЕМЕСТР | 3 |
|  |  |

2

8. Глобально асинхронные, локально синхронные схемы

8.1. Содержание раздела

Глобально асинхронные, локально синхронные схемы (GALS – Globally Asynchronous, Locally Synchronous) стали реакцией на возрастание площади цифровых микросхем и невозможность обеспечить трассировку тактового сигнала по такой большой площади. Вместо попытки синхронизировать все триггеры кристалла можно разбить проект на несколько фрагментов, внутри каждого из которых выполняются правила синхронного проектирования. Однако передавать данные между такими фрагментами («*тактовыми доменами*») можно с соблюдением определенных мер, иначе можно получить в приемном триггере неопределенное состояние («*метастабильное*»).

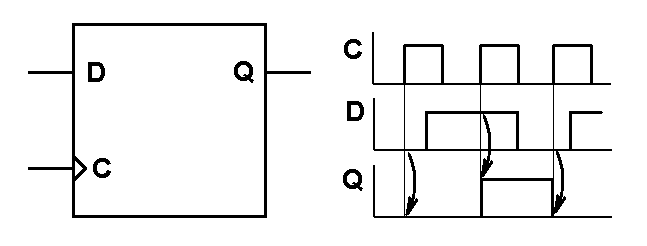
Кроме этого, внешние источники сигналов (микросхемы, кнопки и переключатели и т.д.) также обычно не привязаны к внутреннему источнику тактового сигнала. Для надежного приема логического уровня в проекте такие сигналы необходимо синхронизировать.

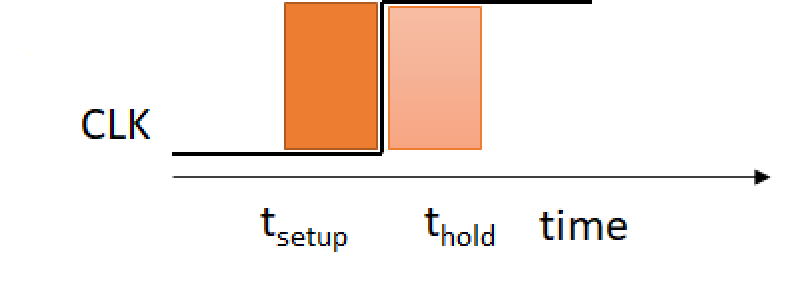
В этом разделе рассматривается понятие метастабильности и методы проектирования схем, позволяющие избежать негативных эффектов от его появления.

8.2. Временные характеристики синхронных устройств и понятие метастабильности

Для того, чтобы рассмотреть причины появления метастабильности, необходимо сначала уточнить требования к работе триггера. Ранее было показано, что D-триггер по фронту тактового сигнала записывает состояние, которое в этот момент было на его входе данных D. Этот процесс показан на рис. 8.1. Однако возникает вопрос: что именно понимается под термином «в этот момент»? Или, иными словами, насколько мал может быть тот интервал времени, который оказывает существенное влияние на процесс записи данных?

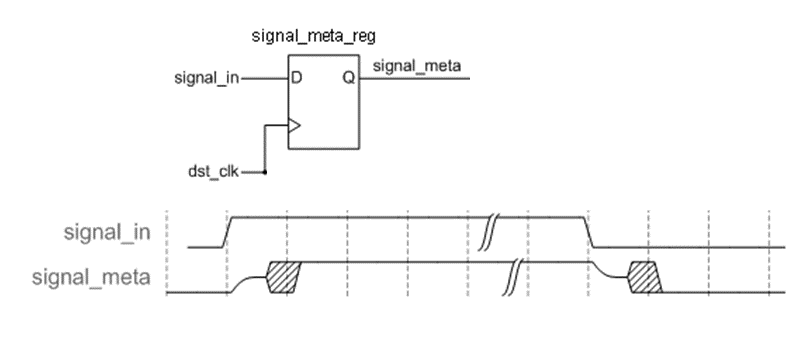
Производители микросхем на основе своих технологических библиотек предоставляют разработчикам данные об этих временных интервалах. Данные на входе D должны быть стабильными в течение *времени установки* tsetup до фронта тактового сигнала и оставаться такими же в течение *времени удержания* thold после фронта тактового сигнала.





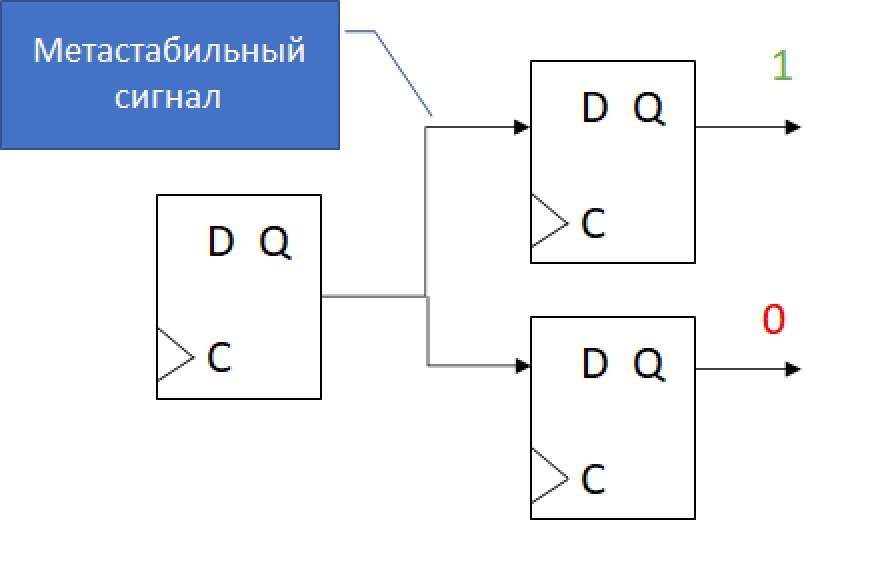
*Рисунок 8.1 Временные диаграммы работы триггера*

Естественный вопрос – что произойдет, если эти требования будут нарушены и состояние входа D изменится внутри показанных интервалов? Такое нарушение (называемое *timing violation*) может привести к попаданию транзисторов триггера в аналоговый режим работы, поскольку в момент записи данных на их затворах будет промежуточное напряжение, не соответствующее ни гарантированному нулю, ни гарантированной единице. На выходе триггера при этом также может появиться аналоговое напряжение, не соответствующее ни нулю, ни единице, как показано на рис. 8.2. С течением времени триггер придет к определенному цифровому состоянию, но будет это 0 или 1, невозможно сказать заранее.



