

Павел Агуров



Последовательные интерфейсы ПК

Практика программирования



+ CD-ROM

- Протоколы последовательной связи
- Стандарты RS232/RS485
- Работа в DOS, Windows 98/ME/NT/2000/XP
- Примеры на языке Pascal
- Все уровни доступа
- Поддержка Plug and Play



МАСТЕР ПРОГРАММ

Павел Агуров

Последовательные интерфейсы ПК

Практика программирования

Санкт-Петербург

«БХВ-Петербург»

2004

УДК 681.3.06
ББК 32.973-018
А23

Агуров П. В.

А23 Последовательные интерфейсы ПК. Практика программирования. — СПб.: БХВ-Петербург, 2004. — 496 с.: ил.

ISBN 5-94157-468-1

Книга представляет собой практическое руководство по программированию последовательных интерфейсов. В первой части книги представлены теоретические сведения о последовательных интерфейсах, во второй — практические примеры и листинги программ на языках Pascal и Delphi, а третья содержит справочные данные, облегчающие поиск необходимой информации. В приложениях приведены дополнительные сведения и ответы на часто задаваемые вопросы. Большое количество практических советов, примеров программ, а также последовательность и простота изложения позволяют читателю уверенно овладеть изложенным в книге материалом. Для удобства читателей все исходные коды приводятся на прилагаемом к книге компакт-диске.

Для программистов, занятых в сфере промышленной автоматизации

УДК 681.3.06
ББК 32.973-018

Группа подготовки издания:

Главный редактор	<i>Екатерина Кондукова</i>
Зам. главного редактора	<i>Евгений Рыбаков</i>
Зав. редакцией	<i>Григорий Добин</i>
Редактор	<i>Алексей Семенов</i>
Компьютерная верстка	<i>Натали Смирновой</i>
Корректор	<i>Наталия Першакова</i>
Оформление серии	<i>Via Design</i>
Дизайн обложки	<i>Игоря Цырульников</i>
Зав. производством	<i>Николай Тверских</i>

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 13.02.04.
Формат 70×100^{1/16}. Печать офсетная. Усл. печ. л. 39,99.
Тираж 3000 экз. Заказ №

"БХВ-Петербург", 190005, Санкт-Петербург, Измайловский пр., 29.

Гигиеническое заключение на продукцию, товар № 77.99.02.953.Д.001537.03.02 от 13.03.2002 г. выдано Департаментом ГСЭН Минздрава России.

Отпечатано с готовых диапозитивов
в Академической типографии "Наука" РАН
199034, Санкт-Петербург, 9 линия, 12.

ISBN 5-94157-468-1

© Агуров П. В., 2004
© Оформление, издательство "БХВ-Петербург", 2004

Содержание

Введение.....	1
Для кого эта книга	1
Структура книги	2
Краткое описание глав	3
Программные требования	6
О программном коде.....	6
Обозначения	7
Аппаратные требования.....	7
Благодарности.....	7
Обратная связь.....	7
ЧАСТЬ I. ПРОТОКОЛЫ И ИНТЕРФЕЙСЫ.....	9
Глава 1. Стандарты последовательной связи	11
1.1. Стандарты последовательной связи	11
1.1.1. Протокол RS-232	13
1.1.2. Протокол RS-422A.....	15
1.1.3. Протокол RS-423A.....	15
1.1.4. Протокол RS-485	15
1.1.5. Протокол RS-499	16
1.1.6. Протокол RS-562	16
1.1.7. Протокол V.24.....	17
1.1.8. Протокол V.28.....	17
1.1.9. Протокол V.35.....	17
1.1.10. Протокол X.21.....	18
1.1.11. Рекомендация X.21bis.....	18
1.1.12. Краткое сравнение RS-протоколов	18
1.2. Принципы последовательной связи.....	19
1.3. Разъемы коммуникационного порта.....	20
1.3.1. Сигнальная "земля" (AB/SG).....	22
1.3.2. Защитная "земля" (AA).....	22
1.3.3. Передаваемые данные (BA/TxD/TD)	23
1.3.4. Принимаемые данные (BB/RxD/RD)	23
1.3.5. Запрос передачи (CA/RTS).....	23

1.3.6. Готовность к передаче (CB/CTS).....	24
1.3.7. Готовность DCE (CC/DSR).....	25
1.3.8. Готовность DTE (CD/DTR)	25
1.3.9. Индикатор вызова (CE/RI)	25
1.3.10. Обнаружение несущей (CF/DCD).....	25
1.3.11. Детектор качества сигнала (CG/SQ)	26
1.3.12. Переключатель скорости передачи данных от DTE (CH)	26
1.3.13. Переключатель скорости передачи данных от DCE (CI)	26
1.3.14. Готовность к приему (CJ).....	26
1.3.15. Местный шлейф (LL).....	27
1.3.16. Удаленный шлейф (RL).....	27
1.3.17. Индикатор тестирования (TM)	27
1.3.18. Синхронизация передачи от DTE (DA)	28
1.3.19. Синхронизация передачи от DCE (DB/TC).....	28
1.3.20. Синхронизация приема от DCE (DD/RC)	28
1.3.21. Передаваемые данные дополнительного канала (SBA/STD).....	28
1.3.22. Принимаемые данные дополнительного канала (SBB/SRD)	28
1.3.23. Запрос передачи по дополнительному каналу (SCA/SRTS).....	29
1.3.24. Готовность к передаче по дополнительному каналу (SCB/SCTS)	29
1.3.25. Обнаружение несущей дополнительного канала (SCF/SDCD)	29
1.4. Ресурсы IBM PC для последовательной связи	29
1.4.1. Сервисы BIOS	29
1.4.2. Коммуникационные порты	30
1.4.3. Использование прерываний	30
1.4.4. Прямое программирование портов в Windows.....	30
1.4.5. Функции Windows	31
Глава 2. Протоколы	32
2.1. Что такое протокол обмена.....	32
2.2. ASCII-протокол передачи данных.....	33
2.3. Бинарный протокол передачи данных.....	35
2.4. Предотвращение потери данных	35
2.4.1. Прерывания и потоки	35
2.4.2. Буферизация.....	37
Синхронизация в DOS.....	38
Синхронизация в Windows	38
2.4.3. Обратная связь	40
2.5. Методы обнаружения ошибок передачи	40
2.5.1. Контроль четности	41
2.5.2. Контрольная сумма	42
Простая контрольная сумма.....	42
Контрольная сумма LRC	42
Контрольная сумма CRC16	43

