English Version
Die deutsche Version
Карта сайта

Поделиться

...
Расширенный поиск

Нашли опечатку?
Пожалуйста, сообщите об этом - просто выделите ошибочное слово или фразу и нажмите **Shift Enter**.

Блог одного Сумасшествия
Светлана, видеомонтаж
ВКонтакте
Телеграм

ATmega32U4: последовательный интерфейс I2C

Добавил(а) microsin

В микроконтроллер **ATmega32U4**, как и почти во все микроконтроллеры **AVR** ATmega, встроен аппаратный интерфейс **I2C**. По терминологии Atmel он называется Two Wire Interface, сокращенно **TWI**, так что I2C и TWI это одно и то же. Этот аппаратный TWI обладает следующими возможностями:

- Простой, но в то же время мощный и гибкий коммуникационный интерфейс, которому нужны только 2 сигнальных линии.
- Поддерживается работа в обоих режимах шины I2C (Master и Slave).
- Микроконтроллер (MCU) может работать либо как передатчик, либо как приемник.
- Стандартное 7-разрядное адресное пространство шины I2C позволяет одновременно адресовать (подключить параллельно к шине) до 128 разных Slave-устройств.
- Поддерживается арбитраж нескольких мастеров на одной шине (Multi-master Arbitration).
- Скорость передачи до 400 кГц (400 кГц это так называемый "быстрый" режим I2C).
- Выходные драйверы с ограничением скорости переключения (Slew-rate Limited Output Drivers).
- Схема подавления помех удаляет выбросы на сигналах шины.
- Полностью программируемый адрес Slave-устройства с поддержкой функции общего вызова (General Call).
- Распознавание адреса выводит AVR из сна, когда он находится в режиме пониженного энергопотребления (Sleep Mode).

[**Краткое описание I2C/TWI**]

2-wire Serial Interface (TWI) идеально подходит для приложений, где используется микроконтроллер. Протокол TWI позволяет разработчику системы соединить друг с другом до 128 разных устройств, используя только 2 двунаправленных сигнала шины: такты (SCL) и данные (SDA). Для организации шины требуется минимальный внешний обвес - всего лишь 2 верхних нагрузочных резистора (pull-up), по одному на каждый сигнал шины. Все устройства, подключенные параллельно к одной шине, должны иметь индивидуальный адрес и механизмы разрешения конфликтных ситуаций на шине, свойственных протоколу TWI.

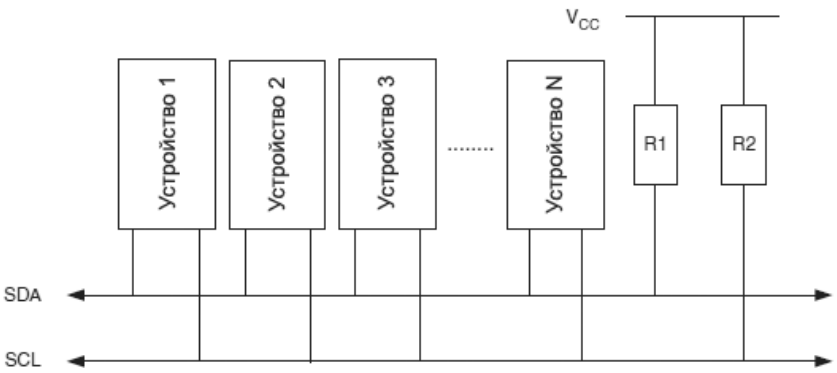

Рис. 20-1. Соединения на шине TWI/I2C.

Терминология TWI. В этой документации (перевод даташита [1]) часто встречаются следующие определения:

Master. Устройство на шине (обычно это MCU), которое инициирует и завершает транзакцию передачи данных (в любом направлении). Master также генерирует импульсы тактов SCL.

Slave. Это устройство адресуется Master-устройством.

Передатчик. Устройство, которое помещает свои данные на шину (выводит их на сигнал SDA).

Приемник. Устройство, которое считывает данные с шины (с сигнала SDA).

START. Сигнал начала передачи на шине (далее START), задаваемый специальной комбинацией уровней SCL и SDA.

STOP. Сигнал завершения передачи на шине (далее STOP), задаваемый специальной комбинацией уровней SCL и SDA.

ACK. Сигнал положительного подтверждения от подчиненного устройства.

NACK. Сигнал отрицательного подтверждения от подчиненного устройства.

General Call. Функция общего вызова, означающая передачу всем подчиненным устройствам на шине (соответствует зарезервированному адресу 0).

Бит снижения потребления энергии TWI (Power Reduction TWI, бит PRTWI) в регистре Power Reduction Register 0 - PRR0 [1] должен быть установлен в лог. 0, чтобы разрешить работу TWI.

Электрические соединения. Как показано на рисунке 20-1 выше, оба сигнала шины подтянуты к плюсу питания (VCC) через pull-up резисторы (обычно их номинал от 4.7 кОм до 10 кОм, и может меняться в зависимости от емкости шины и скорости передачи). Драйверы шины TWI-совместимых устройств всегда имеют выход с общим стоком или с общим коллектором. Этим реализуется логическая функция "проводное И", которая важна для организации работы интерфейса I2C. Низкий уровень (лог. 0) на шине генерируется, когда одно или большее количество устройств на шине выводит сигнал лог. 0. Высокий уровень на шине (лог. 1) появляется, когда все устройства переводят свои выходы отключенное (третье) состояние, позволяя тем самым pull-up резисторам подтянуть уровень к лог. 1. Обратите внимание, что все подключенные к шине TWI устройства AVR MCU должны получать питание, чтобы была возможна любая операция на шине.

Количество одновременно подключенных к шине устройств ограничивается только пределом емкости сигналов шины 400 пикофарад и 7-битным адресным пространством для подчиненных устройств. Подробная спецификация электрических характеристик шины TWI дана в секции "SPI Timing Characteristics" на странице 388 даташита [1]. Здесь представлено два отличающихся набора характеристик, один относится к стандартной скорости 100 кГц и ниже, и другая к скоростям до 400 кГц.

[**Передача данных и формат фрейма**]

Передача бит. Каждый бит данных, передаваемый по шине TWI через сигнал SDA, сопровождается импульсом тактов на сигнале SCK. Уровень сигнала SDA должен оставаться стабильным, пока сигнал тактов SCK находится в состоянии лог. 1. Единственное исключение из этого правила относится к генерации start condition и stop condition.

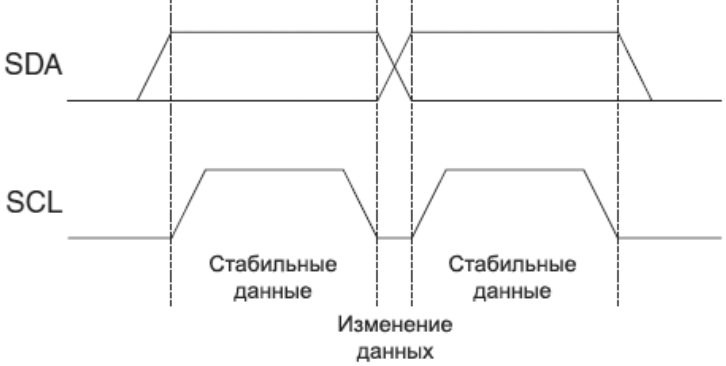

Рис. 20-2. Достоверность данных.

Сигналы START и STOP. Master инициирует и завершает передачу данных. Передача инициируется, когда Master выдает на шину сигнал START, и передача завершается, когда Master выдает на шину сигнал STOP. Между сигналами START и STOP шина считается занятой, и никакие другие устройства master не должны пытаться получить управление шиной. Специальный случай, когда выдается новый сигнал START между сигналами START и STOP. Это называется повторным сигналом запуска (REPEATED START, и используется, когда Master хочет инициировать новую передачу без завершения своего контроля над шиной. После сигнала REPEATED START шина считается занятой до следующего сигнала STOP. Это идентично поведению сигнала START, и поэтому далее в тексте описание START относится одинаково и к описанию REPEATED START, если не указано нечто иное. Далее описывается, что сигналы START и STOP генерируются путем изменения уровня SDA, когда SCL находится в состоянии лог. 1.

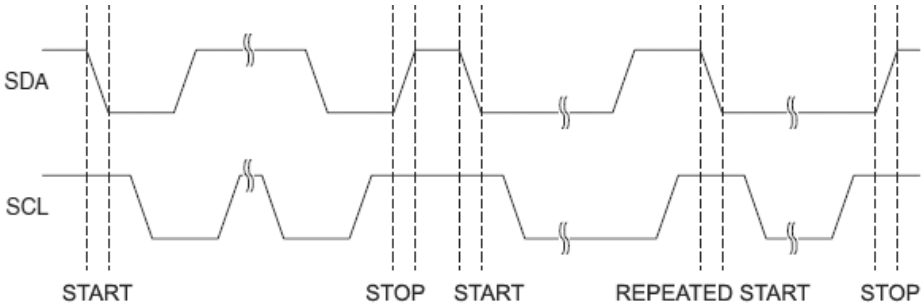

Рис. 20-3. Сигналы START, REPEATED START и STOP.

Формат пакета адреса. Все пакеты адреса передаются по шине TWI посылками из 9 бит, один из которых бит управления чтением/записью (READ/WRITE bit) и один бит подтверждения (acknowledge bit). Если бит READ/WRITE установлен, то будет выполнена операция чтения, иначе будет выполнена операция записи. Когда Slave-устройство распознает свой адрес, оно должно перевести свой сигнал SDA в лог. 0, когда появился девятый импульс SCL, что называется положительным подтверждением (ACK). Если адресованное Slave-устройство занято, или по какой-то другой причине, или если адресованное устройство не присутствует на шине, то в момент девятого импульса SCL сигнал SDA будет оставаться в лог. 1, что называется отрицательным подтверждением (NACK). После этого Master может выдать сигнал STOP или сигнал REPEATED START, чтобы инициировать новую передачу. Пакет адреса, состоящий из slave-адреса и бита READ или WRITE, называется соответственно SLA+R или SLA+W.

