ЮРИЙ МАГДА

bhv®

ПРОГРАММИРОВАНИЕ ПОСЛЕДОВАТЕЛЬНЫХ ИНТЕРФЕЙСОВ

ПОСЛЕДОВАТЕЛЬНЫЕ ИНТЕРФЕЙСЫ: ПРОТОКОЛЫ И СИГНАЛЫ

ПРОГРАММИРОВАНИЕ АСИНХРОННЫХ ПРИЕМОПЕРЕДАТЧИКОВ

ПРОГРАММИРОВАНИЕ ПОСЛЕДОВАТЕЛЬНЫХ ИНТЕРФЕЙСОВ В WINDOWS И LINUX

ПОСЛЕДОВАТЕЛЬНЫЕ ИНТЕРФЕЙСЫ В СЕТИ ИНТЕРНЕТ: ПРАКТИКА ПРОГРАММИРОВАНИЯ

ВИРТУАЛЬНЫЕ ПОСЛЕДОВАТЕЛЬНЫЕ ПОРТЫ В РАЗРАБОТКЕ И ОТЛАДКЕ ПРИЛОЖЕНИЙ

Юрий Магда

ПРОГРАММИРОВАНИЕ ПОСЛЕДОВАТЕЛЬНЫХ ИНТЕРФЕЙСОВ

Санкт-Петербург
«БХВ-Петербург»
2009

УДК 681.3.068
ББК 32.973.26-018.1
М12

Магда Ю. С.

М12 Программирование последовательных интерфейсов. — СПб.: БХВ-Петербург, 2009. — 304 с.: ил. + CD-ROM — (Профессиональное программирование)

ISBN 978-5-9775-0274-0

Рассматривается широкий круг вопросов функционирования последовательных интерфейсов обмена данными. Проанализированы основные протоколы последовательного обмена данными, характеристики сигналов и базовые аппаратные средства на основе асинхронных приемопередатчиков. Подробно изложена методика программирования протоколов последовательного обмена на низком уровне. Значительная часть материала книги посвящена программированию последовательного обмена данными в популярных операционных системах Windows и Linux, а также разработке приложений для Интернета. Рассмотрены методы разработки программного обеспечения с использованием виртуальных последовательных интерфейсов обмена данными. Прилагаемый компакт-диск содержит файлы с исходными текстами описанных в книге программ.

Для программистов

УДК 681.3.068
ББК 32.973.26-018.1

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Евгений Рыбаков*
Зав. редакцией	*Григорий Добин*
Редактор	*Юрий Рожко*
Компьютерная верстка	*Натальи Смирновой*
Корректор	*Виктория Пиотровская*
Дизайн серии	*Инны Тачиной*
Оформление обложки	*Елены Беляевой*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 30.09.08.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 24,51.
Тираж 2000 экз. Заказ №
"БХВ-Петербург", 194354, Санкт-Петербург, ул. Есенина, 5Б.

Санитарно-эпидемиологическое заключение на продукцию № 77.99.60.953.Д.003650.04.08 от 14.04.2008 г. выдано Федеральной службой по надзору в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов
в ГУП "Типография "Наука"
199034, Санкт-Петербург, 9 линия, 12

ISBN 978-5-9775-0274-0

Оглавление

Введение

Последовательный интерфейс является одним из наиболее ранних интерфейсов обмена данными, но, несмотря на это, он до сих пор является одним из самых распространенных. Это обусловлено его относительной простотой в реализации, надежностью, доступностью оборудования. Существенно и то, что программирование последовательного обмена данными довольно легко выполняется с помощью стандартных функций прикладного интерфейса программирования, которые доступны во всех популярных операционных системах, таких как Windows и Linux.

Последовательный интерфейс очень широко используется как в простейших коммуникационных системах, так и в большинстве промышленных систем обмена данными, сбора информации и управления, поскольку при создании надежных распределенных систем управления и контроля стандартам последовательного обмена данными RS-232 и RS-485 реальной альтернативы пока нет.

Основной недостаток последовательного интерфейса — относительно низкая скорость обмена данными по сравнению с другими интерфейсами — в настоящее время успешно преодолевается в новых аппаратных разработках, а аппаратно-программные методы "виртуализации" позволяют работать практически с любым количеством последовательных интерфейсов в одной системе.

Интерес к последовательному интерфейсу в последнее время не только не ослабевает, но даже возрастает, о чем свидетельствует как огромное количество выпускаемого оборудования (промышленного и лабораторного), работающего с последовательным интерфейсом, так и большое количество статей и книг, выпускаемых за рубежом по данной тематике.

В книге рассматриваются вопросы аппаратной архитектуры и программирования последовательного интерфейса в операционных системах Windows и Linux. Детально анализируются аппаратно-архитектурные особенности последовательного интерфейса персонального компьютера, стандарты и интерфейсы, методы тестирования и настройки. Большое внимание уделено анализу практических проектов аппаратно-программных систем обмена данными с использованием интерфейсов RS-232 и RS-485. Все примеры программного кода, приведенные в книге, имеются на компакт-диске.

Книга существенно отличается от аналогичных изданий. Во-первых, подробно рассмотрен аппаратный интерфейс протоколов последовательной передачи данных на базе универсальных асинхронных приемопередатчиков (*UART* — Universal Asynchronous Receiver/Transmitter) и проанализированы вопросы программирования этих устройств в защищенных операционных системах Windows. Во-вторых, на большом числе практических примеров рассмотрены методы программирования последовательного порта в наиболее популярных средах программирования, таких как Delphi и Microsoft Visual Studio .NET последних версий. Программирование последовательного порта в среде .NET и Delphi, благодаря введению новых компонентных моделей для данного интерфейса, стало значительно более эффективным и расширило возможности разработчиков систем обработки и обмена данными. В-третьих, материал книги раскрывает аспекты создания и программирования сетевых систем сбора и обработки данных, на нижнем уровне которых функционирует последовательный интерфейс.

Книга состоит из шести глав, краткое описание каждой из них приведено далее.

- *Глава 1. "Последовательный интерфейс в системах обмена данными".*

  Материал главы посвящен обзору возможностей протоколов последовательного обмена данными и областей их применения. Здесь приводятся и общие характеристики наиболее распространенных стандартов последовательного обмена данными.

