Простейший SDRAM-контроллер на VHDL

mindango Разное (/11-Blog) Создано: 13 ноября 2014 Просмотров: 38850

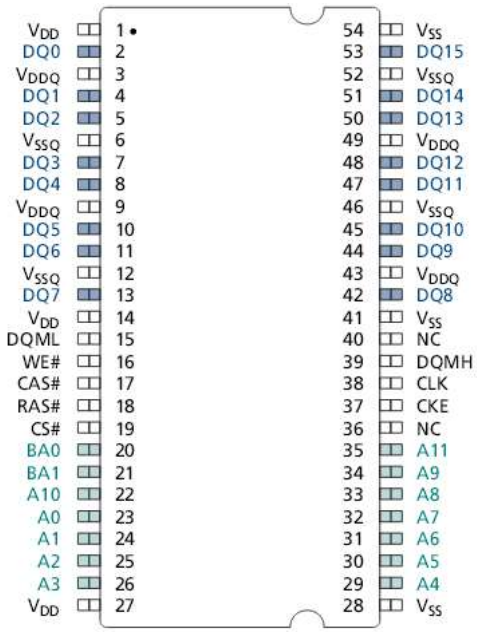


Рис. 1. Micron MT48LC4M16A2-75

Большинство современных микросхем программируемой логики имеет некоторый объём внутренней памяти, удовлетворяющий минимальные потребности разработчика. Например, в ПЛИС Cyclone III, на которой базируется плата **Марсоход2**, есть около 51 Кбайт собственной памяти. Этого достаточно, например, для реализации простого текстового VGA-режима, однако для полноценного кадрового буфера, который бы хранил информацию о цветовых величинах каждого отдельного пикселя, памяти ПЛИС явно мало. Тут на помощь разработчику приходят внешние запоминающие устройства различного типа.

На плате Марсоход2 установлена микросхема динамического ОЗУ (SDRAM) ёмкостью 8 Мбайт. К сожалению, по сравнению со статической памятью ПЛИС управлять этой микросхемой сложнее: требуется специальный контроллер, для самой SDRAM характерны задержки, которые необходимо учитывать разработчику, а если долго не обращаться к записанным в ОЗУ данным, они начинают деградировать и могут быть безвозвратно утеряны. Но так ли всё сложно на самом деле? В этой статье я попытаюсь рассказать о том, что представляет из себя динамическое ОЗУ и как им следует управлять на примере платы Марсоход2 и установленной на ней микросхемы Micron MT48LC4M16A2-75 (Рис. 1.).

Важным отличием памяти SDRAM от SRAM является другая система адресации. Если статическую память можно представить в виде очень длинного одномерного массива слов, то в динамическом ОЗУ используется матричная организация, то есть, слово выбирается по строке и столбцу. Для повышения быстродействия микросхема SDRAM обычно разделяется на несколько таких матриц, называемых "банками". Такой подход обусловлен тем, что некоторые операции могут занимать продолжительное время, поэтому контроллер может работать с одним банком, ожидая другого.

Таким образом, адрес слова складывается из количества банков, а также количества строк и столбцов в одном банке. У микросхемы MT48LC4M16A2-75 есть четыре банка (2 бита адресной шины), представляющих собой массив из 4096 на 256 слов (соответственно ещё 12 и 8 бит адресной шины), состоящих из 16 бит каждое. Что характерно, адрес передаётся в микросхему за несколько шагов: сначала отправляется адрес банка и строки, затем контроллер должен выждать некоторое время до того, как нужная строка будет активирована, и только тогда контроллер получает доступ к чтению и записи слов из этой самой строки. Перед активацией новой строки контроллер должен закрыть предыдущую. Поскольку на открытие и закрытие строки тратится время, для оптимизации быстродействия с активной строкой следует выполнить как можно больше операций (пакетная запись или пакетное чтение). Некоторые контроллеры умеют открывать несколько банков, что, как упоминалось выше, позволяет сократить время ожидания на промежуточные действия.

Если мы посмотрим на интерфейс нашей микросхемы SDRAM, мы обнаружим 12-разрядную шину, по которой передаются адреса строк и столбцов, а также двухразрядную шину указателя банка. Кроме того, особого внимания заслуживают сигналы wren, cas и ras (с активным нулём), по которым в микросхему передаются команды. О системе команд динамического ОЗУ следует рассказать отдельно.

Основными командами являются:

- **NOP –** "no operation", указывает микросхеме, что в настоящий момент никаких действий не требуется;
- **ACTIVE –** выбор банка и активация строки по установленному адресу;
- **READ –** начало чтения из активной строки, установленный адрес указывает на столбец; бит A10 даёт команду автоматического закрытия строки после завершения цикла чтения;
- **WRITE –** начало записи в активную строку, установленный адрес указывает на столбец; бит A10 даёт команду автоматического закрытия строки после завершения цикла записи;
- **PRECHARGE –** закрытие активной строки;
- **REFRESH –** обновление данных в ячейках памяти для предотвращения их деградации и необратимого повреждения;
- **LOAD MODE REGISTER –** эта команда позволяет загрузить в микросхему различные настройки (главным образом, задержки и длину пакета) посредством адресной шины.