2.5.3. Стартовый байт.....	44
2.5.4. Другие способы повышения достоверности данных.....	45
Глава 3. Последовательные протоколы IBM PC.....	46
3.1. Мышь.....	46
3.1.1. Базовый протокол Microsoft Mouse.....	46
3.1.2. Протокол Microsoft Plus.....	47
3.1.3. Протокол 3D Serial Mouse.....	47
3.1.4. Протокол PC Mouse.....	48
3.1.5. Совместимость протоколов.....	48
3.2. Модем.....	48
3.2.1. Команды управления модемом.....	49
3.2.2. Регистры модемов.....	50
3.2.3. Протоколы передачи файлов.....	50
ASCII-протокол.....	50
Протокол XModem.....	51
Протокол XModem-CRC.....	52
Протокол XModem-1K.....	52
Протокол YModem.....	53
Протокол YModem-G.....	54
Протокол ZModem.....	54
Протокол ViModem.....	55
Протокол Kermit.....	55
Глава 4. Промышленные последовательные протоколы.....	56
4.1. Протокол MODBUS.....	56
4.1.1. Протокол MODBUS-ASCII.....	57
4.1.2. Протокол MODBUS-RTU.....	58
4.1.3. Поля MODBUS протокола.....	59
Поле "Адрес абонента".....	59
Поле "Код функции".....	60
Поле "Данные".....	60
Поле "Контрольная сумма".....	60
4.2. Протокол CAN.....	61
4.2.1. Характеристики протокола CAN.....	62
4.2.2. Обмен данными в протоколе CAN.....	63
4.2.3. Обнаружение ошибок в протоколе CAN.....	63
4.3. Протокол Profibus.....	64
Глава 5. COM-порты и Plug and Play.....	66
5.1. Кратко о Plug and Play.....	66
5.2. Запуск процедуры PnP.....	67
5.3. Если устройство не находится автоматически.....	68

5.4. Где хранится информация о найденных устройствах	69
5.4.1. Структура реестра Windows 98	69
5.4.2. Структура реестра Windows 2000	71
5.5. Алгоритм Plug and Play для COM-портов	73
5.5.1. Инициализация порта (<i>Port initialization</i>)	73
5.5.2. Обнаружение устройств (<i>Check for device</i>)	73
5.5.3. Установка устройства, фаза 1 (<i>COM port Setup-1</i>)	74
5.5.4. Ожидание ответа, фаза 1 (<i>Wait for response-1</i>)	74
5.5.5. Установка устройства, фаза 2 (<i>COM port Setup-2</i>)	74
5.5.6. Ожидание ответа, фаза 2 (<i>Wait for response-2</i>)	75
5.5.7. Получение идентификатора (<i>Collect device ID</i>)	75
5.5.8. Проверка отключения (<i>Verify Disconnect</i>)	76
5.5.9. Дежурное состояние (<i>Connect Idle</i>)	76
5.5.10. Состояние ожидания отключения (<i>Disconnect Idle</i>)	76
5.6. Формат данных для передачи PnP-идентификатора	77
5.7. Передача PnP-идентификатора	77
5.8. Поля PnP-идентификатора	78
5.8.1. Поле <i>Other ID</i>	80
5.8.2. Поле <i>Begin PnP</i>	80
5.8.3. Поле <i>PnP Rev</i>	80
5.8.4. Поле <i>EISA ID</i>	80
5.8.5. Поле <i>Product ID</i>	81
5.8.6. Поле <i>Extend</i>	81
5.8.7. Поле <i>Serial Number</i>	81
5.8.8. Поле <i>Class Name</i>	81
5.8.9. Поле <i>Device ID</i>	82
5.8.10. Поле <i>User Name</i>	82
5.8.11. Поле <i>Checksum</i>	82
5.8.12. Поле <i>End PnP</i>	82
5.8.13. Пример PnP-идентификатора	83
5.9. INF-файл и его структура	84
5.9.1. Секция <i>Version</i>	84
5.9.2. Секция <i>Manufacturer</i>	85
5.9.3. Секция <i>Destination Dirs</i>	88
Ключ <i>DefaultDescDir</i>	88
Ключи <i>file-list-section</i>	88
Ключ <i>dirid</i>	88
Ключ <i>subdir</i>	89
5.9.4. Секция описания модели	90
5.9.5. Секции <i>xxx.AddReg</i> и <i>xxx.DelReg</i>	91
5.9.6. Секция <i>xxx.Log Config</i>	92
5.9.7. Секция <i>xxx.CopyFiles</i>	93
5.9.8. Секция <i>Strings</i>	94
5.9.9. Связи секций	95

Часть II. ПРАКТИКА ПРОГРАММИРОВАНИЯ	97
Глава 6. Использование сервисов BIOS	99
6.1. Подготовка	99
6.2. Первая программа последовательной связи	100
6.3. Другие функции BIOS	102
Глава 7. Прямое программирование портов	104
7.1. Коммуникационные порты	104
7.2. Программа определения наличия COM-портов	105
7.3. Обходим ограничения BIOS	106
7.4. Чтение и запись с помощью модуля <i>RS232DOS</i>	111
Глава 8. Использование обработки прерываний	113
8.1. Прерывания	113
8.2. Модуль <i>RS232Int</i> для работы с прерываниями	115
8.3. Последовательный обмен с помощью прерываний	124
Глава 9. Переход в Windows	127
9.1. Переходим из DOS в Windows	127
9.2. Первая программа для Windows	131
9.3. Перенос программ из Windows 9x в Windows NT/2000	142
9.3.1. Получение доступа к портам в Windows 2000/XP	143
9.3.2. Расширение возможностей GiveIO	148
9.3.3. Работа с драйвером GiveIOEx	157
9.3.4. Еще немного о прямом доступе к портам	165
Глава 10. Использование функций Windows	166
10.1. Обзор функций Windows для работы с последовательными портами	166
10.2. Специальная настройка порта	167
10.3. Получение состояния линий модема	168
10.4. Используем функции Windows	168
10.5. Несколько замечаний о контроле четности	183
Глава 11. Использование потоков	185
11.1. Преимущества потоков	185
11.2. Создание потока опроса порта	185
Глава 12. Функции асинхронного доступа и события	194
12.1. Асинхронный доступ	194
12.2. События последовательного порта	197
12.3. Использование событий	197

Глава 13. Специальные коммуникационные функции	201
13.1. Выполнение дополнительных операций.....	201
13.2. Прямое управление драйвером	203
13.3. Обнаружение всех портов системы	222
13.4. Имена портов.....	225
13.5. APC-функции Windows 2000/XP	231
Глава 14. Реализация протоколов обмена	234
14.1. Прием и передача простого пакета	234
14.2. Создание компонента	241
14.3. Реализация ASCII-протокола.....	254
14.4. Реализация бинарного протокола.....	262
Глава 15. Вычисление контрольных сумм	273
15.1. Вычисление простой контрольной суммы	273
15.2. Вычисление LCR.....	274
15.3. Вычисление CRC16.....	275
15.4. Вычисление CRC32.....	279
Глава 16. Интерфейс RS-485	282
16.1. Стандарт RS-485	282
16.2. Как работает COM-порт IBM PC.....	283
16.3. Реализация RS-485	288
Глава 17. Работа Plug and Play	291
17.1. Эмулятор устройства.....	291
17.2. Установка драйвера устройства.....	300
17.3. Обнаружение изменений	303
Часть III. СПРАВОЧНИК	309
Глава 18. Сервис BIOS INT14H/INT21H	311
18.1. Сервис BIOS INT14H.....	311
18.1.1. Функция 00: инициализация порта	312
18.1.2. Функция 01: посылка символа в порт.....	313
18.1.3. Функция 02: получение символа	313
18.1.4. Функция 03: получить статус порта.....	313
18.1.5. Функция 04: расширенная инициализация (System/2)	314
18.1.6. Функция 05: расширенное управление портом (System/2).....	315
18.2. Сервис BIOS INT21H.....	316
18.2.1. Функция 03H: вспомогательный ввод.....	317
18.2.2. Функция 04H: вспомогательный вывод	317

