MCP2510 – автономный контроллер CAN с интерфейсом SPI

Введение

Характеристики

• Полная реализация протокола CAN версий 2.0A и 2.0B при скорости 1Мбит/с:
- длина сообщения 0 – 8 байтов (длина данных);
- стандартный и расширенный форматы кадров;
- программируемая скорость передачи до 1Мбит/с;
- поддержка кадров удалённого запроса данных;
- два приёмных буфера с разными приоритетами;
- шесть полных приёмных фильтров;
- две полные приёмные маски фильтров;
- три передающих буфера с возможностями назначения приоритетов и отмены передачи сообщения;
- режим петли (*Loopback*) для тестирования.

• Аппаратные характеристики:
- высокоскоростной интерфейс SPI (5 МГц при 4.5 В в индустриальном диапазоне температур);
- поддержка режимов SPI 0,0 и 1,1;
- выход тактовой частоты с программируемым предделителем;
- выход прерывания с возможностью выбора источников прерывания;
- выходы "буфер полон" для каждого приёмного буфера, настраиваемые как вывод прерывания или как цифровой выход основного назначения;
- входы "запрос на передачу" для каждого передающего буфера, настраиваемые как входы для непосредственного управления запросом передачи сообщения или как цифровые входы общего назначения;
- режим низкого потребления Sleep.

• Технология КМОП с низким энергопотреблением:
- работает на напряжениях от 3.0 В до 5.5 В;
- потребление тока в активном состоянии 5 мА;
- потребление тока в режиме пониженного энергопотребления 10 мкА при напряжении 5.5 В.

• 18-выводные корпуса PDIP/SOIC и 20-выводный корпус TSSOP.

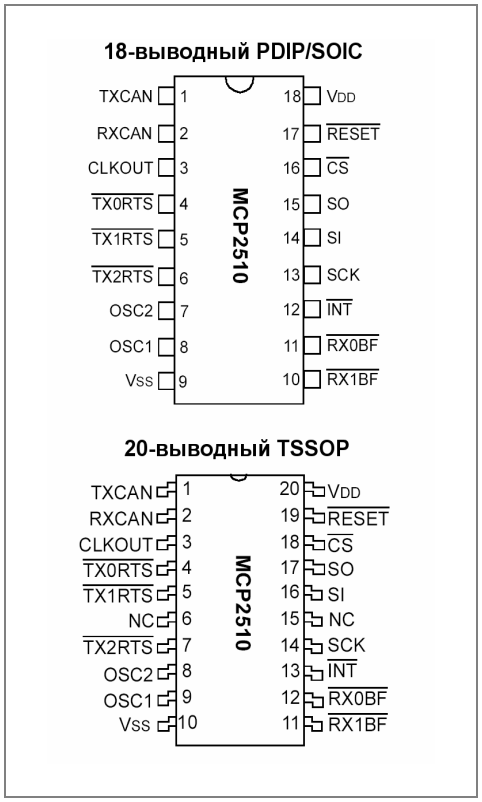
• Поддерживаемые диапазоны температур:
- промышленный (*I - Industrial*): от –40 ºC до +85 ºC.
- расширенный (*E - Extended*): от –40 ºC до +125 ºC.

Описание

MCP2510 – контроллер протокола CAN[1], полностью реализующий спецификации CAN версий 2.0A/B. Он поддерживает версии протокола CAN 1.2, CAN 2.0A, CAN 2.0B пассивный и CAN 2.0B активный и способен передавать и принимать стандартные и расширенные сообщения с возможностью приёмной фильтрации и управления сообщениями. Он включает три передающих буфера и два приёмных буфера. Связь с ЦПУ реализована через интерфейс SPI[2] со скоростью передачи до 5 Мбит/с.

[1] *Controller Area Network* – локальная сеть контроллеров.
[2] *Serial Peripheral Interface* – последовательный интерфейс связи с периферийными устройствами.

