**23.1 Введение**

**23.1.1 Особенности DCAN**

Общие характеристики контроллера DCAN следующие:

- Поддержка протокола CAN версии 2.0 часть A, B (ISO 11898-1)

- Скорость передачи данных до 1 МБит/с

- Двойной источник синхронизации

- 16, 32, 64 или 128 объектов сообщений (в данном устройстве инстанцируется как 64)

- Индивидуальная маска идентификатора для каждого объекта сообщения

- Программируемый режим FIFO для объектов сообщений

- Программируемые режимы обратного цикла для самотестирования

- Режим приостановки для поддержки отладки

- Программный сброс модуля

- Автоматическое включение шины после состояния Bus-Off по программируемому 32-битному таймеру

- Механизм проверки четности оперативной памяти сообщений

- Прямой доступ к ОЗУ сообщений в тестовом режиме

- Выводы CAN Rx / Tx конфигурируются как выводы ввода-вывода общего назначения

- Две линии прерываний (плюс дополнительная линия прерывания по ошибке четности)

- Инициализация оперативной памяти

- Поддержка DMA

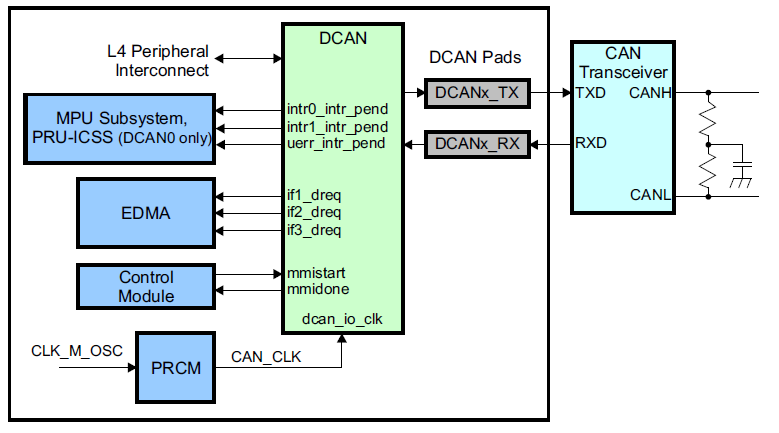
**23.1.2 Неподдерживаемые функции DCAN**

Модуль DCAN в этом устройстве не поддерживает режим работы с выводами GPIO. Вся функциональность GPIO отображается через GPIO-модули и муссируются на пины. Управляющие сигналы пинов GPIO от модулей DCAN не подключены.

**23.2 Интеграция**

CAN - это последовательный протокол связи, который эффективно поддерживает распределенное управлениe в реальном времени с высоким уровнем безопасности. Модуль DCAN поддерживает скорость передачи данных до 1 Мбит/с и соответствует спецификации протокола CAN 2.0B. Ядро IP для DCAN предоставлено компанией Bosch.

Это устройство включает в себя два варианта контроллера DCAN: DCAN0 и DCAN1. На рисунке 23-1 показанаинтеграция модуля DCAN.



**Рисунок 23-1. Интеграция DCAN**

**23.2.1 Атрибуты подключения DCAN**

Общие атрибуты подключения для модуля DCAN приведены в таблице 23-1.

**Таблица 23-1. Атрибуты подключения DCAN**

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Power Domain | Peripheral Domain |
| Clock Domain | PD\_PER\_L4LS\_GCLK (OCP)  PD\_PER\_CAN\_CLK (Func) |
| Reset Signals | PER\_DOM\_RST\_N |
| Idle/Wakeup Signals | Smart Idle |
| Interrupt Requests | 3 Interrupts per instance  Intr0 (DCANx\_INT0) – Error, Status, Msg Object interrupt  Intr1 (DCANx\_INT1) – Msg Object interrupt  Uerr (DCANx\_PARITY) – Parity error interrupt  All DCAN0 interrupts to MPU Subsystem and PRU-ICSS  All DCAN1 interrupts to only MPU Subsystem |
| DMA Requests | 3 DMA requests per instance to EDMA (CAN\_IFxDMA) |
| Physical Address | L4 Peripheral slave port |

**23.2.2 Управление тактовыми импульсами и сбросом DCAN**

Контроллеры DCAN имеют разделенные интерфейс шины и функциональное тактирование.

**Таблица 23-2. Сигналы тактового генератора DCAN**

|  |  |  |  |
| --- | --- | --- | --- |
| **Clock Signal** | **Max Freq** | **Reference / Source** | **Comments** |
| DCAN\_ocp\_clk  Interface clock | 100 MHz | CORE\_CLKOUTM4 / 2 | pd\_per\_l4ls\_gclk  from PRCM |
| DCAN\_io\_clk  Functional clock | 26 MHz | CLK\_M\_OSC | pd\_per\_can\_clk  from PRCM |

**23.2.3 Список выводов DCAN**

Внешние сигналы модуля DCAN приведены в следующей таблице.