Сначала передается старший бит (MSB) адреса. Разработчик системы свободен в распределении адресного пространства шины (руководствуясь при этом параметрами адресов подчиненных устройств), однако адрес 0000 000 зарезервирован для функции общего вызова (general call).

Когда выдается general call, все slave-устройства должны ответить переводом сигнала SDA в лог. 0 на слоте бита подтверждения (т. е. выдать ACK). Функция general call используется, когда Master хочет передать одинаковое сообщение одновременно всем slave-устройствам в системе. Когда за адресом general call идет бит WRITE, все slave-устройства, настроенные на подтверждение general call, будут выводить на лог. 0 на SDA в момент цикла подтверждения. Последующие пакеты данных будут приняты всеми slave-устройствами, которые подтвердили general call. Обратите внимание, что передача адреса general call, за которым идет биты READ, будет бессмысленной, потому что приведет к коллизии, если несколько slave-устройств начнут передавать разные данные.

Все адреса формата 1111 xxx должны быть зарезервированы для будущего использования.

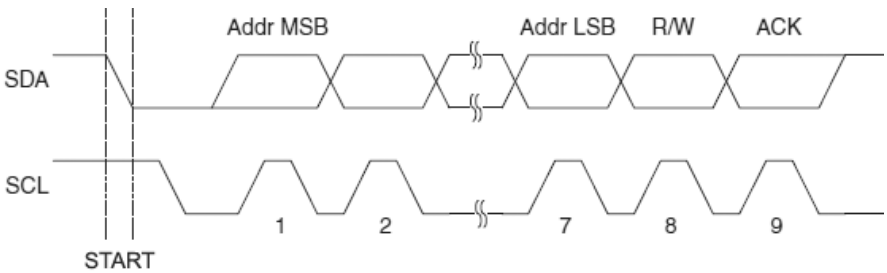

Рис. 20-4. Формат пакета адреса.

Примечание: Addr MSB означает старший бит адреса, Addr LSB младший бит адреса.

Формат пакета данных. Все пакеты, передаваемые по шине TWI, имеют длину 9 бит, состоящие из байта данных (8 бит) и бита подтверждения. Во время передачи данных Master генерирует сигналы START и STOP, в то время как приемник (slave-устройство) отвечает за выдачу бита подтверждения приема. Как уже упоминалось, положительное подтверждение (ACK) выдается приемником путем перевода сигнала SDA в лог. 0 во время девятого импульса SCL. Если приемник оставил линию SDA в лог. 1, то это означает отрицательное подтверждение (NACK). Когда приемник принял последний байт, или по некоторым причинам не может принимать больше байты, он должен оповестить об этом передатчик сигналом NACK после последнего байта. Старший бит (MSB) байта данных передается первым.

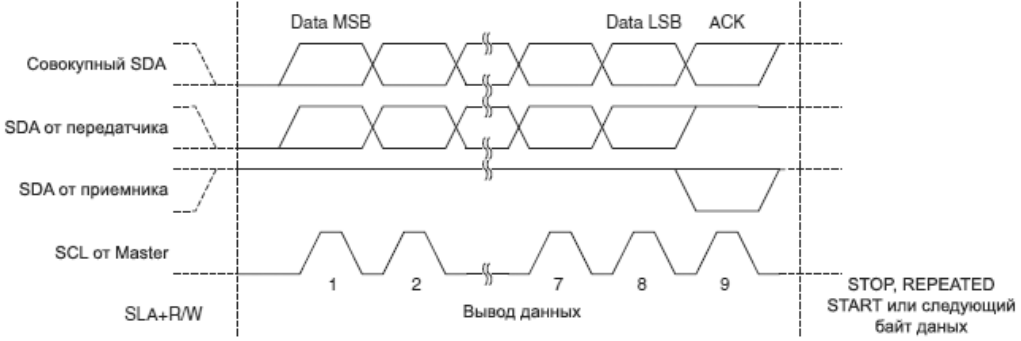

Рис. 20-5. Формат пакета данных.

Примечание: Data MSB означает старший бит данных, Data LSB младший бит данных.

Комбинирование в передаче пакетов адреса и данных. Передача обычно состоит из START, SLA+R или SLA+W, одного или большего количества пакетов данных и STOP. Пустое сообщение, состоящее из START, за которым следует сразу STOP, является незаконным. Обратите внимание, что проводное И на сигнале SCL может использоваться для "рукопожатия" между Master и Slave. Slave-устройство может расширить время генерации SCL путем перевода сигнала SCL в лог. 0. Это полезно, если тактовая частота, установленная Master, слишком велика для Slave-устройства, и ему требуется дополнительное время для обработки между передачами данных. Расширение Slave-устройством периода лог. 0 SCL не повлияет на длительность периода лог. 1 SCL, которая задается Master. Как следствие Slave может снизить скорость передачи данных TWI путем удлинения цикла SCL.

На рис. 20-6 показаны типовые передачи данных. Обратите внимание, что некоторые байты данных могут передаваться между SLA+R (или SLA+W) и STOP, в зависимости от реализованного алгоритма программы.

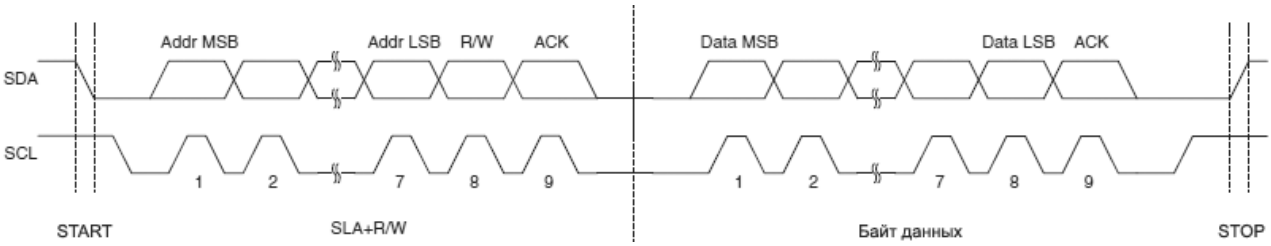

Рис. 20-6. Диаграммы времени передачи данных.

Multi-master, арбитраж и синхронизация

[**Обзор модуля TWI**]

Модуль TWI состоит из нескольких блоков, как показано на рис. 20-9. Все регистры, обведенные толстой линией, доступны через шину данных AVR.

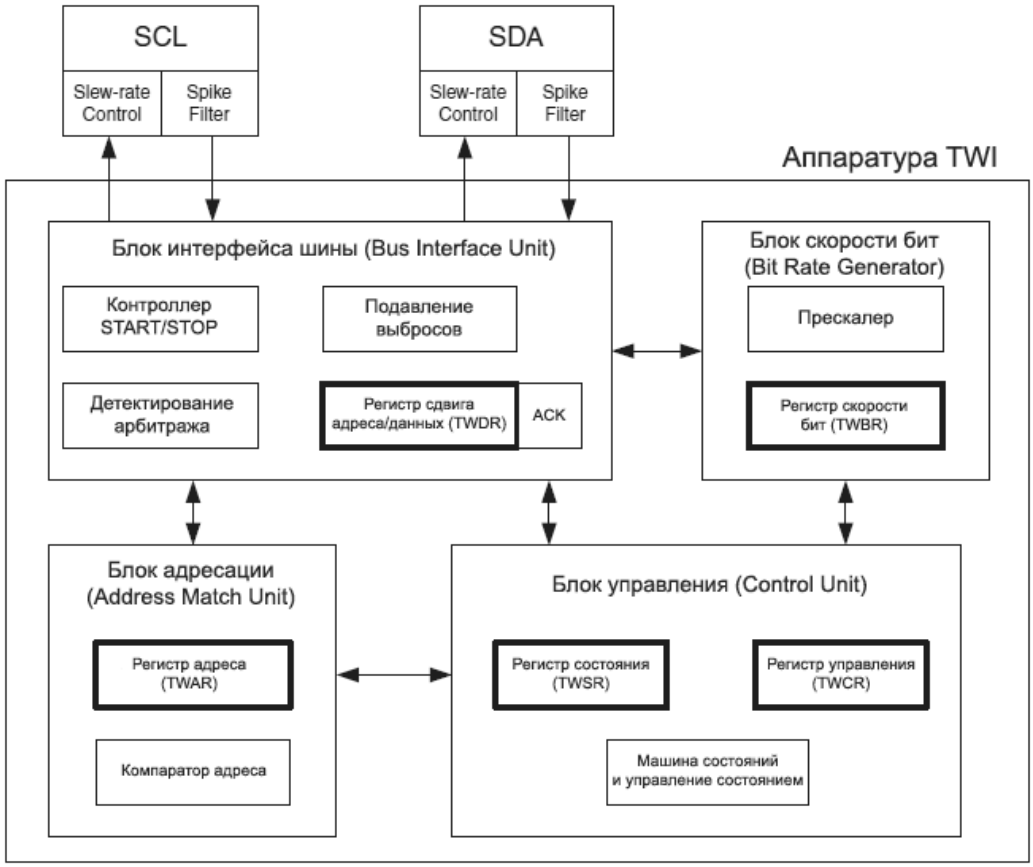

Рис. 20-9. Структурная схема модуля TWI.

Примечание: Slew-rate Control означает узел управления скоростью переключения драйвера, Spike Filter означает фильтр коротких импульсов сигнала.

