Продолжение, начало в № 2'2002

Микроконтроллер для встроенного применения – NIOS.

Конфигурация шины и периферии

Иосиф Каршенбойм

ik@lmail.loniis.ru

6. Синхронизация шины

Этот раздел демонстрирует последовательность синхронизации (но не масштаб) для ряда циклов шины. Стандарт шины Avalon предлагает ряд параметров сопряжения сигналов шины для различных типов периферийных устройств.

Параметры, которые позволяют подключить любое специфическое периферийное устройство к определенным типам портов (как это определяется соответствующим разделом PORT_WIRING и другими назначениями, определенными в ptf).

Ряд назначений MODULE/SYSTEM_ BUILDER_ INFO, которые изменяют синхронизацию шины, описаны в разделе 4.2.5.

Несколько независимых вариантов синхронизации шины может быть определено и, во многих случаях, объединено. В любом случае, все циклы шины Avalon получены из «простого» основного цикла чтения (раздел 6.1.1) и основного цикла записи (раздел 6.2.1).

Эта временная диаграмма схематически изображает синхронизацию и показывает все доступные параметры синхронизации шины.

6.0.1. Синхронизация, примечания к диаграммам

Многие сигналы, произведенные мастером Avalon и внутренней логикой шины Avalon и направленные к периферийным устройствам, снимаются с выходов регистров Q. Эти сигналы будут иметь малое время T_{co} (clock-to-out) и не будут иметь никаких комбинаторных задержек от промежуточной логики. Переходы на этих сигналах, формируемых на регистрах, обозначены вертикальными областями «|».

Другие сигналы, поступающие от комбинаторной логики, не формируются непосредственно на Q-выводах регистров. Такие сигналы будут иметь некоторую задержку после фронта частоты (в дополнение к T_{co}), прежде чем их значение будет действительно.

Переходы на этих «комбинаторных» сигналах обозначены как дребезг сигналов: « # « (см. рис. 15).

6.1. Циклы чтения

6.1.1. Основной цикл чтения

* Состояние с нулевым временем ожидания.
* Нулевое время установки/хранения.
* Нет динамического изменения размеров шины.
* Пример показывает чтение данных от асинхронного периферийного устройства (см. рис. 16).

1) Передний фронт синхрочастоты, определяющий начало цикла шины.

2) Все выводы от мастера, формируемые на регистрах, достоверны.

3) Все декодированные сигналы chip-select достоверны.

4) Данные, полученные от выбранного периферийного устройства, достоверны.

5) Мастер производит выборку данных в регистры на этом переднем фронте.

Эта диаграмма показывает простой цикл шины чтения, без состояний ожидания. Адрес установлен мастером на переднем фронте частоты, и он получает данные от выбранного периферийного устройства на следующем переднем фронте синхрочастоты.

Модуль шины Avalon управляет сигналом выбора периферийного устройства (произведенный автоматически генерируемым декодером адреса внутри шины Avalon).

