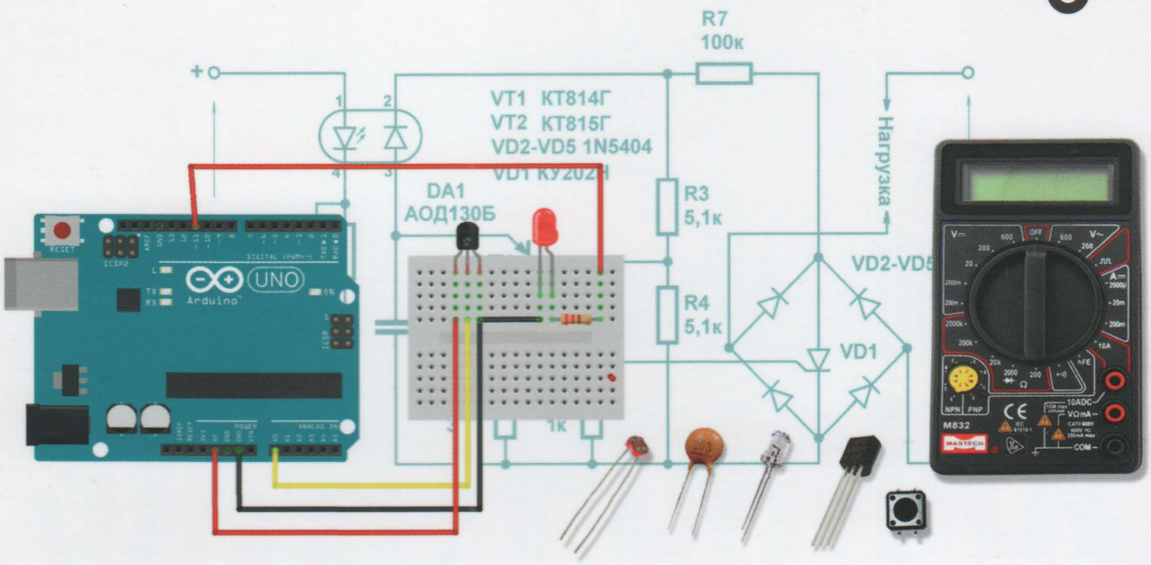
Электроника

Юрий Ревич

bhv®



занимательная ЭЛЕКТРОНИКА

- Начала начал электроники
- Оборудуем домашнюю лабораторию
- Резисторы, конденсаторы, транзисторы, микросхемы…
- Аналоговые и логические схемы
- Arduino — современная электроника для чайников
- Разнообразие применений Arduino на практике

5-е издание

Юрий Ревич

занимательная ЭЛЕКТРОНИКА

5-е издание

Санкт-Петербург
«БХВ-Петербург»
2018

УДК 621.3
ББК 32
Р32

Ревич Ю. В.

Р32 Занимательная электроника. — 5-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2018. — 672 с.: ил. — (Электроника)

ISBN 978-5-9775-3961-6

На практических примерах рассказано о том, как проектировать, отлаживать и изготавливать электронные устройства в домашних условиях. От физических основ электроники, описания устройства и принципов работы различных радиоэлектронных компонентов, советов по оборудованию домашней лаборатории автор переходит к конкретным аналоговым и цифровым схемам, включая устройства на основе микроконтроллеров. Приведены элементарные сведения по метрологии и теоретическим основам электроники, методики расчета трансформаторов и радиаторов, выбора силовых транзисторов. Даны практические советы по выбору паяльника, правильной организации электропитания, изготовлению и оформлению корпусов и многие другие. В 5-м издании обновлены разделы, содержавшие устаревшие сведения, а также значительно расширен раздел о платформе Arduino, с которой читателю становятся доступными самые современные радиоэлектронные средства. В подробностях изложены особенности применения компонентов самого разного направления — датчиков различных величин, часов реального времени, дисплеев разнообразных конфигураций, дистанционных передатчиков и приемников и т. п.

Для широкого круга любителей электроники

УДК 621.3
ББК 32

Группа подготовки издания:

Руководитель проекта	*Евгений Рыбаков*
Зав. редакцией	*Екатерина Капалыгина*
Компьютерная верстка	*Ольги Сергиенко*
Дизайн обложки	*Марины Дамбиевой*

Подписано в печать 18.
Формат 70×100$^{1}/_{16}$. Печать офсетная. Усл. печ. л. 54,18.
Тираж 1500 экз. Заказ
"БХВ-Петербург", 191036, Санкт-Петербург, Гончарная ул., 20.

Отпечатано с готового оригинал-макета
ООО "Принт-М", 142300, М.О., г. Чехов, ул. Полиграфистов, д. 1

ISBN 978-5-9775-3961-6

Оглавление

К читателю **11**
Радиолюбительство — что это такое? 12
Как пользоваться книгой? 15
Как разрабатывать электронные схемы? 17

ЧАСТЬ I. ОСНОВЫ ОСНОВ **21**

ГЛАВА 1. ЧЕМ ОТЛИЧАЕТСЯ ТОК ОТ НАПРЯЖЕНИЯ? **23**
Связь тока и напряжения 24
Регулирование тока с помощью сопротивления 27
Источники напряжения и тока 30

ГЛАВА 2. ДЖЕНТЛЬМЕНСКИЙ НАБОР
Оборудуем домашнюю лабораторию **33**
Мультиметр 34
Источник питания 38
Осциллограф 41

ГЛАВА 3. ХОРОШИЙ ПАЯЛЬНИК — ПОЛОВИНА УСПЕХА
Инструменты и технологические советы **47**
Инструменты и материалы 48
Паяльники 51
Флюсы для пайки 53
Макетные платы 55
Печатные платы 60
Монтаж 61
Немного о проводах 63
Корпуса 65
Новые подходы в любительском конструировании 67
Платформа Arduino 67
Редактор Fritzing для рисования схем и плат 68

Глава 4. Тригонометрическая электроника
О частотах, периодах, мощности, переменных напряжениях и токах и немного о сигналах 73
Мощность 77
Что показывал вольтметр? 78
Сигналы 81
О переменном токе и электропитании 82
Децибелы 83

Глава 5. Электроника без полупроводников
Резисторы, конденсаторы и схемы на их основе 85
Резисторы 85
Переменные резисторы 89
Параллельное и последовательное соединение резисторов 91
Конденсаторы 92
Параллельное и последовательное включение конденсаторов 101
Конденсаторы в цепи переменного тока 102
Дифференцирующие и интегрирующие цепи 103
Индуктивности 105

Глава 6. Изобретение, которое потрясло мир
Диоды, транзисторы и простейшие схемы на их основе 109
Диоды 109
Транзисторы 111
Ключевой режим работы биполярного транзистора 114
Усилительный режим работы биполярного транзистора 117
Включение транзистора с общим коллектором 119
Стабильный усилительный каскад на транзисторе 120
Дифференциальный каскад 123
Полевые транзисторы 124
Выбор транзисторов 129

Глава 7. Ошеломляющее разнообразие электронного мира
Реле, стабилитроны, светодиоды... 135
Электромагнитные реле 135
Стабилитроны 141
Оптоэлектроника и светодиоды 143
Оптоэлектроника 144
Светодиоды 146
Светодиодные индикаторы 149
ЖК-дисплеи 150
Простейший уровнемер для водяных баков 153

ЧАСТЬ II. АНАЛОГОВЫЕ СХЕМЫ 157

Глава 8. Звуковой усилитель без микросхем
Классическая схема УМЗЧ 159
Схема базового УМЗЧ 160
Мощность усилителя 163
Стабильность 164
О мощности выходных транзисторов 166
Проверка и отладка 166
Классы усилителей или немного высшей математики 168
Действующее значение напряжения 168
Классификация усилителей 168
О мощности и качестве звуковых усилителей 172

Глава 9. Правильное питание — залог здоровья
О питании электронных устройств 175
Электрохимические элементы 175
Аккумуляторы 179
Вторичные линейные источники питания 183
Трансформаторы 183
Расчет сетевого трансформатора 185
Простейший нестабилизированный однополярный источник питания 187
Стабилизаторы 190
Интегральные стабилизаторы 193
Однополярный регулируемый источник питания 195
Рассеивание тепла 200
Принудительное охлаждение и элементы Пельтье 203
Импульсные источники питания 205
Как правильно питаться? 211

Глава 10. Тяжеловесы
Устройства для управления мощной нагрузкой 215
Базовая схема регулирования напряжения на нагрузке 216
Мощность в нагрузке при тиристорном управлении 219
Ручной регулятор мощности 221
Базовая схема регулятора (диммера) 221
Регулятор переменного напряжения с двумя тиристорами 225
Регулятор с симистором 226
Устройство плавного включения ламп накаливания 228
Помехи 229

Глава 11. Слайсы, которые стали чипами
О микросхемах 233
Некоторые типовые узлы микросхем и особенности их эксплуатации 237
Звуковые усилители на микросхемах 243
Мощный УМЗЧ 243
Микроусилитель мощности 247

Глава 12. Самые универсальные
Обратная связь и операционные усилители 249
Опасные связи 250
Основные свойства системы с отрицательной обратной связью 254
Базовые схемы усилителей на ОУ 256
Неидеальность ОУ, ее последствия и борьба с ними 258
Дифференциальные усилители 260
Другие распространенные схемы на ОУ 263
Аналоговый генератор 265
Релейное регулирование и термостаты 267
Термостат вообще 268
Простой термостат для аквариума 272
О гистерезисе 276
Терморегулятор «для дома для семьи» 277

Глава 13. Как измерить температуру?
Электронные термометры 283
Основы термометрии 284
Датчики 285
Термисторы 286
Металлические датчики 288
Полупроводниковые датчики 288
Методы измерения сопротивления 289
Очень точный ручной измеритель температуры 291
Простейшие электронные термометры на батарейке 293
Электронный термометр со стрелочным индикатором 294
Немного о метрологии и ошибках аналоговых схем 296
Точность и разрешающая способность 298
Систематические ошибки 299
Случайные ошибки измерения и их оценка 299
Регрессия и метод наименьших квадратов 304
Разновидности погрешностей 306

Часть III. Цифровой век 307

Глава 14. На пороге цифрового века
Математическая логика и ее представление в технических устройствах 309
Основные операции алгебры Буля 311
Булева алгебра на выключателях и реле 314
То же самое, но на транзисторах и диодах 318
О двоичной и других системах счисления 320
Позиционные и непозиционные системы счисления. Десятичная система 320
Двоичная и шестнадцатеричная системы 323
Перевод из одной системы счисления в другую 324
Байты 326
Запись чисел в различных форматах 328
Немного двоичной арифметики 329
Отрицательные двоичные числа 330

Дробные числа ... 332
Коды, шифры и дешифраторы ... 333
Код Грея ... 337

Глава 15. Математическая электроника или игра в квадратики
Устройство логических микросхем и двоичные операции ... 341
ТТЛ ... 342
Основные характеристики КМОП ... 344
Характеристики различных серий КМОП ... 346
Двоичный сумматор на логических микросхемах ... 353
Обработка двоичных сигналов с помощью логических элементов ... 358
Мультиплексоры/демультиплексоры и ключи ... 361

Глава 16. Устройства на логических схемах
Мультивибраторы, формирователи, триггеры, счетчики ... 363
Генераторы ... 363
Схемы на основе триггера Шмидта ... 369
Кварцевые генераторы ... 370
Формирователи импульсов ... 374
Одновибраторы ... 376
Одновибраторы и генераторы на микросхеме 555 ... 379
Триггеры, регистры и счетчики ... 384
Самый простой триггер ... 384
D-триггеры ... 387
Регистры ... 389
Счетчики ... 390
Цифровой лабораторный генератор ... 396

Глава 17. Откуда берутся цифры
Цифроаналоговые и аналого-цифровые преобразователи ... 399
Принципы оцифровки сигналов ... 401
ЦАП ... 404
АЦП ... 408
АЦП параллельного действия ... 408
АЦП последовательного приближения ... 409
Интегрирующие АЦП ... 410
Конструируем цифровой термометр ... 419
АЦП 572ПВ2 и ПВ5 ... 419
Практическая схема термометра ... 424

Часть IV. Микроконтроллеры ... 431

Глава 18. Начала микроэлектроники
Микропроцессоры, память и микроконтроллеры ... 433
Как работает микропроцессор? ... 436
Лечение амнезии ... 443
Изобретаем простейшую ROM ... 444
Общее устройство памяти ... 445

RAM 447
EPROM, EEPROM и flash-память 448
Микроконтроллеры Atmel AVR 452
Почему AVR? 453
Classic, Mega и Tiny 454
Структура МК AVR 454
Память программ 455
Память данных (ОЗУ, SRAM) 456
Энергонезависимая память данных (EEPROM) 457
Способы тактирования 458
Параллельные порты ввода/вывода 460
Прерывания 462
Таймеры-счетчики 464
Последовательные порты 465

Глава 19. Персональный компьютер вместо паяльника
О программировании МК на языке ассемблера 469
Железо 469
Софт 473
О конфигурационных битах 476
Примеры программирования 478
Самая простая программа 479
Таймер без прерываний 483
Применение прерываний 487
Прерывание таймера по переполнению 490
Прерывание таймера по сравнению 493
Арифметика многоразрядных чисел на ассемблере 496
Операции с числами в формате BCD 499

Глава 20. Основы Arduino
Контроллеры, среда и примеры программирования 503
Что такое Arduino? 504
Основные платы Arduino 506
Установка среды программирования Arduino 510
Настройки Arduino IDE 514
Программы для Arduino 515
Примеры программирования 522
Обмен через последовательный порт 522
Термостат на Arduino 525
Правильное подключение кнопки 528
Правильная мигалка на Ardiuno 532

Глава 21. Компоненты для Arduino
Как на Arduino делать устройства лучше фирменных 535
Техническое задание 536
О выборе компонентов 537
Интерфейс TWI (I^2C) 537

Датчики метеорологических параметров 540
Датчики температуры и влажности 540
Барометры 543
Другие метеорологические датчики 545
Особенности калибровки цифровых датчиков 546
Часы 548
Простейшие дисплеи 549
Подключение цифрового 4-разрядного дисплея к Arduino 551
Часы на 4-разрядном дисплее 552
Простой электронный термометр 555
Arduino и поразрядные матричные индикаторы 558
Схема подключения драйвера MAX6953 с I²C-интерфейсом 560
Программа 563
Работа с текстом на графическом дисплее MT-12864J 566
Подключение MT-12864J 566
Русификация модуля MT-12864J 568
Строчные OLED-дисплеи 571
Контроллер WS0010 и библиотека LiquidCrystal 572
Пишем по-русски 573
Подключение строчных дисплеев Winstar 575
Часы на основе OLED-дисплея 576
Графические дисплеи Winstar 579
I²C-интерфейс для дисплеев Winstar 584
Передача данных по радиоканалу 586
Подключение и настройка Xbee-модулей 587
Подключение передатчика и приемника 433 МГц 591
О конструктивном оформлении устройств на Arduino 596
О режиме энергосбережения, Watchdog-таймере и питании метеостанции 598
Watchdog Timer 599
О мерах по снижению энергопотребления 603

Глава 22. Применения Arduino

Избранные возможности платформы 605

Аналоговое управление внешними устройствами (ШИМ) 605
Принцип ШИМ-регулирования 606
ШИМ и Arduino 607
Подбор MOSFET-ключей и драйверов для мощной нагрузки 610
Запись на SD-карту 613
Подключение тастатуры 3×4 615
Измерение частоты в Arduino 620
Метод первый — измерение частоты 620
Метод второй — измерение периода 621
Пироэлектрический датчик 623
Управляем с ИК-пульта от телевизора 625
Определение и применение кодов команд с ИК-пульта 626
Двухкнопочный плавный регулятор с запоминанием состояния 629
Программа регулятора 630
О преимуществах и недостатках Arduino (вместо послесловия) 633

ПРИЛОЖЕНИЯ **637**

Приложение 1. Резисторы и конденсаторы **639**
Международная цветная маркировка резисторов (с допуском 5 и 10%) 639
Таблицы номиналов резисторов и конденсаторов 640

Приложение 2. Стандартные обозначения, размеры и характеристики некоторых гальванических элементов **641**

Приложение 3. Соответствие наименований и функциональности некоторых зарубежных и отечественных цифровых микросхем **643**

Приложение 4. Словарь часто встречающихся аббревиатур и терминов **647**

Приложение 5. Единицы измерения и обозначения **653**
Физические величины и их единицы измерения по умолчанию 653
Приставки и множители для образования десятичных кратных и дольных единиц 654
Некоторые буквенные обозначения в электрических схемах 654
Некоторые символические обозначения в электрических схемах 655
Символические обозначения мощности резисторов на схемах 657

Приложение 6. Программирование Arduino Mini и Pro Mini **659**
Программирование Mini с помощью платы Arduino Uno 659
Программирование Mini через адаптер USB-UART 660
Программирование Arduino Mini через адаптер USB-UART с автоматическим сбросом 661

Литература и источники **664**

Предметный указатель **666**

К читателю

Как известно, каждый сходит с ума по-своему. Есть люди, сдвинутые на собирании спичечных этикеток или монет, есть те, кто прыгает с парашютом, лазает по городской канализации или спускается на плотах по горным рекам Северного Урала и Сибири. Одна из самых распространенных разновидностей подобных психических сдвигов — радиолюбительство.

Когда-то радиолюбители были действительно только «любителями радио» — просто потому, что в 20–30-х годах XX века, когда появились первые энтузиасты, кроме радиоэлектроники, почти никакой другой электроники не существовало. О, в те времена это было жутко престижное хобби! Это сейчас мы, не успев привыкнуть к магнитолам в каждом автомобиле и двум-трем телевизорам в каждой квартире, как-то незаметно для себя оказались в мире глобальных электронных коммуникаций. А тогда сама возможность слушать кого-то с другого конца Земли казалась чудом, магическим действием. В те времена радиолюбители нередко участвовали в важнейших событиях наравне с профессионалами, или даже опережали их — достаточно вспомнить радиолюбителя Николая Шмидта, деревенского жителя из Северо-Двинской губернии, который в 1928 году первым поймал SOS экспедиции Нобиле, тем самым установив постоянную связь с потерпевшим катастрофу экипажем дирижабля «Италия».

У обычных людей в те времена даже настройка на нужную станцию простого домашнего приемника, ставшего к тридцатым годам уже довольно обычным атрибутом не только в богатой Америке, но и в СССР, поначалу вызывала не меньше вопросов, чем сейчас установка и настройка скайпа или регистрация своего аккаунта в социальной сети.

Но довольно быстро — начиная с 1940-х годов — электроника стала «широко простирать руки свои в дела человеческие». Термин «радиолюбительство» сохранился и по сей день, но под ним стали понимать уже отнюдь не только увлечение радиопередатчиками и приемниками. Первым делом выделилась в отдельное направление звукозапись и все с ней связанное: различные усилители, эквалайзеры, акустические устройства. Затем электроника вторглась в электротехнику и управление разными механизмами. Потом начался компьютерный бум, и стало модным все, связанное с информационными технологиями (вообще-то в них, по справедливо-

сти, следовало бы включить и радио с телевидением). Заметим, что в развитии ИТ-отрасли любители сыграли не последнюю роль — так, из увлечения Стива Возняка конструированием игровых приставок выросла компания Apple, теперь одна из самых известных и дорогих компаний мира. А из сообщества любителей программирования родилось течение «открытых исходников» (Open Source), в существенной мере определяющее ландшафт современного ИТ-мира.

В результате первоначально единая наука о применении электричества в технике разделилась на множество течений, значительная часть которых уже к собственно электричеству имеет лишь косвенное отношение. Выдающийся советский ученый Сергей Алексеевич Лебедев, один из создателей отечественной компьютерной промышленности, начинал с проектирования устойчивых энергосистем (т. е. с чисто электротехнических задач), а к концу своей творческой деятельности занимался уже проблемами операционных систем для сверхмощных вычислительных комплексов.

> В настоящее время не более одной пятой части объема журнала «Радио», выходящего в нашей стране с 1924 года, посвящено именно радио. Следовало бы придумать иное название, но слово «радиолюбитель» прижилось и ныне означает любого, кто увлекается электроникой. Правда, так только по-русски — например, в английском языке все иначе: в нем radio ham (или на жаргоне просто ham) есть лишь «оператор любительской радиостанции», а для любителей электроники (amateurs electronics) вообще нет никакого специального термина. Только не думайте, что в Англии или в Штатах таких любителей вовсе нет, просто там это хобби традиционно имело несколько иной статус. В последние годы число таких amateurs на Западе растет опережающими темпами — возникли даже целые отрасли промышленности, выпускающие полуфабрикаты и конструкторы для любителей создания самых разных электронных изделий. Об одном из самых ярких примеров этого направления — платформе Arduino, быстро завоевывающей и отечественную аудиторию, пойдет речь в последних главах этой книги. Другим отличным примером может стать конструирование «интеллектуальных» роботов. В быту, за пределами военных и промышленных сфер применения, примеры массово выпускаемых роботов пока еще можно пересчитать по пальцам, зато любительских конструкций создано неисчислимое количество: по отдельным разновидностям «умных роботов» проводятся даже чемпионаты мира — совсем, как когда-то по любительской радиосвязи.

Потому не удивляйтесь, что в этой книге, адресованной начинающим (и просто желающим повысить свою квалификацию) радиолюбителям, о радио вы вообще не прочтете ни слова. Зато довольно часто будут упомянуты компьютеры — по причинам, которые вы поймете, прочитав эту книгу. Так что же такое радиолюбительство?

Радиолюбительство — что это такое?

Чем отличается «электронное» устройство от «электрического»? Оба они используют электрическую энергию, однако электрическое устройство, как правило, не содержит никаких заумных штучек, вроде транзисторов или микросхем, — простейшим примером электрического (электротехнического) устройства в чистом виде является настольная лампа с выключателем. Но та же лампа, снабженная регулятором яркости или бесконтактным сенсорным выключателем, является уже

устройством электронным. Электронными устройствами по сути стали и давно известные люминесцентные лампы — теперь во многие из них встроена довольно серьезная схема, управляющая запуском и свечением. А в таких сложных электротехнических системах, как, например, современные генераторы энергии или устройства электропривода, электронную часть от электротехнической отделить уже невозможно.

Эту тенденцию легко проследить на примере эволюции автомобильных электросистем, где первоначально генератор электроэнергии совместно с электромеханическим регулятором и аккумулятором представляли собой отдельную вспомогательную систему, при выходе из строя которой автомобиль тем не менее можно было завести «ручкой» и заставить передвигаться. Нарастание количества электронных узлов в конце концов связало электротехнические компоненты (стартер, генератор, аккумулятор), электронные устройства (электронное зажигание, датчики, контроллеры, вплоть до бортового компьютера) и механические узлы (двигатель с трансмиссией) в неразрывное целое, где ни одна из систем не может функционировать без другой. А некоторые последние модели автомобилей высшего класса фактически уже ближе к роботам, а не просто к автомашинам, т. е. представляют собой сложнейший электронно-программно-механический агрегат (пример — модели «Вольво», оборудованные системой Drive Me).

Занятие радиолюбительством в нашей стране некоторое время назад, в 1960–1980-е годы, было не просто модным, а очень модным. Этому способствовало много причин: и относительно высокий уровень технического образования, и бесплатный доступ к компонентам (да-да, купить в свободной продаже что-то электронное было очень сложно, а вот вынести с завода или из НИИ — всегда пожалуйста), и, наконец, то, что промышленность явно не справлялась с обеспечением потребности населения в «продвинутых» бытовых приборах, а качество тех, что выпускались, чаще всего было ниже всякой критики. Дешевле, лучше и интереснее было сделать самому. Поэтому в те времена стать меломаном в современном смысле (т. е. «любителем качественной звукозаписи») означало фактически, что человек был вынужден сам изучать азы электроники и браться за паяльник.

Положение, конечно, резко изменилось с приходом в страну дешевого и качественного ширпотреба с Запада и Востока, и теперь уже вряд ли кто будет самостоятельно изобретать, скажем, карманный плеер-«дебильник». Но, как ни странно, радиолюбительство не только не погибло, но даже расцвело — прежде всего потому, что стали доступны, пусть и за деньги, практически любые, как импортные, так и отечественные компоненты, и, что немаловажно, исчерпывающая документация к ним — в 70-е годы прошлого века какой-нибудь «Справочник по транзисторам» сметали с прилавков со скоростью, которой могли бы позавидовать сами братья Стругацкие. Популярная сеть электронных супермаркетов «Чип и Дип», ориентированная в основном на радиолюбителей, недаром хвастается миллионом посетителей в год...

Мудрый римлянин Публий Корнелий Тацит в свое время промолвил: «Можно восхищаться древностью, но следовать нужно современности». Он был совершенно прав, но сам остался в веках именно благодаря своим историческим трудам. «Вос-

хищение древностью» как ничто другое помогает «следовать современности». Например, если покопаться в истории науки и техники, то выясняется интересная вещь — почти все эпохальные изобретения и открытия делались дилетантами, а если и специалистами, то из другой области. Чаще всего это было обусловлено тем, что до них соответствующих направлений просто не существовало, и они вынуждены были все изобретать «с нуля». Крайний случай такого «любительства» — изобретение телеграфа Сэмюэлом Морзе (рис. 1), который вообще не имел никакого — даже самостоятельного — технического образования и до начала работы над телеграфом был признанным художником, в том числе основателем существующей и поныне Нью-Йоркской академии дизайна. Были и другие, не менее интересные случаи.

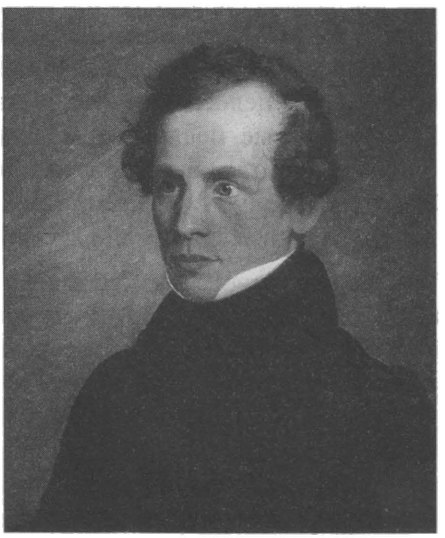


Рис. 1. Сэмюэл Морзе (Morse, Samuel F., 1791–1872), автопортрет, 1818 г.

В области электроники существует некая иерархия знаний, в которой неявно подразумевается, что при переходе от одной ступеньки к следующей знания предыдущего уровня усвоены в полной мере. Например, в радиолюбительской литературе сорокалетней давности вы могли встретить подробнейшее описание принципов работы аналого-цифровых преобразователей (АЦП), ибо их собирали из отдельных деталей, и без понимания их устройства такая работа просто не могла быть успешной. Сейчас АЦП всем доступны и в виде законченных микросхем, и в составе микроконтроллеров, управляются они программно, и, казалось бы, понимание принципов их функционирования уже никому не требуется — подавай нужные команды в соответствии с инструкцией, и все будет работать.

Это верно только на первый поверхностный взгляд. Никакие инструкции вам не заменят глубокого понимания принципов работы электронных компонентов. При условии, конечно, что вы хотите подойти к проектированию своих устройств творчески и что-то изобрести самостоятельно, а не тупо скопировать готовую конструкцию. В последнем случае вам эта книга не потребуется — вполне достаточно

сборников радиолюбительских схем, которые ныне доступны каждому через Интернет. Для особо ленивых многими фирмами выпускаются готовые конструкторы с подробной пошаговой инструкцией. Но вот как только вы захотите чужую схему приспособить к своим нуждам, модифицировать или хотя бы просто заменить детали более доступными и современными, без понимания принципов работы вы окажетесь в тупике.

Вообще говоря, без ряда фундаментальных знаний: от школьной физики до понимания того, как работают выходные каскады цифровых микросхем (а значит, и знания принципов работы транзисторов и других компонентов), — невозможно даже правильно подсоединить светодиод к микроконтроллеру. Можно скопировать чужое подключение, но никогда нет уверенности, что и сам автор схемы все сделал правильно. Развивая эту мысль, можно утверждать, что без знания закона Ома написать корректно работающую программу для Arduino можно только случайно, и пусть это не покажется вам таким уж большим преувеличением.

Поэтому мы дадим совет, который может показаться несколько неожиданным. Но сначала попробуйте мысленно ответить на один вопрос, каким бы идиотским он вам ни показался: «Какова величина тока в комнатной розетке?» То, что этот вопрос отнюдь не столь прост, как кажется, доказывают результаты ответов на него, полученные после опроса группы студентов одного технического вуза (по специальности, не связанной напрямую с электротехникой или электроникой), — из нескольких десятков опрошенных только двое смогли дать вразумительный ответ.

Итак, если вы, читатель, замялись с ответом или просто не уверены в нем, то вот обещанный совет, причем адресованный любому независимо от возраста, — прежде чем читать дальше, возьмите учебники физики за седьмой и восьмой классы и перечитайте главы, посвященные электричеству. Можете также захватить из учебника девятого класса главу, посвященную строению атома. Еще лучше обратиться не к обычным к школьным учебникам, а изучить соответствующие главы из учебника Г. С. Ландсберга [1], где то же самое изложено куда более увлекательно и подробно. Хотя ряд элементарных сведений и дается в первых главах книги, которую вы держите в руках, вам будет много легче читать ее дальше.

Как пользоваться книгой?

Несколько слов о том, как пользоваться книгой. Она рассчитана на тех, кто делает все своими руками и дома (поэтому, например, я не рекомендую компоненты для поверхностного монтажа — платы под них своими руками изготовить достаточно сложно, и еще труднее их отлаживать). Для тех, кто уже имеет известный опыт работы и оборудованную домашнюю лабораторию, первые главы можно пропустить, хотя *главу 6* о диодах и транзисторах я все же прочитать рекомендую. Книга отличается от большинства имеющихся руководств тем, что почти все описанные в книге схемы подробно рассмотрены шаг за шагом, включая самые мелкие детали, — так, чтобы при повторении конструкции у вас не возникало вопросов, зачем нужен тот или иной резистор и почему его сопротивление именно такое.

Однако, чтобы вопросов действительно возникало как можно меньше, при чтении книги следует проверять все рассуждения и расчеты с карандашом и калькулятором[1]. В процессе чтения вам, вероятно, придется возвращаться к уже освоенным главам. Логика построения книги подчинена принципу постепенного усложнения конструкций, но в то же время широко используется метод опережающего изложения — скажем, операционные усилители и обратная связь подробно рассматриваются только в *главе 12*, но примеры их использования вы сможете встретить уже в *главах 9* и *10*. Опыт показывает, что такой прием очень способствует усвоению информации — уже немного знакомое легче изучать подробно, чем незнакомое вовсе. С другой стороны, не удивляйтесь, что звуковой усилитель подробно рассмотрен в *главе 8*, а источник питания для него — только в следующей. Сведения из этих глав во многом пересекаются между собой, поэтому, прежде чем браться за реальный усилитель, вам придется прочесть обе главы.

Огромное большинство конструкций, описанных в книге, рассчитано на то, чтобы их использовать как готовые образцы в других проектах. Таковы, например, практически все проекты из раздела про микроконтроллеры и Arduino. Метеостанция, например, там взята как практический пример, позволяющий охватить в одном проекте очень много полезных узлов, которые могут использоваться в любых других приложениях этой платформы. При описании этих узлов автор старался представить как можно более широкую картину их разновидностей. Но это касается не только Arduino — в книге есть целые главы, посвященные описанию в основном заготовок (например, *глава 12* про операционные усилители или *глава 17* про применение цифровых микросхем). В этих главах тоже есть описания отдельных законченных конструкций, но и они сделаны с упором на то, чтобы разъяснить принципы построения подобных схем, а не просто тупо повторить их для конкретных целей.

Книга несамодостаточна — для того, чтобы повторить многие конструкции, вам потребуется дополнительный справочный материал, например, по разводке выводов транзисторов или иных компонентов, а для работы с последней частью книги, посвященной микроконтроллерам, обязательно нужно иметь фирменное описание микропроцессоров AVR и системы команд для них, а также справочник по языку Arduino. Слава Богу, у нас теперь есть Интернет, так что можно и не забивать шкаф справочниками, но кое-что все же потребуется иметь и на полке или, как минимум, в электронном виде на своем компьютере.

Не нужно рассматривать любую схему из этой книги или из других источников, как нечто вроде текста из Библии, который никому не позволено изменять. Даже рекомендации производителей по включению того или иного компонента есть всего лишь рекомендации. Можно и нужно экспериментировать, изменяя параметры компонентов и подгоняя схему к своим условиям, нелишне и поспорить с автором, если то или иное решение кажется чересчур усложненным. Однако стоит делать это разумно — если вы замените рекомендуемый операционный усилитель MAX474

[1] Одна никем не замеченная ошибка в расчетах выдержала 4 издания этой книги, пока на нее не указал один особо внимательный читатель. Так что, если что-то непонятно, пишите, буду очень благодарен!

похожим на него по выводам, но предназначенным совершенно для других целей MAX473, то можете быть весьма удивлены результатами.

Как разрабатывать электронные схемы?

И, наконец, рискуя утомить читателя, все же скажу несколько слов о том, как вообще следует разрабатывать и отлаживать схемы. Самый эффективный метод — сборка нужной схемы из готовых и заранее отлаженных фрагментов. Эта операция аналогична тому, как программисты собирают программы из готовых и заранее отлаженных библиотечных процедур. Каждая такая процедура есть «черный ящик», у которого имеются входы и выходы для общения с другими частями программы, причем в общем случае вы даже не знаете, как она устроена внутри, — точно так же, как вы не знаете, что именно размещается внутри микросхемы. Вы берете микросхему, подсоединяете к ней внешние элементы в соответствии с рекомендациями производителя и получаете готовый узел, который соединяете с другими подобными узлами.

> Повторим, что именно на этой стадии можно сильно «попасть», если вообще отказаться от попыток понять, как работают используемые узлы, и лишь тупо следовать рекомендациям производителя, которые по понятным причинам не исчерпывают всего разнообразия жизненных ситуаций. Лучше всего, если производитель предлагает не только описания компонентов (datasheets), но и рекомендации по их применению (application notes), — в этом случае их совсем не вредно изучить перед проектированием. Практика предоставления рекомендаций производителей по проектированию типовых узлов — давно уже не исключение, а правило, потому элементарное знание технического английского становится обязательным условием для любого более-менее грамотного радиолюбителя. Для облегчения преодоления этого порога в конце книги приведен словарик специфических терминов и аббревиатур (см. *приложение 4*).

При рисовании схемы обязательно обозначайте на ней конкретные типы и значения параметров элементов — не откладывайте это до выполнения практической ее отладки. Изменить эти параметры вы всегда сможете, но все, что можно посчитать, нужно посчитать заранее — это сохранит вам очень много времени. Когда вы берете, наконец, паяльник в руки, то не следует сразу собирать всю схему устройства целиком. Разбейте ее на как можно более мелкие самостоятельно работающие узлы и отлаживайте каждый узел по отдельности. Не верьте печатному слову и все рекомендации из литературы проверяйте на макетах (в конце концов у вас образуется библиотека таких самостоятельно отлаженных узлов, и вы будете экономить огромное количество времени). Отладив все, обязательно нанесите на чертеж схемы полученные в результате отладки точные значения компонентов (те, что все еще требуют окончательной подгонки «по месту», обозначаются звездочкой), проверьте правильность соединения этих узлов и разводку питания и лишь затем собирайте всю схему целиком — сначала на макетной плате. И только убедившись в работоспособности макета схемы, переносите ее на настоящую рабочую плату.

В отличие от большинства радиолюбительских изданий, рисунки плат в книге не приводятся, чему есть много причин: во-первых, повторить конструкцию в точности с теми компонентами, которые приведены в описании, как правило, не получа-

ется, да это и совершенно не требуется. Во-вторых, лично я никогда не повторял опубликованных конструкций в точности, стараясь улучшить или упростить схему, и в этой книге вы почти всегда найдете рекомендации по улучшению характеристик или расширению функциональности описанного прибора, так что публикация рисунка платы вообще теряет смысл. Наконец, есть и еще один момент, скорее психологический — раскладывая плату самостоятельно, вы намного лучше вникаете в работу схемы, после чего отладка и регулировка ее значительно упрощаются. Мое глубокое убеждение состоит в том, что плату нужно делать самостоятельно, под выбранную конструкцию и корпус, а не подгонять габариты под имеющуюся плату — в результате такой подгонки самодельные конструкции иногда получаются весьма уродливыми. Исключением будут схемы на основе готовых узлов (таких, например, как Arduino и аксессуары к нему — см. *главы 20–22*), но там раскладывать платы самостоятельно чаще всего и не требуется.

Если вы разрабатываете серьезный прибор, который должен служить годами, — постарайтесь заложить в разработку время и деньги, необходимые для выполнения следующих этапов:

- **разработка технического задания** с возможно более подробным описанием требуемой функциональности. Не пренебрегайте мелочами, особенно если вы работаете «на сторону», а не для себя — так, будет очень печально, если вы собрали и проверили прибор дома на столе, а потом выяснится, что он должен работать круглогодично на улице;
- **разработка принципиальной схемы** с отладкой отдельных узлов на макетах;
- **изготовление полного макета** и его отладка;
- **разработка окончательной принципиальной схемы**, подбор деталей и разработка печатной платы;
- **изготовление опытного образца** и его отладка, корректировка печатной платы;
- **изготовление окончательного варианта** печатной платы, корпуса и монтаж прибора.

Отдельно стоит остановиться на составлении технического описания и инструкции по эксплуатации. Я знаю, что большинство разработчиков искренне ненавидит этот этап работы (то же относится, увы, и к программистам), но советую себя пересилить и начинать составление документации прямо сразу, одновременно с началом проектирования. Значительная часть из проектов по Arduino, имеющихся в Сети, страдает отсутствием более-менее внятного описания, довольно часто даже не имеет принципиальной схемы. Учтите, что модная наглядная схема на макете, созданная в пакетах типа Fritzing, ни в коем случае принципиальную схему не заменяет! Подобные проекты почти бесполезны для использования вне рамок того конкретного применения, для которого были созданы, и даже такое их использование весьма затруднено, — отладка схемы, о которой ничего не известно, превращается в сущий кошмар. Не берите с них пример! Во-первых, при попытке описать словами «как это работает» в расчете на стороннего читателя обычно всплывают все недостатки и упущения. Ладно еще любительские Arduino-проекты — иногда на примере некоторых вполне промышленных изделий бытовой техники (как и компьютерных программ, кстати) отчетливо видно, что их разработчики сами никогда и не пытались взглянуть на свое творение с точки зрения того, кто это будет применять на практике, а инструкцию по эксплуатации писали наспех совершенно другие люди.

Во-вторых, с уверенностью можно сказать, что через пару лет вы напрочь забудете, как у вас работал тот или иной узел, и почему выбраны именно такие компоненты. Поэтому написание технической документации нужно вам самим не меньше, чем пользователю.

Приведенный идеальный вариант последовательности разработки редко осуществим на практике — либо времени не хватает, либо денег, либо того и другого. Есть одна известная фирма, которая занимается разработкой заказных электронных устройств, — так там берут несколько килобаксов только за написание технического задания. И они правы! Но на практике часто получается так, что макетный либо опытный образец становится и окончательным изделием. И все же по мере возможности не пренебрегайте этими промежуточными этапами — поверьте, так получится намного быстрее, чем, зажмурившись, собрать все сразу, а потом в лучшем случае обнаружить, что ничего не работает, а в худшем — выветривать из комнаты очень неприятный и стойкий запах горелой пластмассы. Учтите, что почти ни одна незнакомая дотоле схема никогда не работает сразу, — будьте к этому готовы и заранее наберитесь терпения.

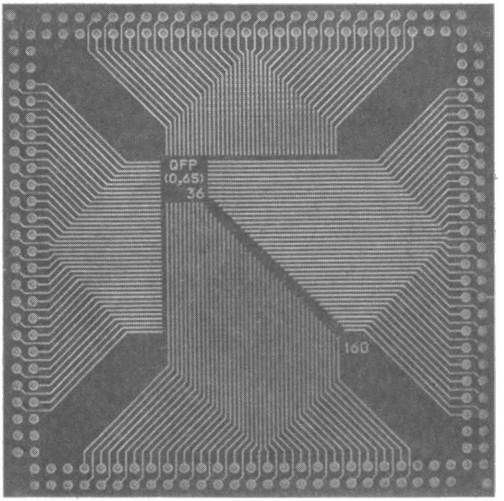
Итак, приступим.

Юрий Ревич, **revich@lib.ru**

* * *

Автор выражает искреннюю благодарность администрации интернет-магазина «Амперка» (**amperka.ru**) за поддержку подготовки глав, посвященных платформе Arduino.

Схемы, чертежи и фотографии компонентов подготовлены автором. Все остальные иллюстрации взяты из источников, допускающих свободное копирование, за исключением фотографии первого транзистора из *главы 6* и портрета Клода Шеннона из *главы 14*, любезно предоставленных автору корпорацией Lucent Technologies Inc./Bell Labs в лице ее сотрудницы Франциски Мэттьюз (Francisca Matthews).



ЧАСТЬ I

Основы основ

Глава 1. Чем отличается ток от напряжения?

Глава 2. Джентльменский набор
Оборудуем домашнюю лабораторию

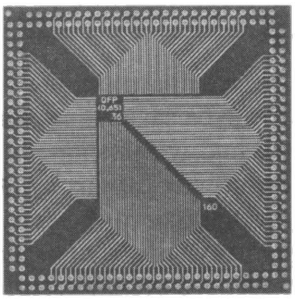
Глава 3. Хороший паяльник — половина успеха
Инструменты и технологические советы

Глава 4. Тригонометрическая электроника
О частотах, периодах, мощности, переменных напряжениях и токах и немного о сигналах

Глава 5. Электроника без полупроводников
Резисторы, конденсаторы и схемы на их основе

Глава 6. Изобретение, которое потрясло мир
Диоды, транзисторы и простейшие схемы на их основе

Глава 7. Ошеломляющее разнообразие электронного мира
Реле, стабилитроны, светодиоды...



ГЛАВА 1

Чем отличается ток от напряжения?

Поэтому-то мне будет легче, facilius natans, взять тему по моему выбору, которая для этих трудных вопросов богословия явилась бы тем же, чем мораль является для метафизики и философии.

А. Дюма. «Три мушкетера»

Дурацкий вопрос, скажете вы? Отнюдь. Опыт показал, что не так уж и много людей могут на него ответить правильно. Известную путаницу вносит и язык: в выражении «имеется в продаже источник постоянного тока 12 В» смысл искажен. На самом деле в этом случае имеется в виду, конечно, источник напряжения, а не тока, поскольку ток в вольтах не измеряется, но так говорить не принято. Самое правильное будет сказать — «источник питания постоянного напряжения 12 вольт», а написать можно и «источник питания =12 В» где символ «=» обозначает, что это именно постоянное напряжение, а не переменное. Впрочем, и в этой книге мы тоже иногда будем «ошибаться» — язык есть язык.

Чтобы разобраться во всем этом, для начала напомним строгие определения из учебника (зазубривать их — весьма полезное занятие!). Итак, *ток*, точнее, его величина, *есть количество электрического заряда, протекающее через сечение проводника за единицу времени*: $I = Q/t$. Единица тока называется «ампер», и ее размерность в системе СИ — кулоны в секунду. Знание сего факта пригодится нам позднее.

Куда более запутанно выглядит определение напряжения: *величина напряжения есть разность электрических потенциалов между двумя точками пространства.* Измеряется она в вольтах, и размерность этой единицы измерения — джоуль на кулон, т. е. $U = E/Q$. Почему это так, легко понять, вникнув в смысл строгого определения единицы напряжения: *1 вольт есть такая разность потенциалов, при которой перемещение заряда в 1 кулон требует затраты энергии, равной 1 джоулю.*

Все это наглядно можно представить себе, сравнив проводник с трубой, по которой течет вода (и это будет довольно точная аналогия). При таком сравнении величину тока можно себе представить, как количество (расход) протекающей воды за се-

кунду, а напряжение — как разность давлений на входе и выходе трубы. Чаще всего труба заканчивается открытым краном, так что давление на выходе равно атмосферному давлению, и его можно принять за нулевой уровень. Точно так же в электрических схемах существует общий провод (или «общая шина» — в просторечии для краткости ее часто называют «землей», хотя это и не точно — мы еще вернемся к этому вопросу позднее), потенциал которого принимается за ноль и относительно которого отсчитываются все напряжения в схеме. Обычно (но не всегда!) за общий провод принимают минусовый вывод основного источника питания схемы.

Итак, вернемся к вопросу, сформулированному в заголовке, — так чем же отличается ток от напряжения? Правильный ответ будет звучать так: ток — это количество электричества, а напряжение — мера его потенциальной энергии. Неискушенный в физике собеседник, разумеется, начнет трясти головой, пытаясь вникнуть, и тогда можно дать такое пояснение. Представьте себе падающий камень. Если он маленький (количество электричества мало), но падает с большой высоты (велико напряжение), то он может наделать столько же несчастий, сколько и большой камень (много электричества), но падающий с малой высоты (напряжение невелико).

Связь тока и напряжения

На самом деле аналогия с камнем наглядна, но не точна, — труба с текущей жидкостью подходит куда больше. Дело в том, что напряжение и ток обычно связаны между собой. (Слово «обычно» я употребил потому, что в некоторых случаях — в источниках напряжения или тока — от этой связи стараются избавиться, хотя полностью никогда и не удается.) В самом деле, если вернуться к нашей трубе, то легко представить, как с увеличением давления (напряжения) увеличивается количество протекающей жидкости (ток). Иначе зачем нужны были бы насосы? Сложнее представить себе наглядно обратную зависимость — как ток влияет на напряжение. Для этого нужно сначала понять, что такое *сопротивление*.

Вплоть до середины XIX века физики не знали, как выглядит зависимость тока от напряжения. Этому есть одна важная причина. Попробуйте сами экспериментально выяснить, как выглядит график этой зависимости. Схема эксперимента приведена на рис. 1.1, а примерные результаты — на рис. 1.2. Показанные на графике результаты весьма приблизительны, т. к. реальный вид кривой сильно зависит от того, как именно выполнен подопытный проводник («П» на рис. 1.1), — плотно или редко он намотан, на толстый массивный каркас или на тонкий стакан из бумаги, а также от температуры в комнате, наличия сквозняка и еще от множества других причин. Именно такое непостоянство и смущало физиков — не только сама кривая загибается (т. е. ток в общем случае непропорционален напряжению), но вид и форма этого загиба весьма непостоянны и меняются как при изменении условий внешней среды, так и для различных материалов.

Понадобился гений Георга Ома (рис. 1.3), чтобы за всеми этими деревьями увидеть настоящий лес — а именно понять, что зависимость тока от напряжения описывается элементарно простой формулой: $I = U/R$, а все несуразности, упомянутые ранее, проистекают от того, что величина сопротивления R зависит от материала

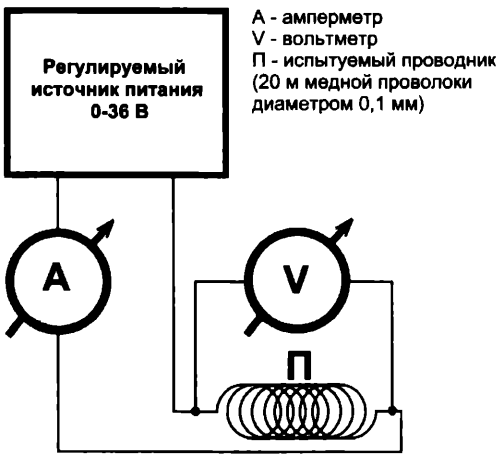


Рис. 1.1. Схема эксперимента по проверке закона Ома

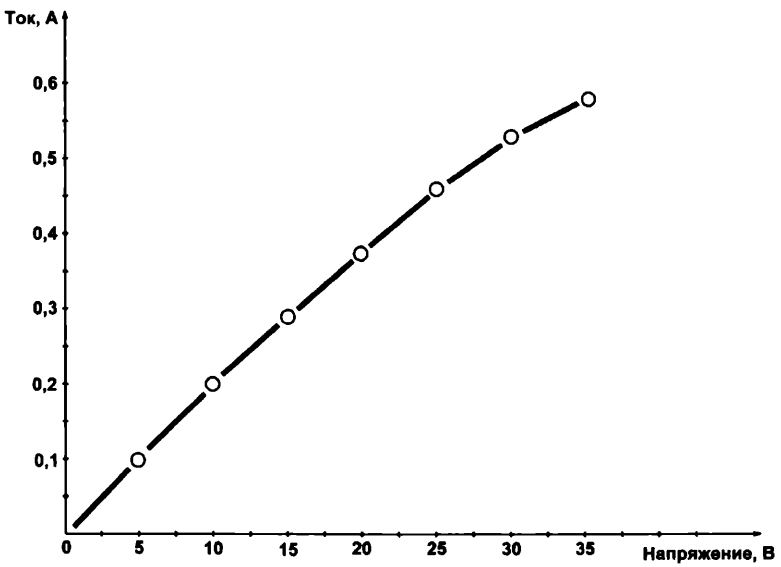


Рис. 1.2. Примерные результаты проверки закона Ома

проводника и от условий внешней среды, в первую очередь от температуры. Так, в нашем эксперименте загиб кривой вниз происходит потому, что при прохождении тока проводник нагревается, а сопротивление меди с повышением температуры увеличивается (примерно на 0,4% с каждым градусом изменения температуры). А вот сама величина этого нагрева зависит от всего, чего угодно, — намотайте провод поплотнее и заверните его в асбест — он будет нагреваться сильнее, а размотайте его и поместите на сквозняк — нагрев резко уменьшится.

В знак признания заслуг Георга Ома единица измерения сопротивления так и называется — ом (ohm по-английски). Согласно формуле, приведенной в предыдущем

абзаце, *1 Ом есть сопротивление такого проводника, через который течет ток в 1 А при напряжении на его концах, равном 1 В.* Обратная сопротивлению величина называется *проводимостью* и измеряется в сименсах, названных так в честь другого немецкого ученого, электротехника и предпринимателя Вернера фон Сименса: 1 См = 1/Ом. В электронике почти всегда оперируют именно величиной сопротивления, так что сименсы мы оставим для физиков.

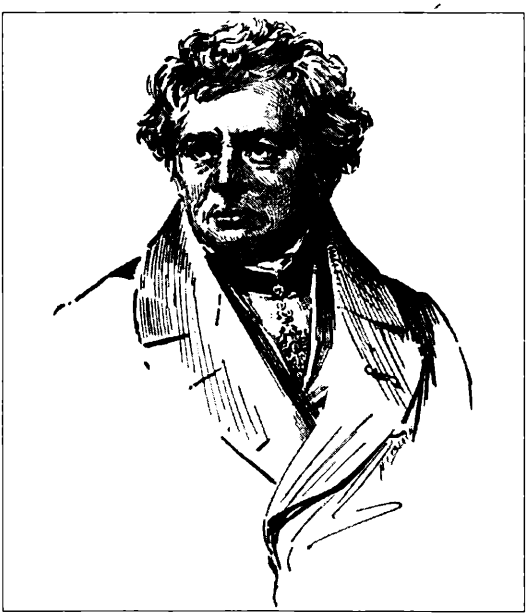


Рис. 1.3. Георг Симон Ом (1787–1854) — немецкий физик, член-корреспондент Берлинской АН. Окончил Эрландский университет. Преподавал математику, затем физику в различных гимназиях. С 1833 года — профессор Нюрнбергской высшей политехнической школы. В 1849–52 годах — ректор Мюнхенского университета. Член Лондонского королевского общества

Заметки на полях

Обратите внимание, что *название* единицы измерения «ом» мы пишем со строчной буквы, а ее *обозначение* («Ом») — с прописной. Это общее правило — все обозначения единиц измерения, которые образованы от фамилий ученых, пишутся с прописной буквы: Дж (джоуль), Вт (ватт), В (вольт), А (ампер). В то же время обозначения единиц измерения, которые не образованы от имен собственных, а являются обычными словами, пишутся со строчной буквы: с (секунды), м (метры).

Сопротивление проводника зависит от его геометрических размеров — оно увеличивается пропорционально длине и уменьшается пропорционально площади сечения: $R = \rho \cdot L/S$. Большое практическое значение имеет коэффициент пропорциональности, обозначаемый буквой ρ (ро), — так называемое *удельное сопротивление* материала проводника. При определенной температуре (обычно берется 20 °С) эта величина почти постоянна для каждого материала. «Почти» я тут написал потому, что на самом деле эта величина сильно зависит от химической чистоты и даже от способа изготовления материала проводника (например, формировался ли проводник штамповкой, протяжкой или электрохимическим напылением). Для проводни-

ков стараются употреблять очень чистые металлы, скажем, обычный медный провод изготавливают из меди с количеством примесей не более 0,1% (как говорят, с чистотой в «три девятки»), — это позволяет уменьшить сопротивление такого провода и избежать лишних потерь на его нагрев.

Удельное сопротивление проводника, по определению, есть сопротивление (Ом) проводника единичной площади (1 м^2) и единичной длины (1 м), и если подставить эти величины в формулу, приведенную ранее, вы получите размерность для удельного сопротивления: Ом·м^2/м или просто Ом·м. Практически в таких единицах измерять удельное сопротивление страшно неудобно, т. к. для металлов величина получается крайне маленькой — представляете сопротивление куба меди с ребром в 1 м?! На практике часто употребляют единицу в 100 раз больше: Ом·см. Эта величина часто приводится в справочниках (см., например, [2]), но и она не слишком удобна для практических расчетов. Так как диаметр проводников измеряют обычно в миллиметрах (а сечение, соответственно, в квадратных миллиметрах), то на практике наиболее удобна старинная внесистемная единица Ом·мм^2/м, которая равна сопротивлению проводника сечением в 1 квадратный миллиметр и длиной 1 метр. Для того чтобы выразить официальный Ом·м в этих единицах, нужно умножить его величину на 10^6, а для Ом·см — на 10^4. Подглядев в справочнике величину удельного сопротивления меди (0,0175 Ом·мм^2/м при 20°С), мы легко можем вычислить, что сопротивление проводника с параметрами, приведенными на рис. 1.1, составляет около 45 Ом (проверьте!).

Надо сказать, что человечество весьма преуспело в изготовлении специальных материалов, имеющих коэффициент удельного сопротивления, мало зависящий от температуры. Это прежде всего специальные сплавы — константан и манганин — температурный коэффициент сопротивления (ТКС) которых в несколько сотен раз меньше, чем у чистых металлов. А для обычных стандартных углеродистых или металлопленочных резисторов ТКС составляет приблизительно 0,1% на градус, т. е. примерно в 4 раза лучше, чем у меди (причем, отметьте, что у резисторов он может быть как положительным, так и отрицательным, в отличие от металлов). Есть и специальные прецизионные резисторы (среди отечественных это, например, С2-29В, проволочные С5-54В и др.), у которых этот коэффициент значительно меньше. Есть и другие материалы, у которых температурный коэффициент, наоборот, весьма велик (несколько процентов на градус, и при этом, в отличие от металлов, отрицателен) — из них делают так называемые *термисторы*, которые используют как чувствительные датчики температуры (подробнее о них см. *главу 13*). Однако для точного измерения температуры все же используют чистые металлы — чаще всего платину и медь. К этому вопросу мы еще вернемся.

Регулирование тока с помощью сопротивления

Познакомившись, таким образом, с понятием сопротивления и его особенностями, вспомним, для чего мы все это делали. Ах да, мы же хотели понять, как практически представить зависимость напряжения от тока! Но ведь мы пока не умеем про-

извольно изменять ток в проводнике, так? Напряжение изменять просто — нужно взять регулируемый источник питания, как это изображено на рис. 1.1, или, на худой конец, набор батареек, при последовательном соединении которых (одной, двух, трех и т. д.) мы получим некий набор напряжений. А вот источников тока (именно тока, а не напряжения) мы еще не имеем. Как же быть?

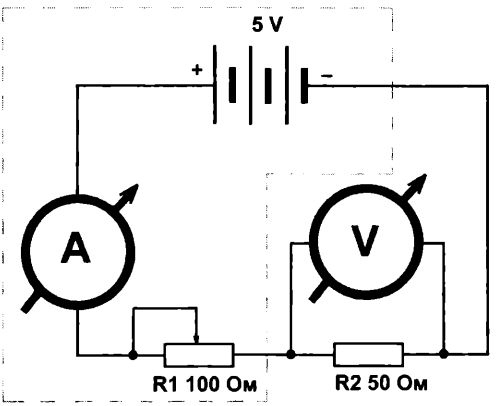


Рис. 1.4. Схема для изучения свойств цепи с двумя резисторами

Выход из этой ситуации показан на рис. 1.4 (заметьте, мы от схематического изображения проводника, как катушки из длинной проволоки, имеющей некое сопротивление, перешли к стандартному изображению резисторов, как это делается в настоящих электрических схемах). Здесь мы уже не используем регулируемый источник питания — он нам не нужен. Питается схема от батареи из трех гальванических элементов, например, типоразмера D, соединенных последовательно. Каждый такой элемент (если он еще не был в эксплуатации) дает напряжение примерно 1,62 В, так что суммарное напряжение будет почти 5 В, как и указано на схеме (под нагрузкой и по мере истощения элементов напряжение немного упадет, но ошибка в нашем случае не играет большой роли).

Как работает эта схема? Допустим, что движок переменного резистора (подробнее о них рассказано в *главе 5*) R1 выведен в крайнее правое (по схеме) положение. Проследим путь тока от плюсового вывода батареи: амперметр, вывод движка резистора R1, крайний правый вывод R1, резистор R2, минусовый вывод батареи. Получается, что резистор R1 в схеме как бы не участвует — ток от плюсового вывода батареи сразу попадает на R2 (амперметр, как мы узнаем из *главы 2*, можно не принимать во внимание), и схема становится фактически такой же, как на рис. 1.1. Что покажут наши измерительные приборы? Вольтметр покажет напряжение батареи — 5 В, а показания амперметра легко вычислить по закону Ома: ток в цепи составит 5 В/50 Ом = 0,1 А или 100 мА. На всякий случай еще раз напомним, что эти значения приблизительные — реальное напряжение батареи несколько меньше 5 В.

Теперь поставим движок R1 в среднее положение. Ток в цепи теперь пойдет от плюса батарей через амперметр, вывод движка R1, половину резистора R1, резистор R2 и далее, как и раньше, вернется к минусу батареи. Как изменятся показания

приборов? Раньше резистор R1 в процессе не участвовал, а теперь участвует, хотя и половинкой. Соответственно, общее сопротивление цепи станет уже не 50 Ом (один резистор R2), а 50 (R2) + 50 (половинка R1), т. е. 100 Ом. Амперметр покажет уже не 100 мА, а 5 В/100 Ом = 0,05 А или 50 мА — в два раза меньше.

А вот что покажет вольтметр? Так сразу и не скажешь, не правда ли? Придется посчитать, и для этого рассмотрим отдельно участок цепи, состоящий из R2 с присоединенным к нему вольтметром. Очевидно, что току у нас деться некуда, — все то количество заряда, которое вышло из плюсового вывода батареи, пройдет через амперметр, через половинку R1, через R2 и вернется в батарею[1]. Значит, и на этом отдельном участке, состоящем из одного R2, ток будет равен тому, что показывает амперметр — т. е. 50 мА. Получается, как будто резистор R2 подключен к источнику тока! И это действительно так — источник напряжения с последовательно включенным резистором (в нашем случае половинкой R1) представляет собой *источник тока* (хотя и плохой, как мы увидим в дальнейшем). Так каковы же будут показания вольтметра? Очень просто: из закона Ома следует, что $U = I \cdot R$, где R — сопротивление нужного нам участка цепи, т. е. R2, и в нашем случае вольтметр покажет $0{,}05 \cdot 50 = 2{,}5$ В. Эта величина называется *падением напряжения*, и в нашем случае — падением напряжения на резисторе R2. Легко догадаться, даже не подключая вольтметр, что падение напряжения на резисторе R1 будет равно[2] тоже 2,5 В, причем его можно вычислить двумя путями: как разницу между 5 В от батареи и падением на R2 (2,5 В) или по закону Ома, аналогично расчету для R2.

А что будет, если вывести движок переменного резистора R1 в крайнее левое положение? Я сразу приведу результат: амперметр покажет 33 мА, а вольтметр — 1,66 В. Пожалуйста, проверьте это самостоятельно! Если вы получите те же значения, то это будет означать, что вы усвоили закон Ома и теперь умеете отличать ток от напряжения. А более сложными схемами включения резисторов мы займемся в *главе 5*.

Прежде чем пойти дальше, я хочу ответить на вопрос, который, несомненно, у вас уже возник: а как можно себе представить резистор в нашей аналогии с водопроводной трубой? Ток — расход воды, напряжение — давление, а сопротивление? Оказывается, это очень просто, мало того, соответствующая величина в гидравлике называется точно так же: *сопротивление потоку*. Аналогами резисторов будут всякие устройства, установленные на трубе: краны, задвижки, муфты или просто местные сужения. Определенным сопротивлением обладает и сама труба, причем чем она тоньше, тем ее сопротивление выше, в точности как и с проводником. И точно так же, как на включенном в электрическую цепь резисторе происходит падение напряжения, на этих гидравлических сопротивлениях происходит падение давления, которое пропорционально величине самого сопротивления. Прикрывая кран,

---

[1] На самом деле это не совсем точно — часть тока, хотя и очень небольшая, все же пойдет через вольтметр, минуя R2. Но на практике этим всегда пренебрегают (подробности см. в *главе 2*).

[2] И это не совсем точно — амперметр тоже имеет некоторое сопротивление и может быть представлен, как еще один последовательный резистор. Но, как и в случае с вольтметром, этим на практике пренебрегают.

мы увеличиваем сопротивление потоку, и расход воды уменьшается, т. е. здесь мы производим в точности то же действие, что и при экспериментах с переменным резистором в электрической цепи.

Источники напряжения и тока

В схеме на рис. 1.4 мы можем выделить, как показано пунктиром, ее часть, включив туда батарейку и переменный резистор R1. Тогда этот резистор (вместе с сопротивлением амперметра, конечно) можно рассматривать как *внутреннее сопротивление* источника электрической энергии, каковым выделенная часть схемы станет для нагрузки, роль которой будет играть R2. Любой источник, как легко догадаться, имеет свое внутреннее сопротивление (электронщики часто употребляют выражение *выходное сопротивление*) — хотя бы потому, что у него внутри есть провода определенной толщины.

Но на самом деле не провода служат ограничивающим фактором. В *главе 4* мы узнаем, что такое электрическая мощность в строгом значении этого понятия, а пока, опираясь на интуицию, читатель может сообразить сам: чем мощнее источник, тем у него меньше должно быть свое внутреннее сопротивление, иначе все напряжение «сядет» на этом внутреннем сопротивлении, и на долю нагрузки ничего не достанется. На практике так и происходит — если вы попытаетесь запустить от набора батареек типа AA какой-нибудь большой прибор, питающийся от источника с низким напряжением (вроде настольного сканера или ноутбука), то прибор, конечно, не заработает, хотя формально напряжения должно хватать, — однако это напряжение «сядет» почти до нуля. А вот от автомобильного аккумулятора, который гораздо мощнее, все получится как надо.

Такой источник, у которого внутреннее сопротивление мало по отношению к нагрузке, называют еще *идеальным источником напряжения*. Физики предпочитают название *идеальный источник ЭДС* (электродвижущей силы), но на практике это абстрактное понятие используется реже, чем менее строгое, но всем понятное «напряжение». К ним относятся в первую очередь все источники электрического питания: от батареек до промышленной сети. Кстати, для снижения внутреннего сопротивления вовсе не обязательно увеличивать мощность источника напряжения до бесконечности — к тому же эффекту приводят специальные меры по стабилизации напряжения, с которыми мы познакомимся в *главе 9*.

Наоборот, *идеальный источник тока*, как нетрудно догадаться, обязан обладать бесконечным внутренним сопротивлением — только тогда ток в цепи совсем не будет зависеть от нагрузки. Понять, как источник реального тока (не бесконечного малого) может обладать бесконечным выходным сопротивлением, довольно трудно, и в быту такие источники вы не встретите. Однако уже обычный резистор, включенный последовательно с источником напряжения (не тока!), как R1 на рис. 1.4, при условии, что сопротивление нагрузки мало (R2<<R1), может служить хорошей моделью источника тока. Еще ближе к желаемому будут транзисторы в определенном включении, и мы с этим разберемся позднее. А с помощью обратной связи и операционных усилителей можно добиться и результатов, близких к теоре-

тическому идеалу, — построить источник тока конечной величины с выходным сопротивлением, весьма близким к бесконечности.

Источники напряжения и тока обозначаются на схемах так, как показано на рис. 1.5, *а* и *б*. Не перепутайте — логики в этих обозначениях немного, но так уж принято. А так называемые *эквивалентные схемы* (их еще называют *схемами замещения*) реальных источников приведены на рис. 1.5, *в* и *г*, где $R_в$ обозначает внутреннее сопротивление источника.

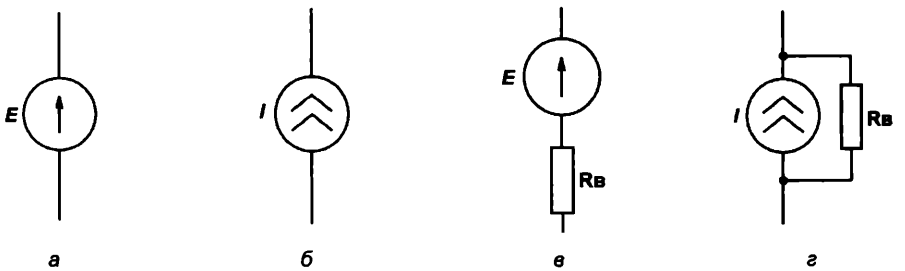
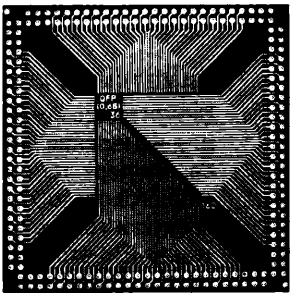


Рис. 1.5. Источники тока и напряжения: *а* — обозначение идеального источника напряжения; *б* — обозначение идеального источника тока; *в* — эквивалентная схема реального источника напряжения; *г* — эквивалентная схема реального источника тока

Теперь нам придется отвлечься от схем и разобраться с тем, как оборудовать себе рабочее место. Теоретические знания — это важно, но на практике все познается куда лучше.



ГЛАВА 2

Джентльменский набор

Оборудуем домашнюю лабораторию

> Затем он сходил на набережную Железного Лома и отдал приделать новый клинок к своей шпаге.
>
> *А. Дюма. «Три мушкетера»*

Скажу сразу: если у вас среди добрых родственников нет олигархов, то оборудовать домашнюю лабораторию по последнему слову техники и не пытайтесь, — она встанет в такую сумму, что домашние еще долго будут вам это вспоминать. Потому делайте все постепенно — конечно, очень заманчиво купить цифровую измерительную станцию (мультиметр, запоминающий осциллограф, генератор сигналов и т. п. в одном флаконе) стоимостью примерно как подержанный импортный автомобиль в хорошем состоянии, но, уверяю, что для дела это необязательно. Есть, однако, вещи, без которых не обойтись, вот их-то мы и приведем, а затем рассмотрим по очереди.

Итак, вот малый джентльменский набор радиолюбителя:

- мультиметр;
- источник питания постоянного тока (два одинаковых однополярных или один двуполярный, иногда требуется и третий источник);
- осциллограф;
- генераторы прямоугольных и синусоидальных сигналов.

Указанного достаточно, чтобы повторить все примеры из этой книги, однако в общем случае может понадобиться и иное оборудование (например, если вы собираетесь заняться радиоприемом или телевидением). Но мы остановимся на этом. Причем источники питания и генераторы несложно сделать самому (генераторы — даже предпочтительнее, т.к. они обойдутся вам в день труда и сотню-другую рублей, потраченных на компоненты, а работать будут не хуже, а в некоторых отношениях даже лучше промышленных, и окажутся много удобнее в использовании). В дальнейшем мы все это сделаем, а пока рассмотрим то, что имеется в продаже.

Заметки на полях

Внимательный читатель может заметить, что в списке нет частотомера — прибора для измерения частоты колебаний. В обычной практике он действительно не требуется. Во-первых, его с успехом заменяет осциллограф, — даже с помощью простого аналогового осциллографа можно оценить период колебаний с достаточной для практики точностью и притом намного нагляднее, чем это может сделать любой частотомер. Во-вторых, функцию измерения частоты имеют многие современные мультиметры. И наконец, откалиброванные по точным значениям выходной частоты генераторы сигналов разной формы делают частотомер уж и вовсе ненужным. Тем не менее если он и понадобится, то в задачах, где требуется знать значение частоты с точностью намного большей, чем могут дать упомянутые только что средства (от десятых долей процента для цифровых приборов до единиц процентов для мультиметра и самодельного генератора из *главы 12*). А подобный образцовый прибор, для которого в строке «цена» обычно указывается «по запросу», покупать для домашней лаборатории так же нецелесообразно, как приобретать в личное пользование электричку.

Мультиметр

Его еще часто называют по старинке тестером. Покупать нужно обязательно цифровой и не самый дешевый. На рынке полно миниатюрных китайских мультиметров стоимостью в несколько сотен рублей — их покупать не надо! Как правило, они работают примерно неделю, а потом если не разваливаются, то просто перестают показывать. Но самый главный их недостаток — даже не низкая надежность, а недостаточная точность, которая может быть довольно далека от декларируемых в описании величин.

В магазинах типа «Чип и Дип» или на радиорынках есть большой выбор приличных универсальных мультиметров. Нас интересует средний ценовой диапазон — высокая цена означает, что в приборе наличествуют не слишком актуальные навороты: связь с компьютером, память или еще что-то в этом роде. Выбирать нужно по количеству функций и диапазонам измерений. Укажем главные функции, без которых не обойтись (в скобках — желательные диапазоны):

- измерение постоянного напряжения (2 мВ–600 В);
- измерение постоянного тока (2 мА–10 А);
- измерение переменного напряжения (1–700 В);
- измерение переменного тока (20 мА–10 А);
- измерение сопротивления (100 Ом–10 МОм, функция «прозвонки»).

Все остальные функции, как говорят компьютерщики, опциональны, но пренебрегать ими не следует (располагаю их по степени практической нужности):

- измерение емкости;
- измерение частоты;
- измерение индуктивности;
- измерение параметров транзисторов;
- измерение температуры.

Мультиметры работают, как правило, от 9-вольтовой батарейки в типоразмере «Крона». Батарейки хватает надолго, однако у некоторых моделей было замечено, что при снижении напряжения питания ниже допустимого мультиметр не выключается, а начинает врать. Это может стать источником неприятностей и крупных недоразумений, поэтому я советую сразу после покупки разориться на самую дорогую щелочную (alkaline) батарейку фирмы Duracell, Varta или Energizer, гарантированный срок хранения которой составляет 6–7 лет. В последние годы появились литиевые аналоги «Кроны», которые примерно втрое дороже щелочных, но для этой цели они подойдут еще лучше — у литиевых срок хранения доходит до 10–12 лет. Это будет неплохим вложением денег, учитывая, что батарейка в мультиметре в основном хранится. Этот совет, кстати, относится не только к мультиметрам, а вообще к любой технике, которая должна долго питаться от гальванических элементов, — скажем, к пультам управления телевизорами.

Совет

При покупке мультиметра обратите внимание на наличие функции автовыключения (для экземпляра, изображенного на рис. 2.1, о присутствии такой функции говорит надпись «Auto power off»). Большинство современных мультиметров ее имеют, но может быть, при некотором опыте обращения с ними вы предпочтете, чтобы ее не было: такой мультиметр имеет привычку выключаться посреди работы как раз в тот момент, когда у вас заняты руки. По крайней мере, перед покупкой стоит ознакомиться с описанием того, как реализована эта функция.

Слабым местом всех мультиметров является функция измерения тока, т. е. режим амперметра. Если вы творчески изучили *главу 1*, то должны легко сообразить,

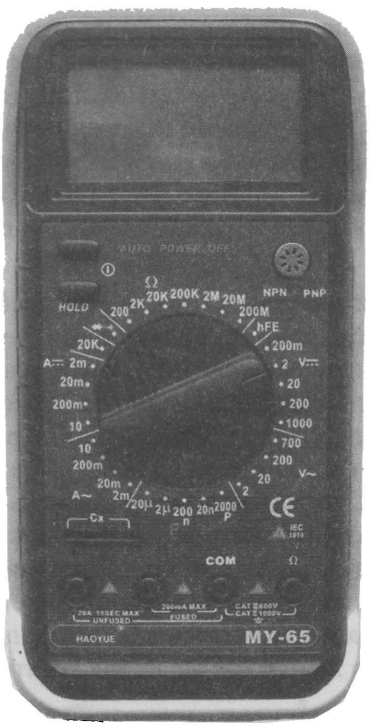


Рис. 2.1. Типовой мультиметр с дополнительной функцией измерения емкости

что амперметр, включаемый всегда последовательно с нагрузкой, должен иметь очень маленькое собственное (внутреннее) сопротивление, иначе на нем образуется большое падение напряжения, что внесет искажения в измеряемую величину тока. Если вы все же не поняли предыдущую фразу, то вернитесь к рис. 1.4 и перерисуйте его на листочке бумажки так: вместо амперметра нарисуйте резистор R_A (который будет представлять упомянутое внутреннее сопротивление амперметра), а переменный резистор R1 исключите. Вы получите почти точно такую же картинку, как и раньше, только вместо R1 у вас теперь есть R_A, и все расчеты будут совершенно аналогичными: чем больше R_A, тем меньшее падение напряжения покажет вольтметр на R2, и тем меньше будет общий ток в цепи. Это очень плохо, т. к. оба прибора должны только измерять, но никак не участвовать в процессах, происходящих в цепи. В принципе полностью избежать влияния амперметра не удается, но получается сделать внутреннее сопротивление амперметра достаточно малым, чтобы пренебречь его влиянием.

Вот это-то замечательное свойство амперметров одновременно и является их самым слабым местом — достаточно перепутать и включить амперметр не последовательно, а параллельно источнику питания (подобно вольтметру на рис. 1.1), как через него, в полном соответствии с законом Ома, потечет огромный ток, ограниченный только возможностями источника. Действительно — характерное сопротивление амперметра при измерении больших токов составляет порядка нескольких миллиом, что даже при 5-вольтовом источнике дает токи в 1000 А и более! На самом деле никакой обычный источник питания (включая даже бытовую электросеть) такого тока отдать не сможет, но того, что сможет, будет достаточно, чтобы прибор сгорел.

Однако не отчаивайтесь — обычно в хороших мультиметрах на этот случай внутри стоит плавкий предохранитель, а самых «продвинутых» — даже самовосстанавливающийся. Поэтому если ваш прибор вдруг перестал показывать ток (а вы можете и не заметить, как случайно подсоединили его в режиме измерения тока к выводам питания), то прежде всего откройте его и проверьте этот самый предохранитель. Кстати, именно для того, чтобы дополнительно защитить мультиметр от описанных неприятностей, клемму для подключения щупа в режиме измерения тока всегда делают отдельной от других.

Теперь немного о режиме вольтметра. От вольтметра, наоборот, требуется максимально высокое сопротивление, иначе часть тока будет проходить через него, т. е. мимо измеряемого участка, что эквивалентно уменьшению суммарного сопротивления этого участка (подробнее о параллельном включении резисторов мы поговорим в *главе 5*, а пока, чтобы понять сказанное, поглядите еще раз на рис. 1.4). Итак, чтобы вольтметр не вносил искажения, его сопротивление делают максимально большим. И это хорошо: по крайней мере, нет такой опасности сжечь прибор, как в случае амперметра, — если вы включите мультиметр в режиме вольтметра в цепь последовательно с источником питания, ток просто не пойдет (точнее, пойдет, но очень маленький). Зато вольтметр можно испортить, если включить его на предел в 1 В, а подсоединить к сети 220 В. Надо сказать, что обычно современные вольтметры выдерживают такое издевательство, но лучше все же не рисковать и соблюдать следующее правило:

Обязательно заранее устанавливайте прибор на тот диапазон, в котором вы собираетесь производить измерения! Если примерное значение величины заранее неизвестно, то следует установить прибор на максимальный диапазон.

Впрочем, если финансовые возможности позволяют, то можно и приобрести мультиметр с автоматическим выбором пределов измерения, — в этом случае вам придется заботиться только о том, чтобы не перепутать, что именно вы собрались измерять: ток или напряжение. Такими функциями всегда обладают настольные мультиметры, и они, конечно, для наших целей во всем предпочтительнее универсальных, — если бы только не цена, которая для самых дешевых моделей начинается от 5 тыс. рублей.

Кстати, а можно ли обойтись одним вольтметром, если вдруг амперметр сгорел необратимо? Вполне можно. Соорудить амперметр из вольтметра — пара пустяков (как, кстати, и наоборот — просто обычно этого не требуется). Для этого нужно запастись точным резистором с номинальным значением сопротивления, например, ровно 1 Ом (это подойдет для измерения токов в единицы-десятки-сотни-тысячи миллиампер, что есть обычное значение для наших схем, для токов в других диапазонах нужен больший или меньший номинал). Мощность этого резистора должна быть как можно выше — не менее 1–2 Вт, а если достанете мощностью в 10 Вт, то это будет просто прекрасно.

Совет

Если не сможете приобрести такой резистор, то в крайнем случае можно его изготовить самостоятельно. Лучше всего использовать нихромовую проволоку толщиной 0,2–0,4 мм. Из подобной проволоки раньше делались спирали для отечественных утюгов и электроплиток, а также паяльников, но сейчас, кроме разве что паяльников, вы такого в продаже не найдете и запчастей к этому антикварату тоже. Проще всего приобрести такую проволоку в интернет-магазинах, торгующих аксессуарами к электронным сигаретам — из нее делаются спирали испарителей. Возьмите кусок этой проволоки подлиннее (как минимум, несколько метров) и измерьте его сопротивление (его-то ваш мультиметр еще не потерял способность измерять?). Величина эта составит порядка нескольких десятков ом. Затем растяните и измерьте рулеткой длину. Поделив одно на другое, вы получите сопротивление проволоки в Ом/м. Осталось отмерить (как можно точнее) нужный кусок, соответствующий одному ому (скорее всего получится порядка 4–12 см), свернуть его спиралькой, намотав на карандаш и затем аккуратно сняв, — и сопротивление (причем достаточно точное) готово. Чтобы оно было более долговечным, следует взять любой крупный резистор достаточно высокого сопротивления (более нескольких сотен ом), намотать на него этот отрезок нихрома так, чтобы витки не касались друг друга, и туго обернуть его кончики вокруг имеющихся выводов. Затем их следует пропаять с помощью кислотного флюса (см. *главу 3*).

Теперь включите этот одноомный резистор последовательно в измеряемую цепь, а параллельно ему подключите вольтметр, как показано на рис. 2.2. Величина тока

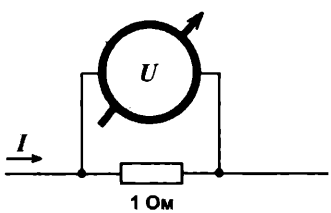


Рис. 2.2. Схема измерения величины тока с помощью вольтметра

в цепи будет численно равна показаниям вольтметра (если вольтметр установлен на диапазон в вольтах, то, значит, ток в амперах, если в милливольтах — то в миллиамперах). Думаю, вы легко сообразите, почему это так, и сами сможете придумать, как подсчитать напряжение, если резистор имеет номинал, отличающийся от 1 Ом.

> ***Совет***
>
> Одновременно с мультиметром приобретите подходящие для него зажимы «крокодилы» (с запасом — они легко теряются). С ними ситуация обычно такая: чаще всего стандартные отечественные «крокодилы» не подходят, а «родные» импортные слишком громоздкие и очень плохо держатся на измеряемом проводе. В этом случае рекомендуется либо, если возможно, подогнуть отечественные, чтобы они держались на импортном щупе, либо просто откусить импортные наконечники от проводов и заменить их отечественными штекерами, на которых «крокодил» держится очень крепко. При этом вы лишаетесь фирменного заостренного щупа, но его легко изготовить из «крокодила», причем он будет даже лучше оригинального — плотно зажмите с помощью плоскогубцев (а еще лучше пропаяйте мощным паяльником) в «крокодиле» толстую швейную иглу и натяните на нее изолирующую кембриковую трубку, оставив свободными только несколько миллиметров кончика иглы (чтобы плотно держалось, лучше использовать термоусадочный кембрик). Сам «крокодил» тоже нужно изолировать — можно использовать более толстый термоусадочный кембрик или обмотать липкой эластичной полихлорвиниловой лентой (но не скотчем!). Этот же щуп удобно использовать для осциллографа.

Более подробно об особенностях проведения измерений в различных случаях мы поговорим в соответствующих главах, а сейчас продолжим обсуждать оборудование нашей лаборатории.

Источник питания

Лабораторный источник питания, как я уже говорил ранее, не представляет особых трудностей сделать самому (см. *главу 9*), но вы пока этого не умеете, а хотя бы один источник понадобится сразу — например, для того, чтоб отладить собственные. Поэтому его следует приобрести. Можно, конечно, приобрести и три источника, но к собственноручной сборке я призываю не столько в целях экономии денег, и даже не из педагогических соображений, но еще и потому, что собранный нами источник будет именно таким, какой нам надо. Если мультиметры (о которых мы говорили в предыдущем разделе) и осциллографы (о которых пойдет речь далее) не имеет никакого смысла собирать самому, потому что лучше и дешевле промышленных вы наверняка не сделаете, то источники питания и генераторы — совсем другое дело.

Заметим, что во время отладки простых устройств Arduino, не содержащих много потребляющих компонентов, можно вообще обойтись без отдельного источника — плата питается от компьютера через порт USB. Как временный выход из положения можно рекомендовать источники, встроенные в сетевую вилку с фиксированным напряжением на выходе. Если вы занимаетесь одним лишь Arduino, то для начала работы только один такой источник и понадобится — стабилизированный или даже нестабилизированный на напряжение 7–9 В и ток 1–1,5 А, с круглым разъемом диаметром 5,5 мм на выходе (рис. 2.3, *слева*). Подобный источник следует покупать у тех же продавцов, что торгуют Arduino-модулями: в «Чипе-Дипе», «Амперке»

или аналогичных магазинах. Некачественный дешевый источник, купленный у случайного торговца, может обойтись слишком дорого.

Совет

Не выбрасывайте адаптеры от отслужившей свой срок мобильной техники! Чаще всего они имеют выходное напряжение около 5 вольт, которое вполне годится для многих радиолюбительских целей. Чтобы не возиться с ни к чему не подходящими разъемами, корпус такого адаптера аккуратно, чтобы не задеть внутренности, распиливается по периметру ножовкой по металлу, затем плата отсоединяется от сетевых контактов. Получаете готовый модуль питания от сети, который обычно имеет весьма высокие характеристики: так, некоторые фирмы прилагают к своим мобильникам импульсный источник 5 В с выходным током до 1,5 А, который практически не греется и не потребляет лишней энергии. Речь тут идет о типах зарядников, которыми комплектовала свои телефоны, например, старая «Nokia». Эти телефоны сейчас как раз начинают выбрасываться, и их зарядники никуда больше не годятся. Я как минимум половину своих радиолюбительских надобностей обеспечиваю за счет такого бесплатного источника.

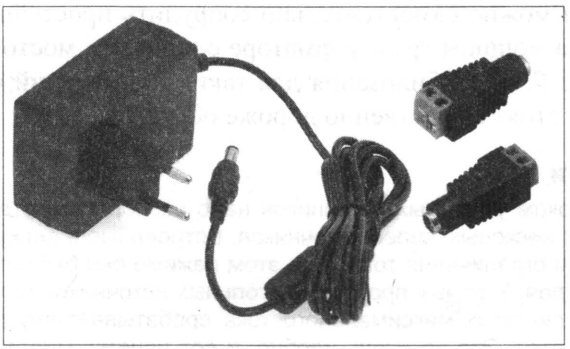


Рис. 2.3. Источник, встроенный в вилку (*слева*), и ответные разъемы для него с клеммной колодкой (*справа*)

Для подключения подобного источника к беспаечной макетной плате можно использовать гнездо разъема 5,5 мм с установленной прямо на нем клеммной колодкой для зажима проводов (рис. 2.3, *справа*). Напряжение 3,3 В, требующееся для некоторых модулей, обычно обеспечивают сами платы Arduino или модули расширения для них — «шилды» (shields).

Но уже для подключения различных нагрузок может понадобиться источник с другим выходным напряжением и/или мощностью, а для ряда работ потребуется отдельный стабилизированный источник 5 или 3,3 В. В Arduino имеется свой встроенный стабилизатор 5 В, и именно поэтому вход внешнего питания Vin нужно подключать к более высокому напряжению (см. *главу 20*). Так что одним источником, даже если не выходить за рамки Arduino, вы все равно не обойдетесь. А для большинства применений за пределами платформы Arduino (которым и посвящена значительная часть этой книги), понадобится отдельный стабилизированный источник, лучше с двумя независимыми регулируемыми выходами как минимум от 3 до 12 В (обычно это просто два источника в одном корпусе, которые можно соединить по клеммам).

В продаже имеются регулируемые стабилизированные источники с набором выходных напряжений от 3 до 12 вольт в исполнении «встроенные в вилку». На крайний случай можно приобрести два таких источника, хотя пользоваться ими крайне неудобно. Кроме того, ползунковый переключатель на таких источниках при частом пользовании быстро выходит из строя. Потому предпочтительно либо соорудить двухполярный источник самостоятельно (см. *главу 9*), либо приобрести готовый в настольном исполнении.

Если будете приобретать готовый, то учтите, что напряжение на выходе должно регулироваться не плавно, а ступенчато, что позволяет устанавливать точное значение напряжения без дополнительного контроля. Настольный источник лучше выбирать с диапазоном от нуля до 20–24 В (заодно он может служить для подключения микродрели). Что же касается выходного тока, то смотрите по средствам, но в любом случае желательно иметь источник как можно мощнее: 1–2 А нам будет, пожалуй, хватать для большинства наших экспериментов, но запас иметь никогда не помешает.

Для бóльших токов можно самостоятельно соорудить простейший нестабилизированный источник на мощном трансформаторе с диодным мостом и конденсатором, как описано в *главе 9*, — стабилизация для таких применений обычно не важна, а мощные источники стоят существенно дороже обычных.

Подробности

С выходным током у готовых источников надо разобраться отдельно — дело в том, что существует несколько типов источников. Встроенные в вилку источники чаще всего имеют режим ограничения тока, но в этом режиме они быстро перегреваются и могут выйти из строя. У самых простых настольных источников при превышении указанного в характеристиках максимального тока срабатывает внутренняя защита, и нагрузка отключается. Это не очень удобно, и вот почему: многие нагрузки, включая и определенную часть приведенных в этой книге устройств, в момент включения в течение некоторого времени (от миллисекунд до секунд) потребляют ток, значительно больший номинального потребления (обычно он принимается, как минимум, равным удвоенному номинальному току). Объясняется это тем, что в первый момент либо заряжается фильтрующий конденсатор источника питания, либо, как в случае, скажем, подключения микродрели, большой ток призван раскрутить движок до номинальных оборотов, после чего потребление тока резко снижается. Если используется источник с ограничением тока, то это приводит просто к замедлению пуска нагрузки, а если с отключением нагрузки, то отключение это чаще всего успевает сработать до того, как потребляемый ток снизится до приемлемого уровня. В результате вы попадаете в замкнутый круг — источник номинально может обслуживать вашу нагрузку, а фактически включить ее не получается.

Более «продвинутые» источники ограничивают ток на указанном уровне (т. е. фактически превращаются в источник тока, а напряжение при этом зависит от нагрузки), и иногда еще умеют отключать нагрузку вовсе, если в ней наблюдается короткое замыкание. Такие источники гораздо удобней. Чаще всего величину тока ограничения можно тоже, как и напряжение, регулировать, потому подобные источники сразу можно отличить по внешнему виду — в них имеются отдельные ручки регулирования напряжения и тока (часто даже по две на каждый параметр: для грубой и точной установки).

Для подключения мощной нагрузки с заданием предельного значения тока удобно также использовать источник для зарядки автомобильных аккумуляторов. Только нужно выбрать обязательно такой, в котором имеется ручка-регулятор и амперметр для ручной установки значения тока — полностью автоматические зарядники здесь не подойдут. Перед покупкой проверьте, чтобы в паспорте было указано «можно использовать, как многоцелевой блок питания для электронных приборов». В пределах нагрузки от нуля до установленного значения тока такой источник выдает примерно 15 вольт с хорошей стабильностью. Только следует учесть, что у некоторых старых конструкций зарядников напряжение на выходе нефильтрованное (пульсирующее), поэтому на входе вашей схемы по выводам питания должен быть установлен конденсатор достаточно большой емкости.

Осциллограф

Осциллограф — это вещь почти незаменимая. Если вкратце, то это прибор, который позволяет увидеть на экране все, что происходит с напряжением в наших схемах. Мало того, во многих случаях он может заменить и мультиметр. Но одновременно это будет и самое дорогое ваше приобретение. Портативный прибор с ЖК-экраном можно приобрести за несколько тысяч рублей, но вы быстро убедитесь, что пользоваться таким устройством не слишком удобно, а цена настольных конструкций начинается от 10 тысяч рублей и уходит далеко в бесконечность. Потому выбор предстоит только между традиционным аналоговым и современным цифровым прибором. Цифровые осциллографы имеют намного больше возможностей, и о них мы поговорим далее в этой главе. А начнем с подробного рассмотрения обычного аналогового прибора, потому что в базовых функциях с точки зрения пользователя они не различаются, а устройство традиционного осциллографа все равно полезно изучить каждому знатоку электроники.

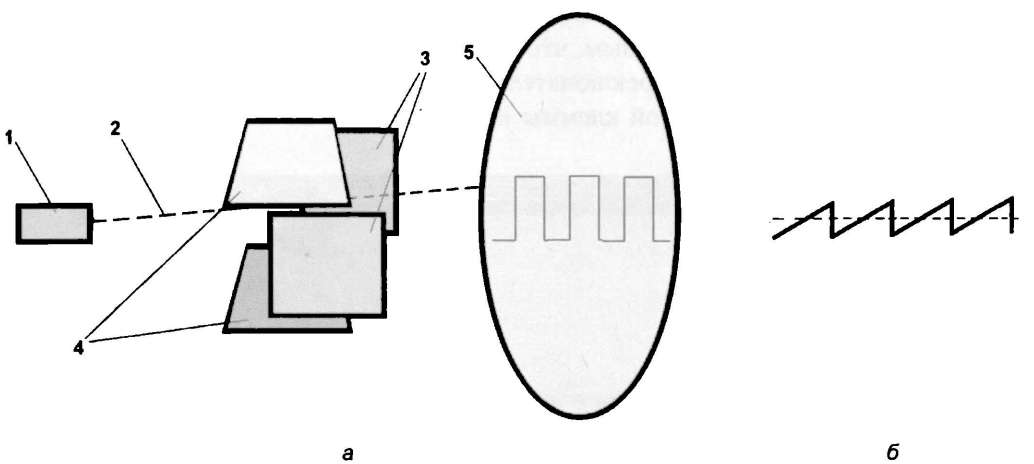


Рис. 2.4. *а*) принцип работы осциллографической трубки: 1 — электронная пушка; 2 — электронный луч; 3 — горизонтальные отклоняющие пластины X; 4 — вертикальные отклоняющие пластины Y; 5 — экран с люминофором; *б*) форма напряжения развертки на пластинах X

Как работает простейший аналоговый осциллограф? Главная его деталь — электронно-лучевая трубка (cathode ray tube, CRT), аналогичная той, что используется в старинных черно-белых телевизорах, только носящая специальное название «осциллографической». Если не углубляться в физику, то принцип ее работы можно пояснить картинкой, показанной на рис. 2.4, *а*. В узком конце трубки расположена так называемая *электронная пушка*, излучающая узкий поток электронов, разогнанных до большой скорости. Попадая на экран, покрытый изнутри люминофором, он образует маленькую светящуюся точку (в отличие от телевизионных трубок, где люминофор светится белым, здесь часто используется люминофор зеленого или, например, синего свечения, причем обычно с небольшим послесвечением — луч как бы оставляет за собой постепенно затухающий след). Пластины X служат для развертки луча по горизонтали — на них подается пульсирующее напряжение пилообразной формы (его график приведен на рис. 2.4, *б*). В результате в отсутствие напряжения на пластинах Y луч прочерчивает горизонтальную линию, начинающуюся у края экрана слева и заканчивающуюся у его правого края.

Если теперь подать напряжение на пластину Y, связанную через регулируемый усилитель со входом осциллографа, то луч будет сдвигаться вверх или вниз в зависимости от знака поданного напряжения, рисуя на экране график, соответствующий изменениям формы входного напряжения во времени. В простейшем случае, если на вход Y подано постоянное напряжение, отличающееся от нуля, то линия просто сдвинется, но останется горизонтальной. Поверх экрана размещается координатная сетка, по которой можно узнать все характеристики видимого сигнала: его размах в вольтах и период изменения во времени (о периодах сигналов говорится в *главе 4*).

Основных управляющих ручек у простого аналогового осциллографа как минимум четыре (см. фото панели малогабаритного осциллографа С1-73 на рис. 2.5). Две ручки, обычно помеченные стрелочками вверх-вниз и вправо-влево (на рис. 2.5 сверху по обе стороны экрана), позволяют устанавливать линию развертки в отсутствие сигнала в нужное начальное положение — например, по центру экрана. Для того чтобы при этом быть уверенным, что сигнала действительно нет, обычно имеется специальная кнопка или переключатель, помеченная символом «земли», которая отключает вход Y от входной клеммы и замыкает его на корпус, соединенный

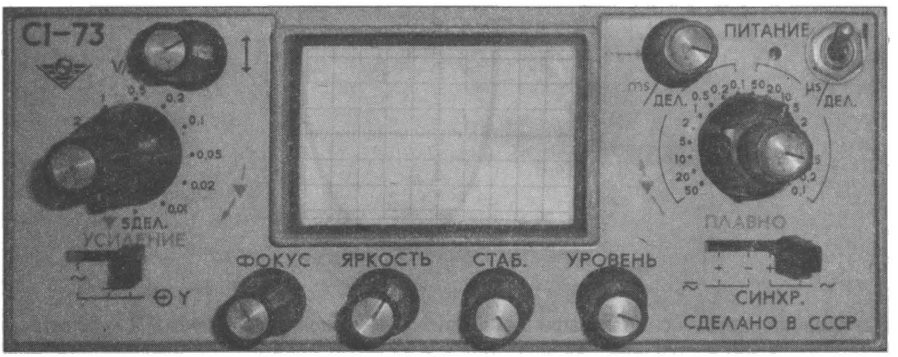


Рис. 2.5. Панель малогабаритного аналогового осциллографа С1-73

с общим проводом («землей») прибора. На рис. 2.5 это переключатель слева внизу под надписью «Усиление» — в среднем положении он как раз и замыкает вход Y на «землю».

Две другие ручки, представляющие собой переключатели с большим числом фиксированных положений, позволяют управлять временем развертки (на рис. 2.5 справа от экрана) и усилителем входного сигнала (слева от экрана), для того чтобы увидеть сигнал в удобном масштабе по обеим осям. У этих переключателей против каждого положения написаны значения времени (для развертки) и напряжения (для усилителя Y), которые соответствуют одному делению координатной сетки экрана. Таким образом, осциллографом можно довольно точно измерять период (частоту) сигнала и его размах. Поверх этих переключателей выступают ручки, позволяющие плавно менять заданную величину развертки или усиления вблизи установленного переключателями значения (точному установленному значению соответствует крайнее правое положение этих ручек).

Кроме этих основных управляющих элементов, обычно имеются еще вспомогательные: для управления яркостью луча и его фокусировкой (т. е. размерами пятна на экране) — они видны на рис. 2.5 внизу. Часто есть специальная кнопка под названием «поиск луча» (в С1-73 она отсутствует) — дело в том, что луч запросто может уехать за пределы экрана, и вам по неопытности даже сперва покажется, что все сломалось.

Подробности

Отдельного разговора заслуживает также обязательно присутствующая регулировка синхронизации: на рис. 2.5 это две ручки с подписями «СТАБ.» (стабильность) и «УРОВЕНЬ», а также переключатель справа внизу с подписью «СИНХР.» (синхронизация). Дело в том, что на практике частота развертки никогда точно не кратна частоте сигнала, поэтому сигнал «бежит» по экрану, не давая как следует рассмотреть его форму и измерить параметры. Регулировка синхронизации служит для того, чтобы сигнал можно было «остановить» — при этом начало развертки (т. е. начало хода луча от левого края экрана) будет всегда совпадать с каким-то характерным моментом в изменении повторяющегося сигнала — например, с переходом через ноль или максимумом напряжения по спаду или по фронту сигнала. Эти параметры и регулируются указанными элементами управления. Переключатель синхронизации выбирает форму сигнала (постоянный, переменный, по положительному или отрицательному фронту), а для того чтобы сигнал «остановить», обычно используют такой прием: следует вывести ручку регулировки уровня в минимум, затем ручку регулировки стабильности установить в состояние полного пропадания сигнала и медленно поднимать уровень, пока сигнал опять не появится. Во многих простейших аналоговых осциллографах, подобных С1-73, синхронизация работает плохо, и установить ее — занятие, требующее большого практического опыта. Хорошо помогает в этом случае дополнительный вход для синхронизации от внешнего сигнала, который имеется в большинстве даже самых простых моделей (в С1-73 он расположен сбоку корпуса). Заметим, что у цифровых осциллографов, в силу принципиально иного механизма отображения на экране, эти проблемы отсутствуют.

Конечно, во многих моделях могут быть и другие органы управления — скажем, кнопка для переворота (инвертирования) сигнала, или «лупа» для выделения интересного участка, или клемма для подачи пилообразного напряжения развертки от внешнего источника, но вы с ними легко разберетесь по ходу дела.

Проверить осциллограф просто — надо схватиться рукой за щуп, и тогда вы увидите на экране наведенную помеху от бытовой электросети частотой 50 Гц. Если вы ее не видите, 99% за то, что вы забыли отключить заземление входа после настройки положения луча (такое часто случается).

С электрической точки зрения осциллограф представляет собой вольтметр, т. е. имеет высокое входное сопротивление (стандартно — 1 МОм, хотя есть специальные высокочастотные осциллографы, которые имеют входное сопротивление 50 Ом, — естественно, они для наших целей не годятся), поэтому наводка от сети и других помех может быть весьма значительна. Если такое входное сопротивление все же слишком мало (что бывает при исследовании схем с очень малыми токами), то следует использовать прилагаемый к осциллографу или приобретаемый отдельно щуп с делителем 1:10. При этом входное сопротивление возрастает соответственно до 10 МОм, а величину сигнала на экране нужно умножить на 10. Этим же щупом следует пользоваться, если требуется исследовать сигналы высокого напряжения, например, сетевого (220 В), т. к. обычно имеющейся шкалы не хватает, и большая часть сигнала сверху и снизу при использовании простого щупа окажется за пределами экрана. Производители не рекомендуют насиловать входной усилитель такими высокими напряжениями, и, хотя лично мне ни разу не удавалось сжечь вход осциллографа, все же к рекомендациям изготовителей нужно прислушиваться.

При подсоединении щупа к исследуемой схеме нужно помнить, что корпус осциллографа «заземлен», т. е. соединен с общим проводом щупа, потому он не должен касаться корпусов источников питания и других приборов, — довольно часто бывает, что нужно разглядеть сигнал не относительно общего провода схемы, а, скажем, относительно шины питания.

Внимание!

При исследовании узлов, напрямую связанных с бытовой сетью 220 В, нужно соблюдать особую осторожность: осциллограф обязательно должен стоять на сухом изолирующем основании, ни в коем случае нельзя касаться каких-либо металлических предметов (скажем, корпуса стоящего рядом компьютера), и за его металлические части, включая элементы щупа, ни в коем случае нельзя браться руками! Если вам придется проводить подобные операции, то последовательность их проведения такая:

- отключить питающее схему напряжение (обязательно оба сетевых провода);
- надежно подсоединить щуп к измеряемой схеме, используя зажимы «крокодилы» и следя за тем, чтобы они не касались проводников и держались как можно прочнее;
- включить напряжение, держа руки подальше, и наблюдать сигнал;
- при необходимости изменения параметров развертки, усиления и синхронизации внимательно следить за тем, чтобы в ажиотаже не задеть зажимы «крокодилы» и не коснуться металлических частей корпуса;
- при необходимости перенести щупы в другое место схемы снова полностью выключить питание и повторить операции.

Все здесь сказанное относилось к простейшим аналоговым осциллографам. Их функций вам будет хватать во всех случаях, описанных в этой книге. Ранее выпускались и более навороченные аналоговые модели: многолучевые и многоканаль-

ные, а также запоминающие осциллографы, которые позволяют получить моментальный «снимок» одноразового процесса, скажем, всплеска напряжения в схеме. В настоящее время все цифровые осциллографы имеют эти функции, и еще много полезных сверх того, потому если уж выбирать аналоговую модель, то только простейшую и исключительно из-за низкой цены.

Принцип работы цифрового осциллографа основан на очень быстрой, с частотой в десятки и сотни мегагерц, оцифровке входного сигнала (см. *главу 17*). Отобразить полученный массив цифр в виде графика на дисплее — задача тривиальная и хорошо отработанная в компьютерной технике. Так как сам цифровой осциллограф уже есть маленький компьютер, его ничего не стоит обучить многим вещам: показывать в цифровой форме напряжение и время в указанной точке, анализировать спектры, производить над осциллограммами математические операции, сохранять график в памяти, передавать данные в «большой» компьютер и т. п. Примерами таких устройств могут служить весьма продвинутые и относительно недорогие модели цифровых осциллографов Owon китайской фирмы Lilliput, самая дешевая модель из которых — SDS5032E с предельной частотой сигнала 30 МГц — с избытком годится для любых любительских и большинства профессиональных целей.

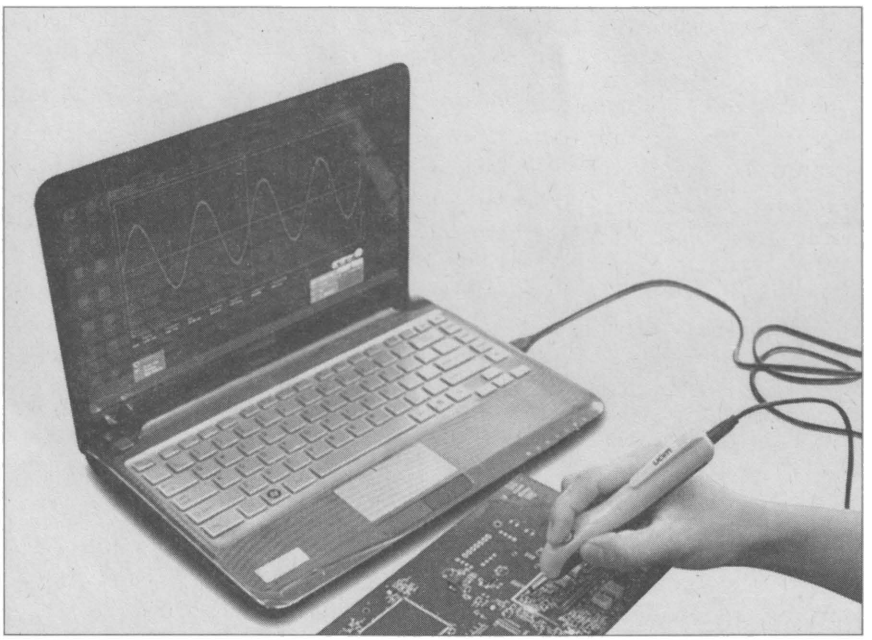
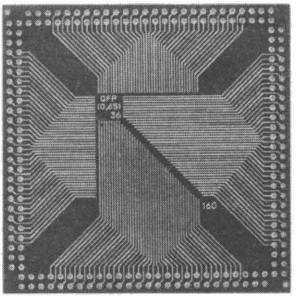


Рис. 2.6. Осциллограф-ручка

В качестве отличного компромисса можно приобрести цифровой осциллограф-приставку к компьютеру, часто также называемую *осциллографом-ручкой* (Pen-type Oscilloscope, рис. 2.6). «Ручка» подключается к порту USB, позволяет наблюдать сигнал на большом экране ПК и управлять настройками с помощью мыши. Обладая всеми характеристиками довольно продвинутых цифровых моделей (выборка, пиковый детектор, усреднение, память на несколько осциллограмм, полоса пропус-

кания в десятки мегагерц), осциллограф-приставка в разы дешевле настольных цифровых моделей, и по стоимости не превышает самые простые аналоговые образцы. При этом я очень не советую городить самостоятельно и тем более приобретать самодельные конструкции такого рода — в деньгах, может быть, вы и выиграете, но времени на доводку изделия «до ума» потратите немерянно, если вам вообще это удастся: осциллограф — прибор, простой только по принципу работы, а определяющая часть его функциональности заключается во всяких тонкостях и нюансах, дилетантскому подходу не поддающихся.

Ну, а теперь перейдем к технологическому оснащению нашей домашней лаборатории.



ГЛАВА 3

Хороший паяльник — половина успеха

Инструменты и технологические советы

> На следующий день д'Артаньян поднялся в пять утра, сам спустился в кухню, попросил достать ему кое-какие снадобья, точный список которых не дошел до нас, к тому же еще вина, масла, розмарину, и, держа в руке рецепт, данный ему матерью, изготовил бальзам, которым смазал свои многочисленные раны, сам меняя повязки и не допуская к себе никакого врача.
>
> *А. Дюма. «Три мушкетера»*

Не будет преувеличением утверждать, что стабильность и покой в нашей жизни основываются на мелочах. Отсутствие всего двух-трех термозащитных плиток из десятков тысяч у шаттла «Коламбия» привело семерых астронавтов к гибели. Невероятное стечение обстоятельств, противоречащее всем законам вероятности, — и пятьдесят башкирских детей гибнут в небе над швейцарским озером. Есть такой эмпирический закон, известный под названием «закона Мерфи», который известен во множестве вариантов, но его основную мысль можно сформулировать так: всегда полагайтесь на худший из возможных исходов. В моей практике этот закон не нарушался никогда — например, если некий прибор сломался, то обязательно следует предполагать, что поломка произошла как минимум в двух местах. И это невероятное предположение, противоречащее основным положениям теории надежности, обычно оправдывается на практике!

Я веду вот к чему — чем больше мелочей вы предусмотрите заранее, тем надежнее будут работать ваши приборы. Кстати, в радиоэлектронике также в полной мере оправдывается правило, которое заметили еще авиаконструкторы: красивый самолет имеет и лучшие летные качества. Аккуратно и эстетично смонтированный прибор будет работать лучше и надежнее — этому можно, кстати, отыскать вполне рациональные объяснения. Например: если у вас соединительные провода между блоками имеют произвольную длину и толщину и кое-как запиханы в корпус прибора, напоминая мочалку для мытья посуды, то велика вероятность того, что вы зацепите тот или иной провод при сборке, и он просто оторвется. А если этот про-

вод слишком толстый и жесткий, то и цеплять не надо, — пайка отломится при малейшей попытке отогнуть провод в сторону. Наоборот, слишком тонкий и мягкий провод будет цепляться за все подряд и обязательно попадет под крепежные винты. Кроме того, слишком длинные и хаотично расположенные проводники могут в некоторых случаях привести к неработоспособности схемы из-за самовозбуждения.

Ни в коем случае не берите за образец сборку настольных компьютеров — там совершенно другая технологическая база, и спроектировано все настолько надежно, что хаотичное расположение кабелей в корпусе уже помешать работоспособности не может, разве что провода попадут в вентилятор. Хотя в фирменно собранных ПК кабели все же убирают в аккуратные жгуты. «На коленке» такого уровня достичь сложно, потому берите лучше пример со старой отечественной военной сборки, которая немногим отличалась от «наколеночной», но, тем не менее, все же довольно надежно работала.

Для того чтобы монтаж и сборка были на уровне, оборудовать свою домашнюю лабораторию надо как можно лучше. Это не значит, что нужно покупать самые дорогие фирменные инструменты. Вовсе не так — сторублевые (в современном исчислении) китайские[1] пассатижи-утконосы служат автору верой и правдой уже двадцать лет. И отечественный паяльник с деревянной рукояткой будет исполнять свои функции ничуть не хуже импортного. Правда, то же самое нельзя сказать про дешевые сверла или напильники, но сейчас нам важно другое — применяемый инструмент должен точно соответствовать той операции, для которой мы его используем. Если мы попытаемся припаять провод к толстому стальному стержню с помощью 18-ваттного паяльника с жалом, заточенным под распайку выводов микросхем, и с использованием канифоли в качестве флюса, то, помучившись с полчасика, мы, возможно, добьемся своего, но гарантии, что пайка не отвалится, если дернуть за провод посильнее, не будет.

Инструменты и материалы

Далее — (неполный) список того, что желательно бы иметь всегда под рукой. Я исключил из него слесарные инструменты вроде тисков (в том числе ювелирных), обычных отверток (в том числе со сменными жалами разной формы) и напильников (в том числе алмазных надфилей), приведя лишь позиции, специфичные для процесса монтажа и отладки собственно схем (хотя и напильники в этом процессе иногда тоже участвуют, не говоря уж об отвертках). Не пугайтесь столь длинного перечня — даже он не исчерпывает всех необходимых мелочей, но в большинстве своем это недорогие (кроме электроинструмента) и легко доступные вещи. Чуть позже мы рассмотрим некоторые позиции подробнее.

---

[1] В этой книге не раз еще будут упомянуты различные изделия «Made in China» — как с положительной интонацией, так и с отрицательной. Прошу китайских товарищей не обижаться на последнюю — в Китае действительно делают образцы как самых плохих, так и самых качественных товаров в мире. Более того — в последние годы стало появляться все больше уникальных товаров, которые разработаны и выпускаются *только* в Китае.

- Инструменты:
  - три паяльника разной мощности с подставкой;
  - микродрель с набором цанговых зажимов (0,5–2 мм) и набором сверл (0,5–2,0 мм);
  - обычная электродрель с патроном от 1,5 до 13 мм;
  - врачебный пинцет;
  - бокорезы;
  - малогабаритные пассатижи (утконосы);
  - лупа диаметром 5–7 см;
  - часовые отвертки;
  - скальпель или канцелярский резак;
  - некоторые специальные инструменты (см. далее).
- Расходные материалы:
  - припой свинцово-оловянный ПОС-30 (40) в прутке (без канифоли);
  - припой свинцово-оловянный ПОС-61 в проволоке ∅ 1–2 мм с канифолью;
  - активный флюс;
  - пассивный флюс (канифоль в кусочках и ее спиртовой раствор);
  - шкурка (самая мелкая на бумажной основе и покрупнее — на тканевой);
  - «Раствор спиртовой технический»;
  - чистый бензин (типа «Галоша» или «для зажигалок»);
  - ацетон или аналогичный растворитель (646 или 647);
  - клеи «Момент-кристалл» и «Гель» (или клей-герметик в тюбике);
  - фольгированный стеклотекстолит 1,5 мм;
  - готовые макетные платы под пайку компонентов;
  - беспаечная макетная плата;
  - термопаста;
  - кембриковые (изолирующие) трубки (термоусадочные и обычные), диаметром 1,5–20 мм;
  - клейкая полихлорвиниловая изоляционная лента черного и белого цвета;
  - скотч;
  - водостойкие фломастеры (1 и 2,5 мм).

К **специальным** следует отнести различные инструменты вроде:

- клещей для зачистки проводов (очень советую приобрести, только перед покупкой проверьте, чтобы зачищать можно было даже самый тонкий провод);

- отсоса для расплавленного припоя (он может пригодиться при демонтаже микросхем);
- обжимных клещей для плоских кабелей, кримперов для различных разъемов и т. п.

Конечно, не все из указанного вам потребуется немедленно. Большинство пунктов не требуют пояснений, но некоторые стоит рассмотреть подробнее. Сначала — пара слов о двух наиболее употребительных вещах, которые всегда должны быть под рукой: бокорезах и пинцете.

- **Бокорезы** — т. е. малогабаритные кусачки с губками, расположенными вертикально или под 45° — должны иметь острые, а не закругленные концы, быть сделанными из хорошей стали (бокорезы из маникюрного набора не подойдут решительно), а лезвия должны точно, без перехлеста, сходиться (рис. 3.1). Для проверки качества посмотрите через сомкнутые лезвия на свет — никакой щели наблюдаться не должно. Хорошо подогнанные и заточенные бокорезы должны резать бумагу. Долго будут служить имеющиеся кое-где в продаже бокорезы из твердого сплава (если у них лезвия хорошо подогнаны друг к другу, то ими можно кусать даже кевларовую нить, которая обычным ножницам и резакам не поддается). Однако, какого бы качества лезвия ни были, не следует кусать такими бокорезами стальную и даже толстую медную проволоку — для этой цели следует иметь обычные слесарные кусачки. Если в процессе работы на лезвиях все же образовались зазубрины — следует подправить их плоским алмазным надфилем или отдать их в переточку в профессиональную точильную мастерскую.

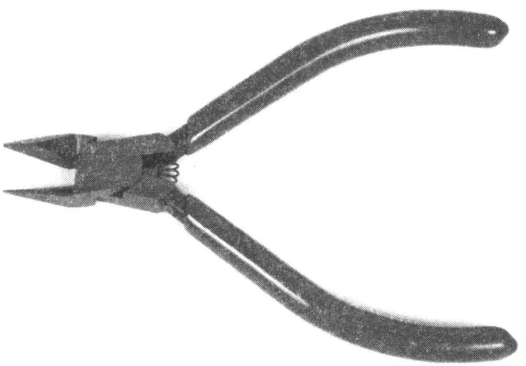


Рис. 3.1. Пример недорогих бокорезов удобной формы

- **Пинцет** лучше всего приобрести в магазине медицинского оборудования. Те пинцеты, которые предлагаются в радиомагазинах и тем более в магазинах для фотолюбителей, обычно не выдерживают никакой критики. Хотя они и могут пригодиться для некоторых технологических целей (например, переворачивать платы при травлении), но только не для радиомонтажа. Губки стандартного медицинского пинцета следует с помощью точила несколько заострить к концам с тыльной стороны — так удобнее браться за тонкие и короткие детали.

Паяльники

Три упомянутых в списке разновидности паяльников предназначаются для следующих работ (в скобках указана примерная мощность):

- пайки выводов микросхем и других компонентов с выводами до 1 мм, а также тонких проводов (15–20 Вт);
- пайки компонентов с толстыми выводами (разъемы, лепестки, штекеры и пр.) и толстых проводов сечением 1 мм^2 и более (40–65 Вт);
- пайки крупногабаритных деталей, например, пластин стеклотекстолита между собой (150–200 Вт).

В качестве бюджетного варианта вполне годятся отечественные паяльники на 220 В с деревянной ручкой — огромным их преимуществом является то, что деревянная ручка не греется так, как пластмассовая, а огромным недостатком — то, что жало приходится периодически править напильником и заново облуживать (см. далее).

А вот самый маломощный паяльник, который будет употребляться чаще всего и для самых ответственных работ, стоит приобретать фирменный и с запасными жалами разной формы (рис. 3.2). Жало у них гораздо долговечнее обычного медного, его не придется часто затачивать и облуживать. Приемлемые по цене паяльники такого рода обычно имеют только один крупный недостаток — они снабжены слишком толстым, тяжелым и жестким проводом, даже более жестким, чем у самых дешевых отечественных с обычным медным жалом. В таких случаях провод следует сразу заменить, причем вы вряд ли сможете приобрести достаточно мягкий готовый шнур с вилкой. Можно приобрести несколько метров провода типа ШВПТ повышенной гибкости — такого, какой часто используется в сетевых блоках питания со встроенной вилкой (один из проводов в нем обычно помечен цветной полосой), и подсоединить к нему обычную вилку. На выходе из ручки паяльника провод

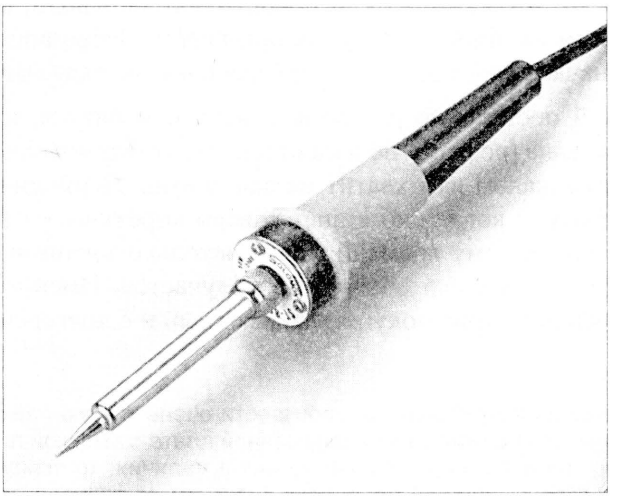


Рис. 3.2. Недорогой фирменный паяльник мощностью 20 Вт

дополнительно закрепляется герметиком, клином из кусочка пенопласта или пробки или просто обматывается изоляцией, чтобы не болтался.

Конечно, фирменный паяльник для пайки микросхем с мгновенным разогревом и специальным источником питания, гарантирующим изоляцию от сети и позволяющим регулировать температуру нагрева, весьма дорог. Неплохой промежуточный вариант — фирменные паяльники с долговечными жалами (ценой до 1 тыс. руб — фирмы Solomon, например), к которым можно приобрести запасные жала разной конфигурации.

Для такого самого употребительного маломощного паяльника следует проверить, правильно ли выбрана мощность и соответствует ли она выбранному жалу (чем длиннее и тоньше последнее, тем ниже температура его кончика). Вот простой эмпирический метод определения того, насколько мы правильно выбрали мощность, — следует коснуться разогретым жалом кусочка канифоли. Если канифоль плавится и некоторое время (порядка пяти-десяти секунд) потом дымится — все в порядке. На перегретом паяльнике канифоль вскипает и испаряется практически мгновенно. Наиболее кардинальным решением будет включение паяльника через ЛАТР (лабораторный автотрансформатор), что позволяет тонко регулировать температуру жала.

Паяльники большей мощности на 40–65 и 150–200 ватт используются гораздо реже, потому здесь вполне можно обойтись самыми дешевыми с обычным медным жалом. Такое жало следует облудить — для этого паяльник с отформованным и зачищенным с помощью напильника жалом следует включить в сеть и после некоторого прогрева окунуть в канифоль. Следя за тем, как канифоль растекается, важно не пропустить момент, когда паяльник еще не нагрелся окончательно, но температура уже достигла точки плавления припоя. С этого момента нужно водить жалом по припою, стараясь, чтобы припой растекся по как можно большей площади как можно более равномерно (особое внимание следует уделить самому кончику жала). Хороший вариант — окунуть паяльник в смесь опилок, напиленных крупным напильником из бруска припоя, и порошковой канифоли, а затем быстро несколько раз провести им по кусочку мелкой шкурки. Операцию облуживания следует провести до начала работ для всех трех паяльников, включая и фирменный.

Учтите, что припой постепенно растворяет медь или латунь, из которой обычно делаются дешевые жала, поэтому не надейтесь, что отформованного и залуженного паяльника с медным жалом вам хватит на всю жизнь. Периодически кончик жала надо будет затачивать, а когда оно станет совсем коротким, — паяльник придется менять. К сожалению, к тому времени жало настолько заклинивает в нагревательном элементе, что сменить его обычно не получается. Именно по этой причине самый ходовой паяльник стоит покупать фирменный и с долговечными жалами.

Совет

Если вам внезапно потребовалось произвести очень тонкую пайку (например, компонентов поверхностного монтажа на фирменной плате с высокой плотностью упаковки), а подходящего миниатюрного паяльника нет в наличии, то поможет следующий прием. Возьмите отрезок медной проволоки 1–1,5 мм от провода для бытовой электропроводки, зачистите и облудите его кончик, предварительно аккуратно срезанный под

косым углом, и плотно намотайте эту проволоку на жало паяльника. Облуженный кончик проволоки, который будет теперь играть роль жала, следует отогнуть вперед, но не на слишком большую длину, иначе он не прогреется как следует. После нагревания жала проволока расширится и «захочет» проворачиваться на жале, как бы плотно вы ее ни намотали. Чтобы этого избежать, следует после нагревания еще раз обжать витки большими плоскогубцами, а еще лучше закрепить их небольшим хомутиком из жести с винтовым зажимом.

Два слова о подставках: довольно удобные подставки есть в продаже, или их можно сделать самим, если конечно, подставка не продавалась вместе с паяльником. В любом случае подставку следует дополнительно снабдить металлической щеткой для очистки жала, почему-то отсутствующей даже в самых фирменных конструкциях. Идеально для этих целей подходит отрезок так называемой «корщетки» — короткой стальной щетины на толстой тканевой основе, использующейся в текстильном производстве. Если не найдете — купите обычную хозяйственную металлическую щетку самого маленького размера с деревянной ручкой и отпилите от нее ручку. Можно также использовать грубую канитель, из которой делают щетки для мытья посуды.

Флюсы для пайки

Это отдельная тема, которой не жаль посвятить некоторое время. Хотя рынок сейчас наполнен всяческими паяльными примочками (в магазинах «Чип и Дип» они занимают целую полку), опыт показывает, что в 99% случаев для получения надежных результатов можно обойтись всего двумя рецептами: обычным пассивным спирто-канифольным раствором и специальным активным флюсом для пайки плохо залуженных поверхностей.

Сначала о **спирто-канифольном растворе**, или, как его еще называют, канифольном лаке. Его можно приобрести или сделать самому. Единственное условие, которое следует соблюдать, — использование технического или медицинского 95–96% спирта. Денатурат, который иногда продают в хозяйственных магазинах, не подойдет из-за большого содержания воды. Приобрести технический спирт можно на радиорынках и в радиомагазинах (под названием «спиртовой раствор технический»), а медицинский — сами знаете где. (Вообще-то, он продается в аптеках под названием «Медицинский антисептический раствор» как раз в удобной расфасовке по сто грамм, но для его приобретения потребуется рецепт врача. Кстати, ни «технический раствор», ни даже «медицинский антисептический» я бы решительно не рекомендовал пробовать на вкус даже заядлым алкоголикам ввиду нешуточной угрозы для здоровья.)

Для изготовления канифольного флюса надо измельчить канифоль в порошок и наполнить им примерно на две трети пузырек, выбранный для этой цели (удобно употреблять пузырьки от лекарств емкостью 100–150 мл с пластмассовой завинчивающейся пробкой). Затем вы заливаете туда спирт в таком количестве, чтобы он полностью покрыл порошок с небольшим избытком, завинчиваете пробку и немедленно несколько раз энергично встряхиваете пузырек. Собственно раствор будет готов через несколько дней — в зависимости от степени измельчения канифоли. В течение этого времени пузырек нужно периодически встряхивать. Если в кани-

фоли имеются посторонние включения — они осядут на дно. В пластмассовую пробку пузырька неплохо заделать небольшую кисточку, обрезав ее ручку почти до уровня пробки, чтобы не торчала, иначе ваш пузырек легко перевернется, а растекание жидкой канифоли по столу — одно из самых тяжелых стихийных бедствий, которое можно себе представить. Для нанесения флюса также удобно применять деревянные зубочистки.

Простой канифольный лак, приготовленный по этому рецепту, имеет одно, но важное преимущество, — он совершенно не проводит электричество, ни в жидком, ни в застывшем состоянии, потому идеален для отладки схем. «Продвинутые» в радиолюбительском деле знакомые непременно предложат вам не возиться с этим делом, а использовать широко распространенный универсальный флюс, известный под названием «ЛТИ». Надо сказать, что под этим названием ходит довольно много разных продуктов, простейший из которых представляет собой тот же самый спирто-канифольный раствор с активирующими добавками, которые превращают жидкость в пасту, — так якобы удобнее с ней обращаться. Если вам нравится — используйте, однако имейте в виду, что очень часто под названием «ЛТИ» продают активный флюс, в состав которого входит солянокислый диэтиламин, выделяющий при нагревании пары соляной кислоты, очищающие место пайки. Это здорово, но такой флюс неудобен для отладочных радиолюбительских работ вследствие его электропроводности, — каждый раз после пайки его необходимо тщательно смывать спиртом, ацетоном или растворителем.

В качестве **активного флюса** для пайки незалуженных деталей, проволоки или поверхностей из стали, грязной меди, латуни или, скажем, нихрома удобно применять совершенно отдельную композицию. Из имеющихся в продаже можно рекомендовать «Паяльную кислоту» на основе хлористого цинка или «ХАФ» на основе хлористого аммония — оба они смываются водой.

> ***Совет***
>
> Автор же вот уже в течение трех с лишним десятков лет использует самостоятельно приготовляемый активный флюс, который дает восхитительные результаты даже для нержавеющих сталей (для пайки которых обычно рекомендуют использовать ортофосфорную кислоту). Приготавливается он следующим образом: нужно засыпать в пузырек примерно на одну треть высоты порошок хлористого аммония и залить доверху смесью, состоящей из 70% глицерина и 30% воды, взболтать это дело и оставить на одну-две недели. Если хлористый аммоний по истечении этого срока полностью растворится — досыпьте еще, если нет — осадок не помешает. Насыщенным раствором удобно заполнить одноразовый шприц или полиэтиленовую пипетку с завинчивающейся крышечкой (от лекарства, которое закапывается в нос при насморке). После применения остатки такого флюса обязательно нужно смыть теплой водой под краном или стереть мокрой тряпочкой, а затем тщательно высушить место пайки. Флюс совершенно нейтрален, не ядовит, безопасен для рук, не разъедает деревянное и пластиковое покрытие стола, но чрезвычайно текуч и страшно медленно испаряется, потому остатки его со стола и с других предметов следует тщательно удалять влажной тряпкой. Не следует его употреблять совместно с канифолью — они будут друг другу мешать, и смывать остатки станет куда труднее.

Описанные здесь два состава исчерпывают почти все нужды радиолюбительской практики. Один совет — не жалейте канифоли, даже для временных паек! Это по-

том выльется в потерю куда большего времени, чем будет затрачено на покрытие места пайки флюсом с помощью кисточки или зубочистки.

Макетные платы

Даже к готовым модулям-полуфабрикатам, выпускаемым специально для изготовления радиолюбительских конструкций (о них мы будем говорить в конце этой главы), нередко приходится подключать компоненты без специальных приспособлений для быстрого монтажа. А при сборке из отдельных компонентов такая задача возникает всегда. И встает вопрос о том, как быстро смакетировать и проверить схему.

Тут нам может помочь современная индустрия — на рис. 3.3 изображена макетная плата, которая позволяет осуществлять соединения компонентов без пайки. Такие платы известны достаточно давно, однако лишь в последние годы их стало приобрести довольно просто. Только не следует выбирать самые дешевые варианты в неизвестных интернет-магазинах — они отличаются от фирменных примерно тем же, чем дешевые электророзетки отечественного и китайского нонейм от розеток фирмы «Легранд» — т. е. быстрым исчезновением контакта из-за потери упругости лепестками в гнезде. Видимые на фото перемычки и соединительные провода также продаются в готовом к использованию виде вместе с платами. Впрочем, фирменные провода со штекерами приобретать необязательно: для соединения компонентов вполне подойдут отрезки обычного одножильного медного провода с жилой диаметром 0,5–0,7 мм.

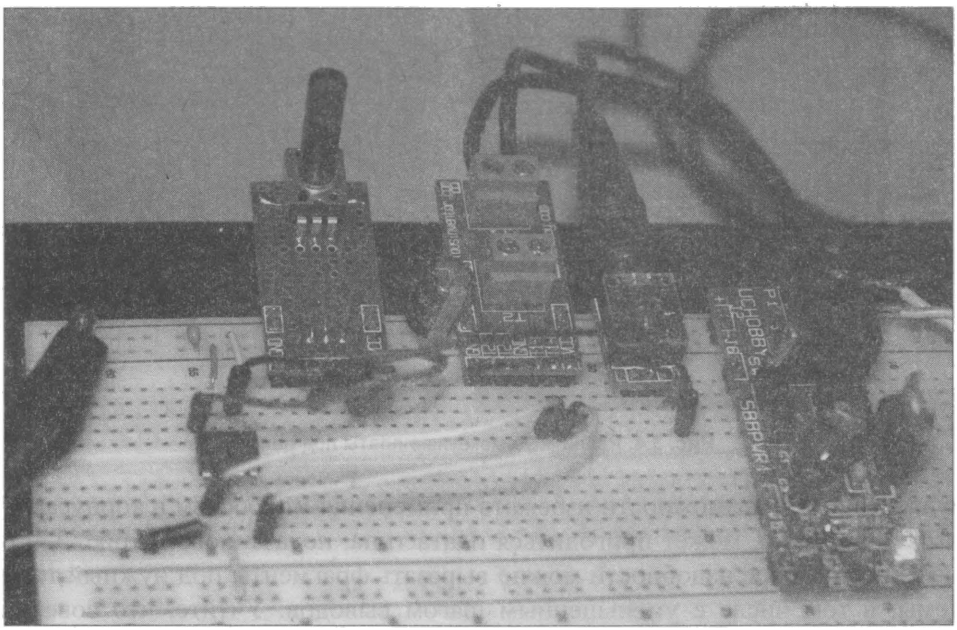


Рис. 3.3. Пример беспаечной (контактной) макетной платы с собранной схемой (фото с сайта **uchobby.com**)

Иногда собранную на такой макетной плате конструкцию размещают и в готовом изделии. Однако в принципе подобная плата пригодна лишь на этапе макетирования — в качественно выполненном изделии оставлять висящие гирлянды проводов, держащихся лишь на трении, не годится. Более универсальными являются макетные платы, где крепление компонентов и проводов осуществляется пайкой, — они годятся и для окончательной сборки достаточно сложных схем, если по каким-то причинам разработка специальной печатной платы оказывается невозможной.

Обычно, говоря о макетной плате, имеют в виду простую печатную плату с местами для установки компонентов (отверстиями и контактными площадками), как не соединенными проводниками вовсе, так и соединенными по некоей специальной универсальной схеме. В продаже имеется большое разнообразие подобных плат. Предпочтительно, чтобы под рукой всегда была, как минимум, простейшая универсальная плата с отдельными металлизированными отверстиями, расположенными по сетке с шагом 2,5 мм (рис. 3.4, *слева*). По мере надобности из нее ножницами по металлу вырезаются нужные фрагменты.

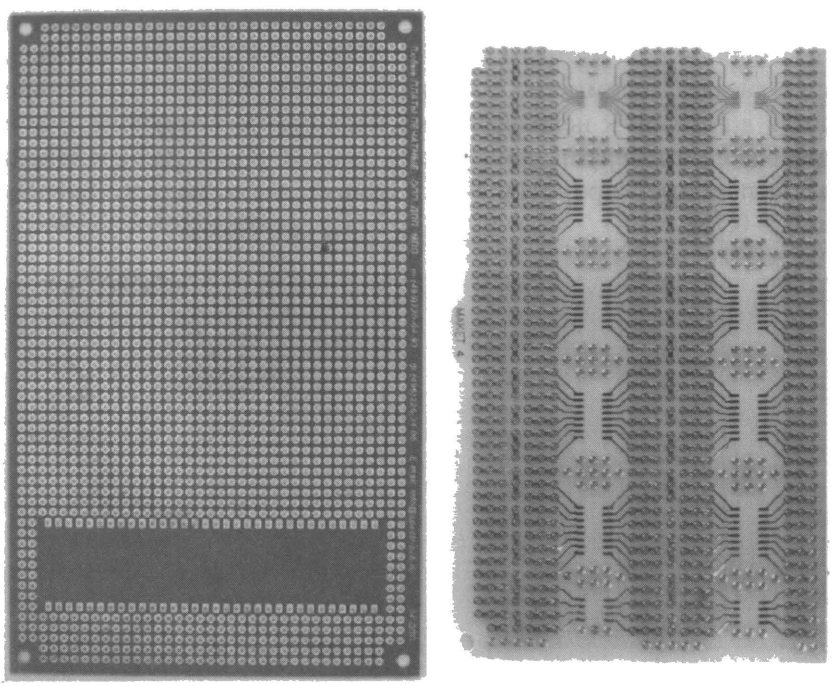


Рис. 3.4. Различные макетные платы под пайку

В случае, если нет возможности избежать применения микросхем в корпусах с планарными выводами, может пригодиться плата типа, показанного на рис. 3.4, *справа*, — из нее по мере надобности можно вырезать фрагменты под нужный тип микросхемы, в том числе с уменьшенным шагом выводов. Учтите, что совершенно универсальной платы, пригодной для расположения любых компонентов, не существует, так что чаще всего имеющиеся приходится дорабатывать, — например,

чтобы установить на универсальную плату (см. рис. 3.4, *слева*) клеммник для внешних соединений с зажимом «под винт» (подобный показанному в *главе 16* на рис. 16.17), приходится рассверливать отверстия до диаметра 1 мм, жертвуя металлизацией.

Если предварительное макетирование производилось на беспаечной макетной плате (подобной показанной на рис. 3.3), то очень удобно переносить такой макет на плату с пайкой, специально разработанную для такого переноса (рис. 3.5). Как видите, рисунок дорожек на ней повторяет конфигурацию беспаечной платы. Потому перенос возможен «один в один» — без дополнительных раздумий над размещением элементов.

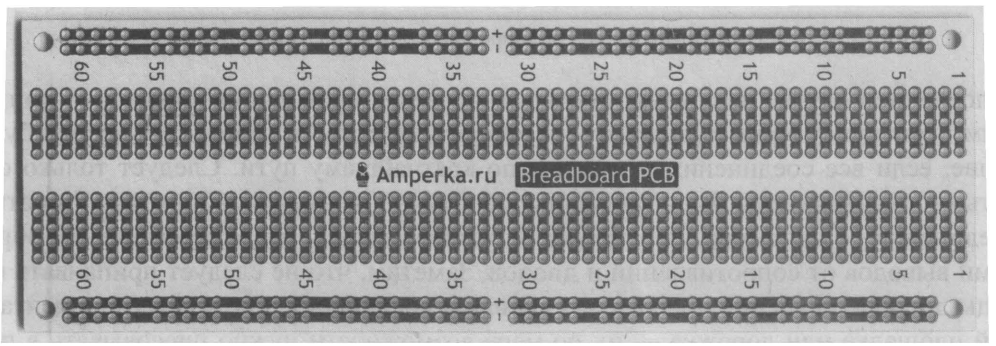


Рис. 3.5. Макетная плата под пайку для переноса схемы с беспаечной платы

Перед пайкой компонентов такую плату следует покрыть с обеих сторон канифольным лаком и высушить его под лампой, пока плата не перестанет липнуть к рукам. Канифольный лак хорош тем, что после окончательной сборки, пайки и проверки схемы его можно не отмывать. А если это все-таки необходимо для придания «товарного вида», его отмывают спиртом или спирто-бензиновой смесью. Проще всего налить немного такого растворителя в плоскую ванночку и погрузить туда плату минут на пятнадцать. Затем жесткой кисточкой удалить остатки канифоли там, где они не растворились, и окончательно промыть плату чистым растворителем. Учтите, что могут встречаться компоненты, которые не выносят воздействия растворителя, тем более загрязненного канифолью, — к ним относятся, например, датчики с активным поверхностным слоем (влажности, присутствия газов в воздухе и т. п.). В таких случаях плату не «купают» в растворителе, а аккуратно снимают остатки канифольного флюса смоченной в растворителе кисточкой.

Для соединения выводов компонентов на макетных платах под пайку в идеале следует их расположить так, чтобы нужные выводы были как можно ближе друг к другу. При этом перемычки между контактными площадками можно просто залить припоем (рис. 3.6, *слева*). Если разместить их настолько близко не удается, то соединения выполняются либо отрезками неизолированного одножильного провода диаметром 0,5–0,7 мм, покрытого припоем («луженкой»), как показано на рис. 3.6, *посередине*, либо с помощью отрезков обычного провода типа МГТФ в термостойкой (фторопластовой) изоляции, хотя монтаж с его помощью получается не столь красивым и надежным (рис. 3.6, *справа*).

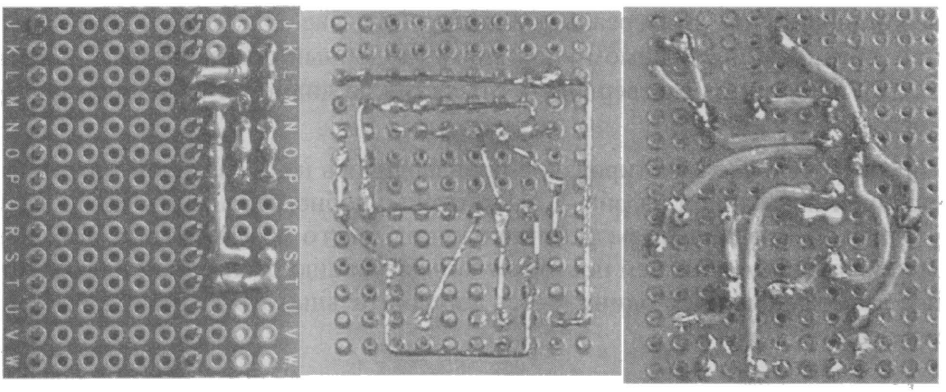


Рис. 3.6. Различные приемы выполнения соединений на макетной плате

В последнем случае не следует стараться провести проводники «красиво» (по прямым перпендикулярным линиям) — наоборот, качество и надежность схемы будет выше, если все соединения разведены по кратчайшему пути. Следует только стараться, чтобы провода были припаяны «внатяг», а не змеились по плате. Короткие соединения — например, перемычки — удобно делать неизолированными обрезками выводов от сопротивлений и диодов. Заметим, что не следует припаивать выводы деталей, особенно провода для внешних соединений платы, просто к контактной площадке или дорожке — их по мере возможности нужно просовывать в предусмотренное отверстие. Внешние соединения вообще не следует делать простой пайкой к контактным площадкам — это допустимо лишь как исключение. Для внешних выводов на плате устанавливаются упомянутые ранее клеммники, а для межплатных соединений предпочтительно употреблять штырьковые разъемы типа PLS (подробнее о них см. *разд. «Железо» главы 19*) и соответствующие плоские кабели.

Если вы делаете капитальную конструкцию, которая должна прослужить много лет, то нелишне прикрепить выходящий жгут к плате и к стенкам корпуса прибора хомутами так, чтобы места подключения проводников к плате не подвергались механическим воздействиям. И обязательно подобное делать для проводов сетевого питания, причем сетевые и другие высоковольтные провода следует «тащить» отдельным жгутом.

Правильные способы пайки проводников между собой показаны на рис. 3.7. Рисунок взят из старой книги для радиолюбителей, и его актуальность даже возросла в современных условиях, когда правильные рекомендации по пайке встречаются довольно редко.

Кое-кто мне может не поверить, но обычная скрутка, если она грамотно выполнена, может быть ничуть не менее надежна, чем пайка (по крайней мере в сухом воздухе и при исключении механических воздействий). Выполнение этой операции в разных вариантах показано на рис. 3.8, взятом из того же древнего пособия. И при пайке, и, особенно, при скрутке следует обязательно постараться предотвратить механическое напряжение, приложенное к месту их выполнения, — по крайней

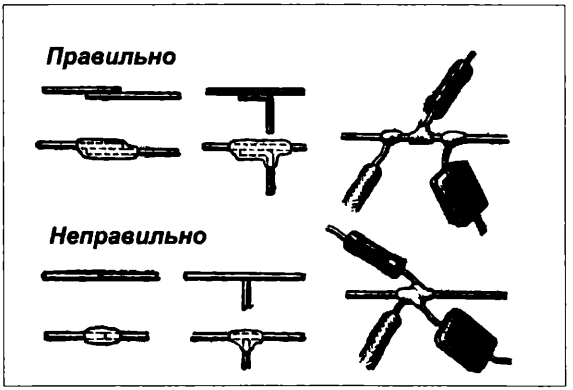


Рис. 3.7. Способы выполнения паяных соединений

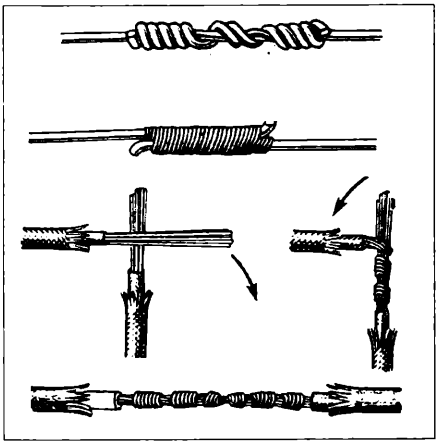


Рис. 3.8. Способы выполнения скруток

мере, нужно надежно обматывать место соединения изолентой, а еще лучше — защищать термоусадочной кембриковой трубкой (см. далее).

Если применение плоских кабелей для внешних соединений невозможно (как, например, в случае проводников между платой и регулирующими компонентами, установленными на передней панели прибора), то соединительные провода, в отличие от внутриплатных соединений, убирать в аккуратные жгуты как раз следует. Так получается значительно надежнее, потому что нагрузка распределяется между несколькими пайками (следует свивать в жгут провода, даже если их всего два!). Не рекомендуется для этого использовать изоленту или кембрик (потом очень трудно разобраться, куда чего идет), значительно лучше применить специальные затягивающиеся хомуты, закрепление нитками или даже обойтись вообще без креплений — в процессе монтажа завивать очередной провод вокруг имеющихся на один-два оборота (одинаковое количество оборотов для всех проводов в жгуте).

