MCP2510 — контроллер CAN-интерфейса фирмы MICROCHIP

Промышленный стандарт передачи сообщений CAN 2.0B

Промышленный стандарт передачи сообщений CAN 2.0B нашел наиболее широкое применение в автомобильной промышленности в качестве стандарта передачи информации между электронными узлами автомобиля. Стандарт позволяет соединять устройства различных производителей без необходимости их перенастройки при добавлении или исключении устройства в сети.

Основное отличие от существовавших стандартов заключается в том, что передаваемый кадр сообщения не содержит адрес приемника устройства — назначения, а содержит идентификатор данных пакета. Один и тот же пакет может быть одновременно прочитан и использован многими устройствами. Пример — данные о напряжении аккумуляторной батареи, поступающие в CAN-шину от датчика-измерителя, могут быть прочитаны и использованы всеми устройствами, которым нужна эта информация. Приемник только настраивает внутренний фильтр сообщений на соответствующий идентификатор.

Алгоритм работы CAN-интерфейса сертифицирован в виде стандарта 2.0B, что позволяет изготавливать CAN-контроллеры в виде отдельных микросхем для реализации всех функций интерфейса на аппаратном уровне. Такой контроллер полностью разгружает основной микропроцессор от выполнения рутинных функций и позволяет обмениваться целыми информационными пакетами с другими узлами сети. Основные ресурсы микропроцессора в этом случае могут быть направлены на обработку поступающей информации. Правильная настройка фильтров и масок CAN-контроллера отфильтрует все не нужные кадры внутри **MCP2510** без использования ресурсов CPU.

Можно отметить растущую популярность CAN-интерфейса в системах управления. В недалеком времени системы, построенные на RS485 интерфейсе, останутся только в нижнем ценовом классе устройств передачи данных, так как они реализуют практически все функции передачи пакетов программным способом. Для развивающихся систем передачи данных такое построение может оказаться узким местом и поэтому оправданно уже сейчас закладывать перспективные решения на основе **MCP2510**.

Уровни протокола передачи данных

- Уровень прикладной программы — получает отфильтрованные сообщения и посылает запросы на получение данных с нужным идентификатором. Интерпретирует полученные данные и использует их для реализации алгоритма пользователя.
- Уровень сообщения — отвечает за фильтрацию сообщений и определение их статуса.
- Транспортный уровень — составляет ядро протокола. Определяет способ передачи принятого сообщения на уровень сообщений и приема сообщения для передачи по физическому каналу. Транспортный уровень отвечает за битовую синхронизацию, разбивку сообщения по кадрам, доступ к каналу, подтверждение сообщений, распознавание ошибок.
- Физический уровень — определяет способ передачи битов. В спецификации CAN 2.0B физическая реализация канала оставлена на усмотрение разработчика для оптимизации конкретной системы.

Информация по CAN-шине посылается на фиксированной частоте в фиксированном формате. Шина имеет два состояния: «dominant» (доминирующее) и «recessive» (изменяемое). Если шина свободна, любой узел может начать передачу нового сообщения. Идентификатор определяет приоритет сообщения при доступе к шине. Если два или более узла одновременно начинают передачу, то конфликт доступа к шине разрешается чтением состояния шины при передаче битов идентификатора и сравнением передаваемого уровня с наблюдаемым на шине. Если уровни одинаковы, то происходит дальнейшая передача. Если передается «recessive», а на шине «dominant», то есть сработал принцип монтажного «или», то узел завершает передачу и теряет доступ к шине. Механизм доступа гарантирует корректную передачу сообщения с наивысшим приоритетом без потери времени и дополнительных задержек. Например, если одновременно начинается передача кадра запроса данных (remote frame) и кадра данных (data frame) с одним и тем же идентификатором, то реально на шине останется кадр данных в связи с тем, что бит RTR (Remote Transmission Request bit) в кадре запроса имеет значение «recessive», а в кадре данных «dominant». Понятно, что по принципу монтажного «или» должен остаться «dominant» бит.