**Таблица 23-3. Список выводов DCAN**

|  |  |  |
| --- | --- | --- |
| **Pin** | **Type** | **Description** |
| DCAN*x*\_TX | O | DCAN transmit line |
| DCAN*x*\_RX | I | DCAN receive line |

**23.3 Функциональное описание**

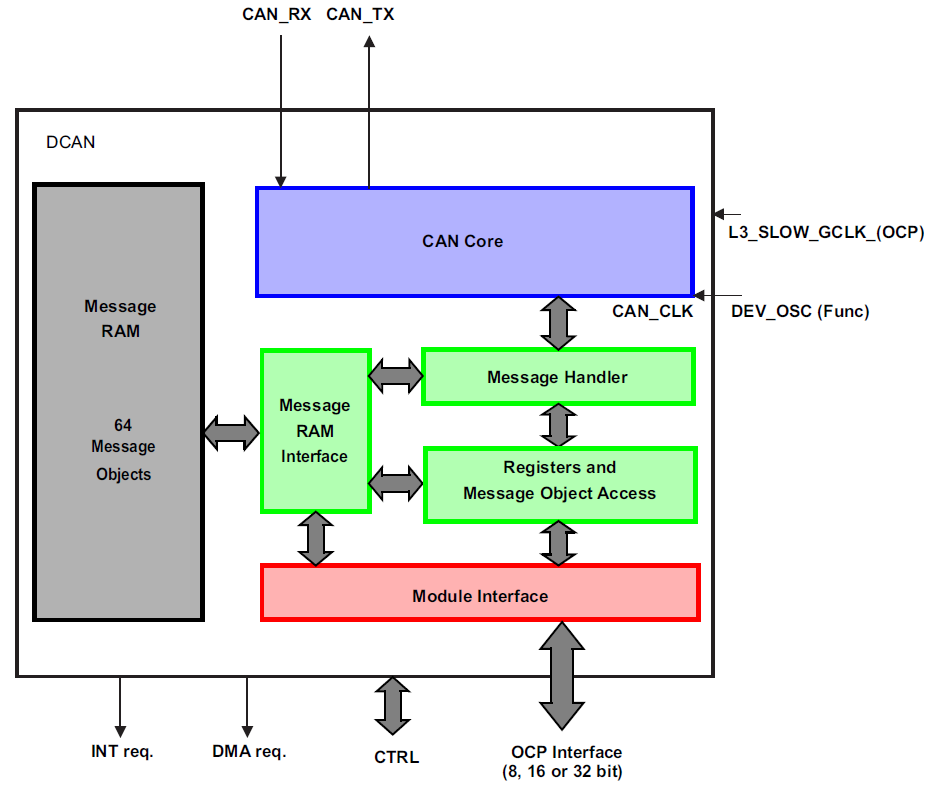
Модуль DCAN осуществляет обмен данными по протоколу CAN в соответствии с ISO 11898-1. Скорость передачи данных может быть запрограммирована на значения до 1 МБит/с. Для подключения к физическому уровню (шине CAN) требуется дополнительное приемопередающее оборудование физического уровня (шиной CAN).

Для связи в сети CAN можно сконфигурировать отдельные объекты сообщений. Объекты сообщений и маски идентификаторов хранятся в оперативной памяти сообщений. Все функции, связанные с обработкой сообщений, реализованы в обработчике сообщений. К таким функциям относятся фильтрация приема, передача сообщений между CAN-ядром и оперативной памятью сообщений(RAM), обработка запросов на передачу, а также генерация прерываний или запросов DMA.

Набор регистров модуля DCAN может быть доступен непосредственно центральному процессору через интерфейс модуля.Эти регистры используются для управления и конфигурирования CAN-ядра и обработчика сообщений, а также для доступа к оперативной памяти сообщений.

На рисунке 23-2 показана блок-схема модуля DCAN, а его особенности описаны ниже.

**Рисунок 23-2. Блок-схема DCAN**



**23.3.1 Ядро CAN**

Ядро CAN состоит из контроллера протокола CAN и регистра сдвига Rx/Tx. Оно обрабатывает все функции протокола ISO 11898-1.

**23.3.2 Обработчик сообщений**

Обработчик сообщений - это машина состояний, которая управляет передачей данных между однопортовым ОЗУ сообщений и сдвиговым регистром Rx/Tx CAN-ядра. Он также обрабатывает фильтрацию приема и генерацию запросов прерывания/DMA, как запрограммировано в регистрах управления.

**23.3.3 ОЗУ сообщений**

DCAN0 и DCAN1 позволяют хранить 64 сообщения CAN.