Выводы сигналов SCL и SDA. Эти ножки подключают AVR TWI к остальной части всей системы. Выходные драйверы содержат ограничитель скорости передачи (slew-rate limiter), чтобы сигналы удовлетворяли спецификации TWI. Входные каскады содержат схему подавления помех, которая отбрасывает импульсы, которые короче 50 нс. Обратите внимание, что можно разрешить внутренние верхние подтягивающие резисторы (pull-up) на ножках AVR путем установки соответствующих бит регистра PORT (у микроконтроллера ATmega32U4 сигналам SCL и SDA соответствуют ножки портов PD0 и PD1 соответственно, т. е. для настройки внутренних pull-upp необходимо установить биты 0 и 1 регистра PORTD). В некоторых простых системах с короткой, незначительно нагруженной шиной I2C внутренние резисторы pull-up могут заменить внешние.

Генератор скорости бит. Этот блок (Bit Rate Generator Unit) управляет периодом сигнала SCL, когда MCU работает в режиме Master. Период SCL управляется настройкой регистра TWI Bit Rate Register (TWBR) и битами прескалера (Prescaler) в регистре TWI Status Register (TWSR). Slave-режим не зависит от настроек Bit Rate или Prescaler, однако тактовая частота CPU в Slave-устройстве MCU должна быть как минимум в 16 раз выше частоты SCL. Имейте в виду, что slave-устройства могут увеличивать длину периода лог. 0 сигнала SCL, увеличивая тем самым период тактов шины TWI.

Генерируемая частота SCL определяется параметрами TWBR и TWPS.

TWBR = значению TWI Bit Rate Register.

TWPS = значению бит прескалера в TWI Status Register.

Формула для частоты SCL:

$$SQL_{freq} = \frac{\text{Тактовая частота CPU}}{16 + 2 \cdot (TWBR) \cdot 4^{TWPS}}$$

Примечание: значение TWBR должно быть 10 или больше, если TWI работает в режиме Master. Если TWBR меньше 10, то Master может генерировать некорректные сигналы SDA и SCL для завершающей части байта. Проблема встречается, когда в режиме TWI Master отправляется Start + SLA+R (или SLA/W) в Slave-устройство (Slave-устройство не обязательно должно быть подключено к шине, чтобы случилось ложное подтверждение).

Блок интерфейса шины. Этот блок (Bus Interface Unit) содержит регистр сдвига данных и адреса (TWDR), контроллер START/STOP и аппаратуру детектирования арбитража. TWDR содержит адрес или байты данных для передачи, или принимаемые адрес или байты данных. В дополнение к 8-битному TWDR блок интерфейса шины также содержит регистр, содержащий передаваемый или принимаемый бит ACK/NACK, который косвенно доступен из программного обеспечения. Однако когда идет прием, этот бит может быть установлен или очищен путем манипуляции регистром TWI Control Register (TWCR). В режиме передатчика значение принятого бита ACK/NACK может быть определено значением в регистре TWSR.

Контроллер START/STOP отвечает за генерацию и детектирование сигналов START, REPEATED START и STOP. Контроллер START/STOP может определить сигналы START и STOP, даже когда AVR MCU находится в одном из режимов сна, что дает возможность вывести подчиненное устройство MCU из режима пониженного энергопотребления, когда он адресован Master-устройством.

Если TWI был инициирован на передачу как Master, аппаратура Arbitration Detection постоянно мониторит передачу. Если TWI проиграл арбитраж, то об этом информируется блок управления (Control Unit). После этого может быть выполнена корректная операция в зависимости от генерируемых соответствующих кодов состояния.

Блок совпадения адреса. Address Match Unit проверяет, были ли приняты байты адреса, совпадающие с 7-разрядным адресом в TWI Address Register (TWAR). Если бит разрешения общего вызова (TWI General Call Recognition Enable бит, TWGCE) в регистре TWAR установлен в лог. 1, то все приходящие биты также сравниваются с адресом General Call (т. е. с нулями). При совпадении адреса об этом информируется Control Unit, позволяя выполнить корректные действия. TWI может подтвердить (ACK) или не подтвердить (NACK), этот адрес, в зависимости от настроек в TWCR. Блок совпадения адреса может сравнивать адреса, даже когда AVR MCU находится в режиме сна, что дает возможность вывести подчиненное устройство MCU из режима пониженного энергопотребления, когда он адресован Master-устройством. Если во время совпадения адреса произойдет другое прерывание (например INT0), которое разбудит CPU, то TWI оборвет операцию и вернется в режим ожидания. Если это вызывает проблемы, то гарантируйте, что разрешено только прерывание TWI Address Match, когда происходит вход в режим пониженного потребления/выключения (Power-down).

Блок управления. Control Unit мониторит шину TWI и генерирует ответные действия в соответствии с настойками в TWI Control Register (TWCR). Когда событие, произошедшее на шине, требует внимания приложения, устанавливается флаг прерывания TWI (TWINT). На следующем такте ядра регистр состояния TWI Status Register (TWSR) обновится кодом, идентифицирующим событие. TWSR содержит достоверную информацию о состоянии только когда установлен флаг прерывания TWI. Во все другие моменты TWSR содержит специальный код состояния, который показывает недоступность информации о состоянии. Пока установлен флаг TWINT, сигнал SCL удерживается в лог. 0. Это позволяет программе приложения завершить все задачи перед тем, как может продолжиться передача TWI.

Флаг TWINT установится в следующих ситуациях:

- После того, как TWI передал сигнал START или REPEATED START.
- После того, как TWI передал SLA+R или SLA+W.
- После того, как TWI передал байт адреса.
- После того, как TWI проиграл арбитраж.
- После того, как TWI был адресован собственным slave-адресом или функцией general call.
- После того, как TWI принял байт данных.
- После того, как был принят STOP или REPEATED START, когда устройство остается адресованным как Slave-устройство.
- Когда происходит ошибка шины из-за недопустимого сигнала START или STOP.

[**Использование TWI**]

AVR TWI является байт-ориентированным, основанным на прерываниях. Прерывания срабатывают после всех событий шины, наподобие принятия байта или передачи сигнала START. Из-за того, что TWI основан на прерываниях, программа приложения может свободно выполнять другие операции во время передачи байта TWI. Обратите внимание, что бит разрешения прерывания TWI Interrupt Enable (TWIE) в регистре TWCR вместе с битом глобального разрешения прерываний Global Interrupt Enable в регистре SREG позволяет приложению определить, должна ли установка флага TWINT генерировать запрос прерывания. Если бит TWIE очищен, приложение должно опрашивать бит TWINT, чтобы детектировать действия на шине TWI.

Когда установлен флаг TWINT, TWI завершил операцию и ждет от приложения какого-либо ответа. В этом случае регистр TWI Status Register (TWSR) содержит значение, показывающее текущее состояние шины TWI. Программа приложения может принять решение, как TWI должен вести себя на следующем цикле TWI, путем манипуляции регистрами TWCR и TWDR.

Рис. 20-10 показывает простой пример, как приложение может работать с аппаратурой TWI. В этом примере Master хочет передать один байт данных Slave-устройству. Это описание довольно абстрактное, в этой секции будет дано более подробное описание. Также предоставлен простой пример кода, реализующий желаемое поведение.

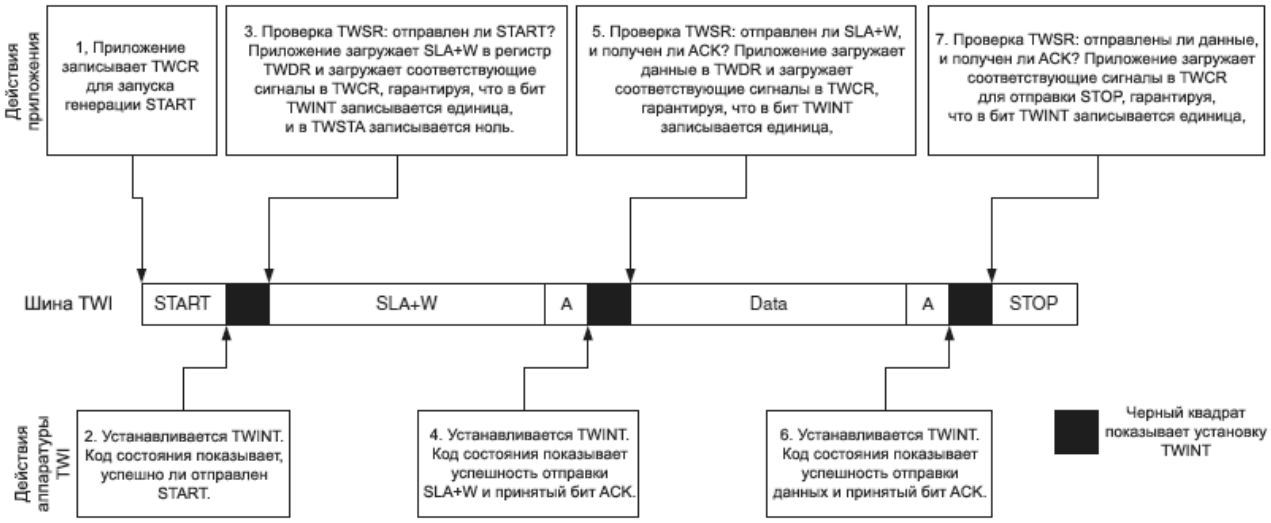

Рис. 20-10. Подключение приложения к TWI в типовой передаче байта.

1. Первый шаг в передаче TWI состоит в генерации START. Это осуществляется записью специального значения в TWCR, что инструктирует аппаратуру TWI передать сигнал START (какое значение надо записать, будет объяснено позже). Однако важно, чтобы в записанном значении был установлен бит TWINT. Запись лог. 1 в TWINT очищает этот флаг. TWI не запустит любую операцию, пока установлен бит TWINT в регистре TWCR. Сразу после того, как приложение очистит TWINT, аппаратура TWI инициирует передачу сигнала START.

2. Когда передан START, установится флаг TWINT в регистре TWCR, и регистр TWSR обновится кодом состояния, показывающим, что сигнал START был успешно отправлен.

