Справочник по среднему семейству микроконтроллеров PICmicro™

Перевод основывается на технической документации DS33023A компании Microchip Technology Incorporated, USA.

тел. (095) 737-7545
www.microchip.ru

PICmicro™ Mid-Range MCU Family Reference Manual

Оглавление

Оглавление (продолжение)

Оглавление (продолжение)

Оглавление (продолжение)

Оглавление (продолжение)

Оглавление (продолжение)

Оглавление (продолжение)

Оглавление (продолжение)

Оглавление (продолжение)

Оглавление (продолжение)

Оглавление (продолжение)

Раздел 1. Общие сведения

Содержание

1.1 Введение

Microchip is the Embedded Control Solutions Company®. Компания Microchip Technology Inc. специализируется на выпуске электронных компонентов для построения систем контроля и управления. Основные виды выпускаемой продукции:

- 8 - разрядные универсальные микроконтроллеры (PICmicro™ MCU);
- Специализированные микросхемы энергонезависимой памяти;
- Устройства ограничения доступа (KeeLoq);
- Программное обеспечение и инструментальные средства проектирования.

Полный список выпускаемых изделий, доступных к заказу, смотрите в документе "Product Line Card". Этот документ Вы можете получить в региональном представительстве компании Microchip или загрузить его с Web узлов технической поддержки **www.microchip.com** и **www.microchip.ru**.

Ранее в традиционных 8-разрядных микроконтроллерах требовалось использовать ROM технологию памяти. Компания Microchip была лидером в изменении этого установившегося правила, доказав, что OTP микросхемы имеют более низкую стоимость и больший срок эксплуатации по сравнению с ROM версиями.

Компания Microchip применяет EPROM технологию для организации памяти программ в микроконтроллерах PICmicro MCU. Microchip минимизировал разницу в стоимости между микросхемами выполненными по EPROM и ROM технологии, что естественно положительно сказывается на доходах наших заказчиков. В прайс-листах других производителей были замечены существенные различия в стоимости микросхем, выполненных по технологии EPROM и ROM.

Завоевание доли рынка 8-разрядных MCU фирма Microchip завершила разработкой микроконтроллеров PICmicro, способных удовлетворить требования разработчиков для большинства приложений. Архитектура микроконтроллеров PICmicro является одной из трех наиболее распространенных архитектур, доступных на сегодняшний день на мировом рынке электронных компонентов. Широкому распространению использования PICmicro способствовало предвидение компанией Microchip выгоды от применения не дорогостоящего решения OTP. Можно отметить несколько основных преимуществ применения микроконтроллеров PICmicro:

- Малое время проектирования устройства;
- Возможность изменение кода программы на этапе выпуска изделий;
- Низкая стоимость изменение программы (не требуется изменять маску);
- Простая возможность последовательно присваивать номера изделиям;
- Возможность сохранения калибровочной информации без дополнительных аппаратных решений;
- Меньший риск разработок, т.к. одна и та же микросхема может использоваться при проектировании и в готовом устройстве.

8-разрядные микроконтроллеры PICmicro имеют наилучшее соотношение цена-качество, позволяя их использовать в традиционных 8-разрядных приложениях, в некоторых 4-разрядных приложениях (базовое семейство), заменять специализированные логические элементы, DSP приложения (старшее семейство). Эти особенности, совместно с ценовой эффективностью, делают микроконтроллеры PICmicro привлекательными для большинства приложений.

1.2 Цель документа

Все микроконтроллеры PICmicro разделены на три группы по разрядности команд:

1. Базовое семейство: 12-разрядные команды.
2. Среднее семейство: 14-разрядные команды.
3. Старшее семейство: 16-разрядные команды.

Этот документ ориентирован на микроконтроллеры среднего семейства, иногда обозначаемые как PIC16CXXX. В этом документе объясняется архитектура микроконтроллеров семейства PIC16CXXX и работа периферийных модулей, но не затрагиваются детали работы каждого микроконтроллера. Поэтому он полностью не заменяет технической документации на конкретный микроконтроллер, а лишь дополняет ее. Другими словами, в этом документе даны общие принципы архитектуры и работы микроконтроллеров, а в технической документации Вы найдете точное описание структуры памяти, работы периферийных модулей и электрических параметров.

Во всех разделах этого документа даны примеры инициализации. Эти примеры иногда должны быть изменены в соответствии с требованиями конкретного микроконтроллера, хотя они могут быть правильными для большинства других микроконтроллеров. Некоторые изменения могут потребоваться в связи с отличиями размещения управляющих регистров в памяти данных.

Примечание. Небольшая часть микроконтроллеров среднего семейства (младшая часть) имеют незначительные отличия по сравнению с большинством описываемых микросхем. Эти отличия описаны во всех разделах этого документа. Для получения полной информации по конкретному микроконтроллеру обратитесь к технической документации на соответствующий микроконтроллер.

1.3 Структура микроконтроллеров

Каждая часть микроконтроллера может быть отнесена к одной из трех групп:

1. Ядро микроконтроллера;
2. Периферийные модули;
3. Специальные особенности микроконтроллеров.

1.3.1 Ядро микроконтроллера

Ядро относится к основным особенностям, оно заставляет микроконтроллер работать. В состав этой группы входит:

1. Тактовый генератор	Пересмотренный документ DS31002A
2. Логика сброса	Пересмотренный документ DS31003A
3. Центральный процессор (CPU)	Пересмотренный документ DS31005A
4. Арифметико-логическое устройство (АЛУ)	Пересмотренный документ DS31005A
5. Организация памяти	Пересмотренный документ DS31006A
6. Прерывания	Пересмотренный документ DS31008A
7. Система команд	Пересмотренный документ DS31029A

1.3.2 Периферийные модули

Периферийные модули - особенности, которые добавляются независимо от центрального процессора. Периферийные модули позволяют организовать интерфейс связи с внешней схемой (например, универсальные порты ввода/вывода, драйверы ЖКИ, входы АЦП, выходы ШИМ) и выполнять отсчет временных интервалов (таймеры).

Периферийные модули, описываемые в этом документе:

1. Универсальные порты ввода/вывода	Пересмотренный документ DS31009A
2. Таймер TMR0	Пересмотренный документ DS31011A
3. Таймер TMR1	Пересмотренный документ DS31012A
4. Таймер TMR2	Пересмотренный документ DS31013A
5. Захват/Сравнение/ШИМ (CCP)	Пересмотренный документ DS31014A
6. Синхронный последовательный порт (SSP)	Пересмотренный документ DS31015A
7. Основной синхронный последовательный порт (SSP)	Пересмотренный документ DS31016A
8. Ведущий синхронный последовательный порт (MSSP)	Пересмотренный документ DS31017A
9. USART	Пересмотренный документ DS31018A
10. Источник опорного напряжения	Пересмотренный документ DS31019A
11. Компараторы	Пересмотренный документ DS31020A
12. 8-разрядное АЦП	Пересмотренный документ DS31021A
13. Основное 8-разрядное АЦП	Пересмотренный документ DS31022A
14. 10-разрядное АЦП	Пересмотренный документ DS31023A
15. Интегрирующее АЦП	Пересмотренный документ DS31024A
16. Драйвер ЖКИ	Пересмотренный документ DS31025A
17. Ведомый параллельный порт (PSP)	Пересмотренный документ DS31010A

1.3.3 Специальные особенности микроконтроллеров

Специальные особенности - уникальные особенности микроконтроллера, позволяющие придать одно или более следующих свойств конечному изделию:

- Уменьшить стоимость устройства;
- Увеличить надежность системы;
- Предоставить дополнительную гибкость разработчикам при проектировании устройства.

Специальные особенности, описываемые в этом документе.

1. Биты конфигурации	Пересмотренный документ DS31027A
2. Интегрированная схема сброса по включению питания (POR)	Пересмотренный документ DS31003A
3. Схема сброса по снижению напряжения питания (BOR)	Пересмотренный документ DS31003A
4. Сторожевой таймер	Пересмотренный документ DS31026A
5. Режим энергосбережения (SLEEP)	Пересмотренный документ DS31026A
6. Интегрированный тактовый RC генератор	Пересмотренный документ DS31002A
7. Внутрисхемное программирование	Пересмотренный документ DS31028A

1.4 Поддержка разработчиков

Microchip предлагает широкую номенклатуру инструментальных средств проектирования, позволяющие проектировщикам эффективно разрабатывать и отлаживать код программы. Инструментальные средства Microchip условно можно разделить на четыре категории:

1. Генерация объектного кода;
2. Отладчики программы;
3. Программаторы;
4. Демонстрационные платы.

Все инструментальные средства, разработанные компаний Microchip, работают под управлением интегрированной среды проектирования MPLAB IDE, кроме некоторых инструментальных средств других производителей.

Состав инструментальных средств генерации объектного кода:

- MPASM;
- MPLAB-C;
- MP-DriveWay™.

В состав этих программ входят дополнительные файлы для каждого микроконтроллера. В этих файлах определяются названия регистров (в соответствии с технической документацией на микроконтроллер) с указанием адреса или номера бита. Дополнительные файлы предназначены для упрощения написание программы, исключая необходимость запоминать адреса регистров и позицию битов в регистре.

Примечание. Рекомендуется использовать дополнительные файлы при написании исходного текста программы. Это упрощает написание программы и значительно увеличивает качество технической поддержки, которую может предоставить компания Microchip.

Состав инструментальных средств отладки программы:

- Внутрисхемный эмулятор PICMASTER;
- Внутрисхемный эмулятор ICEPIC;
- Программный симулятор MPLAB-SIM.

После отладки кода программы необходимо запрограммировать микроконтроллер. Microchip предлагает два программатора разного уровня:

- PICSTART;
- PROMATE II.

Демонстрационные платы позволяют разработчикам произвести оценку возможности использования микроконтроллеров для конкретного приложения.

Предлагаемые демонстрационные платы:

- PICDEM-1;
- PICDEM-2;
- PICDEM-3;
- PICDEM-14A.

Полное описание каждого инструментального средства проектирования Microchip смотрите в разделе "Поддержка разработчиков". Поскольку постоянно разрабатываются новые средства проектирования, обновления и техническая документация на них может быть получена в региональном представительстве компании Microchip или на Web узлах технической поддержки **www.microchip.com** и **www.microchip.ru**.

Методы и рекомендации написания исходного текста программы смотрите в разделе "Создание программы" (Раздел "Создание программы" отсутствует в оригинальной документации). Для ускорения цикла проектирования Microchip предлагает дополнительный сервис:

- Примеры применения микроконтроллеров;
- Рекомендации по разработке;
- Web узел технической поддержки;
- Microchip BBS;
- Региональные представительства компании Microchip;
- Корпоративная линия поддержки.

Дополнительную информация и рекомендации могут быть найдены на Web узлах технической поддержки **www.microchip.com, www.microchip.ru** и других ресурсах сети Интернет.

1.5 Множество микроконтроллеров

После определения функциональных требований к микроконтроллеру необходимо выбрать следующие параметры:

- Технология памяти;
- Рабочий диапазон напряжения питания;
- Рабочий температурный диапазон;
- Тактовая частота;
- Тип корпуса.

Микроконтроллеры Microchip имеют большое количество параметров и их комбинаций, одна из них должна удовлетворять Вашим требованиям.

1.5.1 Технология памяти

Технология, по которой выполнена память, не влияет на логические операции микроконтроллеров. Из-за различной последовательности изготовления кристалла некоторые электрические параметры могут отличаться для микроконтроллеров с разной технологией памяти. Например, электрический параметр V_{IL} (входное напряжение низкого уровня) может отличаться в типовом микроконтроллере с EPROM памятью и типовым микроконтроллером с ROM памятью.

Каждый микроконтроллер имеет ряд диапазонов тактовой частоты и доступных упаковочных параметров. В зависимости от приложения и требований параметры микроконтроллера могут быть определены, используя систему обозначений размещенной на последней странице каждой технической документации. При размещении заказов на микроконтроллеры воспользуйтесь пожалуйста правилами системы обозначений, размещенной на последней странице каждой технической документации, чтобы правильно определить тип микроконтроллера.

При выборе функциональных возможностей микроконтроллера технология памяти и диапазон напряжения питания не имеют значения. Microchip предлагает три типа памяти программ. Код типа памяти программ обозначен символами в наименовании микроконтроллера после цифр семейства микроконтроллеров.

1. C, как в PIC16CXXX - EPROM память программ;
2. CR, как в PIC16CRXXX - ROM память программ;
3. F, как в PIC16FXXX - FLASH память программ.

1.5.1.1 EPROM

Microchip ориентируется на технологию однократно программируемой памяти программ (EPROM), чтобы дать заказчикам дополнительную гибкость проектирования устройств на любом этапе разработок. Для микроконтроллеров с EPROM технологией памяти Microchip предлагает различные упаковочные параметры и сервис.

1.5.1.2 ROM

Компания Microchip предоставляет возможность заказывать микроконтроллеры с масочной памятью. Они обеспечивают минимальную стоимость при крупносерийных заказах.

Микроконтроллеры с ROM памятью не позволяют дописать информацию в память программ по последовательному интерфейсу. Для получения информации о передаче кода ROM обратитесь к региональному представителю Microchip.

1.5.1.3 FLASH

Электрически перепрограммируемые микроконтроллеры, выпускаемые в недорогом пластмассовом корпусе. Память программ этих микроконтроллеров может быть стерта и повторно запрограммирована без удаления со схемы. Микроконтроллеры будут иметь одинаковые параметры для опытного образца, экспериментальной партии и выпуска продукции.

1.5.2 Рабочий диапазон напряжения питания

Все микроконтроллеры среднего семейства PICmicro MCU работают в стандартном диапазоне напряжения питания. Некоторые микроконтроллеры могут работать в расширенном диапазоне напряжения питания (с уменьшением тактовой частоты). В таблице 1-1 показаны все возможные типы памяти и рабочий диапазон напряжения питания для PIC16CXXX.

Указатели выделены **полужирным** шрифтом.

Таблица 1-1 Тип памяти программ и рабочий диапазон напряжения питания

Тип памяти	**Диапазон напряжения питания**	
	Стандартный	**Расширенный**
EPROM	PIC16**C**XXX	PIC16**LC**XXX
ROM	PIC16**CR**XXX	PIC16**LCR**XXX
FLASH	PIC16**F**XXX	PIC16**LF**XXX

Примечание. Не все типы памяти могут быть доступны для конкретного микроконтроллера.

В таблице 1-2 Вы можете увидеть, что если не известны точные параметры устройства, то минимальное напряжение питания для расширенного диапазона несколько ограничено. Для выполнения спецификации необходимо точно определить параметры устройства.

Таблица 1-2 Диапазон напряжения питания для каждого типа микроконтроллера

Диапазон напряжения питания[1]		**EPROM**		**ROM**		**FLASH**	
Стандартный		**C**	4.5 - 6.0 В	**CR**	4.5 - 6.0 В	**L**	4.5 - 6.0 В
Расширенный	Предварительные параметры	**LC**	3.0 - 6.0 В	**LCR**	3.0 - 6.0 В	**LF**	3.0 - 6.0 В
	Окончательные параметры [2]	**LC**	2.5 - 6.0 В	**LCR**	2.5 - 6.0 В	**LF**	2.0 - 6.0 В

Примечания:

1. Микроконтроллеры, выполненные по технологии Microchip 120K, имеют максимальное напряжение питания V_{DD} = 5.5В. В новой технической документации на микроконтроллеры учтена технология изготовления Microchip.
2. Этот диапазон напряжения питания зависит от параметров устройства.

1.5.3 Тип корпуса

В зависимости от стадии проектирования устройства может использоваться микроконтроллер в одном из следующих корпусов:

1. Корпус с окном для стирания памяти. Обычно используется керамический корпус. Микроконтроллеры в таком корпусе как правило используются на этапе проектирования, т.к. память программ может быть стерта и повторно запрограммирована много раз.
2. Недорогой пластмассовый корпус. Этот тип корпуса применяется в готовом устройстве, с целью минимизировать его стоимость.
3. DIE - проверенный, не упакованный кристалл. DIE применяется для недорогих приложений, в которых необходимо минимизировать размер печатной платы.

В таблице 1-3 представлена сводная информация.

Таблица 1-3 Типовое применение микроконтроллеров в различных корпусах

Тип корпуса	**Типовое применение**
С окном для стирания памяти	Разработка проекта
Пластмассовый	Выпуск продукции
DIE	Специальные приложения, требующие минимальные размеры печатной платы

1.5.3.1 Микроконтроллеры с ультрафиолетовым стиранием памяти

Микроконтроллеры с УФ стираемой EPROM памятью программ оптимальны для подготовки опытного образца устройства и экспериментальных программ.

Эти микроконтроллеры могут быть стерты и повторно запрограммированы в любой конфигурации. Допускается использовать программаторы других производителей для программирования микроконтроллеров с УФ стиранием (см. документацию DS00104 "Third Party Guide").

Интервал времени, необходимый для стирания памяти, зависит от следующих параметров: длина волны излучаемого УФ света, мощность источника, расстояние от источника до микроконтроллера, технология изготовления кристалла (размер ячейки памяти).

Примечание. Флуоресцентные лампы и солнечный свет излучают УФ лучи с длиной волны стирания памяти. Если окно для стирания оставлено раскрытым, то через некоторое время ячейки памяти программ могут быть стерты. Ориентировочное время стирания для флуоресцентной лампы три года, а для солнечного света около одной недели. Для предотвращения стирания ячеек памяти необходимо закрыть окно на микроконтроллере непрозрачным материалом.

1.5.3.2 Однократно программируемые микроконтроллеры OTP

OTP микроконтроллеры выпускаются в пластмассовых корпусах с однократно программируемой EPROM памятью программ. Вместе с памятью программ должны быть запрограммированы биты конфигурации. Эти микроконтроллеры предназначены для изделий, выпускаемых небольшими партиями с возможным изменением текста программы.

1.5.3.3 FLASH микроконтроллеры

FLASH микроконтроллеры позволяют выполнять электрическое перепрограммирование памяти. Устройство может быть разработано таким образом, что микроконтроллер программируется после установки его на плату. В корпусе данного вида микроконтроллеров не требуется делать окно для стирания, что позволяет их упаковывать в недорогой пластмассовый корпус.

1.5.3.4 EEPROM микроконтроллеры

EEPROM микроконтроллеры позволяют выполнять электрическое стирание памяти. Устройство может быть разработано таким образом, что микроконтроллер программируется после установки его на плату. В корпусе данного вида микроконтроллеров не требуется делать окно для стирания, что позволяет их упаковывать в недорогой пластмассовый корпус.

1.5.3.5 ROM микроконтроллеры

Код программы в память программ ROM микроконтроллеров заносится на этапе изготовления кристалла. Память программ этих микроконтроллеров не может быть изменена. ROM микроконтроллеры могут быть упакованы в недорогой пластмассовый корпус.

1.5.3.6 DIE микроконтроллеры

Опция DIE позволяет минимизировать размер печатной платы. Информацию по использованию DIE микроконтроллеров смотрите в документации DS30258 "DIE Support" и в технической документации на микроконтроллер. Использование микроконтроллеров DIE требует определенный уровень технологического оборудования и квалификации специалистов. Это означает, что возможность использования DIE технологии ограничена. Если Вы решили использовать DIE микроконтроллеры, то произведите оценку Ваших производственных мощностей на соответствие требованиям применения DIE технологии.

1.5.3.7 Специальные услуги

Для OTP микроконтроллеров Microchip предлагает две специализированные услуги, позволяющие заказчикам сокращать их производственные циклы: микроконтроллеры, программируемые производителем; серийный выпуск продукции.

1.5.3.8 Микроконтроллеры, программируемые производителем QTP

Компания Microchip предоставляет возможность заказать запрограммированные микроконтроллеры заранее предоставленным кодом. Данный сервис следует использовать при средних и больших объемах закупок микроконтроллеров и отработанном программном обеспечении. Поставляемые микроконтроллеры полностью соответствуют параметрам стандартных EPROM микроконтроллеров, за исключением того, что код программы и биты конфигурации были записаны на заводе изготовителе. Прежде чем микроконтроллеры будут поставлены заказчику, они пройдут серию испытаний на заводе изготовителе. Для получения дополнительной информации обратитесь к региональному представителю Microchip.

1.5.3.9 Серийный выпуск продукции SQTP SM

Компания Microchip предоставляет уникальную возможность заказывать запрограммированные микроконтроллеры, в которых пользователь может определить место размещения уникального серийного номера генерируемого случайным, псевдослучайным и последовательным методом. Запрограммированный уникальный серийный номер может служить: кодом доступа, паролем или идентификационным номером устройства.

1.6 Стиль и обозначения

Во всех разделах этого документа используется принятый стиль, тип шрифта и обозначения. В большинстве случаев изменение формата подразумевает, что различие выполнено для подчеркнутого текста. В документации по микроконтроллерам встречается много необычных определений и сокращений. В глоссарии, расположенном в конце документа, содержится большинство используемых сокращений и определений, применяемых в этом документе. В таблице 1-4 представлено описание основных стилей и обозначений.

1.6.1 Соглашения

Таблица 1-4 содержит описание основных стилей и обозначений, применяемых в этом документе.

Таблица 1-4 Соглашения документа

Символ или термин	Описание
Установить	Перевести бит/регистр в логическую '1'
Сбросить	Перевести бит/регистр в логический '0'
Сброс	1). Перевести бит/регистр к значению по умолчанию. 2). Условие, по которому микроконтроллер переходит в состояние после сброса. Некоторые биты/регистры примут значение '0' (например, биты разрешения прерываний), другие биты/регистры примут значение '1' (например, биты направления портов ввода/вывода).
0xnn или nnh	'nn' - число в шестнадцатеричной системе счисления. Это обозначение используется в примерах программы.
B'bbbbbbbb'	'bbbbbbbb' - число в двоичной системе счисления. Это обозначение используется в тексте, рисунках и таблицах.
R-M-W	Чтение - Модификация -Запись. Случай, когда значение регистра или порта читается, изменяется, а затем записывается обратно в регистр или порт. Это действие может быть вызвано отдельной командой (например, BSF - установить бит в регистре) или последовательностью команд.
: (двоеточие)	Используется, чтобы определить диапазон или совмещение регистров, битов, выводов. Например, TMR1H:TMR1L - совмещение двух 8-разрядных регистров для получения 16- разрядного значения таймера. Другой пример, SSPM3:SSPM0 - совмещение 4-х битов для определения режима работы модуля SSP. Порядок совмещения обычно выполняется по правилу - старший регистр/бит - младший регистр/бит с лево на право.
< >	Определение бит в регистре специального назначения. Например, SSPCON<SSPM3:SSPM0> (или SSPCON <3:0>) - определяется регистр и позиция связанных битов.
Шрифт `Courier`	Используется в примерах, написании двоичных чисел и мнемоники команд в тексте.
Шрифт Times	Используется для уравнений и переменных.
Примечание	Предназначено для обращения внимания на существенную информацию, подчеркивание важного момента, помогают избежать наиболее часто встречаемой ошибки или выделяет отличие между микроконтроллерами одного семейства. Примечание всегда выделяется затененным блоком (как показано ниже). Если примечание не указано в таблице, то оно размещается внизу таблицы (как в данном случае).
	Примечание. Здесь размещается текст примечания.
Предостережение(1)	Инструкция предостережения описывает ситуацию, в которой есть потенциальная возможность повреждения программного обеспечения или оборудования.
Предупреждение(1)	Инструкция предупреждения описывает ситуацию, в которой есть потенциальная возможность нанесения вреда персоналу.

Примечание 1. Предостережения и предупреждения предназначены для вашей безопасности. Пожалуйста, внимательно читайте каждое предостережение или предупреждение.

1.6.2 Электрические характеристики

Во всех разделах этого документа будут встречаться ссылки на спецификации электрических характеристик. Номер параметра представляется в виде уникального набора характеристик и условий, который является непротиворечивым между разделами документа. Фактическое значение параметра может отличаться в конкретном микроконтроллере.

В разделе "**Электрические характеристики**" представлены все возможные спецификации параметров для микроконтроллеров. Ни один из микроконтроллеров не содержит все представленные спецификации. Целью раздела "Электрические характеристики" является ознакомление разработчиков с типами параметров, определяемых для микроконтроллеров компанией Microchip. Значение параметров спецификации может быть недостоверным для конкретного типа микроконтроллера, хотя была попытка свести непротиворечивые параметры для всех микроконтроллеров.

Таблица 1-5 Соглашение нумерации электрических параметров

Формат номера параметра	Описание
Dxxx	Характеристика по постоянному току (DC)
Axxx	Характеристика по постоянному току для аналогового периферийного модуля (DC)
xxx	Временной параметр (AC)
PDxxx	Характеристики по постоянному току при программировании микроконтроллера (DC)
Pxxx	Временные характеристики при программировании микроконтроллера (AC)

Обозначения: xxx - номер параметра.

1.7 Техническая документация

Компания Microchip предлагает дополнительную документацию, которая может помочь в ваших разработках на микроконтроллерах PICmicro MCU. В списках этого документа представлена только общая документация. Последнюю версию технической документации Вы можете найти на Web узлах технической поддержки **www.microchip.com** и **www.microchip.ru**.

1.7.1 Документация от Microchip

Ниже представлен список документов, предоставленные компанией Microchip. Во многих перечисленных ниже документах содержится прикладная информация, в которой представлены рабочие примеры применения, программирования и проектирования PICmicro MCU.

Номер документа	Название	Описание
DS33014	**MPASM User's Guide**	Руководство пользователя по ассемблеру MPASM.
DS51014	**MPLAB™-C Compiler User's Guide**	Руководство пользователя по С компилятору MPLAB-C.
DS51025	**MPLAB User's Guide**	Руководство пользователя интегрированной среды проектирования MPLAB-IDE.
DS30420	**MPLAB Editor User's Guide**	Работа с интегрированным в MPLAB-IDE редактором.
DS30421	**PICMASTER ® User's Guide**	Техническая документация на внутрисхемный эмулятор PICMASTER.
DS30027	**MPSIM User's Guide**	Руководство пользователя по симулятору MPLAB.
DS30082	**PRO MATE ® User's Guide**	Техническая документация на программатор PROMATE.
DS51028	**PICSTART ® -Plus User's Guide**	Техническая документация на программатор PICMASTER.
DS30389	***fuzzy*TECH ® -MP User's Guide**	Руководство пользователя по использованию генератора кода *fuzzy*TECH ® -MP.
DS51027	**MP-DriveWay™ User's Guide**	Руководство пользователя по использованию генератора кода MP-DriveWay™.
DS30238	***fuzzy*TECH-MP Fuzzy Logic Handbook**	Описание принципов использования *fuzzy*TECH ® -MP.
DS00092	**Embedded Control Handbook Volume I**	В этом документе содержится много примеров использования микроконтроллеров. Документ полезен для лучшего понимания работы микроконтроллеров. Вы можете использовать в своей программе представленные примеры или их часть текста программы.
DS00167	**Embedded Control Handbook Volume II**	Вторая часть документа - примеры применения.
DS30277	**In-Circuit Serial Programming Guide™**	Техническая документация по последовательному внутрисхемному программированию микроконтроллеров.
DS351079	**PICDEM-1 User's Guide**	Описание и примеры программ к демонстрационной плате PICDEM-1.
DS30374	**PICDEM-2 User's Guide**	Описание и примеры программ к демонстрационной плате PICDEM-2.
DS33015	**PICDEM-3 User's Guide**	Описание и примеры программ к демонстрационной плате PICDEM-3.
DS00104	**Third Party Guide**	Перечень фирм, выполняющих консультативные функции по продукции Microchip.
DS30258	**DIE Support**	Техническая документация по использованию DIE микроконтроллеров.

1.7.2 Документация от других фирм

Существует несколько документов, опубликованных другими фирмами во всем мире. Microchip не рассматривает эти документы как технически точные, но они могут быть полезны для общего понимания работы микроконтроллеров PICmicro MCU. Ниже представлен далеко не полный список, он соответствует состоянию на момент подготовки данного документа. В качестве дополнительных сведений указана контактная информация авторов документов. Наиболее полную информацию по публикуемым документам Вы можете найти на Web узлах технической поддержки **www.microchip.com** и **www.microchip.ru**.

Документ	Язык
The PIC16C5X Microcontroller: A Practical Approach to Embedded Control Bill Rigby/ Terry Dalby, Tecksystems Inc. 0-9654740-0-3	Английский
Easy PIC'n David Benson, Square 1 Electronics 0-9654162-0-8	Английский
A Beginners Guide to the Microchip PIC ® Nigel Gardner, Bluebird Electronics 1-899013-01-6	Английский
PIC Microcontroller Operation and Applications DN de Beer, Cape Technikon	Английский
Digital Systems and Programmable Interface Controllers WP Verburg, Pretoria Technikon	Английский
Mikroprozessor PIC16C5X Michael Rose, Huthig 3-7785-2169-1	Немецкий
Mikroprozessor PIC17C42 Michael Rose, Huthig 3-7785-2170-5	Немецкий
Les Microcontrolleurs PIC et mise en oeuvre Christian Tavernier, Dunod 2-10-002647-X	Французский
Micontrolleurs PIC a structure RISC C.F. Urbain, Publitronic 2-86661-058-X	Французский
New Possibilities with the Microchip PIC RIGA	Русский
PIC16C5X/71/84 Development and Design, Part 1 United Tech Electronic Co. Ltd 957-21-0807-7	Китайский
PIC16C5X/71/84 Development and Design, Part 2 United Tech Electronic Co. Ltd 957-21-1152-3	Китайский
PIC16C5X/71/84 Development and Design, Part 3 United Tech Electronic Co. Ltd 957-21-1187-6	Китайский
PIC16C5X/71/84 Development and Design, Part 4 United Tech Electronic Co. Ltd 957-21-1251-1	Китайский
PIC16C5X/71/84 Development and Design, Part 5 United Tech Electronic Co. Ltd 957-21-1257-0	Китайский
PIC16C84 MCU Architecture and Software Development ICC Company 957-8716-79-6	Китайский

1.8 Дополнительная литература

В последней главе каждого раздела будут даны ссылки на другие документы или примеры применения, связанные с соответствующим разделом. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с общими сведениями по микроконтроллерам PICmicro MCU:

Документ	Номер
A Comparison of Low End 8-bit Microcontrollers Сравнение "младших" 8-разрядных микроконтроллеров	AN520
PIC16C54A EMI Results Результаты EMI для PIC16C54A	AN577
Continuous Improvement Постоянное усовершенствование	AN503
Improving the Susceptibility of an Application to ESD Улучшенная стойкость устройств к статическому электричеству (ESD)	AN595
Plastic Packaging and the Effects of Surface Mount Soldering Techniques Пластмассовый корпус и эффект пайки, методы пайки	AN598

Раздел 2.Тактовый генератор

Содержание

2.1 Введение

Для формирования тактового сигнала микроконтроллера предусмотрен внутренний генератор. Тактовый сигнал необходим для выполнения инструкций микроконтроллера и работы периферийных модулей. Внутренний машинный цикл микроконтроллера (T_{CY}) состоит из четырех периодов тактового сигнала.

Тактовый генератор микроконтроллера может работать в одном из восьми режимов. Существует два режима внутреннего RC генератора, отличающихся между собой режимом работы вывода микроконтроллера (вывод микроконтроллера работает как CLKOUT или как универсальный порт ввода/вывода). Режим работы тактового генератора определяется битами в слове конфигурации, расположенными в энергонезависимой памяти. Настроить биты конфигурации можно только при программировании микроконтроллера. Возможные режимы тактового генератора:

- LP - низкочастотный кварцевый резонатор (пониженное энергопотребление);
- XT - стандартный кварцевый/керамический резонатор;
- HS - высокочастотный кварцевый резонатор;
- RC - внешний резистор/конденсатор (идентичен EXTRC с CLKOUT);
- EXTRC - внешний резистор/конденсатор;
- EXTRC - внешний резистор/конденсатор с CLKOUT;
- INTRC - внутренний резистор/конденсатор (4МГц);
- INTRC - внутренний резистор/конденсатор (4МГц) с CLKOUT;

Различные режимы тактового генератора позволяют использовать один тип микроконтроллеров в приложениях с разными требованиями к генератору. RC режим генератора снижает стоимость устройства, а LP режим генератора имеет меньшее энергопотребление. С помощью битов конфигурации устанавливается требуемый режим тактового генератора. Дополнительную информацию о битах конфигурации смотрите в разделе "Биты конфигурации".

2.2 Настройка тактового генератора

2.2.1 Режимы тактового генератора

Среднее семейство микроконтроллеров PICmicro может иметь до восьми режимов тактового генератора. Для выбора режима тактового генератора пользователь должен запрограммировать до трех битов конфигурации (FOSC2, FOSC1 и FOSC0):

- LP - низкочастотный кварцевый резонатор (пониженное энергопотребление);
- XT - стандартный кварцевый/керамический резонатор;
- HS - высокочастотный кварцевый резонатор;
- RC - внешний резистор/конденсатор (идентичен EXTRC с CLKOUT);
- EXTRC - внешний резистор/конденсатор;
- EXTRC - внешний резистор/конденсатор с CLKOUT;
- INTRC - внутренний резистор/конденсатор (4МГц);
- INTRC - внутренний резистор/конденсатор (4МГц) с CLKOUT;

Основным отличием между режимами LP, XT и HS является значение коэффициента усиления инвертора внутренней схемы генератора. В таблице 2-1 и 2-2 указан допустимый диапазон частоты тактового генератора в различных режимах работы. Рекомендуется использовать режим тактового генератора с минимальным коэффициентом усиления для выбранной частоты, что позволяет получить меньший динамический ток потребления (I_{DD}). При выборе режима и частоты тактового генератора необходимо учитывать рекомендуемый диапазон частот и выполнение дополнительных требований (напряжение питания, рабочая температура, параметры компонентов (резистор, конденсатор, внутренняя схема генератора микроконтроллера)).

Режимы тактового генератора RC и EXTRC с CLKOUT имеют одинаковые функциональные особенности. Они имеют разные названия, чтобы облегчить описание других режимов генератора.

Таблица 2-1 Выбор режима тактового генератора для микроконтроллеров с FOSC1:FOSC0

Биты конфигурации FOSC1:FOSC0	Режим генератора	Коэффициент усиления инвертора	Описание
11	RC	-	Минимальная стоимость тактового генератора (требуется только внешний резистор и конденсатор). Большой диапазон частот тактового генератора. Режим работы по умолчанию.
10	HS	Высокий	Для приложений с высокой частотой тактового генератора. Высокий ток потребления тактового генератора из трех режимов с кварцевым резонатором.
01	XT	Средний	Стандартная частота кварцевого/керамического резонатора. Средний ток потребления тактового генератора из трех режимов с кварцевым резонатором.
00	LP	Низкий	Для приложений с низкой частотой тактового генератора. Низкий ток потребления тактового генератора из трех режимов с кварцевым резонатором.

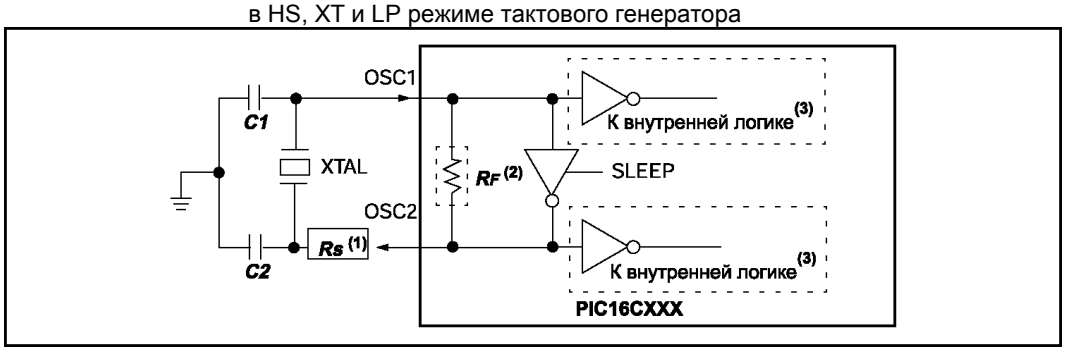
Таблица 2-2 Выбор режима тактового генератора для микроконтроллеров с FOSC2:FOSC0

Биты конфигурации FOSC2:FOSC0	Режим генератора	Коэффициент усиления инвертора	Описание
111	EXTRC с CLKOUT	-	Минимальная стоимость тактового генератора. Большой диапазон частот тактового генератора. Функция CLKOUT включена. Режим работы по умолчанию.
110	EXTRC	-	Минимальная стоимость тактового генератора. Большой диапазон частот тактового генератора. Функция CLKOUT выключена (вывод используется как порт ввода/вывода).
101	INTRC с CLKOUT	-	Минимальная стоимость тактового генератора. Настраиваемый генератор 4МГц. Функция CLKOUT включена.
100	INTRC	-	Минимальная стоимость тактового генератора. Настраиваемый генератор 4МГц. Функция CLKOUT выключена (вывод используется как порт ввода/вывода).
011	-	-	Резерв.
010	HS	Высокий	Для приложений с высокой частотой тактового генератора. Высокий ток потребления тактового генератора из трех режимов с кварцевым резонатором.
001	XT	Средний	Стандартная частота кварцевого/керамического резонатора. Средний ток потребления тактового генератора из трех режимов с кварцевым резонатором.
000	LP	Низкий	Для приложений с низкой частотой тактового генератора. Низкий ток потребления тактового генератора из трех режимов с кварцевым резонатором.

2.3 Кварцевый/керамический резонатор

В режимах тактового генератора XT, LP и HS кварцевый или керамический резонатор подключается к выводам OSC1, OSC2 (см. рисунок 2-1). Для микроконтроллеров PICmicro нужно использовать резонаторы с параллельным резонансом. Использование резонаторов с последовательным резонансом может привести к получению тактовой частоты, не соответствующей параметрам резонатора. В режимах XT, LP и HS микроконтроллер может работать от внешнего источника тактового сигнала OSC1 (см. рисунок 2-3).

Рис. 2-1 Подключение кварцевого/керамического резонатора в HS, XT и LP режиме тактового генератора



Примечания:

1. Для некоторых типов резонаторов может потребоваться последовательно включенный резистор Rs.
2. Значение резистора обратной связи RF колеблется от 2МОм до 10МОм в зависимости от режима генератора.
3. В зависимости от типа микроконтроллера буфер подключения к внутренней логике может быть на входе или на выходе инвертора.

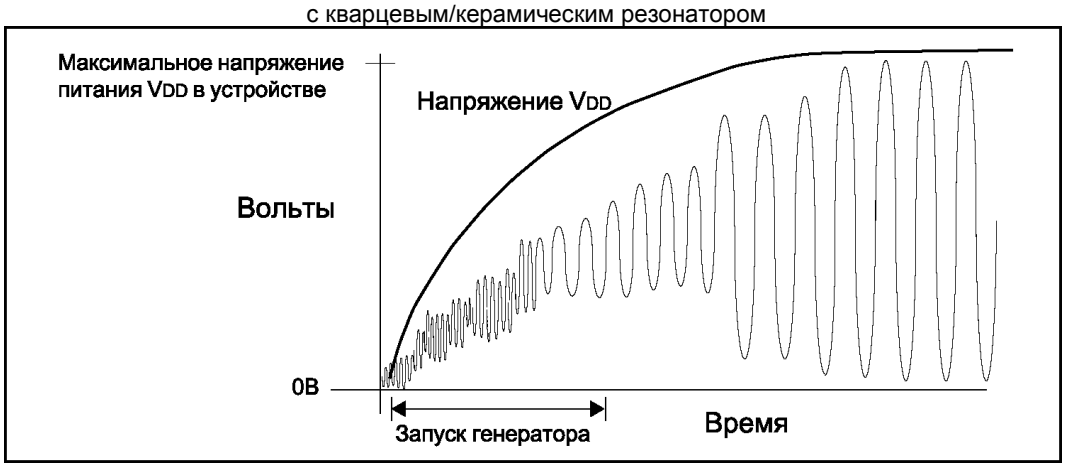
2.3.1 Запуск тактового генератора с кварцевым/керамическим резонатором

Как только напряжение питания микроконтроллера станет выше V_{SS}, тактовый генератор начнет генерацию сигнала. Время, необходимое для запуска генератора, зависит от большого числа факторов:

- Частота кварцевого керамического резонатора;
- Емкость конденсаторов C1 и C2 (см. рисунок 2-1);
- Скорость нарастания напряжения питания V_{DD};
- Рабочая температура;
- Сопротивление резистора Rs, если подключен (см. рисунок 2-1);
- Режим тактового генератора (коэффициент усиления внутреннего инвертора);
- Качество резонатора;
- Размещение компонентов тактового генератора на печатной плате;
- Помехи.

На рисунке 2-2 показана временная диаграмма запуска тактового генератор с кварцевым/керамическим резонатором. Напряжение питания, при котором форма сигнала тактового генератора удовлетворяет требования, может составлять 50% от номинального напряжения питания. Постоянная составляющая тактового сигнала равна $V_{DD}/2$ (смотрите параметры D033 и D043 в разделе "Электрические характеристики").

Рис. 2-2 Временная диаграмма запуска тактового генератора с кварцевым/керамическим резонатором



2.3.2 Выбор компонентов

На рисунке 2-1 показана схема подключения кварцевого/керамического резонатора к микроконтроллеру. Значение резистора обратной связи RF внутреннего инвертора колеблется от 2МОм до 10МОм в зависимости от режима генератора, напряжения питания и температуры. Последовательный резистор Rs может потребоваться для предотвращения возбуждения резонатора на низкой частоте. Убедитесь, что напряжение питания микроконтроллера и режим работы микроконтроллера соответствуют требованиям резистора. Внутренняя логика микроконтроллера может быть подключена к входу или выходу инвертора тактового генератора (см. рисунок 2-1). Для уточнения схемы подключения внутренней логики смотрите техническую документацию на микроконтроллер.

Типовые значения емкости конденсаторов (C1, C2) для кварцевого/керамического резонатора представлены в таблице 2-3 и 2-4. Точное значение емкости конденсаторов для конкретного микроконтроллера смотрите в соответствующей технической документации.

Таблица 2-3 Параметры конденсаторов для керамического резонатора

Протестированные диапазоны:		
Режим	**Частота**	**C1 / C2 (1)**
XT	455 кГц 2.0 МГц 4.0 МГц	22 - 100 пФ 15 - 68 пФ 15 - 68 пФ
HS	8.0 МГц 16.0 МГц 20.0 МГц	10 - 68 пФ 10 - 22 пФ TBD

Применяемые резонаторы:		
455 кГц	Panasonic EFO-A455K04B	±0.3%
2.0 МГц	Murata Erie CSA2.00MG	±0.5%
4.0 МГц	Murata Erie CSA4.00MG	±0.5%
8.0 МГц	Murata Erie CSA8.00MT	±0.5%
16.0 МГц	Murata Erie CSA16.00MX	±0.5%
20.0 МГц	TBD	TBD

Примечание 1. Значения емкости конденсаторов соответствовали указанным в таблице диапазонам. Большая емкость увеличивает стабильность генератора, но увеличивается и время запуска. Значения емкости конденсаторов являются оценочными, т.к. каждый резонатор имеет собственные характеристики. Проконсультируйтесь у производителя резонаторов для правильного подбора внешних компонентов.

Примечание 2. Для всех используемых резонаторов необходимы внешние конденсаторы.

Таблица 2-4 Параметры конденсаторов для кварцевого резонатора

Протестированные диапазоны:			
Режим	**Частота**	**C1 (1)**	**C2 (1)**
LP	32 кГц 200 кГц	68 - 100 пФ 15 - 30 пФ	68 - 100 пФ 15 - 30 пФ
XT	100 кГц 2.0 МГц 4.0 МГц	68 - 150 пФ 15 - 30 пФ 15 - 30 пФ	68 - 150 пФ 15 - 30 пФ 15 - 30 пФ
HS	8.0 МГц 10.0 МГц 20.0 МГц	15 - 30 пФ 15 - 30 пФ 15 - 30 пФ	15 - 30 пФ 15 - 30 пФ 15 - 30 пФ

Применяемые резонаторы:		
32.768 кГц	Epson C001R32.768-A	±20 PPM
100 кГц	Epson C-2 100.00 KC-P	±20 PPM
200 кГц	STD XTL 200.000 kHz	±20 PPM
2.0 МГц	ECS ECS-20-S-2	±50 PPM
4.0 МГц	ECS ECS-40-S-4	±50 PPM
8.0 МГц	ECS ECS-80-S-4	±50 PPM
10.0 МГц	ECS ECS-100-S-4	±50 PPM
20.0 МГц	ECS ECS-200-S-4	±50 PPM

Примечание 1. Большая емкость увеличивает стабильность генератора, но увеличивается и время запуска. Последовательный резистор Rs может потребоваться в HS и XT режиме для предотвращения возбуждения резонатора на низкой частоте. Значения емкости конденсаторов являются оценочными, т.к. каждый резонатор имеет собственные характеристики. Проконсультируйтесь у производителя резонаторов для правильного подбора внешних компонентов.

2.3.3 Настройка схемы генератора

Существует большое число факторов, влияющих на работу тактового генератора: рабочий диапазон (частота тактового генератора, напряжение питания, рабочая температура и режим работы); внешние компоненты (резонатор, конденсаторы, …); качество изделия и микроконтроллера. Поэтому необходимо выполнять проверку выбранных параметров, чтобы гарантировать выполнение требований приложения.

При выборе внешних компонентов необходимо учитывать большое число факторов:

- Коэффициент усиления внутреннего инвертора генератора;
- Требуемая частота;
- Частота резонатора;
- Рабочая температура;
- Диапазон напряжения питания;
- Время запуска;
- Стабильность частоты;
- Наработка микроконтроллера;
- Потребляемая мощность;
- Упрощение схемы;
- Использование стандартных компонентов;
- Схема с минимальным числом внешних компонентов.

2.3.3.1 Рекомендации по выбору кварцевого резонатора, режима тактового генератора, C1, C2 и Rs

Наилучший метод выбора внешних компонентов можно сформулировать так: на основе несложных правил создать схему, провести испытания и тестирование устройства.

Кварцевый резонатор необходимо выбирать с параллельным резонансом, но в вашем проекте могут потребоваться и другие параметры резонаторов (например, температурный дрейф или стабильность частоты). В документации AN588 Вы можете найти дополнительную информацию по выбору кварцевого резонатора.

Внутренний тактовый генератор PICmicro выполнен по схеме параллельного генератора, требующей использования кварцевых резонаторов с параллельным резонансом. Типовое значение емкости нагрузочных конденсаторов от 20пФ до 32пФ. Частота тактового сигнала, при указанной емкости нагрузочных конденсаторов, будет наиболее близкой к требуемой. Иногда может возникнуть необходимость изменить частоту генерации в небольших пределах для достижения других целей (эта возможность будет описана позже).

Режим работы тактового генератор выбирается в соответствии с технической документацией на микроконтроллер битами FOSC. Выбор режима тактового генератора (кроме RC) заключается в выборе коэффициента усиления инвертора генератора (малый коэффициент усиления - низкая частота, большой коэффициент усиления - высокая частота тактового генератора). Допускается выбирать высокий коэффициент усиления внутреннего инвертора для реализации определенных требований к схеме тактового генератора.

Первоначально емкость конденсаторов C1 и C2 выбирается в соответствии с требованиями производителя кварцевого резонатора и представленными в технической документации на микроконтроллер таблицами. Значение емкости конденсаторов, указанное в технической документации на микроконтроллер, может использоваться только как отправная точка, т.к. технология изготовление резонатора, напряжение питания и другие уже упомянутые факторы могут изменить параметры работы резонатора в вашей схеме, по сравнению с заявленными производителем.

В идеале значение емкости конденсаторов должно выбираться с учетом максимально возможной рабочей температуры и минимально возможного напряжения питания V_{DD} (в пределах, рекомендуемых изготовителем резонатора). Высокая температура и низкое напряжение питания имеют воздействие на коэффициент усиления инвертора, поэтому если устройство устойчиво работает в этом режиме, то проектировщик может быть более уверен в нормальной работе устройства при других комбинациях рабочей температуры и напряжения питания. Синусоидальный сигнал не должен ограничиваться при самом высоком коэффициенте усиления (самое высокое V_{DD} и самое низкая температура) и амплитуда сигнала на выводе должна быть достаточно большой при минимальном коэффициенте усиления инвертора (самое низкое V_{DD} и самая высокая температура), чтобы удовлетворять требованиям логического входа тактового сигнала микроконтроллера, описанным в технической документации.

Метод улучшенного запуска тактового генератор заключается в том, что значение емкости C2 выбирается больше, чем C1. Это вызывает большой сдвиг фазы при включении питания, ускоряя запуск генератора.

Помимо нагрузки резонатора для стабилизации частоты генератора конденсаторы C1 и C2 могут иметь эффект понижения коэффициента усиления при увеличении емкости. Значение емкости C2 может быть изменено с целью изменения коэффициента усиления инвертора. Большее значение емкости C2 может понизить коэффициент усиления до полного прекращения генерации (смотрите описание Rs). Значение емкости конденсаторов C1 и C2 не должно быть очень большим, поскольку это может привести к значительному снижению тока через резонатор. Измерение параметров работы тактового генератора достаточно сложный процесс, но если Вы получили результаты, незначительно отличающиеся от предположительных, то Вас это не должно беспокоить.

Последовательный резистор Rs необходимо использовать, если подбор внешних компонентов не дал удовлетворительной работы тактового генератора. Резистор может быть подключен к выводу OSC2, к которому подключен осциллограф. Подключение осциллографа к выводу OSC1 может являться причиной срыва генерации, поскольку он будет оказывать существенное влияние на отрицательную обратную связь инвертора. Необходимо учитывать то, что подключение измерительного оборудования добавляет собственную емкость к схеме. Например, если тактовый генератор лучше всего работал при емкости 20пФ и был подключен измерительный прибор с емкостью входа 10пФ, то необходимо устанавливать конденсатор в 30пФ. Сигнал с выхода не должен ограничиваться или нагружаться измерительной цепью. Перевозбуждение резонатора может привести к переходу генерации сигнала на более высокой гармонике или повреждению резонатора.

На выводе OSC2 должен присутствовать сигнал в виде чистой синусоиды, размах которой легко достигает минимального и максимального значения сигнала на тактовом входе (хороший тактовый сигнал - уровень сигнала от 4В до 5В при напряжении питания 5В). Необходимо добиться указанных параметров тактового сигнала, а затем проверить схему при минимальной температуре и максимальном V_{DD}, ожидаемом в проекте. В этом режиме будет получена максимальная амплитуда тактового сигнала. Если происходит ограничение амплитуды или постоянная составляющая смещена к V_{DD} или к V_{SS}, а значение емкости конденсаторов значительно превысило рекомендованное производителем резонатора, необходимо подключить переменный резистор между выводом микроконтроллера и конденсатором C2. Переменным резистором добиться "чистого" синусоидального сигнала. При низкой температуре и высоком напряжении питания получается максимальная амплитуда тактового сигнала, что гарантирует предотвращение перевозбуждения. Вместо переменного резистора должен быть установлен постоянный резистор с наиболее близким сопротивлением. Если сопротивление Rs более 20кОм, вход более изолирован от выхода, что делает схему более восприимчивой к шуму. Если Вы решили, что большое сопротивление Rs необходимо, то для предотвращения перевозбуждения попробуйте увеличить емкость C2. Попытайтесь получить комбинацию, в которой сопротивление Rs не более 10кОм, а значение нагрузочных конденсаторов не сильно отличается от 20пФ - 32пФ или спецификации изготовителя резонатора.

2.3.3.2 Запуск генератора

Наиболее сложные условия запуска тактового генератора возникают при выходе из режима SLEEP, потому что нагрузочные конденсаторы заряжены до некоторого постоянного значения и дифференциальная фаза при выходе из SLEEP минимальна. Поэтому требуется больший интервал времени для достижения устойчивого колебания. Необходимо также учитывать, что малое напряжение питания, высокая рабочая температура и низкая тактовая частота накладывают ограничения на коэффициент усиления инвертора генератора, который в свою очередь воздействует на запуск генерации. Каждый из следующих факторов усложняет запуск генератора:

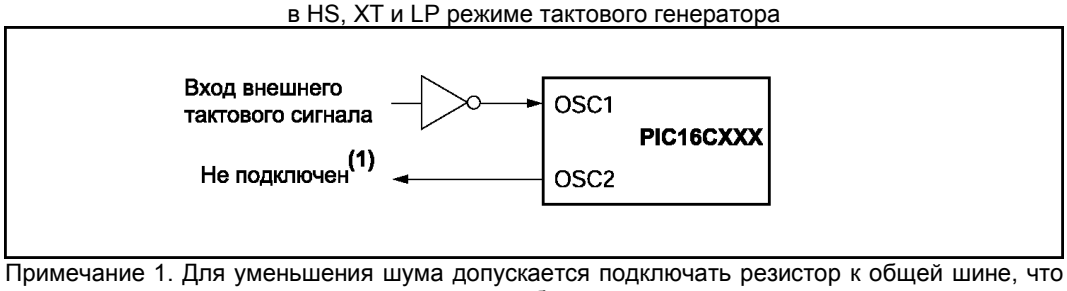
- Низкая частота тактового генератора (низкий коэффициент усиления инвертора генератора);
- Отсутствие шума по цепи питания (устройства с батарейным питанием);
- Эксплуатация устройства в условиях малого электромагнитного шума
- Малое напряжение питания;
- Высокая температура;
- Выход из режима SLEEP.

Электромагнитные помехи или помехи по цепи питания могут служить стартовым импульсом при запуске генератора.

2.3.4 Внешний тактовый сигнал

Если внутренний тактовый генератор не используется, а тактовый сигнал генерируется внешней схемой, необходимо выбрать один из режимов LP, XT или HS. Т.е. любой режим, кроме RC, т.к. в RC режиме генератора будет возникать наложение двух сигналов. В идеале нужно выбирать режим тактового генератора, соответствующий частоте внешнего тактового сигнала, но в данном случае это имеет меньшую важность, т.к. внешний тактовый сигнал сразу поступает на внутреннюю логику микроконтроллера без использования цепей внутреннего генератора. Допускается выбирать режим тактового генератора с меньшим диапазоном тактовых частот, чем частота тактового сигнала, с целью снижения потребляемого тока. Необходимо удостоверится, что амплитуда сигнала на выходе OSC2 удовлетворяет требования логических уровней микроконтроллера.

Рис. 2-3 Подключение внешнего тактового сигнала в HS, XT и LP режиме тактового генератора



Примечание 1. Для уменьшения шума допускается подключать резистор к общей шине, что может вызвать увеличение потребляемого тока.

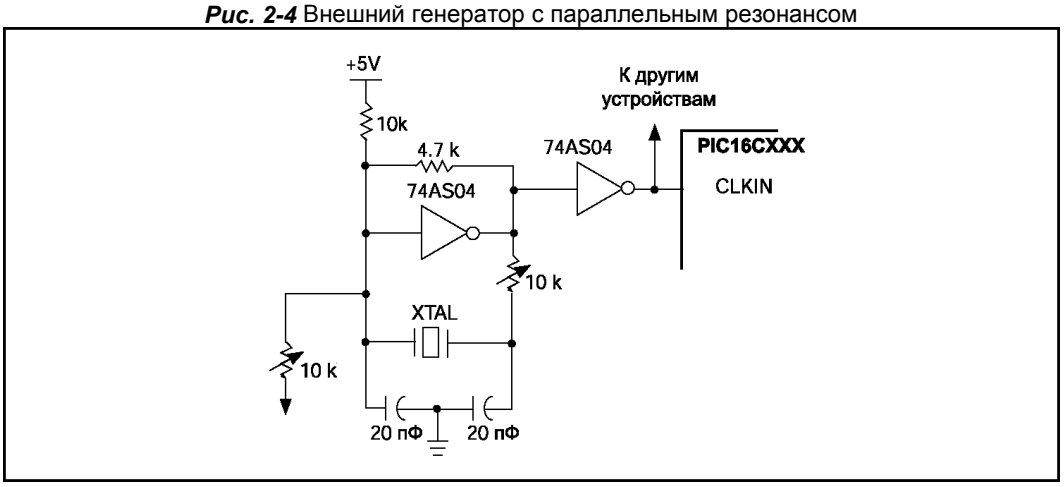
2.3.5 Внешний тактовый генератор

Иногда требуется, чтобы более одного устройства работало от тактового генератора. Фирма Microchip не рекомендует подключать дополнительные цепи к внутренней схеме тактового генератора, поэтому должна использоваться внешняя схема генератора. В этом случае каждое устройство схемы будет иметь внешний генератор тактового сигнала. Число подключаемых устройств зависит от нагрузочной способности выходного буфера тактового генератора. Применение внешнего тактового генератора полезно, когда необходимо синхронизировать работу нескольких устройств.

В качестве внешнего тактового генератора можно использовать готовый генератор, либо собрать простую схему с ТТЛ выходом. Качественный кварцевый резонатор обеспечивает высокую эффективность ТТЛ схемы. Существует две основных схемы включения кварцевых резонаторов: с параллельным резонансом, с последовательным резонансом.

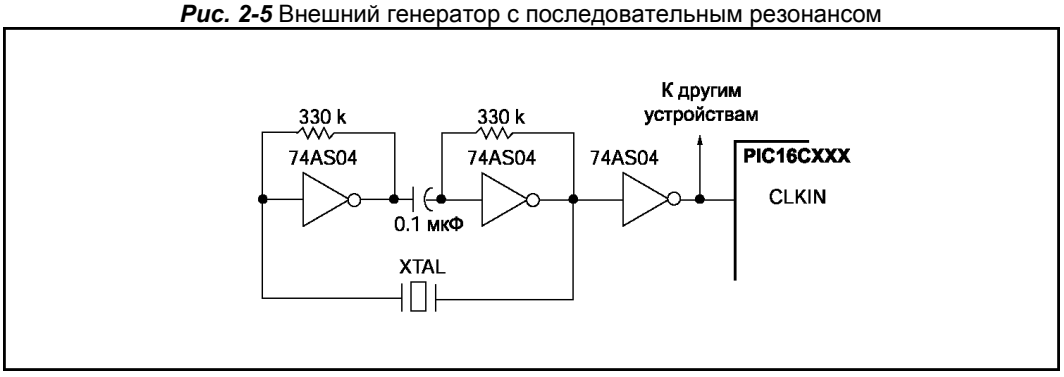
На рисунке 2-4 показана типовая схема генератора с параллельным резонансом, предназначенная для работы на основной частоте кварцевого резонатора. Инвертор 74AS04 производит необходимый для параллельного резонанса сдвиг фазы на 180°. Для обеспечения стабильности схемы в отрицательной обратной связи включен резистор 4.7кОм. Потенциометр 10кОМ предназначен для смещения рабочей точки инвертора в линейную область.

Рис. 2-4 Внешний генератор с параллельным резонансом



На рисунке 2-5 показана типовая схема генератора с последовательным резонансом, тоже предназначенная для работы на основной частоте кварцевого резонатора. Инверторы выполняют сдвиг фазы на 180°. Резисторы 330кОм создают отрицательную обратную связь для смещения рабочих точек инверторов в линейную область.

Рис. 2-5 Внешний генератор с последовательным резонансом



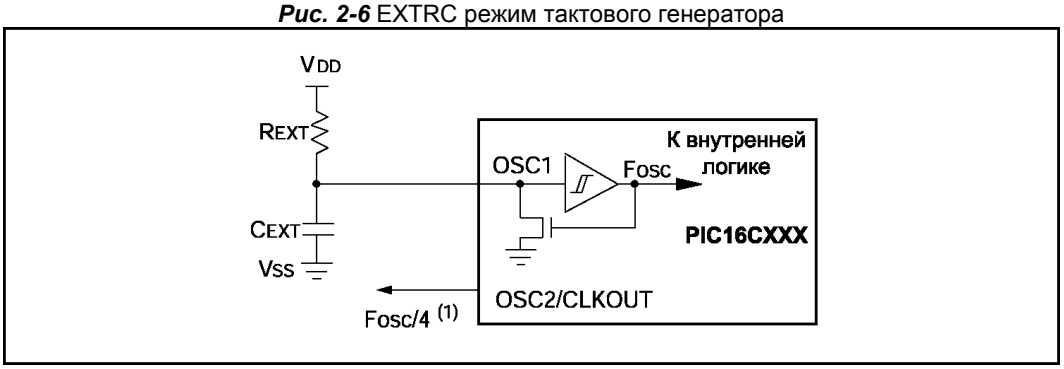
Когда микроконтроллер работает от внешнего источник тактового сигнала (см. рисунки 2-4, 2-5), тактовый генератор должен быть настроен в режиме HS, XT или LP (см. рисунок 2-3).

2.4 Внешний RC генератор

В приложениях, не требующей высокостабильной тактовой частоты, возможно использовать RC (EXTRC) режим генератора, уменьшающий стоимость устройства. Частота RC генератора зависит от напряжения питания, значения сопротивления (R_{EXT}), емкости (C_{EXT}) и рабочей температуры. Дополнительно частота будет варьироваться в некоторых пределах из-за технологического разброса параметров кристалла. Различные паразитные емкости также будут влиять на частоту генератора, особенно при малых значениях C_{EXT}. Необходимо учитывать технологический разброс параметров внешних компонентов R и C.

На рисунке 2-6 показана схема подключения RC цепочки к PIC16CXXX. Для сопротивления резистора меньше 2.2кОм частота тактового генератора может быть нестабильна или генерация может прекратиться. Для очень большого сопротивления (больше 1МОм) генератор тактового сигнала становится чувствителен к внешним помехам, токам утечки и влажности. Рекомендуется выбирать сопротивления резисторов от 3кОм до 100кОм.

Рис. 2-6 EXTRC режим тактового генератора



Примечание 1. Этот вывод может настраиваться как порт ввода/вывода.

Тактовый генератор может работать без внешнего конденсатора (C_{EXT}=0пФ), но для стабильной работы генератора рекомендуется подключать конденсатор с емкостью более 20пФ. Без внешнего конденсатора (или конденсатор имеет очень малую емкость) частота тактового генератора может зависеть от емкости проводников печатной платы и выводов компонентов.

В разделе электрических характеристик представлены данные технологического разброса частоты RC генератора. Разброс частоты возрастает с увеличением сопротивления R (т.к. возрастает влияние токов утечки) и уменьшением емкости C (т.к. усиливается влияние паразитной емкости проводников и выводов компонентов).

Также в разделе электрических характеристик показано влияние напряжения питания V_{DD} на частоту генератора при различных значениях R_{EXT}, C_{EXT} и влияние температуры для определенных значениях R,C и V_{DD}.

Для испытательных целей или для синхронизации внешней логики на выводе OSC2/CLKOUT присутствует тактовый сигнал с частотой F_{OSC}/4 (см. рисунок 4-3 в разделе "Архитектура микроконтроллеров").

2.4.1 Запуск RC генератора

Внешний RC генератор немедленно начнет формировать тактовый сигнал, после достижения порогового уровня напряжения на выводах микроконтроллера (см. параметры D032 и D042 в разделе электрические характеристики). Время запуска RC генератора зависит от большого числа факторов, вот основные из них:

- Сопротивление внешнего резистора;
- Емкость внешнего конденсатора;
- Скорость нарастания напряжения питания;
- Температура.

2.5 Внутренний RC генератор 4МГц

Внутренний тактовый генератор (не для всех микроконтроллеров) формирует тактовый сигнал с частотой 4МГц (номинальное значение) при напряжении питания V_{DD}=5В и температуре 25°С. Графики зависимости частоты внутреннего RC генератора от температуры и напряжения питания смотрите в разделе "Электрические характеристики".

Запись калибровочной константы в регистр OSCCAL позволяет устранить технологический разброс параметров внутреннего RC генератора. Биты CAL3:CAL0 используются для настройки частоты тактового генератора в пределах окна калибровки. Увеличение значения битов CAL3:CAL0 (от 0000 до 1111) приводит к увеличению тактовой частоты.

Если тактовая частота 4МГц внутреннего RC генератора не может быть достигнута изменением битов CAL3:CAL0, то частота тактового генератора может быть смещена битами CALFST и CALSLW. Эти биты дают возможность выполнить положительное или отрицательное смещение окна калибровки частоты тактового генератора.

Установка бита CALFST в '1' смещает окно калибровки к более высоким частотам RC генератора, а установка бита CALSLW в '1' смещает окно калибровки к более низкой частоте.

При сбросе микроконтроллера в регистр OSCCAL загружается значение, соответствующее среднему положению калибровки (CAL3:CAL0 = 7h, CALFST и CALSLW не создают смещения).

Регистр OSCCAL

R/W-0	R/W-1	R/W-1	R/W-1	R/W-0	R/W-0	U-0	U-0
CAL3	**CAL2**	**CAL1**	**CAL0**	**CALFST**	**CALSLW**	**-**	**-**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

биты 7-4: **CAL3:CAL0**: Биты калибровки внутреннего RC генератора
0000 = наименьшая частота в пределах окна калибровки
:
1111 = наивысшая частота в пределах окна калибровки

бит 3: **CALFST**: Бит смещения окна калибровки внутреннего RC генератора
1 = увеличение частоты тактового генератора
0 = смещения нет

бит 2: **CALSLW**: Бит смещения окна калибровки внутреннего RC генератора
1 = уменьшение частоты тактового генератора
0 = смещения нет

Примечание. Если бит CALFST = 1, то значение бита CALSLW игнорируется.

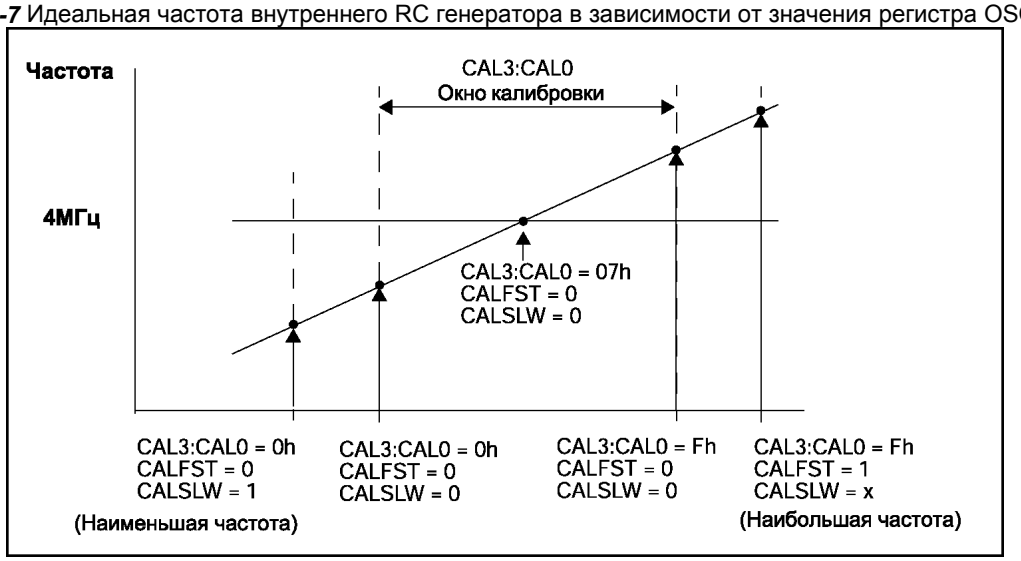
биты 1-0: **Не реализованы**: читаются как '0'

Примечание. При записи в регистр OSCAL эти биты всегда должны равняться '0' для совместимости с последующими версиями микроконтроллеров.

Примечание. Запись калибровочной константы в регистр OSCCAL позволяет устранить технологический разброс параметров внутреннего RC генератора. Калибровочное значение, сохраненное компанией Microchip, не должно изменяться. Все функции отсчета времени должны быть откорректированы программным обеспечением.

На рисунке 2-7 показана возможная частота некалиброванного тактового генератора (V_{DD} = 5В, 25°C, OSCCAL=07h) и реализуемое смещение частоты генератора за счет изменения значения регистра OSCCAL.

Рис. 2-7 Идеальная частота внутреннего RC генератора в зависимости от значения регистра OSCCAL



На рисунке 2-8 показан пример, в котором частота тактового генератора исправляется к 4МГц за счет изменения битов CAL3:CAL0. В данном случае удалось скорректировать частоту тактового генератора, изменяя только биты CAL3:CAL0. Иногда частота внутреннего RC генератора не может быть скорректирована к 4МГц изменением битов CAL3:CAL0. Поэтому были предусмотрены два дополнительных (CALSLW и CALFST), создающих большее смещение частоты генератора. После грубого смещения частоты RC генератора можно выполнить точную подстройку битами CAL3:CAL0. Действие битов CALFST и CALSLW показано на рисунках 2-9, 2-10.

Рис. 2-8 Корректировка частоты внутреннего RC генератора с помощью битов CAL3:CAL0

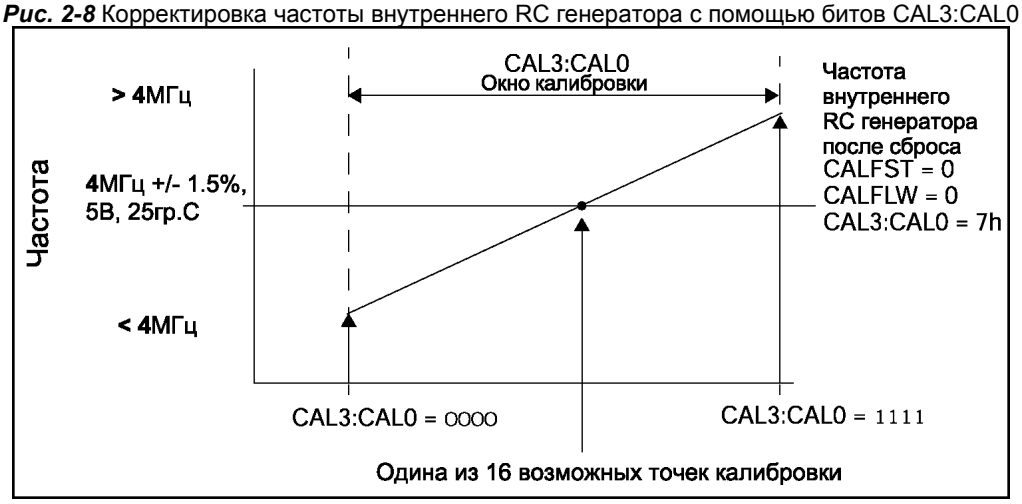


Рис. 2-9 CALFST положительное смещение частоты внутреннего RC генератора

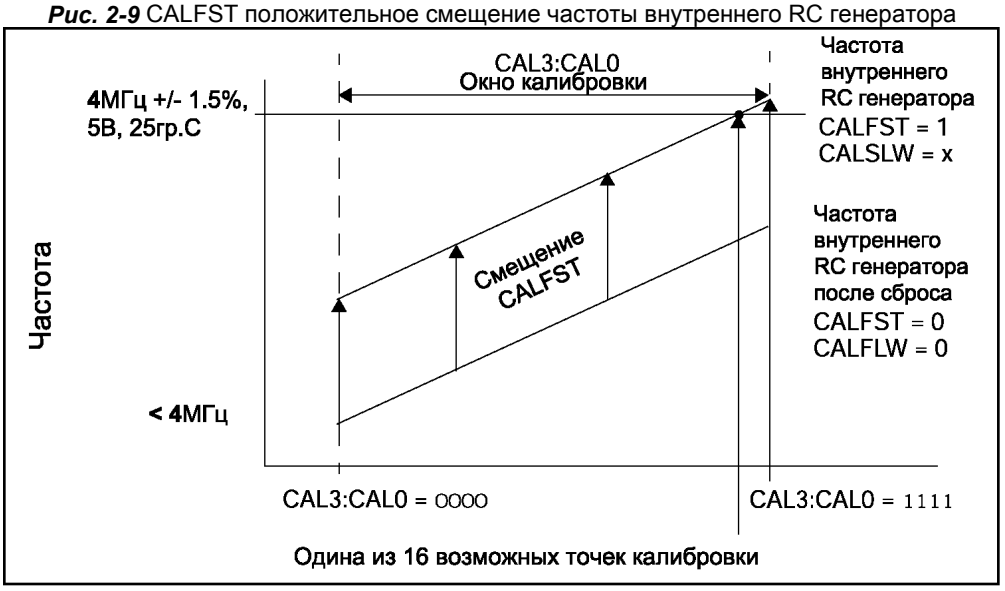
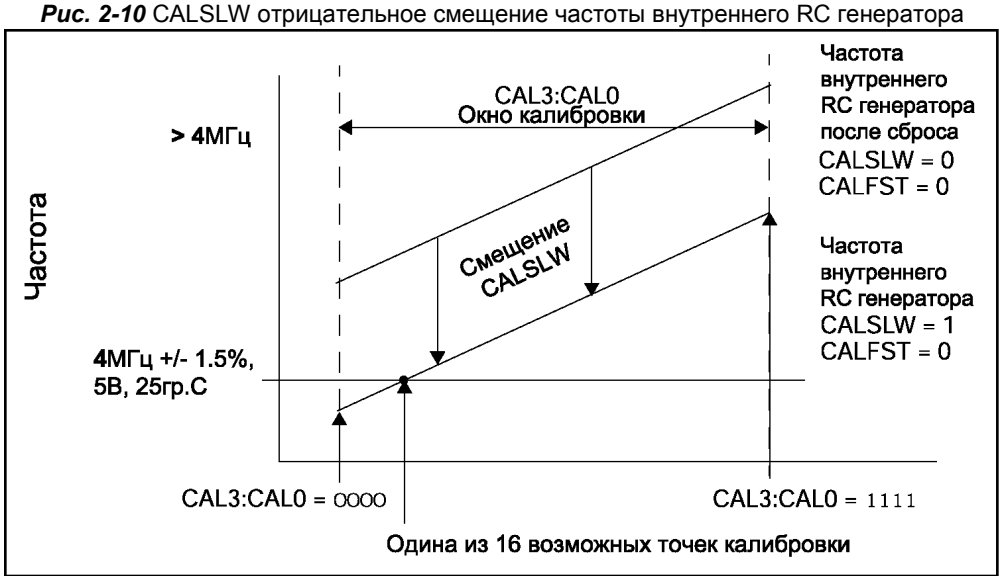


Рис. 2-10 CALSLW отрицательное смещение частоты внутреннего RC генератора



В последней ячейки памяти программ сохраняется калибровочная константа для внутреннего RC генератора. Калибровочная константа сохраняется в виде команды RETLW XX, где XX - калибровочное значение. Чтобы загрузить калибровочную константу выполните инструкцию CALL YY, где YY - последняя, доступная пользователю, ячейка памяти программ. Значение калибровки будет загружено в регистр W. Затем необходимо выполнить инструкцию MOVWF OSCCAL, чтобы загрузить калибровочную константу в регистр калибровки внутреннего RC генератора. В таблице 2-5 показано расположение калибровочной константы в зависимости от объема памяти программ.

Таблица 2-5 Размещение калибровочной константы

Объем памяти программ (слов)	Адрес калибровочной константы
512	1FFh
1к	3FFh
2к	7FFh
4к	FFFh
8к	1FFFh

Примечание 1. Стирание памяти микроконтроллера (с УФ стиранием памяти) также сотрет предварительно запрограммированную калибровочную информацию. Для сохранения калибровочной информации ее рекомендуется прочитать перед стиранием памяти микроконтроллера. Калибровочная информация должна быть восстановлена перед программированием микроконтроллера.

Примечание 2. При записи в регистр OSCCAL не реализованные биты <1:0> всегда должны равняться '0' для совместимости с последующими версиями микроконтроллеров.

2.5.1 Выход тактового сигнала

Внутренний RC генератор может быть настроен для работы в режиме формирования на выводе CLKOUT тактового сигнала с частотой $F_{OSC}/4$ (биты FOSC2, FOSC1, FOSC0 в слове конфигурации (адрес 2007h) должны равняться '101' для внутреннего RC генератора и '111' для внешнего RC генератора). Выход тактового сигнала может использоваться для измерения частоты генератора или синхронизации внешней логики.

Если калибровочная информация внутреннего RC генератора стерта, выходной тактовый сигнал позволяет скорректировать частоту генератора. Это может быть реализовано написанием дополнительной программы, изменяющей значение регистра OSCCAL. Когда на выводе CLKOUT присутствует сигнал с частотой 1 МГц (±1.5%) при V_{DD} = 5В и температуре 25°C, то значение в регистре OSCCAL правильное. Это значение должно быть передано через порты ввода/вывода для сохранения его в калибровочной ячейки памяти программ.

2.6 Воздействие режима SLEEP на тактовый генератор

При выполнении инструкции SLEEP тактовый генератор выключается, а внутренняя логика микроконтроллера переводится в начало цикла команды (такт Q1). При выключенном тактовом генераторе на выводах OSC1 и OSC2 не будет наблюдаться гармонических колебаний. Т.к. тактовый генератор выключен, то в SLEEP режиме микроконтроллера достигается наименьший ток потребления (только токи утечек). Включение любого периферийного модуля, работающего в SLEEP режиме микроконтроллера, увеличит суммарный ток потребления. Микроконтроллер может выйти из режима SLEEP по внешнему сбросу, переполнению сторожевого таймера WDT или возникновению прерывания.

Таблица 2-6 Состояние выводов OSC1 и OSC2 в SLEEP режиме

Режим генератора	Вывод OSC1	Вывод OSC2
EXTRC	Свободное состояние, внешний резистор должен подтянуть к напряжению питания	Низкий логический уровень
INTRC	-	-
LP, XT и HS	Выключена обратная связь инвертора, постоянное напряжение	Выключена обратная связь инвертора, постоянное напряжение

Время запуска генератор после сброса -MCLR и выхода из режима SLEEP смотрите в таблице 3-1 раздела "Сброс".

2.7 Воздействие сброса микроконтроллера на тактовый генератор

Сброс микроконтроллера не вызывает никакого воздействия на работу тактового генератора. Он продолжает работать, поскольку сброс произошел в нормальном режиме микроконтроллера. Сброс переводит логику микроконтроллера в исходное состояние (в начало цикла команды, такт Q1).

В режиме внешнего RC генератора (EXTRC) при использовании сигнала CLKOUT на выводе OSC2 будет присутствовать низкий логический уровень сигнала, пока на -MCLR активный уровень. Как только на выводе -MCLR появится высокий логический уровень V_{IH} , RC генератор начнет формировать тактовый сигнал.

Время запуска генератор после сброса -MCLR и выхода из режима SLEEP смотрите в таблице 3-1 раздела "Сброс".

2.7.1 Задержка сброса микроконтроллера при включении питания

Два дополнительных таймера выполняют задержку старта работы микроконтроллера. Первый, таймер запуска генератора (OST), удерживает микроконтроллер в состоянии сброса, пока не стабилизируется частота тактового генератора. Второй, таймер включения питания (PWRT), срабатывает после включения питания и удерживает микроконтроллер в состоянии сброса в течение 72мс (типовое значение), пока не стабилизируется напряжение питания. В большинстве приложений эти функции микроконтроллера позволяют исключить внешние схемы сброса. Дополнительную информацию по сбросу микроконтроллеров смотрите в разделе "Сброс".

2.8 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Подключился осциллографом к выводу OSC2, включил питание микроконтроллера. Тактовый сигнал не генерируется. Что может быть причиной этого?

Ответ 1:

1. Микроконтроллер выполнил команду SLEEP без предусмотренного условия выхода из режима SLEEP (WDT, сброс -MCLR или прерывания). Проверьте код программы на наличие команды SLEEP без предусмотренного условия выхода из режима SLEEP. Если такая ситуация возможна, то подайте на вывод -MCLR низкий логический уровень. Сброс -MCLR запустит тактовый генератор (время запуска будет чуть больше) и выведет микроконтроллер из режима SLEEP, но счетчик команд PC не будет инкрементироваться пока на выводе -MCLR не появится высокий логический уровень.
2. Выбран неправильный режим тактового генератора. Для "чистого" микроконтроллера по умолчанию установлен EXTRC режим генератора. Большинство микроконтроллеров имеют по умолчанию RC режим тактового генератора, который не будет запускать генерацию при подключенном кварцевом/керамическом резонаторе. Проверьте выбранный режим работы тактового генератора.
3. Не выдержаны требования к последовательности включения питания. Если до включения питания КМОП буфер подключен через вывод порта к другим цепям схемы с включенным питанием, могут возникать трудности запуска тактового генератора. Подобная ситуация возможна при возникновении сбросов по снижению напряжения питания (BOR), большого уровня шума при включении питания и малой скорости нарастания напряжения питания V_{DD}. Попробуйте включить микроконтроллер с не подсоединенными портами ввода/вывода и хорошей скоростью нарастания напряжения питания. Это не основная причина отсутствия генерации тактового сигнала, но теоретически она возможна. Смотрите разделы "Сброс по снижению напряжения питания" и "Последовательность включения питания" в технической документации на микроконтроллер.
4. Ненадежное подключение конденсаторов C1 и C2 к резонатору или выбраны неправильные значения емкости конденсаторов. Проверьте, что соединение выполнено правильно. Значения емкости конденсаторов, указанные в технической документации на микроконтроллеры, будут почти всегда вызывать генерацию тактового сигнала, но они могут быть не оптимальны для вашего проекта.

Вопрос 2: "Запустил" микроконтроллер PICmicro, но частота такового генератора намного больше чем частота кварцевого резонатора.

Ответ 2:

Слишком большой коэффициент усиления внутреннего инвертора тактового генератора. Смотрите главу 2.3 "Кварцевый/керамический резонатор", чтобы правильно выбрать C2 (может его емкость должна быть больше), Rs (может необходим последовательно включенный резистор) и режим тактового генератора (возможно неправильно выбран режим тактового генератора). Подобная ситуация особенно возможна при использовании низкочастотных резонаторов (например, 32.768кГц).

Вопрос 3: Микроконтроллер работает хорошо, но частота тактового генератора немного отличается от номинального значения резонатора. Как откорректировать частоту генератора?

Ответ 3:

Изменение емкости конденсатора C1 в небольших пределах повлияет на частоту тактового генератора. Если вы используете резонатор с последовательным резонансом, то частота тактового генератора будет отличаться от номинальной частоты резонатора. Необходимо использовать резонаторы с параллельным резонансом.

Вопрос 4: Устройство работает хорошо, но иногда внезапно микроконтроллер приостанавливается.

Ответ 4:

Необходимо выполнять программную проверку для исследования остановки микроконтроллера. Такая ситуация возможна, если амплитуда тактового сигнала не достаточно большая, чтобы надежно формировать тактовые импульсы.

Вопрос 5: В моем устройстве микроконтроллер работает в режиме внутреннего RC генератора. При стирании микроконтроллера стерлась калибровочная информация. Что можно сделать?

Ответ 5:

Если частота микроконтроллера не имеет критического значения, Вы можете продолжать его использовать. В противном случае Вам нужно приобрести новый микроконтроллер или воспользоваться методикой, описанной в параграфе 2.5.1 " Выход тактового сигнала"

2.9 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные тактовым генератором микроконтроллеров PICmicro MCU:

Документ	Номер
PIC16/17 Oscillator Design Guide Руководство по выбору тактового генератора для PIC16/17	AN588
Low Power Design using PIC16/17 Устройства с малым энергопотреблением на микроконтроллерах PIC16/17	AN606

Раздел 3.Сброс

Содержание

3.1 Введение

Логика сброса предназначена для перевода микроконтроллера в исходное состояние с заведомо известными параметрами работы. Источник сброса микроконтроллера может быть идентифицирован с помощью битов состояния. Особенности логики сброса позволяют снизить стоимость устройства и увеличить его надежность.

Микроконтроллеры среднего семейства различают следующие виды сброса:

a) Сброс по включению питания (POR);
b) Сброс по сигналу -MCLR в нормальном режиме работы;
c) Сброс по сигналу -MCLR в режиме SLEEP;
d) Сброс по переполнению WDT в нормальном режиме;
e) Сброс по снижению напряжения питания (BOR);
f) Сброс по ошибке паритета (PER);

Большинство регистров не изменяются после любого вида сброса, но после сброса по включению питания POR они содержат неизвестное значение. Некоторые регистры сбрасываются в начальное состояние при сбросах POR, BOR, -MCLR и WDT в нормальном режиме, -MCLR в режиме SLEEP, PER.

Бит паритета может использоваться для проверки содержимого памяти программ.

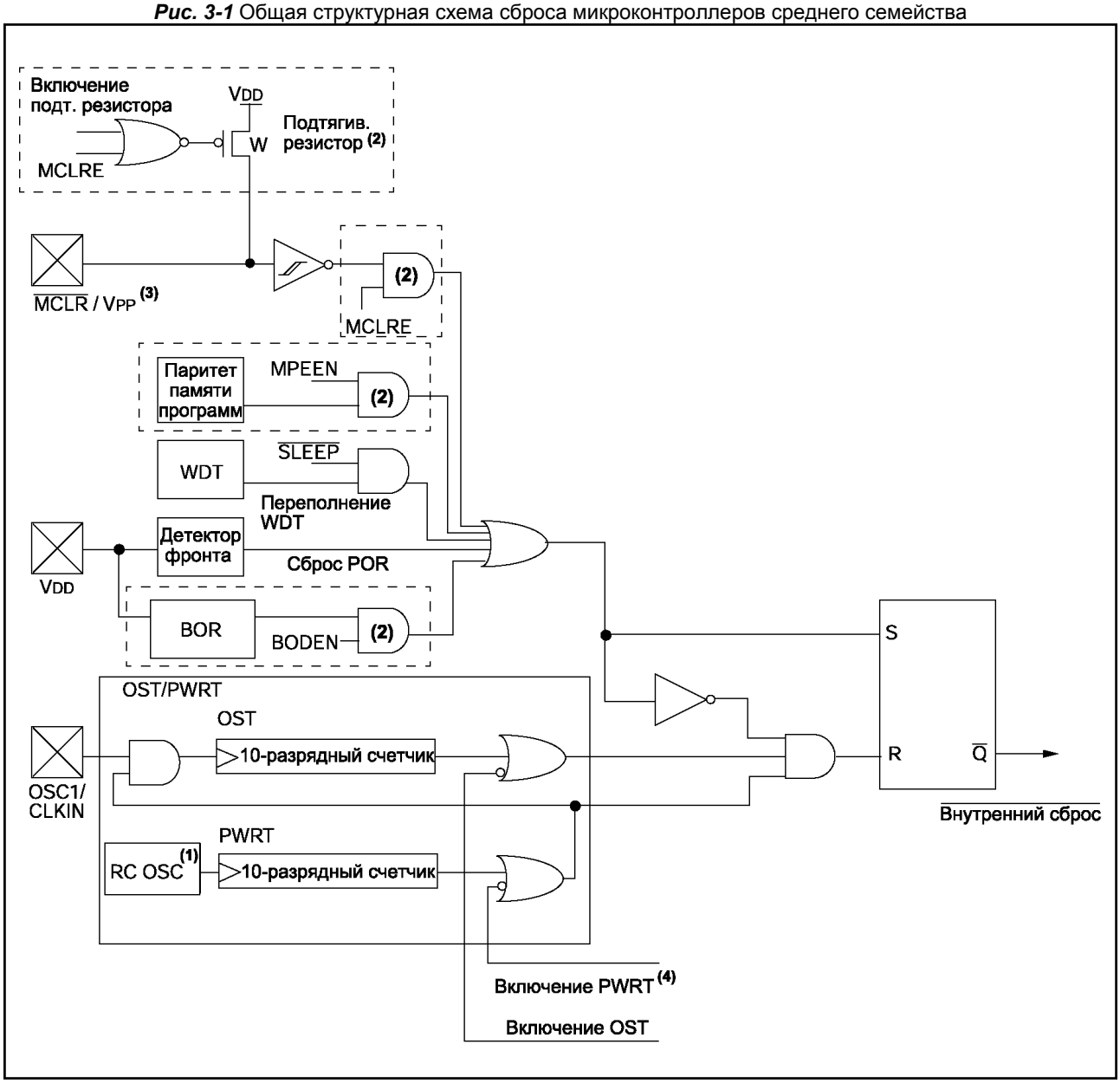
Сброс WDT в SLEEP режиме рассматривается как возобновление нормальной работы и на значение регистров не влияет. Биты -TO, -PD, -POR, -BOR и -PER принимают определенные значения при различных видах сброса (см. таблицу 3-2). Программное обеспечение может использовать эти биты для детектирования вида сброса микроконтроллера. Состояние регистров специально назначения после сброса смотрите в таблице 3-4.

Упрощенная структурная схема сброса показана на рисунке 3-1. На этом рисунке представлены все возможные виды сброса. Чтобы определить какие виды сброса реализованы на конкретном микроконтроллере, воспользуйтесь технической документацией на соответствующий микроконтроллер.

Примечание. Пока микроконтроллер PICmicro находится в состоянии сброса внутренний тактовый сигнал соответствует такту Q1 (начало цикла команды).

Во всех новых микроконтроллерах на входе -MCLR есть внутренний фильтр, не пропускающий короткие импульсы (см. параметр 30 в разделе "Электрические характеристики"). Необходимо отметить, что сброс WDT не управляет выводом -MCLR.

Рис. 3-1 Общая структурная схема сброса микроконтроллеров среднего семейства



Примечания:

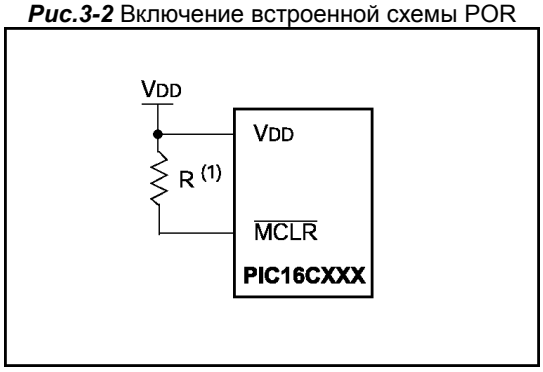
1. Это отдельный RC генератор от вывода OSC1/CLKIN (не внутренний тактовый RC генератор).
2. Блоки, обведенные пунктирной линий, доступны не на всех микроконтроллерах. Смотрите техническую документацию на микроконтроллер.
3. В некоторых микроконтроллерах этот вывод может работать как вход порта ввода/вывода.
4. В ранее выпускаемых микроконтроллерах PWRT включен, когда бит PWRTE в слове конфигурации равнялся '1' (в настоящее время выпускаются микроконтроллеры, в которых работа PWRT разрешена, когда PWRTE=0).
5. Время удержания микроконтроллера в состоянии сброса смотрите в таблице 3-1.

3.2 POR, PWRT, OST, BOR, PER

3.2.1 Сброс по включению питания POR

Интегрированная схема POR удерживает микроконтроллер в состоянии сброса, пока напряжение V_{DD} не достигнет требуемого уровня. Для включения схемы POR необходимо соединить вывод -MCLR с V_{DD} через резистор (см. рисунок 3-2), не требуя внешней RC цепочки, обычно используемой для сброса. Максимальное время нарастания V_{DD} смотрите в разделе "Электрические характеристики" (параметры D003, D004).

Рис.3-2 Включение встроенной схемы POR

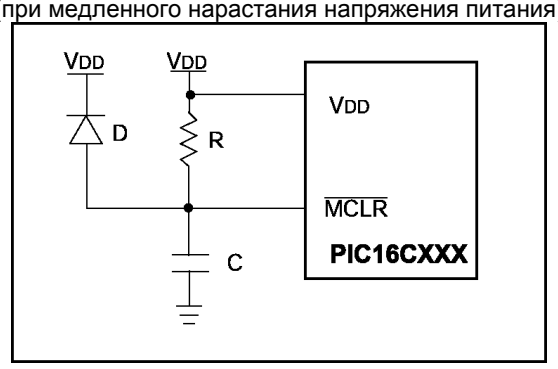


Примечание 1. Для некоторых микроконтроллеров подключение резистора не обязательно.

Когда микроконтроллер переходит в режим нормальной работы из состояния сброса, рабочие параметры (напряжение питания, частота, температура и т.д.) должны соответствовать указанным в разделе "Электрические характеристики". Если рабочие параметры не удовлетворяют требованиям, микроконтроллер должен находиться в состоянии сброса.

На рисунке 3-3 показана схема внешнего сброса POR при медленном нарастании напряжения питания. Внешняя схема сброса требуется только, если скорость нарастания напряжения питания очень мала. Диод D предназначен для быстрой разрядки конденсатора при снижении напряжения питания.

Рис. 3-3 Схема внешнего сброса по включению питания
(при медленного нарастания напряжения питания)



Примечание. Сопротивление резистора R рекомендуется выбирать меньше 40кОм, чтобы падение напряжения на резисторе не превышало 0.2В. Большее падение напряжения на резисторе может приблизить уровень сигнала на выводе -MCLR/V_{PP} к минимальному V_{IH}.

3.2.2 Таймер включения питания PWRT

Таймер включения питания обеспечивает задержку в 72мс (номинальное значение) по сигналу схемы сброса включения питания POR или сброса по снижению напряжения питания BOR (см. параметр 33 в разделе "Электрические характеристики"). Таймер включения питания работает от отдельного внутреннего RC генератора и удерживает микроконтроллер в состоянии сброса по активному сигналу от PWRT. Задержка PWRT позволяет достигнуть напряжению V_{DD} номинального уровня.

Битом -PWRTE в слове конфигурации можно выключить (-PWRTE=1) или включить (-PWRTE=0) таймер включения питания. Таймер PWRT должен быть включен, если используется сброс по снижению напряжения питания BOR. В ранее выпускаемых микроконтроллерах PWRT включен, когда бит PWRTE в слове конфигурации равнялся '1' (в настоящее время выпускается микроконтроллеры, в которых работа PWRT разрешена, когда PWRTE=0).

Время задержки PWRT варьируется в каждом микроконтроллере и зависит от напряжения питания V_{DD}, температуры и технологического разброса параметров кристалла (см. раздел "Электрические характеристики").

3.2.3 Таймер запуска генератора OST

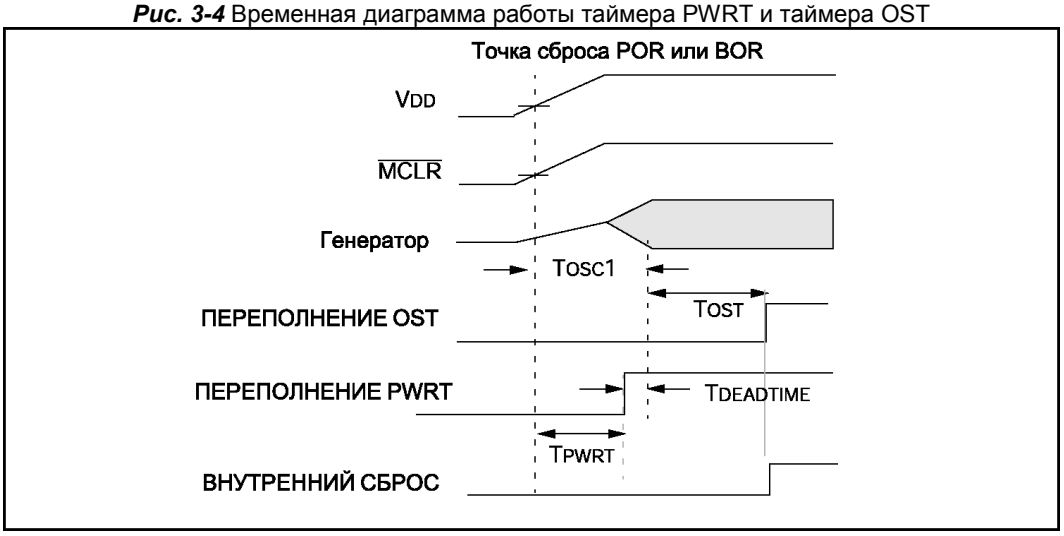
Таймер запуска генератора (OST) обеспечивает задержку в 1024 такта генератора (вход OSC1) после окончания задержки от PWRT (если она включена). Это гарантирует, что частота кварцевого/керамического резонатора стабилизировалась.

Задержка OST включается только в HS, XT и LP режимах тактового генератора после сброса POR, BOR или выхода микроконтроллера из режима SLEEP.

Счетчик OST инкрементируется по каждому импульсу со входа OSC1/CLCIN после того, как амплитуда сигнала достигнет порога входного буфера генератора. Задержка OST гарантирует стабилизацию частоты генератора с кварцевым/керамическим резонатором перед началом выполнения программы микроконтроллера. Длительность задержки OST зависит от частоты резонатора.

На рисунке 3-4 показан временная диаграмма работы схемы таймера OST вместе со схемой таймера PWRT. Для низкочастотных резонаторов длительность запуска может быть достаточно большая. Это происходит из-за того, что время запуска генератора на низкой частоте значительно больше задержки таймера PWRT. Интервал времени от момента окончания счета таймера PWRT до начала генерации тактового сигнала называется "мертвая зона". В электрических спецификациях не существует понятия минимальная/максимальная длительность "мертвой зоны" ($T_{DEADTIME}$).

Рис. 3-4 Временная диаграмма работы таймера PWRT и таймера OST



$T_{OSC}1$ = Интервал времени от сброса POR или BOR до обнаружения таймером запуска генератора (OST) тактовых импульсов.

$T_{OST} = 1024\ T_{OSC}$

3.2.4 Последовательность удержания микроконтроллера в состоянии сброса

При включении питания выполняется следующая последовательность удержания микроконтроллера в состоянии сброса: обнаружен сброс POR, задержка PWRT (если она разрешена), задержка OST (после завершения задержки PWRT). Полное время задержки изменяется в зависимости от режима работы тактового генератора и состояния бита -PWRTE. Например, в режиме RC генератора и при -PWRTE=1 (таймер PWRT выключен) задержка будет отсутствовать. На рисунках 3-5, 3-6 и 3-7 показаны последовательности удержания микроконтроллера в состоянии сброса.

Если сигнал -MCLR удерживается в низком уровне достаточно долго (дольше времени всех задержек), после перехода -MCLR в высокий уровень программа начнет выполняться немедленно (см. рисунок 3-7). Это может быть полезно при одновременном запуске нескольких микроконтроллеров, работающих параллельно.

Если напряжение питания не соответствует спецификации электрических характеристик после окончания счета таймеров OST, PWRT (если они включены), то на выводе -MCLR/V_{PP} должен присутствовать низкий логический уровень пока напряжение не достигнет требуемого уровня. Использование внешней RC цепочки для формирования сброса по включению питания достаточно для большинства приложений. На рисунке 3-8 показана временная диаграмма сброса микроконтроллера при медленном нарастании напряжения питания.

В таблице 3-1 представлена длительность задержки старта микроконтроллера в различных ситуациях.

Таблица 3-1 Время задержки при различных видах сброса

Режим генератора	Сброс POR		Сброс BOR	Выход из режима SLEEP
	-PWRTE=0	-PWRTE=1		
XT, HS, LP	72мс + 1024 T_{OSC}	1024 T_{OSC}	72мс + 1024 T_{OSC}	1024 T_{OSC}
RC	72мс	-(1)	72мс	-(1)

Примечание 1. Для микроконтроллером с внутреннем/внешнем RC генератором время запуска ориентировочно равно 250мкс.

Рис. 3-5 Временная диаграмма задержки при включении питания (-MCLR подключен к V_{DD})

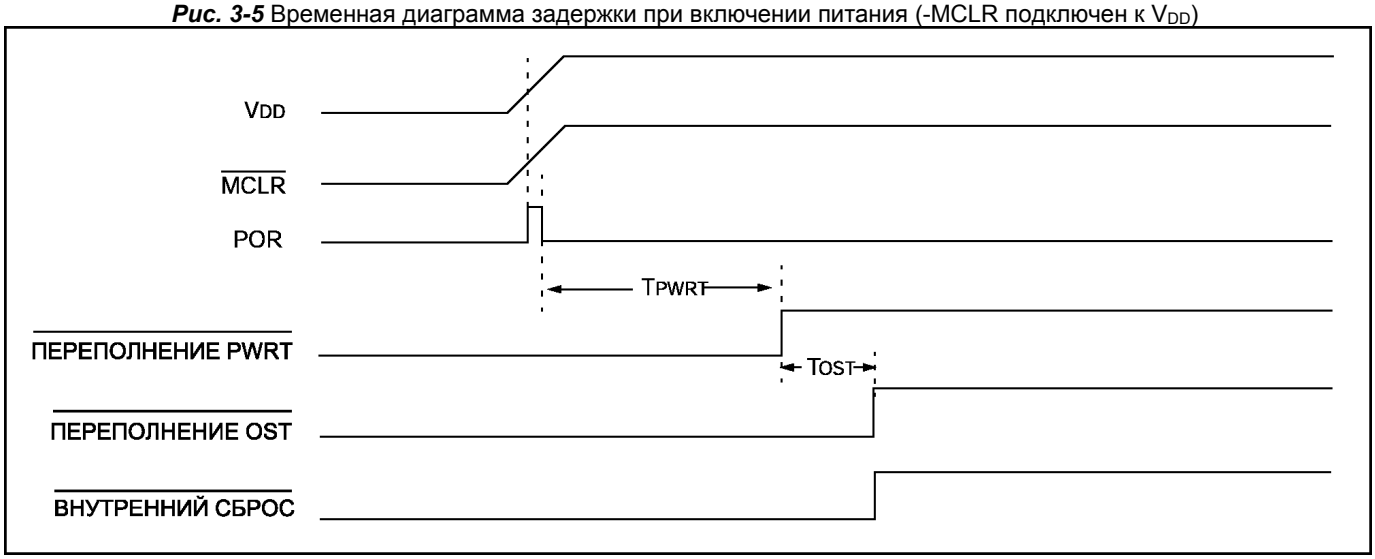


Рис. 3-6 Временная диаграмма задержки при включении питания (-MCLR не подключен к V_{DD}), случай 1

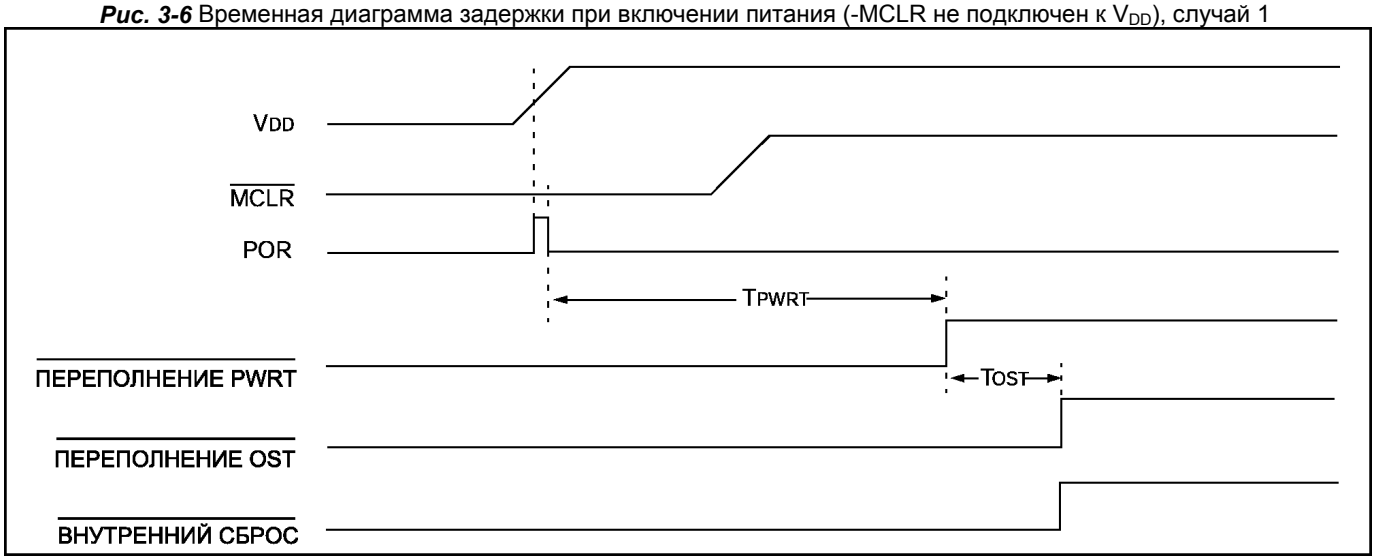


Рис. 3-7 Временная диаграмма задержки при включении питания (-MCLR не подключен к V_{DD}), случай 2

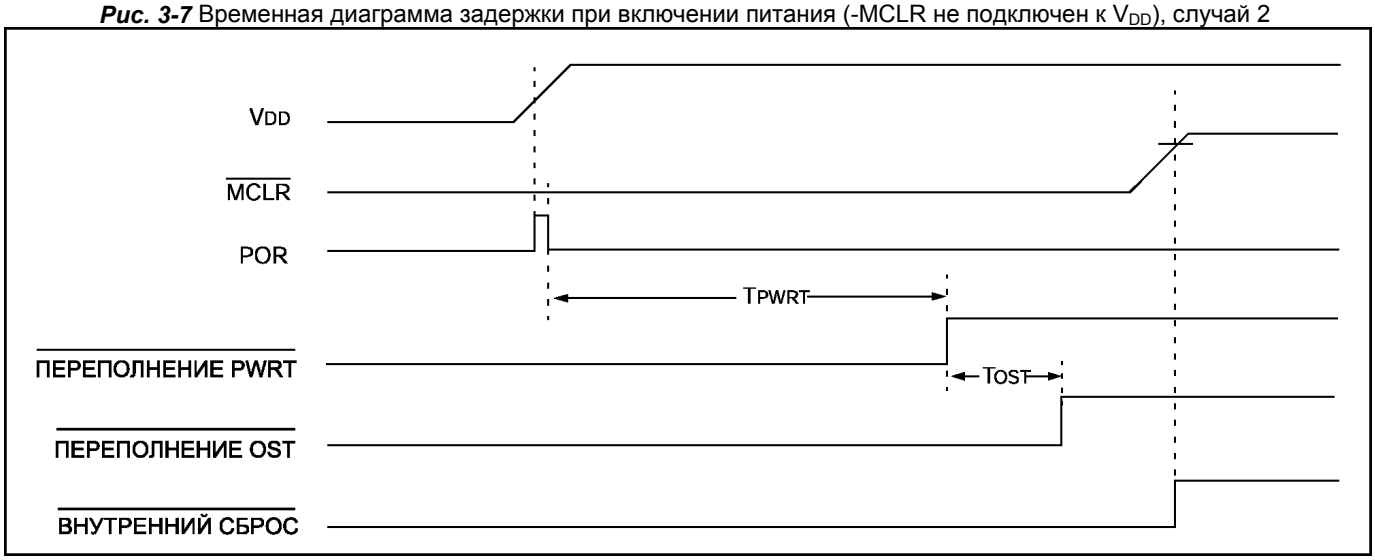
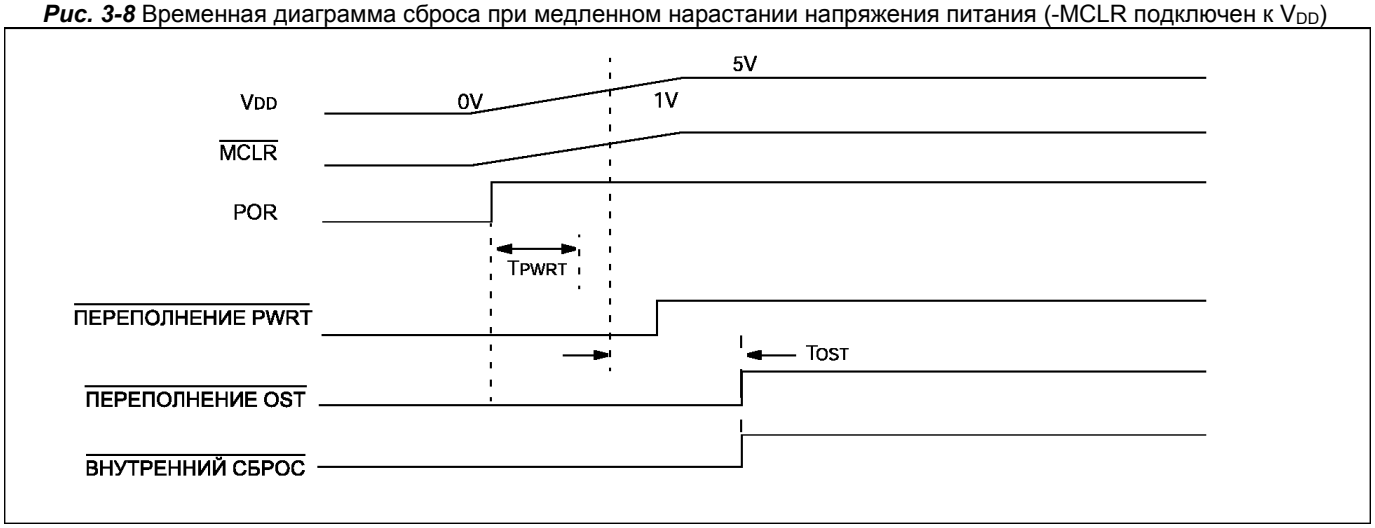


Рис. 3-8 Временная диаграмма сброса при медленном нарастании напряжения питания (-MCLR подключен к V_{DD})



3.2.5 Сброс по снижению напряжения питания BOR

Интегрированная схема BOR переводит микроконтроллер в состояние сброса, когда напряжение питания ниже установленного уровня BV_{DD}, что гарантирует прекращение выполнения программы при выходе напряжения питания за установленные нормы. Сброс по снижению напряжения питания обычно используется в приложениях с питанием от сети переменного тока или в устройствах с питанием от аккумуляторных батарей, в которых возможно кратковременное уменьшение питающего напряжения ниже минимального напряжения питания микроконтроллера вследствие коммутации нагрузки большой мощности (например в автомобильных приложениях).

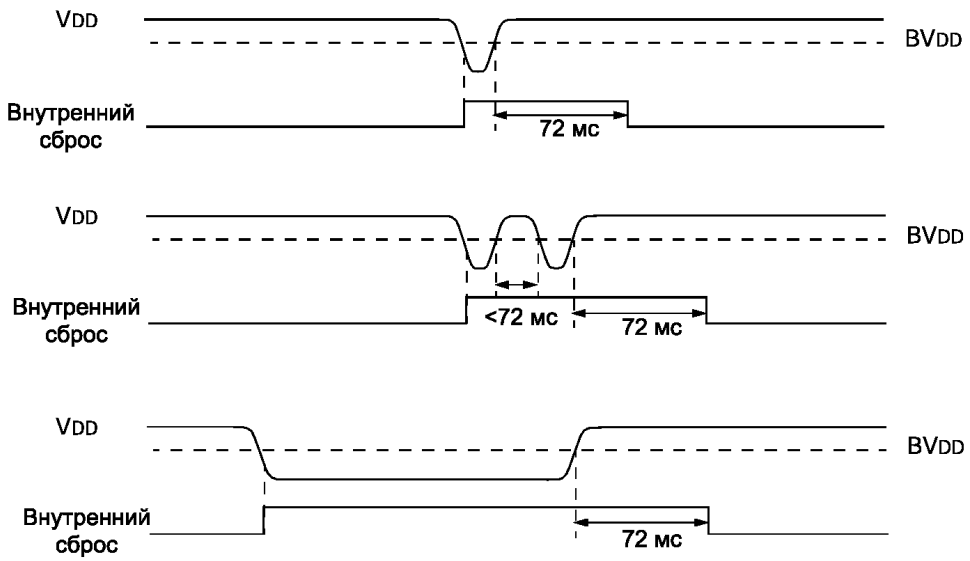
Примечание. Перед использованием сброса по снижению напряжения питания обязательно изучите электрические характеристики микроконтроллера, чтобы гарантировать выполнение требований вашего приложения.

Битом BODEN в слове конфигурации можно выключить (BODEN = 0) или включить (BODEN = 1) детектор снижения напряжения питания. Если напряжение V_{DD} опускается ниже BV_{DD} (типовое значение 4.0В, см. параметр D005 в разделе "Электрические характеристики") на время больше(или равное) T_{BOR} (см. параметр 35, 100мкс), произойдет сброс по снижению напряжения питания. Если длительность снижения напряжения питания меньше T_{BOR}, сброс микроконтроллера не произойдет.

При любом виде сброса (POR, -MCLR, WDT и т.д.) микроконтроллер находится в состоянии сброса, пока напряжение V_{DD} не будет выше BV_{DD}. После нормализации напряжения питания микроконтроллер находится в состоянии сброса еще 72мс (см. параметр 33, T_{PWRT}). Если напряжение питания V_{DD} стало ниже BV_{DD} во время работы таймера по включению питания, микроконтроллер возвращается в состояние сброса BOR, а таймер инициализируется заново (см. рисунок 3-9). Каждый переход напряжения питания V_{DD} через границу BV_{DD} инициализирует PWRT, создавая задержку в 72мс. При включении схемы сброса BOR всегда нужно включать таймер PWRT.

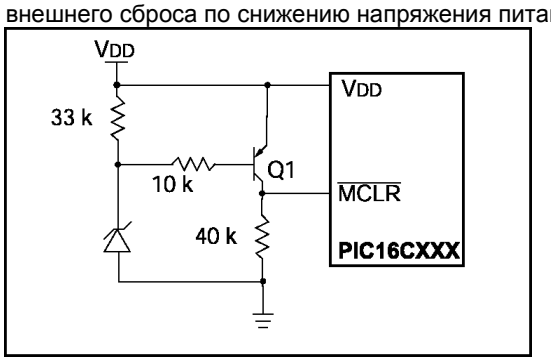
Если BODEN=1, то при каждом снижении напряжения питания ниже BV_{DD} микроконтроллер будет переходить в состояние сброса, включая задержку внутреннего сброса при включении питания.

Рис. 3-9 Типовые ситуации сброса BOR



Некоторые микроконтроллеры не содержат интегрированной схемы сброса по снижению напряжения питания или возможны ситуации, когда параметры схемы сброса BOR не удовлетворяют требованиям приложения. В этом случае необходимо использовать внешнюю схему сброса. На рисунках 3-10 и 3-11 представлено два варианта внешней схемы сброса при снижении напряжения питания. Выбор внешней схемы сброса необходимо выполнять в соответствии с требованиями приложения.

Рис. 3-10 Схема внешнего сброса по снижению напряжения питания (1 вариант)



Примечания:

1. Эта схема будет сбрасывать микроконтроллер, когда V_{DD} ниже V_Z + 0.7В, где V_Z–напряжение стабилизации стабилитрона.
2. Внутренняя схема сброса по снижению напряжения питания должна быть выключена.
3. Номиналы резисторов должны быть выбраны с учетом типа транзистора.

Рис. 3-11 Схема внешнего сброса по снижению напряжения питания (2 вариант)

Примечания:

1. Недорогая схема сброса, но менее точная по сравнению с 1 вариантом. Транзистор Q1 закрывается, когда напряжение питания ниже определенного порога.

$$Vdd \bullet \frac{R1}{R1+R2} = 0.7$$

2. Внутренняя схема сброса по снижению напряжения питания должна быть выключена.
3. Номиналы резисторов должны быть выбраны с учетом типа транзистора.

3.3 Состояние регистров и битов после сброса

Таблица 3-2 Состояние некоторых битов регистров STATUS/PCON

-POR	-BOR	-TO	-PD	Условие
0	x	1	1	Сброс по включению питания
0	x	0	x	Недействительный -TO, если -POR=0
0	x	x	0	Недействительный -PD, если -POR=0
1(2)	0	1	1	Сброс по снижению напряжения питания
1(2)	1(2)	0	1	Сброс от WDT
1(2)	1(2)	0	0	Выход из режима SLEEP от WDT
1(2)	1(2)	u	u	Сброс -MCLR при нормальном режиме работы
1(2)	1(2)	1	0	Сброс -MCLR в SLEEP режиме

Обозначения: u = не изменяется; x = неопределенное значение

Примечания:

1. Не все микроконтроллеры содержат схему сброса BOR.
2. Эти биты для данных условий не изменяются. Они должны быть установлены в '1' после сброса POR, BOR.

Таблица 3-3 Состояние особых регистров после сброса

Вид сброса	Счетчик команд PC	Регистр STATUS	Регистр PCON
Сброс по включению питания	000h	0001 1xxx	u--- -10x
Сброс по сигналу -MCLR в нормальном режиме	000h	000u uuuu	u--- -uuu
Сброс по сигналу -MCLR в SLEEP режиме	000h	0001 0uuu	u--- -uuu
Сброс от WDT	000h	0000 1uuu	u--- -uuu
Выход из режима SLEEP от WDT	PC + 1	uuu0 0uuu	u--- -uuu
Сброс по снижению напряжения питания	000h	0001 1uuu	u--- -uu0
Выход из режима SLEEP от прерываний	PC + 1(1)	uuu1 0uuu	u--- -uuu

Обозначения: - = не используется, читается как '0'; u = не изменяется; x = не известно.

Примечания:

1. При выходе из режима SLEEP по возникновению прерывания, если GIE=1, в счетчик команд PC загружается вектор прерываний (0004h) после выполнения PC+1.
2. Если бит в регистре STATUS не реализован, он читается как '0'.

Таблица 3-4 Состояние регистров специального назначения после сброса

Регистр	Сброс POR или BOR	Сброс -MCLR или WDT	Выход из режима SLEEP по прерыванию или WDT
ADCAPL	0000 0000	0000 0000	uuuu uuuu
ADCAPH	0000 0000	0000 0000	uuuu uuuu
ADCON0	0000 00-0	0000 00-0	uuuu uu-u
ADCON1	0--- 0000	0--- 0000	u--- uuuu
ADRES	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADTMRL	0000 0000	0000 0000	uuuu uuuu
ADTMRH	0000 0000	0000 0000	uuuu uuuu
CCP1CON	--00 0000	--uu uuuu	--uu uuuu
CCP2CON	0000 0000	0000 0000	uuuu uuuu
CCPR1L	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1H	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR2L	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR2H	xxxx xxxx	uuuu uuuu	uuuu uuuu
CMCON	00-- 0000	00-- 0000	uu-- uuuu
EEADR	xxxx xxxx	uuuu uuuu	uuuu uuuu
EECON1	x--- x000	u--- u000	u--- uuuu
EECON2	-	-	-
EEDATA	xxxx xxxx	uuuu uuuu	uuuu uuuu
FSR	xxxx xxxx	uuuu uuuu	uuuu uuuu
GPIO	--xx xxxx	--uu uuuu	--uu uuuu
I2CADD	0000 0000	0000 0000	uuuu uuuu
I2CBUF	xxxx xxxx	uuuu uuuu	uuuu uuuu
I2CCON	0000 0000	0000 0000	uuuu uuuu
I2CSTAT	--00 0000	--00 0000	--uu uuuu
INDF	-	-	-
INTCON	0000 000x	0000 000u	uuuu uuuu(1)
LCDCON	00-0 0000	00-0 0000	uu-u uuuu
LCDD00:LCDD15	xxxx xxxx	uuuu uuuu	uuuu uuuu
LCDPS	---- 0000	---- 0000	---- uuuu
LCDSE	1111 1111	1111 1111	uuuu uuuu
OPTION_REG	1111 1111	1111 1111	uuuu uuuu
OSCCAL	0111 00--	uuuu uu--	uuuu uu--
PCL	0000 0000	0000 0000	PC+ 1 (2)
PCLATH	---0 0000	---0 0000	---u uuuu
PCON	---- --0u	---- --uu	---- --uu
PIE1	0000 0000	0000 0000	uuuu uuuu
PIE2	---- ---0	---- ---0	---- ---u
PIR1	0000 0000	0000 0000	uuuu uuuu
PIR2	---- ---0	---- ---0	---- ---u

Обозначения: - = не используется, читается как '0'; u = не изменяется; x = не известно; q = зависит от условий.

Примечания:

1. Один или несколько битов INTCON, PIR1 и/или будут изменены при выходе из режима SLEEP.
2. Если бит GIE=1 при выходе из режима SLEEP, в счетчик команд будет загружен вектор прерываний (0004h).
3. Смотрите в таблице 3-3 состояние битов регистра STATUS.

Таблица 3-4 Состояние регистров специального назначения после сброса (продолжение)

Регистр	Сброс POR или BOR	Сброс -MCLR или WDT	Выход из режима SLEEP по прерыванию или WDT
PORTA	--0x 0000	--uu uuuu	--uu uuuu
PORTB	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTC	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTD	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTE	---- -xxx	---- -uuu	---- -uuu
PORTF	0000 0000	0000 0000	uuuu uuuu
PORTG	0000 0000	0000 0000	uuuu uuuu
PR2	1111 1111	1111 1111	1111 1111
PREFA	0000 0000	0000 0000	uuuu uuuu
PREFB	0000 0000	0000 0000	uuuu uuuu
RCSTA	0000 000x	0000 000x	uuuu uuuu
RCREG	0000 0000	0000 0000	uuuu uuuu
SLPCON	0011 1111	0011 1111	uuuu uuuu
SPBRG	0000 0000	0000 0000	uuuu uuuu
SSPBUF	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSPCON	0000 0000	0000 0000	uuuu uuuu
SSPADD	0000 0000	0000 0000	uuuu uuuu
SSPSTAT	0000 0000	0000 0000	uuuu uuuu
STATUS	0001 1xxx	000q quuu(3)	uuuq quuu(3)
T1CON	--00 0000	--uu uuuu	--uu uuuu
T2CON	-000 0000	-000 0000	-uuu uuuu
TMR0	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1H	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR2	0000 0000	0000 0000	uuuu uuuu
TRIS			
TRISA	--11 1111	--11 1111	--uu uuuu
TRISB	1111 1111	1111 1111	uuuu uuuu
TRISC	1111 1111	1111 1111	uuuu uuuu
TRISD	1111 1111	1111 1111	uuuu uuuu
TRISE	0000 -111	0000 -111	uuuu -uuu
TRISF	1111 1111	1111 1111	uuuu uuuu
TRISG	1111 1111	1111 1111	uuuu uuuu
TXREG	0000 0000	0000 0000	uuuu uuuu
TXSTA	0000 -010	0000 -010	uuuu -uuu
VRCON	000- 0000	000- 0000	uuu- uuuu
W	xxxx xxxx	uuuu uuuu	uuuu uuuu

Обозначения: - = не используется, читается как '0'; u = не изменяется; x = не известно; q = зависит от условий.

Примечания:

1. Один или несколько битов INTCON, PIR1 и/или будут изменены при выходе из режима SLEEP.
2. Если бит GIE=1 при выходе из режима SLEEP, в счетчик команд будет загружен вектор прерываний (0004h).
3. Смотрите в таблице 3-3 состояние битов регистра STATUS.

3.3.1 Регистры PCON и STATUS

Регистр PCON содержит до 4 битов, с помощью которых можно определить источник сброса микроконтроллера:

- Сброс по включению питания (POR);
- Сброс по сигналу на выводе -MCLR;
- Сброс по переполнению сторожевого таймера WDT;
- Сброс по обнаружению снижения напряжения питания (BOR).

Бит -BOR имеет неопределенное значение после сброса POR. Пользователь должен программно установить бит -BOR в ‘1’ и проверять его состояние при возникающих сбросах микроконтроллера. Ели -BOR =0, то произошел сброс по снижению напряжения питания (BOR). Бит -BOR не устанавливается в ‘1’ аппаратно и имеет непредсказуемое значение, если детектор пониженного напряжения питания выключен (BODEN=0 в слове конфигурации).

Бит -POR сбрасывается в ‘0’ при возникновении сброса по включению питания. Пользователь должен программно установить этот бит в ‘1’ после сброса по включению питания. При последующих сбросах, если -POR=0, то произошел сброс по включению питания (или напряжение V_{DD} стало слишком низким).

Бит -PER сбрасывается в ‘0’ при возникновении сброса по ошибке паритета памяти. Пользователь должен программно установить этот бит в ‘1’. Бит -PER сбрасывается в '0' после сброса POR.

MPEEN (разрешение сброса при возникновении ошибки паритета) отображает состояние бита MPEEN в слове конфигурации. Бит MPEEN не изменяется при любом виде сброса или прерывании.

Примечание. При включении питания бит -BOR имеет непредсказуемое значение и не должен учитываться. Бит -BOR предназначен для обнаружения последующих сбросов микроконтроллера при снижении напряжения питания. Состояние бита -BOR также непредсказуемое, если работа детектора пониженного напряжения заблокирована в битах конфигурации при программировании микроконтроллера (BODEN=0).

Регистр PCON

R-u	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
MPEEN	**-**	**-**	**-**	**-**	**-PER**	**-POR**	**-BOR**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **MPEEN**: Бит состояния схемы контроля паритета памяти
Отображает состояние бита MPEEN в слове конфигурации.

биты 6-3:**Не реализованы**: читаются как '0'

бит 2: **-PER**: Флаг сброса по ошибке паритета памяти
1 = сброса по ошибке паритета памяти не было
0 = произошел сброс по ошибке паритета памяти программ микроконтроллера при выборке команды (программно должен быть установлен в '1' после сброса POR или PER)

бит 1: **-POR**: Флаг сброса по включению питания
1 = сброса по включению питания не было
0 = произошел сброс микроконтроллера по включению питания (программно должен быть установлен в '1' для обнаружения сброса POR)

бит 0: **-BOR**: Флаг сброса по снижению напряжения питания
1 = сброса по снижению напряжения питания не было
0 = произошел сброс микроконтроллера по снижению напряжения питания (программно должен быть установлен в '1' для обнаружения сброса BOR)

Примечание. В некоторых микроконтроллерах не все биты реализованы.

Регистр STATUS содержит два бита (-TO и -PD), учитывая которые совместно с битами регистра PCON пользователь может определить причину сброса микроконтроллера.

Регистр STATUS

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	**RP1**	**RP0**	**-TO**	**-PD**	**Z**	**DC**	**C**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **IRP**: Бит выбора банка при косвенной адресации
1 = банк 2, 3 (100h – 1FFh)
0 = банк 0, 1 (000h - 0FFh)

биты 6-5: **RP1:RP0**: Биты выбора банка при непосредственной адресации
11 = банк 3 (180h – 1FFh)
10 = банк 2 (100h – 17Fh)
01 = банк 1 (080h – 0FFh)
00 = банк 0 (000h – 07Fh)

бит 4: **-TO**: Флаг переполнения сторожевого таймера
1 = после POR или выполнения команд CLRWDT, SLEEP
0 = после переполнения WDT

бит 3: **-PD**: Флаг включения питания
1 = после POR или выполнения команды CLRWDT
0 = после выполнения команды SLEEP

бит 2: **Z**: Флаг нулевого результата
1 = нулевой результат выполнения арифметической или логической операции
0 = не нулевой результат выполнения арифметической или логической операции

бит 1: **DC**: Флаг десятичного переноса/заема (для команд ADDWF, ADDWL, SUBWF, SUBWL), заем имеет инверсное значение
1 = был перенос из младшего полубайта
0 = не было переноса из младшего полубайта

бит 0: **C**: Флаг переноса/заема (для команд ADDWF, ADDWL, SUBWF, SUBWL), заем имеет инверсное значение
1 = был перенос из старшего бита
0 = не было переноса из старшего бита

Примечание. Флаг заема имеет инверсное значение. Вычитание выполняется путем прибавления дополнительного кода второго операнда. При выполнении команд сдвига (RRF, RLF) бит C загружается старшим или младшим битом сдвигаемого регистра.

3.4 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Мое устройство подвергнуто воздействию электростатических разрядов и сильным электромагнитным помехам из-за чего работает неустойчиво. Что можно сделать?

Ответ 1:

Если микроконтроллер не имеет интегрированной схемы фильтрации помех на выводе -MCLR (см. приложение С), то необходимо предусмотреть внешнюю схему фильтрации коротких импульсов. Длительность импульса, способного привести к сбросу микроконтроллера, смотрите в разделе "Электрические характеристики" параметр 35.

Вопрос 2: Отладил программу на микроконтроллере с УФ стиранием памяти, сброс и выполнение программы происходило должным образом. При использовании OTP микроконтроллера устройство работает неустойчиво. Что может быть причиной этого?

Ответ 2:

Наиболее частой причиной этого является оставленное незакрытым окно для УФ стирания памяти программ при установке микроконтроллера в устройство. Замечено, что регистры памяти данных, после сброса по включению питания, содержат разные значения при воздействии фонового освещения на кристалл и закрытым окном для УФ стирания памяти. В большинстве случаев регистры специального и общего назначения не были инициализированы должным образом программой микроконтроллера после сброса.

3.5 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные со сбросом микроконтроллеров PICmicro MCU:

Документ	Номер
Power-up Trouble Shooting Решение проблем, возникающих при включении питания	AN607
Power-up Considerations Рекомендации по включению питания	AN522

Раздел 4.Архитектура

Содержание

4.1 Введение

Высокая эффективность микроконтроллеров PICmicro достигается за счет архитектуры ядра, подобная архитектура обычно применяется в RISC микропроцессорах.

Основные особенности архитектуры микроконтроллеров PICmicro:

- Гарвардская архитектура;
- Длинное слово команды;
- Команда состоит из единственного слова;
- Конвейерная обработка команд;
- Команды выполняются за один машинный цикл;
- Небольшое число команд;
- Файловая структура данных;
- Все команды ортогональны (симметричны).

На рисунке 4-2 показана общая структурная схема микроконтроллеров PICmicro среднего семейства.

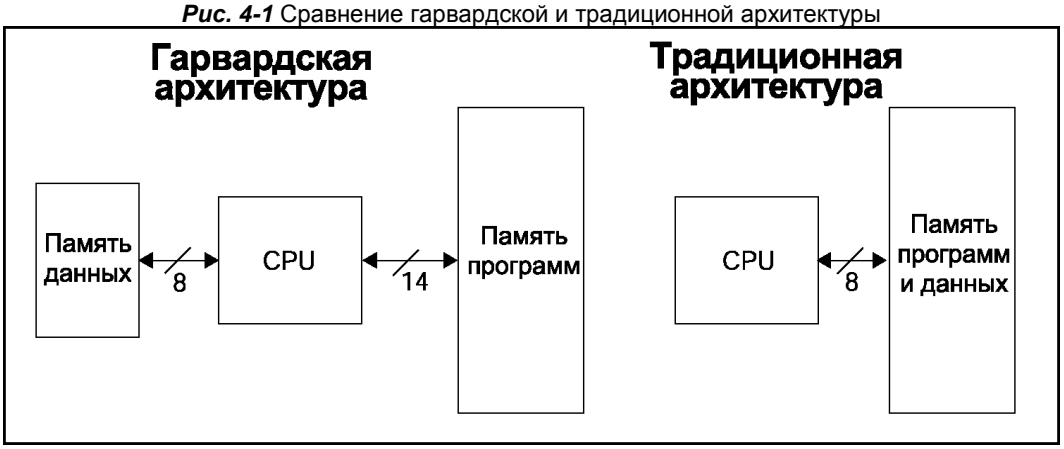
Гарвардская архитектура:

В гарвардской архитектуре разделена память программ и память данных. Обращение к памяти происходит по отдельным шинам адреса и данных, что значительно повышает производительность процессора по сравнению с традиционной архитектурой.

В микроконтроллерах с традиционной архитектурой ядра команды и данные запрашиваются по одной и той же шине. Чтобы выполнить выборку команды необходимо сделать несколько запросов по 8-разрядной (или кратной 8 разрядам) шине. Затем (если необходимо) запросить данные, выполнить команду и сохранить результат. Как может быть замечено шина с традиционной архитектурой ядра значительно загружена.

В микроконтроллерах с гарвардской архитектурой ядра выборка команды происходит за один цикл (все команды 14 - разрядные). При обращении к памяти программ можно выполнить запись или чтение данных, т.к. память данных подключена к ядру микроконтроллера по отдельной шине. Раздельные шины доступа к памяти программ и к памяти данных позволяют исполнять текущую команду и производить выборку следующей команды, организуя конвейерную обработку команд. Сравнение гарвардской и традиционной архитектуры показано на рисунке 4-1.

Рис. 4-1 Сравнение гарвардской и традиционной архитектуры



Длинное слово команды:

Разрядность команд микроконтроллера несколько больше чем 8-разрядная шина памяти данных. Это стало возможным из-за раздельных шин доступа к памяти программ и памяти данных. Разделение шин доступа к разным видам памяти позволяет произвольно выбирать разрядность команд микроконтроллера (не кратное 8-разрядной шине данных), что в свою очередь дает возможность эффективно использовать память программ и оптимизировать разрядность шины программ к архитектурным требованиям микроконтроллера.

Команда состоит из единственного слова:

Все команды микроконтроллеров PIC16CXXX однословные 14 - разрядные. 14 - разрядная шина доступа к памяти программ позволят выполнить выборку 14 - разрядной команды за один машинный цикл микроконтроллера. При использовании однословных команд число слов в памяти программ равняется максимальному числу команд программы микроконтроллера. Это означает, что все ячейки памяти имеют силу команды.

Как правило в традиционной архитектуре большинство команд многобайтные. Микроконтроллер, имеющий 4к байт памяти, содержит примерно 2к команд. Коэффициент использования памяти примерно равен 2:1 и зависит от конкретного приложения. Поскольку каждая команда может состоять из нескольких байтов, то нет никакой гарантии, что каждая ячейка памяти программ имеет силу команды.

Конвейерная обработка команд:

Конвейерная обработка команд состоит из двух стадий: выборка команды из памяти, выполнение команды. Выборка команды происходит в первый машинный цикл T_{CY}, а выполнятся команда во втором машинном цикле T_{CY}. Однако, из-за одновременной выборки текущей команды и выполнения предыдущей в каждом машинном цикле T_{CY} происходит выборка и выполнение команд.

Команды выполняются за один машинный цикл:

Полная выборка команды происходит за один машинный цикл (T_{CY}) из-за того, что шина доступа к памяти программ 14 - разрядная. Каждая команда содержит всю необходимую информацию и выполняется за один машинный цикл. При выполнении команды может возникать задержка в один машинный цикл, если результат команды изменяет содержимое счетчика команд PC. Задержка в один машинный цикл необходима для выборки новой команды, которая должна быть выполнена следующей.

Небольшое число команд:

Когда система команд хорошо проработана и команды ортогональны (симметричны), то требуется меньшее число команд для решения всех необходимых задач. С меньшим числом команд изучение микроконтроллера значительно упрощается.

Файловая структура данных:

Обращение к регистрам памяти данных можно выполнить прямой или косвенной адресацией. Все регистры специального назначения, включая счетчик команд PC, отображается в памяти данных.

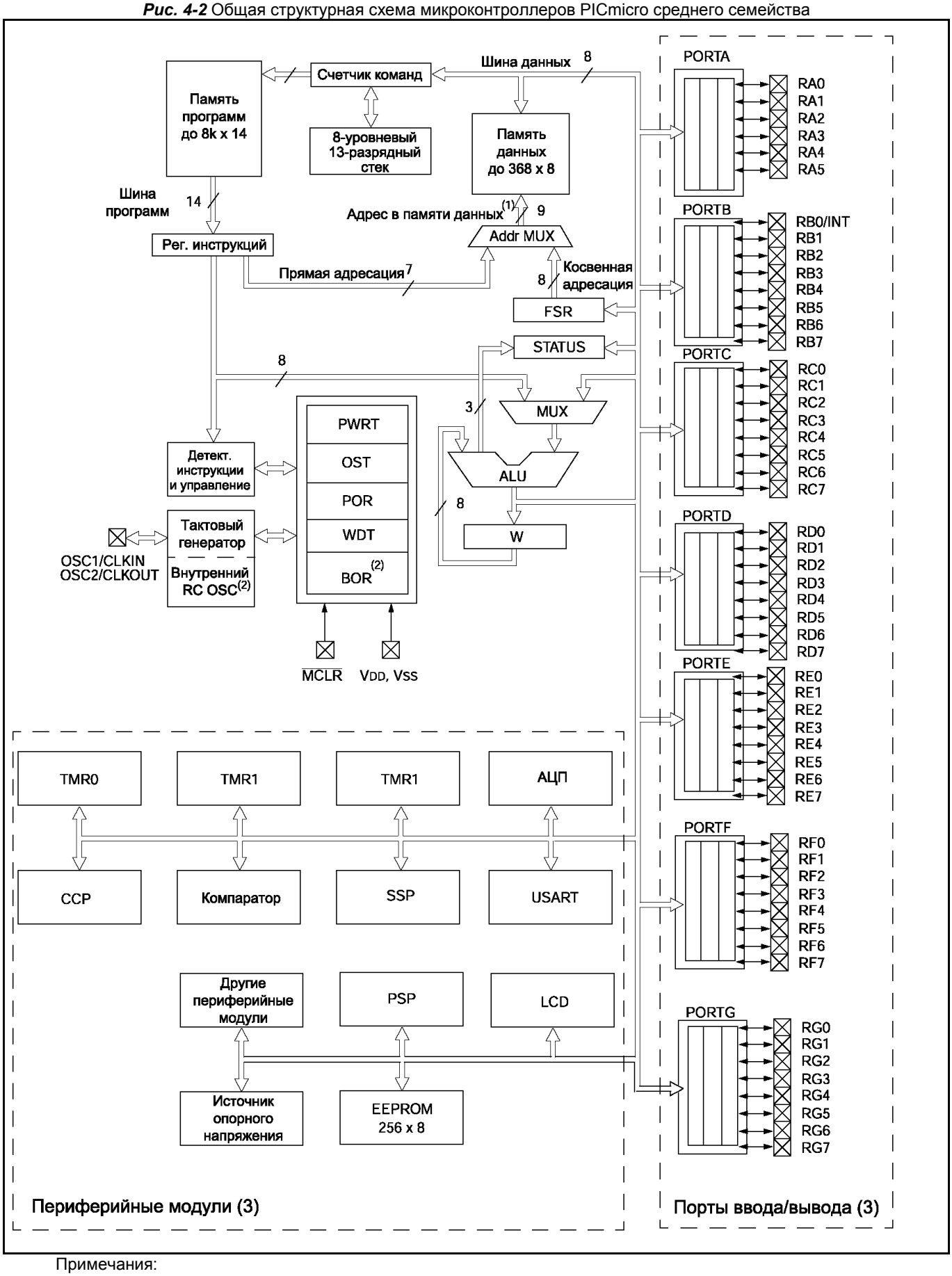
Все команды ортогональны (симметричны):

Ортогональная система команд дает возможность выполнить любую операцию с любым регистром памяти данных прямой или косвенной адресацией. В ортогональной системе команд малое количество "специальных команд", что упрощает изучение и программирование микроконтроллеров не теряя эффективности кода программы. В микроконтроллерах среднего семейства используется только две не ортогональные команды, реализующие особенности ядра.

Команда SLEEP - переводит микроконтроллер в режим пониженного энергопотребления.

Команда CLRWDT - подтверждает нормальную работу микроконтроллера, предотвращая сброс по переполнению сторожевого таймера WDT.

Рис. 4-2 Общая структурная схема микроконтроллеров PICmicro среднего семейства



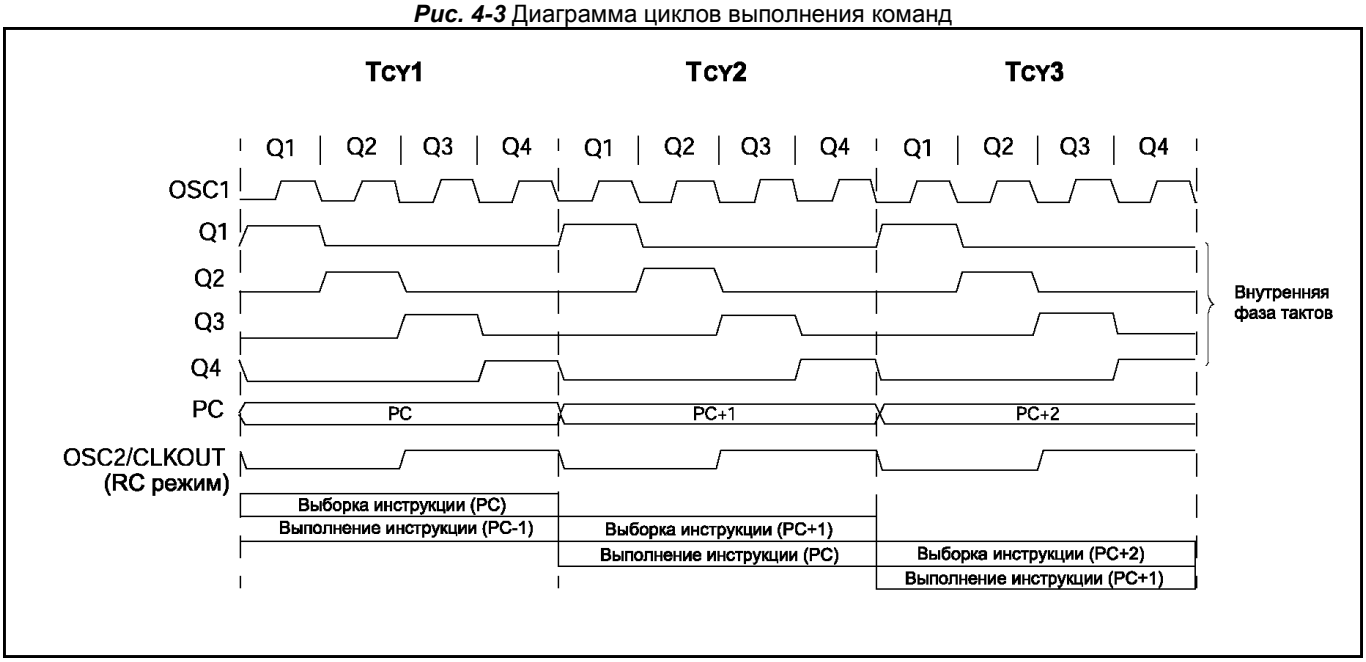
Примечания:

1. Старшие биты адреса при обращении к ОЗУ прямой адресацией из регистра STATUS.
2. Не все микроконтроллеры имеют эти особенности, смотрите техническую документацию на соответствующий микроконтроллер.
3. Большинство универсальных каналов портов ввода/вывода мультиплицированы с одним или более периферийным модулем. Смотрите техническую документацию на соответствующий микроконтроллер.

4.2 Синхронизация выполнения команд

Входной тактовый сигнал (вывод OSC1) внутренней схемой микроконтроллера разделяется на четыре последовательных неперекрывающихся такта Q1, Q2, Q3 и Q4. Внутренний счетчик команд (PC) увеличивается на единицу в каждом такте Q1, а выборка команды из памяти программ происходит на каждом такте Q4. Декодирование и выполнение команды происходит с такта Q1 по Q4. На рисунке 4-3 и в примере 4-1 показаны циклы выполнения команд.

Рис. 4-3 Диаграмма циклов выполнения команд



4.3 Конвейерная выборка и выполнение команд

Цикл выполнения команды состоит из четырех тактов Q1, Q2, Q3 и Q4. Выборка следующей команды и выполнение текущей совмещены по времени, таким образом, выполнение команды происходит за один цикл. Если команда изменяет счетчик команд PC (команды ветвления, например GOTO), то необходимо два машинных цикла для выполнения команды (см. пример 4-1).

Цикл выборки команды начинается с приращения счетчика команд PC в такте Q1.

В цикле выполнения команды, код загруженной команды, помещается в регистр команд IR на такте Q1. Декодирование и выполнение команды происходит в тактах Q2, Q3 и Q4. Операнд из памяти данных читается в такте Q2, а результат выполнения команды записывается в такте Q4.

В примере 4-1 показаны две стадии конвейерной обработки команд для представленной последовательности. В цикле $T_{CY}0$ происходит выборка первой команды из памяти программ. На цикле $T_{CY}1$ первая команда исполняется, а вторая команда выбирается из памяти программ. В течение цикла $T_{CY}2$ вторая команда исполняется, а третья выбирается из памяти программ. На цикле $T_{CY}3$ происходит выборка четвертой команды и выполняется команда третья команда (CALL SUB_1). Когда завершается выполнение третьей команды CPU загружает адрес четвертой команды в вершину стека и изменяет счетчик команд PC на адрес SUB_1. Это означает, что команда, загруженная в цикле $T_{CY}3$, должна быть удалена из конвейера. В течение цикла $T_{CY}4$ четвертая команда удаляется из конвейера (выполняется пустой цикл NOP) и происходит выборка команды по адресу SUB_1. В цикле $T_{CY}5$ выполняется команда пять и выбирается из памяти программ команда с адресом SUB_1 + 1.

Пример 4-1 Выборка и выполнения команд

Все команды выполняются за один цикл, кроме команд ветвления. Команды ветвления требуют два машинных цикла, т.к. необходимо удалить предварительно выбранную команду из конвейера. Во время удаления выбирается новая команда, а затем она исполняется в следующем машинном цикле.

4.4 Описание портов ввода/вывода

В таблице 4-1 дано краткое описание функций, которые могут быть мультиплицированы к каналам портов ввода/вывода. Возможна ситуация, когда на один вывод мультиплицируется несколько функций. При использовании вывода периферийным модулем действие битов регистра TRIS может быть заблокировано (например для АЦП или LCD модуля).

Таблица 4-1 Описание выводов

Имя вывода	Тип вывода	Тип буфера	Описание
			Аналоговые входы.
AN0	I	Аналоговый	
AN1	I	Аналоговый	
AN2	I	Аналоговый	
AN3	I	Аналоговый	
AN4	I	Аналоговый	
AN5	I	Аналоговый	
AN6	I	Аналоговый	
AN7	I	Аналоговый	
AN8	I	Аналоговый	
AN9	I	Аналоговый	
AN10	I	Аналоговый	
AN11	I	Аналоговый	
AN12	I	Аналоговый	
AN13	I	Аналоговый	
AN14	I	Аналоговый	
AN15	I	Аналоговый	
A_{VDD}	P	P	Аналоговое питание.
A_{VSS}	P	P	Аналоговый общий.
C1	I	Аналоговый	Напряжение LCD.
C2	I	Аналоговый	Напряжение LCD.
CCP1	I/O	ST	Вывод Захват/Сравнение/ШИМ модуля CCP1.
CCP2	I/O	ST	Вывод Захват/Сравнение/ШИМ модуля CCP2.
CDAC	O	Аналоговый	Токовый вывод АЦП. Обычно используется для подключения внешнего конденсатора, чтобы формировать линейное увеличение напряжения.
CK	I/O	ST	Тактовый сигнал USART в синхронном режиме. Всегда связан с функциями вывода TX (см. TX, RX, DT).
CLKIN	I	ST/CMOS	Вход внешнего тактового сигнала. Всегда связан с функциями вывода OSC1 (см. OSC1/CLKIN, OSC2/CLKOUT).
CLKOUT	O	-	Вывод тактового генератора. Подключается кварцевый/керамический резонатор в HS,XT, LP режиме генератора. В RC режиме генератора на выводе OSC2 присутствует сигнал CLKOUT с 1/4 частоты OSC1, равной циклам выполнения команд. Всегда связан с функциями вывода OSC2 (см. OSC1, OSC2).
CMPA	O	-	Выход компаратора A.
CMPB	O	-	Выход компаратора B.
COM0	L	-	Общий драйвер 0 LCD.
COM1	L	-	Общий драйвер 1 LCD.
COM2	L	-	Общий драйвер 2 LCD.
COM3	L	-	Общий драйвер 3 LCD.
-CS	I	TTL	Вход выбора микросхемы ведомого параллельного порта (см. -RD,-WR)
DT	I/O	ST	Сигнал данных USART в синхронном режиме. Всегда связан с функциями вывода RX (см. TX, RX, CK).
			GP - двунаправленный порт ввода/вывода. На некоторых входах порта могут быть программно включены внутренние подтягивающие резисторы.
GP0	I/O	TTL/ST	TTL буфер в режиме порта ввода/вывода. Буфер с триггером Шмидта в режиме последовательного программирования.
GP1	I/O	TTL/ST	TTL буфер в режиме порта ввода/вывода. Буфер с триггером Шмидта в режиме последовательного программирования.
GP2	I/O	ST	
GP3	I	TTL	
GP4	I/O	TTL	
GP5	I/O	TTL	

Обозначения:

TTL = входной буфер TTL
CMOS = КМОП совместимый вход или выход
NO-PD = нет внутр. диода, подключ. к V_{DD}
I = вход
L = драйвер LCD
ST = входной буфер с триггером Шмидта
NPU = N-канальный подтягивающий элемент
PU = внутренний подтягивающий элемент
O = выход
P = питание
SM = соответствие спецификации SMBus. Требуется внешний подтягивающий резистор

Таблица 4-1 Описание выводов (продолжение)

Имя вывода	Тип вывода	Тип буфера	Описание
INT	I	ST	Вход внешних прерываний.
-MCLR/V_{PP}	I/P	ST	Вход сброса микроконтроллера (активный низкий логический уровень) или вход напряжения программирования.
NC	-	-	Эти выводы должны быть оставлены не подключенными.
OSC1	I	ST/CMOS	Вход тактового генератора или вход внешнего тактового сигнала. Входной буфер с триггером Шмидта в RC режиме генератора. КМОП буфер в остальных режимах.
OSC2	O	-	Вывод тактового генератора. Подключается кварцевый/керамический резонатор в HS,XT, LP режиме генератора. В RC режиме генератора на выводе OSC2 присутствует сигнал CLKOUT с 1/4 частоты OSC1, равной циклам выполнения команд.
PBTN	I	ST	Вход с внутренним подтягивающим резистором. Может использоваться для генерации прерываний.
			Ведомый параллельный порт для связи с микропроцессором. Выводы имеют входной буфер TTL, когда модуль PSP включен.
PSP0	I/O	TTL	
PSP1	I/O	TTL	
PSP2	I/O	TTL	
PSP3	I/O	TTL	
PSP4	I/O	TTL	
PSP5	I/O	TTL	
PSP6	I/O	TTL	
PSP7	I/O	TTL	
			PORTA - двунаправленный порт ввода/вывода.
RA0	I/O	TTL	
RA1	I/O	TTL	
RA2	I/O	TTL	
RA3	I/O	TTL	
RA4	I/O	ST	RA4 имеет выход с открытым стоком.
RA5	I/O	TTL	
			PORTB - двунаправленный порт ввода/вывода. На входах порта могут быть программно включены внутренние подтягивающие резисторы.
RB0	I/O	TTL	
RB1	I/O	TTL	
RB2	I/O	TTL	
RB3	I/O	TTL	
RB4	I/O	TTL	Прерывание по изменению уровня сигнала на входе.
RB5	I/O	TTL	Прерывание по изменению уровня сигнала на входе.
RB6	I/O	TTL/ST	Прерывание по изменению уровня сигнала на входе. Тактовый вход в режиме программирования. Буфер ТТЛ в нормальном режиме. Буфер с триггером Шмидта в режиме программирования.
RB7	I/O	TTL/ST	Прерывание по изменению уровня сигнала на входе. Вывод данных в режиме программирования. Буфер ТТЛ в нормальном режиме. Буфер с триггером Шмидта в режиме программирования.
			PORTC - двунаправленный порт ввода/вывода.
RC0	I/O	ST	
RC1	I/O	ST	
RC2	I/O	ST	
RC3	I/O	ST	
RC4	I/O	ST	
RC5	I/O	ST	
RC6	I/O	ST	
RC7	I/O	ST	
-RD	I	TTL	Управление чтением ведомого параллельного порта (см. -WR, -CS).

Обозначения:

TTL = входной буфер TTL
CMOS = КМОП совместимый вход или выход
NO-PD = нет внутр. диода, подключ. к V_{DD}
I = вход
L = драйвер LCD
ST = входной буфер с триггером Шмидта
NPU = N-канальный подтягивающий элемент
PU = внутренний подтягивающий элемент
O = выход
P = питание
SM = соответствие спецификации SMBus. Требуется внешний подтягивающий резистор

Таблица 4-1 Описание выводов (продолжение)

Имя вывода	Тип вывода	Тип буфера	Описание
			PORTD - двунаправленный порт ввода/вывода.
RD0	I/O	ST	
RD1	I/O	ST	
RD2	I/O	ST	
RD3	I/O	ST	
RD4	I/O	ST	
RD5	I/O	ST	
RD6	I/O	ST	
RD7	I/O	ST	
			PORTE - двунаправленный порт ввода/вывода.
RE0	I/O	ST	
RE1	I/O	ST	
RE2	I/O	ST	
RE3	I/O	ST	
RE4	I/O	ST	
RE5	I/O	ST	
RE6	I/O	ST	
RE7	I/O	ST	
REFA	O	CMOS	Выход A программируемого источника опорного напряжения.
REFB	O	CMOS	Выход B программируемого источника опорного напряжения.
			PORTF - цифровые входы или порт драйвера сегментов LCD.
RF0	I/O	ST	
RF1	I/O	ST	
RF2	I/O	ST	
RF3	I/O	ST	
RF4	I/O	ST	
RF5	I/O	ST	
RF6	I/O	ST	
RF7	I/O	ST	
			PORTG - цифровые входы или порт драйвера сегментов LCD.
RG0	I/O	ST	
RG1	I/O	ST	
RG2	I/O	ST	
RG3	I/O	ST	
RG4	I/O	ST	
RG5	I/O	ST	
RG6	I/O	ST	
RG7	I/O	ST	
RX	I	ST	Вывод приемника в асинхронном режиме USART.
SCL	I/O	ST	Вывод тактового сигнала в режиме I^2C.
SCLA	I/O	ST	Вывод тактового сигнала интерфейса I^2C.
SCLB	I/O	ST	Вывод тактового сигнала интерфейса I^2C.
SDA	I/O	ST	Вывод данных в режиме I^2C.
SDAA	I/O	ST	Вывод данных интерфейса I^2C.
SDAB	I/O	ST	Вывод данных интерфейса I^2C.
SCK	I/O	ST	Вход/выход тактового сигнала в режиме SPI.
SDI	I		Вывод приемника SPI.
SDO	O		Вывод передатчика SPI.
-SS	I		Вход выбора ведомого SPI.
SEG00:SEG31	I/L	ST	Драйверы сегментов LCD от 00 до 31.
SUM	O	AN	AN1 подтверждение перехода. К выводу может быть подключен внешний конденсатор
T0CKI	I	ST	Вход внешнего тактового сигнала для TMR0.
T1CKI	I	ST	Вход внешнего тактового сигнала для TMR1.
T1OSO	O	CMOS	Выход генератора TMR1.
T1OSI	I	CMOS	Вход генератора TMR1.
TX	O	-	Выход передатчика USART в асинхронном режиме (см. RX).

Обозначения:

TTL = входной буфер TTL
CMOS = КМОП совместимый вход или выход
NO-PD = нет внутр. диода, подключ. к V_{DD}
I = вход
L = драйвер LCD
ST = входной буфер с триггером Шмидта
NPU = N-канальный подтягивающий элемент
PU = внутренний подтягивающий элемент
O = выход
P = питание
SM = соответствие спецификации SMBus. Требуется внешний подтягивающий резистор

Таблица 4-1 Описание выводов (продолжение)

Имя вывода	Тип вывода	Тип буфера	Описание
$V_{LCD}1$	P	-	Напряжение LCD.
$V_{LCD}2$	P	-	Напряжение LCD.
$V_{LCD}3$	P	-	Напряжение LCD.
VLCDADJ	I	Аналоговый	Напряжение LCD.
V_{REF}	I	Аналоговый	Вход верхнего опорного напряжения. Вывод входа опорного напряжения присутствует на микроконтроллерах с компараторами.
V_{REF+}	I	Аналоговый	Вход верхнего опорного напряжения. Обычно мультиплицируется на аналоговый вход.
V_{REF-}	I	Аналоговый	Вход нижнего опорного напряжения. Обычно мультиплицируется на аналоговый вход.
VREG	O	-	Вывод управления внешним компонентом N-FET для регулировки напряжения.
V_{SS}	P	-	Общий вывод для внутренней логики и портов ввода/вывода.
V_{DD}	P	-	Положительное напряжение питания для внутренней логики и портов ввода/вывода.
-WR			Управление записью в ведомый параллельный порт (см. -RD, -CS).

Обозначения:

TTL = входной буфер TTL
CMOS = КМОП совместимый вход или выход
NO-PD = нет внутр. диода, подключ. к V_{DD}
I = вход
L = драйвер LCD
ST = входной буфер с триггером Шмидта
NPU = N-канальный подтягивающий элемент
PU = внутренний подтягивающий элемент
O = выход
P = питание
SM = соответствие спецификации SMBus. Требуется внешний подтягивающий резистор

4.5 Ответы на часто задаваемые вопросы

На момент выполнения перевода в оригинальной технической документации вопросы отсутствовали. Если у вас есть вопрос, задайте его, написав нам письмо по адресу support@microchip.ru.

4.6 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные архитектурой микроконтроллеров PICmicro MCU:

Документ	Номер

В настоящее время документы не подготовлены.

Раздел 5.ЦПУ и АЛУ

Содержание

5.1 Введение

Центральное Процессорное Устройство (ЦПУ) предназначено для детектирования команд, расположенных в памяти программ, и управления работой микроконтроллера. Большинство команд микроконтроллера обращаются к ячейкам памяти данных. Для работы с памятью данных требуется арифметико-логическое устройство (АЛУ). АЛУ выполняет арифметические, логические операции и управляет флагами состояния (флаги состояния расположены в регистре STATUS). Выполнение некоторых команд приводит к изменению битов состояния в зависимости от полученного результата.

В таблице 5-1 сведены коды команд, поддерживаемые ЦПУ (также представлена мнемоника команд для MPASM, чтобы генерировать эти коды).

Таблица 5-1 Список команд микроконтроллеров среднего семейства

Мнемоника команды		Описание	Циклов	14-разрядный код		Изм. флаги	Прим.
				Бит 13	Бит 0		
Байт ориентированные команды							
ADDWF	**f,d**	Сложение W и f	1	00 0111 dfff ffff		C,DC,Z	1,2
ANDWF	**f,d**	Побитное 'И' W и f	1	00 0101 dfff ffff		Z	1,2
CLRF	**f**	Очистить f	1	00 0001 1fff ffff		Z	2
CLRW	**-**	Очистить W	1	00 0001 0xxx xxxx		Z	
COMF	**f,d**	Инвертировать f	1	00 1001 dfff ffff		Z	1,2
DECF	**f,d**	Вычесть 1 из f	1	00 0011 dfff ffff		Z	1,2
DECFSZ	**f,d**	Вычесть 1 из f и пропустить если 0	1(2)	00 1011 dfff ffff			1,2,3
INCF	**f,d**	Прибавить 1 к f	1	00 1010 dfff ffff		Z	1,2
INCFSZ	**f,d**	Прибавить 1 к f и пропустить если 0	1(2)	00 1111 dfff ffff			1,2,3
IORWF	**f,d**	Побитное 'ИЛИ' W и f	1	00 0100 dfff ffff		Z	1,2
MOVF	**f,d**	Переслать f	1	00 1000 dfff ffff		Z	1,2
MOVWF	**f**	Переслать W в f	1	00 0000 1fff ffff			
NOP	**-**	Нет операции	1	00 0000 0xx0 0000			
RLF	**f,d**	Циклический сдвиг f влево через перенос	1	00 1101 dfff ffff		C	1,2
RRF	**f,d**	Циклический сдвиг f вправо через перенос	1	00 1100 dfff ffff		C	1,2
SUBWF	**f,d**	Вычесть W из f	1	00 0010 dfff ffff		C,DC,Z	1,2
SWAPF	**f,d**	Поменять местами полубайты в регистре f	1	00 1110 dfff ffff			1,2
XORWF	**f,d**	Побитное 'исключающее ИЛИ' W и f	1	00 0110 dfff ffff		Z	1,2
Бит ориентированные команды							
BCF	**f,b**	Очистить бит b в регистре f	1	01 00bb bfff ffff			1,2
BSF	**f,b**	Установить бит b в регистре f	1	01 01bb bfff ffff			1,2
BTFSC	**f,b**	Проверить бит b в регистре f, пропустить если 0	1(2)	01 10bb bfff ffff			3
BTFSS	**f,b**	Проверить бит b в регистре f, пропустить если 1	1(2)	01 11bb bfff ffff			3
Команды управления и операций с константами							
ADDLW	**k**	Сложить константу с W	1	11 111x kkkk kkkk		C,DC,Z	
ANDLW	**k**	Побитное 'И' константы и W	1	11 1001 kkkk kkkk		Z	
CALL	**k**	Вызов подпрограммы	2	10 0kkk kkkk kkkk			
CLRWDT	**-**	Очистить WDT	1	00 0000 0110 0100		-TO,-PD	
GOTO	**k**	Безусловный переход	2	10 1kkk kkkk kkkk			
IORLW	**k**	Побитное 'ИЛИ' константы и W	1	11 1000 kkkk kkkk		Z	
MOVLW	**k**	Переслать константу в W	1	11 00xx kkkk kkkk			
RETFIE	**-**	Возврат из подпрограммы с разрешением прерываний	2	00 0000 0000 1001			
RETLW	**k**	Возврат из подпрограммы с загрузкой константы в W	2	11 01xx kkkk kkkk			
RETURN	**-**	Возврат из подпрограммы	2	00 0000 0000 1000			
SLEEP	**-**	Перейти в режим SLEEP	1	00 0000 0110 0011		-TO,-PD	
SUBLW	**k**	Вычесть W из константы	1	11 110x kkkk kkkk		C,DC,Z	
XORLW	**k**	Побитное 'исключающее ИЛИ' константы и W	1	11 1010 kkkk kkkk		Z	

Примечания:

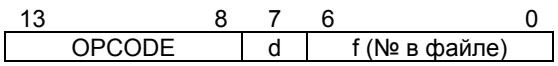
1. При выполнении операции "чтение - модификация - запись" с портом ввода/вывода (например MOVF PORTB,1) исходные значения считываются с выводов порта, а не из выходных защелок. Например, если в выходной защелке было записана '1', а на соответствующем выходе низкий уровень сигнала, то обратно будет записано значение '0'.
2. При выполнении записи в TMR0 (и d=1) предделитель TMR0 сбрасывается, если он подключен к модулю TMR0.
3. Если условие истинно или изменяется значение счетчика команд PC, то инструкция выполняется за два цикла. Во втором цикле выполняется команда NOP.

5.2 Общий формат команд микроконтроллеров среднего семейства

Все команды микроконтроллеров среднего семейства могут быть разделены на четыре основных группы (см. рисунок 5-1). Код операции команды изменяется от 3 бит до 6 бит, что позволяет реализовать 35 команд.

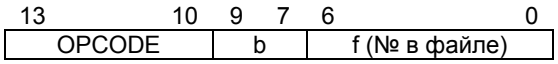
Рис. 5-1 Общий формат команд микроконтроллеров среднего семейства

Байт ориентированные операции с регистрами



d = 0 - результат сохраняется в w
d = 1 - результат сохраняется в f
f - 7-разрядный адрес регистра

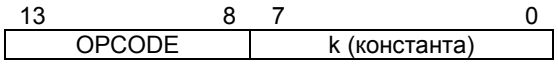
Бит ориентированные операции с регистрами



b - 3-разрядный номер бита в регистре
f - 7-разрядный адрес регистра

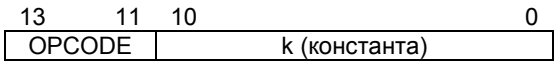
Команды управления и операций с константами

Общее



k - 8-разрядное значение

Только для инструкций CALL и GOTO



k - 11-разрядное значение

5.3 Центральное Процессорное Устройство (ЦПУ)

ЦПУ можно рассматривать как "мозги" микроконтроллера. ЦПУ отвечает за выборку команды из памяти программ, ее детектирование и выполнение. Иногда ЦПУ работает совместно с АЛУ, чтобы выполнить арифметические или логические операции. ЦПУ управляет шиной адреса памяти программ и памяти данных, а также обращением к стеку.

5.4 Такты выполнения команд

Каждый цикл команды (T_{CY}) состоит из четырех тактов (Q1-Q4). Такт Q равен по длительности периоду тактового генератора (T_{OSC}). Такты Q обеспечивают жесткую синхронизацию декодирования, чтения данных, обработки данных, записи результата для каждого цикла команды. На диаграмме показано соотношение тактов Q к циклу команды.

Цикл команды (T_{CY}), состоящий из 4-х тактов, обобщенно выглядит следующим образом:

Q1: Детектирование команды или принудительной пустой операции (NOP)
Q2: Операция чтения данных или отсутствие операции
Q3: Обработка данных
Q4: Операция записи данных или отсутствие операции

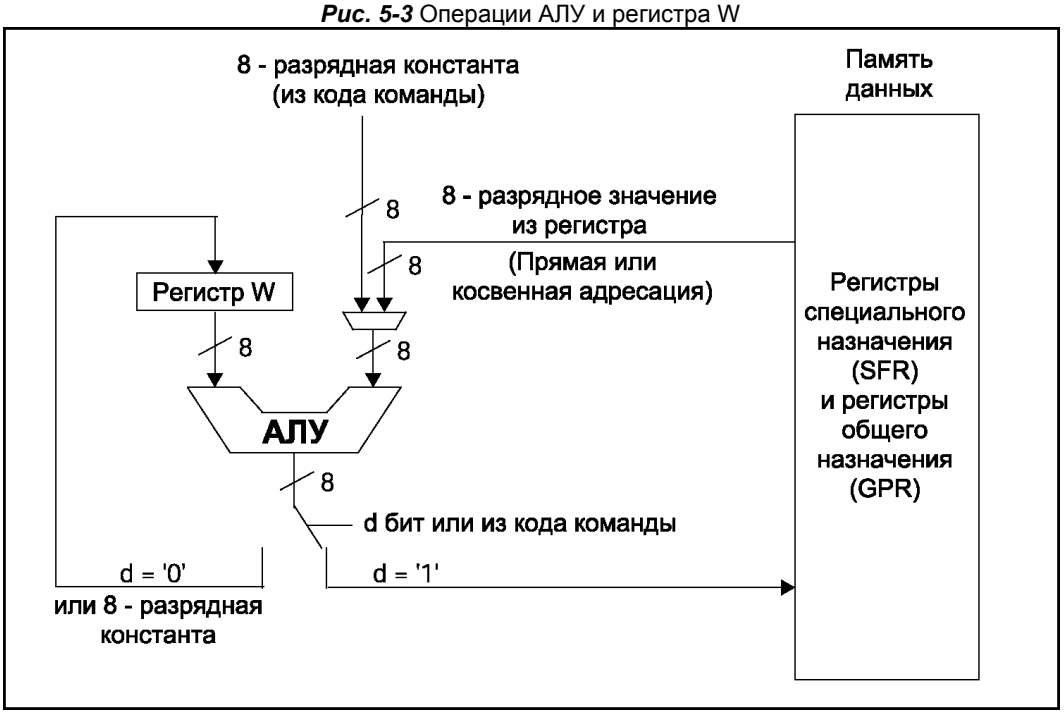
Детальный состав операций команды по тактам Q смотрите в описании команды.

Рис. 5-2 Временная диаграмма циклического повторения тактов Q

5.5 Арифметико-логическое Устройство (АЛУ)

Микроконтроллеры PICmicro MCU содержат 8-разрядный универсальный арифметический модуль (АЛУ) и 8 - разрядный рабочий регистр (W). АЛУ выполняет арифметические и булевы операции между рабочим регистром и любым регистром памяти данных.

Рис. 5-3 Операции АЛУ и регистра W



8 - разрядное АЛУ может выполнять сложение, вычитание, поразрядный сдвиг и логические операции. Арифметические операции выполняются по принципу дополнения до двух, если не указано иначе. В командах с двумя операндами: первый операнд находится в рабочем регистре W, а второй операнд расположен в регистре памяти данных или константа. В командах с одним операндом: операндом является регистр W или регистр памяти данных.

Регистр W - не адресуемый 8-разрядный рабочий регистр, который используется в операциях АЛУ. В зависимости от типа команды и результат команды АЛУ может воздействовать на следующие флаги состояния в регистре STATUS: перенос (C), полуперенос (DC), флаг нулевого результата (Z). Биты C и DC работают как биты заема и десятичного заема при выполнении команд вычитания. Примеры смотрите в описании команд SUBWF и SUBLW.

5.6 Регистр STATUS

В регистре STATUS содержатся флаги состояния АЛУ, флаги причины сброса микроконтроллера и биты управления банками памяти данных. Поскольку в регистре STATUS присутствуют биты управления банками памяти необходимо, чтобы он отображался во всех банках памяти данных и имел одинаковое смещение относительно начала банка (см. рисунок 6-5 в разделе "Организация памяти").

Регистр STATUS может быть адресован любой командой, как и любой другой регистр памяти данных. Если обращение к регистру STATUS выполняется командой, которая воздействует на флаги Z, DC и C, то изменение этих трех битов командой заблокировано. Эти биты сбрасываются или устанавливаются согласно логике ядра микроконтроллера. Команды изменения регистра STATUS также не воздействуют на биты -TO и -PD. Поэтому результат выполнения команды с регистром STATUS может отличаться от ожидаемого. Например, команда CLRF STATUS сбросит три старших бита и установит бит Z (состояние регистра STATUS после выполнения команды 000uu1uu, где u - не изменяемый бит).

При изменении битов регистра STATUS рекомендуется использовать команды, не влияющие на флаги АЛУ (SWAPF, MOVWF, BCF и BSF). Список команд, не воздействующих на флаги АЛУ, смотрите в таблице 5-1.

Примечание 1. Некоторые микроконтроллеры не требуют битов IRP и RP1 (STATUS<7:6>). В этом случае биты IRP и RP1 не используются ЦПУ и АЛУ. Не рекомендуется использовать эти биты как биты общего назначения для совместимости программы с другими микроконтроллерами (при записи в регистр STATUS биты IRP и RP1 должны равняться '0').

Примечание 2. Флаги C и DC используются как биты заема и десятичного заема соответственно, например, при выполнении команд вычитания SUBLW и SUBWF.

Регистр STATUS

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	**RP1**	**RP0**	**-TO**	**-PD**	**Z**	**DC**	**C**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **IRP**: Бит выбора банка при косвенной адресации
1 = банк 2, 3 (100h – 1FFh)
0 = банк 0, 1 (000h - 0FFh)

биты 6-5: **RP1:RP0**: Биты выбора банка при непосредственной адресации
11 = банк 3 (180h – 1FFh)
10 = банк 2 (100h – 17Fh)
01 = банк 1 (080h – 0FFh)
00 = банк 0 (000h – 07Fh)

бит 4: **-TO**: Флаг переполнения сторожевого таймера
1 = после POR или выполнения команд CLRWDT, SLEEP
0 = после переполнения WDT

бит 3: **-PD**: Флаг включения питания
1 = после POR или выполнения команды CLRWDT
0 = после выполнения команды SLEEP

бит 2: **Z**: Флаг нулевого результата
1 = нулевой результат выполнения арифметической или логической операции
0 = не нулевой результат выполнения арифметической или логической операции

бит 1: **DC**: Флаг десятичного переноса/заема (для команд ADDWF, ADDWL, SUBWF, SUBWL), заем имеет инверсное значение
1 = был перенос из младшего полубайта
0 = не было переноса из младшего полубайта

бит 0: **C**: Флаг переноса/заема (для команд ADDWF, ADDWL, SUBWF, SUBWL), заем имеет инверсное значение
1 = был перенос из старшего бита
0 = не было переноса из старшего бита

Примечание. Флаг заема имеет инверсное значение. Вычитание выполняется путем прибавления дополнительного кода второго операнда. При выполнении команд сдвига (RRF, RLF) бит C загружается старшим или младшим битом сдвигаемого регистра.

5.7 Регистр OPTION_REG

Регистр OPTION_REG доступен для чтения и записи, содержит биты управления:

- Предварительным делителем TMR0/WDT;
- Активным фронтом внешнего прерывания RB0/INT;
- Подтягивающими резисторами на входах PORTB.

Регистр OPTION_REG

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
-RBPU	**INTEDG**	**T0CS**	**T0SE**	**PSA**	**PS2**	**PS1**	**PS0**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **-RBPU**: Включение подтягивающих резисторов на входах PORTB
1 = подтягивающие резисторы отключены
0 = подтягивающие резисторы включены

бит 6: **INTEDG**: Выбор активного фронта сигнала на входе внешнего прерывания INT
1 = прерывания по переднему фронту сигнала
0 = прерывания по заднему фронту сигнала

бит 5: **T0CS**: Выбор тактового сигнала для TMR0
1 = внешний тактовый сигнал с вывода RA4/T0CKI
0 = внутренний тактовый сигнал CLKOUT

бит 4: **T0SE**: Выбор фронта приращения TMR0 при внешнем тактовом сигнале
1 = приращение по заднему фронту сигнала (с высокого к низкому уровню) на выводе RA4/T0CKI
0 = приращение по переднему фронту сигнала (с низкого к высокому уровню) на выводе RA4/T0CKI

бит 3: **PSA**: Выбор включения предделителя
1 = предделитель включен перед WDT
0 = предделитель включен перед TMR0

биты 2-0: **PS2: PS0**: Установка коэффициента деления предделителя

Значение	Для TMR0	Для WDT
000	1:2	1:1
001	1:4	1:2
010	1:8	1:4
011	1:16	1:8
100	1:32	1:16
101	1:64	1:32
110	1:128	1:64
111	1:256	1:128

Примечание. При использовании режима низковольтного программирования и включенных подтягивающих резисторах на PORTB необходимо сбросить в '0' 3-й бит регистра TRISB для выключения подтягивающего резистора на выводе RB3.

Примечание. Если предварительный делитель включен перед WDT, то коэффициент деления тактового сигнала для TMR0 равен 1:1.

5.8 Регистр PCON

Регистр PCON содержит до 4 дополнительных битов (к битам -TO, - PD регистра STATUS), с помощью которых можно определить источник сброса микроконтроллера:

Примечание 1. При включении питания бит -BOR имеет непредсказуемое значение и не должен учитываться. Бит -BOR предназначен для обнаружения последующих сбросов микроконтроллера при снижении напряжения питания. Состояние бита -BOR также непредсказуемое, если работа детектора пониженного напряжения заблокирована в битах конфигурации при программировании микроконтроллера (BODEN=0).

Примечание 2. Пользователь должен программно установить бит -POR в ‘1’ после сброса по включению питания. При последующих сбросах, если -POR=0, то произошел сброс по включению питания (или напряжение V_{DD} стало слишком низким).

Регистр PCON

R-u	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
MPEEN	**-**	**-**	**-**	**-**	**-PER**	**-POR**	**-BOR**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **MPEEN**: Бит состояния схемы контроля паритета памяти
Отображает состояние бита MPEEN в слове конфигурации.

биты 6-3:**Не реализованы**: читаются как '0'

бит 2: **-PER**: Флаг сброса по ошибке паритета памяти
1 = сброса по ошибке паритета памяти не было
0 = произошел сброс по ошибке паритета памяти программ микроконтроллера при выборке команды (программно должен быть установлен в '1' после сброса POR или PER)

бит 1: **-POR**: Флаг сброса по включению питания
1 = сброса по включению питания не было
0 = произошел сброс микроконтроллера по включению питания (программно должен быть установлен в '1' для обнаружения сброса POR)

бит 0: **-BOR**: Флаг сброса по снижению напряжения питания
1 = сброса по снижению напряжения питания не было
0 = произошел сброс микроконтроллера по снижению напряжения питания (программно должен быть установлен в '1' для обнаружения сброса BOR)

5.9 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Алгоритм программы работает неправильно.
Ответ 1:

1. Ошибочно адресатом результата выполнения команды может быть регистр W (d=0) вместо регистра памяти данных (d=1).
2. Неправильно настраиваются биты выбора банка памяти (RP1:RP0 или IRP) в регистре STATUS. Если используются прерывания, то возможно неправильно восстанавливаются биты регистра STATUS при выходе из обработки прерываний.

Вопрос 2: Не могу изменить флаги регистра STATUS.
Ответ 2:

Если обращение к регистру STATUS выполняется командой, которая воздействует на флаги Z, DC и C, то изменение этих трех битов командой заблокировано. Эти биты сбрасываются или устанавливаются согласно логике ядра микроконтроллера. Поэтому, чтобы изменить состояние битов регистра STATUS рекомендуется использовать команды BCF и BSF.

5.10 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с ЦПУ и АЛУ микроконтроллеров PICmicro MCU:

Документ	Номер
Fixed Point Routines Математические подпрограммы с фиксированной точкой	AN617
IEEE 754 Compliant Floating Point Routines Математические подпрограммы с плавающей точкой, совместимые с IEEE 754	AN575
Digital Signal Processing with the PIC16C74 Цифровая обработка сигналов в PIC16C74	AN616
Math Utility Routines Математические gjlghjuhfvvs	AN544
Implementing IIR Digital Filters Реализация цифровых фильтров IIR	AN540
Implementation of Fast Fourier Transforms Выполнение быстрого преобразование Фурье	AN542
Tone Generation Генератор тона	AN543
Servo Control of a DC Brushless Motor Управление двигателем постоянного тока	AN532
Implementation of the Data Encryption Standard using the PIC17C42 Выполнение стандарта шифрования данных на PIC17C42	AN583
PIC16C5X / PIC16CXX Utility Math Routines Математические подпрограммы для PIC16C5X/PIC16CXX	AN526
Real Time Operating System for PIC16/17 Выполнение операций в режиме реального времени	AN585

Раздел 6.Организация памяти

Содержание

6.1 Введение

В разделе 6 "Организация памяти" описывается два независимых блока памяти: память программ и память данных. Каждый блок имеет собственную шину данных и шину адреса, позволяя организовать одновременный доступ к обоим типам памяти в течение одного машинного цикла.

Память данных состоит из регистров общего (GPR) и специального (SFR) назначения. Регистры SFR, управляющие ядром микроконтроллера, будут описаны в данном разделе. Описание регистров SFR, управляющие периферийными модулями, смотрите в соответствующем разделе документации.

6.2 Организация памяти программ

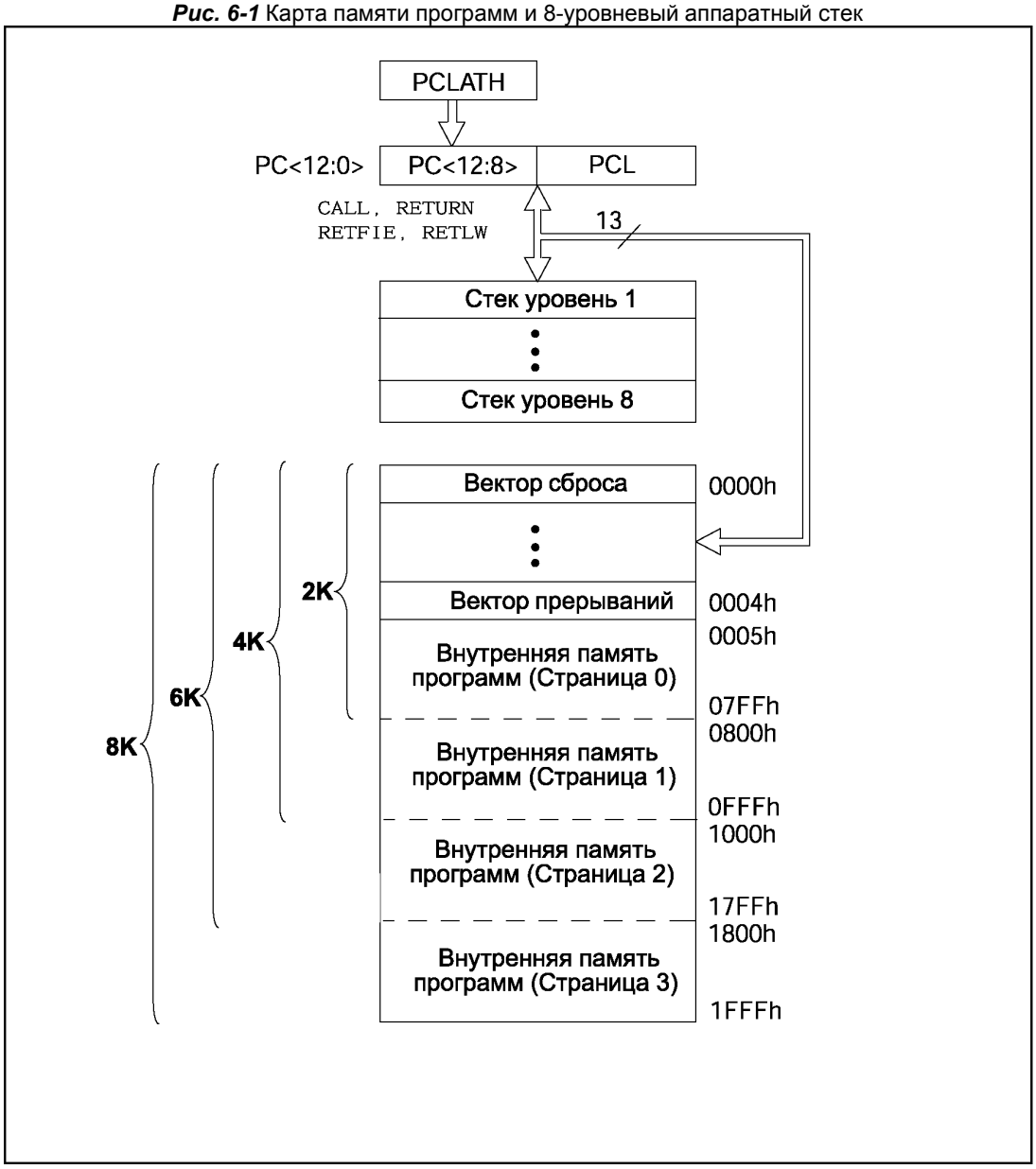
Микроконтроллеры среднего семейства имеют 13-разрядный счетчик команд, способный адресовать 8К x 14 слов памяти программ, и 14-разрядную шину данных памяти программ. Все команды микроконтроллера состоят из 14-разрядного слова, поэтому микроконтроллер с объемом памяти программ 8К x 14 может содержать 8К команд. Это позволяет легко определить достаточность объема памяти программ для желаемого приложения.

Вся память программ разделена на 4 страницы по 2Кслов каждая (0000h-07FFh, 0800h–0FFFh, 1000h–17FFh, 1800h–1FFFh). На рисунке 6-1 показана карта памяти программ и 8-уровневый аппаратный стек. В зависимости от типа микроконтроллера, только некоторая часть доступной памяти программ реализована аппаратно (смотрите техническую документацию на конкретный микроконтроллер).

Для перехода между страницами памяти программ необходимо изменить старшие биты регистра счетчика команд PC, записью в регистр специального назначения PCLATH (старший байт счетчика команд). Изменив значение регистра PCLATH и выполнив команду ветвления, счетчик команд PC пересечет границу страницы памяти программ без дополнительного вмешательства пользователя.

Для микроконтроллеров, имеющих память программ меньше 8Кслов, обращение к памяти программ выше фактически реализованного значения приведет к циклической адресации. Например, в микроконтроллере с памятью программ 4Кслов и попытке перехода по адресу 17FFh переход будет выполнен по адресу 07FFh. В микроконтроллерах с памятью программ 2Кслов управление страницами памяти не требуется.

Рис. 6-1 Карта памяти программ и 8-уровневый аппаратный стек



Примечания:

1. Не во всех микроконтроллерах полностью реализовано адресное пространство памяти программ.
2. В памяти программ может размещаться калибровочная информация.

6.2.1 Вектор сброса

В любом микроконтроллере PICmicro сброс приведет к очистке счетчика команд (PC), устанавливая адрес 0h. Адрес 0000h называется "адрес вектора сброса", т.к. будет выполнен переход по этому адресу при сбросе микроконтроллера. Вместе со счетчиком команд (PC) очищается регистр PCLATH, устанавливая рабочую страницу памяти программ 0.

6.2.2 Вектор прерываний

Когда возникает разрешенное прерывание, в счетчик команд PC записывается адрес 0004h, называемый "адрес вектора прерываний", при этом значение регистра PCLATH не изменяется.

Если в подпрограмме обработки прерываний требуется выполнять команды ветвления, то необходимо предварительно записать в регистр PCLATH значение, определяющее нужную страницу памяти программ. Прежде чем регистр PCLATH будет изменен, его значение должно быть сохранено в другом регистре памяти данных, а затем восстановлено перед выходом из подпрограммы обработки прерываний.

6.2.3 Калибровочная информация

В некоторых типах микроконтроллеров, во время заключительного заводского испытания в памяти программ сохраняется калибровочная информация. Использование калибровочной информации позволяет приложению получать наилучшие результаты работы. Как правило, калибровочная информация сохраняется в конце памяти программ набором инструкций RETLW.

Примечание. Для микроконтроллеров, в которых перед программированием очищается вся память программ, рекомендуется сначала прочитать калибровочную информацию, а затем запрограммировать микроконтроллер.

6.2.4 Счетчик команд PC

13-разрядный регистр счетчика команд PC указывает адрес выбираемой команды для выполнения. Младший байт счетчика программ PCL доступен для чтения и записи. Старший байт PCH, содержащий <12:8> биты счетчика команд PC, не доступен для чтения и записи. Все операции с регистром PCH происходят через дополнительный регистр PCLATH.

На рисунках 6-2 показано 4 примера изменения значения счетчика команд PC.

Рис. 6-2а Непосредственная запись значения в регистр PCL (PCLATH<4:0> → PCH)

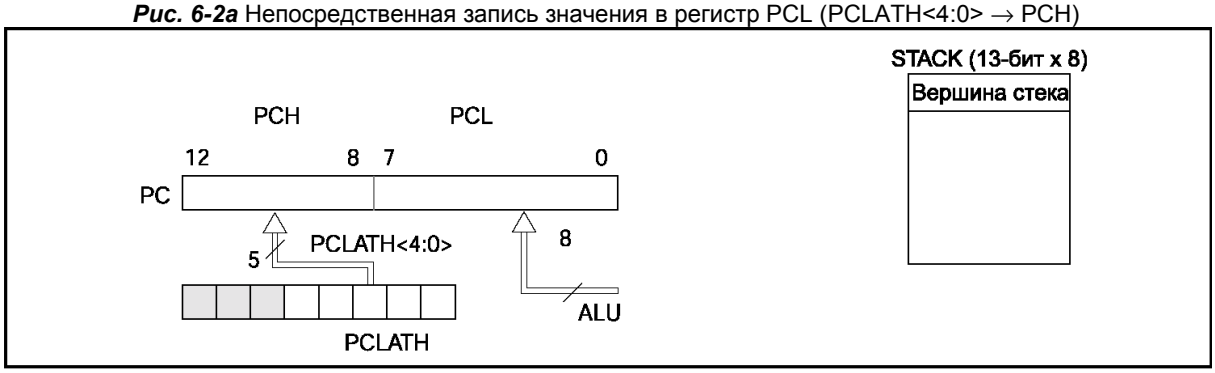


Рис. 6-2б Изменение значения PC при выполнении инструкции GOTO (PCLATH<4:3> → PCH)

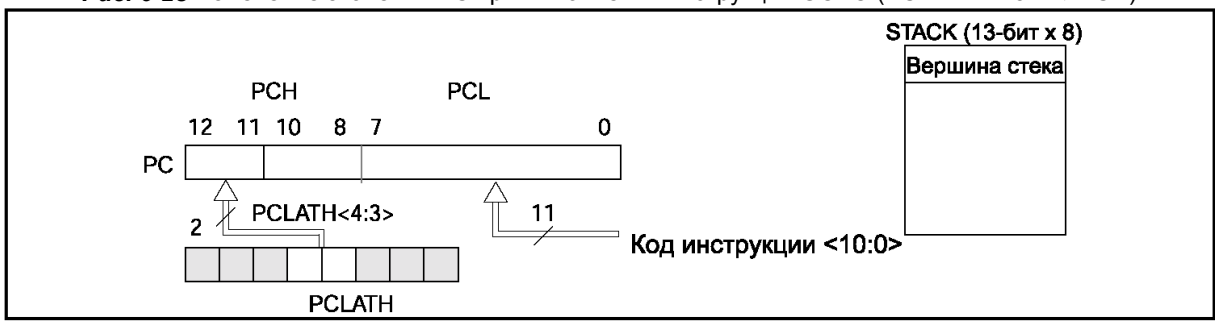


Рис. 6-2в Изменение значения PC при выполнении перехода к подпрограмме CALL (PCLATH<4:3> → PCH), при чем старое значение PC сохраняется в вершине стека

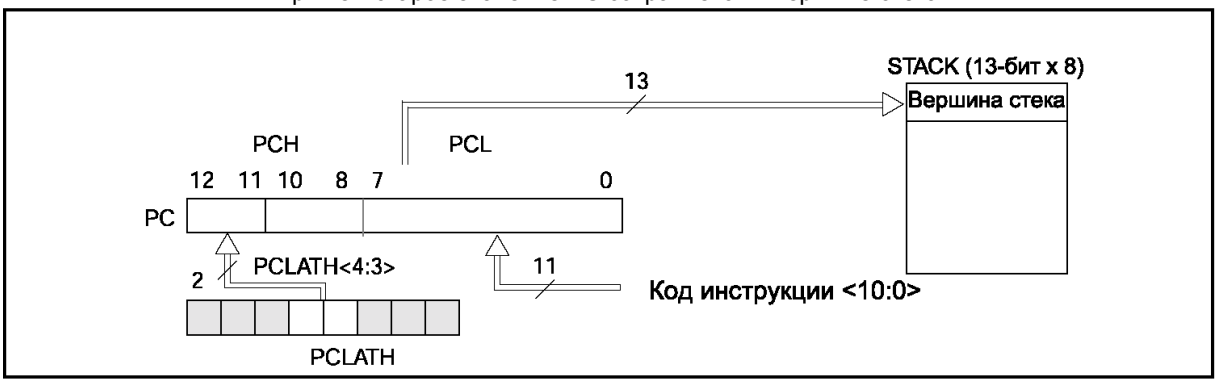
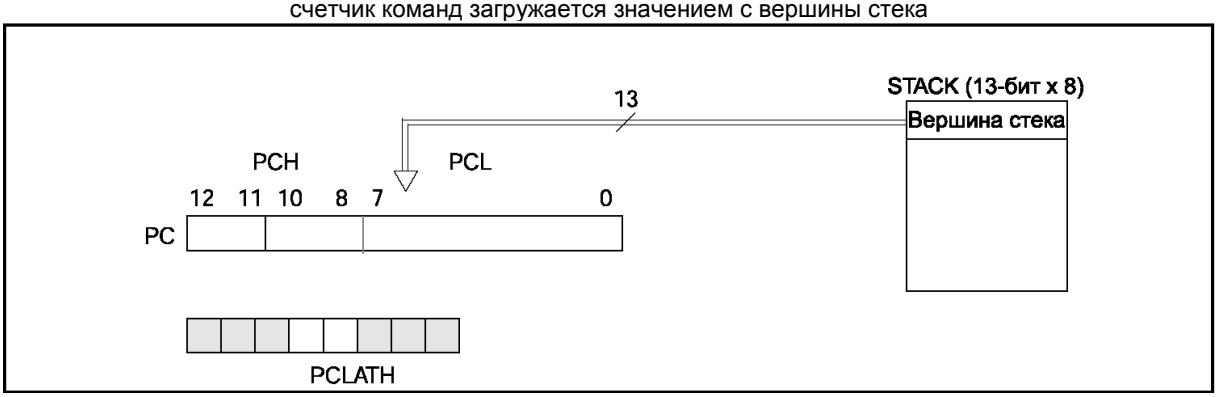


Рис. 6-2г Возвращение из подпрограммы (RETURN, RETFIE или RETLW), счетчик команд загружается значением с вершины стека



Примечание. Регистр PCLATH не изменяется при изменении PCH.

6.2.4.1 Вычисляемый переход

Вычисляемый переход может быть выполнен командой приращения к регистру PCL (например, ADDWF PCL). При выполнении вычисляемого перехода следует заботиться о том, чтобы значение PCL не пересекло границу блока памяти (каждый блок 256 байт).

Примечание. При записи значения в регистр PCL, автоматически происходит перезапись 5 младших бит из регистра PCLATH<4:0> в регистр PCH.

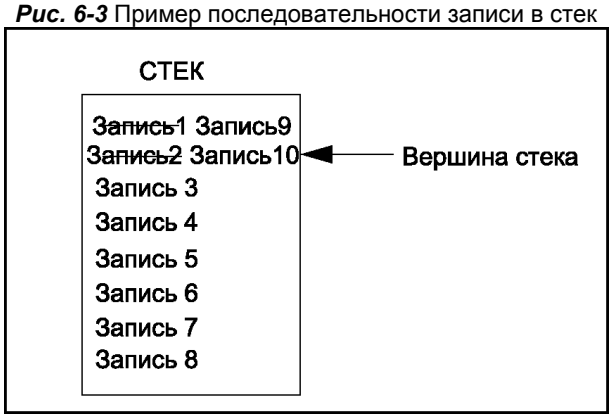
6.2.5 Аппаратный стек

Стек поддерживает до 8 уровней вложенности подпрограмм пользователя, включая обработку прерываний. В стеке сохраняется адрес возврата в основную программу.

