**gitЛабораторная работа - Системные шины. Шина APB.**

1. **Введение**

Системная шина в компьютере используется для передачи данных между различными компонентами системы, такими как процессор, память и периферийные устройства. Она представляет собой совокупность линий передачи сигналов, предназначенных для обмена данными между компонентами, расположенными на микросхеме или печатной плате.

В зависимости от их назначения, сигнальные линии системной шины можно классифицировать на три основные категории:

1. Линии, по которым передаются данные, называются шиной данных (ШД).

2. Линии, по которым передаются адреса, называются шиной адреса (ША).

3.Линии, по которым передаются управляющие сигналы, называются шиной управления (ШУ).

Такое разделение позволяет эффективно организовать обмен информацией: шина данных передает непосредственно информацию, шина адреса указывает, куда её передать, а шина управления координирует процесс обмена.

Как упоминалось ранее, основная задача системной шины заключается в обмене информацией. На наиболее простом уровне этот обмен происходит между единственным ведущим устройством (в данном примере — процессорным ядром), которое инициирует транзакции, и несколькими ведомыми устройствами (в данном примере — контроллерами), которые отвечают на эти запросы. Общая структурная схема подключения процессорного ядра к контроллерам через системную шину представлена на рисунке 1. На рисунке 2 показана обобщенная схема архитектуры системной шины, которая поможет лучше понять её организацию и принципы работы.

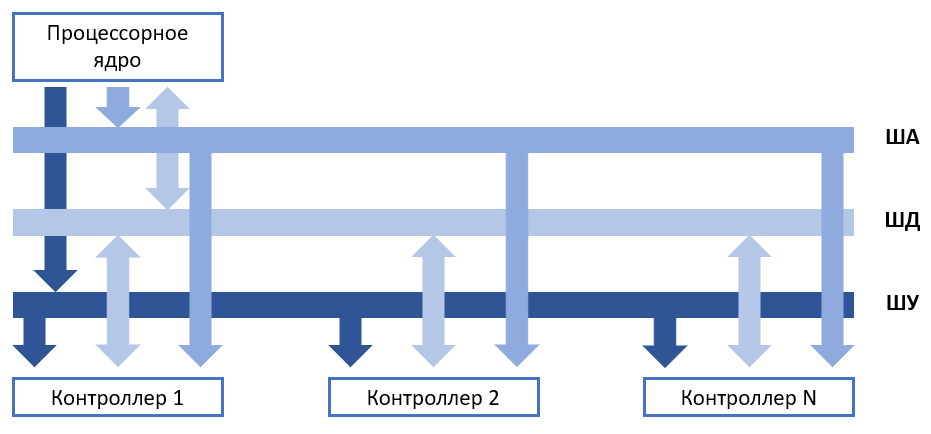


Рисунок 1 – Общая схема включения контроллеров к процессорному ядру.

Существуют различные термины, которые часто встречаются в спецификациях шин для систем и в технической литературе:

1. **Ведущее устройство (Master)** – это компонент, который инициирует процесс передачи данных для чтения или записи через системную шину, как, например, процессор или цифровой сигнальный процессор (ЦСП).

2. **Ведомое устройство (Slave)** – это устройство, которое не стартует обмен данными и отвечает только на запросы, поступающие от ведущего устройства.

**3.** **Декодер** – это логическое устройство, которое преобразует двоичный адрес, заданный ведущим устройством, в сигнал, активирующий конкретное ведомое устройство, которое должно получить данные.

**4. Мультиплексор** – это устройство, которое выбирает один из нескольких входных сигналов (например, данные от ведомых устройств) и передает его на выход. Управление мультиплексором осуществляется с помощью декодера, который определяет, какой сигнал должен быть выбран.

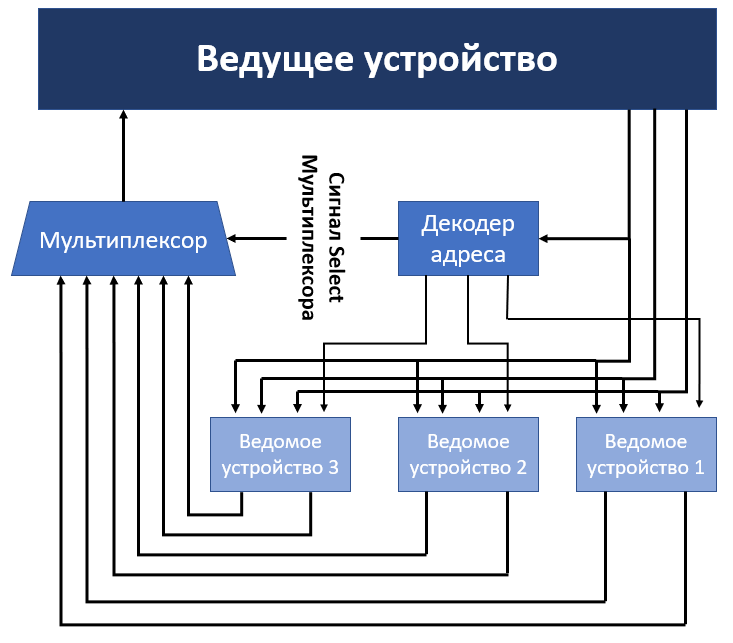


Рисунок 2 - Обобщенная схема архитектуры системной шины.