18.2.3. Функция 3FH: чтение данных через описатель	317
18.2.4. Функция 40H: запись данных через описатель	318
Глава 19. Порты IBM PC	319
19.1. Порт 3F8H (BasePort+0): данные и делитель	320
19.2. Порт 3F9H (BasePort+1): разрешение прерываний и делитель	320
19.3. Порт 3FAH (BasePort+2): идентификация прерываний	321
19.4. Порт 3FBH (BasePort+3): управление линией	322
19.5. Порт 3FCH (BasePort+4): управление модемом	323
19.6. Порт 3FDH (BasePort+5): статус линии	324
19.7. Порт 3FEH (BasePort+6): статус модема	325
19.8. Порт 3FFH (BasePort+7): заглушка	325
19.9. Скорость передачи	325
19.10. Порты контроллера прерываний	326
Глава 20. Структуры Windows для работы с COM-портами	328
20.1. Структура настроек порта <i>COMMCONFIG</i>	328
20.2. Структура <i>COMMPROP</i>	330
20.3. Структура тайм-аутов <i>COMMTIMEOUTS</i>	336
20.4. Структура статуса порта <i>COMSTAT</i>	339
20.5. Структура <i>DCB</i>	340
Глава 21. Функции Windows для работы с COM-портами	347
21.1. Функции <i>CreateFile</i> и <i>CloseHandle</i> : открытие и закрытие порта	347
21.1.1. Дополнительные сведения	349
21.1.2. Возвращаемое значение	349
21.1.3. Пример вызова	349
21.2. Функция <i>ReadFile</i> : чтение данных из порта	350
21.2.1. Дополнительные сведения	351
21.2.2. Возвращаемое значение	351
21.2.3. Пример вызова	351
21.3. Функция <i>WriteFile</i> : передача данных	352
21.3.1. Дополнительные сведения	353
21.3.2. Возвращаемое значение	353
21.3.3. Пример вызова	353
21.4. Функция <i>ReadFileEx</i> : APC-чтение данных	354
21.4.1. Возвращаемое значение	356
21.4.2. Дополнительные сведения	356
21.4.3. Пример вызова	356
21.5. Функция <i>WriteFileEx</i> : APC-передача данных	357
21.5.1. Возвращаемое значение	358
21.5.2. Пример вызова	358

21.6. Функция <i>BuildCommDCB</i> : создание структуры <i>DCB</i> из строки.....	359
21.6.1. Дополнительные сведения.....	360
21.6.2. Возвращаемое значение	360
21.6.3. Пример вызова.....	360
21.7. Функция <i>BuildCommDCBAndTimeouts</i> : создание структуры <i>DCB</i> и тайм-аутов из строки.....	361
21.8. Функции <i>SetCommBreak</i> и <i>ClearCommBreak</i> : управление выводом данных	362
21.8.1. Возвращаемое значение	362
21.9. Функция <i>ClearCommError</i> : получение и сброс ошибок порта	363
21.9.1. Возвращаемое значение	364
21.10. Функция <i>CancelIo</i> : прерывание асинхронных операций.....	364
21.10.1. Возвращаемое значение	364
21.11. Функция <i>EscapeCommFunction</i> : управление портом	365
21.11.1. Возвращаемое значение	366
21.12. Функции <i>GetCommMask</i> и <i>SetCommMask</i> : маска вызова событий	366
21.12.1. Возвращаемое значение	367
21.12.2. Пример вызова.....	367
21.13. Функция <i>CreateEvent</i> : создание объекта ожидания.....	367
21.13.1. Возвращаемое значение	368
21.13.2. Пример вызова.....	368
21.14. Функция <i>WaitCommEvent</i> : ожидание события СОМ-порта	368
21.14.1. Возвращаемое значение	369
21.14.2. Дополнительные сведения.....	369
21.14.3. Пример вызова.....	369
21.15. Функции <i>GetCommConfig</i> и <i>SetCommConfig</i> : конфигурирование параметров порта.....	372
21.15.1. Возвращаемое значение	372
21.15.2. Пример вызова.....	373
21.16. Функция <i>CommConfigDialog</i> : диалог конфигурирования порта	374
21.16.1. Возвращаемое значение	375
21.16.2. Дополнительные сведения.....	375
21.16.3. Пример вызова.....	375
21.17. Функция <i>GetCommProperties</i> : прочитать свойства порта	375
21.17.1. Возвращаемое значение	376
21.17.2. Пример вызова.....	376
21.18. Функции <i>GetCommState</i> и <i>SetCommState</i> : состояние порта	376
21.18.1. Возвращаемое значение	377
21.18.2. Пример вызова.....	377
21.19. Функции <i>GetCommTimeouts</i> и <i>SetCommTimeouts</i> : тайм-ауты порта.....	377
21.19.1. Возвращаемое значение	378
21.19.2. Пример вызова.....	378

21.20. Функция <i>PurgeComm</i> : сброс буферов порта	378
21.20.1. Возвращаемое значение	379
21.20.2. Пример вызова	379
21.21. Функция <i>SetupComm</i> : конфигурирование размеров буферов	380
21.21.1. Возвращаемое значение	381
21.22. Функции <i>GetDefaultCommConfig</i> и <i>SetDefaultCommConfig</i> : настройки порта по умолчанию	381
21.22.1. Возвращаемое значение	382
21.23. Функция <i>TransmitCommChar</i> : передача специальных символов	382
21.23.1. Возвращаемое значение	382
21.24. Функция <i>GetCommModemStatus</i> : статус модема	383
21.24.1. Возвращаемое значение	383
21.24.2. Пример вызова	383
21.25. Функция <i>WaitForSingleObject</i> : ожидание сигнального состояния объекта	384
21.25.1. Возвращаемое значение	385
21.26. Функция <i>WaitForMultipleObjects</i> : ожидание сигнального состояния объектов	385
21.26.1. Возвращаемое значение	386
21.27. Функция <i>GetOverlappedResult</i> : результат асинхронной операции	387
21.27.1. Возвращаемое значение	388
21.28. Функция <i>DeviceIoControl</i> : прямое управление драйвером	388
21.28.1. Возвращаемое значение	390
21.29. Функция <i>EnumPorts</i> : перечисление портов	390
21.29.1. Дополнительные сведения	391
21.29.2. Возвращаемое значение	391
21.29.3. Пример вызова	392
21.30. Функция <i>QueryDosDevice</i> : получение имени устройства по его DOS-имени	393
21.30.1. Возвращаемое значение	393
21.30.2. Пример вызова	394
21.31. Функция <i>DefineDosDevice</i> : операции с DOS-именем устройства	395
21.31.1. Возвращаемое значение	395
21.31.2. Пример вызова	395
Глава 22. Функции <i>DeviceIoControl</i> для последовательных портов	397
22.1. Функции драйвера последовательного порта	397
22.2. Структуры драйвера последовательного порта	404
22.3. Соответствие функций <i>DeviceIoControl</i> и Windows API	406
Глава 23. AT-команды и регистры модемов	409
23.1. Основные AT-команды модемов	409
23.2. Основные регистры модемов	416