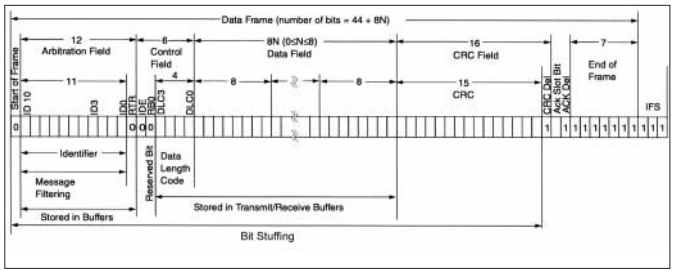
Тот факт, что в узлах нет информации о конфигурации сети (такой, как сетевой адрес), дает дополнительные преимущества при построении системы:

- Любое количество узлов может принимать одно и то же сообщение одновременно.
- Узлы можно добавлять и исключать без изменения программного и аппаратного обеспечения узлов любого уровня.
- Система фильтрации использует идентификатор передаваемого сообщения для принятия решения о необходимости передавать полученную информацию верхнему уровню.

Формат кадра данных

MCP2510 поддерживает все кадры спецификации CAN2.0B. Рассмотрим формат кадра данных.

1. Начало кадра **Start of frame** (1 бит):
   - Кадр начинается старт-битом (**SOF — Start of frame**).
2. Поле управления доступом **Arbitration Field** (12 бит):
   - Одиннадцать бит идентификатора сообщения (ID10-ID0).
   - Бит **RTR** (**Remote Transmission Request**) — определяет сообщение как кадр данных (RTR=0 «dominant») или кадр запроса данных (RTR=1 «recessive»).
3. Поле управления **Control Field** (6 бит):
   - Идентификатор расширенного кадра (IDE).
   - Резервный бит (RBO).
   - Четыре бита (DCL3...DCL0) длины поля данных в байтах (N).
4. Поле данных **Data Field** (8*N бит):
   - N=0,...,8
5. Поле контрольной суммы **CRC Field** (16 бит):
   - 15 бит контрольной суммы.
   - Бит разделителя (**CRC Del**) по определению «recessive».
6. Поле подтверждения (2 бита):
   - **ACKSlot**. Во время передачи узел формирует «recessive» бит в поле ACKSlot.



Стандартный кадр данных

Любой узел, принявший кадр без ошибок, подтверждает корректность приема кадра установкой значения «dominant». Бит устанавливается при корректном приеме кадра вне зависимости от того, удовлетворяет он условиям фильтров или нет.

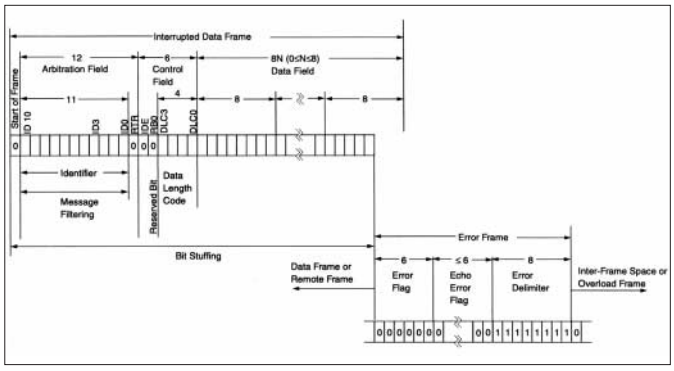
- **ACKDel**. Разделительный бит по определению сохраняет значение «recessive».

7. Поле окончания кадра **End of Frame** (7 бит):
   - Состоит из семи разделительных «recessive» битов.

Кадр ошибок (ERROR FRAME)

Кадр ошибок генерируется любым узлом, обнаружившим ошибку. Он состоит из двух частей. Первая часть получается как объединение флагов ошибок от различных узлов. Вторая часть — биты разделителей. Тип поля кадра ошибок зависит от статуса узла, который распознал ошибку.

Если узел с признаком error-active определяет ошибку на шине, тогда узел прерывает передачу текущего сообщения генерацией активного флага ошибки, состоящего из шести последовательных «dominant» битов. Такая последовательность нарушает правило набивки битов. Все станции распознают ошибку и генерируют флаг ошибки (эхо). Таким образом, флаг ошибки будет состоять из 6...12 бит, генерируемых одним или несколькими узлами. Кадр заканчивается полем разделителя из 8 «recessive» бит. После завер-



Кадр ошибок

шения кадра ошибок шина переходит в нормальный режим, и узел повторяет попытку послать прерванное сообщение.

