

# RM0091 Reference manual

Это справочное руководство предназначено для разработчиков приложений. Оно предоставляет полную информацию о том, как использовать память микроконтроллера STM32F0x1 / STM32F0x2 / STM32F0x8 и периферийные устройства.

Это относится к устройствам STM32F031x4 / x6, STM32F051x8 / xB, STM32F091xB / xC, STM32F042x4 / x6, STM32F072x8 / xB, STM32F038x6, STM32F048x6, STM32F058x8, STM32F078xB и STM32F098xC.

Для целей настоящего руководства микроконтроллеры STM32F0x1 / STM32F0x2 / STM32F0x8 называются "STM32F0xx".

STM32F0xx — это семейство микроконтроллеров с различными размерами памяти, пакетами и периферийными устройствами.

Page 755

## 28 Последовательный интерфейс периферийных устройств / inter-IC sound (SPI / I<sup>2</sup>S)

### 28.1. Введение

Интерфейс SPI / I<sup>2</sup>S можно использовать для связи с внешними устройствами с использованием протокола SPI или звукового протокола I<sup>2</sup>S. Режим SPI или I<sup>2</sup>S выбирается программным обеспечением. Режим SPI Motorola выбран по умолчанию после сброса устройства.

Протокол последовательного периферийного интерфейса (SPI) поддерживает полудуплексную, полнодуплексную и симплексную синхронную последовательную связь с внешними устройствами. Интерфейс может быть сконфигурирован как ведущий, и в этом случае он предоставляет коммуникационный такт (SCK) внешнему подчиненному устройству. Интерфейс также может работать в конфигурации с несколькими хозяевами.

Протокол Inter-IC sound (I<sup>2</sup>S) также является синхронным последовательным коммуникационным интерфейсом. Он может работать в режиме ведомого или ведущего с полудуплексной связью. Полнодуплексные операции возможны благодаря объединению двух блоков I<sup>2</sup>S. Он может адресовать четыре разных стандарта звука, включая стандарт Philips I<sup>2</sup>S, стандарты MSB и LSB, а также стандарт PCM.

### 28.2 Основные функции SPI

- Ведущая или ведомая операция
- Полнодуплексные синхронные передачи на трех линиях
- Полудуплексная синхронная передача на двух линиях (с двунаправленной линией передачи данных)
- Симплексные синхронные передачи на двух линиях (с однонаправленной линией передачи данных)
- Выбор размера данных от 4 до 16 бит
- Возможность многомастерового режима
- 8 предварительных делителей скорости в режиме ведущего режима до  $f_{PCLK} / 2$ .
- Частота ведомого режима до  $f_{PCLK} / 2$ .
- Управление NSS аппаратным или программным обеспечением как для ведущего, так и для ведомого: динамическое изменение операций ведущий-ведомый
- Программируемая полярность и фаза синхронизации
- Программируемый порядок данных с MSB-первым или LSB-первым переключением
- Специальные флаги передачи и приема с возможностью прерывания
- Флаг состояния занятости шины SPI

- Поддержка SPI Motorola
- Аппаратная функция CRC для надежной связи:
  - Значение CRC может передаваться как последний байт в режиме Tx
  - Автоматическая проверка ошибок CRC для последнего принятого байта
- Сбой главного режима, флаги превышения с возможностью прерывания
- Флаг ошибки CRC
- Два 32-битных встроенных RX и Tx FIFO с возможностью DMA
- Поддержка режима SPI TI

### 28.3. Основные функции I<sup>2</sup>S

- Полудуплексная связь (только передатчик или приемник)
- Основные или подчиненные операции
- 8-битный программируемый линейный предделитель для достижения точных частот дискретизации аудио (от 8 кГц до 192 кГц)
  - Формат данных может быть 16-битным, 24-битным или 32-битным
  - Кадр пакета фиксируется на 16-разрядный (16-разрядный кадр данных) или 32-разрядный (16-разрядный, 24-разрядный, 32-разрядный кадр данных) по звуковому каналу
  - Программируемая полярность синхронизации (устойчивое состояние)
  - Флаг недогрузки при режиме ведомой передачи, флаг переполнения в режиме приема (ведущий и ведомый) и Флаг ошибки фрейма в режиме приема и передачи (только ведомый)
  - 16-разрядный регистр для передачи и приема с одним регистром данных для обеих сторон канала
- Поддерживаемые протоколы I<sup>2</sup>S:
  - Стандарт I<sup>2</sup>S Philips
  - MSB-обоснованный стандарт (выравнивание по левому краю)
  - Стандарт с выравниванием по LSB (выравнивание по правому краю)
  - стандарт PCM (с синхронизацией коротких и длинных кадров в 16-битном канальном кадре или 16-битном кадре данных, расширенном до 32-битного кадра канала)
- Направление данных всегда сначала MSB
- Возможность DMA для передачи и приема (16-битная широта)
- Ведущие часы могут быть выведены для управления внешним аудиокomпонентом. Соотношение фиксировано:
 
$$256 \times FS$$
 (где FS — частота дискретизации аудиосигнала)

### 28.4 Внедрение SPI / I<sup>2</sup>S

В таблице 111 описывается реализация SPI / I<sup>2</sup>S в устройствах STM32F0xx.

Таблица 111. Реализация SPI STM32F0xx (1)

| SPI Features             | STM32F03x | STM32F04x<br>STM32F05x |      | STM32F07x<br>STM32F09x |      |
|--------------------------|-----------|------------------------|------|------------------------|------|
|                          | SPI1      | SPI1                   | SPI2 | SPI1                   | SPI2 |
| Hardware CRC calculation | X         | X                      | X    | X                      | X    |
| Rx/Tx FIFO               | X         | X                      | X    | X                      | X    |
| NSS pulse mode           | X         | X                      | X    | X                      | X    |
| I2S mode                 | X         | X                      | -    | X                      | X    |
| TI mode                  | X         | X                      | X    | X                      | X    |

1. X = supported.

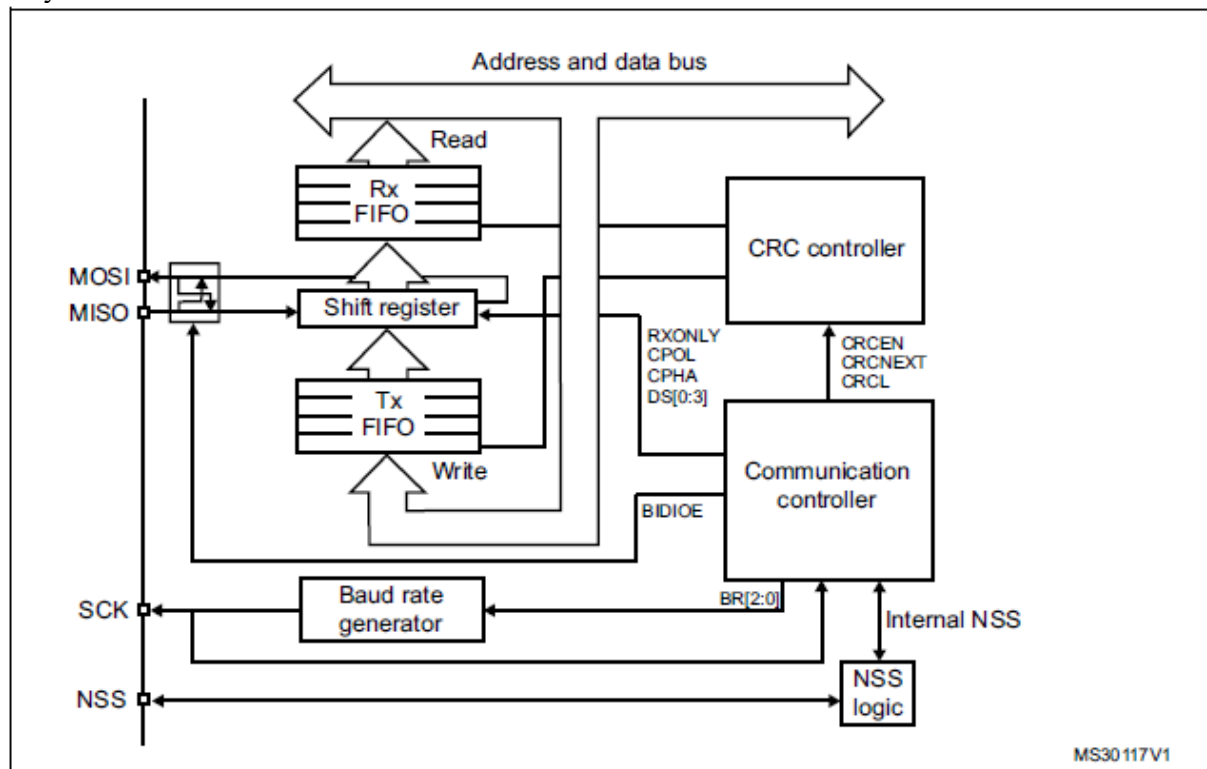
## 28.5 Описание функций SPI

### 28.5.1 Общее описание

SPI обеспечивает синхронную последовательную связь между MCU и внешними устройствами.

Прикладное программное обеспечение может управлять коммуникацией, опрашивая флаг состояния или используя специализированное прерывание SPI. Основные элементы SPI и их взаимодействия показаны на следующей блок-схеме. Рисунок 270.

Рисунок 270. Блок-схема SPI



Четыре вывода ввода-вывода предназначены для связи SPI с внешними устройствами.

- MISO: данные ведущего / ведомого устройства. В общем случае этот вывод используется для передачи данных в подчиненном режиме и приема данных в режиме ведущего.
- MOSI: данные «Master Out / Slave In». В общем случае этот вывод используется для передачи данных в режиме ведущего устройства и приема данных в режиме ведомого.
- SCK: Выходной сигнал последовательного тактового сигнала для мастеров SPI и входной вывод для ведомых устройств SPI.
- NSS: Штырь выбора ведомого. В зависимости от настроек SPI и NSS этот вывод может использоваться для:

- выберите индивидуальное ведомое устройство для связи
- синхронизировать кадр данных или
- обнаружить конфликт между несколькими мастерами

Подробнее см. Раздел 28.5.5: Управление выводом подчиненного устройства (NSS).

Шина SPI обеспечивает связь между одним ведущим устройством и одним или несколькими ведомыми устройствами. Шина состоит как минимум из двух проводов — один для тактового сигнала, а другой для синхронной передачи данных. Другие сигналы могут добавляться в зависимости от обмена данными между узлами SPI и управления их сигналами выбора ведомого устройства.

### 28.5.2. Связь между одним ведущим и одним подчиненным

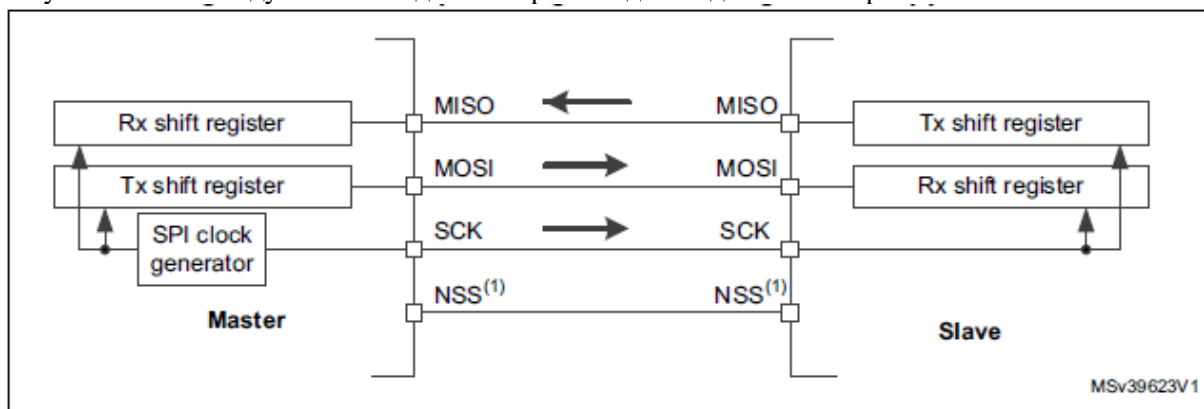
SPI позволяет микроконтроллеру связываться с использованием различных конфигураций, в зависимости от целевого устройства и требований приложения. Эти конфигурации используют 2 или 3 провода (с программным управлением NSS) или 3 или 4 провода (с аппаратным управлением NSS).

Связь всегда инициируется мастером.

#### Полнодуплексная связь

По умолчанию SPI настроен для полнодуплексной связи. В этой конфигурации регистры сдвига ведущего и ведомого соединены с использованием двух однонаправленных линий между MOSI и выводами MISO. Во время SPI-связи данные сдвигаются синхронно на тактовых фронтах SCK, предоставляемых ведущим. Ведущее устройство передает данные, которые должны быть отправлены ведомому через линию MOSI, и принимает данные от подчиненного устройства через линию MISO. Когда передача фрейма данных завершена (все биты сдвинуты), информация между ведущим и ведомым обменивается.

Рисунок 271. Полнодуплексное одностороннее / одно подчиненное приложение

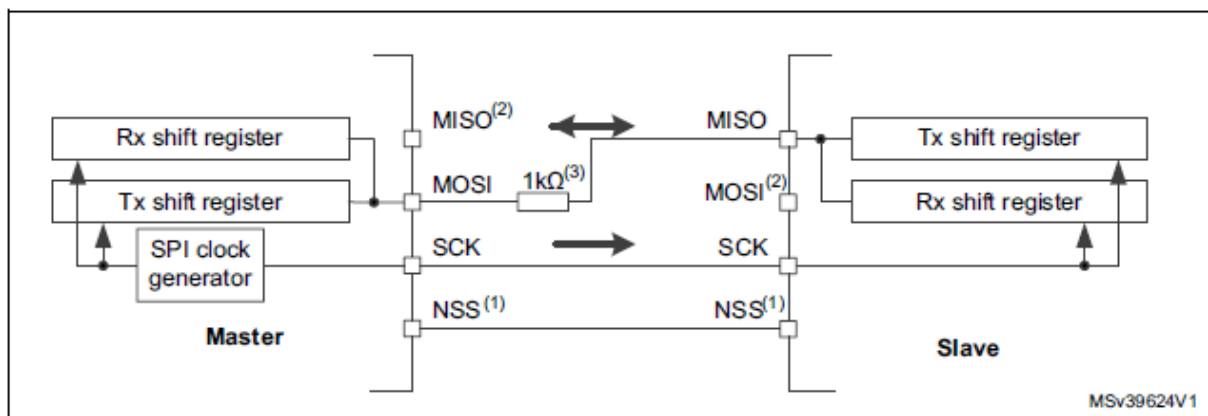


1. Выводы NSS могут использоваться для обеспечения потока аппаратного управления между ведущим и ведомым. По желанию, штифты могут быть оставлены неиспользованными периферийным устройством. Затем поток должен обрабатываться как для главного, так и для ведомого. Подробнее см. В Разделе 28.5.5: Управление выводом подчиненного устройства (NSS).

#### Полудуплексная связь

SPI может взаимодействовать в полудуплексном режиме, устанавливая бит BIDIMODE в регистр SPIx\_CR1. В этой конфигурации одна линия перекрестного соединения используется для связи регистров сдвига ведущего и ведомого вместе. Во время этой связи данные синхронно сдвигаются между сдвиговыми регистрами на краю синхроимпульса SCK в направлении передачи, выбранном взаимно как ведущим, так и ведомым, с битом BDIOE в их регистрах SPIx\_CR1. В этой конфигурации вывод MISO ведущего устройства и вывод MOSI подчиненного устройства являются свободными для других приложений и действуют как GPIO.

Рисунок 272. Полудуплексное одностороннее / одно подчиненное приложение



1. Выводы NSS могут использоваться для обеспечения потока аппаратного управления между ведущим и ведомым. По желанию, штифты могут быть оставлены неиспользованными периферийным устройством. Затем поток должен обрабатываться как для главного, так и для ведомого. Подробнее см. В Разделе 28.5.5: Управление выводом ведомого устройства (NSS).

2. В этой конфигурации вывод MISO ведущего устройства и вывод MOSI подчиненного устройства могут использоваться как GPIO.

3. Критическая ситуация может произойти, когда направление связи изменяется не синхронно между двумя узлами, работающими в двунаправленном режиме, и новый передатчик обращается к общей линии данных, в то время как прежний передатчик все еще сохраняет противоположное значение на линии (значение зависит от конфигурации SPI и данных связи). Оба узла затем сражаются, обеспечивая при этом противоположные уровни выхода на общей линии временного до следующего узла, соответственно изменяя его настройки направления. В этом режиме рекомендуется вставлять последовательное сопротивление между контактами MISO и MOSI для защиты выходов и ограничения тока между ними в этой ситуации.