В микроконтроллерах среднего семейства PICmicro реализован 8-уровневый 13-разрядный аппаратный стек. Стек не имеет отображения на память программ и память данных, нельзя записать или прочитать данные из стека. Значение счетчика команд заносится в вершину стека при выполнении инструкций перехода на подпрограмму (CALL) или обработку прерываний. Чтение из стека и запись в счетчик команд PC происходит при выполнении инструкций возврата из подпрограммы или обработки прерываний (RETURN, RETLW, RETFIE), при этом значение регистра PCLATH не изменяется.

После 8 записей в стек, девятая запись запишется на место первой, а десятая запись заменит вторую и так далее. Смотрите пример на рисунке 6-3.

Рис. 6-3 Пример последовательности записи в стек



Примечание 1. В микроконтроллерах не имеется никаких указателей о переполнении стека.

Примечание 2. В микроконтроллерах не предусмотрено команд записи/чтения из стека, кроме команд вызова/возврата из подпрограммы (CALL, RETURN, RETLW и RETFIE) или условий перехода по вектору прерываний.

6.2.6 Страницы памяти программ

Команды переходов (CALL, GOTO) в микроконтроллерах среднего семейства PICmicro имеют 11-разрядное поле для указания адреса, что позволяет непосредственно адресовать 2Кслов памяти программ. Некоторые микроконтроллеры имеют память программ более 2Кслов. Для адресации верхних страниц памяти программ используются 2 бита в регистре PCLATH<4:3>. Перед выполнением команды перехода (CALL или GOTO) необходимо запрограммировать биты регистра PCLATH<4:3> для адресации требуемой страницы (см. рисунки 6-2б, 6-2в).

При выполнении инструкций возврата из подпрограммы 13-разрядное значение для счетчика программ PC берется с вершины стека, поэтому манипуляция битами регистра PCLATH<3:4> не требуется (см. рисунок 6-2г).

Примечание.

В микроконтроллерах с объемом памяти программ до 2Кслов биты регистра PLATH<4:3> игнорируются. Не рекомендуется их использовать как биты общего назначения, т.к. может возникнуть необходимость переноса программы на микроконтроллер с большим объемом памяти программ.

В микроконтроллерах с объемом памяти программ до 4Кслов бит регистра PLATH<4> игнорируется. Не рекомендуется его использовать как бит общего назначения, т.к. может возникнуть необходимость переноса программы на микроконтроллер с большим объемом памяти программ.

В примере 6-1 показан переход со страницы 0 на страницу 1 памяти программ. Этот пример предполагает, что в подпрограмме сохраняется и восстанавливается значение регистра PCLATH.

Пример 6-1 Выполнение перехода со страницы 0 на страницу 1 памяти программ

```
                ORG     0x500
                BSF     PCLATH,3        ; Выбор страницы 1 (800h-FFFh)
                CALL    SUB1_P1         ; Переход на страницу 1 (800h-FFFh)
                :
                :

                ORG     0x900
SUB1_P1:                                ; Страница 1 (800h-FFFh)
                :
                RETURN                  ; Возврат на страницу 0 (000h-7FFh)
```

6.3 Организация памяти данных

Память данных разделяется на регистры двух типов:

- Регистры специального назначения (SFR), управляют работой микроконтроллера;
- Регистры общего назначения (GPR), для хранения данных программы.

Память данных разделена на банки, содержащие регистры общего и специального назначения. Регистры общего назначения размещаются в разных банках памяти данных для того, чтобы была возможность организовать более 96 байт ОЗУ. Регистры специального назначения предназначены для управления периферийными модулями и функциями микроконтроллера. Управление банками памяти выполняется битами в регистре STATUS<7:5>. На рисунке 6-5 представлена одна из разновидностей карты памяти данных. Организация памяти данных зависит от типа микроконтроллера.

Чтобы передать данные из одного регистра в другой, необходимо использовать дополнительный регистр W. Эта операция выполняется двумя командами за два машинных цикла микроконтроллера.

Обращение к всем регистрам памяти данных может быть выполнено прямой или косвенной адресацией:

- Прямая адресация - для указания банка памяти данных необходимо использовать биты RP1:RP0 регистра STATUS;
- Косвенная адресация - адрес регистра сохраняется в FSR, а в бите IRP регистра STATUS указывается к какой паре банков памяти данных выполняется обращение (Банк0/Банк1 или Банк2/Банк3).

6.3.1 Регистры общего назначения (GRP)

Регистры общего назначения размещаются в разных банках памяти данных. Эти регистры не инициализируются при сбросе по включению питания и имеют неизвестное значение, а при всех остальных сбросах микроконтроллера не изменяют своего значения.

Обращение к регистрам общего назначения может быть выполнено прямой или косвенной адресацией (через регистры FSR и INDF). В некоторых микроконтроллерах существуют регистры общего назначения, адресуемые к одной и той же ячейке ОЗУ, независимо от текущего банка памяти данных. Обратите внимание на эти регистры, т.к. они расположены в общем ОЗУ.

6.3.2 Регистры специального назначения (SFR)

Регистры специального назначения используются для управления ядром и периферийными модулями микроконтроллера. Эти регистры реализованы как статическое ОЗУ. Описание регистров SFR, управляющих периферийными модулями, смотрите в соответствующем разделе документации.

Регистры специального назначения размещены в различных банках памяти данных, а некоторые из регистров отображаются во всех банках.

Переключение рабочего банка памяти выполняется настройкой битов RP1:RP0 регистра STATUS. При сбросе по включению питания и других видах сброса микроконтроллера в некоторые регистры специального назначения записывается определенное значение. Существуют регистры SFR, которые содержат неизвестное значение при сбросе по включению питания, а при других видах сброса не изменяются (см. техническую документацию на соответствующий микроконтроллер).

Обращение к регистрам специального назначения может быть выполнено прямой или косвенной адресацией.

Примечание. В области регистров специального назначения могут размещаться регистры общего назначения.

6.3.3 Банки памяти данных

Память данных разделена на 4 банка. Каждый банк содержит регистры специального назначения (в начале адресного пространства банка) и регистры общего назначения. Переключение между банками осуществляется с помощью битов:

- RP1:RP0 в регистре STATUS при прямой адресации;
- IRP в регистре STATUS при косвенной адресации.

Таблица 6-1 Выбор банка памяти данных при прямой и косвенной адресации

Доступ к банку	Прямая адресация (RP1:RP0)	Косвенная адресация (IRP)
0	0 0	0
1	0 1	
2	1 0	1
3	1 1	

Вся память данных выполнена по технологии статического ОЗУ с максимальным размером банка памяти 7Fh (128 байт). В начале банка располагаются регистры специального назначения, за ними регистры общего назначения. Некоторые, часто используемые регистры специального назначения банка 0 отображаются в других банках памяти для упрощения программы микроконтроллера и получения быстрого доступа к ним.

В процессе разработки новых микроконтроллеров, адреса размещения регистров специального назначения в памяти данных претерпели некоторые изменения. Организация памяти данных, показанная на рисунке 6-5, является стандартом для всех новых микроконтроллеров. Последние 16 байт всех банков памяти данных, показанных на карте, отображаются на банк 0, что должно упростить программу, работающую с банками памяти данных. Регистры, имена которых выделены полужирным текстом, присутствуют во всех микроконтроллерах.

Состав регистров специального назначения, их адреса в памяти данных и объем регистров общего назначения для конкретного типа микроконтроллера смотрите в технической документации на соответствующий микроконтроллер.

Рис. 6-4 Механизм прямой адресации памяти данных

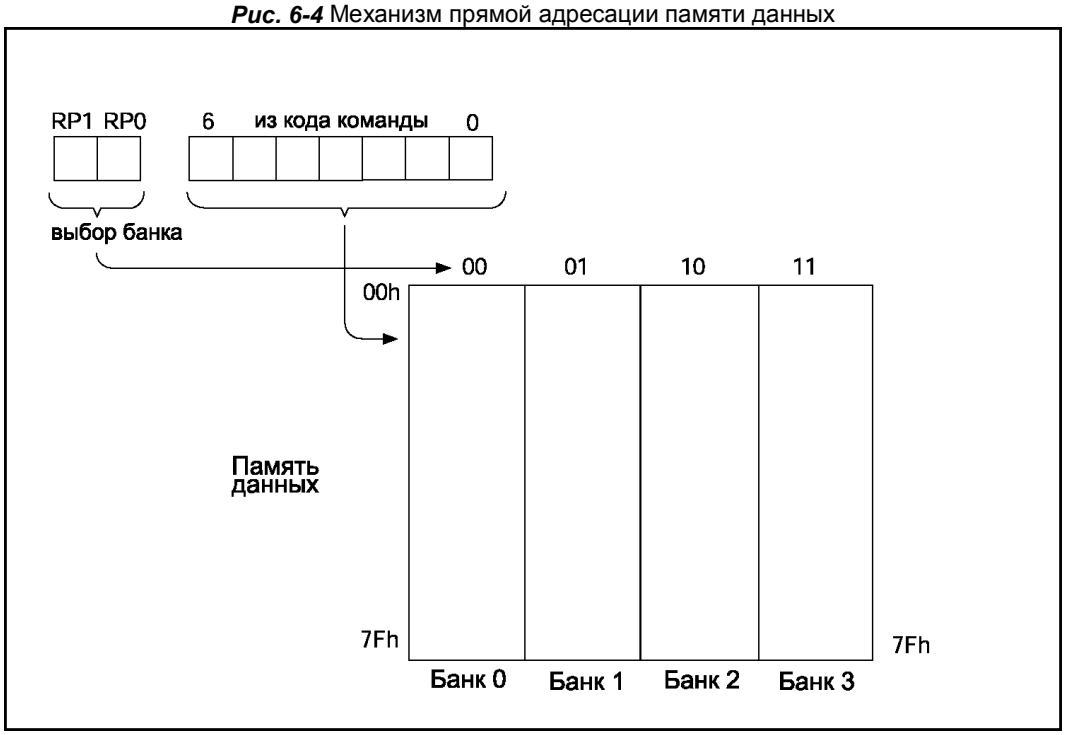


Рис. 6-5 Карта памяти данных

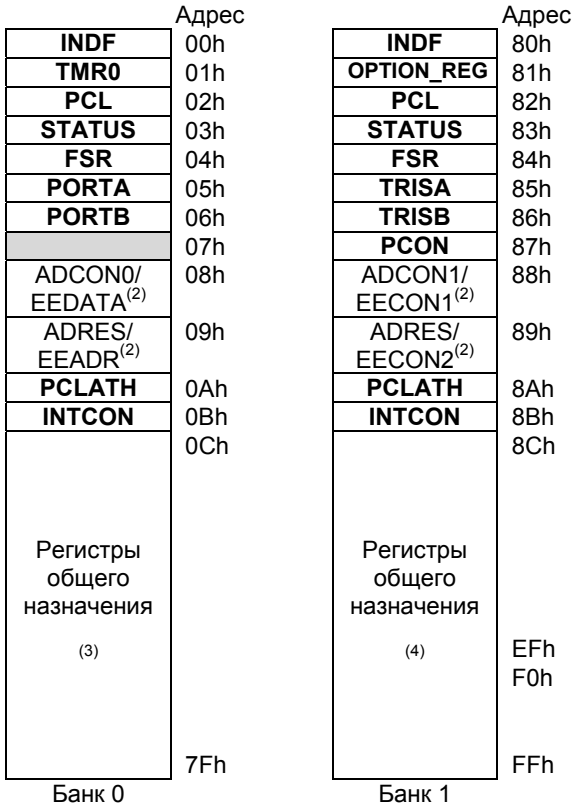
Банк 0	Адрес	Банк 1	Адрес	Банк 2[5]	Адрес	Банк 3[5]	Адрес
INDF	00h	**INDF**	80h	**INDF**	100h	**INDF**	180h
TMR0	01h	**OPTION_REG**	81h	**TMR0**	101h	**OPTION_REG**	181h
PCL	02h	**PCL**	82h	**PCL**	102h	**PCL**	182h
STATUS	03h	**STATUS**	83h	**STATUS**	103h	**STATUS**	183h
FSR	04h	**FSR**	84h	**FSR**	104h	**FSR**	184h
PORTA	05h	TRISA	85h		105h		185h
PORTB	06h	TRISB	86h	PORTB	106h	TRISB	186h
PORTC	07h	TRISC	87h		107h		187h
PORTD	08h	TRISD	88h		108h		188h
PORTE	09h	TRISE	89h		109h		189h
PCLATH	0Ah	**PCLATH**	8Ah	**PCLATH**	10Ah	**PCLATH**	18Ah
INTCON	0Bh	**INTCON**	8Bh	**INTCON**	10Bh	**INTCON**	18Bh
PIR1	0Ch	**PIE1**	8Ch	EEDATA	10Ch	EECON1	18Ch
PIR2	0Dh	PIE2	8Dh	EEADR	10Dh	EECON2	18Dh
TMR1L	0Eh	**PCON**	8Eh		10Eh		18Eh
TMR1H	0Fh		8Fh		10Fh		18Fh
T1CON	10h		90h		110h		190h
TMR2	11h	SSPCON2	91h		111h		191h
T2CON	12h	PR2	92h		112h		192h
SSPBUF	13h	SSPADD	93h		113h		193h
SSPCON	14h	SSPSTAT	94h		114h		194h
CCPR1L	15h		95h		115h		195h
CCPR1H	16h		96h		116h		196h
CCP1CON	17h		97h		117h		197h
RCSTA	18h	TXSTA	98h		118h		198h
TXREG	19h	SPBRG	99h		119h		199h
RCREG	1Ah		9Ah		11Ah		19Ah
CCPR2L	1Bh		9Bh		11Bh		19Bh
CCPR2H	1Ch		9Ch		11Ch		19Ch
CCP2CON	1Dh		9Dh		11Dh		19Dh
ADRES	1Eh		9Eh		11Eh		19Eh
ADCON0	1Fh	ADCON1	9Fh		11Fh		19Fh
Регистры общего назначения [2]	20h – 7Fh	Регистры общего назначения [3]	A0h – EFh	Регистры общего назначения [3]	120h – 16Fh	Регистры общего назначения [3]	1A0h – 1EFh
		Доступ к 70h-7Fh[4]	F0h – FFh	Доступ к 70h-7Fh[4]	170h – 17Fh	Доступ к 70h-7Fh[4]	1F0h – 1FFh

Примечания:

1. Регистры, названия которых выделены полужирным текстом, будут присутствовать в каждом микроконтроллере.
2. В некоторых микроконтроллерах часть банка памяти данных может быть не реализована. Чтение по адресу несуществующего ОЗУ будет давать результат 00h.
3. Эта часть ОЗУ может быть не реализована. Чтение по адресу несуществующего ОЗУ будет давать непредсказуемый результат.
4. В некоторых микроконтроллерах эти регистры общего назначения отображается на банк 0.
5. Банк может быть не реализован. Чтение по адресу несуществующего ОЗУ будет давать результат 00h.
6. Регистры общего назначения могут быть расположены в области регистров специального назначения.

На рисунке 6-6 показана карта памяти данных некоторых 18-выводных микроконтроллеров. Чтение не реализованных регистров будет давать результат '0'.

Рис. 6-6 Карта памяти данных некоторых 18 - выводных микроконтроллеров



Примечания:

1. Регистры, названия которых выделены жирным текстом, будут присутствовать в каждом микроконтроллере.
2. Эти регистры могут быть не реализованы или иметь другое назначение.
3. Не вся часть банка ОЗУ может быть реализована. Чтение по адресу несуществующего ОЗУ будет давать результат '0'.
4. Регистры памяти данных в банке 1 могут быть не реализованы. Обращение к не реализованным регистрам банка 1 вызовет действие с регистром банка 0.

6.3.4 Косвенная адресация, регистры INDF и FSR

Косвенная адресация – такой режим адресации регистров, при котором в команде не указывается адрес памяти данных. Регистр специального назначения FSR, доступный для записи/чтения, используется в качестве указателя адреса в памяти данных. Этот режим адресации может быть полезен при обращении к таблицам данных.

На рисунке 6-7 показана операция косвенной адресации (запись значения в регистр памяти данных с адресом, указанном в регистре FSR).

Для выполнения косвенной адресации необходимо обратиться к регистру INDF. Обращение к регистру INDF фактически вызовет действие с регистром, адрес которого указан в FSR. Косвенное чтение регистра INDF (FSR=0) даст результат 00h. Косвенная запись в регистр INDF не вызовет никаких действий (вызывает воздействия на флаги АЛУ в регистре STATUS). 9-й бит косвенного адреса IRP сохраняется в регистре STATUS<7>. Пример 9-разрядной косвенной адресации показан на рисунке 6-8.

Рис. 6-7 Косвенная адресация

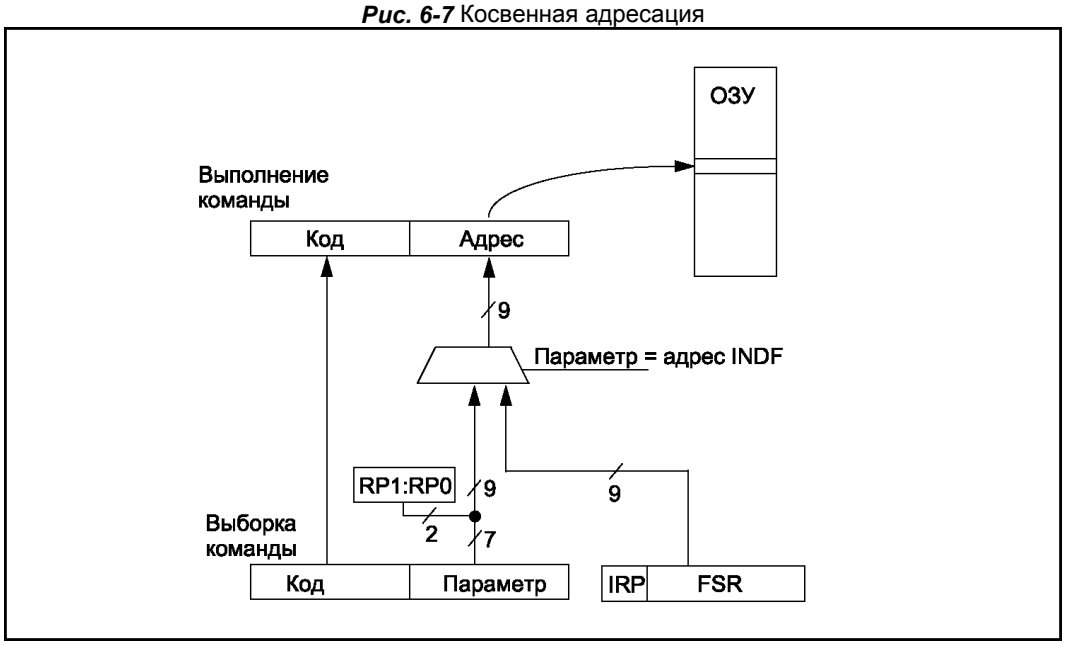
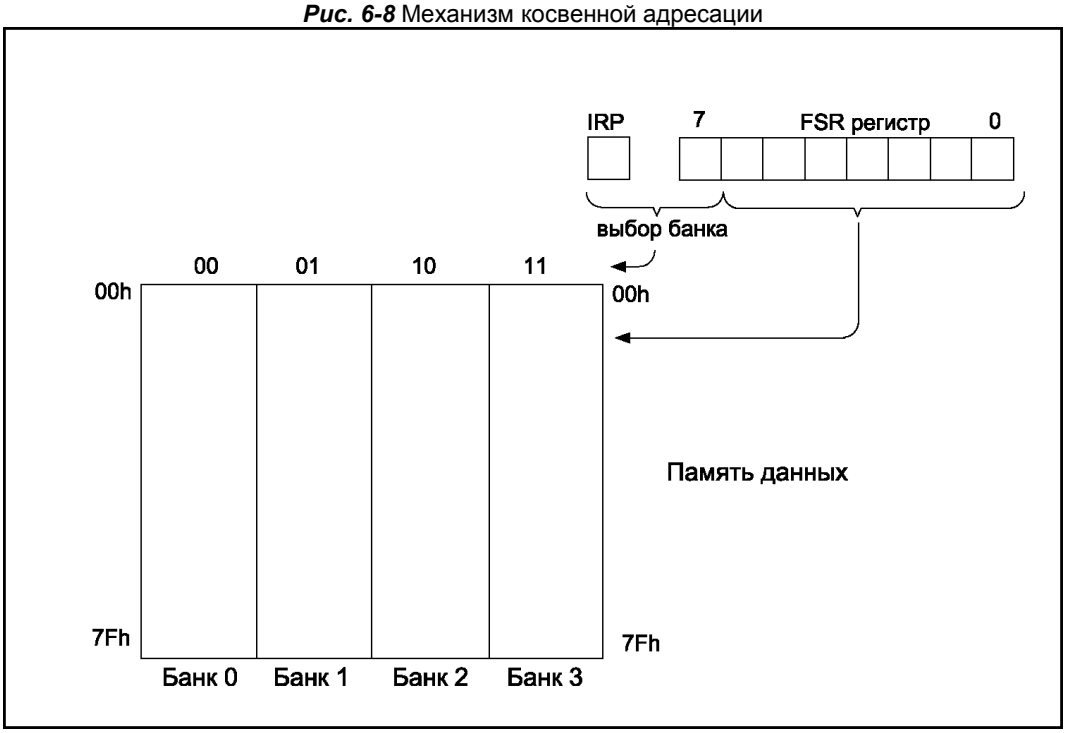


Рис. 6-8 Механизм косвенной адресации



В примере 6-2 показано использование косвенной адресации для очистки памяти данных (адреса 20h – 2Fh) минимальным числом команд микроконтроллера. Подобный метод может использоваться для передачи блока данных в регистр TXREG передатчика USART. Начальный адрес блока данных, подготовленного для передачи, может быть легко изменен в соответствии с требованиями программы.

Пример 6-2 Очистка регистров памяти данных с адреса 20h по 2Fh

```
                BCF     STATUS, IRP     ; Установить банк 0,1
                MOVLW   0x20            ; Указать первый регистр в ОЗУ
                MOVWF   FSR
NEXT:
                CLRF    INDF            ; Очистить регистр
                INCF    FSR,F           ; Увеличить адрес
                BTFSS   FSR,4           ; Завершить?
                GOTO    NEXT            ; Нет, продолжить очистку

CONTINUE:
                                        ; Да
```

6.4 Инициализация

В примере 6-3 показано переключение между банками памяти данных для прямой адресации, а в примере 6-4 представлен код программы, выполняющей инициализацию (очистку) регистров общего назначения.

Пример 6-3 Переключение банков памяти данных

```
CLRF   STATUS        ; Очистка регистра STATUS (Банк 0)
:
BSF    STATUS, RP0   ; Банк 1
:
BCF    STATUS, RP0   ; Банк 0
:
MOVLW 0x60           ; Установить RP0 и RP1 в STATUS регистре
XORWF STATUS, F      ; (Банк 3)
:
BCF    STATUS, RP0   ; Банк 2
:
BCF    STATUS, RP1   ; Банк 0
```

Пример 6-4 Инициализация регистров общего назначения

```
                CLRF   STATUS        ; Очистить регистр STATUS (Банк 0)
                MOVLW  0x20          ; 1-й адрес регистра GPR
                MOVWF  FSR           ; записать в регистр косвенного адреса
Bank0_LP
                CLRF   INDF0         ; Очистить регистр GPR с адресом в регистре FSR
                INCF   FSR           ; Следующий регистр GPR
                BTFSS  FSR, 7        ; Очистка регистров GPR в этом банке завершена?
                                     ; (FSR = 80h, C = 0)
                GOTO   Bank0_LP      ; НЕТ, продолжать очистку
                                     ;
                                     ; Следующий банк (Банк 1)
                                     ; (** Только для микроконтроллеров с банком 1 **)
                                     ;
                MOVLW  0xA0          ; 1-й адрес регистра GPR
                MOVWF  FSR           ; записать в регистр косвенного адреса
Bank1_LP
                CLRF   INDF0         ; Очистить регистр GPR с адресом в регистре FSR
                INCF   FSR           ; Следующий регистр GPR
                BTFSS  STATUS, C     ; Очистка регистров GPR в этом банке завершена?
                                     ; (FSR = 00h, C = 1)
                GOTO   Bank1_LP      ; НЕТ, продолжать очистку
                                     ;
                                     ; Следующий банк (Банк 2)
                                     ; (** Только для микроконтроллеров с банком 2 **)
                                     ;
                BSF    STATUS, IRP   ; Выбор банков 2 и 3  для косвенной адресации
                MOVLW  0x20          ; 1-й адрес регистра GPR
                MOVWF  FSR           ; записать в регистр косвенного адреса
Bank2_LP
                CLRF   INDF0         ; Очистить регистр GPR с адресом в регистре FSR
                INCF   FSR           ; Следующий регистр GPR
                BTFSS  FSR, 7        ; Очистка регистров GPR в этом банке завершена?
                                     ; (FSR = 80h, C = 0)
                GOTO   Bank2_LP      ; НЕТ, продолжать очистку
                                     ;
                                     ; Следующий банк (Банк 3)
                                     ; (** Только для микроконтроллеров с банком 3 **)
                                     ;
                MOVLW  0xA0          ; 1-й адрес регистра GPR
                MOVWF  FSR           ; записать в регистр косвенного адреса
Bank3_LP
                CLRF   INDF0         ; Очистить регистр GPR с адресом в регистре FSR
                INCF   FSR           ; Следующий регистр GPR
                BTFSS  STATUS, C     ; Очистка регистров GPR в этом банке завершена?
                                     ; (FSR = 00h, C = 1)
                GOTO   Bank3_LP      ; НЕТ, продолжать очистку
                :                    ; ДА, все регистры GPR очищены
```

6.5 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Выполнение программы прекращается, что может быть причиной?
Ответ 1:
Если микроконтроллер содержит более 2кслов памяти программ, то возможно, что перед переходом на подпрограмму по команде CALL (или безусловному переходу GOTO) необходимо настроить регистр PCLATH для выбора нужной страницы памяти. В примере показана последовательность команд загрузки регистра PCLATH вне зависимости от расположения метки SUB_1 в памяти программ.


```
        MOVLW  HIGH   (SUB_1)      ; Выбор страницы памяти программ,
        MOVWF  PCLATH              ; в которой размещается подпрограмма
        CALL   SUB_1               ; Переход на подпрограмму
        :
        :
SUB_1   :                          ; Начало подпрограммы
        :
```

Вопрос 2: Мне необходимо записать во все регистры общего назначения 00h. Как наиболее просто это сделать?
Ответ 2:
Смотрите пример 6-4 этого раздела. Если в микроконтроллере не реализованы все 4 банка памяти данных, то часть кода может быть удалена.

6.6 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с организацией памяти в микроконтроллерах PICmicro MCU:

Документ	Номер
Implementing a Table Read Выполнение табличного чтения	AN556

Раздел 7.EEPROM память данных

Содержание

7.1 Введение

EEPROM память данных доступна для записи/чтения в нормальном режиме работы микроконтроллера во всем диапазоне рабочего напряжения питания (V_{DD}). EEPROM память не отображается на адресное пространство памяти данных, а доступна через регистры специального назначения. Для доступа к EEPROM памяти данных используются 4 регистра специального назначения:

- EECON1
- EECON2 (не физический регистр)
- EEDATA
- EEADR

В регистре EEDATA сохраняются 8-разрядные данные записи/чтения, а регистр EEADR содержит адрес ячейки EEPROM памяти данных. С помощью 8 - разрядного регистра EEADR можно адресовать 256 байт EEPROM памяти данных. EEADR можно рассматривать как регистр косвенной адресации для ячеек EEPROM памяти. Регистр EECON1 содержит биты управления чтением/записью EEPROM памяти данных, а не физический регистр EECON2 предназначен для защиты от случайной записи в EEPROM память данных.

В некоторых микроконтроллерах реализовано вся возможная EEPORM память данных. Область реализованной EEPROM памяти данных всегда начинается с адреса 00h. В таблице 7-1 показаны возможные объемы EEPROM памяти данных и адреса реализованных ячеек.

Таблица 7-1 Объем EEPROM памяти данных

Объем EEPROM памяти данных (байт)(1)	**Адреса реализованных ячеек**
64	00h - 3Fh
128	00h - 7Fh
256	00h - FFh

Примечание 1. В выпускаемых микроконтроллерах среднего семейства минимальный объем EEPROM памяти данных 64 байта.

EEPROM память данных позволяет выполнить чтение и запись байта. При записи байта происходит автоматическое стирание ячейки и запись новых данных (стирание перед записью). EEPROM память данных рассчитана на большое количество циклов стирание/запись. Время записи управляется интегрированным таймером и зависит от напряжения питания, температуры и технологического разброса параметров кристалла (см. раздел "Электрические характеристики").

При установке защиты на доступ к EEPROM памяти данных, программа микроконтроллера имеет возможность выполнить запись/чтение EEPROM памяти данных. Доступ закрыт для записи/чтения программатором.

7.2 Управляющий регистр

Регистр EECON1

U-0	U-0	U-0	R/W-0	R/W-x	R/W-0	R/S-0	R/S-x
-	**-**	**-**	**EEIF(1)**	**WRERR**	**WREN**	**WR**	**RD**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

биты 7-5: **Не используются:** читаются как ‘0’

бит 4: **EEIF**: Флаг прерывания по окончанию записи в EEPROM данных
1 = запись в EEPROM данных завершена (сбрасывается программно)
0 = запись в EEPROM данных не завершена или не была начата

бит 3: **WRERR:** Флаг ошибки записи в EEPROM память данных
1 = запись прервана (произошел один из сбросов: по сигналу -MCLR, по переполнению WDT в нормальном режиме, по снижению напряжения питания BOR)
0 = запись завершена

бит 2: **WREN:** Разрешение записи в EEPROM память данных
1 = запись разрешена
0 = запись запрещена

бит 1: **WR:** Инициализировать запись в EEPROM память данных (программно может быть только установлен в ‘1’)
1 = инициализировать запись (сбрасывается в ‘0’ аппаратно)
0 = запись завершена

бит 0: **RD:** Инициализировать чтение из EEPROM памяти данных (программно может быть только установлен в ‘1’)
1 = инициализировать чтение (сбрасывается в ‘0’ аппаратно)
0 = чтение завершено

Примечание 1. В некоторых микроконтроллерах этот бит находится в регистре PIR.

7.3 Регистр EEADR

С помощью регистра EEADR можно адресовать 256 байт EEPROM памяти данных. Старший биты регистра EEADR тоже участвует в декодировании адреса, поэтому всегда должен равняться нулю (для гарантированной адресации памяти).

7.4 Регистры EECON1, EECON2

Регистр EECON1 содержит 5 (младших) физически реализованных управляющих битов. Три старших бита не реализованы и читаются как ‘0’.

Управляющие биты RD и WR инициализируют соответственно чтение и запись данных. Программно эти биты могут быть только установленны в ‘1’, сброс в ‘0’ происходит аппаратно по завершению операции чтения/записи. Защита от программного сброса этих битов позволяет предотвратить преждевременное завершение операции записи.

Если бит WREN=1, то разрешена запись в EEPROM память данных. После сброса по включению питания (POR) бит WREN равен ‘0’. Бит WRERR устанавливается в ‘1’, если во время выполнения записи в EEPROM память данных произошел сброс по сигналу -MCLR или по переполнению сторожевого таймера WDT в нормальном режиме. Проверив состояние бита WREER, пользователь может повторить запись (регистры EEDATA и EEADR не изменяют своего значения).

После завершения записи в EEPROM память данных устанавливается флаг EEIF в ‘1’ (сбрасывается программно).

Регистр EECON2 не реализован физически, читается как 00h. Он используется в операциях записи в EEPROM память данных для реализации обязательной последовательности команд.

7.5 Чтение из EEPROM памяти данных

Для чтения EEPROM памяти данных необходимо записать адрес в регистр EEADR и установить бит RD (EECON1<0>) в ‘1’. В следующем машинном цикле данные доступны для чтения из регистра EEDATA. Прочитанное значение из EEPROM памяти данных будет храниться в регистре EEDATA до следующего чтения или записи в этот регистр по команде микроконтроллера.

Пример 7-1 Чтение из EEPROM памяти данных

```
        BCF     STATUS, RP0         ; Выбрать банк 0
        MOVLW   CONFIG_ADDR         ;
        MOVWF   EEADR               ; Адрес считываемого регистра
        BSF     STATUS, RP0         ; Выбрать банк 1
        BSF     EECON1, RD          ; Чтение
        BCF     STATUS, RP0         ; Выбрать банк 0
        MOVF    EEDATA, W           ; W = EEDATA
```


7.6 Запись в EEPROM память данных

Для записи в EEPROM память данных необходимо записать адрес в регистр EEADR, данные в регистр EEDATA и выполнить последовательность команд, показанных в примере 7-2.

Пример 7-2 Запись в EEPROM память данных

```
        BSF     STATUS, RP0         ; Выбрать банк 1
        BSF     EECON1, WREN        ; Разрешить запись

Обязательная последовательность
        BCF     INTCON, GIE         ; Запретить прерывания
        MOVLW   55h                 ;
        MOVWF   EECON2              ; Записать 55h
        MOVLW   AAh                 ;
        MOVWF   EECON2              ; Записать AAh
        BSF     EECON1,WR           ; Установить бит WR
                                    ; для начала записи
        BSF     INTCON, GIE         ; Разрешить прерывания
```

Запись байта не будет произведена, если не выполнена указанная последовательность (запись 55h в EECON2, запись AAh в EECON2, установка бита WR в ‘1’ для каждого байта). Рекомендуется запрещать прерывания при выполнении обязательной последовательности команд. Если во время выполнения указанной последовательности произойдет переход по вектору прерывания, запись байта выполнена не будет.

Чтобы разрешить запись в EEPROM память данных, необходимо установить бит WREN (EECON1<2>) в ‘1’, защищающий от случайной записи. Пользователь должен установить бит WREN в ‘1’ перед началом записи, а после окончания записи сбросить его в ‘0’ (аппаратно бит WREN в ‘0’ не сбрасывается).

После инициализации записи сброс бита WREN в ‘0’ не повлияет на цикл записи, но установка бита WR в ‘1’ будет запрещена, пока WREN = 0.

По окончанию записи бит WR аппаратно сбрасывается в ‘0’, а флаг прерывания EEIF устанавливается в ‘1’. Пользователь может использовать прерывания для проверки окончания записи в EEPROM память данных. Флаг EEIF сбрасывается в ’0’ программно.

7.7 Проверка записи

Рекомендуется после выполнения операции записи в EEPROM память данных произвести контрольное чтение (см. пример 7-3). Выполнять контрольное чтение особенно рекомендуется, если возможно исчерпание гарантированных циклов стирание/запись. Основные ошибки возникают при записи отдельных битов равных 1, чтение будет давать результат 0.

Пример 7-3 Проверка записи

```
            BCF     STATUS, RP0     ; Выбрать банк 0
            :                       ; Текст программы
            :                       ;
            MOVF    EEDATA, W       ; Чтение записываемых данных
            BSF     STATUS, RP0     ; Выбрать банк 1
            BSF     EECON1, RD      ; Инициализация чтения из EEPROM
                                    ; записанных данных
            BCF     STATUS, RP0     ; Выбрать банк 0
                                    ;
                                    ; Проверить, равно значение в регистре W
                                    ; и прочитанные данные из EEPROM (EEDATA)?
                                    ;
            SUBWF   EEDATA, W       ;
            BTFSS   STATUS, Z       ; Результат 0?
            GOTO    WRITE_ERR       ; НЕТ, данные записаны неправильно
            :                       ; ДА, данные записаны правильно
            :                       ; Продолжение программы
```

7.8 Защита от случайной записи в EEPROM память данных

Существует несколько условий, когда запись байта в EEPROM память данных не выполняется:

1. После сброса по включению питания POR бит WREN = 0.
2. Таймер включения питания (в течение 72мс) запрещает запись в EEPROM память данных.
3. Обязательная последовательность инициализации записи и бит WREN предотвращают случайную запись.

Все эти меры предотвращают случайную запись в EEPROM память данных при сбое программы, снижении напряжения питания и других ненормальных режимах работы микроконтроллера.

7.9 Операции с EEPROM памятью при установленном бите защиты

При установке защиты на доступ к EEPROM памяти данных, программа микроконтроллера имеет возможность выполнить запись/чтение EEPROM памяти данных. Доступ закрыт для записи/чтения программатором. В микроконтроллерах среднего семейства предусмотрено два бита защиты: бит защиты памяти программ; бит защиты EEPROM памяти данных. Дополнительную информацию смотрите в спецификации программирования микроконтроллеров.

7.10 Инициализация

Модуль EEPROM памяти данных не имеет последовательности инициализации как другие периферийные модули. Выполнять чтение EEPROM памяти данных можно как показано в примере 7-1, а операция записи показана в примере 7-2. Рекомендуется проверять сохраненные данные в EEPROM памяти (см. пример 7-3).

Как и для регистров общего назначения ОЗУ может потребоваться инициализация ячеек EEPROM памяти данных к определенному значению. Инициализация может быть выполнена при программировании микроконтроллера или в специальном диагностическом режиме, предусмотренном программой пользователя.

Условием перехода в диагностический режим может быть определенные логические уровни на входах портов микроконтроллера при включении питания. После перехода в диагностический режим микроконтроллер выполняет определенные действия (например, инициализация ячеек EEPROM памяти). Необходимо заботиться о том, чтобы не возникло случайного перехода в диагностический режим.

7.11 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Почему в ячейку EEPROM памяти не записываются мои данные?
Ответ 1:
Существует несколько причин, но основной является не выполнение обязательной последовательности, показанной в примере 7-2. Если код программы правильный, то необходимо гарантировать, что все прерывания выключены на время выполнения обязательной последовательности.

Вопрос 2: Данные, записанные в EEPROM память разрушаются. Что может быть причиной?
Ответ 2:
Изменение данных EEPROM памяти происходит только при инициализации записи. Неправильная запись может происходить, если микроконтроллер не находится в состоянии сброса при работе в условиях пониженного напряжения питания (ниже уровня, указанного в электрических спецификациях). Необходимо использовать внутреннюю или внешнюю схему сброса по снижению напряжения питания, чтобы гарантировать отсутствие записи в EEPROM память данных при снижении напряжения питания.

7.12 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с EEPROM памятью данных в микроконтроллерах PICmicro MCU:

Документ	Номер
EEPROM Endurance Tutorial Выносливость EEPROM памяти	AN601
How to get 10 Million Cycles out of your Microchip Serial EEPROM Как получить 10 миллионов циклов стирания/запись последовательной EEPROM памяти фирмы Microchip	AN602
Basic Serial EEPROM Operation Основные операции при работе с последовательной EEPROM памятью	AN536
Everything a System Engineer needs to know about Serial EEPROM Endurance Информация по выносливости последовательной EEPROM памяти для инженеров	AN537
Using the Microchip Endurance Predictive Software Использование программного обеспечения, прогнозирующего выносливость EEPROM памяти	AN562

Раздел 8.Прерывания

Содержание

8.1 Введение

Микроконтроллеры PICmicro среднего семейства могут иметь несколько источников прерываний. Для каждого периферийного модуля назначен отдельный источник прерываний, хотя некоторый периферийный модули содержат несколько источников прерываний (например, модуль USART).

Возможные источники прерываний в микроконтроллерах PICmicro среднего семейства:

- Внешний источник прерываний INT;
- Переполнение таймера TMR0;
- Изменение уровня сигнала на входах PORTB (выводы RB7:RB4);
- Изменение выходного уровня компаратора;
- Прерывание от ведомого параллельного порта;
- Прерывания от USART;
- Прерывание от приемника:
- Прерывание от передатчика;
- Завершение преобразования АЦП;
- Прерывания от LCD;
- Завершение цикла записи в EEPROM память данных;
- Переполнение таймера TMR1;
- Переполнение таймера TMR2;
- Прерывания от модуля CCP;
- Прерывания от модуля SSP.

В микроконтроллерах среднего семейства присутствует как минимум один регистр, управляющий прерываниями. Это регистр:

- INTCON

Если в микроконтроллере есть дополнительные периферийные модули, то в нем будут реализованы регистры для управления прерываниями от периферийных модулей (регистр маски, чтобы разрешить/запретить прерывания; регистр флагов прерываний, указывающий на возникшее прерывание). В зависимости от типа микроконтроллера в нем могут быть реализованы регистры:

- PIE1
- PIR1
- PIE2
- PIR2

В этой документации мы будем обращаться к этим регистрам как PIR и PIE. Если новые микроконтроллеры будут содержать больше источников прерываний, то будут реализованы регистры PIR3 и PIE3.

Регистр управления прерываниями INTCON содержит индивидуальные биты флагов прерываний для ядра микроконтроллера, биты маски разрешения прерываний, а также бит глобального разрешения прерываний.

Если бит глобального разрешения прерываний GIE (INTCON<7>) установлен в '1', то разрешены все немаскированные прерывания. Все прерывания запрещены, если GIE (INTCON<7>) сброшен в '0'. Прерывания индивидуально запрещены сбросом соответствующего бита в регистре INTCON. При сбросе микроконтроллера бит GIE сбрасывается в '0'.

Возврат из обработки прерываний выполняется по команде RETFIE, при этом происходит установка бита GIE в '1', что позволяет обработать любое отложенное прерывание.

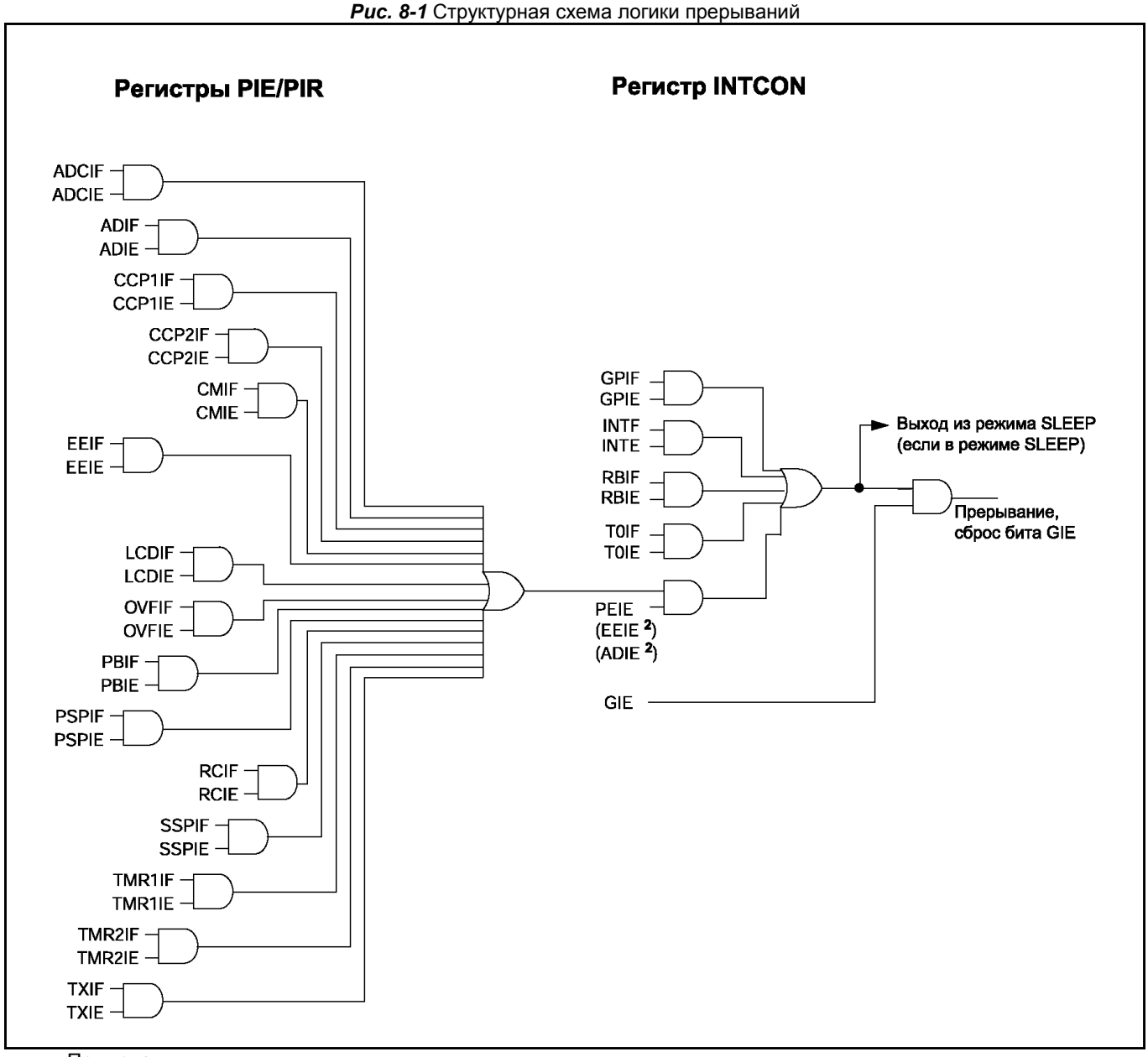
Регистр INCON содержит биты управления следующими прерываниями: внешнее прерывание INT; изменение сигнала на входах RB7:RB4; переполнение TMR0. В регистре INCON также расположен бит разрешения прерываний от периферийных модулей PEIE. Если PEIE=1, то разрешен переход по вектору прерываний при возникновении периферийного прерывания.

При обработке прерываний бит GIE=0, чтобы предотвратить повторную загрузку счетчика команд PC в стек и запись в PC адреса вектора прерываний 0004h. В обработчике прерываний источник прерываний может быть идентифицирован проверкой флагов прерываний. Как правило флаги прерываний должны быть сброшены в обработчике прерываний перед разрешением прерываний в системе, чтобы предотвратить повторный переход на обработку прерываний. Индивидуальные флаги прерываний устанавливаются независимо от состояния бита общего разрешения прерываний GIE и соответствующих битов маски.

Примечание 1. Индивидуальные флаги прерываний устанавливаются независимо от состояния бита общего разрешения прерываний GIE и соответствующих битов маски.

Примечание 2. При выполнении команды, сбрасывающей бит GIE в '0', любое прерывание, ожидающее выполнения в следующем машинном цикле, игнорируется. Микроконтроллер выполнит пустой цикл NOP поле команды, сбрасывающей бит GIE в '0'. Игнорированные прерывания ставятся в ожидание выполнения, пока бит GIE не будет установлен в '1'.

Рис. 8-1 Структурная схема логики прерываний



Примечания:

1. На рисунке показаны все возможные источники прерываний для микроконтроллеров PICmicro среднего семейства. Наличие управляющих битов в микроконтроллере зависит от реализованных периферийных модулей. Смотрите техническую документацию на микроконтроллер.
2. Часть микроконтроллеров среднего семейства имеют только один периферийный модуль. В этих микроконтроллерах нет бита PEIE, а реализован бит разрешения прерываний от периферийного модуля в регистре INTCON.

8.2 Регистры управления

Как правило микроконтроллеры среднего семейства имеют три регистра управления прерываниями. Регистр INTCON содержит бит глобального разрешения прерываний GIE, а также бит разрешения прерываний от периферийных модулей PEIE. В паре регистров PIE/PIR размещаются индивидуальные биты разрешения прерываний от периферийных модулей и флаги возникшего прерывания.

8.2.1 Регистр INTCON

Регистр INTCON доступен для записи и чтения. В этом регистре содержатся различные биты разрешений и флагов прерываний.

Примечание. Флаги прерываний устанавливаются при возникновении условий прерываний вне зависимости от соответствующих битов разрешения и бита общего разрешения прерываний GIE (INTCON<7>). Это позволяет выполнять программный контроль возникновения условия прерываний.

Регистр INTCON

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE	**PEIE**(3)	**T0IE**	**INTE**(2)	**RBIE**(1,2)	**T0IF**	**INTF**(2)	**RBIF**(1,2)
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **GIE**: Глобальное разрешение прерываний
1 = разрешены все немаскированные прерывания
0 = все прерывания запрещены

бит 6: **PEIE**: Разрешение прерываний от периферийных модулей
1 = разрешены все немаскированные прерывания периферийных модулей
0 = прерывания от периферийных модулей запрещены

бит 5: **T0IE**: Разрешение прерывания по переполнению TMR0
1 = прерывание разрешено
0 = прерывание запрещено

бит 4: **INTE**: Разрешение внешнего прерывания INT
1 = прерывание разрешено
0 = прерывание запрещено

бит 3: **RBIE**(1): Разрешение прерывания по изменению сигнала на входах RB7:RB4 PORTB
1 = прерывание разрешено
0 = прерывание запрещено

бит 2: **T0IF**: Флаг прерывания по переполнению TMR0
1 = произошло переполнение TMR0 (сбрасывается программно)
0 = переполнения TMR0 не было

бит 1: **INTF**: Флаг внешнего прерывания INT
1 = выполнено условие внешнего прерывания на выводе RB0/INT (сбрасывается программно)
0 = внешнего прерывания не было

бит 0: **RBIF**(1): Флаг прерывания по изменению уровня сигнала на входах RB7:RB4 PORTB
1 = зафиксировано изменение уровня сигнала на одном из входов RB7:RB4 (сбрасывается программно)
0 = не было изменения уровня сигнала ни на одном из входов RB7:RB4

Примечание 1. В некоторых микроконтроллерах бит RBIE может быть заменен битом GPIE, а бит RBIF может быть заменен битом GPIF.
Примечание 2. Некоторые микроконтроллеры могут не содержать эту функцию.
Примечание 3. В микроконтроллерах с одним периферийным модулем этот бит может быть EEIE или ADIE.

8.2.2 Регистры PIE

В зависимости от числа источников прерываний периферийных модулей в микроконтроллере могут содержаться регистры разрешения периферийных прерываний PIE1 и PIE2. В этих регистрах располагаются индивидуальные биты разрешения прерываний от периферийных модулей. В данной документации эти регистры будут обозначаться как PIE. Если в микроконтроллере присутствует регистр PIE, то необходимо установить в '1' бит PEIE для разрешения прерываний от периферийных модулей.

Примечание. Чтобы разрешить любое периферийное прерывание необходимо установить в '1' бит PEIE(INTCON<6>).

Расположение битов в регистрах PIE стандартизовано, однако во вновь разрабатываемых микроконтроллерах оно может измениться. Не будет возникать проблем с разрядным размещением битов в управляющих регистрах, если Вы будете использовать дополнительный файл от Microchip Inc. с символьным обозначением битов. Это позволит ассемблеру выполнить компиляцию исходного текста программы с правильным указанием адреса регистра и номера бита.

Регистры PIE

R/W-0
(Примечание 1)
Бит 7 Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит: **TMR1IE**: Разрешение прерывания по переполнению TMR1
1 = прерывание разрешено
0 = прерывание запрещено

бит: **TMR2IE**: Разрешение прерывания по переполнению TMR2
1 = прерывание разрешено
0 = прерывание запрещено

бит: **CCP1IE**: Разрешение прерывания от модуля CCP1
1 = прерывание разрешено
0 = прерывание запрещено

бит: **CCP2IE**: Разрешение прерывания от модуля CCP2
1 = прерывание разрешено
0 = прерывание запрещено

бит: **SSPIE**: Разрешение прерывания от модуля синхронного последовательного порта
1 = прерывание разрешено
0 = прерывание запрещено

бит: **RCIE**: Разрешение прерывания от приемника USART
1 = прерывание разрешено
0 = прерывание запрещено

бит: **TXIE**: Разрешение прерывания от передатчика USART
1 = прерывание разрешено
0 = прерывание запрещено

бит: **ADIE**: Разрешение прерывания по окончании преобразования АЦП
1 = прерывание разрешено
0 = прерывание запрещено

бит: **ADCIE**: Разрешение прерывания по окончании преобразования АЦП
1 = прерывание разрешено
0 = прерывание запрещено

бит: **OVFIE**: Разрешение прерывания по переполнению таймера АЦП
1 = прерывание разрешено
0 = прерывание запрещено

бит: **PSPIE**: Разрешение прерывания записи/чтения ведомого параллельного порта
1 = прерывание разрешено
0 = прерывание запрещено

бит: **EEIE**: Разрешение прерывания по окончании записи в EEPROM память данных
1 = прерывание разрешено
0 = прерывание запрещено

бит: **LCDIE**: Разрешение прерывания от модуля LCD
1 = прерывание разрешено
0 = прерывание запрещено

бит: **CMIE**: Разрешение прерывания от модуля компараторов
1 = прерывание разрешено
0 = прерывание запрещено

Примечание 1. Размещение битов в управляющих регистрах смотрите в технической документации на микроконтроллер.

8.2.3 Регистры PIR

В зависимости от числа источников прерываний периферийных модулей в микроконтроллере могут содержаться регистры флагов прерываний от периферийных модулей PIR1 и PIR2. В этих регистрах располагаются индивидуальные биты флагов прерываний от периферийных модулей. В данной документации эти регистры будут обозначаться как PIR.

Примечание 1. Флаги прерываний устанавливаются при возникновении условий прерываний вне зависимости от соответствующих битов разрешения и бита общего разрешения прерываний GIE (INTCON<7>).
Примечание 2. Программное обеспечение пользователя должно сбрасывать соответствующие флаги прерываний в '0' перед разрешением прерывания и в подпрограмме обработки прерывания.

Расположение битов в регистрах PIR стандартизовано, однако во вновь разрабатываемых микроконтроллерах оно может измениться. Не будет возникать проблем с разрядным размещением битов в управляющих регистрах, если Вы будете использовать дополнительный файл от Microchip Inc. с символьным обозначением битов. Это позволит ассемблеру выполнить компиляцию исходного текста программы с правильным указанием адреса регистра и номера бита.

Регистры PIR

R/W-0
(Примечание 1)
Бит 7 Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит: **TMR1IF**: Флаг прерывания по переполнению TMR1
1 = произошло переполнение TMR1 (сбрасывается программно)
0 = переполнения TMR1 не было

бит: **TMR2IF**: Флаг прерывания по переполнению TMR2
1 = произошло переполнение TMR2 (сбрасывается программно)
0 = переполнения TMR2 не было

бит: **CCP1IF**: Флаг прерывания от модуля CCP1
Режим захвата
1 = выполнен захват значения TMR1 (сбрасывается программно)
0 = захвата значения TMR1 не происходило
Режим сравнения
1 = значение TMR1 достигло указанного в регистрах CCPR1H:CCPR1L(сбрасывается программно)
0 = значение TMR1 не достигло указанного в регистрах CCPR1H:CCPR1L
ШИМ режим
Не используется

бит: **CCP2IF**: Флаг прерывания от модуля CCP2
Режим захвата
1 = выполнен захват значения TMR1 (сбрасывается программно)
0 = захвата значения TMR1 не происходило
Режим сравнения
1 = значение TMR1 достигло указанного в регистрах CCPR2H:CCPR2L(сбрасывается программно)
0 = значение TMR1 не достигло указанного в регистрах CCPR2H:CCPR2L
ШИМ режим
Не используется

бит: **SSPIF**: Флаг прерываний от модуля SSP
1 = выполнено условие возникновения прерывания от модуля SSP (сбрасывается программно)
0 = условие возникновения прерывания от модуля SSP не выполнено

Примечание 1. Размещение битов в управляющих регистрах смотрите в технической документации на микроконтроллер.

Регистры PIR (продолжение)

R/W-0
(Примечание 1)
Бит 7 ... Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит: **RCIF**: Флаг прерывания от приемника USART
1 = буфер приемника USART полон (сбрасывается чтением регистра RCREG)
0 = буфер приемника USART пуст

бит: **TXIF**: Флаг прерывания от передатчика USART
1 = буфер передатчика USART пуст (сбрасывается записью в регистр TXREG)
0 = буфер передатчика USART полон

бит: **ADIF**: Флаг прерывания от модуля АЦП
1 = преобразование АЦП завершено (сбрасывается программно)
0 = преобразование АЦП не завершено

бит: **ADCIF**: Флаг прерывания от модуля АЦП
1 = преобразование АЦП завершено (сбрасывается программно)
0 = преобразование АЦП не завершено

бит: **OVFIF**: Флаг прерывания по переполнению таймера АЦП
1 = произошло переполнение таймера АЦП (сбрасывается программно)
0 = переполнение таймера АЦП не происходило

бит: **PSPIF**: Флаг прерывания от ведомого параллельного порта
1 = произошла операция чтения или записи (сбрасывается программно)
0 = операции чтения или записи не происходило

бит: **EEIF**: Флаг прерывания по окончании записи в EEPROM данных
1 = запись в EEPROM данных завершена (сбрасывается программно)
0 = запись в EEPROM данных не завершена или не была начата

бит: **LCDIF**: Флаг прерывания от модуля LCD
1 = возникло прерывание от модуля LCD (сбрасывается программно)
0 = прерывания от модуля LCD не было

бит: **CMIF**: Флаг прерывания от модуля компараторов
1 = возникло прерывание от модуля компараторов (сбрасывается программно)
0 = прерывания от модуля компараторов не было

Примечание 1. Размещение битов в управляющих регистрах смотрите в технической документации на микроконтроллер.

8.3 Время перехода на обработку прерываний

Временем переход на обработку прерываний считается интервал времени от установки флага возникшего прерывания до момента начала выполнения команды по адресу 0004h в памяти программ (если прерывание разрешено).

Для синхронных прерываний (внутренних) время перехода равно $3T_{CY}$.

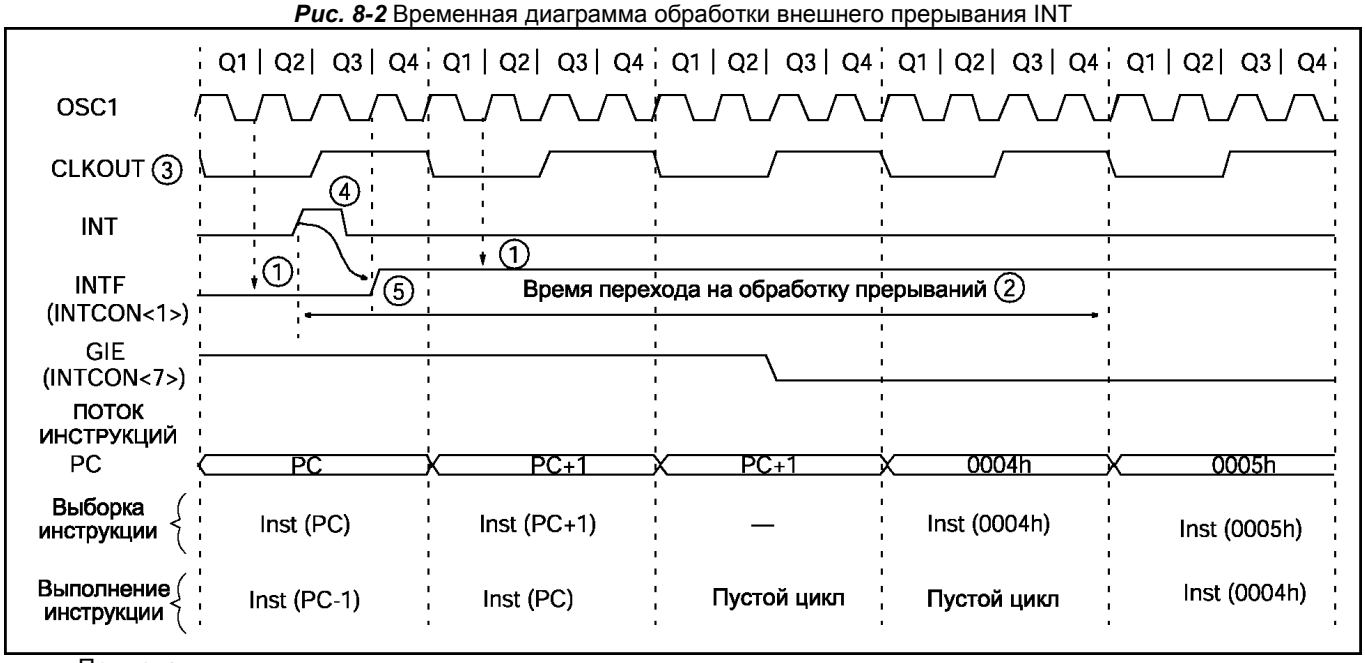
Для асинхронных прерываний (внешних), например внешнее прерывание INT или изменение уровня сигнала на входах RB7:RB4, время перехода на обработку прерываний будет составлять 3 - $3.75T_{CY}$. Точное время перехода на обработку прерываний зависит от момента возникновения прерывания (см. рисунок 8-2).

Время перехода на обработку прерываний одинаково для одно и двух цикловых команд.

8.4 Внешние прерывание INT

Внешнее прерывание с входа RB0/INT происходит: по переднему фронту сигнала, если бит INTEDG (OPTION_REG<6>) установлен в ‘1’; по заднему фронту сигнала, если бит INTEDG сброшен в ‘0’. Когда активный фронт сигнала появляется на входе RB0/INT, бит INTF (INTCON<1>) устанавливается в ‘1’. Прерывание может быть запрещено сбросом бита INTE (INTCON<4>) в ‘0’. Флаг прерывания INTF должен быть сброшен программно в подпрограмме обработки прерываний. Прерывание INT может вывести микроконтроллер из режима SLEEP, если бит INTE=1 до перехода в режим SLEEP. Состояние бита GIE определяет: переходить ли на подпрограмму обработки прерываний после выхода из режима SLEEP. Дополнительную информацию смотрите в разделах, описывающих сторожевой таймер WDT и режим энергосбережения SLEEP.

Рис. 8-2 Временная диаграмма обработки внешнего прерывания INT



Примечания:

1. Флаг INTF проверяется в такте Q1.
2. Время перехода на обработку прерываний: не синхронизированный сигнал $3\text{-}4T_{CY}$; синхронизированный сигнал $3T_{CY}$. Время перехода не зависит от выполняемой инструкции (одно или двух цикловая команда).
3. CLKOUT доступен только в RC режиме генератора.
4. Минимальную длительность импульса INT смотрите в разделе "Электрические характеристики".
5. Флаг INTF может быть установлен в любой момент (Q1-Q4).

8.5 Сохранение контекста

При переходе на подпрограмму обработки прерываний в стеке аппаратно сохраняется только адрес возврата. Как правило, дополнительно необходимо сохранять ключевые регистры (например W, STATUS), что выполняется программно.

Операция сохранения значения регистров обычно обозначается 'PUSH", а восстановление значения регистров обозначается "POP". Обратите внимание, что PUSH, POP не являются мнемоникой команд, а лишь обозначают действие, которое может быть выполнено последовательностью команд. Для упрощения текста программы можно эти сегменты кода программы представить в виде макросов (описание использования макрокоманд смотрите в документации "Руководство пользователя MPASM").

В примере 8-1 показано восстановление регистров STATUS, W для микроконтроллеров с общим ОЗУ (например, PIC16C77). Регистр W_TEMP должен быть определен во всех банках памяти с одинаковым смещением относительно начала банка. Регистр STATUS_TEMP может быть определен в одном банке памяти данных. В примере 8-1 регистр STATUS_TEMP определен в банке 0.

Последовательность операций примера 8-1:

1. Сохранить регистр W независимо от текущего банка памяти.
2. Сохранить регистр STATUS в банке 0.
3. Выполнить подпрограмму обработки прерываний.
4. Восстановить регистр STATUS и текущий банк памяти данных.
5. Восстановить регистр W.

Если необходимо сохранить и другие регистры, то сохранение нужно выполнять после сохранения регистра STATUS (шаг 2), а восстановление перед восстановлением STATUS (шаг 4).

Пример 8-1 Сохранение регистров STATUS, W в ОЗУ (для микроконтроллеров с общим ОЗУ)

```
MOVWF     W_TEMP              ; Копировать W во временный регистр
                              ; независимо от текущего банка
SWAPF     STATUS,W            ; Обменять полубайты в регистре STATUS
                              ; и записать в W
MOVWF     STATUS_TEMP         ; Сохранить STATUS во временном регистре
                              ; банка 0
:
: (Выполнить код подпрограммы обработки прерываний )
:
SWAPF     STATUS_TEMP,W       ; Обменять полубайты оригинального значения STATUS
                              ; и записать в W (восстановить текущий банк)
MOVWF     STATUS              ; Восстановить значение STATUS
                              ; из регистра W
SWAPF     W_TEMP,F            ; Обменять полубайты в регистре W_TEMP и сохранить
                              ; результат в W_TEMP
SWAPF      W_TEMP,W           ; Обменять полубайты в регистре W_TEMP и восстановить
                              ; оригинальное значение W без воздействия на STATUS
```

В примере 8-2 показано восстановление регистров STATUS, W для микроконтроллеров без общего ОЗУ (например PIC16C74A). Регистр W_TEMP должен быть определен во всех банках памяти с одинаковым смещением относительно начала банка. Регистр STATUS_TEMP может быть определен в одном банке памяти данных. В примере 8-2 регистр STATUS_TEMP определен в банке 0.

Последовательность операций примера 8-2:

1. Сохранить регистр W независимо от текущего банка памяти.
2. Сохранить регистр STATUS в банке 0.
3. Выполнить подпрограмму обработки прерываний.
4. Восстановить регистр STATUS и текущий банк памяти данных.
5. Восстановить регистр W.

Если необходимо сохранить и другие регистры, то сохранение нужно выполнять после сохранения регистра STATUS (шаг 2), а восстановление перед восстановлением STATUS (шаг 4).

Пример 8-2 Сохранение регистров STATUS, W в ОЗУ (для микроконтроллеров без общего ОЗУ)

```
MOVWF     W_TEMP              ; Копировать W во временный регистр
                              ; независимо от текущего банка
SWAPF     STATUS,W            ; Обменять полубайты в регистре STATUS
                              ; и записать в W
BCF       STATUS,RP0          ; Выбрать банк 0
MOVWF     STATUS_TEMP         ; Сохранить STATUS во временном регистре
                              ; банка 0
:
: (Выполнить код подпрограммы обработки прерываний )
:
SWAPF     STATUS_TEMP,W       ; Обменять полубайты оригинального значения STATUS
                              ; и записать в W (восстановить текущий банк)
MOVWF     STATUS              ; Восстановить значение STATUS
                              ; из регистра W
SWAPF     W_TEMP,F            ; Обменять полубайты в регистре W_TEMP и сохранить
                              ; результат в W_TEMP
SWAPF      W_TEMP,W           ; Обменять полубайты в регистре W_TEMP и восстановить
                              ; оригинальное значение W без воздействия на STATUS
```

В примере 8-3 показано восстановление регистров STATUS, W для микроконтроллеров с универсальным ОЗУ только в банке 0 (например PIC16C620). Банк памяти должен быть проверен перед сохранением любого регистра. Регистр W_TEMP должен быть определен во всех банках памяти с одинаковым смещением относительно начала банка. Регистр STATUS_TEMP может быть определен в одном банке памяти данных. В примере 8-3 регистр STATUS_TEMP определен в банке 0.

Последовательность операций примера 8-3:

1. Проверить текущий банк.
2. Сохранить регистр W независимо от текущего банка памяти.
3. Сохранить регистр STATUS в банке 0.
4. Выполнить подпрограмму обработки прерываний.
5. Восстановить регистр STATUS и текущий банк памяти данных.
6. Восстановить регистр W.

Если необходимо сохранить и другие регистры, то сохранение нужно выполнять после сохранения регистра STATUS (шаг 3), а восстановление перед восстановлением STATUS (шаг 5).

Пример 8-3 Сохранение регистров STATUS, W в ОЗУ (для микроконтроллеров с универсальным ОЗУ, только в банке 0)

```
Push
            BTFSS     STATUS, RP0          ; В банке 0?
            GOTO      RP0CLEAR             ; Да
            BCF       STATUS, RP0          ; Нет
            MOVWF     W_TEMP               ; Сохранить регистр W
            SWAPF     STATUS, W            ; Обменять полубайты в регистре STATUS
            MOVWF     STATUS_TEMP          ; и записать в STATUS_TEMP
            BSF       STATUS_TEMP, 5       ; Установить бит RP0 в сохраненном значении STATUS
            GOTO      ISR_Code             ; Сохранение регистров завершено
RP0CLEAR
            MOVWF     W_TEMP               ; Сохранить регистр W
            SWAPF     STATUS, W            ; Обменять полубайты в регистре STATUS
            MOVWF     STATUS_TEMP          ; и записать в STATUS_TEMP
                                           ;
ISR_Code
            :
            : (Выполнить код подпрограммы обработки прерываний )
            :
Pop
            SWAPF     STATUS_TEMP, W       ; Восстановить значение STATUS
            MOVWF     STATUS               ;
            BTFSS     STATUS, RP0          ; Банк 1?
            GOTO      Restore_WREG         ; Нет
            BCF       STATUS, RP0          ; Да
            SWAPF     W_TEMP, F            ; Восстановить значение регистра W
            SWAPF     W_TEMP, W            ;
            BSF       STATUS, RP0          ; Восстановить банк 1
            RETFIE                         ; Восстановление регистров завершено
Restore_WREG
            SWAPF W_TEMP, F                ; Восстановить значение регистра W
            SWAPF W_TEMP, W        ;
            RETFIE                         ; Восстановление регистров завершено
```

8.6 Инициализация

В примере 8-4 показана инициализация прерываний, где PIE1_MASK - значение, записываемое в регистр маски периферийных прерываний.

Создание макрокоманд сохранения/восстановления значений регистров показано в примере 8-5. Макрокоманды должны быть определены прежде, чем они будут использоваться. Для простоты отладки текста программы макрокоманды рекомендуется помещать в отдельные файлы, включаемые в исходный файл программы, до применения макрокоманды. Рекомендуется включать файлы с макрокомандами в начале исходного файла (см. пример 8-6).

В примере 8-7 представлена типовая структура проверки возникшего прерывания. В этом примере используются макрокоманды для сохранения значения регистров перед выполнением кода обработки прерываний.

Пример 8-4 Инициализация прерываний

```
PIE1_MASK1   EQU B'01101010'        ; Значение для регистра
             :                      ; маски прерываний
             :
             CLRF    STATUS         ; Банк 0
             CLRF    INTCON         ; Выключить прерывания и сбросить флаги
             CLRF    PIR1           ; Сбросить все флаги
             BSF     STATUS, RP0    ; Банк 1
             MOVLW   PIE1_MASK1     ; Записать маску прерываний в регистр PIE1
             MOVWF   PIE1           ;
             BCF     STATUS, RP0    ; Банк 0
             BSF     INTCON, GIE    ; Включить прерывания
```

Пример 8-5 Макрокоманды сохранения/восстановления значения регистров

```
PUSH_MACRO   MACRO                  ; Макрос сохранения регистров
             MOVWF   W_TEMP         ; Копировать W во временный регистр
                                    ; независимо от текущего банка
             SWAPF   STATUS,W       ; Обменять полубайты в регистре STATUS
                                    ; и записать в W
             MOVWF   STATUS_TEMP    ; Сохранить STATUS во временном регистре
                                    ; банка 0
             ENDM                   ; Конец макроса
                                    ;
POP_MACRO    MACRO                  ; Макрос восстановления регистров
             SWAPF   STATUS_TEMP,W  ; Обменять полубайты оригинального значения STATUS
                                    ; и записать в W (восстановить текущий банк)
             MOVWF   STATUS         ; Восстановить значение STATUS
                                    ; из регистра W
             SWAPF   W_TEMP,F       ; Обменять полубайты в регистре W_TEMP и сохранить
                                    ; результат в W_TEMP
             SWAPF   W_TEMP,W       ; Обменять полубайты в регистре W_TEMP и восстановить
                                    ; оригинальное значение W без воздействия на STATUS
             ENDM                   ; Конец макроса
```

Пример 8-6 Шаблон исходного файла

```
            LIST      p = p16C77          ; Список директив
                                          ;
                                          ;
            #INCLUDE  <P16C77.INC>        ; Дополнительный файл к микроконтроллеру
                                          ;
            #INCLUDE  <MY_STD.MAC>        ; Подключить файл стандартных макрокоманд
            #INCLUDE  <APP.MAC>           ; подключить файл специальных макрокоманд
                                          ; для этого приложения
            ; Определение битов конфигурации
            __CONFIG _XT_OSC & _PWRTE_ON & _BODEN_OFF & _CP_OFF & _WDT_ON
                                          ;
            org 0x00                      ; Начало памяти программ
RESET_ADDR :                              ; Первая выполняемая инструкция после сброса
            end
```

Пример 8-7 Типовая обработка прерываний

```
            org       ISR_ADDR            ;
            PUSH_MACRO                    ; Макрокоманда сохранения регистров,
                                          ; или другой код
            CLRF      STATUS              ; Банк 0
            BTFSC     PIR1, TMR1IF        ; Прерывание от TMR1?
            GOTO      T1_INT              ; Да
            BTFSC     PIR1, ADIF          ; Нет, прерывание от АЦП?
            GOTO      AD_INT              ; Да, от АЦП
            :                             ; Нет, проверка других источников прерываний
            :                             ;
            BTFSC     PIR1, LCDIF         ; Нет, прерывание от LCD?
            GOTO      LCD_INT             ; Да, прерывание от LCD
            BTFSC     INTCON, RBIF        ; Нет, прерывание по изменению сигнала на RB7:RB6?
            GOTO      PORTB_INT           ; Да, прерывание по изменению сигнала на RB7:RB6
INT_ERROR_LP1                             ; Нет, процедура восстановления при ошибке
            GOTO      INT_ERROR_LP1       ; Здесь должна располагаться процедура
                                          ; обработки возникновения неожидаемого
                                          ; прерывания
T1_INT                                    ; Обработка прерываний от TMR1
            :                             ;
            BCF       PIR1, TMR1IF        ; Сброс флага прерывания от TMR1
            GOTO      END_ISR             ; Завершение обработки прерываний
AD_INT                                    ; Обработка прерываний от АЦП
            :                             ;
            BCF       PIR1, ADIF          ; Сброс флага прерывания от АЦП
            GOTO      END_ISR             ; Завершение обработки прерываний
LCD_INT                                   ; Обработка прерываний от LCD
            :                             ;
            BCF       PIR1, LCDIF         ; Сброс флага прерывания от LCD
            GOTO      END_ISR             ; Завершение обработки прерываний
PORTB_INT                                 ; Обработка прерываний по изменению сигнала на RB7:RB6
            :                             ;
END_ISR                                   ;
            POP_MACRO                     ; Макрокоманда восстановления значения регистров
                                          ; или другой код
            RETFIE                        ; Возвращение из обработки прерываний,
                                          ; разрешение прерываний
```

8.7 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Алгоритм программы дает неправильные результаты.
Ответ 1:
При разрешенных прерываниях во время выполнения алгоритма необходимо гарантировать, что регистры, используемые алгоритмом, сохраняются и восстанавливаются в подпрограмме обработки прерываний. Проверьте подпрограмму обработки прерываний, т.к. некоторые регистры могут быть изменены.

Вопрос 2: Выполнение программы прекращается, что может быть причиной?
Ответ 2:
Если в программе используются прерывания, то необходимо следить за тем, чтобы перед выходом из обработки прерываний (выполнения команды RETFIE) был сброшен флаг источника прерываний. Если флаг прерывания останется установленным, то после исполнения команды RETFIE выполнение программы опять перейдет по вектору прерываний и останется невыполненным разрешенное прерывание.

8.8 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с прерываниями в микроконтроллерах PICmicro MCU:

Документ	Номер
Using the PortB Interrupt On Change as an External Interrupt Применение внешнего прерывания и прерывания по изменению сигнала на входах PORTB	AN566

Раздел 9.Порты ввода/вывода

Содержание

9.1 Введение

Универсальные порты ввода/вывода могут рассматриваться как самые простые периферийные модули. Они позволяют микроконтроллерам PICmicro контролировать работу и управлять другими устройствами. С целью расширения функциональных возможностей некоторые каналы портов ввода/вывода мультиплицированы с другими периферийными модулями. Набор дополнительных функций каналов портов ввода/вывода зависит от реализованных периферийных модулей в микроконтроллере. Как правило, при включенном периферийном модуле, соответствующий вывод микроконтроллера не может использоваться как универсальный канал ввода/вывода.

Для большинства каналов портов ввода/вывода регистры TRIS управляют направлением данных на выводе. Бит TRIS<x> управляет направлением данных на канале PORT<x>. Если бит TRIS установлен в '1', то соответствующий канал порта ввода/вывода работает как вход, а если бит TRIS сброшен в '0', то канал ввода/вывода работает как выход. Простой способ запомнить направление канала ввода/вывода и состояние битов регистров TRIS:

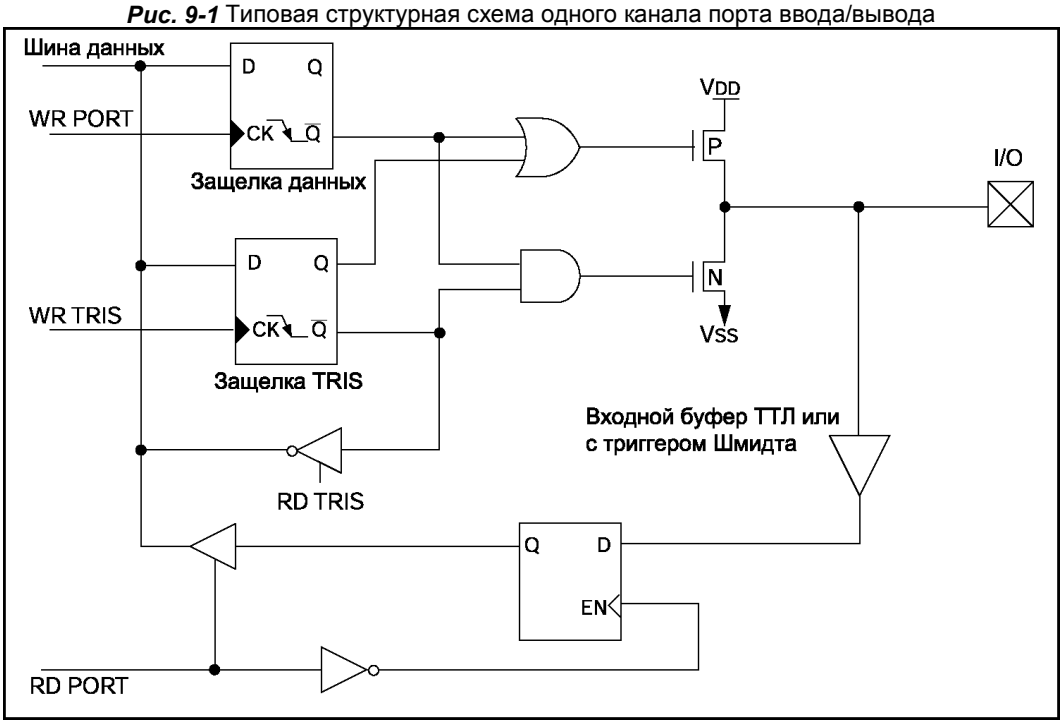
'1' - напоминает 'In' (ввод);

'0' - напоминает 'Out' (выход).

Регистр PORT - защелка данных, выводимых на порт ввода/вывода. При чтении регистра PORT возвращается состояние выводов порта. Это означает, что необходима некоторая осторожность при выполнении команд со структурой "чтение - модификация - запись" для изменения логического уровня на выходах порта.

На рисунке 9-1 показана типовая структурная схема одного канала порта ввода/вывода. На этом рисунке не показана ситуация подключения дополнительного периферийного модуля к каналу порта. Чтение регистра PORT возвращает состояние на выводах порта, а запись выполняется в выходную защелку. Обратите внимание на операции "чтение - модификация - запись" (например, BSF и BCF). Сначала происходит чтение состояния выводов порта, изменение полученного значения, а затем выполняется запись в выходную защелку порта.

Рис. 9-1 Типовая структурная схема одного канала порта ввода/вывода



Примечание. Все выводы портов имеют защитные диоды, подключенные к V_{DD} и V_{SS}.

Когда периферийный модуль подключен к выводу порта, функциональные возможности канала порта ввода/вывода могут измениться в соответствии с требованиями периферийного модуля. Например, модуль АЦП или LCD, которые настраивают соответствующие каналы портов ввода/вывода для работы с периферийным модулем при сборе микроконтроллера. В случае с АЦП это может предотвратить повышенное энергопотребление при подаче аналоговых уровней на входы микроконтроллера после сброса.

При включении некоторых периферийных модулей отменяется действие битов TRIS. Поэтому следует избегать команд "чтение - модификация - запись" с регистрами TRIS (например BSF, BCF, XORWF и т.д.). Обратитесь к описанию соответствующего периферийного модуля для правильной настройки битов TRIS.

Выводы портов могут быть мультиплицированы с аналоговыми входами и входом V_{REF}. Для каждого вывода необходимо определить режим его работы (аналоговый вход или цифровой канал ввода/вывода) настройкой управляющих битов в регистре ADCON1 (регистр управления АЦП). Когда вывод работает как аналоговый вход, то чтение состояния этого вывода будет давать результат '0'.

Регистры TRIS управляет направлением каналов ввода/вывода, даже когда он работает в режиме аналогового входа. Пользователь должен гарантировать, что соответствующий бит TRIS установлен в '1', если вывод используется как аналоговый вход.

Примечание 1. Если выводы мультиплицированы с аналоговыми входами АЦП, то при сбросе микроконтроллера они будут настроены как аналоговые входы (управляется регистром ADCON1). Чтение каналов, настроенных как аналоговый вход, будет давать результат '0'.

Примечание 2. Если выводы мультиплицированы с аналоговыми входами компаратора, то при сбросе микроконтроллера они будут настроены как аналоговые входы (управляется регистром CMCON). Чтение каналов, настроенных как аналоговый вход, будет давать результат '0'.

Примечание 3. Если выводы мультиплицированы с драйверами сегментов LCD, то при сбросе микроконтроллера они будут настроены как драйверы сегментов LCD (управляется регистром LCDSE). Чтобы настроить выводы как цифровые каналы ввода/вывода, необходимо сбросить биты в регистре LCDSE. Установка бита в регистре LCDSE отменяет действие соответствующего бита в регистре TRIS.

Примечание 4. Выводы портов могут быть мультиплицированы с ведомым параллельным портом (PSP). Для работы PSP необходимо установить бит PSPMODE в '1', а соответствующие выводы должны быть настроены как цифровые каналы ввода/вывода.

Примечание 5. В настоящее время ведомый параллельный порт мультиплицирован только с PORTD, PORTE. Порт микропроцессора становится доступным только, когда бит PSPMODE установлен в '1'. В этом режиме биты TRISE должны быть установлены в '1' (выводы настроены как цифровые входы). Для PORTD отменяется действие битов TRISD. В режиме ведомого параллельного порта ко входам PORTD и PORTE подключены буферы ТТЛ. Биты управления PSP расположены в регистре TRISE.

9.2 Регистры PORTA и TRISA

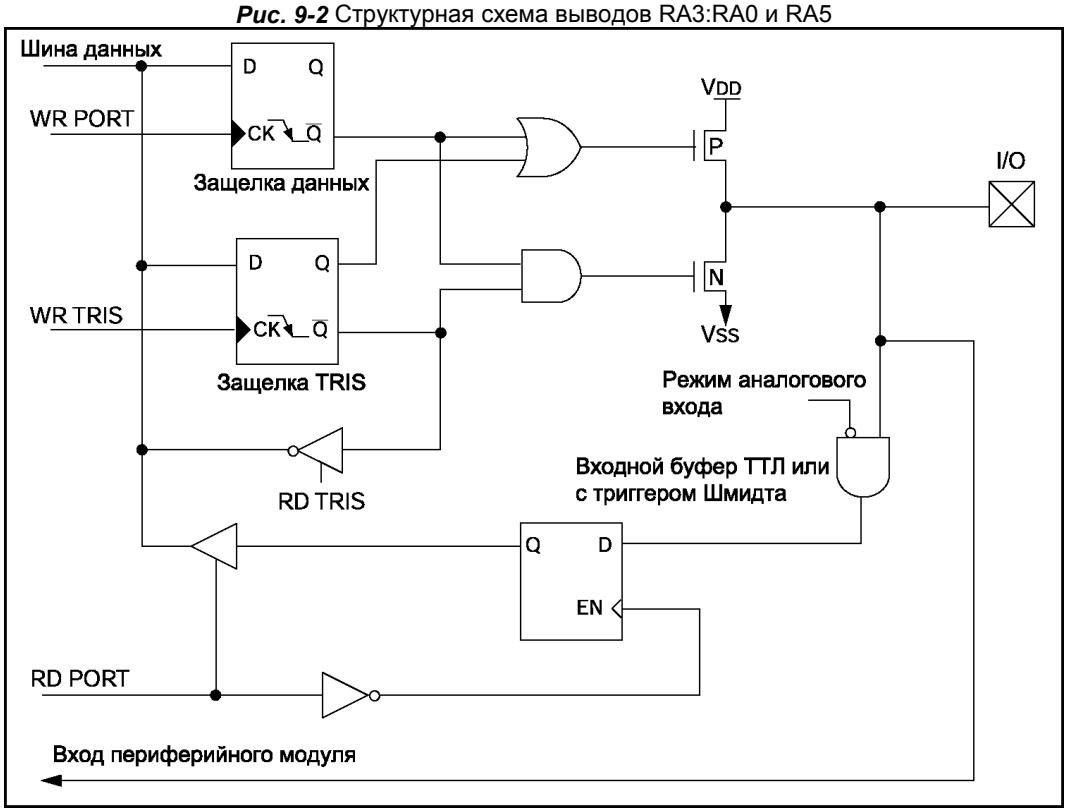
RA4 - имеет триггер Шмидта на входе и открытый сток на выходе. Все остальные каналы PORTA имеют ТТЛ буфер на входе и полнофункциональные выходные КМОП буферы. Все выводы имеют биты управления направления данных в регистре TRISA, с помощью которых можно настроить выводы как входы или выходы.

Запись ‘1’ в TRISA переводит соответствующий выходной буфер в 3-е состояние. Запись ‘0’ в регистр TRISA определяет соответствующий канал как выход, содержимое защелки PORTA передается на вывод микроконтроллера.

Пример 9-1 Инициализация PORTA

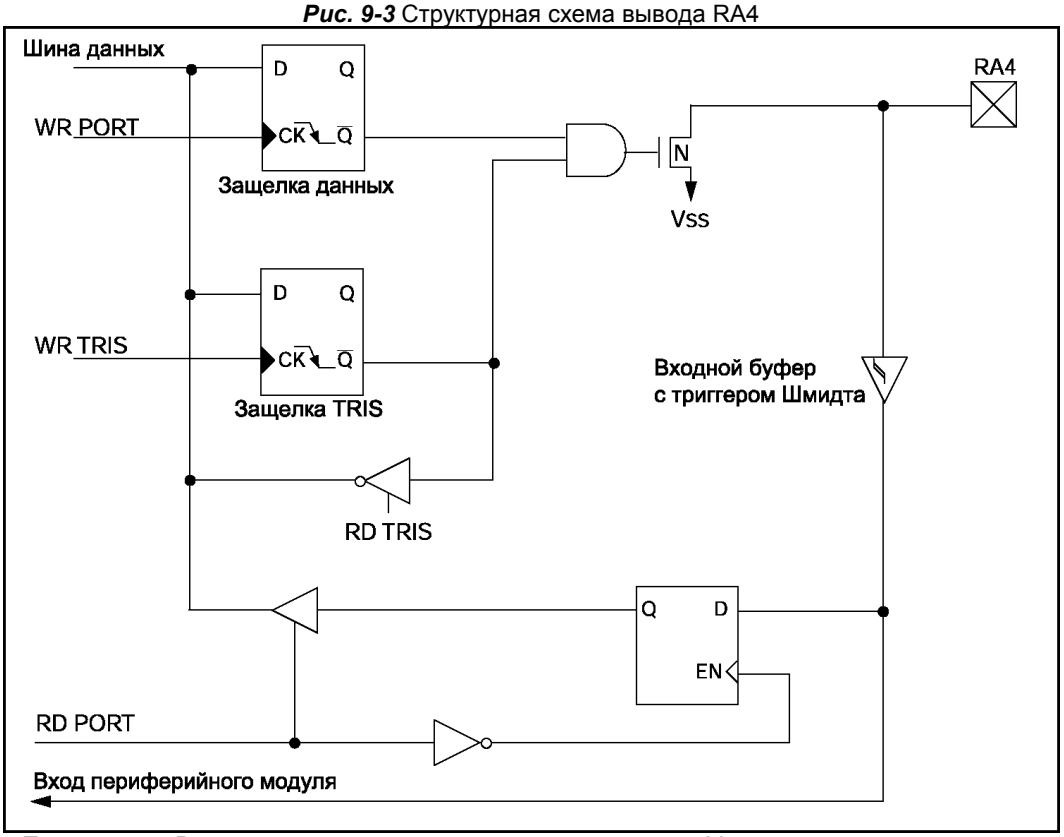
```
BCF     STATUS, RP0   ; Выбрать банк 0
CLRF    PORTA         ; Инициализация защелок PORTA

BSF     STATUS, RP0   ; Выбрать банк 1

MOVLW   0xCF          ; Значение для инициализации
                      ; направления каналов PORTA
MOVWF   TRISA         ; Настроить RA<3:0> как входы,
                      ; настроить RA<5:4> как выходы
                      ; Биты TRISA<7:6> всегда читаются как ’0’.
```

Рис. 9-2 Структурная схема выводов RA3:RA0 и RA5



Примечание. Выводы имеют защитные диоды, подключенные к V_{DD} и V_{SS}.

Рис. 9-3 Структурная схема вывода RA4



Примечание. Вывод имеет защитный диод, подключенный к V_{SS}.

9.3 Регистры PORTB и TRISB

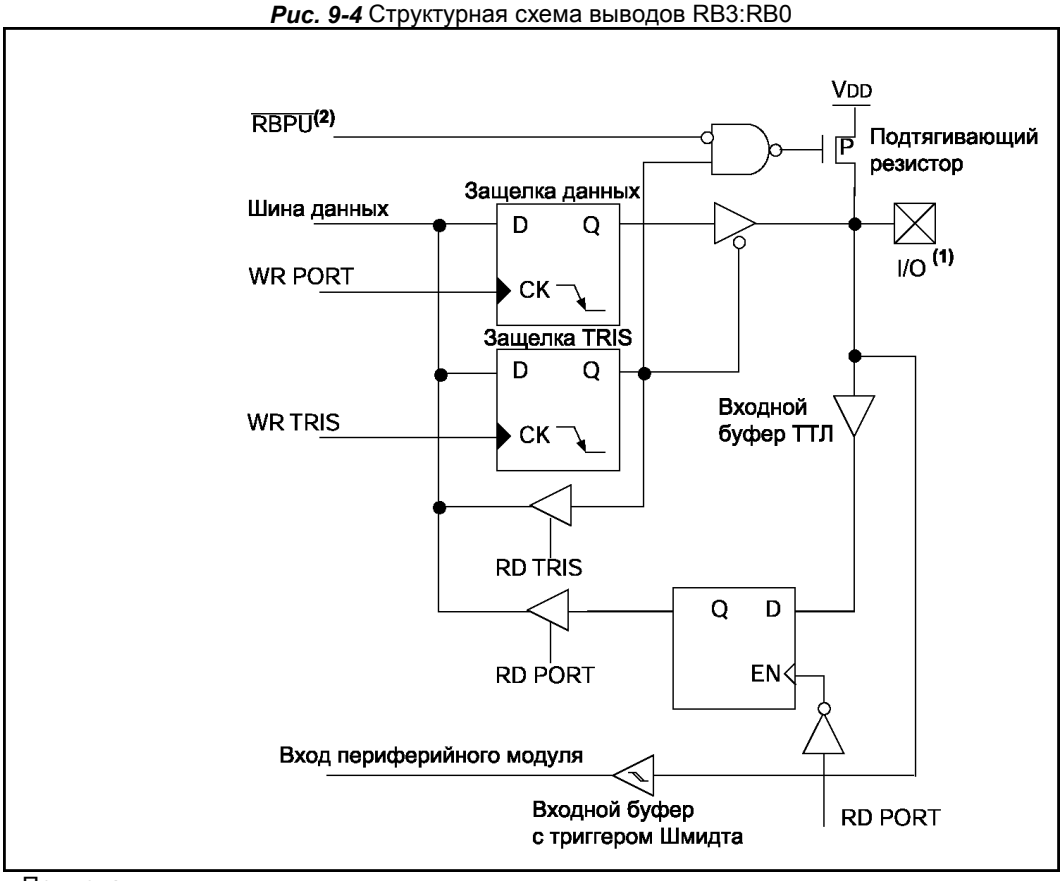
PORTB – 8-разрядный двунаправленный порт ввода/вывода. Биты регистра TRISB определяют направление каналов порта. Установка битва в ’1’ регистра TRISB переводит выходной буфер в 3-е состояние. Запись ‘0’ в регистр TRISB настраивает соответствующий канал как выход, содержимое защелки PORTB передается на вывод микроконтроллера (если выходная защелка подключена к выводу микроконтроллера).

Пример 9-2 Инициализация PORTB

```
BCF     STATUS, RP0   ; Выбрать банк 0
CLRF    PORTB         ; Инициализация защелок PORTB

BSF     STATUS, RP0   ; Выбрать банк 1

MOVLW   0xCF          ; Значение для инициализации
                      ; направления каналов PORTB
MOVWF   TRISB         ; Настроить RB<3:0> как входы,
                      ; RB<5:4> как выходы, RB<7:6> как входы
```

К каждому выводу PORTB подключен внутренний подтягивающий резистор. Бит -RBPU (OPTION_REG <7>) определяет подключены (-RBPU=0) или нет (-RBPU=1) подтягивающие резисторы. Подтягивающие резисторы автоматически отключаются, когда каналы порта настраиваются на выход и после сброса по включению питания POR.

Рис. 9-4 Структурная схема выводов RB3:RB0



Примечания:

1. Выводы имеют защитные диоды, подключенные к V_{DD} и V_{SS}.
2. Для включения подтягивающего резистор необходимо установить бит в регистре TRISB и сбросить бит -RBPU (OPTION_REG <7>).

Четыре канала PORTB RB7:RB4, настроенные на вход, могут генерировать прерывания по изменению логического уровня сигнала на входе. Если один из каналов RB7:RB4 настроен на выход, то он не может быть источником прерываний. Сигнал на выводах RB7:RB4 сравнивается со значением, сохраненным при последнем чтении PORTB. В случае несовпадения одного из значений устанавливается флаг RBIF (INTCON<0>), и если разрешено, генерируется прерывание.

Это прерывание может вывести микроконтроллер из режима SLEEP. В подпрограмме обработки прерываний необходимо сделать следующие действия:

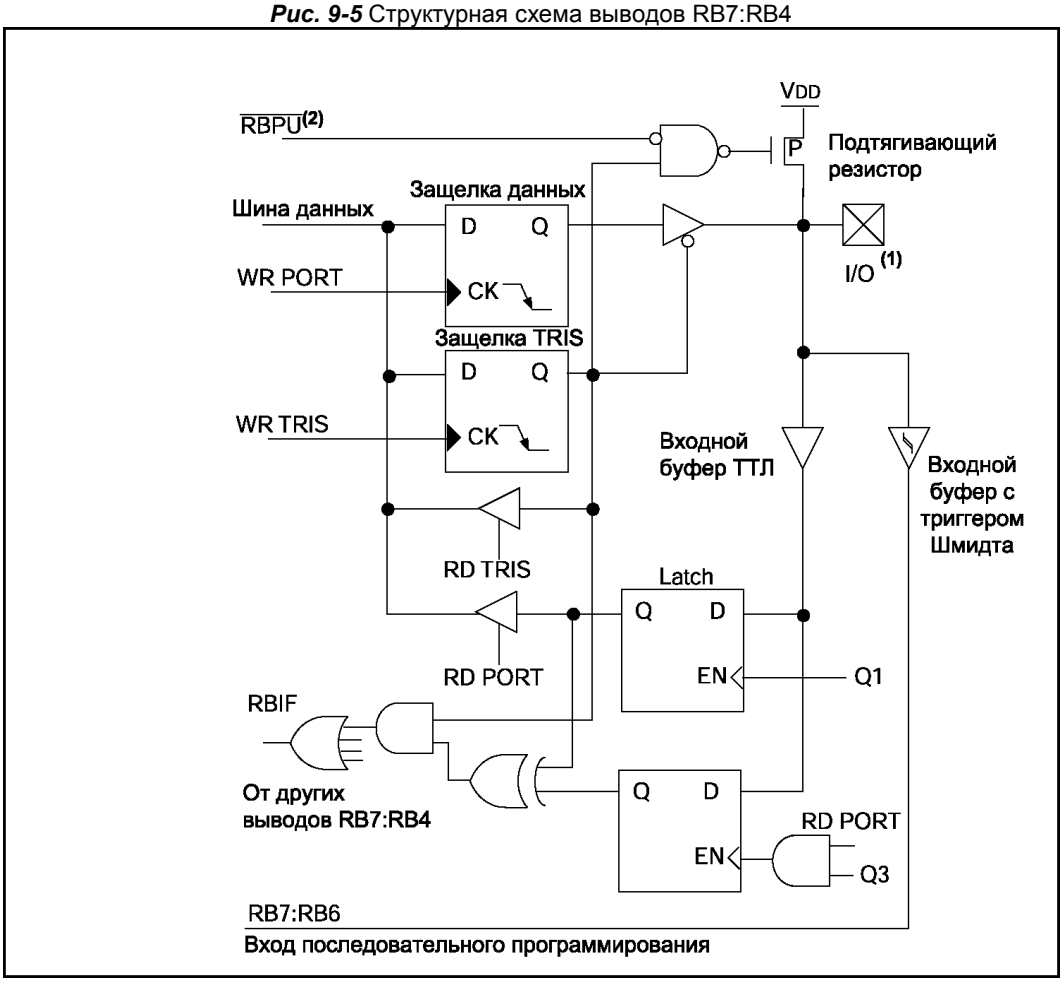
1. Выполнить чтение или запись в PORTB, исключив несоответствие;
2. Сбросить флаг RBIF в ’0’.

Несоответствие сохраненного значения с сигналом на входе PORTB всегда устанавливает бит RBIF в ‘1’. Чтение из PORTB прервет условие несоответствия и позволит сбросить флаг RBIF в ‘0’.

Прерывания по изменению сигнала на входах PORTB и программа переключения конфигурации этих каналов позволяет реализовать простой интерфейс обслуживания клавиатуры с выходом из режима SLEEP по нажатию клавиш.

Прерывания по изменению сигнала на входах рекомендуется использовать для определения нажатия клавиш, когда PORTB полностью задействован для реализации клавиатуры. Не рекомендуется опрашивать PORTB при использовании прерываний по изменению входного сигнала.

Рис. 9-5 Структурная схема выводов RB7:RB4



Примечания:

1. Выводы имеют защитные диоды, подключенные к V_{DD} и V_{SS}.
2. Для включения подтягивающего резистор необходимо установить бит в регистре TRISB и сбросить бит -RBPU (OPTION_REG <7>).
3. В SLEEP режиме микроконтроллер находится на такте Q1.

9.4 Регистры PORTC и TRISC

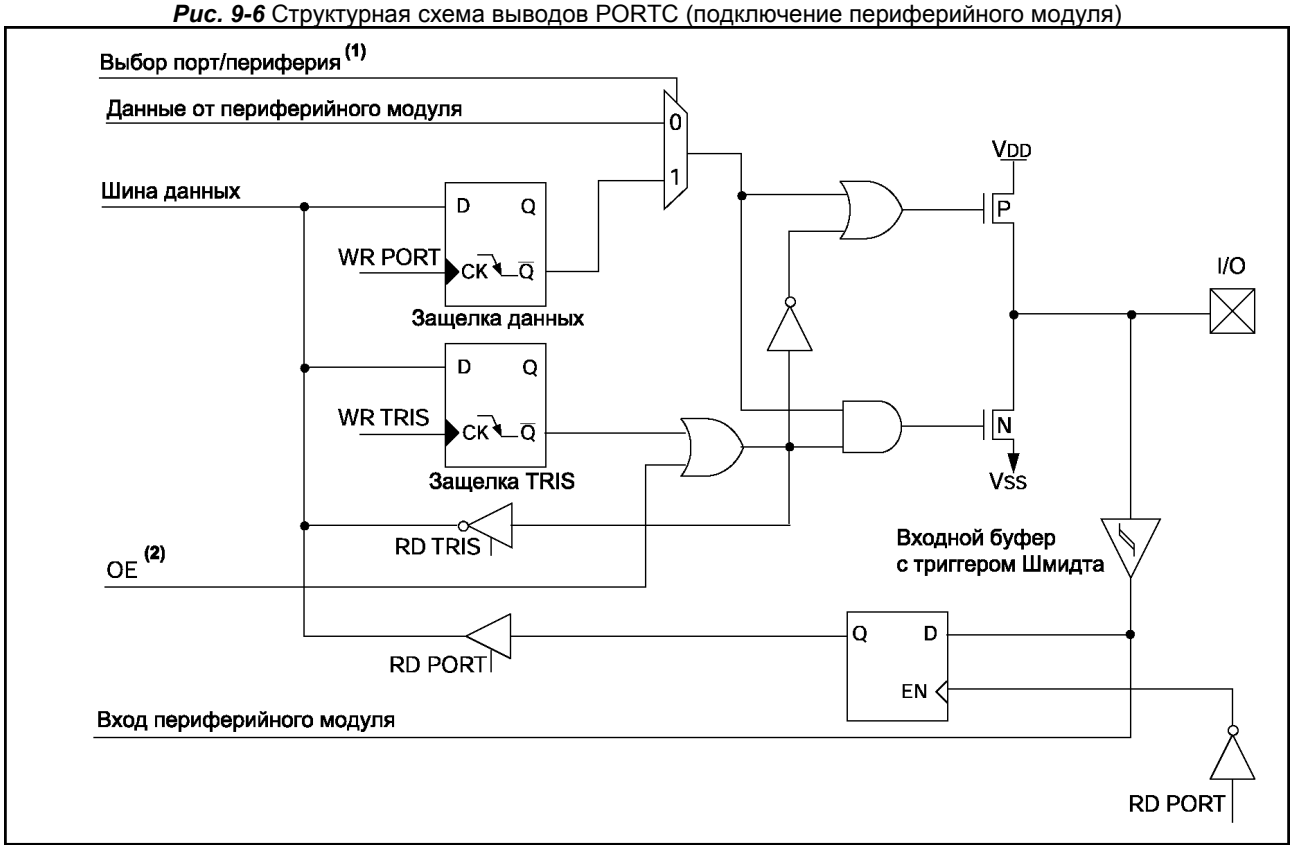
PORTC – 8-разрядный двунаправленный порт ввода/вывода. Биты регистра TRISC определяют направление каналов порта. На каналах PORTC присутствует входной буфер с триггером Шмидта.

При использовании периферийных модулей необходимо соответствующим образом настраивать биты регистра TRISC для каждого вывода. Некоторые периферийные модули отменяют действие битов TRISC, принудительно настраивая вывод на вход или выход.

Пример 9-3 Инициализация PORTC

```
BCF     STATUS, RP0   ; Выбрать банк 0
CLRF    PORTC         ; Инициализация защелок PORTC

BSF     STATUS, RP0   ; Выбрать банк 1

MOVLW   0xCF          ; Значение для инициализации
                      ; направления каналов PORTC
MOVWF   TRISC         ; Настроить RC<3:0> как входы,
                      ; RC<5:4> как выходы, RC<7:6> как входы
```

Рис. 9-6 Структурная схема выводов PORTC (подключение периферийного модуля)



Примечания:

1. Сигнал режима канала – вывод используется периферийным модулем или цифровой порт ввода/вывода.
2. Сигнал разрешения (OE) от периферийного модуля, настраивать канал как выход.
3. Выводы портов имеют защитные диоды, подключенные к V_{DD} и V_{SS}.

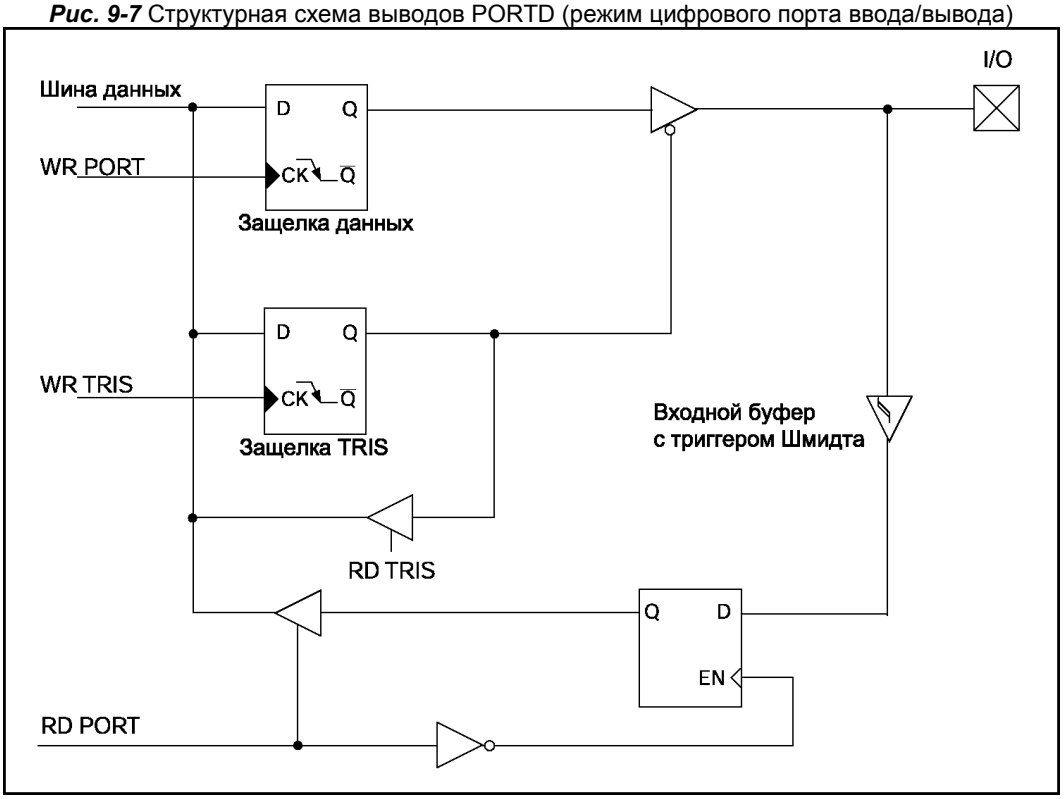
9.5 Регистры PORTD и TRISD

PORTD – 8-разрядный двунаправленный порт ввода/вывода. Биты регистра TRISD определяют направление каналов порта.

Пример 9-4 Инициализация PORTD

```
BCF     STATUS, RP0   ; Выбрать банк 0
CLRF    PORTD         ; Инициализация защелок PORTD

BSF     STATUS, RP0   ; Выбрать банк 1

MOVLW   0xCF          ; Значение для инициализации
                      ; направления каналов PORTD
MOVWF   TRISD         ; Настроить RD<3:0> как входы,
                      ; RD<5:4> как выходы, RD<7:6> как входы
```

Рис. 9-7 Структурная схема выводов PORTD (режим цифрового порта ввода/вывода)



Примечание. Выводы имеют защитные диоды, подключенные к V_{DD} и V_{SS}.

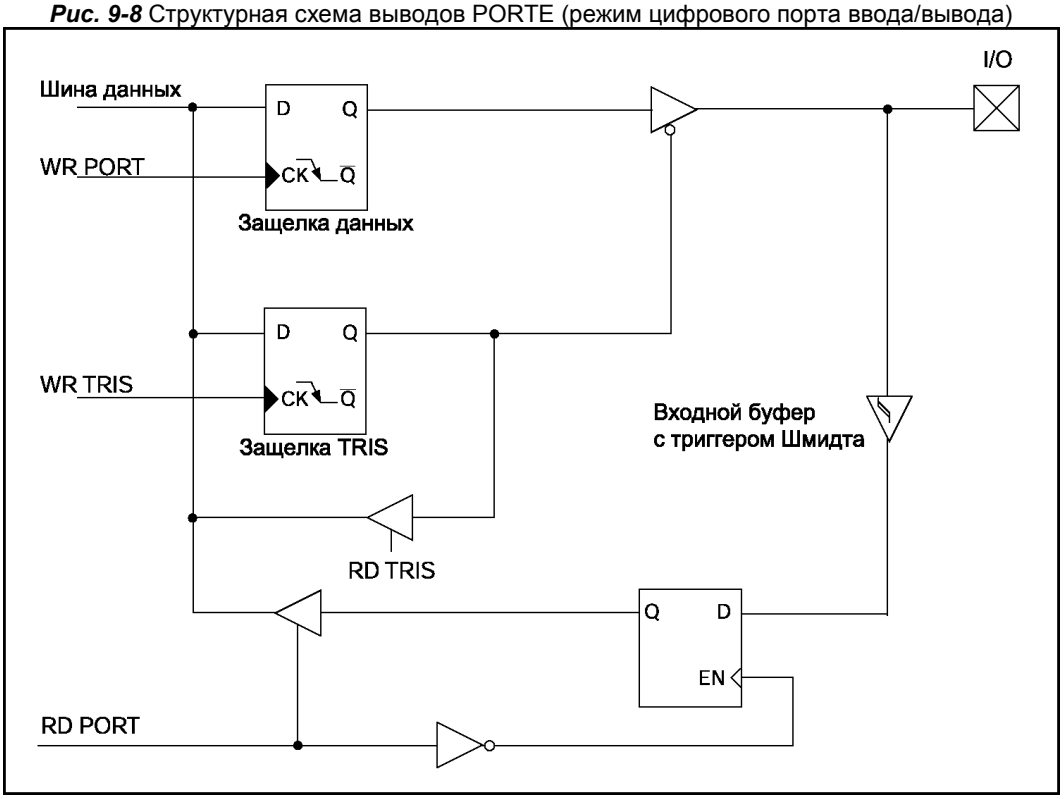
9.6 Регистры PORTE и TRISE

PORTE – 8-разрядный двунаправленный порт ввода/вывода. Биты регистра TRISE определяют направление каналов порта.

Пример 9-5 Инициализация PORTE

```
BCF         STATUS, RP0   ; Выбрать банк 0
CLRF        PORTE         ; Инициализация защелок PORTE

BSF         STATUS, RP0   ; Выбрать банк 1

MOVLW       0x03          ; Значение для инициализации
                          ; направления каналов PORTE
MOVWF       TRISE         ; Настроить RE<1:0> как входы,
                          ; RE<7:2> как выходы
```

Рис. 9-8 Структурная схема выводов PORTE (режим цифрового порта ввода/вывода)



Примечание. Выводы имеют защитные диоды, подключенные к V_{DD} и V_{SS}.

Примечание. В некоторых микроконтроллерах с PORTE старшие биты регистра TRISE используются для управления ведомым параллельным портом (PSP).

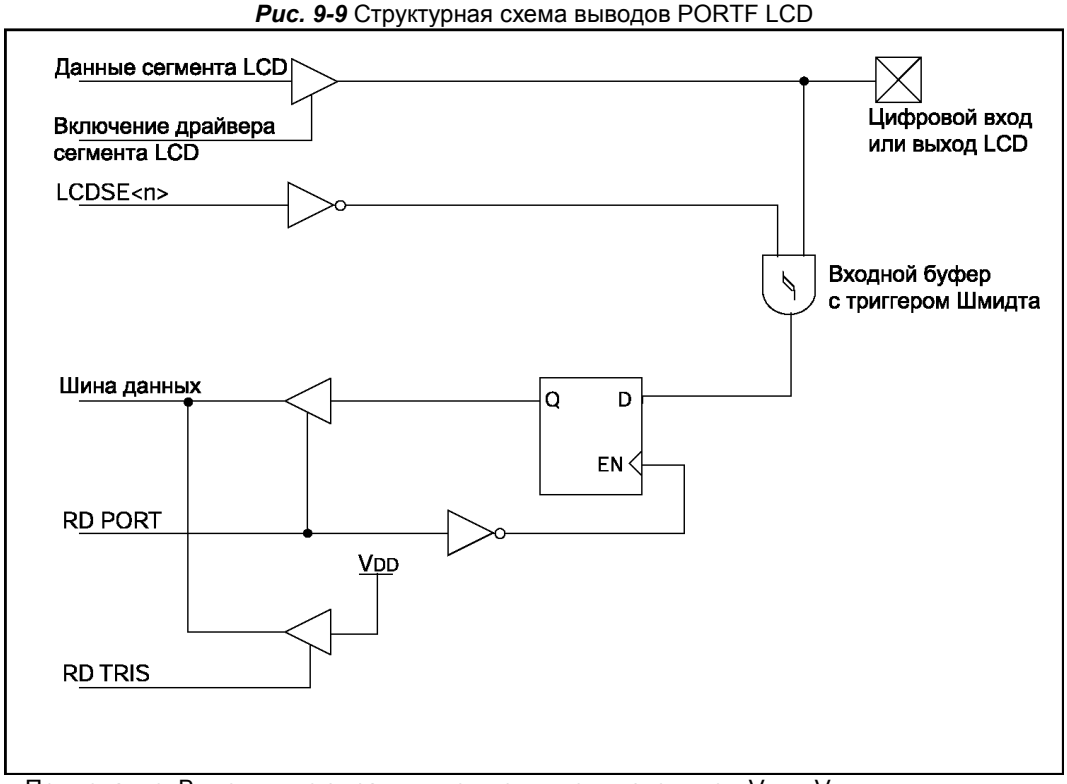
9.7 Регистры PORTF и TRISF

PORTF – 8-разрядный цифровой порт, работающий только как вход. Каждый вывод порта мультиплицирован с драйвером сегмента LCD. К выводам порта подключены входные буферы с триггером Шмидта.

Пример 9-6 Инициализация PORTF

```
BCF         STATUS, RP0         ; Выбрать банк 2
BSF         STATUS, RP1         ;
BCF         LCDSE,SE16          ; Настроить все выводы PORTF
BCF         LCDSE,SE12          ; как цифровые входы
```

Рис. 9-9 Структурная схема выводов PORTF LCD



Примечание. Выводы имеют защитные диоды, подключенные к V_{DD} и V_{SS}.

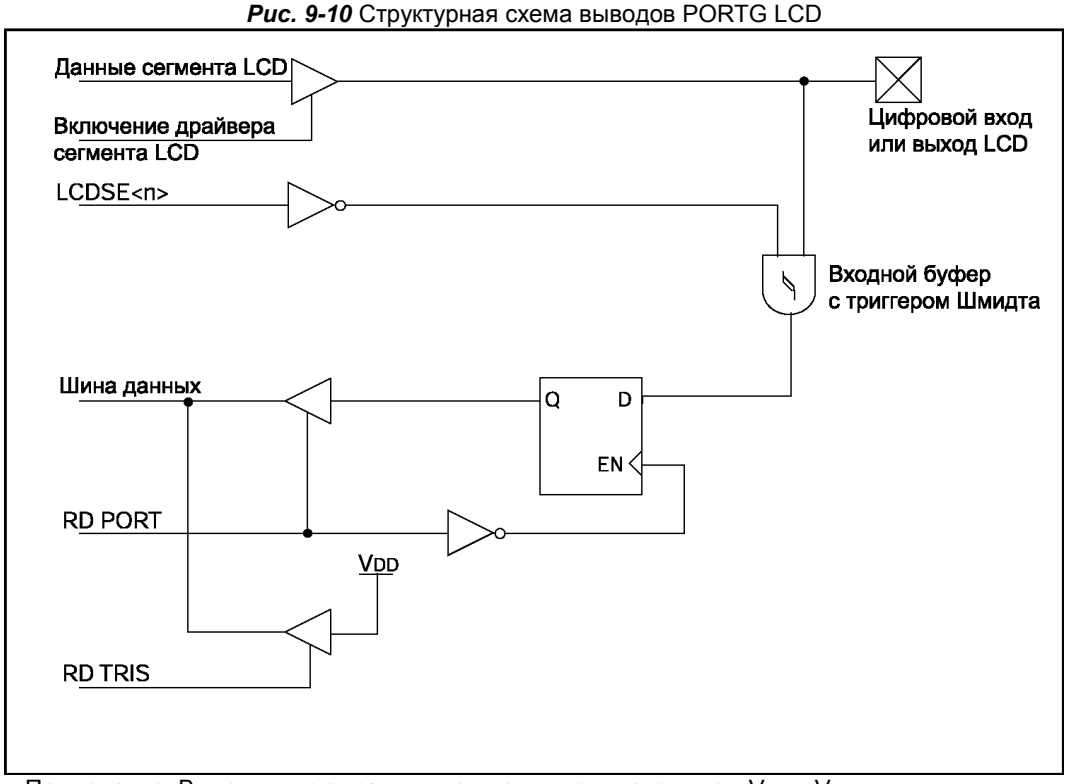
9.8 Регистры PORTG и TRISG

PORTG – 8-разрядный цифровой порт, работающий только как вход. Каждый вывод порта мультиплицирован с драйвером сегмента LCD. К выводам порта подключены входные буферы с триггером Шмидта.

Пример 9-7 Инициализация PORTG

```
BCF         STATUS, RP0         ; Выбрать банк 2
BSF         STATUS, RP1         ;
BCF         LCDSE,SE27          ; Настроить все выводы PORTE и PORTG
BCF         LCDSE,SE20          ; как цифровые входы
```

Рис. 9-10 Структурная схема выводов PORTG LCD



Примечание. Выводы имеют защитные диоды, подключенные к V_{DD} и V_{SS}.

9.9 Регистры GPIO и TRISGP

GPIO - 8-разрядный регистр порта ввода/вывода, в котором реально используется только 6 младших битов (GP5:GP0). Биты 7 и 6 не реализованы и читаются как '0'. Любой вывод GPIO (кроме GP3) может индивидуально настроен на вход или выход. Канал GP3 работает только как вход.

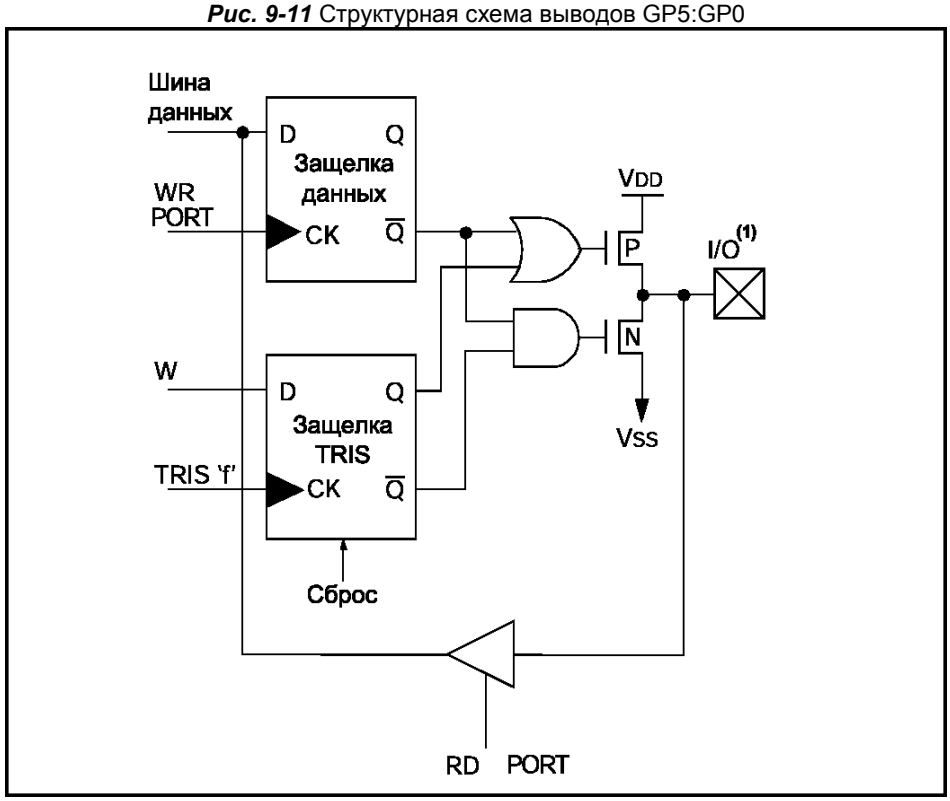
Регистр TRISGP управляет направлением данных каналов порта GPIO. Запись ‘1’ в TRISGP переводит соответствующий выходной буфер в 3-е состояние. Запись ‘0’ в регистр TRISGP определяет соответствующий канал как выход, содержимое защелки передается на вывод микроконтроллера. Исключением является канал GP3, который может работать только как вход. Чтение бита TRISGP вывода GP3 будет давать результат '1'. При сбросе микроконтроллера все порты ввода/вывода настраиваются на вход, т.к. все биты регистра TRISGP устанавливаются в '1'.

Все каналы порта ввода/вывода не имеют входных защелок для чтения. Входной сигнал должен присутствовать на входе, пока выполняется операция чтения порта (например, MOVF GPIO,W). Выходные данные сохраняются в защелке и остаются неизменными, пока не будут перезаписаны.

Пример 9-8 Инициализация GPIO

```
BCF     STATUS, RP0   ; Выбрать банк 0
CLRF    GPIO          ; Инициализация защелок GPIO

BSF     STATUS, RP0   ; Выбрать банк 1

MOVLW   0xCF          ; Значение для инициализации
                      ; направления каналов GPIO
MOVWF   TRISGP        ; Настроить GP<3:0> как входы,
                      ; GP<5:4> как выходы
                      ; Чтение TRISGP<7:6> будет давать результат '0'
```

Рис. 9-11 Структурная схема выводов GP5:GP0



Примечание 1. Выводы порта имеют защитные диоды, подключенные к V_{DD} и V_{SS}. GP3 может работать только как вход (нет выходного буфера).

Каналы порта ввода/вывода могут быть задействованы для реализации других функций микроконтроллера (настраивается в битах конфигурации). Чтение этих каналов будет давать результат '0'. Выводы GP0, GP1 и GP3 имеют управляемые подтягивающие резисторы и могут генерировать прерывание при изменении уровня сигнала на входах. При включении прерываний по изменению уровня сигнала на входах, внутренние подтягивающие резисторы не могут быть подключены. Прерывания разрешаются установкой бита INTCON<3> в '1'. Если битами конфигурации выбран один из внешних режимов тактового генератора, то функции портов GP4 и GP5 выключены, а выводы используются для работы тактового генератора.

9.10 Программирование портов ввода/вывода

При использовании портов (в том числе GPIO) как каналов ввода/вывода, необходимо учитывать некоторые нюансы работы портов ввода/вывода для получения требуемого результата.

9.10.1 Двунаправленные порты ввода/вывода

Все операции записи в порт выполняются по принципу "чтение - модификация - запись". Например, команды BCF и BSF считывают значение в регистр ЦПУ, выполняют битовую операцию и записывают результат обратно в регистр. Требуется некоторая осторожность при применении подобных команд к регистрам портов ввода/вывода. Например, команда BSF PORTB,5 считывает все восемь битов PORTB в ЦПУ, изменяет состояние бита 5 и записывает результат в выходные защелки PORTB. Если другой канал PORTB (например, RB0) настроен на вход, то сигнал на выводе будет считан в ЦПУ и записан в защелку данных, поверх предыдущего значения. Пока RB0 настроен как вход, никаких проблем не возникает. Однако, если RB0 будет позже настроен как выход, значение в защелке данных может отличаться от требуемого.

При чтении регистра порта, читается текущее состояние порта ввода/вывода. Запись в регистр порта, сохраняет значение в защелке порта ввода/вывода. Когда используются команды "чтение - модификация - запись" (например BSF, BCF и т.д.), считывается текущее состояние порта ввода/вывода, выполняется требуемая операция и полученное значение записывается в защелку порта.

В примере 9-9 показан эффект последовательного выполнения команд "чтение - модификация - запись" с регистром порта ввода/вывода.

Пример 9-9 Эффект выполнения команд "чтение - модификация – запись"
Начальные установки порта: PORTB<7:4> входы, PORTB <3:0> выходы.
Выводы RB7:RB6 имеют внешние подтягивающие резисторы и не подключены к другим цепям схеме.

```
                        Защелка PORTB    Выводы PORTB
BCF         STATUS,RP0    ;
BCF         PORTB, 7      ; 01pp pppp   11pp pppp
BCF         PORTB, 6      ; 10pp pppp   11pp pppp
BSF         STATUS,RP0 ;
BCF         TRISB, 7      ; 10pp pppp   11pp pppp
BCF         TRISB, 6      ; 10pp pppp   10pp pppp
```

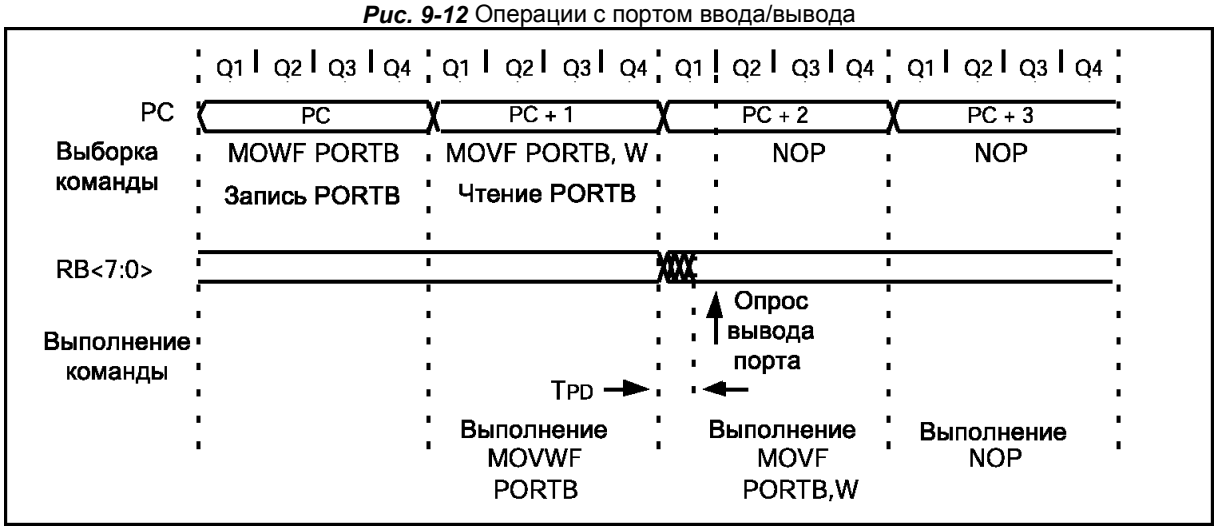
Обратите внимание. Возможно пользователь ожидал, что после выполнения программы на выходах PORTB будет значение 00pp pppp. Однако 2-я команда BCF установила в ‘1’ RB7.

На активный вывод порта не должны подключаться нагрузки, включенные по схемам "монтажное И" или "монтажное ИЛИ". Возможные большие токи могут повредить микроконтроллер.

9.10.2 Последовательность операций с портами ввода/вывода

Запись в порт ввода/вывода фактически происходит в конце машинного цикла, а чтение данных выполняется в начале цикла (см. рисунок 9-12). Поэтому требуется некоторая осторожность при записи в порт ввода/вывода, если перед записью выполняется чтение состояния этого порта. Последовательность команд должна быть такой, чтобы установилось напряжение на выводе порта прежде, чем будет выполнена команда записи в порт, сопровождаемая чтением состояния выводов (иначе вместо нового значения может быть считано предыдущее). Если возможна описанная ситуация, разделите команды записи инструкциями NOP или любыми другими командами, которые не обращаются к порту ввода/вывода.

Рис. 9-12 Операции с портом ввода/вывода



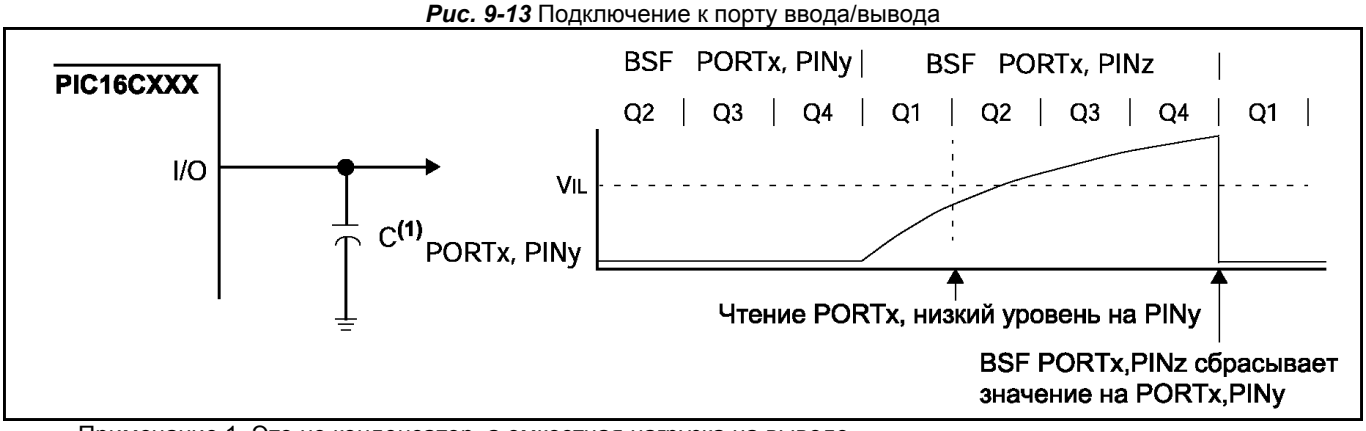
Примечание к рисунку. На рисунке показан пример чтения из PORTB сразу поле записи в него. Время установления данных на PORTB равно $T = 0.25\ T_{CY} - T_{PD}$. Где: T_{CY} – длительность машинного цикла микроконтроллера, T_{PD} – задержка распространения. Следовательно, при высокой тактовой частоте микроконтроллера, чтение с порта ввода/вывода непосредственно после записи может возвращать неверные значения.

На рисунке 9-13 показана модель канала ввода/вывода с описанной ситуацией. С увеличением емкости (C) увеличивается длительность нарастания/спада уровня напряжения на выводе. При увеличении тактовой частоты микроконтроллера, усиливается влияние емкости на выполнение команд "чтение - модификация - запись" с регистром порта ввода/вывода PORTX. В состав емкости (C) входит паразитная емкость проводника и выводов компонентов, подключенных к выводу порта ввода/вывода.

Лучшем способом решения этой проблемы может быть подключение дополнительного резистора к выводу порта. Подключенный резистор позволяет достигнуть на выводе порта напряжения требуемого уровня перед выполнением следующей команды.

Использование команд NOP между последовательными обращениями к регистру PORTX "чтение - модификация - запись" является наиболее дешевым, но имеет существенный недостаток: число команд NOP зависит от значения емкости и тактовой частоты микроконтроллера.

Рис. 9-13 Подключение к порту ввода/вывода



Примечание 1. Это не конденсатор, а емкостная нагрузка на выводе.

9.11 Инициализация

Смотрите примеры инициализации в разделах, описывающих соответствующие порты ввода/вывода.

Примечание. При инициализации портов ввода/вывода рекомендуется сначала записать стартовое значение в выходную защелку порта (регистр PORT), а затем настроить направление каналов порта (регистр TRIS). Эта последовательность устраняет возможность ложного уровня на выходе порта, т.к. при включении питания в выходных защелках порта содержится случайное значение.

9.12 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Программа микроконтроллера не переключает состояние портов ввода/вывода, тактовый генератор работает. Что я сделал неправильно?

Ответ 1:

1. Регистры TRIS инициализированы правильно? Эти регистры расположены в банке 1. В большинстве случаев пользователи не переключают банк памяти (BSF STATUS,RP0) при записи значения в регистры TRIS.
2. Если вы правильно выполнили запись в регистр TRIS (выбрали банк 1, RP0=1), то возможно Вы не восстановили банк 0 (BCF STATUS,RP0) перед записью в защелки портов.
3. Существуют периферийные модули, мультиплицированные с требуемым каналом порта ввода/вывода?
4. Сторожевой таймер WDT включен (настраивается при программировании)? Если WDT включен, выполняется команда CLRWDT по крайне мере каждые 9мс (или больше, если подключен выходной делитель к WDT)?
5. Вы используете правильные команды для записи в порт? Много людей использовали команду MOVF для записи в порт ввода/вывода, вместо MOVWF.
6. Для случаев, если используются прерывания. Попробуйте выключить прерывания, чтобы удостовериться, что они не влияют на состояние портов ввода/вывода.

Вопрос 2: При чтении состояния порта я получаю значение отличное от записываемого. Что может быть причиной?

Ответ 2:

1. При чтение порта ввода/вывода возвращается состояние выводов порта независимо от настройки каналов вводы/вывода (вход или выход). Если вывод настроен как вход, Вы будете получать состояние выводов порта независимо от значения регистра.
2. Если вывод настроен как выход, в защелке данных записана '1', а вывод порта подтянут к нулю, то чтение будет давать результат '0'. Это полезно при обработке ошибок или конфликтов на шине I^2C (Управление низким логическим уровнем на шине I^2C осуществляется за счет изменение битов TRIS. Если на линии низкий логический уровень, а Вы его не формируете, то другое устройство захватило арбитраж шины).
3. Большинство микроконтроллеров среднего семейства имеют как минимум один вывод с открытым стоком (открытым коллектором). На этих выводах можно управлять только низким логическим уровнем. Для микроконтроллеров среднего семейство это вывод RA4. К выходам с открытым коллектором должен быть подключен внешний подтягивающий резистор. Этот вывод может использоваться для формирования уровней напряжения не соответствующих логическим уровням микроконтроллера. Подтягивающий резистор может быть подключен к другому источнику напряжения (обычно ниже V_{DD}), которое устанавливается на выводе при формировании логической единицы.

Вопрос 3: Я использую внешнее прерывание на выводе RB0 в микроконтроллере PIC16CXXX. Прерывания не генерируются, хотя когда я изменяю свою программу и проверяю логические уровни на входе RB0, все работает. Почему?

Ответ 3:

PORTB имеет входной буфер ТТЛ. Если на входе порта присутствует напряжение 3В (при V_{DD}=5В), то будет прочитана логическая единица. Однако для обслуживая внешних прерываний на RB0 используется входной буфер с триггером Шмидта, который требует более высокого напряжения (чем буфер ТТЛ) для регистрации логической единицы. Поэтому возможна ситуации, в которой чтение будет давать правильное состояние порта, но сигнал на входе не будет вызывать генерацию прерывания. Гистерезис входного буфера с триггером Шмидта позволяет минимизировать влияние внешних помех. Следует различать влияние внешних кратковременных помех между получением недостоверных данных при чтении состояния порта и возникновению ложных прерываний.

Вопрос 4: При использовании команды BCF другие выводы порта принимают низкий логический уровень. Почему?

Ответ 4:

1. Случай, когда команда "чтение - модификация - запись" изменяет состояние других выводов порта, можно продемонстрировать следующим образом. Предположим, что все каналы PORTC настроены на выход и имеют низкий логический уровень. К каждому выводу подключен светодиод, который светится при формировании высокого логического уровня на выводах порта. Параллельно каждому светодиоду подключен конденсатор с емкостью 100мкФ. Программа микроконтроллера выполняется очень быстро, тактовая частота 20МГц. Теперь последовательно формируем команды, включающие светодиоды: BSF PORTC,0; BSF PORTC,1; BSF PORTC,2 и т.д. Вы можете видеть, что только на последнем выводе высокий уровень сигнала и только последний светодиод светится. Это произошло потому , что конденсаторы требуют некоторого времени для зарядки до напряжения высокого логического уровня. Поскольку каждый вывод устанавливался в '1' прежде, чем зарядится конденсатор предыдущего вывода, чтение давало результат '0'. Это '0' записывается в выходную защелку, восстанавливая низкий логический уровень на выводе (команда "чтение - модификация - запись"). Учитывать этот эффект необходимо только при высокой тактовой частоте микроконтроллера и выполнении последовательных операций с портами ввода/вывода.
2. Такая ситуация возможна в микроконтроллерах PIC16C7XX, если Вы не настроили каналы порта ввода/вывода должным образом в регистре ADCON1. Если вывод настроен как аналоговый вход, то чтение будет давать результат '0' независимо от уровня напряжения на выводе. Это исключение к правилу, что всегда выполняется чтение состояние вывода. Вы можете настраивать аналоговый вывод как цифровой выход в регистре TRISA, и управлять логическим уровнем на выходе, но чтение всегда будет давать результат '0'. Поэтому, если вы обращаетесь к порту командой "чтение - модификация - запись", все аналоговые выводы читаются как '0', командой изменяется прочитанное значение и записывается назад в защелку порта как '0'. На аналоговых входах могут присутствовать нелогические уровни, поэтому входные цифровые буферы выключены для предотвращения возможного повышенного энергопотребления.

9.13 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с портами ввода/вывода в микроконтроллерах PICmicro MCU:

Документ	Номер
Improving the Susceptibility of an Application to ESD Улучшение устойчивость устройства к ESD	AN595
Clock Design using Low Power/Cost Techniques Реализация часов с малым энергопотреблением и низкой стоимостью	AN615
Implementing Wake-up on Keystroke Реализация выхода из режима SLEEP при нажатии на кнопки	AN528
Interfacing to AC Power Lines Подключение к электросети	AN521
Multiplexing LED Drive and a 4 x 4 Keypad Sampling Реализация светодиодного драйвера и клавиатуры 4x4	AN529
Using PIC16C5X as an LCD Drivers Использование PIC16C5X как драйвера LCD	AN563
Serial Port Routines Without Using TMR0 Реализация последовательного порта без использования TMR0	AN593
Implementation of an Asynchronous Serial I/O Реализация асинхронного последовательного порта ввода/вывода	AN510
Using the PORTB Interrupt on Change Feature as an External Interrupt Применение внешнего прерывания и прерывания по изменению сигнала на входах PORTB	AN566
Implementing Wake-up on Keystroke Реализация выхода из режима SLEEP при нажатии на кнопки	AN522
Apple Desktop Bus Шина Apple Desktop	AN591
Software Implementation of Asynchronous Serial I/O Программная реализация асинхронного последовательного порта ввода/вывода	AN555
Communicating with the I^2C Bus using the PIC16C5X Обмен данными по шине I^2C для микроконтроллеров PIC16C5X	AN515
Interfacing 93CX6 Serial EEPROMs to the PIC16C5X Microcontrollers Работа с последовательной EEPROM памятью 93CX6 для микроконтроллеров PIC16C5X	AN530
Logic Powered Serial EEPROMs Логика подключения последовательной EEPROM памяти	AN535
Interfacing 24LCXXB Serial EEPROMs to the PIC16C54 Работа с последовательной EEPROM памятью 24LCXXB для микроконтроллеров PIC16C54	AN567
Using the 24XX65 and 24XX32 with Stand-alone PIC16C54 Code Набор стандартных программ работы с микросхемами 24XX65 и 24XX32 для PIC16C54	AN558

Раздел 10. Ведомый параллельный порт

Содержание

10.1 Введение

Некоторые микроконтроллеры содержат ведомый параллельный порт, мультиплицированный на один из портов ввода/вывода. PORT работает как 8-разрядный параллельный порт (или порт микропроцессора), когда бит PSPMODE установлен в '1' (к выводам подключены входные буферы ТТЛ).

В режиме ведомого данные асинхронно читаются или записываются внешними сигналами -RD и -WR соответственно.

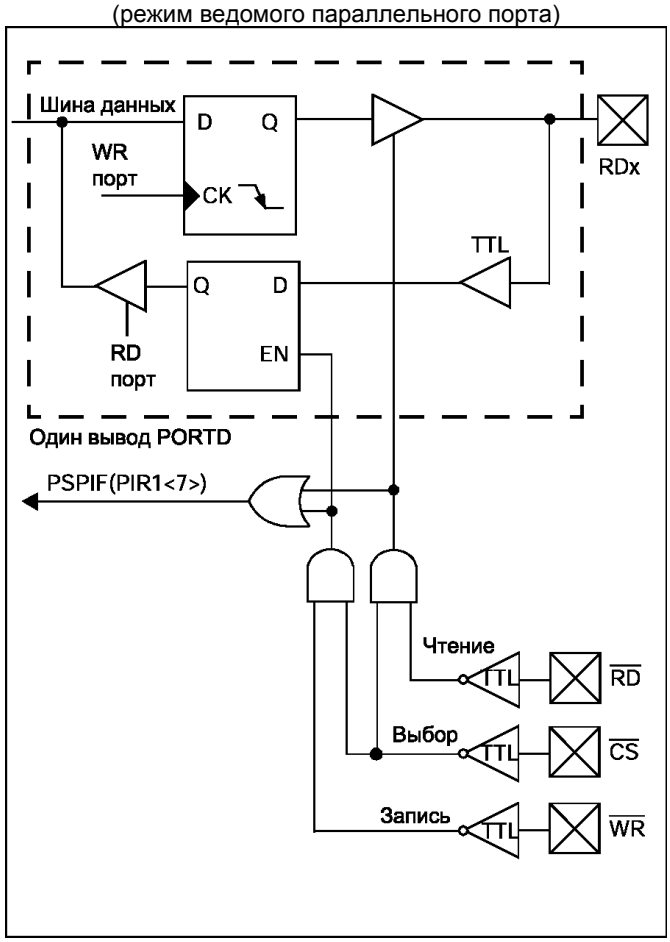
С помощью 8-разрядного ведомого параллельного порта можно организовать интерфейс связи с микропроцессором. Установка бита PSPMODE в '1' принудительно настаивает выводы -RD, -WR и -CS как входы.

Примечание 1. В настоящее время ведомый параллельный порт мультиплицирован только с PORTD, PORTE. Порт микропроцессора становится доступным только, когда бит PSPMODE установлен в '1'. В этом режиме биты TRISE должны быть установлены в '1' (выводы настроены как цифровые входы, АЦП отключено). Для PORTD отменяется действие битов TRISD.

Примечание 2. В режиме ведомого параллельного порта ко входам PORTD и PORTE подключены буферы ТТЛ. Биты управления PSP расположены в регистре TRISE.

Фактически существуют два 8-разрядных регистра: один регистр для приема данных, другой - для передачи. Пользователь записывает 8-разрядные данные в выходную защелку PORT, а читает данные со входной защелки (обратите внимание, выходная и входная защелка имеют один и тот же адрес). В этом режиме значение битов регистра TRIS игнорируется, т.к. направлением данных управляет внешнее устройство.

Рис. 10-1 Структурная схема выводов PORTD и PORTE (режим ведомого параллельного порта)



Примечание. Выводы портов имеют защитные диоды, подключенные к V_{DD} и V_{SS}.

10.2 Управляющий регистр

Регистр TRISE

R-0	R-0	R-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
IBF	**OBF**	**IBOV**	**PSPMODE**	**-**	**BIT2**	**BIT1**	**BIT0**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **IBF**: Бит статуса приемного буфера
1 = принят байт данных
0 = байт данных не был получен

бит 6: **OBF**: Бит статуса передающего буфера
1 = предварительно записанный байт данных еще не прочитан
0 = выходной буфер был прочитан

бит 5: **IBOV**: Флаг переполнения приемного буфера
1 = произошла новая запись, а предыдущий байт не был прочитан (сбрасывается программно)
0 = переполнения не было

бит 4: **PSPMODE**: Режим работы ведомого параллельного порта
1 = режим ведомого параллельного порта
0 = режим цифровых каналов ввода/вывода

бит 3: **Не реализован**: читается как '0'

бит 2: **BIT2**: Направление вывода RE2
1 = вход
0 = выход

бит 1: **BIT1**: Направление вывода RE1
1 = вход
0 = выход

бит 0: **BIT0**: Направление вывода RE0
1 = вход
0 = выход

10.3 Работа ведомого параллельного порта

Запись в PSP происходит, если выводы -CS и -WR имеют низкий уровень сигнала. После перехода сигнала на выводе -CS или -WR в высокий уровень (передний фронт сигнала) данные сохраняются в входной защелке на такте Q2. На такте Q4 устанавливаются в '1' бит IBF(TRISE<7>) и флаг прерываний PSPIF. Очистка бита IBF заблокирована в дополнительном цикле T_{CY} (см. параметр 66). Бит IBF может быть сброшен в '0' только чтением регистра PORTD (командой MOVF, команды "чтение - модификация - запись" не должны использоваться). Бит переполнения IBOV(TRISE<5>) устанавливается в '1', если произошла следующая запись в PSP, а предыдущий байт не был прочитан.

Чтение из PSP происходит, если выводы -CS и -RD имеют низкий уровень сигнала. Немедленно сбрасывается в '0' бит OBF(TRISE<6>), указывающий, что PORTD прочитан внешней шиной. После перехода сигнала на выводе -CS или -RD в высокий уровень (передний фронт сигнала) устанавливается флаг прерывания PSPIF на такте Q4 (только после такта Q2), указывая, что чтение завершено. Бит OBF остается сброшенным в '0', пока не будут загружены новые данные в PORTD программой пользователя.

Бит флага полного входного буфера IBF устанавливается в '1', если получено новое слово, ожидаемое чтения ЦПУ. Как только PORTD будет прочитан, бит IBF аппаратно сбрасывается в '0'. Бит IBF отображает только состояние приемного буфера. Состояние выходного буфера отображает бит OBF. Если бит OBF=1, то в буфер записано слово, ожидаемое чтения внешней шиной. Как только выходной буфер читается микропроцессором, бит OBF аппаратно сбрасывается в '0'. Бит флага переполнения входного буфера IBOV устанавливается в '1', если микропроцессор выполняет запись нового слова, а предыдущее слово не было прочитано ЦПУ (первое слово сохраняется в буфере).

Когда режим PSP выключен, биты IBF и OBF равняются нулю, а предварительно установленный в '1' бит IBOV должен быть сброшен программно.

Флаг прерывания PSPIF устанавливается в '1' по завершению каждой операции чтения или записи (сбрасывается в '0' программно). Разрешить/запретить прерывания от модуля PSP можно установкой/сбросом бита PSPIE.

Таблица 10-1 Функциональное назначение выводов PORTE

Обозначение вывода	Описание
RE0/-RD	Двунаправленный порт ввода/вывода или вход управления чтением ведомого параллельного порта или аналоговый вход: -RD 1 = Ожидание 0 = Операция чтения. Защелка PORTD подключена к выводам PORTD (если -CS = 0)
RE1/-WR	Двунаправленный порт ввода/вывода или вход управления записью ведомого параллельного порта или аналоговый вход: -WR 1 = Ожидание 0 = Операция записи. Данные с выводов PORTD сохраняются во внутренней защелке PORTD (если -CS = 0)
RE2/-CS	Двунаправленный порт ввода/вывода или вход выбора микросхемы ведомого параллельного порта или аналоговый вход: -CS 1 = Микросхема не выбрана 0 = Микросхема выбрана

Примечание. PSP может мультиплицировать другие функции на эти выводы. Для работы PSP необходимо, чтобы выводы были настроены как цифровые каналы ввода/вывода.

10.4 Работа в SLEEP режиме

В SLEEP режиме микроконтроллера микропроцессор может выполнить запись или чтение ведомого параллельного порта. Чтение и запись в PSP вызывают установку в '1' флага прерывания PSPIF. Если прерывания от PSP разрешены, то микроконтроллер выйдет из режима SLEEP для обработки (подготовки) новых данных от микропроцессора.

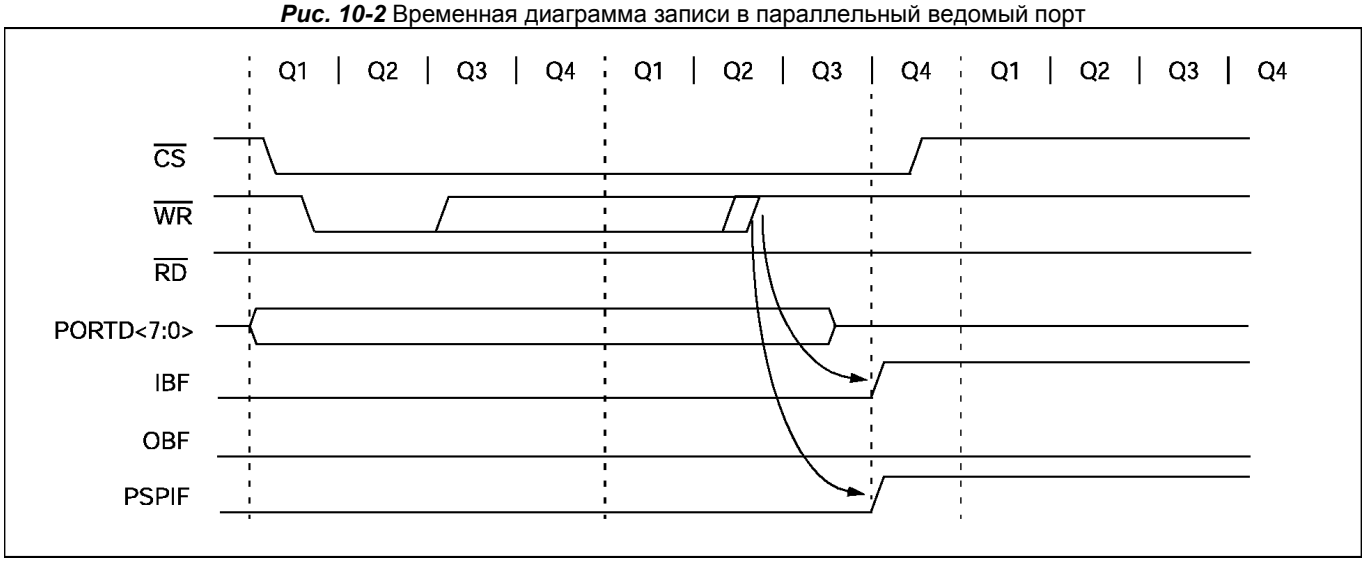
10.5 Эффект сброса

После любого вида сброса микроконтроллера PSP выключен, а PORTD и PORTE переходят в режим по умолчанию.

10.6 Временные диаграммы работы

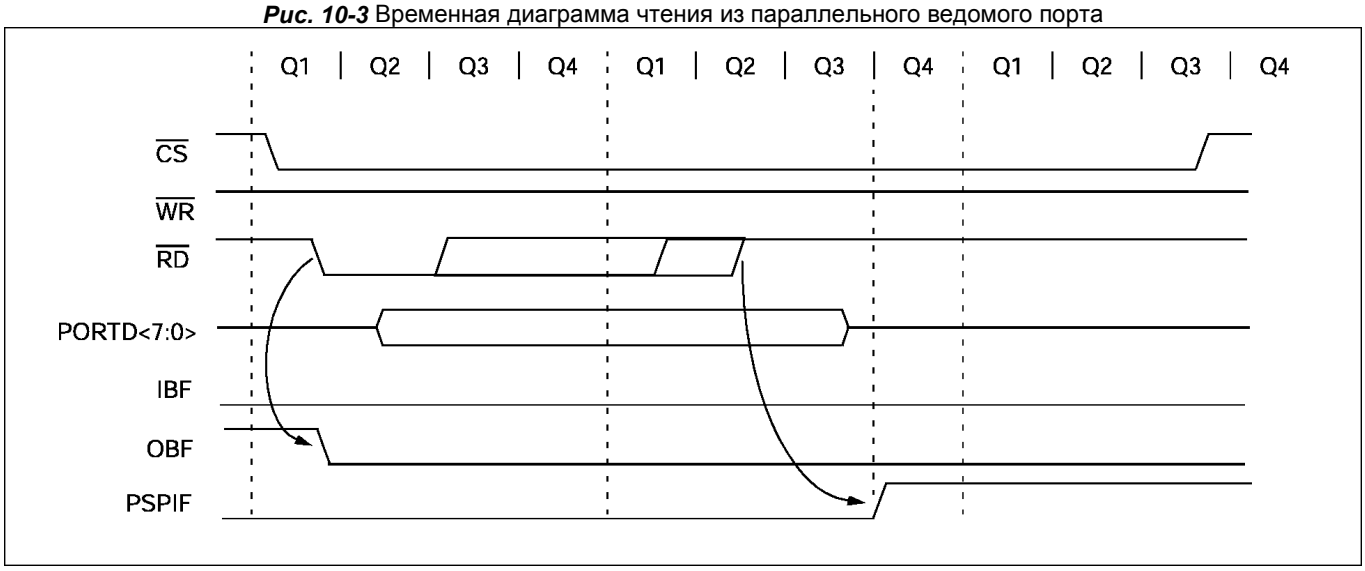
На рисунке 10-2 показана временная диаграмма записи в PSP, а на рисунке 10-3 представлена временная диаграмма чтения PSP микропроцессором.

Рис. 10-2 Временная диаграмма записи в параллельный ведомый порт



Примечание. Сброс бита IBF заблокирован до этого момента.

Рис. 10-3 Временная диаграмма чтения из параллельного ведомого порта



10.7 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Перейдя с микроконтроллеров PIC16C74 на PIC16C74A заметил, что изменилась работа PSP.
Ответ 1:
Действительно, были сделаны некоторые изменения. В PIC16C74A операция происходит по получению активного фронта управляющего сигнала (в PIC16C74 управление выполнялось уровнем сигнала, а не фронтом). Дополнительную информацию смотрите в приложении C.9.

10.8 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с ведомым параллельным портом в микроконтроллерах PICmicro MCU:

Документ	Номер
Using the 8-bit Parallel Slave Port Использование 8 - разрядного ведомого параллельного порта	AN579

Раздел 11. Таймер TMR0

Содержание

11.1 Введение

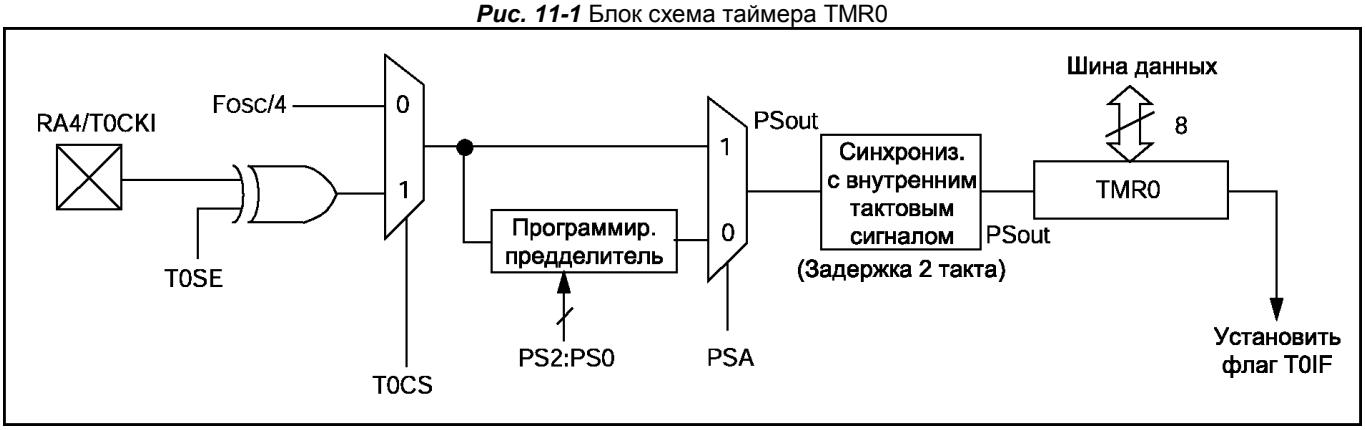
TMR0 - таймер/счетчик, имеет следующие особенности:

- 8-разрядный таймер/счетчик;
- возможность чтения и записи текущего значения счетчика;
- 8-разрядный программируемый предделитель;
- внутренний или внешний источник тактового сигнала;
- выбор активного фронта внешнего тактового сигнала
- прерывания при переполнении (переход от FFh к 00h).

Примечание. Если предварительный делитель включен перед WDT, то коэффициент деления тактового сигнала для TMR0 равен 1:1.

Блок схема модуля TMR0 показана на рисунке 11-1.

Рис. 11-1 Блок схема таймера TMR0



Примечания:

1. Биты управления T0CS, T0SE, PS2, PS1, PS0, PSA расположены в регистре OPTION_REG.
2. Схему включения предделителя перед WDT смотрите на рисунке 11-6.

11.2 Управляющий регистр

Регистр OPTION_REG доступен для чтения и записи, содержит биты управления:

- Предварительным делителем TMR0/WDT;
- Активным фронтом внешнего прерывания RB0/INT;
- Подтягивающими резисторами на входах PORTB.

Примечание. Если предварительный делитель включен перед WDT, то коэффициент деления тактового сигнала для TMR0 равен 1:1.

Регистр OPTION_REG

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
-RBPU(1)	**INTEDG**	**T0CS**	**T0SE**	**PSA**	**PS2**	**PS1**	**PS0**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **-RBPU(1)**: Включение подтягивающих резисторов на входах PORTB
1 = подтягивающие резисторы отключены
0 = подтягивающие резисторы включены

бит 6: **INTEDG**: Выбор активного фронта сигнала на входе внешнего прерывания INT
1 = прерывания по переднему фронту сигнала
0 = прерывания по заднему фронту сигнала

бит 5: **T0CS**: Выбор тактового сигнала для TMR0
1 = внешний тактовый сигнал с вывода T0CKI
0 = внутренний тактовый сигнал CLKOUT

бит 4: **T0SE**: Выбор фронта приращения TMR0 при внешнем тактовом сигнале
1 = приращение по заднему фронту сигнала (с высокого к низкому уровню) на выводе T0CKI
0 = приращение по переднему фронту сигнала (с низкого к высокому уровню) на выводе T0CKI

бит 3: **PSA**: Выбор включения предделителя
1 = предделитель включен перед WDT
0 = предделитель включен перед TMR0

биты 2-0: **PS2: PS0**: Установка коэффициента деления предделителя

Значение	Для TMR0	Для WDT
000	1:2	1:1
001	1:4	1:2
010	1:8	1:4
011	1:16	1:8
100	1:32	1:16
101	1:64	1:32
110	1:128	1:64
111	1:256	1:128

Примечание 1. В некоторых микроконтроллерах этот бит обозначается как -GPPU. Если микроконтроллер содержит бит -RBPU, то подтягивающие резисторы подключены к PORTB. Если микроконтроллер содержит бит -GPPU, то подтягивающие резисторы подключены к GPIO.

11.3 Работа таймера TMR0

Когда бит T0CS сброшен в ‘0’ (OPTION_REG<5>), TMR0 работает от внутреннего тактового сигнала. Приращение счетчика TMR0 происходит в каждом машинном цикле (если предделитель отключен). После записи в TMR0 приращение счетчика запрещено два следующих цикла (см. рисунки 11-2 и 11-3). Пользователь должен скорректировать эту задержку перед записью нового значения в TMR0.

Если бит T0CS установлен в ‘1’ (OPTION_REG<5>), TMR0 работает от внешнего источника тактового сигнала на входе T0CKI. Активный фронт внешнего тактового сигнала выбирается битом T0SE (OPTION_REG<4>) (T0SE=0 - активным является передний фронт сигнала). Работа модуля TMR0 с внешним источником тактового сигнала будет рассмотрена в разделе 11.5.

Предделитель может быть включен перед WDT или TMR0, в зависимости от состояния бита PSA в регистре OPTION_REG<3>. Если бит PSA сброшен в ‘0’, то предделитель включен перед TMR0. Нельзя прочитать или записать новое значение в предделитель. Когда предделитель включен перед TMR0, можно выбрать его коэффициент деления 1:2, 1:4,…,1:256. Подробное описание работы с предделителем смотрите в разделе 11.6.

Любая запись в регистр TMR0 вызовет запрещение приращения таймера TMR0 в течение двух следующих машинных циклов ($2T_{CY}$). Т.е. после записи нового значения в TMR0 таймер не инкрементируется пока не определено, что 3-я команда не циклически повторяющаяся (см. рисунок 11-2). Если предделитель включен перед TMR0, то запись в регистр TMR0 вызовет немедленное изменение TMR0 и сброс предделителя. Приращение TMR0 и предделителя запрещено в течение 2-х машинных циклов ($2T_{CY}$), после записи в TMR0. Например, если коэффициент предделителя равен 2, то после операции записи в регистр TMR0 приращение таймера не будет происходить в течение 4 циклов для TMR0 (см. рисунок 11-3). Далее таймер работает в нормальном режиме.

Рис. 11-2 Временная диаграмма работы TMR0 от внутреннего источника тактового сигнала (предделителе выключен)

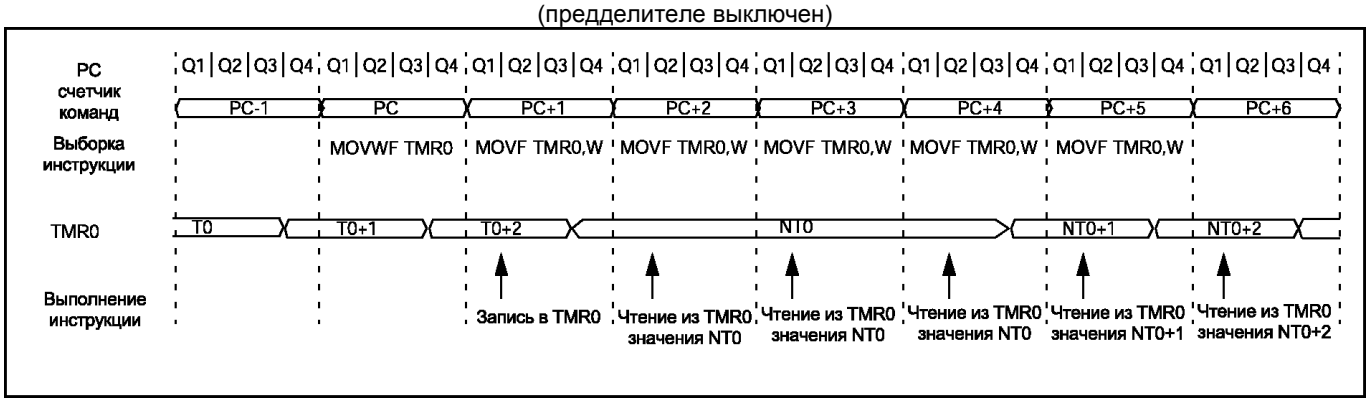
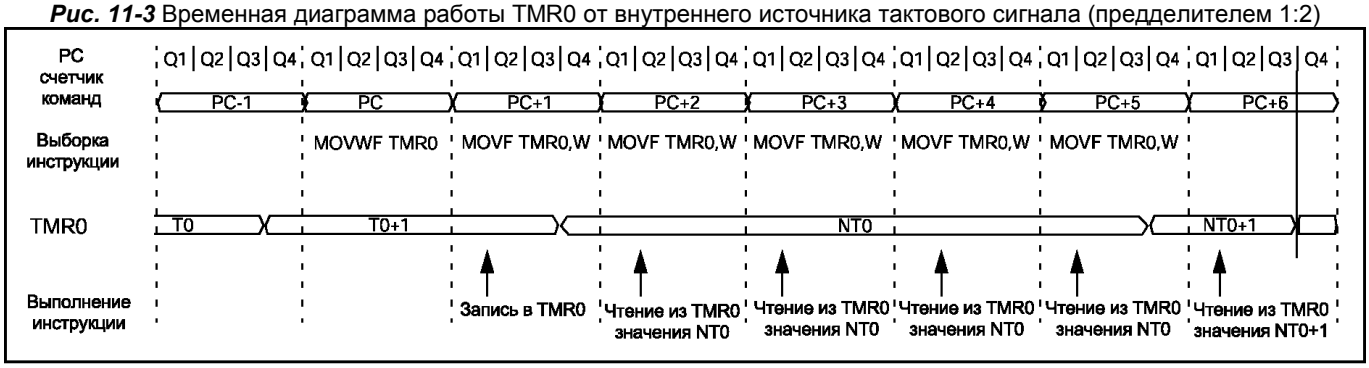


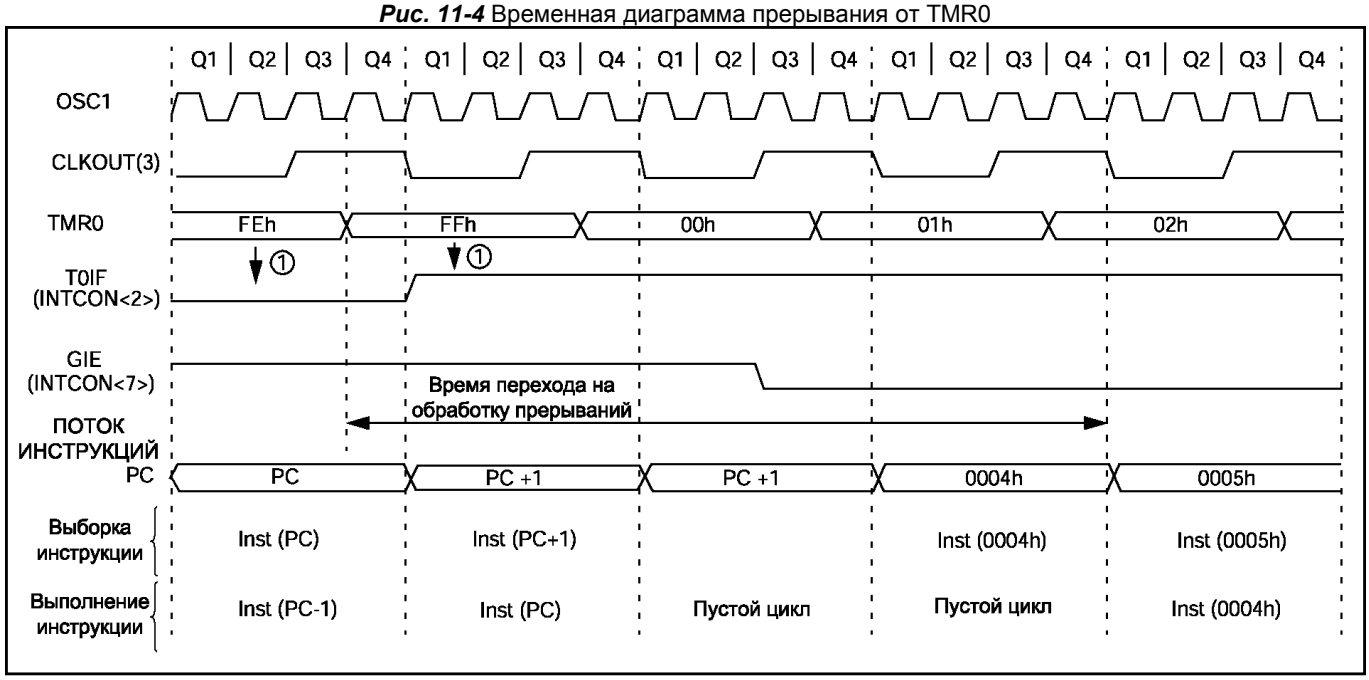
Рис. 11-3 Временная диаграмма работы TMR0 от внутреннего источника тактового сигнала (предделителем 1:2)



11.4 Прерывания от TMR0

Прерывания от TMR0 возникают при переполнении счетчика, т.е. при переходе его значения от FFh к 00h. При возникновении прерывания устанавливается в ‘1’ бит T0IF (INTCON<2>). Само прерывание может быть разрешено/запрещено установкой/сбросом бита T0IE в регистре INTCON<5>. Флаг прерывания от TMR0 T0IF (INTCON<2>) должен быть сброшен в подпрограмме обработки прерываний. В SLEEP режиме микроконтроллера модуль TMR0 выключен и не может генерировать прерывания. На рисунке 11-4 показана временная диаграмма возникновения прерывания от TMR0.

Рис. 11-4 Временная диаграмма прерывания от TMR0



Примечания:

1. Проверка установленного флага T0IF выполняется в каждом цикле на такте Q1.
2. Время перехода на обработку прерывания $3T_{CY}$, где T_{CY} – машинный цикл.
3. CLKOUT доступен только в RC режиме тактового генератора.

11.5 Использование внешнего источника тактового сигнала для TMR0

При использовании внешнего тактового сигнала для TMR0 необходимо учитывать некоторые детали работы таймера. Активный фронт внешнего тактового сигнала синхронизируется с внутренней тактовой частотой микроконтроллера (F_{OSC}), из-за чего возникает задержка от получения активного фронта сигнала до приращения TMR0.

11.5.1 Синхронизация внешнего сигнала

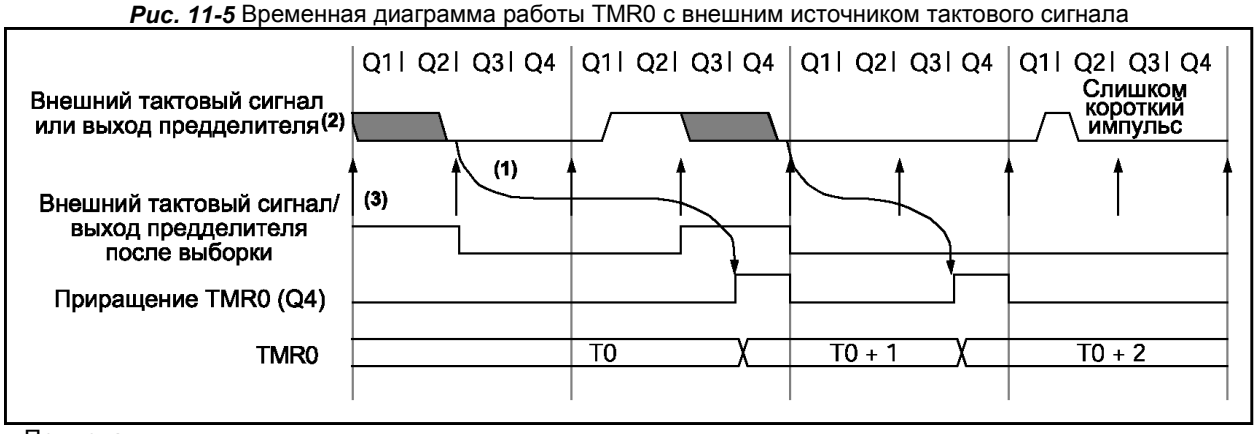
Если предделитель не используется, внешний тактовый сигнал поступает непосредственно на синхронизатор. Синхронизация T0CKI с таковым сигналом микроконтроллера усложняется из-за опроса выхода синхронизатора в машинные циклы Q2 и Q4 (см. рисунок 11-5). Поэтому длительность высокого или низкого логического уровня внешнего сигнала должна быть не меньше $2T_{OSC}$ (плюс небольшая задержка внутренней RC цепи 20нс). Дополнительную информацию смотрите в разделе "Электрические характеристики" (параметры 40, 41, 42).

Если предделитель включен перед TMR0, то на вход синхронизатора поступает сигнал с асинхронного предделителя. Период сигнала T0CKI должен быть не менее $4T_{OSC}$ (плюс небольшая задержка внутренней RC цепи 40нс) деленное на коэффициент предделителя. Дополнительное требование: высокий и низкий логический уровень внешнего сигнала должен быть не менее 10нс. Смотрите параметры 40, 41 и 42 в разделе "Электрические характеристики".

11.5.2 Задержка приращения TMR0

Поскольку сигнал с выхода предделителя синхронизируется с внутренним тактовым сигналом микроконтроллера, возникает задержка от получения активного фронта сигнала до приращения TMR0 (см. рисунок 11-5).

Рис. 11-5 Временная диаграмма работы TMR0 с внешним источником тактового сигнала



Примечания:

1. Задержка от активного фронта тактового сигнала до приращения TMR0 от $3T_{OSC}$ до $7T_{OSC}$. Следовательно, максимальная ошибка измерения интервала между двумя активными фронтами тактового сигнала $\pm 4T_{OSC}$.
2. Если предделитель выключен, на вход синхронизатора поступает внешний тактовый сигнал.
3. Стрелками указаны точки выборки уровня сигнала.

11

11.6 Предделитель

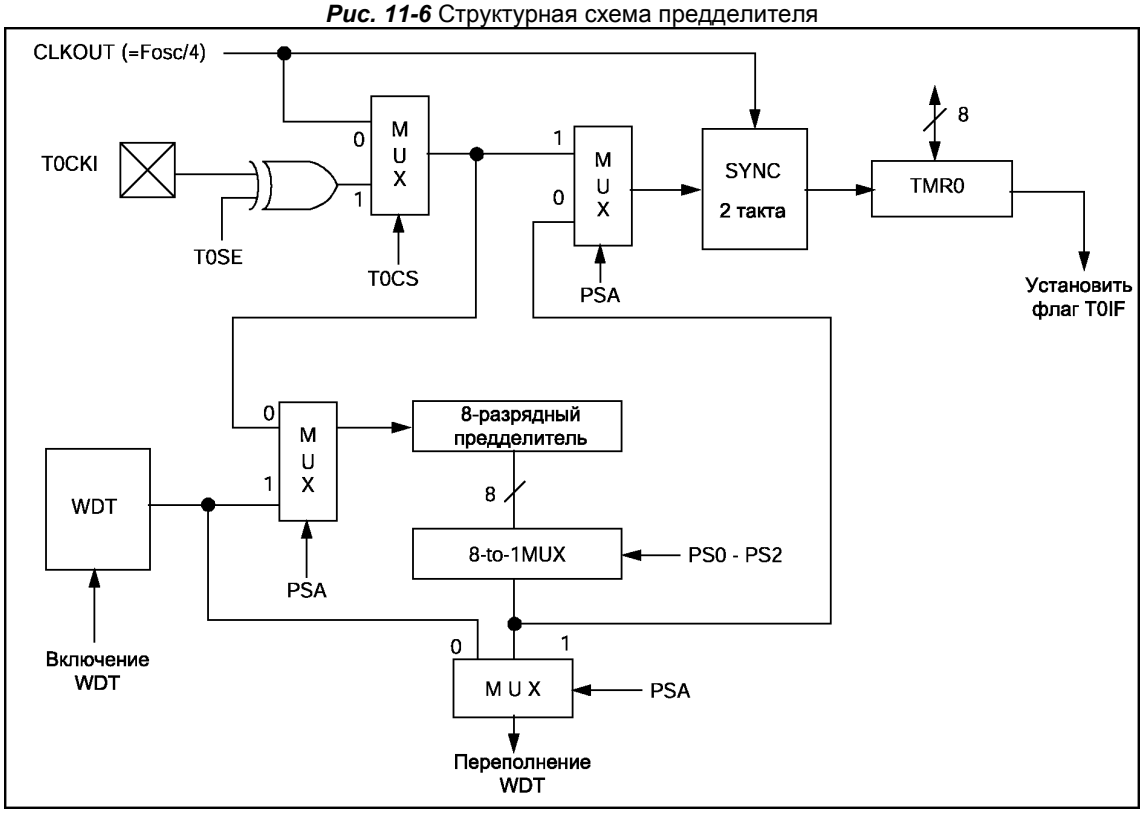
8-разрядный счетчик может работать как предделитель TMR0 или выходной делитель WDT (см. рисунок 11-6). Для простоты описания этот счетчик всегда будем называть "предделитель". Использование предделителя пред TMR0 означает, что WDT работает без предделителя, и наоборот.

Примечание. Существует только один предделитель, который может быть включен перед TMR0 или WDT.

Коэффициент деления предделителя определяется битами PSA и PS2:PS0 (OPTION_REG<3:0>).

Если предделитель включен перед TMR0, любые команды записи в TMR0 (например, CLRF TMR0; MOVWF TMR0; BSF TMR0,x и т.д.) сбрасывают предделитель. Когда предделитель подключен к WDT, команда CLRWDT сбросит предделитель вместе с WDT. Предделитель также очищается при сбросе микроконтроллера.

Рис. 11-6 Структурная схема предделителя



Примечание. Биты управления T0CS, T0SE, PS2, PS1, PS0, PSA расположены в регистре OPTION_REG.

11.6.1 Переключение предделителя

Переключение предделителя выполняется программным способом, т.е. переключение можно сделать во время выполнения программы.

Примечание. Для предотвращения случайного сброса микроконтроллера следует выполнять переключение предделителя от TMR0 к WDT как показано в примере 11-1, даже если WDT выключен.

В примере 11-1 первая часть изменения регистра OPTION_REG не должна выполняться, если желаемый коэффициент предделителя отличный от 1:1. Если требуется настройка коэффициента предделителя 1:1, то необходимо установить промежуточное значение коэффициента (отличное от 1:1), а затем установить коэффициент предделителя 1:1 в последней части изменения OPTION_REG.

Переключение предделителя от WDT к TMR0 смотрите в примере 11-2.

Пример 11-1 Переключения предделителя от TMR0 к WDT

```
1)      BSF         STATUS, RP0         ; Банк 1
2)      MOVLW       b'xx0x0xxx'         ; Выбрать источник тактового сигнала и
3)      MOVWF       OPTION_REG          ; коэффициент предделителя, отличный от 1:1
4)      BCF         STATUS, RP0         ; Банк 0
5)      CLRF        TMR0                ; Сбросить TMR0 и предделитель
6)      BSF         STATUS, RP1         ; Банк 1
7)      MOVLW       b'xxxx1xxx'         ; Включить предделитель перед WDT,
8)      MOVWF       OPTION_REG          ; но не выбирать коэффициент деления
9)      CLRWDT                          ; Сбросить WDT и предделитель
10)     MOVLW       b'xxxx1xxx'         ; Выбрать новое значение коэффициента
11)     MOVWF       OPTION_REG          ; предделителя
12)     BCF         STATUS, RP0         ; Банк 0
```

Примечания к примеру. Если желаемое значение коэффициента деления отличное от 1:1, то строки 2 и 3 в текст программы не должны включаться. Если требуется настройка коэффициента предделителя 1:1, то необходимо установить промежуточное значение коэффициента (отличное от 1:1) в строках 2 и 3, а затем установить коэффициент предделителя 1:1 в строках 10 и 11.

Пример 11-2 Переключения предделителя от WDT к TMR0

```
        CLRWDT                          ; Сбросить WDT и предделитель
        BSF         STATUS, RPO         ; Банк 1
        MOVLW       b'xxxx0xxx'         ; Включить предделитель перед TMR0 и
        MOVWF       OPTION_REG          ; выбрать новое значение коэффициента деления
        BCF         STSTUS, RPO         ; Банк 0
```

11.6.2 Инициализация

В примере 11-3 показана инициализация TMR0 с внутренним источником тактового сигнала, а в примере 11-4 представлена инициализация TMR0 с внешним источником тактового сигнала.

Пример 11-3 Инициализация TMR0 (внутренний источник тактового сигнала)

```
                CLRF    TMR0                ; Сброс TMR0
                CLRF    INTCON              ; Выключить прерывания и сбросить T0IF
                BSF     STATUS, RP0         ; Банк 1
                MOVLW   0xC3                ; Выключить подтягивающие резисторы на PORTB,
                MOVWF   OPTION_REG          ; прерывания по переднему фронту сигнала на RB0
                                            ; TMR0 инкрементируется от внутреннего тактового сигнала
                                            ; предделитель 1:16.
                BCF     STATUS, RP0         ; Банк 0
;**             BSF     INTCON, T0IE        ; Разрешить прерывания от TMR0
;**             BSF     INTCON, GIE         ; Разрешить все прерывания
;
; Если прерывания от TMR0 выключены, то выполняйте проверку бита переполнения.
;
T0_OVFL_WAIT
                BTFSS   INTCON, T0IF
                GOTO    T0_OVFL_WAIT
; Произошло переполнение TMR0
```

Пример 11-4 Инициализация TMR0 (внешний источник тактового сигнала)

```
                CLRF    TMR0                ; Сброс TMR0
                CLRF    INTCON              ; Выключить прерывания и сбросить T0IF
                BSF     STATUS, RP0         ; Банк 1
                MOVLW   0x37                ; Включить подтягивающие резисторы на PORTB,
                MOVWF   OPTION_REG          ; прерывания по заднему фронту сигнала на RB0
                                            ; TMR0 инкрементируется от внешнего тактового сигнала
                                            ; предделитель 1:256.
                BCF     STATUS, RP0         ; Банк 0
;**             BSF     INTCON, T0IE        ; Разрешить прерывания от TMR0
;**             BSF     INTCON, GIE         ; Разрешить все прерывания
;
; Если прерывания от TMR0 выключены, то выполняйте проверку бита переполнения.
;
T0_OVFL_WAIT
                BTFSS   INTCON, T0IF
                GOTO    T0_OVFL_WAIT
; Произошло переполнение TMR0
```

11.7 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: В моей программе выполняется отсчет времени, но часы работают не точно или вовсе теряют значение времени.

Ответ 1:

Если Вы выполняете проверку переполнения TMR0 следующем образом:

```
wait    MOVF    TMR0,W          ; Пересылка TMR0 в W
        BTFSS   STATUS,Z        ; Проверка на нуль.
                                ; Если нуль, то завершить цикл,
        GOTO    wait            ; иначе продолжать ожидать переполнения TMR0
```

Возможны два варианта потери точности часов:

1. Если приращение TMR0 выполняется от внутреннего или внешнего источника тактового сигнала, имеющий достаточно высокую частоту, переполнение таймера может произойти в течении двух циклов команды GOTO. В этом случае необходимо использовать предделитель перед TMR0 или проверку на переполнение TMR0 выполнять по другому:

```
wait    MOVLW   03
        SUBWF   TMR0,W
        BTFSS   STATUS,Z
        GOTO    wait
```

2. При записи TMR0 следующие два цикла команд таймер не инкрементируется. Часто необходимо отсчитать определенный интервал времени, например, десятичное число 100. Вы записываете в TMR0 значение 156 (256 - 100 = 156). Из-за того, что в течение двух машинных циклов приращение TMR0 не происходит, Вы должны записать значение 158 (при внутреннем тактовом сигнале и коэффициенте деления 1:1).

11.8 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с таймером TMR0 в микроконтроллерах PICmicro MCU:

Документ	Номер
Frequency Counter Using PIC16C5X Частотомер на PIC16C5X	AN592
A Clock Design using the PIC16C54 for LED Display and Switch Inputs Проект часов на PIC16C54 со светодиодным индикатором и кнопками управления	AN590

Раздел 12. Таймер TMR1

Содержание

12.1 Введение

TMR1 - 16-разрядный таймер/счетчик, состоящий из двух 8-разрядных регистров (TMR1H и TMR1L), доступных для чтения и записи. Счет выполняется в спаренном регистре TMR1 (TMR1H : TMR1L), инкрементируя его значение от 0000h до FFFFh, далее считает с 0000h. При переполнении счетчика устанавливается в ‘1’ флаг прерывания TMR1IF. Само прерывание можно разрешить/запретить установкой/сбросом бита TMR1IE.

TMR1 может работать в одном из трех режимах:

- Режим таймера;
- Режим синхронного счетчика;
- Режим асинхронного счетчика.

Включение модуля TMR1 осуществляется установкой бита TMR1ON в ‘1’ (T1CON<0>).

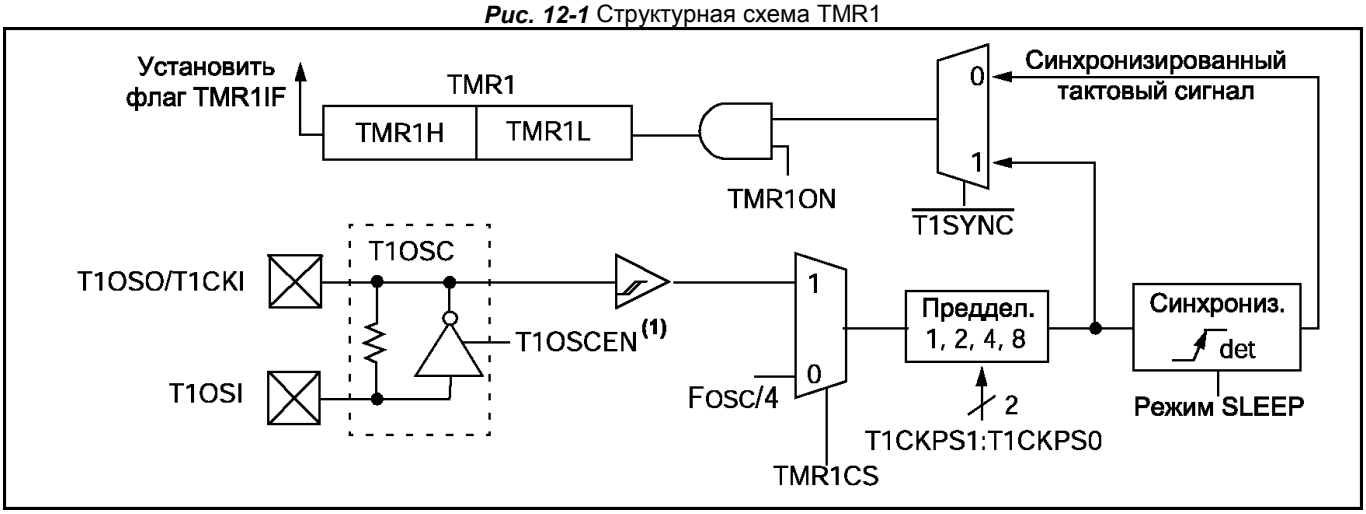
Битом TMR1CS (T1CON<1>) выбирается источник тактовых импульсов. В режиме таймера TMR1 инкрементируется на каждом машинном цикле. Если TMR1 работает с внешним источником тактового сигнала, то приращение происходит по каждому переднему фронту сигнала.

TMR1 имеет внутренний вход сброса от модуля CCP.

Когда включен генератор тактовых импульсов (T1OSCEN=1), выводы T1OSI и T1OSO настроены как входы. Значение битов TRISC<1:0> игнорируется, а чтение данных с этих выводов дает результат '0'.

Управляющие биты TMR1 находятся в регистре T1CON.

Рис. 12-1 Структурная схема TMR1



Примечание. Если T1OSCEN=0, то инвертирующий элемент и резистивная обратная связь выключены для уменьшения тока потребления.

12.2 Управляющий регистр

Регистр T1CON

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
-	-	**T1CKPS**	**T1CKPS**	**T1OSCE**	**-T1SYNC**	**TMR1CS**	**TMR1ON**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

биты 7-6: **Не реализованы**: читаются как '0'

биты 5-4: **T1CKPS1:T1CKPS0:** Выбор коэффициента деления предделителя TMR1
11 = 1:8
10 = 1:4
01 = 1:2
00 = 1:1

бит 3: **T1OSCEN:** Включение тактового генератора TMR1
1 = генератор включен
0 = генератор выключен (инвертирующий элемент и резистивная обратная связь выключены для уменьшения тока потребления)

бит 2: **-T1SYNC:** Синхронизация внешнего тактового сигнала
TMR1CS = 1
1 = не синхронизировать внешний тактовый
0 = синхронизировать внешний тактовый

TMR1CS = 0
Значение бита игнорируется

бит 1: **TMR1CS:** Выбор источника тактового сигнала
1 = внешний источник с вывода T1OSO/T1CKI (активным является передний фронт сигнала)
0 = внутренний источник Fosc/4

бит 0: **TMR1ON:** Включение модуля TMR1
1 = включен
0 = выключен

12.3 Работа TMR1 в режиме таймера

Приращение таймера происходит от внутреннего сигнала $F_{OSC}/4$, когда бит TMR1CS (T1CON<1>) сброшен в ‘0’. В этом режиме бит синхронизации T1SYNC (T1CON<2>) игнорируется, потому что внутренний тактовый сигнал всегда синхронизирован.

12.4 Работа TMR1 в режиме синхронного счетчика

Работа TMR1 от внешнего источника тактового сигнала выбирается установкой бита TMR1CS в ‘1’. В этом режиме приращение таймера происходит по каждому переднему фронту сигнала на выводе T1OSI (если T1OSCEN=1) или T1OSO/T1CKI (если T1OSCEN=0).

Если -T1SYNC=0, то активный фронт внешнего тактового сигнала синхронизируется с внутренним тактовым сигналом на выходе асинхронного предделителя.

В SLEEP режиме микроконтроллера счетчик не будет инкрементироваться (при наличии тактового сигнала), т.к. синхронизатор выключен (предделитель продолжает счет тактовых импульсов).

12.4.1 Синхронизация внешнего тактового сигнала

Когда используется синхронизация внешнего тактового сигнала, необходимо учитывать ряд требований. Фаза внешнего сигнала синхронизируется с внутренним тактовым сигналом микроконтроллера (период T_{OSC}), из-за чего возникает задержка от получения активного фронта сигнала до приращения TMR1.

Если коэффициент предделителя 1:1, то внешний тактовый сигнал поступает непосредственно на вход синхронизатора. Синхронизация T1CKI с таковым сигналом микроконтроллера усложняется из-за опроса выхода синхронизатора в машинные циклы Q2 и Q4, поэтому длительность высокого или низкого логического уровня внешнего сигнала должна быть не меньше $2T_{OSC}$ (плюс небольшая задержка внутренней RC цепи 20нс). Дополнительную информацию смотрите в разделе "Электрические характеристики" (параметры 45, 46 и 47).

Если предделитель имеет коэффициент деления отличный от 1:1, то на вход синхронизатора поступает сигнал с асинхронного предделителя. Период сигнала T1CKI должен быть не менее $4T_{OSC}$ (плюс небольшая задержка внутренней RC цепи 40нс) деленное на коэффициент предделителя. Дополнительное требование: высокий и низкий логический уровень внешнего сигнала должен быть не менее 10нс. Смотрите параметры 40, 42, 45, 46 и 47 в разделе "Электрические характеристики".

12.5 Работа TMR1 в режиме асинхронного счетчика

Если бит -T1SYNC (T1CON<2>) установлен в ‘1’, внешний тактовый сигнал TMR1 не будет синхронизироваться с внутренним тактовым сигналом микроконтроллера, таймер продолжает работать в SLEEP режиме микроконтроллера. Переполнение таймера вызовет "пробуждение" микроконтроллера, если разрешено прерывание от TMR1. Однако требуется осторожность при записи/чтении TMR1 (см. раздел 12.5.2). Работа TMR1 в SLEEP режиме микроконтроллера позволяет реализовать часы реального времени.

В асинхронном режиме TMR1 не может использоваться для захвата/сравнения данных модуля CCP.

12.5.1 Параметры внешнего не синхронизированного тактового сигнала

Если бит -T1SYNC=1, то приращение таймера выполняется асинхронно. Минимальная длительность логических уровней внешнего тактового сигнала смотрите в разделе "Электрические характеристики" (параметры 45, 46 и 47).

12.5.2 Чтение/запись TMR1 в асинхронном режиме

Чтение TMR1H или TMR1L, во время счета в асинхронном режиме, гарантирует получение текущего значения счетчика (реализовано аппаратно). Однако пользователь должен иметь в виду, что чтение 16-разрядного значения выполняется по байтно. Это накладывает некоторые ограничения, т.к. таймер может переполниться между чтениями байт.

Запись в TMR1 рекомендуется выполнять после остановки таймера. Запись в регистры TMR1 во время приращения таймера может привести к непредсказуемому значению регистра.

Чтение 16 - разрядного значения требуется некоторой осторожности, т.к. требуется два цикла чтения для получения всех 16 разрядов. В примере 12-1 показано, почему нельзя выполнять прямое чтение 16 - разрядного регистра таймера.

Пример 12-1 Чтение 16 - разрядного регистра TMR1

TMR1	**Последовательность 1**		**Последовательность 2**	
	Действие	**TMPH:TMPL**	**Действие**	**TMPH:TMPL**
04EFh	Чтение TMR1L	xxxxh	Чтение TMR1H	xxxxh
0500h	Запись в TMPL	xxFFh	Запись в TMPH	04xxh
0501h	Чтение TMR1H	xxFFh	Чтение TMR1L	04xxh
0502h	Запись в TMPH	05FFh	Запись в TMPL	0401h

В примере 12-2 представлена рекомендованная последовательность операций чтения 16 - разрядного значения TMR1 в асинхронном режиме с решением проблем переполнения, показанных в примере 12-1. В примере 12-2 таймер не останавливается.