В процессе транзакции мастер выбирает периферийное устройство, указывает адрес и тип операции (чтение или запись), одновременно устанавливая управляющие сигналы, а затем ожидает ответ. Как только ведомое устройство готово, оно отправляет данные и сигнал готовности, после чего мастер считывает информацию и может инициировать новую транзакцию.

Стандартизированные системные шины активно применяются благодаря нескольким ключевым преимуществам, которые упрощают разработку и интеграцию компонентов:

1. Существуют многочисленные IP-блоки (IP — англ. intellectual property), которые можно подключать без изменения их исходных кодов, что делает проект легко масштабируемым. Ближайший аналог из области программирования – закрытые скомпилированные программные библиотеки.

2. Разработка программного обеспечения упрощается благодаря прямолинейной организации адресного пространства.

1. **Транзакции**

Транзакции на системной шине бывают трех типов, которые отличаются скоростью передачи и сложностью арбитража (управления):

1. Одиночная транзакция – Single;
2. Конвейерная транзакция – Pipeline;
3. Пакетная транзакция – Burst.

**1. Одиночная транзакция**

Это самый простой тип транзакции, при котором передается один блок данных (например, одно слово или одна команда) за один цикл шины. Диаграмма передачи одиночной транзакции представлена на рисунке 3.

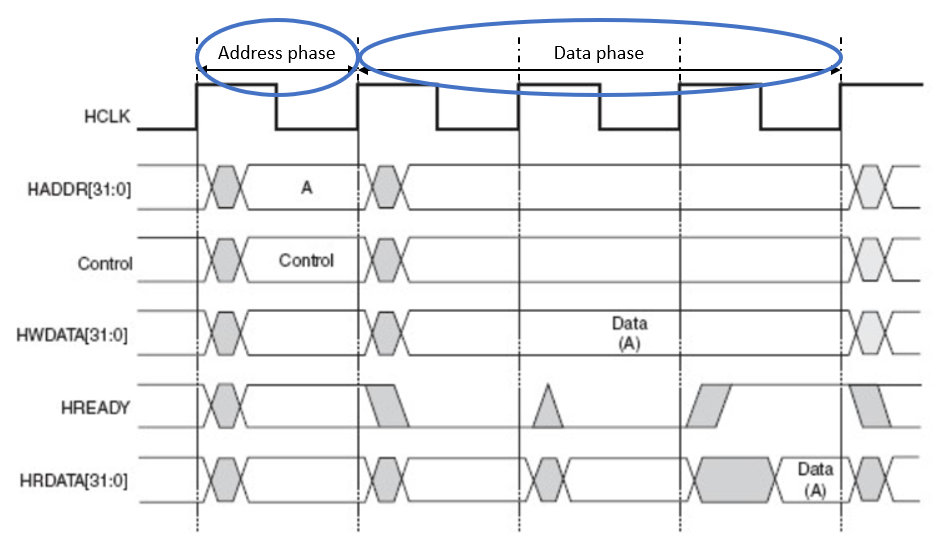


Рисунок 3 – Пример одиночной транзакции на шине.

**Особенности:**

* Каждая транзакция инициируется отдельно;
* Адрес и данные передаются в одном цикле;
* Простота арбитража: шина освобождается после завершения каждой транзакции.

**Преимущества:**

* Низкая сложность реализации.
* Подходит для задач, где требуется передача небольших объемов данных.

**Недостатки:**

* Низкая эффективность при передаче больших объемов данных, так как каждый раз требуется повторная инициализация транзакции.

**Пример использования:** Чтение или запись одного слова из/в память.

**2. Конвейерная транзакция**

В этом режиме транзакции выполняются в конвейерном режиме, то есть следующая транзакция может начинаться до завершения предыдущей, что позволяет увеличить пропускную способность за счет параллельной обработки. Диаграмма передачи конвейерной транзакции представлена на рисунке 4.

### 

### Рисунок 4 – Конвейерная транзакция.

**Особенности:**

* Шина используется более эффективно, так как несколько транзакций могут находиться на разных стадиях выполнения.
* Требуется более сложный арбитраж для управления конвейером.

**Преимущества:**

* Увеличение пропускной способности шины.
* Снижение задержек между транзакциями.

**Недостатки:**

* Повышенная сложность управления и синхронизации.
* Возможность возникновения конфликтов при доступе к ресурсам.

**Пример использования:** Передача данных между процессором и кэшем, где требуется высокая скорость и непрерывность.

**3. Пакетная транзакция**

В этом режиме за одну транзакцию передается несколько блоков данных (пакет) подряд, без необходимости повторной инициализации для каждого блока. Этот режим особенно эффективен при работе с блоками данных, такими как строки кэша или страницы памяти. Диаграмма передачи пакетной транзакции представлена на рисунке 5.

### 

### Рисунок 5 – Пакетная транзакция.

**Особенности:**

* Адрес указывается только один раз в начале транзакции, а данные передаются последовательно.
* Высокая скорость передачи данных за счет минимизации накладных расходов.

**Преимущества:**

* Максимальная эффективность при передаче больших объемов данных.
* Снижение нагрузки на шину и арбитраж.

**Недостатки:**

* Требуется поддержка со стороны устройств (например, памяти) для обработки пакетных транзакций.
* Более сложный арбитраж по сравнению с одиночными транзакциями.

**Пример использования:** Передача больших блоков данных между процессором и оперативной памятью.

Передача данных в транзакциях различается по скорости, сложности арбитража и способам реализации. Например, Pipeline рассматривается как компромиссный вариант между Single и Burst, позволяющий увеличить пропускную способность за счет параллельной обработки транзакций, что улучшает общую производительность системы. С другой стороны, Burst-транзакция подходит для передачи большого объёма информации, например, при записи или чтении внешней памяти, минимизируя накладные расходы за счет передачи нескольких блоков данных за одну инициализацию.