**23.3.4 Интерфейс ОЗУ сообщений**

Три набора интерфейсных регистров управляют доступом процессора к ОЗУ сообщений для чтения и записи. Имеются два набора интерфейсных регистров для доступа на чтение и запись, IF1 и IF2, и один набор интерфейсных регистров для доступа только на чтение, IF3. Дополнительную информацию можно найти в разделе 23.3.15.12.

Интерфейсные регистры имеют ту же длину слова, что и ОЗУ сообщений.

**23.3.5 Доступ к регистрам и объектам сообщений**

Согласованность данных обеспечивается косвенным доступом к объектам сообщений. Во время нормальной работы все доступы процессора и DMA к ОЗУ сообщений осуществляются через интерфейсные регистры. В специальном тестовом режиме, ОЗУ сообщений отображается на память и может быть напрямую доступно как процессору, так и DMA.

**23.3.6 Интерфейс модуля**

Доступ к регистрам модуля DCAN осуществляется центральным процессором или пользовательским программным обеспечением через 32-битный интерфейс периферийной шины.

**23.3.7 Двойной источник тактовых импульсов**

Модуль DCAN имеет два тактовых домена: периферийный синхронный тактовый домен

(L3\_SLOW\_GCLK) и домен периферийного асинхронного источника тактовых импульсов (CLK\_M\_OSC) для CAN\_CLK.

**23.3.8 Работа с CAN**

После аппаратного сброса бит Init в регистре управления CAN (CTL) установлен, и все функции протокола CAN отключены. Перед началом работы модуль CAN необходимо инициализировать. На рисунке 23-3 показан основной основной процесс инициализации модуля CAN.

**23.3.8.1 Инициализация модуля CAN**

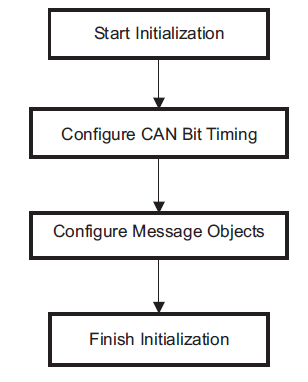
Общая инициализация CAN-модуля включает в себя следующие два важных этапа:

- Конфигурация синхронизации битов CAN

- Конфигурация объектов сообщений

Для инициализации CAN-контроллера процессор должен настроить синхронизацию битов CAN и те объекты сообщений, которые должны использоваться для связи по CAN. Объекты сообщений, которые не нужны, могут быть деактивированы.

**Рисунок 23-3. Общий поток инициализации модуля CAN**



**23.3.8.1.1 Конфигурация синхронизации битов CAN**

Для конфигурирования синхронизации битов CAN модуль должен находиться в режиме инициализации.

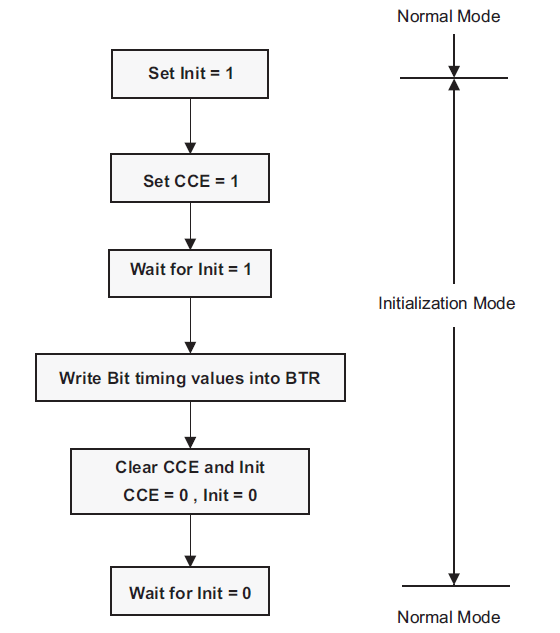
Поток программной конфигурации синхронизации битов CAN см. на рисунке 23-4.

Шаг 1: Войдите в режим инициализации, установив бит Init (Initialization) в регистре управления CAN.

Пока бит Init установлен, передача сообщений с и на шину CAN остановлена, а состояние выхода CAN\_TX является рецессивным (высоким). Счетчики ошибок CAN не обновляются. Установка бита Init не изменяет никаких других конфигурационных регистров.

Также следует учитывать, что CAN-модуль находится в режиме инициализации при аппаратном сбросе и во время отключения шины.

**Рисунок 23-4. Конфигурация синхронизации битов CAN**



Шаг 2: Установите бит Configure Change Enable (CCE) в регистре управления CAN.

Доступ к регистру битовой синхронизации (BTR) для конфигурирования битовой синхронизации разрешен, если установлены оба бита Init и CCE в регистре управления CAN.

Шаг 3: Дождитесь установки бита Init. Это позволит убедиться, что модуль вошел в режим инициализации.