Пример 12-2 Чтение 16-разрядного значения TMR1

```
; Выключить все прерывания
            MOVF        TMR1H, W      ; Чтение старшего байта
            MOVWF       TMPH          ;
            MOVF        TMR1L, W      ; Чтение младшего байта
            MOVWF       TMPL          ;
            MOVF        TMR1H, W      ; Чтение старшего байта
            SUBWF       TMPH, W       ; Сравнение с предыдущим чтением
            BTFSC       STATUS, Z     ;
            GOTO        CONTINUE      ; 16-разрядное значение прочитано правильно

                                      ; Возможно между чтениями байтов произошло
                                      ; переполнение таймера
                                      ; Прочитать значения заново
            MOVF        TMR1H, W      ; Чтение старшего байта
            MOVWF       TMPH          ;
            MOVF        TMR1L, W      ; Чтение младшего байта
            MOVWF       TMPL          ;
CONTINUE:
; Включить прерывания (если необходимо)
```

16 - разрядное значение записывается непосредственно в регистры TMR1. Сначала нужно очистить регистр TMR1L, чтобы в запасе было большое число тактов TMR1 прежде, чем произойдет перенос из младшего регистра TMR1L в TMR1H. Выполнить запись в TMR1H, а затем записать значение в TMR1L. Эта последовательность действий показана в примере 12-3.

Пример 12-3 Запись 16-разрядного значения в TMR1

```
; Выключить все прерывания
            CLRF        TMR1L         ; Очистить младший байт
                                      ; для предотвращения переноса в TMR1H
            MOVLW       HI_BYTE       ; Значение для TMR1H
            MOVWF       TMR1H, F      ; Записать старший байт
            MOVLW       LO_BYTE       ; Значение для TMR1L
            MOVWF       TMR1H, F      ; Записать младший байт
; Включить прерывания (если необходимо)
CONTINUE:
```

12.6 Генератор TMR1

Резонатор подключается к выводам T1OSI (вход) и T1OSO (выход усилителя). Включение генератора производится установкой бита T1OSEN в регистре T1CON<3>. Максимальная частота резонатора 200КГц. Тактовый генератор (идентичный LP генератору) позволяет работать TMR1 в SLEEP режиме микроконтроллера. В основном предназначен для кварцевого резонатора 32кГц., что позволяет реализовать часы реального времени.

Примечание. В SLEEP режиме микроконтроллера таймер продолжает инкрементироваться. Это позволяет использовать TMR1 для реализации часов реального времени.

Пользователь должен обеспечить программную задержку, чтобы гарантировать надлежащий запуск генератора. В таблице 12-1 указаны рекомендуемые значения конденсаторов для генератора TMR1.

Таблица 12-1 Выбор конденсаторов для генератора TMR1

Тип генератора	Частота	C1	C2
LP	32 кГц	33 пФ	33 пФ
	100 кГц	15 пФ	15 пФ
	200 кГц	15 пФ	15 пФ
Протестированные резонаторы:			
32.768кГц	Epson C-001 R32.768K-A		±20 PPM
100кГц	Epson C-2 100.00 KC-P		±20 PPM
200кГц	STD XTL 200.000 kHz		±20 PPM

Примечания:

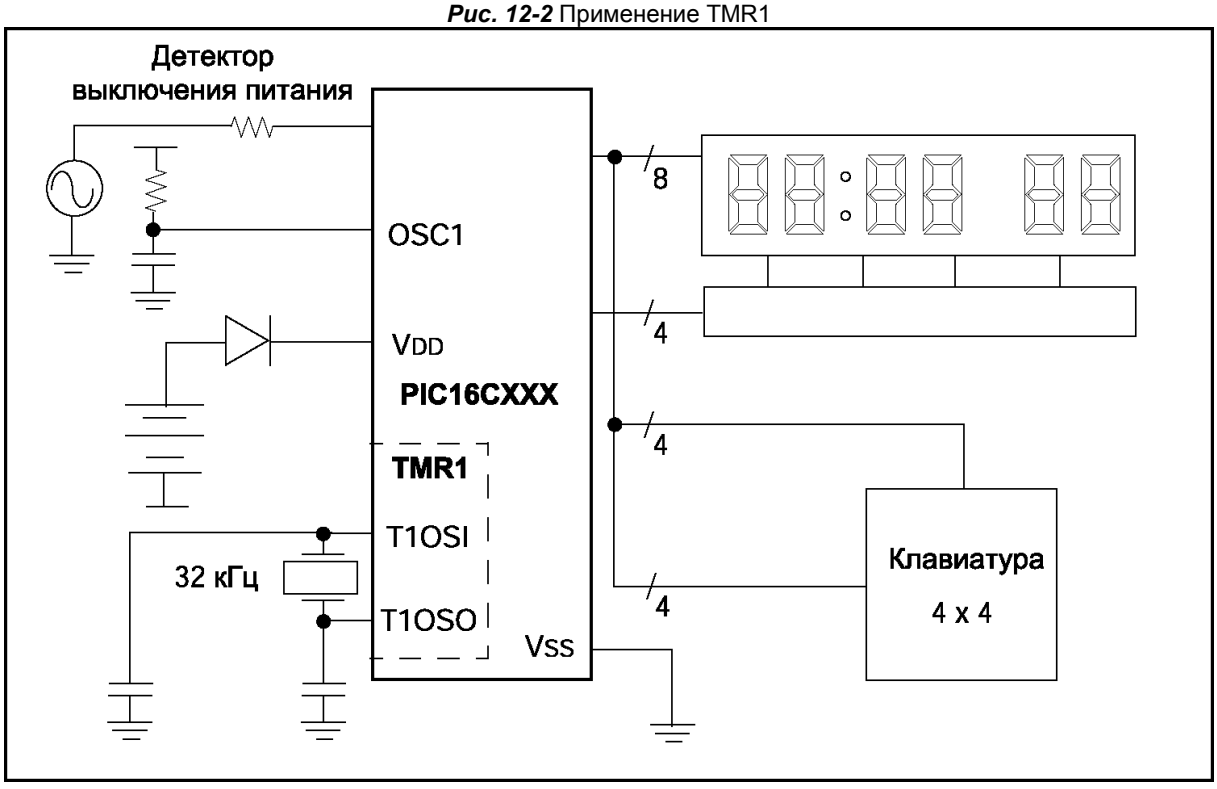
1. Большая емкость увеличивает стабильность генератора, но также увеличивает время запуска.
2. Каждый резонатор имеет собственные характеристики. Проконсультируйтесь у производителя резонаторов для правильного подбора внешних компонентов.

12

12.6.1 Типовое применение

Генератор TMR1 как правило используется в приложениях, в которых требуется сохранение работы микроконтроллера в режиме реального времени с минимальным энергопотреблением. TMR1 позволяет перевести микроконтроллер в SLEEP режим, а таймер будет продолжать инкрементироваться. Когда произойдет переполнение таймера, микроконтроллер выйдет из режима SLEEP по возникшему прерыванию, чтобы была возможность изменить служебные регистры.

Рис. 12-2 Применение TMR1



12.7 Работа в SLEEP режиме

Когда TMR1 работает в режиме асинхронного счетчика, регистр TMR1 продолжает инкрементироваться по каждому активному фронту внешнего сигнала (с учетом предделителя) в SLEEP режиме микроконтроллера. При переполнении TMR1 устанавливается в '1' флаг прерывания TMR1IF, и если разрешены прерывания от TMR1, микроконтроллер выйдет из режима SLEEP.

Включенный генератор TMR1 увеличит ток потребления устройства, т.е. потребление логики микроконтроллера в SLEEP режиме не увеличится (будут присутствовать только токи утечки). Это ток потребление генератора TMR1 и остальной схемы TMR1.

12.8 Сброс TMR1 триггером модуля CCP

Если модуль CCP1 или CCP2 работает в режиме сравнения с триггером специальных функций (CCP1M3 : CCP1M0=1011), то сигнал триггера сбросит TMR1.

Примечание. Сигнал с триггера специальных функций модуля CCP1 не будет устанавливать флаг TMRIF в ‘1’.

TMR1 должен работать в режиме синхронизированного внешнего тактового сигнала или внутреннего тактового сигнала. В асинхронном режиме эта функция не работает.

Когда запись в TMR1 совпадает с сигналом сброса от триггера специальных событий, приоритет отдается записи в TMR1.

В этом режиме модуля CCP период сброса TMR1 сохраняется в регистрах CCPRxH:CCPRxL.

12.9 Сброс регистров TMR1 (TMR1H, TMR1L)

Регистры TMR1H и TMR1L не сбрасываются в 00h при сбросе по включению питания POR и других видах сброса, кроме сброса по сигналу триггера специальных событий модуля CCP1 или CCP2.

Регистр T1CON сбрасывается в 00h при сбросе POR и BOR (TMR1 выключается, коэффициент предделителя равен 1:1). При всех остальных видах сброса значение регистра T1CON не изменяется.

12.10 Предделитель TMR1

Предделитель TMR1 очищается при записи в регистр TMR1L или TMR1H.

Таблица 12-2 Регистры и биты, связанные с работой TMR1

Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Сброс POR, BOR	Другие сбросы
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000u
PIR	TMR1IF(1)								0	0
PIE	TMR1IE(1)								0	0
TMR1L	Младший байт 16-разрядного таймера 1								xxxx xxxx	uuuu uuuu
TMR1H	Старший байт 16-разрядного таймера 1								xxxx xxxx	uuuu uuuu
T1CON	-	-	T1CKPS1	T1CKPS0	T1OSCEN	-T1SYNC	TMR1CS	TMR1ON	--00 0000	--uu uuuu

Обозначения: - = не используется, читается как 0; u = не изменяется; x = не известно; q = зависит от условий.
Затененные биты на работу не влияют.

Примечания:

1. Расположение битов смотрите в технической документации на микроконтроллер.
2. В некоторых микроконтроллерах эти биты могут обозначаться как GPIE и GPIF.

12.11 Инициализация

В данной главе будут представлены примеры инициализации TMR1 в каждом из трех возможных режимов. В примере 12-4 показана инициализация TMR1 с внутренним тактовым сигналом, а в примере 12-5 - инициализация TMR1 с внешним тактовым сигналом. Инициализацию TMR1 с выключенным генератором смотрите в примере 12-6.

Пример 12-4 Инициализация TMR1 (внутренний тактовый сигнал)

```
        CLRF        T1CON           ; Выключить TMR1, внутренний тактовый сигнал,
                                    ; генератор TMR1 выключен, предделитель = 1:1
        CLRF        TMR1H           ; Очистить старший байт регистра TMR1
        CLRF        TMR1L           ; Очистить младший байт регистра TMR1
        CLRF        INTCON          ; Выключить прерывания
        BSF         STATUS, RP0     ; Банк 1
        CLRF        PIE1            ; Выключить периферийные прерывания
        BCF         STATUS, RP0     ; Банк 0
        CLRF        PIR1            ; Очистить флаги периферийных прерываний
        MOVLW       0x30            ; Внутренний тактовый сигнал с предделителем 1:8
        MOVWF       T1CON           ; TMR1 и генератор TMR1 выключены
        BSF         T1CON, TMR1ON   ; Включить TMR1
;
; Прерываний от TMR1 выключены, проверяйте бит переполнения
;
T1_OVFL_WAIT
        BTFSS       PIR1, TMR1IF
        GOTO        T1_OVFL_WAIT
;
; Переполнение TMR1
;
        BCF         PIR1, TMR1IF
```

Пример 12-5 Инициализация TMR1 (внешний тактовый сигнал)

```
        CLRF        T1CON           ; Выключить TMR1, внутренний тактовый сигнал,
                                    ; генератор TMR1 выключен, предделитель = 1:1
        CLRF        TMR1H           ; Очистить старший байт регистра TMR1
        CLRF        TMR1L           ; Очистить младший байт регистра TMR1
        CLRF        INTCON          ; Выключить прерывания
        BSF         STATUS, RP0     ; Банк 1
        CLRF        PIE1            ; Выключить периферийные прерывания
        BCF         STATUS, RP0     ; Банк 0
        CLRF        PIR1            ; Очистить флаги периферийных прерываний
        MOVLW       0x32            ; Внешний синхронизированный тактовый сигнал
        MOVWF       T1CON           ; с предделителем 1:8, TMR1 и генератор TMR1 выключены
        BSF         T1CON, TMR1ON   ; Включить TMR1
;
; Прерываний от TMR1 выключены, проверяйте бит переполнения
;
T1_OVFL_WAIT
        BTFSS       PIR1, TMR1IF
        GOTO        T1_OVFL_WAIT
;
; Переполнение TMR1
;
        BCF         PIR1, TMR1IF
```

Пример 12-6 Инициализация TMR1 (внешний сигнал от тактового генератора TMR1)

```
        CLRF        T1CON               ; Выключить TMR1, внутренний тактовый сигнал,
                                        ; генератор TMR1 выключен, предделитель = 1:1
        CLRF        TMR1H               ; Очистить старший байт регистра TMR1
        CLRF        TMR1L               ; Очистить младший байт регистра TMR1
        CLRF        INTCON              ; Выключить прерывания
        BSF         STATUS, RP0         ; Банк 1
        CLRF        PIE1                ; Выключить периферийные прерывания
        BCF         STATUS, RP0         ; Банк 0
        CLRF        PIR1                ; Очистить флаги периферийных прерываний
        MOVLW       0x3E                ; Внешний не синхронизированный сигнал от генератора
        MOVWF       T1CON               ; с предделителем 1:8, TMR1 и генератор TMR1 выключены
        BSF         T1CON, TMR1ON       ; Включить TMR1
;
; Прерываний от TMR1 выключены, проверяйте бит переполнения
;
T1_OVFL_WAIT
        BTFSS       PIR1, TMR1IF
        GOTO        T1_OVFL_WAIT
;
; Переполнение TMR1
;
        BCF         PIR1, TMR1IF
```

12.12 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Используя TMR1 не удается получить точный отсчет времени.
Ответ 1:
Существует несколько причин из-за чего это может происходить:

1. Вы не должны выполнять запись в регистры TMR1 в моменты, когда может возникнуть потеря отсчета. В большинстве случаев не должна выполняться запись в регистр TMR1L, если не выполнены условия записи в регистр TMR1H. Как правило запись в регистр TMR1H используется, чтобы прерывание по переполнению TMR1 возникло быстрее (чем полный счет 16-разрядного значения).
2. Необходимо гарантировать качественное размещение компонентов на печатной плате, чтобы внешние помехи минимально влияли на генератор TMR1.

12.13 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с таймером TMR1 в микроконтроллерах PICmicro MCU:

Документ	Номер
Using Timer1 in Asynchronous Clock Mode Использование TMR1 в режиме асинхронного счетчика	AN580
Low Power Real Time Clock Часы реального времени с малым энергопотреблением	AN582
Yet another Clock using the PIC16C92X Часы на микроконтроллере PIC16C92X	AN649

Раздел 13. Таймер TMR2

Содержание

13.1 Введение

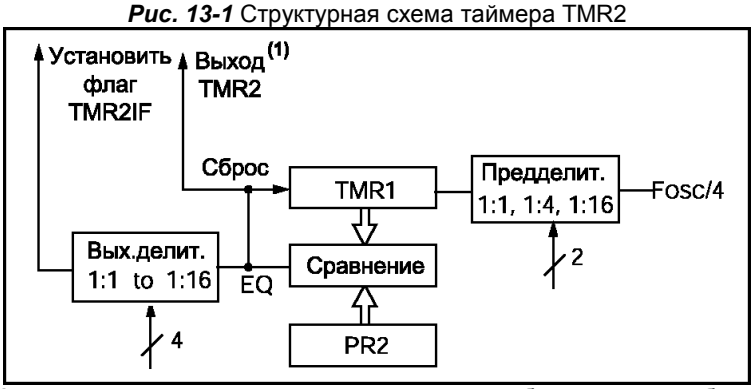
TMR2 – 8-разрядный таймер с программируемыми предделителем и выходным делителем, 8-разрядным регистром периода PR2. При максимальном значении коэффициентов деления тактового сигнала TMR2 соответствует 16 - разрядному таймеру.

TMR2 может быть опорным таймером для CCP модуля в ШИМ режиме.

На рисунке 13-1 показана структурная схема TMR2.

Выходной делитель считает число раз достижения TMR2 значения, записанного в регистре PR2. Это может быть полезно, когда необходимо организовать переход на сервисную подпрограмму через определенный промежуток времени.

Рис. 13-1 Структурная схема таймера TMR2



Примечание 1. TMR2 может использоваться для программного выбора скорости обмена данными модуля SSP.

13.2 Управляющий регистр

Регистр T2CON

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
-	**TOUTPS**	**TOUTPS**	**TOUTPS**	**TOUTPS**	**TMR2ON**	**T2CKPS**	**T2CKPS**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **Не реализован**: читается как '0'

биты 6-3: **TOUTPS3:TOUTPS0:** Выбор коэффициента выходного делителя TMR2
0000 = 1:1
0001 = 1:2
:
:
1111 = 1:16

бит 2: **TMR2ON:** Включение модуля TMR2
1 = включен
0 = выключен

биты 1-0: **T2CKPS1:T2CKPS0:** Выбор коэффициента деления предделителя TMR2
00 = 1:1
01 = 1:4
1x = 1:16

13.3 Источник тактового сигнала

TMR2 имеет один источник тактового сигнала $F_{OSC}/4$. Сигнал поступает через предделитель с программируемым коэффициентом деления (1:1, 1:4 или 1:16), определяемый битами T2CKPS1:T2CKPS0 (T2CON<1:0>).

13.4 Регистр таймера TMR2 и периода PR2

Регистр TMR2 доступен для записи/чтения и очищается при любом виде сброса. TMR2 считает, инкрементируя от 00h до значения в регистре PR2, затем сбрасывается в 00h на следующем машинном цикле. Регистр PR2 доступен для записи и чтения.

Регистр TMR2 очищается при сбросе от WDT, POR, -MCLR и BOR.

После сброса значение регистра PR2 равно FFh.

Для уменьшения энергопотребления таймер TMR2 может быть выключен сбросом бита TMR2ON (T2CON<2>) в ‘0’.

13.5 Сигнал TMR2

Сигнал переполнения TMR2 (до выходного делителя) передается в:

- Выходной делитель TMR2;
- Модуль SSP для управления скоростью передачи данных.

Четыре управляющих бита устанавливают коэффициент выходного делителя (от 1:1 до 1:16 включительно). После переполнения выходного делителя устанавливается в '1' флаг прерывания TMR2IF. Эта функция используется для упрощения программного обеспечения, поскольку прерывания будут возникать только после переполнения выходного делителя.

Сигнал переполнения TMR2 также поступает в модуль SSP для программного выбора скорости передачи данных.

13

13.6 Очистка предделителя и выходного делителя TMR2

Счетчик предделителя и выходного делителя сбрасываются в случае:

- Записи в регистр TMR2;
- Записи в регистр T2CON;
- Любого вида сброса микроконтроллера (POR, BOR, сброс WDT или активный сигнал -MCLR).

Примечание. Регистр TMR2 не очищается при записи в T2CON.

13.7 Работа в SLEEP режиме

В SLEEP режиме микроконтроллера приращение TMR2 не инкрементируется. Текущее значение предделителя и выходного делителя не изменяются, после выхода из SLEEP режима продолжают счет тактовых импульсов.

Таблица 13-1 Регистры и биты, связанные с работой TMR2

Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Сброс POR, BOR	Другие сбросы
INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
PIR	TMR2IF(1)								0	0
PIE	TMR2IE(1)								0	0
TMR2	Регистр таймера 2								0000 0000	0000 0000
T2CON	-	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	-uuu uuuu
PR2	Регистр периода таймера 2								1111 1111	1111 1111

Обозначения: - = не используется, читается как 0; u = не изменяется; x = не известно; q = зависит от условий. Затененные биты на работу не влияют.

Примечания:

1. Расположение битов смотрите в технической документации на микроконтроллер.
2. В некоторых микроконтроллерах эти биты могут обозначаться как GPIE и GPIF.

13.8 Инициализация

В примере 13-1 показана инициализация TMR2, включая настройку таймера и выходного делителя.

Пример 13-1 Инициализация TMR2

```
        CLRF        T2CON           ; Выключить TMR2, предделитель = 1:1,
                                    ; выходной делитель = 1:1
        CLRF        TMR2            ; Очистить регистр TMR2
        CLRF        INTCON          ; Выключить прерывания
        BSF         STATUS, RP0     ; Банк 1
        CLRF        PIE1            ; Выключить периферийные прерывания
        BCF         STATUS, RP0     ; Банк 0
        CLRF        PIR1            ; Очистить флаги периферийных прерываний
        MOVLW       0x72            ; Предделитель = 1:15, выходной делитель = 1:16,
        MOVWF       T2CON           ; TMR2 выключен
        BSF         T2CON, TMR2ON   ; Включить TMR2
;
; Прерываний от TMR2 выключены, проверяйте бит переполнения
;
T2_OVFL_WAIT
        BTFSS       PIR1, TMR2IF    ; Произошло переполнение TMR2?
        GOTO        T2_OVFL_WAIT    ; Нет, оставаться в цикле
;
; Переполнение TMR2
;
        BCF         PIR1, TMR2IF
```

13.9 Ответы на часто задаваемые вопросы

На момент выполнения перевода в оригинальной технической документации вопросы отсутствовали. Если у вас есть вопрос, задайте его, написав нам письмо по адресу support@microchip.ru.

13.10 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с таймером TMR2 в микроконтроллерах PICmicro MCU:

Документ	Номер
Using the CCP Module Применение модуля CCP	AN594
Air Flow Control using Fuzzy Logic Управление потоками воздуха на основе Fuzzy Logic	AN600
Adaptive Differential Pulse Code Modulation using PICmicros Реализация ADPCM на микроконтроллерах PICmicro	AN643

Раздел 14. Модуль CCP

Содержание

14.1 Введение

Каждый модуль CCP содержит 16-разрядный регистр, который может использоваться в качестве:

- 16-разрядного регистра захвата данных;
- 16-разрядного регистра сравнения;
- Двух 8-разрядных (ведущий и ведомый) регистров ШИМ.

Работа модулей CCP1 и CCP2 идентична, за исключением функционирования триггера специального события, поэтому в этой технической документации регистры модуля CCP будут обозначаться как показано в таблице 14-1.

Таблица 14-1 Обозначение регистров модуля CCP

Обозначение	CCP1	CCP2	Примечание
CCPxCON	CCP1CON	CCP2CON	Управляющий регистр CCP
CCPRxH	CCPR1H	CCPR2H	Старший байт CCP
CCPRxL	CCPR1L	CCPR2L	Младший байт CCP
CCPx	CCP1	CCP2	Вывод CCP

В таблице 14-2 указаны ресурсы, используемые модулем CCP, а в таблице 14-3 представлено взаимодействие между модулями CCP, где CCPx - первый модуль CCP, CCPy - второй модуль CCP.

Таблица 14-2 Использование таймеров модулями CCP

Режим модуля CCP	Таймер
Захват	TMR1
Сравнение	TMR1
ШИМ	TMR2

Таблица 14-3 Взаимодействие двух модулей CCP

Режим CCPx	Режим CCPy	Взаимодействие
Захват	Захват	Базовый таймер TMR1
Захват	Сравнение	Модуль CCP, работающий в режиме сравнения, должен сбрасывать таймер TMR1 триггером специального события.
Сравнение	Сравнение	Модули CCP, работающие в режиме сравнения, должны сбрасывать таймер TMR1 триггером специального события.
ШИМ	ШИМ	Оба ШИМ имеют одинаковую частоту и фазу (базовый таймер TMR2)
ШИМ	Захват	Нет
ШИМ	Сравнение	Нет

14.2 Управляющий регистр

Регистр CCPxCON

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
-	-	**DCxB1**	**DCxB0**	**CCPxM3**	**CCPxM2**	**CCPxM1**	**CCPxM0**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

биты 7-6: **Не используются:** читаются как ‘0’

биты 5-4: **DCxB1:DCxB0:** Младшие биты длительности импульса ШИМ

Режим захвата
Не используются

Режим сравнения
Не используются

Режим ШИМ
Два младших бита 10 - разрядного значения длительности импульса ШИМ. Восемь старших битов (DCxB9:DCxB2) находятся в CCPRxL.

биты 3-0: **CCPxM3:CCPxM0:** Режим работы модуля CCPx
0000 = модуль CCPx выключен (сброс модуля CCPx)
0100 = захвата по каждому заднему фронту сигнала
0101 = захват по каждому переднему фронту сигнала
0110 = захват по каждому 4-му переднему фронту сигнала
0111 = захват по каждому 16-му переднему фронту сигнала
1000 = сравнение, устанавливает выходной сигнал (устанавливается флаг CCPIF в ‘1’)
1001 = сравнение, сбрасывает выходной сигнал (устанавливается флаг CCPIF в ‘1’)
1010 = сравнение, на выходной сигнал не влияет (устанавливается флаг CCPIF в ‘1’)
1011 = сравнение, триггер специальных функций (устанавливается флаг CCPIF в ‘1’)
11xx = ШИМ режим

14.3 Режим захвата

При возникновении события захвата 16-разрядное значение счетчика TMR1 переписывается в регистры CCPRxL:CCPRxH модуля CPPx. Событием захвата может быть:

- Каждый задний фронт сигнала на входе CCPx;
- Каждый передний фронт сигнала на входе CCPx;
- Каждый 4-й передний фронт сигнала на входе CCPx;
- Каждый 16-й передний фронт сигнала на входе CCPx.

Тип события захвата устанавливается битами CCPxM3:CCPxM0 в регистре CCPxCON. После выполнения захвата устанавливается флаг прерывания CCPxIF в ‘1’, который должен быть сброшен программно. Если происходит события захвата до того, как предыдущие данные были прочитаны, старое значение будет потеряно.

Примечание. В режиме захвата TMR1 должен работать в режиме таймера или синхронизированного счетчика. В режиме асинхронного счетчика операция захвата не работает.

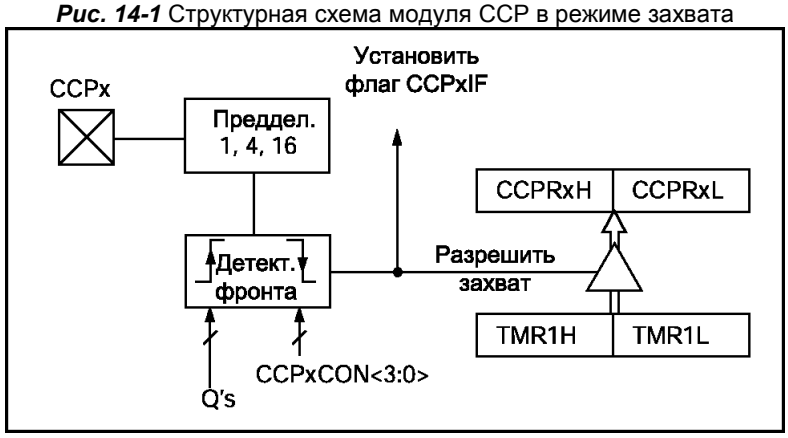
Операция захвата данных не сбрасывает 16-разрядный регистр TMR1 (см. рисунок 14-1). TMR1 может использоваться как базовый таймер для других операций. Промежуток времени между двумя операциями захвата данных может быть легко вычислен (разница между значением первого и второго захвата). При переполнении TMR1 устанавливается флаг прерывания TMR1IF в '1', и если разрешено, произойдет переход на обработку прерываний, что позволяет программно реализовать таймер разрядностью больше чем 16 бит.

14.3.1 Настройка вывода модуля CCP

В режиме захвата порт ввода/вывода CCPx должен быть настроен на вход установкой бита TRIS в’1’.

Примечание. Если порт ввода/вывода CCPx настроен на выход, то захват может происходить командой из программы.

Рис. 14-1 Структурная схема модуля CCP в режиме захвата



Входной предделитель может использоваться, чтобы получить хорошую разрешающую способность при относительно постоянной частоте входного сигнала. Например, если частота входного сигнала постоянна и коэффициент предделителя равен 1:16, то ошибка измерения в течение 16 периодов входного сигнала равна $1T_{CY}$. В этом случае разрешающая способность равна $T_{CY}/16$ (при тактовой частоте 20МГц разрешающая способность 12.5нс). Использовать эту методику целесообразно, когда частота входного сигнала не изменяется в течение 16 периодов. Без предделителя (1:1) каждая выборка имеет разрешающую способность T_{CY}.

14.3.2 Изменение режима работы модуля CCP

Когда изменяется режим работы модуля CCP, необходимо запрещать прерывания сбросом бита CCPxIE в ‘0’ для предотвращения ложных прерываний. После изменение режима работы модуля CCPx, перед разрешением прерываний, необходимо сбросить флаг CCPxIF в ‘0’.

14.3.2.1 Предварительный счетчик событий модуля CCP

Существует четыре режима работы предварительного счетчика событий (определяется битами CCPxM3:CCPxM0). Включение режима захвата очищает предварительный счетчик событий. Переключение между типами событий не очищает счетчик событий, поэтому результат первого захвата после переключения может быть недостоверным. Любой сброс микроконтроллера очищает счетчик событий.

Переключение от одного режима захвата к другому может привести к генерации ложного прерывания. Счетчик событий не очищается при переключении режима захвата, поэтому первое полученное значение может быть недостоверным. В примере 14-1 показано, как нужно производить переключение типа события, чтобы очистить предделитель и не вызвать ложное прерывание.

Пример 14-1 Переключение типа события захвата данных с очисткой предделителя

```
CLRF    CCP1CON          ; Выключить CCP модуль
MOVLW   NEW_CAPT_PS      ; Записать в W новый тип захвата и режим работы CCP
MOVWF   CCP1CON          ; Загрузить настройку в регистр CCP1CON
```

Чтобы очистить предделитель, модуль CCP должен быть настроен в другой режим работы (сравнение, ШИМ) или выключен.

14.3.3 Работа в SLEEP режим микроконтроллера

В SLEEP режиме микроконтроллера TMR1 не будет инкрементироваться (т.к. работает в синхронизированном режиме), но предделитель продолжает считать входные импульсы (т.к. он не синхронизирован). Когда возникает условие захвата данных, устанавливается в '1' флаг CCPxIF, но регистр данных изменен не будет. Если разрешены прерывания от модуля CCP, то микроконтроллер выйдет из режима SLEEP. 16 - разрядное значение TMR1 не передается в регистры захвата, т.к. в SLEEP режиме TMR1 не увеличивался (это значение не имеет никакого смысла). Эта функция позволяет использовать вывод CCP как дополнительный источник внешнего прерывания.

14.3.4 Эффект сброса

После любого вида сброса модуль CCP выключен, а предварительный счетчик событий очищен.

14.4 Режим сравнения

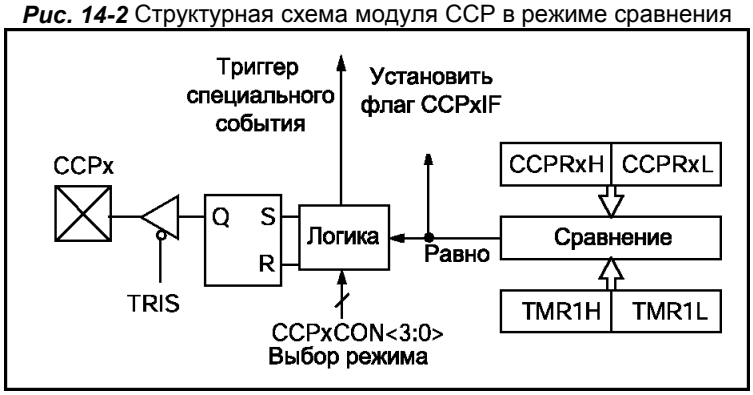
В этом режиме 16-разрядный регистр CCPRx сравнивается со значением TMR1. Как только значения в регистрах становятся одинаковые, модуль CCP изменяет состояние вывода CCPx:

- Устанавливает высокий уровень сигнала;
- Устанавливает низкий уровень сигнала;
- На вывод не воздействует.

Действие при совпадении может быть выбрано битами CCPxM3:CCPxM0 (CCPxCON<3:0>). В момент изменения состояния вывода устанавливается флаг прерывания CCPxIF в ‘1’.

Примечание. В режиме сравнения TMR1 должен работать в режиме таймера или синхронизированного счетчика. В режиме асинхронного счетчика операция сравнения не работает.

Рис. 14-2 Структурная схема модуля CCP в режиме сравнения



14.4.1 Настройка вывода модуля CCP

Для изменения состояния вывода CCPx, он должен быть настроен на выход сбросом соответствующего бита TRIS в ‘0’.

Примечание. При очистке регистра CCPxCON на выводе CCPx появится сигнал низкого уровня, что не является результатом сравнения или данными из выходной защелки.

При выборе режима сравнения на выводе CCP появляется логический уровень сигнала, соответствующий состоянию сравнения. Например, если выбран режим сравнения с переводом выхода в низкий логический уровень, то на выводе будет присутствовать высокий логический уровень, пока не произойдет соответствие.

14.4.2 Программное прерывание

При программной генерации прерывания от модуля CCPx изменение уровня сигнала на выводе CCPx не происходит. Только возникает переход на обработку прерываний, если прерывания от модуля CCPx разрешены.

14.4.3 Триггер специального события

В режиме сравнения модуля CCPx может быть включен триггер специального события.

Триггер специального события CCPx сбрасывает значения таймера TMR1 при каждом положительно выполненном сравнении. Регистр CCPxR является 16-разрядным программируемым регистром периода для TMR1.

Для некоторых микроконтроллеров триггер специального события CCPx сбрасывает значения таймера TMR1 и запускает преобразование АЦП (если модуль АЦП включен).

14.4.4 Работа в SLEEP режим микроконтроллера

В SLEEP режиме микроконтроллера TMR1 не будет инкрементироваться (т.к. работает в синхронизированном режиме) и состояние вывода модуля CCPx не изменяется. По выходу из SLEEP режима модуль CCPx продолжит работать с текущего состояния.

14.4.5 Эффект сброса

После любого вида сброса модуль CCP выключен.

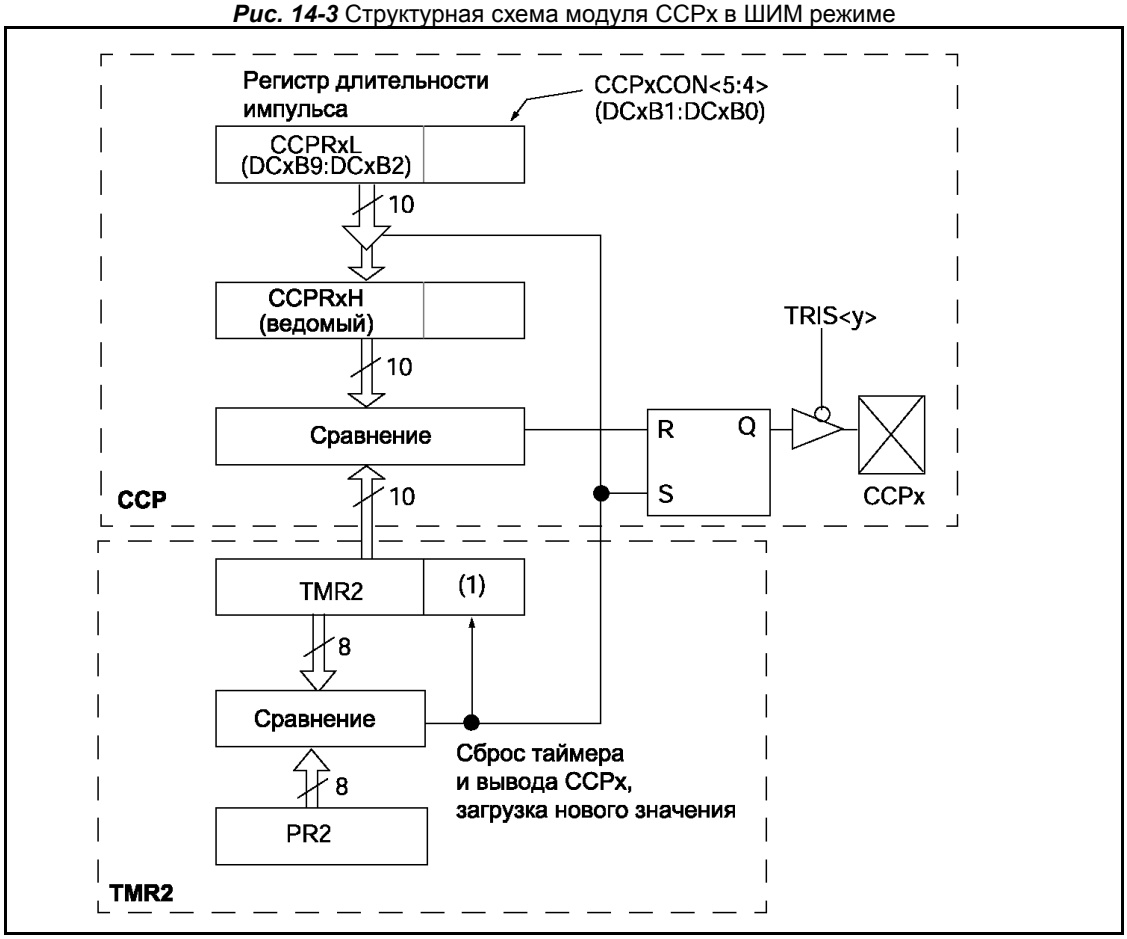
14.5 Режим ШИМ

В ШИМ режиме модуля CCP вывод CCPx используется в качестве выхода 10-разрядного ШИМ. Т.к. вывод CCPx мультиплицирован с цифровым каналом порта ввода/вывода, соответствующий бит направления TRIS должен быть сброшен в ‘0’.

Примечание. Очистка регистра CCPxCON вынудит перевести вывод CCPx в низкий логический уровень. Низкий логический уровень не является данными из защелки PORT.

На рисунке 14-3 показана структурная схема модуля CCPx в ШИМ режиме. Пошаговое описание настройки модуля CCPx в ШИМ режиме смотрите в разделе 14.5.3.

Рис. 14-3 Структурная схема модуля CCPx в ШИМ режиме



Примечание 1. 8-разрядный таймер связан с 2-разрядным внутренним счетчиком Q или 2 битами предделителя, чтобы создать 10 - разрядный счетчик.

На рисунке 14-4 показана временная диаграмма одного цикла ШИМ (период ШИМ и длительность высокого уровня сигнала). Частота ШИМ есть обратная величина периоду (1/период).

Рис. 14-4 Временная диаграмма одного цикла ШИМ

14.5.1 Период ШИМ

Период ШИМ определяется значением в регистре PR2 и может быть вычислен по формуле:

Период ШИМ = [(PR2) + 1] x 4 x Tosc x (коэффициент предделителя TMR2)

Частота ШИМ = 1 / Период ШИМ

Когда значение TMR2 сравнивается с PR2, выполняются следующие действия:

- TMR2 сбрасывается в 00h;
- Устанавливается высокий уровень сигнал на выводе CCPx (если скважность равна 0%, то сигнал в высокий уровень устанавливаться не будет);
- Модуль ШИМ начинает новый цикл, загружая значение из регистра CCPRxL в CCPRxH.

Примечание. Выходной делитель TMR2 не влияет на частоту ШИМ. Он может использоваться для отсчета времени, когда необходимо изменить скважность ШИМ.

14.5.2 Длительность импульса ШИМ

Длительность импульса ШИМ определяется битами в регистрах CCPRxL и CCPxCON<5:4>. Для 10-разрядного ШИМ старшие восемь бит сохраняются в регистре CCPRxL, а младшие два бита - в регистре CCPxCON<5:4> (CCPRxL:CCPxCON<5:4>). 10- разрядное значение представляется битами DCxB9:DCxB0.

Для вычисления длительности сигнала высокого уровня, воспользуйтесь следующей формулой:

Длительность импульса ШИМ = (DCxB9:DCxB0) x Tosc x (коэффициент предделителя TMR2)

Биты в регистре CCPRxL и CCPxCON<5:4> могут быть изменены в любое время, но значение в регистре CCPRxH не изменяется, пока не произойдет соответствие PR2 и TMR2. В ШИМ режиме регистр CCPRxH доступен только для чтения.

Регистр CCPRxH и внутренняя двух разрядная защелка образуют буфер ШИМ. Эффект буферизации необходим при записи нового значения длительности импульса ШИМ.

Когда значение CCPRxH и 2-разрядной внутренней защелки соответствует значению TMR2 и внутреннему 2-разрядному счетчику, в такте Q2 на выводе CCPx будет установлен низкий уровень сигнала (конец импульса ШИМ).

Максимальную разрядность ШИМ для данной частоты можно вычислить по формуле (бит):

$$= \frac{\log(\frac{Fosc}{Fpwm})}{\log(2)}$$

Примечание. Если длительность импульса ШИМ больше периода ШИМ, вывод CCP1 не будет иметь низкий уровень сигнала, что позволяет реализовать скважность выходного сигнала 100%.

14.5.2.2 Минимальная разрешающая способность

Минимальная разрешающая способность каждого бита импульса ШИМ зависит от режима работы предделителя TMR2.

Таблица 4-4 Минимальная длительность бита импульса ШИМ

Коэффициент предделителя	T2CKPS1:T2CKPS0	Минимальное разрешение
1	00	T_{OSC}
4	01	T_{CY}
16	1x	$4T_{CY}$

Пример 14-2 Расчет периода ШИМ и длительности импульса

Частота дискретизации ШИМ = 78.125кГц
F_{OSC} = 20МГц
Предделитель TMR2 = 1

1/78.125кГц = [(PR2) +] • 4 • 1/20МГц • 1
12.8мкс = [(PR2) + 1] • 4 • 50нс • 1
PR2 = 63

Найдем максимальную разрядность длительности импульса ШИМ при частоте дискретизации 78.125кГц и тактовой частоте микроконтроллера 20МГц.

1/78.125кГц = $2^{\text{РАЗРЯДНОСТЬ ШИМ}}$ • 1/20МГц • 1
12.8мкс = $2^{\text{РАЗРЯДНОСТЬ ШИМ}}$ • 50нс • 1
256 = $2^{\text{РАЗРЯДНОСТЬ ШИМ}}$
log(256) = (Разрядность ШИМ) • log(2)
8.0 = Разрядность ШИМ

При тактовой частоте микроконтроллера 20МГц и частоте дискретизации ШИМ 78.125кГц самая большая разрядность ШИМ 8 бит (т.е. 0 ≤ DCxB9:DCxB0 ≤ 255). Любое значение больше 255 приведет к значению скважность цикла ШИМ 100%. Чтобы получить большую разрядность ШИМ необходимо уменьшить частоту дискретизации ШИМ. Для повышение частоты дискретизации ШИМ необходимо уменьшить разрядность ШИМ.

В таблице 14-5 представлено соответствие частоты ШИМ и разрядности ШИМ при F_{OSC} = 20МГц (также показаны значение PR2 и коэффициента предделителя TMR2).

Таблица 14-5 Соответствие частоты ШИМ и разрядности ШИМ при тактовой частоте микроконтроллера 20МГц

Частота ШИМ	1.22кГц	4.88кГц	19.53кГц	78.12кГц	156.3кГц	208.3кГц
Коэффициент предделителя TMR2	16	4	1	1	1	1
Значение PR2	0xFF	0xFF	0xFF	0x3F	0x1F	0x17
Разрешение ШИМ (бит)	10	10	10	8	7	5.5

14.5.3 Последовательность настройки модуля CCP в ШИМ режиме

Рекомендованная последовательность включения модуля CCP в ШИМ режиме:

1. Установить период ШИМ в регистре PR2;
2. Установить длительность импульса в битах DCxB9:DCxB0;
3. Настроить вывод CCPx как выход, сбросив соответствующий бит TRIS;
4. Настроить предделитель и включить TMR2 в регистре T2CON;
5. Включить CCP в режиме ШИМ.

14.5.4 Работа в SLEEP режим микроконтроллера

В SLEEP режиме микроконтроллера TMR2 не будет инкрементироваться, и состояние модуля CCPx не изменяется. Если на выходе CCP присутствует определенный уровень сигнала, он будет удерживаться пока микроконтроллер находится в SLEEP режиме. По выходу из SLEEP режима модуль CCPx продолжит работать с текущего состояния.

14.5.5 Эффект сброса

После любого вида сброса модуль CCP выключен.

14.6 Инициализация

Модуль CCP может работать в одном из трех режимов. В примере 14-3 показана инициализация модуля CCP в режиме захвата, в примере 14-4 - инициализация в режиме сравнения, а в примере14-5 представлена инициализация в ШИМ режиме.

Пример 14-3 Инициализация модуля CCP в режиме захвата

```
        CLRF    CCP1CON             ; Выключить модуль CCP
        CLRF    TMR1H               ; Очистить старший байт TMR1
        CLRF    TMR1L               ; Очистить младший байт TMR1
        CLRF    INTCON              ; Выключить прерывания
        BSF     STATUS, RP0         ; Банк 1
        BSF     TRISC, CCP1         ; Настроить вывод CCP на вход
        CLRF    PIE1                ; Выключить периферийные прерывания
        BCF     STATUS, RP0         ; Банк 0
        CLRF    PIR1                ; Сбросить все флаги периферийных прерываний
        MOVLW   0x06                ; Режим захвата, захват данных выполнять
        MOVWF   CCP1CON             ; по каждому 4-му переднему фронту сигнала
        BSF     T1CON, TMR1ON       ; Разрешить приращение TMR1
;
; Прерывание от модуля CCP1 выключены,
; проверка флага прерываний от модуля CCP1
;
Capture_Event
        BTFSS   PIR1, CCP1IF
        GOTO    Capture_Event
;
; Произошел захват данных
;
        BCF     PIR1, CCP1IF        ; Необходимо сбросить флаг до следующего захвата
```

Пример 14-4 Инициализация модуля CCP в режиме сравнения

```
        CLRF    CCP1CON             ; Выключить модуль CCP
        CLRF    TMR1H               ; Очистить старший байт TMR1
        CLRF    TMR1L               ; Очистить младший байт TMR1
        CLRF    INTCON              ; Выключить прерывания
        BSF     STATUS, RP0         ; Банк 1
        BCF     TRISC, CCP1         ; Настроить вывод CCP на выход
        CLRF    PIE1                ; Выключить периферийные прерывания
        BCF     STATUS, RP0         ; Банк 0
        CLRF    PIR1                ; Сбросить все флаги периферийных прерываний
        MOVLW   0x08                ; Режим сравнения, установить высокий логический
        MOVWF   CCP1CON             ; уровень на выходе CCP по соответствию данных
        BSF     T1CON, TMR1ON       ; Разрешить приращение TMR1
;
; Прерывание от модуля CCP1 выключены,
; проверка флага прерываний от модуля CCP1
;
Compare_Event
        BTFSS   PIR1, CCP1IF
        GOTO    Compare_Event
;
; Произошло сравнение данных
;
        BCF     PIR1, CCP1IF        ; Необходимо сбросить флаг до следующего сравнения
```

Пример 14-5 Инициализация модуля CCP в ШИМ режиме

```
        CLRF    CCP1CON             ; Выключить модуль CCP
        CLRF    TMR2                ; Очистить TMR2
        BSF     STATUS, RP0         ; Банк 1
        MOVLW   0x7F                ;
        MOVWF   PR2                 ;
        BCF     STATUS, RP0         ; Банк 0
        MOVLW   0x1F                ;
        MOVWF   CCPR1L              ; Длительность импульса 25% от цикла ШИМ
        CLRF    INTCON              ; Выключить прерывания
        BSF     STATUS, RP0         ; Банк 1
        BCF     TRISC, CCP1         ; Настроить вывод CCP на выход
        CLRF    PIE1                ; Выключить периферийные прерывания
        BCF     STATUS, RP0         ; Банк 0
        CLRF    PIR1                ; Сбросить все флаги периферийных прерываний
        MOVLW   0x2C                ; Режим ШИМ, два младших бита длительности
        MOVWF   CCP1CON             ; импульса ШИМ = 10
        BSF     T2CON, TMR2ON       ; Разрешить приращение TMR2
;
; Прерывание от модуля TMR2 выключены,
; проверка флага переполнения TMR2
;
PWM_Period_Match
        BTFSS   PIR1, TMR2IF
        GOTO    PWM_Period_Match
;
; Здесь можно изменить период и длительность импульса ШИМ
;
        BCF     PIR1, TMR2IF
```

14.7 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Какой таймер я могу использовать для режима захвата и сравнения модуля CCP?
Ответ 1:
В режимах захвата и сравнения модуль CCP использует только TMR1, никакой другой таймер для этих режимов модуля CCP использоваться не может. Это означает, что если больше чем один модуль CCP работают в режиме захвата или сравнения, то они используют один и тот же таймер.

Вопрос 2: Какой таймер я могу использовать в ШИМ режиме модуля CCP?
Ответ 2:
В ШИМ режиме модуля CCP опорным является таймер TMR2, никакой другой таймер не может использоваться для этого режима (это единственный таймер, имеющий регистр периода). Если несколько модулей CCP работают в ШИМ режиме, то они используют один опорный таймер TMR2. Это означает, что оба ШИМ будут иметь одинаковую частоту дискретизации.

Вопрос 3: Можно использовать один модуль CCP для работы в режиме захвата (сравнения) и в ШИМ режиме в одно и тоже время? В этих режимах модуль CCP использует разные таймеры, как его правильно настроить?
Ответ 3:
Таймеры могут быть разные, но другие логические функции объединены. Вы можете переключать режим работы модуля CCP. Если микроконтроллер содержит два модуля CCP, то вы можете настроить модуль CCP1 в режиме ШИМ, а модуль CCP2 настроить для работы в режиме сравнения или захвата данных (или наоборот), т.к. эти модули независимы.

Вопрос 4: Как влияет сброс микроконтроллера на работу модуля CCP?
Ответ 4:
Любой сброс микроконтроллера выключает модуль CCP. Дополнительную информацию смотрите в разделе "Сброс".

Вопрос 5: Я установил в регистре CCP1CON режим сравнения с триггером специального события (1011), который сбрасывает TMR1. Когда произойдет сравнение данных, я получу прерывания от модуля CCP1 и TMR1 (флаги TMR1IF и CCP1IF установлены в '1')?
Ответ 5:
Флаг CCP1IF устанавливается в '1', когда возникает условие соответствия. Флаг TMR1IF устанавливается в '1', когда происходит переполнение TMR1. Сигнал от триггера специального события не рассматривается как переполнение TMR1. Однако, если значение регистров CCPR1L и CCPR1H равно FFh, то переполнение TMR1 происходит одновременно с соответствием данных, поэтому флаги CCP1IF и TMR1IF устанавливаются в '1' одновременно.

Вопрос 6: Как мне использовать TMR2 в качестве универсального таймера с генерацией прерываний при переполнении?
Ответ 6:
TMR2 всегда сбрасывается в 0, когда его значение сравнивается со значением в регистре PR2, при этом устанавливается в '1' флаг прерывания TMR2IF. Записав в регистр PR2 значение FFh, прерывание от TMR2 будет возникать при переполнении FFh (как переполнение TMR0). Достаточно часто необходимо изменить период переполнения таймера и возникновения прерываний. Обычно в таймер записывается начальное значение, чтобы ускорить переполнение таймера. Это означает, что при каждом переполнении таймера необходимо записывать начальное значение, чтобы получить требуемый период возникновения прерываний от таймера. Выгода использования TMR2 заключается в том, что требуемый период переполнения TMR2 может быть сохранен в регистре PR2. При каждом переполнении TMR2 вам не нужно записывать в него начального значение, т.к. таймер будет считать до значения в регистре PR2.

Вопрос 7: Я использую модуль CCP в ШИМ режиме. Длительность импульса ШИМ практически всегда равняется 100% даже, когда в регистр длительности импульса ШИМ записано значение 7Fh, хотя скважность должна равняться 50%. Что я делаю неправильно?
Ответ 7:
1. Значение в регистре CCPRxL больше чем PR2. Это возникает, когда необходимо получить большую частоту дискретизации ШИМ записью маленького значения в регистр PR2. В данном случае, если в регистр PR2 записано 7Eh, то CCPRxL = 7Fh будет вызывать скважность ШИМ 100%.
2. Если соответствующий бит TRIS настраивает вывод CCP как вход, то модуль CCP в ШИМ режиме не может управлять состоянием вывода. В этом случае на выводе присутствует "плавающий" уровень сигнала, скважность ШИМ может казаться 0% или 100% (или иметь другой постоянный уровень сигнала).

Вопрос 8: Я хочу определить частоту входного сигнала используя режим захвата модуля CCP. Мой алгоритм работает следующим образом: по первому активному фронту сигнала сбрасываю TMR1, по второму фронту сигнала в регистры захвата данных переписывается значение TMR1, которое является периодом сигнала. Проблема состоит в том, что очистка таймера происходит после исполнения 12 команд микроконтроллера, получив первый фронт сигнала (переход на обработку прерываний и сохранение контекста), поэтому я не могу измерять большие частоты сигнала. Как решить эту проблему?

Ответ 8:

Нет необходимости сбрасывать TMR1 в '0' по первому фронту сигнала. Первое значение захвата сохраните в дополнительных регистрах. При возникновении второго захвата получите разность между первым и вторым значением. Предположив, что переполнение TMR1 не происходит между первым и вторым фронтом сигнала, Вы всегда будете получать правильное значение периода. Это показано в следующем примере:

1. Сначала зафиксировано значение FFFEh, сохраните его в двух дополнительных регистрах.
2. Второе значение захвата данных 0001h (счетчик увеличился на 3).
3. 0001h - FFFEh = 0003h. Это такое же значение, если бы Вы очистили TMR1 и ожидали второй фронт сигнала (из-за необходимости очистки TMR1 значение может отличаться). Длительность перехода на прерывание теперь имеет минимальное влияние, т.к. значение сохраняется автоматически. Для более точного измерения малого периода не разрешайте прерывания, а проверяйте состояние флага в цикле. Если возможно переполнение TMR1 при измерении периода, рассмотрите возможность использования методики авто-масштабирования, в которой измерения начинаются с большим коэффициентом предделителя TMR1, а затем переключайте к меньшему коэффициенту, чтобы повысить точность измерений.

14.8 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с модулем CCP в микроконтроллерах PICmicro MCU:

Документ	Номер
Using the CCP Module Применение модуля CCP	AN594
Implementing Ultrasonic Ranging Ультразвуковое измерение расстояния	AN597
Air Flow Control using Fuzzy Logic Управление потоками воздуха на основе Fuzzy Logic	AN600
Adaptive Differential Pulse Code Modulation using PICmicros Реализация ADPCM на микроконтроллерах PICmicro	AN643

Раздел 15. Модуль SSP

Содержание

Примечание. Обратитесь к приложению C.2 или технической документации на микроконтроллеры, чтобы определить в каких микроконтроллерах реализован модуль SSP.

15.1 Введение

Модуль синхронного последовательного порта (SSP) может использоваться для связи с периферийными микросхемами или другими микроконтроллерами. Периферийными микросхемами могут быть: EEPROM память, сдвиговые регистры, драйверы ЖКИ, АЦП и др. Модуль SSP может работать в одном из двух режимах:

- Последовательный периферийный интерфейс (SPI);
- Inter-Integrated Circuit (I^2C):
  - ведомой режим;
  - контроль состояния портов ввода/ввода для обнаружения битов START, STOP с целью упрощения программного обеспечения в режиме ведущего и при конкуренции на шине.

15.2 Управляющие регистры

SSPSTAT: Регистр статуса модуля SSP

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	**CKE**	**D/-A**	**P**	**S**	**R/-W**	**UA**	**BF**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **SMP:** Фаза выборки бита данных в режиме SPI
Ведущий режим SPI
1 = опрос входа в конце периода вывода данных
0 = опрос входа в середине периода вывода данных

Ведомый режим SPI
Для режима ведомого SPI этот бит всегда должен быть сброшен в '0'

бит 6: **CKE:** Выбор фронта тактового сигнала в режиме SPI (см. рис. 15-3, 15-4 и 15-5)
CKP=0 (SSPCON<4>)
1 = данные передаются по переднему фронту сигнала на выводе SCK
0 = данные передаются по заднему фронту сигнала на выводе SCK

CKP=1 (SSPCON<4>)
1 = данные передаются по заднему фронту сигнала на выводе SCK
0 = данные передаются по переднему фронту сигнала на выводе SCK

бит 5: **D/-A:** Бит Данные/Адрес (только для режима I^2C)
1 = последний принятый или переданный байт является информационным
0 = последний принятый или переданный байт является адресным

бит 4: **P:** Бит STOP (только для режима I^2C)
Этот бит сбрасывается в '0' когда модуль SSP выключен.
1 = указывает, что бит STOP был обнаружен последним (этот бит равен '0' после сброса)
0 = бит STOP не является последним

бит 3: **S:** Бит START (только для режима I^2C)
Этот бит сбрасывается в '0' когда модуль SSP выключен.
1 = указывает, что бит START был обнаружен последним (этот бит равен '0' после сброса)
0 = бит START не является последним

бит 2: **R/-W:** Бит чтения/записи (только для режима I^2C)
Значение бита действительно только после совпадения адреса и до приема бита START, STOP или -ACK.
1 = чтение
0 = запись

бит 1: **UA:** Флаг обновления адреса устройства (только для режима 10-разрядного I^2C)
1 = необходимо обновить адрес в регистре SSPADD
0 = обновление адреса не требуется

бит 0: **BF:** Бит статуса буфера
Прием (SPI и I^2C режимы)
1 = прием завершен, буфер SSPBUF полон
0 = прием не завершен, буфер SSPBUF пуст

Передача (только I^2C режима)
1 = выполняется передача данных, буфер SSPBUF полон
0 = передача данных завершена, буфер SSPBUF пуст

SSPCON: Регистр управления модуля SSP

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	**SSPOV**	**SSPEN**	**CKP**	**SSPM3**	**SSPM2**	**SSPM1**	**SSPM0**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **WCOL:** Бит конфликта записи (сбрасывается в '0' программно)
1 = была предпринята попытка записи в SSPBUF во время передачи предыдущего байта
0 = конфликта не было

бит 6: **SSPOV:** Бит переполнения приемника
SPI режим
1 = принят новый байт, а SSPBUF содержит предыдущие данные(байт в SSPSR будет потерян). В ведомом режиме пользователь должен прочитать содержимое регистра SSPBUF даже, если только передает данные. В ведущем режиме бит в '1' не устанавливается, т.к. каждая операция инициализируется записью в SSPBUF. (сбрасывается в '0' программно)
0 = нет переполнения

I^2C режим
1 = принят новый байт, а SSPBUF содержит предыдущие данные. Значение бита не действительно при передаче данных. (сбрасывается в '0' программно)
0 = нет переполнения

бит 5: **SSPEN:** Бит включения модуля SSP
Когда модуль включен, соответствующие порты ввода/вывода настраиваются на выход или вход
SPI режим
1 = модуль SSP включен, выводы SCK, SDO, SDI, -SS используются модулем SSP
0 = модуль SSP выключен, выводы работают как цифровые порты ввода/вывода

I^2C режим
1 = модуль SSP включен, выводы SDA, SCL используются модулем SSP
0 = модуль SSP выключен, выводы работают как цифровые порты ввода/вывода

бит 4: **CKP:** Бит выбора полярности тактового сигнала
SPI режим
1 = пассивный высокий уровень сигнала
0 = пассивный низкий уровень сигнала

I^2C режим
Управление тактовым сигналом SCK
1 = не управлять тактовым сигналом
0 = удерживать тактовый сигнал в низком логическом уровне (используется для подготовки данных)

биты 3-0: **SSPM3:SSPM0:** Режим работы модуля SSP
0000 = ведущий режим SPI, тактовый сигнал = $F_{OSC}/4$
0001 = ведущий режим SPI, тактовый сигнал = $F_{OSC}/16$
0010 = ведущий режим SPI, тактовый сигнал = $F_{OSC}/64$
0011 = ведущий режим SPI, тактовый сигнал = выход TMR2 / 2
0100 = ведомый режим SPI, тактовый сигнал с вывода SCK. Вывод -SS подключен к SSP
0101 = ведомый режим SPI, тактовый сигнал с вывода SCK. Вывод -SS не подключен к SSP
0110 = ведомый режим I^2C, 7-разрядная адресация
0111 = ведомый режим I^2C, 10-разрядная адресация
1000 = резерв
1001 = резерв
1010 = резерв
1011 = программная поддержка ведущего режима I^2C (ведомый режим выключен)
1100 = резерв
1101 = резерв
1110 = ведомый режим I^2C, 7-разрядная адресация с разрешением прерываний по приему бит START и STOP
1111 = ведомый режим I^2C, 10-разрядная адресация с разрешением прерываний по приему бит START и STOP

15.3 Режим SPI

В SPI режиме возможен одновременный синхронный прием и передача 8-разрядных данных. Модуль SSP поддерживает четыре режима SPI с типовым использованием трех выводов микроконтроллера. В режиме ведущего SPI легко реализовать интерфейс Microwire ™.

- Вход последовательных данных (SDI);
- Выход последовательных данных (SDO);
- Тактовый сигнал (SCK).

Дополнительно может быть задействован четвертый вывод для работы в режиме ведомого:

- Выбор ведомого (-SS).

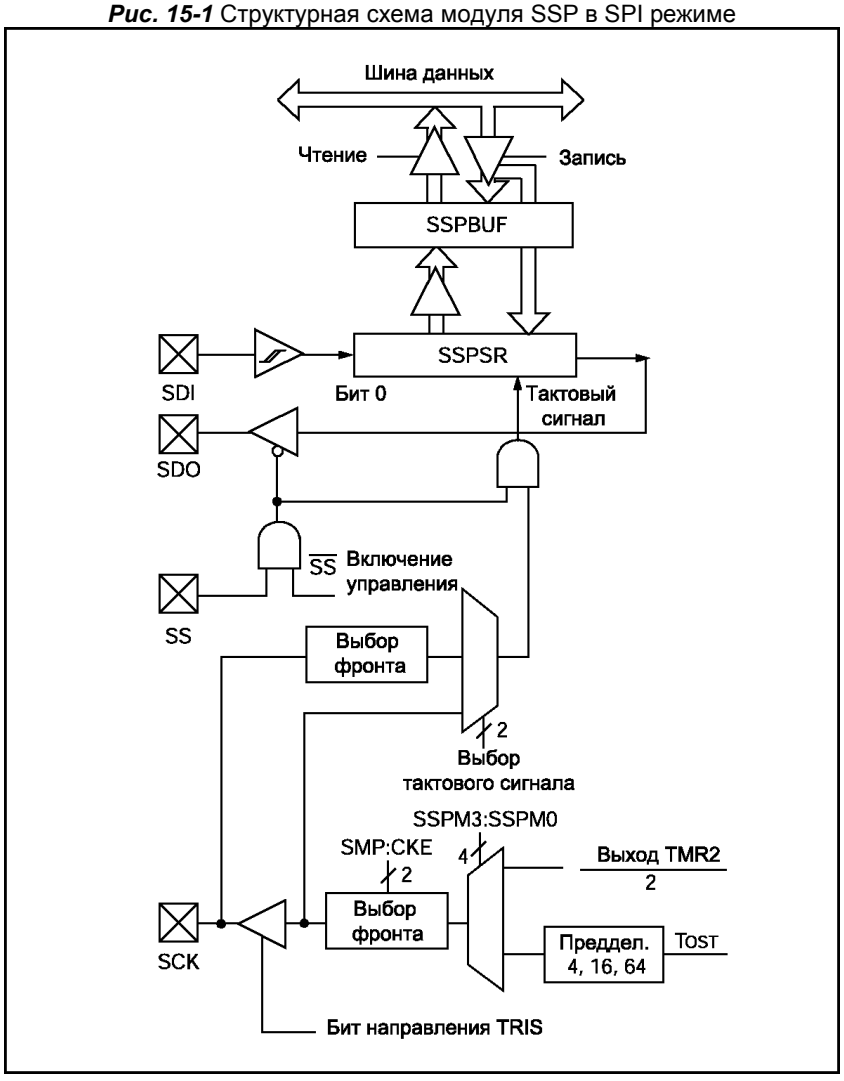
15.3.1 Работа модуля SSP в режиме SPI

При инициализации SPI необходимо определить параметры работы модуля SPI битами SSPCON<5:0>, SSPSTAT<7:6>. Управляющие биты определяют следующие параметры работы:

- Ведущий режим (SCK выход);
- Ведомый режим (SCK вход);
- Полярность тактового сигнала (пассивный уровень SCK);
- Фаза выборки входных данных;
- Активный фронт тактового сигнала (передний, задний);
- Частота тактового сигнала (только в ведущем режиме);
- Режим выбора ведомого (только в режиме ведомого).

На рисунке 15-1 показана структурная схема модуля SSP в SPI режиме.

Рис. 15-1 Структурная схема модуля SSP в SPI режиме



Модуль SSP состоит из приемного/передающего регистра сдвига (SSPSR) и буферного регистра (SSBUF). В регистре SSPSR выполняется сдвиг данных из/в микроконтроллер старшим битом вперед. В регистре SSPBUF сохраняются записанные данные, пока не будут получены новые. Приняв 8 бит данных в регистр SSPSR они переписываются в SSPBUF, устанавливается в '1' флаг полного приемного буфера BF (SSPSTAT<0>) и флаг прерывания SSPIF. Двойная буферизация принимаемых данных позволяет принимать следующий байт до чтения предыдущего. Любая запись в регистр SSPBUF во время выполнения операции приема/передачи данных будет игнорирована, при этом устанавливается в '1' флаг WCOL (SSPCON<7>). Пользователь должен программно сбросить бит WCOL в '0', чтобы была возможность проверки выполнения записи в регистр SSPBUF. При приеме данных в режиме SPI регистр SSPBUF должен быть прочитан до момента окончания приема следующего байта. Бит статуса приемного буфера BF (SSPSTAT<0>) указывает на получение нового байта данных. Бит BF аппаратно сбрасывается в '0' при чтении регистра SSPBUF. Принятые данные могут быть недостоверными, если режим SPI используется только для передачи данных. Прерывания от модуля SSP используются для определения завершения приема/передачи данных (в подпрограмме обработки прерываний необходимо прочитать/записать регистр SSPBUF). Если не планируется использовать прерывания от модуля SSP, то необходимо предусмотреть программную проверку выполнения записи в регистр SSPBUF для передачи данных. В примере 15-1 показана загрузка данных в регистр SSPBUF (SSPSR) для передачи данных. Затененная команда требуется только, если принимаемые данные имеют какое-то значение (в некоторых приложениях модуль SSP в режиме SPI используется только для передачи данных).

Пример 15-1 Загрузка данных в регистр SSPBUF(SSPSR)

```
        BCF    STATUS, RP1   ;Банк 1
        BSF    STATUS, RP0   ;
LOOP    BTFSS  SSPSTAT, BF   ;Данные приняты?
        GOTO   LOOP          ;Нет
        BCF    STATUS, RP0   ;Банк 0
        MOVF   SSPBUF, W     ;Загрузить в W значение из SSPBUF
        MOVWF  RXDATA        ;Если необходимо, сохранить значение в памяти
        MOVF   TXDATA, W     ;Загрузить в W значение из TXDATA
        MOVWF  SSPBUF        ;Передать новые данные
```

Регистр SSPSR не доступен для непосредственного чтения или записи, все операции выполняются через регистр SSPBUF. В регистре SSPSTAT находятся биты, указывающие текущее состояние модуля SSP.

15.3.2 Настройка выводов в режиме SPI

Для включения модуля SSP необходимо установить бит SSPEN (SSPCON<5>) в '1'. Для сброса или перенастройки режима SPI рекомендуется сбросить бит SSPEN в '0', выполнить изменения параметров работы, а затем вновь установить бит SSPEN в '1'. После включения SSP в режиме SPI выводы SDI, SDO, SCK, -SS используются последовательным портом. Для корректной работы последовательного порта биты регистров TRIS должны быть настроены следующем образом:

- SDI, бит TRIS должен быть установлен в '1';
- SDO, бит TRIS должен быть сброшен в '0';
- SCK (ведущий режим), бит TRIS должен быть сброшен в '0';
- SCK (ведомый режим), бит TRIS должен быть установлен в '1';
- -SS, бит TRIS должен быть установлен в '1'.

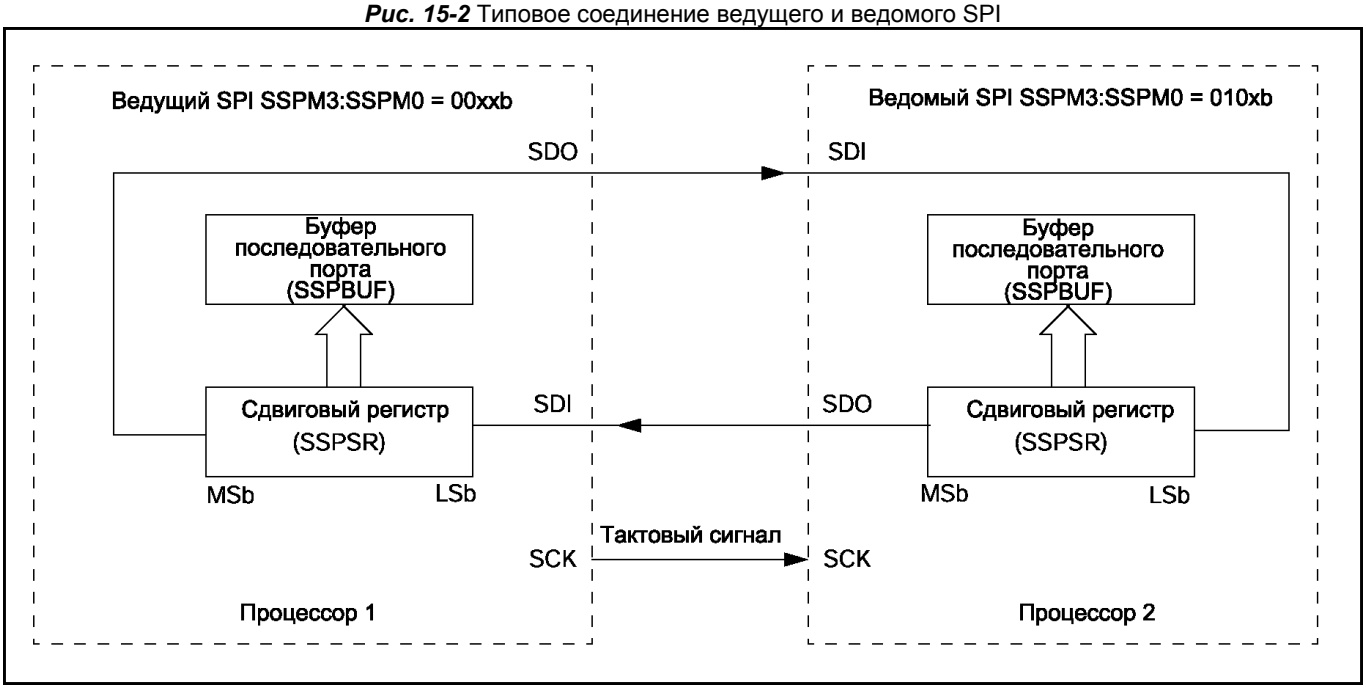
Любая нежелательная функция последовательного порта может быть выключена, настраивая соответствующие биты регистров направления данных TRIS. Например, если в режиме ведущего SPI выполняется только передача данных, то выводы SDI и -SS могут использоваться как цифровые выходы, сбросив соответствующие биты TRIS в '0'.

15.3.3 Типовое включение

На рисунке 15-2 показано типовое соединение двух микроконтроллеров. Главный микроконтроллер (процессор 1) инициализирует передачу, формируя тактовый сигнал SCK. Данные сдвигаются по установленному битом SMP фронту тактового сигнала. Для одновременного приема/передачи данных (фиктивных данных) оба микроконтроллера должны иметь одинаковую полярность тактового сигнала (бит CKP). Всего существует три сценария передачи данных:

- Ведущий передает данные - ведомый передает фиктивные данные;
- Ведущий передает данные - ведомый передает данные;
- Ведущий передает фиктивные данные - ведомый передает данные.

Рис. 15-2 Типовое соединение ведущего и ведомого SPI



15.3.4 Режим ведущего SPI

Ведущий шины может инициализировать передачу данных в любой момент, поскольку он генерирует тактовый сигнал, и определяет, когда ведомый (процессор 2) должен передать данные в соответствии с используемым протоколом.

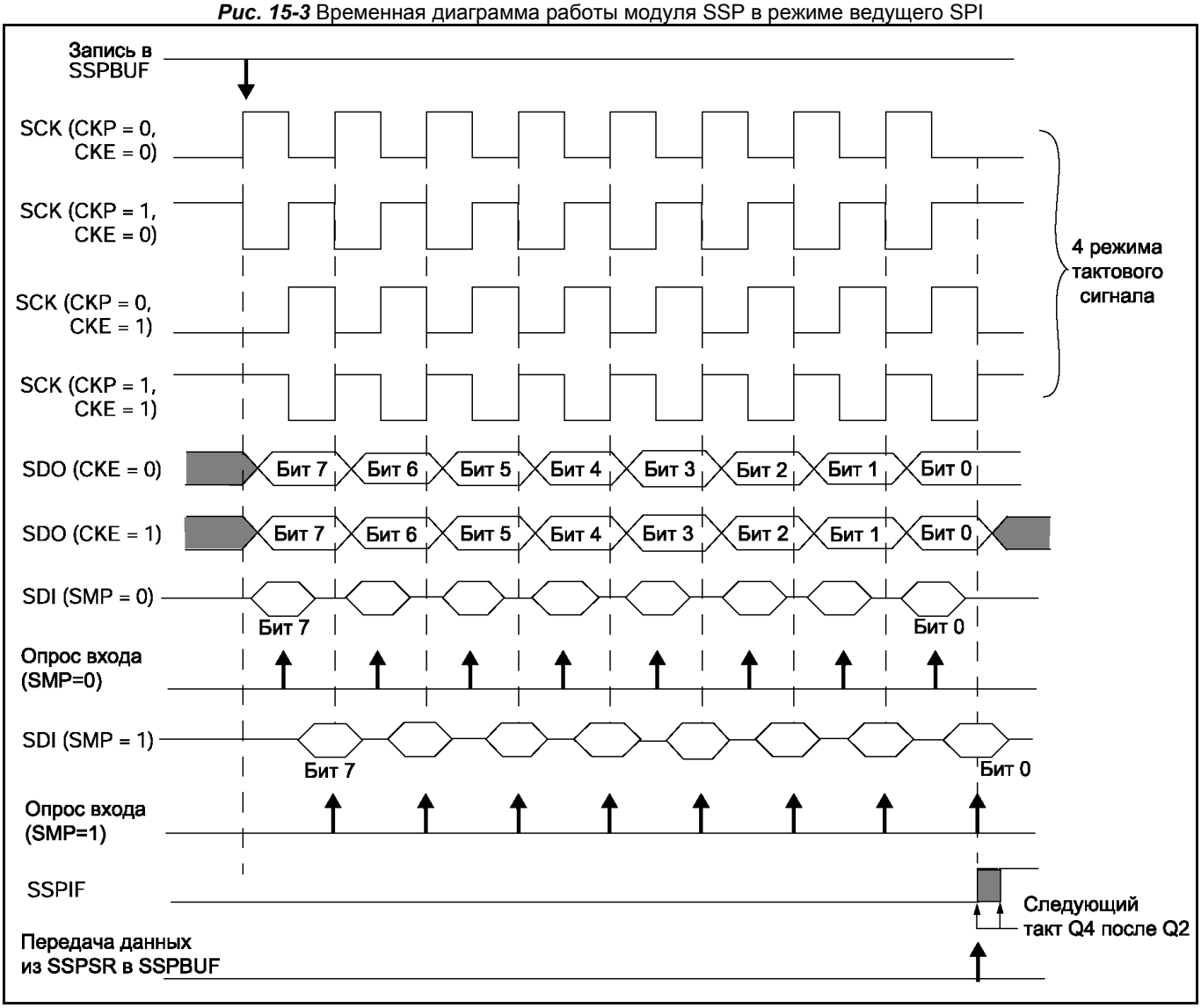
В режиме ведущего данные передаются/приняты после их записи/чтения из регистра SSPBUF. Если в SPI режиме требуется только принимать данные, вывод SDO может быть заблокирован (настроен как вход). Данные с вывода SDI последовательно сдвигаются в регистр SSPSR с установленной скоростью. Каждый принятый байт загружается в регистр SSPBUF (как нормально полученный байт) с формированием прерываний и воздействием на соответствующие биты статуса. Эта функция может быть полезна при реализации "монитора шины".

Полярность тактового сигнала устанавливается битом CKP (SSPCON<4>), что позволяет получить различные методы передачи данных (см. рисунки 15-3, 15-4 и 15-5). Данные всегда передаются старшим битом вперед. В ведущем режиме частота тактового сигнала выбирается программно:

- $F_{OSC}/4$ (или T_{CY});
- $F_{OSC}/16$ (или 4 x T_{CY});
- $F_{OSC}/64$ (или 16 x T_{CY});
- Выход таймера TMR2 / 2.

Максимальная частота передачи данных 5МГц при тактовой частоте микроконтроллера 20МГц.

Рис. 15-3 Временная диаграмма работы модуля SSP в режиме ведущего SPI



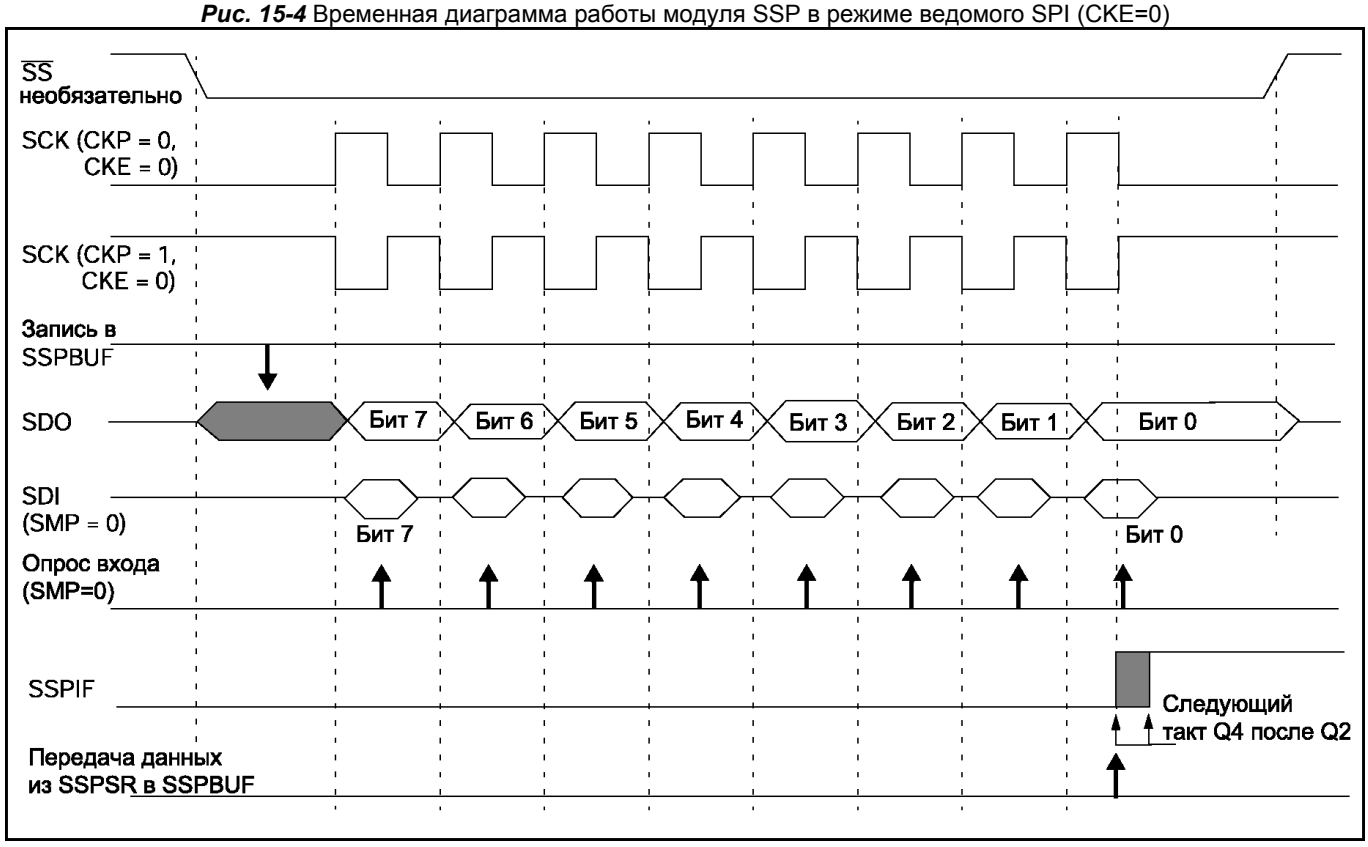
15.3.5 Режим ведомого SPI

В режиме ведомого данные передаются/принимаются по внешнему тактовому сигналу на выводе SCK. Когда принимается последний бит байта, устанавливается в '1' флаг прерываний SSPIF.

Полярность тактового сигнала выбирается битом CKP (SSPCON<4>). Временные диаграммы передачи данных по интерфейсу SPI смотрите на рисунке 15-3, 15-4 и 15-5 (данные передаются старшим битом вперед). Внешний тактовый сигнал должен удовлетворять требованиям длительности низкого и высокого логического уровня, описанным в разделе электрических характеристик.

В SLEEP режиме микроконтроллера ведомый может принимать/передавать данные. После приема данных микроконтроллер выходит из режима SLEEP, если разрешены прерывания от модуля SSP.

Рис. 15-4 Временная диаграмма работы модуля SSP в режиме ведомого SPI (CKE=0)



15

15.3.6 Выбор ведомого в режиме SPI

В режиме SPI вывод -SS позволяет подключать несколько ведомых к одному ведущему. Модуль SSP должен находиться в режиме ведомого SPI (SSPCON<3:0> = 0100), бит TRIS для вывода -SS установлен в '1', чтобы позволить ведущему выбирать ведомого. Когда на выводе -SS присутствует низкий логический уровень, передача и прием данных разрешены, а вывод SDO управляется модулем SSP. Если на выводе -SS высокий уровень сигнала, то вывод SDO переходит в 3-е состояние. В зависимости от приложения может потребоваться внешний подтягивающий резистор на выводе SDO.

В режиме ведомого SPI с поддержкой выбора ведомого по сигналу на выводе -SS (SSPCON<3:0>=0100) SPI модуль сброшен, если на выводе -SS напряжение питания V_{DD}. Если модуль SSP работает в режиме ведомого SPI и CKE = 1, необходимо разрешить управление с вывода -SS.

При сбросе модуля SSP в режиме SPI счетчик битов сдвигового регистра очищается. Сброс модуля в режиме SPI происходит при появлении высокого логического уровня на выводе -SS и сбросе в '0' бита SSPEN (см. рис. 15-6).

Для реализации двух проводного интерфейса вывод SDO может быть соединен с SDI. Когда SPI должен работать как приемник, вывод SDO настраивается на вход, что отключает передатчик от SDO. SDI всегда должен быть настроен как вход (функция SDI), т.к. это не создает конфликт шины.

Рис. 15-5 Временная диаграмма работы модуля SSP в режиме SPI с выбором ведомого (CKE=1)

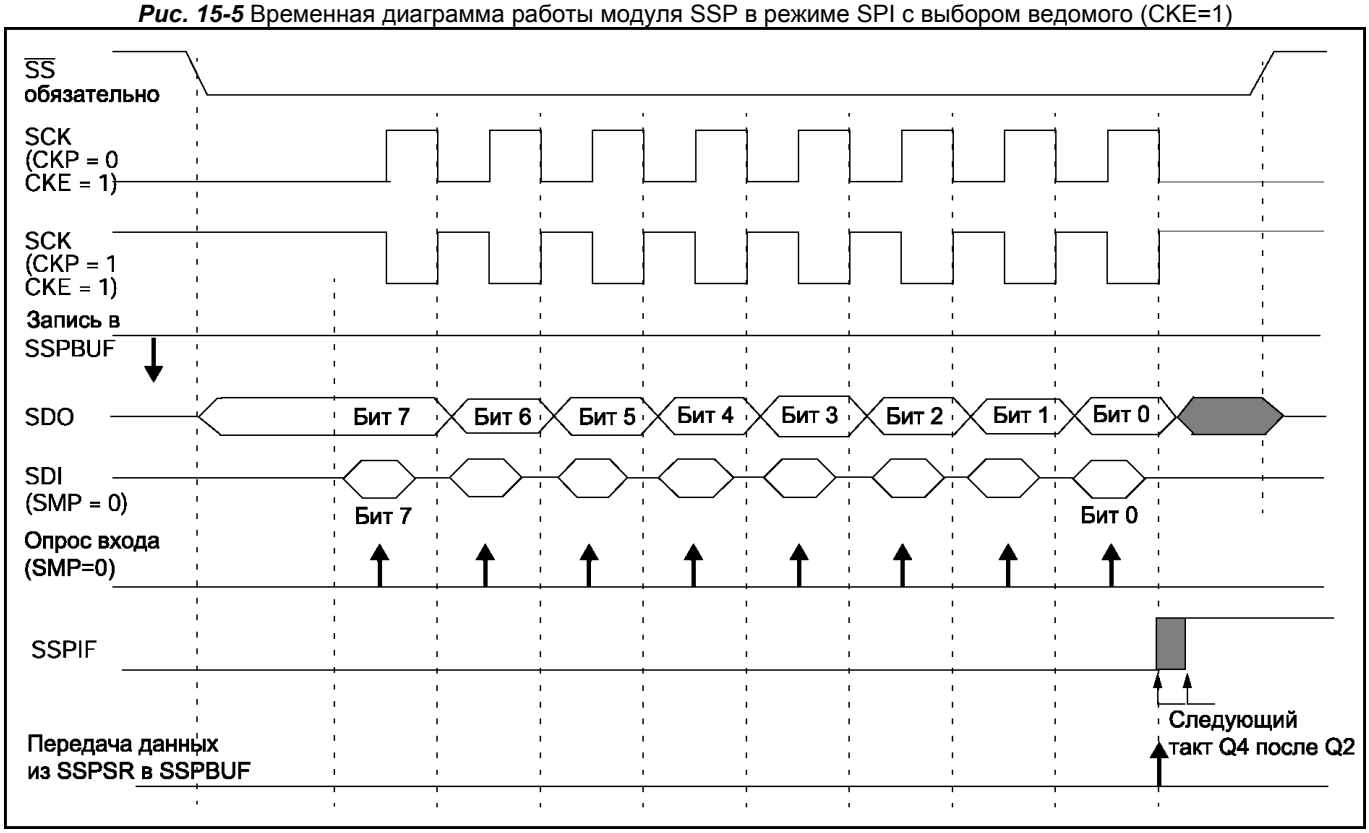
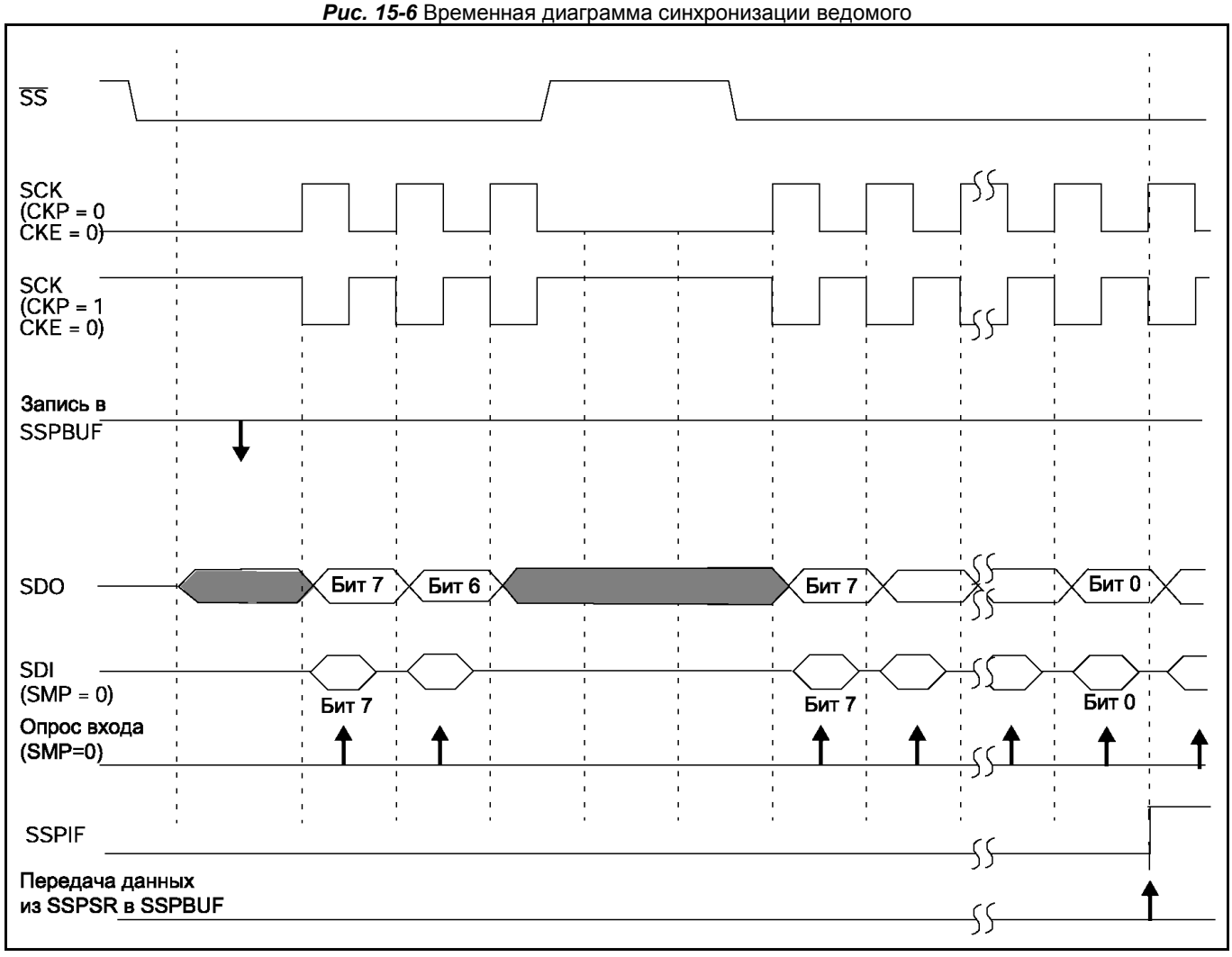


Рис. 15-6 Временная диаграмма синхронизации ведомого



15.3.7 Работа в SLEEP режиме микроконтроллера

В режиме ведущего SPI тактовый сигнал модуля SSP отсутствует, состояние приема/передачи данных не изменяется до выхода микроконтроллера из режима SLEEP. После выхода микроконтроллера из режима SLEEP модуль SSP продолжит передачу/прием данных.

В режиме ведомого SPI данные могут быть приняты/переданы, т.к. сдвиговый регистр работает асинхронно. Это позволяет в SLEEP режиме микроконтроллера принять/передать данные в/из сдвигового регистра. Как только будут приняты все 8 бит данных, устанавливается в '1' флаг прерывания от модуля SSP, и если прерывания разрешены, микроконтроллер выйдет из режима SLEEP.

15.3.8 Эффект сброса

Любой сброс микроконтроллера выключает модуль SSP, прием/передача данных прекращается.

Таблица 15-1 Регистры и биты, связанные с работой модуля SSP в режиме SPI

Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Сброс POR, BOR	Другие сбросы
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000u
PIR	SSPIF(1)								0	0
PIE	SSPIE(1)								0	0
SSPBUF	Буфер приемника SSP / регистр передатчика								xxxx xxxx	uuuu uuuu
SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
TRISA	-	-	Регистр направления данных PORTA						--11 1111	--11 1111
TRISC	Регистр направления данных PORTC								1111 1111	1111 1111
SSPSTAT	SMP	CKE	D/-A	P	S	R/-W	UA	BF	0000 0000	0000 0000

Обозначения: - = не используется, читается как 0; u = не изменяется; x = не известно; q = зависит от условий. Затененные биты на работу не влияют.

Примечания:

1. Расположение битов смотрите в технической документации на микроконтроллер.
2. В некоторых микроконтроллерах эти биты могут обозначаться как GPIE и GPIF.

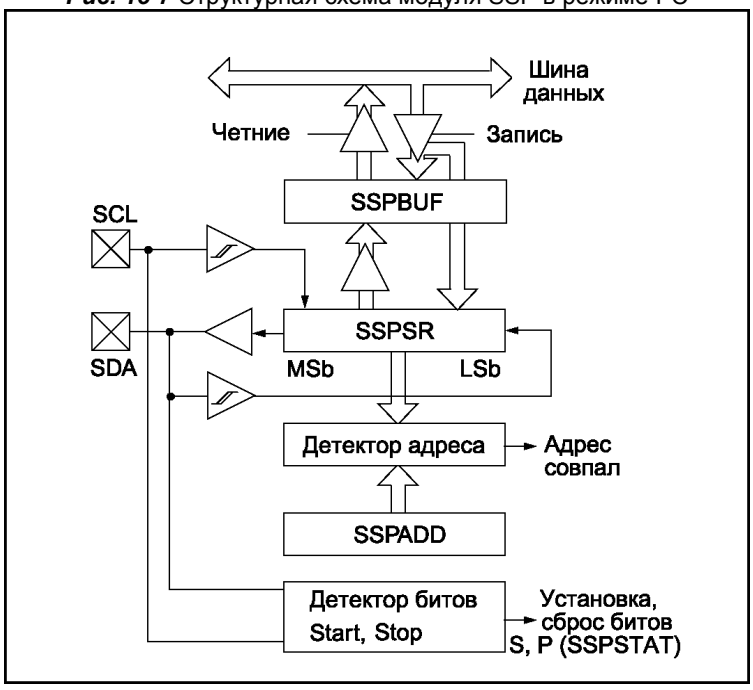
15.4 Режим I²C

Модуль SSP полностью поддерживает все функции ведомых устройств, включая поддержку общего вызова, аппаратные прерывания по детектированию битов START и STOP для определения занятости шины I²C при программной реализации режима ведущего. В SSP модуле реализована поддержка стандартного режима 7, 10-разрядной адресации. Дополнительно смотрите приложение A, в котором дано краткое описание шины I²C.

Для работы с шиной I²C используется два вывода SCL (сигнал синхронизации) и SDA (данные). Выводы SDA и SCL автоматически настраиваются при включении режима I²C. Включение модуля SSP выполняется установкой бита SSPEN (SSPCO<5>) в '1'.

Фильтр "glitch" подключен к выводам SDA и SCL, когда они настроены на вход. Фильтр работает в режимах 100кГц и 400кГц. В режиме 100кГц, когда выводы SDA и SCL настроены на выход, фильтр контролирует длительность формируемых сигналов независимо от тактовой частоты микроконтроллера.

Рис. 15-7 Структурная схема модуля SSP в режиме I²C



Для управления модулем SSP в режиме I²C используется пять регистров:

- SSPCON, регистр управления SSP;
- SSPSTAT, регистр статуса SSP;
- SSPBUF, буфер приемника/передатчика;
- SSPSR, сдвиговый регистр (пользователю не доступен);
- SSPADD, регистр адреса.

В регистре SSPCON устанавливается требуемый режим I²C. С помощью четырех битов (SSPCON<3:0>) можно выбрать один из режимов I²C:

- Ведомый режим I²C, 7-разрядная адресация;
- Ведомый режим I²C, 10-разрядная адресация;
- Программная поддержка ведущего режима I²C с конкуренцией на шине (разрешение прерываний по приему битов START и STOP);
- Программная поддержка ведущего режима I²C (ведомый режим выключен).

При выборе любого режима I²C выводы SCL и SDA должны быть настроены на вход, установкой соответствующих битов регистра TRISC в '1'. После выбора режима I²C и установки бита SSPEN в '1' выводы SDA (линия данных), SCL (линия синхронизации) подключаются к модулю SSP.

Регистр SSPSTAT содержит биты статуса передачи данных: обнаружение на шине битов START (S) или STOP (P), флаг приема байта данных или адреса, указатель загрузки старшего байта 10-разрядного адреса, бит операции приема/передачи.

В регистр SSPBUF загружаются данные для передачи по шине I²C, и из него читаются принятые данные. Регистр SSPSR выполняет сдвиг принимаемых/передаваемых данных. При приеме данных регистры SSPBUF, SSPSR работают как двухуровневый буфер приемника. Буфер позволяет принимать следующий байт до чтения предыдущего принятого байта из регистра SSPBUF. Когда байт полностью загружен в SSPSR, он передается в регистр SSPBUF и устанавливается флаг прерывания SSPIF в '1'. Если полностью принят следующий байт до чтения предыдущего байта из SSPBUF, то устанавливается бит SSPOV (SSPCON<6>) в '1', а байт в регистре SSPSR будет потерян.

В регистр SSPADD записывается адрес ведомого устройства. В 10-разрядном режиме пользователь должен сначала записывать старший байт адреса (1111 0 A9 A8 0). После соответствия старшего байта адреса необходимо загрузить младший байт адреса (A7:A0).

15.4.1 Режим ведомого I^2C

В режиме ведомого I^2C выводы SCL, SDA должны быть настроены на вход. Модуль SSP автоматически изменит направление вывода SDA при передаче данных ведомым.

При совпадении адреса или после приема байта данных (если предварительно совпал адрес) аппаратно генерируется бит подтверждения (-ACK), а затем данные из регистра SSPSR загружаются в SSPBUF.

Существует несколько условий, при которых бит -ACK не формируется (эти условия могут возникать одновременно):

a) Бит BF (SSPSTAT<0>) = 1 перед приемом данных;
b) Бит переполнения SSPOV (SSPSTAT<6>) = 1 перед приемом данных.

Если бит BF = 1, то значение из SSPSR не переписывается в регистр SSPBUF, а биты SSPIF и SSPOV устанавливаются в '1'. В таблице 15-2 показаны операции после приема байта при различных значениях битов BF, SSPOV. В затененных ячейках показана ситуация, когда вовремя не был сброшен бит переполнения SSPOV в '0'. Заметьте, что бит BF аппаратно сбрасывается в '0' при чтении из регистра SSPBUF, а бит SSPOV необходимо сбрасывать в '0' программно.

Минимальная длительность логических уровней входного сигнала синхронизации SCL должна удовлетворять требованиям раздела электрических характеристик (см. параметры 100 и 101).

15.4.1.1 Адресация

После включения модуля SSP ожидается формирование на шине бита START. Получив бит START, принимается 8 бит в сдвиговый регистр SSPSR. Выборка битов происходит по переднему фронту синхронизирующего сигнала на выводе SCL. По заднему фронту восьмого такта сигнала SCL значение в регистре SSPSR<7:1> сравнивается с содержимым регистра SSPADD. Если значение адреса совпадает, а биты BF и SSPOV равны нулю, то выполняются следующие действия:

a) Значение регистра SSPSR загружается SSPBUF по 8-му заднему фронту сигнала SCL;
b) Устанавливается флаг BF в '1' (буфер полон) по 8-му заднему фронту сигнала SCL;
c) Генерируется бит -ACK;
d) Устанавливается флаг прерываний SSPIF в '1' (если разрешено, генерируется прерывание) по 9-му заднему фронту сигнала SCL.

В режиме ведомого при 10-разрядной адресации необходимо принять два байта адреса. Пять старших бит первого байта определяют: является ли полученный байт первым байтом 10-разрядного адреса. Бит R/-W(SSPSTAT<2>) должен быть настроен для приема второго байта адреса. Для 10-разрядной адресации первый байт адреса должен иметь формат '1111 0 A9 A8 0', где A9:A8 два старших бита адреса. Рекомендуемая последовательность действий при 10-разрядной адресации (шаги 7-9 для передачи ведомым):

1. Принять старший байт адреса (устанавливаются биты SSPIF, BF и UA (SSPSTAT<1> в '1').
2. Записать младший байт адреса в регистр SSPADD (аппаратно сбрасывается бит UA в '0' и "отпускается" линия SCL).
3. Выполнить чтение из регистра SSPBUF (сбрасывается бит BF в '0') и сбросить флаг SSPIF в '0'.
4. Принять младший байт адреса (устанавливаются биты SSPIF, BF и UA (SSPSTAT<1> в '1').
5. Записать старший байт адреса в регистр SSPADD (аппаратно сбрасывается бит UA в '0' и "отпускается" линия SCL).
6. Выполнить чтение из регистра SSPBUF (сбрасывается бит BF в '0') и сбросить флаг SSPIF в '0'.
7. Принять бит повторный START.
8. Принять старший байт адреса (устанавливаются биты SSPIF и BF в '1').
9. Выполнить чтение из регистра SSPBUF (сбрасывается бит BF в '0') и сбросить флаг SSPIF в '0'.

Примечание. В 10-разрядном режиме после команды повторный START (шаг 7) не требуется обновлять значение в регистре SSPADD. В данном случае требуется соответствие только первого байта адреса.

Таблица 15-2 Операции после приема байта при различных значениях битов BF, SSPOV

Биты статуса приемника		Запись из SSPSR в SSPBUF	Формирование бита -ACK	Установка флага прерываний SSPIF
BF	**SSPOV**			
0	0	Есть	Есть	Есть
1	0	Нет	Нет	Есть
1	1	Нет	Нет	Есть
0	1	Есть	Нет	Есть

Примечание. В затененных ячейках показана ситуация, когда вовремя не был сброшен бит переполнения SSPOV в '0'.

15.4.1.2 Прием данных

Если бит R/-W в адресном байте равен нулю, а принятый адрес совпадает с адресом устройства, то бит R/-W в регистре SSPSTAT сбрасывается в '0'. Принятый адрес загружается в регистр SSPBUF.

Если бит BF (буфер полон) или SSPOV (переполнение буфера) установлен в '1', то бит подтверждения -ACK не формируется. Эту ошибку необходимо обработать программно. Если было выполнено чтение из регистра SSPBUF но не был сброшен бит SSPOV в '0', то бит -ACK не формируется.

Прерывание от модуля SSP генерируются при каждом принятом байте с шины I^2C, установкой флага SSPIF в '1' (сбрасывается программно). Регистр SSPSTAT используется для определения типа принятого байта.

Рис. 15-8 Временная диаграмма приема данных ведомым I^2C (7-разрядная адресация)

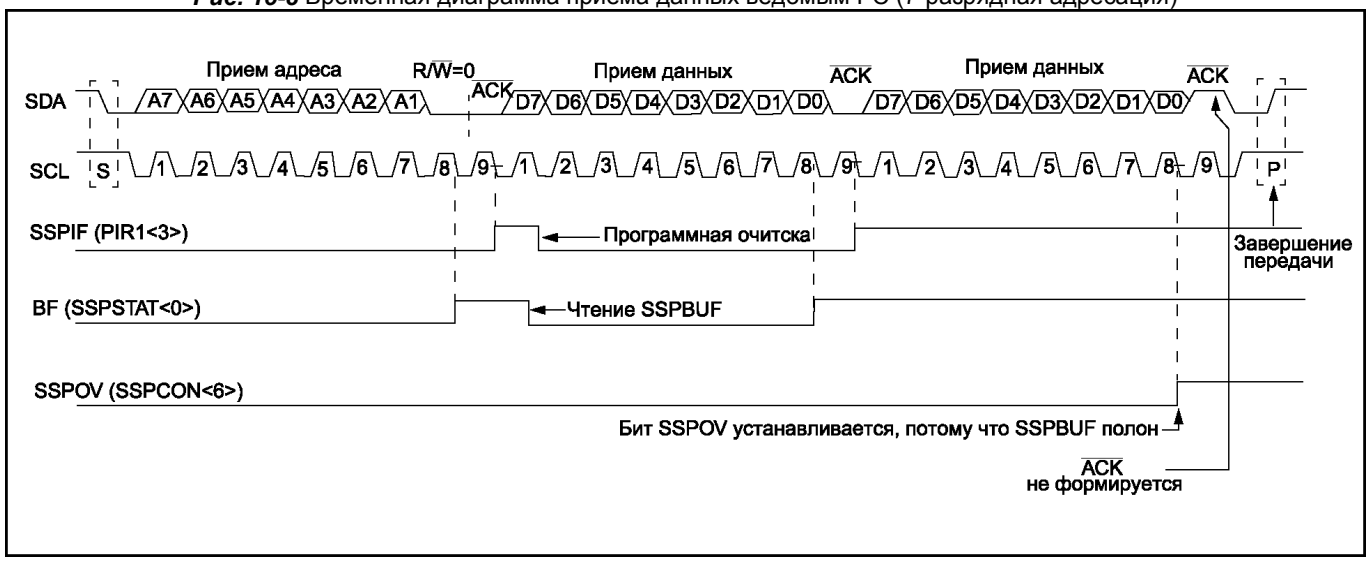
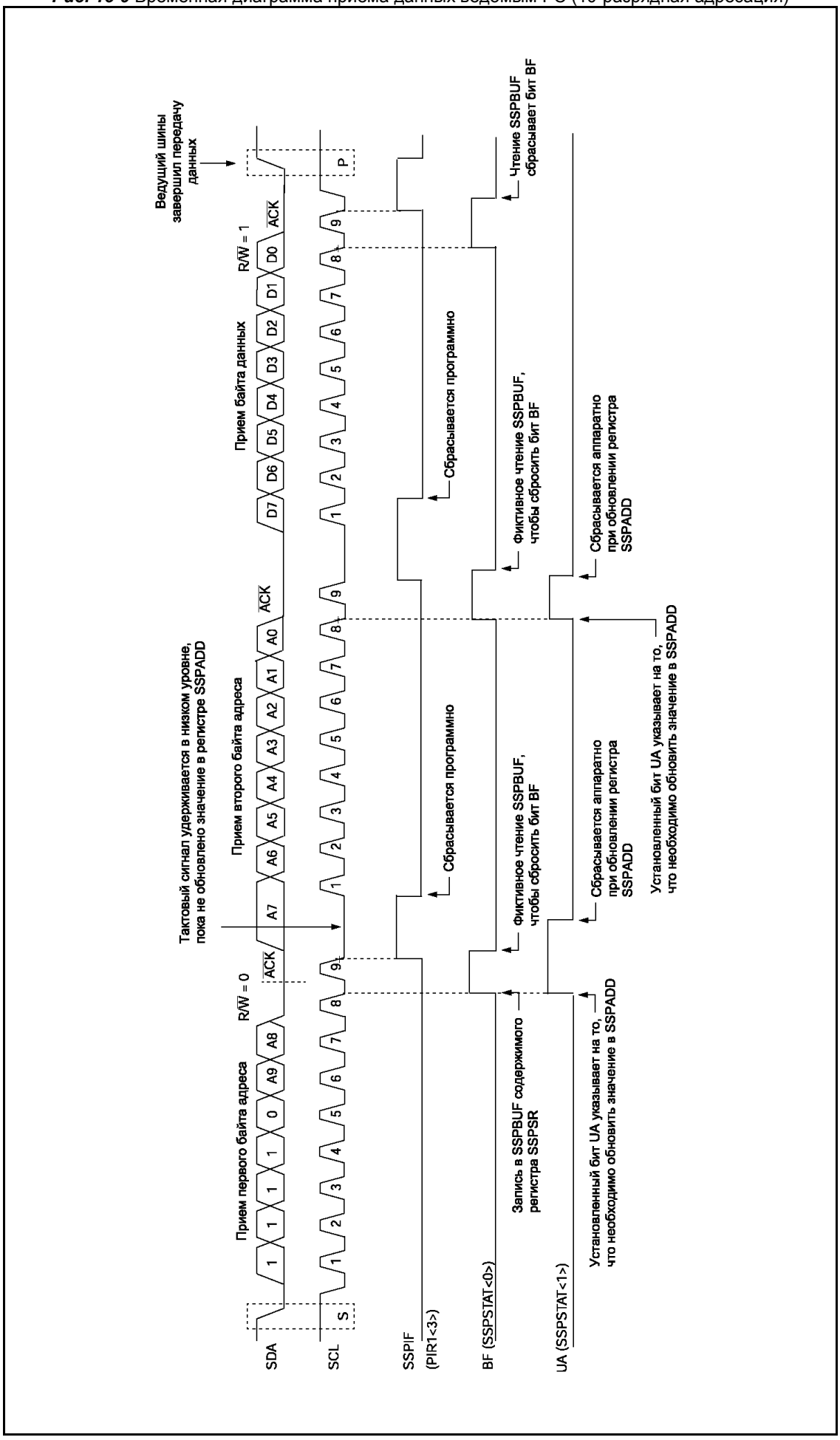


Рис. 15-9 Временная диаграмма приема данных ведомым I^2C (10-разрядная адресация)



15.4.1.3 Передача данных

Если бит R/-W в адресном байте равен '1', а принятый адрес совпадает с адресом устройства, то бит R/-W в регистре SSPSTAT устанавливается в '1'. Принятый адрес загружается в регистр SSPBUF. Бит -ACK формируется девятым битом, после чего линия SCL удерживается в низком логическом уровне. Передаваемые данные должны быть записаны в регистр SSPBUF, после чего они автоматически переписываются в регистр SSPSR. После записи данных необходимо "отпустить" сигнал SCL установкой бита CKP(SSPCON<4>) в '1'. Ведущий шины контролирует состояние линии SCL, ожидая смены уровня сигнала. Восемь бит загруженных данных последовательно сдвигаются по заднему фронту сигнала SCL, что гарантирует достоверное значение данных на линии SDA (см. рисунок 15-10).

Модуль SSP генерирует прерывание по каждому переданному байту, устанавливая бит SPPIF в '1' по заднему фронту девятого такта сигнала SCL. Флаг SSPIF должен быть сброшен программно. Регистр SSPSTAT используется для определения статуса передачи данных.

Ведущее устройство формирует бит подтверждения -ACK на девятом такте сигнала SCL для каждого принятого байта. Если бит подтверждения -ACK не сформирован (высокий уровень сигнала SDA), передача данных завершена. Логика ведомого устройства настраивается на обнаружение бита START. Если бит подтверждения -ACK был получен (низкий уровень сигнала SDA), в регистр SSPBUF необходимо записать новый байт для передачи. Линию SCL также необходимо "отпустить", установкой бита CKP в '1'.

Рис. 15-10 Временная диаграмма передачи данных ведомым I^2C (7-разрядная адресация)

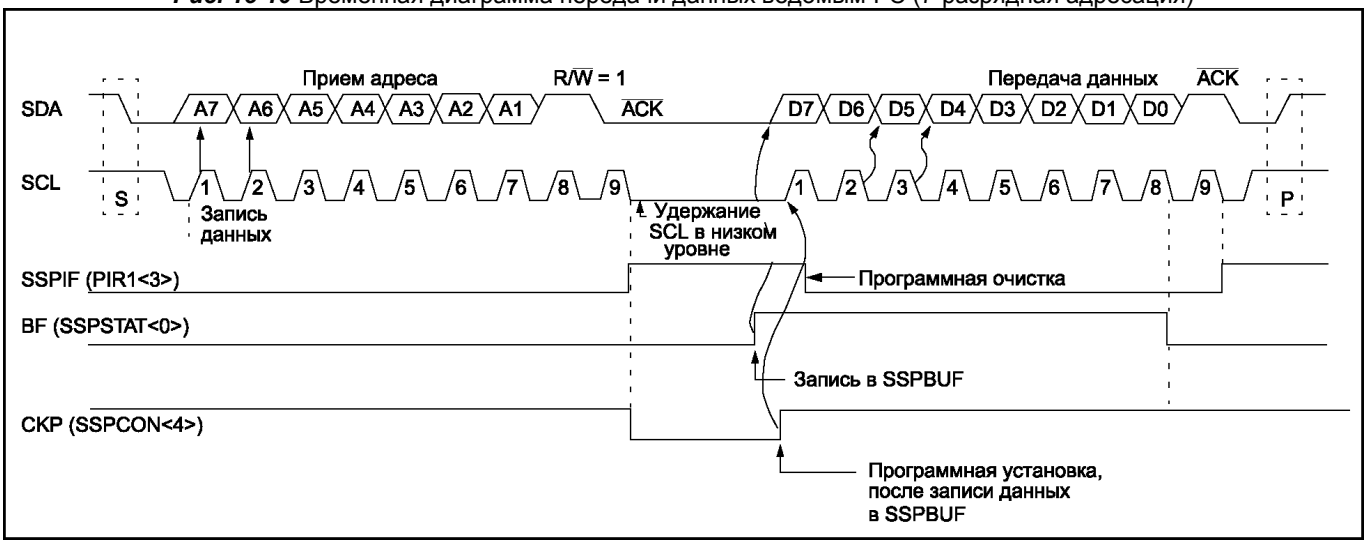
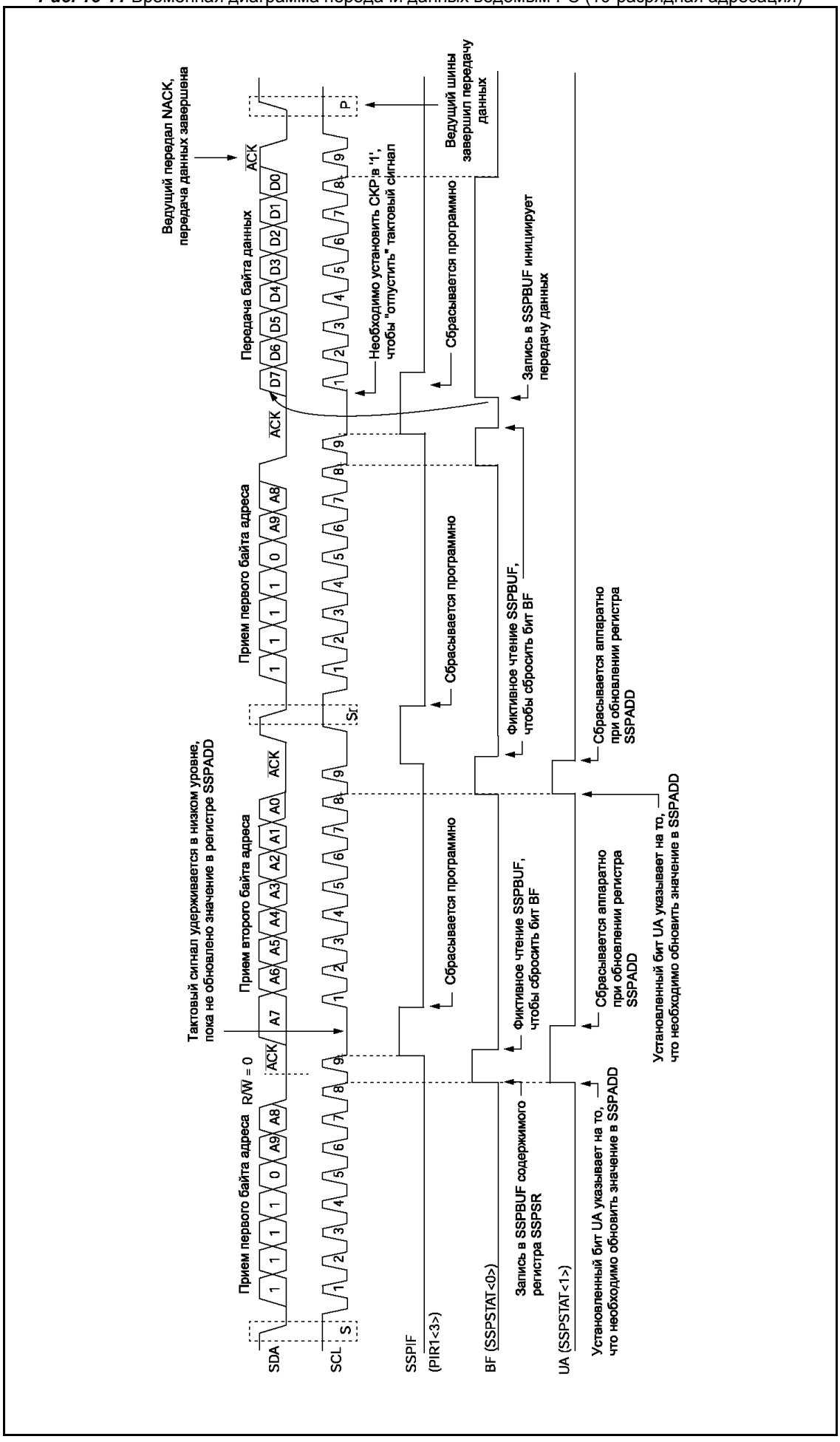


Рис. 15-11 Временная диаграмма передачи данных ведомым I^2C (10-разрядная адресация)



15.4.1.4 Арбитраж тактового сигнала

Арбитраж выполняется на линии SCL, чтобы запретить ведомому формировать следующий тактовый импульс. В режиме ведомого I^2C линия SCL будет удерживаться в низком логическом уровне, пока ЦПУ не ответит на прерывание (SSIF=1, CKP=0). Данные, которые нужно передать ведомому, записываются в регистр SSPBUF, затем устанавливается в '1' бит CKP, позволяя ведущему формировать тактовый сигнал.

15.4.2 Режим ведущего I^2C (программная реализация)

В режиме ведущего поддерживается генерация прерываний при обнаружении на шине битов START и STOP. Биты STOP (P) и START (S) в регистре SSPSTAT равны '0' после сброса микроконтроллера или при выключенном модуле SSP. Шина находится в неактивном состоянии, если бит P=1 или оба бита S, P равны '0'.

В режиме ведущего управлением уровнем сигнала на линиях SCL и SDA выполняется сбросом соответствующих битов TRIS. На выходе всегда присутствует низкий логический уровень вне зависимости от состояния битов регистра PORT. Для передачи логической '1' соответствующий бит TRIS должен быть установлен в '1' (вывод настроить на вход), а для передачи '0' - сбросить бит TRIS в '0' (вывод настроить на выход). Аналогично выполняется управление сигналом SCL.

Следующие события на шине I^2C могут привести к установки флага прерываний SSPIF в '1':

- Выполнено условие START;
- Выполнено условие STOP;
- Передан/принят байт данных.

Режим ведущего может быть выбран с выключенным ведомым (SSPM3:SSPM0 = 1011) или включенным ведомым (SSPM3:SSPM0 = 1110 или 1111). Когда режим ведомого включен, программное обеспечение должно дифференцировать источник прерываний.

15.4.3 Режим ведущего I^2C с конкуренцией на шине (программная реализация)

В режиме ведущего с конкуренцией на шине поддерживается генерация прерываний при обнаружении на шине битов START и STOP. Биты STOP (P) и START (S) в регистре SSPSTAT равны '0' после сброса микроконтроллера или при выключенном модуле SSP. Шина находится в неактивном состоянии, если бит P=1 (SSPSTAT<4>) или оба бита S, P равны '0'. Если шина занята, можно разрешить прерывания от SSP для обнаружения бита STOP на шине.

При конкуренции линия SDA должна проверяться на соответствия уровня, при ожидаемом высоком уровне на выходе. Если ожидается высокий уровень сигнала, а на линии присутствует сигнал с низким логическим уровнем, то необходимо "отпустить" линии SCL, SDA (установить в '1' биты TRIS). Арбитраж на шине I^2C может быть потерян во время:

- Передачи адреса;
- Передачи данных.

Когда ведомый режим включен, ведомый I^2C продолжает принимать данные. Когда арбитраж шины потерян во время передачи адреса, то сеанс связи можно продолжить, если получен бит подтверждения -ACK. Если арбитраж шины потерян во время передачи данных, то устройство должно повторить обмен данными позже.

15.4.4 Работа в SLEEP режиме

Ведомый I^2C может принимать адресные байты или байты данных в SLEEP режиме микроконтроллера. После приема байта микроконтроллер выходит из SLEEP режима, если разрешены прерывания от SSP модуля.

15.4.5 Эффект сброса

При сбросе микроконтроллера модуль SSP выключается, прекращается любой обмен данными.

Таблица 15-3 Регистры и биты, связанные с работой модуля SSP в режиме I^2C

Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Сброс POR, BOR	Другие сбросы
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000u
PIR	SSPIF(1)								0	0
PIE	SSPIE(1)								0	0
SSPBUF	Буфер приемника SSP / регистр передатчика								xxxx xxxx	uuuu uuuu
SSPADD	Регистр адреса SSP (I^2C режим)								0000 0000	0000 0000
SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
SSPSTAT	SMP	CKE	D/-A	P	S	R/-W	UA	BF	0000 0000	0000 0000

Обозначения: - = не используется, читается как 0; u = не изменяется; x = не известно; q = зависит от условий.
Затененные биты на работу не влияют.
Примечания:

1. Расположение битов смотрите в технической документации на микроконтроллер.
2. В некоторых микроконтроллерах эти биты могут обозначаться как GPIE и GPIF.

15.5 Инициализация

Пример 15-2 Инициализация модуля SSP в режиме ведущего SPI

```
CLRF    STATUS          ; Банк 0
CLRF    SSPSTAT         ; SMP = 0, CKE = 0, и сбросить биты статуса
BSF     SSPSTAT, CKE    ; CKE = 1
MOVLW   0x31            ; Установить режим ведущего SPI, CLK/16,
MOVWF   SSPCON          ; сдвиг данных по заднему фронту (CKE=1 & CKP=1)
                        ; Выборка данных в середине такта (SMP=0 & режим ведущего)
BSF     STATUS, RP0     ; Банк 1
BSF     PIE, SSPIE      ; Разрешить прерывания от SSP модуля
BCF     STATUS, RP0     ; Банк 0
BSF     INTCON, GIE     ; Разрешить прерывания
MOVLW   DataByte        ; Получить байт передаваемых данные из памяти
MOVWF   SSPBUF          ; Начать передачу байта данных
```

15.5.1 Совместимость модуля SSP и основного модуля SSP (BSSP)

В модуле SSP (по сравнению с BSSP) в регистре SSPSTAT содержится два дополнительных служебных бита, которые используются только в режиме SPI:

- SMP - управление выборкой данных в режиме SPI;
- CKE - выбор активного фронта тактового сигнала в режиме SPI.

Для обеспечения совместимости модулей SSP и BSSP эти биты должны находиться в состоянии, показанном в таблице 15-4. Если не выдержать требования таблицы 15-4, данные передаваемые по интерфейсу SPI могут быть искажены.

Таблица 15-4 Требования к состоянию служебных битов для совместимости SSP и BSSP модулей

Модуль BSSP	**Модуль SSP**		
CKP	**CKP**	**CKE**	**SMP**
1	1	0	0
0	0	0	0

15.6 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Не могу организовать обмен данными с другим устройством, работающим по интерфейсу SPI.
Ответ 1:
Необходимо гарантировать, что Вы выбрали правильный режим SPI для этого устройства. Модуль SSP поддерживает все четыре режима SPI, вероятно Вы где-то ошиблись. Проверьте полярность тактового сигнала и выборку данных.

Вопрос 2: Не могу включить режим ведущего I^2C.
Ответ 2:
Модуль SSP аппаратно полностью не поддерживает режим ведущего I^2C, необходимы дополнительные программные модули. Обратите внимание на документ AN578, в нем представлено программное обеспечение, использующее модуль SSP для работы в режиме ведущего I^2C. Некоторые микроконтроллеры PICmicro содержат модуль MSSP, аппаратно поддерживающий режим ведущего I^2C.

Вопрос 3: В режиме I^2C не могу передать данные, хотя запись в регистр SSPBUF выполняю.
Ответ 3:
После записи в SSPBUF необходимо установить в '1' бит CKP, чтобы "отпустить" тактовый сигнал I^2C.

15.7 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с модулем SSP в микроконтроллерах PICmicro MCU:

Документ	Номер
Use of the SSP Module in the I 2 C Multi-Master Environment Использование модуля SSP в режиме ведущего I^2C с конкуренцией на шине	AN578
Using Microchip 93 Series Serial EEPROMs with Microcontroller SPI Ports Использование интерфейса SPI для связи с последовательной памятью EEPROM серии 93	AN613
Software Implementation of I 2 C Bus Master Программная реализация ведущего шины I^2C	AN554
Interfacing PIC16C64/74 to Microchip SPI Serial EEPROM Подключение к PIC16C64/74 последовательной EEPROM памяти с интерфейсом SPI	AN647
Interfacing a Microchip PIC16C92x to Microchip SPI Serial EEPROM Подключение к PIC16C92x последовательной EEPROM памяти с интерфейсом SPI	AN668

Раздел 16. Основной модуль SSP (BSSP)

Содержание

Примечание. Обратитесь к приложению C.2 или технической документации на микроконтроллеры, чтобы определить в каких микроконтроллерах реализован модуль BSSP.

16.1 Введение

Модуль основного синхронного последовательного порта (BSSP) может использоваться для связи с периферийными микросхемами или другими микроконтроллерами. Периферийными микросхемами могут быть: EEPROM память, сдвиговые регистры, драйверы ЖКИ, АЦП и др. Модуль BSSP может работать в одном из двух режимов:

- Последовательный периферийный интерфейс (SPI);
- Inter-Integrated Circuit (I^2C):
  - ведомой режим;
  - контроль состояния портов ввода/ввода для обнаружения битов START, STOP с целью упрощения программного обеспечения в режиме ведущего и при конкуренции на шине.

16.2 Управляющие регистры

SSPSTAT: Регистр статуса модуля BSSP

U-0	U-0	R-0	R-0	R-0	R-0	R-0	R-0
-	**-**	**D/-A**	**P**	**S**	**R/-W**	**UA**	**BF**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

биты 7,6: **Не реализованы:** Читаются как '0'

бит 5: **D/-A:** Бит Данные/Адрес (только для режима I^2C)
1 = последний принятый или переданный байт является информационным
0 = последний принятый или переданный байт является адресным

бит 4: **P:** Бит STOP (только для режима I^2C)
Этот бит сбрасывается в '0' когда модуль BSSP выключен.
1 = указывает, что бит STOP был обнаружен последним (этот бит равен '0' после сброса)
0 = бит STOP не является последним

бит 3: **S:** Бит START (только для режима I^2C)
Этот бит сбрасывается в '0' когда модуль BSSP выключен.
1 = указывает, что бит START был обнаружен последним (этот бит равен '0' после сброса)
0 = бит START не является последним

бит 2: **R/-W:** Бит чтения/записи (только для режима I^2C)
Значение бита действительно только после совпадения адреса и до приема бита START, STOP или -ACK.
1 = чтение
0 = запись

бит 1: **UA:** Флаг обновления адреса устройства (только для режима 10-разрядного I^2C)
1 = необходимо обновить адрес в регистре SSPADD
0 = обновление адреса не требуется

бит 0: **BF:** Бит статуса буфера
Прием (SPI и I^2C режимы)
1 = прием завершен, буфер SSPBUF полон
0 = прием не завершен, буфер SSPBUF пуст

Передача (только I^2C режима)
1 = выполняется передача данных, буфер SSPBUF полон
0 = передача данных завершена, буфер SSPBUF пуст

SSPCON: Регистр управления модуля BSSP

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	**SSPOV**	**SSPEN**	**CKP**	**SSPM3**	**SSPM2**	**SSPM1**	**SSPM0**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **WCOL:** Бит конфликта записи (сбрасывается в '0' программно)
1 = была предпринята попытка записи в SSPBUF во время передачи предыдущего байта
0 = конфликта не было

бит 6: **SSPOV:** Бит переполнения приемника
SPI режим
1 = принят новый байт, а SSPBUF содержит предыдущие данные(байт в SSPSR будет потерян). В ведомом режиме пользователь должен прочитать содержимое регистра SSPBUF даже, если только передает данные. В ведущем режиме бит в '1' не устанавливается, т.к. каждая операция инициализируется записью в SSPBUF. (сбрасывается в '0' программно)
0 = нет переполнения

I^2C режим
1 = принят новый байт, а SSPBUF содержит предыдущие данные. Значение бита не действительно при передаче данных. (сбрасывается в '0' программно)
0 = нет переполнения

бит 5: **SSPEN:** Бит включения модуля BSSP
Когда модуль включен, соответствующие порты ввода/вывода настраиваются на выход или вход
SPI режим
1 = модуль BSSP включен, выводы SCK, SDO, SDI, -SS используются модулем BSSP
0 = модуль BSSP выключен, выводы работают как цифровые порты ввода/вывода

I^2C режим
1 = модуль BSSP включен, выводы SDA, SCL используются модулем BSSP
0 = модуль BSSP выключен, выводы работают как цифровые порты ввода/вывода

бит 4: **CKP:** Бит выбора полярности тактового сигнала
SPI режим
1 = пассивный высокий уровень сигнала
0 = пассивный низкий уровень сигнала

I^2C режим
Управление тактовым сигналом SCK
1 = не управлять тактовым сигналом
0 = удерживать тактовый сигнал в низком логическом уровне (используется для подготовки данных)

биты 3-0: **SSPM3:SSPM0:** Режим работы модуля BSSP
0000 = ведущий режим SPI, тактовый сигнал = $F_{OSC}/4$
0001 = ведущий режим SPI, тактовый сигнал = $F_{OSC}/16$
0010 = ведущий режим SPI, тактовый сигнал = $F_{OSC}/64$
0011 = ведущий режим SPI, тактовый сигнал = выход TMR2 / 2
0100 = ведомый режим SPI, тактовый сигнал с вывода SCK. Вывод -SS подключен к SSP
0101 = ведомый режим SPI, тактовый сигнал с вывода SCK. Вывод -SS не подключен к SSP
0110 = ведомый режим I^2C, 7-разрядная адресация
0111 = ведомый режим I^2C, 10-разрядная адресация
1000 = резерв
1001 = резерв
1010 = резерв
1011 = программная поддержка ведущего режима I^2C (ведомый режим выключен)
1100 = резерв
1101 = резерв
1110 = ведомый режим I^2C, 7-разрядная адресация с разрешением прерываний по приему бит START и STOP
1111 = ведомый режим I^2C, 10-разрядная адресация с разрешением прерываний по приему бит START и STOP

16.3 Режим SPI

В SPI режиме возможен одновременный синхронный прием и передача 8-разрядных данных. Модуль BSSP в типовом включении использует три вывода микроконтроллера:

- Вход последовательных данных (SDI);
- Выход последовательных данных (SDO);
- Тактовый сигнал (SCK).

Дополнительно может быть задействован четвертый вывод для работы в режиме ведомого:

- Выбор ведомого (-SS).

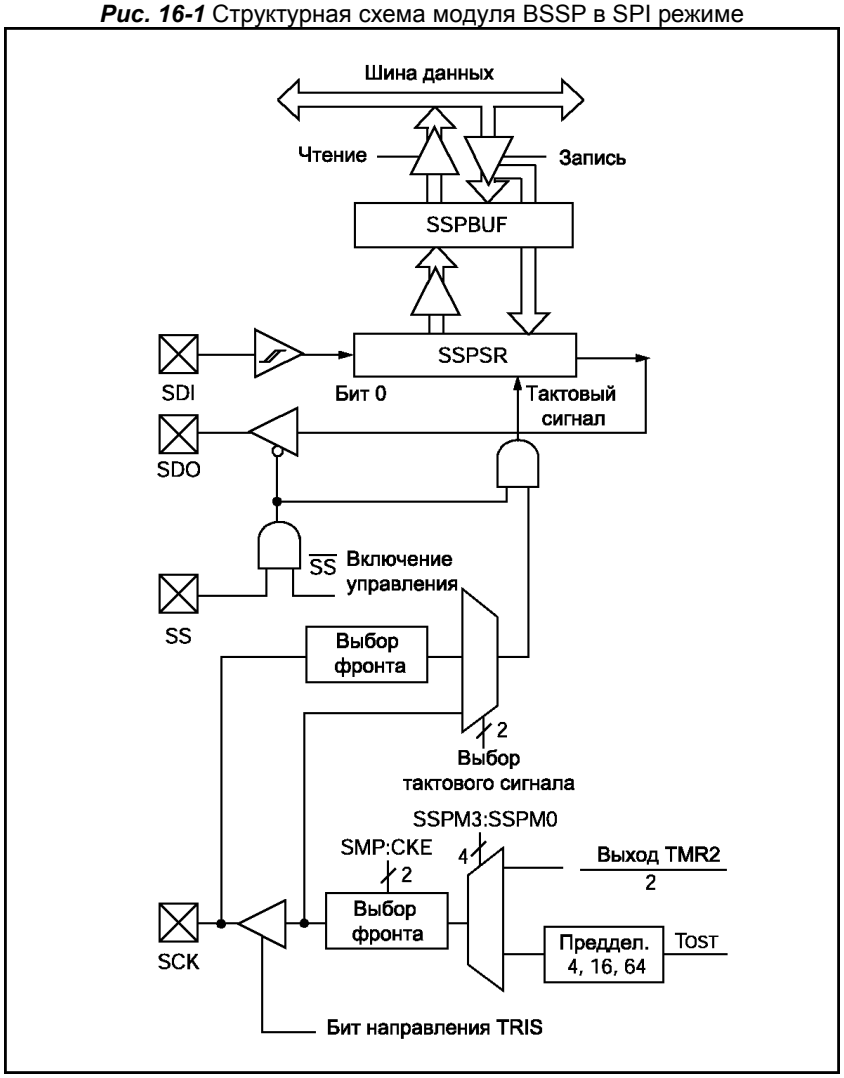
16.3.1 Работа модуля BSSP в режиме SPI

При инициализации SPI необходимо определить параметры работы модуля SPI битами SSPCON<5:0>. Управляющие биты определяют следующие параметры работы:

- Ведущий режим (SCK выход);
- Ведомый режим (SCK вход);
- Полярность тактового сигнала (пассивный уровень SCK);
- Частота тактового сигнала (только в ведущем режиме);
- Режим выбора ведомого (только в режиме ведомого).

На рисунке 16-1 показана структурная схема модуля BSSP в SPI режиме.

Рис. 16-1 Структурная схема модуля BSSP в SPI режиме



Модуль BSSP состоит из приемного/передающего регистра сдвига (SSPSR) и буферного регистра (SSBUF). В регистре SSPSR выполняется сдвиг данных из/в микроконтроллер старшим битом вперед. В регистре SSPBUF сохраняются записанные данные, пока не будут получены новые. Приняв 8 бит данных в регистр SSPSR они переписываются в SSPBUF, устанавливается в '1' флаг полного приемного буфера BF (SSPSTAT<0>) и флаг прерывания SSPIF. Двойная буферизация принимаемых данных позволяет принимать следующий байт до чтения предыдущего. Любая запись в регистр SSPBUF во время выполнения операции приема/передачи данных будет игнорирована, при этом устанавливается в '1' флаг WCOL (SSPCON<7>). Пользователь должен программно сбросить бит WCOL в '0', чтобы была возможность проверки выполнения записи в регистр SSPBUF. При приеме данных в режиме SPI регистр SSPBUF должен быть прочитан до момента окончания приема следующего байта. Бит статуса приемного буфера BF (SSPSTAT<0>) указывает на получение нового байта данных. Бит BF аппаратно сбрасывается в '0' при чтении регистра SSPBUF. Принятые данные могут быть недостоверными, если режим SPI используется только для передачи данных. Прерывания от модуля BSSP используются для определения завершения приема/передачи данных (в подпрограмме обработки прерываний необходимо прочитать/записать регистр SSPBUF). Если не планируется использовать прерывания от модуля BSSP, то необходимо предусмотреть программную проверку выполнения записи в регистр SSPBUF для передачи данных. В примере 16-1 показана загрузка данных в регистр SSPBUF (SSPSR) для передачи данных. Затененная команда требуется только, если принимаемые данные имеют какое-то значение (в некоторых приложениях модуль BSSP в режиме SPI используется только для передачи данных).

Пример 16-1 Загрузка данных в регистр SSPBUF(SSPSR)

```
        BCF    STATUS, RP1   ;Банк 1
        BSF    STATUS, RP0   ;
LOOP    BTFSS  SSPSTAT, BF   ;Данные приняты?
        GOTO   LOOP          ;Нет
        BCF    STATUS, RP0   ;Банк 0
        MOVF   SSPBUF, W     ;Загрузить в W значение из SSPBUF
        MOVWF  RXDATA        ;Если необходимо, сохранить значение в памяти
        MOVF   TXDATA, W     ;Загрузить в W значение из TXDATA
        MOVWF  SSPBUF        ;Передать новые данные
```

Регистр SSPSR не доступен для непосредственного чтения или записи, все операции выполняются через регистр SSPBUF. В регистре SSPSTAT находятся биты, указывающие текущее состояние модуля BSSP.

16.3.2 Настройка выводов в режиме SPI

Для включения модуля BSSP необходимо установить бит SSPEN (SSPCON<5>) в '1'. Для сброса или перенастройки режима SPI рекомендуется сбросить бит SSPEN в '0', выполнить изменения параметров работы, а затем вновь установить бит SSPEN в '1'. После включения BSSP в режиме SPI выводы SDI, SDO, SCK, -SS используются последовательным портом. Для корректной работы последовательного порта биты регистров TRIS должны быть настроены следующем образом:

- SDI, бит TRIS должен быть установлен в '1';
- SDO, бит TRIS должен быть сброшен в '0';
- SCK (ведущий режим), бит TRIS должен быть сброшен в '0';
- SCK (ведомый режим), бит TRIS должен быть установлен в '1';
- -SS, бит TRIS должен быть установлен в '1'.

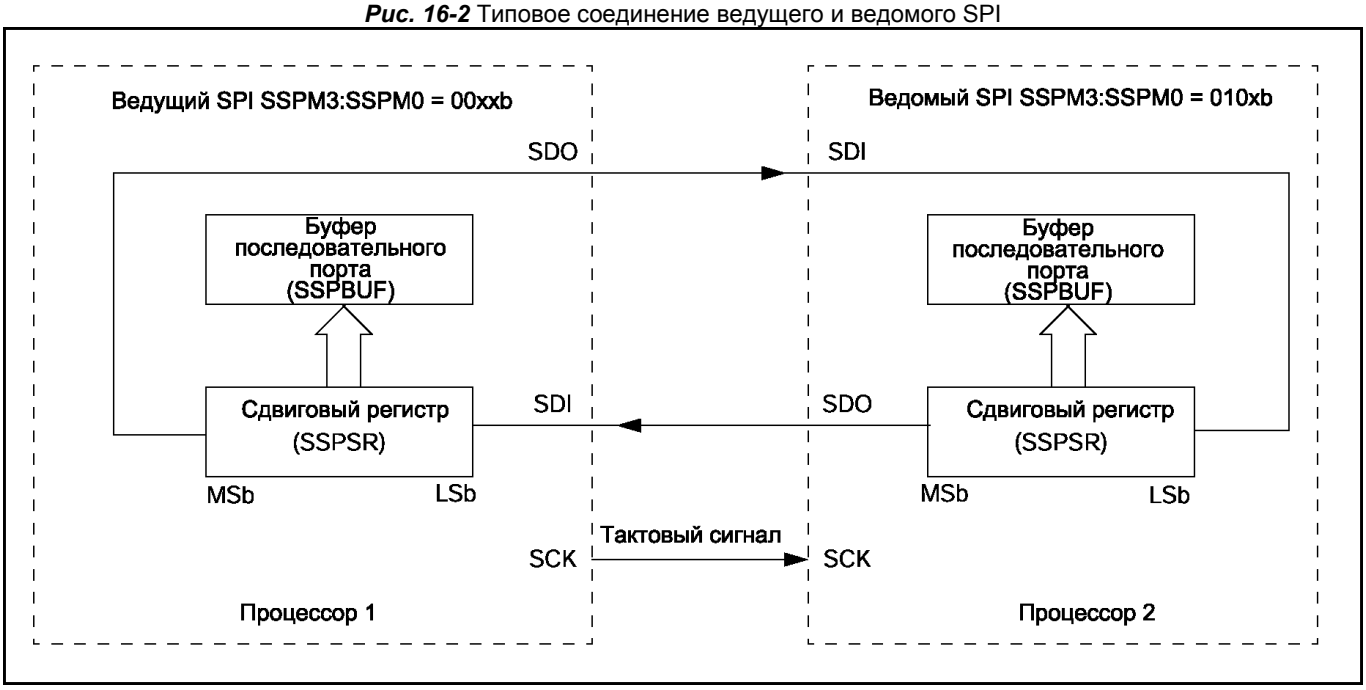
Любая нежелательная функция последовательного порта может быть выключена, настраивая соответствующие биты регистров направления данных TRIS. Например, если в режиме ведущего SPI выполняется только передача данных, то выводы SDI и -SS могут использоваться как цифровые выходы, сбросив соответствующие биты TRIS в '0'.

16.3.3 Типовое включение

На рисунке 16-2 показано типовое соединение двух микроконтроллеров. Главный микроконтроллер (процессор 1) инициализирует передачу, формируя тактовый сигнал SCK. Данные сдвигаются по активному фронту тактового сигнала. Для одновременного приема/передачи данных (фиктивных данных) оба микроконтроллера должны иметь одинаковую полярность тактового сигнала (бит CKP). Всего существует три сценария передачи данных:

- Ведущий передает данные - ведомый передает фиктивные данные;
- Ведущий передает данные - ведомый передает данные;
- Ведущий передает фиктивные данные - ведомый передает данные.

Рис. 16-2 Типовое соединение ведущего и ведомого SPI



16.3.4 Режим ведущего SPI

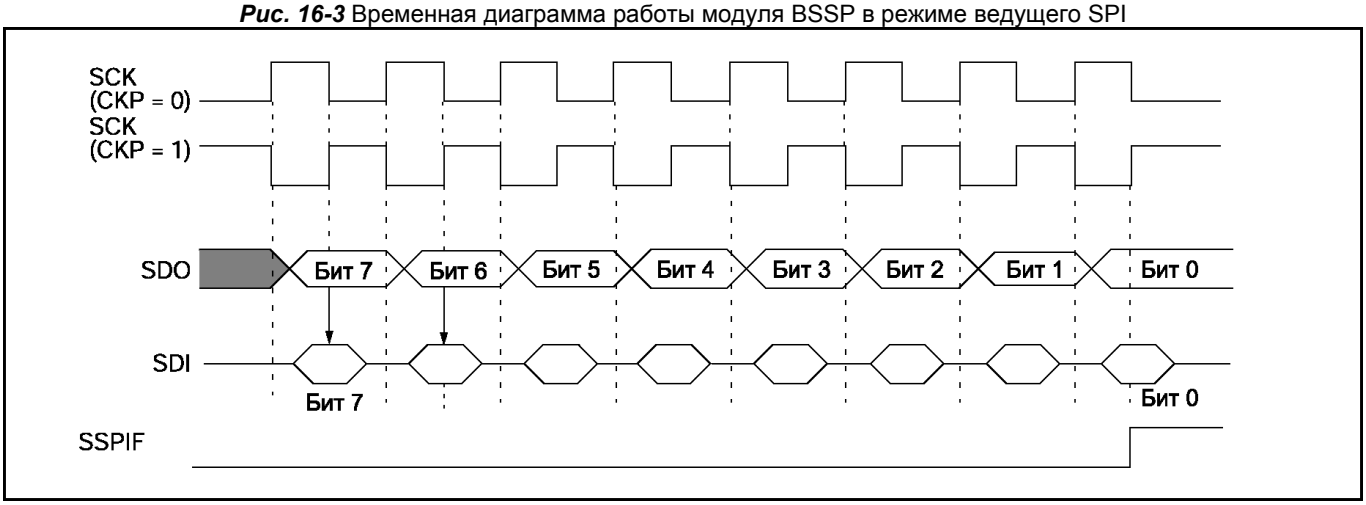
Ведущий шины может инициализировать передачу данных в любой момент, поскольку он генерирует тактовый сигнал, и определяет, когда ведомый (процессор 2) должен передать данные в соответствии с используемым протоколом.

В режиме ведущего данные передаются/приняты после их записи/чтения из регистра SSPBUF. Если в SPI режиме требуется только принимать данные, вывод SDO может быть заблокирован (настроен как вход). Данные с вывода SDI последовательно сдвигаются в регистр SSPSR с установленной скоростью. Каждый принятый байт загружается в регистр SSPBUF (как нормально полученный байт) с формированием прерываний и воздействием на соответствующие биты статуса. Эта функция может быть полезна при реализации "монитора шины".

Полярность тактового сигнала устанавливается битом CKP (SSPCON<4>), что позволяет получить различные методы передачи данных. Данные всегда передаются старшим битом вперед. В ведущем режиме частота тактового сигнала выбирается программно:

- $F_{OSC}/4$ (или T_{CY});
- $F_{OSC}/16$ (или 4 x T_{CY});
- $F_{OSC}/64$ (или 16 x T_{CY});
- Выход таймера TMR2 / 2.

Рис. 16-3 Временная диаграмма работы модуля BSSP в режиме ведущего SPI



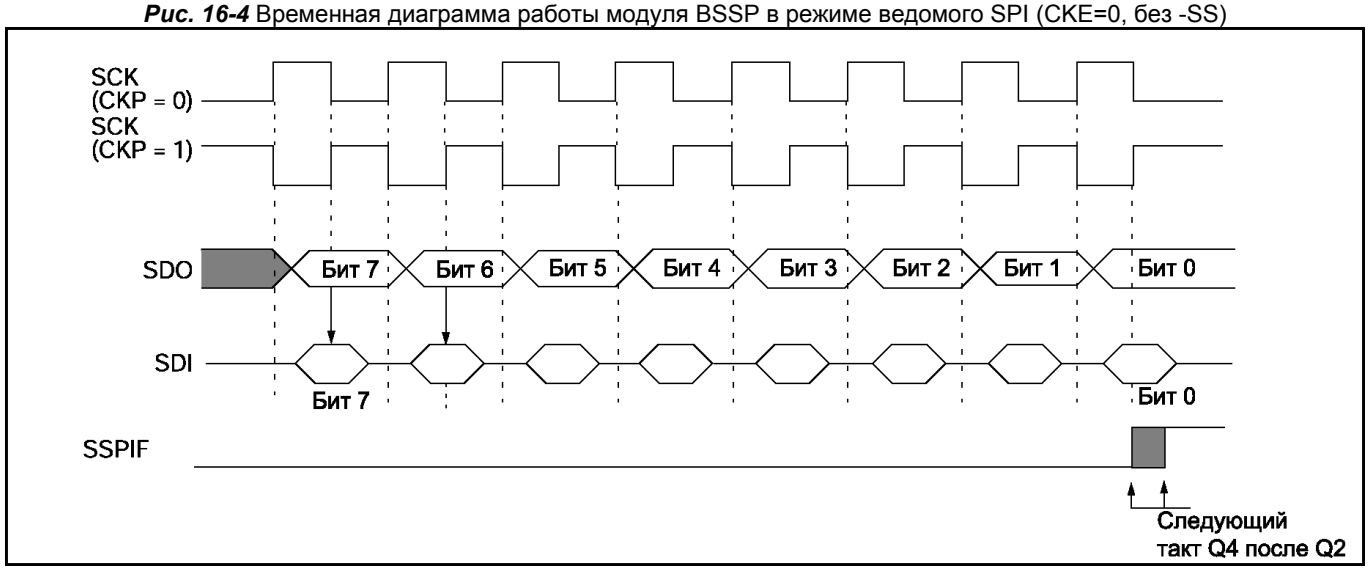
16.3.5 Режим ведомого SPI

В режиме ведомого данные передаются/принимаются по внешнему тактовому сигналу на выводе SCK. Когда принимается последний бит байта, устанавливается в '1' флаг прерываний SSPIF.

Полярность тактового сигнала выбирается битом CKP (SSPCON<4>). Временные диаграммы передачи данных по интерфейсу SPI смотрите на рисунке 16-3, 16-4 (данные передаются старшим битом вперед). Внешний тактовый сигнал должен удовлетворять требованиям длительности низкого и высокого логического уровня, описанным в разделе электрических характеристик.

В SLEEP режиме микроконтроллера ведомый может принимать/передавать данные. После приема данных микроконтроллер выходит из режима SLEEP, если разрешены прерывания от модуля BSSP.

Рис. 16-4 Временная диаграмма работы модуля BSSP в режиме ведомого SPI (CKE=0, без -SS)



16

16.3.6 Выбор ведомого в режиме SPI

В режиме SPI вывод -SS позволяет подключать несколько ведомых к одному ведущему. Модуль BSSP должен находиться в режиме ведомого SPI (SSPCON<3:0> = 0100), бит TRIS для вывода -SS установлен в '1', чтобы позволить ведущему выбирать ведомого. Когда на выводе -SS присутствует низкий логический уровень, передача и прием данных разрешены, а вывод SDO управляется модулем BSSP. Если на выводе -SS высокий уровень сигнала, то вывод SDO переходит в 3-е состояние. В зависимости от приложения может потребоваться внешний подтягивающий резистор на выводе SDO.

Для реализации двух проводного интерфейса вывод SDO может быть соединен с SDI. Когда SPI должен работать как приемник, вывод SDO настраивается на вход, что отключает передатчик от SDO. SDI всегда должен быть настроен как вход (функция SDI), т.к. это не создает конфликт шины.

Рис. 16-5 Временная диаграмма работы модуля BSSP в режиме SPI с выбором ведомого

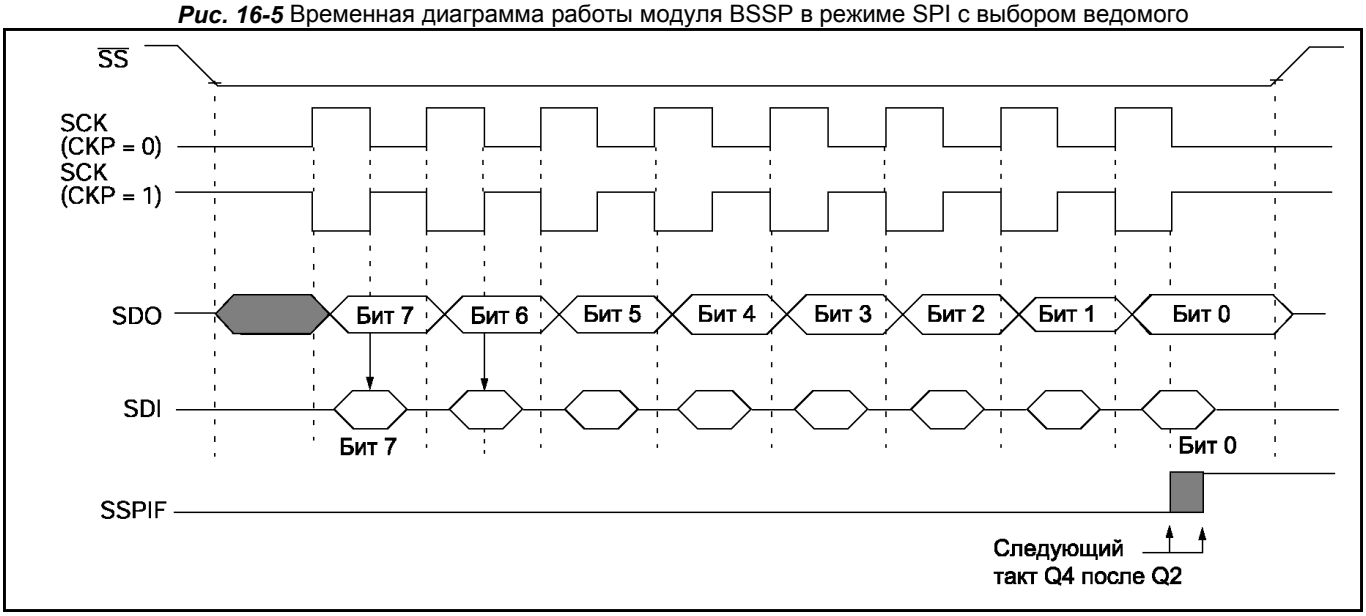
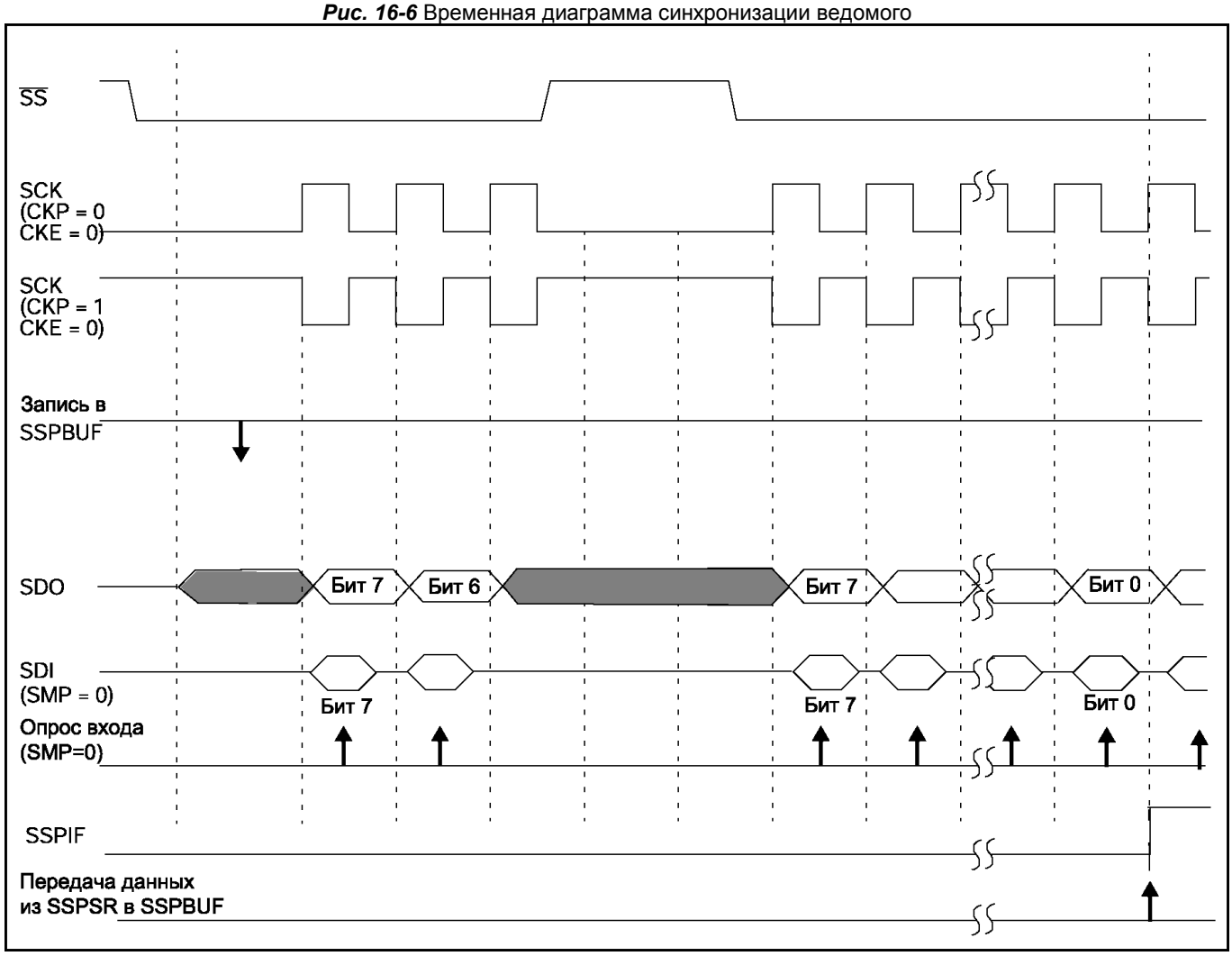


Рис. 16-6 Временная диаграмма синхронизации ведомого



16.3.7 Работа в SLEEP режиме микроконтроллера

В режиме ведущего SPI тактовый сигнал модуля BSSP отсутствует, состояние приема/передачи данных не изменяется до выхода микроконтроллера из режима SLEEP. После выхода микроконтроллера из режима SLEEP модуль SSP продолжит передачу/прием данных.

В режиме ведомого SPI данные могут быть приняты/переданы, т.к. сдвиговый регистр работает асинхронно. Это позволяет в SLEEP режиме микроконтроллера принять/передать данные в/из сдвигового регистра. Как только будут приняты все 8 бит данных, устанавливается в '1' флаг прерывания от модуля BSSP, и если прерывания разрешены, микроконтроллер выйдет из SLEEP режима.

16.3.8 Эффект сброса

Любой сброс микроконтроллера выключает модуль BSSP, прием/передача данных прекращается.

Таблица 16-1 Регистры и биты, связанные с работой модуля BSSP в режиме SPI

Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Сброс POR, BOR	Другие сбросы
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000u
PIR	SSPIF(1)								0	0
PIE	SSPIE(1)								0	0
SSPBUF	Буфер приемника BSSP / регистр передатчика								xxxx xxxx	uuuu uuuu
SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
SSPSTAT	-	-	D/-A	P	S	R/-W	UA	BF	--00 0000	--00 0000

Обозначения: - = не используется, читается как 0; u = не изменяется; x = не известно; q = зависит от условий.
Затененные биты на работу не влияют.

Примечания:

1. Расположение битов смотрите в технической документации на микроконтроллер.
2. В некоторых микроконтроллерах эти биты могут обозначаться как GPIE и GPIF.

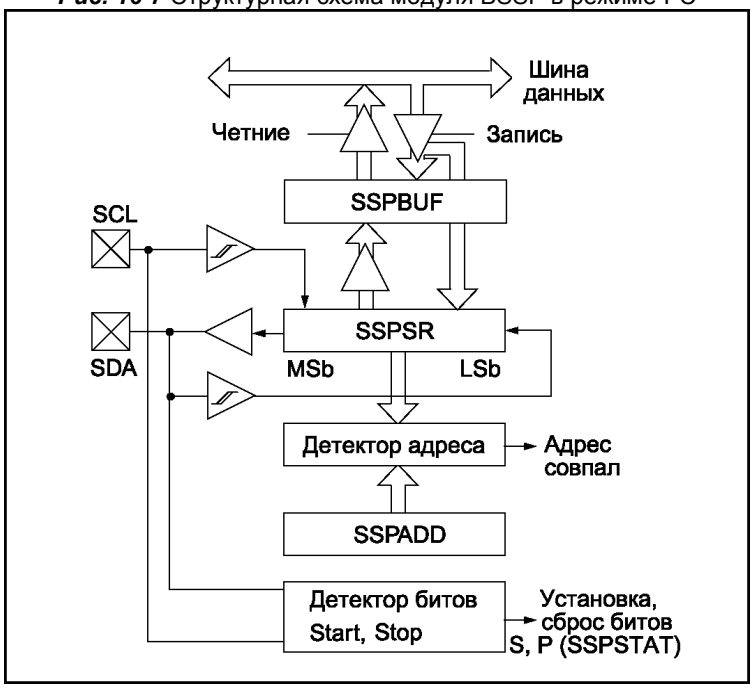
16.4 Режим I²C

Модуль BSSP полностью поддерживает все функции ведомых устройств, включая поддержку общего вызова, аппаратные прерывания по детектированию битов START и STOP для определения занятости шины I²C при программной реализации режима ведущего. В BSSP модуле реализована поддержка стандартного режима 7, 10-разрядной адресации. Дополнительно смотрите приложение A, в котором дано краткое описание шины I²C.

Для работы с шиной I²C используется два вывода SCL (сигнал синхронизации) и SDA (данные). Выводы SDA и SCL автоматически настраиваются при включении режима I²C. Включение модуля BSSP выполняется установкой бита SSPEN (SSPCO<5>) в '1'.

Фильтр "glitch" подключен к выводам SDA и SCL, когда они настроены на вход. Фильтр работает в режимах 100кГц и 400кГц. В режиме 100кГц, когда выводы SDA и SCL настроены на выход, фильтр контролирует длительность формируемых сигналов независимо от тактовой частоты микроконтроллера.

Рис. 16-7 Структурная схема модуля BSSP в режиме I²C



Для управления модулем BSSP в режиме I²C используется пять регистров:

- SSPCON, регистр управления BSSP;
- SSPSTAT, регистр статуса BSSP;
- SSPBUF, буфер приемника/передатчика;
- SSPSR, сдвиговый регистр (пользователю не доступен);
- SSPADD, регистр адреса.

В регистре SSPCON устанавливается требуемый режим I²C. С помощью четырех битов (SSPCON<3:0>) можно выбрать один из режимов I²C:

- Ведомый режим I²C, 7-разрядная адресация;
- Ведомый режим I²C, 10-разрядная адресация;
- Программная поддержка ведущего режима I²C с конкуренцией на шине (разрешение прерываний по приему битов START и STOP);
- Программная поддержка ведущего режима I²C (ведомый режим выключен).

При выборе любого режима I²C выводы SCL и SDA должны быть настроены на вход, установкой соответствующих битов регистра TRIS в '1'. После выбора режима I²C и установки бита SSPEN в '1' выводы SDA (линия данных), SCL (линия синхронизации) подключаются к модулю SSP.

Регистр SSPSTAT содержит биты статуса передачи данных: обнаружение на шине битов START (S) или STOP (P), флаг приема байта данных или адреса, указатель загрузки старшего байта 10-разрядного адреса, бит операции приема/передачи.

В регистр SSPBUF загружаются данные для передачи по шине I²C, и из него читаются принятые данные. Регистр SSPSR выполняет сдвиг принимаемых/передаваемых данных. При приеме данных регистры SSPBUF, SSPSR работают как двухуровневый буфер приемника. Буфер позволяет принимать следующий байт до чтения предыдущего принятого байта из регистра SSPBUF. Когда байт полностью загружен в SSPSR, он передается в регистр SSPBUF и устанавливается флаг прерывания SSPIF в '1'. Если полностью принят следующий байт до чтения предыдущего байта из SSPBUF, то устанавливается бит SSPOV (SSPCON<6>) в '1', а байт в регистре SSPSR будет потерян.

В регистр SSPADD записывается адрес ведомого устройства. В 10-разрядном режиме пользователь должен сначала записывать старший байт адреса (1111 0 A9 A8 0). После соответствия старшего байта адреса необходимо загрузить младший байт адреса (A7:A0).

16.4.1 Режим ведомого I^2C

В режиме ведомого I^2C выводы SCL, SDA должны быть настроены на вход. Модуль BSSP автоматически изменит направление вывода SDA при передаче данных ведомым.

При совпадении адреса или после приема байта данных (если предварительно совпал адрес) аппаратно генерируется бит подтверждения (-ACK), а затем данные из регистра SSPSR загружаются в SSPBUF.

Существует несколько условий, при которых бит -ACK не формируется (эти условия могут возникать одновременно):

a) Бит BF (SSPSTAT<0>) = 1 перед приемом данных;
b) Бит переполнения SSPOV (SSPSTAT<6>) = 1 перед приемом данных.

Если бит BF = 1, то значение из SSPSR не переписывается в регистр SSPBUF, а биты SSPIF и SSPOV устанавливаются в '1'. В таблице 16-2 показаны операции после приема байта при различных значениях битов BF, SSPOV. В затененных ячейках показана ситуация, когда вовремя не был сброшен бит переполнения SSPOV в '0'. Заметьте, что бит BF аппаратно сбрасывается в '0' при чтении из регистра SSPBUF, а бит SSPOV необходимо сбрасывать в '0' программно.

Минимальная длительность логических уровней входного сигнала синхронизации SCL должна удовлетворять требованиям раздела электрических характеристик (см. параметры 100 и 101).

16.4.1.1 Адресация

После включения модуля BSSP ожидается формирование на шине бита START. Получив бит START, принимается 8 бит в сдвиговый регистр SSPSR. Выборка битов происходит по переднему фронту синхронизирующего сигнала на выводе SCL. По заднему фронту восьмого такта сигнала SCL значение в регистре SSPSR<7:1> сравнивается с содержимым регистра SSPADD. Если значение адреса совпадает, а биты BF и SSPOV равны нулю, то выполняются следующие действия:

a) Значение регистра SSPSR загружается SSPBUF по 8-му заднему фронту сигнала SCL;
b) Устанавливается флаг BF в '1' (буфер полон) по 8-му заднему фронту сигнала SCL;
c) Генерируется бит -ACK;
d) Устанавливается флаг прерываний SSPIF в '1' (если разрешено, генерируется прерывание) по 9-му заднему фронту сигнала SCL.

В режиме ведомого при 10-разрядной адресации необходимо принять два байта адреса. Пять старших бит первого байта определяют: является ли полученный байт первым байтом 10-разрядного адреса. Бит R/-W(SSPSTAT<2>) должен быть настроен для приема второго байта адреса. Для 10-разрядной адресации первый байт адреса должен иметь формат '1111 0 A9 A8 0', где A9:A8 два старших бита адреса. Рекомендуемая последовательность действий при 10-разрядной адресации (шаги 7-9 для передачи ведомым):

1. Принять старший байт адреса (устанавливаются биты SSPIF, BF и UA (SSPSTAT<1> в '1').
2. Записать младший байт адреса в регистр SSPADD (аппаратно сбрасывается бит UA в '0' и "отпускается" линия SCL).
3. Выполнить чтение из регистра SSPBUF (сбрасывается бит BF в '0') и сбросить флаг SSPIF в '0'.
4. Принять младший байт адреса (устанавливаются биты SSPIF, BF и UA (SSPSTAT<1> в '1').
5. Записать старший байт адреса в регистр SSPADD (аппаратно сбрасывается бит UA в '0' и "отпускается" линия SCL).
6. Выполнить чтение из регистра SSPBUF (сбрасывается бит BF в '0') и сбросить флаг SSPIF в '0'.
7. Принять бит повторный START.
8. Принять старший байт адреса (устанавливаются биты SSPIF и BF в '1').
9. Выполнить чтение из регистра SSPBUF (сбрасывается бит BF в '0') и сбросить флаг SSPIF в '0'.

Примечание. В 10-разрядном режиме после команды повторный START (шаг 7) не требуется обновлять значение в регистре SSPADD. В данном случае требуется соответствие только первого байта адреса.

Таблица 16-2 Операции после приема байта при различных значениях битов BF, SSPOV

Биты статуса приемника		Запись из SSPSR в SSPBUF	Формирование бита -ACK	Установка флага прерываний SSPIF
BF	**SSPOV**			
0	0	Есть	Есть	Есть
1	0	Нет	Нет	Есть
1	1	Нет	Нет	Есть
0	1	Есть	Нет	Есть

Примечание. В затененных ячейках показана ситуация, когда вовремя не был сброшен бит переполнения SSPOV в '0'.

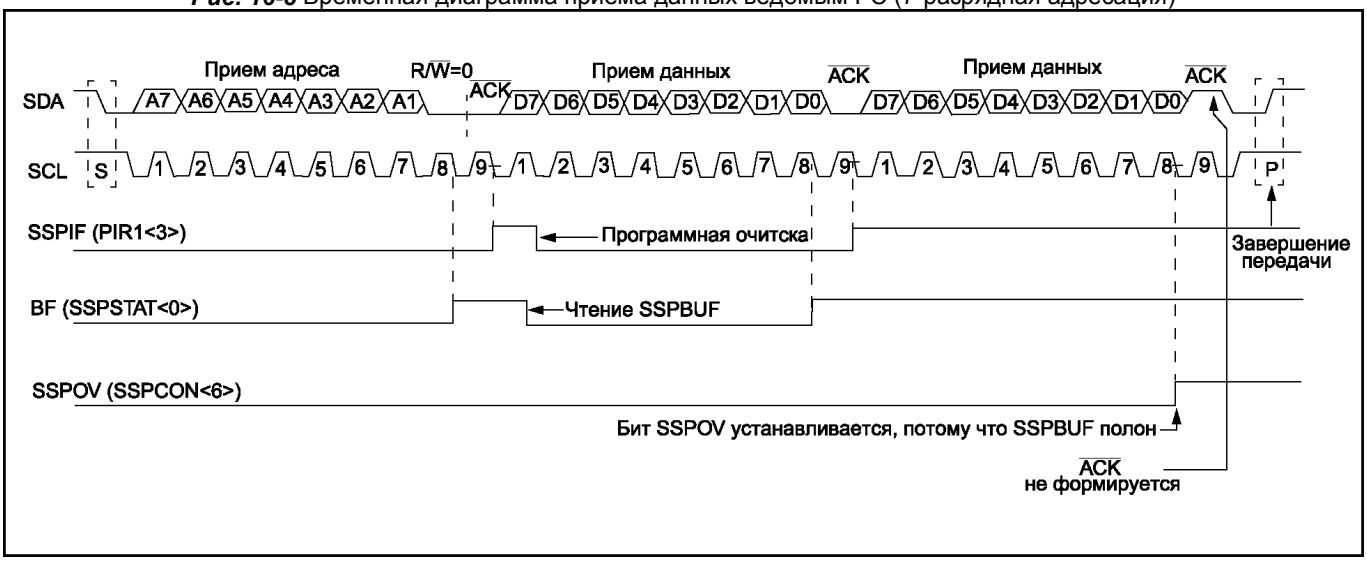
16.4.1.2 Прием данных

Если бит R/-W в адресном байте равен нулю, а принятый адрес совпадает с адресом устройства, то бит R/-W в регистре SSPSTAT сбрасывается в '0'. Принятый адрес загружается в регистр SSPBUF.

Если бит BF (буфер полон) или SSPOV (переполнение буфера) установлен в '1', то бит подтверждения -ACK не формируется. Эту ошибку необходимо обработать программно. Если было выполнено чтение из регистра SSPBUF но не был сброшен бит SSPOV в '0', то бит -ACK не формируется.

Прерывание от модуля BSSP генерируются при каждом принятом байте с шины I^2C, установкой флага SSPIF в '1' (сбрасывается программно). Регистр SSPSTAT используется для определения типа принятого байта.

Рис. 16-8 Временная диаграмма приема данных ведомым I^2C (7-разрядная адресация)



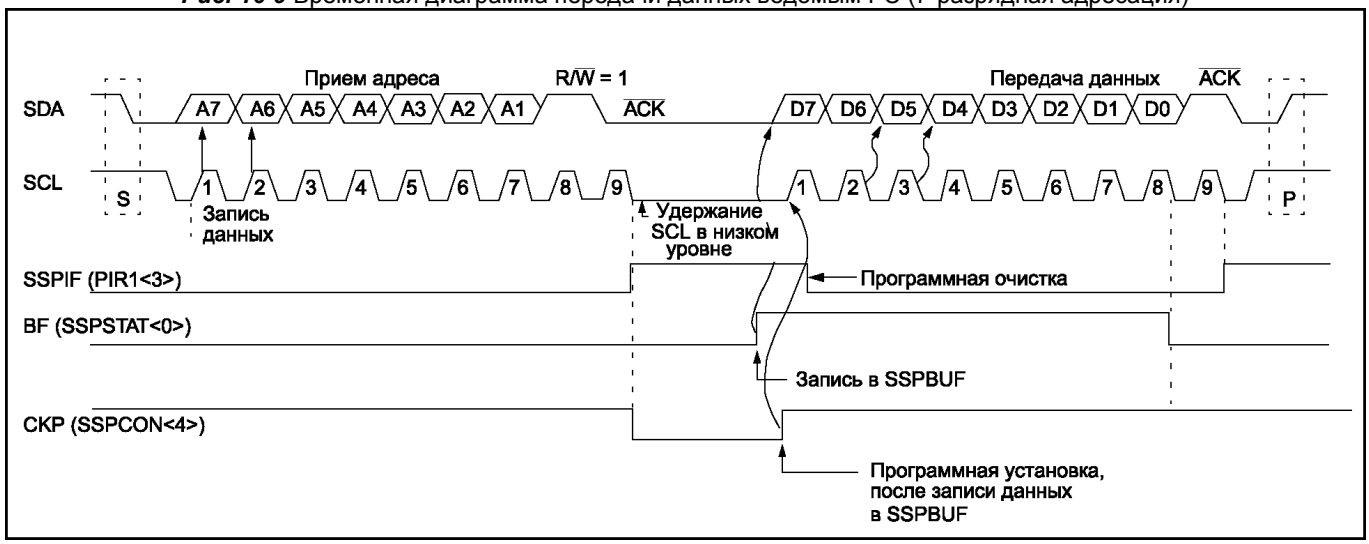
16.4.1.3 Передача данных

Если бит R/-W в адресном байте равен '1', а принятый адрес совпадает с адресом устройства, то бит R/-W в регистре SSPSTAT устанавливается в '1'. Принятый адрес загружается в регистр SSPBUF. Бит -ACK формируется девятым битом, после чего линия SCL удерживается в низком логическом уровне. Передаваемые данные должны быть записаны в регистр SSPBUF, после чего они автоматически переписываются в регистр SSPSR. После записи данных необходимо "отпустить" сигнал SCL установкой бита CKP(SSPCON<4>) в '1'. Ведущий шины контролирует состояние линии SCL, ожидая смены уровня сигнала. Восемь бит загруженных данных последовательно сдвигаются по заднему фронту сигнала SCL, что гарантирует достоверное значение данных на линии SDA (см. рисунок 16-9).

Модуль BSSP генерирует прерывание по каждому переданному байту, устанавливая бит SPPIF в '1' по заднему фронту девятого такта сигнала SCL. Флаг SSPIF должен быть сброшен программно. Регистр SSPSTAT используется для определения статуса передачи данных.

Ведущее устройство формирует бит подтверждения -ACK на девятом такте сигнала SCL для каждого принятого байта. Если бит подтверждения -ACK не сформирован (высокий уровень сигнала SDA), передача данных завершена. Логика ведомого устройства настраивается на обнаружение бита START. Если бит подтверждения -ACK был получен (низкий уровень сигнала SDA), в регистр SSPBUF необходимо записать новый байт для передачи. Линию SCL также необходимо "отпустить", установкой бита CKP в '1'.

Рис. 16-9 Временная диаграмма передачи данных ведомым I^2C (7-разрядная адресация)



16.4.1.4 Арбитраж тактового сигнала

Арбитраж выполняется на линии SCL, чтобы запретить ведомому формировать следующий тактовый импульс. В режиме ведомого I^2C линия SCL будет удерживаться в низком логическом уровне, пока ЦПУ не ответит на прерывание (SSIF=1, CKP=0). Данные, которые нужно передать ведомому, записываются в регистр SSPBUF, затем устанавливается в '1' бит CKP, позволяя ведущему формировать тактовый сигнал.

16.4.2 Режим ведущего I^2C (программная реализация)

В режиме ведущего поддерживается генерация прерываний при обнаружении на шине битов START и STOP. Биты STOP (P) и START (S) в регистре SSPSTAT равны '0' после сброса микроконтроллера или при выключенном модуле BSSP. Шина находится в неактивном состоянии, если бит P=1 или оба бита S, P равны '0'.

В режиме ведущего управлением уровнем сигнала на линиях SCL и SDA выполняется сбросом соответствующих битов TRIS. На выходе всегда присутствует низкий логический уровень вне зависимости от состояния битов регистра PORT. Для передачи логической '1' соответствующий бит TRIS должен быть установлен в '1' (вывод настроить на вход), а для передачи '0' - сбросить бит TRIS в '0' (вывод настроить на выход). Аналогично выполняется управление сигналом SCL.

Следующие события на шине I^2C могут привести к установки флага прерываний SSPIF в '1':

- Выполнено условие START;
- Выполнено условие STOP;
- Передан/принят байт данных.

Режим ведущего может быть выбран с выключенным ведомым (SSPM3:SSPM0 = 1011) или включенным ведомым (SSPM3:SSPM0 = 1110 или 1111). Когда режим ведомого включен, программное обеспечение должно дифференцировать источник прерываний.

16.4.3 Режим ведущего I^2C с конкуренцией на шине (программная реализация)

В режиме ведущего с конкуренцией на шине поддерживается генерация прерываний при обнаружении на шине битов START и STOP. Биты STOP (P) и START (S) в регистре SSPSTAT равны '0' после сброса микроконтроллера или при выключенном модуле BSSP. Шина находится в неактивном состоянии, если бит P=1 (SSPSTAT<4>) или оба бита S, P равны '0'. Если шина занята, можно разрешить прерывания от SSP для обнаружения бита STOP на шине.

При конкуренции линия SDA должна проверяться на соответствия уровня, при ожидаемом высоком уровне на выходе. Если ожидается высокий уровень сигнала, а на линии присутствует сигнал с низким логическим уровнем, то необходимо "отпустить" линии SCL, SDA (установить в '1' биты TRIS). Арбитраж на шине I^2C может быть потерян во время:

- Передачи адреса;
- Передачи данных.

Когда ведомый режим включен, ведомый I^2C продолжает принимать данные. Когда арбитраж шины потерян во время передачи адреса, то сеанс связи можно продолжить, если получен бит подтверждения -ACK. Если арбитраж шины потерян во время передачи данных, то устройство должно повторить обмен данными позже.

16.4.4 Работа в SLEEP режиме

Ведомый I^2C может принимать адресные байты или байты данных в SLEEP режиме микроконтроллера. После приема байта микроконтроллер выходит из SLEEP режима, если разрешены прерывания от BSSP модуля.

16.4.5 Эффект сброса

При сбросе микроконтроллера модуль BSSP выключается, прекращается любой обмен данными.

Таблица 16-3 Регистры и биты, связанные с работой модуля BSSP в режиме I^2C

Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Сброс POR, BOR	Другие сбросы
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000u
PIR	SSPIF(1)								0	0
PIE	SSPIE(1)								0	0
SSPBUF	Буфер приемника SSP / регистр передатчика								xxxx xxxx	uuuu uuuu
SSPADD	Регистр адреса SSP (I^2C режим)								0000 0000	0000 0000
SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
SSPSTAT	-	-	D/-A	P	S	R/-W	UA	BF	--00 0000	--00 0000

Обозначения: - = не используется, читается как 0; u = не изменяется; x = не известно; q = зависит от условий.
Затененные биты на работу не влияют.
Примечания:

1. Расположение битов смотрите в технической документации на микроконтроллер.
2. В некоторых микроконтроллерах эти биты могут обозначаться как GPIE и GPIF.

16.5 Инициализация

Пример 16-2 Инициализация модуля BSSP в режиме ведущего SPI

```
CLRF    STATUS          ; Банк 0
CLRF    SSPSTAT         ; Сбросить биты статуса
MOVLW   0x31            ; Установить режим ведущего SPI, CLK/16,
MOVWF   SSPCON          ;
BSF     STATUS, RP0     ; Банк 1
BSF     PIE, SSPIE      ; Разрешить прерывания от BSSP модуля
BCF     STATUS, RP0     ; Банк 0
BSF     INTCON, GIE     ; Разрешить прерывания
MOVLW   DataByte        ; Получить байт передаваемых данные из памяти
MOVWF   SSPBUF          ; Начать передачу байта данных
```

16.5.1 Совместимость модуля SSP и основного модуля SSP (BSSP)

В модуле SSP (по сравнению с BSSP) в регистре SSPSTAT содержится два дополнительных служебных бита, которые используются только в режиме SPI:

- SMP - управление выборкой данных в режиме SPI;
- CKE - выбор активного фронта тактового сигнала в режиме SPI.

Для обеспечения совместимости модулей SSP и BSSP эти биты должны находиться в состоянии, показанном в таблице 16-4. Если не выдержать требования таблицы 16-4, данные передаваемые по интерфейсу SPI могут быть искажены.

Таблица 16-4 Требования к состоянию служебных битов для совместимости SSP и BSSP модулей

Модуль BSSP	**Модуль SSP**		
CKP	**CKP**	**CKE**	**SMP**
1	1	0	0
0	0	0	0

16.6 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Не могу организовать обмен данными с другим устройством, работающим по интерфейсу SPI.
Ответ 1:
Необходимо гарантировать, что Вы выбрали правильный режим SPI для этого устройства. Модуль BSSP поддерживает два из четырех режимов SPI (проверьте режимы SPI). Проверьте полярность тактового сигнала.
Если подключаемое устройство не поддерживает режим SPI модуля BSSP, то необходимо выбрать другой микроконтроллер PICmicro.

Вопрос 2: Не могу включить режим ведущего I^2C.
Ответ 2:
Модуль SSP аппаратно полностью не поддерживает режим ведущего I^2C, необходимы дополнительные программные модули. Обратите внимание на документ AN578, в нем представлено программное обеспечение, использующее модуль SSP для работы в режиме ведущего I^2C. Некоторые микроконтроллеры PICmicro содержат модуль MSSP, аппаратно поддерживающий режим ведущего I^2C.

Вопрос 3: В режиме I^2C не могу передать данные, хотя запись в регистр SSPBUF выполняю.
Ответ 3:
После записи в SSPBUF необходимо установить в '1' бит CKP, чтобы "отпустить" тактовый сигнал I^2C.

16.7 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с модулем BSSP в микроконтроллерах PICmicro MCU:

Документ	Номер
Use of the SSP Module in the I 2 C Multi-Master Environment Использование модуля SSP в режиме ведущего I^2C с конкуренцией на шине	AN578
Using Microchip 93 Series Serial EEPROMs with Microcontroller SPI Ports Использование интерфейса SPI для связи с последовательной памятью EEPROM серии 93	AN613
Software Implementation of I 2 C Bus Master Программная реализация ведущего шины I^2C	AN554
Interfacing PIC16C64/74 to Microchip SPI Serial EEPROM Подключение к PIC16C64/74 последовательной EEPROM памяти с интерфейсом SPI	AN647
Interfacing a Microchip PIC16C92x to Microchip SPI Serial EEPROM Подключение к PIC16C92x последовательной EEPROM памяти с интерфейсом SPI	AN668

Раздел 17. Модуль MSSP

Содержание

17.1 Введение

Модуль ведущего синхронного последовательного порта (MSSP) может использоваться для связи с периферийными микросхемами или другими микроконтроллерами. Периферийными микросхемами могут быть: EEPROM память, сдвиговые регистры, драйверы ЖКИ, АЦП и др. Модуль MSSP может работать в одном из двух режимах:

- Последовательный периферийный интерфейс (SPI);
- Inter-Integrated Circuit (I^2C):
  - ведущий режим;
  - ведомой режим (с поддержкой адреса общего вызова).

На рисунке 17-1 показана структурная схема модуля MSSP в режиме SPI, а на рисунках 17-2, 17-3 в двух разных режимах I^2C.

Рис. 17-1 Структурная схема модуля MSSP в SPI режиме

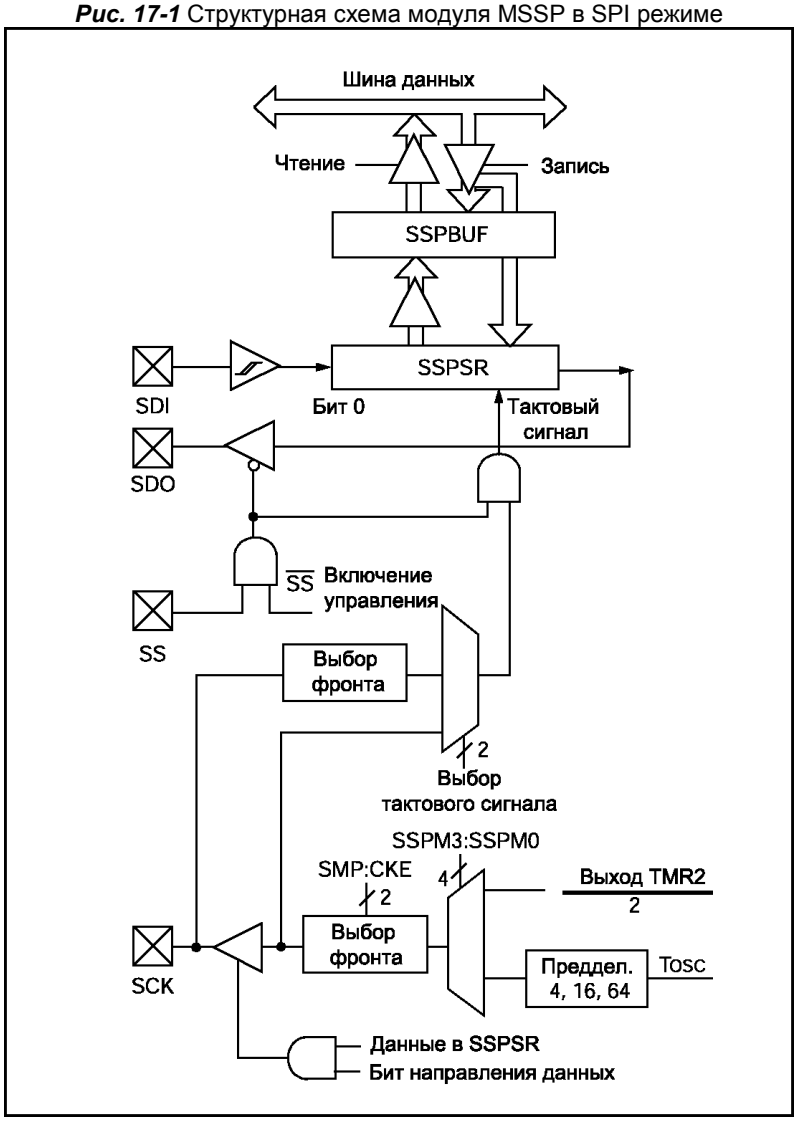


Рис. 17-2 Структурная схема модуля MSSP в режиме ведомого I^2C

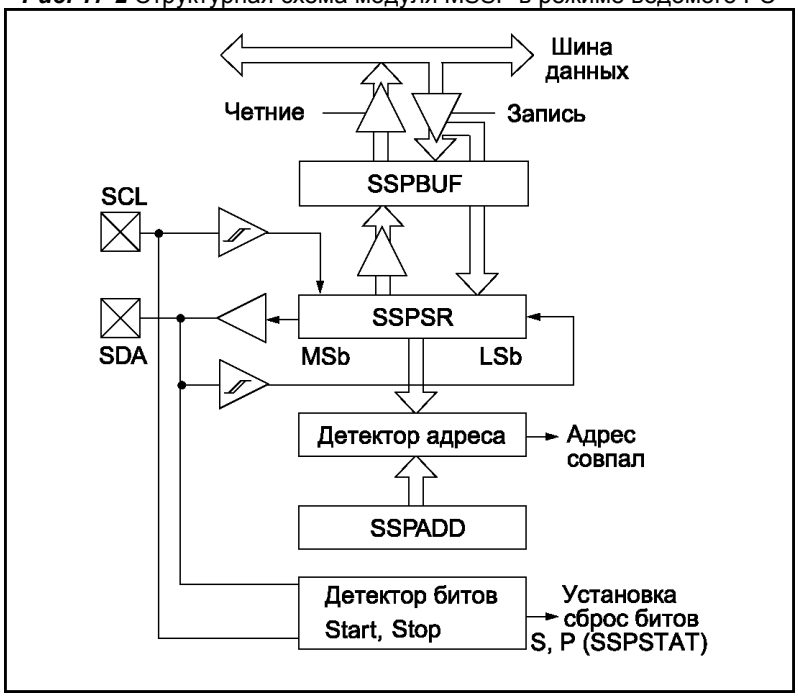
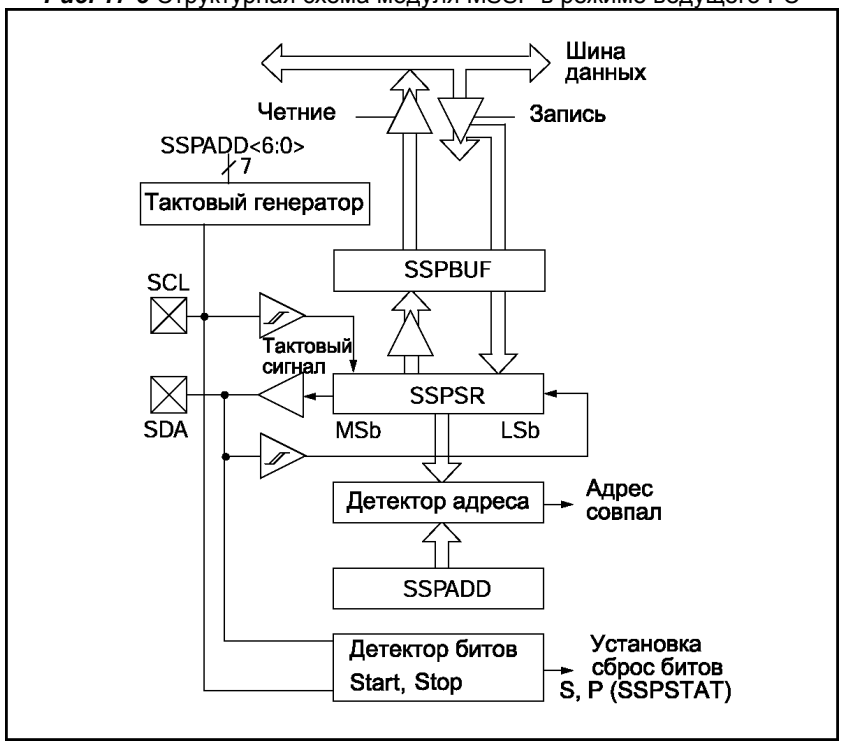


Рис. 17-3 Структурная схема модуля MSSP в режиме ведущего I^2C



17

17.2 Управляющие регистры

SSPSTAT: Регистр статуса модуля MSSP

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	**CKE**	**D/-A**	**P**	**S**	**R/-W**	**UA**	**BF**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **SMP:** Фаза выборки бита
Ведущий режим SPI
1 = опрос входа в конце периода вывода данных
0 = опрос входа в середине периода вывода данных

Ведомый режим SPI
Для режима ведомого SPI этот бит всегда должен быть сброшен в '0'

Ведущий или ведомый режим I^2C
1 = управление длительностью фронта выключено в стандартном режиме (100кГц и 1МГц)
0 = управление длительностью фронта включено в скоростном режиме (400кГц)

бит 6: **CKE:** Выбор фронта тактового сигнала
SPI режим, CKP=0
1 = данные передаются по переднему фронту сигнала на выводе SCK
0 = данные передаются по заднему фронту сигнала на выводе SCK

SPI режим, CKP=1
1 = данные передаются по заднему фронту сигнала на выводе SCK
0 = данные передаются по переднему фронту сигнала на выводе SCK

Ведущий или ведомый режим I^2C
1= входные уровни соответствуют спецификации SMBus
0= входные уровни соответствуют спецификации I^2C

бит 5: **D/-A:** Бит Данные/Адрес (только для режима I^2C)
1 = последний принятый или переданный байт является информационным
0 = последний принятый или переданный байт является адресным

бит 4: **P:** Бит STOP (только для режима I^2C)
Этот бит сбрасывается в '0' когда модуль MSSP выключен, SSPEN=0.
1 = указывает, что бит STOP был обнаружен последним (этот бит равен '0' после сброса)
0 = бит STOP не является последним

бит 3: **S:** Бит START (только для режима I^2C)
Этот бит сбрасывается в '0' когда модуль MSSP выключен, SSPEN=0.
1 = указывает, что бит START был обнаружен последним (этот бит равен '0' после сброса)
0 = бит START не является последним

бит 2: **R/-W:** Бит чтения/записи (только для режима I^2C)
Значение бита действительно только после совпадения адреса и до приема бита START, STOP или -ACK.
Ведомый режим I^2C
1 = чтение
0 = запись

Ведущий режим I^2C
1 = выполняется передача данных
0 = передачи данных не происходит
Логическое ИЛИ этого бита с битами SEN, RSEN, PEN, RCEN или ACKEN укажет на неактивное состояние модуля MSSP.