1. **Взаимодействие устройств по системной шине**

Рассмотрим пример взаимодействия по системной шине:

1. Ведущее устройство выбирает периферийное устройство и задаёт адрес на системной шине, добавляя к этому управляющие сигналы, такие как сигнал считывания.
2. Оно ожидает отклик от выбранного ведомого устройства, например, периферийного.
3. Как только ведомое устройство готово, оно отправляет необходимые данные ведущему устройству, одновременно устанавливая сигнал готовности на шине управления.
4. Затем мастер принимает эти данные и может начать следующую транзакцию.

Последовательная диаграмма обмена между ведущим и ведомым устройством представлена на рисунке 6.

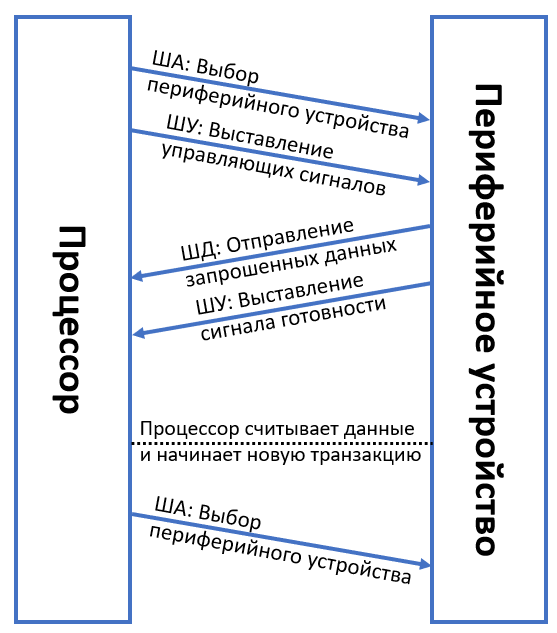


Рисунок 6 – Последовательная диаграмма обмена данными между ведущим устройством – процессором и ведомым устройством – периферийным блоком.

1. **Классификация системных шин и существующие стандарты**

Современные шины различаются по следующим параметрам:

1. **Разрядность** (8, 16, 32, 64 бита): Большее количество бит в шине позволяет передавать больше данных за один цикл чтения или записи. Разрядность адресной шины может быть определена отдельно от разрядности шины данных. Она указывает, сколько ячеек памяти может быть адресовано, что определяет максимальный объем адресуемой памяти.

2. **Метод передачи сигнала:** Шины могут использовать последовательный или параллельный метод передачи сигнала. Последовательные шины передают данные по одному биту за раз, а параллельные — по нескольким битам одновременно.

3. **Пропускная способность:** Ширина полосы пропускания, или пропускная способность, определяет количество данных, которое может быть передано по шине за определённый промежуток времени, и измеряется в битах в секунду (бит/с) или байтах в секунду (Б/с).

4. **Тип синхронизации:** Шины могут быть синхронными, передающими данные исключительно по тактовым импульсам, что требует наличия тактового сигнала для синхронизации, или асинхронными, передающими данные в любое время и использующими дополнительные сигналы для управления передачей.

Примеры стандартов шин:

- **PCI** (Peripheral Component Interconnect): Синхронная шина с высокой пропускной способностью.

- **USB** (Universal Serial Bus): Последовательная шина с поддержкой plug-and-play.

- **SATA** (Serial ATA): Последовательная шина для подключения накопителей.

1. **Стандарт AMBA.**

Advanced Microcontroller Bus Architecture (AMBA) — это открытый стандарт, разработанный компанией ARM, который определяет требования к внутрикристальным соединениям и служит для управления и соединения функциональных блоков в системах на кристалле (SoC). AMBA способствует развитию многопроцессорных систем с множеством контроллеров и периферийных устройств. Благодаря своей гибкости и масштабируемости, он активно применяется в ASIC и SoC, включая процессоры для современных мобильных устройств, таких как смартфоны.

Хотя AMBA изначально был разработан для микроконтроллеров, он стал стандартом де-факто для проектирования сложных систем на кристалле и нашел применение в различных областях, включая мобильные устройства, автомобильную электронику и IoT.

Существует несколько версий стандарта AMBA, каждая из которых вводила новые функции и улучшения для поддержки современных требований. На рисунке 7 показаны первые пять стандартов AMBA. В таблице 1 представлены самые популярные шины стандартов AMBA, включая их ключевые характеристики и области применения.