Совет

Для того чтобы пара скрученных проводов с одним свободным концом со временем самостоятельно не раскручивалась, поступают следующим образом: если сплетаемая пара короткая (в пределах 20–50 см), то ее следует свивать, пропуская между пальцами так, чтобы каждый провод закручивался в отдельности. Если же пара длинная (несколько метров), то нужно отмерить и отрезать пару проводов (обязательно одинакового диаметра и типа) с припуском по длине примерно процентов на 30%. Затем привязать концы обоих проводов совместно к какой-нибудь жесткой опоре и вытянуть их на всю длину, захватив в каждую руку по проводу и натянув их. Далее следует перехлестывать провода, перехватывая их из одной руки в другую и обязательно натягивая с разводом рук в стороны после каждого витка. Все вместе это несколько напоминает упражнение из комплекса утренней зарядки. Такой способ позволяет соорудить почти идеальную витую пару (с положенным одним витком на сантиметр длины), которая гарантирована от раскручивания. Более длинные провода так изящно скрутить не удается. Стандартный способ с использованием электродрели, к сожалению, от постепенного раскручивания не предохраняет — вся фишка в том, что при одновременном закручивании в одну сторону провода всегда сохраняют напряжение и постепенно стараются раскрутиться (вы, несомненно, не раз могли наблюдать это на примере длинного провода для телефонного аппарата или сетевого удлинителя).

Печатные платы

Все схемы в настоящее время располагают на печатных платах. Название «печатные» произошло от того, что промышленные платы изготавливаются методом фотопечати. В промышленности рисунок проводников на плате заранее подготавливают в одном из специализированных программных пакетов (наиболее известны PCAD и OrCAD), представляющих собой векторные графические редакторы с множеством дополнительных специализированных функций, включающих, в том числе, и автоматизированную раскладку (хотя несложные одно- или двусторонние платы раскладываются обычно вручную). Затем этот рисунок переносится на прозрачную пленку (примерно так же, как делаются пленки для полиграфической печати), образуя фотошаблон.

Если плата двусторонняя, т. е. дорожки-проводники у нее расположены с обеих сторон платы (самый распространенный случай), то фотошаблона приходится делать, естественно, два. Иногда готовую плату еще дополнительно покрывают термостойким ламинирующим составом, который защищает дорожки, оставляя свободными контактные площадки и отверстия для пайки (вы его не раз видели, скажем, на компьютерных картах — ранее он был только зеленого цвета, а последнее время в моде самые немыслимые расцветки), — в этом случае приходится делать дополнительные шаблоны.

Сразу заметим, что раскладка платы и изготовление шаблонов — самая дорогая стадия в этом процессе. Поэтому если вы решите отдавать свои разработки в подобное производство — убедитесь сначала, что в схеме нет ошибок.

Сами платы изготавливают из фольгированного стеклотекстолита, представляющего собой стеклоткань, пропитанную эпоксидным составом и покрытую с одной или двух сторон тонкой медной фольгой (когда-то основой для плат был гетинакс — т. е. пропитанная синтетической смолой бумага, но качество таких плат намного ниже). Стеклотекстолит бывает разной толщины — для наших целей удобнее всего употреблять двусторонний стеклотекстолит толщиной 1,5 мм.

Основная идея изготовления плат заключается в том, что места будущих дорожек покрываются нерастворимой пленкой, а остальная часть вытравливается в растворе какого-нибудь химического соединения, растворяющего медь. Травильных растворов известно довольно много (азотная кислота, перекись водорода, персульфат аммония и т. п.), из них в домашних условиях наименьшее количество проблем доставляет насыщенный раствор хлорного железа, которое продается в магазинах и на радиорынках. Хлорное железо безопасно для рук, но готовить его можно только в стеклянной или пластиковой посуде, а окружающие предметы следует беречь от попадания брызг, — высохшие капли раствора хлорного железа могут оставлять неудаляемые пятна на глазури многих не слишком качественных сантехнических изделий.

Подогревать раствор для ускорения процесса, как рекомендуется во многих рецептах, я не советую — растворение меди и без того идет достаточно быстро, зато риск подтравливания дорожек резко возрастает. А вот хорошее перемешивание обязательно — плату в растворе следует все время покачивать и периодически перевора-

чивать пинцетом, а в идеале следует иметь для этой цели лабораторную магнитную мешалку. Как только на свободных участках исчезнут последние следы меди, плату надо немедленно извлечь из раствора и промыть теплой водой под краном. При передержке в растворе подтравливание дорожек неизбежно. Правильно организованный процесс травления протекает за 10–20 минут, в зависимости от концентрации раствора и его температуры.

Самое сложное — сформировать на плате рисунок дорожек, достаточно стойкий к травильному раствору. Изготовить плату с не слишком тесно расположенными дорожками в домашних условиях можно, например, нарисовав их на обезжиренном стеклотекстолите водостойким (т. н. «перманентным» — permanent) маркером. Очень удобны специальные маркеры с тонким пером (подобные показанному на рис. 3.9). Маркер лучше брать новый — в процессе хранения даже из плотно закрытого маркера спиртовые чернила постепенно испаряются. Вести маркер вдоль линейки следует без торопливости — прочерченная линия должна быть без разрывов и утончений, через нее не должно просвечивать медное покрытие стеклотекстолита. Для удаления высохших чернил с платы применяются спирт и некоторые другие органические растворители (646-й или ацетон). Вносить исправления можно, тщательно смывая отдельные участки рисунка медицинской ватной палочкой, слегка смоченной растворителем.

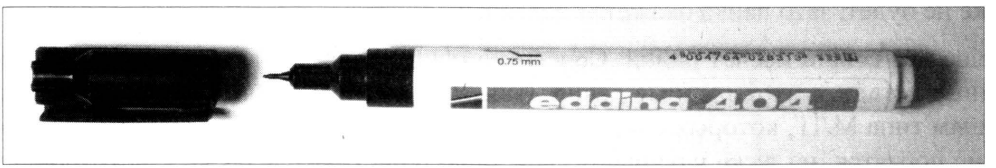


Рис. 3.9. Водостойкий маркер для нанесения рисунка на стеклотекстолит

Распространен среди радиолюбителей также способ переноса изображения с рисунка, напечатанного лазерным принтером на плотной бумаге, с помощью утюга. Автор этих строк перепробовал множество разных способов производства плат в домашних условиях (некоторые из них описаны в предыдущих изданиях этой книги) и свидетельствует: единственный реально работающий способ, позволяющий обойтись без пары квадратных километров безвозвратно испорченного стеклотекстолита, изложен вот в этой старой статье: **http://www.ixbt.com/mainboard/pcb-at-home.shtml**. Процесс там описан во всех подробностях и с привлечением конкретного опыта автора статьи. Ознакомившись с этим текстом, вы, возможно, предпочтете свою плату все-таки где-нибудь заказать.

Монтаж

Здесь и далее мы предполагаем использование навесных компонентов с гибкими выводами и микросхем в DIP-корпусах, а не деталей для поверхностного монтажа (SMD). Для радиолюбительских конструкций на макетных платах SMD-компонентов стоит избегать, ибо что-то менее приспособленное для отладки придумать трудно. Хотя в некоторых случаях без них не обойтись — все больше компонентов

или вообще выпускаются только в таких корпусах, или в DIP-корпусах найти их в продаже невозможно. В этом случае возможно обойтись фрагментом макетной платы, показанной на рис. 3.4, *справа*, который используется в виде модуля-переходника к шагу 2,5 мм. Конечно, в каждом таком случае решения придется принимать отдельно.

Если схема отлажена, а плата промышленного изготовления, то наиболее быстрый и надежный способ монтажа, к которому прибегают профессиональные монтажники, заключается в следующем. Первым делом в плату «натыкиваются» компоненты, причем выводы не следует откусывать — пусть они торчат. Далее плату, заполненную деталями, следует расположить компонентами вниз на поролоновой подкладке (чтобы они не выпадали и прижались к плате), после чего можно приступать к собственно пайке (именно в этом процессе огромное значение имеет, чтобы жало паяльника было правильно отформовано и имело достаточную длину). В процессе пайки следите, чтобы компоненты не перегревались, — если все сделано правильно, то для хорошей пайки достаточно трех секунд.

Для пайки удобно использовать тонкий припой с канифолью внутри — вы утыкаете одной рукой такую проволочку в место пайки, а другой прислоняете к этому месту кончик жала паяльника — секунда, и пайка готова. Несмотря на канифоль в припое, все же не забывайте заранее «покрасить» плату канифольным лаком, — хуже не будет, зато пайка окажется надежней.

И еще два совета насчет пайки. Совсем не исключено, что вам попадутся отечественные детали, изготовленные давно. В первую очередь это относится к сопротивлениям типа МЛТ, которых сохранилось довольно много (со временем они не только не портятся, но даже улучшают свои характеристики), а также к некоторым типам конденсаторов и других компонентов. Я не знаю, что за материалы тогда использовались, но ножки этих деталей при хранении чернеют (т. е. покрываются тонкой темной пленкой соединений типа сульфидов), и пайка их представляет определенные трудности. Выводы таких компонентов перед пайкой нужно обработать: сперва зачистить тонкой шкуркой-нулевкой (или просто несколько раз пропустить между губками медицинского пинцета), а затем облудить со всех сторон, стараясь не наносить лишнего припоя (иначе вывод может не влезть в предназначенное для него отверстие).

Второй совет относится к распайке компонентов на платах промышленного изготовления. Дело в том, что в процессе производства контактные площадки и дорожки покрываются припоями (типа сплава «Розе»), имеющими очень низкую температуру плавления. Потому, припаивая к ним вывод некоего компонента, не следует удерживать этот вывод на весу трясущейся рукой с пинцетом — припой застынет тогда, когда тонкий слой сплава на поверхности дорожки еще будет жидким, и очень надежное по внешнему виду паяное соединение на поверку окажется просто блямбой припоя, слегка прижатой к контакту на плате за счет упругости вывода.

После пайки выводы откусывают на нужную длину. Для промышленных плат с металлизированными отверстиями достаточно, чтобы места пайки выступали на 1 мм над поверхностью платы. Для плат собственного изготовления нужно не забывать,

что сквозные отверстия не имеют металлизации, и их нужно пропаивать на обеих сторонах платы.

Теперь несколько слов о **демонтаже** тех компонентов, которые следует удалить или заменить. Демонтаж, который не повреждает компонента с большим количеством выводов, — целое искусство, включающее использование газовых паяльников и разнообразных отсосов жидкого припоя. Мы не будем здесь на этом подробно останавливаться, т. к. освоить это дело со слов не проще, чем научиться водить автомобиль по рассказам бывалых шоферов. Расскажем только, как правильно подготовить плату при замене компонента (не обращая внимания на судьбу удаляемого) — такое требуется не только для собственно замены, но и, например, при очистке макетных плат для дальнейшего использования.

Сначала выкусите использованный компонент бокорезами. Детали с двумя выводами — резисторы или диоды — можно попытаться выпаять. Для этого нужно плату закрепить в тисках, а деталь подцепить отверткой или пинцетом и коснуться паяльником места пайки — вывод под нагрузкой вылезет из отверстия сам. Иногда такая операция проходит удачно, и деталь даже годится для дальнейшего использования!

После удаления обязательно нужно прочистить отверстия, которые оказываются заполненными припоем и обрезками выводов. Для этого смочите нужные места канифольным лаком и закрепите плату в тисочках в вертикальном положении. Затем возьмите в одну руку тонкую заостренную деревянную палочку (удобно для этой цели использовать зубочистки или заостренные спички), прижмите ее острым концом к залитому припоем контактному отверстию, стараясь направить в центр, а затем коснитесь паяльником припоя вокруг отверстия. Если отверстие металлизированное, то можно паяльник направлять с обратной стороны платы, если нет — следует касаться им рядом с палочкой. Заостренный конец влезет в отверстие, после чего паяльник следует убрать и выждать до полного застывания припоя. Остатки припоя, иногда вместе с обрезками выводов, нужно отломить пинцетом. Деревянная палочка довольно быстро изнашивается, потому ее следует часто менять или затачивать.

Немного о проводах

Мы ограничимся только кратким, без подробностей, рассмотрением монтажных и обмоточных проводов лишь самых употребительных типов, потому что типов проводов очень много. Более подробные сведения о них вы можете найти на многочисленных справочных ресурсах в Сети.

Для монтажа макетов и для межплатных соединений, если их немного, удобнее всего применять МГТФ — многожильный провод в тефлоновой (фторопластовой) оболочке. Он удобен тем, что изоляция у него не плавится, даже если ее непосредственно греть паяльником. Обычный провод МГТФ розоватого или белого цвета, но есть и высокопрочный МГТФ в изоляции черного цвета, который еще удобен

тем, что у него проводники заранее залужены. Разноцветный МГТФ тоже существует, но т. к. тефлон окрашивается плохо, цвет ему придают с помощью второй наружной оболочки из лакоткани, что делает его жестким и значительно увеличивает его толщину. Выпускается и экранированный МГТФ — с его помощью удобно протягивать, скажем, внутрикорпусные соединения от входных разъемов к платам в аудио- и измерительной аппаратуре, однако экран у него ничем не изолирован, и, как правило, приходится его убирать в кембриковую трубку.

Второй, самый употребительный, тип известен под собирательным названием МГШВ (так называют, на самом деле, много разных, но похожих типов проводов, но мы не будем на этом останавливаться, — продавцы и ваши коллеги всегда поймут, о чем речь). Он имеет полихлорвиниловую изоляцию, внутри которой может быть дополнительная «шелковая» намотка (собственно МГШВ), хотя иногда она отсутствует (МГВ). Если намотка имеется, то такой провод годится для прокладки проводников с высоким (220 В) напряжением. Провода без шелка для такой цели употреблять не рекомендуется — если только изоляция специально не утолщена (как в проводе ШВПТ повышенной гибкости, который мы рекомендовали для замены сетевого провода паяльников). МГШВ лучше приобретать с уже залуженными жилами, т. к. изоляция легко плавится, и провод лишний раз нагревать не стоит.

Провода в полиэтиленовой изоляции (вроде телефонной «лапши») для монтажа путем пайки употреблять не следует — они жесткие, а полиэтилен очень легко деформируется при нагревании, оголяя жилу на недопустимую длину.

Заметки на полях

Импортные монтажные провода часто имеют маркировку, нанесенную прямо на изоляцию. Обычно такая маркировка соответствует американскому стандарту AWG (American Wire Gauge), и по ней можно узнать приблизительный диаметр провода по довольно сложной формуле: $d = 0{,}127 \quad 92^{((36-AWG)/39)}$. Из формулы следует, что чем меньше цифра в стандарте AWG, тем толще провод. Для удобства пересчета проще воспользоваться специальными таблицами, которые можно найти в Интернете, заодно в них сразу приведено точное сечение провода (для одножильных и многожильных проводов значения при одной и той же маркировке заметно различаются). По сечению можно приблизительно определить допустимую токовую нагрузку на провод: так, для скрытой проводки (что примерно соответствует условиям в корпусах приборов) допустимой считается нагрузка 8 А на 1 мм^2 сечения (многожильный провод AWG 18 диаметром около 1,2 мм). Провод AWG 18 также считается стандартным для компьютерных блоков питания с нагрузкой около 7,5 А на каждый провод, и по этому признаку тоже можно оценить необходимый в вашем случае диаметр. Обратите внимание, что для проводов в плотной намотке эта норма не действует, там провода должны быть толще (см. *главу 9*). В электронных приборах запас по толщине проводников вообще должен быть значительно выше, чем для силовых проводов, — во избежание локального падения напряжения на тонких участках, которое может вызывать непредсказуемые эффекты в импульсных и высокочастотных схемах.

Провода для намотки трансформаторов называют *обмоточными*. Подобно МГШВ, они ходят в народе под собирательным названием ПЭЛ или ПЭВ (точнее, ПЭВ-2, что означает двухслойное покрытие лаком). Эти провода имеют покрытие из специального гибкого, термостойкого и устойчивого к растворителям лака. Оно доста-

точно тонкое, так что при измерении диаметра этих проводов микрометром или штангенциркулем в первом приближении толщину покрытия можно не учитывать. Единственное, чего это покрытие «не любит», — слишком маленьких радиусов изгиба, поэтому при использовании обмоточных проводов следует тщательно избегать «колышек» и запутывания в узлы. Так как покрытие термостойкое, то при пайке конец такого провода зачистить для облуживания довольно сложно. Это следует делать либо механически (скажем, при помощи тонкой шкурки — это надежнее, чем ножом или скальпелем), либо при помощи народного средства, которое заключается в том, что конец провода кладется на таблетку обыкновенного аспирина и прижимается к ней хорошо разогретым жалом паяльника (после чего конец следует очистить от обгоревших остатков лака). Есть и специальные высоковольтные обмоточные провода (ПЭЛШО), которые покрыты дополнительно слоем шелка (почему-то они чаще всего синего цвета).

В современных устройствах при необходимости провести много проводников между платами часто применяются плоские кабели с уже установленными разъемами типа IDC, PLD (подробнее о последних см. в *разд. «Железо» главы 19*) или аналогичных. Чаще всего такие *шлейфы* продаются уже готовыми вместе с модулями, но иногда их приходится делать самостоятельно. Разъемы на плоские кабели, конечно, можно монтировать и пайкой (при этом обязательно укрепляя место соединения термоусадочной трубкой, как описано далее), но быстрее, проще и надежнее устанавливать их на кабель с помощью упоминавшегося инструмента, называемого кримпером. При этом потребуются специальные модификации разъемов «под обжим».

Два слова о кембриковых трубках, которые употребляют для изоляции, — например, при пайке разъемов или при необходимости сращивания проводов между собой (имейте в виду, что профессионалы редко употребляют липкую ленту, а всегда пользуются кембриком). Они бывают двух типов: обычные и термоусадочные. У термоусадочных трубок при нагревании до 110–120 градусов диаметр уменьшается примерно вдвое, и они становятся жестче, прочно обволакивая место пайки и не давая проводу гнуться и ломаться. Для осаживания таких трубок используют специальные монтажные фены, но в крайнем случае можно обойтись и зажигалкой (лучше типа «турбо» — они дают некоптящее пламя), только следует постараться не нагревать трубку слишком сильно. Трубки большой длины можно осаживать выдерживанием в кипятке в течение нескольких минут.

Корпуса

Проблема корпусов для радиоаппаратуры не стоит особенно остро — все крупные фирмы, торгующие компонентами, торгуют и различными корпусами, а есть еще и много фирм помельче. Беда тут примерно та же, что с покупкой, скажем, обуви — вроде ее много на любой вкус и кошелек, да одни ботинки не смотрятся, в других кантик не такой, третьи цветом не вышли, четвертые в подъеме жмут... Короче, подобрать под конкретный прибор готовый корпус — задача очень непростая. Потратив 20–30 баксов на блестящее заморское изделие, очень не хочется браться за

напильник, чтобы доводить его до ума, но приходится — здесь должно быть окно для индикатора, эту стенку вообще надо удалить, ибо тут будет стоять радиатор для мощного транзистора, тут требуются фигурные отверстия под разъемы... Тогда, спрашивается, зачем тратили баксы-то? А если еще ошибешься, что нередко случается даже с опытными слесарями?

В общем, есть простой способ изготовления корпусов в домашних условиях под конкретные нужды, причем если руки на месте, то такие корпуса в готовом изделии будут выглядеть не хуже фабричных. Заключается он в том, что вы сначала рисуете эскизы всех стенок и перегородок, располагаете детали и платы, чтобы они не наезжали друг на друга, выверяете размеры (компьютер дает простор для такого рода творчества), а затем по готовым эскизам переносите размеры на фольгированный стеклотекстолит и вырезаете заготовки. Можно даже все отверстия заранее сделать — удобнее работать с пластинкой, чем с готовой коробкой. Затем, закрепляя заготовки под прямым углом друг к другу, пропаиваете место стыка обычным припоем. Вот тут очень пригодится самый мощный паяльник и упомянутый водорастворимый флюс.

Секрета в таком процессе всего два: во-первых, надо не забывать давать припуски на толщину материала по нужным сторонам заготовок, во-вторых, иметь в виду, что припой сокращается в объеме при застывании. Потому пластинки под прямым углом относительно друг друга надо прочно закреплять, иначе угол окажется совсем не прямым, а распаять будет уже очень трудно. Готовый корпус обтягивается самоклеящейся пленкой — например, под дерево. Если делать все аккуратно и продуманно, получается ничуть не хуже фирменных изделий!

В дальнейшие детали этого процесса я вдаваться не буду, т. к. мы собрались все же заниматься схемотехникой, а не дизайном корпусов для радиоаппаратуры. Дам еще только два совета. Первый — если у вас в корпусе предусмотрено окно для индикаторов, то его надо делать из дымчатого, а не прозрачного пластика, а все, что за этим окном расположено, кроме, естественно, самих индикаторов (но включая плату с деталями и блестящими проводниками), следует выкрасить в черный цвет из аэрозольного баллончика. Это придаст несравненно больше «фирменности» вашей конструкции. Конструкции, в которых через стекло виднеются пайки на печатной плате, выглядят ужасно. Можно к тому же заклеить всю незадействованную поверхность окна изнутри черной липкой лентой. Если следовать этому совету, то можно не выпиливать окна точно по размеру индикатора, что довольно сложно сделать красиво, а выполнить из дымчатого стекла, скажем, всю переднюю панель.

Второй совет касается нанесения надписей на переднюю панель. Скажу сразу: наилучший способ — заказать панель с лазерной гравировкой. Но это дорого и хлопотно, поэтому хочется сделать ее самому. Ручной способ отвергаем с порога — ничего не может выглядеть кошмарнее, чем надписи, сделанные вручную, и никакие трафареты и гравировальные машинки здесь не помогут. Это вообще была одна из самых тяжелых проблем до самого последнего времени, и не только для радиолюбителей, — даже мелкосерийные приборы на советских заводах выпускались с гравированными вручную надписями, и это было ужасно.

К счастью, в последние годы в связи со всеобщей доступностью принтеров проблема качественной печати любым размером шрифта, любым цветом и на любом фоне решена полностью. На струйных принтерах это делается на специальной пленке, которая с одной стороны липкая и покрыта защитным слоем, как самоклеящаяся пленка, а с другой имеет специальную пористую фактуру, хорошо удерживающую принтерные чернила. Она довольно дорогая, но десяти листочков вам хватит на всю оставшуюся жизнь, если вы, конечно, не собираетесь налаживать серийное производство. Если же такой пленки под рукой нет, то можно напечатать надписи на плотной мелованной бумаге — например, на обратной стороне настенного календаря, причем тут предпочтительно использовать лазерный принтер из-за более высокой водостойкости. Затем полученные лейблы вырезаются по размеру и приклеиваются тонким двусторонним скотчем или кусочком обычного прозрачного, аккуратно вырезанным по размеру, несколько большему, чем прижимаемый ярлык. Неплохо выглядят надписи, напечатанные вывороткой, — т. е. белым цветом на черном фоне, только не забудьте закрасить белые торцы бумаги по месту отреза черным фломастером, иначе они будут очень бросаться в глаза.

Новые подходы в любительском конструировании

Еще лет десять-пятнадцать назад, когда я готовил первое издание этой книги, способы изготовления любительских конструкций, которые обычно делаются в одном экземпляре, в основном сводились к применению готовых печатных макетных плат с металлизированными отверстиями, о которых мы говорили ранее. В настоящее время все эти способы никуда не исчезли и даже значительно усовершенствовались. Однако появился и ряд новых способов — радиолюбительское хобби на Западе становится все популярнее, и за дело взялась индустрия, которая придумала ряд новшеств, существенно облегчающих этапы макетирования и изготовления законченного прибора.

В первую очередь следует отметить появление множества различных более-менее законченных модулей-полуфабрикатов, или, по-английски, kits — «комплектов». Ранее подобные «киты» предлагались почти исключительно профессионалам для облегчения разработки новых изделий, ныне же они сильно подешевели и стали доступными для широкого круга любителей. Существенное значение это имеет главным образом для сложных многофункциональных компонентов — таких как микроконтроллеры. Зато из-за расширенных возможностей подобных платформ с готовыми типовыми узлами стало реальностью воплощение «малой кровью» многих задумок, малодоступное ранее для «классического» радиолюбителя, в чем вы можете убедиться сами, дочитав эту книгу до конца.

Платформа Arduino

Ярким примером такого подхода может служить платформа Arduino и ее многочисленные аналоги под другими названиями: Freeduino, Seeeduino, Craftduino,

Carduino и т. д., и т. п. Платформа Arduino стала одним из примеров использования принципов Open Source к аппаратным средствам — документация на саму платформу и на средства ее программирования полностью открыта и доступна для любого желающего. Немаловажную роль сыграло то, что готовые модули Arduino продаются по всему миру за сравнительно небольшую цену — в пределах 30–40 долларов США, а множество компаний и торговых домов предлагают различную периферию, дополняющую базовые функции Arduino. Многие торговые организации и в России продают платы Arduino и конструкторы на их основе. Подробнее об этой платформе читайте в *главах 20–22*.

Отметим, что по примеру Arduino были созданы и многие более сложные платформы, из которых наиболее известной стала Raspberry Pi, — одноплатный компьютер с простой операционной системой на основе Linux. Рассмотрение таких платформ выходит далеко за рамки этой книги, но упомянуть о них здесь следует, — их применение резко снижает порог вхождения в тему построения самых сложных современных устройств своими руками.

Чем же так хороши все эти «дуины»? Типовой модуль Arduino (см. рис. 20.1) содержит все необходимые узлы для того, чтобы можно было приступить к проектированию без каких-то предварительных операций, — не требуется ничего сверлить и паять, только подключить платформу к ПК и установить программу проектирования. При этом дополнительная периферия часто просто пристегивается к основной плате также без применения пайки. Итого весь порог освоения сводится к приобретению некоторых навыков программирования, что облегчается существованием многочисленных интернет-ресурсов по этой теме, в том числе на русском языке. А изготовление конечного продукта сведется просто к подбору и доработке подходящего корпуса.

К микроконтроллерам и Arduino мы еще вернемся — современному радиолюбителю просто нельзя не уметь ориентироваться в ее возможностях, и этой теме мы полностью посвятим последние главы книги. Заметим, что одно из главных удобств такого «индустриального» подхода к радиолюбительству — то, что почти все комплектующие, заранее подобранные по совместимости, можно приобрести в одном месте (примерами могут служить интернет-магазин «Амперка», ориентированный на Arduino и родственные ему платформы, сайт **iarduino.ru**, интернет-магазин «Терраэлектроника» и многие другие подобные организации).

Редактор Fritzing для рисования схем и плат

Автор неукоснительно придерживается линии, которую можно кратко охарактеризовать принципом Оккама: «не изобретать сущностей сверх необходимого». Если можно обойтись простыми инструментами, то их и следует применять. Но в этом своем стремлении к простоте желательно соблюдать меру — нередко новые инструменты дают новые возможности, делают жизнь проще и удобнее. К таким инструментам относятся графические редакторы, «заточенные» под проектирование электронных устройств. Самые «продвинутые» из них (такие, как популярный

Eagle или OrCAD) представляют собой практически полноценную систему автоматизированного проектирования (CAD).

Мы здесь рассмотрим бесплатный графический редактор Fritzing. Этот редактор является развитием идеологии Arduino по предоставлению непрофессионалам простых способов разработки электронных схем и изготовления печатных плат. Пока проект Fritzing еще находится в стадии вечной «беты», но уже вполне работоспособен, по крайней мере, для не слишком сложных конфигураций схем и плат. Из Fritzing можно и загружать код в платы Arduino, но этой возможностью я лично пока поостерегся пользоваться, используя Fritzing по прямому назначению — для разработки электронных схем. Есть гораздо более продвинутая и отработанная профессиональная среда проектирования схем на микроконтроллерах под названием Proteus (Fritzing основан на тех же принципах), но она стоит денег и не таких уж маленьких.

Заметим, что идеальное направление использования Fritzing — для подготовки и распространения схем на макетных платах, и подготовленные в этом пакете иллюстрации вы можете встретить на многих ресурсах в Сети. Правда, в этой книге, к сожалению, вы таких картинок почти не увидите — они обретают смысл лишь при воспроизведении в цвете.

Редактор Fritzing, как уже было отмечено, является бесплатным и свободно распространяемым (Open Source) программным обеспечением. Скачивать его лучше с официального сайта **fritzing.org**. Русификация (частичная — переведены только основные пункты меню) при наличии русскоязычной версии Windows производится автоматически. Устанавливать ничего не требуется — с оригинального сайта сразу скачивается готовый ZIP-архив, соответствующий разрядности установленной системы — 32 или 64 (уточнить ее можно через **Панель управления | Система**). Остается только развернуть архив в желаемое место на диске, после чего можно при желании вынести на рабочий стол значок, соответствующий файлу Fritzing.exe.

Основное окно редактора Fritzing показано на рис. 3.10. Прилагаемая к редактору библиотека содержит большое количество типовых элементов, которые показаны на панели справа. Можно создавать свои элементы, а также подключать другие библиотеки (довольно много компонентов можно найти на известном ресурсе **GitHub.com**). Все рисунки на вкладках редактора цветные, что значительно облегчает восприятие схем.

На вкладке **Принципиальная схема** (см. рис. 3.10) изображена сейчас схема подключения к Arduino матричного LED-индикатора. Обратите внимание на изображение резисторов — в оригинале они выполнены, разумеется, в западном стиле (в виде ломаной линии). Чтобы изменить начертание на привычные прямоугольники, показанные на рис. 3.10, необходимо отредактировать SVG-изображение резистора именно для принципиальной схемы, которое находится в папке программы по адресу Fritsing\parts\svg\core\schematic\resistor.svg. Редактирование можно выполнить в любом векторном редакторе, поддерживающем формат SVG (есть также подобные онлайновые сервисы или плагин для Photoshop), только не забудьте на всякий случай сохранить копию оригинального файла resistor.svg.

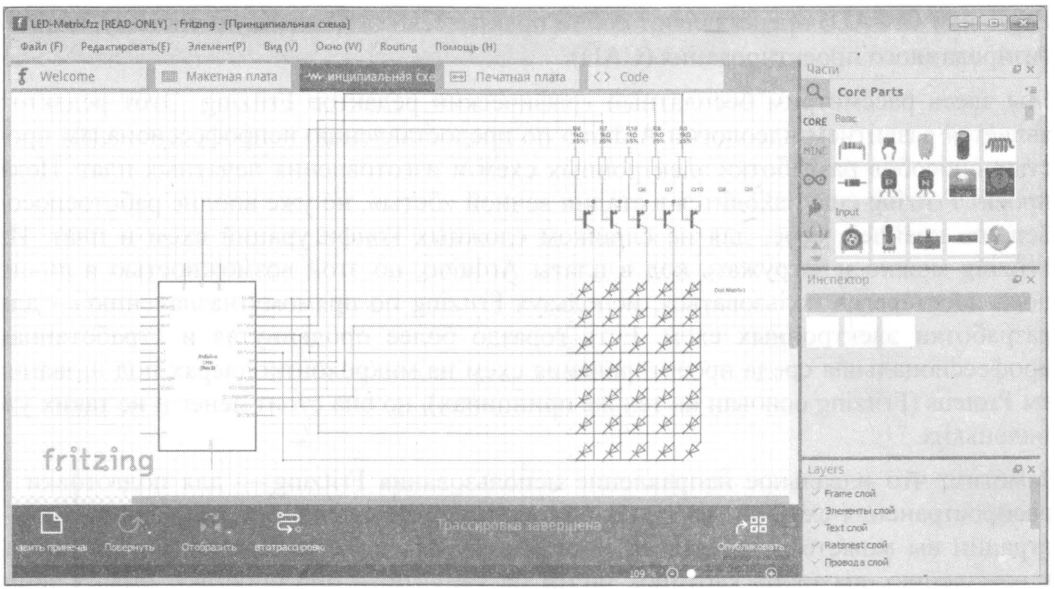


Рис. 3.10. Редактор Fritzing: вкладка **Принципиальная схема**

На вкладке **Макетная плата** (рис. 3.11) при создании принципиальной схемы автоматически появляются соответствующие элементы, и наоборот — вы можете начинать создание проекта не с принципиальной схемы, а с макета. На макете или на принципиальной схеме (порядок неважен) расставляются компоненты и рисуются соединения. Для этого сначала делается щелчок и удержание кнопки мыши на начале соединения, потом курсор перемещается на конец соединения, после чего

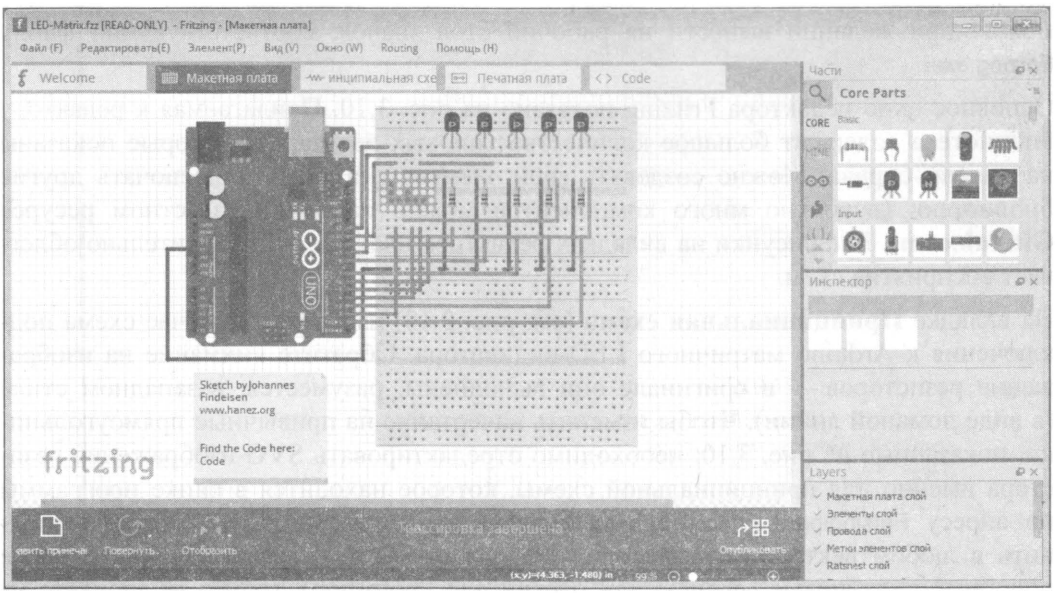


Рис. 3.11. Редактор Fritzing: вкладка **Макетная плата**

кнопку мыши нужно отпустить, — образуется прямая сплошная линия, на которой можно добавить нужные углы перетаскиванием мышью. После этого вы нажимаете на кнопку **Автотрассировка**, и на остальных вкладках появляются созданные вами связи. Когда схема нарисована и отлажена, автоматически созданный макет (или, соответственно, принципиальную схему) следует привести в опрятный вид, передвигая и поворачивая элементы, как удобно.

Как и профессиональные «взрослые» специализированные CAD для электроники, Fritzing имеет возможность автоматической разводки печатной платы. Нарисовав схему и отладив ее на макете и на принципиальной схеме, вы переходите на вкладку **Печатная плата** (рис. 3.12), где обнаруживаете заготовку платы с кучкой ваших деталей, размещенных в произвольном порядке. Вы задаете размер платы и количество слоев, размещаете компоненты в нужных местах и, если надо, снова нажимаете на кнопку **Автотрассировка**. Результаты автоматической трассировки, как правило, потом требуется довести «до ума», как и всякое творение компьютерного интеллекта, но это несравненно проще, чем «раскладывать» плату вручную с самого начала.

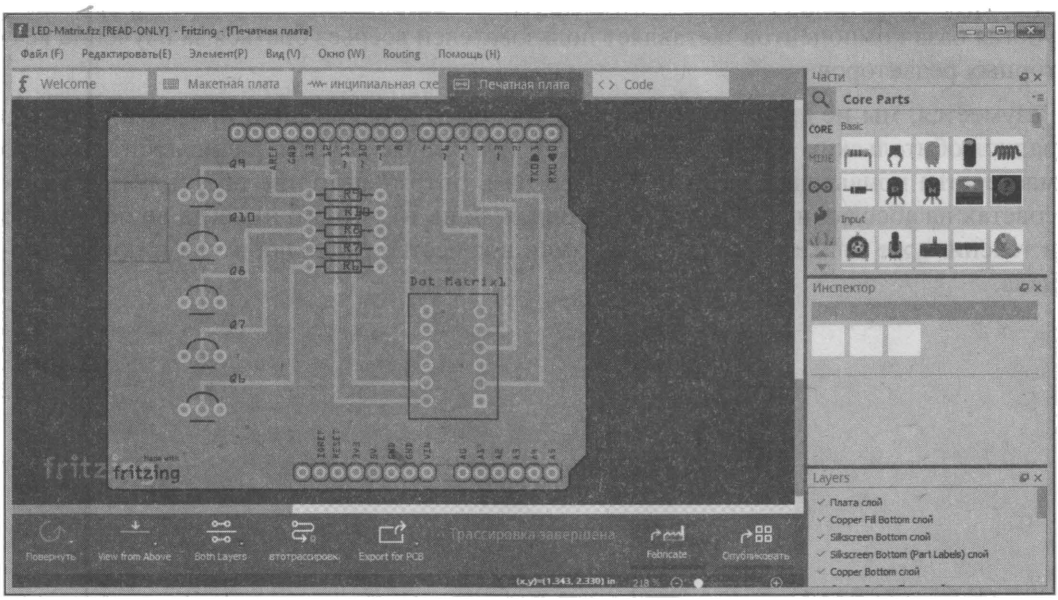


Рис. 3.12. Редактор Fritzing: вкладка **Печатная плата**

Заметки на полях

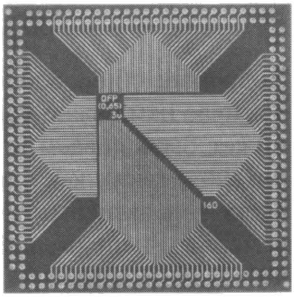
Для того чтобы измененные или вновь созданные изображения компонентов работали в процедурах автоматического создания схем (например, разводки принципиальной схемы и платы по созданному макету или в любом другом порядке), эти изображения должны иметь привязанные (assign) к SVG-рисунку выводы, которые и соединяются между собой. Самый простой способ получить такие выводы — выполнять новый рисунок на основе уже имеющегося. Если изменения велики, или рисунок сделан «с нуля», то привязанные выводы расставляются в готовом изображении в специальном редакторе Edit Part, входящем в состав Fritzing.

Полученный рисунок платы можно экспортировать в графические форматы для публикации или печати, а также в профессиональный формат Gerber для посылки проекта производителю печатных плат. Можно также отдельно сформировать список используемых компонентов. Следует учесть, что прямая конвертация в обычные рисунки PNG или JPEG допустима лишь для быстрой публикации схем в Интернете. При желании точно сохранить размеры в графических форматах следует предпочесть экспорт в PDF, а для того, чтобы получить качественные графические изображения, экспортировать следует в формат SVG, который затем отдельным редактором конвертируется в формат PNG с заданным разрешением.

Вы можете не поверить, но изложенное в этих нескольких абзацах — практически все, что требуется знать для успешного освоения редактора Fritzing. Разумеется, там есть много практических нюансов, но управление редактором сделано настолько наглядно и понятно, что с ним с лёта справится любой, кто хотя раз в жизни пробовал рисовать на компьютере.

Есть, конечно, у Fritzing и недостатки, самый крупный из которых — крайне ограниченные возможности настроек. Как и у всякого софта породы Open Source, у него практически отсутствует внятная инструкция пользователя. И, наконец, довольно убогая база компонентов заставляет пользователей всерьез браться за изучение векторных редакторов.

Разумеется, мы не смогли в этой главе раскрыть и малой доли правил и секретов радиолюбительских технологий. К тому же разные школы радиолюбительского мастерства привыкли к разным технологиям, потому автор не претендует в своих советах на абсолютную истину, рекомендуя лишь то, что его никогда не подводило в течение уже почти сорока лет практики в качестве радиолюбителя и профессионала.



ГЛАВА 4

Тригонометрическая электроника

О частотах, периодах, мощности, переменных напряжениях и токах и немного о сигналах

И оба во весь опор помчались в сторону столицы.
А. Дюма. «Три мушкетера»

Электрохимические (гальванические) элементы и аккумуляторы, с которыми мы экспериментировали в *главе 1*, являются источниками постоянного напряжения. Определение «постоянное» не означает, что такое напряжение вообще не меняется. Отнюдь — типичный график зависимости напряжения от времени (так называемые *разрядные кривые*) для гальванических элементов разных типов приведен на рис. 4.1. Отметим, что большинство литиевых элементов имеет номинальное напряжение 3 В (для них значение напряжения на графике следует умножить на два), но, как мы писали в *главе 2*, в последние годы появились и элементы этого типа, аналогичные обычным щелочным. Как видите, зависит напряжение не только от времени, — отдельные пики на графиках относятся к моментам, когда нагрузка отключалась, при этом напряжение элемента скачкообразно росло, а затем, при ее подключении, снова падало.

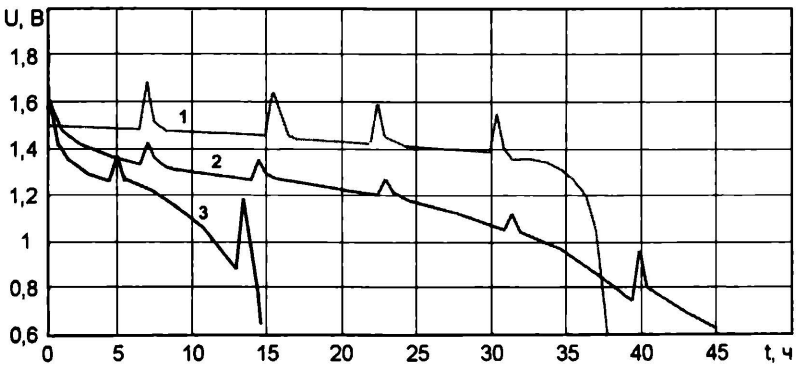


Рис. 4.1. Зависимость напряжения от времени для гальванических элементов при токе нагрузки 100 мА:
1 — литиевый (в пересчете на напряжение 1,5 В); 2 — щелочной типоразмера АА;
3 — традиционный марганец-цинковый типоразмера АА
(по данным И. Подушкина, «Радио», № 2, 2004)

Подробнее об особенностях электрохимических элементов мы поговорим в *главе 9*, а сейчас нам важно усвоить, что даже самое-самое постоянное напряжение на деле может быть совсем и не постоянным. Даже для самых качественных источников питания, таких как электрохимические элементы, оно обязательно немножко «гуляет» — в зависимости от тока нагрузки и ее характера. Что же тогда называть переменным напряжением? Строгого определения, как ни странно, не существует — часто приводимое в учебниках выражение «напряжение, которое изменяется с течением времени», как видите, прекрасно подходит и к нашим батарейкам, хотя они являются типичными источниками напряжения постоянного. Поэтому мы договоримся *переменными* называть такие напряжения или токи, которые изменяются во времени, во-первых, периодически, а, во-вторых, делают это «сами по себе», без влияния со стороны нагрузки и других внешних причин.

Слово «периодически» таким образом для нас означает, что, начиная с какого-то момента времени, форма графика той или иной величины повторяется снова и снова (хотя, возможно, и с некоторыми изменениями). Время повтора называется *периодом* переменной величины. Как вы хорошо знаете из школьного курса физики, наиболее простым и наглядным примером переменной периодической величины является величина, изменяющаяся во времени по синусоидальному закону.

На рис. 4.2 приведен график такой величины в зависимости от времени в условном масштабе. По оси ординат могут быть отложены как напряжение или ток, так и любой другой физический параметр. Отрезок времени T есть период изменения, а величина A носит название *амплитуды* и представляет собой максимальное значение нашей переменной в одном периоде (отметим, что для синусоидального закона минимальное значение — на части графика ниже оси абсцисс — по абсолютной величине строго равно максимальному). Величина, обратная периоду, обозначается буквой f и носит название *частоты* (см. формулу на рис. 4.2, *вверху*). Для нее придумана специальная единица измерения — это хорошо всем знакомый герц (Гц), названный так в честь немецкого физика XIX века Генриха Герца, доказавшего существование радиоволн. Как следует из определения частоты, размерность герца есть единица, деленная на секунду: 1 Гц = 1/с. Это просто-напросто означает, что колебание с частотой 1 Гц имеет период повторения ровно 1 секунду. Соответственно, 1 кГц (килогерц) означает, что в одной секунде укладывается тысяча периодов, 1 МГц (мегагерц) — миллион периодов и т. п.

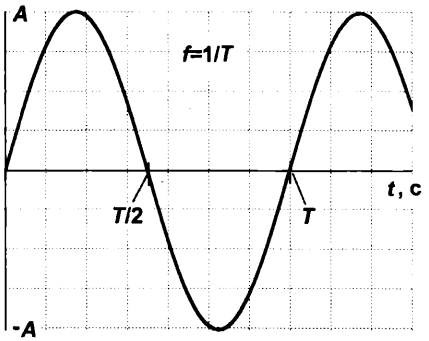


Рис. 4.2. График простого синусоидального колебания

В дальнейшем под «величиной» мы чаще всего будем иметь в виду напряжение (для тока все выглядит аналогично). Математический закон, описывающий поведение синусоидального напряжения (U) от времени (t), выглядит так:

$$U = A \cdot \sin(2\pi f t). \quad (1)$$

Здесь π есть хорошо нам знакомое число «пи», т. е. отношение длины окружности к ее диаметру, равное 3,1415... Произведение $2\pi f$ носит специальное название *круговая частота* и обозначается буквой ω (омега). Физический смысл круговой частоты — величина угла (измеряемого в радианах), пробегаемого нашей синусоидальной кривой за секунду. Поскольку мы обещали не заниматься радиочастотной техникой, то углубляться в дальнейшие абстракции вроде представления переменных колебаний через комплексные числа, где понятие круговой частоты является ключевым, мы не станем — для практических нужд нам пока хватит и более наглядных определений обычной частоты через период.

А что будет, если график немного подвигать вдоль оси абсцисс? Как видно из рис. 4.3, это равносильно признанию того факта, что в нулевой момент времени наше колебание не равно нулю. На рис. 4.3 второе колебание начинается с максимального значения амплитуды, а не с нуля. При этом сдвигаются моменты времени, соответствующие целому и половине периода, а в уравнении (1) появляется еще одна величина, обозначаемая буквой φ (фи) и измеряемая в единицах угла — радианах:

$$U = A \cdot \sin(2\pi f t + \varphi). \quad (2)$$

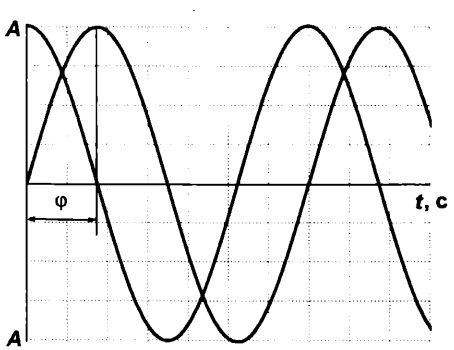


Рис. 4.3. График синусоидальных колебаний, сдвинутых по фазе на четверть периода

Эта величина носит название *фазы*. Взятая для одного отдельного колебания, величина фазы выглядит не имеющей особого смысла, так как мы всегда можем сместить точку начала отсчета времени так, чтобы привести уравнение к виду (1), а, соответственно, график — к виду рис. 4.2, и при этом ничего не изменится. На экране осциллографа это можно сделать очень наглядно — поворотом ручки «Уровень» (имеется в виду уровень синхронизации). Однако все будет выглядеть иначе, если мы имеем два связанных между собой колебания — скажем, напряжения в разных точках одной схемы. В этом случае нам может быть важно, как соотносятся их величины в каждый момент времени, и тогда фаза одного переменного напряжения относительно другого (называемая в этом случае *сдвигом* или *разностью*

фаз) и будет характеризовать такое соотношение. Для колебаний, представленных на рис. 4.3, сдвиг фаз равен 90° (π/2 радиан). Именно для наблюдения таких колебаний совместно и предназначен многоканальный или многолучевой осциллограф — в обычном фаза колебания определяется только настройками синхронизации.

Интересно, что получится, если мы такие «сдвинутые» колебания суммируем? Не надо думать, что это есть лишь теоретическое упражнение — суммировать электрические колебания разного вида нам придется довольно часто. Математически это будет выглядеть, как сложение формул (1) и (2):

$$U = A_1 \cdot \sin(2\pi f_1 t) + A_2 \cdot \sin(2\pi f_2 t + \varphi). \quad (3)$$

Обратите внимание, что в общем случае амплитуды и частоты колебаний различны (на рис. 4.3 они одинаковы!).

Чтобы представить себе результат наглядно, надо проделать следующее: разделить период колебаний на некоторое количество отрезков и для каждого отрезка сложить величины колебаний (естественно, с учетом знака), а затем построить график по полученным значениям. На компьютере можно написать программу, которая вычисляет значения по формуле (3) и строит соответствующие графики. Конечно, можно и не писать собственную программу, а использовать готовую, — скажем, Excel прекрасно умеет выполнять подобные операции.

Для иллюстрации продемонстрируем (рис. 4.4), что получится, если сложить два колебания, которые были представлены на рис. 4.3. Я не буду приводить картинки для иных случаев, так как интересных комбинаций может быть довольно много, но очень рекомендую потратить время на эти упражнения, потому что результаты могут быть весьма неожиданными и вовсе неочевидными. Скажем, при сложении двух синусоидальных колебаний с одинаковой частотой и амплитудой, но со сдвигом фаз в 180° (когда колебания находятся в противофазе), результирующая сумма будет равна нулю на всем протяжении оси времени! А если амплитуды таких колебаний не равны друг другу, то в результате получится такое же колебание, амплитуда которого в каждой точке равна разности амплитуд исходных. Запомним этот факт — он нам пригодится, когда мы будем рассматривать усилители звуковой частоты с обратной связью (см. *главу 8*).

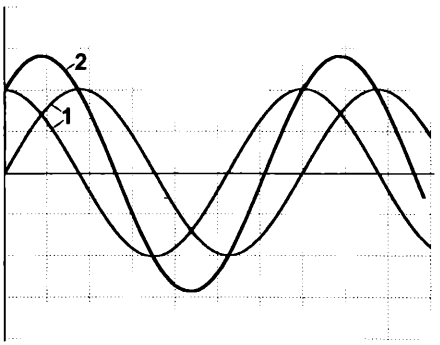


Рис. 4.4. Суммирование колебаний, сдвинутых по фазе на четверть периода:
1 — исходные колебания; 2 —.их сумма

Можно ли проверить на практике это положение? Для этого нам придется немного забежать вперед: потребуется сетевой трансформатор с двумя вторичными обмотками. Обмотки эти нужно соединить последовательно так, чтобы конец одной обмотки соединялся с концом другой (как находить начала и концы обмоток трансформатора, будет рассказано в *главе 9*). В обмотках трансформатора напряжения имеют одинаковую частоту и фазу, зависящую от способа их соединения, — если соединить так, как указано (конец с концом), то сдвиг фаз составит ровно 180°, т. е. мы воспроизведем условия нашего эксперимента. Теперь осталось только включить трансформатор в сеть и присоединить к свободным выводам обмоток вольтметр (естественно, настроенный для измерения переменного напряжения). Мы получим именно то, что предсказано расчетом: если обмотки одинаковые (т. е. амплитуды напряжений в них одни и те же), то вольтметр не покажет ничего — несмотря на то, что сами напряжения в обмотках могут быть сколь угодно велики! Если же обмотки имеют разное количество витков, то результат измерения будет равен разности напряжений. Комбинируя различные обмотки таким образом, мы можем заставить трансформатор выдавать напряжения, которые в нем вовсе не были предусмотрены!

А вот вопрос на засыпку — что показывал вольтметр в предыдущем эксперименте? Ведь измеряемая величина все время, с частотой 50 раз в секунду, меняется от отрицательного до точно такого же положительного значения, т. е. в среднем напряжение строго равно нулю — и тем не менее, вольтметр нам показывал совершенно определенное значение. Для ответа на этот вопрос отвлечемся от колебаний и поговорим о еще одной важнейшей величине, которая характеризует электрический ток, — о мощности.

Мощность

Согласно определению, *мощность* есть энергия (работа), выделяемая в единицу времени. Единица мощности называется ватт (Вт). По определению, 1 ватт есть такая мощность, при которой за 1 секунду выделяется (или затрачивается — смотря с какой стороны поглядеть) 1 джоуль энергии. Для электрической цепи ее очень просто подсчитать по закону Джоуля — Ленца:

$$N\,(\text{ватт}) = U\,(\text{вольт}) \cdot I\,(\text{ампер})$$

Эту формулу несложно вывести из определений тока и напряжения (см. *главу 1*). Действительно, размерность напряжения есть джоуль/кулон, а размерность тока — кулон/секунду. Если их перемножить, то кулоны сокращаются и получаются джоули в секунду — что, согласно приведенному ранее определению, и есть мощность. Если подставить в формулу для электрической мощности выражения связи между током и напряжением по закону Ома, то можно вывести еще два часто употребляющихся представления закона Джоуля — Ленца:

$$N = I^2 \cdot R \text{ и } N = U^2/R$$

Обратите внимание на одно важное следствие из этих формул — мощность в цепи пропорциональна квадрату тока или напряжения. Это означает, что если повысить

напряжение на некоем резисторе вдвое, то мощность, выделяющаяся на нем, возрастет вчетверо.

А вот от сопротивления мощность зависит линейно — если вы при том же источнике питания уменьшите сопротивление вдвое, то мощность в нагрузке также возрастет только вдвое. Это именно так, хотя факт, что согласно закону Ома ток в цепи увеличится также вдвое, мог бы нас привести к ошибочному выводу, будто в этом случае выделяющаяся мощность возрастет вчетверо. Но если вы внимательно проанализируете формулировку закона Джоуля — Ленца, то поймете, где здесь зарыта собака, — ведь в произведении $U \cdot I$ увеличивается только ток, а напряжение остается тем же самым.

В электрических цепях энергия выступает чаще всего в роли тепловой энергии, поэтому электрическая мощность в подавляющем большинстве случаев физически означает просто количество тепла, которое выделяется в цепи (если в ней нет электромоторов, или, скажем, источников света). Вот и ответ на вопрос, который мог бы задать пытливый читатель еще при чтении первой главы, — куда расходуется энергия источника питания, гоняющего по цепи ток? Ответ — на нагревание сопротивлений нагрузки, включенных в сеть. И даже если нагрузка представляет собой, скажем, источник света (лампочку или светодиод), то большая часть энергии все равно уходит в тепло — КПД лампы накаливания (т. е. та часть энергии, которая превращается в свет), как известно, не превышает единиц процентов. У светодиодов эта величина заметно выше (порядка 10%), но и там огромная часть энергии уходит в тепло. Кстати, из всего этого следует, например, что ваш компьютер последней модели, который потребляет сотни ватт энергии, также всю эту энергию переводит в тепло — за исключением исчезающе малой ее части, которая расходуется на свечение экрана и вращение жесткого диска (впрочем, энергия вращения тоже в конце концов переходит в тепло). Такова цена информации!

Если мощность, выделяемая на нагрузке, превысит некоторую допустимую величину, то нагрузка просто сгорит. Поэтому различные типы нагрузок характеризуют *предельно допустимой мощностью*, которую они могут рассеять без необратимых последствий. Подробнее об этом для разных видов нагрузок мы поговорим в дальнейшем, а сейчас зададимся вопросом — что означает мощность в цепях переменного тока?

Что показывал вольтметр?

Для того чтобы понять смысл этого вопроса, давайте внимательно рассмотрим график синусоидального напряжения на рис. 4.2. В каждый момент времени величина напряжения в нем разная — соответственно, будет разной и величина тока через резистор нагрузки, на который мы подадим такое напряжение. В моменты времени, обозначенные $T/2$ и T (т. е. кратные половине периода нашего колебания), напряжение на нагрузке вообще будет равно нулю (ток через резистор не течет), а в промежутках между ними — меняется вплоть до некоей максимальной величины, равной амплитудному значению A. Точно так же будет меняться ток через нагрузку, а, следовательно, и выделяемая мощность (которая от направления тока не зави-

сит, — физики скажут, что мощность есть величина скалярная, а не векторная). Но процесс выделения тепла крайне инерционен — даже такой маленький предмет, как волосок лампочки накаливания, за 1/100 секунды, которые проходят между пиками напряжения в промышленной сети частотой 50 Гц, не успевает заметно остыть. Поэтому нас чаще всего интересует именно средняя мощность за большой промежуток времени. Чему она будет равна?

Чтобы точно ответить на этот вопрос, нужно брать интегралы, — средняя мощность за период есть интеграл по времени от квадрата функции напряжения. Здесь мы приведем только результат — величина средней мощности в цепи переменного тока определяется так называемым *действующим значением напряжения* ($U_д$), которое для синусоидального колебания связано с амплитудным его значением ($U_а$) следующей формулой: $U_а = U_д \cdot \sqrt{2}$ (вывод этой формулы приведен в *главе 8*). Точно такая же формула справедлива и для тока. Когда говорят «переменное напряжение 220 В», то всегда имеется в виду именно действующее значение. При этом амплитудное значение равно примерно 310 В, что легко подсчитать, если умножить 220 на корень из двух. Это значение нужно всегда иметь в виду при выборе компонентов для работы в сетях переменного тока — если взять диод, рассчитанный на 250 В, то он легко может выйти из строя при работе в обычной сети, в которой мгновенное значение превышает 300 В, хотя действующее значение и равно 220 В. А вот для компонентов, использующих эффект нагревания (лампочек, резисторов и т. п.), при расчете допустимой мощности нужно иметь в виду именно действующее значение.

Называть действующее значение «средним» неверно, правильно называть его *среднеквадратическим* (по способу вычисления — через квадрат функции от времени). Но существует и понятия среднего значения, причем не одно, а даже два. Просто *среднее* (строго по смыслу названия) — сумма всех мгновенных значений за период. И так как нижняя часть синусоиды (под осью абсцисс) строго симметрична относительно верхней, то можно даже не брать интегралов, чтобы сообразить, что среднее значение синусоидального напряжения, показанного на рис. 4.2, в точности равно нулю, — положительная часть компенсирует отрицательную. Но такая величина малоинформативна, поэтому чаще используют *средневыпрямленное* (среднеамплитудное) значение, при котором знаки не учитываются (т. е. в интеграл подставляется абсолютная величина напряжения). Эта величина ($U_с$) связана с амплитудным значением ($U_а$) по формуле $U_а = \pi \cdot U_с/2$, т. е. $U_а$ равно примерно $1{,}57 \cdot U_с$.

Для постоянного напряжения и тока действующее, среднее и среднеамплитудное значения совпадают и равны просто величине напряжения (тока). Однако на практике часто встречаются переменные колебания, форма которых отличается и от постоянной величины, и от строго синусоидальной. Осциллограммы некоторых из них показаны на рис. 4.5. Для таких сигналов приведенные ранее соотношения для действующего и среднего значений недействительны! Самый простой случай изображен на рис. 4.5, *в* — колебание представляет собой синусоиду, но сдвинутую вверх на величину амплитуды. Такой сигнал можно представить как сумму постоянного напряжения величиной A (постоянная составляющая) и переменного сину-

соидального (переменная составляющая). Соответственно, среднее значение его будет равно A, а действующее $A+A/\sqrt{2}$. Для прямоугольного колебания (рис. 4.5, *б*) с равными по длительности положительными и отрицательными полуволнами (симметричного меандра[1]) соотношения очень просты: действующее значение равно среднеамплитудному, как и для постоянного тока, а вот среднее значение равно, как и для синуса, нулю.

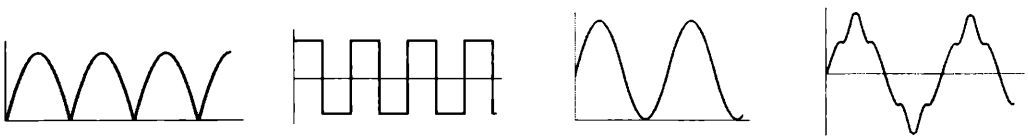


Рис. 4.5. Графики некоторых колебаний несинусоидальной формы

В часто встречающемся на практике случае, когда минимум прямоугольного напряжения совпадает с нулем, т. е. напряжение колеблется от нуля до напряжения питания (на рис. 4.5 не показано), такой меандр можно рассматривать аналогично случаю, показанному на рис. 4.5, *в*, — как сумму постоянного напряжения и прямоугольного. Для случая, показанного на рис. 4.5, *а*, который представляет собой синусоидальное напряжение, пропущенное через двухполупериодный выпрямитель (см. *главу 9*), действующее и среднеамплитудное значения будут равны соответствующим значениям для синусоиды, а вот среднее будет равно не нулю, а совпадать со среднеамплитудным. Для случая, приведенного на рис. 4.5, *г*, указать все эти величины вообще непросто, так как они зависят от формы сигнала.

Но, даже выучив все это, вы все равно не сможете измерять величины напряжений и токов несинусоидальной формы с помощью мультиметра! Не забывайте об этом, как и о том, что для каждого мультиметра есть предельные значения частоты колебаний, — если вы включите мультиметр в цепь с иными параметрами, он может показать все, что угодно, — «погоду на Марсе», по распространенному выражению. Измерительные приборы для переменного напряжения проградуированы в значениях действующего напряжения, но измеряют они, как правило, среднеамплитудное (по крайней мере, большинство — на подробностях мы не будем сейчас задерживаться), и сообразить, как именно пересчитать показания, далеко не всегда просто. А для сложных сигналов, как на рис. 4.5, *г*, это выливается в сущую головоломку на уровне задач для студентов мехмата. Выручить может осциллограф и знание соотношений, приведенных ранее для сигналов самой распространенной формы, ну а для более сложных вычислять действующие и средние значения нам и не потребуется.

Заметки на полях

Единственный прибор, который правильно покажет значение действующего напряжения любой формы, — это аналоговый вольтметр электромагнитной системы (их легко

[1] Меандр — тип геометрического узора с повторяющимися ломаными линиями (по названию извилистой реки Меандр в Малой Азии).

узнать по неравномерной шкале, деления на которой по мере приближения к нулю размещаются все ближе друг к другу). Для того чтобы несинусоидальное напряжение измерить цифровым прибором, между измеряемой величиной и вольтметром можно вставить интегрирующий фильтр (фильтр нижних частот), описанный в *главе 5*.

Для прямоугольных напряжений, представляющих собой меандр, подобный показанному на рис. 4.5, *б*, существует еще одна важная характеристика. Никто ведь не запрещает представить себе прямоугольное напряжение, в котором впадины короче или длиннее всплесков. В электронике меандр без дополнительных пояснений означает симметричную форму прямоугольного напряжения, при которой впадины строго равны всплескам по длительности, но, вообще говоря, это необязательно. На рис. 4.6 приведены два примера таких напряжений в сравнении с симметричным меандром. Характеристика соотношений между длительностями частей периода называется *скважностью* и определяется как *отношение длительности всего периода к длительности положительной части* (именно так, а не наоборот, т. е. величина скважности всегда больше 1). Для меандра скважность равна 2, для узких коротких импульсов она будет больше 2, для широких — меньше.

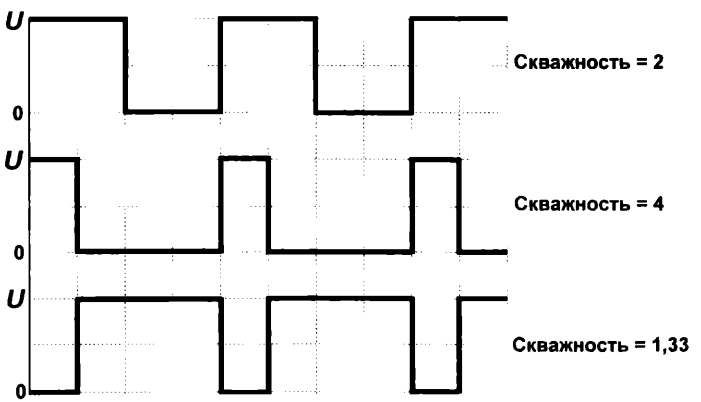


Рис. 4.6. Примеры напряжений в сравнении с симметричным меандром

Сигналы

Несколько слов о сигналах. Электрический сигнал, по смыслу его названия, — какое-то состояние электрической цепи, которое несет информацию. Различают источники сигналов и их приемники. Так как минимальное количество информации (1 бит) подразумевает по крайней мере два различимых состояния (подробнее об этом будет идти речь в *главе 14*), то и сигнал должен иметь как минимум два состояния. Еще со времен телеграфа Морзе известен самый простой сигнал: наличие или отсутствие постоянного напряжения или тока в цепи. Именно такими сигналами обмениваются логические микросхемы. Однако на большое расстояние такой простейший сигнал не передашь — слишком сложно защититься от помех, из-за которых приемник легко может обнаружить наличие сигнала там, где на самом деле всего лишь помеха или, наоборот, пропустить нужный импульс. Поэтому при-

думывают разные сложные методы — некоторые из них, например, предусматривают передачу переменного напряжения разной частоты или фазы (именно так устроены модемы). Теория передачи сигналов тесно связана с теорией колебаний — одно только радио чего стоит!

Подробнее о разных сигналах мы будем говорить в соответствующих главах, а сейчас нам важно только одно, — когда мы говорим о сигналах, то подразумеваем, что соответствующее напряжение или ток не предназначено для совершения иной работы, кроме как заставить сработать приемник. Потому мощности, передаваемые при пересылке сигналов, значительно меньше, чем при передаче электроэнергии для совершения полезной работы. Действительно — никто еще не придумал, как питать, скажем, спутники на орбите по радиолучу, а вот информация передается вполне успешно даже за пределы Солнечной системы. В этом и заключается разница между силовыми и сигнальными цепями (если помните, то в *главе 3* мы даже специально отмечали, что проводники питания следует делать как можно толще, а для сигнальных цепей это необязательно). И понимание этого тонкого различия очень пригодится нам в дальнейшем изложении.

О переменном токе и электропитании

Кстати, отдельный вопрос: а почему нам вообще надо возиться с переменным током, как основой электропитания? Сколько можно было бы сэкономить на трансформаторах и сглаживающих фильтрах, которые зачастую составляют большую часть габаритов и стоимости схемы! Недаром ненавидящие это дело схемотехники и дизайнеры в последнее время полюбили готовые выносные блоки питания, объединенные с сетевой вилкой, — крайне некрасивое решение, которое просто переносит головную боль о габаритах с плеч разработчиков на плечи потребителей.

Дело в том, что никаких других эффективных первичных генераторов электроэнергии (тех, что преобразуют энергию вращения ротора водяной или паровой турбины в электричество на электростанциях), кроме как вырабатывающих переменный ток, не придумали. Интересно, что многие линии электропередач в мире делают-таки на постоянном (точнее, выпрямленном, т. е. пульсирующем) токе, что позволяет во многом избежать реактивных потерь в проводах (см. *главу 5*). При этом приходится сначала преобразовывать переменный ток в постоянный, а затем производить обратное преобразование, которое куда сложнее, — исключительно для того, чтобы состыковать имеющиеся линии электропередач со стандартными.

Аналогичная задача, только в меньших масштабах, стоит перед разработчиками источников бесперебойного питания (UPS) — питающий ток из сети нужно преобразовать в постоянный для зарядки низковольтного (12 или 24 вольта) резервного аккумулятора, а в случае пропадания сетевого питания это напряжение аккумулятора следует опять преобразовать к стандартному виду переменного сетевого напряжения, причем желательно, чтобы форма его была максимально близка к синусоидальной (такое преобразование называется *инверсией*). Приходится поломать голову, чтобы компьютер, питающийся через UPS, не заметил подмены!

Децибелы

Эрудированный читатель, несомненно, отметит, что я почти не употребляю в этой книге такой распространенной единицы измерения, как децибелы (дБ). Это вызвано некоторыми трудностями в их определении и понимании, потому по возможности мы постараемся их избегать. Но в некоторых случаях децибелы нам понадобятся, потому иметь представление о том, что это такое, необходимо.

Децибел (одна десятая бела, названного так по имени изобретателя телефона Александра Белла) есть единица, использующаяся для измерения отношений величин. Перевести отношение в децибелы и обратно можно по формуле:

$$K\,(\text{дБ}) = 20 \cdot \lg(A_1/A_0),$$

где A_1/A_0 — есть отношение значений некоторых амплитуд (напряжений, токов, амплитуд колебаний воздуха или воды при распространении звука и т. п.).

Новичков очень смущает то, что в радиотехнике (например, при сравнении интенсивностей радиосигнала в разных точках) применяют несколько иные — «мощностные» — децибелы, для которых величина A должна иметь размерность энергии (или мощности), и формула приобретает иной вид:

$$1\ \text{dB} = 10\ \lg(A/A_0)$$

В этой книге мы всегда будем иметь в виду «амплитудные» децибелы — например, коэффициент усиления звукового усилителя, равный 20 dB, будет означать, что напряжение на выходе будет в 10 раз больше напряжения на входе: $U_{\text{вых}}/U_{\text{вх}} = 10^{\text{dB}/20}$ (для «мощностных» децибел величина 20 означает изменение мощности сигнала в 100 раз).

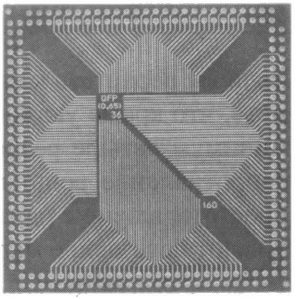
Децибелы удобно использовать для характеристики изменения величин, меняющихся по степенному закону. Их широко используют при расчетах фильтров, анализе частотных и амплитудных характеристик операционных усилителей (ОУ). График степенной функции, которая быстро возрастает или падает в обычных координатах, в широком диапазоне значений практически невозможно изобразить, а при использовании децибел он будет выглядеть прямой линией (это часто встречающиеся вам графики, где по осям отложены величины, возрастающие не линейно, а в геометрической прогрессии: 1, 10, 100, 1000...). В акустике звуковое давление практически всегда измеряют в «амплитудных» децибелах (относительно порога слышимости) — это связано с тем, что наше ухо реагирует именно на отношение громкостей, а не их абсолютное возрастание. Так, болевой порог звука, определяемый в 120 дБ, означает интенсивность звука в миллион раз выше порога слышимости.

Если отношение величин больше 1, то величина в децибелах будет положительной, если меньше — отрицательной. Для перевода «амплитудных» децибел в обычные относительные единицы и обратно необязательно всегда использовать указанную ранее формулу, достаточно просто запомнить несколько приблизительно выполняющихся соотношений:

- 3 дБ соответствует увеличению/уменьшению на треть;
- 6 дБ соответствует отношению в 2 раза;

- 10 дБ соответствует отношению в 3 раза;
- 20 дБ соответствует отношению в 10 раз.

Руководствуясь этими соотношениями, легко перевести любую величину, выраженную в децибелах, — например, 73 дБ есть 20 + 20 + 20 + 10 + 3 дБ, что соответствует отношению в 10·10·10·3·1,33 = 4000 раз. Собственный коэффициент микросхемы звукового усилителя TDA2030 (см. *главу 11*) равен 30 000, т. е. $3 \cdot 10^4$, или 10 + 4·20 = 90 дБ, а максимальный рекомендуемый коэффициент усиления усилителя на ее основе, согласно техническому описанию, равен 46 = 20 + 20 + 6 дБ, что соответствует усилению в 200 раз. Коэффициент ослабления синфазного сигнала (КОСС), о котором речь пойдет в *главе 12*, также чаще всего измеряют в децибелах: так, его величина, равная −60 (= −3·20) дБ, означает, что синфазный сигнал ослабляется в 1000 раз. Крутизна характеристик простейших RC-фильтров низкой и высокой частоты из *главы 5* равна, соответственно, −6 и +6 дБ на октаву, что означает уменьшение/увеличение сигнала в 2 раза при изменении частоты также в 2 раза.



ГЛАВА 5

Электроника без полупроводников

Резисторы, конденсаторы и схемы на их основе

> Глаза миледи метали такие молнии, что, хотя лорд Винтер был мужчина и стоял вооруженный перед беззащитной женщиной, он почувствовал, как в душе его зашевелился страх.
>
> *А. Дюма. «Три мушкетера»*

Резисторы и конденсаторы — основа основ электроники. Эти элементы вместе с индуктивностями относятся к разряду так называемых *линейных*, потому что ток и напряжение в них зависят друг от друга по линейному закону, и образуют группу пассивных элементов. Диоды, транзисторы и прочие нелинейные компоненты относят к *активным* элементам. Эти названия произошли от того, что эквивалентные схемы нелинейных элементов не могут обойтись без включения в них источников тока или напряжения (активных составляющих). Так как практически ни одна схема без резисторов и конденсаторов не обходится, то мы рассмотрим их свойства подробно.

Резисторы

Резистор — самый распространенный компонент электронных схем. Несмотря на его простоту (в самом деле — это всего-навсего кусок материала с определенным сопротивлением), не существует практически ни одной работоспособной схемы, в которой бы не присутствовали резисторы в том или ином виде. Даже если вы их и специально не ставили, они все равно есть. Скажем, в простейшем случае настольной лампы или карманного фонарика, где вся схема состоит из источника питания (сети или батарейки), выключателя и лампочки, резисторы неявно присутствуют — это и нить лампочки, которая светится, нагреваясь за счет своего высокого сопротивления, и сопротивление проводов, и внутреннее сопротивление источника питания. Все эти элементы могут быть представлены на схеме, как резисторы. Причем последние два элемента из перечисленных только мешают, забирая на себя часть полезной мощности, но избавиться от них невозможно, они присутствуют всегда и везде, поэтому их нужно учитывать и стараться свести их влияние к минимуму.

Если вы вернетесь к рис. 1.4 в *главе 1*, то при внимательном его рассмотрении поймете, что кроме указанных на схеме резисторов R1 и R2 в деле участвуют еще как минимум четыре резистора: сопротивление проводов, сопротивление амперметра, сопротивление вольтметра и внутреннее сопротивление источника питания. Для простоты влияние паразитных резисторов обычно игнорируют, считая, что они оказывают исчезающе малое влияние на работу схемы, однако это не всегда так.

Ко всем этим тонкостям мы еще будем возвращаться не раз, а пока рассмотрим резисторы, как таковые, — т. е. фабрично выпускаемые компоненты электронных схем под таким названием. Они встречаются разных типов, размеров и конструкций. Наиболее часто употребляемые типы — металлопленочные (металлодиэлектрические), или углеродистые резисторы. Наиболее распространены импортные углеродистые резисторы (CF, отечественный аналог С1-4) — они вполне заменяют старые отечественные металлопленочные (МЛТ), которые часто указываются на схемах. Отечественные МЛТ старых выпусков имеют обычно красный или розовый цвет (хотя иногда встречаются и другие цвета, например зеленый), а номинальное значение сопротивления написано прямо на них, в то время как современные резисторы маркируются международным цветным кодом. Есть и другие типы резисторов общего назначения. По функциональным свойствам все они практически идентичны.

В *приложении 1* приводится таблица цветных кодов для маркировки резисторов, но сам я практически этим кодом не пользуюсь. Читать цветной код неопытному человеку — мука мученическая, учитывая особенно, что понятие, скажем, «оранжевый» очень часто трактуется производителями весьма вольно, и отличить его от «желтого» на голубом, к примеру, фоне может только человек с большим опытом. Проще и быстрее просто измерить сопротивление мультиметром.

Таблицы рядов номинального сопротивления в зависимости от допустимого разброса значений (*допуска*), также приведенные в *приложении 1*, нужно пояснить. У непосвященных может возникнуть вопрос: почему резисторы имеют такие странные номинальные значения: 4,3 или 5,1 кОм? Почему нельзя привязать номиналы к привычным для нас «круглым» значениям: 4 или 5 кОм? Все объясняется очень просто.

Возьмем, например, широко распространенные резисторы с пятипроцентным допуском и посчитаем резистор 1 кОм за основу ряда. Какой следующий номинал взять? Так как допуск равен 5%, то в большой партии резисторов могут встретиться сопротивления во всем диапазоне: от 0,95 до 1,05 кОм. Мы, естественно, хотим, чтобы можно было бы (хотя бы теоретически), найти резистор с любым значением сопротивления. Поэтому следующий номинал, который мы выбираем, будет равен 1,1 кОм — а так как его допуск тоже 5%, то минимальное допустимое значение для него — 1,045 кОм, и, как мы видим, диапазоны перекрываются. Точно так же рассчитываются остальные номиналы, вплоть до 9,1 кОм, возможные значения которого перекрываются с допусками от первого значения из следующей декады — 10 кОм. Чем строже допуск, тем больше сопротивлений в ряду, — если мы встретим резистор с номинальным сопротивлением 2,43 кОм, то можем быть уверены, что допуск у него не хуже 1%. Конечно, для малых допусков (вроде 0,1%) ряд по-

лучился бы слишком большим, потому его ограничивают, и допуски там уже не пересекаются. Кстати, забегая вперед, отметим, что те же ряды значений справедливы и для емкости конденсаторов.

Осталось научиться вычислять значения сопротивления для всего диапазона выпускаемых промышленностью резисторов — для обычных типов это значения от 1 Ом до 10 МОм. Как вы уже догадались, в каждой декаде номиналы получаются из табличного ряда значений путем умножения на соответствующую степень десяти. При этом для краткости часто используют условные обозначения для каждого диапазона: R (или E) — обозначает омы, к — килоомы, М — мегомы. Эти буквы могут использоваться вместо десятичной точки: так, запись 1к2 есть то же самое, что и 1,2 кОм, а 3R3 (или 3E3) — то же самое, что 3,3 Ом. При обозначении на схемах целые омы в большинстве случаев вообще опускают — именно так мы будем поступать в этой книге, так что имейте в виду: запись «360» на схеме означает просто 360 Ом.

Хотя я не рекомендую иметь дело в домашних условиях с компонентами поверхностного монтажа (как их еще называют, чип-компонентами или SMD-компонентами), но рано или поздно они вам, безусловно, могут встретиться. Для SMD-резисторов принята другая система маркировки. Самые мелкие SMD-резисторы (допустимой мощностью 0,063 Вт) не маркируются вообще. Наиболее часто встречающиеся SMD-резисторы с допуском 2, 5 и 10% всех типоразмеров маркируются тремя цифрами. Первые две цифры обозначают мантиссу, а последняя цифра — показатель степени по основанию 10 для определения номинала в омах. Для обозначения десятичной точки к значащим цифрам может добавляться буква R. Например, маркировка 242 означает, что чип-резистор имеет номинал 24×10^2 Ом = = 2,4 кОм. Кодом «000» маркируют перемычки с нулевым сопротивлением.

> ***Заметки на полях***
>
> Забегая вперед, заметим, что на похожих принципах основаны обозначения емкости малогабаритных конденсаторов (и SMD, и обычных), только за основу шкалы там приняты пикофарады (10^{-12} Ф), так что надпись, скажем, 474 расшифровывается как $47\cdot10^4\cdot10^{-12} = 0{,}47\cdot10^{-6}$ фарады или 0,47 мкФ. Такая маркировка на корпусе характерна для керамических конденсаторов с гибкими выводами — электролитические конденсаторы маркируются обычной надписью, а наиболее часто встречающиеся типы малогабаритных керамических конденсаторов в SMD-исполнении, к сожалению, обычно не маркируются вовсе, и от резисторов их можно отличить только по цвету корпуса (светлых тонов, в отличие от черных резисторов) или по надписям на плате.

Гораздо реже встречаются прецизионные SMD-резисторы с допуском 1%. Крупные (0,5 Вт и более) такие резисторы маркируются четырьмя цифрами, которые читаются аналогично обычной маркировке, — например, 4752 означает, что чип-резистор имеет номинал 475×10^2 Ом = 47,5 кОм. Более мелкие маркируются двумя цифрами от 01 до 96 и буквой, и номинал можно определить по специальным таблицам.

Как уже было отмечено, обычные резисторы выпускаются и с 1%-ным разбросом, но практически в продаже встречаются только пятипроцентные разновидности. Более точные (прецизионные) резисторы с разбросом в 1% и ниже носят другие

наименования и значительно дороже. Дело в том, что простым отбором нельзя добиться того, чтобы номинал резистора укладывался в однопроцентный допуск — температурный коэффициент сопротивления (ТКС) для рядовых резисторов, как мы уже отмечали в *главе 1*, может составлять до 0,1% на каждый градус изменения температуры, причем у наиболее распространенных углеродистых типа С1-4 он преимущественно отрицательный, а так как резисторы при работе греются, то весь наш отбор пойдет насмарку. Поэтому прецизионные резисторы с малыми допусками имеют одновременно и значительно меньший температурный коэффициент. Причем это их качество — сохранять номинальное значение в большом диапазоне температур — значительно важнее, чем собственно точность номинала, — чаще всего нам не столь важно, чтобы, скажем, коэффициент усиления усилителя был равен в точности 3, сколько то, чтобы он не изменялся при изменениях температуры и со временем. Из отечественных прецизионных резисторов наиболее распространен тип С2-29В, который выпускают с допуском 0,05% и менее при ТКС от 0,0075%/°С до 0,03%/°С. Есть и более точные разновидности — например, проволочные С5-54В при допуске до 0,01% имеют ТКС не более 0,005%/°С, а тип С5-61 с ТКС не более 0,003%/°С встречается даже с допуском 0,005%. Имейте также в виду, что проволочные резисторы (типа С5-54В, к примеру) имеют очень узкий диапазон рабочих частот — фактически они предназначены только для постоянного тока.

Два слова о номинальной мощности резисторов. Резисторы с гибкими выводами выпускаются следующих предельно допустимых мощностей: 0,0625, 0,125, 0,25, 0,5, 1 и 2 Вт, которые отличаются размерами (см. рис. 5.1, *вверху*). В *приложении 5* вы можете посмотреть, как обозначаются на схемах резисторы разной мощности. Наиболее употребительные — резисторы мощностью 0,125 и 0,25 Вт, которые имеют близкие (а иногда и вовсе одинаковые) размеры, и если обозначение мощности на схеме отсутствует, то обычно имеется в виду именно мощность в 0,125–0,25 Вт. Учтите также, что прецизионные резисторы при той же допустимой мощности имеют больший размер — так, С2-29В мощностью 0,25 Вт выглядит, как полуваттный обычный. В подавляющем большинстве случаев замена на резистор большей мощности может только приветствоваться.

Надо учитывать, что резисторы имеют и предельно допустимое напряжение, которое также зависит от их размеров, — так, для мощностей 0,125 и 0,25 Вт это напряжение не превышает 250 В, поэтому их нельзя употреблять в цепях с сетевым питанием, — независимо от того, что тепловая мощность может быть и не превышена. Для цепей, в которых «гуляют» переменные напряжения с действующим напряжением 200 В и выше, минимально допустимая мощность резистора — 0,5 Вт.

Резисторы мощностью более 2 Вт требуются нечасто, и для таких случаев ранее выпускались специальные проволочные резисторы, залитые термостойким составом (их часто называют «остеклованными»). Современные резисторы большой мощности чаще всего имеют прямоугольный керамический корпус в виде бруска сероватого цвета, из которого по торцам торчат выводы. Номинальное сопротивление написано на одной из сторон такого корпуса. Резисторы номиналом меньше, чем 1 Ом, предназначенные обычно для пропускания больших токов, также, как

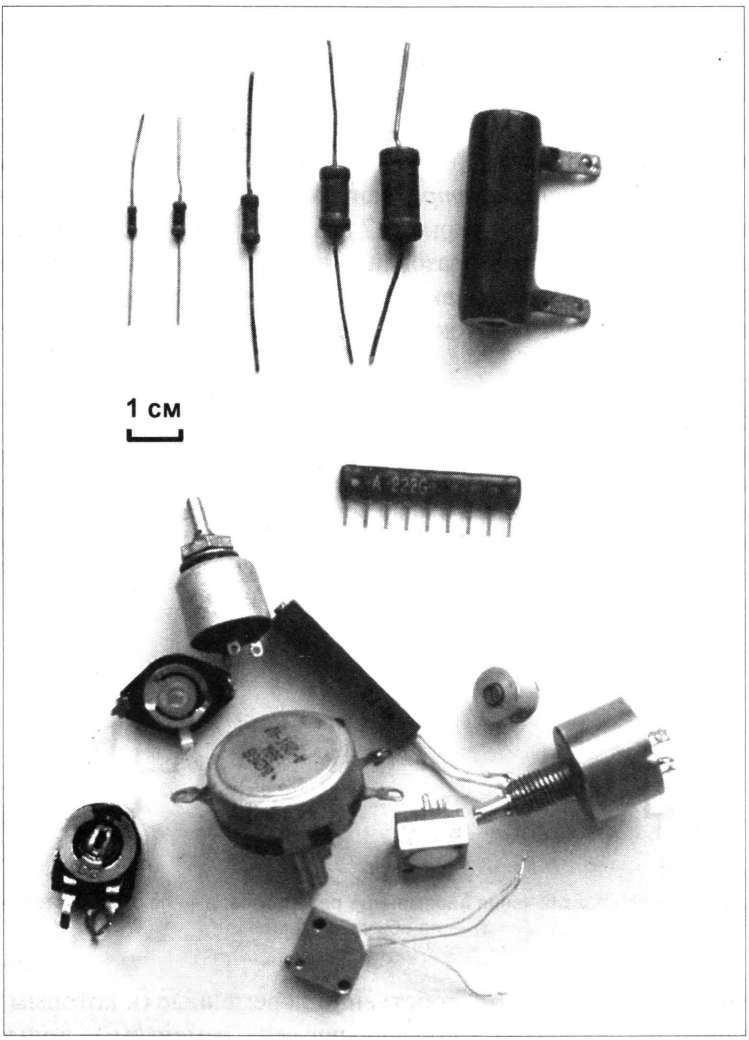


Рис. 5.1. Различные типы резисторов: *вверху* — сравнительные размеры резисторов разной мощности (слева направо: МЛТ 0,125; 0,25; 0,5; 1; 2 Вт; остеклованный резистор 10 Вт); в *середине* — резисторная сборка из восьми резисторов 0,125 Вт в одном корпусе SIP; *внизу* — различные переменные и подстроечные резисторы

правило, имеют большие размеры и выпускаются в очень разнообразном исполнении.

Переменные резисторы

Переменные резисторы отличаются от постоянных наличием третьего вывода — движка, представляющего собой подпружиненный ползунок, который может механически передвигаться по резистивному слою. Соответственно, в одном крайнем положении движка сопротивление между его выводом и одним из выводов резистивного слоя равно нулю, в другом — максимуму, соответствующему номинальному сопротивлению.

В переводной литературе переменные резисторы часто именуют потенциометрами, что не совсем правильно, т. к. название «потенциометр» относится только к одной из двух возможных схем включения переменников, причем реже употребляемой. Поскольку вывода три, то переменный резистор может подключаться двумя способами: как простой резистор (тогда вывод движка объединяется с одним из крайних выводов), и по схеме *потенциометра*, когда все три вывода задействованы. Оба способа подключения показаны на рис. 5.2 *а* и *б*. Переменные резисторы по своему предназначению служат для преобразования напряжения в ток и обратно — в соответствии с этим схема обычного включения переменного резистора служит для преобразования напряжения U в ток I, а схема потенциометра (делителя напряжения) — тока I в напряжение U.

Подробности

Кажется, что в схеме обычного включения необязательно соединять вывод движка с одним из крайних выводов, — если оставить незадействованный крайний вывод «висящим в воздухе», то ничего в принципе не изменится. Но это не совсем так — на «висящем» выводе возникают наводки от «гуляющего» в пространстве электрического поля, и правильно подключать переменный резистор следует именно так, как показано на рис. 5.2, *б*.

Рис. 5.2. Два способа подключения переменных резисторов: *а* — по схеме потенциометра; *б* — по обычной схеме

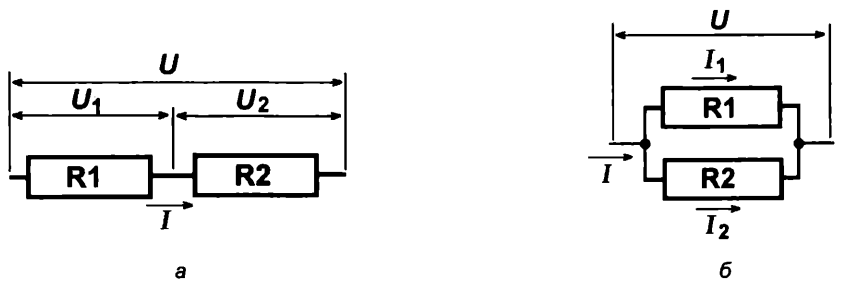
Переменные резисторы делятся на собственно переменные (к которым подсоединена ручка внешней регулировки) и *подстроечные* — изменяемые только в процессе настройки схемы путем вращения движка отверткой (см. рис. 5.1, *внизу*). Переменные резисторы мало изменились за все время своего существования еще со времен реостата Майкла Фарадея, и всем им присущ один и тот же недостаток — нарушение механического контакта между ползунком и резистивным слоем. Особенно это касается дешевых открытых подстроечных резисторов типа СП3-1 (на рис. 5.1 *внизу* два крайних типа *слева*) — представьте себе работу этого резистора, например, в телевизоре, находящемся в атмосфере домашней кухни!

Поэтому, если есть возможность, применения переменных резисторов следует избегать или ставить их последовательно с постоянными так, чтобы они составляли только необходимую часть всей величины сопротивления. Подстроечные резисторы хороши на стадии отладки схемы, а затем лучше заменить их постоянными и предусмотреть на плате возможность подключения параллельных и/или последовательных постоянных резисторов для окончательной подстройки. От внешних переменных резисторов (вроде регулятора громкости приемника), казалось бы, никуда

не денешься, но и это не так — использование аналоговых регуляторов с цифровым управлением дает отличную альтернативу переменникам в схемах с управлением от микроконтроллеров. В обычных схемах, по возможности, следует вместо переменного резистора ставить многопозиционный ступенчатый переключатель — так гораздо надежнее.

Параллельное и последовательное соединение резисторов

Это хотя и довольно простая тема, но очень важная. Правил всего два: при последовательном соединении (рис. 5.3, *а*) складываются сопротивления резисторов, а при параллельном (рис. 5.3, *б*) — их проводимости, которые, по определению из *главы 1*, есть величины, обратные сопротивлению. Понять, почему правила именно таковы, можно, если рассмотреть течение токов в обоих случаях: при последовательном соединении ток I через резисторы один и тот же, поэтому падения напряжения на них складываются ($U = U_1 + U_2$), что равносильно сложению сопротивлений. При параллельном соединении, наоборот, равны падения напряжений U, а складывать приходится токи ($I = I_1 + I_2$), что равносильно сложению проводимостей. Если вы не поняли сказанного, то посидите над рис. 5.3 с карандашом и бумагой и выведите выражения закона Ома для каждого из случаев, — и все станет на свои места.



а *б*

Рис. 5.3. Последовательное (*а*) и параллельное (*б*) соединение резисторов

Из приведенных общих правил вытекает несколько практических, которые полезно заучить:

❒ *при последовательном соединении:*

- пара резисторов имеет сопротивление всегда больше, чем сопротивление резистора с большим номиналом (правило «больше большего»);
- если номиналы резисторов равны, то суммарное сопротивление ровно вдвое больше каждого номинала;
- если номиналы резисторов различаются во много раз, то общее сопротивление примерно равно большему номиналу (типичный случай упоминался в *главе 1* — в примере на рис. 1.4 мы игнорируем сопротивление проводов, т. к. оно много меньше сопротивления резисторов);

- *при параллельном соединении:*
  - пара резисторов имеет сопротивление всегда меньше, чем сопротивление резистора с меньшим номиналом (правило «меньше меньшего»);
  - если номиналы резисторов равны, то суммарное сопротивление ровно вдвое меньше каждого номинала;
  - если номиналы резисторов различаются во много раз, то общее сопротивление примерно равно меньшему номиналу (это также можно проиллюстрировать на примере рис. 1.4, где мы игнорируем наличие вольтметра, включенного параллельно R2, т. к. его сопротивление намного больше сопротивления резистора).

Знание этих правил поможет вам быстро оценивать схему, не занимаясь алгебраическими упражнениями и не прибегая к помощи калькулятора. Даже если соотношение сопротивлений не попадает под перечисленные случаи, результат все равно можно оценить «на глаз» с достаточной точностью. При параллельном соединении, которое представляет бóльшую сложность при расчетах, для такой оценки нужно прикинуть, какую долю меньшее сопротивление составляет от их арифметической суммы, — приблизительно во столько раз снизится их общее сопротивление по отношению к меньшему. Проверить это легко: пусть одно сопротивление имеет номинал 3,3 кОм, а второе — 6,8 кОм. В соответствии с изложенным мы будем ожидать, что общее сопротивление должно быть на 30% меньше, чем 3,3 кОм, т. е. 2,2 кОм (3,3 составляет примерно одну треть от суммы 3,3 + 6,8, т. е. общее сопротивление должно быть меньше, чем 3,3, на треть от этого значения, равную 1,1 — в результате и получаем 2,2). Если мы проверим результат, полученный такой прикидкой в уме, точным расчетом, то получим в итоге очень близкое значение 2,22 кОм.

В большинстве случаев нам такой точности и не требуется — помните, что и сами сопротивления имеют разброс по номиналу, и в большинстве обычных схем допуски на номиналы стандартных компонентов могут быть довольно велики (по крайней мере, в правильно составленных схемах). Если же схема в некоторых случаях должна все же иметь какие-то строго определенные параметры, то с помощью стандартных компонентов вы все равно этого не добьетесь — параметры будут «гулять» в пределах допусков от дуновения ветерка из форточки, и если нет другого выхода, придется применять прецизионные резисторы и конденсаторы, а во времязадающих цепях использовать кварцевые резонаторы. Но составлять схему так, чтобы она теряла работоспособность от замены резистора 1 кОм на резистор 1,1 кОм, — не наш метод!

Конденсаторы

Все конденсаторы ведут свою родословную от лейденской банки (рис. 5.4), названной так по имени голландского города Лейдена, в котором трудился ученый середины XVIII века Питер ван Мушенбрук.

Банка эта представляла собой большой стеклянный стакан, обклеенный изнутри и снаружи станиолем (тонкой оловянной фольгой, использовавшейся в те времена для тех же целей, что и современная алюминиевая, — металл алюминий еще не был известен). Так как банку заряжали от электростатической машины (другого искусственного источника электричества тогда еще не придумали), которая запросто может выдавать напряжения в несколько сотен тысяч вольт, действие ее было весьма впечатляющим, — в учебниках физики любят приводить случай, когда Мушенбрук продемонстрировал эффект от разряда своей банки через цепь гвардейцев, держащихся за руки. Ну не знали тогда, что электричество может и убить, — гвардейцам сильно повезло, что емкость этого примитивного конденсатора была весьма невелика, и запасенной энергии хватило только на то, чтобы люди ощутили чувствительный удар током!

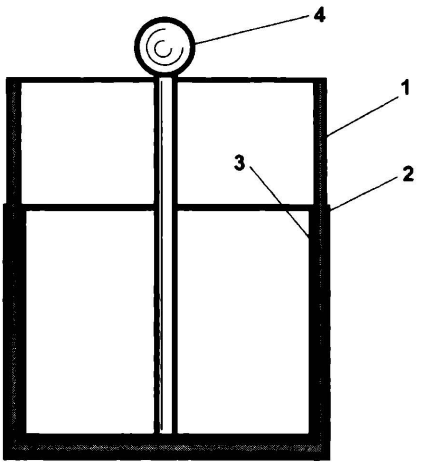


Рис. 5.4. Прадедушка современных конденсаторов — лейденская банка: 1 — стеклянный стакан; 2 — внешняя обкладка из станиоля; 3 — внутренняя обкладка, 4 — контакт для зарядки

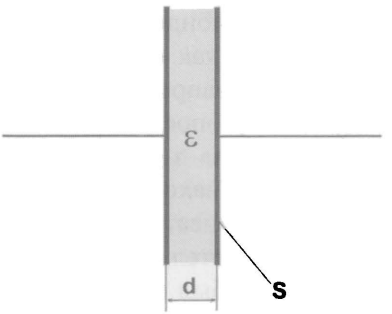


Рис. 5.5. Схематическое изображение плоского конденсатора

Схематическое изображение простейшего конденсатора показано на рис. 5.5. Его емкость рассчитывается по следующей формуле (она носит специальное название *формула плоского конденсатора*, потому что для конденсаторов иной геометрии соответствующее выражение будет другим):

$$C = \frac{\varepsilon S}{4\pi d},$$

где: C — емкость, Ф; S — площадь пластин, м^2; d — расстояние между пластинами, м; ε — диэлектрическая проницаемость.

Из этой формулы следует, что емкость тем больше, чем больше площадь пластин и чем меньше расстояние между ними. Что же такое емкость? Согласно определению, *емкость есть отношение заряда (в кулонах) к разности потенциалов на пластинах (в вольтах)*: $C = Q/U$, т. е. размерность емкости есть кулон/вольт. Такая

единица называется фарадой, по имени знаменитого английского физика и химика Майкла Фарадея (1791–1867).

Следует подчеркнуть, что величина емкости есть индивидуальная характеристика конденсатора, — подобно тому, как номинальное сопротивление есть индивидуальная характеристика конкретного резистора, — и характеризует количество энергии, которое может быть в нем запасено. Емкость в одну фараду весьма велика — обычно на практике используют микрофарады и еще более мелкие единицы, — так, емкость упомянутой лейденской банки составляла величину всего-навсего порядка 1 нФ.

Смысл понятия емкости раскрывается так: если напряжение от источника напряжения составляет 1 В, то емкость в одну нанофараду, как у лейденской банки, может запасти 10^{-9} кулон электричества. Если напряжение составит 10^5 вольт (типичная величина при заряде от электростатической машины, как в опытах Мушенбрука), то и запасенный на этой емкости заряд увеличится в той же степени — до 10^{-4} кулон. Любой конденсатор фиксированной емкости сохраняет это соотношение — заряд на нем в любой момент времени тем больше, чем больше напряжение, а сама величина заряда определяется номинальной емкостью.

Если замкнуть конденсатор на резистор, то в первый момент времени конденсатор будет работать как источник напряжения с нулевым выходным сопротивлением и номинальным напряжением той величины, до которой он был заряжен, т. е. ток через резистор определяется по обычному закону Ома. Скажем, в случае гвардейцев Мушенбрука характерное сопротивление цепи из нескольких человек, взявшихся за руки, находится в пределах 10^5–10^6 Ом — т. е. ток при начальном напряжении на конденсаторе 10^5 В составит от 0,1 до 1 А, что примерно в 100–1000 раз превышает смертельное для человека значение тока! Выручило гвардейцев то, что такой импульс был весьма кратковременным — по мере разряда конденсатора, т. е. стекания заряда с пластин, напряжение быстро снижается: емкость-то остается неизменной, потому при снижении заряда, согласно формуле плоского конденсатора, падает и напряжение.

Интересно, что при фиксированном заряде (если цепь нагрузки конденсатора отсутствует) можно изменить напряжение на нем, меняя емкость. Например, при раздвижении пластин плоского конденсатора емкость его падает (т. к. расстояние d между пластинами увеличивается), потому для сохранения заряда напряжение должно увеличиться — что и происходит на деле, когда в эффектном школьном опыте между раздвигаемыми пластинами конденсатора проскакивает искра при превышении предельно допустимого напряжения пробоя для воздуха.

На рис. 5.6 изображено подключение конденсатора C к нагрузке R. Первоначально переключатель K ставится в нижнее по схеме положение, и конденсатор заряжается до напряжения батареи Б. При переводе переключателя в верхнее положение конденсатор начинает разряжаться через сопротивление R, и напряжение на нем снижается. Насколько быстро происходит падение напряжения при подключении нагрузки? Можно предположить, что чем больше емкость конденсатора и сопротивление резистора нагрузки, тем медленнее происходит падение напряжения. Правда ли это?

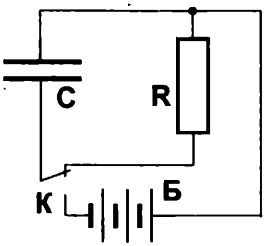


Рис. 5.6. Подключение конденсатора к нагрузке: К — переключатель; Б — батарея; С — конденсатор; R — сопротивление нагрузки

Это легко попробовать оценить через размерности связанных между собой электрических величин: тока, емкости и напряжения. В самом деле, в определение тока входит и время (напомним, что ток есть заряд, протекающий за единицу времени), и это время должно быть тем самым временем, которое нас интересует. Если вспомнить, что размерность емкости есть кулоны на вольт, то искомое время можно попробовать описать формулой: $t = CU/I$, где C — емкость, а U и I — ток и напряжение соответственно (проверьте размерность!). Для случая рис. 5.6 эта формула справедлива на малых отрезках времени, пока ток I не падает значительно из-за уменьшения напряжения на нагрузке. Отметим, что формула эта полностью справедлива и на больших отрезках времени, если ток разряда — или заряда — конденсатора стабилизировать, что означает подключение его к источнику втекающего (при разряде) или вытекающего (при заряде) тока.

При обычной фиксированной нагрузке с сопротивлением R так, конечно, не происходит — напряжение на конденсаторе падает по мере истощения заряда, значит, ток через нагрузку также пропорционально снижается — в полном соответствии с законом Ома (помните, мы говорили, что простой резистор есть плохой источник тока?). Опять приходится брать интегралы, потому мы приведем только конечный результат: формула для расчета процесса снижения напряжения на емкости при разряде ее через резистор и соответствующий график показаны на рис. 5.7, *а*. А на рис. 5.7, *б* показан аналогичный процесс, который происходит при заряде емкости через резистор.

Нужно отметить два момента: во-первых, процесс разряда по рис. 5.7, *а* бесконечен (полностью конденсатор не разрядится никогда, если сопротивление нагрузки не

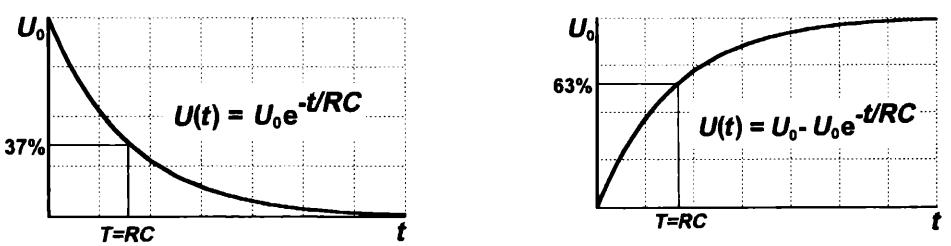


Рис. 5.7. Процессы при разряде (*а*) и заряде (*б*) конденсатора: C — емкость; R — сопротивление нагрузки; t — время; e — основание натуральных алгоритмов (2,718282)

равно нулю), но практически это не имеет значения, потому что напряжение на конденсаторе становится исчезающе малым очень скоро. Во-вторых, из формул на рис. 5.7 следует очень интересный вывод: если сопротивление R равно нулю, то время процесса разряда или заряда становится бесконечно малым, а ток через нагрузку — по закону Ома — бесконечно большим!

Обратимся снова к рис. 5.6 — именно нечто подобное должно происходить при переключении К в положение заряда емкости от батареи. Естественно, в реальной жизни ни о каких бесконечных токах речи не идет — для этого батарея должна иметь нулевое выходное сопротивление, т. е. бесконечно большую мощность (подумайте, почему эти утверждения равносильны?). Да и проводники должны обладать нулевым сопротивлением. Поэтому на практике процесс заряда от источника (и разряда при коротком замыкании пластин) происходит за малое, но конечное, время, а ток, хоть и не бесконечно велик, но все же может достигать очень больших значений. Потому-то источники питания с отключением по превышению максимально допустимого тока (см. *главу 2*) могут выключаться при работе на нагрузку с конденсатором большой емкости, установленным параллельно источнику питания (мы дальше увидим, что такой конденсатор устанавливают практически всегда), хотя ток в рабочем режиме может быть и невелик.

Один из методов борьбы с этой напастью — включение последовательно с нагрузкой небольшого резистора, ограничивающего ток в начальный момент времени. Как рассчитать необходимый номинал? Для этого нужно представить, что *конденсатор при заряде в первый момент времени ведет себя так, как будто цепь в месте его установки замкнута накоротко* (это очень точное представление!). Тогда нужный номинал резистора определится просто по закону Ома, в который подставляется предельно допустимый ток источника и его напряжение.

Интуитивно кажется, что должна существовать какая-то характеристика цепи из конденсатора и сопротивления, которая позволяла бы описать процесс зарядаразряда во времени, — независимо от напряжения на конденсаторе. Такая характеристика рассчитывается по формуле $T = RC$. Приведением единиц мы бы здесь занимались довольно долго, потому поверьте, что размерность произведения RC есть именно время в секундах. Эта величина, которая носит название *постоянной времени* RC-цепи, физически означает время, за которое напряжение на конденсаторе при разряде его через резистор (см. рис. 5.7, *а*) снижается на величину 0,63 от начального (т. е. до величины, равной доле $1/e$ от первоначального U_0, что и составляет примерно 37%). За следующий отрезок времени, равный RC, напряжение снизится еще на столько же от оставшегося и т. д. — в полном соответствии с законом экспоненты.

Аналогично при заряде конденсатора (см. рис. 5.7, *б*), постоянная времени T означает время, за которое напряжение увеличится до доли $(1 - 1/e)$ от конечного значения U_0, т. е. до 63% от U_0. Дальше мы увидим, что произведение RC играет огромную роль при расчетах различных схем.

Есть еще одна вещь, которая следует из формулы для плоского конденсатора. В самом деле, там нет никаких ограничений на величины S и d — даже если развести пластины очень далеко, все же какую-то емкость, хотя небольшую, конденсатор

будет иметь. То же происходит при уменьшении площади пластин. Практически это означает, что небольшую емкость между собой имеют любые два проводника, независимо от их конфигурации и размеров, хотя эти емкости и могут быть исчезающе малы. Этот факт имеет огромное значение на высоких частотах — в радиочастотной технике нередко конденсаторы образуют прямо из параллельных дорожек на печатной плате. А емкости между параллельными проводами в обычном проводе-«лапше» или кабеле могут достигать и весьма больших значений — ввиду их большой длины. В большинстве случаев этот эффект весьма вреден, и такие емкости называют *паразитными*.

Если же учесть, что проводники имеют еще и собственное сопротивление, то мы приходим к выводу, что любую пару проводов можно представить в виде «размазанной» по длине (распределенной) RC-цепи — и это действительно так, со всеми вытекающими последствиями! Например, если подать на вход пары проводников в длинном кабеле перепад напряжения (фронт прямоугольного импульса), то на выходе мы получим картину, которая ничем не отличается от рис. 5.7, *б* — импульс «размажется», а если он короткий, то вообще может пропасть.

Заметки на полях

Мало того, провода обладают еще и собственной индуктивностью (об индуктивности мы поговорим в конце главы), что еще более запутывает картину. Крайне неприятное явление, но «такова се ля ви», как любил выражаться один мой знакомый инженер. Впервые с этим делом столкнулись еще при попытке прокладки первого трансатлантического кабеля в 1857 году — телеграфные сигналы (точки-тире) представляют собой именно такие прямоугольные импульсы, и при длине кабеля в 4000 км они по дороге искажались до неузнаваемости. За время до следующей попытки прокладки кабеля (1865 год) английскому физику У. Томсону пришлось разработать теорию передачи сигналов по длинным линиям, за что он получил рыцарство от королевы Виктории и вошел в историю под именем лорда Кельвина — по названию городка Кельвин на западном побережье Ирландии, откуда начиналась прокладка кабеля[1].

В выражении для емкости плоского конденсатора фигурирует постоянная ε, представляющая собой диэлектрическую проницаемость среды. Для воздуха и большинства обычных изолирующих материалов (полиэтилена, хлорвинила, лавсана, фторопласта) константа ε близка к величине ее для полного вакуума ε_0. Величина ε_0 зависит от применяемой системы единиц измерения, и в международной системе единиц измерения СИ равна $8{,}854 \cdot 10^{-12}$ Ф/м. На практике удобно применять относительную диэлектрическую проницаемость конкретного материала: $\varepsilon_r = \varepsilon/\varepsilon_0$. Естественно, что в практических конструкциях конденсаторов желательно, чтобы величина ε_r была как можно выше, — если вы заполните промежуток между пластинами, скажем, ацетоном или спиртом, то емкость такого конденсатора сразу возрастет раз в двадцать! К сожалению, чем выше ε_r, тем обычно выше и собственная проводимость материала, потому такой конденсатор быстро разрядится за счет собствен-

[1] Так рассказывает Д. Л. Шарле в своей книге «По всему земному шару», оговаривая, что это «одна из версий». Вероятно, это действительно не более чем легенда — англоязычная «Википедия» утверждает, что знаменитая фамилия образована от реки Кельвин, протекающей неподалеку от университета в городе Глазго, где работал У. Томсон.

ных токов утечки через среду между пластинами. Ясно, что производители конденсаторов стараются упаковать как можно большую емкость в как можно меньшие размеры, пытаясь одновременно обеспечить токи утечки на приемлемом уровне. По этой причине количество практически используемых типов конденсаторов значительно больше, чем резисторов. Причем надо также учесть, что чем тоньше прослойка диэлектрика между пластинами, тем меньше предельно допустимое напряжение (т. е. напряжение, при котором наступает электрический пробой, и конденсатор выходит из строя).

Самым высоким соотношением емкость/габариты обладают электролитические (оксидные) конденсаторы, которые в настоящее время широко представлены серией, известной под отечественным наименованием К50-35 (импортные конденсаторы такого же типа обычно все равно продают под этим названием). Емкости их достигают 100 000 мкФ, а допустимые напряжения — 600 В, но у них есть три главных недостатка, которыми приходится платить за повышенную емкость. Первый и самый главный — эти конденсаторы полярны, т. е. подразумевают включение только в определенной ориентации по отношению к полярности источника питания. Обычно на корпусе таких конденсаторов обозначается либо отрицательный (жирным «минусом»), либо положительный (знаком «плюс») вывод. Если же габариты корпуса не позволяют применить обозначение (либо производителям лень налаживать соответствующую полиграфию), то полярность пытаются обозначить толщиной или длиной вывода — более длинный и/или более толстый вывод обычно обозначает положительный контакт (но не всегда!). Если же включить такой электролитический конденсатор в противоположной полярности, то он может просто взорваться, забрызгав электролитом всю остальную схему. Есть и другие, более дорогие типы полярных конденсаторов (например, танталовые К52 или ниобиевые К53), которые обладают значительно меньшими токами утечки. Электролитические конденсаторы обычно используют в качестве фильтров в источниках питания — хотя и иные применения не исключены.

Следует учесть и вторую дурную особенность «электролитов» (как их называют на инженерном жаргоне, и как мы будем называть их в дальнейшем) — они обеспечивают номинальную емкость только на низких частотах. При быстром перезаряде их емкость существенно снижается — поэтому в фильтрах источников питания рекомендуется параллельно ставить неполярные (керамические или иные) конденсаторы — в целях лучшей защиты от высокочастотных помех. На самом деле эти конденсаторы лучше ставить не в источнике питания, а непосредственно вблизи выводов компонента, для которого высокочастотные помехи критичны, и на практике так и поступают (такие конденсаторы называют *развязывающими*).

Эта особенность связана с третьим паразитным свойством электролитов — эффектом «аккумулятора» (или «накопления заряда»). То есть, если вы полностью разрядите электролитический конденсатор (например, коротким замыканием выводов), через некоторое время напряжение на выводах опять восстановится до некоторого значения (обычно небольшого — около 1–1,5 В), и чтобы этот заряд полностью рассосался, требуется довольно значительное время (часы или даже сутки). Эффект «аккумулятора» тем сильнее, чем больше емкость и чем выше допустимое напря-

жение электролита. Имеют электролиты и высокий заводской разброс номинального значения — до нескольких десятков процентов. По этим причинам полярные конденсаторы очень не рекомендуется употреблять во времязадающих цепях, если требуется хоть какая-то точность.

Для использования в других областях применяют конденсаторы с неполярным диэлектриком: бумажные, слюдяные, керамические, полиэтилентерефталатные (лавсановые) или фторопластовые (тефлоновые). Емкость их (в соотношении емкость/габариты) значительно меньше, и номинальная емкость обычно не превышает нескольких микрофарад (сравнительные размеры конденсаторов показаны на рис. 5.8).

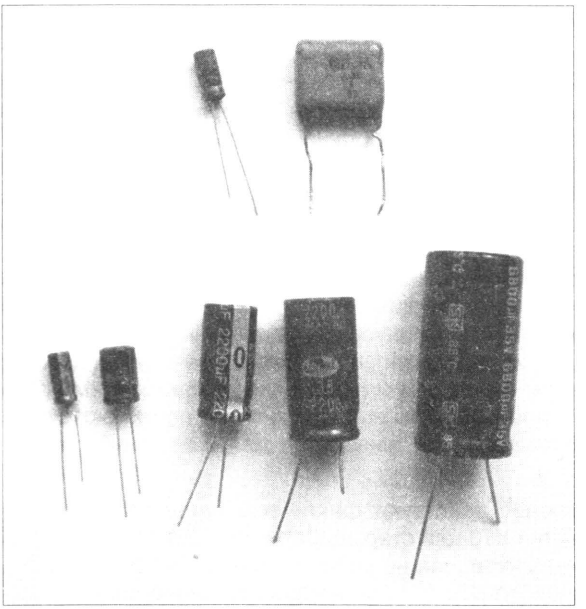


Рис. 5.8. Сравнительные размеры конденсаторов большой емкости:
вверху слева — электролитический конденсатор К50-35 3,3 мк × 25 В;
вверху справа — близкий к нему по допустимому напряжению неполярный конденсатор К73-17 3,3 мк с лавсановым диэлектриком; *внизу* — электролитические конденсаторы К50-35 (*справа налево*: 6800 мк × 35 В; 2200 мк × 35 В; 2200 мк × 16 В; далее два идентичных конденсатора 100 мк ×16 В, но производства разных фирм)

У старинных металлобумажных конденсаторов (типа МБГО или МБГЧ) есть интересная особенность — они могут самовосстанавливаться после пробоя. Судя по каталогам торгующих фирм, они в нашей стране выпускаются и популярны до сих пор. Но чаще всего сейчас употребляются неполярные конденсаторы с керамическим или органическим диэлектриком (типы К10, К73 и др.), и под *неполярными* мы будем обычно понимать конденсаторы именно этих серий. Именно они обеспечивают наиболее точное соответствие кривой заряда-разряда теоретической форме (как на рис. 5.7). Причем для применения в точных времязадающих цепях рекомендуется не просто выбирать конденсатор с подходящим изолятором, но, в отличие от электролитических, и с как можно большим допустимым напряжением, — в при-

менении керамического или лавсанового конденсатора с номинальным допустимым напряжением 630 В в цепях с напряжением 12 В нет ничего особенного.

Наиболее распространены неполярные керамические конденсаторы (отечественный аналог — К10), которые имеют оптимальное соотношение емкость/габариты и приемлемые характеристики по долговечности и стабильности. Они выпускаются как с гибкими выводами (обычно почему-то в корпусах желтого цвета), так и в SMD-исполнении. Емкости их могут варьировать в широком диапазоне от 1 пФ до 47 мкФ, а максимально допустимое напряжение, как правило, не менее 50 В. У керамических конденсаторов зависимость емкости от температуры практически линейна, потому значение температурного коэффициента емкости (ТКЕ) имеет наибольшее разнообразие из всех типов и всегда обозначается на корпусе.

ТКЕ конденсаторов

Температурный коэффициент емкости измеряется в миллионных долях на градус (ppm/°C) и обозначается по довольно простой системе: первой идет буква М или П, что обозначает знак температурной зависимости — минус или плюс (импортные, соответственно, N или P), после которого идет максимальное для данного типа значение ТКЕ в ppm/°C. Например, П33 обозначает положительный коэффициент в 33 миллионных доли на градус, а М75 — отрицательный в 75 долей на градус. В реальности эти цифры лишь очень приблизительно описывают возможные отклонения: цифра равна математическому ожиданию (см. *главу 13*) значения ТКЕ с большим разбросом, так что реальный ТКЕ для минусовой маркировки (буква М или N) может быть как отрицательным, так и положительным. Самые лучшие с точки зрения температурной стабильности типы с нулевым математическим ожиданием ТКЕ обозначаются как МП0 (импортное NP0 или C0G), что еще не означает полного отсутствия температурной зависимости: разброс (цифра приводится для импортных) может составить от –75 до +33 ppm/°C.

Все эти обозначения действуют фактически только для керамических конденсаторов (отечественный тип К10 или старые КМ), а также для некоторых конденсаторов с органическим диэлектриком, самые распространенные из которых — полиэтилентерефталатные (лавсановые) К73-17. Существует большое количество типов (например, все электролитические), для которых ТКЕ вообще никак не обозначается и не приводится в справочниках, а из точностны́х характеристик указывается лишь допустимый разброс в процентах.

Ненормируемые по ТКЕ конденсаторы (т. е. с существенным отклонением зависимости от линейной) обозначаются буквой Н с указанием величины процентов максимального изменения емкости во всем допустимом диапазоне температур, составляющем обычно от –30 или –55 до +85 °C. Например, Н90 означает, что в этом диапазоне отклонение емкости составит не более ±90% (впечатляет, да? но «такова се ля ви»). Для импортных конденсаторов с ненормируемым ТКЕ существует огромное количество обозначений (типа Y5V или X5F), выучивать которые совершенно не нужно, стоит запомнить только, что часто встречающееся обозначение X7R означает расширенный температурный диапазон от –55 до +125 °C.

Надо иметь в виду, что самые стабильные (в том числе по временно́му дрейфу) конденсаторы — слюдяные (старинные КСО или современные К31), их и было бы предпочтительно ставить во времязадающих цепях, если бы не стоимость, размеры и маленький диапазон емкостей, ограниченный сверху значениями в единицы нанофарад. Для отечественных слюдяных конденсаторов классы по ТКЕ обозначают

буквами от А до Г (самая лучшая — Г — соответствует разбросу ±50 ppm/°C). Из высокостабильных есть еще полистироловые (лучшие — К70-7 и К71-1 с допуском в единицы процентов) и поликарбонатные (К77). Последние встречаются крайне редко (они предназначены в основном для военной техники) и заменяются импортными полифенилсульфидными, доступными, однако, лишь в корпусах для поверхностного монтажа. Указанные типы зато встречаются для значений емкостей вплоть до единиц микрофарад. Если во времязадающих цепях все-таки необходимы большие значения емкостей, и приходится употреблять электролитические конденсаторы, то лучше выбирать ниобиевые (К53) или танталовые (К52), а не обычные алюминиевые.

В добавление к тому, что было сказано в *разд. «Резисторы»* про условные обозначения, нужно отметить, что, поскольку емкости обычно употребляемых конденсаторов находятся в пределах от пико- до микрофарад, то при обозначении на схемах единицу измерения Ф часто опускают и пишут просто «мк» (мкФ), «н» или «n» (нФ), «п» или «р» (пФ). Пикофарады (подобно омам) могут и не писать вообще. Часто микрофарады обозначаются просто лишним десятичным знаком (мы именно так и будем поступать) — например, запись 100,0 означает 100 мкФ, в то время как просто 100 — 100 пФ.

Заметки на полях

В разговоре о конденсаторах нельзя не упомянуть о получивших в последние годы распространение *ионисторах* — так в русском техническом языке называется специальный тип конденсаторов супербольшой емкости. На Западе термин «ионистор» неизвестен, и их называют по-разному: конденсаторы с двойным электрическим слоем (Double-Layer capacitors), суперконденсаторы (SuperCaps), резервные конденсаторы (Backup capacitors) и т. п. Ионисторы имеют недостижимую для других типов емкость, измеряющуюся единицами фарад (примерно такую емкость имеет проводящий шар размером с Землю). Поэтому их во многих случаях можно использовать в качестве резервного источника питания вместо аккумулятора. Энергетическая емкость ионисторов (в мА·ч или Вт·ч) оказывается примерно на три порядка меньше, чем у настоящих аккумуляторов, зато они гораздо долговечнее, лишены капризов электрохимических элементов при зарядке (см. *главу 9*) и могут непрерывно подзаряжаться через обычный диод. Следует, однако учесть, что ионистор емкостью в 1 фараду током в 1 ампер не будет разряжаться всего 1 секунду: из-за высокого внутреннего сопротивления (порядка десятков ом) это произойдет гораздо медленнее, а большая часть напряжения упадет на самом ионисторе и рассеется бесполезным теплом, потому для мощных устройств они малоприменимы. Но для маломощных схем (вроде микроконтроллеров или часов реального времени) это неплохая замена аккумуляторам.

Параллельное и последовательное включение конденсаторов

Как и резисторы, конденсаторы могут включаться последовательно или параллельно, однако расчет полученных величин производится противоположным образом: при параллельном соединении емкости складываются (по правилу «больше большего»), а при последовательном соединении складываются их обратные величины (правило «меньше меньшего»). К счастью, в отличие от резисторов, конденсаторы включают практически только параллельно — можно это представить так, как буд-

то при этом складываются площади их пластин, следовательно, складываются и емкости. Последовательное же соединение емкостей само по себе не имеет практического смысла, и знание правил сложения для них необходимо лишь изредка при анализе цепей переменного тока.

Конденсаторы в цепи переменного тока

Из этой большой темы мы здесь рассмотрим только самое необходимое. В дальнейшем мы будем иметь дело в основном с цепями постоянного тока или низкой частоты, и углубленное изучение поведения компонентов при высокой частоте нам не потребуется. В предыдущей фразе слова «низкой частоты» нужно понимать условно, и вот почему — любой перепад напряжения (например, при включении или выключении питания) есть импульс высокой частоты, и тем она выше, чем быстрее происходит сам процесс снижения или повышения напряжения. Если представить себе фронт импульса постоянного тока как сумму гармонических (т. е. синусоидальных) колебаний[1], то импульс этот предстанет перед нами как сумма колебаний, начиная сверху с той частоты, при которой происходило бы наблюдающееся нами на деле нарастание (или спад) напряжения импульса, если бы сигнал был чисто гармонический. То есть если импульс строго прямоугольный, то эта самая верхняя частота должна быть равна бесконечности, чего на деле, конечно, не бывает, поэтому реальные импульсы всегда не строго прямоугольны. Прохождение прямоугольных импульсов через конденсаторно-резисторную цепочку мы рассмотрим далее, а пока изучим поведение конденсаторов в цепях с обычным синусоидальным переменным током.

Постоянный ток конденсатор не пропускает по определению — поскольку представляет собой разрыв в цепи. Однако переменный ток через него протекает — при этом происходит постоянный перезаряд конденсатора из-за того, что напряжение все время изменяется по величине и полярности. Поэтому конденсатор в цепи переменного тока можно представить себе как некое сопротивление — чем меньше емкость конденсатора и чем ниже частота, тем выше величина этого условного сопротивления. Ее можно подсчитать по формуле $R = 1/2\pi fC$ (если емкость C выражена в фарадах, а частота f в герцах, то сопротивление получится в омах). В пределе конденсаторы очень малой емкости (которые представляют собой, как мы выяснили, почти все пары проводников на свете) будут выглядеть в цепи полными разрывами, и ток в этой цепи окажется исчезающе мал.

Сам по себе конденсатор в такой цепи энергии не потребляет (в отличие от обычного резистора), потому его сопротивление переменному току называют еще *реактивным* — в то время, как обычное резистивное сопротивление называют *активным* (не путать с активными и пассивными компонентами схем, о которых шла речь в начале главы). Понять, почему так происходит, можно, если нарисовать гра-

[1] Любое колебание можно представить в виде такой суммы согласно теореме Фурье, великого французского математика, работавшего еще в конце XVIII века. Возможен и обратный процесс — восстановление изначальной формы колебания через известную сумму гармоник.

фики тока и напряжения в цепи с конденсатором, — ток опережает напряжение по фазе ровно на 90°, поэтому их произведение, которое и есть потребляемая мощность по закону Джоуля — Ленца, в среднем равно нулю — можете проверить! Однако если в цепи присутствуют еще и обычные резисторы (а, как мы знаем, они всегда присутствуют — взять хотя бы сопротивление проводов), то этот реактивный ток приведет ко вполне материальным потерям на их нагревание, — именно поэтому, как мы упоминали в *главе 4*, линии электропередач выгоднее делать на постоянном токе.

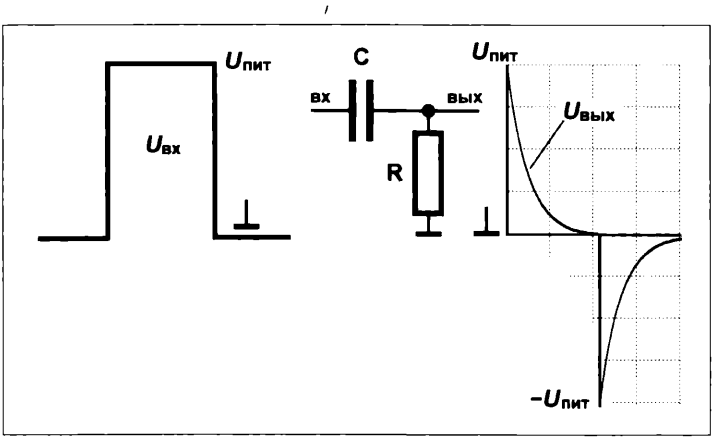
Дифференцирующие и интегрирующие цепи

Если подать на вход цепи, состоящей из резистора R и конденсатора C, прямоугольный импульс напряжения, то результат будет различным в зависимости от включения R и C. Переходные процессы в таких цепях подчиняются основным закономерностям, представленным на рис. 5.7, но имеют и свою специфику. На рис. 5.9 показаны два способа включения RC-цепочки в схему с прямоугольными импульсами на входе (здесь они не такие, как на рис. 4.5, *б*, а однополярные, т. е. напряжение меняется по величине от потенциала «земли» до напряжения источника питания).

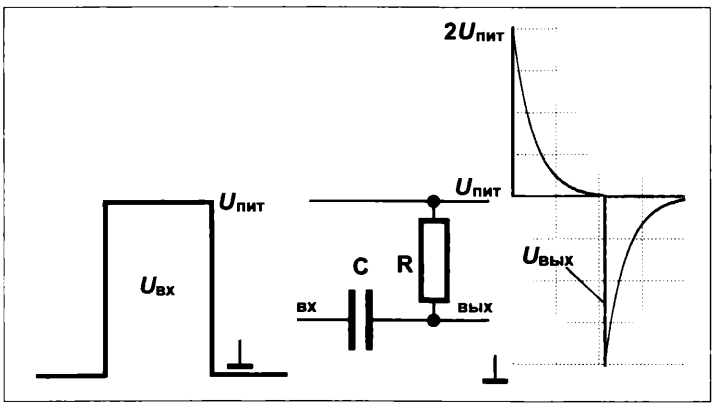
Такое включение называется *дифференцирующей цепочкой* или *фильтром высоких частот* — потому что оно пропускает высокочастотные составляющие, полностью отрезая постоянный ток. Чем больше постоянная времени *RC* в этой схеме, тем ниже частота, которая может быть пропущена без изменений, — в пределе импульсы высокой частоты пройдут почти неизмененными. Наоборот, если постоянную времени уменьшать, то пики на графике будут все больше утончаться. Этим эффектом широко пользуются для выделения фронтов и спадов прямоугольных импульсов (см. *главу 16*).

Так как через эти схемы постоянная составляющая напряжения не проходит, то полученные импульсы привязаны к выходному потенциалу схемы — в зависимости от того, куда подключен резистор. На графиках (см. рис. 5.9) резистор подключен к «земле» (*а*) или к источнику питания (*б*), потому и для выходного напряжения базовым будет либо нулевой потенциал, либо потенциал источника (при этом амплитуда импульсов будет такой, как у входного напряжения). Но вы можете подключать резистор на выходе такой схемы к любому потенциалу — она все равно передаст только переменную составляющую (с чем мы еще столкнемся при конструировании звукового усилителя).

Этим широко пользуются при необходимости формирования двухполярного напряжения из имеющегося однополярного или для умножения напряжения: если выходное напряжение на рис. 5.9, *б* пропустить через выпрямитель и сглаживающий фильтр низкой частоты (см. далее, а также *главу 9*), то на выходе получится напряжение выше, чем напряжение питания, причем в отсутствие нагрузки оно будет в точности вдвое превышать исходное напряжение («удвоитель напряжения»). Иногда эффект удвоения вреден — подачей отрицательного или превышающего



а



б

Рис. 5.9. Дифференцирующие цепочки: *а* — при подключении резистора к нулевому потенциалу; *б* — к потенциалу источника питания

потенциал источника питания напряжения можно вывести из строя компоненты схемы (о защите от этого см. *главы 11* и *16*).

А *интегрирующая цепочка* (фильтр нижних частот) получается из схем рис. 5.9, если в них R и C поменять местами. График выходного напряжения при этом будет соответствовать показанному на рис. 5.10. Такие цепочки, наоборот, пропускают постоянную составляющую, в то время как высокие частоты станут отрезаться. Если в такой цепочке увеличивать постоянную времени *RC*, то график будет становиться все более плоским — в пределе пройдет только постоянная составляющая (которая для случая рис. 5.10 равна среднеамплитудному значению исходного напряжения, т. е. ровно половине его амплитуды). Этим широко пользуются при конструировании вторичных источников питания, в которых нужно отфильтровать переменную составляющую сетевого напряжения (см. *главу 9*). Интегрирующими свойствами обладает и обычный кабель из пары проводов, о котором мы упоминали ранее, потому-то и теряются высокие частоты при прохождении сигнала через него.

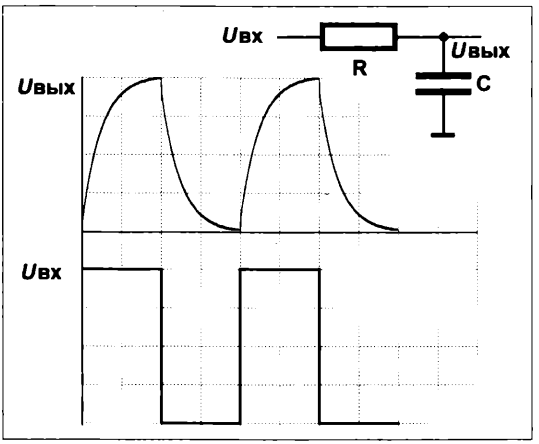


Рис. 5.10. Интегрирующая цепочка и ее график выходного напряжения в одном масштабе с входным

Индуктивности

Таким же свойством реактивного сопротивления в цепи переменного тока обладают *индуктивности* — хотя они по всему противоположны конденсаторам. Мы не будем здесь рассматривать индуктивности подробно по простой причине — в обычной схемотехнике (кроме радиочастотной, а в настоящее время уже и там) индуктивностей в основном стараются избегать, и используют их лишь в трансформаторах и еще разве что в фильтрах для защиты от помех. Но вкратце все же рассмотрим их свойства.

Простейшая индуктивность — катушка из провода, а если ее намотать на основу из ферромагнитного материала, то ее индуктивные свойства значительно улучшатся. Индуктивности очень сложно делать автоматизированным способом, кроме самых простых (не говоря уж об их включении в состав микросхем), и это одна из причин того, почему их стараются не использовать в массовой аппаратуре.

Измеряют индуктивность в генри (Гн), по имени выдающегося американского физика Джозефа Генри (1797–1878). Стандартные индуктивности со значениями порядка микро- и миллигенри выпускаются промышленно, внешне они похожи на резисторы и точно так же маркируются цветным кодом. Обычно они покрашены в светло-зелено-голубой цвет — чтобы отличить их от резисторов.

Если конденсатор для постоянного тока представляет собой разрыв цепи, то индуктивность, наоборот, — нулевое сопротивление. С ростом частоты переменного тока реактивное сопротивление индуктивности растет (у конденсатора, напомним, падает). Реактивное сопротивление индуктивности величиной L (Гн) можно вычислить по формуле: $R_L = 2\pi f L$.

Мы уже знаем, что любой перепад напряжения есть импульс высокой частоты, и попытка разорвать (или наоборот, соединить) цепь, содержащую индуктивность,

приводит к неожиданным последствиям. Из курса физики известно, что после разрыва цепи за счет самоиндукции ток продолжает некоторое время течь в витках катушки, а так как сопротивление цепи становится бесконечно велико, и течь ему некуда, то на индуктивности возникает большой (тем больший, чем больше величина индуктивности и чем меньше ее активное сопротивление, т. е. чем она ближе к идеалу) выброс напряжения — в полном соответствии с законом Ома. Этот эффект, например, приводит к выбросам напряжения на фронтах прямоугольных импульсов в схемах с использованием быстродействующих компонентов.

Подробности

Гораздо хуже, что эффект самоиндукции приводит к недопустимо большим выбросам напряжения при коммутации в цепях с индуктивными нагрузками (явление так и называется *коммутационным перенапряжением*). Разрыв цепи с индуктивностью не симметричен аналогичному процессу в емкостных цепях, где выбросы ни при каких условиях не могут превысить амплитуды напряжения питания (см. рис. 5.9). В отличие от этого случая, выбросы напряжения в цепях с индуктивностями определяются тем, что в индуктивности мгновенно не может измениться ток, а в емкостных цепях — напряжение. Потому выбросы, во-первых, возникают только при разрыве цепи (при подключении тока еще нет, и перенапряжений не возникает), во-вторых, их амплитуда может многократно превышать напряжение питания. В общем идеализированном случае, когда активное сопротивление индуктивной обмотки мало, амплитуда выброса $U_в$ достигает значения $I \cdot \sqrt{(L/C)}$, где I — значение тока в момент разрыва, А, L — индуктивность цепи, Гн, C — паразитная емкость обмотки и окружающих проводников.

Из анализа этой формулы следует, что при стремлении паразитной емкости к нулю, амплитуда выброса стремится к бесконечности. В реальности этого не бывает — в цепях, где активное сопротивление действительно мало в сравнении с индуктивным (обмотки мощных трансформаторов или двигателей), процесс носит характер затухающего колебания с максимумом амплитуды в начале, характерная величина которого примерно в 4–5 раз превышает значение напряжения питания. Если коммутация производится механическими контактами (и особенно, если это электрическая сеть с напряжением 220 или 380 вольт), то такое перенапряжение вызывает искрение на контактах вследствие превышения напряжения пробоя воздуха и возникновения электрической дуги. Если же коммутация, как в наших устройствах, производится полупроводниковым прибором (транзистором), то даже в цепи постоянного тока с напряжениями не более пары десятков вольт такое перенапряжение может привести к его выходу из строя.

В реальности этого нужно опасаться лишь при условии, что индуктивное сопротивление действительно велико в сравнении с активным сопротивлением обмотки и подводящих проводов. Для маломощных низковольтных электромагнитных реле и двигателей выброс эффективно гасится на этом сопротивлении, и за сохранность коммутирующих транзисторов можно не опасаться. Тем не менее, поскольку теоретический анализ всех этих явлений в каждом реальном случае провести невозможно, проще всего в схемах всегда предусматривать диод, включенный параллельно индуктивной нагрузке в обратном направлении, — он будет эффективно гасить любые перенапряжения. В MOSFET-транзисторах, предназначенных для использования в качестве ключей, такой диод уже имеется в составе их структуры (см. рис. 6.10, *б*), и специально об этом заботиться не приходится.

Надо знать, что индуктивные выбросы напряжения, даже погашенные защитными диодами, могут нарушить нормальную работу микроконтроллеров и других многофункциональных микросхем — например, счетчиков или регистров. Потому управлять индуктивными нагрузками непосредственно от их выходов нежелательно,

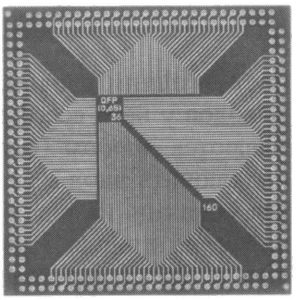
даже если характеристики выводов формально это позволяют. В этих случаях также бывает необходимым развязать питание индуктивной нагрузки и контроллера. Это касается не только электромагнитных реле, но и, например, двигателей для моделей, — широко растиражированная в Интернете схема управления сервоприводом непосредственно от Arduino (см. например, **http://edurobots.ru/2014/04/arduino-servoprivod/**) на практике неработоспособна. Причем в данном случае можно оставить цепи управления непосредственно от выводов (они у сервопривода маломощные), а вот изолировать цепи питания сервопривода и контроллера, установив отдельный стабилизатор, оказывается совершенно необходимым. Мы еще вспомним об этом явлении, когда будем говорить о реле в *главе 7*.

Ток в цепи, содержащей индуктивность, отстает от напряжения на 90° (для конденсатора ток, наоборот, опережает напряжение), но результат оказывается аналогичным — чистая индуктивность, включенная последовательно с нагрузкой, не потребляет энергии в цепи переменного тока, хотя ток в цепи будет зависеть от величины индуктивности. Только эффект этот проявляется обратно случаю с конденсатором — ток в цепи с индуктивностью падает с увеличением частоты (у конденсатора, как мы видели, он увеличивается), а для постоянного тока индуктивность представляет собой нулевое сопротивление. Для того чтобы получить эффект, близкий к расчетному, активное сопротивление индуктивности (т. е. ее сопротивление постоянному току) должно быть как можно ближе к нулю, чего на практике достичь довольно сложно.

Это другая причина того, что индуктивности очень не любят схемотехники, — их характеристики гораздо дальше от идеала, чем у резисторов и конденсаторов. Но надо помнить, что любой проводник всегда наделен этими тремя свойствами: т. е. в небольшой степени является и резистором, и конденсатором, и индуктивностью. Эти мелочи могут иногда сыграть довольно неожиданную роль в разных схемах.

Подробности

В силу указанных причин при наличии реактивной нагрузки в цепи переменного тока полезная мощность (в нагрузке) может отличаться от величины произведения потребляемого тока на напряжение — она всегда меньше. Поэтому в электротехнике различают реактивную мощность, выраженную в вольт-амперах, и активную мощность в ваттах, а отношение их называют *коэффициентом мощности*. Другое его общепринятое название — «косинус фи», потому что коэффициент мощности есть не что иное, как cos(φ), где φ — угол фазового сдвига тока относительно напряжения. При постоянном токе, а также в случае чисто активной нагрузки, этот угол равен нулю, потому коэффициент мощности равен 1. В другом предельном случае — когда нагрузка чисто реактивная — коэффициент мощности равен 0. В реальных цепях с электродвигателями или, скажем, с мощными вторичными импульсными источниками питания в качестве потребителей (офис с большим количеством компьютеров), коэффициент мощности может лежать в пределах 0,6–0,95. Следует подчеркнуть, что коэффициент мощности — это не КПД, как можно себе вообразить. Разница между вольт-амперами и ваттами никуда не теряется в физическом смысле, она всего лишь приводит к таким неприятным последствиям, как увеличение потерь в проводах, о котором мы упоминали (потери пропорциональны именно вольт-амперам), а также к возникновению разбаланса между фазами трехфазной промышленной сети, в результате чего через нейтраль (т. е. нулевой, обычно более тонкий, чем все остальные, провод) начинают протекать значительные токи.



ГЛАВА 6

Изобретение, которое потрясло мир

Диоды, транзисторы и простейшие схемы на их основе

> Не имея намерения подробно описывать осаду и приводя лишь те события, которые имеют непосредственную связь с рассказываемой нами историей, скажем вкратце, что предприятие удалось.
>
> *А. Дюма. «Три мушкетера»*

Казалось бы, все так просто: есть очень хорошие проводники (металлы), есть очень плохие — изоляторы (фарфор или пластмасса), а есть — полупроводники. Подумаешь! Причем полупроводников на свете гораздо больше, чем проводников и изоляторов, что, если подумать, представляется естественным. Однако когда научились получать очень чистые полупроводники со строго дозированными определенными примесями, то оказалось, что это — революция, потому что такие материалы обладают совершенно необыкновенными электрическими характеристиками.

Электроника на основе полупроводников носит также название *твердотельной*. Полупроводниковые приборы делают с использованием разных материалов — в основном, кремния. Первым промышленным полупроводником стал германий (а до него еще существовали купроксные и селеновые диоды), но сейчас он практически не употребляется, и только небольшая часть полупроводниковых компонентов — например, светодиоды — делается из материалов, отличных от кремния.

Из всех полупроводниковых устройств исторически первыми были диоды.

Диоды

Вообще-то *диод* — устройство вовсе не обязательно полупроводниковое. Были и ламповые диоды (кенотроны), но они давно вымерли, потому мы будем рассматривать только твердотельные.

Диод — простейший активный электронный прибор, проще не бывает. В одну сторону диод проводит ток (т. е. представляет собой в идеале проводник с малым сопротивлением), в другую — нет (т. е. превращается в очень большое сопротивле-

ние) — одним словом, обладает односторонней проводимостью. Соответственно, выводов у него всего два: они, как повелось еще со времен ламповой техники, называются *анодом* (положительным выводом) и *катодом* (отрицательным). Если подключить диод к регулируемому источнику напряжения, то он будет вести себя, как показано на рис. 6.1, где представлена так называемая *вольт-амперная характеристика* диода. Из нее, в частности, следует, что в прямом включении (т. е. анодом к плюсу источника) после превышения некоторого порогового напряжения ($U_{пр}$) прямой ток через диод ($I_{пр}$) растет неограниченно и будет лимитироваться только мощностью источника (скорее всего, диод сгорит раньше, чем эта мощность будет достигнута).