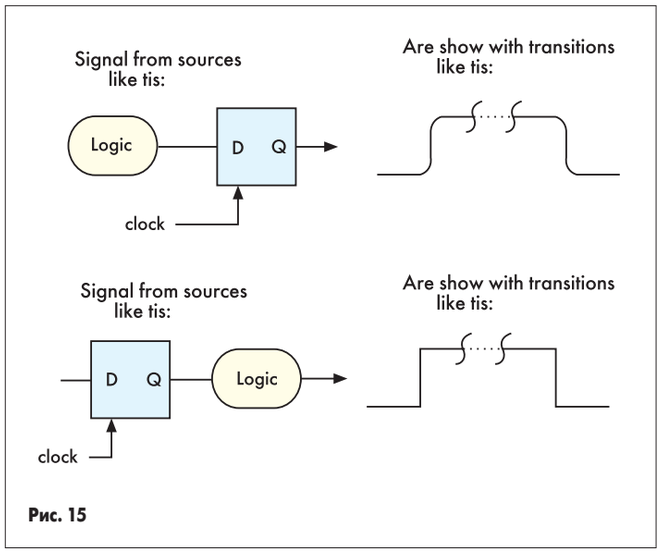

Рис. 15

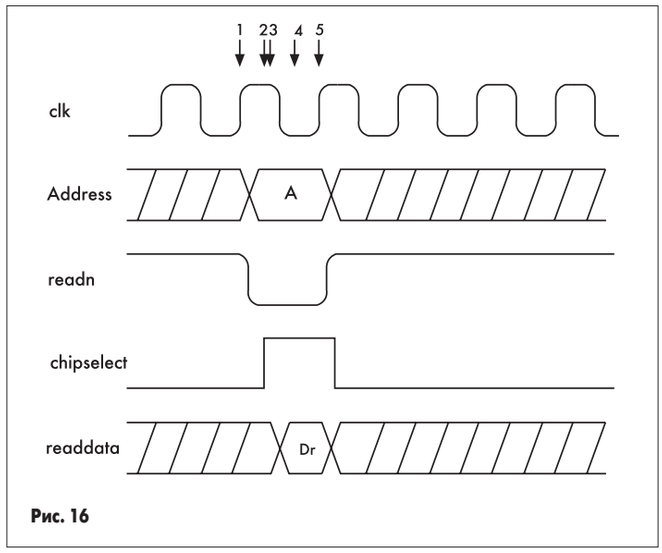

Рис. 16

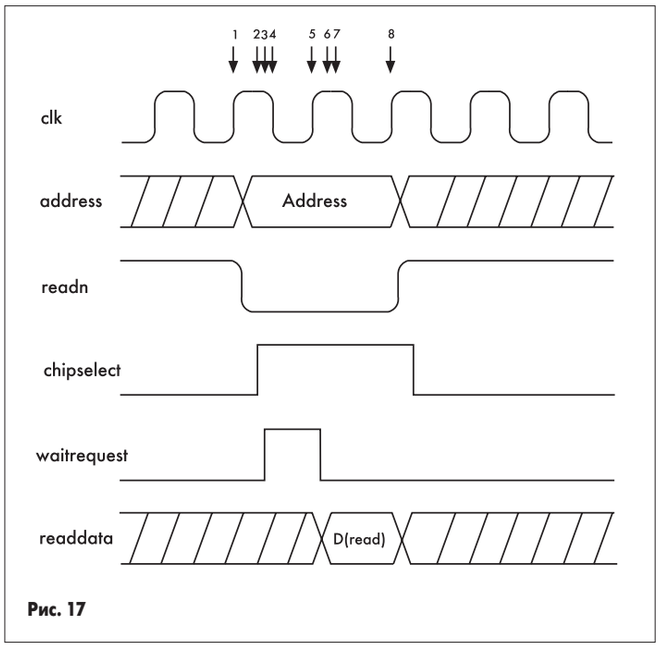

Рис. 17

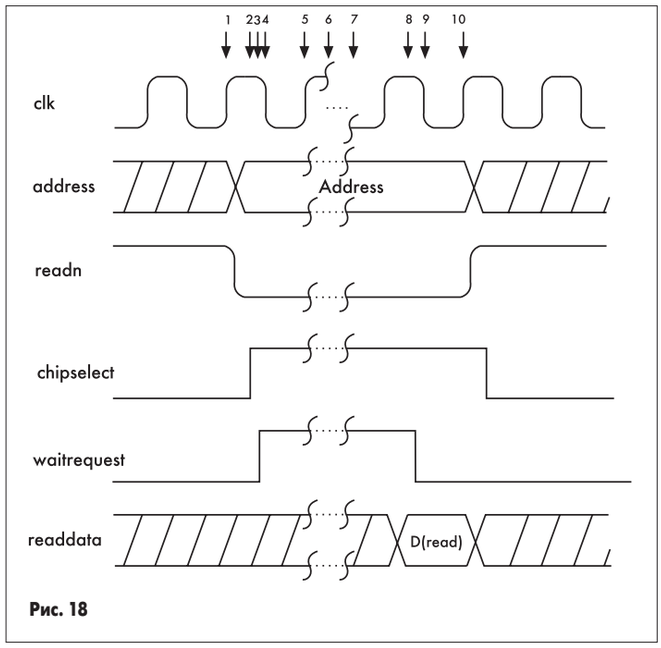

Рис. 18

Заметьте, что цикл чтения с нулевыми состояниями ожидания верен только для асинхронных периферийных устройств. Периферийное устройство должно выполнять свои функции, как только изменяется адрес — если бы оно ждало следующий фронт частоты, то это было бы слишком поздно.

Таким образом, все синхронные периферийные устройства не могут использовать цикл чтения состояния с нулевым временем ожидания подобно тому, как показано на этой диаграмме. Обычно все периферийные устройства на кристалле синхронны, и поэтому требуют по крайней мере одного состояния ожидания.

Асинхронная память, однако, является основным элементом системы Avalon, и состояние доступа с нулевым временем ожидания к главной памяти — скорее правило, чем исключение.

Учтите, что сигнал выбор устройства формируется на комбинаторной логике, и задержка в формировании этого сигнала, приходящего к самому устройству уменьшает запас по времени доступа для этого устройства.

Чтобы решить эту проблему, шина Avalon предлагает «registeredselectn» — сигналы выбора устройства, которые поступают непосредственно от Q-выводов регистров.

Сигнал «registeredselectn» будет чистый по сравнению с уродливо выглядящими переходными процессами, и в данном цикле синхрочастоты будет достоверен раньше, позволяя иметь большее время доступа для устройства.

Это звучит довольно хорошо, но есть также и другие последствия. Пример цикла шины с периферийным устройством с портом типа «registeredselectn» показан в разделе 6.1.6.