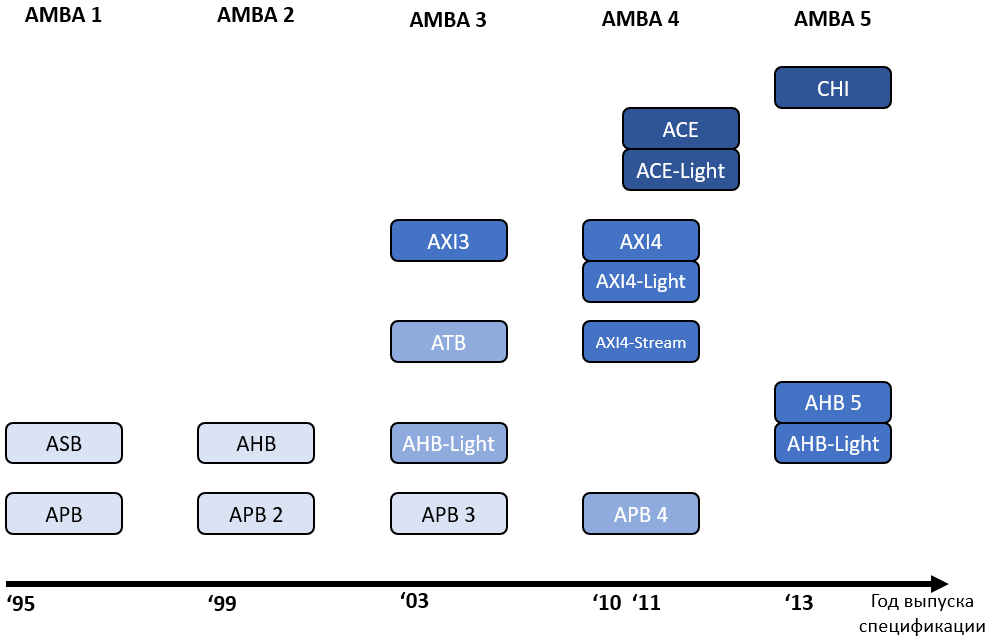


Рисунок 7 – Исторически первые пять стандартов AMBA.

Таблица 1 - Самые популярные шины стандартов AMBA.

|  |  |  |
| --- | --- | --- |
| Стандарт | Шина | Описание |
| AMBA1 | Advanced System Bus (ASB) | ASB (Advanced System Bus) представляет собой первое поколение системной шины AMBA, разработанное для высокопроизводительных систем. Она обеспечивает высокую пропускную способность и низкую задержку, что делает её подходящей для задач, требующих быстрого обмена данными. Эта шина поддерживает такие функции, как пакетная передача данных, конвейерная передача и режим мульти-мастера, при котором может быть более одного ведущего устройства, что позволяет нескольким устройствам конкурировать за доступ к шине.  Периферийные устройства с низкой пропускной способностью, такие как таймеры, интерфейсы ввода-вывода и другие низкоскоростные компоненты, обычно размещаются на шине APB (Advanced Peripheral Bus). |
| AMBA1, AMBA2, AMBA3, AMBA4 | Advanced Peripheral Bus (**APB**, APB2, APB3, APB4) | APB (Advanced Peripheral Bus), будучи частью иерархии AMBA, оптимизирована для минимизации энергозатрат и упрощения интерфейса. Она выступает в роли локальной вторичной шины, интегрированной в виде одного из ведомых устройств AHB или ASB, и используется для подключения периферийных устройств, которые не требуют высокой пропускной способности.  Мост, который связывает APB, функционирует как подчиненный модуль, преобразующий сигналы между высокоскоростной шиной (AHB/ASB) и низкоскоростной APB, и осуществляющий передачу управляющих сигналов для локальной периферийной шины. APB предназначена для подключения периферийных устройств с низкой пропускной способностью, таких как таймеры, интерфейсы ввода-вывода, GPIO и другие низкоскоростные компоненты, которые не требуют высокой производительности интерфейсной шины. |
| AMBA2, AMBA5 | Advanced High-performance Bus (AHB, AHB5) | AHB обозначает второе поколение шин AMBA и отвечает требованиям высокопроизводительных синтезируемых систем. Эта системная шина позволяет использовать несколько мастеров и предоставляет высокую пропускную способность. Основные функции AHB включают пакетные передачи, разделенные транзакции, однотактную арбитражную процедуру, возможность передачи без третьего состояния и расширенные конфигурации данных шины (64/128 бит). С её помощью можно создать мост между уровнями ASB/APB для легкой интеграции существующих систем. Она широко используется в разработках на базе ARM7, ARM9 и ARM Cortex-M. |
| AMBA3, AMBA5 | Advanced High-performance Bus Lite (AHB-Lite) | AHB-Lite удовлетворяет потребности в высокопроизводительных синтезируемых конструкциях, выступая в роли интерфейса для одного мастера шины и обеспечивая широкую полосу пропускания без арбитража. Функциональность AHB-Lite включает в себя пакетные передачи, реализацию без синхронизации и широкие конфигурации данных шины (64, 128, 256, 512 и 1024 бит). Чаще всего AHB-Lite работает с устройствами внутренней памяти, интерфейсами внешней памяти и высокопропускными периферийными устройствами. Устройства с низкой пропускной способностью могут быть ведомыми, но обычно они подключаются через APB для повышения производительности системы. |
| AMBA3, AMBA4 | Advanced Extensible Interface (AXI3, AXI4) | Протокол AXI разработан для высокопроизводительных систем с высокими частотами, подходящих для систем с широкими полосами и минимальными задержками. AXI позволяет обходиться без сложных мостов и поддерживает широкий спектр интерфейсов, включая контроллеры памяти с долгим начальным временем доступа. Он демонстрирует гибкость в создании межсоединений и сохраняет совместимость с уже существующими интерфейсами AHB и APB. Среди главных особенностей AXI — раздельные фазы адресных и управляемых данных, поддержка передачи данных без выравнивания через байтовые строб-управления, и раздельные каналы чтения и записи, обеспечивающие дешевый DMA-доступ. Протокол также поддерживает множество ожидающих адресов и позволяет завершать транзакции вне порядка. Он широко применяется в ARM Cortex-A процессорах, включая Cortex-A9. |
| AMBA4 | Advanced Extensible Interface 4 Lite (AXI4-Lite) | AXI включает в себя спецификацию AXI4-Lite, часть AXI4, что предназначена для взаимодействия с более простыми интерфейсами внутри компонентов. AXI4-Lite реализует транзакции длиной в 1 пакет, использует полную ширину шины данных и поддерживает 32-битную или 64-битную шину данных. Все обращения в AXI4-Lite не модифицируются и не буферизируются, а исключительные обращения не поддерживаются. |
| AMBA4 | AXI Coherency Extensions (ACE) | Протокол ACE расширяет возможности AXI4, добавляя средства для передачи данных широкой когерентности и поддержку аппаратно-согласованных кэшей, что позволяет множеству процессоров использовать общую память. Функции ACE включают барьерные транзакции, гарантирующие упорядочение в системе, и управление виртуальной памятью для распределенной виртуальной памяти. |
| AMBA4 | AXI Coherency Extensions Lite (ACE-Lite) | Интерфейс ACE-Lite представляет собой сокращённую версию полного ACE-интерфейса. Он применяется в ведущих компонентах, которые не имеют аппаратной когерентности кэша. Этот упрощённый вариант основан на интерфейсе AXI4, к которому добавлены дополнительные сигналы на адресных каналах чтения и записи. ACE-Lite исключает каналы: snoop-адресов и ответов, а также сигналы подтверждения операций чтения и записи и любые дополнительные битовые ответы на чтение. |
| AMBA4 | Advanced Extensible Interface 4 Stream (AXI4-Stream v1.0) | Протокол AXI4-Stream служит стандартным интерфейсом для соединения компонентов, обменивающихся данными. Он позволяет соединить одно устройство-источник данных с одним устройством-приёмником, но также способен поддерживать связь между большим числом ведущих и подчинённых узлов. Протокол способен управлять множеством потоков данных, используя один набор общих проводов, что помогает создать универсальную межсоединенную систему, выполняющую операции увеличения, уменьшения и маршрутизации. Интерфейс AXI4-Stream также поддерживает множество различных типов потоков, устанавливая правило связи между передачами данных и пакетами. |