В обратном же включении (катодом к плюсу) ток через диод ($I_{обр}$) пренебрежимо мал и составляет несколько микро- или даже наноампер для обычных маломощных кремниевых диодов или до единиц миллиампер для мощных выпрямительных. Причем для германиевых диодов обратный ток намного выше, чем для кремниевых, отчего их сейчас практически и не употребляют. Этот ток сильно зависит от температуры и может возрасти на несколько порядков (от нано- до микроампер) при повышении температуры от −50 до +50 °С, поэтому на графике его величина показана очень приблизительно (обратите внимание, что для наглядности верхняя и нижняя половины графика по оси токов построены в разных масштабах).

В отличие от обратного тока, прямое падение напряжения $U_{пр}$ гораздо меньше зависит как от типа и конструкции, так и от температуры. *Для кремниевых диодов прямое падение напряжения $U_д$ всегда можно считать равным примерно 0,6–0,7 В*, для германиевых и так называемых *диодов Шоттки* (маломощных диодов с переходом металл-полупроводник) — порядка 0,2–0,4 В. Для кремниевых диодов при постоянном токе в цепи с увеличением температуры $U_{пр}$ падает примерно на 2,3 мВ на один градус, и этот эффект нередко используют для измерения температуры. В германиевых диодах, кстати, этот эффект в разы больше (порядка 10 мВ на градус).

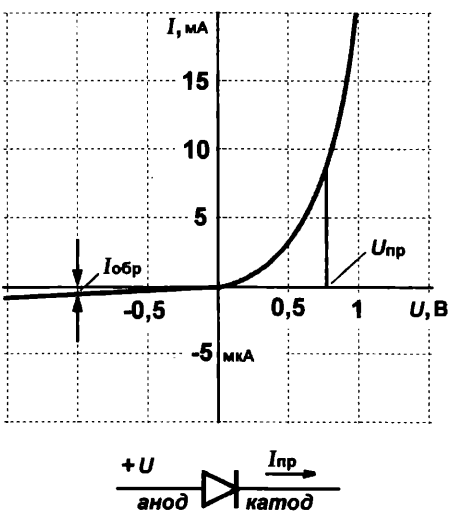


Рис. 6.1. Вольт-амперная характеристика диода

Если умножить указанное прямое падение напряжения на проходящий через диод в прямом включении ток, то мы получим *тепловую мощность*, которая выделяется на диоде. Именно она приводит диоды к выходу из строя — при превышении допустимого тока они просто сгорают. Обычное предельно допустимое среднее значение тока через маломощные диоды — десятки и сотни миллиампер. Впрочем, тепловые процессы инерционны, и мгновенное значение тока, в зависимости от длительности импульса, может превышать предельно допустимое среднее значение в сотни раз! Мощные диоды (рассчитанные на токи 3–5 А и выше) часто приходится устанавливать на радиаторы.

Другая характеристика диодов — *предельно допустимое обратное напряжение*. Если оно превышено, то диоды также выходят из строя — электрически пробиваются и замыкаются накоротко. Обычная допустимая величина обратного напряжения для маломощных диодов — десятки вольт, для выпрямительных диодов — сотни вольт, но есть диоды, которые выдерживают и десятки тысяч вольт (обычно они составляются из нескольких последовательно включенных диодов с меньшим допустимым значением — это так называемые *выпрямительные столбы*). Интересно, что кремниевые диоды при кратковременном превышении максимального обратного напряжения пробиваются обратимо — если ток невелик, и допустимая мощность не превышена, то после спада напряжения диод восстанавливает свои свойства. Далее мы увидим, что существуют и диоды, для которых пробой в обратном включении является рабочим режимом, — они называются *стабилитронами*.

Физически диод состоит из небольшого кристаллика полупроводникового материала, в котором в процессе производства формируются две зоны с разными проводимостями, называемыми проводимостью *n*- и *p*-типа. Ток всегда течет от направления *p*-зоны по направлению к *n*-зоне, в обратном направлении диод заперт. Более подробные сведения о физике процессов, происходящих в *p-n*-переходе, излагаются во множестве пособий, включая школьные учебники, но для практической деятельности почти не требуются.

Транзисторы

Транзистор — это электронный полупроводниковый прибор, предназначенный для усиления сигналов. Первым таким прибором в истории была электронная лампа (а еще до нее — электромагнитные реле, которые с некоторыми оговорками тоже можно отнести к усилителям тока или напряжения, — их мы рассмотрим в *главе 7*). Лампа сумела сделать немало — именно в ламповую эпоху возникли радио и телевидение, компьютеры и домашняя звукозапись. Но только транзистор и возникшие на его основе микросхемы сумели действительно перевернуть мир, ввести электронные устройства в наш повседневный быт так, что мы теперь уже и не мыслим себя без них.

Транзисторы делятся на *биполярные* и *полевые* (или униполярные). Пока мы будем говорить только о биполярных транзисторах.

Физически биполярный транзистор сложен из трех слоев полупроводника, разделенных двумя *p-n*-переходами. Поэтому можно себе представить, что он состоит

как бы из двух диодов, один из слоев у которых общий, и это представление весьма близко к действительности! Скомбинировать два диода можно, сложив их либо анодами, либо катодами, соответственно, получаются *n-p-n-* и *p-n-p*-транзисторы. Различаются эти разновидности только полярностями соответствующих напряжений, поэтому, чтобы заменить *n-p-n*-транзистор на аналогичный *p-n-p*, надо просто поменять знаки напряжений во всей схеме на противоположные (все полярные компоненты: диоды и электролиты, естественно, тоже надо перевернуть). *N-p-n*-типов транзисторов выпускается гораздо больше, и употребляются они чаще, поэтому мы пока что будем вести речь исключительно о них, но помнить, что все сказанное справедливо и для *p-n-p* с учетом обратной их полярности. Правильные полярности и направления токов для *n-p-n*-транзистора показаны на рис. 6.2.

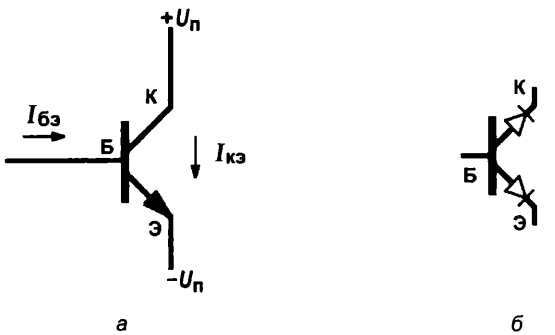


Рис. 6.2. *N-p-n*-транзистор: *а* — рабочие полярности напряжений и направления токов в *n-p-n*-транзисторе (К — коллектор, Б — база, Э — эмиттер); *б* — условное представление транзистора как элемента, состоящего из двух диодов

Заметки на полях

Первый в истории транзистор (рис. 6.3) был создан в 1947 году в знаменитых Лабораториях Белла (Bell Labs) Дж. Бардиным и У. Браттейном по идеям Уильяма Брэдфорда Шокли. Кроме изобретения транзистора, У. Шокли известен, как основатель знаменитой Кремниевой долины — технополиса в Калифорнии, где сегодня расположено большинство инновационных полупроводниковых и компьютерных фирм. Из фирмы Шокли под названием Shockley Semiconductor Labs вышли, в частности, Гордон Мур и Роберт Нойс — будущие основатели фирмы Intel, крупнейшего ныне производителя микропроцессоров. Г. Мур еще известен как автор знаменитого «закона Мура», а Р. Нойс — как изобретатель микросхемы (совместно с Д. Килби — подробнее об этом см. *главу 11*).

Три вывода биполярного транзистора носят названия *коллектор*, *эмиттер* и *база*. Как ясно из рис. 6.2, *б*, база присоединена к среднему из трех полупроводниковых слоев. Так как, согласно показанной на рисунке полярности включения, потенциал базы *n-p-n*-транзистора более положителен, чем у эмиттера, то соответствующий диод всегда открыт для протекания тока. Парой страниц ранее мы убедились, что в этом случае на нем должно создаваться падение напряжения в 0,6 В. Именно так и есть: *в рабочем режиме напряжение между эмиттером и базой кремниевого транзистора всегда составляет 0,6 В*, причем на базе напряжение выше, чем на эмиттере (еще раз напомним, что для *p-n-p*-транзисторов напряжения обратные,

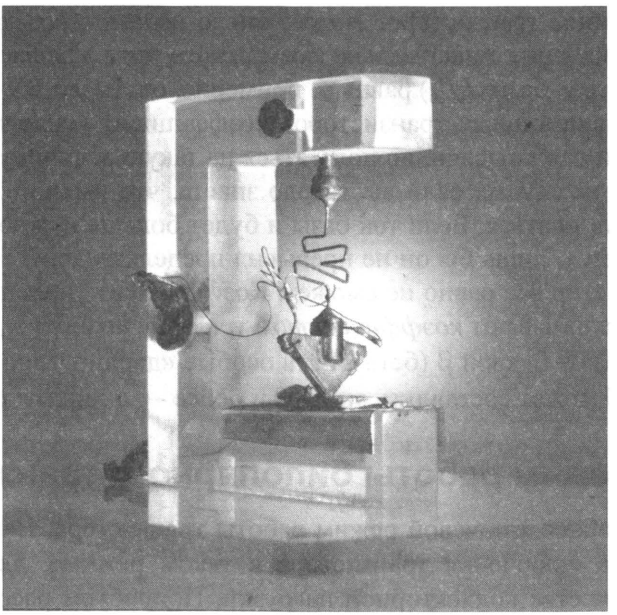


Рис. 6.3. Первый транзистор (Фото Lucent Technologies Inc./Bell Labs)

хотя их абсолютные величины совпадают). А вот диод между коллектором и базой заперт обратным напряжением. Как же может работать такая структура?

Практически это можно себе представить, как если бы ток, втекающий в базу, управлял бы неким условным резистором, расположенным между коллектором и эмиттером (пусть вас не смущает помещенный там диод коллектор-база, через него-то ток все равно не потечет). Если тока базы нет, т. е. выводы базы и эмиттера закорочены (необязательно, чтобы непосредственно, главное, чтобы напряжение на базе относительно эмиттера было бы близко к нулю), тогда промежуток эмиттер-коллектор представляет собой очень высокое сопротивление, и ток через коллектор пренебрежимо мал (сравним с током обратной утечки диода). В таком состоянии транзистор находится в *режиме отсечки* (транзистор заперт — закрыт).

В противоположном режиме ток базы велик ($U_{бэ}$ = 0,6–0,7 В, как мы говорили ранее), а промежуток эмиттер-коллектор тогда представляет собой очень малое сопротивление. Это *режим насыщения*, когда транзистор полностью открыт. Естественно, в коллекторной цепи должна присутствовать какая-то нагрузка, иначе транзистор в этом режиме может просто сгореть. Остаточное напряжение на коллекторе транзистора может при этом составлять всего около 0,3 В. Эти два режима представляют часто встречающийся случай, когда транзистор используется в качестве «ключа» (говорят, что он *работает в ключевом режиме*), т. е. просто как обычный выключатель тока.

А в чем смысл такого режима, спросите вы? Такой режим вполне осмыслен — ток базы может управлять током коллектора, который на порядок-другой больше — т. е. налицо усиление сигнала по току (за счет, естественно, энергии источника питания). Насколько велико может быть такое усиление? Для режима «ключа» почти

у всех обычных типов транзисторов можно смело полагать коэффициент усиления по току (т. е. отношение максимально возможного тока коллектора к минимально возможному току базы $I_к/I_б$) равным примерно от 10 до 30 — не ошибетесь. У современных маломощных транзисторов коэффициент усиления по току может составлять сотни и даже тысячи, но полагаться на такую величину при расчете схем можно только в том случае, если вы твердо знаете, что именно такой тип транзистора будет использоваться. Если ток базы и будет больше нужного — не страшно, он никуда не денется, лишь бы он не превысил предельно допустимого, а открыться сильнее транзистор все равно не сможет. Коэффициент усиления по току в ключевом режиме еще называют *коэффициентом усиления по току в режиме большого сигнала* и обозначают буквой β (бета). Есть особые «дарлингтоновские» транзисторы, для которых β может составлять до 1000 и более — о них мы расскажем позже.

Ключевой режим работы биполярного транзистора

Рассмотрим подробнее ключевой режим работы транзистора. На рис. 6.4 показана простейшая схема включения транзистора в таком режиме, для наглядности — с лампочкой в качестве коллекторной нагрузки. Попробуем рассчитать необходимую величину резистора в базе.

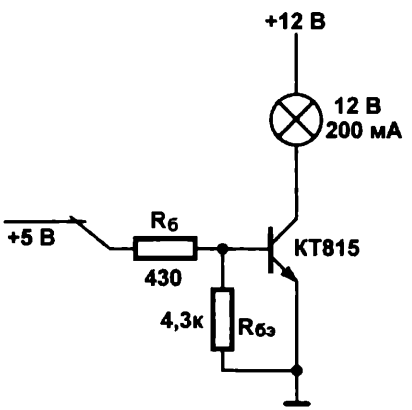


Рис. 6.4. Включение биполярного транзистора в ключевом режиме

Для почти любых схем с биполярными транзисторами характерно, что напряжения в схеме никакой роли не играют, только токи. Можно подключить коллекторную нагрузку хоть к напряжению 200 В, а базовый резистор питать от 5-вольтового источника — если соотношение $\beta > I_к/I_б$ соблюдается, то транзистор (при условии, конечно, что он рассчитан на такое высокое напряжение) будет послушно переключать 200-вольтовую нагрузку, управляясь от источника 5 В. То есть налицо и усиление сигнала по напряжению!

В нашем примере используется небольшая автомобильная лампочка 12 В, 100 мА (примерно, как для подсветки приборной доски в «Ладе»), а цепь базы питается от источника 5 В (например, через контакты реле). Расчет такой схемы элементарно

прост: при токе в коллекторе 100 мА в базе должно быть минимум 10 мА (рассчитываем на самый «дубовый» советский транзистор ранних типов, реально можно и меньше). О падении между базой и эмиттером забывать не следует, поэтому считаем, что напряжение на базовом резисторе $R_б$ составит 5 В – 0,6 В = 4,4 В, т. е. нужное сопротивление будет 440 Ом. Выбираем ближайшее из стандартного ряда и получаем 430 Ом. Все?

Нет, не все. Схема еще не совсем доделана. Она будет работать нормально, если вы поступите так: подключите базовый резистор к напряжению 5 В (лампочка горит), а затем *переключите его к «земле»* (лампочка гаснет). Но довольно часто встречается случай, когда напряжение на базовый резистор подается-то нормально, а вот при его отключении резистор не присоединяется к «земле», а просто «повисает в воздухе» (именно этот случай и показан на схеме в виде контактов). Так мы не договаривались — чтобы транзистор был в режиме отсечки, надо, чтобы база и эмиттер имели один и тот же потенциал, а какой потенциал у базы, если она «в воздухе»? Это только формально, что ноль, а на самом деле всякие наводки и внутренние процессы в транзисторе формируют небольшой базовый ток. И транзистор не закроется полностью — лампочка будет слабо светиться! Это раздражающий и очень неприятный эффект, который даже может привести к выходу транзистора из строя (а старые германиевые транзисторы приводил с гарантией).

Избежать такого эффекта просто — надо замкнуть базу и эмиттер еще одним резистором $R_{бэ}$. Самое интересное, что рассчитывать его практически не надо, — лишь бы падение напряжения на нем при подаче напряжения на базу не составило меньше, чем 0,6 В. Чем оно больше, тем лучше, но все же сопротивление не должно быть слишком велико. Обычно его выбирают примерно в 10 раз больше, чем резистор $R_б$, но если вы здесь поставите не 4,3 кОм, как указано на схеме, а, к примеру, 10 кОм, тоже не ошибетесь. Работать этот резистор будет так: если включающее напряжение на $R_б$ подано, то он не оказывает никакого влияния на работу схемы, поскольку напряжение между базой и эмиттером все равно 0,6 В, и он только отбирает на себя очень небольшую часть базового тока (легко подсчитать какую — примерно 0,15 мА из 10 мА). А если напряжения нет, то $R_{бэ}$ надежно обеспечивает равенство потенциалов базы и эмиттера, независимо от того, подключен ли базовый резистор к «земле» или «висит в воздухе».

Я так подробно остановился на этом моменте потому, что о необходимости наличия резистора $R_{бэ}$ при работе в ключевом режиме часто забывают, — даже в очень интересной во всех отношениях книге [3] повсеместно встречается эта ошибка.

Простейшая ключевая схема представляет собой вариант так называемой схемы включения транзистора *с общим эмиттером* (о. э.). В наших примерах есть два момента, на которые стоит обратить внимание. Во-первых, это подключение базовой цепи к питанию от 5 В. Это очень часто встречающийся случай, с которым и в этой книге вам придется иметь дело. Напряжением 5 В обычно питаются распространенные типы контроллеров и логических микросхем, и управление таким напряжением устройствами, которым требуется более высокое напряжение питания, чаще всего осуществляется именно по схеме, приведенной на рис. 6.4.

Во-вторых, обратите внимание, что сигнал на коллекторе транзистора *инвертирован* (т. е. противоположен по фазе) по отношению к входному сигналу. Так, если на базе (точнее, на базовом резисторе) напряжение имеется — то на коллекторе оно равно нулю, и наоборот! Это и имеют в виду, когда говорят, что *транзисторный каскад в схеме с общим эмиттером инвертирует сигнал* (это относится не только к ключевому, но и к усилительному режиму работы, о котором будет рассказано далее). При этом на нагрузке (лампочке), которая подключена к питанию, а не к общей для входа и выхода каскада «земле», все в порядке — она горит, когда на входе сигнал есть, так что визуальный сигнал не инвертирован.

Для подключения мощных нагрузок к маломощным управляющим схемам употребляют различные схемы совместного включения транзисторов. Так, *транзистор Дарлингтона* (его часто называют транзистором с *супербетой*, мы будем называть его и так, и так) представляет собой две транзисторные структуры, включенные каскадно (рис. 6.5, *а*). Разумеется, можно соорудить такую структуру самостоятельно (левый транзистор обычно меньшей мощности, чем правый), но существуют и приборы, выпускаемые промышленно (на рис. 6.5, *а* общий корпус показан пунктиром). Величина β для них равна произведению коэффициентов усиления для каждого из транзисторов и может составлять до нескольких тысяч. При использовании таких «супербета»-транзисторов обязательно следует иметь в виду то обстоятельство, что рабочее напряжение между базой и эмиттером у них будет составлять примерно удвоенную величину от обычного транзистора — т. е. 1,2–1,4 В. Сопротивление резистора, как сказано ранее, принципиального значения не имеет и для мощных транзисторов может составлять несколько килоом.

а *б*

Рис. 6.5. Другие схемы подключения: *а* — транзистор Дарлингтона; *б* — параллельное включение транзисторов

На рис. 6.5, *б* приведена редко требующаяся, но весьма полезная, схема параллельного включения мощных транзисторов с целью увеличения допустимого коллекторного тока и рассеиваемой мощности (см. далее). Она немного напоминает схему Дарлингтона, но никакого умножения «бет» там, естественно, не происходит — суммируются только предельно допустимые показатели. Поскольку транзисторы всегда немного отличаются друг от друга, то для выравнивания токов через них в этой схеме служат резисторы в эмиттерных цепях, которые нужно выбирать так, чтобы падение напряжения на них при максимальном токе составляло примерно

0,2 В. Естественно, эти резисторы ухудшают КПД, поэтому для подобных целей удобнее использовать мощные полевые транзисторы, для которых в аналогичном включении дополнительных резисторов не требуется.

Усилительный режим работы биполярного транзистора

Теперь займемся собственно *усилительным режимом*. Из сказанного ясно, что между режимом насыщения и режимом отсечки должен существовать какой-то промежуточный режим — например, когда лампочка на рис. 6.4 горит вполнакала. Действительно, в некотором диапазоне базовых токов (и соответствующих им базовых напряжений) ток коллектора и соответствующее ему напряжение на коллекторе будут плавно меняться. Соотношение между токами здесь будет определяться величиной коэффициента усиления по току для малого сигнала (*коэффициента передачи тока в статическом режиме*), который по некоторым причинам обозначается весьма сложно: $h_{21э}$ (на Западе — h_{FE}) — обозначение, пришедшее из математической модели транзистора. В первом приближении коэффициент $h_{21э}$ можно считать равным коэффициенту β, хотя он всегда больше последнего. Учтите, что в справочниках иногда приводятся именно $h_{21э}$, а иногда β, так что будьте внимательны. Разброс $h_{21э}$ для конкретных экземпляров весьма велик (и сама величина сильно зависит от температуры и текущих значений потенциалов на выводах), поэтому в справочниках приводят либо минимальное, либо граничные значения (от и до).

Поэкспериментировать с усилительным режимом транзистора и заодно научиться измерять $h_{21э}$ можно с помощью схемы, приведенной на рис. 6.6.

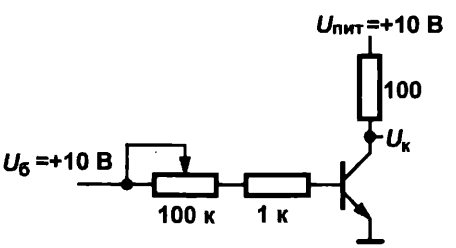


Рис. 6.6. Включение биполярного транзистора по схеме с общим эмиттером в усилительном режиме

Переменный резистор должен иметь достаточно большое сопротивление, чтобы при выведенном в крайнее левое положение его движке ток базы заведомо удовлетворял соотношению $I_б \cdot h_{21э} << I_к$. Коллекторный ток $I_к$ в этом случае определяется нагрузкой. Для современных маломощных транзисторов $h_{21э}$ составляет величину от 100 до 1000. Так как в коллекторе нагрузка 100 Ом, то для них переменник разумно выбирать номиналом примерно 100 кОм. Для транзисторов средней и большой мощности $h_{21э}$ обычно меньше и лежит в пределах нескольких десятков, потому для них величину переменного резистора можно уменьшить до 10–20 кОм.

Выведя движок в крайнее левое по схеме положение, мы задаем минимально возможный ток базы. В этом его положении следует включить питание и убедиться с помощью осциллографа или мультиметра, что транзистор близок к отсечке: напряжение на коллекторе $U_к$ будет почти равно напряжению питания (но не совсем — мы уже говорили, что для полной отсечки нужно соединить выводы базы и эмиттера между собой). Осторожно перемещая движок переменника, мы увидим, как напряжение на коллекторе будет падать (а на нагрузке, соответственно, расти). Когда напряжение на коллекторе станет почти равным нулю (т. е. транзистор придет в состояние насыщения), эксперимент следует прекратить, иначе можно выжечь переход база-эмиттер слишком большим прямым током (для предотвращения этой ситуации последовательно с переменником установлен постоянный резистор небольшого номинала).

Вернем движок переменника в состояние, когда напряжение на коллекторе примерно равно половине напряжения питания. Это так называемая *рабочая точка* транзистора в схеме с общим эмиттером — если напряжение на базовом резисторе станет в определенных пределах колебаться, изменяя ток базы, то переменная составляющая напряжения на коллекторе будет повторять его форму (с точностью до наоборот, т. е. инвертируя сигнал, как мы говорили ранее), но усиленную по напряжению и току. Это и есть усилительный режим транзистора.

В какой степени входной сигнал может быть усилен? Все определяется знакомым нам коэффициентом $h_{21э}$. Его величину для того или иного экземпляра транзистора можно определить так: пусть при напряжении на коллекторе, равном ровно половине напряжения источника питания (т. е. 5 В — случай, показанный на рис. 6.6), сопротивление базового резистора составляет 10 кОм. Ток коллектора (при коллекторной нагрузке 100 Ом) составит 50 мА. Ток базы составит (10 В – 0,6 В)/10 кОм, т. е. чуть меньше 1 мА. Тогда их отношение и будет равно $h_{21э}$, т. е., в нашем случае, 50. Кстати, померить его позволяют и некоторые конструкции мультиметров (хотя лично мне ни разу в жизни не пришлось этого делать, и далее вы поймете почему).

А каков коэффициент усиления такой схемы по напряжению? Это зависит от соотношения резисторов в базе и в коллекторе. Например, если сопротивление базового резистора составляет 1 кОм, то изменение тока базы при изменении входного напряжения на 1 В составит 1 мА. А в пересчете через $h_{21э}$ это должно привести к изменению тока коллектора на 50 мА, что на нагрузке 100 Ом составит 5 В. То есть усиление по напряжению при таком соотношении резисторов будет равно 5. Чем ниже номинал резистора в базе и чем выше сопротивление нагрузки, тем больше коэффициент усиления по напряжению. В пределе, если положить базовый резистор равным нулю, а коллекторный — бесконечности, то предельный коэффициент усиления современных транзисторов по напряжению может составить величину порядка нескольких сотен (но не бесконечность, за счет того, что база имеет собственное входное сопротивление, а коллектор — собственное выходное). Обратите внимание на это обстоятельство — что при повышении величины сопротивления в коллекторе коэффициент усиления увеличивается. В частности, это означает, что лучше вместо резистора включать источник тока, у которого выходное сопротив-

ление очень велико. Именно так и поступают во многих случаях, особенно часто — в усилительных микросхемах.

Схема усилителя, показанная на рис. 6.6, исключительно плоха. В самом деле, все зависит от величины коэффициента $h_{21э}$, а он, во-первых, «гуляет» от транзистора к транзистору, а во-вторых, очень сильно зависит от температуры (при повышении температуры повышается) и к тому же меняется прямо в ходе нашего эксперимента при изменении потенциала коллектора. И чтобы понять, как правильно построить усилительный транзисторный каскад со стабильными параметрами, нужно ознакомиться еще с одной схемой включения транзистора — схемой с общим коллектором.

Включение транзистора с общим коллектором

Схема *с общим коллектором* (о. к.) показана на рис. 6.7. Памятуя, что напряжение базы и эмиттера никогда не отличается более чем на 0,6–0,7 В, мы придем к выводу, что выходное напряжение такой схемы должно быть меньше входного именно на эту величину. Это так и есть — схема с общим коллектором иначе называется *эмиттерным повторителем*, поскольку выходное напряжение повторяет входное — за вычетом все тех же 0,6 В. Каков же смысл этой схемы?

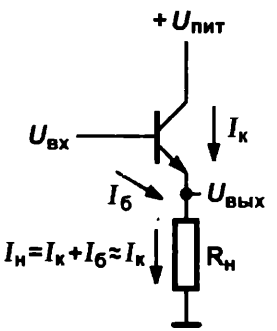


Рис. 6.7. Включение биполярного транзистора по схеме с общим коллектором

Дело в том, что схема, показанная на рис. 6.7, усиливает сигнал по току (в количество раз, определяемое величиной $h_{21э}$), что равносильно тому, что собственное входное сопротивление этой схемы ровно в $h_{21э}$ больше того сопротивления, которое стоит в цепи эмиттера. Поэтому в этой схеме мы можем подавать на «голый» вывод базы напряжение без опасности сжечь переход база-эмиттер. Иногда это полезно само по себе, если не слишком мощный источник (т. е. обладающий высоким выходным сопротивлением) нужно согласовать с мощной нагрузкой (в *главе 9* мы увидим, как это используется в источниках питания). Кстати, схема с о. к. *не инвертирует* сигнал — в отличие от схемы с о. э.

Но главной особенностью схемы с общим коллектором является то, что ее характеристики исключительно стабильны и не зависят от конкретного транзистора, — до тех пор, пока вы, разумеется, не выйдете за пределы возможного. Так как сопро-

тивление нагрузки в эмиттере и входное напряжение схемы практически однозначно задают ток коллектора, то характеристики транзистора в этом деле никак не участвуют.

Для объяснения этого факта заметим, что ток коллектора и ток эмиттера, т. е. ток через нагрузку, связаны между собой соотношением $I_н = I_к + I_б$, но ток базы мал по сравнению с током коллектора, потому мы им пренебрегаем и с достаточной степенью точности полагаем, что $I_н = I_к$. Но напряжение на нагрузке будет всегда равно входному напряжению минус $U_{бэ}$, которое, как мы уже выучили, всегда 0,6 В, т. е. ток в нагрузке есть $(U_{вх} - U_{бэ})/R_н$, и окончательно получаем, что $I_к = (U_{вх} - U_{бэ})/R_н$. Разумеется, мы по ходу дела использовали два допущения (что $I_б << I_к$ и что $U_{бэ}$ есть в точности 0,6 В — и то, и другое не всегда именно так), но мы же давно договорились, что не станем высчитывать характеристики схем с точностью до единиц процентов!

Ограничение, которое накладывается транзистором, будет проявляться, только если мы попробуем делать $R_н$ все меньше и меньше: в конце концов, либо ток коллектора, либо мощность, на нем выделяемая (она равна $(U_{пит} - U_{вых})\cdot I_к$), превысят предельно допустимые значения, и либо сгорит коллекторный переход, либо (если $I_к$ чем-то лимитирован) то же произойдет с переходом база-эмиттер. Зато в допустимых пределах мы можем со схемой эмиттерного повторителя творить что угодно, и соотношение $I_к = (U_{вх} - U_{бэ})/R_н$ будет всегда выполняться.

Про такую схему говорят, что она *охвачена стопроцентной отрицательной обратной связью по напряжению*. Об обратной связи мы подробнее поговорим в *главе 12*, посвященной операционным усилителям, а сейчас нам важно, что такая обратная связь ведет к стабилизации параметров схемы и их независимости как от конкретного экземпляра транзистора, так и от температуры. Но ведь это именно то, чего нам так не хватало в классической схеме с общим эмиттером! Нельзя ли их как-то скомбинировать?

Стабильный усилительный каскад на транзисторе

Действительно, «правильный» усилительный каскад на транзисторе есть комбинация обеих схем, показанная на рис. 6.8. Для конкретности предположим, что $U_{пит} = 10$ В, $U_{вх} = 5$ В (постоянная составляющая). Как правильно рассчитать сопротивления $R_э$ и $R_к$? Заметим, что схема обладает двумя выходами, из которых нас больше интересует выход 1 (выход усилителя напряжения, соответствующий выходу в схеме с общим эмиттером, показанной на рис. 6.6).

При нормальной работе каскада (чтобы обеспечить максимально возможный размах напряжения на выходе) разумно принять, что в состоянии покоя, т. е. когда $U_{вх}$ равно именно 5 В, на выходе (на коллекторе транзистора) должна быть половина питания, т. е., в нашем случае, тоже примерно 5 В. Это напряжение зависит от коллекторного тока и от сопротивления нагрузки по этому выходу, которое равно в нашем случае $R_к$. Как правило, сопротивление нагрузки $R_к$ нам задано — примем для определенности, что $R_к = 5{,}1$ кОм. Это означает, что в «хорошем» режиме, чтобы обеспечить $U_{вых1} = 5$ В, ток коллектора должен составлять 1 мА — посчитайте по закону Ома!

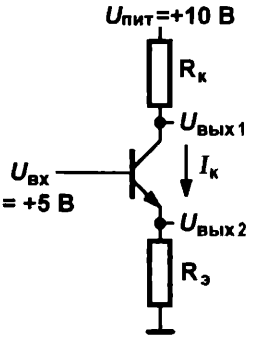


Рис. 6.8. Стандартный усилительный каскад на биполярном транзисторе

Заметки на полях

На самом деле средний ток коллектора в маломощном биполярном транзисторном каскаде и должен составлять величину порядка 1 мА — если он много меньше, то в дело вступают шумы и прочие неидеальности транзистора, а если много больше — это неэкономно с точки зрения расходования энергии источника, да и транзисторы нужно тогда выбирать более мощные, а у них намного больше шумы, утечки, они дороже, крупнее...

Но ток коллектора мы уже умеем рассчитывать, исходя из закономерностей для каскада с о. к., — он ведь равен $(U_{вх} - U_{бэ})/R_э$ (в нашем случае $R_э$ и есть $R_н$). Из этих условий получается, что резистор $R_э$ должен быть равен 4,3 кОм (мы всегда выбираем ближайшее из стандартного ряда сопротивлений и больше не будем об этом упоминать). Мы не сильно нарушим законы природы, если просто положим в этой схеме $R_э = R_к = 5{,}1$ кОм, — с точностью до десятых вольта выходные напряжения по обоим выходам будут равны — проверьте!

Такая (очень хорошая и стабильная) схема не обеспечит нам никакого усиления по напряжению, что легко проверить, если при рассчитанных параметрах увеличить $U_{вх}$, скажем, на 1 В. Напряжение на эмиттере увеличится также на 1 В, общий ток коллектора-эмиттера возрастет на 0,2 мА (1 В/5 кОм), что изменит падение напряжения на коллекторном резисторе также на 1 В в меньшую сторону (помните, что выходы инвертированы?). Зато! Мы в таком случае имеем схему, которая имеет два совершенно симметричных выхода: один инвертирующий, другой точно совпадающий по фазе с входным сигналом. Это дорогого стоит!

Единственное, что портит картинку — факт, что выходные сопротивления такой схемы сильно разнятся. Нагрузив нижний выход ($U_{вых2}$) еще какой-то нагрузкой (что равносильно присоединению параллельного резистора к $R_э$), мы изменим общий ток коллектора, и напряжение верхнего выхода ($U_{вых1}$) также изменится. А обратного не получается, т. е. если мы уменьшим $R_к$, нагрузив его, то $U_{вых1}$ изменится — но это практически никоим образом не скажется на $U_{вых2}$!

Как нам обеспечить полную (или близкую к таковой) симметричность схемы усилителя — чуть далее. А пока нас занимает вопрос — так как же на этом якобы усилителе что-нибудь усилить? У меня есть микрофон или гитарный звукосниматель

с выходом 1 мВ. Хочу получить на выходе хотя бы 100 мВ, чтобы хватило для линейного входа усилителя — ну и? Оказывается, все просто, нужно лишь поступиться принципами, от чего предостерегала незабвенная Нина Андреева еще в советские времена.

Принципы заключаются в следующем: в рассчитанной схеме мы старались все сбалансировать и обеспечить наилучший режим работы транзистора. Но зло и добро в мире, говорят, существуют в одинаковых количествах: если режим наилучший, значит, что-то будет наихудшим. В нашем случае — усиление по напряжению. Ранее мы утверждали, что коэффициент усиления по напряжению каскада с общим эмиттером зависит от соотношения сопротивлений (т. е. токов в базе и коллекторе). Сделав его неоптимальным для транзистора, мы можем что-то улучшить для себя.

Практически это делается так: пусть мы предполагаем, что максимально возможная амплитуда на входе каскада (относительно среднего значения) не превысит, допустим, 1 В. При минимуме сигнала напряжение на базе не должно быть меньше 1,7 В, иначе транзистор запрется, и сигнал будет ограничен «снизу». Примем его равным 2 В для надежности. Номинал эмиттерного резистора $R_э$ (при все том же оптимальном токе коллектора 1 мА) будет тогда равен 1,3 кОм. Нагрузка коллектора ($R_к$) пусть останется такой же — 5,1 кОм. Обратите внимание, что на выходе $U_{вых1}$ среднее напряжение — напряжение покоя — в этом случае осталось тем же самым (5 В).

При таких параметрах каждый вольт изменения напряжения на входе даст уже примерно 4 вольта изменения напряжения на выходе $U_{вых1}$, т. е. коэффициент усиления по напряжению составит 4 и всегда будет примерно равен соотношению резисторов в коллекторе и эмиттере. Мы можем в определенных пределах увеличить этот коэффициент, уменьшая номинал $R_э$ вплоть до нуля, тем самым все больше дестабилизируя схему (как показано при описании схемы с общим эмиттером) и одновременно уменьшая диапазон усиливаемых входных напряжений. Интересным свойством рассмотренной схемы является то, что абсолютное значение напряжения питания здесь не важно, — рассчитанный на одно питание, каскад сохранит все свои свойства (кроме максимально допустимого выходного напряжения) и при другом, — таковы свойства систем с обратной связью.

Для усилителей переменного тока хорошим — и часто используемым — приемом является шунтирование эмиттерного резистора конденсатором большой емкости. В результате режим усилителя по постоянному току (точка покоя, т. е. напряжение на коллекторе) обеспечен, а при наличии переменного входного напряжения эмиттерный резистор по номиналу уменьшается (к нему оказывается подключен параллельно конденсатор, сопротивление которого тем меньше, чем выше частота, как мы узнали из *главы 5*), поэтому растет и коэффициент усиления по напряжению всей схемы.

Теоретически транзисторный каскад на хорошем биполярном транзисторе в одиночку может усилить переменное напряжение с размахом 1 мВ раз в сто — если он правильно спроектирован, однако лучше полагаться на величину в 10–30 раз (если

помните, максимальное увеличение по напряжению для современных биполярных транзисторов ограничено несколькими сотнями, если без всякой стабилизации). Еще лучше в этом случае работают операционные усилители, но чтобы перейти к ним (см. *главу 12*), надо еще много понять о работе простых транзисторных каскадов.

Дифференциальный каскад

Значительно улучшает схему использование в паре двух одинаковых транзисторов, соединенных эмиттерами, — так называемого *дифференциального каскада* (рис. 6.9). Дифференциальные каскады в силу их удобства широко применяли еще в эпоху недоступности микросхем (в том числе даже и в ламповые времена), но теперь их отдельно почти не используют, кроме некоторых областей вроде звукотехники. Они являются основой операционных усилителей, которые имеет смысл рассматривать как единое целое. Тем не менее, понимание принципов работы дифференциального каскада необходимо, и мы рассмотрим его вкратце, а потом (в *главе 8*) построим на его основе простейший звуковой усилитель.

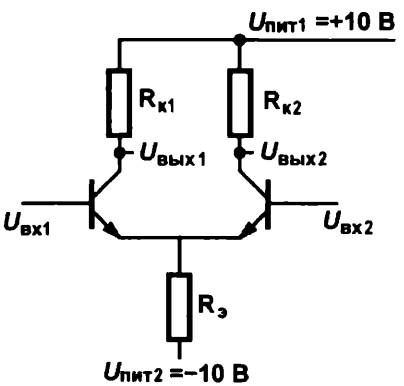


Рис. 6.9. Дифференциальный каскад на биполярных транзисторах

Дифференциальный каскад, как он показан на рис. 6.9, предполагает два раздельных одинаковых питания (плюс и минус) относительно «земли», но для самого каскада это не более, чем условность, — питание всего каскада можно рассматривать, как однополярное (и равное в нашем случае 10 + 10 = 20 В), просто входной сигнал должен находиться где-то посередине между питаниями. Ради удобства проектирования схем источник входного напряжения всегда привязывают к «земле», потенциал которой находится посередине между потенциалами источников питания самого каскада, т. е. общее питание рассматривают как разделенное на два: положительное и отрицательное. Относительно этой же «земли» мы будем также отсчитывать выходные напряжения $U_{вых1}$ и $U_{вых2}$.

С учетом того, что база и эмиттер транзистора всегда привязаны друг к другу, в этой схеме обе базы в рабочем режиме всегда будут иметь одинаковый потенциал. Поэтому если на них подавать один и тот же сигнал (базовые резисторы на рис. 6.9 не показаны), то ничего происходить не будет: току течь некуда, т. к. все

под одним и тем же напряжением. Вся конструкция из двух транзисторов будет смещаться относительно «земли» в соответствии с поданным сигналом, а на выходах ничего и не шелохнется — это легко проверить. Такой сигнал называют *синфазным*.

Иное дело, если сигналы на входах различаются, — их разность будет усиливаться. Такой сигнал называют *дифференциальным*. Это основное свойство дифференциального усилителя, которое позволяет выделять небольшой сигнал на фоне довольно большой помехи. Помеха одинаково — синфазно — действует на оба входа, а полезный сигнал усиливается.

Мы не станем здесь подробно разбирать работу этой схемы (рекомендую [4, 5]), только укажем некоторые ее особенности:

- входное сопротивление дифференциального каскада равно входному сопротивлению каскада с общим коллектором;
- усиление по напряжению (дифференциальному) составляет 100 и более раз. Если вы хотите получить точно определенный коэффициент усиления, в каждый из эмиттеров нужно ввести по одинаковому резистору — тогда K_{yc} будет определяться, как для каскада, показанного на рис. 6.7. Но обычно в таком режиме дифференциальные усилители не используют — их включают в системы с общей обратной связью, которая и задает необходимый коэффициент усиления (см. *главу 8*);
- выходы строго симметричны;
- резистор $R_{к1}$, если не используется $U_{вых1}$, вообще можно исключить, или наоборот — смотря, какой выход (прямой или инверсный) использовать.

Полевые транзисторы

Типы *полевых транзисторов* гораздо более разнообразны, чем биполярных (к полевым, кстати, и принадлежал самый первый прототип транзистора, изобретенный Шокли еще в 1946 году). Только основных разновидностей существует более десятка, но всем им присущи общие черты, которые мы сейчас кратко и рассмотрим. Отметим, что полевики гораздо ближе к той модели транзистора, когда промежуток коллектор-эмиттер (сток-исток — как принято их называть у полевых транзисторов) представляется как управляемое сопротивление, — для полевых транзисторов это действительно сопротивление.

Простейший полевой транзистор с *p-n*-переходом показан на рис. 6.10, *а* — в данном случае с *n*-каналом. Аналогичные базе, коллектору и эмиттеру выводы называются здесь *затвор* (*gate*), *сток* (*drain*) и *исток* (*source*). Если потенциал затвора равен потенциалу истока (т. е. имеется в виду аналог замыкания цепи база-эмиттер у биполярного), то, в отличие от биполярного, полевой транзистор с *p-n*-переходом открыт. Но есть и еще одно существенное отличие: если биполярный транзистор при полном открывании имеет почти нулевое сопротивление цепи коллектор-эмиттер, то полевой в этих условиях работает довольно стабильным источником

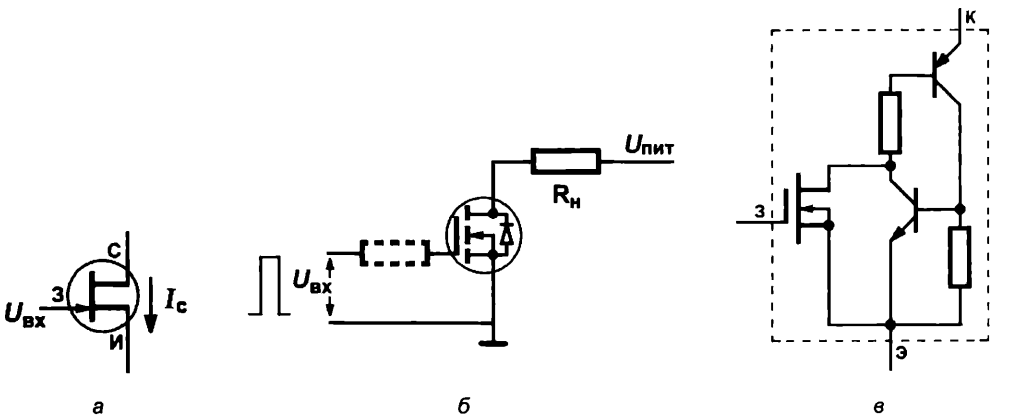
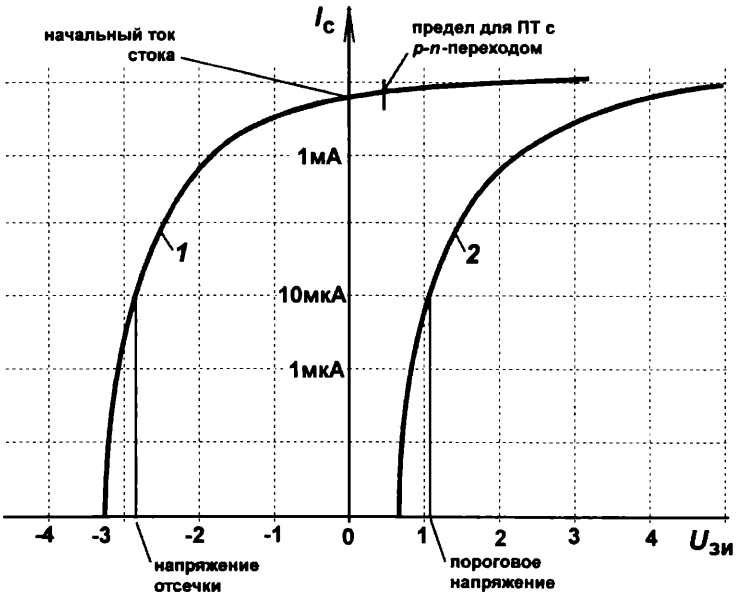


Рис. 6.10. Полевые транзисторы: *а* — включение полевого транзистора с *p-n*-переходом и *n*-каналом; *б* — полевой транзистор с изолированным затвором (MOSFET) в режиме ключа; *в* — внутренняя структура IGBT-транзистора

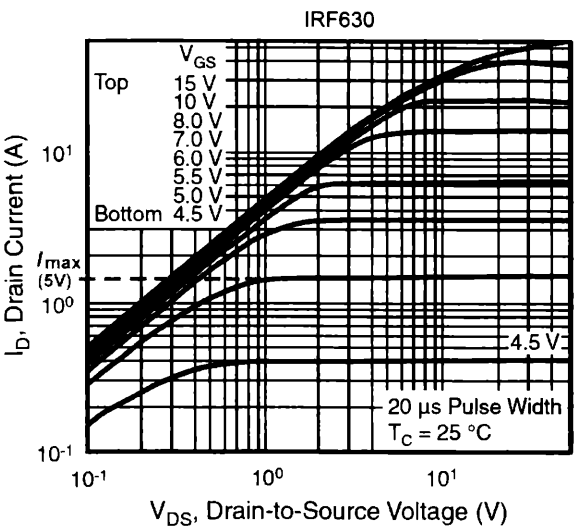
тока, — ток в цепи истока, как мы увидим, почти не зависит от напряжения на стоке. В *главе 9* мы воспользуемся этим фактом для создания простейшего источника стабильного тока (см. там рис. 9.12). Запереть транзистор с *p-n*-переходом и *n*-каналом удается подачей отрицательного напряжения порядка 2–5 В на затвор относительно истока — это т. н. *напряжение отсечки* (рис. 6.11, *а*), а в промежутке он находится в активном режиме, когда ток стока зависит от напряжения на затворе.

Уникальной особенностью полевого транзистора является то, что в рабочем режиме он фактически не потребляет тока по входу затвора, — достаточно иметь на нем соответствующий потенциал, ведь диод затвор-исток в рабочем режиме смещен в обратном направлении, и ток через него определяется только токами утечки, которые равны нано- и микроамперам, как говорилось ранее! В этом отношении полевой транзистор аналогичен электронной лампе. А если мы сместим этот переход в положительном направлении (когда потенциал затвора превысит потенциал истока, и диод затвор-исток откроется), то полевой транзистор с *p-n*-переходом уже перестанет работать как транзистор (см. рис. 6.11, *а*).

В различных типах полевых транзисторов с изолированным затвором (так называемых *МОП-транзисторах* — от «металл-оксид-полупроводник». По-английски их называют MOS, или, иначе, MOSFET) этот самый затвор вообще изолирован от цепи сток-исток тонким слоем оксида кремния SiO_2, и там вообще нет и не может быть никакого тока через цепь затвора (рис. 6.10, *б*). На подобных транзисторах построены почти все современные логические микросхемы, отличающиеся практически нулевым потреблением тока в статическом режиме (см. *главу 15*). Правда, когда на затвор подается переменное напряжение или, что то же самое, возрастающий или падающий фронт импульса, в дело вступает конденсатор, образованный затвором и истоком. Его емкость может составлять десятки, а у мощных транзисторов — сотни и даже тысячи пикофарад. Как следует из *главы 5*, перезаряд этого конденсатора может приводить к значительному реактивному току в цепи затвора.



а



б

Рис. 6.11. *а* — зависимость тока стока (*Ic*) от напряжения затвор-исток (*Uзи*) для двух основных типов ПТ с *n*-каналом: 1 — с *p-n*-переходом, 2 — с изолированным затвором (МОП, MOSFET); *б* — зависимость тока стока (I_D) от напряжения сток-исток (V_{DS}) при разных потенциалах затвора относительно истока (V_{GS}) для мощного MOSFET-транзистора IRF630

На рис. 6.11, *а* показана типовая зависимость тока стока (*Ic*) от напряжения затвор-исток (*Uзи*) для двух самых распространенных типов полевых транзисторов (ПТ). Оба типа представлены в вариантах с *n*-каналом (для *p*-канала все полярности меняются на противоположные). Кривая *1* на графике представляет полевые транзисторы с *p-n*-переходом. К ним относится большинство старых советских маломощных разновидностей — например, популярные КП302 и КП303. Из графика видно, на чем основано употребление такого транзистора, как источника тока: если замкнуть затвор с истоком, то по цепи сток-исток потечет стабильный ток, мало зависящий от напряжения питания. Сама величина тока в этом состоянии зависит от конкретного экземпляра транзистора и называется *начальным током стока*.

Кривая *2* на рис. 6.11, *а* демонстрирует такую же зависимость, но для транзисторов с изолированным затвором: МОП-транзисторов или MOSFET, как их величают на Западе. К ним относится большинство современных типов, в том числе все мощные MOSFET-ключи. Старые образцы маломощных MOSFET-транзисторов (например, отечественные КП305, КП313, КП329) требовали для полного запирания небольшого отрицательного смещения на затворе относительно истока (порядка 2–4 В), их характеристика аналогична той, что показана на рис. 6.11, *а*, только у них нет ограничения на положительное напряжение на затворе, обязательного для транзисторов на *p-n*-переходе. Современные MOSFET-транзисторы (см. рис. 6.10, *б*), как показывает кривая *2* на рис. 6.11, *а*, управляются аналогично биполярному в схеме с общим эмиттером: при нулевом напряжении на затворе относительно истока транзистор заперт, при положительном напряжении порядка 2–10 В — открыт, причем в полностью открытом состоянии он представляет собой крайне малое сопротивление, — у некоторых типов менее 0,01 Ом. Обратите внимание на обратно включенный диод в структуре MOSFET-транзистора — он служит для ограничения выброса напряжения при коммутации индуктивной нагрузки (см. *главу 5*), т. е. (с некоторыми оговорками, касающимися очень мощных нагрузок) схемотехнику специально заботиться об этом не надо.

Подробности

Зависимость тока стока (I_D) от напряжения сток-исток (V_{DS}) при разных величинах управляющего напряжения на затворе (V_{GS}) для конкретного транзистора IRF630 представлена на рис. 6.11, *б*. Формально он подходит в качестве мощного ключа — предельно допустимый ток у него 9 А при напряжении аж до 200 В. Наклон кривых на линейном участке слева примерно соответствует указанному в характеристиках сопротивлению канала 0,4–0,5 Ом. *Пороговое напряжение затвора* (Gate Threshold Voltage), после которого транзистор начинает открываться, для этого типа равно 2–4 В, т. е. он формально должен управляться от выхода логических микросхем с питанием 5 В. Однако следите за руками: случай, когда такой транзистор управляется от микросхем с питанием 5 В, соответствует второй снизу кривой на графике (V_{GS} = 5 В). Максимальный ток, который сможет обеспечить этот транзистор при напряжении на затворе 5 В, составляет ~1,1 А и быстро падает при снижении управляющего напряжения всего на несколько десятых вольта (нижняя кривая на графике). Таким образом, как следует из графика, показанного на рис. 6.11, *б*, чтобы повысить рабочий ток, надо выбирать большее управляющее напряжение. Для IRF630 минимальное сопротивление канала, по спецификациям, равно 0,4 Ом при напряжении V_{GS} = 10 В — тогда, согласно графикам, ток стока даже превышает допустимые для этого транзистора в постоянно открытом состоянии 9 А. У других MOSFET-ключей положение может быть не столь тяжелым — IRF630 я здесь выбрал специально, как наглядный пример. Но если

вы не имеете под рукой подробных характеристик, то сочтите за правило, что у мощных ключевых MOSFET-транзисторов для максимального открытия канала управляющее напряжение затвор-исток должно быть 8–10 вольт, и вы не ошибетесь. Подробнее о подборе мощных полевых транзисторов в качестве MOSFET-ключей рассказано в *главе 22*.

Кстати, мощные полевые транзисторы — один из немногих типов полупроводниковых приборов, для которых в сторону отечественных разновидностей даже и смотреть не стоит. Сопротивление канала у большинства из пригодных для наших целей отечественных типов составляет единицы ома, т. е. на порядок больше современных западных. Приятное исключение составляет, например, КП922 с сопротивлением канала 0,2–0,4 Ом, но и его характеристики на фоне многочисленных образцов продукции International Rectifier совсем не впечатляют.

Если все-таки нужно управлять мощной нагрузкой от пятивольтового источника, то транзистор-ключ необходимо выбирать очень тщательно, благо фирма International Rectifier наплодила их предостаточно (подробнее о выборе ключа для этой цели рассказано в *главе 22*). Но одного такого отбора недостаточно для эффективного управления MOSFET-ключами при больших токах нагрузки.

Есть и еще одна важная причина, почему мощные MOSFET-транзисторы не подключают непосредственно к выходу микросхем (контроллеров) с пятивольтовым питанием. Ведь затвор мощного полевика, кроме всего прочего, представляет собой довольно большую емкость (для старинного IRFZ44 — 1900 пФ, для IRF530 и других более современных несколько поменьше, порядка 900–1100 пФ), которую еще надо зарядить. И от переходных процессов при этом также происходят потери мощности, и тем более значительные, чем менее мощный выход контроллера. Его приходится умощнять дополнительно, и схема существенно усложняется.

Простой подсчет показывает, что при предельном токе, обеспечиваемом, например, выходом AVR-контроллеров (о них рассказано в *части IV* книги), равном 40 мА (при таком токе, конечно, ни о каких стабильных уровнях напряжения и речи не идет), заряд емкости около 1 нФ до напряжения 5 В будет проходить за время порядка 10 мкс. То есть уже на частотах порядка десятков килогерц фронт и спад импульсов отнимут существенную часть времени импульсов, а на полуоткрытых в это время транзисторах будет рассеиваться существенная часть мощности. Ну и, конечно, работа в непрерывном режиме фактически короткого замыкания далека от нормального режима функционирования выхода контроллера.

На частотах в сотни герц (как в штатном режиме ШИМ Arduino — см. *главу 22*) это еще будет сказываться не слишком сильно — время фронта-спада составит единицы процентов от времени открытого/закрытого состояния. Но, конечно, многое зависит от мощности: при токе в десятки ампер через ключ даже при напряжении 5–12 В эти проценты составят уже единицы ватт, и, как минимум, ключ придется ставить на радиатор. Один из простых способов умощнить выход микросхемы и снизить влияние подобного эффекта, а заодно уберечь этот выход от перегрузок, показан на рис. 22.2, *б*. А самым грамотным, конечно, будет использование интегрального драйвера, который выпускается специально для управления MOSFET-ключами (они обсуждаются в *главе 22*).

MOSFET-транзисторы выпускаются на мощности от единиц до сотен ватт и широко используются для управления мощностью двигателей, в импульсных источниках питания и в других подобных случаях. Подробнее применение таких транзисторов мы рассмотрим в *главе 9*, посвященной источникам питания, и в *главе 22* — в связи со схемами ШИМ-управления мощными нагрузками. Теперь, нагруженные этими знаниями, при оценке конкретной схемы, найденной в Интернете, вы будете более критично относиться к выбору транзисторов ее авторами, — во многих случаях можно подобрать гораздо более подходящий тип, ведь на сайте International Rectifier/Infineon (**www.irf.com**) их предлагаются сотни разновидностей.

Приведенные здесь примеры не исчерпывают разнообразия типов полевых транзисторов. Например, так называемые *IGBT-транзисторы* (**I**nsulated **G**ate **B**ipolar **T**ransistors, биполярный транзистор с изолированным затвором), появившиеся в 1980-е годы, объединяют в себе полевую и биполярную структуры, отчего управляющий электрод в них зовется, как и в полевых, затвором, а два других аналогично биполярным: коллектором и эмиттером. На самом деле IGBT-транзистор представляет собой довольно сложную полупроводниковую структуру (рис. 6.10, *в*) с положительной обратной связью между разнополярными «обычными» транзисторами и с управлением от полевого (сравните со структурой аналога однопереходного транзистора на рис. 10.3).

IGBT-транзисторы используются в качестве мощных ключей: десятки-сотни ампер при напряжениях более 1000 вольт. Управляются они положительным напряжением на затворе относительно эмиттера, причем у некоторых типов насыщение наступает уже при подаче 2,7–4 В на затвор, и эти транзисторы могут управляться непосредственно от логических схем. Платой за такую роскошь является довольно высокое напряжение насыщения между коллектором и эмиттером, характерное для биполярных транзисторов: от 1 В для относительно маломощных приборов (единицы ампер) до 2–3 В для более мощных (десятки и сотни ампер).

Выбор транзисторов

В заключение главы приведем критерии подбора биполярных и полевых транзисторов для конкретной схемы. Сейчас мы оставляем за скобками частотные характеристики транзисторов — будем считать, что достаточно выбрать прибор с рабочей частотой, примерно в 10 раз превышающей самые высокие частоты в схеме.

Особенности ключевого режима на высокой частоте

В силу того, что у прямоугольного импульса, как сказано в *главе 5*, верхняя частота неограниченна, может создаться искушение выбирать как можно более высокочастотные приборы. Но это не вполне разумно — достаточно выбрать компоненты с рабочей частотой примерно в 10–20 раз выше, чем основная частота прямоугольных сигналов. Быстродействие ключевых схем с общим эмиттером все равно будет существенно ниже ожидаемого, причем повышение частотных свойств транзистора не сильно поможет, и вот почему.

Если ток базы увеличить скачком, то нарастание тока коллектора будет происходить не сразу, а по кривой, аналогичной показанной на рис. 6.1 (если бы по оси абсцисс откладывалось не напряжение, а время). Иными словами, вывод биполярного транзи-

стора из состояния насыщения занимает определенное время, а форма прямоугольных импульсов на коллекторной нагрузке весьма сильно искажается. Это не будет иметь существенного значения для низкочастотных схем, рассматриваемых в этой книге, но может доставить много неприятностей, если вы попробуете, например, с помощью простого ключевого каскада управлять передачей импульсов в скоростных линиях связи. В свое время преодоление этого эффекта доставило немало хлопот конструкторам транзисторных логических схем. Для того чтобы обойти эту неприятность, существует несколько способов держать запертый транзистор на грани насыщения, но мы их в этой книге рассматривать не станем — ныне для упомянутых целей существуют готовые решения в интегральном исполнении.

Если у диодов определяющих критериев всего три (допустимый прямой ток, допустимое обратное напряжение и допустимая выделяющаяся мощность), то у транзисторов их много больше. Приведем часть из них:

- допустимый ток коллектора;
- допустимый ток базы;
- допустимая мощность на коллекторе (стоке);
- допустимое напряжение коллектор-эмиттер (сток-исток);
- допустимое напряжение коллектор-база (сток-затвор);
- допустимое обратное напряжение база-эмиттер и др.

Самыми критичными являются опять же три: допустимый ток коллектора, допустимая мощность на коллекторе и допустимое напряжение коллектор-эмиттер. Допустимое обратное напряжение база-эмиттер (т. е. отрицательное напряжение на базе при запертом транзисторе) для большинства типов кремниевых транзисторов, независимо от их мощности, составляет, увы, всего 5 В — это необходимо учитывать в схемах с участием последовательно включенных конденсаторов, где на базе могут возникать отрицательные выбросы напряжения (типичный пример приведен на рис. 16.1, *б*). На самом деле большинство транзисторов при кратковременном воздействии выдерживает много больше, но лучше не экспериментировать. Допустимое напряжение коллектор-база, как правило, примерно равно допустимому напряжению коллектор-эмиттер, которое для обычных типов маломощных транзисторов составляет несколько десятков вольт (хотя есть и экстремальные типы, которые могут коммутировать и сотни вольт). Чаще всего в пределах одного типа разные буквы означают разброс в допустимых напряжениях (и/или в коэффициентах усиления β): так, для КТ815А допустимое постоянное напряжение коллектор-эмиттер составляет 40 В, а для КТ815Г — 100 В.

Предельно допустимая мощность на коллекторе (то же самое справедливо для диодов) обычно определяется типом корпуса — один и тот же транзистор, помещенный в разные корпуса, может обеспечить разную выделяемую мощность. Критерием тут служит температура самого кристалла, которую померить ох как непросто! Для ориентировки можно указать, что транзисторы (и другие приборы), помещенные в распространенный корпус ТО-220 (корпуса транзисторов показаны на рис. 6.12), могут без дополнительного радиатора рассеивать мощность до 1–2 Вт, а маломощные типа КТ3102 (корпус типа ТО-92) — до 0,5 Вт. С радиатором возможности сильно возрастают — корпус типа ТО-220 может рассеять до 60 Вт тепла

без вреда для кристалла! Образцом тут могут служить микропроцессоры — какой-нибудь Pentium 4 на частоте 3 ГГц потребляет порядка 70–80 Вт мощности, но с внешним радиатором, дополнительно охлаждаемым специальным вентилятором, работает без вреда для многих миллионов транзисторов, которые он содержит. (Расчетом радиатора мы будем заниматься в *главе 9.*)

В любом случае следует выбирать минимально необходимый по мощности прибор — не только в целях экономии денег и места на печатной плате, но и потому, что чем меньше диод или транзистор, тем лучше у него остальные второстепенные характеристики: быстродействие, уровень собственных шумов, токи утечки и т. д. Но, как и в других случаях, запас обязательно следует иметь: если вы выберете для работы в цепи с напряжением 100 В и с токами до 1,5 А транзистор КТ815Г — это будет формально правильно, но я бы — для надежности — выбрал сюда что-нибудь помощнее.

Подробности

Есть правило, касающееся любых компонентов, не только диодов или транзисторов: из всех предельных параметров максимально допустимого значения в процессе работы может достигать только один, остальные должны оставаться как можно ниже (для транзисторов даже приводятся специальные графики, называемые *областью безопасной работы*). Так, если вы выбрали упомянутый КТ815Г для работы в цепи с напряжениями до 100 В — пусть предельные токи через него заведомо никогда не смогут превысить 0,5 А. Это будет правильно! Представьте себе йога, который тренирован для пребывания голым на холоде в минус 30° в течение часа, спокойно ходит по раскаленным угольям, выдерживает давление на грудную клетку большегрузного автомобиля в 10 тонн и при этом ломает кирпичи одним ударом ладони. А теперь заставьте его проделать все это одновременно! Конечно, не исключено, что он выдержит, — ну, а как нет?

В подавляющем большинстве случаев номенклатура отечественных транзисторов способна удовлетворить самого взыскательного разработчика. Я это пишу не для того, чтобы «поддержать отечественного производителя», а потому, что так и есть, — на практике достаточно располагать пятком-десятком типов транзисторов, чтобы этого хватило почти на все случаи жизни. Среди маломощных транзисторов это КТ3102 (КТ3107 — здесь и далее в скобках указывается комплементарный[1] *p-n-p*-вариант) с коэффициентом передачи тока $h_{21Э} = 100$. Из западных к ним ближе всего пара BC325 и BC327 (с индексом -40 в названии эти транзисторы имеют $h_{21Э} = 250$). Последние также превышают отечественные по допустимому току (0,8 А против 100 мА у КТ3012/07).

Лично мне в не слишком ответственных применениях очень нравятся архаичные маломощные транзисторы КТ315 (КТ361) — они имеют малые размеры и легко вписываются в современные платы с микросхемами (в том числе и с SMD-компонентами), потому что у них шаг между выводами 2,5 мм, выводы плоские и расположены также в одной плоскости. Хороши невзыскательные и дешевые тран-

[1] Комплементарный — букв. дополнительный, означает транзистор с такими же характеристиками, но противоположной полярности.

зисторы средней мощности в корпусе TO-126: КТ815 (КТ814) или КТ817 (КТ816), если требуется ток до 1–2 А. Западные практически полные их аналоги: BD135, BD137, BD139 (*n-p-n*) и BD136, BD138, BD140 (*p-n-p*). Если требуется высокий коэффициент усиления для средней мощности, стоит присмотреться к отечественным КТ646Б (КТ639В, КТ639И), а также к КТ972 (КТ973), построенным по «дарлингтоновской» схеме (западные аналоги BD875, BD682).

Среди мощных транзисторов можно отдать предпочтение КТ819 (КТ818) или, когда требуется «супербета», — КТ829 (*n-p-n*), а также очень мощной комплементарной «дарлингтоновской» паре КТ827 (КТ825). Последнюю пару можно заменить на импортные BDW93C, BDW94C (см. рис. 8.1 и 9.10), но по характеристикам они отличаются мало. Выпускаются почти все эти типы мощных транзисторов в основном в корпусах типа TO-220, но самая мощная пара КТ827 (КТ825) доступна в металлических корпусах TO-3, что лучше, чем дешевый TO-220, т. к. рассеиваемая мощность оказывается раза в 2–4 выше: типовая мощность транзистора в корпусе TO-220 равна 20–45 Вт, а в корпусе TO-3: 80–125 Вт. Но корпус TO-3 намного неудобнее в технологическом плане, потому что крепление его к теплоотводящему радиатору гораздо сложнее, и готовый радиатор подобрать под них нелегко. Впрочем, и мощности такие требуются нечасто.

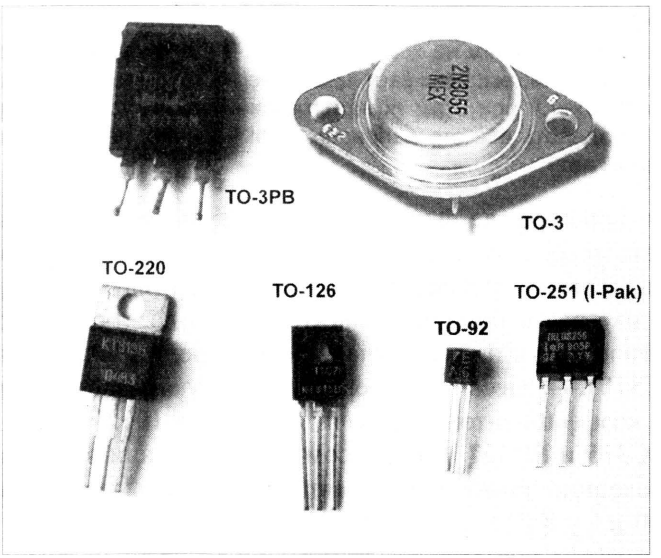


Рис. 6.12. Различные типы корпусов транзисторов

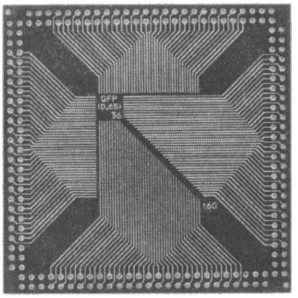
Если трудно подобрать мощную «дарлингтоновскую» пару, то не забывайте, что дарлингтоновский транзистор всегда можно изготовить самостоятельно (см. рис. 6.5, *а*). Причем, в этом случае, оба транзистора, составляющие дарлингтоновскую пару, должны устанавливаться на один радиатор, но так как они соединены коллекторами, то это проблем не добавляет, и изолирующая прокладка не потребуется.

Подробности

Для удобства мы употребляем западные наименования корпусов транзисторов, и далее то же самое будет относиться к микросхемам. Разумеется, названия корпусов для микросхем DIP, SOIC, SOT и др., как и транзисторных TO-92, T-220 и пр., применимы только к импортным компонентам, однако соответствующие отечественные корпуса микросхем имеют столь замысловатую систему обозначений, что для удобства и унификации мы будем в этой книге пользоваться исключительно западными названиями корпусов, — еще и по той причине, что найти чертежи любого импортного корпуса намного легче, чем отечественного. Учтите, что имеется некоторая разница в шаге выводов (поскольку на Западе приняты дюймовые стандарты, а у нас — метрические), хотя существенна она только для микросхем с большим числом выводов (см. *главу 11*).

При замене следует учитывать, что с корпусом мощного транзистора всегда электрически соединен его коллектор (если это корпус TO-220, то коллектор — всегда средний контакт), а вот разводка двух оставшихся выводов может различаться, и ее нужно обязательно проверять по справочнику.

Все сказанное относится к низкочастотным транзисторам, которые можно употреблять для источников питания, в качестве ключей управления индикацией или для усилителей звуковой частоты. Для многих других применений, где требуется быстрое срабатывание в ключевом режиме или высокие частоты усиления, такие транзисторы, естественно, не годятся, но точных рекомендаций на все случаи жизни дать нельзя. Учтите только, что транзисторы в одинаковых корпусах обычно имеют близкую мощность, и в безвыходном положении можно попробовать заменить неизвестный импортный транзистор похожим по внешнему виду из того, что есть под рукой.



ГЛАВА 7

Ошеломляющее разнообразие электронного мира

Реле, стабилитроны, светодиоды...

> Нет такой пустыни, где бы птица не могла пролететь над головой, где бы рыба не могла вынырнуть из воды, где бы кролик не мог выскочить из своей норки. А мне кажется, что и птицы, и рыбы, и кролики — все стали шпионами кардинала.
>
> *А. Дюма. «Три мушкетера»*

Полупроводниковая промышленность выпускает очень много типов электронных приборов. В этой главе мы подробно рассмотрим только основные, самые широко употребляемые их разновидности. По ходу изложения нам встретятся и другие приборы, имеющие более узкий спектр применения, и необходимые пояснения будут даны в соответствующих главах. Все эти компоненты, вместе с диодами и транзисторами, в отличие от интегральных микросхем, называют еще *дискретными*, а основанную на их использовании схемотехнику — *дискретной схемотехникой*. Не следует думать, что последняя, в связи с дешевизной микросхем, полностью вышла из моды: во-первых, многие узлы современных схем все равно делают на дискретных компонентах, во-вторых, как ни странно, иногда только использование дискретной схемотехники позволяет достичь наивысшего качества — типичным примером служат высококачественные звуковые усилители мощности.

Электромагнитные реле

Конечно, выдающийся американский физик Джозеф Генри, помогая художнику Сэмюэлу Морзе в постройке телеграфа, и не думал ни о какой электронике, которая потом завоюет мир. *Электромагнитное реле* он изобрел даже не в рамках фундаментальной науки, которая, как известно, есть способ познания мира, и чурается практики, а просто, чтобы «помочь товарищу», который, впрочем, наверняка платил неплохие деньги.

Так это было или иначе — важно, что электромагнитное реле стало одним из самых главных технологических изобретений XIX века. По популярности ему не затмить,

конечно, электрического освещения, электрогенератора и электродвигателя, телеграфа, телефона и прочих достижений «века электричества», но факт, что именно этот не очень известный широкой публике приборчик еще недавно был одним из важнейших компонентов любой электрической системы и широко используется до сих пор.

Реле стало первым в истории — задолго до ламп и транзисторов — усилителем электрических сигналов. С помощью реле напрямую не усилить предвыборную речь кандидата в президенты, но если ее, по современной моде, закодировать нулями-единицами, то реле справится с такой задачей ничуть не хуже любого другого устройства, — именно на этом свойстве было основано его применение в телеграфе Морзе.

Конечно, быстродействие реле, как ключевого элемента, оставляет желать лучшего — даже о килогерцах здесь речь не идет, обычная скорость срабатывания для самых малогабаритных и быстродействующих реле составляет десятки миллисекунд, что соответствует частотам в десятки герц. Но в режиме быстрого переключения реле использовать и не надо, для этого существуют другие электронные компоненты. Реле применяют там, где нужно надежно коммутировать нагрузку с минимальными потерями в контакте. Огромным преимуществом реле является не только полная изоляция между входом и выходом (как говорят, *гальваническая развязка*), но и низкое сопротивление контактов. По этой причине их в наше время используют, например, для коммутации в измерительных схемах, где очень важно, чтобы сопротивление измерительных цепей было минимальным и стабильным. Учтите, что указываемые в справочниках параметры контактов (типа «переходное сопротивление не более 1 Ом») обычно сильно завышены, они рассчитаны на наихудший случай.

Подробности

Немаловажное преимущество электромагнитного реле перед более современными способами коммутации сигнала с помощью электронных ключей заключается в том, что оно принципиально безразлично к полярности тока, проходящего через контакты. Одним и тем же реле можно коммутировать постоянный и переменный ток любой формы, направления и амплитуды (не превышающей предельно допустимые значения, конечно). Иными словами, контакты реле являются полностью пассивным элементом схемы — подобно обычному выключателю, не оказывая никакого влияния на коммутируемый сигнал. Практически влияние, конечно, имеется, хотя бы потому, что контакты обладают весьма малым, но конечным сопротивлением, но правильным выбором реле это влияние можно свести к минимуму.

На рис. 7.1, *а* схематически изображено устройство простейшего электромагнитного реле. Любое реле — независимо от конструкции — обязательно содержит три главных компонента: обмотку с сердечником, якорь и контакты. Исключение составляют так называемые *герконовые* реле, у которых якорем служат сами контакты.

Геркон

Геркон расшифровывается как «*гер*метизированный *кон*такт». Герконы выпускаются отдельно и представляют собой стеклянную трубочку с двумя или тремя выводами от запаянного в нее контакта (простого или перекидного), защищенного таким образом от

влияния внешней среды. Контакт под воздействием внешнего магнитного поля — например, при поднесении постоянного магнита — может замыкаться и размыкаться. Герконовые реле обычно представляют собой подобный геркон, на который намотана обмотка с теми или иными параметрами.

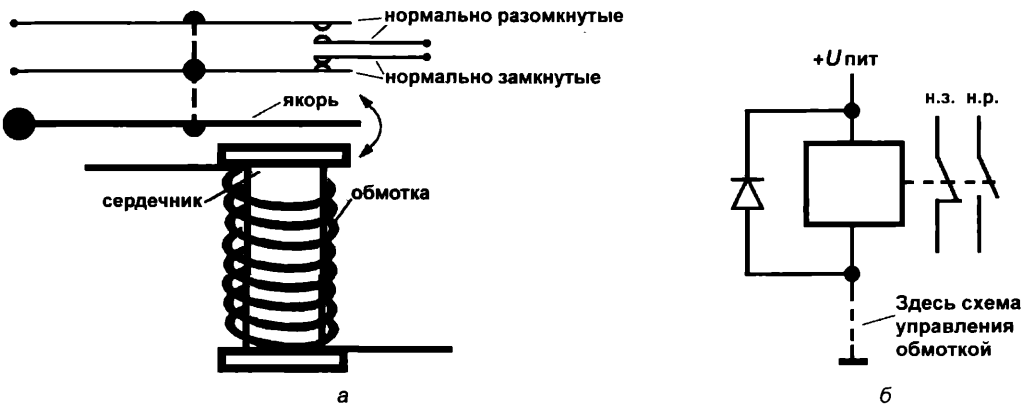


Рис. 7.1. Схематическое устройство (*а*) и рекомендуемая схема включения (*б*) электромагнитного реле

Обмотка электромагнитного реле представляет собой катушку индуктивности (соленоид), около которой (или внутри которой) при подаче тока перемещается якорь, выполненный из ферромагнитного материала. Теорию этого процесса излагать слишком долго, да к тому же она и не нужна для практических целей. Важно понимать, что при подаче переменного или постоянного тока якорь притягивается к сердечнику обмотки и через тягу из изолирующего материала (на рис. 7.1, *б* она показана пунктиром) приводит к перемещению подпружиненных контактов, которые замыкаются (если были «нормально разомкнутыми») или размыкаются (если были «нормально замкнутыми»).

Также существует и вариант «перекидных» контактов, в которых присутствует центральный общий подпружиненный контакт, в нормальном положении замкнутый с одним из соседних, а при подаче тока перекидывающийся к другому. Когда же ток через обмотку снимается, то все возвращается в исходное состояние. Большинство типов реле содержит не одну, а несколько групп таких контактов, управляемых одним якорем. Это можно увидеть на рис. 7.2, где представлены некоторые типы реле.

Разумеется, вокруг этого базового принципа работы за много лет были накручены различные «прибамбасы»: так, существуют реле, которые при каждой подаче импульса тока перебрасываются в противоположное положение (пускатели), реле, контакт в которых может иметь три положения (трехпозиционные: замкнуто-нейтраль-замкнуто) и т. п., но мы не будем их рассматривать, потому что большинство функций таких специализированных реле давно выполняют логические микросхемы.

В обычных реле (кроме так называемых *поляризованных*) эффект не зависит от направления тока в обмотках — якорь при любом направлении тока в обмотке будет

к ней притягиваться, но все-таки некоторая разница в конструкции у реле, специально предназначенных для управления переменным током, имеется. В реле переменного тока вокруг сердечника размещают отдельный короткозамкнутый виток из медной пластинки, который снижает возможный дребезг слишком легкого якоря. Мы будем заниматься только реле постоянного тока — т. е. такими, обмотки которых работают от постоянного тока, хотя коммутировать они могут любой сигнал, — потому углубляться в этот вопрос не станем.

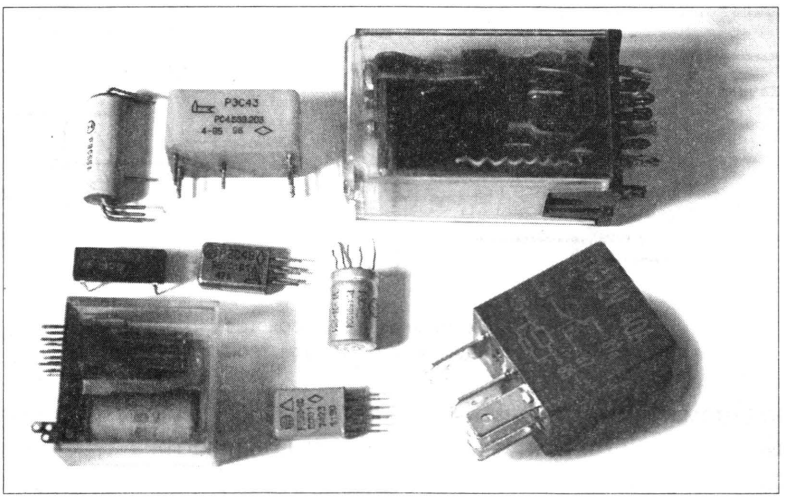


Рис. 7.2. Некоторые разновидности электромагнитных реле

В справочниках приводится либо величина тока через обмотку, либо величина рабочего напряжения, что равнозначно, потому что величина сопротивления обмотки тоже всегда приводится. Обычно конкретные типы реле имеют разновидности с разными сопротивлениями обмоток (это определяется так называемым *паспортом* реле).

В сравнении с оптоэлектронными или просто электронными реле, о которых мы будем говорить далее, электромагнитные имеют и преимущества, и недостатки. Электронные реле работают бесшумно, управляются гораздо меньшими токами и напряжениями, и в правильно выбранном режиме эксплуатации намного долговечнее электромеханических. Но, во-первых, в электронных реле, в которых коммутирующий элемент построен обычно на тиристорах, всегда присутствует довольно заметное падение напряжения на «контактах», которое приводит к невозможности пропуска малых сигналов через мощные реле. (В случае электромагнитных реле все ровно наоборот — чем мощнее реле, тем лучше, не редкость, когда в измерительных схемах для неискаженного пропуска малых токов и напряжений специально ставят реле или переключатели, рассчитанные на ток в десятки ампер.) Кроме того, в случае коммутации больших токов это падение напряжения ведет к разогреву реле, в результате чего приходится принимать специальные меры охлаждения. Во-вторых, если электронное реле предназначено для коммутации постоянного тока, то переменный через него уже пропускать нельзя. Существенно также, что элек-

тромеханические реле, особенно мощные, гораздо дешевле сравнимых по характеристикам электронных.

Главный недостаток электромагнитных реле в сравнении с полупроводниковыми устройствами — то, что энергетический порог, с которого начинается управление обмотками, весьма велик. Все же токи в 30–50 мА при напряжениях 5–30 В, т. е. мощности порядка ватта (и это для малогабаритных реле, а для реле покрупнее нужна еще большая мощность) — запредельны для современной электроники и являются слишком большой роскошью, если требуется всего только включить нагрузку в виде лампочки. А вот когда необходимо от маломощного сигнала включить, например, мощный нагреватель — тут по совокупности показателей стоимость/надежность электромагнитные реле оказываются вне конкуренции. В большинстве современных бытовых нагревательных приборов (в калориферах, электродуховках, хлебопечках и пр.) для включения/отключения мощного нагревателя применяют именно электромагнитные реле, которые вы можете распознать по характерным щелчкам во время работы.

Заметки на полях

Кстати, а как определить напряжение срабатывания незнакомого реле, если справочника нет под рукой? Это несложно, только надо иметь регулируемый источник питания. Найдите с помощью тестера выводы обмотки (она имеет обычно сопротивление от десятков ом до нескольких килоом, а если реле в прозрачном корпусе, то найти ее можно просто визуально) и подключите обмотку к источнику. Найдите нормально замкнутые контакты (прозвонкой) и подключите к ним тестер. Выведите источник на минимальное напряжение, включите его, а затем постепенно добавляйте напряжение. Вместо подключения тестера можно просто поднести реле к уху, но если оно малогабаритное и, тем более, герконовое, то щелчок при срабатывании можно и не услышать. Отметьте значение напряжения, когда реле сработает, а затем умножьте его на полтора — это и будет приблизительное значение номинального напряжения срабатывания.

Другим недостатком реле, как нагрузки для полупроводниковых приборов, является то, что его обмотка представляет собой индуктивность. Для постоянного тока это просто сопротивление, но в момент разрыва цепи, как показано в *главе 5*, на обмотке реле возникает импульс напряжения (по полярности он противоположен направлению изменения тока в обмотке). Если индуктивность обмотки велика, а ее собственное (активное) сопротивление мало, то импульс этот может вывести из строя коммутирующий прибор (например, транзистор) и в любом случае создаст сильные помехи остальным элементам схемы по шине питания. Поэтому при стандартном включении реле всегда рекомендуется включать параллельно его обмотке диод (даже если коммутация происходит не от полупроводниковых источников, а от таких же реле, — чтобы избежать искрения на контактах) в таком направлении, чтобы в статическом режиме, когда все успокоилось, диод этот тока не пропускал (см. рис. 7.1, *б*). Тогда выброс напряжения ограничивается уровнем напряжения на открытом диоде, т. е. 0,6 В. Напомним, что у MOSFET-транзисторов такой диод уже имеется в структуре (см. рис. 6.10, *б*).

Напомним также (см. *главу 5*), что управлять любыми электромагнитными реле непосредственно от выходов микроконтроллеров (или других многофункциональных

микросхем — например, счетчиков или регистров) нежелательно, даже если характеристики выводов формально это позволяют, — выбросы при коммутации могут вызвать сбои в работе этих микросхем. В таких случаях обязателен развязывающий ключ на обычном или полевом транзисторе, а иногда бывает необходимо и развязать питание, установив отдельный стабилизатор (или питая нагрузку от входного нестабилизированного напряжения).

Следует учитывать еще одну особенность электромагнитных реле. Ток (напряжение) срабатывания у них намного превышает ток (напряжение) отпускания — так, если в характеристиках указано, что номинальное напряжение реле составляет 27 В, то при этом напряжении гарантируется замыкание нормально разомкнутых до этого контактов и их удержание с заданной силой. Но совершенно необязательно выдерживать это напряжение длительное время — так, 27-вольтовые реле спокойно могут удерживать контакты в замкнутом состоянии вплоть до того момента, пока напряжение на их обмотке не снизится до 8–10 вольт. Подобный *гистерезис* — очень удобное свойство электромагнитных реле, которое позволяет избежать дребезга при срабатывании/отключении и даже сэкономить энергию при работе с ними. Например, на рис. 7.3, *а* приведена схема управления реле, которое в начальный момент времени подает на него нужное номинальное напряжение для срабатывания, а затем неограниченное время удерживает реле в сработавшем состоянии при пониженной величине тока через обмотку.

На рис. 7.3 также приведены еще две классические схемы. Схема на рис. 7.3, *б* называется *схемой самоблокировки* (после кратковременного нажатия кнопку «Пуск» отпускают, и реле останется замкнутым — самоблокируется) и очень часто применяется в управлении различными мощными устройствами — например, электродвигателями станков или насосов. Мощные реле-пускатели для таких целей (контакторы) имеют даже специальную отдельную пару маломощных контактов, предназначенную для осуществления самоблокировки. В этих случаях ток через стандартные кнопки «Пуск» и «Стоп» не превышает тока через обмотку пускателя (который составляет несколько десятков или сотен миллиампер), в то время как мощность разрываемой цепи может составлять многие киловатты, притом это может быть трехфазная цепь со всякими дополнительными неприятностями вроде огромных индуктивностей обмоток мощных двигателей.

Другая схема (рис. 7.3, *в*) скорее забавна и представляет собой дань прошлому, когда никакой электроники не существовало. Это схема простейшего электрического звонка, который может быть реализован на любом реле. Оно и само по себе при подключении по этой схеме задребезжит (правда, звук может быть самым разным, в зависимости от быстродействия и размеров реле, потому лучше употребить слово «зазуммерит»), но в обычном звонке якорь еще связывают со специальной тягой, которая в процессе работы стучит по металлической чашке, формируя звуковой сигнал. Есть и более простая конструкция электромеханического звонка, когда на обмотку реле просто подают переменное напряжение, от чего якорь вибрирует с его частотой (так устроены, например, звонки старинных телефонов с крутящимся диском), но нас тут интересует именно классическая схема, потому что в ней в чистом виде реализован другой основополагающий принцип электроники, так

или иначе присутствующий в любых генераторах колебаний, — *принцип положительной обратной связи*. Якорь в первый момент притягивается — питание размыкается — якорь отпускает — питание замыкается — якорь притягивается и т. д. Частота генерируемых колебаний зависит исключительно от механической инерции деталей реле.

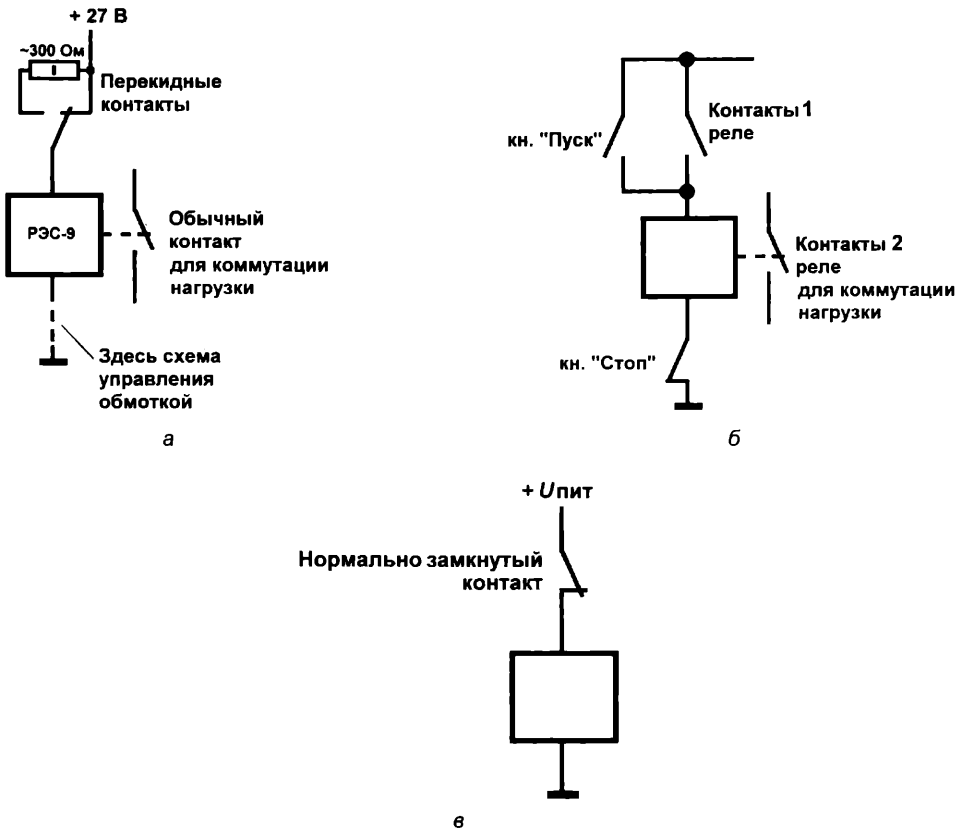


Рис. 7.3. Некоторые схемы включения реле: *а* — со снижением напряжения удержания; *б* — схема самоблокировки с кнопками «Пуск» и «Стоп»; *в* — схема классического электромеханического звонка

Стабилитроны

Стабилитрон представляет собой обычный диод с вольт-амперной характеристикой, подобной показанной на рис. 6.1, за одним исключением: при превышении некоторого обратного напряжения (индивидуального для каждого типа стабилитрона) он обратимо пробивается и начинает работать как очень малое сопротивление, при этом уровень напряжения сохраняется. Это можно представить себе, как если бы обычное прямое падение напряжения, составляющее 0,6 В, увеличилось вдруг до большой величины. Стоит только снизить напряжение ниже оговоренного — стабилитрон опять запирается и больше не участвует в работе схемы. Напряжения

стабилизации могут быть самыми разными — от 2 до 300 В. Учтите, что тепловая мощность, равная произведению тока через стабилитрон на его напряжение стабилизации, выделяется на нем самом, поэтому чем выше напряжение стабилизации, тем ниже допустимый ток. В характеристиках также указывается обычно минимально допустимое значение тока, при котором стабилитрон еще «держит» нужное напряжение.

Удобно использовать двусторонние стабилитроны (которые представляют собой два обычных, соединенных анодами), для того, чтобы и в положительном, и в отрицательном направлении включения характеристики были бы симметричны. Вольт-амперная характеристика такого двустороннего стабилитрона (типа КС170) показана на рис. 7.4. Отметьте, что характеристика в области пробоя все же имеет некоторый наклон, — т. е. при возрастании тока через прибор напряжение на нем не остается строго постоянным, а растет (это называется *дифференциальным сопротивлением*). К тому же напряжение стабилизации меняется с температурой.

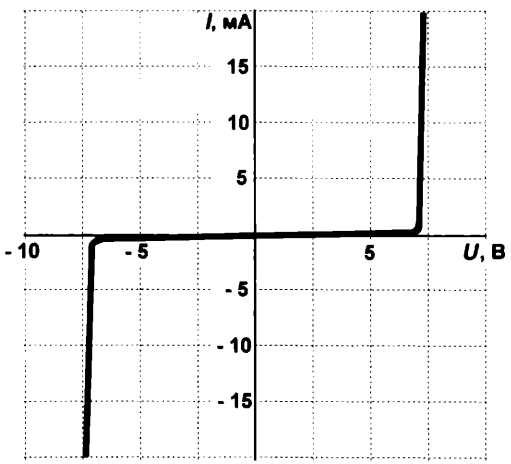


Рис. 7.4. Вольт-амперная характеристика двустороннего стабилитрона

Кстати, простейший стабилитрон — это обычный диод, включенный в прямом направлении, и их часто употребляют в таком качестве. Напряжение стабилизации составит при этом, естественно, 0,6 В (для его увеличения можно включить последовательно два и более диодов). Как видно из вольт-амперной характеристики диода (см. рис. 6.1), стабильность пресловутого напряжения 0,6 В оставляет желать лучшего (зависит и от тока, и от температуры), но во многих случаях особой стабильности и не требуется.

На рис. 7.5 приведена схема ограничителя напряжения на двух диодах (если требуется более высокое напряжение ограничения, их можно заменить на стабилитроны или на один двусторонний стабилитрон). Эту схему удобно применять, например, для защиты высокоомного входа микрофонного усилителя — нормальное напряжение с микрофона составляет несколько десятков милливольт, и диоды никак не влияют на работу схемы, поскольку таким маленьким напряжением не открываются. Но если микрофон присоединен через длинный кабель, то на входе могут созда-

ваться помехи от промышленного оборудования, от поднесенного к неподключенному входу пальца, или, скажем, от грозовых разрядов, которые сильно превышают указанные милливольты и могут вывести из строя чувствительные входные каскады усилителя. В приведенной схеме такие помехи любой полярности замыкаются через диоды, и входное напряжение не может превысить 0,6–0,7 В ни при каких условиях.

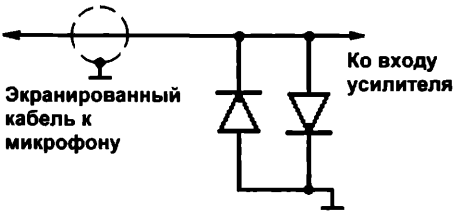


Рис. 7.5. Схема для защиты входа микрофонного усилителя

У внимательного читателя может возникнуть вопрос — ведь согласно вольтамперной характеристике и стабилитрона, и диода ток при превышении соответствующего напряжения растет очень быстро, так не сгорят ли эти входные диоды при наличии высоковольтной помехи? Ответ прост — энергия помехи обычно очень мала, поэтому ток хоть и может быть достаточно велик, но действует на протяжении очень короткого промежутка времени, а такое воздействие и диоды, и стабилитроны выдерживают без последствий.

Стабилитроны в чистом виде хороши в качестве ограничителей напряжения, а для формирования действительно стабильного напряжения (например, опорного для АЦП и ЦАП) следует применять специальные меры для стабилизации тока через стабилитрон и одновременно обращать внимание на стабильность его температурных характеристик. Хотя и существуют специальные прецизионные стабилитроны, но все же если вам нужен действительно качественный результат, то лучше применять *интегральные стабилизаторы*, которые дают на выходе гораздо более стабильное напряжение. Например, интегральный стабилизатор типа MAX873, который в диапазоне 4–30 В на входе дает на выходе ровно 2,5 В, обладает еще и весьма высокой стабильностью, — если даже положить на него паяльник (тем самым нагрев его градусов до 200), то напряжение на выходе этого стабилизатора и не шелохнется. В современной интегральной технике обычно источники опорного напряжения встраивают прямо в нужные микросхемы, но часто предусматривают вход и внешнего такого источника, потому что вы всегда можете захотеть изобрести что-нибудь получше.

Оптоэлектроника и светодиоды

Очень многие физические процессы обратимы. Типичный пример — если пластинка кварца изгибается под действием электрического поля, то можно предположить, что принудительное изгибание пластинки приведет к возникновению зарядов на ее

концах. Так и происходит в действительности, и этот эффект лежит в основе устройства кварцевых резонаторов для реализации высокоточных генераторов частоты (см. *главу 16*). Не давало покоя физикам и одно из первых обнаруженных свойств полупроводникового *p-n*-перехода — зависимость его проводимости от освещения. Этот эффект немедленно стал широко использоваться в различных датчиках освещенности (фотосопротивлениях, фотодиодах, фототранзисторах), которые пришли на замену хоть и весьма чувствительным, но крайне неудобным для широкого применения вакуумным фотоэлементам и фотоумножителям. Затем вырос целый класс устройств — *оптоэлектронные приборы*.

Заметки на полях

Кстати, любой полупроводниковый диод в стеклянном корпусе является неплохим датчиком освещенности, поскольку его обратный ток сильно зависит от наличия света, — особенно этим отличаются старые германиевые диоды (например, Д2 или Д9). Можете попробовать поэкспериментировать, только не забывайте, что, во-первых, сам этот ток очень мал (обратное сопротивление диода весьма велико), что потребует хороших высокоомных усилителей, а во-вторых, от температуры этот обратный ток зависит еще больше, чем от света.

Оптоэлектроника

В оптоэлектронных приборах (оптронах) через светодиод (обычно инфракрасный, о них мы поговорим позже) пропускается зажигающий его ток, в результате чего в воспринимающем *p-n*-переходе фотодиода или фототранзистора ток резко возрастает. Между входным светодиодом и выходом при этом имеется прозрачная изолирующая прокладка, которая позволяет гальванически развязать выводы входа и выхода.

Самый простой вариант такого прибора — *диодная оптопара* (рис. 7.6), которая обычно служит для электрически изолированной передачи линейных сигналов (например, звуковых колебаний или уровней постоянного тока в регулирующих устройствах). В ней обратный ток ($I_{вых}$) приемного диода линейно зависит от управляющего тока через светодиод ($I_{упр}$). Обратите внимание, что рабочая полярность у фотодиода обратная, чем у обычного диода, поэтому у таких компонентов, если они выпускаются в отдельном корпусе, плюсом помечен катод, а не анод.

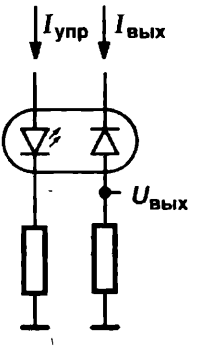


Рис. 7.6. Диодная оптопара

Один из главных параметров оптопар — коэффициент передачи по току $K_п$. Это величина, равная отношению выходного тока приемника оптопары (за вычетом темнового тока) при определенном напряжении на выходе, к входному току. Он характеризует чувствительность оптопары. Диодные оптопары (АОД101, АОД130), подобные показанной на рис. 7.6, имеют высокое быстродействие (типовое время нарастания сигнала — десятки наносекунд), но небольшой $K_п$, порядка единицы и даже меньше. Их основное назначение — преобразование линейных аналоговых сигналов. У транзисторных оптопар, в которых приемником служит фототранзистор, $K_п$ намного больше (порядка сотен), зато быстродействие гораздо ниже, типичная транзисторная оптопара (АОТ110, TLP521) может работать с прямоугольными сигналами на частотах не выше 10 кГц. При этом для обеспечения достаточного быстродействия входные токи таких оптопар должны составлять порядка 10 мА и более, а коллекторное сопротивление — не превышать сотен ом.

> **Совет**
>
> При выборе оптопар стоит также учесть, что многие отечественные оптопары имеют низкую стойкость изоляции (предельное напряжение 100–200 В), причем в справочниках приводятся противоречивые данные. Диодная оптопара АОД130 выдерживает не менее 1500 В между входом и выходом и заведомо годится для работы с сетевым напряжением, а вот популярная АОД101 с допустимым напряжением 100 В — увы, нет. Импортные оптопары имеют допустимое напряжение изоляции не менее единиц киловольт, т. е. подходят для работы с сетевым напряжением без оговорок.

Если требуется обеспечить гальваническую развязку при передаче прямоугольных импульсов с достаточным быстродействием, то можно воспользоваться более сложными по конструкции оптопарами. Так, одноканальные 6N135/136 или двухканальная HCPL-0560 (специально предназначенная для интерфейса RS-232) с диодом в качестве приемника и дополнительным усилительным транзистором обеспечат передачу импульсов до 1 МГц, более чувствительная 6N139 с дарлингтоновским транзистором — до 100 кГц. Специализированная оптопара TLP558 с логическим элементом на выходе может обеспечить скорость передачи до 6 МГц при входном токе всего 1,6 мА. Такие оптопары часто применяют для гальванической изоляции линий передачи данных. Например, медицинские приборы должны быть очень хорошо изолированы от потенциалов, связанных с электрической сетью, и прямое их подключение к компьютерным портам недопустимо, — тогда и применяют оптопары, которые обеспечивают их полную изоляцию.

Упомянутые ранее оптоэлектронные реле (часто их также называют *твердотельными*) устроены аналогично оптопарам, но с мощным фототиристором (о тиристорах см. *главу 10*) в качестве приемника: так, бесконтактное реле типа D24125 фирмы Crydom позволяет коммутировать сетевой переменный ток до 280 В при 125 А путем подачи напряжения 3–5 В при токе 3 мА (т. е. непосредственно от логической микросхемы) через управляющий светодиод. Мощностью 10 мВт напрямую управляют мощностью примерно в 35 кВт при полной гальванической развязке — ей-богу, совершенно беспрецедентный случай, обычным электромагнитным реле недоступный! Недостатком оптоэлектронных реле, как мы говорили, является большое сопротивление «контактов» — так, указанное реле D24125 приходится ставить на теплорассеивающий радиатор уже при коммутируемых токах порядка

5–8 А, что совершенно не требуется для обычных электромагнитных реле. Пример использования такого реле, управляемого непосредственно от микроконтроллера, мы увидим в *главе 21*.

Светодиоды

Набиравшая обороты космическая отрасль быстро сосредоточила усилия вокруг реализации другого эффекта — возможности генерации тока в полупроводниковом переходе под действием света, и картинка искусственного спутника Земли с широко раскинутыми темно-синими панелями солнечных батарей теперь стала уже традиционной. Но, может быть, таким же образом возможно генерировать свет, если подавать на *p-n*-переход напряжение? Оказалось, что можно, но это было реализовано далеко не сразу.

Первыми поддались инфракрасный (ИК) и красно-зеленый участки спектра. К началу 1980-х годов полупроводниковые *светодиоды* (LED, **L**ight **E**mission **D**iode), излучающие в ИК-диапазоне, уже стали широко использоваться в дистанционных пультах управления, а красненькие и зелененькие сигнальные светодиоды, вмонтированные в панели различных электро- и радиоустройств, хоть и были тогда еще куда тусклее традиционных лампочек накаливания, зато оказались намного более долговечны и потребляли принципиально меньше энергии.

В настоящее время в целом проблемы решены, и освоен фактически весь видимый спектр, включая синий и даже ультрафиолетовый диапазоны. Характерной особенностью цветных светодиодов является то, что они излучают свет одной (точнее, близкой к этой одной) длины волны, из-за чего насыщенность излучаемого света превосходит все чаяния художников. Существуют не менее двух десятков разновидностей светодиодов для разных длин волн, охватывающих все цвета видимого спектра (частично они приведены в табл. 7.1, где обозначения соответствуют продукции фирмы Kingbright и некоторых других фирм). Часто можно встретить в продаже и светодиоды белого свечения (они получаются из цветных, если покрыть излучающую часть кристалла люминофором, преобразующим оттенки света), которые все больше используются в качестве экономичных и долговечных источников света.

Таблица 7.1. *Некоторые разновидности светодиодов*

Длина волны, нм	Обозначение	Цвет свечения
700	H	красный
660	SR	красный
640	SU	красный
625	I (E)	чистый красный
610	N (SE)	оранжевый
590	Y (SY)	желтый
565	G (SG, MG)	зеленый

Таблица 7.1 (окончание)

Длина волны, нм	Обозначение	Цвет свечения
555	PG	чистый зеленый
465	MB	голубой
445	NB	голубой
430	PB	чистый синий
405	UV	фиолетовый*

* Часто продают под названием ультрафиолетовых, потому что этот тип LED действительно захватывает УФ-область, и может вызывать свечение многих типов люминесцентных красителей.

Подробности

Со времени предыдущего издания этой книги светодиодные лампочки, бывшие тогда еще запредельно дорогими, сумели заметно потеснить на прилавках уже успевшие стать привычными «энергосберегающие» люминесцентные. В сравнении с люминесцентными лампами светодиодные потребляют еще меньше энергии впустую (12-ваттная светодиодная лампа равносильна 20-ваттной люминесцентной или 100-ваттной лампе накаливания — проверено лично автором), не содержат ядовитых паров ртути и еще более долговечны. По стандарту LM-80, вместо среднего срока «перегорания», применимого к обычным лампам, для светодиодных светильников оценивается уровень снижения яркости свечения до 70% от начального. У качественных лампочек от известного «бренда» он не меньше 30 000 часов (т. е. около 4 лет), при этом правдивость этой цифры косвенно можно оценить по сроку гарантии производителя, которая должна составлять не менее 3 лет. Не верьте указанным на упаковке срокам работы в 50–65 тыс. часов при гарантии в 1–2 года — лампочка может продержаться и дольше, но светить к концу срока будет не ярче керосиновой.

Кстати, в отличие от осветительных, обычные светодиоды служат намного дольше — теоретически нет никаких препятствий для того, чтобы маломощный светодиод или светодиодный индикатор служил вечно (в многолетней практике автора встретился один-единственный случай необходимости замены семисегментного индикатора примерно через десять лет после его установки, больше отказов светодиодов не случалось). В светодиодах же, предназначенных для освещения, выделяющаяся мощность намного больше, чем у обычных, предназначенных для сигнализации: типичные рабочие токи маломощных светодиодов — единицы миллиампер, а осветительных светодиодов — 0,7 А и выше. Поэтому они в процессе работы сильно греются, и от этого полупроводниковый кристалл постепенно деградирует. Собственно, долговечность и «фирменность» светодиодных ламп определяется тем, насколько хорошо производитель сумел обеспечить охлаждение кристалла во всех вариантах условий эксплуатации. И по той же причине не существует (пока?) мощных светодиодных лампочек в достаточно миниатюрном корпусе, сравнимом по размерам хотя бы с «ксеноновыми» лампами накаливания, не говоря уж о галогенных. Все сказанное в этой главе далее касается только маломощных «сигнальных» светодиодов — мощные осветительные светодиоды требуют иного подхода и в этой книге подробно не рассматриваются.

Маломощные светодиоды делятся на обычные и повышенной яркости. Не следует сломя голову кидаться на повышенную яркость — в большинстве случаев она не нужна и только будет слепить глаза, если светодиод используется в качестве, скажем, индикатора наличия напряжения, причем регулировать такую яркость непро-

сто. Очень тщательно следует подходить и к выбору корпуса — матовый (диффузный) рассеиватель обеспечивает меньшую яркость, зато светящуюся полусферу видно под углом почти 180° во все стороны.

Со схемотехнической точки зрения все светодиоды, независимо от цвета свечения, представляют собой обычные диоды, за одним исключением — прямое падение напряжения на них превышает обычные для кремниевых *p-n*-переходов 0,6 В и составляет: для красных и инфракрасных 1,5–1,8 В, для желтых, зеленых и синих 2–3 В. В остальном их включение не отличается от включения обычных диодов в прямом направлении. То есть светодиод — это прибор, управляемый током (а не напряжением, как лампа накаливания), поэтому *обязательно должен иметь токоограничивающий резистор*. При питании осветительных светодиодных приборов в токоограничивающем резисторе впустую терялась бы слишком большая мощность, потому в них используют источники не напряжения, а стабилизированного тока.

Значение тока, при котором практически любой маломощный светодиод нормально светится, составляет 3–8 мА (хотя предельно допустимое может быть и 40 мА), на эту величину и следует рассчитывать схему управления светодиодами. При этом нужно учитывать, что яркость, воспринимаемая глазом, не зависит линейно от тока, — вы можете и не заметить разницу в свечении при токе 5 и 10 мА, а разница между 30 и 40 мА еще менее заметна. Светодиоды — одни из самых удобных электронных компонентов, т. к. один можно поменять на другой практически без ограничений и без необходимости пересчета схемы. Кстати, в схемах, где требуется экономия энергии, ее вполне можно обеспечить, применяя как раз суперъяркие светодиоды в режиме пониженной мощности: по опыту автора, нормальное свечение «суперъяркого» светодиода обеспечивает примерно в 50–100 раз меньший ток, чем для обычных.

Иногда токоограничивающий резистор встраивают прямо в светодиод (в этом случае яркость свечения уже управляется напряжением, как у обычной лампочки, а не током) — это обычная практика для «мигающих» светодиодов со встроенным генератором частоты. Обычное предельное напряжение для таких светодиодов составляет 12–15 В. В остальных случаях вопрос «каким напряжением питать светодиод», вообще говоря, не имеет смысла.

Светодиоды делают разной формы — обычно они круглые, но используются также плоские, квадратные и даже треугольные. Широкое распространение сейчас имеют двухцветные и даже трехцветные светодиоды. Двухцветные светодиоды бывают двух- и трехвыводными. С последними все понятно — это просто два разноцветных светодиода (зеленый и красный) с одним общим контактом в общем корпусе, управляющиеся раздельно. Подал ток на один — зажегся красный, на другой — зеленый, на оба — желтый, причем, манипулируя величиной токов, можно получить все промежуточные переходы. Но еще интереснее двухвыводной тип, который представляет собой два разноцветных светодиода, включенные встречно-параллельно. Поэтому в них цвет свечения зависит от полярности тока: в одну сторону красный, в другую — зеленый. Самое интересное получается, если подать на такой светодиод переменный ток — тогда он светится желтым!

Заметки на полях

Двухцветные светодиоды с тремя выводами (т. е. с раздельным управлением красным и зеленым) по какой-то неясной причине чаще всего поступают в продажу в прозрачном корпусе. Такой светодиод имеет небольшой угол рассеяния и сбоку почти не виден — прозрачные светодиоды ориентированы на применение в случаях, когда нужно сконцентрировать поток в пределах небольшого угла (для видимости издалека, например). В остальных случаях целесообразно использовать светодиоды с матовым диффузным рассеивателем.

Для того чтобы превратить прозрачный светодиод в матовый, его можно покрасить «молочным» лаком. Такой лак не стоит искать в продаже — проще сделать его самому на один раз. Для этого возьмите на самый кончик кисточки чуть-чуть белой краски на основе масляного связующего (например, художественные белила из школьного набора, подойдет и алкидная или пентафталевая белая эмаль) и интенсивно перемешайте ее в посуде небольших размеров, вроде рюмочки или пробки от шампанского, с 5–10 граммами бесцветного нитроцеллюлозного мебельного лака (НЦ-222, НЦ-218 и т. п.). Посуда должна быть стеклянной или полиэтиленовой (одноразовую посуду применять нельзя — она может расползтись). Окуните светодиод в этот лак и снимите отжатой кисточкой, которой производилось размешивание, образующуюся каплю (просто осторожно прикоснитесь к ней, и лишний лак перейдет на кисточку). Через час светодиод готов к установке на место.

Светодиодные индикаторы

Поскольку собственное падение напряжения на светодиодах невелико, их можно включать последовательно, чем пользуются производители цифровых сегментных индикаторов. Но тут дело осложняется тем, что отдельный светодиод представляет собой фактически точечный источник света, и нарисовать с его помощью длинную светящуюся полоску непросто даже при наличии рассеивающей свет пластмассы (причем, как ни парадоксально, чем мельче, тем хуже выглядят плоские светодиоды). Мелкие цифровые индикаторы (с высотой цифры до 12,7 мм) содержат по одному светодиоду в сегменте, а более крупные — по два и более. Это нужно учитывать при проектировании, т. к. семисегментный цифровой индикатор с высотой цифры более 12,7 мм имеет падение напряжения на каждом сегменте, превышающее 4 В, и управлять им от пятивольтового микроконтроллера напрямую затруднительно — номинальный запас в несколько десятых вольта легко «сожрется» собственным сопротивлением выхода контроллера, отчего ваш индикатор вообще может и не загореться. Перед проектированием схемы обязательно проверяйте величину прямого падения напряжения сегмента для выбранного типа индикатора!

Для таких случаев приходится идти на заведомые потери и питать индикаторы от повышенного напряжения через транзисторные ключи или специальные схемы управления индикаторами. Красота требует жертв! Набор семисегментных цифровых светодиодных индикаторов в четыре цифры в каком-нибудь мультиметре может потреблять до 100–200 мА тока — зато насколько он выглядит красивее по сравнению с почти ничего не потребляющими, но совершенно слепыми черно-белыми жидкокристаллическими панелями!

Семисегментные индикаторы (рис. 7.7, *слева*) бывают сдвоенными и строенными, встречаются и шестнадцатисегментные индикаторы, которые позволяют формиро-

вать буквы и специальные знаки. Такие индикаторы для удобства управления ими делают с общим анодом (тогда на индикатор подается общее питание, а зажигание сегментов производится коммутацией их к «земле» через токоограничивающие резисторы) или с общим катодом (сегменты имеют общую «землю», а зажигание производится подачей тока на каждый сегмент). Почти всегда выпускаются идентичные внешне типы и той и другой конфигурации. Для формирования длинных строк используют матричные индикаторы (рис. 7.7, *справа*), которые нередко встречаются в виде довольно больших дисплеев в несколько сотен точек, управляемых встроенным контроллером.

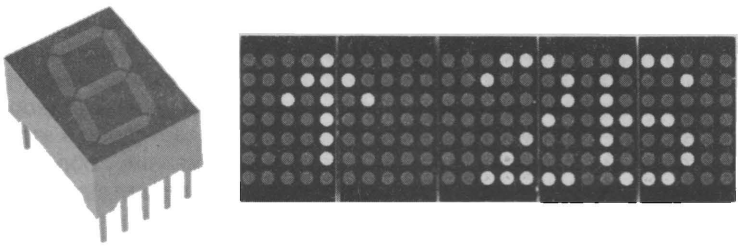


Рис. 7.7. Светодиодные индикаторы:
слева — семисегментный; *справа* — дисплей на основе матричного индикатора

В последние годы появились и сборные индикаторы на органических светодиодах (OLED), причем как символьные, так и графические (т. е. состоящие из матрицы точек). По внешнему виду они практически идентичны жидкокристаллическим (см. *разд. «ЖК-дисплеи»*), но сегменты или точки в них светятся сами. Отличаются от обычных LED они гораздо меньшим потреблением тока — например, двухстрочный дисплей по 16 символов в строке может потреблять около 40 мА при напряжении 3–5 В. Подробнее с такими дисплеями мы познакомимся в *главе 21*, когда будем проектировать метеостанцию на платформе Arduino.

ЖК-дисплеи

Жидкокристаллические (ЖК) индикаторы в бытовой аппаратуре встречаются чаще всего только в виде готовых ЖК-дисплеев для распространенных применений — например, для часов, магнитол, музыкальных центров или в виде многоразрядного набора цифр. Есть и матричные ЖК-дисплеи для формирования бегущей строки, многострочные — для текстовых сообщений и т. п., вплоть до полнофункциональных цветных ЖК-матриц, — тех, что используются в качестве экранов большинства современных массовых устройств: от мобильных телефонов до широкоэкранных телевизионных панелей.

Все ЖК-дисплеи сами по себе отличаются практически нулевым потреблением энергии в статическом режиме, энергия уходит только на переключение ЖК-ячейки. Правда, большинство матричных ЖК-дисплеев, предназначенных для демонстрации произвольных изображений, не могут обойтись без подсветки, которая будет потреблять довольно много (так, в ноутбуках — более половины общего

потребления). Но нас здесь интересуют лишь обычные ЖК-дисплеи, используемые в качестве цифровых или цифробуквенных табло. Устройство ячейки такой простейшей (пассивной) матрицы или индикатора, вместо подсветки использующей зеркало, отражающее внешний свет, показано на рис. 7.8.

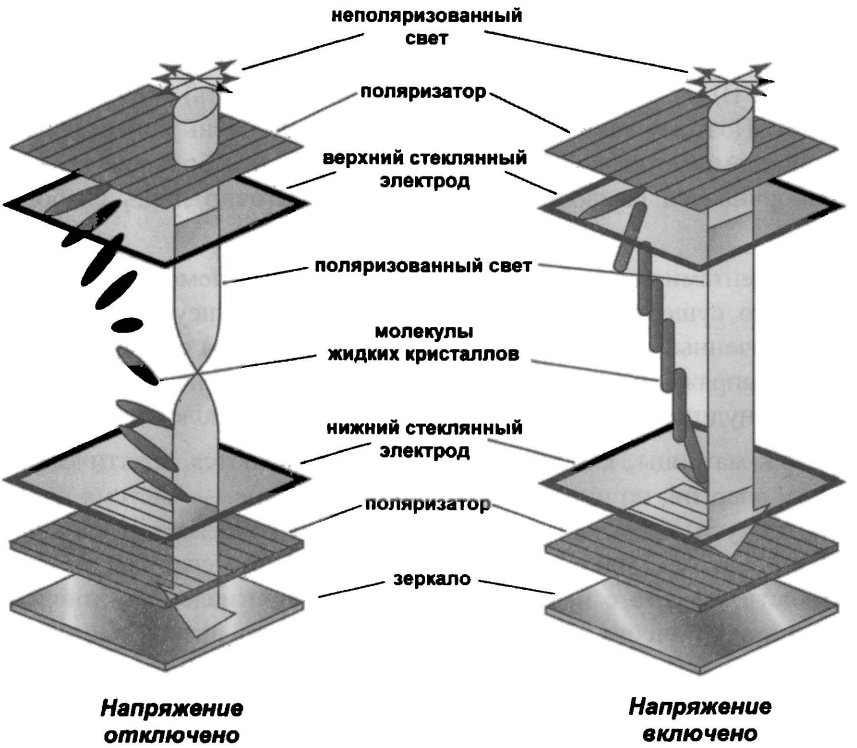


Рис. 7.8. Устройство пассивной ЖК-ячейки

Здесь слой жидких кристаллов толщиной несколько микрон находится между двумя стеклянными электродами, причем, за счет специальной структуры поверхности стекла, молекулы кристалла ориентированы параллельно плоскости этих электродов. Сверху и снизу такого «сэндвича» расположены пластины-поляризаторы, ориентированные перпендикулярно друг другу. Толщина слоя жидких кристаллов рассчитана так, что в исходном состоянии он поворачивает плоскость поляризации световой волны ровно на 90°. В результате в обесточенной ячейке (рис. 7.8, *слева*) свет беспрепятственно проходит через весь «пирог», отражается от зеркала (оно сделано матовым, чтобы не отражало окружающих предметов) и возвращается обратно. Подобная матрица в обесточенном состоянии выглядит, как обычная стеклянная пластинка.

Когда вы подаете на электроды напряжение (рис. 7.8, *справа*), электрическое поле ориентирует молекулы жидкого кристалла вдоль его силовых линий, т. е. перпендикулярно плоскости электродов. Жидкий кристалл теряет свои свойства и перестает поворачивать плоскость поляризации света. За счет перпендикулярной ориен-

тации поляризационных пластин весь «пирог» перестает пропускать свет. Образуется черная точка (или сегмент цифрового индикатора — в зависимости от конфигурации электродов). Это так называемая *TN-технология* (от **T**wisted **N**ematic — название разновидности жидких кристаллов).

Подобные монохромные ЖК-дисплеи всем хорошо знакомы по наручным и настольным часам, портативным измерительным приборам, дисплеям калькуляторов, плееров, магнитол. Величина напряжения сверх некоего, очень небольшого, предела (порядка 1–3 В) на «яркость» (точнее, на контрастность) ЖК-ячейки практически не влияет. Поэтому таким способом получаются очень контрастные, выразительные монохромные цифробуквенные индикаторы и небольшие табло, для приличной разборчивости символов на которых достаточно лишь слабой внешней засветки.

Управлять сегментами такого индикатора приходится с помощью разнополярного напряжения (это существенное, но не принципиальное неудобство), потому что однажды «засвеченный» сегмент может оставаться в таком состоянии часами даже после снятия напряжения с электродов, и возвращать в исходное состояние его приходится принудительно, подачей напряжения противоположной полярности.

Пассивные ЖК-матрицы, как уже говорилось, отличаются практически нулевым потреблением энергии, но имеют малое быстродействие, — система параллельных электродов, по сути, представляет собой отличный конденсатор, да еще и заполненный электролитом (жидкими кристаллами) как будто специально для увеличения его емкости. Вместе с неизбежно высоким сопротивлением тончайших прозрачных электродов ячейка образует отличный фильтр низкой частоты. Поэтому время реакции при подаче импульса напряжения — сотня-другая миллисекунд. Для цифровых индикаторов это не имеет никакого значения, но для компьютерных и телевизионных дисплеев с сотнями тысяч и миллионами ячеек это никуда не годится, потому там используют активные матрицы, содержащие усилительные тонкопленочные транзисторы (TFT).

Управляют ЖК-дисплеями обычно с помощью специальных микросхем-драйверов (с некоторыми из таких микросхем мы познакомимся в *главе 14*, а в *главе 20* рассмотрим подключение графического ЖК-дисплея со встроенным драйвером). Следует отметить, что применение ЖК-индикаторов, на взгляд автора, оправданно лишь в автономных устройствах, где важно низкое потребление.

В приборах, питающихся от сети, целесообразнее использовать светодиодные индикаторы — они значительно красивее и эргономичнее. Конечно, есть и ЖК-дисплеи с подсветкой, но тогда теряется их главное преимущество, заключающееся в малом потреблении, а в эстетике выигрыш получается невелик. Однако сформировать на светодиодах произвольное изображение (например, даже просто отобразить названия месяцев и дней недели в часах-календаре) гораздо сложнее, чем на ЖК-дисплее, конфигураций которых выпускается значительно больше. Выпуск упомянутых в *разд. «Светодиодные индикаторы»* OLED-панелей, аналогичных ЖК-дисплеям по конфигурации и управлению, значительно облегчил эту задачу.

Простейший уровнемер для водяных баков

Эта простая конструкция здесь приводится как пример того, что для создания весьма полезных устройств часто не требуется сложной электроники. За простоту схемы придется заплатить относительной сложностью ее изготовления и настройки. Но дело того стоит — описываемая конструкция содержит минимум деталей, имеет высокую надежность и информативность. Особенно удобно, что такой уровнемер позволяет его устанавливать на баке любой конфигурации — в том числе плоском, где максимальный перепад уровней незначителен.

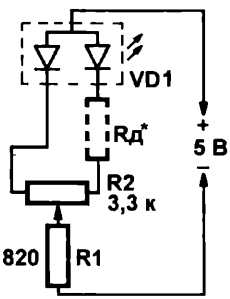


Рис. 7.9. Электрическая схема уровнемера

Электрическая схема уровнемера предельно проста (рис. 7.9). Главная ее деталь — переменное сопротивление, включенное по схеме потенциометра. Оно управляет перераспределением тока через плечи двухцветного светодиода. При нахождении ползунка в крайних положениях горит практически только один из светодиодов (либо красный, либо зеленый), а в промежуточном состоянии цвет свечения меняется от красного к зеленому через оттенки розового и желтого.

Двухцветный красно-зеленый светодиод VD1 трехвыводной, любого типа (автор использовал большой 10-миллиметровый L-819EGW фирмы Kingbright). Общим выводом он присоединен к источнику питания, двумя остальными — к крайним выводам потенциометра R2. Ползунок потенциометра соединяется с другим выводом источника питания через резистор R1, ограничивающий ток при нахождении ползунка в крайних положениях. Резистор Rд, показанный на схеме пунктиром, необязательный и служит для выравнивания яркостей, если одно из плечей (обычно — красное) горит сильнее другого, и изменение цвета при вращении ползунка происходит неравномерно. Номиналы резисторов R1 и переменного R2 должны находиться примерно в соотношении R1/R2 = $^1/_4$–$^1/_5$. При этом яркость в крайних положениях ползунка максимальна, а к середине диапазона, когда светятся оба светодиода примерно одинаково, падает, но зрительно это почти незаметно — меняется лишь оттенок свечения.

Такая полярность подключения источника питания, какая показана на рис. 7.9, годится для светодиода с общим анодом (положительным выводом), а для светодиода с общим катодом (например, BL-L109EGW фирмы Betlux) она, соответственно, будет противоположной. Сам источник питания может быть любым низковольтным (вплоть до батареек), напряжением от 5 до 18 вольт. От напряжения ис-

точника зависит подбор номиналов резисторов — такие номиналы, как на схеме, годятся для питающего напряжения 5 В, при 12 вольтах их надо увеличить примерно втрое-вчетверо. Автор использовал стабилизированный источник, встроенный в вилку, разобрав его и поместив в общий с постоянными резисторами и светодиодом корпус. При использовании батареек можно обойтись тремя штуками типоразмера АА (не меняя номиналов, указанных на схеме), только тогда целесообразно встроить в схему выключатель или кнопку, — при непрерывном горении щелочных АА-батареек хватит всего примерно на неделю.

Вся электрическая часть, исключая потенциометр, смонтирована в герметичном пластиковом корпусе, в котором делается отверстие, куда должен изнутри плотно входить светодиод (для надежности все стыки стоит проклеить сантехническим армированным скотчем, а место стыка светодиода с крышкой и отверстия для вывода проводов промазать герметиком). Потенциометр, который устанавливается на баке, может соединяться со схемой трехжильным гибким проводом в двойной изоляции (тем, на который обычно вешают подвесные светильники). Для надежности не стоит распаивать такой толстый провод непосредственно к потенциометру — лучше сделать промежуточную колодку с винтовыми соединениями.

Теперь о конструкции механической части, которая схематически показана на рис. 7.10. Ползунок переменного резистора жестко закреплен на одном валу с вращающимся шкивом, через который перекинута капроновая нитка с поплавком на одной стороне и грузом-противовесом на другой. Поплавок тоже подгружен дополнительным грузом так, чтобы сила натяжения нити на воздухе была достаточно большой, и нить не пыталась размотаться со шкива (вес этих грузов может составлять несколько сотен граммов). Поплавок делается из пенопласта, и разность между его весом в воздухе (вместе с весом груза) и весом груза-противовеса должна быть меньше плавучести поплавка (последняя по закону Архимеда равна в граммах объему вытесненной поплавком воды, выраженной в кубических сантиметрах). Грузы могут быть изготовлены из свинца, латуни, бронзы и даже алюминия — в общем, из любого материала, не подверженного воздействию воды.

Угол поворота подвижной системы φ у разных типов переменных резисторов несколько различается, и обычно находится в пределах 240–260° (если не найдете данных по справочнику, придется измерить). Длина окружности шкива должна быть такой, чтобы обеспечить полный поворот ползунка на эту величину при движении поплавка от самого дна до поверхности. То есть диаметр шкива рассчитывается по формуле: $(360/\varphi \times H)/\pi$, где φ — максимальный угол поворота подвижной системы потенциометра, H — максимальный перепад уровней. Диаметр шкива, измеренный по месту прилегания нити, должен быть подогнан под эту величину максимально точно.

Нить несколько раз оборачивается вокруг шкива, чтобы она по нему не скользила. Самое сложное в этой конструкции — смонтировать систему так, чтобы длина нити точно соответствовала перепаду уровней, и при нахождении поплавка в верхнем положении, движок потенциометра находился также в крайнем положении (соответствующем зеленому свечению светодиода). Автор делал это «насухую», разместив всю конструкцию со шкивом и поплавком на возвышении, точно соответствующем по высоте перепаду уровней воды в баке, и установив для удобства подгонки в одном из грузов винтовой зажим для нитки. Затем отлаженную и про-

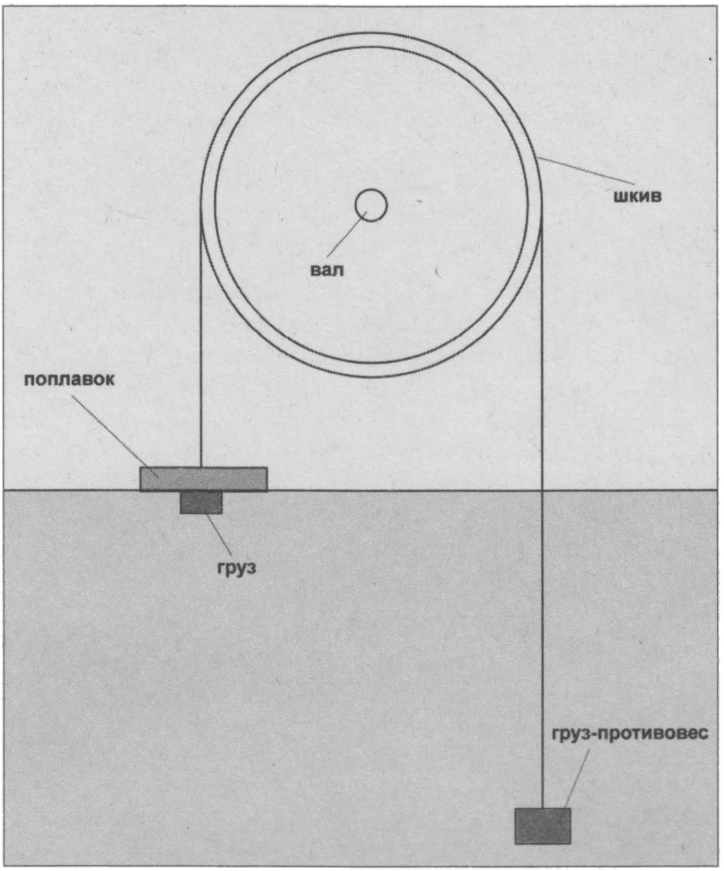
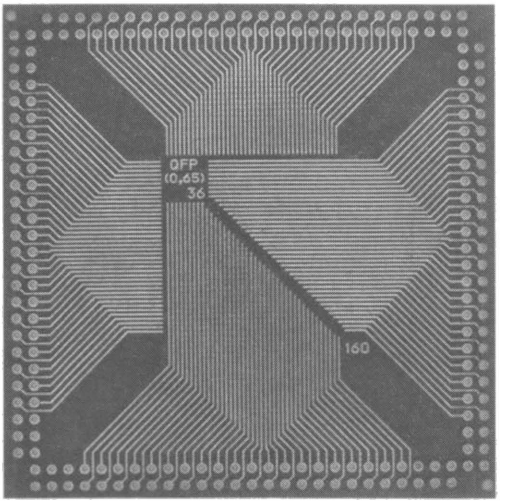


Рис. 7.10. Конструкция механической части уровнемера

веренную конструкцию аккуратно переносят, закрепляют на баке и проверяют с водой. Если над баком нет крыши, то всю конструкцию стоит накрыть отдельным колпаком для защиты от дождя и любопытных птиц.

Пара слов о подборе переменного резистора, от которого в значительной степени зависит надежность конструкции. Для наших целей категорически не подходят современные резисторы для аудиоаппаратуры, особенно с открытым резистивным слоем. Идеально для этой цели пригодны старинные отечественные герметичные резисторы типа СПО-1, но сегодня их можно разыскать только случайно на развалах или разобрав какой-нибудь старый прибор военной приемки. Если вы их не найдете, попробуйте разыскать отечественные герметизированные СП3-9а или аналогичные — они меньше по размерам, но тоже очень надежные. Для глубоких емкостей заманчиво попробовать приспособить многооборотные потенциометры, но автор так и не смог найти типа, достаточно надежного для работы в этой конструкции. Разумеется, характеристика резистора предпочтительно должна быть линейной (т. е. типа А), нелинейные (Б или В) дадут неравномерную зависимость цвета свечения от уровня. Если все указанные операции выполнить аккуратно, то уровнемеру какой-либо дополнительной подстройки не понадобится, и он будет надежно служить годами, не требуя никакого обслуживания.



ЧАСТЬ II

Аналоговые схемы

Глава 8. Звуковой усилитель без микросхем
Классическая схема УМЗЧ

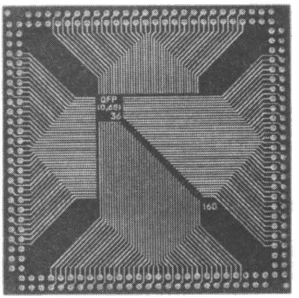
Глава 9. Правильное питание — залог здоровья
О питании электронных устройств

Глава 10. Глава 10. Тяжеловесы
Устройства для управления мощной нагрузкой

Глава 11. Слайсы, которые стали чипами
О микросхемах

Глава 12. Самые универсальные
Обратная связь и операционные усилители

Глава 13. Как измерить температуру?
Электронные термометры



ГЛАВА 8

Звуковой усилитель без микросхем

Классическая схема УМЗЧ

> Голос миледи, на редкость полнозвучный и проникнутый страстным воодушевлением, придавал грубоватым, неуклюжим стихам псалма магическую силу и такую выразительность, какую самые восторженные пуритане редко находили в пении своих братьев, хотя они и украшали его всем пылом своего воображения.
>
> *А. Дюма. «Три мушкетера»*

Казалось бы, конструировать звуковые усилители — в наше время дело мало кому нужное, чистое развлечение. Но это не совсем так — усилители в дешевых бытовых приборах качеством не отличаются, а у высококачественных мощных усилителей (класса Hi-Fi — от английского High Fidelity, что значит «высокая точность») цена может зашкаливать далеко за, как выражаются маркетологи, психологический барьер с тремя-четырьмя нулями на конце суммы в долларах. Это тот самый случай, когда собственная разработка не только греет душу, но и становится экономически целесообразной. Конечно, самостоятельно не соорудить 6-канальный усилитель с фазовой подстройкой объемного звука под расположение колонок (одни только затраты времени не оправдаются), но вполне работоспособный стереоусилитель можно сделать даже лучше фирменного.

Заметки на полях

Современные предварительные усилители (например, те, что встроены в типовые звуковые карты компьютеров) могут быть весьма совершенными даже при небольшой цене. Поэтому качество воспроизведения от них зависит в минимальной степени, тем более, что сам по себе цифровой звук передается без искажений и практически не зависит ни от плеера, ни от носителя, ни от количества перезаписей. Если вынести за скобки эффекты, связанные с потерями при сжатии (т. е. с цифровым форматом хранения звука), то в остальном качество воспроизведения в современных условиях будет полностью определяться оконечным усилителем и колонками. Из-за этого вопросы проектирования мощных усилителей и вообще аудиосистем не только не потеряли свое значение, но их важность даже возросла.

Однако, эта книга не посвящена аудиотехнике, поэтому мы разберем только базовую схему усилителя мощности звуковой частоты (УМЗЧ), на основе которой делается большинство более качественных конструкций. Эта схема, кроме того,

позволит нам познакомиться с некоторыми важными принципами построения электронных схем вообще.

Схема базового УМЗЧ

Рассмотрим схему на рис. 8.1. Это схема простейшего транзисторного УМЗЧ, стабилизированного обратной связью, с двухполярным питанием ±15 В. Усилитель охвачен отрицательной обратной связью (подробнее об обратных связях см. *главу 12*) с выхода на вход. По постоянному току эта обратная связь стопроцентная — т. к. ток через конденсатор C2 не течет, то резистор R4 спокойно можно считать «висящим в воздухе». Таким образом, выход с эмиттеров выходных мощных транзисторов VT4 и VT5 просто присоединен (через резистор R5) ко второму входу дифференциального входного усилителя и имеет практически одинаковый с ним потенциал. Из *главы 6* мы знаем, что все эмиттерные и базовые выводы дифференциального усилителя связаны между собой, поэтому на базовом выводе VT2 будет (при отсутствии сигнала) то же напряжение, что и на базе VT1. Последняя привязана к «земле» резистором R1 — т. е. имеет в состоянии покоя нулевой относительно «земли» потенциал. Получается, что выход усилителя (эмиттеры выходных мощных транзисторов) также привязан к этому же потенциалу, следовательно, на выходе в состоянии покоя будет практически нулевое напряжение, и через динамик ток не пойдет.

Ток покоя дифференциального каскада задается резистором R3 и равен примерно 2 мА (учтите, что потенциал соединенных эмиттеров чуть ниже потенциала баз). За

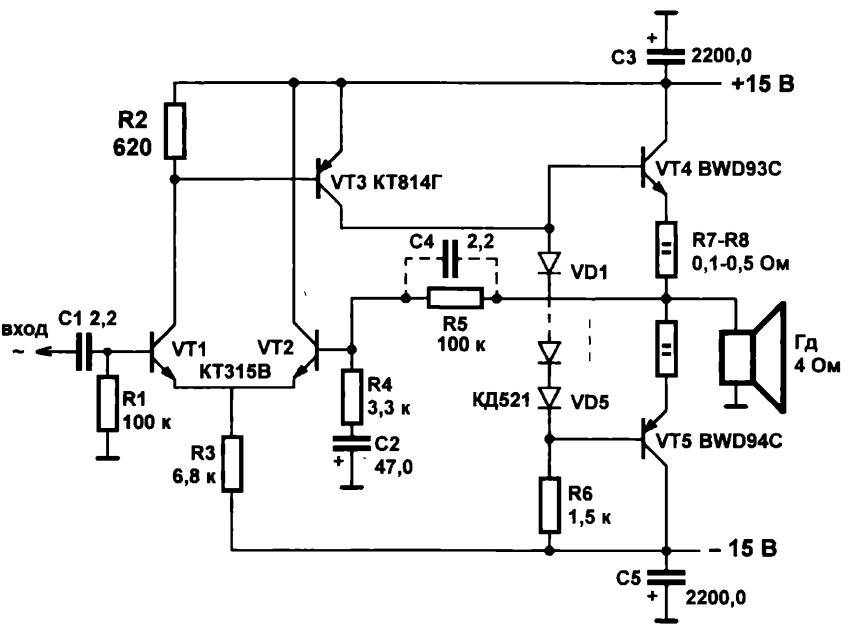


Рис. 8.1. Классическая схема усилителя звуковой частоты

счет того, что базы имеют одинаковый потенциал, равны и коллекторные токи VT1 и VT2, т. е. они составляют по 1 мА. Тогда на базе *p-n-p*-транзистора VT3, включенного по самой простой схеме усиления с общим эмиттером, за счет резистора R2 потенциал составит как раз величину падения на его переходе база-эмиттер, и он будет слегка приоткрыт, обеспечивая за счет тока в коллекторе потенциал на резисторе R6 такой, чтобы держать на эмиттерах комплементарных выходных транзисторов (одинаковых, но противоположной полярности: один *n-p-n*, другой *p-n-p*-типа), включенных по схеме с общим коллектором, потенциал, равный нулю. Смысл диодной цепочки между базами выходных транзисторов мы рассмотрим далее.

Подадим на вход какой-нибудь сигнал в виде переменного напряжения (конденсатор C1 нужен, чтобы не пропускать постоянную составляющую). Переменное напряжение пройдет через конденсатор C1 (ток перезарядки пойдет через резистор R1) и начнет менять потенциал на базе транзистора VT1 (а следовательно, и на его эмиттере). В результате изменится ток коллектора этого транзистора, отчего будет меняться падение напряжения на резисторе R2 и, соответственно, на базе *p-n-p*-транзистора VT3. Предположим, что в какой-то момент времени напряжение на входе возросло. Тогда ток через VT1 увеличится, на резисторе R2 напряжение также возрастет, транзистор VT3 приоткроется, ток его коллектора вырастет. Соответственно вырастет напряжение на резисторе R6 и на соединенных диодной цепочкой базах выходных транзисторов. При этом верхний (по схеме) выходной транзистор (VT4) приоткроется, а нижний (VT5) — призакроется, отчего напряжение на выходе также сместится в положительную сторону (поскольку пара выходных комплементарных транзисторов, в сущности, представляет собой усложненный эмиттерный повторитель, то сигнал она не инвертирует). При перемене полярности на входе все произойдет в обратную сторону.

Таким образом, входной сигнал передается на выход с неким коэффициентом усиления — как по току, так и по напряжению. Сразу возникает вопрос — а с каким? И еще вопросы — а зачем здесь нужна, во-первых, обратная связь, во-вторых, диоды в базовой цепи выходных транзисторов, и в-третьих, низкоомные резисторы (R7–R8) в эмиттерных цепях на выходе?

Давайте начнем с обратной связи. По постоянному току, как мы уже выяснили, обратная связь обеспечивает стабильность выходного напряжения покоя — ток через динамик не идет. Кроме того, эта обратная связь поддерживает в нужном режиме транзистор VT3, который при ее отсутствии, как мы видели в *главе 6*, находился бы в исключительно нестабильном состоянии. Что происходит при подаче переменного напряжения на вход? Учитывая, что конденсатор C2 в такой ситуации представляет собой малое сопротивление, часть выходного напряжения (обусловленная цепочкой R5–R4, т. е. в данном случае одна тридцатая его) подается обратно на вход дифференциального усилителя в противофазе к входному сигналу, вычитается из него и тем самым его уменьшает. Это могло бы показаться бессмысленной тратой ресурсов, но такое состояние вещей обеспечивает стабильность и предсказуемость схемы — фактически коэффициент усиления этого усилителя по напряжению на звуковых частотах определяется соотношением R5:R4, т. е. стабилен, практически независим от частоты (уже на частоте 10 Гц сопротивление конденсатора

С2 не превышает 330 Ом) и равен примерно 30. Добавим, что максимальный достижимый коэффициент усиления при рядовых используемых транзисторах составляет примерно 2–3 тысячи, но без обратной связи эта схема работала бы исключительно нестабильно, — скорее всего, динамик сгорел бы при первом же включении.

У таких схем есть одна нехорошая особенность — из-за собственных емкостей и индуктивностей участвующих в процессе компонентов (и переходов транзисторов, и резисторов, и проводников в макетном монтаже или на печатной плате) фаза обратной связи за счет задержек в элементах схемы на некоторых частотах (причем даже и много выше звуковых) может меняться с отрицательной на положительную. Отчего схема начинает «гудеть» при включении питания. «Гудение» это может и не восприниматься на слух, и вы даже не поймете, отчего вдруг выходные резисторы (R7–R8) чернеют и дымятся, а динамик выдает «чпок», после чего замолкает навсегда. Происходит следующее: малая наводка на вход вызывает сигнал на выходе, который передается опять на вход (базу VT2), но на этот раз не в виде отрицательной обратной связи, когда выходной сигнал вычитается из входного. За счет упомянутых задержек фаза получается такой, что выходной сигнал складывается с входным, и усилитель переходит в режим генерации.

Заметки на полях

Похожий, но имеющий другие физические причины, эффект может получиться, если вы подключите ко входу усилителя микрофон — звук от динамика попадает обратно в микрофон и усиливается, если он совпадает по фазе. Несомненно, вы не раз встречались с этим «микрофонным эффектом», если пытались наладить систему микрофон-усилитель в большом зале, и слышали в этом случае нарастающий свист. Для предотвращения микрофонного эффекта иногда достаточно бывает заслонить микрофон рукой или поместить его в поролон, или даже просто изменить полярность подключения динамиков на выходе.

Предотвращения этих явлений добиваются специальными схемотехническими мерами. Для ограничения коэффициента усиления по высокой частоте в цепь обратной связи включен конденсатор С4 (показан на схеме пунктиром), который ограничивает коэффициент передачи по цепи обратной связи для высоких частот, — чем его номинал больше, тем больше и ограничивает. Поскольку его емкость много меньше, чем конденсатора С2, то коэффициент передачи по цепи обратной связи на звуковых частотах получается более единицы, и усиление хоть и имеет место, но завал усиления на высоких частотах, обеспечиваемый С4 (чем он больше, тем ниже усиление с ростом частоты), предотвращает нежелательное «гудение» усилителя на высоких частотах.