#### Симплексная связь

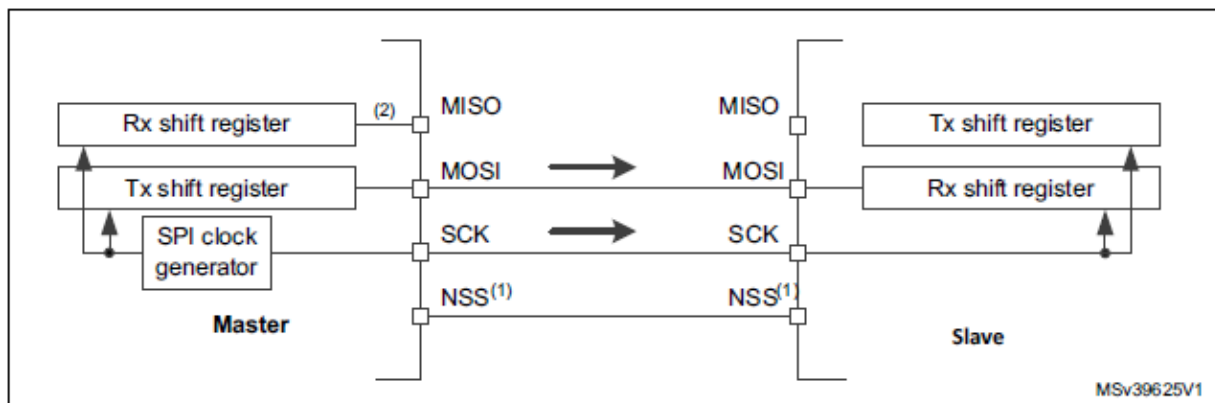
SPI может взаимодействовать в симплексном режиме, устанавливая SPI только в режиме передачи или только в режиме приема, используя бит RXONLY в регистре SPIx\_CR2. В этой конфигурации для передачи между сдвиговыми регистрами ведущего и ведомого используются только одна линия. Остальные пары контактов MISO и MOSI не используются для связи и могут использоваться как стандартные GPIO.

- Режим только передачи (RXONLY = 0): Настройки конфигурации такие же, как и для полнодуплексного режима. Приложение должно игнорировать информацию, записанную на неиспользуемом входном выводе. Этот вывод может использоваться как стандартный GPIO.

- Режим только для приема (RXONLY = 1): приложение может отключить функцию выхода SPI, установив бит RXONLY. В подчиненной конфигурации выход MISO отключен, а вывод может использоваться как GPIO. Ведомое устройство продолжает принимать данные с вывода MOSI, когда активен его сигнал выбора ведомого (см. 28.5.5: управление выводом ведомого устройства (NSS)).

Полученные события данных появляются в зависимости от конфигурации буфера данных. В основной конфигурации выход MOSI отключен, и вывод может использоваться как GPIO. Сигнал синхронизации генерируется непрерывно до тех пор, пока включен SPI. Единственный способ остановить часы — очистить бит RXONLY или бит SPE и дождаться окончания формирования входящего паттерна с контакта MISO и заполнить структуру буфера данных в зависимости от его конфигурации.

Рисунок 273. Односимвольное однопользовательское / одно подчиненное приложение (ведущее устройство в режиме только для передачи / подчиненный в режиме только для приема)



1. Выводы NSS могут использоваться для обеспечения потока аппаратного управления между ведущим и ведомым. По желанию, штифты могут быть оставлены неиспользованными периферийным устройством. Затем поток должен обрабатываться как для главного, так и для ведомого. Подробнее см. В Разделе 28.5.5: Управление выводом ведомого устройства (NSS).

2. Случайная входная информация фиксируется на входе регистра сдвига Rx передатчика. Все события, связанные с потоком приема передатчика, должны игнорироваться в стандартном режиме только для передачи (например, флаг OVF).

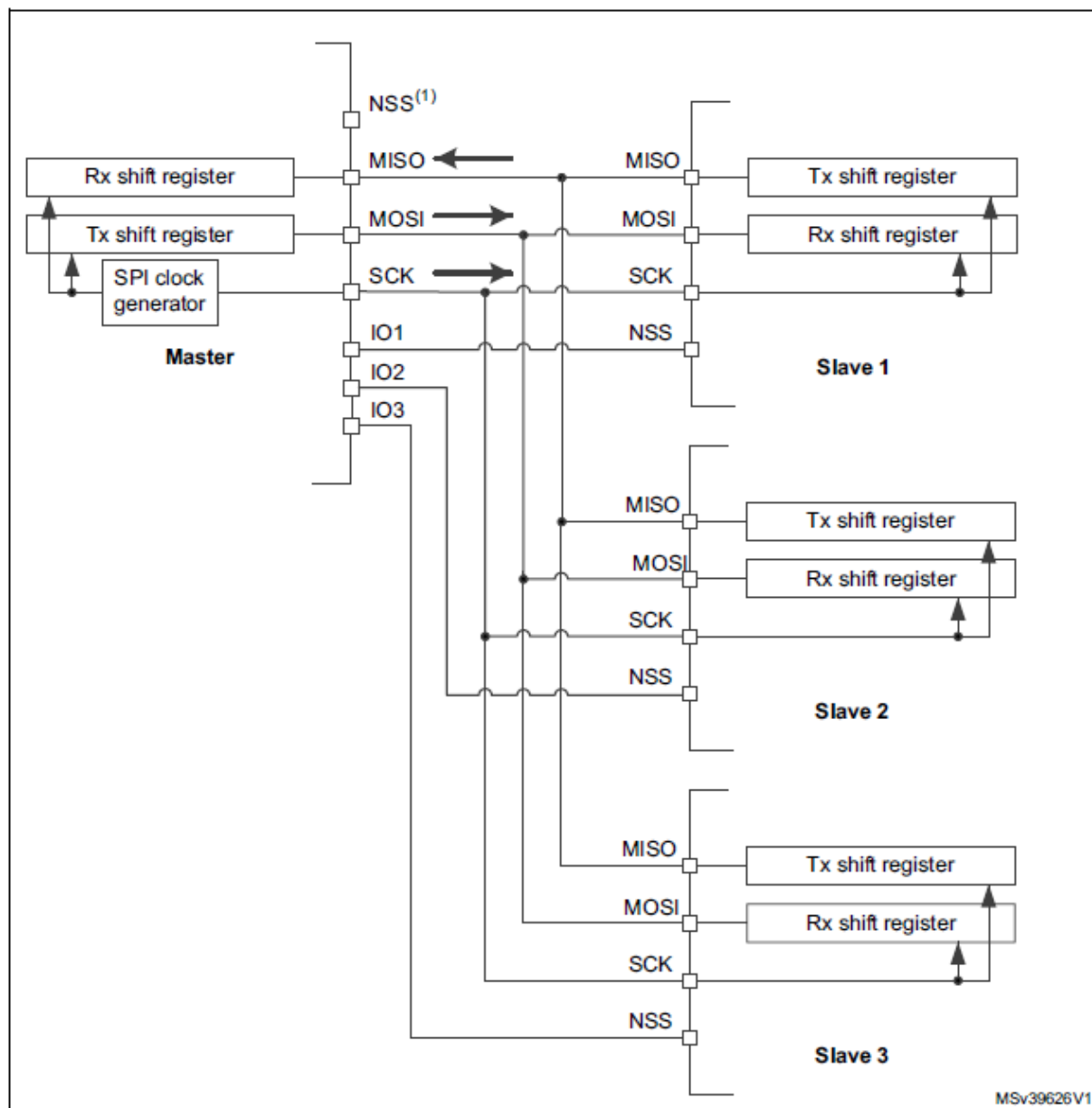
3. В этой конфигурации оба вывода MISO могут использоваться как GPIO.

Примечание. Любая симплексная связь может быть альтернативно заменена вариантом полудуплексной связи с постоянной настройкой направления транзакции (двунаправленный режим включен, в то время как бит BDIO не изменяется).

### 28.5.3 Стандартная связь с несколькими подчиненными устройствами

В конфигурации с двумя или более независимыми подчиненными устройствами мастер использует контакты GPIO для управления линиями выбора кристалла для каждого подчиненного устройства (см. Рис. 274.). Ведущий должен выбрать один из ведомых отдельно, потянув за низкий GPIO, подключенный к ведомому входу NSS. Когда это будет сделано, будет установлено стандартное сообщение ведущего и специализированного ведомого.

Рисунок 274. Мастер и три независимых подчиненных



1. Штырь NSS не используется на главной стороне в этой конфигурации. Он должен управляться изнутри (SSM = 1, SSI = 1), чтобы предотвратить любую ошибку MODF.

2. Поскольку контакты MISO подчиненных устройств соединены вместе, все ведомые устройства должны иметь конфигурацию GPIO их штыря MISO в качестве альтернативной функции открытого стока (см. Раздел 8.3.7: Вход / выход альтернативных функций ввода / вывода на стр. 152).

#### 28.5.4. Связь с несколькими ведущими устройствами

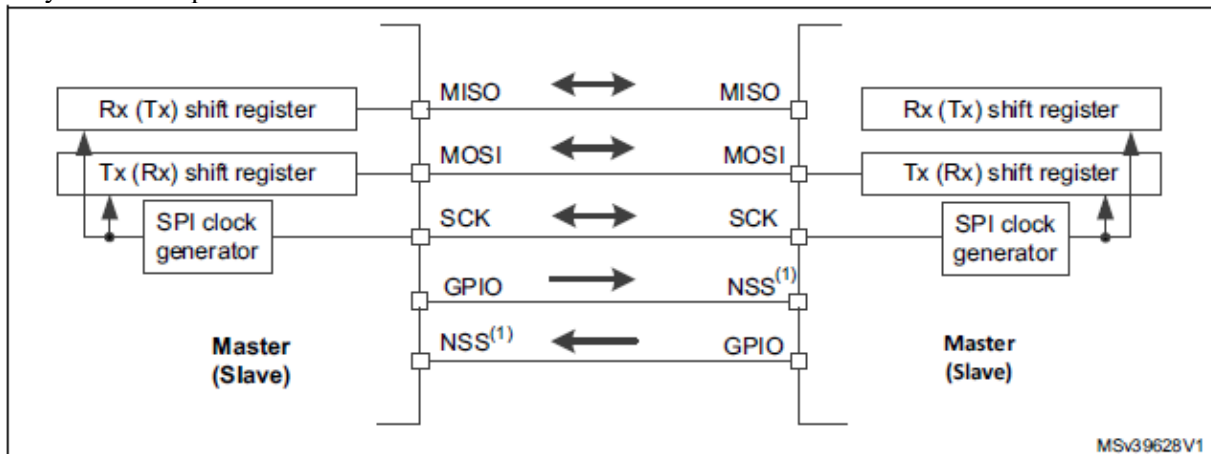
Если, в первую очередь, SPI-шина не предназначена для использования с несколькими мастерами, пользователь может использовать встроенную функцию, которая обнаруживает потенциальный конфликт между двумя узлами, пытающимися одновременно управлять шиной. Для этого обнаружения используется вывод NSS, сконфигурированный в режиме аппаратного ввода.

Соединение более чем двух узлов SPI, работающих в этом режиме, невозможно, так как только один узел может применить свой вывод к общей линии данных во время.

Когда узлы неактивны, оба остаются в подчиненном режиме по умолчанию. Как только один узел хочет перегнать управление на шине, он переключается в главный режим и применяет активный уровень на входе выбора ведомого другого узла через выделенный вывод GPIO. После завершения сеанса активный сигнал выбора ведомого освобождается, и узел, управляющий временной шиной, возвращается в пассивный подчиненный режим, ожидая следующего запуска сеанса.

Если потенциально оба узла одновременно повысили свой запрос на мастеринг, появится конфликт с шиной (см. Событие MODF fault mode). Затем пользователь может применить некоторый простой арбитражный процесс (например, чтобы отложить следующую попытку с помощью предопределенных различных тайм-аутов, применяемых в обоих узлах).

Рисунок 275. Приложение Multi-master



1. Вывод NSS сконфигурирован в режиме аппаратного ввода на обоих узлах. Его активный уровень позволяет управлять выводом линии MISO, поскольку пассивный узел настроен как подчиненный.

### 28.5.5 Управление вышестоящим выводом (NSS)

В подчиненном режиме NSS работает как стандартный вход «выбор кристалла» и позволяет ведомому взаимодействовать с ведущим. В ведущем режиме NSS может использоваться как выход или вход.

В качестве входного сигнала он может предотвратить столкновение с многомашинной шиной, и в качестве выходного сигнала он может управлять сигналом выбора ведомого отдельного ведомого устройства.

Управление аппаратным или программным управлением ведомым можно установить с помощью бита SSM в регистре SPIx\_CR1:

- Управление программным NSS (SSM = 1): в этой конфигурации информация о выборе ведомого определяется внутренним битом SSI в регистре SPIx\_CR1. Внешний штырек NSS доступен для других приложений.

- Аппаратное управление NSS (SSM = 0): в этом случае возможны два варианта конфигураций. Используемая конфигурация зависит от конфигурации вывода NSS

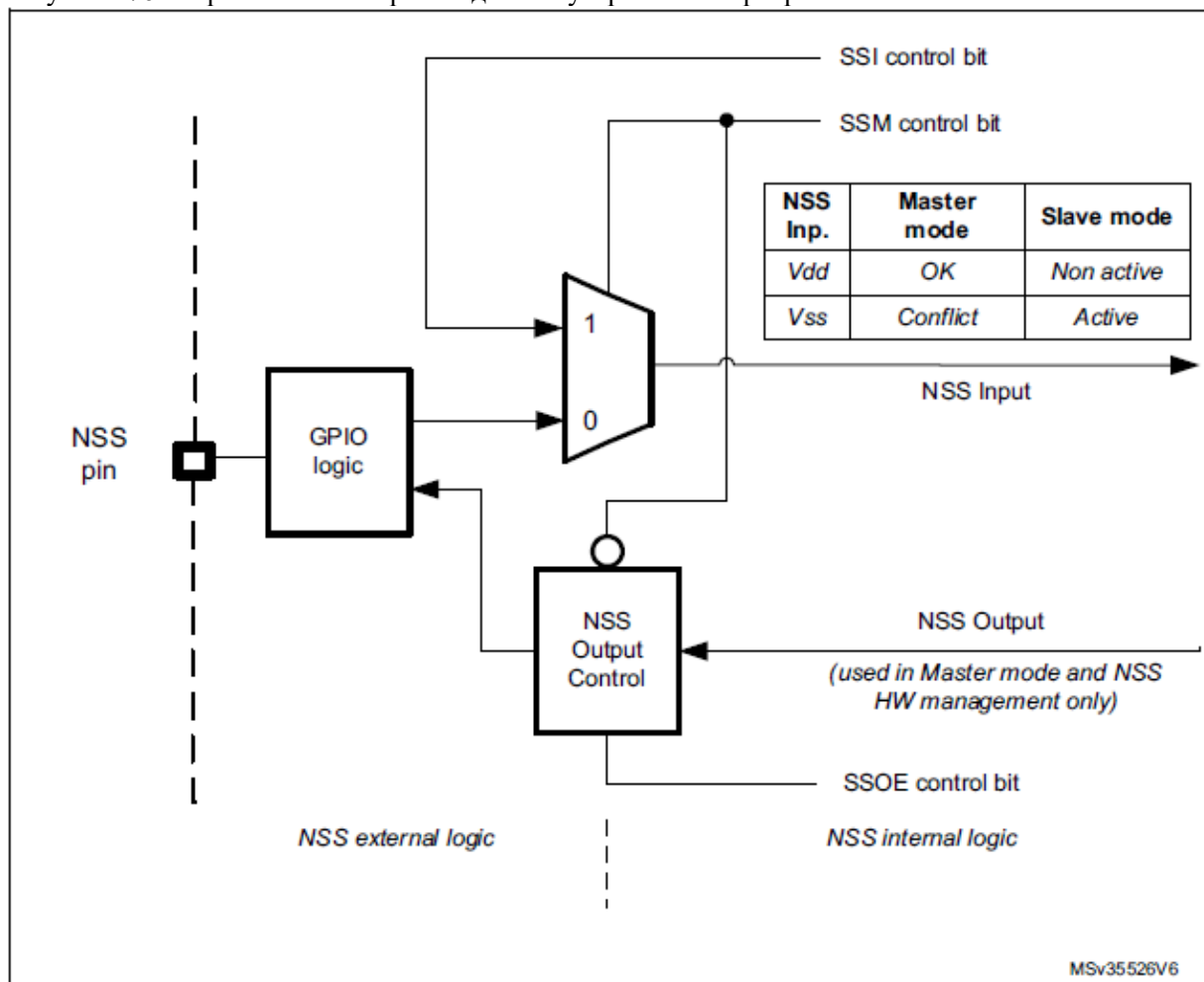
(Бит SSOE в регистре SPIx\_CR1).

- Разрешение выхода NSS (SSM = 0, SSOE = 1): эта конфигурация используется только тогда, когда MCU установлен как ведущий. Вывод NSS управляется аппаратным обеспечением. Сигнал NSS управляется низким, как только SPI включен в режиме ведущего (SPE = 1), и поддерживается низким, пока SPI не отключен (SPE = 0). Импульс может генерироваться между непрерывными сеансами связи, если активирован импульсный режим NSS (NSSP = 1). SPI не может работать в конфигурации multimaster с этой настройкой NSS.