- *Глава 2. "Стандарты и протоколы последовательного интерфейса".*

  В главе подробно рассматриваются характеристики популярных протоколов обмена данными RS-232 и RS-485, их аппаратная реализация и программирование на низком уровне. Значительная часть материала посвящена практическим аспектам реализации аппаратных интерфейсов и их программированию на примерах простых систем последовательной передачи данных.

- *Глава 3. "Аппаратная реализация последовательного интерфейса в компьютерных системах".*

  Здесь рассмотрены вопросы аппаратной реализации интерфейса RS-232 в персональных компьютерах. На многочисленных примерах демонстрируются методы программирования стандартных асинхронных приемопередатчиков на низком уровне в операционных системах Windows.

- *Глава 4. "Программирование последовательного интерфейса в операционных системах Windows".*

  В этой главе анализируются основные аспекты программирования последовательного обмена данными с использованием функций интерфейса прикладного программирования операционных систем Windows. Значительная часть материала главы посвящена программированию приложений с графическим интерфейсом пользователя в Microsoft Visual Studio .NET и Delphi.

- *Глава 5. "Программирование последовательного интерфейса в Linux".*

  Материал этой главы посвящен программированию последовательного обмена данными в операционных системах Linux. Приводятся многочисленные примеры программного кода на языке GNU C.

- *Глава 6. "Расширения ввода-вывода последовательного интерфейса".*

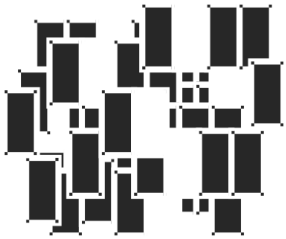
  В главе рассматриваются современные аппаратно-программные технологии программирования приложений для работы с последовательными интерфейсами на основе преобразования RS-232 в *USB* (Universal Serial Bus), а также использование виртуальных последовательных портов. Значительная часть главы посвящена разработке простых систем обработки данных с использованием интернет-технологий с детальным анализом программного кода.

Материал книги будет полезен всем желающим, которые хотели бы существенно пополнить свои знания в области разработки и программирования систем передачи данных на основе последовательных интерфейсов.

Благодарности

Автор выражает огромную благодарность сотрудникам издательства "БХВ-Петербург" за подготовку материалов книги к изданию. Особая признательность жене Юлии за понимание и поддержку при подготовке рукописи.

Глава 1



Последовательный интерфейс в системах обмена данными

Обмен данными между различными устройствами телекоммуникационных и компьютерных систем принципиально можно выполнить методом параллельной или последовательной передачи всех битов данных. При последовательной передаче данные в коммуникационный канал передаются бит за битом, синхронно с сигналами тактовой частоты, которая определяет скорость обмена данными. Последовательная передача данных используется при обмене данными между удаленными системами (компьютерными или телекоммуникационными) и обеспечивает надежную передачу данных на большие расстояния. Для последовательной передачи данных разработаны и очень широко применяются стандарты RS-232, RS-485, SPI, I2C, 1-Wire, которые используются, как правило, в компьютерных системах общего назначения. На базе протоколов RS-232 и RS-485 разработаны и специализированные интерфейсы для обмена данными в промышленных сетях. Наибольшее распространение в таких сетях получили протоколы Profibus, Modbus, Fieldbus и др.

При сравнении систем параллельной передачи данных, в которых все биты данных передаются одновременно, с последовательными системами может показаться, что параллельная передача данных всегда будет выполняться быстрее. Теоретически это верно, но на практике не все так однозначно. Во многих случаях при обмене данными между удаленными системами линии последовательной передачи данных синхронизируются на более высокой частоте, чем это делается в системах параллельной передачи данных, что обеспечивает и более высокую скорость передачи.

Преимуществом систем последовательной передачи данных по сравнению с системами параллельной передачи является то, что в них используется на-

много меньше сигнальных линий, а это упрощает реализацию физического интерфейса и значительно удешевляет стоимость оборудования. Физически последовательный обмен данными можно реализовать намного меньшим количеством сигнальных линий (при параллельной передаче данных потребуется как минимум 8 линий данных плюс сигнал строба, при этом мы не учитываем сигналы квитирования как со стороны передатчика, так и со стороны приемника). Иногда при параллельной передаче пытаются уменьшить количество сигнальных линий, используя две 4-битовые посылки данных при передаче одного байта, но при такой "гибридной" конфигурации возникают дополнительные проблемы.

Хочу отметить, что все внутренние шины обмена данными в самом оборудовании, например, в системах на базе процессоров и микроконтроллеров, реализованы по принципу параллельной передачи, поскольку при небольших внутрисхемных расстояниях параллельная шина при прочих равных условиях будет всегда работать быстрее. Тем не менее, для удаленных систем в настоящее время наиболее оптимальным является обмен данными по последовательным линиям.

Рассмотрим вкратце наиболее распространенные стандарты (интерфейс) последовательной передачи данных и области их применения. Начнем с RS-232.

Интерфейс *RS-232* используется, как правило, для подключения к компьютеру стандартных внешних устройств (принтера, сканера, модема, мыши и др.), а также для связи компьютеров между собой на относительно небольших расстояниях. Широкое применение интерфейс RS-232 находит при настройке и диагностике различного телекоммуникационного и компьютерного оборудования. Все современные системы выпускаются с внешним интерфейсом последовательного порта и прошитым программным обеспечением, которое позволяет настраивать такое оборудование с помощью простых терминальных программ, запущенных, например, на персональном компьютере.

Если сравнивать, например, интерфейс RS-232 и интерфейс параллельного порта *SPP* (Standard Parallel Port) (Centronics) персонального компьютера, то при использовании первого данные можно передавать на значительно бóльшие расстояния, при этом соединительный кабель можно реализовать в трехпроводной конфигурации. Программирование последовательного интерфейса, с другой стороны, несколько сложнее, чем параллельного порта, особенно если выполнять его на уровне аппаратных средств системы. Тем не менее, нужно отметить, что все компьютерные системы общего назначения работают, как правило, под управлением одной из популярных операцион-

ных систем (чаще всего Windows), в которых предусмотрен специальный программный интерфейс для работы с файлами и периферийными устройствами, значительно упрощающий работу с последовательным портом на программном уровне. Например, в операционных системах Windows для работы с последовательным портом можно использовать функции прикладного интерфейса программирования *WIN API* (Windows Application Programming Interface).