Вторая обязательная мера — правильная разводка питания (см. также *главу 9*). Выходные мощные каскады усиления (спаренный эмиттерный повторитель на комплементарных транзисторах) должны питаться через отдельные достаточно толстые провода (сечением не менее 1 мм^2 — чем толще, тем лучше), соединенные прямо с источником питания, а входной дифференциальный каскад и «раскачивающий» транзистор VT3 должны быть также соединены с источником отдельными проводниками. В точках соединения проводника «+15 В» с коллектором VT2,

резистором R2, эмиттером VT3 (на плате или макете они должны быть физически как можно ближе друг к другу), как показано на схеме, должны быть установлены «развязывающие» конденсаторы большой емкости. В точке соединения проводников питания с резисторами R2 и R3 желательно также установить «развязывающие» керамические конденсаторы, соединенные с «землей» (на схеме они не показаны). Емкость этих конденсаторов может составлять 0,1–1 мкФ.

«Землю» также следует прокладывать как можно более толстыми проводниками с аналогичной разводкой. Как можно более толстыми делаются и выходные проводники к динамику. Все соединительные провода в схеме следует делать как можно короче, а вход должен соединяться с входным разъемом и регулирующим резистором экранированным проводом, экран которого будет «землей» входного сигнала.

Для того чтобы понять назначение диодов в базах транзисторов и резисторов в нагрузочной цепи, сначала попробуем ответить еще на два вопроса: какова мощность усилителя и какие меры нужно принять, чтобы обеспечить прохождение этой мощности через выходные транзисторы и правильно сконструировать усилитель?

Мощность усилителя

Мощность мы будем подсчитывать довольно примитивным способом, считая, что динамическая головка, индуктивность которой не слишком велика, имеет на всех интересующих нас частотах сопротивление, равное ее сопротивлению по постоянному току (в данном случае 4 Ом). Теоретически при полном размахе синусоидального напряжения на выходе усилителя амплитудное значение его может составить 15 В (30 В «от пика до пика»). На самом деле эта величина немного меньше, т. к. минимум два вольта теряется с каждой (положительной и отрицательной) стороны за счет падения напряжения на переходах выходных транзисторов VT4–VT5, на раскачивающем транзисторе VT3, на резисторах R7–R8 и т. п. Примем, что максимальная амплитуда на выходе может составить 13 В (при условии неискаженного синусоидального сигнала). Амплитудное значение связано с действующим значением известным нам из *главы 4* соотношением, т. е. оно составит в данном случае 13,5/1,41 = 9,2 В. Тогда действующее значение тока составит 9,2 В / 4 Ом = 2,3 А, а синусоидальная мощность достигнет 9,2 В · 2,3 А = 21 Вт. Следует подчеркнуть, что это максимальная возможная мощность, которую можно выжать из этого усилителя на нагрузке 4 Ом, — реальная может быть меньше.

Для того чтобы получить указанный размах напряжения на выходе, в соответствии со значением коэффициента усиления требуется входной сигнал не менее 0,5 В (амплитудного значения), поэтому, если вы подключите на вход стандартный микрофон, который обычно выдает не более единиц-десятков милливольт, такого размаха вы не получите, — потребуется еще микрофонный предусилитель. С другой стороны, подключение ко входу, например, выхода с диктофона или плеера вполне может вам обеспечить такой размах и даже более — фактическое выходное напряжение современных источников сигнала составляет не менее 2 В. Следовательно, все выходные компоненты нужно рассчитать так, чтобы они не сгорели при максимальной возможной мощности.

Прежде всего это касается динамической головки 4 Ом — это довольно стандартное сопротивление для динамиков, но если вы включите сюда головку 4ГД-4 (т. е. мощностью 4 Вт), то рискуете тем, что при максимальной громкости у вас ее диффузор вместе с толкателем просто улетят в потолок, даже не успев сгореть. Потому головка должна быть рассчитана на нужную мощность. В нашем случае необязательно, чтобы был запас по мощности, вполне достаточно колонки на 15 Вт, — в реальной музыке или речи максимальные мощности практически никогда не достигаются (подробнее об этом далее), а изредка появляющиеся экстремальные значения такая головка выдержит.

Куда сложнее обеспечить нормальный режим транзисторов. Сначала поговорим о выборе транзисторов выходного каскада. Ток коллектора «раскачивающего» каскада на VT3 равен примерно 10 мА в точке покоя (падение напряжения на резисторе R6 составляет около 15 В), следовательно, чтобы обеспечить 3,3 А на выходе и тем самым полностью использовать возможности источника питания, нужно иметь коэффициент $h_{21э}$ более 230 (именно поэтому выбраны транзисторы с «супербетой»). Есть и другие выходы из такого положения (в том числе позволяющие не терять целых два вольта от питания и при этом обеспечить меньшие искажения сигнала), — предложение которых, одно изящнее другого, стало своеобразным спортом в годы главенства дискретной аналоговой техники.

Стабильность

Теперь попробуем ответить на ранее заданный вопрос — зачем нужны диоды VD1–VD5 (целых пять штук!) между базами выходных транзисторов и резисторы R7–R8 между их эмиттерами?

Представьте себе, что диодов и резисторов этих не существует, и базы и эмиттеры комплементарных транзисторов просто соединены (рис. 8.2). Будет работать такая схема? Конечно, ведь если один из транзисторов открыт, то другой закрыт, а в промежутке они «перетягивают» друг друга (Хоровиц и Хилл, авторы основополагающего труда «Искусство схемотехники» [5], именно так и называют такой каскад: «push-pull», т. е. «тяни-толкай»). Но если на вход подать малый сигнал, то в пределах падения напряжений база-эмиттер создается мертвая зона, когда ни верхний, ни нижний транзистор не открыты, и оттого на выходном синусоидальном сигнале наличествует довольно большая (примерно в полтора вольта для обычных транзисторов и в три вольта для транзисторов с «супербетой») ступенька, что и показано на рис. 8.2.

Для нормального, без хрипов и искажений, воспроизведения звукового сигнала такое, естественно, недопустимо, и выходные транзисторы придется изначально слегка приоткрыть, — именно для этого и служит цепочка диодов между базами. Для обычных транзисторов достаточно трех диодов, для транзисторов с «супербетой» — пяти. Усилитель с таким режимом включения транзисторов еще называют *усилителем класса АВ* (см. далее). При токе около 10 мА, как на схеме, падение напряжения на цепочке диодов превысит падение напряжения между базами транзисторов примерно на полвольта, отчего транзисторы слегка приоткроются, и через

соединенные эмиттеры потечет небольшой ток (ток покоя). Теперь достаточно совсем малого сигнала, чтобы он повторился на выходе. Чтобы ток покоя меньше менялся с температурой, диоды следует приклеить или плотно прижать к тому же радиатору, что и транзисторы.

Для достижения наилучшего эффекта можно заменить диоды подстроечным резистором (или добавить его к ним, параллельно или последовательно) и, изменяя его сопротивление, обеспечить нужный ток покоя более точно (для схемы на рис. 8.1 это порядка 50 мА). Подстроечный резистор (рис. 8.3) нужно вращать очень аккуратно, при включенном в эмиттерную цепь нагрузки амперметре, чтобы не превысить ток покоя и не сжечь транзисторы. Еще лучше, чем замена диодов резисторами, будет решение с маленьким подстроечным резистором (порядка 100–150 Ом), включенным последовательно с диодами (их тогда понадобится на одну штуку меньше, чем по схеме рис. 8.1). Иногда вместо диодов-резисторов сооружают источник тока на маломощном транзисторе, что надежнее.

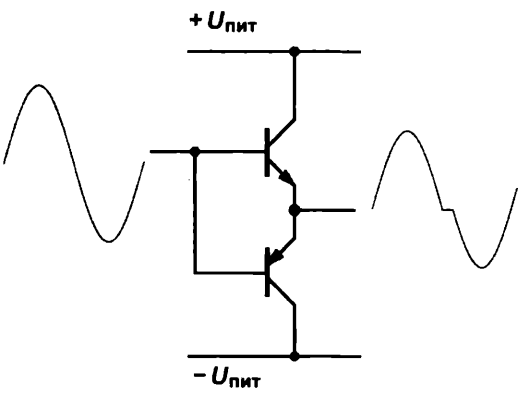


Рис. 8.2. Простейший каскад усиления по мощности на комплементарных транзисторах

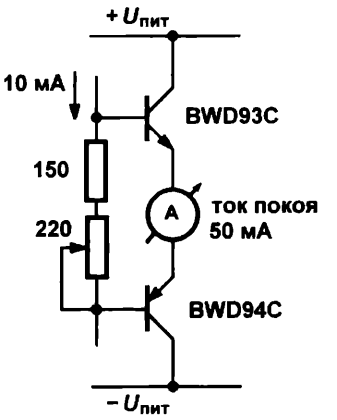


Рис. 8.3. Вариант замены диодов на резисторы для установки начального тока покоя

Крупный недостаток всей этой конструкции — то, что при случайном разрыве в базовой цепи транзисторов (например, нарушении контакта в подстроечнике) они оба распахнутся на «полную», и далее все будет происходить в соответствии с цитатой из повести писателя М. Анчарова: «Вы думали, что в аду воняет серой? Ничего подобного — в аду воняет горелой резиной». Потому употреблять подстроечник будет правильным только для макета, а для окончательного варианта нужно все же заменить его соответствующим постоянным резистором. Правда, в аналогичном случае у одного моего знакомого транзистор самостоятельно выпаялся из макета и упал на стол, отчего цепь разорвалась и — что самое удивительное — ничего даже не вышло из строя, в том числе и злополучный транзистор. Когда его, остывшего, посиневшего и полностью потерявшего внешний вид, впаяли обратно — все заработало!

И, наконец, зачем на рис. 8.1 показаны низкоомные резисторы R7–R8 в эмиттерной цепи (на рис. 8.2 и 8.3 они отсутствуют)? Они вносят некоторую долю стабилизирующей обратной связи в выходной каскад с целью лучшей стабилизации тока покоя, т. к. температурные коэффициенты диодов и эмиттерных переходов транзисторов, конечно, не равны в точности. Поскольку при токе 2 А на резисторе 0,5 Ом выделяется 2 Вт (подсчитайте сами!), то эти резисторы проще всего сделать самостоятельно из медной или нихромовой проволоки, как рассказано в *главе 2*. Чем выше номинал этих резисторов, тем выше стабильность схемы и тем лучше линейность сигнала, но тем выше и потери мощности.

О мощности выходных транзисторов

Теперь разберемся с мощностью на выходных транзисторах. Посчитать ее удобно, руководствуясь понятием КПД усилителя, который в режиме АВ (см. далее) лежит в пределах 60–70%. Почему это так, можно догадаться из следующих соображений: напряжение на транзисторах выходного каскада в каждый момент времени определяется разницей между напряжением источника питания (постоянное) и напряжением на нагрузке (меняется по синусоидальному закону). Более точный расчет приведен, например, в [6], а также далее в этой главе. Не углубляясь здесь в математику, мы можем принять, что мощность, которая выделяется на выходных транзисторах в указанной схеме включения, примерно равна 0,7 мощности в нагрузке, т. е. по 35% на каждый транзистор. Я же лично всегда просто принимал мощность на транзисторах равной выходной мощности и ни разу не ошибся. Такая прикидка может показаться слишком грубой, но поскольку в подобных расчетах нужно всегда закладываться на худшее, то можно считать, что на каждом из транзисторов будет выделяться по 10 Вт мощности.

Разумеется, ни один реальный транзистор этого не выдержит, потому их надо устанавливать на радиаторы, рассеивающие тепло. Причем учтите, что с корпусом транзистора, как правило, электрически соединен его коллектор, так что радиаторов должно быть два — или, как чаще делают, устанавливают один общий радиатор, но добавляют электроизолирующие теплопроводящие прокладки между корпусами-коллекторами и радиатором. О том, как рассчитывать радиаторы, мы поговорим в *главе 9*, когда будем рассматривать линейные источники питания.

Проверка и отладка

Проверить собранный усилитель очень просто. Для этого потребуется двухполярный источник питания на ±15 В (его мы также будем конструировать в *главе 9*). «Землю» можно подключить сразу к схеме, а провода питания — через сдвоенный тумблер, чтобы можно было включать питание одновременно. Сначала отключите динамик, чтобы его не спалить ненароком, вместо него нужно подключить мощное низкоомное сопротивление (необязательно 4 Ом, лучше даже побольше, порядка 10 Ом). Подключите все провода питания, затем присоедините к «земле» и выходу усилителя вольтметр и параллельно ему осциллограф, а вход (свободный вывод C1) временно соедините с «землей» перемычкой. После этого включите сначала блоки

питания (заранее проверив, что тумблер разомкнут) и установите на них по 15 В. Проверьте еще раз правильность подключения питания и, глядя на вольтметр, включите питание тумблером.

Внимание!

Перед внесением любых изменений в схему питание надо обязательно отключать!

Это тот момент, когда пожалеешь, что у человека только два глаза: нужно смотреть одновременно на вольтметр (в первый момент показания могут дернуться, но затем должно установиться близкое к нулю напряжение), осциллограф (не должно возникнуть генерации) и на блоки питания — не произошло ли короткое замыкание, и не отключились ли они от этого? Если что-то не так, отключите питание и начинайте думать, что именно подключено неправильно. Если же все в порядке, то приблизьте руку к сопротивлению нагрузки — и оно, и радиаторы выходных транзисторов, и сопротивления R7–R8 должны оставаться холодными. Проверьте вольтметром с помощью щупа напряжения на выводах остальных транзисторов — они должны быть примерно такими, как указано ранее в тексте. Если и тут все нормально, то можете себя поздравить — все собрано правильно, и можно приступать к следующему этапу.

Заметки на полях

А если не все нормально? Самое паршивое, если усилитель «загудит». Это будет видно на экране осциллографа — на выходе усилителя как будто появится сигнал, а выходные транзисторы и нагрузка начнут греться. Тогда попробуйте подпаять конденсатор С4, который указан на схеме пунктиром, проверьте, правильно ли установлены упомянутые ранее развязывающие конденсаторы по питанию, поставьте дополнительно параллельно электролитам С3 и С5 керамические конденсаторы и включите усилитель опять. «Гудение» должно либо пропасть, либо уменьшиться по амплитуде. Чтобы добиться полного пропадания эффекта, можно увеличить емкость конденсатора С4 и еще попробовать подпаять керамический конденсатор небольшой емкости между коллектором и базой транзистора VT3. Помните, что слишком длинные и тонкие соединительные провода в макете также не способствуют стабильности усилителя. В конце концов вы обязательно добьетесь того, что нежелательный эффект пропадет.

Следующий этап — при включенном питании и подключенном к выходу осциллографе коснитесь пальцем входа усилителя (свободного вывода конденсатора С1), предварительно отсоединив его от «земли». Если все в порядке, вы увидите на экране нечто, напоминающее синусоиду частотой 50 Гц, — это усилитель усиливает помеху, которая наводится вашим пальцем. Теперь можно подключать ко входу усилителя источник звуковых колебаний. Это не обязательно должен быть генератор (который мы сконструируем только в *главе 12*) — можно просто взять карманный плеер и снять сигнал с гнезда для наушников (естественно, придется приобрести соответствующий разъем и изготовить кабель для подключения). Один провод от плеера присоединяется к «земле», а второй — к входу усилителя. Выходной сигнал плеера при самых больших всплесках не должен превышать 0,5 В (заранее проверьте осциллографом и подстройте регулятором громкости, если необходимо!). При работающем усилителе вы увидите на экране осциллографа усиленный сигнал звуковой частоты. После этого уже можно вместо нагрузочного сопротивления

подключить динамик или колонки и наслаждаться музыкой. Громкость можно регулировать регулятором плеера или превратить в регулятор резистор R1, заменив его на переменный резистор, включенный по схеме потенциометра (см. рис. 5.1), — движком ко входу усилителя.

Классы усилителей или немного высшей математики

В зависимости от режима работы выходных транзисторов усилители (не только УМЗЧ — усилители мощности звуковой частоты) делятся на классы. Различают классы A, B, AB, C, D, G и H, недавно был также предложен еще один класс — T. Классы D и T относятся к дискретным (цифровым) усилителям, остальные — к линейным (аналоговым). О дискретном классе D, приобретающем все большее значение, мы немного поговорим в *главе 22*, а здесь рассмотрим особенности режимов работы выходного каскада для распространенных классов A, B и AB.

Действующее значение напряжения

Для того чтобы было более понятно рассмотрение важнейшего вопроса о КПД для различных классов, приведем сначала точное определение действующего значения напряжения переменного тока $U_д$ (именно действующее значение определяет мощность на нагрузке и выходных транзисторах, а следовательно, и КПД):

$$U_д = U_а \sqrt{\frac{1}{T}\int_0^T u^2(t)dt}. \qquad (1)$$

Здесь $U_а$ — амплитудное значение напряжения $u(t)$, T — период (в данном случае мы рассматриваем полпериода от 0 до π). Для синусоидального напряжения $u(t) = \sin(t)$:

$$U_д = U_а \sqrt{\frac{1}{\pi}\int_0^\pi \sin^2(t)dt} = U_а \sqrt{\frac{1}{\pi}\left(\frac{t}{2} - \frac{\sin 2t}{4}\right)\Big|_0^\pi} = \frac{U_а}{\sqrt{2}}. \qquad (2)$$

При выводе учитывается, что $\int \sin^2(t)dt = t/2 + \sin(2t)/4$. Соотношение (2) для действующего значения (без вывода) уже приводилось в *главе 4*.

Классификация усилителей

Класс A (рис. 8.4) — это фактически режим, соответствующий усилительному каскаду с общим эмиттером (см. рис. 6.6).

В классе A на коллекторе транзистора устанавливается ровно половина питания. Если считать переходную характеристику каскада строго линейной (сплошная линия на рис. 8.4), то амплитуда выходного сигнала может достигать напряжения питания. Для оценки КПД в этом идеализированном случае обратим внимание, что незатемненная область на графике выходного напряжения соответствует мгновен-

ным (в каждый момент времени) значениям напряжения на нагрузке, а затемненная — напряжениям на выходном транзисторе. Как мы видим из графика, эти области строго равны друг другу по площади, поэтому соответствующие интегралы (1) и действующие значения напряжения на нагрузке и на транзисторе будут равны, так что КПД окажется равен ровно 50% — половина затрачиваемой мощности выделяется на нагрузке, половина — на транзисторе. В реальности же переходная характеристика имеет S-образный вид (пунктир на рис. 8.4), поэтому во избежание искажений приходится ограничивать амплитуду сигнала, так что в действительности КПД может быть значительно меньше, да и реальный сигнал никогда не достигает максимальных амплитудных значений.

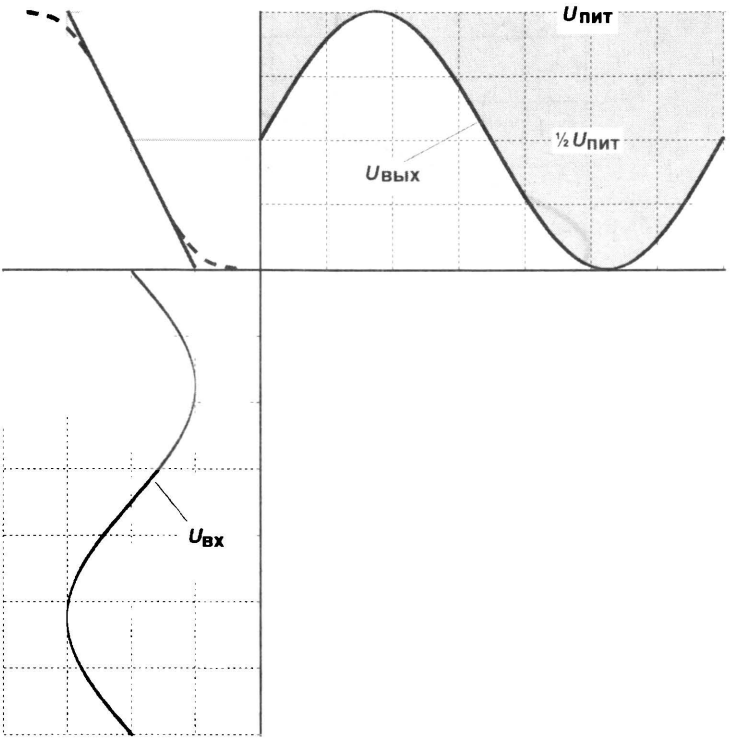


Рис. 8.4. Режим работы усилителя класса А

Другим крупнейшим недостатком класса А является то, что в отсутствие входного сигнала через транзистор течет большой ток (причем легко показать, что именно в отсутствие сигнала мощность, выделяющаяся на транзисторе, будет максимальной, и в этом случае КПД фактически равен нулю). Вместе с тем, режим класса А позволяет без лишних проблем получить неискаженный сигнал, усиленный как по току, так и по напряжению, и потому широко используется в маломощных каскадах, где КПД не имеет существенного значения. Например, в этом режиме работает «раскачивающий» каскад на транзисторе VT3 в схеме, приведенной на рис. 8.1.

Режим усилителя **класса В** фактически используется только в двухтактных схемах эмиттерных повторителей, подобных показанной на рис. 8.2. На рис. 8.5 представ-

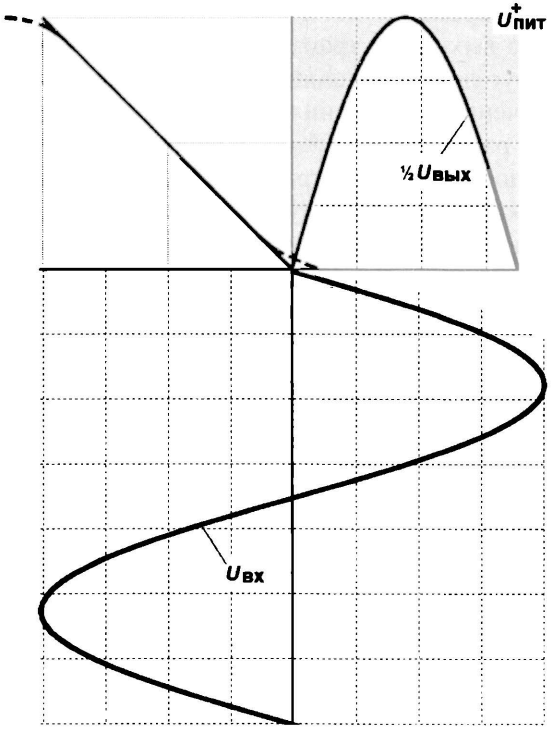


Рис. 8.5. Режим работы усилителя класса B

лены соответствующие графики для одной (положительной) половины такого каскада (для второй половины все — в случае идеального согласования характеристик выходных транзисторов — строго симметрично). Как мы видим, выходное напряжение представляет собой половину синусоиды, и в отсутствие входного сигнала ток через транзистор(ы) равен нулю. Примем, как и ранее для класса A, что переходная характеристика строго линейна, и попробуем оценить теоретический КПД.

Действующее значение напряжения на нагрузке равно, как следует из формулы (2), $U_н = U_а / \sqrt{2}$ (в общем случае $U_а \neq U_{пит}$), отсюда мощность в нагрузке будет равна:

$$N_н = \frac{U_а^{\,2}}{2R},$$

где R — сопротивление нагрузки.

Мгновенное значение напряжения на транзисторе можно определить как «остаток» от того, что выделяется на нагрузке (затемненная область на рис. 8.5), т. е. $u_т = U_{пит} - u_н(t)$. (Маленькими буквами мы здесь обозначаем мгновенные значения.) Ток через транзистор тот же самый, что и через нагрузку, и его величина будет равна $i_н = u_н(t)/R$. Тогда мгновенная мощность на транзисторе выразится формулой:

$$n_т = u_т i_н = \left[U_{пит} - u_н(t)\right]\frac{u_н(t)}{R}.$$

Средняя же мощность в одном плече определится следующей формулой (обратите внимание, что хотя мы считаем для одного плеча, осреднение происходит по полному значению периода 2π, просто в течение второго полупериода плечо не работает):

$$N_т = \frac{1}{2\pi R}\int_0^{\pi}[U_{пит} - u_н(t)]u_н(t)dt .$$

Для синусоидального напряжения подставим $u_н(t) = U_а \sin(t)$, $\int\sin(t)dt = -\cos(t)$, а также выражение для $\int\sin^2(t)dt$ (см. ранее), и получим:

$$N_т = \frac{1}{R}\left(\frac{U_а U_{пит}}{\pi} - \frac{U_а{}^2}{4}\right).$$

Суммарная мощность, потребляемая от источника, будет равна сумме мощностей на обоих транзисторах и нагрузке, а КПД выразится формулой (величина сопротивления нагрузки R в числителе и знаменателе сокращается):

$$КПД = \frac{N_н}{(N_н + 2N_т)} = \frac{U_а{}^2}{2\left[\frac{U_а{}^2}{2} + 2\left(\frac{U_а U_{пит}}{\pi} - \frac{U_а{}^2}{4}\right)\right]} = \frac{\pi U_а{}^2}{4U_а U_{пит}} .$$

Учтем, что в нашем случае $U_{пит} = U_а$, и окончательно получим, что теоретический КПД для усилителя класса В составляет $\pi/4 = 0{,}785 = 78{,}5\%$. Практически же КПД будет существенно меньше по целому ряду причин. Первая причина — мы производили расчет для максимального значения сигнала, а в реальности, как и для класса А, сигналы достигают этой величины только изредка. На рис. 8.6 приведены

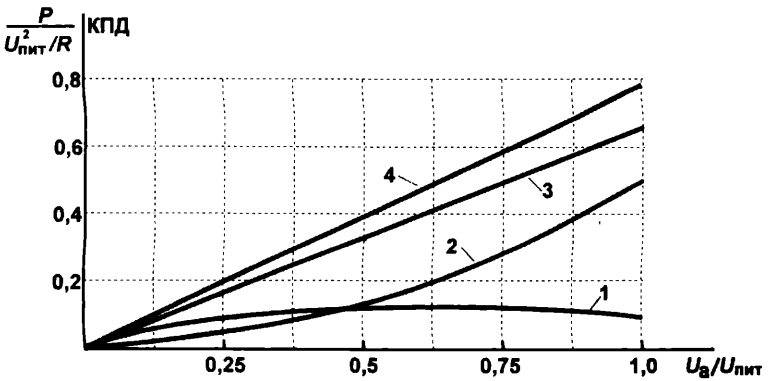


Рис. 8.6. Распределение мощностей и величина КПД в зависимости от относительной амплитуды выходного сигнала в усилителях класса В: 1 — мощность на каждом из транзисторов; 2 — мощность в нагрузке; 3 — суммарная мощность, потребляемая от источника; 4 — КПД

графики распределения мощностей и изменения КПД в зависимости от амплитуды сигнала.

Интересно, что в отсутствие сигнала КПД, как и для класса А, равен нулю, но есть одно существенное различие — сама мощность, потребляемая от источника питания, при этом также — в идеализированном случае — равна нулю.

Кроме этого, есть и другие причины снижения КПД. Прежде всего, переходная характеристика, как и для класса А, не является прямой линией — практически это выражается, в частности, и в том, что напряжение на выходе будет ограничиваться величинами, меньшими, чем напряжение питания. Напряжение на выходе будет всегда ниже входного по крайней мере на величину падения база-эмиттер. Все это само по себе уменьшит незатемненную область на графике и увеличит затемненную.

Главное же, что в чистом виде класс В для усиления аналоговых сигналов не используют — из-за больших искажений типа «ступенька» (см. *разд. «Стабильность»*). Поэтому практически классы В и А объединяют, создавая некоторый начальный ток смещения комплементарной пары, — такой режим усилителя известен, как **класс АВ**, и большинство аналоговых схем построено по этому принципу (на рис. 8.1 мы обеспечивали режим АВ с помощью цепочки диодов).

Все это касается случая, когда выходной каскад составлен из биполярных транзисторов. Применение комплементарных полевых транзисторов может существенно снизить искажения, однако там появится новая напасть — в момент равенства сигнала нулю оба транзистора пары будут приоткрыты (эффект «сквозного тока» — явление, аналогичное происходящему при переключении в КМОП-микросхемах), и это равносильно принудительному смещению биполярных транзисторов. В результате КПД «хороших» усилителей класса АВ составляет от силы 60%, а часто еще меньше. Именно поэтому при расчете радиаторов необходимо задаваться величиной мощности на выходных транзисторах, равной мощности в нагрузке.

О мощности и качестве звуковых усилителей

Рассчитанная для этого усилителя максимальная мощность на выходе составляет чуть больше 20 Вт. Много это или мало? Вообще-то, звуковой мощности в 20 Вт хватит, чтобы возмущенные соседи начали выламывать у вас дверь. Однако в этом деле есть один большой нюанс. Реальный музыкальный сигнал не является синусоидальным с определенной амплитудой. В нем встречаются как большие всплески, так и очень тихие звуки. Поэтому для качества усилителя играет роль не сама по себе его мощность, а динамический диапазон громкости звука, который он может воспроизвести без искажений. Попробуем подсчитать, каков он, скажем, для звука CD-качества. Сигнал аудио-CD имеет теоретический динамический диапазон в 16 двоичных разрядов, что составляет 65 536 градаций напряжения звукового сигнала (про двоичные разряды в применении к оцифрованным аналоговым сигналам мы будем подробно говорить в *главе 17*, посвященной АЦП и ЦАП). Предположим, что уровень шумов звуковоспроизводящего тракта у нас таков, что они вызывают

на выходе (в отсутствие входного сигнала) переменное напряжение с амплитудой в 1 мВ.

Тогда, чтобы даже самый тихий звуковой сигнал не потерялся на фоне этих шумов, максимально допустимая амплитуда на выходе (при условии передачи сигнала без искажений во всем диапазоне) должна составлять более 65 В, т. е. нам требуется усилитель мощностью примерно 400 Вт! При меньшей мощности, казалось бы, вы не получите нужного качества, потому что тихие звуки пропадут в шумах, а громкие «обрежутся», вызывая искажения. Но этот наш расчет слишком уж прямолинеен и слишком многого не учитывает: во-первых, редкие реальные записи имеют динамический диапазон 65 536 градаций (96 дБ) — большинство коммерческих записей компрессируются, а даже если исходный сигнал очень качественный, то искажение мощных редких выбросов сигнала на качестве не скажется. Во-вторых, есть и другие, не столь «тупые» методы повышения качества — например, снижение шумов.

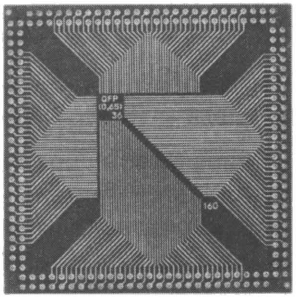
Вообще, качественная звукотехника — целое искусство, огромная и довольно специфическая область схемотехники (например, там и по сей день используют лампы!), поэтому дальнейшее углубление в эту увлекательную тему увело бы нас слишком далеко от основной линии книги, и интересующихся я отсылаю к другим источникам, коих великое множество, особенно в Интернете (см., к примеру, [7]). Среди бумажных изданий можно порекомендовать в первую очередь классический труд [4], который очень давно не переиздавался, по крайней мере на русском, но доступен в Сети, а из современных изданий — [9, 10].

Остановимся кратко еще на нескольких моментах, связанных с многочисленными мифами, сознательно распространяемыми производителями аппаратуры и автоматически, попугайски, тиражируемыми т. н. «меломанами». Среди этих затверженных стереотипов есть и разумные требования: например, коммутация мощных аналоговых сигналов ни в коем случае не может производиться электронными реле — только механическими контактами! Но вот искажения звукового воспроизводящего тракта в 0,5% и ниже при обычных среднего класса колонках и, что еще важнее, в 15–20-метровой комнате современных бетонных многоэтажек практически невозможно заметить на слух. Это связано с тем, что при таком прослушивании основную долю искажений вносит акустическая система — как сама, так и в комплексе с характеристиками помещения, где она расположена.

Преувеличено и влияние кабелей — в подавляющем большинстве случаев нет совершенно никакой нужды тратиться на кабели из «рафинированной» меди стоимостью 100 долларов за метр. Медь в обычных проводах и без того «рафинированная», а разница в электропроводности между особо чистой и обычной электротехнической — в лучшем случае доли процента. О «серебряных» и тем более «золотых» кабелях вообще умолчим. Особенно забавно существование последних — с учетом того, что электропроводность золота примерно на 30% хуже, чем у меди. А что касается серебра, то его 7–8% преимущества в этом случае тоже оказываются практически незаметны. Поэтому, если у вас занозой в печенке сидит представление о том, что на проводах от усилителя к колонкам теряется какая-то часть напряжения, приводя к искажениям, — просто возьмите провод потолще, и все. Но на

самом деле в бытовой аппаратуре мощностью до 50 Вт и этого совершенно не требуется, т. к. потери эти объективно настолько малы, что на субъективное восприятие могут оказывать еще меньшее влияние, чем упомянутые искажения из-за ограничений сигнала, — проще говоря, вообще никакого. Другое качество этих проводов — снижение их индуктивности за счет особого переплетения жил (так называемый *литцендрат*) — на звуковых частотах также не оказывает практически никакого влияния. Часто встречающееся возражение типа «но я же слышу, что с новыми кабелями стало лучше!» стоит отнести исключительно на счет психологического эффекта.

Самый смешной миф из джентльменского набора приемов для «разводки лохов» — направленность кабеля. Естественно, провод не может иметь никакой направленности — если, разумеется, разъемы у него на концах одинаковы. Абсолютно не имеет значения для звука и качество цифровых каналов — например, CD-читалка из карманного плеера справится со своей задачей ничуть не хуже, чем из навороченного музыкального центра, единственное условие при этом — чтобы при дальнейшей обработке цифровой сигнал не просачивался в звуковой тракт, что при грамотной — и ничуть не удорожающей систему — постановке дела есть вполне стандартное требование. Более подробно все эти вопросы освещены, например, в [8].



ГЛАВА 9

Правильное питание — залог здоровья

О питании электронных устройств

> Мы ежедневно просовываем ему через отдушину хлеб на вилах, а когда он требует, то и мясо, но — увы! — не хлеб и не мясо составляют главную его пищу.
>
> *А. Дюма. «Три мушкетера»*

О том, что трансформаторы вкупе с фильтрующими конденсаторами зачастую составляют основную часть массы и габаритов современных электронных устройств, известно всем. Реальных альтернатив обычным линейным трансформаторным источникам питания всего, в сущности, две (экзотику вроде солнечных батарей мы рассматривать не будем). Самую распространенную составляют электрохи мические источники тока (батареи и аккумуляторы), с которых мы и начнем. Об импульсных источниках питания, получающих все большее распространение, мы поговорим в конце главы.

Электрохимические элементы

Главное преимущество электрохимических (гальванических) элементов — мобильность, в чем им замены нет. Главный недостаток — они не обеспечивают долговременной эксплуатации для подавляющего большинства электронных устройств, за исключением специально спроектированных малопотребляющих либо редко используемых — таких, как наручные часы, пульты управления бытовой техникой или наши любимые мультиметры. В любом случае правильный выбор типа электрохимического источника — довольно важное дело.

Из всех электрохимических элементов для наших целей актуальнее всего щелочные (alkaline) *пальчиковые* батарейки. Вообще говоря, батарейками их называть неправильно — батарея, по определению, есть несколько элементов, соединенных в единый источник: так, батарейка типоразмера «Крона» — это действительно батарейка, а пальчиковая АА-типа — всего лишь элемент (о типоразмерах и характеристиках различных гальванических элементов см. *приложение 2*). Но в быту их принято

называть именно так, и мы тоже будем следовать традиции, употребляя вперемешку слова «элемент» и «батарейка».

Номинальное напряжение щелочных (alkaline) элементов — 1,5 В (у свежих элементов без нагрузки — 1,62 В). Для некоторых целей (например, в качестве резервных источников питания часов реального времени) в радиолюбительской практике используются литиевые батарейки-«монетки» с номинальным напряжением 3 В, но в качестве основных, кроме очень малопотребляющих устройств, их применять не рекомендуется из-за высокой стоимости. Литиевые аналоги мощных щелочных элементов типоразмеров C или D также появились в последние годы, но почему-то только с обычным «литиевым» напряжением 3,6 В, потому со щелочными элементами они не взаимозаменяемы, и рассчитывать на них не слишком удобно: они дороги, и продаются далеко не на каждом углу. А вот появившиеся не столь давно литиевые (точнее, литий-железо-дисульфидные) элементы типоразмеров AA и AAA со стандартным напряжением 1,5 В хотя пока и довольно дороги, но весьма неплохи, и мы их рассмотрим далее в сравнении со щелочными.

С точки зрения потребителя, основное отличие литиевых элементов от щелочных заключается в характере снижения напряжения по мере истощения: литиевые держат напряжение практически на номинальном уровне до последнего момента, после чего оно быстро падает до нуля (рис. 9.1). Литиевые элементы имеют исключительно низкий саморазряд (срок хранения в 10–12 лет для них типичен), высокую морозоустойчивость и могут быть рекомендованы для малопотребляющих или относительно редко включающихся устройств в жестких условиях эксплуатации. Следует также учесть, что из-за низкого внутреннего сопротивления литиевые лучше всего себя проявляют при работе на мощную или импульсную нагрузку. В таком режиме они покажут гораздо большее время работы, чем щелочные, и практически сравняются с последними по стоимости в расчете на каждый ватт-час, в то время как в низкопотребляющих приборах щелочные по емкости от них почти не отличаются, зато гораздо дешевле.

Важнейшей характеристикой электрохимических элементов является их *энергоемкость*. Для электрохимических источников ее традиционно измеряют в миллиам-

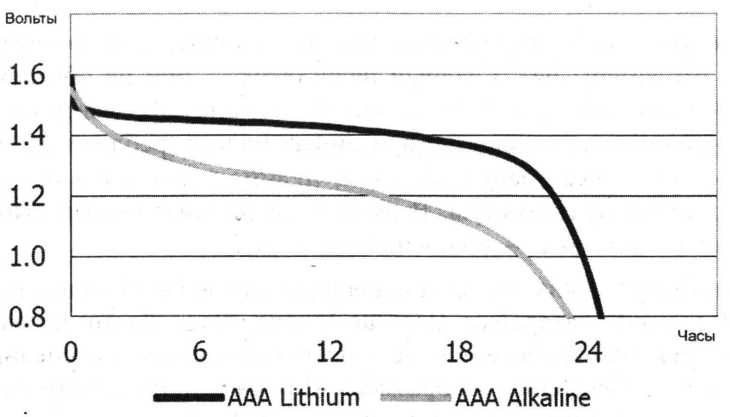


Рис. 9.1. Сравнительные разрядные характеристики литиевых и щелочных AAA-элементов при малых токах (по данным фирмы Energizer)

пер-часах (мА·ч). Величина энергоемкости, умноженная на напряжение элемента или батареи, даст энергию элемента в милливатт-часах, т. е. абсолютное количество энергии, запасенное в элементе (если дополнительно умножить эту величину на коэффициент 3,6, то получится энергия в привычных джоулях). Но в джоулях, милливатт-часах или ватт-часах для наших нужд энергию измерять неудобно, т. к. само напряжение элемента в процессе разряда меняется и существенно (см. графики на рис. 9.2 и 9.3, представляющие процесс разряда во времени). Зато

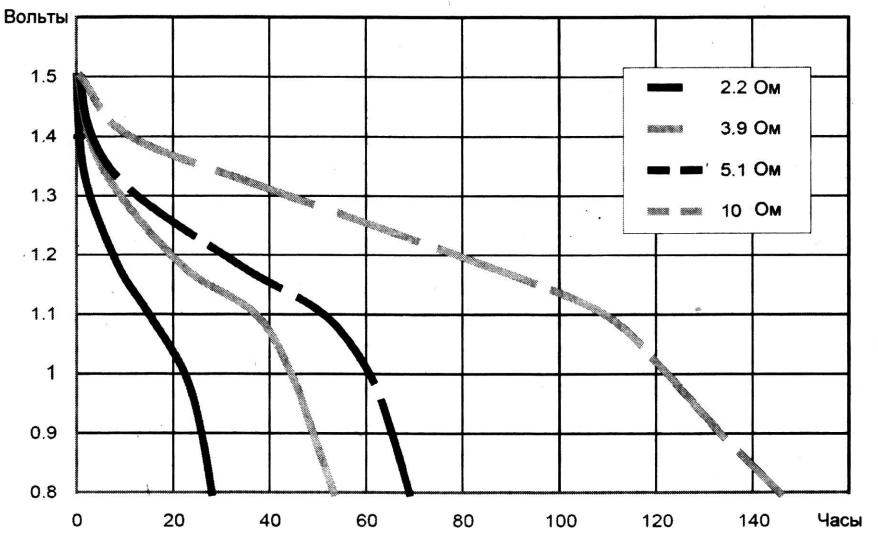


Рис. 9.2. Типовые разрядные кривые щелочного элемента типоразмера D при 20 °С и различных сопротивлениях нагрузки (по данным Duracell/Procter & Gamble)

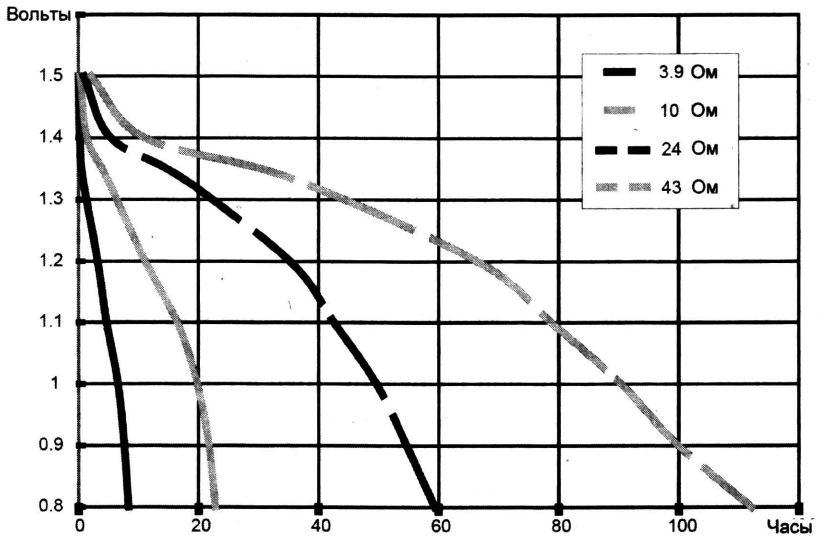


Рис. 9.3. Типовые разрядные кривые щелочного элемента типоразмера AA при 20 °С и различных сопротивлениях нагрузки (по данным Duracell/Procter & Gamble)

выраженная в миллиампер-часах энергоемкость легко поддается измерению и расчету — достаточно поделить эту величину на средний потребляемый устройством ток, и получится допустимое время работы устройства.

Некоторые типовые разрядные кривые для различных элементов и режимов показаны на рис. 9.2–9.3. Такие графики приводятся в документации, которую можно разыскать на сайтах производителей, и с их помощью уточнить энергоемкость. При необходимости подобные данные несложно получить и самостоятельно, замкнув элемент на нужное сопротивление в требуемых условиях и периодически замеряя напряжение. Для того чтобы получить из этих данных энергоемкость в мА·ч, следует поделить среднее за время разряда значение напряжения на нагрузку в омах и умножить на время. Так, для элемента АА при разряде до 0,9 В и нагрузке 43 Ом время разряда равно 100 часам, а среднее значение напряжения составит примерно 1,25 В, т. е. средний ток разряда будет около 30 мА. Итого энергоемкость при этих условиях приблизительно равна 3000 мА·ч. А вот при нагрузке 3,9 Ом (средний ток — примерно 320 мА) энергоемкость составит всего около 2200 мА·ч.

Ориентировочная удельная энергоемкость щелочных элементов — примерно 300 мА·ч на см^3. Таким образом, энергоемкость батареек типоразмера АА — около 2200–2500 мА·ч, типоразмера ААА — 1000–1200 мА·ч, примерно столько же дают пальчиковые NiMH-аккумуляторы тех же размеров (о них далее). Для щелочного элемента типоразмера D энергоемкость составит 15–18 А·ч, для типоразмера С — вполовину меньше. У аналогичных «обычных» батареек (их еще называют *солевыми*) — энергоемкость в три раза меньше, чем у щелочных.

Однако эти ориентировочные цифры очень приблизительные вследствие того, что энергоемкость щелочного элемента сильно зависит от условий разряда, — так, если при разрядном токе 0,1 А считать емкость за номинальную, то при разряде вдесятеро большим током (1 А) она может упасть в полтора-два раза (в зависимости от типа элемента), а при снижении тока до 1 мА, наоборот, возрастет на 30–50%. Самый выгодный режим разряда для щелочных элементов — прерывистый: если батарейке периодически давать «отдохнуть», то даже при больших токах ее емкость почти не снижается. Кроме того, многое зависит от допустимого конечного напряжения. Например, если схема допускает минимальное напряжение питания 2,7 В, что при питании от трех щелочных элементов означает конечное напряжение 0,9 В на каждый элемент, то емкость окажется почти на четверть выше, чем при допустимом конечном напряжении 3,3 В (по 1,1 В на элемент).

У литиевых изменение емкости в зависимости от условий гораздо меньше. Если для щелочных 9-вольтовых батареек типоразмера «Крона» энергоемкость составляет приблизительно 500–600 мА·ч при разрядном токе в пределах 100 мА и падает вдвое при токах 300–500 мА, то литиевый аналог (1604LC) имеет несколько бóльшую энергоемкость — 750 мА·ч при малых токах, и при этом она падает всего до 600 мА·ч при возрастании разрядного тока аж до 0,5–1 А. Надо также учитывать, что при снижении температуры до 0 °С энергоемкость щелочных элементов падает на величину от 25 до 50%, а вот литиевые тот же результат показывают только при минус 20 °С.

При этом для щелочных элементов напряжение в начале разряда при постоянной нагрузке очень быстро падает с начальных 1,5–1,6 В до 1,3–1,4 В, а затем снижается уже более плавно (для литиевых падение в процессе разряда меньше, зато, как мы говорили, в конце они разряжаются до нуля почти скачком). Для щелочных батареек типоразмера «Крона» напряжение в конце разряда составляет приблизительно 5–6 В. Внутреннее сопротивление щелочных батареек составляет в начале порядка 0,12–0,17 Ом (для «Кроны» — до 1,7 Ом) и быстро растет по мере разряда.

По этим сведениям вы можете прикинуть необходимый тип питающих элементов для вашей схемы. Следует добавить, что при включении электрохимических элементов последовательно их энергоемкости, выраженные в миллиампер-часах, естественно, не складываются, а остаются теми же (при этом их энергии, выраженные в ватт-часах, суммируются). А параллельное включение электрохимических элементов практикуется только в исключительных случаях, если нет другого выхода. Из-за разброса параметров по технологическим причинам они в этом случае заметную часть времени будут работать друг на друга, особенно в конце разряда. У полностью разряженных щелочных элементов даже возможна переполюсовка выводов (и такой режим опасен для сохранности устройства). Энергоемкость параллельно включенных элементов (естественно, одного типа и из одной партии) будет на четверть-треть меньше суммарной емкости тех же элементов по отдельности. Развязка таких элементов через диоды помогает обезопасить устройство от протечек электролита и деформации элементов при глубоком разряде, но зато вы будете терять драгоценные доли вольта падения на диодах (даже диоды Шоттки «съедают» не менее 0,3–0,4 В). В результате выигрыш окажется не настолько большим, чтобы отказаться от идеи просто поставить элемент побольше размером.

Аккумуляторы

У любых типов аккумуляторов, в отличие от одноразовых элементов, намного выше саморазряд при хранении, а в остальном характеристики современных пальчиковых NiMH-аккумуляторов практически такие же, как у щелочных одноразовых батареек, разве что номинальное напряжение несколько ниже — 1,3 В против 1,5 В у щелочных. Но давайте немного разберемся, какие вообще бывают аккумуляторы, ибо они существенно различаются по свойствам, и каждый тип оптимален для применения в своей области.

Аккумуляторы встречаются кислотные, щелочные, никель-кадмиевые (NiCd), никель-металлгидридные (NiMH), литий-ионные (Li-ion), и еще попадаются литий-полимерные (Li-pol). Кроме перечисленных, существует еще море разновидностей аккумуляторов (в теории любая электрохимическая реакция обратима и может использоваться как для выработки электрического тока, так и для откладывания его «про запас»), но на рынке доминируют именно эти типы.

Кислотные аккумуляторы правильнее называть *свинцово-кислотными* (Lead-Acid, СКА), но других кислотных, кроме как на основе свинца, в быту вы не встретите. Это, вероятно, самая древняя разновидность аккумуляторов — первый работоспособный СКА был создан аж в 1859 году. В начале XX века выяснилось, что именно

этот тип аккумуляторов неплохо подходит для того, чтобы крутить стартер автомобиля, и с тех пор их производят десятками миллионов.

Еще лет двадцать-тридцать назад автомобильные аккумуляторы были весьма капризными и даже несколько опасными для здоровья — конструкторы никак не могли справиться с выделением газов, сопровождающим процесс заряда. Из-за этого СКА приходилось делать негерметичными, а электролитом в них, между прочим, служит серная кислота, которую периодически требовалось доводить до нужной плотности дистиллированной водой, — занятие, мягко говоря, небезопасное. С тех пор СКА значительно облагородились, стали герметичными и необслуживаемыми, но в основе они все те же, что тридцать и пятьдесят лет назад. У них низкая удельная энергоемкость (30–50 Вт·ч/кг в самых лучших образцах), и они боятся глубокого разряда, отчего в процессе хранения их надо все время подзаряжать.

Зато у СКА высокая перегрузочная способность — стартерная батарея даже на морозе без особых усилий отдает ток в несколько сотен ампер, необходимый для того, чтобы прокрутить холодный двигатель. При этом СКА дешевы и относительно неплохо держат заряд — хороший автомобильный аккумулятор разряжается в среднем на 5% в месяц или на 50% за год. Именно этот тип аккумуляторов традиционно используется в источниках бесперебойного питания (ИБП). Так как там батареи пребывают в тепличных условиях (постоянно подзаряжаются), то срок службы аккумуляторов в ИБП может достигать 5–7 лет.

Заметки на полях

Заметим, что кроме обычных СКА, есть еще и усовершенствованные типы: так, например, аккумуляторы с гелевым электролитом можно эксплуатировать в любом положении. И гелевые, и т. н. AGM-аккумуляторы, отличаются от обычных свинцовых повышенным сроком службы, особенно в буферном режиме, когда они не подвергаются частому глубокому разряду (в источниках бесперебойного питания, например), — до 10–12 лет, в отличие от обычных свинцовых, у которых срок службы не превышает 3–4 лет. В этих случаях применять стоит именно AGM или гелевые, несмотря на то, что их стоимость примерно вдвое выше, — она окупается за счет более долгой эксплуатации и повышения надежности всей системы.

СКА заряжать довольно просто (они не очень боятся перезаряда), и автоматические зарядники для автомобильных СКА доступны каждому. В радиолюбительской практике герметизированные СКА можно рекомендовать для питания мощных устройств (например, содержащих электродвигатели), а также в качестве буферного элемента при питании от солнечных батарей.

Для **никель-кадмиевых** (NiCd) аккумуляторов также характерна высокая нагрузочная способность (хотя и поменьше, чем для СКА), но есть у них и три капитальных недостатка. Первый: относительно малая удельная энергоемкость (хотя и несколько большая, чем у СКА) — 45–60 Вт·ч/кг. Второй: нелюбовь к зарядке не «с нуля» — так называемый *эффект памяти*. Третий: высокий саморазряд — до 10% в первые сутки, потом около 10% в месяц.

Правильный режим зарядки NiCd-аккумуляторов — сначала полная разрядка (формально — до напряжения 1 В на элемент), а потом уже полная зарядка. Потому для NiCd-аккумуляторов рекомендуется вырабатывать заряд до полного «умирания» устройства — редкие зарядные устройства позволяют себе тратить время на пред-

варительную разрядку. «Фирменная» зарядка производится до достижения определенного напряжения с дополнительным контролем по температуре (так работают зарядники, например, к дорогому электроинструменту). Более простой способ — заряжать определенным током в течение конкретного времени. Это лишний аргумент для того, чтобы предварительно разряжать батарею, потому что иначе определить необходимое время затруднительно. Правда, и умеренной перезарядки NiCd-аккумуляторы боятся меньше, чем рассматриваемые далее NiMH.

NiCd-аккумуляторы традиционно используются там, где требуется высокая нагрузочная способность и большой кратковременный ток. В первую очередь это электроинструмент, снабжаются такими аккумуляторами и профессиональные ТВ-камеры, шахтерские фонари или мобильные радиостанции. Одно из крупных преимуществ NiCd — это единственный тип, который без последствий может храниться полностью разряженным.

Никель-металлгидридные (NiMH) — это все пальчиковые аккумуляторы, которые продаются в киосках. Номинальная емкость для элементов одного размера различается, и обычно написана на них большими цифрами. Когда-то эту нишу занимали NiCd (они еще выпускались с этикетками на белом фоне, чтобы отличить от батареек), но «зеленые» настояли, и теперь NiCd можно приобрести лишь в специализированных точках продаж. Конечно, дело не только в загрязнении окружающей среды — NiMH-аккумуляторы имеют бóльшую, чем NiCd, удельную емкость (60–120 Вт·ч/кг) и не склонны к «эффекту памяти», потому заряжать их можно не обязательно «с нуля». Зато они боятся глубокого разряда (хотя и не в такой степени, как СКА), и хранить их надо хотя бы частично заряженными. При этом они имеют самый высокий из всех типов саморазряд (вдвое больше, чем у NiCd) и страшно не любят перезарядки, потому что сильно нагреваются в конце процесса заряда (это, кстати, может служить одним из признаков того, что зарядку пора заканчивать). Типичные кривые зависимости напряжения от времени работы для таких аккумуляторов показаны на рис. 9.4.

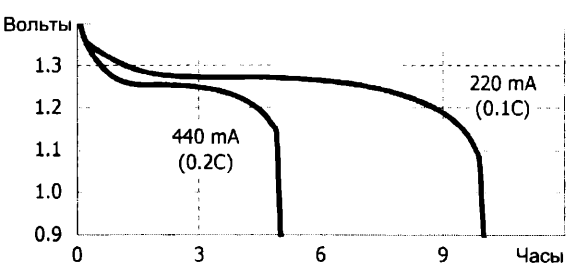


Рис. 9.4. Типовые разрядные кривые NiMH-аккумулятора типоразмера AA емкостью 2200 мА·ч при 20 °C (по данным Energizer Holdings, Inc.)

Как ни старались, но заставить NiMH отдавать бóльшой импульсный ток при перегрузках не удалось. Тем не менее, NiMH-элементы сейчас наиболее распространены среди универсальных аккумуляторов для бытовой электронной аппаратуры, исключая только такую, где зарядное устройство целесообразно встроить в сам прибор или «бесплатно» прикладывать к нему. Дело в том, что Li-ion-разновидность,

о которой пойдет речь далее, абы как заряжать решительно не рекомендуется, и лишь «фирменный» зарядник гарантирует, что все будет как надо.

Прежде всего, отметим главную и очень удобную черту **литий-ионных** (Li-ion) аккумуляторов — никакого «эффекта памяти» они не имеют, и вообще никакой профилактики (в виде специальной «тренировки» при хранении) не требуют. Но это мало помогает — Li-ion отличаются еще и тем, что портятся просто при хранении практически так же, как и во время эксплуатации. А вот будете ли вы их разряжать до конца или подзаряжать каждые полчаса — от этого почти ничего не зависит (допустимое число циклов заряд/разряд превышает 1000), причем частая дозарядка для этого типа даже предпочтительнее, т. к. хранить их полагается в заряженном виде.

Li-ion-аккумуляторы отличаются большой энергоемкостью (110–160 Вт·ч/кг) и малым саморазрядом — менее 10% в месяц, причем около трети этой величины обусловлено потреблением встроенных схем защиты. Схемы защиты нужны потому, что эти аккумуляторы совершенно не выносят перезаряда и при нарушении режима просто взрываются без предупреждения. Li-ion также плохо относятся к низким температурам. Все эти качества в совокупности и обусловили область применения Li-ion — для мобильных устройств с собственным зарядным устройством (сотовые телефоны, ноутбуки и т. п.), в последнее время ими также стали снабжаться мощные потребители: электроинструмент, электромобили и т. д. Использования же литий-ионных аккумуляторов в радиолюбительской практике мы здесь касаться не будем — это совершенно отдельная тема.

Литий-полимерные (Li-pol) аккумуляторы — разновидность Li-ion, которая отличается в худшую сторону тем, что совершенно не выносит низких температур (ниже 0 °С они отказываются работать) и имеет меньшую долговечность (100–200 циклов заряд/разряд). Зато они имеют «твердый» электролит, похожий на обычную пластиковую пленку, что позволяет делать батареи очень тонкими (до 1 мм), гибкими или имеющими произвольную форму. В силу этого обстоятельства аккумуляторы Li-pol нашли широкое применение, например, в планшетах.

Зарядка аккумуляторов

В радиолюбительской практике и в быту обычно приходится самостоятельно заряжать универсальные аккумуляторы: пальчиковые NiMH или, изредка, NiCd-разновидности. Самому сооружать для них зарядные устройства бессмысленно, проще и дешевле их приобрести. В любом случае лучше не использовать дешевый блок зарядки без автоматики, внутри которого всего только и есть, что диод да ограничивающий ток резистор. Взрываться такие аккумуляторы, скорее всего, не станут, а вот перезаряда они не любят и быстро от этого портятся (NiCd, в частности, имеют привычку при регулярном перезаряде вздуваться). Если все же вам жаль потратиться на приличный «интеллектуальный» зарядник фирмы AcmePower или Sony, то покупайте хотя бы такой, который имеет таймер для своевременного выключения. Правда, таймер обычно рассчитывается на «среднепотолочную» емкость, но в описании к зарядному должно быть указано, на какую емкость номинально он рассчитан.

Как правильно рассчитать время заряда, если у вас нет «умного» зарядника или емкость отличается от номинальной? Просто поделите энергоемкость аккумулятора (в мА·ч) на зарядный ток, который выдает ваше устройство (в мА), и вы получите время в часах, которое нужно умножить примерно на 1,3–1,4. Если величина тока не ука-

зана, то в инструкции обычно приводится таблица времени зарядки в зависимости от емкости, тогда ток можно ориентировочно подсчитать самостоятельно, можно и попытаться померить его мультиметром. Обычный «универсальный» режим заряда, который не может повредить никакому аккумулятору (кроме, конечно, литиевых), предполагает зарядку током в одну десятую от емкости — например, АА-тип емкостью 2000 мА·ч надо заряжать 13–14 часов током 200 мА. Разумеется, этот расчет относится к полностью разряженному аккумулятору, т. к. точный расчет времени при частичном разряде — задача практически нерешаемая.

Вторичные линейные источники питания

Остальные варианты источников питания мобильными не являются и носят общее название *вторичных источников питания* (ИВЭП, источники вторичного электропитания), потому что они преобразуют энергию бытовой электросети в нужное напряжение постоянного тока. Кстати, об аналогии между потоком в трубе и электричеством: известное потребителям баллонного газа устройство, называемое редуктором (оно превращает газ высокого давления из баллона в газ с почти постоянным низким давлением для подачи потребителям), есть полный аналог такого вторичного источника питания.

В малопотребляющих конструкциях, вроде рассматриваемых в этой книге, чаще всего целесообразно использовать обычные трансформаторные («линейные») источники в силу их простоты, надежности и дешевизны, к рассмотрению которых мы сейчас и перейдем.

Но перед этим упомянем еще об одной альтернативе, которая была весьма модной в радиолюбительских кругах в советские времена, — бестрансформаторных источниках питания от сети. Вы можете наткнуться на нечто подобное, если перелистаете старые журналы «Радио». В связи с этим следует сказать только одно:

НИКОГДА НЕ СТРОЙТЕ ПРИБОРА,
РАБОТАЮЩЕГО ОТ СЕТИ ПЕРЕМЕННОГО ТОКА БЕЗ ТРАНСФОРМАТОРА!

Это опасно для жизни — ваша схема будет всегда находиться под высоким напряжением относительно земли (земли без кавычек — т. е. водопроводных труб, батарей отопления и т. п.). Если схема предназначена для управления сетевой нагрузкой, то это управление следует обязательно осуществлять через гальванически развязывающие элементы: реле, оптроны, электронные реле, трансформаторы и т. п.

Единственное исключение вы встретите в следующей главе, где будет идти речь об управлении мощной нагрузкой, — там безопасность должна обеспечиваться конструктивными методами (изоляцией корпуса, элементов управления и пр.).

Трансформаторы

Основой трансформаторных источников служит сетевой трансформатор. Независимо от конкретной конструкции, трансформаторы всегда устроены по одному принципу: на замкнутом каркасе из металлических пластин или ленты находятся несколько обмоток. Две самые распространенные разновидности трансформаторов — с Ш-образным и тороидальным сердечником — схематично показаны на

рис. 9.5. Если есть выбор, то лучше предпочесть тороидальный трансформатор, — у него меньшее магнитное поле рассеяния и, главное, в случае чего на него легко домотать недостающие обмотки или добавить витков к имеющимся. При выборе готового трансформатора следует предпочесть те, которые залиты компаундом (в старинных конструкциях употреблялся просто парафин). Или, по крайней мере, у трансформаторов с Ш-образным сердечником катушка с обмотками должна прочно, без люфта, держаться на стержне, а сами пластины должны быть обязательно плотно сжаты специальной скобой, изолированной от пластин пластиковой или картонной прокладкой. Иначе трансформатор неизбежно будет во время работы гудеть.

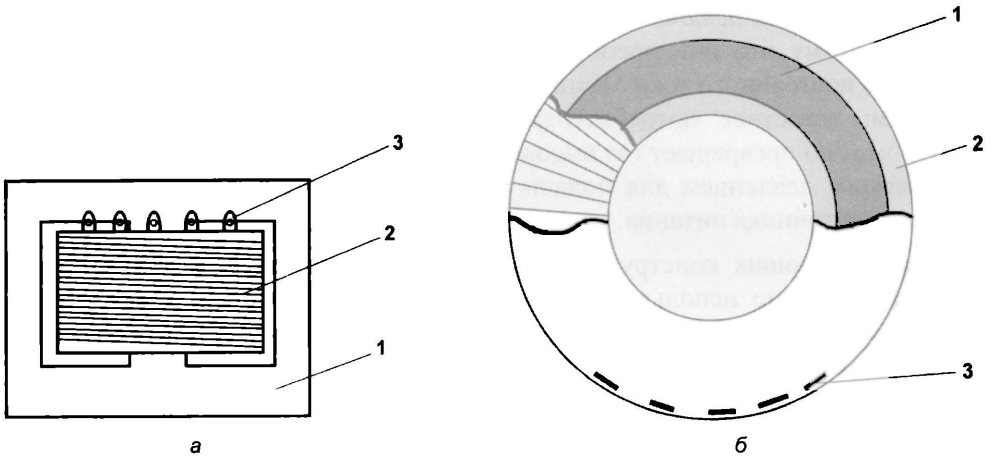


Рис. 9.5. Трансформаторы с Ш-образным (*а*) и тороидальным (*б*) сердечником:
1 — сердечник; 2 — обмотки; 3 — выводы обмоток

Одна из обмоток называется *первичной* — т. к. мы рассматриваем сетевые трансформаторы, то она всегда рассчитана на сетевое напряжение. Найти ее, если характеристики обмоток неизвестны, не очень сложно — она всегда имеет наибольшее сопротивление из всех, причем для малогабаритных трансформаторов это сопротивление может достигать сотен ом или даже нескольких килоом (с уменьшением габаритов число витков растет, а диаметр провода падает). Иногда она поделена на две, которые перед включением нужно соединить, а также может иметь отводы для более точной подгонки напряжений или для обеспечения возможности переключения 220/120 В. Сравнивая сопротивления выводов между собой, можно найти эти отводы. Другой способ нахождения первичной обмотки — она всегда намотана наиболее тонким проводом (вообще — чем толще провод, тем меньше напряжение на обмотке, как мы увидим далее). Исключение из этого правила представляют трансформаторы от старинной ламповой техники (там самым тонким проводом наматывают обмотку анодного напряжения), но для современных схем они не годятся (и хранить их «на всякий случай» не стоит даже завзятым барахольщикам).

Остальные обмотки — *вторичные*, их можно соединять между собой в любой комбинации. Обмотки имеют начало и конец. Для суммирования напряжений обмоток

надо соединять конец одной обмотки с началом другой. Смысл понятий начала и конца обмоток прост — где начинали мотать обмотку, там и начало. Если намотать следующую обмотку *в том же направлении* (а так всегда и поступают), то у нее начало будет с той же стороны, что и у первой. Если это фабричный трансформатор, и выводы у него нумерованы, то нечетные выводы принимаются за начала обмоток, а четные — за концы, т. е. при соединении двух обмоток с нумерацией выводов 1–2 и 5–6 для сложения напряжений нужно соединить вывод 2 первой обмотки с выводом 5 второй (или вывод 1 первой с выводом 6 второй), а оставшиеся выводы 1–6 (или 5–2) будут, соответственно, началом и концом объединенной обмотки. Для серийно выпускающихся трансформаторов в торгующих ими организациях имеются справочники по типовым разновидностям с указанием характеристик обмоток и нумерации их выводов.

Расчет сетевого трансформатора

Я надеюсь, что вам никогда не придется самим мотать сетевые трансформаторы, но все же приведу полуэмпирическую, но проверенную практикой, методику для их расчета, т. к. некоторые из формул могут вам пригодиться для доработки серийных трансформаторов и определения характеристик трансформаторов, имеющихся в наличии. Кроме того, знание закономерностей расчета способствует правильному выбору при их приобретении.

Главное соотношение (можно назвать это *законом трансформатора*):

$$U_1/U_2 = n_1/n_2, \quad (1)$$

где:

- U_1 — напряжение первичной обмотки;
- U_2 — напряжение вторичной обмотки;
- n_1 — число витков первичной обмотки;
- n_2 — число витков вторичной обмотки.

Как видите, все необычайно просто. Если, скажем, первичная обмотка имеет 220 витков (это должен быть довольно мощный трансформатор, у маломощных число витков может составлять несколько тысяч), а вторичная — 22 витка, то при подключении к сети 220 В на вторичной обмотке будет 22 В. Токи находятся в обратном соотношении — если ток вторичной обмотки составляет 1 А, то первичная обмотка будет потреблять от сети 100 мА. Если вторичных обмоток несколько, то для определения потребления тока от сети их токи нужно пересчитать на первичную обмотку в отдельности (количество витков при этом знать необязательно, достаточно только напряжения), а затем сложить. Можно пойти и другим путем — суммировать мощности, потребляемые вторичными обмотками (которые равны произведениям токов на напряжения), а затем поделить полученную сумму на 220 — получим ток в первичной обмотке.

Кстати, из этого закона вытекает простой метод определения количества витков в обмотках готового трансформатора, если это зачем-то нужно: намотайте поверх

имеющихся обмоток несколько витков любого провода, включите трансформатор и измерьте напряжение на этой импровизированной обмотке. Поделив количество намотанных витков на полученное значение напряжения, вы получите величину количества витков на один вольт, которая едина для всех обмоток, а далее пересчитать полученный результат на другие обмотки уже не составляет трудностей.

При выборе напряжений вторичных обмоток учтите, что их нужно выбирать с запасом (это относится и к покупным трансформаторам), — под нагрузкой напряжение садится, и это просаживание тем больше, чем меньше мощность трансформатора. Поэтому, если вам задано минимально допустимое напряжение 7 В, — выбирайте трансформатор с 9–10-вольтовой обмоткой, не ошибетесь.

Итак, сформулируем задачу: допустим, необходимо иметь трансформатор с двумя вторичными обмотками, рассчитанными на напряжение 27 В и ток 200 мА каждая, и еще одной обмоткой, рассчитанной на напряжение 9 В и ток до 3 А. Подсчитаем суммарную мощность: 27·0,2·2 + 9·3 = 37,8 Вт, округляем до 40 Вт. Ток в первичной обмотке составит 40/220 = 0,18 А, округляем до 0,2 А. Теперь у нас есть все исходные данные.

Сначала определяем сечение магнитопровода в см^2 (для Ш-образных трансформаторов это есть сечение центрального стержня, на котором находится катушка с обмотками, для тороидального — просто поперечное сечение тора). Это делается по формуле:

$$S = 1{,}15 \cdot \sqrt{P}, \qquad (2)$$

где: S — сечение в см^2, P — мощность в Вт. Получаем 7,3 см^2 — уже можно подбирать магнитопровод. Они стандартизированы, так что выбираем из справочника подходящий с округлением в большую сторону: для Ш-образного сердечника можно взять, например, 8 см^2 (25×32 мм). По формуле (2) также всегда можно определить неизвестную мощность имеющегося в наличии трансформатора — достаточно измерить сечение его магнитопровода.

Затем нужно подсчитать требующееся при такой мощности количество витков первичной обмотки:

$$n_1 = 50 \cdot U_1 / S, \qquad (3)$$

где: n_1 — число витков, U_1 — напряжение (220 В), S — подобранное сечение в см^2. Получаем 1375 витков. Рассчитать теперь количество витков вторичных обмоток — дело техники, только не забывайте всегда округлять в большую сторону.

И, наконец, рассчитываем необходимый диаметр провода в мм^2 для каждой обмотки:

$$d_i = 0{,}8 \cdot \sqrt{I_i}, \qquad (4)$$

где: d_i — диаметр провода в i-й обмотке, а I_i — ток в этой обмотке. Получаем для первичной обмотки диаметр 0,36, для 27-вольтовых также 0,36, а для 9-вольтовой — 1,4 мм. Для первичной и 27-вольтовых обмоток можно выбрать стандартный провод диаметром 0,355 или 0,38 мм, а для 9-вольтовой расчет совпадает со стандартным диаметром 1,4 мм, хотя можно взять и 1,5 мм. Конечно, чем больше диаметр по сравнению с расчетным, тем только лучше, но при чрезмерном превы-

шении вы рискуете при намотке не влезть в окно стандартного сердечника, так что увлекаться не стоит.

Все, расчет закончен. Формулу (4) стоит запомнить, т. к. она может пригодиться, если придется домотывать витки к имеющимся в наличии трансформаторам: сначала по приведенной ранее методике определяется количество витков на вольт, из чего высчитывается необходимое количество витков, а затем по формуле (4) — нужный диаметр провода.

Учтите, что закон трансформатора (1) справедлив для всех видов трансформаторов, а вот все остальные соотношения, за исключением разве что (4), годятся только для расчета сетевых трансформаторов, работающих на частоте 50 Гц. Ни для каких других трансформаторов (согласующих с ферритовыми сердечниками) эта методика не действует.

Ну, а теперь перейдем к более интересным вещам.

Простейший нестабилизированный однополярный источник питания

Схема простейшего источника питания приведена на рис. 9.6. Именно по такой схеме устроены почти все недорогие маломощные блоки питания, встроенные в сетевую вилку. Иногда в них вторичная обмотка имеет несколько отводов и присутствует ползунковый переключатель, который коммутирует эти отводы, меняя выходное напряжение. Так как эти блоки весьма дешевы, то в случае, когда вам не требуется большой мощности, спокойно можно покупать такой блок, разбирать его и встраивать в вашу аппаратуру (или даже не встраивать — хотя, на мой вкус, громоздкие надолбы на розетках отнюдь не украшают интерьер, все время хотят вывалиться и к тому же не во всякую розетку влезают). Нужно только обратить внимание на допустимый ток нагрузки, который указан на корпусе блока. Что касается номинального напряжения, то этот вопрос мы сейчас рассмотрим.

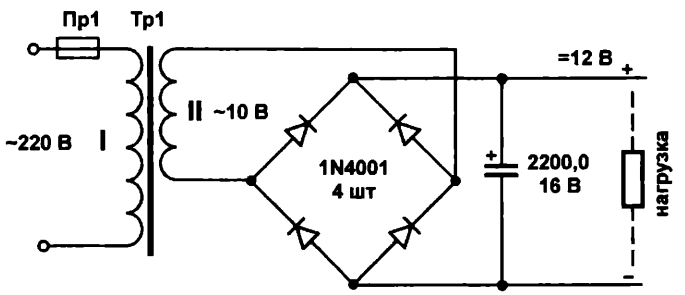


Рис. 9.6. Простейший нестабилизированный однополярный источник питания

Как работает эта схема? Здесь переменный синусоидальный ток со вторичной обмотки трансформатора (II) подается на конструкцию из четырех диодов, которая называется *диодным мостом* и представляет собой простейший двухполупериодный выпрямитель (есть и другие способы двухполупериодного выпрямления —

см. далее рис. 9.18 и пояснения к нему). В мосте могут быть использованы любые типы выпрямительных диодов, лишь бы их предельно допустимый ток был не меньше необходимого (для указанных на схеме диодов 1N4001 это 1 А), а предельно допустимое напряжение — не меньше половины амплитудного значения входного переменного напряжения (так как в нашем случае это всего 7 В, то здесь этому требованию удовлетворяют вообще все выпрямительные диоды на свете). Такие мосты выпускаются уже в сборе, в одном корпусе, на котором иногда даже нарисовано, куда подключать переменное напряжение и откуда снимать постоянное. Их, конечно, тоже можно и нужно использовать.

Проследим за работой моста. Предположим, что на верхнем по схеме выводе вторичной обмотки в данный момент переменное напряжение, поступающее с обмотки, больше, чем на нижнем. Тогда ток в нагрузку (она обозначена пунктиром) потечет через правый верхний диод моста, а возвратится в обмотку через левый нижний. Полярность на нагрузке, как видим, соблюдается. В следующем полупериоде, когда на верхнем выводе обмотки напряжение меньше, чем на нижнем, ток через нагрузку потечет, наоборот, через левый верхний диод и возвратится через правый нижний. Как видим, полярность опять соблюдается.

Отсюда и название такого выпрямителя — *двухполупериодный*, т. е. он работает во время обоих полупериодов переменного тока. Форма напряжения на выходе такого моста (в отсутствие конденсатора) соответствует пульсирующему напряжению, показанному на рис. 4.5, *а*. Естественно, такое пульсирующее напряжение нас не устраивает, — мы хотим иметь настоящее постоянное напряжение без пульсаций, потому в схеме присутствует сглаживающий (фильтрующий) конденсатор, который вместе с выходным активным сопротивлением трансформатора и сопротивлением диодов представляет собой не что иное, как известный нам по *главе 5* интегрирующий фильтр низкой частоты. Все высокие частоты отфильтровываются, а на выходе получается ровное постоянное напряжение. К сожалению, такая идиллия имеет место только в отсутствие нагрузки, к чему мы вернемся чуть далее, а пока попробуем определить, какова величина этого постоянного напряжения на выходе фильтра.

В отсутствие нагрузки конденсатор с первых же полупериодов после включения питания заряжается до амплитудного значения пульсирующего напряжения, которое равно амплитудному значению напряжения на вторичной обмотке за вычетом падения напряжения на *двух* диодах, стоящих на пути тока. Так как в установившемся режиме через эти диоды ток весьма мал (только для подпитки собственных токов утечки конденсатора), то и падение напряжения на них мало и в сумме составляет менее 1 В. Амплитудное значение напряжения на вторичной обмотке равно $10\sqrt{2} = 14{,}1$ В, так что на холостом ходу напряжение на выходе источника составит чуть более 13 В.

При подключении нагрузки происходит сразу много всего. Во-первых, снижается напряжение на вторичной обмотке, — трансформатор имеет конечную мощность. Во-вторых, увеличивается падение напряжения на диодах, которое может при максимально допустимом для них токе достичь 1 В на каждом. В-третьих и в-главных, во время «провалов» пульсирующего напряжения нагрузка питается только за счет того, что через нее разряжается конденсатор. Естественно, напряжение на нем при

этом каждый раз немного снижается. Поэтому график выходного напряжения при подключенной нагрузке представляет собой уже не ровную постоянную линию, а выглядит примерно так, как показано на рис. 9.7 (причем снижение входного напряжения за счет «просаживания» трансформатора здесь не учитывается). То есть, выходное напряжение немного пульсирует — тем больше, чем больше ток в нагрузке, и тем меньше, чем больше емкость конденсатора. Именно поэтому в источниках применяют электролитические конденсаторы столь большой емкости. Наличие пульсаций также снижает постоянную составляющую выходного напряжения.

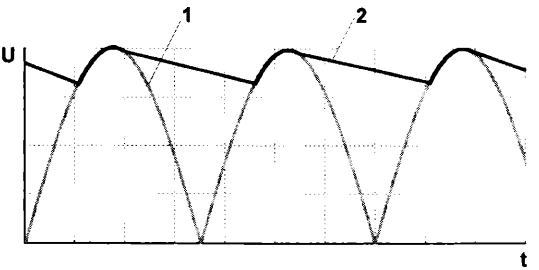


Рис. 9.7. Вид пульсаций на выходе нестабилизированного источника:
1 — исходное пульсирующее напряжение в отсутствие фильтрующего конденсатора;
2 — выходное напряжение при наличии фильтрующего конденсатора и нагрузки

В этой схеме избавиться от таких пульсаций полностью невозможно, как бы вы ни увеличивали емкость конденсатора. Кстати, а как подсчитать нужную емкость? Тут без интегралов не обойтись, потому для простоты будем руководствоваться эмпирическим правилом: на каждый ампер нагрузки минимально достаточно конденсатора около 1000 мкФ. Для надежности можно увеличить ее до 2200 мкФ. Первая величина ближе к тому случаю, когда на выходе источника предполагается поставить стабилизатор напряжения, вторая — если такого стабилизатора не предполагается.

Заметки на полях

А вот поднимать емкость выше этих величин практически бесполезно — габариты возрастут, а пульсации принципиально не уменьшатся. Это происходит оттого, что с ростом емкости растет и собственное внутреннее сопротивление электролитического конденсатора, на которое он успевает бесполезно разряжаться. Как мы говорили, у ионисторов (см. *главу 5*) это внутреннее сопротивление возрастает до 50 Ом, что равносильно току 0,1 А при напряжении 5 В. Так что, поставив емкость в целую фараду вместо 1000 мкФ, вы добьетесь снижения пульсаций всего в несколько раз. И не забудьте еще, что чем больше емкость, тем больше мгновенный ток через диоды при ее зарядке, а он хоть и может превышать средний ток в разы (для специальных выпрямительных диодов даже в десятки раз), но тоже не бесконечно большая величина.

Все указанные причины совместно приводят к тому, что под нагрузкой сверхмаломощные источники (вроде тех, что со встроенной вилкой) могут выдавать в полтора-два раза меньшее напряжение, чем на холостом ходу. Поэтому не удивляйтесь, если вы приобрели такой блочок с указанным на шильдике номинальным напряжением 12 В, а мультиметр на холостом ходу показывает аж все 18!

Чтобы покончить с темой простейшего источника, нужно сказать пару слов об имеющемся на схеме предохранителе Пр1. В упомянутых блоках со встроенной вилкой предохранитель часто отсутствует, и это вызвано, кроме стремления к удешевлению блока, очевидно тем обстоятельством, что маломощный трансформатор сам служит неплохим предохранителем, — провод первичной обмотки у него настолько тонок и сопротивление его настолько велико, что при превышении допустимого тока обмотка довольно быстро сгорает, отключая весь блок (после чего его, естественно, остается только выбросить). Но в стационарных устройствах и, тем более, в устройствах большей мощности предохранитель должен быть обязательно. Обычно его выбирают на ток в два-четыре раза больший, чем расчетный максимальный ток первичной обмотки.

Приведем еще одну полезную схему нестабилизированного источника, на этот раз двухполярного, т. е. выдающего два одинаковых напряжения относительно средней точки — «земли» (рис. 9.8). Пояснений она не требует, потому что очень похожа на однополярную, только возврат тока в обмотки от обеих нагрузок происходит непосредственно через общую «землю», минуя диодный мост. В качестве упражнения предлагаю вам самостоятельно разобраться, как она работает. Вторичные обмотки (II и III) здесь, в сущности, представляют собой две одинаковые половины одной обмотки. Жирными точками около вторичных обмоток обозначены их начала, чтобы не перепутать порядок их соединения.

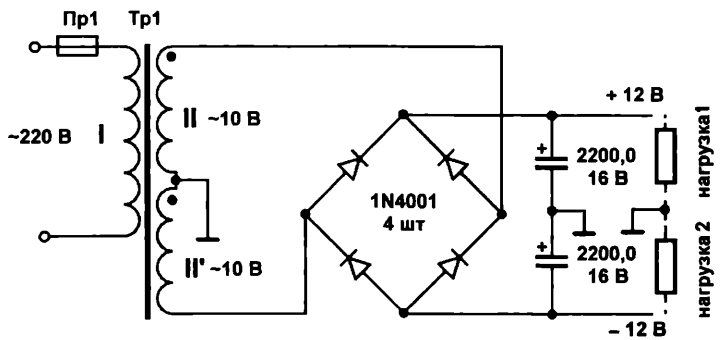


Рис. 9.8. Нестабилизированный двухполярный источник питания

Стабилизаторы

Простейший стабилизатор — это стабилитрон, который мы рассматривали в *главе 7*. Если параллельно ему подключить нагрузку $R_н$ (рис. 9.9, *а*), то напряжение на ней останется стабилизированным до тех пор, пока ток через нее не будет слишком велик. Рассчитывается схема так, чтобы в отсутствие стабилитрона напряжение в средней точке делителя из $R_{ст}$ и $R_н$ было не ниже номинального напряжения стабилизации стабилитрона $U_{ст}$, иначе при его подключении ток через него не пойдет, и стабилитрон не откроется. В результате максимальный ток, который мы можем получить в такой схеме, не превышает нескольких десятков миллиампер — в зависимости от мощности стабилитрона. Такой стабилизатор называют еще *параметрическим*.

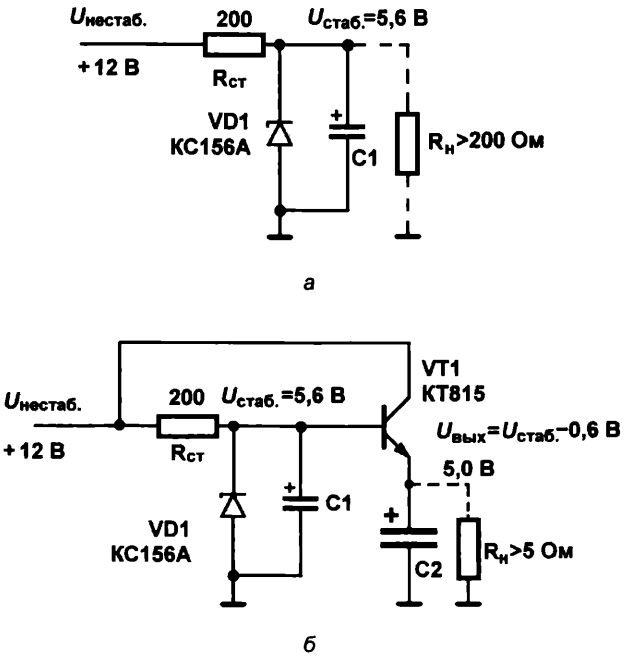


Рис. 9.9. Два стабилизатора для источников питания:
а — самый простой на стабилитроне; *б* — с эмиттерным повторителем

Вы зададите вопрос — а зачем здесь конденсатор? Ведь в нестабилизированном источнике, который мы рассмотрели ранее, и откуда поступает напряжение на этот стабилизатор, один фильтрующий конденсатор уже имеется, не так ли? Ответ простой: на выходе всех типов стабилизаторов всегда ставится конденсатор, как и до них. Он позволяет сгладить остаточные пульсации, которые все равно просочатся на выход, потому что стабилитрон имеет свое дифференциальное сопротивление, и при изменении входного напряжения или тока в нагрузке напряжение на нем также будет меняться, хоть и в небольшой степени. Величина емкости здесь может быть значительно меньше, чем на выходе выпрямительного моста, но не жадничайте — стоимость конденсаторов нынче такова, что поставить здесь конденсатор емкостью, к примеру, 470 мкФ ничто вам помешать не может, а по размерам и стоимости он будет мало отличаться от такого же, но емкостью 47 мкФ. Для интегральных стабилизаторов, которые мы будем рассматривать далее, конденсатор на выходе положен по рекомендациям производителя, но он может быть меньше, — обычно рекомендуется ставить керамический, емкостью 0,1–1 мкФ.

Значительно интересней схема на рис. 9.9, *б*. Здесь транзистор включен эмиттерным повторителем, который, во-первых, имеет высокое входное сопротивление (поэтому ток через стабилитрон практически не зависит от изменений тока в нагрузке), во-вторых, служит усилителем тока (подробности см. в *главе 6*). То есть, мощностные возможности здесь определяются только транзистором. Конденсаторов здесь целых два: первый помогает сглаживать пульсации на стабилитроне, второй — оставшиеся пульсации на выходе транзистора.

При указанных на схеме параметрах она выдаст нам около 1 А. Статический коэффициент передачи тока для транзистора КТ815А равен (по справочнику) 40, поэтому базовый ток при 1 А на выходе должен составить не менее 25 мА, а ток через стабилитрон КС156А ни при каких условиях не должен быть меньше 3 мА (минимальное допустимое значение). Из этих соображений выбирается величина сопротивления $R_{ст}$ = 200 Ом.

Да, кстати, а какая мощность выделится на «проходном» транзисторе VT1? Не такая уж и маленькая: (12 В – 5 В) · 1 А = целых 7 Вт! То есть, его явно надо ставить на радиатор, методику расчета которых мы будем рассматривать далее. Отсюда виден главный недостаток подобных аналоговых стабилизаторов — низкий КПД. В данном случае он всего около сорока процентов (проверьте!), остальное рассеивается в пространстве. Мы можем его несколько повысить, снижая входное напряжение, но только до определенного предела — здесь он равен примерно 8 В, иначе эта схема не справится. Помните, однако, что 8 В — это действительно нижний предел, а не среднее значение пульсирующего напряжения на выходе конденсатора фильтра, которое показывает вольтметр, — если вы еще раз взглянете на рис. 9.7, то поймете, о чем я. В противном случае стабилизатор просто перестанет стабилизировать. Потому в таких примитивных схемах всегда следует иметь запас, и не слишком маленький.

Заменой *n-p-n*-транзистора на *p-n-p* с соответствующим изменением всех полярностей (в том числе переворотом конденсаторов и стабилитрона) на обратные, мы получим стабилизатор отрицательного напряжения. А для получения большего тока на выходе вместо обычного транзистора можно поставить транзистор с «супербетой». Так если мы заменим КТ815 на «дарлингтоновский» КТ829, то можем «выжать» уже до 10 А, только для сохранения значения выходного напряжения вместо стабилитрона КС156А придется взять КС162А. И не забудьте, что и нестабилизированный источник тоже должен обеспечить такой ток, да и радиатор придется ставить существенно больший!

Идя по этому пути, мы можем построить недорогой двухполярный источник питания для нашего усилителя из *главы 8*. Если вы ее внимательно перечтете, то сообразите, что номинальная мощность источника для такого усилителя должна составлять не менее 100 Вт (пиковый ток в нагрузке может достигать 3,3 А при максимальной выходной мощности усилителя) или по 50 Вт на каждом из двухполярных напряжений по 15 В. Соответствующая этим условиям схема источника питания для усилителя, описанного в *главе 8*, приведена на рис. 9.10. Я привожу ее без пояснений, потому что всеми необходимыми сведениями, чтобы в ней разобраться, вы уже обладаете. Стабилитрон 1N4745A — достаточно мощный (ток стабилизации — до 57 мА), с напряжением стабилизации 16 В. Светодиоды (VD7, VD8) сигнализируют о наличии напряжения по обоим каналам. Импортную мощную «дарлингтоновскую» пару BDW93C/BDW94C можно заменить на отечественную КТ827/КТ825, расчет радиатора для них представлен далее в *разд. «Рассеивание тепла»* этой главы.

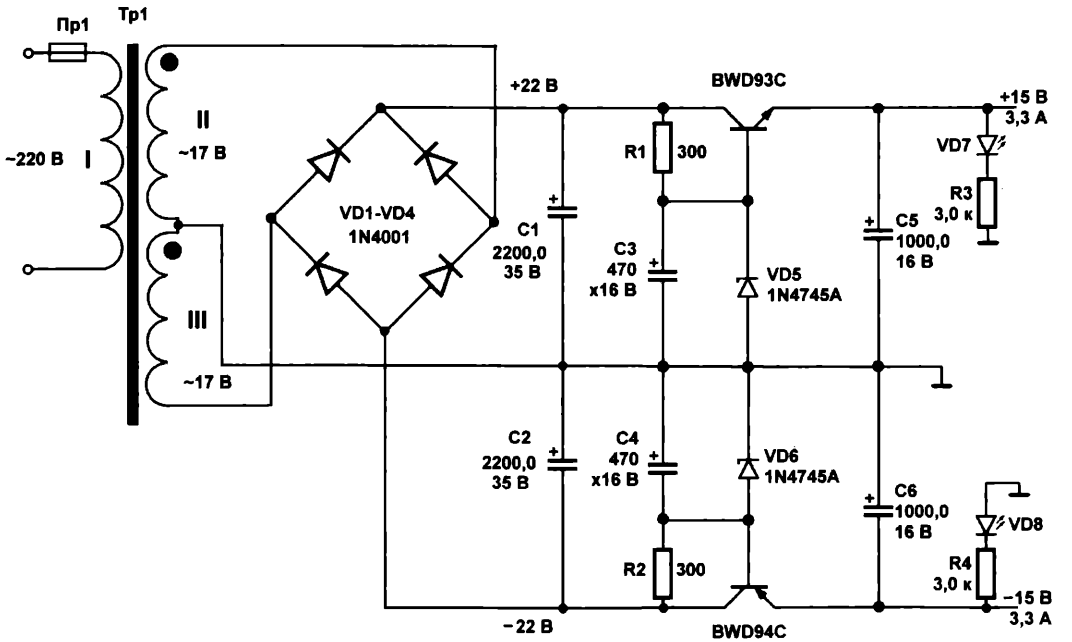


Рис. 9.10. Мощный двухполярный стабилизатор на ±15 В, 4А (для усилителя из *главы 8*)

Интегральные стабилизаторы

Совершенно естественным ходом было бы упаковать типовой узел, состоящий из стабилитрона, транзистора и резистора в одну микросхему. Однако выдающийся схемотехник и разработчик аналоговых микроэлектронных устройств Р. Видлар, о котором мы еще вспомним в связи с изобретением интегрального операционного усилителя, рассудил несколько иначе.

Действительно, такая простейшая схема, как на рис. 9.9, *б*, обладает целым рядом недостатков, главным из которых является низкий коэффициент стабилизации. В зависимости от входного напряжения и от выходного тока напряжение на выходе может довольно сильно меняться и медленно отрабатывать быстрые изменения в нагрузке. Поэтому наилучшим выходом было бы использовать в стабилизаторах принцип отрицательной обратной связи, с которым мы уже отчасти познакомились, изучая работу звукового усилителя в *главе 8*. Далее мы более подробно рассмотрим стабилизатор с обратной связью, а пока заметим, что такую схему не особенно трудно построить и на дискретных транзисторах, но с повышением качества ее сложность и, соответственно, стоимость резко возрастают. А вот в производстве микросхем безразлично — три транзистора они содержат или тридцать. Кроме того, все транзисторы находятся на одном кристалле и имеют одинаковую температуру и близкие характеристики, что недостижимо в дискретных схемах. Видлар этим воспользовался и сконструировал микросхему μA723, которая и легла в основу современных семейств интегральных стабилизаторов.

Наиболее широко распространена и доступна серия стабилизаторов LM78/79*xx*. Имейте в виду, что семейство LM содержит и другие типы микросхем, и это название не должно вас смущать. Выпускаются они очень многими производителями, вследствие чего буквы могут различаться, но цифры остаются теми же. Цифры означают вот что: первые две — наименование серии (78 — стабилизатор положительного напряжения, 79 — отрицательного), вторые две — напряжение стабилизации (напр. 7805 — пятивольтовый стабилизатор положительного напряжения). Выпускаются аналоги этой серии и в России, однако принцип наименования у нас другой — это серия 142ЕН*xx* и др. Напряжения стабилизации в серии LM78/79 фиксированы, однако имеются и регулируемые типы (LM317, КР142ЕН12).

На рис. 9.11, *вверху*, приведена типовая схема включения такого стабилизатора и показано, как он может выглядеть внешне. В корпусе TO-220, показанном на рис. 9.11, *внизу*, такой стабилизатор может выдать ток до 2,4 А, если рассеиваемая мощность не превышает 20 Вт (с радиатором, естественно). Но есть большой выбор и других корпусов, включая корпуса для поверхностного монтажа. Особенно удобен маленький корпус TO-92 (тогда в название вклинивается буква L, например: 78L05) — он позволяет стабилизировать питание отдельных узлов независимо друг от друга, избежав таким образом их взаимного влияния. Выходной ток стабилизаторов LM78L/79L в корпусе TO-92 — до 100 мА. Их, вообще-то, можно использовать и вместо стабилитронов в схемах по типу приведенных на рис. 9.8, *б*, но не забывайте, что выходное напряжение в них ниже стандартного на величину падения напряжения U_{63}. Потому, кроме редких случаев устройств экстремально высокого потребления при хорошо стабилизованном напряжении, когда волей-неволей приходится стабилизатор разрабатывать специально, проще (и дешевле) просто поискать готовый стабилизатор помощнее.

Разумеется, серия 78/79*xx* — не единственная в своем роде, есть и множество других, аналогичных по функциональности. Так, стабилизаторы серий LM2931

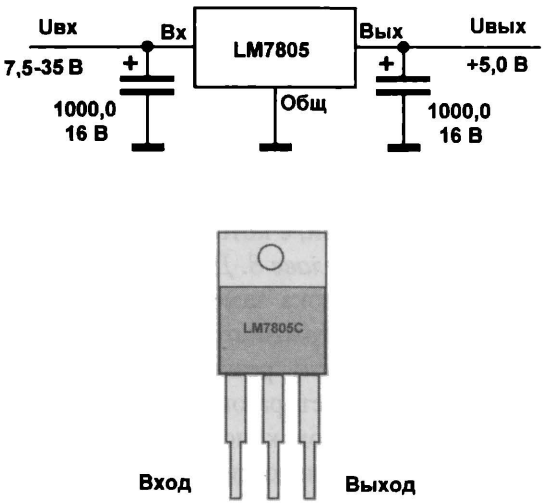


Рис. 9.11. Схема включения интегрального стабилизатора (*вверху*); корпус TO-220 (*внизу*)

(5-вольтовый) или LP2950 (на напряжение 5; 3,3 и 3 В) с выходным током до 100 мА отличаются сверхмалым собственным потреблением (несколько десятков микроампер) и сверхнизкой разницей напряжений на входе и выходе, при которой стабилизатор еще выполняет свои функции (достаточно перепада в несколько сотен милливольт, только не забывайте про пульсации!). Более современное семейство на замену 78*xx* — это LM1117 (со стандартным для цифровой техники рядом напряжений от 1,8 до 5,0 В, а также регулируемым типом), которое выпускается в разных корпусах, в том числе и в корпусе TO-220, как и 78*xx*. Независимо от корпуса, это семейство работает вплоть до разницы между входом и выходом всего 1,2 В (для 78*xx* требуется не менее 2 В) и ограничивает выходной ток значением 1,1–1,2 А. Следует учесть, что у LM1117, как у аналогичного LM1085, довольно велик собственный ток покоя (до 5–10 мА), и условием хорошего регулирования является ограничение *минимального* тока потребления, потому для малопотребляющих устройств они не годятся. Специальные малопотребляющие стабилизаторы (как, например, MIC5205 или LP2985 с токами покоя порядка микроампера), кроме упомянутых в начале абзаца, к сожалению, в корпусах с гибкими выводами не производятся.

Однополярный регулируемый источник питания

Схема на рис. 9.12 представляет собой лабораторный источник питания, который, как я обещал вам еще в *главе 2*, можно изготовить самим. Взглянув на эту схему, вы можете сначала слегка растеряться, — настолько вам покажется все незнакомо. На самом деле там есть только одна вещь, которую мы еще «не проходили», — микросхема *операционного усилителя* (ОУ) DA1. Подробно с ОУ мы будем знакомиться в *главе 12*, а сейчас нам важно только вот что: ОУ всегда стремится сделать так, чтобы потенциалы входов, обозначенных «плюс» и «минус», были равны. Эти входы эквивалентны входам дифференциального усилителя, у которого, как вы помните, потенциалы входов тоже связаны между собой (на самом деле внутри микросхемы на входе ОУ действительно стоит дифференциальный каскад). Для того чтобы это осуществлялось на практике, ОУ включают с отрицательной обратной связью с выхода на тот вход, который обозначен знаком «минус». В схеме, показанной на рис. 9.12, такая связь осуществляется весьма заковыристым способом, и для того чтобы понять, как это происходит, давайте посмотрим на рис. 9.13, на котором изображена та же самая схема, но в предельно упрощенном варианте.

Предположим, что R1 и R2 на рис. 9.13 равны между собой. Какое напряжение будет на выходе, т. е. на эмиттере транзистора VT1? Определить это очень просто. Если на «плюсовом» входе ОУ напряжение 1 В, как обозначено на схеме, то на минусовом тоже должен быть 1 В, как мы только что узнали. При каком условии это возможно? Только если на верхнем выводе R1, т. е. на выходе всей системы, будет 2 В, — ведь R1 и R2 делят это напряжение пополам. То есть ОУ автоматически установит на базе транзистора VT1 такое напряжение, чтобы на его эмиттере было ровно 2 В (можно даже догадаться, какое именно — на 0,6 В больше, чем на выходе, т. е. 2,6 В, но на самом деле это нас мало интересует). А если предположить, что R1 в два раза больше, чем R2? Повторив предыдущие рассуждения, мы обнаружим,

что на выходе должно быть 3 В. Отсюда можно вывести некоторую закономерность: система, показанная на рис. 9.13, усиливает напряжение, поданное на «плюсовой» вход, ровно в (R1/R2 + 1) раз.

Именно так и работает схема источника на рис. 9.12. Переключатель П1 имеет 6 положений, в каждом из которых он изменяет соотношение делителя в обратной связи таким образом, чтобы при напряжении 1 В на «плюсовом» входе на выходе

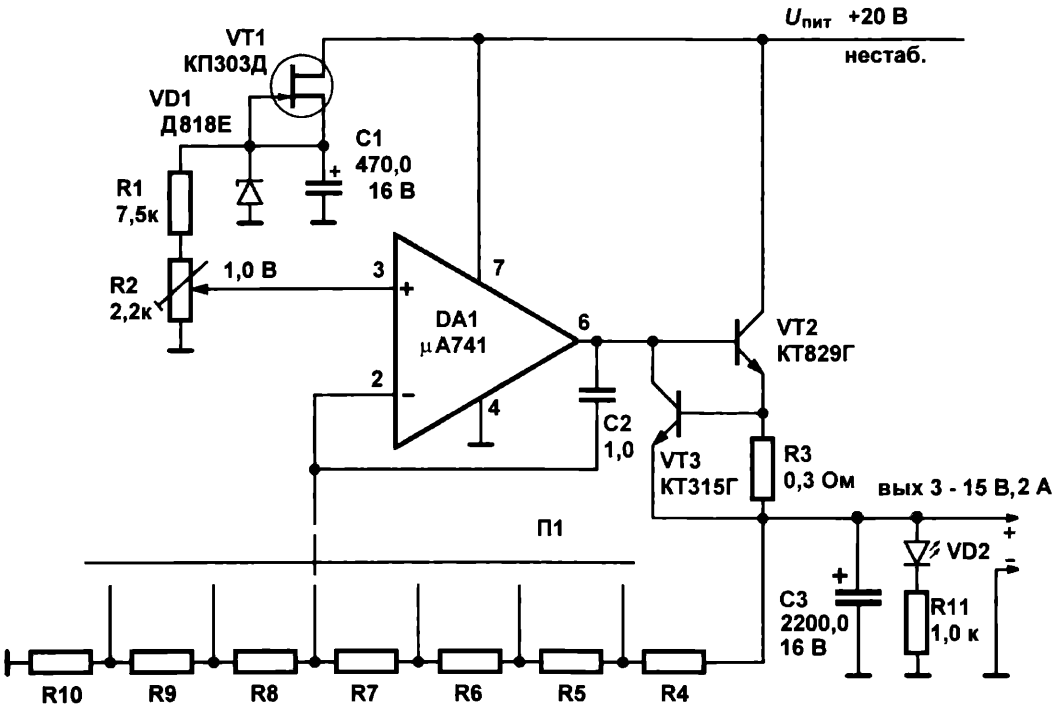


Рис. 9.12. Схема лабораторного источника питания

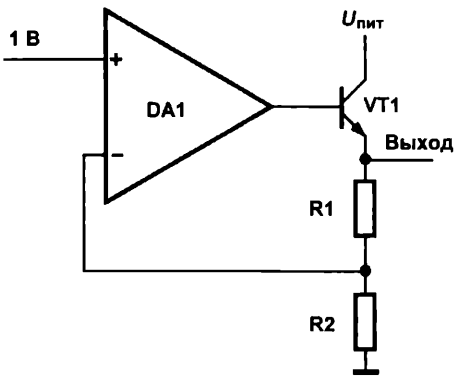


Рис. 9.13. Упрощенная схема лабораторного источника питания

получался некий ряд фиксированных напряжений. При указанных в табл. 9.1 номиналах резисторов R4–R10 этот ряд будет следующим: 3; 5; 7,5; 10; 12 и 15 В, чего достаточно для большинства наших нужд.

Таблица 9.1. *Значение сопротивлений делителя П1 (см. рис. 9.12)*

Резистор	Точное значение	Ближайшее из ряда E96
R4	4,0 к	4,02 к
R5	2,0 к	2,00 к
R6	770	768
R7	430	432
R8	133	133
R9	133	133
R10	533	536

Конечно, можно не возиться с переключателем и подбором сопротивлений, а просто поставить вместо цепочки R5–R9 переменный резистор (по схеме потенциометра), равный сумме этих сопротивлений, — эффект будет таким же, только напряжение станет меняться плавно: от 3 до 15 В. Однако иметь набор фиксированных напряжений намного удобнее — тут вы получите точно известное напряжение, а при плавной регулировке его каждый раз придется подгонять по вольтметру. Разумеется, бывают изредка ситуации, когда нужно получить напряжение, скажем, 4,75 вольта, но на этот случай лучше завести отдельный плавно регулируемый источник.

Делитель можно устроить совершенно по-разному: возьмите переключатель на 12 положений — получите переключение через 1 В. Пересчитать номиналы резисторов из описанного ранее общего соотношения несложно: так, если хочется вместо 10 В в приведенном ряду иметь 9 В, то номинал R8 следует увеличить до 224 Ом, а R7 — уменьшить до 205 Ом (при этом сумма сохранится, и остальные напряжения не изменятся). Можно добавить переменный резистор и плавно регулировать напряжение внутри каждого фиксированного диапазона. (Подумайте, как это сделать? Подсказка: переключатель должен быть на два направления.) Отметим, что в этой схеме применять прецизионные резисторы С2-29В совершенно необязательно, — не те точности требуются. Поэтому можно требуемые номиналы просто подобрать из набора обычных, стараясь выдержать их как можно ближе к расчетным (см. табл. 9.1). Допускается также весь расчетный ряд умножить или поделить на любое число, лишь бы все значения изменились в одинаковой степени. Границы, которыми следует при этом руководствоваться, — это нижний предел суммы всех резисторов в 1–2 кОм, а верхний — в пару десятков кОм.

Теперь перейдем к подробному рассмотрению остальных, вспомогательных узлов схемы. Монструозная конструкция с полевым транзистором наверху на самом деле всего лишь узел, который позволяет получить стабильное опорное напряжение

ровно 1 В, — от его стабильности точность шкалы выходных напряжений зависит напрямую.

В педагогических целях рассмотрим подробнее, как работает такая древняя схема. Полевой *n*-канальный транзистор VT1 включен источником тока, известным нам из *главы 6*, — когда потенциалы затвора и истока равны, то ток сток-исток мало зависит от напряжения на стоке. Этот ток питает прецизионный стабилитрон VD1 типа Д818Е, напряжение которого мало зависит от температуры (но очень даже зависит от тока). Если будете искать замену транзистору VT1, то в первую очередь надо смотреть на начальный ток стока, — именно такой ток будет протекать через стабилитрон в этой схеме, а стабилитрон Д818Е хорошо работает довольно в узком диапазоне токов: не менее 5 и не более 15 мА. Напряжение стабилизации стабилитрона равно 9 В, поэтому оно подается на делитель, составленный из большего постоянного (R1) и меньшего подстроечного (R2) резистора «под отвертку», с движка которого и снимается напряжение, равное 1 В.

Учтите, что вся эта схема имеет смысл только в сочетании именно с Д818Е, стабилитроны Д818 с другой буквой или их аналоги имеют намного большую температурную нестабильность. При подготовке нового издания книги я честно пытался найти более современную замену этому древнему стабилитрону, но так ничего равноценного и не обнаружил. Очевидно, такие точные стабилитроны (ТКН = = 0,001 %/°С) выпускать теперь не имеет смысла, — их с большим запасом заменяют источники опорного напряжения, выполненные в виде микросхемы. То есть, вместо всей этой конструкции можно поставить такую микросхему или просто любой стабилизатор из серии LM. Только сопротивление R1 придется пересчитать так, чтобы в среднем положении движка R2 на нем сохранилось около 1 В. Потенциометром этим можно плавно менять всю шкалу напряжений на выходе (но до определенного предела, ограниченного как снизу, так и сверху). Разумеется, цепочку R1–R2 вполне можно заменить двумя постоянными резисторами.

Теперь перейдем к транзистору VT3 вкупе с резистором R3. Эта простая и остроумная конструкция выполняет важнейшую функцию — она ограничивает выходной ток. Как это происходит? Обратите внимание, что весь выходной ток протекает через резистор R3, номинальное значение которого всего 0,3 Ом. В нормальном состоянии (например, на холостом ходу) падение напряжения на этом резисторе мало, поэтому транзистор VT3 закрыт, и весь этот фрагмент не оказывает никакого влияния на работу схемы. Когда же выходной ток достигает значения примерно 2 А, падение напряжения на нем достигает сакраментальных 0,6 В, транзистор VT3 приоткрывается и начинает шунтировать переход база-эмиттер силового транзистора VT2, призакрывая его. В результате схема приходит в равновесие: если бы VT3 приоткрылся еще больше, закрывая силовой транзистор, выходной ток бы упал, падение напряжения на R3 уменьшилось бы, VT3 призакрылся бы, ну и т. д., — и все застывает на уровне 2 А выходного тока, даже при коротком замыкании на выходе! Как только избыточная нагрузка на выходе будет снята, схема автоматически вернется в нормальный режим. Если вместо резистора R3 поставить переключатель с набором сопротивлений, то можно регулировать уровень стабилизации выходного тока. Так, набор резисторов 0,3; 0,6; 1,2; 2,4; 6 и 62 Ом даст ряд ограничений тока на уровне 2; 1; 0,5, 0,25 А, 100 и 10 мА.

Кстати, к следящему транзистору VT3 никаких требований не предъявляется — т. е. вообще никаких: можно взять любой кремниевый транзистор, только он должен быть маломощным (чтобы не шунтировать силовой транзистор токами утечки) и не составным по схеме Дарлингтона. А вот силовой транзистор, наоборот, должен быть именно дарлингтоновский, с «супербетой».

В этой схеме есть одно, однако большое, НО. Заключается оно в том, что при коротком замыкании на выходе все напряжение питания будет падать на переходе коллектор-эмиттер транзистора VT2, — ему больше просто некуда деваться. То есть, выделяющаяся мощность на VT2 составит аж целых 40 Вт! И в нормальном режиме при маленьких установленных выходных напряжениях (3 или 5 В) и максимальной нагрузке эта мощность будет практически такой же. В этом и заключается главный недостаток рассматриваемой схемы, общий для всех линейных стабилизаторов, — крайне низкий КПД.

Есть, впрочем, немало способов этот КПД повысить. Продаю идею простейшего из них, который годится именно для стабилизатора с дискретным набором выходных напряжений: надо взять трансформатор нестабилизированного источника, от которого питается вся эта схема, с несколькими обмотками на разное напряжение, а к переключателю делителя добавить еще одно направление переключения так, чтобы при снижении напряжения на выходе напряжение питания стабилизатора также снижалось (с учетом того, что минимальный перепад между входом и выходом здесь должен составить не менее 4–5 В, а если используется стабилитрон, как на рис. 9.12, то напряжение на входе должно быть не меньше 12 В). Есть и более изощренные способы — скажем, регулировать действующее значение выпрямленного пульсирующего напряжения перед фильтром с помощью тиристорного моста. Но в таком случае схема настолько усложняется, что проще просто взять и построить импульсный источник (см. далее).

И, наконец, несколько слов про основного нашего героя — операционный усилитель. Здесь указан классический ОУ типа μA741, который выпускается уже много десятков лет, и приведена его нумерация выводов (цоколевка). У него есть и отечественный аналог — КР140УД7 (учтите на будущее, что отечественные аналоги западных микросхем не всегда имеют ту же цоколевку, так что это на всякий случай надо проверять). Вообще же можно взять почти любой ОУ широкого применения с надлежащим допустимым питанием, — но эти подробности мы будем рассматривать уже в *главе 12*.

В заключение этой темы — еще два слова о регулируемом двухполярном лабораторном источнике. Нет никакого смысла изобретать его специально — надо просто взять два одинаковых однополярных источника, разместить их в одном корпусе (и даже запитать их от одного трансформатора, но обязательно от разных вторичных обмоток) и вывести наружу все четыре выходные клеммы по отдельности. Соединяя «плюс» одного источника с «минусом» другого перемычкой, вы получаете общую «землю» двухполярного источника, убирая перемычку — имеете два раздельных однополярных.

Рассеивание тепла

Сразу скажем — теоретической методики для расчета охлаждающих радиаторов не существует. По этому поводу можно написать не одну диссертацию или монографию (и написаны, и много), но стоит изменить конфигурацию охлаждающих ребер или стержней, расположить радиатор не вертикально, а горизонтально, приблизить к нему любую другую поверхность снизу, сверху или сбоку, — все изменится, и иногда кардинально. Именно поэтому производители микропроцессоров или процессоров для видеокарт предпочитают не рисковать, а снабжать свои изделия радиаторами с вентилятором — принудительный обдув, даже слабенький, повышает эффективность теплоотвода в десятки раз, и, хотя нередко это совершенно не требуется, но они поступают по закону «лучше перебдеть, чем недобдеть», и это правильно. Здесь мы приведем только пару эмпирических способов, которые оправдали себя на практике и годятся для того, чтобы рассчитывать пассивные (т. е. без обдува) радиаторы для усилителя из *главы 8* или для линейных источников питания из текущей главы.

Сначала посмотрим, как рассчитывать площадь радиаторов, исходя из их геометрии. На рис. 9.14 схематично показан типичный пластинчатый радиатор. Для расчета его площади нужно к площади его основания прибавить суммарную площадь его ребер (также с каждой стороны). Если нижней стороной радиатор прижимается к плате, то лучше считать рабочей только одну сторону основания, но мы предположим, что радиатор «висит в воздухе» (как часто и бывает) и поэтому площадь основания удваивается: $S_{осн} = 2 \cdot L_1 \cdot L_2$. Площадь одного ребра (тоже с двух сторон): $S_р = 2 \cdot L_1 \cdot h$, но к этой величине нужно еще прибавить боковые поверхности ребра, площадь которых равна $S_{бок} = 2 \cdot h \cdot \delta$. Ребер всего 6, поэтому общая площадь радиатора равна: $S = S_{осн} + 6 \cdot S_р + 6 \cdot S_{бок}$. Пусть $L_1 = 3$ см, $L_2 = 5$ см, $h = 3$ см, $\delta = 0{,}2$ см, тогда общая площадь такого радиатора 145 см^2. Разумеется, это приближенный расчет (мы не учли, скажем, боковую поверхность основания), но для наших целей высокая точность и не требуется.

Вот два эмпирических способа для расчета рассеиваемой мощности в зависимости от площади поверхности, и пусть меня не слишком строго осудят за то, что никаких особенных научных выкладок вы здесь не увидите.

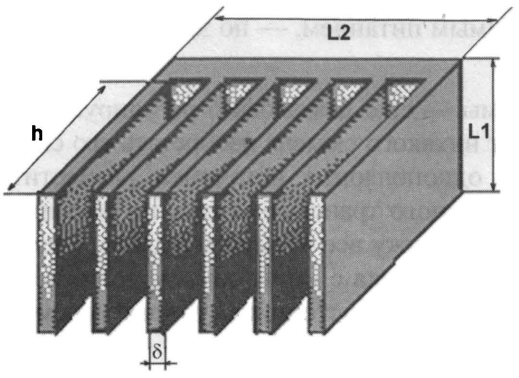


Рис. 9.14. Типичный пластинчатый радиатор

Способ первый и наипростейший — площадь охлаждающего радиатора должна составлять 10 см^2 на каждый ватт выделяющейся мощности. Так что радиатор с приведенными на рис. 9.14 размерами, согласно этому правилу, может рассеять 14,5 Вт мощности — как раз под наш усилитель с некоторым запасом. И если вас не жмут размеры корпуса, то вы вполне можете ограничиться этим прикидочным расчетом.

Если же вы хотите подсчитать поточнее, то вот один из более сложных способов, который годится для радиаторов средних размеров (L_1 = 20–180 мм, L_2 = 40–125 мм).

Для оценки тепловой мощности радиатора можно использовать формулу:

$$W = \alpha_{эфф} \cdot \theta \cdot S,$$

где:

- W — мощность, рассеиваемая радиатором, Вт;
- $\alpha_{эфф}$ — эффективный коэффициент теплоотдачи, Вт/м^2·°С (см. график на рис. 9.15);
- θ — величина перегрева теплоотдающей поверхности, °С, $\theta = T_c - T_{o.c.}$ (T_c — средняя температура поверхности радиатора, $T_{o.c.}$ — температура окружающей среды);
- S — полная площадь теплоотдающей поверхности радиатора, м^2.

Обратите внимание, что площадь в эту формулу подставляется в квадратных метрах, а не сантиметрах.

Итак, приступим: сначала зададимся желательным перегревом поверхности, выбрав не слишком большую величину, равную 30 °С. Грубо говоря, можно считать, что

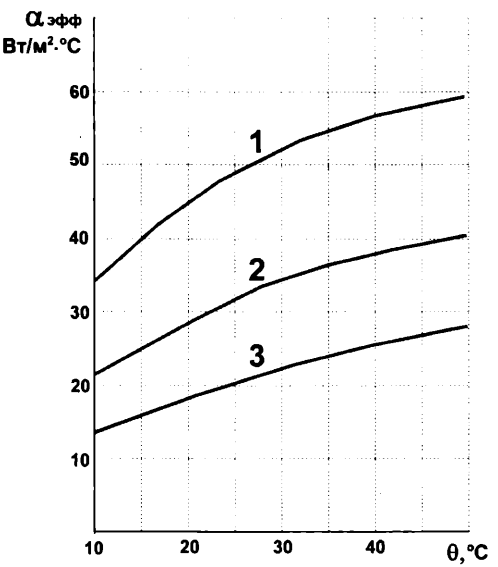


Рис. 9.15. Эффективный коэффициент теплоотдачи ребристого радиатора в условиях свободной конвекции при различной длине ребра: 1 — h = 32 мм; 2 — h = 20 мм; 3 — h = 12,5 мм

при температуре окружающей среды 30 °С, температура поверхности радиатора составит 60 °С. Если учесть, что разница между температурой радиатора и температурой кристалла транзистора или микросхемы при хорошем тепловом контакте (о котором далее) может составить примерно 5 °С, то это приемлемо для практически всех полупроводниковых приборов. Высота ребер h у нас составляет 30 мм, поэтому смотрим на верхнюю кривую из графика на рис. 9.15, откуда узнаем, что величина коэффициента теплоотдачи составит примерно 50 Вт/м^2·°С. После вычислений получим, что W = 22 Вт. По простейшему правилу ранее мы получили 14,5 Вт, а сейчас, проведя более точные расчеты, мы можем несколько уменьшить площадь, тем самым сэкономив место в корпусе. Однако повторим, если место нас не жмет, то лучше всегда иметь запас.

Радиатор следует располагать вертикально, и ребра также должны располагаться вертикально (как на рисунке), а поверхность его нужно покрасить в черный цвет. Красить следует тонким слоем краски из аэрозольного баллончика или распылителя — толстый слой даст дополнительное тепловое сопротивление. Площадки, куда прижимаются транзисторы или микросхемы, перед покраской необходимо защитить кусочком малярного скотча.

Я еще раз хочу напомнить, что все эти расчеты очень приблизительны, и даже сама методика может измениться, если вы поставите радиатор не вертикально, а горизонтально, или ребра у него будут игольчатые вместо пластинчатых. К тому же, мы никак не учитываем здесь тепловое сопротивление переходов кристалл-корпус и корпус-радиатор (просто предположив, что разница температур составит 5 °С).

Тем не менее, указанные методы дают хорошее приближение к истине, но если мы не обеспечим хороший тепловой контакт, все наши расчеты могут пойти насмарку. Просто плотно прижать винтом транзистор к радиатору, конечно, можно, но эффективно это будет только в том случае, если поверхность радиатора в месте прижима идеально плоская и хорошо отшлифована. Практически так никогда не бывает, поэтому радиатор в месте прижима покрывают специальной теплопроводящей пастой. Ее можно купить в магазинах, а иногда тюбик с такой пастой прикладывают к «кулерам» для микропроцессоров. Наносить пасту надо тонким, но равномерным слоем, не перебарщивать в количестве. Если на один радиатор ставятся два прибора, у которых коллекторы находятся под разным напряжением, то под корпус нужно проложить изолирующую прокладку, под крепежные винты — изолирующие пластиковые шайбы, а на сами винты надеть отрезок изолирующей кембриковой трубки длиной, равной толщине радиатора в месте отверстия (рис. 9.16).

> ***Внимание!***
>
> Транзистор надо изолировать от корпуса и в том случае, если радиатор с транзистором находится вне корпуса прибора.

Самые лучшие изолирующие прокладки — слюдяные, очень хороши также прокладки из анодированного алюминия (но за ними надо внимательно следить, чтобы не процарапать тонкий слой изолирующего окисла) и из керамики (которые, впрочем, довольно хрупки и могут треснуть при слишком сильном нажиме). Кстати, за неимением фирменных прокладок можно использовать тонкую фторопластовую

(но не полиэтиленовую, разумеется!) пленку, следя за тем, чтобы ее не прорвать. При установке на прокладку теплопроводящая паста наносится тонким слоем на обе поверхности: и на транзистор, и на радиатор.

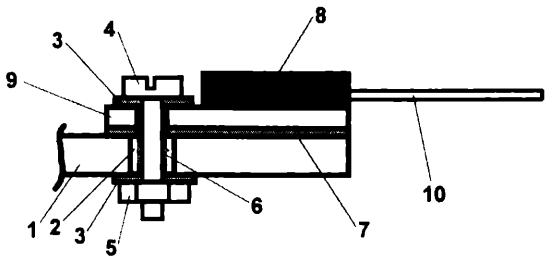


Рис. 9.16. Крепление транзистора в корпусе ТО-220 к радиатору при необходимости его изоляции: 1 — радиатор; 2 — отверстие в радиаторе; 3 — изолирующие шайбы; 4 — стягивающий винт; 5 — гайка; 6 — изолирующая трубка; 7 — слюдяная прокладка; 8 — пластмассовая часть корпуса транзистора; 9 — металлическая часть корпуса транзистора; 10 — выводы транзистора

Принудительное охлаждение и элементы Пельтье

Тема принудительного охлаждения слишком обширна, чтобы ее можно было сколько-нибудь подробно осветить в этой книге, — тем более, что в радиолюбительской практике интенсивное охлаждение требуется редко. Потому мы опустим темы про обдув, а также про использование тепловых трубок и прочую подобную экзотику, остановимся кратко лишь на применении *элементов Пельтье*. Эта тема заслуживает нашего внимания хотя бы просто как иллюстрация к принципу действия одного из самых интересных электронных приборов. Кстати, элементы Пельтье могут использоваться не только для охлаждения, но и для подогрева (простым изменением направления тока), т. е. с их помощью можно построить идеальный термостат, пригодный для любых условий внешней среды. И, что самое интересное, — они пригодны для выработки электроэнергии в небольших количествах, о чем несколько слов далее.

Эффект поглощения и выделения тепла в месте контакта разнородных металлов при прохождении электрического тока был открыт в 1834 году французским часовщиком Жаном Пельтье. Интересно, что противоположный эффект (Зеебека) — возникновения ЭДС в замкнутой цепи при контакте разнородных металлов, находящихся при различных температурах, был открыт на 13 лет раньше, но тогда еще физики не знали, что любой подобный эффект обратим, и предсказать эффект Пельтье не смогли. Как это было почти со всеми такими физическими явлениями (тензоэффект, пьезоэффект), вторую жизнь эффект Пельтье обрел с появлением полупроводников, где он проявляется гораздо сильнее. Современные элементы Пельтье изготавливаются из теллурида висмута с дозированными присадками селена и сурьмы.

Для того, чтобы правильно применять элементы Пельтье в системах охлаждения (или, неважно, подогрева), надо хорошо представлять себе, как они работают и ка-

ковы их ключевые характеристики. Потому давайте для начала проведем небольшой ликбез на эту тему.

Для тепловых насосов (к которым относится и домашний холодильник, и элемент Пельтье) понятие КПД неприменимо — отношение полезной работы к затраченной у них зависит от условий работы. Если кому будет понятней, можно провести такую аналогию: возьмите АА-батарейку и замкните контакты накоротко. Какой КПД у батарейки в таких экстремальных условиях? Очевидно, он равен нулю: потенциалы контактов равны, вся энергия расходуется на подогрев самой батарейки. Будет он равен нулю и в противоположном случае — когда батарейка просто лежит на столе, и тока в цепи нет (ну, или почти нет, — какая-то часть энергии всегда уходит на саморазряд).

Для элементов Пельтье картина аналогичная, только в роли напряжения на контактах выступает разность температур, а в роли тока — количество переданного тепла. Если замкнуть между собой пластины элемента массивным куском меди, то разница температур будет равна нулю, а количество поглощаемого тепла — максимально. Это максимальное количество тепла (*максимальная холодопроизводительность*) обозначается $Q_{макс}$ и служит одной из характеристик элемента. Если сделать наоборот — максимально изолировать пластины друг от друга и от внешней среды (например, поместив их в безвоздушное пространство), то количество переданного тепла будет равно нулю, а разность температур максимальна. Эта величина — тоже одна из главных характеристик элемента, и обозначается $T_{макс}$.

Типичный график в координатах температура-холодопроизводительность для реального элемента Пельтье размером 40×40 мм, поступающего в продажу под названием FROST-72, показан на рис. 9.17. Там же приведены максимальные значения электрических параметров этого элемента (для которых составлен верхний график). Как показывает нижняя линия, при меньших величинах напряжения питания прямая сдвинется вниз, т. е. тепловые показатели упадут. Из этих параметров можно подсчитать максимальную эффективность элемента (не путать с холодильным коэффициентом и, тем более, с КПД): потребляемая электрическая мощ-

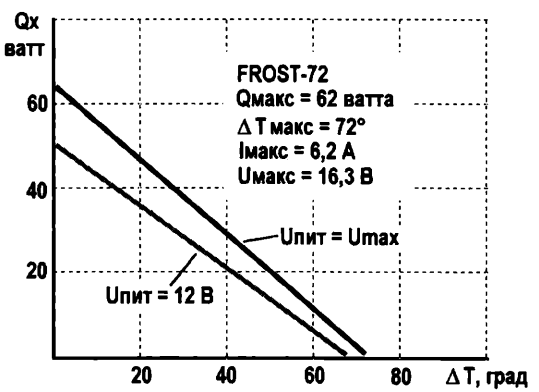


Рис. 9.17. Зависимость холодопроизводительности Q_x от разности температур пластин для элемента FROST-72

ность составит 16,3×6,2 ≈ 100 ватт, т. е. максимальная эффективность будет численно равна максимальной холодопроизводительности и составит 62%.

Подробности

Для идеального холодильника, работающего по циклу Карно: КПД = $\Delta T/T_1$ (T_1 — абсолютная температура горячего источника), а холодильный коэффициент = $T_2/\Delta T$ (T_2 — абсолютная температура холодного приемника), т. е. холодильный коэффициент обычно больше единицы.

Из этих данных легко рассчитать охладитель с элементом Пельтье, скажем, для компьютерного процессора. Если процессор в максимуме производительности выделяет, например, 50 ватт тепла, то единичный элемент типа FROST-72 с ним просто не справится, — при 12 вольтах питания он окажется близок к нулевому перепаду температур. В этом случае придется ставить два элемента, причем если их поставить параллельно рядом (теплораспределительная прокладка должна быть очень хорошей!), то на два следует умножать максимальное количество тепла, а если последовательно друг над другом, то удвоится максимальный перепад температур. Построив соответствующую прямую аналогично рис. 9.17, можно прикинуть, в каком режиме будут работать элементы и какой перепад температур они примерно обеспечат.

При этом стоит учитывать, что ошибиться в холодную сторону тоже не слишком хорошо, — вспомните, что современные процессоры могут самостоятельно менять потребление в зависимости от нагрузки. И если вы рассчитали элемент на перепад температур, допустим, в 40 градусов при 40 ваттах отводимого тепла, то при снижении до 10 ватт температура у вас запросто залезет в минусовые значения. Кстати, это и необязательно: точка росы при среднемосковской влажности летом на улице в 70% достигается уже при 12° тепла. И вы имеете большой шанс залить материнскую плату конденсатом из воздуха в совершенно нормальном режиме работы. Потому обязательным компонентом охлаждающей системы с элементом Пельтье будет отдельный контроллер, который следит за температурой и регулирует мощность, подводимую к элементу, — т. е. даже в таком простейшем случае мы приходим к конструкции термостата.

Заставить элемент Пельтье вырабатывать электричество теоретически тоже несложно — для этого надо обеспечить нужный перепад температур (как можно больший и как можно более стабильный). На практике это условие, однако, может вырасти в серьезную проблему. Эксперименты показывают, что реальная выходная мощность, которую можно получить от подобной конструкции не запредельных габаритов и стоимости, составляет несколько ватт. Так что ноутбук запитать таким образом не получится, а вот подзарядить мобильник вполне реально.

Импульсные источники питания

Главное преимущество *импульсных источников* — экономичность и значительно лучшие массогабаритные характеристики по сравнению с трансформаторными источниками. Поэтому практически все современные бытовые приборы: компьютеры, телевизоры, музыкальные центры и т. д. — снабжаются именно такими источника-

ми. Главным же недостатком их является сложность конструкции и вытекающая отсюда высокая стоимость, из-за чего импульсные источники ранее было целесообразно применять для относительно мощных приборов с энергопотреблением 50–100 Вт и выше. Ныне комплектующие подешевели, ассортимент их стал разнообразнее, и импульсные источники начали все решительнее теснить традиционные, — так, например, зарядные устройства для фирменной мобильной электроники чаще всего делаются именно на импульсных источниках.

Качественный импульсный источник самостоятельно соорудить довольно сложно. Базовая схема промышленного импульсного стабилизатора, подобного тем, что используются в источниках питания бытовой техники, приведена на рис. 9.18. Она слишком громоздка для того, чтобы воспроизводить ее в деталях, а для повторения «на коленке» не годится, — неоправданно трудоемка. Однако в компьютерном блоке питания рано или поздно приходится ковыряться, наверное, каждому радиолюбителю, и для того, чтобы вы понимали, как это работает, опишем устройство этой схемы.

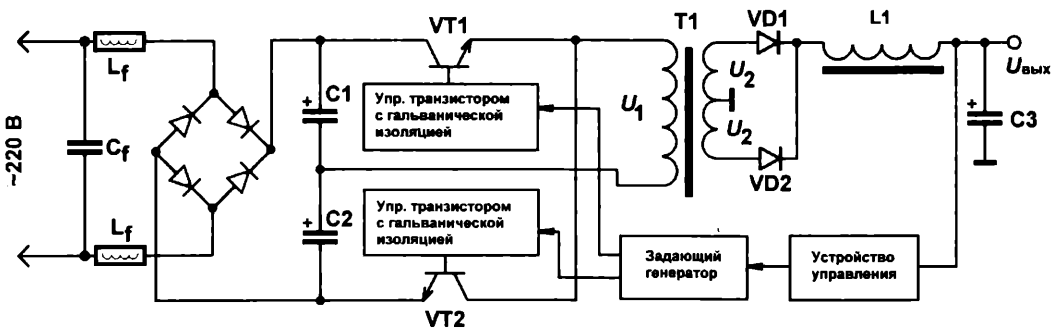


Рис. 9.18. Устройство промышленного импульсного стабилизатора

Здесь сетевое напряжение 220 В выпрямляется стандартным мостом, а затем делится пополам с помощью конденсаторов C1 и C2. Ключевые транзисторы VT1 и VT2 попеременно подключают обмотку высокочастотного трансформатора T1 на ферритовом сердечнике то к плюсу входного напряжения, то к минусу. Все эти элементы должны быть рассчитаны как минимум на половину амплитудного значения сетевого напряжения (т. е. на 160–170 В, если с некоторым запасом). Напряжение со вторичной обмотки выпрямляется по стандартной схеме двухполупериодного выпрямителя на двух диодах (сравните с мостовыми схемами на рис. 9.6 и 9.8, где диодов четыре). Выходное напряжение сглаживается *LC*-фильтром. Оно поступает на устройство управления, где сравнивается с заданным. Устройство это управляет включением генератора импульсов, который, в свою очередь, управляет ключевыми транзисторами. В качестве гальванической развязки обычно используют малогабаритные трансформаторы. Входной фильтр из двух дросселей L_f и конденсатора C_f служит для защиты внешней сети от помех.

Обычная частота работы таких устройств — 10–30 кГц (малогабаритные импульсные источники могут работать и на более высокой частоте). При такой частоте трансформатор на небольшом ферритовом кольце (30–40 мм в диаметре) может

передать десятки и даже сотни ватт мощности. КПД таких источников может достигать 60–80%, вход и выход гальванически изолированы. Основные потери обусловлены рассеиванием тепла на ключевых транзисторах из-за их недостаточного быстродействия, а при малых выходных напряжениях еще и потерями за счет прямого падения напряжения на диодах VD1 и VD2.

Более реально самостоятельно соорудить преобразователь постоянного напряжения (DC) в постоянное другой величины (преобразователь DC-DC) или в переменное (DC-AC, его еще часто называют *инвертором*). На рис. 9.19 приведена схема импульсного преобразователя с гальванической развязкой входа и выхода и стабилизацией выходного напряжения, пригодная для самостоятельного повторения. Эта конструкция в данном случае преобразует входное напряжение +9 В в два высоких напряжения ±165 В. Я специально рассмотрел такой крайний случай — и далее покажу, как изменением всего нескольких параметров схемы получить на выходе практически любую пару симметричных напряжений. Общая максимальная мощность схемы — приблизительно 4 Вт (при том же выходном напряжении максимальный нагрузочный ток составит до 12 мА по каждому из выходов).

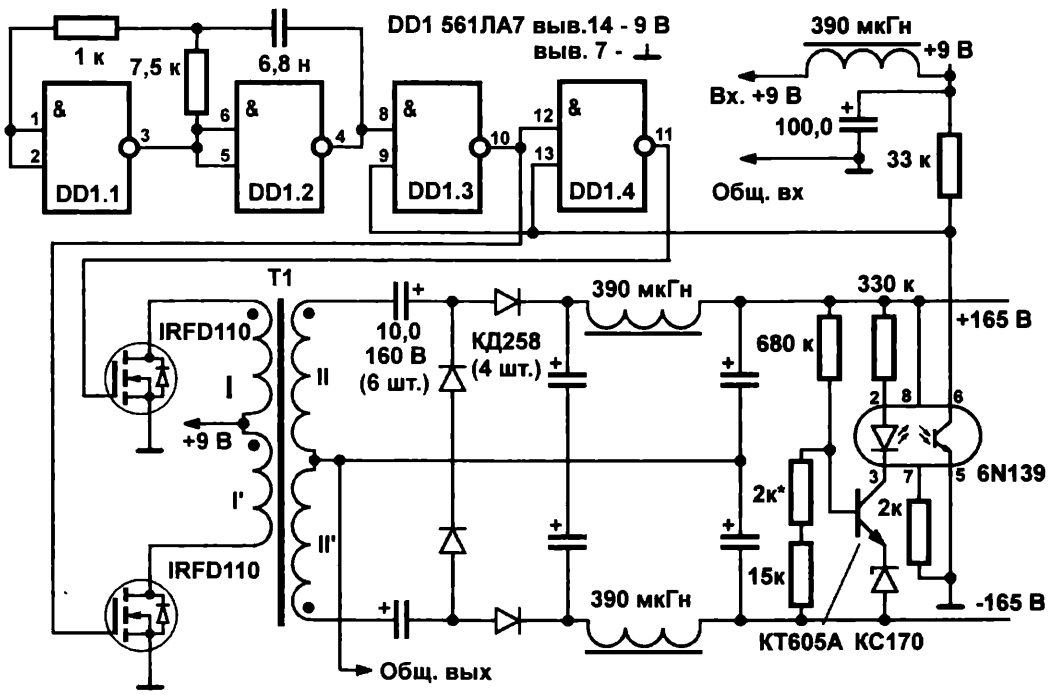


Рис. 9.19. Схема импульсного преобразователя с гальванической развязкой входа и выхода

Рассмотрим работу схемы. Единственный компонент, который мы еще не «проходили», — это логическая КМОП-микросхема К561ЛА7 (для уменьшения габаритов ее можно заменить на аналог 564ЛА7 в планарном корпусе). О таких микросхемах пойдет речь в *главе 15*, а генератор прямоугольных импульсов, который на этой микросхеме построен, будет рассмотрен в *главе 16*.

На выходе элементов DD1/3 и DD1/4 образуются противофазные прямоугольные импульсы, которые поочередно с частотой примерно 10 кГц открывают транзисторные ключи. В результате на вторичных обмотках трансформатора образуется высоковольтное напряжение, которое дополнительно умножается вдвое на системе из диодов КД258, конденсаторов 10 мкФ и индуктивностей (дросселей) 390 мкГн.

Стабилизирующая часть схемы построена на оптроне 6N139, который имеет внутри довольно сложную конструкцию, но практически представляет собой транзисторный оптрон: подавая на вход (выводы 2, 3) напряжение, мы открываем гальванически развязанный от входа транзистор, и тогда на выходе (вывод 6) получаем напряжение, практически равное нулю.

В результате все вместе работает так: если выходное напряжение схемы недопустимо повысилось, то ключ на транзисторе КТ605А открывается, на выходе оптрона появляется близкое к нулю напряжение, логические элементы DD1/3 и DD1/4 при этом запираются, и на ключи ничего не подается. Напряжение на выходе снижается, ключ КТ605А запирается, напряжение на выходе оптрона становится близким к напряжению питания, и импульсы опять поступают на трансформатор. Вместо оптрона 6N139 без изменений в схеме можно использовать 6N135, 6N136 (эти даже лучше — они более быстродействующие) или 6N138.

Таблица 9.2. *Характеристики трансформатора T1*

Сердечник	**Кольцо М2000НМ 20×16×6**		
Обмотки I и I′	10 витков	∅0,3 ПЭВ-2	Размещать на разных секциях кольца без перекрытия
Обмотки II и II′	130 витков	∅0,2 ПЭВ-2	

Трансформатор намотан на ферритовом кольце с характеристиками, указанными в табл. 9.2. Мотаются обмотки медным обмоточным проводом ПЭВ-2 парами совместно, причем, обратите внимание, что у входной пары обмоток соединен конец одной с началом другой, а у выходной — начала обеих обмоток. С помощью подбора дополнительного резистора 2 кОм (на схеме он помечен звездочкой) выходное напряжение устанавливается более точно. Дроссель по питанию +9 В (390 мкГ) служит для защиты внешних сетей от помех. Учтите, что схема довольно заметно «фонит» в радиодиапазоне, потому ее надо заключать в металлический экран, который должен быть соединен со входной (обозначенной на схеме, как «Общ. вх») «землей» в одной точке, вблизи входного контакта на плате.

Малогабаритные MOSFET-транзисторы IRFD110 можно заменить на любые другие с пороговым напряжением затвора 2–4 вольта, а если задаться целью получить больший ток на выходе, то необходимы будут более мощные в корпусе TO-220 (IRFZ40, IRFZ44, IRF530 и др.). Реально это устройство при указанных на схеме элементах работает приблизительно от 7 до 12 В входного напряжения (при этом выходное остается равным номинальному с точностью примерно 2,5%).

Если нужно снизить входное напряжение до 5 В, то пороговое напряжение должно быть на уровне 1,5–2 В (IRLU8256 или аналогичные — подробнее о выборе

MOSFET-ключей см. *главы 6* и *22*). Вместо логической КМОП-микросхемы К561ЛА7 (К564ЛА7) тогда лучше использовать более быстродействующую и мощную 74HC00, только разводка выводов будет немного иная (см. *главу 15*). Если требуется еще и повышенная мощность (теоретически 20-мм кольцо 2000НМ может вытянуть до 50–60 ватт мощности!), следует дополнительно умощнить выходы генератора буферными усилителями на микросхемах 74HC04, 74HC4049 (шесть инверторов) или 54HC4050 (шесть буферных повторителей), а еще лучше 74AC04 или 74AC34 (серия AC вдвое быстрее, чем HC). В каждый из выходов подключается по три инвертора или повторителя, включенных параллельно по входам и выходам, а в цепь затвора транзисторов последовательно вводится резистор 15–20 Ом (см. для образца схему на рис. 22.2, *б* в *главе 22*).

Вместо MOSFET-транзисторов можно также использовать обычные биполярные транзисторы (лучше включенные по схеме Дарлингтона), однако КПД при этом снизится. При изменениях пределов входного напряжения схемы не забудьте пересчитать витки первичной обмотки пропорционально снижению питания.

Для того чтобы изменить выходное напряжение, следует, прежде всего, изменить коэффициент резистивного делителя в базе ключа на КТ605А. При этом, конечно, надо снижать номинал верхнего по схеме резистора (680 кОм), а не повышать нижнего (15 кОм). Например, при выходном напряжении ±24 В номинал верхнего резистора должен составлять примерно 75–82 кОм. Но для хорошей работы преобразователя этого изменения недостаточно — для получения максимального КПД следует также изменить число витков во вторичных обмотках и, насколько возможно, увеличить толщину провода, из которого они наматываются. Рассчитывать обмотки следует так: желаемое выходное напряжение умножить на коэффициент 1,3, затем полученную величину поделить на 9 (входное напряжение) и умножить на 10 (число витков в первичной обмотке). Например, при ±24 В выходного напряжения число витков в каждой из вторичных обмоток должно быть равно 35. При пониженном выходном напряжении можно упростить схему, убрав умножитель напряжения (удалить последовательно включенные конденсаторы 4,7 мкФ, подключив диоды аналогично VD1 и VD2 на схеме рис. 9.13 и увеличив вдвое число витков вторичной обмотки), при этом КПД повысится. Ток при выходном напряжении ±24 В может составить до 120–150 мА, при ±7,5 В — до 500 мА по каждому из напряжений.

Подробности

Зачем в схеме обсуждаемого преобразователя вообще умножитель напряжения? Если вы проанализируете процессы, происходящие в трансформаторе, то обнаружите, что действующее значение напряжения на первичной обмотке равно напряжению питания — т. е. 9 В. Соответственно, чтобы получить после выпрямления и фильтрации значение напряжения 165 В, нам понадобилось бы как минимум $10 \cdot 165/9 \approx 180$ витков в каждой вторичной обмотке, а с запасом на потери и регулирование примерно на 20–30% больше — около 240. Такое количество витков (в сумме около 500) намотать на кольце диаметром 20 мм физически сложно. А когда мы снижаем величину выходного напряжения, число витков уменьшается, и умножитель, который отрицательно сказывается на КПД устройства, можно убрать.

Главным недостатком рассматриваемой схемы с точки зрения КПД, однако, является не умножитель, а форма сигнала на первичных обмотках. Так как включение

одного ключа и выключение другого совпадают во времени, существует момент, когда через обе обмотки течет сквозной ток. Это очень плохо сказывается на КПД и ведет к излишним потерям на нагревание транзисторов. Для небольших мощностей, как здесь, этим эффектом можно пренебречь, но для больших его приходится учитывать и разносить моменты включения одного ключа и выключения другого во времени. Это делается обычно с помощью специализированных микросхем для управления ключами (например, известной TL494), хотя несложно сымитировать их на любом микроконтроллере.

Добавим, что по схеме, аналогичной приведенной на рис. 9.19, построены многие фирменные преобразователи напряжения с развязкой входа и выхода. Такие готовые модули на самые различные напряжения (в том числе и с гальванической развязкой между входом и выходом), например, выпускает фирма TRACO. Фирмой RECOM выпускаются импульсные преобразователи-стабилизаторы напряжения — аналоги 78-й серии под названием R-78. Их КПД достигает 96% при входном напряжении до 34 В, а единственный недостаток — рекомендованное минимальное потребление 10 мА, ниже этого значения стабилизатор может отказывать. Главным недостатком всех импульсных стабилизаторов является относительно большой (не менее 5–10 мА) ток покоя, потому в малопотребляющих устройствах они непригодны.