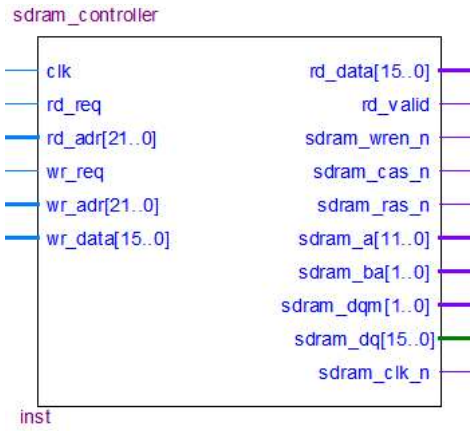
Важно, что команда NOP разрешена всегда, в то время как LOAD MODE REGISTER требует, чтобы все банки были неактивны, а после её выполнения микросхема на какое-то время будет недоступна для выполнения других операций. Также работает и команда REFRESH, а операции READ, WRITE и PRECHARGE могут выполняться только тогда, когда активен хотя бы один из банков. С более подробной информацией о системе команд SDRAM можно ознакомиться, например, в соответствующей статье википедии (https://ru.wikipedia.org/wiki/SDRAM) или в руководстве Micron к используемой микросхеме (

> Micron 64Mbit SDRAM 4x1Mx16bit (/downloads/category/18?download=122) (3717917 bytes)

).

Говоря о динамическом ОЗУ, нельзя не упомянуть о свойственных этой памяти задержках. Ключевой характеристикой зачастую называется CAS-задержка: время в тактах синхрочастоты от момента передачи контроллером микросхеме памяти адреса столбца до появления на информационной шине микросхемы соответствующих данных. Важно, что желаемая строка уже должна быть активна, иначе время задержки увеличивается. RAS-задержка характеризует время от обращения к памяти до момента появления данных на шине. Ещё одним показательным параметром является RC-задержка – время цикла, или характеристика того, как быстро может быть произведено два последовательных доступа к данным.

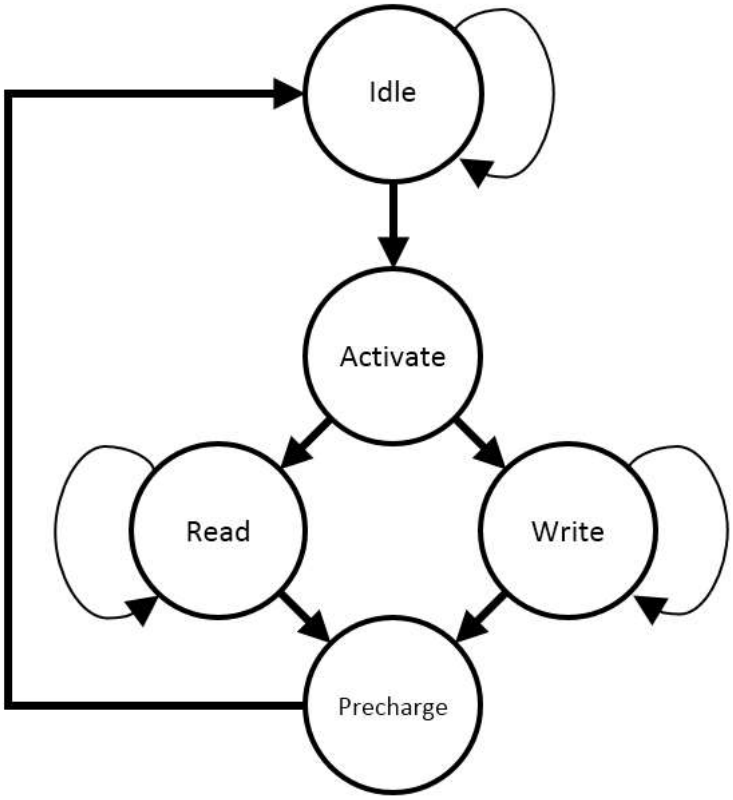
В изучение архитектуры SDRAM можно погрузиться значительно глубже, но изложенного выше уже достаточно для попытки создания простейшего контроллера (Рис. 2), что и является целью моего проекта. Контроллер должен обладать двумя главными качествами, а именно: простотой в использовании и поддержкой пакетной записи и пакетного чтения. Для начала попробуем разобраться с его интерфейсной частью:



Контроллер SDRAM. Рис.2.