Передача данных по интерфейсу RS-232 осуществляется побайтово, причем начало и конец передачи синхронизируются стартовым и стоповым битами. Сам процесс передачи данных можно контролировать как программными, так и аппаратными средствами (это зависит от конфигурации системы). Данные между устройствами могут передаваться как в одном (*полудуплексный режим*), так и в обоих направлениях (*дуплексный режим*). Физическая линия подсоединяется к интерфейсу RS-232 посредством стандартного 25-контактного разъема DB25 (в настоящее время используется редко) или 9-контактного разъема DB9.

Интерфейс RS-232 может соединять только два устройства (конфигурация "точка-точка"), при этом линия передачи одного устройства соединяется с линией приема другого и наоборот. Эта конфигурация используется при полнодуплексном режиме. При работе в полудуплексном режиме передающая линия одного устройства соединяется с приемной линией другого. Различные сигналы квитирования (подтверждения) могут генерироваться как аппаратными средствами (за счет использования дополнительных линий управления), так и программным способом (например, включением в поток передаваемых данных соответствующих управляющих символов).

Вот основные характеристики интерфейса RS-232:

- скорость передачи — до 115 Кбит/с;
- расстояние передачи — до 15 м;
- тип сигнала — несимметричный по напряжению, один передатчик и один приемник;
- тип соединения — полный дуплекс, одноточечное соединение.

Попытки увеличить расстояние, на которое могут передаваться данные, и скорость передачи данных привели к модернизации интерфейса RS-232. В результате появился интерфейс *RS-423*, который мог обеспечивать высокоскоростную передачу данных (до 10 Мбит/с) на расстояние до 1200 м в несимметричной конфигурации, а также ставший наиболее популярным в про-

мышленных системах интерфейс *RS-485*, который обеспечивает передачу симметричного (дифференциального) сигнала.

Интерфейс RS-485 реализует двунаправленную полудуплексную передачу данных и допускает одновременное подключение 32 приемников к шине. На базе интерфейса RS-485 созданы и получили широкое распространение многие промышленные стандарты обмена данными, при этом RS-485 обеспечивает канальный уровень (если рассматривать промышленные шины в виде стека протоколов).

Рассмотрим основные технические характеристики интерфейса RS-485 более подробно. Он является наиболее широко используемым промышленным стандартом, использующим двунаправленную сбалансированную линию передачи. Протокол поддерживает многоточечные соединения, обеспечивая создание сетей с количеством узлов до 32 и передачу данных на расстояние до 1200 м. Использование буферных усилителей сигнала в интерфейсе RS-485 позволяет увеличить расстояние передачи еще на 1200 м или добавить еще 32 узла. Стандарт RS-485 поддерживает полудуплексный режим обмена данными, при этом для передачи и приема данных достаточно одной витой пары проводников.

Интерфейс RS-485 имеет следующие характеристики:

- скорость передачи — до 10 Мбит/с;
- расстояние передачи — до 1200 м;
- тип сигнала — дифференциальное напряжение;
- тип линии передачи — витая пара, до 32 передатчиков и до 32 приемников;
- тип соединения — полудуплекс, многоточечное соединение.

На основе протоколов RS-232, RS-422 и RS-485 разработаны практически все популярные промышленные стандарты. Рассмотрим некоторые из них и начнем с протокола *Modbus*. Этот коммуникационный протокол базируется на архитектуре клиент-сервер и был разработан фирмой *Modicon* для использования в контроллерах с программируемой логикой. Является фактическим стандартом для промышленных сетей передачи данных и широко применяется для организации связи промышленного электронного оборудования.

Для передачи данных в таком протоколе используются последовательные линии связи на базе интерфейсов RS-485, RS-422, RS-232, а также протоколы сетевого обмена *TCP/IP* (Transmission Control Protocol/Internet Protocol). Аппаратно протокол Modbus реализован в виде последовательного интерфейса, совместимого с протоколом RS-232, который интегрирован в стандартные

Modbus-порты в контроллерах Modicon. Сами контроллеры можно соединять между собой напрямую или посредством модемов. Обмен данными между контроллерами происходит по схеме "ведущий-ведомый". Только одно устройство (ведущий) может инициировать обмен данными, в то время как остальные устройства (ведомые) передают ведущему данные или выполняют указанные в запросе ведущего действия.

Как правило, ведущий реализован аппаратно как хост-процессор с набором модулей программирования, а ведомый представляет собой программируемый контроллер. Ведущий может обращаться к отдельному устройству или инициировать широковещательный запрос по шине Modbus. Ведомый, обнаружив в заголовке запроса свой адрес, отвечает ведущему, после чего начинается обмен данными. При широковещательном запросе ответы ведущему не возвращаются. Для передачи данных по протоколу Modbus используется единый простой формат передачи данных PDU, который является частью более широкой спецификации ADU.

Шина Modbus может работать в одном из трех режимов:

- ❒ режим RTU, который используется для передачи данных по последовательным линиям связи с интерфейсами RS-485, RS-422 и RS-232;
- ❒ режим ASCII, в котором также используются интерфейсы RS-485, RS-422 и RS-232;
- ❒ режим сетевой передачи, в котором используется протокол TCP.

На шине Modbus, в которой используются сигнальные линии интерфейсов RS-232 и RS-485, можно адресовать 247 ведомых устройств.

Еще одним протоколом обмена данными, о котором хотелось бы упомянуть, является протокол Profibus, в котором широко используются последовательные интерфейсы RS-485 (версия FMS). Profibus объединяет технологические и функциональные особенности последовательной связи полевого уровня. Он позволяет объединять разрозненные устройства автоматизации в единую систему на уровне датчиков и приводов.

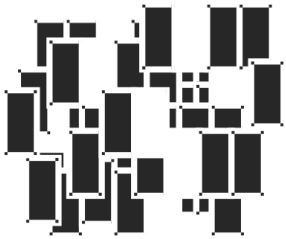
Profibus использует обмен данными между ведущим и ведомыми устройствами (протоколы DP и PA) или между несколькими ведущими устройствами (протоколы FDL и FMS). Требования пользователей к получению открытой, независимой от производителя, системы связи базируется на использовании стандартных протоколов Profibus.