*Рисунок 8.2 Временные диаграммы работы триггера при нарушении требований установки и удержания сигнала*

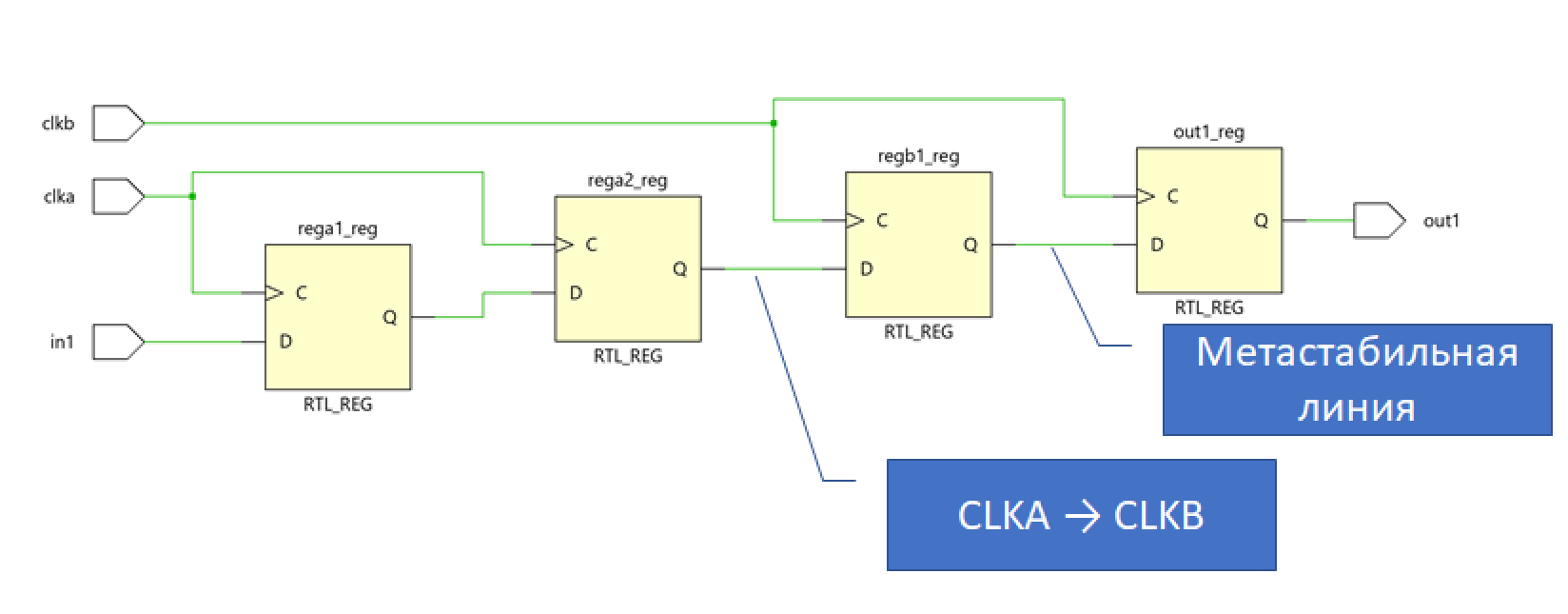
Именно это состояние триггера, в котором на его выходе временно присутствует аналоговое напряжение, не соответствующее ни 0, ни 1, называется метастабильным состоянием. Негативный эффект от метастабильности показан на рис. 8.3.



*Рисунок 8.3 Пример негативного эффекта от метастабильности*

На этом рисунке видно, что выход триггера, попавшего в метастабильное состояние, подан на входы двух других триггеров. По схеме ожидается, что в эти триггеры на следующем такте будет записано одно и то же логическое значение. Однако из-за метастабильности напряжение на входах этих триггеров находится в неопределенной зоне, и триггеры вследствие технологического разброса параметров вполне могут записать разные значения. Работа системы в этом случае становится непредсказуемой.

Самым простым способом устранения этого эффекта является применение цепочки из двух триггеров, как показано на рис. 8.4.



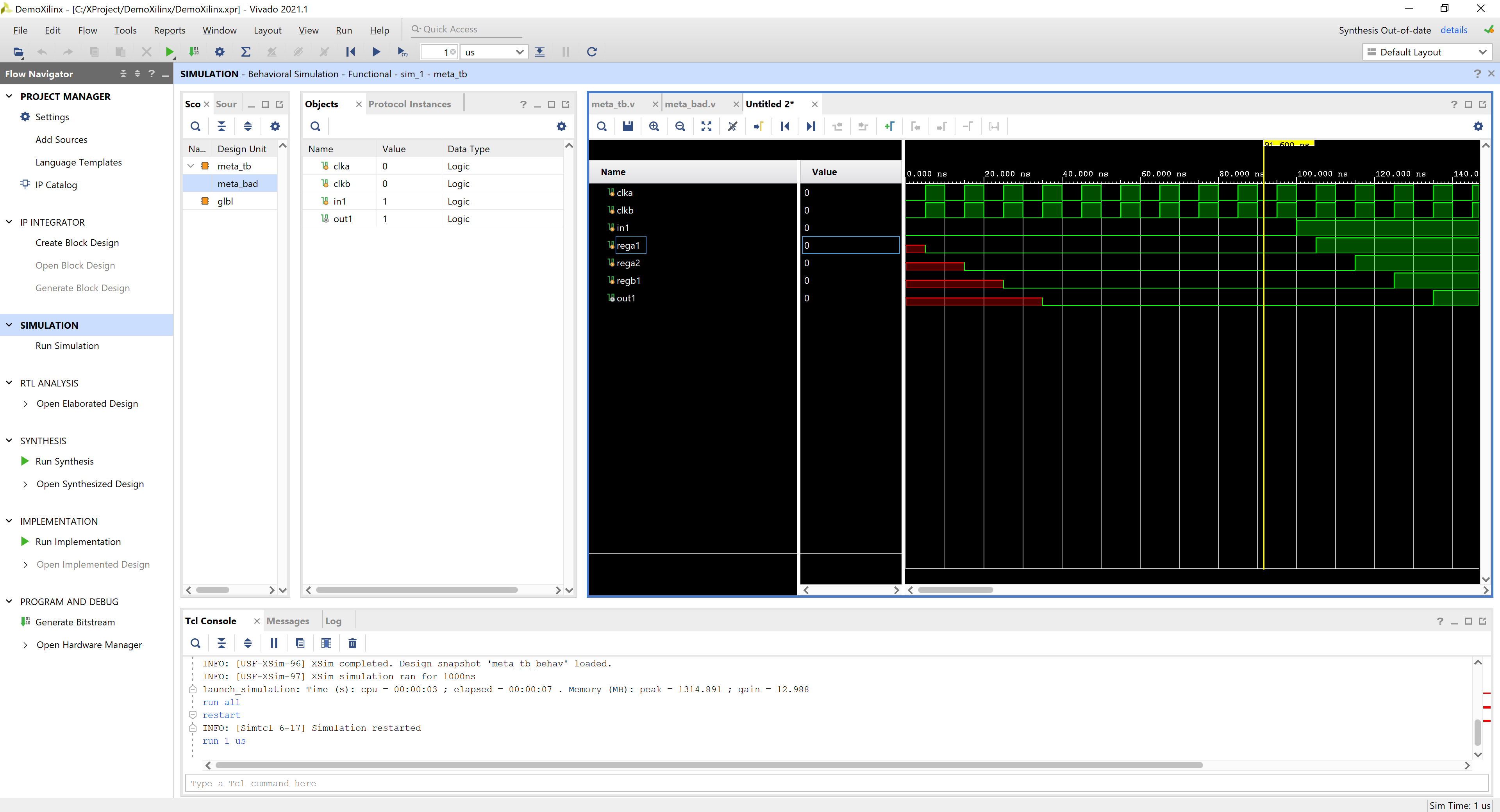
*Рисунок 8.4 Пример передачи сигнала между тактовыми доменами*

На этом рисунке существует линия, передаваемая от триггера, тактируемого сигналом CLKA, к триггеру, тактируемому сигналом CLKB. Совокупность компонентов, тактируемых одним и тем же тактовым сигналом, называется *тактовым доменом* (clock domain). Поэтому более короткими словами можно сказать, что происходит передача данных из домена CLKA в домен CLKB.

Подавление негативного эффекта происходит за счет того, что получаемая метастабильная линия подключена только к одному триггеру. Этот триггер (самый правый на схеме) однозначно определяет, перейдет ли он в состояние 0 или в состояние 1. Даже если это произойдет не в соответствии с ожиданиями разработчика, ситуация будет исправлена на следующем такте.