Заметки на полях

В интернет-магазинах, торгующих товарами для радиолюбителей, можно встретить недорогой модуль импульсного преобразователя напряжения с отличными характеристиками. Он сделан на основе регулируемой микросхемы LM2596 фирмы Texas Instruments и обычно встречается под этим названием. Модуль позволяет получить из входного напряжения от 3 до 40 вольт выходные напряжения от 1,5 до 35 вольт при токах до 3 А, причем с высокой эффективностью: от 70% (при входном напряжении 5 В и выходном 3,3 В) до 90–95% (при входном 35 В и выходном 12–20 В). На миниатюрной плате модуля уже смонтирован подстроечный резистор, позволяющий устанавливать желаемое выходное напряжение. Модуль особенно удобен в случаях, когда нужно получить напряжение питания 3–5 В из достаточно высокого напряжения 12 или 24 вольта, поступающего, например, от буферных аккумуляторов источника на основе солнечной батареи. Обычные линейные стабилизаторы в этих условиях большую часть энергии бесполезно рассеивают в виде тепла, к тому же требуя радиаторов значительных габаритов, а подобный источник и сам почти не греется, и энергию зазря не расходует.

Схему на рис. 9.19 очень легко переделать в преобразователь-инвертор из автомобильных 12 В постоянного тока в 220 В переменного. Один из вариантов такой схемы показан на рис. 9.20. Здесь генератор построен на элементе, называемом *триггером Шмидта* (подробнее о нем см. *главу 16*). Шесть таких элементов входят в микросхему 74С14, и «лишние» две пары из них служат для управления затворами мощных полевых транзисторов. Частота генератора снижена до ~50 Гц, часть схемы, касающаяся стабилизации, удалена. Трансформатор здесь самый обычный на 50 Гц, с двумя обмотками на 12 В и одной на 220 В, мощностью примерно на 100–120 Вт. Можно купить готовый или пересчитать обмотки для имеющегося под рукой по методике, описанной в этой главе ранее.

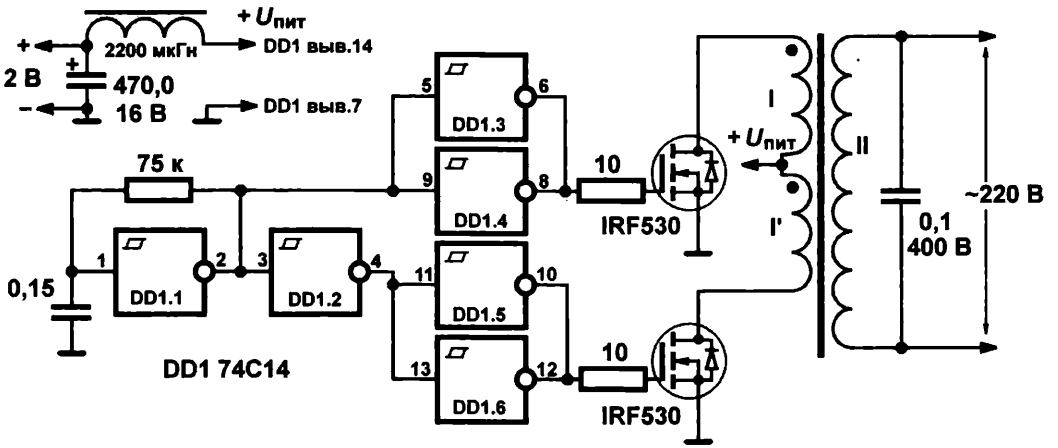


Рис. 9.20. Преобразователь-инвертор =12 В / ~220 В

Выжать из этой схемы можно до тех же 100–120 Вт мощности, на которые рассчитан трансформатор, однако если предполагается ток от источника 12 В более 3–4 А (т. е. мощность более 30–40 Вт), то транзисторные ключи необходимо поставить на радиаторы 100 см^2 и защитить их от возможных высоковольтных выбросов напряжения на обмотках, поставив параллельно выходу каждого транзистора (плюсом к стоку, минусом на «землю») по стабилитрону на напряжение 20–30 В (например, отечественные КС522А1 (КС5227А1), китайские типа BZX55, DL4749A или аналогичные). MOSFET-транзисторы IRF530 можно заменить на более современные IRF3205L или аналогичные с пороговым напряжением 2–4 вольта.

На самом деле подобные преобразователи давно уже стали фактически ненужными, и здесь схема приведена больше в качестве примера того, как можно делать мощные преобразователи такого рода. Вся современная мобильная бытовая техника имеет возможность питания от напряжения 12 вольт. К тому же такой самодельный преобразователь имеет крупный недостаток в виде напряжения прямоугольной формы, что, несмотря ни на какие фильтры по выходу, далеко не для всякой техники годится, — помехи все задушат. А мощный резервный инвертор (источник бесперебойного питания, UPS) с близким к синусоидальной форме напряжением на выходе и автоподзарядкой аккумуляторов от сети — целая интеллектуальная станция с довольно навороченным алгоритмом работы, и для радиолюбителя вещь не столько неподъемная, сколько не окупает затраченных на нее усилий и средств.

Как правильно питаться?

Мы уже слегка коснулись темы правильной разводки питания в *главе 8*, когда рассказывали об усилителе звуковой частоты. Сейчас мы сформулируем несколько общих принципов.

Стандартная схема грамотной разводки питания между источниками и потребителями в электронных устройствах приведена на рис. 9.21, *вверху*. На практике, если источник расположен в отдельном корпусе, то указанной на блок-схеме общей точ-

кой соединения «земли» служит выходная клемма «минус» этого корпуса (кстати, поэтому лучше делать выходные клеммы источника не по одной, а, например, парами, чтобы можно было подключить как минимум две нагрузки к одной точке). Если же вся конструкция — и источники, и нагрузки — представляет собой набор плат в едином корпусе, то за общую точку удобно выбрать минусовый вывод основного фильтрующего конденсатора.

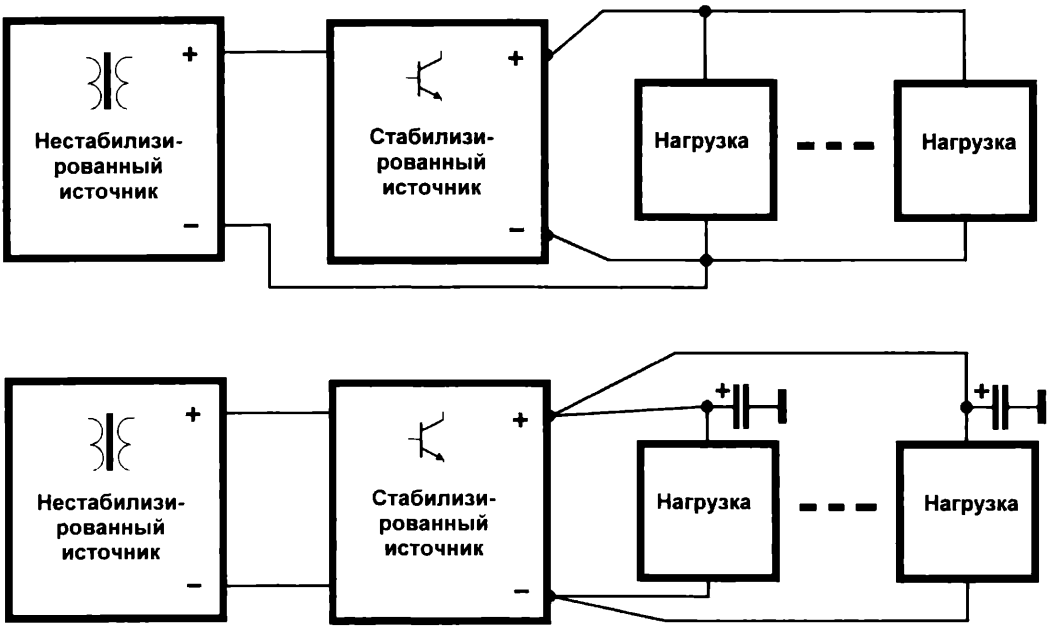


Рис. 9.21. Схема разводки питания между источниками и потребителями

Смысл такой разводки заключается в том, чтобы токи от разных потребителей не протекали по одному и тому же проводу, — это может вызвать их взаимное влияние и всякие другие нежелательные явления. Характерное подобное явление под названием *захват частоты* можно наблюдать, если на двух разных, но с общим питанием, платах имеются генераторы, работающие на близких или кратных частотах, — вдруг по непонятным причинам они начинают работать на одной и той же частоте! Иногда от этого очень трудно избавиться, поэтому лучше сразу делать все правильно.

Обычно по каким-то причинам идеала по образцу рис. 9.21, *вверху*, достичь не получается. В этом случае помогут установленные как можно ближе к выводу питания так называемые *развязывающие* конденсаторы (они как раз и показаны на рис. 9.21, *внизу*). Причем если это отдельная плата, то их ставят на ней прямо около входного разъема, но ни в коем случае не в дальнем конце платы! Кроме того, во всех случаях провода и проводники питания на плате должны быть как можно толще: если провод тонкий, то на нем самом за счет протекающего тока происходит падение напряжения, и разные потребители оказываются под разными потенциалами, — как по «земле», так и по питанию.

Заметки на полях

Кстати, о «земле» — почему я ее все время заключаю в кавычки? Схемотехническую «землю» самое правильное называть общим проводом, просто термин прижился, да и звучит короче. Дело в том, что в электротехнике существует совершенно определенное понятие «земли» — когда нечто находится под потенциалом земной поверхности, который принимается за истинный ноль напряжения. Под таким потенциалом по понятным причинам находятся, например, металлические водопроводные трубы или батареи отопления. Есть еще понятие нулевого провода (один из проводов в вашей домашней розетке всегда нулевой, второй называется фазным) — он теоретически находится тоже под потенциалом земли, но практически соединяется (возможно) с истинной землей только где-то на электростанции, а за счет несбалансированности протекающего по различным фазам тока потенциал его может «гулять», и довольно сильно. Поэтому правильно организованная бытовая электросеть всегда должна включать в себя третий провод, который есть истинное заземление. Если у вас такого третьего провода нет (печально, но в нашей стране до сих пор строили именно так, и только в последние годы положение начинает выправляться), то его можно организовать путем присоединения к металлической водопроводной трубе (СНиПы это допускают). Но это не только неудобно (представляете, сколько проводов придется растаскивать по всей квартире?), но иногда и опасно — в случае попадания фазного напряжения на такое заземление, до тех пор, пока сработает предохранитель, сопротивления между трубой и землей вполне может хватить, чтобы основательно тряхнуть кого-нибудь, кто будет в соседней квартире в этот момент мыть руки под краном. И не стоит забывать, что современную разводку водоснабжения и отопления делают пластиковыми трубами, а они заземлением, конечно, служить не могут.

На рис. 9.22, *а* показана схема развязывающего фильтра для маломощной нагрузки в пределах одного электронного узла. Это может быть входной каскад усиления микрофонного усилителя, который особо чувствителен к качеству питания, и его требуется развязать от следующих более мощных каскадов. На рис. 9.22, *б* показана правильная организация питания с такими фильтрами для быстродействующих или прецизионных измерительных усилителей — в частности, в измерительных схемах, о которых мы будем говорить в следующих главах.

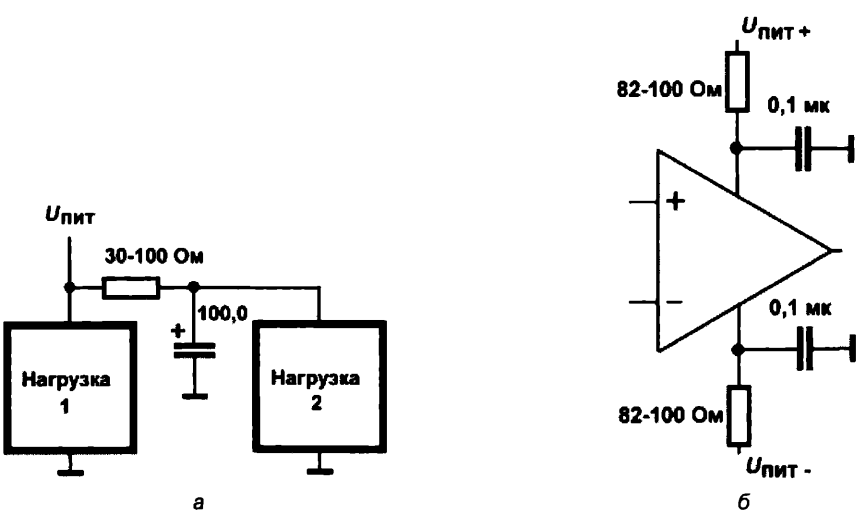
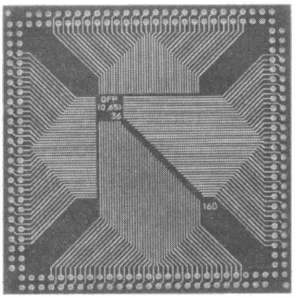


Рис. 9.22. Разводка питания: *а* — схема разделения нагрузок с помощью развязывающего фильтра; *б* — организация питания для быстродействующих и прецизионных усилителей



ГЛАВА 10

Тяжеловесы

Устройства для управления мощной нагрузкой

— Что вы делаете? — с удивлением воскликнула миледи.

— Положите мне руки на шею и не бойтесь ничего.

— Но из-за меня вы потеряете равновесие, и оба мы упадем и разобьемся.

А. Дюма. «Три мушкетера»

Многие практические задачи состоят в том, чтобы маломощное управляющее устройство, например, простой переменный резистор или схема управления, построенная на логических или аналоговых микросхемах, могло бы управлять мощной нагрузкой, как правило, работающей от бытовой электрической сети. Это одна из тех областей техники, где за последние полвека электроника совершила настоящий переворот.

Представьте себе работу, скажем, осветителя в театре еще в середине XX века. Для плавного регулирования яркости прожектора тогда использовался последовательно включенный реостат — проще говоря, регулирование осуществлялось по схеме, приведенной на рис. 1.4. Более экономичный, но и более дорогой и громоздкий вариант, — ставить на каждый прожектор по регулируемому автотрансформатору с ползунком, управляемым вручную. Иногда в таких автотрансформаторах для дистанционного вращения ползунка приспосабливали моторчик, и вся система управления освещением с гудящими трансформаторами, завывающими моторчиками и клацающими реле-пускателями начинала напоминать небольшой цех. То ли дело сейчас, когда осветитель сидит за клавиатурой вроде компьютерной (а иногда и просто за компьютерной) и управляет этим хозяйством легкими движениями пальцев. А нередко — как в массовых театрализованных представлениях — человек оказывается вообще нужен только для контроля, система управляется компьютером по заранее заданной программе. Все это стало возможным только лишь с появлением электронных устройств управления мощными нагрузками.

В *главе 9* мы уже упоминали о том, что электронные устройства ни в коем случае нельзя строить по бестрансформаторной схеме, — так, чтобы органы управления были напрямую связаны с сетью. При построении схем, управляющих сетевой нагрузкой, возникает непреодолимое искушение избавиться от трансформаторов

питания и последующих устройств сопряжения — в самом деле, электричество в конечном счете одно и то же, так, спрашивается, зачем возиться? Но не поленимся повторить: поступать так не следует, потому что это **опасно для жизни**. И не только вашей жизни, которая подвергнется опасности при отладке подобных устройств, но и для жизни тех, кто будет вашими устройствами пользоваться. Тем не менее, здесь вы найдете некоторые исключения из этого правила, — они касаются случая, когда управление сетевой нагрузкой осуществляется в автоматическом режиме, и доступ людей к элементам схемы во время ее работы исключен.

Самая простая схема управления мощной нагрузкой — *релейная*. Она применима в тех случаях, когда нагрузку нужно просто включать и выключать. Мы не будем подробно останавливаться на этом случае, т. к. о реле достаточно сказано в *главе 7*. Однако отметим один существенный момент, о котором мы ранее не упоминали, — дело в том, что при релейном управлении сетевая нагрузка может отключаться и включаться, естественно, в произвольный момент времени. В том числе, этот момент может попадать и на самый пик переменного напряжения, когда ток через нагрузку максимален. Разрыв — или соединение — цепи с большим током, как мы уже знаем (см. *главы 5* и *7*) приводит к разного рода неприятностям. Во-первых, это искрение на контактах из-за выброса напряжения, что ведет к их повышенному износу, во-вторых и в-главных, это создает очень мощные помехи, причем как другим потребителям в той же сети, так и электромагнитные помехи, распространяющиеся в пространстве. В моей практике был случай, когда включение мощного двигателя станка через пускатель приводило к тому, что в микроконтроллере, установленном в блоке управления на расстоянии пяти метров от станка, стиралась память программ! И это несмотря на то, что все стандартные меры по защите от помех по питанию были приняты.

Чтобы избежать такой ситуации, для коммутации мощной нагрузки лучше применять не обычные электромагнитные реле или пускатели, а оптоэлектронные. Чтобы оправдать соответствующее повышение стоимости, нужно при выборе устройства проверять, имеется ли у него встроенный *zero-детектор*, — схема, которая при получении команды на отключение или включение дожидается ближайшего момента, когда переменное напряжение переходит через ноль, и только тогда выполняет команду.

А теперь перейдем к более интересным вещам — к плавному регулированию мощности в нагрузке. Мы будем это делать, управляя действующим значением напряжения, которое на нее поступает.

Базовая схема регулирования напряжения на нагрузке

Для этой цели нам придется применить электронный прибор, который мы до сих пор не рассматривали, — *тиристор*, представляющий собой управляемый диод и соединяющий в себе свойства диода и транзистора. По схеме включения тиристор несколько напоминает транзистор в ключевом режиме — у него тоже три вывода, которые работают аналогично соответствующим выводам транзистора (рис. 10.1, *а*).

В обычном состоянии тиристор заперт и представляет собой бесконечное сопротивление, а для его открывания достаточно подать напряжение на управляющий электрод — аналог базы у транзистора. Разница между тиристором и транзистором заключается в том, что для удержания транзистора в открытом состоянии через базу нужно все время гнать управляющий ток, а тиристору для открывания достаточно короткого импульса, после чего он остается открытым.

Величина тока через управляющий электрод составляет несколько единиц или десятков миллиампер в зависимости от мощности тиристора — для очень мощных приборов она может составлять единицы ампер (причем в ряде случаев ограничительный резистор можно не ставить — на схеме рис. 10.1, *а* он показан пунктиром). При этом напряжение должно достигать определенной величины — амплитуда управляющих импульсов для тиристоров средней мощности (рассчитанных на токи порядка 3–10 А) должна составлять примерно 5–10 В, а длительность импульса может не превышать 0,05 мс.

В отсутствие открывающего импульса тиристор все равно можно открыть, если подать на анод достаточно высокое напряжение, — ток через управляющий электрод всего лишь снижает это открывающее анодное напряжение практически до нуля (но при этом управляющий импульс также должен иметь напряжение не ниже некоторого порога). Существует даже отдельный класс приборов под названием *динисторы*, представляющих собой тиристоры без управляющего электрода, — они открываются при превышении анодным напряжением определенной величины, которая обычно составляет несколько десятков вольт. Тиристоры могут также открываться самопроизвольно, если анодное напряжение нарастает слишком быстро (со скоростью порядка 10 В/мкс и более). Во избежание этого в схемах на тиристорах следует шунтировать промежуток катод-управляющий электрод резистором (на схеме рис. 10.1, *а* не показан). В настоящее время выпускаются специальные тиристоры и *симисторы* (о них рассказано далее), лишенные этого недостатка и предназначенные для работы в импульсных цепях.

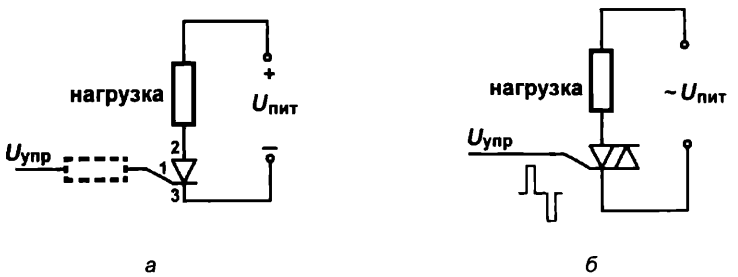


Рис. 10.1. Схемы включения тиристоров и симисторов: *а* — основная схема включения тиристора (1 — управляющий электрод; 2 — анод; 3 — катод); *б* — включение симистора

Как и все диоды, тиристоры выдерживают большие перегрузки по току при условии, что они кратковременны. Во включенном состоянии тиристор ведет себя как обычный диод, а закроется только тогда, когда ток через него (именно через него, в цепи анод-катод, а не по управляющему переходу) снизится до нуля. Если ис-

пользовать его в цепи переменного тока, то это произойдет почти сразу, в конце ближайшего полупериода, при переходе напряжения через ноль. А вот в цепи постоянного тока тиристор сам не отключится, пока через него идет ток. Вообще-то, тиристор можно закрыть и подачей на управляющий электрод импульса противоположной полярности, но практически этим никто не пользуется (и возможность эта для обычных тиристоров относится к числу недокументированных), потому что и напряжение, и ток такого импульса должны быть сравнимы с напряжением и током в силовой цепи анод-катод.

Одиночный тиристор может обеспечить регулирование только положительного напряжения. В сети переменного тока в открытом состоянии он будет работать как диод, отрезая отрицательную полуволну. Чтобы регулировать переменное напряжение в течение обоих полупериодов, нужен еще один тиристор, включенный наоборот. Так как тиристоры во всем, кроме управления, ведут себя подобно диодам, их можно включать встречно-параллельно. Для обычных диодов такое включение применяется только в схемах, подобных показанной на рис. 7.5, — они будут всегда открыты, так что, если не обращать внимания на падение напряжения в 0,6 В, при включении последовательно с нагрузкой такая схема просто ничего не делает. Иное дело тиристоры — если на управляющие электроды ничего не подавать, то нагрузка будет отключена, если же подавать управляющие импульсы в нужной фазе и полярности относительно питающего напряжения, то они будут открываться и подключать нагрузку.

Симметричные тиристоры, или *симисторы* (рис. 10.1, *б*), естественно, выпускаются и отдельно. На западный манер симистор называется *триаком*. В симисторе имеется один управляющий электрод, причем в общем случае знак управляющего напряжения должен совпадать с полярностью на аноде в тот или иной момент времени. Популярные в нашей стране симисторы КУ208 при положительном напряжении на аноде могут включаться импульсами любой полярности, подаваемыми на управляющий электрод относительно катода, а при отрицательном — импульсами только отрицательной полярности.

На осциллограммах (рис. 10.2) перед нами пример управления мощностью в нагрузке с помощью пары встречно-параллельно включенных тиристоров или одного симистора. В начале каждого полупериода тиристор закрыт, управляющий импульс подается только через промежуток времени, равный трети длительности этого полупериода (т. е. со сдвигом фаз, равным $\pi/3$ относительно напряжения питания), и тогда тиристор открывается. Закрывается он, как уже говорилось, автоматически в момент перехода питающего напряжения через ноль.

В результате напряжение на нагрузке будет иметь необычный вид, показанный на графике (см. рис. 10.2, *внизу*). Каково будет действующее значение напряжения? Ясно, что оно будет меньше, чем в отсутствие тиристора, — или чем в случае, если бы управляющий импульс подавался в самом начале периода. Если же, наоборот, подавать управляющий импульс в самом конце, то действующее значение будет близко к нулю. Таким образом, сдвигая фазу управляющих импульсов, мы можем плавно менять мощность в нагрузке с достаточно высоким КПД.

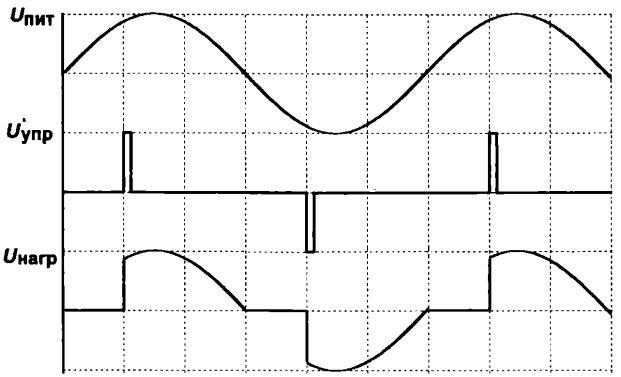


Рис. 10.2. Графики напряжения в схеме фазового управления с помощью пары тиристоров (или симистора)

Мощность в нагрузке при тиристорном управлении

А можно ли вычислить, чему будет равно действующее значение во всех этих случаях? Обычно такие расчеты не требуются, но в некоторых случаях, как мы увидим далее, полезно эту величину знать, т. к. стандартным цифровым мультиметром измерить ее невозможно, — по причинам, указанным в *главе 4*, он покажет для напряжения такой формы все, что угодно, только не истинную величину. Для того чтобы рассчитать величину действующего значения для разных величин сдвига фазы, нужно взять интеграл от квадрата мгновенного значения напряжения в течение всего полупериода. Полученная в результате формула будет выглядеть так:

$$U_д = \sqrt{\frac{{U_a}^2}{\pi}\left(\frac{t}{2} - \frac{\sin 2t}{4}\right)},$$

где:

- $U_д$ — действующее значение напряжения на нагрузке;
- U_a — амплитудное значение питающего напряжения;
- t — определяется по формуле $t = \pi - \varphi$, если сдвиг фазы φ выражать в радианах или по формуле $t = \pi(180 - \varphi)/180$, если сдвиг фазы φ выражать в градусах.

При сдвиге фазы больше, чем половина периода (т. е. $\varphi > \pi/2$), полезно знать также максимальное значение напряжения на нагрузке U_{max}, потому что от этого иногда зависит выбор элементов (при сдвиге фазы меньше половины периода максимальное значение попросту равно амплитудному значению питающего напряжения). Его можно рассчитать по простой формуле: $U_{max} = U_a \sin(\varphi)$.

В табл. 10.1 приведены результаты расчета по этим формулам для синусоидального напряжения 220 В. В последней колонке таблицы указаны величины мощности, которая будет выделяться в нагрузке, в процентах от максимальной мощности, которая выделялась бы при прямом включении нагрузки в сеть или, что то же самое, при сдвиге фазы управляющего импульса, равной нулю.

Таблица 10.1. Действующие и амплитудные напряжения при тиристорном управлении

φ, град	φ, рад	$U_{действ}$	$U_{ампл}$	P, %
0	0,00	219,9	311,0	100,0
10	0,17	219,8	311,0	99,8
20	0,35	218,9	311,0	99,0
30	0,52	216,7	311,0	97,0
40	0,70	212,6	311,0	93,4
50	0,87	206,2	311,0	87,8
60	1,05	197,2	311,0	80,4
70	1,22	185,7	311,0	71,3
80	1,40	171,8	311,0	60,9
90	1,57	155,5	311,0	50,0
100	1,75	137,3	306,3	39,0
110	1,92	117,7	292,2	28,6
120	2,09	97,2	269,3	19,5
130	2,27	76,5	238,2	12,1
140	2,44	56,3	199,9	6,5
150	2,62	37,3	155,5	2,9
160	2,79	20,6	106,4	0,9
170	2,97	7,4	54,0	0,1
180	3,14	0,0	0,0	0,0

Анализ данных таблицы приводит нас к довольно интересным выводам. Зависимость действующего значения напряжения и мощности в нагрузке практически не меняется по сравнению с максимальной вплоть до сдвига фаз, равного 30° (в радианах π/6 или примерно 0,5) — помните из школьной тригонометрии правило: «синусы малых углов примерно равны самому углу»? Это оно и действует. Дальше значения мощности очень быстро падают вплоть до 150–160 градусов, когда мощность становится уже исчезающе малой, — но обратите внимание на величину амплитудного значения! При сдвиге фаз в 160 градусов, когда мощности практически никакой уже нет, амплитудное значение все еще равно аж целых 106 В, — такое напряжение вполне способно вывести из строя, скажем, маломощные диоды, у которых допустимое обратное напряжение часто не превышает нескольких десятков вольт.

Самый важный вывод, который следует из анализа данных таблицы, — изменение мощности в нагрузке в зависимости от угла сдвига фазы происходит нелинейно. По этой причине при проектировании устройств регулирования не имеет смысла начинать регулировку с малых углов сдвига фаз — реально ничего меняться не будет, и

значительная часть хода регулировочного элемента будет холостой, практические изменения начнутся с углов в 30° и более.

Закончив на этом со скучной теорией, перейдем к практическим схемам.

Ручной регулятор мощности

Такое устройство будет незаменимо, скажем, в фотостудии, где используются мощные осветительные лампы: сначала вы уменьшаете яркость до половины, спокойно настраиваете освещение, не заставляя клиента щуриться и обливаться пóтом, затем выводите яркость на полную и производите съемку. Можно его также применить для плавного регулирования мощности нагревателя электроплитки или электродуховки, нагревателя для рукомойника и в других подобных случаях.

Так как устройство предполагает ручное управление, нам надо позаботиться о том, чтобы изолировать орган управления — это будет переменный резистор — от сетевого напряжения. Самое удобное было бы использовать для этого симисторную оптопару — к примеру, MOC2A60-10 фирмы Motorola. Такая оптопара работает совершенно так же, как отдельный симистор, только вход у нее — не управляющий электрод симистора, а светодиод, подобно тому, как это делается в диодных оптронах и оптоэлектронных реле, описанных в *главе 7*. Сами электронные реле, особенно если они содержат упомянутый ранее zero-детектор, использовать в такой схеме невозможно, т. к. никакого фазового управления не получится.

Базовая схема регулятора (диммера)

Но мы попробуем построить схему самостоятельно. Основную управляющую часть будем питать прямо от сети, а вот регулировочный резистор в этом варианте изолируем от нее с помощью оптрона, — только не симисторного, а простого диодного или резисторного, выходное сопротивление которого линейно зависит от входного тока. Обеспечить питание управляющей части схемы при этом можно от любого изолированного от цепи источника (хоть покупного выпрямителя со встроенной вилкой). Конечно, это неудобно, и на практике так никто не делает, но такая универсальная безопасная схема приведена для примера, потому что она позволяет управлять тиристором от любого внешнего низковольтного устройства, — например, компаратора или микроконтроллера, подав с них нужное напряжение на светодиод оптрона (переменник при этом можно исключить, а постоянный резистор придется оставить, подобрав его величину).

Схема регулятора (подобные регуляторы еще называют *диммерами*) приведена на рис. 10.3. Сначала представим себе, что вместо фотодиода оптрона у нас в схеме стоит обычный постоянный резистор. Узел, который включает этот резистор, транзисторы VT1 и VT2, конденсатор C1 и резисторы R3–R6, представляет собой так называемый *релаксационный генератор* на аналоге одноперехoдного транзистора с *n*-базой. Хитрая схема включения разнополярных транзисторов VT1 и VT2 и есть этот самый аналог. Подробно свойства одноперехoдного транзистора мы рассмат-

ривать не будем, потому что за все время моей практики единственное применение для них нашлось только вот в такой схеме релаксационного генератора, причем описываемый тут аналог работает лучше, чем настоящий однопереходный транзистор (КТ117). Подобная схема, кстати, составляет основу IGBT-транзисторов, о которых шла речь в *главе 6*.

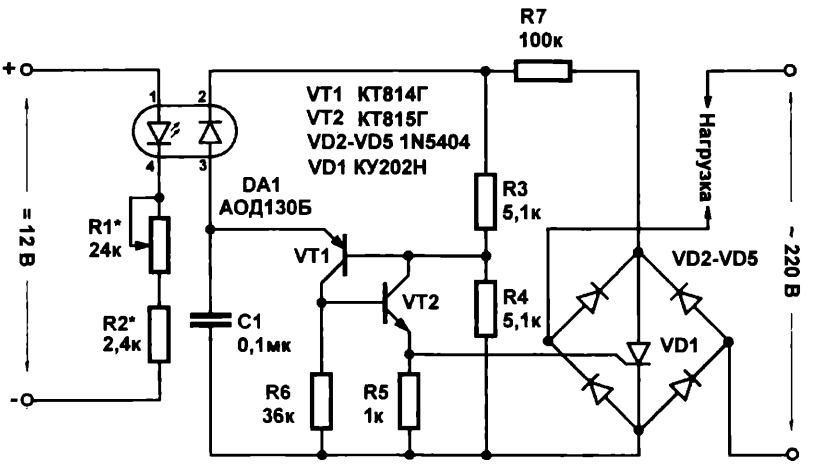


Рис. 10.3. Схема ручного регулятора мощности в нагрузке

Для нас достаточно знать, что такое устройство работает следующим образом: если напряжение на входе (т. е. на соединенных эмиттерах VT1 и VT2) меньше, чем на соединенных базе VT1 и коллекторе VT2 (т. е. на делителе R3–R4), то такой транзистор заперт. Если же напряжение на входе превысит напряжение на делителе R3–R4, то транзистор откроется, причем необычным образом — ток потечет от входа к эмиттеру транзистора VT2 и создаст падение напряжения на резисторе R5. В открытом состоянии он будет, подобно тиристору, пребывать до тех пор, пока ток через него (напряжение на входе) не упадет до нуля.

Резистор R6 нужен для гарантированного запирания транзистора VT2. Предупреждаю: в сети полно подобных схем, где резистор R6 отсутствует, но их авторы, вероятно, ни разу не попытались проверить конструкцию на практике, бездумно копируя схему друг у друга. Когда *p-n-p*-транзистор VT1 заперт (т. е. напряжение на C1 ниже порога делителя R3/R4, см. далее), база VT2 «повисает в воздухе», и его поведение при этом непредсказуемо.

Теперь понятно, как работает генератор: сначала конденсатор заряжается с постоянной времени, обусловленной его емкостью *C* и сопротивлением приемника фоторезистора (обозначим его *R*), и когда напряжение на нем достигнет половины напряжения питания (что обусловлено одинаковостью резисторов R3 и R4), он очень быстро разрядится через открывшийся однопереходный транзистор, резистор R5 и подключенный параллельно с ним управляющий электрод тиристора, формируя импульс включения. Когда напряжение на конденсаторе станет мало́, однопереходный транзистор закроется, и все начнется сначала: конденсатор начнет заряжаться и т. д. Частоту генератора можно оценить по формуле $f = 1/RC$.

А что тиристор? Он теперь останется открытым до очередного перехода сетевого напряжения через ноль, а затем будет ожидать следующего открывающего импульса. Меняя сопротивление фоторезистора, т. е. изменяя входной ток светодиода оптрона, мы можем менять промежуток между открывающими импульсами и тем самым сдвигать их фазу относительно периода сетевого напряжения.

Однако это еще довольно приблизительное описание того, что на самом деле происходит в этой схеме. Внимательный читатель давно заметил, что питание генератора осуществляется прямо от выпрямленного напряжения сети через резистор R7, величина которого подобрана таким образом, чтобы напряжение на элементах схемы даже на максимуме синусоиды не превышало бы примерно 30 В и не вывело бы элементы схемы из строя. Такое пульсирующее питание в этом случае вовсе не просто суровая необходимость — оно крайне полезно.

Все дело в том, что частота любых генераторов с времязадающей RC-цепочкой весьма нестабильна и зависит от множества причин. Если бы мы использовали для питания такого генератора отфильтрованное постоянное напряжение, то установленный нами промежуток между импульсами быстро бы «уехал», и ни о каком стабильном сдвиге фазы и речи бы не шло, — напряжение на нагрузке менялось бы хаотически. В нашем же случае, когда тиристор открывается, он шунтирует мост (ток ограничен током нагрузки), все падение напряжения сети теперь приходится на нагрузку, и напряжение питания генератора снижается почти до нуля (точнее — до утроенного значения падения на диоде). Когда это происходит, однопереходный транзистор, согласно описанному ранее алгоритму, откроется — ведь на входе у него напряжение, накопленное на конденсаторе, и оно рано или поздно превысит небольшое остаточное напряжение на делителе. Причем в конце концов это произойдет, даже если тиристор не откроется вовсе (в схемах без моста в цепи нагрузки, приведенных далее, работа тиристора не оказывает влияния на питание схемы), потому что в конце полупериода напряжение так или иначе упадет. Потому, независимо от того, насколько конденсатор заряжен, он к концу полупериода обязательно разрядится и к началу нового полупериода придет «чистеньким». В конце очередного полупериода тиристор запирается, и с началом следующего генератор опять начинает работать.

Это означает, что схема наша автоматически синхронизируется с частотой сети, и промежуток времени от начала очередного полупериода до возникновения запускающего тиристор импульса (фаза управляющего импульса) будет достаточно стабилен, независимо от внешних условий. Если вдруг вы захотите использовать в этой схеме вместо аналогового генератора микроконтроллер или просто логическую схему, то вам придется тоже обязательно синхронизировать их выходные импульсы с сетевым напряжением. В нашей схеме можно, как это часто делают, ограничить напряжение на элементах схемы управления с помощью стабилитрона (его следует включать параллельно делителю R3–R4), но ни в коем случае не следует дополнительно еще и включать сглаживающий конденсатор.

Если вникнуть в описанный алгоритм работы поглубже, то станет понятно, что при малых углах регулирования (до половины полупериода) генератор может выдать (а в схемах, описанных далее, — и выдаст) за полупериод несколько импульсов, но

это не должно нас смущать, — тиристор запустится с первым пришедшим, а остальные просто сработают вхолостую.

Вот сколько тонкостей скрыто в такой, казалось бы, простой схеме!

Оптрон АОД130Б можно заменить на любой другой диодный оптрон, однако учтите, что отечественные оптроны старых моделей имеют очень небольшое пробивное напряжение изоляции (100–200 В). Впрочем, это критично только в том случае, если регулирующая схема (переменный резистор) гальванически соединена с потенциалом, связанным с сетью, — например, закорочена на корпус, который связан с настоящей землей. Поскольку это маловероятно, то в крайнем случае можно не обращать внимания на этот параметр, но все же использовать «нормальные» оптопары как-то спокойнее. Транзисторы КТ815Г и КТ814Г, вообще говоря, можно заменить любыми соответствующими маломощными транзисторами, скажем, КТ315Г/КТ361Г или КТ3102/КТ3107, потому что мощность транзистора тут большой роли не играет. Но с более мощными схема может работать стабильнее из-за того, что у них в открытом состоянии внутренние сопротивления переходов существенно ниже. Конденсатор С1, естественно, неполярный, керамический или с органическим диэлектриком.

Для больших токов нагрузки (превосходящих 1–2 А) тиристор придется поставить на радиатор 15–30 $см^2$. Крупным недостатком этой простой и надежной схемы является наличие моста, через который течет тот же ток, что и через нагрузку. При указанных на схеме диодах, рассчитанных каждый на ток до 3 А, и тиристоре с предельным током 10 А мощность в нагрузке может достигать 1,3 кВт (т. к. через каждый диод ток течет только в течение полупериода, то ток через него и выделяющаяся на нем мощность наполовину меньше, чем на тиристоре). Производители диодов из серии 1N54*xx* в описании их характеристик хвастаются, что даже при максимальном токе дополнительного теплоотвода для них не требуется. Однако, если рассчитывать на максимальную мощность, и, тем более, если устройство предполагается установить в герметичном корпусе, где будет, несомненно, очень жарко, то их все же лучше поменять на такие, которые можно устанавливать на радиатор, — например, из серии КД202 с буквами от К до Р (т. к. эти диоды рассчитаны на ток до 5 А, то можно выжать мощность уже 2 кВт). Естественно, можно использовать и готовый мост — скажем, импортный KBL04.

Отладку надо начинать со сборки всей схемы, исключая тиристор с мостом и резистор R7. Регулирующую цепочку R1–R2 на входе оптрона (вместо переменника R1 впаяйте пока постоянный резистор) следует подсоединить к тому источнику питания, который будет использоваться в реальном регуляторе (можно применить любой нестабилизированный источник со встроенной вилкой или только его внутренности, как рассказывалось в *главе 9*). Напряжение источника большого значения не имеет, оно может быть любым в диапазоне от 7 до 20 В. Питание остальной части схемы мы на период отладки обеспечиваем также от источника постоянного тока — можно от того же самого, что питает и регулирующую цепочку.

Затем постоянный резистор, заменяющий R1, перемыкаем накоротко с помощью проволочной перемычки, все включаем и смотрим осциллографом импульсы, кото-

рые должны появиться на резисторе R5. Если импульсов нет, это означает одно из двух: либо что-то неправильно собрано, либо вы их просто не видите, т. к. они достаточно короткие. Посмотрите тогда форму напряжения на конденсаторе C1 — там вы точно должны все поймать. Если конденсатор заряжается и разряжается как надо, попробуйте опять поймать импульсы, меняя длительность развертки и используя синхронизацию. Поймав импульсы, определите по сетке осциллографа и установкам времени развертки длительность промежутка между ними. Изменяя номинал резистора R2, это время нужно установить в пределах одной-полутора миллисекунд, меньше не надо, — ранее мы уже узнали, что при малых фазовых сдвигах регулирования все равно никакого не будет (30° сдвига и соответствует примерно 1,5 мс для частоты 50 Гц). После этого снимаем перемычку с R1. При этом промежуток между импульсами должен оставаться в пределах 10–11 мс. Если это не так, подберите величину резистора R1. Затем на его место следует впаять переменный резистор точно такого же номинала.

Наконец, отключаем осциллограф, подключаем резистор R6 и мост с тиристором, а в качестве нагрузки подсоединяем обычную бытовую лампочку накаливания (ни в коем случае не «энергосберегающую», она от такой схемы управляться не будет!). Насчет мер предосторожности при работе с сетевым напряжением вам уже все, надеюсь, известно (если нет — перечитайте соответствующий фрагмент из *главы 2*). Не забудьте убедиться, что на макете не валяются обрезки выводов компонентов, которые могут замкнуть сетевое питание и устроить тем самым маленький атомный взрыв. Сначала включаете питание регулирующей цепочки, потом — сеть. При вращении движка резистора R3 яркость лампы должна плавно меняться от максимума до полной темноты. В последнем случае волосок не должен светиться совсем, даже темно-красным свечением. Чтобы убедиться в том, что регулирование происходит именно до максимума, надо просто временно перемкнуть тиристор (Осторожно! Перемычку надо устанавливать только при выключенном сетевом питании!) — это и будет номинальная яркость лампы. Если диапазон регулировки недостаточен или, наоборот, в начале или конце наблюдается значительный холостой ход — подберите резисторы R1–R2 поточнее.

Подобрав переменный резистор на несколько десятков или сотен килоом, у которого корпус и ручка надежно изолированы от контактов, можно упростить конструкцию, просто заменив оптрон таким переменным резистором (в этом случае, возможно, придется подобрать емкость конденсатора C1 под выбранный переменник, — см. для образца схему на рис. 10.4 далее). Однако будьте осторожны, особенно при отладке схемы! Корпус такого устройства обязательно должен быть из пластика, а не из металла.

Регулятор переменного напряжения с двумя тиристорами

На рис. 10.4 представлен улучшенный вариант только что рассмотренной схемы, который не требует мощного моста (управляющая оптроном цепочка не показана, она идентична предыдущему случаю) и обеспечивает через нагрузку не пульси-

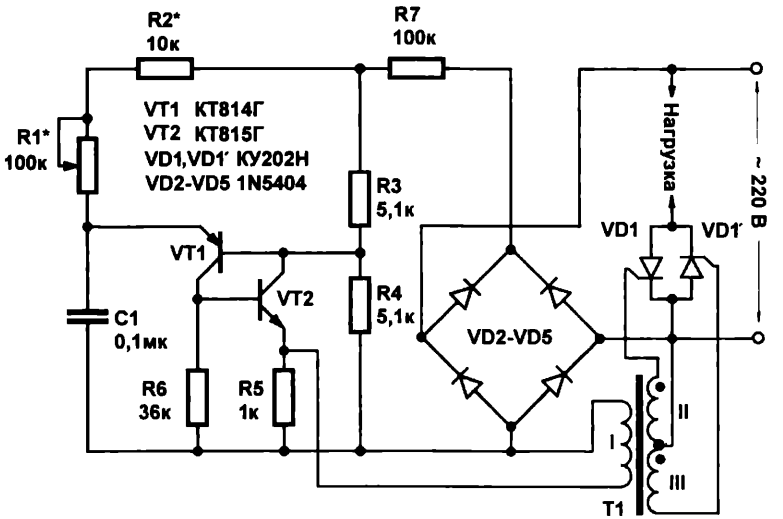


Рис. 10.4. Вариант регулятора с двумя встречно-параллельными тиристорами

рующее, а переменное напряжение (как на осциллограмме, приведенной на рис. 10.2, *внизу*). Для того чтобы получить напряжение в нагрузке в оба полупериода, используются два тиристора VD1 и VD1′, включенные встречно-параллельно. Управление ими осуществляется через импульсный трансформатор T1, который представляет собой ферритовое кольцо марки 1000НН–2000НН диаметром от 10 до 20 мм. Обмотки намотаны проводом МГТФ-0,35. Первичная обмотка (I) содержит 20–30 витков, вторичные (II и III) наматываются вместе и содержат от 30 до 50 витков каждая. Обратите внимание на противоположную полярность включения вторичных обмоток — если она иная, то включение нагрузки будет только в один из полупериодов.

Через маломощный мост КЦ407 питается схема генератора, работа которой не отличается от описанной ранее, только оптрон сразу заменен на простой переменный резистор (Осторожно! Его движок находится под потенциалом бытовой сети!). Резистор R7 можно поставить и до моста в цепь переменного напряжения, тогда требования к предельно допустимому напряжению диодов моста снижаются. А лучше всего питать управляющую схему здесь от отдельного трансформатора на 12–15 вольт, — управляющие электроды тиристоров все равно уже развязаны от схемы через импульсный трансформатор. В случае, если схема при малых значениях R2 будет работать нестабильно (это зависит от используемых транзисторов), его значение следует увеличить до 36–82 кОм, одновременно в такое же число раз уменьшив емкость C1 и увеличив номинал переменного резистора R1.

Регулятор с симистором

Еще один вариант схемы, который позволяет вместо двух тиристоров использовать симистор (триак), показан на рис. 10.5. Отличается этот вариант тем, что генератор работает в обеих полярностях сетевого напряжения: в положительном полупериоде

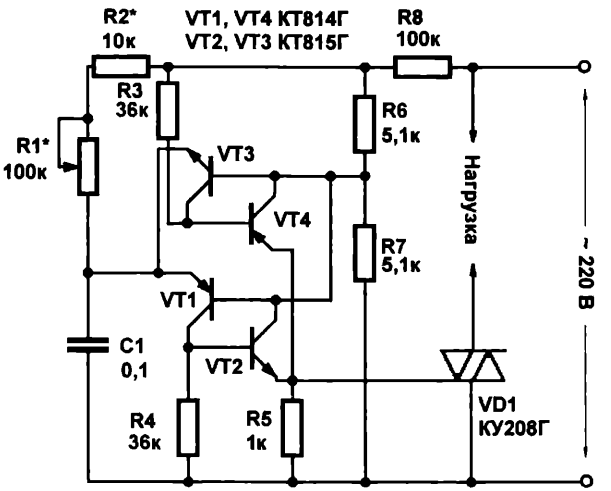


Рис. 10.5. Вариант регулятора с симистором вместо тиристора

задействован аналог однопереходного транзистора с *n*-базой на транзисторах VT1 и VT2, как и ранее, а аналог однопереходного транзистора противоположной полярности (с *p*-базой) на транзисторах VT3 и VT4 делает все то же самое, но в отрицательном полупериоде напряжения. Таким образом управление симистором обеспечивается в обоих полупериодах. Это остроумное решение заимствовано с сайта **http://electrostar.narod.ru/**. Для надежной работы схемы добавлены два запирающих резистора R3 и R4, отсутствующие в оригинале.

Здесь, как и в предыдущих схемах, переменный резистор находится под потенциалом сети. Поскольку выпрямительного моста здесь не требуется, то схему несложно переделать в безопасный вариант, просто запитав всю управляющую часть от отдельного трансформатора на 12–15 вольт и развязав от нее управляющий электрод симистора с помощью импульсного трансформатора, как показано в схеме на рис. 10.4 (вторичная обмотка здесь потребуется всего одна).

Совет

Если вы и в этой схеме хотите сохранить изоляцию от сетевого напряжения в целях управления от внешнего низковольтного устройства (как на рис. 10.3), то надо учесть, что диодный оптрон для этого не годится, т. к. он может работать только в определенной полярности. Можно использовать резисторный оптрон АОР124Б или другой подобный (их не так-то и много разновидностей) или даже изготовить его самостоятельно из светодиода и фотосопротивления (последних как раз в продаже предостаточно). Для этого достаточно закрепить светодиод эпоксидной смолой в стоячем положении на фотосопротивлении так, чтобы он смотрел прямо «в лицо» последнему, а потом плотно закрасить оставшуюся часть окна фоторезистора густой темной краской или залепить черной липкой лентой. Единственный, но существенный недостаток этой схемы по сравнению с предыдущими вариантами, — резисторный оптрон может вести себя не слишком стабильно, особенно при изменениях температуры. Поэтому подобная схема, в силу своей простоты, может быть рекомендована для использования в схемах регулирования мощности с обратной связью, которая устраняет последствия нестабильности регулятора, — например, в схемах термостатов (см. *главу 12*).

Устройство плавного включения ламп накаливания

Лампы накаливания практически всегда перегорают при включении. Это происходит потому, что сопротивление вольфрамового волоска, как и любого металла, зависит от температуры, — с повышением температуры оно повышается, причем из-за огромного перепада температур (порядка 2000 градусов) сопротивление холодной лампы может быть в десятки раз ниже, чем горящей. Например, у лампы 100 Вт, 220 В рабочее сопротивление должно быть почти 500 Ом, однако измерение с помощью мультиметра у выкрученной из цоколя лампы покажет величину меньше 40 Ом. Большой начальный ток и приводит к выходу лампы из строя. Целесообразно при включении постепенно (в течение 0,5–1 с) повышать напряжение — это может продлить срок службы лампы в несколько раз.

Такое устройство — *диммер* — легко соорудить из схемы ручного регулятора в любом из ее вариантов путем небольшой переделки узла управления. Поскольку это устройство не будет содержать органов ручного управления, то его можно питать целиком прямо от сети без оговорок. Оптрон, тем не менее, мы сохраним — как удобное устройство управления. Переделки сведутся к тому, что мы заменим цепочку R1–R2 узлом, показанным на рис. 10.6. Здесь конденсатор C2 (нумерация компонентов сохранена в соответствии с рис. 10.3) после включения питания заряжается через резистор R1 с постоянной времени *RC*. Так как изначально конденсатор разряжен, то тока через светодиод оптрона не будет, и генератор не работает, — темновое сопротивление фоторезистора слишком велико. По мере заряда конденсатора напряжение на выходе эмиттерного повторителя будет возрастать, ток через оптрон станет увеличиваться, и в течение примерно 1 с он возрастет настолько, что фаза управляющих импульсов сдвинется к самому началу полупериода, и яркость горения лампы достигнет максимума. После выключения питания C2 разрядится через цепочку переход база-эмиттер–R2–светодиод оптрона, и схема придет в начальное состояние. Питание управляющего узла должно быть положительным, поэтому мы его питаем через диод VD2.

Удобством в этой схеме является то, что особо тонкой настройки она не требует. Соберите ее при указанных номиналах и сразу включите в сеть. Если яркость растет слишком быстро или, наоборот, медленно — подберите резистор R1. Если же она вообще не достигает максимальной, уменьшите значение резистора R2.

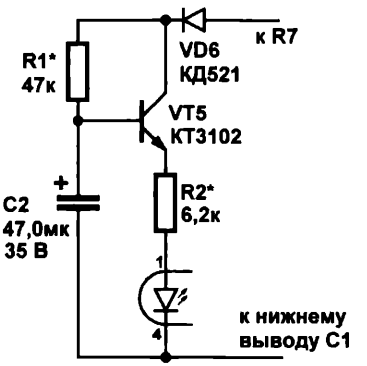


Рис. 10.6. Переделка узла управления для устройства плавного включения ламп накаливания

Подобных схем диммеров очень много в радиолюбительской литературе и в Сети (см., например, сайт **Shema.ru**), имеются и более компактные конструкции, в том числе такие, которые представляют собой двухполюсник и могут подключаться в разрыв цепи нагрузки. Естественно, схемы подобных регуляторов выпускают и в интегральном исполнении.

Заметки на полях

Набирающие популярность энергосберегающие лампы (как обычные люминесцентные, так и светодиодные) таким способом регулировать, конечно, нельзя. Тиристорные диммеры в цепи этих ламп попросту откажутся работать и могут даже вывести лампу из строя. Хотя и есть специальные системы включения таких ламп, совместимые с диммерными регуляторами, но, в общем случае, учитывая, что лампы эти питаются фактически постоянным напряжением, то и регулируются они совсем другим способом, — с помощью изменения скважности высокочастотных питающих импульсов. Подход этот основан на широтно-импульсном регулировании (ШИМ), рассмотренном в *главе 22*. Там же рассказывается о самодеятельных регулирующих конструкциях такого рода. Учтите заранее, что пригодны они будут только для светодиодных осветителей или люминесцентных ламп без встроенных пускорегулирующих устройств, — для регулирования обычных ламп, у которых пускорегулятор встроен в цоколь, они не годятся точно так же, как и тиристорные диммеры.

Помехи

В заключение главы о мощной нагрузке нужно прояснить еще один момент, связанный с помехами. В начале главы я долго распинался по поводу того, что резко выключать мощную нагрузку в сети нельзя, и что оптоэлектронные реле даже имеют специальные средства для отслеживания момента перехода через ноль. Между тем, все рассмотренные схемы с фазовым управлением по принципу своего устройства как раз и переключают нагрузку посередине периода. Потому, если вы включите такой регулятор напрямую в сеть, то помех не избежать: как электрических по проводам сети, так и электромагнитных, распространяющихся в пространстве, и чем мощнее нагрузка, тем больше эти помехи. Особенно чувствительны к этому делу АМ-приемники — мощный регулятор может давить передачи радио «Свобода» не хуже советских глушилок. Для того чтобы свести помехи к минимуму, необходимо, во-первых, заземлить корпус прибора, а, во-вторых, на входе питания устройства вместе с нагрузкой поставить LC-фильтр. Это относится и к регуляторам в интегральном исполнении.

Заметки на полях

Внимательный читатель, несомненно, давно уже задает вопрос: если тиристор при отсутствии тока через него выключается, то как можно запустить тиристорную схему в момент перехода напряжения через ноль? Отвечаю: естественно, никак. Поэтому схема zero-коррекции на самом деле запускает мощный тиристор не точно в момент равенства анодного напряжения нулю, а тогда, когда ток через него уже достигает некоторой небольшой, но конечной величины. Практически это обеспечить несложно — надо дождаться момента очередного перехода через ноль и сразу запустить генератор открывающих импульсов на достаточно высокой частоте. Тиристор «сам выберет» из последовательности импульсов тот, при котором «уже можно открываться».

Для заземления корпус, естественно, должен быть металлический или металлизированный изнутри. В выигрышном положении окажутся те, кто будет изготавливать корпуса самостоятельно из стеклотекстолита, по технологии, изложенной в *главе 3*, — у них уже есть прекрасный экран из медной фольги, достаточно только припаять провод заземления в любом удобном месте на внутренней стороне корпуса и присоединить его к желто-зелененькому третьему проводу в сетевой вилке. Если же корпус пластмассовый, то его нужно изнутри оклеить алюминиевой фольгой потолще (предназначенная для применения в микроволновых печах, конечно, не подойдет). Надежно обеспечить контакт вывода заземления с таким экраном можно, приклеив зачищенный на несколько сантиметров провод широким скотчем или соорудив прижимной контакт из упругой бронзы (например, из контакта старого мощного реле).

На рис. 10.7 приведены два варианта построения развязывающего *LC*-фильтра. В *верхнем* варианте предпочтительно, чтобы через *дроссель* (так называют индуктивности, если они служат для фильтрации высоких частот в шинах питания и в некоторых других случаях) проходил фазный (обозначен буквой «Ф»), а не нулевой («Н») провод бытовой сети. Второй вариант (на рисунке *внизу*) более «продвинутый».

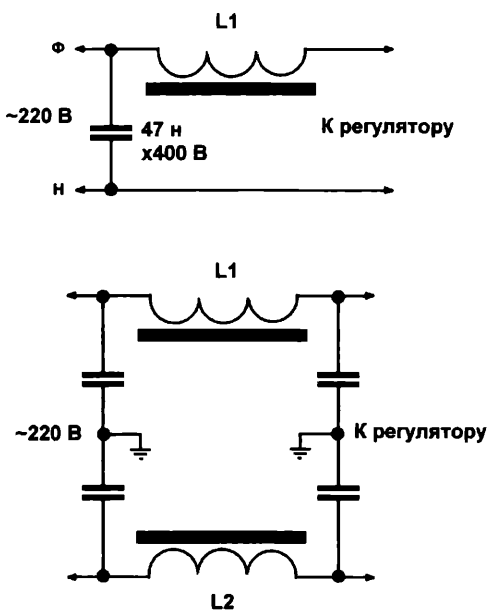
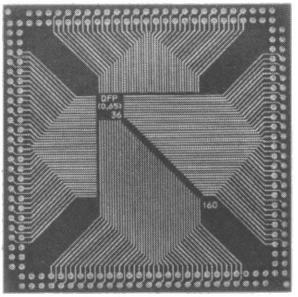


Рис. 10.7. Схемы фильтров сетевого питания для подавления помех

Для изготовления дросселя нужно взять ферритовое кольцо марки 600–1000НН диаметром 20–30 мм и намотать на него виток к витку провод МГШВ сечением около 1 мм^2 — сколько уместится. В *нижнем* варианте фильтра дроссели L1 и L2 можно объединить, намотав их на одном кольце, — причем если помехи будут по-

давляться плохо, то надо поменять местами начало и конец одной из обмоток. Можно использовать и готовые дроссели подходящей мощности.

Если нагрузка совсем маломощная (до 10 Вт), то дроссели можно в крайнем случае заменить резисторами в 5–10 Ом мощностью не менее 2 Вт. Конденсаторы — любые неполярные на напряжение не менее 400 В, среднюю точку их в *нижнем* варианте нужно подсоединить к настоящему заземлению (т. е. к уже заземленному корпусу). Если таковое отсутствует, то все равно надо присоединить эту точку к корпусу прибора, но без настоящего заземления работа фильтра заметно ухудшится — фактически он превратится в несколько улучшенный *верхний* вариант.



ГЛАВА 11

Слайсы, которые стали чипами
О микросхемах

Ему предстояло увидеть наяву тот заветный сундук, который он двадцать раз представлял в своих грезах.

А. Дюма. «Три мушкетера»

Самые первые микросхемы были совсем не такими, как сейчас. Они изготавливались гибридным способом: на изолирующую подложку напылялись алюминиевые проводники, приклеивались маленькие кристаллики отдельных транзисторов и диодов, малогабаритные резисторы и конденсаторы, и затем все это соединялось в нужную схему тонюсенькими золотыми проволочками — вручную, точечной сваркой под микроскопом. Можно себе представить, какова была цена таких устройств, которые назывались тогда *гибридными микросхемами*. Гибридные микросхемы какое-то время были лидерами рынка, настойчиво требовавшего миниатюризации и повышения надежности электронных схем, — на их основе, например, были созданы многие топовые компьютерные системы 1960–70 гг., такие, как знаменитая IBM/360 или отечественная М-10 конструктора М. А. Карцева. К гибридным микросхемам относятся и некоторые современные типы — к примеру, оптоэлектронные реле — но, конечно, сейчас выводы отдельных деталей уже вручную не приваривают.

Ведущий специалист и один из основателей компании Fairchild Semiconductor Роберт Нойс позднее признавался, что ему стало жалко работников, терявших зрение на подобных операциях, и в 1959 году он выдвинул идею *микросхемы* — «слайса», или «чипа» (slice — ломтик, chip — щепка, осколок), где все соединения наносятся на кристалл прямо в процессе производства. Несколько ранее аналогичную идею выдвинул сотрудник Texas Instruments Джек Килби, однако опоздавший Нойс, химик по образованию, разработал детальную технологию изготовления (это была так называемая *планарная технология* с алюминиевыми межсоединениями, которая часто используется и по сей день). Спор о приоритете между Килби и Нойсом продолжался в течение десяти лет, и в конце концов победила дружба — было установлено считать Нойса и Килби изобретателями микросхемы совместно. В 2000 году Килби (Нойс скончался в 1990) получил за изобретение микросхемы Нобелевскую премию (одновременно с ним, но за достижения в области оптоэлектроники, ее получил и российский физик Жорес Алферов).

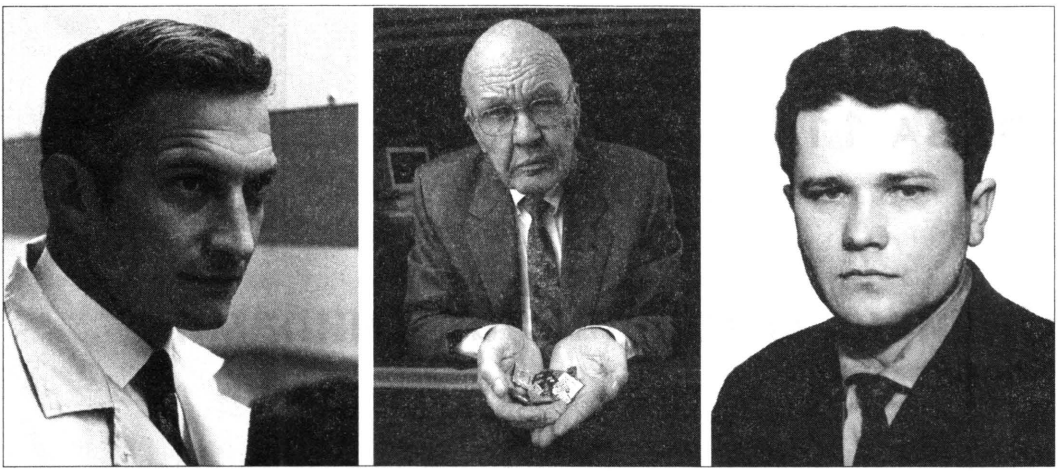


Рис. 11.1. Изобретатели микросхемы Роберт Нойс (Robert Noyce), 1927–1990 (*слева*), Джек Килби (Jack St. Clair Kilby), 1923–2005 (*в центре*) и Юрий Валентинович Осокин, 1938–2013 (*справа*)

Увидев фото Юрия Валентиновича Осокина рядом со знаменитыми западными изобретателями микросхемы, иной читатель может ухмыльнуться: мол, наверняка очередная история из серии «Россия — родина слонов». Однако это не так — никто до времени не подозревал о существовании Осокина и его изобретений просто потому, что изобретение это в советское время было строго засекречено, а в девяностые годы Осокин, ставший к тому времени президентом АО «Альфа» (бывший Рижский завод полупроводниковых приборов), видимо, не счел нужным, как говорится, «поднимать волну», — все-таки прошло тридцать лет со времени его работы. Но никто из знающих его историю не сомневается, что по справедливости Нобелевскую премию 2000-го года он должен был разделить с Джеком Килби.

Следует подчеркнуть, что Рижскому заводу полупроводниковых приборов, где работал 24-летний Юрий Осокин, поручили найти способ повышения плотности компоновки аппаратуры весной 1962 года, когда компания Fairchild еще только налаживала массовый выпуск своей первой серии Micrologic, и притом пока только в интересах Минобороны США и НАСА. Осокин ничего не знал о технологиях Нойса и Килби, и его технология выглядит совершенно иначе. В результате он получил однокристальную микросхему Р12-2, которая была, а возможно, и остается, самой миниатюрной в мире корпусированной интегральной схемой, — она была выполнена в виде «таблетки» (круглой металлической чашечки) диаметром 3 мм и высотой 0,8 мм.

Историю изобретения Осокина в подробностях можно прочесть в статье Б. М. Малашевича «Первые интегральные схемы» на сайте «Виртуальный компьютерный музей» (**http://computer-museum.ru/histekb/integral_1.htm**). Борис Михайлович свидетельствует: *Р12-2 и модули «Квант» на ее основе, в 1969 году получившие обозначения ИС 1ЛБ021 и «серия 116», производились РЗПП до середины 1990-х годов в объемах до нескольких миллионов в год. Они работают в аппаратуре и в XXI веке, например, в составе разработанного НИИРЭ бортового компьютера «Гном», стоящего в штурманской кабине «Ил-76» и некоторых других самолетов.*

Fairchild Semiconductor

Компания Fairchild Semiconductor в области полупроводниковых технологий была примерно тем, чем фирма Маркони в области радио или фирма Xerox в области размножения документов. Началось все еще с ее рождением: восемь инженеров, уволившихся в 1957 году из основанной изобретателем транзистора Уильямом Шокли компании Shockley Semiconductor Labs, обратились к начинающему финансисту Артуру Року — единственному, кому их идеи показались интересными. Рок нашел компанию из холдинга Шермана Файрчайлда, которая согласилась инвестировать основную часть из требуемых 1,5 миллиона долларов, и с этого момента принято отсчитывать появление нового способа финансирования инновационных проектов — *венчурных* (т. е. «рисковых») *вложений*, что в дальнейшем позволило родиться на свет множеству компаний, названия которых теперь у всех на слуху.

Следующим достижением Fairchild стало изобретение микросхем Робертом Нойсом, и первые образцы многих используемых и поныне их разновидностей были созданы именно тогда (например, в одном из первых суперкомпьютеров на интегральных схемах, знаменитом ILLIAC IV, были установлены микросхемы памяти производства Fairchild). А в 1963 году отдел линейных интегральных схем в Fairchild возглавил молодой специалист по имени Роберт Видлар, который стал «отцом» интегральных операционных усилителей, основав широко распространенные и поныне серии, начинающиеся с букв μ и LM (мы о нем уже упоминали в *главе 9* в связи с интегральными стабилизаторами питания). Логические микросхемы по технологии КМОП (см. *главу 15*) изобрел в 1963 году также сотрудник Fairchild Фрэнк Вонлас, получивший на них патент № 3 356 858.

Intel и AMD

В 1965 году знаменитый Гордон Мур, тогда — один из руководителей Fairchild, входивший вместе с Нойсом в восьмерку основателей, сформулировал свой «закон Мура» о том, что производительность и число транзисторов в микросхемах удваиваются каждые 1,5 года, — этот закон фактически соблюдается и по сей день! В 1968 году Нойс с Муром увольняются из Fairchild и основывают фирму, название которой еще недавно знал каждый школьник: Intel. В ней был построен первый микропроцессор (см. *главу 18*), появились новые устройства памяти (такие, как флэш-память, например). Именно процессоры Intel доминировали в настольных компьютерах и ноутбуках, бывших основой компьютерной революции 1980-х — начала 2000-х. Инвестором новой компании стал все тот же Артур Рок. А другой работник Fairchild, Джереми Сандерс, в следующем, 1969 году основывает фирму почти столь же известную, как и Intel, — ее «заклятого друга» и конкурента AMD. Процессоры обеих компаний лидируют в своем секторе до сих пор, но мобильную революцию обе они, к сожалению, прозевали, и на слуху теперь в основном другие производители.

Что же дало использование интегральных микросхем, кроме очевидных преимуществ типа миниатюризации схем и сокращения количества операций при проектировании и изготовлении электронных устройств?

Рассмотрим прежде всего экономический аспект. Первым производителям чипов это было еще не очевидно, но экономика производства микросхем отличается от экономики других производств. Одним из первых, кто понял, как именно нужно торговать микросхемами, был уже упомянутый Джереми Сандерс (тогда — сотрудник Fairchild, впоследствии — руководитель компании AMD на протяжении более трех десятилетий).

Пояснить разницу можно на следующем примере. Если вы закажете архитектору проект загородного дома, то стоимость этого проекта будет сравнима со стоимостью самого строительства дома. Если вы по этому проекту построите сто домов,

то стоимость проекта поделится на сто, но построить дом дешевле, чем стоят материалы и оплата труда рабочих, нельзя, а они-то и составляют значительную часть стоимости строительства. То есть строительство каждого из ста одинаковых домов по готовому проекту будет почти вдвое дешевле, чем одного-единственного, но еще больше удешевить производство вы уже не сможете. В производстве же микросхем все иначе: цена материалов, из которых они изготовлены, и издержек производства в пересчете на каждый «чип» настолько мала, что она составляет едва ли единицы процентов от стоимости конечного изделия. Поэтому основная часть себестоимости чипа складывается из стоимости его проектирования и стоимости самого предприятия, на котором они изготавливаются, — фабрика для выпуска полупроводниковых компонентов может обойтись в сумму порядка 2–4 миллиардов долларов. Ясно, что в этой ситуации определяющим фактором стоимости микросхемы будет их количество — обычно, если вы заказываете меньше миллиона экземпляров, то с вами даже разговаривать не станут, а если вы будете продолжать настаивать, то один экземпляр обойдется вам во столько же, сколько и весь миллион. Именно массовость производства приводит к тому, что сложнейшие схемы, которые в дискретном виде занимали бы целые шкафы и стоили бы десятки и сотни тысяч долларов, продаются дешевле томика технической документации к ним.

Попутно заметим, что позднее специально для изготовления микросхем единичными экземплярами или малыми сериями придумали программируемые логические матрицы (FPGA) — заготовки, в которых разработчик может программно сформировать любую структуру по своему желанию. Их мы в этой книге рассматривать не будем, т. к. порог освоения для них весьма велик, и любителями они обычно не применяются.

Вторая особенность экономики производства микросхем — то, что их цена мало зависит от сложности. Микросхема простого операционного усилителя содержит несколько десятков транзисторов, микросхема микроконтроллера — несколько десятков или сотен тысяч, однако их стоимости сравнимы. Эта особенность тоже не имеет аналогов в дискретном мире — с увеличением сложности обычной схемы ее цена растет пропорционально количеству использованных деталей. Единственный фактор, который фактически ведет к увеличению себестоимости сложных микросхем по сравнению с более простыми (кроме стоимости проектирования), — это процент выхода годных изделий, который может снижаться при увеличении сложности. Если бы не это, то стоимость Intel Core i7 не намного бы превышала стоимость того же операционного усилителя. Однако в Core i7, извините, несколько сотен миллионов транзисторов! Это обстоятельство позволило проектировщикам без увеличения стоимости и габаритов реализовать в микросхемах такие функции, которые в дискретном виде было бы реализовать просто невозможно или крайне дорого.

Заметки на полях

Кстати, выход годных — одна из причин того, что кристаллы микросхем такие маленькие. В некоторых случаях разработчики даже рады были бы увеличить размеры, но тогда резко снижается и выход годных. Типичный пример такого случая — борьба производителей профессиональных цифровых фотоаппаратов за увеличение размера светочувствительной матрицы. Матрицы размером с пленочный кадр (24×36 мм) и на

момент первого издания этой книги в 2005 году, и сейчас, имеют только лучшие (и самые дорогие) модели зеркальных фотокамер. А так называемые *цифровые задники* для высокоразрешающей съемки фирм PhaseOne, Sinar, Mamiya и других производителей с еще большими физическими размерами вообще никто не покупает в собственность, — их только сдают напрокат за суммы порядка 2000 долларов в месяц.

Но, конечно, тенденция к миниатюризации имеет и другую причину — чем меньше технологические нормы, тем меньше потребляет микросхема и тем быстрее она работает. Простые логические микросхемы КМОП серии 4000В (см. *главу 15*) выпускали в процессе с технологическими нормами 4 мкм, микропроцессор i8086 — по технологии 3 мкм, и работали они на частотах в единицы, в лучшем случае — десятки мегагерц. Процессор Pentium 4 с ядром Willamette (нормы 0,18 мкм, рабочая частота 1,3–2 ГГц) имел тепловыделение до 72 Вт, а тот же Pentium 4 с ядром Northwood (нормы 0,13 мкм, рабочая частота 1,6—3,4 ГГц) — уже 41 Вт. В настоящее время большая часть микропроцессоров выпускается по нормам 0,022–0,045 мкм, осваиваются нормы 14 и даже 10 нм. Вспомните, что диаметр единичного атома имеет порядок 0,2–0,3 нм, так что по ширине дорожки на кристалле, изготовленном с такими нормами, укладывается всего полсотни атомов кремния!

Еще одна особенность микросхем — надежность. Дискретный аналог какого-нибудь устройства вроде аналого-цифрового преобразователя содержал бы столько паек, что какая-нибудь в конце концов обязательно оторвалась бы. Между тем, если вы эксплуатируете микросхему в штатном режиме, то вероятность ее выхода из строя измеряется миллионными долями единицы. Это настолько редкое явление, что его можно практически не учитывать на практике, — если у вас сломался какой-то прибор, ищите причину в контактах переключателей, в пайках внешних выводов, в заделке проводов в разъемах — но про возможность выхода из строя микросхемы забудьте. Разумеется, это, повторяю, относится к случаю эксплуатации в штатном режиме — если вы подали на микрофонный вход звуковой карты напряжение 220 В, конечно, в первую очередь пострадает именно микросхема. Но сами по себе они практически не выходят из строя никогда.

Некоторые типовые узлы микросхем и особенности их эксплуатации

Наконец, для схемотехников микросхемы обладают еще одним бесценным свойством — все компоненты в них изготавливаются в едином технологическом процессе и находятся при этом в строго одинаковых температурных условиях. Это совершенно недостижимо для дискретных приборов — например, пары транзисторов, для которых желательно иметь идентичные характеристики, ранее приходилось подбирать вручную (такие подобранные пары специально поставлялись промышленностью) и иногда даже ставить их на медную пластину, чтобы обеспечить одинаковый температурный режим.

Рассмотрим типичный пример такого случая — так называемое *токовое зеркало* (рис. 11.2). Эта схема работает следующим образом. Левый по схеме транзистор представляет собой фактически диод, т. к. у него коллектор соединен с базой. Из характеристики диода (см. рис. 6.1 в *главе 6*) видно, что при изменении прямого

тока на нем несколько меняется и падение напряжения (оно не равно точно 0,6 В). Это напряжение без изменений передается на базу второго, ведомого транзистора, в результате чего он выдает точно такой же ток — но только при условии, что характеристики транзисторов согласованы с высокой степенью точности. То есть, если току 1 мА через первый транзистор соответствует напряжение на его переходе база-эмиттер, равное, к примеру, 0,623 В, то такому же напряжению на переходе второго транзистора должен соответствовать такой же ток. Мало того, это соответствие должно сохраняться во всем диапазоне рабочих температур! Естественно, столь высокая идентичность характеристик практически недостижима для дискретных приборов, а для транзисторов, входящих в состав микросхемы, она получается сама по себе, без дополнительных усилий со стороны разработчиков.

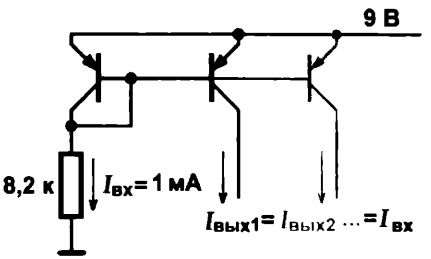


Рис. 11.2. Токовое зеркало

Схемы подобных токовых зеркал получили широкое распространение в интегральных операционных усилителях (ОУ) в качестве нагрузки входного дифференциального каскада, что значительно лучше, чем использование простых резисторов. Их применение вместо резисторов гарантирует повторяемость характеристик ОУ в широком диапазоне питающих напряжений. Отметим также, что ведомых транзисторов может быть много (на рис. 11.2 второй такой транзистор показан серым цветом), их количество ограничивается только тем обстоятельством, что базовые токи вносят погрешность в работу схемы, отбирая часть входного тока на себя. Впрочем, и с этим обстоятельством можно успешно бороться.

Кстати, резисторы в микросхемах в некритичных случаях все равно предпочитают делать из транзисторов — сформировать обыкновенный резистор, как проводник с заданным сопротивлением, в процессе производства микросхем значительно труднее, чем соорудить, скажем, полевой транзистор с заданным начальным током стока. По этой причине, если точных значений номиналов резисторов согласно функциональным особенностям микросхемы не требуется, то они имеют большой разброс, — скажем, сопротивление «подтягивающих» резисторов портов микроконтроллеров AVR может колебаться в пределах от 20 до 50 кОм. Ну, а если все же точные или хотя бы согласованные номиналы резисторов иметь необходимо (как, к примеру, в микросхемах ЦАП и АЦП, которые мы будем рассматривать в *главе 18*), то после изготовления микросхемы их приходится специально подгонять с помощью лазера, что значительно удорожает производство.

На рис. 11.2 соединение баз транзисторов не случайно показано необычным способом — в реальности они действительно представляют собой одну структуру, на

которую «навешиваются» коллекторы и эмиттеры отдельных транзисторов. В микросхемах могут использоваться такие разновидности транзисторных структур, которые в обычной дискретной жизни не имеют смысла, — скажем, многоэмиттерные или многоколлекторные транзисторы. Для примера на рис. 11.3 приведена схема входного каскада микросхемы транзисторно-транзисторной логики (ТТЛ), осуществляющей логическую функцию «ИЛИ» (подробно об этом рассказано в *главах 14* и *15*). В этой схеме замыкание любого из трех эмиттеров (или двух, или всех вместе, — потому функция и называется «ИЛИ») на общий провод питания приведет к тому, что транзистор откроется и обеспечит ток через нагрузку.

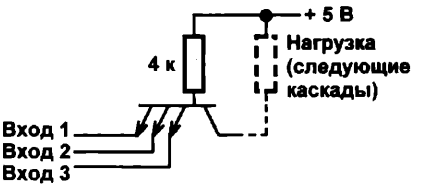


Рис. 11.3. Входной каскад элемента ТТЛ

Кратко рассмотрим общие особенности эксплуатации различных типов микросхем (более подробно о конкретных типах будет рассказываться в соответствующих главах). Вы, возможно, слышали о том, что микросхемы боятся статического электричества. Действительно, потенциал заряда, накапливающегося при ходьбе на нейлоновом халатике симпатичной монтажницы, одетой к тому же в синтетические юбочку, кофточку и колготки, может составлять тысячи вольт (правда, сама величина заряда невелика). И необязательно носить синтетическую одежду — достаточно походить по полу, покрытому обычным линолеумом, ламинатом или недорогим ковровым покрытием, чтобы накопить на себе вполне достаточный заряд (в дорогих покрытиях это чаще всего предусмотрено, и зарядам накапливаться не дают). Такое напряжение, конечно, может вывести из строя микросхемы и не только микросхемы — особенно чувствительны к нему полевые транзисторы с изолированным затвором. Поскольку заряду на выводе затвора у них стекать некуда, то весь накопленный на вас потенциал будет приложен к тоненькому промежутку между затвором и каналом, и не исключено, что изолирующий слой окисла кремния не выдержит такого надругательства.

Поэтому при монтаже всегда следует соблюдать несколько правил: не носить синтетическую одежду и не использовать синтетические покрытия для пола (по возможности) и монтажного стола (профессиональные монтажные столы вообще покрывают заземленным металлическим листом). Неплохую гарантию дает заземление корпуса паяльника, но на практике это осуществить непросто, учитывая, что заземление, как таковое, в наших домах нередко отсутствует. Можно также привести еще несколько рекомендаций:

- не хвататься руками за выводы микросхем без нужды, а при необходимости взять микросхему в левую (для левшей — в правую) руку так, чтобы пальцы касались выводов питания;

- первыми всегда следует припаивать выводы питания микросхемы (для дискретных транзисторов — эмиттер или исток);
- перед началом монтажа, особенно если вы только что переодевались, подержаться руками за заземленный металлический предмет (водопроводный кран);
- при стирке рабочей одежды обязательно использовать антистатик.

Хорошую защиту также дает метод, при котором вы не впаиваете микросхему в плату непосредственно, а устанавливаете ее на панельку. Панельку, естественно, можно совершенно безопасно монтировать любым паяльником, а микросхема устанавливается в нее в самый последний момент. Правда, такой метод снижает надежность конструкции (см. далее), но для любительских применений это не имеет большого значения и в «ширпотребовской» электронике, если габариты позволяют, панельки применяют довольно часто.

Впрочем, случаи выхода микросхем из строя от статического электричества все же довольно редкие, т. к. производители эту опасность учитывают и для критичных случаев принимают меры по защите выводов. Самой распространенной мерой является установка защитных диодов — по два на каждый вывод так, чтобы один из них был присоединен катодом к плюсу питания, а другой — анодом к минусу (рис. 11.4). Если напряжение на выводе не выходит за пределы питания, то такие диоды не оказывают никакого влияния на работу схемы (или почти никакого, см. далее). Если же напряжение на выводе выходит за эти пределы, то оно замыкается через соответствующий диод либо на шину питания, либо на общую шину. Подобной защитой снабжены, например, все КМОП-микросхемы серии 4000, а также порты ввода/вывода микроконтроллеров фирмы Atmel, которые мы далее будем использовать в примерах в этой книге.

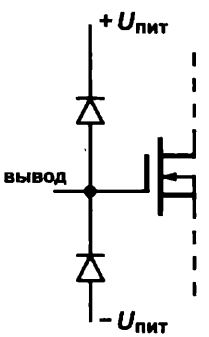


Рис. 11.4. Защита выводов микросхем от перенапряжения

Кстати, подобный прием позволяет иногда защитить микросхему и от неправильного включения питания — если плюс и минус питания на схеме рис. 11.4 поменять местами, то весь ток пойдет через диоды, и питание упадет до двойного падения напряжения на диоде. В этих целях иногда ставят и еще один отдельный защитный диод — прямо от питания до питания. Правда, тут весь вопрос в том, насколько долго диоды могут выдерживать прямой ток от источника. В моей практике бывали случаи, когда и выдерживали. Иногда в технических характеристиках

указывают максимальный ток, который могут выдержать защитные диоды без повреждения.

Наличие защитных диодов следует учитывать. Скажем, микросхемы с комплементарными полевыми транзисторами (КМОП), которые мы будем подробно изучать в *главе 15*, в статическом режиме и на низких частотах потребляют настолько малый ток, что вполне могут питаться и через защитный диод от входного сигнала, даже если напряжение питания вообще не подключено. Правда, при этом с выходным сигналом творятся всякие чудеса, однако выглядит это довольно эффектно. Конечно, обычные КМОП-микросхемы никто в таком режиме не использует, но иногда в микросхему специально встраивают небольшой конденсатор по питанию, который накапливает заряд от входного сигнала и позволяет ей некоторое время работать, — скажем, ответить на запрос по последовательному интерфейсу. По этому принципу устроены, например, цифровые полупроводниковые датчики температуры фирмы Maxim/Dallas — они могут соединяться с показывающим прибором по интерфейсу, который так и назвали: *1-wire* («однопроводной»), где передается только сигнал, тащить питание отдельно не требуется.

Заметки на полях

С другой стороны, наличие защитных диодов может приводить к неприятностям, — наиболее распространенная ошибка разработчиков электронных схем состоит в том, что при переходе на резервное питание они в целях экономии отключают питание всех узлов, кроме центрального контроллера, или, к примеру, генератора часов реального времени, забывая при этом отключить у них внешние соединения. Тогда схема начинает потреблять даже больше, чем она потребляла в нормальном режиме: если на выходе контроллера есть напряжение, а микросхема, ко входу которой этот выход подсоединен, обесточена, то указанный выход оказывается фактически замкнутым накоротко через защитный диод на шину питания. Мне могут возразить, что на шине питания при выключенном источнике потенциала нет, и току течь некуда — действительно, если питание оборвано, то плюсовая и минусовая шины вроде бы никак не связаны между собой, и плюс питания обязан «висеть в воздухе». Однако это рассуждение справедливо только в теории. В сложно устроенном кристалле при неопределенных потенциалах его отдельных областей возникает достаточно открытых *p-n*-переходов, чтобы ток утекал в неизвестном направлении.

В некоторых случаях защитные диоды не ставят — они все же имеют хотя и очень небольшой, но конечный ток утечки, который может быть важен, скажем, в случае так называемого *зарядового усилителя*, т. е. устройства, которое измеряет величину накопленного заряда. Часто не ставят их и в микросхемах для обработки высокочастотных сигналов. Так что, на всякий случай, особенно если вы не уверены в наличии защитных диодов, меры предосторожности при монтаже следует соблюдать.

Некоторые характерные типы корпусов микросхем приведены на рис. 11.5 и 11.6. *Вверху слева* на рис. 11.5 показан пример панельки («сокета») для корпусов типа DIP-40 (т. е. для DIP-корпуса с 40 выводами), которая позволяет не впаивать микросхему в плату. *Внизу* на рис. 11.6 показана аналогичная панелька для корпусов типа PLCC (пример самого корпуса приведен на этом рисунке *выше*). Корпуса PLCC специально разработаны на такой случай — их легко вынимать и вставлять обратно, не опасаясь повредить выводы. Подобные панельки особенно удобны в макетах,

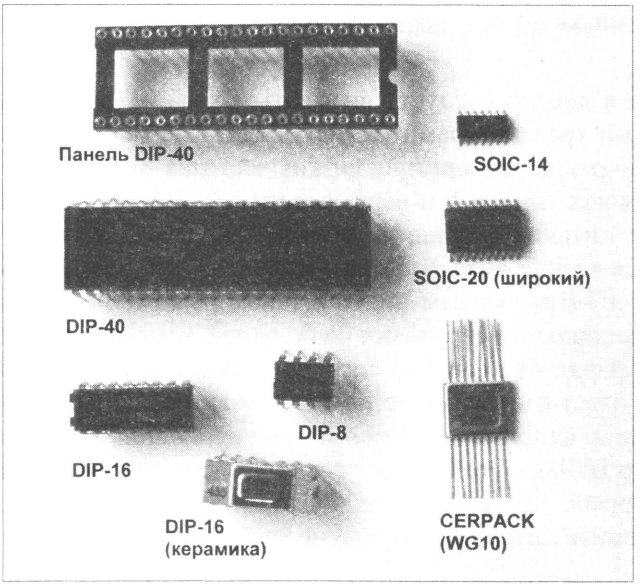


Рис. 11.5. Некоторые распространенные корпуса микросхем: *вверху слева* — панелька для корпусов типа DIP-40

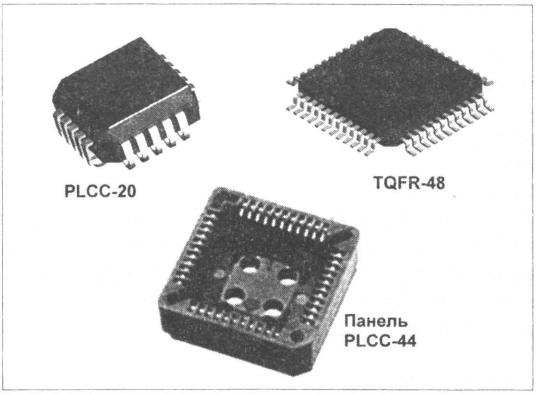


Рис. 11.6. Некоторые особые типы корпусов для микросхем: *слева вверху* — корпус PLCC, *внизу* — панелька для такого типа микросхем; *справа вверху* одна из разновидностей корпусов для микросхем с большим количеством выводов

а также их всегда применяют в случаях, если микросхему нужно иногда извлекать из устройства для ее перепрограммирования. Понятно, что механические контакты в панельке понижают надежность конструкции, и если извлечение микросхемы не предполагается, то все же лучше панельку не использовать, а если без нее никак не обойтись, то тип PLCC предпочтительнее, чем DIP.

Подробности

Как и в разговоре о транзисторах, для отечественных микросхем мы приводим наименования импортных аналогов корпусов, ибо отечественная система столь сложна, что только путает. При замене отечественных аналогов на импортные и обратно следует учитывать, что у нас шаг между выводами метрический и кратен величине 2,5 мм (для

планарных корпусов — долям от нее, т. е. 1,25 мм, 0,625 мм и т. п.). У импортных шаг вычисляется, исходя из десятой доли дюйма (2,54 и 1,27 мм, для меньших шагов кратность уже обычно не соблюдается). Для корпусов с числом выводов по одной стороне, равным 8 и менее, эта разница несущественна — взаимозаменяемы и платы, и панельки, особенно для DIP-корпусов. А вот для планарных, где шаг мельче, разница начинает сказываться уже для пяти-семи выводов по одной стороне. Для микросхем с большим числом выводов простая замена импортных микросхем отечественными и наоборот оказывается невозможна — приходится заранее рассчитывать плату под определенный шаг выводов.

Имейте в виду, что очень часто выводов в микросхеме больше, чем требует ее функциональность. В этом случае существует правило: *незадействованные выводы никуда не присоединять!* В западной документации специально даже принято для таких незадействованных выводов обозначение NC (no connected). Будет грубой ошибкой, например, присоединить *незадействованные выводы* 13 и 16 КМОП-микросхемы 561ПУ4 (см. *главу 15*) к «земле» или питанию, как это следует делать со входами *незадействованных элементов*.

Звуковые усилители на микросхемах

В качестве первого примера использования микросхем мы попробуем построить несколько звуковых усилителей на различные случаи жизни. Надо сказать, что дело это намного более простое, чем конструировать их из дискретных компонентов, что мы делали в *главе 8*, — нужно только следовать рекомендациям разработчика, и все гарантированно получится. Это правило касается всех микросхем без исключения. Разумеется, если вы хорошо изучили особенности соответствующей разновидности интегральных схем, то имеете полное право поэкспериментировать, но — на свой страх и риск.

Заметки на полях

В точности то же самое правило, кстати, действует в программировании: при разработке сложных программ используются типовые библиотеки или системные API (Application Program Interface, программный интерфейс пользователя), разработанные другими фирмами и сопровождаемые рекомендациями к их использованию. В результате и программа, и схема собираются из готовых «кирпичиков», а их разработчику остается только придать конечному продукту нужную функциональность в пределах, предусмотренных производителем «полуфабриката».

Такой страх и риск вполне уместен, если вы попытаетесь использовать для построения звукового усилителя отечественные микросхемы серии К174 — характеристики их настолько ужасны, что попробовать их улучшить, как говорится, сам бог велел. Мы не будем с ними разбираться, просто потому что они, по большому счету, внимания вообще не заслуживают, а рассмотрим две конструкции на доступных импортных микросхемах.

Мощный УМЗЧ

Первой мы рассмотрим стандартную схему усилителя мощности низкой частоты (УМЗЧ) на популярной микросхеме TDA2030 производства фирмы ST Microelectronics. Схема обладает примерно такими же характеристиками, как усилитель,

описанный в *главе 8*. Производитель гарантирует при выходной мощности 14 Вт на нагрузке 4 Ом искажения сигнала не более 0,5% (что заведомо лучше нашей самодеятельной конструкции). Если снизить требования к величине искажений, то при наших ±15 В питания из микросхемы можно выжать и те же самые 21 Вт. Предельно допустимое значение напряжения питания для TDA2030 достигает ±18 В, но, разумеется, при таком питании ее эксплуатировать не рекомендуется. Увеличение искажений при повышении выходной мощности, вероятно, связано с тем, что в чип встроена защита от перегрева выходных транзисторов, которая ограничивает выходной ток, когда температура корпуса повышается.

Кроме уровня искажений, эта схема обладает и рядом других преимуществ перед нашей дискретной конструкцией. Производитель гарантирует такие характеристики, как широкий диапазон частот, которые передаются с заданным коэффициентом усиления и при заданных искажениях сигнала (40 Гц–15 кГц), или подавление влияния нестабильности источника питания на качество выходного сигнала (в 100–300 раз, что позволяет спокойно использовать наш простейший источник, описанный в *главе 9*). Гарантируется устойчивость усилителя при использовании рекомендуемых номиналов резисторов и конденсаторов и даже приводятся рекомендации по размерам охлаждающего радиатора. Выход микросхем, как мы уже сказали, защищен от короткого замыкания в нагрузке (точнее, ограничен выходной ток).

От перечисления весьма греющих душу свойств этой конструкции перейдем, наконец, к рассмотрению ее схемы (рис. 11.7). Собственно усилитель включает саму микросхему DA1, конденсаторы C1, C2 и резисторы R1–R4. Если внимательно присмотреться, то мы увидим, что структурно она ничем не отличается от нашей

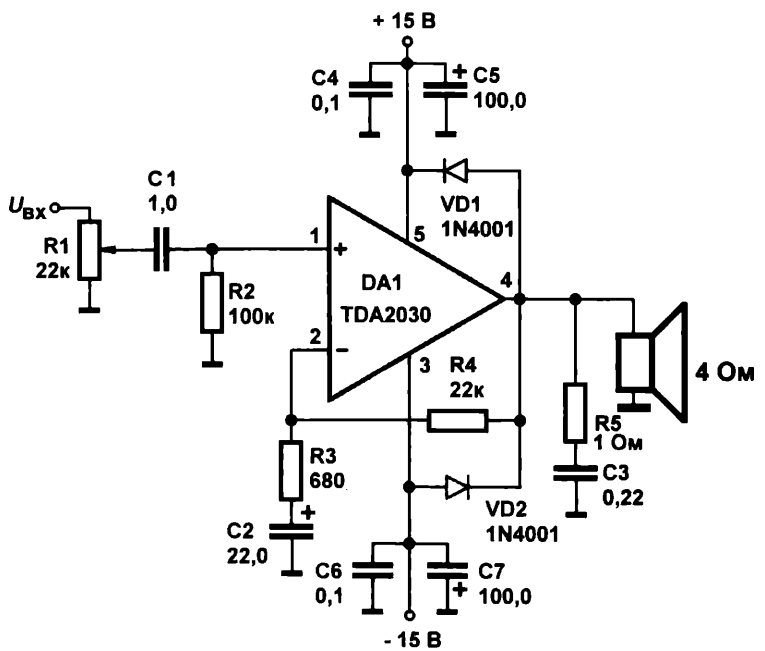


Рис. 11.7. Основная схема усилителя мощности звуковой частоты на микросхеме TDA2030

схемы из *главы 8*. Мало того, здесь даже установлен с помощью обратной связи тот же самый коэффициент усиления, примерно равный 30. Как будто взяли нашу схему и упаковали ее в отдельный корпус, обеспечив вывод наружу входов дифференциального усилителя, выхода двухтактного (push-pull) каскада усиления мощности и, естественно, выводов питания.

На самом деле так оно и есть — подавляющее большинство УМЗЧ имеет приблизительно одинаковую структуру, отличающуюся лишь в частностях, касающихся обеспечения качества или повышения эффективности работы. Скажем, в микросхеме TDA2030 коэффициент усиления по напряжению при разомкнутой цепи обратной связи, согласно уверениям производителя, равен примерно 30 000, а у нас он не более 2000–2500. Что, конечно, значительно увеличивает линейность усиления «фирменной» схемы и уменьшает уровень искажений (почему это так, мы узнаем в *главе 12*).

Остальные элементы схемы — вспомогательные. Конденсаторы C4–C7 — развязывающие по питанию (в нашей схеме тоже рекомендовалось их устанавливать), причем заметьте, что в целях лучшей защиты от помех и повышения устойчивости схемы здесь рекомендуется установить неполярные (например, керамические) конденсаторы (C4 и C6) параллельно с электролитическими (C5 и C7). Цепочка R5–C3, которой у нас не было, здесь обеспечивает повышение линейности усилителя при работе на индуктивную нагрузку. Диоды VD1–VD2 служат для предотвращения возможного выхода из строя выходных каскадов микросхемы при индуктивных выбросах напряжения, например, при включении питания (ох, до чего же нежные эти западные транзисторы!). Все электролитические конденсаторы должны быть рассчитаны на напряжение не менее 16 В.

В случае если усилитель все же «загудит» (хотя прямо об этом в тексте фирменной инструкции не сказано), здесь рекомендуется параллельно резистору обратной связи R4 установить цепочку из последовательно включенных резистора и конденсатора, которые ограничат полосу частот. При указанных на схеме номиналах всех остальных компонентов резистор должен быть равен 2,2 кОм, а конденсатор — не менее 0,5 нФ. Увеличение емкости конденсатора сверх этой величины ведет к ограничению полосы частот, но и к повышению устойчивости схемы.

Сама микросхема TDA2030 выпускается в корпусе TO220, знакомом нам по мощным транзисторам, только здесь он имеет не три вывода, а пять. Разводка выводов приведена на схеме, а для того чтобы определить расположение выводов, надо положить микросхему маркировкой вверх — тогда вывод номер 1 будет находиться первым слева (в однорядных корпусах микросхем и транзисторов ключ для определения начала отсчета выводов часто отсутствует, но первый вывод всегда расположен именно так).

Рекомендованная в инструкции площадь охлаждающего радиатора для выходной мощности 14 Вт должна составлять 350–400 см^2, однако, на мой взгляд, эта величина завышена, как минимум, вдвое. Впрочем, подобное заключение я могу подтвердить, кроме весьма приблизительной методики расчета из *главы 8*, лишь личным опытом, и оно не должно быть воспринято, как руководство к действию, — это со-

вет из той самой серии «на ваш страх и риск». Скорее всего, разработчики из фирмы ST Microelectronics взяли двукратный запас специально: во-первых, ориентируясь на наихудший случай, когда радиатор будет стоять горизонтально в какомнибудь тесном непроветриваемом пространстве (ведь мы говорили, что все расчеты радиаторов очень приблизительны!), и, во-вторых, чтобы уменьшить уровень искажений при больших мощностях из-за встроенного механизма тепловой защиты, о котором мы упоминали ранее.

На рис. 11.8 показано, как можно построить усилитель с удвоенной выходной мощностью при тех же напряжениях питания и используемых деталях. Это так называемая *мостовая схема*, которая представляет собой два идентичных усилителя, работающих на одну нагрузку в противофазе: когда на выходе одного усилителя положительный максимум напряжения, на другом — отрицательный. Таким образом, амплитуда и действующее значение напряжения на нагрузке возрастает ровно в два раза, соответственно растет и мощность, которая здесь составит при условии неискаженного сигнала почти 30 Вт.

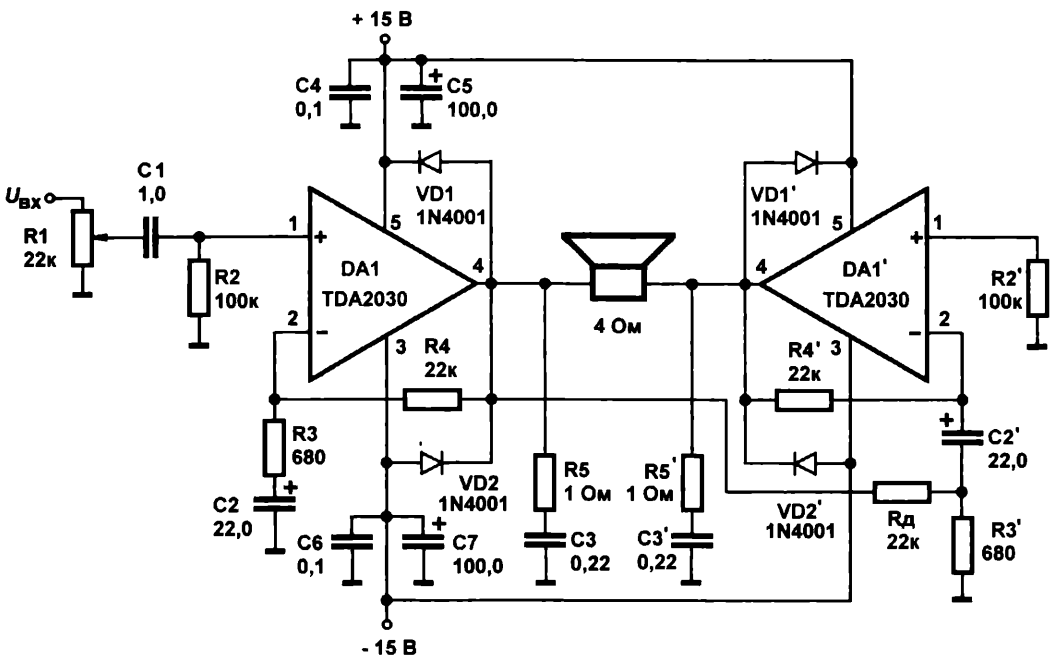


Рис. 11.8. Схема мостового усилителя звуковой частоты

Для того чтобы усилители работали именно так, как предусмотрено, обычный (неинвертирующий) вход второго усилителя заземляется, а входной сигнал для него поступает на другой (инвертирующий) вход, — туда же, куда и заведена его обратная связь. Сам этот входной сигнал берется с того места, куда поступает сигнал от первого усилителя (с левого по схеме вывода динамика), и ослабляется в той же степени, в которой оно было усилено первым усилителем, — вследствие равенства резисторов цепочки обратной связи R4–R3, задающей коэффициент усиления пер-

вого усилителя, и делителя Rд–R3′. То есть на вход 2 второго усилителя поступает фактически то же самое входное напряжение, но так как вход имеет противоположную полярность, то на выходе второго усилителя повторится сигнал на выходе первого, только в противофазе, чего мы и добивались. Отметим, что для такого усилителя придется соорудить более мощный источник питания, чем тот, что описан в *главе 9*.

Микроусилитель мощности

Не так уж редко возникает задача вывести звуковой сигнал на маломощный динамик или на головные наушники. Кроме очевидных применений вроде воспроизведения музыки, такой усилитель пригодился бы, скажем, в многочисленных конструкциях металлоискателей (их полно в Сети и радиолюбительской литературе), а также и в иных сигнальных устройствах. Такие усилители применяют и в различных звуковых модулях, ориентированных на управление от Arduino.

Существует, естественно, масса типов микросхем от разных производителей, которые осуществляют усиление звукового сигнала с возможностью выхода на низкоомную нагрузку, мы же остановимся на одной из самых когда-то популярных, — MC34119 (изготавливается не только фирмой Motorola, как можно было бы заключить из названия, но и другими фирмами, возможно, с другими буквенными префиксами). Микросхема выпускается в обычном корпусе всего с восемью выводами (DIP-8) и никаких радиаторов не требует.

Микросхема MC34119 обладает весьма неплохими характеристиками, основные из которых таковы:

- напряжение питания: 2–16 В (однополярное);
- сопротивление нагрузки: 8 Ом (минимальное);
- частота единичного усиления: 1,5 МГц;
- выходная мощность при напряжении питания 6 В и нагрузке 32 Ом: 250 мВт (коэффициент гармоник 0,5–1%);
- время готовности после включения питания — не более 0,36 с.

Самое главное — не надо ни о чем думать, все уже придумано за вас. Вариант типовой схемы включения приведен на рис. 11.9. Коэффициент усиления задается двумя резисторами R1 и R2 и равен их отношению R2/R1, т. е., в нашем случае, 25. Максимально возможная мощность в нагрузке (0,5 Вт) достигается при питании 12 В и нагрузке 32 Ом (головные наушники). В других сочетаниях нагрузки и питания такая мощность при допустимом уровне искажений не достигается. Обратите внимание, что динамик не имеет соединения с «землей» (что естественно для схемы с однополярным питанием). Имеется также интересная возможность выключения усилителя с помощью сигнала от логических микросхем (например, от микроконтроллера) — если подать на вывод 1 напряжение более 2 В, микросхема выключится и будет потреблять ток не более нескольких десятков микроампер (правда, сопротивление по этому входу не очень велико — 90 кОм, что создаст дополнительное потребление).

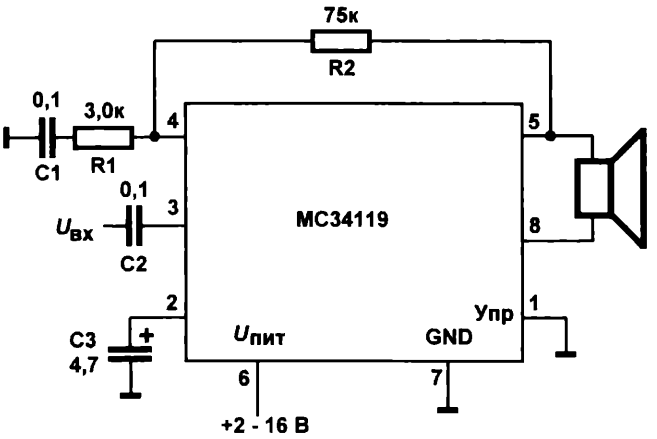
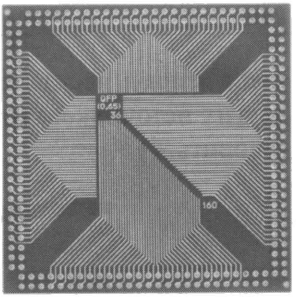


Рис. 11.9. Схема включения микросхемы MC34119

Другие схемы УМЗЧ на микросхемах вы, без сомнения, найдете в достаточном количестве в литературе и в Сети, а мы, наконец, вплотную займемся самыми универсальными аналоговыми микросхемами — операционными усилителями.



ГЛАВА 12

Самые универсальные
Обратная связь и операционные усилители

> Нам нужны надежные исполнители наших поручений не только для того, чтобы добиться успеха, но также и для того, чтобы не потерпеть неудачи.
>
> *А. Дюма. «Три мушкетера»*

Классическое определение гласит: *операционным усилителем* называется дифференциальный усилитель постоянного тока (УПТ) с большим коэффициентом усиления. Наличие в этом определении слов «постоянного тока» не означает, что ОУ усиливают только сигналы частотой 0 Гц, — здесь имеется в виду, что они могут усиливать сигналы, начиная с частоты 0 Гц. Слова «с большим коэффициентом усиления» означают, что он действительно большой, — хороший ОУ имеет коэффициент усиления порядка нескольких сотен тысяч или даже миллионов (куда там микросхеме TDA2030 с ее 30 тысячами!).

Название *операционный* закрепилось за такими усилителями исторически, потому что во времена господства ламповой техники они использовались в основном для моделирования различных математических операций (интегрирования, дифференцирования, суммирования и пр.) в так называемых *аналоговых* вычислительных машинах. Других применений у тех ОУ практически не было и быть не могло, потому что для достижения приемлемых характеристик не годилась не только ламповая, но и дискретно-транзисторная схемотехника. Настоящий переворот произошел только в середине 1960-х годов после пионерских работ по конструированию интегральных ОУ неоднократно уже упоминавшимся на этих страницах Робертом Видларом (рис. 12.1).

Разумеется, практически использовать ОУ можно только в схемах с отрицательной обратной связью (за одним исключением, описанным далее). Огромный коэффициент усиления приведет к тому, что без обратной связи такой усилитель будет находиться в состоянии, когда напряжение его выхода равно (или, как мы увидим дальше, почти равно) одному из напряжений питания, положительному или отрицательному, — такое состояние еще называют, по аналогии с транзисторами, *состоянием насыщения выхода*. В самом деле, чтобы получить на выходе напряжение 15 В, ОУ достаточно иметь на входе сигнал в несколько десятков микровольт, а

такой сигнал всегда имеется, — если это не наводка от промышленной сети или других источников, то достаточно и внутренних причин, о которых мы еще будем говорить.

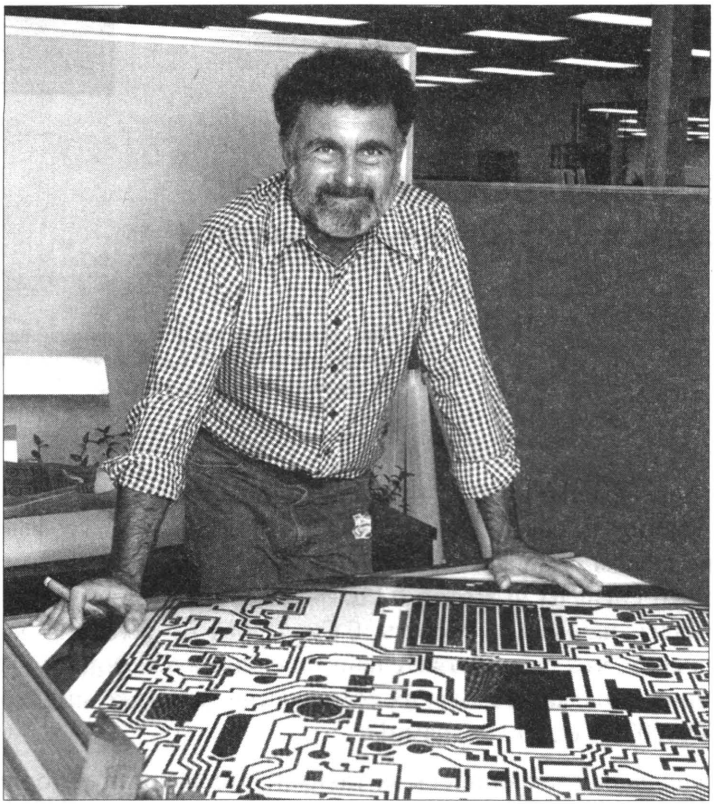


Рис. 12.1. Роберт Видлар (Robert J. Widlar), 1937–1991

Впрочем, есть и исключение — так называемые *компараторы* представляют собой ОУ, которые предназначены для использования без отрицательной обратной связи и иногда даже наоборот, с положительной обратной связью. К компараторам мы еще вернемся в этой главе, а пока рассмотрим некоторые общие принципы построения стандартных схем на ОУ.

Опасные связи

Согласно определению, *отрицательная* обратная связь — это связь выхода со входом, при которой часть выходного сигнала вычитается из входного. В противоположность отрицательной, в случае *положительной* обратной связи часть выходного сигнала со входным сигналом суммируется. Эти определения справедливы не только для усилителей и других электронных устройств, но и во всех других случаях, когда обратная связь имеет место. В общем случае можно воздействие обратной связи на некую систему описать так: наличие отрицательной обратной связи по-

вышает ее устойчивость, наличие положительной — наоборот, ведет к неустойчивости.

Принцип обратной связи

Впервые использовать принцип обратной связи в электронных усилителях с целью повышения их линейности, устойчивости и других эксплуатационных характеристик предложил американский инженер, сотрудник Лабораторий Белла (Bell Labs) Харольд Блэк в 1927 году. О сложности предмета говорит тот факт, что первый патент Блэка с описанием его усилителя имел объем целых 87 страниц, а всего он получил 347 патентов. Построение общей теории обратных связей было завершено математиком Хендриком Ваде Боде к 1945 году. В 1948 году Норберт Винер в своей знаменитой «Кибернетике» впервые показал, как использовать принцип обратной связи при рассмотрении любых систем: технических, биологических, социальных и пр.

Принцип действия обратных связей можно пояснить, скажем, на примере классической взаимосвязи спроса и предложения в экономике. Предположим, у нас имеется некая фирма, которая состоит из производственных структур и каналов сбыта. На входе такой системы — задание на производство, на выходе — объем произведенной продукции. Сколько нужно производить товара? Естественно, столько, сколько его могут потребить. В идеальной системе происходит следующее: фирма производит один экземпляр товара и, как только его покупают, немедленно выдает на прилавок следующий экземпляр. Если фирма произведет два экземпляра, и один из них на прилавке задержится, то производство приостанавливается до тех пор, пока этот экземпляр не купят. Здесь мы наблюдаем типичное действие отрицательной обратной связи, роль которой играет спрос, — лежащий на прилавке экземпляр товара как бы вычитается из задания на производство, и оно приостанавливается. Такая система очень устойчива и к тому же обладает множеством приятных свойств: не имеет перерасхода энергии и материалов, не приводит к перепроизводству или, в пределах мощности производства, наоборот, к дефициту. Объем перепроизводства может составить максимум один экземпляр, который придется выбросить, если спрос на него упал до нуля. Интересно, что примеры таких близких к идеалу производств можно встретить и в реальной жизни — это, скажем, торговля горячей выпечкой, когда следующий пирожок изготавливается только, если предыдущий уже купили.

Но в большинстве случаев в реальной жизни все обстоит гораздо сложнее: и прямых, и обратных связей существенно больше одной, реакция на спрос не может быть мгновенной, да и система не изолирована от всей остальной экономики. Посмотрим, например, что произойдет с нашей идеальной системой, если производство не может остановиться и возобновить работу мгновенно, или, что то же самое, сведения об изменении спроса поступают не сразу, а с некоторым запаздыванием. Предположим, фирма делает 10 экземпляров товара в день, и указанное запаздывание составляет также 1 день. Допустим, в какой-то из дней спрос упал на 2 штуки. Из-за запаздывания реакции на изменение спроса в этот день фирма произведет по-прежнему 10 штук, так что на следующее утро на прилавке их окажется 12. Если в этот день спрос по-прежнему будет составлять 8 штук, то к следующему утру на прилавке окажутся те же 12 экземпляров (8 произведенных — фирма отреагировала на изменение, плюс 4 оставшихся от предыдущего дня). Согласно реакции предыдущего дня и в этот день фирма также произведет всего 8 экземпляров. Но предпо-

ложим, что в этот день спрос внезапно возрос и составил 12 экземпляров, т. е. все имеющиеся оказались раскуплены. На следующее утро на прилавке будет лежать 8 штук (произведенных накануне), и если спрос сохранится, то 4 из 12 гипотетических клиентов уйдут неудовлетворенными. Им предложат зайти через два дня, и на следующий день фирма вынуждена будет произвести 4 + 12 = 16 экземпляров товара! Легко сообразить, что будет происходить дальше с производством и удовлетворением спроса, — система станет раскачиваться все сильнее и сильнее, пока в дело не вступят естественные ограничения: объем производства не может быть меньше нуля и больше фактической мощности производства (в случае электронных систем роль таких ограничений выполняет напряжение питания или достижимая мощность выходного каскада усиления). Работоспособность системы будет полностью нарушена — отрицательная обратная связь превратилась в положительную.

Поскольку реальные системы не могут иметь нулевое запаздывание по цепям прямой и обратной связи, возникает вопрос — какие меры нужно принять для того, чтобы система не раскачивалась все сильнее и сильнее? Обсуждение теории устойчивости систем с обратной связью в общем случае (скажем, известного метода Найквиста) увело бы нас слишком далеко, однако практические меры в простых системах не так уж и сложны. В основном они сводятся к тому, чтобы ограничить коэффициент усиления исходной системы и/или глубину обратной связи на таких частотах, когда отрицательная обратная связь начинает превращаться в положительную. Иными словами, чтобы фазовый сдвиг части выходного сигнала, поступающей обратно на вход, относительно самого входного сигнала не достигал бы близких к 180° величин при сравнимой или даже большей входного сигнала амплитуде этой части (поглядите на графики суммирования синусоидальных сигналов в *главе 4*, чтобы лучше понять, в чем тут дело).

Грубо эти частоты можно оценить следующим образом: если задержка сигнала в ОУ составляет 1 мкс, то (при мгновенной обратной связи, как это имеет место в случае ее осуществления с помощью резистивного делителя) при подаче сигнала частотой около 1 МГц с выхода на вход усилителя фазовый сдвиг составит ровно 180°, и усилитель будет раскачиваться неограниченно. Значит, нужно сделать так, чтобы усиление самого усилителя без обратной связи еще задолго до достижения указанной частоты падало и становилось равным единице ровно на частоте, соответствующей задержке. Это и есть так называемая *коррекция усилителей*. Причем, чем выше установленный с помощью обратной связи коэффициент усиления (т. е. чем меньше глубина обратной связи), тем выше допустимый порог по предельной частоте исходного усилителя — это обусловлено тем, что на вход при росте этого коэффициента передается меньшая часть выходного сигнала. Разница между фазой входного сигнала ОУ после суммирования и 180 градусами называется *запасом по фазе* — если он невелик, то при прохождении через усилитель, скажем, сигнала прямоугольной формы на выходе могут наблюдаться выбросы или даже небольшие колебания по фронту и по спаду выходного напряжения.

Наибольшую опасность несет в себе режим с установленным коэффициентом усиления, равным единице (т. е. использование ОУ в качестве *повторителя*). Роберт Видлар был сторонником того, чтобы переложить заботу о коррекции ОУ на плечи

пользователей, и первые его конструкции ОУ (например, μA702, выпускавшийся в нашей стране под названием 140УД1, или получивший широкую известность μA709) имели специальные выводы для коррекции с помощью внешних резисторов и конденсаторов. Разработчик мог в некоторых пределах выбирать ширину полосы пропускания частот в зависимости от установленного коэффициента усиления. Практически же этим никто не пользовался (подобно тому, как подавляющее большинство пользователей компьютерных программ работают с установками, введенными в них разработчиками по умолчанию), и такая возможность только приводила к необходимости введения в схему лишних компонентов. Так что в настоящее время выводы для внешней коррекции сохранились лишь для некоторых моделей высокочастотных ОУ, где полоса частот — действительно критичный фактор.

Заметки на полях

Кстати, а каковы в свете всего изложенного могут быть рекомендации нашим предпринимателям из производственной фирмы? Они совершенно аналогичны методам для обеспечения стабильности ОУ — нужно ограничить глубину обратной связи и коэффициент усиления на высоких частотах. Проще говоря, им следует при наличии запаздывания не пытаться реагировать на каждый проданный или непроданный экземпляр, а выпускать некое среднее количество товара в сутки, изменяя его только тогда, когда изменился средний объем продаж за промежуток времени, значительно больший времени реакции производства, — это равносильно ограничению усиления на высоких частотах. Если вы попробуете повторить рассуждения про нашу фирму, введя время реакции производства, скажем, на среднее за неделю количество проданных в сутки экземпляров, а не реагируя на продажи за каждый день, как ранее, то увидите, что система стала значительно устойчивее, хотя на ее выходе и могут наблюдаться некоторые высокочастотные колебания — т. е. количество товаров на складе может колебаться с частотой несколько экземпляров в день, но в среднем будет примерно следовать за колебаниями спроса.

Кстати, по всем этим причинам большинство ОУ представляют собой низкочастотные приборы — обычная частота единичного усиления f_1 (т. е. частота, на которой собственный коэффициент усиления снижается до 1) для распространенных типов не превышает 1–3 МГц. Например, для использованного в схеме лабораторного источника (см. *главу 9*) древнего μA741 эта частота равна 0,8 МГц. Для некоторых моделей ОУ, специально предназначенных для усиления постоянного тока и медленно меняющихся сигналов, частота f_1 еще меньше — скажем, для очень хорошего прецизионного ОУ MAX478/479 она равна всего 60 кГц. С другой стороны, есть и быстродействующие ОУ, для которых f_1 достигает десятков МГц. С частотой единичного усиления тесно связана другая характеристика ОУ — скорость нарастания выходного сигнала.

Не забудем также, что в реальных системах часто могут иметь место многочисленные так называемые *паразитные* обратные связи, учет которых весьма затруднен, если вообще возможен. Именно наличие таких связей приводит к «гудению» УМЗЧ даже в том случае, если с основными связями все в порядке, и в том числе именно для борьбы с этим явлением ставят развязывающие конденсаторы по питанию. О том, как используют в практических целях положительную обратную связь, мы поговорим в *главе 16*, а сейчас займемся детальным изучением свойств отрицательной обратной связи.

Основные свойства системы с отрицательной обратной связью

Отрицательная обратная связь в усилителях не только позволяет точно установить коэффициент усиления, как мы уже знаем из примеров в *главах 8* и *11*, но и приводит еще ко многим приятным улучшениям схемы. Попробуем разобраться, почему это так, и каково влияние характеристик реальных ОУ на параметры схемы.

На рис. 12.2 приведена обобщенная схема некоторой системы, охваченной отрицательной обратной связью. Коэффициент усиления К основной системы обычно больше единицы — в случае ОУ это и есть его собственный коэффициент усиления, который может достигать сотен тысяч и миллионов. Коэффициент передачи по обратной связи β обычно, наоборот, меньше единицы (хотя ничего, кроме вышеуказанных частотных ограничений, не мешает нам сделать его и больше единицы — просто вся система тогда будет не усиливать, а ослаблять сигнал). Кружок с плюсиком в нем означает устройство для суммирования сигналов — *сумматор*.

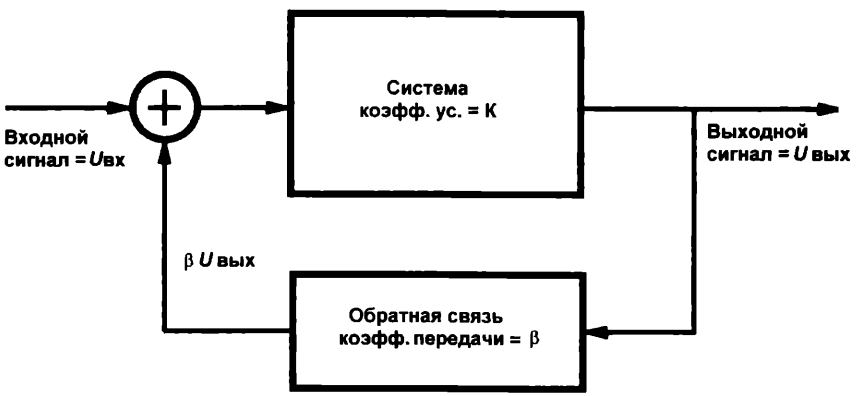


Рис. 12.2. Обобщенная схема системы с отрицательной обратной связью

Если разорвать петлю обратной связи, то сигнал на выходе $U_{вых}$ был бы равен $K \cdot U_{вх}$ (разумеется, в реальной системе напряжение питание его бы ограничило, но для наших рассуждений это неважно). Однако при действии обратной связи это не так. На вход выходной сигнал передается с коэффициентом ослабления β, и сигнал после сумматора, т. е. на входе основной системы, будет равен $U_{вх} - \beta U_{вых}$ (минус, т. к. обратная связь отрицательная). Этот сигнал передается на выход с коэффициентом К, т. е. $U_{вых} = K(U_{вх} - \beta U_{вых})$, или $U_{вых} = K \cdot U_{вх}/(1 + K \cdot \beta)$. Поскольку коэффициент передачи $K_{ус}$ всей системы по определению есть $U_{вых}/U_{вх}$, то в результате для него получаем следующую формулу:

$$K_{ус} = \frac{U_{вых}}{U_{вх}} = \frac{KU_{вх}}{U_{вх}(1+K\beta)} = \frac{K}{1+K\beta}. \quad (1)$$

Отсюда следует интересный вывод: если К много больше 1 (а в случае ОУ это действительно так с огромной степенью точности), то единицу в формуле (1) можно не

принимать во внимание, и коэффициент передачи будет выражаться простым соотношением:

$$К_{ус} = 1/\beta. \quad (2)$$

Формула (2) и означает, что коэффициент передачи входного сигнала на выход будет определяться только параметрами обратной связи и никак не зависит от характеристик системы. Причем, чем выше собственный коэффициент усиления системы К, тем точнее соблюдается это положение (мы об этом упоминали в *главе 11* при сравнении характеристик УМЗЧ, построенных на фирменной микросхеме и на дискретных элементах по схеме из *главы 8*).

Введение отрицательной обратной связи приводит также еще к некоторым последствиям. Для практических целей достаточно их просто запомнить, не углубляясь в математические выкладки:

- входы ОУ не потребляют тока (входное сопротивление ОУ практически равно бесконечности, точнее — увеличивается по сравнению с ОУ без обратной связи в К·β раз);
- ОУ с отрицательной обратной связью всегда стремится установить потенциалы на его входах равными между собой.

Характеристики конкретной схемы определяются соотношением собственного коэффициента усиления ОУ и коэффициента передачи системы с замкнутой обратной связью: чем выше это соотношение, тем ближе схема к идеалу. Интересно, что если на практике для обеспечения фактической независимости коэффициента усиления схемы от характеристик ОУ достаточно было бы иметь собственный коэффициент усиления всего в несколько тысяч (что и демонстрируют нам схемы УМЗЧ), то для того чтобы получить, например, действительно высокое входное сопротивление (измеряемое гигаомами и более), приходится увеличивать К до величин в сотни тысяч и более.

Отметим также, что использование обратной связи в указанной ранее степени уменьшает и выходное сопротивление всего усилителя, которое становится очень близким к нулю, — точнее, примерно равным $R_{вых}/(1 + К\beta)$, где $R_{вых}$ — это собственное выходное сопротивление ОУ, лежащее обычно в диапазоне сотен ом. Так что выходное сопротивление получается порядка 1 миллиома. Только не забывайте, что мощность выходного каскада ограниченна, и если вы его перегрузите, то от падения напряжения на нагрузке вас уже никакая обратная связь не спасет.

Для общего развития попутно заметим, что в системе, представленной на рис. 12.2, ничего не изменится, если схему перевернуть: считать за усилитель узел обратной связи, за узел обратной связи для него — сам усилитель, за входной сигнал — выходной и наоборот. Типичный пример такой двойственности мы увидим в схеме простейшего термостата далее. Все зависит только от терминологии, которая есть лишь вопрос удобства. Это хорошо иллюстрирует то философское положение, что мы слишком часто оперируем реальными вещами в зависимости от того, как мы их *назвали*, в то время как на самом деле их поведение совершенно от этого не зависит.

Базовые схемы усилителей на ОУ

Схема неинвертирующего усилителя (рис.12.3, *а*) нам хорошо знакома — именно она составляет основу лабораторного источника питания из *главы 9* (см. рис. 9.12). Анализ ее элементарно прост и исходит из рассмотренных ранее правил: $U_{ос} = U_{вх}$, т. е.:

$$U_{вх} = U_{вых} \cdot R2/(R1 + R2).$$

Тогда коэффициент усиления:

$$K_{ус} = U_{вых}/U_{вх} = (R1 + R2)/R2 = 1 + R1/R2,$$

каким мы его и предполагали в *главе 9*.

Единица, которая плюсуется к отношению сопротивлений резисторов обратной связи в выражении для коэффициента усиления, — очень важное дополнение, потому что если убрать в схеме неинвертирующего усилителя резистор R2 (т. е. принять его равным бесконечности), то отношение сопротивлений станет равным нулю, а $K_{ус}$ — равным 1. Соответствующая схема показана на рис. 12.3, *в* и носит название *повторителя*. Зачем она нужна, если ничего не усиливает? Эта схема обладает одним бесценным свойством: ее входное сопротивление равно практически

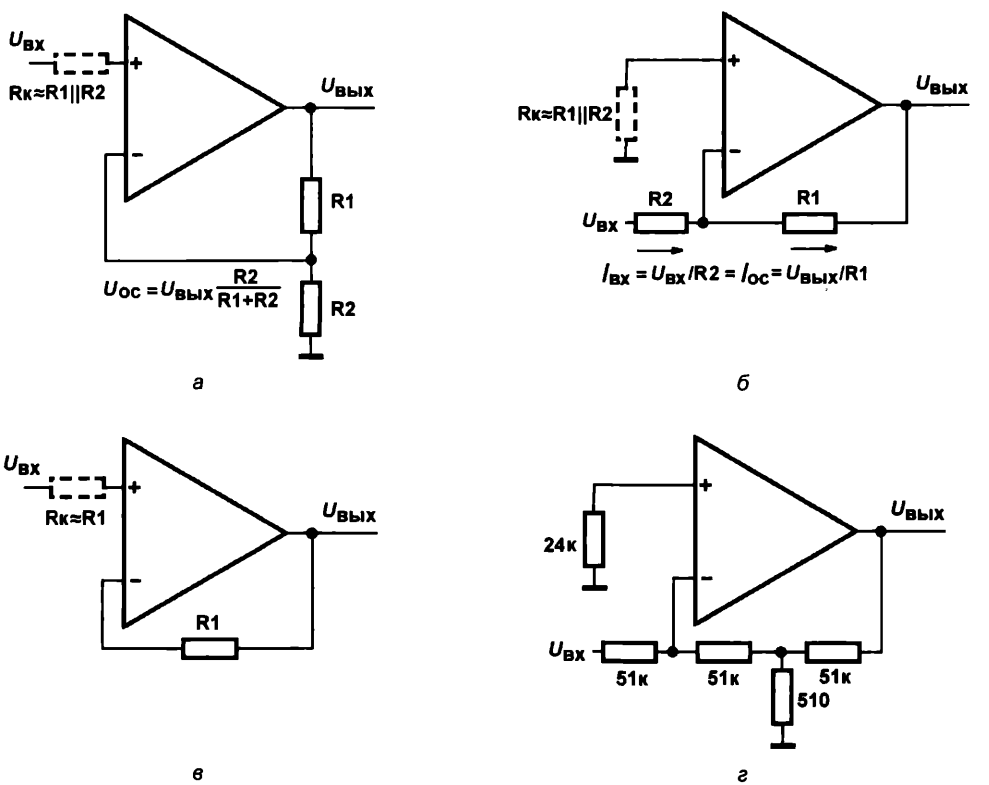


Рис. 12.3. Базовые схемы на ОУ: *а* — неинвертирующий усилитель; *б* — инвертирующий усилитель; *в* — повторитель; *г* — инвертирующий усилитель с высоким коэффициентом усиления

бесконечности, а выходное — практически нулю (в пределах, конечно, мощности выходного каскада, как мы уже говорили). Поэтому повторитель очень часто используют в случаях, когда нужно согласовать источник сигнала с высоким выходным сопротивлением с низкоомным приемником, и мы еще увидим примеры такого согласования.

В неинвертирующем усилителе (рис. 12.3, *а*) обратная связь носит название *обратной связи по напряжению*. В отличие от него, в инвертирующем усилителе (рис. 12.3, *б*) обратная связь имеет характер *обратной связи по току*, и вот почему. Так как здесь неинвертирующий вход имеет потенциал «земли», то и инвертирующий тоже *всегда будет иметь такой же потенциал*. Следовательно, от входа через резистор R2 потечет некий ток ($I_{вх}$). А раз мы договорились, что сам вход ОУ тока не потребляет, то этот ток должен куда-то деваться, и он потечет через резистор R1 на выход ОУ. Таким образом, входной ток ($I_{вх}$) и ток обратной связи ($I_{ос}$) — это один и тот же ток. Причем потенциал выхода ОУ вынужденно станет противоположным по знаку потенциалу входа — иначе току некуда будет течь. Чему равен коэффициент усиления? Поскольку $U_{вх}/R2 = U_{вых}/R1$, то $К_{ус} = U_{вых}/U_{вх} = R1/R2$. Обратите внимание, что в этом случае, в отличие от неинвертирующей схемы, единицу прибавлять не нужно. Поэтому R2 здесь является необходимым элементом схемы и не может равняться ни нулю, ни бесконечности, за исключением того случая, когда источник сигнала сам по себе представляет источник тока, а не напряжения, — тогда R2 из схемы можно (и нужно) исключить и подать токовый сигнал прямо на вход ОУ.

Похожее на приведенные соотношения уравнение для коэффициента усиления мы получали при рассмотрении транзисторного усилительного каскада в *главе 6*, где оно было равно отношению коллекторной нагрузки к сопротивлению в эмиттерной цепи. Это обусловлено тем, что в транзисторном каскаде также имеет место обратная связь.

Отметим, что подавать именно нулевой потенциал на неинвертирующий вход совершенно необязательно — скажем, если вы используете однополярный источник питания, то на неинвертирующий вход подается потенциал «искусственной средней точки», как это было сделано в схеме УМЗЧ из *главы 11*. Можно и любой другой, и мы еще будем этим широко пользоваться.

Максимальное значение выходного напряжения ОУ не всегда может равняться положительному или отрицательному напряжению питания — как правило, оно меньше его на величину порядка 0,5–1,5 В (простейшим примером для понимания того, почему это так, служит наш звуковой усилитель из *главы 8*). То же самое относится и к входным напряжениям — как правило, достигать значений питания не разрешается. Однако многие современные типы ОУ это все же позволяют, и выходное/входное допустимое напряжение у них достигает значений питания (чаще — только одно выходное). Это свойство в западной технической документации обозначается как *Rail-to-Rail* (т. е. «от шины до шины»), и на него нужно обращать внимание при выборе ОУ. При этом следует учитывать, что выходное напряжение может достигать напряжения питания только на холостом ходу, а с подключением нагрузки оно снижается.

Мы сейчас ведем речь об ОУ общего применения, к которым относятся старички μA741 (К140УД7), отечественные 140УД6, 140УД8 (последний — с полевыми транзисторами на входе) или счетверенный LM324 (который поддерживает Rail-to-Rail по входу и, частично — в отношении потенциала «земли», — по выходу), но, конечно, есть и более современные типы, многие из которых упоминаются далее. Как выбрать подходящий ОУ из всего разнообразия, имеющегося на рынке? Кроме очевидных характеристик, таких как ток потребления и допустимое напряжение питания, следует учитывать параметры, которые характеризуют неидеальность ОУ.

Неидеальность ОУ, ее последствия и борьба с ними

Если входное сопротивление неинвертирующего усилителя равно практически бесконечности, то инвертирующего почти в точности равно R2. Почти — по ряду различных причин, на которых мы не будем останавливаться, потому что эта разница несущественна для практических нужд. Важнее другое — входы реального ОУ все же потребляют ток, называемый *током смещения*, хотя и очень небольшой. Ток смещения на инвертирующем входе (в любой из двух схем) создаст падение напряжения на резисторе обратной связи, и оно воспринимается как входной сигнал. Если этот ток равен, к примеру, 0,2 мкА (казалось бы — так мало!), как у нашего любимого μA741, то при сопротивлении R1 = 1 МОм напряжение на выходе при отсутствии напряжения на входе достигнет 0,2 В.

Как обычно, в большинстве случаев важно не само по себе смещение, а его нестабильность. Борьба с этим явлением может вестись в трех направлениях: во-первых, не следует использовать в цепочке обратной связи сопротивления большого номинала, стандартный диапазон их — от килоом до десятков килоом. Если же при необходимости сохранить достаточно высокое входное сопротивление инвертирующего усилителя при большом коэффициенте усиления применение высокоомных резисторов желательно, то следует использовать схему, показанную на рис. 12.3, *г*. В этом случае вся цепочка в обратной связи работает, как один резистор с номинальным сопротивлением 5,1 МОм, и коэффициент усиления равен 100 при входном сопротивлении 50 кОм.

Во-вторых, в схему следует вводить компенсирующий резистор Rк (на рис. 12.3, *а–в* он показан пунктиром) — падение напряжения от тока смещения по неинвертирующему входу на нем отчасти компенсирует падение напряжения по входу инвертирующему. Тогда будет уже не столь важен сам ток смещения, сколько разница токов смещения, потребляемых по каждому из входов усилителя, которая определенно меньше каждого из токов.

В-третьих, если наличие именно тока смещения критично, то можно выбрать ОУ с малыми токами смещения, — например, с полевыми транзисторами на входе. Так как сами токи там исчезающе малы, то их разница, естественно, вообще может не приниматься во внимание.

Правда, в ОУ с полевыми транзисторами еще больше, чем в обычных ОУ, проявляется другая напасть — *входное напряжение сдвига*, которое есть величина разности напряжений между входами, при котором выходной сигнал ОУ в точности равен

нулю. (Я употребляю термин *напряжение сдвига* вслед за авторами [5], но часто используется и термин *напряжение смещения*. В любом случае здесь идет речь о напряжении, приведенном ко входу усилителя, — если смещение измерено на выходе, как это обычно и делается, то его надо поделить на коэффициент усиления схемы.) Возникает напряжение сдвига вследствие нестрогой идентичности транзисторов входных каскадов и для разных типов ОУ имеет довольно большой разброс: от десятков микровольт у прецизионных ОУ до единиц и даже десятков милливольт у ОУ с полевыми транзисторами. Естественно, оно, как и токи смещения, зависит от температуры. Бороться с напряжением сдвига гораздо сложнее, чем с токами смещения. Во многих типах ОУ традиционно имеются специальные выводы, присоединив к которым переменный резистор, можно регулировать смещение нуля на выходе. Однако пользоваться этой возможностью я не рекомендую (ничего хорошего в перекосе входного дифференциального каскада внешним вмешательством нет), и практических схем, где ей пользуются, я не встречал — подобно тому, как никто не пользуется возможностью внешней коррекции частотной характеристики, предусмотренной Видларом. Потому у большинства современных ОУ таких выводов нет.

В критичных случаях проще выбрать прецизионный ОУ с минимальным сдвигом, которых сейчас предлагается довольно много, — укажем на упоминавшиеся уже MAX478 (сдвоенный) и MAX479 (счетверенный), отличающиеся, кстати, исключительно широким диапазоном допустимых напряжений питания: от ±2,2 до ±18 В при очень небольшом потреблении — не более 25 мкА на каждый усилитель при максимальном напряжении питания. Правда, они довольно медленные (полоса усиления — десятки килогерц), но для схем по постоянному току быстродействие не имеет значения.

Подробности

К сожалению, усилители MAX478, которые мы будем в схемах этой книги широко использовать, фирмой MAXIM более не выпускаются, и прямой замены им не предусмотрено. Без изменений в схеме и в погрешностях, к сожалению, из всех возможных замен только операционники OPA241, OPA2241 и OPA4241 (соответственно, одинарный, сдвоенный и счетверенный) фирмы Burr-Brown до сих пор выпускаются в удобном для наших целей DIP-корпусе, и последние два полностью совпадают с MAX478 и MAX479 по выводам. Без учета корпусов и разводки выводов MAX478 и MAX479 могут быть заменены, например, на MAX44244, MAX44245 и MAX44248, или на модели фирмы Analog Devices OP193, OP293 и OP493 (эти два семейства еще и существенно быстрее — до мегагерца), а также OP2177 и OP4177 (с питанием от ±2,5 до ±15 В) и некоторые другие. Очень хорош прецизионный усилитель OP97 с питанием до ±20 В, смещением всего 20 мкВ и вдесятеро большим быстродействием, чем у MAX478, только он имеет повышенное потребление 0,6 мА и выпускается лишь в одинарном корпусе, а не сдвоенным или счетверенным. Если не жадничать и ограничиться питанием до ±5–6 В (как и будет в большинстве схем с ОУ далее), то малопотребляющих прецизионных ОУ можно найти больше: для примера укажем OP196, OP296 и OP496 (питание до ±6 В, потребление 60–80 мкА, Rail-to-Rail по входу и, практически, по выходу). Только при выборе учтите, что в характеристиках часто указывают суммарное напряжение питания (так, если указано, что максимальное напряжение питания может составлять 6 В, то двухполярное должно составлять, соответственно, не более ±3 В — таких типов сейчас выпускается больше всего). Ориентироваться нужно на напряжение смещения (Input Offset Voltage) не более 100–200 мкВ и учитывать возможности

размаха напряжения по выходу и по входу (в большинстве случаев они ограниченны). Если в схеме еще критично и потребление, то из кажущегося разнообразия типов останется не так уж и много вариантов.

Укажем — до кучи — еще один довольно экзотический, но, тем не менее, использовавшийся на практике до самого последнего времени способ борьбы со всеми этими сложностями: в случаях, требующих особой точности, всю схему вместе с ОУ попросту размещали в термостате! Как проектировать термостаты, мы узнаем в конце главы, а пока вернемся на землю.

Из тех ситуаций, когда требуется особо точное усиление некоего сигнала постоянного тока, все же имеется выход без использования подобной экзотики. Во-первых, это так называемые МДМ-усилители (от «модулятор-демодулятор», подобно «модему»). Их широко применяли еще в ламповые времена, т. к. тогда усилители постоянного тока с более-менее приемлемыми характеристиками строить было больше не из чего. В основе МДМ-усилителей лежит довольно остроумная идея — модулировать входным сигналом некое переменное напряжение относительно высокой частоты, усилить полученное переменное напряжение с меняющейся амплитудой, что значительно проще, потому что при этом все неприятные сдвиги-смещения остаются за бортом, после чего опять детектировать, выделив постоянную составляющую.

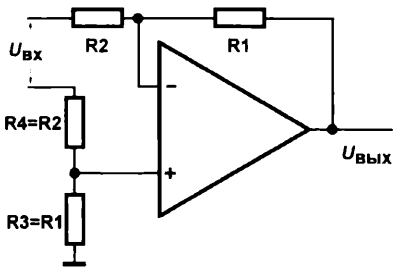
К сожалению, и здесь есть свои подводные камни (низкая предельная частота, невысокий коэффициент усиления, вероятность просачивания модулируемой частоты на выход, наконец, дороговизна), поэтому большее распространение получили так называемые *операционные усилители, стабилизированные прерыванием* (chopper stabilized amplifiers). Внутри таких ОУ автоматически производится периодическая компенсация смещения входных параметров. Для разработчика — это просто обычный ОУ, имеющий весьма высокие характеристики: типичное напряжение сдвига составляет 5–10 мкВ, коэффициент усиления — более 10^6, очень маленький входной ток смещения (порядка долей наноампера, как у ОУ с полевыми транзисторами) и т. п., при полосе единичного усиления порядка сотен килогерц или единиц мегагерц. К ним относятся отечественные 140УД21 и 149УД24, а также MAX420, MAX430/432, ICL7652, AD8629/AD8630, AD8638/AD8639 и др.

Дифференциальные усилители

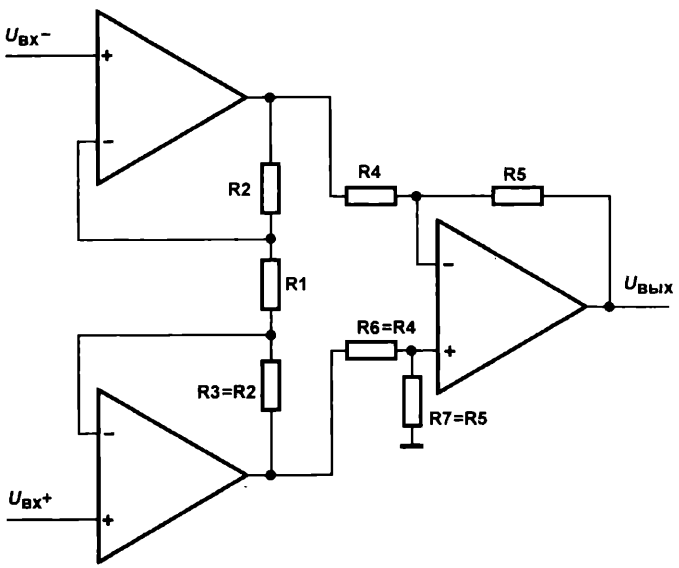
Кроме всего прочего, ОУ имеют замечательное свойство подавлять синфазный входной сигнал. *Синфазный сигнал*, в отличие от обычного, дифференциального — то напряжение, которое действует на оба входа сразу. Это свойство приводит не только к возможности выделять полезный сигнал на фоне значительных наводок, но и, что иногда еще важнее, к подавлению нестабильности источника питания — ведь изменение напряжения питания равносильно действию синфазного входного сигнала.