6.1.2. Цикл чтения с установленным состоянием ожидания

* 1 установленный цикл состояния ожидания, показанный в этом примере.

* Нулевое время установки/хранения.

* Нет динамического переключения размеров шины.

* Пример показывает чтение данных от синхронного периферийного устройства (см. рис. 17).

1) Передний фронт синхрочастоты определяет начало цикла шины.

2) Все регистровые выводы от мастера достоверны.

3) Весь декодированные сигналы выбора устройства достоверны.

4) Логика шины Avalon, передает установленный сигнал waitrequest (1) мастеру.

5) Периферийное устройство синхронно производит выборку сигналов адреса и вводов управления.

Мастер производит выборку waitrequest, определяет его активное состояние и ждет.

Логика шины Avalon внутренне заканчивает первый цикл состояния ожидания.

6) Логика шины Avalon передает сброшенный сигнал waitrequest (0) мастеру.

7) Данные от выбранного периферийного устройства достоверны.

8) Это периферийное устройство будет иметь одно состояние ожидания.

Мастер проверяет сигнал запроса ожидания, определяет что он сброшен и затем начинает следующий цикл.

Такой цикл чтения типичен для синхронных периферийных устройств на кристалле. Мастер представляет адрес и декодированный сигнал выбора чипа.

И адрес, и сигналы управления проверяются выбранным периферийным устройством на следующем переднем фронте частоты.

На 3-м фронте частоты мастер получает данные от выбранного периферийного устройства и переходит к следующему циклу шины.

Генератор состояния ожидания внутри шины Avalon управляет сигналом «waitrequest» мастера и устанавливает его в течение «среднего» фронта (событие 5, см. диаграмму выше), предоставляя периферийному устройству целый цикл частоты для получения его адреса и сигналов управления и для вывода его данных.

Эта диаграмма показывает доступ с единственным «установленным» циклом ожидания, что означает, что в ptf-разделе MODULE определено, что весь доступ по чтению к этому периферийному устройству будет иметь одно состояние ожидания.

Внутренний генератор состояния ожидания модуля шины Avalon всегда производит правильное число состояний ожидания для любого устройства, которое их запрашивает.

Устройства могут запрашивать различные числа установленных состояний ожидания для циклов шины по чтению и по записи в соответствии с тем, что записано в их ptf-разделе MODULE.

6.1.3. Цикл чтения с несколькими установленными состояния ожидания

Периферийным устройствам может быть определено любое (неограниченное) число установленных состояний ожидания в их разделе MODULE.

6.1.4. Цикл чтения с состояниями ожидания, управляемыми периферийным устройством

* Нулевое время предустановки/хранения.

* Нет динамического переключения размеров шины.

* Пример показывает чтение данных от синхронного периферийного устройства (см. рис. 18).

1) Передний фронт синхрочастоты определяет начало цикла шины.

2) Все регистровые выводы от мастера достоверны.

3) Весь декодированные сигналы выбора устройства достоверны.

4) Периферийное устройство устанавливает сигнал waitrequest (1).

5) Мастер проверяет сигнал запроса ожидания, определяет что он установлен и ждет.

6–7) Проходит произвольное неограниченное число периодов синхрочастоты.

8) Периферийное устройство выдает достоверные данные.