Метастабильное состояние имеет вероятностный характер. Нарушение требований по времени установки и удержания не означает автоматического попадания в метастабильное состояние, а только увеличивает вероятность такого события. Невозможно также и предсказать, в какое состояние (0 или 1) попадет триггер, выходя из этого состояния, как невозможно, например, угадать, в какую сторону упадет карандаш, если поставить его вертикально на острие. Можно лишь достаточно уверенно утверждать, что карандаш рано или поздно упадет, а если отклонить его от вертикали на какой-то угол, то он почти гарантированно упадет в соответствующую сторону. Аналогично тому, как карандаш может некоторое время стоять на острие, триггер также может некоторое время находиться в промежуточном состоянии. Тогда и второй триггер имеет вероятность попасть в свою очередь в метастабильное состояние на следующем такте. Однако вероятность этого на практике довольно мала. Тем не менее, в ряде случаев используют цепочки из трех триггеров – вероятность попадания в метастабильное состояние каждого последующего триггера уменьшается по экспоненциальному закону.

Неправильная работа с метастабильными сигналами часто является источником труднообнаружимых ошибок. Эта проблема усугубляется тем, что системы моделирования не могут адекватно воспроизвести попадание в метастабильное состояние, поскольку даже использование генератора случайных чисел не сможет воспроизвести именно то состояние, в которое попадет конкретный триггер. Поэтому логическое моделирование основывается на том, что триггер запишет то значение, которое было вычислено симулятором для его входа в момент времени, соответствующий фронту тактового сигнала. Независимо от того, насколько раньше фронта это состояние появилось, и как быстро сменится, состояние триггера будет смоделировано как стабильное. Например, на рис. 8.5 показаны результаты моделирования схемы, представленной на рис. 8.4. Видно, что никакого метастабильного состояния нет, и предупреждения о его возможном возникновении также не выдается.



*Рисунок 8.5 Моделирование передачи сигнала между тактовыми доменами. Метастабильность не может быть выявлена в модели*

Важным выводом здесь является то, что успешное моделирование цифровой схемы с несколькими тактовыми доменами не является гарантированным подтверждением ее нормальной работы.

Может возникнуть вопрос, почему же смоделированная схема будет работать некорректно, если показанные тактовые сигналы имеют одинаковую частоту? Моменты появления их фронтов также совпадают, поэтому нарушений времени установки или удержания в схеме быть не должно.

На практике оказывается, что тактовые сигналы, номинально имеющие одинаковую частоту, в действительно имеют разную форму. Прежде всего, у каждого тактового генератора имеется технологический разброс. Типичным значением разброса для кварцевого генератора общего назначения является 20-50 ppm (points per million, «единиц на миллион»). Т.е., при номинальной частоте 100 МГц разброс составит 100МГц \* 50 / 1000000 = 5000 Гц. Именно на это значение могут отличаться реальные частоты двух кварцевых генераторов. Кроме того, в процессе работы частота может изменяться из-за колебаний температуры. Поэтому модель тактового сигнала, описанная в виде номинального значения периода, не будет воспроизводиться в реальной микросхеме. Применение генератора случайных чисел также не помогает, поскольку смоделированный таким образом сигнал никак не определит поведения реального кварцевого генератора.

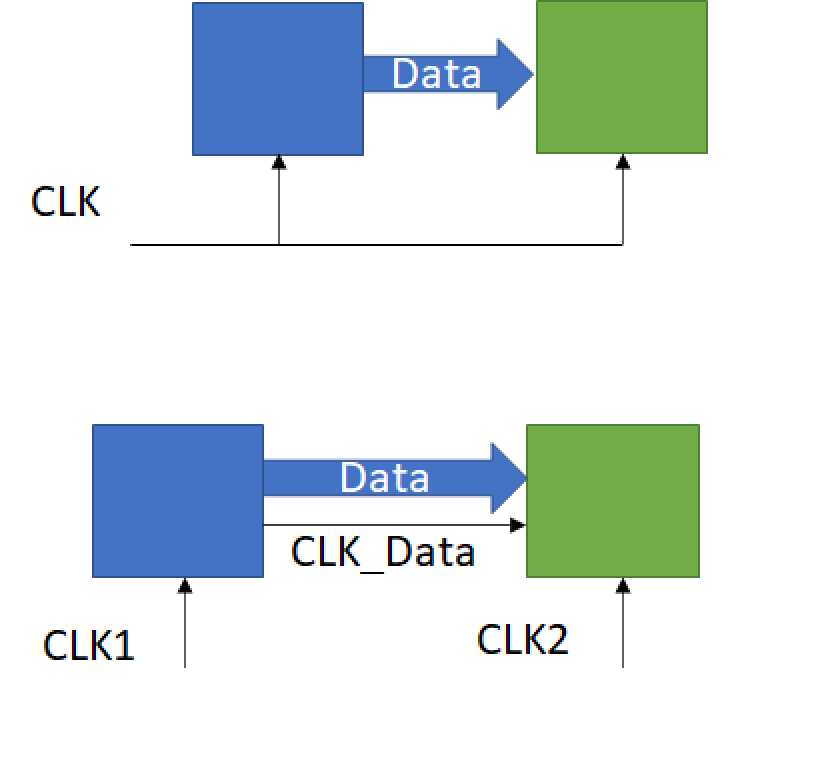
Единственным способом устранения негативных эффектов от метастабильного состояния является применение рекомендованных схем ресинхронизации – передачи сигналов между тактовыми доменами.

8.3. Виды синхронизации цифровых узлов

По способу взаимодействия тактовых сигналов цифровые схемы можно разделить на следующие типы:

* System synchronous
  + Микросхемы (или модули одной микросхемы) имеют общий тактовый сигнал. САПР обеспечивает анализ времени распространения данных и проверяет tsetup и thold
* Source synchronous («синхронизированный с источником»)
  + Микросхемы (или модули одной микросхемы) тактируются собственными тактовыми сигналами. С точки зрения микросхемы-приемника, данные могут измениться в любой момент времени
* Self-synchronous
  + Отдельного тактового сигнала нет. Отдельные фрагменты данных определяются по перепадам, фиксированным временным интервалам, частоте, фазе и т.д.

На рис. 8.6 показаны две схемы синхронизации цифровых узлов (это же можно применить и к двум отдельным микросхемам). Системная синхронизация удобна тем, что для нее нет необходимости выполнять ресинхронизацию, поскольку обе части схемы используют один и тот же тактовый сигнал. Однако такое поведение удается обеспечить не всегда. Препятствовать этому может разрастание площади синхронной схемы, вынуждающей использовать другую тактовую сеть, техническая невозможность подать на два устройства один и тот же тактовый сигнал или другие причины. Можно также отметить, что системно синхронные устройства обычно ограничивают тактовую частоту, особенно если речь идет о двух отдельных микросхемах. Время распространения сигнала по печатным проводникам может быть достаточно большим, что приведет к необходимости снижать тактовую частоту, давая сигналам достаточно времени для передачи из одной микросхемы в другую.