В настоящее время эти протоколы считаются фактическими стандартами для встроенных процессоров, так как они хорошо задокументированы и могут использоваться без необходимости уплаты лицензионных сборов.

В данном лабораторной работе рассматривается только шина **APB** в виду ее простоты и высокой распространенности в различных микроконтроллерах и микропроцессоров общего назначения.

1. **Адресное пространство процессора**

При подключении разных периферийных устройств к системной шине возникает вопрос о способах их идентификации. Проблема решается путем присвоения каждому контроллеру собственного диапазона адресов, что позволяет избежать конфликтов при обращении к устройствам. Это позволяет процессору взаимодействовать с определенным контроллером, обращаясь по адресам из этого диапазона. В технической литературе существует специализированный термин – адресное пространство.

Весь доступный диапазон адресов, включая адреса для памяти и периферийных устройств, называется адресным пространством (Memory Map) системы на кристалле. Диапазон адресов, выделенный для отдельно взятого контроллера, называется адресным пространством контроллера. Базовым адресом контроллера называется адрес начала выделенного диапазона, который используется как точка отсчета для доступа к регистрам контроллера.

Структурная схема адресного пространства системы на кристалле, иллюстрирующая распределение адресов между различными компонентами системы, приведена на рисунке 8.

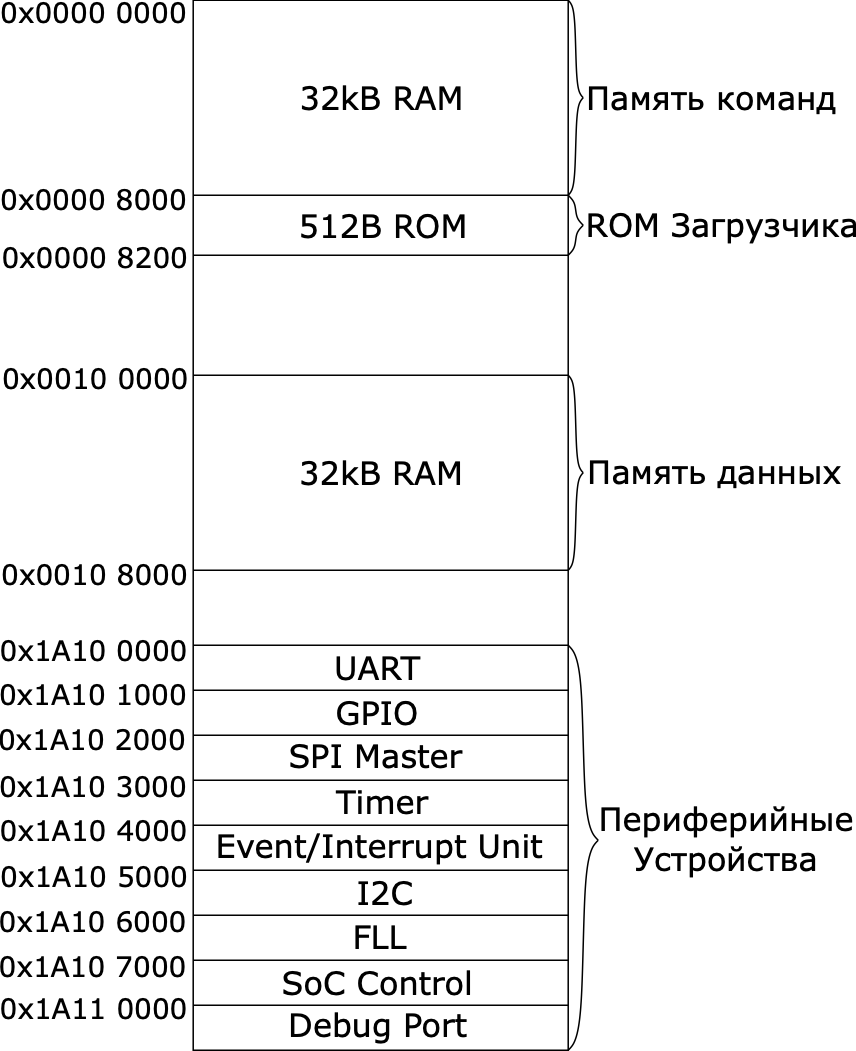


Рисунок 8 – Адресное пространство процессора.

Каждому периферийному устройству выделяется адресное пространство, определяемое его начальным (базовым) адресом и объемом, который зависит от количества регистров и их размера. Для осуществления обмена данными в периферийных устройствах, таких как контроллеры или вычислительные блоки, предусмотрен набор регистров, каждый из которых имеет свой уникальный адрес. Эти регистры используются для хранения данных, управления устройством и отображения его статуса.

Адресация регистров происходит относительно начального адреса. Абсолютный адрес регистра в адресном пространстве системы на кристалле рассчитывается как сумма базового адреса устройства/модуля и сдвига (смещения - offset) (относительного адреса). Например, если базовый адрес устройства равен 0x1000, а относительный адрес регистра — 0x04, то абсолютный адрес регистра будет 0x1004.