3. Теперь приложение должно проверить значение TWSR, чтобы убедиться, что START был успешно отправлен. Если TWSR показывает обратное, то приложение может предпринять некоторые специальные действия наподобие вызова подпрограммы обработки ошибки. Если код статуса оказался ожидаемым, приложение должно загрузить SLA+W в регистр TWDR. Помните, что TWDR используется как для передачи адреса, так и для передачи данных. После того, как TWDR был загружен желаемым значением SLA+W, в регистр TWCR должно быть записано специальное значение, инструктирующее аппаратуру TWI передать SLA+W, присутствующее в TWDR (специальное значение будет описано позже). Однако важно, чтобы в записанном значении был установлен бит TWINT. Запись лог. 1 в TWINT очищает этот флаг. TWI не запустит любую операцию, пока установлен бит TWINT в регистре TWCR. Сразу после того, как приложение очистит TWINT, аппаратура TWI инициирует передачу пакета адреса.

4. Когда пакет адреса передан, установится флаг TWINT в регистре TWCR, и регистр TWSR обновится кодом состояния, показывающим, что пакет адреса быть успешно отправлен. Этот код состояния также содержит информацию о том, подтвержден ли этот адрес Slave-устройством или нет.

Несмотря на простоту этого примера, он показывает основные принципы, действующие во всех передачах TWI:

• Когда TWI завершил операцию и ожидает ответа приложения, устанавливается флаг TWINT. Сигнал SCL подтягивается к лог. 0, пока не будет очищен TWINT.
• Когда установлен флаг TWINT, пользователь должен обновить регистры TWI значением, необходимым для следующего цикла шины TWI. В этом примере TWDR должен быть загружен значением, передаваемым на следующем цикле шины.
• После обновления регистра TWI и выполнения приложением других ожидающих операций, записывается TWCR. При записи TWCR должен быть установлен бит TWINT. Как уже упоминалось, запись единицы в TWINT очистит этот флаг. Тогда TWI начнет выполнение операции работы, установленной настройкой TWCR.

Ниже приведена подборка кода ассемблера и С для этого примера. Имейте в виду, что код подразумевает подключение соответствующих заголовочных файлов директивой #include.

	Пример кода на ассемблере	Пример кода на С	Комментарии

1	`ldir16, (1 << TWINT)|(1 << TWSTA)|`
	`(1 << TWEN)`
	`out TWCR, r16`
		`TWCR = (1 << TWINT)|(1 << TWSTA)`
		`|(1<<TWEN)`
			Отправка START.

2	`wait1:`
	`in r16,TWCR`
	`sbrs r16,TWINT`
	`rjmp wait1`
		`while (!(TWCR & (1 << TWINT)))`	Ожидание установки флага TWINT. Это покажет, что START был передан.

3	`in r16,TWSR`
	`andi r16, 0xF8`
	`cpi r16, START`
	`brne ERROR`
		`if ((TWSR & 0xF8) != START)`
		`ERROR();`
			Проверка значения регистра состояния TWI с маскировкой бит прескалера. Если статус отличается от START, то переход к обработке ошибки.

	`ldi r16, SLA_W`
	`out TWDR, r16`
	`ldi r16, (1 << TWINT)|(1 << TWEN)`
	`out TWCR, r16`
		`TWDR = SLA_W;`
		`TWCR = (1 << TWINT) | (1 << TWEN);`
			Загружает SLA+W в регистр TWDR. Очищает бит TWINT в TWCR для запуска передачи адреса.

4	`wait2:`
	`in r16,TWCR`
	`sbrs r16,TWINT`
	`rjmp wait2`
		`while (!(TWCR & (1 << TWINT)))`	Ожидание установки флага TWINT. Это покажет, что SLA+W был передан, и будет принят ACK/NACK.

5	`in r16,TWSR`
	`andi r16, 0xF8`
	`cpi r16, MT_SLA_ACK`
	`brne ERROR`
		`if ((TWSR & 0xF8) != MT_SLA_ACK)`
		`ERROR();`
			Проверка значения регистра состояния TWI с маскировкой бит прескалера. Если статус отличается от MT_SLA_ACK, то переход к обработке ошибки ERROR.

	`ldi r16, DATA`
	`out TWDR, r16`
	`ldi r16, (1 << TWINT) | (1 << TWEN)`
	`out TWCR, r16`
		`TWDR = DATA;`
		`TWCR = (1 << TWINT) | (1 << TWEN);`
			Загружает данные DATA в регистр TWDR. Очищает бит TWINT в TWCR для запуска передачи данных.

6	`wait3:`
	`in r16,TWCR`
	`sbrs r16,TWINT`
	`rjmp wait3`
		`while (!(TWCR & (1 << TWINT)))`	Ожидание установки флага TWINT. Это показывает, что DATA были переданы, и принят ACK/NACK.

7	`in r16,TWSR`
	`andi r16, 0xF8`
	`cpi r16, MT_DATA_ACK`
	`brne ERROR`
		`if ((TWSR & 0xF8) != MT_DATA_ACK)`
		`ERROR();`
			Проверка значения регистра состояния TWI с маскировкой бит прескалера. Если статус отличается от MT_DATA_ACK, то переход к обработке ошибки.

	`ldi r16, (1 << TWINT)|(1 << TWEN)|` `(1 << TWSTO)out TWCR, r16`	`TWCR = (1 << TWINT)|(1 << TWEN)|` `(1 << TWSTO);`	Передача сигнала STOP.

[**Режимы передачи**]

TWI может работать в одном из 4 основных режимов. Они называются Master Transmitter (MT), Master Receiver (MR), Slave Transmitter (ST) и Slave Receiver (SR). Некоторые из этих режимов можно использовать в одном приложении. Например, TWI может использовать режим MT, чтобы записать данные в TWI EEPROM, режим MR для чтения данных из этого EEPROM. Если в системе используются другие master-устройства, они могут передавать данные в TWI, и тогда должен использоваться режим SR. Программа приложения решает, какой из режимов подходит для определенной ситуации.

В последующих секциях описываются эти режимы. Возможные коды состояния описаны вместе с рисунками, подробно показывающими передачу данных в каждом из режимов. Эти рисунки содержат следующие аббревиатуры:

S: сигнал START
Rs: REPEATED START
R: бит чтения, Read bit (лог. 1 на SDA)
W: бит записи, Write bit (лог. 0 на SDA)
A: бит положительного подтверждения, ACK (лог. 0 на SDA)
~A: бит отрицательного подтверждения, NACK (лог. 1 на SDA)
Data: 8-бит байта данных
P: сигнал STOP
SLA: адрес подчиненного устройства, Slave Address

На рисунках от 20-12 до 20-18 кружочки используются для указания установки флага TWINT. Числа в кружочках показывают код состояния, содержащийся в TWSR, с битами прескалера, замаскированными нулями. В этих точках приложение должно предпринять

какие-либо действия, чтобы продолжить или завершить передачу TWI. Передача TWI приостанавливается, пока флаг TWINT не будет очищен программой.

Когда установлен флаг TWINT, код состояния в TWSR используется для того, чтобы определить соответствующее действие для приложения. Для каждого кода состояния в таблицах от 20-1 до 20-4 даны требуемые действия приложения и описание последующей передачи. Имейте в виду, что в этих таблицах биты прескалера маскированы нулями.

Master Transmitter Mode. В этом режиме некоторое количество байт данных передается приемнику Slave (см. рис. 20-11). Чтобы войти в режим Master, должен быть передан сигнал START. Формат последующего пакета адреса определяет, в какой режим нужно после этого войти - Master Transmitter или Master Receiver. Если передан SLA+W, то входят в режим MT, если передан SLA+R, то входят в режим MR. Подразумевается, что во всех кодах состояния, упомянутых в этой секции, биты прескалера замаскированы нулями.

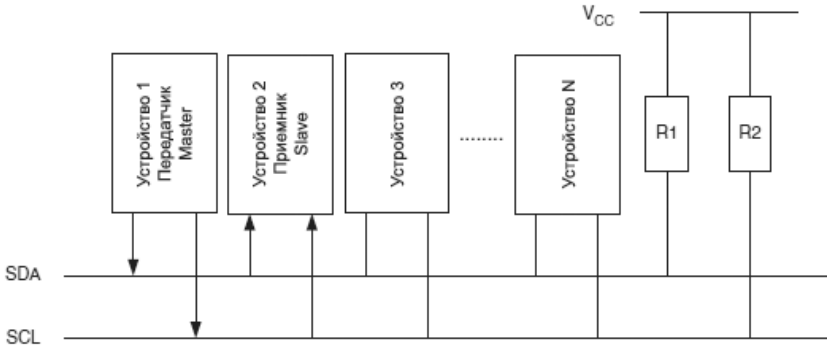

Рис. 20-11. Передача данных в режиме MT.

Сигнал START передается записью в TWCR следующего значения:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	1	0	x	1	0	x

Бит TWEN должен быть установлен, чтобы разрешить TWI, TWSTA должен быть записан в лог. 1, чтобы передать сигнал START, и TWINT должен быть записан в лог. 1, чтобы очистить флаг TWINTg. Затем TWI проверит шину, свободна ли она, и передаст START, как только шина освободится. После того, как START был передан, аппаратно установится флаг TWINT, и в TWSR появится код состояния 0x08 (см. таблицу 20-1). Чтобы войти в режим MT, нужно передать SLA+W. Это осуществляется записью SLA+W в TWDR. После этого должен быть очищен бит TWINT (путем записи в него лог. 1), чтобы передача продолжилась. Это осуществляется записью в TWCR следующего значения:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	0	0	x	1	0	x

Когда передан SLA+W и принят бит подтверждения, TWINT установится снова, и в TWSR могут появиться некоторые коды состояния. Возможные коды состояния для режима Master 0x18, 0x20 или 0x38. Для каждого из этих кодов должно быть предпринято соответствующее действие, что описывается в таблице 20-1.