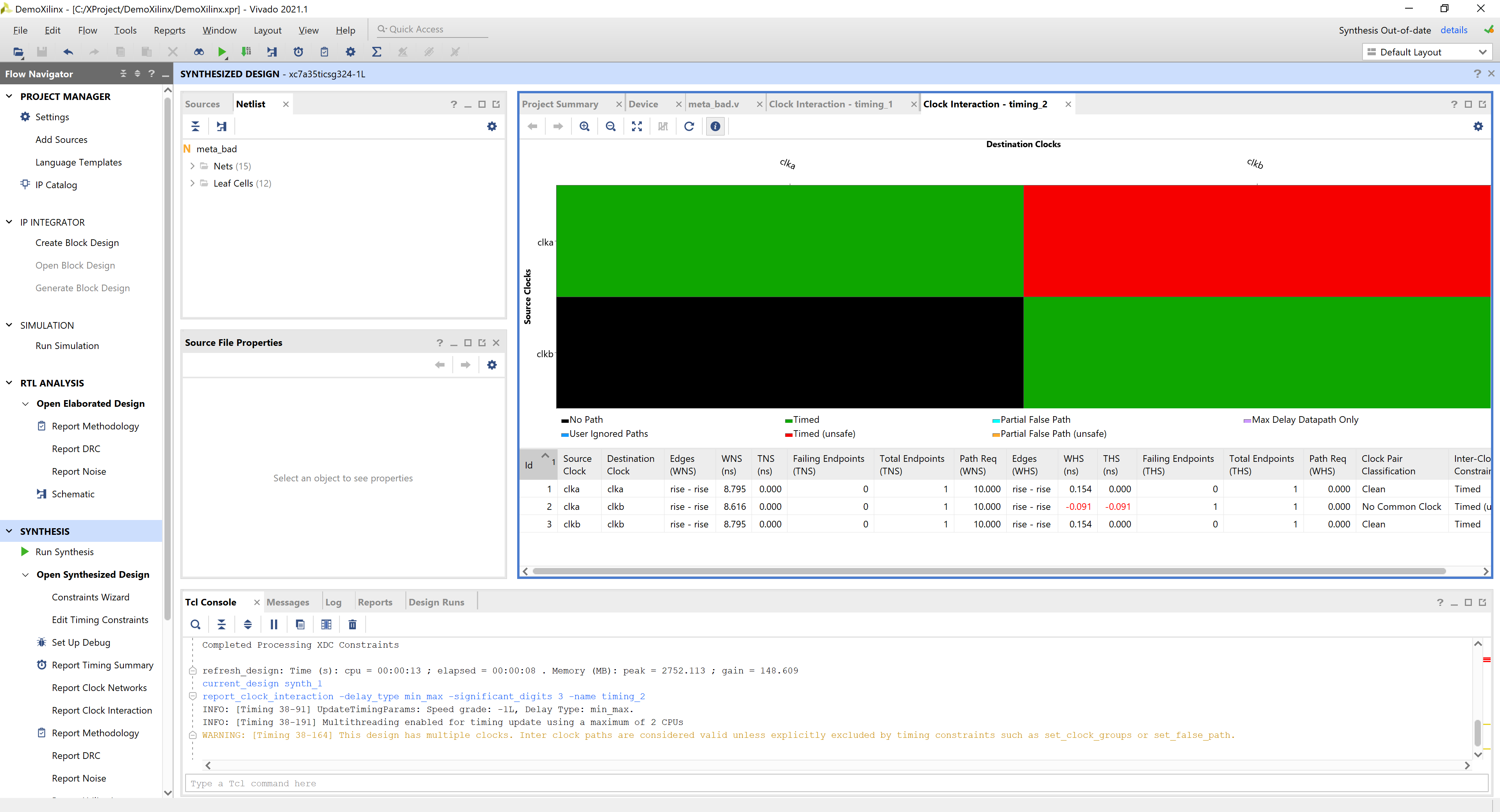
*Рисунок 8.6 Синхронная (сверху) и «синхронизированная с источником» (снизу) схемы*

Сигнал, синхронизированный с источником, может использовать общий тактовый сигнал, однако в любом случае микросхема-источник формирует собственный тактовый сигнал, фронт которого соответствует ситуации гарантированной корректности данных на выходе этой микросхемы. Преимуществом такой схемы является ее практически независимость от длины проводников, соединяющих две микросхемы. Если данные запаздывают на некоторое время из-за распространения по печатным проводникам, то на примерно такое же время будет запаздывать и фронт тактового сигнала. При этом можно рассчитывать на то, что если уж эти сигналы были сформированы в один и тот же момент времени, то при одинаковых задержках они придут на микросхему-приемник также в одинаковые моменты времени.

Чтобы использовать эти соображения, при разработке печатной платы необходимо тщательно следить за выравниванием длин проводников. Кроме того, не должно быть источников помех или заземляющих полигонов большой площади, которые находились бы рядом с некоторыми сигналами – в этом случае такие сигналы даже при равной длине проводника будут иметь другую форму, с более пологими фронтами, что приведет к более позднему пересечению порога устойчивого логического состояния. Обычно эти вопросы решаются отдельно, при разработке печатной платы, однако разработчик печатной платы должен получить соответствующие комментарии в виде требований к максимально допустимой неравномерности длины проводников в определенной группе.

Получаемый приемником сигнал в виде параллельной шины данных является несинхронизированным, поскольку он тактируется на рис. 8.6 сигналом CLK\_Data, а приниматься и обрабатываться должен схемой, тактируемой сигналом CLK2.

Выше было указано, что нет никакого способа принудительно сделать два тактовых сигнала синхронными. Вместо этого необходимо выявлять тактовые сигналы, между доменами которых передаются данные, и проверять для них наличие схем ресинхронизации. Часто САПР имеют встроенные инструменты для выявления взаимодействия между тактовыми доменами. Например, на рис. 8.7 показаны результаты выполнения команды report\_clock\_interaction в САПР Xilinx Vivado. Анализируемая схема соответствует показанной на рис. 8.4.



*Рисунок 8.7 Анализ взаимодействия тактовых доменов в САПР Xilinx Vivado*

Результаты анализа выводятся в виде таблицы, ячейки которой закрашены в зависимости от обнаруженного взаимодействия. По вертикали и горизонтали помещаются тактовые сигналы, имеющиеся в схеме. Зеленый цвет соответствует ситуации, когда взаимодействие нормально проанализировано в САПР. Очевидно, что зеленый цвет показан для ситуаций CLKA – CLKA и CLKB – CLKB, поскольку это синхронные схемы, тактируемые одними и теми же сигналами каждая.

Для ситуации CLKB – CLKA цепочек не обнаружено, поскольку на рис. 8.4 нет ни одного триггера в домене CLKB, который передавал бы данные в домен CLKA. В другой схеме ситуация, очевидно, может быть и не такой. Тем не менее, ячейка показана черной, что соответствует ситуации «такого взаимодействия не обнаружено». Соответственно, эта ячейка и не является проблемной.

Оттенки красного цвета соответствуют ситуациям, когда выявленное взаимодействие может вызвать проблемы из-за некорректного моделирования. В данном случае красным помечена ячейка CLKA – CLKB, и на схеме можно видеть, что есть триггер, тактируемый CLKA, выход которого подключен к входу, тактируемому CLKB. Красным цветом САПР указывает на обнаружение таких цепей, и задачей разработчика является добавление *проектных ограничений*, которые перечислили бы цепи, в работоспособности которых разработчик уверен исходя из архитектурных соображений.