бит 1: **UA:** Флаг обновления адреса устройства (только для режима 10-разрядного I^2C)
1 = необходимо обновить адрес в регистре SSPADD
0 = обновление адреса не требуется

бит 0: **BF:** Бит статуса буфера
Прием (SPI и I^2C режимы)
1 = прием завершен, буфер SSPBUF полон
0 = прием не завершен, буфер SSPBUF пуст

Передача (только I^2C режима)
1 = выполняется передача данных (исключая биты -ACK и STOP), буфер SSPBUF полон
0 = передача данных завершена (исключая биты -ACK и STOP), буфер SSPBUF пуст

SSPCON: Регистр управления модуля MSSP

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	**SSPOV**	**SSPEN**	**CKP**	**SSPM3**	**SSPM2**	**SSPM1**	**SSPM0**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **WCOL:** Бит конфликта записи

Ведущий режим

1 = запись в SSPBUF была выполнена при не выполнении условий шины I^2C

0 = конфликта не было

Ведомый режим

1 = была предпринята попытка записи в SSPBUF во время передачи предыдущего байта

0 = конфликта не было

бит 6: **SSPOV:** Бит переполнения приемника

SPI режим

1 = принят новый байт, а SSPBUF содержит предыдущие данные(байт в SSPSR будет потерян). В ведомом режиме пользователь должен прочитать содержимое регистра SSPBUF даже, если только передает данные. В ведущем режиме бит в '1' не устанавливается, т.к. каждая операция инициализируется записью в SSPBUF. (сбрасывается в '0' программно)

0 = нет переполнения

I^2C режим

1 = принят новый байт, а SSPBUF содержит предыдущие данные. Значение бита не действительно при передаче данных. (сбрасывается в '0' программно)

0 = нет переполнения

бит 5: **SSPEN:** Бит включения модуля MSSP

Когда модуль включен, соответствующие порты ввода/вывода настраиваются на выход или вход

SPI режим

1 = модуль MSSP включен, выводы SCK, SDO, SDI, -SS используются модулем MSSP

0 = модуль MSSP выключен, выводы работают как цифровые порты ввода/вывода

I^2C режим

1 = модуль MSSP включен, выводы SDA, SCL используются модулем MSSP

0 = модуль MSSP выключен, выводы работают как цифровые порты ввода/вывода

бит 4: **CKP:** Бит выбора полярности тактового сигнала

SPI режим

1 = пассивный высокий уровень сигнала

0 = пассивный низкий уровень сигнала

Ведомый режим I^2C

Управление тактовым сигналом SCK

1 = не управлять тактовым сигналом

0 = удерживать тактовый сигнал в низком логическом уровне (используется для подготовки данных)

Ведущий режим I^2C

Не имеет значения

биты 3-0: **SSPM3:SSPM0:** Режим работы модуля MSSP

0000 = ведущий режим SPI, тактовый сигнал = $F_{OSC}/4$

0001 = ведущий режим SPI, тактовый сигнал = $F_{OSC}/16$

0010 = ведущий режим SPI, тактовый сигнал = $F_{OSC}/64$

0011 = ведущий режим SPI, тактовый сигнал = выход TMR2 / 2

0100 = ведомый режим SPI, тактовый сигнал с вывода SCK. Вывод -SS подключен к MSSP

0101 = ведомый режим SPI, тактовый сигнал с вывода SCK. Вывод -SS не подключен к MSSP

0110 = ведомый режим I^2C, 7-разрядная адресация

0111 = ведомый режим I^2C, 10-разрядная адресация

1000 = ведущий режим I^2C, тактовый сигнал = $F_{OSC}/(4 * (SSPADD+1))$

1011 = программная поддержка ведущего режима I^2C (ведомый режим выключен)

1110 = программная поддержка ведущего режима I^2C, 7-разрядная адресация с разрешением прерываний по приему бит START и STOP

1111 = программная поддержка ведущего режима I^2C, 10- разрядная адресация с разрешением прерываний по приему бит START и STOP

1001, 1010, 1100, 1101 = резерв

SSPCON2: Регистр управления модуля MSSP

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GCEN	**ACKSTAT**	**ACKDT**	**ACKEN**	**RCEN**	**PEN**	**RSEN**	**SEN**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **GCEN:** Бит разрешения поддержки общего вызова (только для ведомого режима I^2C)
1 = разрешить прерывания при приеме в регистр SSPSR адреса общего вызова (0000h)
0 = поддержка общего вызова выключена

бит 6: **ACKSTAT:** Бит статуса подтверждения (только для ведущего режима I^2C)
Передача ведущего I^2C
1 = подтверждения не было получено от ведомого
0 = подтверждение от ведомого было получено

бит 5: **ACKDT:** Бит подтверждения (только для ведущего режима I^2C)
Прием ведущего I^2C
Значение этого бита передается при разрешении формирование бита подтверждения.
1 = подтверждение
0 = нет подтверждения

бит 4: **ACKEN:** Сформировать бит подтверждения (только для ведущего режима I^2C)
1 = на выводах SCL, SDA формируется бит ACKDT. Аппаратно сбрасывается в '0'
0 = подтверждение не формируется

бит 3: **RCEN:** Разрешить прием данных (только для ведущего режима I^2C)
1 = разрешить прием данных с шины I^2C
0 = приемник выключен

бит 2: **PEN:** Сформировать бит STOP (только для ведущего режима I^2C)
1 = на выводах SCL, SDA формируется бит STOP. Аппаратно сбрасывается в '0'
0 = бит STOP не формируется

бит 1: **RSEN:** Сформировать бит повторный START (только для ведущего режима I^2C)
1 = на выводах SCL, SDA формируется бит повторный START. Аппаратно сбрасывается в '0'
0 = бит повторный STAT не формируется

бит 0: **SEN:** Сформировать бит START (только для ведущего режима I^2C)
1 = на выводах SCL, SDA формируется бит START. Аппаратно сбрасывается в '0'
0 = бит START не формируется

Примечание. Для битов ACKEN, RCEN, PEN, RSEN, SEN. Если I^2C модуль не находится в пассивном состоянии, то ни один из битов не может быть установлен в '1' (поставлен в очередь), не может быть выполнена запись в регистр SSPBUF (или запись в регистр SSPBUF заблокирована).

17.3 Режим SPI

В SPI режиме возможен одновременный синхронный прием и передача 8-разрядных данных. Модуль SSP поддерживает четыре режима SPI с типовым использованием трех выводов микроконтроллера:

- Вход последовательных данных (SDI);
- Выход последовательных данных (SDO);
- Тактовый сигнал (SCK).

Дополнительно может быть задействован четвертый вывод для работы в режиме ведомого:

- Выбор ведомого (-SS).

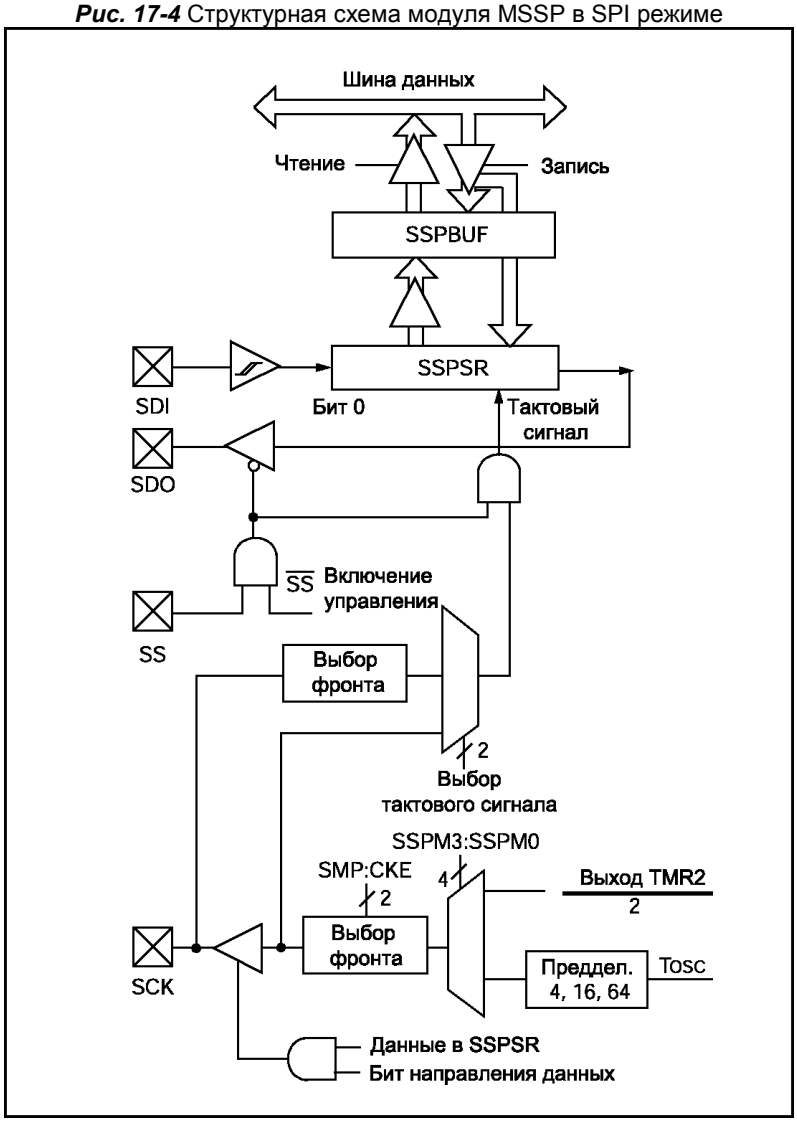
17.3.1 Работа модуля MSSP в режиме SPI

При инициализации SPI необходимо определить параметры работы модуля SPI битами SSPCON<5:0>, SSPSTAT<7:6>. Управляющие биты определяют следующие параметры работы:

- Ведущий режим (SCK выход);
- Ведомый режим (SCK вход);
- Полярность тактового сигнала (пассивный уровень SCK);
- Фаза выборки входных данных;
- Активный фронт тактового сигнала (передний, задний);
- Частота тактового сигнала (только в ведущем режиме);
- Режим выбора ведомого (только в режиме ведомого).

На рисунке 17-4 показана структурная схема модуля MSSP в SPI режиме.

Рис. 17-4 Структурная схема модуля MSSP в SPI режиме



Модуль MSSP состоит из приемного/передающего регистра сдвига (SSPSR) и буферного регистра (SSBUF). В регистре SSPSR выполняется сдвиг данных из/в микроконтроллер старшим битом вперед. В регистре SSPBUF сохраняются записанные данные, пока не будут получены новые. Приняв 8 бит данных в регистр SSPSR они переписываются в SSPBUF, устанавливается в '1' флаг полного приемного буфера BF (SSPSTAT<0>) и флаг прерывания SSPIF. Двойная буферизация принимаемых данных позволяет принимать следующий байт до чтения предыдущего. Любая запись в регистр SSPBUF во время выполнения операции приема/передачи данных будет игнорирована, при этом устанавливается в '1' флаг WCOL (SSPCON<7>). Пользователь должен программно сбросить бит WCOL в '0', чтобы была возможность проверки выполнения записи в регистр SSPBUF. При приеме данных в режиме SPI регистр SSPBUF должен быть прочитан до момента окончания приема следующего байта. Бит статуса приемного буфера BF (SSPSTAT<0>) указывает на получение нового байта данных. Бит BF аппаратно сбрасывается в '0' при чтении регистра SSPBUF. Принятые данные могут быть недостоверными, если режим SPI используется только для передачи данных. Прерывания от модуля MSSP используются для определения завершения приема/передачи данных (в подпрограмме обработки прерываний необходимо прочитать/записать регистр SSPBUF). Если не планируется использовать прерывания от модуля MSSP, то необходимо предусмотреть программную проверку выполнения записи в регистр SSPBUF для передачи данных. В примере 17-1 показана загрузка данных в регистр SSPBUF (SSPSR) для передачи данных. Затененная команда требуется только, если принимаемые данные имеют какое-то значение (в некоторых приложениях модуль MSSP в режиме SPI используется только для передачи данных).

Пример 17-1 Загрузка данных в регистр SSPBUF(SSPSR)

```
        BCF    STATUS, RP1   ;Банк 1
        BSF    STATUS, RP0   ;
LOOP    BTFSS  SSPSTAT, BF   ;Данные приняты?
        GOTO   LOOP          ;Нет
        BCF    STATUS, RP0   ;Банк 0
        MOVF   SSPBUF, W     ;Загрузить в W значение из SSPBUF
        MOVWF  RXDATA        ;Если необходимо, сохранить значение в памяти
        MOVF   TXDATA, W     ;Загрузить в W значение из TXDATA
        MOVWF  SSPBUF        ;Передать новые данные
```

Регистр SSPSR не доступен для непосредственного чтения или записи, все операции выполняются через регистр SSPBUF. В регистре SSPSTAT находятся биты, указывающие текущее состояние модуля MSSP.

17.3.2 Настройка выводов в режиме SPI

Для включения модуля MSSP необходимо установить бит SSPEN (SSPCON<5>) в '1'. Для сброса или перенастройки режима SPI рекомендуется сбросить бит SSPEN в '0', выполнить изменения параметров работы, а затем вновь установить бит SSPEN в '1'. После включения MSSP в режиме SPI выводы SDI, SDO, SCK, -SS используются последовательным портом. Для корректной работы последовательного порта биты регистров TRIS должны быть настроены следующем образом:

- SDI, бит TRIS должен быть установлен в '1';
- SDO, бит TRIS должен быть сброшен в '0';
- SCK (ведущий режим), бит TRIS должен быть сброшен в '0';
- SCK (ведомый режим), бит TRIS должен быть установлен в '1';
- -SS, бит TRIS должен быть установлен в '1'.

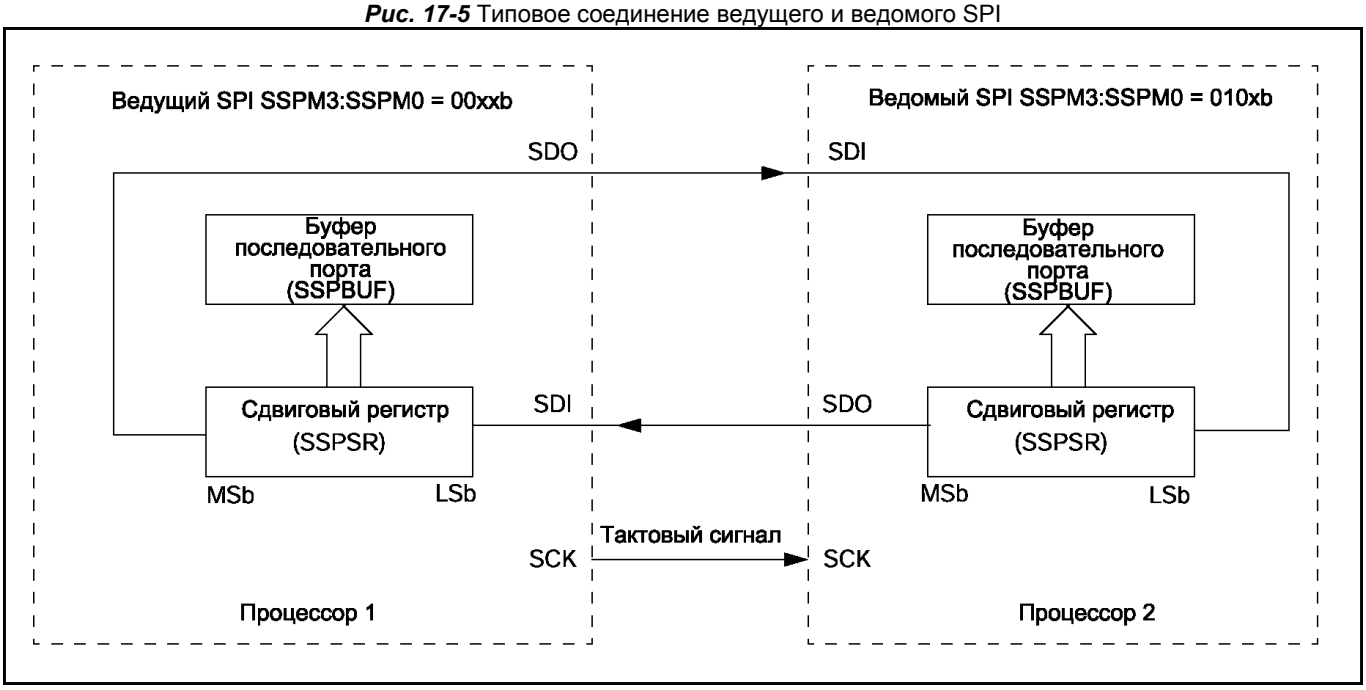
Любая нежелательная функция последовательного порта может быть выключена, настраивая соответствующие биты регистров направления данных TRIS. Например, если в режиме ведущего SPI выполняется только передача данных, то выводы SDI и -SS могут использоваться как цифровые выходы, сбросив соответствующие биты TRIS в '0'.

17.3.3 Типовое включение

На рисунке 17-5 показано типовое соединение двух микроконтроллеров. Главный микроконтроллер (процессор 1) инициализирует передачу, формируя тактовый сигнал SCK. Данные сдвигаются по установленному битом SMP фронту тактового сигнала. Для одновременного приема/передачи данных (фиктивных данных) оба микроконтроллера должны иметь одинаковую полярность тактового сигнала (бит CKP). Всего существует три сценария передачи данных:

- Ведущий передает данные - ведомый передает фиктивные данные;
- Ведущий передает данные - ведомый передает данные;
- Ведущий передает фиктивные данные - ведомый передает данные.

Рис. 17-5 Типовое соединение ведущего и ведомого SPI



17.3.4 Режим ведущего SPI

Ведущий шины может инициализировать передачу данных в любой момент, поскольку он генерирует тактовый сигнал, и определяет, когда ведомый (процессор 2) должен передать данные в соответствии с используемым протоколом.

В режиме ведущего данные передаются/приняты после их записи/чтения из регистра SSPBUF. Если в SPI режиме требуется только принимать данные, вывод SDO может быть заблокирован (настроен как вход). Данные с вывода SDI последовательно сдвигаются в регистр SSPSR с установленной скоростью. Каждый принятый байт загружается в регистр SSPBUF (как нормально полученный байт) с формированием прерываний и воздействием на соответствующие биты статуса. Эта функция может быть полезна при реализации "монитора шины".

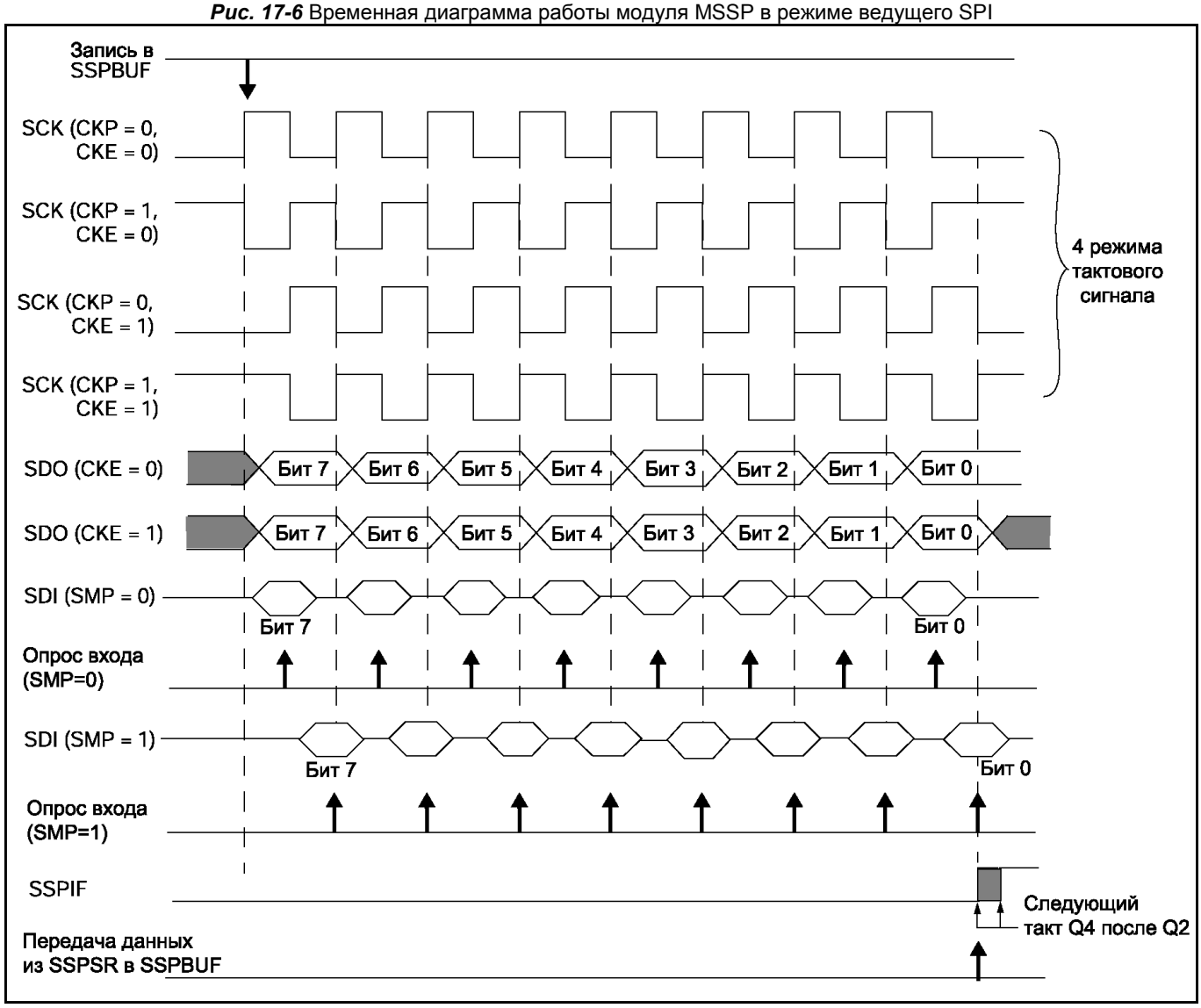
Полярность тактового сигнала устанавливается битом CKP (SSPCON<4>), что позволяет получить различные методы передачи данных (см. рисунки 17-6, 17-8 и 17-9). Данные всегда передаются старшим битом вперед. В ведущем режиме частота тактового сигнала выбирается программно:

- $F_{OSC}/4$ (или T_{CY});
- $F_{OSC}/16$ (или 4 x T_{CY});
- $F_{OSC}/64$ (или 16 x T_{CY});
- Выход таймера TMR2 / 2.

Максимальная частота передачи данных 5МГц при тактовой частоте микроконтроллера 20МГц.

Временная диаграмма передачи данных в режиме ведущего SPI показана на рисунке 17-6. Бит CKE определяет по какому фронту тактового сигнала необходимо выполнять прием данных. Параметры выборки входных данных устанавливаются битом SMP. Поле загрузки принятых данных в регистр SSPBUF устанавливается флаг прерываний SSPIF в '1'.

Рис. 17-6 Временная диаграмма работы модуля MSSP в режиме ведущего SPI



17.3.5 Режим ведомого SPI

В режиме ведомого данные передаются/принимаются по внешнему тактовому сигналу на выводе SCK. Когда принимается последний бит байта, устанавливается в '1' флаг прерываний SSPIF.

Полярность тактового сигнала выбирается битом CKP (SSPCON<4>). Внешний тактовый сигнал должен удовлетворять требованиям длительности низкого и высокого логического уровня, описанным в разделе электрических характеристик.

В SLEEP режиме микроконтроллера ведомый может принимать/передавать данные. После приема данных микроконтроллер выходит из режима SLEEP, если разрешены прерывания от модуля MSSP.

17.3.6 Выбор ведомого в режиме SPI

В режиме SPI вывод -SS позволяет подключать несколько ведомых к одному ведущему. Модуль MSSP должен находиться в режиме ведомого SPI (SSPCON<3:0> = 0100), бит TRIS для вывода -SS установлен в '1', чтобы позволить ведущему выбирать ведомого. Когда на выводе -SS присутствует низкий логический уровень, передача и прием данных разрешены, а вывод SDO управляется модулем SSP. Если на выводе -SS высокий уровень сигнала, то вывод SDO переходит в 3-е состояние. В зависимости от приложения может потребоваться внешний подтягивающий резистор на выводе SDO.

Примечание 1. В режиме ведомого SPI с поддержкой выбора ведомого по сигналу на выводе -SS (SSPCON<3:0>=0100), SPI модуль сброшен, если на выводе -SS напряжение питания V_{DD}.

Примечание 2. В режиме ведомого SPI и CKE = 1, необходимо разрешить управление с вывода -SS.

При сбросе модуля SSP в режиме SPI счетчик битов сдвигового регистра очищается. Сброс модуля в режиме SPI происходит при появлении высокого логического уровня на выводе -SS и сбросе в '0' бита SSPEN.

Для реализации двух проводного интерфейса вывод SDO может быть соединен с SDI. Когда SPI должен работать как приемник, вывод SDO настраивается на вход, что отключает передатчик от SDO. SDI всегда должен быть настроен как вход (функция SDI), т.к. это не создает конфликт шины.

Рис. 17-7 Временная диаграмма синхронизации ведомого

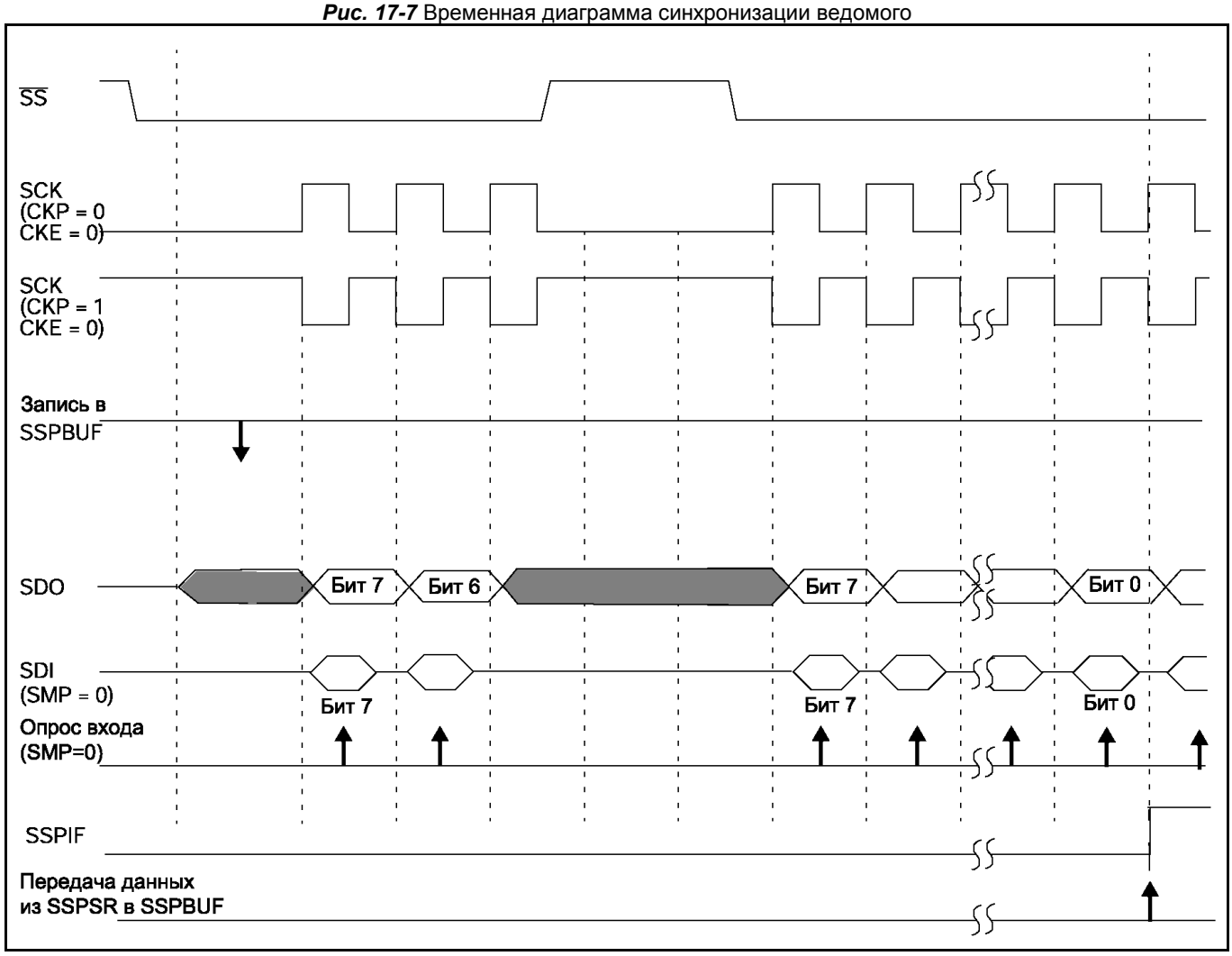


Рис. 17-8 Временная диаграмма работы модуля MSSP в режиме ведомого SPI (CKE=0)

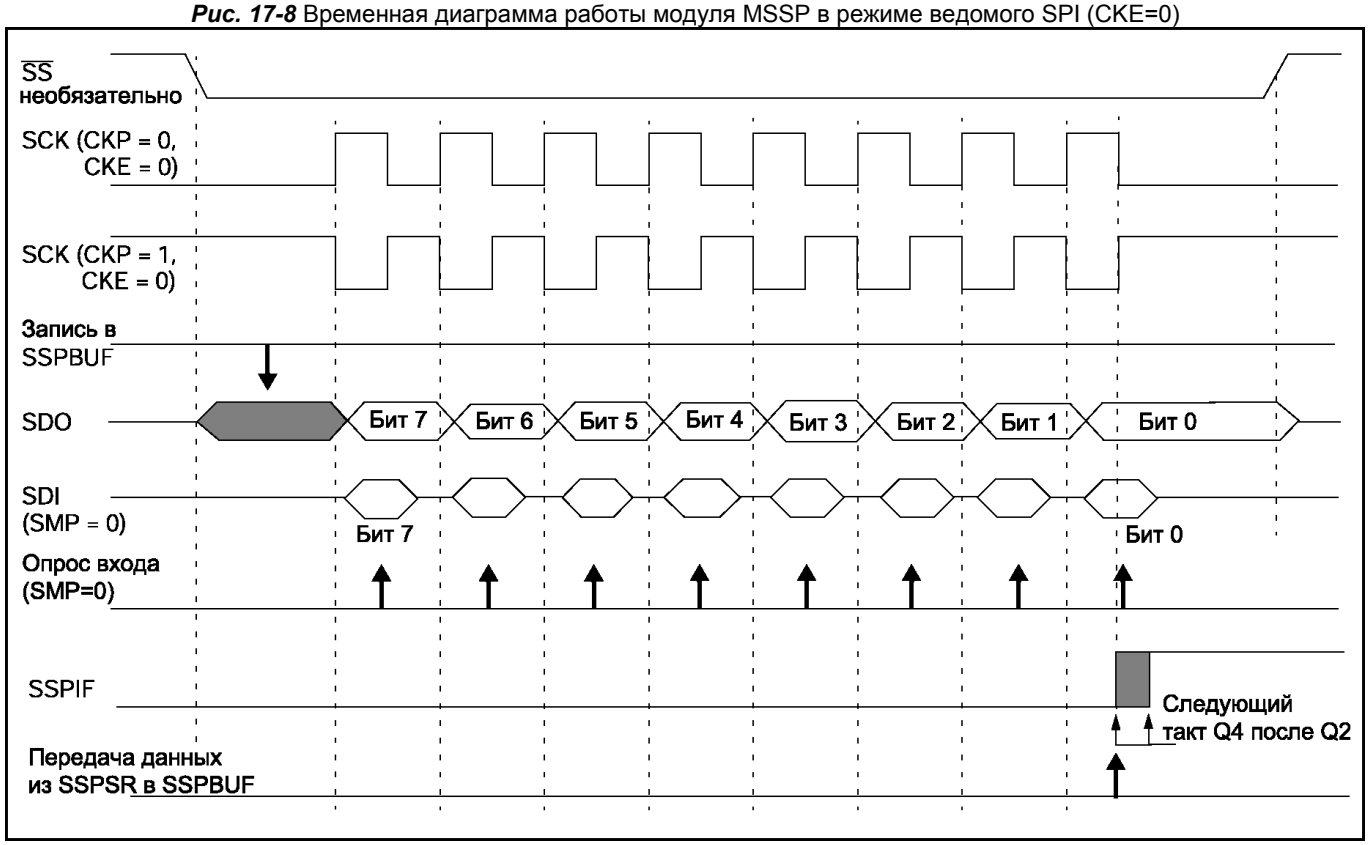
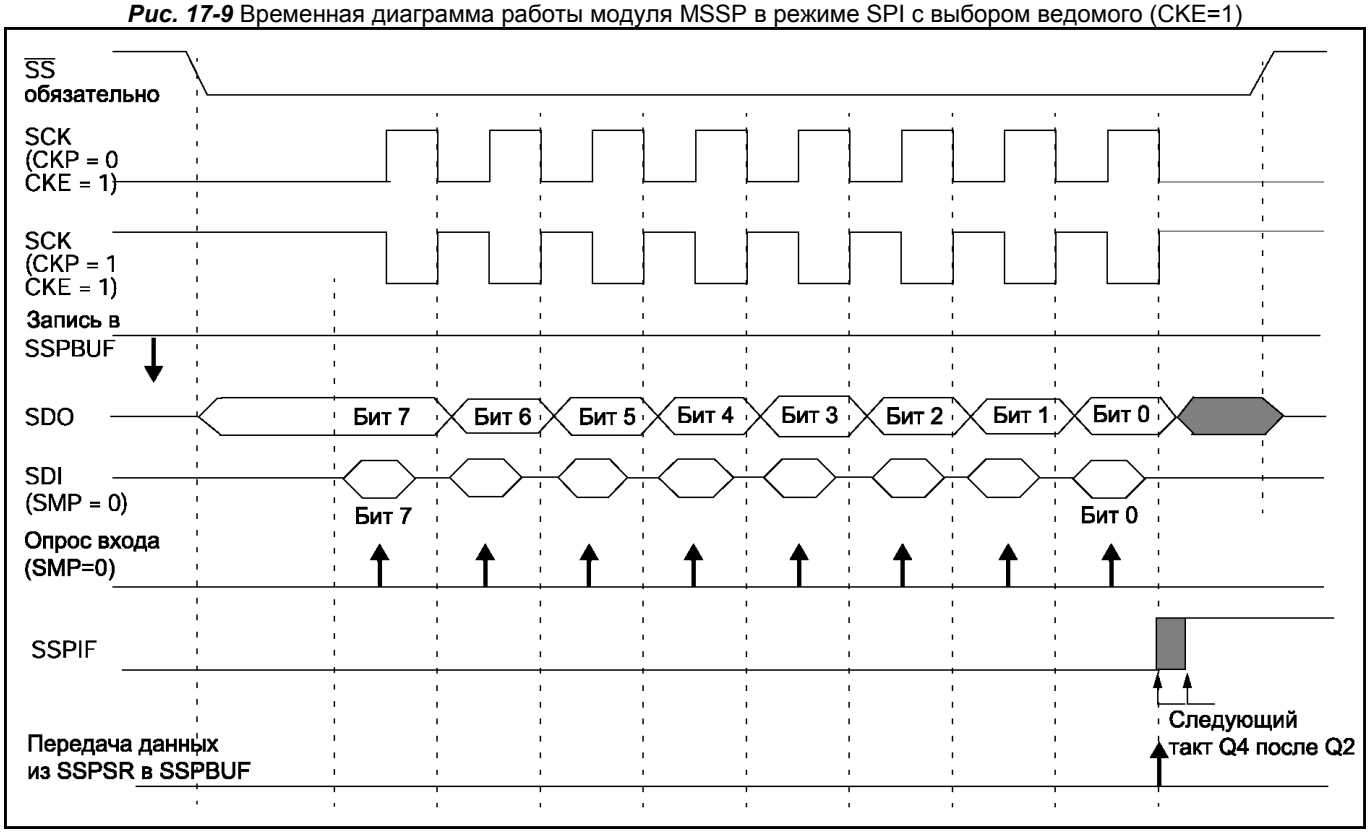


Рис. 17-9 Временная диаграмма работы модуля MSSP в режиме SPI с выбором ведомого (CKE=1)



17

17.3.7 Работа в SLEEP режиме микроконтроллера

В режиме ведущего SPI тактовый сигнал модуля MSSP отсутствует, состояние приема/передачи данных не изменяется до выхода микроконтроллера из режима SLEEP. После выхода микроконтроллера из режима SLEEP модуль SSP продолжит передачу/прием данных.

В режиме ведомого SPI данные могут быть приняты/переданы, т.к. сдвиговый регистр работает асинхронно. Это позволяет в SLEEP режиме микроконтроллера принять/передать данные в/из сдвигового регистра. Как только будут приняты все 8 бит данных, устанавливается в '1' флаг прерывания от модуля MSSP, и если прерывания разрешены, микроконтроллер выйдет из SLEEP режима.

17.3.8 Эффект сброса

Любой сброс микроконтроллера выключает модуль MSSP, прием/передача данных прекращается.

Таблица 17-1 Регистры и биты, связанные с работой модуля SSP в режиме SPI

Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Сброс POR, BOR	Другие сбросы
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000u
PIR	SSPIF(1)								0	0
PIE	SSPIE(1)								0	0
SSPBUF	Буфер приемника SSP / регистр передатчика								xxxx xxxx	uuuu uuuu
SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
SSPSTAT	SMP	CKE	D/-A	P	S	R/-W	UA	BF	0000 0000	0000 0000

Обозначения: - = не используется, читается как 0; u = не изменяется; x = не известно; q = зависит от условий.
Затененные биты на работу не влияют.
Примечания:

1. Расположение битов смотрите в технической документации на микроконтроллер.
2. В некоторых микроконтроллерах эти биты могут обозначаться как GPIE и GPIF.

17.4 Режим I²C

Модуль MSSP полностью поддерживает все функции ведущих и ведомых устройств, включая поддержку общего вызова, аппаратные прерывания по детектированию битов START и STOP для определения занятости шины I²C в режиме ведущего (при конкуренции на шине). В MSSP модуле реализована поддержка стандартного режима 7, 10-разрядной адресации. Дополнительно смотрите приложение A, в котором дано краткое описание шины I²C.

Фильтр "glitch" подключен к выводам SDA и SCL, когда они настроены на вход. Фильтр работает в режимах 100кГц и 400кГц. В режиме 100кГц, когда выводы SDA и SCL настроены на выход, фильтр контролирует длительность формируемых сигналов независимо от тактовой частоты микроконтроллера.

Рис. 17-10 Структурная схема модуля MSSP в режиме ведомого I²C

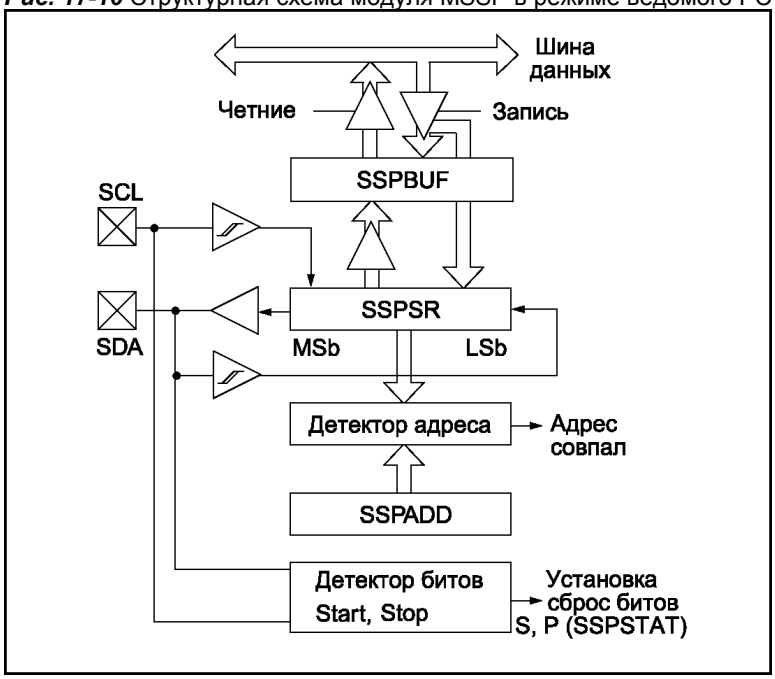
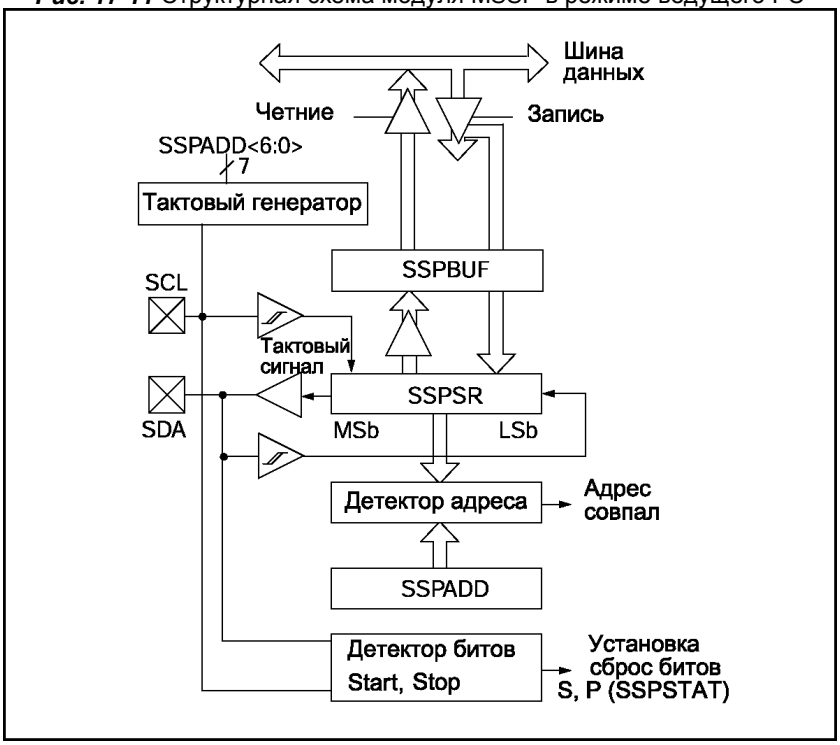


Рис. 17-11 Структурная схема модуля MSSP в режиме ведущего I²C



17

Для работы с шиной I^2C используется два вывода SCL (сигнал синхронизации) и SDA (данные). Выводы SDA и SCL автоматически настраиваются при включении режима I^2C. Включение модуля MSSP выполняется установкой бита SSPEN (SSPCO<5>) в '1'.

Для управления модулем MSSP в режиме I^2C используется шесть регистров:

- SSPCON, регистр управления MSSP;
- SSPCON, регистр управления 2 MSSP;
- SSPSTAT, регистр статуса MSSP;
- SSPBUF, буфер приемника/передатчика;
- SSPSR, сдвиговый регистр (пользователю не доступен);
- SSPADD, регистр адреса.

В регистре SSPCON устанавливается требуемый режим I^2C. С помощью четырех битов (SSPCON<3:0>) можно выбрать один из режимов I^2C:

- Ведомый режим I^2C, 7-разрядная адресация;
- Ведомый режим I^2C, 10-разрядная адресация;
- Ведущий режим I^2C, тактовый сигнал = $F_{OSC}/(4 * (SSPADD+1))$;
- Программная поддержка ведущего режима I^2C.

При выборе любого режима I^2C выводы SCL и SDA должны быть настроены на вход, установкой соответствующих битов регистра TRISC в '1'. После выбора режима I^2C и установки бита SSPEN в '1' выводы SDA (линия данных), SCL (линия синхронизации) подключаются к модулю MSSP.

Регистр SSPSTAT содержит биты статуса передачи данных: обнаружение на шине битов START (S) или STOP (P), флаг приема байта данных или адреса, указатель загрузки старшего байта 10-разрядного адреса, бит операции приема/передачи.

В регистр SSPBUF загружаются данные для передачи по шине I^2C, и из него читаются принятые данные. Регистр SSPSR выполняет сдвиг принимаемых/передаваемых данных. При приеме данных регистры SSPBUF, SSPSR работают как двухуровневый буфер приемника. Буфер позволяет принимать следующий байт до чтения предыдущего принятого байта из регистра SSPBUF. Когда байт полностью загружен в SSPSR, он передается в регистр SSPBUF и устанавливается флаг прерывания SSPIF в '1'. Если полностью принят следующий байт до чтения предыдущего байта из SSPBUF, то устанавливается бит SSPOV (SSPCON<6>) в '1', а байт в регистре SSPSR будет потерян.

В регистр SSPADD записывается адрес ведомого устройства. В 10-разрядном режиме пользователь должен сначала записывать старший байт адреса (1111 0 A9 A8 0). После соответствия старшего байта адреса необходимо загрузить младший байт адреса (A7:A0).

17.4.1 Режим ведомого I^2C

В режиме ведомого I^2C выводы SCL, SDA должны быть настроены на вход. Модуль MSSP автоматически изменит направление вывода SDA при передаче данных ведомым.

При совпадении адреса или после приема байта данных (если предварительно совпал адрес) аппаратно генерируется бит подтверждения (-ACK), а затем данные из регистра SSPSR загружаются в SSPBUF.

Существует несколько условий, при которых бит -ACK не формируется (эти условия могут возникать одновременно):

a) Бит BF (SSPSTAT<0>) = 1 перед приемом данных;
b) Бит переполнения SSPOV (SSPSTAT<6>) = 1 перед приемом данных.

Если бит BF = 1, то значение из SSPSR не переписывается в регистр SSPBUF, а биты SSPIF и SSPOV устанавливаются в '1'. В таблице 17-2 показаны операции после приема байта при различных значениях битов BF, SSPOV. В затененных ячейках показана ситуация, когда вовремя не был сброшен бит переполнения SSPOV в '0'. Заметьте, что бит BF аппаратно сбрасывается в '0' при чтении из регистра SSPBUF, а бит SSPOV необходимо сбрасывать в '0' программно.

Минимальная длительность логических уровней входного сигнала синхронизации SCL должна удовлетворять требованиям раздела электрических характеристик (см. параметры 100 и 101).

17.4.1.1 Адресация

После включения модуля MSSP ожидается формирование на шине бита START. Получив бит START, принимается 8 бит в сдвиговый регистр SSPSR. Выборка битов происходит по переднему фронту синхронизирующего сигнала на выводе SCL. По заднему фронту восьмого такта сигнала SCL значение в регистре SSPSR<7:1> сравнивается с содержимым регистра SSPADD. Если значение адреса совпадает, а биты BF и SSPOV равны нулю, то выполняются следующие действия:

a) Значение регистра SSPSR загружается SSPBUF по 8-му заднему фронту сигнала SCL;
b) Устанавливается флаг BF в '1' (буфер полон) по 8-му заднему фронту сигнала SCL;
c) Генерируется бит -ACK;
d) Устанавливается флаг прерываний SSPIF в '1' (если разрешено, генерируется прерывание) по 9-му заднему фронту сигнала SCL.

В режиме ведомого при 10-разрядной адресации необходимо принять два байта адреса. Пять старших бит первого байта определяют: является ли полученный байт первым байтом 10-разрядного адреса. Бит R/-W(SSPSTAT<2>) должен быть настроен для приема второго байта адреса. Для 10-разрядной адресации первый байт адреса должен иметь формат '1111 0 A9 A8 0', где A9:A8 два старших бита адреса. Рекомендуемая последовательность действий при 10-разрядной адресации (шаги 7-9 для передачи ведомым):

1. Принять старший байт адреса (устанавливаются биты SSPIF, BF и UA (SSPSTAT<1> в '1').
2. Записать младший байт адреса в регистр SSPADD (аппаратно сбрасывается бит UA в '0' и "отпускается" линия SCL).
3. Выполнить чтение из регистра SSPBUF (сбрасывается бит BF в '0') и сбросить флаг SSPIF в '0'.
4. Принять младший байт адреса (устанавливаются биты SSPIF, BF и UA (SSPSTAT<1> в '1').
5. Записать старший байт адреса в регистр SSPADD (аппаратно сбрасывается бит UA в '0' и "отпускается" линия SCL).
6. Выполнить чтение из регистра SSPBUF (сбрасывается бит BF в '0') и сбросить флаг SSPIF в '0'.
7. Принять бит повторный START.
8. Принять старший байт адреса (устанавливаются биты SSPIF и BF в '1').
9. Выполнить чтение из регистра SSPBUF (сбрасывается бит BF в '0') и сбросить флаг SSPIF в '0'.

Примечание. В 10-разрядном режиме после команды повторный START (шаг 7) не требуется обновлять значение в регистре SSPADD. В данном случае требуется соответствие только первого байта адреса.

Таблица 17-2 Операции после приема байта при различных значениях битов BF, SSPOV

Биты статуса приемника		Запись из SSPSR в SSPBUF	Формирование бита -ACK	Установка флага прерываний SSPIF
BF	**SSPOV**			
0	0	Есть	Есть	Есть
1	0	Нет	Нет	Есть
1	1	Нет	Нет	Есть
0	1	Есть	Нет	Есть

Примечание. В затененных ячейках показана ситуация, когда вовремя не был сброшен бит переполнения SSPOV в '0'.

17.4.1.2 Прием данных ведомым

Если бит R/-W в адресном байте равен нулю, а принятый адрес совпадает с адресом устройства, то бит R/-W в регистре SSPSTAT сбрасывается в '0'. Принятый адрес загружается в регистр SSPBUF.

Если бит BF (буфер полон) или SSPOV (переполнение буфера) установлен в '1', то бит подтверждения -ACK не формируется. Эту ошибку необходимо обработать программно. Если было выполнено чтение из регистра SSPBUF но не был сброшен бит SSPOV в '0', то бит -ACK не формируется.

Прерывание от модуля MSSP генерируются при каждом принятом байте с шины I^2C, установкой флага SSPIF в '1' (сбрасывается программно). Регистр SSPSTAT используется для определения типа принятого байта.

Примечание. Значение регистра SSPBUF будет обновлено, если SSPOV=1 BF=0. Если было выполнено чтение из регистра SSPBUF но не был сброшен бит SSPOV в '0', то бит -ACK не формируется.

Рис. 17-12 Временная диаграмма приема данных ведомым I^2C (7-разрядная адресация)

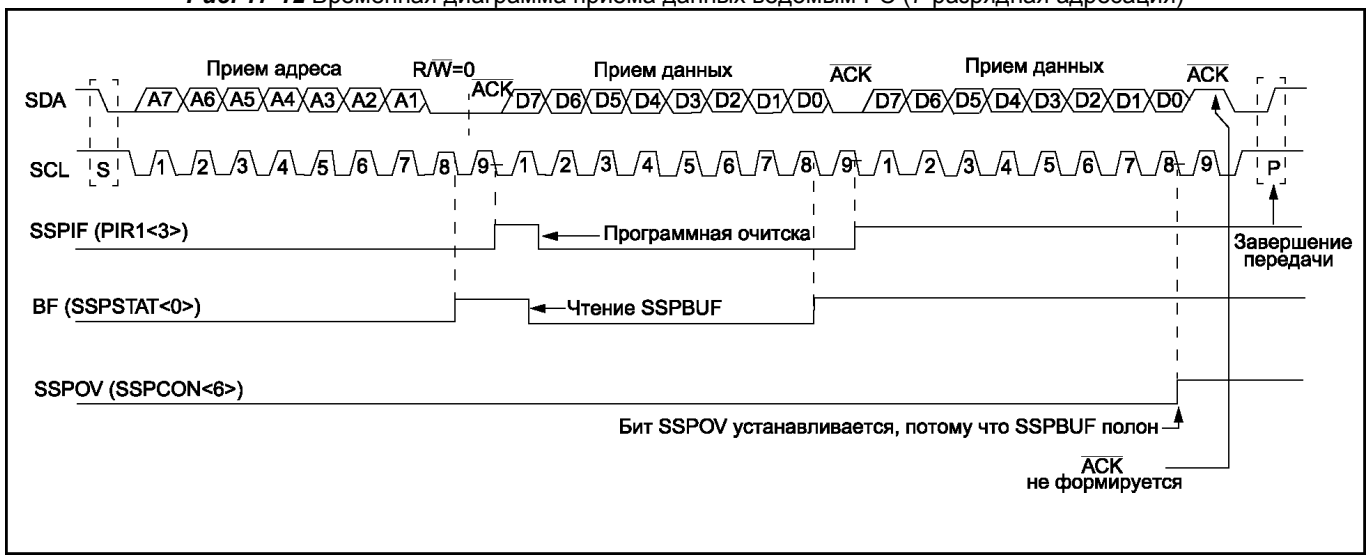
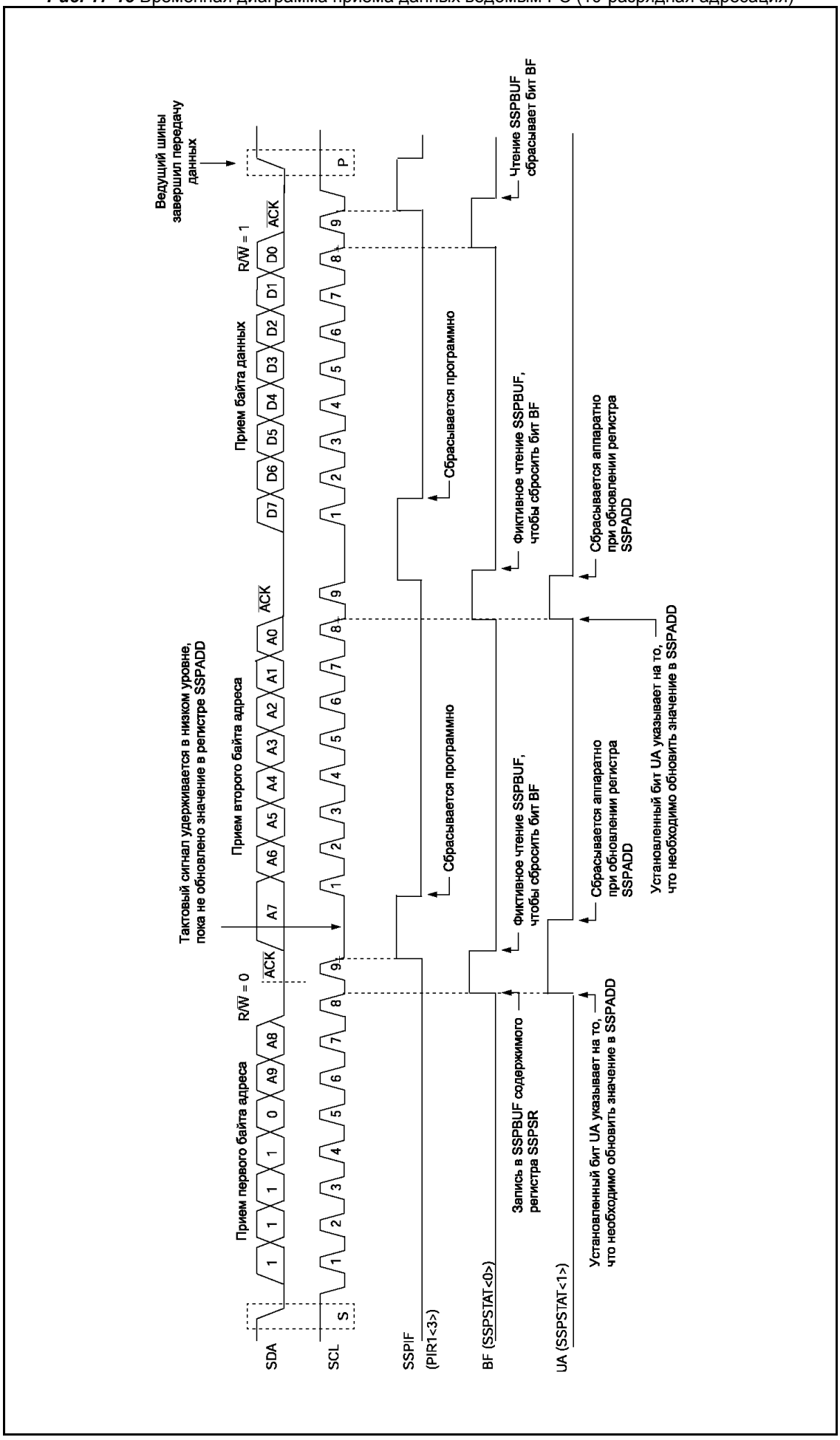


Рис. 17-13 Временная диаграмма приема данных ведомым I^2C (10-разрядная адресация)



15.4.1.3 Передача данных ведомым

Если бит R/-W в адресном байте равен '1', а принятый адрес совпадает с адресом устройства, то бит R/-W в регистре SSPSTAT устанавливается в '1'. Принятый адрес загружается в регистр SSPBUF. Бит -ACK формируется девятым битом, после чего линия SCL удерживается в низком логическом уровне. Передаваемые данные должны быть записаны в регистр SSPBUF, после чего они автоматически переписываются в регистр SSPSR. После записи данных необходимо "отпустить" сигнал SCL установкой бита CKP(SSPCON<4>) в '1'. Ведущий шины контролирует состояние линии SCL, ожидая смены уровня сигнала. Восемь бит загруженных данных последовательно сдвигаются по заднему фронту сигнала SCL, что гарантирует достоверное значение данных на линии SDA (см. рисунок 17-14).

Модуль MSSP генерирует прерывание по каждому переданному байту, устанавливая бит SPPIF в '1' по заднему фронту девятого такта сигнала SCL. Флаг SSPIF должен быть сброшен программно. Регистр SSPSTAT используется для определения статуса передачи данных.

Ведущее устройство формирует бит подтверждения -ACK на девятом такте сигнала SCL для каждого принятого байта. Если бит подтверждения -ACK не сформирован (высокий уровень сигнала SDA), передача данных завершена. Логика ведомого устройства настраивается на обнаружение бита START. Если бит подтверждения -ACK был получен (низкий уровень сигнала SDA), в регистр SSPBUF необходимо записать новый байт для передачи. Линию SCL также необходимо "отпустить", установкой бита CKP в '1'.

Рис. 17-14 Временная диаграмма передачи данных ведомым I^2C (7-разрядная адресация)

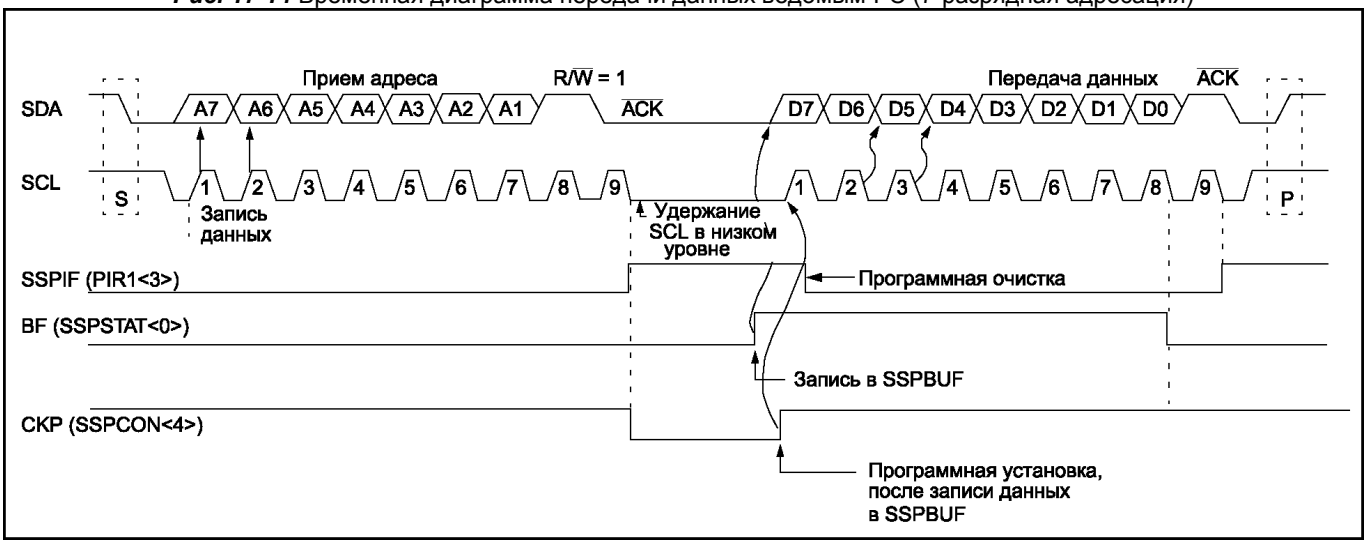
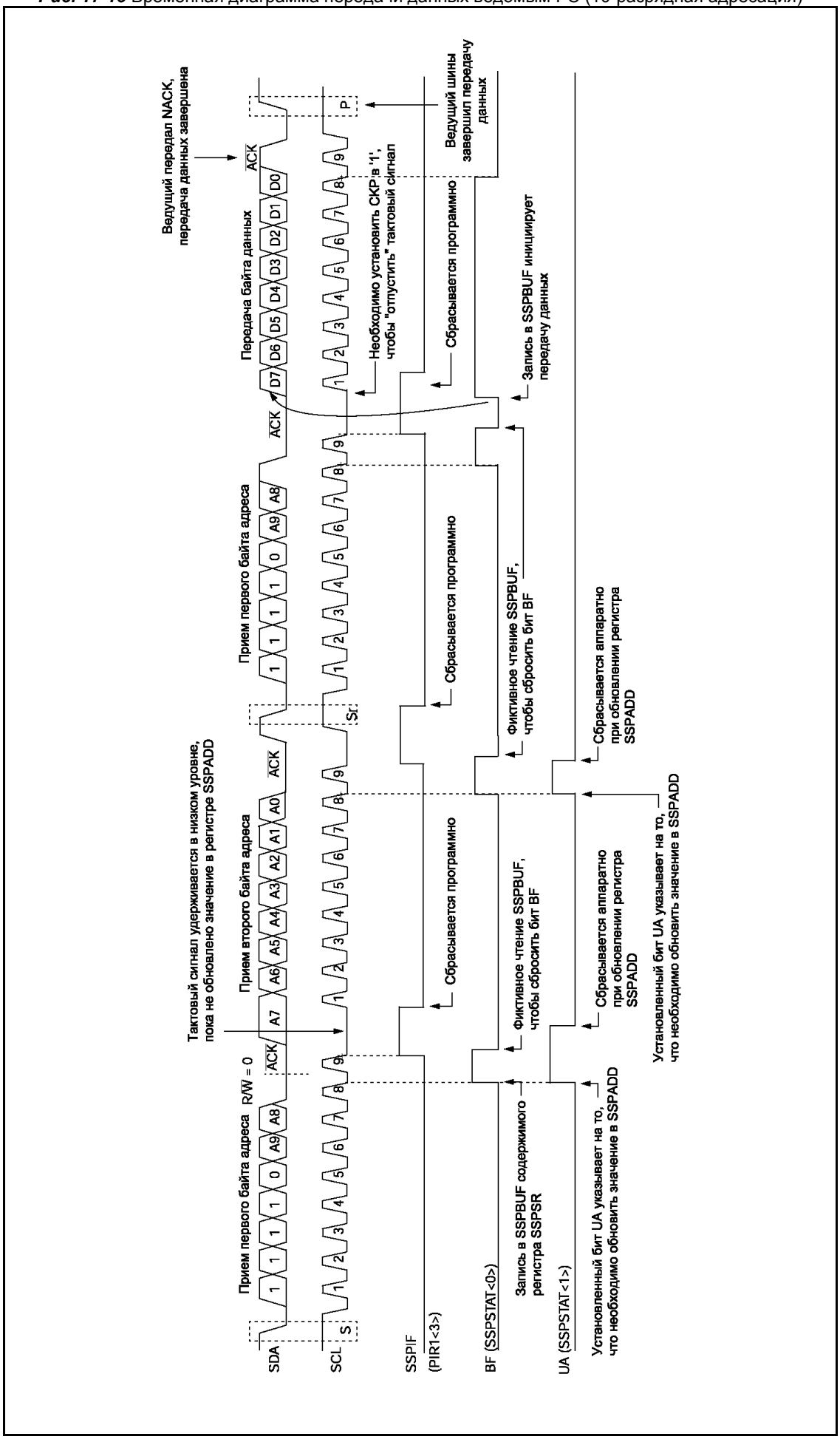


Рис. 17-15 Временная диаграмма передачи данных ведомым I^2C (10-разрядная адресация)



17.4.2 Поддержка общего вызова

Процедура адресации на шине I^2C такова, что первый после START байт определяет, к какому из ведомых устройств обращается ведущий шины. Исключением является адрес общего вызова, при использовании которого теоретически должны откликнуться все ведомые.

Адрес общего вызова – один из восьми зарезервированных адресов шины I^2C, все биты которого равны нулю (в том числе и бит R/-W).

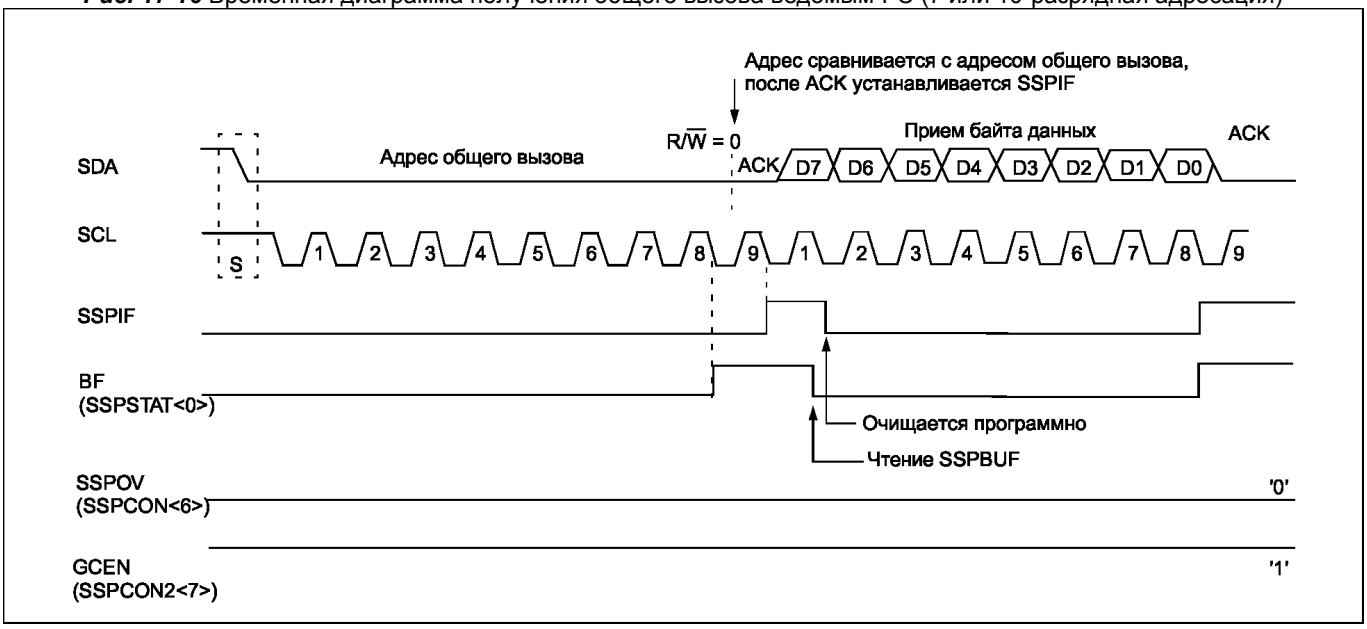
Распознавание адреса общего вызова включается установкой бита GCEN (SSPCON2<7>) в '1'. Следующий за START байт помещается в регистр SSPSR и сравнивается с содержимым SSPADD и фиксированным адресом общего вызова.

При получении адреса общего вызова, содержимое SSPSR переписывается в регистр SSPBUF (устанавливается бит BF в '1') по заднему фронту восьмого такта. На девятом такте формируется бит подтверждения (-ACK) и устанавливается флаг прерываний SSPIF в '1'.

Содержимое регистра SSPBUF позволяет определить получение общего вызова.

В 10-разрядном режиме требуется обновить содержимое регистра SSPADD для проверки соответствия младшего байта адреса после установки бита UA(SSPSTAT<1>) в '1'. Если получен адрес общего вызова в 10-разрядном режиме адресации при GCEN=1, то обновлять значение адреса не требуется. После формирование бита подтверждения ведущее устройство начнет принимать данные (см. рисунок 17-16).

Рис. 17-16 Временная диаграмма получения общего вызова ведомым I^2C (7 или 10-разрядная адресация)



17.4.3 Работа в SLEEP режиме

Ведомый I^2C может принимать адресные байты или байты данных в SLEEP режиме микроконтроллера. После приема байта микроконтроллер выходит из режима SLEEP, если разрешены прерывания от MSSP модуля.

17.4.4 Эффект сброса

При сбросе микроконтроллера модуль MSSP выключается, прекращается любой обмен данными.

Таблица 17-3 Регистры и биты, связанные с работой модуля MSSP в режиме I^2C

Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Сброс POR, BOR	Другие сбросы
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000u
PIR	SSPIF, BCLIF (1)								0, 0	0, 0
PIE	SSPIE, BCLIF (1)								0, 0	0, 0
SSPBUF	Буфер приемника MSSP / регистр передатчика								xxxx xxxx	uuuu uuuu
SSPADD	Регистр адреса MSSP (ведомый режим I^2C), регистр скорости обмена (ведущийрежим I^2C)								0000 0000	0000 0000
SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	0000 0000
SSPSTAT	SMP	CKE	D/-A	P	S	R/-W	UA	BF	0000 0000	0000 0000

Обозначения: - = не используется, читается как 0; u = не изменяется; x = не известно; q = зависит от условий.
Затененные биты на работу не влияют.
Примечания:

1. Расположение битов смотрите в технической документации на микроконтроллер.
2. В некоторых микроконтроллерах эти биты могут обозначаться как GPIE и GPIF.

17.4.5 Режим ведущего I²C

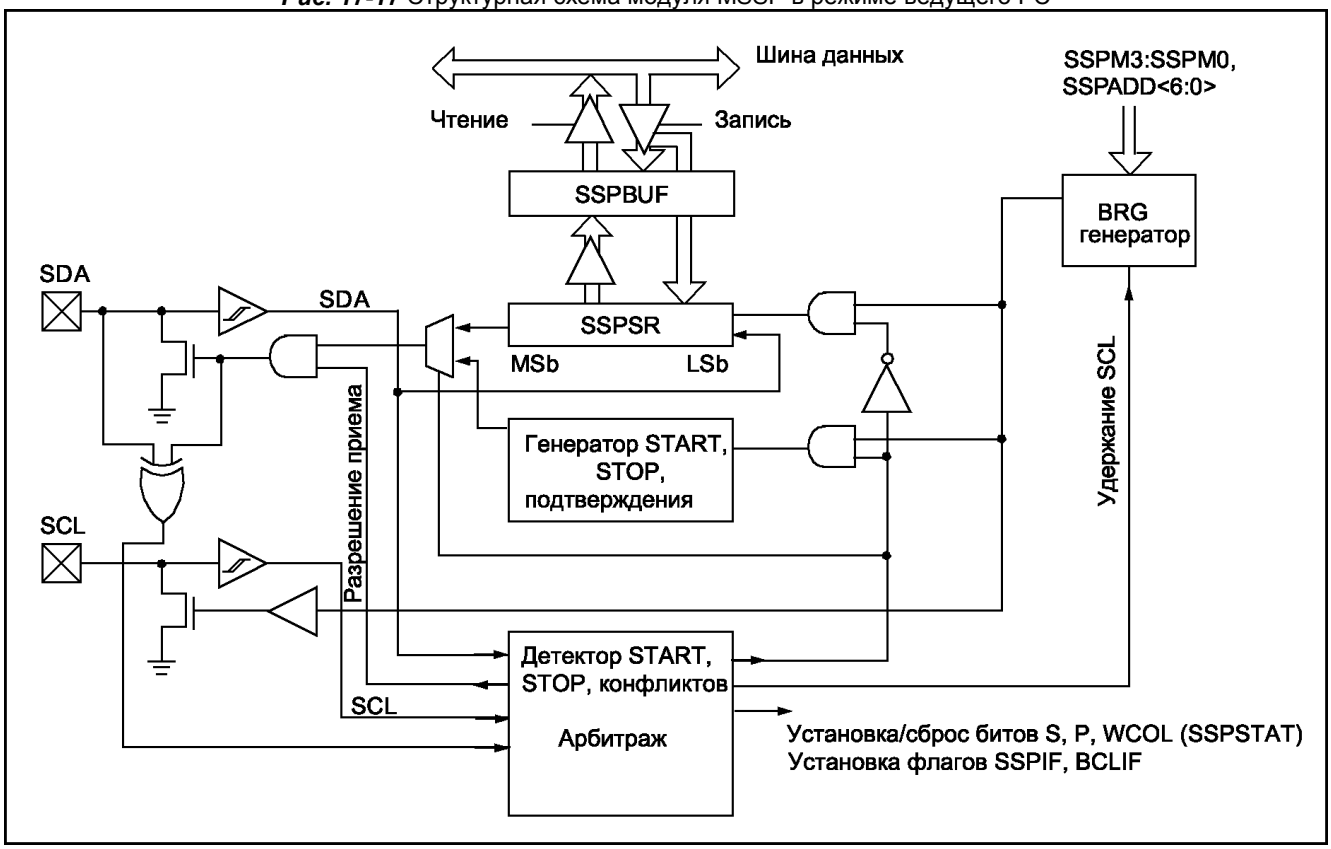
В режиме ведущего поддерживается генерация прерываний при обнаружении на шине битов START и STOP. Биты STOP (P) и START (S) в регистре SSPSTAT равны '0' после сброса микроконтроллера или при выключенном модуле MSSP. Шина находится в неактивном состоянии, если бит P=1 или оба бита S, P равны '0'.

В режиме ведущего выводы SCL, SDA управляются аппаратно.

Следующие события на шине I²C могут привести к установки флага прерываний SSPIF в '1':

- Выполнено условие START;
- Выполнено условие STOP;
- Передан/принят байт данных;
- Передан бит подтверждения;
- Выполнено условие повторный START.

Рис. 17-17 Структурная схема модуля MSSP в режиме ведущего I²C



17.4.6 Режим конкуренции

В режиме конкуренции, прерывания поле START и STOP позволяет определить, когда шина I²C свободна. Биты S и P сбрасываются в '0' при сбросе микроконтроллера или при выключении модуля MSSP. Управление шиной может быть перехвачено, когда бит P=1 или шина простаивает (S=0 и P=0). Если шина занята, можно разрешить прерывания от MSSP для обнаружения бита STOP на шине.

При конкуренции линия SDA должна проверяться на соответствия уровня, при ожидаемом высоком уровне на выходе. Эта проверка производится автоматически, а результат помещается в бит BCLIF.

Арбитраж на шине I²C может быть потерян во время:

- Передачи адреса;
- Передачи данных;
- Формирования бита START;
- Формирования бита повторный START;
- Формирования бита NACK.

17.4.7 Поддержка режима ведущего I^2C

Ведущий режим включается соответствующей настройкой битов SSPM в регистре SSPCON и установкой в '1' бита SSPEN. После включения ведущего режима аппаратно могут выполняться следующие функции:

- Формирование бита START на линии SCL и SDA;
- Формирование бита повторный START на линии SCL и SDA;
- Записью в регистр SSPBUF инициализируется передача байта данных/адреса;
- Формирование бита STOP на линии SCL и SDA;
- Настройка порта I^2C на прием данных;
- Формирование бита подтверждения ACK после приема байта на линии SCL и SDA.

Примечание. Модуль MSSP в ведущем режиме не имеет стека событий. Это означает, что пользователь не может к примеру инициировать передачу бита START и произвести запись в SSPBUF до того, как START будет завершен. При попытке осуществления подобной операции будет установлен бит WCOL в '1', указывая, что запись в регистр SSPBUF не произошла.

17.4.7.1 Работа в режиме ведущего I^2C

Ведущий формирует на шине I^2C тактовый сигнал и биты START, STOP. Текущий обмен данными завершается после формирования бита STOP или повторный START. Поскольку бит повторный START инициирует новый обмен данными, шина I^2C остается занятой.

Передатчик ведущего выдает данные на линию SDA, а тактовый сигнал на линию SCL. Первый передаваемый байт содержит 7-разрядный адрес приемника (при 7-разрядной адресации устройств) и бит направления данных R/-W=0. После каждого переданного 8-разрядного байта принимается бит подтверждения -ACK. Биты START и STOP формируется для указания начала и завершения передачи данных.

В режиме приема ведущем на шину I^2C сначала выдается байт, содержащий 7-разрядный адрес передатчика (при 7-разрядной адресации устройств) и бит направления данных R/-W = 1. Данные принимаются с линии SDA, а на линии SCL формирует тактовый сигнал. После каждого принятого байта формируется бит подтверждения. Биты START и STOP формируется для указания начала и завершения передачи данных.

Генератор скорости обмена BRG используется для установки требуемой частоты тактового сигнала на линии SCL – 100кГц, 400кГц или 1МГц. Значение для перезагрузки BRG берется из 7 младших бит регистра SSPADD. BRG начинает работу сразу после записи данных в регистр SSPBUF. Как только операция завершена (передан последний бит байта и принят бит подтверждения) генератор BRG останавливается, вывод SCL "отпускается".

Рекомендованная последовательность действий при передаче данных:

a) Инициировать START установкой бита SEN (SSPCON2<0>) в '1'.
b) Ожидать прерывание (если оно разрешено) или установку бита SSPIF после завершения выполнения START.
c) Записью в SSPBUF инициируется передача адреса.
d) 7 бит адреса (при 7-разрядной адресации) и бит направления данных выдается на SDA.
e) Принять подтверждение -ACK от приемника, результат записывается в бит ACKSTAT регистра SSPCON2.
f) По заднему фронту девятого такта устанавливается бит SSPIF в '1'.
g) Записью в SSPBUF инициируется передача данных.
h) 8 бит данных выдаются на SDA.
i) Принять подтверждение -ACK от приемника, результат записывается в бит ACKSTAT регистра SSPCON2.
j) По заднему фронту девятого такта устанавливается бит SSPIF в '1'.
k) Инициировать STOP установкой бита PEN (SSPCON2<6>) в '1'.
l) Ожидать прерывание (если оно разрешено) или установку бита SSPIF после завершения выполнения STOP.

17.4.8 Генератор скорости обмена

В ведущем режиме, значение для перезагрузки BRG берется из младших 7 бит регистра SSPADD (см. рисунок 17-18). После загрузки SSPADD в BRG, счетчик BRG считает, декрементируя до нуля (в тактах Q2 и Q4), и останавливается до следующей перезагрузки, которая не всегда производится автоматически. Если после окончания счета сигнал на линии SCL должен перейти в высокий уровень, перезагрузка производится только после этого перехода (см. рисунок 17-19).

Примечание. Скорость обмена = F_{OSC} / (4 x (SSPADD + 1))

Рис. 17-18 Структурная схема генератора скорости обмена

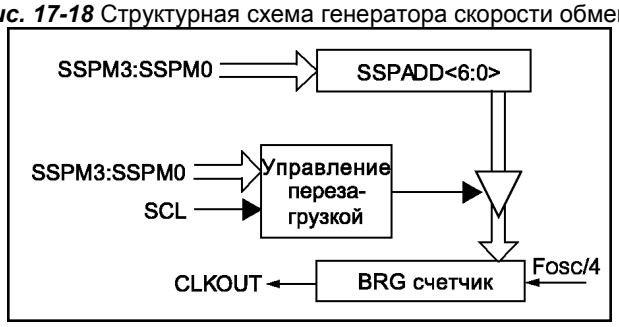
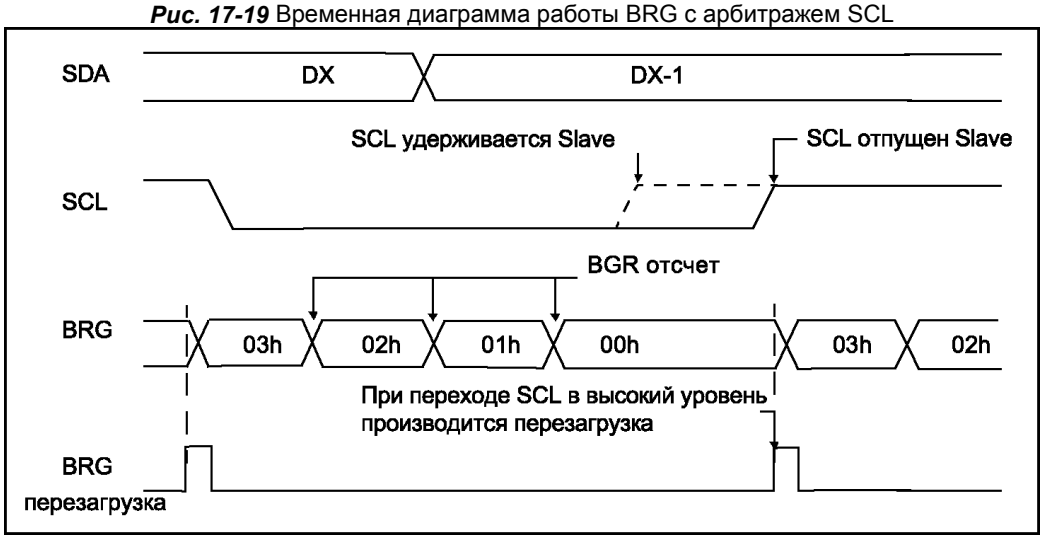


Рис. 17-19 Временная диаграмма работы BRG с арбитражем SCL



17.4.9 Формирование бита START в режиме ведущего I^2C

Чтобы инициировать формирование бита START на шине I^2C, необходимо установить бит SEN (SSPCON2<0>) в '1'. Если на линиях SCL и SDA высокий уровень сигнала, BRG загружается значением из регистра SSPADD <6:0> и начинает счет. Если по окончанию отсчета времени T_{BRG} сохраняется высокий уровень на SCL и SDA, сигнал SDA переводится в низкий логический уровень. Перевод SDA в низкий уровень, в то время когда на линии SCL высокий, и есть бит START на шине I^2C . После формирования бита START устанавливается бит S и флаг прерывания SSPIF в '1', BRG загружается новым значением и начинает счет. По окончании счета бит SEN (SSPCON2<0>) автоматически сбрасывается в '0', генератор останавливается, на SDA остается низкий уровень сигнала. Формирование бита START завершено.

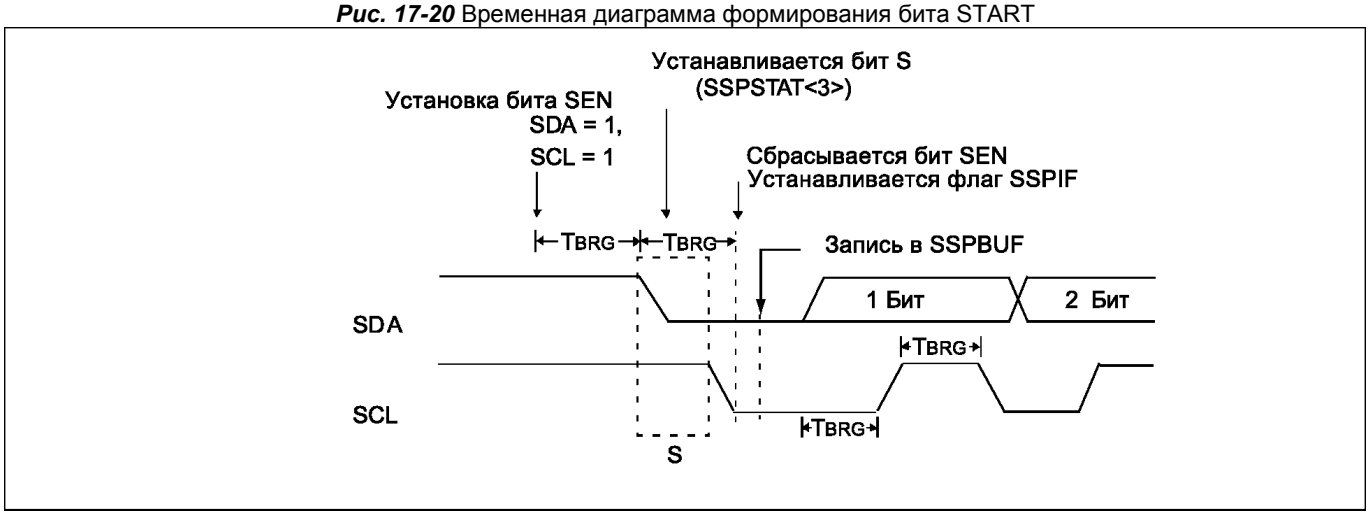
Примечание. Если в начале формирования бита START на SDA или SCL присутствует низкий уровень или во время выполнения START низкий уровень на SCL появляется раньше, чем на SDA, устанавливается флаг прерывания BCLIF (конфликт шины), выполнение START прекращается, MSSP переходит в состояние ожидания.

17.4.9.1 Флаг WCOL

Если во время формирования бита START производится попытка записи в SSPBUF, устанавливается бит WCOL, а запись не происходит.

Примечание. Поскольку MSSP не имеет стека событий, установка любого из младших 5 битов регистра SSPCON2 до завершения формирования бита START запрещено.

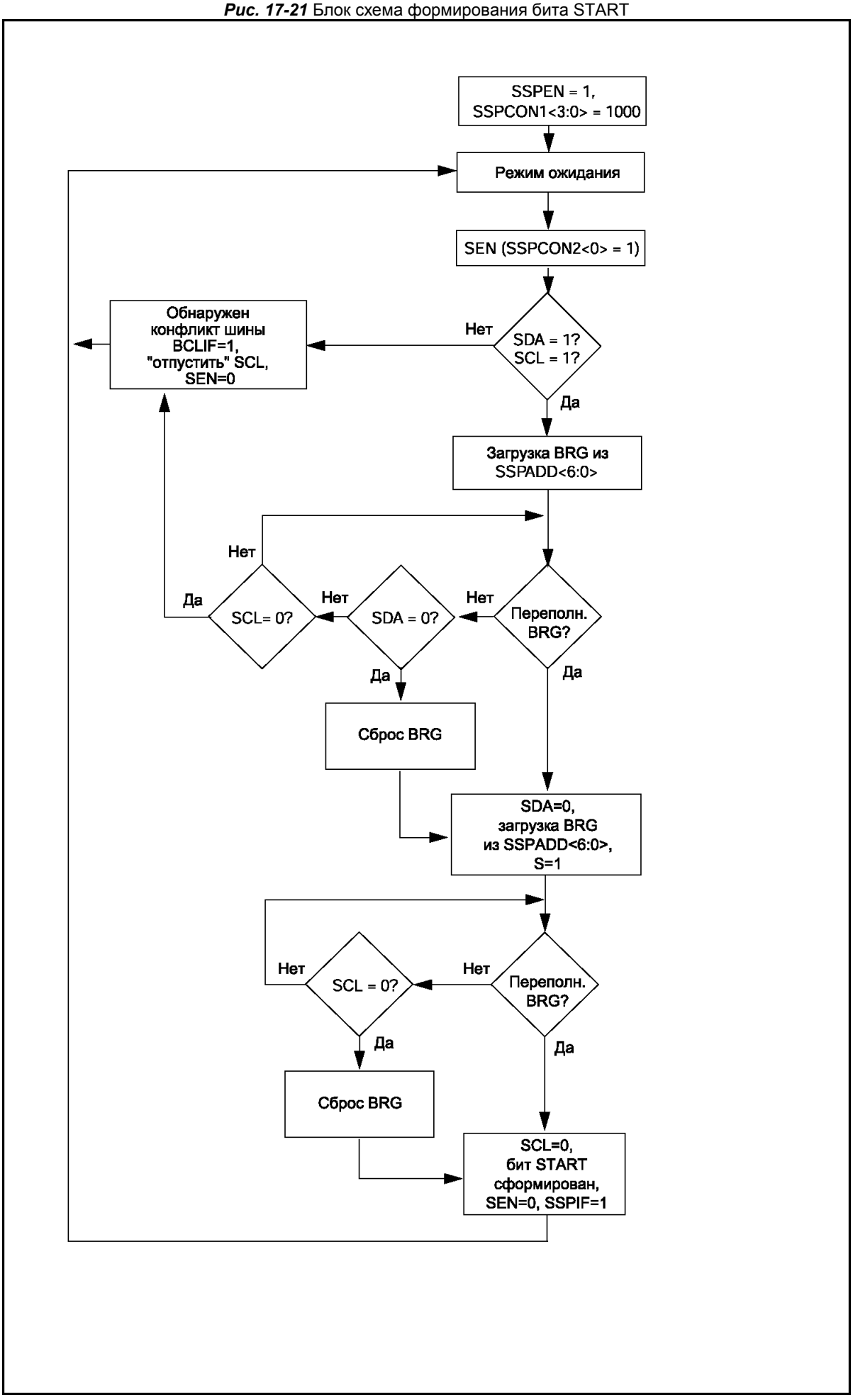
Рис. 17-20 Временная диаграмма формирования бита START



Примечание. T_{BRG} = один период генератора скорости обмена данными.

17

Рис. 17-21 Блок схема формирования бита START



17.4.10 Формирование бита повторный START в режиме ведущего I^2C

Чтобы инициировать формирование бита повторный START, необходимо установить бит RSEN (SSPCON2<1>) в то время, когда модуль MSSP находится в режиме ожидания. При включении формирования бита повторный START линия SCL переводится в низкий логический уровень. Когда на SCL устанавливается низкий уровень сигнала, BRG перезагружается содержимым регистра SSPADD<6:0> и начинает отсчет, при этом SDA "отпускается" в высокий уровень. Если по окончанию счета BRG, на линии SDA сохраняется высокий уровень, SCL также "отпускается". BRG вновь перезагружается и начинает отсчет. Если по окончании отсчета времени T_{BRG} сохраняется высокий уровень на линиях SCL и SDA, сигнал SDA переводится в низкий уровень. Бит RSEN (SSPCON2<1>) аппаратно сбрасывается в '0'. После формирования бита повторный START на шине I^2C устанавливается бит S (SSPSTAT<3>) в '1'. Флаг SSPIF не будет установлен в '1' до тех пор, пока BRG не перезагрузится новым значением и начнет счет.

Примечание 1. Если бит RSEN установлен в '1' во время выполнения какой либо операции на шине, то не будет выполнено никаких действий.
Примечание 2. Если на SDA низкий уровень при переходе SCL из низкого уровня в высокий или низкий уровень на SCL появляется раньше, чем на SDA, устанавливается флаг прерывания BCLIF (конфликт шины), формирование бита повторный START прекращается, MSSP переходит в состояние ожидания.

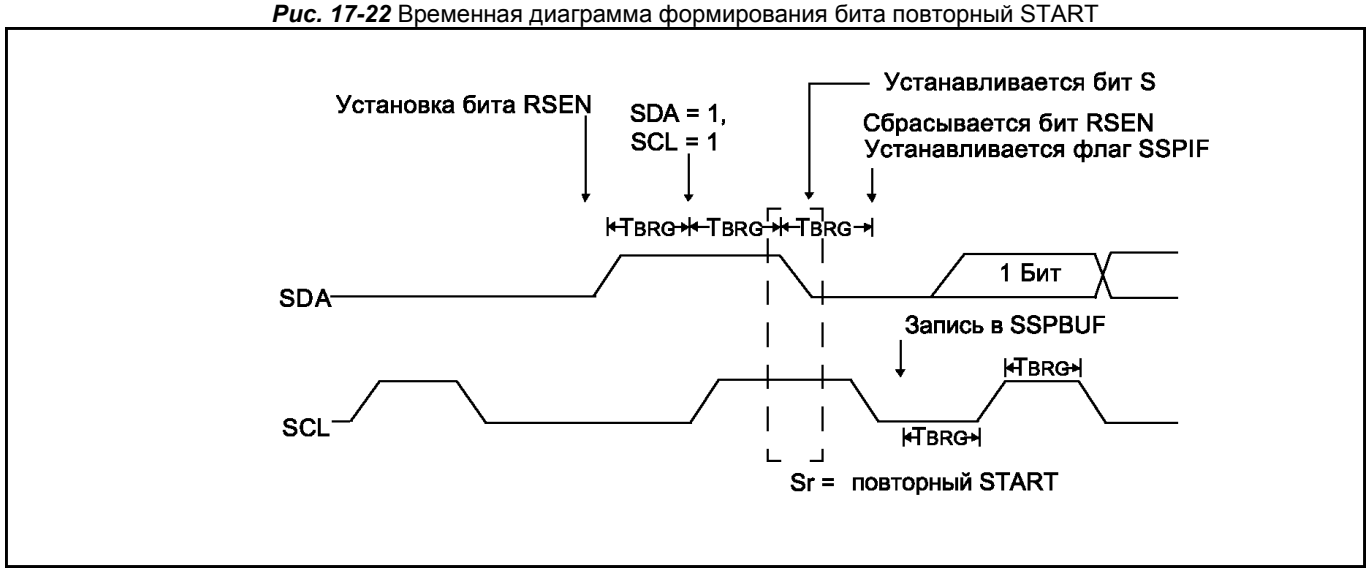
Сразу после установки бита SSPIF пользователь может загрузить регистр SSPBUF 7-разрядным адресом (для 7-разрядного режима адресации) или старшим байтом 10-разрядного адреса. По завершению передачи 8 бит и получения подтверждения ACK, можно передать байт данных или младший байт адреса.

17.4.10.1 Флаг WCOL

Если во время формирования бита повторный START производится попытка записи в SSPBUF, устанавливается бит WCOL, а запись не происходит.

Примечание. Поскольку MSSP не имеет стека событий, установка любого из младших 5 битов регистра SSPCON2 до завершения формирования бита повторный START запрещено.

Рис. 17-22 Временная диаграмма формирования бита повторный START



Примечание. T_{BRG} = один период генератора скорости обмена данными.

Рис. 17-23 Блок схема формирования бита повторный START (часть 1 из 2)

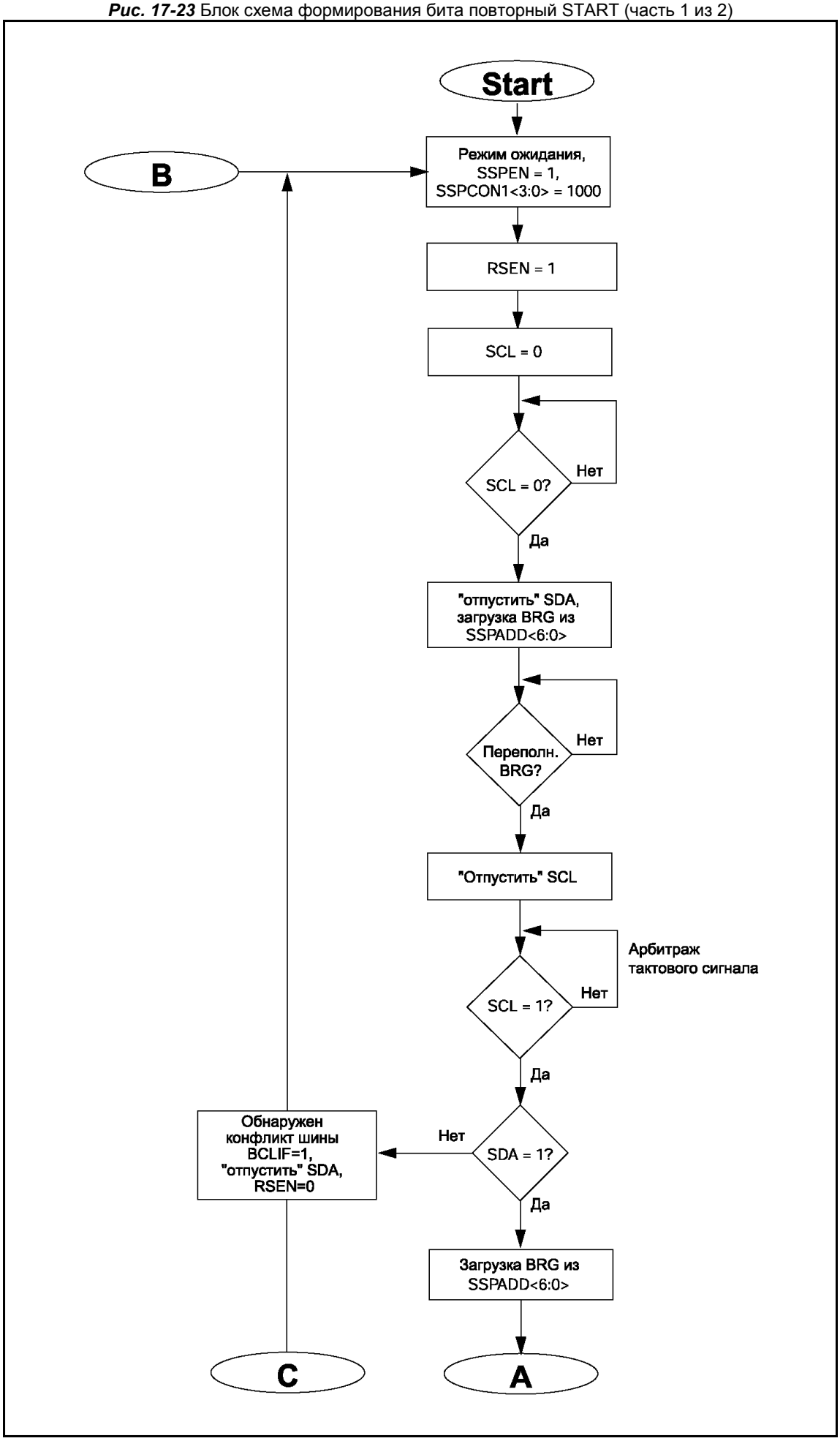
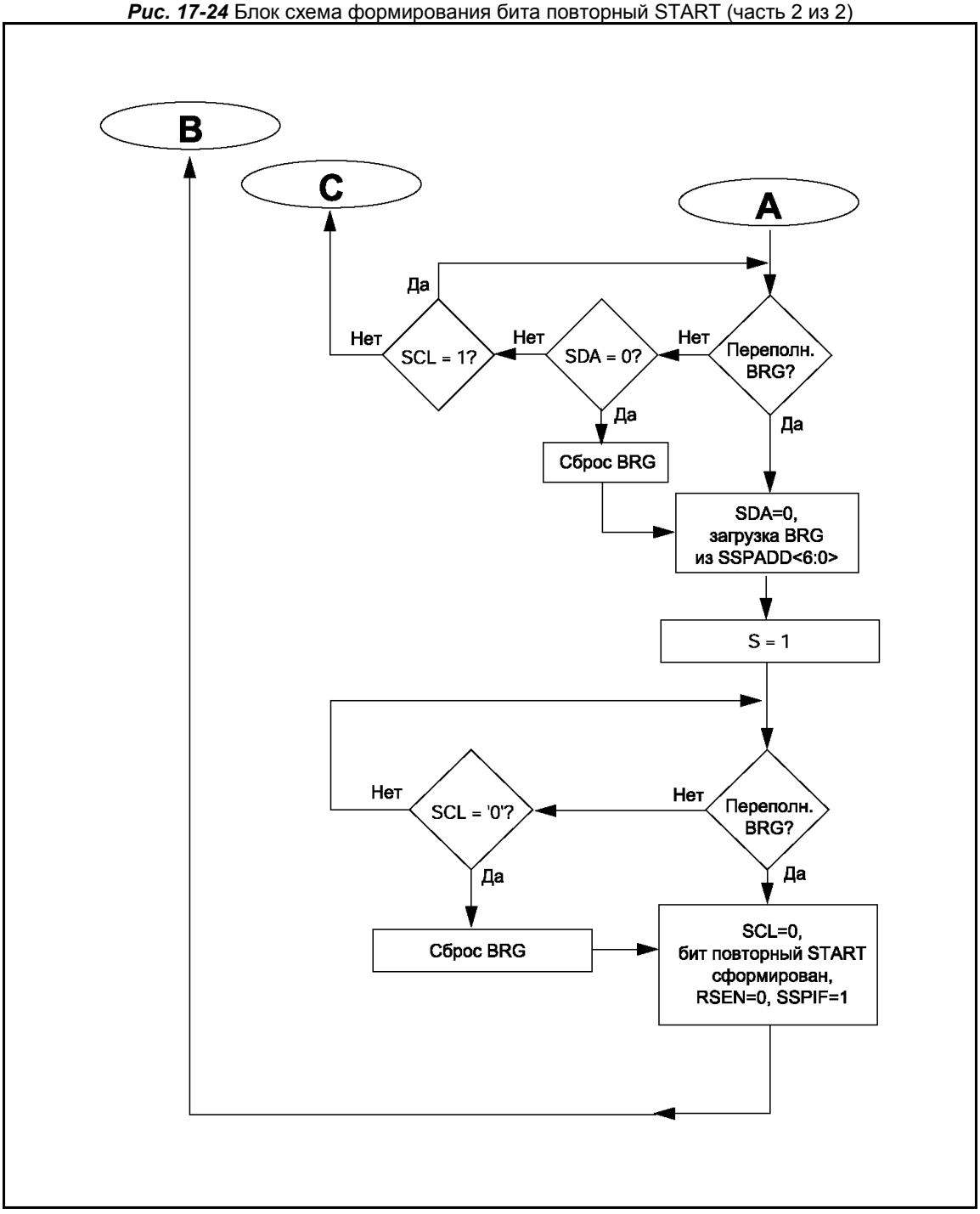


Рис. 17-24 Блок схема формирования бита повторный START (часть 2 из 2)



17.4.11 Передача данных в режиме ведущего I^2C

Для инициализации передачи байта данных, 7-разрядного адреса или любой части 10-разрядного адреса нужно просто записать байт в регистр SSPBUF. В результате чего установится бит BF в '1', а BRG начнет формировать сигнал для передачи данных. Каждый передаваемый бит будет выдаваться на SDA по заднему фронту сигнала SCL. Низкий уровень на SCL удерживается в течение одно периода BRG. Данные должны поступать на SDA до прихода переднего фронта на SCL (см. раздел временных характеристик, параметр 106). После "отпускания" SCL в высокий уровень на время T_{BRG} данные должны удерживаться на SDA в том же состоянии. По окончании передачи 8-го бита сбрасывается флаг BF в '0', а ведущий "отпускает" SDA с тем, чтобы принять бит подтверждения. По заднему фронту 9-го такта значение ACK записывается в бит ACKSTAT регистра SSPCON2. В этот же момент устанавливается флаг SSPIF в '1', а BRG отключается до следующей операции на шине оставляя низкий уровень на SCL и отпуская SDA (см. рисунок 17-26).

17.4.11.1 Флаг BF

В режиме передачи данных бит BF (SSPSTAT<0>) аппаратно устанавливается в '1' после записи данных в регистр SSPBUF и аппаратно сбрасывается после передачи 8 бит данных.

17.4.11.2 Флаг WCOL

Если во время передачи данных производится попытка записи в регистр SSPBUF, устанавливается бит WCOL в '1', а запись не происходит. Бит WCOL сбрасывается программно.

17.4.11.3 Флаг ACKSTAT

В режиме передачи данных бит ACKSTAT(SSPCON2<6>) равен нулю, если ведомый сформировал подтверждение. Ведомый посылает подтверждение, если он распознал адрес (включая общий вызов) или корректно принял данные.

Рис. 17-25 Блок схема передачи данных в режиме ведущего I^2C

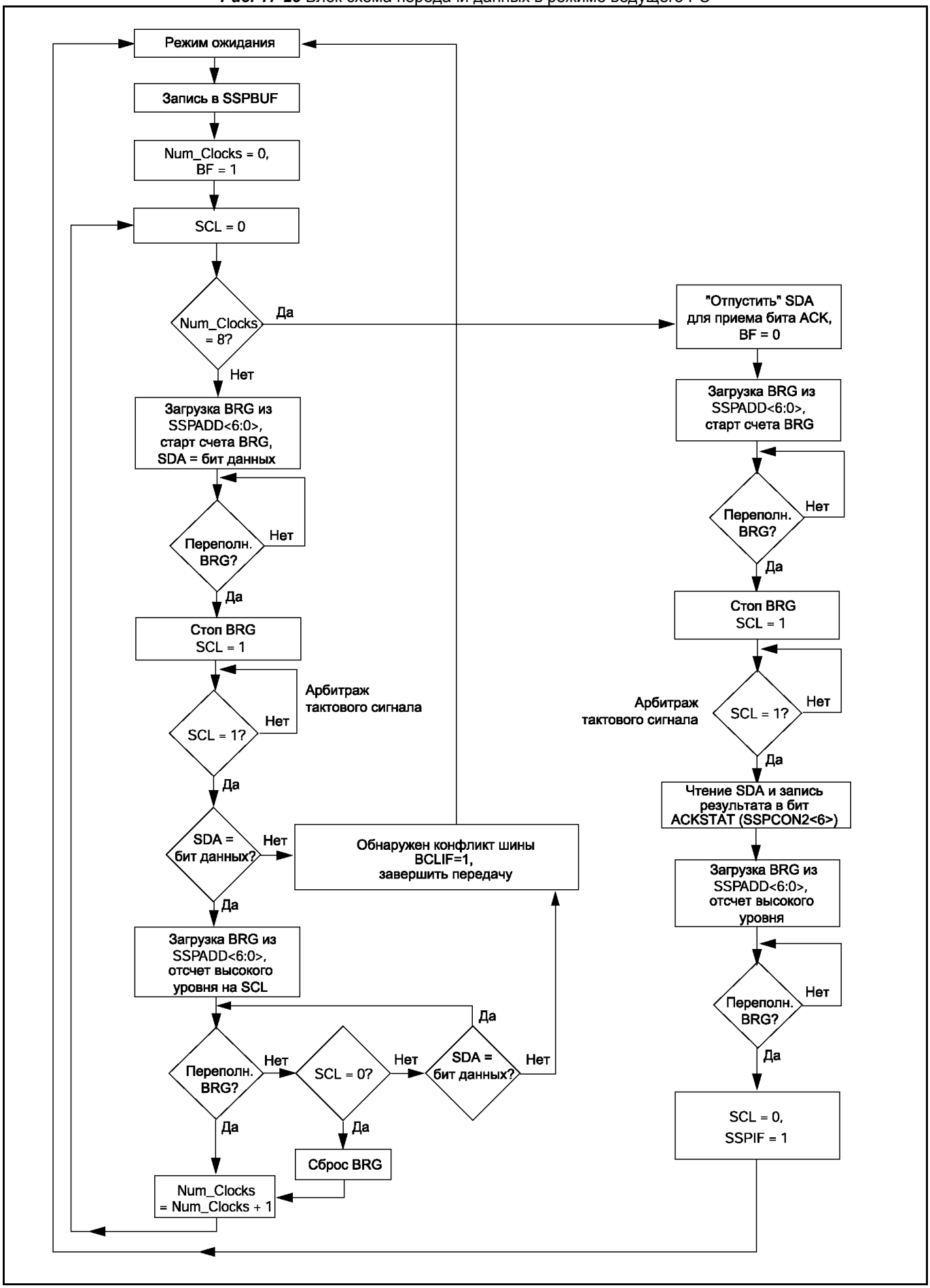
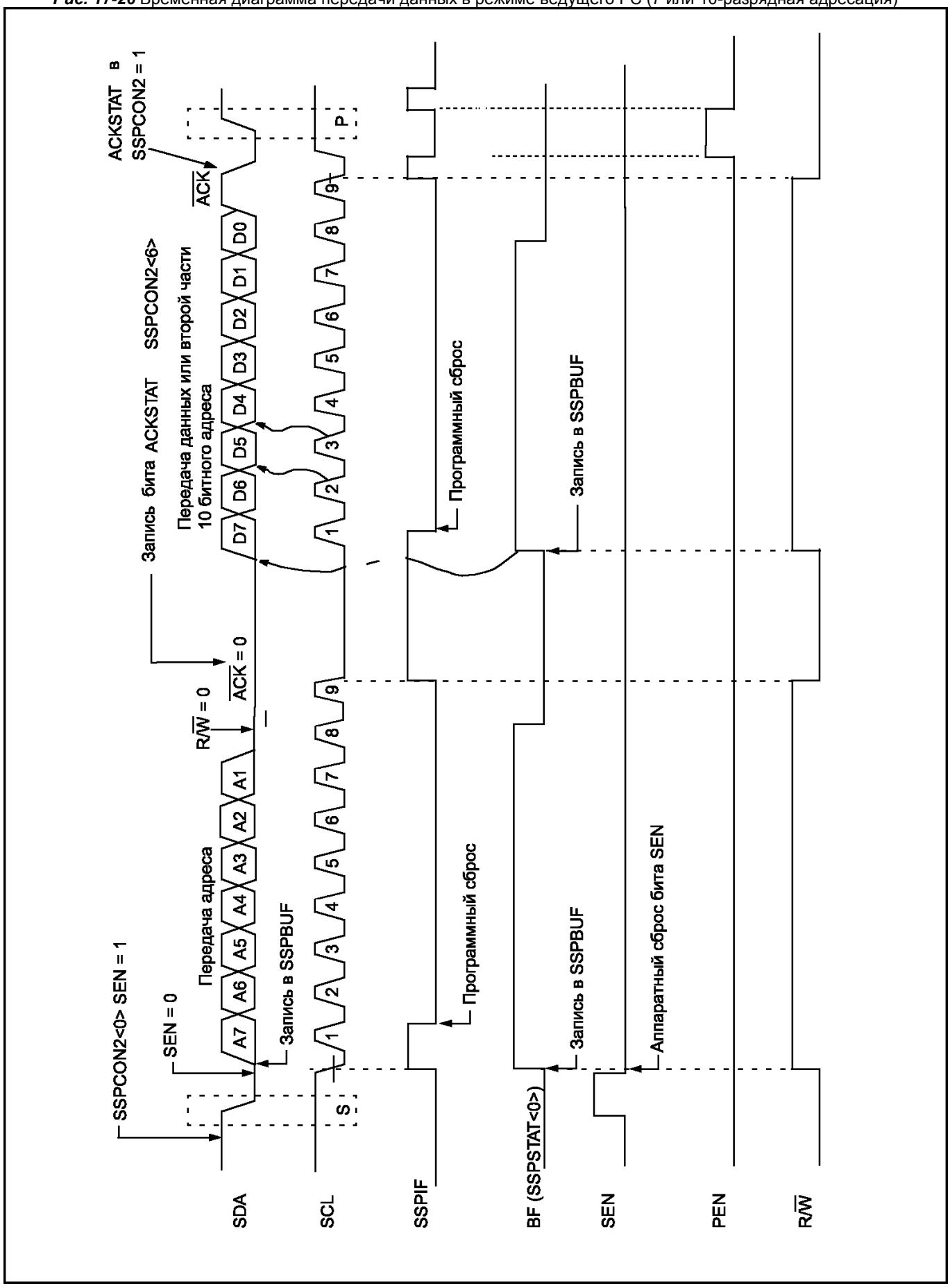


Рис. 17-26 Временная диаграмма передачи данных в режиме ведущего I^2C (7 или 10-разрядная адресация)



17.4.12 Прием данных в режиме ведущего I^2C

Прием данных ведущем шины I^2C разрешается установкой бита RCEN(SSPCON2<3>) в '1'.

Примечание. При установке бита RCEN в '1' модуль MSSP должен находиться в режиме ожидания.

BRG начинает формировать тактовый сигнал SCL, для приема данных в сдвиговый регистр SSPSR. Каждый бит данных будет приниматься с SDA по заднему фронту SCL. По заднему фронту 8-го такта, значение из SSPSR переписывается в SPPBUF, устанавливается бит BF и SSPIF в '1', BGR останавливается, удерживая SCL в низком уровне, а модуль MSSP переходит в режим ожидания. После чтения регистра SSPBUF аппаратно сбрасывается бит BF в '0'. По окончанию приема, ведущий может сформировать бит подтверждения установкой бита ACKEN (SSPCON2<4>) в '1'.

17.4.12.1 Флаг BF

В режиме приема данных бит BF (SSPSTAT<0>) аппаратно устанавливается в '1' после загрузки данных в регистр SSPBUF и аппаратно сбрасывается после чтения регистра SSPBUF.

17.4.12.2 Флаг SSPOV

При приеме данных бит SSPOV устанавливается в '1', если в момент приема 8-го бита следующего байта бит BF=1 после приема предыдущего байта.

17.4.12.3 Флаг WCOL

Если во время приема данных производится попытка записи в регистр SSPBUF, устанавливается бит WCOL в '1', а запись не происходит. Бит WCOL сбрасывается программно.

Рис. 17-27 Блок схема приема данных в режиме ведущего I^2C

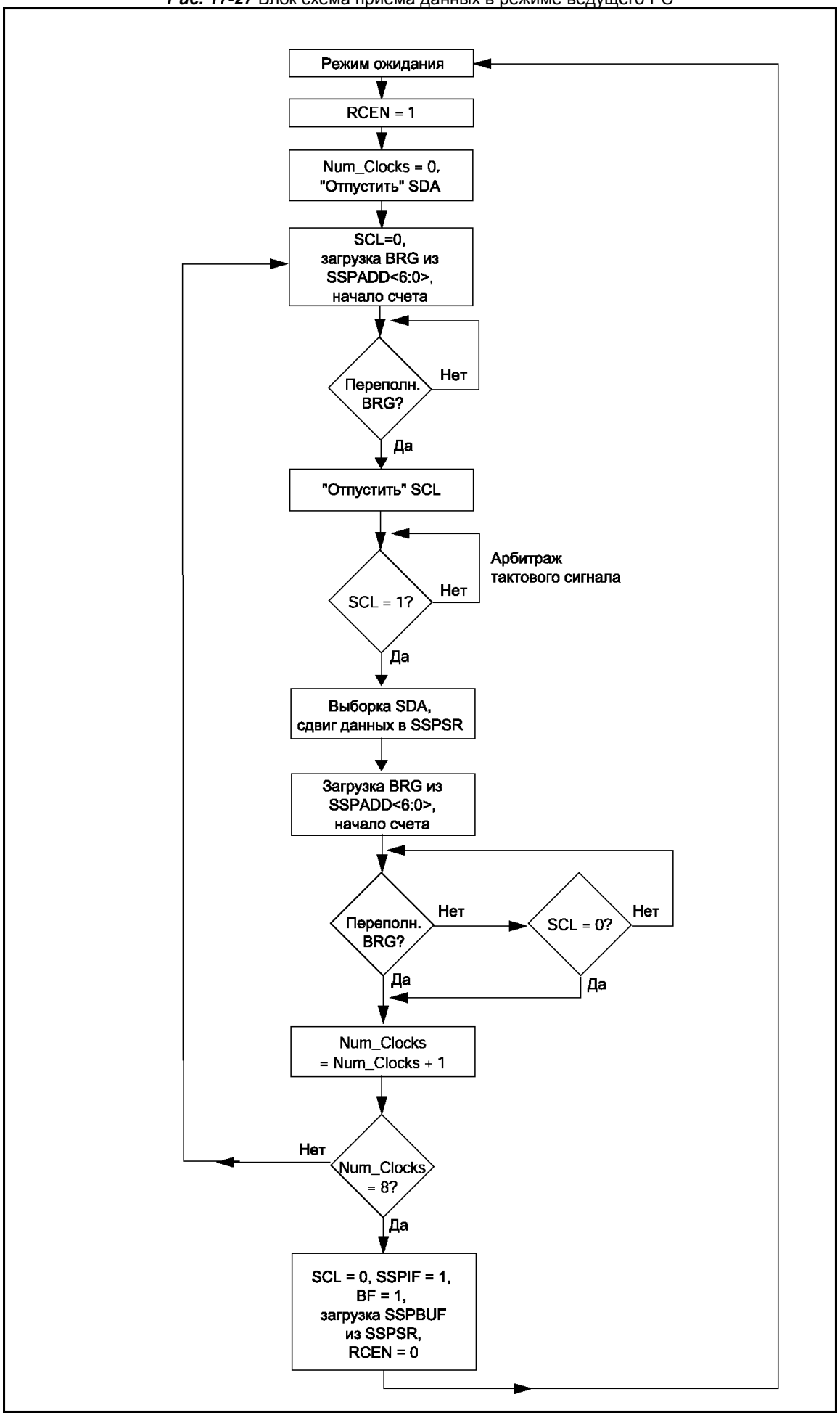
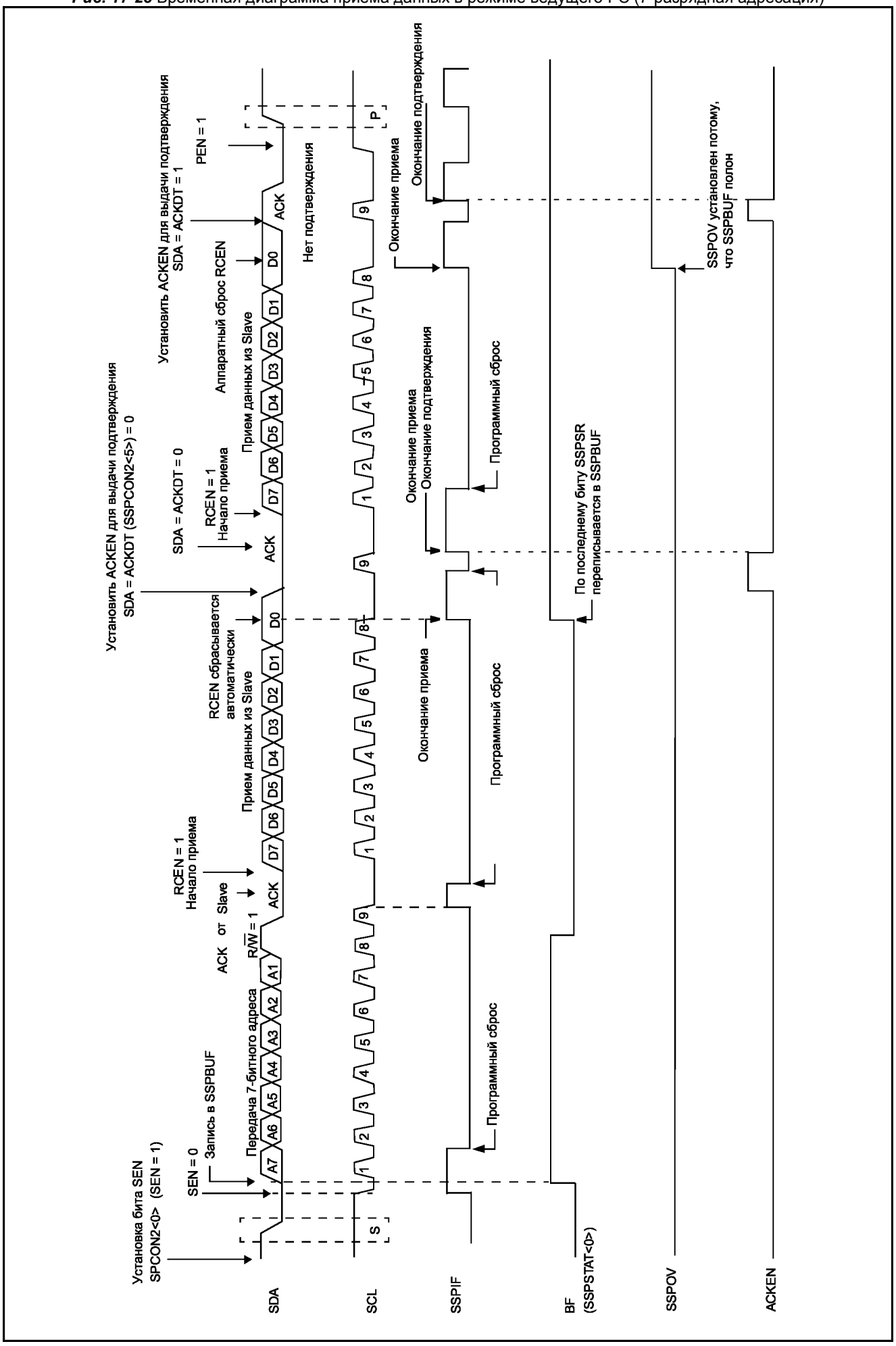


Рис. 17-28 Временная диаграмма приема данных в режиме ведущего I^2C (7-разрядная адресация)



17.4.13 Формирование бита подтверждения в режиме ведущего I²C

Для инициализации формирования бита подтверждения на шине I²C необходимо установить бит ACKEN (SSPCON2<4>) в '1'. При установке этого бита на SCL выдается низкий уровень сигнала, а на SDA содержимое бита ACKDT. Если нужно подтвердить прием, бит ACKDT должен быть равен нулю. По окончанию счета BRG линия SCL "отпускается". Как только SCL перейдет из низкого уровня в высокий, BRG опять начнет счет. После окончания счета SCL переводится в низкий уровень, бит ACKEN автоматически сбрасывается в '0', устанавливается флаг прерывания SSPIF в '1', BGR останавливается, а модуль MSSP переходит в режим ожидания (см. рисунок 9-29).

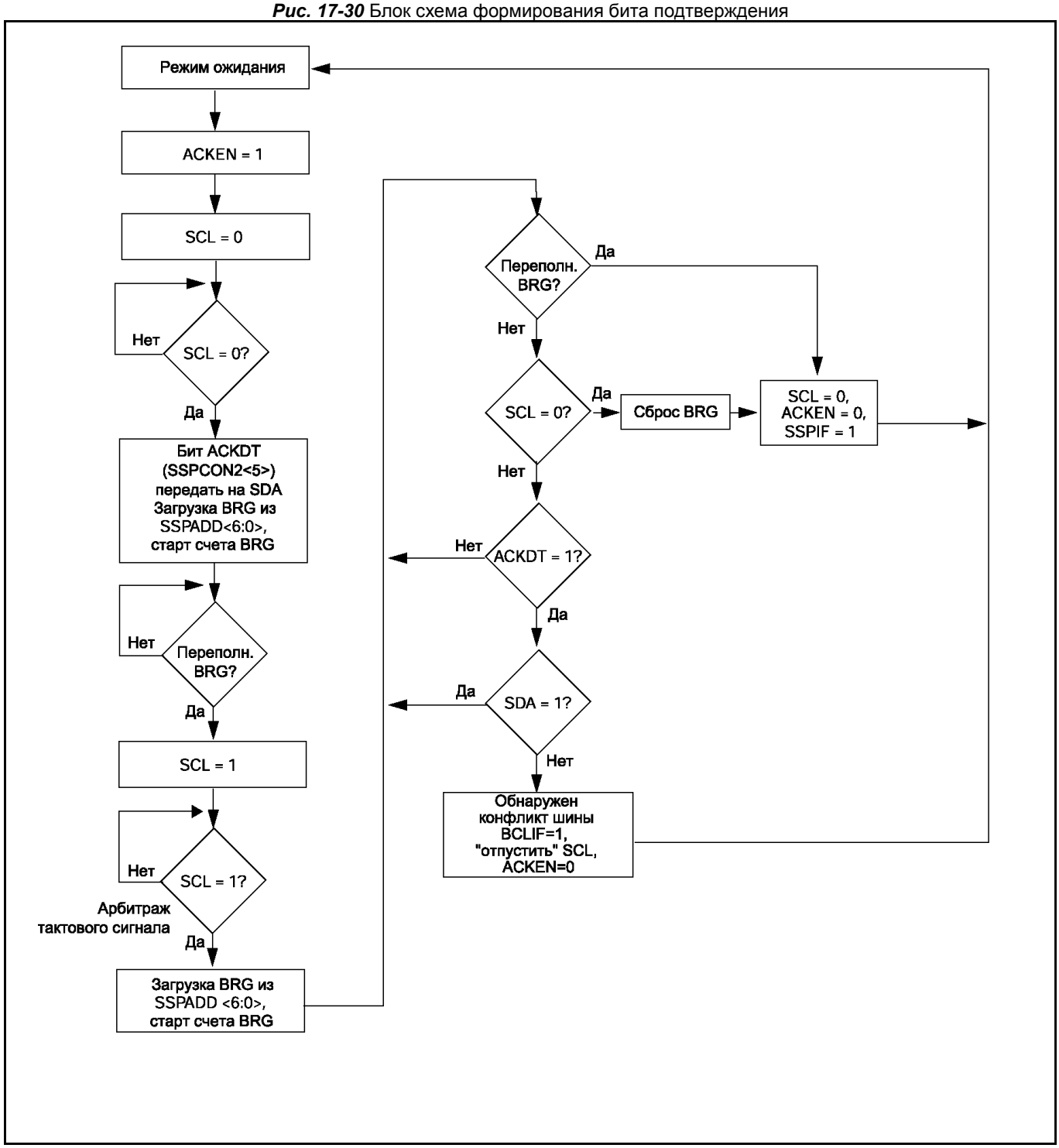
17.4.13.1 Флаг WCOL

Если во время формирования бита подтверждения производится попытка записи в SSPBUF, устанавливается бит WCOL в '1', а запись не происходит.

Рис. 17-29 Временная диаграмма формирования бита подтверждения

Примечание. T_{BRG} = один период генератора скорости обмена данными.

Рис. 17-30 Блок схема формирования бита подтверждения



17.4.14 Формирование бита STOP в режиме ведущего I²C

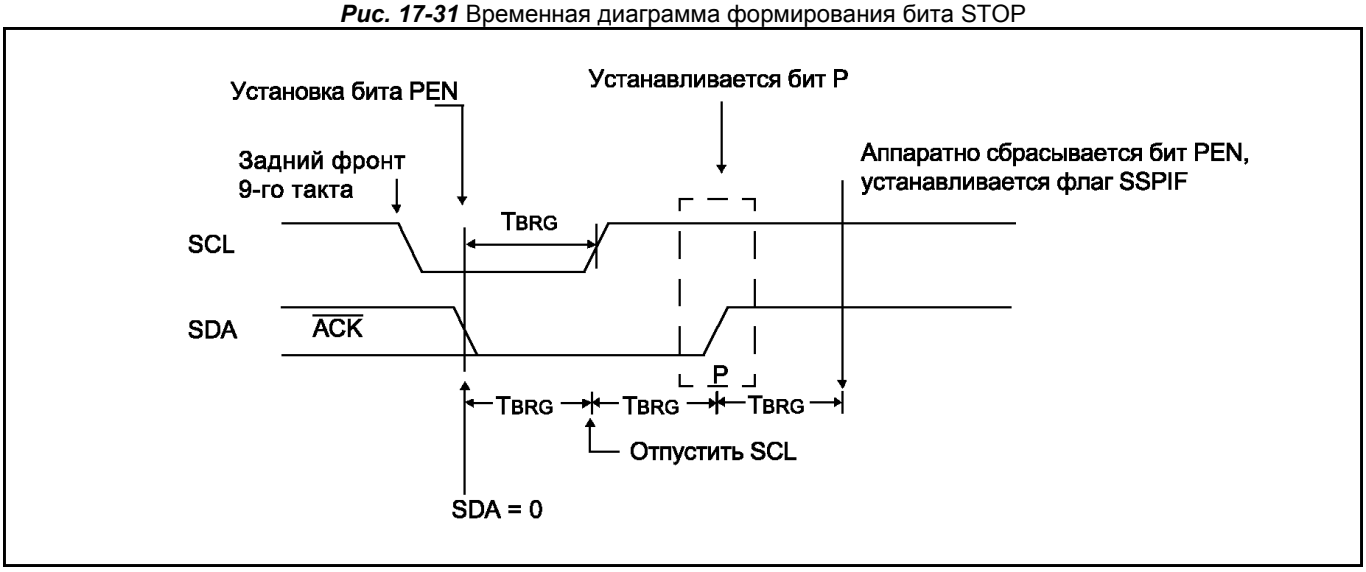
Чтобы инициировать формирование бита STOP, необходимо установить бит PEN (SSPCON2<2>) в '1'. По окончании приема/передачи данных, после прохождения заднего фронта тактового сигнала на SCL удерживается низкий уровень сигнала. При установки бита PEN ведущий выдает низкий уровень на линию SDA, перезагружает BRG и начинает счет до нуля. По окончании счета линия SCL "отпускается". Через время T_{BRG}, после установки высокого уровня на SCL, "отпускается" SDA. Когда на SDA появляется высокий уровень сигнала, устанавливаются биты P и SSPIF в '1', бит PEN автоматически сбрасывается в '0', а генератор BRG останавливается (см. рисунок 17-31).

Перед попыткой передать данные программное обеспечение должно проверить занятость шины (состояние битов S и P в регистре SSPSTAT). Если шина занята, то могут быть разрешены прерывания по обнаружению бита STOP.

17.4.14.1 Флаг WCOL

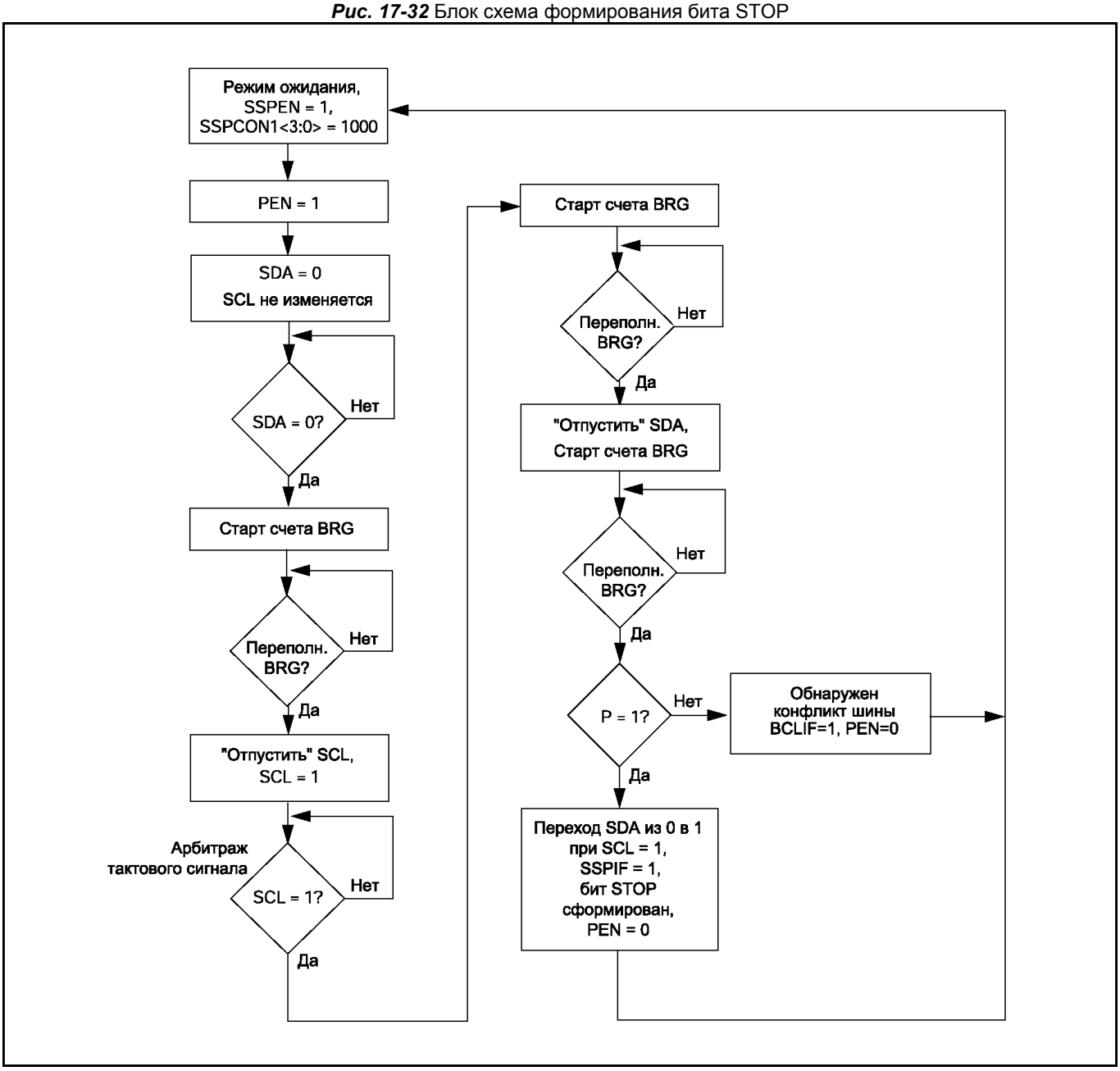
Если во время формирования бита STOP производится попытка записи в SSPBUF, устанавливается бит WCOL в '1', а запись не происходит.

Рис. 17-31 Временная диаграмма формирования бита STOP



Примечание. T_{BRG} = один период генератора скорости обмена данными.

Рис. 17-32 Блок схема формирования бита STOP



17.4.15 Синхронизация тактового сигнала

Синхронизация тактового сигнала производится каждый раз во время приема/передачи данных, формирования бита START или STOP и т.д. При "отпускании" ведущем SCL (SCL должен перейти в высокий уровень). В этот момент BRG приостанавливается пока на SCL не появится высокий уровень сигнала. При появлении сигнала высокого уровня на SCL генератор BRG перегружается значением из SSPADD<6:0> и начинает счет. Это гарантирует, что длительность высокого уровня сигнала на SCL всегда будет не меньше T_{BRG}, даже если другое устройство на шине удерживает тактовый сигнал.

Рис. 17-33 Временная диаграмма синхронизации тактового сигнала при передаче данных ведущем шины

17.4.16 Работа в SLEEP режиме

Ведущий I^2C не может принимать адресные байты или байты данных в SLEEP режиме микроконтроллера.

17.4.17 Эффект сброса

При сбросе микроконтроллера модуль MSSP выключается, прекращается любой обмен данными.

17.4.18 Режим конкуренции, арбитраж и конфликты шины

В режиме конкуренции необходимо поддерживать правила арбитража шины. Во время передачи адреса/данных на SDA ведущий может потерять арбитраж, если он формирует высокий уровень сигнала, а другой ведущий сформировал низкий уровень на SDA. При переходе SCL в высокий уровень, сигнал на SDA изменяться не может. Если на SDA ожидается высокий уровень, а в действительности низкий, значит возник конфликт шины. Обнаружив конфликт шины, ведущий устанавливает флаг прерывания BCLIF в '1', прекращает текущую операцию на шине и переводит порт I^2C в режим ожидания (см. рисунок 17-34).

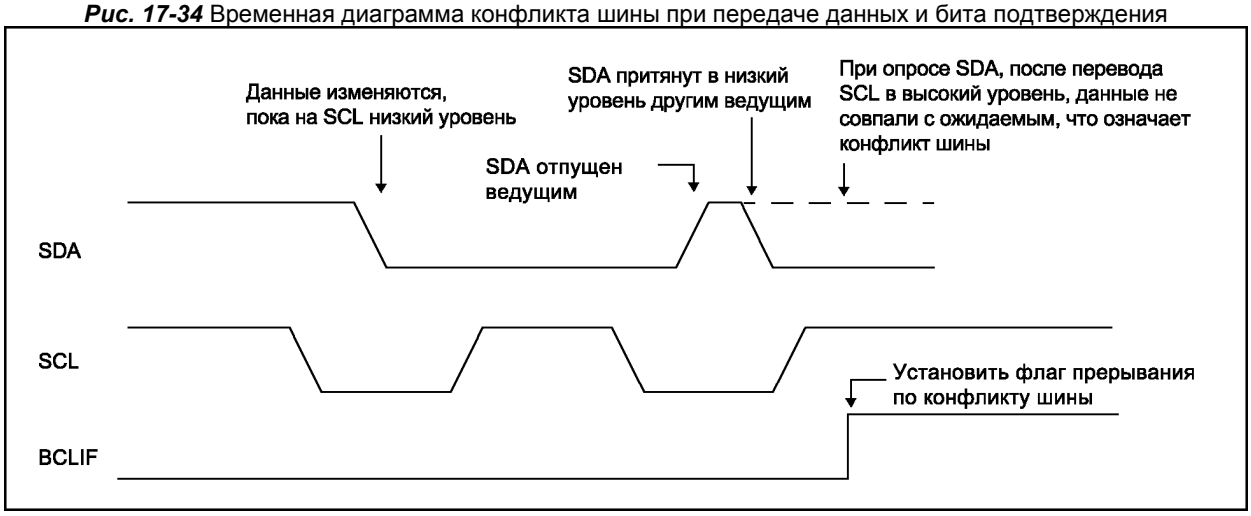
Если при возникновении конфликта шины выполнялась передача данных, она обрывается, устанавливается бит BF в '1', а линии SCL и SDA "отпускаются" в высокое состояние. В регистр SSPBUF может быть произведена запись, причем запись в SSPBUF инициирует передачу независимо от того, в какой момент передатчик отключился при возникновении конфликта шины. Если пользователь обрабатывает прерывания по конфликту шины, после освобождение шины он может продолжить обмен, сформировав бит START.

Если при возникновении конфликта выполнялось формирование бита START, повторный START, STOP или ACK, выполняемая операция обрывается, SCL и SDA "отпускаются", а соответствующий бит управления в SSPCON2 сбрасывается в '0'. Если пользователь обрабатывает прерывания по конфликту шины, после освобождение шины он может продолжить обмен, сформировав бит START.

Ведущий продолжает следить за состоянием шины, и при появлении бита STOP устанавливается флаг прерывания SSPIF в '1'.

В режиме конкуренции использование прерывания при обнаружении битов START и STOP позволяет определить занятость шины. Управление шиной может быть перехвачено при установленном бите P или сброшенных битах S и P.

Рис. 17-34 Временная диаграмма конфликта шины при передаче данных и бита подтверждения



17

17.4.18.1 Конфликт шины при формировании бита START

Во время формирования бита START конфликт шины возникает если:

a) В начале START на SDA или SCL низкий уровень сигнала (см. рисунок 17-35);
b) На SCL низкий уровень появляется раньше чем на линии SDA (см. рисунок 17-36).

Во время формирования бита START сигналы SCL и SDA продолжают отслеживаться. Если SCL или SDA имеют низкий уровень сигнала, то формирование бита START прекращается, устанавливается флаг BCLIF в '1', а модуль MSSP переходит в режим ожидания (см. рисунок 17-35).

Бит START начинается при наличии высокого уровня сигнала на линиях SCL и SDA. Если на SCL появляется низкий уровень раньше, чем на SDA, возникает конфликт шины, поскольку это подразумевает, что другой ведущий пытается в это время передать данные.

Если во время счета BRG на SDA появляется низкий уровень сигнала, BRG сбрасывается, а на SDA формируется низкий уровень раньше времени (см. рисунок 17-37). Если же на SDA высокий уровень, низкий уровень формируется в конце счета BRG. Генератор BRG перезагружается и считает до нуля. Если в это время на SCL появится низкий уровень, конфликт шины не возникает. В конце счета BRG SCL переводится в низкий уровень.

Примечание. Конфликт шины во время START не возникает, потому что два или более ведущих могут сформировать START одновременно, но при этом один из них первым переведет SDA в низкий уровень. Конфликт шины не возникает, поскольку ведущие могут продолжить арбитраж во время передачи адреса, данных, формировании бита повторный START и STOP.

Рис. 17-35 Временная диаграмма конфликта шины во время формирования бита START (только SDA)

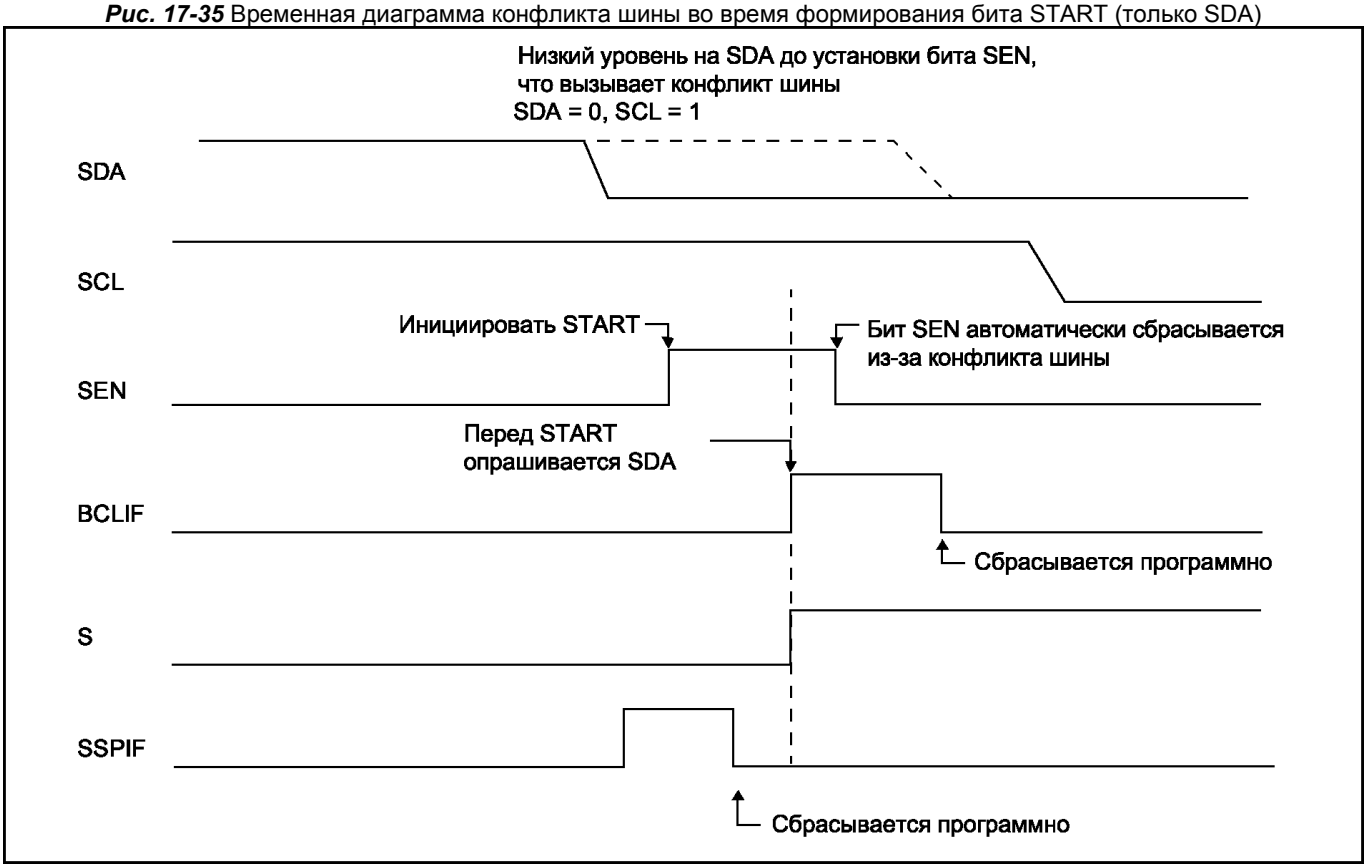


Рис. 17-36 Временная диаграмма конфликта шины во время формирования бита START (SCL=0)

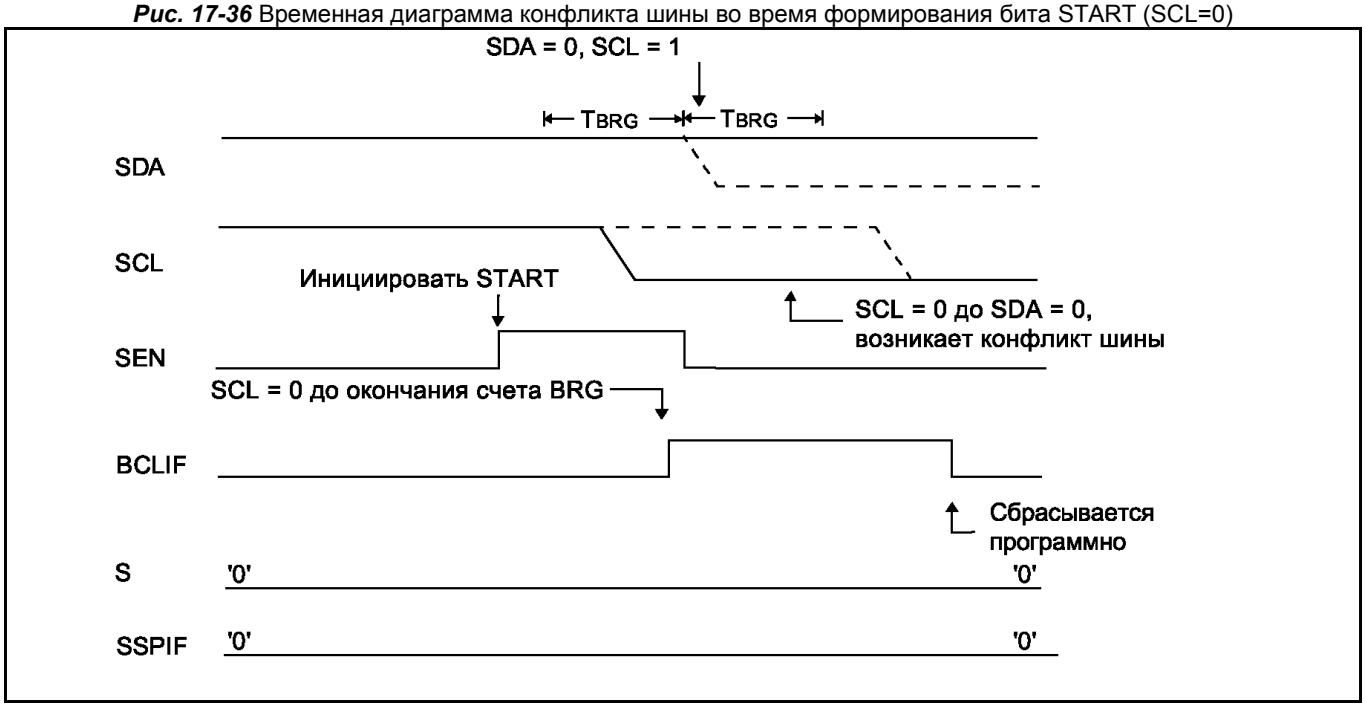
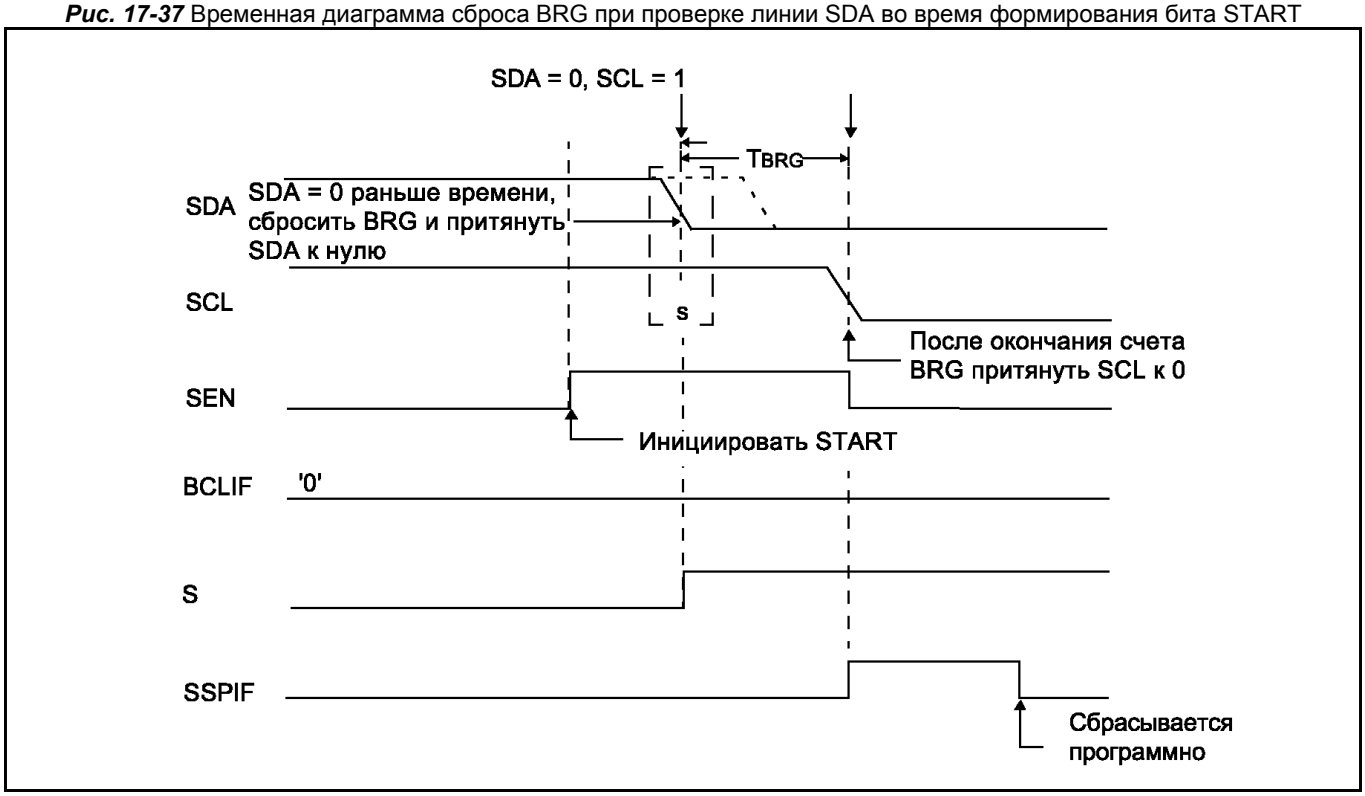


Рис. 17-37 Временная диаграмма сброса BRG при проверке линии SDA во время формирования бита START



17.4.18.2 Конфликт шины при формировании бита повторный START

Во время формирования бита повторный START конфликт шины возникает если:

a) На SDA низкий уровень при переходе SCL из низкого уровня в высокий (см. рисунок 17-38);
b) SCL переходит в низкий уровень раньше SDA, что указывает на то, что другой ведущий пытается передать данные.

После "отпускания" линии SDA сигнал на выводе должен перейти в высокий уровень, после чего BRG перезагружается и начинает счет. Затем "отпускается" SCL и при появлении на нем высокого уровня опрашивается SDA. Если на SDA низкий уровень сигнала, значит произошел конфликт шины, т.е. другой ведущий пытается передать данные. Если на SDA высокий уровень, то BRG снова перезагружается и начинается счет. Если SDA переходит в низкий уровень до окончания счета, конфликт шины не происходит, поскольку два или более ведущих могут пытаться получить доступ к шине одновременно.

Если на линии SCL сигнал переходит в низкий уровень до окончания счета, а на SDA сохраняется высокий уровень, значит, произошел конфликт шины, т.е. другой ведущий пытается передать данные.

Если по окончании счета BGR на SCL и SDA высокий уровень, то SDA переводится в низкий уровень, а BRG перезагружается и начинает счет. По окончании счета, независимо от уровня сигнала на SCL он переводится в низкий уровень (см. рисунок 17-39).

Рис. 17-38 Временная диаграмма конфликта шины во время формирования бита повторный START (случай 1)

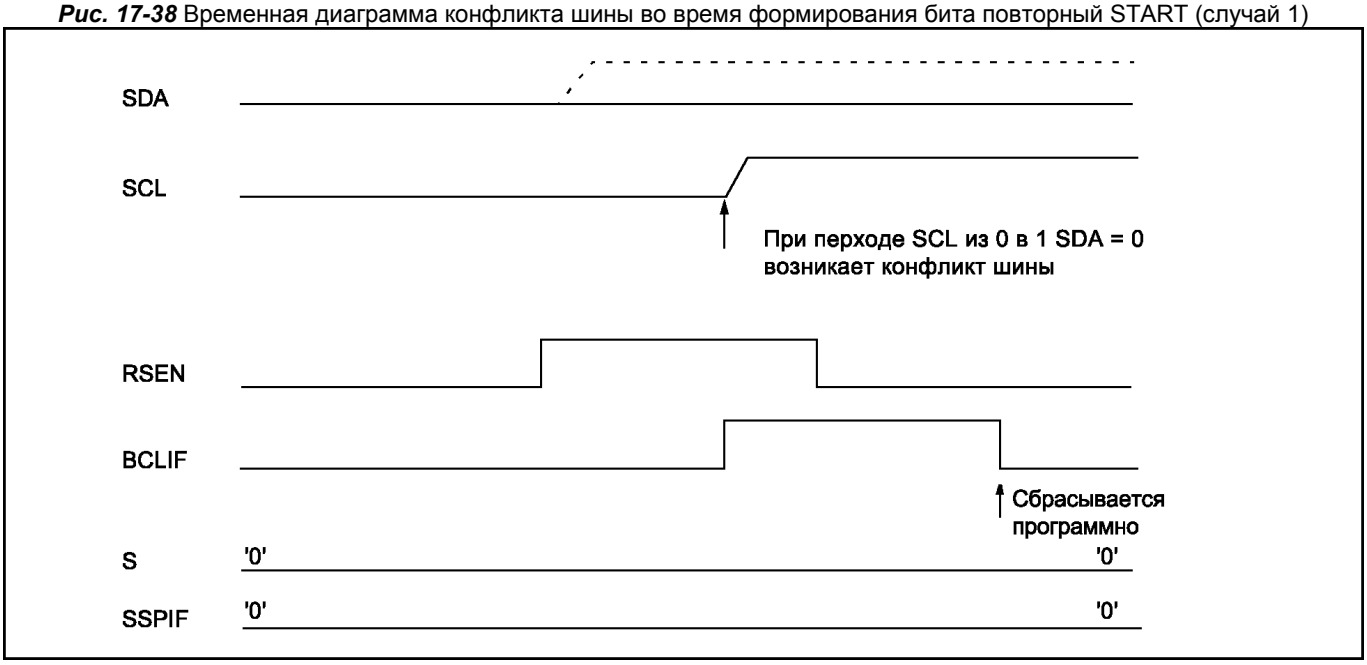
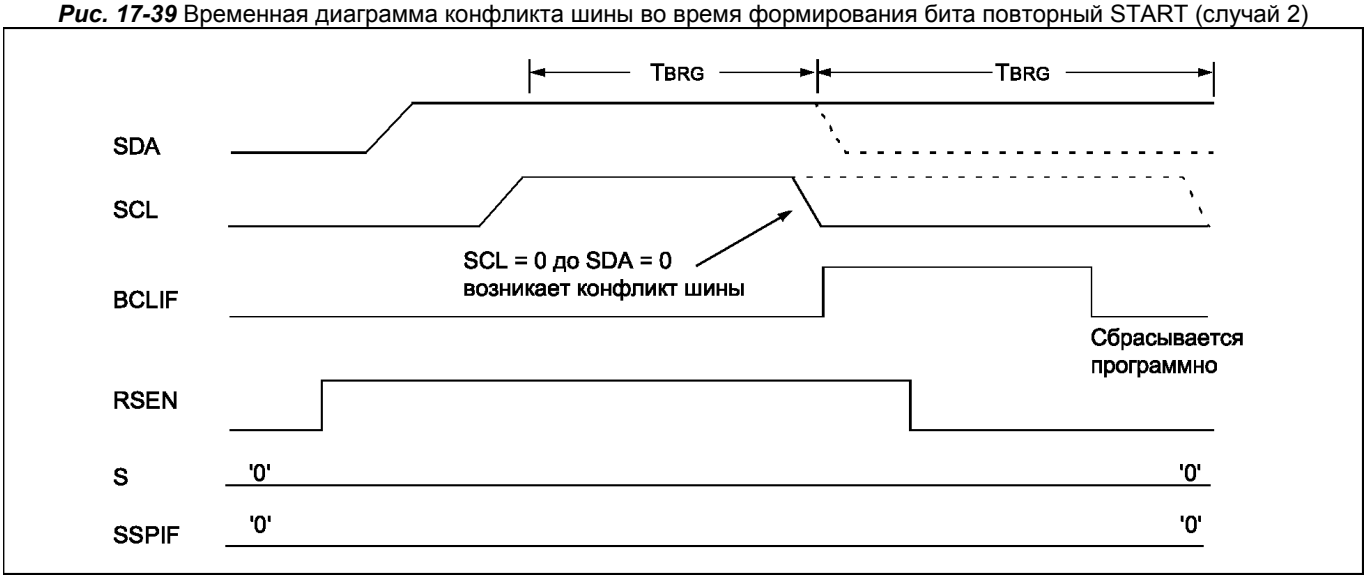


Рис. 17-39 Временная диаграмма конфликта шины во время формирования бита повторный START (случай 2)



17.4.18.3 Конфликт шины при формировании бита STOP

Во время формирования бита STOP конфликт шины возникает если:

a) После "отпускания" линии SDA и окончания счета BRG на SDA по-прежнему низкий уровень сигнала (см. рисунок 17-40);
b) После "отпускания" линии SDA сигнал на SCL переходит в низкий уровень до того, как на SDA перейти в высокий уровень (см. рисунок 17-41).

Формирование бита STOP начинается с перевода линии SDA в низкий уровень, затем SCL "отпускается". После появления на SCL высокого уровня BRG перезагружается и начинает счет. По окончании счета SDA "отпускается", BRG перезагружается и снова начинает счет и опрашивает SDA. Если на нем низкий уровень или на SCL появился низкий уровень до перехода SDA в высокий, значит, произошел конфликт шины, т.е. другой ведущий пытается передать данные.

Рис. 17-40 Временная диаграмма конфликта шины во время формирования бита STOP (случай 1)

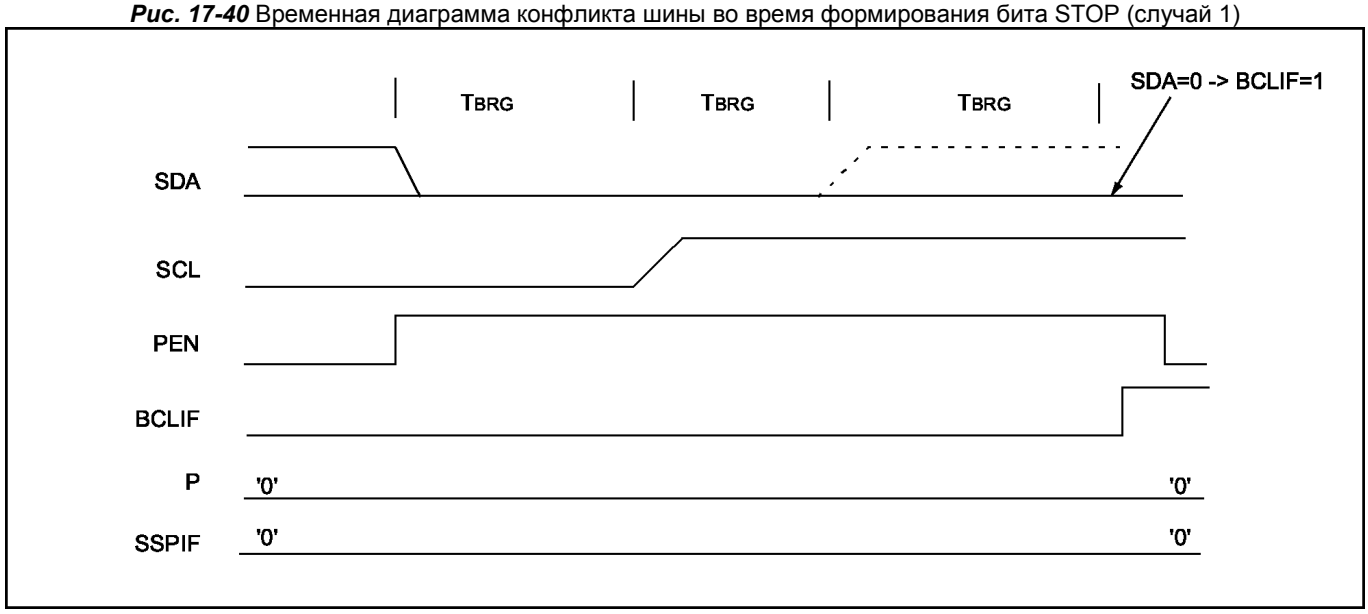


Рис. 17-41 Временная диаграмма конфликта шины во время формирования бита STOP (случай 2)

17

17.5 Подключение к шине I^2C

Для стандартного режима I^2C значение резисторов Rp и Rs (см. рисунок 17-42) зависит от следующих параметров:

- Напряжение питания;
- Емкость шины;
- Количество устройств на шине (входной ток + ток утечки).

Напряжение питания ограничивает минимальное значение сопротивления Rp, из-за ограничения минимального тока стока 3мА при V_{OL} max = 0.4В.

Например:

V_{DD} = 5В +/- 10%
V_{OL} max = 0.4 В при 3мА

Rp min = (5.5 – 0.4) / 0.003 = 1.7 кОм

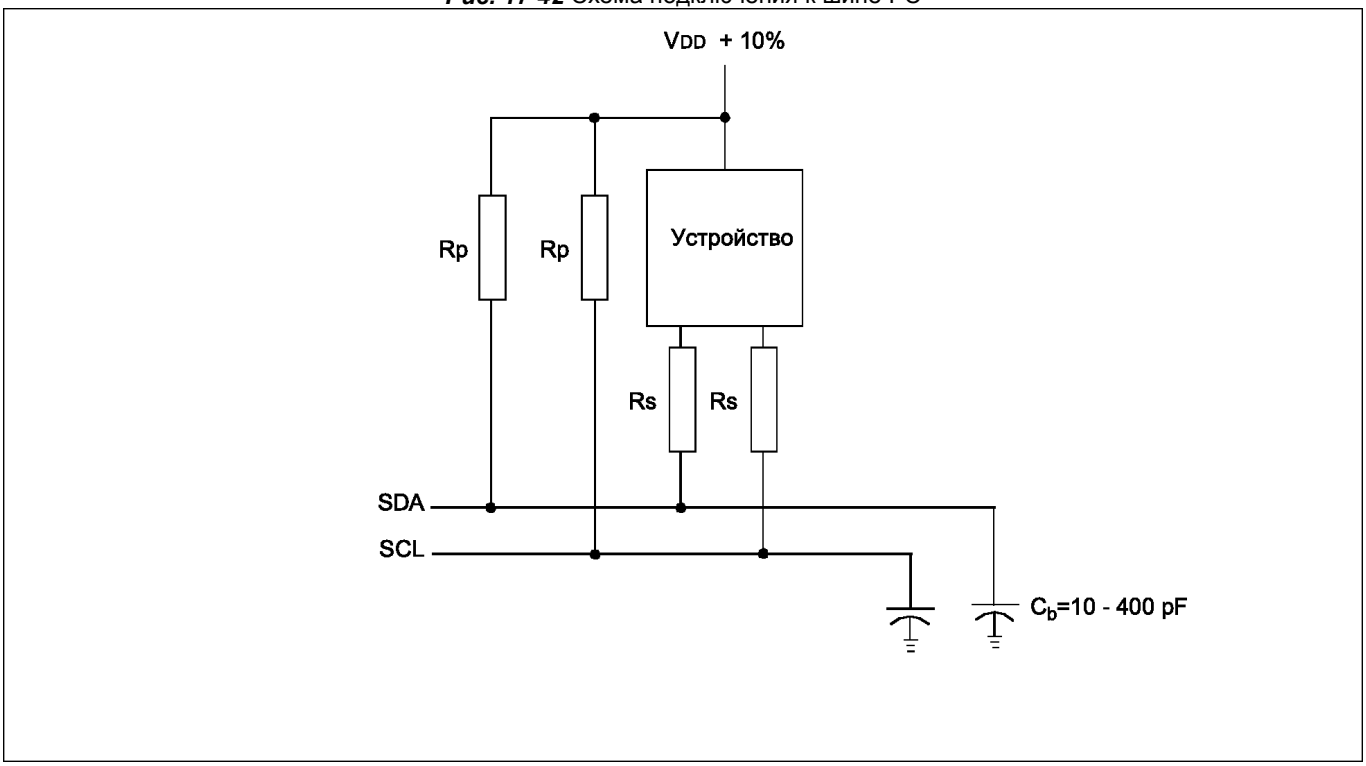
Максимальное значение Rs определяется допустимым уровнем шума.

Емкость шины определяется суммарная емкостью проводников и выводов. Емкость определяет максимальное значение Rp из-за допустимой длительности фронта.

Бит SMP в регистре SSPSTAT включает управление длительностью фронта SDA и SCL для того, чтобы фронты этих сигналов удовлетворяли спецификации при работе в скоростном режиме с частотой 400 кГц.

Устройства на шине I^2C должны иметь один источник питания, к которому подключаются подтягивающие резисторы.

Рис. 17-42 Схема подключения к шине I^2C



17.6 Инициализация

Пример 17-2 Инициализация модуля MSSP в режиме ведущего SPI

```
CLRF    STATUS          ; Банк 0
CLRF    SSPSTAT         ; SMP = 0, CKE = 0, и сбросить биты статуса
BSF     SSPSTAT, CKE    ; CKE = 1
MOVLW   0x31            ; Установить режим ведущего SPI, CLK/16,
MOVWF   SSPCON          ; сдвиг данных по заднему фронту (CKE=1 & CKP=1)
                        ; Выборка данных в середине такта (SMP=0 & режим ведущего)
BSF     STATUS, RP0     ; Банк 1
BSF     PIE, SSPIE      ; Разрешить прерывания от MSSP модуля
BCF     STATUS, RP0     ; Банк 0
BSF     INTCON, GIE     ; Разрешить прерывания
MOVLW   DataByte        ; Получить байт передаваемых данные из памяти
MOVWF   SSPBUF          ; Начать передачу байта данных
```

17.6.1 Совместимость модуля MSSP и основного модуля SSP (BSSP)

В модуле MSSP (по сравнению с BSSP) в регистре SSPSTAT содержится два дополнительных служебных бита, которые используются только в режиме SPI:

- SMP - управление выборкой данных в режиме SPI;
- CKE - выбор активного фронта тактового сигнала в режиме SPI.

Для обеспечения совместимости модулей MSSP и BSSP эти биты должны находиться в состоянии, показанном в таблице 17-4. Если не выдержать требования таблицы 17-4, данные передаваемые по интерфейсу SPI могут быть искажены.

Таблица 17-4 Требования к состоянию служебных битов для совместимости MSSP и BSSP модулей

Модуль BSSP	**Модуль MSSP**		
CKP	**CKP**	**CKE**	**SMP**
1	1	0	0
0	0	0	0

17.7 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Не могу организовать обмен данными с другим устройством, работающим по интерфейсу SPI.
Ответ 1:
Необходимо гарантировать, что Вы выбрали правильный режим SPI для этого устройства. Модуль MSSP поддерживает все четыре режима SPI, вероятно Вы где-то ошиблись. Проверьте полярность тактового сигнала и выборку данных.

Вопрос 2: В режиме ведомого I^2C не могу передать данные, хотя запись в регистр SSPBUF выполняю.
Ответ 2:
После записи в SSPBUF необходимо установить в '1' бит CKP, чтобы "отпустить" тактовый сигнал I^2C.

17.8 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с модулем MSSP в микроконтроллерах PICmicro MCU:

Документ	Номер
Use of the SSP Module in the I 2 C Multi-Master Environment Использование модуля SSP в режиме ведущего I^2C с конкуренцией на шине	AN578
Using Microchip 93 Series Serial EEPROMs with Microcontroller SPI Ports Использование интерфейса SPI для связи с последовательной памятью EEPROM серии 93	AN613
Interfacing PIC16C64/74 to Microchip SPI Serial EEPROM Подключение к PIC16C64/74 последовательной EEPROM памяти с интерфейсом SPI	AN647
Interfacing a Microchip PIC16C92x to Microchip SPI Serial EEPROM Подключение к PIC16C92x последовательной EEPROM памяти с интерфейсом SPI	AN668

Раздел 18. Модуль USART

Содержание

18.1 Введение

USART – это один из модулей последовательного порта ввода/вывода (имеет существенные отличия от модуля SSP), который может работать в полнодуплексном асинхронном режиме для связи с терминалами, персональными компьютерами или синхронном полудуплексном режиме для связи с микросхемами ЦАП, АЦП, последовательными EEPROM и т.д.

USART может работать в одном из трех режимов:

- Асинхронный, полный дуплекс;
- Ведущий синхронный, полудуплекс;
- Ведомый синхронный, полудуплекс.

Биты SPEN (RCSTA<7>) и TRIS должны быть установлены в ‘1’ для использования выводов TX/CK и RX/DT в качестве портов универсального синхронно-асинхронного приемопередатчика. Модуль USART поддерживает режим детектирования 9-разрядного адреса для работы в сетевом режиме.

Примечание 1. Модули USART в некоторых микроконтроллерах не поддерживают режим детектирования адреса. Смотрите техническую документацию на микроконтроллер.

Примечание 2. Описание режима детектирования адреса не входит в оригинальную техническую документацию DS33023A, оно взято из документов DS40300B и DS30292C.

18.2 Регистры управления

TXSTA: Регистр управления и статуса передатчика

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R-1	R/W-0
CSRC	**TX9**	**TXEN**	**SYNC**	**-**	**BRGH**	**TRMT**	**TX9D**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **CSRC:** Выбор источника тактового сигнала
Синхронный режим
1 = ведущий, внутренний тактовый сигнал от BRG
0 = ведомый, внешний тактовый сигнал с входа CK

Асинхронный режим
Не имеет значения

бит 6: **TX9:** Разрешение 9-разрядной передачи
1 = 9-разрядная передача
0 = 8-разрядная передача

бит 5: **TXEN:** Разрешение передачи
1 = разрешена
0 = запрещена
Примечание. В синхронном режиме биты SREN/CREN отменяют действие бита TXEN.

бит 4: **SYNC:** Режим работы USART
1 = синхронный
0 = асинхронный

бит 3: **Не используется:** читается как ‘0’

бит 2: **BRGH:** Выбор высокоскоростного режима
Синхронный режим
Не имеет значения

Асинхронный режим
1 = высокоскоростной режим
0 = низкоскоростной режим

бит 1: **TRMT:** Флаг очистки сдвигового регистра передатчика TSR
1 = TSR пуст
0 = TSR полон

бит 0: **TX9D:** 9-й бит передаваемых данных (может использоваться для программной проверки четности)

RCSTA: Регистр управления и статуса приемника

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	**RX9**	**SREN**	**CREN**	**ADDEN**	**FERR**	**OERR**	**RX9D**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **SPEN:** Разрешение работы последовательного порта
1 = модуль USART включен (выводы RX/DT, TX/CK подключены к USART)
0 = модуль USART выключен

18....1.1

бит 6: **RX9:** Разрешение 9-разрядного приема
1 = 9-разрядный прием
0 = 8-разрядный прием

бит 5: **SREN:** Разрешение одиночного приема
Синхронный режим
1 = разрешен одиночный прием
0 = запрещен одиночный прием
Сбрасывается в ‘0’ по завершению приема.
Примечание. В режиме ведомого не имеет значения

Асинхронный режим
Не имеет значения

бит 4: **CREN:** Разрешение приема
Синхронный режим
1 = прием разрешен (при установке бита CREN автоматически сбрасывается бит SREN)
0 = прием запрещен

Асинхронный режим
1 = прием разрешен
0 = прием запрещен

бит 3: **ADDEN:** Разрешение детектирования адреса(1)
Асинхронный 9-разрядный прием (RX9=1)
1 = детектирование адреса разрешено. Если бит RSR<8>=1, то генерируется прерывание и загружается приемный буфер.
0 = детектирование адреса запрещено. Принимаются все байты, девятый бит может использоваться для проверки четности.

Асинхронный 8-разрядный прием (RX9=0)
Не имеет значения

Синхронный режим
Не имеет значения

бит 2: **FERR:** Ошибка кадра, сбрасывается при чтении регистра RCREG
1 = произошла ошибка кадра
0 = ошибки кадра не было

бит 1: **OERR:** Ошибка переполнения внутреннего буфера, устанавливается в ‘0’ при сбросе бита CREN
1 = произошла ошибка переполнения
0 = ошибки переполнения не было

бит 0: **RX9D:** 9-й бит принятых данных (может использоваться для программной проверки четности)

Примечание 1. Модули USART в некоторых микроконтроллерах не поддерживают режим детектирования адреса. Смотрите техническую документацию на микроконтроллер.

18.3 Генератор частоты обмена USART BRG

BRG используется для работы USART в синхронном ведущем и асинхронном режимах. BRG представляет собой отдельный 8-разрядный генератор скорости обмена в бодах, период которого определяется значением в регистре SPBRG. В асинхронном режиме бит BRGH (TXSTA<2>) тоже влияет на скорость обмена (в синхронном режиме бит BRGH игнорируется). В таблице 18-1 указаны формулы для вычисления скорости обмена в бодах при различных режимах работы модуля USART (относительно внутреннего тактового сигнала микроконтроллера).

Учитывая требуемую скорость и F_{OSC}, выбирается самое близкое целое значение для записи в регистр SPBRG (от 0 до 255), рассчитанное по формулам приведенным в таблице 18-1. Затем рассчитывается ошибка скорости обмена.

Таблица 18-1 Формулы расчета скорости обмена данными

SYNC	BRGH = 0	BRGH = 1
0	(Асинхронный) Скорость обмена = F_{OSC} / (64 (X + 1))	(Асинхронный) Скорость обмена = F_{OSC} / (16 (X + 1))
1	(Синхронный) Скорость обмена = F_{OSC} / (4 (X + 1))	(Синхронный) Скорость обмена = F_{OSC} / (4 (X + 1))

X = значение регистра SPBRG (от 0 до 255)

В примере 18-1 показан расчет значения для регистра SPBRG и погрешность скорости обмена для следующих условий:

F_{OSC} = 16 МГц;
Скорость приема/передачи данных = 9600 бит/с;
BRGH = 0;
SYNC = 0.

Пример 18-1 Расчет значения для регистра SPBRG и погрешность скорости обмена

Желаемое значение скорости = F_{OSC} / (64 (X + 1))

9600 = 16 000 000 / (64 (X + 1))

X = [25.042] = 25

Вычисленное значение скорости = 16 000 000 / (64 (25 + 1)) = 9615

Ошибка = 100 x (Вычисленное – Желаемое) / Желаемое значение скорости

Ошибка = 100 x (9615 – 9600) / 9600 = 0.16%

В некоторых случаях может быть выгодно использовать высокоскоростной режим работы USART (BRGH=1), поскольку уравнение FOSC / (16 (X + 1)) позволяет уменьшить погрешность скорости.

Запись нового значения в регистр SPBRG сбрасывает таймер BRG, гарантируя немедленный переход на новую скорость.

Таблица 18-2 Регистры и биты, связанные с работой генератора BRG

Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Сброс POR, BOR	Другие сбросы
TXSTA	CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D	0000 -010	0000 -010
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
SPBRG	Регистр генератора скорости USART								0000 0000	0000 0000

Обозначения: - = не используется, читается как 0; u = не изменяется; x = не известно; q = зависит от условий.
Затененные биты на работу не влияют.

Таблица 18-3 Скорость обмена в синхронном режиме

Скорость обмена (К)	F_{OSC} = 20 МГц			F_{OSC} = 16 МГц			F_{OSC} = 10 МГц		
	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)
0,3	Нет	-	-	Нет	-	-	Нет	-	-
1,2	Нет	-	-	Нет	-	-	Нет	-	-
2,4	Нет	-	-	Нет	-	-	Нет	-	-
9,6	Нет	-	-	Нет	-	-	9,166	+1,73	255
19,2	19,53	+1,73	255	19,23	+0,16	207	19,23	+0,16	129
76,8	76,92	+0,16	64	76,92	+0,16	51	7576	-1,36	32
96	96,15	+0,16	51	95,24	-0,79	41	96,15	+0,16	25
300	294,1	-1,96	16	307,69	+2,56	12	312,5	+4,17	7
500	500	0	9	500	0	7	500	0	4
Максим.	5000	-	0	4000	-	0	2500	-	0
Миним.	19,53	-	255	15,625	-	255	9,766	-	255

Скорость обмена (К)	F_{OSC} = 7,15909 МГц			F_{OSC} = 5,0688 МГц			F_{OSC} = 4 МГц		
	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)
0,3	Нет	-	-	Нет	-	-	Нет	-	-
1,2	Нет	-	-	Нет	-	-	Нет	-	-
2,4	Нет	-	-	Нет	-	-	Нет	-	-
9,6	9,622	+0,23	185	9,6	0	131	9,615	+0,16	103
19,2	19,24	+0,23	92	19,2	0	65	19,231	+0,16	51
76,8	77,82	+1,32	22	79,2	+3,13	15	76,923	+0,16	12
96	94,20	-1,88	18	97,48	+1,54	12	100	+4,17	9
300	298,3	-0,57	5	316,8	+5,60	3	Нет	-	-
500	Нет	-	-	Нет	-	-	Нет	-	-
Максим.	1789,8	-	0	1267	-	0	1000	-	0
Миним.	6,991	-	255	4,950	-	255	3,906	-	255

Скорость обмена (К)	F_{OSC} = 3,579545 МГц			F_{OSC} = 1 МГц			F_{OSC} = 32,768 кГц		
	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)
0,3	Нет	-	-	Нет	-	-	0,303	+1,14	26
1,2	Нет	-	-	1,202	+0,16	207	1,170	-2,48	6
2,4	Нет	-	-	2,404	+0,16	103	Нет	-	-
9,6	9,622	+0,23	92	9,615	+0,16	25	Нет	-	-
19,2	19,04	-0,83	46	19,24	+0,16	12	Нет	-	-
76,8	74,57	-2,90	11	83,34	+8,51	2	Нет	-	-
96	99,43	+3,57	8	Нет	-	-	Нет	-	-
300	298,3	-0,57	2	Нет	-	-	Нет	-	-
500	Нет	-	-	Нет	-	-	Нет	-	-
Максим.	894,9	-	0	250	-	0	8,192	-	0
Миним.	3,496	-	255	0,9766	-	255	0,032	-	255

Таблица 18-4 Скорость обмена в асинхронном режиме (BRGH=0)

Скорость обмена (К)	F_{OSC} = 20 МГц			F_{OSC} = 16 МГц			F_{OSC} = 10 МГц		
	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)
0,3	Нет	-	-	Нет	-	-	Нет	-	-
1,2	1,221	+1,73	255	1,202	+0,16	207	1,202	+0,16	129
2,4	2,404	+0,16	129	2,404	+0,16	103	2,404	+0,16	64
9,6	9,469	-1,36	32	9,615	+0,16	25	9,615	+1,73	15
19,2	19,53	+1,73	15	19,23	+0,16	12	19,53	+1,73	7
76,8	78,13	+1,73	3	83,33	+8,51	2	78,13	+1,73	1
96	104,2	+8,51	2	Нет	-	-	Нет	-	-
300	312,5	+4,17	0	Нет	-	-	Нет	-	-
500	Нет	-	-	Нет	-	-	Нет	-	-
Максим.	312,5	-	0	250	-	0	156,3	-	0
Миним.	1,221	-	255	0,977	-	255	0,6104	-	255

Скорость обмена (К)	F_{OSC} = 7,15909 МГц			F_{OSC} = 5,0688 МГц			F_{OSC} = 4 МГц		
	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)
0,3	Нет	-	-	0,31	+3,13	255	0,3005	-0,17	207
1,2	1,203	+0,23	92	1,2	0	65	1,202	+1,67	51
2,4	2,380	-0,83	46	2,4	0	32	2,404	+1,67	25
9,6	9,322	-2,90	11	9,9	+3,13	7	Нет	-	-
19,2	18,64	-2,90	5	19,8	+3,13	3	Нет	-	-
76,8	Нет	-	-	79,2	+3,13	0	Нет	-	-
96	Нет	-	-	Нет	-	-	Нет	-	-
300	Нет	-	-	Нет	-	-	Нет	-	-
500	Нет	-	-	Нет	-	-	Нет	-	-
Максим.	111,9	-	0	79,2	-	0	62,500	-	0
Миним.	0,437	-	255	0,3094	-	255	3,906	-	255

Скорость обмена (К)	F_{OSC} = 3,579545 МГц			F_{OSC} = 1 МГц			F_{OSC} = 32,768 кГц		
	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)
0,3	0,301	+0,23	185	0,300	+0,16	51	0,256	-14,67	1
1,2	1,190	-,083	46	1,202	+0,16	12	Нет	-	-
2,4	2,432	+1,32	22	2,232	-6,99	6	Нет	-	-
9,6	9,322	-2,90	5	Нет	-	-	Нет	-	-
19,2	18,64	-2,90	2	Нет	-	-	Нет	-	-
76,8	Нет	-	-	Нет	-	-	Нет	-	-
96	Нет	-	-	Нет	-	-	Нет	-	-
300	Нет	-	-	Нет	-	-	Нет	-	-
500	Нет	-	-	Нет	-	-	Нет	-	-
Максим.	55,93	-	0	15,63	-	0	0,512	-	0
Миним.	0,2185	-	255	0,0610	-	255	0,0020	-	255

Таблица 18-5 Скорость обмена в асинхронном режиме (BRGH=1)

Скорость обмена (К)	F_{OSC} = 20 МГц			F_{OSC} = 16 МГц			F_{OSC} = 10 МГц		
	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)
9,6	9,615	+0,16	129	9,615	+0,16	103	9,615	+0,16	64
19,2	19,230	+0,16	64	19,230	+0,16	51	18,939	-1,36	32
38,4	37,878	-1,36	32	38,461	+0,16	25	39,062	+1,7	15
57,6	56,818	-1,36	21	58,823	+2,12	16	56,818	-1,36	10
115,2	113,636	-1,36	10	111,111	-3,55	8	125	+8,51	4
250	250	0	4	250	0	3	Нет	-	-
625	625	0	1	Нет	-	-	625	0	0
1250	1250	0	0	Нет	-	-	Нет	-	-

Скорость обмена (К)	F_{OSC} = 7,16 МГц			F_{OSC} = 5,068 МГц			F_{OSC} = 4 МГц		
	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)
9,6	9,520	-0,83	46	9,6	0	32	Нет	-	-
19,2	19,454	+1,32	22	18,645	-2,94	16	1,202	+0,17	207
38,4	37,286	-2,90	11	39,6	+3,12	7	2,403	+0,13	103
57,6	55,930	-2,90	7	52,8	-8,33	5	9,615	+0,16	25
115,2	111,860	-2,90	3	105,6	-8,33	2	19,231	+0,16	12
250	Нет	-	-	Нет	-	-	Нет	-	-
625	Нет	-	-	Нет	-	-	Нет	-	-
1250	Нет	-	-	Нет	-	-	Нет	-	-

Скорость обмена (К)	F_{OSC} = 3,579 МГц			F_{OSC} = 1 МГц			F_{OSC} = 32,768 кГц		
	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)	Реальная скорость	Ошибка %	Значение SPBRG (десят.)
9,6	9,727	+1,32	22	8,928	-6,99	6	Нет	-	-
19,2	18,643	-2,90	11	20,833	+8,51	2	Нет	-	-
38,4	37,286	-2,90	5	31,25	-18,61	1	Нет	-	-
57,6	55,930	-2,90	3	62,5	+8,51	0	Нет	-	-
115,2	111,860	-2,90	1	Нет	-	-	Нет	-	-
250	223,721	-10,51	0	Нет	-	-	Нет	-	-
625	Нет	-	-	Нет	-	-	Нет	-	-
1250	Нет	-	-	Нет	-	-	Нет	-	-

18.4 Асинхронный режим USART

В этом режиме USART использует стандартный формат NRZ: один стартовый бит, восемь или девять битов данных и один стоповый бит. Наиболее часто встречается 8-разрядный формат передачи данных. Интегрированный 8-разрядный генератор BRG позволяет получить стандартные скорости передачи данных. Генератор скорости обмена может работать в одном из двух режимов: высокоскоростной (x16 BRGH=1 TXSTA<2>), низкоскоростной (x64 BRGH=0 TXSTA<2>). Приемник и передатчик последовательного порта работают независимо друг от друга, но используют один и тот же формат данных и одинаковую скорость обмена. Бит четности аппаратно не поддерживается, но может быть реализован программно, применяя 9-разрядный формат данных. Все данные передаются младшим битом вперед. В SLEEP режиме микроконтроллера модуль USART(асинхронный режим) выключен.

Выбор асинхронного режима USART выполняется сбросом бита SYNC в ‘0’ (TXSTA<4>).

Модуль USART в асинхронном режиме состоит из следующих элементов:

- Генератор скорости обмена;
- Цепь опроса;
- Асинхронный передатчик;
- Асинхронный приемник.

18.4.1 Асинхронный передатчик USART

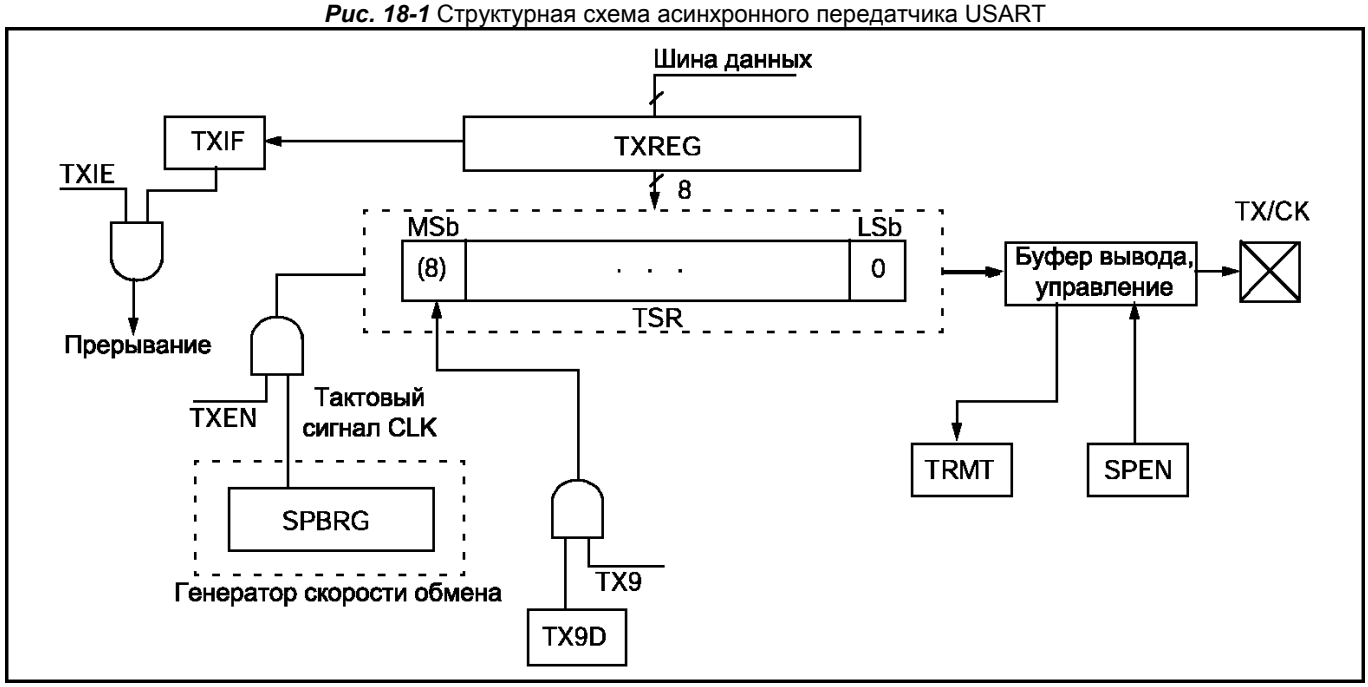
Структурная схема асинхронного передатчика USART показана на рисунке 18-1. Главным в передатчике является сдвиговый регистр TSR, который получает данные из буфера передатчика TXREG. Данные в регистр TXREG загружаются программно. После передачи стопового бита предыдущего байта, в последнем машинном такте цикла BRG, TSR загружается новым значением из TXREG (если оно присутствует), после чего устанавливается флаг прерывания TXIF. Прерывание может быть разрешено или запрещено битом TXIE. Флаг TXIF устанавливается независимо от состояния бита TXIE и не может быть сброшен в ‘0’ программно. Очистка флага TXIF происходит только после загрузки новых данных в регистр TXREG. Аналогичным образом бит TRMT (TXSTA<1>) отображает состояние регистра TSR. Бит TRMT доступен только на чтение и не может вызвать генерацию прерывания.

Примечание 1. Регистр TSR не отображается на память и не доступен для чтения.
Примечание 2. Флаг TXIF устанавливается в ‘1’ только, когда бит TXEN=1 и сбрасывается автоматически в '0' после загрузки новых данных в регистр TXREG.

Для разрешения передачи необходимо установить бит TXEN (TXSTA<5>) в ‘1’. Передача данных не начнется до тех пор, пока в TXREG не будут загружены новые данные; не придет очередной тактовый импульс от генератора BRG (см рисунок 18-2). Можно сначала загрузить данные в TXREG, а затем установить бит TXEN. Как правило, после разрешения передачи регистр TSR пуст, таким образом, данные записываемые в TXREG сразу передаются в TSR, а TXREG остается пустым. Это позволяет реализовать слитную передачу данных (см. рисунок 18-3). Сброс бита TXEN в ‘0‘ вызовет немедленное прекращение передачи, сброс передатчика и перевод вывода TX/CK в третье состояние.

Для разрешения 9-разрядной передачи необходимо установить бит TX9 (TXSTA<6>) в ‘1’. Девятый бит данных записывается в бит TX9D (TXSTA<0>). Девятый бит данных должен быть указан до записи в регистр TXREG, потому что данные, записанные в регистр TXREG, могут быть сразу загружены в сдвиговый регистр TSR (если он пуст).

Рис. 18-1 Структурная схема асинхронного передатчика USART



Рекомендованная последовательность действий для передачи данных в асинхронном режиме:

1. Установить требуемую скорость передачи с помощью регистра SPBRG и бита BRGH (см. раздел 18.3).
2. Выбрать асинхронный режим сбросом бита SYNC в ‘0’ и установкой бита SPEN в ‘1’.
3. Если необходимо, разрешить прерывания установкой битов TXIE, PEIE, GIE в ‘1’.
4. Если передача 9-разрядная, установить бит TX9 в ‘1’.
5. Разрешить передачу установкой бита TXEN в ‘1’, автоматически устанавливается флаг TXIF.
6. Если передача 9-разрядная, записать 9-й бит данных в TX9D.
7. Записать данные в регистр TXREG (начало передачи данных).

Рис. 18-2 Временная диаграмма асинхронной передачи данных

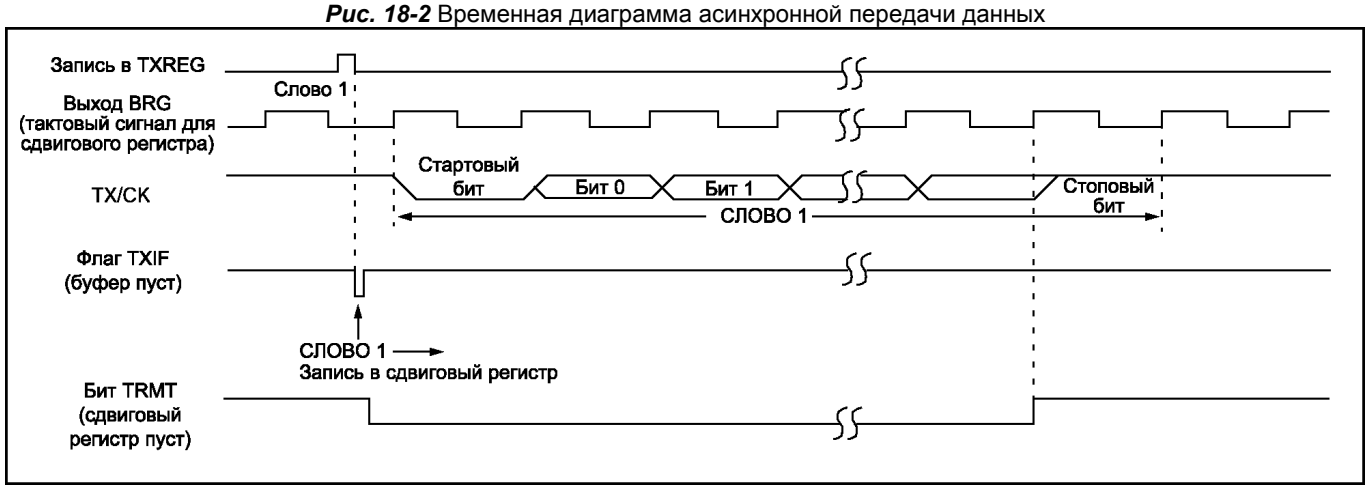


Рис. 18-3 Временная диаграмма слитной асинхронной передачи (последовательная передача двух байт)

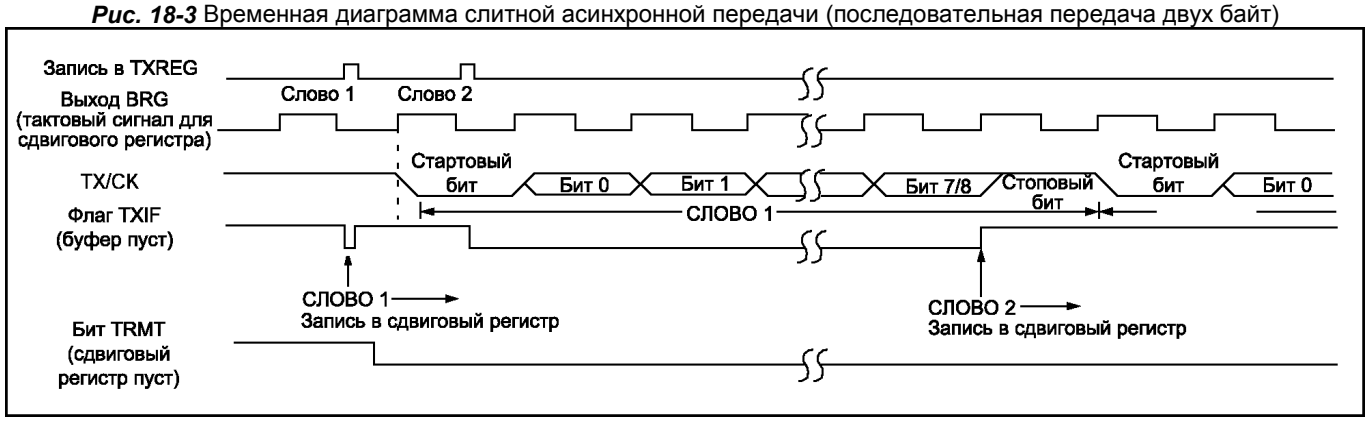


Таблица 18-6 Регистры и биты, связанные с работой передатчика USART в асинхронном режиме

Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Сброс POR, BOR	Другие сбросы
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000u
PIR	TXIF(1)								0	0
PIE	TXIE(1)								0	0
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
TXREG	Регистр данных передатчика USART								0000 0000	0000 0000
TXSTA	CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D	0000 –010	0000 –010
SPBRG	Регистр генератора скорости USART								0000 0000	0000 0000

Обозначения: - = не используется, читается как 0; u = не изменяется; x = не известно; q = зависит от условий.
Затененные биты на работу не влияют.