Когда SLA+W был успешно отправлен, должен быть передан пакет данных. Это осуществляется записью байта данных в TWDR. TWDR должен быть записан только тогда, когда TWINT находится в лог. 1. Если это не соблюсти, то запись будет отброшена, и установится бит коллизии записи Write Collision bit (TWWC) в регистре TWCR. После обновления TWDR должен быть очищен бит TWINT (записью в него лог. 1), чтобы передача продолжилась. Это осуществляется записью в TWCR следующего значения:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	0	0	x	1	0	x

Эта схема повторяется, пока не будет передан последний байт, и передача закончится сигналом STOP или повторной выдачей сигнала START. STOP генерируется записью в TWCR следующего значения:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	0	1	x	1	0	x

REPEATED START генерируется записью в TWCR следующего значения:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	1	0	x	1	0	x

После повторного START (состояние 0x10) интерфейс TWI снова может получить доступ к тому же Slave-устройству, или к новому Slave-устройству без передачи STOP. Повторно выданный START позволяет Master-устройству переключаться между Slave-устройствами, между режимами MT и MR без потери управления над шиной.

Таблица 20-1. Коды состояния для режима MT (в первом столбце настройка прескалера замаскирована нулями).

Код состояния (TWSR)	Состояние шины и аппаратуры TWI	Ответ программы приложения					Следующее действие, предпринимаемое аппаратурой TWI
		В/из TWDR	В регистр TWCR				
			STA	STO	TWINT	TWEA	
0x08	Передан сигнал START	Загрузка SLA+W	0	0	1	x	Будет передан SLA+W и принят ACK или NACK
0x10	Передан сигнал REPEATED START	Загрузка SLA+W	0	0	1	x	Будет передан SLA+W и принят ACK или NACK
		Загрузка SLA+R	0	0	1	x	Будет передан SLA+R, логика переключит TWI в режим MR
0x18	Передан SLA+W, принят ACK	Загрузка байта данных	0	0	1	x	Будет передан байт данных и принят ACK или NACK
		Нет действия с TWDR	1	0	1	x	Будет передан REPEATED START
			0	1	1	x	Будет передан STOP и сброшен флаг TWSTO
			1	1	1	x	Будет передан STOP, за которым последует START, и будет сброшен флаг TWSTO
0x20	Передан SLA+W, принят NACK	Загрузка байта данных	0	0	1	x	Будет передан байт данных и принят ACK или NACK
		Нет действия с TWDR	1	0	1	x	Будет передан REPEATED START
			0	1	1	x	Будет передан STOP и сброшен флаг TWSTO
			1	1	1	x	Будет передан STOP, за которым последует START, и будет сброшен флаг TWSTO

0x28	Передан байт данных, принят ACK	Загрузка байта данных	0	0	1	x	Будет передан байт данных и принят ACK или NACK
			1	0	1	x	Будет передан REPEATED START
		Нет действия с TWDR	0	1	1	x	Будет передан STOP и сброшен флаг TWSTO
			1	1	1	x	Будет передан STOP, за которым последует START, и будет сброшен флаг TWSTO
0x30	Передан байт данных, принят NACK	Загрузка байта данных	0	0	1	x	Будет передан байт данных и принят ACK или NACK
			1	0	1	x	Будет передан REPEATED START
		Нет действия с TWDR	0	1	1	x	Будет передан STOP и сброшен флаг TWSTO
			1	1	1	x	Будет передан STOP, за которым последует START, и будет сброшен флаг TWSTO
0x38	Проигран арбитраж на передаче SLA+W или байт данных	Нет действия с TWDR	0	0	1	x	Освободится шина I2C, интерфейс TWI войдет в режим не адресованного Slave-устройства
			1	0	1	x	Когда шина освободится, будет передан START

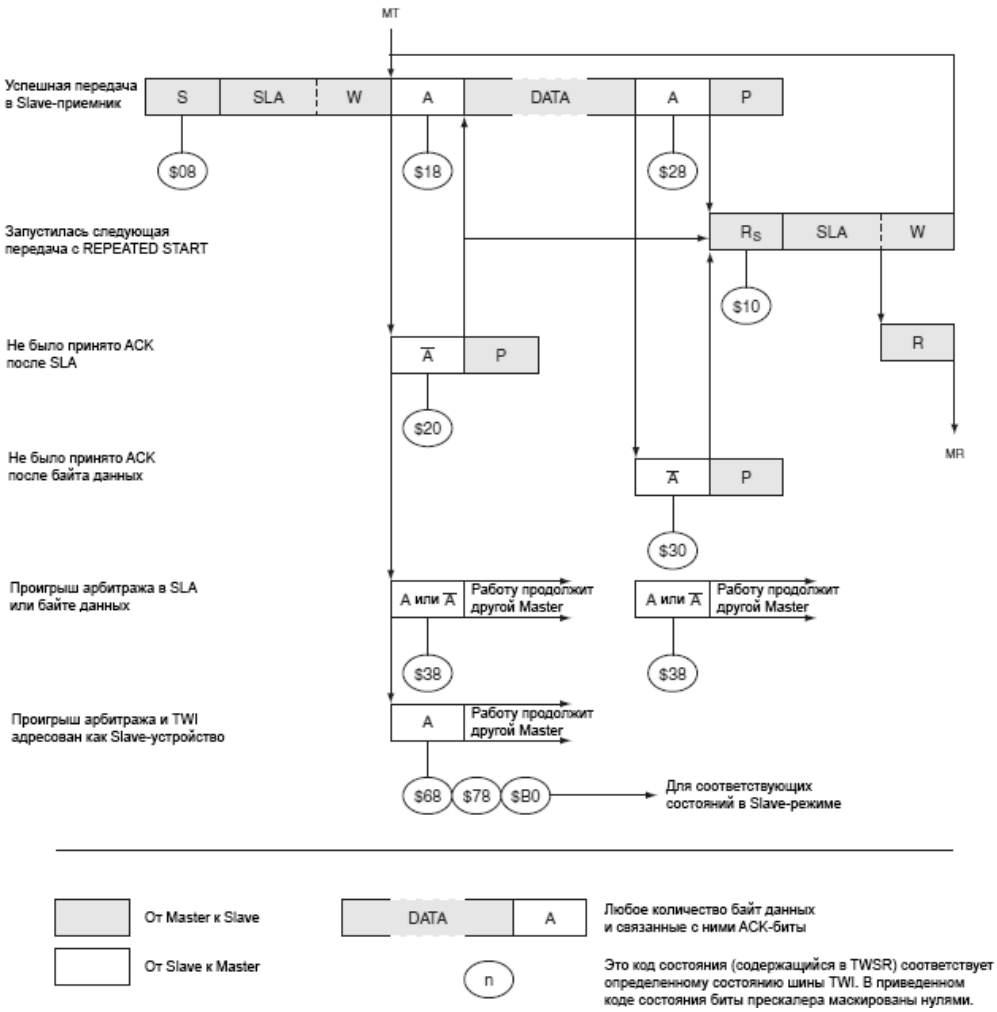

Рис. 20-12. Форматы и коды состояния в режиме MT.

Master Receiver Mode. В режиме MR некоторое количество байт принимается из Slave-передатчика (см. рис. 20-13). Чтобы войти в режим Master, должен быть передан сигнал START. Формат последующего пакета адреса определяет, в какой режим после этого нужно войти - Master Transmitter (MT) или Master Receiver (MR). Если передан SLA+W, то будет осуществлен вход в MT, если передан SLA+R, то будет осуществлен вход в MR. Все коды состояния, упомянутые в этой секции, подразумевают, что в них биты прескалера маскированы нулями.

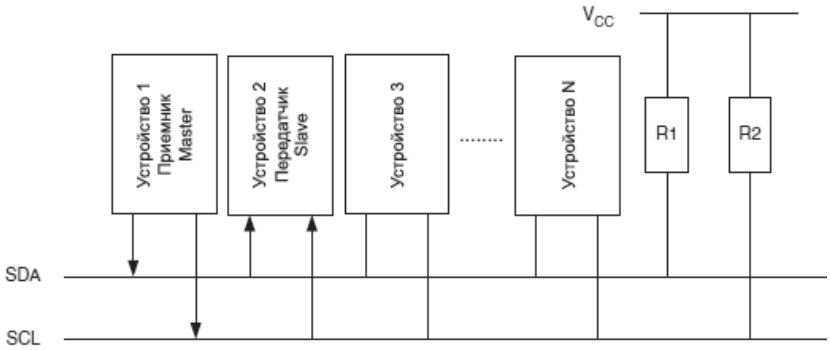

Рис. 20-13. Передача данных в режиме MR.

Сигнал START передается записью в TWCR следующего значения:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	1	0	x	1	0	x

Бит TWEN должен быть установлен, чтобы разрешить TWI, TWSTA должен быть записан в лог. 1, чтобы передать сигнал START, и TWINT должен быть записан в лог. 1, чтобы очистить флаг TWINTg. Затем TWI проверит шину, свободна ли она, и передаст START, как только шина освободится. После того, как START был передан, аппаратно установится флаг TWINT, и в TWSR появится код состояния 0x08 (см.

таблицу 20-1). Чтобы войти в режим MR, нужно передать SLA+R. Это осуществляется записью SLA+R в TWDR. После этого должен быть очищен бит TWINT (путем записи в него лог. 1), чтобы передача продолжилась. Это осуществляется записью в TWCR следующего значения:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	0	0	x	1	0	x

Когда передан SLA+R, и принят бит подтверждения, TWINT установится снова, и в TWSR могут появиться некоторые коды состояния. Возможные коды состояния для режима Master 0x38, 0x40 или 0x48. Для каждого из этих кодов должно быть предпринято соответствующее действие, что описывается в таблице 20-2. Принятые данные могут быть прочитаны из регистра TWDR, когда будет аппаратно установлен флаг TWINT. Эта схема повторяется до тех пор, пока не будет принят последний байт. После того, как последний байт был принят, MR должен оповестить Slave-передатчик отправкой NACK после последнего принятого байта. Передача заканчивается генерацией STOP или повторного START. STOP генерируется записью в TWCR следующего значения:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	0	1	x	1	0	x

REPEATED STOP генерируется записью в TWCR следующего значения:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	1	0	x	1	0	x

После генерации повторного START (состояние 0x10) интерфейс TWI может снова получить доступ к тому же самому Slave-устройству, или к новому Slave-устройству без передачи сигнала STOP. Повторно генерированный START позволяет Master-устройству переключаться между Slave-устройствами, между режимами MT и MR без потери контроля над шиной.