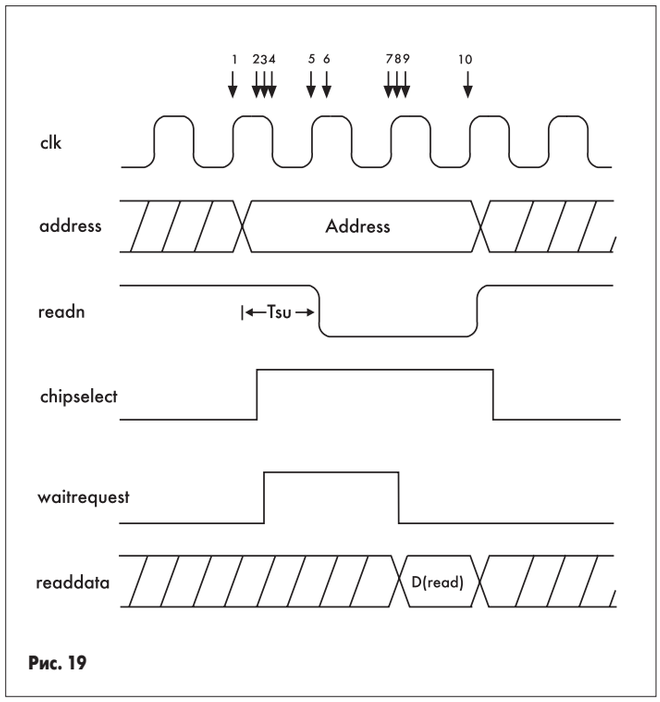

Рис. 19

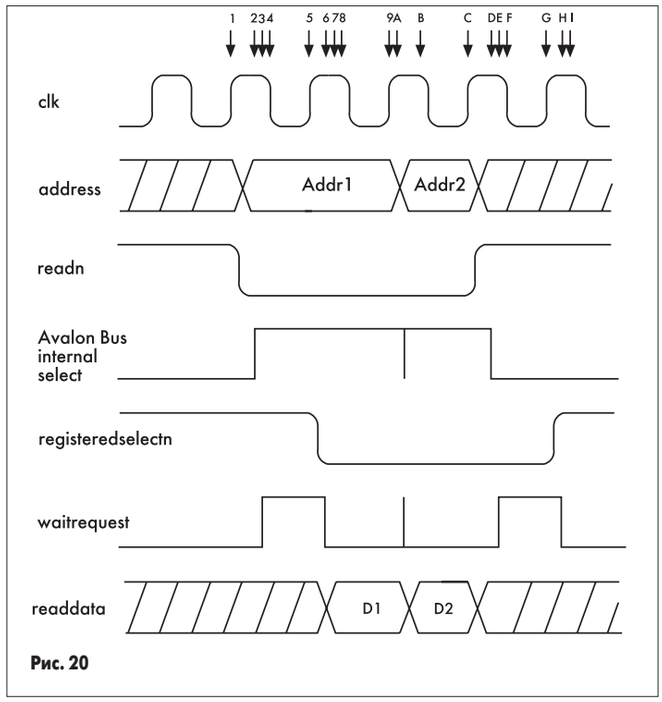

Рис. 20

9) Периферийное устройство сбрасывает waitrequest (0), передает сброшенный сигнал waitrequest (0) мастеру.

Мастер проверяет сигнал запроса ожидания, определяет, что он сброшен, и начинает следующий цикл.

Этот пример показывает цикл шины чтения периферийного устройства с состояниями ожидания «peripheral_controlled». Само устройство управляет количеством требуемых тактов ожидания при посредстве его собственного вывода типа «waitrequest».

Когда этот сигнал установлен в 1, он останавливает мастера (и всю шину Avalon) до тех пор, пока периферийное устройство не переведет свой выход в неактивное состояние. Используя этот механизм, периферийное устройство может иметь переменное количество тактов ожидания для выдачи данных.

Мастер произведет выборку данных от периферийного устройства на следующем переднем фронте синхрочастоты после того, как «watirequest» от периферийного устройства сброшен.

Шина Avalon не имеет особого времени ожидания по таймауту или какого-либо другого механизма для ограничения того, как долго периферийное устройство может останавливать мастера.

Проектировщики периферийных устройств должны знать, что состояние ожидания «peripheral_controlled» — не игрушка и что в конечном счете периферийные устройства являются ответственными за управление по «waitrequest».

Также необходимо отметить, что управление по сигналу мастера waitrequest предоставляется выбранному периферийному устройству только тогда, когда линия выбора периферийных устройств достоверна — даже когда время установки/хранения реально и строб readn/writen может быть сужен.

Схема состояния ожидания «peripheral_controlled» может применяться как для циклов чтения, так и для циклов записи.

6.1.5. Цикл чтения с предустановкой

* 1 установленное состояние ожидания, показанное в этом примере.

* 1 цикл предустановки.

* Нет динамического переключения размеров шины.

* Пример показывает чтение данных от синхронного периферийного устройства (см. рис. 19).

1) Передний фронт синхрочастоты определяет начало цикла шины.

2) Все регистровые выводы адреса от мастера достоверны.

3) Весь декодированные сигналы выбора устройства достоверны.

4) Логика шины Avalon передает установленный сигнал waitrequest (1) мастеру.

5) Мастер проверяет сигнал запроса ожидания, определяет, что он установлен, и ждет.

- Внутренняя логика шины Avalon заканчивает предустановку.
- Внутренняя логика шины Avalon продвигается к установленному состоянию ожидания.
- Для периферийного устройства на этом этапе адрес достоверен, а readn — неактивен.

6) Логика шины Avalon установила на своих регистрах активный readn-сигнал к периферийному устройству.

7) Мастер проверяет сигнал запроса ожидания, определяет, что он установлен, и ждет.

Периферийное устройство производит выборку адресов и проверяет активность сигнала readn (чтение).

Логика шины Avalon внутренне заканчивает состояние ожидания.

8) Логика шины Avalon сбрасывает waitrequest (0), передает сброшенный сигнал waitrequest (0) мастеру.

9) Периферийное устройство выдает достоверные данные.

10) Мастер производит выборку данных в регистры на этом переднем фронте синхрочастоты.

Мастер проверяет сигнал запроса ожидания, определяет что он сброшен и затем начинает следующий цикл.

Avalon slave может запрашивать произвольное число циклов предустановки и (или) хранения через их назначения в соответствующей секции MODULE.