Примечания:

1. Расположение битов смотрите в технической документации на микроконтроллер.
2. В некоторых микроконтроллерах эти биты могут обозначаться как GPIE и GPIF.

18.4.2 Асинхронный приемник USART

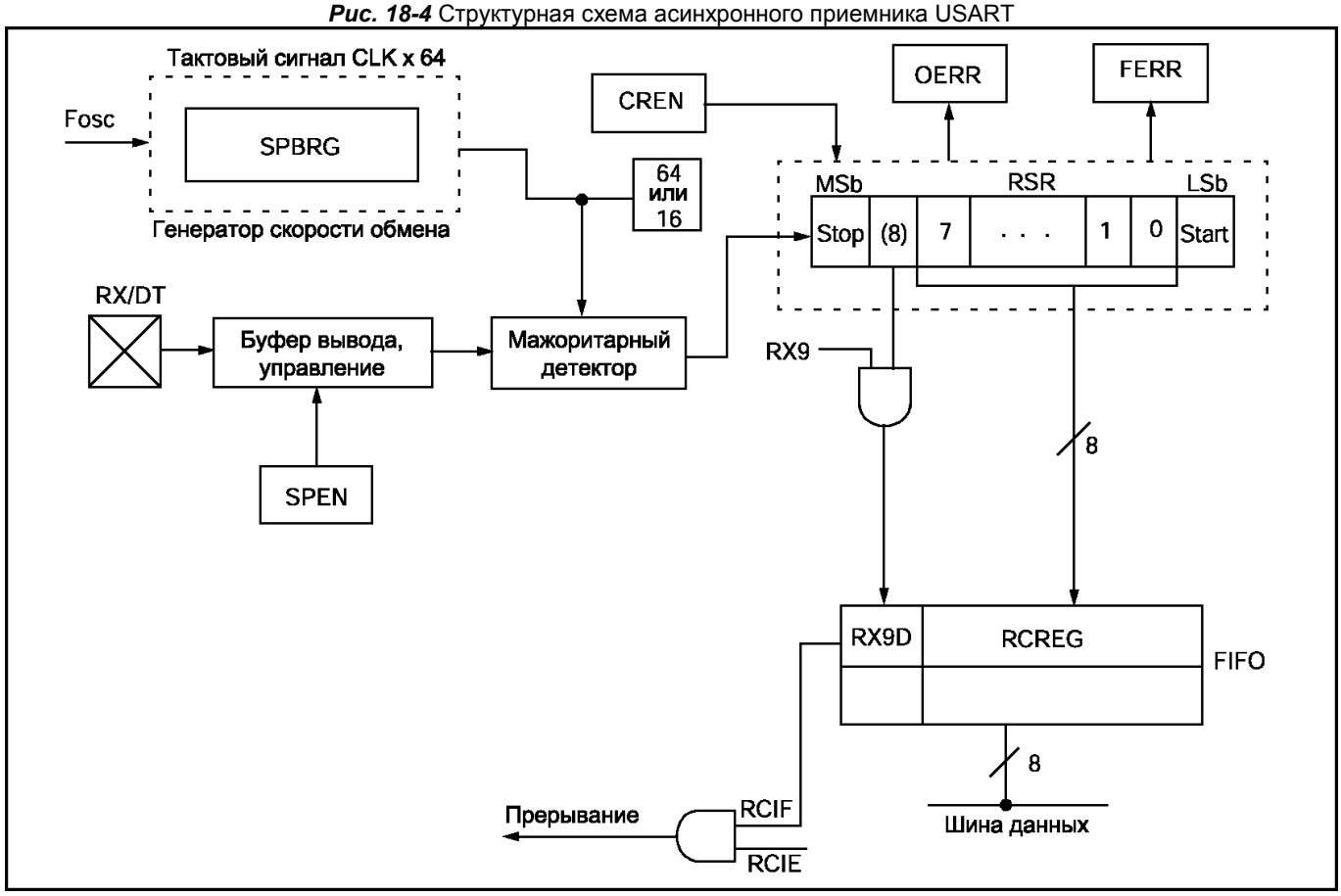
Структурная схема асинхронного приемника USART показана на рисунке 18-4. Данные подаются на вход RX/DT в блок восстановления данных, представляющий собой скоростной сдвиговый регистр, работающий на частоте в 16 раз превышающей скорость передачи или F_{OSC}.

Включение приемника производится установкой бита CREN (RCSTA<4>) в ‘1’.

Главным в приемнике является сдвиговый регистр RSR. После получения стопового бита данные переписываются в регистр RCREG, если он пуст. После записи в регистр RCREG выставляется флаг прерывания RCIF. Прерывание можно разрешить/запретить установкой/сбросом бита RCIE. Флаг RCIF доступен только на чтение, сбрасывается аппаратно при чтении из регистра RCREG. Регистр RCREG имеет двойную буферизацию, т.е. представляет собой двухуровневый буфер FIFO. Поэтому можно принять 2 байта данных в FIFO RCREG и третий в регистр RSR. Если FIFO заполнен и обнаружен стоповый бит третьего байта, устанавливается бит переполнения приемника OERR (RCSTA<1>). Байт, принятый в RSR, будет потерян. Для извлечения двух байт из FIFO необходимо дважды прочитать регистр RCREG. Бит OERR нужно программно очистить сбросом бита CREN, т.е. запрещением приема. В любом случае, если бит OERR установлен, логика приемника выключена.

Бит ошибки кадра FERR (RCSTA<2>) устанавливается в ‘1’, если не обнаружен стоповый бит. FERR и девятый бит принятых данных буферизируются так же, как принятые данные. Рекомендуется сначала прочитать регистр RCSTA, затем RCREG, чтобы не потерять информацию RX9D и FERR.

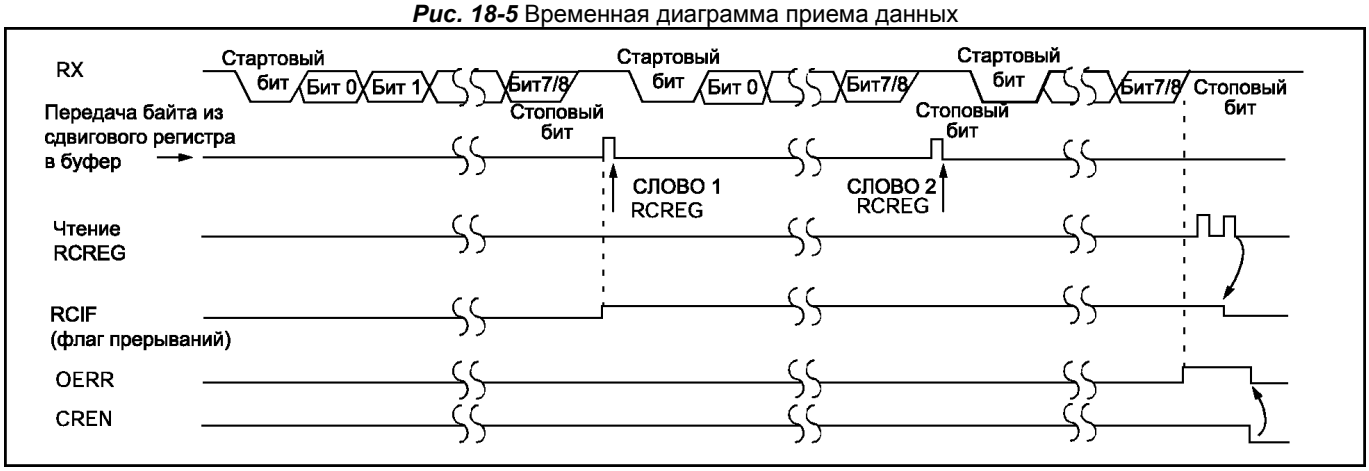
Рис. 18-4 Структурная схема асинхронного приемника USART



Рекомендованные действия при приеме данных в асинхронном режиме:

1. Установить требуемую скорость передачи с помощью регистра SPBRG и бита BRGH (см. раздел 18.3).
2. Выбрать асинхронный режим сбросом бита SYNC в ‘0’ и установкой бита SPEN в ‘1’.
3. Если необходимо, разрешить прерывания установкой битов RCIE, PEIE и GIE в ‘1’.
4. Если прием 9-разрядный, установить бит RX9 в ‘1’.
5. Разрешить прием установкой бита CREN в ‘1’.
6. Ожидать установку бита RCIF, или прерывание, если оно разрешено битом RCIE.
7. Считать 9-й бит данных (если разрешен 9-разрядный прием) из регистра RCSTA и проверить возникновение ошибки.
8. Считать 8 бит данных из регистра RCREG.
9. При возникновении ошибки переполнения сбросить бит CREN в ‘0’.

Рис. 18-5 Временная диаграмма приема данных



Примечание. На временной диаграмме показан последовательный прием трех байт. Регистр RCREG (приемный буфер) читается после приема трех байт, поэтому устанавливается бит OERR в '1'.

18.4.3 Настройка 9-разрядного асинхронного приема с детектированием адреса

Примечание. Модули USART в некоторых микроконтроллерах не поддерживают режим детектирования адреса. Смотрите техническую документацию на микроконтроллер.

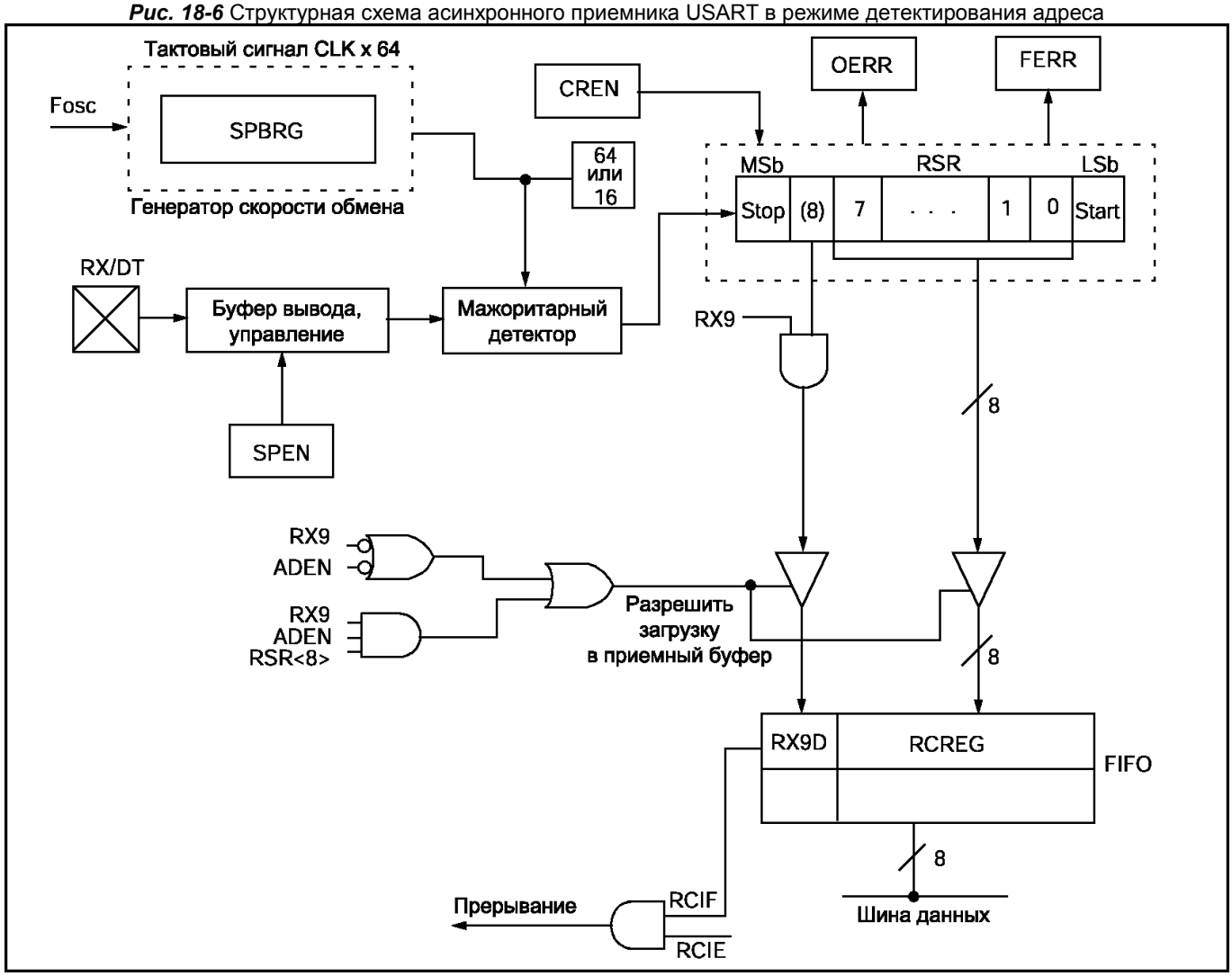
Если установлен бит RX9 в регистре RCSTA, 9-й бит принимаемых данных сохраняется в бите RX9D регистра RCSTA. Модуль USART имеет функцию детектирование адреса для организации сетевого обмена. Детектирование адреса разрешено, если установлены биты ADDEN(RCSTA<3>) и RX9 (RCSTA<6>) в ‘1’. В этом режиме принимаемые данные записываются в регистр RCREG (9-бит в RX9D регистра RCSTA), если девятый бит принимаемых данных равен 1.

Для передачи данных в сетевой структуре ведущее устройство должно сначала передать адрес ведомого устройства. В байте адреса 9 бит (RSR<8>) равен 1 (в байте данных RSR<8>=0). Если в регистре RCSTA биты RX9 и ADDEN установлены в ‘1’, то разрешено детектирование адреса, все байты данных будут игнорироваться. Однако если 9 бит принятых данных будет равен 1 (принят адресный байт), содержимое регистра RSR передается в приемный буфер. Это позволяет ведомому устройству обрабатывать только адресные байты. Если принятый адресный байт соответствует адресу ведомого устройства, необходимо сбросить бит ADDEN в ‘0’ для перехода в режим приема данных.

Когда бит ADDEN = 1, все принимаемые байты данных игнорируются. После приема стопового бита, данные не загружаются в приемный буфер, прерывание не генерируется. Если принят следующий байт, предыдущий байт в регистре RSR будет потерян.

Детектирование адреса разрешено, если только включен 9-разрядный прием данных (RX9 = 1). Если детектирование адреса запрещено (ADDEN = 0), девятый бит принимаемых данных может использоваться для контроля четности.

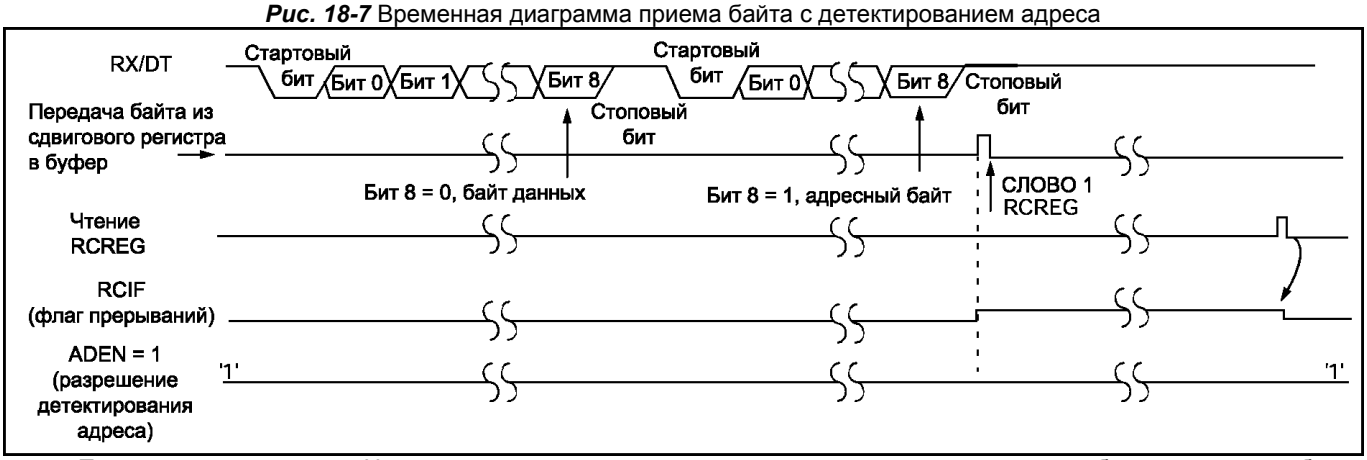
Рис. 18-6 Структурная схема асинхронного приемника USART в режиме детектирования адреса



Рекомендованная последовательность действия при использовании детектора адреса:

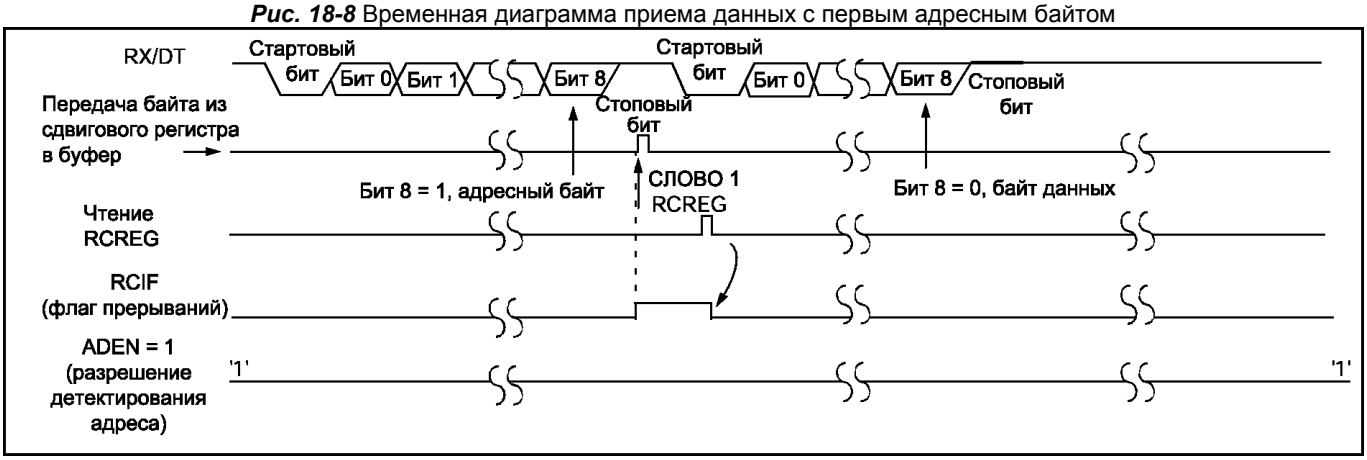
1. Установить требуемую скорость передачи с помощью регистра SPBRG и бита BRGH (см. раздел 18.3).
2. Выбрать асинхронный режим сбросом бита SYNC в ‘0’ и установкой бита SPEN в ‘1’.
3. Если необходимо, разрешить прерывания установкой битов RCIE, PEIE и GIE в ‘1’.
4. Установить бит RX9 в ‘1’ для включения 9-разрядного приема.
5. Установить бит ADDEN в ‘1’ для разрешения детектирования адреса.
6. Разрешить прием установкой бита CREN в ‘1’.
7. Ожидать установку бита RCIF или прерывание, если оно разрешено битом RCIE.
8. Считать 8 бит данных из регистра RCREG для проверки адресации устройства.
9. При возникновении ошибки переполнения сбросить бит CREN в ‘0’.
10. Если принятый адрес соответствует адресу устройства, сбросить биты ADDEN и RCIF в ‘0’ для начала приема данных.

Рис. 18-7 Временная диаграмма приема байта с детектированием адреса



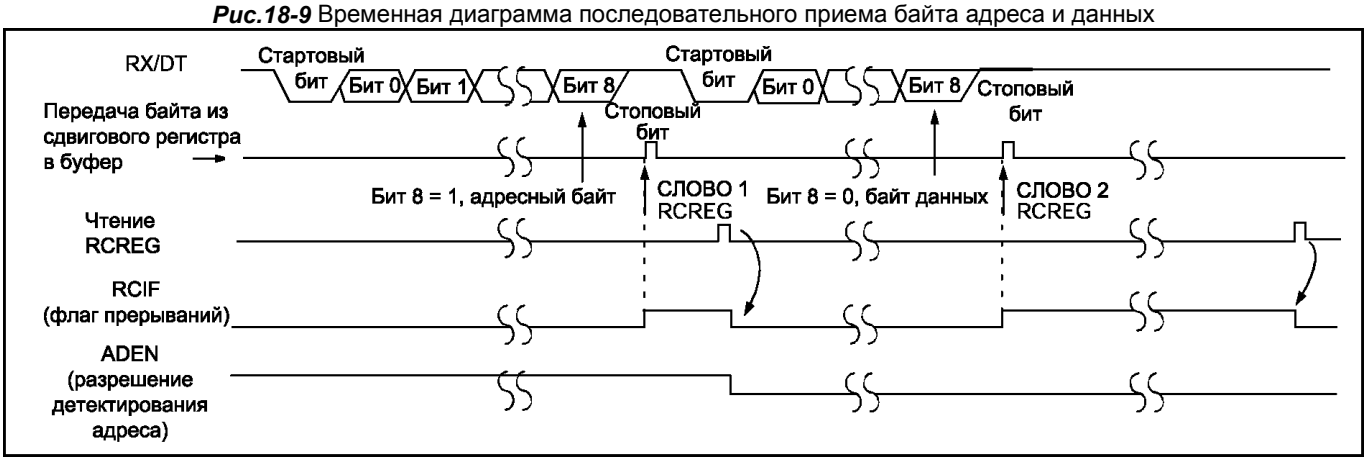
Примечание к рисунку. На временной диаграмме показан последовательный прием байта данных и байта адреса. Байт данных не записывается в RCREG, т.к. ADDEN=1, а бит8 = 0.

Рис. 18-8 Временная диаграмма приема данных с первым адресным байтом



Примечание к рисунку. На временной диаграмме показан последовательный прием байта адреса и байта данных. Байт данных не записывается в RCREG, т.к. ADDEN не был сброшен в ‘0’ (ADDEN=1), а бит8 = 0.

Рис.18-9 Временная диаграмма последовательного приема байта адреса и данных

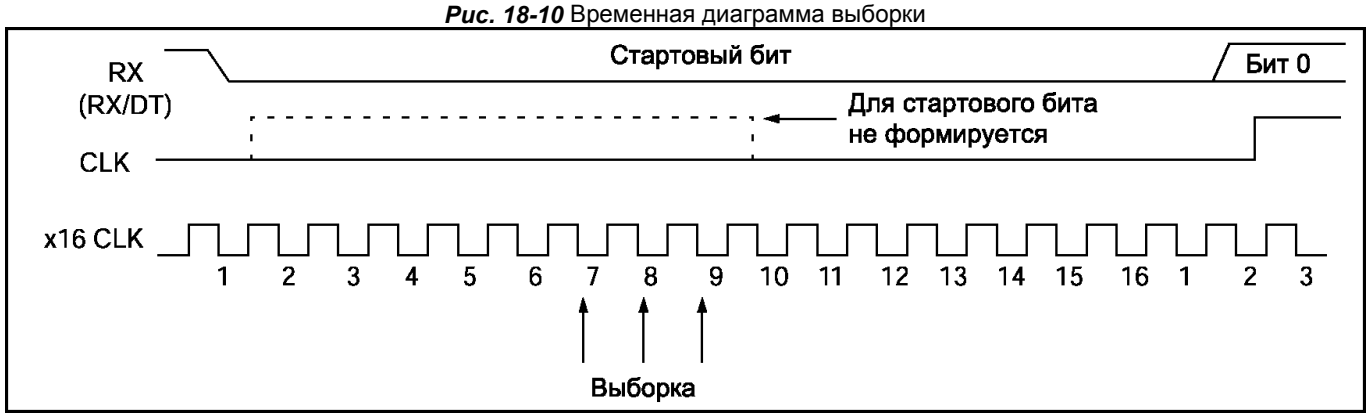


Примечание к рисунку. На временной диаграмме показан последовательный прием байта адреса и байта данных. Байт данных сохраняется в регистре RCREG, потому что бит ADDEN был сброшен в ‘0’ после совпадения адреса. Содержимое регистра RSR всегда будет передаваться в регистр RCREG независимо от состояния бита 8.

18.4.4 Выборка

Сигнал с входа RX/DT опрашивается цепью мажоритарного детектора три раза за такт передачи, чтобы определить, высокого или низкого уровня сигнал присутствует на входе. На рисунке 18-10 показана временная диаграмма выборки данных. Методика выборки одинакова для любого состояния бита BRGH (отличие в частоте тактового сигнала выборки данных x16).

Рис. 18-10 Временная диаграмма выборки



18.4.4.1 Исключения

Все новые микроконтроллеры будут иметь методику выборки данных, показанную на рисунке 18-10. Исключением являются следующие микроконтроллеры:

- PIC16C63
- PIC16C65
- PIC16C65A
- PIC16C73
- PIC16C73A
- PIC16C74
- PIC16C74A

Сигнал с входа RX/DT опрашивается цепью мажоритарного детектора три раза за такт передачи, чтобы определить высокого или низкого уровня сигнал присутствует на входе. Если выбран низкоскоростной режим (BRRH=0), то выборка производится по седьмому, восьмому и девятому заднему фронту тактового сигнала x16 (см. рисунок 18-11). Если BRGH = 1 (выбран высокоскоростной режим), выборка производится на втором такте сигнала x4 тремя запросами (см. рисунки 18-12, 18-13).

Рис. 18-11 Временная диаграмма выборки (BRGH = 0)

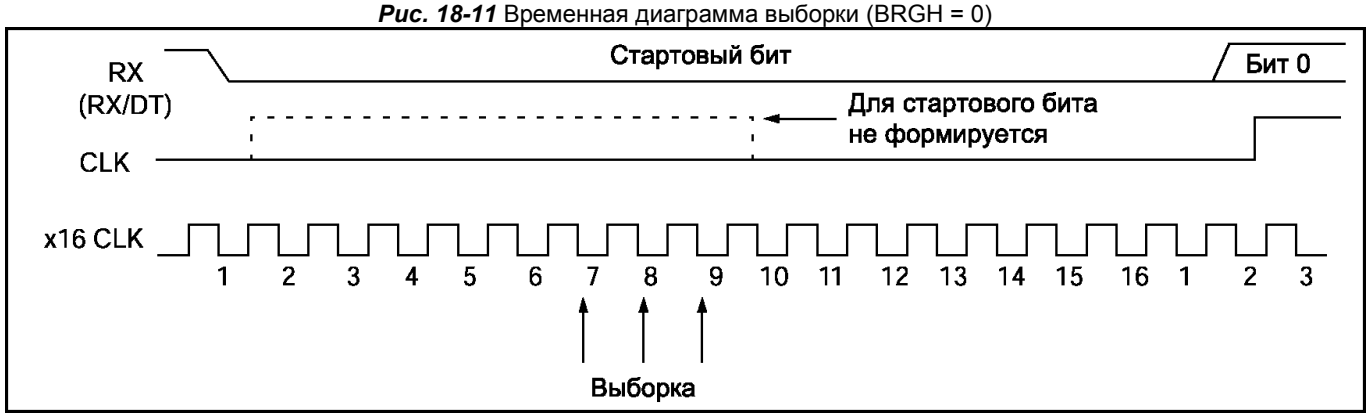


Рис. 18-12 Временная диаграмма выборки (BRGH = 1)

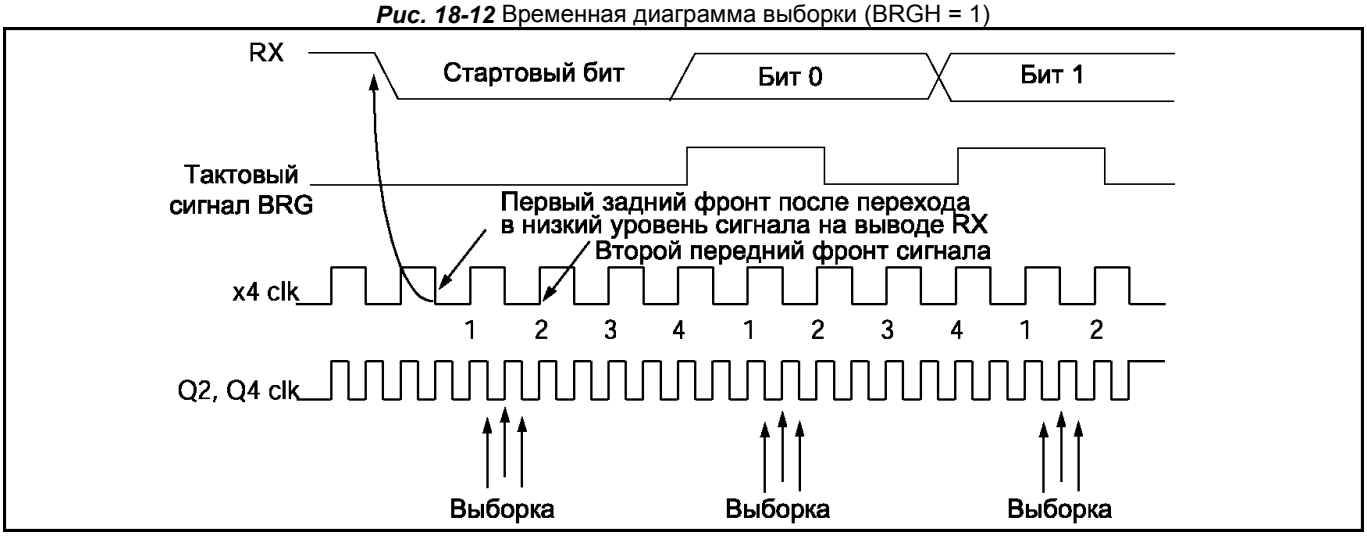


Рис. 18-13 Временная диаграмма выборки (BRGH = 1)

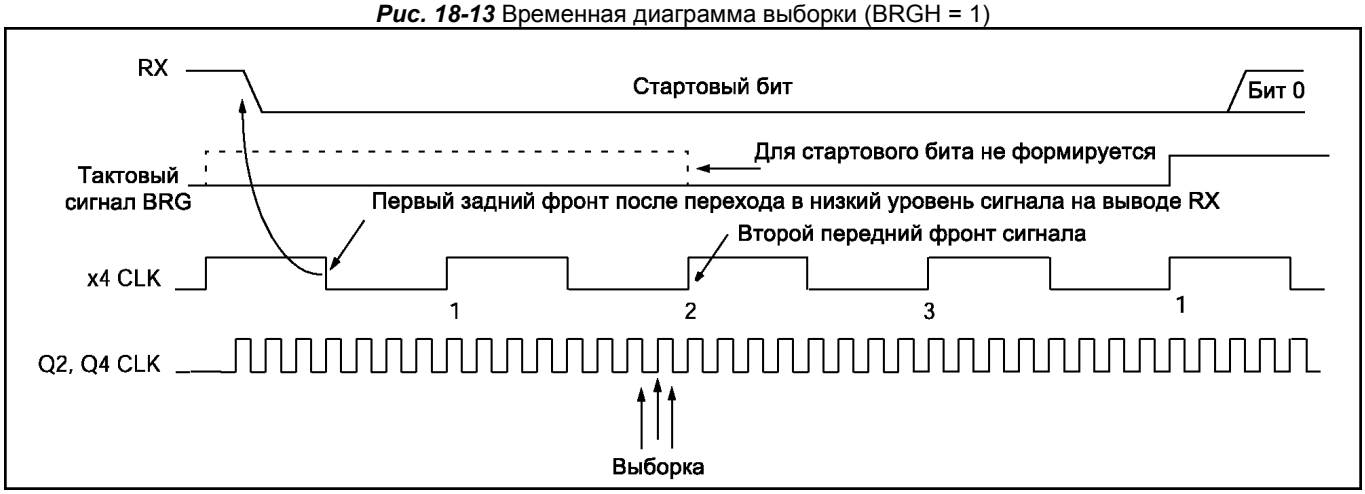


Таблица 18-7 Регистры и биты, связанные с работой приемника USART в асинхронном режиме

Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Сброс POR, BOR	Другие сбросы
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000u
PIR	RCIF(1)								0	0
PIE	RCIE(1)								0	0
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
RCREG	Регистр данных приемника USART								0000 0000	0000 0000
TXSTA	CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D	0000 –010	0000 –010
SPBRG	Регистр генератора скорости USART								0000 0000	0000 0000

Обозначения: - = не используется, читается как 0; u = не изменяется; x = не известно; q = зависит от условий.
Затененные биты на работу не влияют.
Примечания:

1. Расположение битов смотрите в технической документации на микроконтроллер.
2. В некоторых микроконтроллерах эти биты могут обозначаться как GPIE и GPIF.

18.5 Синхронный ведущий режим USART

В ведущем синхронном режиме данные передаются полудуплексом, т.е. прием и передача не происходит одновременно. При передаче запрещен прием и наоборот. Синхронный режим включается установкой бита SYNC (TXSTA<4>) в ‘1’. Также необходимо включить модуль USART, установкой бита SPEN (RCSTA<7>) в ‘1’, для настройки портов ввода/вывода TX/CK и RX/DT в качестве тактового сигнала CK и линии данных DT соответственно. В режиме ведущего модуль USART формирует тактовый сигнал CK. Выбор режима ведущего производится установкой бита CSRC (TXSTA<7>) в ‘1’.

18.5.1 Передача синхронного ведущего

Структурная схема передатчика USART показана на рисунке 18-1. Главным в передатчике является сдвиговый регистр TSR. Сдвиговый регистр получает данные из буфера передатчика TXREG. В регистр TSR не загружаются новые данные, пока не будет передан последний бит предыдущего байта. После передачи последнего бита предыдущего байта TSR загружается новым значением из TXREG (если оно присутствует), и устанавливается флаг прерывания TXIF. Это прерывание может быть разрешено/запрещено битом TXIE. Флаг TXIF устанавливается вне зависимости от состояния бита TXIE и может быть сброшен только загрузкой новых данных в регистр TXREG. Также, как TXIF отображает состояние TXREG, бит TRMT (TSTA<1>) показывает состояние регистра TSR. Этот бит не вызывает генерацию прерывания, доступен только на чтение и устанавливается в ‘1’, когда регистр TSR пуст. Регистр TSR не отображается на память и не доступен пользователю.

Передача разрешается установкой бита TXEN (TXSTA<5>), но не начнется до тех пор, пока не будут загружен регистр TXREG. Данные появятся на выходе по первому переднему фронту тактового сигнала CK. Выходные данные стабилизируются к заднему фронту тактового сигнала (см. рисунок 18-14). Можно сначала загрузить данные в TXREG, и потом установить бит TXEN в ‘1’ (см. рисунок 18-15). Это полезно при низких скоростях передачи данных, когда генератор BRG остановлен, а биты TXEN, CREN, SREN сброшены в нуль. Установка бита TXEN в ‘1’ запустит генератор BRG, который немедленно начнет формировать тактовый сигнал. Обычно после разрешения передачи регистр TSR пуст, и в результате записи в TXREG данные переписываются в TSR, что позволяет реализовать слитную передачу данных.

Сброс бита TXEN в ‘0’ вызовет немедленное прекращение передачи, остановку логики передатчика и переведет выводы CK, DT в третье состояние. Установка бита CREN или SREN во время передачи вызовет ее прекращение и переведет вывод DT в третье состояние (для приема данных), а вывод CK останется выходом тактового сигнала, если бит CSRC установлен. Логика передатчика не сбрасывается, хотя отключена от вывода, для сброса логики передатчика необходимо очистить бит TXEN. Если бит SREN был установлен в ‘1’, чтобы прервать текущую передачу и принять одиночное слово, то получив слово бит SREN сбросится. Последовательный порт продолжит передачу, если установлен бит TXEN. Линия данных DT переключится из третьего состояния для начала передачи данных. Чтобы это предотвратить, необходимо сбросить бит TXEN в ‘0’.

Для разрешения 9-разрядной передачи, необходимо установить бит TX9 (TXSTA<6>) в ‘1’. Девятый бит данных записывается в бит TX9D (TXSTA<0>). Девятый бит данных должен быть указан до записи в регистр TXREG, потому что данные, записанные в регистр TXREG, могут быть сразу загружены в сдвиговый регистр TSR.

Рекомендованная последовательность действий для передачи данных в синхронном ведущем режиме:

1. Установить требуемую скорость передачи с помощью регистра SPBRG и бита BRGH (см. раздел 18.3).
2. Выбрать синхронный ведущий режим установкой битов SYNC, SPEN и CSRC в ‘1’.
3. Если необходимо, разрешить прерывания установкой бита TXIE в ‘1’.
4. Если передача 9-разрядная, установить бит TX9 в ‘1’.
5. Разрешить передачу установкой бита TXEN в ‘1’.
6. Если передача 9-разрядная, записать 9-й бит данных в TX9D.
7. Записать данные в регистр TXREG.

Рис. 18-14 Временная диаграмма синхронной передачи двух 8-разрядных слов (SPBRG = 0)

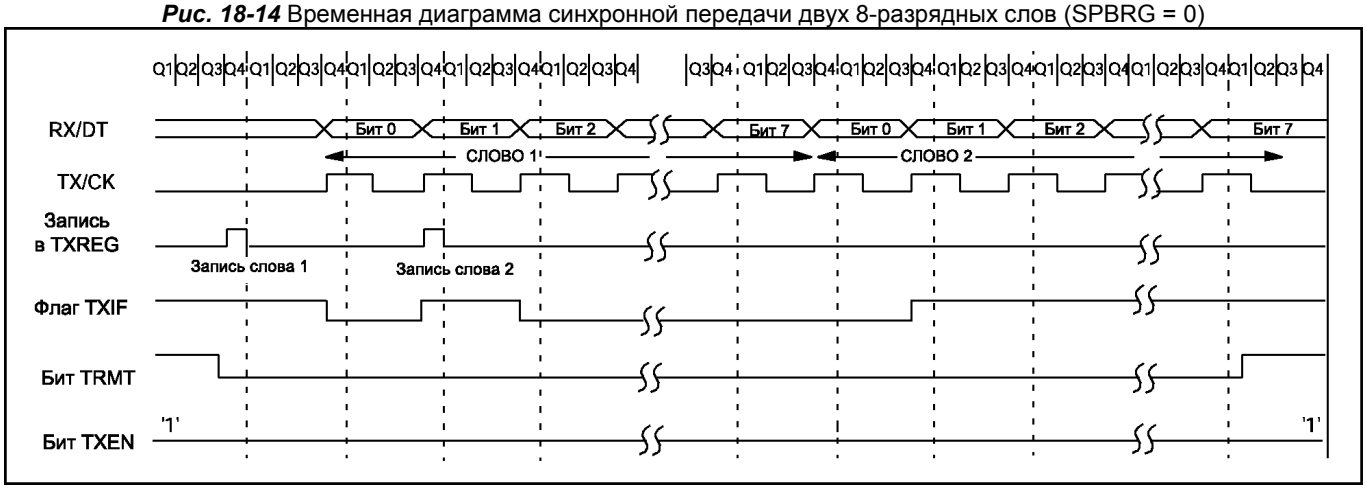


Рис. 18-15 Временная диаграмма синхронной передачи (после установки бита TXEN в ‘1’)

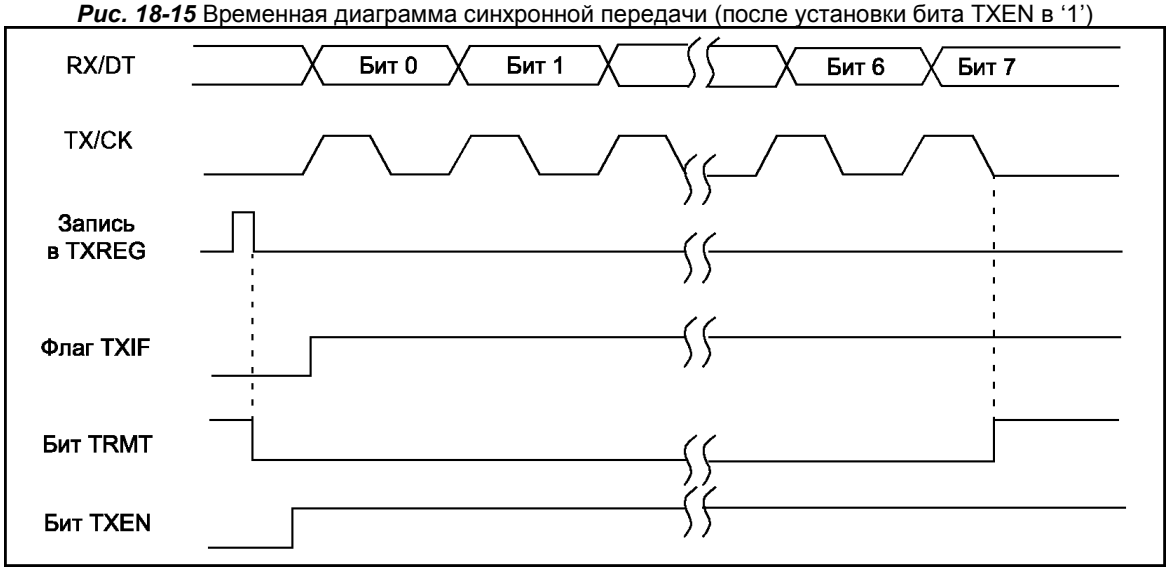


Таблица 18-8 Регистры и биты, связанные с работой передатчика USART в синхронном ведущем режиме

Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Сброс POR, BOR	Другие сбросы
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000u
PIR	TXIF(1)								0	0
PIE	TXIE(1)								0	0
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
TXREG	Регистр данных передатчика USART								0000 0000	0000 0000
TXSTA	CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D	0000 –010	0000 –010
SPBRG	Регистр генератора скорости USART								0000 0000	0000 0000

Обозначения: - = не используется, читается как 0; u = не изменяется; x = не известно; q = зависит от условий.
Затененные биты на работу не влияют.
Примечания:

1. Расположение битов смотрите в технической документации на микроконтроллер.
2. В некоторых микроконтроллерах эти биты могут обозначаться как GPIE и GPIF.

18.5.2 Прием синхронного ведущего

В синхронном режиме прием разрешается установкой битов CREN (RCSTA<4>) или SREN (RCSTA<5>) в ‘1’. Линия данных RX/DT опрашивается по заднему фронту тактового сигнала. Если бит SREN установлен в ‘1’, то принимается одиночное слово. Если бит CREN установлен в ‘1’, то в не зависимости от состояния бита SREN будет производится поточный прием данных, пока CREN не будет равен нулю. Получив последний бит очередного слова, данные переписываются из RSR в регистр RCREG (если он пуст). После записи в регистр RCREG выставляется флаг прерывания RCIF. Прерывание можно разрешить/запретить установкой/сбросом бита RCIE. Флаг RCIF доступен только на чтение, сбрасывается аппаратно при чтении из регистра RCREG. Регистр RCREG имеет двойную буферизацию, т.е. представляет собой двухуровневый FIFO. Поэтому можно принять 2 байта данных в FIFO RCREG и третий в регистр RSR. Если FIFO заполнен и обнаружен последний бит третьего байта, устанавливается бит переполнения приемника OERR (RCSTA<1>) в ‘1’. Байт принятый в RSR будет потерян. Для извлечения двух байт из FIFO необходимо дважды прочитать из регистра RCREG. Бит OERR нужно программно очистить сбросом бита CREN, т.е. запрещением приема. В любом случае, если бит OERR установлен, логика приема отключена.

Девятый бит принятых данных буферизируются так же, как принятые данные. Рекомендуется сначала прочитать регистр RCSTA, затем RCREG, чтобы не потерять бит записанный в RX9D.

Рекомендованные действия при приеме данных в синхронном ведущем режиме:

1. Установить требуемую скорость передачи с помощью регистра SPBRG и бита BRGH (см. раздел 18.3).
2. Выбрать синхронный ведущий режим установкой битов SYNC, SPEN и CSRC в ‘1’.
3. Сбросить биты SREN и CREN в ‘0’.
4. Если необходимо, разрешить прерывания установкой бита RCIE в ‘1’.
5. Если прием 9-разрядный, установить бит RX9 в ‘1’.
6. Если необходимо выполнить одиночный прием, установите бит SREN в ‘1’. Для поточного приема установите бит CREN в ‘1’.
7. Ожидать установку бита RCIF, или прерывание, если оно разрешено битом RCIE.
8. Считать 9-й бит данных (если разрешен 9-разрядный прием) из регистра RCSTA и проверить возникновение ошибки.
9. Считать 8 бит данных из регистра RCREG.
10. При возникновении ошибки переполнения сбросить бит CREN в ‘0’;

Рис. 18-16 Временная диаграмма синхронного приема в режиме ведущего (SREN = 1, SPBRG = 0)

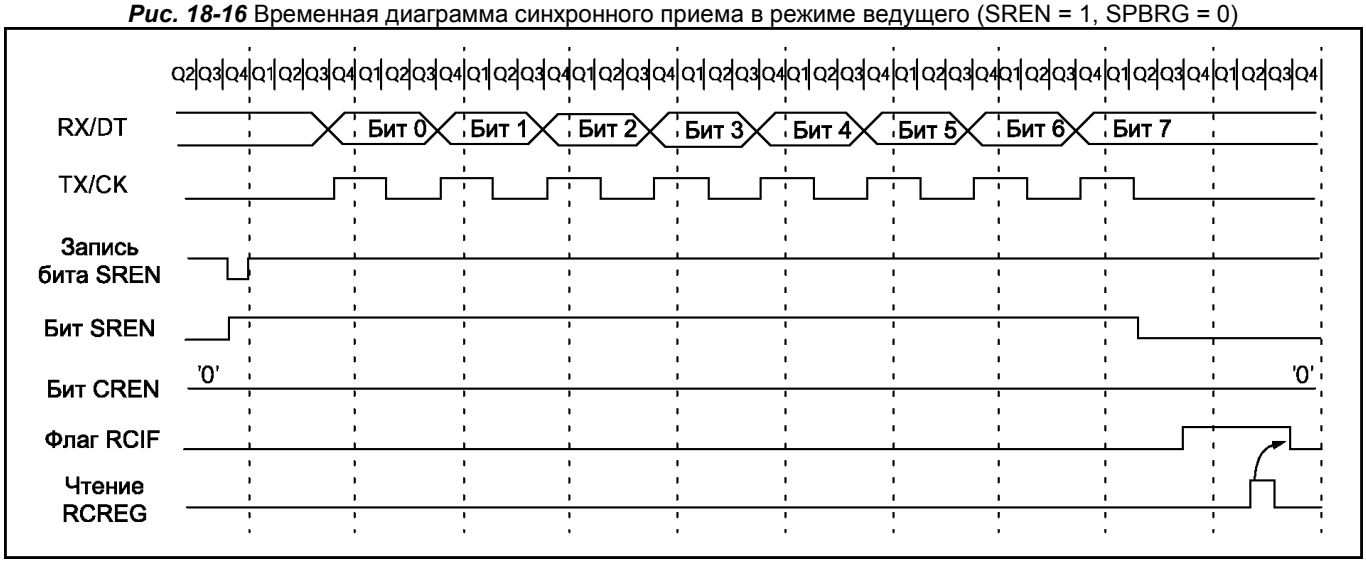


Таблица 18-9 Регистры и биты, связанные с работой приемника USART в синхронном ведущем режиме

Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Сброс POR, BOR	Другие сбросы
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000u
PIR	RCIF(1)								0	0
PIE	RCIE(1)								0	0
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
RCREG	Регистр данных приемника USART								0000 0000	0000 0000
TXSTA	CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D	0000 -010	0000 -010
SPBRG	Регистр генератора скорости USART								0000 0000	0000 0000

Обозначения: - = не используется, читается как 0; u = не изменяется; x = не известно; q = зависит от условий. Затененные биты на работу не влияют.

Примечания:

1. Расположение битов смотрите в технической документации на микроконтроллер.
2. В некоторых микроконтроллерах эти биты могут обозначаться как GPIE и GPIF.

18.6 Синхронный ведомый режим USART

Режим ведомого отличается от ведущего тем, что микроконтроллер использует тактовый сигнал с входа TX/CK, а не формирует его самостоятельно. Это позволяет устройству принимать и передавать данные в SLEEP режиме. Выбрать режим ведомого можно сбросом бита CSRC (TXSTA<7>) в ‘0’.

18.6.1 Передача синхронного ведомого

Работа передатчика в обоих синхронных режимах одинакова, за исключением работы ведомого в SLEEP режиме микроконтроллера.

Если в TXREG были записаны два слова подряд и исполнена команда SLEEP, выполняются следующие действия:

a) Первое слово сразу записывается в TSR и передается по мере прихода тактового сигнала.
b) Второе слово остается в TXREG
c) Флаг TXIF не устанавливается в ‘1’.
d) После передачи первого слова, второе слово передается из TXREG в TSR, и устанавливается флаг TXIF в ‘1’.
e) Если установлен бит TXIE в ‘1’, микроконтроллер выходит из режима SLEEP, происходит переход по вектору 0004h, если GIE=1.

Рекомендованная последовательность действий для передачи данных в синхронном ведомом режиме:

1. Выбрать синхронный ведомый режим установкой битов SYNC, SPEN в ‘1’ и сбросом CSRC в ‘0’.
2. Сбросить биты SREN и CREN в ‘0’.
3. Если необходимо, разрешить прерывания установкой бита TXIE в ‘1’.
4. Если передача 9-разрядная, установить бит TX9 в ‘1’.
5. Разрешить передачу установкой бита TXEN в ‘1’.
6. Если передача 9-разрядная, записать 9-й бит данных в TX9D.
7. Для начала передачи записать данные в регистр TXREG.
8. Если используются прерывания, то биты GIE и PEIE в регистре INTCON должны быть установлены в '1'.

Таблица 18-10 Регистры и биты, связанные с работой передатчика USART в синхронном ведомом режиме

Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Сброс POR, BOR	Другие сбросы
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000u
PIR	TXIF(1)								0	0
PIE	TXIE(1)								0	0
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
TXREG	Регистр данных передатчика USART								0000 0000	0000 0000
TXSTA	CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D	0000 –010	0000 –010

Обозначения: - = не используется, читается как 0; u = не изменяется; x = не известно; q = зависит от условий. Затененные биты на работу не влияют.

Примечания:

1. Расположение битов смотрите в технической документации на микроконтроллер.
2. В некоторых микроконтроллерах эти биты могут обозначаться как GPIE и GPIF.

18.6.2 Прием синхронного ведомого

Работа приемника в обоих синхронных режимах одинакова, кроме работы в режиме SLEEP. В синхронном ведомом режиме не учитывается состояние бита SREN.

Если перед выполнением команды SLEEP был разрешен прием (бит CREN = 1), то модуль USART может принять слово в SLEEP режиме микроконтроллера. По окончанию приема данные передаются из регистра RSR в RCREG. Если бит RCIE = 1, микроконтроллер выйдет из режима SLEEP. Если GIE=1, произойдет переход по адресу вектора прерываний 0004h.

Рекомендованные действия при приеме данных в синхронном ведомом режиме:

1. Выбрать синхронный ведомый режим установкой битов SYNC, SPEN в ‘1’ и сбросом CSRC в ‘0’.
2. Если необходимо, разрешить прерывания установкой бита RCIE в ‘1’.
3. Если прием 9-разрядный, установить бит RX9 в ‘1’.
4. Установите бит CREN в ‘1’ для разрешения приема.
5. Ожидать установку бита RCIF, или прерывание, если оно разрешено битом RCIE.
6. Считать 9-й бит данных (если разрешен 9-разрядный прием) из регистра RCSTA и проверить возникновение ошибки.
7. Считать 8 бит данных из регистра RCREG.
8. При возникновении ошибки переполнения сбросить бит CREN в ‘0’.

Таблица 18-11 Регистры и биты, связанные с работой приемника USART в синхронном ведомом режиме

Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Сброс POR, BOR	Другие сбросы
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000u
PIR	RCIF(1)								0	0
PIE	RCIE(1)								0	0
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
RCREG	Регистр данных приемника USART								0000 0000	0000 0000
TXSTA	CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D	0000 –010	0000 –010

Обозначения: - = не используется, читается как 0; u = не изменяется; x = не известно; q = зависит от условий.
Затененные биты на работу не влияют.

Примечания:

1. Расположение битов смотрите в технической документации на микроконтроллер.
2. В некоторых микроконтроллерах эти биты могут обозначаться как GPIE и GPIF.

18.7 Инициализация

В примере 18-2 показана инициализация приемника/передатчика USART в асинхронном режиме, а в примере 18-3 представлена инициализация модуля USART в синхронном режиме. В обоих режимах используются 8 - разрядные данные и одинаковая скорость передачи данных.

Пример 18-2 Инициализация модуля USART в асинхронном режиме

```
BSF     STATUS,RP0    ; Банк 1
MOVLW   <baudrate>    ; Установить скорость обмена данными
MOVWF   SPBRG
MOVLW   0x40          ; Передача 8 - разрядных данных, включить передатчик,
MOVWF   TXSTA         ; низкоскоростной асинхронный режим
BSF     PIE1,TXIE     ; Разрешить прерывания от передатчика USART
BSF     PIE1,RCIE     ; Разрешить прерывания от приемника USART
BCF     STATUS,RP0    ; Банк 0
MOVLW   0x90          ; Прием 8 - разрядных данных, включить приемник,
MOVWF   RCSTA         ; включить модуль USART
```

Пример 18-3 Инициализация модуля USART в синхронном режиме

```
BSF     STATUS,RP0    ; Банк 1
MOVLW   <baudrate>    ; Установить скорость обмена данными
MOVWF   SPBRG
MOVLW   0xB0          ; Передача 8 - разрядных данных, включить передатчик,
MOVWF   TXSTA         ; синхронный режим
BSF     PIE1,TXIE     ; Разрешить прерывания от передатчика USART
BSF     PIE1,RCIE     ; Разрешить прерывания от приемника USART
BCF     STATUS,RP0    ; Банк 0
MOVLW   0x90          ; Прием 8 - разрядных данных, включить приемник,
MOVWF   RCSTA         ; включить модуль USART, начать прием данных
```

18.8 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: В асинхронном режиме при передаче данных возникает много ошибок.
Ответ 1:
Наиболее часто встречающимися причинами этого может быть:

1. На одном из микроконтроллеров Вы используете высокоскоростной режим (BRGH = 1), который имеет некоторые особенности для PIC16C65/65A/73/73A/74/74A. Смотрите техническую документацию на микроконтроллеры.
2. Неправильно вычислено значение для загрузки в регистр SPBRG.
3. Очень большая ошибка скорости приемника и передатчика.

18.9 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с модулем USART в микроконтроллерах PICmicro MCU:

Документ	Номер
Serial Port Utilities Утилиты последовательного порта	AN547
Servo Control of a DC Brushless Motor Управление двигателем постоянного тока	AN532

Раздел 19. Источник опорного напряжения

Содержание

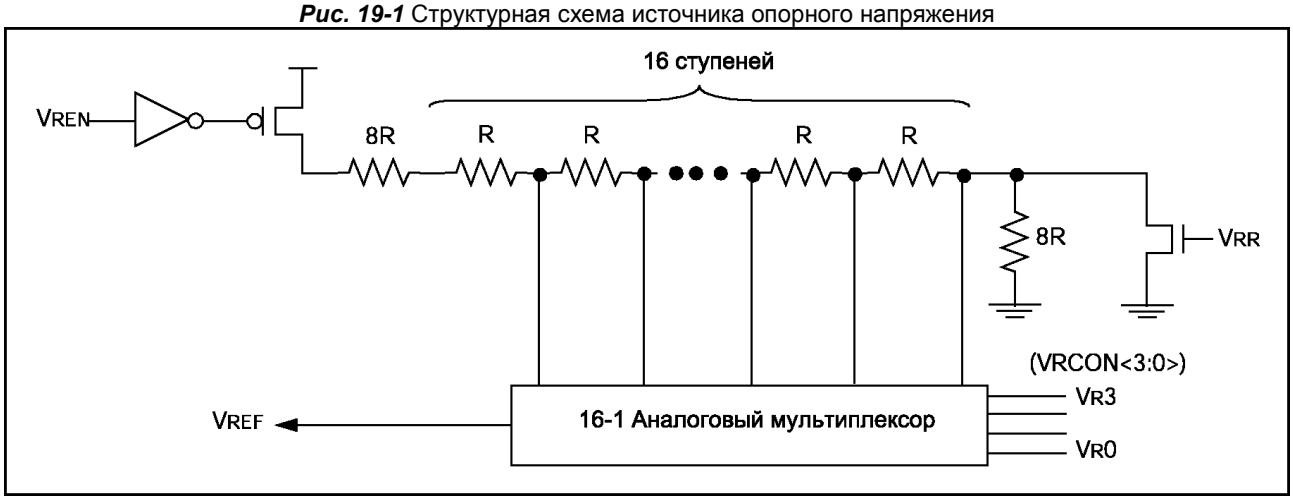
19.1 Введение

Модуль источника опорного напряжения как правило используется совместно с модулем компараторов. Входной ток выводов компаратора очень мал, поэтому источник опорного напряжения имеет малую нагрузочную способность.

Модуль источника опорного напряжения содержит 16 последовательно включенных резисторов, обеспечивающие выбор нужного напряжения. Резисторы разделены на сегменты для организации двух диапазонов напряжений V_{REF} и имеют возможность выключения для уменьшения тока потребления, когда источник опорного напряжения не используется.

В регистре VRCON находятся биты управления источником опорного напряжения. На рисунке 19-1 показана структурная схема источника опорного напряжения.

Рис. 19-1 Структурная схема источника опорного напряжения



Примечание. Смотрите параметр D312 в разделе "Электрические характеристики".

Таблица 19-1 Типовое значение V_{REF} при V_{DD} = 5В

VR3:VR0	V_{REF} (В)	
	VRR=1	VRR=0
0000	0.00	1.25
0001	0.21	1.41
0010	0.42	1.56
0011	0.63	1.72
0100	0.83	1.88
0101	1.04	2.03
0110	1.25	2.19
0111	1.46	2.34
1000	1.67	2.50
1001	1.88	2.66
1010	2.08	2.81
1011	2.29	2.97
1100	2.50	3.13
1101	2.71	3.28
1110	2.92	3.44
1111	3.13	3.59

19.2 Управляющий регистр

Регистр VRCON

R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
VREN	**VROE**	**VRR**	**-**	**VR3**	**VR2**	**VR1**	**VR0**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **VREN:** Включение источника опорного напряжения
1 = источник опорного напряжения включен
0 = источник опорного напряжения выключен и не потребляет тока

бит 6: **VROE:** Включение выхода V_{REF}
1 = выход V_{REF} подключен к выводу микроконтроллера
0 = выход V_{REF} не подключен к выводу микроконтроллера

бит 5: **VRR:** Диапазон выходного напряжения V_{REF}
1 = нижний диапазон, шаг $V_{DD}/24$
0 = верхний диапазон, шаг $V_{DD}/32$

бит 4: **Не используется:** читается как ‘0’

биты 3-0: **VR3:VR0:** Выбор выходного напряжения V_{REF} 0 <= VR[3:0] <= 15
Если VRR = 1: $V_{REF} = (VR<3:0>/24) \times V_{DD}$
Если VRR = 0: $V_{REF} = (V_{DD} \times ¼) + (VR<3:0>/32) \times V_{DD}$

19.3 Настройка источника опорного напряжения

Источник опорного напряжения может иметь 16 различных уровней напряжения для каждого диапазона.
Уравнение вычисления напряжения:

Если VRR = 1: V_{REF} = (VR<3:0>/24) x V_{DD}
Если VRR = 0: V_{REF} = (V_{DD} x ¼) + (VR<3:0>/32) x V_{DD}

Время установки напряжения должно определятся по напряжению на выводе V_{REF}. В примере 19-1 показана настройка опорного источника на напряжения 1.25В при напряжении питания 5В.

При известных значениях V_{REF} и V_{DD} Вам необходимо рассчитать значение битов VR3:VR0. В уравнении 19-1 показана формула вычисления VR3:VR0. Поскольку значение VR3:VR0 целочисленное, то может возникать некоторая ошибка. Уровни V_{REF} и V_{DD} должны быть выбраны таким образом, чтобы значение VR3:VR0 было не более 15.

Уравнение 19-1 Вычисление VR3:VR0

Если VRR=1,

$$VR3:VR0 = \frac{Vref}{Vdd} \times 24$$

Если VRR=0,

$$VR3:VR0 = \frac{Vref - Vdd/4}{Vdd} \times 24$$

19.4 Точность источника опорного напряжения

Полный диапазон выходных напряжений (от V_{SS} до V_{DD}) не может быть реализован из-за особенностей схемы источника опорного напряжения. Транзисторы, включенные в начале и конце резистивной цепочки, создают некоторое смешение (см. рисунок 19-1). Выходное напряжение формируется относительно V_{DD}, поэтому может изменяться с колебаниями V_{DD}. Абсолютную точность источника опорного напряжения смотрите в раздел "Электрические характеристики" параметр D311.

19.5 Работа в SLEEP режиме микроконтроллера

Когда микроконтроллер выходит из SLEEP режима или происходит переполнение сторожевого таймера WDT, значение регистра VRCON не изменяется. Для уменьшения суммарного тока потребления микроконтроллером в SLEEP режиме модуль источника опорного напряжения следует выключать перед переходом в режим SLEEP.

19.6 Эффект сброса

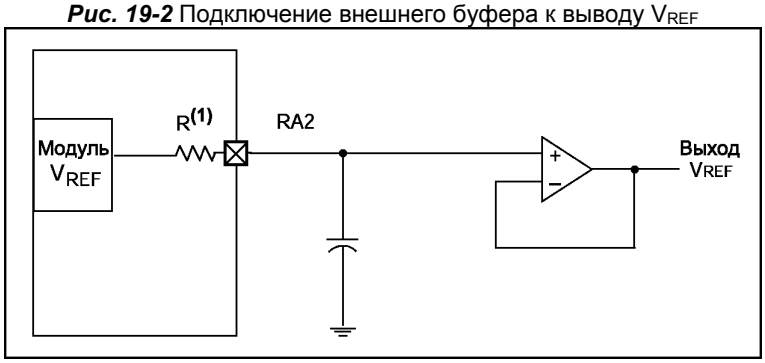
При сбросе микроконтроллера очищаются биты регистра VCON:
VREN (VRCON<7>) – модуль источника опорного напряжения выключен;
VROE (VRCON<6>) – V_{REF} отсоединен от вывода микроконтроллера;
VRR (VRCON<5>) – верхний диапазон напряжений;
VRCON<3:0> - напряжение опорного источника.

19.7 Подключение к источнику опорного напряжения

Модуль источника опорного напряжения работает независимо от модуля компараторов. Вывод источника опорного напряжения может быть подключен к выводу V_{REF}, если биты TRIS и VROE (VRCON<6>) установлены в ‘1’. Подключение источника опорного напряжения к выводу V_{REF}, с присутствующим на выводе внешним сигналом, может увеличить ток потребления микроконтроллера. Настройка вывода микроконтроллера как цифрового выхода также увеличит ток потребления.

Вывод V_{REF} может использоваться как простой ЦАП с малой разрешающей способностью. Выход источника опорного напряжения имеет малую нагрузочную способность, поэтому на выходе V_{REF} необходимо устанавливать дополнительный буфер (см. рисунок 19-2).

Рис. 19-2 Подключение внешнего буфера к выводу V_{REF}



Примечание 1. Выходное сопротивление R зависит от битов VRCON<5> и VRCON<3:0>.

19.8 Инициализация

В примере 19-1 показана инициализация источника опорного напряжения.

Пример 19-1 Настройка опорного источника на напряжения 1.25В при напряжении питания 5В

```
MOVLW  0x02          ; 4 аналоговых входа
MOVWF  CMCON         ; 2-х компараторов.
BSF    STATUS,RP0    ; Выбрать Банк 1
MOVLW  0x07          ; RA3-RA0 настроить
MOVWF  TRISA         ; как выходы
MOVLW  0xA6          ; включить VREF
MOVWF  VRCON         ; нижний диапазон напряжений
                             ; установить VR<3:0>=6
BCF    STATUS,RP0    ; Выбрать Банк 0
CALL   DELAY_10      ; задержка 10 мкс
```

19.9 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Напряжение V_{REF} отличается от ожидаемого.
Ответ 1:
Любое изменение напряжения V_{DD} передается на вывод V_{REF}. Необходимо проверить, что Вы правильно выбрали V_{DD} для требуемого V_{REF}.

Вопрос 2: При подключения к выводу V_{REF} нагрузки с малым сопротивлением напряжение V_{REF} ниже ожидаемого.
Ответ 2:
Модуль источника опорного напряжения не предназначен для подключения нагрузки с малым сопротивлением. Необходимо к выводу V_{REF} подключить буферную схему.

19.10 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с источником опорного напряжения в микроконтроллерах PICmicro MCU:

Документ	Номер
Resistance and Capacitance Meter using a PIC16C622 Измеритель емкости и сопротивления на микроконтроллере PIC16C622	AN611

Раздел 20. Модуль компараторов

Содержание

20.1 Введение

Модуль компараторов содержит два аналоговых компаратора, выводы которых мультиплицированы с каналами ввода/вывода. Выход интегрированного источника опорного напряжения может быть подключен на вход компараторов.

В регистре CMCON находятся биты управления модулем компараторов. Структурная схема модуля компараторов показана на рисунке 20-1.

20.2 Управляющий регистр

Регистр CMCON

R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
C2OUT	**C1OUT**	**C2INV**	**C1INV**	**CIS**	**CM2**	**CM1**	**CM0**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **C2OUT:** Выход компаратора 2
Если C2INV=0
1 = C2 V_{IN+} > C2 V_{IN-}
0 = C2 V_{IN+} < C2 V_{IN-}

Если C2INV=1
0 = C2 V_{IN+} > C2 V_{IN-}
1 = C2 V_{IN+} < C2 V_{IN-}

бит 6: **C1OUT:** Выход компаратора 1
Если C1INV=0
1 = C1 V_{IN+} > C1 V_{IN-}
0 = C1 V_{IN+} < C1 V_{IN-}

Если C1INV=1
0 = C1 V_{IN+} > C1 V_{IN-}
1 = C1 V_{IN+} < C1 V_{IN-}

бит 5: **C2INV:** Инверсный выход компаратора 2
1 = C2 инверсный выход
0 = C2 не инверсный выход

бит 4: **C1INV:** Инверсный выход компаратора 1
1 = C1 инверсный выход
0 = C1 не инверсный выход

бит 3: **CIS:** Подключение входов компараторов
Если CM2:CM3 = 001
1 = C1 V_{IN-} подключен к AN3
0 = C1 V_{IN-} подключен к AN0

Если CM2:CM3 = 010
1 = C1 V_{IN-} подключен к AN3
C2 V_{IN-} подключен к AN2
0 = C1 V_{IN-} подключен к AN0
C2 V_{IN-} подключен к AN1

биты 2-0: **CM2:CM0:** Режим работы компараторов
Смотрите рисунок 20-1.

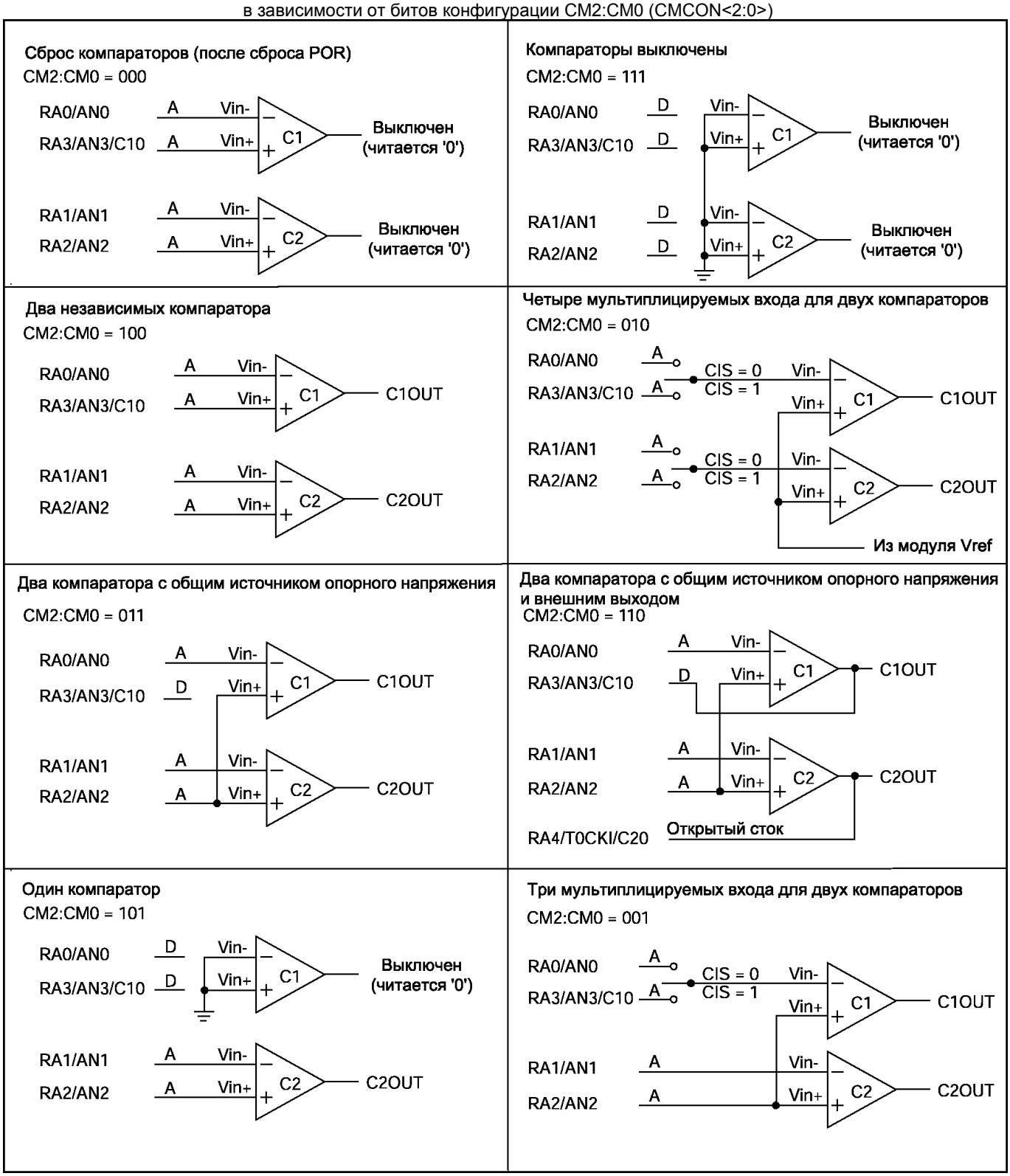
Примечание. В некоторых микроконтроллеров модуль компараторов не содержит биты C2INV, C1INV. Смотрите техническую документацию на микроконтроллер.

20.3 Настройка модуля компараторов

Существует восемь режимов работы модуля компараторов, устанавливаемые битами CM2:CM0 (сморите рисунок 20-1). Биты регистра TRIS управляют направлением каналов ввода/вывода для каждого режима модуля компараторов. При изменении режима работы модуля компараторов, параметры указанные в таблице электрических характеристик могут не соблюдаться.

Примечание. Для предотвращения ложных прерываний рекомендуется запретить прерывания от модуля компараторов, а затем изменить режим его работы.

Рис. 20-1 Структурная схема модуля компараторов
в зависимости от битов конфигурации CM2:CM0 (CMCON<2:0>)



Обозначения:

A = аналоговых вход, канал ввода/вывода читается как ‘0’; D = цифровой вход;
CIS = управляющий бит регистра CMCON<3>

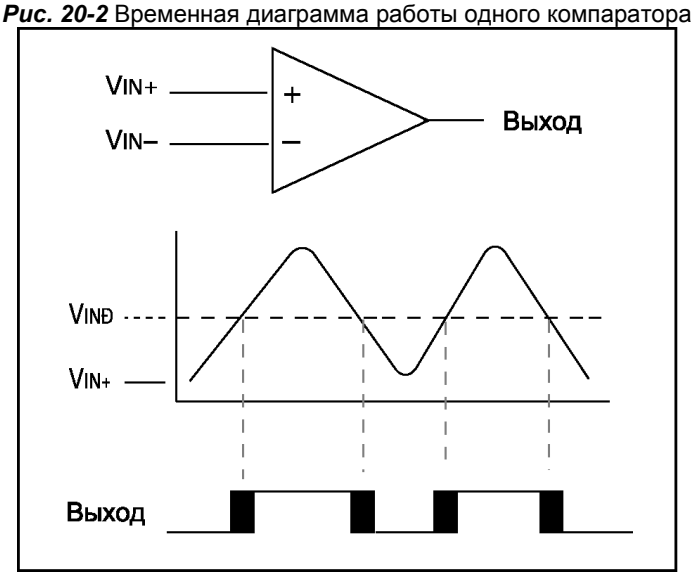
20.4 Работа модуля компараторов

Временная диаграмма работы одного компаратора показана на рисунке 20-2 (соотношение входных аналоговых сигналов и выходного цифрового сигнала). Когда аналоговый сигнал на входе V_{IN+} меньше V_{IN-} на цифровом выходе установлен логический нуль. Если сигнал на входе V_{IN+} больше V_{IN-}, то на цифровом выходе будет установлена логическая единица. Затененные области на рисунке 20-2 показывают неуверенный уровень цифрового сигнала.

20.5 Опорное напряжение для компараторов

Допускается использование внешнего или внутреннего источника опорного напряжения для компараторов (определяется режимом работы модуля компараторов). Аналоговый сигнал, присутствующий на выводе V_{IN-} сравнивается с сигналом V_{IN+}, по результатам сравнения формируется выходной цифровой сигнал (см. рисунок 20-2).

Рис. 20-2 Временная диаграмма работы одного компаратора



20.5.1 Внешний источник опорного напряжения

Модуль компараторов может быть настроен таким образом, что внешний источник опорного напряжения подключается на вход обоих компараторов или для каждого компаратора отдельный источник. Напряжение опорного источника должно быть в пределах от V_{SS} до V_{DD}.

20.5.2 Внутренний источник опорного напряжения

Модуль компараторов позволяет использовать внутренний источник опорного напряжения, описанный в разделе 19. Сигнал внутреннего источника опорного напряжения подключается к выводам V_{IN+} обоих компараторов, когда биты конфигурации CM2:CM0=010 (см. рисунок 20-1).

Источник опорного напряжения может работать в любом режиме модуля компараторов. В этом режиме вывод I/O/V_{REF} может использоваться как цифровой порт ввода/вывода или в качестве выхода источника опорного напряжения V_{REF}.

20.6 Время реакции компараторов

Время реакции – это гарантированная максимальная задержка изменения выходного цифрового сигнала после изменения входных сигналов. Если изменяется напряжение внутреннего опорного источника подключенного к компараторам, то должна рассматриваться задержка установки напряжения внутреннего опорного источника. Во всех остальных случаях используется максимальная задержка, указанная в разделе "Электрические характеристики".

20.7 Выходы компараторов

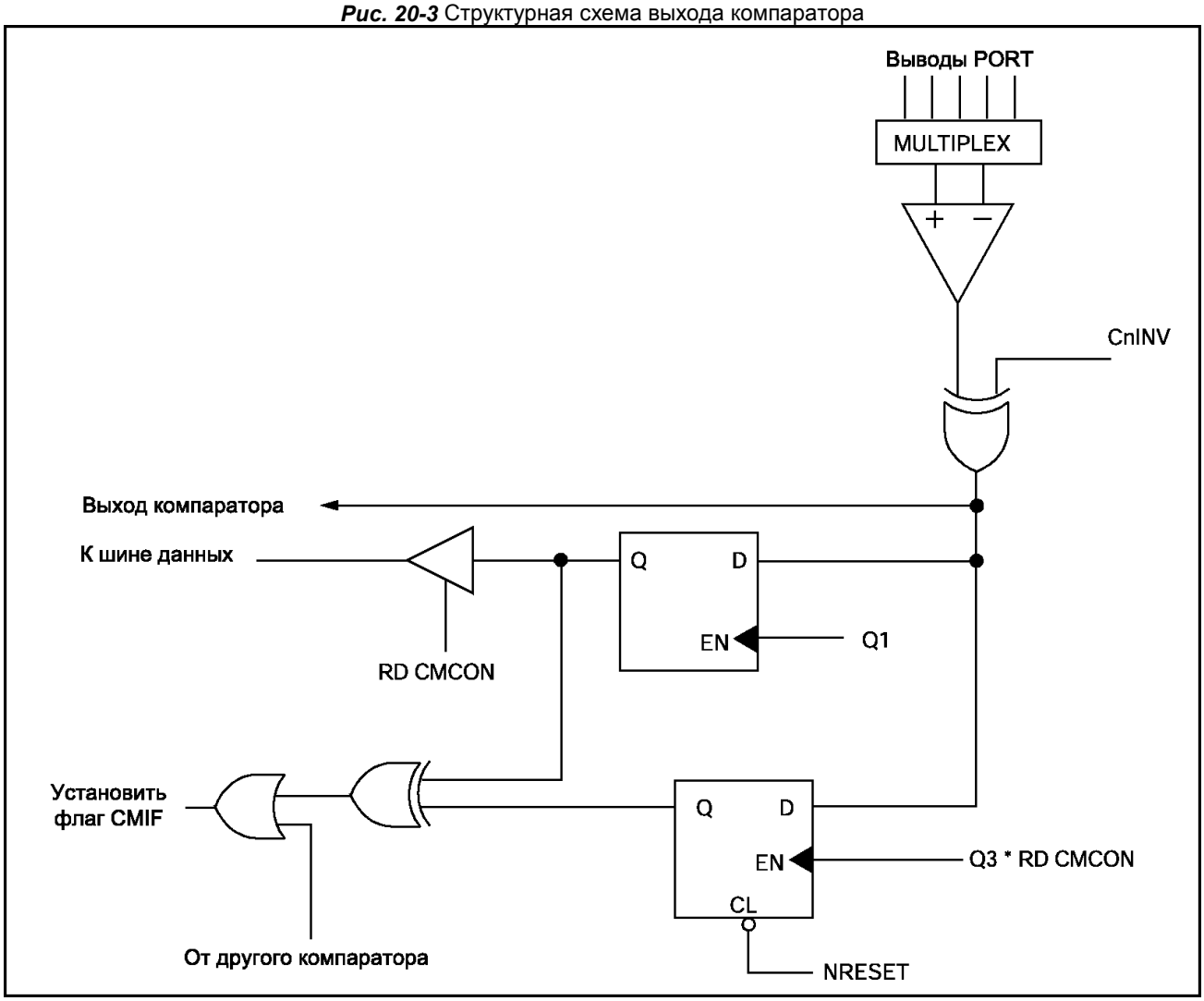
Состояние выходов компараторов можно прочитать в регистре CMCON (биты выходов компараторов доступны только на чтение). Выводы компараторов также могут быть подключены к каналам портов ввода/вывода (CM<2:0> = 110 или 001). В этом режиме модуля компараторов каналы портов ввода/вывода становятся не синхронизированными выходами компараторов. Зона неуверенного сигнала на выходе зависит от источника опорного напряжения и времени реакции компаратора (см. электрические характеристики). На рисунке 20-3 показана структурная схема выхода компаратора.

Соответствующие биты в регистре TRIS определяют подключать или нет выход компараторов к каналам портов ввода/вывода.

Примечание 1. При чтении регистра PORT, все выводы, настроенные как аналоговые, будут давать результат ‘0’. Выводы, настроенные как цифровые входы, конвертируют аналоговый сигнал согласно характеристикам входного триггера Шмидта.

Примечание 2. Аналоговые уровни сигналов, подаваемые на цифровые входы, могут быть причиной повышенного энергопотребления.

Рис. 20-3 Структурная схема выхода компаратора



Примечание. В некоторых микроконтроллеров модуль компараторов не содержит биты C2INV, C1INV. Смотрите техническую документацию на микроконтроллер.

20.8 Прерывания от компараторов

Модуль компараторов устанавливает флаг прерывания CMIF в ‘1’ при изменении уровня сигнала на выходе любого компаратора. Пользователь должен проверить, какой компаратор вызвал установку флага CMIF чтением битов CMCON<7:6>. Флаг прерывания от компараторов CMIF должен быть сброшен в ‘0’ программно. Программной установкой бита CMIF в ‘1’ моделируется возникновение прерывания от модуля компараторов.

Примечание. Если изменения в регистре CMCON (бит C1OUT или C2OUT) произошло, когда выполнялась операция чтения (начало такта Q2), флаг прерывания CMIF может не установиться в ’1’.

Биты CMIE, PEIE (INTCON<6>) и GIE (INTCON<7>) должны быть установлены в ‘1’, чтобы разрешить генерацию прерывания от модуля компараторов. Если любой из битов сброшен в ‘0’, прерывания не генерируются, но флаг CMIF устанавливается в ‘1’ при возникновении условия прерывания.

В подпрограмме обработки прерываний необходимо выполнить следующие действия:

a) Произвести запись или чтение регистра CMCON для устранения условия несоответствия.
b) Сбросить флаг CMIF в ‘0’.

Флаг CMIF будет аппаратно устанавливаться в ‘1’ до тех пор, пока не будет устранено условие несоответствия. Чтение регистра CMCON устранит условие несоответствия и позволит сбросить флаг CMIF в ‘0’.

20.9 Работа модуля компараторов в SLEEP режиме микроконтроллера

Если модуль компараторов включен, то при переходе микроконтроллера в режим SLEEP компараторы продолжают работать. Если прерывания от компараторов разрешены, то по возникновению прерывания микроконтроллер выйдет из режима SLEEP.

При включенных компараторах ток потребления микроконтроллера в режиме SLEEP несколько выше, чем указано в спецификации (каждый включенный компаратор потребляет дополнительный ток). Если в режиме SLEEP компараторы не используются, то рекомендуется их выключать (CM<2:0> = 111) перед переходом в режим SLEEP для уменьшения суммарного тока потребления.

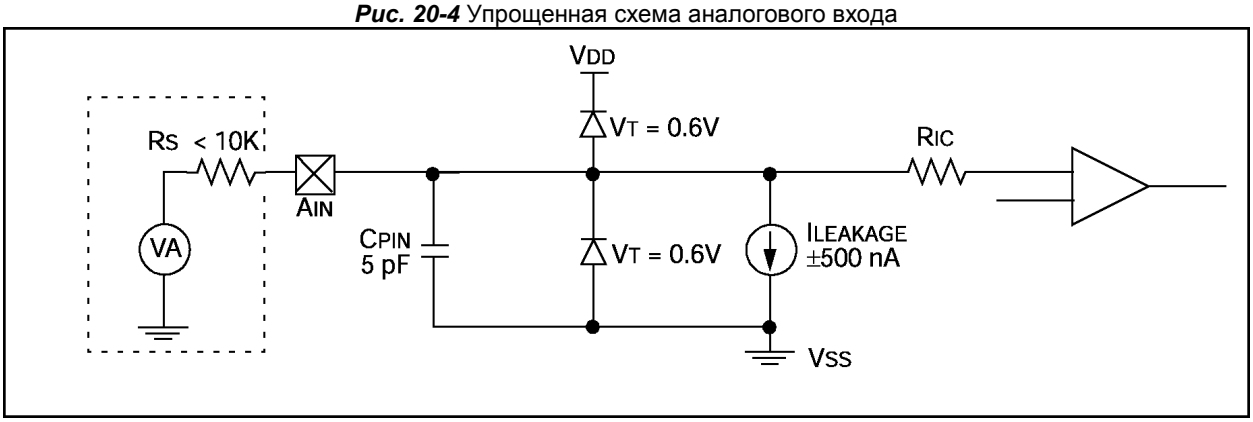
20.10 Эффект сброса

При любом виде сброса микроконтроллера все биты регистра CMCON сбрасываются в ‘0’. Сброс включает оба компаратора (CM2:CM0=000), делая все входы аналоговыми. Настройка каналов ввода/вывода как аналоговые входы при сбросе микроконтроллера позволяет минимизировать потребляемый ток.

20.11 Подключение к аналоговым входам

Упрощенная схема аналогового входа показана на рисунке 20-4. Т.к. аналоговые входы мультиплицированы с цифровыми входами, они имеют пару защитных диодов подключенных к V_{DD} и V_{SS}. Амплитуда аналогового сигнала должна быть в пределах от V_{SS} до V_{DD}. Амплитуда входного сигнала ограничивается в пределах от V_{SS}-0.6В до V_{DD}+0.6В. Внутреннее сопротивление источника аналогового сигнала должно быть меньше 10кОм. Компоненты, подключаемые к аналоговому входу (конденсатор, стабилитрон и т.д.), должны иметь минимальный ток утечки.

Рис. 20-4 Упрощенная схема аналогового входа



Обозначения:

C_{PIN} – входная емкость;
V_T – напряжение ограничения;
$I_{LEAKAGE}$ – ток утечки вывода;
R_{IC} – сопротивление соединения;
R_S – сопротивление источника;
VA – аналоговый сигнал.

Таблица 20-1 Регистры и биты, связанные с работой модуля компараторов

Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Сброс POR, BOR	Другие сбросы
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000u
PIR	CMIF(1)								0	0
PIE	CMIE(1)								0	0
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0000	0000 0000
VRCON	VREN	VROE	VRR	-	VR3	VR2	VR1	VR0	000- 0000	000- 0000

Обозначения: - = не используется, читается как 0; u = не изменяется; x = не известно; q = зависит от условий. Затененные биты на работу не влияют.

Примечания:

1. Расположение битов смотрите в технической документации на микроконтроллер.
2. В некоторых микроконтроллерах эти биты могут обозначаться как GPIE и GPIF.

20

20.12 Инициализация

В примере 20-1 показана настройка модуля компараторов микроконтроллеров PIC16C62X (RA3, RA4 – цифровые выходы; RA0, RA1 – входы компараторов V-; RA2 – вход V+обоих компараторов).

Пример 20-1 Инициализация модуля компараторов (PIC16C62X)

```
FLAG_REG  EQU    0X20
;
          CLRF          FLAG_REG      ; Инициализация регистра флагов
          CLRF          PORTA         ; Инициализация PORTA
          ANDLW         0xC0          ; Маска битов компараторов
          IORWF         FLAG_REG,F    ; Записать биты в регистр флагов
          MOVLW         0x03          ; Установить режим модуля компараторов
          MOVWF         CMCON         ; CM<2:0> = 011
          BSF           STATUS,RP0    ; Выбрать банк 1
          MOVLW         0x07          ; Направление выводов PORTA
          MOVWF         TRISA         ; RA<2:0> входы, RA<4:3> выходы
                                      ; TRISA<7:5> читаются как ‘0’
          BCF           STATUS,RP0    ; Выбрать банк 0
          CALL          DELAY_10      ; Задержка 10мкс
          MOVF          CMCON,F       ; Чтение CMCON для устранения несоответствия
          BCF           PIR1,CMIF     ; Сбросить флаг прерываний
          BSF           STATUS,RP0    ; Выбрать банк 1
          BSF           PIE1,CMIE     ; Разрешить прерывания от модуля компараторов
          BCF           STATUS,RP0    ; Выбрать банк 0
          BSF           INTCON,PEIE   ; Разрешить прерывания от периферийных модулей
          BSF           INTCON,GIE    ; Глобальное разрешение прерываний
```

20.13 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Создается впечатление, что программа микроконтроллера "зависает".
Ответ 1:
Это может происходить из-за бесконечного цикла обработки прерываний от модуля компараторов. Если в обработке прерываний Вы не выполнили требуемую последовательность действий сброса в '0' флага CMIF, то микроконтроллер будет постоянно переходить на обработку прерываний. Сначала нужно прочитать регистр CMCON, а затем сбросить в '0' флаг CMIF.

20.14 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с модулем компараторов в микроконтроллерах PICmicro MCU:

Документ	Номер
Resistance and Capacitance Meter using a PIC16C622 Измеритель емкости и сопротивления на микроконтроллере PIC16C622	AN611

Раздел 21. Модуль 8 - разрядного АЦП

Содержание

21.1 Введение

Модуль аналого-цифрового преобразования (АЦП) имеет до восьми входных каналов.

Входной аналоговый сигнал через коммутатор каналов заряжает внутренний конденсатор АЦП C_{HOLD}. Модуль АЦП преобразует напряжение, удерживаемое на конденсаторе C_{HOLD} в соответствующий 8 - разрядный цифровой код методом последовательного приближения. Источник опорного напряжения может быть программно выбран с вывода V_{DD} или V_{REF}. Допускается работа модуля АЦП в SLEEP режиме микроконтроллера, при этом в качестве источника тактовых импульсов для АЦП должен быть выбран RC генератор.

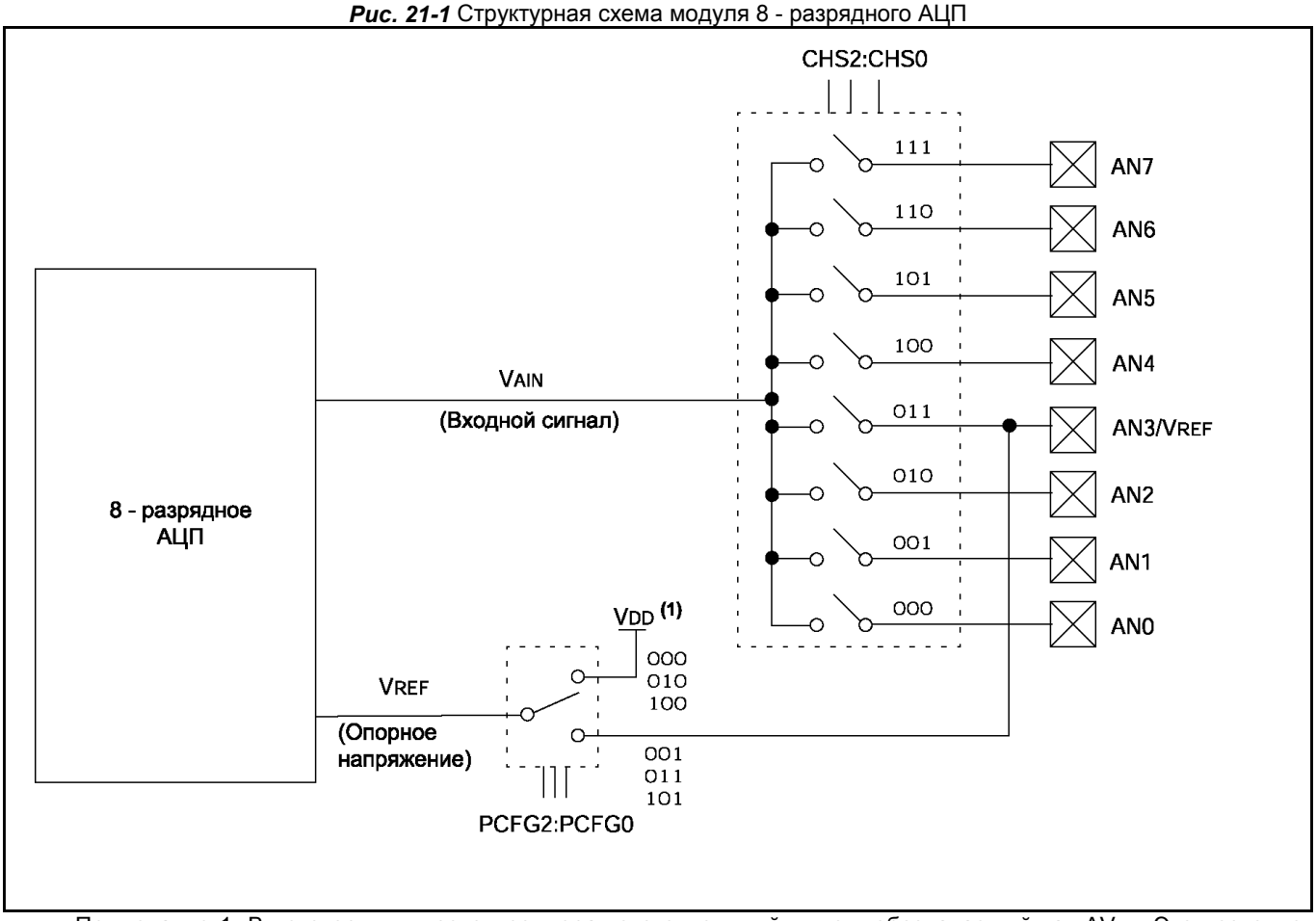
Для управления АЦП в микроконтроллере используется 3 регистра:

- Регистр результата ADRES;
- Регистр управления ADCON0;
- Регистр управления ADCON1.

Регистр ADCON0 используется для настройки работы модуля АЦП, а с помощью регистра ADCON1 устанавливается, какие входы микроконтроллера будут использоваться модулем АЦП и в каком режиме (аналоговый вход или цифровой порт ввода/вывода).

Структурная схема модуля АЦП показана на рисунке 21-1.

Рис. 21-1 Структурная схема модуля 8 - разрядного АЦП



Примечание 1. В некоторых микроконтроллерах это отдельный вывод, обозначаемый как AV_{DD}. Это позволяет подключать модуль АЦП к более точному источнику питающего напряжения.

21.2 Управляющие регистры

Регистр ADCON0:

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	**ADCS0**	**CHS2**	**CHS1**	**CHS0**	**GO/-DONE**	**-**	**ADON**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как '0'
–n – значение после POR
–x – неизвестное значение после POR

биты 7-6: **ADCS1:ADCS0:** Выбор источника тактового сигнала
00 = $F_{OSC}/2$
01 = $F_{OSC}/8$
10 = $F_{OSC}/32$
11 = F_{RC} (внутренний RC генератор модуля АЦП)

биты 5-3: **CHS2:CHS0:** Выбор аналогового канала
000 = канал 0, (AN0)
001 = канал 1, (AN1)
010 = канал 2, (AN2)
011 = канал 3, (AN3)
100 = канал 4, (AN4)
101 = канал 5, (AN5)
110 = канал 6, (AN6)
111 = канал 7, (AN7)

Примечание. Для микроконтроллеров, в которых не реализованы все 8 каналов АЦП. Модуль АЦП аппаратно позволяет выполнять выборку нереализованных каналов.

бит 2: **GO/-DONE:** Бит статуса модуля АЦП
Если ADON=1
1 = модуль АЦП выполняет преобразование (установка бита вызывает начало преобразования)
0 = состояние ожидания (аппаратно сбрасывается по завершению преобразования)

бит 1: **Не используется:** читается как ‘0’

бит 0: **ADON:** Бит включения модуля АЦП
1 = модуль АЦП включен
0 = модуль АЦП выключен и не потребляет тока

Регистр ADCON1:

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
-	-	-	-	-	**PCFG2**	**PCFG1**	**PCFG0**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как '0'
–n – значение после POR
–x – неизвестное значение после POR

биты 7-3: **Не используются:** читаются как ‘0’

биты 2-0: **PCFG2:PCFG0:** Управляющие биты настройки каналов АЦП

PCFG2:PCFG0	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0
000	A	A	A	A	A	A	A	A
001	A	A	A	A	V_{REF}	A	A	A
010	D	D	D	A	A	A	A	A
011	D	D	D	A	V_{REF}	A	A	A
100	D	D	D	D	A	D	A	A
101	D	D	D	D	V_{REF}	D	A	A
11x	D	D	D	D	D	D	D	D

A = аналоговый вход D = цифровой канал ввода/вывода

Примечание. Когда вывод AN3 работает как V_{REF}, то напряжение на AN3 является опорным для АЦП. Если вывод AN3 работает как аналоговый вход (A), то опорное напряжение для АЦП берется с вывода V_{DD}.

Примечание. При сбросе микроконтроллера все выводы мультиплицированные с модулем АЦП (ANx) настраиваются как аналоговые входы.

21.3 Работа модуля АЦП

В регистре ADRES сохраняется 8-разрядный результат аналого-цифрового преобразования. Когда преобразование завершено, результат преобразования записывается в регистр ADRES, после чего сбрасывается бит GO/-DONE (ADCON0<2>) и устанавливается флаг прерывания ADIF.

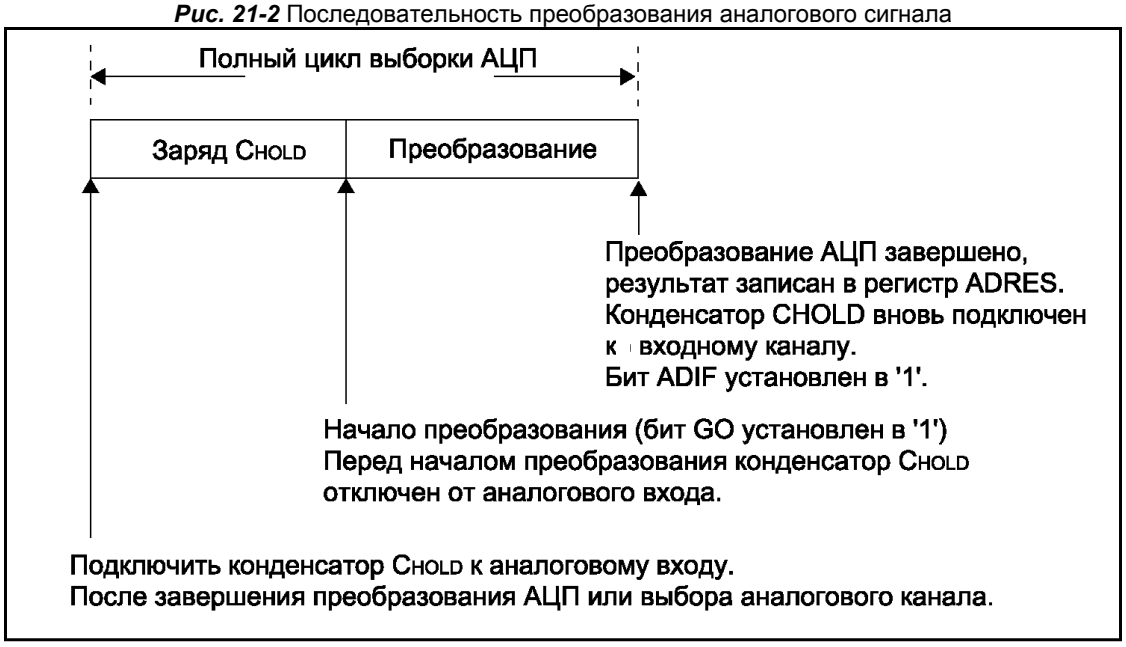
После включения и настройки АЦП необходимо выбрать рабочий аналоговый канал. Соответствующие биты TRIS аналоговых каналов должны настраивать канал порта ввода/вывода на вход. Перед началом преобразования необходимо выдержать временную паузу, расчет которой приведен в разделе 21.4.

Рекомендованная последовательность действий для работы с АЦП:

1. Настроить модуль АЦП:
   - Настроить выводы как аналоговые входы, входы V_{REF} или цифровые каналы ввода/вывода (ADCON1);
   - Выбрать входной канал АЦП (ADCON0);
   - Выбрать источник тактовых импульсов для АЦП (ADCON0);
   - Включить модуль АЦП (ADCON0).
2. Настроить прерывание от модуля АЦП (если необходимо):
   - Сбросить бит ADIF в '0';
   - Установить бит ADIE в '1';
   - Установить бит PEIE в '1';
   - Установить бит GIE в '1'.
3. Выдержать паузу, необходимую для зарядки конденсатора C_{HOLD}.
4. Начать аналого-цифровое преобразование:
   - Установить GO/-DONE бит в '1' (ADCON0).
5. Ожидать, окончания преобразования:
   - Ждать, пока бит GO/-DONE не будет сброшен в '0'; ИЛИ
   - Ожидать прерывание по окончанию преобразования.
6. Считать результат преобразования из регистра ADRES, сбросить бит ADIF в '0', если это необходимо.
7. Для следующего преобразования необходимо выполнить шаги начиная с пункта 1 или 2. Время преобразования одного бита определяется как время T_{AD}. Минимальное время ожидания перед следующим преобразованием должно составлять не менее $2T_{AD}$.

На рисунке 21-2 показана последовательность преобразования аналогового сигнала. Время заряда C_{HOLD} - интервал времени в течение которого на внутренний конденсатор АЦП подается внешний сигнал. Время преобразования равно 10 T_{AD}, отсчет начинается с момента установки в '1' бита GO. Сумма этих двух временных интервалов является длительностью полного цикла преобразования АЦП. Существует минимальный интервал времени, в течение которого внешний сигнал подается на внутренний конденсатор C_{HOLD}, чтобы гарантировать требуемую точность АЦП.

Рис. 21-2 Последовательность преобразования аналогового сигнала



21.4 Временные требования к подключению канала АЦП

Для обеспечения необходимой точности преобразования, конденсатор C_{HOLD} должен успевать полностью заряжаться до уровня входного напряжения. Схема аналогового входа АЦП показана на рисунке 21-3. Сопротивления R_S и R_{SS} непосредственно влияют на время зарядки конденсатора C_{HOLD}. Величина сопротивления ключа выборки (R_{SS}) зависит от напряжения питания V_{DD} (см. рисунок 21-3). **Максимальное рекомендуемое значение внутреннего сопротивления источника аналогового сигнала 10кОм.** При меньших значениях сопротивления источника сигнала меньше суммарное время преобразования.

После того, как будет выбран один из нескольких аналоговых входных каналов, но прежде, чем будет производиться преобразование, должно пройти определенное время. Для нахождения данного времени воспользуетесь уравнением 21-1. Это уравнение дает результат с ошибкой в ½ LSb (512 шагов АЦП). Ошибка в ½ LSb - это максимальная погрешность, позволяющая функционировать модулю АЦП с необходимой точностью.

Уравнение 21-1 Вычисление временной задержки

T_{ACQ} = Время задержки усилителя + Время заряда конденсатора C_{HOLD} + Температурный коэффициент
$= T_{AMP} + T_C + T_{COFF}$

Уравнение 21-2 Минимальное время заряда конденсатора C_{HOLD}

$$V_{HOLD} = (V_{REF} - (V_{REF}/512)) \bullet (1 - e^{(-Tc/Chold(Ric + Rss + Rs))})$$

$$T_C = - 51.2пФ\ (1кОм + R_{SS} + R_S)\ Ln(1/511)$$

В примере 21-1 показано вычисление минимального значения времени T_{ACQ}. Вычисления основываются на следующих исходных данных:

Rs	= 10кОм
Ошибка преобразования	≤ 1/2 Lsb
V_{DD}	= 5В → Rss = 7 кОм (см. график на рисунке 21-3)
Температура	= 50°C (максимально возможная)
V_{HOLD}	= 0В @ t = 0

Пример 21-1 Вычисление минимального значения времени T_{ACQ}

$T_{ACQ} = T_{AMP} + T_C + T_{COFF}$
T_{ACQ} = 5мкс + T_C + [(Температура - 25°C)(0.05мкс/°C)]

T_C = - C_{HOLD} (R_{IC} + R_{SS} + R_S) Ln(1/511)
= - 51.2пФ (1кОм + 7кОм + 10кОм) Ln(0.0020)
= - 51.2пФ (18кОм) Ln(0.0020)
= -0.921мкс (-6.2146)
= 5.724мкс

T_{ACQ} = 5мкс + 5.724мкс + [(50°C - 25°C)(0.05мкс/°C)]
= 10.724мкс + 1.25мкс
= 11.974мкс

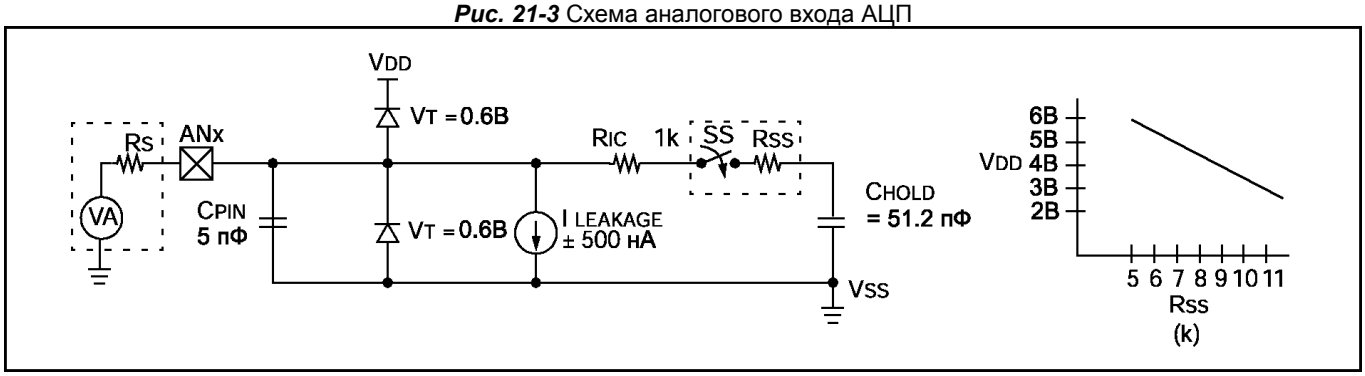
Примечание 1. Напряжение источника опорного напряжения V_{REF} не влияет на уравнение.

Примечание 2. Конденсатор C_{HOLD} после преобразования не разряжается.

Примечание 3. Максимальное рекомендуемое значение внутреннего сопротивления источника аналогового сигнала 10кОм. Это необходимо для компенсации внутреннего тока утечки.

Примечание 4. После того, как преобразование завершено, необходимо программно обеспечить задержку не менее 2.0T_{AD}, прежде чем начнете следующее преобразование. В течение этого времени конденсатор C_{HOLD} не подключен к выбранному входному каналу АЦП.

Рис. 21-3 Схема аналогового входа АЦП



Обозначения:

C_{PIN}	- входная емкость;
V_T	- пороговое напряжение;
$I_{LEAKAGE}$	- ток утечки вывода;
R_{IC}	- сопротивление соединения;
SS	- переключатель защелки;
C_{HOLD}	- конденсатор защелки.

21.5 Выбор источника тактовых импульсов для АЦП

Время получения одного бита результата равно T_{AD}. Для 8-разрядного результата требуется как минимум $9.5T_{AD}$. Параметры тактового сигнала для АЦП определяются программно, T_{AD} может принимать следующие значения:

- $2T_{OSC}$;
- $8T_{OSC}$;
- $32T_{OSC}$;
- Внутренний RC генератор модуля АЦП (2-6мкс).

Для получения корректного результата преобразования необходимо выбрать источник тактового сигнала АЦП, обеспечивающий время T_{AD} не менее 1.6 мкс (см. параметр 130 в разделе "Электрические характеристики").

В таблицах 21-1 и 21-2 указано максимальное значение тактовой частоты микроконтроллера для каждого режима синхронизирующего сигнала АЦП.

Таблица 21-1 Максимальное значение F_{OSC} удовлетворяющие требованию к T_{AD}
(для микроконтроллеров с нормальным диапазоном напряжения питания (C))

Источник импульсов АЦП		**Рабочая частота F_{osc}**			
Источник	**ADCS1:ADCS0**	**20МГц**	**5МГц**	**1.25МГц**	**333.33кГц**
$2T_{OSC}$	00	100нс(2)	400нс(2)	1.6 мкс	6мкс
$8T_{OSC}$	01	400нс(2)	1.6 мкс	6.4 мкс	24мкс(3)
$32T_{OSC}$	10	1.6 мкс	6.4 мкс	25.6 мкс(3)	96мкс(3)
RC-генератор	11	2-6мкс(1,4)	2-6мкс(1,4)	2-6мкс(1,4)	2-6мкс(1)

Обозначение: Затененные ячейки - не рекомендованное значение.

Примечания:

1. Типовое значение времени T_{AD} RC генератора АЦП равно 4мкс, может варьироваться от 2мкс до 6мкс.
2. Это значение выходит за пределы минимально допустимого времени T_{AD}.
3. Для более точного преобразования рекомендуется выбрать другой источник тактовых импульсов.
4. Когда тактовая частота микроконтроллера больше 1МГц, рекомендуется использовать RC генератор АЦП только для работы в SLEEP режиме.

Таблица 21-2 Максимальное значение F_{OSC} удовлетворяющие требованию к T_{AD}
(для микроконтроллеров с расширенным диапазоном напряжения питания (LC))

Источник импульсов АЦП		**Рабочая частота F_{osc}**			
Источник	**ADCS1:ADCS0**	**4МГц**	**2МГц**	**1.25МГц**	**333.33кГц**
$2T_{OSC}$	00	500нс(2)	1.0мкс(2)	1.6 мкс	6мкс
$8T_{OSC}$	01	2.0мкс(2)	4.0 мкс	6.4 мкс	24мкс(3)
$32T_{OSC}$	10	8.0 мкс	16.0 мкс	25.6 мкс(3)	96мкс(3)
RC-генератор	11	2-6мкс(1,4)	2-6мкс(1,4)	2-6мкс(1,4)	2-6мкс(1)

Обозначение: Затененные ячейки - не рекомендованное значение.

Примечания:

1. Типовое значение времени T_{AD} RC генератора АЦП равно 4мкс, может варьироваться от 2мкс до 6мкс.
2. Это значение выходит за пределы минимально допустимого времени T_{AD}.
3. Для более точного преобразования рекомендуется выбрать другой источник тактовых импульсов.
4. Когда тактовая частота микроконтроллера больше 1МГц, рекомендуется использовать RC генератор АЦП только для работы в SLEEP режиме.

21.6 Настройка аналоговых входов

Регистры ADCON1 и TRIS отвечают за настройку выводов АЦП. Если выводы микросхемы настраиваются как аналоговые входы, то при этом должны быть установлены соответствующие биты в регистре TRIS. Если соответствующий бит сброшен в '0', вывод микросхемы настроен как цифровой выход, со значениями выходных напряжений V_{OH} или V_{OL}.

Модуль АЦП функционирует независимо от состояния битов CHS2:CHS0 и битов TRIS.

Примечание 1. При чтении содержимого регистра порта нули будут установлены в тех разрядах, которые были настроены как аналоговые входы. Настроенные на цифровой вход каналы будут преобразовывать входные аналоговые уровни в цифровые, что однако не окажет влияния на точность преобразования.

Примечание 2. Значения напряжений, подаваемых на выводы, настроены как аналоговые входы, включая выводы (AN7:AN0), могут влиять на ток потребления входного буфера, который может выйти за пределы значений, оговоренных в технической спецификации.

21.7 Аналого-цифровое преобразование

В примере 21-2 показана последовательность действий для работы с АЦП. Выводы настроены как аналоговые входы. Источник опорного напряжения – V_{DD}. Разрешены прерывания от модуля АЦП. Источником импульсов преобразования является RC генератор АЦП. Аналоговое цифровое преобразование выполняется с вывода AN0.

Примечание. Бит GO/-DONE и бит включения АЦП должны устанавливаться разными командами.

Сброс бита GO/-DONE в '0' во время преобразования приведет к его прекращению. При этом регистр результата (ADRES) не изменит своего содержимого. После досрочного завершения преобразования необходимо обеспечить временную задержку $2T_{AD}$. Выдержав требуемую паузу, можно начать новое преобразования установкой бита GO/-DONE в '1'.

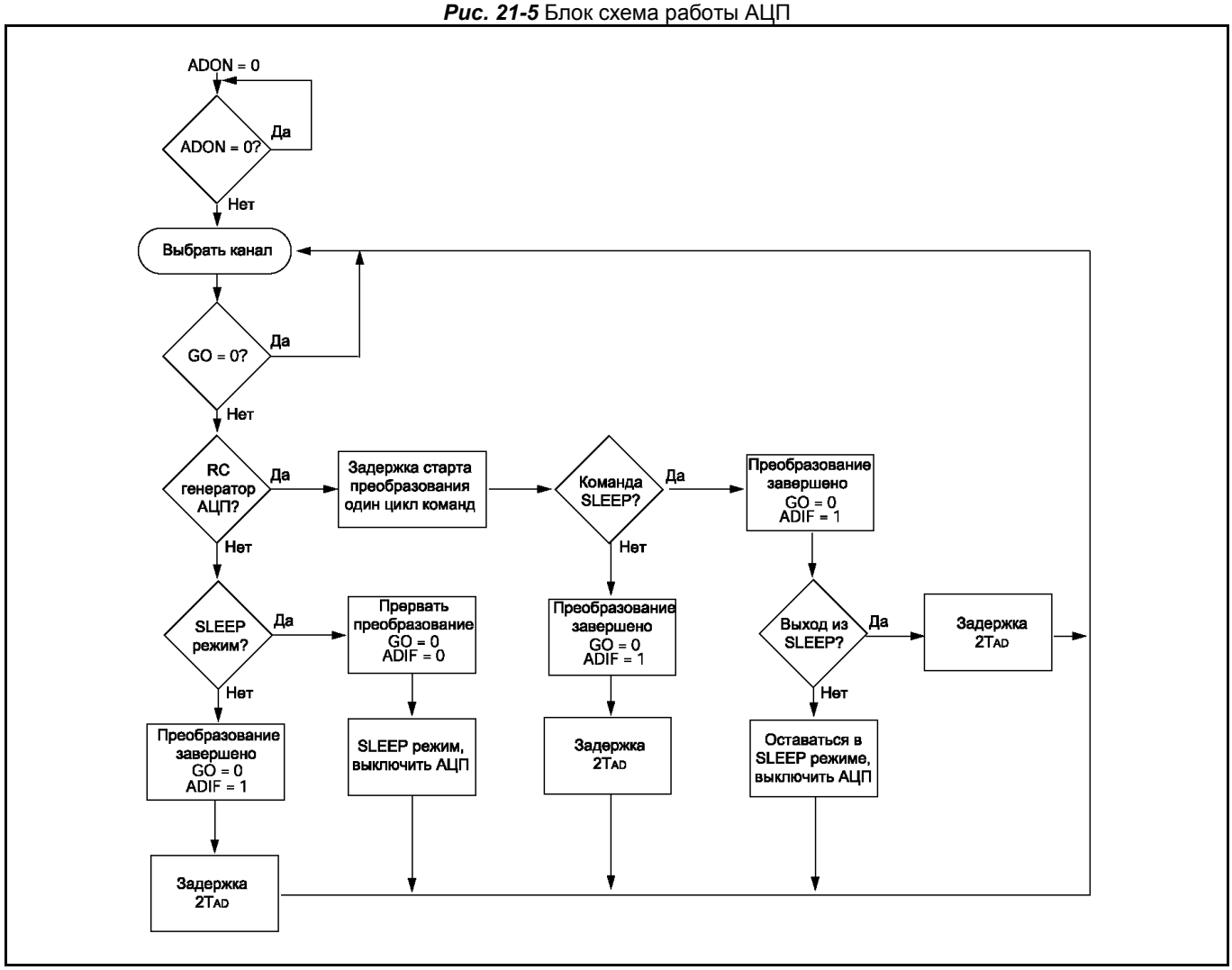
На рисунке 21-4 показана последовательность получения результата после установки бита GO/-DONE в '1'.

Пример 21-2 Выполнение преобразования АЦП

```
            BSF         STATUS, RP0   ; Выбрать банк 1
            CLRF        ADCON1        ; Настроить входы АЦП
            BSF         PIE1, ADIE    ; Разрешить прерывания от АЦП
            BCF         STATUS, RP0   ; Выбрать банк 0
            MOVLW       0xC1          ; Тактовые импульсы от RC генератора АЦП,
            MOVWF       ADCON0        ; включить АЦП, выбрать канал 0
            BCF         PIR1, ADIF    ; Сбросить флаг прерываний от АЦП
            BSF         INTCON, PEIE  ; Разрешить периферийные прерывания
            BSF         INTCON, GIE   ; Разрешить прерывания в системе
;
; Выдержать паузу, необходимую для заряда внутреннего конденсатора CHOLD.
; Затем начинать преобразование АЦП.
;
            BSF         ADCON0, GO    ; Старт преобразования
            :                         ; Ожидать установку флага ADIF или сброс
            :                         ; бита GO/-DONE по завершению преобразования
```

Рис. 21-4 Последовательность получения результата после установки бита GO/-DONE в '1'

Рис. 21-5 Блок схема работы АЦП



21.7.1 Быстрое преобразование взамен разрешающей способности

Иногда бывает необходимо быстрое по времени преобразование за счет снижения разрешающей способности АЦП. Эта особенность работы модуля АЦП может использоваться в программе пользователя. Независимо от используемой разрешающей способности интервал времени T_{ACQ} остается неизменным.

Для ускорения преобразования генератор тактовых импульсов модуля АЦП может быть включен так, чтобы время T_{AD} вышло за пределы минимально-допустимого значения (см. соответствующие параметры в разделе "Электрические характеристики"). При этом модуль АЦП будет выдавать некорректный результат. Источником импульсов преобразования АЦП может быть один из трех источников: $2T_{OSC}$, $8T_{OSC}$, $32T_{OSC}$ (RC-генератор не может использоваться).

Время преобразования может быть вычислено по формуле:

Время преобразования = $T_{AD} + N \bullet T_{AD} + (10 - N)(2T_{OSC})$,
где N - разрядность АЦП.

Из-за того, что время T_{AD} зависит от частоты кварцевого генератора микроконтроллера, пользователь должен использовать какой-нибудь иной методов (например, используя таймер, программный цикл и др.) для определения времени изменения. В примере 21-3 представлено длительность 4 - разрядного преобразования, против 8 - разрядного. Например, для микроконтроллера с тактовой частотой 20МГц (тактовый сигнал АЦП $32T_{OSC}$). Через $5T_{AD}$, после старта преобразования, тактовый сигнал АЦП программно переключается на $2T_{OSC}$.

При $2T_{OSC}$ не выдержано минимальное требование к T_{AD}, поэтому 4 младших бита будут иметь неверное значение.

Пример 21-3 Расчет длительности 4 - разрядного преобразования

	Частота(1)	**Разрешение**	
		4 бита	**8 бит**
T_{AD}	20МГц	1.6 мкс	1.6мкс
T_{OSC}	20МГц	50 нс	50 нс
$T_{AD} + N \bullet T_{AD} + (10 - N)(2T_{OSC})$	20МГц	8.6 мкс	17.6 мкс

Примечания:

1. Минимальное рекомендуемое значение T_{AD} = 1.6мкс.
2. Если требуется полное 8 - разрядное значение преобразования, источник тактовых импульсов для модуля АЦП не должен изменяться.

21.8 Работа модуля АЦП в SLEEP режиме микроконтроллера

Модуль АЦП может работать в SLEEP режиме микроконтроллера при условии, что источником импульсов преобразования АЦП будет внутренний RC генератор (ADCS1:ADCS0=11). При выборе RC генератора импульсов модуль АЦП сделает задержку в один машинный цикл перед началом преобразования. Это позволяет программе пользователя выполнить команду SLEEP, тем самым уменьшить “цифровой шум” во время преобразования. После завершения преобразования аппаратно сбрасывается бит GO/-DONE в '0', результат преобразования сохраняется в регистре ADRES. Если разрешено прерывание от АЦП, то микроконтроллер выйдет из режима SLEEP. Если же прерывание было запрещено, то после преобразования модуль АЦП будет выключен, хотя бит ADON останется установленным.

Если был выбран другой источник тактовых импульсов АЦП (не внутренний RC генератор), то выполнение программой инструкции SLEEP прервет процесс преобразования и выключит модуль АЦП, оставив установленным бит ADON. Выключение модуля АЦП уменьшит ток потребления микроконтроллера.

Примечание. Для работы модуля АЦП в SLEEP режиме необходимо выбрать внутренний RC генератор (ADCS1:ADCS0=11), инструкция SLEEP должна быть выполнена сразу после команды, устанавливающей бит GO/-DONE в '1'.

21.9 Точность преобразования АЦП

Абсолютная точность АЦП определяется суммарной ошибкой, исходя из ошибки дискретизации, интегральной ошибки, ошибки шкалы, ошибки смещения и монотонности. Суммарная ошибка определяется как максимальный разброс между текущим и идеальным результатом для любого значения. Абсолютная ошибка АЦП меньше ± 1 значащего бита при V_{DD}=V_{REF}, но она возрастает при отклонении V_{REF} от V_{DD}.

В некотором диапазоне напряжений на аналоговом входе цифровой результат будет один и тот же. Это возникает из-за дискретизации, которая неизбежна при преобразовании аналоговой величины в цифровую форму. Ошибка дискретизации составляет ± ½ значащего бита, и единственный способ уменьшить ее - увеличить разрядность АЦП.

Ошибку смещения составляет разность между результатом первого преобразования и идеальным значением. Эта ошибка сдвигает всю передаточную функцию, и может быть учтена при помощи калибровки. Ошибка вносится в результате наложения токов утечки и выходного сопротивления источника сигнала.

Ошибка усиления измеряется как максимальное отклонение результата, скорректированного с учетом ошибки смещения. Эта ошибка проявляется в виде изменения наклона передаточной функции. Ошибка усиления может быть откалибрована и учтена.

Ошибка линейности определяется как разница в приращении входного напряжения для получения одинакового приращения выходного кода и не поддается калибровке. Интегральная ошибка вычисляется как отклонение результата, скорректированного с учетом ошибки усиления.

Дифференциальная ошибка вычисляется как отклонение максимальной длины кода результат от идеальной длины кода без учета других ошибок.

Максимальный ток утечки вывода смотрите в разделе "Электрические характеристики" параметр D060.

В системах с низкой тактовой частотой предпочтительно использование встроенного RC генератора. В системах с высокой рабочей частотой следует использовать тактовый сигнал от основного генератора. Предпочтительно использовать АЦП с T_{AD} не больше 8 мкс, но не меньше рекомендованного нижнего предела. Использование тактового сигнала от основного генератора позволяет снизить влияние шумов от переключения внутренних вентилей, т.к. переключение логики АЦП происходит синхронно с другими устройствами, что невозможно при использовании встроенного RC генератора. Если каналы цифрового ввода/вывода постоянно активны, потеря точности из-за шумов при переключении может быть значительной.

В случае использования АЦП в SLEEP режиме, источником тактового сигнала должен быть встроенный RC генератор. В этом режиме отсутствуют цифровые шумы, т.к. другие узлы микроконтроллера остановлены, поэтому точность преобразования получается высокой.

21.10 Эффект сброса

При сбросе микроконтроллера значения всех его регистров устанавливаются по умолчанию. Сброс выключает модуль АЦП, а также останавливает процесс преобразования, если он был начат. Все выводы, используемые модулем АЦП, настраиваются как аналоговые входы. Регистр ADRES после сброса POR будет содержать неизвестное значение, а после остальных видов сброса не изменит своего значения.

21.11 Использование CCP триггера

Аналого-цифровое преобразование может быть запущено при помощи “триггера специального события” модуля CCP. Для этого необходимо, чтобы биты CCPxM3:CCPxM0 (CCPxCON<3:0>) были запрограммированы как 1011 и был включен модуль АЦП (бит ADON должен быть установлен в '1'). При срабатывании триггера бит GO/-DONE будет установлен в '1', тем самым, запуская преобразование, а содержимое таймера TMR1 будет обнулено. Таймер сбрасывается и автоматически повторяет запуск преобразования через определенные промежутки времени. Пользователю необходимо будет только вовремя считывать содержимое регистра ADRES. До начала преобразования необходимо выбрать соответствующий аналоговый канал, прежде чем “триггер специального события“ вызовет установку бита GO/-DONE.

При выключенном модуле АЦП (бит ADON сброшен в '0') сигнал “триггера специального события” игнорируется, но таймер TMR1 продолжает работать и сбрасываться.

21.12 Подключение к модулю АЦП

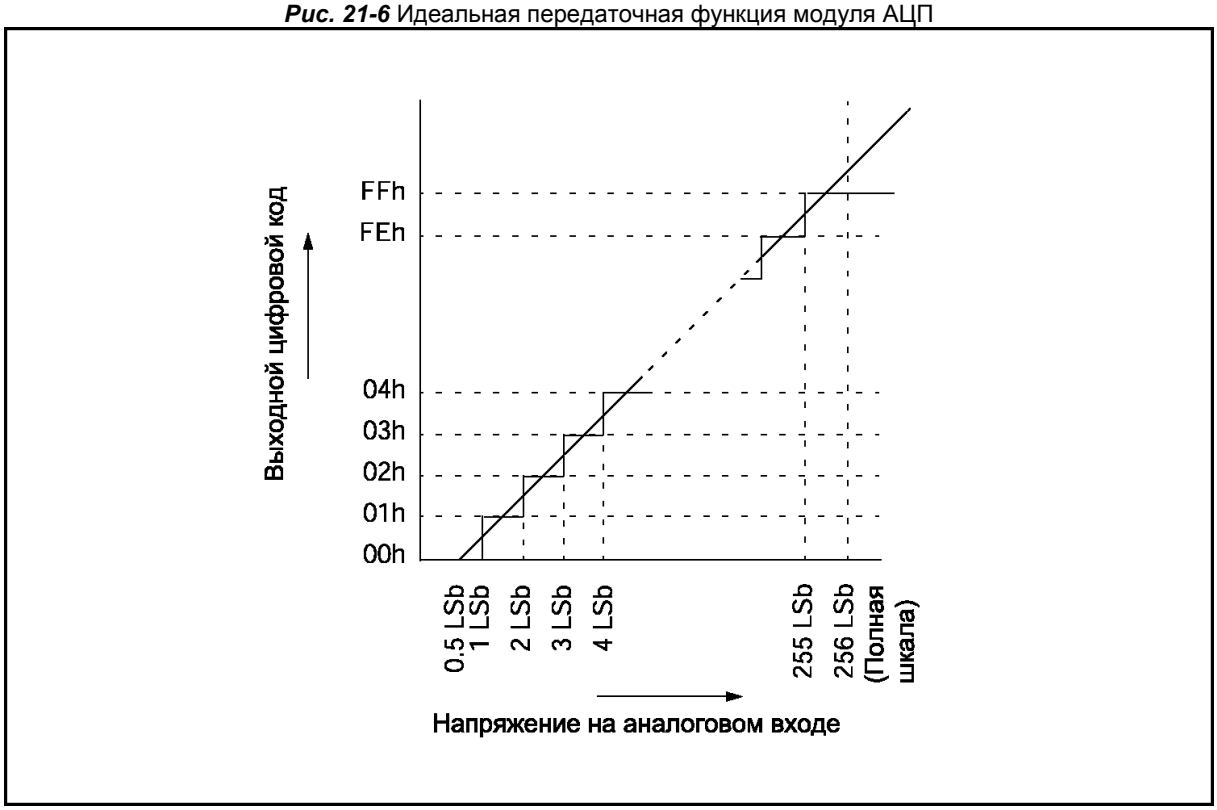
Если значение входного напряжения превышает на 0.2В величину порога питающих напряжений (V_{SS} и V_{DD}), то точность преобразования выйдет за пределы значений, оговоренных в спецификации.

Иногда, для сглаживания пульсаций входного сигнала, на вход АЦП добавляется внешняя RC цепочка. Значение сопротивления R должно выбираться так, чтобы общее сопротивление источника сигнала было в пределах рекомендованной величины 10кОм. Любой внешний электронный компонент, подключенный к аналоговому входу(конденсатор, стабилитрон и др.), должны иметь низкий ток утечки через вывод.

21.13 Передаточная функция модуля АЦП

Идеальная функция модуля АЦП работает по следующему правилу: первый бит значения измеряемой аналоговой величины будет установлен, если входное напряжение (V_{AIN}) на аналоговом входе будет равно 1 Lsb (V_{REF}/256) (см. рисунок 21-6).

Рис. 21-6 Идеальная передаточная функция модуля АЦП



21

21.14 Инициализация

В примере 21-4 показана инициализация модуля АЦП микроконтроллера PIC16C74A.

Пример 21-4 Инициализация модуля АЦП (для PIC16C74A)

```
            BSF         STATUS, RP0   ; Выбрать банк 1
            CLRF        ADCON1        ; Настроить входы АЦП
            BSF         PIE1, ADIE    ; Разрешить прерывания от АЦП
            BCF         STATUS, RP0   ; Выбрать банк 0
            MOVLW       0xC1          ; Тактовые импульсы от RC генератора АЦП,
            MOVWF       ADCON0        ; включить АЦП, выбрать канал 0
            BCF         PIR1, ADIF    ; Сбросить флаг прерываний от АЦП
            BSF         INTCON, PEIE  ; Разрешить периферийные прерывания
            BSF         INTCON, GIE   ; Разрешить прерывания в системе
;
; Выдержать паузу, необходимую для заряда внутреннего конденсатора CHOLD.
; Затем начинать преобразование АЦП.
;
            BSF         ADCON0, GO    ; Старт преобразования
            :             ; Ожидать установку флага ADIF или сброс
            :             ; бита GO/-DONE по завершению преобразования
```

21.15 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Я использую микроконтроллер PIC16C7XX в своем устройстве. Заметил, что результат преобразования АЦП не всегда точен. Что можно сделать?

Ответ 1:

1. Удостоверьтесь, что Вы выполняете требования синхронизации АЦП. Если Вы постоянно включаете/выключаете АЦП или изменяете входной канал, то необходимо обеспечить минимальную задержку перед началом преобразования. Значение T_{AD} (время получение одного бита) должно быть от 2мкс до 6мкс (устанавливается в регистре ADCON0). Если T_{AD} слишком мало, входное напряжение не может быть полностью преобразовано. Если T_{AD} достаточно большое, то напряжение на конденсаторе C_{HOLD} может измениться за счет внутренних токов утечки. Выбор параметров синхронизации описан в технической документации. С помощью представленных формул можно вычислить параметры синхронизации для конкретного случая.
2. Достаточно часто внутреннее сопротивление источника сигнала имеет большое значение (больше 10кОм), поэтому ток, заряжающий конденсатор C_{HOLD}, может влиять на точность преобразования. Если входной сигнал быстро не изменяется, то можно к аналоговому входу подключить конденсатор 0.1мкФ. Этот конденсатор зарядиться до напряжения аналогового сигнала и при выборки обеспечит требуемый кратковременный ток заряда внутреннего конденсатора C_{HOLD} (51.2пФ).
3. Из технической документации: " В системах с низкой тактовой частотой предпочтительно использование встроенного RC генератора. В системах с высокой рабочей частотой следует использовать тактовый сигнал от основного генератора. Предпочтительно использовать АЦП с T_{AD} не больше 8 мкс, но не меньше рекомендованного нижнего предела. Использование тактового сигнала от основного генератора позволяет снизить влияние шумов от переключения внутренних вентилей, т.к. переключение логики АЦП происходит синхронно с другими устройствами, что невозможно при использовании встроенного RC генератора. В случае использования АЦП в SLEEP режиме, источником тактового сигнала должен быть встроенный RC генератор. В этом режиме отсутствуют цифровые шумы, т.к. другие узлы микроконтроллера остановлены, поэтому точность преобразования получается высокой."

Вопрос 2: После старта преобразования АЦП могу я изменить входной канал для следующего измерения?

Ответ 2:

После установки в '1' бита GO (старт преобразования) конденсатор C_{HOLD} отключается от аналогового входа через T_{AD}. Как только конденсатор отсоединился от аналогового входа, может быть выбран другой канал.

Вопрос 3: Посоветуйте хорошую книгу по теории АЦП.

Ответ 3:

“Analog-Digital Conversion Handbook” 3-я редакция, издательство Prentice Hall (ISBN 0-13-03-2848-0).

21.16 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с модулем АЦП в микроконтроллерах PICmicro MCU:

Документ	Номер
Using the Analog to Digital Converter Применение АЦП	AN546
Four Channel Digital Voltmeter with Display and Keyboard Четырех канальный вольтметр с дисплеем и клавиатурой	AN557

Раздел 22. Основной модуль 8 - разрядного АЦП

Содержание

22.1 Введение

Модуль аналого-цифрового преобразования (АЦП) имеет до четырех входных каналов.

Входной аналоговый сигнал через коммутатор каналов заряжает внутренний конденсатор АЦП C_{HOLD}. Модуль АЦП преобразует напряжение, удерживаемое на конденсаторе C_{HOLD} в соответствующий 8 - разрядный цифровой код методом последовательного приближения. Источник опорного напряжения может быть программно выбран с вывода V_{DD} или AN3/V_{REF}. Допускается работа модуля АЦП в SLEEP режиме микроконтроллера, при этом в качестве источника тактовых импульсов для АЦП должен быть выбран RC генератор.

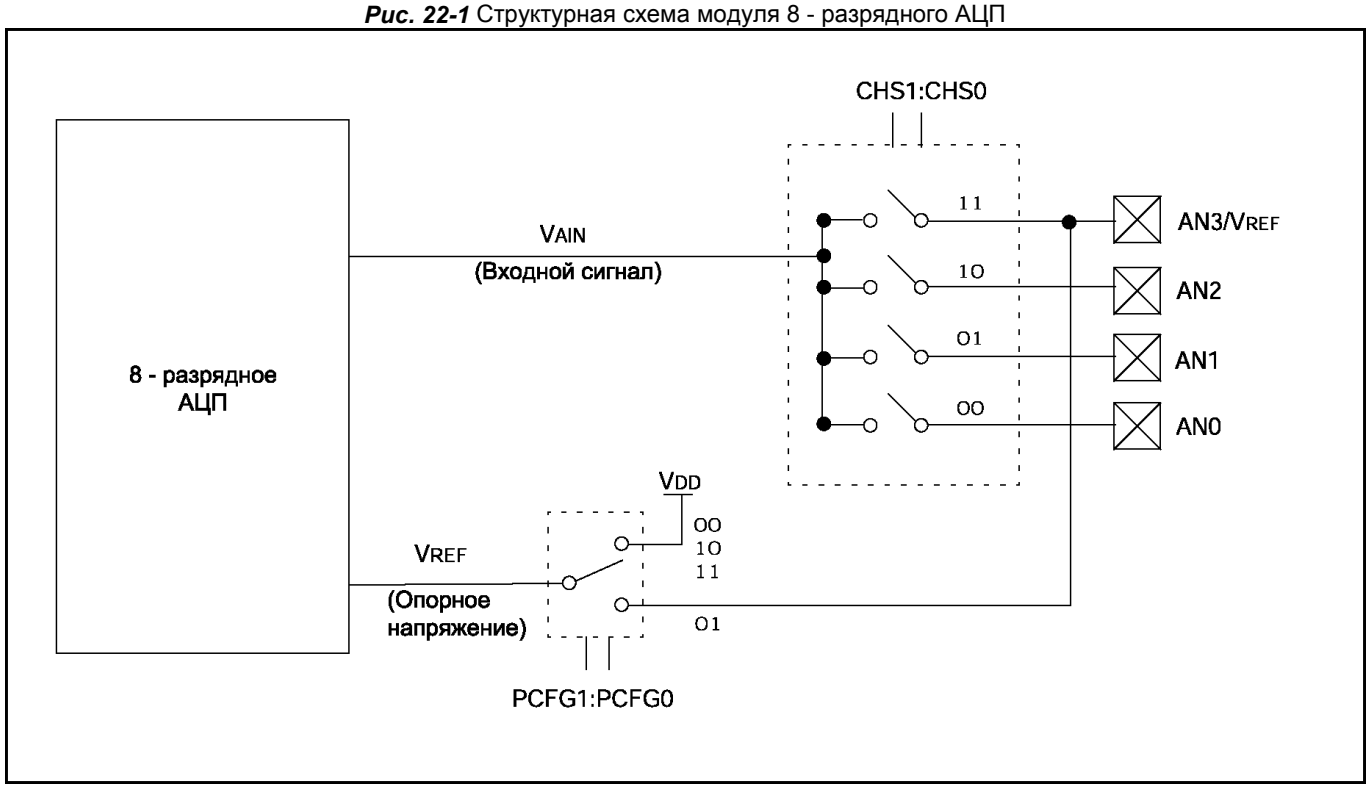
Для управления АЦП в микроконтроллере используется 3 регистра:

- Регистр результата ADRES;
- Регистр управления ADCON0;
- Регистр управления ADCON1.

Регистр ADCON0 используется для настройки работы модуля АЦП, а с помощью регистра ADCON1 устанавливается, какие входы микроконтроллера будут использоваться модулем АЦП и в каком режиме (аналоговый вход или цифровой порт ввода/вывода).

Структурная схема модуля АЦП показана на рисунке 22-1.

Рис. 22-1 Структурная схема модуля 8 - разрядного АЦП



22.2 Управляющие регистры

Регистр ADCON0:

R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADCS1	**ADCS0**	**-**(1)	**CHS1**	**CHS0**	**GO/-DONE**	**ADIF/-**(2)	**ADON**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как '0'
–n – значение после POR
–x – неизвестное значение после POR

биты 7-6: **ADCS1:ADCS0:** Выбор источника тактового сигнала
00 = $F_{OSC}/2$
01 = $F_{OSC}/8$
10 = $F_{OSC}/32$
11 = F_{RC} (внутренний RC генератор модуля АЦП)

бит 5: **Не используется:** читается как ‘0’

биты 4-3: **CHS1:CHS0:** Выбор аналогового канала
00 = канал 0, (AN0)
01 = канал 1, (AN1)
10 = канал 2, (AN2)
11 = канал 3, (AN3)

бит 2: **GO/-DONE:** Бит статуса модуля АЦП
Если ADON=1
1 = модуль АЦП выполняет преобразование (установка бита вызывает начало преобразования)
0 = состояние ожидания (аппаратно сбрасывается по завершению преобразования)

бит 2: **ADIF:** Флаг прерывания от модуля АЦП
1 = преобразование завершено (сбрасывается в '0' программно)
0 = преобразование не завершено

бит 0: **ADON:** Бит включения модуля АЦП
1 = модуль АЦП включен
0 = модуль АЦП выключен и не потребляет тока

Примечание 1. Для PIC16C71 - бит 5 регистра ADCON0 является универсальным и доступен для записи и чтения. В микроконтроллерах PIC16C710/711/715 этот бит не реализован и читается как '0'.

Примечание 2. В микроконтроллерах PIC12CXXX этот бит сохранен, для других микроконтроллеров бит ADIF реализован в регистре PIR. Не рекомендуется использовать этот бит как бит общего назначения. Всегда поддерживайте этот бит сброшенным в '0'.

Регистр ADCON1:

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
-	-	-	-	-	**-/PCFG2(1)**	**PCFG1**	**PCFG0**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как '0'
–n – значение после POR
–x – неизвестное значение после POR

биты 7-2: **Не используются:** читаются как ‘0’

Примечание. В некоторых микроконтроллерах реализован бит PCFG2.

биты 1-0: **PCFG1:PCFG0:** Управляющие биты настройки каналов АЦП

PCFG1:PCFG0	AN3	AN2	AN1	AN0
00	A	A	A	A
01	V_{REF+}	A	A	A
10	D	D	A	A
11	D	D	D	D

A = аналоговый вход
D = цифровой канал ввода/вывода

Примечание. Когда вывод AN3 работает как V_{REF+}, то напряжение на AN3 является опорным для АЦП. Если вывод AN3 работает как аналоговый вход (A), то опорное напряжение для АЦП берется с вывода V_{DD}.

биты 2-0: **PCFG2:PCFG0:** Управляющие биты настройки каналов АЦП(1)

PCFG2:PCFG0	AN3	AN2	AN1	AN0
000	A	A	A	A
001	A	A	V_{REF+}	A
010	D	A	A	A
011	D	A	V_{REF+}	A
100	D	D	A	A
101	D	D	V_{REF+}	A
110	D	D	D	A
111	D	D	D	D

A = аналоговый вход
D = цифровой канал ввода/вывода

Примечание. Когда вывод AN1 работает как V_{REF+}, то напряжение на AN1 является опорным для АЦП. Если вывод AN1 работает как аналоговый вход (A), то опорное напряжение для АЦП берется с вывода V_{DD}.

Примечание 1. Некоторые микроконтроллеры содержат дополнительный бит настройки портов ввода/вывода PCFG2. Этот бит особенно важен в 8 - выводных микроконтроллерах АЦП, в которых число портов ввода/вывода ограничено. В других микроконтроллерах этот бит не реализован и читается как '0'.

Примечание 2. При сбросе микроконтроллера все выводы мультиплицированные с модулем АЦП (ANx) настраиваются как аналоговые входы.

22.3 Работа модуля АЦП

В регистре ADRES сохраняется 8-разрядный результат аналого-цифрового преобразования. Когда преобразование завершено, результат преобразования записывается в регистр ADRES, после чего сбрасывается бит GO/-DONE (ADCON0<2>) и устанавливается флаг прерывания ADIF.

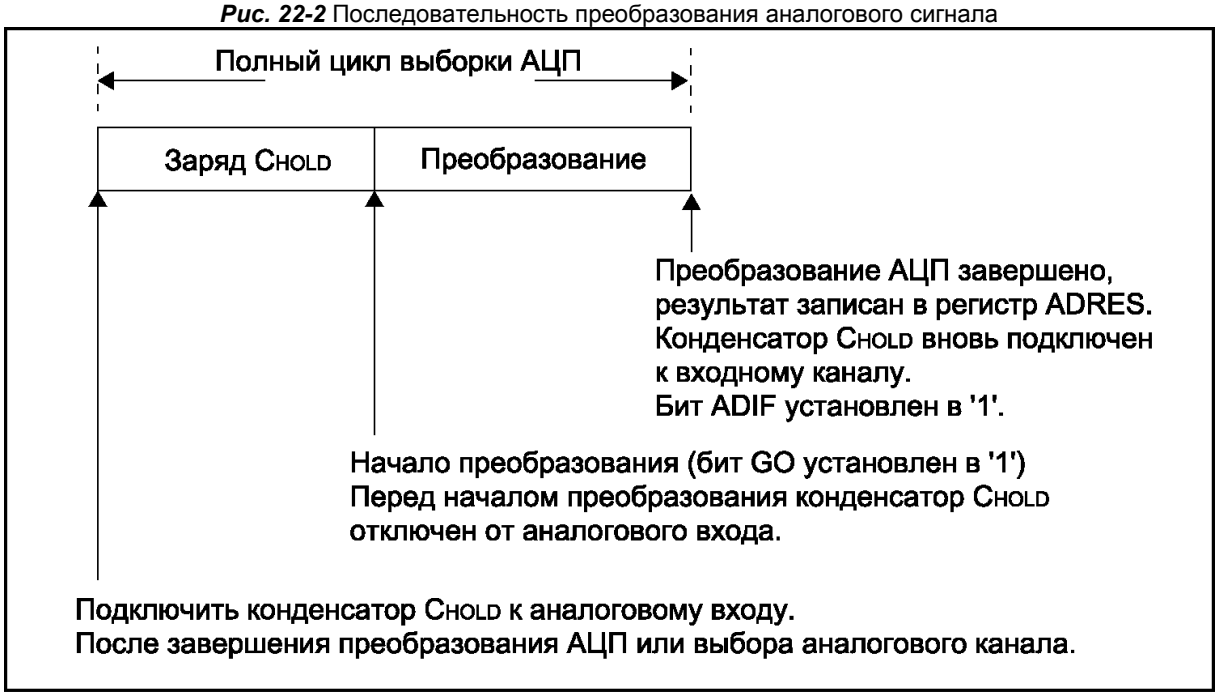
После включения и настройки АЦП необходимо выбрать рабочий аналоговый канал. Соответствующие биты TRIS аналоговых каналов должны настраивать канал порта ввода/вывода на вход. Перед началом преобразования необходимо выдержать временную паузу, расчет которой приведен в разделе 22.4.

Рекомендованная последовательность действий для работы с АЦП:

1. Настроить модуль АЦП:
   - Настроить выводы как аналоговые входы, входы V_{REF} или цифровые каналы ввода/вывода (ADCON1);
   - Выбрать входной канал АЦП (ADCON0);
   - Выбрать источник тактовых импульсов для АЦП (ADCON0);
   - Включить модуль АЦП (ADCON0).
2. Настроить прерывание от модуля АЦП (если необходимо):
   - Сбросить бит ADIF в '0';
   - Установить бит ADIE в '1';
   - Установить бит PEIE в '1';
   - Установить бит GIE в '1'.
3. Выдержать паузу, необходимую для зарядки конденсатора C_{HOLD}.
4. Начать аналого-цифровое преобразование:
   - Установить GO/-DONE бит в '1' (ADCON0).
5. Ожидать, окончания преобразования:
   - Ждать, пока бит GO/-DONE не будет сброшен в '0'; ИЛИ
   - Ожидать прерывание по окончанию преобразования.
6. Считать результат преобразования из регистра ADRES, сбросить бит ADIF в '0', если это необходимо.
7. Для следующего преобразования необходимо выполнить шаги начиная с пункта 1 или 2. Время преобразования одного бита определяется как время T_{AD}. Минимальное время ожидания перед следующим преобразованием должно составлять $2T_{AD}$.

На рисунке 22-2 показана последовательность преобразования аналогового сигнала. Время заряда C_{HOLD} - интервал времени в течение которого на внутренний конденсатор АЦП подается внешний сигнал. Время преобразования равно 10 T_{AD}, отсчет начинается с момента установки в '1' бита GO. Сумма этих двух временных интервалов является длительностью полного цикла преобразования АЦП. Существует минимальный интервал времени, в течение которого внешний сигнал подается на внутренний конденсатор C_{HOLD}, чтобы гарантировать требуемую точность АЦП.

Рис. 22-2 Последовательность преобразования аналогового сигнала



22.4 Временные требования к подключению канала АЦП

Для обеспечения необходимой точности преобразования, конденсатор C_{HOLD} должен успевать полностью заряжаться до уровня входного напряжения. Схема аналогового входа АЦП показана на рисунке 22-3. Сопротивления R_S и R_{SS} непосредственно влияют на время зарядки конденсатора C_{HOLD}. Величина сопротивления ключа выборки (R_{SS}) зависит от напряжения питания V_{DD} (см. рисунок 22-3). **Максимальное рекомендуемое значение внутреннего сопротивления источника аналогового сигнала 10кОм.** При меньших значениях сопротивления источника сигнала меньше суммарное время преобразования.

После того, как будет выбран один из нескольких аналоговых входных каналов, но прежде, чем будет производиться преобразование, должно пройти определенное время. Для нахождения данного времени воспользуетесь уравнением 22-1. Это уравнение дает результат с ошибкой в ½ LSb (512 шагов АЦП). Ошибка в ½ LSb, это максимальная погрешность, позволяющая функционировать модулю АЦП с необходимой точностью.

Уравнение 22-1 Вычисление временной задержки

T_{ACQ} = Время задержки усилителя + Время заряда конденсатора C_{HOLD} + Температурный коэффициент
$= T_{AMP} + T_C + T_{COFF}$

Уравнение 22-2 Минимальное время заряда конденсатора C_{HOLD}

$$V_{HOLD} = (V_{REF} - (V_{REF}/512)) \bullet (1 - e^{(-Tc/Chold(Ric + Rss + Rs))})$$

$$T_C = - 51.2пФ\ (1кОм + R_{SS} + R_S)\ Ln(1/511)$$

В примере 22-1 показано вычисление минимального значения времени T_{ACQ}. Вычисления основываются на следующих исходных данных:

Rs	= 10кОм
Ошибка преобразования	≤ 1/2 Lsb
V_{DD}	= 5В → Rss = 7 кОм (см. график на рисунке 22-3)
Температура	= 50°C (максимально возможная)
V_{HOLD}	= 0В @ t = 0

Пример 22-1 Вычисление минимального значения времени T_{ACQ}

$T_{ACQ} = T_{AMP} + T_C + T_{COFF}$
T_{ACQ} = 5мкс + T_C + [(Температура - 25°C)(0.05мкс/°C)]

T_C = - C_{HOLD} (R_{IC} + R_{SS} + R_S) Ln(1/511)
= - 51.2пФ (1кОм + 7кОм + 10кОм) Ln(0.0020)
= - 51.2пФ (18кОм) Ln(0.0020)
= -0.921мкс (-6.2146)
= 5.724мкс

T_{ACQ} = 5мкс + 5.724мкс + [(50°C - 25°C)(0.05мкс/°C)]
= 10.724мкс + 1.25мкс
= 11.974мкс

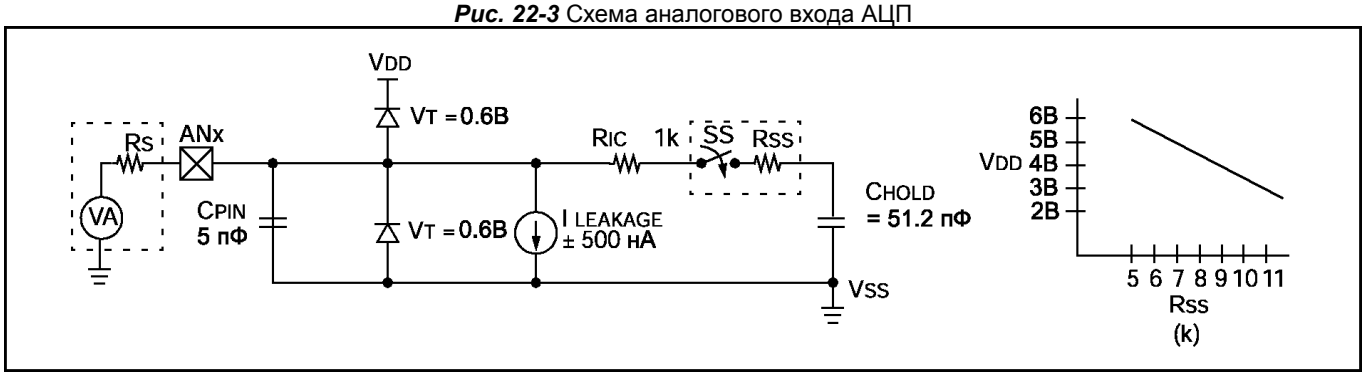
Примечание 1. Напряжение источника опорного напряжения V_{REF} не влияет на уравнение.

Примечание 2. Конденсатор C_{HOLD} после преобразования не разряжается.

Примечание 3. Максимальное рекомендуемое значение внутреннего сопротивления источника аналогового сигнала 10кОм. Это необходимо для компенсации внутреннего тока утечки.

Примечание 4. После того, как преобразование завершено, необходимо программно обеспечить задержку не менее $2.0T_{AD}$, прежде чем начнете следующее преобразование. В течение этого времени конденсатор C_{HOLD} не подключен к выбранному входному каналу АЦП.

Рис. 22-3 Схема аналогового входа АЦП



Обозначения:

C_{PIN}	- входная емкость;
V_T	- пороговое напряжение;
$I_{LEAKAGE}$	- ток утечки вывода;
R_{IC}	- сопротивление соединения;
SS	- переключатель защелки;
C_{HOLD}	- конденсатор защелки.

22.5 Выбор источника тактовых импульсов для АЦП

Время получения одного бита результата равно T_{AD}. Для 8-разрядного результата требуется как минимум $9.5T_{AD}$. Параметры тактового сигнала для АЦП определяются программно, T_{AD} может принимать следующие значения:

- $2T_{OSC}$;
- $8T_{OSC}$;
- $32T_{OSC}$;
- Внутренний RC генератор модуля АЦП (2-6мкс).

Для получения корректного результата преобразования необходимо выбрать источник тактового сигнала АЦП, обеспечивающий время T_{AD} не менее 1.6 мкс (для микроконтроллеров PIC16C71 не менее 2мкс) (см. параметр 130 в разделе "Электрические характеристики").

В таблицах 22-1 по 22-4 указано максимальное значение тактовой частоты микроконтроллера для каждого режима синхронизирующего сигнала АЦП.

Таблица 22-1 Максимальное значение F_{OSC} удовлетворяющие требованию к T_{AD}
(для микроконтроллеров с нормальным диапазоном напряжения питания (C)) (кроме PIC16C71)

Источник импульсов АЦП		**Рабочая частота F_{osc}**			
Источник	**ADCS1:ADCS0**	**20МГц**	**5МГц**	**1.25МГц**	**333.33кГц**
$2T_{OSC}$	00	100нс(2)	400нс(2)	1.6 мкс	6мкс
$8T_{OSC}$	01	400нс(2)	1.6 мкс	6.4 мкс	24мкс(3)
$32T_{OSC}$	10	1.6 мкс	6.4 мкс	25.6 мкс(3)	96мкс(3)
RC-генератор	11	2-6мкс(1,4)	2-6мкс(1,4)	2-6мкс(1,4)	2-6мкс(1)

Обозначение: Затененные ячейки - не рекомендованное значение.
Примечания:

1. Типовое значение времени T_{AD} RC генератора АЦП равно 4мкс, может варьироваться от 2мкс до 6мкс.
2. Это значение выходит за пределы минимально допустимого времени T_{AD}.
3. Для более точного преобразования рекомендуется выбрать другой источник тактовых импульсов.
4. Когда тактовая частота микроконтроллера больше 1МГц, рекомендуется использовать RC генератор АЦП только для работы в SLEEP режиме.

Таблица 22-2 Максимальное значение F_{OSC} удовлетворяющие требованию к T_{AD}
(для микроконтроллеров с расширенным диапазоном напряжения питания (LC)) (кроме PIC16LC71)

Источник импульсов АЦП		**Рабочая частота F_{osc}**			
Источник	**ADCS1:ADCS0**	**4МГц**	**2МГц**	**1.25МГц**	**333.33кГц**
$2T_{OSC}$	00	500нс(2)	1.0мкс(2)	1.6 мкс	6мкс
$8T_{OSC}$	01	2.0мкс(2)	4.0 мкс	6.4 мкс	24мкс(3)
$32T_{OSC}$	10	8.0 мкс	16.0 мкс	25.6 мкс(3)	96мкс(3)
RC-генератор	11	2-6мкс(1,4)	2-6мкс(1,4)	2-6мкс(1,4)	2-6мкс(1)

Обозначение: Затененные ячейки - не рекомендованное значение.
Примечания:

1. Типовое значение времени T_{AD} RC генератора АЦП равно 4мкс, может варьироваться от 2мкс до 6мкс.
2. Это значение выходит за пределы минимально допустимого времени T_{AD}.
3. Для более точного преобразования рекомендуется выбрать другой источник тактовых импульсов.
4. Когда тактовая частота микроконтроллера больше 1МГц, рекомендуется использовать RC генератор АЦП только для работы в SLEEP режиме.

Таблица 22-3 Максимальное значение F_{OSC} удовлетворяющие требованию к T_{AD} для PIC16C71 (нормальный диапазон напряжения питания (C))

Источник импульсов АЦП		**Рабочая частота F_{osc}**				
Источник	**ADCS1:ADCS0**	**20МГц**	**16МГц**	**4МГц**	**1МГц**	**333.33кГц**
$2T_{OSC}$	00	100нс(2)	125нс(2)	500нс(2)	2.0 мкс	6мкс
$8T_{OSC}$	01	400нс(2)	500нс(2)	2.0 мкс	8.0 мкс	24мкс(3)
$32T_{OSC}$	10	1.6 мкс(2)	2.0 мкс	8.0 мкс	32.0 мкс(3)	96мкс(3)
RC-генератор	11	2-6мкс(1,4)	2-6мкс(1,4)	2-6мкс(1,4)	2-6мкс(1,4)	2-6мкс(1)

Обозначение: Затененные ячейки - не рекомендованное значение.

Примечания:

1. Типовое значение времени T_{AD} RC генератора АЦП равно 4мкс, может варьироваться от 2мкс до 6мкс.
2. Это значение выходит за пределы минимально допустимого времени T_{AD}.
3. Для более точного преобразования рекомендуется выбрать другой источник тактовых импульсов.
4. Когда тактовая частота микроконтроллера больше 1МГц, рекомендуется использовать RC генератор АЦП только для работы в SLEEP режиме.

Таблица 22-4 Максимальное значение F_{OSC} удовлетворяющие требованию к T_{AD} для PIC16LC71 (расширенный диапазон напряжения питания (LC))

Источник импульсов АЦП		**Рабочая частота F_{osc}**			
Источник	**ADCS1:ADCS0**	**4МГц**	**2МГц**	**1.25МГц**	**333.33кГц**
$2T_{OSC}$	00	500нс(2)	1.0мкс(2)	1.6 мкс	6мкс
$8T_{OSC}$	01	2.0мкс(2)	4.0 мкс	6.4 мкс	24мкс(3)
$32T_{OSC}$	10	8.0 мкс	16.0 мкс	25.6 мкс(3)	96мкс(3)
RC-генератор	11	2-6мкс(1,4)	2-6мкс(1,4)	2-6мкс(1,4)	2-6мкс(1)

Обозначение: Затененные ячейки - не рекомендованное значение.

Примечания:

1. Типовое значение времени T_{AD} RC генератора АЦП равно 4мкс, может варьироваться от 2мкс до 6мкс.
2. Это значение выходит за пределы минимально допустимого времени T_{AD}.
3. Для более точного преобразования рекомендуется выбрать другой источник тактовых импульсов.
4. Когда тактовая частота микроконтроллера больше 1МГц, рекомендуется использовать RC генератор АЦП только для работы в SLEEP режиме.

22.6 Настройка аналоговых входов

Регистры ADCON1 и TRIS отвечают за настройку выводов АЦП. Если выводы микросхемы настраиваются как аналоговые входы, то при этом должны быть установлены соответствующие биты в регистре TRIS. Если соответствующий бит сброшен в '0', вывод микросхемы настроен как цифровой выход, со значениями выходных напряжений V_{OH} или V_{OL}.

Модуль АЦП функционирует независимо от состояния битов CHS2:CHS0 и битов TRIS.

Примечание 1. При чтении содержимого регистра порта нули будут установлены в тех разрядах, которые были настроены как аналоговые входы. Настроенные на цифровой вход каналы будут преобразовывать входные аналоговые уровни в цифровые, что однако не окажет влияния на точность преобразования.

Примечание 2. Значения напряжений, подаваемых на выводы, настроены как аналоговые входы, включая выводы (AN3:AN0), могут влиять на ток потребления входного буфера, который может выйти за пределы значений, оговоренных в технической спецификации.

22.7 Аналого-цифровое преобразование

В примере 22-2 показана последовательность действий для работы с АЦП. Выводы настроены как аналоговые входы. Источник опорного напряжения – V_{DD}. Разрешены прерывания от модуля АЦП. Источником импульсов преобразования является RC генератор АЦП. Аналоговое цифровое преобразование выполняется с вывода AN0.

Примечание. Бит GO/-DONE и бит включения АЦП должны устанавливаться разными командами.

Сброс бита GO/-DONE в '0' во время преобразования приведет к его прекращению. При этом регистр результата (ADRES) не изменит своего содержимого. После досрочного завершения преобразования необходимо обеспечить временную задержку $2T_{AD}$. Выдержав требуемую паузу, автоматически начинает заряжаться конденсатор C_{HOLD} с выбранного аналогового канала.

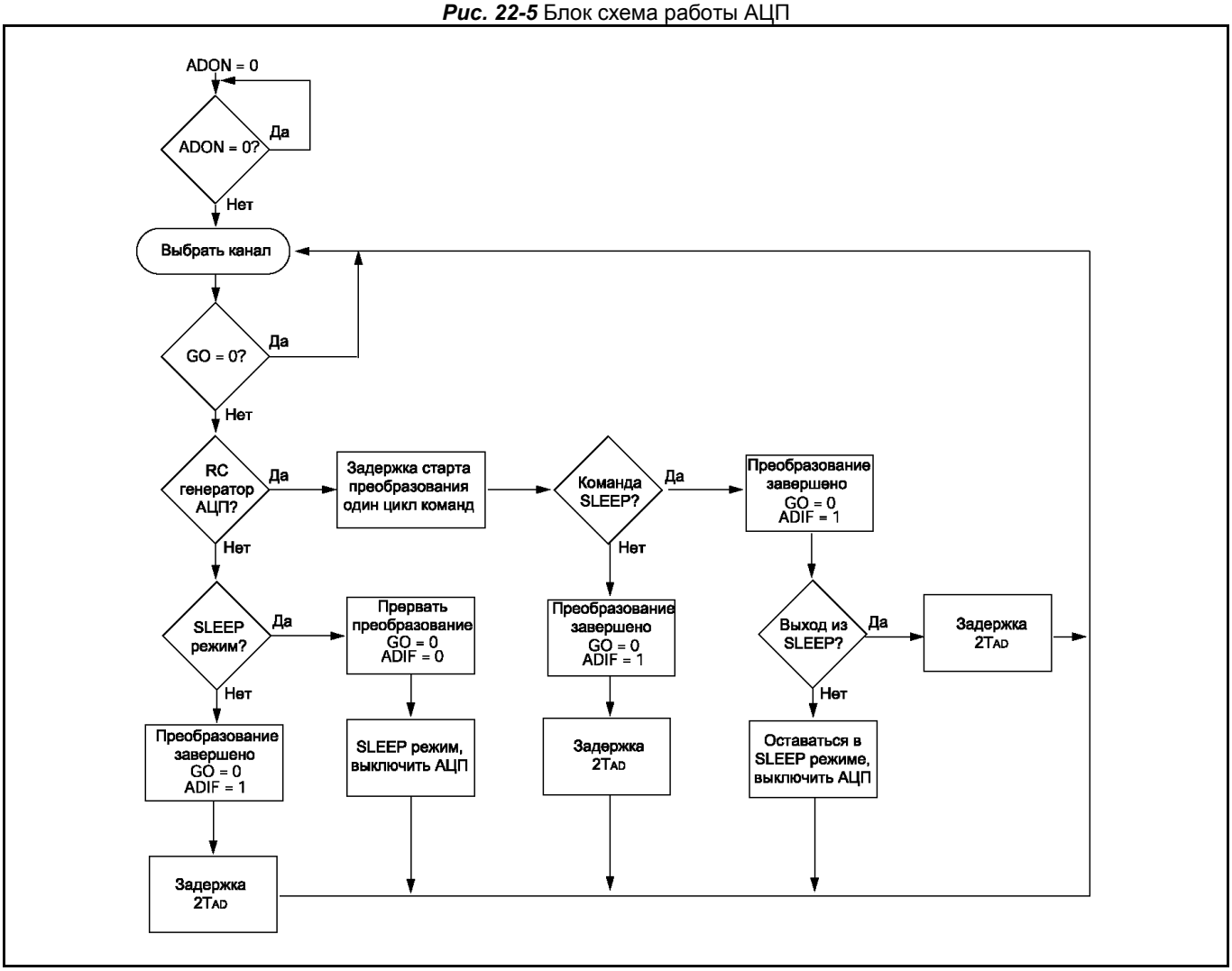
На рисунке 22-4 показана последовательность получения результата после установки бита GO/-DONE в '1'.

Пример 22-2 Выполнение преобразования АЦП

```
            BSF         STATUS, RP0   ; Выбрать банк 1
            CLRF        ADCON1        ; Настроить входы АЦП
            BSF         PIE1, ADIE    ; Разрешить прерывания от АЦП
            BCF         STATUS, RP0   ; Выбрать банк 0
            MOVLW       0xC1          ; Тактовые импульсы от RC генератора АЦП,
            MOVWF       ADCON0        ; включить АЦП, выбрать канал 0
            BCF         PIR1, ADIF    ; Сбросить флаг прерываний от АЦП
            BSF         INTCON, PEIE  ; Разрешить периферийные прерывания
            BSF         INTCON, GIE   ; Разрешить прерывания в системе
;
; Выдержать паузу, необходимую для заряда внутреннего конденсатора CHOLD.
; Затем начинать преобразование АЦП.
;
            BSF         ADCON0, GO    ; Старт преобразования
            :                         ; Ожидать установку флага ADIF или сброс
            :                         ; бита GO/-DONE по завершению преобразования
```

Рис. 22-4 Последовательность получения результата после установки бита GO/-DONE в '1'

Рис. 22-5 Блок схема работы АЦП



22.7.1 Быстрое преобразование взамен разрешающей способности

Иногда бывает необходимо быстрое по времени преобразование за счет снижения разрешающей способности АЦП. Эта особенность работы модуля АЦП может использоваться в программе пользователя. Независимо от используемой разрешающей способности интервал времени T_{ACQ} остается неизменным.

Для ускорения преобразования генератор тактовых импульсов модуля АЦП может быть включен так, чтобы время T_{AD} вышло за пределы минимально-допустимого значения (см. соответствующие параметры в разделе "Электрические характеристики"). При этом модуль АЦП будет выдавать некорректный результат. Источником импульсов преобразования АЦП может быть один из трех источников: $2T_{OSC}$, $8T_{OSC}$, $32T_{OSC}$ (RC-генератор не может использоваться).

Время преобразования может быть вычислено по формуле:

Время преобразования = $T_{AD} + N \bullet T_{AD} + (10 - N)(2T_{OSC})$,
где N - разрядность АЦП.

Из-за того, что время T_{AD} зависит от частоты кварцевого генератора микроконтроллера, пользователь должен использовать какой-нибудь иной методов (например, используя таймер, программный цикл и др.) для определения времени изменения. В примере 22-5 представлено длительность 4 - разрядного преобразования, против 8 - разрядного. Например, для микроконтроллера с тактовой частотой 20МГц или 16МГц (тактовый сигнал АЦП $32T_{OSC}$). Через $5T_{AD}$, после старта преобразования, тактовый сигнал АЦП программно переключается на $2T_{OSC}$.

При $2T_{OSC}$ не выдержано минимальное требование к T_{AD}, поэтому 4 младших бита будут иметь неверное значение.

Пример 22-5 Расчет длительности 4 - разрядного преобразования

	Частота(1)	**Разрешение**	
		4 бита	**8 бит**
T_{AD}	20МГц	1.6 мкс	1.6 мкс
	16МГц	2.0 мкс	2.0 мкс
T_{OSC}	20МГц	50 нс	50 нс
	16МГц	62.5 нс	62.5 нс
$T_{AD} + N \bullet T_{AD} + (10 - N)(2T_{OSC})$	20МГц	8.6 мкс	17.6 мкс
	16МГц	10.75 мкс	22 мкс

Примечания:

1. Минимальное рекомендуемое значение T_{AD} = 1.6мкс (для микроконтроллеров PIC16C71 минимальное время T_{AD} = 2мкс).
2. Если требуется полное 8 - разрядное значение преобразования, источник тактовых импульсов для модуля АЦП не должен изменяться.

22.8 Работа модуля АЦП в SLEEP режиме микроконтроллера

Модуль АЦП может работать в SLEEP режиме микроконтроллера при условии, что источником импульсов преобразования АЦП будет внутренний RC генератор (ADCS1:ADCS0=11). При выборе RC генератора импульсов модуль АЦП сделает задержку в один машинный цикл перед началом преобразования. Это позволяет программе пользователя выполнить команду SLEEP, тем самым уменьшить “цифровой шум” во время преобразования. После завершения преобразования аппаратно сбрасывается бит GO/-DONE в '0', результат преобразования сохраняется в регистре ADRES. Если разрешено прерывание от АЦП, то микроконтроллер выйдет из режима SLEEP. Если же прерывание было запрещено, то после преобразования модуль АЦП будет выключен, хотя бит ADON останется установленным.

Если был выбран другой источник тактовых импульсов АЦП (не внутренний RC генератор), то выполнение программой инструкции SLEEP прервет процесс преобразования и выключит модуль АЦП, оставив установленным бит ADON. Выключение модуля АЦП уменьшит ток потребления микроконтроллера.

Примечание. Для работы модуля АЦП в SLEEP режиме необходимо выбрать внутренний RC генератор (ADCS1:ADCS0=11), инструкция SLEEP должна быть выполнена сразу после команды, устанавливающей бит GO/-DONE в '1'.

22.9 Точность преобразования АЦП

Абсолютная точность АЦП определяется суммарной ошибкой, исходя из ошибки дискретизации, интегральной ошибки, ошибки шкалы, ошибки смещения и монотонности. Суммарная ошибка определяется как максимальный разброс между текущим и идеальным результатом для любого значения. Абсолютная ошибка АЦП меньше ± 1 значащего бита при V_{DD}=V_{REF}, но она возрастает при отклонении V_{REF} от V_{DD}.

В некотором диапазоне напряжений на аналоговом входе цифровой результат будет один и тот же. Это возникает из-за дискретизации, которая неизбежна при преобразовании аналоговой величины в цифровую форму. Ошибка дискретизации составляет ± ½ значащего бита, и единственный способ уменьшить ее - увеличить разрядность АЦП.

Ошибку смещения составляет разность между результатом первого преобразования и идеальным значением. Эта ошибка сдвигает всю передаточную функцию, и может быть учтена при помощи калибровки. Ошибка вносится в результате наложения токов утечки и выходного сопротивления источника сигнала.

Ошибка усиления измеряется как максимальное отклонение результата, скорректированного с учетом ошибки смещения. Эта ошибка проявляется в виде изменения наклона передаточной функции. Ошибка усиления может быть откалибрована и учтена.

Ошибка линейности определяется как разница в приращении входного напряжения для получения одинакового приращения выходного кода и не поддается калибровке. Интегральная ошибка вычисляется как отклонение результата, скорректированного с учетом ошибки усиления.

Дифференциальная ошибка вычисляется как отклонение максимальной длины кода результат от идеальной длины кода без учета других ошибок.

Максимальный ток утечки вывода смотрите в разделе "Электрические характеристики" параметр D060.

В системах с низкой тактовой частотой предпочтительно использование встроенного RC генератора. В системах с высокой рабочей частотой следует использовать тактовый сигнал от основного генератора. Предпочтительно использовать АЦП с T_{AD} не больше 8 мкс, но не меньше рекомендованного нижнего предела. Использование тактового сигнала от основного генератора позволяет снизить влияние шумов от переключения внутренних вентилей, т.к. переключение логики АЦП происходит синхронно с другими устройствами, что невозможно при использовании встроенного RC генератора. Если каналы цифрового ввода/вывода постоянно активны, потеря точности из-за шумов при переключении может быть значительной.

В случае использования АЦП в SLEEP режиме, источником тактового сигнала должен быть встроенный RC генератор. В этом режиме отсутствуют цифровые шумы, т.к. другие узлы микроконтроллера остановлены, поэтому точность преобразования получается высокой.

22.10 Эффект сброса

При сбросе микроконтроллера значения всех его регистров устанавливаются по умолчанию. Сброс выключает модуль АЦП, а также останавливает процесс преобразования, если он был начат. Все выводы, используемые модулем АЦП, настраиваются как аналоговые входы. Регистр ADRES после сброса POR будет содержать неизвестное значение, а после остальных видов сброса не изменит своего значения.

22.11 Подключение к модулю АЦП

Если значение входного напряжения превышает на 0.2В величину порога питающих напряжений (V_{SS} и V_{DD}), то точность преобразования выйдет за пределы значений, оговоренных в спецификации.

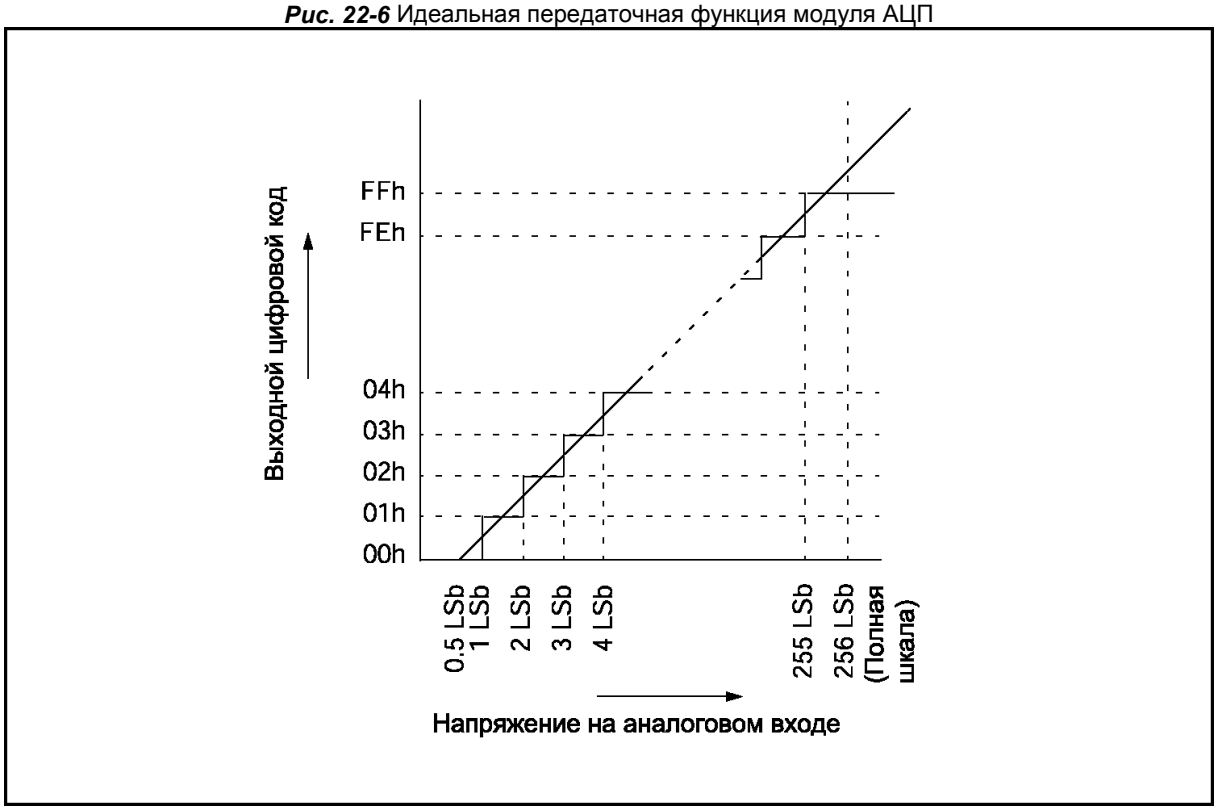
Примечание. Требуется некоторая осторожность при использовании вывода RA0 из-за его непосредственной близости к выводу OSC1.

Иногда, для сглаживания пульсаций входного сигнала, на вход АЦП добавляется внешняя RC цепочка. Значение сопротивления R должно выбираться так, чтобы общее сопротивление источника сигнала было в пределах рекомендованной величины 10кОм. Любой внешний электронный компонент, подключенный к аналоговому входу(конденсатор, стабилитрон и др.), должны иметь низкий ток утечки через вывод.

22.12 Передаточная функция модуля АЦП

Идеальная функция модуля АЦП работает по следующему правилу: первый бит значения измеряемой аналоговой величины будет установлен, если входное напряжение (V_{AIN}) на аналоговом входе будет равно 1 Lsb (V_{REF}/256) (см. рисунок 22-6).

Рис. 22-6 Идеальная передаточная функция модуля АЦП



22

22.13 Инициализация

В примере 22-4 показана инициализация модуля АЦП микроконтроллера PIC16C74A.

Пример 22-4 Инициализация модуля АЦП (для PIC16C74A)

```
        BSF     STATUS, RP0   ; Выбрать банк 1
        CLRF    ADCON1        ; Настроить входы АЦП
        BSF     PIE1, ADIE    ; Разрешить прерывания от АЦП
        BCF     STATUS, RP0   ; Выбрать банк 0
        MOVLW   0xC1          ; Тактовые импульсы от RC генератора АЦП,
        MOVWF   ADCON0        ; включить АЦП, выбрать канал 0
        BCF     PIR1, ADIF    ; Сбросить флаг прерываний от АЦП
        BSF     INTCON, PEIE  ; Разрешить периферийные прерывания
        BSF     INTCON, GIE   ; Разрешить прерывания в системе
;
; Выдержать паузу, необходимую для заряда внутреннего конденсатора CHOLD.
; Затем начинать преобразование АЦП.
;
        BSF     ADCON0, GO    ; Старт преобразования
        :                     ; Ожидать установку флага ADIF или сброс
        :                     ; бита GO/-DONE по завершению преобразования
```

22.14 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Я использую микроконтроллер PIC16C7XX в своем устройстве. Заметил, что результат преобразования АЦП не всегда точен. Что можно сделать?

Ответ 1:

1. Удостоверьтесь, что Вы выполняете требования синхронизации АЦП. Если Вы постоянно включаете/выключаете АЦП или изменяете входной канал, то необходимо обеспечить минимальную задержку перед началом преобразования. Значение T_{AD} (время получение одного бита) должно быть от 2мкс до 6мкс (устанавливается в регистре ADCON0). Если T_{AD} слишком мало, входное напряжение не может быть полностью преобразовано. Если T_{AD} достаточно большое, то напряжение на конденсаторе C_{HOLD} может измениться за счет внутренних токов утечки. Выбор параметров синхронизации описан в технической документации. С помощью представленных формул можно вычислить параметры синхронизации для конкретного случая.
2. Достаточно часто внутреннее сопротивление источника сигнала имеет большое значение (больше 10кОм), поэтому ток, заряжающий конденсатор C_{HOLD}, может влиять на точность преобразования. Если входной сигнал быстро не изменяется, то можно к аналоговому входу подключить конденсатор 0.1мкФ. Этот конденсатор зарядиться до напряжения аналогового сигнала и при выборки обеспечит требуемый кратковременный ток заряда внутреннего конденсатора C_{HOLD} (51.2пФ).
3. Из технической документации: " В системах с низкой тактовой частотой предпочтительно использование встроенного RC генератора. В системах с высокой рабочей частотой следует использовать тактовый сигнал от основного генератора. Предпочтительно использовать АЦП с T_{AD} не больше 8 мкс, но не меньше рекомендованного нижнего предела. Использование тактового сигнала от основного генератора позволяет снизить влияние шумов от переключения внутренних вентилей, т.к. переключение логики АЦП происходит синхронно с другими устройствами, что невозможно при использовании встроенного RC генератора. В случае использования АЦП в SLEEP режиме, источником тактового сигнала должен быть встроенный RC генератор. В этом режиме отсутствуют цифровые шумы, т.к. другие узлы микроконтроллера остановлены, поэтому точность преобразования получается высокой."

Вопрос 2: После старта преобразования АЦП могу я изменить входной канал для следующего измерения?

Ответ 2:

После установки в '1' бита GO (старт преобразования) конденсатор C_{HOLD} отключается от аналогового входа через T_{AD}. Как только конденсатор отсоединился от аналогового входа, может быть выбран другой канал.

Вопрос 3: Посоветуйте хорошую книгу по теории АЦП.

Ответ 3:

“Analog-Digital Conversion Handbook” 3-я редакция, издательство Prentice Hall (ISBN 0-13-03-2848-0).

22.15 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с основным модулем АЦП в микроконтроллерах PICmicro MCU:

Документ	Номер
Using the Analog to Digital Converter Применение АЦП	AN546
Four Channel Digital Voltmeter with Display and Keyboard Четырех канальный вольтметр с дисплеем и клавиатурой	AN557

Раздел 23. Модуль 10 - разрядного АЦП

Содержание

23.1 Введение

Модуль аналого-цифрового преобразования (АЦП) имеет до восьми входных каналов.

Входной аналоговый сигнал через коммутатор каналов заряжает внутренний конденсатор АЦП C_{HOLD}. Модуль АЦП преобразует напряжение, удерживаемое на конденсаторе C_{HOLD} в соответствующий 10-разрядный цифровой код методом последовательного приближения. Источник верхнего и нижнего опорного напряжения может быть программно выбран с выводов V_{DD}, V_{SS}, AN3/V_{REF+} или AN2/V_{REF-}.

Допускается работа модуля АЦП в SLEEP режиме микроконтроллера, при этом в качестве источника тактовых импульсов для АЦП должен быть выбран RC генератор.

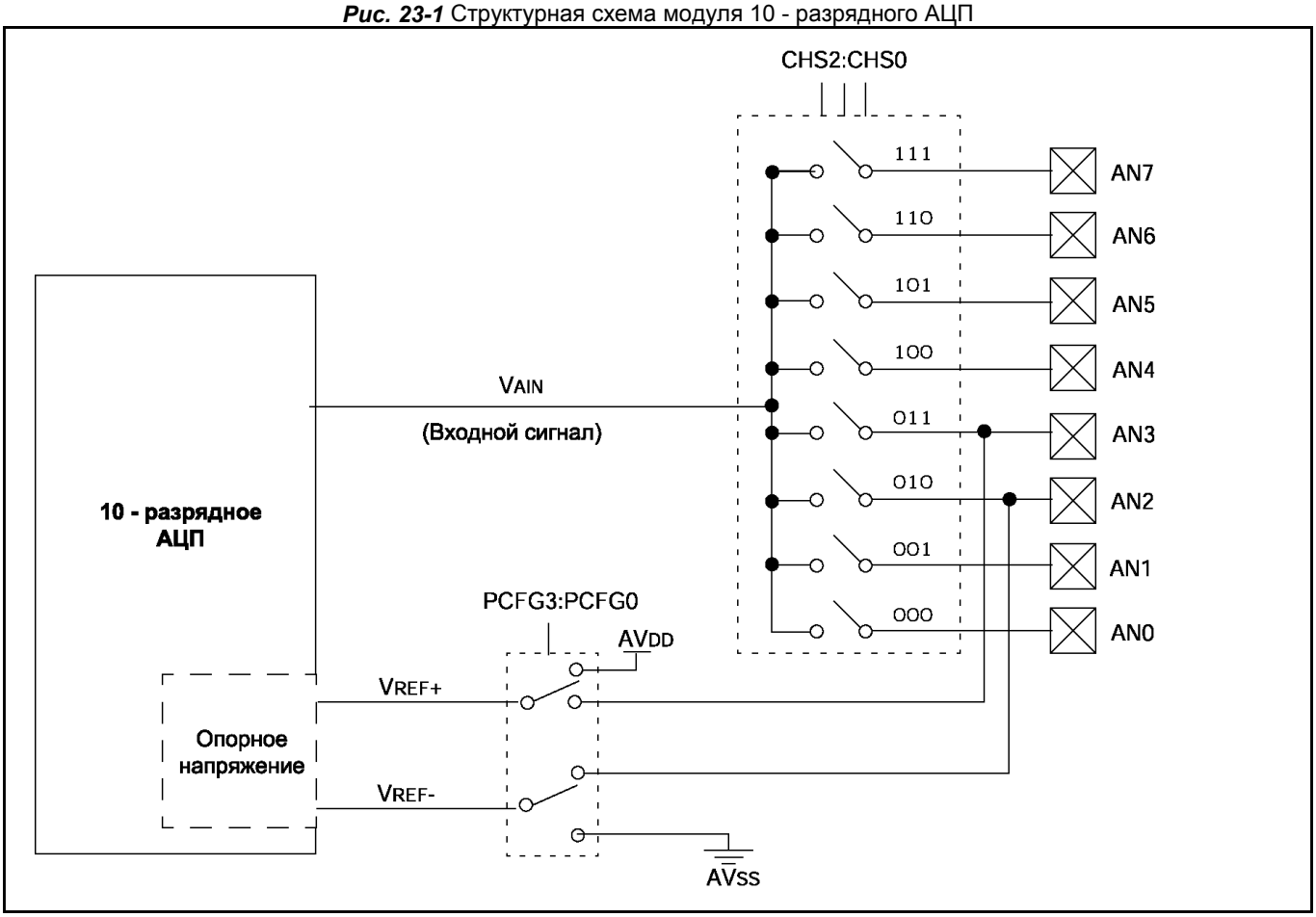
Для управления АЦП в микроконтроллере используется 4 регистра:

- Регистр результата ADRESH (старший байт);
- Регистр результата ADRESL (младший байт);
- Регистр управления ADCON0;
- Регистр управления ADCON1.

Регистр ADCON0 используется для настройки работы модуля АЦП, а с помощью регистра ADCON1 устанавливается, какие входы микроконтроллера будут использоваться модулем АЦП и в каком режиме (аналоговый вход или цифровой порт ввода/вывода).

Структурная схема модуля АЦП показана на рисунке 23-1.

Рис. 23-1 Структурная схема модуля 10 - разрядного АЦП



23.2 Управляющие регистры

Регистр ADCON0:

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	**ADCS0**	**CHS2**	**CHS1**	**CHS0**	**GO/-DONE**	**-**	**ADON**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как '0'
–n – значение после POR
–x – неизвестное значение после POR

биты 7-6: **ADCS1:ADCS0:** Выбор источника тактового сигнала
00 = $F_{OSC}/2$
01 = $F_{OSC}/8$
10 = $F_{OSC}/32$
11 = F_{RC} (внутренний RC генератор модуля АЦП)

биты 5-3: **CHS2:CHS0:** Выбор аналогового канала
000 = канал 0, (AN0)
001 = канал 1, (AN1)
010 = канал 2, (AN2)
011 = канал 3, (AN3)
100 = канал 4, (AN4)
101 = канал 5, (AN5)
110 = канал 6, (AN6)
111 = канал 7, (AN7)

Примечание. Для микроконтроллеров, в которых не реализованы все 8 каналов АЦП. Модуль АЦП аппаратно позволяет выполнять выборку нереализованных каналов.

бит 2: **GO/-DONE:** Бит статуса модуля АЦП
Если ADON=1
1 = модуль АЦП выполняет преобразование (установка бита вызывает начало преобразования)
0 = состояние ожидания (аппаратно сбрасывается по завершению преобразования)

бит 1: **Не используется:** читается как ‘0’

бит 0: **ADON:** Бит включения модуля АЦП
1 = модуль АЦП включен
0 = модуль АЦП выключен и не потребляет тока

Регистр ADCON1:

R/W-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	**-**	**-**	**-**	**PCFG3**	**PCFG2**	**PCFG1**	**PCFG0**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **ADFM:** Формат сохранения 10-разрядного результата (см. рисунок 23-6)
1 = правое выравнивание, 6 старших бит ADRESH читаются как ‘0’
0 = левое выравнивание, 6 младших бит ADRESL читаются как ‘0’

биты 6-4: **Не используются:** читаются как ‘0’

биты 3-0: **PCFG3:PCFG0:** Управляющие биты настройки каналов АЦП

PCFG3: PCFG0	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0	$V_{REF}+$	$V_{REF}-$	Кан./ V_{REF} (1)
0000	A	A	A	A	A	A	A	A	AV_{DD}	AV_{SS}	8/0
0001	A	A	A	A	$V_{REF}+$	A	A	A	AN3	AV_{SS}	7/1
0010	D	D	D	A	A	A	A	A	AV_{DD}	AV_{SS}	5/0
0011	D	D	D	A	$V_{REF}+$	A	A	A	AN3	AV_{SS}	4/1
0100	D	D	D	D	A	D	A	A	AV_{DD}	AV_{SS}	3/0
0101	D	D	D	D	$V_{REF}+$	D	A	A	AN3	AV_{SS}	2/1
011x	D	D	D	D	D	D	D	D	AV_{DD}	AV_{SS}	0/0
1000	A	A	A	A	$V_{REF}+$	$V_{REF}-$	A	A	AN3	AN2	6/2
1001	D	D	A	A	A	A	A	A	AV_{DD}	AV_{SS}	6/0
1010	D	D	A	A	$V_{REF}+$	A	A	A	AN3	AV_{SS}	5/1
1011	D	D	A	A	$V_{REF}+$	$V_{REF}-$	A	A	AN3	AN2	4/2
1100	D	D	D	A	$V_{REF}+$	$V_{REF}-$	A	A	AN3	AN2	3/2
1101	D	D	D	D	$V_{REF}+$	$V_{REF}-$	A	A	AN3	AN2	2/2
1110	D	D	D	D	D	D	D	A	AV_{DD}	AV_{SS}	1/0
1111	D	D	D	D	$V_{REF}+$	$V_{REF}-$	D	A	AN3	AN2	1/2

A = аналоговый вход D = цифровой канал ввода/вывода

Примечание 1. В этом столбце указывается число аналоговых каналов, доступных для выполнения преобразования, и число входов источника опорного напряжения.

Примечание. При сбросе микроконтроллера все выводы мультиплицированные с модулем АЦП (ANx) настраиваются как аналоговые входы.

23.3 Работа модуля АЦП

В регистрах ADRESH:ADRESL сохраняется 10 - разрядный результат аналого-цифрового преобразования. Когда преобразование завершено, результат преобразования записывается в регистры ADRESH:ADRESL, после чего сбрасывается бит GO/-DONE (ADCON0<2>) и устанавливается флаг прерывания ADIF.

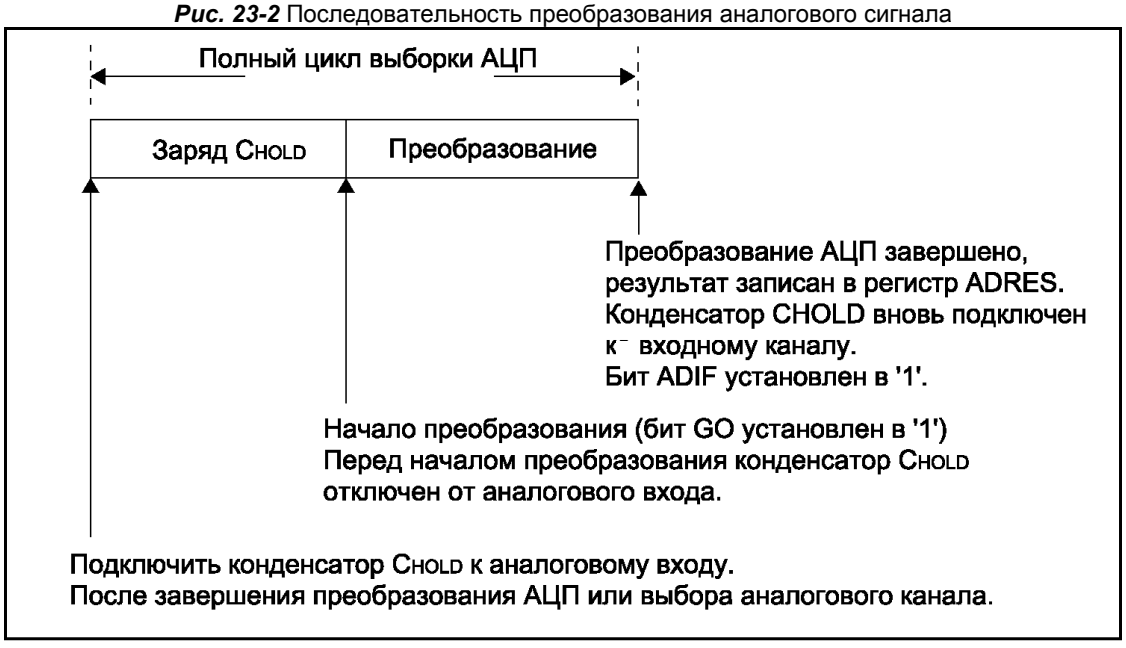
После включения и настройки АЦП необходимо выбрать рабочий аналоговый канал. Соответствующие биты TRIS аналоговых каналов должны настраивать канал порта ввода/вывода на вход. Перед началом преобразования необходимо выдержать временную паузу, расчет которой приведен в разделе 23.4.

Рекомендованная последовательность действий для работы с АЦП:

1. Настроить модуль АЦП:
   - Настроить выводы как аналоговые входы, входы V_{REF} или цифровые каналы ввода/вывода (ADCON1);
   - Выбрать входной канал АЦП (ADCON0);
   - Выбрать источник тактовых импульсов для АЦП (ADCON0);
   - Включить модуль АЦП (ADCON0).
2. Настроить прерывание от модуля АЦП (если необходимо):
   - Сбросить бит ADIF в '0';
   - Установить бит ADIE в '1';
   - Установить бит PEIE в '1';
   - Установить бит GIE в '1'.
3. Выдержать паузу, необходимую для зарядки конденсатора C_{HOLD}.
4. Начать аналого-цифровое преобразование:
   - Установить GO/-DONE бит в '1' (ADCON0).
5. Ожидать, окончания преобразования:
   - Ждать, пока бит GO/-DONE не будет сброшен в '0'; ИЛИ
   - Ожидать прерывание по окончанию преобразования.
6. Считать результат преобразования из регистров ADRESH:ADRESL, сбросить бит ADIF в '0', если это необходимо.
7. Для следующего преобразования необходимо выполнить шаги начиная с пункта 1 или 2. Время преобразования одного бита определяется как время T_{AD}. Минимальное время ожидания перед следующим преобразованием должно составлять $2T_{AD}$.