На рис. 12.4, *а* показана схема простейшего дифференциального усилителя. Делитель R3–R4 по неинвертирующему входу служит сразу двум целям: во-первых, он

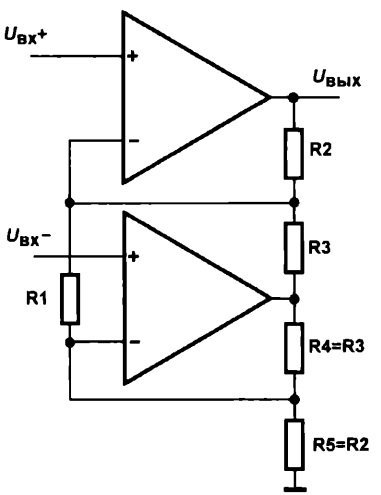
выравнивает входные сопротивления по входам (нетрудно показать, что так как потенциалы самих входов ОУ равны, то равны и входные сопротивления, — естественно, при указанном на схеме равенстве соответствующих резисторов), во-вторых, что еще важнее, он делит входной сигнал ровно в такой степени, чтобы коэффициенты усиления по инвертирующему и неинвертирующему входам сравнялись между собой. Именно при этом условии *коэффициент ослабления синфазного сигнала* (КОСС) будет максимальным. Для того чтобы получить действительно высокий КОСС (ослабление синфазного сигнала тысяч в десять раз и более), согласование сопротивлений должно быть как можно более точным, и в такой схеме следует применять прецизионные резисторы из ряда с погрешностью, по крайней мере, не превышающей 0,1%, причем лучше всего их еще и дополнительно подобрать по строгому равенству. Тогда вы действительно сможете без проблем выделить полезный сигнал в 1 мВ на фоне наводки в 1 В.



а



б



в

Рис. 12.4. Схемы дифференциальных усилителей: *а* — простой дифференциальный усилитель; *б* — классический инструментальный усилитель; *в* — упрощенный инструментальный усилитель

Понятно, что заниматься подобными извращениями при массовом производстве не с руки, да и входными сопротивлениями наш простейший дифференциальный усилитель отличается не в лучшую сторону, потому на практике эту схему применяют редко. Ко всему прочему, в ней еще и почти невозможно изменять коэффициент усиления в процессе работы, если вдруг это понадобится, — тогда потребуется менять одновременно два резистора, а куда денется в таком случае наше согласование?

Для того чтобы увеличить входное сопротивление, целесообразно добавить еще пару ОУ по каждому входу, включенных повторителями. Причем к раздуванию габаритов схемы это практически не приводит, т. к. специально для таких целей выпускают упоминавшиеся сдвоенные (dual) и счетверенные (quad) ОУ в одном корпусе. В результате получаем (рис. 12.4, *б*) классическую схему так называемого *инструментального усилителя*, в которой усиление можно менять одним резистором R1, не нарушая ничего в работе усилителя.

Коэффициент усиления такого усилителя определяется по формуле (при указанных на схеме соотношениях резисторов):

$$U_{вых} = (U_{вх+} - U_{вх-})\frac{R5}{R4}\left(1 + 2\frac{R2}{R1}\right).$$

Кстати, резисторы компенсации тока смещения здесь не нужны — токи эти по общим для системы инвертирующему и неинвертирующему входам взаимно компенсируют влияние друг друга, тем более если ОУ расположены на одном кристалле.

Если мы люди не гордые, и большой КОСС нам не требуется (т. е. в случае, когда помеха мала по сравнению с полезным сигналом), можно упростить схему инструментального усилителя. За исключением КОСС, схема на рис. 12.4, *в* обладает всеми достоинствами классической, но содержит на один ОУ меньше (значит, можно использовать сдвоенный, а не счетверенный чип), да и резисторов там поменьше. При указанных на схеме соотношениях резисторов выходное напряжение такого усилителя будет равно:

$$U_{вых} = (U_{вх+} - U_{вх-})\left(2\frac{R2}{R1} + \frac{R2}{R3} + 1\right).$$

Естественно, в этих усилителях решительно не рекомендуется подгонять ноль выходного напряжения с помощью нарушения баланса резисторов (например, R4/R5 и R6/R7 в схеме на рис. 12.4, *б*). В то же время иногда установка нуля необходима, т. к. начальное смещение выхода может быть, например, отрицательным (и не только вследствие смещения самих ОУ, но и по причине начального смещения у источника сигнала), и в случае, если весь диапазон изменения выходного напряжения должен располагаться в положительной области (скажем, при подаче его куда-нибудь на вход аналого-цифрового преобразователя, отрицательных напряжений не «понимающего»), вы можете потерять заметный кусок диапазона. Иногда для установки нуля рекомендуют воспользоваться корректирующими выводами одного из входных ОУ, но в сдвоенных и счетверенных вариантах эти выводы

обычно отсутствуют просто вследствие элементарной нехватки контактов корпуса, и это дополнительно удержит нас от такой глупости. В действительности установку нуля лучше осуществлять со стороны входов — подмешивая к одному из входных напряжений через развязывающий резистор небольшой ток коррекции. Как это осуществляется на практике, мы увидим, рассмотрев еще несколько типовых схем на ОУ.

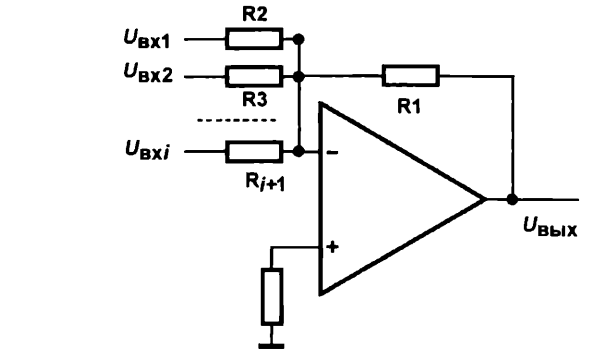
Другие распространенные схемы на ОУ

В начале главы я упоминал о том, что операционные усилители получили свое название потому, что применялись для моделирования математических операций. Схема *аналогового сумматора* (рис. 12.5, *а*) есть одна из таких классических схем. Представляет собой она обычный инвертирующий усилитель, на который подается несколько входных напряжений, — каждое от своего источника. Легко сообразить, что в этой схеме коэффициент усиления будет для каждого из входов определяться соотношением резистора обратной связи R1 и соответствующего входного резистора — так, как если бы остальных входов и не существовало. Потому сигнал на выходе будет равен (усиленной) сумме сигналов на входе (с противоположным знаком).

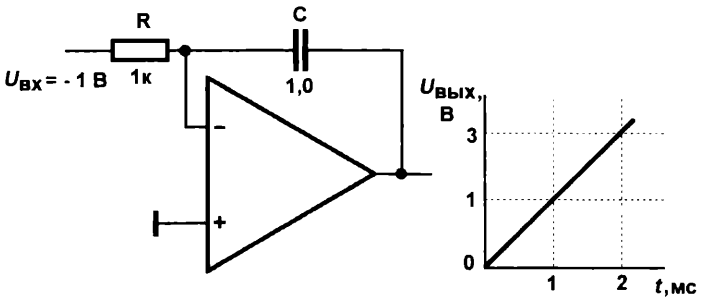
В простейшем случае, если все резисторы (включая и R1) равны между собой, то выходное напряжение будет равно просто сумме входных. Если же значения резисторов варьировать, то можно получить так называемую *взвешенную сумму* — когда каждый из входных сигналов вносит вклад в общее дело в соответствии с заданным ему коэффициентом. Кстати, если взять схему простого дифференциального усилителя (см. рис. 12.4, *а*) и заменить в ней резистор R4 такой же многовходовой цепочкой, то получится неинвертирующий сумматор. А если то же самое проделать еще и на инвертирующем входе, то получим сумматор, в котором весовые коэффициенты могут иметь разные знаки. Сумматор был неотъемлемой частью систем моделирования дифференциальных уравнений, для решения которых операционные усилители в составе аналоговых машин изначально и использовались.

Второй необходимой составляющей таких машин был *интегратор* на ОУ, схема которого приведена на рис. 12.5, *б*. Этот интегратор, в отличие от интегрирующей *RC*-цепочки из *главы 5*, действительно осуществляет операцию интегрирования в корректной форме. Например, если подать на его вход постоянное напряжение (отрицательное), то напряжение на выходе будет линейно возрастать со скоростью $U_{вх}/RC$ вольт в секунду (интеграл от константы есть прямая линия). Входной сигнал можно подать и на неинвертирующий вход, заземлив резистор R, — получим неинвертирующий интегратор. Можно также объединить интегратор с сумматором — тогда интегрирование будет осуществляться по сумме входных напряжений с соответствующими весовыми коэффициентами. Интеграторы, как и сумматоры, используются и по сей день в различных схемах.

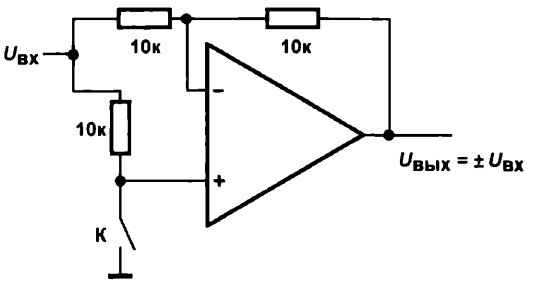
На рис. 12.5, *в* приведена любопытная схема, которая в зависимости от состояния ключа К меняет знак напряжения на выходе. Если К замкнут, то это инвертирующий усилитель с коэффициентом усиления, равным 1.



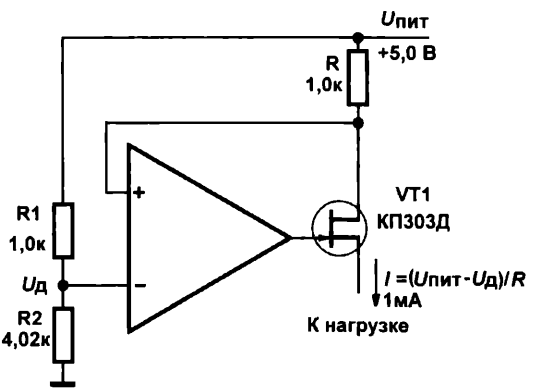
а



б



в



г

Рис. 12.5. Распространенные схемы на ОУ: *а* — аналоговый сумматор; *б* — интегратор; *в* — повторитель/инвертор; *г* — источник тока

Если же ключ разомкнут, то схема превращается в *повторитель* — ведь потенциалы во всех точках схемы в этом случае должны быть равны. В качестве ключа очень удобно использовать, скажем, транзистор или малогабаритное электронное реле — тогда такая схема может пригодиться для автоматического изменения знака усиления при необходимости отобразить отрицательную часть диапазона напряжений на входе в положительную область. Подобная задача может возникнуть, скажем, для датчиков, показывающих температуру, — и выше нуля градусов Цельсия, и ниже его характеристика должна быть возрастающей, т. к. абсолютное значение величины температуры возрастает в обоих случаях, в то время как сам сигнал с выхода датчика меняется линейно в одну сторону.

Еще одна давно обещанная и очень полезная схема (рис. 12.5, *г*) представляет собой почти идеальный источник тока с выходным сопротивлением, равным бесконечности. Здесь может использоваться однополярное питание, что и показано на схеме. Ток можно задавать как соотношением резисторов делителя R1–R2, так и резистором R. Обратите внимание, что отрицательная обратная связь подается на неинвертирующий выход ОУ, — поскольку здесь применен полевой транзистор с *n*-каналом, и стабилизируется его стоковое напряжение, которое есть инверсия напряжения на затворе. Если взять транзистор с *p*-каналом, то его в этой схеме нужно подключить наоборот — стоком в направлении нагрузки, а обратную связь, снимаемую с истока, подавать нормально, на инвертирующий вход.

Для высокой стабильности тока в этой схеме требуется столь же высокая стабильность напряжения питания, поэтому если важна абсолютная величина тока, то резисторы приходится питать от отдельного прецизионного стабилизатора (не только делитель R1–R2, но и цепь резистора R). От характеристик транзистора стабильность тока почти никак не зависит, единственное требование — чтобы начальный ток стока превышал установленный выходной ток схемы. Если применить не полевой, а биполярный транзистор, то будет иметь место некоторая зависимость выходного тока от изменений базового тока транзистора (ибо коллекторный ток отличается от эмиттерного на величину тока базы), потому чаще в таких источниках применяют полевые транзисторы.

Аналоговый генератор

Еще в *главе 2* я обещал, что нами будет построен генератор для домашней лаборатории. Вообще-то их нам требуется два: цифровой (выдающий прямоугольные импульсы) и аналоговый (генератор синусоидальных колебаний). Объединять их в одной конструкции, как это чаще всего делают, неудобно, потому что синусоидальный генератор должен выдавать переменное напряжение с амплитудой в минус и в плюс, а цифровой — однополярное пульсирующее, т. е. от нуля до плюса питания. Поэтому цифровым генератором мы займемся в *главе 16*, после изучения двоичных счетчиков, а пока сделаем аналоговый.

Принципиальная схема его приведена на рис. 12.6. Она выполнена по широко распространенной схеме генератора Вина — Робинсона. Для того чтобы генератор выдавал именно синусоидальные колебания, коэффициент усиления ОУ должен быть

в этой схеме равен ровно 3 — если он меньше, то генератор просто не запустится, если больше — верхушки синусоид начнут обрезаться, и в пределе выходные колебания станут прямоугольными.

Разумеется, подбором компонентов установить коэффициент усиления с нужной точностью невозможно. Поэтому применяют хитрый метод — в обратную связь ставят элемент, сопротивление которого зависит от среднего значения напряжения на нем. Проще всего оказалось использовать для этой цели термозависимые резисторы. В нашем случае используется термистор, у которого зависимость сопротивления от выделяющейся мощности имеет отрицательный наклон. В результате при увеличении амплитуды напряжения на выходе генератора его сопротивление падает, и нужный коэффициент устанавливается автоматически. Можно использовать также обычную маломощную лампочку для карманного фонарика — только наклон зависимости у нее положительный, потому ее следует ставить вместо резистора R2, а R1 тогда оставить постоянным. Для того чтобы обратная связь с лампочкой работала, от ОУ может понадобиться достаточно большой выходной ток, и тогда следует добавить к нему умощняющий выходной каскад на транзисторе (например, как в лабораторном источнике на рис. 9.12). Есть и более тонкие способы стабилизации коэффициента усиления (скажем, с использованием полевого транзистора в обратной связи, см. [19]), но опыт показывает, что и этот старинный рецепт, еще времен господства ламповой схемотехники, прекрасно работает.

Схему по рис. 12.6 можно собрать всю сразу. Здесь можно использовать любой ОУ общего применения. Показанный на схеме сдвоенный ОУ типа 140УД20 представляет собой два знакомых нам μA741 (140УД7), размещенных в одном корпусе. С ними генератор будет приемлемо работать до частот в несколько десятков килогерц. Напряжения питания могут составлять от ±5 до ±20 В, удобно выбрать напряжение около ±7–8 вольт, т. к. бóльшие амплитуды практически никогда не требуются. Термистор может быть любого типа, но не слишком большой по размерам, чтобы он разогревался малыми токами (напр. бусинковый отечественный СТ1-19, СТ3-19 или импортный каплевидный B57861-S близкого номинала).

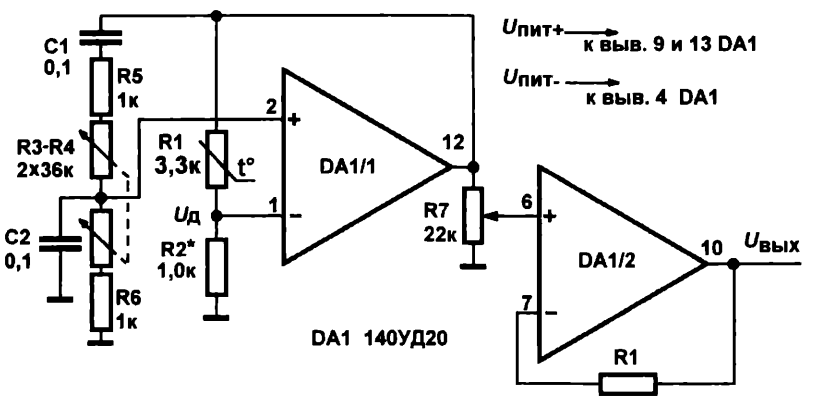


Рис. 12.6. Схема лабораторного генератора синусоидальных колебаний

Наладка будет заключаться в подборе резистора R2 под конкретный экземпляр термистора. Его нужно подобрать так, чтобы сигнал на выходе был чисто синусоидальным, без искажений. Частота регулируется сдвоенным резистором R3–R4. При указанных на схеме номиналах минимальная частота получится около 30 Гц, а максимальная — около 1 кГц. Чтобы расширить диапазон частот, придется поставить сдвоенный переключатель на несколько положений и изменять им емкости конденсаторов. Удобно, например, подобрать сопротивление резисторов R5 и R6 так, чтобы диапазон частот составлял 30–330 Гц, тогда, меняя с помощью переключателя емкости конденсаторов в десять раз (0,1 мкФ, 0,01 мкФ, 1 нФ), вы будете иметь перекрывающиеся диапазоны 30–330, 300–3300 и 3000–33 000 Гц. Обратите внимание, что никакой особой подгонки по равенству номиналов резисторов и конденсаторов не требуется, схема будет работать при любых (в разумных пределах) соотношениях номиналов, и равенство здесь выбрано только из соображений удобства расчета. Амплитуда сигнала на выходе регулируется потенциометром (R7 на схеме), а чтобы иметь низкое выходное сопротивление, добавлен повторитель на втором ОУ из корпуса.

Немало других интересных применений ОУ вы можете найти в многочисленной литературе — например, в классических трудах [4, 11]. А мы на этом с рассмотрением принципов использования ОУ закончим и займемся конструированием практических схем.

Релейное регулирование и термостаты

О термостатах мы здесь будем говорить так подробно потому, что это классический (и самый простой) случай систем автоматического регулирования. Последние есть частный случай систем автоматического управления, играющих сейчас огромную роль в самых разнообразных применениях электроники, от быта до космоса. На несложном примере термостатов можно познакомиться с основными проблемами автоматического регулирования и подводными камнями, лежащими на этом пути.

Термостат, т. е. устройство для поддержания температуры, — простейшее техническое устройство из класса *гомеостатов*, т. е. систем, которые автоматически поддерживают значение некоей величины на заданном уровне. Яркий пример хорошо всем знакомого гомеостата — наш собственный организм, в котором непрерывно с высочайшей точностью поддерживаются оптимальные значения таких величин, как температура, концентрация кислорода в крови, уровень адреналина и прочих параметров, причем практически независимо от вашей воли. Эти системы продолжают работать до тех пор, пока вы живы. Многие болезни есть следствие или причина расстройств гомеостатических функций организма, типичный случай — простуда, при которой в том числе нарушается работа термостатирующей системы, вследствие чего температура начинает расти.

Ключевой особенностью всех гомеостатов является обязательное наличие отрицательной обратной связи, на что обратил внимание еще отец кибернетики Норберт Винер. Поэтому любой гомеостат можно в принципе свести к обобщенной блок-схеме, показанной на рис. 12.2. На примере термостатов можно научиться созда-

вать несложные регуляторы любой физической величины — все зависит от датчика и исполнительного механизма, — причем особо не вникая в сложнейшую теорию автоматического регулирования и управления.

Конструировать термостаты одновременно и просто, и сложно. В частности, со схемотехнической точки зрения термостаты конструировать проще, чем регуляторы других величин. Процесс нагревания очень инерционен, и любой нагревательный элемент, кроме уж совсем миниатюрных (вроде нагревателей в головках термопринтеров), является естественным фильтром низких частот, как мы видели в предыдущем разделе. Поэтому при конструировании термостатов, как правило, не возникают какие бы то ни было проблемы, связанные с фазовыми сдвигами и возможным переходом всей системы в автоколебательный режим, не нужно возиться со сложными схемами дифференциальных или интегральных регуляторов (для других величин это может быть далеко не так). Зато это же самое свойство процесса нагревания заставляет внимательнее относиться к собственно конструкции термостата — стоит расположить датчик в неподходящем месте или не обеспечить равномерное распределение тепла, и качество регулирования резко падает, вплоть до полной неработоспособности устройства.

Термостат вообще

На рис. 12.7 приведена типовая структурная схема термостата. Следует отметить, что для полноты картины в приведенной структурной схеме не хватает одного компонента — холодильного устройства. Термостат, который показан на схеме, может поддерживать температуру только выше температуры окружающей среды — в чем, впрочем, большинство практических задач в области техники и заключается. Введение холодильного агрегата не представляет никаких трудностей теоретически, но есть не всегда тривиальная задача практически, т. к. холодильник — сами знаете, насколько это громоздкая конструкция. Сейчас мы рассмотрим работу схемы без охлаждения, а затем поглядим, с какого бока туда можно пристроить холодильник, если вдруг это понадобится.

> ***Заметки на полях***
>
> Интересно, что схема на рис. 12.7, кроме всего прочего, служит ярким примером упомянутого ранее положения о двойственности систем с обратной связью: за объект, подлежащий регулированию (на рис. 12.2 — верхний квадратик), здесь естественно принять среду, в которой мы поддерживаем температуру. В этом случае элементами обратной связи становятся усилитель и остальные компоненты схемы. Но ничто нам не мешает — и с технической точки зрения это гораздо логичнее — рассматривать в качестве регулируемого объекта усилитель, и тогда наоборот, все остальное есть лишь элементы обратной связи для него! В том, что и тот, и другой подходы равнозначны, вы убедитесь далее.

Итак, мы имеем некий объект регулирования (1), который условно показан на схеме, как бак с водой. Пусть сначала — сразу после включения системы — температура в нем ниже необходимой. Предположим, что датчик температуры (4) имеет характеристику с положительным наклоном — т. е. сигнал на нем увеличивается с увеличением температуры. Выходной сигнал этого термодатчика представляет со-

бой напряжение в некотором диапазоне, которое поступает на инвертирующий вход операционного усилителя (7). Конечно, не все датчики температуры выдают непосредственно напряжение на выходе, чаще у них от температуры зависит какой-нибудь физический параметр (например, сопротивление), но преобразовать этот параметр в напряжение обычно несложно, и мы еще этим будем заниматься.

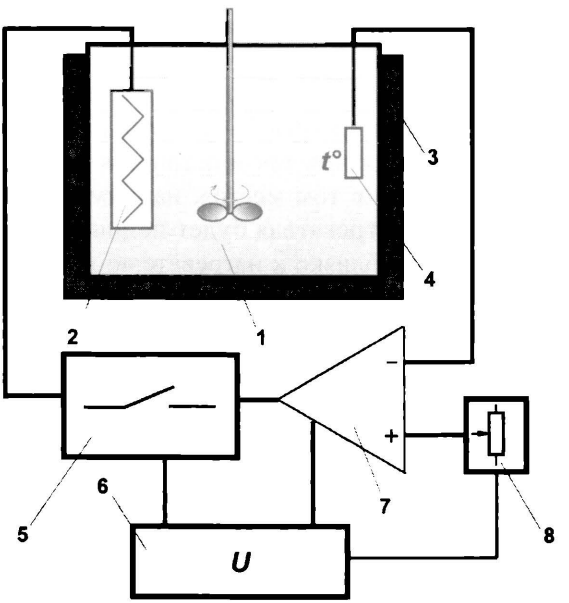


Рис. 12.7. Обобщенная схема термостата: 1 — объект регулирования; 2 — нагреватель; 3 — теплоизоляция; 4 — датчик температуры; 5 — исполнительное устройство; 6 — источник питания; 7 — усилитель; 8 — задающее устройство

Усилитель сравнивает сигнал датчика с сигналом, поступающим с *задающего устройства* (8), — так называется устройство, которым мы можем устанавливать нужную нам температуру в системе. В простейшем случае это переменный резистор, включенный по схеме потенциометра, с которого можно снимать напряжение в таком диапазоне, чтобы его крайние значения соответствовали сигналу с датчика при крайних значениях нужного нам диапазона температур.

Поскольку в начальный момент температура, как мы договорились, меньше заданной, то напряжение с термодатчика ниже напряжения сигнала с задающего устройства, и на выходе усилителя будет большое положительное напряжение насыщения выхода ОУ (меньшее напряжение поступает на инвертирующий вход, потому выход положителен). Это напряжение приведет в действие исполнительное устройство, которое на схеме условно показано в виде контактов реле, — в простейшем случае это и есть реле, электромеханическое или электронное, которое своими контактами подает напряжение от источника питания (например, прямо от бытовой сети) на нагреватель.

Обратная связь для усилителя замыкается через сам объект: когда нагреватель достаточно прогреет воду в баке, сигнал с термодатчика превысит установленный

с помощью задающего устройства уровень, напряжение на выходе усилителя упадет до нуля (или даже станет отрицательным — если питание усилителя двухполярное), исполнительное устройство снимет питание с нагревателя, и вода начнет остывать, пока температура датчика вновь не достигнет заданного значения — теперь уже «сверху», т. е. со стороны больших значений температуры, чем заданная.

Вы не поверите, сколько подводных камней кроется в такой, казалось бы, простой и понятной системе! Начнем с того общего положения, что *термостат всегда поддерживает температуру в той, и только в той точке, в которой установлен датчик*. Поэтому если вода в нашем баке плохо перемешивается, то обязательно возникнет ситуация локального перегрева — вплоть до того, что вокруг нагревателя вода может уже закипеть, а датчик так и останется холодным. Датчик при этом еще может быть установлен «не в том месте», например, слишком близко ко дну, в то время как теплая вода от нагревателя будет подниматься вверх. А если датчик установить, наоборот, слишком близко к нагревателю и, тем более, прямо над ним, в потоке поднимающейся теплой воды, то все произойдет наоборот — система сработает слишком рано, когда вода вокруг еще холодная.

Поэтому первое условие хорошего регулирования — как можно более интенсивное перемешивание среды, в которой температура регулируется. На рис. 12.7 для этой цели изображена мешалка, но, конечно, перемешивать можно и другими способами. Во многих случаях — когда это возможно — бак следует также укутывать теплоизоляцией, а стенки для более равномерного распределения температур делать металлическими. Обратите внимание, что системы климат-контроля в автомобилях, которые устроены в принципе точно так же, как описано (только среда — воздух, а не вода), для эффективной работы требуют минимума притока внешнего воздуха (что фактически равносильно теплоизоляции) и интенсивного его перемешивания.

Но и это далеко не все — напомним, что тепловые процессы крайне инерционны. И нагреватель, и датчик, и масса воды, и стенки бака обладают некоей теплоемкостью и, соответственно, тепловой инерцией, которая на много порядков превышает время срабатывания электронных устройств. Процессы нагревания и остывания протекают во времени примерно так же, как процесс заряда/разряда конденсатора через резистор (см. рис. 5.7), соответственно эти процессы также можно охарактеризовать аналогом постоянной времени *RC* — она так и называется *тепловой постоянной времени*. В данном случае наибольшая постоянная времени будет у системы «стенки бака — вода». Но нас даже больше интересует тепловая постоянная нагревателя (тепловую инерцию датчика пока учитывать не будем — обычно она много меньше остальных).

Что будет происходить в реальной системе? Когда температура, по мнению датчика, достигла заданной, электронные компоненты послушно выключат питание нагревателя. Но он еще некоторое время будет греть воду, отдавая туда тепло, запасенное за счет его собственной теплоемкости. Чем массивнее нагреватель, тем дольше будет длиться этот процесс. Мало того, это остаточное время также зависит от мощности нагревателя — чем он мощнее, тем также количество лишнего отданного тепла будет больше, потому что выше будет начальная температура внутри нагревателя. Произойдет перерегулирование — нагреватель давно выключен, а

температура некоторое время продолжает расти. В точности то же самое, но в обратную сторону, повторится при остывании системы — нагреватель включится, но ему нужно некоторое время, чтобы прогреться, и все это время температура будет продолжать падать.

Отсюда второе условие хорошего регулирования — масса нагревателя и его мощность должны быть минимально возможными, т. е. такими, чтобы при наихудших условиях (при максимальной разнице между установленным значением температуры и окружающей средой) только-только суметь «победить» потери тепла через стенки бака и через поверхность воды. На самом деле это положение в полной мере действительно только в нашей простейшей схеме релейного регулирования (нагреватель или выключен, или включен полностью). Можно ослабить требования, если регулировку производить другим способом — плавным изменением мощности пропорционально разнице температур. Схема такого пропорционального регулятора значительно сложнее простой релейной, но и требуется такой подход лишь в особо точных профессиональных термостатирующих устройствах. В быту практически всегда можно обойтись релейным регулированием.

Естественно, само по себе регулирование будет происходить только в определенных пределах температуры окружающей среды. Если температура среды выше или равна установленной, то бак никогда не остынет, а нагреватель никогда не включится, и система будет просто иметь температуру окружающей среды. Наоборот, при очень низкой температуре среды у нас может не справиться нагреватель — потери тепла превзойдут его мощность.

Холодильник в этой системе может понадобиться, если нам нужно поддерживать температуру ниже температуры окружающей среды или независимо от нее (в рассмотренном случае роль холодильника играет окружающая среда). Как же его сюда при необходимости пристроить? Это несложно — достаточно разместить охлаждающий агрегат в баке, а включать его, например, в противоположной фазе с нагревателем: когда нагреватель включен, холодильник выключен и наоборот. Но холодильник всегда имеет очень большую инерционность, и плавное регулирование мощности (холодопроизводительности) для обычных холодильных агрегатов недоступно. Можно, конечно, применять электронные холодильники на элементах Пельтье (см. *главу 9*), но при большой требуемой мощности (т. е. при задаче охлаждения достаточных объемов воды — уже в размере, например, домашнего аквариума) это не лучший способ. В этих случаях поступают иначе: холодильник нередко не выключают вовсе, а мощность нагревателя подбирают так, чтобы он в любом случае «побеждал» холодильник. При этом, увы, подавляющая часть потребляемой энергии уходит на взаимную «борьбу» холодильника и нагревателя, т. е. с точки зрения целевого назначения совершенно впустую. Зато качество регулирования оказывается на высоте.

Если же вообще нагреватель убрать, а холодильный агрегат включать через регулятор по рис. 12.7 (естественно, где-то инвертировав фазу — холодильник должен включаться при превышении заданной температуры, а не при снижении ее), мы получим в точности схему обычного домашнего холодильника — он ведь и предназначен для того, чтобы поддерживать температуру всегда ниже, чем температура

окружающей среды, и точно так же перестанет что-либо регулировать, если эта температура выйдет за пределы диапазона регулировки. Если холодильник выставить на мороз, то он никогда не нагреется, а если поставить в горячем цеху (или просто открыть дверцу), то никогда не выключится.

Вооружившись таким пониманием процессов, происходящих в термостатах, приступим к практическому их проектированию.

Простой термостат для аквариума

Простые конструкции термостатов, как мы говорили, используют релейный принцип регулирования: «включено-выключено». Иначе такие регуляторы еще называют *позиционными или релейными*. ОУ здесь удобно включать по схеме *компаратора* (от англ. *compare* — сравнивать) — т. е. без собственной обратной связи. Поскольку коэффициент усиления его в таком включении огромен, то он и будет находиться в одном из двух состояний: если сигнал с задающего устройства больше сигнала датчика, на выходе ОУ будем иметь практически положительное напряжение питания, если меньше — отрицательное (или ноль, если питание однополярное).

На рис. 12.8 приведена практическая схема терморегулятора для аквариума. Устроена она, как видите, довольно просто. Датчик температуры Rt представляет собой *термистор*, т. е. элемент, сопротивление которого падает с увеличением температуры (подробнее о термисторах см. *главу 13*), и сигнал на инвертирующем входе ОУ также будет падать (конденсатор C1 обеспечивает сглаживание наведенных помех). С этим связан один нюанс — в рассмотренной ранее обобщенной схеме сигнал датчика возрастал, но включен он был также в инвертирующий вход.

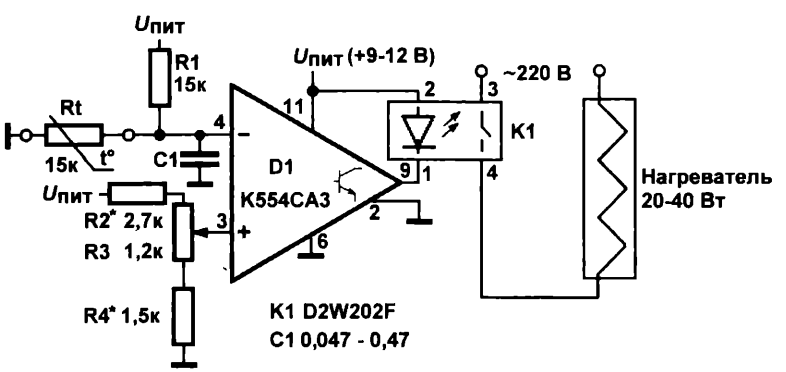


Рис. 12.8. Принципиальная схема терморегулятора для аквариума

Все дело тут в необычном устройстве выходного каскада компаратора 554СА3 (импортный аналог — LM311 в 14-выводном DIP-корпусе). У него в качестве оконечного усилителя используется довольно мощный *n-p-n*-транзистор (напряжение коллектор-эмиттер до 40 В и ток коллектора до 50 мА), который соединяется с остальной схемой внутри корпуса только базой, а эмиттер и коллектор выведены

наружу (эмиттер — вывод 2, коллектор — вывод 9). На самом деле напрямую выведен только коллектор, а эмиттер подключен несколько сложнее, но это для нас не имеет значения. Если мы присоединим эмиттер к «земле», то получим так называемую *схему с открытым коллектором*, и именно так и делается в большинстве практических применений компаратора. Заметим, что в техническом описании компаратора LM311 фирмы National Semiconductor приведено большое количество типовых схем таких применений.

Чтобы получить при этом на выходе напряжение, следует в коллекторную цепь установить нагрузку — в простейшем случае это резистор, но можно подсоединить и обмотку реле или, скажем, лампочку. У нас нагрузкой служит входной светодиод оптоэлектронного реле — токоограничивающий резистор для него устанавливать не требуется, т. к. у этого типа (D2W202F) он уже имеется внутри реле. При наличии датчика с положительным наклоном (например, обычного медного терморезистора, мы их будем изучать в *главе 13*) следует поменять местами либо R1 и Rt, либо входы компаратора 3 и 4.

Заметки на полях

Возникает вопрос: при таком выходном каскаде какой смысл приобретут понятия «инвертирующий» и «неинвертирующий» входы компаратора? Эти наименования были присвоены с учетом того, что одно из основных назначений такого типа компараторов — преобразование аналогового сигнала в логические уровни. При этом выходной транзистор включается обычным способом, с общим эмиттером и нагрузкой в цепи коллектора. Тогда названия входов обретают следующий смысл: при превышении напряжением на инвертирующем входе напряжения на неинвертирующем, на выходе (т. е. на коллекторе выходного транзистора) будет логический ноль (транзистор открыт), и наоборот. Если мы применим это рассуждение к нашему случаю, то увидим, что выходной транзистор откроется, когда температура станет *ниже* необходимой (т. е. когда сопротивление термистора велико). А нам это и надо — при этом реле включится и подключит нагреватель. При увеличении температуры сопротивление термистора упадет, и когда напряжение на делителе R1–Rt станет меньше, чем на делителе R2–R3–R4, то транзистор закроется и отключит через реле нагреватель.

В нашем случае целесообразно использовать именно термистор, потому что у него высокая (3–4 %/°C) крутизна, отчего и чувствительность, и помехоустойчивость системы возрастают. А характерная для термисторов нелинейность нас не волнует — в диапазоне температур для аквариума изменение крутизны датчика можно вообще не принимать во внимание, а в более широком диапазоне (как далее в схеме термостата для водонагревателя) крутизна уменьшится примерно в полтора раза при увеличении температуры на 60–70°, что просто означает некоторое уменьшение чувствительности.

Здесь можно использовать термистор любого типа (например, классический ММТ-1 или подробно описанный в *главе 13* B57164-K) с номинальным (при 20 °C) сопротивлением от нескольких килоом до нескольких десятков килоом. При этом сопротивление резистора R1 должно быть примерно равно номинальному сопротивлению термистора при 20 °C или несколько меньше этого значения (чем оно меньше, тем хуже для термистора, т. к. он может перегреваться питающим током, однако, чем оно больше, тем меньше рабочий диапазон напряжений).

Сам датчик можно изготовить следующим образом (рис. 12.9): термистор с припаянными к нему достаточно длинными выводами помещается в металлическую или пластмассовую трубку и заливается эпоксидной смолой. Для того чтобы смола не вытекала, пока не затвердеет, нужно временно залепить нижнюю часть трубки пластилином. Одновременно в трубке с одного конца закрепляется «ухо» для крепления датчика, которое можно изготовить просто из проволочной петельки. Чтобы исключить выщелачивание вредных веществ из эпоксидной смолы во время эксплуатации датчика, нужно дополнительно покрыть датчик водостойким лаком. Подойдут уретановые лаки для лакирования печатных плат, автомобильные эмали горячего отверждения (или, в крайнем случае, обычная натуральная олифа, которая имеет очень высокую водостойкость, но, к сожалению, сохнуть может при комнатной температуре неделями). Операции окраски можно избежать, если использовать вместо эпоксидной смолы силиконовый герметик, которым, однако, аккуратно заполнить внутреннюю полость трубки значительно сложнее (трубка при этом должна быть, естественно, либо пластиковая, либо нержавеющая).

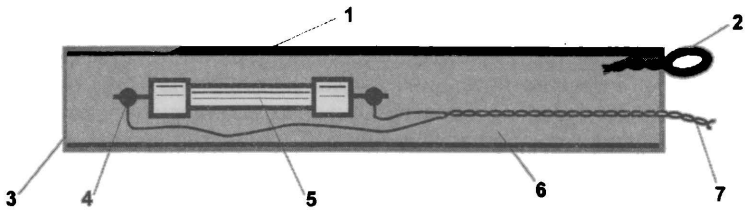


Рис. 12.9. Датчик для терморегулятора по схеме на рис. 12.8:
1 — металлическая или пластмассовая трубка; 2 — «ухо» для крепления; 3 — слой водостойкого лака; 4 — место пайки вывода термистора; 5 — термистор; 6 — эпоксидная смола; 7 — выводы

Электронное реле типа D2W202F (фирмы CRYDOM) можно заменить на любое другое подобное реле или даже на простое электромеханическое, только в последнем случае нужно учитывать то, что написано далее о дребезге контактов.

Настройка регулятора сводится к тому, чтобы подобрать сопротивления R2 и R4 под конкретный экземпляр термистора. Сначала мы подсоединяем вместо них переменные резисторы, выводим движок потенциометра R3 в верхнее по схеме положение, погружаем датчик в воду с температурой 18 °С (это будет нижний предел диапазона регулировки температуры) и, изменяя величину R2, фиксируем момент срабатывания реле (можно просто подсоединить к его контактам тестер в режиме «прозвонки», но удобнее временно вместо нагрузки подсоединить маломощную лампочку накаливания). Далее погружаем датчик в воду с температурой 32 °С (верхний предел), выводим R3 в нижнее положение и подбираем R4 до срабатывания реле. При этом у нас нижний предел также «уедет», поэтому придется сделать несколько итераций, чтобы добиться требуемого результата, и при этом нужно следить за температурой воды — она в обоих случаях не должна меняться от раза к разу. Чтобы не устраивать столь долгую «песню», можно просто измерить напряжение на делителе R1–Rt при нужных температурах и рассчитать величины сопротивлений R4 и R2 заранее, а затем при необходимости их подкорректировать (хотя

этого обычно не требуется — какая разница, будет у нас нижний предел 18 °С или 17 °С? Главное, чтобы мы его знали).

В окончательной конструкции регулировочный резистор R2 снабжается шкалой, по которой мы будем устанавливать поддерживаемую температуру. Следует учесть, что при использовании термистора шкала эта будет неравномерная — к концу промежутки между делениями будут короче, т. к. чувствительность термистора с температурой падает. Поэтому шкалу следует изготовить эмпирическим методом: полностью отлаженный термостат подключается к небольшой емкости с водой (чтобы нагревание и остывание шли не слишком долго), а затем отмечаются углы поворота движка резистора R2, которые соответствуют различным установившимся температурам — именно установившимся, а не температурам в момент срабатывания реле, т. к. они могут отличаться. Эта процедура носит название *калибровки*.

Кстати, а как же здесь быть с теплоизоляцией и перемешиванием, о необходимости которых «так долго говорили большевики»? Теплоизоляцией, естественно, придется пожертвовать, но при столь небольших перепадах температур между водой и окружающей средой она и не требуется. А вот насчет перемешивания «большевики» совершенно правы — без него ничего не выйдет. Поэтому терморегулятор в аквариуме можно использовать только в сочетании с аэратором воды, который очень хорошо ее перемешивает, причем рассеиватель аэратора должен быть размещен на самом дне аквариума. При этом датчик подвешивают на половине высоты аквариума, а нагреватель — также вблизи дна.

Нагреватель указанной мощности лучше всего купить в магазинах для аквариумистов, но можно и изготовить его самостоятельно из мощного остеклованного резистора типа ПЭВ сопротивлением около 1 кОм. Мощность резистора может быть не более 5–10 Вт — в воде коэффициент теплоотдачи возрастает во много раз. Только не забудьте, что такой нагреватель, подобно обычному кипятильнику, нельзя включать на воздухе. Выводы следует тщательно изолировать: сначала они покрываются лаком, затем изолируются термоусадочным кембриком, затем поверх него также покрываются в несколько слоев водостойким лаком или силиконовым герметиком. После изготовления качество изоляции следует проверить: погрузите нагреватель в теплый раствор соли и измерьте сопротивление между выводами и раствором — на *всех* пределах измерения сопротивления мультиметр должен показывать полный разрыв цепи.

Подчеркнем еще раз — если температура воздуха в помещении сама достигнет заданной и превысит ее, то терморегулятор наш перестанет включаться, и температура воды окажется равной температуре воздуха (точнее, она всегда будет несколько ниже ее — из-за испарения с поверхности). Описанный термостат предназначен только для подогрева воды и стабилизации ее температуры на некотором уровне, заведомо более высоком, чем температура окружающей среды. И его использование наиболее актуально зимой, когда отопление в наших квартирах работает сами знаете как.

О гистерезисе

Во всем этом деле есть еще один нюанс. Что будет происходить в момент, когда напряжения на входах компаратора сравняются? Чувствительность у компаратора огромная, а как в сигнале датчика, так и на выводе задающего делителя всегда присутствует хоть маленькая, но помеха, и конденсатор C1 ее не устранит полностью, — если даже все идеально заэкранировать, роль помехи сыграют собственные шумы компонентов схемы, которые имеются принципиально (если температура, конечно, отличается от абсолютного нуля). Поэтому в момент равенства напряжений на выходе компаратора появится «дребезг» — он будет быстро-быстро переключаться туда-сюда, переключая и реле тоже. В случаях, подобных нашему, при использовании в качестве исполнительного механизма электронного реле с zero-контролем (или, скажем, транзистора), на этот дребезг можно закрыть глаза. Отсутствует дребезг и в схемах с пропорциональным регулированием, пример которого мы увидим далее. Но в других случаях нечеткое срабатывание приводит к разным неприятным последствиям: для обычного тиристора (см. *главу 10*) это помехи, для электромеханических реле, сверх того, еще и быстрый износ контактов, да и просто далеко не услаждающий слух шум.

Для того чтобы избежать этого явления, в схему вводят так называемый *гистерезис* (от греческого *hysteresis* — отставание реакции от вызвавшего ее внешнего воздействия). На рис. 12.10 показана идея того, как это делается с помощью положительной обратной связи, охватывающей компаратор, хотя, как мы увидим далее, делать именно так необязательно. Напряжение питания всей схемы в данном случае однополярное. Пусть напряжение $U_{вх}$ ниже напряжения на делителе $U_{зад}$, тогда на выходе компаратора напряжение равно положительному напряжению питания (все компараторы поддерживают полный диапазон напряжений по выходу — Rail-to-Rail). В этом случае резистор R1 шунтирует R2, и напряжение $U_{зад}$ больше того значения, которое оно бы имело в отсутствие резистора R1, — при указанных на схеме номиналах и напряжении питания оно равно 5,24 В. Когда $U_{вх}$ увеличится и достигнет $U_{зад}$, компаратор переключится, и напряжение на выходе станет равным нулю. Резистор R1 теперь шунтирует R3, и напряжение на делителе $U_{зад}$ станет ниже — оно

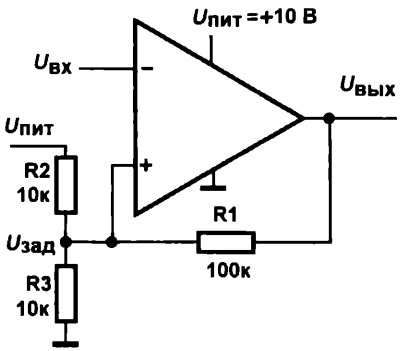


Рис. 12.10. Схема компаратора с гистерезисом

будет равно 4,76 В. Теперь небольшая помеха не страшна — чтобы переключиться обратно, напряжение на входе должно опуститься аж на целых 0,48 вольта. Состояние компаратора при переключении как бы фиксируется.

Величина разницы в порогах (0,48 В в данном случае) называется *зоной нечувствительности*. Естественно, наличие этой зоны усугубляет влияние тепловой инерции нагревателя — включение-выключение нагревателя происходит позже, чем надо бы, и перерегулирование растет. Поэтому величину этой зоны при необходимости качественного регулирования нужно выбирать очень аккуратно. Сложность введения гистерезиса таким, если можно так выразиться, «академическим» способом в реальных схемах обусловлена тем обстоятельством, что половинки входного делителя обычно не равны друг другу, к тому же чаще всего (как в нашем случае) делитель этот есть переменное сопротивление, и зона нечувствительности будет зависеть от положения движка потенциометра.

Должен сказать, что обычные электромеханические реле сами по себе имеют гистерезисную характеристику — как мы отмечали в *главе 7*, напряжение срабатывания у них может в несколько раз превышать напряжение отпускания. Так что простое снижение чувствительности компаратора (превращенного тогда в обычный ОУ с отрицательной обратной связью), казалось бы, могло бы нам в этом случае помочь. И все же оно не поможет, и дребезг будет появляться все равно, потому что выходное напряжение ОУ с наложенной на него помехой тогда станет нарастать очень медленно, и в момент достижения напряжения срабатывания реле начнет вести себя очень неуверенно — несколько раз пытаясь сработать, но затем откатываясь назад, и издавая при этом характерное такое жужжание. Поэтому будет лучше и для нас, и для реле, если мы введем контролируемый гистерезис по всем правилам. Один из способов, как это можно сделать практически, продемонстрирован в следующем разделе. К устройствам с гистерезисом мы вернемся в *главе 16*, при рассмотрении автоколебательных генераторов и одновибраторов — в логических сериях аналогичный элемент носит название «триггер Шмидта».

Терморегулятор «для дома для семьи»

Обычное устройство для нагревания воды в условиях отсутствия центрального горячего водоснабжения (например, в дачном домике) состоит из бака на 5–20 л со встроенным электронагревателем (ТЭНом) мощностью 1–2 КВт. Использовать его без терморегулятора неудобно — приходится внимательно следить за тем, чтобы вода не закипела, да и получается она либо слишком горячая, либо наоборот — недогретая.

На рис. 12.11 изображена схема термостата на этот случай[1]. Она только на вид кажется сложной, а на самом деле отличается от схемы термостата для аквариума практически только тем, что в ней выбрано значительно более мощное электронное реле (до 10 А при естественном, без обдува, охлаждении) и введены дополнитель-

[1] Эта конструкция автора опубликована в журнале «Радио», 2004, № 9.

ные элементы (два маломощных электронных реле и два тумблера), в основном для обеспечения различных режимов работы. Режимы эти следующие:

- автоматический термостатирующий;
- автоматический однократный с отключением по достижении нужной температуры («режим электрочайника»);
- ручной с подключением ТЭНа напрямую к сети.

Сначала отвлечемся от режимов и посмотрим, как работает основная схема регулирования — в ней есть небольшое отличие от схемы на рис. 12.8, которое заключается в том, что в схему введен резистор R4 небольшого номинала, шунтированный контактами маломощного реле К2.

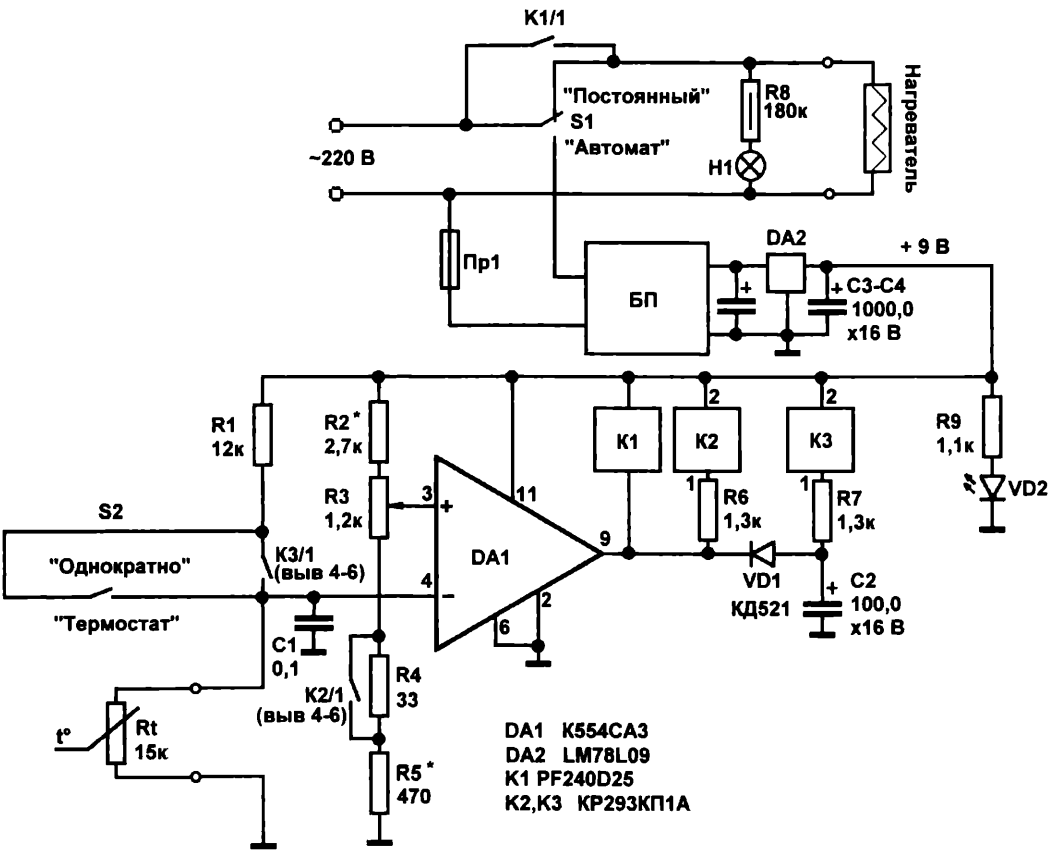


Рис. 12.11. Схема термостата для нагревания воды

После включения питания, если температура еще ниже установленной, срабатывает не только основное мощное реле К1, но и реле К2 (встроенных токоограничивающих резисторов в реле этого типа нет, и с этой целью установлены резисторы R6 и R7). Контакты его замкнуты, и резистор R4 не участвует в работе схемы. По мере увеличения температуры напряжение на датчике падает, и в какой-то момент вре-

мени выходной транзистор компаратора разрывает цепь питания К1 — нагреватель обесточивается. Одновременно отключается реле К2 и резистор R4 включается в цепь делителя R2–R3–R4–R5, еще больше увеличивая разницу напряжений между выводами компаратора. По мере остывания воды напряжение на датчике повышается, и в какой-то момент компаратор снова срабатывает, подключая нагрузку через реле К1. Контакты К2 при этом опять шунтируют резистор R4, и это тоже увеличивает разницу напряжений, но в теперь «в другую сторону».

Как видите, мы получили типичную гистерезисную характеристику — хотя мы здесь и используем электронное реле с zero-коррекцией, но коммутирует оно мощную нагрузку, и слишком частые изменения тока в маломощной деревенской сети в момент включения и выключения реле нам совершенно ни к чему. Разумеется, наличие резистора R4 несколько увеличивает нестабильность поддержания температуры — при приведенных на схеме номиналах разница между температурой включения и выключения составит от 1 до 1,5 °С (например, при установленной температуре в 35 °С нагреватель включится, когда температура упадет до 34 °С, а выключится — когда она достигнет 35,5 °С), однако нам более высокая стабильность здесь совершенно не требуется.

Теперь разберемся с режимами. Сначала — с режимом электрочайника, для обеспечения которого в схему введено еще одно маломощное реле К3, включенное, как видите, довольно хитрым образом. Если тумблер S2 находится в положении «Термостат» (т. е. контакты его замкнуты и шунтируют контакты реле К3), то реле К3 никак не участвует в работе схемы. Если же его переключить в режим «Однократно», то в момент достижения нужной температуры, вместе с отключением основного реле К1, реле К3, ранее включенное через диод VD1 и резистор R7 в ту же коллекторную цепь выходного транзистора микросхемы, также отключается, контакты его размыкаются, и вывод 4 компаратора оказывается подключенным через датчик температуры к потенциалу «земли».

Такое состояние схемы устойчиво, и для возобновления работы в режиме стабилизации температуры необходимо либо на некоторое время отключить напряжение питания, либо тумблером S2 переключить схему в режим «Термостат». Конденсатор С2 вместе с диодом VD1 служат для «правильного» запуска схемы при включении питания. Если тумблер S2 разомкнут, то контакты реле К3 должны замкнуться сразу после подачи напряжения питания, иначе компаратор не сработает даже при низкой температуре, и все реле так и останутся разомкнутыми. При подаче напряжения питания, как мы знаем, конденсатор представляет собой короткозамкнутый участок цепи, поэтому реле К3 на небольшое время, пока конденсатор заряжается (примерно 100 мс), замкнет контакты. Диод VD1 на это время запирается и предохраняет от срабатывания реле К1 и К2. В случае, если температура воды в момент включения превышает установленную, такое срабатывание реле будет кратковременным — только на время зарядки конденсатора С2. Если же температура ниже требуемой, то компаратор успеет сработать, диод VD1 откроется, и все реле останутся в замкнутом состоянии до момента отключения нагрузки. Кстати, опыт эксплуатации подобного устройства показал, что наиболее популярен именно однократный режим (режим электрочайника), т. к. он позволяет экономить элек-

троэнергию и не беспокоиться о том, что вы оставили включенный электроприбор без присмотра.

Ручной режим (резервный, на случай выхода автоматики из строя, чтобы не остаться вовсе без горячей воды) обеспечивается просто: тумблер S1 в положении «Постоянный» подает сетевое питание напрямую на нагреватель (контакты K1 при этом шунтируются, схема обесточивается, а вся система работает так, будто никакой автоматики и не существует). В положении «Автомат» сетевое напряжение переключается на блок питания автоматики, а нагреватель теперь может включаться только контактами реле. Тумблер S1, естественно, должен выдерживать рабочий ток ТЭНа. Здесь подойдет импортный переключатель B1011, рассчитанный на ток до 16 А при напряжении 250 В, или другой аналогичный. В крайнем случае можно использовать автомобильные переключатели от «Жигулей», которые выдерживают большие токи, но это не очень корректно, т. к. на напряжения до 300 В они не рассчитаны.

Когда сетевое напряжение поступает на нагрузку (неважно, через тумблер или контакты реле), горит включенная параллельно ей неоновая лампочка H1, по которой можно контролировать работу схемы. Лампочка может быть любого типа, только не забудьте, что резистор R8 должен иметь мощность не менее 0,5 Вт, т. к. работает при сетевом напряжении. Использованное симисторное реле PF240D25 (разводка его выводов на схеме не показана, все нарисовано прямо на его корпусе), вообще-то допускает ток до 25 А, однако без принудительного охлаждения достаточно сильно греется уже при 10 А. Поэтому возможную мощность ТЭНа лучше ограничить величиной 2 кВт, а в корпусе устройства сверху и снизу обязательно должны наличествовать вентиляционные отверстия. Неплохо также, если реле K1 в рабочем положении корпуса будет расположено выше остальных деталей.

Если вы хотите добиться большей мощности, то лучше использовать аналогичное реле типа D2425 с возможностью установки на дополнительный радиатор (не ставить же, в самом деле, вентилятор, как рекомендуют производители PF240D25). Использовать при таких нагрузках электромагнитное реле вместо оптоэлектронного довольно затруднительно — придется включать мощный пускатель через промежуточное реле, и он отнюдь не будет услаждать ваш слух своим грохотом и жужжанием. А вот реле K2 и K3 вполне можно заменить маломощными электромеханическими, например, типа РЭС-60 или РЭС-49. Естественно, резисторы R6 и R7 в этом случае не требуются, а вот у конденсатора C2, возможно, придется раза в два увеличить емкость для более надежного включения устройства.

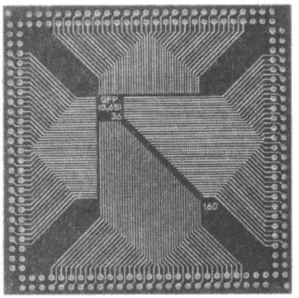
В положении тумблера S1 «Автомат» сетевое напряжение поступает на простейший блок питания, сделанный по схеме на рис. 9.10. Как обычно, его можно извлечь из покупного блока со встроенной вилкой — мощности от него никакой не требуется (вся схема потребляет ток порядка 30 мА), поэтому можно выбирать любой на напряжение 10–15 В. Напряжение с него поступает на стабилизатор типа LM78L09 (в корпусе TO-92, его можно заменить отечественным 142ЕН8Б), откуда стабилизированное напряжение +9 В подается на питание схемы. Светодиод VD2 сигнализирует о том, что автоматика включена, — его лучше выбирать зеленого свечения, чтобы обеспечить контраст с неоновой лампочкой.

При указанных на схеме номиналах термостат обеспечивает установку температуры в диапазоне примерно 35–85 градусов. Настройка его и калибровка ничем не отличаются от таковых для аквариумного термостата, кроме диапазона температур. В процессе настройки основную нагрузку можно не подсоединять, т. к. момент срабатывания и отключения вполне можно контролировать по неоновой лампочке. Только следует учесть, что вовсе без нагрузки «неонка» может гореть даже при выключенном реле из-за токов утечки, и вам даже может показаться, что система не работает. Тогда придется все же подключить какую-то нагрузку — в качестве нее удобно взять лампочку накаливания или даже просто двухваттный резистор сопротивлением около 20 кОм.

С перемешиванием/теплоизоляцией тут ситуация обратная по сравнению с аквариумом — принудительное перемешивание здесь обеспечить довольно сложно, но оно как раз и не очень требуется: во-первых, требования к точности поддержания температуры невысоки, во-вторых, нагреватель настолько мощный, что вода сама неплохо перемешивается за счет конвекции (естественной циркуляции нагретых водных масс). А вот теплоизолировать бак для воды я настоятельно рекомендую — просто обернув его старым ватным одеялом, вы можете экономить до 70% электроэнергии, причем это касается не только описываемой конструкции, но и вообще всех водонагревателей. Можно сделать и «фирменную» теплоизоляцию из упаковочного пенопласта или пенополиэтиленового коврика.

В заключение отметим, что схемы для построения термостатов невысокого класса, подобных описанным, существуют, разумеется, и в интегральном исполнении. Обычно они при этом совмещены с полупроводниковым датчиком температуры, который часто имеет и отдельный выход, что обеспечивает возможность показа температуры. С такими устройствами все знакомы, например, по встроенным в компьютерные материнские платы системам контроля температуры процессора и регулирования оборотов вентилятора.

О цифровых методах регулирования температуры мы немного поговорим в конце книги, а пока краткий курс теплотехники будем считать законченным и перейдем к другой теме — измерению этой самой температуры.



ГЛАВА 13

Как измерить температуру?
Электронные термометры

— Господи, до чего же мне холодно! — вскричал Планше, как только господин его скрылся из виду.

И, торопясь согреться, он немедленно постучался у дверей одного домика.

А. Дюма. «Три мушкетера»

Измерение температуры — одно из самых естественных и востребованных приложений электроники. Обычные спиртовые, а теперь и электронные термометры давно стали привычным атрибутом домашнего или офисного интерьера. Конструирование их — одно из самых благодарных направлений в любительском конструировании электронных приборов. Измерение температуры основано на простых и понятных любому школьнику физических явлениях, и имеет множество вариантов, причем почти все они вполне доступны для повторения в домашних условиях.

> ***Заметки на полях***
>
> На примере измерения температуры мы также познакомимся с общими принципами проведения измерений. Опыт, к сожалению, показывает полную дремучесть в этих вопросах не только большинства радиолюбителей, но даже и производителей вполне приличных в остальном модулей для любительского конструирования, полностью лишенных каких-либо намеков на метрологическую поддержку и на учет факторов, влияющих на точность измерений. Это же самое относится, кстати, и к бытовым метеостанциям, которые вместо истинной температуры и влажности запросто могут показывать «погоду на Марсе». Так что самостоятельное конструирование электронных измерителей еще и имеет практический смысл — не покупать же дорогущие профессиональные метеодатчики для такой тривиальной цели?

В этой главе мы рассмотрим основные принципы измерения температуры и конструкции некоторых простых электронных термометров. Но эти конструкции здесь приведены скорее для иллюстрации общего принципа: куда проще использовать микроконтроллеры, идеально приспособленные для подобных целей. Поэтому мы еще раз подробно вернемся к этой теме в конце книги, когда познакомимся с платформой Arduino.

Прежде чем познакомиться с методами измерения температуры, неплохо бы попытаться понять, что это такое — *температура*? Вопрос не совсем дурацкий, как это

может показаться на первый взгляд, потому что понятие температуры лежит в одном ряду с такими физическими абстракциями, как время, энтропия или электромагнитное поле. В отличие от последних двух, температуру мы можем ощущать физически, подобно расстоянию или массе, но на самом деле ясности в понимание сути дела это не добавляет. Так, течение времени мы тоже ощущаем, но на вопрос «что такое время?» сможет внятно ответить далеко не каждый — если вообще кто-нибудь знает ответ. И время, и температуру в смысле их измерения постигла похожая судьба — научились это делать с достаточной точностью в исторических масштабах совсем недавно.

Основы термометрии

Определение гласит: *температура есть мера внутренней энергии тела.* Мельчайшие частицы (атомы и молекулы), составляющие физические тела, все время движутся либо по некоторым траекториям в пространстве (в жидкостях и газах), либо колеблются около своего положения (в твердых телах). Чем интенсивнее они движутся, тем выше температура. Если в твердом теле она достигает некоторого критического значения, то атомы-молекулы срываются со своих мест, структура тела нарушается, и оно плавится, превращаясь в жидкость. Если повышать температуру дальше, то связи между частицами уже не могут победить возросшую интенсивность их движения, и жидкость начинает испаряться, превращаясь в газ. При высокой температуре нарушаются уже связи внутри молекул и образуется так называемая *холодная плазма* (например, пламя), при очень высокой — и внутри атомов, и вещество превращается в высокотемпературную плазму.

В реальности на эту упрощенную модель накладываются некоторые нюансы. Скажем, вещество может существовать при одних и тех же условиях в нескольких состояниях, например, как твердое тело в равновесии с жидкой и газообразной фазами — это так называемая *тройная точка.* Но нам сейчас важнее другое — из нарисованной картины следует, что должно быть такое состояние вещества, когда движения нет, все частицы стоят на месте и, следовательно, внутренняя энергия равна нулю. Это состояние существует и носит название *абсолютного нуля температуры.* Чему она равна при этом, вычислил теоретически еще в середине позапрошлого века ученый-физик лорд Кельвин. Оказалось, что абсолютный ноль, он же ноль абсолютной температурной шкалы (шкалы Кельвина), отстоит от точки замерзания воды на −273,15 °C. При этом градусы в шкале Кельвина (°K) равны градусам в привычной шкале Цельсия (°C), где за ноль принята точка замерзания воды. Так что перевод очень прост — чтобы получить температуру в градусах Цельсия, надо из градусов Кельвина вычесть величину 273. Чтобы подчеркнуть разницу между °K и °C, первые часто обозначают большой буквой T, а вторые — маленькой t. В англоязычных странах в быту традиционно используют шкалу Фаренгейта (обозначается заглавной F), в которой и ноль другой, и градусы меньше, поэтому пересчет относительно сложен:

$$°C = \frac{5}{9}(°F - 32).$$

Подробности

Так как на практике измерить внутреннюю энергию саму по себе невозможно, температуру измеряют по каким-то ее внешним проявлениям. Логично для этого использовать точки фазового перехода (плавления и кипения) химически чистых веществ. Эти точки стабильны и хорошо воспроизводятся. В настоящее время принята международная практическая температурная шкала, уточненная последний раз в 1990 году (МПТШ-90), в которой около двух десятков таких реперных (опорных) точек, охватывающих диапазон от −259,34 °С (тройная точка водорода) до 1084,62 °С (точка плавления меди). Точки замерзания и кипения воды, которые часто применяются для калибровки термометров на практике, ранее также относились к основным реперным точкам, но в МПТШ-90 они вошли с оговорками[1]. Между опорными точками температуру в этой шкале определяют платиновым термометром, имеющим сопротивление ровно 100 или 10 Ом при температуре 0 °С. Сопротивление платины при повышении температуры возрастает с наклоном 0,39250 %/°С, и, хотя зависимость эта не очень линейна, она весьма хорошо воспроизводится. По методике МПТШ изготавливают эталоны температуры: национальные, первичные, вторичные и т. д. Средства измерения, сертифицированные путем непосредственного сравнения с эталоном, называют *образцовыми*.

Все пользовательские измерительные инструменты (и не только температуры), поступающие на прилавок, на каком-то этапе сравнивались с образцовыми средствами. Сравнение вновь изготовленного измерителя с каким-либо средством измерения, которое мы принимаем за образцовое, называется градуировкой или калибровкой. Строго говоря, это одно и то же, однако под *градуировкой* чаще понимают создание градуировочной таблицы или формулы, по которой показания прибора пересчитываются в соответствующую физическую величину, а под *калибровкой* — подстройку самого прибора так, чтобы он непосредственно показывал эту физическую величину. С появлением компьютерных технологий разница между градуировкой и калибровкой практически исчезла. Процедура проверки уже готового средства измерения на соответствие образцовому средству измерения называется *поверкой*. С этим когда-то весьма специфическим термином в последние годы волей-неволей познакомились все российские домашние хозяйки — после того, как им пришлось установить домашние счетчики воды.

Датчики

На практике для измерения температуры электронными методами используют в основном две разновидности датчиков: металлические термометры сопротивления и полупроводниковые датчики. Термисторы (терморезисторы) для измерения температуры применяют редко, лишь в некоторых специфических случаях, т. к. их единственное достоинство в этом плане — высокая чувствительность — не перевешивает многочисленные недостатки, среди которых в первую очередь нелинейность и, кроме того, невысокая стабильность. Правда, существуют специальные высокостабильные миниатюрные алмазные термисторы (выполненные на основе монокристаллов искусственного алмаза), которые могут работать при температурах до 600 °С, но их температурный коэффициент всего раза в полтора выше, чем у металлов, и они используются также в специфических случаях — например, в печках

[1] Множество подробностей о различных средствах и методиках измерения температуры, включая и сведения об устройстве международной шкалы температуры, можно найти на сайте **Temperatures.ru**.

лазерных принтеров. Термисторы чаще применяют в схемах регуляторов температуры (см. *главы 12* и *20*), где их нелинейность не имеет значения.

Еще один способ очень точного измерения температуры предполагает использование специальных термочувствительных кварцевых резонаторов. О них мы еще будем говорить в *главе 16*, а здесь остановимся лишь на металлических и полупроводниковых датчиках, добавив вначале несколько слов про термисторы.

Термисторы

Для успешного применения термисторов стоит знать их основные свойства. Большинство так называемых *NTC-терморезисторов* (от английского **N**egative **T**emperature **C**oefficient) имеют падающую экспоненциальную зависимость сопротивления от температуры, которая с хорошей точностью описывается уравнением:

$$R_{T2}=R_{T1}\, e^{B(1/T2 - 1/T1)} \qquad (1)$$

Здесь: R_{T1} — номинальное сопротивление при температуре T1 (обычно при 25 °C), B — коэффициент, имеющий размерность °K, который приводится в характеристиках термистора для некоторого диапазона температур, например, для 25–100 °C. При отсутствии фирменного технического описания величину B несложно вычислить, исходя из двух измеренных значений R_T, а для ориентировочных расчетов его можно принять равным в пределах 3500–4500.

График, соответствующий уравнению (1), построенный по данным для конкретного термистора B57164-K 103-J с номинальным сопротивлением 10 кОм при 25 °C, приведен на рис. 13.1, а числовые значения, по которым он построен, сведены в табл. 13.1. Из графика мы видим, что крутизна характеристики термистора с повышением температуры снижается (ее значения приведены в третьей колонке таблицы). Эта нелинейность делает термисторы крайне неудобным средством для измерения температур, зато высокая величина крутизны (в среднем раз в десять большая, чем у металлов) очень удобна при использовании их в качестве датчика для регуляторов температуры. Температурный диапазон применения NTC-тер-

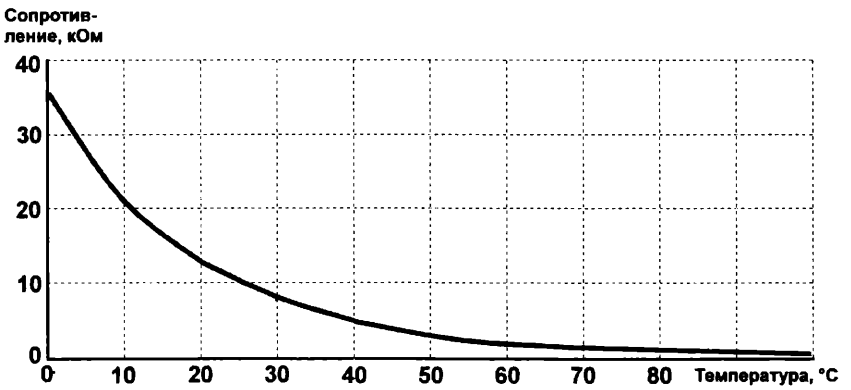


Рис. 13.1. Температурная характеристика NTC-термистора

Таблица 13.1. Теоретические значения сопротивления термистора B57164-K 103-J при разных температурах, кОм ($B_{25/100}$ = 4300, из документации фирмы EPCOS AG)

Температура, °C	Сопротивление, кОм	Крутизна характеристики, %/°K
0,0	35,563	5,5
5,0	27,119	5,3
10,0	20,86	5,1
15,0	16,204	5,0
20,0	12,683	4,8
25,0	10,000	4,7
30,0	7,942	4,6
35,0	6,3268	4,5
40,0	5,074	4,3
45,0	4,1026	4,2
50,0	3,3363	4,1
55,0	2,7243	4,0
60,0	2,237	3,9
65,0	1,8459	3,8
70,0	1,5305	3,7
75,0	1,2755	3,6
80,0	1,0677	3,5
85,0	0,89928	3,4
90,0	0,76068	3,3
95,0	0,64524	3,3
100,0	0,54941	3,2

мисторов ограничен пределами работоспособности полупроводниковых материалов (т. е. диапазоном от –55 до 125 °C).

Еще одно свойство NTC-термисторов надо всегда иметь в виду при их практическом применении — из-за отрицательного температурного коэффициента включение термистора в цепь питания напрямую, без резистора, ограничивающего ток, может спровоцировать лавинообразное возникновение эффекта положительной обратной связи. Нагрев термистора приводит к падению его сопротивления, отчего ток через него увеличивается, в свою очередь увеличивая нагрев еще больше, и если ток не ограничен, то термистор в конце концов попросту расплавится. Потому напрямую к источнику питания термисторы подключать не рекомендуется, а предельная выделяющаяся мощность для обычных «таблеточных» конструкций должна быть ограничена на уровне нескольких десятков, максимум сотен милливатт.

Металлические датчики

Фирменные термометры сопротивления представляют собой обычный резистор из металлической — медной или платиновой[1] — проволоки. *Платиновые датчики* (ТСП, термометр сопротивления платиновый) наиболее стабильны и применяются для высокоточных измерений, но они обладают заметной нелинейностью, поэтому значения температуры приходится рассчитывать по таблицам (см., например, [2]). Использование меди более практично — у нее зависимость сопротивления от температуры наиболее близка к линейной в широком диапазоне температур. В диапазоне от −50 до +100 °С погрешность за счет нелинейности в пересчете на температуру не превысит 0,1 °С.

Сопротивление датчиков промышленного изготовления точно подогнано под стандартные 10, 50 или 100 Ом. Платиновые датчики используют в диапазоне от −260 до +1100 °С, а медные (ТСМ) — от −200 до +200 °С. Доступность меди приводит к искушению изготовить такой датчик самому, и в большинстве случаев это совершенно не возбраняется, хотя прецизионный термометр на самодельном датчике, конечно, не получится (это тот случай, когда структура металла имеет значение, — в отличие от аудиокабелей, см. *главу 8*).

Подробности

Внимательный читатель, несомненно, задаст вопрос: а почему металлические датчики температуры имеют такое небольшое сопротивление — до 100 Ом? Почему не бывает промышленных металлических термометров сопротивления на 1 или 100 кОм? Казалось бы, чем выше сопротивление, тем меньший ток требуется, и тем проще улавливать его изменения, правда? И тем не менее прецизионных металлических датчиков с высоким сопротивлением не существует по одной фундаментальной причине: любой проводник имеет собственный электрический шум за счет флуктуаций «электронного газа» в кристаллической решетке. И чем выше сопротивление проводника, тем больше амплитуда этого шума. То, что это вполне измеряемый физический эффект, подтверждается тем фактом, что когда-то «отец компьютерного века» математик Алан Тьюринг, отчаявшись чисто математическим путем решить проблему генерации истинно случайных чисел, предложил использовать для этой цели как раз собственные шумы резисторов.

Полупроводниковые датчики

Полупроводниковые датчики удобно использовать во всех случаях, когда не требуется профессионально высокая точность. В том числе они вполне годятся и для бытовых целей. Простейший полупроводниковый датчик температуры — это обычный кремниевый диод или транзистор в диодном включении (когда коллектор соединен с базой). Пресловутое прямое падение напряжения на диоде, равное 0,6 В, имеет почти линейный отрицательный температурный коэффициент, равный приблизительно 2,3 мВ/°С. Все промышленные полупроводниковые датчики тем или иным способом используют этот эффект.

---

[1] Пусть вас слово «платиновый» не пугает — самой платины там так мало, что стоит такой датчик не дороже медного, тем более что в датчиках используют медь высокой степени очистки, которая ненамного дешевле драгметаллов.

Фирменные полупроводниковые датчики делятся на две разновидности: с аналоговым и цифровым выходом. Аналоговые датчики (DS60, MAX6605) имеют обычно три вывода (питание, общий и выход), а цифровые иногда всего два (DS1721), питаясь от сигналов запроса, поступающих с внешнего контроллера (см. *главу 11*). К цифровым датчикам мы вернемся в конце книги, а пока следует особо отметить довольно точные и удобные в различных применениях аналоговые датчики TMP35, TMP36 и TMP37 фирмы Analog Devices (аналоги: LM135, LM 235 и LM 335 фирмы ST Microelectronics или 1019EM1 отечественного исполнения), которые включаются подобно диоду, но несут третий вывод для подстройки температурного коэффициента, имеющего величину аж 10 мВ/°С, причем с положительным наклоном.

Полупроводниковым датчикам, как правило, свойственны погрешности заводской установки порядка 1–2 °С, и иногда встречающееся в характеристиках определение «прецизионный», видимо, относится к повышенной их стабильности — после соответствующей калибровки погрешности снижаются до порядка долей градуса. Впрочем, как показал опыт, специальные цифровые датчики со встроенным микроконтроллером, позволяющим выдавать «наружу» непосредственно физическую величину в градусах, довольно точны, и часто дополнительной калибровки не требуют (см. *главу 21*). Так что вопрос только в их правильном размещении, по возможности исключающем влияющие факторы, что обычно и составляет проблему как любительских конструкций, так и многих фирменных электронных термометров.

Средства калибровки

В домашней практике для поверки разрабатываемых самостоятельно приборов лучше всего использовать ртутный лабораторный термометр с делениями не крупнее одной-двух десятых градуса (погрешность таких термометров, однако, может быть выше и составлять 0,2 и даже 0,5 °С). Подобные термометры несложно приобрести в интернет-магазинах, торгующих лабораторным оборудованием. Для основного диапазона нужен термометр от 0 до 50 °С, и может потребоваться еще один для диапазона до 100 °С, а также отдельный в отрицательной области. Выбирать один-единственный на полный диапазон где-нибудь от минус 50 до плюс 100 градусов не следует, потому что у них цена деления существенно меньше, а погрешность выше. За неимением таковых, конечно, можно обойтись и бытовыми спиртовыми или цифровыми термометрами (последние должны иметь выносной датчик), только не следует забывать про их достаточно высокую погрешность, которая может составлять 1–2 °С.

Категорически не рекомендуется применять для калибровки бытовые металлические термометры расширения (с такой спиралькой, соединенной со стрелочкой), всем знакомые по бытовым газовым или электрическим духовкам, — они могут ошибаться на десятки градусов. Если требуется калибровка при повышенных температурах, то лучше использовать термометры на основе термопары, которыми комплектуются некоторые мультиметры. Правда, последние, наоборот, не годятся для обычного диапазона температур, по причине, которую мы рассмотрим далее.

Методы измерения сопротивления

Рассмотрим методы, с помощью которых можно измерять сопротивление металлических датчиков с точностью, достаточной для пересчета его в температуру. Обычный мультиметр тут не подойдет — рядовой прибор измеряет сопротивление с по-

грешностью порядка 1% от всей шкалы. Поэтому, измеряя таким прибором, скажем, сопротивление медного датчика 100 Ом (с крутизной менее 0,4 Ом/°С) на пределе 200 Ом, мы получим погрешность в пересчете на температуру градусов в пять, что неприемлемо даже для самых непритязательных радиолюбителей. Именно по этой причине термометры на основе мультиметров не годятся в качестве средств калибровки в малом диапазоне температур — одно дело измерить с погрешностью в пять градусов 200-градусную температуру жала паяльника и совсем другое — комнатные 20 градусов в воздухе.

Основная идея проведения измерений сопротивления металлических датчиков с приемлемой точностью известна еще со времен английского физика Ч. Уитстона (1802–1875), чьим именем и названа показанная на рис. 13.2 конструкция из четырех сопротивлений. Такой *мостик Уитстона*, как мы увидим, в той или иной форме используется на практике и по сей день. Уитстон прославился еще своими работами в области телеграфии и рядом других достижений, но приведенная схема, без сомнения, самое выдающееся его изобретение.

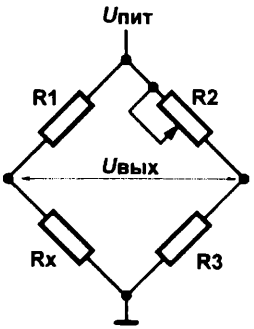


Рис. 13.2. Мостик Уитстона

Для измерения величины сопротивления Rx положение движка переменного сопротивления R2 устанавливается так, чтобы напряжение в выходной диагонали моста ($U_{вых}$) было равно нулю. Если в этот момент измерить установленное значение R2 (можно заранее проградуировать его ползунок в единицах сопротивления), то неизвестное сопротивление, учитывая, что значения сопротивлений резисторов R1 и R3 нам известны, определяется по формуле:

$$Rx = R1\frac{R3}{R2}.$$

Участвующие в схеме резисторы называются *плечами моста*. Можно также объединить R2 и R3 в один переменный резистор, включенный по схеме потенциометра ($U_{вых}$ тогда снимается с его движка, а за плечи R2 и R3 принимаются его части между движком и выводами).

Мостовой способ имеет ряд преимуществ. Во-первых, работа этой схемы в принципе не зависит от напряжения питания, потому что баланс определяется не абсолютными значениями падений напряжения на резисторах, а их соотношением. На

практике некоторая зависимость будет иметь место (т. к. чувствительность схемы со снижением питания падает), но, тем не менее, в довольно широких пределах это положение соблюдается.

Во-вторых, обеспечить фиксацию момента равенства напряжения в диагонали моста нулю (при этом условии мост называется *сбалансированным*) несравненно проще, чем измерить с достаточной точностью абсолютное значение напряжения или сопротивления. Для того чтобы настроить очень точно ноль вольтметра любого класса точности, никакого специального оборудования не надо — достаточно замкнуть накоротко его входные клеммы. От вольтметра при этом требуется только одно — как можно более высокая чувствительность, потому мостовые методы отлично работали уже в XIX веке, когда никаких прецизионных приборов еще не существовало.

Так что точность зависит только от сопротивлений. Постоянные резисторы можно подобрать очень точно (на практике используют катушки из манганиновой калиброванной проволоки или готовые сопротивления класса 0,05). В качестве резистора R2 обычно используют магазины сопротивлений, которые представляют собой по сути дела переменный резистор, составленный из множества постоянных, которые могут коммутироваться с помощью набора десятипозиционных переключателей, называемых *декадными*. Причем все устроено таким образом, что каждый переключатель связан с сопротивлениями в десять раз меньшего или большего номинала, чем соседний.

Очень точный ручной измеритель температуры

Принципиальная схема для ручного измерения сопротивления образцового датчика температуры сопротивлением 100 Ом (платинового или медного) с использованием описанных в предыдущем разделе средств приведена на рис. 13.3. Магазин сопротивлений на ней условно показан в виде переменного резистора Rм. Все резисторы, кроме, конечно, измеряемого сопротивления Rt и магазина Rм (а также, возможно, R1, который лучше подобрать из проволочных) — типа С2-29В. После ручной балансировки моста с помощью магазина сопротивлений Rм (вольтметр на выходе должен показать ноль) измеряемое сопротивление Rt будет определяться по формуле:

$$Rt = R1\frac{Rx}{(Rм - Rx)},$$

где Rx есть величина нижней по схеме части сопротивления магазина. Сравнивая Rt с табличным значением [5], можно узнать измеряемую температуру.

Подробности

Инструментальный усилитель на микросхеме DA1 здесь нужен для обеспечения достаточной чувствительности схемы. Его коэффициент усиления выбирается из следующих соображений: допустим, наш мультиметр имеет на самом маленьком пределе измерения напряжений (200 мВ) чувствительность один знак после запятой, т. е.

0,1 мВ (обычная разрешающая способность рядовых мультиметров). При коротком замыкании его щупов на шкале должны показываться все нули (ноль не сдвинут и не «гуляет»). Некоторую погрешность при измерениях будет вносить всегда наличествующая помеха, поэтому возьмем запас и примем чувствительность его равной 1 мВ. Ток через датчик при выбранных номиналах сопротивлений и напряжении питания будет составлять приблизительно 4,5 мА. Для того чтобы обеспечить необходимую разрешающую способность измерения температуры приборами, которые мы будем конструировать (для большинства применений необходимая и достаточная величина ее составляет 0,1 °С), нам надо обеспечить разрешающую способность нашего образцового термометра не менее, чем в два раза более высокую (т. е. 0,05 °С — бóльшая точность не имеет смысла, см. далее). Зададимся на всякий случай еще меньшей величиной — 0,03 °С. Датчик имеет сопротивление 100 Ом, поэтому при крутизне его характеристики, равной примерно 0,4 %/°С (величина справедлива и для платины, и для меди), изменение сопротивления будет численно равно этой величине — 0,4 Ом/°С. При указанном токе через измерительное плечо моста изменение напряжения на диагонали моста составит 1,8 мВ/°С, т. е. при изменении температуры на 0,03 градуса изменение напряжения составит 0,054 мВ. Нам желательно увеличить это напряжение разбаланса до установленного значения чувствительности мультиметра в 1 мВ, отсюда коэффициент усиления инструментального усилителя должен составить примерно 20.

Диапазон значений измеряемой температуры для этого устройства практически ограничен только возможностями датчика. Подробно погрешности нашей схемы

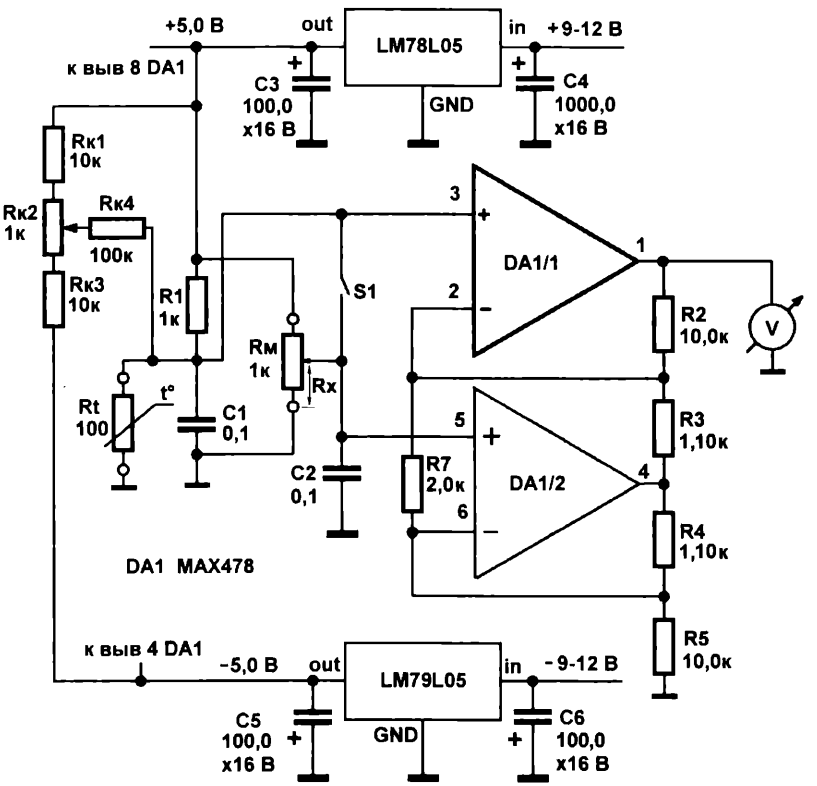


Рис. 13.3. Принципиальная схема измерителя сопротивления образцового датчика температуры

мы анализировать не станем, укажем только, что с точки зрения точности схема обладает одним недостатком — в ней нескомпенсировано влияние соединительных проводов. Как такая компенсация выполняется, мы узнаем из *главы 17*. А здесь просто примем, что провода, соединяющие со схемой как датчик, так и магазин сопротивлений, должны быть минимально возможной длины и достаточно большой толщины — сечением не менее 2 мм. Эта схема критична также, кроме точности резистора R1, к выбору ОУ, и при замене следует применять только ОУ с точностными характеристиками не хуже указанных, а также обратить внимание на возможность их работы при напряжении питания ±5 В (см. *главу 12*).

Заметки на полях

Схему можно украсить, если на выход усилителя параллельно вольтметру подсоединить двухцветный двухвыводной светодиод (с токоограничивающим резистором порядка 300–510 Ом). Когда мост находится в разбалансе, светодиод будет гореть, причем цвет свечения будет зависеть от знака разбаланса, а яркость — от его степени. Когда на выходе установится ноль, светодиод погаснет. Разумеется, более-менее точно проконтролировать ноль можно все равно только по вольтметру, но это удобно при значительном уходе температуры — сразу видно, в какую сторону она ушла.

Схема на рис. 13.3 приведена скорее в иллюстративных целях, чтобы понять, как в принципе устроены измерители температуры. Можно ли автоматизировать работу такой схемы? Естественно можно, но на практике осуществить это весьма и весьма непросто — схемотехническое решение должно быть очень тщательно продумано. Теперь вы можете оценить, почему прецизионное оборудование стоит так дорого.

Простейшие электронные термометры на батарейке

Как ни странно, но такое распространенное устройство, как бытовой термометр, требует достаточно высокой точности — не хуже 0,1–0,2 °С, хотя бы по той причине, что не очень красиво, когда созданный вами прибор показывает +1 градус, в то время как лужи вокруг стойко покрылись льдом. Для обычного диапазона уличных термометров от −50 до +50 °С такая точность эквивалентна относительной погрешности в 0,1%, что достаточно низкая величина для того, чтобы отнестись к ней со всем возможным уважением, — сравните с погрешностью не самых дешевых серийных мультиметров, лежащей в лучшем случае в пределах 0,5%.

Легальный путь замять проблему — не демонстрировать десятые градуса, как это делают на уличных табло, тогда допустимая погрешность повышается по крайней мере до 0,5%. Однако мое убеждение состоит в том, что демонстрировать температуру без десятых градуса все равно, что делать наручные часы без секундной стрелки, — вроде бы «по жизни» и не слишком требуется, но как-то... несолидно. Первое наше детское представление о температуре заключается в магическом числе «36,6», и три цифры эти навсегда переплетаются с самим понятием.

Но мы пока не знаем, как делать точные аналого-цифровые схемы, и окончательно освоимся в этой области только в *главах 17* и *21*. Поэтому здесь мы рассмотрим

простейшую реализацию электронного измерителя температуры, не обращая особого внимания на погрешности. Конструкция имеет свою изюминку, которая компенсирует факт ее не слишком высокой точности, — она малопотребляющая и будет работать от одной 9-вольтовой батарейки типа «Крона». В *главах 20* и *21* вы узнаете, как просто реализовать подобные термометры на цифровой платформе Arduino, а пока ради лучшего усвоения основ электроники остановимся на чисто аналоговых методах.

Электронный термометр со стрелочным индикатором

Схема со стрелочным индикатором приведена на рис. 13.4. В качестве показывающего устройства здесь используется измерительная головка типа М903 с током полного отклонения 50 мкА. Можно использовать любую другую головку магнитоэлектрической системы, но если ток полного отклонения отличается от указанной величины, то придется пересчитать резистор R7. Головку придется доработать: с нее надо снять переднюю крышку со стеклом и очень аккуратно, чтобы не повредить весьма чувствительную стрелку с очень нежным поворотным механизмом, наклеить поверх имеющейся шкалы новую. Шкалу эту можно изготовить, напечатав ее на плотной бумаге с помощью струйного или лазерного принтера.

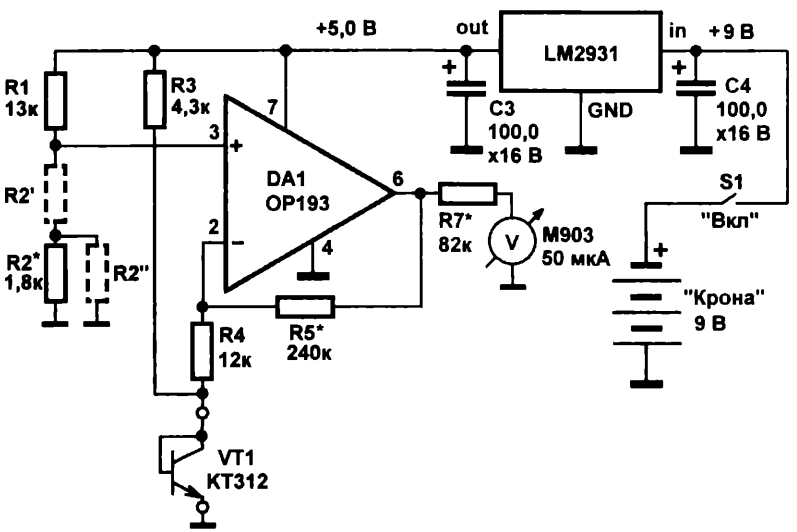


Рис. 13.4. Электронный термометр со стрелочным индикатором

Крайние деления на шкале должны совпадать с делениями на оригинальной шкале (положение ограничителей хода стрелки не должно совпадать с крайними делениями — у стрелки должен оставаться небольшой свободный ход за пределы шкалы). Крайнее левое деление будет соответствовать −50°, а крайнее правое +50°, ноль в этом случае должен располагаться ровно по центру шкалы. Так как длина шкалы равна всего нескольким сантиметрам, то нанести разборчивые деления с шагом меньше, чем через 2 градуса, вряд ли получится, и именно этот параметр будет

определять максимальную требующуюся точность — снижать погрешность ниже половины деления шкалы, т. е. в рассматриваемом случае менее 1°, не имеет смысла. Заметим, что нет никаких проблем в том, чтобы отградуировать шкалу на любой другой диапазон, скажем, от −30° до 70° или от 0° до 100°, — для этого нужно будет только подобрать величину резистора R2.

Датчиком температуры здесь служит транзистор в диодном включении. Можно использовать любой маломощный кремниевый *n-p-n*-транзистор (за исключением «супербета»-разновидностей), единственное, что желательно (но необязательно), чтобы он был в металлическом корпусе. Для изготовления датчика подбирают подходящую по диаметру пластмассовую трубку и заливают в нее эпоксидной смолой транзистор с заранее подпаянными выводами так, чтобы его металлический корпус соприкасался с окружающей средой, — чувствительность и скорость реакции термометра сильно возрастут в сравнении с заделкой его внутрь трубки. Можно использовать и кремниевый диод, но заделывать его придется способом, показанным на рис. 12.9, и прогреваться он станет значительно медленнее.

Ток через датчик будет равен примерно 1 мА, а падение напряжения на нем, естественно, около 0,6 В. Наклон температурной характеристики отрицателен и равен примерно, как мы говорили, 2,3 мВ на один градус, поэтому общее изменение напряжения на датчике составит 230 мВ на диапазон 100 °С. Выходное напряжение ОУ при максимальном сигнале мы хотим сделать как можно больше, чтобы минимизировать как ошибки, связанные с собственным падением напряжения на измерительной головке, так и погрешности схемы вообще. Максимум, что мы можем получить от ОУ в такой схеме, — это напряжение несколько ниже напряжения питания, равного 5 В (именно из этого условия подбирается R7), поэтому выбираем коэффициент усиления, приблизительно равный 20 (с округлением в меньшую сторону).

От ОУ здесь не требуется особо высокой точности, зато существенны малое потребление, низкое питающее напряжение и «умение» работать с выходными напряжениями, равными напряжению «земли». Кроме упомянутого OP193, подойдут OP196, MAX406, MAX409 (они даже совпадают по цоколевке) и многие другие типы.

Общее потребление схемы определяется здесь в основном потреблением цепи датчика, равным приблизительно 1 мА. Потребление стабилизатора, ОУ и делителя R1–R2 добавят еще примерно 0,5 мА, и суммарное потребление составит около 1,5 мА. Емкость щелочной батарейки «Крона» составляет порядка 600 мА·ч, и наша схема сможет проработать от одного элемента в непрерывном режиме около 17–20 суток. Отметим, что если вместо микропотребляющего стабилизатора LM2931 поставить обычный 78L05, то время работы резко уменьшится.

При отладке вместо резисторов R2 и R5 сначала устанавливаются подстроечные резисторы соответствующего номинала (R5 — несколько больше указанного на схеме). Настройку схемы надо начинать с того, что погрузить датчик в среду с температурой 0 °С (тающий снег или мелкоизмельченный лед в равновесии с водой — лучше всего поместить эту смесь в термос и в процессе работы периодически перемешивать) и установить с помощью резистора R2 стрелку головки на 0°. После это-

го датчик переносится в среду с температурой 40–50° (вот тут пригодится термостат!) и путем изменения R5 устанавливаются соответствующие показания стрелки. Ноль градусов у нас тоже при этом «уйдет», потому указанную процедуру следует повторить несколько раз, перенося датчик из среды с температурой 0° в среду с более высокой температурой и обратно.

Точность калибровки будет тем выше, чем больше разница между температурами в калибровочных точках, однако одну из точек обязательно надо выбирать равной или близкой к нулю градусов, потому что это критичное для практики значение. После этого переменные резисторы выпаивают и помещают на их место постоянные резисторы с точно такими же номиналами, при необходимости составляя их из нескольких параллельно и/или последовательно включенных. Особую точность при этом надо соблюдать при подборе R2 (ноль градусов). На плате лучше заранее предусмотреть места для подключения параллельных и последовательных резисторов (показаны на схеме пунктиром для R2, аналогично следует поступить и для R5). Резисторы можно использовать обычные, типа МЛТ, прецизионных резисторов типа С2-29В здесь не требуется.

В предыдущих изданиях книги приводилось также описание аналогичной конструкции с цифровым индикатором, но в настоящее время это потеряло всякий смысл: намного проще и дешевле спроектировать полностью цифровой термометр на микроконтроллере, чем возиться с подгонкой заведомо неточной аналоговой схемы под существующие цифровые модули. Вот о точностях измерительных схем мы сейчас и поговорим.

Немного о метрологии и ошибках аналоговых схем

Доступность цифровых измерений в современных реалиях породила явление массовой безграмотности в отношении таких сущностей, как *ошибки измерений*. В самом деле, уже не раз упоминавшаяся платформа Arduino (см. *главы 20–22*) для проведения аналоговых измерений фактически требует всего лишь одной строчки программного кода — вызова функции `analogRead()`. Это порождает мнимую уверенность в том, что все произойдет само по себе, и никаких знаний об погрешностях тут не требуется. Разумеется, все тут далеко не так, и этот раздел — лишь краткое введение в тему погрешностей электронных схем, изучение которой мы будем продолжать на протяжении всей книги.

Необходимость элементарных знаний в области метрологии для радиолюбителя можно пояснить на примере инструкции к мультиметру: пусть там записано, что погрешность измерения напряжения составляет 0,5% на пределе 2 вольта. Если вы сходу правильно ответите на вопрос, насколько в абсолютных единицах (вольтах или милливольтах) конкретная величина, показываемая прибором (например, «1,000 В») может отличаться от истинной, можете эту часть главы не читать (правильный ответ приведен в конце главы).

Другая типовая задача — построить градуировочную кривую и вычислить нужные коэффициенты пересчета для какого-либо датчика, чтобы прибор показывал физи-

ческие величины, — также трудноосуществима без элементарных знаний в области метрологии. Кроме того, пытаться проектировать измеритель любой физической величины, не проведя хотя бы поверхностного анализа возникающих погрешностей, совершенно бессмысленно — даже при самых мягких требованиях к точности можно основательно «попасть», зря потратив и время, и деньги. Попытаемся очень кратко систематизировать сведения, которые необходимы для такого анализа.

Метрология — наука о том, как правильно проводить измерения. Все началось с того, что возникшая в середине прошлого тысячелетия рациональная наука поставила во главу угла принцип поверки теории экспериментом. Ясно, что это возможно осуществить только в том случае, если эксперимент воспроизводим, т. е. может быть повторен любым другим человеком (это положение еще называют принципом «верификации»). Основная же проблема воспроизводимости состоит в том, что ни один эксперимент не обходится без ошибок. Поэтому метрология занимает очень важное место в современном мире. Без нее технический и научный прогресс был бы вообще невозможен, потому что никто бы тогда не смог ничего сказать о достоверности полученных в эксперименте данных.

Если мы представим себе экспериментальную систему наподобие объекта регулирования, изображенного на рис. 12.2, то кроме входов (входных воздействий), которые контролируются исследователем, на систему действует еще множество различных факторов, которые можно поделить на несколько различных групп. Так, есть незначимые факторы — те, которые нам известны, но для простоты мы их влиянием пренебрегаем, — такие, как отклонения в свойствах реальных физических тел от идеализаций типа «абсолютно твердое тело» или «материальная точка» (типичный пример — влияние базового тока в транзисторе на величину эмиттерного, которое мы обычно не учитываем). Есть факторы вполне значимые, но мы не можем ими управлять и часто даже неспособны их контролировать, — скажем, разброс параметров электронных компонентов. Как бы все упростилось, если бы все транзисторы одного типа были бы совершенно одинаковыми! Наконец, во многих случаях могут присутствовать и неизвестные нам факторы, — содержание науки во многом состоит в том, чтобы такие факторы обнаруживать и влияние их исследовать.

Как же можно учитывать подобные воздействия? Тут на помощь приходит теория вероятностей — точнее, ее дочерняя прикладная дисциплина под названием *математическая статистика*. Основное ее предположение состоит в том, что все неучтенные факторы можно рассматривать как равномерный шум, приводящий к чисто случайному разбросу значений измеряемой величины. Излишне говорить, что довольно часто это предположение не совсем соответствует действительности, но все же в большинстве практических случаев (по крайней мере, в технических приложениях) оно обеспечивает неплохое приближение к истине, и применение методов математической статистики дает на удивление хорошие результаты. Только не следует забывать, что статистика не может повысить точность измерения, если прибор этого не позволяет, — она всего лишь дает нам сведения о том, чего мы достигли в действительности.

Точность и разрешающая способность

Несколько слов о том, насколько вообще целесообразно стремиться к высокой абсолютной точности измерений. Измерительные схемы характеризуются тремя основными параметрами: *точностью*, *разрешающей способностью* и *стабильностью* (временным дрейфом). Что такое точность или обратная ей величина — *погрешность*, понятно интуитивно. Разрешающая же способность (иногда говорят о чувствительности) — это попросту минимальная разница в значениях измеряемого параметра, которую мы еще можем различить. Для аналоговых приборов (стрелочных, или, например, ртутных термометров) это половина самого мелкого деления шкалы, а для цифровых — единица самого младшего разряда. Естественно, повышать точность сверх разрешающей способности бессмысленно. А стабильность (дрейф) — самый сложный для оценки параметр, она характеризует уход показаний с течением времени. Подробнее на вопросах оценки дрейфа мы не станем здесь останавливаться.

Я вас могу удивить, но буду утверждать, что в большинстве практических случаев точное значение абсолютной величины — в определенных пределах, разумеется — не представляет особого интереса. При измерении температуры единственное исключение для бытовых приборов — точка замерзания воды, о чем мы говорили ранее. Но в других случаях обычно нам неважно, 9 градусов на улице или 11, главное — весна, и можно снимать шубу.

С другой стороны, обычно нет никакого смысла конструировать суперстабильные и высокоразрешающие, но неточные приборы — просто потому, что обеспечение стабильности и точности во многом взаимосвязаны, причем первое еще и существенно сложнее. А если мы очень сильно увеличим разрешающую способность по сравнению с точностью, то рискуем попасть в ситуацию, когда десятые градуса просто будут мельтешить на дисплее, что еще хуже, чем если бы их не было вовсе. Но не забывайте, что абсолютная точность, кроме всего прочего, зависит от тщательности градуировки и используемого эталона, а разрешающая способность и стабильность — только от компонентов и конструкции.

Заметки на полях

Точность и погрешность — величины взаимодополняющие, что совершенно ясно по смыслу терминов. Поэтому, вообще говоря, произнести что-то вроде «точность в пределах 1%» — некорректно, естественно, тут идет речь о погрешности, а точность в этом случае выражалась бы числом 99%. Тем не менее, в разговорной речи такое допустимо, и мы сами не раз прибегали к подобным оборотам, — просто потому, что совершенно ясно, о чем идет речь, и запутаться невозможно. А вот в англоязычных странах почему-то вместо погрешности принят термин именно «accuracy», что даже без обращения к словарю легко перевести как точность (вместо отвечающего по смыслу «inaccuracy»). Этот нюанс следует иметь в виду не только при чтении литературы на английском языке — в переводах на русский, например, англоязычных инструкций к измерительным приборам довольно часто можно встретить употребление слова «точность» вместо «погрешность».

Систематические ошибки

Ошибки измерения делятся на случайные (тот самый шум, о котором шла речь ранее) и систематические. Прояснить, что такое *систематическая ошибка*, можно на следующем примере: предположим, мы немного изменим в схеме, собранной по рис. 13.4, сопротивление резистора R2. При этом у нас на определенную величину сдвинется вся шкала измерений: показания термометра будут соответствовать действительности, только если мы *прибавим* (или вычтем, неважно) некоторую константу к полученной величине: $t = t' + \delta$, где t — «правильное» значение температуры (оно все же отличается от истинного значения из-за наличия случайной ошибки); t' — показания термометра; δ — величина систематической ошибки из-за сдвига шкалы. Более сложный случай систематической погрешности — если мы оставим R2 в покое, а немного изменим R5, т. е. изменим наклон характеристики термометра, или, как еще это называют, *крутизну преобразования*. Это равносильно тому, что мы *умножаем* показания на некий постоянный множитель k, и «правильное» значение будет тогда определяться по формуле: $t = k \cdot t'$. Эти виды ошибок носят название *аддитивной* и *мультипликативной* погрешностей.

О систематических погрешностях математическая статистика «ничего не знает», она работает только с погрешностями случайными. Единственный способ избавиться от систематических погрешностей (кроме, конечно, подбора прецизионных компонентов) — это процедуры калибровки (градуировки), о них мы уже говорили в этой главе ранее.

Случайные ошибки измерения и их оценка

Я предполагаю, что читатель знаком с таким понятием, как *вероятность*. Если же нет — для знакомства настоятельно рекомендую книгу [12], которая есть переиздание труда от 1946 года. Расширить кругозор вам поможет и классический учебник [13], который отличает исключительная внятность изложения (автор его, известный математик Елена Сергеевна Вентцель, кроме научной и преподавательской деятельности, также писала художественную литературу под псевдонимом И. Грекова). Более приближен к инженерной практике другой учебник того же автора [14], а конкретные сведения о приложении методов математической статистики к задачам метрологии и обработки экспериментальных данных, в том числе с использованием компьютера, вы можете найти, например, в [15]. Мы же здесь остановимся на главном — расчете случайной погрешности.

В основе математической статистики лежит понятие о *нормальном распределении*. Не следует думать, что это нечто заумное — вся теория вероятностей и матстатистика, как прикладная дисциплина в особенности, основаны на здравом смысле в большей степени, чем какой-либо другой раздел математики.

Не составляет исключения и нормальный закон распределения, который наглядно можно пояснить так. Представьте себе, что вы ждете автобус на остановке. Предположим, что автопарк работает честно, и надпись на табличке «интервал 15 мин» соответствует действительности. Пусть также известно, что предыдущий автобус отправился от остановки ровно в 10:00. Вопрос — во сколько отправится следующий?

Как бы идеально ни работал автопарк, совершенно ясно, что ровно в 10:15 следующий автобус отправится вряд ли. Пусть даже автобус выехал из парка по графику, но наверняка тут же был вынужден его нарушить из-за аварии на перекрестке. Потом его задержал перебегающий дорогу школьник. Затем он простоял на остановке из-за старушки с огромной клетчатой сумкой, которая застряла в дверях. Означает ли это, что автобус всегда только опаздывает? Отнюдь, у водителя есть план по выручке, и он заинтересован в том, чтобы двигаться побыстрее, потому он может кое-где и опережать график, не гнушаясь иногда и нарушением правил движения. Поэтому событие, заключающееся в том, что автобус отправится в 10:15, имеет лишь определенную вероятность, не более.

Если поразмыслить, то станет ясно, что вероятность того, что следующий автобус отправится от остановки в определенный момент, зависит также от того, насколько точно мы определяем этот момент. Ясно, что вероятность отправления в промежутке от 10:10 до 10:20 гораздо выше, чем в промежутке от 10:14 до 10:16, а в промежутке от 10 до 11 часов оно, если не возникли какие-то совсем уж форс-мажорные обстоятельства, скорее всего, произойдет наверняка. Чем точнее мы определяем момент события, тем меньше вероятность того, что оно произойдет именно в этот момент, и в пределе вероятность того, что любое событие произойдет *ровно* в указанный момент времени, равна нулю.

Такое кажущееся противоречие (на которое, между прочим, обращал внимание еще великий отечественный математик А. Н. Колмогоров) на практике разрешается стандартным для математики способом — мы принимаем за момент события некий малый интервал времени δt. Вероятность того, что событие произойдет в этом интервале, уже равна не нулю, а некоей конечной величине δP, а их отношение $\delta P/\delta t$ при устремлении интервала времени к нулю для того или иного момента времени равна некоей величине p, именуемой *плотностью распределения вероятностей*. Такое определение совершенно аналогично определению плотности физического тела (в самом деле, масса исчезающе малого объема тела также стремится к нулю, но отношение массы к объему конечно), и потому многие понятия математической статистики имеют названия, заимствованные из соответствующих разделов физики.

Правильно сформулированный вопрос по поводу автобуса звучал бы так: каково распределение плотности вероятностей отправления автобуса во времени? Зная эту закономерность, мы можем всегда сказать, какова вероятность того, что автобус отправится в определенный промежуток времени.

Интуитивно форму кривой распределения плотности вероятностей определить несложно. Существует ли вероятность того, что конкретный автобус отправится, к примеру, позже 10:30 или, наоборот, даже раньше предыдущего автобуса? А почему нет? — подобные ситуации в реальности представить себе очень легко. Однако ясно, что такая вероятность намного меньше, чем вероятность прихода «около 10:15». Чем дальше в обе стороны мы удаляемся от этого центрального наиболее вероятного срока, тем меньше плотность вероятности, пока она не станет практически равной нулю (то, что автобус задержится на сутки — событие невероятное, скорее всего, если такое случилось, вам уже будет не до автобусов). То есть рас-

пределение плотностей вероятностей должно иметь вид некоей колоколообразной кривой.

В теории вероятностей доказывается, что при некоторых предположениях относительно вероятности конкретных исходов нашего события, эта кривая будет иметь совершенно определенный вид, который называется *нормальным распределением вероятностей* или распределением Гаусса. Формула, описывающая плотность нормального распределения, выглядит так:

$$P(x) = \frac{1}{\sigma\sqrt{\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \qquad (2)$$

Вид соответствующих кривых при различных значениях параметра σ показан на рис. 13.5.

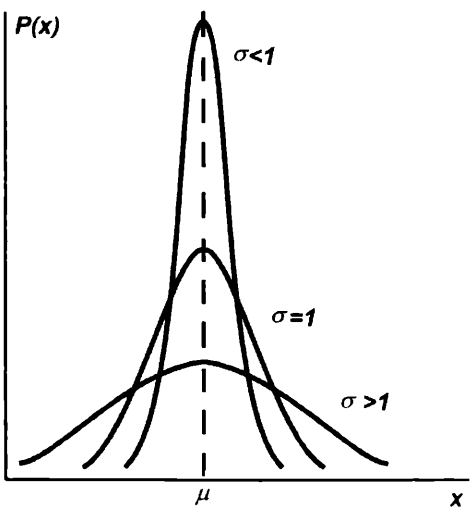


Рис. 13.5. Плотность нормального распределения вероятностей

Далее мы поясним смысл отдельных параметров в этой формуле, а пока ответим на вопрос: действительно ли реальные события, в частности, интересующие нас ошибки измерения, всегда имеют нормальное распределение? Строгого ответа на этот вопрос в общем случае нет, и вот по какой причине. Математики имеют дело с абстракциями, считая, что у нас уже есть сколь угодно большой набор отдельных *реализаций* события (в случае с автобусом это была бы бесконечная таблица пар значений «плотность вероятности — время»). В реальной жизни такой ряд невозможно получить не только потому, что для этого потребовалось бы бесконечно долго стоять около остановки и отмечать моменты отправления, но и потому, что стройная картина непрерывного ряда реализаций одного события (прихода конкретного автобуса) будет в конце концов нарушена совершенно не относящимися к делу вещами: маршрут могут отменить, остановку перенести, автопарк обанкротится, не выдержав конкуренции с маршрутными такси... да мало ли что может произойти такого, что сделает бессмысленным само определение события.

Однако все же интуитивно понятно, что, пока автобус ходит, *какое-то*, пусть теоретическое, распределение имеется. Такой идеальный бесконечный набор реализаций какого-либо события носит название *генеральной совокупности*. Именно генеральная совокупность при некоторых условиях может иметь, в частности, нормальное распределение. В реальности же мы имеем дело с *выборкой* из этой генеральной совокупности. Причем одна из важнейших задач, решаемых в математической статистике, состоит в том, чтобы имея на руках две разные выборки, доказать, что они принадлежат одной и той же генеральной совокупности — проще говоря, что перед нами есть реализации одного и того же события. Другая важнейшая для практики задача состоит в том, чтобы по выборке определить вид кривой распределения и ее параметры.

На свете может быть сколько угодно случайных событий и процессов, имеющих распределение, совершенно отличное от нормального, однако считается (и доказывается с помощью так называемой *центральной предельной теоремы*), что в интересующей нас области ошибок измерений, при большом числе измерений и истинно случайном их характере, все распределения ошибок — нормальные. Предположение о большом числе измерений не слишком жесткое — реально достаточно полутора-двух десятков измерений, чтобы все теоретические соотношения с большой степенью точности соблюдались на практике. А вот про истинную случайность ошибки каждого из измерений можно говорить с изрядной долей условности — неслучайными их может сделать одно только желание экспериментатора побыстрее закончить рабочий день. Но математика тут уже бессильна.

Полученные опытным путем характеристики распределения называются *оценками* параметров, и, естественно, они будут соответствовать «настоящим» значениям с некоторой долей вероятности, — наша задача и состоит в том, чтобы определить интервал, в котором могут находиться отклонения оценок от «истинного» значения, и соответствующую ему вероятность. Но настало время все же пояснить — что же это за параметры?

В формуле (2) таких параметра два: величины μ и σ. Они называются *моментами нормального распределения* (аналогично моментам распределения масс в механике). Параметр μ называется *математическим ожиданием* (или моментом распределения первого порядка), а величина σ — *средним квадратическим отклонением*. Нередко употребляют его квадрат, обозначаемый как D или просто σ^2 и носящий название *дисперсии* (или центрального момента второго порядка).

Математическое ожидание есть абсцисса максимума кривой нормального распределения (в нашем примере с автобусом — это время 10:15), а дисперсия, как видно из рис. 13.5, характеризует «размытие» кривой относительно этого максимума — чем больше дисперсия, тем положе кривая. Эти моменты имеют прозрачный физический смысл (вспомните аналогию с физическим распределением плотностей): математическое ожидание есть аналогия центра масс некоего тела, а дисперсия характеризует распределение масс относительно этого центра (хотя распределение плотности материи в физическом теле далеко от нормального распределения плотности вероятности).

Оценкой m_x математического ожидания μ служит хорошо знакомое нам со школы среднее арифметическое:

$$m_x = \frac{1}{n}\sum_i x_i \qquad (3)$$

Здесь: n — число измерений; i — текущий номер измерения ($i = 1,\ldots,n$); x_i — значение измеряемой величины в i-м случае.

Оценка s^2 дисперсии σ^2 вычисляется по формуле:

$$s^2 = \frac{1}{n-1}\sum_i (x_i - m_x)^2. \qquad (4)$$

Оценка среднего квадратического отклонения, соответственно, будет:

$$s = \sqrt{s^2}. \qquad (5)$$

Здесь $(x_i - m_x)$ — отклонения конкретных измерений от ранее вычисленного среднего.

Следует особо обратить внимание, что сумму квадратов отклонений делить нужно именно на $n - 1$, а не на n, как может показаться на первый взгляд, иначе оценка получится неверной. Второе, на что следует обратить внимание, — разброс относительно среднего характеризует именно среднее квадратическое отклонение, вычисленное по формулам (4) и (5), а не среднее арифметическое отклонение, как ошибочно рекомендуют в некоторых школьных справочниках, — последнее дает заниженную и смещенную оценку (не напоминает ли вам это аналогию со средним арифметическим и действующим значениями переменного напряжения из *главы 4*?).

> ***Заметки на полях***
>
> Кроме математического ожидания, средние значения распределения вероятностей характеризуют еще величинами, называемыми *модой* и *медианой*. В случае нормального распределения все три величины совпадают, но в других случаях они могут оказаться полезными: мода есть абсцисса наивероятнейшего значения (т. е. максимума на кривой распределения, что полностью отвечает бытовому понятию о моде), а медиана выборки есть такая точка, что половина выборки лежит левее ее, а вторая половина — правее.

Этими формулами для расчета случайных погрешностей можно было бы ограничиться, если бы не один важный вопрос: оценки-то мы получили, а вот в какой степени они отвечают действительности? Правильно сформулированный вопрос будет звучать так: какова вероятность того, что среднее арифметическое отклоняется от «истинного» значения (т. е. математического ожидания) не более чем на некоторую величину δ (например, на величину оценки среднего квадратического отклонения s)?

Величина δ носит название *доверительного интервала*, а соответствующая вероятность — *доверительной вероятности* (или *надежности*). Обычно решают задачу, противоположную сформулированной, — задаются величиной надежности и вычисляют доверительный интервал δ. В технике принято задаваться величиной надежности 95%, в очень уж серьезных случаях — 99%. Простейшее правило для обычных измерений при этом таково: *при условии достаточно большого числа*

измерений (практически более 15–20) *доверительной вероятности в 95% соответствует доверительный интервал в 2s, а доверительной вероятности в 99% — доверительный интервал в 3s*. Так формулируется известное правило *трех сигма*, согласно которому за пределы утроенного квадратического отклонения не выйдет ни один результат измерения, но на практике это слишком жесткое требование. Если мы не поленимся провести не менее полутора десятков отдельных измерений величины x, то с чистой совестью можем записать, что результат будет равен:

$$x = m \pm 2s$$

Регрессия и метод наименьших квадратов

Все сказанное относилось к случаю, когда мы измеряем одну величину, имеющую некоторую случайную погрешность. Однако на практике нам часто требуется по экспериментальным данным получить оценку некоторой функции $y(x)$ — фактически это задача построения кривой по результатам опытных данных, которую вам, несомненно, приходилось не раз решать, если вы обучались в техническом вузе.

Процесс проведения кривой через какие-либо точки (расчетные или экспериментальные) в общем случае называется *аппроксимацией*. Аппроксимацию следует отличать от *интерполяции* (когда по совокупности имеющихся значений функции и переменных рассчитывают значение функции в некоторой точке между ними) и *экстраполяции* (когда рассчитывают значения функции вне области, охваченной имеющимися значениями, в предположении, что там кривая ведет себя так же). Насчет последней операции следует отметить, что полиномы, полученные регрессионным способом (см. далее), за исключением разве что прямой линии, обычно для проведения экстраполяции не годятся — т. к. не несут в себе физического смысла и вне экспериментальной области могут очень сильно расходиться с реальной картиной.

Провести кривую, аппроксимирующую опытные данные, можно от руки на миллиметровке, но как решать такую задачу «правильно»? Причем, как и в предыдущем случае, желательно бы иметь возможность оценить погрешности измерений.

Принцип такого построения при наличии случайных ошибок измерения иллюстрирует рис. 13.6. Разумно было бы проводить кривую (в нашем случае — прямую) так, чтобы отклонения Δy_i были бы минимальными в каждой точке. Однако просто минимизировать сумму отклонений не получится — они имеют разный знак, и минимум получился бы при очень больших отрицательных отклонениях. Можно минимизировать сумму абсолютных значений отклонений, однако это неудобно по ряду чисто математических причин, потому используют уже знакомую нам сумму квадратов отклонений, — только ранее это было отклонение от среднего арифметического одной величины x, а теперь это отклонение опытных данных от кривой $y(x)$:

$$\sum_i \Delta y_i^2 \Rightarrow \min$$

Такой метод называется *методом наименьших квадратов*.

Кстати, а какую именно кривую выбрать? Ведь кривые бывают разные: прямая, парабола, экспонента, синусоида... Опыт показывает, что на практике можно ограни-

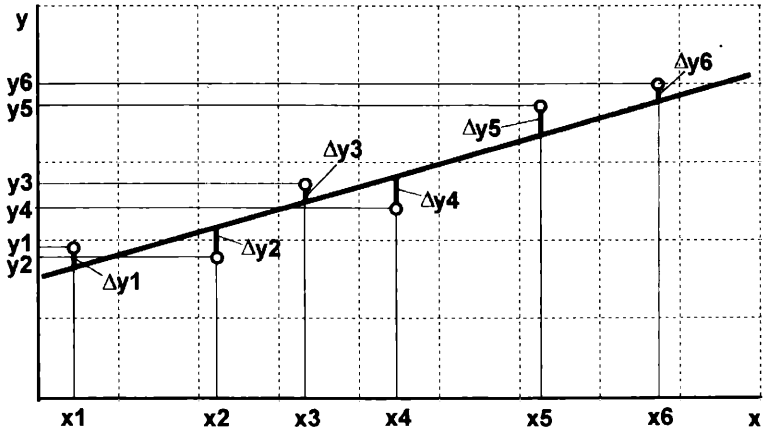


Рис. 13.6. Проведение аппроксимирующей прямой по экспериментальным данным

читься полиномом, соответствующим разложению функции в ряд Тейлора (в математике доказывается, что любую другую непрерывную функцию всегда можно представить в виде такого ряда):

$$y = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \ldots \qquad (6)$$

Это уравнение называется *уравнением регрессии*. Отметим, что здесь мы рассматриваем наипростейший случай — зависимость y от одного параметра x. В общем случае независимых переменных может быть несколько, но для наших целей простейшего случая достаточно. Еще отметим, что величины x_i считаются неслучайными — если в каждой i-й точке проводится несколько измерений, то в формулу надо подставлять среднее. Случайными здесь считаются только величины y.

Итак, в качестве исходных данных у нас имеется некий набор значений x_i в количестве n штук. Надо провести кривую, соответствующую уравнению (6), так, чтобы сумма квадратов отклонений была минимальна:

$$\sum_i \Delta y_i^2 = \sum_i (y_i - a_0 - a_1 x_i - a_2 x_i^2 - \ldots)^2 \Rightarrow \min \qquad (7)$$

Какой степени полином должен быть? Из элементарной геометрии известно, что через две точки можно провести прямую (полином первой степени), через три — параболу (второй степени) и т. д., т. е. максимально возможная степень полинома на единицу меньше, чем число экспериментальных данных. Однако через две точки можно провести только одну прямую, и мы никогда не сможем оценить погрешностей — т. е. узнать, насколько наша прямая отличается от того, что имеет место в действительности. Поэтому чем избыток точек больше, тем лучше (в идеале необходимы те же 15–20 точек, но на практике для линейной зависимости можно обойтись и тремя-пятью точками). Оптимальную же степень определяют так: строят несколько полиномов разной степени и смотрят на среднеквадратическое отклонение. Когда оно с увеличением степени полинома перестанет уменьшаться (или это уменьшение незначительно), то нужная степень достигнута.

Я не буду здесь вдаваться в подробности реализации метода наименьших квадратов — это бессмысленно, т. к. его обычно реализуют в виде готовой программы.

Такую программу под названием RegrStat вы можете скачать с моей домашней странички по адресу **http://revich.lib.ru** из раздела **Программы**. Умеет строить простейшие регрессионные зависимости и Microsoft Excel, причем в том числе и как функцию от многих переменных, но только первого порядка (линейные полиномы). Ну, и конечно, существует множество специальных программных пакетов для этой цели.

Разновидности погрешностей

Мы в предыдущем изложении часто упоминали понятие погрешности, приводя его то в процентах, то в абсолютных величинах. Систематизируем эти представления и определим следующие три вида погрешностей:

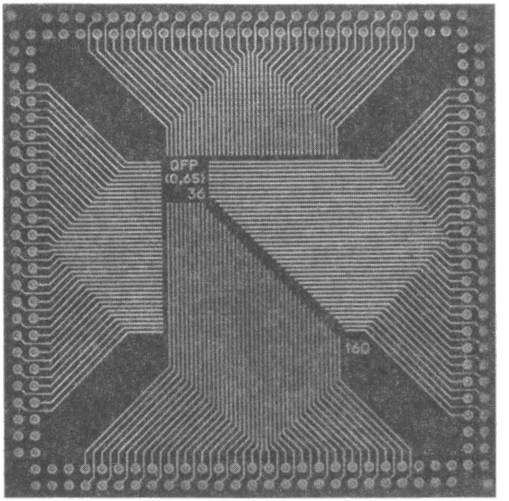
- **абсолютная погрешность** — в единицах измеряемой величины;
- **относительная погрешность** — абсолютная, но выраженная в процентах от значения измеряемой величины;
- **относительная приведенная погрешность** — абсолютная, но выраженная в процентах от всего диапазона измерений.

Последняя величина, если она соответствует стандартному ряду (например, 1,0; 0,75; 0,5; 0,25; 0,1 и т. п.), еще называется *классом точности* и обычно указывается в технических описаниях приборов.

При определении относительной приведенной погрешности учитывают все ошибки (их абсолютные значения): и случайную, и аддитивную, и мультипликативную погрешности. Причем в последнем случае за величину погрешности принимают значение мультипликативной погрешности в конце шкалы — ведь она зависит от измеряемой величины. Отсюда видно, что если мультипликативная погрешность доминирует, то выгоднее как можно больше «ужимать» диапазон измеряемых значений. С другой стороны, аддитивная и случайная погрешности от диапазона не зависят, и уменьшение его приведет к тому, что их вклад увеличится, — в частности, именно поэтому мы старались в схеме на рис. 13.4 «раздуть» выходное напряжение ОУ до максимума, ограничивая максимальный ток значением резистора R7, а не величиной напряжения.

Теперь мы можем грамотно ответить на вопрос, поставленный в начале раздела: если погрешность мультиметра на пределе 2 В составляет 0,5% (без дополнительных оговорок это означает относительную приведенную погрешность), то любое показываемое им значение на этом пределе (в том числе указанное нами ранее значение 1,000 В) отклонится от истинного значения не более, чем на ±10 мВ в 95 случаях из ста. Довольно часто для импортных мультиметров вы можете встретить и такую запись в инструкции «точность указывается как ± % от измеренного ± количество единиц младшего разряда», т. е. погрешность в этом случае просто относительная. Тогда правильный ответ при указании точности на пределе 2 В, равной ±0,5%±1, будет звучать иначе: измеренное значение равно 1,000 В ±5 мВ ±1 единица младшего разряда.

А теперь оставим эти скучные материи и перейдем к куда более интересным вещам — к логическим микросхемам и цифровой электронике.



ЧАСТЬ III

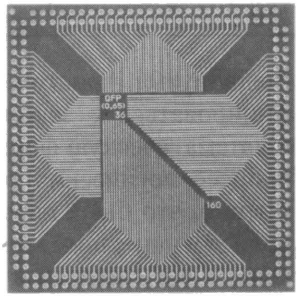
Цифровой век

Глава 14. На пороге цифрового века
Математическая логика и ее представление в технических устройствах

Глава 15. Математическая электроника, или игра в квадратики
Устройство логических микросхем и двоичные операции

Глава 16. Устройства на логических схемах
Мультивибраторы, формирователи, триггеры, счетчики...

Глава 17. Откуда берутся цифры
Цифроаналоговые и аналого-цифровые преобразователи



ГЛАВА 14

На пороге цифрового века

Математическая логика и ее представление в технических устройствах

— Теперь давайте сочтем, сколько у нас всего. Портос?
— Тридцать экю.
— Арамис?
— Десять пистолей.
— У вас, д'Артаньян?
— Двадцать пять.
— Сколько это всего? — спросил Атос.
— Четыреста семьдесят пять ливров! — сказал д'Артаньян, считавший, как Архимед.

А. Дюма. «Три мушкетера»

Все началось, конечно, с Аристотеля, который жил в IV веке до нашей эры. Когда читаешь вступление к любой популярной книге, посвященной чему угодно: от изящных искусств до биологии, химии, физики и математики, — возникает впечатление, что Аристотель был каким-то сверхчеловеком. В самом деле, гении встречаются, но нельзя же быть гением настолько, чтобы разработать основы вообще всего, на чем зиждется современная цивилизация! Тем не менее, и авторы не врут, и Аристотель сверхчеловеком не был. Во-первых, знаний было тогда накоплено еще не очень много, и обозреть их все — задача вполне посильная для человека острого ума и выдающихся способностей. Во-вторых, Аристотель работал не один, его метод — коллективный мозговой штурм, это просто история донесла до нас фактически одно только его имя.

Но главное, пожалуй, в другом — древние рассматривали упомянутые нами дисциплины во взаимосвязи. Аристотель четко разделил только науку и ремесла («техно», по-гречески), наука же делились на практические (этику и политику) и теоретические (физику и логику) дисциплины, но и они рассматривались как составные части единой науки. В чем древние, конечно, были более правы, чем мы, вынужденно поделившие области человеческой деятельности на множество автономных разделов.

Для нас важно, что главной составной частью науки считалась именно логика — искусство рассуждения. Вот она-то и послужила той основой, из которой выросла цифровая техника и все многообразие информационных технологий, которые окружают нас теперь на каждом шагу.

Выдвинутые Аристотелем законы логики, которые с его же подачи стали идентифицироваться с законами мышления вообще, неоднократно пытались привести в математическую форму. Некто Луллий в XIII веке попытался даже механизировать процесс логических рассуждений, построив «Всеобщий решатель задач» (несомненно, это была первая попытка построения «думающей машины»). Формализацией логики занимался Лейбниц, искавший универсальный язык науки, и в конце концов все сошлось в двух работах английского математика Джорджа Буля, который жил и работал уже в середине XIX века. Любопытно название второй из этих работ — «Исследование законов мышления», первая же работа называлась поскромнее, но без «мышления» и тут не обошлось, — в названии фигурировало слово «рассуждения». То есть и сам Буль, и еще сто лет после него, до середины XX века, и все его предшественники в течение двух с большим лишком тысяч лет, прошедших со времен Аристотеля, — никто так и не усомнился, что в основе мышления лежит именно та логика, которая называется «аристотелевой». И лишь в XX веке, после работ Геделя и Тьюринга, и особенно в связи с благополучно провалившимися (как и у Луллия за 700 лет до того) попытками создания «искусственного интеллекта», до ученых, наконец, начало доходить, что мышление вовсе не имеет логической природы, а логика есть лишь удобный способ сделать свои рассуждения доступными окружающим.

Главное же следствие возникновения математической логики выявилось совсем не в исследованиях мышления, как оно виделось Лейбницу и Булю. Его обозначил в своей магистерской диссертации от 1940 года великий Клод Шеннон (рис. 14.1) —

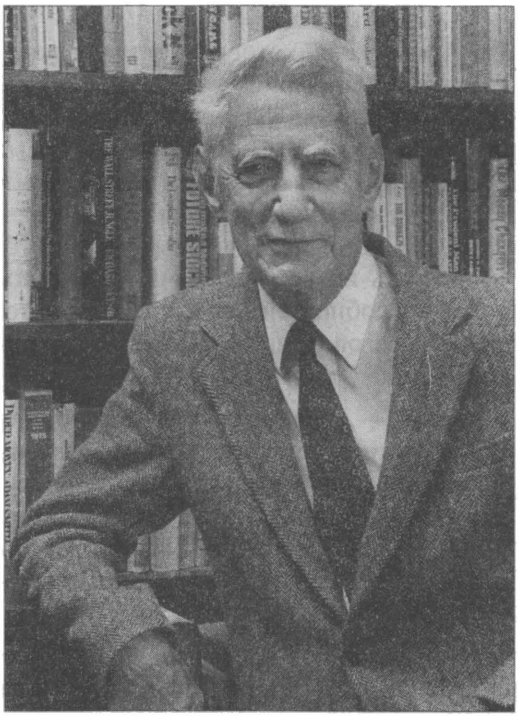


Рис. 14.1. Клод Элвуд Шеннон (Claude Elwood Shannon), 1916–2001.
Фото Lucent Technologies Inc./Bell Labs

оказалось, что булевы законы в точности совпадают с принципами функционирования релейных электрических схем. Что самое поразительное — все компоненты, необходимые для моделирования законов логики с помощью электрических устройств (реле, выключатели), были известны еще до публикации Булем своих работ, но в течение еще почти ста лет никто не обращал на это внимания (Шеннон скромно утверждал, что случилось так, что до него просто никто не владел математикой и электротехникой одновременно). Не обратил на это внимание даже Чарльз Бэббидж, сконструировавший еще задолго до работ Буля механическую вычислительную («аналитическую») машину, — а ведь был знаком и с самим Булем, и с его работами!

Основные операции алгебры Буля

Булева алгебра имеет дело с абстрактными логическими переменными. Эти переменные можно интерпретировать по-разному, но интерпретацию мы пока отложим. Вне зависимости от интерпретации, для логических переменных определены некоторые операции, подчиняющиеся определенным правилам. Базовые операции такие:

- операция логического сложения двух операндов — операция объединения, операция «ИЛИ» («OR»), обозначается обычным знаком сложения;
- операция логического умножения двух операндов — операция пересечения, операция «И» («AND»), мы будем обозначать ее крестиком, чтобы отличить от обычного умножения;
- операция отрицания для одного операнда — операция «НЕ» («NOT»), обозначается черточкой над символом операнда.

В математике операция логического сложения (дизъюнкция) обозначается еще знаком $\vee$, а умножения (конъюнкция) — $\wedge$. Кроме того, операция умножения часто обозначается знаком &, и это обозначение нам встретится, когда мы перейдем к микросхемам. В языке программирования C («Си»), а через него и в огромном количестве других языков программирования, принято логическое умножение «И» обозначать знаком &, сложение «ИЛИ» — знаком вертикальной черты |, а отрицание «НЕ» знаком восклицания ! (см. также «Подробности» далее). Все остальные логические операции могут быть записаны как сочетания этих трех основных, хотя некоторые часто употребляемые также имеют собственные обозначения (как, например, операция «исключающее ИЛИ», обозначаемая значком $\oplus$ или «=1» — о ней мы расскажем дальше).

Любая конкретная интерпретация булевых операндов — математическая или техническая — должна отвечать правилам булевой алгебры. Например, оказалось, что этим правилам отвечают множества (отсюда другие названия тех же операций: «пересечение» и «объединение»). Программисты имеют дело с логическими переменными 0 и 1, которые также есть одно из представлений булевых операндов. Следует отчетливо понимать, что вне зависимости от интерпретации (включая и напряжения в релейных цепях по Шеннону, и операнды в компьютерной програм-

ме, и множества в математике), любые булевы объекты ведут себя одинаково: так, операция пересечения множеств совершенно адекватна операции «И» с логическими переменными или соответствующей манипуляции с выключателями в электрической сети.

В булевой алгебре многое совпадает с обычной — например, справедливы правила типа $A + B = B + A$ или $A + (B + C) = (A + B) + C$), но для нас важны как раз отличия. Вот они: $A + A = A$ (а не $2A$, как было бы в обычной алгебре), а также $A \times A = A$ (а не A^2). Последнее уравнение в обычной алгебре, впрочем, имело бы решение, причем сразу два: 0 и 1. Таким путем обычно и переходят к интерпретации булевых операндов как логических переменных, которые могут иметь только два состояния: 1 и 0 или «правда» (true) и «ложь» (false). В этом представлении мы действительно можем попробовать с помощью определенных ранее операций записывать некоторые высказывания в виде уравнений и вычислять их значения, что дает иллюзию формального воспроизведения процесса мышления.

Но сначала надо определить, как и в обычной алгебре, правила, которым подчиняются операции, — т. е. таблицу логического сложения и таблицу логического умножения. Они таковы:

$$0 + 0 = 0 \qquad 0 \times 0 = 0$$
$$0 + 1 = 1 \qquad 0 \times 1 = 0$$
$$1 + 0 = 1 \qquad 1 \times 0 = 0$$
$$1 + 1 = 1 \qquad 1 \times 1 = 1$$

Операция отрицания «НЕ» меняет 1 на 0 и наоборот.

Примеры записи логических выражений обычно приводят для каких-нибудь бытовых высказываний, но мы поступим нетрадиционно — приведем пример из области математики. Пусть высказывание состоит в следующем: «x меньше нуля или x больше 1 и y меньше 2». Как записать это высказывание? Введем следующие логические переменные: $A = (x < 0)$; $B = (x > 1)$; $C = (y < 2)$. Как мы видим, все они могут принимать только два значения: «правда» (если условие выполняется) и «ложь» (если не выполняется). Обозначим значение всего выражения через D. Тогда высказывание записывается так:

$$D = (A + B) \times C \qquad (1)$$

Можно записать и так:

$$D = (A \text{ ИЛИ } B) \text{ И } C$$

Или так:

$$D = (A \text{ OR } B) \text{ AND } C$$

Или, наконец, так:

$$D = ((x < 0) \text{ OR } (x > 1)) \text{ AND } (y < 2)$$

Последняя запись хорошо знакома всем, кто изучал язык программирования Pascal. На языке C та же запись выглядит непонятнее:

$$D = ((x < 0) \,||\, (x > 1)) \,\&\&\, (y < 2)$$

Подробности

О великий и могучий язык С! В нем самую простую вещь можно запутать до полной потери смысла. В нашем случае то же самое выражение можно было бы записать как ((x < 0) | (x > 1)) & (y < 2), и ничего бы не изменилось. В этом языке (в отличие от Pascal) есть две разновидности логических операций: обычные («логическое И» &&, «логическое ИЛИ» ||) и поразрядные («поразрядное И» &, «поразрядное ИЛИ» |). Есть и, соответственно, «логическое НЕ» (!) и «поразрядное НЕ» (~). Термин «поразрядные» означает, что они применимы к многоразрядным *двоичным* числам. В результате их применения тоже получается многоразрядное двоичное число — необязательно ноль или единица, как в случае логических. Поскольку наши результаты операций сравнения содержат только один двоичный разряд (либо соблюдается, либо не соблюдается), то в этом случае логические и поразрядные операции оказываются идентичны, и можно писать и так, и так. А вот если в операциях участвуют обычные числа, то результат будет разный: «10&&7» равно «логической 1» (отличное от нуля значение всегда интерпретируется, как «правда»), тогда как «10&7» равно 2 (почему, будет рассказано далее). Как мы узнаем в *главе 20*, эти особенности играют большую роль в программировании микроконтроллеров на языке С.

Пусть $x = 0{,}5$, $y = 1$. Чему будет равно D в этом случае? Очевидно, что выражение $(A + B)$ примет значение «ложь» (0), поскольку x не удовлетворяет ни одному из условий A и B. Переменная C примет значение «правда» (1), но на результат это уже не повлияет, т. к. произведение 0 на 1, согласно таблице логического умножения, равно 0. То есть D в этом случае есть «ложь». Если же принять значение $x = -0{,}5$, оставив y равным 1, то D примет значение «правда».

Интересный оборот примут события, если вместо «OR» между A и B поставить «AND», — легко догадаться, что выражение в скобках тогда не будет «правдой» ни при каком значении x, поскольку условия «x меньше 0» и «x больше 1» взаимоисключающие. Потому результирующее условие D всегда будет принимать значение 0, т. е. «ложь». Но вот если мы изменим выражение следующим образом:

$$D = \overline{(A \times B)} \times C, \qquad (2)$$

то есть инвертируем выражение в скобках с помощью операции «НЕ», то получим обратный результат: D всегда будет «правдой» (черточкой над символом или выражением как раз и изображается инверсия). Интересно, что тот же самый результат мы получим, если запишем выражение следующим образом:

$$D = (\overline{A} + \overline{B}) \times C. \qquad (3)$$

Это свойство выражается в так называемых *правилах де Моргана* (учителя Буля):

$$\overline{A \times B} = \overline{A} + \overline{B},$$
$$\overline{A + B} = \overline{A} \times \overline{B}.$$

Отметим, что из таблиц логического умножения и сложения вытекает еще одно любопытное следствие. Дело в том, что ассоциация значения «ложь» с нулем, а «правды» с единицей (положительная логика), есть действие вполне произвольное — ничто не мешает нам поступить наоборот (отрицательная логика). Такая замена приводит к тому, что все операции «ИЛИ» меняются на «И» и наоборот (рассмотрите таблицы внимательно). А вот операция «НЕ» к такой замене индифферентна — 0 меняется на 1 в любой логике.

Далее приведены несколько соотношений, которые вместе с правилами де Моргана помогают создавать и оптимизировать логические схемы. Некоторые из них очевидны, иные же — совсем нет.

Ассоциативный закон умножения:

$$A \times B \times C = (A \times B) \times C = A \times (B \times C).$$

Ассоциативный закон сложения:

$$A + B + C = (A + B) + C = A + (B + C).$$

Другие формулы:

$$A \times A = A, \qquad A + A = A,$$

$$A + \overline{A} = 1, \qquad A \times \overline{A} = 0,$$

$$A \times 1 = A, \qquad A + 1 = 1,$$

$$A \times 0 = 0, \qquad A + 0 = A,$$

$$A \times (B + C) = A \times B + A \times C,$$

$$A + A \times B = A,$$

$$A + B \times C = (A + B) \times (A + B),$$

$$\overline{1} = 0, \qquad \overline{0} = 1.$$

Булева алгебра на выключателях и реле

Для того чтобы представить булевы переменные и операции над ними с помощью технических устройств (то, что сделал Клод Шеннон в своей диссертации), надо придумать схемы, которые воспроизводили бы эти операции согласно изложенным правилам. Для этого осталось сделать только один шаг: пусть наличие напряжения (высокий уровень напряжения) в некоторой точке цепи представляет собой логическую единицу, а отсутствие напряжения (низкий уровень) — логический ноль. В этом случае логика носит название «положительной». Если принять за единицу низкий уровень, а за ноль — высокий, логика будет «отрицательной». Как мы уже знаем, переход с положительной логики на отрицательную означает замену всех «И» на «ИЛИ» и наоборот. В дальнейшем, если это специально не оговорено, мы всегда будем иметь в виду положительную логику.