- Отключение выхода NSS (SSM = 0, SSOE = 0): если микроконтроллер выступает в роли ведущего на шине, эта конфигурация допускает работу с несколькими ведущими устройствами.

Если в этом режиме вывод NSS вытягивается низко, SPI переходит в состояние сбоя в режиме ведущего устройства, и устройство автоматически реконфигурируется в режиме ведомого. В подчиненном режиме вывод NSS работает как стандартный вход «выбор кристалла», а ведомое устройство выбирается, когда линия NSS находится на низком уровне.

Рисунок 276. Управление выбором ведомого устройства / программного обеспечения



### 28.5.6 Форматы связи

Во время связи SPI операции приема и передачи выполняются одновременно. Серийные часы (SCK) синхронизируют сдвиг и выборку информации по линиям данных. Формат связи зависит от фазы синхронизации, полярности тактового сигнала и формата кадра данных. Чтобы иметь возможность взаимодействовать друг с другом, устройства ведущего и ведомого должны следовать одному и тому же формату связи.

#### Контроль фазы и полярности синхронизации

Четыре возможных временных отношения могут выбираться программным обеспечением, используя биты CPOL и CPHA в регистре SPIx\_CR1. Бит CPOL (clock polarity) управляет значением состояния простоя часов, когда данные не передаются. Этот бит влияет как на главный, так и на ведомый. Если CPOL сброшен, вывод SCK имеет состояние низкого уровня ожидания. Если CPOL установлен, вывод SCK имеет состояние высокого уровня бездействия.

Если бит CPHA установлен, то второй фронт на выводе SCK захватывает первый бит данных, который транслируется (спадающий фронт, если бит CPOL сбрасывается, нарастающий фронт, если бит CPOL установлен). Данные фиксируются при каждом появлении этого типа перехода часов. Если бит CPHA сбрасывается, то первый фронт на выводе SCK захватывает пер-

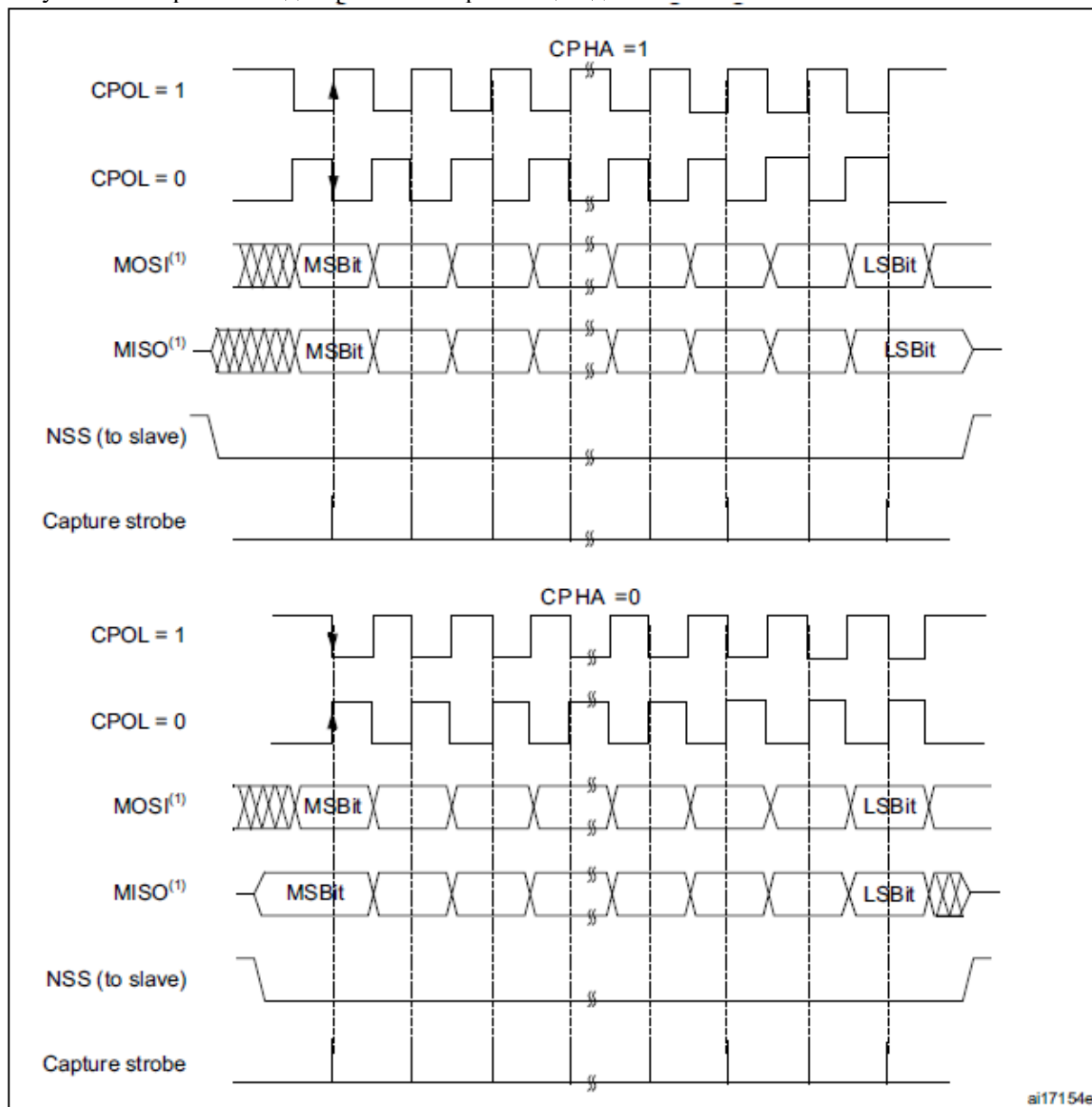
вый бит данных, прошедший транзакцию (спадающий фронт, если бит CPOL установлен, нарастающий фронт, если бит CPOL сбрасывается). Данные фиксируются при каждом появлении этого типа перехода часов.

Комбинация битов CPOL (такты частота) и CPHA (такты частота) выбирает фронт тактового сигнала захвата данных.

На рис. 277 показан полный дуплексный переход SPI с четырьмя комбинациями битов CPHA и CPOL. Примечание: Перед изменением битов CPOL / CPHA, SPI необходимо отключить, сбросив бит SPE.

Неактивное состояние SCK должно соответствовать полярности, выбранной в регистре SPIx\_CR1 (путем подтягивания SCK, если CPOL = 1 или сбрасывания SCK, если CPOL = 0).

Рисунок 277. Временная диаграмма синхронизации данных



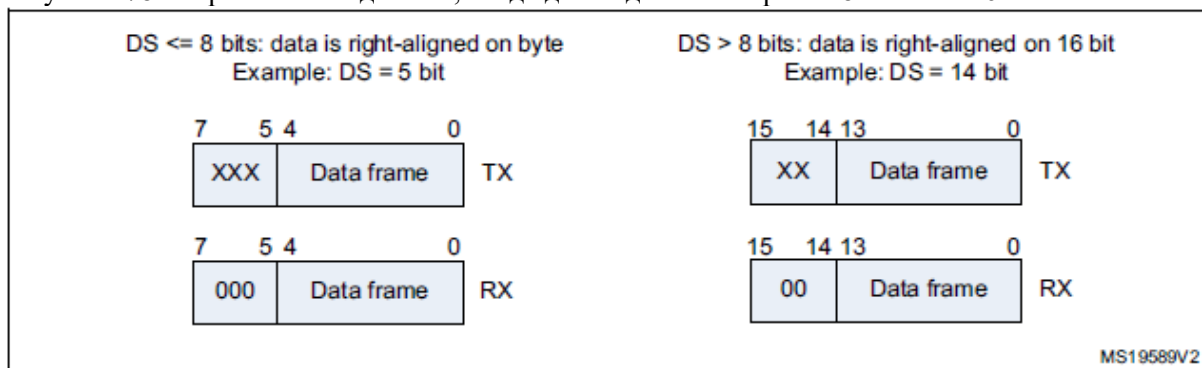
1. Порядок бит данных зависит от установки бита LSBFIRST.

#### Формат кадра данных

Регистр сдвига SPI может быть установлен для смещения MSB-first или LSB-first, в зависимости от значения бита LSBFIRST. Размер фрейма данных выбирается с использованием битов DS. Его можно установить в диапазоне от 4 бит до 16 бит, и установка применяется как для пе-

редачи, так и для приема. Независимо от выбранного размера фрейма данных, доступ на чтение к FIFO должен быть согласован с уровнем FRXTH. Когда осуществляется доступ к регистру SPIx\_DR, кадры данных всегда выровнены по правому краю либо в байтах (если данные вписываются в байт), либо в полуслове (см. Рис. 278). Во время связи только биты в кадре данных синхронизируются и передаются

Рисунок 278. Выравнивание данных, когда длина данных не равна 8-бит или 16-бит



### 28.5.7 Конфигурация SPI

Процедура конфигурирования почти одинакова для ведущего и ведомого устройств. Для специальных настроек режима следуйте специальным разделам. Когда стандартная связь должна быть инициализирована, выполните следующие шаги:

1. Запишите соответствующие регистры GPIO: Настройте GPIO для выводов MOSI, MISO и SCK.

2. Напишите в регистр SPI\_CR1:

А) Сконфигурируйте скорость передачи по последовательному каналу с помощью битов BR [2: 0] (Примечание: 4).

В) Настроить комбинации битов CPOL и CPHA, чтобы определить одно из четырех отношений между передачей данных и последовательным тактовым импульсом (CPHA должен быть очищен в режиме NSSP). (Примечание: 2 — за исключением случая, когда CRC включен в режиме TI).

С) Выберите симплексный или полудуплексный режим, сконфигурировав RXONLY или BIDIMODE и BIDIOE (RXONLY и BIDIMODE не могут быть установлены одновременно).

Д) Настройте бит LSBFIRST, чтобы определить формат кадра (Примечание: 2).

Е) Настройте биты CRCL и CRCEN, если требуется CRC (в то время как SCK-сигнал находится в состоянии ожидания).

Ф) Настроить SSM и SSI (Примечания: 2 и 3).

Г) Настроить бит MSTR (в конфигурации многошагового NSS, избежать конфликтного состояния в NSS, если ведущее устройство сконфигурировано для предотвращения ошибки MODF).

3. Запишите в регистр SPI\_CR2:

А) Настройте биты DS [3: 0], чтобы выбрать длину данных для передачи.

В) Настроить SSOE (Примечания: 1 & 2 & 3).

С) Установите бит FRF, если требуется протокол TI (не сбрасывайте бит NSSP в режиме TI).

Д) Установите бит NSSP, если требуется импульсный режим NSS между двумя блоками данных (сохранение битов CPHA и TI, очищенных в режиме NSSP).

Е) Настроить бит FRXTH. Порог RXFIFO должен быть выровнен с размером доступа для чтения для регистра SPIx\_DR.

Ф) Инициализировать биты LDMA\_TX и LDMA\_RX, если DMA используется в упакованном режиме.

4. Запишите в регистр SPI\_CRCPR: Настройте CRC-полином, если необходимо.

5. Запишите надлежащие регистры DMA: сконфигурируйте потоки DMA, выделенные для SPI Tx и Rx в регистрах DMA, если используются потоки DMA.

Примечание: (1) В подчиненном режиме не требуется шаг.

(2) Шаг не требуется в режиме TI.

(3) Шаг не требуется в режиме NSSP.

(4) Этап не требуется в подчиненном режиме, кроме подчиненного, работающего в режиме TI

Пример кода см. В разделах Приложения A.17.1: пример кода конфигурации SPI и A.17.2: пример кода конфигурации ведомого устройства SPI.

## 28.5.8 Процедура включения SPI

Рекомендуется включить ведомое устройство SPI до того, как ведущее устройство отправит часы. В противном случае может возникнуть нежелательная передача данных. Регистр данных подчиненного устройства должен уже содержать данные, которые должны быть отправлены перед началом связи с ведущим (либо на первом фронте тактового сигнала связи, либо перед завершением текущей связи, если тактовый сигнал является непрерывным). Сигнал SCK должен быть установлен на уровне состояния ожидания, соответствующем выбранной полярности, до того, как ведомое устройство SPI будет включено.

Ведущий в полнодуплексном режиме (или в любом режиме только для передачи) начинает связываться, когда SPI включен, а TXFIFO не пуст или при следующей записи в TXFIFO.

В любом режиме ведущего приема (RXONLY = 1 или BIDIMODE = 1 & BIDIOE = 0) ведущий начинает связь и часы начинают работать сразу же после включения SPI.

Для обработки DMA следуйте специальному разделу.

## 28.5.9 Процедуры передачи и приема данных RXFIFO и TXFIFO

Все транзакции данных SPI проходят через 32-разрядные встроенные FIFO. Это позволяет SPI работать в непрерывном потоке и предотвращает перерасход, когда размер кадра данных короток. Каждое направление имеет свой собственный FIFO, называемый TXFIFO и RXFIFO. Эти FIFO используются во всех режимах SPI, кроме режима только приемника (ведомого или ведущего) с включенным вычислением CRC (см. Раздел 28.5.14: расчет CRC).

Обработка FIFO зависит от режима обмена данными (дуплекс, симплекс), формата кадра данных (количество битов в кадре), размера доступа, выполняемого в регистрах данных FIFO (8-бит или 16-бит), и независимо от того, Упаковка данных используется при обращении к FIFO (см. Раздел 28.5.13: режим TI).

Доступ на чтение к регистру SPIx\_DR возвращает самое старое значение, хранящееся в RXFIFO, которое еще не было прочитано. Доступ на запись к SPIx\_DR сохраняет записанные данные в TXFIFO в конце очереди отправки. Доступ на чтение всегда должен быть выровнен с порогом RXFIFO, сконфигурированным с помощью бита FRXTH в регистре SPIx\_CR2. FTLVL [1: 0] и FRLVL [1: 0] указывают текущий уровень занятости обоих FIFO. Доступ чтения к регистру SPIx\_DR должен управляться событием RXNE. Это событие срабатывает, когда данные хранятся в RXFIFO, и достигается пороговое значение (определенное битом FRXTH). Когда RXNE очищается, RXFIFO считается пустым. Аналогичным образом, доступ для записи кадра данных, подлежащего передаче, управляется событием TXE. Это событие срабатывает, когда уровень TXFIFO меньше или равен половине его емкости. В противном случае TXE очищается, и TXFIFO считается заполненным. Таким образом, RXFIFO может хранить до четырех кадров данных, тогда как TXFIFO может хранить только до трех, если формат кадра данных не превышает 8 бит. Это различие предотвращает возможное повреждение 3х 8-битных фреймов данных, уже сохраненных в TXFIFO, когда программное обеспечение пытается записать больше данных в 16-битном режиме в TXFIFO. Оба события TXE и RXNE могут обрабатываться или обрабатываться прерываниями. См. Рис. 280 по фиг. 283.

Другим способом управления обменом данными является использование DMA (см.).

Если следующие данные будут получены, когда RXFIFO заполнен, произойдет событие переполнения (см. Описание флага OVR в разделе 28.5.10: флаги состояния SPI). Событие переполнения может быть опрошено или обработано прерыванием.

Установленный бит BSY указывает текущую транзакцию текущего кадра данных. Когда тактовый сигнал работает непрерывно, флаг BSY остается установленным между фреймами данных на ведущем устройстве, но становится низким в течение минимальной продолжительности одного такта SPI в ведомом устройстве между каждой передачей кадра данных.

#### **Обработка последовательности**

Для завершения сообщения можно передать несколько кадров данных в одной последовательности. Когда передача разрешена, последовательность начинается и продолжается, пока любые данные присутствуют в TXFIFO ведущего устройства. Сигнал синхронизации подается непрерывно ведущим, пока TXFIFO не станет пустым, после чего он перестает ждать дополнительных данных.

В режимах только для приема, полудуплекс (BIDIMODE = 1, BIDIOE = 0) или симплекс (BIDIMODE = 0, RXONLY = 1) мастер запускает последовательность немедленно, когда включен и SPI, и режим только для приема. Сигнал синхронизации поступает от ведущего устройства, и он не останавливается до тех пор, пока ведущий не отключит режим SPI или только прием. До этого хозяин непрерывно принимает кадры данных.

В то время как ведущий может предоставить все транзакции в непрерывном режиме (сигнал SCK непрерывный), он должен уважать подчиненные возможности для обработки потока данных и его содержимого в любое время.

При необходимости мастер должен замедлять связь и предоставлять либо более медленные часы, либо отдельные кадры или сеансы передачи данных с достаточной задержкой. Имейте в виду, что в режиме SPI отсутствует сигнал ошибки нижнего уровня для ведущего или ведомого, и данные ведомого всегда обрабатываются и обрабатываются ведущим, даже если ведомое устройство не могло подготовить его правильно во времени. Предпочтительно, чтобы ведомое устройство использовало DMA, особенно когда кадры данных короче и скорость шины высока.