На рисунке 23-2 показана последовательность преобразования аналогового сигнала. Время заряда C_{HOLD} - интервал времени в течение которого на внутренний конденсатор АЦП подается внешний сигнал. Время преобразования равно 12 T_{AD}, отсчет начинается с момента установки в '1' бита GO. Сумма этих двух временных интервалов является длительностью полного цикла преобразования АЦП. Существует минимальный интервал времени, в течение которого внешний сигнал подается на внутренний конденсатор C_{HOLD}, чтобы гарантировать требуемую точность АЦП.

Рис. 23-2 Последовательность преобразования аналогового сигнала



23.4 Временные требования к подключению канала АЦП

Для обеспечения необходимой точности преобразования, конденсатор C_{HOLD} должен успевать полностью заряжаться до уровня входного напряжения. Схема аналогового входа АЦП показана на рисунке 23-3. Сопротивления R_S и R_{SS} непосредственно влияют на время зарядки конденсатора C_{HOLD}. Величина сопротивления ключа выборки (R_{SS}) зависит от напряжения питания V_{DD} (см. рисунок 23-3). **Максимальное рекомендуемое значение внутреннего сопротивления источника аналогового сигнала 10кОм.** При меньших значениях сопротивления источника сигнала меньше суммарное время преобразования.

После того, как будет выбран один из нескольких аналоговых входных каналов, но прежде, чем будет производиться преобразование, должно пройти определенное время. Для нахождения данного времени воспользуетесь уравнением 23-1. Это уравнение дает результат с ошибкой в ½ LSb (2048 шагов АЦП). Ошибка в ½ LSb, это максимальная погрешность, позволяющая функционировать модулю АЦП с необходимой точностью.

Уравнение 23-1 Вычисление временной задержки

T_{ACQ} = Время задержки усилителя + Время заряда конденсатора C_{HOLD} + Температурный коэффициент
$= T_{AMP} + T_C + T_{COFF}$

Уравнение 23-2 Минимальное время заряда конденсатора C_{HOLD}

$$V_{HOLD} = (V_{REF} - (V_{REF}/512)) \bullet (1 - e^{(-Tc/Chold(Ric + Rss + Rs))})$$

T_C = - 120пФ (1кОм + R_{SS} + R_S) Ln(1/2047)

В примере 23-1 показано вычисление минимального значения времени T_{ACQ}. Вычисления основываются на следующих исходных данных:

C_{HOLD}	= 120пФ
Rs	= 10кОм
Ошибка преобразования	≤ 1/2 Lsb
V_{DD}	= 5В → Rss = 7 кОм (см. график на рисунке 23-3)
Температура	= 50°C (максимально возможная)
V_{HOLD}	= 0В @ t = 0

Пример 23-1 Вычисление минимального значения времени T_{ACQ} (случай 1)

T_{ACQ} = T_{AMP} + T_C + T_{COFF}
Температурный коэффициент необходимо использовать только при рабочей температуре более 25°C.

T_{ACQ} = 2мкс + T_C + [(Температура - 25°C)(0.05мкс/°C)]

T_C = - C_{HOLD} (R_{IC} + R_{SS} + R_S) Ln(1/2047)
= - 120пФ (1кОм + 7кОм + 10кОм) Ln(0.0004885)
= - 120пФ (18кОм) Ln(0. 0004885)
= -2.16мкс (-7.6241)
= 16.47мкс

T_{ACQ} = 2мкс + 16.47мкс + [(50°C - 25°C)(0.05мкс/°C)]
= 18.47мкс + 1.25мкс
= 19.72мкс

Для более точного представления влияния внутреннего сопротивления источника сигнала на время заряда конденсатора C_{HOLD} рассмотрим пример 23-2, отличающийся от примера 23-1 тем, что внутреннее сопротивление источника сигнала равно 50Ом.

Пример 23-2 Вычисление минимального значения времени T_{ACQ} (случай 2)

T_{ACQ} = T_{AMP} + T_C + T_{COFF}
Температурный коэффициент необходимо использовать только при рабочей температуре более 25°C.

T_{ACQ} = 2мкс + T_C + [(Температура - 25°C)(0.05мкс/°C)]

T_C = - C_{HOLD} (R_{IC} + R_{SS} + R_S) Ln(1/2047)
= - 120пФ (1кОм + 7кОм + 50Ом) Ln(0.0004885)
= - 120пФ (8050Ом) Ln(0. 0004885)
= -0.966мкс (-7.6241)
= 7.36мкс

T_{ACQ} = 2мкс + 7.36мкс + [(50°C - 25°C)(0.05мкс/°C)]
= 9.36мкс + 1.25мкс
= 10.61мкс

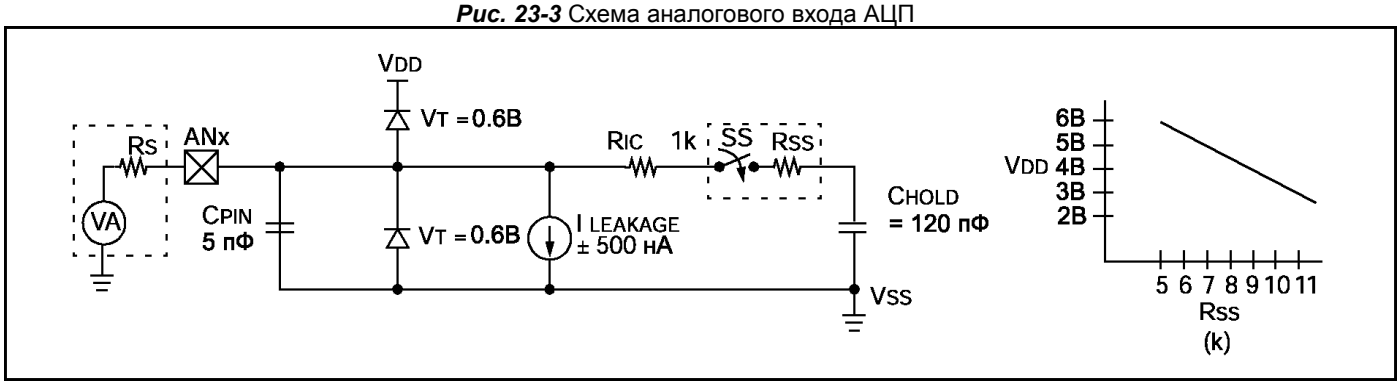
Примечание 1. Напряжение источника опорного напряжения V_{REF} не влияет на уравнение.

Примечание 2. Конденсатор C_{HOLD} после преобразования не разряжается.

Примечание 3. Максимальное рекомендуемое значение внутреннего сопротивления источника аналогового сигнала 10кОм. Это необходимо для компенсации внутреннего тока утечки.

Примечание 4. После того, как преобразование завершено, необходимо программно обеспечить задержку не менее 2.0T_{AD}, прежде чем начнете следующее преобразование. В течение этого времени конденсатор C_{HOLD} не подключен к выбранному входному каналу АЦП.

Рис. 23-3 Схема аналогового входа АЦП



Обозначения:

C_{PIN}	- входная емкость;
V_T	- пороговое напряжение;
$I_{LEAKAGE}$	- ток утечки вывода;
R_{IC}	- сопротивление соединения;
SS	- переключатель защелки;
C_{HOLD}	- конденсатор защелки.

23

23.5 Выбор источника тактовых импульсов для АЦП

Время получения одного бита результата равно T_{AD}. Для 10-разрядного результата требуется как минимум $11.5T_{AD}$. Параметры тактового сигнала для АЦП определяются программно, T_{AD} может принимать следующие значения:

- $2T_{OSC}$;
- $8T_{OSC}$;
- $32T_{OSC}$;
- Внутренний RC генератор модуля АЦП (2-6мкс).

Для получения корректного результата преобразования необходимо выбрать источник тактового сигнала АЦП, обеспечивающий время T_{AD} не менее 1.6 мкс (см. параметр 130 в разделе "Электрические характеристики").

В таблицах 23-1 и 23-2 указано максимальное значение тактовой частоты микроконтроллера для каждого режима синхронизирующего сигнала АЦП.

Таблица 23-1 Максимальное значение F_{OSC} удовлетворяющие требованию к T_{AD}
(для микроконтроллеров с нормальным диапазоном напряжения питания (C))

Источник импульсов АЦП (T_{AD})		**Рабочая частота F_{OSC}**			
Источник	**ADCS1:ADCS0**	**20МГц**	**5МГц**	**1.25МГц**	**333.33кГц**
$2T_{OSC}$	00	100нс(2)	400нс(2)	1.6 мкс	6мкс
$8T_{OSC}$	01	400нс(2)	1.6 мкс	6.4 мкс	24мкс(3)
$32T_{OSC}$	10	1.6 мкс	6.4 мкс	25.6 мкс(3)	96мкс(3)
RC-генератор	11	2-6мкс(1,4)	2-6мкс(1,4)	2-6мкс(1,4)	2-6мкс(1)

Обозначение: Затененные ячейки - не рекомендованное значение.
Примечания:

1. Типовое значение времени T_{AD} RC генератора АЦП равно 4мкс.
2. Это значение выходит за пределы минимально допустимого времени T_{AD}.
3. Для более точного преобразования рекомендуется выбрать другой источник тактовых импульсов.
4. Когда тактовая частота микроконтроллера больше 1МГц, рекомендуется использовать RC генератор АЦП только для работы в SLEEP режиме.

Таблица 23-2 Максимальное значение F_{OSC} удовлетворяющие требованию к T_{AD}
(для микроконтроллеров с расширенным диапазоном напряжения питания (LC))

Источник импульсов АЦП (T_{AD})		**Рабочая частота F_{OSC}**			
Источник	**ADCS1:ADCS0**	**4МГц**	**2МГц**	**1.25МГц**	**333.33кГц**
$2T_{OSC}$	00	500нс(2)	1.0мкс(2)	1.6 мкс	6мкс
$8T_{OSC}$	01	2.0мкс(2)	4.0 мкс	6.4 мкс	24мкс(3)
$32T_{OSC}$	10	8.0 мкс	16.0 мкс	25.6 мкс(3)	96мкс(3)
RC-генератор	11	3-9мкс(1,4)	3-9мкс(1,4)	3-9мкс(1,4)	3-9мкс(1)

Обозначение: Затененные ячейки - не рекомендованное значение.
Примечания:

1. Типовое значение времени T_{AD} RC генератора АЦП равно 6мкс.
2. Это значение выходит за пределы минимально допустимого времени T_{AD}.
3. Для более точного преобразования рекомендуется выбрать другой источник тактовых импульсов.
4. Когда тактовая частота микроконтроллера больше 1МГц, рекомендуется использовать RC генератор АЦП только для работы в SLEEP режиме.

23.6 Настройка аналоговых входов

Регистры ADCON1 и TRIS отвечают за настройку выводов АЦП. Если выводы микросхемы настраиваются как аналоговые входы, то при этом должны быть установлены соответствующие биты в регистре TRIS. Если соответствующий бит сброшен в '0', вывод микросхемы настроен как цифровой выход, со значениями выходных напряжений V_{OH} или V_{OL}.

Модуль АЦП функционирует независимо от состояния битов CHS2:CHS0 и битов TRIS.

Примечание 1. При чтении содержимого регистра порта нули будут установлены в тех разрядах, которые были настроены как аналоговые входы. Настроенные на цифровой вход каналы будут преобразовывать входные аналоговые уровни в цифровые, что однако не окажет влияния на точность преобразования.

Примечание 2. Значения напряжений, подаваемых на выводы, настроены как аналоговые входы, включая выводы (AN7:AN0), могут влиять на ток потребления входного буфера, который может выйти за пределы значений, оговоренных в технической спецификации.

23.7 Аналого-цифровое преобразование

В примере 23-3 показана последовательность действий для работы с АЦП. Выводы настроены как аналоговые входы. Источник опорного напряжения – AV_{DD}, AV_{SS}. Разрешены прерывания от модуля АЦП. Источником импульсов преобразования является RC генератор АЦП. Аналоговое цифровое преобразование выполняется с вывода AN0.

Примечание. Бит GO/-DONE и бит включения АЦП должны устанавливаться разными командами.

Сброс бита GO/-DONE в '0' во время преобразования приведет к его прекращению. При этом регистры результата (ADRESH:ADRESL) не изменят своего содержимого. После досрочного завершения преобразования необходимо обеспечить временную задержку $2T_{AD}$. Выдержав требуемую паузу, можно начать новое преобразования установкой бита GO/-DONE в '1'.

На рисунке 23-4 показана последовательность получения результата после установки бита GO/-DONE в '1'.

Пример 23-3 Выполнение преобразования АЦП

```
        BSF     STATUS, RP0   ; Выбрать банк 1
        CLRF    ADCON1        ; Настроить входы АЦП
        BSF     PIE1, ADIE    ; Разрешить прерывания от АЦП
        BCF     STATUS, RP0   ; Выбрать банк 0
        MOVLW   0xC1          ; Тактовые импульсы от RC генератора АЦП,
        MOVWF   ADCON0        ; включить АЦП, выбрать канал 0
        BCF     PIR1, ADIF    ; Сбросить флаг прерываний от АЦП
        BSF     INTCON, PEIE  ; Разрешить периферийные прерывания
        BSF     INTCON, GIE   ; Разрешить прерывания в системе
;
; Выдержать паузу, необходимую для заряда внутреннего конденсатора CHOLD.
; Затем начинать преобразование АЦП.
;
        BSF     ADCON0, GO    ; Старт преобразования
        :                     ; Ожидать установку флага ADIF или сброс
        :                     ; бита GO/-DONE по завершению преобразования
```

Рис. 23-4 Последовательность получения результата после установки бита GO/-DONE в '1'

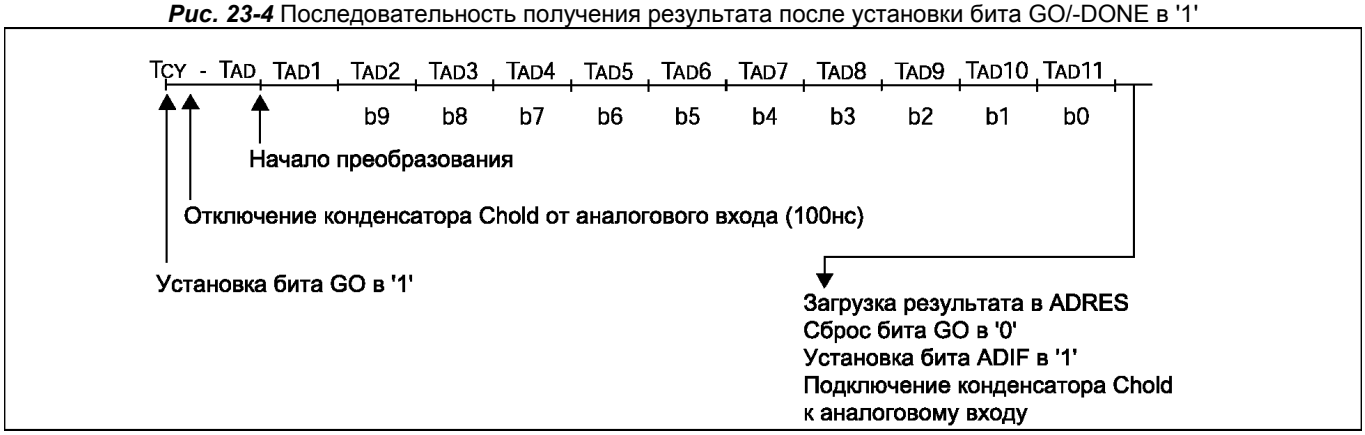
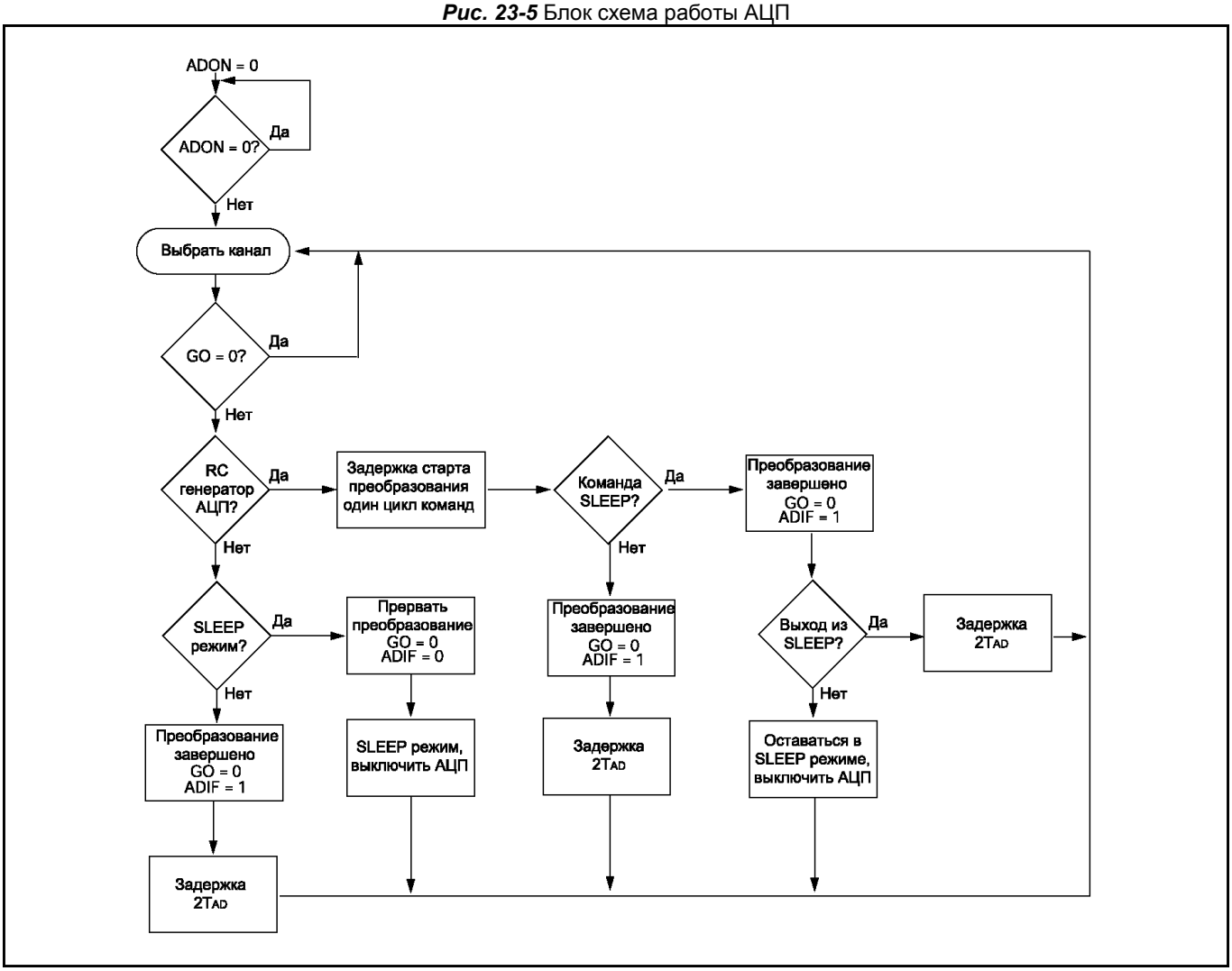


Рис. 23-5 Блок схема работы АЦП



23.7.1 Быстрое преобразование взамен разрешающей способности

Иногда бывает необходимо быстрое по времени преобразование за счет снижения разрешающей способности АЦП. Эта особенность работы модуля АЦП может использоваться в программе пользователя. Независимо от используемой разрешающей способности интервал времени T_{ACQ} остается неизменным.

Для ускорения преобразования генератор тактовых импульсов модуля АЦП может быть включен так, чтобы время T_{AD} вышло за пределы минимально-допустимого значения (см. соответствующие параметры в разделе "Электрические характеристики"). При этом модуль АЦП будет выдавать некорректный результат. Источником импульсов преобразования АЦП может быть один из трех источников: $2T_{OSC}$, $8T_{OSC}$, $32T_{OSC}$ (RC-генератор не может использоваться).

Время преобразования может быть вычислено по формуле:

Время преобразования = $2\ T_{AD} + N \bullet T_{AD} + (11 - N)(2T_{OSC})$,
где N - разрядность АЦП.

Из-за того, что время T_{AD} зависит от частоты кварцевого генератора микроконтроллера, пользователь должен использовать какой-нибудь иной методов (например, используя таймер, программный цикл и др.) для определения времени изменения. В примере 23-4 представлено длительность 4 - разрядного преобразования, против 10 - разрядного. Например, для микроконтроллера с тактовой частотой 20МГц (тактовый сигнал АЦП $32T_{OSC}$). Через $6T_{AD}$, после старта преобразования, тактовый сигнал АЦП программно переключается на $2T_{OSC}$.

При $2T_{OSC}$ не выдержано минимальное требование к T_{AD}, поэтому 6 младших битов будут иметь неверное значение.

Пример 23-4 Расчет длительности 4 - разрядного преобразования

	Частота(1)	**Разрешение**	
		4 бита	**10 бит**
T_{AD}	20МГц	1.6 мкс	1.6мкс
T_{OSC}	20МГц	50 нс	50 нс
$2\ T_{AD} + N \bullet T_{AD} + (11 - N)(2T_{OSC})$	20МГц	8.7 мкс	17.6 мкс

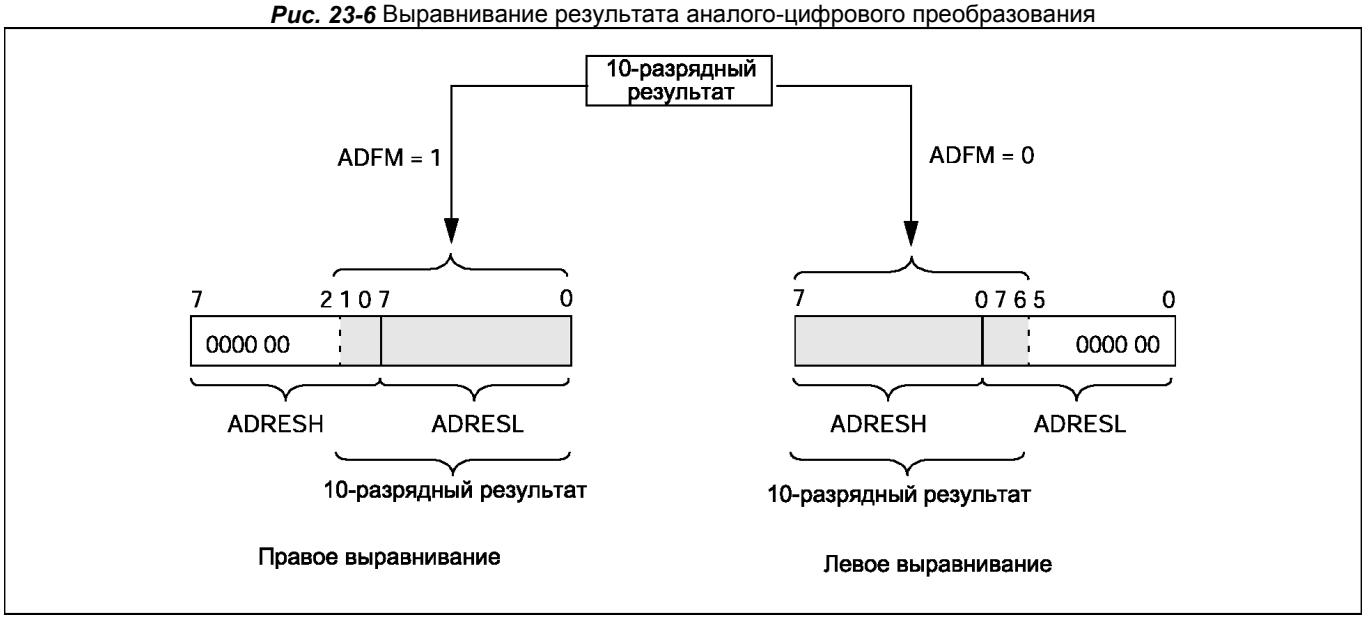
Примечания:

1. Минимальное рекомендуемое значение T_{AD} = 1.6мкс.
2. Если требуется полное 10 - разрядное значение преобразования, источник тактовых импульсов для модуля АЦП не должен изменяться.

23.7.2 Выравнивание результата преобразования

10-разрядный результат преобразования сохраняется в спаренном 16-разрядном регистре ADRESH:ADRESL. Запись результата преобразования может выполняться с правым или левым выравниванием, в зависимости от значения бита ADFM (см. рисунок 23-6). Не задействованные биты регистра ADRESH:ADRESL читаются как ‘0’. Если модуль АЦП выключен, то 8-разрядные регистры ADRESH и ADREL могут использоваться как регистры общего назначения.

Рис. 23-6 Выравнивание результата аналого-цифрового преобразования



23.8 Работа модуля АЦП в SLEEP режиме микроконтроллера

Модуль АЦП может работать в SLEEP режиме микроконтроллера при условии, что источником импульсов преобразования АЦП будет внутренний RC генератор (ADCS1:ADCS0=11). При выборе RC генератора импульсов модуль АЦП сделает задержку в один машинный цикл перед началом преобразования. Это позволяет программе пользователя выполнить команду SLEEP, тем самым уменьшить “цифровой шум” во время преобразования. После завершения преобразования аппаратно сбрасывается бит GO/-DONE в '0', результат преобразования сохраняется в регистрах ADRESH:ADRESL. Если разрешено прерывание от АЦП, то микроконтроллер выйдет из режима SLEEP. Если же прерывание было запрещено, то после преобразования модуль АЦП будет выключен, хотя бит ADON останется установленным.

Если был выбран другой источник тактовых импульсов АЦП (не внутренний RC генератор), то выполнение программой инструкции SLEEP прервет процесс преобразования и выключит модуль АЦП, оставив установленным бит ADON. Выключение модуля АЦП уменьшит ток потребления микроконтроллера.

Примечание. Для работы модуля АЦП в SLEEP режиме необходимо выбрать внутренний RC генератор (ADCS1:ADCS0=11), инструкция SLEEP должна быть выполнена сразу после команды, устанавливающей бит GO/-DONE в '1'.

23.9 Эффект сброса

При сбросе микроконтроллера значения всех его регистров устанавливаются по умолчанию. Сброс выключает модуль АЦП, а также останавливает процесс преобразования, если он был начат. Все выводы, используемые модулем АЦП, настраиваются как аналоговые входы. Регистры ADRESH:ADRESL после сброса POR будут содержать неизвестное значение, а после остальных видов сброса не изменят своего значения.

23.10 Точность преобразования АЦП

Абсолютная точность АЦП определяется суммарной ошибкой, исходя из ошибки дискретизации, интегральной ошибки, ошибки шкалы, ошибки смещения и монотонности. Суммарная ошибка определяется как максимальный разброс между текущим и идеальным результатом для любого значения. Абсолютная ошибка АЦП меньше ± 1 значащего бита при V_{DD}=V_{REF}, но она возрастает при отклонении V_{REF} от V_{DD}.

В некотором диапазоне напряжений на аналоговом входе цифровой результат будет один и тот же. Это возникает из-за дискретизации, которая неизбежна при преобразовании аналоговой величины в цифровую форму. Ошибка дискретизации составляет ± ½ значащего бита, и единственный способ уменьшить ее - увеличить разрядность АЦП.

Ошибку смещения составляет разность между результатом первого преобразования и идеальным значением. Эта ошибка сдвигает всю передаточную функцию, и может быть учтена при помощи калибровки. Ошибка вносится в результате наложения токов утечки и выходного сопротивления источника сигнала.

Ошибка усиления измеряется как максимальное отклонение результата, скорректированного с учетом ошибки смещения. Эта ошибка проявляется в виде изменения наклона передаточной функции. Ошибка усиления может быть откалибрована и учтена.

Ошибка линейности определяется как разница в приращении входного напряжения для получения одинакового приращения выходного кода и не поддается калибровке. Интегральная ошибка вычисляется как отклонение результата, скорректированного с учетом ошибки усиления.

Дифференциальная ошибка вычисляется как отклонение максимальной длины кода результата от идеальной длины кода без учета других ошибок.

Максимальный ток утечки вывода смотрите в разделе "Электрические характеристики" параметр D060.

В системах с низкой тактовой частотой предпочтительно использование встроенного RC генератора. В системах с высокой рабочей частотой следует использовать тактовый сигнал от основного генератора. Предпочтительно использовать АЦП с T_{AD} не больше 8 мкс, но не меньше рекомендованного нижнего предела. Использование тактового сигнала от основного генератора позволяет снизить влияние шумов от переключения внутренних вентилей, т.к. переключение логики АЦП происходит синхронно с другими устройствами, что невозможно при использовании встроенного RC генератора. Если каналы цифрового ввода/вывода постоянно активны, потеря точности из-за шумов при переключении может быть значительной.

В случае использования АЦП в SLEEP режиме, источником тактового сигнала должен быть встроенный RC генератор. В этом режиме отсутствуют цифровые шумы, т.к. другие узлы микроконтроллера остановлены, поэтому точность преобразования получается высокой.

23.11 Использование CCP триггера

Аналого-цифровое преобразование может быть запущено при помощи “триггера специального события” модуля CCP. Для этого необходимо, чтобы биты CCPxM3:CCPxM0 (CCPxCON<3:0>) были запрограммированы как 1011 и был включен модуль АЦП (бит ADON должен быть установлен в '1'). При срабатывании триггера бит GO/-DONE будет установлен в '1', тем самым, запуская преобразование, а содержимое таймера TMR1 будет обнулено. Таймер сбрасывается и автоматически повторяет запуск преобразования через определенные промежутки времени. Пользователю необходимо будет только вовремя считывать содержимое регистров ADRESH:ADRESL. До начала преобразования необходимо выбрать соответствующий аналоговый канал, прежде чем “триггер специального события“ вызовет установку бита GO/-DONE.

При выключенном модуле АЦП (бит ADON сброшен в '0') сигнал “триггера специального события” игнорируется, но таймер TMR1 продолжает работать и сбрасываться.

23.12 Подключение к модулю АЦП

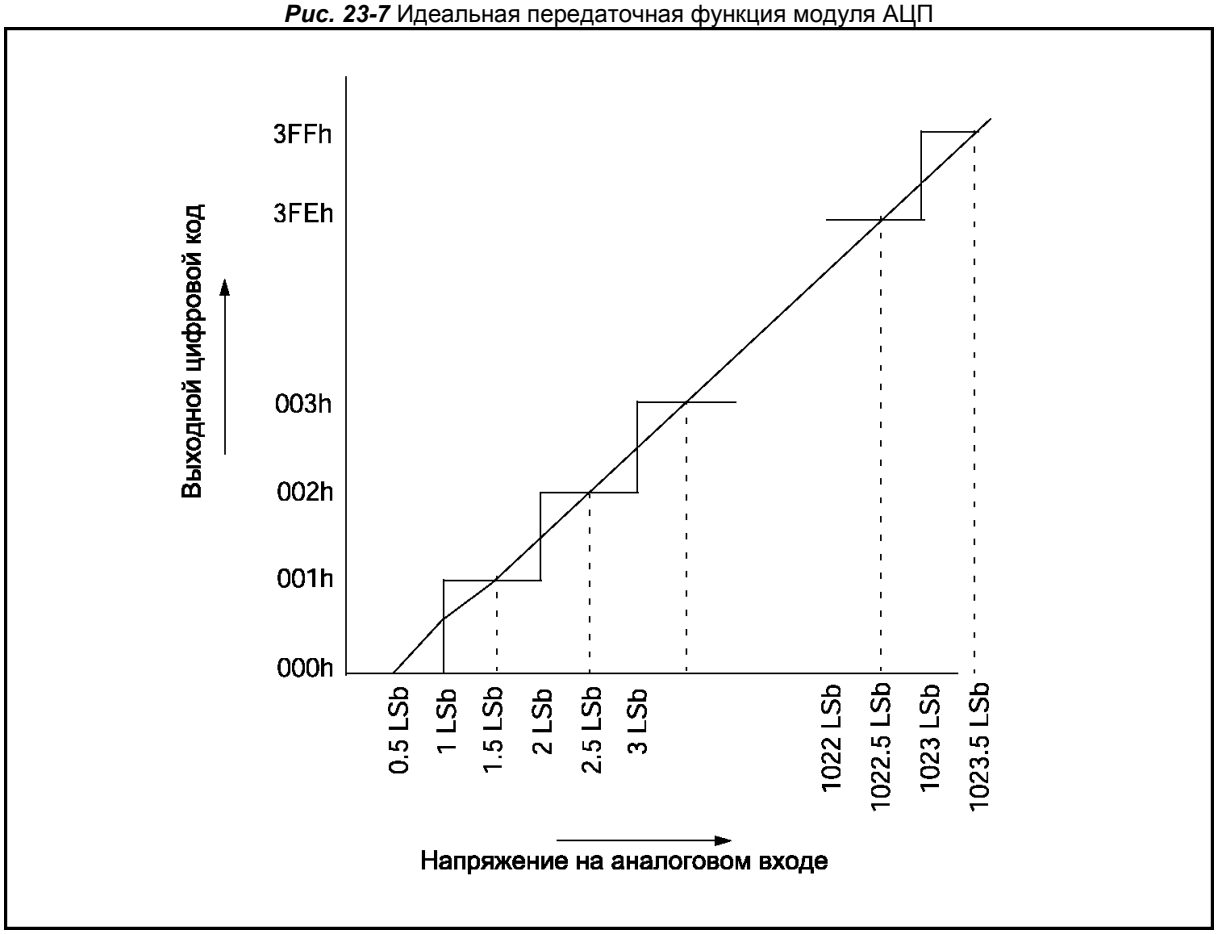
Если значение входного напряжения превышает на 0.3В величину порога питающих напряжений (V_{SS} и V_{DD}), то точность преобразования выйдет за пределы значений, оговоренных в спецификации.

Иногда, для сглаживания пульсаций входного сигнала, на вход АЦП добавляется внешняя RC цепочка. Значение сопротивления R должно выбираться так, чтобы общее сопротивление источника сигнала было в пределах рекомендованной величины 10кОм. Любой внешний электронный компонент, подключенный к аналоговому входу(конденсатор, стабилитрон и др.), должны иметь низкий ток утечки через вывод.

23.13 Передаточная функция модуля АЦП

Идеальная функция модуля АЦП работает по следующему правилу: первый бит значения измеряемой аналоговой величины будет установлен, если входное напряжение (V_{AIN}) на аналоговом входе будет равно 1 Lsb (V_{REF}/1024) (см. рисунок 23-7).

Рис. 23-7 Идеальная передаточная функция модуля АЦП



23.14 Инициализация

В примере 23-5 показана инициализация модуля АЦП.

Пример 23-5 Инициализация модуля АЦП

```
            BSF         STATUS, RP0   ; Выбрать банк 1
            CLRF        ADCON1        ; Настроить входы АЦП
            BSF         PIE1, ADIE    ; Разрешить прерывания от АЦП
            BCF         STATUS, RP0   ; Выбрать банк 0
            MOVLW       0xC1          ; Тактовые импульсы от RC генератора АЦП,
            MOVWF       ADCON0        ; включить АЦП, выбрать канал 0
            BCF         PIR1, ADIF    ; Сбросить флаг прерываний от АЦП
            BSF         INTCON, PEIE  ; Разрешить периферийные прерывания
            BSF         INTCON, GIE   ; Разрешить прерывания в системе
;
; Выдержать паузу, необходимую для заряда внутреннего конденсатора CHOLD.
; Затем начинать преобразование АЦП.
;
            BSF         ADCON0, GO    ; Старт преобразования
            :             ; Ожидать установку флага ADIF или сброс
            :             ; бита GO/-DONE по завершению преобразования
```

23.15 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Заметил, что результат преобразования АЦП не всегда точен. Что можно сделать?
Ответ 1:

1. Удостоверьтесь, что Вы выполняете требования синхронизации АЦП. Если Вы постоянно включаете/выключаете АЦП или изменяете входной канал, то необходимо обеспечить минимальную задержку перед началом преобразования. Значение T_{AD} (время получение одного бита) должно быть от 2мкс до 6мкс (устанавливается в регистре ADCON0). Если T_{AD} слишком мало, входное напряжение не может быть полностью преобразовано. Если T_{AD} достаточно большое, то напряжение на конденсаторе C_{HOLD} может измениться за счет внутренних токов утечки. Выбор параметров синхронизации описан в технической документации. С помощью представленных формул можно вычислить параметры синхронизации для конкретного случая.
2. Достаточно часто внутреннее сопротивление источника сигнала имеет большое значение (больше 10кОм), поэтому ток, заряжающий конденсатор C_{HOLD}, может влиять на точность преобразования. Если входной сигнал быстро не изменяется, то можно к аналоговому входу подключить конденсатор 0.1мкФ. Этот конденсатор зарядиться до напряжения аналогового сигнала и при выборки обеспечит требуемый кратковременный ток заряда внутреннего конденсатора C_{HOLD} (51.2пФ).
3. Из технической документации: " В системах с низкой тактовой частотой предпочтительно использование встроенного RC генератора. В системах с высокой рабочей частотой следует использовать тактовый сигнал от основного генератора. Предпочтительно использовать АЦП с T_{AD} не больше 8 мкс, но не меньше рекомендованного нижнего предела. Использование тактового сигнала от основного генератора позволяет снизить влияние шумов от переключения внутренних вентилей, т.к. переключение логики АЦП происходит синхронно с другими устройствами, что невозможно при использовании встроенного RC генератора. В случае использования АЦП в SLEEP режиме, источником тактового сигнала должен быть встроенный RC генератор. В этом режиме отсутствуют цифровые шумы, т.к. другие узлы микроконтроллера остановлены, поэтому точность преобразования получается высокой."

Вопрос 2: После старта преобразования АЦП могу я изменить входной канал для следующего измерения?
Ответ 2:
После установки в '1' бита GO (старт преобразования) конденсатор C_{HOLD} отключается от аналогового входа через T_{AD}. Как только конденсатор отсоединился от аналогового входа, может быть выбран другой канал.

Вопрос 3: Посоветуйте хорошую книгу по теории АЦП.
Ответ 3:
“Analog-Digital Conversion Handbook” 3-я редакция, издательство Prentice Hall (ISBN 0-13-03-2848-0).

23.16 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с модулем АЦП в микроконтроллерах PICmicro MCU:

Документ	Номер
Using the Analog to Digital Converter Применение АЦП	AN546
Four Channel Digital Voltmeter with Display and Keyboard Четырех канальный вольтметр с дисплеем и клавиатурой	AN557

Раздел 24. Модуль интегрирующего АЦП

Содержание

24.1 Введение

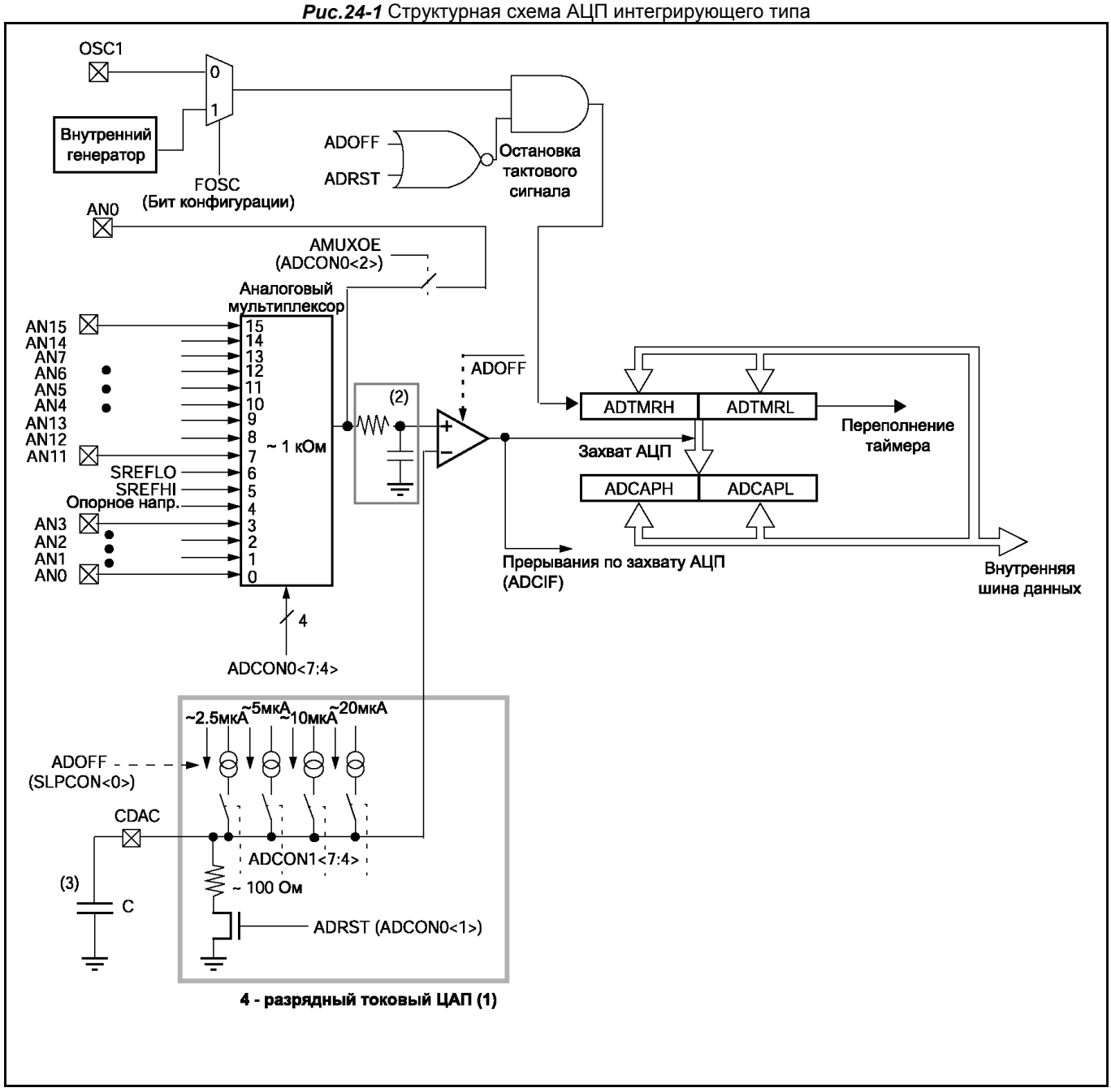
Для реализации модуля АЦП микроконтроллер содержит следующие компоненты:

- Прецизионный компаратор;
- 4 - разрядный программируемый источник тока;
- 16-ти канальный аналоговый мультиплексор;
- 16 - разрядный таймер с регистром захвата.

В этом разделе будет описано использования этих компонентов для реализации АЦП интегрирующего типа.

Каждый аналоговый канал мультиплицируется на единственный аналоговый вход, с которого выполняется преобразование сигнала в цифровую форму интегрирующим методом (используется один прецизионный компаратор). Программируемый источник тока заряжает внешний конденсатор, чтобы формировать линейно изменяющееся напряжение, используемое при преобразовании.

Рис.24-1 Структурная схема АЦП интегрирующего типа



Примечания:

1. Все источники тока выключены, если ADRST=1.
2. Постоянная времени приблизительно 3.5мкс.
3. Зависит от разрядности АЦП и диапазона напряжений входного сигнала.

24.2 Управляющие регистры

Модуль АЦП содержат два управляющих регистра ADCON0 и ADCON1, доступных для записи и чтения.

Регистр ADCON0:

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-1	R/W-0
ADCS3	**ADCS2**	**ADCS1**	**ADCS0**	**-**	**AMUXOE**	**ADRST**	**ADZERO**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как '0'
–n – значение после POR
–x – неизвестное значение после POR

биты 7-4: **ADCS3:ADCS0:** Выбор аналогового канала
0000 = канал 0, (AN0)
0001 = канал 1, (AN1)
0010 = канал 2, (AN2)
0011 = канал 3, (AN3)
0100 = вход опорного напряжения
0101 = опорное напряжение SREFHI
0110 = программируемое опорное напряжение SREFLO
0111 = канал 11, (AN11)
1000 = канал 12, (AN12)
1001 = канал 13, (AN13)
1010 = канал 4, (AN4)
1011 = канал 5, (AN5)
1100 = канал 6, (AN6)
1101 = канал 7, (AN7)
1110 = канал 14, (AN14)
1111 = канал 15, (AN15)

Примечание. Для микроконтроллеров, в которых не реализованы все 16 каналов АЦП. Модуль АЦП аппаратно позволяет выполнять выборку нереализованных каналов.

бит 3: **Не используется:** читается как ‘0’

бит 2: **AMUXOE:** Подключение выхода мультиплексора к выводу AN0
1 = выход мультиплексора подключен к выводу AN0 (отменяет действие бита TRIS)
0 = вывод AN0 работает в обычном режиме

бит 1: **ADRST:** Бит сброса модуля АЦП
1 = остановить таймер АЦП, разрядить конденсатор токового ЦАП
0 = нормальный режим (выполняется преобразование)

бит 0: **ADZERO:** Управление выбором нуля АЦП
1 = разрешено обнуление по AN1 и AN5
0 = нормальный режим, выборка с каналов AN1 и AN5

Регистр ADCON1:

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADDAC3	**ADDAC2**	**ADDAC1**	**ADDAC0**	**PCFG3**	**PCFG2**	**PCFG1**	**PCFG0**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как '0'
–n – значение после POR
–x – неизвестное значение после POR

биты 7-4: **ADDAC3:ADDAC0:** Биты управления источниками тока
0000 = все источники тока выключены
0001 = 2.25 мкА
0010 = 4.5 мкА
0011 = 6.75 мкА
0100 = 9 мкА
0101 = 11.25 мкА
0110 = 13.5 мкА
0111 = 15.75 мкА
1000 = 18 мкА
1001 = 20.25 мкА
1010 = 22.5 мкА
1011 = 24.75 мкА
1100 = 27 мкА
1101 = 29.25 мкА
1110 = 31.5 мкА
1111 = 33.75 мкА

биты 3-0: **PCFG3:PCFG0:** Биты настройки каналов АЦП

PCFG3:PCFG2	AN4	AN5	AN6	AN7
00	A	A	A	A
01	A	A	A	D
10	A	A	D	D
11	D	D	D	D

PCFG1:PCFG0	AN0	AN1	AN2	AN3
00	A	A	A	A
01	A	A	A	D
10	A	A	D	D
11	D	D	D	D

A = аналоговый вход D = цифровой канал ввода/вывода

Примечание. При сбросе микроконтроллера все выводы мультиплицированные с модулем АЦП (ANx) настраиваются как аналоговые входы.

Регистр SLPCON:

R/W-0	U-0	R/W-1	R/W-0	R/W-1	R/W-0	R/W-0	R/W-1
Резерв	**-**	**REFOFF**	**Резерв**	**OSCOFF**	**Резерв**	**Резерв**	**ADOFF**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как '0'
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **Резерв:** Всегда поддерживайте сброшенным в '0'

бит 6: **Не используется:** читается как ‘0’

бит 5: **REFOFF:** Бит включения источника опорного напряжения АЦП
1 = источник опорного напряжения выключен и не потребляет тока
0 = источник опорного напряжения включен

бит 4: **Резерв:** Всегда поддерживайте сброшенным в '0'

бит 3: **OSCOFF:** Режим работы тактового генератора АЦП
1 = тактовый генератор АЦП в SLEEP режиме микроконтроллера выключен и не потребляет тока
0 = тактовый генератор АЦП в SLEEP режиме микроконтроллера включен

бит 2: **Резерв:** Всегда поддерживайте сброшенным в '0'

бит 1: **Резерв:** Всегда поддерживайте сброшенным в '0'

бит 0: **ADOFF:** Включение АЦП
1 = АЦП выключено и не потребляет тока
0 = АЦП включено

24.3 Работа модуля АЦП

Существуют два метода выполнения преобразования. Чтобы определить окончание преобразования в первом методе ожидается прерывание по переполнению таймера ADTMR (бит OVFIF). Во втором методе ожидается прерывание по захвату данных АЦП (бит ADCIF). В обоих методах проверяется состояние каждого бита, чтобы проверить условие завершения преобразования.

В методе 1 используется установленное время преобразования, это означает, что конденсатор всегда заряжается до напряжения полной шкалы. Немедленно, после переполнения ADTMR, рекомендуется установить в '1' бит ADRST, чтобы разрядить внешний конденсатор. Это будет гарантировать, что остаточное напряжение на диэлектрике конденсатора не зависит от входного напряжения или предыдущего преобразования.

Во 2 методе используется переменное время преобразования, время преобразования меньше для низкого входного напряжения.

Последовательность шагов преобразования по методу 1 (установленное время преобразования):

1. Инициализировать модуль АЦП:
   a) Сбросить в '0' бит REFOFF (SLPCON<5>);
   b) Сбросить в '0' бит ADOFF (SLPCON<0>);
   c) Инициализировать биты ADCON1<7:4> для настройки программируемого источника тока.
2. Установить в '1' бит ADRST (ADCON0<1>), пока напряжение на внешнем конденсаторе не достигнет 0В. Рекомендуемое время 200мкс.
3. Выбрать входной канал.
4. Сбросить в '0' биты OVFIF, ADCIF.
5. Инициализировать таймер АЦП ADTMR. Значение ADTMR зависит от требуемой разрешающей способности (см. таблицу 24-1).
6. Для старта преобразование сбросьте бит ADRST в '0'. Это приведет к началу заряда внешнего конденсатора и инкременту ADTMR.
7. Преобразование завершено, когда произошло переполнение таймера АЦП (ADTMR) от FFFFh к 0000h. В момент переполнения устанавливается в '1' бит OVFIF.
8. Проверьте состояние бита ADCIF. Если бит ADCIF=1, то произошел захват данных, результат преобразования записан в регистре ADCAP. Этот метод зависит от минимального времени ожидания, затем проверяется захват данных АЦП после переполнения ADTMR. Если бит ADCIF=0, то входное напряжение было вне диапазона АЦП.
9. Сбросить в '0' бит ADRST (ADCON<1>), чтобы остановить ADTMR и разрядить внешний конденсатор.
10. Вычислить результат преобразования.
11. Перейти на шаг 2.

Последовательность шагов преобразования по методу 2 (переменное время преобразования):

1. Инициализировать модуль АЦП:
   a) Сбросить в '0' бит REFOFF (SLPCON<5>);
   b) Сбросить в '0' бит ADOFF (SLPCON<0>);
   c) Инициализировать биты ADCON1<7:4> для настройки программируемого источника тока.
2. Установить в '1' бит ADRST (ADCON0<1>), пока напряжение на внешнем конденсаторе не достигнет 0В. Рекомендуемое время 200мкс.
3. Выбрать входной канал.
4. Сбросить в '0' биты OVFIF, ADCIF.
5. Инициализировать таймер АЦП ADTMR. Значение ADTMR зависит от требуемой разрешающей способности (см. таблицу 24-1).
6. Для старта преобразование сбросьте бит ADRST в '0'. Это приведет к началу заряда внешнего конденсатора и инкременту ADTMR.
7. Преобразование завершено, когда напряжение на внешнем конденсаторе превышает входное напряжение, сигнал на выходе компаратора переходит с высокого уровня в низкий, что устанавливает бит ADCIF в '1'.
8. Проверьте состояние бита OVFIF (переполнение ADTMR). Если бит OVFIF=1, то произошло переполнение ADTMR (больше, чем допустимая разрешающая способность), входное напряжение было вне диапазона АЦП.
9. Сбросить в '0' бит ADRST (ADCON<1>), чтобы остановить ADTMR и разрядить внешний конденсатор.
10. Вычислить результат преобразования.
11. Перейти на шаг 2.

Примечание. ADTMR продолжает инкрементироваться, после захвата данных АЦП.

Максимальное значение таймера АЦП - 65 536. Источником тактового сигнала может быть внутренний или внешний генератор. При частоте тактового генератора 4Мгц максимальное время преобразования 16.38 мс (до переполнения таймера). Обычно преобразование должно быть завершено до переполнения таймера. Флаг переполнения таймера устанавливается в '1' при переходе от FFFFh к 0000h, если разрешено, генерируется прерывание.

Калибровка АЦП значительно упрощается за сет сохранения в EPROM памяти микроконтроллера калибровочных констант. Значение калибровочных констант определяется при заключительном заводском испытании микроконтроллера, они сохраняются в памяти для использования программой пользователя.

Для компенсации дрейфа компонентов АЦП необходимо периодически выполнять тестовые преобразования опорных напряжений (см. раздел 24.4 "Другие аналоговые модули"). Значения опорного напряжения приравнивают к калибровочному значению, сохраненному в EPROM. Поскольку все измерения выполняются относительно опорного напряжения, влияние напряжения смещения компаратора минимизировано. Модуль АЦП не требует точного источника тактового сигнала, основным требованием является - стабильность частоты генератора.

Дополнительную информацию смотрите в документации AN624 "PIC14000 Slope A/D Theory and Implementation".

24.3.1 Модуль таймера АЦП (ADTMR)

В состав модуля АЦП входит 16 - разрядный таймер ADTMR (ADTMRH:ADTMRL), приращение таймера происходит на каждом такте генератора. Регистры таймера ADTMR сбрасываются только при сбросе микроконтроллера по включению питания (POR), в остальных случаях необходимо инициализировать регистры таймера перед каждым преобразованием. 16 - разрядный регистр захвата данных (ADCAPH:ADCAPL) используется, чтобы фиксировать значение таймера ADTMR при захвате данных АЦП. Регистры захвата данных и регистры таймера доступны для записи и чтения.

Примечание 1. Чтение и запись регистров таймера ADTMR во время преобразования АЦП может привести к непредсказуемым последствиям.

Примечание 2. Правильная последовательность записи в регистры ADTMR - старший байт вслед за младшим. Обратная последовательность может быть причиной отсутствия запуска таймера.

Во время преобразования могут произойти следующие события:

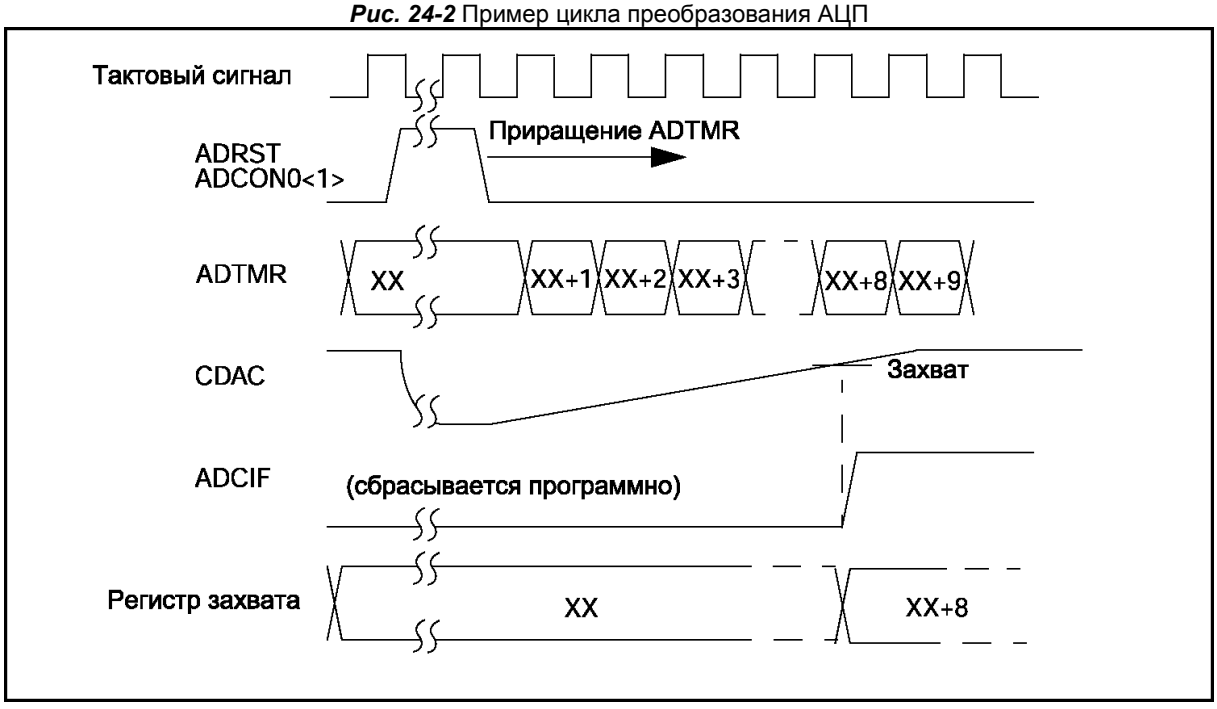
- Захват данных АЦП;
- Переполнение таймера.

В случае захвата данных происходит срабатывание компаратора (переход выходного сигнала с высокого уровня в низкий), напряжение на выводе CDAC превышает входное напряжение с выбранного канала АЦП. В момент переключения компаратора значение таймера переписывается в регистры захвата и устанавливается в '1' бит ADCIF.

Прерывание будет сгенерировано, если бит ADCIE=1 и бит глобального разрешения прерывания GIE=1. Бит ADCIF сбрасывается программно, до следующего цикла преобразования. Прерывание может произойти только один раз, во время преобразования.

В случае переполнения таймера (переход от FFFFh к 0000h) устанавливается в '1' флаг OVFIF. После переполнения таймер продолжает инкрементироваться. Прерывание будет сгенерировано, если бит OVFIE=1 и бит глобального разрешения прерывания GIE=1. Бит OVFIF сбрасывается программно, до следующего цикла преобразования.

Рис. 24-2 Пример цикла преобразования АЦП



24.3.2 Работа в SLEEP режиме микроконтроллера

Модуль АЦП может работать, когда микроконтроллер находится в SLEEP режиме при условии наличия тактового сигнала для АЦП. Для работы модуля АЦП в SLEEP режиме необходимо сбросить в '0' бит OSCOFF до перевода микроконтроллера в SLEEP режим. Биты REFOFF и ADOFF тоже должны быть сброшены в '0' для получения корректного результата преобразования. В SLEEP режиме микроконтроллера результат преобразования АЦП получается наиболее точным. Это связано с уменьшением внутреннего шума микроконтроллера.

Когда тактовый генератор микроконтроллера выключен, таймер ADTMR не инкрементируется. Даже если модуль АЦП включен, он не может вывести микроконтроллер из SLEEP режима, поскольку бит ADCIF не может быть установлен в '1'. Если на момент выхода из режима SLEEP компаратор переключился, то произойдет захват данных и генерация прерывания. Данные в регистре ADCAP не имеют значения.

Для снижения энергопотребления все аналоговые компоненты модуля АЦП должны быть выключены.

24.3.3 Эффект сброса

После любого сброса микроконтроллера модуль АЦП выключен (минимальное энергопотребление). Все выводы, мультиплицированные с АЦП, настроены как аналоговые входы.

24.3.4 Компаратор АЦП

Модуль АЦП содержит прецизионный компаратор. Не инвертирующих вход компаратора связан с выходом аналогового мультиплексора через НЧ RC фильтр. Инвертирующий вход компаратора подключен к внешнему конденсатору.

По сигналу с выхода компаратор происходит захват значения таймера и запись в ADCAP, устанавливается в '1' бит ADCIF.

24.3.5 Аналоговый мультиплексор

16 аналоговых каналов мультиплицируются на один выход, подключенный к положительному входу компаратора. С помощью четырех битов ADCON0<7:4> выбирается с какого канала будет выполняться преобразование сигнала.

24.3.6 Программируемый источник тока

Четыре бита ADCON1<7:4> используются для управления программируемым источником тока, с помощью которого формируется линейно нарастающее напряжение. Возможность изменения тока заряда конденсатора позволяет компенсировать частоту генератора и допуск внешнего конденсатора для широкого диапазона напряжений входного сигнала.

При установке бита ADRST в '1' источник тока отсоединяется от вывода CDAC. Источник тока подключен к выводу, когда ADRST=0.

Программируемый источник тока связан с выводом CDAC, он используется для заряда внешнего конденсатора, на котором формируется линейно нарастающее напряжение (см. рисунок 24-1).

24.3.7 Разрядность АЦП, быстродействие, диапазон напряжений, выбор конденсатора

Для выполнения преобразования пользователь должен выбрать следующие параметры:

- Разрядность результата преобразования;
- Скорость преобразования;
- Диапазон входных напряжений;
- Емкость внешнего конденсатора.

Разрядность АЦП - это число бит, используемое в ADTMR для представления измеренного входного напряжения. Время преобразования напрямую зависит от разрядности АЦП. В таблице 24-1 представлена зависимость разрядности АЦП и максимального времени преобразования.

Максимальное время преобразования может быть вычислено по формуле:

Время преобразования = $(1/F_{OSC}) \bullet 2^N$

Где F_{OSC} - частота тактового сигнала, N - желаемая разрядность АЦП.

При тактовой частоте 4МГц время преобразования равно 16.384мс для 16-разрядного результата и 256мкс для 10-разрядного результата.

Таблица 24-1 Значение инициализации ADTMR и время преобразования

Разрядность (бит)	**Значение, записываемое в ADTMR**	**Максимальное время преобразования**		
		Циклов	**20МГц**	**4МГц**
16	0000h	65536 T_{OSC}	3.28 мс	16.38 мс
15	8000h	32768 T_{OSC}	1.64 мс	8.2 мс
14	C000h	16384 T_{OSC}	820 мкс	4.1 мс
13	E000h	8192 T_{OSC}	410 мкс	2.05 мс
12	F000h	4096 T_{OSC}	204.8 мкс	1.03 мс
11	F800h	2048 T_{OSC}	102.4 мкс	500 мкс
10	FC00h	1024 T_{OSC}	51.2 мкс	250 мкс

Емкость внешнего конденсатор зависит от следующих параметров:

- Диапазон напряжений входного сигнала (необходимо использовать максимальный диапазон для всех каналов);
- Время преобразования;
- Режим работы программируемого источника тока.

Выбор этих параметров необходимо сделать таким образом, чтобы минимизировать время между переключением компаратора (установка бита ADCIF) и переполнением таймера ADTMR (установка бита OVFIF). Это будет гарантировать, что используется полный диапазон счета таймера ADTMR для получения результата преобразования.

Вычислить емкость конденсатора можно по следующей формуле:

C = (время преобразования в секундах) • (тока заряда конденсатора) / (полная шкала напряжений)

В таблице 24-2 представлены примеры значения емкости конденсатора для требуемого разрешения, скорости преобразования и диапазона входных напряжений.

Конденсатор, подключенный к выводу CDAC, должен иметь минимальный ток утечки. Внешний конденсатор необходимо разрежать перед каждым преобразованием установкой в '1' бита ADRST (ADCON0<1>). Время разряда (ADRST=1) зависит от характеристик внешнего конденсатора.

Таблица 24-2 Выбор емкости внешнего конденсатора (F_{OSC} = 4МГц)

Разрядность АЦП (бит)	**Время преобразования (мс)**	**Полная шкала (В)**	**Параметры источника тока**		**Вычисленное значение емкость**	**Ближайшее стандартное значение емкости**
			ADDAC3:ADDAC0	**Выходной ток (мкА)**		
16	16.384	3.5	1100	27	0.126 мкФ	0.12 мкФ
		2.0	1010	22.5	0.184 мкФ	0.18 мкФ
		1.5	1011	24.75	0.270 мкФ	0.27 мкФ
14	4.096	3.5	1101	29.25	34 нФ	33 нФ
		2.0	1011	24.75	50.7 нФ	47 нФ
		1.5	1100	27	73.7 нФ	68 нФ
12	1.024	3.5	1101	29.25	8.56 нФ	8.2 нФ
		2.0	1001	20.25	10.4 нФ	10 нФ
		1.5	1010	22.5	15.4 нФ	15 нФ
10	0.256	3.5	1011	24.75	1.81 нФ	1.8 нФ
		2.0	1010	22.5	2.88 нФ	2.7 нФ
		1.5	1011	24.75	4.22 нФ	3.9 нФ

24.4 Другие аналоговые модули

Для приложений со смешанными сигналами требуются дополнительные аналоговые модули:

- Прецизионный источник опорного напряжения;
- Делитель опорного напряжения.

24.4 Прецизионный источник опорного напряжения

Схема опорного напряжения формирует стабильное напряжение 1.2В для АЦП, детектора пониженного напряжения и делителя опорного напряжения. Опорное напряжение подается на 4 канал аналогового мультиплексора. Для включение источника опорного напряжения необходимо сбросить в '0' бит REFOFF (SLPCON<5>). Источник опорного напряжения необходимо включать при любом преобразовании АЦП.

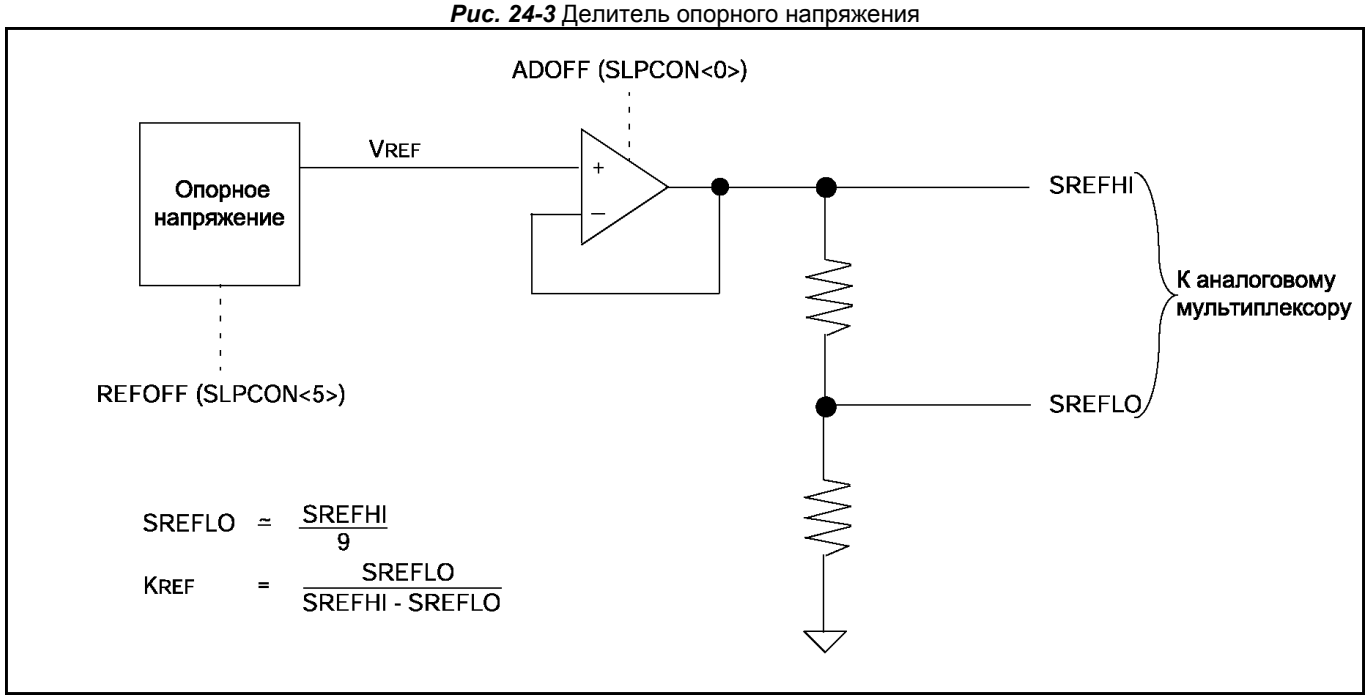
Калибровочная константа источника опорного напряжения хранится в калибровочной области EPROM памяти микроконтроллера.

24.4.2 Делитель опорного напряжения

Схема делителя опорного напряжения состоит из буферного усилителя и резистивного делителя. Схема формирует два других опорных напряжения SREFHI и SREFLO (см. рисунок 24-3). SREFHI - номинальное значение равно напряжению опорного источника 1.2В. Номинальное значение напряжения SREFLO - 0.13В. Эти напряжения доступны на двух каналах аналогового мультиплексора. Программное обеспечение пользователя может измерять напряжения SREFHI, SREFLO и с учетом калибровочных констант K_{REF}, K_{BG} откорректировать смещение АЦП и линейность преобразования.

Дополнительную информацию смотрите в документации AN624.

Рис. 24-3 Делитель опорного напряжения



24.5 Калибровочные параметры

Модуль АЦП имеет несколько аналоговых компонентов. Подобно всей CMOS технологии параметрические значения изменяются в зависимости от рабочей температуры, напряжения питания и времени эксплуатации. Модуль АЦП был разработан таким образом, чтобы минимизировать эффект этих изменений. Кроме того, каждый микроконтроллер с модулем АЦП откалиброван при заключительном заводском испытании, сохраняя калибровочные константы в EPROM памяти. Приложение пользователя может обращаться к калибровочным константам, чтобы математически компенсировать изменение параметров работы модуля АЦП.

Список калибровочных констант смотрите в таблице 24-3. 32-разрядное значение с плавающей точкой состоит из одного байта экспоненты и трех байт мантиссы. Дополнительную информацию относительно операций с плавающей точкой смотрите в документации AN575.

Таблица 24-3 Калибровочные константы

Параметр	Обозначение	Число байт	Представление значения
Коэффициент АЦП	K_{REF}	4	32 бита с плавающей точкой
Коэффициент источника опорного напряжения	K_{BG}	4	32 бита с плавающей точкой

Дополнительную информацию по использованию калибровочных констант смотрите в документации AN624.

24.5.1 Использование калибровочных констант

Калибровочные константы K_{REF}, K_{BG} должны использовать программой пользователя с целью достижения максимальной точности преобразования АЦП.

24.5.2 Изменение параметров

В таблице 24-4 представлен разброс параметров аналоговых компонентов модуля АЦП в случае использования калибровочных констант и пренебрежение ими. Если точность преобразования достаточна без калибровки модуля АЦП, то калибровочные константы можно не использовать. Если необходима большая точность, то в расчетах требуется использовать калибровочные константы.

Таблица 24-4 Изменение параметров

Обозначение	Параметр	Максимальный разброс параметров без использования калибровочных констант	Разброс параметров при использовании калибровочных констант
K_{REF}	Точность АЦП	± 2.2%	±0.13
K_{BG}	Точность источника опорного напряжения	± 4.2%	±0.058%

24.5.3 Программирование микроконтроллеров

24.5.3.1 Однократно программируемые микроконтроллеры

Однократно программируемые микроконтроллеры могут быть запрограммированы так же, как и любой другой микроконтроллер PIC16CXXX. Калибровочная область защищается от записи во время заводских испытаний.

24.5.3.2 Микроконтроллеры с окном для УФ стирания памяти

Предупреждение.
В микроконтроллерах с окном для УФ стирания калибровочная область не должна быть защищена от записи. При стирании памяти программ калибровочная информация тоже сотрется. Нельзя повторно записать калибровочную информацию, если была установлена защита от записи.

Калибровочные константы должны быть прочитаны и сохранены перед стиранием памяти микроконтроллера. Не существует способа обновить калибровочную информацию, если она была потеряна.

24.6 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Я использую рекомендованную емкость конденсатора и режим работы источника тока (согласно таблицы 24-2), но диапазон входного напряжения не соответствует требуемому?

Ответ 1:

Таблица является хорошей отравной точкой, но не включает все разновидности параметров приложений (тактовая частота микроконтроллера отличается от 4МГц, качество внешнего конденсатора, рабочая температура). Выполнение преобразования напряжения опорного источника позволит скорректировать параметры программируемого источника тока, чтобы гарантировать требуемый диапазон входных напряжений. Пример текста программы коррекции измерений (P14_RV10.asm) входит в комплект демонстрационной платы PICDEM-14A и доступен для загрузки с узлов технической поддержки **www.microchip.com** и **www.microchip.ru**.

Вопрос 2: Я использую PIC14C000, который имеет в своем составе датчик температуры. Результаты измерения температуры несколько завышены. Что может быть причиной?

Ответ 2:

Это может быть вызвано саморазогревом кристалла микроконтроллера. Причиной разогрева кристалла могут быть:

- Повышенный втекающий/вытекающий ток портов ввода/вывода;
- Рассеивание мощности при работе микроконтроллера (PIC14C000 может работать в SLEEP режиме);
- Температура корпуса микроконтроллера имеет малый коэффициент связи с внешней температурой.

Для лучших результатов измерения температуры потребляемая мощность микроконтроллера должна быть минимальной. Калибровка микроконтроллера выполняется в состоянии малого энергопотребления.

Вопрос 3: На результат преобразования АЦП влияет переключение силовых компонентов на печатной плате. Что можно сделать, чтобы минимизировать влияние переключений.

Ответ 3:

Компоненты, способные потреблять большой ток, могут приводить к повышению потенциала "земли" схемы относительно остальной части платы. Для минимизации этого эффекта рекомендуется применять в устройстве две системных "земли". Первая "земля" - аналоговая, используется для аналоговых сигналов (внешний конденсатор АЦП, делитель источника опорного напряжения и т.д.). Большие токи и питание цифровой схемы не должно присутствовать на аналоговой "земле".

Вторая "земля" - цифровая, используется для всей остальной цифровой логики в устройстве. Цифровая логика является источником помех на шинах питания, поэтому должны быть приняты соответствующие схемотехнические решения, чтобы минимизировать помехи.

Две "земли" должны быть соединены между собой на выводе микроконтроллера PICmicro. Идеальным случаем является, когда два вида "земли" схемы расположены в разных частях печатной платы устройства. В большинстве случаев это можно реализовать на двухсторонней печатной плате. Один слой используется для цепей "земли", разделенные между собой. Второй слой используется для сигнальных цепей.

24.7 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с модулем интегрирующего АЦП в микроконтроллерах PICmicro MCU:

Документ	Номер
PIC14C000 Calibration Parameters Калибровочные параметры в PIC14C000	AN621
PIC14C000 A/D Theory and Implementation Теория и использование АЦП в PIC14C000	AN624
Lead Acid Battery Charger Implementation using the PIC14C000 Зарядное устройство кислотных батарей на PIC14C000	AN626

Раздел 25. Модуль LCD

Содержание

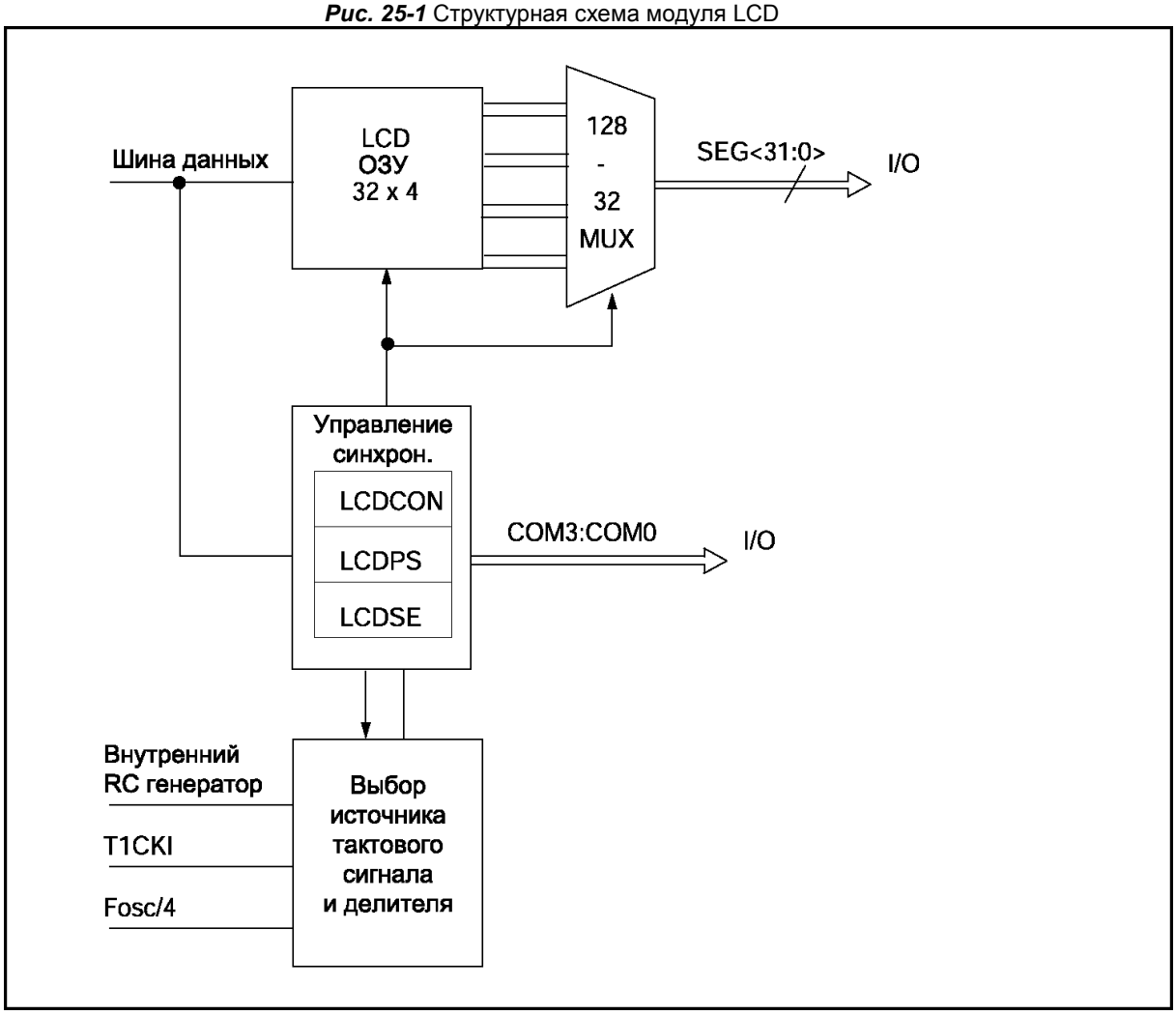
25.1 Введение

Модуль LCD формирует все необходимые сигналы синхронизации для управления статическими или мультиплицированными ЖКИ с поддержкой до 32 сегментов, 4 общих выводов и управление пикселями ЖКИ (вывод информации на ЖКИ).

Для управления модулем LCD используется три регистра (LCDCON, LCDSE и LCDPS), с помощью которых можно настроить параметры работы модуля, и до 16 регистров данных (LCD00-LCD15), где сохраняется массив данных пикселей. В нормальном режиме работы состояние регистров управления должно соответствовать используемому ЖКИ. При инициализации модуля LCD в первую очередь необходимо выбрать число общих выводов и сегментов ЖКИ, затем нужно настроить синхронизацию ЖКИ.

После настройки модуля LCD выполняется запись в регистры данных для определения состояния отдельных пикселей. Включение модуля LCD производится установкой в '1' бита LCDEN (LCDCON<7>). Модуль LCD может работать в SLEEP режиме микроконтроллера, при этом бит SLPEN (LCDCON<6>) должен быть сброшен в '0'.

Рис. 25-1 Структурная схема модуля LCD



25.2 Управляющие регистры

Регистр LCDCON:

R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
LCDEN	**SLPEN**	**-**	**VGEN**	**CS1**	**CS0**	**LMUX1**	**LMUX0**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как '0'
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **LCDEN:** Включение модуля LCD
1 = модуль LCD включен
0 = модуль LCD выключен

бит 6: **SLPEN:** Бит разрешения перевода LCD модуля в SLEEP режим
1 = модуль LCD приостанавливает работу в SLEEP режиме микроконтроллера
0 = модуль LCD продолжает работать в SLEEP режиме микроконтроллера

бит 5: **Не используется:** читается как ‘0’

бит 4: **VGEN:** Бит включения генератора напряжений
1 = внутренний генератор напряжений для LCD модуля включен
0 = внутренний генератор напряжений для LCD модуля выключен, напряжения формируются внешней схемой

биты 3-2: **CS1:CS0:** Выбор источника тактового сигнала
00 = $F_{OSC}/256$
01 = T1CKI (Таймер 1)
1x = внутренний RC генератор

биты 1-0: **LMUX1:LMUX0:** Число общих выводов
Определяет число общих выводов и метод формирования.

LMUX1:LMUX0	Мультиплексор	Метод формирования	Максимальное число сегментов
00	Статический (COM0)	Статический	32
01	1/2 (COM0, 1)	1/3	31
10	1/3 (COM0, 1, 2)	1/3	30
11	1/4 (COM0, 1, 2, 3)	1/3	29

Регистр LCDPS:

U-0	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x
-	-	-	-	**LP3**	**LP2**	**LP1**	**LP0**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как '0'
–n – значение после POR
–x – неизвестное значение после POR

биты 7-4: **Не используются:** читаются как ‘0’

биты 3-0: **LP3:LP0:** Длительность фрейма, выбор предделителя

LMUX1:LMUX0	Мультиплексор	Частота фрейма
00	Статический	Частота источника / (128 x (LP3:LP0 + 1))
01	1/2	Частота источника / (128 x (LP3:LP0 + 1))
10	1/3	Частота источника / (96 x (LP3:LP0 + 1))
11	1/4	Частота источника / (128 x (LP3:LP0 + 1))

Регистры данных LCDD:

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SEGs COMc	**SEGs COMc**	**SEGs COMc**	**SEGs COMc**	**SEGs COMc**	**SEGs COMc**	**SEGs COMc**	**SEGs COMc**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как '0'
–n – значение после POR
–x – неизвестное значение после POR

биты 7-0: **SEGsCOMc:** Данные пикселя сегмента s общего вывода c
1 = пиксель включен
0 = пиксель выключен

Регистр LCDSE:

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
SE29	**SE27**	**SE20**	**SE16**	**SE12**	**SE9**	**SE5**	**SE0**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как '0'
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **SE29:** Выбор режима работы выводов COM1/SEG31-COM3/SEG29
1 = выводы работают как драйверы сегментов LCD
0 = выводы работают как цифровые входы

Примечание. Состояние битов LMUX1:LMUX0 (настройка драйвера общего выхода) имеет более высокий приоритет, чем SE29.

бит 6: **SE27:** Выбор режима работы выводов SEG28 и SEG27
1 = выводы работают как драйверы сегментов LCD
0 = выводы работают как цифровые входы

бит 5: **SE20:** Выбор режима работы выводов SEG26-SEG20
1 = выводы работают как драйверы сегментов LCD
0 = выводы работают как цифровые входы

бит 4: **SE16:** Выбор режима работы выводов SEG19-SEG16
1 = выводы работают как драйверы сегментов LCD
0 = выводы работают как цифровые входы

бит 3: **SE12:** Выбор режима работы выводов SEG15-SEG12
1 = выводы работают как драйверы сегментов LCD
0 = выводы работают как цифровые входы

бит 2: **SE9:** Выбор режима работы выводов SEG11-SEG09
1 = выводы работают как драйверы сегментов LCD
0 = выводы работают как цифровые входы

бит 1: **SE5:** Выбор режима работы выводов SEG08-SEG05
1 = выводы работают как драйверы сегментов LCD
0 = выводы работают как цифровые входы

бит 0: **SE0:** Выбор режима работы выводов SEG04-SEG00
1 = выводы работают как драйверы сегментов LCD
0 = выводы работают как цифровые входы

Примечание. При сбросе по включению питания (POR) все выводы, мультиплицированные с модулем LCD, работают как драйверы LCD.

25.3 Синхронизация LCD

Модуль LCD имеет 3 возможных источника тактовых импульсов и поддерживает статический, 1/2, 1/3 и 1/4 режим синхронизации.

25.3.1 Источник тактового сигнала для модуля LCD

Источником тактового сигнала для модуля LCD может быть:

- Внутренний RC генератор, предназначен для работы модуля LCD в SLEEP режиме микроконтроллера или при низкой тактовой частоте микроконтроллера;
- Генератор TMR1, используется при работе модуля LCD в SLEEP режиме микроконтроллера или при низкой тактовой частоте микроконтроллера;
- Тактовый сигнал микроконтроллера, деленный на 256.

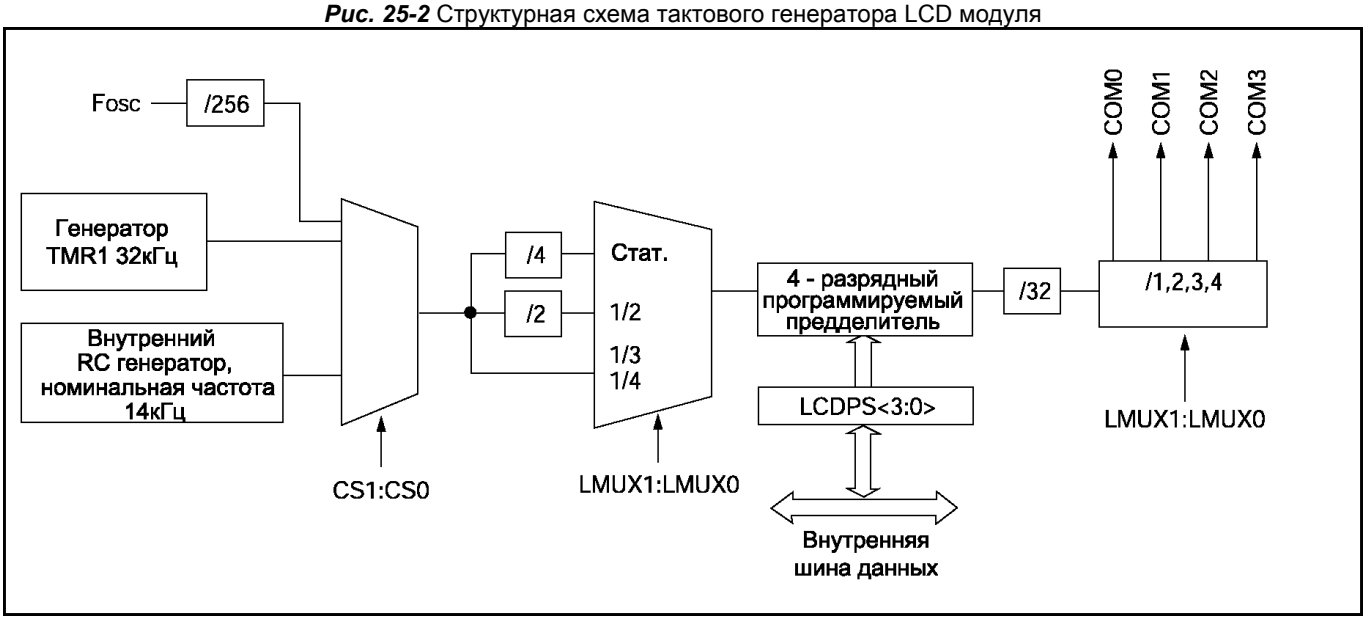
Первый источник тактовых импульсов - внутренний RC генератор с номинальной частотой 14кГц. Этот генератор обеспечивает минимальную частоту тактовых импульсов для LCD модуля, он может использоваться для продолжения работы модуля LCD, когда микроконтроллер находится в SLEEP режиме. RC генератор выключен после сброса микроконтроллера по включению питания (POR), если выбран иной источник тактовых импульсов или модуль LCD выключен.

Второй источник - внешний генератор TMR1. Этот генератор может обеспечивать низкую частоту тактовых импульсов для модуля LCD, он может использоваться для продолжения работы модуля LCD, когда микроконтроллер находится в SLEEP режиме. Считается, что частота сигнала тактового генератора 32кГц. Чтобы использовать генератор TMR1, в качестве источника тактовых импульсов для модуля LCD, необходимо установить в '1' бит T1OSEN (T1CON<3>).

Третий источник - тактовый сигнал микроконтроллера деленный на 256. Этот коэффициент деления выбран таким образом, чтобы обеспечить частоту тактовых импульсов модуля LCD 32кГц при тактовой частоте микроконтроллера 8МГц. Коэффициент деления тактового сигнала не может быть изменен. С помощью управляющих битов регистра LCDPS можно выбрать длительность фрейма ЖКИ.

Источник тактового сигнала выбирается битами CS1:CS0 (LCDCON<3:2>). Пояснения программирования модуля LCD смотрите на рисунке 25-1.

Рис. 25-2 Структурная схема тактового генератора LCD модуля



25.3.2 Синхронизация мультиплексора

Схема синхронизации формирует сигналы для работы от 1 до 4 общих выводов, зависит от типа выбранного ЖКИ. Режим работы определяется битами LMUX1:LMUX0 (LCDCON<1:0>). В таблице 25-1 представлены формулы для вычисления частоты фрейма.

Таблица 25-1 Формулы вычисления частоты фрейма

LMUX1:LMUX0	Мультиплексор	Частота фрейма
00	Статический	Частота источника / (128 x (LP3:LP0 + 1))
01	1/2	Частота источника / (128 x (LP3:LP0 + 1))
10	1/3	Частота источника / (96 x (LP3:LP0 + 1))
11	1/4	Частота источника / (128 x (LP3:LP0 + 1))

Таблица 25-2 Аппроксимированная частота фрейма в Гц
(Генератор TMR1 32кГц или F_{OSC} = 8МГц)

LP3:LP0	Статический	1/2	1/3	1/4
2	85	85	114	85
3	64	64	85	64
4	51	51	68	51
5	43	43	57	43
6	37	37	49	37
7	32	32	43	32

Таблица 25-3 Аппроксимированная частота фрейма в Гц
(Внутренний RC генератор 14кГц)

LP3:LP0	Статический	1/2	1/3	1/4
0	109	109	146	109
1	55	55	73	55
2	36	36	49	36
3	27	27	36	27

Рис 25-3 Временная диаграмма статического режима

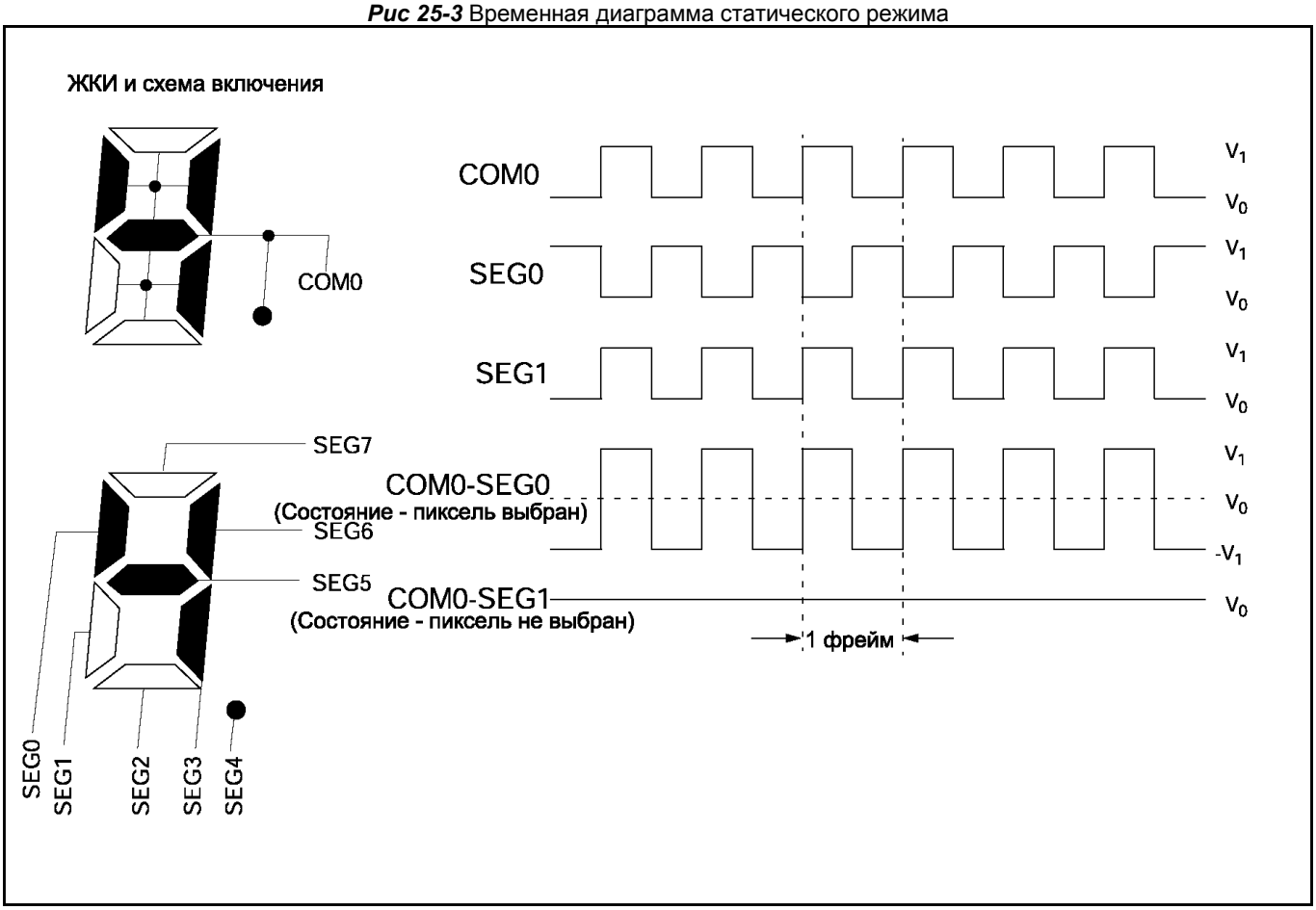


Рис 25-4 Временная диаграмма 1/2 MUX, 1/3 BIAS

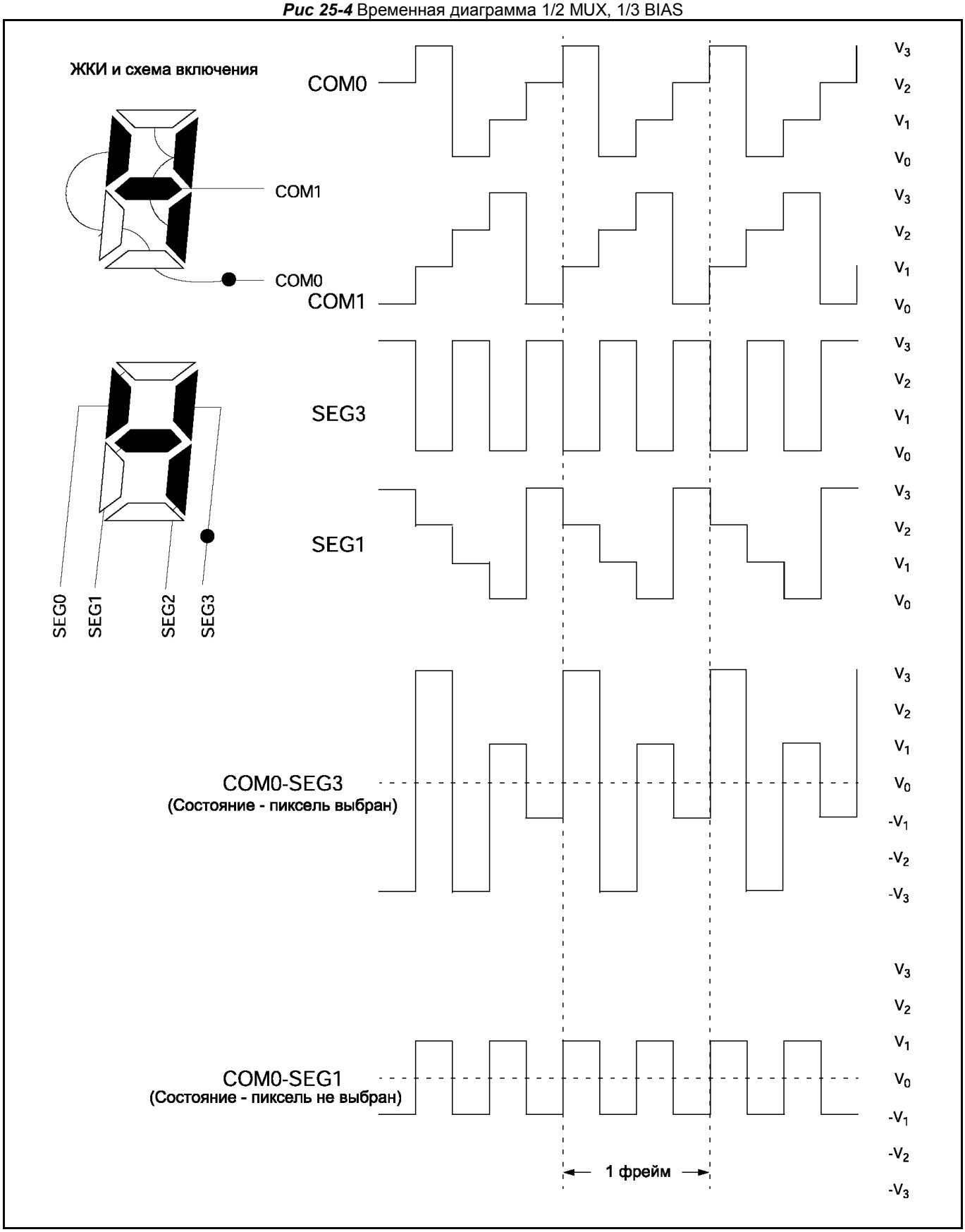


Рис 25-5 Временная диаграмма 1/3 MUX, 1/3 BIAS

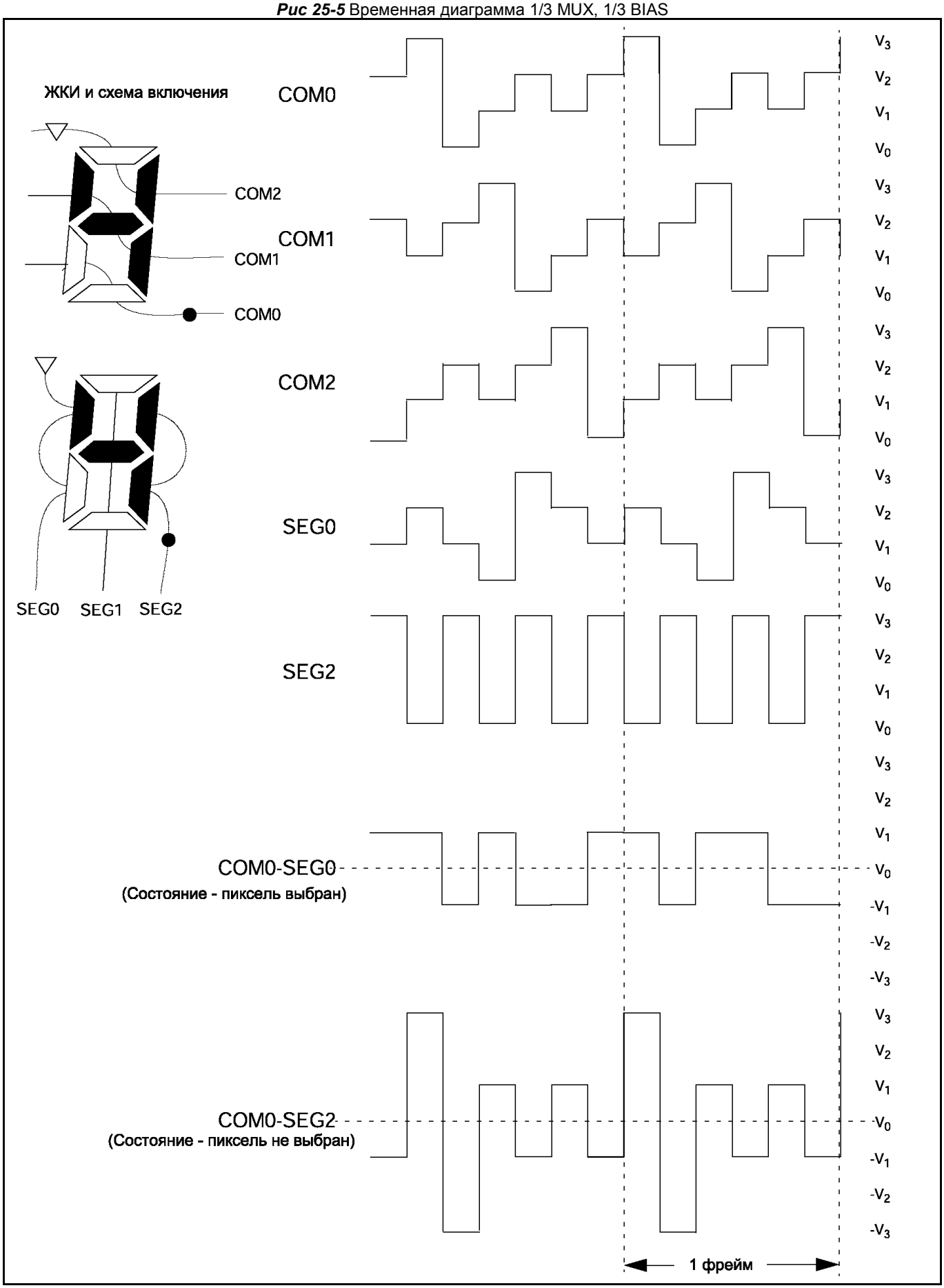
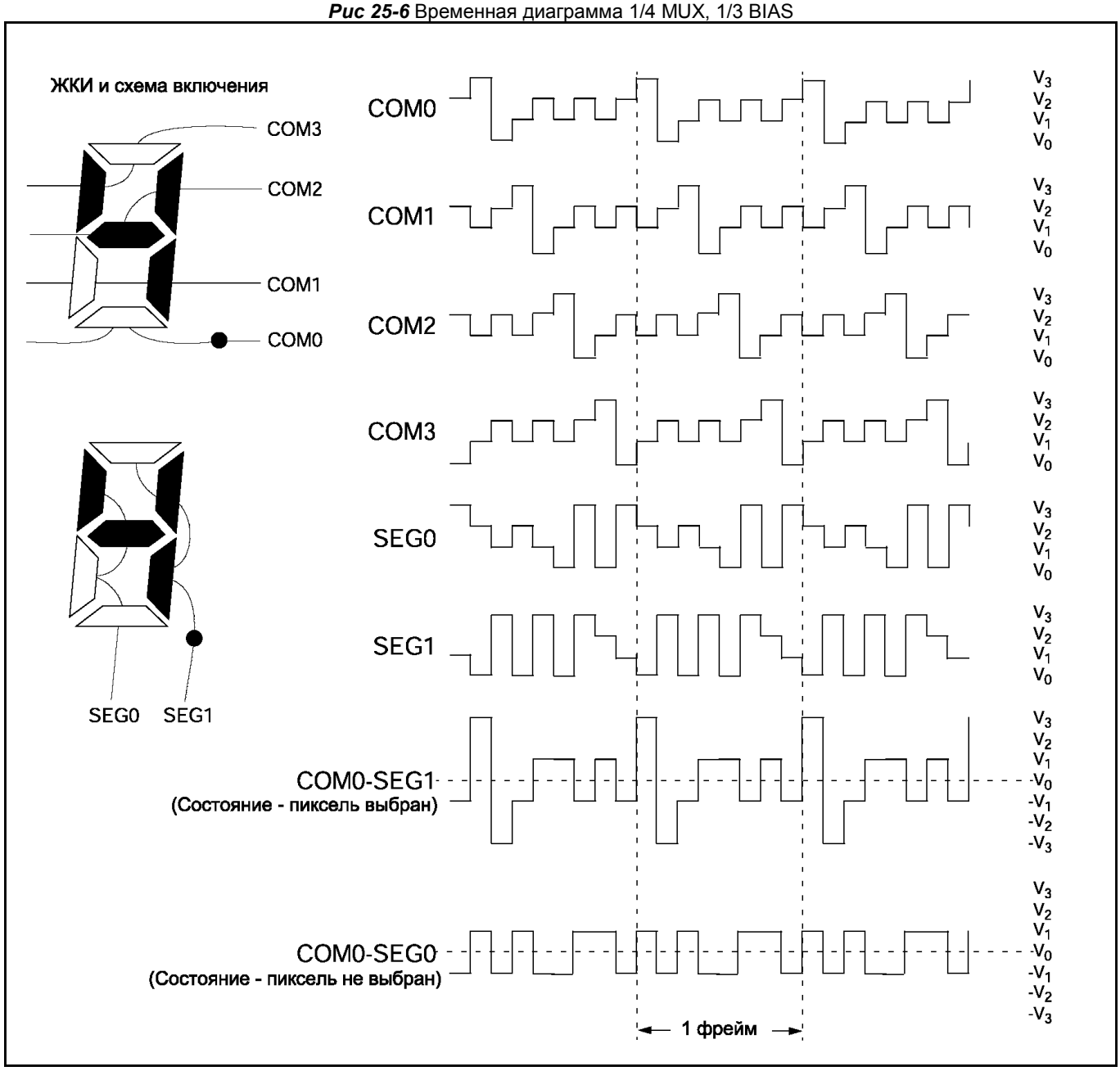


Рис 25-6 Временная диаграмма 1/4 MUX, 1/3 BIAS

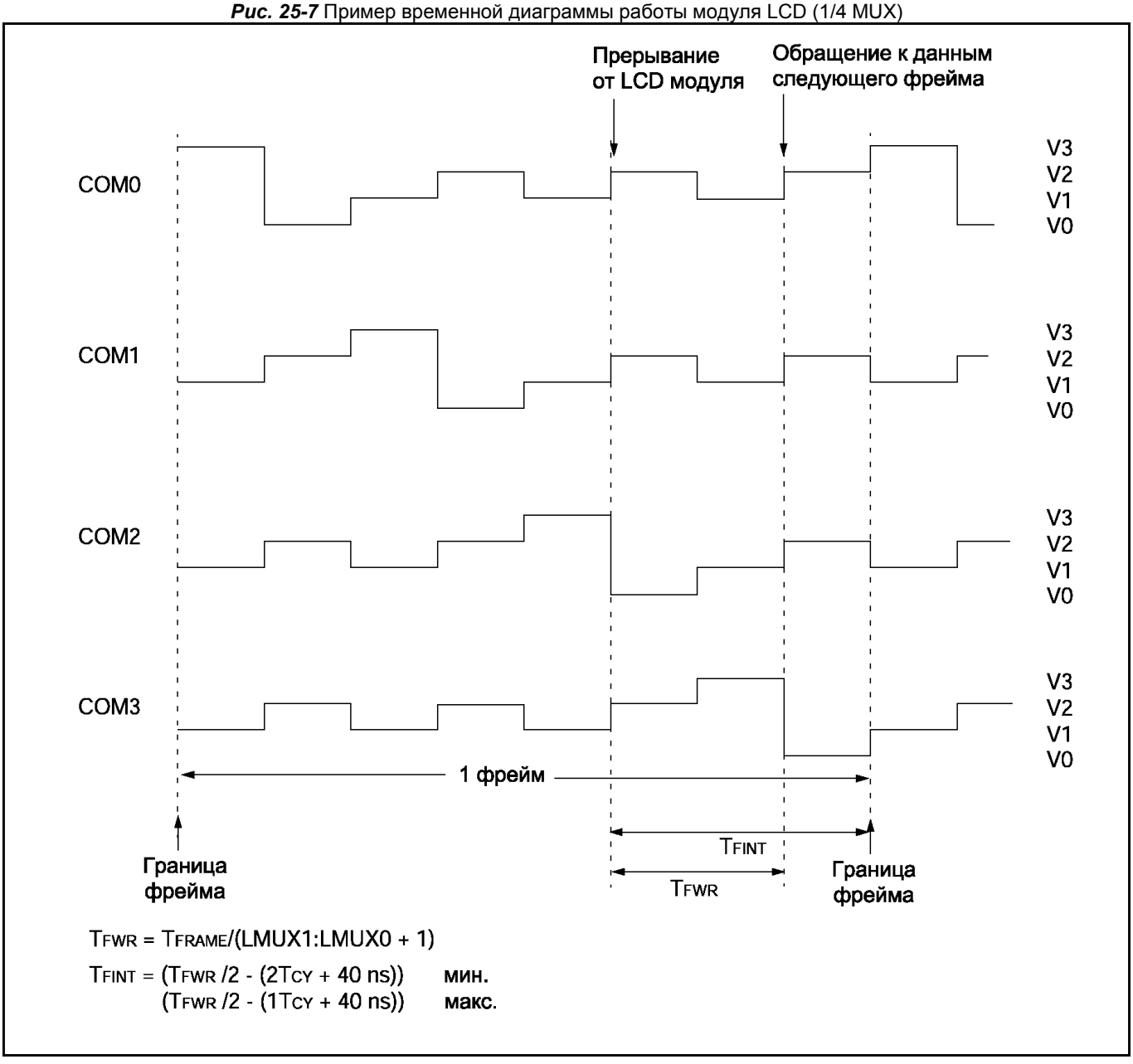


25.4 Прерывания от модуля LCD

Модуль LCD формирует прерывания, соответствующие синхронизации фреймов ЖКИ. Прерывания могут использоваться для координации записи данных пикселей перед началом нового фрейма. Запись данных перед началом фрейма позволяет организовать четкую смену информации на ЖКИ. Также прерывания могут использоваться для синхронизации внешних событий с ЖКИ. Например, интерфейс доступа к внешней микросхеме драйвера ЖКИ AY0438 может синхронизироваться для одновременного изменения выводимой на ЖКИ информации.

Началом нового фрейма считается передний фронт сигнала на общем выводе COM0. Прерывание формируется немедленно, после считывания модулем LCD необходимых данных для текущего фрейма (см. рисунок 25-7). LCD модуль начнет запрашивать данные для следующего фрейма в пределах интервала времени T_{FWR} после прерывания.

Рис. 25-7 Пример временной диаграммы работы модуля LCD (1/4 MUX)



$T_{FWR} = T_{FRAME}/(LMUX1:LMUX0 + 1)$

$T_{FINT} = (T_{FWR}/2 - (2T_{CY} + 40\ ns))$ мин.

$(T_{FWR}/2 - (1T_{CY} + 40\ ns))$ макс.

25.5 Управление пикселями ЖКИ

25.5.1 Регистры данных LCDD

Регистры данных ЖКИ содержат биты, которые определяют состояния каждого пикселя ЖКИ (каждый бит влияет на один пиксель). В таблице 25-4 показано влияние каждого бита в регистрах LCDD на соответствующий сигнал сегмента и общего вывода. Регистры данных ЖКИ, не влияющие на работу ЖКИ, могут использоваться как регистры общего назначения.

Таблица 25-4 Регистры LCDD

Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Сброс POR, BOR	Другие сбросы
LCDD00	SEG07 COM0	SEG06 COM0	SEG05 COM0	SEG04 COM0	SEG03 COM0	SEG02 COM0	SEG01 COM0	SEG00 COM0	xxxx xxxx	xxxx xxxx
LCDD01	SEG15 COM0	SEG14 COM0	SEG13 COM0	SEG12 COM0	SEG11 COM0	SEG10 COM0	SEG09 COM0	SEG08 COM0	xxxx xxxx	xxxx xxxx
LCDD02	SEG23 COM0	SEG22 COM0	SEG21 COM0	SEG20 COM0	SEG19 COM0	SEG18 COM0	SEG17 COM0	SEG16 COM0	xxxx xxxx	xxxx xxxx
LCDD03	SEG31 COM0	SEG30 COM0	SEG29 COM0	SEG28 COM0	SEG27 COM0	SEG26 COM0	SEG25 COM0	SEG24 COM0	xxxx xxxx	xxxx xxxx
LCDD04	SEG07 COM1	SEG06 COM1	SEG05 COM1	SEG04 COM1	SEG03 COM1	SEG02 COM1	SEG01 COM1	SEG00 COM1	xxxx xxxx	xxxx xxxx
LCDD05	SEG15 COM1	SEG14 COM1	SEG13 COM1	SEG12 COM1	SEG11 COM1	SEG10 COM1	SEG09 COM1	SEG08 COM1	xxxx xxxx	xxxx xxxx
LCDD06	SEG23 COM1	SEG22 COM1	SEG21 COM1	SEG20 COM1	SEG19 COM1	SEG18 COM1	SEG17 COM1	SEG16 COM1	xxxx xxxx	xxxx xxxx
LCDD07	SEG31 COM1(1)	SEG30 COM1	SEG29 COM1	SEG28 COM1	SEG27 COM1	SEG26 COM1	SEG25 COM1	SEG24 COM1	xxxx xxxx	xxxx xxxx
LCDD08	SEG07 COM2	SEG06 COM2	SEG05 COM2	SEG04 COM2	SEG03 COM2	SEG02 COM2	SEG01 COM2	SEG00 COM2	xxxx xxxx	xxxx xxxx
LCDD09	SEG15 COM2	SEG14 COM2	SEG13 COM2	SEG12 COM2	SEG11 COM2	SEG10 COM2	SEG09 COM2	SEG08 COM2	xxxx xxxx	xxxx xxxx
LCDD10	SEG23 COM2	SEG22 COM2	SEG21 COM2	SEG20 COM2	SEG19 COM2	SEG18 COM2	SEG17 COM2	SEG16 COM2	xxxx xxxx	xxxx xxxx
LCDD11	SEG31 COM2(1)	SEG30 COM2(1)	SEG29 COM2	SEG28 COM2	SEG27 COM2	SEG26 COM2	SEG25 COM2	SEG24 COM2	xxxx xxxx	xxxx xxxx
LCDD12	SEG07 COM3	SEG06 COM3	SEG05 COM3	SEG04 COM3	SEG03 COM3	SEG02 COM3	SEG01 COM3	SEG00 COM3	xxxx xxxx	xxxx xxxx
LCDD13	SEG15 COM3	SEG14 COM3	SEG13 COM3	SEG12 COM3	SEG11 COM3	SEG10 COM3	SEG09 COM3	SEG08 COM3	xxxx xxxx	xxxx xxxx
LCDD14	SEG23 COM3	SEG22 COM3	SEG21 COM3	SEG20 COM3	SEG19 COM3	SEG18 COM3	SEG17 COM3	SEG16 COM3	xxxx xxxx	xxxx xxxx
LCDD15	SEG31 COM3(1)	SEG30 COM3(1)	SEG29 COM3(1)	SEG28 COM3	SEG27 COM3	SEG26 COM3	SEG25 COM3	SEG24 COM3	xxxx xxxx	xxxx xxxx

Примечание 1. Эти пиксели не отображаются, биты могут использоваться как биты общего назначения.

25.5.2 Включение сегментов

Регистр LCDSE предназначен для выбора режима работы группы выводов микроконтроллера. Каждая группа выводов может быть настроена для работы в качестве драйверов LCD модуля или цифровых входов/выходов. Для настройки группы выводов как цифровые входы/выходы соответствующий бит в регистре LCDSE должен быть сброшен в '0'.

Если вывод работает как цифровой вход/выход, то соответствующий бит TRIS управляет направлением данных. Установка любого бита в регистре LCDSE отменяет действие соответствующих битов TRIS.

Примечание 1. При сбросе по включению питания (POR) все выводы, мультиплицированные с модулем LCD, работают как драйверы LCD.

Примечание 2. Состояние битов LMUX1:LMUX0 (настройка драйвера общего выхода) имеет более высокий приоритет, чем SE29 (настройка выводов RD7, RD6 и RD5).

Пример 25-1 Статический MUX, 32 сегмента

```
BCF     STATUS,RP0      ; Выбрать банк 2
BSF     STATUS,RP1      ;
BCF     LCDCON,LMUX1    ; Выбрать статический MUX
BCF     LCDCON,LMUX0    ;
MOVLW   0xFF            ; Выводы портов PORTD, E,F,G
MOVWF   LCDSE           ; используются модулем LCD
```

Пример 25-2 1/3 MUX, 13 сегментов

```
BCF     STATUS,RP0      ; Выбрать банк 2
BSF     STATUS,RP1      ;
BSF     LCDCON,LMUX1    ; Выбрать 1/3 MUX
BCF     LCDCON,LMUX0    ;
MOVLW   0x87            ; Выводы портов PORTD<7:0> и PORTE<6:0>
MOVWF   LCDSE           ; используются модулем LCD
```

25.6 Генератор напряжений

Существует два метода формирования напряжений для модуля LCD: внутренний генератор напряжения, внешняя резистивная цепочка.

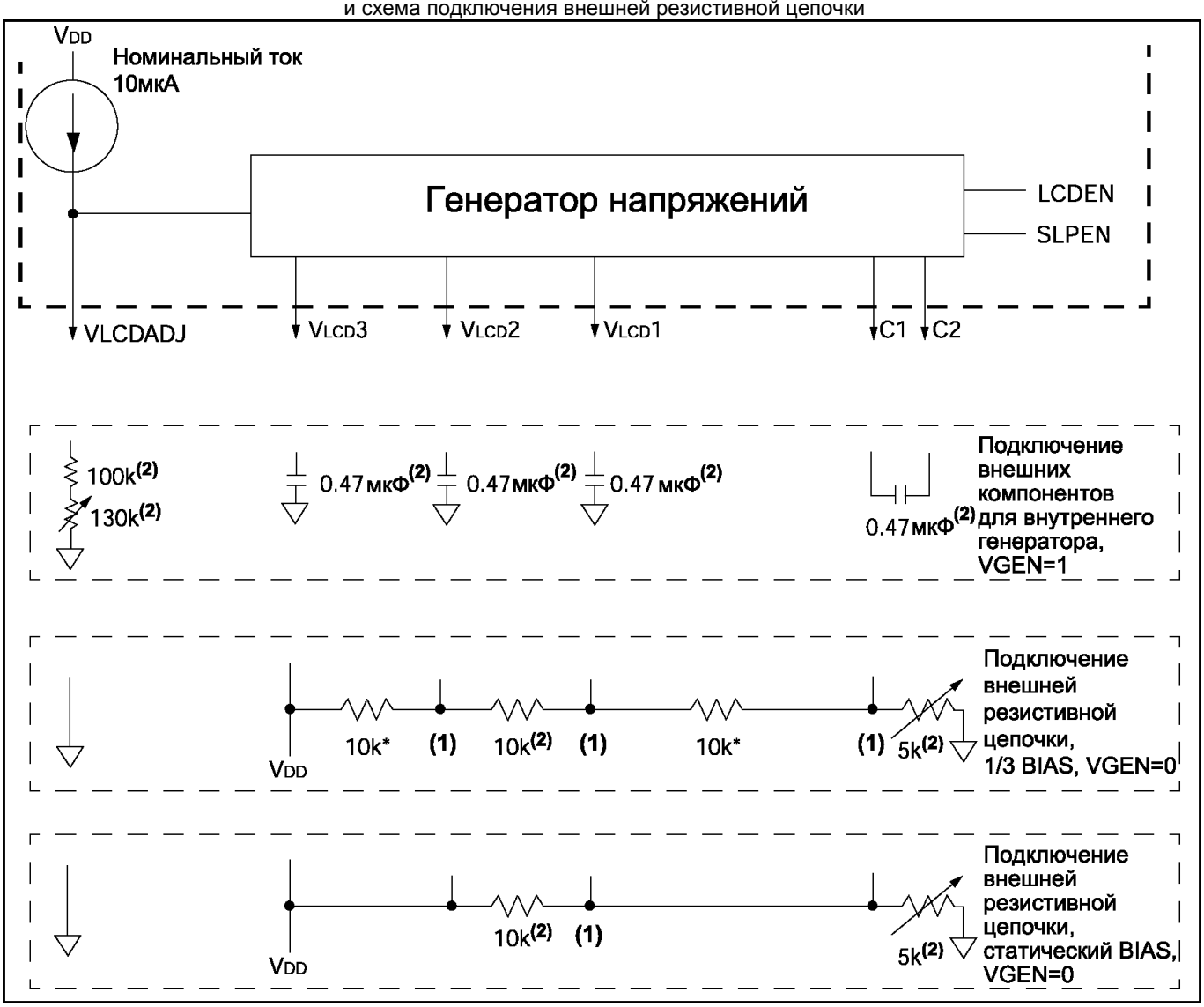
25.6.1 Внутренний генератор напряжений

Структурная схема внутреннего генератор напряжения модуля LCD показана на рисунке 25-8. Генератор напряжения формирует стабилизированное напряжение от 1.0В до 2.3В независимо от напряжения питания. Настройка генератора напряжений осуществляется внешним резистором, подключенным к выводу VLCDADJ. С помощью этого резистора можно изменять контрастность ЖКИ (изменение основного напряжения на выводе V_{LCD1}). На основе напряжения V_{LCD1} формируется два других напряжения: V_{LCD2} = 2 V_{LCD1}, V_{LCD3} = 3 V_{LCD1}. Когда внутренний генератор напряжений выключен, вывод V_{LCD3} внутренне подключен к V_{DD}. Смотрите в разделе "Электрические характеристики" параметры конденсаторов и потенциометра.

25.6.2 Внешняя резистивная цепочка

Модуль LCD может использовать внешнюю резистивную цепочку для формирования необходимых напряжений. На рисунке 25-8 показана схема подключения резисторов для статического и 1/3 MUX режима. Для использования внешней резистивной цепочки бит VGEN (LCDCON<4>) должен быть сброшен в '0'.

Рис. 25-8 Структурная схема внутреннего генератора напряжений и схема подключения внешней резистивной цепочки



Примечания:

1. Точка включения необязательного фильтрующего конденсатора.
2. Оценочные значения.

25.7 Работа модуля LCD в SLEEP режиме микроконтроллера

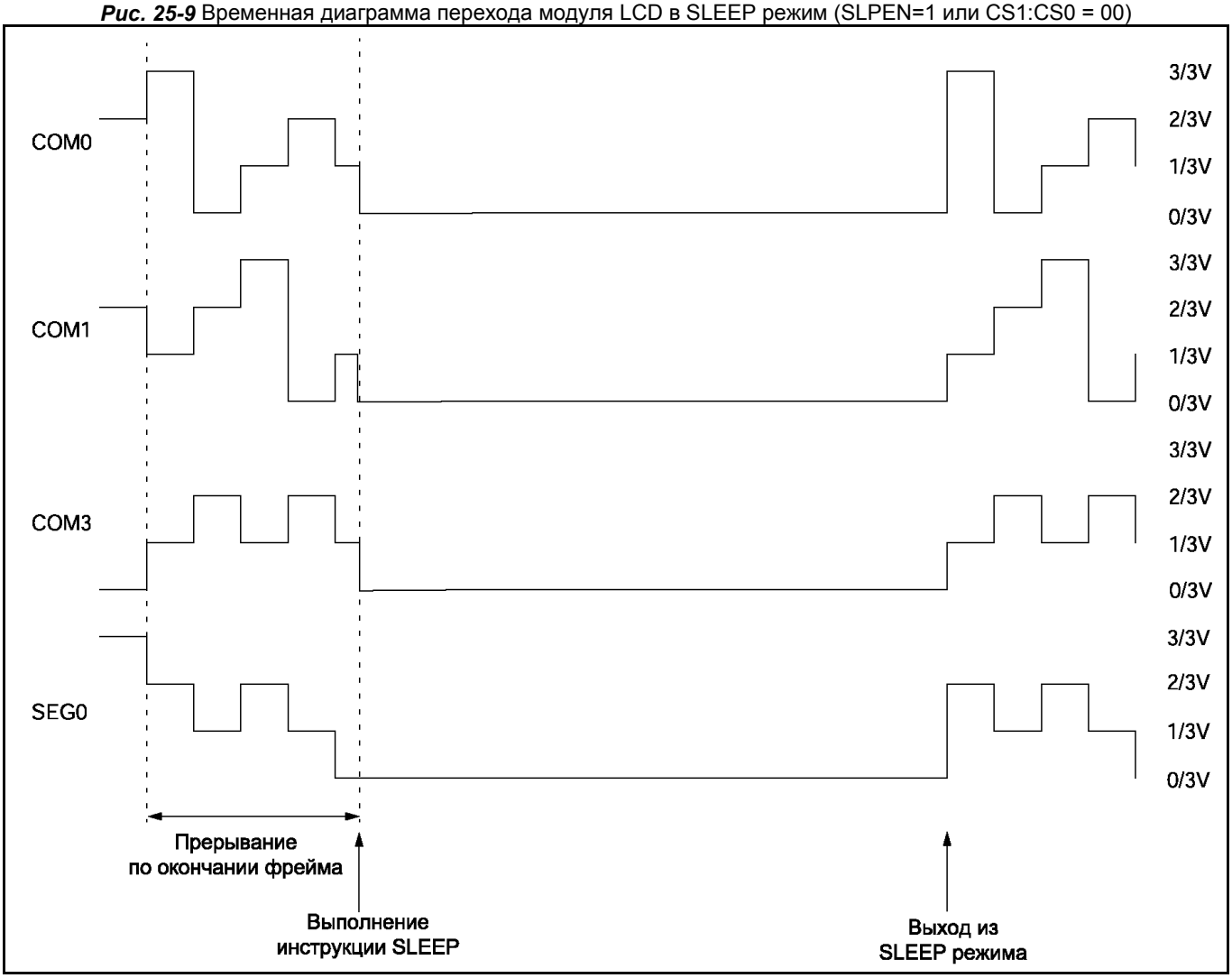
Модуль LCD может продолжать работать в SLEEP режиме микроконтроллера, выбор осуществляется битом SPLEN (LCDCON<6>). Если SLPEN = 0, то модуль LCD продолжает работать в SLEEP режиме микроконтроллера.

Если SLPEN = 1, то после выполнения команды SLEEP модуль LCD перейдет в режим пониженного энергопотребления, прекратив формировать управляющие сигналы. На рисунке 25-9 показана временная диаграмма перехода в SLEEP режим модуля LCD. Для гарантированного завершения формирования фрейма команда SLEEP должна быть выполнена немедленно, после обнаружения границы фрейма (прерывания от модуля LCD могут использоваться для обнаружения границы фрейма). Смотрите раздел 25.4 "Прерывания от модуля LCD".

Если разрешена работа модуля LCD в SLEEP режиме микроконтроллера, то содержимое регистров LCDD будет отображаться на ЖКИ, при этом данные не могут быть изменены. В этом случае необходимо использовать внутренний RC генератор или генератор TMR1 для формирования тактового сигнала модуля LCD. Модуль LCD не будет потреблять меньше тока, однако полное потребление системы будет уменьшено за счет выключения ядра микроконтроллера и других периферийных модулей.

Примечание. При работе модуля LCD в SLEEP режиме микроконтроллера должен использоваться внутренний RC генератор или генератор TMR1.

Рис. 25-9 Временная диаграмма перехода модуля LCD в SLEEP режим (SLPEN=1 или CS1:CS0 = 00)



25.8 Эффект сброса

После сброса микроконтроллера модуль LCD выключен, но выводы, мультиплицированные с модулем LCD, настраиваются как драйверы LCD. Это гарантирует защиту ЖКИ стекла от повреждения возможным постоянным напряжением.

25.9 Настройка модуля LCD

Для настройки модуля LCD рекомендуется следующая последовательность действий:

1. Выбрать параметры синхронизации фреймов, биты LP3:LP0 (LCDPS<3:0>).
2. Настроить необходимые выводы микроконтроллера для работы в качестве драйвера ЖКИ (регистр LCDSE).
3. Настроить модуль LCD с помощью регистра LCDCON:
   - Режим работы мультиплексора (биты LMUX1:LMUX0);
   - Выбрать источник тактового сигнала (биты CS1:CS0);
   - Выбрать режим работы генератора напряжения для модуля LCD (бит VGEN);
   - Выбрать режим работы модуля LCD в SLEEP режиме микроконтроллера (бит SLPEN).
4. Записать начальные данные в регистры LCDD00 - LCDD15.
5. Сбросить флаг прерывания от модуля LCD (бит LCDIF), и если необходимо, разрешить прерывания установкой бита LCDIE в '1'.
6. Включить модуль LCD установив в '1' бит LCDEN (LCDCON<7>).

25.10 Коэффициент дискриминации

С помощью коэффициента дискриминации можно вычислить максимальную контрастность ЖКИ. В первом примере представлен расчет для статического режима (см. рисунок 25-3). Напряжения V_1 и V_0 будут иметь значения 1 и 0. Следующий шаг - создание уравнения для отдельного фрейма, чтобы оценить отображение индивидуального пикселя в включенном и выключенном состоянии при различных режимах синхронизации. В остальной части вычисляется коэффициент дискриминации.

Пример 25-3 Вычисление коэффициента дискриминации (статический MUX)

$$\text{COMx - SEGx [ON]} = 1 - 1, \quad V_{DC} = 0$$

$$\text{COMx - SEGx [OFF]} = 0 + 0, \quad V_{DC} = 0$$

$$V_{RMS}\,[\text{ON}] = \Delta V \sqrt{\frac{(1)^2 + (-1)^2}{2}} = 1\Delta V$$

$$V_{RMS}\,[\text{OFF}] = \Delta V \sqrt{\frac{(0)^2 + (0)^2}{2}} = 0\Delta V$$

$$D = \frac{V_{RMS}\,[\text{ON}]}{V_{RMS}\,[\text{OFF}]} = \frac{1\Delta V}{0\Delta V} = \infty$$

См. рисунок 25-3.

25

В следующем примере вычисляется коэффициент дискриминации для режима 1/4 MUX, 1/3 BIAS (см. рис. 25-6). В этом примере значения 3, 2, 1 и 0 для напряжений V_3, V_2, V_1 и V_0. Уравнения для статического и RSM напряжения, вычисление коэффициента дискриминации показаны в примере 25-4.

Пример 25-4 Вычисление коэффициента дискриминации (1/4 MUX)

COM0 - SEGx [ON] = 3 - 3 + 1 - 1 + 1 - 1 + 1 - 1 $V_{DC} = 0$
COM0 - SEGx [OFF] = 1 - 1 - 1 + 1 - 1 + 1 - 1 + 1 $V_{DC} = 0$

$$V_{RMS}\,[ON] = \Delta V \sqrt{\frac{(3)^2 + (-3)^2 + (1)^2 + (-1)^2 + (1)^2 + (-1)^2 + (1)^2 + (-1)^2}{8}} = \sqrt{3}\,\Delta V$$

$$V_{RMS}\,[OFF] = \Delta V \sqrt{\frac{(1)^2 + (-1)^2 + (-1)^2 + (1)^2 + (-1)^2 + (1)^2 + (-1)^2 + (1)^2}{8}} = \Delta V$$

$$D = \frac{V_{RMS}\,[ON]}{V_{RMS}\,[OFF]} = \frac{\sqrt{3}\,\Delta V}{1\,\Delta V} = 1.732$$

См. рисунок 25-6.

Как можно судить по представленным примерам - ЖКИ со статической синхронизацией имеют лучшую контрастность. Большее значение мультиплицированния - меньший коэффициент дискриминации, меньшая контрастность ЖКИ.

В таблице 25-5 показано соотношение напряжений V_{OFF}, V_{ON} при различных режимах синхронизации.

Для увеличения контрастности ЖКИ можно увеличить разницу напряжений между уровнями.

Таблица 25-5 Коэффициент дискриминации

	1/3 BIAS		
	V_{OFF}	**V_{ON}**	**D**
Статический	0	1	∞
1/2 MUX	0.333	0.745	2.236
1/3 MUX	0.333	0.638	1.915
1/4 MUX	0.333	0.577	1.732

25.11 Формирование напряжения для модуля LCD

Из всех возможных способов формирования напряжения для LCD модуля выделяется два основных:

- Резистивная цепочка;
- Внутренний генератор.

Метод резистивной цепочки, показанный на рисунке 25-10, наиболее часто используется при высоком напряжении питания V_{DD}. Этот метод формирования напряжений нескольких уровней для модуля LCD считается наиболее простым и дешевым. Независимо от числа включенных пикселей ток остается неизменным. Напряжение V_3 обычно подключается к напряжению питания (внутренними цепями микроконтроллера или внешней схемой).

Выбор сопротивления резисторов определяется двумя факторами: качество отображения информации и потребляемая мощность. Качество отображения информации зависит от параметров синхронизации ЖКИ. Необходимо учитывать, что ЖКИ является емкостной нагрузкой из-за чего возникает искажение управляющих сигналов (токи заряда/разряда). Влияние емкостной нагрузки можно снизить уменьшив сопротивление резистивной цепочки. Однако, это приводит к большему энергопотреблению (за счет большего тока через резистивную цепочку). Для ЖКИ большего размера следует использовать резисторы с меньшим сопротивлением, чтобы получить качественное отображение информации.

Допускается параллельно резисторам включать конденсаторы, которые снижают искажения управляющих сигналов обеспечивая требуемые токи заряда/разряда. Ориентировочные значения резисторов: R - от 1кОм до 50 кОм; потенциометр - от 5кОм до 200кОм.

Рис. 25-10 Схема включения резистивной цепочки

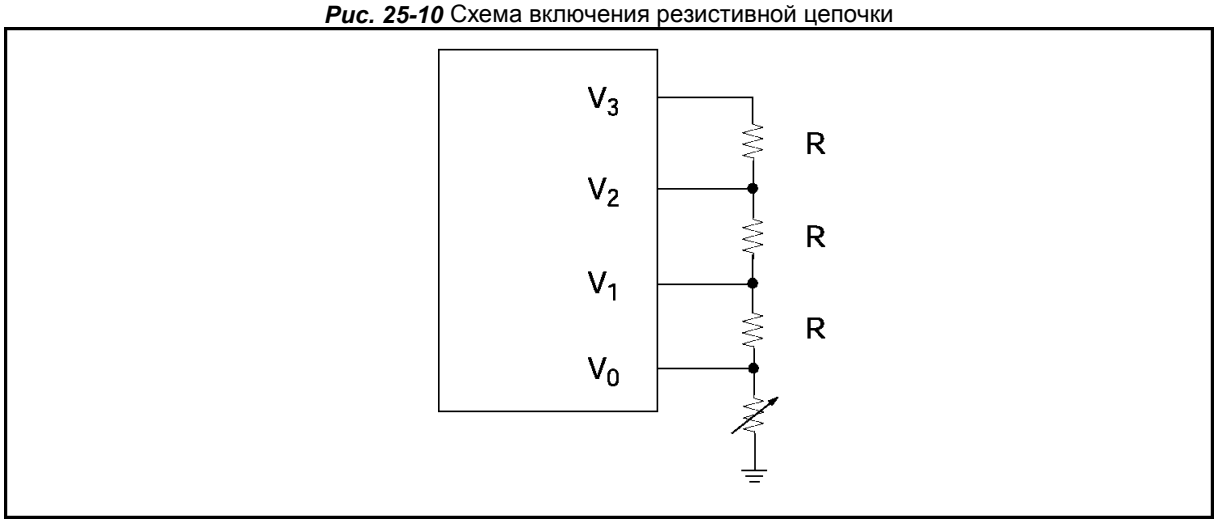
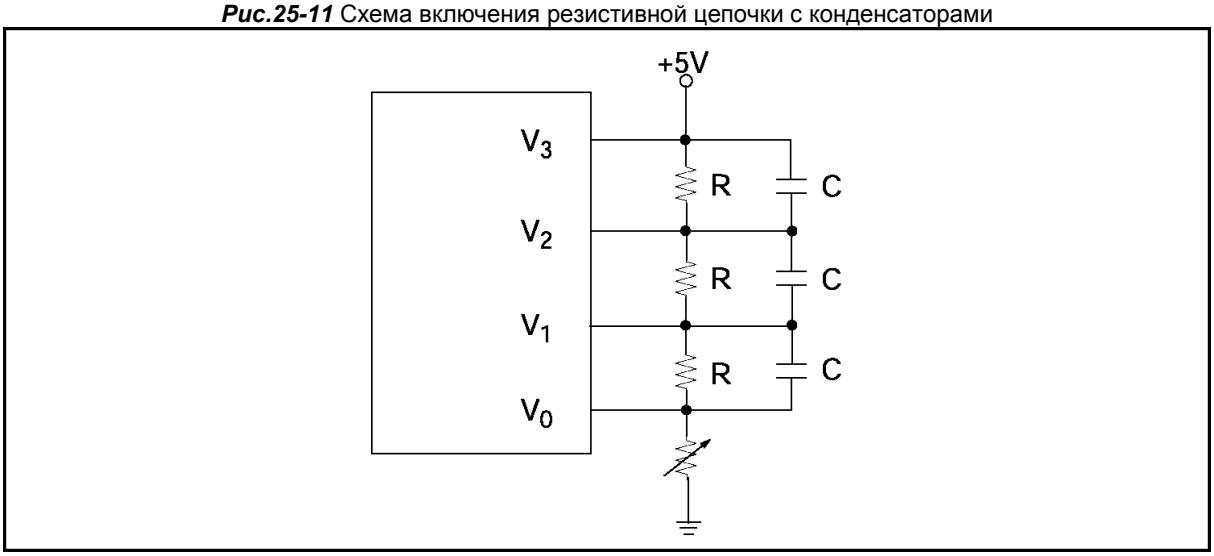
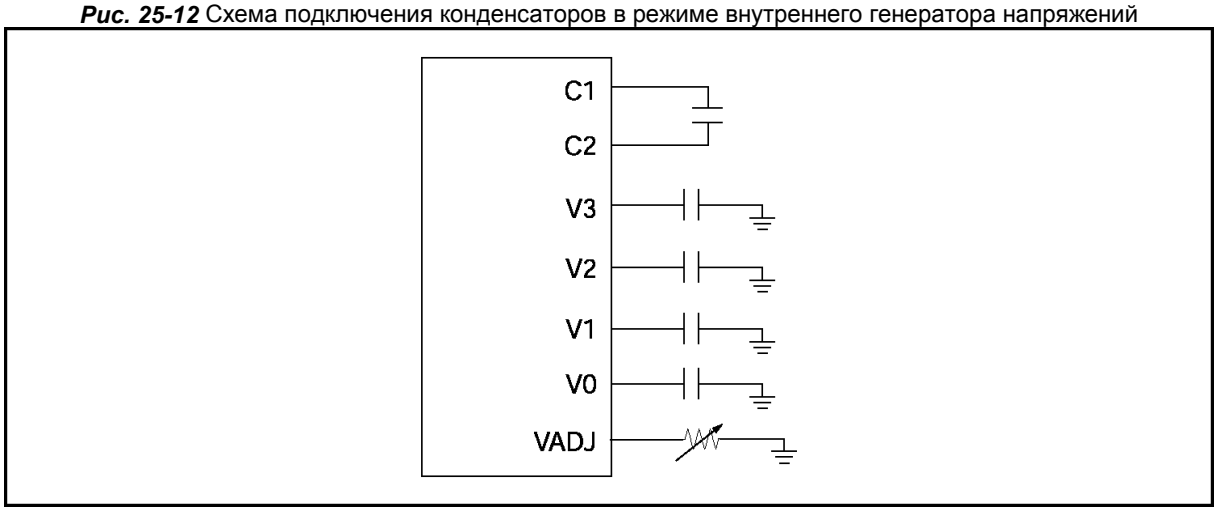


Рис.25-11 Схема включения резистивной цепочки с конденсаторами



Внутренний генератор напряжений целесообразно использовать при низком напряжении питания V_{DD}, т.к. оно может быть увеличено до требуемого уровня. Генератор напряжений требует подключение внешнего заряжающего и фильтрующих конденсаторов для каждого напряжения ЖКИ (см. рисунок 25-12). Применяемые конденсаторы должны иметь низкий ток утечки. Другой особенность использования внутреннего генератора напряжений является то, что потребляемый ток зависит от числа включенных пикселей, что особенно важно в устройствах с питанием от батареек.

Рис. 25-12 Схема подключения конденсаторов в режиме внутреннего генератора напряжений



25.12 Контрастность

Контрастность отображения информации сильно зависит от типа ЖКИ, режима синхронизации, уровня напряжений. Как было отмечено ранее, потенциометр используется для управления контрастностью ЖКИ, устанавливая разность между каждым из напряжений модуля LCD. Чем больше разность напряжений, тем выше контрастность.

25.13 ЖКИ стекло

Характеристика ЖКИ стекла зависит от применяемых материалов. В приложении В дан список некоторых производителей ЖКИ. Обратитесь к производителю для уточнения параметров выпускаемых ЖКИ.

25.14 Инициализация

В примере 25-5 показана инициализация модуля LCD, все сегменты очищены.

Пример 25-5 Инициализация модуля LCD

```
BCF     PIR1,LCDIF          ; Сбросить флаг прерываний от модуля LCD
BCF     STATUS,RP0          ; Выбрать банк 2
BSF     STATUS,RP1
MOVLW   0x06                ; Установить частоту фрейма ~37Гц
MOVWF   LCDPS
MOVLW   0xff                ; Все выводы работают как драйверы LCD
MOVWF   LCDSE
MOVLW   0x17                ; Работа в SLEEP режиме,
                            ; внутренний генератор напряжений включен
MOVWF   LCDCON              ; Тактовый сигнал от генератора TMR1, 1/4 MUX
CLRF    LCDD00              ; Очистить все регистры данных,
CLRF    LCDD01              ; все пиксели выключены
CLRF    LCDD02
CLRF    LCDD03
CLRF    LCDD04
CLRF    LCDD05
CLRF    LCDD06
CLRF    LCDD07
CLRF    LCDD08
CLRF    LCDD09
CLRF    LCDD10
CLRF    LCDD11
CLRF    LCDD12
CLRF    LCDD13
CLRF    LCDD14
CLRF    LCDD15
BSF     PIE1,LCDIE          ; Разрешить прерывания от модуля LCD
BSF     LCDCON,LCDEN        ; Включить модуль LCD
BCF     STATUS,RP1          ; Выбрать банк 0
```

25.15 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Не получается использовать некоторые выводы LCD как цифровые входы.
Ответ 1:
Проверьте, правильно настроены биты в регистре LCDSE, т.к. эти биты отменяют действие соответствующих битов TRIS.

Вопрос 2: Изображение на ЖКИ мерцает.
Ответ 2:
Вероятно, что очень маленькая частота фрейма. Изменить частоту фрейма можно в регистре LCDPS.

Вопрос 3: Изображение на ЖКИ не контрастно.
Ответ 3:
Причиной может быть малая разность между напряжениями LCD модуля:

1. Если Вы используете резистивную цепочку, то попробуйте изменить сопротивления цепочки. Вывод VLCDADJ должен быть подключен к "земле" схемы.
2. Если Вы используете внутренний генератор напряжений, то измените сопротивление на выводе VLCDADJ.

25.16 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с модулем LCD в микроконтроллерах PICmicro MCU:

Документ	Номер
Yet Another Clock Using the PIC16C92X Часы на микроконтроллере PIC16C92X	AN649
LCD Fundamentals Using PIC16C92x Microcontrollers Принципы использования модуля LCD в микроконтроллерах PIC16C92X	AN658
PICDEM3 Demo Board User's Guide Руководство пользователя по демонстрационной плате PICDEM3	DS51079

Раздел 26. Сторожевой таймер WDT и режим энергосбережения SLEEP

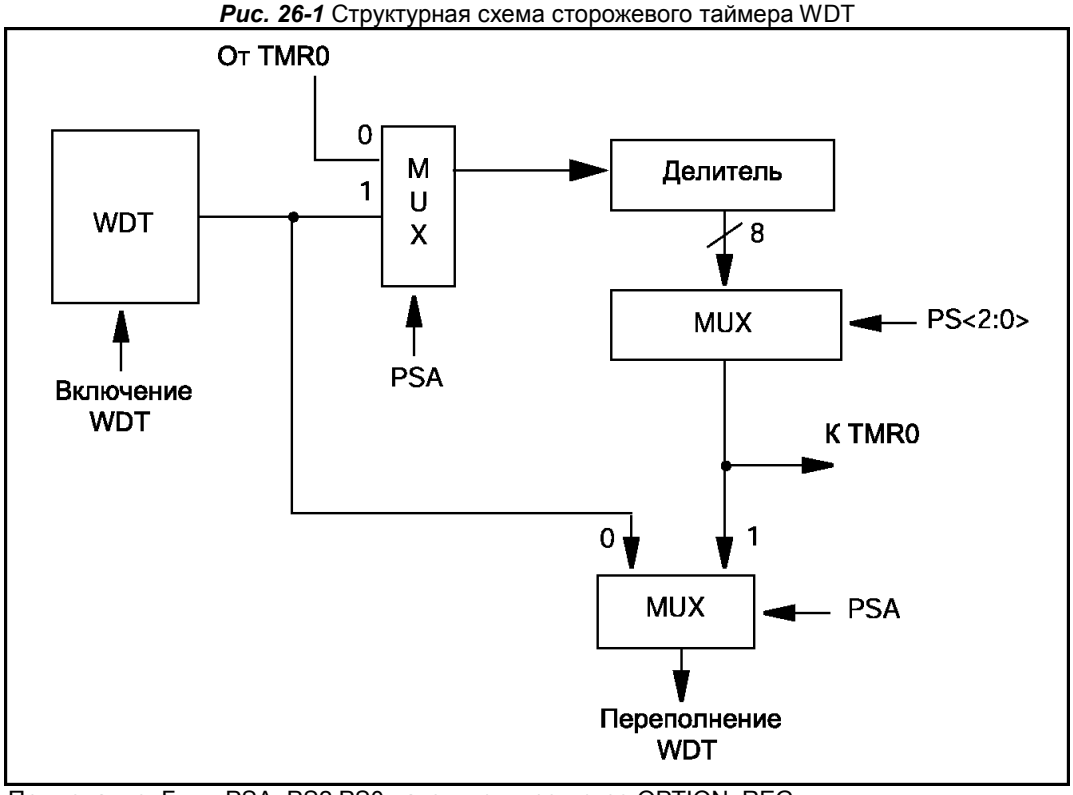
Содержание

26.1 Введение

Встроенный сторожевой таймер WDT работает от отдельного RC генератора, не требующего внешних компонентов (это отдельный RC генератор от генератора, подключенного к выводу OSC1/CLKIN). Структурная схема WDT показана на рисунке 26-1. Отдельный RC генератор позволяет работать сторожевому таймеру WDT при выключенном тактовом генераторе (выводы OSC1, OSC2) в SLEEP режиме микроконтроллера.

Бит включения/выключения WDT расположен в слове конфигурации. Если WDT включен, то его нельзя выключить программным способом.

Рис. 26-1 Структурная схема сторожевого таймера WDT



Примечание. Биты PSA, PS2:PS0 находятся в регистре OPTION_REG.

26.2 Управляющий регистр

Регистр OPTION_REG доступен для чтения и записи, содержит биты управления:

- Предварительным делителем TMR0/WDT;
- Активным фронтом внешнего прерывания RB0/INT;
- Подтягивающими резисторами на входах PORTB.

Примечание. Если предварительный делитель включен перед WDT, то коэффициент деления тактового сигнала для TMR0 равен 1:1.

Регистр OPTION_REG

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
-RBPU(1)	**INTEDG**	**T0CS**	**T0SE**	**PSA**	**PS2**	**PS1**	**PS0**
Бит 7							Бит 0

R – чтение бита
W – запись бита
U – не реализовано, читается как 0
–n – значение после POR
–x – неизвестное значение после POR

бит 7: **-RBPU(1)**: Включение подтягивающих резисторов на входах PORTB
1 = подтягивающие резисторы отключены
0 = подтягивающие резисторы включены

бит 6: **INTEDG**: Выбор активного фронта сигнала на входе внешнего прерывания INT
1 = прерывания по переднему фронту сигнала
0 = прерывания по заднему фронту сигнала

бит 5: **T0CS**: Выбор тактового сигнала для TMR0
1 = внешний тактовый сигнал с вывода T0CKI
0 = внутренний тактовый сигнал CLKOUT

бит 4: **T0SE**: Выбор фронта приращения TMR0 при внешнем тактовом сигнале
1 = приращение по заднему фронту сигнала (с высокого к низкому уровню) на выводе T0CKI
0 = приращение по переднему фронту сигнала (с низкого к высокому уровню) на выводе T0CKI

бит 3: **PSA**: Выбор включения предделителя
1 = предделитель включен перед WDT
0 = предделитель включен перед TMR0

биты 2-0: **PS2: PS0**: Установка коэффициента деления предделителя

Значение	Для TMR0	Для WDT
000	1:2	1:1
001	1:4	1:2
010	1:8	1:4
011	1:16	1:8
100	1:32	1:16
101	1:64	1:32
110	1:128	1:64
111	1:256	1:128

Примечание 1. В некоторых микроконтроллерах этот бит обозначается как -GPPU. Если микроконтроллер содержит бит -RBPU, то подтягивающие резисторы подключены к PORTB. Если микроконтроллер содержит бит -GPPU, то подтягивающие резисторы подключены к GPIO.

26.3 Работа с WDT

В нормальном режиме работы при переполнении WDT происходит сброс микроконтроллера. Если микроконтроллер находится в SLEEP режиме, переполнение WDT выводит его из режима SLEEP с продолжением нормальной работы. WDT выключен, если WDTE = 0 в слове конфигурации.

Переключение предделителя выполняется программным способом, т.е. переключение можно сделать во время выполнения программы.

Примечание. Для предотвращения случайного сброса микроконтроллера следует выполнять переключение предделителя от TMR0 к WDT как показано в примере 26-1, даже если WDT выключен.

В примере 26-1 первая часть изменения регистра OPTION_REG не должна выполняться, если желаемый коэффициент предделителя отличный от 1:1. Если требуется настройка коэффициента предделителя 1:1, то необходимо установить промежуточное значение коэффициента (отличное от 1:1), а затем установить коэффициент предделителя 1:1 в последней части изменения OPTION_REG. Если переключение выполнить другим способом, то будет неизвестен момент сброса микроконтроллера от WDT.

Переключение предделителя от WDT к TMR0 смотрите в примере 26-2.

Пример 26-1 Переключения предделителя от TMR0 к WDT

```
BSF         STATUS, RP0         ; Банк 1
MOVLW       b'xx0x0xxx'         ; Выбрать источник тактового сигнала и
MOVWF       OPTION_REG          ; коэффициент предделителя, отличный от 1:1
BCF         STATUS, RP0         ; Банк 0
CLRF        TMR0                ; Сбросить TMR0 и предделитель
BSF         STATUS, RP0         ; Банк 1
MOVLW       b'xxxx1xxx'         ; Включить предделитель перед WDT,
MOVWF       OPTION_REG          ; но не выбирать коэффициент деления
CLRWDT                          ; Сбросить WDT и предделитель
MOVLW       b'xxxx1xxx'         ; Выбрать новое значение коэффициента
MOVWF       OPTION_REG          ; предделителя
BCF         STATUS, RP0         ; Банк 0
```

Пример 26-2 Переключения предделителя от WDT к TMR0

```
CLRWDT                          ; Сбросить WDT и предделитель
BSF         STATUS, RPO         ; Банк 1
MOVLW       b’xxxx0xxx’         ; Включить предделитель перед TMR0 и
MOVWF       OPTION_REG          ; выбрать новое значение коэффициента деления
BCF         STSTUS, RPO         ; Банк 0
```

26.3.1 Период WDT

WDT имеет номинальное время переполнения 18мс (без предделителя). Время переполнения зависит от температуры, напряжения питания V_{DD} и разброса технологических параметров микроконтроллера (см. раздел "Электрические характеристики"). Если требуется большее время переполнения WDT, необходимо программно подключить предделитель в регистре OPTION_REG с максимальным коэффициентом деления 1:128. С включенным предделителем время переполнения может достигать 2.3с.

Команды CLRWDT и SLEEP сбрасывают сторожевой таймер и предделитель, если он подключен к WDT, откладывая сброс устройства.

В регистре STATUS бит -TO=0, если произошло переполнение WDT (сброс от WDT или выход из режима SLEEP).

26.3.2 Рекомендации по работе с WDT

Даже в самих плохих условиях работы требуется несколько секунд для переполнения WDT (минимальное напряжение питания V_{DD}, максимальная температура, максимальный коэффициент предделителя подключенного к WDT).

Примечание. При выполнении команды CLRWDT сбрасывается WDT и предделитель, если он подключен к WDT.

Таблица 26-1 Регистры и биты, связанные с работой WDT

Имя	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	Сброс POR, BOR	Другие сбросы
Слово конф.	MPEEN	BODEN	CP1	CP0	-PWRTE	WDTE	FOSC1	FOSC0	uuuu uuuu	uuuu uuuu
OPTION_REG	-RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111

Затененные биты не влияют на работу WDT.

26.4 Режим энергосбережения SLEEP

Режим SLEEP является таким режимом работы микроконтроллера, в котором микроконтроллер потребляет минимальный ток. Тактовый генератор выключен, поэтому тактовый сигнал не поступает ни в один модуль микроконтроллера. Переход в режим энергосбережения происходит по команде SLEEP.

При переходе в режим SLEEP сторожевой таймер WDT сбрасывается, но продолжает работать (если он включен). В регистре STATUS бит -PD сбрасывается в '0', бит -TO устанавливается в '1', тактовый генератор микроконтроллера выключен. Порты ввода/вывода остаются в том же состоянии, что и до выполнения команды SLEEP (высокий уровень, низкий уровень, третье состояние).

Для снижения энергопотребления в SLEEP режиме все каналы ввода/вывода должны быть подключены к V_{DD} или V_{SS} при отсутствии токов из внешней схемы через выводы портов, выходы модуля компараторов и источника опорного напряжения выключены. Выводы находящиеся в третьем состоянии должны иметь высокий или низкий уровень сигнала, чтобы избежать токов переключения входных буферов. Вход T0CKI должен быть подключен к V_{DD} или V_{SS} для снижения энергопотребления. Должны учитываться внутренние подтягивающие резисторы, включенные на входах PORTB. На входе -MCLR должен быть высокий уровень сигнала.

Некоторые модули микроконтроллера (сторожевой таймер WDT, детектор сброса по снижению напряжения питания BOR и др.) должны учитываться при определении потребляемого тока микроконтроллера в SLEEP режиме. Биты включения/выключения этих модулей находятся в слове конфигурации.

26.4.1 Выход из режима SLEEP

Микроконтроллер выйдет из режима SLEEP по одному из следующих событий:

1. Любой сброс микроконтроллера;
2. Переполнение сторожевого таймера WDT (если он разрешен);
3. Периферийное прерывание, которое может быть сгенерировано в SLEEP режиме микроконтроллера:
   - Внешнее прерывание INT;
   - Изменение уровня сигнала на входах RB7:RB4;
   - Переключение компараторов;
   - Завершение преобразования АЦП (когда используется внутренний RC генератор для АЦП);
   - Переполнение TMR1 в режиме асинхронного счетчика;
   - Прием/передача байта в режиме ведомого SPI/I^2C;
   - Прием/передача USART в ведомом синхронном режиме;
   - Граница фрейма LCD;
   - Завершение записи в EEPROM.

Первое событие вызывает сброс микроконтроллера. Два других события вызывают продолжение выполнения программы. Биты -TO и -PD в регистре STATUS могут использоваться для определения причины сброса микроконтроллера. Бит -PD сбрасывается в '0' при переходе в режим SLEEP. Бит -TO сбрасывается в '0' если произошло переполнение WDT.

При выполнении команды SLEEP происходит предвыборка следующей инструкции (PC+1). Если прерывание должно выводить микроконтроллер из режима SLEEP, соответствующий бит разрешения прерывания устанавливается в '1'. Микроконтроллер выходит из режима SLEEP независимо от состояния бита GIE. Если GIE=0, выполняется следующая инструкция после SLEEP без перехода по вектору прерываний. Если GIE=1, исполняется следующая инструкция после SLEEP и происходит переход на подпрограмму обработки прерываний (адрес 0004h). Когда выполнение какой-либо команды при выходе из режима SLEEP нежелательно, необходимо поле команды SLEEP использовать инструкцию NOP.

26.4.2 Выход из режима SLEEP по прерыванию

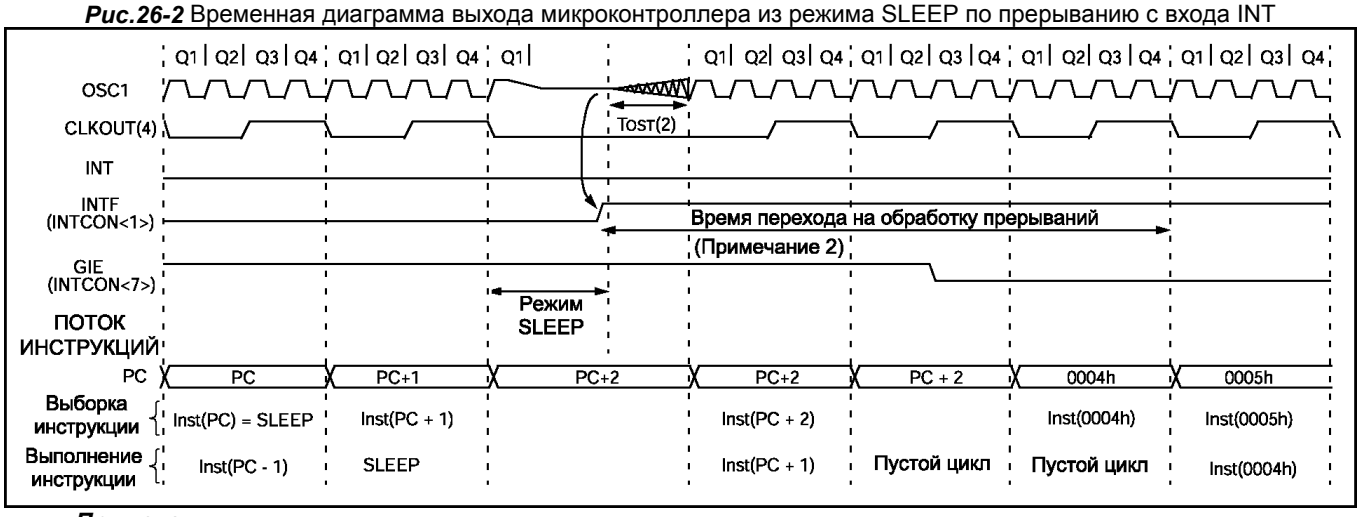
Когда бит глобального разрешения прерываний GIE сброшен в '0', а бит разрешения периферийных прерываний и соответствующий флаг прерывания установлен в '1', то возникнет одно из следующих событий:

- Если прерывание возникает перед выполнением команды SLEEP, то вместо инструкции SLEEP будет выполнен пустой цикл NOP, WDT и предделитель WDT не будут сброшены, бит -TO не будет установлен в '1', а бит -PD не будет сброшен в '0'.
- Если прерывание возникает в течение или после выполнения инструкции SLEEP, то микроконтроллер немедленно выйдет из режима SLEEP, а команда SLEEP выполняется полностью. WDT и предделитель WDT сброшены, бит -TO установлен в '1', бит -PD сброшен в '0'.

Даже если флаги прерываний были проверены перед выполнением команды SLEEP, они могут быть установлены в течение выполнения инструкции SLEEP. Для контроля полного выполнения команды SLEEP проверьте состояние бита -PD. Если -PD = 1, то вместо инструкции SLEEP был выполнен пустой цикл NOP.

Для гарантированного сброса WDT перед инструкцией SLEEP рекомендуется использовать команду CLRWDT.

Рис.26-2 Временная диаграмма выхода микроконтроллера из режима SLEEP по прерыванию с входа INT



Примечания:

1. Режим генератора XT, HS или LP.
2. T_{OST} = 1024 T_{OSC} (не масштабный рисунок). Для RC режима генератора задержка отсутствует.
3. Предполагается, что GIE=1. После выхода из режима SLEEP произойдет переход по вектору прерывания.
4. CLKOUT не доступен для этих режимов генератора, но показан для пояснения диаграммы.

26.5 Инициализация

Текст примера инициализации в настоящее время не подготовлен.

26.6 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Напряжение питания устройства снижается ниже допустимого уровня, а затем восстанавливается. При этом WDT не сбрасывает микроконтроллер и устройство работает неправильно.

Ответ 1:

WDT не предназначен для сброса микроконтроллера по снижению напряжения питания. WDT используется для предотвращения "зависания" программы в рабочем диапазоне напряжений питания. Если возможно понижение напряжения питания ниже рабочего уровня, то необходимо использовать внутреннюю или внешнюю схему сброса по снижению напряжения питания.

Вопрос 2: Микроконтроллер сбрасывается даже при регулярном выполнении команды CLRWDT.

Ответ 2:

Удостоверьтесь в том, что период выполнения команды CLRWDT меньше, чем минимальное время переполнения WDT (не номинальное значение).

Вопрос 3: Микроконтроллер не выходит из состояния сброса.

Ответ 3:

При включении питания необходимо учитывать время запуска тактового генератора (T_{OST}). Иногда эту проблему можно решить разметив в начале программы команду CLRWDT. Затем можно изменить время сброса WDT.

26.7 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с WDT и SLEEP режимом в микроконтроллерах PICmicro MCU:

Документ	Номер
Power-up Trouble Shooting Решение проблем, возникающих при включении питания	AN607

Раздел 27. Биты конфигурации

Содержание

27.1 Введение

Биты конфигурации позволяют настроить некоторые режимы работы микроконтроллера в соответствии с требованиями конкретного приложения. При включении питания состояние этих битов определяет режим работы микроконтроллера. Описание битов конфигурации смотрите в главе 27.2. Биты конфигурации расположены по адресу 2007h в памяти программ. Программа пользователя не может изменять и читать состояние битов конфигурации (эта операция возможна только в режиме программирования микроконтроллера).

Биты конфигурации могут быть запрограммированы (читаются как '0') или оставлены без изменения (читаются как '1'), чтобы выбрать режим работы микроконтроллера. Возможность изменения битов конфигурации после их программирования зависит от технологии изготовления памяти программ и типа корпуса микроконтроллера.

Для микроконтроллеров с масочной памятью (ROM) состояние битов конфигурации определяются во время передачи кода программы, и они не могут быть изменены (необходим новый код маски памяти).

В микроконтроллерах с однократно программируемой памятью (OTP), если бит конфигурации был запрограммирован ('0'), то он не может быть изменен.

Микроконтроллеры с УФ стиранием памяти программ позволяют возвратить биты конфигурации в начальное состояние. При стирании битов конфигурации также будет стерта память программ.

В микроконтроллерах с Flash памятью программ эти биты могут быть стерты и повторно запрограммированы.

Примечание. Не рекомендуется программировать биты защиты в микроконтроллерах с УФ стиранием памяти программ.

27.2 Слово конфигурации

В слове конфигурации расположены биты, управляющие некоторыми режимами работы микроконтроллера. Они могут быть изменены по протоколу программирования ICSP в программаторе или в устройстве пользователя. Размещение управляющих битов в слове конфигурации автоматически определяется при выборе программируемого микроконтроллера (это является функцией качественных программаторов). Дополнительную информацию о программировании микроконтроллеров смотрите в соответствующей спецификации программирования.

Примечание 1. Необходимо гарантировать, что в программаторе указан тип микроконтроллера, который Вы хотите запрограммировать.

Примечание 2. Рекомендуется состояние битов конфигурации указывать в исходном тексте программы. Это легко сделать в ассемблере MPASM с помощью директивы CONFIG (см. главу 27.2.1).

CP1:CP0: Биты защита памяти программ
11 = защита памяти программ выключена
10 = смотрите в технической документации на микроконтроллер
01 = смотрите в технической документации на микроконтроллер
00 = защищена вся память программ
Примечание. Некоторые микроконтроллеры содержат только один бит, управляющий защитой памяти программ.
1 = защита памяти программ выключена
0 = защищена вся память программ

DP: Бит защиты EEPROM памяти данных
1 = защита памяти данных выключена
0 = защита памяти данных включена
Примечание. Этот бит используется только в микроконтроллерах с EEPROM памятью данных.

BODEN: Бит разрешения сброса по снижению напряжения питания
1 = разрешен сброс BOR
0 = запрещен сброс BOR
Примечание. При включении схемы BOR таймер PWRT также включен независимо от состояния бита PWRTE. Необходимо разрешать работу таймера PWRT, если Вы используете сброс по снижению напряжения питания.

-PWRTE: Бит разрешения работы таймера включения питания
1 = PWRT выключен
0 = PWRT включен
Примечание. В некоторых микроконтроллерах бит -PWRT имеет обратную полярность.

MCLRE: Бит выбора режима работы вывода -MCLR
1 = вывод работает как -MCLR
0 = вывод работает как цифровой порт ввода/вывода, используется внутренний сброс -MCLR

WDTE: Бит разрешения работы сторожевого таймера
1 = WDT включен
0 = WDT выключен

FOSC1:FOSC0: Биты выбора режима тактового генератора
11 = RC генератор
10 = HS генератор
01 = XT генератор
00 = LP генератор

FOSC2:FOSC0: Биты выбора режима тактового генератора
111 = EXTRC внешний RC генератор с CLKOUT
110 = EXTRC внешний RC генератор
101 = INTRC внутренний RC генератор с CLKOUT
100 = INTRC внутренний RC генератор
011 = резерв
010 = HS генератор
001 = XT генератор
000 = LP генератор

Примечание. Расположение и состав битов конфигурации смотрите в технической документации на микроконтроллер. При использовании программаторов компании Microchip не требуется знания размещения управляющих битов в слове конфигурации.

27.2.1 Директива CONFIG ассемблера MPASM

В макроассемблере MPASM предоставляется возможность определить биты конфигурации в исходном тексте программы с помощью директивы CONFIG. Использование директивы CONFIG гарантирует запись битов конфигурации при программировании микроконтроллера, что уменьшает риск запрограммировать неправильно слово конфигурации.

В примере 27-1 представлен вариант использования директивы CONFIG.

Пример 27-1 Использование директивы CONFIG в шаблоне исходного файла

```
        LIST    p = p16C77                ; Директива LIST,
;
        #INCLUDE        <P16C77.INC>      ; Вспомогательный файл от Microchip
;
        #INCLUDE        <MY_STD.MAC>      ; Файл со стандартными макросами
        #INCLUDE        <APP.MAC>         ; Файл с набором макросов для данного приложения
;
; Настройка битов конфигурации
;
        __CONFIG _XT_OSC & _PWRTE_ON & _BODEN_OFF & _CP_OFF & _WDT_ON
;
        org     0x00            ; Начало памяти программ
RESET_ADDR :                    ; Первая команда после сброса



        end
```

Символы, описанные в дополнительном файле Microchip (.inc), позволяют напрямую использовать директиву CONFIG (см. таблицу 27-1). Набор символов, доступных для конкретного микроконтроллера, смотрите в соответствующем файле .inc.

Примечание. Правильный выбор микроконтроллера (в директивах LIST и INCLUDE) гарантирует правильную полярность всех битов конфигурации.

Таблица 27-1 Список стандартных символов для директивы __CONFIG

Назначение	Символ
Тактовый генератор	_RC_OSC
	_EXTRC_OSC
	_EXTRC_OSC_CLKOUT
	_EXTRC_OSC_NOCLKOUT
	_INTRC_OSC
	_INTRC_OSC_CLKOUT
	_INTRC_OSC_NOCLKOUT
	_LP_OSC
	_XT_OSC
	_HS_OSC
Сторожей таймер WDT	_WDT_ON
	_WDT_OFF
Таймер включения питания PWRT	_PWRTE_ON
	_PWRTE_OFF
Сброс по снижению напряжения питания	_BODEN_ON
	_BODEN_OFF
Режим работы вывода -MCLR	_MCLRE_ON
	_MCLRE_OFF
Защита кода программы	_CP_ALL
	_CP_ON
	_CP_75
	_CP_50
	_CP_OFF
Защита EEPROM памяти данных	_DP_ON
	_DP_OFF
Защита калибровочной информации	_CPC_ON
	_CPC_OFF

Примечание. Не все символы могут быть доступны в отдельно взятом микроконтроллере. Состав символов Вы можете узнать из дополнительного файла .inc.

27.3 Защита кода программы

Если защита кода программы (EEPROM памяти данных) не была включена, то память программ (EEPROM память данных) может быть прочитана для проверки программирования.

Примечание. Не рекомендуется программировать биты защиты в микроконтроллерах с УФ стиранием памяти программ.

27.3.1 Микроконтроллеры с масочной памятью (ROM)

Когда в микроконтроллере с ROM памятью программ реализована EEPROM память данных, в слове конфигурации может присутствовать дополнительный бит защиты EEPROM памяти данных. Бит защиты памяти программ передается как часть кода программы. Бит защиты EEPROM памяти данных выполнен по технологии EEPROM. После выполнения заключительных испытаний бит защиты EEPROM памяти данных будет иметь тоже состояние, что и бит защиты памяти программ. Защита EEPROM памяти данных выключена, когда выключена защита памяти программ.

Для приложений, в которых требуется предварительно запрограммировать EEPROM память данных перед выпуском изделия, необходимо полностью стереть EEPROM память данных. Последовательность стирания EEPROM памяти данных смотрите в спецификации программирования микроконтроллера. После записи данных в EEPROM память бит защиты EEPROM памяти данных может быть запрограммирован в нужное состояние.

27.4 Размещение идентификатора ID

Четыре ячейки памяти программ (2000h-2003h) предназначены для размещения идентификатора, которые могут использоваться для сохранения контрольной суммы или другой информации. Эти ячейки недоступны программе микроконтроллера, но могут быть прочитаны и изменены при программировании. Используются только 4 младших бита каждой ячейки.

27.5 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Я использую JW микроконтроллер для отладки программы, но микроконтроллер больше не программируется (чтение дает все '0'). Может неисправен микроконтроллер?

Ответ 1:

Быстрее всего Вы включили защиту памяти программ. Если это так, то JW микроконтроллер больше не пригоден для использования (см. главу 27.3).

Вопрос 2: При переходе с PIC16C74 на PIC16C74A моя программа больше не работает.

Ответ 2:

1. При повторной компиляции исходного файла Вы указали микроконтроллер PIC16C74A? Рекомендуется использовать директиву CONFIG со стандартными символами.
2. В программаторе Вы указали микроконтроллер PIC16C74A? Все биты конфигурации правильно настроены?

Вопрос 3: При стирании памяти - память программ стерта, а слово конфигурации еще нет.

Ответ 3:

Это соответствует техническим характеристикам. Помните, что не рекомендуется включать защиту памяти на микроконтроллерах с УФ стиранием памяти.

27.6 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с битами конфигурации в микроконтроллерах PICmicro MCU:

Документ	Номер
В настоящее время документы не подготовлены	

Раздел 28. Последовательный внутрисхемный интерфейс программирования (ICSP)

Содержание

28.1 Введение

Все микроконтроллеры PICmicro среднего семейства могут быть запрограммированы по последовательному интерфейсу ICSP после их установки на плату устройства. Для программирования требуется всего два сигнальных провода (синхронизация, данные) и три проводов питания (общий, напряжение питания микроконтроллера, напряжение программирования).

Внутрисхемное программирование ICSP позволяет упростить учет выпускаемых изделий. Устанавливая в устройство не запрограммированный микроконтроллер, Вы можете подготовить большое число устройств по данному проекту. Перед продажей партии устройств в микроконтроллеры загружается последняя версия программного обеспечения за короткое время. Эта методика позволяет исключить накопление на складе изделий со старой версией программы и быстро проводить обновление программного обеспечения.

Очень много людей думают, что использование ICSP для PICmicro с OTP память программ возможно только на сборочной линии, на которой микроконтроллер будет запрограммирован без возможности изменения кода программы. Однако существует метод с помощью которого OTP микроконтроллер может быть запрограммирован несколько раз в зависимости от объема кода программы (подобно EEPROM и FLASH микроконтроллерам). Эта методика будет описана далее по тексту раздела.

28.2 Перевод микроконтроллера в режим последовательного программирования

Микроконтроллер переходит в режим программирования/проверки, если выводы RB6 и RB7 удерживаются в низком логическом уровне, а на выводе -MCLR (V_{PP}) уровень сигнала изменяется от V_{IL} до V_{IHH} (см. спецификацию программирования), и присутствует напряжение питания V_{DD}. Вывод RB6 становится входом тактового сигнала программирования, а RB7 - входом/выходом данных программирования/проверки. Выводы RB6, RB7 имеют входной буфер с триггером Шмидта. Для передачи данных микроконтроллером на выводе RB7 реализован выходной КМОП буфер.

После перехода в режим программирования/проверки счетчик команд PC принимает значение 0000h. Для управления программированием/проверкой используются 6 - разрядные команды. Некоторые команды сопровождаются 14 - разрядными данными, передаваемые или принимаемые из микроконтроллера в зависимости от назначения команды (чтение или запись). Полную информацию по программированию микроконтроллеров смотрите в спецификации программирования соответствующего микроконтроллера.

В режиме последовательного программирования/проверки сторожевой таймер WDT выключен для предотвращения сброса микроконтроллера.

28.3 Схема включения

Принципиальная схема устройства должна быть разработана таким образом, чтобы она позволяла подключить все сигналы программирования к микроконтроллеру. На рисунке 28-1 показана базовая схема включения микроконтроллера для программирования по интерфейсу ICSP. Схема устройства должна учитывать следующие аспекты:

- Изоляция вывода -MCLR/V_{PP} от остальной схемы;
- Нагрузка на выводах RB6, RB7;
- Емкость на выводах V_{DD}, -MCLR/V_{PP}, RB6 и RB7;
- Минимальное и максимальное напряжение питания V_{DD};
- Тактовый генератор PICmicro;
- Разъем программирования.

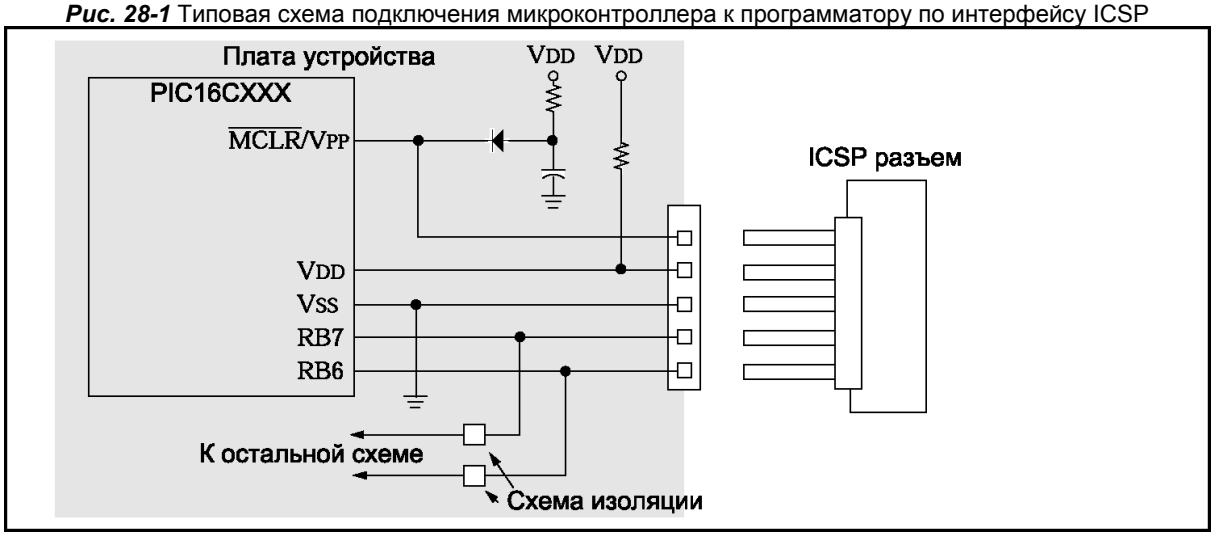
Вывод -MCLR/V_{PP} обычно соединен с RC цепочкой. Подтягивающий резистор подключен к V_{DD}, а конденсатор к общему проводу. RC цепочка может влиять на операцию ICSP в зависимости от емкости конденсатора, т.к. напряжение V_{PP} должно быть изолировано от остальной схемы (в большинстве случаев резистор не может изолировать остальную схему). Рекомендуется применять схему, показанную на рисунке 28-1, когда используется RC цепочка для вывода -MCLR/V_{PP}. Диод должен быть типа Schottky. Другой проблемой является то, что на выводе -MCLR/V_{PP} присутствует напряжение примерно 13В (схема устройства должна быть изолирована от этого напряжения).

Выводы RB6, RB7 используются PICmicro для последовательного программирования (RB6 - линия синхронизации; RB7 -линия данных). RB6 - управляется программатором. RB7 - двунаправленный вывод, который управляется программатором и микроконтроллером. Эти выводы должны быть изолированы от остальной схемы, чтобы она не влияла на сигналы последовательного программирования. Необходимо учитывать выходное сопротивление источника сигналов программатора при изоляции выводов RB6, RB7 от остальной схемы (RB6 - вход на PICmicro; RB7 - двунаправленный вывод). Например, программатор PRO MATE II имеет внутреннее сопротивление выводов 1кОм. Если проект позволяет, то эти выводы не должны использоваться. Проектировщик устройства должен сам оценить необходимость использования выводов RB6, RB7 и требования к буферизации сигналов. На рисунке 28-1 не показана схема изоляции выводов RB6, RB7, поскольку это зависит от конкретного приложения.

Чтобы упростить соединение микроконтроллера с программатором, выполните одно из следующих требований:

1. Не используйте выводы RB6, RB7 в своем приложении, оставив их свободными для ICSP.
2. К этим выводам должна быть подключена маломощная нагрузка.
3. Для выводов RB6, RB7 применять дополнительную схему изоляции, чтобы можно было подать сигналы ICSP на микроконтроллер.

Рис. 28-1 Типовая схема подключения микроконтроллера к программатору по интерфейсу ICSP



Емкостная нагрузка на выводах влияет на фронт сигнала, формируемого программатором. В типовых схемах на цепи питания V_{DD} устанавливаются конденсаторы емкостью несколько сот мкФ, которые позволяют подавить шум на цепи питания. Однако эта емкость требует достаточно мощного источника питания в программаторе, чтобы обеспечить требуемую скорость нарастания напряжения питания. Большинство программаторов разработаны для непосредственной установки микроконтроллера в колодку и не имеют достаточно мощного источника питания. Решением этой проблемы может быть использование дополнительного драйвера. Драйвер требует дополнительные источники питания, обеспечивающие ток для V_{PP} и V_{DD}. В данном случае сигналы на выводы RB6, RB7 не буферизируются, но буферизация может потребоваться в зависимости от приложения. Типовая схема платы драйвера показана на рисунке 28-2.

Примечание. Схема драйвера должна быть проверена на предмет выполнения требований временных характеристик сигналов синхронизации и данных. В случае подключения большой нагрузки к выводам V_{DD}, V_{PP}, RB6 и RB7 может потребоваться изменение схемы драйвера.

Согласно спецификации программирования - микроконтроллер может быть запрограммирован только при напряжении питания 5В. Необходимо учитывать этот факт, если ваша схема рассчитана на напряжение питания 3В. Возможным вариантом решения может быть полная изоляция микроконтроллера во время программирования. Дополнительно должна быть выполнена проверка программирования микроконтроллера при минимальном и максимальном напряжении питания устройства. Например, устройство питается от трех батареек 1.5В, рабочий диапазон напряжений питания от 2.7В до 4.5В. Программирование выполняется при напряжении питания 5В, проверка программирования должна быть выполнена при напряжениях питания 2.7В и 4.5В. Это будет гарантировать работу микроконтроллера во всем диапазоне напряжений питания для данного устройства.

Заключительной проблемой является работа тактового генератора при установленном микроконтроллере в устройстве пользователя. Напряжение на выводе -MCLR/V_{PP} должно достигнуть V_{PP} до начала выполнения кода программы. При использовании кварцевого или керамического резонатора это проблема не возникает, т.к. код программы начнет выполняться после отсчета 1024 тактов генератора. RC генераторы не требуют времени запуска, поэтому таймер запуска генератора не используется. Программатор должен успеть установить напряжение программирования на выводе -MCLR/V_{PP} прежде, чем тактовый генератор успеет сформировать 4 такта. Если RC генератор успеет сформировать 4 и более тактов, то после перехода в режим программирования счетчик команд будет иметь не нулевое значение, а некоторое значение X. Программа в микроконтроллер будет загружена со смещением. Существует несколько способов компенсировать медленное нарастание напряжение на -MCLR/V_{PP}. Первый способ - исключить резистор R при программировании микроконтроллера. Другой способ - возможность программатора управлять выводом OSC1, чтобы предотвратить генерацию тактового сигнала.

28.4 Программаторы

Другой аспект - программаторы. Все микроконтроллеры PIC16CXXX среднего семейства используют только последовательное программирование, поэтому все программаторы, поддерживающие эти микроконтроллеры, поддерживают последовательный интерфейс ICSP. Программатор должен иметь выход последовательного программирования, обеспечивать требуемую скорость нарастания напряжений и нагрузочную способность формируемых сигналов (этот вопрос был рассмотрен в предыдущей главе). На рисунке 28-2 представлен пример схемы драйвера. В этой схеме отсутствует буферизация сигналов для RB6 и RB7 (рекомендуется выполнить оценку схемы устройства для определения необходимости буферизации сигналов синхронизации и данных).

Другой особенностью программаторов является возможность проверки программирования памяти микроконтроллера при различных напряжениях питания V_{DD}. Программатор PRO MATE II поддерживает проверку программирования при минимальном и максимальном напряжении питания конкретного микроконтроллера, поэтому считается промышленным программатором. Программатор PICSTART Plus поддерживает только 5В режим программирования и проверки. Спецификации программирования микроконтроллеров требуют, чтобы программирование было проверено при минимальном и максимально возможном напряжении питания в устройстве. Это подразумевает, что схема устройства приспособлена для тестового изменения напряжения питания.

Существует большое число программаторов других производителей. Вы должны выбрать программатор на основе ваших требований под конкретную среду проектирования. Список инструментальных средств Microchip представлен в документации DS30177. Информацию о средствах проектирования других производителей Вы можете найти в документации DS00104. Воспользуйтесь этими документами при выборе программатора.

Выбирая программатор нужно учитывать большое число факторов: последовательный или параллельный интерфейс подключения к компьютеру, автономная работа, единовременно программируется одна или несколько микросхем.

28.5 Среда программирования

Среда программирования непосредственно влияет на тип программатора (длина кабеля программатора и интерфейс схемы устройства). Некоторые программаторы больше подходят для изготовления небольшой партии изделий при ручной сборке, другие - предназначены для автоматизированных сборочных линий. Возможно требуется одновременное программирование нескольких микросхем.

Длина кабеля от программатора до устройства влияет на емкостную нагрузку сигналов программирования, что непосредственно влияет на мощность управляющих сигналов, чтобы обеспечить требуемый ток. Кабель программирования должен быть максимально защищен от внешних помех.

Разъем программирования зависит от размеров устройства и параметров сборочной линии. В самом простом случае может использоваться специальный кабель для соединения программатора и устройства. Это больше подходит для ручной сборки, когда техник подключает кабель программатора к устройству. Существуют различные методы организации диагностических разъемов в устройствах. На этот разъем выводятся и линии программирования микроконтроллера. На этапе тестирования устройства может быть запрограммирован микроконтроллер, что наиболее оптимально для автоматизированной сборки изделий.

После учета всех требований к программированию микроконтроллеров любое предприятие может организовать программирование микроконтроллеров в готовом устройстве по интерфейсу ICSP.

28.6 Другие преимущества ICSP

ICSP позволят записывать калибровочную информацию и серийные номера изделий в память микроконтроллера. Сохранение калибровочной информации в памяти программ микроконтроллера является наиболее надежным и дешевым по сравнению с применением внешней последовательной EEPROM памяти. Например, если в вашем устройстве используется термистор, имеющий некоторый технологический разброс параметров, то, записав в память калибровочную информацию в виде таблицы, можно программным способом откорректировать работу датчика. Стоимость изделия может быть уменьшена за счет использования программных методов калибровки. Возникает вопрос, как это относится к ICSP? После программирования микроконтроллера проводят калибровочные испытания. Затем микроконтроллер вновь переводится в режим программирования и записывается подготовленная калибровочная информация. Дополнительную информацию по записи калибровочных данных смотрите в документации AN656.

ICSP предоставляет возможность сохранения индивидуального последовательного или случайного номера устройства. Серийный номер может использоваться в системах ограничения доступа как идентификационный ключ. В недорогих приложениях номер устройства устанавливается DIP переключателями. Запись уникального номера устройства в память программ микроконтроллера уменьшает общую стоимость изделия и риск несанкционированного доступа.

28.7 Программирование OTP микроконтроллеров PICmicro

OTP микроконтроллеры не могут быть перепрограммированы, но архитектура микроконтроллеров PICmicro предоставляет Вам эту возможность, если размер кода программы, по крайне мере, меньше в два раза объема памяти программ микроконтроллера, и защита памяти программ выключена. Если Ваш микроконтроллер не имеет требуемого объема памяти программ, то Вы можете выбрать другой с объемом памяти от 0.5кслов до 8кслов с тем же набором периферийных модулей, удовлетворяющий вашим требованиям.

PIC16CXXX имеют два вектора: вектор сброса (0x0000) и вектор прерываний (0x0004). Когда происходит сброс или генерируется прерывание, микроконтроллер начинает выполнять программу с соответствующего адреса. В левой части примера 28-2 показан код программы, первоначально загруженный в микроконтроллер, а в правой части примера - повторное программирование микроконтроллера.

В примере 28-2 показано, что при втором программировании по адресу 0x0000 вместо команды GOTO MAIN (0x2808) было записано 0x0000 (команда NOP). В эту ячейку нельзя записать код 0x2860 потому, что биты '0' не могут быть изменены к '1'. Только биты, значение которых '1', могут изменены на '0'. Следующая ячейка 0001h имеет все '1' поэтому в нее записано GOTO MAIN (0x2860). Когда происходит сброс микроконтроллера, сначала выполняется команда по адресу 0x000 (NOP), а затем - GOTO MAIN для начала выполнения основной программы. В примере также показано, что все ячейки памяти после 0x005A первоначально не запрограммированы, чтобы была возможность записать новый код после изменения программы. Аналогично можно поступать с вектором прерываний 0x0004.

Этот метод слегка изменяется для микроконтроллеров с объемом памяти более 2кслов, поскольку нужно переключать страницы памяти программ(см. пример 28-1).

Пример 28-1 Переход между страницами памяти программ

```
movlw          <page>                    movlw        <page>
movwf          PCLATH                    movwf        PCLATH
goto           Main                      goto         ISR
```

Теперь Ваш OTP микроконтроллер имеет схожие свойства с EEPROM или FLASH микроконтроллерами.

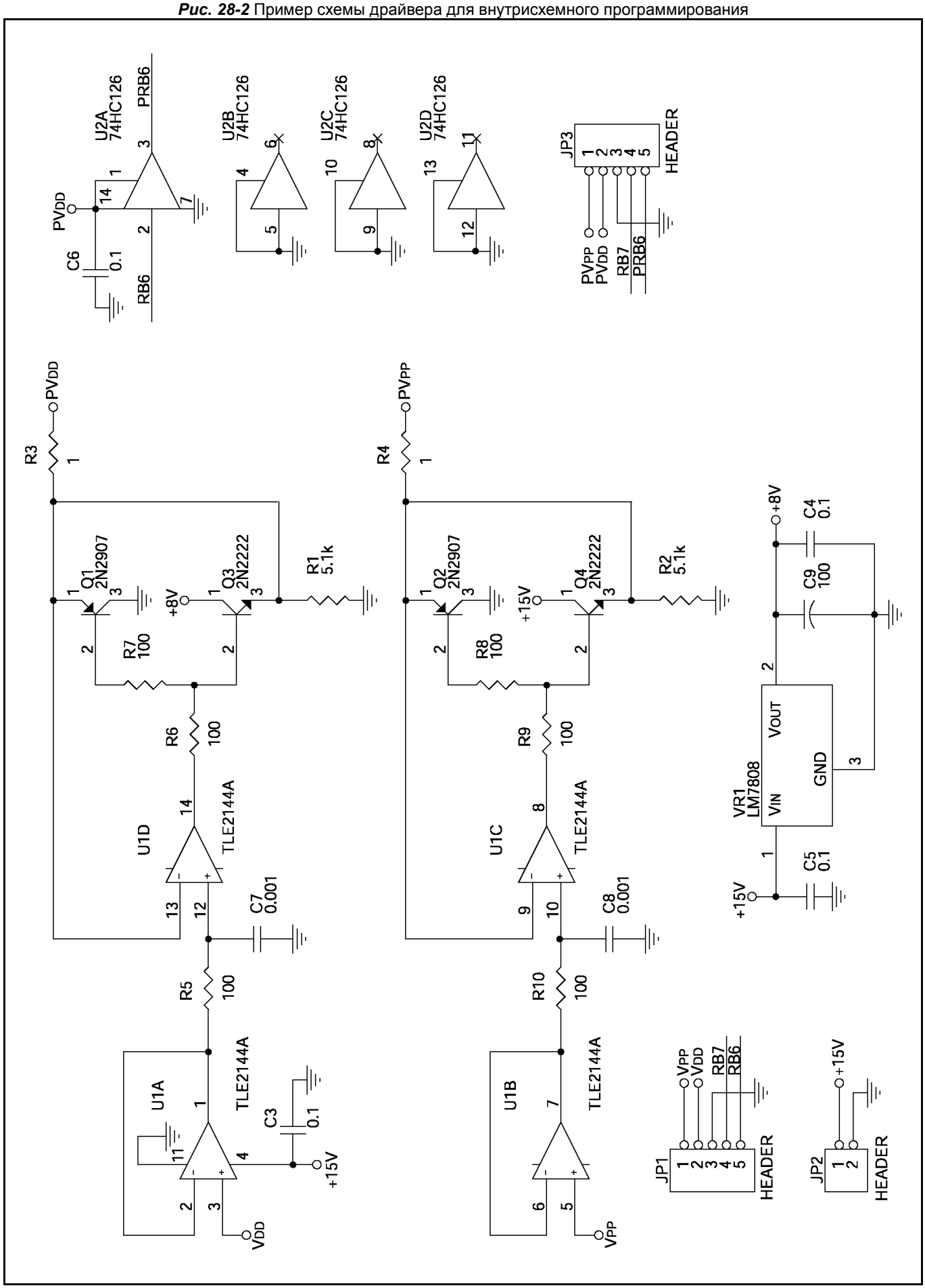
Пример 28-2 Циклы программирования OTP микроконтроллера

Первый цикл программирования			Второй цикл программирования		
Адрес	Код	Мнемоника	Адрес	Код	Мнемоника
0000	2808	goto Main ; Переход на	0000	0000	nop
0001	3FFF	<blank> ; 0x0008	0001	2860	goto Main ; Переход на
0002	3FFF	<blank>	0002	3FFF	<blank> ; 0x0060
0003	3FFF	<blank>	0003	3FFF	<blank>
0004	2848	goto ISR ; Переход на	0004	0000	nop
0005	3FFF	<blank> ; 0x0048	0005	28A8	goto ISR ; Переход на
0006	3FFF	<blank>	0006	3FFF	<blank> ; 0x00A8
0007	3FFF	<blank>	0007	3FFF	<blank>
0008	1683	bsf STATUS,RP0	0008	1683	bsf STATUS,RP0
0009	3007	movlw 0x07	0009	3007	movlw 0x07
000A	009F	movwf ADCON1	000A	009F	movwf ADCON1
.			.		
.			.		
.			.		
0048	1C0C	btfss PIR1,RBIF	0048	1C0C	btfss PIR1,RBIF
0049	284E	goto EndISR	0049	284E	goto EndISR
004A	1806	btfsc PORTB,0	004A	1806	btfsc PORTB,0
.			.		
.			.		
.			.		
0060	3FFF	<blank>	0060	1683	bsf STATUS,RP0
0061	3FFF	<blank>	0061	3005	movlw 0x05
0062	3FFF	<blank>	0062	009F	movwf ADCON1
.			.		
.			.		
.			.		
00A8	3FFF	<blank>	00A8	1C0C	btfss PIR1,RBIF
00A9	3FFF	<blank>	00A9	28AE	goto EndISR
00AA	3FFF	<blank>	00AA	1806	btfsc PORTB,0
.			.		
.			.		
.			.		

28.8 Программирование Flash микроконтроллеров PICmicro

Микроконтроллеры с Flash памятью программ могут быть перепрограммированы в устройстве пользователя даже, если включена защита памяти. Портативный программатор может состоять из персонального компьютера и собственно программатора. Техник подключает ICSP кабель к устройству и загружает новую программу в микроконтроллер PICmicro. С помощью Flash микроконтроллеров легко выполнять отладку программы или обновление программного обеспечения с добавлением новых функций без необходимости демонтажа устройства и доставки его в лабораторию.

Рис. 28-2 Пример схемы драйвера для внутрисхемного программирования



28.9 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Когда я выполняю программирование ICSP, программа смещена в памяти.

Ответ 1:

Если напряжение на выводе -MCLR не увеличивается достаточно быстро при нормальном напряжении питания, то внутренний счетчик команд PC может увеличиваться. Это означает, что счетчик команд не будет указывать на адрес, который Вы ожидаете. Значение счетчика команд зависит от числа тактовых импульсов, которые были сформированы при номинальном напряжении питания.

Вопрос 2: Я использую программатор PRO MATE II с самодельным разъемом для передачи сигналов программирования в мое устройство. Иногда память программ неправильно программируется.

Ответ 2:

Может быть нарушено напряжение/синхронизация сигналов из-за:

- Схемы устройства, к которому подключаете программатор;
- Большой длины соединительного кабеля;
- Большой емкости на цепи питания V_{DD}.