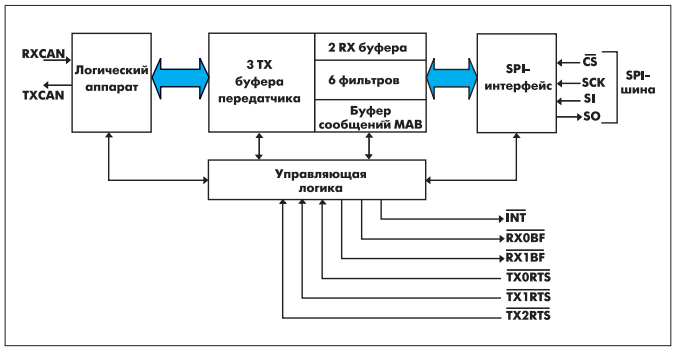
Если узел с признаком error-passive определяет ошибку на шине, то он формирует флаг ошибки из шести «recessive» бит и разделитель из восьми бит той же полярности.

Несмотря на то что ошибка обнаружена передающим узлом, передача пассивного кадра ошибок не будет действовать на другие узлы сети. После передачи кадра ошибок узел должен ждать шесть последовательных «recessive» бит перед началом попытки подключения к шине.

Структура CAN-контроллера MCP2510

CAN-контроллер **MCP2510** состоит из трех основных модулей:

- логического блока, реализующего протокол CAN2.0B;
- буферных регистров и управляющей логики для конфигурирования прибора и настройки режимов его работы;



Структурная схема устройства MCP2510

- SPI-интерфейса для связи с микропроцессором для чтения или записи всех регистров CAN-контроллера.

Сердцем логики работы протокола нижнего уровня является логический аппарат с конечным числом состояний, изменяющий свои состояния под воздействием входного и выходного потока битов. Логический аппарат управляет потоком данных между сдвигающими регистрами приемника, передатчика, регистром подсчета контрольной суммы (CRC) и линиями CAN-шины, управляет автоматической ретрансляцией сообщений шины и т. д.

Передача сообщения

MCP2510 имеет три буфера передатчика. Каждый буфер занимает 14 байт ОЗУ, которые размещены в общем адресном пространстве контроллера.

Первый байт буфера «0» (регистр TXB0CTRL, адрес 30h) используется для управления работой передатчика и контроля результата выполнения передачи сообщения.

U-0	R-0	R-0	R-0	R/W-0	U-0	R/W-0	R/W-0
—	ABTF	MLOA	TXERR	TXREQ	—	TXP1	TXP0
bit 7							bit 0

- Биты TXP0, TXP1 — определяют приоритет сообщения.
- Бит TXREQ — формирует требование на отсылку сообщения.
- Бит TXERR — флаг ошибки при передаче сообщения.
- Бит MLOA — флаг результата доступа к шине.
- Бит ABTF — флаг успешного завершения передачи.

Пять байт (адреса 31h...35h) используются для хранения стандартного или расширенного идентификатора и арбитражных битов; восемь байт (адреса 36h...3Dh) содержат передаваемые данные сообщения.

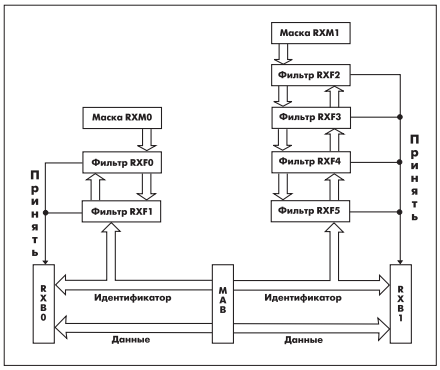
Перед посылкой сообщения микропроцессор инициализирует бит разрешения прерывания.

Подготовка передачи сообщения начинается с загрузки идентификатора и данных соответствующего буфера по адресам 31h...3Dh. Передача начинается после установки бита TXREQ в регистре управления TXRTSCTRL (адрес 0Dh). Бит инициирования передачи устанавливается аппаратно при подаче управляющего сигнала на вывод 4 или передачей команды RTS через SPI-интерфейс. Команда RTS имеет формат 1000 T2 T1 T0, где последние три бита показывают, какой из трех буферов необходимо инициировать. Одной командой можно начать процесс передачи всех трех сообщений.