Особенности адресации в RISC-V

Поскольку RISC-V является архитектурой типа load-store, для доступа к ячейкам адресного пространства применяются инструкции load (для загрузки данных из памяти в регистры процессора) и store (для записи данных из регистров в память). Это разделение упрощает архитектуру и повышает эффективность выполнения операций.

Важно учитывать, что в RISC-V используется побайтовая адресация. Поскольку размер шины данных составляет 32 бита, доступ по адресу ноль приводит к чтению или записи первых четырёх байт (0,1,2,3). Это важно для обеспечения выравнивания данных и предотвращения ошибок при доступе к памяти. Данный аспект необходимо учитывать при разработке вычислительных блоков и назначении адресов регистров.

1. **Системная шина APB.**

Шина APB (Advanced Peripheral Bus) является частью линейки AMBA 3, разработанной компанией ARM. Это универсальный интерфейс, оптимизированный для подключения низкоскоростных периферийных устройств с минимальными накладными расходами. На рисунке 9 показана диаграмма системной шины APB, иллюстрирующая основные компоненты и сигналы, используемые для передачи данных.

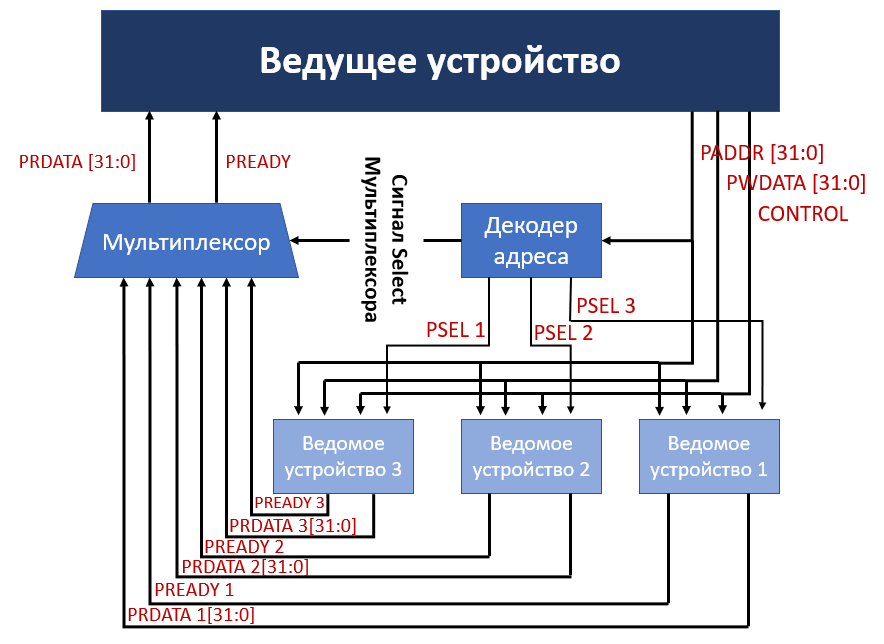


Рисунок 9 - диаграмма системной шины APB.

**Управление передачей данных**

Управлением декодера занимается главное устройство, которое выставляет адрес одного из подчиненных устройств на системную шину. По этому адресу декодер определяет, какое устройство выбрано для передачи данных, и формирует сигнал PSEL, который активирует выбранное устройство. Далее декодер генерирует сигнал Select для мультиплексора, который выбирает один из нескольких входных сигналов (данных от подключенных устройств) для передачи на системную шину.

**Стадии передачи данных**

Процесс передачи данных по шине APB включает две стадии:

1. **Стадия адресации**: Определяет устройство-адресат.

2. **Стадия передачи данных**: Осуществляет обмен информацией.

На выполнение стадии адресации всегда требуется один такт шины, тогда как стадия передачи данных может содержать состояния ожидания и продолжаться несколько тактов.