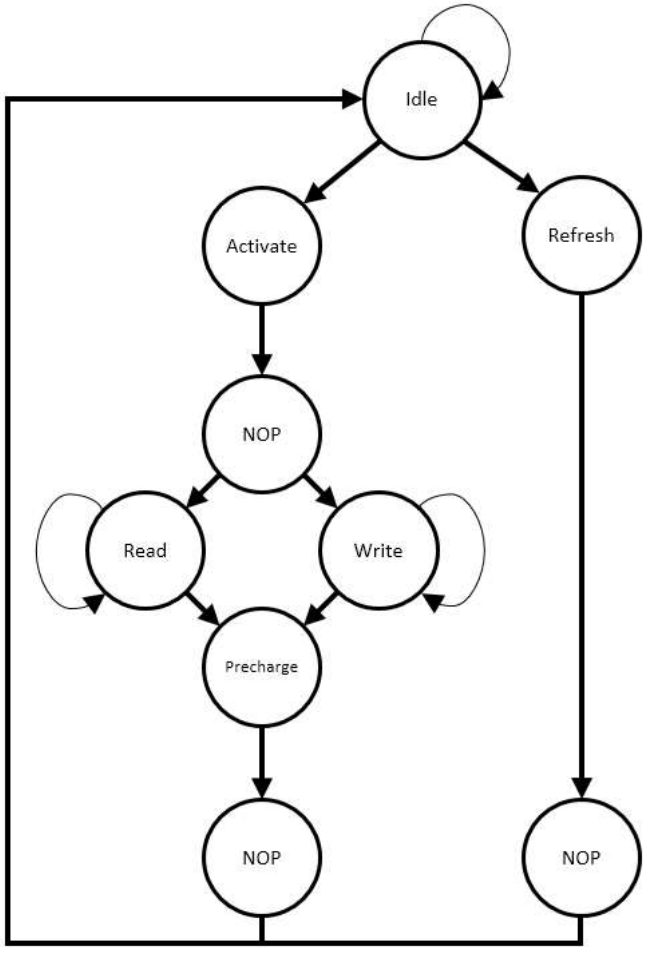
У контроллера обязательно должны быть входы адресных шин rd_adr и wr_adr шириной 22 разряда, шины записываемых данных wr_data (16 разрядов), запросов на чтение rd_req и запись wr_req, а также, разумеется, вход для сигнала синхрочастоты. Выходы – шина rd_data на 16 разрядов и сигнал rd_valid, высокий уровень которого показывает готовность новых данных. Плюс ряд выводов непосредственно на микросхему памяти динамического ОЗУ. Кстати говоря, шина данных sdram_dq двунаправлена, поэтому на стороне ПЛИС её следует переводить в высокоимпедансное состояние, когда не производится операций записи. Это позволяет беспрепятственно считывать данные с шины.

Работа контроллера будет представлена в виде автомата. Графически его наиболее простая форма выглядит так:



Граф работы SDRAM контроллера. Рис.3.

Но, если предполагается работа памяти на высокой частоте (например, не менее 100 МГц), для борьбы с задержками, возможно, придётся добавить в автомат новые состояния. Кроме того, если пользовательская логика не может гарантировать регулярный доступ ко всем данным ОЗУ, нужно предусмотреть периодический запуск команды регенерации. Таким образом, более полный граф будет таким:



Расширенный граф работы SDRAM контроллера. Рис.4.

Дабы не утомить читателя, я не стану приводить описание контроллера на VHDL.

Исходные данные этого проекта можно загрузить по этой ссылке:

> Простой SDRAM контроллер на VHDL (/downloads/category/14?download=159) (5966 bytes)

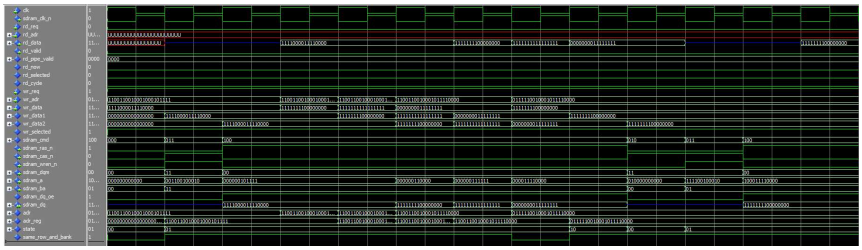
.

Между тем, чтобы убедиться в том, что логика контроллера устроена верно, его следует загнать в симулятор, и проследить за поведением различных сигналов. В качестве симулятора выступит приложение с удивительно "дружелюбным" и "интуитивно понятным" интерфейсом ModelSim, бесплатная версия которого входит в стандартный комплект поставки Quartus II.