Таблица 20-2. Коды состояния для режима MR (в первом столбце настройка прескалера замаскирована нулями).

Код состояния (TWSR)	Состояние шины и аппаратуры TWI	Ответ программы приложения: В/из TWDR	В регистр TWCR: STA	STO	TWINT	TWEA	Следующее действие, предпринимаемое аппаратурой TWI
0x08	Передан сигнал START	Загрузка SLA+R	0	0	1	x	Будет передан SLA+R и принят ACK или NACK
0x10	Передан сигнал REPEATED START	Загрузка SLA+R	0	0	1	x	Будет передан SLA+R и принят ACK или NACK
		Загрузка SLA+W	0	0	1	x	Будет передан SLA+W, логика переключит TWI в режим MT
0x38	Проигрыш арбитража на передаче SLA+R или бита NACK	Нет действия с TWDR	0	0	1	x	Освободится шина I2C, интерфейс TWI войдет в режим не адресованного Slave-устройства
			1	0	1	x	Когда шина освободится, будет передан START
0x40	Передан SLA+R, принят ACK	Нет действия с TWDR	0	0	1	0	Будет принят байт данных и возвращен NACK
			0	0	1	1	Будет принят байт данных и возвращен ACK
0x48	Передан SLA+R, принят NACK	Нет действия с TWDR	1	0	1	x	Будет передан REPEATED START
			0	1	1	x	Будет передан STOP и сброшен флаг TWSTO
			1	1	1	x	Будет передан STOP, за которым последует START, и будет сброшен флаг TWSTO
0x50	Принят байт данных, возвращен ACK	Чтение байта данных	0	0	1	0	Будет принят байт данных и возвращен NACK
			0	0	1	1	Будет принят байт данных и возвращен ACK
0x58	Принят байт данных, возвращен NACK	Чтение байта данных	1	0	1	x	Будет передан REPEATED START
			0	1	1	x	Будет передан STOP и сброшен флаг TWSTO
			1	1	1	x	Будет передан STOP, за которым последует START, и будет сброшен флаг TWSTO

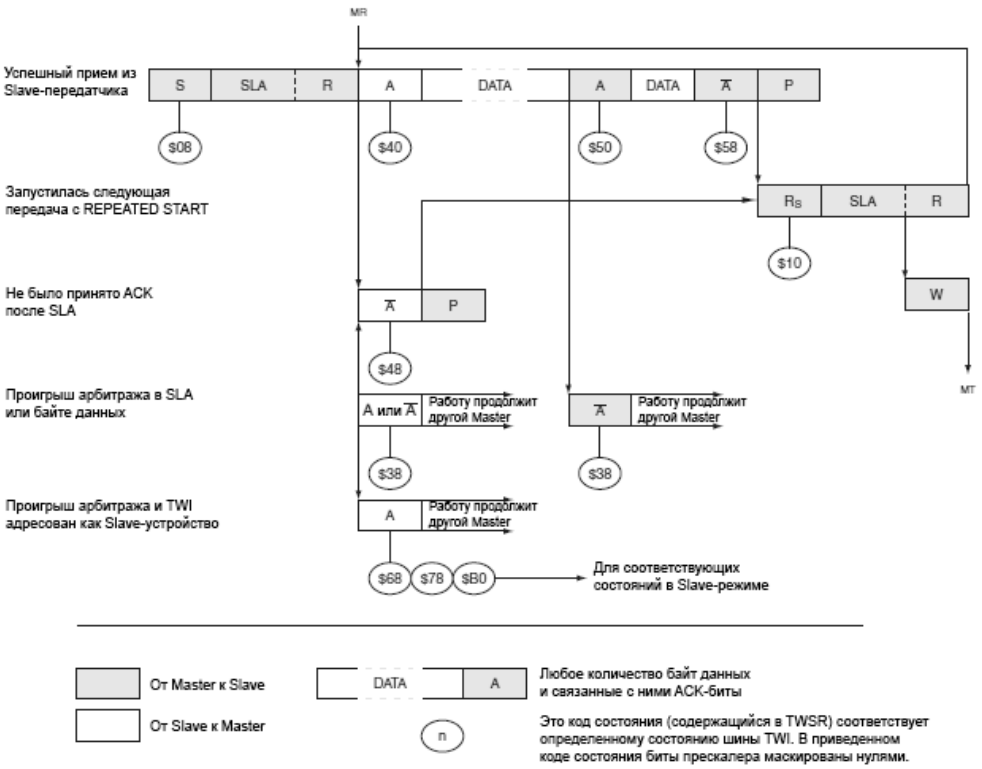

Рис. 20-14. Форматы и коды состояния в режиме MR.

Slave Receiver Mode. В режиме SR некоторое количество байт принимается от передатчика Master (см. рис. 20-15). Подразумевается, что во всех кодах состояния, упомянутых в этой секции, биты прескалера замаскированы нулями.

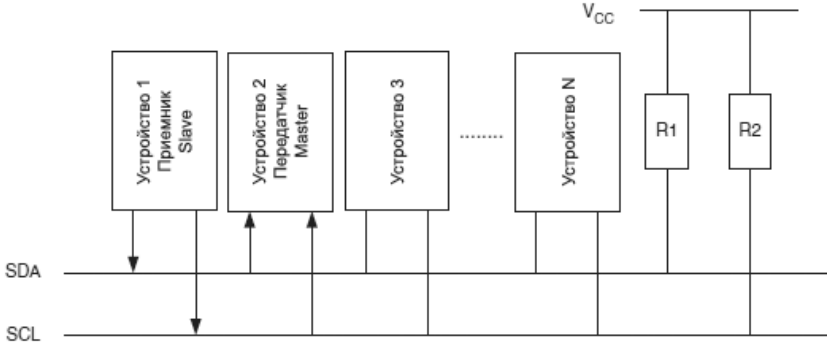

Рис. 20-15. Передача данных в режиме SR.

Для инициации режима SR регистры TWAR и TWCR должны быть инициализированы следующим образом.

Регистр TWAR:

TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE
Собственный адрес Slave-устройства							x

Регистр TWCR:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
0	1	0	0	0	1	0	x

Старшие 7 бит в TWAR это адрес, на который ответит TWI, когда он будет адресован Master-устройством. Если бит LSB установлен, то TWI ответит на адрес общего вызова general call (0x00), иначе этот адрес будет игнорирован.

TWEN должен быть записан в лог. 1, чтобы разрешить TWI. Бит TWEA должен быть записан в лог. 1, чтобы разрешить подтверждение устройством собственного slave-адреса или адреса general call. TWSTA и TWSTO должны быть записаны в лог. 0.

Когда TWAR и TWCR инициализированы, интерфейс TWI ждет, пока не будет адресован своим собственным slave-адресом (или адресом general call, если это разрешено), за которым идет бит направления данных. Если бит направления равен 0 (write), то TWI будет работать в режиме SR, иначе будет осуществлен вход в режим ST. После того, как был принят собственный slave-адрес и бит записи, установится бит TWINT, и допустимый код состояния может быть прочитан из регистра TWSR. Этот код состояния используется для определения соответствующего действия программы. Соответствующее действие для каждого кода состояния показано в таблице 20-3. В режим SR также может быть осуществлен вход, если был потерян арбитраж, когда TWI находится в режиме Master (см. состояния 0x68 и 0x78).

Если во время передачи бит TWEA сброшен, то TWI вернет NACK (лог. 1) на SDA после следующего принятого байта данных. Это может использоваться, чтобы показать Master-устройству, что Slave не может больше принимать байты. Когда TWEA равен 0, TWI не подтверждает собственный slave-адрес. Однако TWI все еще будет мониторить шину и распознавание адреса может быть возобновлено в любое время путем установки TWEA. Это подразумевает, что бит TWEA может использоваться для временной изоляции TWI от шины.

Во всех режимах пониженного энергопотребления, кроме Idle mode, система тактирования TWI отключается. Если установлен бит TWEA, то интерфейс все еще может подтвердить свой собственный slave-адрес или адрес general call путем использования тактов шины. Затем MCU будет выведен из сна и TWI будет удерживать SCL в лог. 0 во время пробуждения до момента очистки флага TWINT (путем записи в него лог. 1). Будущий прием данных будет осуществляться нормальным образом, с обычным тактированием тактами AVR. Если AVR настроен на долгое время пробуждения (long start-up time), то сигнал SCL может удерживаться в лог. 0 долгое время, блокируя другие передачи данных.

Обратите внимание, что регистр данных TWDR не отражает последний байт, представленный на шине, когда MCU выходит из режимов сна.

Таблица 20-3. Коды состояния в режиме SR (в первом столбце настройка прескалера замаскирована нулями).