Часть IV. ПРИЛОЖЕНИЯ.....	419
Приложение 1. Микросхемы UART	421
Общее описание	421
Характерные особенности.....	421
Описание контактов	422
Типы микросхем UART.....	426
Приложение 2. Библиотека функций для Pascal	427
Приложение 3. Формат <i>SINGLE</i> чисел с плавающей точкой	429
Приложение 4. Описание функций Windows для использования в Visual Basic	430
Приложение 5. Использование ActiveX-компонента <i>MSComm32</i>.....	433
Приложение 6. Сервис BIOS — обслуживание INT14H.....	436
Приложение 7. Как работает GiveIO	442
Приложение 8. Реализация процедуры преобразования <i>SINGLE</i>-чисел в строку.....	450
Приложение 9. Индикаторы состояния модема.....	455
Приложение 10. Инструменты	456
Порт-монитор.....	456
Разработка драйверов Driver Wizard.....	457
Разработка драйверов NuMega Driver Studio	457
Отладчик NuMega SoftICE.....	458
Эмулятор компьютера Connectix Virtual PC.....	458
Терминал HyperTerminal	458
Приложение 11. Формат команды <i>Mode</i>.....	460
Приложение 12. Часто задаваемые вопросы (FAQ).....	462
FAQ: Общие вопросы	462
1. Что такое COM и RS и чем они отличаются?	462
2. Что такое порт?	462
3. Чем "протокол RS-232" отличается от "интерфейса RS-232"?.....	462
4. Что такое контрольная сумма и как ее считать?	463

5. Откуда взялось ограничение числа абонентов в протоколе RS-485?	463
6. Почему нужно подключать COM-устройства при выключенном питании?	463
7. Откуда берется питание для мыши?	463
8. Что означает режим <i>HANDSHAKING</i> ?	464
9. Что такое дуплексная и полудуплексная передача?	464
FAQ: Программирование в DOS	464
1. Что такое базовый адрес порта?	464
2. Написал программу обработки прерываний порта, а она периодически "виснет". Для других прерываний все нормально работает	464
3. При выводе на экран данных, полученных в обработчике прерывания, выводится "мусор"	465
4. Прерывание и основная программа используют один буфер. Как их синхронизировать в DOS?	465
5. Как бы перехватить вывод программы в COM-порт под DOS?	465
FAQ: Программирование в Windows	466
1. Почему в Delphi пропал оператор <i>Port</i> и как тогда обращаться к портам?	466
2. Можно ли использовать прямое обращение к портам в Windows NT/2000?	466
3. Программа асинхронно работает с портом. В Windows 98 все работало, а в Windows 2000 или "виснет", или не работает	466
4. Зачем надо использовать <i>WaitForSingle Object</i> , если уже есть вызов <i>GetOverlappedResult</i> ?	466
5. Как найти все доступные COM-порты?	466
6. Как бы перехватить вывод программы в COM-порт под Windows?	467
7. Как выключить Plug and Play для COM-порта при загрузке Windows?	467
FAQ: Связь, модемы, терминалы	467
1. Что такое FIFO и FOSSIL?	467
Приложение 13. Описание компакт-диска	469
Литература и Интернет-ресурсы	471
Предметный указатель	472

Введение

В книге рассматриваются вопросы программирования, которые по непонятным причинам довольно скудно освещены в доступной литературе, — программирование последовательных коммуникационных устройств. В Интернете и на книжных полках можно найти множество книг, посвященных аппаратной организации последовательной связи. А вот вопросы программирования почему-то ограничиваются описанием битов порта или заголовками функций для работы с портами. Хочется надеяться, что представленная книга заполнит этот пробел.

Почему выбран именно последовательный интерфейс? Простота и низкие аппаратные требования (в сравнении, например, с параллельным интерфейсом) делают его заманчивым методом реализации систем обмена информацией. Видимо по этой причине форумы по программированию заполнены вопросами "Как работать с COM-портом из Windows?", "Как прочитать данные с порта" и т. д. И хотя USB-интерфейс постепенно вытесняет COM, последний остается пока наиболее распространенным, например, в системах промышленной автоматизации.

Операционная система Windows имеет настолько много функций, что начинающий программист просто не знает с чего начать и где искать информацию. Тем более, что функции для работы с портами вовсе не называются `OpenComPort`, а конфигурирование параметров порта ничем не напоминает программирование в DOS. Конечно, MSDN Library (Microsoft Development Network Library) содержит ответы на все эти вопросы, но, во-первых, только на английском языке, и, во-вторых, только на уровне заголовков функций, описания их параметров и скелетов программ. Наша книга содержит и описания, и примеры использования как низкоуровневых функций DOS, так и функций Windows.

Книга построена таким образом, чтобы переход из DOS в Windows и из Windows 98 в Windows 2000/XP стал достаточно простым и минимально трудоемким.

Все исходные коды приведены на прилагаемом компакт-диске.

Для кого эта книга

Мы адресуем книгу нескольким категориям читателей:

- тем, кому необходимо быстро разобраться в принципах работы с последовательными портами и интерфейсами, не углубляясь в дебри теории;

- тем, кого не устраивает стандартный уровень, предоставляемый функциями DOS или Windows, и кому требуется более тонкое программирование порта;
- тем, кому досталась поддержка чужих программ, работающих с оборудованием;
- тем, кому необходимо быстро переделать программу, работающую в старых операционных системах (например, DOS, Windows 95/98) в Windows 2000/XP, а времени на полную переработку нет;
- тем, кто хочет повысить свой профессиональный уровень;
- инженерам, связанным с электроникой, желающим написать программу для работы с оборудованием, не вникая в тонкости программирования;
- ну, и наконец, просто любителям подключать к компьютеру разные устройства и проводить эксперименты.

Даже если вы не относите себя ни к одной из вышеперечисленных категорий, не стоит сразу отставлять книгу на полку. Например, занимаясь не последовательными, а параллельными интерфейсами, вы можете найти необходимую информацию о создании драйверов и INF-файлов для них. Для тех, кого интересуют вопросы программирования "железа", полезным станет описание прямого доступа к портам в Windows 2000/XP и т. п.

Вы не найдете в этой книге ни подробных описаний аппаратной части последовательных интерфейсов, ни рекомендаций по разворачиванию сети на основе RS-485, ни формул для расчета максимальной длины проводника... Зато вы сможете найти листинги и исходные тексты действительно работающих программ с русскими комментариями.

Структура книги

Книга состоит из трех частей. В первой части даются теоретические сведения о последовательных интерфейсах, протоколах, методах коррекции ошибок и т. д. Вторая часть книги — практическая, про нее мы расскажем далее. В третьей части книги приводятся справочные материалы, описание функций, портов, регистров и другие необходимые для работы сведения.

Практически вся вторая часть построена на основе одного примера: передачи данных с одного последовательного порта на другой. В качестве приемника и передатчика можно использовать два компьютера, но это затруднительно как в плане их наличия, так и при отладке программы (если конечно между компьютерами нет локальной сети). В нашей книге мы будем ориентироваться на один компьютер и передавать данные с одного последовательного порта на другой. Все примеры построены по принципу усложнения задачи: от совсем простых примеров в DOS мы дойдем до сложных асинхронных программ в Windows.