28.10 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило, примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с последовательным интерфейсом программирования ICSP в микроконтроллерах PICmicro MCU:

Документ	Номер
In-Circuit Serial Programming of Calibration Parameters using a PICmicro Программирование калибровочной информации по последовательному интерфейсу ICSP	AN656
In-Circuit Serial Programming Guide Руководства последовательного программирования	DS30277

Раздел 29. Система команд

Содержание

29.1 Введение

Каждая команда состоит из одного 14 - разрядного слова, разделенного на код операции (OPCODE), определяющий тип команды и один или несколько операндов, определяющие операцию команды. Полный список команд смотрите в таблице 29-1.

Система команд аккумуляторного типа, ортогональна и разделена на три основных группы:

- Байт ориентированные команды;
- Бит ориентированные команды;
- Команды управления и операций с константами.

Описание полей кода операции смотрите в таблице 29-2.

Для байт ориентированных команд 'f' является указателем регистра, а 'd' указателем адресата результата. Указатель регистра определяет, какой регистр должен использоваться в команде. Указатель адресата определяет, где будет сохранен результат. Если 'd'=0, результат сохраняется в регистре W. Если 'd'=1, результат сохраняется в регистре, который используется в команде.

В бит ориентированных командах 'b' определяет номер бита участвующего в операции, а 'f' - указатель регистра, который содержит этот бит.

В командах управления или операциях с константами 'k' представляет восемь или одиннадцать бит константы или значения литералов.

Все команды выполняются за один машинный цикл, кроме команд условия, в которых получен истинный результат и инструкций изменяющих значение счетчика команд PC. В случае выполнения команды за два машинных цикла, во втором цикле выполняется инструкция NOP. Один машинный цикл состоит из четырех тактов генератора. Для тактового генератора с частотой 4 МГц все команды выполняются за 1мкс, если условие истинно или изменяется счетчик команд PC, команда выполняется за 2мкс.

Таблица 29-1 Список команд микроконтроллеров среднего семейства

Мнемоника команды		Описание	Циклов	14-разрядный код		Изм. флаги	Прим.
				Бит 13	Бит 0		
Байт ориентированные команды							
ADDWF	**f,d**	Сложение W и f	1	00 0111 dfff ffff		C,DC,Z	1,2
ANDWF	**f,d**	Побитное 'И' W и f	1	00 0101 dfff ffff		Z	1,2
CLRF	**f**	Очистить f	1	00 0001 1fff ffff		Z	2
CLRW	**-**	Очистить W	1	00 0001 0xxx xxxx		Z	
COMF	**f,d**	Инвертировать f	1	00 1001 dfff ffff		Z	1,2
DECF	**f,d**	Вычесть 1 из f	1	00 0011 dfff ffff		Z	1,2
DECFSZ	**f,d**	Вычесть 1 из f и пропустить если 0	1(2)	00 1011 dfff ffff			1,2,3
INCF	**f,d**	Прибавить 1 к f	1	00 1010 dfff ffff		Z	1,2
INCFSZ	**f,d**	Прибавить 1 к f и пропустить если 0	1(2)	00 1111 dfff ffff			1,2,3
IORWF	**f,d**	Побитное 'ИЛИ' W и f	1	00 0100 dfff ffff		Z	1,2
MOVF	**f,d**	Переслать f	1	00 1000 dfff ffff		Z	1,2
MOVWF	**f**	Переслать W в f	1	00 0000 1fff ffff			
NOP	**-**	Нет операции	1	00 0000 0xx0 0000			
RLF	**f,d**	Циклический сдвиг f влево через перенос	1	00 1101 dfff ffff		C	1,2
RRF	**f,d**	Циклический сдвиг f вправо через перенос	1	00 1100 dfff ffff		C	1,2
SUBWF	**f,d**	Вычесть W из f	1	00 0010 dfff ffff		C,DC,Z	1,2
SWAPF	**f,d**	Поменять местами полубайты в регистре f	1	00 1110 dfff ffff			1,2
XORWF	**f,d**	Побитное 'исключающее ИЛИ' W и f	1	00 0110 dfff ffff		Z	1,2
Бит ориентированные команды							
BCF	**f,b**	Очистить бит b в регистре f	1	01 00bb bfff ffff			1,2
BSF	**f,b**	Установить бит b в регистре f	1	01 01bb bfff ffff			1,2
BTFSC	**f,b**	Проверить бит b в регистре f, пропустить если 0	1(2)	01 10bb bfff ffff			3
BTFSS	**f,b**	Проверить бит b в регистре f, пропустить если 1	1(2)	01 11bb bfff ffff			3
Команды управления и операций с константами							
ADDLW	**k**	Сложить константу с W	1	11 111x kkkk kkkk		C,DC,Z	
ANDLW	**k**	Побитное 'И' константы и W	1	11 1001 kkkk kkkk		Z	
CALL	**k**	Вызов подпрограммы	2	10 0kkk kkkk kkkk			
CLRWDT	**-**	Очистить WDT	1	00 0000 0110 0100		-TO,-PD	
GOTO	**k**	Безусловный переход	2	10 1kkk kkkk kkkk			
IORLW	**k**	Побитное 'ИЛИ' константы и W	1	11 1000 kkkk kkkk		Z	
MOVLW	**k**	Переслать константу в W	1	11 00xx kkkk kkkk			
RETFIE	**-**	Возврат из подпрограммы с разрешением прерываний	2	00 0000 0000 1001			
RETLW	**k**	Возврат из подпрограммы с загрузкой константы в W	2	11 01xx kkkk kkkk			
RETURN	**-**	Возврат из подпрограммы	2	00 0000 0000 1000			
SLEEP	**-**	Перейти в режим SLEEP	1	00 0000 0110 0011		-TO,-PD	
SUBLW	**k**	Вычесть W из константы	1	11 110x kkkk kkkk		C,DC,Z	
XORLW	**k**	Побитное 'исключающее ИЛИ' константы и W	1	11 1010 kkkk kkkk		Z	

Примечания:

1. При выполнении операции "чтение - модификация - запись" с портом ввода/вывода (например MOVF PORTB,1) исходные значения считываются с выводов порта, а не из выходных защелок. Например, если в выходной защелке было записана '1', а на соответствующем выходе низкий уровень сигнала, то обратно будет записано значение '0'.
2. При выполнении записи в TMR0 (и d=1) предделитель TMR0 сбрасывается, если он подключен к модулю TMR0.
3. Если условие истинно или изменяется значение счетчика команд PC, то инструкция выполняется за два цикла. Во втором цикле выполняется команда NOP.

29.2 Формат команд

На рисунке 29-1 показано формат трех групп команд. Код команды может быть от 3 до 6 бит, что позволяет реализовать 35 команд.

Примечание 1. Любой не реализованный код операции сохранен для последующих разработок. Использование недокументированного кода операции может привести к непредсказуемым последствиям.

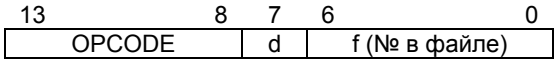
Примечание 2. Для совместимости программного обеспечения со следующими версиями микроконтроллеров PICmicro не используйте команды TRIS и OPTION.

Во всех примерах используется следующий формат шестнадцатеричных чисел:
0xhh, где h - шестнадцатеричная цифра.

Представление двоичного числа:
00000100b, где b - указатель двоичного числа.

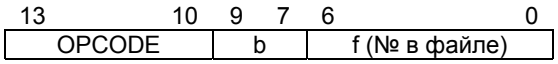
Рис. 29-1 Общий формат команд микроконтроллеров среднего семейства

Байт ориентированные операции с регистрами



d = 0 - результат сохраняется в w
d = 1 - результат сохраняется в f
f - 7-разрядный адрес регистра

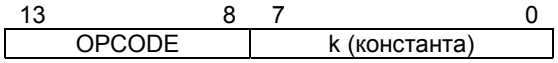
Бит ориентированные операции с регистрами



b - 3-разрядный номер бита в регистре
f - 7-разрядный адрес регистра

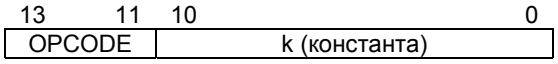
Команды управления и операций с константами

Общее



k - 8-разрядное значение

Только для инструкций CALL и GOTO



k - 11-разрядное значение

Таблица 29-2 Описание полей кода операции

Поле	Описание
f	Адрес регистра (от 0x00 до 0x7F)
w	Рабочий регистр (аккумулятор)
b	Номер бита в 8-разрядном регистре
k	Константа (данные или метка)
x	Не имеет значения (0 или 1). Ассемблер генерирует x=0 для совместимости программы микроконтроллера с инструментальными средствами
d	Указатель адресата результата операции: d = 0 - результат сохраняется в регистре w d = 1 - результат сохраняется в регистре f По умолчанию d = 1
label	Имя метки
TOS	Вершина стека
PC	Счетчик команд
PCLATH	Буфер старшего байта счетчика команд
GIE	Бит глобального разрешения прерываний
WDT	Сторожевой таймер
-TO	Флаг переполнения WDT
-PD	Флаг сброса по включению питания
dest	Приемник, регистр w или регистр памяти
[ ]	Дополнительные параметры
( )	Содержимое
$\rightarrow$	Присвоение
< >	Битовое поле
$\in$	Из набора
Курсив	Термин, определяемый пользователем

29.3 Обращение к регистрам специального назначения

Система команд микроконтроллеров PICmicro среднего семейства позволяет напрямую обращаться ко всем регистрам, включая регистры общего назначения. Существуют нюансы обращения к некоторым регистрам, которые должен учитывать разработчик программного обеспечения.

29.3.1 STATUS как регистр назначения при выполнении команды

Если обращение к регистру STATUS выполняется командой, которая воздействует на флаги Z, DC и C, то изменение этих трех битов командой заблокировано. Эти биты сбрасываются или устанавливаются согласно логике ядра микроконтроллера. Команды изменения регистра STATUS также не воздействуют на биты -TO и -PD. Поэтому результат выполнения команды с регистром STATUS может отличаться от ожидаемого. Например, команда CLRF STATUS сбросит три старших бита и установит бит Z (состояние регистра STATUS после выполнения команды 000uu1uu, где u - не изменяемый бит).

29.3.2 PCL как источник данных и регистр назначения при выполнении команды

Команды чтения, записи и "чтение - модификация - запись" регистра PCL могут иметь следующие результаты:

Чтение PCL:	PCL $\rightarrow$ dest; PCLATH не изменяется.
Запись PCL:	PCLATH $\rightarrow$ PCH; 8 - разрядное значение $\rightarrow$ PCL.
"Чтение - модификация - запись":	PCL $\rightarrow$ операнд АЛУ; PCLATH $\rightarrow$ PCH; 8 - разрядный результат $\rightarrow$ PCL.

Где PCH - старший байт счетчика команд (не адресуемый регистр), PCLATH - буфер старшего байта счетчика команд, dest - регистр назначения.

29.3.3 Битовые операции

При изменении любого бита регистра сначала регистр полностью читается из памяти данных, изменяется бит, обратно данные записываются в регистр ("чтение - модификация - запись"). Необходимо учитывать этот факт при обращении к некоторым регистрам специального назначения (например, регистры портов).

Примечание. Изменение состояния управляющих битов (включая флаги прерываний) выполняется в такте Q1, поэтому не возникает проблем при обращении командой "чтение - модификация - запись" к регистрам, содержащим эти биты.

29.4 Такты выполнения команд

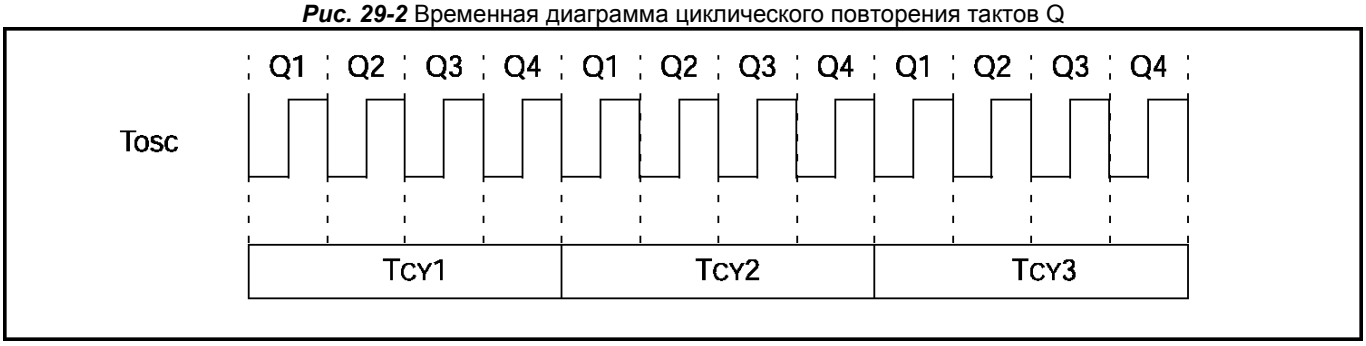
Каждый цикл команды (T_{CY}) состоит из четырех тактов (Q1-Q4). Такт Q равен по длительности периоду тактового генератора (T_{OSC}). Такты Q обеспечивают жесткую синхронизацию декодирования, чтения данных, обработки данных, записи результата для каждого цикла команды. На диаграмме показано соотношение тактов Q к циклу команды.

Цикл команды (T_{CY}), состоящий из 4-х тактов, обобщенно выглядит следующим образом:

Q1: Детектирование команды или принудительной пустой операции (NOP)
Q2: Операция чтения данных или отсутствие операции
Q3: Обработка данных
Q4: Операция записи данных или отсутствие операции

Детальный состав операций команды по тактам Q смотрите в описании команды.

Рис. 29-2 Временная диаграмма циклического повторения тактов Q



29.5 Описание команд

ADDLW Сложить константу с W

Синтаксис: [*label*] ADDLW k

Операнды: $0 \le k \le 255$

Операция: $(W) + k \rightarrow (W)$

Измен. флаги: C, DC, Z

Код:

11	111x	kkkk	kkkk

Описание: Содержимое регистра W складывается с 8 - разрядной константой 'k'. Результат сохраняется в регистре W.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы 'k'	Выполнение	Запись в регистр W

Пример 1: ADDLW 0x15

До выполнения команды

W = 0x10

После выполнения команды

W = 0x25

Пример 2: ADDLW MYREG

До выполнения команды

W = 0x10
MYREG = 0x37 (адрес регистра)

После выполнения команды

W = 0x47

Пример 3: ADDLW HIGH (LU_TABLE)

До выполнения команды

W = 0x10
LU_TABLE = 0x9375 (адрес в памяти программ)

После выполнения команды

W = 0xA3

ADDWF Сложение W и f

Синтаксис: [*label*] ADDWF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (W) + (f) → (dest)

Измен. флаги: C, DC, Z

Код:

00	0111	dfff	ffff

Описание: Сложить содержимое регистров W и 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1:
```
ADDWF   FSR,0
До выполнения команды
        W = 0x17
        FSR = 0xC2
После выполнения команды
        W = 0xD9
        FSR = 0xC2
```

Пример 2:
```
ADDWF   INDF,1
До выполнения команды
        W = 0x17
        FSR = 0xC2
        Значение регистра с адресом в FSR = 0x20
После выполнения команды
        W = 0x17
        FSR = 0xC2
        Значение регистра с адресом в FSR = 0x37
```

Пример 3:
```
ADDWF   PCL,0
Случай 1
До выполнения команды
        W = 0x10
        PCL = 0x37
        C = x
После выполнения команды
        PCL = 0x47
        C = 0
Случай 2
До выполнения команды
        W = 0x10
        PCL = 0xF7
        PCH = 0x08
        C = x
После выполнения команды
        PCL = 0x07
        PCH = 0x08
        C = 1
```

ANDLW Побитное 'И' константы и W

Синтаксис: [*label*] ANDLW k

Операнды: $0 \le k \le 255$

Операция: (W) .AND. k → (W)

Измен. флаги: Z

Код:

11	1001	kkkk	kkkk

Описание: Выполняется побитное 'И' содержимого регистра W и 8 - разрядной константы 'k'. Результат сохраняется в регистре W.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы 'k'	Выполнение	Запись в регистр W

Пример 1: ANDLW 0x5F (0101 1111)

До выполнения команды

W = 0xA3 (1010 0011)

После выполнения команды

W = 0x03 (0000 0011)

Пример 2: ANDLW MYREG

До выполнения команды

W = 0xA3
MYREG = 0x37 (адрес регистра)

После выполнения команды

W = 0x23

Пример 3: ANDLW HIGH (LU_TABLE)

До выполнения команды

W = 0xA3
LU_TABLE = 0x9375 (адрес в памяти программ)

После выполнения команды

W = 0x83

ANDWF Побитное 'И' W и f

Синтаксис: [*label*] ANDWF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (W) .AND. (f) → (dest)

Измен. флаги: Z

Код:

00	0101	dfff	ffff

Описание: Выполняется побитное 'И' содержимого регистров W и 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1:
```
ANDWF   FSR,1
```
До выполнения команды
W = 0x17 (0001 0111)
FSR = 0xC2 (1100 0010)
После выполнения команды
W = 0x17
FSR = 0x02 (0000 0010)

Пример 2:
```
ANDWF   FSR,0
```
До выполнения команды
W = 0x17 (0001 0111)
FSR = 0xC2 (1100 0010)
После выполнения команды
W = 0x02 (0000 0010)
FSR = 0xC2

Пример 3:
```
ANDWF   INDF,1
```
До выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x5A
После выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x15

BCF Очистить бит b в регистре f

Синтаксис: [*label*] BCF f,b

Операнды: $0 \le f \le 127$
$0 \le b \le 7$

Операция: $0 \rightarrow (f<b>)$

Измен. флаги: Нет

Код:

01	00bb	bfff	ffff

Описание: Очистить бит 'b' в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись в регистр 'f'

Пример 1:

```
BCF     FLAG_REG,7
```

До выполнения команды

FLAG_REG = 0xC7 (**1**100 0111)

После выполнения команды

FLAG_REG = 0x47 (**0**100 0111)

Пример 2:

```
BCF     INDF,3
```

До выполнения команды

W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x2F

После выполнения команды

W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x27

BSF Установить бит b в регистре f

Синтаксис: [*label*] BSF f,b

Операнды: $0 \le f \le 127$
$0 \le b \le 7$

Операция: $1 \rightarrow (f<b>)$

Измен. флаги: Нет

Код:

01	01bb	bfff	ffff

Описание: Установить бит 'b' в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись в регистр 'f'

Пример 1:

```
BSF     FLAG_REG,7
```

До выполнения команды

FLAG_REG = 0x0A (**0**000 1010)

После выполнения команды

FLAG_REG = 0x8A (**1**000 1010)

Пример 2:

```
BSF     INDF,3
```

До выполнения команды

W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x20

После выполнения команды

W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x28

BTFSC — Проверить бит b в регистре f, пропустить если 0

Синтаксис: [*label*] BTFSC f,b

Операнды: 0 ≤ f ≤ 127
0 ≤ b ≤ 7

Операция: пропустить если (f<b>) = 0

Измен. флаги: Нет

Код:

01	10bb	bfff	ffff

Описание: Если бит 'b' в регистре 'f' равен '1' , то исполняется следующая инструкция.
Если бит 'b' в регистре 'f' равен '0' , то следующая инструкция не выполняется, команда выполняется за два цикла. Во втором цикле выполняется NOP.

Слов: 1

Циклов: 1(2)

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Нет операции

Если пропустить (2 цикла)

Q1	Q2	Q3	Q4
Нет операции	Нет операции	Нет операции	Нет операции

Пример 1:

```
HERE    BTFSC   FLAG,4
FALSE   GOTO    PROCESS_CODE
TRUE    •
        •
```

Случай 1 До выполнения команды
PC = адрес HERE
FLAG = xxx0 xxxx
После выполнения команды
Т.к. FLAG<4> = 0,
PC = адрес TRUE

Случай 2 До выполнения команды
PC = адрес HERE
FLAG = xxx1 xxxx
После выполнения команды
Т.к. FLAG<4> = 1,
PC = адрес FALSE

BTFSS Проверить бит b в регистре f, пропустить если 1

Синтаксис: [*label*] BTFSS f,b

Операнды: 0 ≤ f ≤ 127
0 ≤ b ≤ 7

Операция: пропустить если (f<b>) = 1

Измен. флаги: Нет

Код:

01	11bb	bfff	ffff

Описание: Если бит 'b' в регистре 'f' равен '0' , то исполняется следующая инструкция.
Если бит 'b' в регистре 'f' равен '1' , то следующая инструкция не выполняется, команда выполняется за два цикла. Во втором цикле выполняется NOP.

Слов: 1

Циклов: 1(2)

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Нет операции

Если пропустить (2 цикла)

Q1	Q2	Q3	Q4
Нет операции	Нет операции	Нет операции	Нет операции

Пример 1:

```
HERE    BTFSS   FLAG,4
FALSE   GOTO    PROCESS_CODE
TRUE    •
        •
```

Случай 1
До выполнения команды
PC = адрес HERE
FLAG = xxx0 xxxx
После выполнения команды
T.к. FLAG<4> = 0,
PC = адрес FALSE

Случай 2
До выполнения команды
PC = адрес HERE
FLAG = xxx1 xxxx
После выполнения команды
T.к. FLAG<4> = 1,
PC = адрес TRUE

CALL Вызов подпрограммы

Синтаксис: [*label*] CALL k

Операнды: $0 \le k \le 2047$

Операция: (PC) + 1 → TOS,
k → PC<10:0>,
(PCLATH<4:3>) → PC<12:11>

Измен. флаги: Нет

Код:

10	0kkk	kkkk	Kkkk

Описание: Вызов подпрограммы. Адрес следующей инструкции (PC+1) помещается в вершину стека. Одиннадцать бит адреса загружаются из кода команды в счетчик команд PC<10:0>. Два старших бита загружаются в счетчик команд PC<12:11> из регистра PCLATH. Команда CALL выполняется за два цикла.

Слов: 1

Циклов: 2

Выполнение команды по тактам

1-й цикл

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы 'k'	Выполнение	Нет операции

2-й цикл

Q1	Q2	Q3	Q4
Нет операции	Нет операции	Нет операции	Нет операции

Пример 1: HERE CALL THERE

До выполнения команды
PC = адрес HERE

После выполнения команды
PC = адрес THERE
TOS = адрес HERE + 1

CLRF Очистить f

Синтаксис: [*label*] CLRF f

Операнды: $0 \le f \le 127$

Операция: $00h \rightarrow (f)$
$1 \rightarrow Z$

Измен. флаги: Z

Код:

00	0001	1fff	ffff

Описание: Очистить содержимое регистра 'f' и установить флаг Z

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Выполнение	Запись в регистр 'f'

Пример 1:

```
CLRF    FLAG_REG
До выполнения команды
        FLAG_REG = 0x5A
После выполнения команды
        FLAG_REG = 0x00
        Z = 1
```

Пример 2:

```
CLRF    INDF
До выполнения команды
        FSR = 0xC2
        Значение регистра с адресом в FSR = 0xAA
После выполнения команды
        FSR = 0xC2
        Значение регистра с адресом в FSR = 0x00
        Z = 1
```

CLRW Очистить W

Синтаксис: [*label*] CLRW

Операнды: Нет

Операция: 00h → (W)
1 → Z

Измен. флаги: Z

Код:

00	0001	0xxx	xxxx

Описание: Очистить содержимое регистра W и установить флаг Z

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Выполнение	Запись в регистр W

Пример 1: CLRW

До выполнения команды
W = 0x5A

После выполнения команды
W = 0x00
Z = 1

CLRWDT Очистить WDT

Синтаксис: [*label*] CLRWDT

Операнды: Нет

Операция: 00h → WDT,
00h → предделитель WDT,
1 → -TO
1 → -PD

Измен. флаги: -TO, -PD

Код:

00	0000	0110	0100

Описание: Инструкция CLRWDT сбрасывает WDT и предделитель, если он подключен к WDT. В регистре STATUS устанавливает биты -TO и -PD.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Выполнение	Сбросить WDT

Пример 1: CLRWDT

До выполнения команды
Счетчик WDT = ?
Делитель WDT = 1:128

После выполнения команды
Счетчик WDT = 0
Счетчик делителя WDT = 0
-TO = 1
- PD = 1
Делитель WDT = 1:128

Примечание. Команда CLRWDT не влияет на коэффициент делителя WDT.

COMF Инвертировать f

Синтаксис: [*label*] COMF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (-f) → (dest)

Измен. флаги: Z

Код:

00	1101	dfff	ffff

Описание: Инвертировать все биты в регистре 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1:
```
COMF    REG1,0
```
До выполнения команды
```
        REG1 = 0x13
```
После выполнения команды
```
        REG1 = 0x13
        W = 0xEC
```

Пример 2:
```
COMF    INDF,1
```
До выполнения команды
```
        FSR = 0xC2
        Значение регистра с адресом в FSR = 0xAA
```
После выполнения команды
```
        FSR = 0xC2
        Значение регистра с адресом в FSR = 0x55
```

Пример 3:
```
COMF    REG1,1
```
До выполнения команды
```
        REG1 = 0xFF
```
После выполнения команды
```
        REG1 = 0x00
        Z = 1
```

DECF Вычесть 1 из f

Синтаксис: [*label*] DECF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (f) - 1 → (dest)

Измен. флаги: Z

Код:

00	0011	dfff	ffff

Описание: Декрементировать содержимое регистра 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1:
```
DECF    CNT,1
```
До выполнения команды
CNT = 0x01
Z = 0
После выполнения команды
CNT = 0x00
Z = 1

Пример 2:
```
DECF    INDF,1
```
До выполнения команды
FSR = 0xC2
Значение регистра с адресом в FSR = 0x01
Z = 0
После выполнения команды
FSR = 0xC2
Значение регистра с адресом в FSR = 0x00
Z = 1

Пример 3:
```
DECF    CNT,0
```
До выполнения команды
CNT = 0x10
W = x
Z = 0
После выполнения команды
CNT = 0x10
W = 0x0F
Z = 0

DECFSZ Вычесть 1 из f и пропустить если 0

Синтаксис: [*label*] DECFSZ f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (f) - 1 → (dest); пропустить если результат равен 0

Измен. флаги: Нет

Код:

00	1011	dfff	ffff

Описание: Декрементировать содержимое регистра 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'. Если результат не равен '0' , то исполняется следующая инструкция. Если результат равен '0' , то следующая инструкция не выполняется, команда выполняется за два цикла. Во втором цикле выполняется NOP.

Слов: 1

Циклов: 1(2)

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Если пропустить (2 цикла)

Q1	Q2	Q3	Q4
Нет операции	Нет операции	Нет операции	Нет операции

Пример 1:

```
HERE      DECFSZ   CNT,1
          GOTO     LOOP
CONTINUE  •
          •
```

Случай 1
До выполнения команды
PC = адрес HERE
CNT = 0x01
После выполнения команды
CNT = 0x00
PC = адрес CONTINUE

Случай 2
До выполнения команды
PC = адрес HERE
CNT = 0x02
После выполнения команды
CNT = 0x01
PC = адрес HERE + 1

GOTO Безусловный переход

Синтаксис: [*label*] GOTO k

Операнды: $0 \le k \le 2047$

Операция: $k \rightarrow PC<10:0>$,
$(PCLATH<4:3>) \rightarrow PC<12:11>$

Измен. флаги: Нет

Код:

10	1kkk	kkkk	kkkk

Описание: Выполнить безусловный переход. Одиннадцать бит адреса загружаются из кода команды в счетчик команд PC<10:0>. Два старших бита загружаются в счетчик команд PC<12:11> из регистра PCLATH. Команда GOTO выполняется за два цикла.

Слов: 1

Циклов: 2

Выполнение команды по тактам

1-й цикл

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы 'k'	Выполнение	Нет операции

2-й цикл

Q1	Q2	Q3	Q4
Нет операции	Нет операции	Нет операции	Нет операции

Пример 1:

```
GOTO    THERE
```

После выполнения команды

PC = адрес THERE

INCF Прибавить 1 к f

Синтаксис: [*label*] INCF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (f) + 1 → (dest)

Измен. флаги: Z

Код:

00	1010	dfff	ffff

Описание: Инкрементировать содержимое регистра 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1:
```
INCF    CNT,1
```
До выполнения команды
CNT = 0xFF
Z = 0
После выполнения команды
CNT = 0x00
Z = 1

Пример 2:
```
INCF    INDF,1
```
До выполнения команды
FSR = 0xC2
Значение регистра с адресом в FSR = 0xFF
Z = 0
После выполнения команды
FSR = 0xC2
Значение регистра с адресом в FSR = 0x00
Z = 1

Пример 3:
```
INCF    CNT,0
```
До выполнения команды
CNT = 0x10
W = x
Z = 0
После выполнения команды
CNT = 0x10
W = 0x11
Z = 0

INCFSZ Прибавить 1 к f и пропустить если 0

Синтаксис: [*label*] INCFSZ f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (f) + 1 → (dest); пропустить если результат равен 0

Измен. флаги: Нет

Код:

00	1111	dfff	ffff

Описание: Инкрементировать содержимое регистра 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'. Если результат не равен '0' , то исполняется следующая инструкция. Если результат равен '0' , то следующая инструкция не выполняется, команда выполняется за два цикла. Во втором цикле выполняется NOP.

Слов: 1

Циклов: 1(2)

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Если пропустить (2 цикла)

Q1	Q2	Q3	Q4
Нет операции	Нет операции	Нет операции	Нет операции

Пример 1:

```
HERE      INCFSZ   CNT,1
          GOTO     LOOP
CONTINUE  •
          •
```

Случай 1
До выполнения команды
PC = адрес HERE
CNT = 0xFF
После выполнения команды
CNT = 0x00
PC = адрес CONTINUE

Случай 2
До выполнения команды
PC = адрес HERE
CNT = 0x00
После выполнения команды
CNT = 0x01
PC = адрес HERE + 1

IORLW Побитное 'ИЛИ' константы и W

Синтаксис: [*label*] IORLW k

Операнды: $0 \le k \le 255$

Операция: (W) .OR. k → (W)

Измен. флаги: Z

Код:

11	1000	kkkk	kkkk

Описание: Выполняется побитное 'ИЛИ' содержимого регистра W и 8-разрядной константы 'k'. Результат сохраняется в регистре W.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы 'k'	Выполнение	Запись в регистр W

Пример 1:

```
IORLW   0x35
```

До выполнения команды
W = 0x9A
После выполнения команды
W = 0xBF
Z = 0

Пример 2:

```
IORLW   MYREG
```

До выполнения команды
W = 0x9A
MYREG = 0x37 (адрес регистра)
После выполнения команды
W = 0x9F
Z = 0

Пример 3:

```
IORLW   HIGH (LU_TABLE)
```

До выполнения команды
W = 0x9A
LU_TABLE = 0x9375 (адрес в памяти программ)
После выполнения команды
W = 0x9B

Пример 4:

```
IORLW   0x00
```

До выполнения команды
W = 0x00
После выполнения команды
W = 0x00
Z = 1

IORWF Побитное 'ИЛИ' W и f

Синтаксис: [*label*] IORWF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (W) .OR. (f) → (dest)

Измен. флаги: Z

Код:

00	0100	dfff	ffff

Описание: Выполняется побитное 'ИЛИ' содержимого регистров W и 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1:
```
IORWF   RESULT,0
```
До выполнения команды
RESULT = 0x13
W = 0x91
После выполнения команды
RESULT = 0x13
W = 0x93
Z = 0

Пример 2:
```
IORWF   INDF,1
```
До выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x30
После выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x37
Z = 0

Пример 3:
```
IORWF   RESULT,1
```
Случай 1
До выполнения команды
RESULT = 0x13
W = 0x91
После выполнения команды
RESULT = 0x93
W = 0x91
Z = 0

Случай 2
До выполнения команды
RESULT = 0x00
W = 0x00
После выполнения команды
RESULT = 0x00
W = 0x00
Z = 1

MOVF Переслать f

Синтаксис: [*label*] MOVF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (f) → (dest)

Измен. флаги: Z

Код:

00	1000	dfff	ffff

Описание: Содержимое регистра 'f' пересылается в регистр адресата. Если d=0, значение сохраняется в регистре W. Если d=1, значение сохраняется в регистре 'f'. d=1 используется для проверки содержимого регистра 'f' на ноль.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1: MOVF FSR,0

До выполнения команды
W = 0x00
FSR = 0xC2

После выполнения команды
W = 0xC2
FSR = 0xC2
Z = 0

Пример 2: MOVF INDF,1

До выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x00

После выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x00
Z = 1

Пример 3: MOVF FSR,1

Случай 1

До выполнения команды
FSR = 0x43

После выполнения команды
FSR = 0x43
Z = 0

Случай 2

До выполнения команды
FSR = 0x00

После выполнения команды
FSR = 0x00
Z = 1

MOVLW Переслать константу в W

Синтаксис: [*label*] MOVLW k

Операнды: $0 \le k \le 255$

Операция: $k \rightarrow (W)$

Измен. флаги: Нет

Код:

11	00xx	kkkk	kkkk

Описание: Переслать константу 'k' в регистр W. В неиспользуемых битах ассемблер устанавливает '0'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы 'k'	Выполнение	Запись в регистр W

Пример 1: MOVLW 0x5A

После выполнения команды

W = 0x5A

Пример 2: MOVLW MYREG

До выполнения команды

W = 0x10
MYREG = 0x37 (адрес регистра)

После выполнения команды

W = 0x37
Z = 0

Пример 3: MOVLW HIGH (LU_TABLE)

До выполнения команды

W = 0x10
LU_TABLE = 0x9375 (адрес в памяти программ)

После выполнения команды

W = 0x93

MOVWF Переслать W в f

Синтаксис: [*label*] MOVWF f

Операнды: $0 \le f \le 127$

Операция: $(W) \rightarrow (f)$

Измен. флаги: Нет

Код:

00	0000	1fff	ffff

Описание: Переслать содержимое регистра W в регистр 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись в регистр 'f'

Пример 1:

```
MOVWF   OPTION_REG
```

До выполнения команды
OPTION = 0xFF
W = 0x4F

После выполнения команды
OPTION = 0x4F
W = 0x4F

Пример 2:

```
MOVWF   INDF
```

До выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x00

После выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x17

NOP Нет операции

Синтаксис: [*label*] NOP

Операнды: Нет

Операция: Нет операции

Измен. флаги: Нет

Код:

00	0000	0xx0	0000

Описание: Нет операции

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Нет операции	Нет операции

Пример 1: HERE NOP

До выполнения команды

PC = адрес HERE

После выполнения команды

PC = адрес HERE + 1

OPTION Загрузить регистр OPTION

Синтаксис:	[*label*] OPTION
Операнды:	Нет
Операция:	(W) → OPTION
Измен. флаги:	Нет

Код:

00	0000	0110	0010

Описание: Переслать содержимое регистра W в регистр OPTION. Инструкция поддерживается для совместимости программы с семейством микроконтроллеров PIC16C5X. Запись/чтение регистра OPTION можно выполнить прямой или косвенной адресацией.

Слов:	1
Циклов:	1
Пример:	

Для совместимости программного обеспечения с последующими выпускаемыми микроконтроллерами семейства PIC16CXX не рекомендуется использовать эту инструкцию.

RETFIE Возврат из подпрограммы с разрешением прерываний

Синтаксис: [*label*] RETFIE

Операнды: Нет

Операция: TOS → PC
1 → GIE

Измен. флаги: Нет

Код:

00	0000	0000	1001

Описание: Возврат из подпрограммы обработки прерываний. Вершина стека TOS загружается в счетчик команд PC. Устанавливается в '1' флаг глобального разрешения прерываний GIE(INTCON<7>). Инструкция выполняется за 2 цикла.

Слов: 1

Циклов: 2

Выполнение команды по тактам

1-й цикл

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Выполнение	Нет операции

2-й цикл

Q1	Q2	Q3	Q4
Нет операции	Нет операции	Нет операции	Нет операции

Пример 1: RETFIE

После выполнения команды
PC = TOS
GIE = 1

RETLW Возврат из подпрограммы с загрузкой константы в W

Синтаксис: [*label*] RETLW k

Операнды: $0 \le k \le 255$

Операция: $k \rightarrow (W)$
$TOS \rightarrow PC$

Измен. флаги: Нет

Код:

11	01xx	kkkk	kkkk

Описание: В регистр W загружается 8-разрядная константа. Вершина стека TOS загружается в счетчик команд PC. Инструкция выполняется за 2 цикла.

Слов: 1

Циклов: 2

Выполнение команды по тактам

1-й цикл

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы 'k'	Выполнение	Запись в регистр W

2-й цикл

Q1	Q2	Q3	Q4
Нет операции	Нет операции	Нет операции	Нет операции

Пример 1:

```
        CALL    TABLE
        •
        •
TABLE   ADDWF   PCL, f
        RETLW   k1
        RETLW   k2
        •
        •
        RETLW   kn
```

До выполнения команды
W = 0x07

После выполнения команды
W = значение k8
PC = TOS = адрес HERE +1

RETURN Возврат из подпрограммы

Синтаксис: [*label*] RETURN

Операнды: Нет

Операция: TOS → PC

Измен. флаги: Нет

Код:

00	0000	0000	1000

Описание: Возврат из подпрограммы. Вершина стека TOS загружается в счетчик команд PC. Инструкция выполняется за 2 цикла.

Слов: 1

Циклов: 2

Выполнение команды по тактам

1-й цикл

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Выполнение	Нет операции

2-й цикл

Q1	Q2	Q3	Q4
Нет операции	Нет операции	Нет операции	Нет операции

Пример 1: RETURN

После выполнения команды

PC = TOS

RLF Циклический сдвиг f влево через перенос

Синтаксис: [*label*] RLF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

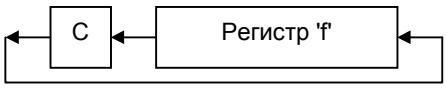
Операция: См. описание

Измен. флаги: C

Код:

00	1101	dfff	ffff

Описание: Выполняется циклический сдвиг влево содержимого регистра 'f' через бит C регистра STATUS. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.



Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1: RLF REG1,0

До выполнения команды
REG1 = 1110 0110
C = 0

После выполнения команды
REG1 = 1110 0110
W = 1100 1100
C = 1

Пример 2: RLF INDF,1

Случай 1 До выполнения команды
W = xxxx xxxx
FSR = 0xC2
Значение регистра с адресом в FSR = 0x3A (0011 1010)
C = 1

После выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x75 (0111 0101)
C = 0

Случай 2 До выполнения команды
W = xxxx xxxx
FSR = 0xC2
Значение регистра с адресом в FSR = 0xB9 (1011 1001)
C = 0

После выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x72 (0111 0010)
C = 1

RRF Циклический сдвиг f вправо через перенос

Синтаксис: [*label*] RRF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

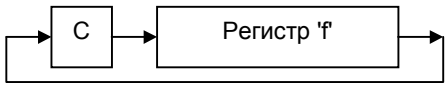
Операция: См. описание

Измен. флаги: C

Код:

00	1100	dfff	ffff

Описание: Выполняется циклический сдвиг вправо содержимого регистра 'f' через бит C регистра STATUS. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.



Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1: RRF REG1,0

До выполнения команды
REG1 = 1110 0110
W = xxxx xxxx
C = 0

После выполнения команды
REG1 = 1110 0110
W = 0111 0011
C = 0

Пример 2: RRF INDF,1

Случай 1 До выполнения команды
W = xxxx xxxx
FSR = 0xC2
Значение регистра с адресом в FSR = 0x3A (0011 1010)
C = 1

После выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x9D (1001 1101)
C = 0

Случай 2 До выполнения команды
W = xxxx xxxx
FSR = 0xC2
Значение регистра с адресом в FSR = 0x39 (0011 1001)
C = 0

После выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x1C (0001 1100)
C = 1

SLEEP Перейти в режим SLEEP

Синтаксис: [*label*] SLEEP

Операнды: Нет

Операция:
00h → WDT
00h → предделитель WDT
1 → - TO
0 → - PD

Измен. флаги: -TO, -PD

Код:

00	0000	0110	0011

Описание: Сбросить флаг включения питания -PD в '0'. Установить флаг переполнения WDT -TO в '1'. Очистить таймер WDT и его предделитель. Перевести микроконтроллер в режим SLEEP и выключить тактовый генератор.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Нет операции	Нет операции	Переход в SLEEP режим

Пример 1: SLEEP

Примечание. Команда SLEEP не влияет на коэффициент делителя WDT.

SUBLW Вычесть W из константы

Синтаксис: [*label*] SUBLW k

Операнды: $0 \le k \le 255$

Операция: $k - (W) \rightarrow (W)$

Измен. флаги: C, DC, Z

Код:

11	110x	kkkk	kkkk

Описание: Вычесть содержимое регистра W из 8-разрядной константы 'k'. Результат сохраняется в регистре W.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы 'k'	Выполнение	Запись в регистр W

Пример 1: SUBLW 0x02

Случай 1

До выполнения команды
W = 0x01
C = ?
Z = ?

После выполнения команды
W = 0x01
C = 1 ; результат положительный
Z = 0

Случай 2

До выполнения команды
W = 0x02
C = ?
Z = ?

После выполнения команды
W = 0x00
C = 1 ; результат нулевой
Z = 1

Случай 3

До выполнения команды
W = 0x03
C = ?
Z = ?

После выполнения команды
W = 0xFF
C = 0 ; результат отрицательный
Z = 0

Пример 2: SUBLW MYREG

До выполнения команды
W = 0x10
MYREG = 0x37 (адрес регистра)

После выполнения команды
W = 0x27
C = 1

SUBWF Вычесть W из f

Синтаксис: [*label*] SUBWF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (f) - (W) → (dest)

Измен. флаги: C, DC, Z

Код:

00	0010	dfff	ffff

Описание: Вычесть содержимое регистра W из регистра 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1: SUBWF REG1,1

Случай 1 До выполнения команды
REG1 = 0x03
W = 0x02
C = x
Z = x

После выполнения команды
REG1 = 0x01
W = 0x02
C =1 ; результат положительный
Z = 0

Случай 2 До выполнения команды
REG1 = 0x02
W = 0x02
C = x
Z = x

После выполнения команды
REG1 = 0x00
W = 0x02
C =1 ; результат нулевой
Z = 1

Случай 3 До выполнения команды
REG1 = 0x01
W = 0x02
C = x
Z = x

После выполнения команды
REG1 = 0xFF
W = 0x02
C =0 ; результат отрицательный
Z = 0

SWAPF Поменять местами полубайты в регистре f

Синтаксис: [*label*] SWAPF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (f<3:0>) → (dest<7:4>)
(f<7:4>) → (dest<3:0>)

Измен. флаги: Нет

Код:

00	1110	dfff	ffff

Описание: Поменять местами старший и младший полубайты регистра 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1:
```
SWAPF   REG,0
```
До выполнения команды
REG = 0xA5
После выполнения команды
REG = 0xA5
W = 0x5A

Пример 2:
```
SWAPF   INDF,1
```
До выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x20
После выполнения команды
W = 0x17
FSR = 0xC2
Значение регистра с адресом в FSR = 0x02

Пример 3:
```
SWAPF   REG,1
```
До выполнения команды
REG = 0xA5
После выполнения команды
REG = 0x5A

TRIS Загрузить регистр TRIS

Синтаксис:	[*label*] TRIS f
Операнды:	$5 \le f \le 7$
Операция:	(W) → TRIS регистр f
Измен. флаги:	Нет

Код:

00	0000	0110	0fff

Описание: Переслать содержимое W в регистр TRIS. Инструкция поддерживается для совместимости программы с семейством микроконтроллеров PIC16C5X. Запись/чтение регистра OPTION можно выполнить прямой или косвенной адресацией.

Слов:

Циклов:

Пример:

Для совместимости программного обеспечения с последующими выпускаемыми микроконтроллерами семейства PIC16CXX не рекомендуется использовать эту инструкцию.

XORLW Побитное 'исключающее ИЛИ' константы и W

Синтаксис: [*label*] IORLW k

Операнды: $0 \le k \le 255$

Операция: (W) .XOR. k → (W)

Измен. флаги: Z

Код:

11	1010	kkkk	kkkk

Описание: Выполняется побитное 'исключающее ИЛИ' содержимого регистра W и 8-разрядной константы 'k'. Результат сохраняется в регистре W.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение константы 'k'	Выполнение	Запись в регистр W

Пример 1:

```
XORLW   0xAF (1010 1111)
До выполнения команды
        W = 0xB5 (1011 0101)
После выполнения команды
        W = 0x1A (0001 1010)
        Z = 0
```

Пример 2:

```
XORLW   MYREG
До выполнения команды
        W = 0xAF
        MYREG = 0x37 (адрес регистра)
После выполнения команды
        W = 0x18
        Z = 0
```

Пример 3:

```
XORLW   HIGH (LU_TABLE)
До выполнения команды
        W = 0xAF
        LU_TABLE = 0x9375 (адрес в памяти программ)
После выполнения команды
        W = 0x3C
        Z = 0
```

XORWF Побитное 'исключающее ИЛИ' W и f

Синтаксис: [*label*] XORWF f,d

Операнды: $0 \le f \le 127$
$d \in [0,1]$

Операция: (W) .XOR. (f) → (dest)

Измен. флаги: Z

Код:

00	0100	dfff	ffff

Описание: Выполняется побитное 'исключающее ИЛИ' содержимого регистров W и 'f'. Если d=0, результат сохраняется в регистре W. Если d=1, результат сохраняется в регистре 'f'.

Слов: 1

Циклов: 1

Выполнение команды по тактам

Q1	Q2	Q3	Q4
Декодирование команды	Чтение регистра 'f'	Выполнение	Запись результата

Пример 1:
```
XORWF   REG,1
```
До выполнения команды
```
REG = 0xAF
W = 0xB5
```
После выполнения команды
```
REG = 0x1A
W = 0xB5
```

Пример 2:
```
XORWF   REG,0
```
До выполнения команды
```
REG = 0xAF
W = 0xB5
```
После выполнения команды
```
REG = 0xAF
W = 0x1A
```

Пример 3:
```
XORWF   INDF,1
```
До выполнения команды
```
W = 0xB5
FSR = 0xC2
Значение регистра с адресом в FSR = 0xAF
```
После выполнения команды
```
W = 0xB5
FSR = 0xC2
Значение регистра с адресом в FSR = 0x1A
```

29.6 Ответы на часто задаваемые вопросы

Если вы не найдете ответа на Ваш вопрос в этой главе раздела, задайте его, написав нам письмо по адресу support@microchip.ru.

Вопрос 1: Как можно непосредственно изменить значение регистра W? Требуется декрементировать значение в регистре W.

Ответ 1:

Существует несколько способов:

1. В микроконтроллерах PICmicro среднего семейства предусмотрены команды, работающие с регистром W. Например, если необходимо выполнить декремент содержимого регистра W, то можно это выполнить командой ADDLW 0xFF (0x - префикс, обозначающий шестнадцатеричное число в ассемблере MPASM).
2. Заметьте, что большинство команд могут изменять содержимое регистра памяти данных с сохранением результата в W. Это означает, что можно выполнить декремент значения при загрузке регистра W. Адресатом выполнения команды должен быть W (DECF адрес регистра, W).

Вопрос 2: Существует ли какая-нибудь опасность использования команды TRIS для микроконтроллеров PIC16CXXX? В документации на микроконтроллер сказано, что не рекомендуется использовать эти команды.

Ответ 2:

Для совместимости программного обеспечения с последующими версиями микроконтроллеров не рекомендуется использовать команду TRIS. Необходимо учитывать, что с помощью команды TRIS можно настроить только порты A, B и C. Новые микроконтроллеры могут не поддерживать эту команду.

Вопрос 3: Нужно переключать банк памяти данных (выбрать банк 1) при использовании команды TRIS? Требуется настроить направление каналов ввода/вывода PORTA.

Ответ 3:

При использовании команды TRIS переключать банк памяти данных не требуется. Для совместимости программного обеспечения с последующими версиями микроконтроллеров не рекомендуется использовать команду TRIS.

Вопрос 4: В документации указано, что требуется осторожность при использовании команд "чтение - модификация - запись". Объясните пожалуйста почему.

Ответ 4:

Простой пример команды со структурой "чтение - модификация - запись" - бит ориентированная команда BCF. Можно предполагать, что выполняется просто сброс бита, управляющего выводом порта. Фактически происходит чтение состояния всего порта ввода/вывода, затем сбрасывается требуемый бит и новое значение записывается в порт ввода/вывода (или регистр). Любая команда, зависящая от текущего значение регистра является командой со структурой "чтение - модификация - запись" (ADDWF, SUBWF, BCF, BSF, INCF, XORWF и др.). Команды, независящие от текущего значения регистра не являются командами "чтение - модификация - запись" (MOVWF, CLRF и др.).

Рассмотрим одну ситуацию выполнения команд "чтение - модификация - запись" для порта ввода/вывода. Например, все биты регистра TRISB настраивают PORTB на выход, и на всех выводах PORTB установлен высокий логический уровень сигнала. Теперь Вы настраиваете RB3 как вход, на котором присутствует низкий логический уровень. Выполняете команду BCF PORTB,6, чтобы на RB6 установить низкий логический уровень. Если Вы опять настроите вывод RB3 как выход, то на нем будет формироваться низкий логический уровень, хотя ранее Вы устанавливали высокий логический уровень. При выполнении команды BCF для другого вывода порта (RB6) происходит чтение состояния всего порта (включая 0 на RB3). Бит 6 изменяется к требуемому значению, но т.к. на RB3 был прочитан '0', он будет записан в защелку порта. Когда вывод будет настроен на выход, новое значение будет передано на вывод.

Вопрос 5: При использовании команды BCF другие выводы порта принимают низкий логический уровень. Почему?

Ответ 5:

1. Случай, когда команда "чтение - модификация - запись" изменяет состояние других выводов порта, можно продемонстрировать следующим образом. Предположим, что все каналы PORTC настроены на выход и имеют низкий логический уровень. К каждому выводу подключен светодиод, который светится при формировании высокого логического уровня на выводах порта. Параллельно каждому светодиоду подключен конденсатор с емкостью 100мкФ. Программа микроконтроллера выполняется очень быстро, тактовая частота 20МГц. Теперь последовательно формируем команды, включающие светодиоды: BSF PORTC,0; BSF PORTC,1; BSF PORTC,2 и т.д. Вы можете видеть, что только на последнем выводе высокий уровень сигнала и только последний светодиод светится. Это произошло потому , что конденсаторы требуют некоторого времени для зарядки до напряжения высокого логического уровня. Поскольку каждый вывод устанавливался в '1' прежде, чем зарядится конденсатор предыдущего вывода, чтение давало результат '0'. Это '0' записывается в выходную защелку, восстанавливая низкий логический уровень на выводе (команда "чтение - модификация - запись"). Учитывать этот эффект необходимо только при высокой тактовой частоте микроконтроллера и выполнении последовательных операций с портами ввода/вывода.
2. Такая ситуация возможна в микроконтроллерах PIC16C7XX, если Вы не настроили каналы порта ввода/вывода должным образом в регистре ADCON1. Если вывод настроен как аналоговый вход, то чтение будет давать результат '0' независимо от уровня напряжения на выводе. Это исключение к правилу, что всегда выполняется чтение состояние вывода. Вы можете настраивать аналоговый вывод как цифровой выход в регистре TRISA, и управлять логическим уровнем на выходе, но чтение всегда будет давать результат '0'. Поэтому, если вы обращаетесь к порту командой "чтение - модификация - запись", все аналоговые выводы читаются как '0', командой изменяется прочитанное значение и записывается назад в защелку порта как '0'. На аналоговых входах могут присутствовать нелогические уровни, поэтому входные цифровые буферы выключены для предотвращения возможного повышенного энергопотребления.

29.7 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило, примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с системой команд микроконтроллеров PICmicro MCU:

Документ	Номер
В настоящее время документы не подготовлены	

Раздел 30. Электрические характеристики

Содержание

30.1 Введение

Данный раздел предназначен для демонстрации электрических параметров, которые могут быть включены в состав технической документации конкретного микроконтроллера. Этот раздел не предназначен для использования представленных данных. Параметры микроконтроллера Вы должны смотреть в технической документации на соответствующий микроконтроллер. В описании микроконтроллеров и их периферийных модулей даны ссылки на раздел "Электрические характеристики".

Примечание. При начале любого проекта Microchip рекомендует использовать последнюю редакцию документации на микроконтроллер и проверять электрические характеристики, чтобы гарантировать выполнение ваших требований.

В этом разделе будут использоваться обозначения, указанные в таблице 30-1.

Таблица 30-1 Принятые обозначения

Обозначение	Описание
PIC16**C**XXX	Для микроконтроллеров, проверенных в стандартном диапазоне напряжений питания.
PIC16**LC**XXX	Для микроконтроллеров, проверенных в расширенном диапазоне напряжений питания.
PIC16**F**XXX	Для микроконтроллеров, проверенных в стандартном диапазоне напряжений питания.
PIC16**LF**XXX	Для микроконтроллеров, проверенных в расширенном диапазоне напряжений питания.
PIC16**CR**XXX	Для микроконтроллеров, проверенных в стандартном диапазоне напряжений питания.
PIC16**LCR**XXX	Для микроконтроллеров, проверенных в расширенном диапазоне напряжений питания.
PIC16XXXX-**04**	Для микроконтроллеров, проверенных при тактовой частоте до 4МГц.
PIC16XXXX-**08**	Для микроконтроллеров, проверенных при тактовой частоте до 8МГц.
PIC16XXXX-**10**	Для микроконтроллеров, проверенных при тактовой частоте до 10МГц.
PIC16XXXX-**20**	Для микроконтроллеров, проверенных при тактовой частоте до 20МГц.
LP osc	Для микроконтроллеров, у которых выбран LP режим тактового генератора.
XT osc	Для микроконтроллеров, у которых выбран XT режим тактового генератора.
HS osc	Для микроконтроллеров, у которых выбран HS режим тактового генератора.
RC osc	Для микроконтроллеров, у которых выбран RC режим тактового генератора.
Коммерческий	Для микроконтроллеров с коммерческим температурным диапазоном ($0°C \le T_A \le +70°C$).
Промышленный	Для микроконтроллеров с промышленным температурным диапазоном ($-40°C \le T_A \le +85°C$).
Расширенный	Для микроконтроллеров с расширенным температурным диапазоном ($-40°C \le T_A \le +125°C$).

30.2 Абсолютный максимум

Абсолютные максимальные значения определяют наихудшие условия эксплуатации и хранения микроконтроллеров, что не является допустимым рабочим уровнем. Напряжение, выше указанного значения, может привести к повреждению микроконтроллера. Некоторые требования не являются независимыми, они могут быть взаимосвязаны с другими параметрами.

Примером может служить "Максимальный втекающий/вытекающий ток канала ввода/вывода". Число выводов, через которые может одновременно протекать максимальный ток зависит от максимально допустимого тока через выводы V_{DD} и V_{SS}. Физической причиной является ширина шины проводников питания и "земли" портов ввода/вывода и внутренней логики микроконтроллера. Превышение указанных значений может привести к внутреннему обрыву цепи. Превышение абсолютного максимума может привести к снижению надежности микроконтроллера.

Входной ток вывода определен как ток через диод, подключенный к V_{SS}/V_{DD}, если напряжение на выводе выходит за указанные значения.

Максимально допустимые значения (*)

Предельная рабочая температура	от -55°C до +125°C
Температура хранения	от -65°C до +150°C
Напряжение V_{DD} относительно V_{SS}	от -0.3В до +7.5В
Напряжение -MCLR относительно V_{SS}	от 0В до +14В
Напряжение RA4 относительно V_{SS}	от 0В до +8.5В
Напряжение на остальных выводах относительно V_{SS}	от -0.3В до V_{DD}+0.3В
Рассеиваемая мощность (1)	1Вт
Максимальный ток вывода V_{SS}	300мА
Максимальный ток вывода V_{DD}	250мА
Входной запирающий ток I_{IK} ($V_I < 0$ или $V_I > V_{DD}$)	±20мА
Выходной запирающий ток I_{OK} ($V_O < 0$ или $V_O > V_{DD}$)	±20мА
Максимальный выходной ток стока канала ввода/вывода	25мА
Максимальный выходной ток истока канала ввода/вывода	25мА
Максимальный выходной ток стока портов ввода/вывода PORTA, PORTB и PORTE	200мА
Максимальный выходной ток истока портов ввода/вывода PORTA, PORTB и PORTE	200мА
Максимальный выходной ток стока портов ввода/вывода PORTC и PORTD	200мА
Максимальный выходной ток истока портов ввода/вывода PORTC и PORTD	200мА
Максимальный выходной ток стока портов ввода/вывода PORTF и PORTG	100мА
Максимальный выходной ток истока портов ввода/вывода PORTF и PORTG	100мА

Примечание 1. Потребляемая мощность рассчитывается по формуле:

$$P = V_{DD} \times \{I_{DD} - \Sigma I_{OH}\} + \Sigma\{(V_{DD} - V_{OH}) \times I_{OH}\} + \Sigma (V_{OL} \times I_{OL})$$

Примечание *. Выход за указанные значения может привести к необратимым повреждениям микроконтроллера. Не предусмотрена работа микроконтроллера в предельном режиме в течение длительного времени. Длительная эксплуатация микроконтроллера в недопустимых условиях может повлиять на его надежность.

Примечание. Броски напряжения на выводе -MCLR ниже V_{SS} приводят к появлению больших токов (около 80мА), что может привести к срабатыванию защелки. Поэтому рекомендуется последовательно включать резистор сопротивлением от 50ОМ до 100Ом для подачи низкого уровня на этот вывод вместо непосредственного подключения к V_{SS}.

30.3 Таблица выбора микроконтроллеров

В эти таблицы предназначены для определения какие генераторы и параметры микроконтроллеров проверены. Любой режим тактового генератора (из возможных) может быть выбран во время программирования микроконтроллера.

Максимальная тактовая частота микроконтроллера при XT и RC режиме генератора 4МГц, поэтому работа микроконтроллера проверена на частоте до 4МГц.

Микроконтроллеры с максимальной тактовой частотой 10МГц и 20МГц проверяются в HS режиме тактового генератора. В таблице 30-2 значение I_{PD} не точное, поскольку нет точки измерения I_{PD} (оно меняется в зависимости от напряжения питания).

Устройства с питанием от батареек обычно требуют расширенного диапазона напряжений питания микроконтроллеров. Микроконтроллеры с индексом LC имеют расширенный диапазон напряжений питания и проверяются в RC, XT и LP режиме тактового генератора.

Микроконтроллеры с окном для УФ стирания памяти программ проверяются во всех режимах тактового генератора (-04, -20 и LC микроконтроллеры). Температурный диапазон, в котором проверяются микроконтроллеры, должен рассматриваться как реклама. В дальнейшем они могут быть проверены в промышленном и расширенном температурном диапазоне.

Таблица 30-2 Пример параметров микроконтроллеров в различных режимах тактового генератора и тактовой частоты (коммерческий температурный диапазон)

OSC	PIC16CXXX-04	PIC16CXXX-10	PIC16CXXX-20	PIC16LCXXX-04	С окном для УФ стирания
RC	V_{DD}: от 4.0В до 6.0В I_{DD}: макс.5мА@ 5.5В I_{PD}: макс.16мкА@4В F_{REQ}: макс. 4МГц	V_{DD}: от 4.5В до 5.5В I_{DD}: макс.2.7мА@ 5.5В I_{PD}: макс.1.5мкА@4В F_{REQ}: макс. 4МГц	V_{DD}: от 4.5В до 5.5В I_{DD}: макс.2.7мА@ 5.5В I_{PD}: макс.1.5мкА@4В F_{REQ}: макс. 4МГц	V_{DD}: от 2.5В до 6.0В I_{DD}: макс.3.8мА@ 3.0В I_{PD}: макс.5мкА@3В F_{REQ}: макс. 4МГц	V_{DD}: от 2.5В до 6.0В I_{DD}: макс.3.8мА@ 5.5В I_{PD}: макс.16мкА@4В F_{REQ}: макс. 4МГц
XT	V_{DD}: от 4.0В до 6.0В I_{DD}: макс.5мА@ 5.5В I_{PD}: макс.16мкА@4.5В F_{REQ}: макс. 4МГц	V_{DD}: от 4.5В до 5.5В I_{DD}: макс.2.7мА@ 5.5В I_{PD}: макс.1.5мкА@4В F_{REQ}: макс. 4МГц	V_{DD}: от 4.5В до 5.5В I_{DD}: макс.2.7мА@ 5.5В I_{PD}: макс.1.5мкА@4В F_{REQ}: макс. 4МГц	V_{DD}: от 2.5В до 6.0В I_{DD}: макс.3.8мА@ 3.0В I_{PD}: макс.5мкА@3В F_{REQ}: макс. 4МГц	V_{DD}: от 2.5В до 6.0В I_{DD}: макс.3.8мА@ 5.5В I_{PD}: макс.16мкА@4В F_{REQ}: макс. 4МГц
HS	V_{DD}: от 4.5В до 5.5В I_{DD}: макс.13.5мА@ 5.5В I_{PD}: макс.1.5мкА@4В F_{REQ}: макс. 4МГц	V_{DD}: от 4.5В до 5.5В I_{DD}: макс.10мА@ 5.5В I_{PD}: макс.1.5мкА@4.5В F_{REQ}: макс. 10МГц	V_{DD}: от 4.5В до 5.5В I_{DD}: макс.20мА@ 5.5В I_{PD}: макс.1.5мкА@4.5В F_{REQ}: макс. 20МГц	Не рекомендуется использовать в HS режиме генератора	V_{DD}: от 4.5В до 5.5В I_{DD}: макс.20мА@ 5.5В I_{PD}: макс.1.5мкА@4.5В F_{REQ}: макс. 20МГц
LP	V_{DD}: от 4.0В до 6.0В I_{DD}: макс.52.5мкА@ 32кГц, 4.0В I_{PD}: макс.0.9мкА@4В F_{REQ}: макс. 200кГц	Не рекомендуется использовать в LP режиме генератора	Не рекомендуется использовать в LP режиме генератора	V_{DD}: от 2.5В до 6.0В I_{DD}: макс.48мкА@ 32кГц, 3.0В I_{PD}: макс.5мкА@3В F_{REQ}: макс. 200кГц	V_{DD}: от 2.5В до 6.0В I_{DD}: макс.48мкА@ 32кГц, 3.0В I_{PD}: макс.5мкА@3В F_{REQ}: макс. 200кГц

В затененный ячейках показаны режимы, в которых проверена работа микроконтроллера, но не в мин./макс. значения. Не рекомендуется выбирать эти режимы работы микроконтроллера.

Примечание. Микроконтроллеры, отмеченные знаком ENG SMP (Engineering Sample), протестированы по определенной программе заводских испытаний. Не гарантируется, что отдельно взятый микроконтроллер был протестирован по всем параметрам, указанным в технической документации.

30.4 Параметры, связанные с напряжение питания микроконтроллера

Эти параметры относятся к напряжению питания и включению микроконтроллера

Напряжение питания - уровень напряжений, при котором микроконтроллер нормально работает.

Напряжение сохранения данных в ОЗУ - напряжение, при котором не происходит разрушение информации в памяти данных микроконтроллера.

Начальное напряжение V_{DD} - уровень напряжения, начиная с которого происходит повышение напряжения питания до номинального уровня для гарантированной работы схемы сброса по включению питания POR.

Скорость нарастания напряжения V_{DD} - минимальная скорость нарастания напряжения при включении питания для гарантированной работы схемы сброса по включению питания POR.

Напряжение сброса по снижению напряжения питания - диапазон напряжений питания, в котором может произойти сброс по снижению напряжения питания. Когда напряжение питания попадает в этот диапазон, микроконтроллер может находиться в состоянии сброса или только что выйти из него.

Таблица 30-3 Пример параметров по постоянному току

Характеристики по постоянному току			**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Коммерческий 0°C ≤ T_A ≤ +70°C Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C				
№ пар.	**Обоз.**	**Описание**	**Мин.**	**Тип.****	**Макс.**	**Ед.**	**Примечание**
	V_{DD}	Напряжение питания					
D001		PIC16**C**XXX	4.0	-	6.0	В	LP, XT, RC режим генератора
		PIC16**LC**XXX	2.5	-	6.0	В	
D001A		PIC16**C**XXX	4.5		5.5	В	HS режим генератора
D002	V_{DR}	Напряжение сохранения данных в ОЗУ (1)	1.5	-	-	В	
D003	V_{POR}	Стартовое напряжение V_{DD} для формирования POR	-	V_{SS}	-	В	Смотрите раздел "сброс POR"
D004	S_{VDD}	Скорость нарастания V_{DD} для формирования POR	0.05	-	-	В/мс	Смотрите раздел "сброс POR"
	V_{BOR}	Напряжение детектора BOR					
D005			3.7	4.0	4.3	В	Бит BODEN = 0
D005A			3.7	4.0	4.4	В	Расширенный темп. диапазон

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Примечание 1. Предел, до которого может быть понижено напряжение питания V_{DD} без потери данных в ОЗУ.

30.5 Параметры, связанные с током потребления микроконтроллеров

I_{DD} - ток потребления микроконтроллера в нормальном режиме работы. Измерения проводились, когда все порты ввода/вывода настроены как входы и имеют низкий или высокий логический уровень (нет выводов с неопределенным уровнем и выводов с подключенной нагрузкой).

I_{PD} - ток потребления микроконтроллера в SLEEP режиме микроконтроллера. Измерения проводились, когда все порты ввода/вывода настроены как входы и имеют низкий или высокий логический уровень (нет выводов с неопределенным уровнем и выводов с подключенной нагрузкой).

Микроконтроллер может иметь дополнительные особенности и периферийные модули, которые могут работать в SLEEP режиме микроконтроллера:

- Сторожевой таймер WDT;
- Схема сброса по снижению напряжения питания;
- Таймер TMR1;
- АЦП;
- Модуль LCD;
- Компараторы;
- Источник опорного напряжения.

Когда все эти модули выключены, микроконтроллер потребляет наименьший ток (ток утечки). Если любой из этих модулей включен, то потребление в SLEEP режиме значительно увеличится. Разница между минимально возможным током потребления и током потребления с включенным одним из перечисленных модулей (например WDT) называется дифференциальным током потребления модуля. Если включено несколько дополнительных модулей, то суммарный ток потребления может быть легко вычислен: основной ток потребления (ток потребления в SLEEP режиме, когда все модули выключены) плюс дифференциальный ток потребления каждого включенного периферийного модуля.

В примере 30-1 показано вычисление тока потребления микроконтроллера в SLEEP режиме с включенным WDT и TMR1 (внутренний генератор TMR1) при напряжении питания 5В.

Пример 30-1 Вычисление I_{PD} при напряжении питания 5В (включен WDT и TMR1 с внутренним генератором)

Основной ток	14нА	; Ток утечки микроконтроллера
Ток потребления WDT	14мкА	; 14мкА - 14нА = 14мкА
Ток потребления TMR1	22мкА	; 22мкА - 14нА = 22мкА
Суммарный ток потребления	36мкА	;

Таблица 30-4 Пример параметров по постоянному току

Характеристики по постоянному току			**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Коммерческий 0°C ≤ T_A ≤ +70°C Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C				
№ пар.	**Обоз.**	**Описание**	**Мин.**	**Тип.****	**Макс.**	**Ед.**	**Примечание**
	I_{DD}	Ток потребления (2, 4, 5)					
D010							XT, RC режим генератора (PIC16**C**XXX-**04**)
			-	2.7	5	мА	F_{OSC} = 4МГц, V_{DD}=5.5В
			-	2.0	3.8	мА	F_{OSC} = 4МГц, V_{DD}=3.0В
D010A			-	22.5	48	мкА	LP режим генератора F_{OSC} = 32кГц, V_{DD}=3.0В, WDT выключен
D010C			-	2.7	5	мА	INTRC режим генератора F_{OSC} = 4МГц, V_{DD}=5.5В
D013			-	13.5	30	мА	HS режим генератора (PIC16**C**XXX-**20**) F_{OSC} = 20МГц, V_{DD}=5.5В
	I_{PD}	Ток потребления в SLEEP режиме (3,5)					
D020			-	10.5	42	мкА	V_{DD}=4.0В, WDT включен, от -40°C до +85°C
			-	7.5	30	мкА	V_{DD}=3.0В, WDT включен, от -40°C до +85°C
			-	1.5	21	мкА	V_{DD}=4.0В, WDT выключен, от -0°C до +70°C
D021			-	0.9	13.5	мкА	V_{DD}=3.0В, WDT выключен, от 0°C до +70°C
			-	1.5	24	мкА	V_{DD}=4.0В, WDT выключен, от -40°C до +85°C
D021A			-	0.9	18	мкА	V_{DD}=3.0В, WDT выключен, от -40°C до +85°C
D021B			-	1.5	-	мкА	V_{DD}=4.0В, WDT выключен, от -40°C до +125°C

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Примечания:

1. Не относится к данной таблице.
2. Ток потребления в основном зависит от напряжения питания и тактовой частоты. Другие факторы, влияющие на ток потребления: выходная нагрузка и частота переключения каналов ввода/вывода; тип тактового генератора; температура и выполняемая программа. Измерения I_{DD} проводилось в следующих условиях: внешний тактовый сигнал (меандр); каналы портов ввода/вывода в третьем состоянии и подтянуты к V_{DD}; -MCLR = V_{DD}; WDT выключен/выключен, указано в спецификации.
3. Потребляемый ток в SLEEP режиме не зависит от типа тактового генератора. При измерении тока все каналы портов ввода/вывода в третьем состоянии и подтянуты к V_{DD} или V_{SS}.
4. В RC режиме генератора ток через внешний резистор не учитывается. Ток, протекающий через внешний резистор, может быть рассчитан по формуле: Ir = $V_{DD}/2R_{EXT}$ (мА), где R_{EXT} в кОм.
5. Генератор TMR1 дополнительно потребляет 20мкА (если включен). Этот параметр используется при разработке устройств, но не тестируется.

Таблица 30-5 Пример параметров по постоянному току

Характеристики по постоянному току			**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Коммерческий 0°C ≤ T_A ≤ +70°C Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C				
№ пар.	**Обоз.**	**Описание**	**Мин.**	**Тип.****	**Макс.**	**Ед.**	**Примечание**
		Дифференциальный ток потребления периферийных модулей					
D022	ΔI_{WDT}	Сторожевой таймер	-	6.0	20	мкА	V_{DD} = 4.0В
			-	-	25	мкА	от -40°C до +85°C
D022A	ΔI_{BOR}	Схема сброса по снижению напряжения питания	-	350	425	мкА	BODEN = 0, V_{DD} = 5.0В
D023	ΔI_{COMP}	Компаратор	-	85	100	мкА	V_{DD} = 4.0В
D023A	ΔI_{REF}	Источник опорного напряжения	-	94	300	мкА	V_{DD} = 4.0В
D024	ΔI_{LCDRC}	Внутренний RC генератор LCD	-	6.0	20	мкА	V_{DD} = 3.0В
D024A	ΔI_{LCDVG}	Генератор напряжения LCD	-	TBD	TBD	мкА	V_{DD} = 3.0В
D025	ΔI_{T1OSC}	Генератор TMR1	-	3.1	6.5	мкА	V_{DD} = 3.0В
D026	ΔI_{AD}	АЦП	-	1.0	-	мкА	АЦП включено, но преобразование не выполняется
D027	ΔI_{SAD}	Интегрирующее АЦП (общее)	-	165*	250*	мкА	REFOFF = 0
D027A	ΔI_{SADVR}	Интегрирующее АЦП Опорное напряжение	-	20*	30*	мкА	REFOFF = 0
D027B	$\Delta I_{SADCDAC}$	Интегрирующее АЦП Источник тока	-	50*	70*	мкА	ADCON1<7:4> = 1111b
D027C	$\Delta I_{SADSREF}$	Интегрирующее АЦП Делитель опорного напряжения	-	55*	85*	мкА	ADOFF = 0
D027D	ΔI_{SADCMP}	Интегрирующее АЦП Компаратор	-	40*	65*	мкА	ADOFF = 0

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

30.6 Пороговые уровни входного напряжения

Входное напряжение низкого уровня (V_{IL}) - уровень напряжения, который будет читаться как '0'. Если напряжение на выводе выше указанного значения, то состояние вывода не будет читаться как '0'. Во всех проектах требуется учитывать это напряжение для каждой микросхемы, чтобы гарантировать выполнение данного требования.

Входное напряжение высокого уровня (V_{IH}) - уровень напряжения, который будет читаться как '1'. Если напряжение на выводе ниже указанного значения, то состояние вывода не будет читаться как '1'. Во всех проектах требуется учитывать это напряжение для каждой микросхемы, чтобы гарантировать выполнение данного требования.

Выводы с входным буфером ТТЛ имеют два типа параметров: в соответствии с промышленным стандартом ТТЛ при напряжении питания от 4.5В до 5.5В; параметры для всего диапазона напряжения питания. В проекте может использоваться лучший тип параметров.

Таблица 30-6 Пример параметров по постоянному току

Характеристики по постоянному току			**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Коммерческий 0°C ≤ T_A ≤ +70°C Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C Параметры напряжения питания смотрите в таблице 30-3.				
№ пар.	**Обоз.**	**Описание**	**Мин.**	**Тип.****	**Макс.**	**Ед.**	**Примечание**
	V_{IL}	Входное напряжение низкого уровня					
		Канал порта ввода/вывода					
D030		ТТЛ буфер	V_{SS}	-	0.8	В	V_{DD} = от 4.5В до 5.5В(4)
D030A			V_{SS}	-	$0.15V_{DD}$	В	Иначе(4)
D031		Триггер Шмидта	V_{SS}	-	$0.2V_{DD}$	В	Весь диапазон V_{DD}
D032		-MCLR, OSC1 (RC)(1)	V_{SS}	-	$0.2V_{DD}$	В	
D033		OSC1 (XT, HS, LP)	V_{SS}	-	$0.3V_{DD}$	В	
	V_{IH}	Входное напряжение высокого уровня					
		Канал порта ввода/вывода					
D040		ТТЛ буфер	2.0	-	V_{DD}	В	V_{DD} = от 4.5В до 5.5В(4)
D040A			$0.25V_{DD}$+0.8	-	V_{DD}	В	Иначе(4)
D041		Триггер Шмидта	$0.8V_{DD}$	-	V_{DD}	В	Весь диапазон V_{DD}
D042		-MCLR	$0.8V_{DD}$	-	V_{DD}	В	
D042A		OSC1 (XT, HS, LP)	$0.7V_{DD}$	-	V_{DD}	В	
D043		OSC1 (RC)(1)	$0.9V_{DD}$	-	V_{DD}	В	
D050	V_{HYS}	Гистерезис входа триггера Шмидта	TBD	-	-	В	

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Примечания:

1. В RC режиме генератора на входе OSC1 включен триггер Шмидта. Не рекомендуется использовать внешний тактовый сигнал в RC режиме тактового генератора.
2. Не относится к данной таблице.
3. Не относится к данной таблице.
4. Может использоваться лучшее из двух значений. Для V_{IL} - более высокое значение, для V_{IH} - более низкое.

30.7 Ток порта ввода/вывода

Ток подтягивающего резистора PORT/GPIO - дополнительный потребляемый ток при включенных подтягивающих резисторах.

Ток утечки - паразитный ток, связанный со схемотехнической реализацией вывода. В идеале не должно быть никаких токов утечки.

Таблица 30-7 Пример параметров по постоянному току

Характеристики по постоянному току			**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Коммерческий 0°C ≤ T_A ≤ +70°C Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C Параметры напряжения питания смотрите в таблице 30-3.				
№ пар.	**Обоз.**	**Описание**	**Мин.**	**Тип.****	**Макс.**	**Ед.**	**Примечание**
	I_{IL}	Входной ток утечки (2,3)					
D060		Порт ввода/вывода	-	-	±1	мкА	$V_{SS} \le V_{PIN} \le V_{DD}$, 3-е сост.
D060A		CDAC	-	-	±1	мкА	$V_{SS} \le V_{PIN} \le V_{DD}$, 3-е сост.
D061		-MCLR, RA4/T0CKI	-	-	±5	мкА	$V_{SS} \le V_{PIN} \le V_{DD}$
D063		OSC1	-	-	±5	мкА	$V_{SS} \le V_{PIN} \le V_{DD}$, XT, HS, LP
		Ток через подтягивающие резисторы					
D070		PORTB	50	250	400	мкА	V_{DD} = 5.0В, $V_{PIN} = V_{SS}$
D070A		GPIO	50	250	400	мкА	V_{DD} = 5.0В, $V_{PIN} = V_{SS}$
		Программируемый источник тока (интегрирующее АЦП)					Вывод CDAC = 0В
D160		Выходной ток	18.75	33.75	48.75	мкА	ADCON1<7:4> = 1111b полная шкала
D160A			1.25	2.25	3.25	мкА	ADCON1<7:4> = 0001b (1 Lsb)
D160B			-0.5	0	0.5	мкА	ADCON1<7:4> = 0000b (нуль шкалы)

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Примечания:

1. В RC режиме генератора на входе OSC1 включен триггер Шмидта. Не рекомендуется использовать внешний тактовый сигнал в RC режиме тактового генератора.
2. Ток утечки на выводе -MCLR зависит от приложенного напряжения. Параметры указаны для нормального режима работы. В других режимах может возникнуть больший ток утечки.
3. Отрицательный ток показывает, что он вытекает из вывода.

30.8 Напряжение выходного драйвера вывода

Выходное напряжение низкого уровня (V_{OL}) - это напряжение зависит от внешних цепей, подключенных к выводу. Если вывод соединить с V_{DD}, то на выводе порта никогда не будет достигнуто напряжение низкого уровня независимо от нагрузочной способности выходного буфера (ток потребления значительно возрастет). V_{OL} - напряжение низкого уровня на выводе порта при условии, что ток вывода не превышает I_{OL}.

Выходное напряжение высокого уровня (V_{OH}) - это напряжение зависит от внешних цепей, подключенных к выводу. Если вывод соединить с V_{SS}, то на выводе порта никогда не будет достигнуто напряжение высокого уровня независимо от нагрузочной способности выходного буфера (ток потребления значительно возрастет). V_{OH} - напряжение высокого уровня на выводе порта при условии, что ток вывода не превышает I_{OH}.

Таблица 30-8 Пример параметров по постоянному току

Характеристики по постоянному току			**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Коммерческий 0°C ≤ T_A ≤ +70°C Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C Параметры напряжения питания смотрите в таблице 30-3.				
№ пар.	**Обоз.**	**Описание**	**Мин.**	**Тип.****	**Макс.**	**Ед.**	**Примечание**
	V_{OL}	Выходное напряжение низкого уровня					V_{DD} = 4.5В
D080		Канал ввода/вывода	-	-	0.6	В	I_{OL}=8.5 мА, -40°C до +85°C
D080A			-	-	0.6	В	I_{OL}=7.0 мА, -40°C до +125°C
D083		OSC2/CLKOUT (RC)	-	-	0.6	В	I_{OL}=1.6 мА, -40°C до +85°C
D083A			-	-	0.6	В	I_{OL}=1.2 мА, -40°C до +125°C
	V_{OH}	Выходное напряжение высокого уровня					V_{DD}=4.5В
D090		Канал ввода/вывода (3)	V_{DD} - 0.7	-	-	В	I_{OH}=-3.0 мА,-40°C до +85°C
D090A			V_{DD} - 0.7	-	-	В	I_{OH}=-2.5 мА,-40°C до +125°C
D092		OSC2/CLKOUT (RC)	V_{DD} - 0.7	-	-	В	I_{OH}=-1.3 мА,-40°C до +85°C
D092A			V_{DD} - 0.7	-	-	В	I_{OH}=-1.0 мА,-40°C до +125°C
D150	V_{OD}	Напряжение на выходе с открытым стоком	-	-	12	В	RA4
		Программируемый источник тока					
D170	V_{PCS}	Диапазон напряжений вывода	V_{SS}	-	V_{DD}-1.4	В	Вывод CDAC
D171	SN_{PCS}	Чувствительность напряжения	-0.1	-0.01	-	%/В	$V_{SS} \le V_{CDAC} \le V_{DD}$ - 1.4
D180	V_{BGR}	Опорное напряжение	1.14	1.19	1.24	В	Вывод AN0, когда AMUXOE=1 и ADCS3:ADCS0 = 0100b

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Примечания:

1. В RC режиме генератора на входе OSC1 включен триггер Шмидта. Не рекомендуется использовать внешний тактовый сигнал в RC режиме тактового генератора.
2. Ток утечки на выводе -MCLR зависит от приложенного напряжения. Параметры указаны для нормального режима работы. В других режимах может возникнуть больший ток утечки.
3. Отрицательный ток показывает, что он вытекает из вывода.

30.9 Емкостная нагрузка ввода/вывода

Эти параметры указывают на условия, при которых они влияют на характеристики работы каналов ввода/вывода по переменному току. Если нагрузка канала ввода/вывода отличается от указанной в таблицах, то вы должны определить как она влияет на временные характеристики. Меньшая емкость не должна влиять на параметры сигналов.

Таблица 30-9 Пример параметров по постоянному току

Характеристики по постоянному току			**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Коммерческий 0°C ≤ T_A ≤ +70°C Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C Параметры напряжения питания смотрите в таблице 30-3.				
№ пар.	**Обоз.**	**Описание**	**Мин.**	**Тип.****	**Макс.**	**Ед.**	**Примечание**
		Емкостная нагрузка на выходах					
D100	C_{OSC2}	Вывод OSC2	-	-	15	пФ	XT, HS, LP
D101	C_{IO}	Все каналы ввода/вывода и OSC2 в RC режиме	-	-	50	пФ	
D102	C_B	SCL, SDA в режиме I^2C	-	-	400	пФ	В режиме I^2C

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Примечания:

1. В RC режиме генератора на входе OSC1 включен триггер Шмидта. Не рекомендуется использовать внешний тактовый сигнал в RC режиме тактового генератора.
2. Ток утечки на выводе -MCLR зависит от приложенного напряжения. Параметры указаны для нормального режима работы. В других режимах может возникнуть больший ток утечки.
3. Отрицательный ток показывает, что он вытекает из вывода.

30.10 EEPROM память данных, FLASH память программ

Таблица 30-10 Пример параметров EEPROM памяти данных, FLASH памяти программ

Характеристики по постоянному току			**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Коммерческий 0°C ≤ T_A ≤ +70°C Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C Параметры напряжения питания смотрите в таблице 30-3.				
№ пар.	**Обоз.**	**Описание**	**Мин.**	**Тип.****	**Макс.**	**Ед.**	**Примечание**
		EEPROM память данных					
D120	E_D	Число циклов стирание/запись	1M	10M	-	C/3	5В @ 25°C
D121	V_{DRW}	Напряжение питания для записи/чтения	V_{MIN}	-	6.0	В	V_{MIN}- минимальное напряжение питания
D122	T_{DEW}	Время цикла стирание/запись	-	-	10	мс	
		FLASH память программ					
D130	E_P	Число циклов стирание/запись	100	1000	-	C/3	5В @ 25°C
D131	V_{PR}	Напряжение питания для чтения	V_{MIN}	-	6.0		V_{MIN}- минимальное
D132A	V_{PEW}	Напряжение питания для стирания/записи	4.5	-	5.5		напряжение питания
D133	T_{PEW}	Время цикла стирание/запись	-	-	10	мс	

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

30.11 LCD

Таблица 30-11 Пример электрических характеристик модуля LCD

Характеристики по постоянному току			**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Коммерческий 0°C ≤ T_A ≤ +70°C Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C Параметры напряжения питания смотрите в таблице 30-3.				
№ пар.	**Обоз.**	**Описание**	**Мин.**	**Тип.****	**Макс.**	**Ед.**	**Примечание**
D200	V_{LCD3}	Напряжение на выводе V_{LCD3}	V_{DD} - 0.3	-	V_{SS}+7.0	В	
D201	V_{LCD2}	Напряжение на выводе V_{LCD2}	-	-	V_{LCD3}	В	
D202	V_{LCD1}	Напряжение на выводе V_{LCD1}	-	-	V_{DD}	В	
D210	R_{COM}	Выходное сопротивление com выводов	-	-	1	кОм	COM выход
D211	R_{SEG}	Выходное сопротивление seg выводов	-	-	10	кОм	SEG выход
D220	V_{OH}	Выходное напряжение высокого уровня	Макс. V_{LCDN} - 0.1	-	Макс. V_{LCDN}	В	COM выхода I_{OH} = 25мкА SEG выхода I_{OH} = 3 мкА
D221	V_{OL}	Выходное напряжение низкого уровня	Мин. V_{LCDN}	-	Мин. V_{LCDN}+ 0.1	В	COM выхода I_{OH} = 25мкА SEG выхода I_{OH} = 3 мкА

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Примечание 1. Внутренне сопротивление V_{LCD} 0 Ом.

Таблица 30-12 Пример параметров генератора напряжения V_{LCD} модуля LCD

Характеристики по постоянному току			**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Коммерческий 0°C ≤ T_A ≤ +70°C Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C Параметры напряжения питания смотрите в таблице 30-3.				
№ пар.	**Обоз.**	**Описание**	**Мин.**	**Тип.****	**Макс.**	**Ед.**	**Примечание**
D250	I_{VADJ}	Выходной ток VLCDADJ	-	10	-	мкА	
D251	I_{VR}	Потребляемый ток VLCDADJ	-	-	20	мкА	
D252	$\frac{\Delta I_{VADJ}}{\Delta V_{DD}}$	Зависимость выходного тока от напряжения V_{DD}	-	-	0.1/1	мкА/В	
D253	$\frac{\Delta I_{VADJ}}{\Delta T}$	Зависимость выходного тока от температуры	-	-	0.1/70	мкА/°C	
D260(1)	R_{VADJ}	Внешний резистор VLCDADJ	100	-	230	кОм	
D265	V_{VADJ}	Диапазон напряжений VLCDADJ	1.0	-	2.3	В	
D271(1)	C_{ECPC}	Внешняя емкость генератора	-	0.5	-	мкФ	

Примечание 1. Оценочные значения.

30.12 Компараторы и источник опорного напряжения

Таблица 30-13 Пример характеристик компаратора по постоянному току

Характеристики по постоянному току			**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Коммерческий 0°C ≤ T_A ≤ +70°C Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C Параметры напряжения питания смотрите в таблице 30-3.				
№ пар.	**Обоз.**	**Описание**	**Мин.**	**Тип.****	**Макс.**	**Ед.**	**Примечание**
D300	V_{IOFF}	Входное напряжение смещения	-	±5.0	±10	мВ	
D301	V_{ICM}	Входное напряжение*	0	-	V_{DD}-1.5	В	
D302	CMRR	Коэффициент отражения*	55	-	-	db	
300	T_{RESP}	Время реакции (1)	-	150	400	нс	PIC16**C**XXX
300A			-	210	600	нс	PIC16**LC**XXX
301	T_{MC2OV}	Время смены режима*	-	-	10	мкс	

Примечание 1. Время реакции измерялось при напряжении на одном из входов (V_{DD}-1.5)/2 , а на другом был сформирован переход от V_{SS} к V_{DD}.

Таблица 30-13 Пример характеристик источника опорного напряжения по постоянному току

Характеристики по постоянному току			**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Коммерческий 0°C ≤ T_A ≤ +70°C Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C Параметры напряжения питания смотрите в таблице 30-3.				
№ пар.	**Обоз.**	**Описание**	**Мин.**	**Тип.****	**Макс.**	**Ед.**	**Примечание**
D310	V_{RES}	Разрешающая способность	V_{DD}/24	-	V_{DD}/32	Lsb	
D311	VR_{AA}	Абсолютная точность	-	-	1/4	Lsb	VRR=1
			-	-	1/2	Lsb	VRR=0
D312	VR_{UR}	Сопротивление резистора R*	-	2	-	кОм	
310	T_{SET}	Время установки(1)	-	-	10	мкс	

Примечание 1. Время измерено при VRR=1 и переходе VR<3:0> от 0000 к 1111.

30.13 Символьное обозначение временных параметров

Символьное обозначение временных параметров имеет один из следующих форматов:

1. TppS2ppS
2. TppS
3. $T_{CC:ST}$ (только спецификация I^2C)
4. T_S (только спецификация I^2C)

T			
F	Частота	T	Время

Строчные символы (pp) и их значение

pp			
cc	CCP1	osc	OSC1
ck	CLKOUT	rd	-RD
cs	-CS	rw	-RD или -WR
di	SDI	sc	SCK
do	SDO	ss	-SS
dt	Входные данные	t0	T0CKI
io	Канал ввода/вывода	t1	T1CKI
mc	-MCLR	wr	-WR

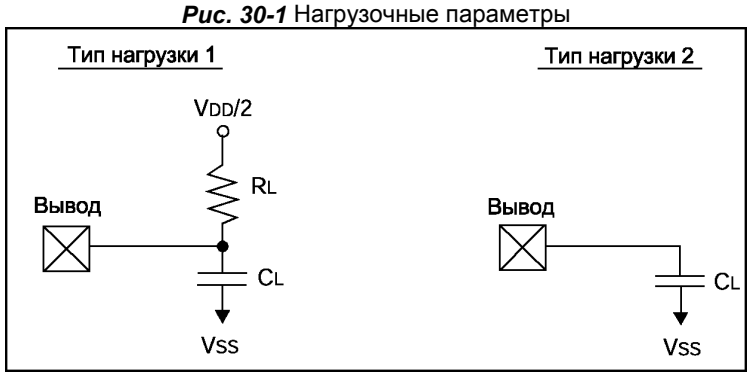
Прописные символы и их значение

S			
F	Задний фронт	P	Период
H	Высокий уровень	R	Передний фронт
I	Неверный (3-е состояние)	V	Верный
L	Низкий уровень	Z	3-е состояние
Только I^2C			
AA	Доступ вывода	High	Высокий уровень
BUF	Шина свободна	Low	Низкий уровень

$T_{CC:ST}$ (только спецификация I^2C)

CC			
HD	Удержание	SU	Установка
ST			
DAT	Сохранение данных на входе	STO	Условие STOP
STA	Условие START	Low	Низкий уровень

Рис. 30-1 Нагрузочные параметры



R_L = 464Ом
C_L = 50пФ (для всех выводов, кроме OSC2)
C_L = 15пФ (для вывода OSC2)

30.14 Пример временных диаграмм и параметров тактового сигнала

Рис. 30-2 Пример временной диаграммы внешнего тактового сигнала

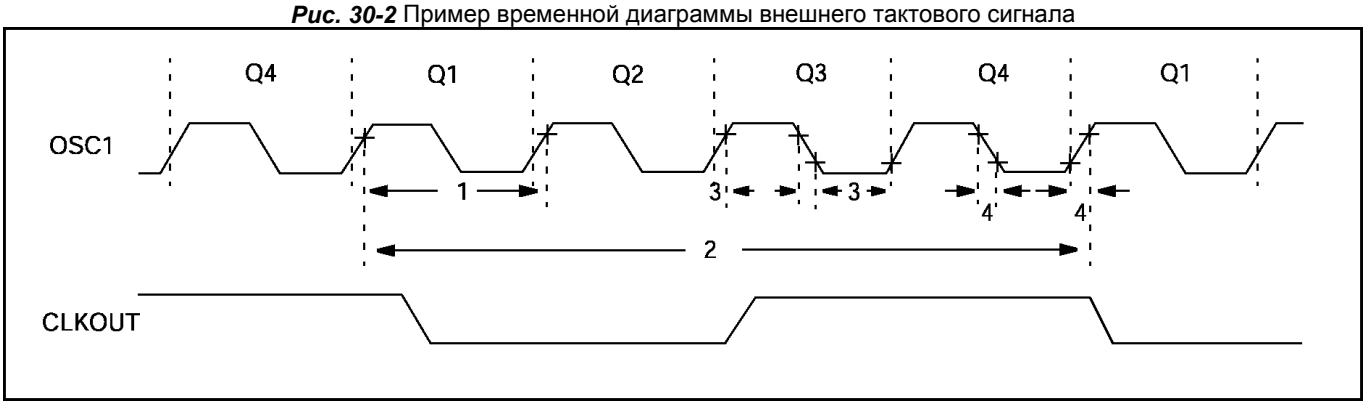


Таблица 30-15 Пример параметров внешнего тактового сигнала

№ пар.	Обоз.	Описание	Мин.	Тип.**	Макс.	Ед.	Примечание
	F_{OSC}	Частота внешнего тактового сигнала(1)	DC	-	4	МГц	XT, RC PIC16**C**XXX-**04**
							PIC16**LC**XXX-**04**
			DC	-	10	МГц	HS PIC16**C**XXX-**10**
			DC	-	20	МГц	HS PIC16**C**XXX-**20**
			Dc	-	200	кГц	LP PIC16**LC**XXX-**04**
		Частота генератора(1)	DC	-	4	МГц	RC PIC16**C**XXX-**04**
							PIC16**LC**XXX-**04**
			0.1	-	4	МГц	XT PIC16**C**XXX-**04**
							PIC16**LC**XXX-**04**
			4	-	10	МГц	HS PIC16**C**XXX-**10**
			4	-	20	МГц	HS PIC16**C**XXX-**20**
			5	-	200	кГц	LP PIC16**LC**XXX-**04**
1	T_{OSC}	Период внешнего тактового сигнала(1)	250	-	-	нс	XT, RC PIC16**C**XXX-**04**
							PIC16**LC**XXX-**04**
			100	-	-	нс	HS PIC16**C**XXX-**10**
			50	-	-	нс	HS PIC16**C**XXX-**20**
			5	-	-	мкс	LP PIC16**LC**XXX-**04**
		Период генератора(1)	250	-	-	нс	RC PIC16**C**XXX-**04**
							PIC16**LC**XXX-**04**
			250	-	10000	нс	XT PIC16**C**XXX-**04**
							PIC16**LC**XXX-**04**
			100	-	250	нс	HS PIC16**C**XXX-**10**
			50	-	250	нс	HS PIC16**C**XXX-**20**
			5	-	-	мкс	LP PIC16**LC**XXX-**04**
2	T_{CY}	Время выполнения инструкции(1)	200.0	T_{CY}	DC	нс	$T_{CY} = 4/F_{OSC}$
3	$T_{OS}L$, $T_{OS}H$	Длительность высокого/низкого уровня CLKIN (OSC1)	50	-	-	нс	XT PIC16**C**XXX-**04**
			60	-	-	нс	XT PIC16**LC**XXX-**04**
			2.5	-	-	мкс	LP PIC16**LC**XXX-**04**
			15	-	-	нс	HS PIC16**C**XXX-**20**
4	TosR, TosF	Длительность переднего/заднего фронта внешнего тактового сигнала (OSC1)	-	-	25	нс	XT PIC16**C**XXX-**04**
			-	-	50	нс	LP PIC16**LC**XXX-**04**
			-	-	15	нс	HS PIC16**C**XXX-**20**

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Примечание 1. Машинный цикл микроконтроллера равняется 4 периодам тактового сигнала. Все приведенные значения основываются на характеристиках конкретного типа генератора в стандартных условиях при выполнении программы. Выход за указанные пределы может привести к нестабильной работе генератора и/или к большему потребляемому току. Все микроконтроллеры проверены в режиме "Мин." при внешнем тактовом сигнале на выводе OSC1/CLKIN.

Рис. 30-3 Пример временной диаграммы CLKOUT и каналов ввода/вывода

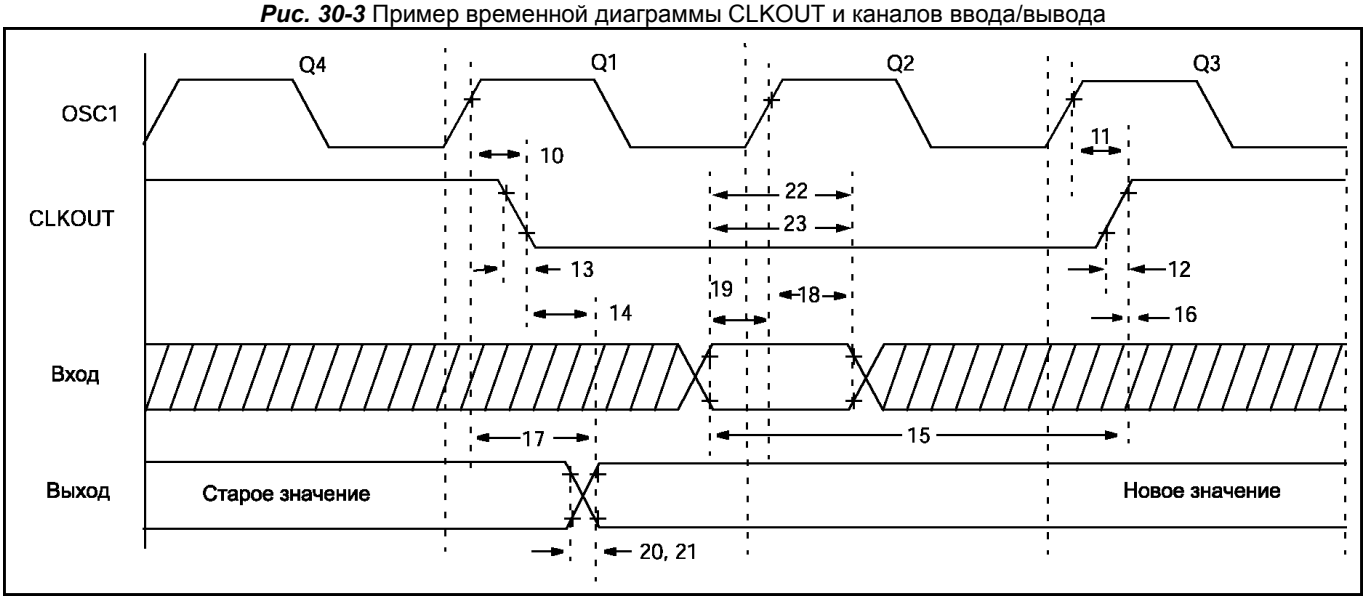


Таблица 30-16 Пример параметров CLKOUT и каналов ввода/вывода

№ пар.	Обоз.	Описание		Мин.	Тип.**	Макс.	Ед.	Примечание
10	TosH2ckL	От OSC1 ↑ до CLKOUT ↓		-	75	200	нс	(1)
11	TosH2ckH	От OSC1 ↑ до CLKOUT ↑		-	75	200	нс	(1)
12	TckR	CLKOUT длит. переднего фронта		-	35	100	нс	(1)
13	TckF	CLKOUT длит. заднего фронта		-	35	100	нс	(1)
14	TckL2ioV	От CLKOUT ↓ до установл. выхода		-	-	$0.5T_{CY}$+20	нс	(1)
15	TioV2ckH	От установл. входа до CLKOUT ↑		$0.25T_{CY}$+25	-	-	нс	(1)
16	TckH2ioI	Удержание входа после CLKOUT ↑		0	-	-	нс	(1)
17	TosH2ioV	От OSC1 ↑ до установл. выхода		-	50	150	нс	
18	TosH2ioI	Удержание входа после OSC1 ↑	**C**	100	-	-	нс	
18A			**LC**	200	-	-	нс	
19	TioV2osH	Переход в режим входа относ. OSC1↑		0	-	-	нс	
20	TioR	Длительность переднего фронта на выходе порта ввода/вывода	**C**	-	10	25	нс	
20A			**LC**	-	-	60	нс	
21	TioF	Длительность заднего фронта на выходе порта ввода/вывода	**C**	-	10	25	нс	
21A			**LC**	-	-	60	нс	
22***	Tinp	Длит. высокого/низкого уровня INT		T_{CY}	-	-	нс	
23***	Trbp	Длит. высокого/низкого уровня RB7:RB4		T_{CY}	-	-	нс	
24***	Trcp	Длит. высокого/низкого уровня RC7:RC4		20	-	-	нс	

* - Эти параметры определены, но не протестированы.

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

*** - Асинхронные события, не связанные с внутренним тактовым сигналом.

Примечание 1. Измерения проведены в RC режиме генератора, где CLKOUT = 4 x T_{OSC}.

30.15 Пример временных диаграмм и параметров сброса микроконтроллера

Рис. 30-4 Пример временной диаграммы сброса, WDT, OST, PWRT

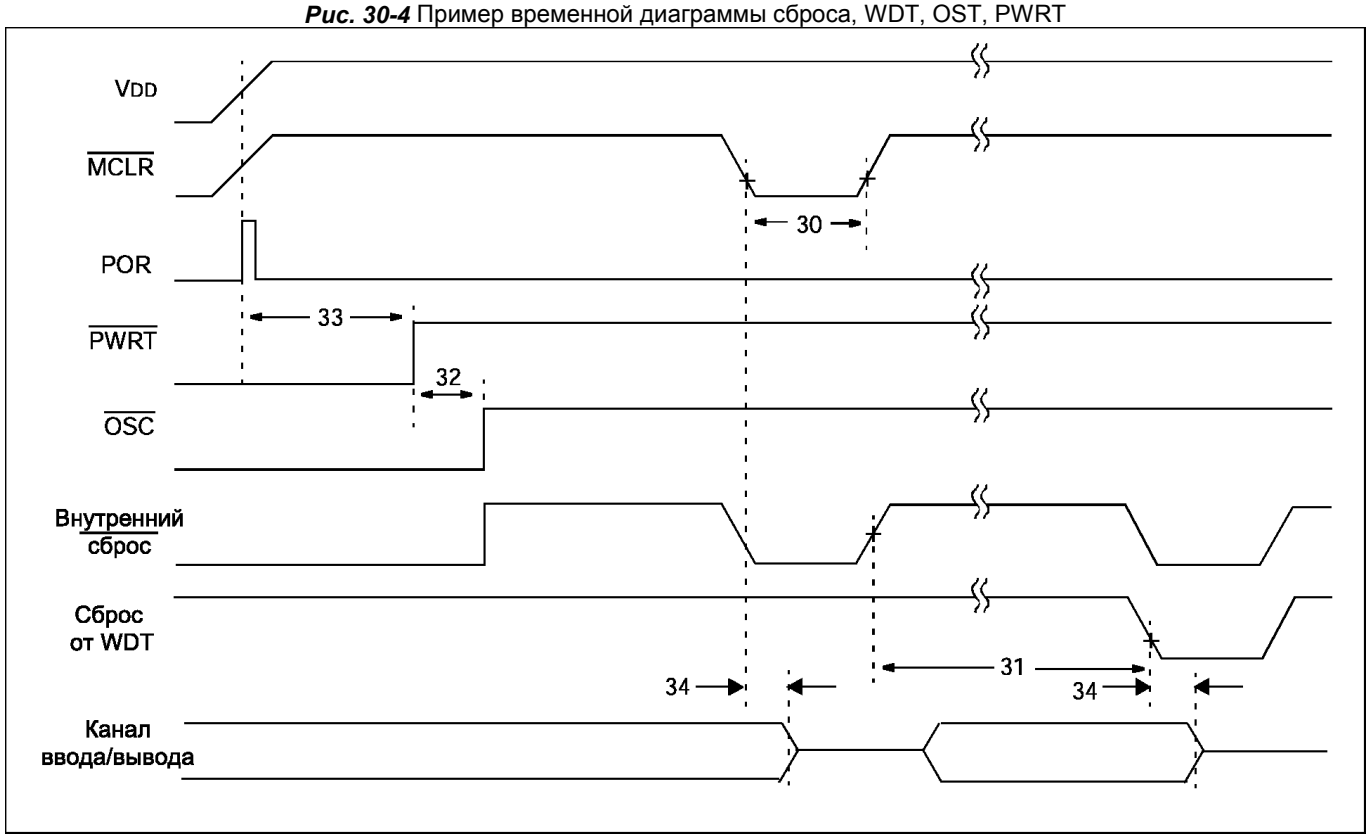


Рис. 30-5 Пример временной диаграммы работы BOD

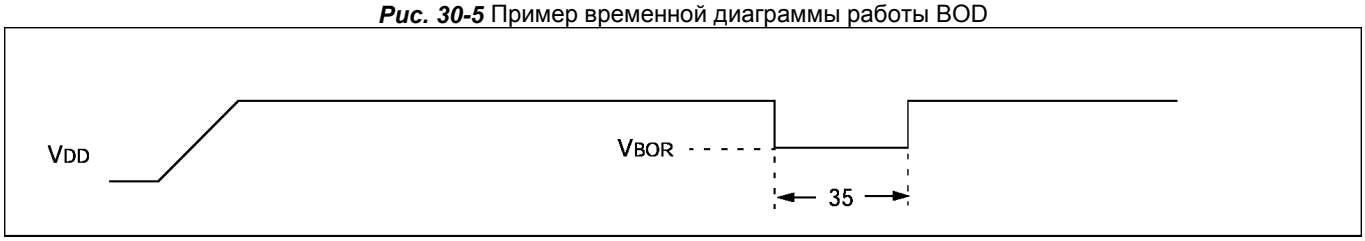


Таблица 30-17 Пример параметров сброса, WDT, OST, PWRT, BOR

№ пар.	Обоз.	Описание	Мин.	Тип.**	Макс.	Ед.	Примечание
30	TmcL	Длительность импульса -MCLR	2	-	-	мкс	V_{DD}=5В,-40°C до +85°C
31	Twdt	Период переполнения WDT (без предделителя)	7	18	33	мс	V_{DD}=5В,-40°C до+125°C
32	Tost	Период OST	-	$1024T_{OSC}$	-	-	T_{OSC} = период OSC1
33	Tpwrt	Период PWRT	28	72	132	мс	V_{DD}=5В,-40°C до +85°C
34	T_{IOZ}	От сброса -MCLR или WDT до перевода каналов ввода/вывода 3-е состояние	-	-	2.1	мкс	
35	T_{BOR}	Длительность импульса BOR	100	-	-	мкс	$V_{DD} \leq V_{BOR}$ (D005)

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

30.16 Пример временных диаграмм и параметров внешнего тактового сигнала для TMR0 и TMR1

Рис. 30-6 Пример временной диаграммы внешнего тактового сигнала для TMR0 и TMR1

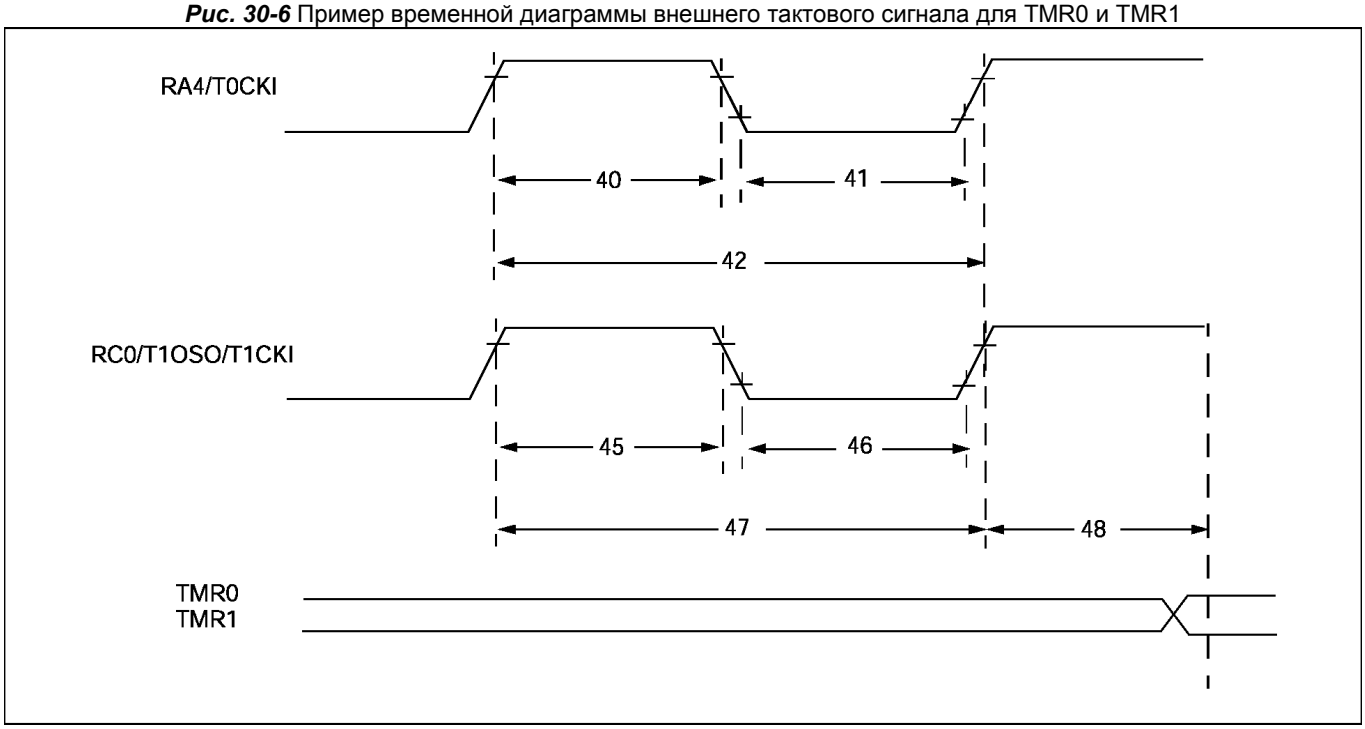


Таблица 30-18 Пример параметров внешнего тактового сигнала для TMR0 и TMR1

№ пар.	Обоз.	Описание			Мин.	Тип.**	Макс.	Ед.	Примечание
40	Tt0H	Длительность высокого уровня T0CKI	Без предделителя		$0.5T_{CY}+20$	-	-	нс	Также должен выполняться параметр 42
			С предделителем		10	-	-	нс	
41	Tt0L	Длительность низкого уровня T0CKI	Без предделителя		$0.5T_{CY}+20$	-	-	нс	
			С предделителем		10	-	-	нс	
42	Tt0P	Период T0CKI	Без предделителя		$T_{CY}+40$	-	-	нс	
			С предделителем		20 или $(T_{CY}+40)/N$	-	-	нс	N = коэфф.предд.
45	Tt1H	Длительность высокого уровня T1CKI	Синхр.реж. без преддел.		$0.5T_{CY}+20$	-	-	нс	Также должен выполняться параметр 47
			Синхр. режим с преддел.	**C**	15	-	-	нс	
				LC	25	-	-	нс	
			Асинхронный режим	**C**	30	-	-	нс	
				LC	50	-	-	нс	
46	Tt1L	Длительность низкого уровня T1CKI	Синхр.реж. без преддел.		$0.5T_{CY}+20$	-	-	нс	Также должен выполняться параметр 47
			Синхр. режим с преддел.	**C**	15	-	-	нс	
				LC	25	-	-	нс	
			Асинхронный режим	**C**	30	-	-	нс	
				LC	50	-	-	нс	
47	Tt1P	Период T1CKI	Синхронный режим	**C**	30 или $(T_{CY}+40)/N$	-	-	нс	N = коэфф.предд.
				LC	50 или $(T_{CY}+40)/N$	-	-	нс	N = коэфф.предд.
			Асинхронный режим	**C**	60	-	-	нс	
				LC	100	-	-	нс	
	Ft1	Частота резонатора для TMR1 (T1OSCEN=1)			DC	-	200	кГц	
48	TCKE1	Задержка от активного фронта тактового сигнала до приращения TMR1			$2T_{OSC}$	-	$7T_{OSC}$	-	

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

30.17 Пример временных диаграмм и параметров модуля CCP

Рис. 30-7 Пример временной диаграммы захват/сравнение/ШИМ

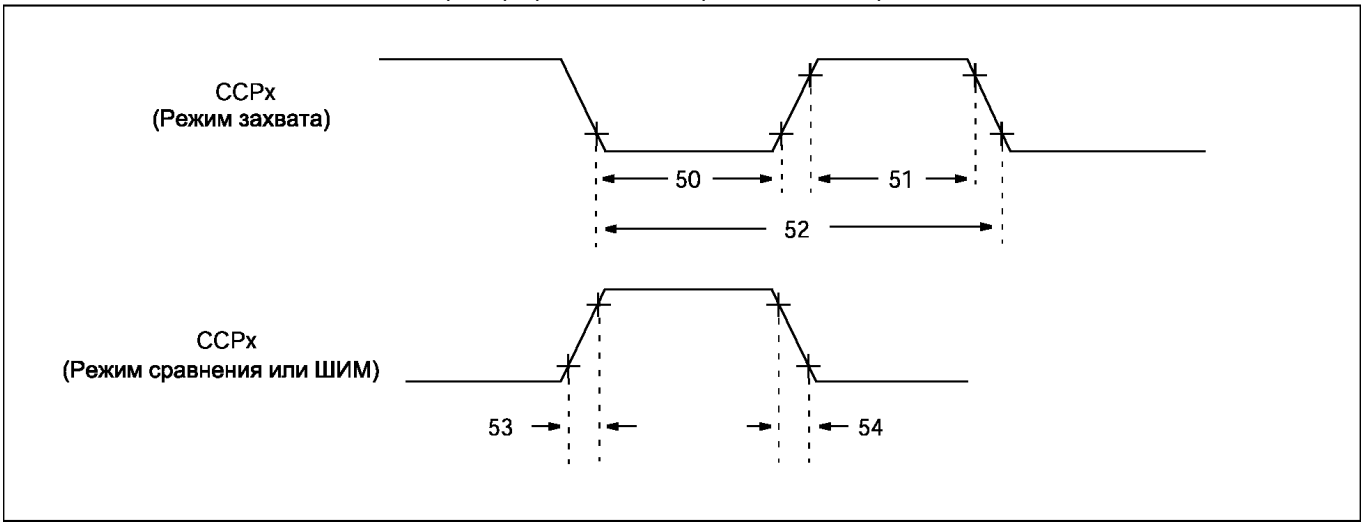


Таблица 30-9 Пример параметров захват/сравнение/ШИМ (CCP1 и CCP2)

№ пар.	Обоз.	Описание			Мин.	Тип.**	Макс.	Ед.	Примечание
50*	TccL	Сигнал низкого уровня CCPx	Без предделителя		$0.5T_{CY}$+20	-	-	нс	
			С предделителем	**C**	10	-	-	нс	
				LC	20	-	-	нс	
51*	TccL	Сигнал высокого уровня CCPx	Без предделителя		$0.5T_{CY}$+20	-	-	нс	
			С предделителем	**C**	10	-	-	нс	
				LC	20	-	-	нс	
52*	TccP	Период входного сигнала CCPx			$(3T_{CY}+40)/N$	-	-	нс	N = коэфф.предд.
53*	TccR	Время установление высокого уровня сигн. на вых. CCPx		**C**	-	10	25	нс	
				LC	-	25	45	нс	
54*	TccF	Время установление низкого уровня сигн. на вых. CCPx		**C**	-	10	25	нс	
				LC	-	25	45	нс	

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

30.18 Пример временных диаграмм и параметров ведомого параллельного порта

Рис. 30-8 Пример временной диаграммы работы ведомого параллельного порта

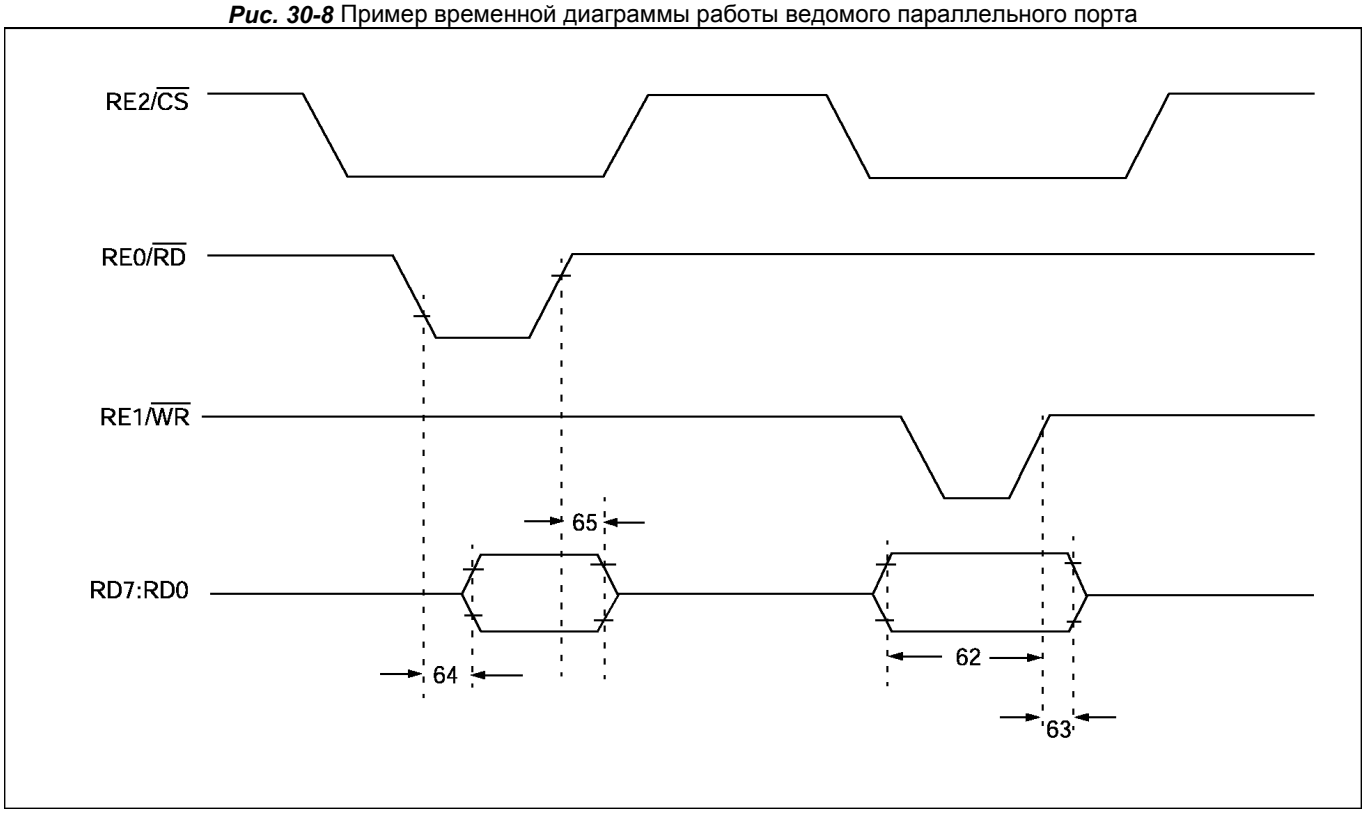


Таблица 30-20 Пример параметров работы ведомого параллельного порта

№ пар.	Обоз.	Описание		Мин.	Тип.**	Макс.	Ед.	Примечание
62	TdtV2H	Установка данных перед -WR↑ или -CS↑		20 25	- -	- -	нс нс	 Только для расшир. диап.
63	TwrH2dtl	Удержание данных после -WR↑ или -CS↑	**C**	20	-	-	нс	
			LC	35	-	-	нс	
64	TrdL2dtV	Формирование данных после -RD↓ и -CS↓		- -	- -	80 90	нс нс	 Только для расшир. диап.
65	TrdH2dtl	Неправильные данные после -RD↑ или -CS↑		10	-	30	нс	
66	TibfINH	Запрет очистки флага IBF после -WR↑ или -CS↑		-	-	$3T_{CY}^{+}$		

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

$^{+}$ - Оценочное значение.

30.19 Пример временных диаграмм и параметров модуля SSP и MSSP в режиме SPI

Рис. 30-9 Пример временной диаграммы работы в режиме ведущего SPI (CKE=0)

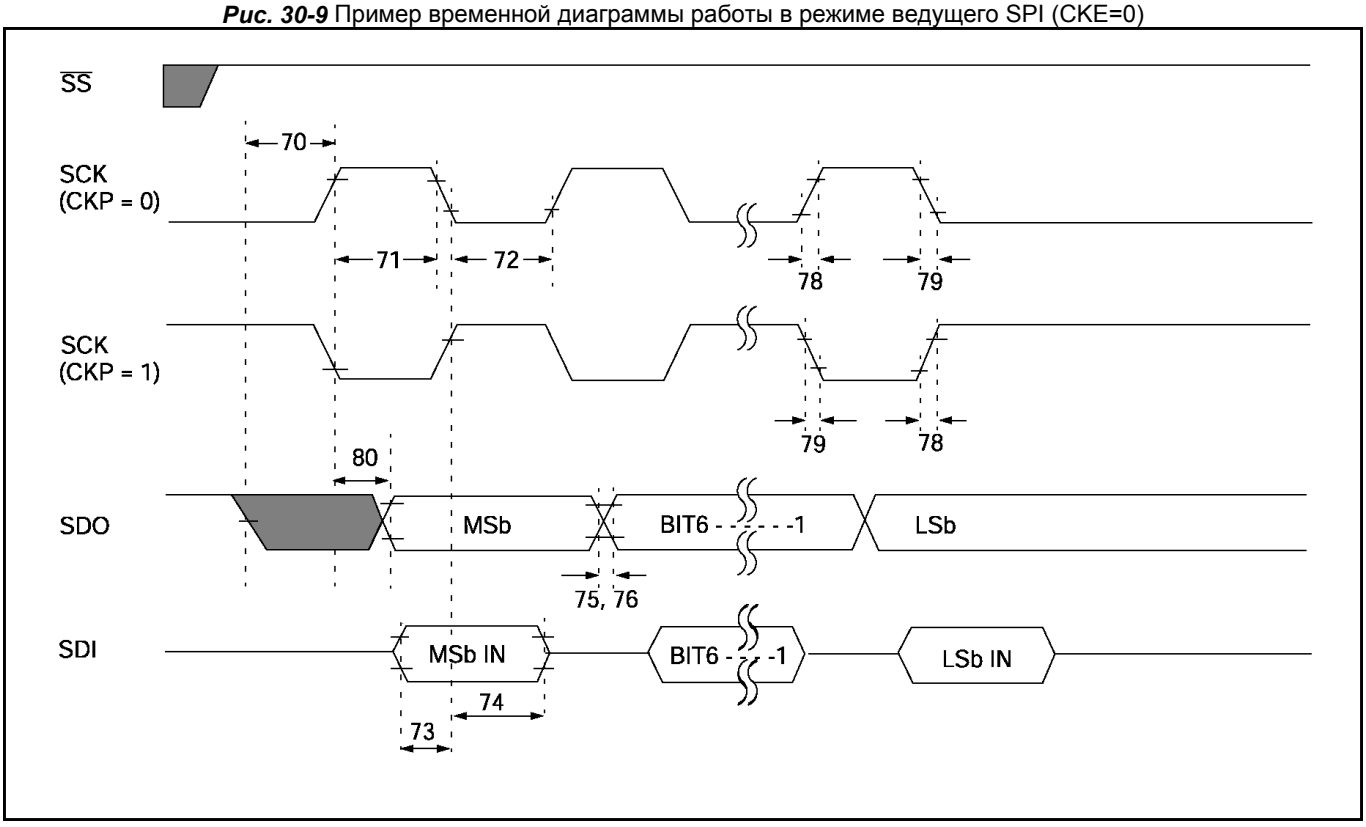


Таблица 30-21 Пример параметров работы в режиме ведущего SPI (CKE=0)

№ пар.	Обоз.	Описание		Мин.	Тип.**	Макс.	Ед.	Примечание
70	TssL2scH, TssL2scL	-SS↓ перед SCK↑ или SCK↓		T_{CY}	-	-	нс	
71	TscH	Высокий уров. сигн. SCK	Непрерыван.	$1.25T_{CY}$ + 30	-	-	нс	
71A			Одиночный	40	-	-	нс	(1)
72	TscL	Низкий уров. сигн. SCK	Непрерыван.	$1.25T_{CY}$ + 30	-	-	нс	
72A			Одиночный	40	-	-	нс	(1)
73	TdiV2scH, TdiV2scL	Установка данных на входе SDI относительно фронта SCK		100	-	-	нс	
73A	T_B2_B	От послед. фронта байта 1 до перв. фронта байта 2		$1.25T_{CY}$ + 30	-	-	нс	(1)
74	TscH2diL, TscL2diL	Удержание данных на входе SDI относительно фронта SCK		100	-	-	нс	
75	TdoR	Длительность переднего фронта на выходе SDO	**C**	-	10	25	нс	
			LC	-	25	45	нс	
76	TdoF	Длительность заднего фронта на SDO		-	10	25	нс	
78	TscR	Длительность переднего фронта на SCK	**C**	-	10	25	нс	
			LC	-	25	45	нс	
79	TscF	Длит. заднего фронта на SCK (ведущий)		-	10	25	нс	
80	TscH2doV, TscL2doV	Достоверные данные на SDO после фронта SCK	**C**	-	-	50	нс	
			LC	-	-	100	нс	

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Примечание 1. Необходимо учитывать параметр 73A только, если используются параметры 71A и 72A.

Рис. 30-10 Пример временной диаграммы работы в режиме ведущего SPI (CKE=1)

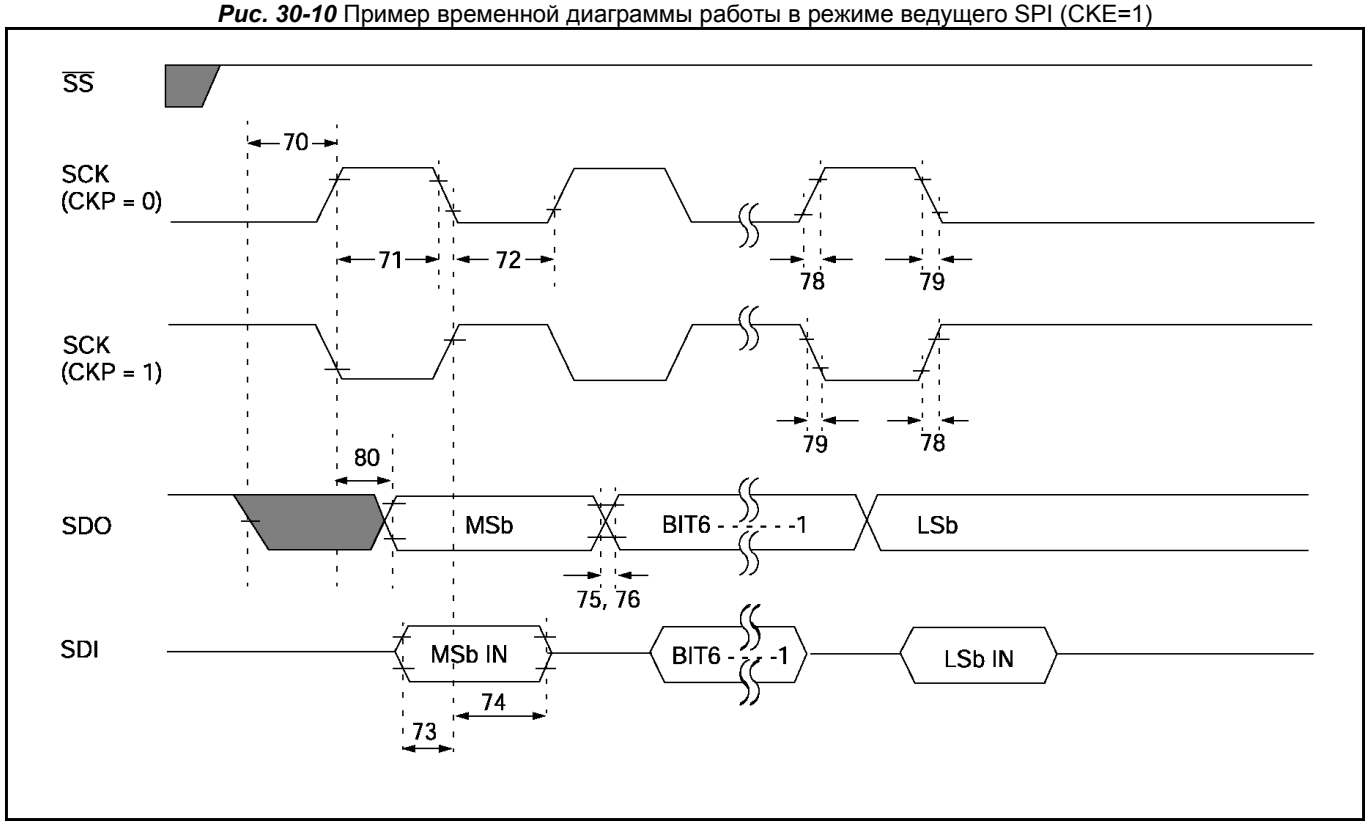


Таблица 30-21 Пример параметров работы в режиме ведущего SPI (CKE=0)

№ пар.	Обоз.	Описание		Мин.	Тип.**	Макс.	Ед.	Примечание
71	TscH	Высокий уров. сигн. SCK	Непрерыван.	$1.25T_{CY}$ + 30	-	-	нс	
71A			Одиночный	40	-	-	нс	(1)
72	TscL	Низкий уров. сигн. SCK	Непрерыван.	$1.25T_{CY}$ + 30	-	-	нс	
72A			Одиночный	40	-	-	нс	(1)
73	TdiV2scH, TdiV2scL	Установка данных на входе SDI относительно фронта SCK		100	-	-	нс	
73A	T_{B2B}	От послед. фронта байта 1 до перв. фронта байта 2		$1.25T_{CY}$ + 30	-	-	нс	(1)
74	TscH2diL, TscL2diL	Удержание данных на входе SDI относительно фронта SCK		100	-	-	нс	
75	TdoR	Длительность переднего фронта на выходе SDO	**C**	-	10	25	нс	
			LC	-	25	45	нс	
76	TdoF	Длительность заднего фронта на SDO		-	10	25	нс	
78	TscR	Длительность переднего фронта на SCK	**C**	-	10	25	нс	
			LC	-	25	45	нс	
79	TscF	Длит. заднего фронта на SCK (ведущий)		-	10	25	нс	
80	TscH2doV, TscL2doV	Достоверные данные на SDO после фронта SCK	**C**	-	-	50	нс	
			LC	-	-	100	нс	
81	TdoV2scH, TdoV2scL	Установка данных на выходе SDO после фронта SCK		T_{CY}	-	-	нс	

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Примечание 1. Необходимо учитывать параметр 73A только, если используются параметры 71A и 72A.

Рис. 30-11 Пример временной диаграммы работы в режиме ведомого SPI (CKE=0)

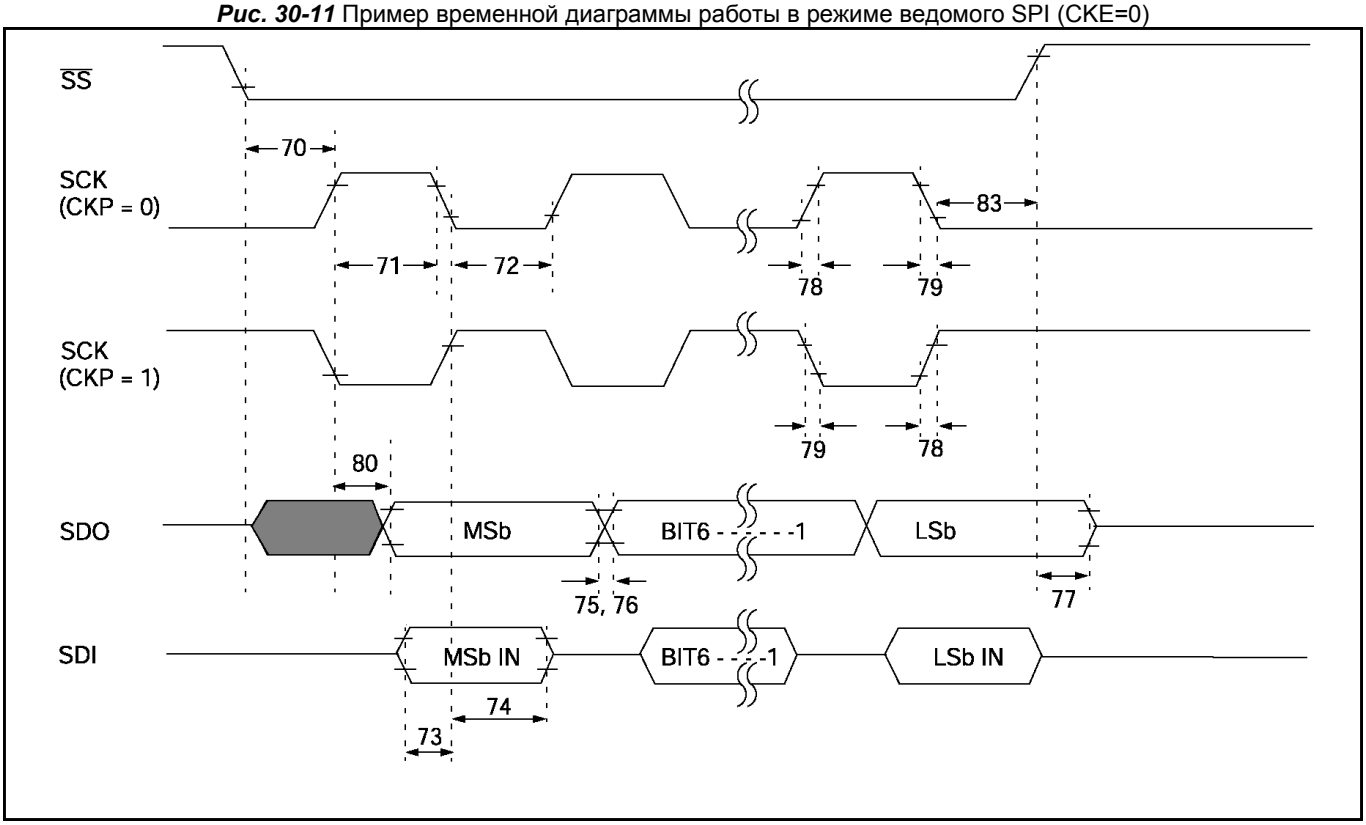


Таблица 30-23 Пример параметров работы в режиме ведомого SPI (CKE=0)

№ пар.	Обоз.	Описание		Мин.	Тип.**	Макс.	Ед.	Примечание
70	TssL2scH, TssL2scL	-SS↓ перед SCK↑ или SCK↓		T_{CY}	-	-	нс	
71	TscH	Высокий уров. сигн. SCK	Непрерыван.	$1.25T_{CY}$ + 30	-	-	нс	
71A			Одиночный	40	-	-	нс	(1)
72	TscL	Низкий уров. сигн. SCK	Непрерыван.	$1.25T_{CY}$ + 30	-	-	нс	
72A			Одиночный	40	-	-	нс	(1)
73	TdiV2scH, TdiV2scL	Установка данных на входе SDI относительно фронта SCK		100	-	-	нс	
73A	$T_{B}2_{B}$	От послед. фронта байта 1 до перв. фронта байта 2		$1.25T_{CY}$ + 30	-	-	нс	(1)
74	TscH2diL, TscL2diL	Удержание данных на входе SDI относительно фронта SCK		100	-	-	нс	
75	TdoR	Длительность переднего фронта на выходе SDO	**C**	-	10	25	нс	
			LC	-	25	45	нс	
76	TdoF	Длительность заднего фронта на SDO		-	10	25	нс	
78	TscR	Длительность переднего фронта на SCK	**C**	-	10	25	нс	
			LC	-	25	45	нс	
79	TscF	Длит. заднего фронта на SCK (ведущий)		-	10	25	нс	
80	TscH2doV, TscL2doV	Достоверные данные на SDO после фронта SCK	**C**	-	-	50	нс	
			LC	-	-	100	нс	
83	TscH2ssH, TscL2ssH	SS↑ после фронта SCK		$1.5T_{CY}$ + 40	-	-	нс	

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Примечание 1. Необходимо учитывать параметр 73A только, если используются параметры 71A и 72A.

Рис. 30-12 Пример временной диаграммы работы в режиме ведомого SPI (CKE=1)

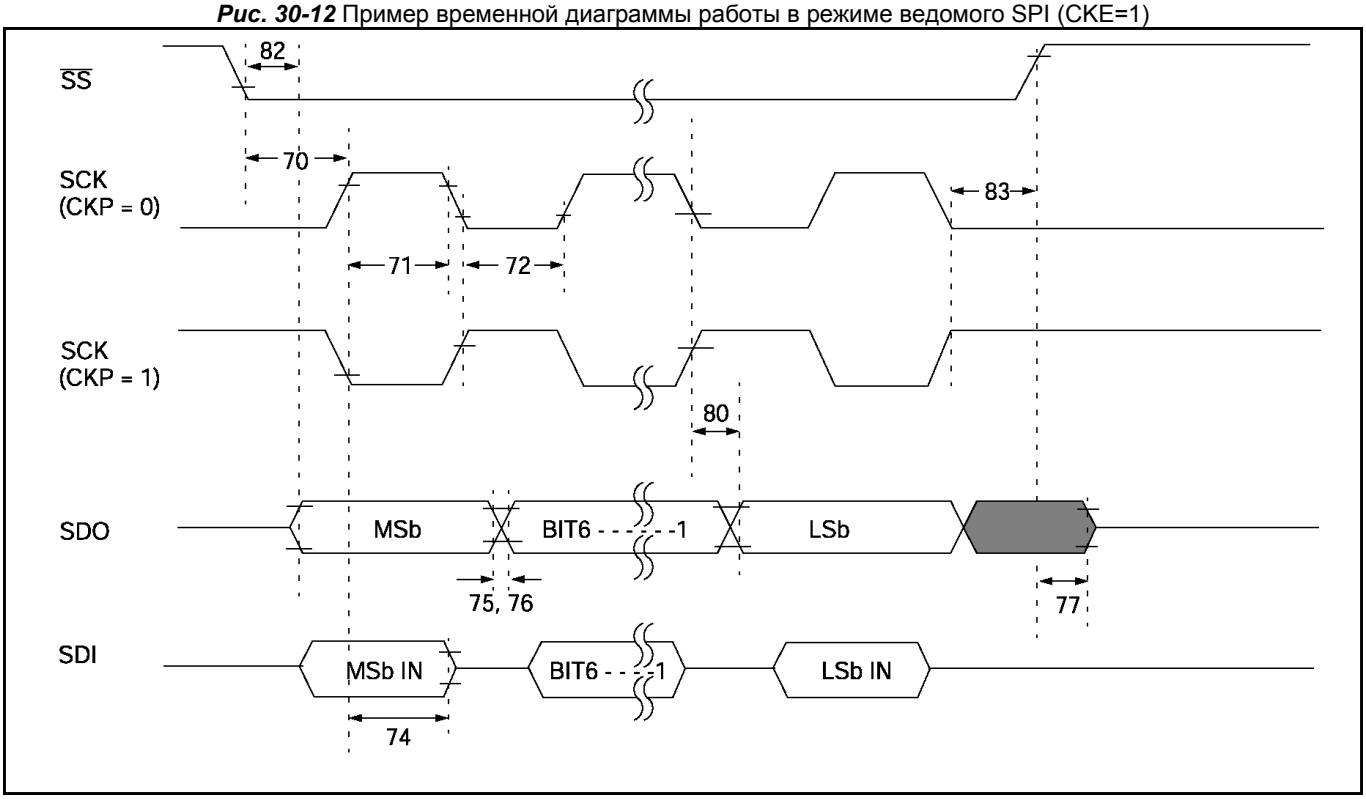


Таблица 30-24 Пример параметров работы в режиме ведомого SPI (CKE=1)

№ пар.	Обоз.	Описание		Мин.	Тип.**	Макс.	Ед.	Примечание
70	TssL2scH, TssL2scL	-SS↓ перед SCK↑ или SCK↓		T_{CY}	-	-	нс	
71 71A	TscH	Высокий уров. сигн. SCK	Непрерыван. Одиночный	$1.25T_{CY}$ + 30 40	- -	- -	нс нс	 (1)
72 72A	TscL	Низкий уров. сигн. SCK	Непрерыван. Одиночный	$1.25T_{CY}$ + 30 40	- -	- -	нс нс	 (1)
73	TdiV2scH, TdiV2scL	Установка данных на входе SDI относительно фронта SCK		100	-	-	нс	
73A	T_B2_B	От послед. фронта байта 1 до перв. фронта байта 2		$1.25T_{CY}$ + 30	-	-	нс	(1)
74	TscH2diL, TscL2diL	Удержание данных на входе SDI относительно фронта SCK		100	-	-	нс	
75	TdoR	Длительность переднего фронта на выходе SDO	**C** **LC**	- -	10 25	25 45	нс нс	
76	TdoF	Длительность заднего фронта на SDO		-	10	25	нс	
78	TscR	Длительность переднего фронта на SCK	**C** **LC**	- -	10 25	25 45	нс нс	
79	TscF	Длит. заднего фронта на SCK (ведущий)		-	10	25	нс	
80	TscH2doV, TscL2doV	Достоверные данные на SDO после фронта SCK	**C** **LC**	- -	- -	50 100	нс нс	
82	TssL2doV	Достов. данные на вых. SDO после SS↓	**C** **LC**	- -	- -	50 100	нс нс	
83	TscH2ssH, TscL2ssH	SS↑ после фронта SCK		$1.5T_{CY}$ + 40	-	-	нс	

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Примечание 1. Необходимо учитывать параметр 73A только, если используются параметры 71A и 72A.

30.20 Пример временных диаграмм и параметров модуля SSP в режиме I²C

Рис. 30-13 Пример временной диаграммы формирования битов START/STOP на шине I²C

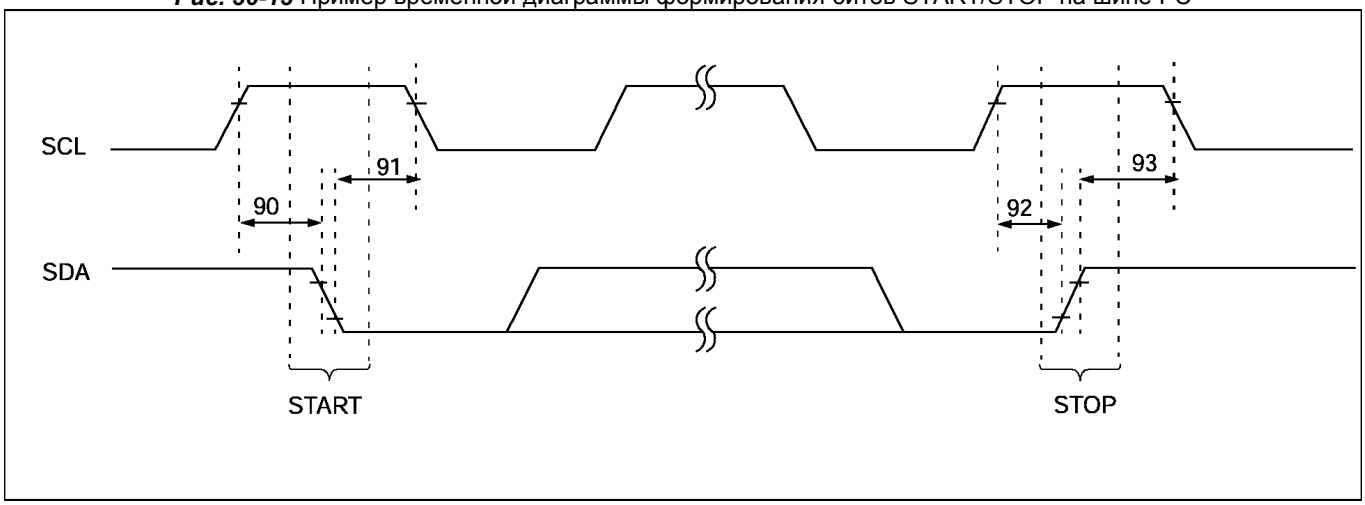


Таблица 30-25 Пример параметров формирования битов START/STOP на шине I²C

№ пар.	Обоз.	Описание		Мин.	Тип.**	Макс.	Ед.	Примечание
90	Tsu:sta	Установка условия START	Режим 100 кГц	4700	-	-	нс	Только при формировании бита повторный START
			Режим 400 кГц	600	-	-		
91	Thd:sta	Удержание условия START	Режим 100 кГц	4000	-	-	нс	После этого форм. первый импульс тактового сигнала
			Режим 400 кГц	600	-	-		
92	Tsu:sto	Установка условия STOP	Режим 100 кГц	4700	-	-	нс	
			Режим 400 кГц	600	-			
93	Thd:sto	Удержание условия STOP	Режим 100 кГц	4000	-	-	нс	
			Режим 400 кГц	600	-	-		

Рис. 30-14 Пример временной диаграммы формирования бита данных на шине I^2C

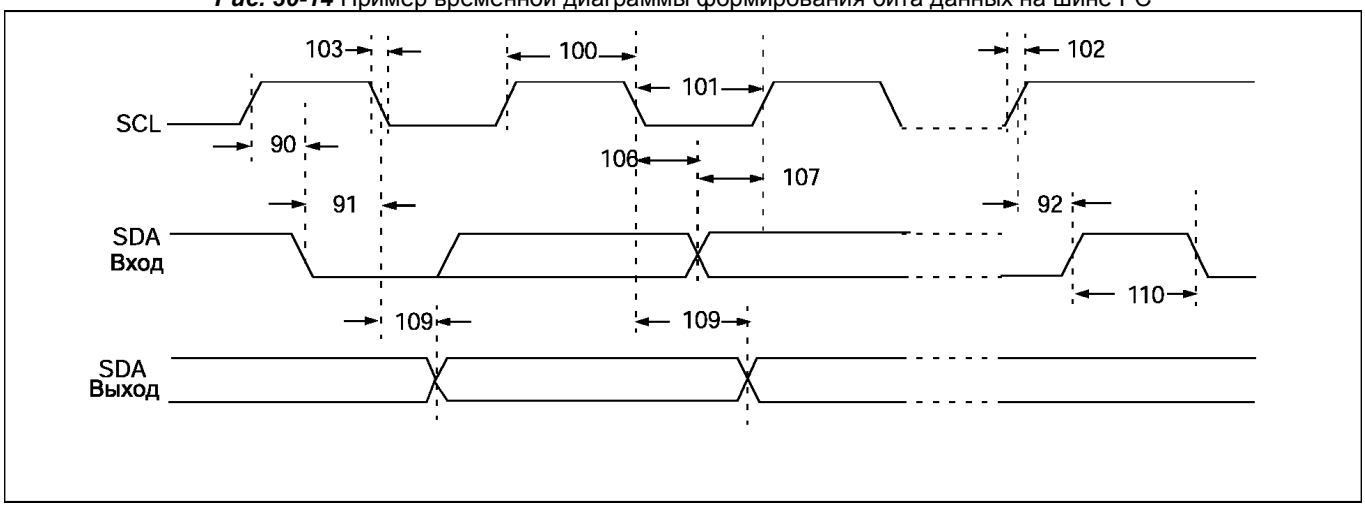


Таблица 30-26 Пример параметров формирования бита данных на шине I^2C

№ пар.	Обоз.	Описание		Мин.	Макс.	Ед.	Примечание
100	Thigh	Длительность высокого уровня тактового сигнала	Режим 100 кГц	4.0	-	мкс	Мин. F_{OSC} 1.5МГц
			Режим 400 кГц	0.6	-	мкс	Мин. F_{OSC} 10МГц
			Модуль SSP	$1.5T_{CY}$	-		
101	Tlow	Длительность низкого уровня тактового сигнала	Режим 100 кГц	4.7	-	мкс	Мин. F_{OSC} 1.5МГц
			Режим 400 кГц	1.3	-	мкс	Мин. F_{OSC} 10МГц
			Модуль SSP	$1.5T_{CY}$	-		
102	Tr	Долит. переднего фронта на SDA и SCL	Режим 100 кГц	-	1000	нс	
			Режим 400 кГц	20 + 0.1 Cb	300	нс	10пФ ≤ Cb ≤ 400пФ
103	Tf	Долит. заднего фронта на SDA и SCL	Режим 100 кГц	-	300	нс	
			Режим 400 кГц	20 + 0.1 Cb	300	нс	10пФ ≤ Cb ≤ 400пФ
90	Tsu:sta	Установка условия START	Режим 100 кГц	4.7	-	мкс	Только при формировании бита повторный START
			Режим 400 кГц	0.6	-	мкс	
91	Thd:sta	Удержание условия START	Режим 100 кГц	4.0	-	мкс	После этого форм. первый импульс тактового сигнала
			Режим 400 кГц	0.6	-	мкс	
106	Thd:dat	Удержание данных на входе	Режим 100 кГц	0	-	нс	
			Режим 400 кГц	0	0.9	мкс	
107	Tsu:dat	Установка данных на входе	Режим 100 кГц	250	-	нс	Примечание 2
			Режим 400 кГц	100	-	нс	
92	Tsu:sto	Установка условия STOP	Режим 100 кГц	4.7	-	мкс	
			Режим 400 кГц	0.6	-	мкс	
109	Taa	Достоверность сигнала на выходе	Режим 100 кГц	-	3500	нс	Примечание 1
			Режим 400 кГц	-	-	нс	
110	Tbuf	Время не занятости шины	Режим 100 кГц	4.7	-	мкс	Задержка перед новой передачей
			Режим 400 кГц	1.3	-	мкс	
	Cb	Емкостная нагрузка линии		-	400	пФ	

Примечания:

1. Необходимо выдерживать эту минимальную задержку относительно заднего фронта SCL, чтобы избежать ложное формирование битов START и STOP.
2. Устройства с высокоскоростным режимом обмена (400кГц) могут использоваться в стандартном режиме (100кГц), но требование Tsu:dat ≥ 250нс необходимо выполнять. Это условие автоматически будет выполняться, если не возникает удержания линии SCL в низком логическом уровне. Если возникает удержание линии SCL в низком логическом уровне, то необходимо сформировать бит данных на SDA Tr.max + Tsu:dat = 1000 + 250 = 1250 нс (согласно спецификации I^2C) прежде, чем SCL будет "отпущена".

30.21 Пример временных диаграмм и параметров модуля MSSP в режиме I²C

Рис. 30-15 Пример временной диаграммы формирования битов START/STOP на шине I²C

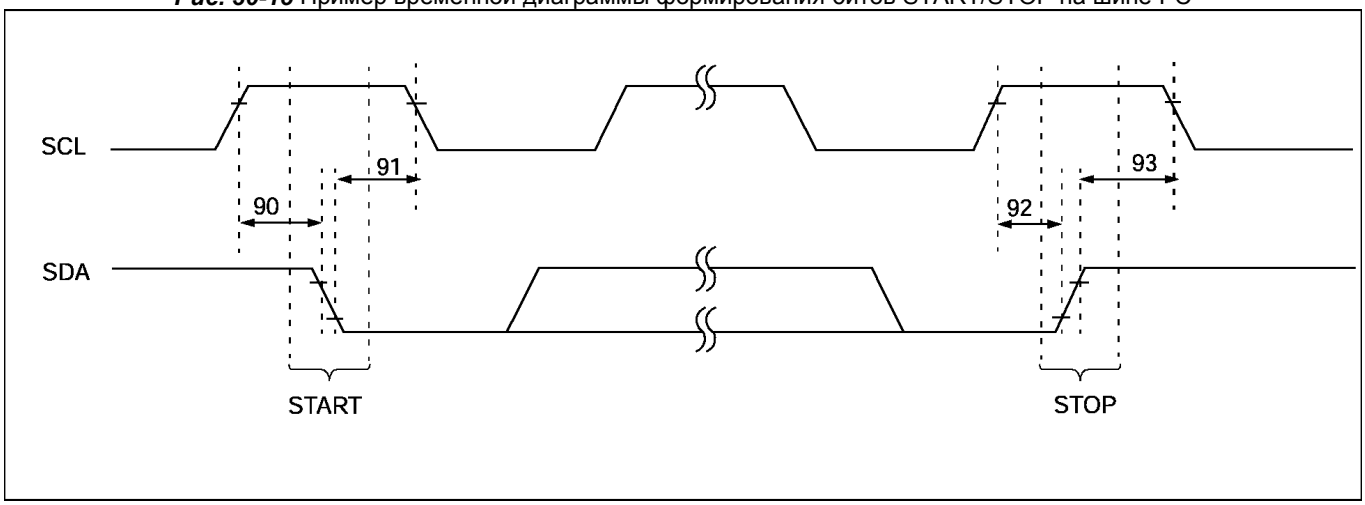


Таблица 30-27 Пример параметров формирования битов START/STOP на шине I²C

№ пар.	Обоз.	Описание		Мин.	Тип.**	Макс.	Ед.	Примечание
90	Tsu:sta	Установка условия START	Режим 100 кГц	$2(T_{OSC})(BRG+1)^{+}$	-	-	нс	Только при формировании бита повторный START
			Режим 400 кГц	$2(T_{OSC})(BRG+1)^{+}$	-	-		
			Режим 1 МГц[1]	$2(T_{OSC})(BRG+1)^{+}$	-	-		
91	Thd:sta	Удержание условия START	Режим 100 кГц	$2(T_{OSC})(BRG+1)^{+}$	-	-	нс	После этого форм. первый импульс тактового сигнала
			Режим 400 кГц	$2(T_{OSC})(BRG+1)^{+}$	-	-		
			Режим 1 МГц[1]	$2(T_{OSC})(BRG+1)^{+}$	-	-		
92	Tsu:sto	Установка условия STOP	Режим 100 кГц	$2(T_{OSC})(BRG+1)^{+}$	-	-	нс	
			Режим 400 кГц	$2(T_{OSC})(BRG+1)^{+}$	-			
			Режим 1 МГц[1]	$2(T_{OSC})(BRG+1)^{+}$	-	-		
93	Thd:sto	Удержание условия STOP	Режим 100 кГц	$2(T_{OSC})(BRG+1)^{+}$	-	-	нс	
			Режим 400 кГц	$2(T_{OSC})(BRG+1)^{+}$	-	-		
			Режим 1 МГц[1]	$2(T_{OSC})(BRG+1)^{+}$	-	-		

$^{+}$ - Параметры соответствуют требованию проекта (см. рисунок А-11 приложения).

Примечание 1. Максимальная емкость вывода 10пФ (для всех выводов I²C).

Рис. 30-16 Пример временной диаграммы формирования бита данных на шине I^2C

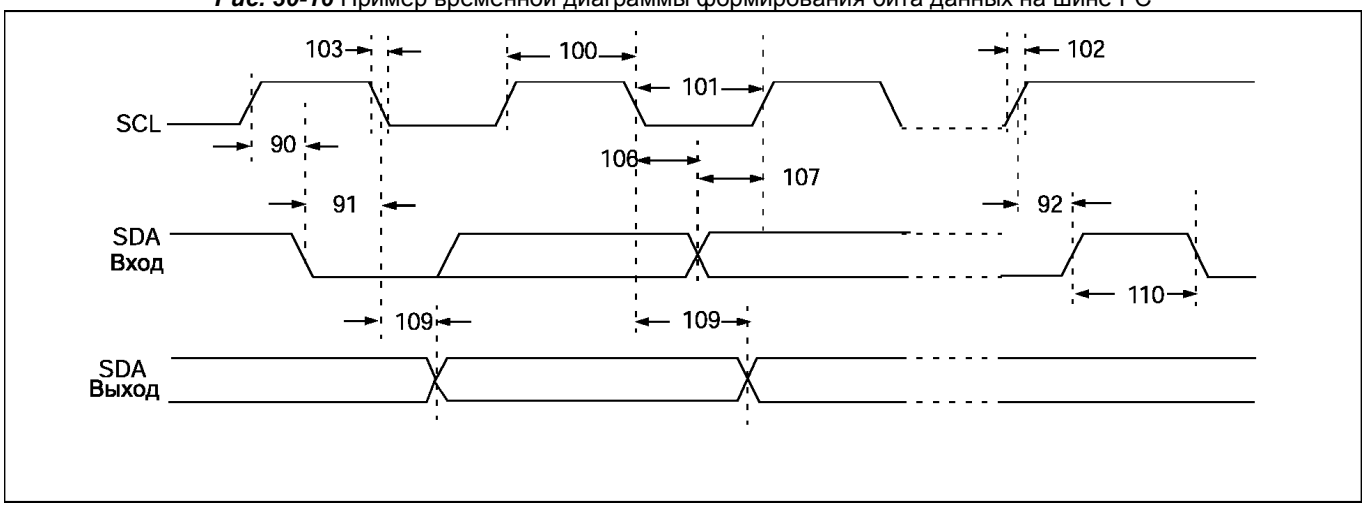


Таблица 30-26 Пример параметров формирования бита данных на шине I^2C

№ пар.	Обоз.	Описание		Мин.	Макс.	Ед.	Примечание
100	Thigh	Длительность высокого уровня тактового сигнала	Режим 100 кГц	$2(T_{OSC})(BRG+1)^+$	-	мкс	
			Режим 400 кГц	$2(T_{OSC})(BRG+1)^+$	-	мкс	
			Режим 1 МГц[1]	$2(T_{OSC})(BRG+1)^+$	-	мкс	
101	Tlow	Длительность низкого уровня тактового сигнала	Режим 100 кГц	$2(T_{OSC})(BRG+1)^+$	-	мкс	
			Режим 400 кГц	$2(T_{OSC})(BRG+1)^+$	-	мкс	
			Режим 1 МГц[1]	$2(T_{OSC})(BRG+1)^+$	-	мкс	
102	Tr	Долит. переднего фронта на SDA и SCL	Режим 100 кГц	-	1000	нс	
			Режим 400 кГц	20 + 0.1 Cb	300	нс	10пФ ≤ Cb ≤ 400пФ
			Режим 1 МГц[1]		300	нс	
103	Tf	Долит. заднего фронта на SDA и SCL	Режим 100 кГц	-	300	нс	
			Режим 400 кГц	20 + 0.1 Cb	300	нс	10пФ ≤ Cb ≤ 400пФ
			Режим 1 МГц[1]		100	нс	
90	Tsu:sta	Установка условия START	Режим 100 кГц	$2(T_{OSC})(BRG+1)^+$	-	мкс	Только при формировании бита повторный START
			Режим 400 кГц	$2(T_{OSC})(BRG+1)^+$	-	мкс	
			Режим 1 МГц[1]	$2(T_{OSC})(BRG+1)^+$	-	мкс	
91	Thd:sta	Удержание условия START	Режим 100 кГц	$2(T_{OSC})(BRG+1)^+$	-	мкс	После этого форм. первый импульс тактового сигнала
			Режим 400 кГц	$2(T_{OSC})(BRG+1)^+$	-	мкс	
			Режим 1 МГц[1]	$2(T_{OSC})(BRG+1)^+$	-	мкс	
106	Thd:dat	Удержание данных на входе	Режим 100 кГц	0	-	нс	
			Режим 400 кГц	0	0.9	мкс	
			Режим 1 МГц[1]	TBD	-	нс	
107	Tsu:dat	Установка данных на входе	Режим 100 кГц	250	-	нс	Примечание 2
			Режим 400 кГц	100	-	нс	
			Режим 1 МГц[1]	TBD	-	нс	
92	Tsu:sto	Установка условия STOP	Режим 100 кГц	$2(T_{OSC})(BRG+1)^+$	-	мкс	
			Режим 400 кГц	$2(T_{OSC})(BRG+1)^+$	-	мкс	
			Режим 1 МГц[1]	$2(T_{OSC})(BRG+1)^+$	-	мкс	
109	Taa	Достоверность сигнала на выходе	Режим 100 кГц	-	3500	нс	
			Режим 400 кГц	-	1000	нс	
			Режим 1 МГц[1]	-	-	нс	
110	Tbuf	Время не занятости шины	Режим 100 кГц	4.7*	-	мкс	Задержка перед новой передачей
			Режим 400 кГц	1.3*	-	мкс	
			Режим 1 МГц[1]	TBD	-	мкс	
	Cb	Емкостная нагрузка линии		-	400	пФ	

$^+$ - Параметры соответствуют требованию проекта (см. рисунок A-11 приложения).

* - Эти параметры определены, но не протестированы.

Примечания:

1. Максимальная емкость вывода 10пФ (для всех выводов I^2C).
2. Устройства с высокоскоростным режимом обмена (400кГц) могут использоваться в стандартном режиме (100кГц), но требование Tsu:dat ≥ 250нс необходимо выполнять. Это условие автоматически будет выполняться, если не возникает удержания линии SCL в низком логическом уровне. Если возникает удержание линии SCL в низком логическом уровне, то необходимо сформировать бит данных на SDA Tr.max + Tsu:dat = 1000 + 250 = 1250 нс (согласно спецификации I2C) прежде, чем SCL будет "отпущена".

30.22 Пример временных диаграмм и параметров USART

Рис. 30-17 Пример временной диаграммы работы передатчика USART в ведущем/ведомом синхронном режиме

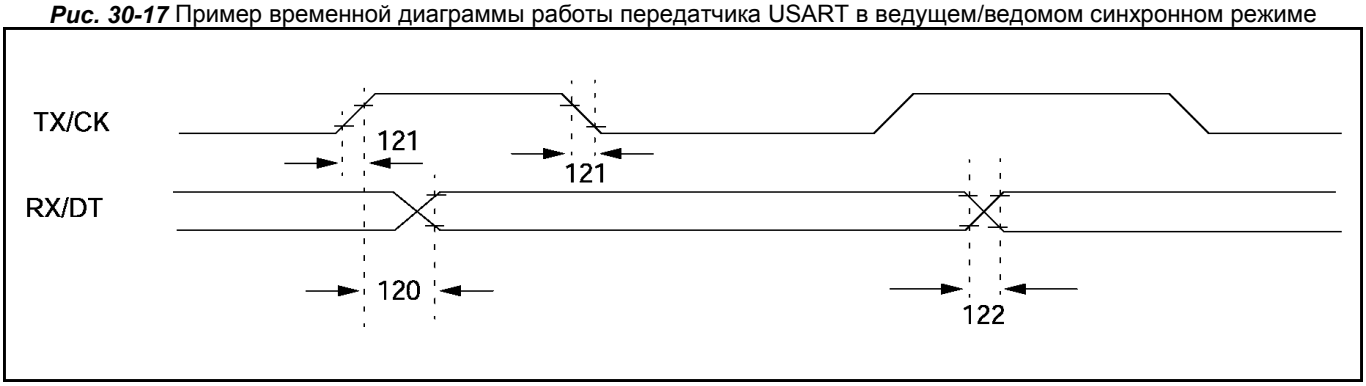


Таблица 30-29 Пример параметров работы передатчика USART в ведущем/ведомом синхронном режиме

№ пар.	Обоз.	Описание		Мин.	Тип.**	Макс.	Ед.	Примечание
120	TckH2dtV	Действ. данные после перехода такт. сигнала в высокий уровень	**C**	-	-	80	нс	
			LC	-	-	100	нс	
121	Tckrf	Длительность заднего/переднего фронта такт. сигн. (ведущий)	**C**	-	-	45	нс	
			LC	-	-	50	нс	
122	Tdtrf	Длительность переднего/заднего фронта данных	**C**	-	-	45	нс	
			LC	-	-	50	нс	

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Рис. 30-18 Пример временной диаграммы работы приемника USART в ведущем/ведомом синхронном режиме

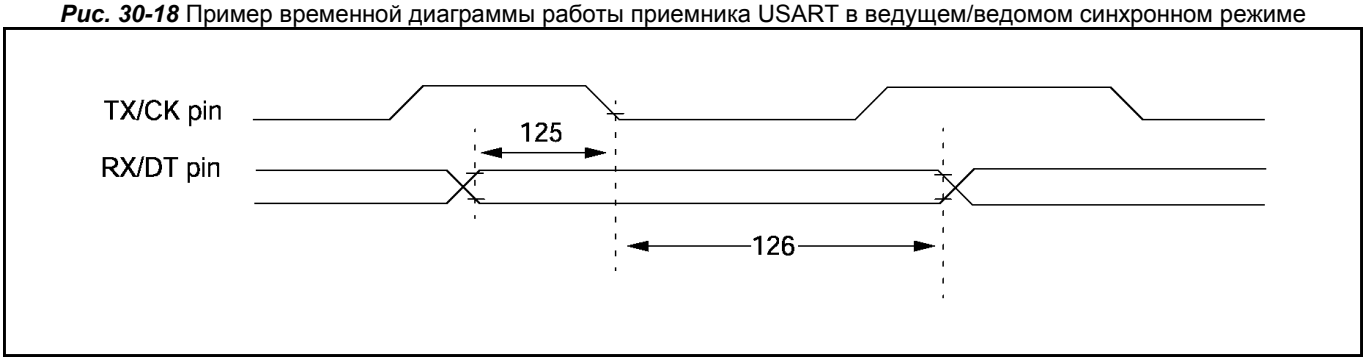


Таблица 30-30 Пример параметров работы приемника USART в ведущем/ведомом синхронном режиме

№ пар.	Обоз.	Описание	Мин.	Тип.**	Макс.	Ед.	Примечание
125	TdtV2ckL	Установка данных после CK↓	15	-	-	нс	
126	TckL2dl	Удержание данных после CK↓	15	-	-	нс	

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

30.23 Пример временных диаграмм и параметров 8 - разрядного АЦП

Таблица 30-31 Пример параметров работы 8 - разрядного АЦП

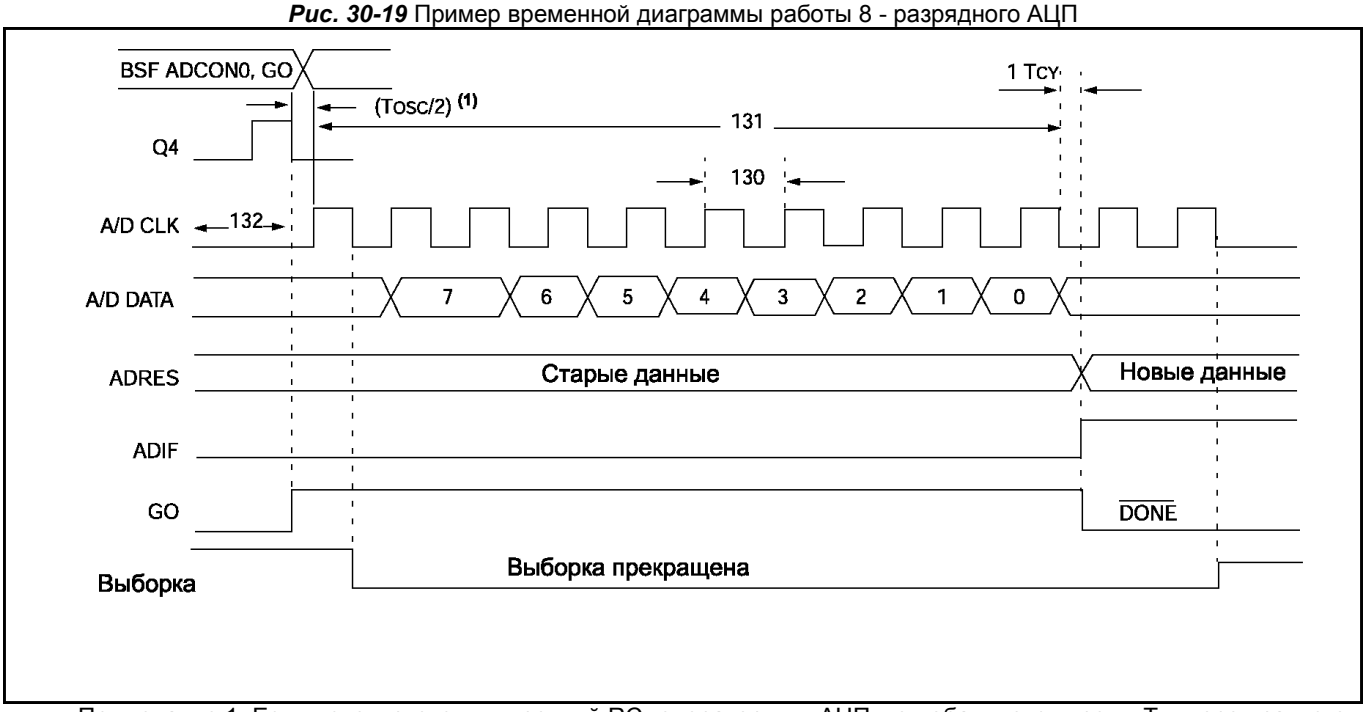
№ пар.	Обоз.	Описание		Мин.	Тип**	Макс.	Ед.	Примечание
A01	N_R	Разрядность		-	-	8	бит	$V_{REF} = V_{DD} = 5.12В$, $V_{SS} \le V_{AIN} \le V_{REF}$
A02	E_{ABS}	Абсолютная погрешность		-	-	$< \pm 1$	LSb	$V_{REF} = V_{DD} = 5.12В$, $V_{SS} \le V_{AIN} \le V_{REF}$
A03	E_{IL}	Интегральная погрешность		-	-	$< \pm 1$	LSb	$V_{REF} = V_{DD} = 5.12В$, $V_{SS} \le V_{AIN} \le V_{REF}$
A04	E_{DL}	Дифференциальная погрешность		-	-	$< \pm 1$	LSb	$V_{REF} = V_{DD} = 5.12В$, $V_{SS} \le V_{AIN} \le V_{REF}$
A05	E_{FS}	Ошибка полной шкалы		-	-	$< \pm 1$	LSb	$V_{REF} = V_{DD} = 5.12В$, $V_{SS} \le V_{AIN} \le V_{REF}$
A06	E_{OFF}	Ошибка смещения		-	-	$< \pm 2$	LSb	$V_{REF} = V_{DD} = 5.12В$, $V_{SS} \le V_{AIN} \le V_{REF}$
A10	-	Монотонность		Гарантируется			-	$V_{SS} \le V_{AIN} \le V_{REF}$
A20	V_{REF}	Опорное напряжение		3.0	-	V_{DD} + 0.3	В	
A25	V_{AIN}	Аналоговый вход		V_{SS} – 0.3	-	V_{REF} + 0.3	В	
A30	Z_{AIN}	Сопротивление источника сигн.		-	-	10.0	кОм	
A40	I_{AD}	Потребляемый ток АЦП	**C**	-	180	-	мкА	Среднее потребление при включенном АЦП(1)
			LC	-	90	-	мкА	
A50	I_{REF}	Потребляемый ток от источника опорного напряжения(2)		10 -	- -	1000 10	мкА мкА	Во время выборки V_{AIN}. Основано на дифференц. значении заряда C_{HOLD} до V_{AIN}. Во время преобразования.

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Примечания:

1. Выключенный модуль АЦП не потребляет тока, кроме токов утечки.
2. Ток со входа V_{REF} или V_{DD} в зависимости от выбранного источника опорного напряжения.
3. Результат АЦП никогда не уменьшается с увеличением напряжения на входе и не имеет кодов отсутствия напряжения.

Рис. 30-19 Пример временной диаграммы работы 8 - разрядного АЦП



Примечание 1. Если используется внутренний RC генератор для АЦП, то добавляется время T_{CY} перед запуском АЦП, позволяющее выполнить команду SLEEP.

Таблица 32-34 Пример параметров работы 8 - разрядного АЦП

№ пар.	Обоз.	Описание		Мин.	Тип.**	Макс.	Ед.	Примечание
130	T_{AD}	Период тактового сигнала АЦП	**C**	1.6	-	-	мкс	Основа T_{OSC}, $V_{REF} \geq 3.0$ В
			LC	3.0	-	-	мкс	Основа T_{OSC}, $V_{REF} \geq 2.0$ В
			C	2.0	4.0	6.0	мкс	RC генератор АЦП
			LC	3.0	6.0	9.0	мкс	RC генератор АЦП
131	T_{CNV}	Время преобразования (1)		11*	-	12*	T_{AD}	
132	T_{ACQ}	Время выборки		(2) 5	20 -	- -	мкс мкс	 Примечание 3
134	T_{GO}	Старт преобразования относительно Q4		-	$T_{OSC}/2$***	-	-	Примечание 4
136	T_{AMP}	Время реакции усилителя		1	-	-	мкс	Примечание 4
135	T_{SWC}	Время переключения от преобразования к выборке		-	-	-		

* - Эти параметры определены, но не протестированы.

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Примечания:

1. Регистр ADRES может быть прочитан в следующем цикле.
2. Смотрите раздел "8 - разрядное АЦП" для выбора минимального значения.
3. Минимальное время - задержка усилителя. Может использоваться, если напряжение на входе изменилось не более, чем на 1 LSb (т.е. 20мВ @ 5.12В) от последнего измерения.
4. Если используется внутренний RC генератор для АЦП, то добавляется время T_{CY} перед запуском АЦП, позволяющее выполнить команду SLEEP.

30.24 Пример временных диаграмм и параметров 10 - разрядного АЦП

Таблица 30-33 Пример параметров работы 10 - разрядного АЦП

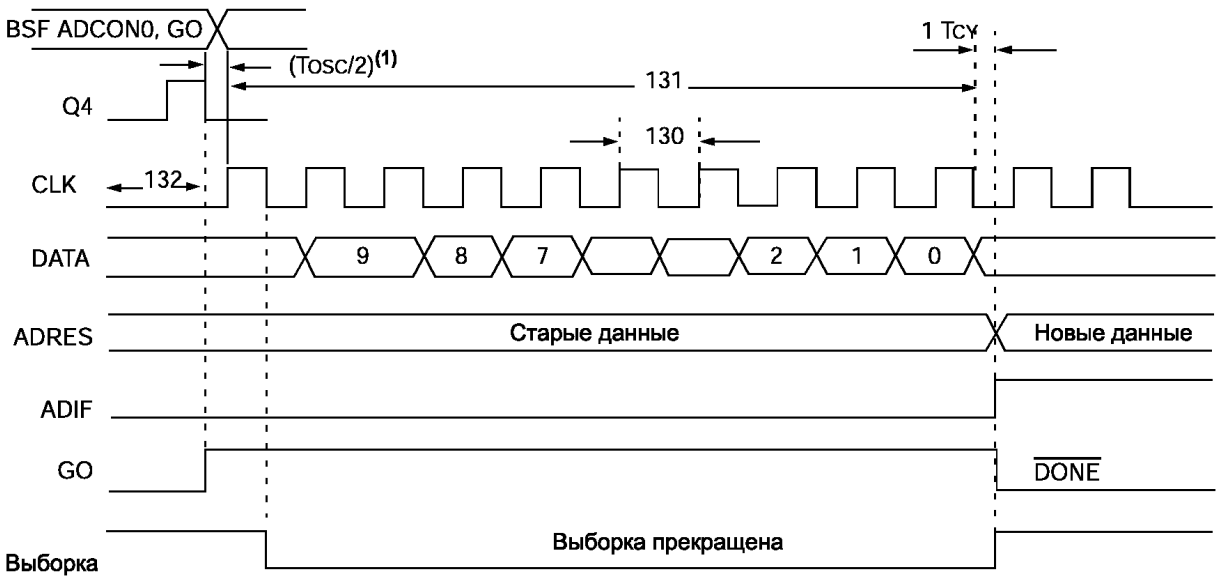
№ пар.	Обоз.	Описание		Мин.	Тип**	Макс.	Ед.	Примечание
A01	N_R	Разрядность		-	-	10	бит	$V_{REF} = V_{DD} = 5.12В$, $V_{SS} \le V_{AIN} \le V_{REF}$
A02	E_{ABS}	Абсолютная погрешность		-	-	< ± 1	LSb	$V_{REF} = V_{DD} = 5.12В$, $V_{SS} \le V_{AIN} \le V_{REF}$
A03	E_{IL}	Интегральная погрешность		-	-	< ± 1	LSb	$V_{REF} = V_{DD} = 5.12В$, $V_{SS} \le V_{AIN} \le V_{REF}$
A04	E_{DL}	Дифференциальная погрешность		-	-	< ± 1	LSb	$V_{REF} = V_{DD} = 5.12В$, $V_{SS} \le V_{AIN} \le V_{REF}$
A05	E_{FS}	Ошибка полной шкалы		-	-	< ± 1	LSb	$V_{REF} = V_{DD} = 5.12В$, $V_{SS} \le V_{AIN} \le V_{REF}$
A06	E_{OFF}	Ошибка смещения		-	-	< ± 2	LSb	$V_{REF} = V_{DD} = 5.12В$, $V_{SS} \le V_{AIN} \le V_{REF}$
A10	-	Монотонность		Гарантируется			-	$V_{SS} \le V_{AIN} \le V_{REF}$
A20 A20A	V_{REF}	Опорное напряжение (V_{REF+}-V_{REF-})		2.0	-	V_{DD} + 0.3	В	Для 10 - раз. результата
A21	V_{REF+}	Положительное опорное напр.		AV_{SS}		AV_{DD} + 0.3	В	
A22	V_{REF-}	Отрицательное опорное напр.		AV_{SS} – 0.3		AV_{DD}	В	
A25	V_{AIN}	Аналоговый вход		AV_{SS} – 0.3	-	V_{REF} + 0.3	В	
A30	Z_{AIN}	Сопротивление источника сигн.		-	-	10.0	кОм	
A40	I_{AD}	Потребляемый ток АЦП	**C**	-	180	-	мкА	Среднее потребление при включенном АЦП(1)
			LC	-	90	-	мкА	
A50	I_{REF}	Потребляемый ток от источника опорного напряжения(2)		10	-	1000	мкА	Во время выборки V_{AIN}. Основано на дифференц. значении заряда C_{HOLD} до V_{AIN}.
				-	-	10	мкА	Во время преобразования.

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Примечания:

1. Выключенный модуль АЦП не потребляет тока, кроме токов утечки.
2. Ток со входа V_{REF} или V_{DD} в зависимости от выбранного источника опорного напряжения.
3. Результат АЦП никогда не уменьшается с увеличением напряжения на входе и не имеет кодов отсутствия напряжения.

Рис. 30-20 Пример временной диаграммы работы 10 - разрядного АЦП



Примечание 1. Если используется внутренний RC генератор для АЦП, то добавляется время T_{CY} перед запуском АЦП, позволяющее выполнить команду SLEEP.

Примечание 2. Минимальная задержка RC цепочки (номинальное значение 100нс) включая отсоединение внутреннего конденсатора C_{HOLD} от аналогового входа.

Таблица 30-34 Пример параметров работы 10 - разрядного АЦП

№ пар.	Обоз.	Описание		Мин.	Тип.**	Макс.	Ед.	Примечание
130	T_{AD}	Период тактового сигнала АЦП	**C**	1.6	-	-	мкс	Основа T_{OSC}, $V_{REF} \geq 3.0$ В
			LC	3.0	-	-	мкс	Основа T_{OSC}, $V_{REF} \geq 2.0$ В
			C	2.0	4.0	6.0	мкс	RC генератор АЦП
			LC	3.0	6.0	9.0	мкс	RC генератор АЦП
131	T_{CNV}	Время преобразования (1)		11*	-	12*	T_{AD}	
132	T_{ACQ}	Время выборки(3)		15	-	-	мкс	от -40°C до +125°C
				10	-	-	мкс	от 0°C до +125°C
136	T_{AMP}	Время реакции усилителя		1	-	-	мкс	Примечание 5
135	T_{SWC}	Время переключения от преобразования к выборке		-	-	-		Примечание 4

* - Эти параметры определены, но не протестированы.

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Примечания:

1. Регистр ADRES может быть прочитан в следующем цикле.
2. Смотрите раздел "10 - разрядное АЦП" для выбора минимального значения.
3. Время заряда конденсатора C_{HOLD} до входного напряжения, когда изменение напряжения соответствует полной шкале (переход от AV_{DD} к AV_{SS} или от AV_{SS} к AV_{DD}).
4. В следующем цикле на такте Q4.
5. Минимальное время - задержка усилителя. Может использоваться, если напряжение на входе изменилось не более, чем на 1 LSb (т.е. 20мВ @ 5.12В) от последнего измерения.

30.25 Пример временных диаграмм и параметров интегрирующего АЦП

Рис. 30-21 Пример временной диаграммы работы интегрирующего АЦП

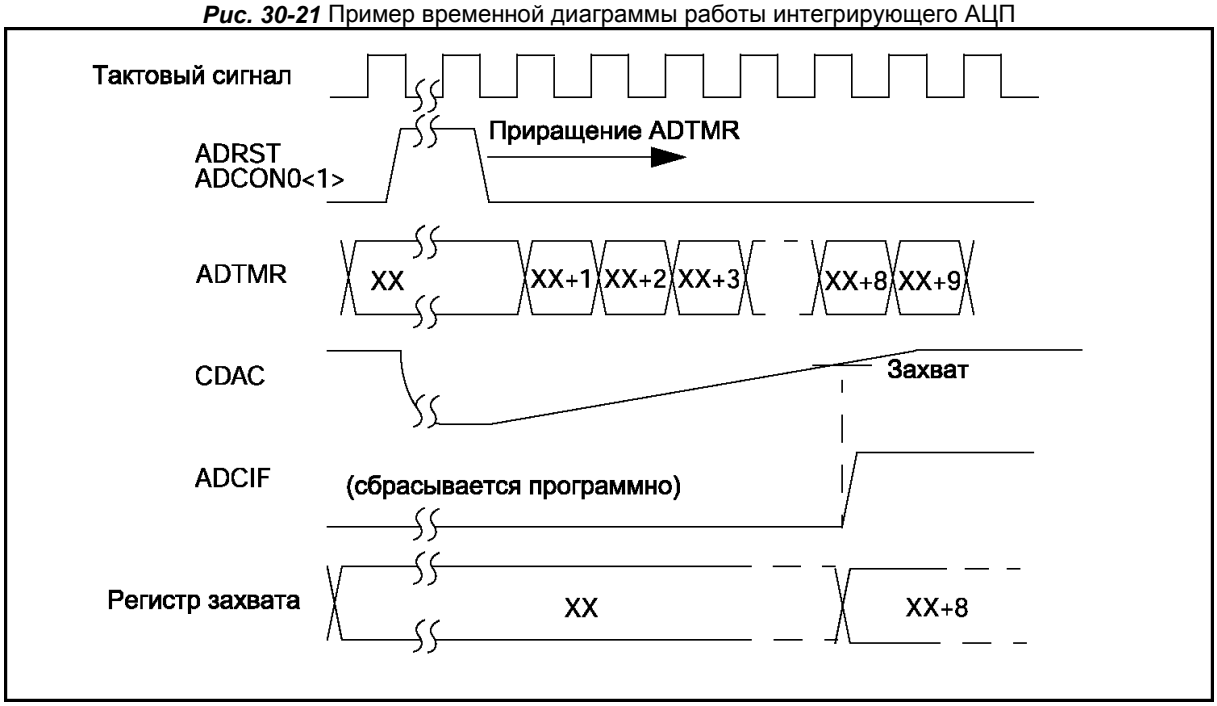


Таблица 30-35 Пример характеристик интегрирующего АЦП

Характеристики по постоянному току			**Стандартные рабочие условия (если не указано иное)** Температурный диапазон: Коммерческий 0°C ≤ T_A ≤ +70°C Промышленный -40°C ≤ T_A ≤ +85°C Расширенный -40°C ≤ T_A ≤ +125°C Параметры напряжения питания смотрите в таблице 30-3.				
№ пар.	**Обоз.**	**Описание**	**Мин.**	**Тип.****	**Макс.**	**Ед.**	**Примечание**
		Компаратор АЦП					
A100	V_{AIN}	Диапазон входного сигнала	V_{SS}	-	V_{DD}-1.4	В	
A101		Смещение входа	-10	2	10	мВ	В общем режиме
A102	G_{DV}	Дифференциальное усиление(1)	-	100	-	дБ	
A103	CMRR	Фильтрация в общем режиме (1)	-	80	-	дБ	V_{DD} = 5В, T_A = 25°C
A104	RR_{ADC}	Фильтрация по питанию (1)	-	70	-	дБ	T_A = 25°C, $V_{DDMIN} \le V_{DD} \le V_{DDMAX}$
	T_{SET}	Время реакции					
140		Источник опорного напряжения (< 0.1%) (1)	-	1	10	мс	Перевод бита REFOFF в регистре SLPCON из 1 в 0
141		Программируемый источник тока (< 0.1%)(1)	-	1	10	мс	Генератор линейно нарастающего напряжения (REFOFF 1→0), старт.
141A			-	1	10	мс	REFOFF = 0, ADCON1<7:4> 0000b → 1111b
		Температурные коэффициенты(1)					
A110	TC_{BGR}	Источник опорного напряжения	-	+50		ppm/°C	-40°C ≤ T_A ≤ +25°C
			-	-50			+25°C ≤ T_A ≤ +85°C
A110A			-	+20		ppm/°C	0°C ≤ T_A ≤ +25°C
			-	-20			+25°C ≤ T_A ≤ +70°C
A111	TC_{PCS}	Программ. источник тока	-	+0.1		%/°C	-40°C ≤ T_A ≤ +25°C
			-	-0.1			+25°C ≤ T_A ≤ +85°C
A112	TC_{kref}	Делитель опорного напряжения	-	20		ppm/°C	-40°C ≤ T_A ≤ +85°C
	CA	Точность калибровки (3,5)					
A120	CA_{BRG}	Источник опорного напряжения	-	0.01	-	%	V_{DD} = 5В, T_A = 25°C
A121	CA_{SRV}	Делитель опорного напряжения	-	0.02	-	%	
	SN	Чувствительность к напряжению питания(1)					
A130	SN_{BRG}	Источник опорного напряжения	-	0.04	-	%/В	$V_{DDMIN} \le V_{DD} \le V_{DDMAX}$
A131	SN_{PCS}	Программ. источник тока	-	0.2	-	%/В	$V_{DDMIN} \le V_{DD} \le V_{DDMAX}$
A132	SN_{kref}	Делитель опорного напряжения	-	0.02	-	%/В	$V_{DDMIN} \le V_{DD} \le V_{DDMAX}$
		Программируемый источник тока					
A140	I_{RES}	Разрешение	1.25	2.25	3.25	мкА	1 Lsb
A141	E_{IL}	Ошибка линейности	-1/2		+1/2	Lsb	CDAC=0B

30.26 Пример временных диаграмм и параметров модуля LCD

Рис. 30-22 Пример временной диаграммы работы модуля LCD

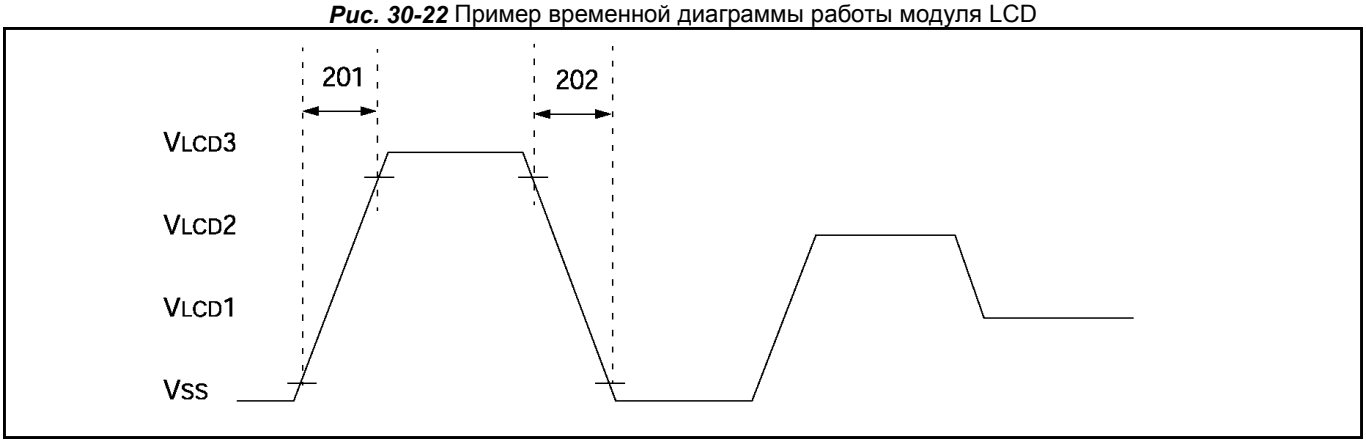


Таблица 30-36 Пример характеристик модуля LCD

№ пар.	Обоз.	Описание	Мин.	Тип.**	Макс.	Ед.	Примечание
200	F_{LCDRC}	Частота RC генератора LCD	-	14	22	кГц	V_{DD} = 5В, -40°C ≤ T_A ≤ +85°C
201	Tr_{LCD}	Длительность переднего фронта сигнала на выходе LCD	-	-	200	мкс	Нагрузка COM выв. = 5нФ, Нагрузка SEG выв. = 500пФ V_{DD} = 5В, T_A = 25°C
202	Tf_{LCD}	Длительность заднего фронта сигнала на выходе LCD(1)	Tr_{LCD} - 0.05Tr_{LCD}	-	Tr_{LCD} + 0.05Tr_{LCD}	мкс	Нагрузка COM выв. = 5нФ, Нагрузка SEG выв. = 500пФ V_{DD} = 5В, T_A = 25°C

** - В столбце "Тип." приведены параметры при V_{DD}=5.0В @ 25°C, если не указано иное. Эти параметры являются ориентировочными, используются при разработке устройств и не измеряются.

Примечание 1. Внутренне сопротивление V_{LCD} 0 Ом.

30.27 Ответы на часто задаваемые вопросы

На момент выполнения перевода в оригинальной технической документации вопросы отсутствовали. Если у вас есть вопрос, задайте его, написав нам письмо по адресу support@microchip.ru.

30.28 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило, примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные с электрическими характеристиками микроконтроллеров PICmicro MCU:

Документ	Номер

В настоящее время документы не подготовлены

Раздел 31. Характеристики микроконтроллеров

Содержание

31.1 Введение

Компания Microchip полностью предоставляет характеристику выпускаемых микросхем. Эти данные становятся доступными для разработчиков после проведения многократных испытаний и анализа результатов. Представленные характеристики позволяют разработчикам наиболее точно оценить возможность применения микросхем для конкретного проекта.

31.2 Различия между характеристиками и электрическими параметрами

Основным отличием данного раздела от электрических характеристик является информация о том, что пользователь должен ожидать при выходе из нормального режима работы. Представленные графики и таблицы предназначены для оценки проекта и не проверяются (не гарантируются). Из-за особенностей испытательного оборудования возможно несоответствие данных с разделом "Электрические характеристики".

31.3 Графики и таблицы характеристик микроконтроллеров

Графики в этом разделе не проверены и предназначены только для оценки при разработке устройств. В некоторых графиках представлены данные вне рабочего диапазона (в частности для напряжения питания V_{DD}). Это только информационные данные.

Примечание. Данные, представленные в этом разделе, являются среднестатистическим результатом испытаний большого числа микроконтроллеров в течение длительного времени. Типовое значение подразумевает среднее (при температуре +25°C), а минимальное и максимальное - соответственно (среднее - 3σ) и (среднее + 3σ), где σ - стандартный разброс.

31.3.1 Зависимость I_{PD} от V_{DD}

I_{PD} - ток потребления микроконтроллера в SLEEP режиме микроконтроллера. Измерения проводились, когда все порты ввода/вывода настроены как входы и имеют низкий или высокий логический уровень (нет выводов с неопределенным уровнем и выводов с подключенной нагрузкой).

Характеристики представлены с различными режимами работы WDT (включен/выключен). Это необходимо, т.к. WDT имеет внутренний RC генератор, который потребляет дополнительный ток.

Микроконтроллер может иметь дополнительные особенности и периферийные модули, которые могут работать в SLEEP режиме микроконтроллера:

- Сторожевой таймер WDT;
- Схема сброса по снижению напряжения питания;
- Таймер TMR1;
- АЦП;
- Модуль LCD;
- Компараторы;
- Источник опорного напряжения.

Когда все эти модули выключены, микроконтроллер потребляет наименьший ток (ток утечки). Если любой из этих модулей включен, то потребление в SLEEP режиме значительно увеличится. Разница между минимально возможным током потребления и током потребления с включенным одним из перечисленных модулей (например WDT) называется дифференциальным током потребления модуля. Если включено несколько дополнительных модулей, то суммарный ток потребления может быть легко вычислен: основной ток потребления (ток потребления в SLEEP режиме, когда все модули выключены) плюс дифференциальный ток потребления каждого включенного периферийного модуля.

В примере 31-1 показано вычисление тока потребления микроконтроллера в SLEEP режиме с включенным WDT и TMR1 (внутренний генератор TMR1) при напряжении питания 5В.