Каждая последовательность должна быть заключена в оболочку импульсом NSS параллельно с мультиклавной системой, чтобы выбрать только одного из подчиненных устройств для связи. В одной ведомой системе нет необходимости управлять ведомым устройством с помощью NSS, но часто лучше также обеспечить здесь импульс, чтобы синхронизировать ведомое устройство с началом каждой последовательности данных. NSS может управляться как программным обеспечением, так и аппаратным обеспечением (см. Раздел

### **28.5.5: управление выводом подчиненного устройства (NSS)).**

Когда бит BSY установлен, он обозначает текущую транзакцию фрейма данных. Когда выделенная транзакция кадра завершена, флаг RXNE поднимается. Последний бит просто отбирается, и полный фрейм данных сохраняется в RXFIFO.

#### **Процедура отключения SPI**

Когда SPI отключен, необходимо следовать процедурам отключения, описанным в этом параграфе. Это важно сделать, прежде чем система перейдет в режим пониженного энергопотребления, когда периферийные часы будут остановлены. В этом случае текущие транзакции могут быть повреждены. В некоторых режимах отключить процедуру — единственный способ остановить непрерывный обмен данными.

Мастер в режиме полного дуплекса или только для передачи может завершить любую транзакцию, когда прекращает предоставление данных для передачи. В этом случае часы останавливаются после последней транзакции данных.

Особую осторожность следует проявлять в режиме упаковки, когда выполняется нечетное количество фреймов данных для предотвращения обмена фиктивными байтами (см. Раздел «Упаковка данных»). Перед отключением SPI в этих режимах пользователь должен следовать стандартной процедуре отключения. Когда SPI отключен на ведущем передатчике, когда тран-

закция фрейма продолжается или следующий фрейм данных сохраняется в TXFIFO, поведение SPI не гарантируется.

Когда ведущее устройство находится в любом режиме приема, единственный способ остановить непрерывные тактовые импульсы — отключить периферийное устройство с помощью  $SPE = 0$ . Это должно происходить в определенном временном окне в транзакции последнего кадра данных только между временем выборки его первого бита и началом его последней передачи бита (чтобы получить полное число ожидаемых кадров данных и предотвратить любые дополнительные «фиктивные» чтения данных После последнего действительного кадра данных). При отключении SPI в этом режиме необходимо соблюдать особую процедуру.

Данные, полученные, но не прочитанные, сохраняются в RXFIFO, когда SPI отключен, и должны быть обработаны при следующем включении SPI, перед началом новой последовательности. Чтобы предотвратить наличие непрочитанных данных, убедитесь, что RXFIFO пуст при отключении SPI, используя правильную процедуру отключения, или путем инициализации всех регистров SPI с помощью программного сброса с помощью управления определенным регистром, предназначенным для периферийного сброса (см. Биты  $SPI_{IRST}$  В регистрах  $RCC\_APB_{iRSTR}$ ).

Стандартная процедура отключения основана на вытягивании статуса  $BSY$  вместе с  $FTLVL [1: 0]$ , чтобы проверить, завершен ли сеанс передачи. Эта проверка может быть выполнена и в особых случаях, когда необходимо определить окончание текущих транзакций, например:

- Когда сигнал NSS управляется программным обеспечением, и ведущий должен обеспечить надлежащий конец импульса NSS для ведомого или
- Когда потоки транзакций из DMA или FIFO завершены, в то время как последний фрейм данных или транзакция цикла CRC все еще продолжают в периферийной шине.

Правильная процедура отключения (кроме случаев, когда используется режим только приема):

1. Подождите, пока  $FTLVL [1: 0] = 00$  (больше нет данных для передачи).
2. Подождите, пока  $BSY = 0$  (обрабатывается последний кадр данных).
3. Отключите SPI ( $SPE = 0$ ).
4. Чтение данных до  $FRLVL [1: 0] = 00$  (чтение всех полученных данных).

Правильная процедура отключения для определенных режимов только:

1. Прекратите приемный поток, отключив SPI ( $SPE = 0$ ) в определенном временном окне во время последнего кадра данных.
2. Подождите, пока  $BSY = 0$  (обрабатывается последний кадр данных).
3. Чтение данных до  $FRLVL [1: 0] = 00$  (считывание всех полученных данных).

Примечание: если используется режим уплотнения и должно быть получено нечетное число фреймов данных с форматом меньше или равным 8 битам (вставка в один байт),  $FRXTH$  должен быть установлен, когда  $FRLVL [1: 0] = 01$ , в Чтобы генерировать событие  $RXNE$ , чтобы прочитать последний нечетный кадр данных и сохранить хорошее выравнивание указателя FIFO.

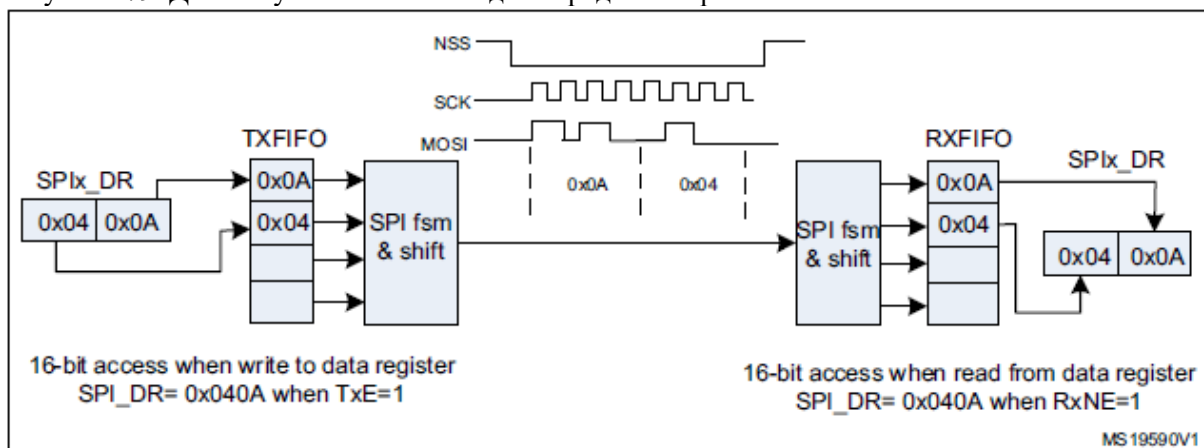
#### **Упаковка данных**

Когда размер кадра данных вписывается в один байт (меньше или равен 8 битам), упаковка данных используется автоматически, когда любой 16-разрядный доступ для чтения или записи выполняется в регистре  $SPIx\_DR$ . В этом случае двойной кадр данных обрабатывается параллельно. Сначала SPI работает с использованием шаблона, сохраненного в LSB доступного слова, затем с другой половиной, сохраненной в MSB. На рисунке 279 приведен пример обработки последовательности режима упаковки данных. Два кадра данных отправляются после одного 16-разрядного доступа к регистру  $SPIx\_DR$  передатчика. Эта последовательность может генерировать только одно событие  $RXNE$  в приемнике, если порог RXFIFO установлен в 16 бит ( $FRXTH = 0$ ). Затем получателю необходимо получить доступ к обоим кадрам данных с помощью одного 16-разрядного чтения  $SPIx\_DR$  в качестве ответа на это одно событие  $RXNE$ . Значение порога  $RxFIFO$  и следующий доступ на чтение всегда должны быть выровнены со стороны приемника, поскольку данные могут быть потеряны, если они не находятся в линии.

Конкретная проблема возникает, если необходимо обработать нечетное число таких «вписывается в один байт» фреймов данных. На передающей стороне достаточно записать последний

кадр данных любой нечетной последовательности с 8-битным доступом к SPIx\_DR. Получатель должен изменить пороговый уровень Rx\_FIFO для последнего кадра данных, принятого в нечетной последовательности кадров, чтобы генерировать событие RXNE.

Рисунок 279. Данные упаковки в FIFO для передачи и приема



### Связь с использованием DMA (прямая адресация памяти)

Для работы на максимальной скорости и облегчения процесса чтения / записи данных регистра, необходимого во избежание переполнения, SPI имеет функцию DMA, которая реализует простой протокол запроса / подтверждения.

Доступ к DMA запрашивается, когда бит разрешения TXE или RXNE в регистре SPIx\_CR2 установлен. Отдельные запросы должны выдаваться буферам Tx и Rx.

- При передаче запрос DMA выдается каждый раз, когда TXE устанавливается в 1. Затем DMA записывает в регистр SPIx\_DR.

- При приеме запрос DMA выдается каждый раз, когда RXNE устанавливается в 1. Затем DMA считывает регистр SPIx\_DR.

См. Рис. 280 на Рис. 283.

Когда SPI используется только для передачи данных, можно включить только канал SPI Tx DMA. В этом случае флаг OVR устанавливается, потому что полученные данные не читаются. Когда SPI используется только для приема данных, можно включить только канал SPI Rx DMA.

В режиме передачи, когда DMA записал все данные, подлежащие передаче (флаг TCIF установлен в регистре DMA\_ISR), флаг BSY можно контролировать, чтобы гарантировать, что связь SPI завершена. Это необходимо, чтобы не испортить последнюю передачу до отключения SPI или входа в режим остановки. Программное обеспечение должно сначала подождать, пока FTLVL [1:0] = 00, а затем до BSY = 0.

При запуске обмена данными с использованием DMA, чтобы предотвратить появление ошибок в управлении каналом DMA, эти шаги должны выполняться в следующем порядке:

1. Включить буфер DMA Rx в бите RXDMAEN в регистре SPI\_CR2, если используется DMA Rx.
2. Включите потоки DMA для Tx и Rx в регистрах DMA, если используются потоки.
3. Включите буфер DMA Tx в бите TXDMAEN в регистре SPI\_CR2, если используется DMA Tx.
4. Включите SPI, установив бит SPE.

Пример кода см. В разделах Приложения A.17.5: Конфигурация главного SPI с примером кода DMA и A.17.6: Конфигурация slave-устройства SPI с примером кода DMA.

Чтобы закрыть связь, необходимо выполнить следующие действия для того, чтобы:

1. Отключите потоки DMA для Tx и Rx в регистрах DMA, если используются потоки.
2. Отключите SPI, выполнив процедуру отключения SPI.
3. Отключите буферы DX Tx и Rx, очистив биты TXDMAEN и RXDMAEN в регистре SPI\_CR2, если используются DMA Tx и / или DMA Rx.

**Упаковка с DMA** Если управление передачей осуществляется с помощью DMA (TXDMAEN и RXDMAEN, установленных в регистре SPIx\_CR2), то режим упаковки включается / отключается автоматически в зависимости от значения PSIZE, настроенного для SPI TX и канала SPI RX DMA. Если значение PSIZE канала DMA равно 16-битовому, а размер данных SPI меньше или равен 8-битовому, то режим упаковки разрешен. Затем DMA автоматически управляет операциями записи в регистр SPIx\_DR.

Если используется режим упаковки данных и количество передаваемых данных не кратно двум, должны быть установлены биты LDMA\_TX / LDMA\_RX. Затем SPI рассматривает только одни данные для передачи или приема, чтобы обслуживать последний перенос DMA (подробнее см. «Упаковка данных» на стр. 768).

#### **Диаграммы связи**

В этом разделе объясняются некоторые типичные схемы синхронизации. Эти схемы действительны независимо от того, обрабатываются ли события SPI посредством опроса, прерываний или DMA. Для простоты в качестве общего предположения используется установка LSBFIRST = 0, CPOL = 0 и CPHA = 1. Полная настройка потоков DMA не предусмотрена.

Следующие пронумерованные заметки являются общими для рисунка 280 на стр. 772 — рис. 283 на стр. 775.

1. Ведомое устройство начинает управлять линией MISO, когда активна NSS и включен SPI, и отключается от линии, когда один из них освобождается. Должно быть предусмотрено достаточное время для ведомого, чтобы заранее подготовить данные, предназначенные для ведущего, до начала транзакции.

На ведущем устройстве периферийное устройство SPI берет на себя управление сигналами MOSI и SCK (иногда также и сигналом NSS), только если включен SPI. Если SPI отключен, периферийное устройство SPI отключается от логики GPIO, поэтому уровни в этих линиях зависят исключительно от настройки GPIO.

2. На ведущем устройстве BSY остается активным между кадрами, если связь (тактовый сигнал) непрерывна. В подчиненном устройстве сигнал BSY всегда идет вниз, по меньшей мере, для одного тактового цикла между кадрами данных.

3. Сигнал TXE очищается, только если TxFIFO заполнен.

4. Процесс арбитража DMA начинается сразу после установки бита TXDMAEN. Прерывание TXE генерируется сразу после установки TXEIE. Поскольку сигнал TXE находится на активном уровне, передача данных в TxFIFO начинается до тех пор, пока TxFIFO не заполнится или передача DMA не завершится.

5. Если все данные, которые должны быть отправлены, могут поместиться в TxFIFO, флаг TCMA Tx TIF может быть поднят даже до начала связи на шине SPI. Этот флаг всегда поднимается до завершения транзакции SPI.

6. Значение CRC для пакета вычисляется непрерывно по кадрам в Регистры SPIx\_TxCRCR и SPIx\_RxCRCR. Информация CRC обрабатывается после того, как весь пакет данных завершен, либо автоматически DMA (канал Tx должен быть установлен на количество кадров данных, подлежащих обработке), либо SW (пользователь должен обрабатывать бит CRCNEXT во время обработки последнего кадра данных). Хотя значение CRC, вычисленное в SPIx\_TxCRCR, просто отправляется передатчиком, полученная информация CRC загружается в RxFIFO и затем сравнивается с содержимым регистра SPIx\_RxCRCR (здесь может быть поднят флаг ошибки CRC, если есть какая-либо разница).

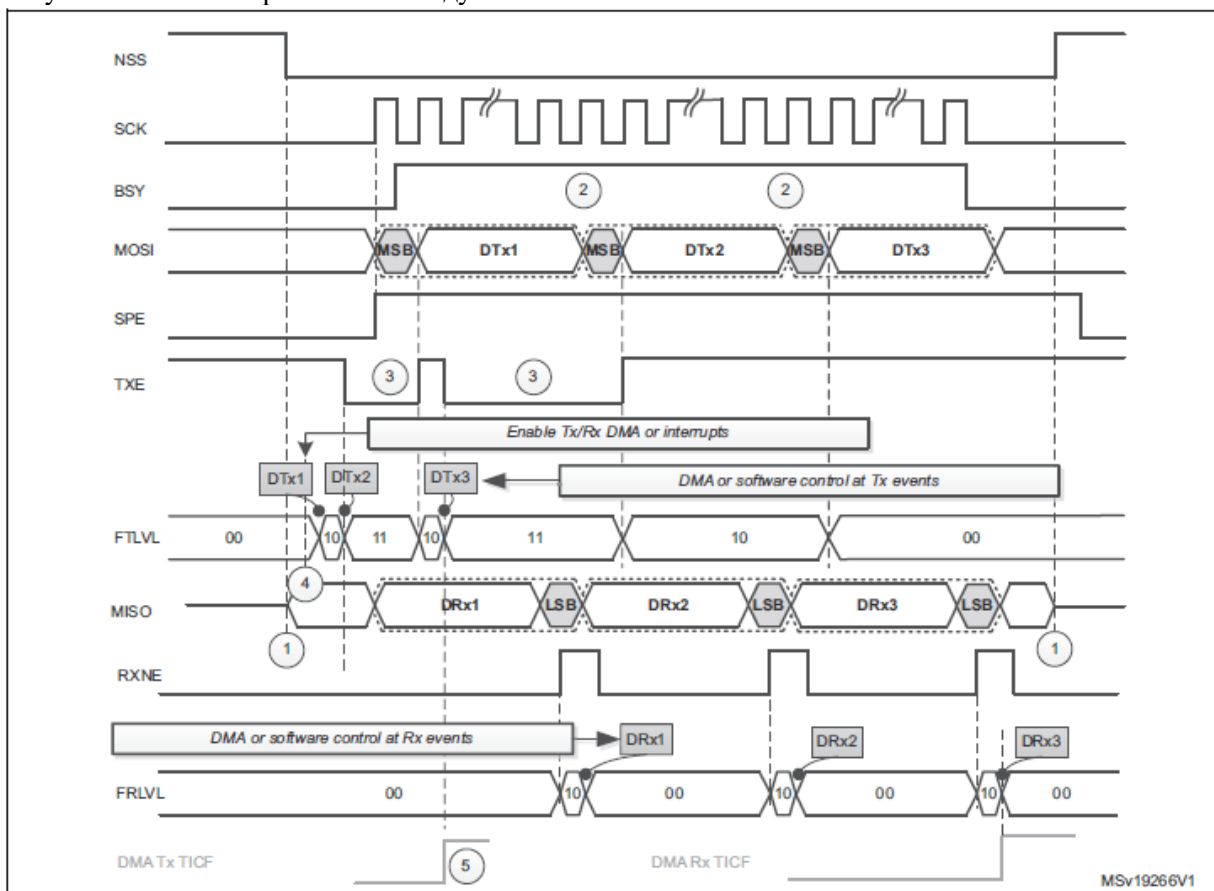
Именно поэтому пользователь должен позаботиться о том, чтобы сбросить эту информацию из FIFO либо программным обеспечением, считывающим все сохраненное содержимое RxFIFO, либо DMA, когда для Rx-канала задано правильное количество кадров данных (количество кадров данных + число Кадров CRC) (см. Настройки в примере предположения).