Самые простые варианты схем, реализующих базовые логические операции в этом случае, показаны на рис. 14.2. Здесь операции «И» и «ИЛИ» выполняются обычными кнопками без фиксации. Каждая из них соответствует одной логической переменной, которая принимает значение 1, если контакты замкнуты, и 0 — если разомкнуты. На выходе значению 0 соответствует погасший светодиод, значению 1 — горящий. Легко понять, что работать эти схемы будут именно так, как указано в таблицах для соответствующих логических операций. Для технических устройств правила соответствия входов и выхода называются «таблицами истинности» (или «таблицами состояния») и показаны в табл. 14.1.

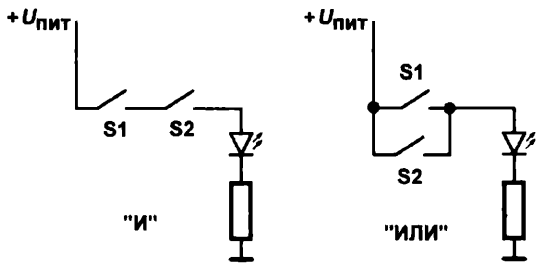


Рис. 14.2. Схемы реализации логических функций на кнопочных выключателях

Таблица 14.1. *Таблицы истинности логических операций «ИЛИ» и «И»*

«ИЛИ»			«И»		
Вх1	**Вх2**	**Вых**	**Вх1**	**Вх2**	**Вых**
0	0	0	0	0	0
0	1	1	0	1	0
1	0	1	1	0	0
1	1	1	1	1	1

Однако если разобраться поглубже, то придется констатировать, что настоящими входными логическими переменными для таких схем являются движения пальца, нажимающего на кнопку. В частности, операция «НЕ» здесь будет означать нажатие на кнопку с замкнутыми контактами, а каскадное соединение таких схем для реализации сложных выражений предполагает наличие человека, транслирующего выходной сигнал одной схемы (состояние светодиода) во входной другой схемы (состояние контактов). Логично поставить вместо такого человека, тупо выполняющего предопределенные действия, техническое устройство. И здесь помогут уже хорошо нам известные электромагнитные реле.

В схемах на рис. 14.3 как для входов, так и для выхода, наличие напряжения соответствует логической 1, отсутствие его — логическому 0. (Можно для наглядности подключить к выходу светодиод или лампочку, но суть дела от этого не изменится.) Способ подачи входного сигнала не указан, т. к. предполагается, что источник входного напряжения может быть произвольным (разумеется, его мощность должна быть достаточной, чтобы заставить реле сработать) — в том числе и такая же схема на реле. Это иллюстрируется схемой на рис. 14.3, *справа*, где изображена схема составного элемента «И-НЕ» на трех реле, представляющего собой объединение элемента «И» (такого же, как на рис. 14.3, *слева*) и элемента просто «НЕ» (инвертора), который есть не что иное, как одиночное реле с выходом через нормально замкнутые, а не нормально разомкнутые контакты.

Таблица истинности для элемента «И-НЕ», приведенная в табл. 14.2, будет инверсией представленной в табл. 14.1 таблицы истинности для элемента «И». Легко видеть, что она не совпадает с таблицей для «ИЛИ», как могло бы показаться на

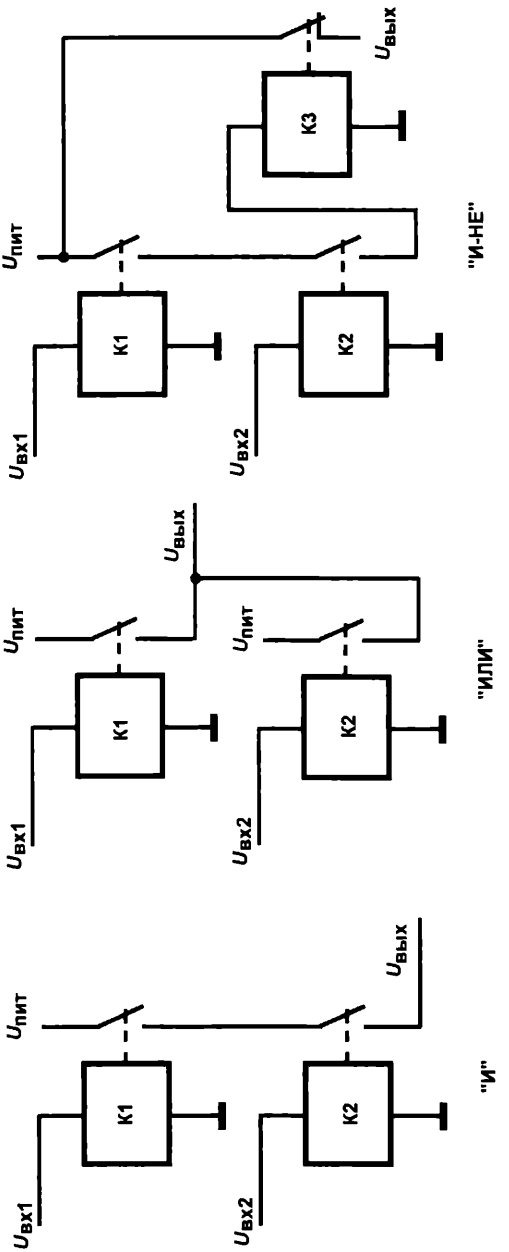


Рис. 14.3. Реализация логических функций на электромагнитных реле

первый взгляд. Аналогично составляется элемент «ИЛИ-НЕ» — из схемы «ИЛИ», показанной на рис. 14.3, *посередине*, и инвертора. Таблица истинности для него также приведена в табл. 14.2.

Таблица 14.2. *Таблица истинности логических элементов «И-НЕ» и «ИЛИ-НЕ»*

«И-НЕ»			«ИЛИ-НЕ»		
Вх1	**Вх2**	**Вых**	**Вх1**	**Вх2**	**Вых**
0	0	1	0	0	1
0	1	1	0	1	0
1	0	1	1	0	0
1	1	0	1	1	0

В большинстве современных применений логических микросхем используются именно элементы «И-НЕ» и «ИЛИ-НЕ», а не чистые «И» и «ИЛИ», — так удобнее и для разработчиков микросхем (где в качестве ключей служат транзисторы, которые инвертируют сигнал, см. далее), и для схемотехников. Для того чтобы было проще разбираться в логических схемах, не заучивая таблицы истинности, работу элементов можно запомнить следующим образом: элемент «И» дает единицу на выходе только, если на входах *одновременно* есть единица, элемент «ИЛИ» — если *хотя бы на одном* из входов единица. Возможно, вам еще проще будет запомнить так: элемент «ИЛИ» дает *единицу* на выходе, если на входах «хотя бы одна единица», а элемент «И» дает *ноль* на выходе, если на входах «хотя бы один ноль». Элементы с инверсией по выходу будут давать в тех же случаях обратные значения.

Интересно рассмотреть вопрос — а нельзя ли упростить схемы этих комбинированных элементов, исключив из них третье реле, выполняющее инверсию? В самом деле, большинство реле имеют перекидные контакты, так за чем же дело стало? — меняем нормально разомкнутые контакты на нормально замкнутые, и все! Легко заметить, что такая замена не будет адекватной — мы инвертируем здесь не общий выход элемента, а выходы каждого реле в отдельности, что равносильно инвертированию входов. Если обратиться к правилам де Моргана, то мы увидим, что такое изменение схемы приведет к тому, что элемент «И» превратится в «ИЛИ-НЕ», а «ИЛИ», соответственно, в «И-НЕ». Я советую читателю посидеть над этими соображениями и вывести таблицы истинности самостоятельно, чтобы убедиться, что все сказанное — правда. Другое полезное упражнение состоит в том, чтобы попытаться самому построить трехвходовые элементы, соответствующие уравнениям $A + B + C$ и $A \times B \times C$ (они будет состоять из трех реле).

Тем, кто не разобрался как следует в этом по необходимости кратком изложении, среди прочих источников особенно порекомендую обратиться к [16] — книге, написанной очень простым и понятным языком, ориентированным на неподготовленного читателя, но вместе с тем излагающей предмет во всех подробностях.

То же самое, но на транзисторах и диодах

Ясно, что использование реле для построения логических схем — метод, мягко говоря, несовременный. Хотя в истории и отмечены случаи построения целых компьютеров на основе реле (к ним принадлежали, в частности, легендарные Mark-I и Mark-II Говарда Эйкена, запущенные в эксплуатацию в 1944 и 1947 годах соответственно), но у них было слишком много недостатков: прежде всего, крайне низкие надежность и быстродействие, порядка 20–30 Гц (не килогерц и тем более не мегагерц, а именно герц). Конечно, по сравнению с электромеханическими ручными калькуляторами это было просто сказкой (типовое время операции сложения у Mark-I составляло 0,3 с, т. е. в десятки и сотни раз превышало «быстродействие» человека с механическим калькулятором), и машины эти широко использовались на практике. Тем не менее, такое быстродействие казалось уже тогда недостаточным, потому довольно быстро перешли к использованию ламп, что позволило достичь порогов в десятки и сотни килогерц, а затем и транзисторов, с которыми частота работы возросла до единиц и десятков мегагерц.

> ***Легенда о «баге»***
>
> С использованием реле в компьютерной технике связана легендарная история о возникновении термина «баг», как ошибки в программе. В буквальном переводе «bug» означает «жучок». В 1947 году между контактами одного из реле Mark-II застряла мошка, вызвав неисправность. Когда мошку извлекли, молодая сотрудница Эйкена Грейс Хоппер (позднее — крупнейший авторитет в программировании и единственная в истории женщина-адмирал флота США) приклеила ее между страницами лабораторного журнала с подписью: «первый случай выловленного бага». Страница эта сейчас хранится в музее Смитсоновского института.

Как же можно построить наши логические элементы на транзисторах? На рис. 14.4 показаны для примера схемы так называемой *диодно-транзисторной логики*, которая широко использовалась в производстве гибридных микросхем (т. е. еще до окончательной победы твердотельной электроники, см. *главу 11*). В элементе «И-НЕ» (*слева*) в нормальном состоянии транзистор открыт, и на выходе его логический ноль, так что подача логической единицы на входы ничего не изменит. А подача логического нуля хотя бы на один из входов приведет к тому, что соот-

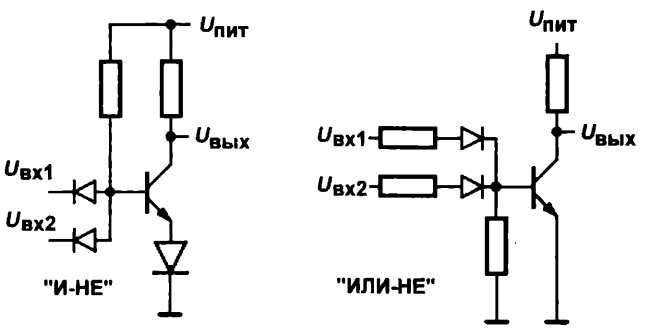


Рис. 14.4. Схемы реализации логических функций на диодах и транзисторах

ветствующий диод откроется и станет шунтировать переход база-эмиттер, в результате чего транзистор закроется, и на выходе возникнет логическая единица, что соответствует функции «И-НЕ». Диод в эмиттере нужен для обеспечения надежного запирания транзистора.

На схеме *справа* наоборот, транзистор в нормальном состоянии заперт, и на выходе логическая единица, а подача хотя бы одной логической единицы на входы откроет соответствующий диод и через него — транзистор, на выходе тогда установится логический ноль, что соответствует функции «ИЛИ-НЕ». «Подпирающий» диод здесь не требуется, зато требуются токоограничивающие резисторы на входах. Схему инверсии «НЕ» специально рисовать не имеет смысла, т. к. любой транзистор, включенный по схеме с общим эмиттером, как мы знаем из *главы 6*, есть инвертор.

В случае необходимости обычные «И» и «ИЛИ» можно соорудить из этих схем, просто добавив к ним еще по одному транзисторному каскаду, но это снизит быстродействие и повысит потребление схемы. Отсюда понятно, почему разработчикам было удобнее проектировать микросхемы с инверсией, а не с «чистыми» булевыми функциями.

В этих схемах всплывает один вопрос, который для релейных схем был неактуален: с какого именно уровня напряжение считать логическим нулем, а с какого — единицей? В релейных схемах ноль — это полный разрыв цепи, а единица — полное ее замыкание. Здесь же не совсем так: на коллекторе открытого транзистора в *левой* схеме будет напряжение около 0,8–1 В, в то время как в *правой* — всего около 0,2–0,3 В. В то же время при закрытом транзисторе вроде бы напряжение логической единицы должно быть равно напряжению питания (токами утечки пренебрегаем). Однако оно тут же упадет, если мы нагрузим выход входом другой схемы типа «ИЛИ-НЕ», поскольку там требуется обеспечить определенный ток базы.

Поэтому для транзисторов и микросхем задают не точный порог изменения с нуля на единицу, который, как мы видим, непостоянен, а пределы: ниже определенного значения считают выход находящимся в состоянии нуля (для схем на рис. 14.4 подойдет значение 1,2–1,5 В), а выше другого определенного значения (например, при питании 5 В пусть это будет 3,5 В) — в состоянии логической единицы. В промежутке схему считают находящейся в нерабочем режиме (в зоне неопределенности). При этом приходится ограничивать число устройств, подключаемых одновременно к выходу (или потребляемый по выходу ток). Для того чтобы расширить возможности таких схем, в серию одинаковых по типу применяемой схемотехники микросхем вводят специальные чипы *буферов* — т. е. усилителей мощности сигнала, которые никакой логической функции не осуществляют, а просто усиливают сигнал по току. Иногда такие буферы совмещают с функцией инверсии.

Но прежде чем мы перейдем к логическим микросхемам, необходимо немного углубиться в теорию и терминологию и понять, что такое двоичные коды и как двоичная арифметика соотносится с булевой алгеброй.

О двоичной и других системах счисления

О том, что мы считаем в десятичной системе потому, что у нас десять пальцев на двух руках, осведомлены, вероятно, все. У древних ацтеков и майя в ходу была двадцатеричная система (вероятно потому, что закрытая обувь в их климате была не в моде). Вместе с тем, история показывает, что привязка к анатомическим особенностям строения человеческого тела совершенно необязательна. Со времен древних вавилонян у нас в быту сохранились остатки двенадцатеричной и шестидесятеричной систем, что выражается в количестве часов в сутках и минут в часах или, скажем, в том, что столовые приборы традиционно считают дюжинами или полудюжинами (а не десятками и пятерками). Так что само по себе основание системы счисления не имеет значения — точнее, оно есть дело привычки и удобства.

Однако такое положение справедливо лишь для ручного счета — для компьютеров выбор системы счисления имеет большее значение. Попробуем ответить на вопрос — почему? Для этого нам придется сначала разобраться — как мы, собственно говоря, считаем, что при этом происходит, что такое вообще система счисления и ее основание.

Позиционные и непозиционные системы счисления. Десятичная система

Число — одна из самых удивительных абстрактных сущностей. Нет никаких сомнений, что число, количество предметов — есть вполне объективно существующая характеристика. В отличие, к примеру, от понятия цвета, она совершенно независима от самого факта наличия разума у считающего субъекта и даже от наличия самого субъекта. Тем не менее, материального воплощения числа не имеют — «количество», представленное в виде комбинации пальцев рук и ног, зарубок на палочке (вспомните, как Робинзон Крузо вел свой календарь), разложенных на земле веточек, костяшек на счетах или — что для нас самое главное! — черточек или значков на бумаге, есть всего лишь физическая модель некоего идеального абстрактного понятия «числа». Умение считать в уме, которое отличает цивилизованного человека от дикаря, и состоит в том, что мы можем оторваться от такой материальной модели и оперировать непосредственно с абстракцией.

Из понятия числа, как объективно существующей абстракции, вытекает, что его материальное представление может быть произвольным, лишь бы оно подчинялось тем же правилам, что и сами числа (совершенно аналогично булевым переменным). Проще всего считать палочками (и в детском саду нас учат именно такому счету) — в качестве которых могут выступать и пластмассовые стерженьки, и пальцы, и черточки на бумаге. Один — одна палочка, два — две палочки, десять — десять палочек. А сто палочек? Уже посчитать затруднительно, потому придумали сокращение записи: доходим до пяти палочек, ставим галочку, доходим до десяти — ставим крестик:

1	2	5	7	10	11
I	II	V	VII	X	XI

Узнаете? Конечно, это всем знакомая римская система, сохранившаяся до настоящих времен на циферблатах часов или в нумерации столетий. Она представляет собой пример *непозиционной* системы счисления — потому что значение определенного символа, *обозначающего* то или иное число, в ней не зависит от позиции относительно других символов — все значения в записи просто *суммируются*. То есть записи «XVIII» и «IIIXV» в принципе должны означать одно и то же. На самом деле это не совсем так — в современной традиции принято в целях сокращения записи использовать и позицию: скажем, в записи «IV» факт, что палочка стоит перед галочкой, а не после нее, означает придание ей отрицательного значения, т. е. в этом случае единица не прибавляется, а вычитается из пяти (то же самое относится и к записи девятки «IX»). Если вы человек наблюдательный, то могли заметить, что на часах четверку часто обозначают как «IIII», а не как «IV» (см. рис. 14.5), что, несомненно, более отвечает духу непозиционной системы. Однако при всех возможных отклонениях главным здесь остается факт, что в основе системы лежит операция *суммирования*.

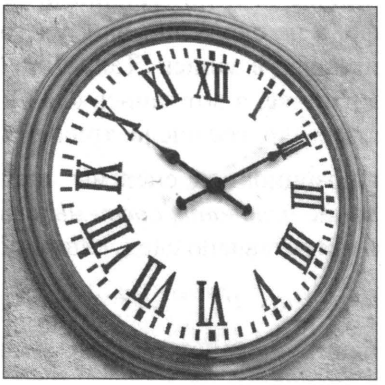


Рис. 14.5. Циферблат часов с римскими числами

Большие числа в римской системе записывать трудно, а еще сложнее осуществлять с ними арифметические действия. Поэтому еще в древнем Вавилоне придумали позиционную систему. Позднее в Европе позиционную систему переоткрыл (видимо) Архимед, затем от греков она была воспринята индусами и арабами, а на рубеже I и II тысячелетий опять попала в Европу[1] — с тех пор мы называем цифры арабскими, хотя по справедливости их следовало бы назвать индийскими. Это была уже современная десятичная система в том виде, в котором мы ее используем по сей день, у арабов отличается только написание цифр. С тем фактом, что заимствована она именно у арабов, связано не всеми осознаваемое несоответствие порядка записи цифр в числе с привычным для нас порядком следования текста: арабы, как известно, пишут справа налево. Поэтому значение цифры в зависимости от позиции ее в записи числа возрастает именно справа налево.

---

[1] Перевод соответствующего трактата арабского ученого Мухаммеда аль-Хорезми на латынь относится к 1120 году. От его имени произошло слово «алгоритм», а от выражения «ал-джабр» из названия трактата — слово «алгебра».

Заметки на полях

Еще один нюанс, дошедший до нас от древнегреческих времен, связан с тем, что греки и римляне не знали нуля. Именно поэтому первым годом нового тысячелетия считается 2001, а не 2000 год — год с двумя нулями относится к предыдущему столетию или тысячелетию. Это происходит потому, что после последнего года до нашей эры («минус первого») идет сразу первый год нашей эры, а не нулевой. На самом деле древние греки были совсем не такими дураками и ноль игнорировали не по скудоумию. Дело в том, что в последовательности объектов, нумерованных от нуля до, например, девяти, содержится не девять предметов, а десять! Чтобы избежать этой путаницы, в быту обычно нумерацию производят, начиная с 1, тогда последний *номер* будет одновременно означать и *количество*. В электронике же и в программировании обычно принято нумеровать объекты, начиная с 0, и всегда следует помнить, что номер и количество различаются на единицу (так байт, о котором далее, может содержать 256 возможных значений, но номер последнего значения равен 255). На всякий случай всегда следует уточнять, откуда ведется нумерация, иначе можно попасть в неприятную ситуацию (скажем, элементы строки в языке Pascal нумеруются с единицы, а в языке C — с нуля).

Позиционные системы, в отличие от непозиционных, основаны не на простом сложении входящих в них цифр, а на сложении их с учетом присвоенного им «веса» в зависимости от положения цифр в записи. Так, запись «3» и в римской системе, и в арабской означает одно и то же, а вот запись «33» в римской системе означала бы шесть, а в арабской — совсем другое число, тридцать три.

Для строгого определения позиционной системы сначала выбирается некоторое число p, которое носит название *основания системы счисления*. Тогда любое число в такой системе может быть представлено следующим образом:

$$a_n \cdot p^n + a_{n-1} \cdot p^{n-1} + \ldots + a_1 \cdot p^1 + a_0 \cdot p^0. \qquad (4)$$

В самой записи числа степени основания подразумеваются, а не пишутся (и для записи основания даже нет специального значка), поэтому запись будет представлять собой просто последовательность $a_n \ldots a_0$ (еще раз обратим внимание на то, что запись производится справа налево по старшинству, — обычная математическая запись выглядела бы наоборот). Отдельные позиции в записи числа называются *разрядами*. Например, в десятичной системе (т. е. в системе с основанием 10) полное представление четырехразрядного числа 1024 таково:

$$1 \cdot 10^3 + 0 \cdot 10^2 + 2 \cdot 10^1 + 4 \cdot 10^0.$$

Ну а как можно представить число в системе счисления с другим основанием? Для любой системы с основанием p нужно ровно p различных *цифр* — т. е. *значков* для изображения чисел. Для десятичной системы их десять — это и есть известные всем символы от 0 до 9. Заметим, что выбор начертания этих значков совершенно произволен — так, у арабов и по сей день 1 обозначается, как и у нас, вертикальной палочкой, а вот цифра 2 — знаком ٢, похожим на латинскую строчную «г».

Самые употребительные системы в настоящее время, кроме десятичной, связаны с электроникой и потому имеют непосредственное отношение к нашему повествованию. Это знаменитая двоичная система и менее известная широкой публике, но также очень распространенная шестнадцатеричная.

Двоичная и шестнадцатеричная системы

В двоичной системе необходимо всего два различных знака для цифр: 0 и 1. Это и вызвало столь большое ее распространение в электронике: смоделировать два состояния электронной схемы и затем их безошибочно различить неизмеримо проще, чем три, четыре и более, не говоря уж о десяти. В середине прошлого столетия советский инженер Николай Петрович Брусенцов построил вычислительную машину, которая работала в троичной системе, и потом всю свою долгую жизнь (он скончался в 2014 году, немного не дотянув до 90) доказывал ее неоспоримые преимущества. Но несмотря на это, его изобретение так и осталось почти единственным примером такого рода — слишком сложна реализация электронных элементов, работающих в троичной логике.

Еще важнее, что двоичная система прекрасно согласуется как с представленными ранее логическими переменными, так и с тем фактом, что величина, могущая принимать два и только два состояния и получившая названия *бит*, есть естественная единица количества информации. Это было установлено в 1948 году одновременно Клодом Шенноном и Норбертом Винером, отцом кибернетики, — меньше, чем один бит, информации не бывает. Разряды двоичных чисел (т. е. чисел, представленных в двоичной системе) также стали называть битами. Слово bit — по-английски означает «кусочек, частица чего-либо». Как термин для обозначения количества информации, слово «бит», говорят, возникло от сокращения **BI**nary digi**T** — «двоичная цифра».

Представление двоичных цифр с помощью уровней напряжения, как это делается в электронных устройствах, если точно такая же модель числа, как раскладывание на земле палочек и проведение черточек на бумаге. В последних случаях мы оперируем с числами вручную, по правилам арифметики, а в электронных схемах это происходит в автоматическом режиме, без участия человека — вот и вся разница! Это понятие о «модели числа» — очень важный момент, который следует хорошо осмыслить, если вы действительно хотите вникнуть в суть работы цифровых электронных схем.

Итак, запись числа в двоичной системе требует всего двух цифр, начертание которых заимствовано из десятичной системы и выглядит, как 0 и 1. Число, например, 1101 тогда будет выглядеть так:

$$1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 13.$$

Чтобы отличить запись числа в различных системах, часто внизу пишут основание системы:

$$1101_2 = 13_{10}.$$

Если система не указана, то имеется в виду обычно десятичная, но не всегда — часто, когда из контекста понятно, что идет речь об электронных устройствах, не указывают не только основание два, но и под словом «разрядность» имеют в виду количество именно двоичных, а не десятичных разрядов (таков, скажем, смысл термина «24-разрядный цвет»).

Шестнадцатеричная система имеет, как ясно из ее названия, основание шестнадцать. Для того чтобы получить шестнадцать различных знаков, изобретать ничего

нового не стали, а просто использовали те же цифры от 0 до 9 для первых десяти и заглавные латинские буквы от A до F для одиннадцатого-шестнадцатого знаков (часто вместо заглавных букв употребляют и строчные, с теми же значениями). Таким образом, известное нам число 13_{10} выразится в шестнадцатеричной системе как просто D_{16}. Соответствие шестнадцатеричных знаков десятичным числам следует выучить наизусть: A — 10, B — 11, C — 12, D — 13, E — 14, F — 15. Значения больших чисел вычисляются по обычной формуле, например:

$$A2FC_{16} = 10 \cdot 16^3 + 2 \cdot 16^2 + 15 \cdot 16^1 + 12 \cdot 16^0 = 40960 + 512 + 240 + 12 = 41724_{10}.$$

Перевод из одной системы счисления в другую

Как следует из изложенного, перевод в десятичную систему любых форматов не представляет сложности и при надлежащей тренировке может осуществляться даже в уме. Для того чтобы быстро переводить в десятичную систему двоичные и шестнадцатеричные числа, следует выучить наизусть таблицу степеней двойки до 16 (табл. 14.3) и представления некоторых чисел в двоичной и шестнадцатеричной формах (табл. 14.4).

Таблица 14.3. *Степени двойки в десятичном виде*

2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8
2	4	8	16	32	64	128	256
2^9	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}	2^{15}	2^{16}
512	1024	2048	4096	8192	16 384	32 768	65 536

Таблица 14.4. *Числа в десятичном, двоичном и шестнадцатеричном представлении*

Число	Десятичное (Dec)	Двоичное (Bin)	Шестнадцатеричное (Hex)
$2^1 - 1$	1	01	1h
2^1	2	10	2h
$2^2 - 1$	3	11	3h
2^2	4	100	4h
$2^3 - 1$	7	111	7h
2^3	8	1000	8h
$2^4 - 1$	15	1111	Fh
2^4	16	1 0000	10h
$2^5 - 1$	31	1 1111	1Fh
2^5	32	10 0000	20h
$2^6 - 1$	63	11 1111	3Fh
2^6	64	100 0000	40h

Таблица 14.4 (окончание)

Число	Десятичное (Dec)	Двоичное (Bin)	Шестнадцатеричное (Hex)
$2^7 - 1$	127	111 1111	7Fh
2^7	128	1000 0000	80h
$2^8 - 1$	255	1111 1111	FFh
2^8	256	1 0000 0000	100h
$2^9 - 1$	511	1 1111 1111	1FFh
2^9	512	10 0000 0000	200h
$2^{10} - 1$	1023	11 1111 1111	3FFh
2^{10}	1024	100 0000 0000	400h
$2^{11} - 1$	2047	111 1111 1111	7FFh
2^{11}	2048	1000 0000 0000	800h
$2^{12} - 1$	4095	1111 1111 1111	FFFh
2^{12}	4096	1 0000 0000 0000	1000h
$2^{13} - 1$	8191	1 1111 1111 1111	1FFFh
2^{13}	8192	10 0000 0000 0000	2000h
$2^{14} - 1$	16 383	11 1111 1111 1111	3FFFh
2^{14}	16 384	100 0000 0000 0000	4000h
$2^{15} - 1$	32 767	111 1111 1111 1111	7FFFh
2^{15}	32 768	1000 0000 0000 0000	8000h
$2^{16} - 1$	65 535	1111 1111 1111 1111	FFFFh
2^{16}	65 536	1 0000 0000 0000 0000	10000h

Буква *h* добавляется к шестнадцатеричному представлению числа, чтобы отличить его от десятичного, не используя индекса 16 (в разных языках программирования есть и другие способы, см. далее). На первое время достаточно запомнить только маленькие числа, а для образования двоичных и шестнадцатеричных чисел — понять принцип, остальное выучится позже само.

Из табл. 14.4 видно, что из двоичной в шестнадцатеричную системы и обратно переводить совсем просто. Двоичное число достаточно разбить на тетрады (т. е. группы из 4-х цифр) и перевести каждую в отдельности. Например, число 59_{10}, т. е. $0011\ 1011_2$, будет равно 3Bh. Еще проще обратный перевод — каждое шестнадцатеричное число заменяется группой из 4-х двоичных цифр. Соответственно, число $A2FC_{16}$ выразится так: $1010\ 0010\ 1111\ 1100_2$. Заметьте, что пробелы между тетрадами в двоичных числах введены просто для удобства восприятия, подобно пробелам между тройками разрядов (классами) в записи больших десятичных чисел, и никакой иной нагрузки не несут. При записи двоичных чисел в тексте программ, как ассемблерных, так и программ на языках высокого уровня, естественно, эти

пробелы ставить запрещается. Обычно при этом для удобства ведущие нули нередко не опускаются (хотя в табл. 14.4 они и опущены) — число 15 чаще всего записывается в виде 0Fh. Почему так удобнее, вы поймете далее.

Заметки на полях

Почему в табл. 14.4 приведены именно эти числа? Числа, на единицу меньшие степени двойки, имеют в электронике и в программировании большое значение. Если вы посмотрите в таблицу внимательно, то увидите, что наибольшее число с количеством разрядов, равным степени двойки, содержит единицы во всех разрядах, т. е. как раз и равно этой степени минус единица. При этом количество всех чисел с таким количеством разрядов равно степени двойки. Другими словами, память, содержащая ровно 256 (2^8) ячеек, будет иметь номер (адрес) последней ячейки, равный 255, или FFh. Наибольшее число, до которого может досчитать 8-разрядный двоичный счетчик, также равно FFh, поскольку если подать на него еще один импульс (257-й по счету), он обнулится, ибо 9-го разряда, где могла бы записаться старшая единица, у него не существует. Потому с такими числами приходится иметь дело очень часто, даже чаще, чем с собственно степенями двойки.

Сложнее переводить из десятичной системы, и для этого в учебниках рекомендуется устрашающая процедура, основанная на делении столбиком (см., например [17]). Я сейчас попробую вам показать модификацию этого способа, который позволяет переводить числа в двоичную систему несколько более простым методом, причем небольшие числа можно переводить даже в уме.

Для начала запомним, что разряды двоичного числа нумеруются с нуля — т. е., к примеру, разряд номер 3 окажется четвертым справа. Теперь пусть мы имеем, например, десятичное число 59. Подбираем наибольшую степень двойки из табл. 14.3, не превышающую этого числа: 32, что есть 5-я степень. Ставим 1 в 5-м разряде: 100000. Вычитаем подобранную степень из исходного числа ($59 - 32 = 27$) и подбираем для остатка также степень, его не превышающую: 16 (2^4). Ставим единицу в 4-м разряде: 110000. Повторяем процедуру вычитания-подбора: $27 - 16 = 11$, степень равна 8 (2^3), ставим единицу в 3-м разряде: 111000. Еще раз: $11 - 8 = 3$, степень равна 2 (2^1), так что 2-й разряд оказался пропущен, ставим в нем ноль, а единицу в 1-м разряде: 111010. Последнее вычитание дает 1, которую и ставим в младший (нулевой) разряд, окончательно получив $59_{10} = 111011_2$.

Байты

Слово «байт» (byte) представляет собой сокращение от **B**inar**Y** digi**T** **E**ight — «восемь двоичных цифр». Другими словами, *байт* — это просто восьмиразрядное двоичное число. Соответственно, он имеет ровно два шестнадцатеричных разряда, или две двоичных тетрады. Такой байт был введен фирмой IBM в конце 50-х годов прошлого века, до этого (а в СССР — и после этого, вплоть до 1969 года) применялись байты с другим количеством разрядов: 5, 6 и 7.

Почему именно 8 разрядов? Да просто потому, что так удобно — число кратно степени двойки, т. е. легко масштабируется. Скажем, шестнадцатиразрядное число — просто два байта, записанные подряд, подобно тетрадам в самом байте. В то же время число в 8 разрядов — относительно невелико и одновременно достаточно

емко: имеет 256 значений, которых с лихвой хватает, к примеру, для представления всех печатных знаков европейских алфавитов.

Поэтому в настоящее время байт — общепринятая единица измерения информации, т. к. основную единицу — бит — на практике применять неудобно из-за ее мелковатости, числа получаются слишком длинными. Применяют и меньшие единицы: кроме бита, это полубайт, или тетрада — одно шестнадцатеричное число. Для современной электронной техники характерны бóльшие, чем байт, числа — двухбайтовые (65 536 значений), четырехбайтовые (32 двоичных разряда или 4 294 967 296 значений) и даже восьмибайтовые числа. Все они часто называются словами: «word». Так, в памяти современных персональных компьютеров минимальная ячейка памяти содержит 32 или 64 бита (4 или 8 байтов), хотя измеряют объем памяти все равно в традиционных байтах.

О ЕДИНИЦАХ ИНФОРМАЦИИ

В однобуквенных сокращениях принято обозначать байт большой буквой (Б), чтобы отличить его от бита (б), но в критичных случаях во избежание разночтений следует писать полностью: «Мбайт», «Мбит». Тем более что отечественный ГОСТ 8.417-2002 предусматривает однобуквенное сокращение для байта (Б), но не предусматривает его для бита. С приставками также имеется некоторая путаница. В 1999 году Международная электротехническая комиссия (МЭК) с большим опозданием попыталась устранить неоднозначность в обозначениях кратности, введя специальные двоичные приставки киби (вместо «кило»), меби (вместо «мега») и гиби (вместо «гига»), означающие умножение на 1024 вместо 1000. Однако «килобайты» и «мегабайты» к тому времени настолько прижились, что эти не очень удачно звучащие обозначения так и не стали общепринятыми. Приставку «кило» в единицах информации, согласно тому же отечественному стандарту, следует писать с большой буквы (Кбайт), чтобы подчеркнуть, что речь идет об умножении на 1024, а не на 1000. Однако для приставок «мега» и «гига» (а также всех остальных) такого удобного приема уже нет, поэтому это правило на практике соблюдают редко (в этой книге я его также не придерживаюсь). На практике еще можно встретить обозначение единиц информации из одной буквы (например, «256 К памяти»), но пользоваться таким способом не следует, т. к. часто приходится гадать — идет ли тут речь о килобитах или килобайтах. Кстати, соответствующее прилагательное по правилам пишется, как «битовый» или «байтовый» (а не «битный» или «байтный»), хотя в технической литературе вы чаще встретите выражение «восьмибитный контроллер» или «двадцатичетырехбитный цвет», в то время, как «код» или «регистр» скорее будет «восьмибитовым», по правилам.

Впрочем, есть одна область, где традиционно употребляются именно биты (а также мегабиты и гигабиты), а не байты, — это характеристики последовательных цифровых линий передач. К примеру, характеристика модемов в 56 К означает именно 56 Кбит в секунду, а скорости передачи в так называемых *широкополосных линиях связи* измеряются в мегабитах в секунду. Это связано с тем, что передача именно восьмиразрядными пакетами битов там применяется редко — скажем, стандарт RS232, который мы еще будем рассматривать в этой книге, предусматривает восемь значащих разрядов (т. е. один байт), но помимо этого передаются, как минимум, еще два бита: стоповый и стартовый, итого 10. В компьютерных сетях пакеты и вовсе могут иметь переменную длину, поэтому байтами в сетях информацию считают только тогда, когда речь идет об отправленной или принятой информации, но не о той, которая реально передается, — последняя может включать в себя достаточно много всяких служебных битов и целых байтов.

Кстати, часто употребляющийся термин «боды» не равносилен битам в секунду, как это часто ошибочно считают, бездумно записывая фразы вроде «скорость модема 56 бод». Бод — это количество посылок в секунду, где одна посылка может нести несколько битов. Так, базовая для телефонных модемов скорость передачи составляет 2400 бод, но при этом она может быть равна 4800 или 9600 бит/с, в зависимости от того, сколько битов содержится в посылке. Кроме того, скорость, выраженная в бодах, всегда представляет собой полную скорость канала, включающую все служебные поля и отдельные дополнительные биты (вроде бита четности), и потому количество реально передаваемой информации представляет плохо и интересна только связистам для каких-то своих надобностей. По этим причинам единицей «бод» в повседневной практике лучше не пользоваться.

Запись чисел в различных форматах

Шестнадцатеричный формат записи часто еще обозначают как HEX (hexadecimal), двоичный — как BIN (binary), а десятичный — как DEC (decimal). Кроме этого, в ходу еще так называемый *двоично-десятичный формат — BCD* (binary-coded decimal), о котором далее. Дело в том, что с помощью матричных шрифтов на компьютерах с текстовыми дисплеями воспроизводить индексы было невозможно, и вместо того, чтобы обозначать основания системы цифрой справа внизу, их стали обозначать буквами: «B» (или «b») — означает двоичную систему, «H» (или «h») — шестнадцатеричную, отсутствие буквы означает десятичную систему:

$$13 = 00001101b = 0Dh.$$

Такая запись принята в языке ассемблера для процессоров Intel и одно время была общепринятой. Популярность языка C внесла в это дело некоторый разнобой: там десятичная система не обозначается никак, двоичная — буквой «b» (см. далее), а вот шестнадцатеричная — буквой «x», причем запись во всех случаях предваряется нулем (чтобы не путать запись числа с идентификаторами переменных, которые всегда начинаются с буквы):

$$13 = 0b00001101 = 0x0D.$$

Такая запись также принята в ассемблере для микроконтроллеров Atmel AVR, с которым мы будем знакомиться далее (см. *главу 19*). Следует учитывать, что стандартами языков C и C++ запись констант в двоичном виде не поддерживается, потому с этим вопросом существует большая путаница в языках на их основе. Созданный на основе C/C++ язык Arduino, которым мы также будем пользоваться (см. *главу 20* и далее), унаследовал от AVR-ассемблера поддержку такой записи, но, формально говоря, в нем определена другая форма представления двоичных чисел: с заглавной латинской B перед числом (B00001101). Так что в Arduino можно использовать обе этих формы. Нужно только учитывать, что во избежание проблем не следует записывать в двоичном виде числа более одного байта — так, AVR-ассемблер поймет запись 0b100110011 (т. е. 307_{10}), а вот компилятор Arduino на записи B100110011 выдаст ошибку и мы точно не знаем, всегда ли он будет правильно интерпретировать запись в первой форме.

Что касается HEX-чисел, то запись 0Dh ассемблер AVR «не понимает», зато «понимает» представление HEX-формата, принятое в языке Pascal: $0D. Обратите

внимание, что запись в HEX-формате обычно ведется в двухразрядном виде — даже если число имеет всего один значащий разряд, как в нашем случае, то в старшем пишется 0. То же самое относится и к двоичной записи, которая дополняется нулями до восьми разрядов. Таким образом подчеркивается, что речь идет о восьмиразрядных устройствах. А вот для десятичного представления ведущих нулей следует остерегаться — чтобы еще больше запутать пользователя, разработчики AVR-ассемблера вслед за создателями языка C приняли для представления редко употребляемых восьмеричных чисел запись просто с ведущим нулем, без букв: например, 77 означает просто десятичное 77, а вот 077 будет означать $7 \cdot 8 + 7 = 63_{10}$.

Несколько слов о двоично-десятичном формате BCD (подробнее о нем см. в *главе 19*). Электронные устройства «заточены» под использование двоичных и родственных им систем счисления, потому что основой являются два состояния, т. е. двоичная цифра. Так что, соединив несколько устройств вместе с целью оперирования с многоразрядными числами, мы всегда будем получать именно двоичное число, и оперировать с ним оказывается значительно проще. Но для понимания человеком десятичный формат необходим — в десятичном виде числа приходится представлять всегда, когда речь идет о выводе, например, на цифровой дисплей. Для этой цели приходится преобразовывать двоичные/шестнадцатеричные числа в десятичные и хранить их в таких же байтовых регистрах или ячейках памяти.

Это можно делать двумя путями: в виде упакованного и неупакованного BCD. *Неупакованный* формат попросту означает, что мы тратим на каждую десятичную цифру не тетраду, как минимально необходимо, а целый байт. Зато при этом не возникает разночтений: $05h = 05_{10}$, и никаких проблем. Однако ясно, что это крайне неэкономично — байтов требуется в два раза больше, а старший полубайт при этом все равно всегда ноль. Потому BCD-числа при хранении и передаче по каналам связи всегда *упаковывают*, занимая и старший разряд второй десятичной цифрой, — скажем, число 59 при этом и запишется, как просто 59h. Однако 59h — это не 59! Ведь мы раньше установили, что записи 59h соответствует $5 \cdot 16 + 9 = 89$, что вообще ни в какие ворота не лезет.

Причина в том, что двоично-десятичная запись числа не совпадает с шестнадцатеричной. Поэтому в общем случае перед проведением операций с упакованными BCD-числами их распаковывают, перемещая старший разряд в отдельный байт и заменяя в обоих байтах старшие полубайты нулями. Иногда для проведения операций с BCD-числами в микропроцессоре или микроконтроллере предусмотрены специальные команды, так что «вручную» заниматься упаковкой-распаковкой не требуется. В качестве примера хранения чисел в упакованном BCD-формате можно привести значения часов, минут и секунд в микросхемах энергонезависимых часов RTC (о них см. *главу 21*).

Немного двоичной арифметики

Правила двоичной арифметики значительно проще, чем десятичной, и включают две таблицы: сложения и умножения — несколько похожие на те же таблицы для логических переменных:

$0 + 0 = 0$ $\quad$ $0 \cdot 0 = 0$

$0 + 1 = 1$ $\quad$ $0 \cdot 1 = 0$

$1 + 0 = 1$ $\quad$ $1 \cdot 0 = 0$

$1 + 1 = 10$ $\quad$ $1 \cdot 1 = 1$

Как мы видим, правила обычного умножения одноразрядных двоичных величин совпадают с таковыми для логического умножения. Однако правила сложения отличаются, поскольку при сложении двух единиц результат равен 2, и появляется перенос в следующий разряд. Учитывая, что умножение многоразрядных чисел сводится к сложению отдельных произведений, там придется этот перенос учитывать (как это делается на практике, мы увидим в *главе 15*).

Сложности начинаются, когда мы хотим в двоичной системе представить отрицательные и дробные числа, причем не выходя за рамки двух знаков «ноль» и «единица», ибо в электронной схеме другие знаки представлять нечем.

Отрицательные двоичные числа

Самый простой метод представления отрицательных чисел — отвести один бит (логичнее всего — старший) для хранения знака. По причинам, которые вы поймете далее, значение 1 в этом бите означает знак «минус», а 0 — знак «плюс». Что произойдет с нашим числом при таком представлении?

В области положительных чисел не произойдет ничего, кроме того, что их диапазон сократится вдвое, — например, для числа в байтовом представлении вместо диапазона 0...255 мы получим всего лишь 0...127 (0000 0000–0111 1111). А отрицательные числа будут иметь тот же диапазон, только старший бит у них будет равен 1. Все просто, не правда ли?

Нет, неправда. Такое представление отрицательных чисел совершенно не соответствует обычной числовой оси, на которой влево от нуля идет минус единица, а затем числа по абсолютной величине увеличиваются. Здесь же мы получаем, во-первых, два разных нуля («обычный» 0000 0000 и «отрицательный» 1000 0000), во-вторых, оси отрицательных и положительных чисел никак не стыкуются, и производство арифметических операций превратится в головоломку.

Поэтому поступим так: договоримся, что −1 соответствует число 255 (1111 1111), −2 — число 254 (1111 1110) и т. д. вниз до 128 (1000 0000), которое будет соответствовать −128 (и общий диапазон всех чисел получится от −128 до 127). Очевидно, что если вы при таком представлении хотите получить отрицательное число в обычном виде, то надо из значения числа (например, 240) вычесть максимальное значение диапазона (255) плюс 1 (256). Если отбросить знак, то результат такого вычитания (16 в нашем случае) называется еще *дополнением до 2* для исходного числа (а само исходное число 240 — *дополнением до 2* для 16). Название «дополнение до 2» используется независимо от разрядности числа, потому что верхней границей всегда служит степень двойки (в десятичной системе аналогичная операция называется «дополнение до 10»).

Что произойдет в такой системе, если вычесть, например, 2 из 1? Запишем это действие в двоичной системе обычным столбиком:

```
 00000001
–00000010
```

В первом разряде результата мы без проблем получаем 1, а уже для второго нам придется занимать 1 из старших, которые сплошь нули, поэтому представим себе, что у нас будто бы есть девятый разряд, равный 1, из которого заем в конечном итоге и происходит:

```
(1)00000001
 –00000010
 ----------
  11111111
```

На самом деле девятиразрядное число 1 0000 0000 есть не что иное, как 256, т. е. то же самое максимальное значение плюс 1, и мы здесь выполнили две операции: прибавили к уменьшаемому эти самые 256, а затем выполнили вычитание, но уже в положительной области для всех участвующих чисел.

А что результат? Он будет равен 255, т. е. тому самому числу, которое, как мы договорились, и представляет –1. Получается, что вычитание в такой системе происходит автоматически правильно, независимо от знака участвующих чисел. Если хотите, можете потренироваться и проверить, скажем, что будет, если в этой системе вычесть 240 из 100.

Немного смущает только эта самая операция нахождения дополнения до 2, точнее, в рассматриваемом случае, до 256 — как ее осуществить на практике, если схема всего имеет 8 разрядов? В дальнейшем мы увидим, что иногда ее осуществлять вовсе не надо — некоторые электронные схемы ведут себя так, что при осуществлении вычитания вся процедура осуществляется автоматически. Особенно наглядно это выглядит для двоичных реверсивных счетчиков, которые мы будем рассматривать в *главе 16*. В точности так же ведут себя и соответствующие команды в микропроцессорах — и если вы захотите произвести операцию вычитания числа 2 из содержимого восьмибитового регистра, содержащего число 1, то в регистре окажется число 255 (все единицы). А интерпретация результата — как отрицательного числа или как положительного — это уже ваши трудности.

В микропроцессорах есть обычно и команда, которая возвращает дополнение до 2, в большинстве ассемблеров она называется NEG, от слова «негативный», потому что меняет знак, если мы договариваемся считать числа «со знаком». А как ее можно было бы осуществить «вручную», не обращаясь в действительности к 9-му разряду? Вернемся к рассмотренным ранее примерам и выпишем столбиком исходные числа, результаты операции нахождения дополнения до 2 и результат еще одной манипуляции, которая представляет собой вычитание единицы из дополнения до 2, т. е., что то же самое, просто вычитания исходного числа из наивысшего числа диапазона (255):

2	0000 0010
256 – 2 = 254	1111 1110
255 – 2 = 253	1111 1101

Если мы сравним двоичные представления в верхней и нижней строках, то увидим, что они могут быть получены друг из друга путем инверсии каждого из битов. Эта операция называется нахождением *дополнения до 1* (потому что число, из которого вычитается, содержит все 1 во всех разрядах; для десятичной системы аналогичная операция называется *дополнение до 9*). Для нахождения дополнения до 1 девятый разряд не требуется, да и схему можно построить так, чтобы никаких вычитаний не производить, а просто переворачивать биты. То есть, для полного сведения вычитания к сложению надо проделать три операции:

1. Найти дополнение до 1 для вычитаемого (инвертировать его биты).
2. Прибавить к результату 1, чтобы найти дополнение до 2.
3. Сложить уменьшаемое и дополнение до 2 для вычитаемого.

Заметим, что все сложности с этими многочисленными дополнениями связаны с наличием нуля в ряду натуральных чисел — если бы его не было, дополнение было бы всего одно, и операция вычитания упростилась. Так может, греки все же были в чем-то правы?

В заключение обратим внимание на еще одно замечательное свойство двоичных чисел, которое часто позволяет значительно облегчить операции умножения и деления, а именно: умножению на 2 соответствует операция сдвига всех разрядов числа на один разряд влево, а операции деления на 2 — вправо. Крайние разряды (старший при умножении и младший при делении) в общем случае при этом должны теряться, но в микропроцессорах есть специальный бит переноса, в который эти «потерянные» разряды помещаются. Противоположные крайние разряды (младший при умножении и старший при делении) в общем случае замещаются нулями, но могут и замещаться значением бита переноса, что позволяет без лишних проблем делить и умножать числа с разрядностью больше одного байта. Как можно догадаться, умножению и делению на более высокие степени двойки будет соответствовать операция сдвига в нужную сторону на иное (равное степени) число разрядов.

Излишне говорить, что операцию сдвига разрядов в электронных схемах производить неизмеримо проще, чем операции деления и умножения. Есть и специальные схемы для этой операции — *сдвиговые регистры*, которые мы также будем «проходить» (в *главе 16*).

Дробные числа

Сразу заметим, что в некомпьютерной электронике дробными числами стараются не пользоваться. При необходимости их переводят в целые, умножая на соответствующую степень десяти (а чаще — даже на степень 2, что проще), при этом все остальные участвующие в расчетах величины также масштабируются в нужное число раз. Затем при выводе, к примеру, на цифровой дисплей, запятая просто устанавливается в нужном месте (иногда заранее, и без возможности изменения ее положения). То есть, для цифровой схемы не существует значения температуры, равного 30,81 градуса, а есть число 3081 в BCD-формате. Примерно те же действия нам придется произвести, когда мы станем конструировать цифровой термометр в *гла-*

ве 17, — на самом деле он будет показывать целое число милливольт в нужном масштабе.

И все же — как мы можем при необходимости представить дробные числа, если двоичные разряды ничего о таковых «не знают» и могут воспроизводить только целые числа в соответствии с формулой (4)? Мы не будем рассматривать расширение этой формулы, включающее в себя представление в позиционной системе не только целых, но и всех действительных чисел «с плавающей запятой», т. к. в электронике такой вариант хождения не имеет. В электронике и компьютерной технике используют другой способ представления действительных чисел — с помощью мантиссы и порядка, в так называемом *нормализованном виде*. При этом место запятой фиксируется так, чтобы перед запятой всегда был один ненулевой десятичный разряд (т. е. число от 1 до 9):

$$0{,}0125 = 1{,}25 \cdot 10^{-2},$$

$$1254{,}81 = 1{,}25481 \cdot 10^{3}.$$

Разумеется, в электронных схемах все это лучше делать в двоичной форме, записывая порядок, как степень двух (скажем, операция выравнивания порядков при сложении таких чисел сведется просто к сдвигу мантиссы, как мы видели ранее).

Легко заметить, что и саму запятую, и основание степени тут можно опустить, записывая в память лишь мантиссу и порядок, — конечно, если всегда помнить, что мы имеем в виду. Например, можно сделать так: отвести в памяти четыре байта, из которых первые три байта хранят цифры и знак мантиссы, а четвертый отведен под порядок. Если старший бит порядка равен единице, то порядок отрицательный, а если нулю — положительный, итого двоичный порядок составит диапазон от –128 до +127, а весь диапазон чисел в десятичном виде составит от примерно $-3{,}4 \cdot 10^{38}$ до $1{,}7 \cdot 10^{38}$. В реальных языках программирования за счет манипуляций с битами диапазон делают симметричным, т. е. верхняя граница равна нижней и составляет также $3{,}4 \cdot 10^{38}$. Наименьшее число при таком подходе будет равно не нулю, а 2^{-24}, т. е. около $5{,}9 \cdot 10^{-8}$. Число значащих десятичных цифр будет переменным и составляет 7–8 десятичных знаков.

В обычных языках программирования такие числа называют *числами ординарной точности* (в языке Pascal это тип single) и давно уже не употребляют, т. к. все персоналки, как минимум, 32-разрядные. Но для 8-разрядных контроллеров они сохранились — именно так представляются действительные числа в компиляторе AVRGCC, на котором основана среда программирования Arduino (см. *главу 20*). Что же касается языка ассемблера, то там стараются действительных чисел избегать. На практике ассемблерные операции с дробными числами можно производить проще, переводя их в целый диапазон (см. *главу 19*).

Коды, шифры и дешифраторы

По необходимости кратко коснемся темы кодирования и связанной с ней темы шифрования. Слово «код» (особенно в сочетании *двоичный код*) вы будете встречать очень часто, и следует понимать, что именно подразумевается под этим поня-

тием в том или ином случае. Кроме того, сразу забежим немного вперед и покажем, как обращаться с реальными схемами на этом примере.

Код в общем случае — это совокупность правил (т. е. алгоритм) для представления информации в какой-либо форме. Процесс применения к информационному сообщению этих правил называется *кодированием*, при этом полученная группа знаков или сигналов также называется *кодом*. Обратная операция — восстановление сообщения по известному коду — носит название *декодирования*. Например, код Морзе позволяет записать с помощью двух знаков — точки и тире — любую букву или цифру. Закодированное таким образом сообщение можно передавать с высокой надежностью в различных средах с высоким уровнем помех (в виде звуковых сигналов, в том числе по радио, в виде вспышек света разной длительности, перестукиванием через стены и т. п.).

Понятие кода применимо к формам представления информации лишь в тех случаях, когда между содержанием сообщения и его представлением в какой-либо форме существует взаимно-однозначное соответствие, т. е. по закодированной записи смысл сообщения может быть восстановлен единственным образом (если не считать возможных искажений при передаче). В этом смысле понятие кода может быть лишь ограниченно применимо, например, к письменной или устной речи, для которых характерна многозначность смысловых единиц. В гуманитарных дисциплинах, как правило, понятие кода используется в качестве метафоры (например, «смысловой код произведения искусства»).

Понятие кода нашло наиболее широкое применение для представления информации в цифровой форме. Самым распространенным цифровым кодом является двоичный код (т. е. представление любого знака или числа в виде набора двоичных цифр 0 и 1). Двоичный код, в свою очередь, служит основой для различных кодов, представляющих конкретные разновидности информации. В информатике часто также говорят про *исходный код программ*, что означает текст программы, записанный на одном из языков программирования.

На рис. 14.6 приведены различные стандартные коды для первых девяти букв латинского алфавита. Следует отметить, что азбука Брайля — система письменности для слепых (где жирная точка соответствует наличию выпуклости, а «худая» — ее отсутствию) — в полной мере является двоичным кодом, мало того, из принципа ее построения было многое заимствовано в современных системах компьютерных кодировок. А вот код Морзе, хотя и состоит из точек и тире, двоичным кодом не является: в нем используется как минимум еще один знак — пауза. Зато код Морзе намного экономичнее обычных двоичных кодов, таких, как ASCII, поскольку имеет переменное число точек-тире для каждого символа, и часто встречающиеся буквы в нем короче, чем редко встречающиеся.

Понятие шифра, строго говоря, к электронике не относится. Шифр — это просто код, построенный специальным образом для обеспечения секретности при передаче сообщений, и занимается этим довольно сложная математическая дисциплина — криптография. Тем не менее, в электронике нередко употребляют наименование «дешифраторы» — для обозначения элементов, которые предназначены для преобразования одной разновидности кода в другую.

Буква	Азбука Морзе		Азбука Брайля	Коды ASCII	
	точка-тире	в двоичн. записи		прописная	строчная
A	•▬	101100	⠁	0100 0001 (65)	0110 0001 (97)
B	▬•••	1101010100	⠃	0100 0010 (66)	0110 0010 (98)
C	▬•▬•	11010110100	⠉	0100 0011 (67)	0110 0011 (99)
D	▬••	11010100	⠙	0100 0100 (68)	0110 0100 (100)
E	•	100	⠑	0100 0101 (69)	0110 0101 (101)
F	••▬•	1010110100	⠋	0100 0110 (70)	0110 0110 (102)
G	▬▬•	110110100	⠛	0100 0111 (71)	0110 0111 (103)
H	••••	101010100	⠓	0100 1000 (72)	0110 1000 (104)
I	••	10100	⠊	0100 1001 (73)	0110 1001 (105)

Рис. 14.6. Некоторые коды первых латинских букв.
Для кодировки ASCII в скобках дано десятичное значение

Типичным примером дешифратора может служить микросхема К561ИД5 (рис. 14.7, *а*), которая преобразует двоично-десятичное число, подаваемое на входы *X*, в специальный код для управления семисегментными индикаторами. Сам такой индикатор вместе с таблицей его состояний представлен на рис. 14.8. В этой таблице в колонке под заголовком «BIN» представлены двоичные числа, соответствующие входам *X* дешифраторов на рис. 14.7.

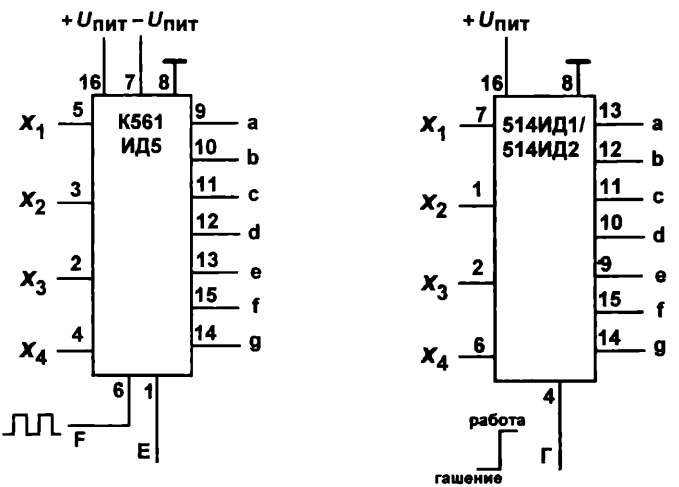
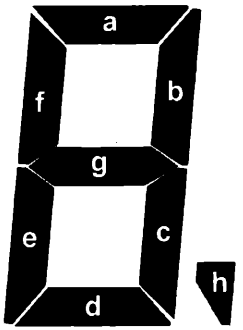


Рис. 14.7. Разводка выводов дешифраторов К561ИД5 (*а*) и 514ИД1/514ИД2 (*б*)



DEC	BIN	a	b	c	d	e	f	g
0	0000	1	1	1	1	1	1	0
1	0001	0	1	1	0	0	0	0
2	0010	1	1	0	1	1	0	1
3	0011	1	1	1	1	0	0	1
4	0100	0	1	1	0	0	1	1
5	0101	1	0	1	1	0	1	1
6	0110	1	0	1	1	1	1	1
7	0111	1	1	1	0	0	0	0
8	1000	1	1	1	1	1	1	1
9	1001	1	1	1	1	0	1	1

Рис. 14.8. Обозначения сегментов и таблица состояний семисегментного индикатора

Отметим, что К561ИД5 «заточена» под управление ЖК-дисплеями, и потому имеет двухполярное питание и содержит отдельный вход F, на который подаются прямоугольные импульсы частотой несколько десятков герц (иначе, как мы знаем из *главы 7*, сегмент ЖК-индикатора будет засвечен навсегда). Сигнал на входе E «защелкивает» выход — если на нем логический ноль (напряжение «земли»), то коды на выходе не будут меняться независимо от состояния входов, в нормальном состоянии там должна быть логическая единица (напряжение питания).

Если нужно управлять светодиодными индикаторами, для которых двухполярное пульсирующее питание не требуется, то выходы отрицательного питания и «земли» объединяют и на вход F подают напряжение «земли» (логического нуля). По выходам a–f при этом приходится ставить дополнительные ключевые транзисторы или эмиттерные повторители для управления сегментами индикаторов, т. к. выходной ток микросхем серии 561 для непосредственного управления светодиодами недостаточен.

Микросхема К561ИД5 основана на так называемой *технологии КМОП*. А специально предназначенные для управления светодиодными индикаторами дешифраторы 514ИД1 и 514ИД2 (рис. 14.7, *б*) основаны на технологии ТТЛ и потому ограничены одним напряжением питания (максимум 5,5 В), зато выходы у них намного мощнее (подробнее о технологиях логических микросхем и их электрических характеристиках мы узнаем из *главы 15*).

Различаются 514ИД1 и 514ИД2 полярностью выхода — у первого наружу выведен эмиттер выходного транзистора, коммутирующий ток через сегмент на «землю», а у второго — на выходе открытый коллектор. Таким образом, 514ИД1 предназначена для управления сегментными индикаторами с общим катодом, который подсоединяется к «земле», а 514ИД2 — индикаторами с общим анодом, который подсоединяется к питанию. Другое отличие — у 514ИД1 имеются встроенные токоограничивающие резисторы номиналом приблизительно 1 кОм, т. е. при падении напряжения на сегменте примерно 1,8–2 В (стандартном для красного светодиода) выходной ток будет составлять около 3 мА.

У 514ИД2 таких резисторов не имеется, и их приходится ставить дополнительно, непосредственно к выходу дешифратора индикатор присоединять нельзя. Зато здесь можно гибко управлять яркостью свечения в зависимости от индивидуальных характеристик индикаторов — зеленые и желтые индикаторы при одинаковом токе

имеют большее падение напряжения, чем красные (номинально 2,4 В против 1,8 В), и могут светиться тусклее. При токе через сегмент около 5–6 мА сопротивление токоограничивающего резистора должно быть в пределах 360–510 Ом. Большими по размеру индикаторами, где прямое падение напряжения на сегменте удвоенное, с помощью этих микросхем приходится управлять также через внешние ключи, подключенные к более высокому напряжению.

При подаче напряжения «земли» на вывод «Г» этих микросхем все сегменты оказываются погашенными. Это позволяет осуществлять динамическую индикацию — управление многоразрядными индикаторами, при котором в каждый момент времени светится только один десятичный разряд. В результате удается сэкономить на количестве соединений (одноименные сегменты всех разрядов соединяются параллельно) и отчасти на потреблении схемы. Формально говоря, ток через каждый разряд при таком подключении должен увеличиваться пропорционально числу разрядов, чтобы сохранить ту же яркость, — например, при четырех разрядах ток должен быть в четыре раза больше, т. к. каждый разряд горит лишь четвертую часть времени (но не забудьте, что у 514ИД2 есть предельное значение — 22 мА). Однако опыт показывает, что пропорциональное увеличение тока необязательно, визуально яркость сохраняется и при меньших значениях мгновенного тока через сегмент, что и позволяет немного сэкономить на питании. О динамической индикации мы еще будем говорить при изучении микроконтроллеров, с помощью которых организовать ее гораздо проще и удобнее, чем с помощью обычных микросхем.

Код Грея

Код Грея (полное название — рефлексивный двоичный код Грея) заменяет представление чисел в двоичной форме на такое, в котором два соседних значения различаются только в одном разряде. Придуман он связистом Фрэнком Греем с целью избежать неоднозначности при дешифровке состояния нескольких переключателей. Например, если обычный код меняется из состояния 3 (011) в 4 (100), то из-за ошибки считывателя можно прочесть старший бит из старого состояния, а два последующих — из нового, и получить число 000, что очень далеко от истины. Еще хуже ситуация при круговом счете, когда происходит переход от единиц во всех разрядах во все нули, — в этом случае можно получить вообще любое число. Код Грея гарантирует, что этого не произойдет ни при каких условиях.

Четырехзначный код Грея в сравнении с двоичным представлен в табл. 14.5.

Таблица 14.5. *Четырехзначный двоичный код и код Грея*

число	0	1	2	3	4	5	6	7
дв. код	0000	0001	0010	0011	0100	0101	0110	0111
код Грея	0000	0001	0011	0010	0110	0111	0101	0100
число	8	9	10	11	12	13	14	15
дв. код	1000	1001	1010	1011	1100	1101	1110	1111
код Грея	1100	1101	1111	1110	1010	1011	1001	1000

Из таблицы легко понять, как формируются коды Грея с другим количеством разрядов. Обратим внимание, что младшие три бита в нижней строке соответствуют младшим трем битам в верхней строке, расположенным в обратном порядке по старшинству чисел (от этого в названии слово «рефлексивный», что значит «отраженный»). Поэтому для кода, например, с тремя разрядами достаточно взять три младших бита из любой строки таблицы (обычно берут из верхней, где возрастание более-менее соответствует естественному порядку). А для большего числа разрядов тогда сформировать код Грея можно следующим способом: прибавив к указанным значениям в старшем разряде 0, получим первые 16 разрядов пятизначного кода Грея. Для получения второй половины пятизначного кода нужно расположить указанные в таблице четырехзначные значения в обратном порядке, а потом прибавить старший разряд, равный 1.

Код Грея очень просто получается из двоичного путем побитовой операции «Исключающее ИЛИ» с тем же числом, сдвинутым вправо на один бит, причем старший бит должен замещаться нулем, а не младшим битом (нециклический сдвиг). На языке С эта операция будет выглядеть так:

```
unsigned int grayencode(unsigned int g)
{
    return g ^ (g >> 1);
}
```

Обратное преобразование сложнее: необходим последовательный сдвиг вправо и суммирование исходного двоичного числа до тех пор, пока очередной сдвиг не обнулит слагаемое. На языке С эту операцию можно представить в таком виде:

```
unsigned int graydecode(unsigned int gray)
{
    unsigned int bin;
    for (bin = 0; gray; gray >>= 1) {
      bin ^= gray;
    }
    return bin;
}
```

Особенность кода Грея состоит в том, что он не теряет своих свойств при любой перестановке битов, естественно, если это делать во всей таблице кодов одновременно. В том числе можно прочитать биты наоборот, поменяв старшие с младшими. Но, конечно, при таких перестановках соответствие естественному порядку чисел нарушается, и алгоритмы перевода в обычное число придется видоизменять.

Код Грея на практике применяют в случаях, когда фронты импульсов нечеткие, а подача синхроимпульсов для фиксации момента чтения невозможна, откуда возникает неоднозначность при определении двоичного уровня: то ли ноль, то ли единица. Наиболее часто эта задача встречается в определении положения вращающихся конструкций — например, в старых механических мышах, как и в колесике прокрутки новых оптических, используется специальный вариант кода Грея.

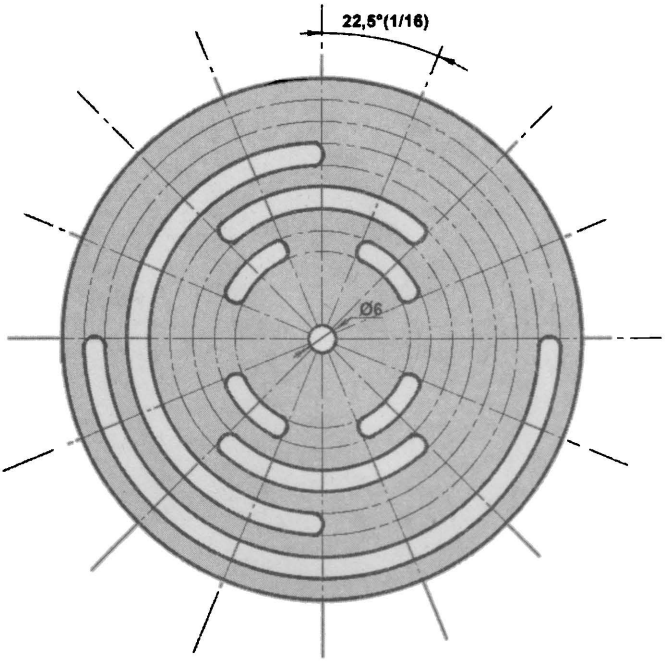
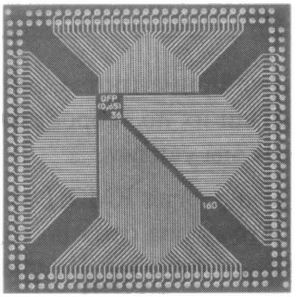


Рис. 14.9. Кодирующий диск с 4-битным кодом Грея

Пример кодирующего диска с четырехзначным кодом Грея показан на рис. 14.9. Он позволяет определить положение вращающегося вала с точностью до 1/16 полного оборота, т. е. до 22,5°. В случае электронного компаса это соответствует двум румбам направления. То есть для того, чтобы получить стандартный компас, разделенный на 32 румба, нужен пятизначный код Грея, но для большинства практических применений такой точности достаточно. В статье [27] описана практическая конструкция флюгера с электронным съемом угла поворота, построенная на указанных принципах.



ГЛАВА 15

Математическая электроника или игра в квадратики

Устройство логических микросхем и двоичные операции

> Трактирщик отворил дверь большой комнаты, где совсем недавно вместо прежней дрянной печурки поставили прекрасный большой камин.
>
> *А. Дюма. «Три мушкетера»*

Типов логических (или цифровых — будем считать, что это синонимы) микросхем не так уж и много. Если не считать памяти или, например, процессорных ядер, то подавляющее большинство микросхем, с которыми имеет дело пользователь, относится к одной и той же технологической разновидности, называемой КМОП (CMOS, Комплементарные [транзисторы] Металл-Окисел-Полупроводник). Но и упомянутая память с процессорами, и многие другие представители полупроводниковой техники, как сложно они бы ни были устроены внутри, лицом к пользователю обращены все равно через КМОП-совместимые выводы (порты).

Первые лет двадцать своего существования КМОП-технология конкурировала с другими технологиями, важнейшей из которых была ТТЛ (TTL, Транзисторно-Транзисторная Логика), не терявшая своих свойств на существенно более высоких частотах. Существовали еще некоторые специальные технологии, например, скоростная эмиттерно-связанная-логика (ЭСЛ). На нее когда-то возлагали большие надежды конструкторы быстродействующей вычислительной техники, но уж слишком неудобна оказалась ЭСЛ-технология по причине огромного потребления энергии. Знаменитому Сеймуру Крэю, конструктору лидирующих в мире суперкомпьютеров 1960–1970-х годов, даже пришлось погрузить платы Cray-1 полностью в охлаждающую жидкость (масло), иначе с отводом тепла было не справиться.

Заметим, что различие между элементами разных серий чисто технологическое — на заре развития электроники существовали еще многие другие серии (некоторые из них мы затронули в предыдущей главе), но функционально одноименные элементы из разных серий делают одно и то же, и на схемах обозначаются одинаково, хотя по электрическим параметрам могут сильно различаться. И ТТЛ и ЭСЛ-микросхемы всех поколений легко найти в продаже, но на практике они уже почти

и не применяются, потому что современные модификации КМОП догнали и перегнали их по скоростным параметрам, а во всем остальном КМОП гораздо удобнее, как мы увидим.

Технология КМОП за полвека своего развития претерпела множество изменений, потому она сама по себе делится на несколько семейств, обладающих, однако, общими чертами. На практике в наших устройствах мы будем применять как традиционные КМОП-микросхемы, так и современные быстродействующие их разновидности. Тем не менее, несколько слов про ТТЛ сказать придется, потому что ее электрические параметры — своеобразный стандарт, на который традиционно равняются остальные технологии. Многие существующие термины, стандарты и спецификации выросли из требования обеспечить совместимость и взаимозаменяемость микросхем той или другой технологии.

ТТЛ

Транзисторно-транзисторная логика, введенная в практику фирмой Texas Instruments в 1965 году, возникла раньше КМОП (первая КМОП-микросхема была выпущена в 1968 году фирмой National Semiconductor) и стала наследницей диодно-транзисторной логики (ДТЛ), примеры которой мы приводили в *главе 14* (см. рис. 14.4). Основной родовой признак ТТЛ — использование биполярных транзисторов, причем структуры только *n-p-n*. КМОП же, как следует из ее названия, основана на полевых транзисторах с изолированным затвором структуры МОП, причем комплементарных, т. е. обеих полярностей: и с *n*-, и с *p*-каналом. Схемотехника базовых логических элементов ТТЛ и КМОП приведена на рис. 15.1. На Западе их еще называют *вентилями* — чем можно оправдать такое название, мы увидим в конце главы.

Входной многоэмиттерный транзистор ТТЛ мы уже рисовали в *главе 11* (см. рис. 11.3) — он может иметь сколько угодно (на практике — до восьми) эмит-

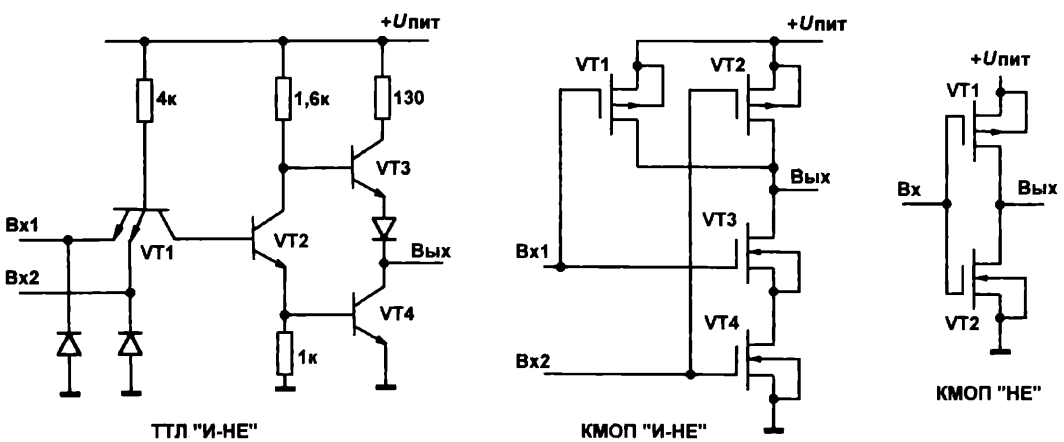


Рис. 15.1. Схемы базовых элементов ТТЛ и КМОП

теров, и у элемента тогда будет соответствующее число входов. Если любой из эмиттеров транзистора VT1 замкнуть на «землю», то транзистор откроется, а фазорасщепляющий транзистор VT2 (с его работой мы знакомы по рис. 6.8) — закроется. Соответственно, выходной транзистор VT3 откроется, а VT4 — закроется, на выходе будет высокий логический уровень (уровень логической единицы). Если же все эмиттеры присоединены к высокому потенциалу (или просто «висят» в воздухе), то ситуация будет обратная — VT2 откроется током через переход база-коллектор VT1 (такое включение транзистора называется *инверсным*), и на выходе установится ноль за счет открытого транзистора VT4. Такой ТТЛ-элемент будет осуществлять функцию «И-НЕ» (логический ноль на выходе только при единицах на всех входах).

Выходной каскад ТТЛ-элемента представляет собой некое подобие комплементарного (*пушпульного*) каскада класса В, знакомого нам по аналоговым усилителям (см. рис. 8.2). Однако воспроизведение *p-n-p*-транзисторов оказалось для ТТЛ-технологии слишком сложным, потому такой каскад носит еще название *псевдокомплементарного*: верхний транзистор VT3 работает в режиме эмиттерного повторителя, а нижний — в схеме с общим эмиттером.

Кстати, заметим, что из-за недоступности *p-n-p*-структуры транзисторов воспроизведение схемы «ИЛИ» для ТТЛ-технологии оказалось крепким орешком, и ее схемотехника довольно существенно отличается от показанной на рис. 15.1 базовой схемы элемента «И-НЕ».

Как видно из схемы, ТТЛ-элемент существенно несимметричен и по входам, и по выходам. По входу напряжение логического нуля должно быть достаточно близко к «земле», при напряжении на эмиттере около 1,5 В (при стандартном для ТТЛ питании 5 В) входной транзистор уже запирается. Причем при подаче нуля нужно обеспечить отвод довольно значительного тока база-эмиттер — около 1,6 мА для стандартного элемента, отчего для элементов ТТЛ всегда оговаривается максимальное количество одновременно подсоединенных к выходу других таких элементов (стандартно — не более десятка). В то же время логическую единицу на входы можно не подавать вовсе. Практически, однако, подавать ее следует — по правилам незадействованные входы ТТЛ должны быть присоединены к питанию через резисторы 1 кОм.

Еще хуже дела обстоят на выходе: напряжение логического нуля обеспечивается открытым транзистором и действительно довольно близко к нулю — даже при нагрузке в виде десятка входов других таких же элементов оно не превышает 0,5 В, а в нормах на сигнал ТТЛ оговорена величина не более 0,8 В. А вот напряжение логической единицы довольно далеко отстоит от питания и составляет при питании 5 В в лучшем случае (без нагрузки) от 3,5 до 4 В, практически же в нормах оговаривается величина 2,4 В.

При всем этом ТТЛ-элемент может работать в достаточно узком диапазоне напряжения питания — стандартно от 4,5 до 5,5 В. Самым крупным (и даже более серьезным, чем остальные) недостатком ТТЛ является высокое потребление — до 2,5 мА на один такой элемент, и это без учета вытекающих токов по входу и потребления нагрузки по выходу.

В дальнейшем развитие ТТЛ шло по линии уменьшения потребления и улучшения электрических характеристик, в основном за счет использования так называемых *переходов Шоттки*, на которых падение напряжения может составлять 0,2–0,3 В вместо обычных 0,6–0,7 В (технология ТТЛШ, обозначается буквой S в наименовании серии, отечественные аналоги: серии 531 и 530). Базовая технология, которая составляла основу широко распространенной в 1960–1970-х годах серии 74 без дополнительных букв в обозначении (аналоги — знаменитые отечественные серии 155 и 133), давно не используется. Если очень надо, то ТТЛ-микросхемы следует выбирать из вариантов, представленных малопотребляющими сериями типа 74LS*xx* (серии 555 и 533) или быстродействующими типа 74F*xx* (серия 1531).

Основные характеристики КМОП

КМОП-элементы намного ближе к представлению о том, каким должен быть идеальный логический элемент. Для начала, как можно видеть из рис. 15.1, они практически симметричны, как по входу, так и по выходу. Открытый полевой транзистор на выходе (либо *p*-типа для логической единицы, либо *n*-типа для логического нуля) фактически представляет собой, как мы знаем, просто сопротивление, которое для обычных КМОП-элементов может составлять от 100 до 300 Ом (под «обычными» или «классическими» КМОП мы подразумеваем здесь серию 4000A или 4000B, см. далее). Для дополнительной симметрии на выходе обычно ставят последовательно два инвертора, подобных показанному на рис. 15.1, *справа* (жалко, что ли, транзисторов, если потребление не растет?). Поэтому на симметрии выхода не сказывается то, что в нижнем плече для схемы «И-НЕ» стоят два таких транзистора последовательно.

Для схемы «ИЛИ» такие транзисторы будут стоять в верхнем плече — она полностью симметрична схеме «И», что тоже плюс технологии КМОП по сравнению с ТТЛ. Обратите также внимание, что выходной каскад инвертора построен не по схеме «пушпульного» каскада, т. е. это не истоковые повторители напряжения, а транзисторы в схеме с общим истоком, соединенные стоками, что позволяет получить дополнительный коэффициент усиления по напряжению.

На практике особенности построения элемента приводят к тому, что в КМОП-микросхемах:

- на ненагруженном выходе напряжение логической единицы практически равно напряжению питания, а напряжение логического нуля практически равно потенциалу «земли»;
- порог переключения близок к половине напряжения питания;
- входы практически не потребляют тока, т. к. представляют собой изолированные затворы МОП-транзисторов;
- в статическом режиме весь элемент также не потребляет тока от источника питания.

Из последнего положения вытекает, что схема любой степени сложности, построенная с помощью КМОП-элементов, в «застывшем» состоянии или при малых

рабочих частотах, не превышающих десятка-другого килогерц, практически не потребляет энергии! Отсюда ясно, как стали возможными такие фокусы, как наручные часы, которые способны идти от малюсенькой батарейки годами, или sleep-режим микроконтроллеров, в котором они потребляют от 1 до 50 мкА на все десятки тысяч составляющих их логических элементов.

Другое следствие упомянутых особенностей — исключительная помехоустойчивость, достигающая половины напряжения питания. Но это еще не все преимущества. КМОП-микросхемы «классических» серий могут работать в диапазоне напряжений питания от 2 до 18 В, а современные быстродействующие — от 2 до 7 В. Единственное, что при этом происходит: при снижении питания довольно резко — в разы — падает быстродействие и ухудшаются некоторые другие характеристики. Но именно по причине широкого диапазона напряжения питания КМОП-микросхемы классических серий имеет смысл применять и сегодня.

Кроме того, выходные транзисторы КМОП, как и любые другие полевые транзисторы, при перегрузке (например, в режиме короткого замыкания) работают как источники тока — при напряжении питания 15 В этот ток составит около 30 мА, при 5 В — около 5 мА. Причем это может быть долгосрочный режим работы таких элементов (с некоторыми оговорками при высоких напряжениях питания). Единственное, что при этом надо проверить, — не превышается ли значение суммарного допустимого тока через вывод питания, которое обычно составляет около 50 мА. То есть, возможно, придется ограничить число выходов, одновременно подключенных к низкоомной нагрузке. Естественно, о логических уровнях в таком режиме уже речи не идет, только о втекающем или вытекающем токе.

Подробности

Формально ток через один вывод микросхемы серии 4000 не должен превышать 5–7 мА при питании 12–15 вольт и 1–2 мА при питании 5 В. При этом логические уровни сохраняются. Не надо еще забывать, что у выхода есть ограничение по мощности, которое составляет для рядовых микросхем серии 4000 (например, микросхемы 4011) 100 мВт. Поэтому максимальное значение долговременного значения тока для питания 12–15 вольт через один вывод составляет примерно 7–8 мА, но при уменьшении напряжения питания это требование быстро снижается, и уже при 10 вольтах микросхема все равно не выдаст тока, перегружающего выход по выделяющейся на нем мощности.

И тут мы подходим к основному недостатку «классической» КМОП-технологии — низкому в сравнении с ТТЛ быстродействию. Обусловлено оно тем, что изолированный затвор МОП-транзистора представляет собой конденсатор довольно большой емкости — в базовом элементе до 10–15 пФ. В совокупности с выходным сопротивлением предыдущей схемы такой конденсатор образует фильтр низких частот. Обычно рассматривают не просто частотные свойства, а время задержки распространения сигнала на один логический элемент. Задержка возникает из-за того, что фронт сигнала не строго вертикальный, а наклонный, и напряжение на выходе еще только начнет нарастать (или снижаться), когда изменение напряжения на входе достигнет уже значительной величины. Время задержки могло достигать у ранних серий КМОП величины 200–250 нс (сравните — у базовой серии ТТЛ всего 7,5 нс). На практике при напряжении питания 5 В максимальная рабочая частота

«классического» КМОП не превышает 1–3 МГц. Попробуйте соорудить на логических элементах генератор прямоугольных сигналов по любой из схем, которые будут рассмотрены в *главе 16*, и вы увидите, что уже при частоте 1 МГц форма сигнала будет скорее напоминать синусоиду, чем прямоугольник.

Другим следствием наличия высокой входной емкости является то, что при переключении возникает импульс тока перезарядки этой емкости, — т. е., чем выше рабочая частота, тем больше потребляет микросхема, и считается, что при максимальных рабочих частотах ее потребление может сравниться с потреблением ТТЛ (по крайней мере, ТТЛ серии 74LS). Дело еще усугубляется тем, что из-за затянутых фронтов импульсов элемент достаточно длительное время находится в активном состоянии, когда оба выходных транзистора приоткрыты (т. е. возникает так называемый эффект *сквозного тока*). Это же затягивание фронтов в сочетании с высокоомным входом приводит к снижению помехоустойчивости при переключении — если на фронте сигнала «сидит» высокочастотная помеха, то это может приводить к многократным переключениям выхода, как это было у компаратора (см. *главу 13*).

Незадействованные входы элемента КМОП нужно обязательно подключать куданибудь: либо к земле, либо к питанию (резисторов при этом не требуется, т. к. вход тока не потребляет), либо объединять с соседним входом — иначе наводки на столь высокоомном входе полностью нарушат работу схемы. Причем в целях снижения потребления следует делать это и по отношению к незадействованным элементам в том же корпусе (но не ко всем незадействованным выводам, конечно). «Голый» вход КМОП из-за своей высокоомности может стать также причиной повышенной «смертности» чипов при воздействии статического электричества (хотя на практике входы всегда шунтируют диодами, как показано на рис. 11.4). Допустимый ток через эти диоды также оговаривается в спецификациях.

> ***Заметки на полях***
>
> Один из самых интересных эффектов, связанных с КМОП, мы уже упоминали в *главе 11* — если вы забудете включить питание, но подадите на КМОП-микросхему сигнал, то она может заработать, как ни в чем не бывало: на выходах появятся сигналы (правда, это только видимость работы, потому что функциональность схемы будет, скорее всего, нарушаться). Этот факт объясняется наличием защитных диодов по входу (см. рис. 11.4) и иногда даже используется для удаленного питания датчиков по сигнальному проводнику. Повторим, что этот факт должен служить напоминанием о том, что наличие обесточенной микросхемы в окружении работающих источников сигналов может не только не снизить потребление, но даже его увеличить.

В современных КМОП, в отличие от «классических», большинство недостатков, связанных с низким быстродействием, удалось преодолеть (правда, за счет снижения допустимого диапазона питания).

Характеристики различных серий КМОП

Развитие КМОП было, естественно, направлено в сторону устранения или хотя бы сглаживания имеющихся недостатков. Самая первая серия КМОП 4000 была не очень удачной — средние задержки в 200 нс и более, напряжение питания от 5 до

15 В, нагрузочная способность выхода — не более 0,6 мА. Причем советские аналоги (серии К176 и 164) были еще хуже, т. к. требовали питания ровно 9 В (мне так и не удалось нигде узнать, с какими допусками). Для стыковки с ТТЛ таких микросхем пришлось придумывать специальные «преобразователи уровня», которые тянули относительно большой вытекающий ток по входу ТТЛ-элементов и умели подгонять уровни при различном напряжении питания.

А вот серия 4000А (отечественные К561 в корпусе DIP и «военная» 564 в планарном корпусе SOIC или похожем на него SOP) и, особенно, 4000В (частично К561 и вся К1561[1]) применяются и по сей день — в основном из-за неприхотливости и беспрецедентно широкого диапазона питающих напряжений: от 3 до 18 В, что позволяет без излишних проблем совмещать цифровые и аналоговые узлы в одной схеме. Задержки для этих серий составляют порядка 90–100 нс, предельная рабочая частота — до 10 МГц, а выходные токи без ущерба для логического уровня — от 1 до 6 мА (большее значение при большем напряжении питания). Отметим, что указанные скоростные характеристики достигаются при достаточно высоком напряжении питания, при обычных для современной электроники напряжениях быстродействие существенно снижается: при 5 В рабочая частота уже не превышает 3–5 МГц.

В настоящее время доступны быстродействующие серии КМОП с крайне неудачным общим названием 74, совпадающим с ТТЛ (серия 54 — то же самое, но для военно-космических применений). Чтобы не запутаться, имейте в виду, что если в наименовании серии присутствует буква С, то это КМОП, а все остальные (менее многочисленные) представители семейства 74 есть ТТЛ-микросхемы. Самые популярные разновидности — серия 74НС*xx* (отечественный аналог — 1564 или КР1564) и 74АС*xx* (1554). Номер серии, обозначения и разводка выводов элементов, совпадающие с таковыми для ТТЛ, были, вероятно, первоначально выбраны из маркетинговых соображений, чтобы подчеркнуть быстродействие и совместимость с ТТЛ, однако по другим параметрам это совсем не ТТЛ.

Серии 74НС и 74АС совмещают в себе многие удобства КМОП (симметричность уровней, отсутствие потребления в статическом режиме) и быстродействие ТТЛ, достигающее десятков мегагерц (у 74АС даже до сотни). Пожертвовать, как мы говорили, пришлось расширенным диапазоном питания: рекомендованный диапазон напряжения питания для всех КМОП из 74-й серии — от 2 до 5 В, максимально допустимое — 7 В, поэтому серию 4000В они заменяют не полностью. Зато эти серии без проблем совместимы с любыми современными микросхемами, рассчитанными на пониженное питание в пределах 2,2–3,3 вольта, потому что нижний предел у НС и АС даже меньше, чем у «классики».

В этой книге мы будем ориентироваться на наиболее популярные у нас и по сей день микросхемы 561-й серии (4000В), но учтите, что при напряжениях питания 5 В и менее их можно *почти* без ограничений заменять микросхемами серии 74НС

---

[1] В дальнейшем букву К, означающую «бытовое применение», в наименовании серий мы будем часто опускать. Напомним также, что для названий корпусов отечественных микросхем приводятся импортные аналоги, но при замене следует учитывать разницу в шаге между выводами (см. *главу 11*).

или отечественными 1564. В некоторых случаях применение новых серий даже предпочтительнее, т. к. они формируют более крутые фронты сигналов.

«Почти» по отношению к взаимозаменяемости относится к потреблению — в покое все КМОП-элементы не потребляют тока, но с ростом частоты потребление быстродействующих растет быстрее. На рис. 15.2 показаны эмпирические графики потребления двух разновидностей КМОП-микросхем в расчете на один логический элемент в зависимости от частоты. Видно, что для «классической» 561-й серии потребление растет строго линейно с крутизной примерно 25 мкА на каждые 300 кГц увеличения частоты. Для элемента 74HC оно сначала быстро растет, затем темпы прироста снижаются, но в любом случае ток потребления остается как минимум вдвое больше, чем у «классического» элемента.

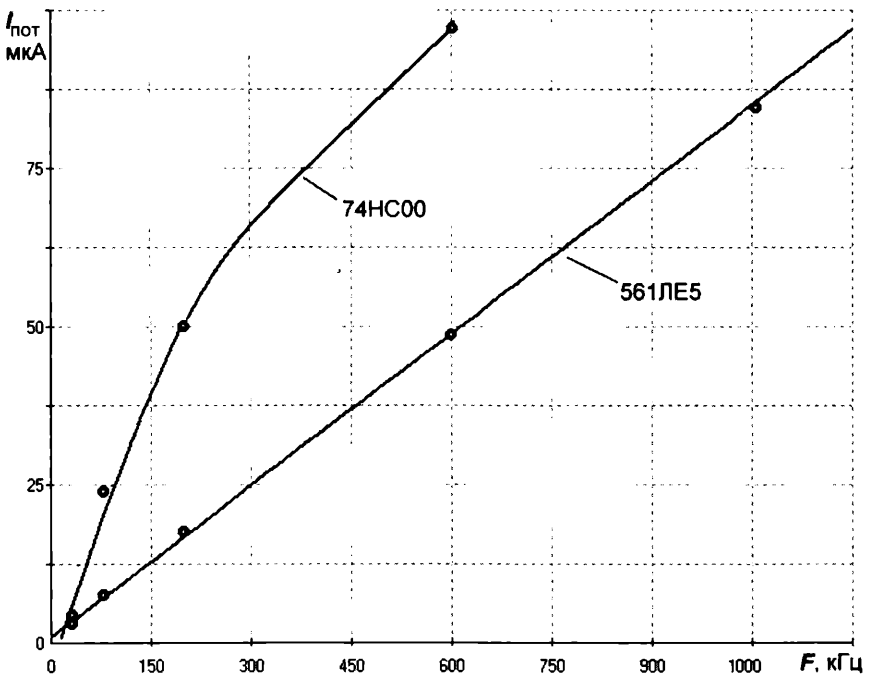


Рис. 15.2. Потребление КМОП-микросхем в расчете на один логический элемент в зависимости от частоты (напряжение питания 5 В)

Заметки на полях

Как видите, в абсолютном выражении потребление всех КМОП-элементов достаточно мало, и на частотах в десятки килогерц составляет единицы микроампер. Однако это потребление резко возрастает при увеличении напряжения питания — так, потребление схемы лабораторного генератора по рис. 16.14 составляет не более 150–200 мкА при 5 В, но при 15 В оно уже будет составлять порядка 1,5 мА. Отсюда общее правило для КМОП-микросхем: для уменьшения потребления снижение напряжения питания даст больший эффект, чем снижение рабочей частоты.

Серия 74AC еще мощнее, чем 74HC, и более быстродействующая (задержки порядка 5–7 нс против 10–20 нс у 74HC) — соответственно, при той же частоте она по-

требляет еще больше. Выходные максимально допустимые токи некоторых микросхем серии 74HC могут достигать 25 мА, а серии 74AC — аж 50 мА. Но в долгосрочном режиме такие токи гонять через выводы не рекомендуется: нормальный ток для выхода 74AC без нарушения логических уровней составляет 20–24 мА, а для 74HC — 4–8 мА (причем, напомним, что через вывод питания суммарный ток не должен превышать величины порядка 50 мА).

В продаже иногда встречаются еще микросхемы с одной буквой C (74C*xx*), которые по всем параметрам аналогичны серии 4000B, но имеют несколько более мощные выходы. Другое обозначение должно подчеркнуть их совместимость с остальными членами семейства 74 (в том числе по названиям и разводке выводов, отличном от серии 4000). При построении генератора для импульсного преобразователя-инвертора (см. рис. 9.20) мы применяли микросхему из этой серии — 74C14 (шесть триггеров Шмидта в одном корпусе).

У разных производителей могут быть разные приставки (префиксы) к основному названию серии, как 4000B, так и серий 74, — например, у Fairchild Semiconductor микросхема будет называться CD4001B, у Texas Instruments — SN4001B, у Motorola — MC14001B (более подробно об этом рассказано в *приложении 3*).

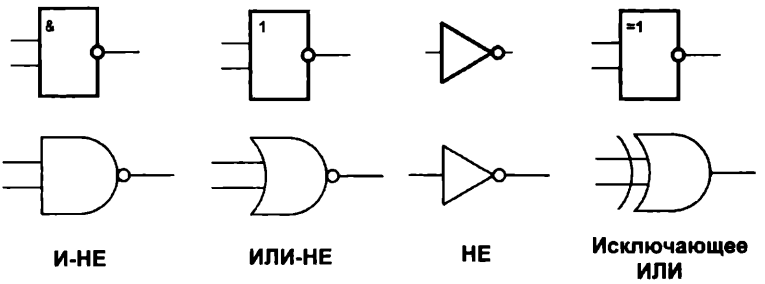
Как мы уже говорили, наименования одинаковых по функциональности элементов для КМОП разных серий и ТТЛ различаются. Причем функциональные наименования у серий 1564 и 1554 соответствуют ТТЛ (если аналоги в «классических» сериях существуют), а не КМОП. Микросхема, содержащая в одном корпусе четыре элемента «И-НЕ», подобные показанным на рис. 15.1, в отечественном варианте КМОП носит имя 561ЛА7, ТТЛ носит название 155ЛА3, в западном варианте это 4011 в «классической» серии и 74xx00 в быстродействующих версиях как КМОП, так и ТТЛ. Подробнее функциональные наименования описаны в *приложении 3*.

На рис. 15.3 показаны условные обозначения основных логических элементов на электрических схемах, причем нельзя не согласиться, что отечественные обозначения намного логичнее, легче запоминаются и проще выполняются графически, чем зарубежные, поэтому-то их обозначения логических элементов у нас так и не прижились (как, кстати, и многие другие, — например, обозначения резисторов и электролитических конденсаторов). *Крайний справа* элемент под наименованием «Исключающее ИЛИ» нам еще неизвестен, но скоро мы его будем изучать.

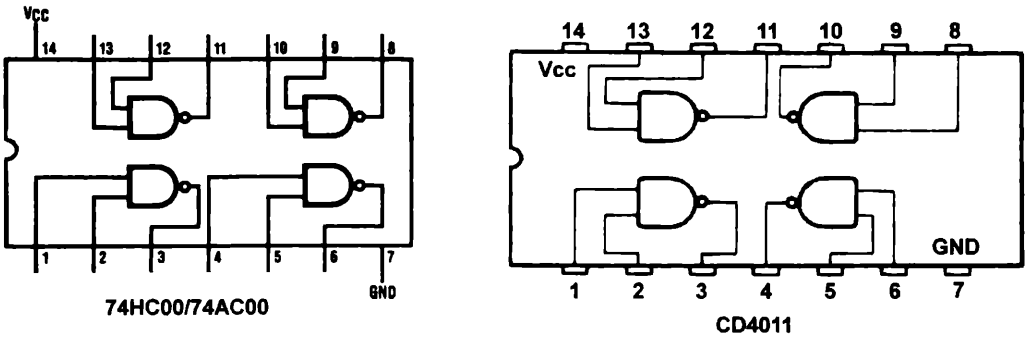
Как вы можете убедиться, сравнив фирменные описания (datasheets) микросхем разных серий, в «классической» серии для всех двухвходовых логических микросхем разводка выводов одинакова, а вот другие серии с универсализацией подкачали. Для ТТЛ всех разновидностей (в том числе и отечественных) разводка совпадает с 74AC/HC, чем подчеркивается, что они взаимозаменяемы. Кстати, заимствованная у «классической» ТТЛ разводка для микросхем 2И-НЕ 74HC00 (1564ЛА3) и 74AC00 (1554ЛА3), а также их сестер типа 2ИЛИ-НЕ удобнее других (см. рис. 15.3, *б*) — при каскадном соединении двух элементов достаточно соединить идущие подряд выводы 3 и 4 (или 3, 4 и 5, если входы объединяются) и аналогично поступить с другой половиной микросхемы.

Естественно, все элементы в одном корпусе абсолютно идентичны и взаимозаменяемы, поэтому для таких микросхем, если даже и номера выводов корпуса на

принципиальной схеме указаны, элементы можно заменять друг на друга в процессе изготовления платы. Нередко на схеме не указывают и расположение выводов питания.



а



б

Рис. 15.3. *а* — обозначения основных логических элементов на схемах: *вверху* — отечественные; *внизу* — зарубежные; *б* — разводка выводов КМОП-микросхем 4×2И-НЕ быстродействующих серий 74AC00/74HC00 (*слева*) и классической серии CD4011 (*справа*)

Специальные буферные элементы с мощным выходом применяют для того, чтобы усилить выходы КМОП: с инверсией — 561ЛН2 или без нее — 561ПУ4. Они содержат шесть таких инверторов или буферов в одном корпусе. Отметим, что точного импортного аналога микросхемы 561ЛН2 не существует, при копировании наши разработчики улучшили микросхему CD4049, избавившись от лишних контактов корпуса. Правда, они не пошли до конца и не сделали то же самое для микросхемы 561ПУ4, содержащей 6 буферных усилителей без инверсии. В результате у 561ПУ4, у ее аналога CD4050, как и у CD4049, плюс питания присоединяется нестандартно — к выводу 1, а выводы 13 и 16 — не задействованы (и, напомним, не должны никуда присоединяться, что обозначено буковками NC — No Connected).

Заметки на полях

Объяснение этой кажущейся несуразице с лишними выводами одновходовых элементов простое — в первых сериях 4000 и 4000А (К176) существовали преобразователи уровня для перехода от 9-вольтовой КМОП-логики к 5-вольтовой ТТЛ (откуда и отечественное название ПУ). В этих преобразователях на вывод 16 подавалось еще одно

напряжение питания 9 В. В сериях 4000А и 4000В необходимость в дополнительном питании отпала (они и сами чудесно работают при питании 5 В), а разводка выводов осталась.

Следует отметить, что по внутреннему устройству микросхемы с одновходовыми элементами, вероятно, самые простые из всех микросхем вообще. Элемент с инверсией (ЛН2, 4049) состоит всего из двух транзисторов (см. рис. 15.1, *справа*), а буферный элемент (ПУ4, 4050) — из двух таких инверторов, включенных последовательно. Однако у них есть один эксплуатационный нюанс, который также унаследован от времен, когда такие микросхемы служили для перехода от КМОП к ТТЛ. Он заключается в том, что нижний транзистор выходного каскада мощнее верхнего. В результате в состоянии логического нуля по выходу эти микросхемы могут принять большой втекающий ток без ущерба для логических уровней: 5–6 мА при питании 5 В и аж до 48 мА при 15 В. А вот в состоянии логической единицы значения выходного тока у них стандартные для «классической» КМОП: 1,2 мА при 5 В и до 8 мА при 15 В питания. При практическом применении ЛН2 и ПУ4 эту несимметрию нужно учитывать.

В отличие от «классических», быстродействующие аналоги 74HC04 (74HC4049) и 74HC4050 (в AC-версии это 74AC04 и 74AC34, соответственно) полностью симметричны, допускают как втекающий, так и вытекающий долговременный выходной ток через каждый вывод до 20 мА (но не более 50 мА в сумме по всем выводам) и предпочтительны для использования при напряжениях питания 3–5 В.

Выходы обычных логических элементов можно объединять с целью умощнения — например, выход одного инвертора в составе микросхемы ЛН2 формально «тянет» ток по высокому уровню до 6 мА при 5 В питания (на самом деле гораздо больше, если выйти за пределы ограничений, накладываемых условием ненарушения логических уровней), а если соединить выходы всех шести входящих в состав микросхемы элементов, то можно подключать нагрузку до 30–40 мА, — главное, не превысить допустимый ток через вывод питания. Естественно, при этом необходимо также объединить и входы, превратив всю микросхему как бы в один мощный инвертор. Именно этот прием и использовался в упомянутой схеме на рис. 9.20 — выходы 74С14 допускают 15–16 мА по каждому выводу (т. е. 30 мА при включении попарно, как в схеме). Более быстродействующие и мощные микросхемы из серии AC в данном случае применять нельзя из-за меньшего допустимого питания, но мы их используем в параллельном включении для умощнения выходов контроллера с 5-вольтовым питанием при работе в режиме ШИМ (см. рис. 22.2).

Есть, разумеется, и логические элементы с бóльшим количеством входов. В качестве справочника по «классической» КМОП, в котором приведены не только основные сведения и разводка выводов, но и подробно объясняется работа микросхем базовых серий с многочисленными примерами, я бы рекомендовал разыскать у букинистов или скачать из Сети книгу [18]. Это суперпопулярное пособие вышло в свое время несколькими изданиями и почти не устарело (правда, некоторых современных типов микросхем в нем нет). В Сети, разумеется, тоже можно найти и другие подобные справочники.

Следующей широко употребляемой разновидностью логических микросхем являются элементы, имеющие выход с открытым истоком (с открытым коллектором для

ТТЛ). Такой выход, как мы помним, имеет компаратор 521СА3 (см. *главу 12*). Есть такие элементы и с чисто логическими функциями: в КМОП-серии это CD40107 (561ЛА10, содержит два двухвходовых элемента «И-НЕ»), в быстродействующих КМОП это 74HC05 (шесть инверторов, аналог ЛА10 под названием 74HC22 снят с производства). Причем CD40107 может коммутировать значительный втекающий ток: аж до 136 мА при 25 °С и 10 В питания и 70 мА при 5 В. 74HC05 скромнее, и коммутирует стандартные для этой серии 20 мА.

Эти элементы используют не только для коммутации мощной нагрузки, но и для объединения на общей шине в так называемое *проводное* или *монтажное* «И» (рис. 15.4). Иногда по традиции его именуют «монтажным ИЛИ», но название это некорректное, ибо соответствует отрицательной логике — на общей шине логическая единица будет только тогда, когда выходы всех элементов установятся в 1, а если хотя бы один выход находится в нуле, то и на шине будет ноль, что в положительной логике соответствует операции «И». Авторы пособий сами часто путаются в такой системе обозначений, потому в Сети можно встретить названия «монтажное И» и «монтажное ИЛИ» почти одинаково часто. Заметим еще, что в ЭСЛ-логике есть аналогичные элементы, где вместо открытых коллекторов выведены открытые эмиттеры, — вот там название «монтажное ИЛИ» полностью отвечает смыслу операции.

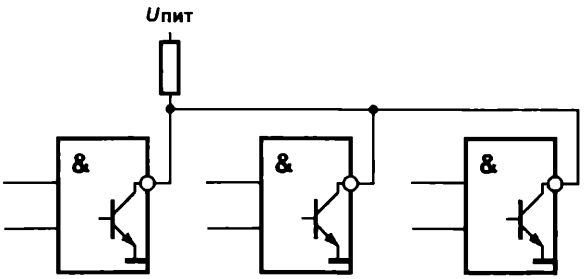


Рис. 15.4. Объединение элементов с открытым коллектором по схеме «монтажное И»

Для объединения выходов могут служить и так называемые *элементы с третьим состоянием*. Это соответствует не логическому понятию состояния, а электрическому, — третье состояние в этом случае обозначает просто обрыв, отключение выхода элемента от вывода микросхемы. Такие элементы имеются и в составе серий, но наиболее часто применяются в составе более сложных микросхем. Например, выводы многих микроконтроллеров или микропроцессоров имеют возможность переключения в третье состояние.

Заметки на полях

Мы часто будем усиливать выходы КМОП-микросхем с помощью отдельного ключевого транзистора, и схема его включения может представлять в таком случае исключение из того правила, что в ключевом режиме обязательно «привязывать» базу к «земле», как это было оговорено в *главе 6* (см. рис. 6.4 и относящийся к нему текст). Дело в том, что подключенная к выходу логического элемента база транзистора всегда будет привязана через токоограничивающий резистор к какому-нибудь потенциалу, и в воздухе никогда не «повиснет», поэтому и запирающий резистор можно не ставить. Однако это не относится к случаю, когда база управляется от выхода ТТЛ-микро-

схемы через диод, включенный в прямом направлении, как это часто делают, чтобы обеспечить надежное запирание транзистора (см., например, [3]) На мой взгляд, ставить такой диод совершенно не требуется, но если уж автор построил схему именно так, то нужно ставить и запирающий резистор, потому что при нулевом потенциале на выходе микросхемы диод запирается, и база тогда «повисает в воздухе».

Двоичный сумматор на логических микросхемах

Заметим сразу, что схема этого устройства в том виде, в котором мы ее сейчас будем конструировать, сама по себе довольно бесполезна — если вы, конечно, не хотите повторить подвиг советского конструктора Михаила Александровича Карцева. Он создал в 1970-х годах на микросхемах малой степени интеграции (т. е. фактически на отдельных логических элементах) очень удачную ЭВМ под названием М-10, замечательную тем, что отдельные ее экземпляры в нашем оборонном комплексе продержались аж до начала нового тысячелетия. При желании повторить такой подвиг учтите, что основная проблема, которую вам придется решать, состоит вовсе не в том, чтобы такую машину сконструировать схемотехнически — это не самая трудная часть работы. Самое трудное для подобных суперконструкций — решить проблему отвода тепла, выделяемого сотнями тысяч быстродействующих логических микросхем. Суперкомпьютер Cray-1 на отдельных логических элементах, как мы говорили, даже имел платы, полностью погруженные в масло, а в последующих версиях «Крэя» было применено водяное охлаждение через медные каналы, примыкающие к платам.

Наконец, если очень хочется, то готовый двоичный сумматор есть в интегральном исполнении (561ИМ1, есть сумматоры и помощнее). Зачем же мы тогда будем его конструировать? А затем, что его устройство очень хорошо иллюстрирует две вещи: во-первых, само применение логических микросхем, во-вторых — разве не любопытно знать, как устроен самый главный узел компьютера, арифметико-логическое устройство, АЛУ? Знание этого вам очень пригодится для лучшего понимания работы микроконтроллеров и принципов их программирования. Кроме того, мы на этом примере познакомимся еще с одним важным типом логических элементов.

Итак, вспомним, что нам, собственно, нужно делать — а именно: воспроизвести таблицу сложения двоичных чисел, которая была показана для одноразрядных чисел в *главе 14*. Так как при сложении единиц получается двухразрядное двоичное число, то перепишем эту таблицу в двухразрядном представлении:

Вх1	Вх2	Вых
0	0	00
0	1	01
1	0	01
1	1	10

Теперь разобьем таблицу на две: одну для разряда собственно суммы, другую для значения переноса в следующий разряд:

Сумма		
Вх1	Вх2	Вых
0	0	0
0	1	1
1	0	1
1	1	0

Перенос		
Вх1	Вх2	Вых
0	0	0
0	1	0
1	0	0
1	1	1

Сравним вторую таблицу с таблицей состояний для базовых логических функций (я их повторю, чтобы не пришлось листать книгу):

«ИЛИ»		
Вх1	Вх2	Вых
0	0	0
0	1	1
1	0	1
1	1	1

«И»		
Вх1	Вх2	Вых
0	0	0
0	1	0
1	0	0
1	1	1

Для переноса имеем полное совпадение с функцией «И». То есть, для того чтобы обеспечить перенос, нам нужен всего лишь один логический элемент «И», который получается комбинированием стандартного «И-НЕ» с инвертором[1]. Хуже с разрядом суммы: первые три значения обеспечивает элемент «ИЛИ», однако при сложении единиц возникает несоответствие (логическое и арифметическое сложения, как мы говорили, не адекватны друг другу). Нужен специальный элемент, который мог бы получить название *элемент несовпадения*: в самом деле, у него логическая единица на выходе тогда, когда входы имеют разное состояние, а если они одинаковы — на выходе ноль. Для того чтобы его сконструировать, взглянем на таблицу истинности элемента «И-НЕ» (для наглядности я повторю и ее):

«И-НЕ»		
Вх1	Вх2	Вых
0	0	1
0	1	1
1	0	1
1	1	0

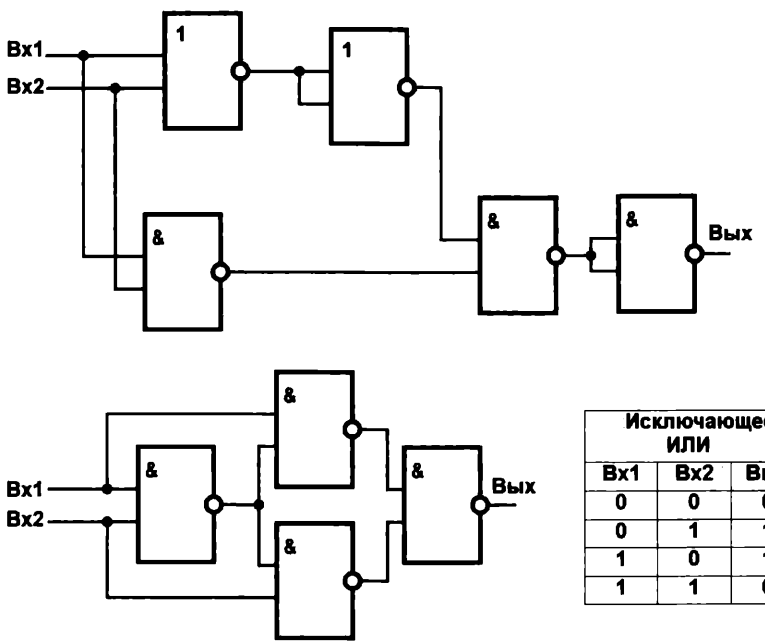
[1] Вообще-то, в различных сериях микросхем есть и непосредственно элементы «И» (как и «ИЛИ») без инверсии, но в «классической» КМОП их нет, и в целях унификации мы будем пользоваться только элементами «И-НЕ» и «ИЛИ-НЕ» (для КМОП это 561ЛА7 и 561ЛЕ5 соответственно).

Сравним таблицы «ИЛИ», «И-НЕ» и необходимой нам суммы: в первом случае мы получаем то, что надо, в верхних трех строках, во втором — в нижних. Как бы их объединить? Да очень просто — через функцию «И»:

Сумма				
Вх1	**Вх2**	**Вых «ИЛИ»**	**Вых «И-НЕ»**	**«ИЛИ» × «И-НЕ» («Исключающее ИЛИ»)**
0	0	0	1	0
0	1	1	1	1
1	0	1	1	1
1	1	1	0	0

Логический элемент с такой функцией «несовпадения» носит специальное название — «Исключающее ИЛИ». Существует и обратный элемент «совпадения», который представляет собой инверсию выхода «Исключающего ИЛИ» и носит название «Включающего ИЛИ».

Обозначение элемента «Исключающее ИЛИ» уже было показано на рис. 15.3. А как можно его составить из элементов «И-НЕ» и «ИЛИ-НЕ», показано на рис. 15.5. *Верхний* вариант полностью соответствует нашим рассуждениям и потребует двух корпусов микросхем, а *нижний* — пример того, как можно построить «Исключаю-



Исключающее ИЛИ		
Вх1	**Вх2**	**Вых**
0	0	0
0	1	1
1	0	1
1	1	0

Рис. 15.5. Варианты реализации элемента «Исключающее ИЛИ» и его таблица истинности

щее ИЛИ» с использованием только одного типа элементов, в нашем случае — «И-НЕ». Он более экономичен, т. к. потребует всего одного корпуса типа 561ЛА7. Попробуйте построить таблицу истинности для второго варианта, и вы убедитесь, что он работает «как заказывали» (есть и много других способов). Отметьте, что в первом варианте специальных элементов-инверторов мы не используем, а с целью экономии корпусов микросхем образуем их из элементов «И-НЕ» или «ИЛИ-НЕ» путем объединения входов — обычно так и поступают.

В любой логической серии есть, разумеется, и специальные микросхемы «Исключающее ИЛИ» (561ЛП2). Элемент «Исключающее ИЛИ», помимо способности выдавать сумму одноразрядных чисел, обладает многими интересными свойствами, к которым мы обратимся далее, а пока вернемся к сумматору.

На самом деле одноразрядный сумматор мы уже построили. Его схема приведена на рис. 15.6, *вверху*, а *внизу* на рис. 15.6 мы обозначили все устройство одним квадратиком. Однако почему там написано «полусумматор»? Такой одноразрядный сумматор носит название полусумматора, потому что он не «умеет» одной важной вещи, а именно: разряд переноса-то он выдает, а вот учесть перенос от предыдущего разряда не может. Поэтому, чтобы складывать многоразрядные числа по-настоящему, нужно в каждом разряде поставить по два таких полусумматора, причем объединить их выходы переносов через «ИЛИ» (рис. 15.7).

Так мы получили одноразрядный полный сумматор. Объединением таких сумматоров несложно соорудить устройство для сложения чисел любой разрядности. Если вы попробуете нарисовать схему сумматора, скажем, для восьми разрядов полностью с использованием принципиальных схем логических элементов по рис. 15.1, то ужаснетесь — это же сколько транзисторов надо, чтобы построить такое устройство? Много — в восьмибитовом КМОП-сумматоре их получается 480 штук (а современные микросхемы, бывает, содержат и больше транзисторов). И это без учета

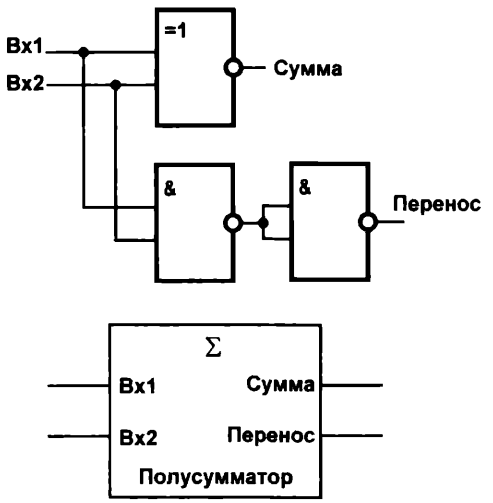


Рис. 15.6. Схема одноразрядного полусумматора

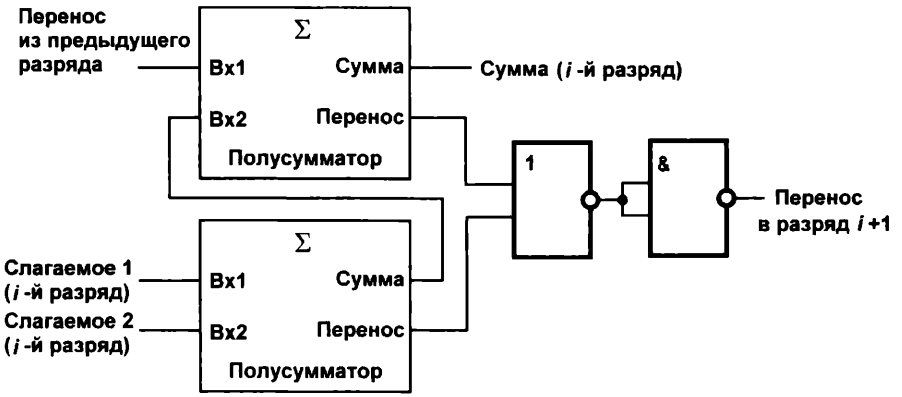


Рис. 15.7. Схема одного разряда многоразрядного сумматора

того, что в систему должны входить еще, как минимум, два регистра для хранения исходных чисел и результата (в целях экономии результат помещают в регистр одного из слагаемых, тем самым слагаемое это уничтожая), а также другие логические схемы (для сдвига, для инверсии битов при манипуляции с отрицательными числами). То есть общее количество транзисторов составляет порядка тысячи.

Теперь понятно, почему микросхемы высокой степени интеграции содержат миллионы транзисторов и почему проблема отвода тепла стоит так остро! Один логический элемент КМОП из четырех транзисторов выделяет, согласно рис. 15.2, при частоте в единицы мегагерц всего-навсего полмилливатта тепла, но что будет, если таких элементов приходится ставить в количестве миллион штук? И еще при этом как можно больше повышать рабочую частоту?

Сумматоры, построенные по описанной схеме, выпускают, естественно, в интегральном исполнении (в «классической» КМОП это микросхема четырехразрядного сумматора 561ИМ1, есть и схема более универсального АЛУ — 561ИП3). В связи с многоразрядным сумматором возникает только один вопрос — а что делать с входом переноса самого первого, младшего разряда? Если мы просто складываем двоичные числа разрядности, соответствующей возможностям нашего сумматора (например, восьмиразрядные, т. е. длиной в один байт), то вход переноса младшего разряда присоединяется к логическому нулю. Соответственно, выход переноса старшего разряда остается «висеть в воздухе». Легко сообразить, что при этом, если складываются числа, в сумме составляющие более величины диапазона, старший разряд суммы «потеряется». Это явление при всей своей очевидности стоит того, чтобы рассмотреть его подробнее.

Предположим, у нас есть такой байтовый сумматор, и под числами мы имеем в виду обычные беззнаковые положительные числа, диапазон которых составит 0–255. Если мы сложим 128 (1000 0000) и 128 (1000 0000), то получим число 256 (1 0000 0000), которое имеет единицу в 9-м, отсутствующем у нас, разряде (заодно отметим этот результат как хорошую иллюстрацию к положению, гласящему, что умножение на 2 есть просто сдвиг всех разрядов влево на одну позицию). Таким образом, в разрядах сумматора мы получаем одни нули, что, конечно, есть резуль-

тат некорректный. Для корректного сложения, к примеру, восьмибитовых чисел нам надо иметь 9 разрядов результата.

А как наращивать разряды, если, например, в микроконтроллере все регистры восьмиразрядные? Да очень просто — надо взять два таких регистра и соединить выход переноса одного со входом переноса другого. Тогда мы получим двухбайтовое число (или *слово*, как его чаще называют). В 8-разрядных микропроцессорах, в том числе и в микроконтроллерах, мы, конечно, физически такое объединение сделать не можем — схема уже создана заранее, но в них зато есть специальный отдельный бит переноса (carry bit), в который автоматически помещается перенос в результате операций сложения, умножения и, кстати, вычитания и деления тоже.

Обработка двоичных сигналов с помощью логических элементов

В начале главы мы упоминали, что логические элементы носят еще название вентилей. На самом деле вентиль — это устройство для регулирования потока жидкости или газа. Каким же образом оправдано это название в приложении к нашим схемам? Оказывается, если на один из входов логического элемента подавать последовательность прямоугольных импульсов (некую аналогию потока), а на другой — логические уровни, то в этом случае элемент будет себя вести совершенно аналогично вентилю настоящему.

Соответствующие диаграммы показаны на рис. 15.8, *а*. Из них вытекают следующие правила:

- для элемента «И-НЕ» логический уровень 1 является «разрешающим», т. е. в этом случае последовательность на другом входе пропускается на выход без изменения (за исключением того, что она инвертируется, поскольку элемент у нас «И-НЕ», а не просто «И»). При логическом уровне 0 вентиль запирается, на выходе будет логическая 1;
- для элемента «ИЛИ-НЕ» ситуация полностью противоположна: «разрешающим» является логический уровень 0, т. е. в этом случае последовательность на другом входе пропускается на выход (также с инверсией). При логическом уровне 1 вентиль запирается, на выходе будет логический 0;
- для «Исключающего ИЛИ» все еще интересней — в зависимости от того, 0 на входе или 1, относительно другого входа элемент ведет себя, соответственно, как повторитель или как инвертор, что дает довольно широкие возможности для управления двоичными последовательностями.

На рис. 15.8, *б* показана интересная схема на основе элемента «Исключающее ИЛИ». Эта схема устраняет неизбежный дребезг механических контактов, который может вызвать (более того, вызывает обязательно) многократное срабатывание некоторых электронных схем — например, триггеров или счетчиков. При наличии свободного элемента «Исключающее ИЛИ» устранить дребезг, как видите, очень просто. Чтобы понять, как это работает, надо учесть, что подвижные контакты

кнопки, тумблера или реле *никогда* не пролетают несколько раз расстояние от одного неподвижного контакта до другого — подвижный контакт только несколько (иногда до нескольких десятков) раз за короткое время оказывается «висящим в воздухе» (представьте себе, что он как бы подпрыгивает на неподвижном контакте, причем как при размыкании, так и при замыкании). При этом подачи напряжения, соответствующего *противоположному* логическому уровню, не происходит.

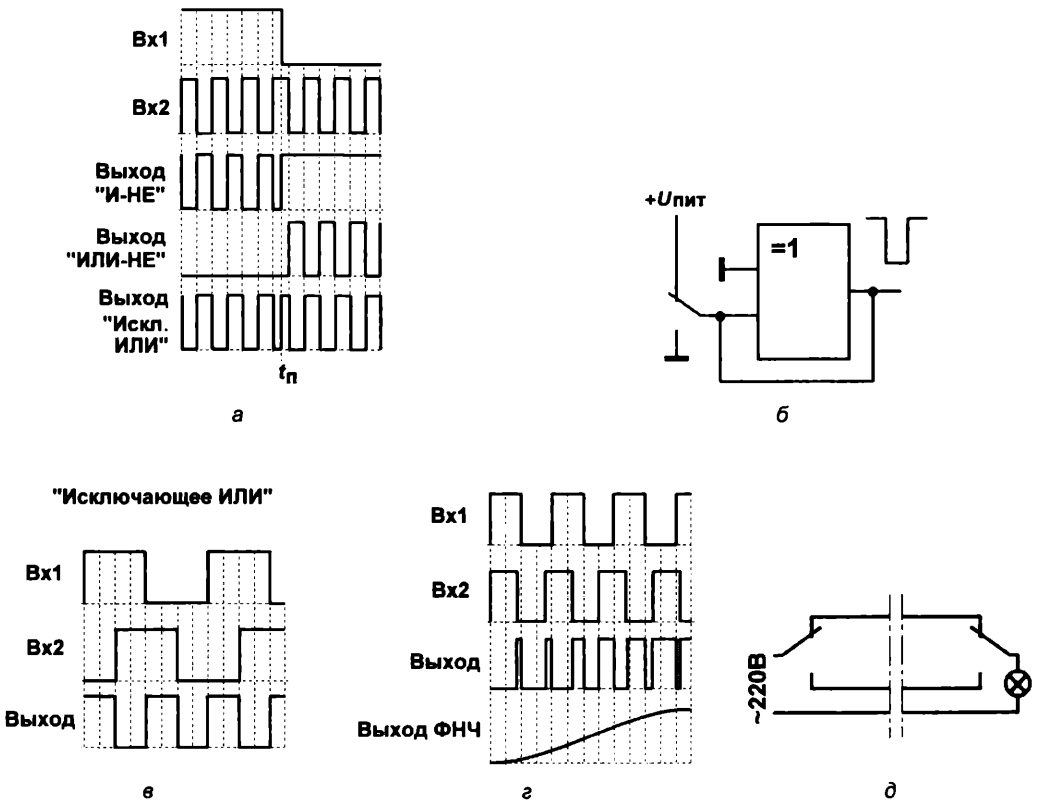


Рис. 15.8. Обработка цифровых сигналов при помощи логических элементов:
а — диаграммы прохождения сигналов через основные типы логических элементов;
б — «анитидребезг» на основе элемента «Исключающее ИЛИ»; *в* и *г* — использование элемента «Исключающее ИЛИ» для выявления разности фаз (*в*) и частот (*г*) сигналов;
д — логический элемент «Исключающее ИЛИ» на двух переключателях

В этих условиях на схеме рис. 15.8, *б* происходит следующее: при наличии 0 на одном из входов элемент «Исключающее ИЛИ» работает как повторитель. Если контакт был замкнут (надежно) с потенциалом питания (логической единицей), то на выходе будет также единица. Когда контакт в процессе дребезга разомкнется и «повиснет в воздухе», то потенциал на выходе все равно останется равным единице, т. к. поддерживается обратной связью, замыкающей выход со входом. Сколько бы контакт ни дребезжал таким образом, потенциал останется равным единице до первого касания контактом «земли», когда элемент перебросится в другое состояние и будет в нем пребывать опять-таки независимо от того, дребезжит контакт или

нет. Разумеется, можно и инвертировать сигнал, если присоединить второй вход к питанию, а не к «земле». Заметьте, что в схеме по рис. 15.8, *б* обязательно требуется именно перекидной контакт — для простой кнопки с двумя выводами нужно использовать иные способы антидребезга, и мы их еще будем рассматривать.

Однако самое интересное будет, если на входы «Исключающего ИЛИ» подать две последовательности импульсов с разными частотами и/или фазами. На рис. 15.8, *в* показано, что произойдет, если обе последовательности имеют одинаковую частоту, но фазы при этом сдвинуты на полпериода. На выходе при этом возникнет колебание с удвоенной частотой! Попробуйте изменить фазу — вы увидите, что скважность результирующего колебания будет меняться, пока фазы не совпадут, и тогда сигнал на выходе исчезнет, — одинаковые состояния выходов дают на выходе «Исключающего ИЛИ» всегда логический ноль. Это позволяет использовать такой элемент в качестве «фазового компаратора», что широко используется в фазовых модуляторах и демодуляторах сигнала.

Не менее интересный случай показан на рис. 15.8, *г* — здесь на входы подаются последовательности с *различающейся* частотой. Мы видим, что на выходе возникнет сигнал с изменяющейся скважностью, причем легко показать, что период изменения этой скважности от минимума к максимуму и обратно будет в точности равен периоду сигнала с частотой, равной *разности* исходных частот. Если при этом поставить на выходе элемента фильтр низкой частоты (если разность частот невелика в сравнении с исходными частотами, то достаточно простой *RC*-цепочки), то мы получим синусоидальное колебание с частотой, равной этой разности! Это колебание можно как-то использовать или, например, можно его подать в качестве сигнала обратной связи на генератор, управляемый напряжением (ГУН), который тогда изменит частоту одного из сигналов так, чтобы она в точности совпадала со второй (опорной). Таким образом, например, делают схемы умножителей частоты, получая целый набор точных частот с использованием одного-единственного опорного кварцевого генератора.

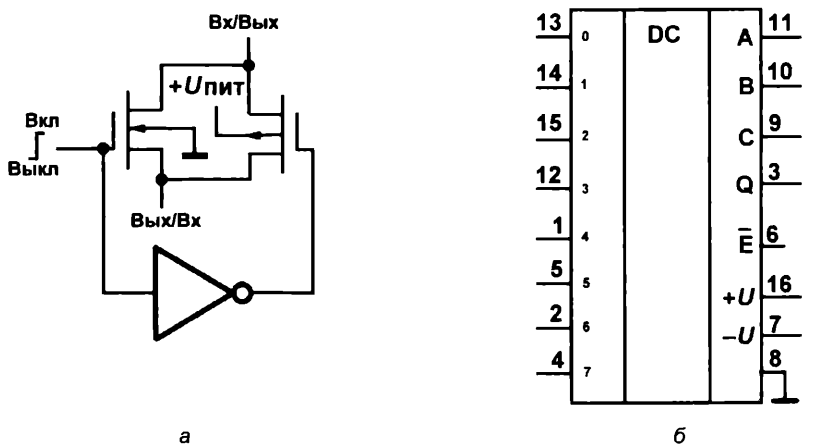
Наконец, на рис. 15.8, *д* показана очень простая, но полезная схема, которая реализует функцию «Исключающее ИЛИ» на двух выключателях с перекидными контактами. Если выключатели в этой схеме находятся в одном положении, то лампочка горит, если в противоположных — выключена. Если лампочка находится в прихожей, то один из выключателей располагается при входе с улицы, а другой — при выходе во внутренние помещения. Заходя в прихожую, вы включаете свет одним выключателем, покинув ее — выключаете либо вторым, либо тем же самым (смотря в какую сторону уходите), причем независимо от того, в какой последовательности это происходит. К сожалению, бытовые выключатели почти всегда имеют одну пару контактов, но некоторые клавишные конструкции несложно доработать так, чтобы они стали перекидными.

Другие, не менее интересные применения логических функций мы рассмотрим в следующих главах, а пока остановимся еще на одной важной разновидности логических элементов.

Мультиплексоры/демультиплексоры и ключи

Мультиплексоры/демультиплексоры — важный класс логических схем малой степени интеграции. Их довольно часто применяют и в современных схемах совместно с микроконтроллерами — для сокращения числа необходимых соединений. *Мультиплексором* называют схему, которая соединяет единственный входной вывод напрямую с одним из нескольких выходных (как правило, четырех или восьми), в зависимости от поданного на нее двоичного кода (схема «1 → 8»). Соответственно, *демультиплексор* осуществляет обратную операцию — пропускает сигнал с одного из нескольких выводов на единственный выходной (схема «8 → 1»). Фишка состоит в том, что в КМОП-версии они прекрасно коммутируют не только цифровые, но и аналоговые сигналы, причем в обе стороны!

Такие мультиплексоры/демультиплексоры делают на *ключах* — специальным образом включенных полевых транзисторах по технологии КМОП. Простейший такой ключ изображен на рис. 15.9, *а*. Выпускаются также и микросхемы, содержащие просто наборы отдельных ключей, — например, 590КН2 и аналогичные, мы еще с ними столкнемся. Такие ключи широко используются в составе микросхем средней и большой степени интеграции — в аналого-цифровых и цифроаналоговых преобразователях, например. Также они практически заменили механические переключатели в коммутаторах телевизионных каналов, используются в цифровых переменных резисторах, электронных реле и т. д.



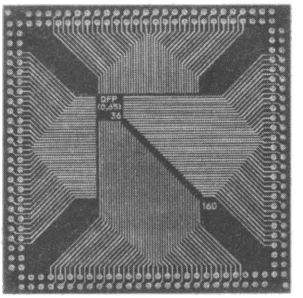
а *б*

Рис. 15.9. Использование КМОП-ключей: *а* — простейший униполярный ключ; *б* — разводка выводов мультиплексора/демультиплексора 561КП2

На рис. 15.9, *б* приведена для примера схема разводки выводов микросхемы 561КП2, которая представляет собой восьмиканальный мультиплексор/демультиплексор (561КП1 делает то же самое, но содержит два четырехканальных мультиплексора). Эта микросхема коммутирует один из выводов, обозначенных как 0–7, к выводу Q, в зависимости от поданного на управляющие входы A–C двоичного кода. Очень важную функцию осуществляет вход E (с инверсией, т. е. активный

уровень на нем — низкий) — это вход разрешения, и если на нем присутствует высокий уровень, то все каналы размыкаются.

Специально для коммутации переменных аналоговых сигналов у 561КП2 предусмотрено подключение отрицательного питания (вывод 7), в случае цифровых же сигналов этот вывод коммутируется просто на «землю». Размах питания между выводами 7 и 16 не может превышать предельно допустимого для однополярного питания 561-й серии значения 15–18 В, т. е. двухполярное питание возможно примерно до ±8 В. Однако уровень сигнала управления (как по входам A–C, так и по E) при этом отсчитывается от «цифровой земли», которая установлена потенциалом вывода 8. При этом аналоговый сигнал по амплитуде может достигать почти значений питания, только для получения минимума искажений коммутируемые токи также должны быть малы.



ГЛАВА 16

Устройства на логических схемах

Мультивибраторы, формирователи, триггеры, счетчики...

> Сердце молодого гасконца билось так сильно, что готово было разорвать ему грудь. Видит бог, не от страха — он и тени страха не испытывал — а от возбуждения.
>
> *А. Дюма. «Три мушкетера»*

Из описания устройства логических элементов, приведенного в *главе 15*, ясно, что любой логический элемент есть в сущности не что иное, как усилитель. Когда-то радиолюбители даже пытались применять логические микросхемы в качестве аналоговых усилителей. В самом деле, с формальной точки зрения между простым многокаскадным усилителем без обратной связи и логическим инвертором разницы нет никакой. Правда, аналоговым усилителем логический элемент будет очень плохим — коэффициент усиления по напряжению у КМОП-элементов составляет всего несколько десятков, в отличие от сотен тысяч и миллионов у операционных усилителей и компараторов, и даже введение обратной связи не поможет получить качественный сигнал. Если кого-то интересует такое экзотическое использование логических микросхем, то в упоминавшейся книге [18] есть схема линейного усилителя на КМОП-элементах, — можете поэкспериментировать.

Но зато логические микросхемы идеально приспособлены для работы в схемах, так сказать, «полуаналоговых» — т. е. в схемах генераторов, формирователей и преобразователей импульсов. Ими мы сначала и займемся.

Генераторы

До сих пор мы рассматривали только два способа построения генераторов колебаний: один раз это был релаксационный генератор коротких импульсов на однопереходном транзисторе (см. рис. 10.3) для фазового управления тиристорами, второй раз — аналоговый генератор синусоидальных колебаний на ОУ (см. рис. 12.6). Был еще «зуммер» из реле, приведенный на рис. 7.3, *в*. Здесь же мы рассмотрим релаксационные генераторы прямоугольных импульсов на логических микросхемах.

Подробности

Релаксационными, в отличие от гармонических, называются колебания в системах, где существенную роль играет рассеяние энергии, или, как говорят физики, ее дисси-

пация. Типичными примерами систем с гармоническими колебаниями служат описанные в любом школьном учебнике физики колебательный контур или механический маятник. В них энергия непрерывно переходит из одной формы в другую, и если не учитывать потери на нагревание проводов в контуре или на трение в маятнике, то эти колебания могут продолжаться бесконечно безо всякой подпитки извне. В отличие от таких систем, релаксационные генераторы без внешнего источника неработоспособны, в них энергия, запасенная в накопителе (например, конденсаторе), не переходит в другую форму, а теряется — переходит в тепло. Для возникновения релаксационных колебаний обязательно требуется наличие нелинейного порогового элемента, меняющего свое состояние скачком, а также определенный характер обратных связей (о чем далее). Релаксационные генераторы обычно выдают скачкообразный сигнал (прямоугольный, как в большинстве генераторов, описанных далее, или импульсный, как в генераторе на однопереходном транзисторе), но не всегда. Так, генератор синусоидальных колебаний из *главы 12* также является релаксационным, но с помощью хитро подобранных характеристик цепей обратной связи сделано так, что форма сигнала меняется по синусоидальному закону.

Но сначала рассмотрим такой генератор на ОУ (рис. 16.1, *а*). Работает он следующим образом. Мы помним, что в первый момент времени заряжающийся конденсатор эквивалентен короткозамкнутой цепи. Поэтому после включения питания коэффициент усиления по инвертирующему входу окажется равен бесконечности, и на выходе ОУ будет фактически положительное напряжение питания. Конденсатор начнет заряжаться через резистор R1, но в силу большого коэффициента усиления ОУ напряжение на выходе останется вблизи напряжения питания, пока потенциал на конденсаторе не достигнет порога, заданного делителем R2/R3, — в данном случае половины положительного напряжения питания. Тогда выход ОУ скачком перебросится в состояние, близкое к отрицательному напряжению питания, и конденсатор начнет разряжаться через тот же резистор R1. Напряжение на неинвертирующем входе станет равным половине отрицательного напряжения питания, и, чтобы привести схему в первоначальное состояние, конденсатору придется перезарядиться до этого напряжения. Затем все повторится сначала. Таким образом, на выходе мы получим меандр с периодом, который определяется параметрами *RC*-цепочки (см. формулу на рис. 16.1, *а*). На инвертирующем входе, между прочим, при этом будет напряжение, очень близкое к треугольной форме, которое можно где-нибудь использовать, если подключить потребителя через отдельный развязывающий повторитель на другом ОУ.

На схеме рис. 16.1, *б* приведена классическая схема генератора прямоугольных импульсов (мультивибратора) на транзисторах, которая ранее разбиралась в каждом учебнике по электронике, а сейчас почти позабыта ввиду распространения микросхем (одно из возможных современных применений этой схемы описано в начале *главы 22*, в разделе о ШИМ). Но вообще-то она довольно познавательна, и может служить иллюстрацией к тому, что мы обрели и потеряли с появлением интегральной электроники. Схема хороша тем, что крайне неприхотлива: работает с любыми типами транзисторов, и притом в очень широких пределах изменения емкости конденсаторов C1 и C2 и сопротивления резисторов R2 и R3, которые задают частоту. Для конденсаторов теоретических ограничений нет — лишь минимальная их емкость ограничена естественной паразитной емкостью монтажа и быстродействием транзисторов. А для базовых резисторов R2 и R3 требования такие: величина каж-

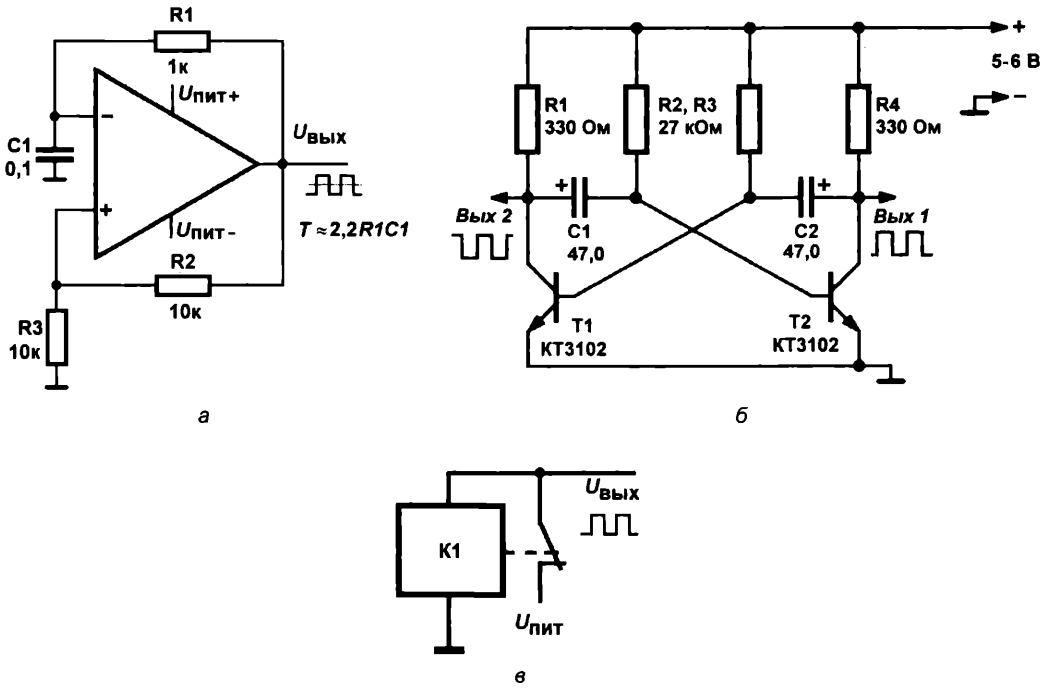


Рис. 16.1. Схема генератора на ОУ (*а*), классического мультивибратора на транзисторах (*б*) и зуммера на реле (*в*)

дого из них должна быть больше соответствующего коллекторного резистора (R1 или R4), и меньше, чем величина последнего, умноженная на коэффициент усиления транзистора $h_{21э}$.

Длительность каждой половины периода $T½ \approx 0{,}7 \cdot RC$, где R — величина базового резистора, C — емкость конденсатора в том же плече. (Обратите внимание, что длительность на выходе 1 задает произведение R2·C1, а на выходе 2 — R3·C2). При указанных на схеме параметрах длительность эта составит около 1 секунды, т. е. частота генерации будет 0,5 Гц. Схема делает то же самое, что и остальные схемы генераторов в этом разделе, за одним исключением: она имеет два симметричных (инвертированных относительно друг друга) выхода, длительность импульса по каждому из которых можно менять в очень широких пределах, притом совершенно независимо от другого. При соблюдении соотношений, указанных выше, величина коллекторных нагрузок R1 и R4 не влияет на отсчет времени и потому здесь может быть сразу поставлено, например, маломощное электромагнитное реле без дополнительных усилительных каскадов.

Недостаток у этой схемы только один, о чем нередко забывали даже авторы официальных пособий тех времен: т. к. на базах транзисторов в момент переключения возникают обратные импульсы с размахом, практически равным напряжению питания (см. свойства дифференцирующей цепочки в *главе 5*), то напряжение питания у этой схемы не может превышать 5–6 вольт, поскольку предельно допустимое обратное напряжение эмиттер-база у *всех* кремниевых транзисторов не превышает

5 вольт (см. *главу 6*). Забывчивость тех авторов понятна и извинительна: изобреталась эта схема во времена, когда в полупроводниках доминировал германий, а у германиевых транзисторов допустимое обратное напряжение на базовом переходе выше на порядок и обычно не отличается от допустимого напряжения эмиттер-коллектор. А вот тот факт, что эта ошибка повторяется не менее, чем в каждом втором случае рассмотрения этой схемы на современных интернет-ресурсах, конечно, непростителен. Причем ограничивать отрицательные выбросы на базах с помощью диодов, как это делается для входов микросхем (см. *главу 11*) или на коллекторах (стоках) мощных транзисторных ключей при индуктивных нагрузках (см. *главу 5*), здесь нельзя, потому что длительность этих выбросов и задает время переключения (подробнее см. [4]).

Теперь посмотрим, что нужно сделать, чтобы построить такой генератор на логике. Сначала обратимся к зуммеру, приведенному на рис. 7.3, *в*, и перерисуем его, как показано на рис. 16.1, *в*. В таком виде в схеме легко узнать релейный инвертор (см. рис. 14.3, крайний элемент *справа*), у которого в данном случае выход управляет входом. Не получится ли выполнить тот же самый фокус, если замкнуть вход с выходом у обычного инвертора в интегральном исполнении? К сожалению, нет — такое включение просто выведет инвертор в линейный режим, при котором на выходе установится половина питания. А почему? А потому, что логические элементы, грубо говоря, слишком быстродействующие.