**Простой цикл записи без ожидания**

Простой цикл записи без ожидания осуществляется следующим образом. На стадии адресации, при положительном фронте тактового сигнала PCLK, управляющее устройство устанавливает следующие сигналы, необходимые для инициализации передачи данных:

- Адрес устройства-адресата **PADDR**;

- Сигнал записи **PWRITE** (активен на высоком уровне);

- Сигнал выбора устройства **PSEL** (активен на высоком уровне);

- Данные для записи **PWDATA**.

Потактовая диаграмма для общего случая работы шины APB представлена на рисунке 10.

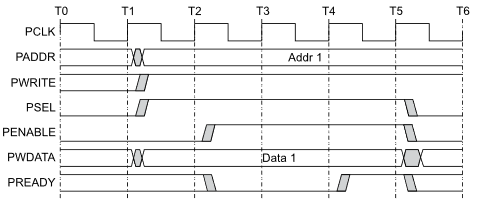


Рисунок 10 – диаграмма общего для работы шины APB циклов чтения без ожидания.

**Фаза передачи данных**

Состояния этих сигналов сохраняются и в фазе передачи данных. На втором фронте тактового импульса активируется сигнал PENABLE (высокий уровень активности), что подтверждает начало фазы записи информации. К следующему такту периферийное устройство обязано установить сигнал PREADY на высокий уровень, сигнализируя о готовности к приему передаваемой информации.

Как только сигнал PREADY активен, ведущее устройство на третьем такте деактивирует сигнал PENABLE. Сигнал PSEL также отключается, даже если последующее обращение будет направлено к тому же самому устройству. На этом завершается текущий цикл записи.

**Отсрочка записи**

Периферийное устройство способно отсрочить окончание записи, если при активном PENABLE не активирует сигнал PREADY, пока не завершит приём данных. Это позволяет устройству обработать данные перед завершением цикла. Завершение цикла записи произойдет на первом фронте тактового сигнала, когда будет зафиксирован активный уровень сигнала PREADY.

**Циклы чтения**

Графики временных диаграмм для циклов чтения без ожидания и с ожиданием выглядят схоже, но иллюстрируют ключевые различия в работе шины. На изображении 11 представлена диаграмма цикла без ожидания, а на рисунке 12 — диаграмма с ожиданием.

Главное различие заключается в том, что на этапе адресации сигнал PWRITE находится в низком состоянии, указывая на операцию чтения. В этой ситуации, когда сигнал PENABLE активен, ведомое устройство обязано выставить данные на шину PRDATA, сопровождая их активным состоянием сигнала PREADY.

Если периферийное устройство хочет задержать процесс чтения, оно должно деактивировать сигнал PREADY при активном состоянии PENABLE. Тогда ведущее устройство перейдет в режим ожидания до тех пор, пока снова не получит активный уровень PREADY, что позволяет устройству подготовить данные для чтения.

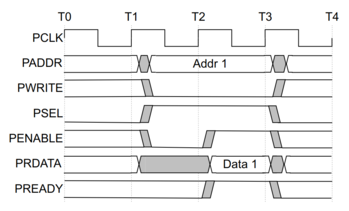


Рисунок 11 - Временная диаграмма чтения без состояния ожидания.

На рисунке 12 представлена временная диаграмма чтения с ожиданием.

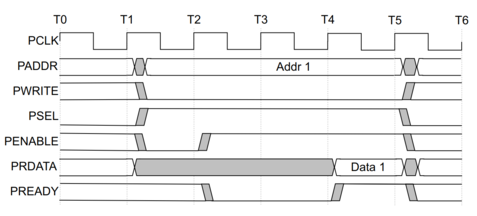


Рисунок 12 - Временная диаграмма чтения с ожиданием.

Таблица 2 - Сигналы системной шины APB:

|  |  |  |
| --- | --- | --- |
| Сигнал | Тип | Описание |
| PCLK | Вход | Тактовый сигнал для синхронизации. Все действия по шине APB происходят по переднему фронту сигнала PCLK. |
| PRESETn | Вход | Сигнал сброса (активный низкий) |
| PADDR | Вход | Шина адреса. Может иметь разрядность до 32 бит. Поступает от ведущего устройства. |
| PSELx | Вход | Выбор ведомого устройства. Ведущее устройство вырабатывает этот сигнал отдельно для каждого периферийного устройства. Этот сигнал показывает, что ведомое устройство выбрано для обмена данными. |
| PENABLE | Вход | Разрешение работы. Этот сигнал указывает второй и следующие такты передачи по APB(фаза данных). |
| PWRITE | Вход | Направление передачи (0-чтение, 1-запись) |
| PWDATA | Вход | Шина данных для записи. Может иметь разрядность до 32 бит. |
| PREADY | Выход | Сигнал готовности периферийного устройства. |
| PRDATA | Выход | Шина данных для чтения. Может иметь разрядность до 32 бит. |
| PSLVERR | Выход | Ошибка на шине. Периферийные устройства не обязаны его поддерживать. Если сигнал не используется, то не соответствующий вход ведущего устройства подается низкий уровень. |

1. **Практические пример использование шины APB**

В данной лабораторной работе рассматривается 2 дизайна таймеров:

* + - 1. Таймер – счетчик EF\_TCC32 доступный в гит репозитории: [https://github.com/YarosLove91/EF\_TCC32.git](https://github.com/YarosLove91/EF_TCC32.git%20) (ветка **apb\_timer\_pwm**)