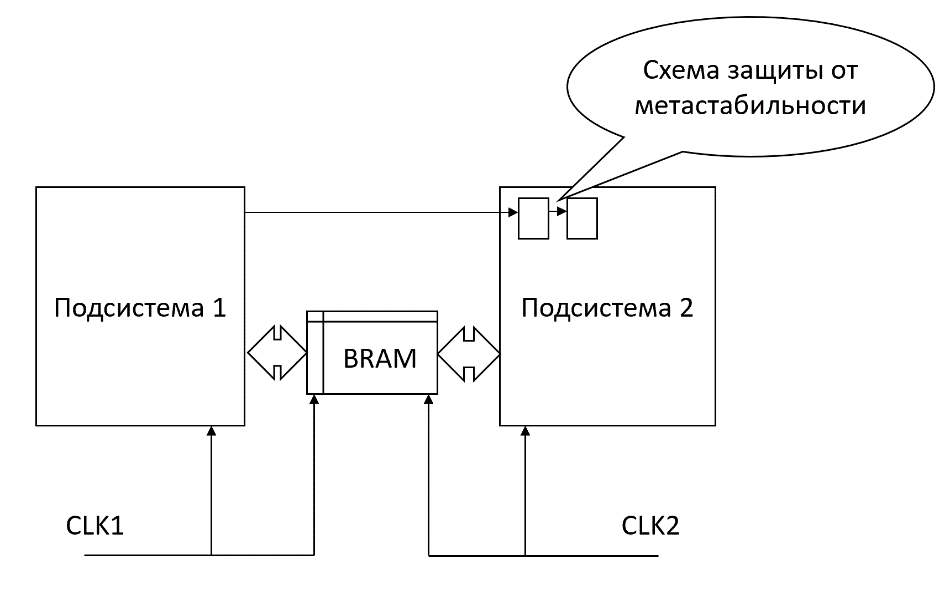
Можно подчеркнуть, что добавление проектных ограничений не превращает несинхронизированные сигналы в синхронизированные. Это просто информирование САПР о том, что какой-то сигнал в действительности не должен анализироваться, поскольку схема построена так, чтобы его метастабильное поведение не вызвало проблем в проекте. САПР не может выполнить ресинхронизацию самостоятельно или каким-то образом подстроить фазу тактового сигнала на основании проектных ограничений.

Если несколько тактовых сигналов формируются в одном генераторе (PLL или MMCM), то они не образуют перехода между тактовыми доменами. Поскольку тактовый генератор подстраивает фазу для целой группы своих выходов, можно сформировать в одном PLL сигналы 50, 100, 200 МГц, и положение их фронтов будет совпадать. В терминах САПР такие сигналы называются *related clocks*, в технически корректном переводе «связанные тактовые сигналы». В противоположность этому сигналы, полученные из разных источников, или даже сигналы одинаковой частоты, сформированные разными PLL, обозначаются как *unrelated clocks*, и к ним применимы все описанные соображения – передача данных между доменами этих сигналов требует ресинхронизации.

8.4. Архитектура GALS и схемы ресинхронизации

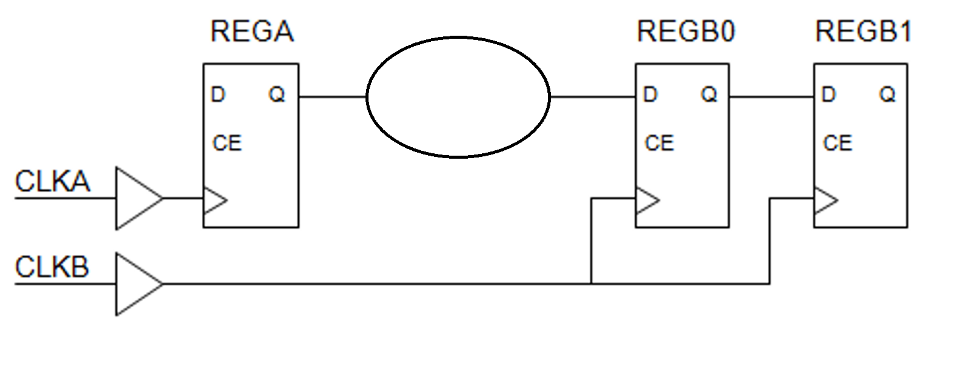
Для общего представления о схемах ресинхронизации необходимо ознакомиться с архитектурой GALS (Globally Synchronous, Locally Asynchronous). В общем виде ее можно представить на рис. 8.8, где имеются две подсистемы, каждая из которых является синхронной внутри, однако они используют независимые тактовые сигналы CLK1 и CLK2. Попытка соединить компоненты подсистем вызовет появление метастабильного сигнала, поэтому любая передача данных требует применения схем ресинхронизации.

В целом, ресинхронизация может быть выполнена либо в виде цепочки триггеров (минимум двух), как было показано выше, либо с помощью компонентов, которые аппаратно поддерживают два тактовых сигнала. Именно с этой целью в ПЛИС используют двупортовую блочную память, каждый порт которой может использовать собственный тактовый сигнал.



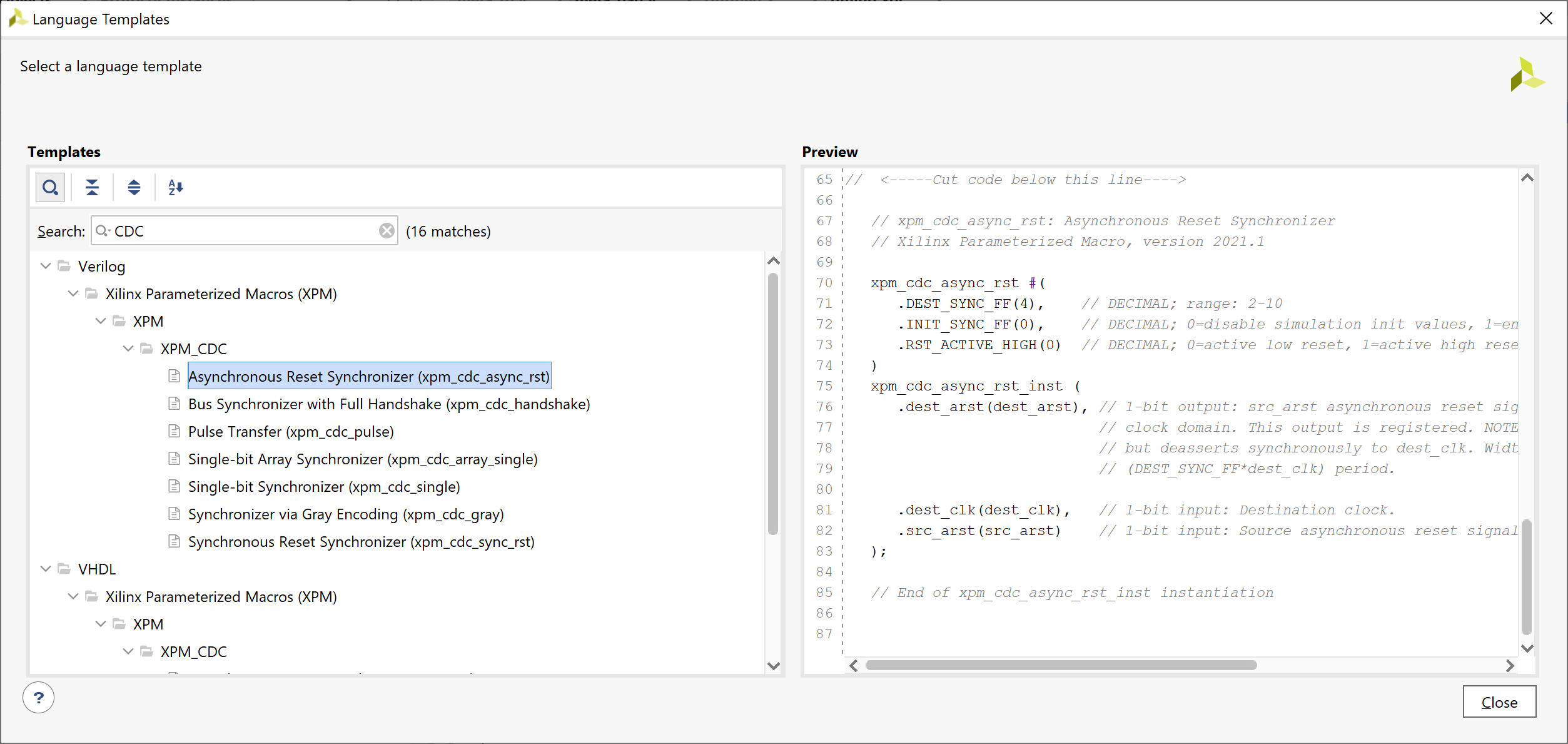
*Рисунок 8.8 Пример системы, выполненной в соответствии с архитектурой GALS*

Можно еще раз перечислить основные приемы ресинхронизации данных. На рис. 8.9 показана схема на основе цепочки триггеров. Предполагается, что второй триггер с меньшей вероятностью попадет в метастабильное состояние, поэтому его выходом можно пользоваться, не опасаясь появления промежуточного уровня напряжения, неоднозначно воспринимаемого другими компонентами.