Протокол *Profibus-FMS* разработан для связи ведущих устройств (контроллеров и ПК) друг с другом. Этот протокол используется там, где степень функциональности более важна, чем быстрое время реакции системы, и применяется на высоком уровне. Линия передачи информационных сигналов соответствует стандарту RS-485.

Протокол *Profibus-DP* спроектирован для высокоскоростной передачи данных между ведущим (контроллер) и оконечными устройствами (датчики, первичные преобразователи) сети и применяется на нижнем уровне системы промышленной автоматизации. Передача данных, как и для Profibus-FMS, основана на протоколе RS-485. Скорость передачи прямо зависит от протяженности сегмента сети и может изменяться от 100 Кбит/с для расстояния 1200 м до 12 Мбит/с для расстояния 100 м.

Существуют и другие популярные протоколы обмена данными с использованием последовательных интерфейсов, но далее мы будем рассматривать только те, о которых только что упомянули и которые базируются на интерфейсах RS-232 и RS-485.

Глава 2



Стандарты и протоколы последовательного интерфейса

Обмен данными по последовательным каналам связи регламентируется несколькими стандартами (протоколами, интерфейсами), определяющими аппаратно-программную архитектуру систем последовательной передачи данных. Основным протоколом последовательного обмена данными является RS-232. В соответствии с этим протоколом функционирует множество коммуникационных устройств, включая последовательные, или, по-другому, сериальные порты (serial ports) компьютерных систем. В подавляющем большинстве случаев для физической связи устройств, работающих по этому стандарту, используются 9-контактные разъемы, хотя иногда можно встретить и 25-контактные. Передача-прием данных по RS-232 осуществляется последовательно, бит за битом, откуда, собственно, и произошло название "последовательный порт".

Последовательный порт — один из наиболее ранних интерфейсов, разработанных для использования обмена данными. Последовательные порты компьютерных систем обычно известны под названием COM-портов. Обмен данными через последовательные порты осуществляется с использованием специальных чипов, известных под названием UART (Universal Asynchronous Receiver/Transmitter). Скорость обмена данными по последовательному порту может достигать 115 Кбит/с (или иначе в английском варианте kbps), хотя последние разработки позволяют увеличить эту скорость до 460 Кбит/с.

В настоящее время последовательные порты используются, в основном, для подключения модемов и специальных терминальных устройств. Довольно часто они применяются для настройки и отладки различного оборудования. Более новые скоростные технологии, такие, например, как USB и Fireware, с

каждым днем все больше ограничивают применение последовательных интерфейсов. Тем не менее, последовательные порты все еще широко используются как в уже выпущенном оборудовании, так и в разрабатываемом.

Посмотрим, каким образом осуществляется обмен данными через последовательные порты. Принципиально возможны два способа обмена данными: синхронный и асинхронный.

При *синхронном* режиме передачи данных и передатчик, и приемник должны работать на одинаковой частоте. В начале обмена передатчик посылает приемнику последовательность синхробит, за которой следуют посылки бит. При синхронном обмене прием-передача данных может осуществляться на значительно более высоких скоростях, чем при асинхронном обмене, поскольку не нужно передавать старт-бит, стоп-бит, бит паритета при передаче одного байта. Недостатком синхронного обмена является то, что и передатчик, и приемник должны обладать высокой стабильностью частоты, поскольку даже небольшое рассогласование в частотах приведет к появлению быстро накапливающейся ошибки.

При *асинхронной* передаче каждому байту данных предшествует *старт-бит*, за ним следуют биты данных, после них может передаваться *бит паритета* (четности), и в завершение посылки данных передается *стоп-бит*, гарантирующий определенную выдержку времени между соседними посылками. Структуру посылки можно представить так, как показано на рис. 2.1.

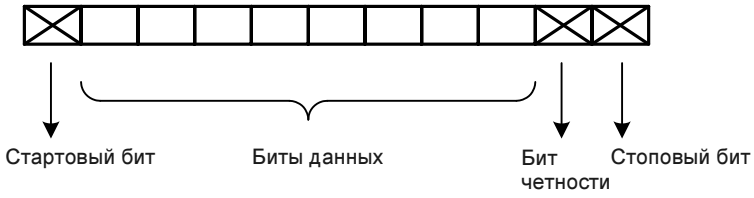


Рис. 2.1. Структура посылки данных

Формат данных последовательного порта обычно представляют в форме записи:

```
количество_бит_данных-тип_четности-количество_стоповых_бит
```

Например, запись 8-N-1 интерпретируется как посылка данных, содержащая 8 бит данных, без контроля четности и с одним стоповым битом.

Запись 7-E-2 интерпретируется как посылка, содержащая 7 бит данных, с контролем четности и с двумя стоповыми битами.

Биты данных чаще всего интерпретируются как биты ASCII-символа. Стартовый бит следующей посылки может передаваться в любой момент времени после окончания стопового бита предыдущей посылки. Количество бит данных может быть равным 5, 6, 7 или 8 битам, а количество стоповых битов может быть 1, 1.5, 2. Бит четности может отсутствовать.

Асинхронный обмен данными может осуществляться на одной из скоростей: 50, 75, 110, 150, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600 и 115200 бит/с. Для обмена данными в асинхронном режиме необходимо установить одинаковые параметры приемника и передатчика по скорости, количеству бит данных, проверке четности и количеству стоповых бит.

В компьютерных системах наиболее широко используется стандарт асинхронного обмена данными RS-232. Он применяется как для обмена данными через последовательные (COM) порты компьютера, так и для связи автономных периферийных устройств между собой. Рассмотрим интерфейс более подробно.

2.1. Интерфейс RS-232

На аппаратном уровне интерфейса RS-232 используются несимметричные приемопередающие устройства, при этом уровень сигнала отсчитывается относительно общего провода (сигнальной земли). Аппаратная часть интерфейса RS-232 не обеспечивает гальванической развязки устройств по цепям питания (в отличие от другого интерфейса, известного под названием *"токовая петля"*).

Логические уровни сигналов интерфейса находятся в диапазонах от –3 до –25 В (логическая "1") и от +3 до +25 В (логический "0"). Диапазон от –3 до +3 В соответствует неактивному состоянию (зона нечувствительности или гистерезис). Обычно сигнал, соответствующий уровню логической "1", называют "пробелом" (Space), а сигнал, соответствующий уровню логического "0" — "маркером" (Mark). Дальность связи при использовании интерфейса RS-232 достигает 15 м, поэтому чаще всего этот протокол используют в лабораторных и учебных системах обмена данными, а также при наладке и тестировании коммуникационного оборудования.