Типы корпусов



1 Функциональные возможности устройства

1.1 Краткий обзор

MCP2510 – автономный контроллер протокола CAN, разработанный для приложений, в которых требуется подключение к шине CAN. Простая блок-схема MCP2510 представлена на **Рис. 1-1**.

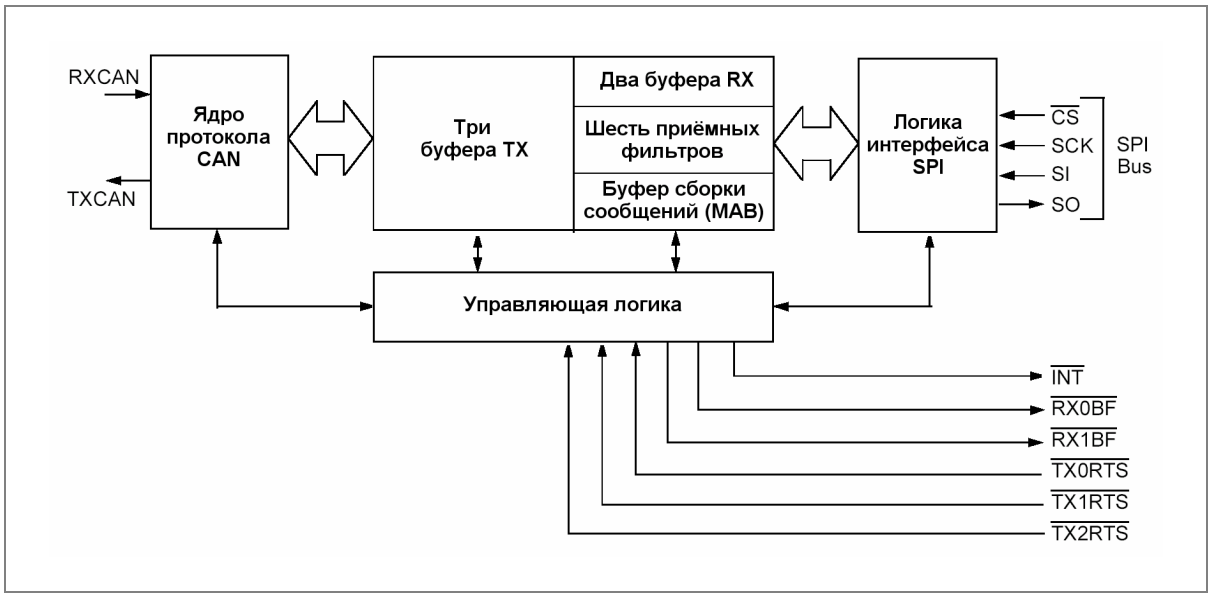


Рис. 1-1. Блок-схема MCP2510

MCP2510 состоит из трёх основных блоков:
1) ядро протокола CAN;

2) логика управления и регистры SRAM, которые используются для настройки контроллера, а также во время его работы;

3) блок протокола SPI.

Типичная реализация системы с применением MCP2510 показана на **Рис. 1-2**.