Прием сообщения

Принимаемое сообщение из CAN-шины поступает в специальный буфер сообщений Message Assembly Buffer (MAB). Полученное и сформированное в MAB сообщение перезаписывается в один из буферов RXB0 или RXB1, если оно удовлетворяет критерию, установленному в регистрах фильтра. Микропроцессор может работать с одним буфером, в то время как второй буфер ис-

пользуется для приема нового сообщения или хранения предыдущего. Когда сообщение передается в буфер, то устанавливается соответствующий бит флага (CANINTF.RxnIF). После завершения обработки данных из буфера, микропроцессор сбрасывает бит CANINTF.RxnIF и готовит буфер к приему нового сообщения. Этот бит обеспечивает защиту от перезаписи буфера новым значением до окончания процедуры обработки предыдущего. Если бит разрешения прерывания CANINTE.RxnIE установлен, то на выходе INT формируется сигнал запроса прерывания.



Структурная схема буферов приемника

Буфер приема высокого приоритета RXB0 имеет два фильтра; буфер низкого приоритета RXB1 имеет четыре фильтра. Меньшее количество фильтров накладывает больше ограничений на RXBO и подразумевает более высокий приоритет этого буфера. Кроме того, регистром RXB0CTRL можно настроить алгоритм работы таким образом, что если RXB0 содержит корректное сообщение и принимает другое корректное сообщение, то новое сообщение будет записано в RXB1 вне зависимости от критериев приема RXB1 и без установки бита ошибки переполнения.

Кроме системы фильтров для каждого буфера существует специальная маска. Маска используется для определения, какие биты в идентификаторе сообщения будут анализироваться фильтрами.

Таблица истинности показывает, как каждый бит идентификатора сравнивается с маской и фильтрами для определения необходимости загрузки сообщения в буфер приемника. Если все биты маски установить в «0», то все сообщения будут загружаться в буфер приемника.

Любое сообщение, обнаруженное на CAN-шине, проверяется на наличие ошибки и на соответствие определенных пользователем фильтров. Если сообщение соответствует установленному фильтру, то оно попадает в один из двух буферов приема. Входной буфер MAB функционирует как третий буфер приемника.

Если фильтр совпадает с идентификатором сообщения в соответствии с таблицей истинности и сообщение загружается в буфер приемника RXBn, то номер фильтра, который выбрал это сообщение, загружается в RXBn CTRL регистр (биты FILHIT).

Если принятому сообщению соответствует более одного фильтра, то загружается младший номер (фильтры с меньшим номером имеют более высокий приоритет). Таким образом, сообщение сравнивается с фильтрами по возрастанию, и принимается первое совпадение. Отметим, что все регистры маски и фильтра можно модифицировать только тогда, когда **MCP2510** находится в режиме инициализации.

Таблица истинности

Бит маски	Бит фильтра	Бит идентификации сообщения	Принять/отвергнуть
0	X	X	Принять
1	0	0	Принять
1	0	1	Отвергнуть
1	1	0	Отвергнуть
1	1	1	Принять

Настройка аппаратных средств

Микросхема **MCP2510** имеет несколько дополнительных выводов для придания гибкости использования контроллера в системе.

Выход — 12 (INT) — прерывание общего назначения. Предназначен для информирования микроконтроллера о возникновении асинхронных событий — приема, передачи и т. д.

Три входа — 4,5,6 (TXnRTS). Позволяют инициировать процедуру немедленной передачи сообщения, загруженного в один из трех буферов передачи. Использование этих выводов является не обязательным, так как инициировать передачу сообщения можно путем использования управляющих регистров контроллера через SPI-интерфейс. В этом случае выводы можно настроить на ввод произвольных логических сигналов. Настройка осуществляется через регистр TXRTSCTRL (адрес 0Dh). Младшие три бита (BnRTSM) установки режима работы вывода TXnRTS задают режим работы вывода.

Если бит установки режима вывода BnRTSM = 0, то вывод TXnRTS настроен как логический вход, причем состояние этого входа можно прочитать через тот же регистр управления TXRTSCTRL (адрес 0Dh) в битах (BnRTS) соответственно.

Если бит установки режима вывода BnRTSM = 1, то вывод TXnRTS настроен как вход управляющего сигнала «начать передачу данных из буфера».

Два выхода, 11 (RX0BF) и 10 (RX1BF) запроса прерывания от приемника, индицируют принятие корректного сообщения в первый или во второй буфер приема соответственно. Конфигурирование выводов осуществляется через регистр BFPCTRL (0Ch) установкой битов режима BnBFM. Бит разрешения BnBFE = 0 переводит выход м/с в высокоимпедансное состояние.