Сначала мы будем работать в DOS. Для этого существует несколько причин. Во-первых, эта информация может потребоваться программистам, которым досталась поддержка чужих программ. Примеры будут построены таким образом, что переход в Windows будет достаточно простым делом. Более того, специальные модули позволят обращаться к коммуникационным портам из Windows NT/2000/XP точно так же, как это делается в DOS. Во-вторых, достаточно часто DOS (или его клон) используется в качестве операционной системы промышленных компьютеров. Ну, и наконец, не зная основ, читателю будет сложно ориентироваться в сложных функциях Windows. В DOS мы пройдем полный путь от простейшей программы в несколько строк, использующей сервис BIOS, до программы, обрабатывающей прерывания коммуникационного порта.

Программирование в Windows мы начнем с создания программы, в точности повторяющей структуру программы в DOS. Мы даже заставим работать такую программу в Windows 2000/XP. Затем освоим функции Windows. На этом пути мы познакомимся с синхронной и асинхронной связью, научимся использовать потоки и функции синхронизации. Отдельно рассмотрим функции, которые были добавлены только в Windows 2000/XP.

Программирование в Windows мы завершим созданием программы, демонстрирующей работу Plug and Play. Мы создадим эмулятор устройства, работающий с портом COM1, и заставим Windows найти новое устройство на порту COM2.

В отдельной главе мы рассмотрим интерфейс RS-485, требующий более глубоких знаний о работе коммуникационного порта.

Одна из глав будет посвящена вычислению контрольных сумм и реализации протоколов обмена.

В приложениях вы сможете найти ответы на наиболее часто задаваемые вопросы по программированию коммуникационных портов и некоторые полезные функции.

Краткое описание глав

Главы 1—5 содержат теоретическую часть книги.

- Глава 1 ("Стандарты последовательной связи") дает общие представления об аппаратной части последовательных протоколов, разъемах и о самом стандарте. Специального заострения на аппаратной части не делается. Мы расскажем о многих RS-протоколах, принципах последовательной связи, кратко опишем разъемы и кабели, а также дадим представление о ресурсах и функциях компьютера.
- В главе 2 ("Протоколы") рассматриваются основные методы организации последовательной связи с точки зрения программных протоколов обмена.

Мы расскажем о текстовых и бинарных протоколах, о методах предотвращения и обнаружения потери данных.

- В главе 3 ("Последовательные протоколы IBM PC") мы расскажем об использовании последовательной связи в персональном компьютере. Мы не будем рассматривать аппаратные реализации интерфейсов, а только лишь протоколы обмена устройств с компьютером. Эта глава будет полезна не только тем, кто интересуется работой устройств компьютера, но и тем, кто думает о создании своих протоколов обмена.
- В главе 4 ("Промышленные последовательные протоколы") рассмотрена возможность использования последовательных протоколов обмена в промышленных условиях. Мы познакомимся с протоколами MODBUS, ProfiBus и CAN. Опять же, мы будем рассматривать только программную составляющую протоколов, оставляя аппаратную реализацию за рамками книги.
- В главе 5 ("COM-порты и Plug and Play") мы расскажем о работе системы автоматического обнаружения устройств, дадим некоторые сведения о формате INF-файла, необходимого для установки драйвера устройства, и полностью опишем алгоритм работы Plug and Play для последовательных портов.

Главы 6—17 содержат практическую часть книги.

- Глава 6 ("Использование сервисов BIOS") начинает практическую часть с самой простой программы последовательной связи. Использование сервисов BIOS позволяет написать программу последовательного обмена всего из нескольких строк. К сожалению, такая программа будет очень ограничена: сервис BIOS не позволяет использовать микросхему UART на полную мощность. Эта глава будет интересна тем, кому приходится применять чужие модули последовательной связи, использующие сервисы BIOS. Кроме того, промышленные компьютеры с клоном DOS обычно предоставляют именно такой способ работы с внешними устройствами.
- В главе 7 ("Прямое программирование портов") избавиться от ограничений BIOS нам поможет прямое обращение к портам. Эти сведения будут полезны не только программистам, работающим в DOS, но и тем, кто создает свой драйвер для Windows.
- Использование прямого доступа не избавляет нас от неприятной необходимости постоянного опроса порта. В главе 8 ("Использование обработки прерываний") мы научимся обрабатывать прерывания последовательного порта и на этом завершим программирование в режиме DOS. Сведения этой главы пригодятся программистам, использующим промышленные контроллеры с установленным DOS.
- Глава 9 ("Переход в Windows") демонстрирует переход из DOS в Windows. Приведенные здесь программы еще не используют функций Windows, но

уже работают под управлением операционных систем Microsoft. Эта глава будет полезна тем, кто хочет перенести свои программы из DOS в Windows с минимальными изменениями. Вы сможете работать с портами в Windows 2000/XP так же как в DOS.

- В главе 10 ("Использование функций Windows") мы, наконец, доберемся до непосредственного использования функций Windows. Так же как в DOS мы начнем с простейшей программы обмена, но уже используя функции Windows.
- Глава 11 ("Использование потоков") по смыслу похожая на главу 8 ("Использование обработки прерываний"), рассказывает о том, как разгрузить основной процесс от постоянного опроса порта в Windows.
- Отдельный поток, созданный в главе 11, к сожалению, не избавляет от необходимости постоянного опроса порта, а значит, лишней загрузки процессора. В главе 12 ("Функции асинхронного доступа и события") используются асинхронные функции работы с портом совместно с функциями событий, которые позволяют нам создать поток, "просыпающийся" только в момент появления новых данных. Как мы покажем в нашем примере, загрузка процессора при этом будет практически незаметна.
- В главе 13 ("Специальные коммуникационные функции") мы опишем функции, которые не требовались в предыдущих главах, но потребуются в следующих. Кроме того, будут описаны функции, которые появились только в Windows 2000.
- До сих пор мы занимались приемом и передачей только одного байта. Для реальной программы этого маловато. В главе 14 ("Реализация протоколов обмена") мы создадим специальные компоненты, которые позволят нам разделить транспортные функции последовательного порта и функции работы с конкретным протоколом обмена.
- В главах теоретической части мы описывали протоколы обмена, такие как MODBUS, протоколы модемов XModem, YModem и другие, а в главе 15 ("Вычисление контрольных сумм") приведем примеры вычисления контрольных сумм, применяемых в этих протоколах.
- Несмотря на договоренность не рассматривать аппаратных реализаций протоколов, интерфейс RS-485 заслуживает отдельной главы 16 ("Интерфейс RS-485"), т. к. его программная реализация требует дополнительных знаний и функций.
- Конечно, описание протокола Plug and Play, которое мы привели в главе 5, достаточно интересно. Но ведь всегда гораздо интереснее попробовать самому! Программа из главы 17 ("Работа Plug and Play") будет эмулировать присутствие некоторого устройства, подключенного к порту COM2 с помощью обработки сообщений порта COM1. Также мы напишем INF-файл, устанавливающий "драйверы" нашего устройства.