7. В режиме упаковки данных события TxE и RxNE сопряжены, и каждый доступ для чтения / записи к FIFO имеет ширину 16 бит, пока количество кадров данных не станет четным. Если TxFIFO равен 3/4, полный статус FTLVL остается на полном уровне FIFO. Вот почему последний нечетный кадр данных не может быть сохранен до того, как TxFIFO станет полным.

Этот кадр хранится в TxFIFO с 8-битным доступом либо программным обеспечением, либо автоматически DMA, когда установлен элемент управления LDMA\_TX.

8. Чтобы получить последний нечетный кадр данных в упакованном режиме, пороговое значение Rx должно быть изменено на 8-бит, когда обрабатывается последний кадр данных, либо программным значением FRXTH = 1, либо автоматически внутренним сигналом DMA, когда установлен LDMA\_RX.

Рисунок 280. Магистральная полнодуплексная связь



Предположения для примерной полной дуплексной связи:

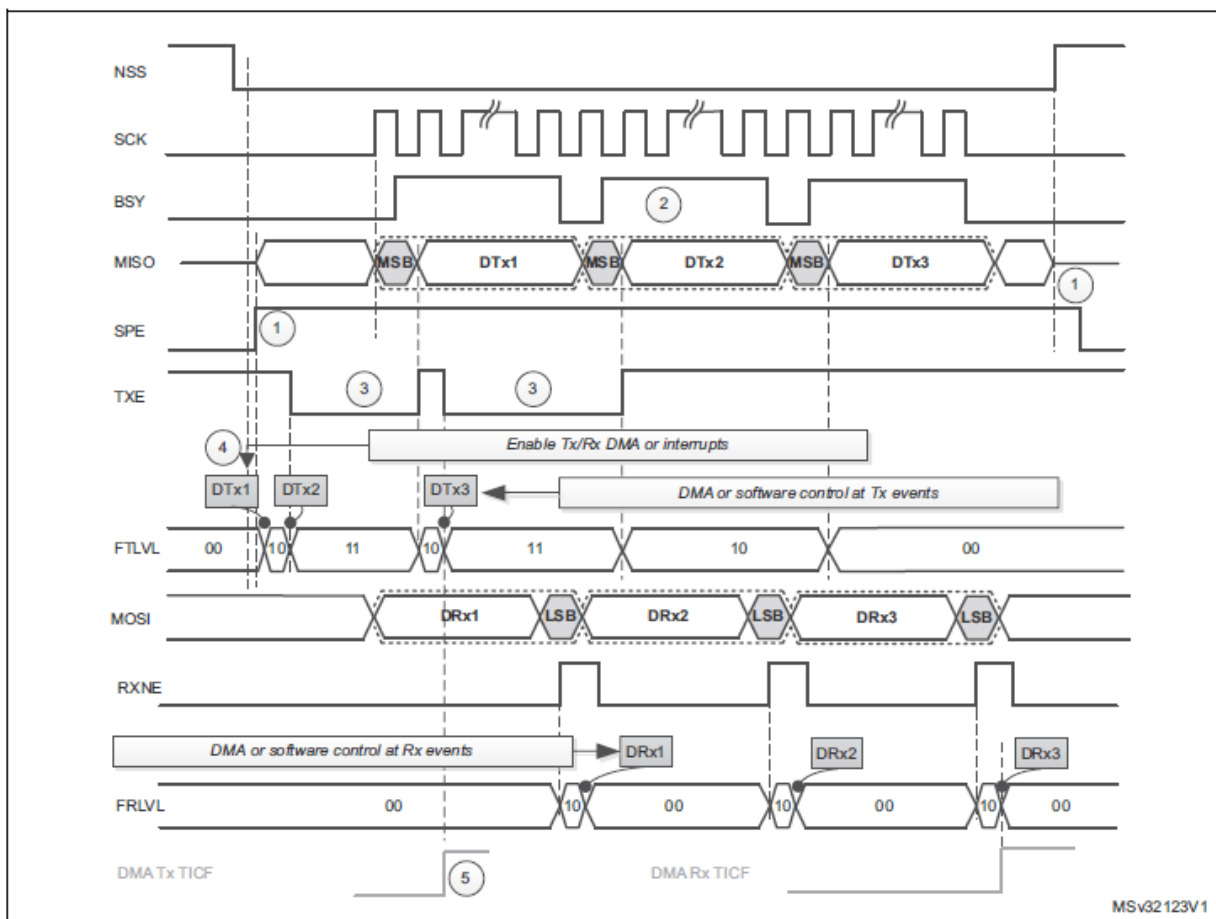
- Размер данных > 8 бит

Если используется DMA:

- Количество кадров Tx, транслируемых DMA, установлено равным 3
- Количество кадров Rx, транслируемых DMA, установлено равным 3

См. Также: диаграммы связи на стр. 771 для получения подробной информации о распространенных предположениях и примечаниях.

Рисунок 281. Ведомая полнодуплексная связь



Предположения для подчиненного полнодуплексного обмена данными Пример:

- Размер данных > 8 бит

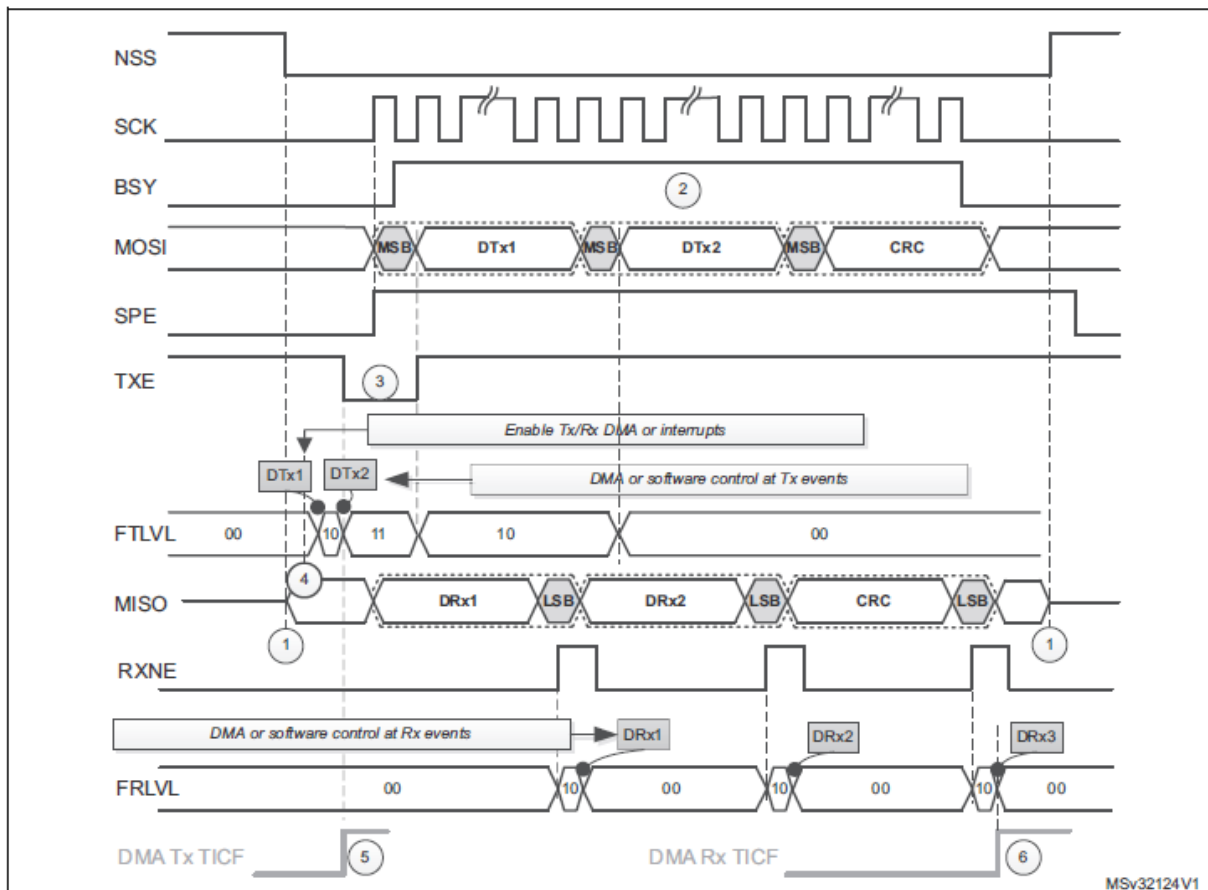
Если используется DMA:

- Количество кадров Tx, транслируемых DMA, установлено равным 3
- Количество кадров Rx, транслируемых DMA, установлено равным 3

См. Также: диаграммы связи на стр. 771 для получения подробной информации о распространенных предположениях и примечаниях.

Пример кода см. В разделе Приложение А.17.3: Пример кода полной дуплексной связи SPI.

Рисунок 282. Мастер-дуплексная связь с CRC



Предположения для основной полнодуплексной связи с примером CRC:

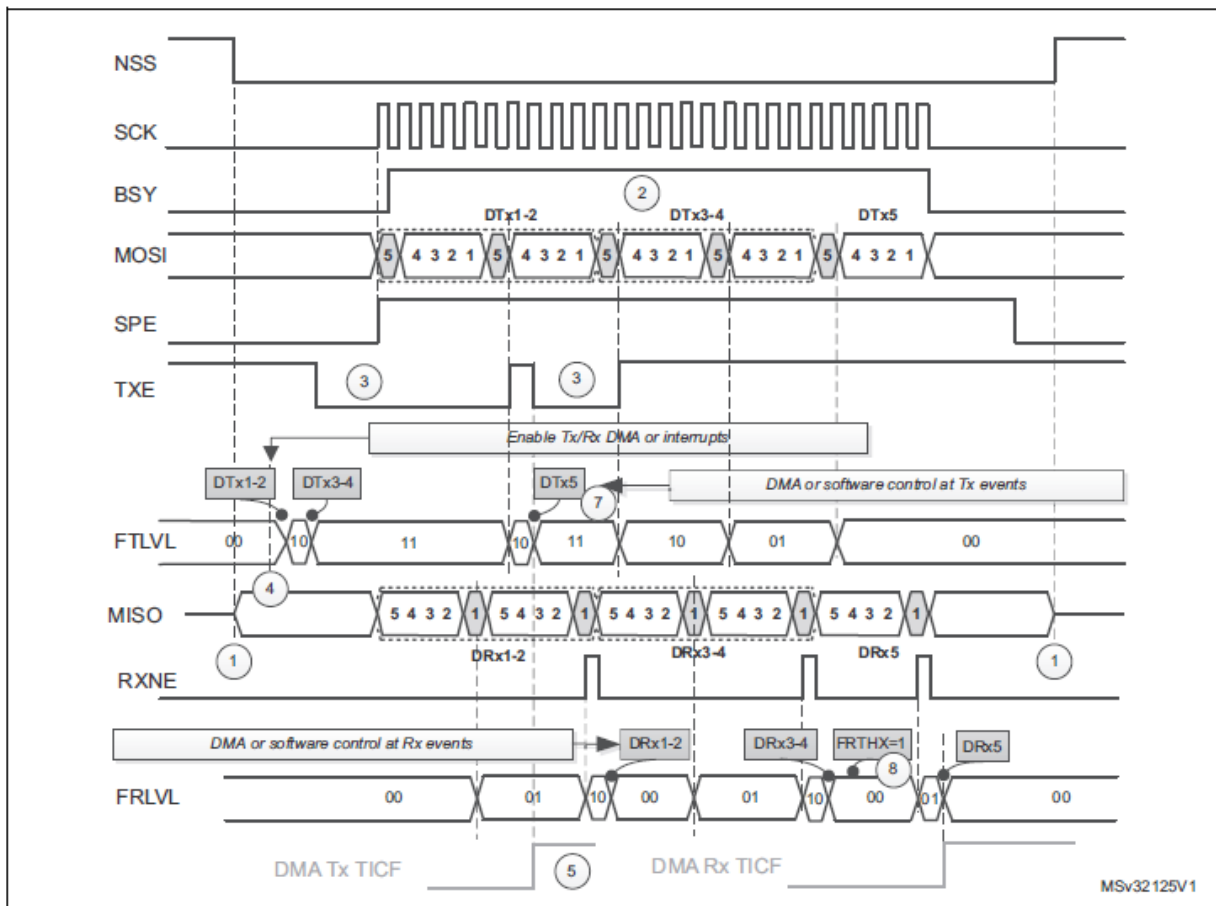
- Размер данных = 16 бит
- CRC включен

Если используется DMA:

- Количество кадров Tx, транслируемых DMA, установлено равным 2
- Количество кадров Rx, транслируемых DMA, установлено равным 3

См. Также: диаграммы связи на стр. 771 для получения подробной информации о распространенных предположениях и примечаниях.

Рисунок 283. Основная полнодуплексная связь в упакованном режиме



Предположения для основной полнодуплексной связи в упакованном режиме:

- Размер данных = 5 бит
- Чтение / запись FIFO выполняется в основном с помощью 16-битного доступа
- FRXTH = 0

Если используется DMA:

- Количество кадров Tx, подлежащих транзакции с помощью DMA, установлено равным 3
- Количество кадров Rx, подлежащих транзакции с помощью DMA, равно 3
- PSIZE для обоих каналов Tx и Rx DMA установлен в 16-бит • LDMA\_TX = 1 и LDMA\_RX

= 1

См. Также: Схемы связи на стр. 771 для получения подробной информации о распространенных предположениях и примечаниях.

#### 28.5.10 Флаги состояния SPI

Предусмотрено три состояния для приложения, чтобы полностью контролировать состояние шины SPI.

##### Пустой флаг буфера Tx (TXE)

Флаг TXE устанавливается, когда TXFIFO передачи имеет достаточно места для хранения данных для отправки.

Флаг TXE связан с уровнем TXFIFO. Флаг становится высоким и остается высоким до тех пор, пока уровень TXFIFO не станет ниже или равен 1/2 от глубины FIFO. Прерывание может быть сгенерировано, если бит TXEIE установлен в регистр SPIx\_CR2. Бит очищается автоматически, когда уровень TXFIFO становится больше 1/2.

##### Rx-буфер не пустой (RXNE)

Флаг RXNE устанавливается в зависимости от значения бита FRXTH в регистре SPIx\_CR2:

- Если установлен FRXTH, RXNE поднимается высоко и остается высоким до тех пор, пока уровень RXFIFO не будет больше или равен 1/4 (8 бит).

- Если FRXTH сброшен, RXNE поднимается высоко и остается высоким до тех пор, пока уровень RXFIFO не будет больше или равен 1/2 (16-бит).

Прерывание может быть сгенерировано, если бит RXNEIE установлен в регистр SPIx\_CR2.

RXNE автоматически очищается аппаратными средствами, если указанные выше условия больше не соответствуют действительности.

Флаг занятости (BSY)

Флаг BSY устанавливается и очищается аппаратными средствами (запись в этот флаг не имеет эффекта).

Когда установлен BSY, это указывает на то, что на SPI идет передача данных (шина SPI занята).

Флаг BSY может использоваться в определенных режимах для определения конца передачи, чтобы программное обеспечение могло отключить SPI или его периферийные тактовые импульсы перед входом в режим с низким энергопотреблением, который не обеспечивает тактов для периферийного устройства. Это предотвратит повреждение последней передачи.

Флаг BSY также полезен для предотвращения конфликтов записи в системе с несколькими хозяевами.

Флаг BSY очищается при любом из следующих условий:

- Когда SPI правильно отключен
- Когда ошибка обнаружена в режиме Master (бит MODF установлен в 1)
- В режиме мастера, когда он заканчивает передачу данных и новые данные не готовы к отправке
- В режиме ведомого, когда флаг BSY установлен в '0', по крайней мере, для одного тактового цикла SPI между каждой передачей данных.

Примечание: Когда следующая передача может быть обработана сразу же ведущим устройством (например, если ведущее устройство находится в режиме «Только получение» или его FIFO передачи не пуст), связь непрерывна, и флаг BSY остается установленным в «1» между передачами на сторону мастера. Хотя это не относится к подчиненному устройству, рекомендуется использовать всегда флаги TXE и RXNE (вместо флагов BSY) для обработки операций передачи или приема данных.

### 28.5.11 Флаги ошибок SPI

Прерывание SPI генерируется, если установлен один из следующих флагов ошибки, и прерывание разрешено установкой бита ERRIE.

Флаг переполнения (OVR)

Условие переполнения возникает, когда данные принимаются ведущим или ведомым, а RXFIFO не хватает места для хранения этих полученных данных. Это может произойти, если программное обеспечение или DMA не имеют достаточно времени для чтения ранее принятых данных (сохраненных в RXFIFO) или когда пространство для хранения данных ограничено, например. RXFIFO недоступен, когда CRC включен в режиме только приема, поэтому в этом случае буфер приема ограничен буфером данных (см. Раздел 28.5.14: Расчет CRC).