Шаг 4: Запишите значения битовой синхронизации в регистр битовой синхронизации. См. раздел 23.3.16.2 для расчета значения BTR для заданного битового тайминга.

Шаг 5: Очистите биты CCE и Init.

Шаг 6: Дождитесь, пока бит Init очистится. Это гарантирует, что модуль вышел из режима инициализации.

После выполнения этих шагов модуль переходит в рабочий режим, синхронизируясь с шиной CAN, при условии, что BTR настроен в соответствии со скоростью передачи данных шины CAN, хотя объекты сообщений должны быть сконфигурированы перед выполнением любого обмена данными.

*ПРИМЕЧАНИЕ: Модуль не выйдет из режима инициализации, если в шаге 4 записаны неверные значения BTR.*

*ПРИМЕЧАНИЕ: Как описано в разделе 23.3.8.1, необходимые объекты сообщений должны быть сконфигурированы как объекты передачи или приема до начала передачи данных.*

**23.3.8.1.2 Конфигурация объектов сообщений**

Объекты сообщений могут быть сконфигурированы только через интерфейсные регистры; процессор не имеет прямого доступа к объекту сообщения (RAM сообщений). Ознакомьтесь с набором интерфейсных регистров (IFx) (см. раздел 23.3.17) и структурой объектов сообщений (см. раздел 23.3.18) перед конфигурированием объектов сообщений.

Более подробную информацию о процедуре конфигурирования объектов сообщений см. в разделе 23.3.14. Все объекты сообщений должны быть настроены на определенные идентификаторы или установлены на недействительные до начала передачи сообщения. Конфигурацию объектов сообщений можно изменить во время нормальной работы (то есть между передачами данных).

*ПРИМЕЧАНИЕ: Инициализация объектов сообщений не зависит от конфигурации битовой синхронизации.*

**23.3.8.1.3 Аппаратная инициализация оперативной памяти DCAN**

Аппаратная инициализация памяти для модуля DCAN включается в регистре управления устройством (DCAN\_RAMINIT), который инициализирует оперативную память нулями и устанавливает биты четности соответствующим образом. Дождитесь RAMINIT\_DONE, чтобы убедиться в успешной инициализации ОЗУ. Убедитесь, что тактирование модуля DCAN включено перед началом такой инициализации.

Более подробную информацию об аппаратной инициализации ОЗУ см. в главе 9 «Модуль управления».

**23.3.8.2 Передача сообщений CAN (нормальная работа)**

После инициализации DCAN и сброса бита Init в ноль ядро CAN синхронизируется с шиной CAN и готово к передаче сообщений в соответствии с конфигурированными объектами сообщений.

Процессор может включить линии прерываний (установив IE0 и IE1 в '1') одновременно с очисткой Init и CCE. Статусные прерывания EIE и SIE могут быть включены одновременно.

Связь по CAN может осуществляться в любом из следующих двух режимов: прерывание и опрос.

Регистр прерываний указывает на те объекты сообщений, для которых IntPnd = '1'. Он обновляется, даже если линии прерывания линии прерываний для CPU отключены (IE0/IE1 равны нулю).

CPU может параллельно опрашивать все биты NewDat и TxRqst объекта сообщения из регистров NewData X и регистров запроса передачи X (TXRQ X). Опрос может быть упрощен, если все объекты передачи сгруппированы по младшим номерам, а все объекты приема - по старшим.

Принятые сообщения сохраняются в соответствующих объектах сообщений, если они прошли фильтрацию приема.

Все сообщение (включая все арбитражные биты, DLC и до восьми байтов данных) сохраняется в объекте сообщений. Как следствие (например, при использовании маски идентификатора), арбитражные биты, которые маскируются маской «неважно», могут измениться в объекте сообщения при сохранении принятого сообщения.

Центральный процессор может читать или записывать каждое сообщение в любое время через интерфейсные регистры, поскольку обработчик сообщений гарантирует согласованность данных при одновременном доступе.

Если для сообщения существует постоянный объект сообщения (арбитражные и управляющие биты, установленные при конфигурировании и остающиеся для сообщения), то у него можно обновить только байты данных.

Если одному объекту сообщения должно быть назначено несколько передаваемых сообщений, весь объект сообщения должен быть сконфигурирован до того, как будет запрошена передача этого сообщения.

Одновременно может быть запрошена передача нескольких объектов сообщений. Впоследствии они передаются в соответствии с их внутренним приоритетом.

Сообщения могут быть обновлены или признаны недействительными в любой момент, даже если запрошенная передача еще не завершена. Однако байты данных будут отброшены, если сообщение будет обновлено до того, как будет начата ожидающая передача.

В зависимости от конфигурации объекта сообщения, передача может быть запрошена автоматически при получении удаленного кадра с соответствующим идентификатором.