Предустановка используется только для шинных циклов чтения. В стандарте шины Avalon времена хранения применяются к циклам записи, но не к циклам чтения.

Невозможно определить отдельно предустановку для чтения и отдельно для записи. Для ptf-назначения Setup_Time slave-устройства ptf-назначение обращается к обоим циклам шины — и к чтению, и к записи.

Отличное от нуля время хранения означает, что адрес, установленный для периферийного устройства, будет стабильно устойчив для N полных циклов синхрочастоты, перед тем как будет установлен сигнал управления периферийным устройством «readn». (N — значение Setup_Time в ptf.) В действительности строб (сигнал управления) «readn» будет сужен. Учтите, что сигнал chipselect — не суженный, он сформирован на полную продолжительность цикла шины (кроме задержки на логике).

Если устройство требует узкого сигнала выбора устройства (вместо узкого строба чтения), то непосредственно устройство или логика его интерфейса должна это сделать.

Может быть запрошено любое число циклов предустановки — в дополнение к любым установленным состояниям ожидания, которые требуются для периферийного устройства.

Так, периферийное устройство с двумя тактами предустановки и тремя установленными состояниями ожидания фактически имело бы шинный цикл длительностью шесть циклов (2 предустановки, 3 ожидания, и 1 активный).

Заметьте, что в этих диаграммах суженный сигнал readn показывается как сигнал, сформированный на регистре.

Это имеет место потому, что шина Avalon будет иметь специализированный регистр для сигналов readn и writen для каждого периферийного устройства, которое запрашивает предустановку, отличную от нуля и/или время хранения.

Следовательно, любая периферия с ненулевыми предустановкой или временами хранения не может совместно использовать readn или writen сигналы (см. раздел 5.2.4).

6.1.6. Цикл чтения с сигналом выбора устройства «registeredselectn»

* 0 установленных состояний ожидания, показанных в этом примере.

* Нулевое время предустановки/хранения.

* Нет динамического установления размеров шины.

* Пример показывае многократные последовательные операции чтения от асинхронного периферийного устройства (см. рис. 20).

1) Передний фронт синхрочастоты, определяющий начало цикла шины.

2) Все сформированные на регистре выводы от мастера достоверны.

3) Внутренний сигнал шины Avalon — сигнал выбора периферийного устройства достоверен.

4) Сигнал для мастера waitrequest достоверен, потому что «reg. select != select».

5) Мастер производит выборку waitrequest, замечает что он установлен, ждет.

Внутренний сигнал Avalon «select» проверяется регистром registeredselectn.

6) Registeredselectn на периферийное устройство установлен (активный).

7) Логика шины Avalon сбрасывает сигнал waitrequest для мастера.

8) Периферийное устройство выдает достоверные данные под чтение (которые находились в Addr1).

9) Мастер производит выборку данных в регистры на этом переднем фронте.

Мастер проверяет сигнал запроса ожидания, замечает что он сброшен, и начинает следующий цикл.

A) При изменении сигналов адреса внутренний сигнал waitrequest в Avalon может дать сбой.

B) Периферийное устройство выдает достоверные данные под чтение (которые находились в Addr2).

C) Мастер производит выборку данных в регистры на этом переднем фронте.

Мастер проверяет сигнал запроса ожидания, замечает, что он сброшен, начинает следующий цикл.

D) Адрес (не для того же самого периферийного устройства) достоверный.

E) Внутренний сигнал Avalon — сигнал выбора достоверный (сброшенный)

F) Waitrequest для мастера достоверный, потому что «reg. select != select».

G) Мастер проверяет сигнал запроса ожидания, замечает, что он установлен, и ждет.

Внутренний сигнал Avalon — сигнал выбора проверяется в регистре registeredselectn.

H) Registeredselectn на периферийное устройство достоверный (неактивный).

I) Логика шины Avalon сбрасывает сигнал waitrequest для мастера.

Эта диаграмма показывает два последовательных цикла чтения устройства со входом типа «registeredselectn».

Цель применения входа типа «registeredselectn» — получить «чистый» сигнал по отношению к комбинаторно формируемому сигналу chipselect. Кроме того, это также (как это ни парадоксально) дает асинхронным устройствам (например, SRAM) большее количество времени от цикла синхрочастоты, чтобы использовать его как время доступа, потому что сигнал выбора готов сразу же после фронта синхрочастоты — даже при том, что он задержан на один полный цикл частоты!

В основном «registeredselectn» — то же самое, что и chipselect, за исключением того, что он имеет инверсную логику (потому что большая часть внешних устройств, например ОЗУ, используют отрицательные сигналы для выбора устройства) и затем пропущенный через регистр задержки (чтобы уменьшить время clock-to-out).

Это очень просто, но внутренняя логика шины Avalon должна иметь некоторые противоречия, чтобы удостовериться, что отсроченный сигнал выбора устройства принят во внимание.