Пример 31-1 Вычисление I_{PD} при напряжении питания 5В (включен WDT и TMR1 с внутренним генератором)

Основной ток	14нА	; Ток утечки микроконтроллера
Ток потребления WDT	14мкА	; 14мкА - 14нА = 14мкА
Ток потребления TMR1	22мкА	; 22мкА - 14нА = 22мкА
Суммарный ток потребления	36мкА	;

Рис. 31-1 Пример типового I_{PD} от V_{DD} (WDT выключено, RC режим)

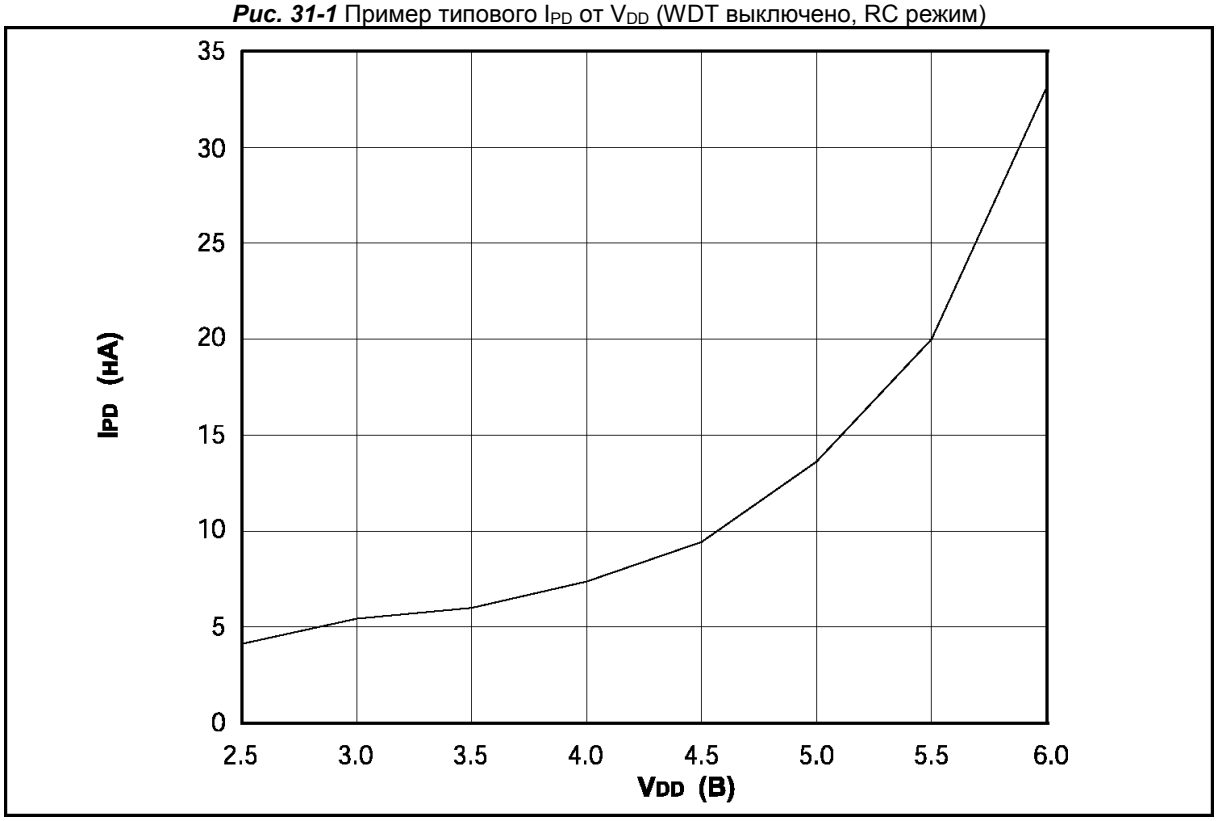


Рис. 31-2 Пример максимального I_{PD} от V_{DD} (WDT выключено, RC режим)

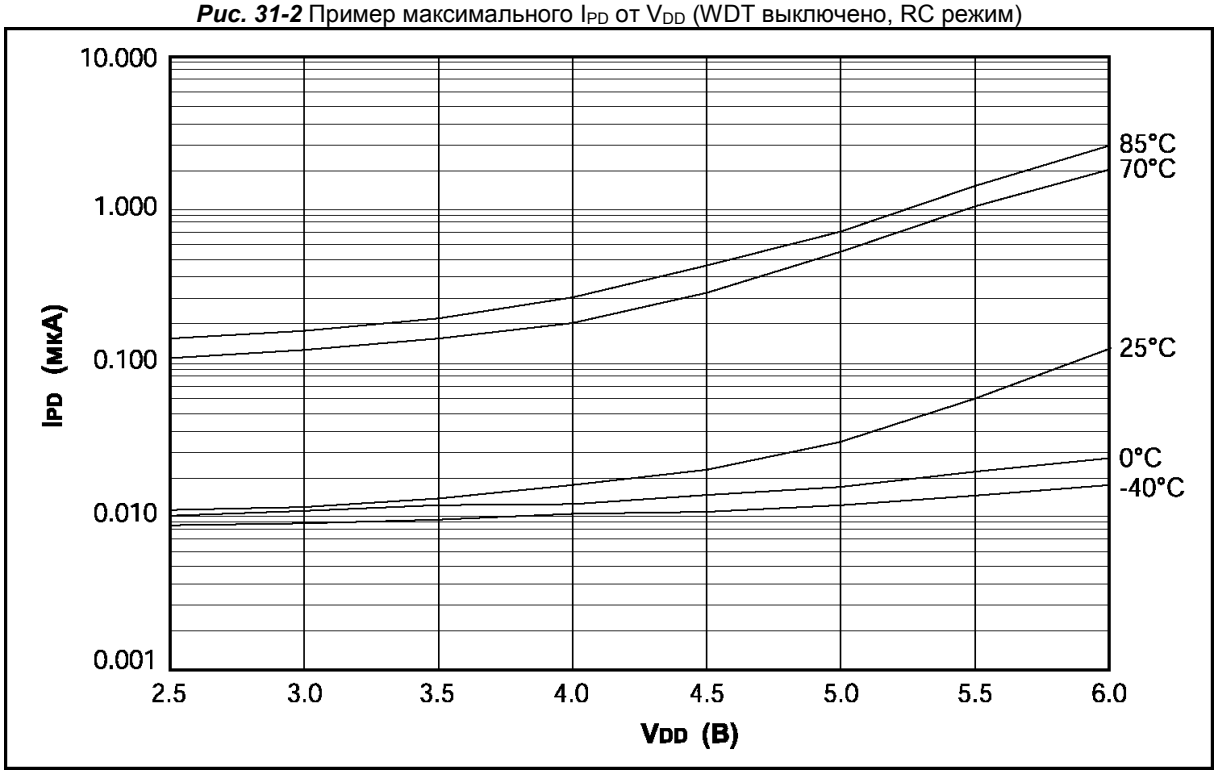


Рис. 31-3 Пример типового I_{PD} от V_{DD} (WDT включено, RC режим)

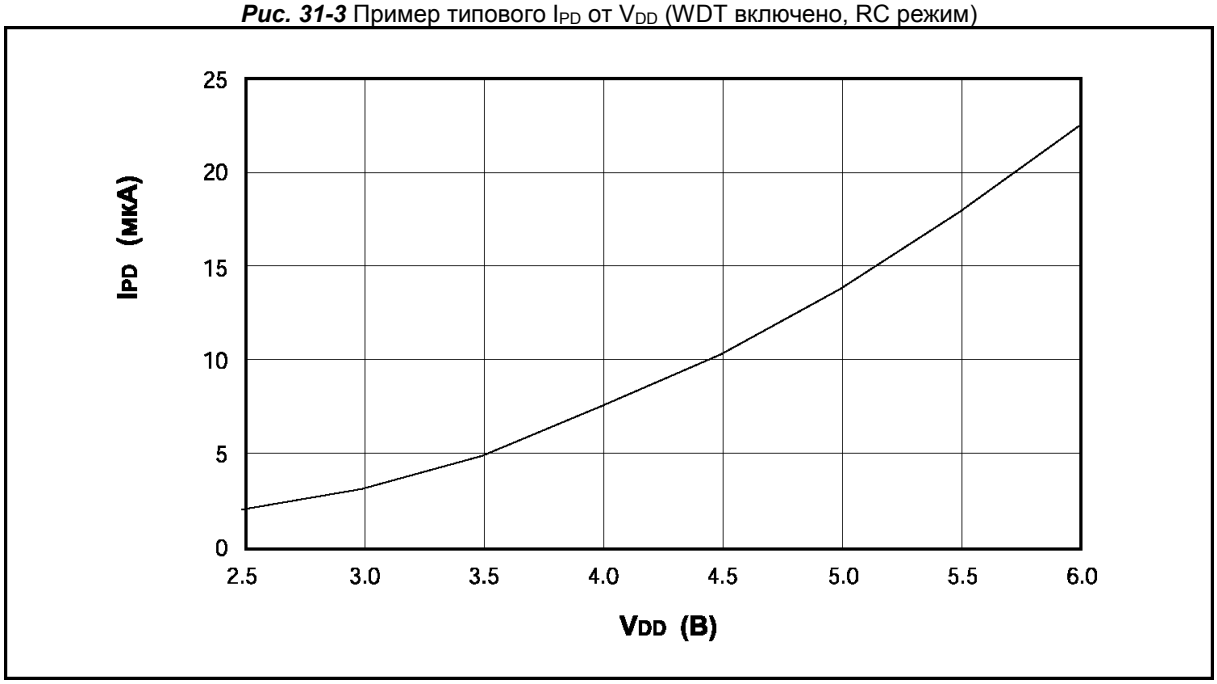


Рис. 31-4 Пример максимального I_{PD} от V_{DD} (WDT включено, RC режим)

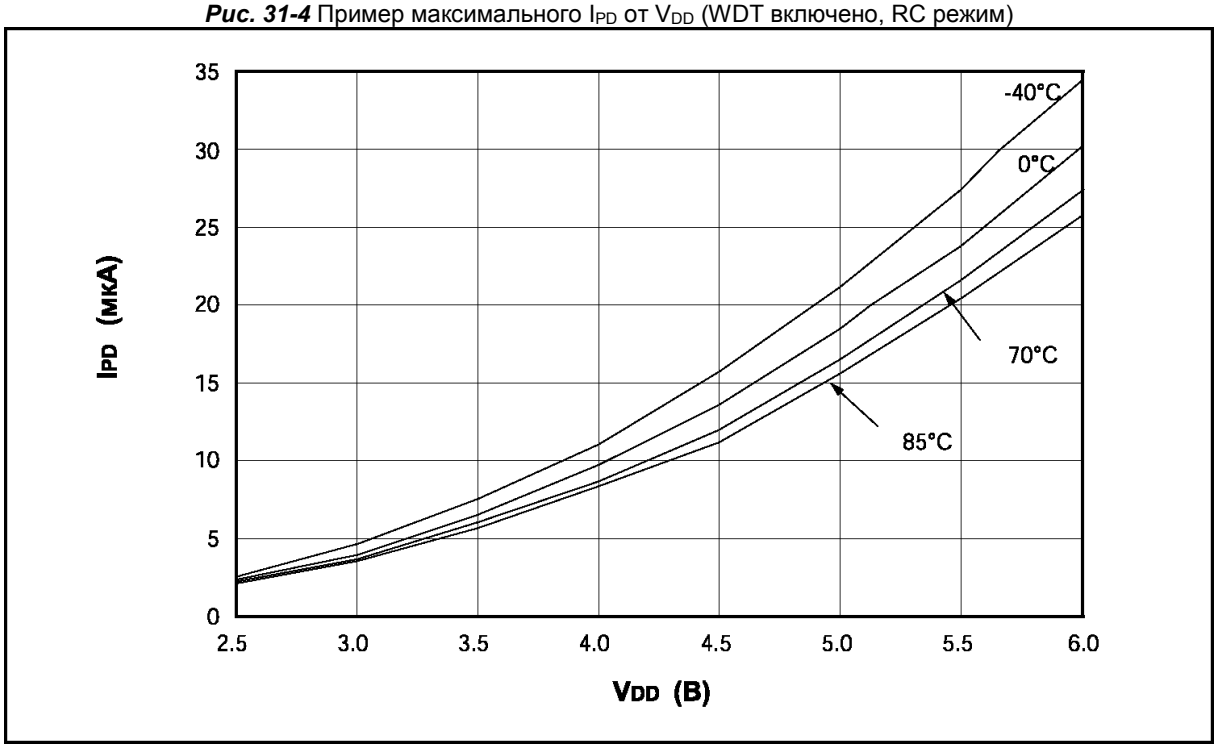
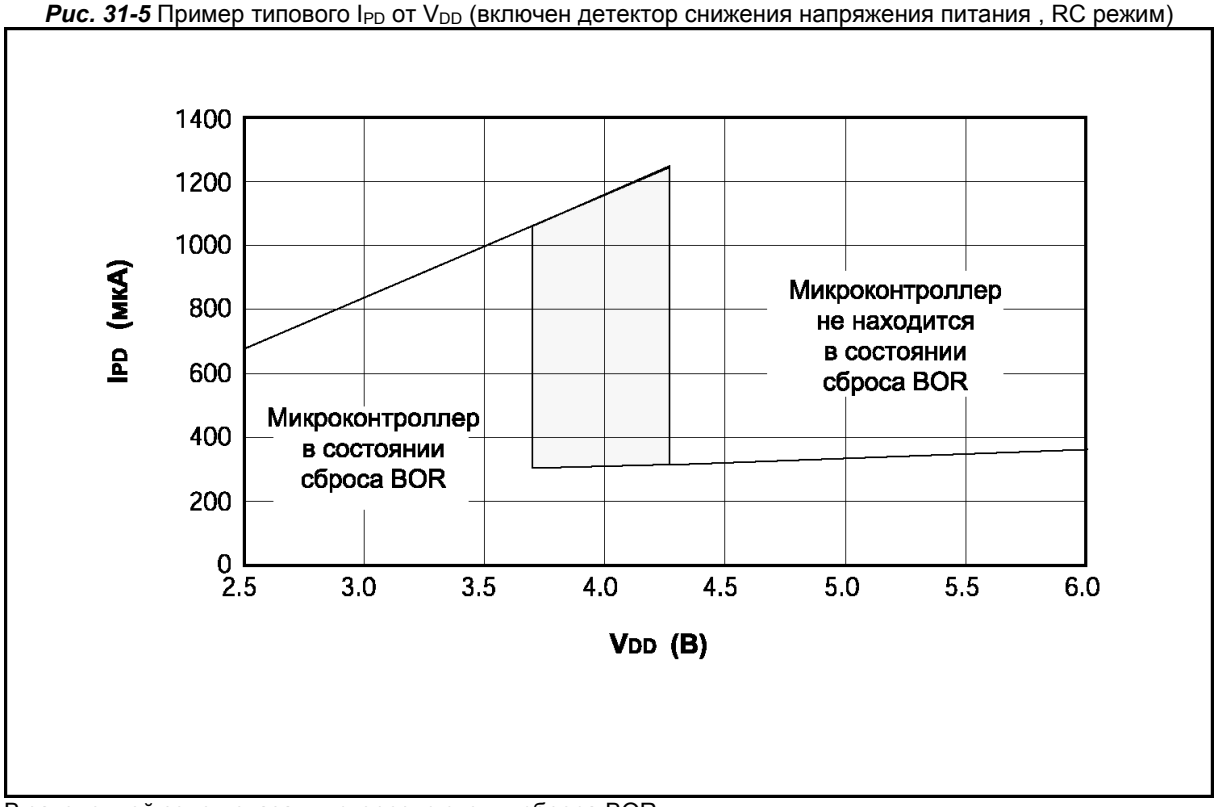
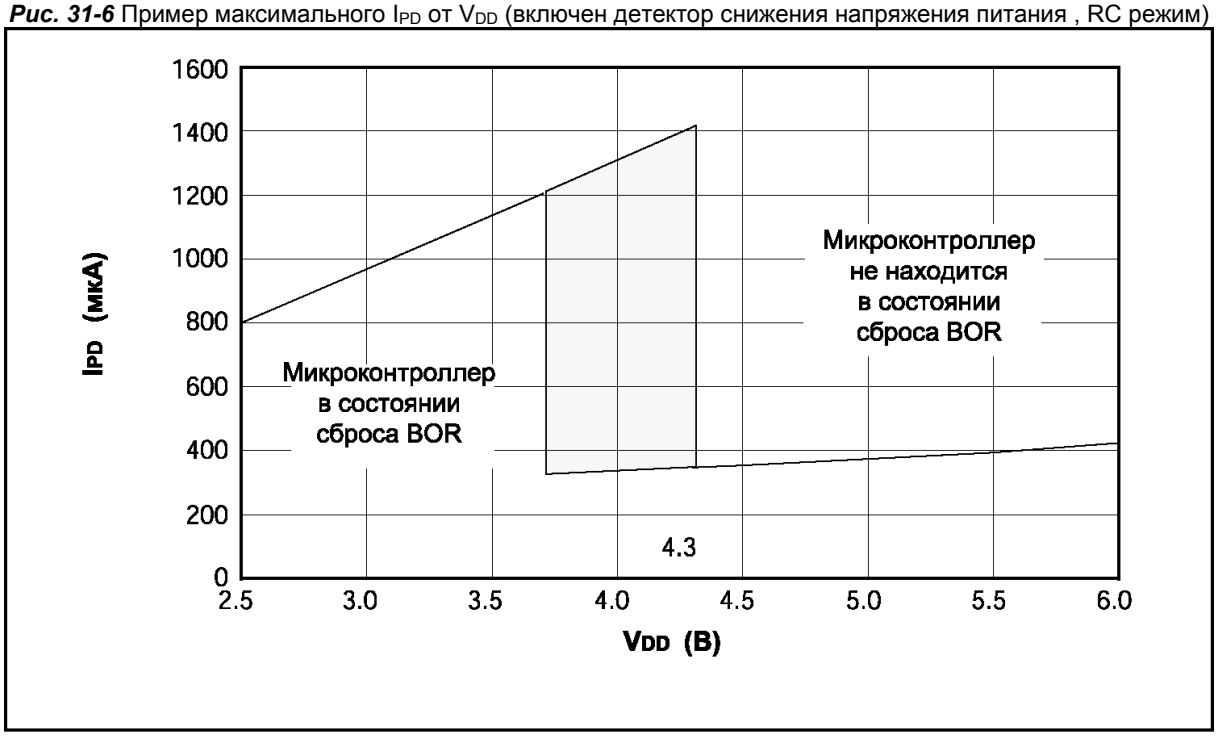


Рис. 31-5 Пример типового I_{PD} от V_{DD} (включен детектор снижения напряжения питания , RC режим)



В затененной зоне показан гистерезис схемы сброса BOR.

Рис. 31-6 Пример максимального I_{PD} от V_{DD} (включен детектор снижения напряжения питания , RC режим)



В затененной зоне показан гистерезис схемы сброса BOR.

Рис. 31-7 Пример типового I_{PD} от V_{DD} (включен генератор TMR1 33кГц, RC0/RC1 = 33пФ/33пФ, RC режим)

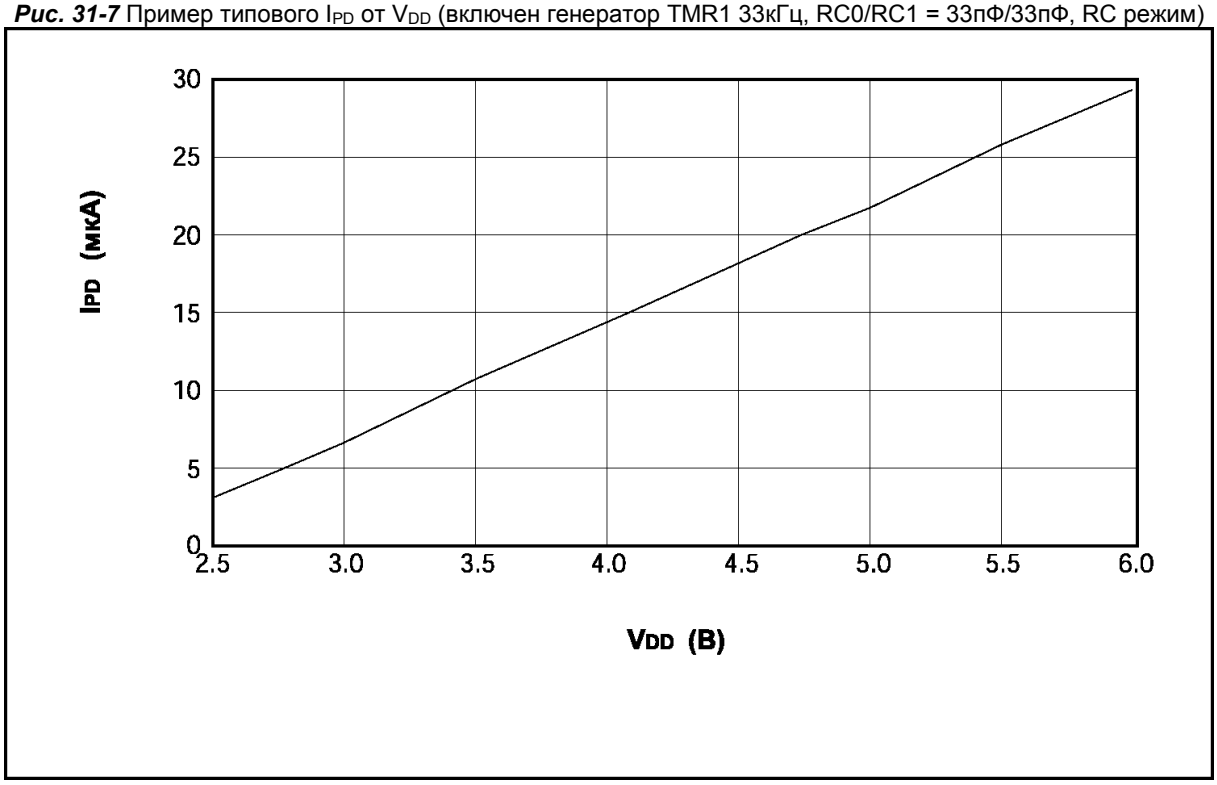
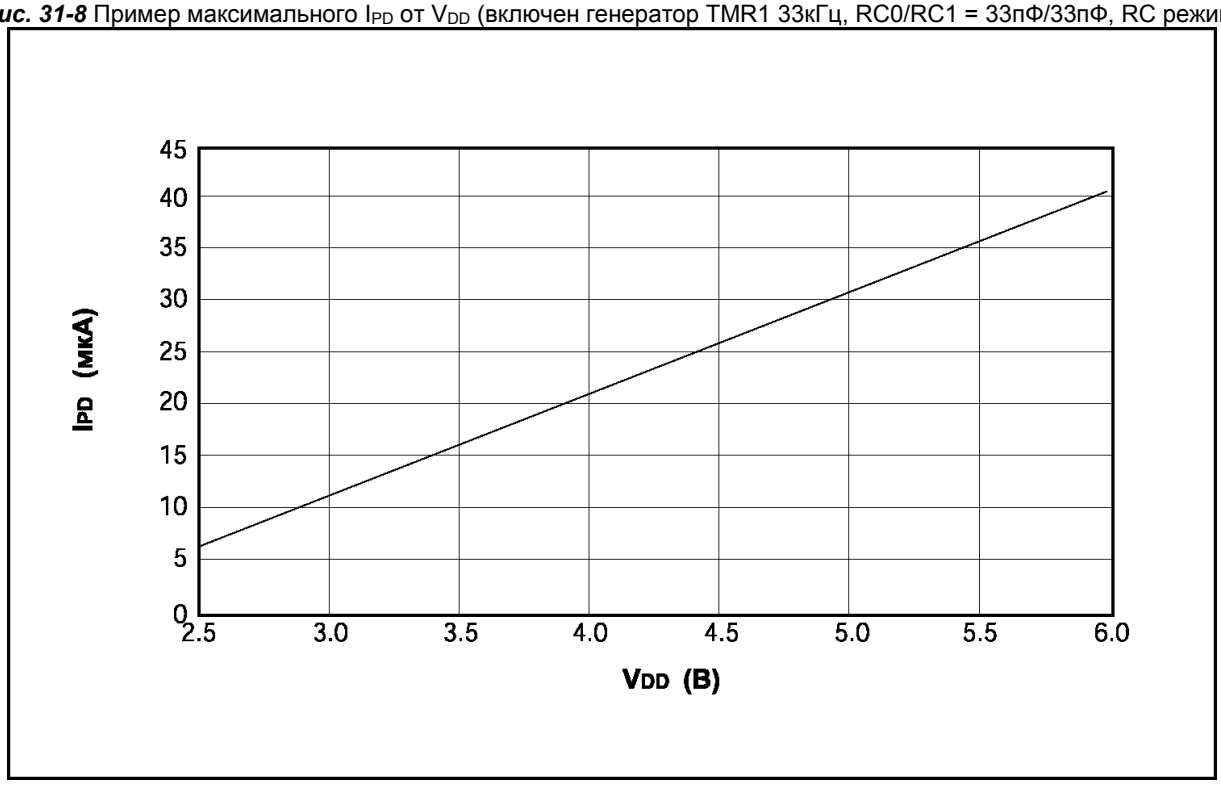


Рис. 31-8 Пример максимального I_{PD} от V_{DD} (включен генератор TMR1 33кГц, RC0/RC1 = 33пФ/33пФ, RC режим)



31.3.2 Зависимость I_{DD} от тактовой частоты

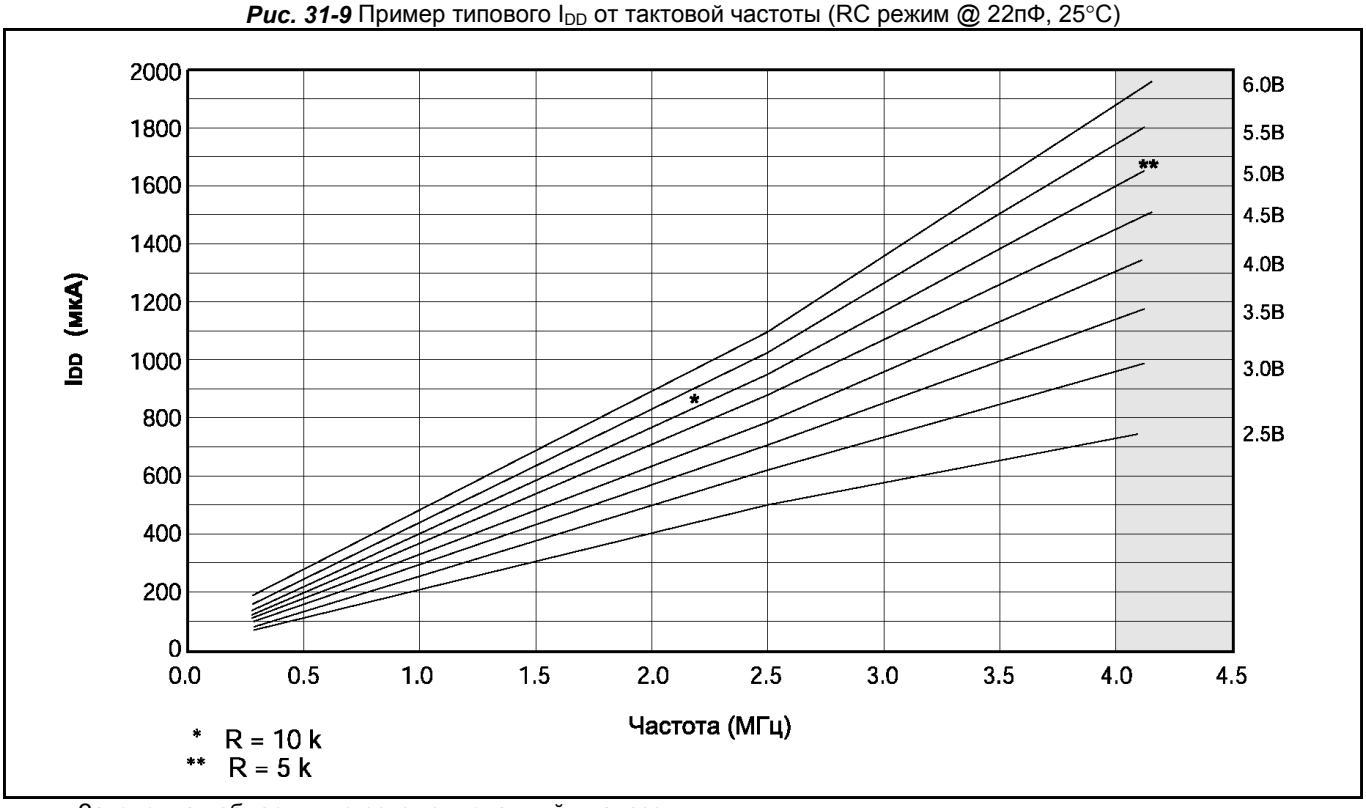
I_{DD} - ток потребления микроконтроллера в нормальном режиме работы. Измерения проводились, когда все порты ввода/вывода настроены как входы и имеют низкий или высокий логический уровень (нет выводов с неопределенным уровнем и выводов с подключенной нагрузкой).

Измерения для построения диаграмм проводилась на автоматизированном оборудовании Microchip DCS. Оборудование не вносит дополнительной паразитной емкости и токов утечки, что позволяет выполнять измерения достаточно точно.

31.3.2.1 Выбор параметров RC цепочки

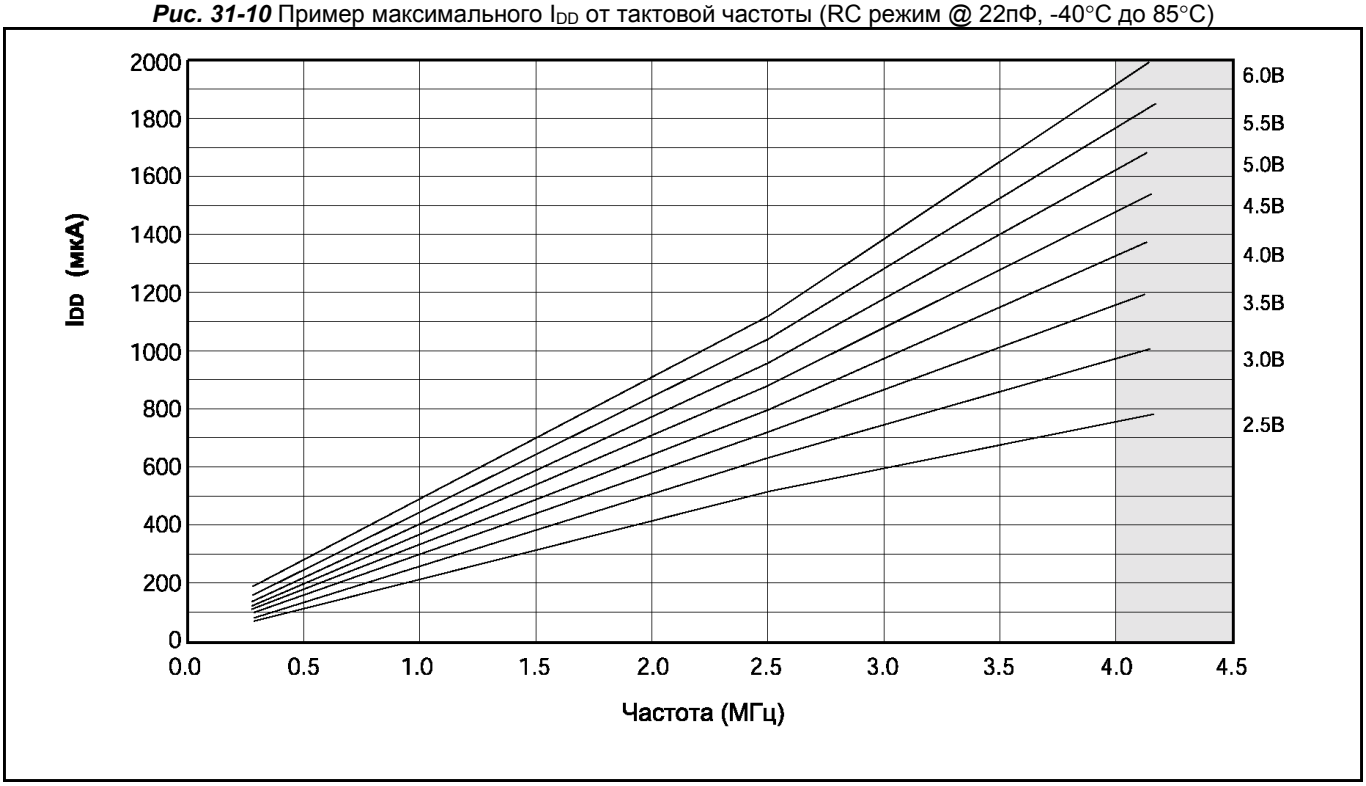
Для RC генератора оборудование DCS выбирает значение резистора и конденсатора, а затем изменяется напряжение в рабочем диапазоне. Из-за изменения напряжения питания изменяется тактовая частота генератора. Для постоянного значения RC при увеличении V_{DD} тактовая частота тоже увеличивается. После цикла испытаний изменяются значения R,C и измерения повторяются. Каждая точка в графике отображает определенное напряжение питания и конкретные значения R, C.

Рис. 31-9 Пример типового I_{DD} от тактовой частоты (RC режим @ 22пФ, 25°C)



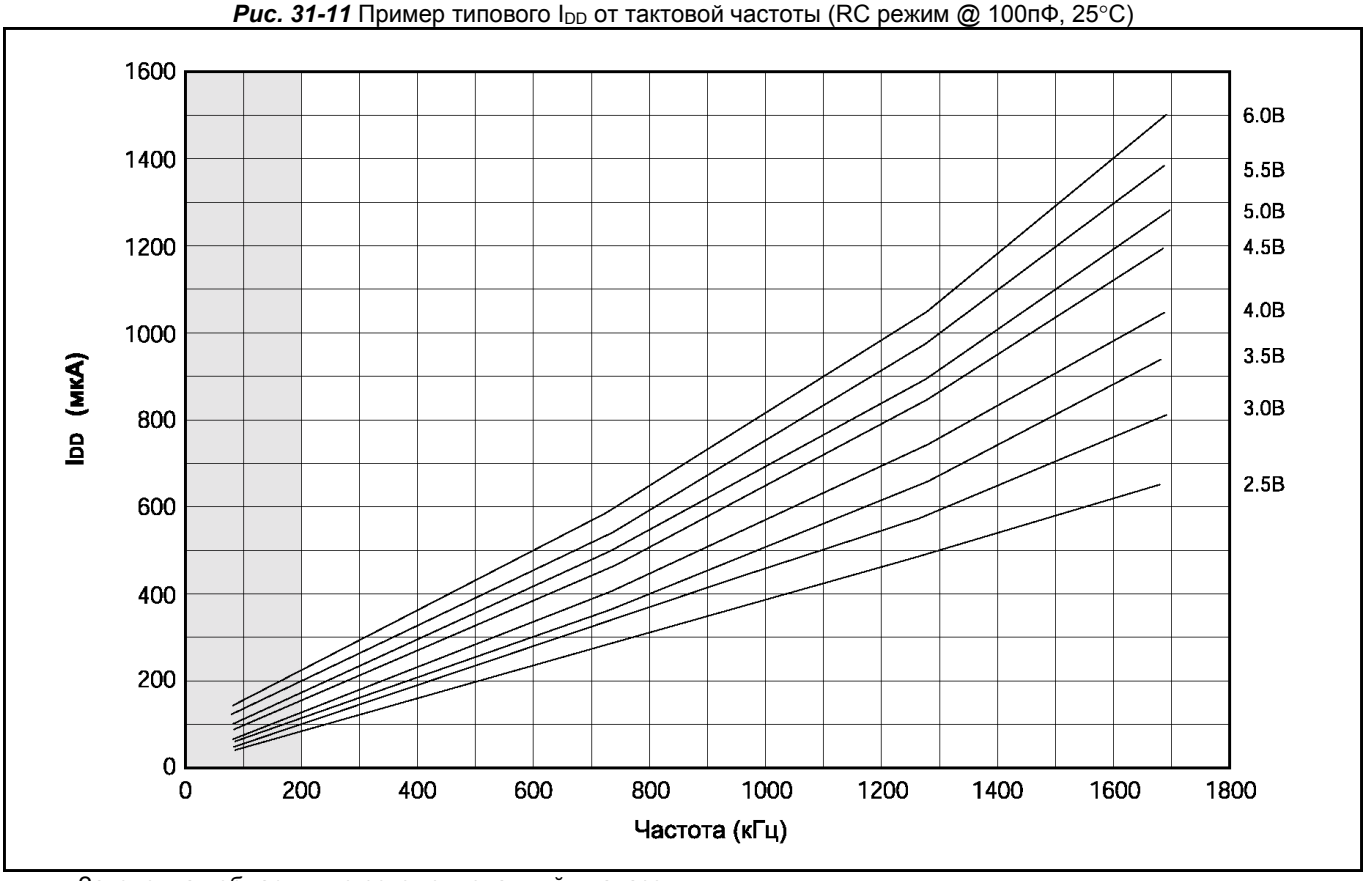
Затененная область - не рекомендованный диапазон.

Рис. 31-10 Пример максимального I_{DD} от тактовой частоты (RC режим @ 22пФ, -40°C до 85°C)



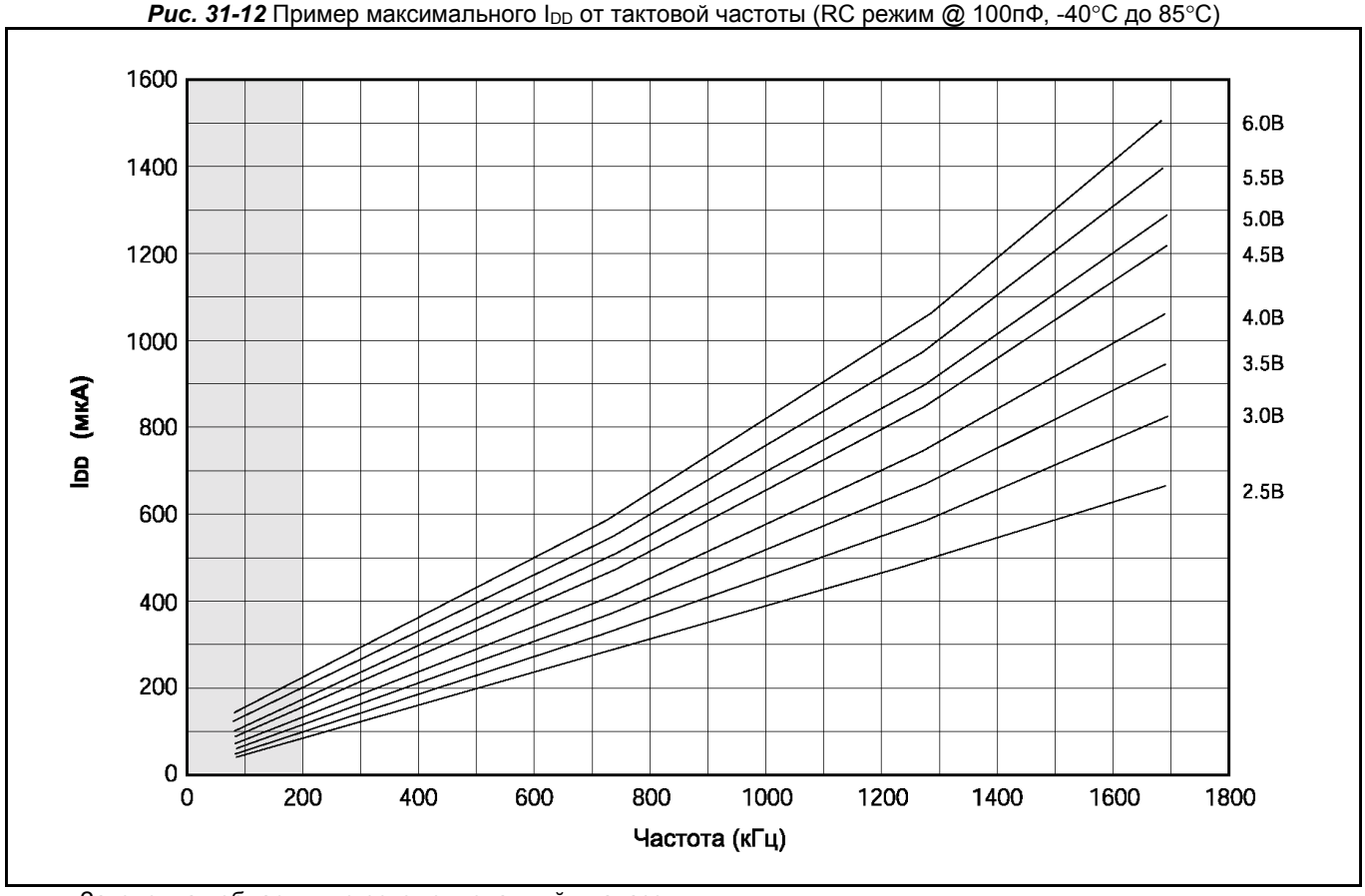
Затененная область - не рекомендованный диапазон.

Рис. 31-11 Пример типового I_{DD} от тактовой частоты (RC режим @ 100пФ, 25°C)



Затененная область - не рекомендованный диапазон.

Рис. 31-12 Пример максимального I_{DD} от тактовой частоты (RC режим @ 100пФ, -40°C до 85°C)



Затененная область - не рекомендованный диапазон.

Рис. 31-13 Пример типового I_{DD} от тактовой частоты (RC режим @ 300пФ, 25°C)

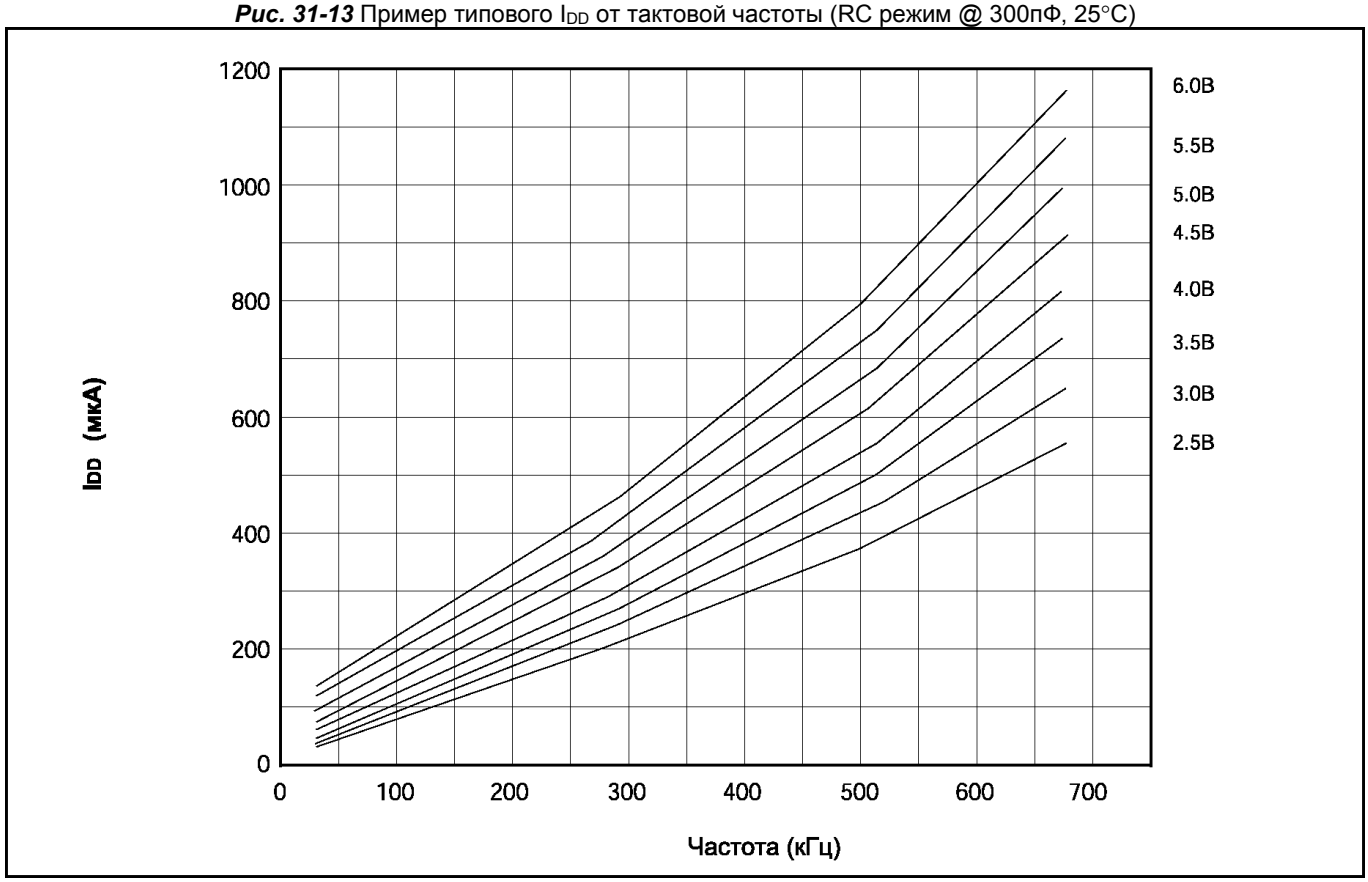


Рис. 31-14 Пример максимального I_{DD} от тактовой частоты (RC режим @ 300пФ, -40°C до 85°C)

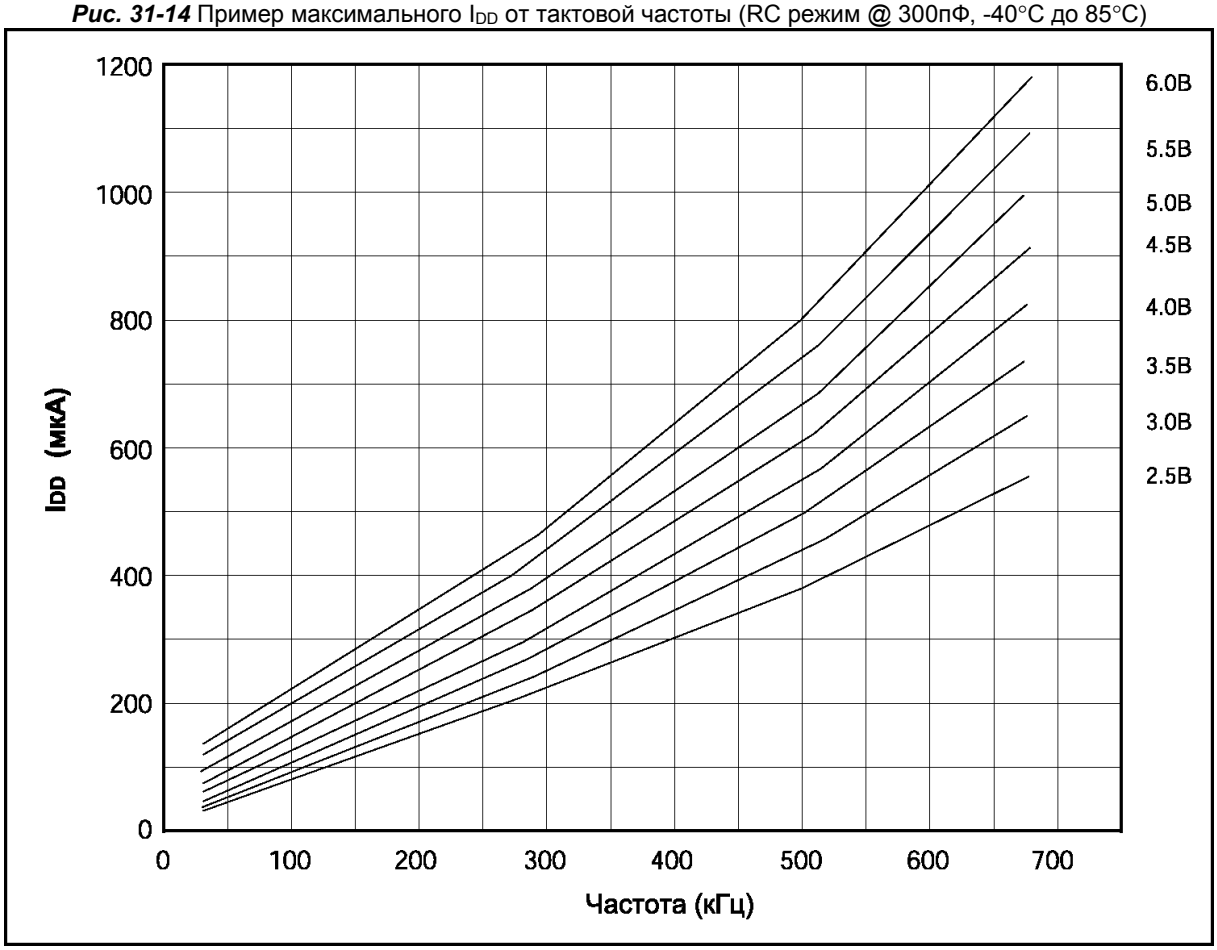
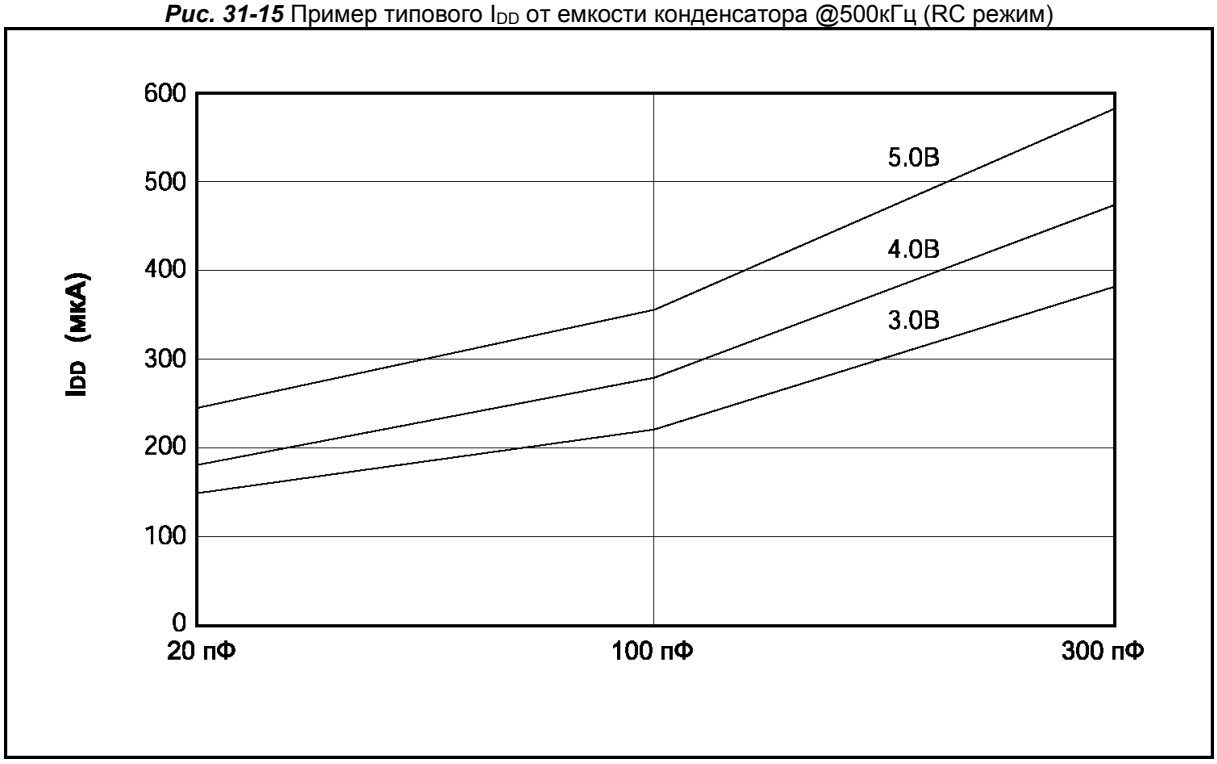


Рис. 31-15 Пример типового I_{DD} от емкости конденсатора @500кГц (RC режим)



31.3.2.2 Испытания в режиме генератора с кварцевым резонатором

В автоматизированной системе проверки предусмотрено несколько стандартных кварцевых резонаторов, эти резонаторы мультиплицируются к соответствующим выводам микроконтроллера. Емкость нагрузочных конденсаторов и напряжение питания подбирается таким образом, чтобы получить наилучшие характеристики (ток потребления, форма сигнала генератора, время запуска генератора), а затем измеряется ток потребления в зависимости от напряжения питания. Подключается следующий резонатор и испытания проводятся повторно.

Рис. 31-16 Пример типового I_{DD} от тактовой частоты (LP режим, 25°C)

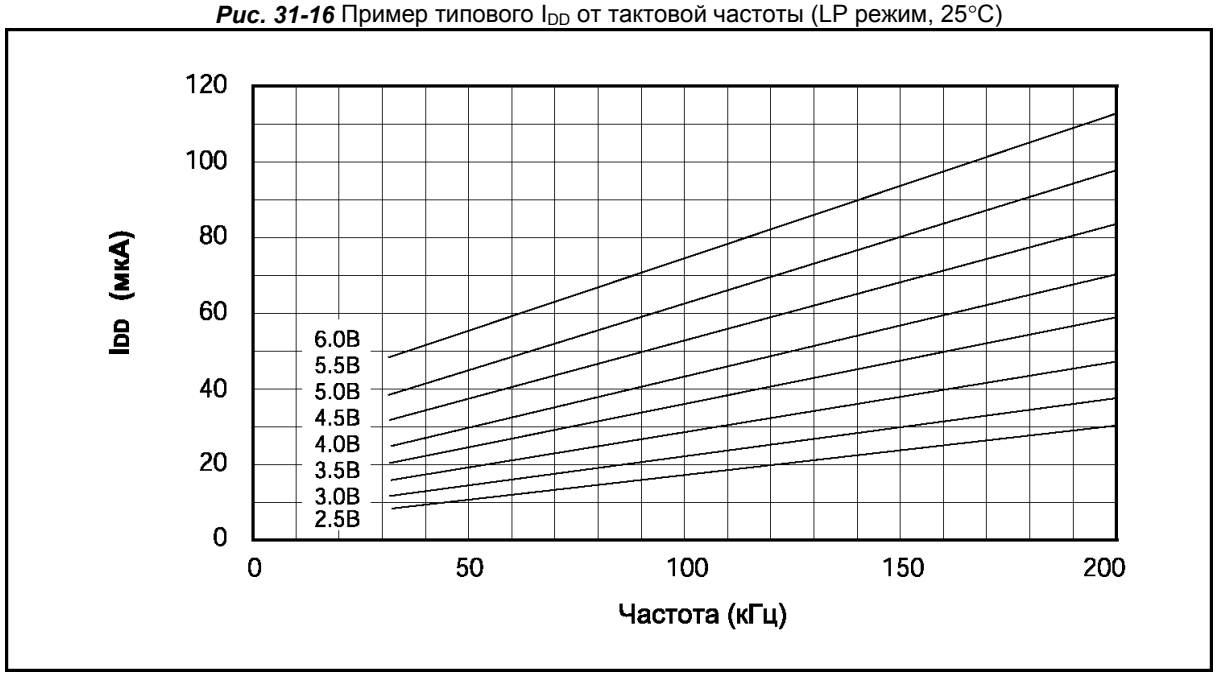


Рис. 31-17 Пример максимального I_{DD} от тактовой частоты (LP режим, -40°C до 85°C)

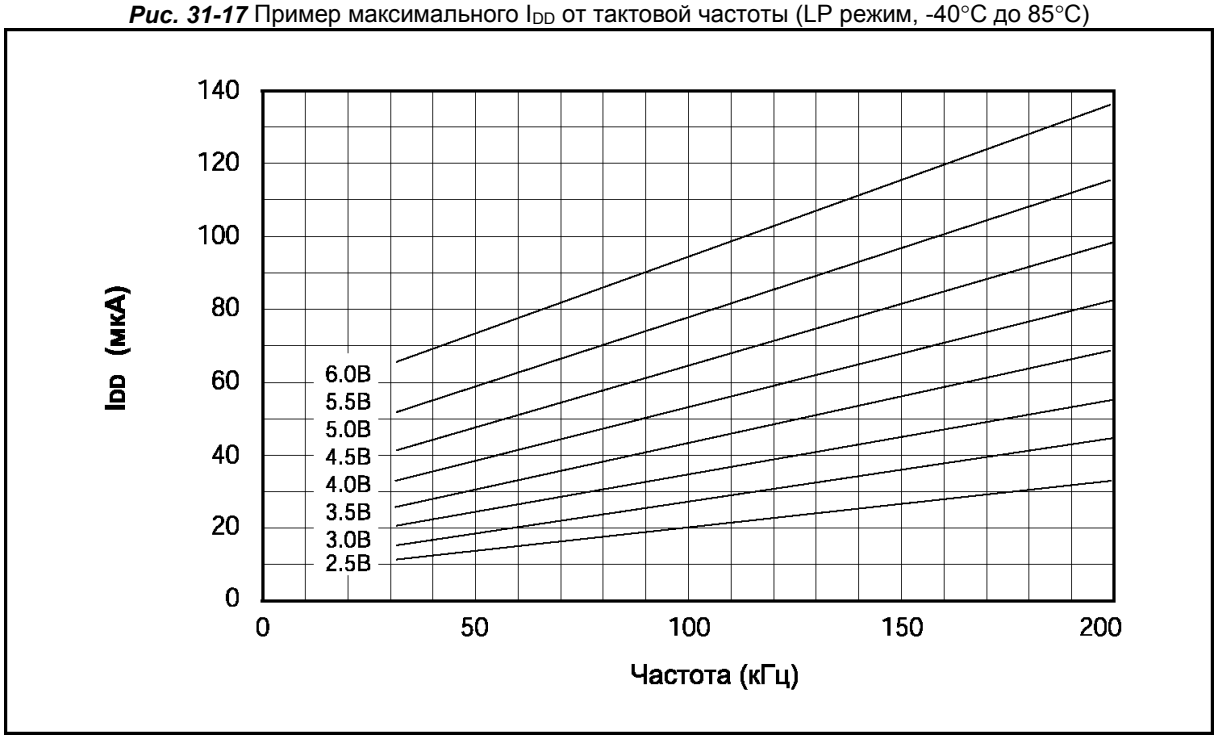


Рис. 31-18 Пример типового I_{DD} от тактовой частоты (XT режим, 25°C)

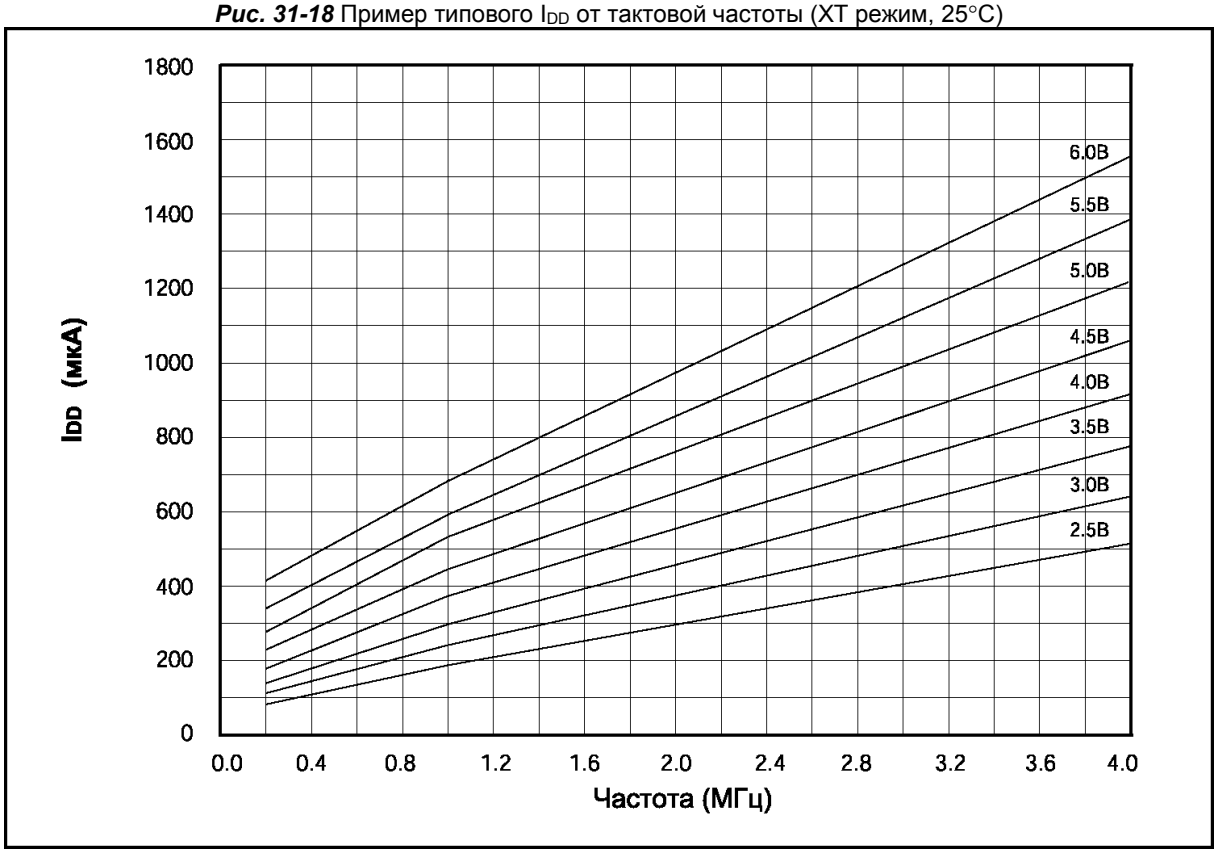


Рис. 31-19 Пример максимального I_{DD} от тактовой частоты (XT режим, -40°C до 85°C)

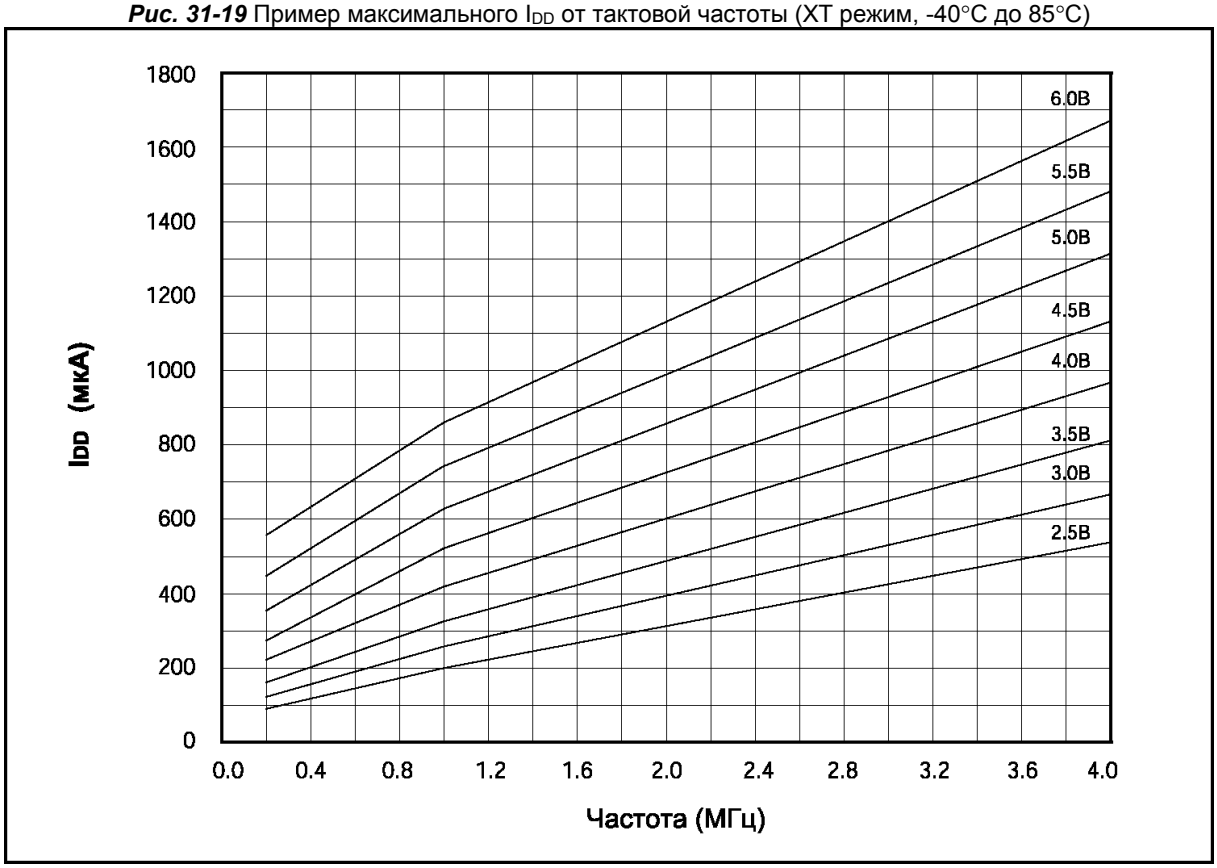


Рис. 31-20 Пример типового I_{DD} от тактовой частоты (HS режим, 25°C)

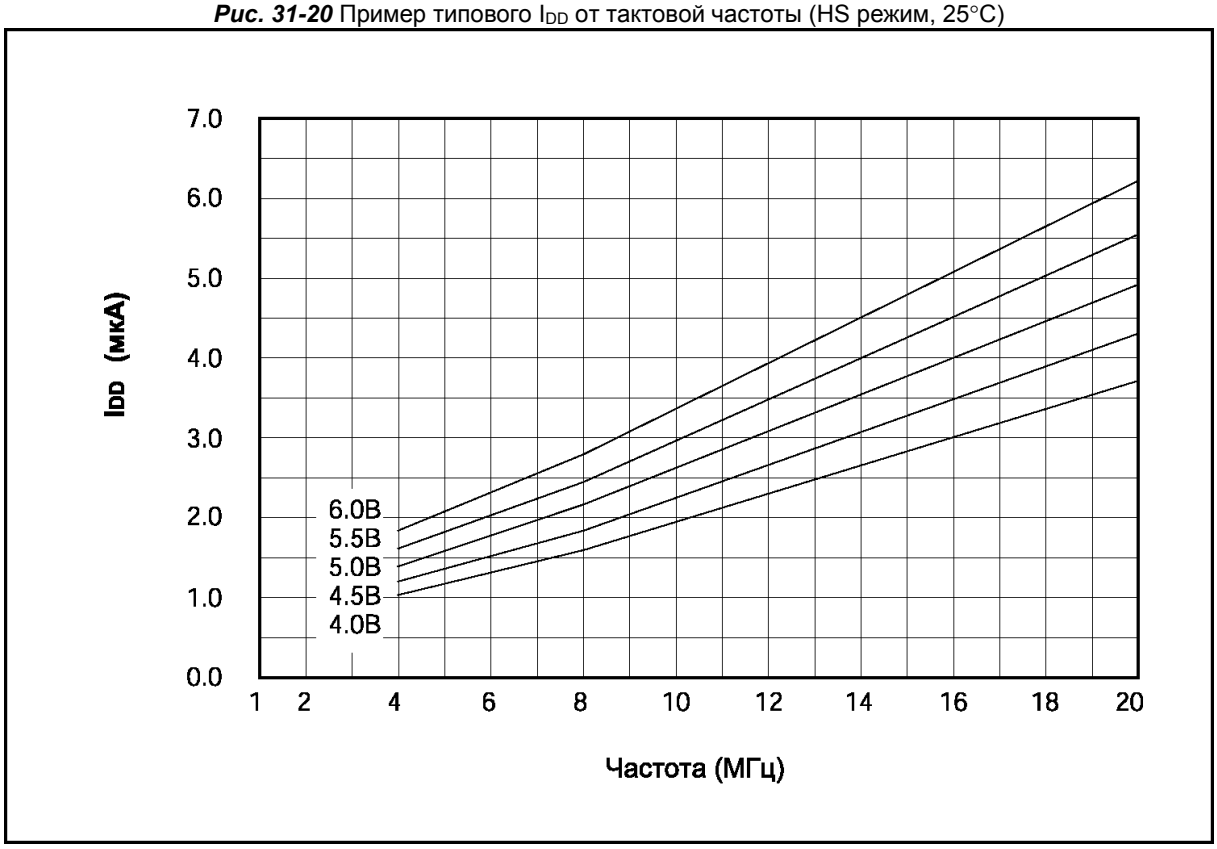
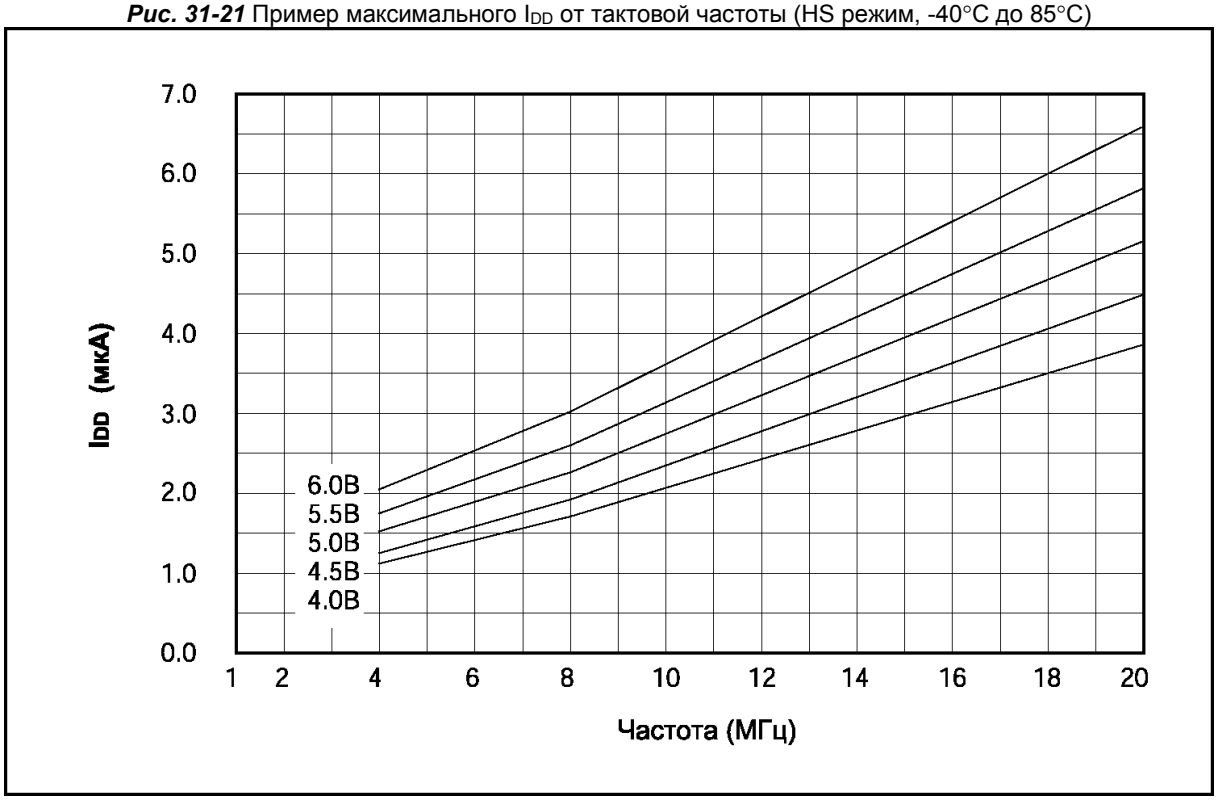


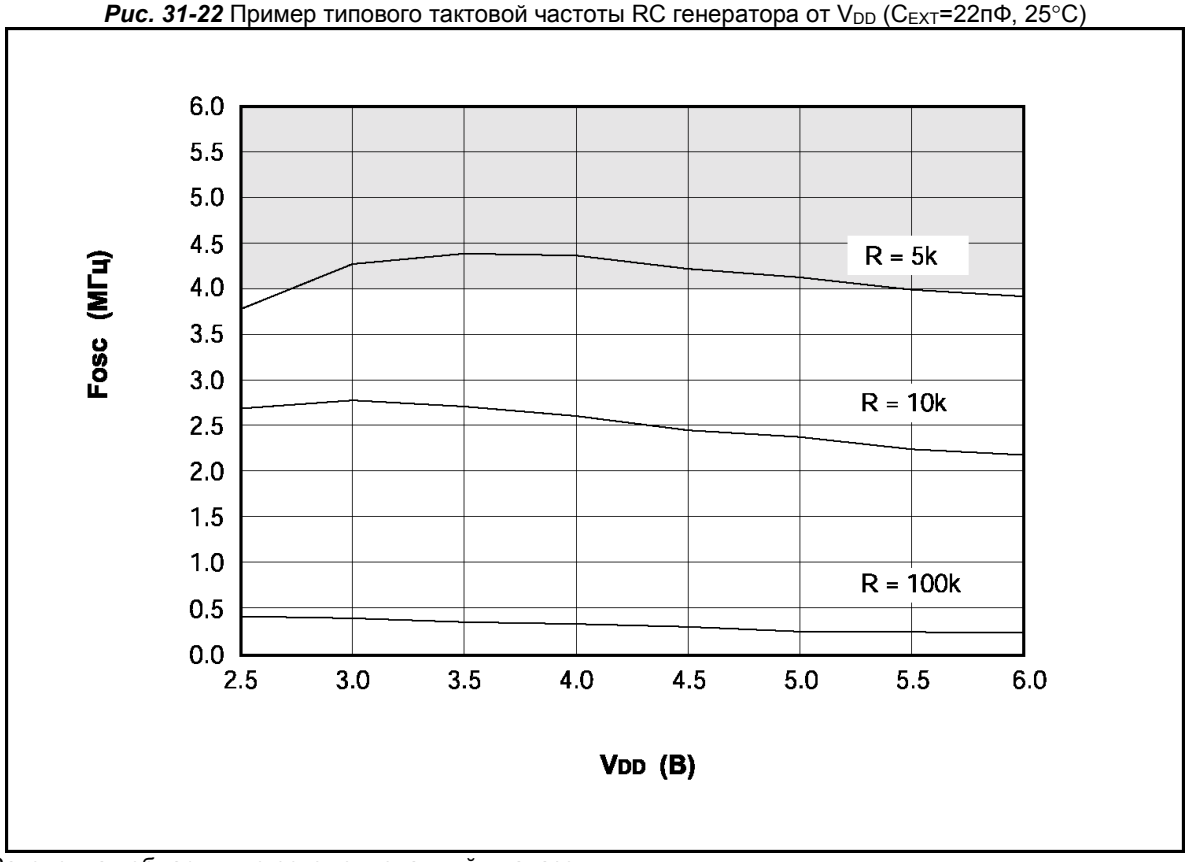
Рис. 31-21 Пример максимального I_{DD} от тактовой частоты (HS режим, -40°C до 85°C)



31.3.3 Частота RC генератора

В таблицах и графиках данной главы представлены зависимости частоты тактового RC генератора от напряжения питания V_{DD}. Испытания проводились при фиксированных значениях R, C и изменяющемся напряжении питания. В таблице показано типовое значение тактовой частоты при определенных значениях R и C, напряжении питания 5В.

Рис. 31-22 Пример типового тактовой частоты RC генератора от V_{DD} (C_{EXT}=22пФ, 25°C)



Затененная область - не рекомендованный диапазон.

Рис. 31-23 Пример типового тактовой частоты RC генератора от V_{DD} (C_{EXT}=100пФ, 25°C)

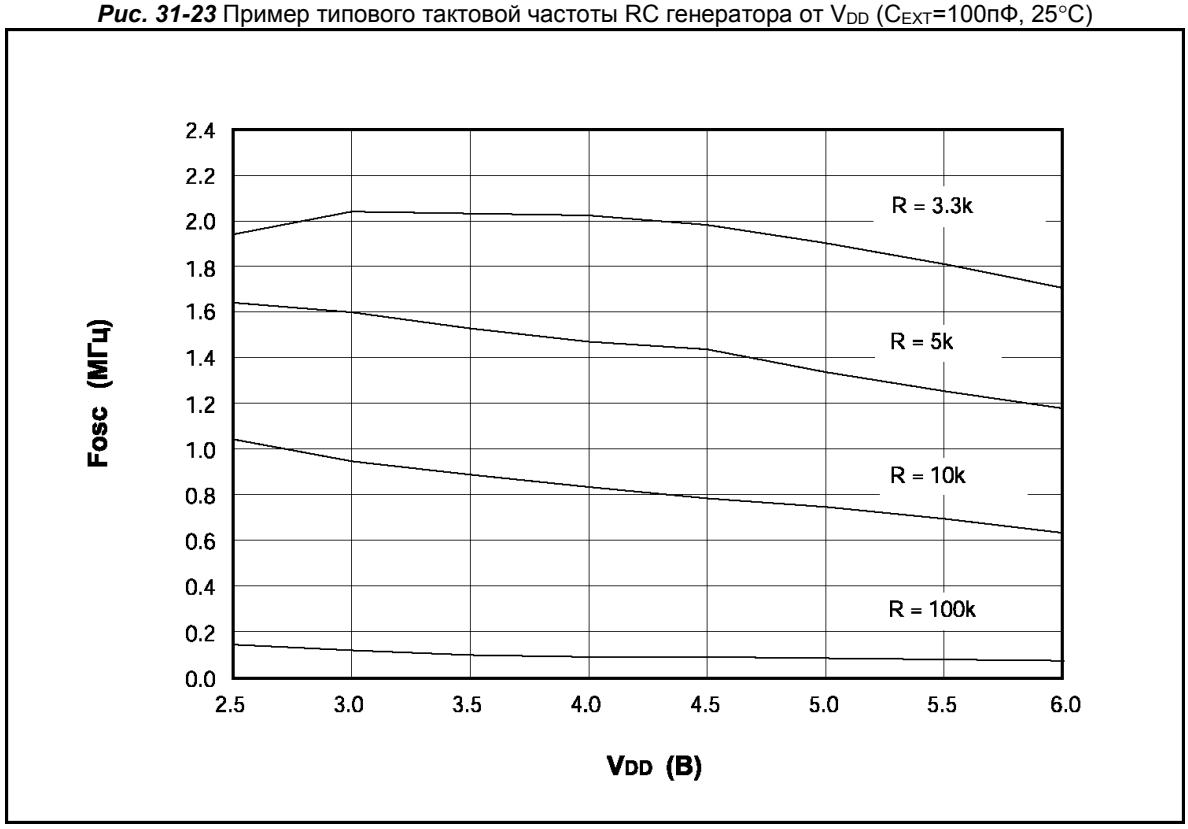


Рис. 31-24 Пример типового тактовой частоты RC генератора от V_{DD} (C_{EXT}=300пФ, 25°C)

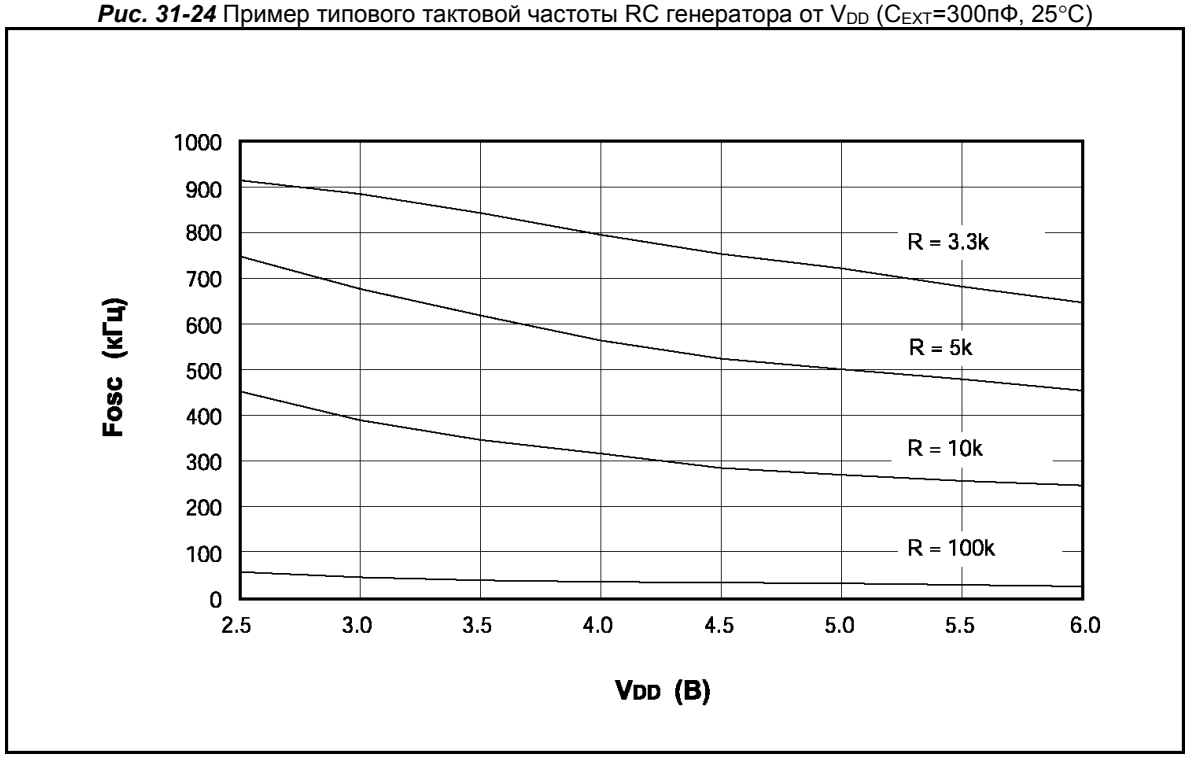


Таблица 31-1 Пример частоты RC генератора

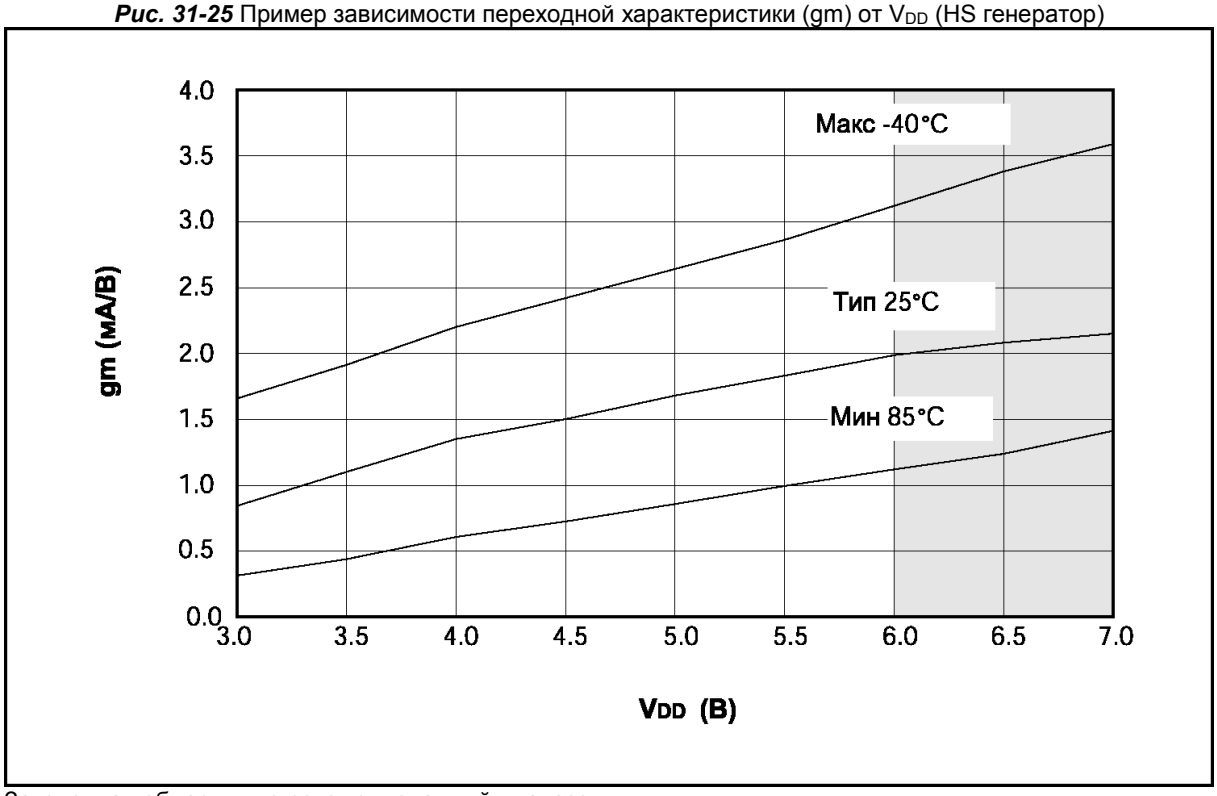
C_{EXT}	R_{EXT}	Среднее значение	
		F_{OSC} @ 5В, 25°C	
22 пФ	5к	4.12 МГц	± 1.4%
	10к	2.35 МГц	± 1.4%
	100к	268 кГц	± 1.1%
100 пФ	3.3к	1.80 МГц	± 1.0%
	5к	1.27 МГц	± 1.0%
	10к	688 кГц	± 1.2%
	100к	77.2 кГц	± 1.0%
300 пФ	3.3к	707 кГц	± 1.4%
	5к	501 кГц	± 1.2%
	10к	269 кГц	± 1.6%
	100к	28.3 кГц	± 1.1%

Отклонение в процентах, представленное в таблицах, является нормально распределенным.
Обозначение ±3 - стандартное отклонение от среднего значения при V_{DD} = 5В.

31.3.4 Переходная характеристика генератора

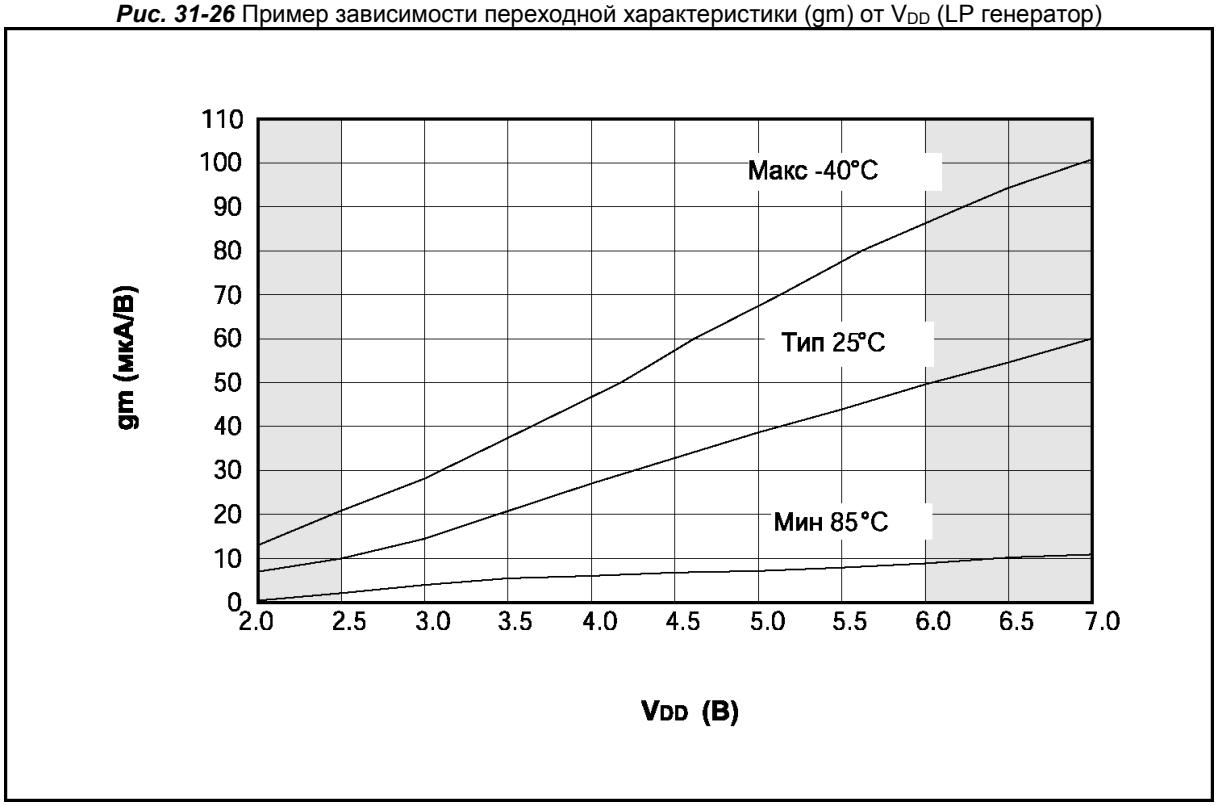
Переходная характеристика генератора непосредственно связана с коэффициентом усиления инвертора. При увеличении значения переходной характеристики - увеличивается ток потребления, возможно получения более высокой тактовой частоты и уменьшения времени запуска генератора.

Рис. 31-25 Пример зависимости переходной характеристики (gm) от V_{DD} (HS генератор)



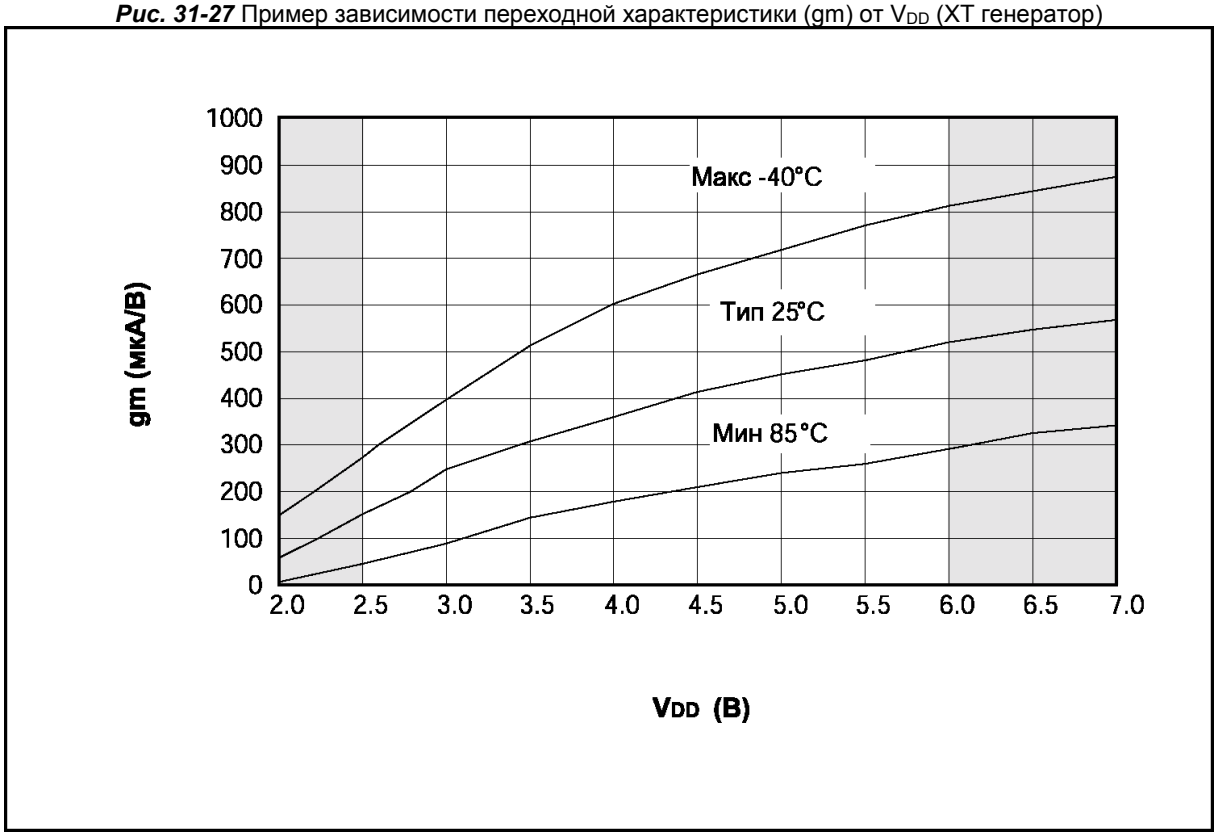
Затененная область - не рекомендованный диапазон.

Рис. 31-26 Пример зависимости переходной характеристики (gm) от V_{DD} (LP генератор)



Затененная область - не рекомендованный диапазон.

Рис. 31-27 Пример зависимости переходной характеристики (gm) от V_{DD} (XT генератор)



Затененная область - не рекомендованный диапазон.

31.3.5 Время запуска генератора

На графиках показано время запуска генератора для определенного напряжения питания, типа резонатора и емкости нагрузочных конденсаторов.

Рис. 31-28 Пример зависимости времени запуска генератора от V_{DD} (LP режим, 25°C)

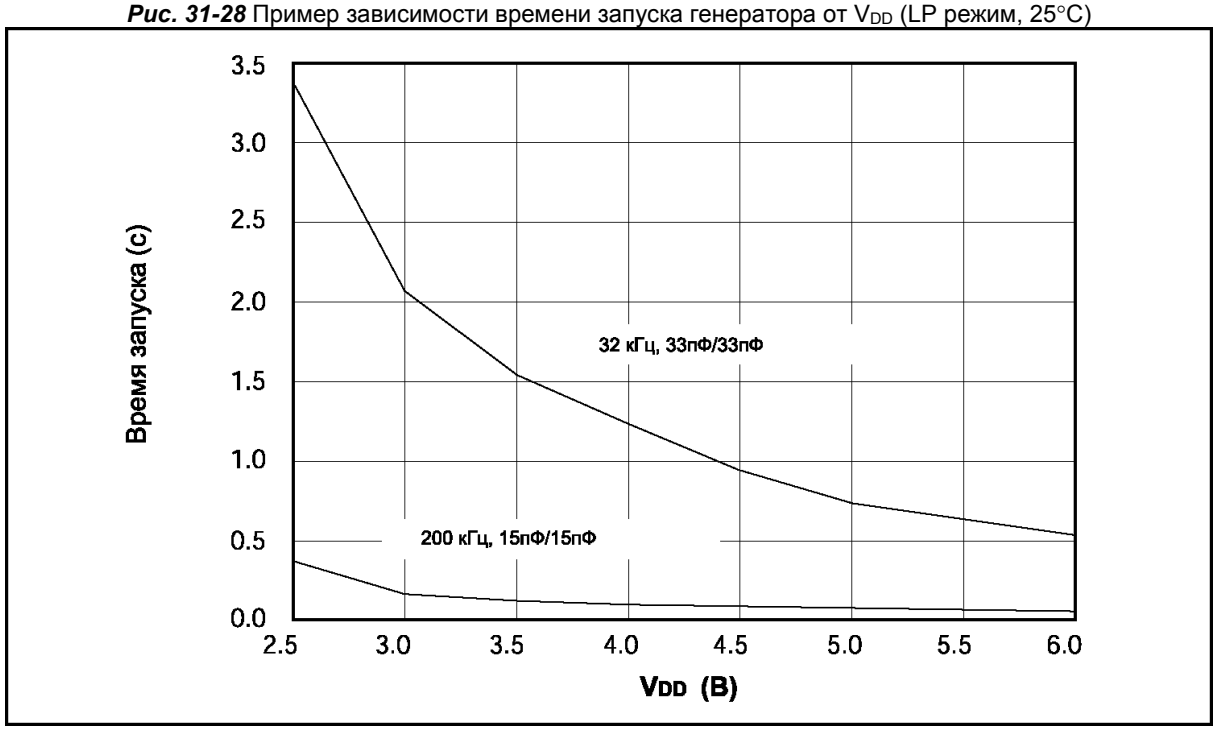


Рис. 31-29 Пример зависимости времени запуска генератора от V_{DD} (HS режим, 25°C)

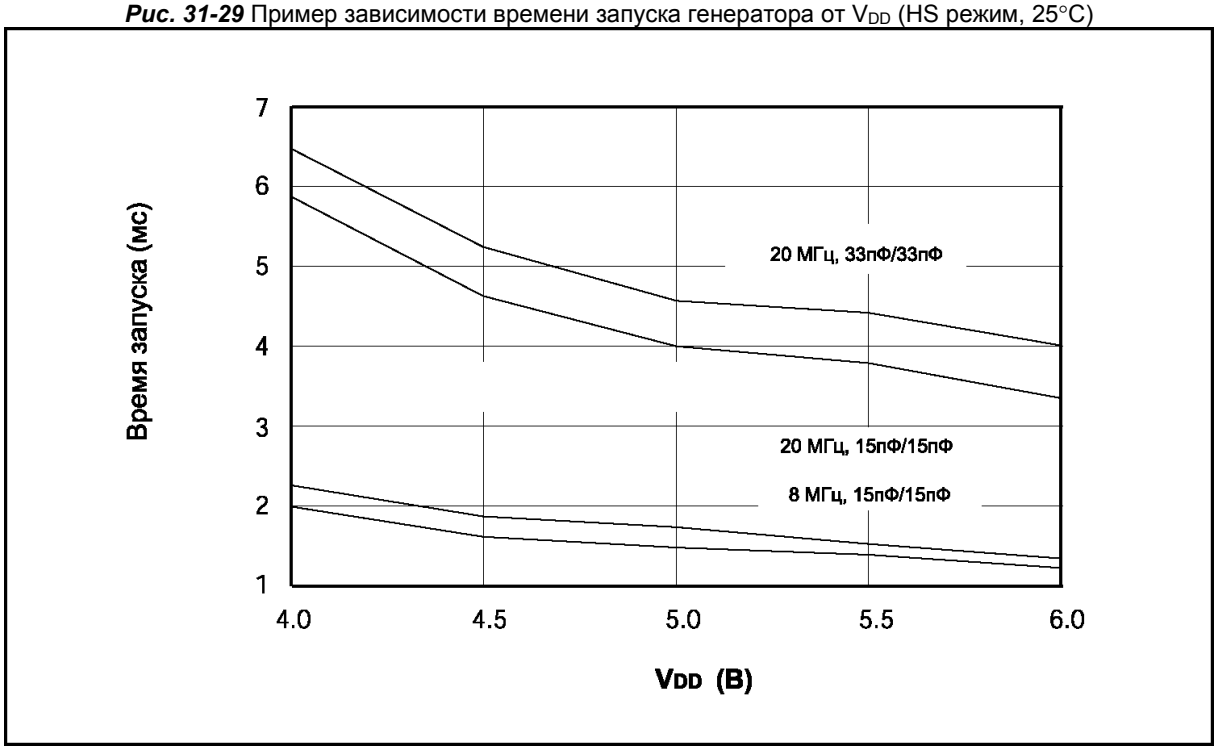
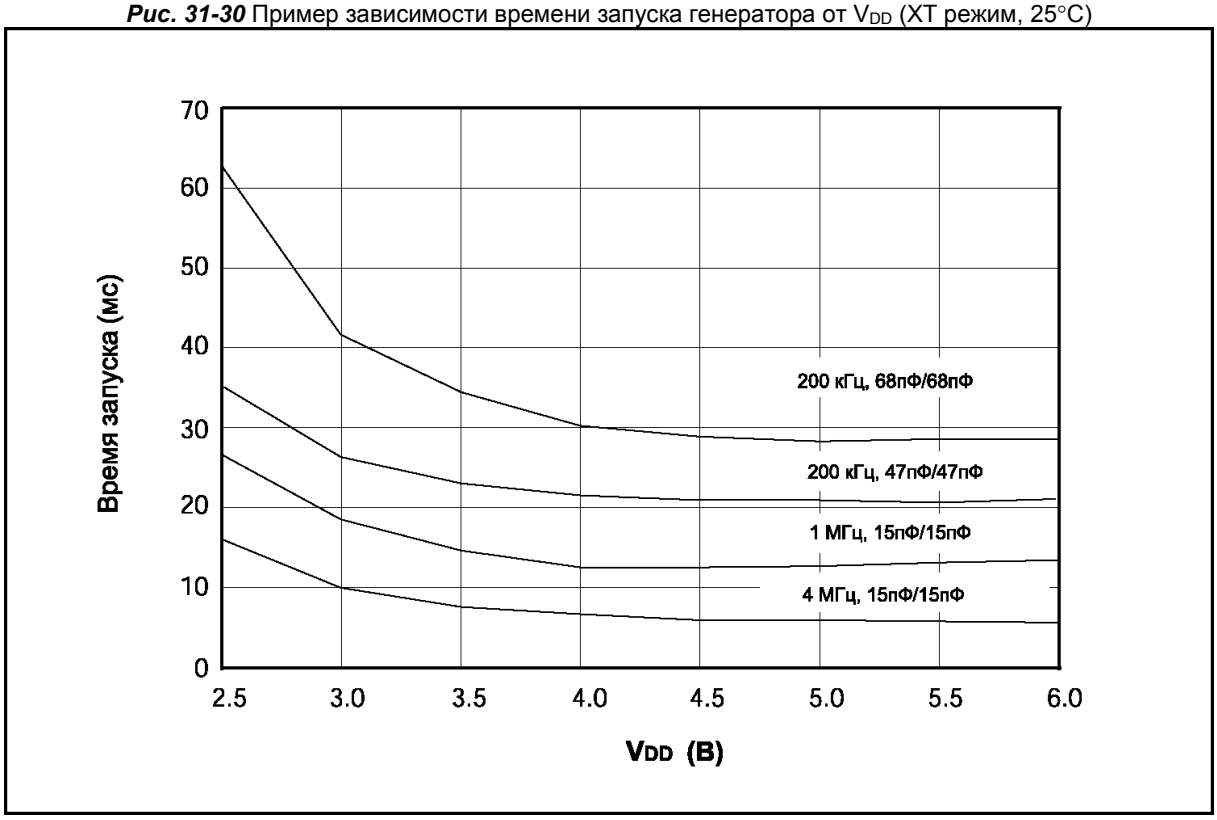


Рис. 31-30 Пример зависимости времени запуска генератора от V_{DD} (XT режим, 25°C)



31.3.6 Тестируемые кварцевые резонаторы и значения нагрузочных конденсаторов

В таблице представлены параметры и типы кварцевых резонаторов, используемые при проведении испытаний, а также значения нагрузочных конденсаторов, при которых были получены наилучшие характеристики.

Таблица 31-2 Пример емкости нагрузочных конденсаторов для кварцевых резонаторов

Протестированные диапазоны:			
Режим	**Частота**	**C1 (1)**	**C2 (1)**
LP	32 кГц 200 кГц	33 пФ 15 пФ	33 пФ 15 пФ
XT	200 кГц 1.0 МГц 4.0 МГц	47 - 68 пФ 15 пФ 15 пФ	47 - 68 пФ 15 пФ 15 пФ
HS	4.0 МГц 8.0 МГц 20.0 МГц	15 пФ 15 - 33 пФ 15 - 33 пФ	15 пФ 15 - 33 пФ 15 - 33 пФ
Примечание. Большая емкость увеличивает стабильность генератора, но увеличивается и время запуска. Последовательный резистор Rs может потребоваться в HS и XT режиме для предотвращения возбуждения резонатора на низкой частоте. Значения емкости конденсаторов являются оценочными, т.к. каждый резонатор имеет собственные характеристики. Проконсультируйтесь у производителя резонаторов для правильного подбора внешних компонентов.			
Применяемые резонаторы:			
32.768 кГц	Epson C001R32.768-A		±20 PPM
200 кГц	STD XTL 200.000 kHz		±20 PPM
1.0 МГц	ECS ECS-10-13-1		±50 PPM
4.0 МГц	ECS ECS-40-20-1		±50 PPM
8.0 МГц	Epson CA-301 8.000M-C		±30 PPM
20.0 МГц	Epson CA-301 20.000M-C		±30 PPM

31.3.7 Пример времени УФ стирания EPROM памяти

Время УФ стирания EPROM памяти зависит от геометрического размера ячеек и технологии изготовления памяти. В таблице 31-3 представлено ожидаемое время УФ стирания памяти.

Таблица 31-3 Пример рекомендованного времени УФ стирания EPROM памяти

Микроконтроллер	**Длина волны**	**Интенсивность (мкВт/см^2)**	**Расстояние до УФ лампы (дюймы)**	**Время стирания (мин.)(1)**
1	2537	12.000	1	15 - 20
2	2537	12.000	1	20
3	2537	12.000	1	40
4	2537	12.000	1	60

Примечание 1. Если указанные условия не выполнены, то время стирание будет другим.

31.4 Ответы на часто задаваемые вопросы

На момент выполнения перевода в оригинальной технической документации вопросы отсутствовали. Если у вас есть вопрос, задайте его, написав нам письмо по адресу support@microchip.ru.

31.5 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило, примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные характеристиками микроконтроллеров PICmicro MCU:

Документ	Номер
В настоящее время документы не подготовлены	

Раздел 32. Поддержка разработчиков

Содержание

32.1 Введение

Микроконтроллеры PICmicro обеспечены большим спектром аппаратных и программных инструментальных средств проектирования.

Основные инструментальные средства:

- Интегрированная среда проектирования MPLAB IDE с полнофункциональным редактором.
- Ассемблер/Компилятор/Линкер:
  - Ассемблер MPASM;
  - Компиляторы MLAB-C17 и MPLAB-C18;
  - Линкер MPLINK/ Организатор библиотек MPLIB.
- Программный симулятор MLAB-SIM.
- Эмуляторы:
  - Внутрисхемный эмулятор PICMASTER/PICMASTER-CE;
  - ICEPIC - недорогой эмулятор с возможностью указания точки остановки.
- Программаторы:
  - Универсальный программатор PRO MATE II;
  - Недорогой программатор PICSTART для начала работы.

Дополнительные инструментальные средства:

- Другие программные средства:
  - Среда проектирования *fuzzy*TECH-MP;
  - Генератор кода MP-DriveWay.
- Демонстрационные платы:
  - SIMICE;
  - PICDEM-1;
  - PICDEM-2;
  - PICDEM-3;
  - PICDEM-14A.

Минимальная конфигурация среды проектирования MPLAB IDE содержит: ассемблер MPASM; программный симулятор MPLAB-SIM. Другие инструментальные средства могут быть добавлены при установке MPLAB IDE, что позволяет использовать одну платформу для разработки проекта: от написания исходного текста программы до симуляции/эмуляции работы микроконтроллера и программирования.

Примечание. Последняя версия программного обеспечения может быть свободно загружена с узлов технической поддержки **www.microchip.com** и **www.microchip.ru**.

Для микросхем фирмы Microchip выпускается большое число инструментальных средств другими производителями.

32.2 Интегрированная среда проектирования (IDE)

Основной набор инструментальных средств под знаком IDE называется MPLAB, в котором удачно объединяются все необходимые инструментальные средства, позволяя минимизировать время изучения нового интерфейса. В состав MPLAB IDE входит:

- Редактор исходного текста программы;
- Организатор проекта;
- Компиляторы исходного текста программы;
- Программный симулятор микроконтроллеров;
- Внутрисхемный эмулятор;
- Программатор.

MPLAB работает на PC совместимых компьютерах с установленной операционной системой Windows 3.x/9x. MPLAB IDE позволяет выполнять полный цикл проектирования без необходимости использования других программ.

32.2.1 MPLAB

Программное обеспечение MPLAB-IDE предназначено для разработки программного обеспечения 8-разрядных микроконтроллеров PICmicro, работающее под управлением операционной системы Windows.

Основные характеристики MPLAB-IDE:

- Многофункциональные возможности:
  - редактор;
  - симулятор;
  - программатор (приобретается отдельно);
  - эмулятор (приобретается отдельно).
- Полнофункциональный редактор.
- Организатор проекта.
- Настройка панелей инструментов и параметров отображения.
- Строка состояния.
- Интерактивная помощь.

MPLAB-IDE позволяет Вам:

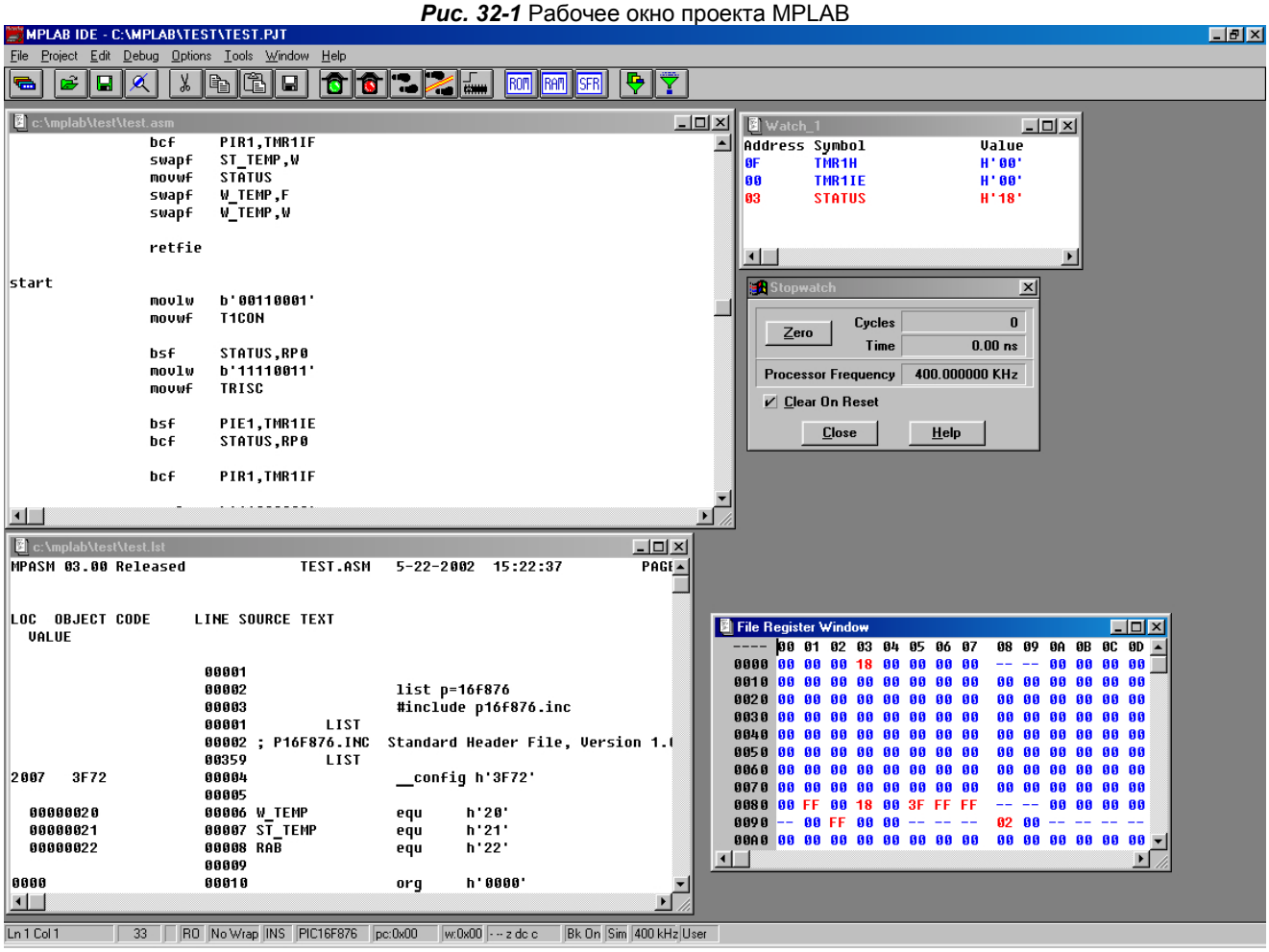
- Редактировать исходные файлы, написанные на языке ассемблера или C.
- Быстро выполнять трансляцию и компиляцию проекта, автоматически загружая параметры используемого микроконтроллера PICmicro.
- Выполнять отладку программы с использованием:
  - исходных файлов;
  - листинга программы;
  - объектного кода.
- Поддерживать до 4-х эмуляторов на одном PC.
- Выполнять программу в режиме реального времени или по шагам на основе:
  - исходных файлов;
  - листинга программы;
  - объектного кода.

Однотипная работа инструментальных модулей интегрированной среды проектирования MPLAB-IDE позволяет легко перейти от программного симулятора MPLAB-SIM к использованию полнофункционального эмулятора.

На рисунке 32-1 представлен пример рабочего стола MPLAB-IDE, который может содержать следующие элементы:

- Область инструментов, с возможностью настройки под конкретный проект;
- Строка состояния с информацией о текущем режиме работы;
- Большое число рабочих окон, например:
  - исходный текст программы;
  - листинг программы (особенно полезна для программ, написанных на языке C);
  - память данных;
  - окно секундомера для вычисления длительности выполнения программы;
  - окно с переменными;
- Поддержка программатора (в этом случае доступно меню PRO MATE).

Рис. 32-1 Рабочее окно проекта MPLAB



32.3 Поддержка языков программирования

Микроконтроллер работает в устройстве в соответствии с загруженной программой. Программа может быть написана на одном из доступных языков программирования для данного микроконтроллера. В настоящее время MPLAB IDE поддерживает два языка программирования Microchip:

- Ассемблер Microchip (MPASM);
- С компилятор Microchip (MPLAB-C);

Другие компиляторы, поддерживающие генерацию *.COD файлов, могут работать совместно со средой проектирования MPLAB-IDE.

32.3.1 Ассемблер MPASM

MPASM - полнофункциональный универсальный макроассемблер для всех семейств микроконтроллеров PICmicro. Ассемблер может генерировать шестнадцатиразрядный файл пригодный для записи в микроконтроллер или формировать перемещаемые объектные файлы для линкера MPLINK.

MPASM имеет интерфейс командной строки и оконный интерфейс, работает под управлением операционной системы Windows 3.X и выше, может работать как автономное приложение. MPASM генерирует объектные файлы, шестнадцатеричные HEX файлы в стандарте Intel, файл карты памяти (для детализации использования памяти микроконтроллера), файл листинга программы (текст программы совмещен с кодами микроконтроллера) и файл отладки для MPLAB-IDE.

Особенности MPASM:

- MPASM и MPLINK интегрированы в MPLAB-IDE;
- MPASM поддерживает систему макрокоманд, упрощающих написание текста программы;
- Позволяет выполнять компиляцию условных блоков текста программы;
- Директивы MPASM дают возможность управлять компиляцией исходного текста программы.

32.3.2 С компиляторы MPLAB-C

MPLAB-C - полнофункциональный ANSI 'C' компилятор с интегрированной средой обработки для микроконтроллеров семейства PIC16CXXX. Для упрощения отладки текста программы компиляторы обеспечивают интеграцию в средства проектирования с передачей информации об используемых переменных в формате совместимом с MPLAB-IDE.

32.3.3 Линкер MPLINK

MPLINK - линкер перемещаемых объектных файлов, сгенерированных программами MPASM и MPLAB-C. Линкер MPLINK, входящий в состав MPLAB-C v2.00, может использоваться только с этой или более поздней версией компилятора.

Линкер выполняет связь объектных файлов с предварительно компилированными файлами. Управление объединением объектных файлов выполняется за счет файлов сценария и параметров, указанных в командной строке. MPLINK гарантирует сохранения всех символьных меток и соответствие необходимого объема памяти программ и памяти данных выбранному микроконтроллеру.

MPLINK собирает отдельно компилированные объектные модули (полученные от MPLAB-C и MPASM) в один файл кода программы. Адреса данных и расположение функций будет назначено по завершению работы MPLINK. Это означает, что Вы можете рекомендовать размещение кода программы в определенные области памяти программ или данных, не указывая физические адреса.

После анализа доступной памяти программ и памяти данных в микроконтроллере выполняется попытка размещения функций программы и назначения переменных в ОЗУ. Если требуется слишком много памяти программ или данных, то генерируется сообщение об ошибке.

MPLINK предоставляет дополнительную гибкость, позволяя назначать переменные с временным хранением данных. Это дает возможность различным подпрограммам использовать одну и туже область памяти данных (подпрограммы не зависят от этих данных, они используются только в вычислениях).

32.3.4 Организатор библиотек MPLIB

MPLIB - организатор библиотек предварительно откомпилированных исходных файлов, которые нужно использовать с MPLINK (MPASM v2.0, MPASMWIN v2.0, MPLAB-C v2.0). Когда подпрограмма библиотечного файла вызывается из исходного файла, в приложение будет включен только необходимый модуль. Это позволяет эффективно использовать большие библиотеки в различных приложениях. MPLIB управляет созданием и изменением библиотечных файлов.

- Использование библиотек позволяет упростить компиляцию проекта, т.к. библиотеки содержат много объектных файлов, то необходимо указывать только имя файла библиотеке вместо нескольких объектных файлов.
- При компоновке включаются только те объектные модули, которые используются в программе.
- Если необходимо использовать дополнительную функцию, включенную в библиотеку, то не требуется выполнять перенастройку параметров компиляции.
- В библиотечных файлах могут группироваться функционально схожие объектные файлы. Например, назначение библиотеки math.lib более понятно, чем назначение объектных файлов power.o, ceiling.o и floor.o.

32.3.5 Программный симулятор MPLAB-SIM

Симулятор MPLAB-SIM позволяет проследить выполнение программы микроконтроллеров PICmicro на уровне команд по шагам или в режиме анимации. В зависимости от сложности проекта и требованиям к срокам разработки следует делать выбор между эмулятором и программным симулятором. На любой команде выполнение программы может быть остановлено для проверки и изменения памяти. Функции стимула позволяют моделировать сигнал с логическими уровнями на входах микроконтроллера. MPLAB-SIM полностью поддерживает символьную отладку, используя MPLAB-C и MPASM. MPLAB-SIM является доступным и удобным средством отладки программ микроконтроллеров PICmicro.

32.5 Поддержка аппаратных эмуляторов

Microchip предлагает два эмулятора: высококачественный PICMASTER и недорогой ICEPIC. Оба эмулятора имеют хорошее соотношение цена/качество и выбор эмулятора должен зависеть от ваших требований к набору его функций. Для разработчиков, выполняющих проекты на нескольких микроконтроллерах PICmicro, наиболее выгодным может быть PICMASTER, позволяющий уменьшить время разработки за счет назначения сложных точек остановки.

32.5.1 PICMASTER - высокоэффективный внутрисхемный эмулятор

PICMASTER - универсальный эмулятор компании Microchip для профессиональных разработчиков с полным набором инструментальных средств для микроконтроллеров младшего, среднего и старшего семейства. PICMASTER работает под управлением MPLAB-IDE (редактирование, компиляция и загрузка программы выполняется в одной среде проектирования). Взаимозаменяемые модули позволяют легко перенастроить эмулятор для работы с другим микроконтроллером. Универсальная архитектура PICMASTER предусматривает поддержку новых микроконтроллеров Microchip.

Эмулятор PICMASTER был разработан как система эмуляции (анимации) в реальном масштабе времени с дополнительными возможностями, присутствующих в дорогих инструментальных средствах. Эмуляторы PICMASTER продаются во всем мире, эмуляторы с префиксом CE предназначены для стран ЕС.

32.5.2 ICEPIC - Недорогой внутрисхемный эмулятор для PIC16CXXX

ICEPIC - недорогой эмулятор, предназначенный для однократно программируемых (OTP) 8-разрядных микроконтроллеров среднего семейства PICmicro.

ICEPIC предназначен для PC совместимых компьютеров от моделей 286-AT до Pentium™ с операционной системой Windows 3.x. ICEPIC поддерживает эмуляцию в режиме реального времени под управлением MPLAB-IDE.

ICEPIC разработан компанией Neosoft Inc.

32.6 Поддержка программаторов

Microchip предлагает два вида программаторов. В большинстве случаев достаточен программатор PICSTART Plus. Для профессиональной разработки необходимо использовать PRO MATE II как минимум из-за того, что он позволяет проверить программирование микроконтроллера при минимальном и максимальном напряжении питания V_{DD}.

32.6.1 Универсальный программатор PRO MATE II

Универсальный программатор PRO MATE II может работать автономно и под управлением PC совместимого компьютера с загруженным MPLAB-IDE или специализированной DOS программой.

В PRO MATE II можно указать напряжения V_{DD} и V_{PP}, что позволяет проверить программирование микроконтроллера при минимальном и максимальном напряжении питания. В программатор встроен ЖКИ дисплей для вывода сообщений об ошибках и клавиатура для ввода команд. Модульная колодка позволяет программировать микросхемы в различных корпусах. В автономном режиме программатор PRO MATE II может проверять микроконтроллер и устанавливать биты защиты всех микроконтроллеров PICmicro. Программатор PRO MATE II поддерживает программирование микросхем последовательной EEPROM памяти и микросхем KeeLoq.

Предусмотрен дополнительный модуль для внутрисхемного программирования микроконтроллеров в производственных условиях (см. техническую документацию на программатор PRO MATE II).

32.6.2 Недорогой программатор PICSTART Plus

Недорогой программатор PICSTART Plus предназначен для начала работы с микроконтроллерами PICmicro, подключается к PC совместимому компьютеру через COM (RS-232) порт и работает под управлением интегрированной среды проектирования MPLAB-IDE. Не рекомендуется использовать PICSTART Plus для выпуска продукции, т.к. он не проверяет программирование микроконтроллера при минимальном и максимальном напряжении питания.

PICSTART поддерживает все микроконтроллеры PICmicro в корпусах до 40 выводов. Микроконтроллеры с большим числом выводов (PIC16C92X, PIC17C76X) поддерживаются при использовании адаптеров.

32.7 Дополнительные инструментальные средства

Microchip пытается обеспечить заказчиков готовыми решениями задач. Некоторые изделия могут не относиться к классическим инструментальным средствам проектирования: языки высокого уровня; технология Fuzzy Logic; визуальные средства проектирования. Подобные инструментальные средства рассматриваются как дополнительные и могут быть приобретены в компании Microchip или у другого продавца. Полный перечень инструментальных средств других производителей смотрите в документе "Third Party Guide".

32.7.1 Среда проектирования fuzzyTECH-MP

Среда проектирования нечеткой логики *fuzzy*TECH-MP разработана в двух версиях: недорогая версия *fuzzy*TECH-MP предназначена для ознакомления проектировщиков с возможностями использования нечеткой логики в разрабатываемых проектах; полнофункциональная версия.

В комплект обоих версий входит демонстрационная плата *fuzzy*LAB™ с рабочими примерами.

32.7.2 Генератор кода MP-DriveWay

MP-DriveWay - удобный в работе генератор кода инициализации микроконтроллера. Вы указываете конфигурацию микроконтроллера PIC16/17 и даете команду сгенерировать текст инициализации на языке C. Выходной файл полностью совместим с MPLAB-C Microchip с возможностью простой интеграции в ваш текст программы.

32.7.3 Средства проектирования других производителей

Microchip поддерживает разработчиков инструментальных средств для выпускаемых микросхем. Полный список выпускаемых средств проектирования смотрите в документе "Third Party Guide". По каждому производителю представлены следующие данные:

- Название компании;
- Выпускаемая продукция;
- Контактная информация;
- Пояснения.

Более 100 компаний выпускаю больше 200 изделий в состав которых входит: эмуляторы, программаторы, языки программирования и др.

32.8 Демонстрационные платы

Демонстрационные платы предоставляют возможность быстро ознакомиться с особенностями работы конкретного микроконтроллера и начать разработку устройства. Демонстрационные программы могут быть изменены в соответствии с вашими требованиями.

32.8.1 Демонстрационная плата PICDEM-1

Демонстрационная плата PICDEM-1 предназначена для микроконтроллеров PIC16C5X (PIC26C54, PIC16C58A), PIC16C61, PIC16C62X, PIC16C71, PIC16C8X, PIC17C42, PIC17C43 и PIC17C44. В комплект поставки входят необходимые аппаратные модули, программное обеспечение и демонстрационные программы. Записать демонстрационные программы в микроконтроллер можно с помощью программатора PRO MATE II или PICSTART. Пользователь может подключить к демонстрационной плате эмулятор MPLAB-ICE и выполнять отладку программы. На демонстрационной плате имеется полигон для установки дополнительных элементов пользователя. В состав демонстрационной платы входит: драйвер интерфейса RS-232, потенциометр для моделирования аналогового входа, выключатели и восемь светодиодов подключенных к PORTB.

32.8.2 Демонстрационная плата PICDEM-2 для PIC16CXXX

Демонстрационная плата PICDEM-2 предназначена для микроконтроллеров PIC16C62, PIC16C64, PIC16C65, PIC16C73 и PIC16C74. В комплект поставки входят необходимые аппаратные модули, программное обеспечение и демонстрационные программы. Записать демонстрационные программы в микроконтроллер можно с помощью программатора PRO MATE II или PICSTART. Пользователь может подключить к демонстрационной плате эмулятор MPLAB-ICE и выполнять отладку программы. На демонстрационной плате имеется полигон для установки дополнительных элементов пользователя. В состав демонстрационной платы входит: драйвер интерфейса RS-232, потенциометр для моделирования аналогового входа, последовательная EEPROM для демонстрации работы шины I^2C, выводы для подключения ЖКИ и дополнительной клавиатуры.

32.8.3 Демонстрационная плата PICDEM-3 для PIC16CXXX

Демонстрационная плата PICDEM-3 предназначена для микроконтроллеров PIC16C923 и PIC16C924 выполненных в 44-выводном PLCC корпусе с интегрированным ЖКИ модулем. В комплект поставки входят необходимые аппаратные модули, программное обеспечение и демонстрационные программы. Записать демонстрационные программы в микроконтроллер можно с помощью программатора PRO MATE II или PICSTART. Пользователь может подключить к демонстрационной плате эмулятор MPLAB-ICE и выполнять отладку программы. На демонстрационной плате имеется полигон для установки дополнительных элементов пользователя. В состав демонстрационной платы входит: драйвер интерфейса RS-232, выключатели; потенциометр для моделирования аналогового входа; термистор; выводы для подключения ЖКИ и дополнительной клавиатуры; 12-разрядный ЖКИ для отображения времени, даты и температуры; дополнительный интерфейс RS-232; программное обеспечение работающее под управлением операционной системы Windows 3.x для передачи данных на PC совместимый компьютер.

32.8.4 Демонстрационная плата PICDEM-14A для PIC14C000

Демонстрационная плата PICDEM-14A предназначена для микроконтроллера PIC14C000. На демонстрационной плате выполняется измерение напряжения и температуры с интегрированного датчика. Полученные данные калибруются относительно внутреннего источника опорного напряжения. Измеренное напряжение и температура передаются по интерфейсу RS-232. Просмотреть данные можно с помощью простой программы терминала. В состав демонстрационной платы входит: драйвер интерфейса RS-232, потенциометр для моделирования аналогового входа, последовательная EEPROM для демонстрации работы шины I^2C, выводы для подключения ЖКИ и дополнительной клавиатуры.

32.9 Средства проектирования для другой продукции Microchip

32.9.1 SEEVAL (с функциями программатора)

Комплект SEEVAL SEEPROM поддерживает весь спектр 2-х/3-х проводных последовательных микросхем EEPROM фирмы Microchip. Комплект позволяет выполнять чтение, стирание и запись любой микросхемы последовательного EEPROM фирмы Microchip. Система позволяет сделать анализ обмена данных, число циклов и надежность записи. Полный комплект SEEVAL позволяет уменьшить время проектирование устройства.

32.9.2 KeeLoq (с функциями программатора)

Оценочная система KeeLoq предназначена для микросхем HCS фирмы Microchip. В состав комплекта входит: ЖКИ дисплей для отображения изменяющихся кодов, декодер, интерфейс программирования.

32.10 Дополнительная литература

Дополнительная литература и примеры применения, связанные с этим разделом документации. Примеры применения не могут использоваться для всех микроконтроллеров среднего семейства (PIC16CXXX). Как правило, примеры применения написаны для конкретной группы микроконтроллеров, но принципы примеров могут использоваться, сделав незначительные изменения (с учетом существующих ограничений).

Документы, связанные со средствами проектирования для микроконтроллеров PICmicro MCU:

Документ	Номер
Air Flow Control using Fuzzy Logic Управление потоками воздуха на основе Fuzzy Logic	AN600

Раздел 33. Приложения

Содержание

Приложение A. Введение в I^2C

В этой главе представлено краткое описание внутрисхемной шины передачи данных I^2C с рассмотрением вопросов адресации и работы модуля SSP.

I^2C - двухпроводный последовательный интерфейс, разработанный корпорацией Philips. В Первоначальном техническом требовании к интерфейсу максимальная скорость передачи данных составляла 100 Кбит/с. Однако позже появились стандартные более скоростные режимы работы шины I^2C (400Кбит/с и 1Мбит/с). К одной шине I^2C могут быть подключены устройства с различными скоростями доступа, если скорость передачи данных будет удовлетворять требованиям самого низкоскоростного устройства.

Протокол передачи данных по шине I^2C разработан таким образом, чтобы гарантировать надежный качественный прием/передачу данных. При передаче данных одно устройство является "Ведущим", которое инициирует передачу данных и формирует сигналы синхронизации. Другое устройство "Ведомое", которое может начать передачу данных только по команде ведущего шины. Модуль SSP микроконтроллеров PIC16CXXX полностью поддерживает режим ведомого I^2C, за исключением поддержки адреса общего вызова (режим ведущего реализуется программно). Модуль MSSP аппаратно поддерживает режим ведущего/ведомого I^2C, адрес общего вызова и скорость обмена данными до 1Мбит/с. Скорость передачи данных 1Мбит/с используют некоторые микросхемы последовательной EEPROM памяти. В таблице A-1 представлены основные термины, связанные с шиной I^2C.

Каждое устройство на шине I^2C имеет уникальный адрес. Когда ведущий инициирует передачу данных, то сначала передается адрес устройства, к которому выполняется обращение. Остальные устройства проверяют переданный ведущим адрес. В состав байта адреса устройства входит бит направления передачи данных (выполняется чтение из ведомого или запись). Ведомый и ведущий шины всегда находятся в противоположном режиме работы, что можно представить в виде двух состояний:

- Ведущий передатчик - ведомый приемник.
- Ведомый передатчик - ведущий приемник.

В обоих случаях ведущий формирует тактовый сигнал.

Вывод тактового сигнала (SCL) и данных (SDA) должны иметь выход с открытым коллектором, чтобы выполнять требования "монтажного И" на шине. Для формирования высокого уровня сигнала на линиях к ним подключаются подтягивающие резисторы. Число устройств, которые могут быть подключены к шине I^2C, ограничивается только максимальной емкостью шины (400пФ) и способностью адресации этих устройств.

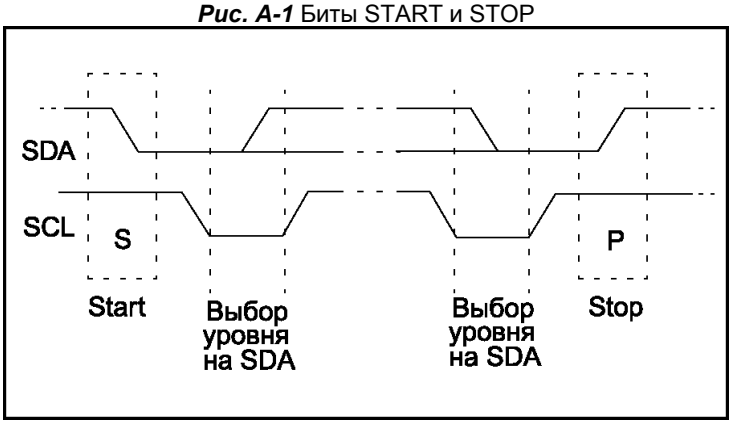
Таблица A-1 Основные термины, связанные с шиной I^2C

Термин	Описание
Передатчик	Устройство, передающее данные по шине I^2C.
Приемник	Устройство, принимающее данные с шины I^2C.
Ведущий	Устройство, инициирующее передачу данных и формирующее тактовый сигнал.
Ведомый	Устройство, к которому обращается ведущий.
Конкуренция	Более чем один ведущий на шине. Несколько ведущих могут пытаться передать данные без разрушения текущего сообщения.
Арбитраж	Процедура, гарантирующая, что только один ведущий управляет шиной.
Синхронизация	Процедура синхронизации тактовых сигналов от двух или более устройств.

A.1 Инициализация и завершение передачи данных

В то время, когда передача данных на шине I^2C отсутствует, сигнал синхронизации (SCL) и данных (SDA) имеют высокий логический уровень за счет подтягивающих резисторов. Биты START и STOP формируются ведущим для определения начала и окончания передачи данных соответственно. Бит START формируется переходом сигнала SDA из высокого уровня в низкий при высоком уровне сигнала SCL. Бит STOP формируется переходом SDA из низкого уровня в высокий при высоком уровне SCL. На рисунке A-1 показано формирование битов START и STOP. Ведущий шины формирует биты START и STOP для указания начала и завершения передачи данных. При передаче данных сигнал SDA может изменяться только, когда SCL имеет низкий логический уровень.

Рис. A-1 Биты START и STOP

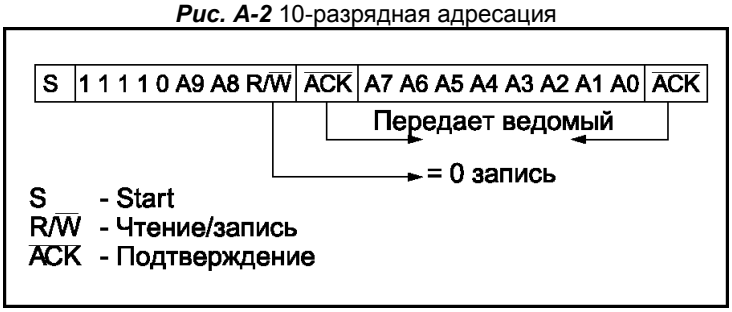


A.2 Адресация устройств на шине I^2C

Для адресации устройств используется два формата адреса: простой 7-разрядный формат с битом чтения/записи R/W (см. рис. A-2); 10-разрядный формат, передается два байта. В первом байте передается: пять битов, определяющих, что это 10-разрядный адрес; два старших бита адреса; бит записи/чтения. Во втором байте передается 8 младших бит адреса (см. рис. A-3).

Рис. A-2 7-разрядная адресация

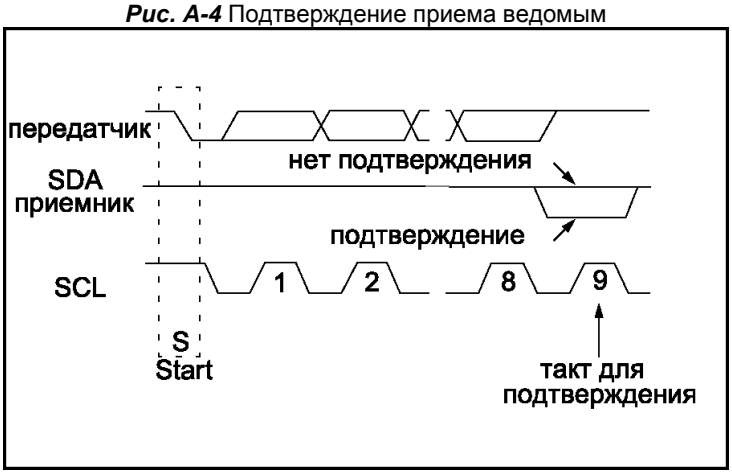
Рис. A-2 10-разрядная адресация



A.3 Подтверждение приема

При передаче данных после каждого переданного байта приемник должен подтвердить получения байта сигналом ACK. (см. рис. A-4). Если ведомый не подтверждает получение байта адреса или данных, ведущий должен прервать передачу сформировав сигнал STOP (ведомый должен отпустить SDA для формирования STOP ведущим).

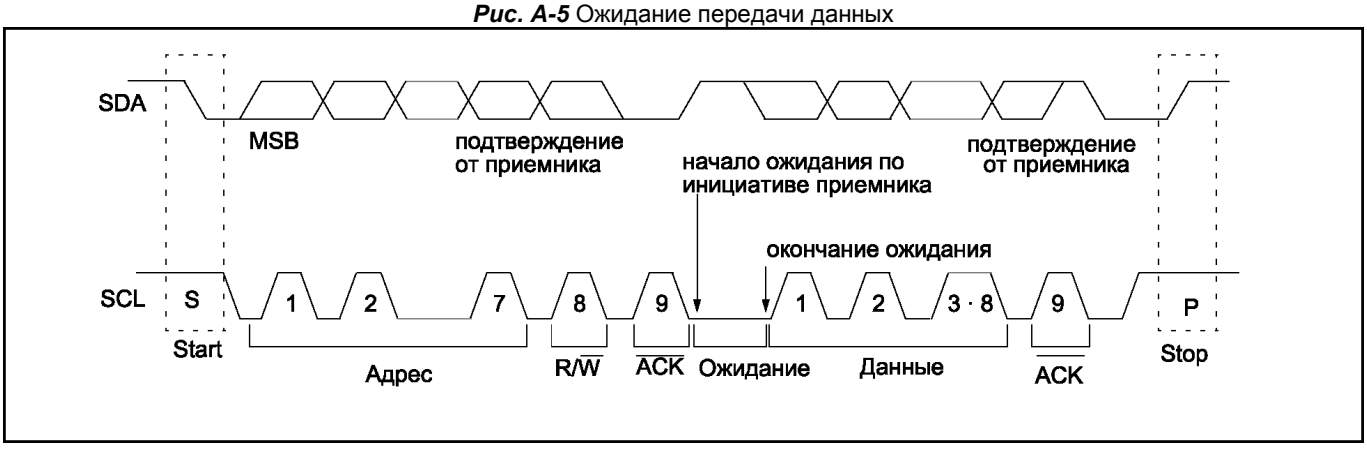
Рис. A-4 Подтверждение приема ведомым



Когда ведущий шины принимает данные, то на каждый принимаемый байт формируется бит подтверждения, если принятый байт не последний. Для сообщения ведомому о том, что ведущий прекращает принимать данные по приему последнего байта -ACK не формируется. Ведомый отпускает SDA, чтобы ведущий смог передать бит STOP. Ведущий может формировать бит STOP на месте бита подтверждения.

Если ведомому необходимо задержать передачу данных, то он может удерживать SCL в низком логическом уровне. Передача данных продолжится, когда ведомый отпустит SCL. Это позволяет ведомому подготовить новые данные для передачи. Методика задержки передачи данных может использоваться и при передаче отдельных битов (см. рис. A-5).

Рис. A-5 Ожидание передачи данных



На рисунках А-6 и А-7 показаны последовательности приема и передачи данных ведущим шины.

Рис. А-6 Последовательность передачи данных ведущим

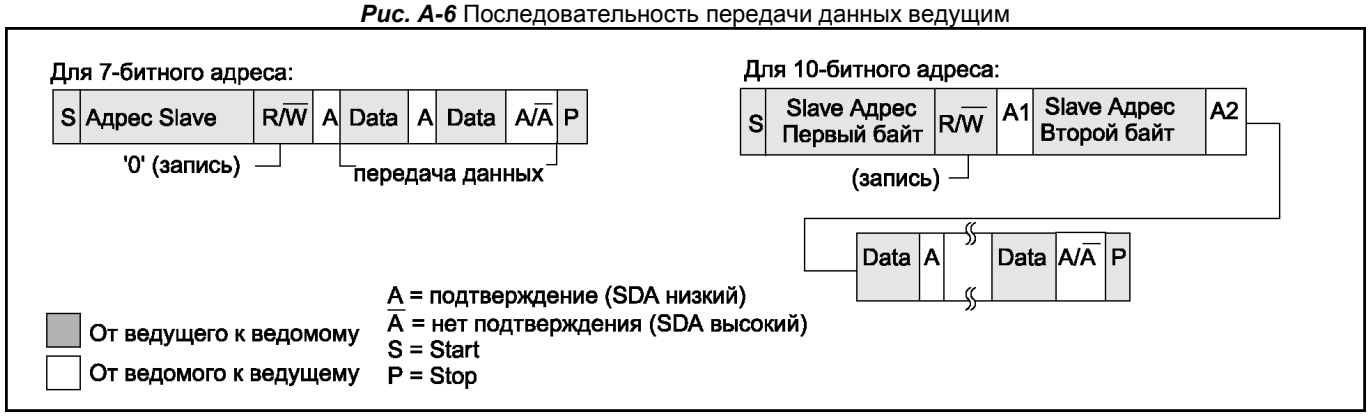
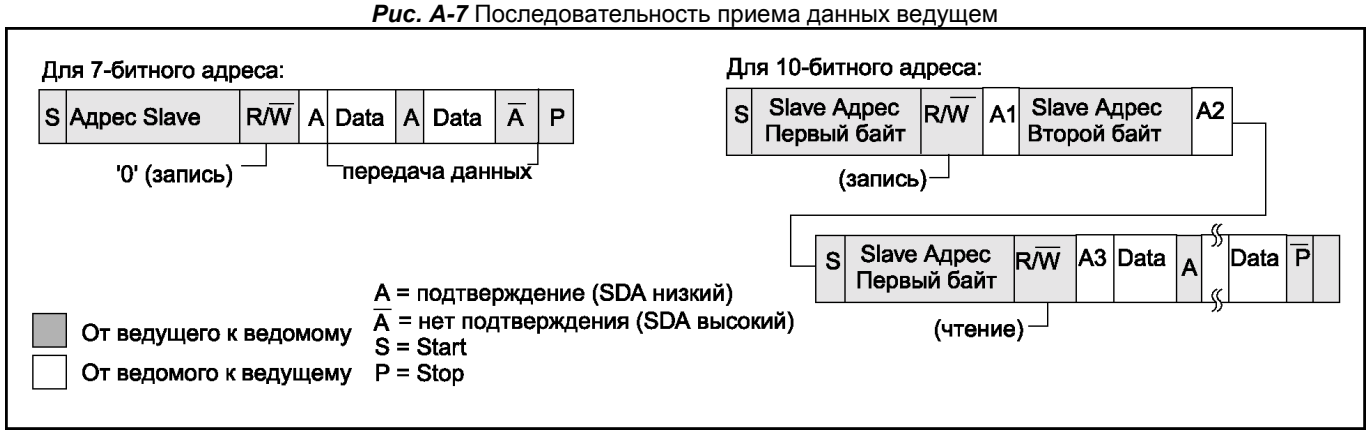
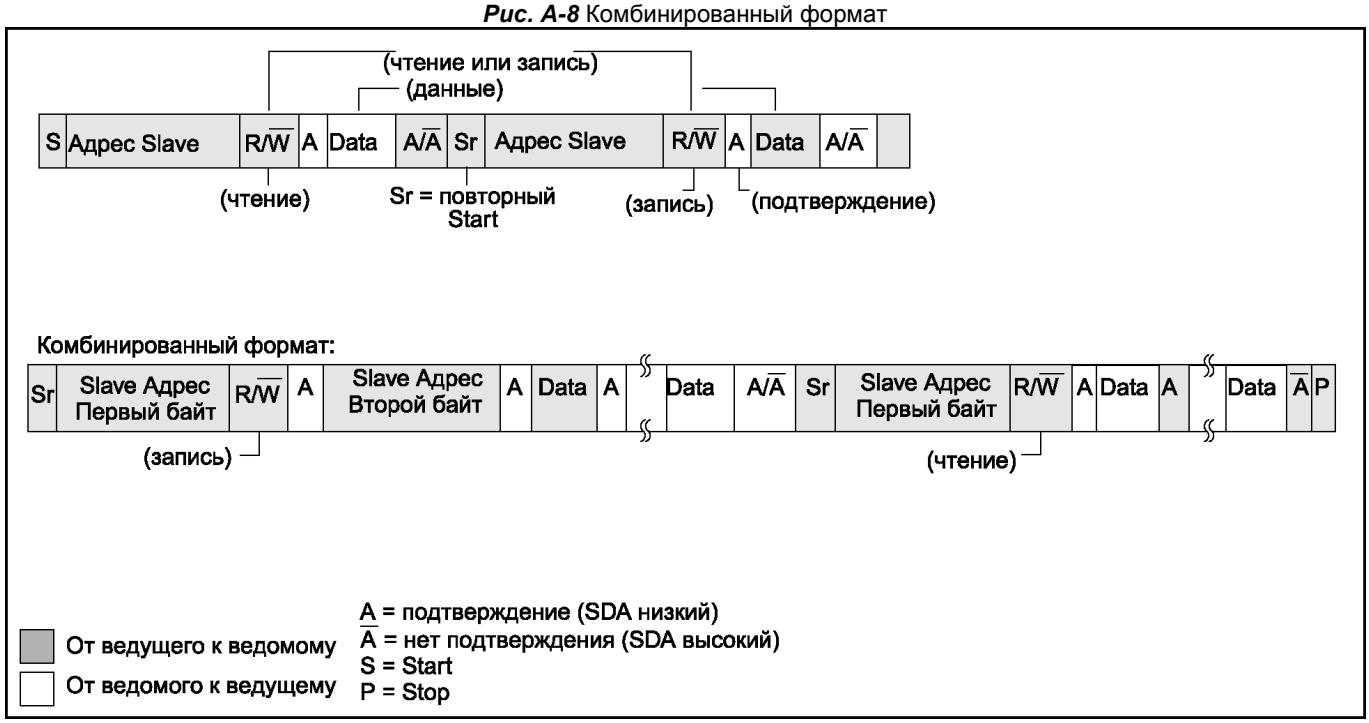


Рис. А-7 Последовательность приема данных ведущем



Когда ведущему шины необходимо продолжить обмен данными (при формировании бита STOP управление шиной теряется) может быть передан бит повторный START. Условие повторный START идентично условию START (на SDA формируется переход с высокого логического уровня в низкий при высоком уровне сигнала на SCL), но формируется после передачи бита подтверждения. Это позволяет ведущему продолжить обмен с текущим устройством или адресовать новое (см. рис. А-8).

Рис. А-8 Комбинированный формат



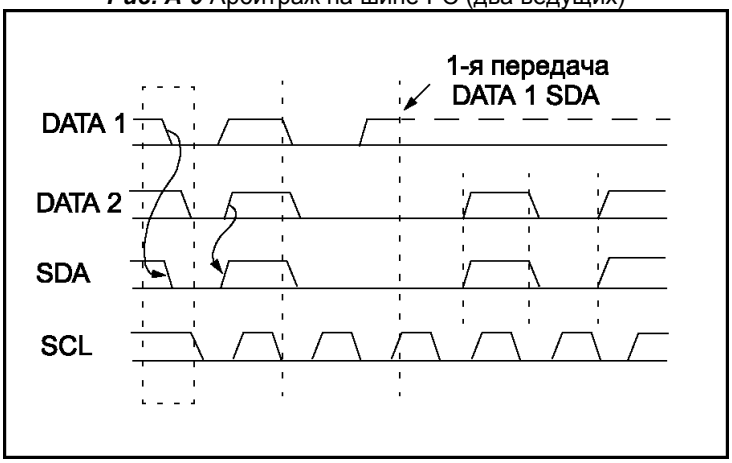
A.4 Режим конкуренции

Протокол передачи данных I^2C позволяет иметь более одного ведущего на шине. Для разрешения конфликтов на шине при инициализации передачи используются функции арбитража и синхронизации.

A.4.1 Арбитраж

Арбитраж выполняется на линии SDA при высоком уровне сигнала на SCL. Устройство, которое формирует на линии SDA высокий уровень, когда другое устройство передает низкий, теряет право быть ведущим и должно перейти в режим ведомого. Ведущий, потерявший инициативу на шине I^2C, может формировать тактовые импульсы до конца байта, в котором потерял управление шиной. Когда несколько ведущих адресуют одно и тоже устройство, то арбитраж выполняется при передаче данных.

Рис. A-9 Арбитраж на шине I^2C (два ведущих)



Ведущий, потерявший арбитраж, должен немедленно перейти в режим ведомого, поскольку он может быть адресован текущим ведущим.

Арбитраж не допускается между:

- Битами повторный START;
- Битом STOP и битом данных;
- Битами повторный START и STOP.

Ведущий шины должен гарантировать отсутствие указанных условий.

A.4.2 Синхронизация

Синхронизация тактового сигнала выполняется, когда устройства начинают арбитраж. Синхронизация реализуется за счет включение линии SCL по схеме "монтажное И". Переход сигнала на SCL с высокого логического уровня в низкий заставляет устройства, выполняющие арбитраж, начать отсчет длительности низкого логического уровня. После того, как тактовый сигнал устройства перешел в низкий уровень, оно будет удерживать этот уровень на SCL до тех пор, пока тактовый сигнал не перейдет в высокий уровень, но на SCL может бить по прежнему низкий уровень, если другое устройство формирует низкий логический уровень. Низкий уровень на SCL удерживается устройством с минимальной частотой тактового сигнала передачи данных. Устройства с меньшей длительностью низкого уровня на SCL переходят в состояние ожидания, пока на SCL не появится высокий логический уровень сигнала. Затем все устройства начинают отсчет длительность высокого уровня сигнала. Устройство, с минимальной длительность высокого уровня сигнала, первым переведет SCL в низкий уровень (см. рис. A-10).

Рис. A-10 Синхронизация тактового сигнала на шине I^2C

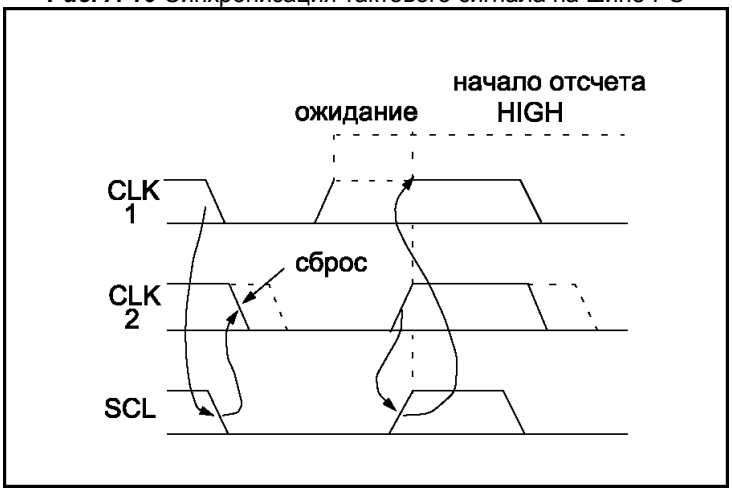


Рис. А-11 Временная диаграмма формирования битов START/STOP на шине I^2C

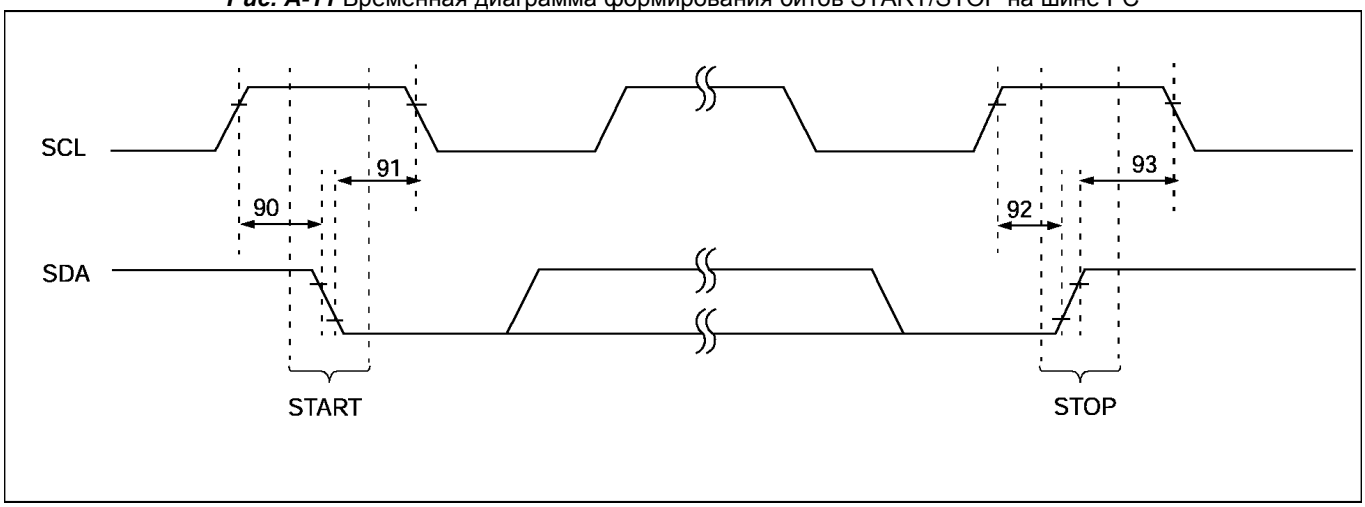


Таблица А-2 Параметры формирования битов START/STOP на шине I^2C

№ пар.	Обоз.	Описание		Мин.	Тип.**	Макс.	Ед.	Примечание
90	Tsu:sta	Установка условия START	Режим 100 кГц	4700	-	-	нс	Только при формировании бита повторный START
			Режим 400 кГц	600	-	-		
91	Thd:sta	Удержание условия START	Режим 100 кГц	4000	-	-	нс	После этого форм. первый импульс тактового сигнала
			Режим 400 кГц	600	-	-		
92	Tsu:sto	Установка условия STOP	Режим 100 кГц	4700	-	-	нс	
			Режим 400 кГц	600	-			
93	Thd:sto	Удержание условия STOP	Режим 100 кГц	4000	-	-	нс	
			Режим 400 кГц	600	-	-		

Рис. А-12 Временная диаграмма формирования бита данных на шине I^2C

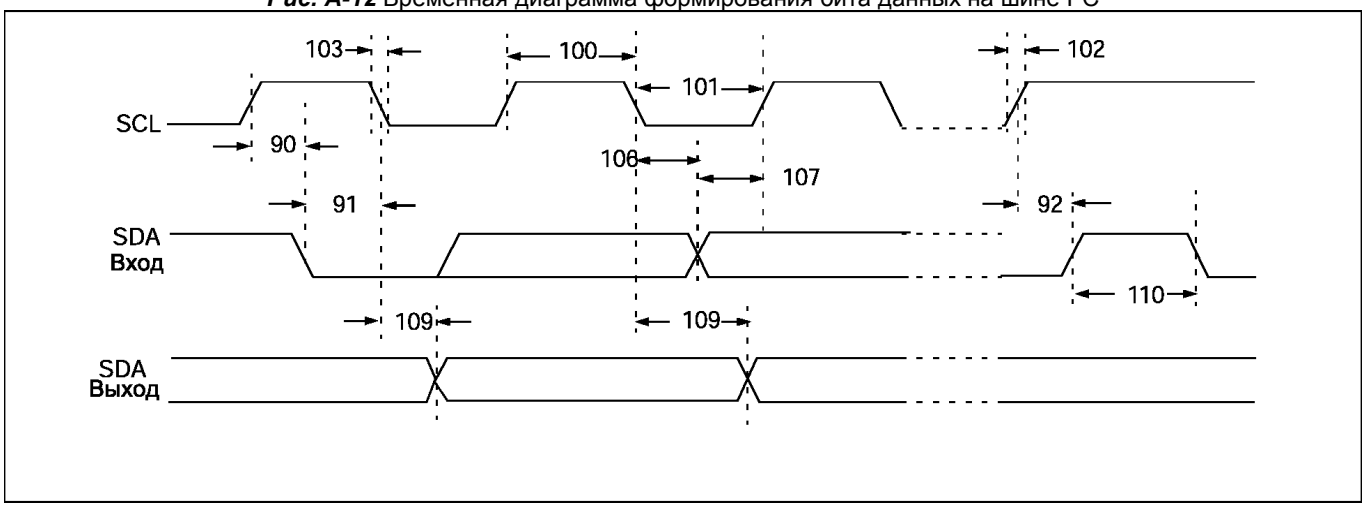


Таблица А-3 Параметры формирования бита данных на шине I^2C

№ пар.	Обоз.	Описание		Мин.	Макс.	Ед.	Примечание
100	Thigh	Длительность высокого уровня тактового сигнала	Режим 100 кГц	4.0	-	мкс	Мин. F_{OSC} 1.5МГц
			Режим 400 кГц	0.6	-	мкс	Мин. F_{OSC} 10МГц
			Модуль SSP	$1.5T_{CY}$	-		
101	Tlow	Длительность низкого уровня тактового сигнала	Режим 100 кГц	4.7	-	мкс	Мин. F_{OSC} 1.5МГц
			Режим 400 кГц	1.3	-	мкс	Мин. F_{OSC} 10МГц
			Модуль SSP	$1.5T_{CY}$	-		
102	Tr	Долит. переднего фронта на SDA и SCL	Режим 100 кГц	-	1000	нс	
			Режим 400 кГц	20 + 0.1 Cb	300	нс	10пФ ≤ Cb ≤ 400пФ
103	Tf	Долит. заднего фронта на SDA и SCL	Режим 100 кГц	-	300	нс	
			Режим 400 кГц	20 + 0.1 Cb	300	нс	10пФ ≤ Cb ≤ 400пФ
90	Tsu:sta	Установка условия START	Режим 100 кГц	4.7	-	мкс	Только при формировании бита повторный START
			Режим 400 кГц	0.6	-	мкс	
91	Thd:sta	Удержание условия START	Режим 100 кГц	4.0	-	мкс	После этого форм. первый импульс тактового сигнала
			Режим 400 кГц	0.6	-	мкс	
106	Thd:dat	Удержание данных на входе	Режим 100 кГц	0	-	нс	
			Режим 400 кГц	0	0.9	мкс	
107	Tsu:dat	Установка данных на входе	Режим 100 кГц	250	-	нс	Примечание 2
			Режим 400 кГц	100	-	нс	
92	Tsu:sto	Установка условия STOP	Режим 100 кГц	4.7	-	мкс	
			Режим 400 кГц	0.6	-	мкс	
109	Taa	Достоверность сигнала на выходе	Режим 100 кГц	-	3500	нс	Примечание 1
			Режим 400 кГц	-	-	нс	
110	Tbuf	Время не занятости шины	Режим 100 кГц	4.7	-	мкс	Задержка перед новой передачей
			Режим 400 кГц	1.3	-	мкс	
D102	Cb	Емкостная нагрузка линии		-	400	пФ	

Примечания:

1. Необходимо выдерживать эту минимальную задержку относительно заднего фронта SCL, чтобы избежать ложное формирование битов START и STOP.

2. Устройства с высокоскоростным режимом обмена (400кГц) могут использоваться в стандартном режиме (100кГц), но требование Tsu:dat ≥ 250нс необходимо выполнять. Это условие автоматически будет выполняться, если не возникает удержания линии SCL в низком логическом уровне. Если возникает удержание линии SCL в низком логическом уровне, то необходимо сформировать бит данных на SDA Tr.max + Tsu:dat = 1000 + 250 = 1250 нс (согласно спецификации I^2C) прежде, чем SCL будет "отпущена".

Приложение B. Рекомендованные производители ЖКИ стекол

AEG-MIS
3340 Peachtree Rd. NE Suite 500
Atlanta, GA 30326
TEL: 404-239-0277
FAX: 404-239-0383

All Shore INDS Inc.
1 Edgewater Plaza
Staten Island, NY 10305
TEL: 718-720-0018
FAX: 718-720-0225

Crystaloid
5282 Hudson Drive
Hudson, OH 44236-3769
TEL: 216-655-2429
FAX: 216-655-2176

DCI Inc.
14812 W. 117th St.
Olathe, KS 66062-9304
TEL: 913-782-5672
FAX: 913-782-5766

Excel Technology International Corporation
Unit 5, Bldg. 4, Stryker Lane
Belle Mead, NJ 08502
TEL: 908-874-4747
FAX: 908-874-3278

F-P Electronics/Mark IV Industries
6030 Ambler Drive
Mississauga, ON Canada L4W 2PI
TEL: 905-624-3020
FAX: 905-238-3141

Hunter Components
24800 Chagrin Blvd, Suite 101
Cleveland, OH 44122
TEL: 216-831-1464
FAX: 216-831-1463

Interstate Electronics Corp.
1001 E. Bull Rd.
Anaheim, CA 92805
TEL: 800-854-6979
FAX: 714-758-4111

Kent Display Systems
343 Portage Blvd.
Kent, OH 44240
TEL: 330-673-8784

LCD Planar Optics Corporation
2100-2 Artic Ave.
Bohemia, NY 11716
TEL: 516-567-4100
FAX: 516-567-8516

LXD Inc.
7650 First Place
Oakwood Village, OH 44146
TEL: 216-786-8700
FAX: 216-786-8711

Nippon Sheet Glass
Tomen America Inc.
1285 Avenue of the Americas
New York, NY 10019
TEL: 212-397-4600
FAX: 212-397-3351

OPTREX America
44160 Plymouth Oaks Blvd.
Plymouth, MI 48170
TEL: 313-416-8500
FAX: 313-416-8520

Phillips Components
LCD Business Unit
1273 Lyons Road, Bldg G
Dayton, OH 45459
TEL: 573-436-9500
FAX: 573-436-2230

Satori Electric
23717 Hawthorne Blvd. 3rd Floor
Torrance, CA 90505
TEL: 310-214-1791
FAX: 310-214-1721

Seiko Instruments USA Inc.
Electronic Components Division
2990 West Lomita Blvd.
Torrance, CA 90505
TEL: 213-517-7770
213-517-8113
FAX: 213-517-7792

Standish International
European Technical Center
Am Baumstuck II
65520 Bad Camberg/Erbach
Germany
TEL: 011 49 6434 3324
FAX: 011 49 6434 377238

Standish LCD
W7514 Highway V
Lake Mills, WI 53551
TEL: 414-648-1000
FAX: 414-648-1001

Truly Semiconductors Ltd. (USA)
2620 Concord Ave.
Suite 106
Alhambra, CA 91803
TEL: 818-284-3033
FAX: 818-284-6026

Truly Semiconductor Ltd.
2/F, Chung Shun Knitting Center
1-3 Wing Yip Street,
Kwai Chung, N.T., Hong Kong
TEL: 852 2487 9803
FAX: 852 2480 0126

Varitronix Limited Inc.
3250 Wilshire Blvd. Suite 1901
Los Angeles, CA 90010
TEL: 213-738-8700
FAX: 213-738-5340

Varitronix Limited Inc.
4/F, Liven House
61-63 King Yip Street
Kwun Tong, Kowloon
Hong Kong
TEL: 852 2389 4317
FAX: 852 2343 9555

Varitronix (France) S.A.R.L.
13/15 Chemin De Chilly
91160 Champlain
France
TEL:(33) 1 69 09 7070
FAX:(33) 1 69 09 0535

Varitronix Italia, S.R.L.
Via Bruno Buozzi 90
20099 Sesto San Giovanni
Milano, Italy
TEL:(39) 2 2622 2744
FAX:(39) 2 2622 2745

Varitronix (UK) Limited
Display House, 3 Milbanke Court
Milbanke Way, Bracknell
Berkshire RG12 1BR
United Kingdom
TEL:(44) 1344 30377
FAX(44) 1344 300099

Varitronix (Canada) Limited
18 Crown Steel Drive, Suite 101
Markham, Ontario
Canada L3R 9X8
TEL:(905) 415-0023
FAX:(905) 415-0094

Vikay America Inc.
195 W. Main St.
Avon, CT 06001-3685
TEL: 860-678-7600
FAX: 860-678-7625

Приложение C. Усовершенствование микроконтроллеров

Поскольку постоянно выполняется усовершенствование микроконтроллеров, некоторые периферийные модули и особенности были изменены, в частности это касается:

1. Карты памяти данных;
2. Модуля SSP;
3. Модуля АЦП;
4. Добавлен сброс по снижению напряжения питания BOR;
5. Фильтр на входе -MCLR;
6. Модуль USART;
7. Генератор тактового сигнала.

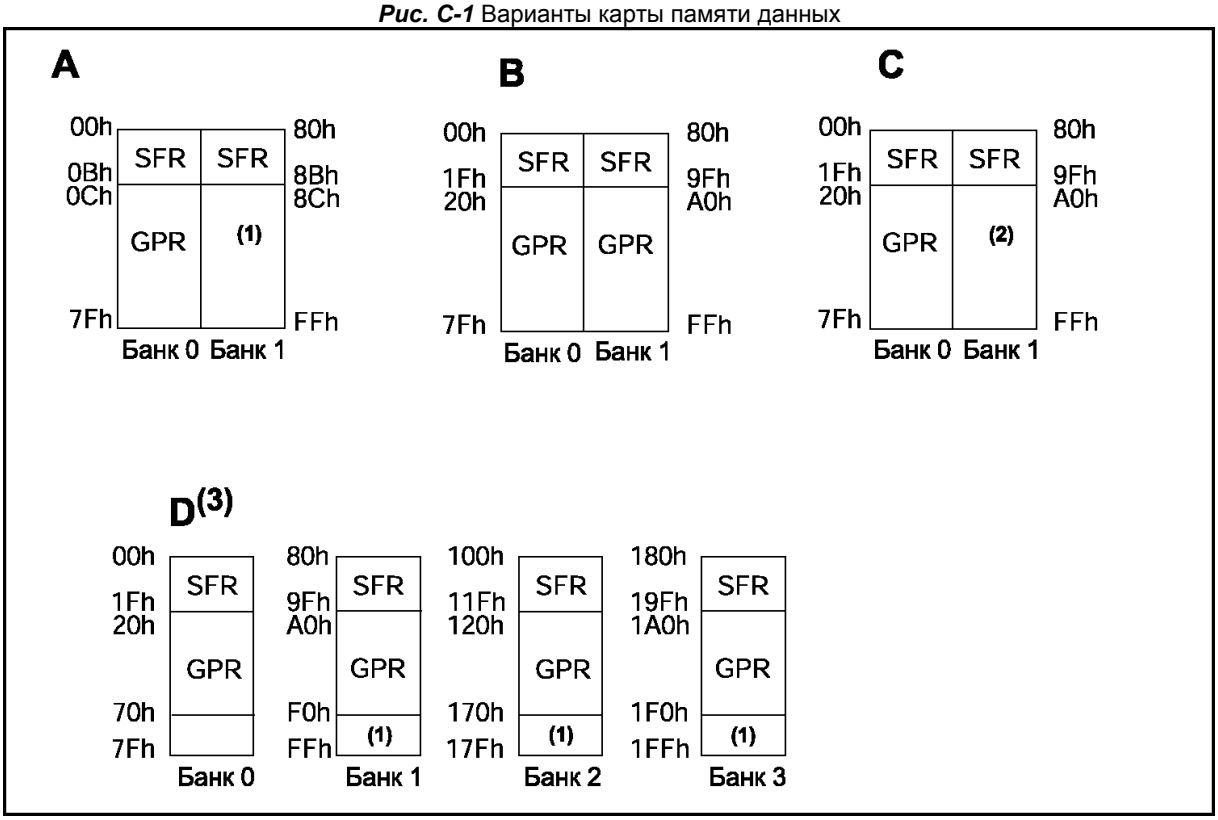
В следующих главах раздела будет подробно рассмотрено каждое из изменений.

C.1 Карта памяти данных

На карте памяти данных показано расположение регистров специального назначения (SFR) и регистров общего назначения (GPR). Регистры SFR предназначены для управления ядром микроконтроллера и периферийными модулями, а регистры GPR - это универсальное ОЗУ пользователя.

На рисунке C-1 показаны различные карты памяти данных, которые были реализованы в микроконтроллерах среднего семейства. Карта памяти **A** была реализована в первых микроконтроллерах среднего семейства. Это были 18/20 - выводные микроконтроллеры с ограниченными периферийными функциями. Когда развитие электроники стало диктовать новые требования, были выпушены микроконтроллеры с большим числом портов ввода/вывода и расширенными периферийными функциями (карта памяти **B**). Карта памяти **C** фактически является подмножеством карт памяти **B**, но позволяет упростить сохранение контекста при обработке прерываний потому, что нет регистров GPR в банке 1. Для уменьшения программы сохранения/восстановления контекста при обработке прерываний была реализована карта памяти **D**. Подобная организация памяти (**D**) будет использоваться во всех новых микроконтроллерах. Смотрите раздел "Организация памяти" для уточнения деталей использования памяти данных.

Рис. C-1 Варианты карты памяти данных



Примечания:

1. Отображается на банк 0.
2. Не реализовано, читается как '0'.
3. В некоторых микроконтроллерах регистры GPR могут размещаться в области регистров SFR.

C.2 Модуль SSP

Модуль SSP имеет два режима работы:

- SPI (Последовательный периферийный интерфейс);
- I2C (Inter-Integrated Circuit).

Существует три варианта модулей SSP, которые применяются в микроконтроллерах Microchip. Первый SSP модуль (теперь называется основной модуль BSSP) поддерживает два из четырех режимов SPI и режим ведомого I^2C. Второй модуль SSP (обозначаемый - модуль SSP) поддерживает все четыре режима SPI и режим ведомого I^2C. Третий модуль SSP (обозначаемый - модуль MSSP) поддерживает четыре режима SPI и ведомый/ведущий режим I^2C. В таблице C-1 указано какой модуль SSP реализован в конкретном микроконтроллере. В новых микроконтроллерах будет реализовываться только модули SSP или MSSP. Только некоторые микроконтроллеры будут содержать модуль MSSP, поскольку существует прямая зависимость стоимости микроконтроллера от размера кристалла. Если в вашем приложении необходима аппаратная поддержка режима ведущего I^2C, то используйте микроконтроллеры PICmicro старшего семейства.

Таблица C-1 Микроконтроллеры с модулем SSP

Микроконтроллер	**Вариант модуля SSP**		
	SSP	**BSSP**	**MSSP**
PIC16C62	-	Да	-
PIC16C62A	-	Да	-
PIC16CR62	-	Да	-
PIC16C63	-	Да	-
PIC16CR63	-	Да	-
PIC16C64	-	Да	-
PIC16C64A	-	Да	-
PIC16C64A	-	Да	-
PIC16CR64	-	Да	-
PIC16C65	-	Да	-
PIC16C65A	-	Да	-
PIC16CR65	-	Да	-
PIC16C66	Да	-	-
PIC16C67	Да	-	-
PIC16C72	-	Да	-
PIC16CR72	Да	-	-
PIC16C73	-	Да	-
PIC16C73A	-	Да	-
PIC16C74	-	Да	-
PIC16C74A	-	Да	-
PIC16C76	Да	-	-
PIC16C77	Да	-	-
PIC16C923	Да	-	-
PIC16C924	Да	-	-
Новые микроконтроллеры с модулем SSP	Смотрите техническую документацию на микроконтроллер		

C.3 Модуль АЦП

Существует несколько вариантов модулей АЦП, которые применяются в микроконтроллерах Microchip. Первый модуль АЦП (основной модуль АЦП) - 8-разрядный модуль АЦП с четырьмя входными каналами. Второй модуль АЦП (8 - разрядный модуль АЦП) - 8-разрядный модуль АЦП, поддерживающий до 8 входных каналов. Третий модуль АЦП (10 - разрядный модуль АЦП) - 10 - разрядный модуль АЦП, поддерживающий до 16 входных каналов. В таблице C-2 указана какой модуль АЦП реализован в конкретном микроконтроллере. В новых микроконтроллерах будет использоваться 8 - разрядный или 10 - разрядный модуль АЦП (основной 8 - разрядный модуль АЦП применяться не будет). Если в вашем приложении необходимо 10 - разрядное АЦП, то используйте микроконтроллеры PICmicro старшего семейства.

Таблица C-2 Микроконтроллеры с модулем АЦП

Микроконтроллер	8 - разрядное АЦП	Основное 8 - разрядное АЦП	10 - разрядное АЦП	Интегрирующее АЦП
PIC16C710	-	Да	-	-
PIC16C71	-	Да	-	-
PIC16C711	-	Да	-	-
PIC16C715	-	Да	-	-
PIC16C72	Да	-	-	-
PIC16CR72	Да	-	-	-
PIC16C73	Да	-	-	-
PIC16C73A	Да	-	-	-
PIC16C74	Да	-	-	-
PIC16C74A	Да	-	-	-
PIC16C76	Да	-	-	-
PIC16C77	Да	-	-	-
PIC16C924	Да	-	-	-
PIC14C000	-	-	-	Да
Новые микроконтроллеры с модулем АЦП	Смотрите техническую документацию на микроконтроллер			

C.4 Сброс по снижению напряжения питания

Внутренняя схема сброса по снижению напряжения питания (BOR) была добавлена к особенностям микроконтроллеров. Схема BOR будет присутствовать во всех новых микроконтроллерах. Исключения будут составлять микроконтроллеры, в которых основное напряжение питания ниже уровня схемы BOR (микроконтроллеры для носимой аппаратуры с питанием от батареек). В таблице C-3 представлены микроконтроллеры, в которых была добавлена схема BOR.

Таблица C-3 микроконтроллеры, в которые была добавлена схема BOR

Микроконтроллеры без схемы сброса BOR	Микроконтроллеры со сбросом BOR
PIC16C62	PIC16C62A
PIC16C64	PIC16C64A
PIC16C65	PIC16C65A
PIC16C71	PIC16C711
PIC16C73	PIC16C73A
PIC16C74	PIC16C74A

C.5 Модуль компараторов

Если изменение C1OUT и C2OUT регистра CMCON произошло, когда выполнялась операция чтения (начало такта Q2), то флаг прерываний CMIF может не установиться в '1'.

C.6 Фильтр на выводе -MCLR

В логику сброса микроконтроллера (-MCLR) добавлен фильтр, который предотвращает сброс микроконтроллера короткими импульсами на выводе -MCLR (защита от помех). В таблице C-4 указано в каких микроконтроллерах реализован фильтр на входе -MCLR.

Таблица C-4 Микроконтроллеры с фильтром на входе -MCLR

Микроконтроллер	**Вывод -MCLR**	
	Без фильтра	**С фильтром**
PIC16C61	Да	-
PIC16C62	Да	-
PIC16C62A	-	Да
PIC16CR62	-	Да
PIC16C63	-	Да
PIC16CR63	-	Да
PIC16C64	Да	-
PIC16C64A	-	Да
PIC16CR64	-	Да
PIC16C65	Да	-
PIC16C65A	-	Да
PIC16CR65	-	Да
PIC16C66	-	Да
PIC16C67	-	Да
PIC16C620	-	Да
PIC16C621	-	Да
PIC16C622	-	Да
PIC16C710	-	Да
PIC16C71	Да	-
PIC16C711	-	Да
PIC16C715	-	Да
PIC16C72	-	Да
PIC16CR72	-	Да
PIC16C73	Да	-
PIC16C73A	-	Да
PIC16C74	Да	-
PIC16C74A	-	Да
PIC16C76	-	Да
PIC16C77	-	Да
PIC16C83	Да	-
PIC16C84	Да	-
PIC16F83	Да	-
PIC16F84	Да	-
PIC16C923	-	Да
PIC16C924	-	Да
Все новые микроконтроллеры	-	Да

C.7 Модуль USART

В новых микроконтроллерах среднего семейства применяется оригинальный модуль USART с "высокоскоростным режимом" (если реализован бит BRGH). Используемая схема выборки данных работала не удовлетворительно, поэтому была разработана новая. Отличия методов выборки описано в разделе "Модуль USART". В таблице C-5 указана какие микроконтроллеры используют новую/старую методику выборки данных.

Таблица C-5 Логика выборки данных модуля USART

Микроконтроллер	Логика выборки	
	Старая	Новая
PIC16C63	Да	-
PIC16CR63	Да	-
PIC16C65	Да	-
PIC16C65A	Да	-
PIC16CR65	Да	-
PIC16C66	-	Да
PIC16C67	-	Да
PIC16C73	Да	-
PIC16C73A	Да	-
PIC16C74	Да	-
PIC16C74A	Да	-
PIC16C76	-	Да
PIC16C77	-	Да
Все новые микроконтроллеры с модулем USART	-	Да

C.8 Тактовый генератор

Добавлен новый режим тактового генератора, который позволяет работать микроконтроллеру от внутреннего RC генератора. Выбор режима тактового генератора выполняется во время программирования микроконтроллера в слове конфигурации. Этот режим тактового генератора будет включен во многие будущие микроконтроллеры PICmicro. Смотрите техническую документацию на микроконтроллер.

C.9 Ведомый параллельный порт

В микроконтроллерах PICmicro реализовано два варианта управления ведомым параллельным портом: управление уровнем сигнала; управление фронтом сигнала.

Таблица C-6 Управляющие сигналы ведомого параллельного порта

Микроконтроллер	Управление	
	Уровнем	Фронтом
PIC16C64	Да	-
PIC16C64A	-	Да
PIC16C65	Да	-
PIC16C65A	-	Да
PIC16C67	-	Да
PIC16C74	Да	-
PIC16C74A	-	Да
PIC16C77	-	Да
Все новые микроконтроллеры с модулем PSP	-	Да

Раздел 34. Глоссарий

A

A/D

АЦП

Смотрите Analog to Digital.

Acquisition Time (T_{AQC})

Длительность заряда конденсатора

Этот параметр связан с модулем АЦП. T_{AQC} - интервал времени, в течение которого внутренний конденсатор АЦП заряжается до напряжения подключенного входного канала. Когда бит GO установлен в '1', то аналоговый вход отсоединен от внутреннего конденсатора, выполняется преобразование.

ALU

АЛУ

Арифметико-логическое устройство. Модуль ядра микроконтроллера, отвечающий за математические (сложение, вычитание и др.), логические ("и", "или" и др.) и операции сдвига.

Analog to Digital (A/D)

Аналого-цифровое преобразование

Входной аналоговый сигнал преобразуется в эквивалентный цифровой код.

Assembly Language

Ассемблер

Символический язык программирования, с помощью которого машинные коды представляются в удобно читаемой форме.

B

Bank

Банк

Метод адресации памяти данных. Команды среднего семейства микроконтроллеров PICmicro имеют 7 бит для прямой адресации памяти данных (максимум 128 байт), включая регистры специального назначения. Для того, чтобы была возможность реализовать больший объем памяти данных, она была разбита на банки по 128 байт. Выбрать требуемый банк можно с помощью битов RP1:RP0. Максимум может быть реализовано 4 банка памяти данных (два управляющих бита).

Baud

Бод

Скорость передачи данных по последовательным интерфейсам (эквивалентно бит/с).

BCD

Смотрите Binary Coded Decimal (BCD).

Binary Coded Decimal (BCD)

Двоично-десятичное кодирование чисел

Каждые 4 бита определяют цифру от 0 до 9. Как правило, один байт содержит две цифры (диапазон чисел от 0 до 99).

BOR

Смотрите Brown-out Reset.

Brown-out

Снижение напряжение питания

Условие, при котором напряжение питания опускается ниже определенного значения. Это может происходить при коммутации мощной нагрузки.

Brown-out Reset (BOR)

Сброс по снижению напряжения питания

Схема, которая переводит микроконтроллер в состояние сброса, если напряжение питания стало ниже установленного значения. Некоторые микроконтроллеры имеют интегрированную схему BOR (если нет внутренней схемы BOR, то может возникнуть необходимость в построении внешней схемы).

Bus width

Разрядность шины

Число бит данных передаваемых по шине. Разрядность шины данных - 8 бит. Разрядность шины программ для микроконтроллеров среднего семейства - 14 бит.

C

Capture

Захват

Функция CCP модуля, в которой значение таймера записывается в регистры захвата при возникновении условия захвата.

CCP

Захват, сравнение, широтно-импульсный модулятор (PWM). Этот модуль может быть настроен для работы в одном из режимов: захват данных, сравнение или ШИМ.

Common RAM

Общее ОЗУ

Область памяти данных, которая доступна во всех банках памяти данных. Как правило, эта область имеет адреса от 70h до 7Fh (включительно). В этой области удобно сохранять часто меняющиеся переменные и контекст программы при обработке прерываний.

Compare

Сравнение

Функция модуля CCP, при которой выполнится указанное действие, когда значение таймера соответствует значению в регистрах сравнения.

Compare Register

Регистр сравнения

16 - разрядный регистр, в котором хранится значение, сравниваемое с 16 - разрядным значением таймера. Однократно выполняется указанное действие, когда значение таймера становится равным значению регистра сравнения.

Capture Register

Регистр захвата

16 - разрядный регистр, в который загружается 16 - разрядное значение таймера TMR1, когда выполняется условие захвата.

Configuration Word

Слово конфигурации

В слове конфигурации определяются параметры работы микроконтроллера (режим работы тактового генератора, включение WDT, включение таймера PWRT и др.). Эти параметры определяются во время программирования микроконтроллера. Для микроконтроллеров с EPROM памятью программ значение бита '1' может быть изменено на '0'. Память программ должна быть стерта, чтобы восстановить значение '1'.

Conversion Time (Tconv)

Время преобразования

Параметр связан с модулем АЦП. Интервал времени, необходимый для нормального преобразования входного аналогового сигнала в соответствующий цифровой код.

CPU

ЦПУ

Центральное процессорное устройство. Выполняет декодирование команд, определяет необходимые операнды и требуемую операцию. Управляет работой АЛУ для выполнения логических, арифметических и других операций.

D

D/A

ЦАП

Смотрите Digital to analog.

Data Bus

Шина данных

Шина, необходимая для передачи данных из/в память данных.

Data EEPROM

EEPROM память данных

Электрически перепрограммируемая память данных. Эта память данных может быть запрограммирована командами ЦПУ для сохранения необходимых приложению данных при выключении питания (энергонезависимая память).

Data Memory

Память данных

Память, подключенная к шине данных, выполненная как статическое ОЗУ. В памяти данных размещаются регистры общего и специального назначения.

Direct Addressing

Прямая адресация

Адрес памяти данных содержится в команде микроконтроллера. Обращение будет выполняться к регистру с указанным адресом.

Digital to Analog

Цифро-аналоговое преобразование

Цифровой код преобразуется в соответствующее аналоговое напряжение (ток).

E

EEPROM

Электрически стираемое постоянно запоминающее устройство. Микросхемы, с данным типом памяти программ, могут быть внутрисхемно стерты и повторно запрограммированы.

EPROM

Электрически программируемое постоянное запоминающее устройство. Микросхемы, с данным типом памяти программ, могут быть внутрисхемно запрограммированы. Стирание EPROM памяти выполняется под действием УФ излучения.

EXTRC

Внешняя RC цепочка. Некоторые микроконтроллеры имеют режим тактового генератора с внешней RC цепочкой. Эквивалентно RC режиму тактового генератора.

F

Flash Memory

Flash память

Микросхемы, с данным типом памяти программ, могут быть внутрисхемно стерты и повторно запрограммированы. Flash технология памяти программ функционально эквивалентна EEPROM памяти.

F_{osc}

Тактовая частота микроконтроллера.

G

GIO

Общий порт ввода/вывода.

GPIO

Универсальный порт ввода/вывода

GPR

Регистры общего назначения (ОЗУ). Эти регистры могут использоваться для хранения переменных программы пользователя.

H

Harvard Architecture

Гарвардская архитектура

В данной архитектуре микроконтроллеров шины памяти данных и памяти программ разделены между собой. Это позволяет выполнять одновременный доступ к памяти программ и памяти данных, что увеличивает производительность ядра микроконтроллера.

Holding Capacitor

Удерживающий конденсатор

Конденсатор расположен в модуле АЦП. Этот конденсатор должен заряжаться до напряжения на аналоговом входе перед началом преобразования. Как только начато преобразование, конденсатор отсоединяется от аналогового входа. Напряжение на конденсаторе используется для преобразования.

HS

Высокоскоростной режим генератора

Один из режимов тактового генератора. Тактовый генератор настроен таким образом, чтобы поддерживать высокую тактовую частоту микроконтроллера (от 4МГц до 20МГц).

I

I^2C

Inter-Integrated Circuit. Двухпроводный интерфейс связи. Один из режимов SSP модуля.

Indirect Addressing

Косвенная адресация

Случай, когда адрес регистра памяти данных не содержится в команде. Обращение выполняется к регистру INDF, а операция выполняется с регистром, адрес которого указан в FSR. Всегда будет выполняться обращение к регистру с адресом, который записан в регистр FSR.

Instruction Bus

Шина команд

Шина для передачи кода команды из памяти программ в ЦПУ.

Instruction Fetch

Выборка команды

Поскольку реализована гарвардская архитектура, то одновременно происходит выполнение текущей команды и выборка следующей. Как только будет выполнена текущая команда, следующая команда подготовлена к детектированию.

Instruction cycle

Цикл команды

Выполнение каждой команды состоит из нескольких действий: декодирование, чтение данных, выполнение, запись данных. Некоторые команды могут содержать не все действия (см. описание конкретной команды). Цикл команды (T_{CY}) состоит из четырех тактов генератора (T_{OSC}).

Interrupt

Прерывания

Событие, по которому ЦПУ вынужден перевести выполнение программы по адресу вектора прерываний (0004h). Перед изменением значение счетчика команд PC текущее значение сохраняется в вершине стека, чтобы была возможность продолжить выполнение программы.

INTRC

Внутренняя RC цепочка. Некоторые микроконтроллеры имеют режим тактового генератора с внутренней RC цепочкой.

L

LCD

ЖКИ

Жидкокристаллический дисплей. Используется для визуального контроля работы устройства.

LED

Светодиод

Используется для визуального контроля работы устройства.

Literal

Константа

Неизменяемое значение, которое входит в состав команды.

Long Word Instruction

Длинное слово команды

В слово команды входит вся необходимая информация для выполнения операции (код операции и данные). Выполнение и выборка команды происходит за один машинный цикл, т.к. все команды однословные.

LP

Низкоскоростной режим генератора

Один из режимов тактового генератора. Тактовый генератор настроен таким образом, чтобы поддерживать низкую тактовую частоту микроконтроллера (до 200кГц).

LSb

Самый младший бит.

LSB

Самый младший байт.

M

Machine cycle

Машинный цикл

Единица времени выполнения программы микроконтроллера. Для PICmicro эта единица времени равна 4 тактам тактового генератора (4 T_{OSC}). Обозначается как T_{CY}.

MSb

Самый старший бит.

MSB

Самый старший байт.

N

Non-Return to Zero

Без возвращения к нулю

Метод кодирования данных при передаче по каналам связи. Логическая '1' передается как высокий уровень сигнала, логический '0' - как низкий уровень сигнала. Уровень сигнала в линии по умолчанию - высокий.

NRZ

Смотрите Non-Return to Zero.

O

Opcode

Код операции

Часть 14-разрядного слова команды, определяющая выполняемую операцию. Код операции может иметь разную длину в зависимости от типа команды (от 4 бит). В остальной части слова команды содержится аргумент.

Oscillator Start-up Timer (OST)

Таймер запуска генератора

Таймер отсчитывает 1024 такта генератора пред отпусканием внутреннего сигнала сброса микроконтроллера.

OST

Смотрите Oscillator Start-up Timer.

P

Pages

Страницы

Метод адресации памяти программ. Микроконтроллеры среднего семейство имеют в слове команд CALL и GOTO 11 - разрядное поле для адресации памяти программ, что позволяет непосредственно адресовать 2кслов памяти. Для адресации большего объема памяти вся память программ была разделена на страницы по 2кслова. Выбрать нужную страницу можно настройкой битов в регистре PCLATH<5:4>. Всего может быть реализовано 4 страницы памяти программ (два управляющих бита).

Parallel Slave Port (PSP)

Ведомый параллельный порт

Параллельный коммуникационный 8 - разрядный порт для подключения к шине микропроцессора.

POP

Термин, обозначающий восстановление информации из стека (программными или аппаратными средствами). Смотрите PUSH.

Postscaler

Выходной делитель

Схема, замедляющая возникновение прерывания (или сброс WDT) от таймера/счетчика.

Power-on Reset (POR)

Сброс по включению питания

Схема, обнаруживающая повышение напряжения питания от уровня 0В. Если напряжение повышается с 0В, то происходит сброс по включению питания и запускается таймер PWRT.

Power-up Timer (PWRT)

Таймер включения питания

Таймер, удерживающий микроконтроллер в состоянии сброса после выполнения сброса POR, чтобы позволить напряжению питания достигнуть номинального уровня. После завершения отсчета таймера PWRT, запускается таймер OST, если он включен (таймер OST включен для любого режима тактового генератора с кварцевым или керамическим резонатором).

Prescaler

Предделитель

Схема, уменьшающая частоту входного тактового сигнала для таймера/счетчика.

Program Bus

Шина программ

Шина предназначенная для передачи кода команды из памяти программ в ЦПУ.

Program Counter

Счетчик команд

Регистр счетчика команд, в котором хранится адрес следующей выполняемой команды.

Program Memory

Память программ

Любая память, подключенная к шине памяти программ. Статические данные могут сохраняться в памяти программ (например в виде таблиц).

PSP

Смотрите Parallel Slave Port.

Pulse Width Modulation (PWM)

Широтно-импульсная модуляция (ШИМ)

Последовательный сигнал, информация в котором представлена как длительность импульса высокого уровня с постоянной частотой. Вывод PWM модуля CCP требует минимального программного обеспечения для генерации ШИМ сигнала.

PUSH

Термин, обозначающий сохранение информации в стеке (программными или аппаратными средствами). Смотрите POP.

PWM

ШИМ

Смотрите Pulse Width Modulation.

Q

Q - cycles

Q - циклы

Тоже самое, что и цикл тактового генератора. 4 Q - цикла равно циклу команд T_{CY}.

R

RC

Резистор-конденсатор

Заданный по умолчанию режим тактового генератора микроконтроллера. Наиболее дешевый (менее точный) режим тактового генератора. Максимальная рекомендованная частота 4МГц. (см. EXTRC).

Read-Modify-Write

Чтение - Модификация - Запись

Обозначение операции - чтение данных из регистра, изменение значения, запись нового значения в регистр. Это может быть выполнено в одном или нескольких циклах команды.

Register File

Файл регистров

Память данных с регистрами общего и специального назначения.

ROM

ПЗУ

Постоянное запоминающее устройство. Память, которая запрограммирована и не может быть изменена.

S

Sampling Time

Время выборки

Интервал времени, необходимый для получения одного результат преобразования АЦП. Он включает время заряда конденсатора и время преобразования.

Serial Peripheral Interface (SPI)

Последовательный периферийный интерфейс

Один из режимов модуля SSP. Как правило 3-х проводной интерфейс: линия входящих данных, линия исходящих данных, линия синхронизации. Это синхронный интерфейс, т.к. присутствует сигнал синхронизации.

SFR

Регистры специального назначения, содержащие биты управления ядром микроконтроллера и периферийными модулями.

Single cycle instruction

Одно-цикловые команды

Команды, которые выполняются за один машинный цикл (T_{CY}).

Sleep

Режим пониженного энергопотребления с выключенным тактовым генератором. В этом режиме микроконтроллер потребляет минимальный ток. Некоторые периферийные модули могут продолжать работать в Sleep режиме.

Special Function Registers (SFR)

Регистры специального назначения.

Содержат биты управления ядром микроконтроллера и периферийными модулями.

SPI

Смотрите Serial Peripheral Interface.

Stack

Стек

Часть ЦПУ, в которой сохраняется адрес возврата для продолжения выполнения программы. Стек загружается из счетчика команд при выполнении команды CALL или возникновении прерывания.

T

T_{AD}
Время получения одного бита результата при выполнении аналого-цифрового преобразования.

T_{CY}
Длительность выполнения одно-цикловой команды микроконтроллера (4 T_{OSC}).

T_{OSC}
Период тактового генератора микроконтроллера.

U

USART
Универсальный синхронно-асинхронный приемопередатчик. Этот периферийный модуль может использоваться как полнодуплексный последовательный интерфейс связи или полудуплексный синхронный интерфейс. В асинхронном режиме может использоваться для связи с персональным компьютером.

V

Voltage Reference (V_{REF})
Источник опорного напряжения
Уровень напряжения, который может использоваться как опорный для модуля АЦП или модуля компараторов.

Von Neumann Architecture
Традиционная архитектура
Память программ и память данных находятся в одной и той же области. Это означает, что обращение к памяти программ и памяти данных выполняется последовательно (меньшая производительность ядра).

W

W Register

Смотрите Working Register.

Watchdog Timer (WDT)

Сторожевой таймер

Применяется для улучшения помехоустойчивости устройства. WDT выполняет сброс микроконтроллера, если не был вовремя очищен, что позволяет предотвратить "зависание" программы. Источником тактового сигнала для WDT является отдельный внутренний RC генератор.

WDT

Смотрите Watchdog Timer.

Working Register (W)

Рабочий регистр (W)

Этот регистр можно рассматривать как аккумулятор микроконтроллера. Используется как операнд в АЛУ при выполнении команд с двумя операндами.

X

XT

Один из режимов тактового генератора. Применяется при тактовой частоте от 100кГц до 4МГц.

Уважаемые господа!

ООО «Микро-Чип» поставляет полную номенклатуру комплектующих фирмы **Microchip Technology Inc** и осуществляет качественную техническую поддержку на русском языке.

С техническими вопросами Вы можете обращаться по адресу support@microchip.ru

По вопросам поставок комплектующих Вы можете обращаться к нам по телефонам:
(095) 963-9601
(095) 737-7545
и адресу sales@microchip.ru

На сайте
www.microchip.ru
Вы можете узнать последние новости нашей фирмы, найти техническую документацию и информацию по наличию комплектующих на складе.