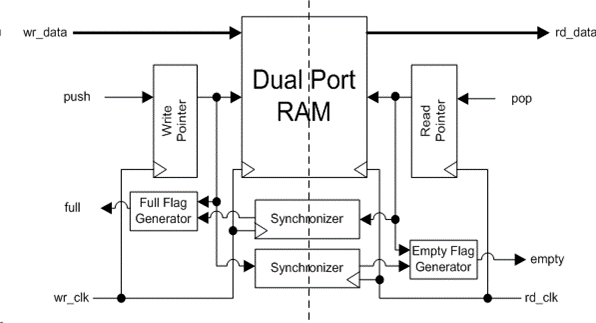
*Рисунок 8.9 Простейшая схема передачи данных между тактовыми доменами («схема ресинхронизации»)*

Для упрощения реализации схем ресинхронизации их шаблоны приведены в справочной системе САПР, как показано на рис. 8.10. Они сгруппированы в разделе Clock-Domain Crossing (CDC), что соответствует понятию «ресинхронизация».



*Рисунок 8.10 Шаблоны схем ресинхронизации в справочной системе САПР Vivado*

Один из подходов ресинхронизации данных с помощью двупортовой памяти показан на рис. 8.11.

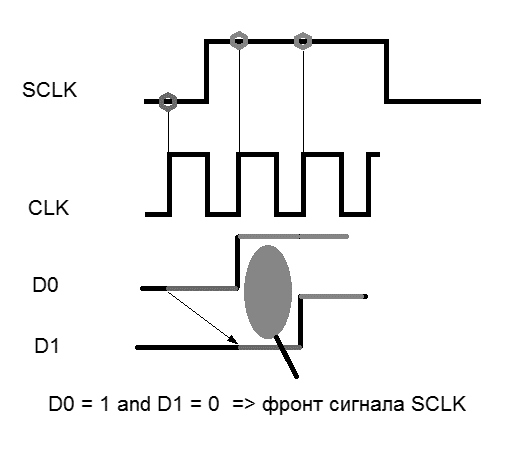


*Рисунок 8.11 Схема ресинхронизации на основе двупортовой памяти*

В этой схеме двупортовая память используется для организации очереди (FIFO – First In, First Out). Данные, передаваемые между тактовыми доменами, записываются в FIFO с одной стороны и читаются с другой. В действительности физического перемещения данных внутри FIFO не происходит, вместо этого перемещаются указатели на «голову» и «хвост» очереди данных. Для FIFO используются однобитные сигналы («флаги») empty и full, т.е. «пустое» и «заполненное». Если указатели головы и хвоста совпадают, это можно трактовать так, что данных для чтения нет. При записи данных указатель записи смещается и флаг empty снимается. Поскольку флаги передаются в другой тактовый домен через схемы ресинхронизации (показанные на рис. 8.11 как Synchronizer), домен-получатель узнает о наличии данных гарантированно после их физической записи в память. Иными словами, если флаг empty становится равным 0, можно производить чтение данных.

Указатели чтения и записи, а также поддержка флагов empty и full часто генерируются в модуле FIFO. Например, в ПЛИС Xilinx существуют аппаратные контроллеры FIFO, совмещенные с компонентами BRAM, которые могут применяться при необходимости. Автоматический синтез таких компонентов из поведенческого описания в настоящий момент не поддерживается.

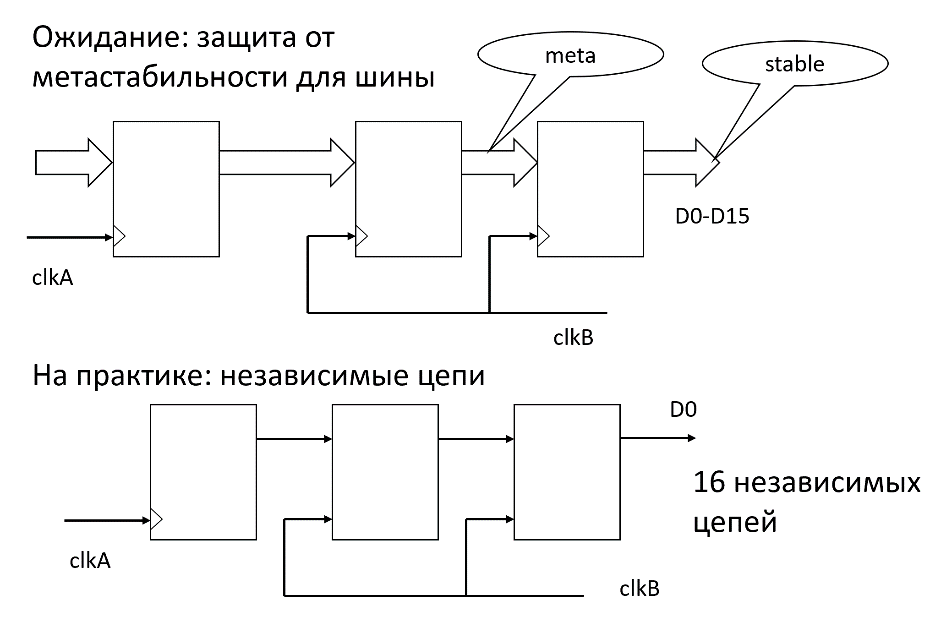
Еще одним приемом ресинхронизации является выделение фронта тактового сигнала. Этот прием можно эффективно использовать при разработке контроллеров SPI, работающих на прием данных (slave, или «ведомый» контроллер). Иллюстрация к работе такой схемы показана на рис. 8.12.