Чтобы это сделать, используют простое правило:

Всякий раз, когда внутренний (текущий) и сформированный на регистре (задержанный) версии сигнала выбора slave различны — ожидание.

Учтите, что шина Avalon устанавливает waitrequest всякий раз, когда внутренний выбор и вывод !registeredselectn отличается.

Конечный результат этой схемы в том, что сигнал выбора устройства скорее устанавливается в цикле шины и что состояния ожидания предоставляются перед первым и после последнего цикла, в любой последовательности доступа к устройству.

Однако состояния ожидания не вставляются, когда к устройству непрерывно обращаются, что обычно имеет место для основной памяти.

Поэтому сигналы типа registeredselectn часто используют (вместе со временем хранения в полцикла) для SRAM, используемой в качестве основной памяти.

Заметьте, что в диаграмме синхронизации (см. выше) сигнал registeredselectn установлен для «дополнительного» целого цикла частоты после того, как шина Avalon получила данные от выбранного устройства.

Обычно это означает, что от этого устройства выполнена «поддельная» операция чтения.

Если устройство — SRAM, то нет никакого вреда от сделанного «поддельного» чтения (кроме как большего количества потребляемой мощности).

Но что, если немедленно последующий цикл шины будет (когда сигнал registeredselectn еще активен) циклом записи?

Тогда «поддельная» операция записи будет выполнена в это устройство. Не может быть двух мнений — это плохо.

Чтобы устранить этот эффект, мастер шины Avalon должен иметь возможность представлять «неактивный цикл» всякий раз, когда они исполняют операцию записи после операции чтения.

Процессор Nios автоматически вставит такой неактивный цикл перед каждой операцией записи, если он используется в системе, где присутствуют сигналы с любым «registeredselectn» (это определяется при генерации системы, в ptf-файле).

6.2. Циклы записи

Каждому циклу записи шины может предшествовать неактивный цикл шины, в течение которого мастер Nios ни читает, ни пишет.

Операция	readn	writen
Чтение	0	1
Запис	1	0
Простой	1	1

Это состояние, когда ни writen, ни readn не установлен.

Следующая таблица показывает значения сигналов Авалона readn и witen.

Все нижеследующие диаграммы циклов записи показывают, что мастер мог быть в состоянии «простоя» в течение цикла частоты, предшествующего началу строба записи.

В течение этого неактивного цикла адрес для последующей операции записи может быть представлен на шине address, и поэтому сигнал chipselect выбранного периферийного устройства может стать активным «раньше на один цикл».

Вот почему сигнал chipselect изображается как «неизвестный» перед каждой операцией записи.

Необходимо следить за тем, чтобы периферийные устройства выполняли операции записи только тогда, когда writen активен (и, конечно, их сигнал chipselect установлен).

6.2.1. Простой цикл записи

* Состояние с нулевым временем ожидания.

* Нулевое время предустановки/хранения.

* Нет динамической установки размеров шины.

* Пример показывает запись данных в синхронное периферийное устройство (см. рис. 21).

1) Передний фронт синхрочастоты, определяющий начало цикла шины.

2) Все сформированные на регистре выводы от мастера достоверны.

3) Все декодированные сигналы выбора устройства достоверны.

4) Периферийное устройство производит выборку данных в свой входной регистр, осуществляя запись.

5) Мастер переходит к следующему циклу шины.

Эта диаграмма показывает самый простой цикл записи, без установленных состояний ожидания и без предустановки или хранения. Мастер управляет адресом, сигналами управления и сигналами записи данных для slave-устройства после переднего фронта синхрочастоты. Периферийное устройство принимает адрес, данные и сигналы управления на следующем переднем фронте синхрочастоты.

Заметьте, что эта синхронизация шины не применима к асинхронным устройствам. Цикл записи закончится, когда сигнал «writen» сбрасывается, что совпадает с переходами на шине адреса и данных записи.