Когда возникает переполнение, новое принятое значение не перезаписывает предыдущее в RXFIFO. Новое принятое значение отбрасывается, и все данные, переданные впоследствии, теряются. Очистка бита OVR выполняется путем доступа на чтение к регистру SPI\_DR, за которым следует доступ на чтение к регистру SPI\_SR.

Ошибка режима (MODF)

Ошибка режима возникает, когда ведущее устройство имеет свой внутренний сигнал NSS (вывод NSS в аппаратном режиме NSS или бит SSI в программном режиме NSS) вытягивается низким уровнем. Это автоматически устанавливает бит MODF. Сбой основного режима влияет на интерфейс SPI следующими способами:

- Бит MODF установлен и прерывание SPI генерируется, если установлен бит ERRIE.

- Бит SPE очищается. Это блокирует все выходные данные с устройства и отключает интерфейс SPI.

- Бит MSTR очищается, что приводит к переходу устройства в подчиненный режим.

Для очистки бита MODF используйте следующую последовательность программного обеспечения:

1. Сделайте доступ на чтение или запись к регистру SPIx\_SR, когда установлен бит MODF.
2. Затем запишите в регистр SPIx\_CR1.

Чтобы избежать конфликтов нескольких ведомых устройств в системе, содержащей несколько микроконтроллеров, вывод NSS должен быть поднят высоко во время последовательности очистки бита MODF. Биты SPE и MSTR могут быть восстановлены в исходное состояние после этой последовательности очистки. В качестве защиты аппаратное обеспечение не позволяет устанавливать биты SPE и MSTR, пока установлен бит MODF. В ведомом устройстве бит MODF не может быть установлен, кроме как в результате предыдущего конфликта с несколькими хозяевами.

#### Ошибка CRC (CRCERR)

Этот флаг используется для проверки достоверности значения, полученного при установке бита CRCEN в регистре SPIx\_CR1. Флаг CRCERR в регистре SPIx\_SR устанавливается, если значение, полученное в сдвиговом регистре, не соответствует значению SPIx\_RXCRCR получателя. Этот флаг очищается программным обеспечением.

#### Ошибка формата кадра TI (FRE)

Ошибка формата кадра TI обнаруживается, когда импульс NSS возникает во время непрерывной связи, когда SPI работает в подчиненном режиме и сконфигурирован в соответствии с протоколом режима TI. При возникновении этой ошибки флаг FRE устанавливается в регистр SPIx\_SR. SPI не отключается, когда возникает ошибка, импульс NSS игнорируется, и SPI ожидает следующего импульса NSS перед началом новой передачи. Данные могут быть повреждены, так как обнаружение ошибок может привести к потере двух байтов данных.

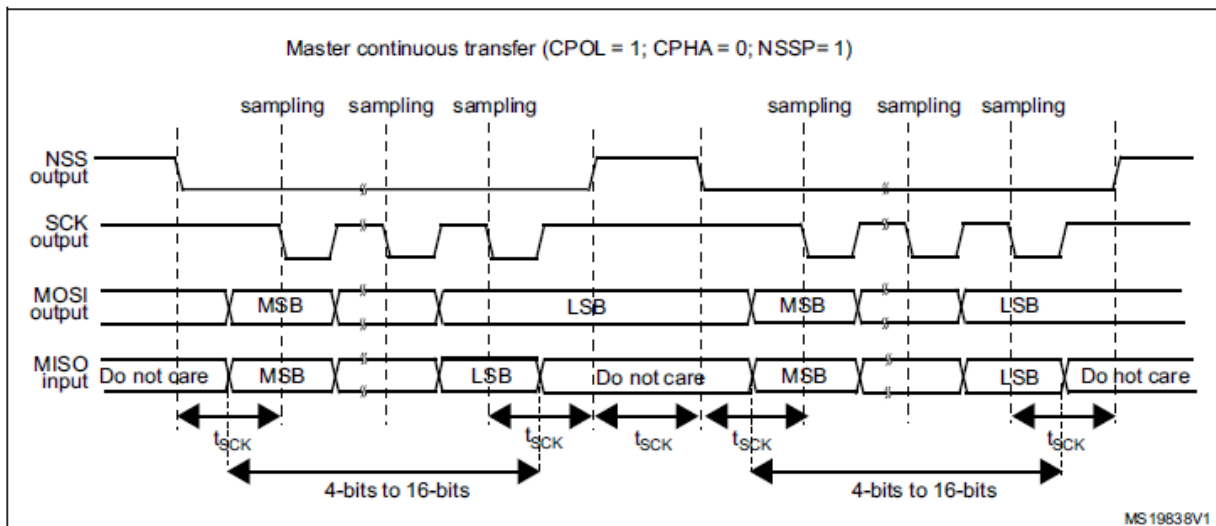
Флаг FRE очищается при чтении регистра SPIx\_SR. Если установлен бит ERRIE, при обнаружении ошибки NSS генерируется прерывание. В этом случае SPI должен быть отключен, так как согласованность данных больше не гарантируется, и связь должна возобновляться мастером, когда ведомый SPI снова включается.

### 28.5.12 Импульсный режим NSS

Этот режим активируется битом NSSP в регистре SPIx\_CR2, и он вступает в силу только в том случае, если интерфейс SPI сконфигурирован как master-устройство Motorola SPI (FRF = 0) с захватом по первому фронту (SPIx\_CR1 CPHA = 0, параметр CPOL игнорируется). При активации импульс NSS генерируется между двумя последовательными передачами кадров данных, когда NSS остается на высоком уровне в течение, по меньшей мере, одного периода синхронизации. Этот режим позволяет ведомому устройству фиксировать данные. Импульсный режим NSSP предназначен для приложений с одной парой ведущий-ведомый.

На рисунке 284 показано управление штырями NSS, когда включен импульсный режим NSSP.

Рисунок 284. Генерация импульсов NSSP в режиме ведущего устройства Motorola SPI



Примечание: Аналогичное поведение встречается при CPOL = 0. В этом случае край выборки представляет собой возрастающий фронт SCK, а утверждение NSS и снятие с отрисовки относятся к этому краю выборки.

### 28.5.13 Режим TI

Протокол TI в режиме ведущего

Интерфейс SPI совместим с протоколом TI. FRF бит регистра SPIx\_CR2 может использоваться для настройки SPI, чтобы он был совместим с этим протоколом.

Полярность и фаза тактового сигнала вынуждены соответствовать требованиям протокола TI независимо от значений, установленных в регистре SPIx\_CR1. Управление NSS также специфично для протокола TI, который делает невозможной в этом случае конфигурацию управления NSS через регистры SPIx\_CR1 и SPIx\_CR2 (SSM, SSI, SSOE).

В подчиненном режиме предварительный делитель скорости SPI используется для управления моментом, когда состояние штыря MISO изменяется на HiZ, когда завершается текущая транзакция (см. Рис. 285). Можно использовать любую скорость передачи, что позволяет определить этот момент с оптимальной гибкостью. Тем не менее, скорость передачи в бодах обычно устанавливается равной скорости передачи данных от ведущего устройства. Задержка для сигнала MISO, чтобы стать HiZ (trelease), зависит от внутренней повторной синхронизации и от значения скорости передачи, установленного через биты BR [2: 0] в регистре SPIx\_CR1. Он определяется по формуле:

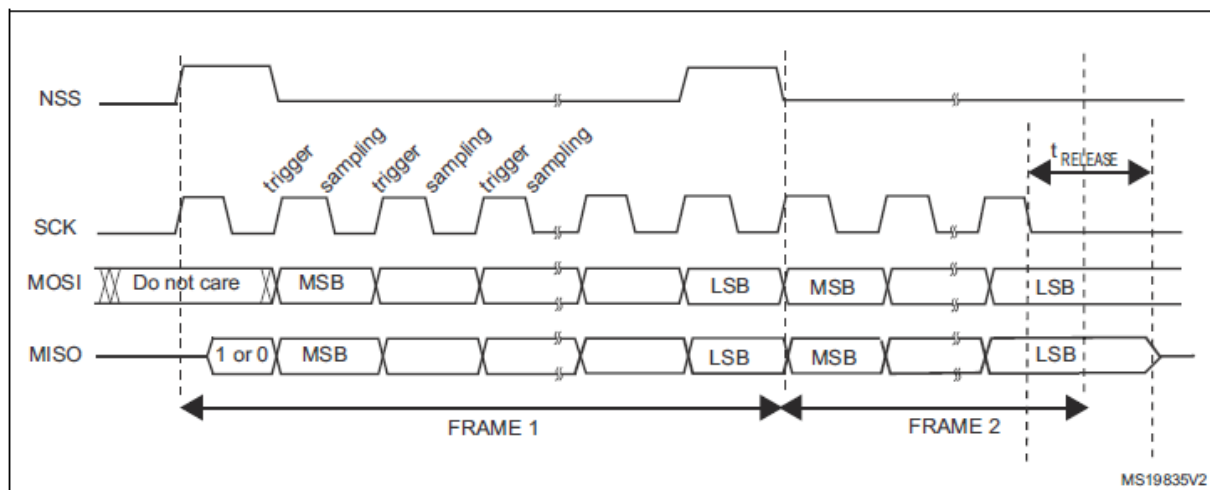
$$\frac{t_{\text{baud rate}}}{2} + 4 \times t_{\text{pclk}} < t_{\text{release}} < \frac{t_{\text{baud rate}}}{2} + 6 \times t_{\text{pclk}}$$

Если ведомое устройство обнаруживает неупорядоченный импульс NSS во время транзакции фрейма данных, устанавливается флаг TIFRE.

Если размер данных равен 4 битам или 5 битам, мастер в полнодуплексном режиме или режиме только для передачи использует протокол с еще одним фиктивным битом данных, добавленным после LSB. TI импульс NSS генерируется выше этого тактового битового цикла вместо LSB в каждом периоде.

Эта функция недоступна для коммуникаций Motorola SPI (бит FRF установлен в 0).

Рис.285: Передача режима TI показывает формы сигналов SPI, когда выбран режим TI. Рисунок 285. Передача режима TI



#### 28.5.14 Расчет CRC

Для проверки надежности используются два отдельных калькулятора CRC.

Переданных и полученных данных. SPI предлагает вычисления CRC8 или CRC16 независимо от длины данных кадра, которая может быть установлена на 8-бит или 16-бит. Для всех остальных кадров данных CRC недоступен.

CRC-расчет CRC-расчет включается путем установки бита CRCEN в регистре SPIx\_CR1 до включения SPI (SPE = 1). Значение CRC рассчитывается с использованием нечетного программируемого полинома для каждого бита. Вычисление обрабатывается на границе тактовой синхронизации, определенной битами CPHA и CPOL в регистре SPIx\_CR1. Вычисленное значение CRC автоматически проверяется в конце блока данных, а также для передачи, управляемой CPU или DMA. Когда обнаружено несоответствие между CRC, внутренне рассчитанным по принятым данным, и CRC, посланным передатчиком, устанавливается флаг CRCERR, указывающий на ошибку повреждения данных. Правильная процедура обработки вычисления CRC зависит от конфигурации SPI и выбранного управления передачей.

Примечание. Значение полинома должно быть нечетным. Четные значения не поддерживаются.

Передача CRC управляется процессором

Коммуникация начинается и продолжается нормально, пока последний регистр данных не будет отправлен или принят в регистре SPIx\_DR. Затем бит CRCNEXT должен быть установлен в регистре SPIx\_CR1, чтобы указать, что транзакция кадра CRC будет следовать после транзакции текущего обработанного кадра данных. Бит CRCNEXT должен быть установлен до конца последней транзакции фрейма данных. Калькуляция CRC замораживается во время транзакции CRC.

Полученный CRC хранится в RXFIFO как байт данных или слово. Именно поэтому в режиме CRC буфер приема должен рассматриваться как единственный 16-битный буфер, используемый одновременно для приема только одного кадра данных.

Для транзакции формата CRC обычно требуется еще один кадр данных для связи в конце последовательности данных. Однако при установке 8-битного кадра данных, проверяемого 16-битным CRC, для отправки полного CRC необходимо еще два кадра.

Когда принимаются последние данные CRC, выполняется автоматическая проверка сравнения полученного значения и значения в регистре SPIx\_RXCRC. Программное обеспечение должно проверять флаг CRCERR в регистре SPIx\_SR, чтобы определить, были ли повреждены данные или нет. Программное обеспечение очищает флаг CRCERR, записывая в него «0».

После приема CRC значение CRC сохраняется в RXFIFO и должно быть прочитано в регистре SPIx\_DR, чтобы очистить флаг RXNE.

### Передача CRC, управляемая DMA

Когда связь SPI разрешена с CRC-соединением и режимом DMA, передача и прием CRC в конце связи происходит автоматически (за исключением чтения данных CRC в режиме только приема). Бит CRCNEXT не должен обрабатываться программным обеспечением. Счетчик для канала DMA передачи SPI должен быть установлен на количество кадров данных для передачи, исключая кадр CRC. Со стороны получателя полученное значение CRC обрабатывается автоматически DMA в конце транзакции, но пользователь должен позаботиться о сбросе полученной информации CRC из RXFIFO, так как он всегда загружается в нее. В полнодуплексном режиме счетчик приемного канала DMA может быть установлен на число кадров данных для приема, включая CRC, что означает, например, в конкретном случае 8-битного кадра данных, проверяемого 16-битным CRC:

$$\text{DMA\_RX} = \text{Numb\_of\_data} + 2$$

В режиме только приема, счетчик канала приема DMA должен содержать только количество переданных данных, исключая вычисление CRC. Затем, основываясь на полной передаче от DMA, все значения CRC должны быть прочитаны программным обеспечением из FIFO, поскольку он работает как один буфер в этом режиме.

В конце передачи данных и CRC флаг CRCERR в регистре SPIx\_SR устанавливается, если во время передачи произошел сбой.

Если используется режим упаковки, бит LDMA\_RX требует управления, если количество данных нечетное.

Сброс значений SPIx\_TXCRC и SPIx\_RXCRC

Значения SPIx\_TXCRC и SPIx\_RXCRC автоматически очищаются, когда новые данные дискретизируются после фазы CRC. Это позволяет использовать циклический режим DMA (недоступно в режиме только для приема) для передачи данных без прерывания (несколько блоков данных, покрываемых промежуточными этапами проверки CRC).

Если SPI отключен во время связи, необходимо соблюдать следующую последовательность действий:

1. Отключите SPI.
2. Очистите бит CRCEN
3. Включите бит CRCEN
4. Включить SPI.

Примечание: Когда SPI находится в подчиненном режиме, калькулятор CRC чувствителен к тактовому импульсу входного сигнала SCK, как только бит CRCEN установлен, и это имеет место независимо от значения бита SPE. Чтобы избежать любого неправильного вычисления CRC, программное обеспечение должно разрешить вычисление CRC только тогда, когда часы стабильны (в устойчивом состоянии).

Когда интерфейс SPI сконфигурирован как подчиненный, внутренний сигнал NSS должен оставаться низким во время транзакции фазы CRC после отпускания сигнала CRCNEXT. Вот почему расчет CRC нельзя использовать в режиме NSS Pulse, когда аппаратный режим NSS должен применяться в подчиненном режиме (подробнее см. Лист исправлений продукта).

В режиме TI, несмотря на то, что настройка фазы и тактовой частоты синхроимпульса является фиксированной и независимой от регистра SPIx\_CR1, соответствующая настройка CPHA = 0 CPOL = 1 должна быть сохранена в регистре SPIx\_CR1 в любом случае, если применяется CRC. Кроме того, расчет CRC должен быть сброшен между сеансами с помощью последовательности отключения SPI с повторным включением описанного выше бита CRCEN как на главной, так и на ведомой стороне, иначе вычисление CRC может быть повреждено в этом конкретном режиме.

## 28.6 Прерывания SPI

Во время связи с SPI прерывания могут генерироваться следующими событиями:

- Передача TXFIFO готова для загрузки
- Данные, полученные в Receive RXFIFO

- Неисправность главного режима
- Ошибка переполнения
- Ошибка формата кадра TI
- Ошибка протокола CRC

Прерывания могут быть включены и отключены отдельно.

Таблица 112. Запросы прерываний SPI

| Interrupt event                    | Event flag | Enable Control bit |
|------------------------------------|------------|--------------------|
| Transmit TXFIFO ready to be loaded | TXE        | TXEIE              |
| Data received in RXFIFO            | RXNE       | RXNEIE             |
| Master Mode fault event            | MODF       | ERRIE              |
| Overrun error                      | OVR        |                    |
| TI frame format error              | FRE        |                    |
| CRC protocol error                 | CRCERR     |                    |

Пример кода см. В разделе Приложение А.17.4: Пример кода прерывания SPI

## **28.7 I2S functional description**

**Page 782**

## **28.8 I2S interrupts**

**Page 800**

## 28.9 Регистры SPI и I<sup>2</sup>S

Доступ к периферийным регистрам может быть полусловом (16-бит) или словом (32-разрядный). SPI\_DR, кроме того, может быть доступен 8-битным доступом.