Код состояния (TWSR)	Состояние шины и аппаратуры TWI	Ответ программы приложения					Следующее действие, предпринимаемое аппаратурой TWI
		В/из TWDR	В регистр TWCR				
			STA	STO	TWINT	TWEA	
0x60	Принят свой собственный адрес SLA+W, возвращен ACK	Нет действия с TWDR	x	0	1	0	Будет принят байт данных и возвращен NACK
			x	0	1	1	Будет принят байт данных и возвращен ACK
0x68	Проигрыш арбитража на SLA+R/W, принят собственный SLA+W, возвращен ACK	Нет действия с TWDR	x	0	1	0	Будет принят байт данных и возвращен NACK
			x	0	1	1	Будет принят байт данных и возвращен ACK
0x70	Принят адрес общего вызова (general call), возвращен ACK	Нет действия с TWDR	x	0	1	0	Будет принят байт данных и возвращен NACK
			x	0	1	1	Будет принят байт данных и возвращен ACK
0x78	Проигрыш арбитража на SLA+R/W, принят адрес общего вызова (general call), возвращен ACK	Нет действия с TWDR	x	0	1	0	Будет принят байт данных и возвращен NACK
			x	0	1	1	Будет принят байт данных и возвращен ACK
0x80	Был ранее успешно адресован собственным SLA+W, был принят байт данных, возвращен ACK	Чтение байта данных	x	0	1	0	Будет принят байт данных и возвращен NACK
			x	0	1	1	Будет принят байт данных и возвращен ACK
0x88	Был ранее успешно адресован собственным SLA+W, был принят байт данных, возвращен NACK	Чтение байта данных	0	0	1	0	Произойдет переключение в режим not addressed Slave mode; нет распознавания собственного SLA или адреса глобального вызова
			0	0	1	1	Произойдет переключение в режим not addressed Slave mode; собственный SLA будет распознан; адрес глобального вызова будет распознан, если TWGCE=1
			1	0	1	0	Произойдет переключение в режим not addressed Slave mode; нет распознавания собственного SLA или адреса глобального вызова; когда шина освободится, будет передан START
			1	0	1	1	Произойдет переключение в режим not addressed Slave mode; будет распознан собственный SLA; адрес глобального вызова будет распознан, если TWGCE=1; когда шина освободится, будет передан START
0x90	Был ранее успешно адресован адресом общего	Чтение байта	x	0	1	0	Будет принят байт данных и возвращен NACK

	вызова, был принят байт данных, возвращен ACK	данных	x	0	1	1	Будет принят байт данных и возвращен ACK
			0	0	1	0	Произойдет переключение в режим not addressed Slave mode; нет распознавания собственного SLA или адреса глобального вызова
			0	0	1	1	Произойдет переключение в режим not addressed Slave mode; собственный SLA будет распознан; адрес глобального вызова будет распознан, если TWGCE=1
0x98	Был ранее успешно адресован адресом общего вызова, был принят байт данных, возвращен NACK	Чтение байта данных	1	0	1	0	Произойдет переключение в режим not addressed Slave mode; нет распознавания собственного SLA или адреса глобального вызова; когда шина освободится, будет передан START
			1	0	1	1	Произойдет переключение в режим not addressed Slave mode; будет распознан собственный SLA; адрес глобального вызова будет распознан, если TWGCE=1; когда шина освободится, будет передан START
			0	0	1	0	Произойдет переключение в режим not addressed Slave mode; нет распознавания собственного SLA или адреса глобального вызова
			0	0	1	1	Произойдет переключение в режим not addressed Slave mode; собственный SLA будет распознан; адрес глобального вызова будет распознан, если TWGCE=1
0xA0	Был принят сигнал STOP или REPEATED START, когда TWI все еще был адресован как Slave-устройство	Нет никакого действия	1	0	1	0	Произойдет переключение в режим not addressed Slave mode; нет распознавания собственного SLA или адреса глобального вызова; когда шина освободится, будет передан START
			1	0	1	1	Произойдет переключение в режим not addressed Slave mode; будет распознан собственный SLA; адрес глобального вызова будет распознан, если TWGCE=1; когда шина освободится, будет передан START

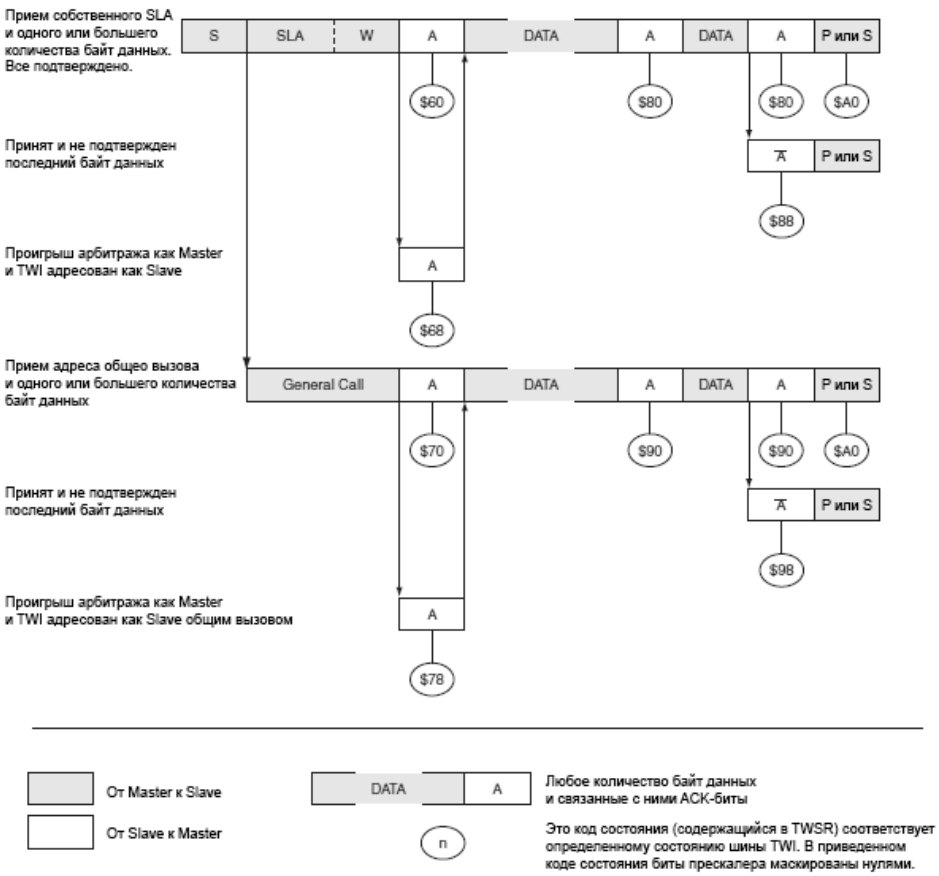

Рис. 20-16. Форматы и коды состояния режима SR.

Slave Transmitter Mode. В режиме ST некоторое количество байт передается в приемник Master (см. рис. 20-17). Подразумевается, что во всех кодах состояния, упомянутых в этой секции, биты прескалера замаскированы нулями.

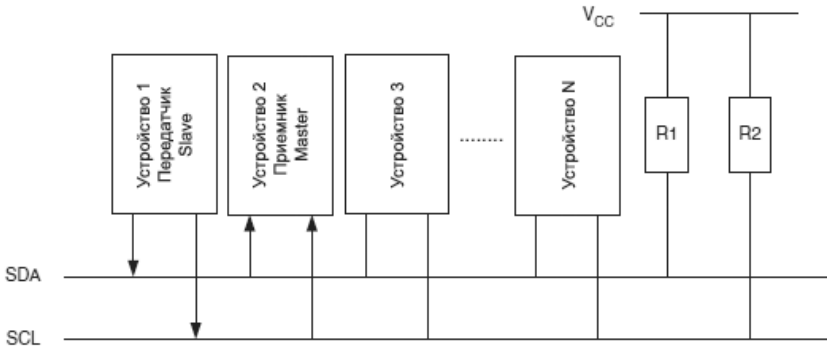

Рис. 20-17. Передача данных в режиме ST.

Для инициации режима ST регистры TWAR и TWCR должны быть инициализированы следующим образом.

Регистр TWAR:

TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE
Собственный адрес Slave-устройства							x

Регистр TWCR:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
0	1	0	0	0	1	0	x

Старшие 7 бит в TWAR это адрес, на который ответит TWI, когда он будет адресован Master-устройством. Если бит LSB установлен, то TWI ответит на адрес общего вызова general call (0x00), иначе этот адрес будет игнорирован.

TWEN должен быть записан в лог. 1, чтобы разрешить TWI. Бит TWEA должен быть записан в лог. 1, чтобы разрешить подтверждение собственного slave-адреса устройства или адреса general call. TWSTA и TWSTO должны быть записаны в лог. 0.

Когда TWAR и TWCR инициализированы, TWI ждет момента адресации по собственному slave-адресу (или по адресу general call, если это разрешено), за которым идет бит направления данных. Если бит направления равен 1 (read), то TWI будет работать в режиме ST, иначе произойдет вход в режим SR. После того, как был принят собственный slave-адрес и бит записи, установится бит TWINT, и допустимый код состояния может быть прочитан из регистра TWSR. Этот код состояния используется для определения соответствующего действия программы. Соответствующее действие для каждого кода состояния показано в таблице 20-4. В режим ST также может быть осуществлен вход, если был потерян арбитраж, когда TWI находится в режиме Master (см. состояние 0xB0).

Если во время передачи бит TWEA сброшен, то TWI передаст последний байт. Произойдет вход в состояние 0xC0 или состояние 0xC8, в зависимости от того, что передал приемник Master после последнего байта - NACK или ACK. TWI переключается в не адресуемый режим подчиненного устройства (not addressed Slave mode), и будет игнорировать Master-устройство, если он продолжает передачу. Таким образом, приемник Master получит все единицы в последовательных данных. В состояние 0xC8 будет осуществлен вход, если Master запрашивает дополнительные байты данных (путем передачи ACK), даже когда Slave передал последний байт (TWEA в лог. 0 и ожидается NACK от Master).

Когда TWEA в лог. 0, TWI не отвечает на свой собственный slave-адрес. Однако шина все еще мониторится, и и распознавание адреса может возобновиться в любой момент установкой TWEA. Это подразумевает, что бит TWEA может использоваться для временной изоляции TWI от шины.