В основе асинхронного обмена данными, который используется при функционировании интерфейса RS-232 (как, впрочем, и многих других стандартов), положен принцип обработки отдельных кадров (*фреймов*) данных, передаваемых между стартовым и стоповым(и) битами.

Интерфейс RS-232 использует несколько сигнальных линий, но для иллюстрации передачи-приема данных мы посмотрим то, что происходит на любой из линий TD или RD. Для этого проанализируем процесс передачи-приема байта данных, который в упрощенном виде можно представить так, как показано на рис. 2.2.

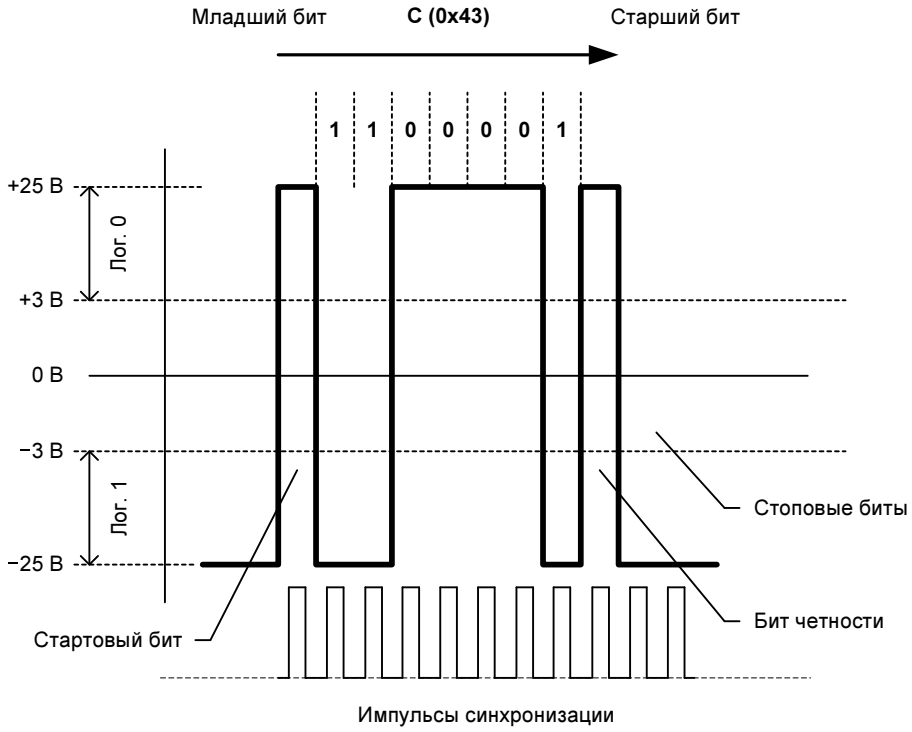


Рис. 2.2. Передача байта данных

На рисунке демонстрируется передача символа C в асинхронном режиме: 7 бит данных, бит четности и два стоповых бита. При этом каждому байту данных предшествует стартовый бит, за которым следуют биты данных (в данном случае 7 бит, представляющих символ C). Первым передается младший бит, последним — старший.

После передачи битов данных передается бит четности (в данном случае равный 0), и завершают передачу байта данных два стоповых бита, гарантирующих определенную выдержку времени между соседними посылками.

Обратите внимание на то, что логические уровни сигналов инверсные по отношению к полярности напряжения сигналов, т. е. отрицательным перепадам напряжения соответствует логическая "1", а положительным — логический "0".

Стартовый бит следующей посылки может передаваться в любой момент времени после окончания стопового бита предыдущей посылки, поэтому между двумя последовательными посылками данных могут быть паузы произвольной длительности.

Синхронизация приемника осуществляется с помощью стартового бита передатчика — при этом приемник и передатчик должны работать на одной и той же тактовой частоте (см. рис. 2.2). По спаду стартового бита передатчика генератор синхроимпульсов приемника сбрасывается и начинает формировать импульсы синхронизации, которые через программируемый делитель частоты и схемы синхронизации управляют приемом данных.

Аппаратно прием и передача данных через интерфейс RS-232 обычно реализуются в многофункциональном контроллере, который сокращенно называют UART. Это общее название для данного типа чипов, хотя выпускается очень много разновидностей этого устройства. Наиболее распространенными контроллерами приема-передачи по последовательному порту являются устройства, совместимые с серией 8250, которая включает следующие чипы: 16450, 16550, 16650, 16750.

Для безошибочного приема каждый бит данных должен захватываться в середине интервала времени, когда он является действительным (см. рис. 2.2). С возрастанием скорости передачи синхронизация передатчика и приемника усложняется — начинают сказываться паразитные емкости электронных узлов, затягивающие фронты синхроимпульсов, а также распределенная индуктивность физической линии передачи.

На практике интерфейс RS-232 реализован в виде сигнальных линий, имеющих следующее назначение:

- SG (Signal Ground) — сигнальная земля (относительно нее отсчитываются уровни сигналов);
- TD (Transmit Data) — выход передатчика (TXD);
- RD (Receive Data) — вход приемника (RXD);
- RTS (Request-To-Send) — запрос обмена данными, который выставляется контроллером последовательного порта (UART);

- CTS (Clear-To-Send) — указывает на готовность модема или терминального устройства к обмену данными;
- DTR (Data Terminal Ready) — указывает модему или другому устройству, что контроллер последовательного порта (UART) готов установить соединение;
- DSR (Data Set Ready) — указывает контроллеру (UART) последовательного порта на то, что периферийное устройство готово установить соединение;
- DCD (Data Carrier Detect) — входной сигнал от удаленного модема или устройства. Когда модем или другое устройство обнаруживает появление несущей на другом конце линии, то линия становится активной;
- RI (Ring Indicator) — вход индикатора вызова.

Сигналы RD и TD, определяющие прием-передачу данных, мы рассмотрели на примере передачи-приема байта данных. Обратите внимание на то, что линии данных являются последовательными, в то время как сигнальные линии RTS, CTS, DCD, DSR, DTR, RTS и RI являются, по сути, параллельными.