### 28.9.1 Регистр1 управления SPI (SPIx\_CR1)

Смещение адреса: 0x00

Значение сброса: 0x0000

| 15           | 14         | 13        | 12          | 11   | 10         | 9   | 8   | 7            | 6   | 5        | 4  | 3  | 2    | 1    | 0    |
|--------------|------------|-----------|-------------|------|------------|-----|-----|--------------|-----|----------|----|----|------|------|------|
| BIDI<br>MODE | BIDI<br>OE | CRC<br>EN | CRC<br>NEXT | CRCL | RX<br>ONLY | SSM | SSI | LSB<br>FIRST | SPE | BR [2:0] |    |    | MSTR | CPOL | CPHA |
| rw           | rw         | rw        | rw          | rw   | rw         | rw  | rw  | rw           | rw  | rw       | rw | rw | rw   | rw   | rw   |

**Бит 15 BIDIMODE:** Разрешает двунаправленный режим передачи данных. Этот бит обеспечивает полудуплексную связь с использованием единой однонаправленной линии передачи данных. Держите RXONLY чистым, когда активен двунаправленный режим.

**0:** Выбран однонаправленный режим данных с двумя линиями

**1:** Выбран двунаправленный режим данных по одной линии

Примечание: Этот бит не используется в режиме I<sup>2</sup>S.

**Бит 14 BIDIOE:** Разрешение выхода в двунаправленном режиме

Этот бит в сочетании с битом BIDIMODE выбирает направление передачи в двунаправленном режиме

**0:** Выход отключен (режим только для приема)

**1:** Выход включен (только для передачи)

Примечание. В ведущем режиме используется контакт MOSI, а в подчиненном режиме используется контакт MISO.

Этот бит не используется в режиме I<sup>2</sup>S.

**Бит 13 CRCEN:** аппаратное обеспечение вычисления CRC

**0:** расчет CRC отключен

**1:** Расчет CRC включен

Примечание: Этот бит должен быть записан, только если SPI отключен (SPE = '0') для правильной работы.

Этот бит не используется в режиме I<sup>2</sup>S.

**Бит 12 CRCNEXT:** передать CRC next

**0:** следующее значение передачи — из буфера Tx

**1:** следующее значение передачи — из регистра Tx CRC

Примечание: Этот бит должен быть записан, как только последние данные будут записаны в регистр SPIx\_DR.

Этот бит не используется в режиме I<sup>2</sup>S.

**Бит 11 CRCL:** длина CRC

Этот бит устанавливается и очищается программным обеспечением для выбора длины CRC.

**0:** 8-разрядная длина CRC

**1:** 16-разрядная длина CRC

Примечание: Этот бит должен быть записан, только если SPI отключен (SPE = '0') для правильной работы.

Этот бит не используется в режиме I<sup>2</sup>S.

**Бит 10 RXONLY:** включен только режим приема.

Этот бит обеспечивает симплексную связь с использованием единственной однонаправленной линии для приема данных исключительно. Сохраняйте бит BIDIMODE битом, когда активен режим только приема. Этот бит полезен также в многозабойной системе, в которой к данному ведомому устройству не обращаются, выход от раба, к которому осуществляется доступ, не поврежден.

**0:** Полнодуплексный (передача и прием)

**1:** Выход отключен (режим только для приема)

Примечание: Этот бит не используется в режиме I<sup>2</sup>S.

**Бит 9 SSM:** ведомое устройство программного обеспечения

Когда бит SSM установлен, вход штыря NSS заменяется значением от бита SSI.

**0:** управление ведомым программным обеспечением отключено

**1:** Управление ведомым программным обеспечением включено

Примечание: Этот бит не используется в режиме I<sup>2</sup>S и режиме SPI TI.

**Бит 8 SSI:** выбор внутреннего подчиненного устройства

Этот бит действует только тогда, когда бит SSM установлен. Значение этого бита принудительно подается на вывод NSS, и значение ввода / вывода штыря NSS игнорируется.

Примечание: Этот бит не используется в режиме I<sup>2</sup>S и режиме SPI TI.

**Бит 7 LSBFIRST:** формат кадра

**0:** данные сначала передаются / принимаются с MSB

**1:** данные сначала передаются / принимаются с LSB

Примечание: Этот бит не следует изменять, если связь продолжается.

Этот бит не используется в режиме I<sup>2</sup>S и режиме SPI TI.

**Бит 6 SPE:** включение SPI

**0:** Периферийное отключено

**1:** Включено периферийное устройство

Примечание. При отключении SPI выполните процедуру, описанную в процедуре для отключения SPI на стр. 767.

Этот бит не используется в режиме I<sup>2</sup>S.

**Биты 5: 3 BR [2: 0]:** управление скоростью передачи

**000:** fPCLK / 2

**001:** fPCLK / 4

**010:** fPCLK / 8

**011:** fPCLK / 16

**100:** fPCLK / 32

**101:** fPCLK / 64

**110:** fPCLK / 128

**111:** fPCLK / 256

Примечание. Эти биты не должны изменяться, когда связь продолжается.

Этот бит не используется в режиме I<sup>2</sup>S.

**Бит 2 MSTR:** выбор мастера

**0:** Конфигурация ведомого устройства

**1:** Основная конфигурация

Примечание. Этот бит не следует изменять, если связь продолжается.

Этот бит не используется в режиме I<sup>2</sup>S.

**Бит1 CPOL:** полярность синхронизации

**0:** SCK 0 (земля) при простое

**1:** SCK 1 (питание) при простое

Примечание. Этот бит не следует изменять, если связь продолжается.

Этот бит не используется в режиме I<sup>2</sup>S и режиме SPI TI, за исключением случая, когда CRC применяется в режиме TI.

**Бит 0 CPHA:** фаза синхронизации

**0:** первый тактовый переход — это первый край захвата данных

**1:** Второй тактовый переход — это первый край захвата данных

Примечание. Этот бит не следует изменять, если связь продолжается.

Этот бит не используется в режиме I<sup>2</sup>S и режиме SPI TI, за исключением случая, когда CRC применяется в режиме TI.

## 28.9.2 Регистр2 управления SPI (SPIx\_CR2)

Смещение адреса: 0x04

Сбросить значение: 0x0700

| 15   | 14      | 13      | 12    | 11       | 10 | 9  | 8  | 7     | 6      | 5     | 4   | 3    | 2    | 1       | 0       |
|------|---------|---------|-------|----------|----|----|----|-------|--------|-------|-----|------|------|---------|---------|
| Res. | LDMA_TX | LDMA_RX | FRXTH | DS [3:0] |    |    |    | TXEIE | RXNEIE | ERRIE | FRF | NSSP | SSOE | TXDMAEN | RXDMAEN |
|      | rw      | rw      | rw    | rw       | rw | rw | rw | rw    | rw     | rw    | rw  | rw   | rw   | rw      | rw      |

**Бит 15** Зарезервирован, должен храниться в значении сброса.

**Бит 14 LDMA\_TX:** последний перенос DMA для передачи

Этот бит используется в режиме упаковки данных, чтобы определить, является ли общее количество данных для передачи по DMA нечетным или четным. Это имеет значение только в том случае, если бит TXDMAEN в регистре SPIx\_CR2 установлен и используется режим упаковки (длина данных = <8 бит и доступ для записи к SPIx\_DR составляет 16 бит). Он должен быть записан, когда SPI отключен (SPE = 0 в регистре SPIx\_CR1).

**0:** Количество данных для передачи четное

**1:** Количество данных для передачи нечетное

Примечание: см. Процедуру отключения SPI на стр. 767, если установлен бит CRCEN.

Этот бит не используется в режиме I<sup>2</sup>S.

**Бит 13 LDMA\_RX:** последний перенос DMA для приема

Этот бит используется в режиме упаковки данных, чтобы определить, является ли общее количество данных для приема DMA четным или нечетным. Это имеет значение только в том случае, если бит RXDMAEN в регистре SPIx\_CR2 установлен и используется режим упаковки (длина данных = <8 бит и доступ для записи к SPIx\_DR имеет ширину 16 бит). Он должен быть записан, когда SPI отключен (SPE = 0 в регистре SPIx\_CR1).

**0:** Количество данных для передачи четное

**1:** Количество данных для передачи нечетное

Примечание: см. Процедуру отключения SPI на стр. 767, если установлен бит CRCEN.

Этот бит не используется в режиме I<sup>2</sup>S.

**Бит 12 FRXTH:** порог приема FIFO

Этот бит используется для установки порога RXFIFO, который запускает событие RXNE

**0:** Событие RXNE генерируется, если уровень FIFO больше или равен 1/2 (16 бит)

**1:** Событие RXNE генерируется, если уровень FIFO больше или равен 1/4 (8 бит)

Примечание. Этот бит не используется в режиме I<sup>2</sup>S.

**Биты 11: 8 DS [3: 0]:** размер данных

Эти биты конфигурируют длину данных для передач SPI:

**0000:** Не используется

**0001:** Не используется

**0010:** Не используется

**0011:** 4-бит

**0100:** 5 бит

**0101:** 6-бит

**0110:** 7-бит

**0111:** 8-бит

**1000:** 9 бит

**1001:** 10-бит

**1010:** 11-бит

**1011:** 12-бит

**1100:** 13 бит

**1101:** 14 бит

**1110:** 15 бит

**1111:** 16 бит

Если программное обеспечение пытается записать одно из значений «Не используется», оно принудительно переходит на значение «0111» (8 бит).

Примечание. Этот бит не используется в режиме I<sup>2</sup>S.

**Бит 7 TXEIE:** Разрешение прерывания буфера Tx буфера

**0:** прерывание TXE замаскировано

**1:** Прерывание TXE не замаскировано. Используется для генерации запроса прерывания, когда установлен флаг TXE.

**Бит 6 RXNEIE:** разрешение буфера RX не пустых прерываний

**0:** прерывание RXNE замаскировано

**1:** прерывание RXNE не замаскировано. Используется для генерации запроса прерывания, когда установлен флаг RXNE.

**Бит 5 ERRIE:** Разрешение ошибки прерывания

Этот бит управляет генерацией прерывания при возникновении ошибки (CRCERR, OVR, MODF в режиме SPI, FRE в режиме TI и UDR, OVR и FRE в режиме I<sup>2</sup>S).

**0:** Прерывание ошибки маскируется

**1:** Прерывание ошибки разрешено

**Бит 4 FRF:** формат кадра

**0:** Режим SPI Motorola

**1:** Режим SPI TI

Примечание: Этот бит должен быть записан только тогда, когда SPI отключен (SPE = 0).

Этот бит не используется в режиме I<sup>2</sup>S.

**Бит 3 NSSP:** управление импульсом NSS

Этот бит используется только в режиме ведущего. Он позволяет SPI генерировать импульс NSS между двумя последовательными данными при выполнении непрерывных передач. В случае одиночной передачи данных, он выдает высокий уровень NSS-штыря после передачи. Это не имеет значения, если CPHA = '1', или FRF = '1'.

**0:** Нет импульса NSS

**1:** Сгенерирован импульс NSS

Примечание: Этот бит должен быть записан только тогда, когда SPI отключен (SPE = 0).

Этот бит не используется в режиме I<sup>2</sup>S и режиме SPI TI.

**Бит 2 SSOE:** разрешение выхода SS

**0:** выход SS отключен в главном режиме, и интерфейс SPI может работать в конфигурации с несколькими ведущими

**1:** выход SS включен в главном режиме и при включенном интерфейсе SPI. Интерфейс SPI не может работать в среде с несколькими хозяевами.

Примечание: Этот бит не используется в режиме I<sup>2</sup>S и режиме SPI TI.

**Бит 1 TXDMAEN:** Разрешение DX буфера Tx

Когда этот бит установлен, запрос DMA генерируется всякий раз, когда устанавливается флаг TXE.

**0:** Tx буфер DMA отключен

**1:** Tx-буфер DMA включен

**Бит 0 RXDMAEN:** включить буферизацию Rx-буфера

Когда этот бит установлен, запрос DMA генерируется всякий раз, когда установлен флаг RXNE.

**0:** Rx-буфер DMA отключен

**1:** Rx-буфер DMA включен

### 28.9.3 Регистр состояния SPI (SPIx\_SR)

Смещение адреса: 0x08

Сбросить значение: 0x0002

| 15   | 14   | 13   | 12        | 11 | 10         | 9 | 8   | 7   | 6   | 5    | 4          | 3   | 2      | 1   | 0    |
|------|------|------|-----------|----|------------|---|-----|-----|-----|------|------------|-----|--------|-----|------|
| Res. | Res. | Res. | FTLV[1:0] |    | FRLVL[2:0] |   | FRE | BSY | OVR | MODF | CRC<br>ERR | UDR | CHSIDE | TXE | RXNE |
|      |      |      | r         | r  | r          | r | r   | r   | r   | r    | rc_w0      | r   | r      | r   | r    |

**Биты 15: 13:** Зарезервированы, должны храниться в значении сброса.

**Биты 12:11 FTLVL [1: 0]:** Уровень передачи FIFO

Эти биты устанавливаются и очищаются аппаратными средствами.

**00:** FIFO пустой

**01:** 1/4 FIFO

**10:** 1/2 FIFO

**11:** FIFO полный (считается FULL, когда порог FIFO больше 1/2)

Примечание: Эти биты не используются в режиме I<sup>2</sup>S.

**Биты 10: 9 FRLVL [1: 0]:** уровень приема FIFO

Эти биты устанавливаются и очищаются аппаратными средствами.

**00:** FIFO пустой

**01:** 1/4 FIFO

**10:** 1/2 FIFO

**11:** FIFO полный

Примечание: Эти биты не используются в режиме I<sup>2</sup>S и в режиме только для приема SPI, в то время как расчет CRC разрешен.

**Бит 8 FRE:** ошибка формата кадра

Этот флаг используется для SPI в подчиненном режиме TI и режиме ведомого I<sup>2</sup>S. См. Раздел 28.5.11: Флаги ошибок SPI и Раздел 28.7.9: Флаги ошибок I<sup>2</sup>S.

Этот флаг устанавливается аппаратно и сбрасывается, когда SPIx\_SR считывается программным обеспечением.

**0:** Нет ошибки формата кадра

**1:** Произошла ошибка формата кадра

**Бит 7 BSY:** флаг занятости

**0:** SPI (или I<sup>2</sup>S) не занят

**1:** SPI (или I<sup>2</sup>S) занят связью или буфер Tx не пуст

Этот флаг устанавливается и очищается аппаратными средствами.

Примечание. Флаг BSY должен использоваться с осторожностью: см. Раздел 28.5.10: флаги состояния SPI и Процедура отключения SPI на стр. 767.

**Бит 6 OVR:** флаг переполнения

**0:** переполнение не произошло

**1:** произошло переполнение

Этот флаг устанавливается аппаратно и сбрасывается программной последовательностью. См. Флаги ошибок I<sup>2</sup>S на стр. 799 для последовательности программного обеспечения.

**Бит 5 MODF:** сбой режима

**0:** Не возникла ошибка режима

**1:** произошел сбой режима

Этот флаг устанавливается аппаратно и сбрасывается программной последовательностью. Обратитесь к разделу: Ошибка режима (MODF) на стр. 777 для последовательности программного обеспечения.

Примечание: Этот бит не используется в режиме I<sup>2</sup>S.

**Бит 4: CRCERR:** флаг ошибки CRC

**0:** полученное значение CRC соответствует значению SPIx\_RXCRCR

**1:** Полученное значение CRC не соответствует значению SPIx\_RXCRCR

Этот флаг устанавливается аппаратно и очищается программным обеспечением 0.

Примечание: Этот бит не используется в режиме I<sup>2</sup>S.

**Бит 3 UDR:** флаг опустошения

**0:** Не произошло опустошения

**1:** Опустошение произошло

Этот флаг устанавливается аппаратно и сбрасывается программной последовательностью. См. Флаги ошибок I<sup>2</sup>S на стр. 799 для последовательности программного обеспечения.

Примечание: Этот бит не используется в режиме SPI.

**Бит 2 CHSIDE:** сторона канала

**0:** Канал влево должен быть передан или принят

**1:** Канал вправо должен быть передан или принят

Примечание: Этот бит не используется в режиме SPI. Это не имеет никакого значения в режиме PCM.

**Бит 1 TXE:** пустой буфер передачи

**0:** буфер Tx не пуст

**1:** пустой буфер Tx

**Бит 0 RXNE:** буфер приема не пуст

**0:** пустой буфер Rx

**1:** буфер Rx не пуст

#### 28.9.4 Регистр данных SPI (SPIx\_DR)

Смещение адреса: 0x0C

Значение сброса: 0x0000

| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7  | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| DR[15:0] |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| rw       | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

**Биты 15: 0 DR [15: 0]:** регистр данных

Данные, полученные или подлежащие передаче

Регистр данных служит интерфейсом между Rx и Tx FIFO. Когда считывается регистр данных, RxFIFO получает доступ, а запись в регистр данных обращается к TxFIFO (см. Раздел 28.5.9: Процедуры передачи и приема данных).