Главы 18—23 содержат справочную часть книги.

- Глава 18 ("Сервис BIOS INT14H/INT21H") — справочник функций BIOS.
- Глава 19 ("Порты IBM PC") — справочник коммуникационных портов и их значений. С этими сведениями мы пользовались в главах 7—9.
- Глава 20 ("Структуры Windows для работы с COM-портами") — справочник структур Windows и описания полей этих структур.
- Глава 21 ("Функции Windows для работы с COM-портами") — справочник функций Windows.
- Глава 22 ("Функции DeviceIoControl для последовательного порта") — описание кодов прямого доступа к драйверу последовательного порта.
- Глава 23 ("AT-команды и регистры модемов") — краткий справочник основных AT-команд и регистров модемов.

Программные требования

Все программы мы будем реализовывать на языках Borland Pascal 7.0 и Borland Delphi 6. Поскольку мы не будем использовать никаких специфических функций, присущих именно этим версиям языков, все примеры могут быть скомпилированы в других версиях практически без модификации.

Версия Windows может быть любой — от Windows 98 до Windows 2000/XP. Конечно, примеры работы с функциями, которые появились только в Windows 2000, не будут работать в Windows 98, но обратной совместимости мы будем придерживаться: даже прямое программирование портов станет доступным в Windows 2000 с помощью специального модуля.

Все примеры, приведенные в книге, тестировались в версиях Windows 98 SE, Windows 2000 Workstation и Windows 2000 Advanced Server.

Для компиляции драйверов, приведенных в книге, потребуется MS Visual Studio и Windows 2000 DDK. На компакт-диске содержатся уже готовые модули драйверов, так что эту часть можно пропустить.

О программном коде

Книга содержит полные исходные коды всех программ, однако многие листинги содержат только изменения кода относительно предыдущего листинга. Такое сокращение позволяет не только экономить место, но и улучшить понимание кода, делая акцент только на новой функциональности. Код на компакт-диске содержит полные тексты без сокращений.

В программах для DOS мы не будем приводить код функций работы со строками. Код этих функций можно найти в приложениях или на компакт-диске к книге. В программах на Delphi мы не приводим код самого

проекта (DPR-файл) и код формы (DFM-файлы). Все исходные коды можно найти на компакт-диске. Еще раз повторим, что такие сокращения не означают отсутствие возможности скомпилировать и попробовать приведенные примеры на своем компьютере, а преследуют цель экономии места. Все необходимые исходные коды и модули находятся на компакт-диске.

Обозначения

При написании чисел мы будем придерживаться следующих правил:

- шестнадцатеричные числа будут иметь префикс "\$", например "\$45";
- шестнадцатеричные числа могут иметь префикс "0x" или постфикс "H", если того требует контекст изложения или формат строки, например "INT 14H";
- битовые последовательности заключены в угловые скобки, например "<0010>".

Аппаратные требования

Достаточно обычного домашнего компьютера, на котором компиляция программы в Delphi занимает приемлемое для вас время.

Для тестирования последовательной связи необходимо иметь либо два компьютера, либо один компьютер с двумя последовательными портами. В последнем случае нам потребуется PS/2- или USB-мышь, т. к. оба последовательных порта мы займем для экспериментов.

Для соединения портов между собой нам потребуется нуль-модемный кабель, желательно полностью распаянный, но для большей части программ достаточно трехпроводного кабеля.

Благодарности

В первую очередь автор хочет поблагодарить заместителя главного редактора Евгения Рыбакова. Без его советов и критики эта книга никогда бы не была создана. Автор благодарит всех друзей и родных, которые терпели его в процессе написания книги, а также коллектив издательства "БХВ-Петербург". Отдельная благодарность Вадиму Шарикову (компания Bixel, <http://bixel.ru>), дизайнеру сайта PvaSoft.

Обратная связь

Обо всех ошибках или пожеланиях сообщайте по адресу books@pvasoft.com.



ЧАСТЬ I

**ПРОТОКОЛЫ
И ИНТЕРФЕЙСЫ**

Глава 1



Стандарты последовательной связи

Стандарт есть легкоусвояемая форма, маскирующая присутствие или отсутствие содержания¹.

Глава дает общие представления об аппаратной части последовательных протоколов, развѣмах и о самом стандарте. Специального заострения внимания на аппаратной части не делается.

1.1. Стандарты последовательной связи

Последовательный интерфейс используется для связи двух устройств между собой. Данные в одну сторону передаются по одному проводу с помощью последовательности битов.

Мы будем рассматривать подключение одного или нескольких устройств к компьютеру, т. к. такая комбинация наиболее часто используется в реальной жизни. Естественно, при подключении нескольких устройств к компьютеру обмен производится только с одним из этих устройств.

Для соединения со стороны компьютера используется интерфейс, называемый *СОМ-порт* (COMmunication port, коммуникационный порт). Этот порт обеспечивает асинхронный обмен и реализуется на микросхемах универсальных асинхронных приемопередатчиков (UATR, Universal Asynchronous Receiver Transmitter), совместимых с семейством i8250. Хотя стандарт RS-232C предусматривает и асинхронный, и синхронный режимы обмена, СОМ-порт компьютера поддерживает только асинхронный режим. Для реализации синхронного обмена применяются специальные адаптеры, например, SDLC или V.35.

¹ Все афоризмы взяты с юмористического сайта <http://www.anekdot.ru>. Часть эпиграфов принадлежит перу Стаса Янковского <http://syy.narod.ru/wordsd.htm>.

Далее мы опишем основные протоколы последовательной связи, но сначала несколько замечаний.

Слово "протокол" вносит некоторую путаницу. С одной стороны, RS-232 и RS-485 называют протоколами, а с другой, MODBUS, ZModem и CAN — тоже протоколы. Какой же из них настоящий? Дело в том, что последовательная связь состоит из двух составляющих — аппаратной и программной.

Стандарты RS-232, RS-485 и другие описывают аппаратную часть: разъемы, назначение сигналов, уровни напряжений и т. п. Вторая составляющая — программная реализация протоколов, т. е. договоренность о правилах передачи данных. В нашей книге мы не будем подробно рассматривать аппаратные реализации всех интерфейсов. Исключение составит описание разъемов, сигналов и соединительного кабеля, необходимого нам для работы. А вот токовая петля, тонкости организации сетей RS-485 и подробное описание различий между RS-протоколами останутся за рамками книги. Интерфейс RS-485 удостоится отдельного рассмотрения из-за необходимости дополнительных программных решений при его реализации.

Вторую составляющую протокола — программную реализацию — мы будем рассматривать в *гл. 2*, а создание программ отложим до второй части книги.

И еще одно замечание. Все стандарты имеют двойные, а то и тройные названия. Двойственность названий связана с тем, что Международный союз электросвязи ИТУ-Т использует аналогичные стандарты с названиями V.xx. И, вообще говоря, два названия это еще не предел. Так, стандарту RS-232 соответствует ISO 2110, Министерство обороны США выпустило практически идентичный стандарт Mil-Std-188C, а в нашей стране подобный стандарт введен ГОСТом 18145-81.