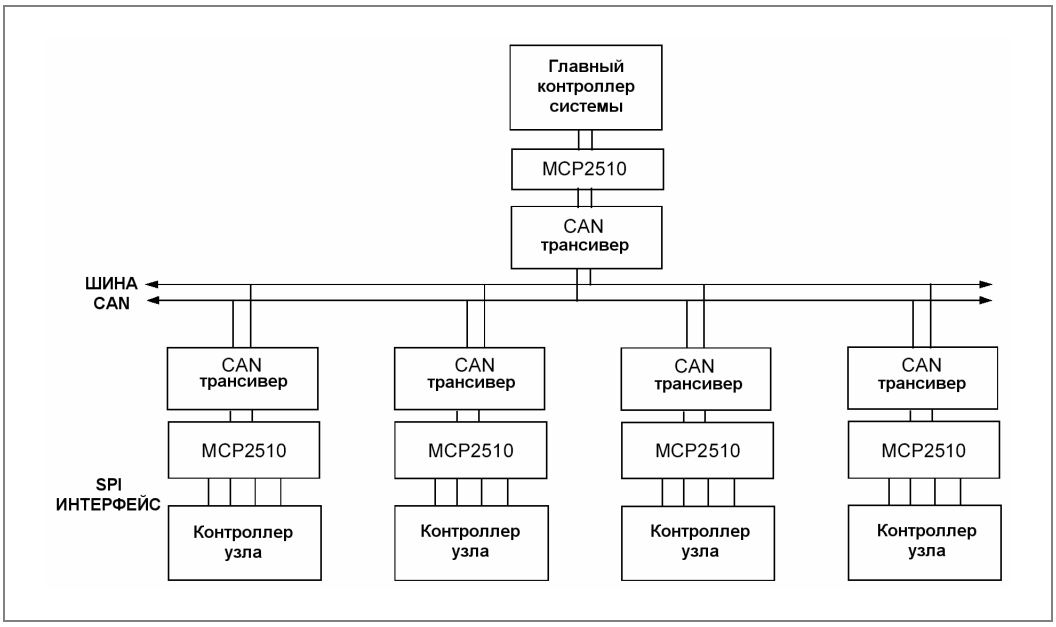


Рис. 1-2. Типичная реализация системы с применением MCP2510

Ядро протокола CAN управляет всеми функциями приёма и передачи сообщения. Для передачи сообщения сначала необходимо загрузить его в передающий буфер и настроить соответствующие управляющие регистры. Передача инициируется либо установкой бита управляющего регистра через интерфейс SPI, либо посредством выводов разрешения передачи. Проверка состояния и наличия ошибок осуществляется чтением соответствующих регистров. Любое сообщение, обнаруженное на шине CAN, проверяется на ошибки, а затем – на соответствие пользовательским фильтрам, в результате чего принимается решение, следует ли помещать это сообщение в один из приёмных буферов.

Взаимодействие MCP2510 с ЦПУ осуществляется через интерфейс SPI с помощью определённого набора команд.

Для обеспечения большей гибкости системы реализовано несколько выводов прерываний. Имеется один многоцелевой вывод прерывания, а также отдельные выводы прерывания для каждого из приёмных буферов, с помощью которых можно узнать, что сообщение принято и загружено в соответствующий буфер. Получить информацию о приёме сообщения можно также с помощью многоцелевого вывода прерывания и посредством чтения регистра состояния через интерфейс SPI.

Кроме того, доступны три вывода для непосредственной инициации передачи сообщения, загруженного в соответствующий буфер. Инициировать передачу сообщения можно также посредством соответствующих управляющих регистров, доступных через интерфейс SPI.

В **Таблице 1-1** представлен полный список всех выводов MCP2510.

Описание выводов MCP2510

Таблица 1-1. Описание выводов MCP2510

Название	DIP/ SOIC № вывода	TSSOP № вывода	Тип I/O/P[1]	Описание
TXCAN	1	1	O	Выход передачи на шину CAN.
RXCAN	2	2	I	Вход приёма с шины CAN.
CLKOUT	3	3	O	Выход синхронизации с программируемым предделителем.