Наверное, контроллер следовало бы рассматривать в совокупности с моделью памяти, используемой на плате Марсоход2 (соответствующее описание на языке Verilog предоставляет компания Micron – разработчик микросхемы). Но симулятор с этим оказался категорически не согласен, мотивируя свою точку зрения массой ограничений, свойственных бесплатной версии ModelSim. Я полагаю, доказательством работы контроллера будет установка правильных выходных сигналов на выводах, предназначенных для подключения к интерфейсу SDRAM, в зависимости от входных данных (запросов на чтение и запись, значений адресов, например). Если контроллер будет выдавать правильные сигналы, у нас не останется оснований полагать, что он не сможет управлять любой микросхемой динамического ОЗУ, соответствующей отраслевым стандартам.

На временных диаграммах ниже представлены все сигналы, существующие в контроллере – не только входные и выходные, но и ряд внутренних. Хотя иллюстрация особой наглядностью не отличается, она даёт возможность понять, что происходит внутри рассматриваемого объекта на каждом этапе его работы. Я проводил функциональную симуляцию, которая представляет собой не более чем тестирование логики.

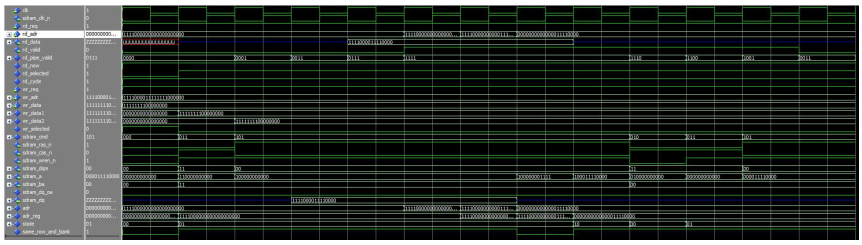
Первая ситуация – запрос на запись wr_req активен, на входах wr_data и wr_adr установлены какие-то величины, запрос на чтение rd_req отсутствует. Мы видим, что команда на запись и данные на выходе sdram_dq появляются спустя два такта после подачи первого тестового воздействия. При этом видно, что автомат внутри контроллера меняет своё состояние, открывая банк и строку по запрошенному адресу. Меняя на входе адрес и данные, мы получаем соответствующие изменения на выходе, причём на изменение адреса банка или строки контроллер реагирует должным образом.



(/images/stories/c3/mindango/p4/sdram_write.png)

Запись данных (кликнуть, чтобы увеличить изображение). Рис.5.

Теперь попробуем аналогичные воздействия, но с запросом на чтение. Кстати говоря, операциям чтения контроллер отдаёт приоритет, поэтому при активном запросе на запись и чтение исполняться будет именно последний. Также на диаграмме можно разглядеть, что на переключение банка и строки контроллер тратит два такта, а успеет ли микросхема отреагировать на запрос вовремя зависит уже от задержек конкретного решения.



(/images/stories/c3/mindango/p4/sdram_read.png)

Чтение данных (кликнуть, чтобы увеличить изображение). Рис.6.

Статья получилась несколько сумбурной и, вероятно, не слишком наглядной. Реальные испытания контроллера – в составе VGA-модуля с кадровым буфером – показали его работоспособность, но об этом проекте я расскажу в одной из следующих статей. Как обычно, проект можно обсудить в комментариях – с вашей помощью статьи могут стать интереснее и нагляднее.

P.S.: Читайте продолжение истории здесь: Симуляция контроллера SDRAM в ModelSim. (/11-blog/282-sdrammodelsim)

P.P.S.В статье, посвящённой симуляции моей реализации контроллера динамической оперативной памяти совместно с моделью микросхемы SDRAM, предоставленной компанией Micron, Николай обнаружил (/11-blog/282-sdrammodelsim) ряд нежелательных и потенциально критических ошибок. Все они вызваны тем, что в контроллере не предусмотрена процедура начального запуска микросхемы, которая подразумевает подачу определённой последовательности команд и загрузку настроек в регистр режима (Mode Register). Хотя работа с микросхемой возможна и без инициализации – настройки берутся из некоего состояния "по умолчанию" – нарушать рекомендации производителя всё же не стоит, поэтому я немного модернизировал контроллер, научив его инициализировать микросхему ОЗУ.

Таким образом, в автомате, на котором базируется работа контроллера, появилось ещё одно состояние. В нём контроллер находится в момент включения, и только после прохождения процедуры инициализации переходит в состояние idle (см. иллюстрации выше). Кроме того, появился настраиваемый параметр **sdram_frequency**, исчисляемый в МГц, на основе которого рассчитывается размерность счётчика, запускающего команды инициализации, и выход ready, высокое состояние которого указывает на то, что начальные процедуры выполнены. Пока на выходе ready сохраняется низкое логическое состояние, контроллер не будет воспринимать информацию с входов (за исключением тактового сигнала).

С процедурой инициализации можно ознакомиться в руководстве к микросхеме памяти. После исправлений ошибки, обнаруженные Николаем, больше не возникают. Обновлённую версию контроллера можно получить по этой ссылке (https://drive.google.com/file/d/0B-wpDy75OfZ1YlIJSGE0SWc3bnM/view?usp=sharing).