Для краткости мы будем использовать два специальных обозначения:

- *DTE* (Data Terminal Equipment) — оконечное оборудование, принимающее или передающее данные. В качестве DTE может выступать компьютер, принтер, плоттер или другое периферийное оборудование;
- *DCE* (Data Communications Equipment) — аппаратура канала данных. Функция DCE состоит в обеспечении возможности передачи информации между двумя или большим числом DTE. Для этого DCE должно обеспечить соединение с DTE, с одной стороны, и с каналом передачи — с другой. Роль DCE чаще всего выполняет модем, например, как показано на рис. 1.1.

Устройства DTE могут быть соединены напрямую, без DCE, с помощью *нуль-модемного кабеля*, как показано на рис. 1.2. Разводка такого кабеля показана на рис. 1.3.



Рис. 1.1. Полная схема соединения по RS-232

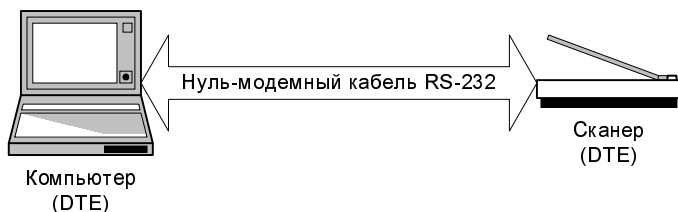
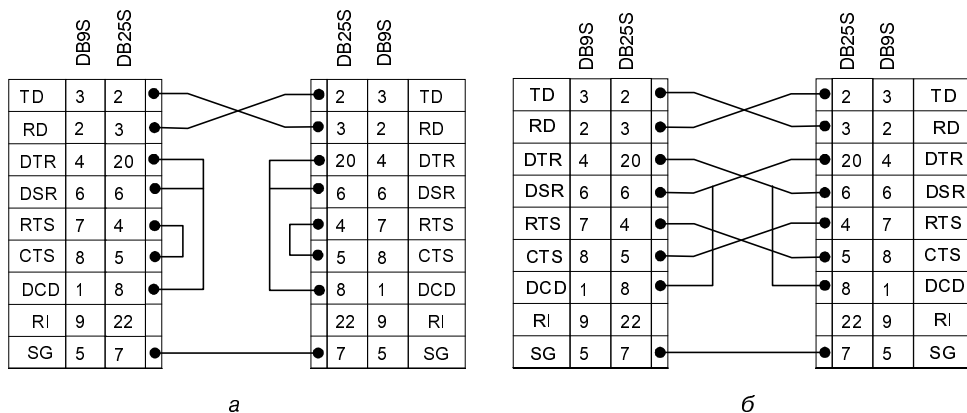


Рис. 1.2. Соединение по RS-232 нуль-модемным кабелем

Рис. 1.3. Нуль-модемный кабель:
а — минимальный; б — полный

1.1.1. Протокол RS-232

Из-за простоты и низких аппаратных требований (в сравнении, например, с параллельным интерфейсом), последовательные интерфейсы активно используются в электронной промышленности. В настоящее время наиболее распространенным является стандарт, разработанный Ассоциацией промышленных средств связи (TIA, Telecommunication Industry Association, <http://www.tiaonline.org>) и Ассоциацией электронной промышленности (EIA,

Electronic Industries Alliance, <http://www.eia.org>) "EIA/TIA-232-E", более известный под названием "RS-232".

Стандарт RS-232 (его официальное название "Interface Between Data Terminal Equipment and Data Circuit-Termination Equipment Employing Serial Binary Data Interchange") предназначен для подключения аппаратуры, передающей или принимающей данные, к оконечной аппаратуре каналов данных. Стандарт описывает управляющие сигналы интерфейса, пересылку данных, электрический интерфейс и типы разъемов.

Интерфейс RS-232 используется и во многих устройствах обычного персонального компьютера, начиная с "мыши" и модема до ключей аппаратной защиты. И хотя уже все компьютеры имеют интерфейс USB, интерфейс RS-232 еще жив и активно применяется.

Согласно стандарту RS-232, сигнал (последовательность битов) передается напряжением. Передатчик и приемник являются *несимметричными*: сигнал передается относительно общего провода (в отличие от симметричной передачи протокола RS-485 или RS-422). В табл. 1.1 приведены границы напряжений для сигналов приемника и передатчика. Логическому нулю на входе приемника соответствует диапазон +3...+12 В, а логической единице — диапазон -12...-3 В. Диапазон -3...+3 В — зона нечувствительности, обеспечивающая гистерезис приемника (передатчика). Уровни сигнала на выходах должны быть в диапазоне -12...-5 В для представления логической единицы и +5...+12 В для представления логического нуля.

Вообще говоря, стандарт RS-232 состоит из трех частей. Первая часть, стандарт RS-232C, была принята в 1969 году и содержит описание электрических цепей и сигналов несимметричной последовательной связи. Вторая часть, стандарт RS-232D, принята в 1987 году и определяет дополнительные линии тестирования, а также формально описывает разъем DB-25. Третья часть, RS-232E, принята в 1991 году.

Таблица 1.1. Границы напряжений COM-порта (стандарт RS-232)

	Диапазон напряжения входа приемника	Диапазон напряжения выхода передатчика	Состояние управляющего сигнала	Состояние линии данных
Логический 0	от -12 до -3 В	от -12 до -5 В	ON	MARX
Логическая 1	от +3 до +12 В	от +5 до +12 В	OFF	SPACE

Важно!

Интерфейс не обеспечивает гальванической развязки устройств. Подключение и отключение интерфейсных кабелей устройств с независимым питанием должно производиться при отключенном питании.

Один из самых неприятных недостатков стандарта RS-232 — плохая помехозащищенность и, соответственно, короткие линии передачи. Естественно, были созданы стандарты, решающие эти проблемы.

1.1.2. Протокол RS-422A

Стандарт RS-422A (другое название ITU-T V.11) определяет электрические характеристики симметричного цифрового интерфейса. Он предусматривает работу на более высоких скоростях (до 10 Мбит/с) и больших расстояниях (до 1000 м) в интерфейсе DTE-DCE. Для его практической реализации, в отличие от RS-232, требуются два физических провода на каждый сигнал. Реализация симметричных цепей обеспечивает наилучшие выходные характеристики.

Подобно V.28, данный стандарт является простым описанием электрических характеристик интерфейса и не определяет параметры сигналов, типы разъемов и протоколы управления передачей данных. Для линий интерфейсов RS-422A и RS-423A могут быть использованы различные проводники (или пары проводников) одного и того же кабеля.

Стандарт RS-422A был разработан совместно с RS-423A и позволяет размещать линии этих интерфейсов в одном кабеле. Он несовместим с RS-232, и взаимодействие между RS-422A и RS-232 может быть обеспечено только при помощи специального интерфейсного конвертера.

Практически это полнодуплексный протокол RS-485. Прием и передача идут по двум отдельным парам проводов, причем на каждой паре может быть только по одному передатчику.

1.1.3. Протокол RS-423A

Стандарт RS-423A (другое название V.6) определяет электрические характеристики несимметричного цифрового интерфейса. "Несимметричность" означает, что данный стандарт, подобно RS-232, для каждой линии интерфейса использует только один провод. При этом для всех линий используется единый общий провод.