Для соединения устройств, работающих с использованием данного интерфейса, применяется, как уже упоминалось, 9-контактный разъем, выводы которого соответствуют сигналам, указанным в табл. 2.1.

Таблица 2.1. *Сигналы и выводы интерфейса RS-232 для 9-контактного разъема*

Номер вывода на разъеме	Сигнал	Номер вывода на разъеме	Сигнал
3	TD	5	SG
2	RD	1	DCD
7	RTS	4	DTR
8	CTS	9	RI
6	DSR		

Эти же сигналы присутствуют и на COM-порту компьютера, к которому подсоединено какое-либо устройство, например, модем.

Для соединения двух устройств по последовательному каналу используются так называемые *"нуль-модемные" соединения*. В зависимости от аппаратно-программной конфигурации системы обмена данными может использоваться как полнофункциональное соединение (рис. 2.3, *а*), так и максимально упрощенный вариант (рис. 2.3, *б*), который довольно часто используется как для диагностических целей, так и для управления системами промышленной автоматики на базе специализированных модулей, например, микроконтроллеров.

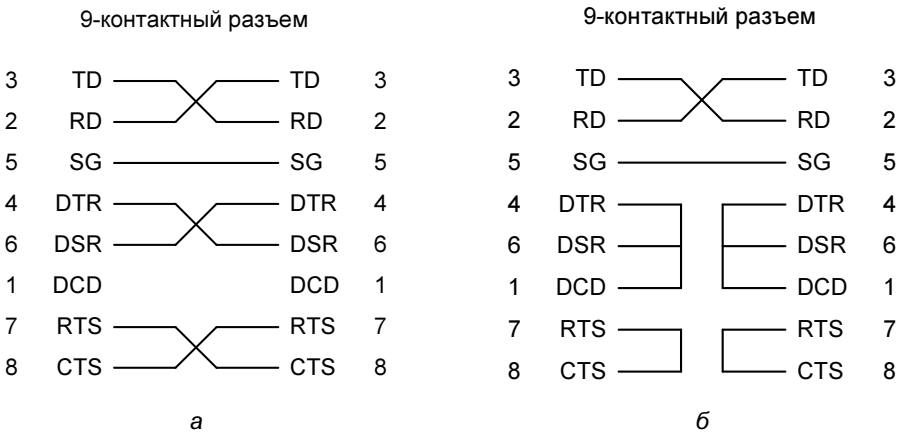


Рис. 2.3. Нуль-модемное соединение:
а — полнофункциональное соединение, *б* — упрощенный вариант

Еще один, весьма распространенный случай, когда разработчик должен проверить функционирование приложений, работающих с последовательными портами. Для быстрой проверки таких программ можно воспользоваться так называемым *интерфейсом обратной связи* (loopback interface), схема которого приведена на рис. 2.4.

Сигналы интерфейса RS-232 позволяют управлять потоком данных в зависимости от состояния передатчика и приемника. При этом могут использоваться несколько протоколов обмена данными. Хочу сразу же уточнить, что эти протоколы не касаются базовых принципов передачи данных, рассмотренных ранее, их назначение иное — они позволяют синхронизировать устройства, имеющие различные скорости приема-передачи данных, различные размеры буферов памяти для хранения данных. Например, принтер, получающий данные через последовательный порт, может откладывать их

прием до окончания обработки предыдущей порции данных, сигнализируя об этом передатчику.

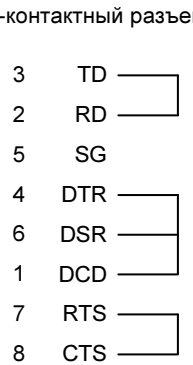


Рис. 2.4. Схема интерфейса обратной связи

Для управления потоком данных используются аппаратный или программный протоколы. Рассмотрим некоторые, наиболее часто используемые протоколы управления обменом данными:

- аппаратный протокол RTS/CTS — используется для соединения принтера с компьютерной системой или для соединения компьютеров, при этом нуль-модемный кабель использовать нельзя;
- аппаратный протокол DTR/DSR — аналогичен RTS/CTS, но использует другие сигналы;
- программный протокол XON/XOFF — используется в двунаправленных каналах обмена при буферизованном обмене данными. При полном заполнении буфера приемника передача данных прекращается и ее возобновление возможно после обработки данных в буфере приемника. Приемник останавливает передачу данных передатчиком, посылая ему команду `XOFF`, и возобновляет ее посылкой команды `XON`;
- программный протокол ACK — синхронизация получения одного байта данных или группы байтов. Приемник посылает передатчику команду `ACK`, в ответ на которую передатчик посылает приемнику байт (или пакет байтов).

Рассмотрим примеры аппаратно-программной реализации протокола RS-232 в компьютерных системах обмена данными.

2.2. Примеры аппаратно-программной реализации протокола RS-232

К настоящему времени разработано и используется множество различных аппаратно-программных конфигураций интерфейса RS-232. Рассмотрим наиболее распространенные из них. Первая задача, которую приходится решать разработчикам интерфейса — согласование уровней напряжения логических сигналов. Уровень напряжения логической "1" по спецификации протокола RS-232 должен находиться в пределах от –3 до –25 В, а уровень логического "0" — в диапазоне от +3 до +25 В, при этом уровни напряжений в диапазоне от –3 до +3 В являются неопределенными. Внутренние цифровые схемы большинства устройств работают с уровнями напряжений от 1,8 до 5 В, поэтому для передачи сигналов интерфейса RS-232 в линию необходимо использовать либо отдельный, либо интегрированный в устройство преобразователь уровня сигналов. Например, микросхема асинхронного приемопередатчика (UART) персонального компьютера работает с уровнями сигналов, соответствующих TTL-логике, что требует использования буферного приемопередатчика.

Наибольшее распространение в аппаратных интерфейсах RS-232 получил буферный приемопередатчик сигналов интерфейса MAX232. Эта микросхема была разработана относительно давно, но в настоящее время все еще очень широко используется при разработке интерфейса RS-232 и считается промышленным стандартом для подобного типа устройств. Микросхема MAX232 позволяет связать различные устройства по последовательному каналу (рис. 2.5).

Здесь продемонстрирована организация канала связи между последовательным портом персонального компьютера и внешним устройством на базе микроконтроллера/микропроцессора при помощи MAX232.