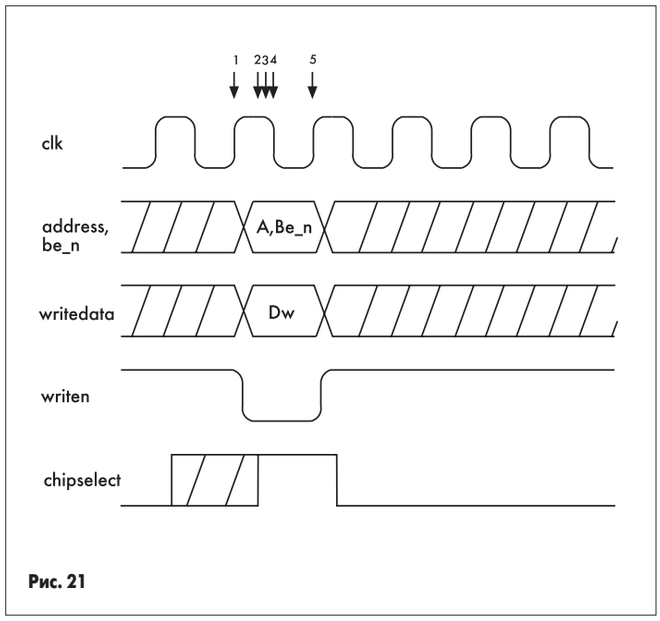

Рис. 21

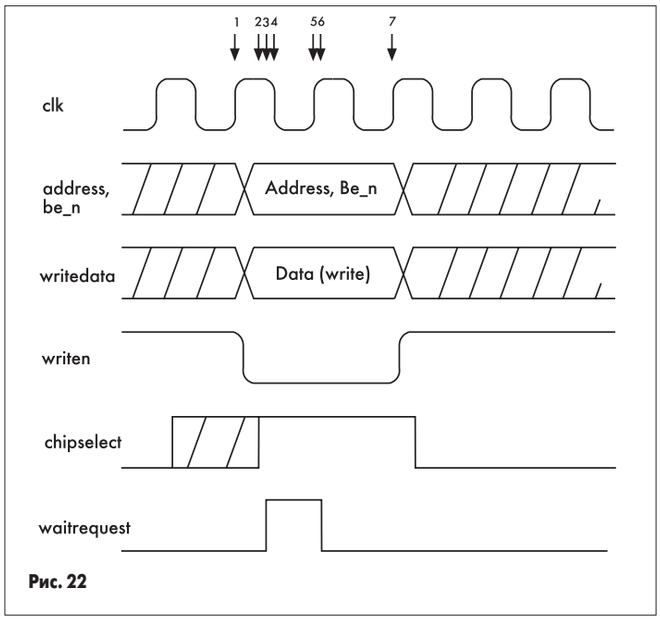

Рис. 22

Таким образом, данные, записываемые в асинхронное устройство при такой операции будут испорчены.

Чтобы решить эту проблему, шина Avalon обеспечивает несколько параметров хранения, которые позволяют правильно записывать в асинхронные устройства (что показано в последующих примерах).

Итак, в этом случае только простой цикл записи с нулевым временем ожидания подходит для синхронных периферийных устройств.

Большинство «стандартных» периферийных устройств на кристалле (подобно авалоновскому UART и таймеру) запрашивает нулевые состояния ожидания при записи и одно состояние ожидания при чтении (см. 6.1.2).

6.2.2. Цикл записи с установленным состоянием ожидания

* 1 установленное состояние ожидания, показанное в этом примере.

* Нулевое время предустановки/хранения.

* Нет динамической установки размеров шины.

* Пример показывает запись данных в синхронное периферийное устройство (см. рис. 22).

1) Передний фронт синхрочастоты, определяющий начало цикла шины.

2) Все сформированные на регистре выводы от мастера достоверны.

3) Все декодированные сигналы сигнала выбора устройства достоверны.

4) Логика шины Avalon устанавливает сигнал waitrequest для мастера.

5) Мастер производит выборку waitrequest, определяет, что он установлен, ждет.

Внутренняя логика шины Avalon заканчивает только одно состояние ожидания.

6) Логика шины Avalon сбрасывает сигнал waitrequest для мастера.

7) Периферийное устройство получает запрашиваемый им 2-й шанс производить выборку данных, адреса и управления.

Эта диаграмма показывает цикл записи с установленными состояниями ожидания. Заметьте, что он идентичен циклу записи без состояний ожидания, за исключением того, что есть состояния ожидания.

Циклы записи с состояниями ожидания «peripheral_controlled» выглядят точно так же, за исключением того, что состояния ожидания в них управляются периферийным устройством.

6.2.3. Цикл записи с предустановкой и временами хранения

* Ноль установленных состояний ожидания, показанных в этом примере.

* В этом примере предустановка — 1 цикл синхрочастоты, хранение — 1 цикл синхрочастоты.

* Нет динамического установления размеров шины.

* Пример показывает запись данных в синхронное периферийное устройство (см. рис. 23).

1) Передний фронт синхрочастоты, определяющий начало цикла шины.

2) Все сформированные на регистре выводы от мастера достоверны.

3) Все декодированные сигналы выбора устройства достоверны.

4) Логика шины Avalon устанавливает сигнал waitrequest для мастера.