Синхронизация

Все узлы системы на CAN-шине должны иметь одну и ту же номинальную скорость передачи. Так как тактовая частота и время передачи кадра может изменяться от узла к узлу, то приемник должен иметь узел фазовой автоподстройки частоты (ФАПЧ) по фронтам сигнала передатчика.

CAN-протокол использует кодирование «без возвращения к нулю», которое не содержит информации о тактовой частоте в каждом бите. Для получения информации о тактовой частоте используют специальную процедуру вставки дополнительного инверсного бита в передаваемую последовательность (bit stuffing). В результате можно быть уверенным, что фронт сигнала придет на систему ФАПЧ по крайней мере один раз за шесть битовых периодов. Частота приемника подстраивается и синхронизируется с частотой передатчика с использованием цифровой ФАПЧ.

Номинальной скоростью передачи считается количество бит, переданных за секунду.

При номинальной скорости передачи 1 Мбит/с битовый временной интервал (Tbit) составляет 1мкс и может быть мысленно поделен на следующие временные сегменты:

- **Sync** (сегмент синхронизации) — это часть временного интервала, отведенный для синхронизации всех узлов сети.
- **Prop Segment** (пауза) — сегмент времени сразу после сегмента синхронизации. Предназначен для компенсации задержек в линии передачи сети и внутренних задержек внутри узла. Значение рассчитывается исходя из времени распространения сигнала от передатчика к приемнику и обратно и задержек входного компаратора и выходного драйвера передатчика. Длина сегмента паузы устанавливается в битах PRSEG2: PRESEG0 регистра CNF2 и может быть выбрана от одного до восьми квантов Tq.
- **Phase Segment 1 и 2** (фазовые сегменты 1 и 2) — используются внутри Tbit для оптимизации момента считывания значения на шине. Значение фазового сегмента 1 или 2 варьируется от 2Tq до 8Tq.
- **Sample Point** (момент выборки). По определению Tbit состоит из целого числа

Настройка выводов микросхемы 4, 5, 6

№ вывода/Имя	Назначение	Режим вывода	Чтение состояния вывода
Pin 4/(TX0RTS)	Логический вход Вход «начать передачу»	B0RTSM=0 B0RTSM=1	B0RTS = состоянию вывода 4 (не имеет значения)
Pin 5/(TX1RTS)	Логический вход Вход «начать передачу»	B1RTSM=0 B1RTSM=1	B1RTS = состоянию вывода 5 (не имеет значения)
Pin 6/(TX2RTS)	Логический вход Вход «начать передачу»	B2RTSM=0 B2RTSM=1	B2RTS = состоянию вывода 6 (не имеет значения)

Биты BnRTSM (режим) и BnRTS (состояние) находятся в регистре **TXRTSCTRL (0Dh)**

Настройка выводов микросхемы 11, 10

№ вывода/Имя	Назначение	Режим вывода	Вывод значения в порт I/O
Pin 11/(RX0BF)	Логический выход Выход «начало приема»	B0BFM=0 B0BFM=1	Pin 11=B0BFS, если B0BFE=1 Pin 11=___сообщение принято
Pin 10/(RX1BF)	Логический выход Выход «начало приема»	B1BFM=0 B1BFM=1	Pin 10=B1BFS, если B1BFE=1 Pin 10=___сообщение принято

Биты BnBFM (режим), BnBFE (разрешение) находятся в регистре **BFPCTRL (0Ch)**

квантов (Tq). Число квантов в Tbit программируется в диапазоне от 8Tq до 25Tq. В свою очередь временной квант Tq формируется специальным делителем из частоты генератора тактовых импульсов (Fosc).

Tq = 2x(Band Rate+1)x(1/Fosc) — где скорость передачи (Band Rate) задается в регистре CNF.BRP <S:0>

Пример

Fosc = 16 МГц, CNF.BRP <S:0> = 00h, Tbit = 8Tq,
тогда Tq = 125 нс
Скорость передачи = 1М бит/с.

Отметим, что после выбора временного кванта Tq необходимо позаботиться о выборе кварцевых резонаторов для контроллеров всех узлов сети с номинальными значениями Fosc как 1/Tq , умноженное на целое число. Обычно все кварцевые резонаторы контроллеров выбирают одинаковыми, но при согласовании разнородных контроллеров и устройств на одной шине на эту проблему необходимо обратить внимание.