Кристалл MAX232 включает внутренний удвоитель-инвертор напряжения на переключаемых конденсаторах с подкачкой заряда (они подключаются к выводам C1+, C1–, C2+ и C2–). Это позволяет получить +10 В и –10 В из одного источника питания +5 В. Сигналы с уровнями ±10 В используются для обмена данными по интерфейсу RS-232. В одном корпусе микросхема содержит два буферных приемника и два передатчика сигнала. В настоящее время используются и более совершенные версии этого кристалла, причем некоторые из микросхем, например, DS275, не содержат навесных компонентов. Что же касается емкости конденсаторов, то для некоторых модификаций MAX232

можно использовать и конденсаторы меньшей емкости, например, 1 мкФ. В наших последующих примерах мы будем использовать стандартную микросхему MAX232.

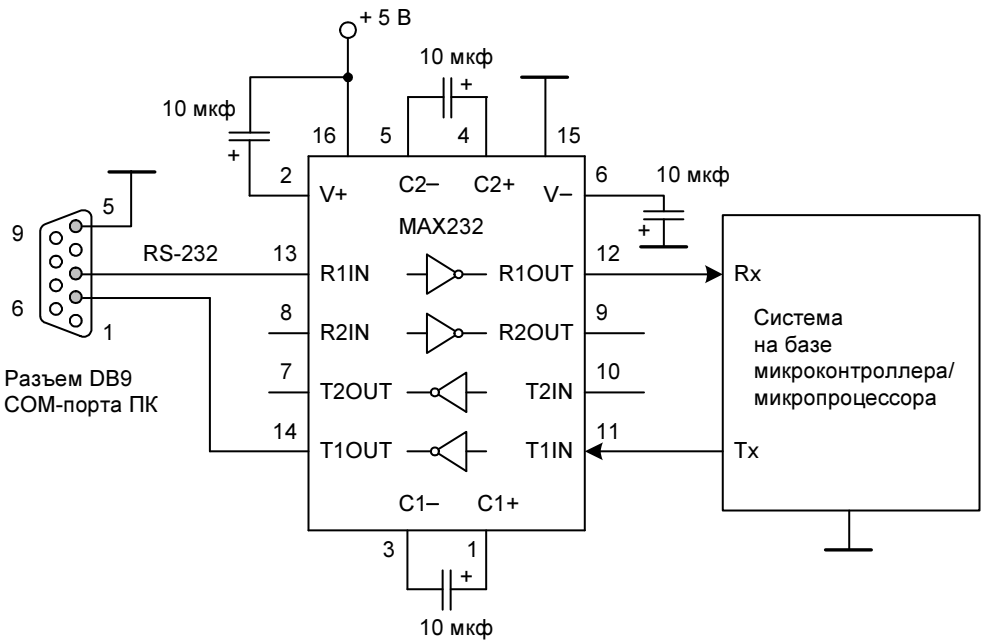


Рис. 2.5. Реализация интерфейса на базе RS-232 между двумя устройствами

Для передачи данных по последовательному интерфейсу в кристаллы большинства промышленных микроконтроллеров или микропроцессоров интегрируется модуль последовательного интерфейса. Сигналы этого интерфейса имеют уровни TTL-логики, а количество сигнальных линий и линий аппаратного контроля может варьироваться в зависимости от круга возможных задач, для которых предназначено устройство. Например, в большинстве микроконтроллеров серии 8051/8052 имеется, как правило, один стандартный интерфейс последовательного обмена, в котором используются только две сигнальные линии — TxD и RxD. В этом случае обеспечивается программный контроль потока данных, чего вполне достаточно для простых систем обмена данными. В микроконтроллерах PIC24 фирмы Microchip на кристалле может находиться несколько модулей последовательной передачи данных, работающих независимо друг от друга, причем для обмена данными могут

использоваться и линии аппаратного управления потоком данных. Кроме того, наличие нескольких независимых модулей последовательного обмена данными позволяет организовать работу в "чистом" полнодуплексном режиме с отдельным управлением потоком данных по двум каналам.

При небольших расстояниях (до 1,5—2 м) и небольших скоростях два подобных устройства могут обмениваться данными без использования буферных формирователей, однако при увеличении расстояния и скорости обмена без драйверов интерфейса такие системы будут работать с ошибками.

Далее мы рассмотрим различные демонстрационные проекты обмена данными по интерфейсу RS-232, в которых будут проанализированы схемы обмена данными на уровне сигналов протокола RS-232. В подавляющем большинстве современных приложений интерфейс RS-232 используется, как правило, в лабораторных системах управления и контроля. Такие системы состоят из двух частей: компьютерной системы (обычно это персональный компьютер) и внешнего устройства управления и контроля на базе специализированной системы с микроконтроллером или микропроцессором. В качестве процессорного модуля таких внешних систем очень часто используются микроконтроллеры серий 8051/8052 или PIC-микроконтроллеры.

В большинстве демонстрационных проектов этой главы будут применяться микроконтроллеры 8051/8052. Такой выбор обусловлен несколькими причинами. Во-первых, во всех проектах будут задействованы те или иные сигнальные линии интерфейса RS-232, функционирование которых проще всего показать с помощью аппаратно-программной системы на микроконтроллере. Во-вторых, устройства 8052 очень широко распространены в промышленных и учебных системах управления и контроля, имеют интуитивно понятную систему команд и очень легко программируются для последовательного обмена данными на языке C.

И, наконец, интерфейс последовательного обмена микроконтроллеров включает две базовые линии, TxD и RxD, для передачи и приема данных, к которым легко подключаются популярные драйверы линии, такие, как MAX232. В отдельных проектах мы будем использовать также и мощные 16-битовые микроконтроллеры PIC24.

Пример простейшей системы обмена данными по интерфейсу RS-232 между двумя микроконтроллерами 8052 показан на рис. 2.6.

Здесь используется нуль-модемная конфигурация, в которой задействованы две сигнальные линии передачи и приема данных. При такой конфигурации выход TxD одного устройства подключается ко входу RxD другого устройства, и наоборот, вход RxD одного устройства подключается к выходу TxD

другого устройства. Кроме того, общие провода обоих устройств должны быть соединены вместе. Для согласования TTL-уровней сигналов микроконтроллеров DD1—DD2 с уровнями интерфейса RS-232 используются два стандартных драйвера линии MAX232. В этой схеме отсутствует аппаратный и программный контроль обмена данными, поэтому подобную конфигурацию можно использовать лишь в диагностических целях или при работе устройств, в которых не возникнет "зависание" оборудования при приеме или передаче данных.