Примечание. Данные всегда выравниваются по правому краю. Неиспользуемые биты игнорируются при записи в регистр и считываются как ноль при чтении регистра. Установка порога Rx должна всегда соответствовать используемому в настоящее время доступу.

#### 28.9.5. Регистр многочлена CRC SPI (SPIx\_CRCPR)

Смещение адреса: 0x10

Сбросить значение: 0x0007

| 15            | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7  | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
|---------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| CRCPOLY[15:0] |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| rw            | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

**Биты 15: 0 CRCPOLY [15: 0]:** CRC-полиномиальный регистр

Этот регистр содержит полином для вычисления CRC.

Многочлен CRC (0007h) является значением сброса этого регистра. Другой многочлен можно настроить по необходимости.

Примечание. Значение полинома должно быть нечетным. Четное значение не поддерживается.

#### 28.9.6 Регистр Rx CRC SPI (SPIx\_RXCRCR)

Смещение адреса: 0x14

Значение сброса: 0x0000

| 15          | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------------|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| RxCRC[15:0] |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| r           | r  | r  | r  | r  | r  | r | r | r | r | r | r | r | r | r | r |

**Биты 15: 0 RXCRC [15: 0]:** Регистр CRx Rx

Когда вычисление CRC включено, биты RxCRC [15: 0] содержат вычисленное значение CRC для полученных впоследствии байтов. Этот регистр сбрасывается, когда бит CRCEN в регистре SPIx\_CR1 записывается в 1. CRC вычисляется последовательно с использованием полинома, запрограммированного в регистре SPIx\_CRCPR.

Будут рассмотрены только 8 битов LSB, если формат кадра CRC задан как 8-разрядная длина (бит CRCL в SPIx\_CR1 очищен). Расчет CRC выполняется на основе любого стандарта CRC8. Все 16 бит этого регистра учитываются, когда выбран 16-битный формат кадра CRC (бит CRCL в регистре SPIx\_CR1 установлен). Расчет CRC выполняется на основе любого стандарта CRC16. Примечание: Чтение в этот регистр, когда флаг BSY установлено, может вернуть неправильное значение.

Эти биты не используются в режиме I<sup>2</sup>S.

### 28.9.7 Регистр Tx CRC SPI (SPIx\_TXCRCR)

Смещение адреса: 0x18

Значение сброса: 0x0000

| 15          | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------------|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| TxCRC[15:0] |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| r           | r  | r  | r  | r  | r  | r | r | r | r | r | r | r | r | r | r |

**Биты 15: 0 TxCRC [15: 0]:** регистр Tx CRC

Когда вычисление CRC включено, биты TxCRC [7: 0] содержат вычисленное значение CRC переданных впоследствии байтов. Этот регистр сбрасывается, когда бит CRCEN SPIx\_CR1 записывается в 1. CRC вычисляется последовательно с использованием полинома, запрограммированного в регистре SPIx\_CRCPR.

Будут рассмотрены только 8 битов LSB, если формат кадра CRC задан как 8-разрядная длина (бит CRCL в SPIx\_CR1 очищен). Расчет CRC выполняется на основе любого стандарта CRC8.

Все 16 бит этого регистра учитываются, когда выбран 16-битный формат кадра CRC (бит CRCL в регистре SPIx\_CR1 установлен). Расчет CRC выполняется на основе любого стандарта CRC16.

Примечание: Чтение в этот регистр, когда установлен флаг BSY, может вернуть неправильное значение.

Эти биты не используются в режиме I<sup>2</sup>S.

### 28.9.8 Регистр конфигурации SPIx\_I2S (SPIx\_I2SCFGR)

Смещение адреса: 0x1C

Значение сброса: 0x0000

| 15   | 14   | 13   | 12   | 11     | 10   | 9      | 8   | 7       | 6    | 5      | 4   | 3     | 2      | 1   | 0     |
|------|------|------|------|--------|------|--------|-----|---------|------|--------|-----|-------|--------|-----|-------|
| Res. | Res. | Res. | Res. | I2SMOD | I2SE | I2SCFG |     | PCMSYNC | Res. | I2SSTD |     | CKPOL | DATLEN |     | CHLEN |
|      |      |      |      | r/w    | r/w  | r/w    | r/w | r/w     |      | r/w    | r/w | r/w   | r/w    | r/w | r/w   |

**Биты 15:12** Зарезервировано: принудительное до 0 аппаратными средствами

Бит 11 I2SMOD: выбор режима I<sup>2</sup>S

**0:** выбран режим SPI

**1:** выбран режим I<sup>2</sup>S

Примечание: Этот бит должен быть сконфигурирован, когда SPI или I2S отключены.

**Бит 10 I2SE:** включение I<sup>2</sup>S

**0:** периферийное устройство I<sup>2</sup>S отключено

**1:** периферийное устройство I<sup>2</sup>S включено

Примечание: Этот бит не используется в режиме SPI.

**Биты 9: 8 I2SCFG:** режим конфигурации I<sup>2</sup>S

**00:** Ведомый — передача

**01:** ведомый — прием

**10:** Мастер — передача

**11:** Мастер — прием

Примечание: Эти биты должны быть сконфигурированы, когда I<sup>2</sup>S отключен.

Они не используются в режиме SPI.

**Бит 7 PCMSYNC:** синхронизация кадров ПКМ

**0:** Синхронизация коротких кадров

**1:** Синхронизация длинных кадров

Примечание. Этот бит имеет смысл только в том случае, если I2SSTD = 11 (используется стандарт PCM).

Он не используется в режиме SPI.

**Бит 6** Зарезервировано: принудительное на 0 аппаратным обеспечением

**Биты 5: 4 I2SSTD:** стандартный выбор I<sup>2</sup>S

**00:** стандарт I<sup>2</sup>S Philips.

**01:** Обоснованный стандарт MSB (выравнивание по левому краю)

**10:** Обоснованный стандарт LSB (выравнивание по правому краю)

**11:** стандарт PCM

Для получения дополнительной информации о стандартах I<sup>2</sup>S см. Раздел 28.7.3 на стр. 784

Примечание: для правильной работы эти биты должны быть сконфигурированы, когда I<sup>2</sup>S отключен.

Они не используются в режиме SPI.

**Бит 3 CKPOL:** Полярность неактивного состояния

**0:** неактивное состояние часов I<sup>2</sup>S — низкий уровень

**1:** Неактивное состояние часов I<sup>2</sup>S — высокий уровень

Примечание: для правильной работы этот бит должен быть сконфигурирован, когда I<sup>2</sup>S отключен.

Он не используется в режиме SPI.

Бит CKPOL не влияет на чувствительность к фронту СК, используемую для приема или передачи сигналов SD и WS.

**Биты 2: 1 DATLEN:** длина передаваемой информации

**00:** 16-разрядная длина данных

**01:** 24-разрядная длина данных

**10:** 32-разрядная длина данных

**11:** Не допускается

Примечание: для правильной работы эти биты должны быть сконфигурированы, когда I<sup>2</sup>S отключен.

Они не используются в режиме SPI.

**Бит 0 CHLEN:** длина канала (количество бит на аудиоканал)

**0:** 16-разрядная ширина

**1:** 32-разрядная ширина

Операция записи битов имеет смысл только в том случае, если DATLEN = 00, иначе длина канала фиксирована до 32-битного аппаратным значением независимо от значения, заполненного.

Примечание: для правильной работы этот бит должен быть сконфигурирован, когда I<sup>2</sup>S отключен.

Он не используется в режиме SPI.

### 28.9.9. Регистр предделителя SPIx\_I2S (SPIx\_I2SPR)

Смещение адреса: 0x20

Сбросить значение: 0x0002

| 15   | 14   | 13   | 12   | 11   | 10   | 9     | 8   | 7           | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|------|------|------|------|------|-------|-----|-------------|---|---|---|---|---|---|---|
| Res. | Res. | Res. | Res. | Res. | Res. | MCKOE | ODD | I2SDIV[7:0] |   |   |   |   |   |   |   |
|      |      |      |      |      |      | rw    | rw  | rw          |   |   |   |   |   |   |   |

**Биты 15:10** Зарезервировано: принудительное до 0 аппаратными средствами

**Бит 9 MCKOE:** Разрешение выхода главного тактового сигнала

**0:** Выход главного таймера отключен

**1:** Выход главного таймера включен

Примечание: Этот бит должен быть сконфигурирован, когда I<sup>2</sup>S отключен. Он используется только тогда, когда I<sup>2</sup>S находится в режиме ведущего.

Он не используется в режиме SPI.

**Бит 8 ODD:** Нечетный коэффициент для предделителя

**0:** Действительное значение делителя =  $I2SDIV * 2$

**1:** Действительное значение делителя =  $(I2SDIV * 2) + 1$

См. Раздел 28.7.4 на стр. 791

Примечание: Этот бит должен быть сконфигурирован, когда I<sup>2</sup>S отключен. Он используется только тогда, когда I<sup>2</sup>S находится в режиме ведущего.

Он не используется в режиме SPI.

**Биты 7: 0 I2SDIV [7: 0]:** линейный предделитель I<sup>2</sup>S

I2SDIV [7: 0] = 0 или I2SDIV [7: 0] = 1 являются запрещенными значениями.

См. Раздел 28.7.4 на стр. 791

Примечание: Эти биты должны быть сконфигурированы, когда I<sup>2</sup>S отключен. Они используются только тогда, когда I<sup>2</sup>S находится в главном режиме.

Они не используются в режиме SPI.

### 28.9.10 Карта регистров SPI / I<sup>2</sup>S

Таблица 115 показывает карту регистров SPI / I<sup>2</sup>S и значения сброса.

Таблица 115. Карту регистров и значений сброса регистров SPI

| Offset | Register     | 31  | 30  | 29  | 28  | 27  | 26  | 25  | 24  | 23  | 22  | 21  | 20  | 19  | 18  | 17  | 16  | 15            | 14      | 13         | 12         | 11       | 10     | 9   | 8     | 7        | 6     | 5        | 4    | 3      | 2       | 1       | 0    |  |
|--------|--------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|---------------|---------|------------|------------|----------|--------|-----|-------|----------|-------|----------|------|--------|---------|---------|------|--|
| 0x00   | SPIx_CR1     | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | BIDMODE       | BIDOE   | CRCEN      | CRCNEXT    | CRCL     | RXONLY | SSM | SSI   | LSBFIRST | SPE   | BR [2:0] |      |        | MSTR    | CPOL    | CPHA |  |
|        | Reset value  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     | 0             | 0       | 0          | 0          | 0        | 0      | 0   | 0     | 0        | 0     | 0        | 0    | 0      | 0       | 0       | 0    |  |
| 0x04   | SPIx_CR2     | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res           | LDMA_TX | LDMA_RX    | FRXTH      | DS[3:0]  |        |     | TXEIE | RXNEIE   | ERRIE | FRF      | NSSP | SSOE   | TXDMAEN | RXDMAEN |      |  |
|        | Reset value  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |               | 0       | 0          | 0          | 0        | 1      | 1   | 1     | 0        | 0     | 0        | 0    | 0      | 0       | 0       | 0    |  |
| 0x08   | SPIx_SR      | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res           | Res     | FTLVL[1:0] | FRLVL[1:0] |          |        | FRE | BSY   | OVR      | MODF  | CRCERR   | UDR  | CHSIDE | TXE     | RXNE    |      |  |
|        | Reset value  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |               |         | 0          | 0          | 0        | 0      | 0   | 0     | 0        | 0     | 0        | 0    | 0      | 0       | 1       | 0    |  |
| 0x0C   | SPIx_DR      | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | DR[15:0]      |         |            |            |          |        |     |       |          |       |          |      |        |         |         |      |  |
|        | Reset value  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     | 0             | 0       | 0          | 0          | 0        | 0      | 0   | 0     | 0        | 0     | 0        | 0    | 0      | 0       | 0       | 0    |  |
| 0x10   | SPIx_CRCPR   | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | CRCPOLY[15:0] |         |            |            |          |        |     |       |          |       |          |      |        |         |         |      |  |
|        | Reset value  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     | 0             | 0       | 0          | 0          | 0        | 0      | 0   | 0     | 0        | 0     | 0        | 0    | 0      | 1       | 1       | 1    |  |
| 0x14   | SPIx_RXCR    | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | RxCRC[15:0]   |         |            |            |          |        |     |       |          |       |          |      |        |         |         |      |  |
|        | Reset value  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     | 0             | 0       | 0          | 0          | 0        | 0      | 0   | 0     | 0        | 0     | 0        | 0    | 0      | 0       | 0       | 0    |  |
| 0x18   | SPIx_TXCR    | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | TxCRC[15:0]   |         |            |            |          |        |     |       |          |       |          |      |        |         |         |      |  |
|        | Reset value  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     | 0             | 0       | 0          | 0          | 0        | 0      | 0   | 0     | 0        | 0     | 0        | 0    | 0      | 0       | 0       | 0    |  |
| 0x1C   | SPIx_I2SCFGR | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res           | I2SMOD  | I2SE       | I2SCFG     | PCMSSYNC |        |     | Res   | I2SSTD   | CKPOL |          |      | DATLEN | CHLEN   |         |      |  |
|        | Reset value  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |               | 0       | 0          | 0          | 0        | 0      | 0   |       | 0        | 0     | 0        | 0    | 0      | 0       | 0       | 0    |  |
| 0x20   | SPIx_I2SPR   | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res | Res           | Res     | Res        | MCKOE      | ODD      | I2SDIV |     |       |          |       |          |      |        |         |         |      |  |
|        | Reset value  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |               |         |            | 0          | 0        | 0      | 0   | 0     | 0        | 0     | 0        | 0    | 0      | 0       | 1       | 0    |  |

Обратитесь к разделу 2.2.2 на стр. 46 для адресов границ регистра.

## SPI code example

### A.17.1 SPI master configuration code example

```

/* (1) Master selection, BR: Fpclk/256 (due to C27 on the board, SPI_CLK is
set to the minimum) CPOL and CPHA at zero (rising first edge) */
/* (2) Slave select output enabled, RXNE IT, 8-bit Rx fifo */
/* (3) Enable SPI1 */
SPI1->CR1 = SPI_CR1_MSTR | SPI_CR1_BR; /* (1) */
SPI1->CR2 = SPI_CR2_SS0E | SPI_CR2_RXNEIE | SPI_CR2_FRXTH
| SPI_CR2_DS_2 | SPI_CR2_DS_1 | SPI_CR2_DS_0; /* (2) */

```

```
SPI1->CR1 |= SPI_CR1_SPE; /* (3) */
```

## SPI slave configuration code example

```
/* nSS hard, slave, CPOL and CPHA at zero (rising first edge) */
/* (1) RXNE IT, 8-bit Rx fifo */
/* (2) Enable SPI2 */
SPI2->CR2 = SPI_CR2_RXNEIE | SPI_CR2_FRXTH
| SPI_CR2_DS_2 | SPI_CR2_DS_1 | SPI_CR2_DS_0; /* (1) */
SPI2->CR1 |= SPI_CR1_SPE; /* (2) */
```

## SPI full duplex communication code example

```
if ((SPI1->SR & SPI_SR_TXE) == SPI_SR_TXE) /* Test Tx empty */
{
    /* Will initiate 8-bit transmission if TXE */
    *(uint8_t *)&(SPI1->DR) = SPI1_DATA;
}
```

## SPI interrupt code example

```
if ((SPI1->SR & SPI_SR_RXNE) == SPI_SR_RXNE)
{
    SPI1_Data = (uint8_t)SPI1->DR; /* receive data, clear flag */
    /* Process */
}
```

## SPI master configuration with DMA code example

```
/* (1) Master selection, BR: Fpclk/256 (due to C27 on the board, SPI_CLK is
set to the minimum)
CPOL and CPHA at zero (rising first edge) */
/* (2) TX and RX with DMA,
enable slave select output,
enable RXNE interrupt,
select 8-bit Rx fifo */
/* (3) Enable SPI1 */
SPI1->CR1 = SPI_CR1_MSTR | SPI_CR1_BR; /* (1) */
SPI1->CR2 = SPI_CR2_TXDMAEN | SPI_CR2_RXDMAEN | SPI_CR2_SS0E
| SPI_CR2_RXNEIE | SPI_CR2_FRXTH
| SPI_CR2_DS_2 | SPI_CR2_DS_1 | SPI_CR2_DS_0; /* (2) */
SPI1->CR1 |= SPI_CR1_SPE; /* (3) */
```

## SPI slave configuration with DMA code example

```
/* nSS hard, slave, CPOL and CPHA at zero (rising first edge) */
/* (1) Select TX and RX with DMA, enable RXNE interrupt, select 8-bit Rx fifo */
/* (2) Enable SPI2 */
SPI2->CR2 = SPI_CR2_TXDMAEN | SPI_CR2_RXDMAEN
| SPI_CR2_RXNEIE | SPI_CR2_FRXTH | SPI_CR2_DS_2 | SPI_CR2_DS_1 | SPI_CR2_DS_0; /* (1)
*/
SPI2->CR1 |= SPI_CR1_SPE; /* (2) */
```