Добавлю ещё примерный алгоритм работы с контроллером:

- все операции возможны только после инициализации микросхемы (обнаруживается по высокому состоянию выхода **ready**);
- для записи данных нужно:
  1. установить запрос на запись (**wr_req** = '1'), убедившись, что неактивен запрос на чтение (**rd_req** = '0');
  2. одновременно подать адрес первого записываемого слова на вход **wr_adr** и само слово на **wr_data**;
  3. информация с входов **wr_adr** и **wr_data** при активном **wr_req** считывается каждый такт синхронизирующего сигнала;
  4. слово будет записано в память с задержкой на 2 такта после его "принятия" контроллером;
  5. при пакетной записи (в пределах одной строки) поток входной информации может быть непрерывным (каждый такт – новое слово и новый адрес);
  6. при изменении адреса банка или строки контроллер должен закрыть активную строку и открыть новую, поэтому в этом случае перед подачей новых данных следует выждать 3 такта;
- для чтения данных нужно:
  1. установить запрос на чтение (**rd_req** = '1'), активность запроса на запись в этом случае не играет роли;
  2. одновременно подать адрес **rd_adr** читаемого из памяти слова;
  3. адрес **rd_adr** считывается контроллером каждый такт синхронизирующего сигнала;
  4. готовность считанных из памяти данных на выходе **rd_data** обнаруживается по активности выхода **rd_valid**, задержка чтения составляет 4 такта;
  5. при пакетном чтении поток входной информации может быть непрерывным;
  6. при изменении адреса банка или строки контроллер должен закрыть активную строку и открыть новую, поэтому в этом случае перед подачей новых данных следует выждать 5 тактов.

P.P.P.S.: еще статьи на сайте https://marsohod.org (https://marsohod.org) связанные и использованием памяти SDRAM:

1) Фоторамка. Часть3. Фреймбуффер. (/projects/plata1/174-phframe3)

2) Тест SDRAM или Фреймбуффер2 (/projects/marsohod2/214-c3fb2).

3) Поключение SDRAM к системе на кристалле Amber (/m2/amber-arm-soc/24-ourblog/armsoc/224-2012-10-08-10-25-17)

Tags: SDRAM (/component/search/?searchword=SDRAM&ordering=&searchphrase=all) контроллер памяти (/component/search/?searchword=контроллер%20памяти&ordering=&searchphrase=all) VHDL (/component/search/?searchword=VHDL&ordering=&searchphrase=all) Altera (/component/search/?searchword=Altera&ordering=&searchphrase=all) Micron (/component/search/?searchword=Micron&ordering=&searchphrase=all)

Комментарии

#26 (/11-blog/281-sdramvhdl#comment-7231) **DROZD01** 13.10.2020 13:21 **0**
Здравствуйте! Пытаюсь применить ваш код для платы RZ-EasyFPGA A3.1 (EP4CE6E22C8)с чипом SDRAM HY57V641620FTP-H.

С вашим контроллером и вашим же тестбенчем (и "макетом" чипа памяти) не происходит команды чтения, а при записи откуда-то на портах чтения появляются записанные данные.

yadi.sk/i/LyTQFC9wLS9w0w (https://yadi.sk/i/LyTQFC9wLS9w0w)

#25 (/11-blog/281-sdramvhdl#comment-5960) **fadeev** 07.06.2019 10:03 **0**
На ваших платах входы DQML\DQMH подключены к ПЛИС, а можно ли их просто замкнуть на GND, тем самым освободим еще два вывода?
Будет ли работать, сильно затронет логику SDRAM контроллера?

#24 (/11-blog/281-sdramvhdl#comment-4934) **umarsohod** 20.12.2017 09:20 **0**

Цитирую lis:3k:

> Скажите пожалуйста, а возможно ли в общем случае использовать микросхему SDRAM в 8-ми или даже 4-х битном режиме? Мне нужно для экономии выводов это, из даташита по таблице истинности смотрел и не очевидно это.

Я использовал три. Здесь внимательно почитайте - marsohod.org/.../174-phframe3 (https://marsohod.org/projects/plata1/174-phframe3)

#23 (/11-blog/281-sdramvhdl#comment-4933) **lis:3k** 20.12.2017 08:02 **0**
Скажите пожалуйста, а возможно ли в общем случае использовать микросхему SDRAM в 8-ми или даже 4-х битном режиме? Мне нужно для экономии выводов это, из даташита по таблице истинности смотрел и не очевидно это.

#22 (/11-blog/281-sdramvhdl#comment-4133) **Николай Коврижных** 04.01.2017 09:47 **0**
Спасибо автору статьи. Благодаря нему с лёгкостью защитил курсовой по этой теме. НУК им. адмирала Макаророва 2016г.

#21 (/11-blog/281-sdramvhdl#comment-4050) **Flip-fl0p** 08.09.2016 13:50 **0**
Простите за некропост, но почему конечный автомат был реализован битовыми значениями регистра, а не машиной состояний ? Ведь по сути ничего не меняется...