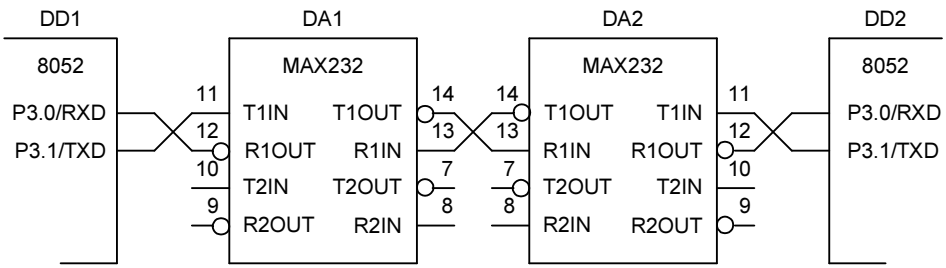


Рис. 2.6. Система обмена данными по интерфейсу RS-232 между микроконтроллерами

Программное обеспечение для обмена данными по такой схеме определяется конкретными особенностями работы системы. Предположим, что микроконтроллер DD1 является передатчиком данных, а микроконтроллер DD2 — их приемником. Программа для передатчика на языке Keil C51 будет включать следующий код, представленный в листинге 2.1.

Листинг 2.1. Программа передачи строки данных микроконтроллеру по интерфейсу RS-232

```
#include <REG52.H>
#include <stdio.h>

void main(void)
{
 char *s1 = "Test String for RS-232 Receiver!";

 SCON = 0x50;
 TMOD |= 0x20;
```

```
 TH1 = 0x0FD;
 TR1 = 1;
 TI = 1;

 printf(s1);
 while(1);
 }
```

Программа, записанная в устройство DD1, посылает строку данных микроконтроллеру DD2. В передатчике последовательного порта задействована всего одна сигнальная линия TXD, по которой данные передаются на вход передачи T1IN драйвера MAX232 интерфейса RS-232 (DA1 на схеме рис. 2.6) и далее через буфер приемника DA2 на вход RXD микроконтроллера 8052 (DD2). Скорость передачи и приема данных устанавливается равной 9600 бод.

Операторы

```
SCON = 0x50;
TMOD |= 0x20;
TH1 = 0x0FD;
TR1 = 1;
TI = 1;
```

позволяют настроить скорость передачи данных 9600 бод при стандартной тактовой частоте микроконтроллера 11,059 МГц.

Далее приводится исходный текст программы приемника (микроконтроллер DD2).

Листинг 2.2. Программа приема строки данных микроконтроллером по интерфейсу RS-232

```
#include <REG52.H>
#include <stdio.h>

void main(void)
{
 char buf[64];
 char *pbuf = buf;
```

```
 SCON = 0x50;
 TMOD |= 0x20;
 TH1 = 0x0FD;
 TR1 = 1;
 TI = 1;

 while (1)
  {
   *pbuf = getchar();
   if (*pbuf == 0)
     break;
   pbuf++;
  }
  printf("%s", buf);
}
```

Программа побайтово принимает строку данных из последовательного порта в цикле `while (1)` с помощью функции `getchar`. Принятые байты записываются в буфер `buf`, а прием байтов выполняется до обнаружения в потоке данных нулевого символа. По окончании приема принятая строка отправляется обратно устройству-передатчику. Группа операторов

```
SCON = 0x50;
TMOD |= 0x20;
TH1 = 0x0FD;
TR1 = 1;
TI = 1;
```

выполняет ту же функцию, что и в передатчике — она используется для настройки скорости приема 9600 бод для стандартной тактовой частоты микроконтроллера 11,059 МГц. Скорость приема должна совпадать со скоростью передачи в передатчике данных.

В роли передатчика данных может выступать последовательный порт персонального компьютера. В этом случае получим простую систему управления на базе протокола RS-232. В такой системе микроконтроллер будет получать данные от программы, работающей под управлением какой-либо операционной системы, и выполнять соответствующий алгоритм их обработки. Очень часто данные с последовательного порта персонального компьютера переда-

ются во внешнее устройство по интерфейсу RS-232, где используются для управления исполнительными устройствами. Пример такой системы управления на базе популярного микроконтроллера 8051 показан на рис. 2.7.

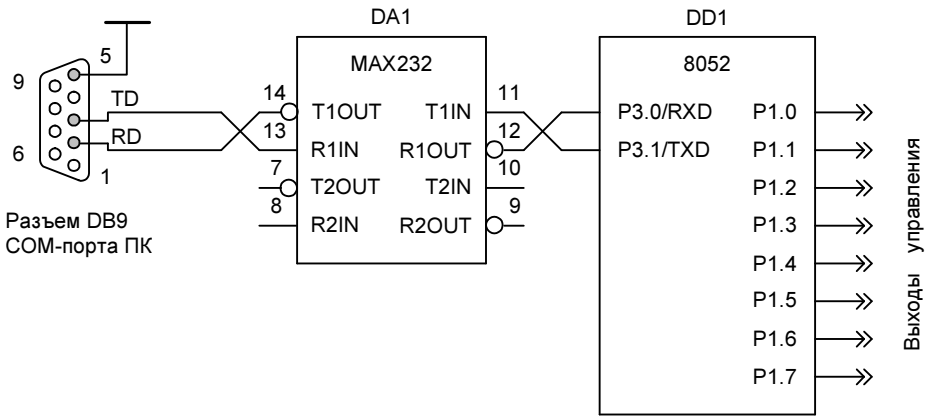


Рис. 2.7. Система управления на базе интерфейса RS-232

В этой системе данные, полученные по последовательному порту, выводятся в порт P1 микроконтроллера и могут управлять различными нагрузками. Исходный текст программы управления, записанной в микроконтроллере 8052, показан в листинге 2.3.

Листинг 2.3. Программа управления внешними устройствами по интерфейсу RS-232

```
#include <REG52.H>
#include <stdio.h>

void SerialISR(void) interrupt 4
{
 if (RI)
  {
   RI = 0;
   P1 = SBUF;
  }
}
```