TX0RTS	4	4	I	Запрос передачи для передающего буфера TXB0 или цифровой вход основного назначения. Внутренняя подтяжка к V_{DD} через резистор 100 кОм.
TX1RTS	5	5	I	Запрос передачи для передающего буфера TXB1 или цифровой вход основного назначения. Внутренняя подтяжка к V_{DD} через резистор 100 кОм.
TX2RTS	6	7	I	Запрос передачи для передающего буфера TXB2 или цифровой вход основного назначения. Внутренняя подтяжка к V_{DD} через резистор 100 кОм.
OSC2	7	8	O	Выход генератора.
OSC1	8	9	I	Вход генератора.
VSS	9	10	P	Земля для выводов логики и ввода/вывода.
RX1BF	10	11	O	Вывод прерывания приёмного буфера RXB1 или цифровой выход основного назначения.
RX0BF	11	12	O	Вывод прерывания приёмного буфера RXB0 или цифровой выход основного назначения.
INT	12	13	O	Выход прерывания.
SCK	13	14	I	Вход синхронизации для интерфейса SPI.
SI	14	16	I	Вход данных для интерфейса SPI.
SO	15	17	O	Выход данных для интерфейса SPI.
CS	16	18	I	Вход выбора ведомого для интерфейса SPI.
RESET	17	19	I	Вход сброса устройства, активный уровень – низкий.
VDD	18	20	P	Подача питания для выводов логики и ввода/вывода.
NC	—	6,15	—	Не подключен (не используется).

Примечание 1: обозначение типов выводов: I = вход, O = выход, P = питание.

1.2 Приёмные/передающие буферы

MCP2510 имеет три передающих и два приёмных буфера, две приёмные маски (по одной для каждого приёмного буфера) и шесть полных приемных фильтра. На **Рис. 1-3** показана блок-схема этих буферов и их связи с ядром протокола.