#20 (/11-blog/281-sdramvhdl#comment-3816) **Vokonik** 19.10.2015 11:28 **0**
Здравствуйте уважаемые гуру. Не пинайте если мои вопросы покажутся глупыми... Я попытался использовать этот контроллер и столкнулся с такой проблемой. Я пишу данные по адресам 000h-003h и пробую прочитать данные по адресу 281h. Но почему-то чтения не происходит. Частота тактирования модуля 50МГц, частота захвата 200МГц. ЧТо я делаю не так? Принтскрин сигналтапа прилагаю.
yadi.sk/i/jKv25evjjqPuD (https://yadi.sk/i/jKv25evjjqPuD) Спасибо.

#19 (/11-blog/281-sdramvhdl#comment-3725) **Chaosorg** 11.06.2015 16:44 **0**
Я смотрю тут продолжают появляться вопросы в комментариях, но ведь есть же более свежая версия с регенерацией и сигналом ready - может стоит как-то в P.P.P.S. это отметить

Или как-то обозначить в каких случаях надо использовать эту версию

В комментариях ко второй статье пытался получить ответ на один вопрос - попробую теперь тут задать его иными словами:

Насколько быстрому SRAM можно приравнять эту память с контроллером при произвольном доступе ?
100MHz/5 = 20MHz или еще меньше?

Я пытаюсь понять какой максимальный пикселькклок можно будет позволить графическому контроллеру при имитации двухпортовой работы мультиплексированием. Один канал будет читать из соседних ячеек, а второй будет рушить пакетное чтение своими случайными обращениями.

#18 (/11-blog/281-sdramvhdl#comment-3724) **fpga1234** 10.06.2015 05:35 **0**
Сигнал CKE самими надо на 3.3v запитывать?

#17 (/11-blog/281-sdramvhdl#comment-3658) **Nedd** 20.04.2015 11:30 **0**
Возникла необходимость поменять разрядность на 24
Исправил wr_data, wr_data1, wr_data2, rd_data и sdram_dq на
(31 downto 0)
и получил ошибку Error (13076): The node "...|rd_data[16]" has multiple drivers due to the always-enabled I/O buffer "...|sdram_dq[16]".
Описание ошибки quartushelp.altera.com/.../... (http://quartushelp.altera.com/13.0/mergedProjects/msgs/msgs/emls_opt_tri_bus_multiple_drivers.htm)
Подскажите, пожалуйста в чем дело?

#16 (/11-blog/281-sdramvhdl#comment-3654) **Nedd** 17.04.2015 16:51 **0**
Спасибо за проделанную работу!
Несколько дней разбираюсь с различными реализациями sdram, и на мой взгляд, ваша работа лучшая, с точки зрения простоты, удобства пользования и функционала.

#15 (/11-blog/281-sdramvhdl#comment-3561) **alman** 15.01.2015 13:53 **0**

Цитирую mindango:

> To Alman
> Текущая версия контроллера не делает регенерацию. Я планировал её добавить, но, к сожалению, сейчас сильно занят на учёбе (сессия).
>
> Постараюсь на днях всё-таки выкроить немного времени и сделать хотя бы черновой вариант контроллера с регенерацией.

О, это было бы волшебно. Слишком много времени я потратил в попятках добиться стабильной работы.

А можно полюбопытсвовать, Вы случайно не родственник автора Demos Commander? Фамилия, конечно, распространённая, ну а вдруг?

И конечно масса пожеланий сдать сессию на отлично. Если не секрет, на каком курсе учитесь? Предложения о работе уже поступали?

#14 (/11-blog/281-sdramvhdl#comment-3558) **mindango** 14.01.2015 07:23 **+2**
To Alman

Текущая версия контроллера не делает регенерацию. Я планировал её добавить, но, к сожалению, сейчас сильно занят на учёбе (сессия).

Постараюсь на днях всё-таки выкроить немного времени и сделать хотя бы черновой вариант контроллера с регенерацией.

#13 (/11-blog/281-sdramvhdl#comment-3551) **alman** 09.01.2015 13:34 **0**
Что-то странное происходит с контроллером. Не могу понять где проблемы. Сделал простейший монитор, который позволяет загружать по X-modem в память файлы и прстейшую программу, выводящую дамп памяти на экран. Иногда в дамп попадают не те значения, которые записал - выглядит как внезапное появление "лишних" бит. Допустим, загрузил в SDRAM 4096 байт нулей, смотрю в мониторе - начинают появляться артефакты. Подозревал ошибку чтения, но артефакты всегда в определённых местах. Также заметил что со временем количество артефактов увеличивается - не может ли быть проблемы с регенерацией ОЗУ? Игрался с различными задержками чтения и записи, но проблему не решило.
Остался вариант попробовать использовать контроллер SDRAM из других проектов, но этот сильно приглянулся своей простотой.