*Рисунок 8.12 Выделение фронта тактового сигнала*

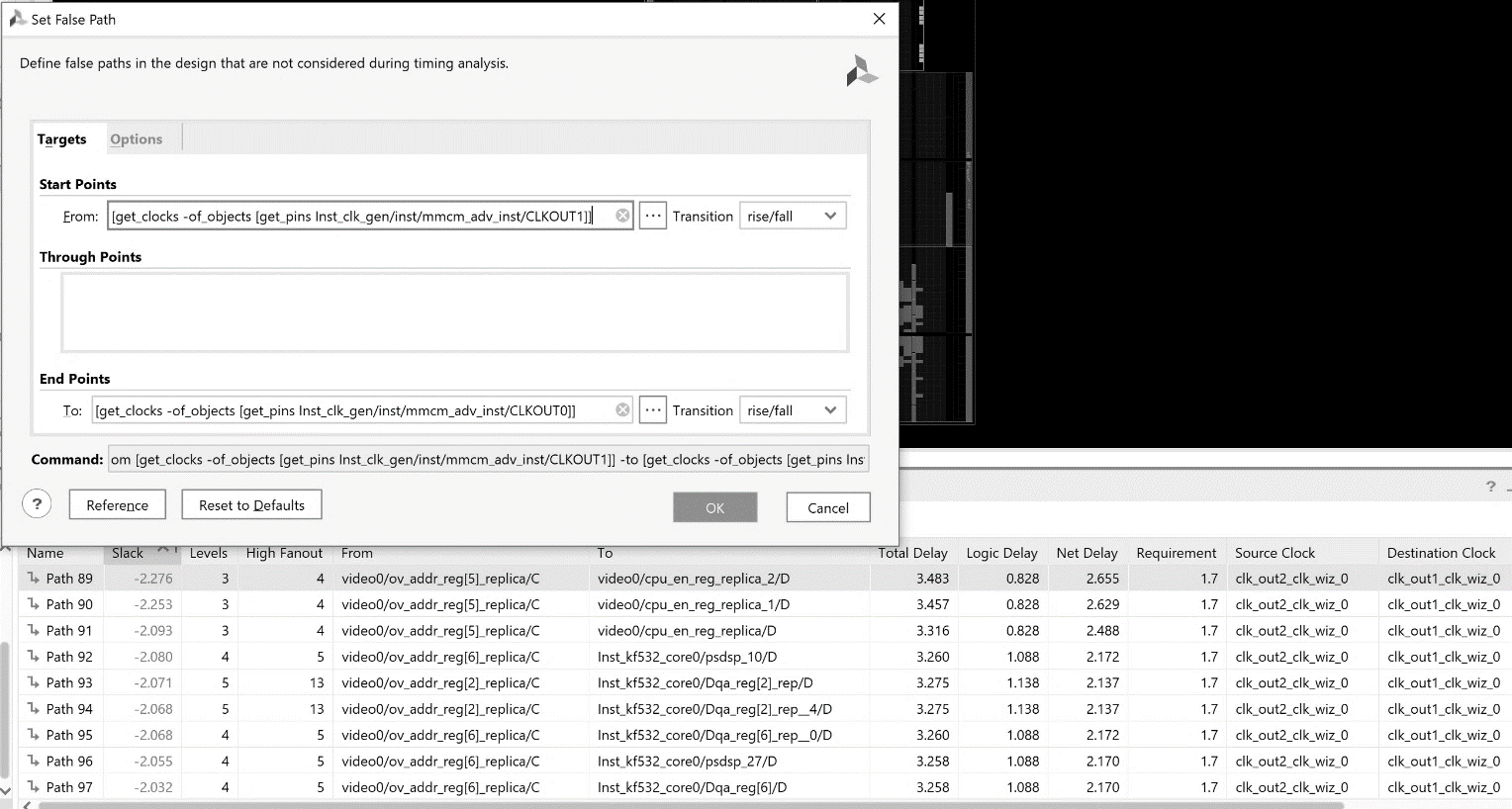
Если тактировать цепочку триггеров D0, D1 высокочастотным тактовым сигналом CLK, то эта цепочка будет захватывать последовательные состояния линии SCLK, причем триггер D1 будет содержать предыдущее состояние линии SCLK, а D0 – текущее состояние. Появление фронта сигнала на линии SCLK приведет к тому, что старое состояние линии, хранящееся в D1, будет равно 0, а новое, прочитанное в D0, станет равно 1. Если условие D0 = ‘0’ and D1 = ‘1’ использовать в качестве разрешения работы, сигналы данных интерфейса SPI смогут быть приведены к тактовой сети CLK.

Важным замечанием к ресинхронизации является вопрос использование цепочки триггеров для параллельных шин. На рис. 8.13 показана иллюстрация такого подхода. Если разработчик ожидает, что цепочка триггеров, обеспечивающая ресинхронизацию одного сигнала, точно так же работает с параллельной шиной, это будет неявно методической ошибкой, которая может быть выявлена только экспериментально. В действительности попадание в метастабильное состояние и сценарий выхода из него будет индивидуален для каждого разряда параллельной шины, и в большинстве случае на выходе такого блока будет комбинация «старых» и «новых» данных. Если учесть, что состояние какого-то разряда может и не изменяться, идентифицировать подобные ошибки достаточно сложно.



*Рисунок 8.13 Замечание к вопросу ресинхронизации параллельных шин*

Для правильного анализа метастабильных линий в проект следует добавить *проектные исключения* (constraint). Это необходимо для того, чтобы САПР исключала из анализа временные характеристики сигналов, передающих данные между тактовыми доменами. Пример диалогового окна, помогающего настройке этих параметров, приведено на рис. 8.14.



*Рисунок 8.14 Диалоговое окно настройки проектных исключений для ресинхронизации данных*

Ниже приведены строки на языке описания проектных ограничений xdc. Он основан на скриптовом языке Tcl и используется для ПЛИС Xilinx. Язык очень близок к более распространенному sdc, используемому для описания проектных ограничений СБИС.

set\_false\_path –from [get\_clocks CLKA] –to [get\_clocks CLKB]

set\_clock\_groups –physically\_exclusive –group CLKA –group CLKB

set\_clock\_groups –asynchronous –group CLKA –group CLKB

set\_max\_delay -from [get\_cells REGA] -to [get\_cells REGB0] 5 -datapath\_only

Смыслом приведенных описаний является указание САПР на то, что время распространения сигнала между триггерами, тактируемыми CLKA и CLKB, можно не анализировать. Первые три выражения являются более общими и работают на всю группу таких сигналов.

Команда set\_false\_path указывает, что цепь является «ложной» (false), т.е. несмотря на ее наличие задержка сигнала на этой цепи может быть любой, и это не окажет влияние на работоспособность проекта. Формально, любую цепь можно объявить «ложной», однако ответственность за такое решение лежит целиком на разработчике. Для передачи данных между тактовыми доменами такое объявление корректно.

Две следующие строки примерно одинаковы и используют варианты команды set\_clock\_groups, различаясь параметрами. В первом случае указывается параметр physically\_exclusive, который в том числе делает линии данных «ложными». Во втором случае параметр asynchronous имеет тот же эффект. Детали различия требуют более глубокого ознакомления с правилами описания тактовых сетей.

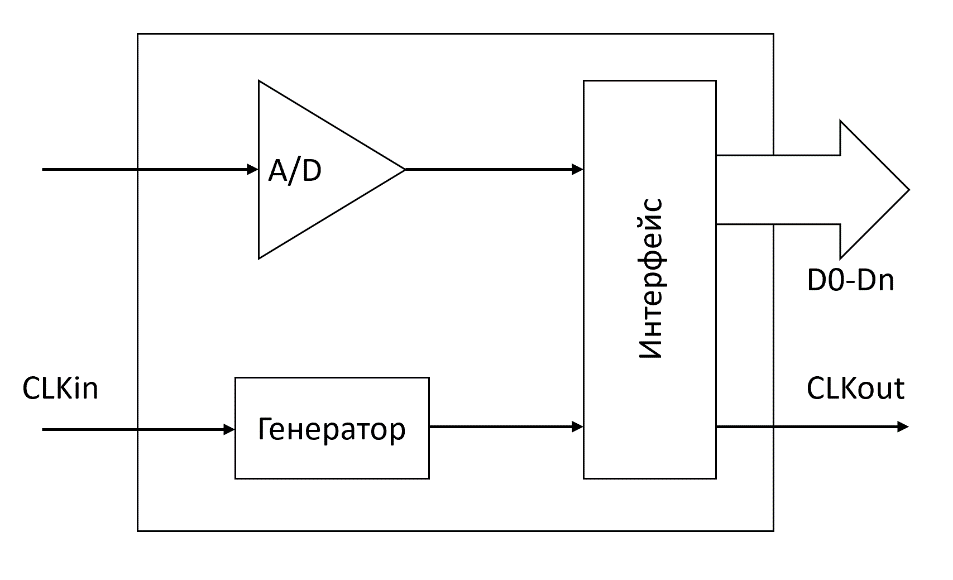
Наконец, команда set\_max\_delay является более низкоуровневой с точки зрения xdc. В данном случае она указывает САПР, что линия между двумя конкретными регистрами не может иметь задержку более заданной. Параметр 5 означает наносекунды и не имеет каких-либо жестких требований. Отличие от предыдущих команд заключается в том, что исключение цепей из анализа задержки формально дает возможность размещать соединяемые компоненты на каком угодно удалении друг от друга. Указание большой, но реалистичной максимальной задержки требует размещения компонентов относительно близко друг от друга.