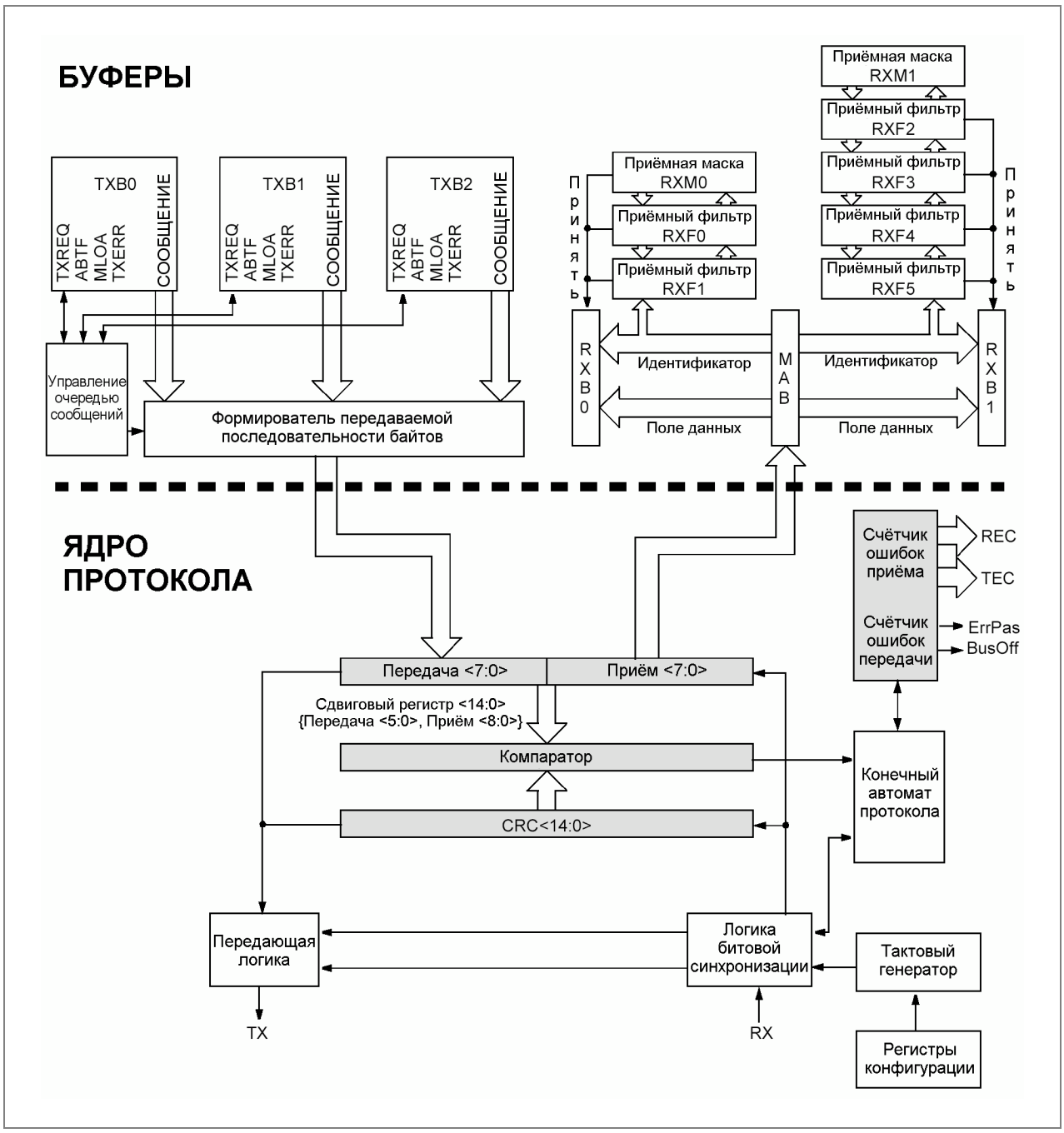


Рис. 1-3. Блок схема буферов и ядра протокола CAN

1.3 Ядро протокола CAN

Ядро протокола CAN состоит из нескольких функциональных блоков, показанных на **Рис. 1-4**. Эти блоки и их функции описаны ниже.