Сам таймер описан на языке verilog в файле: **EF\_TCC\_32/hdl/rtl/EF\_TCC32.v**

Низкоуровневый файл-адаптер представлен в **EF\_TCC\_32/hdl/rtl/bus\_wrappers/EF\_TCC32\_apb\_PWM\_tb.v**

Сценарные тесты находятся в: **EF\_TCC\_32/verify/utb**

Схема модуля таймера-счетчика и регистры таймера описаны в файле **EF\_TCC\_32/README.md**

Драйвера для управления таймером расположены в EF\_TCC32/fw

* + - 1. Таймер реального времени — RTC доступный в гит репозитории:  
         <https://github.com/YarosLove91/rtc.git> (ветка sync\_repo)

Код таймера представлен в директории **rtc/rtl**

Файл – адаптер шины APB представлен в **rtc/rtl/bus\_wrappers/rtc\_apb.sv**

Сценарный тест представлен в: **rtc/tb/tb\_rtc.sv**

Высокоуровневый тест для среды verilator представлен в: **rtc/tb\_verilator**

3. Модуль таймеров «в сборе» на основе таймеров EF\_TCC32 и RTC доступен по адресу:

<https://github.com/YarosLove91/Lab_System_bus/tree/periphery-top> (ветка periphery-top).

Представляет собой модуль в сборе и используется только для решения дополнительного задания № 2 для получения максимальной оценки 10.

В составе дизайна есть 2 сценарных теста для среды verilator:

1) Тест на доступ к регистрам

Файл — Lab\_System\_bus/blob/periphery-top/tb/regs\_access\_test.svh содержит адаптер позволяющий одновременно взаимодействовать с таймером EF\_TCC32 и RTC (они внутри periphery\_top).

Тест демонстрирует работу демультиплексора APB по адресам, т.е. показывает, что можно получить доступ до разных таймеров через общий модуль и что у регистров есть разные типы.

Запускается из корня репозитория командой make apb\_top\_tb

2) Тест на RTC

Файл Lab\_System\_bus/tree/periphery-top/tb/rtc\_tb в составе periphery\_top.

Тест повторяет сценарий отдельного теста на RTC (<https://github.com/YarosLove91/rtc/blob/sync_repo/tb_verilator/main.cpp>). Тест демонстрирует запись в регистры и дает возможность посмотреть, что счётчик внутри RTC функционирует. Запускается из корня репозитория командой make rtc\_tb

**Задание к лабораторной работе**

В рамках основной части задания данной лабораторной работы требуется:

1. Ознакомиться с теоретическим материалом лабораторной работы;

2. Изучить принцип работы системной шины APB;

3. Исследуйте исходные коды представленных модулей – Таймер EF\_TCC32 и RTC;

4. Смоделируйте работу устройств **без шины APB**;

5. Смоделировать работу таймера EF\_TCC32 с низкоуровневым адаптером шины APB (EF\_TCC32\_apb\_PWM\_tb.v).

6. Запустите таймер EF\_TCC32 (EF\_TCC32\_apb\_PWM\_tb.v) с настройками согласно вариантам заданий, которые представлены в таблице 3.

6.1 Таймер должен отсчитать заданное время и по его истечению сигнализировать, выставив флаг прерывания.

6.2 Таймер должен генерировать сигнал ШИМ с частотой в кГц, которая равна числу Вашего дня рождения (1…31).

7. Оформите отчет. Отчет должен содержать вывод консоли и диаграммы логических уровней шины APB.

**Дополнительное задание №1.**

Разработайте сценарный тест (тестбенч) для высокоуровневого адаптера дизайна RTC. Докажите, что дизайн RTC с адаптером APB (rtl/bus\_wrappers/rtc\_apb.sv) функционирует должным образом.

Установите будильник на свою дату и время рождения и запустите отсчет времени начиная с полуночи за 3 суток до Вашего дня рождения. В случае корректно работающего дизайна должен производится счет времени, даты и по достижению времени срабатывает будильник.

**Дополнительное задание №2.**

Выполните лабораторную работу используя модуль таймеров в сборе. При выполнений данного дополнительного задания. Выполнение данного задания оценивается в 10 баллов.

Основная часть задания оценивается максимум в 6 баллов. Дополнительная часть задания оценивается в 2 балла.

Таблица 3 – Варианты заданий.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № варианта | Направление счета | Время срабатывания таймера, сек | № варианта | Направление счета | Время срабатывания таймера, сек |
| **1** | UP | 12 | **11** | UP | 32 |
| **2** | Down | 14 | **12** | Down | 13 |
| **3** | UP | 16 | **13** | UP | 15 |
| **4** | Down | 18 | **14** | Down | 17 |
| **5** | UP | 20 | **15** | UP | 19 |
| **6** | Down | 22 | **16** | Down | 21 |
| **7** | UP | 24 | **17** | UP | 23 |
| **8** | Down | 26 | **18** | Down | 25 |
| **9** | UP | 28 | **19** | UP | 27 |
| **10** | Down | 30 | **20** | Down | 29 |

## Контрольные вопросы

* Дайте определение системной шины. Назначение системной шины.
* Классификация системных шин.
* Каким образом различаются контроллеры в адресном пространстве? Что такое базовый адрес контроллера?
* Системная шина APB. Каково назначение каждого из сигналов системной шины?
* Системная шина APB. Изобразить цикл записи с задержкой.
* Системная шина APB. Изобразить цикл чтения без задержки.
* Таймеры — счетчики. Таймеры реального времени.