Собственно вопрос - кто-нибудь использует активно этот контроллер? Замечали ли каклие-либо проблемы?

#12 (/11-blog/281-sdramvhdl#comment-3412) **alman** 25.11.2014 16:25 **0**

Цитирую mindango:

> To Alman: также я добавил в конец статьи примерный алгоритм работы с контроллером. Надеюсь, описал достаточно подробно и понятно.

Спасибо.
Кстати, Гугл неохотно отдал исходник. Ни разу такого не видел, чтобы гуглодиск так жадничал.

#11 (/11-blog/281-sdramvhdl#comment-3408) **mindango** 17.11.2014 06:22 **+2**
Я обновил контроллер: теперь он выполняет процедуру инициализации и конфигурирования микросхемы при начальном запуске. Это позволяет избавиться от ошибок, обнаруженных Николаем во время симуляции.

To Alman: также я добавил в конец статьи примерный алгоритм работы с контроллером. Надеюсь, описал достаточно подробно и понятно.

#10 (/11-blog/281-sdramvhdl#comment-3407) **nckm** 15.11.2014 16:01 **0**

Цитирую bfgroup:

Цитирую nckm:

> такой вопрос. По идее контроллер после включения должен инициализировать микросхему - а этого не видно по тексту контроллера. В документации на чип микросхемы четко прописана последовательность действий по инициализации:
> ...

ага, вот тоже стало интересно. главным образом, тут даже не в контроллере уважаемого mindango дело.
если схематик Марсоход2 выложенный здесь на сайте соответствует действительности, то сигнал SDRAM CKE наглухо привязан к 3.3V

в таком случае как, вообще, можно осуществить этот инит?

объясните, как амбер это делает?

вот так и делает - не управляет этим сигналом, он всегда в единице.. экономили пины микросхемы как могли

#9 (/11-blog/281-sdramvhdl#comment-3406) **bfgroup** 15.11.2014 14:52 **0**

Цитирую nckm:

> такой вопрос. По идее контроллер после включения должен инициализировать микросхему - а этого не видно по тексту контроллера. В документации на чип микросхемы четко прописана последовательность действий по инициализации:
> ...

ага, вот тоже стало интересно. главным образом, тут даже не в контроллере уважаемого mindango дело.
если схематик Марсоход2 выложенный здесь на сайте соответствует действительности, то сигнал SDRAM CKE наглухо привязан к 3.3V

в таком случае как, вообще, можно осуществить этот инит?

объясните, как амбер это делает?

#8 (/11-blog/281-sdramvhdl#comment-3405) **nckm** 15.11.2014 07:19 **0**

такой вопрос. По идее контроллер после включения должен инициализировать микросхему - а этого не видно по тексту контроллера. В документации на чип микросхемы четко прописана последовательность действий по инициализации:

The recommended power-up sequence for SDRAM:
1. Simultaneously apply power to VDD and VDDQ.
2. Assert and hold CKE at a LVTTL logic LOW since all inputs and outputs are LVTTL- compatible.
3. Provide stable CLOCK signal. Stable clock is defined as a signal cycling within tim- ing constraints specified for the clock pin.
4. Wait at least 100µs prior to issuing any command other than a COMMAND INHIB- IT or NOP.
5. Starting at some point during this 100µs period, bring CKE HIGH. Continuing at least through the end of this period, 1 or more COMMAND INHIBIT or NOP com- mands must be applied.
6. Perform a PRECHARGE ALL command.
7. Wait at least tRP time; during this time NOPs or DESELECT commands must be given. All banks will complete their precharge, thereby placing the device in the all banks idle state.
8. Issue an AUTO REFRESH command.
9. Wait at least tRFC time, during which only NOPs or COMMAND INHIBIT com- mands are allowed.
10. Issue an AUTO REFRESH command.
11. Wait at least tRFC time, during which only NOPs or COMMAND INHIBIT com- mands are allowed.
12. The SDRAM is now ready for mode register programming. Because the mode reg- ister will power up in an unknown state, it should be loaded with desired bit values prior to applying any operational command. Using the LMR command, program the mode register. The mode register is programmed via the MODE REGISTER SET command with BA1 = 0, BA0 = 0 and retains the stored information until it is pro- grammed again or the device loses power. Not programming the mode register upon initialization will result in default settings which may not be desired. Out- puts are guaranteed High-Z after the LMR command is issued. Outputs should be High-Z already before the LMR command is issued.
13. Wait at least tMRD time, during which only NOP or DESELECT commands are al- lowed.

At this point the DRAM is ready for any valid command.

#7 (/11-blog/281-sdramvhdl#comment-3404) **alman** 14.11.2014 20:55 **0**

Спасибо за ответы.

Цитирую mindango:

> To Alman
> 3. О каком сигнале "завершения записи" идёт речь?