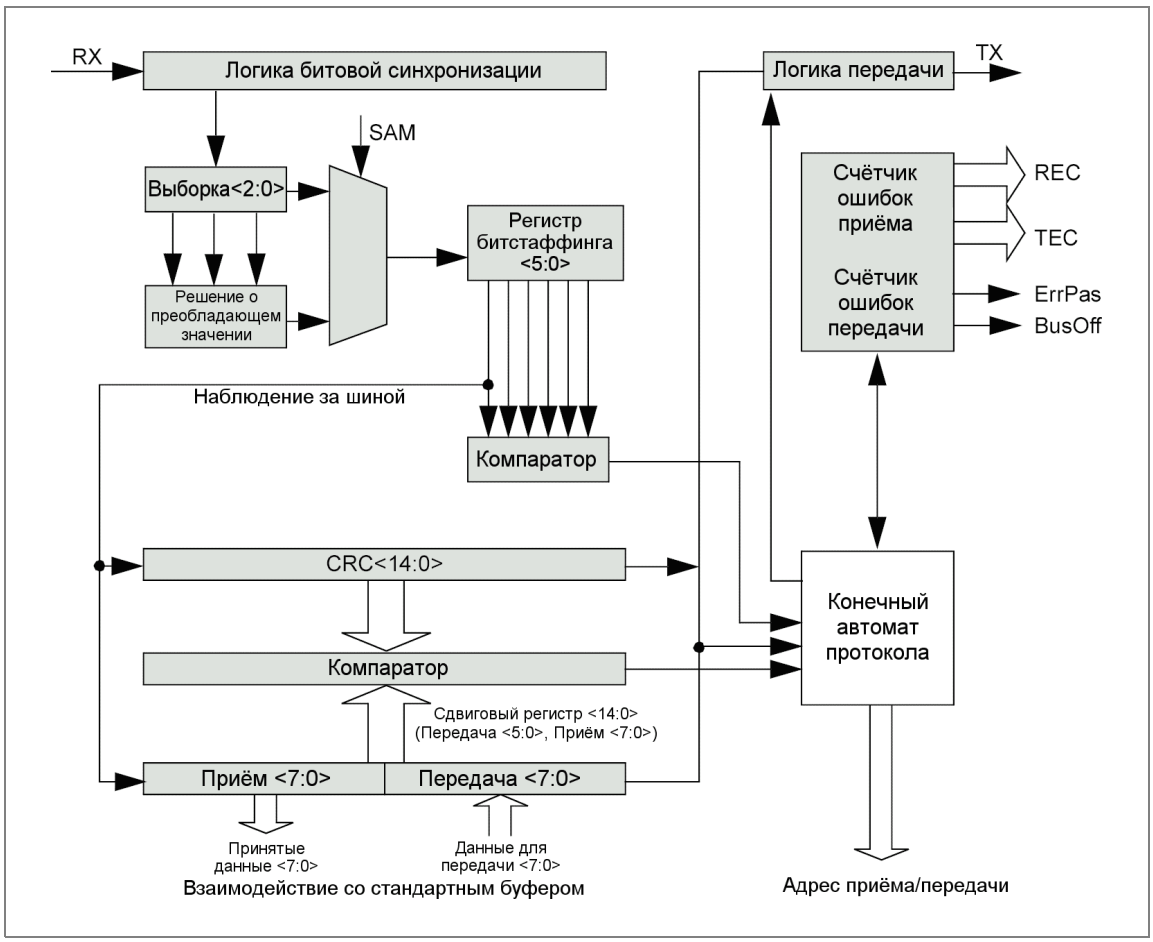


Рис. 1-4. Блок-схема ядра протокола

1.4 Конечный автомат протокола

Сердцем протокола является конечный автомат протокола (*Finite State Machine - FSM*). Этот конечный автомат последовательно проходит через сообщения бит за битом, меняя своё состояние в зависимости от передаваемых или принимаемых полей различных типов кадров. FSM – это контроллер последовательности, управляющий последовательным потоком данных между сдвиговыми регистрами TX/RX, регистром CRC и линией шины. Кроме того, FSM управляет логикой обработки ошибок (*Error Management Logic - EML*) и параллельными потоками данных между сдвиговыми регистрами RX/TX и буферами. FSM обеспечивает выполнение процессов приёма, арбитража, передачи и сигнализации об ошибках согласно протоколу CAN. Автоматическая повторная передача сообщения на линию шины тоже управляется FSM.

1.5 CRC

Регистр CRC[3] генерирует циклической избыточной код (код CRC), который передаётся после поля данных, либо управляющего поля (в сообщениях с нулевым количеством байтов данных), а также используется для проверки поля CRC поступающего сообщения.

1.6 Логика управления ошибками

Логика управления отвечает за локализацию неисправного устройства CAN. Она представляет собой два счётчика: счётчик ошибок приёма (*Receive Error Counter - REC*) и счётчик ошибок передачи (*Transmit Error Counter - TEC*), инкрементируемые и декрементируемые по команде от обработчика

[3] Cyclic Redundancy Check – контроль циклическим избыточным кодом.

потока битов (*Bit Stream Processor*). Согласно значениям счётчиков ошибок, контроллер CAN переводится в состояние активной ошибки, пассивной ошибки или отключается от шины.

1.7 Логика битовой синхронизации

Логика битовой синхронизации (*Bit Timing Logic - BTL*) наблюдает за входной линией шины и управляет зависимой от шины битовой синхронизацией согласно протоколу CAN. BTL синхронизируется по переходу шины из рецессивного состояния в доминантное в начале кадра (*Start of Frame*) (жёсткая синхронизация), а также на любом последующем переходе линии шины из рецессивного состояния в доминантное, если контроллер CAN не сам передаёт доминантный бит (синхронизация с восстановлением тактовых интервалов). BTL также позволяет задавать длины временных сегментов, чтобы компенсировать время задержки распространения и фазовые сдвиги и определить позицию точки выборки внутри битового интервала. Параметры BTL зависят от скорости передачи данных и времён внешних физических задержек.