В целом можно отметить, что управление проектными ограничениями является отдельным процессом при разработке. Добавление исключений следует проводить до тех пор, пока анализ взаимодействия тактовых доменов (clock interaction) не перестанет выявлять взаимодействие без явно описанных исключений (красные и оранжевые ячейки).

8.5. Примеры подключения устройств, требующих ресинхронизации данных

Ряд распространенных периферийных устройств являются асинхронными по отношению к той микросхеме, с которой они взаимодействуют. Поэтому при разработке интерфейсов для их подключения требуется реализовать схему ресинхронизации.

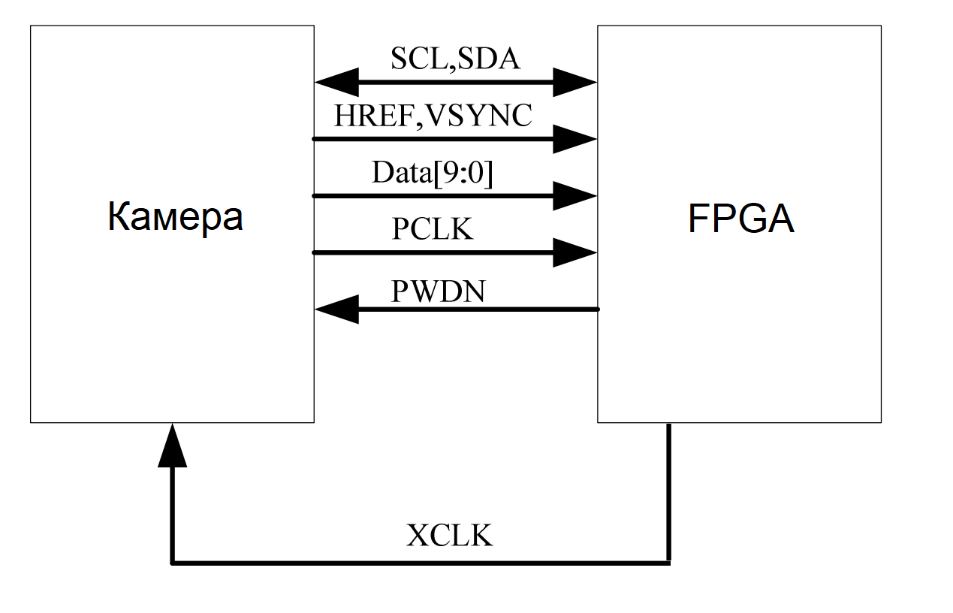
На рис. 8.15 показан интерфейс аналого-цифрового преобразователя с синхронным параллельным интерфейсом, синхронизированным с источником (source-synchronous). Такой подход часто используется по мере повышения тактовой частоты (например, на уровне 100 МГц или больше). Микросхема АЦП использует внешний опорный сигнал тактовой частоты, который преобразуется внутренним генератором (например, PLL). В результате выходная тактовая частота, сопровождающая передаваемые данные, имеет то же номинальное значение, однако отличающееся по фазе от опорного сигнала. Поэтому даже если входной сигнал подается из проектируемого интерфейса, нельзя ожидать, что данные можно будет захватывать по сигналу CLKin.



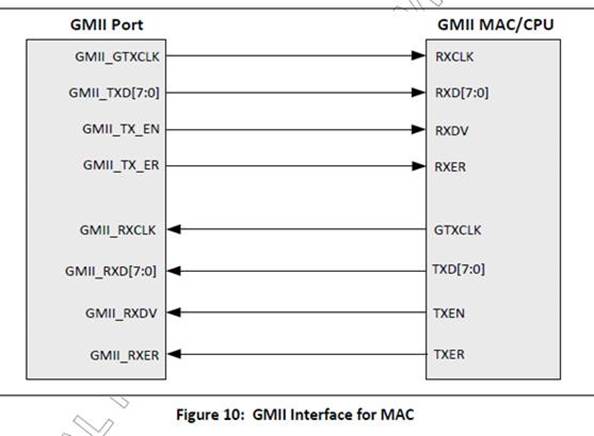
*Рисунок 8.15 Аналого-цифровой преобразователь, подключаемый по схеме входа, синхронизированного с источником*

Другим примером является цифровая камера. На рис. 8.16 показано подключение камеры с интерфейсом Camera Parallel Interface (CPI). В качестве опорного сигнала тактовой частоты камере требуется сигнал XCLK. Передаваемые данные Data сопровождаются фронтом тактового сигнала PCLK (Pixel Clock), который имеет то же номинальное значение, что и XCLK, однако отличается по фазе. Также камера передает синхросигналы HREF, VSYNC, а для ее настройки используется двухпроводной интерфейс с сигналами SCL, SDA, который подобен I2C.

На рис. 8.17 показано подключение микросхемы Ethernet PHY («физического уровня»). Можно видеть, что приемник (RX) и передатчик (TX) имеют собственные тактовые сигналы RXCLK и TXCLK, совпадающие по направлению с передаваемыми данными.



*Рисунок 8.16 Видеокамера с интерфейсом CPI (Camera Parallel Interface)*



*Рисунок 8.17 Подключение внешней микросхемы Ethernet PHY*

8.6. Выводы по разделу

Попытка изменять данные на входе триггера при подаче фронта тактового сигнала может привести к тому, что триггер попадает в аналоговый режим работы и на его выходе в течение какого-то времени присутствует напряжение между логическими 0 и 1. Невозможно заранее определить, как повлияет такое состояние триггера на подключенные к нему элементы, поэтому работа цифровой схемы становится непредсказуемой. Это состояние называется метастабильным и не устраняется проектными ограничениями или моделированием. Единственным способом избежать его негативного влияния на проект является применение схем ресинхронизации – привязки моментов изменения данных к фронтам тактового сигнала.

Простейшей схемой ресинхронизации является применение цепочки триггеров. Если первый триггер попадает в метастабильное состояние, то для второго триггера вероятность уменьшается. Этот прием не следует применять для ресинхронизации многоразрядных шин, поскольку каждый разряд в такой шине будет вести себя индивидуально.

Архитектурно устойчивым к метастабильности является компонент двупортовой памяти. На его основе можно создать очередь (FIFO), заполняемую на основе одного тактового сигнала и читаемую на основе второго тактового сигнала. Важно, чтобы чтение происходило при гарантированном наличии записанных данных, для чего необходимо анализировать однобитовый сигнал («флаг»), показывающий, что FIFO не пусто.

При анализе схем с несколькими тактовыми доменами не следует полагаться на результаты моделирования, поскольку формальные модели тактовых сигналов показывают поведение схемы в идеализированном варианте и не могут адекватно моделировать метастабильное состояние, которое имеет вероятностный характер.

Контрольные вопросы:

1. Что такое метастабильное состояние триггера?

2. Почему данные могут быть не синхронизированы с тактовым сигналом внутри микросхемы?

3. Второй триггер в цепочке не исправляет метастабильное состояние первого триггера, в чем же тогда заключается его роль?

4. Как использовать двупортовую память для ресинхронизации данных?

5. Как узнать, имеются ли в проекте взаимодействующие тактовые домены?

6. Какие проектные исключения применяются для описания сигналов, требующих ресинхронизации?

7. Почему крупные проекты могут разбиваться на несколько тактовых доменов?

8. Что такое «интерфейс, синхронизированный с источником»? Где он применяется?