К примеру, уставновил wr_addr, wr_data, wr_req. Как узнать что можно следущий адрес и данные устанавливать? Поэтому хотелось бы видеть нечто выходного сигнала wr_ack, чтобы записывать следующую порцию данных.

О! Ещё вопросы. Нужно ли сбрасывать wr_req при установке адреса и данных для следующей операции записи?

В принципе, для начала мне бы было достаточно точно знать как писать непрерывный поток данных. Например, нечто вроде такого:
1. Сброс wr_req
2. Установка wr_addr и wr_data
3. Установка wr_req
4. Ожидание три такта.
5. Переход на шаг 1.

Такой алгоритм будет рабочим?
Нет ли здесь какого-нибудь варианта для оптимизации?

Простите, если я назойлив и задаю слишком много вопросов.

#6 (/11-blog/281-sdramvhdl#comment-3403) **Alucard** 14.11.2014 19:56 **+1**

Цитирую mindango:

> To Alucard
>
> Я, конечно, пытался приблизить реализацию к контроллеру SRAM, но от задержек микросхемы SDRAM всё равно не избавиться – архитектура такая.
>
> Вы правильно поняли: достаточно установить запрос на чтение\запись, соответствующий адрес и данные (если пишем). Но нельзя забывать, что первые данные запишутся только на третий такт, а валидная информация на шине rd_data появится на пятый такт. Плюс надо учитывать, что на переключение строк тратится ещё какое-то время, поэтому одним беспрерывным потоком микросхему не записать и не прочитать.

Отлично! Я давно искал подобную вещь.
Спасибо.

#5 (/11-blog/281-sdramvhdl#comment-3402) **mindango** 14.11.2014 19:35 **0**
To Alucard

Я, конечно, пытался приблизить реализацию к контроллеру SRAM, но от задержек микросхемы SDRAM всё равно не избавиться – архитектура такая.

Вы правильно поняли: достаточно установить запрос на чтение\запись, соответствующий адрес и данные (если пишем). Но нельзя забывать, что первые данные запишутся только на третий такт, а валидная информация на шине rd_data появится на пятый такт. Плюс надо учитывать, что на переключение строк тратится ещё какое-то время, поэтому одним беспрерывным потоком микросхему не записать и не прочитать.

#4 (/11-blog/281-sdramvhdl#comment-3401) **Alucard** 14.11.2014 19:04 **0**
Спасибо большое за статью.
Я немного не понял диаграммы: этот контроллер позволит работать с SDRAM по принципу SRAM, то есть, устанавливаем адрес и получаем/записываем слово в адресованной ячейке?

#3 (/11-blog/281-sdramvhdl#comment-3400) **mindango** 14.11.2014 06:58 **+1**
To Alman

1. Можете использовать контроллер так, как вам угодно, но не забывайте указывать ссылку на эту статью.

2. В простоте все отличия. У меня не реализованы некоторые функции, которые можно было бы реализовать при необходимости. Вообще, с особенности реализации контроллера, взятого для проекта Amber, я не вникал, поэтому детально не могу сравнить.

3. О каком сигнале "завершения записи" идёт речь? Если это о команде Precharge, то, наверное, никак. Это команда микросхеме на закрытие активной строки с данными. Не закрыв одну строку, вы не получите доступ к другой – такова особенность архитектуры SDRAM.

4. Если нужна 32-разрядная шина данных, то достаточно просто увеличить разрядность соответствующих портов и сигналов. Например:

wr_data : in std_logic_vector(15 downto 0)
поменять на
wr_data : in std_logic_vector(31 downto 0)

То есть, вам нужно найти wr_data, wr_data1, wr_data2, rd_data и sdram_dq и изменить их разрядность.

5. Вы можете попробовать подключить контроллер как отдельный модуль. Но я не знаю, как это в Verilog делается.

#2 (/11-blog/281-sdramvhdl#comment-3399) **alman** 14.11.2014 02:01 **+1**
Какова лицензия этого устройства?
Какие отличия (кроме простоты и языка) от контроллера из проекта Amber Marsohod?
Как обойтись без сигнала завершения записи?
Как быть, если с VHDL незнаком, а хочется 32-х битные шины данных?

Как использовать это устройство из Verilog кода?

#1 (/11-blog/281-sdramvhdl#comment-3398) **tfwbbt** 13.11.2014 14:47 **0**
У меня одного первая картинка (с распиновкой) съехала вправо за границу шапки?

Обновить список комментариев

Добавить комментарий

Имя (обязательное)

E-Mail (обязательное)

Осталось: 1000 символов

Подписаться на уведомления о новых комментариях

Я не робот
reCAPTCHA
Конфиденциальность - Условия использования

Отправить

(https://vk.com/marsohod4you) (https://www.facebook.com/Marsohod4you/) (https://github.com/marsohod4you) (https://youtube.com/marsohod4you) (https://twitter.com/marsohod4you)