Во всех режимах пониженного энергопотребления, кроме Idle mode, система тактирования TWI отключается. Если установлен бит TWEA, то интерфейс все еще может подтвердить свой собственный slave-адрес или адрес general call путем использования тактов шины. Затем MCU будет выведен из сна и TWI будет удерживать SCL в лог. 0 во время пробуждения до момента очистки флага TWINT (путем записи в него лог. 1). Будущая передача данных будет осуществляться нормальным образом, с обычным тактированием тактами AVR. Если AVR настроен на долгое время пробуждения (long start-up time), то сигнал SCL может удерживаться в лог. 0 долгое время, блокируя другие передачи данных.

Обратите внимание, что регистр данных TWDR не отражает последний байт, представленный на шине, когда MCU выходит из режимов сна.

Таблица 20-4. Коды состояния для режима ST (в первом столбце настройка прескалера замаскирована нулями).

Код состояния (TWSR)	**Состояние шины и аппаратуры TWI**	**Ответ программы приложения**					**Следующее действие, предпринимаемое аппаратурой TWI**
		В/из TWDR	**В регистр TWCR**				
			STA	**STO**	**TWINT**	**TWEA**	
0xA8	Принят свой собственный адрес SLA+R, возвращен ACK	Загрузка байта данных	x	0	1	0	Будет передан последний байт данных и должен быть принят NACK
			x	0	1	1	Будет передан байт данных и должен быть принят ACK
0xB0	Проигрыш арбитража на SLA+R/W, принят собственный SLA+R, возвращен ACK	Загрузка байта данных	x	0	1	0	Будет передан последний байт данных и должен быть принят NACK
			x	0	1	1	Будет передан байт данных и должен быть принят ACK
0xB8	Передан байт данных в TWDR, принят ACK	Загрузка байта данных	x	0	1	0	Будет передан последний байт данных и должен быть принят NACK
			x	0	1	1	Будет передан байт данных и должен быть принят ACK
0xC0	Передан байт данных в TWDR, принят NACK	Нет действия с TWDR	0	0	1	0	Произойдет переключение в режим not addressed Slave; нет распознавания собственного SLA или адреса глобального вызова
			0	0	1	1	Произойдет переключение в режим not addressed Slave; собственный SLA будет распознан; адрес глобального вызова будет распознан, если TWGCE=1
			1	0	1	0	Произойдет переключение в режим not addressed Slave; нет распознавания собственного SLA или адреса глобального вызова; когда шина освободится, будет передан START
			1	0	1	1	Произойдет переключение в режим not addressed Slave; будет распознан собственный SLA; адрес глобального вызова будет распознан, если TWGCE=1; когда шина освободится, будет передан START
0xC8	Передан последний байт данных в TWDR (TWEA=0), принят ACK	Нет действия с TWDR	0	0	1	0	Произойдет переключение в режим not addressed Slave; нет распознавания собственного SLA или адреса глобального вызова
			0	0	1	1	Произойдет переключение в режим not addressed Slave; собственный SLA будет распознан; адрес глобального вызова будет распознан, если TWGCE=1
			1	0	1	0	Произойдет переключение в режим not addressed Slave; нет распознавания собственного SLA или адреса глобального вызова; когда шина освободится, будет передан START
			1	0	1	1	Произойдет переключение в режим not addressed Slave; будет распознан собственный SLA; адрес глобального вызова будет распознан, если TWGCE=1; когда шина освободится, будет передан START

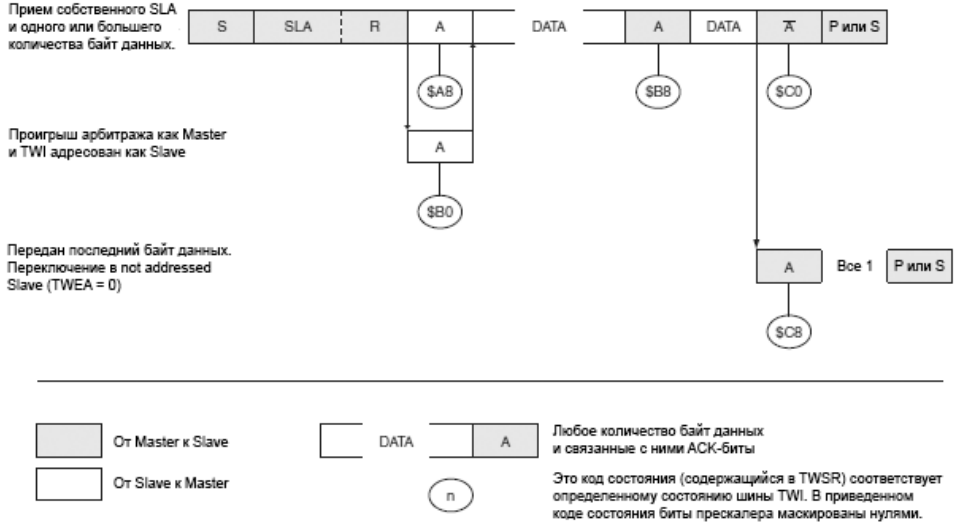

Рис. 20-18. Форматы и коды состояния режима ST.

Различные состояния. Имеется два кода состояния, которые не соответствуют определенным состояниям TWI, см. таблицу ниже.

Состояние 0xF8 показывает, что нет соответствующей информации, потому что не установлен флаг TWINT. Это происходит между другими состояниями, и когда TWI не вовлечен в последовательную передачу.

Состояние 0x00 показывает, что произошла ошибка шины во время передачи. Ошибка шины происходит, когда START или STOP выданы в недопустимой позиции фрейма. Примеры таких недопустимых позиций во время последовательной передачи - передача адреса, передача байта данных или бит подтверждения. Когда произошла ошибка шины, установится бит TWINT. Для восстановления из состояния ошибки шины, должен быть установлен флаг TWSTO и бит TWINT должен быть очищен записью в него лог. 1. Это приведет к тому, что TWI войдет в режим неадресуемости Slave-устройства и к очистке флага TWSTO (другие биты в TWCR не изменятся). Сигналы SDA и SCL освобождаются, и сигнал STOP не передается.

Таблица 20-5. Различные состояния (в первом столбце настройка прескалера замаскирована нулями).

Код состояния (TWSR)	**Состояние шины и аппаратуры TWI**	**Ответ программы приложения**					**Следующее действие, предпринимаемое аппаратурой TWI**
		В/из TWDR	**В регистр TWCR**				
			STA	**STO**	**TWINT**	**TWEA**	
0xF8	Нет доступной информации о состоянии, TWINT=0	Нет действия с TWDR	Нет действия с TWDR				Ожидание или выполнение текущей передачи
0x00	Ошибка шины из-за недопустимого появления сигналов START или STOP	Нет действия с TWDR	0	1	1	x	Влияет только на внутреннюю аппаратуру, по шине не посылается сигнал STOP. Во всех случаях шина освобождается, и TWSTO очищается.

Комбинирование нескольких режимов TWI. В некоторых случаях нужно скомбинировать несколько режимов TWI, чтобы выполнить желаемое действие. Рассмотрим пример чтения данных из микросхемы последовательного EEPROM. Обычно такая передача вовлекает следующие шаги:

1. Должна быть инициирована передача.
2. EEPROM должна быть проинструктирована, какое место будет прочитано.
3. Должно быть выполнено чтение.
4. Должна быть завершена передача.

Обратите внимание, что здесь данные передаются от Master к Slave, и в обратном направлении. Master-устройству нужно проинструктировать Slave-устройство о месте, откуда должно быть осуществлено чтение, что требует использования режима MT. После этого данные должны быть прочитаны из Slave-устройства, что подразумевает использование режима MR. Таким образом, должно меняться направление передачи данных. Master-устройство должно удерживать контроль над шиной во время всех этих шагов, и они должны осуществиться атомарно. Если этот принцип нарушается в системе multi-master, другой Master может изменить указатель данных в EEPROM между шагами 2 и 3, и Master в результате прочитает данные из не той области данных. Такое изменение направления передачи осуществляется передачей REPEATED START между передачей байта адреса и приемом данных. После REPEATED START Master-устройство сохраняет контроль над шиной. Следующий рисунок показывает, как осуществляется эта передача.

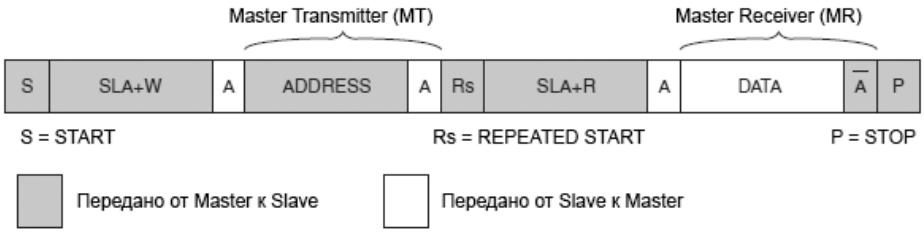

Рис. 20-19. Комбинирование несколько режимов TWI для доступа к Serial EEPROM.

Системы Multi-master и арбитраж

[**Описание регистров TWI**]

TWBR - TWI Bit Rate Register

TWCR - TWI Control Register

TWSR - TWI Status Register

TWDR - TWI Data Register

TWAR - TWI (Slave) Address Register

TWAMR - TWI (Slave) Address Mask Register

[**Ссылки**]

1. ATmega16U4/ATmega32U4 DATASHEET site:microchip.com.
2. Пример использования аппаратного TWI (I2C) ATmega.
3. AVR245: кодовый замок с клавиатурой 4x4 и I2C LCD.
4. Подключение RTC DS1307 к микроконтроллеру AVR.

Добавить комментарий

Имя (обязательное)

E-Mail (обязательное)

Сайт

Осталось: 1000 символов

Подписаться на уведомления о новых комментариях

Обновить

Отправить

JComments