Синхронизация — это алгоритм, по которому функционирует ФАПЧ. Когда приходит фронт сигнала, внутренняя логика сравнивает положение фронта с ожидаемым. Предполагается, что фронт входного сигнала уложится в один квант Tq. Соответствующая схема установит значение фазового сегмента 1 и фазового сегмента 2 в соответствии с одним из двух механизмов синхронизации:

1. Жесткая аппаратная синхронизация. Выполняется в случае прихода фронта сигнала начала сообщения (SOF) в состоянии шины «BUSIDLE». При выполнении аппаратной синхронизации состояние внутренних таймеров битовой синхронизации обнуляется, и таймеры запускаются заново.
2. Подстройка значения битового периода Tbit по величине фазовой ошибки. Результатом подстройки может быть увеличение фазового сегмента 1 или укорачивание фазового сегмента 2. Значение увеличения или уменьшения соответствующего сегмента ограничивается так называемым значением SJW (ширины скачка синхронизации). Это значение будет добавлено к сегменту 1 или вычтено из сегмента 2. Фазовая ошибка получается путем позиционирования пришедшего фронта сигнала по отношению к сегменту синхронизации. Ошибка вычисляется в единицах Tq следующим образом:

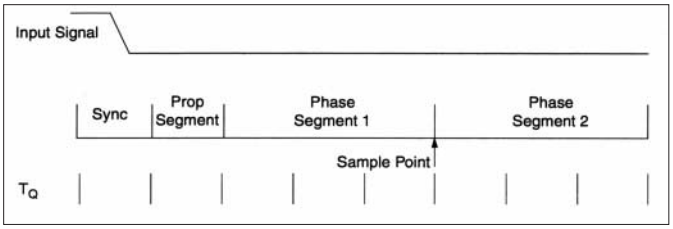
- e = 0, если фронт попал на сегмент синхронизации;
- e > 0, если фронт лежит перед моментом выборки;
- e < 0, если фронт лежит после момента выборки предыдущего бита.

Если значение фазовой ошибки меньше или равно SJW, то эффект подстройки будет тем же самым, что и при жесткой аппаратной синхронизации.

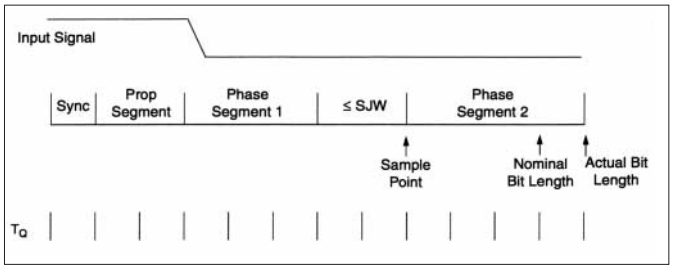
Если значение фазовой ошибки положительно и больше SJW, тогда сегмент 1 увеличивается на SJW, а если отрицательно, то сегмент 2 уменьшается на SJW .

Основные данные MCP2510

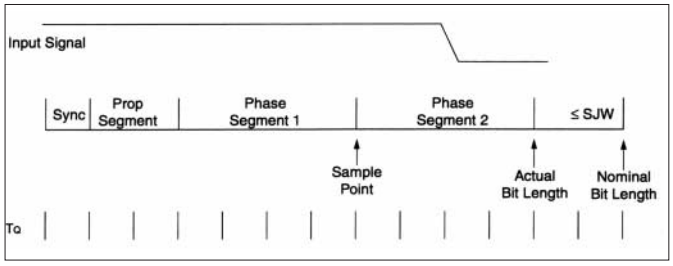
- Полностью поддерживает стандарт CAN2.0B 1 Мбит/с.
- 0–8 байт данных.
- Стандартный и расширенный кадр данных.
- Программируемая скорость передачи до 1 Мбит/с.
- Два буфера приема.
- Шесть полноформатных фильтров приема.
- Две полноформатные маски.
- Три буфера передатчика с системой приоритетов.
- Режим внутреннего тестирования с входа на выход.
- SPI-интерфейс 5 MHz.
- Режим малого потребления в sleep режиме 10 мкА.
- Корпус 18 pin PDIP/SOIC 20pin TSSOP.
- Температурное исполнение -40 °C...+85 °C (I) -40 °C...+125°C (E).



Временной интервал Tbit



Увеличение Tbit



Уменьшение Tbit