Как и RS-422A, этот стандарт не определяет сигналы, конфигурацию выводов или типы разъемов. Он содержит только описание электрических характеристик интерфейса. Стандарт RS-423A предусматривает максимальную скорость передачи 100 Кбит/с.

1.1.4. Протокол RS-485

Полное название этого стандарта TIA/EIA-485A (аналогичный стандарт описывается в ISO 8482). Он включает в себя 17 страниц и определяет требования к электрическим характеристикам формирователей и приемников.

В нем ничего не говорится ни о качестве сигнала, ни о методах доступа к линиям связи, ни о протоколе обмена, ни о технологической схеме (назначении контактов и т. п.). "Приложение А" этого стандарта описывает принципы работы устройств на основе RS-485.

В дополнение к описанию стандарта был опубликован документ TSB89, называющийся "Application Guidelines for TIA/EIA-485A". TSB89 состоит из 23 страниц и посвящается разъяснению, как применять устройства, определяемые стандартом TIA/EIA-485A для физических сетей.

Подробно этот интерфейс будет обсуждаться в *гл. 16*.

1.1.5. Протокол RS-499

Стандарт RS-449, в отличие от RS-422A и RS-423A, содержит информацию о параметрах сигналов, типах разъемов, расположении контактов и т. п. В этом отношении RS-449 является дополнением к стандартам RS-422A и RS-423A. Стандарту RS-449 соответствует международный стандарт V.36.

Комбинация RS-449, RS-422A и (или) RS-423A первоначально предназначалась для замены RS-232. Однако этого не произошло, хотя данные стандарты нашли достаточно широкое применение в качестве высокоскоростного интерфейса DTE-DCE.

Стандарт RS-449 определяет 30 сигналов интерфейса, большинство из которых имеют эквиваленты в RS-232, но добавлены и новые. Обозначения большинства сигналов были изменены во избежание путаницы.

Десять сигналов RS-449 определены как линии первой категории. Эта группа сигналов включает в себя все основные сигналы данных и синхронизации, такие как "Передаваемые данные", "Принимаемые данные", "Синхронизация терминала". Скорость передачи сигналов первой категории существенно зависит от длины кабеля. Для линий этой категории на скоростях до 20 Кбит/с могут использоваться стандарты RS-422A либо RS-423A; на скоростях выше 20 Кбит/с (до 2 Мбит/с) — только RS-422A.

Оставшиеся 20 линий классифицируются как линии второй категории и используются стандартом RS-423A. К этой категории относятся такие управляющие линии, как "Качество сигнала", "Выбор скорости передачи" и др.

Стандарт RS-449 определяет тип разъема и, в отличие от RS-232, распределение контактов разъема. Используемые разъемы имеют 37 контактов для прямого и 9 контактов для обратного канала.

1.1.6. Протокол RS-562

Протокол RS-562 представляет собой низковольтную реализацию RS-232 с максимальным уровнем сигнала 3,5 В.

1.1.7. Протокол V.24

Рекомендация V.24 содержит описание линий и набора сигналов обмена между DTE и DCE. В RS-232 используются другие обозначения линий, однако линии интерфейса RS-232 и рекомендации V.24 выполняют совершенно одинаковые функции. V.24 определяет большее количество линий, чем RS-232, поскольку стандарт V.24 используется и в других интерфейсах. В этом смысле RS-232 является подмножеством V.24. Рекомендация V.24 не определяет электрические характеристики или другие физические аспекты реализации, такие как тип разъема, расположение контактов, длина кабеля и скорость обмена. Технические вопросы реализации интерфейса подробно изложены в стандарте V.28.

1.1.8. Протокол V.28

Рекомендация V.28 определяет только электрические характеристики интерфейса V.24, обеспечивающего работу по несимметричным двухполярным линиям обмена на скоростях до 20 Кбит/с. К таким характеристикам относятся уровни используемых сигналов, емкостное сопротивление и т. д. Данная рекомендация не содержит требований к длине кабеля, типу разъемов и расположению их контактов. Поэтому рекомендация V.28 может рассматриваться как подмножество стандарта RS-232.

1.1.9. Протокол V.35

Стандарт V.35 появился в начале 1980-х годов как спецификация интерфейса между устройствами доступа к сети (мультиплексором, модемом или др.) и высокоскоростной сетью с коммутацией пакетов. Первоначально эта спецификация использовалась для подключения групповых модемов (модемных пулов) к коммутационному устройству.

Рекомендация V.35 определяет синхронный интерфейс для работы по аналоговым широкополосным каналам с полосой пропускания 60—108 кГц и скоростью передачи до 48 Кбит/с.

В приложении к стандарту определяется вид электрического соединения, обеспечивающего высокоскоростной последовательный интерфейс между мультиплексором и коммутационным оборудованием сети.

Модемы, использующие стандарт V.35, не получили широкого распространения на рынке, но интерфейс в качестве высокоскоростной замены RS-232 прижился. В спецификации стандарта не был определен тип электрического разъема, но фирма IBM в свое время стала выпускать совместимые с V.35 большие прямоугольные разъемы с массивными прижимными винтами. Получилось очень надежное соединение. Остальные производители коммутационной техники стали повторять конструкцию соединителя IBM, который и стал стандартом де-факто, и был принят в качестве рекомендации ISO 2593.

1.1.10. Протокол X.21

Стандарт X.21 впервые был опубликован в 1972 году. Он определяет физические характеристики и процедуры управления для интерфейса DTE-DCE в режиме синхронной передачи данных и может применяться как в сетях с коммутацией каналов, так и в сетях на выделенных линиях. Стандарт предусматривает дуплексную работу при условии, что устройства связаны друг с другом реальными, а не виртуальными цифровыми линиями связи. Функциональные процедуры X.21 формализованы в виде диаграмм состояний, рассмотрение которых выходит за рамки данной книги.

Рекомендация ITU-T X.21 определяет формат передаваемых символов, которые представляются в коде МТК-5 (Международный телеграфный код № 5). Данный интерфейс рассчитан на сквозную цифровую передачу. В нем процесс установления соединения и разъединения полностью автоматизирован при помощи набора сигналов о состоянии соединения и о его неисправностях. В ходе передачи данных через интерфейс могут передаваться любые последовательности битов.

Создатели этого стандарта стремились максимально упростить его. Так, соединение DTE с DCE требует существенно меньшего числа сигнальных линий, чем аналогичное соединение для интерфейса RS-232.

1.1.11. Рекомендация X.21bis

Рекомендация X.21bis была разработана для обеспечения возможности подключения к сетям передачи данных общего пользования тех пользователей, которые используют для этого аналоговые выделенные или коммутируемые каналы и имеют синхронные модемы, работающие согласно рекомендациям серии V.

1.1.12. Краткое сравнение RS-протоколов

В табл. 1.2 мы привели краткое сравнение 4 наиболее распространенных RS-протоколов. Более подробную информацию можно найти в соответствующей литературе.

Таблица 1.2. Сравнение последовательных интерфейсов

	RS-232CV.24	RS-422AV.11	RS-423AV.6	RS-485
Максимальная длина кабеля, м	15	1200	1200	1200
Число приемников	1	10	10	32
Число передатчиков	1	1	1	32