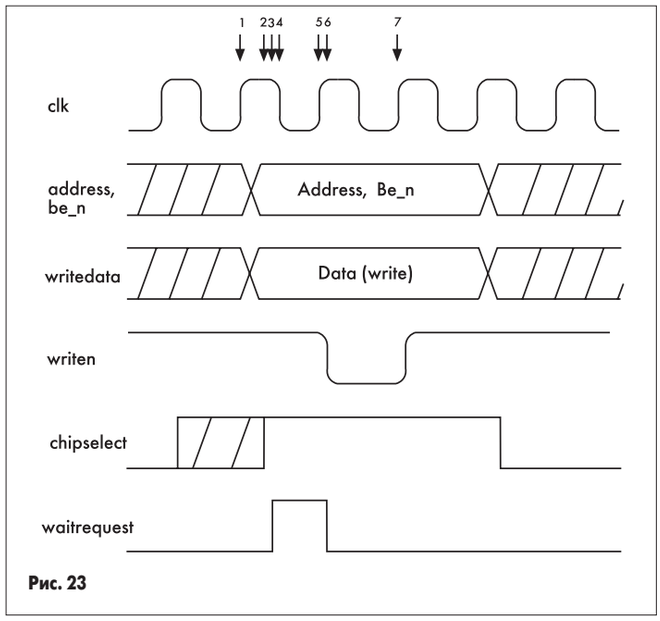

Рис. 23

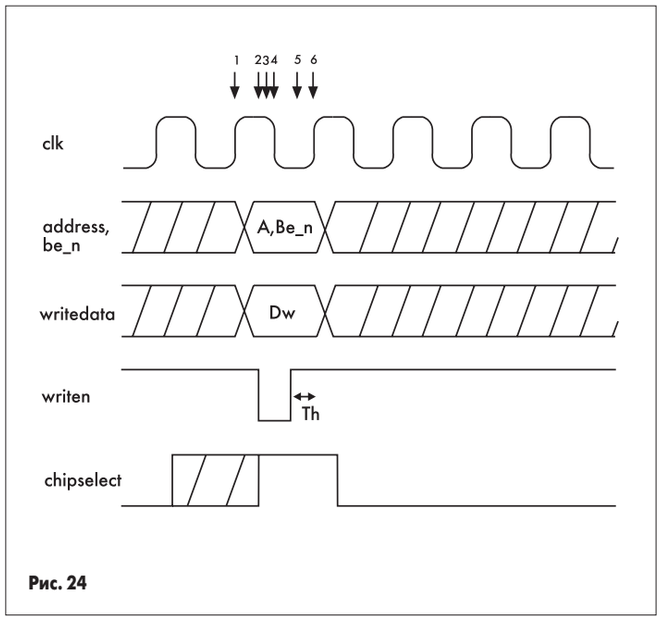

Рис. 24

5) Мастер производит выборку waitrequest, определяет, что он установлен, ждет.

Внутренняя логика шины Avalon заканчивает только одно состояние ожидания.

6) Логика шины Avalon сбрасывает сигнал waitrequest для мастера.

7) Периферийное устройство получает запрашиваемый им 2-й шанс производить выборку данных, адреса и управления.

Эта диаграмма показывает цикл записи с предустановкой и временем хранения. Циклы записи могут иметь только предустановку или только время хранения, и они могут не быть равными, как они показаны в этом примере.

Главный результат предустановки и хранения — узкий импульс записи.

Он может быть особенно удобен при записи данных в асинхронные устройства. Таким устройствам необходим хороший импульс записи, с чистым задним фронтом, чтобы произвести запись только нужного значения по желаемому адресу.

Единственный недостаток — цикл шины с 1 тактом предустановки и 1 тактом хранения будет длиться 3 цикла шины. Это надежное решение, но оно медленное.

6.2.4. Цикл записи со временем хранения в половину цикла синхрочастоты

* Состояние с нулевым временем ожидания.

* Нулевое время предустановки, время хранения — половина цикла синхрочастоты.

* Нет динамического установления размеров шины.

* Пример показывает запись данных в синхронное периферийное устройство (см. рис. 24).

1) Передний фронт синхрочастоты, определяющий начало цикла шины.

2) Все сформированные на регистре выводы от мастера достоверны.

3) Все декодированные сигналы выбора устройства достоверные.

Комбинаторный writen, выданный на периферийное устройство, достоверен (активен).

4) Задний фронт синхрочастоты заставляет логику шины Avalon переключиться и сократить writen.

5) Комбинаторный writen, выданный на периферийное устройство, достоверен (неактивен).

6) Конец цикла шины.

Slave может запрашивать половину цикла синхрочастоты для времени хранения, устанавливая в ptf назначение «Hold_Time» на «волшебное значение» «half_clock».

В этом случае шина Avalon генерирует специальный импульс writen этому slave, что заставляет логику переключиться на отрицательном фронте синхрочастоты.

Импульс записи для этого slave будет сокращен на половину цикла синхрочастоты, позволяя получить некоторое время предустановки без того, чтобы брать любые дополнительные циклы синхрочастоты.

Времена хранения в половину цикла синхрочастоты позволяют производить «чистые» операции записи с минимальным потерями быстродействия, но выбранное устройство должно иметь возможность работать с относительно узким импульсом записи (много асинхронных SRAM-устройств могут так работать).

Такая опция обычно выбирается наряду с сигналами выбора устройства типа «registeredselectn» для SRAM-устройств, используемых как основная память.