Теория гласит, что для получения устойчивых колебаний необходимо, чтобы присутствовали обе разновидности обратной связи, причем действие отрицательной обратной связи (ООС) должно отставать от действия положительной (ПОС). Именно это и происходит и в схеме генератора на основе компаратора — за счет использования *RC*-цепочки, и в зуммере — за счет механической инерции деталей. Действие одной только ПОС приведет к тому, что выход устройства «зависнет» в одном из крайних положений, а одной только ООС — к тому, что на выходе установится некое среднее состояние равновесия. Сравните поведение одновибраторов, рассмотренных в этой главе далее, в которых наличествует только ООС, и RS-триггеров (в конце главы), в которых присутствует только ПОС. А вот вместе они дадут то, что надо.

Существует огромное количество схем *мультивибраторов* — т. е. генераторов прямоугольных колебаний, реализующих эти теоретические положения. Если кому любопытно, то не менее десятка разнообразных схем можно найти только в одной книге [11], и этим их многообразие далеко не исчерпывается. Вариант простейшей схемы на основе т. н. «триггера Шмидта» будет показан далее, а здесь я приведу одну из самых удобных схем, выбранную из многих из-за минимального количества задействованных компонентов, и два ее варианта, разница между которыми заключается в используемых элементах («И-НЕ» или «ИЛИ-НЕ»).

Схема на рис. 16.2, *а* базовая. При включении питания она начинает работать сразу и, как и остальные схемы подобного рода, выдает меандр с размахом от 0 до $U_{пит}$. Частота на выходе определяется параметрами R1 и C1: период $T \approx 2R1 \cdot C1$. Схема устойчиво работает при величине резистора R1 от нескольких килоом до 10 МОм, что составляет достаточный диапазон для того, чтобы избежать искушения при ма-

лых частотах использовать электролитические конденсаторы, — напомним, что они очень нестабильны при работе во времязадающих цепях.

Резистор R2 в работе схемы почти не участвует и нужен только для того, чтобы оградить защитные диоды микросхемы от перегрузки током разряда конденсатора C1. Величина его может изменяться от сотен ом до нескольких килоом при условии, что он много меньше R1. Его можно и вообще исключить из схемы, отчего он показан пунктиром (о необходимости установки этого резистора мы будем говорить позже). Конденсатор C1 может применяться любой, с емкостью не меньшей нескольких десятков пикофарад. Указанные параметры элементов позволяют получить частоты от сотых долей герца вплоть до верхней границы рабочей частоты «классических» КМОП-микросхем в 1–2 МГц. Для получения более высоких частот целесообразно использовать быстродействующие серии КМОП, а не ТТЛ, т. к. для последней ограничения гораздо жестче, — например, резистор R1 не должен выходить за пределы 0,5–2 кОм.

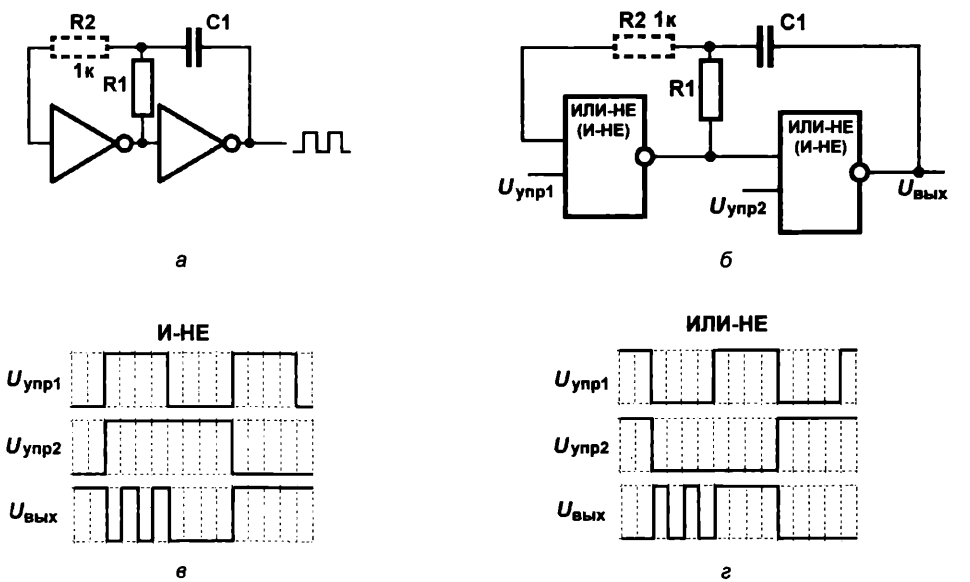


Рис. 16.2. Схемы мультивибратора на логических элементах: *а* — базовая схема на инверторах; *б* — схема на двухвходовых элементах с управлением; *в* — диаграмма состояний схемы на двухвходовых элементах «И-НЕ»; *г* — диаграмма состояний схемы на двухвходовых элементах «ИЛИ-НЕ»

Если в схеме на рис. 16.2, *б* объединить входы логических элементов между собой, она превратится в схему на рис. 16.2, *а*. Но дополнительные входы можно использовать и для управления генерацией. Нередко возникает потребность остановить генерацию на время и при этом обеспечить определенный логический уровень на выходе генератора. Эти задачи как раз и решаются с помощью дополнительных входов. Диаграммы состояния выхода в зависимости от состояния входов при использовании разных типов логических элементов приведены на рис. 16.2, *в* и *г*. Запоминать эти диаграммы нет необходимости, если обратиться к рис. 15.8. Из него

следует, что единица на входе «И-НЕ» и ноль на входе «ИЛИ-НЕ» являются разрешающими уровнями, следовательно, при этих уровнях на управляющих входах наша схема будет функционировать, как если бы входы элемента были объединены. При запрещающих же уровнях на входе уровень на выходе будет устанавливаться так, как если бы никаких *RC*-цепочек не существовало.

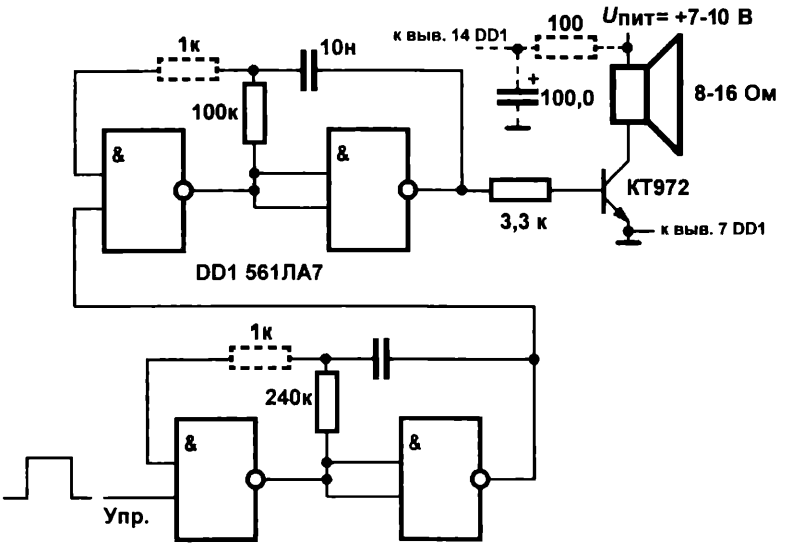


Рис. 16.3. Схема звуковой сигнализации с динамиком на выходе

Простейшее применение схемы с управлением — решение задачи приостановки генератора на время переходных процессов при включении питания, для чего по управляющему входу нужно поставить интегрирующую *RC*-цепочку, как в схеме триггеров с предустановкой (см. далее рис. 16.9). Другое применение — генерация пачек импульсов с меньшей частотой, если управляющий вход одного генератора присоединить к выходу другого. На рис. 16.3 показана схема звуковой сигнализации на одной микросхеме 561ЛА7 и одном транзисторе. Это пример случая, когда требуется определенный логический уровень при выключенной генерации, чтобы избежать протекания постоянного тока через динамик и не ставить при этом разделительный конденсатор.

Схема выдает сигнал около 500 Гц с периодом повторения около 0,5 с, если на управляющий вход подать сигнал высокого уровня. При сигнале низкого уровня на управляющем входе, на выходе будет также низкий уровень, и постоянный ток через динамик не потечет. Транзисторный каскад лучше питать нестабилизированным напряжением от входа стабилизатора питания микросхем, потому что тогда достаточно мощные импульсы тока через динамик будут фильтроваться стабилизатором и не окажут вредного воздействия на остальные элементы схемы. Динамик можно заменить и на пьезоэлектрический звуковой излучатель, тогда мощный транзистор ставить необязательно (но вовсе без транзистора не обойтись, поскольку звук будет слишком тихим). А о пьезоэффекте мы подробнее поговорим далее.

Схемы на основе триггера Шмидта

Если на рис. 16.1, *а* исключить из рассмотрения интегрирующую цепочку R1C1, то остальная часть схемы есть упрощенный вариант компаратора с гистерезисом, приведенного на рис. 12.10. Для того чтобы генератор работал от одного напряжения питания, придется неинвертирующий вход подключить в точности так же, как там — к искусственной средней точке. Но можно и ввести гистерезис прямо в цифровую микросхему, что реализовано в очень удобной конструкции, называемой *триггером Шмидта*. Существует известная простая схема триггера Шмидта на двух инверторах с положительной обратной связью на двух резисторах, но я ее здесь даже не привожу — практически она совершенно бесполезна, потому что ее входное сопротивление равно сумме этих резисторов, тогда как большинство применений триггера Шмидта подразумевают, что его входное сопротивление достаточно велико.

Существуют несколько модификаций этой полезной схемы в интегральном исполнении, самая удобная из которых — шестиэлементный КМОП-вариант одновходовых элементов 74C14 с диапазоном питания от 3 до 15 вольт. Прямого аналога этой микросхемы в отечественном варианте не имеется. Предельная рабочая частота для 74C14 составляет, как и для обычной КМОП, порядка мегагерца. Есть и более быстродействующие аналоги 74HC14 (1564ТЛ2) и 74AC14 (1554ТЛ2) с ограниченным, однако, напряжением питания (от 2 до 6 вольт). В «классической» КМОП есть также набор двухходовых триггеров Шмидта под названием CD4093, имеющий широкий диапазон напряжений питания от 3 до 18 вольт (отечественный аналог К561ТЛ1).

Схема базового генератора (рис. 16.4, *а*) на одновходовом триггере Шмидта предельно проста и состоит всего из трех компонентов. На выходе микросхемы мы имеем меандр со скважностью, практически равной 2, и размахом в величину напряжения питания. Некоторое отклонение скважности от двойки может наблюдаться вследствие разброса и несимметричности верхнего и нижнего уровней порогов гистерезиса, обычно составляющих примерно по трети напряжения питания сверху (от $+U_{пит}$) и снизу (от 0 вольт — см. уровни $U+$ и $U-$ на рис. 16.4, *б*). Период колебаний можно подсчитать по формуле $T \approx 1{,}7 \cdot R1 \cdot C1$.

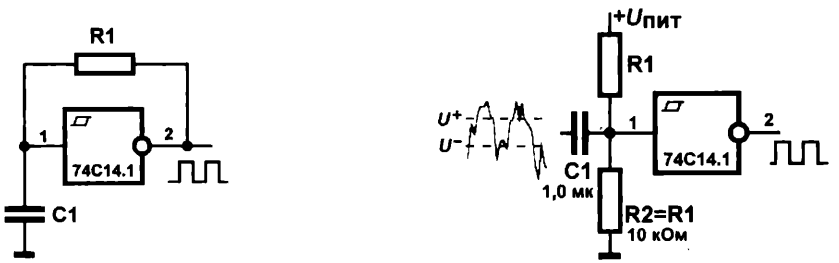


Рис. 16.4. Схемы на одновходовом триггере Шмидта: простой генератор (*а*) и формирователь прямоугольных импульсов из колебаний произвольной формы (*б*)

На рис. 16.4, *б* показана очень необходимая в некоторых случаях схема — формирователь прямоугольных импульсов из колебаний произвольной формы, в том числе синусоидальных или очень зашумленных. Такие колебания могут быть получены, например, на выходе электромагнитного датчика оборотов и в других случаях, когда источник частотного сигнала не имеет цифровой природы. Подобные формирователи приходится ставить, например, на входе частотомера или счетчика элементарных частиц (дозиметра). Разумеется, подобную схему можно создать и на основе простого ОУ и многими другими способами, но с помощью триггера Шмидта она получается наиболее компактной и с минимумом навесных элементов. На основе триггера Шмидта можно также сделать, например, надежный датчик освещенности (фотореле) — т. к. естественная освещенность обычно меняется крайне медленно, то защита от дребезга там даже более необходима, чем в случае термостата, рассмотренного в *главе 12*.

Резисторы R1 и R2 могут быть в пределах от единиц до сотен килоом (большее значение, если источник сигнала маломощный). Если исходный сигнал выходит по амплитуде за пределы напряжения питания (как чаще всего и бывает на практике), то между источником сигнала и конденсатором C1 необходимо установить резистор в несколько сотен ом, а после конденсатора параллельно резистивному делителю — цепочку защитных диодов по питанию, как показано на рис. 11.4. Диоды могут быть любые маломощные импульсные — например, 1N4148 или аналогичные.

Триггер Шмидта можно задействовать в некоторых случаях и для подавления дребезга контактов — например, если у вас в схеме уже используется триггер Шмидта, то наверняка в корпусе останутся свободные элементы. Вот тогда и целесообразно попробовать подключить к этим элементам кнопки, если их использование необходимо. Кнопка подключается, например, к «земле» и ко входу свободного триггера, и тот же вход «притягивается» резистором к питанию (точно так же, как это делается для одновибраторов, — см. далее). Для надежности можно попробовать в этом случае дополнительно зашунтировать кнопку конденсатором емкостью в 10–100 нФ (но не пробуйте это делать для одновибраторов!). Учтите, однако, что эффективное подавление дребезга в такой схеме гарантируется не для всех конструкций кнопок.

Кварцевые генераторы

Точность поддержания частоты в приведенных ранее схемах невысока. Частота «уходит» примерно на 10–20% при изменении напряжения питания от 5 до 15 В и в достаточно большой степени зависит от температуры (использование высокостабильных резисторов и конденсаторов не поможет, и потому нецелесообразно). Чтобы избавиться от этого эффекта, необходимо применить *кварцевый резонатор*, в просторечии — просто кварц.

Здесь не место для того, чтобы подробно излагать принципы работы кварцевого (или реже употребляемого керамического, который обладает несколько меньшей стабильностью) резонатора — это нужно делать в курсе радиотехники в сравнении со свойствами колебательного контура. Вкратце дело заключается в следующем:

если приложить напряжение к кварцевому параллелепипеду, выпиленному из целого кристалла в определенной ориентации относительно его осей, то кристалл деформируется — очень не намного, но все же достаточно, чтобы на этом принципе даже делать прецизионные манипуляторы для электронных микроскопов или выталкивающие жидкий краситель поршни в струйных принтерах Epson. Это так называемый *обратный пьезоэлектрический эффект*. Имеет место и противоположный *прямой* эффект — если такой кристалл деформировать, то у него на гранях появляется разность потенциалов, — явление используется в специальных тензометрических кварцах.

Получается, что если мы включим такой кристалл в схему с обратной связью, то она начнет генерировать колебания, причем частота генерации будет зависеть исключительно от размеров кристалла — и ни от чего больше! Как, спросите вы, даже от температуры не будет зависеть? Вот именно — *пьезоэлектриков*, как называют вещества, ведущие себя подобно кварцу, много, но чаще всего используют именно кварц, т. к. он помимо пьезоэлектрических свойств обладает еще и одним из самых низких на свете температурных коэффициентов расширения.

В результате кварцевые генераторы без каких-либо дополнительных ухищрений дают погрешности порядка 10^{-6} долей от номинальной частоты. Такие доли обозначаются как *ppm* (part per million), а иногда просто как 10^{-6}. Температурная нестабильность хороших кварцев не превышает долей или единиц *ppm*. Это значит, что уход часов с таким генератором составляет не более 1 секунды в сутки. Правда, для того чтобы реализовать потенциал кварцевых резонаторов полностью, нужны специальные схемы включения, иногда довольно громоздкие (обычно их делают на дискретных элементах), но и схемы на цифровых инверторах, приводимые далее, дают результат не хуже примерно 10^{-4} во всем диапазоне питающих напряжений и температуры.

На кварцах работают все бытовые электронные часы, и вообще в любом современном бытовом электронном устройстве вы, скорее всего, найдете кварц, а иногда и не один. Кварцы выпускают на определенные частоты, и при их приобретении следует обращать внимание на возможное отклонение частоты от номинальной, которая может составлять от долей *ppm* до десятков и даже сотен *ppm*. Если нужна повышенная точность, то можно приобрести специализированные очень стабильные резонаторы с погрешностью начальной установки до 10^{-7}, выпускаются и готовые генераторы на разные частоты (особенно большой выбор предлагает фирма, название которой обычно ассоциируется совсем с другими продуктами, — Epson, приобретшая в свое время компанию, известную своей часовой торговой маркой Seiko).

Большинство кварцевых генераторов в цифровой технике строят по одной и той же схеме, которая очень проста и требует всего одного инвертора, резистора и двух конденсаторов. Схема эта показана на рис. 16.5, *а*. Чтобы не перегружать выход (это будет влиять на стабильность), нагружать такой генератор можно только на один-два КМОП-входа, поэтому обычно на выходе ставят еще и буферный элемент. Если же частота с выхода подается, например, только на вход КМОП-счетчика, то его можно и не ставить. Параметры всех элементов можно менять в довольно

больших пределах — так, емкость конденсаторов может меняться от 10 до 100 пФ (как рассчитать значение емкости более точно, см. *«Подробности»* далее), причем они не обязательно должны быть одинаковыми, — изменением C1 можно подстраивать частоту в пределах 4–5 знака после запятой. Сопротивление резистора R1 может меняться от 1 до 20 МОм, R2 — от нуля до сотен килоом (меньшие значения получаются при более высокочастотных кварцах). Схема потребляет несколько десятков микроампер при напряжении питания 5 В и устойчиво работает для кварцев с частотами от десятков килогерц до 1 МГц для «классической» КМОП и до 10 МГц для КМОП-элементов из быстродействующих серий. Правда, с некоторыми старыми отечественными кварцами (вроде РК-72) могут быть проблемы — придется подбирать значения резисторов.

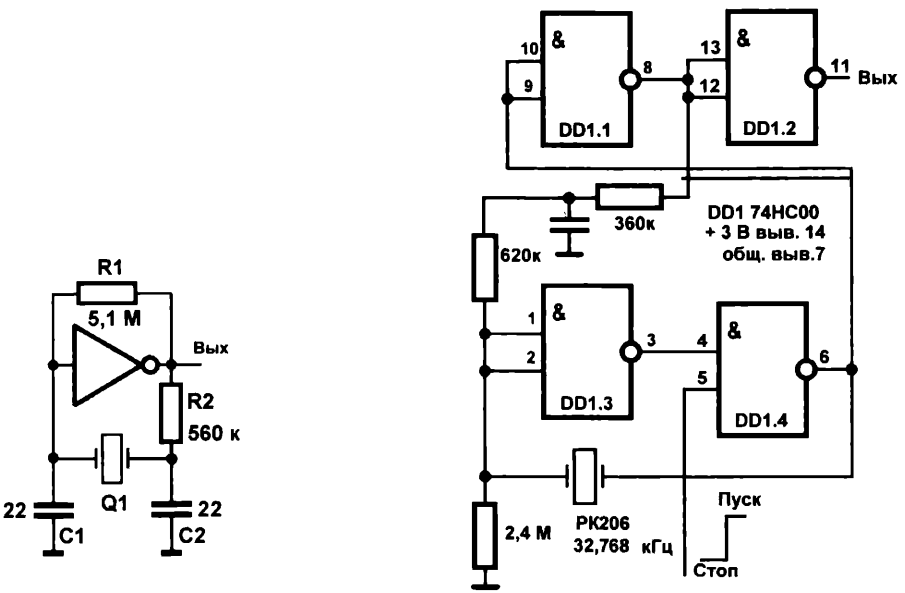


Рис. 16.5. Схемы кварцевых генераторов на КМОП-элементах

Инвертор, естественно, может представлять собой и просто инвертор, и многовходовой логический элемент с объединенными входами. Во втором случае один из входов можно использовать для запуска и остановки генерации, как в схеме на рис. 16.2, *б*.

Подробности

Конденсаторы C1 и C2 в схеме на рис. 16.5, *а* рассчитывают, исходя из номинальной емкости нагрузки $C_н$, которая указывается для каждого кварцевого резонатора. Если она неизвестна, то можно ориентироваться на значение 12,5 пФ для «часовых» кварцев и на значение 16–32 пФ для кварцев частотой 1–16 МГц. Номинал C каждого из конденсаторов (при условии их равенства) можно рассчитать по приблизительной формуле $C = 2C_н - 10$ пФ, где 10 пФ — «среднепотолочное» значение емкости монтажа плюс емкость входа/выхода инвертора. Таким образом, для «часовых» кварцев эти

емкости должны быть порядка 15 пФ, а для более высокочастотных — 22–56 пФ. Для микросхем вроде часов реального времени, где конденсаторы уже имеются в составе микросхемы, указывается номинальная нагрузочная емкость внешнего кварцевого резонатора. Правильно подобранное значение емкости гарантирует более точное соответствие частоты генератора номинальной, но это не значит, что при других значениях емкости генератор не заработает, — чем больше значение емкостей, тем больше и потребляет схема, но и тем быстрее она «заводится». Указанные на схеме значения 22 пФ оптимальны, если использовать резонатор «не глядя».

Кварцевые резонаторы имеют предельно допустимую мощность рассеяния, которая невелика: от 1–3 мкВт для «часовых» кварцев в цилиндрических корпусах 6×2 или 8×3 мм до 30–50 мкВт в низких прямоугольных корпусах (HC-49S) и 1–2 мВт для кварцев в стандартных прямоугольных корпусах типа HC-49U. Превышение допустимой мощности еще не означает выхода резонатора из строя (хотя может случиться и такое — смотря, насколько превысить), но стабильность генератора снижается. Значение рассеиваемой мощности на кристалле W можно грубо прикинуть, исходя из падения напряжения на резонаторе: $W = U_к \cdot I_к$, где $I_к$ — ток через резонатор, который определяется в основном резистором R2. Его величина подсчитывается, исходя из напряжения на выходе инвертора $U_{вых}$: $I_к = U_{вых}/R_2 = U_{пит}/2R_2$ (делитель 2 появляется, т. к. на выходе мы имеем меандр, а не постоянное напряжение). Рассчитать $U_к$, форма которого близка к синусоидальной, непросто, но его можно измерить экспериментально, — для «часового» кварца в схеме на рис. 16.5, *а* действующее значение $U_к$ равно примерно 0,05 от напряжения питания. Итого при номиналах резисторов и конденсаторов, близких к указанным на схеме, мощность на «часовом» резонаторе составляет около 1 мкВт при напряжении питания 5 В и линейно растет с напряжением питания, поэтому при 12–15 вольтах самые миниатюрные кварцы лучше не ставить.

Недостатком схемы на рис. 16.5, *а* является то, что на низких частотах она достаточно долго «заводится» при включении — установление режима для «часового» кварца 32 768 Гц может занимать секунды, в зависимости от значения емкостей, и в это время схема потребляет довольно большой ток — до 15 мА. Этого недостатка лишена более сложная схема на рис. 16.5, *б*, которая, однако, работает только при частотах в десятки килогерц, т. е. ориентирована на «часовые» кварцы. Потребление такой схемы при напряжении питания 3,3 В и использовании указанных на схеме элементов серии 74HC составляет 180 мкА (3 мА в момент включения), а время выхода на режим при включении питания или подаче разрешающего высокого уровня на вход «Пуск/Стоп» не превышает 0,2–0,3 с. При отключении подачей низкого уровня на вход «Пуск/Стоп» схема потребляет меньше 1 мкА. В этой схеме резонатор работает в более щадящем режиме, чем в схеме на рис. 16.5, *а*.

Специально для измерения температуры производятся термочувствительные кварцы, обладающие чувствительностью порядка 50–90 *ppm* изменения частоты на каждый градус изменения температуры. Кварцы эти выпускают на разные частоты: 30–40 кГц, 5 МГц, 10–40 МГц и пр. Если заменить в схеме на рис. 16.5, *б* «часовой» кварц аналогичным термочувствительным (например, отечественным РКТ-206 с частотой 32,7 кГц), то получится отличный малопотребляющий датчик температуры с частотным выходом и отрицательным наклоном зависимости частоты от температуры. Зависимость эту для бытовых применений можно считать линейной, однако для прецизионных измерений температуры (для чего, собственно, такие кварцы и существуют) приходится ее аппроксимировать полиномом 2-й и даже 3-й степени.

Заметки на полях

Кстати, мало кто знает, но в случае, если под рукой нет подходящего кварца, схему на рис. 16.5, *а* вполне можно «завести», просто заменив резонатор малогабаритной индуктивностью. То же относится и к встроенным генераторам микроконтроллеров, которые организуются по аналогичной схеме. Частоту можно грубо прикинуть, если учесть, что постоянная времени LC-контура равна $\sqrt{LC}$, где в качестве величины C нужно подставить сумму емкостей обоих конденсаторов. Тогда частота будет примерно равна единице, деленной на удвоенную величину этой постоянной. Естественно, главное преимущество кварца — высокая стабильность — при этом пропадет, зато можно менять частоту, в том числе и плавно. В приложении к микросхемам малой интеграции такой прием кажется не имеющим особого практического смысла, но может пригодиться, если вдруг возникает задача плавно менять тактовую частоту генератора микроконтроллера, который устроен аналогичным образом.

Формирователи импульсов

К формирователям импульсов, строго говоря, следует отнести и устройство, показанное на рис. 16.4, *б*. Здесь, однако, пойдет речь о формирователях в несколько ином смысле — мы попытается ответить на вопрос, как получить из той или иной последовательности импульсов сигнал определенной формы или длительности.

Все приведенные схемы генераторов выдают меандр, в котором длительность паузы приблизительно равна длительности импульса, т. е. скважность их равна примерно двум (на величину скважности влияет и величина резистора R2 — см. схемы на рис. 16.2). Но нам могут потребоваться симметричные импульсы со скважностью, равной двум с большой точностью или вообще с другим значением скважности. На рис. 16.6 показана схема, которая формирует импульсы со скважностью ровно 2 и 4 из исходного сигнала с любой скважностью. В ней используется делитель частоты на два (*счетный триггер*) — элемент, который мы еще «не про-

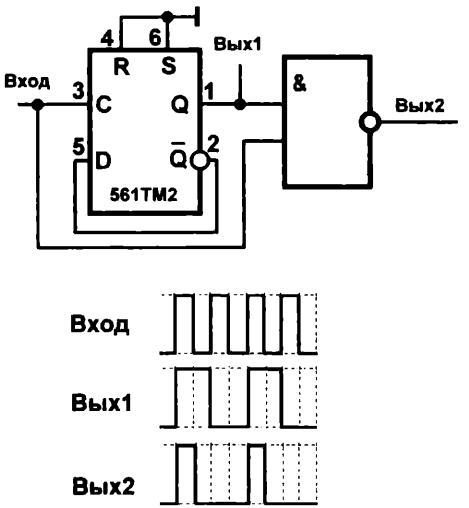


Рис. 16.6. Схема формирователя последовательности со скважностью 2 и 4

ходили», но будем рассматривать далее, а пока он приводится без пояснений. Диаграммы выходного напряжения показаны на рис. 16.6, *внизу*.

Следует отметить, что за счет задержки сигнала в триггере в момент, соответствующий фронту второго по счету сигнала исходной последовательности, на втором выходе может возникнуть короткая «иголка», т. к. спад импульса на выходе триггера наступит несколько позже наступления этого фронта. Она не страшна для статических схем (например, дешифраторов с выводом на индикаторы) или для управления внешними достаточно инерционными устройствами, но может вызвать срабатывание другого триггера или одновибратора (см. далее), если к его входу подключить выход такой схемы. Если это критично, то в подобных схемах вместо простого счетного триггера обычно используют специальные синхронные счетчики (о них также далее). Разумеется, если требуется только симметричный меандр, то одного триггера достаточно, и элемент «И-НЕ» можно исключить.

Микросхема 561ТМ2 (CD4013) содержит два триггера, поэтому схему легко дополнить, получив на выходе другие значения частоты и скважности. Применяя дополнительные логические элементы, можно получить 4 выхода, на каждом из которых фаза сдвинута ровно на полпериода исходной частоты, — такие схемы применяют, например, для управления шаговыми двигателями или для управления елочной гирляндой «бегущие огни» (попробуйте составить такую схему сами!).

Большое значение на практике имеют формирователи коротких импульсов, называемые еще *схемами выделения фронтов*. На рис. 16.7, *а* приведена схема, которая делает это, *как положено*, используя эффект задержки сигнала в логическом элементе. При поступлении положительного фронта на вход он сразу же переключает выход последнего элемента «И-НЕ» в состояние логического нуля. На выходе цепочки из трех инверторов также возникнет логический ноль, который вернет выход в единичное состояние, но это произойдет не сразу, а спустя время, равное утроенной задержке срабатывания логических элементов. Поэтому на выходе возникнет короткая «иголка», достаточная по длительности (задержка-то тройная!) для надежного срабатывания других элементов схемы. Длительность таких импульсов составит для КМОП несколько десятков или сотен наносекунд. При желании можно выделить не фронт, а спад импульса (и получить при этом на выходе «иголку» положительной полярности), — для этого нужно использовать элементы «ИЛИ-НЕ». А если использовать «Исключающее ИЛИ», то можно получать положительные импульсы при каждом переключении сигнала: и по фронту, и по спаду.

«ФРОНТ ИМПУЛЬСА» И «ОТРИЦАТЕЛЬНЫЙ ПЕРЕПАД»

В интуитивно понятном термине «фронт импульса» имеется некоторая неоднозначность, связанная с тем, что термином этим иногда обозначают только положительный перепад напряжения (т. е. переход из состояния нуля в единицу), чтобы отличить его от отрицательного (перехода из единицы в ноль), который тогда называют «спадом импульса». А иногда под «фронтом» понимают вообще любой перепад напряжения (чтобы уточнить, о чем конкретно идет речь, в таком случае говорят о положительном или отрицательном фронте или перепаде). В англоязычной литературе соответствующие термины звучат, как «rising edge» и «falling edge» (букв. «возрастающая кромка» и «падающая кромка»), что более соответствует смыслу явления.

Подобно тому, как термин «отрицательный перепад» отнюдь не означает наличия отрицательного напряжения относительно «земли», так и «полярность сигнала» в приложении к логическим уровням означает не полярность напряжения относительно той же «земли», а просто состояние логической единицы (положительный сигнал, высокий уровень) или логического нуля (отрицательный сигнал, низкий уровень).

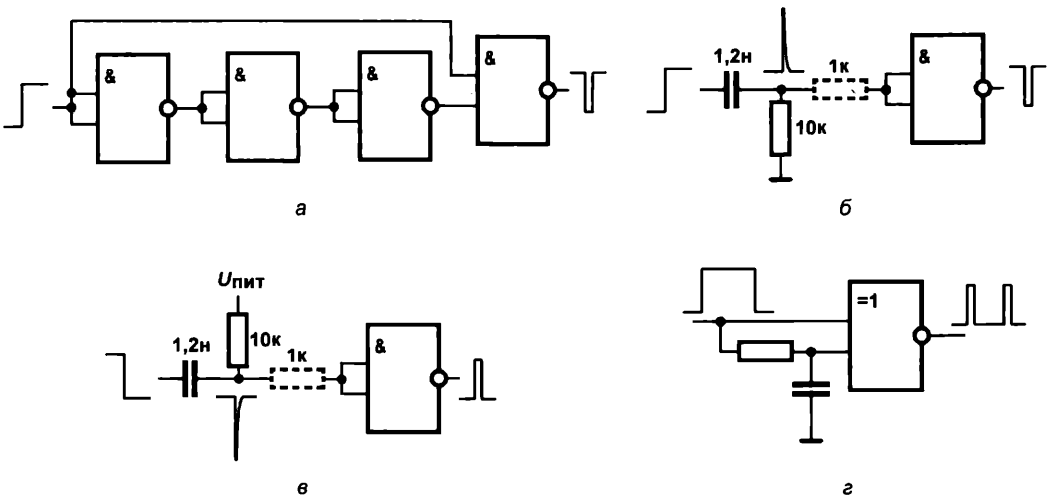


Рис. 16.7. Схемы формирователей импульсов: *а* — стандартная схема выделения фронтов; *б*, *в* — схемы с использованием дифференциальных *RC*-цепочек; *г* — схема формирователя импульсов по фронту и по спаду

Все здорово, но схема (см. рис. 16.7, *а*) уж больно громоздкая для такой простой функции — целый корпус! К тому же столь короткие импульсы очень сложно наблюдать на обычном аналоговом осциллографе. Поэтому на рис. 16.7, *б* и *в* приведены гораздо более экономичные схемы, которые делают то же самое, но с нарушением чистоты цифровых принципов, ибо являются наполовину аналоговыми. Длительность импульса на выходе схем выделения фронтов при указанных на схеме номиналах составит около 10 мкс. Схема 16.7, *б* выделяет фронт импульса, а схема 16.7, *в* — спад, обе они не реагируют на фронты противоположной полярности.

На рис. 16.7, *г* показано аналогичное использование элемента «исключающее ИЛИ» для выделения одновременно и фронта, и спада прямоугольного импульса. Такая схема может пригодиться, например, для измерения длительности импульса в детекторе событий с использованием двоичных счетчиков или триггеров для подсчета количества срабатываний с одновременным измерением временно́го интервала.

Одновибраторы

Одновибратор — это устройство, которое по внешнему сигналу выдает один-единственный импульс определенной длительности, не зависящей от длительности входного импульса. В некоторых современных текстах, посвященных одновибра-

торам, можно встретить их под названием «моностабильный мультивибратор», представляющим собой кальку соответствующего английского термина. Стоит иметь в виду, что «одновибратор», «моностабильный мультивибратор» и «ждущий мультивибратор» — все это одно и то же устройство.

Запуск одновибратора происходит либо по фронту, либо по спаду входного импульса. Для одновибратора без перезапуска возникновение на входе нового перепада напряжений той же полярности во время действия выходного импульса игнорируется, для одновибратора с перезапуском длительность выходного импульса в этот момент начинает отсчитываться заново. Как и в случае мультивибраторов, существует огромное количество схемотехнических реализаций этого устройства. Мы подробно изучим вариант схемы без перезапуска, который получается небольшой модификацией схем выделения фронта (см. рис. 16.7, *б* и *в*), — нужно только ввести в них положительную обратную связь, которая будет фиксировать состояние выхода на время заряда конденсатора.

Схема на рис. 16.8, *а* работает следующим образом. В состоянии покоя на выходе схемы состояние логической единицы, т. к. вход второго (правого) элемента «И-НЕ» заземлен через резистор R. Поскольку на входе тоже логическая единица, то на выходе первого (левого) элемента «И-НЕ» логический ноль, и конденсатор разряжен. При возникновении на входе схемы отрицательного уровня, на выходе первого элемента типа «И-НЕ» возникает состояние логической единицы, которое через дифференцирующую цепочку RC передается на вход второго элемента, так что на выходе схемы и, соответственно, на втором входе первого элемента оказывается логический ноль. Это состояние схемы, уже независимо от уровня входного сигнала, будет устойчиво: обратная связь как бы перехватила и зафиксировала уровень нуля на выходе на время, пока конденсатор заряжается от выхода первого элемента через резистор R. Через время, примерно равное произведению *RC*, конденсатор зарядится до порога срабатывания выходного элемента «И-НЕ», и выход

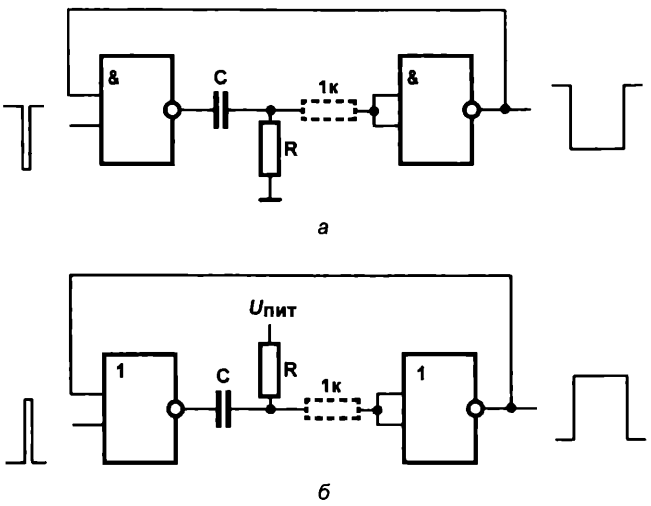


Рис. 16.8. Одновибраторы

схемы скачком перейдет обратно в состояние логической единицы по выходу опять же независимо от состояния входа.

Если к этому времени по входу схемы уже установился уровень логической единицы, как бывает в большинстве случаев (одновибраторы в основном предназначены для работы с короткими импульсами на входе), то первый элемент также перебросится в начальное состояние, конденсатор С быстро разрядится через ограничительное сопротивление 1 кОм (если оно установлено, см. далее) и входные защитные диоды второго элемента, и схема придет в начальное состояние в ожидании следующего запускающего импульса. Длительность импульса на выходе всегда будет примерно равна *RC*, даже в случае, если входной импульс длиннее (в этом случае конденсатор просто разрядится не сразу, а только тогда, когда закончится входной импульс). Совершенно аналогично работает схема на рис. 16.8, *б*, только с противоположными полярностями импульсов.

Главное применение одновибраторов — в качестве таймера, который формирует сигнал определенной длительности вне зависимости от работы всей остальной схемы. Естественно, о высокой точности выдержки времени тут говорить не приходится, но часто этого и не требуется. Например, если вы хотите ограничить по времени тревожный сигнал, подающийся с помощью устройства по схеме на рис. 16.3, то целесообразно управлять им от одновибратора, который запускается, скажем, нажатием кнопки. В одновибраторах для больших выдержек не возбраняется использовать электролитические конденсаторы, хотя даже при использовании только керамических или полимерных типов с максимальными емкостями порядка 1–3 мкФ вполне достижимы выдержки в несколько десятков секунд.

Одновибраторы с перезапуском, в которых выходной импульс в случае прихода нового импульса продлевается, мы проектировать не станем, потому что они более громоздкие, и в этом случае проще использовать готовую микросхему 561АГ1 (1561АГ1, CD4098), с подключением которой вы, без сомнения, легко разберетесь самостоятельно.

Возможно, вы уже сообразили, что одновибратор может служить эффективным средством подавления дребезга механических контактов (см. также *главу 15* и эту главу далее), поскольку будет запускаться только от первого перепада уровней, причем даже независимо от того, пролетают подвижные контакты весь промежуток «туда-обратно» или нет. Главным его преимуществом в этом качестве, несмотря на довольно сложную схему, является возможность использования двухвыводной кнопки, а не переключающей. Вход одновибратора при этом соединяют с питанием (в схеме на рис. 16.8, *а*) или с «землей» (в схеме на рис. 16.8, *б*) через резистор, а кнопкой замыкают этот вход, соответственно, на «землю» или на питание. Недостатком такого варианта является то, что приходится четко рассчитывать необходимую длительность импульса, иначе дребезг можно пропустить. Вторым недостатком схемы с одновибратором является неопределенность ситуации с размыканием ранее замкнутой кнопки, т. к. если кнопка удерживается в замкнутом состоянии дольше, чем длится импульс, то из-за дребезга при размыкании одновибратор может выдать импульс повторно. Заметим, что из одновибратора с перезапуском схема подавления дребезга получается еще лучше, чем из обычного — схема просто

не сбросится, пока на вход приходят какие-либо импульсы, так что в критичных случаях, несмотря на сложность, лучше применять именно ее.

О ТОКООГРАНИЧИВАЮЩИХ РЕЗИСТОРАХ В ИМПУЛЬСНЫХ СХЕМАХ

В схемах генераторов на рис. 16.2, формирователей на рис. 16.7 и одновибраторов на рис. 16.8 показан пунктиром дополнительный резистор, ограничивающий ток через защитные диоды микросхемы. С функциональной точки зрения он не требуется, но дело в том, что дифференцирующая *RC*-цепочка, которая составляет основу этих схем, вырабатывает импульсы не только по нужному переключению сигнала, но и по противоположному, и при этом импульсы выходят за пределы питания, в чем вы можете убедиться, если взглянете на рис. 5.9.

Посмотрим, когда в этих схемах можно обойтись без токоограничивающих резисторов. Типовой допустимый постоянный ток через защитные диоды составляет порядка 10 мА для «классической» КМОП (20–30 мА для быстродействующих серий). Мы знаем, что в режиме короткого замыкания элемент «классической» КМОП выдает примерно 5 мА при 5 В питания, т. е. для серии 561 при низких напряжениях питания ограничительных резисторов не требуется. Но та же серия при напряжениях питания 9 В и выше, и тем более серии быстродействующей КМОП, которые гораздо мощнее по выходу, могут перегружать защитные диоды. Тем не менее, как мы знаем, у диодов достаточно высокая перегрузочная способность, если только они не перегреваются, поэтому короткие импульсы им все равно не страшны. Так что, при больших выдержках времени или при низких частотах (т. е. при больших значениях емкости конденсатора) и напряжении питания выше 7–9 вольт такой резистор лучше поставить, а в остальных случаях можно без него обойтись.

Одновибраторы и генераторы на микросхеме 555

Микросхема 555 (или «таймер 555») представляет собой по сути объединение аналогового компаратора (точнее, даже двух компараторов) с RS-триггером, который мы рассмотрим далее в этой главе. Однако основные применения микросхемы 555 вплотную примыкают к теме рассмотренных нами мультивибраторов и одновибраторов на цифровой логике, потому мы рассмотрим этот таймер до ознакомления со свойствами триггеров. Заметим, что авторы известного труда [4] рассматривают микросхему 555 просто как прецизионный триггер Шмидта, что многое объясняет в ее свойствах.

ЗАМЕТКИ НА ПОЛЯХ

Популярность микросхемы 555 во многом обусловлена грамотным маркетингом компании Signetics, выпустившей ее впервые в 1971 году, — микросхема продавалась по беспрецедентно низкой для своего времени цене в 75 центов. Все схемы на основе 555 с легкостью воспроизводятся на других компонентах, причем в последнем случае нередко оказываются удобнее и проще, однако к концу 1970-х годов для нее было уже придумано около 500 применений. Существенных преимуществ у различных схем генераторов и одновибраторов на основе микросхемы 555 перед схемами, рассмотренными в предыдущих разделах этой главы, практически нет, за одним исключением: временны́е схемы на 555 обладают более высокой стабильностью, чем на дискретной логике, потому что период колебаний задан внутренними порогами компараторов, отсчитываемыми в долях от напряжения питания, и, следовательно, независимыми от его абсолютной величины (в первом приближении). Стабильность базовой схемы генератора на 555 (см. далее) примерно в 10 раз выше, чем у любой из схем на КМОП-

логике из приведенных ранее — не хуже 1–3% во всем диапазоне допустимого напряжения питания, составляющего от 4,5 до 18 вольт для классического варианта NE555 (отечественная версия КР1006ВИ1 — от 3 до 15 В) и от 2 до 18 вольт для КМОП-версии (ICM7555, КР1441ВИ1). Современные версии 555 декларируют температурную стабильность кристалла на редком для полупроводниковых схем уровне 0,005 %/°С (лучший показатель характерен разве что для микросхем прецизионных источников опорного напряжения).

Следует еще учесть, что все версии микросхемы 555 традиционно обладают мощным выходом — от 100 до 200 мА, что позволяет напрямую подключать даже обмотки малогабаритных реле. Такая стабильность и неприхотливость в 1970-е годы могла быть существенным фактором в пользу применения 555, однако в настоящее время при дешевых кварцевых резонаторах получение точных значений частоты доступно каждому, а в большинстве остальных применений между разбросом в 10% и разбросом в 1% разницы нет почти никакой. Тем более, что под высокие показатели микросхемы еще надо правильно подогнать остальные параметры схемы, а это, как мы увидим далее, не такая простая задача, как кажется. По указанным причинам, а также из-за плохой управляемости входными и выходными уровнями (если требуется поменять полярности сигналов, то все равно приходится ставить внешние инверторы), невозможности непосредственного управления одним таймером от выхода другого и вдобавок раздражающей несимметричности сигнала в автоколебательном режиме (скважность полученного меандра принципиально не равна 2, см. далее), в первых изданиях этой книги я о микросхеме 555 только упоминал мимоходом. Тем не менее, таймер 555 продолжают выпускать практически все значимые полупроводниковые фирмы, и рассмотреть хотя бы основные применения таймера 555 заставляет его незатухающая популярность.

Функциональная схема таймера 555 в многочисленных пособиях и даже в фирменных описаниях по неясной причине воспроизводится очень разнообразным и не всегда понятным образом (это касается даже авторитетнейшей книги [5]). Автор постарался исключить недостатки разных источников, ориентируясь в основном на [4], а также на принципиальную схему таймера, в результате чего получилась схема, представленная на рис. 16.9. Таймер выпускается только в 8-контактном корпусе DIP или SO (с плоскими планарными выводами), потому разводка контактов у всех разновидностей 555-й одинакова (исключение представляют только сдвоенные версии — по два таймера в одном корпусе, микросхема 556). Диаграммы показывают уровни напряжений на управляющих выводах и на выходе микросхемы в основных рабочих режимах. Отметим, что если вход сброса Reset (вывод 4) не задействован, то он должен подключаться к напряжению питания. Если не задействован вывод управления делителем (вывод 5), то он через конденсатор 10–100 нФ подключается к «земле».

Приятная особенность микросхемы 555 — высокая и симметричная нагрузочная способность: 200 мА как в состоянии логической единицы, так и логического нуля (у КМОП-версии, как и у отечественного варианта, эти значения снижены до 100 мА). Это позволяет напрямую подключать самую различную нагрузку — например, обмотки маломощных электромагнитных реле.

Рассмотрим подробнее принцип работы таймера 555 согласно приведенной функциональной схеме. Пусть в исходном состоянии на входе «Запуск» (Trigger выв. 2) высокий уровень, на входе «Порог» (Threshold, выв. 6) — низкий, конденсатор C1 разряжен через открытый транзистор VT1, выходной сигнал (на выводе 3) имеет

низкий уровень. При переключении сигнала на входе «Запуск» в низкий уровень выход переключается в высокий уровень, и таймер ждет положительного перепада на выводе «Порог». Как только напряжение на этом выводе превысит 2/3 напряжения питания, выход опять переключается в низкий уровень и одновременно открывается транзистор VT1 («Разряд» или Discharge, выв. 7). Из этого описания следуют основные применения таймера 555 в качестве генератора или одновибратора.

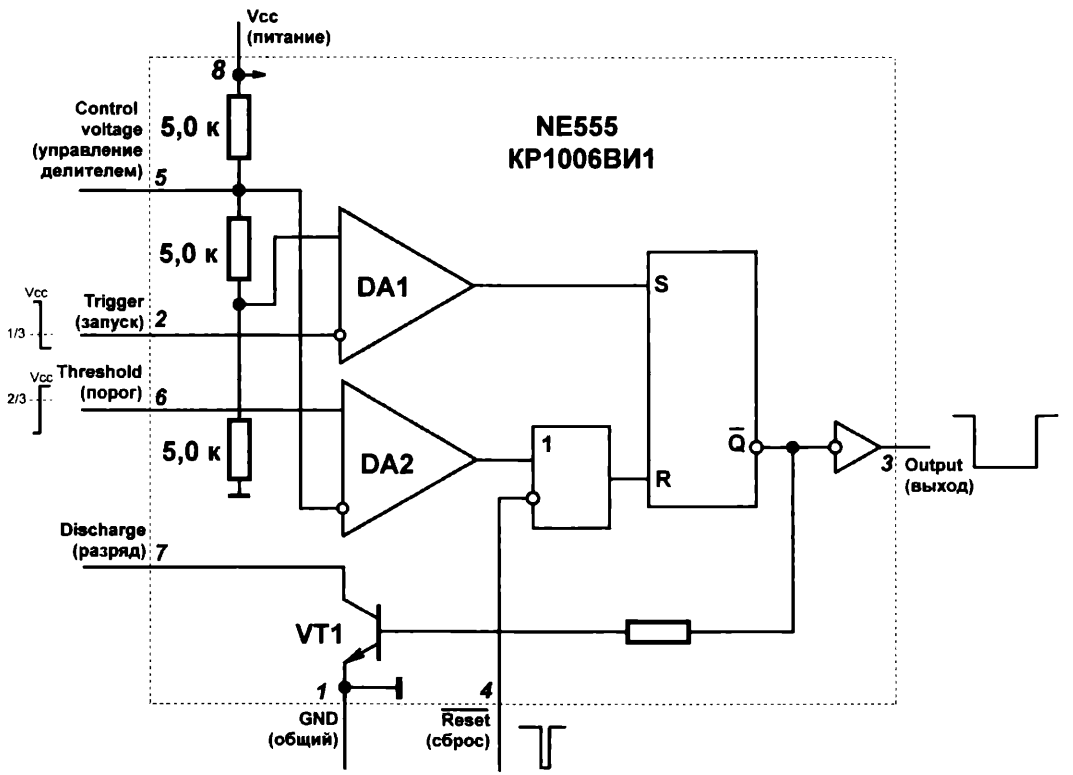


Рис. 16.9. Функциональная схема таймера NE555

Стандартная схема автоколебательного генератора (мультивибратора) на основе таймера 555 показана на рис. 16.10, *а*. В первый момент после включения питания конденсатор C1 разряжен, что равносильно подаче падающего фронта на вывод «Запуск», т. е. началу отсчета. Конденсатор заряжается через пару резисторов R1 и R2. Согласно описанному ранее алгоритму работы таймера, в момент, когда на выводе «Порог» превышается уровень 2/3 $U_{пит}$, на выходе появляется низкий уровень, а конденсатор начинает разряжаться через резистор R2 и открывшийся транзистор по выводу «Разряд». В момент, когда на выводе «Запуск» напряжение снижается до 1/3 $U_{пит}$, на выходе появляется высокий уровень, разрядный транзистор запирается, конденсатор вновь начинает заряжаться, и все возвращается к исходному состоянию. Возникает автоколебательный режим, при котором напряжение на конденсаторе принимает вид пилообразных колебаний между значениями 1/3 и 2/3 $U_{пит}$ (для его использования можно подключить к конденсатору буферный усилитель на

ОУ — например, по рис. 12.3, *в*), а на выходе имеется несимметричный меандр, в котором длительность высокого уровня $T_в$ на выходе превышает длительность низкого $T_н$ (этот факт, может быть, несколько утрированно отражен на рис. 16.10, *а*).

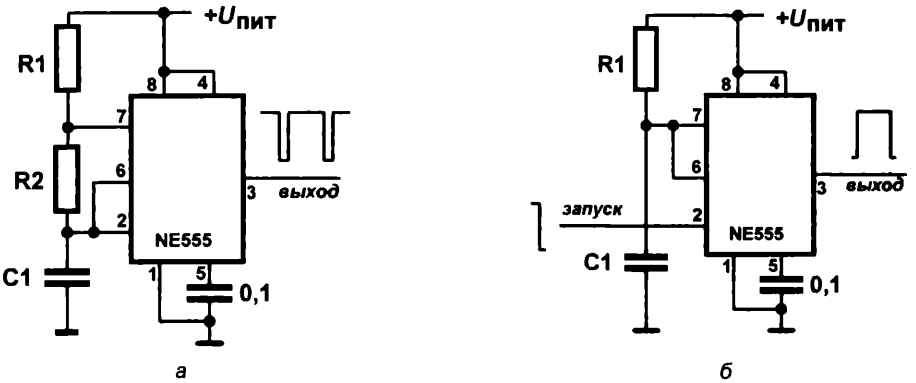


Рис. 16.10. Схемы на микросхеме NE555: *а* — генератор; *б* — одновибратор

Соотношения, определяющие длительности этих импульсов, зависят от резисторов R1 и R2: $T_в = 0{,}7 \cdot (R_1 + R_2) \cdot C_1$, $T_н = 0{,}7 \cdot R_2 C_1$ (теоретически рассчитанную величину коэффициента, равную 0,693, я позволил себе округлить до 0,7). Итого, период $T = T_в + T_н = 0{,}7 \cdot (R_1 + 2R_2) \cdot C_1$, а частота колебаний, соответственно, равна обратной величине. Рекомендуемая величина резисторов здесь может быть от 1 кОм до единиц мегаом, а конденсатора — от 1 нФ до практически любой величины, при которой реальный электролитический конденсатор еще будет работать в этой схеме (иными словами, примерно до нескольких сотен микрофарад). Разумеется, как и всегда во времязадающих цепях, электролитических конденсаторов следует избегать (или, как минимум, пользоваться танталовыми или ниобиевыми их разновидностями, а не «ширпотребовскими» алюминиевыми), но факт, что схема с такими большими временами будет работать, пусть с заранее неизвестной точностью. Что касается верхнего предела частоты, то 1 нФ на 1 кОм дает, согласно приведенным формулам, частоту около 500 кГц, что и считается максимально достижимой частотой работы таймера 555.

Из соотношений для $T_в$ и $T_н$ следует, что для приближения к величине скважности, равной 2, можно попробовать уменьшать значение R_1 относительно R_2, но беспредельно делать это, естественно, нельзя: в открытом состоянии через транзистор VT1 (см. рис. 16.9) и резистор R1 идет ток от источника питания, и большого значения ни транзистор, ни, возможно, источник питания попросту не вынесут. Потому правильный путь исправления несимметричности колебания состоит не в увеличении соотношения R_2/R_1, а во введении в схему особым образом ориентированных диодов, на чем мы здесь останавливаться не станем. Все равно самым простым и надежным будет способ, изложенный ранее в *разд. «Формирователи импульсов»* (см. рис. 16.6) — ведь и в обычных мультивибраторах на дискретной логике (см. рис. 16.2) скважность в точности не равна значению 2, хотя и гораздо

ближе к нему, чем в случае таймера 555. В практических приложениях чаще всего точное значение скважности значения не имеет, и специально заботиться об этом не нужно.

Более интересным с практической точки зрения будет применение таймера 555 в качестве прецизионного одновибратора, т. е. таймера в прямом смысле слова. Рекомендуемая схема одновибратора показана на 16.10, *б*. Отличается она от предыдущей схемы другим подключением входов «Запуск» (выв. 2) и «Разряд» (выв. 7), а также наличием только одного резистора. В начальном состоянии, как и в схеме генератора, на выходе низкий уровень, транзистор VT1 открыт, конденсатор C1 разряжен. При подаче падающего фронта на вывод «Запуск» триггер переключается, на выходе устанавливается высокий уровень и длится до тех пор, пока конденсатор не зарядится до порога в 2/3 напряжения питания. Схема приходит в исходное состояние и ждет следующего запускающего импульса. Отметим, что одновибратор этот без перезапуска, как и простейшие схемы на рис. 16.8, т. е. приход новых импульсов до окончания выдержки времени никак на схему не повлияет. Однако и продлевать действие низкого уровня на входе сверх заданной длительности на выходе тоже нельзя, что, на взгляд автора этой книги, является крупнейшим недостатком таймеров на основе 555, не позволяющим их включать каскадно без дополнительных ухищрений (попробуйте, например, воспроизвести на основе 555 схему по рис. 16.3, и вы поразитесь, насколько она окажется сложнее).

Разумеется, чтобы получить действительно прецизионный таймер, придется устанавливать резисторы с малым температурным дрейфом (типа С2-29В или аналогичные), а главное, — искать стабильные конденсаторы (например, К10-17Б с со значениями ТКЕ класса МП0 или его импортные аналоги класса NP0).

Заметки на полях

Подобрать компоненты, соответствующие классу таймера, — дело непростое. Предельный интервал, который можно получить с достаточно высокой точностью, определяется емкостью доступных конденсаторов. Для приемлемых по стабильности конденсаторов К10-17 или К73-17 импортной группы NP0 или отечественной в диапазоне от М150 до П100 трудно найти типы емкостью более 50 нФ. Если вы не наткнетесь где-нибудь на военный склад с прецизионными полистироловыми или поликарбонатными разновидностями (см. *главу 5*) или не захотите помучиться с импортными полифенилсульфидными, доступными лишь в корпусах для поверхностного монтажа, то этими номиналами придется и ограничиться. В таком случае при максимально допустимой величине резистора длительность интервала окажется равна долям секунды, что бессмысленно с точки зрения практических нужд. Потому придется поступиться принципами и применять конденсаторы распространенных ненормируемых классов (таких как Y5V или отечественного Н90), для которых можно найти номиналы, как минимум, в единицы микрофарад. В сочетании с мегаомными резисторами они дадут времена уже порядка секунд, и для некоторых простых целей такая схема на таймере наверняка окажется целесообразнее, чем громоздкие варианты с кварцевым генератором и цепочкой счетчиков. Конечно, таймер может потянуть времена и в десятки-сотни секунд (если применить электролитические конденсаторы), но о стабильности тогда говорить сложно. Еще раз напомним, что в этом случае предпочтительно применять ниобиевые (К53) или танталовые (К52) конденсаторы, а не обычные алюминиевые. А лучше всего в случае необходимости точно отмерять длительные интервалы времени все-таки воспользоваться возможностями микроконтроллера (см. *часть IV*) — пусть это сложнее и дороже, зато гарантирует результат.

Вход управления делителем (вывод 5 на схеме рис. 16.9) может использоваться для точной подстройки частоты или временно́го интервала, в других задачах он используется редко — в основном из-за неудобства, связанного с ограниченной допустимой величиной управляющего напряжения, которое можно подавать на этот вывод. Для эффективного управления выходной частотой напряжение на выводе 5 должно колебаться вокруг значения в 2/3 напряжения питания, не выходя за пределы последнего. По этой причине строить на микросхеме 555, например, схему ГУН или модулятора (как это иногда рекомендуется) нецелесообразно. Если уж вы хотите построить ГУН именно с использованием таймера 555, то следует вместо примитивной времязадающей цепочки резистор-конденсатор, как в базовой схеме мультивибратора (см. рис. 16.10, *а*), установить интегратор или управляемый источник тока на ОУ. Такая схема все равно без дополнительных ухищрений не будет обладать высокими характеристиками и к тому же получится неоправданно сложной. Намного проще использовать для построения ГУН специальные микросхемы (например, MC4024 или LM331), с гарантированной нелинейностью и стабильностью.

Триггеры, регистры и счетчики

Триггеры и построенные на их основе счетчики и регистры играют огромную роль в электронике. В состав любого процессора, кроме собственно АЛУ на основе комбинационной логики (рассмотренных в *главе 15* сумматоров), входят регистры и память, которые являются прямыми родственниками счетчиков, — потому что для построения и тех и других используются триггеры. Вот со знакомства с классическими типами триггеров мы и начнем.

Между прочим, одно из главных «неэлектронных» значений слова «trigger» — спусковой крючок у огнестрельного оружия. В электронике *триггер* — это устройство для записи и хранения информации в количестве одного бита. Существуют и многостабильные триггеры, которые могут хранить более одного бита, но на практике они почти не используются. Любая элементарная ячейка памяти, будь то магнитный домен на пластинах жесткого диска, отражающая область на поверхности CD-ROM или конденсаторная ячейка электронного ОЗУ, обязательно обладает триггерными свойствами, т. е. может хранить информацию спустя еще долгое время после того, как она была в нее введена.

Самый простой триггер

Самый простой триггер можно получить, если в схемах одновибраторов на рис. 16.8 удалить *RC*-цепочку и соединить напрямую выход первого элемента со входом второго. Если схема находится в состоянии, при котором на выходе уровень логической единицы, то кратковременная подача отрицательного уровня на вход, как и в случае одновибратора, перебросит выход в состояние логического нуля. Но теперь уже нет конденсатора, который осуществляет отрицательную обратную связь и в конце концов возвращает схему в исходное состояние, потому в таком состоянии схема останется навечно, если мы что-то не предпримем. Чтобы вернуть ее в исходное состояние, надо подать точно такой же сигнал, но на вход второго

элемента, который (вход) в схеме одновибратора у нас отсутствует. Если мы его введем, то получим симметричную схему с двумя входами, которые обозначаются буквами R и S (от слов **R**eset и **S**et, т. е. «сброс» и «установка»). Само же устройство носит название RS-триггера. Оба варианта такой схемы на элементах «И-НЕ» и «ИЛИ-НЕ» показаны на рис. 16.11.

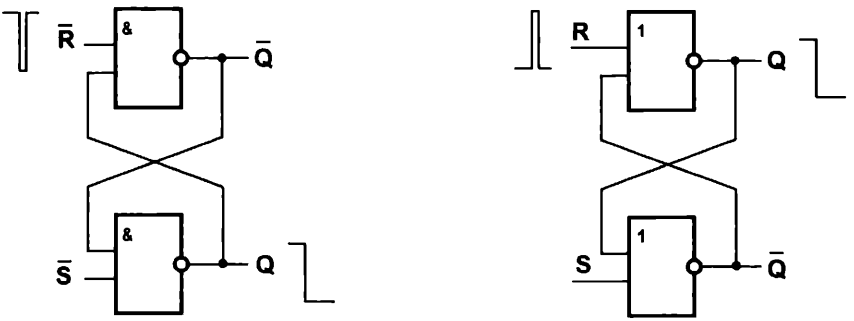


Рис. 16.11. Схемы триггеров на элементах «И-НЕ» (*а*) и «ИЛИ-НЕ» (*б*)

Нет нужды перебирать все состояния этих схем и приводить соответствующие таблицы истинности, нужно только твердо запомнить, что подача импульса на вход R *всегда* устанавливает на *прямом* выходе Q состояние логического нуля. (Легко сообразить, что если поменять все обозначения местами: R на S, а прямой выход Q на инверсный $\overline{Q}$, то в схеме ничего не изменится.)

Вход S, естественно, здесь означает ровно противоположное — установку выхода Q в состояние логической единицы, но, в отличие от входа R, который всегда означает обнуление, вход S в различных устройствах может использоваться и в несколько иных целях, а чаще вообще отсутствует. Входы R и S могут управляться различными полярностями сигнала в зависимости от построения триггера: для схемы на элементах «И-НЕ» (см. рис. 16.11, *а*) — это низкий уровень, поэтому входы R и S обозначены с инверсией. Уровни, которые меняют состояние триггера, называются *активными*, — так, для схемы на рис. 16.9, *а* активным является низкий уровень. «Более правильная» схема в этом смысле — на элементах «ИЛИ-НЕ» (см. рис. 16.11, *б*), в ней активный уровень соответствует положительной логике, т. е. он высокий. Обратите внимание, что установка дополнительных инверторов по входу, казалось бы, превращает схему на рис. 16.11, *а* в схему на рис. 16.11, *б*, но только в смысле полярности активных уровней, прямой и инверсный выходы при этом останутся на своих местах.

В схемах RS-триггеров подача активного уровня на R-вход ничего не меняет, если выход Q уже был в состоянии логического нуля, то же самое справедливо для S-входа при выходе в состоянии логической единицы. Однако, пока на соответствующем входе действует напряжение активного уровня, подача активного уровня на второй вход запрещена. Это не означает, что триггер при этом сгорит, просто он потеряет свои триггерные свойства, — на обоих выходах установится один и тот же

уровень, а после одновременного снятия активного уровня со входов состояние будет неопределенным (точнее, будет определяться тем элементом, который переключится чуть позже другого). Неопределенное состояние будет и после подачи питания, поэтому нужно принимать специальные меры для установки схемы в нужное состояние после включения. Наиболее распространенной такой мерой является подача определенного уровня в начальный момент времени на один из нужных входов с помощью *RC*-цепочки.

Ввиду практической важности этого способа я приведу варианты соответствующей схемы, несмотря на ее очевидность (рис. 16.12). В них конденсатор в первый момент времени после подачи питания разряжен, и на входе логического элемента оказывается положительный уровень, который устанавливает триггер в состояние логического нуля на выходе Q. Затем конденсатор заряжается, и в дальнейшем *RC*-цепочка больше не оказывает влияния на работу схемы. Постоянную времени *RC* лучше выбирать побольше, чтобы к моменту зарядки конденсатора успели пройти все переходные процессы, — на схемах по рис. 16.12 она равна примерно 0,5 мс. Естественно, при этом следует позаботиться, чтобы на «настоящих» RS-входах к моменту окончания заряда конденсатора был неактивный уровень, иначе все пойдет насмарку. Чтобы избежать нагромождения инверторов, можно в этой схеме использовать трехвходовые элементы (561ЛЕ10), как показано на рис. 16.12, *б*.

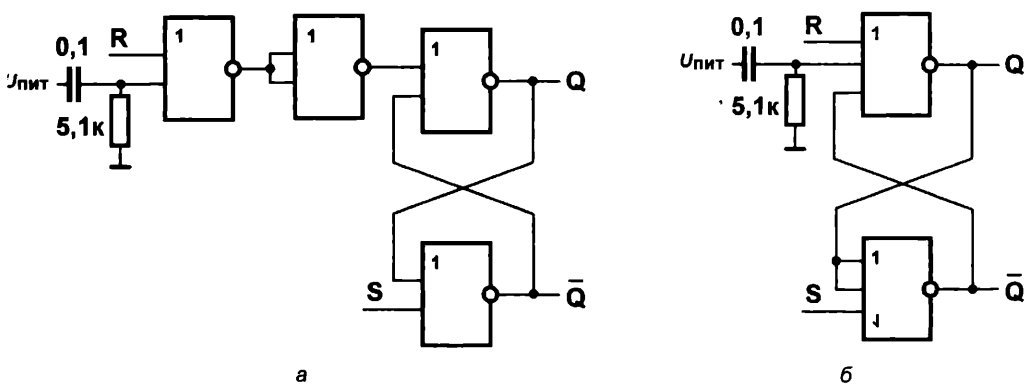


Рис. 16.12. Схемы триггеров с предустановкой при включении питания: на двухвходовых элементах (*а*) и на трехвходовых элементах (*б*)

Естественно, RS-триггеры выпускают и в интегральном исполнении (561ТР2 содержит четыре простых RS-триггера). Все более сложные триггеры, а также счетчики в интегральном исполнении, обязательно имеют отдельные асинхронные RS- или хотя бы только R-входы. Причем соответствующий вход у любого устройства, его имеющего — от микропроцессоров до счетчиков, — является асинхронным, т. е. вся система обнуляется в момент подачи импульса по входу R, независимо от того, что в этот момент она делает. Говорят еще, что вход Reset — вход обнуления — имеет *наивысший приоритет*. Именно это происходит, скажем, когда вы нажимаете кнопку Reset на системном блоке вашего компьютера.

Использование RS-триггера является самым кардинальным способом решения проблемы дребезга контактов. Стандартная схема показана на рис. 16.13, *а*, однако нет

никакой нужды городить такую схему с резисторами, относительно которых еще нужно соображать, к чему их подключать (для варианта с «ИЛИ-НЕ» их пришлось бы присоединять к «земле», а не к питанию). На рис. 16.13, *б* показана упрощенная схема, которая работает точно так же, и при этом в ней можно использовать любые инверторы, в том числе и одновходовые. Общим недостатком схем «антидребезга» как с RS-триггерами, так и на основе элемента «Исключающее ИЛИ» (см. рис. 15.8, *б*), является необходимость использования переключающей кнопки с тремя выводами, которых в продаже предлагается гораздо меньше, чем обычных замыкающих и размыкающих с двумя контактами. Попробуйте приспособить двухвыводную кнопку к любой из указанных схем, и вы сами придете к выводу, что это невозможно. Поэтому на практике часто приходится использовать схему «антидребезга» с использованием одновибратора (в том числе реализованного программными способами в микроконтроллерах) — при всех ее недостатках.

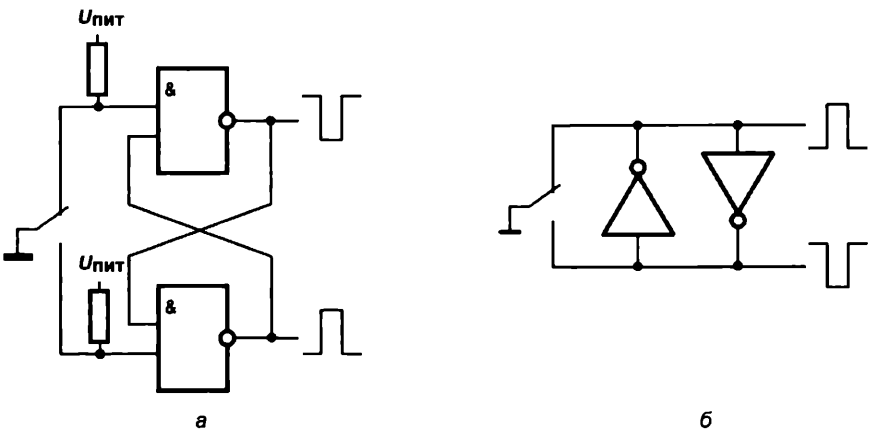


Рис. 16.13. Схемы «антидребезга» на RS-триггерах

D-триггеры

D-триггеры получили свое название от слова «delay», что означает «задержка». На самом деле существуют две их разновидности, формально различающиеся только тем, что первая из них управляется уровнем сигнала (статический D-триггер или триггер-защелка), а вторая — фронтом импульса (динамический D-триггер). Фактически же это разные по устройству и области использования схемы.

Для того чтобы отличить статический D-триггер от динамического, мы в обозначении на схеме для первого поставим букву L (от слова «level» — уровень), а для второго — букву «E» (от слова «edge» — фронт). Эти обозначения не являются общепринятыми, и в дальнейшем мы их использовать не будем, только здесь — для наглядности. Микросхема 561ТМ3 содержит четыре статических триггера-защелки с общим входом синхронизации, а 561ТМ2 — два динамических D-триггера с раздельными дополнительными входами R и S (мы с ней уже знакомы — см. рис. 16.6). Если тип вообще не указывается, то обычно по умолчанию предполагается, что речь идет о динамических D-триггерах.

Статический D-триггер легко получить из RS-триггера путем небольшой модификации его схемы. Если из схемы на рис. 16.14, *а* исключить вход C (например, объединив входы каждого элемента и превратив их тем самым в простые инверторы), то получится довольно бесполезное устройство, которое на выходе Q будет просто повторять входные сигналы, а на втором выходе, соответственно — выдавать их инверсии. Наличие тактового входа C (от слова «clock», которое в цифровой электронике значит «тактовый импульс») все меняет.

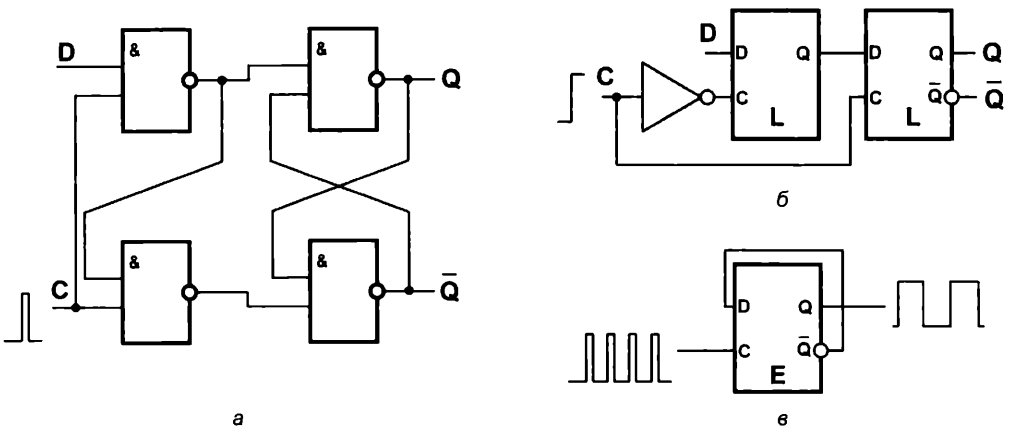


Рис. 16.14. D-триггеры: *а* — схема статического D-триггера; *б* — схема динамического D-триггера на основе двух статических; *в* — счетный триггер на основе динамического D-триггера

Если мы обратимся к диаграммам на рис. 15.8, *а*, то увидим, что при наличии на этом входе уровня логической единицы входные сигналы будут пропускаться на вход RS-триггера, и схема станет повторять на выходе Q уровни на входе D. Если же мы установим на входе C уровень нуля, то схема немедленно «зависнет» в состоянии выхода, соответствующем входному уровню непосредственно перед приходом отрицательного фронта на вход C, — т. е. запомнит его! Поэтому такой триггер и называют *защелкой* — при подаче на вход C короткого положительного тактового импульса он как бы «защелкивает» состояние входа. Статический D-триггер можно использовать в качестве буферного регистра для хранения данных — например, результатов счета импульсов на то время, пока идет сам процесс счета. Статическая энергозависимая память (SRAM) также, как правило, использует такие триггеры в качестве элементарных ячеек.

Динамические D-триггеры более универсальны, и область применения у них куда шире, чем у статических. Однако динамический триггер устроен более сложно. Один из способов построения динамического D-триггера из двух статических показан на рис. 16.14, *б*. Эта схема работает следующим образом: когда на общем входе C наличествует отрицательный уровень, состояние входа D переписывается на выход первого (слева) триггера, при этом второй триггер заперт. Сразу после положительного фронта на входе C это состояние переписывается во второй триггер и появляется на выходе Q, а первый триггер запирается. Таким образом, запоминание состояния общего D-входа происходит в точности в момент положительного пере-

пада уровней и никогда больше. Если изменить местоположение инвертора и присоединить его ко входу второго триггера, а на первый триггер подавать тактовые импульсы напрямую, то срабатывание станет происходить по отрицательному фронту, и такой тактовый вход будет считаться инверсным. Для того чтобы получить дополнительные входы асинхронной принудительной установки триггера в нулевое и единичное состояние (R- и S-входы), нужно для *обоих* статических триггеров выходные (правые на схеме рис. 16.14, *а*) элементы сделать трехвходовыми и объединить соответствующие входы у обоих триггеров. Устанавливать по входам R и S только выходной триггер недостаточно (подумайте почему?).

А на рис. 16.14, *в* показана самая простая схема счетного триггера на основе динамического D-триггера, та самая, знакомая нам по рис. 16.6. Из описанного ясно, как она работает, — при каждом положительном перепаде на выход Q будет переписываться состояние противоположного выхода $\overline{Q}$, т. е. система станет с приходом каждого тактового импульса менять свое состояние на противоположное, в результате чего на выходе мы получим симметричный (независимо от скважности входных импульсов) меандр с частотой, вдвое меньшей, чем входная. Такой триггер можно считать делителем частоты на два или одноразрядным двоичным счетчиком — в зависимости от того, для чего он используется. В отличие от всех остальных типов триггеров (а кроме описанных, распространены еще, например, и так называемые *JK-триггеры*, но мы их рассматривать не станем), счетные триггеры в интегральном исполнении отдельно не выпускаются (при случае их легко, как вы видели, соорудить, например, из D-триггеров), а выпускаются только готовые многоразрядные двоичные счетчики, из таких триггеров составленные. К рассмотрению счетчиков мы перейдем чуть позднее, а пока кратко остановимся на регистрах.

Регистры

Регистрами называют устройства для хранения одного двоичного числа. Количество разрядов в регистрах, выпускаемых отдельно, обычно не превышает восьми, но в составе других микросхем могут быть и регистры с большей разрядностью — вплоть до 128 или 256 битов в «продвинутых» микропроцессорах. Большинство типов электронных запоминающих устройств, вообще говоря, можно рассматривать как совокупность регистров. Но собственно регистры, как входящие в состав процессоров, так и выпускаемые отдельно, отличаются тем, что позволяют не только записывать и считывать информацию, но и производить некоторые простейшие операции, — например, сдвиг разрядов.

Простейший регистр — это упомянутый ранее статический D-триггер. Четыре таких триггера, входящих в микросхему 561ТМ3, образуют четырехразрядный регистр с параллельной записью и считыванием, причем тактовый вход в этой микросхеме у всех четырех разрядов общий. Как и сам триггер, такой регистр называют «защелкой».

Если регистр-защелка позволяет осуществлять только параллельную запись, то последовательный регистр (пример — 561ИР2) наоборот, имеет возможность записи только через один вход, который является D-входом самого младшего разряда. По-

следовательный регистр представляется неким обобщением конструкции D-триггера. Работу динамического D-триггера можно рассматривать, как процесс сдвига информации от входа через первый триггер ко второму при поступлении соответствующих перепадов на тактовом входе. В последовательном регистре, который в простейшем случае представляет собой просто последовательное соединение таких триггеров, происходит нечто подобное — с каждым фронтом тактового импульса информация сдвигается от младшего разряда к старшему, при этом в младший разряд записывается состояние входа. Считывать информацию при этом можно из каждого разряда в отдельности, как и в случае регистра-защелки. Такие регистры получили еще название *сдвиговых*. Они широко используются для последовательного ввода и вывода информации — скажем, для вывода восьми битов через последовательный порт RS-232 достаточно записать их в такой регистр, а потом подать на него восемь тактовых импульсов с нужной частотой.

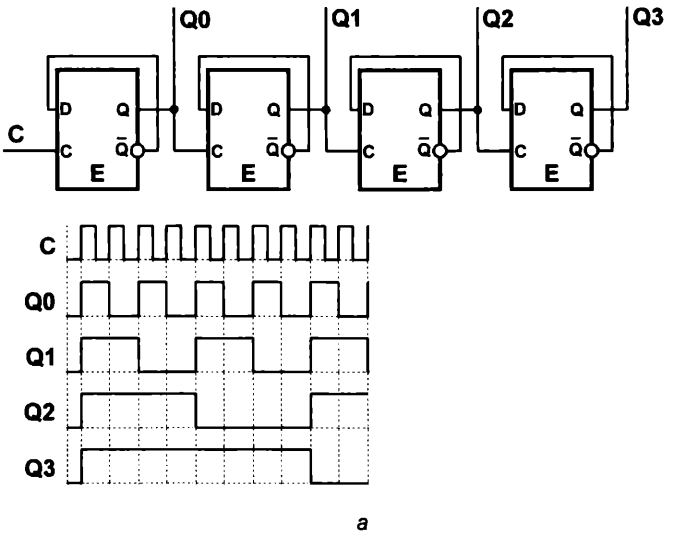
Сдвиговый регистр можно закольцевать — соединить выход старшего разряда со входом младшего и получить нечто подобное слону из анекдота, который засунул хобот себе в известное место. Однако в случае одного сдвигового регистра такое соединение приведет к тому же результату, что и для слона, т. е. оно довольно бесполезно практически, ибо мы без дополнительных ухищрений запись информации производить уже не сможем. Поэтому используют объединение параллельной и последовательной записи/считывания в одном устройстве (пример — четырехразрядный регистр 561ИР9 или восьмиразрядный 561ИР6).

Заметки на полях

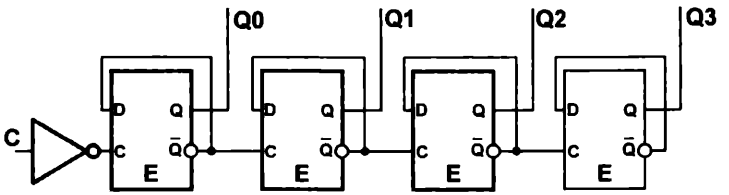
Такие сдвиговые регистры с параллельной записью и последовательным считыванием информации — неотъемлемая часть устройств памяти большой емкости, без них чтение и запись в большие массивы запоминающих ячеек были бы невозможны. Имеются они, например, в матрицах цифровых камер. Интересное применение таких регистров — организация последовательного интерфейса SPI, широко используемого для скоростного обмена информацией между различными микросхемами (например, между энергонезависимой памятью вроде флэш-карточек и микроконтроллером). В SPI наличествуют два восьмибитовых регистра, соединенных в кольцо входами/выходами, но они разделены пространственно: один регистр находится в одном устройстве, другой — в другом. Если подавать тактовые импульсы на оба регистра одновременно (это осуществляет одно из устройств — ведущее), то после подачи ровно 8 импульсов устройства обменяются содержимым своих регистров.

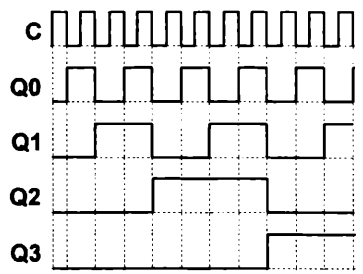
Счетчики

Самый простой счетчик можно получить, если соединить последовательно ряд счетных триггеров, как показано на рис. 16.15, *а*. У этой схемы есть две особенности. В первой из них легко разобраться, если построить диаграмму работы такого счетчика, начиная с состояния, в котором все триггеры находятся в состоянии низкого уровня на выходе («0000»). В самом деле, при подаче первого же импульса триггеры перейдут в состояние со всеми единицами («1111»)! Если строить диаграмму дальше, то мы увидим, что последовательные состояния будут такими: «1110», «1101» и т. д. В этом легко узнать последовательный ряд чисел 15, 14, 13 — т. е. счетчик получился вычитающим, а не суммирующим.



а





б

Рис. 16.15. Схемы асинхронных счетчиков на D-триггерах:
а — вычитающего; *б* — суммирующего

А как можно получить суммирующий счетчик? Очень просто — надо ко входу каждого следующего триггера подсоединить не прямой выход предыдущего, а инверсный. Порядка ради можно тактовые импульсы подавать также через инвертор (рис. 16.15, *б*), тогда все разряды счетчика, включая самый младший, будут срабатывать по заднему (отрицательному) фронту входного импульса, а не по переднему (у «настоящих» счетчиков тактовый вход и делается инверсным). В этом случае будет все в порядке — входные импульсы будут суммироваться (см. диаграмму), и мы получим ряд последовательных состояний: «0000», «0001», «00010», «0011» и т. д.

Заметки на полях

Удивительная все же штука — электроника! Сначала мы получили полную аналогию между абстрактной математической теорией и состояниями переключателей на реле, теперь вот — между не менее абстрактным арифметическим счетом и последовательными состояниями счетчика на триггерах. Чем этот счетчик отличается от палочек, раскладываемых на земле дикарем? Ничем, кроме того, что он раскладывает не палочки, а уровни напряжений, причем выгодно отличается от первобытного сознания тем, что еще и «владеет» позиционной системой счисления. Начинаешь понимать, почему ученые середины прошлого века были так обольщены возможностями электронных схем, что даже заговорили о «машинном разуме». Но это уже другая тема...

Однако у счетчиков, построенных по такой простейшей схеме, есть один крупный недостаток, которого мы отчасти касались в этой главе. А именно — переключение триггеров происходит асинхронно, сигнал от входа должен пройти всю цепочку, пока на выходе также изменится уровень. Эти, казалось бы, незначительные задержки могут, однако, привести к существенным неприятностям вроде возникновения лишних «иголок» при дешифрировании состояний выхода. А при больших частотах входных импульсов, на пределе возможностей логических элементов, фронты сигналов на выходах могут приобрести совершенно хаотическое расположение относительно входного сигнала, так что дешифрировать состояние счетчика будет невозможно. Поэтому большинство счетчиков в интегральном исполнении делают по иным, синхронным, схемам, когда входной тактовый сигнал подается одновременно на все разряды, и фронты выстраиваются строго «по линеечке», независимо от задержек в том или ином триггере. Подробно изучать синхронные схемы мы не будем, т. к. самим нам их строить не придется, а здесь рассмотрим пару конкретных типов серийно выпускаемых счетчиков.

Первый из счетчиков, который мы изучим подробно, — 561ИЕ10. Микросхема содержит два одинаковых четырехразрядных синхронных счетчика в одном корпусе. Разводка выводов ее показана на рис. 16.16, *а*, где вроде бы все понятно, кроме назначения вывода E. Каждый четырехразрядный счетчик, входящий в состав этой микросхемы, работает так: если на выводе E присутствует напряжение высокого уровня, то счетчик будет переключаться по *положительному* фронту на входе C. Однако это касается только первого триггера — все остальные станут работать в соответствии с диаграммой на рис. 16.15, *б*, т. е. счетчик будет суммировать импульсы. Вывод E тут является разрешающим («enable») для тактового входа C.

Однако если оставить на входе C напряжение логического нуля, а тактовые импульсы подавать на вход E, то счетчик будет срабатывать от *отрицательного* перепада напряжений на этом входе, т. е. диаграмма его окажется в полном соответствии с диаграммой на рис. 16.15, *б*. В этом варианте вход C будет разрешающим для входа E. Как видите, можно было бы поменять обозначения E и C местами, однако в этом случае их следует дополнить знаком инверсии.

Поэтому если вы хотите каскадировать два счетчика из этой микросхемы, получив в результате один восьмиразрядный счетчик, то выход Q3 первого счетчика нужно присоединить именно ко входу E второго, подав на вход C потенциал логического нуля. Учтите, однако, что при этом обе половинки результирующей конструкции (старшая и младшая тетрады) станут работать асинхронно относительно друг друга,

и срабатывание четырех старших разрядов будет происходить позднее, чем срабатывание младших.

Вывод обнуления R обозначен без инверсии, что означает установку всех разрядов в состояние 0 при подаче высокого уровня на этот вход. Пока этот уровень присутствует, счетчик будет игнорировать любые изменения на тактовых входах. Максимальная рабочая частота микросхемы ИЕ10 при напряжении питания 5 В — 1,5 МГц, минимальная длительность импульса сброса — 250 нс. Кстати, при составлении таблицы для *приложения 3* я с некоторым трудом разыскал для нее быстродействующий аналог, и соответствие счетчика 74*xx*393 (ИЕ19) микросхеме 561ИЕ10 неполное: хотя у них даже совпадают основные выводы корпуса, но ИЕ10 выпускается в корпусе с 16-ю выводами, а 74*xx*393 — с 14-ю. Отсутствующие в аналоге два вывода у ИЕ10 выполняют как раз функцию разрешения счета E для двух половинок микросхемы, т. е. в аналоге он отсутствует, а входы тактовых импульсов C — инверсные.

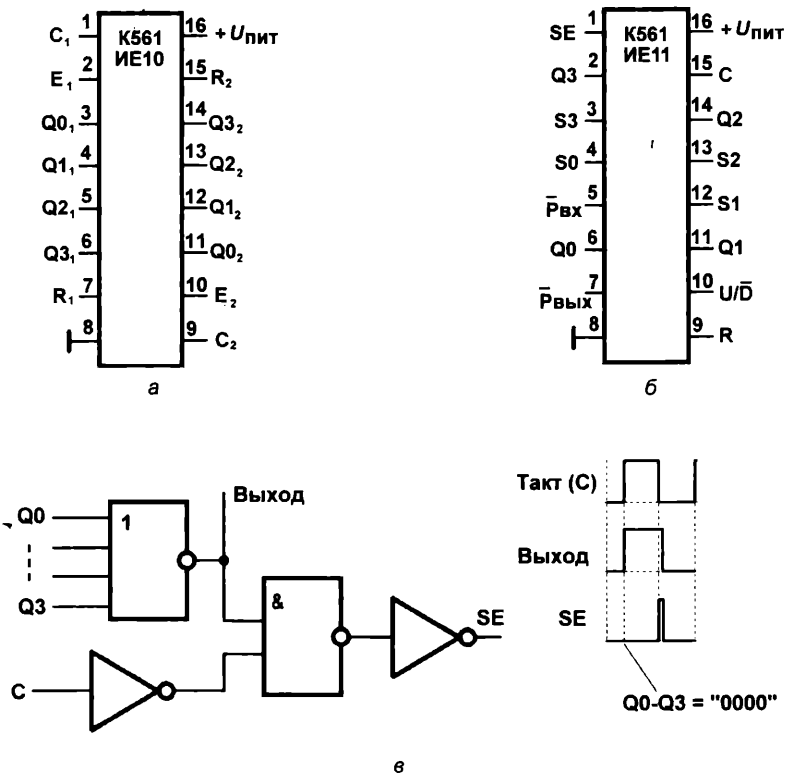


Рис. 16.16. Разводка выводов счетчиков 561ИЕ10 (*а*) и 561ИЕ11 (*б*); организация автоматической предустановки счетчиков типа ИЕ11 (*в*)

Счетчик 561ИЕ11 более универсален, и управляющих выводов у него значительно больше, поэтому в 16-выводном корпусе умещается только один четырехразрядный счетчик. Разводка и обозначение выводов для него показаны на рис. 16.16, *б*. Не

правда ли, можно запутаться? Однако на самом деле все гораздо проще, чем выглядит.

Если на выводах R, SE и $\overline{P_{вх}}$ присутствуют низкие уровни напряжения, а на входе $U/\overline{D}$ — высокий, то счетчик считает по положительному фронту на входе C, в точности так же, как это делает половинка ИЕ10 при высоком уровне на входе E. Разберемся с действием остальных входов. Со входом обнуления R все понятно — при подаче на него высокого уровня все обнуляется. Вход $U/\overline{D}$ служит для реверсии (потому такой счетчик еще называется *реверсивным*) — если на него подать напряжение логического нуля, то счетчик будет не суммировать, а вычитать, подобно тому, как это делает счетчик, показанный на рис. 16.15, *а*.

Самый интересный — вход SE («set enable» — разрешение установки). Если на него подать напряжение логической единицы, то в триггеры счетчика запишутся значения, установленные на входах S0–S3. Возможность такой параллельной предустановки значительно расширяет возможности счетчика. А вход и выход переноса $\overline{P_{вх}}$ и $\overline{P_{вых}}$ предназначены для каскадирования счетчиков — для получения синхронного (в отличие от ИЕ10) счетчика большей разрядности, надо входы C у всех микросхем объединить, а выход $\overline{P_{вых}}$ предыдущего счетчика соединить со входом $\overline{P_{вх}}$ следующего. У самого первого счетчика, естественно, $\overline{P_{вх}}$ присоединяется к «земле».

Сколько удовольствия можно получить от этой схемы! Я покажу только один из вариантов того, как ее использовать. Наличие возможности предустановки произвольного значения позволяет соорудить из этой микросхемы счетчик с любым коэффициентом деления входной частоты (в пределах емкости исходного счетчика) — если используется один корпус ИЕ11, то это значения от 1 до 16. В самом деле, если счетчик считает в стандартном режиме, то частота на выходах Q_x будет равна входной, поделенной на 2, 4, 8 и 16. Принцип установки другого коэффициента проще всего показать на примере обратного (вычитающего) режима счета. Арифметика тут простая: предположим, мы установили на входах предустановки число 3 (0011) и организовали схему так, чтобы в состоянии, когда все выходы Q_i равны нулю, это число каждый раз записывалось бы в счетчик. Поскольку режим вычитающий, то при подаче тактовых импульсов на вход счетчик будет последовательно проходить состояния: предустановка (0011) — фронт тактового импульса (0010) — фронт тактового импульса (0001) — фронт тактового импульса (0000) — предустановка (0011) и т. д. То есть после каждых трех тактовых импульсов счетчик будет возвращаться в исходное состояние (если предустановки нет, то он это делает после каждых 16 импульсов).

Таким образом мы получили коэффициент деления, равный 3. Можно резюмировать: в вычитающем режиме коэффициент деления будет такой, каково число на входах предустановки. А что будет, если запустить счетчик в обычном режиме, суммирующем? Нетрудно подсчитать, что коэффициент деления при этом будет равен разности между максимально возможным коэффициентом (16) и установленным числом (3) — в данном случае 13.

Я так подробно на этом останавливаюсь, кроме всего прочего, еще и потому, что счетчики-таймеры в микропроцессорных системах (и в составе микроконтроллеров типа AVR, и в других микроконтроллерах, и системный счетчик в IBM PC, который без изменений воспроизводится во всех системах от Intel) работают совершенно аналогично ИЕ11. И для того чтобы успешно программировать микроконтроллерные системы, необходимо очень хорошо понимать принцип их работы, — без таймеров микроконтроллеры, можно сказать, вообще не нужны. Но в микроконтроллерах все остальное за нас уже сделали, а если вернуться к обычной интегральной логике, то сразу встают два вопроса. Первый их них звучит так: а откуда, собственно, считывать эту поделенную частоту?

При использовании вычитающего режима считывать частоту придется с того выхода счетчика, который соответствует реальной разрядности делителя, — по расписанной «диаграмме» видно, что в случае коэффициента, равного 3, старшие разряды вообще не используются. Это приемлемо, если мы хотим иметь раз и навсегда установленный коэффициент, но в общем случае неудобно — если коэффициент по ходу дела меняется. Поэтому нужно либо использовать суммирующий режим, при котором старший разряд всегда задействован, и результирующая частота снимается именно с него (например, при приведенных значениях счетчик будет все время считать от 3 до 15), либо... либо есть еще одна возможность, для знакомства с которой придется ответить сначала на второй вопрос: как организовать предустановку значения счетчика каждый раз при достижении им состояния «0000»?

Схема на рис. 16.16, *в* демонстрирует, как можно это сделать (это не единственный вариант, но нам подойдет). Приведенная справа от схемы диаграмма ее работы показана, начиная с момента, когда вот-вот должен прийти такт, при котором счетчик установится в состояние «0000». В любом другом состоянии на входе четырехвходового элемента «ИЛИ-НЕ» присутствует хотя бы одна единица и на выходе его, соответственно, логический ноль. Выход же элемента «И-НЕ» пребывает по этой причине в состоянии логической единицы, а на входе SE — логический ноль, как и положено при счете.

Как только с очередным положительным тактовым перепадом счетчик установится в состояние «0000», то на выходе элемента «ИЛИ-НЕ» появится логическая единица. Поскольку тактовый импульс пропускается через инвертор, то на нижнем по схеме входе «И-НЕ» в этот момент логический ноль, и на выходе его по-прежнему единица. И только после отрицательного перепада на тактовом входе C (который для остальной схемы является нерабочим) на обоих входах «И-НЕ» установится логическая единица, на выходе — логический ноль, и на SE появится, наконец, долгожданный высокий уровень, разрешающий запись значений, установленных по входам S0–S3. Как только запись произойдет, все немедленно отработает назад, т. к. на выходе «ИЛИ-НЕ» высокого уровня уже не будет. Импульс на входе SE окажется очень кратковременным, но нам длиннее и не надо.

Легко сообразить, что частота положительных импульсов на выходе «ИЛИ-НЕ» (либо коротких импульсов перезаписи на входе SE) как раз в точности равна частному от деления входной частоты на установленный коэффициент. И ее в этом качестве использовать удобнее, чем выходы разрядов, потому что не нужно гадать,

с какого разряда снимать частоту при вычитающем режиме. Если поставить многопозиционный переключатель, меняющий код на входах предустановки S0–S3, то можно получить счетчик с изменяемым коэффициентом деления от 1 до 16.

Микросхема 561ИЕ14 полностью аналогична ИЕ11, за исключением того, что у нее есть еще вход переключения двоичного/десятичного счета B/D. Так как все выводы 16-выводного корпуса у ИЕ11 заняты, для этого пришлось пожертвовать входом R, вместо которого имеется вывод B/D. Если на этом входе напряжение низкого уровня, счетчик ИЕ14 считает в двоично-десятичном коде, если высокого — становится полностью аналогичным ИЕ11. Постойте, а как же его обнулять, если вход R отсутствует? Очень просто — надо подать на все входы предустановки S0–S3 потенциал «земли», а импульс обнуления подавать на вход SE вместо R.

Цифровой лабораторный генератор

Напоследок мы рассмотрим одну практическую схему на счетчиках. Это давно обещанный (см. *главы 2* и *12*) цифровой лабораторный генератор, для которого нам придется использовать еще один тип счетчика, «заточенного» для работы именно в качестве делителя частоты.

Счетчик 561ИЕ16, который мы здесь применим, ничего особенного собой не представляет и является простым асинхронным счетчиком, подобным показанному на рис. 16.15, *б*. Мы могли бы спокойно соорудить его сами из отдельных триггеров, но для этого их понадобилось бы целых 14 штук (т. е. семь корпусов типа ТМ2). А микросхема 561ИЕ16 в 16-выводном корпусе не только заключает в себе 14-разрядный счетчик, но и включает специальные буферные усилители по выходу каждого каскада деления, — для того чтобы случайная перегрузка одного каскада внешней нагрузкой не привела бы к остановке всех последующих. Кроме выходов Q0–Q13, наружу выведен вход тактовых импульсов C и вход сброса R.

Позвольте, спросите вы, но ведь еще два вывода заняты под питание, так что из 16 выводов на выходы Q*x* приходится всего 12, а не 14, откуда же взять еще два? Разработчики пошли по пути наименьшего сопротивления — почему-то вместо того чтобы использовать корпус, например, DIP-20 (заодно еще и вход S вывести, и еще что-нибудь, например, поставить буферный усилитель для входной частоты), они просто исключили выходы второго и третьего каскадов Q1 и Q2. Ну да ладно, обойдемся и тем, что дают.

Итак, с помощью этого счетчика мы получим генератор, который выдает 13 значений частоты (считая входную), каждая в два раза меньше предыдущей, за исключением небольшого «провала», — частоты с коэффициентом 4 и 8 будут отсутствовать. Чтобы получить при этом самые низкие частоты, кратные целым, без дробных частей, значениям, выраженным в герцах, нужно на входе использовать генератор с частотой, равной какой-нибудь степени двойки. Генератор можно, конечно, соорудить по одной из приведенных ранее схем мультивибраторов, но это не есть приемлемое решение для *лабораторного* прибора. Для цифрового генератора в нашем случае важна именно стабильность частоты, т. к. мы собираемся испытывать с его помощью разные измерительные схемы, — это одна из главных причин, по-

чему лучше сделать для себя такой генератор, а не использовать аналоговые серийные приборы с плавной перестройкой частоты. Есть, разумеется, в продаже и точные цифровые генераторы промышленного производства, но они на много порядков дороже того, что мы можем соорудить на двух микросхемах и пяти навесных деталях, — как говорится, одним взмахом паяльника. И при этом функциональность полученного генератора будет перекрывать наши потребности если и не на все 100%, то по крайней мере на 99% совершенно точно — исключая редкий случай потребности в частотах порядка мегагерц.

Мы возьмем тот самый кварц, который в народе называют «часовым» (см. *разд. «Кварцевые генераторы»* в этой главе). Он имеет частоту 32 768 Гц, что есть 2^{15}. Чаще всего такие резонаторы встречаются в продаже в малогабаритном цилиндрическом корпусе 8×3 мм, они без проблем работают при напряжениях питания до 12–15 В. Отечественный РК-206 имеет еще меньший корпус 6×2 мм, но и рассеиваемая мощность у него меньше, так что с ним лучше ограничиться напряжением питания 3–6 В.

Пристроив к такому генератору двоичный счетчик из пятнадцати ступеней, мы получим на выходе частоту ровно 1 Гц, которую можно использовать для отсчета времени в привычных нам секундах. Однако на одном счетчике ИЕ16 мы 1 Гц все равно не получим, ибо разрядов у нас только 14, поэтому придется ставить дополнительные делители. Кроме того, *длительность* положительного или отрицательного импульса с частотой 1 Гц, полученной на выходе счетчика, составляет не секунду, а полсекунды, так что для получения интервала («ворот») длительностью в 1 секунду придется поделить частоту еще раз пополам. Кстати, немаловажной и очень удобной особенностью нашего генератора станет то, что частоты будут точно (ну, почти точно, т. к. счетчик асинхронный) сфазированы, что позволит нам получать импульсы разной скважности, если использовать дополнительные логические элементы.

Сама схема представлена на рис. 16.17. Как видите, она крайне проста, и сборка ее требует только аккуратности при отсчете выводов микросхем. Если хотите еще упростить схему, то удалите дополнительный счетчик на DD3, тогда придется обходиться минимальной частотой 2 Гц (но не забудьте, что остаются свободные элементы «И-НЕ», и их входы надо присоединить к «земле»). Схему следует собрать на макетной плате и вывести все четырнадцать значений частоты на ее край, оформив в виде клеммника с соединениями под винт. Внешний вид такого клеммника показан на рис. 16.17, *справа*. Клеммники бывают разной конструкции и разных размеров, а также с дюймовым (5,08 мм) или метрическим (5 мм) шагом между выводами, сдвоенные (как на рис. 16.17), строенные или счетверенные, но все отличаются тем, что их можно соединять друг с другом в одну линейку наподобие деталей конструктора «Лего», получая таким образом клеммники с любым количеством выводов. Естественно, каждую клемму следует надписать, чтобы не заниматься каждый раз отсчетом выводов.

Да, а питание? Отдельного блока питания для этой конструкции делать вовсе не надо. Потребляет вся схема около 150–200 мкА при напряжении питания 5 В, при 15 В потребление возрастает до 1,5 мА. А так как 561-я серия безупречно работает

при питании от 3 до 15 В (и менять ее на быстродействующие аналоги в этой конструкции не следует), то целесообразно питать генератор от того же напряжения, что и испытываемые схемы, чтобы сразу получить нужные логические уровни. В случае чего схему несложно запитать от лабораторного источника.

На плату можно отдельно поставить еще одну микросхему с логическими элементами (ЛА7 или ЛЕ5) — они позволят собирать выходы счетчика для получения нужной скважности и для разных других целей. Можно добавить к схеме и ключевой транзистор, т. к. довольно часто приходится подключать различные мощные нагрузки вроде динамика или светодиодов. Если потребуется более высокая частота, то «часовой» кварц можно просто заменить другим резонатором частотой до 1 МГц. В общем, творите, как сумеете.

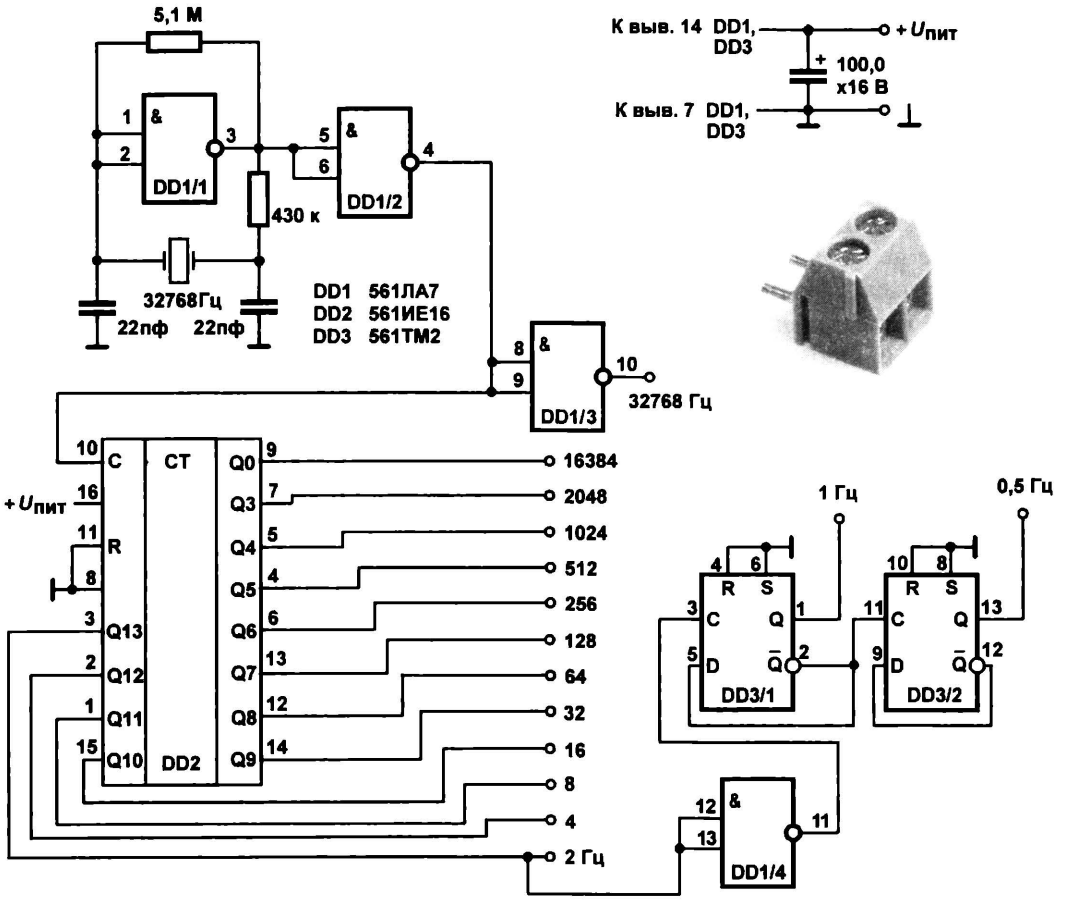
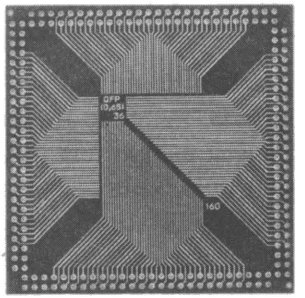


Рис. 16.17. Схема цифрового лабораторного генератора



ГЛАВА 17

Откуда берутся цифры

Цифроаналоговые и аналого-цифровые преобразователи

> Орудуйте мушкетом и шпагой, мой милый, в этих двух занятиях вы проявляете большое искусство, а перо предоставьте господину аббату, это по его части.
>
> *А. Дюма. «Три мушкетера»*

С человеческой точки зрения все природные явления носят непрерывный, аналоговый характер. Одно глобальное исключение из этого правила немало потрясло ученых, когда его обнаружили: речь идет об атомно-молекулярной структуре вещества и всей огромной совокупности явлений, которые являются следствием этого феномена. И все же даже это универсальное свойство материи нашими органами чувств непосредственно не обнаруживается — для нас все протекает так, как если бы явления природы были полностью непрерывными, т. е. характеризовались бы рядом действительных чисел, отстоящих друг от друга на бесконечно малые отрезки по числовой оси. В масштабах, которыми занимается атомная и молекулярная физика, все обстоит совершенно иначе, чем в привычном для нас мире, — например, такие характеристики, как температура или давление, теряют смысл, ибо они применимы только к очень большому, непрерывному ансамблю частиц.

Природа даже устроила нам некоторые препятствия на пути к полной дискретизации всего и вся — все элементарные частицы, как известно, могут вести себя и как дискретные частицы, и как непрерывные волны, в зависимости от условий эксперимента. В то же время мы обнаружили, что считать и вообще обрабатывать информацию лучше все-таки в цифровой форме, которая является универсальной и не зависит от природы физической величины, с которой мы манипулируем. Встает задача преобразования аналоговой величины в дискретную. Между прочим, термин «аналоговый» не слишком хорошо отражает сущность явления (что чему там «аналогично»?) — точнее говорить «непрерывный», а термин «аналоговый» есть лишь дань традиции, подобно «операционному» усилителю.

Естественно, когда мы хотим, чтобы преобразованная информация опять предстала перед нами в форме, воспринимаемой нашими органами чувств, то мы вынуждены делать и обратное преобразование — цифроаналоговое. Именно этим занимаются звуковые или видеокарты в компьютере. Однако такая задача возникает гораздо

реже, потому что во многих случаях информацию можно оставить в цифровой форме, так ее и отобразив — в виде смены кадров на дисплее, в виде дискретной шкалы цветов для цифрового изображения, в виде небольших «ступенек» на кривой нарастания звукового сигнала. В этих случаях мы полагаемся на устройство органов чувств человека: выше некоторого порога разрешения канала передачи перестает хватать, глаз или ухо работают подобно фильтру низких частот, отрезая пульсации, и мозгу кажется, что перед ним действительно непрерывный процесс.

Заметки на полях

Интересно, что непосредственный цифровой способ отображения информации, например, в виде совокупности цифр на семисегментном индикаторе, хотя и значительно более корректен, чем аналоговый (мы не теряем информации), не всегда может оказаться более правильным. Если вы вглядитесь в пульт управления каким-нибудь сложным устройством — не обязательно атомной электростанцией, достаточно «торпеды» обычного автомобиля, — вы увидите, что большинство показывающих приборов там стрелочные, аналоговые. Хотя, как вы понимаете, нет никаких проблем в современном автомобиле демонстрировать скорость, уровень топлива или температуру двигателя непосредственно в цифрах, но этого не делают сознательно, потому что в очень многих случаях человека не интересует точное значение того или иного параметра. Его интересует только отклонение от некоторого значения, или превышение некоторого порога, или вообще только тенденция изменения величины — но не сама эта величина, и не сам порог. Информация о том, что температура охлаждающей жидкости составляет 80 °С, для водителя совершенно излишняя, ему важно знать, что если вот эта стрелочка не достигла вот этой красненькой черточки, значит, все в порядке. Но бывают и другие случаи — например, отсчет пробега того же автомобиля имеет смысл, только будучи представленным именно в цифровом виде, поэтому еще на заре автомобилестроения пришлось придумывать разные — тогда еще, конечно, механические — счетчики, отображающие число пройденных километров. Все это следует учитывать при проектировании различных показывающих устройств, и при необходимости приходится даже идти на усложнение схемы, причем, что обидно, нередко с заведомой потерей информации или даже с ее искажением. Типичный пример из этой области — датчик количества топлива в том же автомобиле, который проектировщики традиционно заставляют врать, занижая показания, иначе слишком много водителей оказывалось бы на дороге с сухими баками в полукилометре от ближайшей заправочной станции.

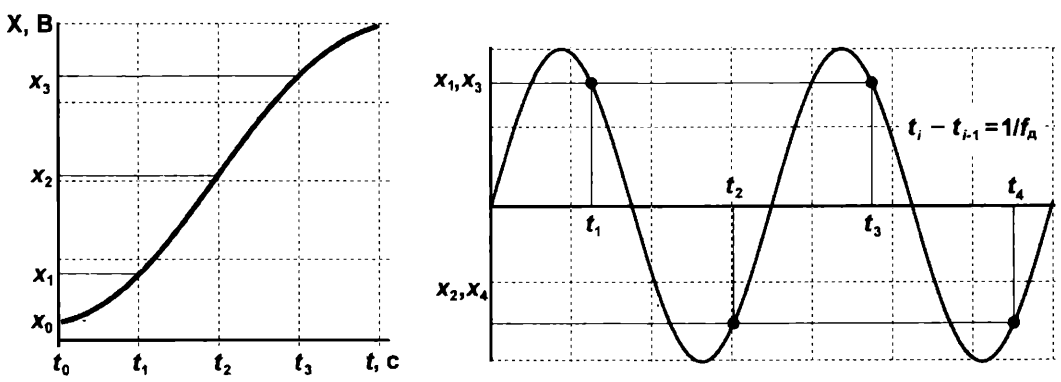
Другой пример: обычные часы с цифровым дисплеем. Все мы уже к этому делу привыкли, но ведь большинство практических задач заключаются не в определении точного времени, а в определении интервала — сколько мы уже ждем или сколько осталось до некоторого момента. Для этого знания точного значения часов и минут не требуется, и обладателю часов с цифровым дисплеем приходится в уме производить довольно сложные арифметические действия, вычисляя этот интервал, вместо того чтобы просто мысленно передвинуть стрелки на циферблате. Как видите, проектирование показывающих устройств не слишком простое дело, т. к. не может производиться только из соображений компактности схемы или высокой точности измерений, а непременно должно учитывать требования эргономики и «usability» — удобства пользования. Если читателю кажется, что меня очередной раз занесло несколько в сторону от темы книги, то это не так, потому что проектирование аналого-цифровых и особенно цифроаналоговых устройств неразрывно связано с проблемами человеко-машинного взаимодействия — не будь потребителя, незачем все это было бы и затевать. Компьютеры в общении между собой спокойно обойдутся и двоичным кодом.

Принципы оцифровки сигналов

Займемся сначала общими принципами аналого-цифрового преобразования. Основной принцип оцифровки любых сигналов очень прост и показан на рис. 17.1, *а*. В некоторые моменты времени t_1, t_2, t_3 мы берем мгновенное значение аналогового сигнала и как бы прикладываем к нему некоторую меру, линейку, проградуированную в двоичном масштабе. Обычная линейка содержит крупные деления (метры), поделенные каждое на десять частей (дециметры), каждая из которых также поделена на десять частей (сантиметры), и т. д. Двоичная линейка содержала бы деления, поделенные пополам, затем еще раз пополам и т. д. — сколько хватит разрешающей способности. Если вся длина такой линейки составляет, допустим, 2,56 м, а самое мелкое деление — 1 см (т. е. мы можем померить ей длину с точностью не хуже 1 см, точнее, даже половины его), то таких делений будет ровно 256, и их можно представить двоичным числом размером 1 байт или 8 двоичных разрядов.

Ничего не изменится, если мы меряем не длину, а напряжение или сопротивление, только смысл понятия «линейка» будет несколько иной. Так мы получаем последовательные отсчеты величины сигнала x_1, x_2, x_3. Причем заметьте, что при выбранной разрешающей способности и числе разрядов мы можем померить величину не больше некоторого значения, которое соответствует максимальному числу, — в данном случае 255. Иначе придется или увеличивать число разрядов (удлинять линейку), или менять разрешающую способность в сторону ухудшения (растягивать ее). Все изложенное и есть сущность работы аналого-цифрового преобразователя — АЦП.

На рис. 17.1, *а* график демонстрирует этот процесс для случая, если мы меряем какую-то меняющуюся во времени величину. Если измерения производить регулярно с известной частотой (ее называют *частотой дискретизации* или *частотой квантования*), то записывать можно только значения сигнала. Если стоит задача потом восстановить первоначальный сигнал по записанным значениям, то, зная частоту



а *б*

Рис. 17.1. Оцифровка аналоговых сигналов: *а* — основной принцип; *б* — пояснение к теореме Котельникова — Найквиста

дискретизации и принятый масштаб (т. е. какому значению физической величины соответствует максимальное число в принятом диапазоне двоичных чисел), мы всегда можем восстановить исходный сигнал, просто отложив точки на графике и соединив их плавной линией.

Но что мы при этом теряем? Посмотрите на рис. 17.1, *б*, который иллюстрирует знаменитую теорему Котельникова (как водится, за рубежом она носит другое имя — Найквиста, на самом деле они оба сформулировали ее независимо друг от друга). Здесь показана синусоида предельной частоты, которую мы еще можем восстановить, располагая массивом точек, полученных с частотой дискретизации $f_д$. Так как в формуле для синусоидального колебания $A \cdot \sin(2\pi ft)$ имеется два независимых коэффициента (A — амплитуда и f — частота), то для того чтобы вид графика восстановить однозначно, нужны как минимум две точки на каждый период[1], т. е. *частота оцифровки должна быть как минимум в два раза больше, чем самая высокая частота в спектре исходного аналогового сигнала.* Это и есть одна из расхожих формулировок теоремы Котельникова — Найквиста.

Попробуйте сами нарисовать другую синусоиду без сдвига по фазе, проходящую через указанные на графике точки, и вы убедитесь, что это невозможно. В то же время можно нарисовать сколько угодно разных синусоид, проходящих через эти точки, если их частота в целое число раз выше частоты дискретизации $f_д$. В сумме эти синусоиды, или *гармоники* (т. е. члены разложения сигнала в ряд Фурье — см. *главу 5*) дадут сигнал любой сложной формы, но восстановить их нельзя, и если такие гармоники присутствуют в исходном сигнале, то они пропадут навсегда. Только гармонические составляющие с частотами ниже предельной восстанавливаются однозначно. То есть процесс оцифровки равносилен действию ФНЧ с прямоугольным срезом характеристики на частоте, равной ровно половине частоты дискретизации.

Теперь об обратном преобразовании. В сущности, никакого преобразования цифра-аналог в ЦАП, которые мы будем здесь рассматривать, не происходит, просто мы выражаем двоичное число в виде пропорциональной величины напряжения, т. е. занимаемся, с точки зрения теории, всего лишь преобразованием масштабов. Вся аналоговая шкала поделена на кванты — градации, соответствующие разрешающей способности нашей двоичной «линейки». Если максимальное значение сигнала равно, к примеру, 2,56 В, то при восьмиразрядном коде мы получим квант в 10 мВ, и что происходит с сигналом между этими значениями, а также и в промежутки времени между отсчетами, мы не знаем и узнать не можем. Если взять ряд последовательных отсчетов некоего сигнала, например, тех, что показаны на рис. 17.1, *а*, то мы в результате получим ступенчатую картину, показанную на рис. 17.2.

Если вы сравните графики на рис. 17.1, *а* и на рис. 17.2, то увидите, что второй график представляет первый, мягко говоря, весьма приблизительно. Для того чтобы повысить степень достоверности полученной кривой, следует, во-первых, брать

[1] Если сами параметры синусоиды A и f не меняются во времени, то достаточно вообще двух точек на все время. Именно такой случай показан на графике рис. 17.1, *б*.

отсчеты почаще, и во-вторых, увеличивать разрядность. Тогда ступеньки будут все меньше и меньше, и есть надежда, что при некотором достаточно высоком разрешении, как по времени, так и по квантованию, кривая станет, в конце концов, неотличима от непрерывной аналоговой линии.

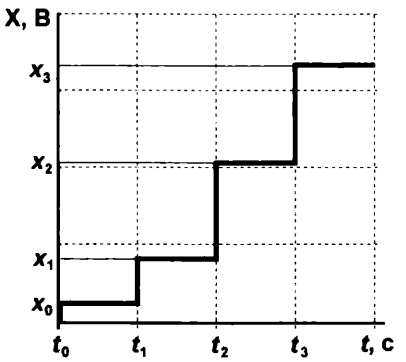


Рис. 17.2. Восстановление оцифрованного сигнала с рис. 17.1, *а*

Заметки на полях

Очевидно, что в случае звуковых сигналов дополнительное сглаживание, например, с помощью ФНЧ, здесь попросту не требуется, ибо оно только ухудшит картину, отрезая высокие частоты еще больше. К тому же всякие аналоговые усилители сами сгладят сигнал, и органы чувств человека тоже поработают в качестве фильтра. Так что наличие ступенек само по себе несущественно, если они достаточно мелкие, а вот резкий спад частотной характеристики выше некоторой частоты сказывается на качестве звука фатальным образом. Многие люди с хорошим музыкальным слухом утверждают, что они безошибочно отличают цифровой звук CD-качества (дискретизация которого производится с частотой 44,1 кГц, т. е. со срезом на частоте заведомо более высокой, чем уровень восприятия человеческого слуха, и с числом градаций не менее 65 тысяч на весь диапазон) от настоящего аналогового звука, например, с виниловой пластинки или с магнитофонной ленты. По этой причине качественный цифровой звук записывается с гораздо более высокими частотами дискретизации, чем формально необходимо, например, 192 и даже 256 кГц, и тогда он становится действительно неотличим от исходного. Правда, напрямую оцифрованный звук записывают разве что на диски в формате Audio CD, а почти для всех остальных форматов используют компрессию — сжатие по специальным алгоритмам. Если бы не компрессия, для записи не хватило бы ни емкости современных носителей, ни быстродействия компьютерных сетей: всего одна минута стереозвука с параметрами CD-качества занимает на носителе около 10 Мбайт, можете проверить самостоятельно.

Углубляться в особенности дискретизации аналоговых периодических сигналов мы не будем, т. к. это очень обширная область в современной инженерии, связанная в первую очередь с оцифровкой, хранением, тиражированием и воспроизведением звука и видео, и об этом нужно, как минимум, писать отдельную книгу. Для наших же целей достаточно изложенных сведений, а теперь мы перейдем непосредственно к задаче оцифровки и обратного преобразования отдельного значения сигнала.

ЦАП

Начнем мы с конца, т. е. с цифроаналоговых преобразователей — почему, вы увидите далее. Будем считать, что на входе мы имеем числа в двоичной форме — неважно, результат оцифровки какого-то реального сигнала или синтезированный код. Нам его нужно преобразовать в аналоговый уровень напряжения в соответствии с выбранным масштабом.

Самый простой ЦАП — десятичный или шестнадцатеричный дешифратор-распределитель, подобный 561ИД1. Если на него подать четырехразрядный код, то на выходе мы получим логическую единицу для каждого значения кода на отдельном выводе. Присоединив к выходам такого дешифратора линейку светодиодов, получаем полосковый (шкальный) индикатор, который с разрешением в 10 или 16 ступеней на весь диапазон будет показывать уровень некоей величины. Причем очень часто для практики такого относительно грубого индикатора, заменяющего стрелочные приборы, вполне достаточно. Выпускаются специальные микросхемы для управления такими дискретными шкальными индикаторами, которые позволяют показывать значение не в виде отдельной точки или полоски, а в виде светящегося столбика. Есть и микросхемы, которые могут управлять не дискретными, а линейными вакуумными индикаторами. Есть даже микросхема К1003ПП1 (аналог UAA180), которая преобразует аналоговую величину (напряжение) сразу в управляющий сигнал для шкального индикатора. Довольно эффектная конструкция может получиться, если в схеме термометра (см. рис. 13.3 или 13.4) заменить показывающую головку на такую микросхему и шкальный индикатор, — получится как бы полноценная имитация термометра традиционного!

У такого примитивного ЦАП есть два недостатка: во-первых, повысить его разрешение свыше 16–20 градаций нереально, т. к. выходов тогда потребуется чересчур много. Но главное, он предназначен для узкой задачи визуализации цифровой величины и за пределами этой области беспомощен. Куда более широкое применение имел бы преобразователь, осуществляющий функцию по рис. 17.2, т. е. выдающий на выходе аналоговое напряжение, пропорциональное коду на входе.

«Тупой» метод получения такого напряжения состоял бы в следующей модификации метода с дешифратором-распределителем типа 561ИД1. Для этого надо выстроить делитель из цепочки одинаковых резисторов, подключить его к источнику опорного напряжения и коммутировать отводы этого делителя ключами, управляемыми от дешифратора-распределителя. Для двух-трехразрядного кода можно использовать описанные в *главе 15* мультиплексоры типа 561КП1 и 561КП2. Но для большего количества разрядов такой ЦАП с непосредственным преобразованием превращается в совершенно чудовищную конструкцию. Для восьмиразрядного кода потребовалось бы 256 резисторов (строго одинаковых!), столько же ключей и дешифратор с таким же количеством выходов, а ведь восьмиразрядный код — довольно грубая «линейка», ее разрешающая способность не превышает четверти процента. Поэтому на практике такой метод употребляют для построения АЦП, а не ЦАП (потому что, несмотря на сложность, он обладает одним уникальным свойством, см. далее), и здесь мы даже не будем рисовать такую схему.

Рассмотрим один из самых распространенных методов, который позволяет осуществлять преобразование код-напряжение без использования подобных монструозных конструкций. На рис. 17.3, *а* показан вариант реализации ЦАП на основе ОУ с коммутируемыми резисторами в цепи обратной связи. В качестве коммутирующих ключей можно применить, например, малогабаритные электронные реле серии 293, т. е. того же типа, что мы применяли в конструкции термостата (см. рис. 12.9), или специализированные ключи из серии 590. Однако для осуществления переключающего контакта потребовалось бы ставить по два таких ключа на каждый разряд, потому в серии 561 предусмотрена специальная микросхема 561КТ3 (CD4066), которая содержит четыре одинаковых ключа, работающие именно так, как показано на приведенной схеме.

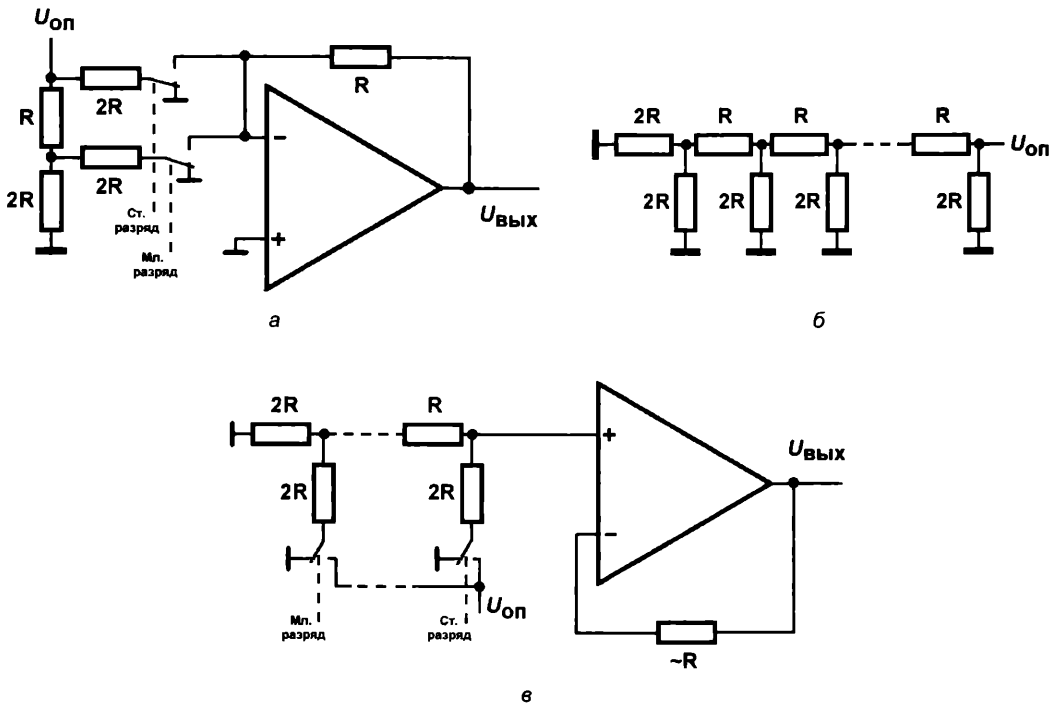


Рис. 17.3. *а* — двухразрядный ЦАП с отрицательным выходом; *б* — цепочка R–2R произвольной длины; *в* — ЦАП с положительным выходом

Ключи эти двунаправленные, но их выводы работают по-разному. Тот вывод, который обозначается OUT/IN (в отечественном варианте обычно просто «Выход»), в одном состоянии коммутируется с другим входом/выходом, в другом просто отключен, как обычно. А вывод, обозначаемый IN/OUT (в отечественном варианте просто «Вход»), в одном состоянии подключается к первому входу, а вот при разрыве ключа не «повисает в воздухе», как первый, а заземляется. Таким образом, если подать на вход управления ключом в составе 561КТ3 сигнал логической единицы, то вывод IN/OUT соответствующим образом подключенного ключа комму-

тируется на вход OUT/IN, а если сигнал управления равен логическому нулю, то вывод IN/OUT замыкается на «землю», как нам и нужно.

Заметки на полях

Отметим, что есть еще микросхема 176КТ1 (CD4016A, в 561-й серии ей аналога нет, но есть импортная версия CD4016B с питанием до 20 В), с которой 561КТ3 часто путают — у нее ключи самые обычные двусторонние, без заземления. И, несмотря на то, что в классическом справочнике [18] эти микросхемы описаны исчерпывающим образом, в сетевых самодеятельных справочниках по поводу 561КТ3 нередко приводятся ошибочные сведения. Самим строить такие ЦАП, конечно, вряд ли придется, но на всякий случай следует учесть, что сопротивление ключа 561КТ3, как и более современных модификаций (1561КТ3 или CD4066B), довольно велико, порядка сотни ом, что может сказываться на точности. Хотя для практических целей в ряде схем (но не в рассматриваемой!) важнее не абсолютное значение сопротивления, а разница в этом параметре между ключами, которая, если верить справочникам, не превышает 5 Ом.

Рассмотрим, наконец, как же работает такая схема. Для лучшего уяснения принципов я нарисовал всего лишь двухразрядный вариант. Два разряда — это четыре градации, т. е. выходное напряжение ОУ должно принимать 4 значения с равными промежутками, в данном случае эти напряжения равны 0, а также ¼, ½ и ¾ от опорного напряжения $U_{оп}$. Как это происходит?

Рассмотрим сначала схему в исходном состоянии, когда на входах управления ключами код имеет значения «00». Так как оба нижних по схеме резистора $2R$ в исходном состоянии присоединены к «земле», т. е. включены параллельно, то их суммарное сопротивление равно R.

Тогда верхний по схеме резистор R и эти два резистора образуют делитель, напряжение на котором равно ровно половине от $U_{оп}$. Параллельный делителю резистор $2R$ в делении напряжения не участвует. Ключи разомкнуты, цепочка резисторов отсоединена от входа ОУ, и на его выходе будет напряжение, равное 0.

Пусть теперь код примет значение «01». В этом случае резистор с номиналом $2R$ младшего разряда (нижнего по схеме) переключается ко входу усилителя. Для самой цепочки резисторов R–$2R$ все равно, к «земле» присоединен этот резистор или ко входу, потому что потенциал входа ОУ равен тому же потенциалу «земли». Таким образом, ко входу ОУ через сопротивление с номиналом $2R$ потечет ток, величина которого будет равна величине напряжения на его входе ($U_{оп}/2$, как мы выяснили), деленной на величину этого резистора ($2R$). Итого значение тока будет $U_{оп}/4R$, и ток этот создаст на резисторе обратной связи ОУ, сопротивление которого равно R, падение напряжения, равное $U_{оп}/4$. Можно считать и по-другому — рассматривать инвертирующий усилитель с коэффициентом усиления 0,5, что определяется отношением сопротивлений R/2R и напряжением на входе $U_{оп}/2$. Тогда на выходе всей схемы будет напряжение $U_{оп}/4$ (но с обратным знаком, т. к. усилитель инвертирующий).

Пусть теперь код принимает значение «10». Тогда все еще проще — ко входу ОУ подключается напряжение $U_{оп}$ через верхний резистор $2R$. Коэффициент усиления тот же самый (0,5), так что на выходе будет напряжение $U_{оп}/2$. Самый сложный случай — когда код принимает значение «11», и подключаются оба резистора.

В этом случае ОУ надо рассматривать как аналоговый сумматор (см. *главу 12*, рис. 12.5, *а*). Напряжение на выходе будет определяться суммой токов через резисторы $2R$, умноженной на величину сопротивления обратной связи R, т. е. будет равно $(U_{оп}/2R + U_{оп}/4R)\cdot R$, или просто $3U_{оп}/4$.

Я так подробно рассмотрел этот пример, чтобы наглядно продемонстрировать свойства цепочки R–2R. Способ ее построения с любым количеством звеньев показан на рис. 17.3, *б*. Крайние резисторы $2R$ включены параллельно и в сумме дают сопротивление R, поэтому следующее звено оказывается состоящим из тех же номиналов по $2R$ и в сумме тоже даст R и т. д. Какой бы длины цепочку ни сделать, она будет делить входное напряжение в двоичном соотношении: на самом правом по схеме конце цепочки будет напряжение $U_{оп}$, на следующем отводе $U_{оп}/2$, на следующем $U_{оп}/4$ и т. д.

Поэтому с помощью всего двух типономиналов резисторов, отличающихся ровно в два раза, можно строить ЦАП в принципе любой разрядности. Так, восьмиразрядный ЦАП будет содержать 16 резисторов и 8 ключей (если с переключением, как в 561КТ3), не считая резистора обратной связи, который у нас для наглядности был равен также R, но может быть любого удобного номинала. В интегральных ЦАП часто этот резистор вообще не устанавливают заранее, а выносят соответствующие выводы наружу, так что можно легко получать любой масштаб напряжения по выходу. Например, если в нашей схеме сделать этот резистор равным $1{,}33R$, то на выходе мы получим напряжения, равные $U_{оп}$, $2U_{оп}/3$, $U_{оп}/3$ и 0.

Правда, неудобство в такой простейшей схеме заключается в том, что выходные напряжения будут с обратным знаком, но эта проблема легко решается. На рис. 17.3, *в* показан простейший вариант ЦАП с «нормальным» положительным выходом. Проанализировать работу этой схемы я предоставляю читателю самостоятельно — она, вообще-то, даже проще, чем инвертирующий вариант. Недостатком этого варианта по сравнению с инвертирующим будет то, что коэффициент усиления не регулируется (сопротивление в обратной связи должно быть примерно равно R, чтобы минимизировать влияние разности токов смещения), и масштаб будет определяться только величиной $U_{оп}$. Но и этот недостаток легко исправить небольшим усложнением схемы. Такие ЦАП называют еще *перемножающими*.

Заметки на полях

Я не стану рассматривать серийные интегральные схемы ЦАП (например, 572ПА1), основанные на этом принципе, потому что в целом они работают так же, а ЦАП сами по себе, без использования в составе АЦП, требуются нечасто. Тем не менее, скажем несколько слов о проблемах, связанных с метрологией. Ясно, что получить точные значения резисторов при изготовлении микросхемы подобного ЦАП непросто, поэтому на практике абсолютные величины R могут иметь довольно большой разброс. Между собой номиналы их тщательно согласовывают с помощью лазерной подгонки. Собственное сопротивление ключей также может оказывать большое влияние на работу схемы, особенно в старших разрядах, где токи больше, чем в младших. В интегральном исполнении даже делают эти ключи разными — в старших разрядах ставят более мощные с меньшим сопротивлением. А если попытаться сделать самодельный ЦАП на основе упомянутых ранее 516КТ3, то величина R должна составлять десятки килом, не менее, иначе ключи начнут вносить слишком большую погрешность.

Еще один момент связан с получением стабильного опорного напряжения, поскольку это непосредственно сказывается на точности преобразования, причем абсолютно для всех АЦП и ЦАП, как мы увидим далее. В настоящее время успехи электроники позволили почти забыть про эту проблему — все крупные производители выпускают источники опорного напряжения, позволяющие достигать стабильности порядка 16 разрядов (т. е. 65 536 градаций сигнала). К тому же всегда можно исхитриться построить схему так, чтобы измерения стали относительными.

Быстродействие ЦАП рассмотренного типа в основном определяется быстродействием ключей и типом применяемой логики, и в случае КМОП-ключей не слишком высокое — примерно такое же, как у обычных КМОП-элементов.

Большинство интегральных ЦАП построено с использованием описанного принципа суммирования взвешенных токов или напряжений. Другой класс цифроаналоговых преобразователей составляют *интегрирующие* ЦАП, которые служат для преобразования величин, меняющихся во времени. Эти ЦАП в идеале позволяют сразу получить действительно аналоговый, непрерывный сигнал без признаков ступенек. В *главе 22* мы рассмотрим один из широко распространенных методов такого преобразования, использующихся в том числе для воспроизведения цифрового звука, — метод широтно-импульсной модуляции.

АЦП

Номенклатура аналого-цифровых преобразователей существенно больше, чем ЦАП. Однако все разнообразие их типов можно свести к трем разновидностям: это АЦП параллельного действия, АЦП последовательного приближения и интегрирующие АЦП. Рассмотрим их по порядку.

АЦП параллельного действия

АЦП параллельного действия — это зеркально отраженный простейший ЦАП на основе дешифратора, описанный в предыдущем разделе. В таких АЦП имеется делитель из k одинаковых резисторов, к каждой ступени которого подключен компаратор, сравнивающий напряжение на делителе с входным сигналом. Выходы компараторов образуют равномерный код, вроде того, что используется для управления шкальными индикаторами в описанном ранее простейшем ЦАП. Эти выходы подключены к шифратору с k входами, который преобразует этот код в двоичный с числом разрядов n, равным $\log_2(k)$.

Трудности на этом пути уже описывались: схема получается крайне громоздкая, для n-разрядного кода требуется $k = 2^n$ резисторов и компараторов, причем резисторов точно согласованных между собой, и компараторов также с как можно более идентичными характеристиками. Поэтому такие АЦП с разрядностью, большей 8, почти и не выпускают. А зачем их делают вообще? По одной простой причине — этот тип АЦП является самым быстродействующим из всех, преобразование происходит фактически мгновенно и лимитируется только быстродействием применяемых компараторов и логики. Фактическое быстродействие АЦП такого типа

может составлять десятки и сотни мегагерц (наиболее экстремальные типы, как MAX108, допускают частоты до единиц гигагерц). Все остальные типы АЦП, как мы увидим, работают значительно медленнее.

АЦП последовательного приближения

АЦП последовательного приближения мы рассмотрим чуть подробнее — ввиду их практической важности. Хотя самим в настоящее время такие АЦП строить также не приходится, но для успешного использования их в интегральном исполнении следует хорошо понимать, как они работают. Именно такого типа АЦП обычно встроены в микроконтроллеры (см. *главы 18* и *20*).

Главная деталь АЦП последовательного приближения — ЦАП нужной разрядности (именно поэтому мы рассматривали ЦАП раньше, чем АЦП). На его цифровые входы подается код по определенному правилу, о котором далее. Выход ЦАП соединяется с одним из входов компаратора, на другой вход которого подается преобразуемое напряжение. Результат сравнения подается на схему управления, которая связана с регистром — формирователем кодов.

Есть несколько вариантов реализации процедуры преобразования. Самый простой выглядит следующим образом: сначала все разряды кода равны нулю. В первом такте самый старший разряд устанавливается в единицу. Если выход ЦАП при этом превысил входное напряжение, т. е. компаратор перебросился в противоположное состояние, то разряд возвращается в состояние логического нуля, в противном же случае он остается в состоянии логической единицы. В следующем такте процедуру повторяют для следующего по старшинству разряда. Такой метод позволяет за число тактов, равное числу разрядов, сформировать в регистре код, соответствующий входному напряжению. Способ довольно экономичен в смысле временных затрат, однако имеет один существенный недостаток — если за время преобразования входное напряжение меняется, то схема может ошибаться, причем иногда вплоть до полного сбоя. Поэтому в такой схеме обязательно приходится ставить на входе устройство выборки-хранения, о котором далее.

В другой модификации этой же схемы для формирования кодов используется реверсивный счетчик, подобный 561ИЕ11, с нужным числом разрядов. Выход компаратора попросту подключают к выводу переключения направления счета. Изначально счетчик сбрасывают в нули во всех разрядах, после чего подают на него тактовые импульсы. Как только счетчик досчитает до соответствующего значения кода, и выход ЦАП превысит входное напряжение, компаратор переключает направление счета, и счетчик отрабатывает назад. После окончания этого периода установления, если напряжение на входе не меняется, величина кода все время колеблется в пределах младшего разряда. Здесь выбросы не так страшны, но большое время установления и неизвестное заранее время реакции на быстрые изменения входного сигнала являются недостатками такого АЦП, получившего название *следящего*.

Теперь об устройствах выборки-хранения (УВХ). В простейшем случае это все тот же аналоговый электронный ключ, на вход которого подается измеряемый сигнал,

а на выходе стоит конденсатор. До начала измерения ключ открыт, и напряжение на конденсаторе повторяет входное напряжение со всеми его изменениями. В момент начала измерения ключ запирается, и в дальнейшем в качестве измеряемого фигурирует уже напряжение, запасенное на конденсаторе, а изменения на входе на измерительную схему не влияют.

Все, казалось бы, просто, но наличие УВХ, прежде всего, достаточно сильно замедляет процесс, т. к. ключ имеет конечное сопротивление и вместе с конденсатором образует ФНЧ, который требует времени для установления нового значения напряжения и может искажать форму сигнала. Кроме того, как бы ни было велико входное сопротивление компаратора, оно конечно, да и ключ также имеет не бесконечно большое сопротивление в закрытом состоянии. Иногда в схеме присутствует и элемент для принудительного сброса конденсатора (обнуления его), наконец, конденсатор также имеет собственные утечки — все это вынуждает увеличивать емкость конденсатора и еще больше снижать быстродействие схемы. В интегральных АЦП подобного рода нередко даже предоставляется выбор между точностью и быстродействием.

Кроме выборки-хранения, в АЦП последовательного приближения требуется также время на вывод данных и подготовку к следующему циклу измерения. Все указанные причины приводят к тому, что наиболее распространенные 10–12-разрядные АЦП последовательного приближения имеют реальное быстродействие не выше 50–200 кГц. Как пример достаточно продвинутой модели приведем MAX1132, который имеет разрешение 16 битов при частоте выборок 200 кГц. Тем не менее, АЦП последовательного приближения очень распространены и применяются там, где требуется средняя точность при достаточно высоком быстродействии.

Интегрирующие АЦП

Наиболее точными и одновременно самыми медленными являются интегрирующие АЦП. Их мы рассмотрим наиболее подробно, потому что, во-первых, они могут быть достаточно просты схемотехнически, и иногда даже целесообразно самому соорудить такой узел схемы на дискретных элементах, чем подбирать подходящий чип, и, во-вторых, этот тип АЦП наиболее часто применяется в радиолюбительской практике (если не считать встроенных в микроконтроллеры АЦП последовательного приближения). Далее в этой главе мы сконструируем на основе готового АЦП такого типа цифровой термометр с достаточно хорошими характеристиками.

Разных типов интегрирующих АЦП вообще-то не меньше десятка, но здесь мы подробно рассмотрим только три разновидности. Кстати, интегрирующие АЦП являются примером того, что цифровая техника вовсе не всегда достигает наивысшей точности в сравнении с аналоговой — центральным узлом этих, как мы уже сказали, наиболее точных преобразователей, является чисто аналоговый интегратор на ОУ.

Схема самого простого интегрирующего АЦП показана на рис. 17.4. Это так называемый *АЦП с однократным интегрированием*. В начале преобразования на вход C динамического D-триггера поступает положительный фронт, который устанавлива-

ет выход Q в состояние логической единицы. Она является разрешающим уровнем для элемента «И-НЕ», и на вход счетчика поступают импульсы. Одновременно через выход $\overline{Q}$ запирается транзистор VT1. Конденсатор начинает заряжаться от источника стабильного тока. При равенстве значения входного измеряемого напряжения и напряжения на конденсаторе компаратор срабатывает и обнуляет триггер («ворота» на логическом элементе «И-НЕ» запираются, транзистор открывается и разряжает конденсатор, счетчик обнуляется). Количество импульсов, накопленных в счетчике к этому моменту, пропорционально входному напряжению.

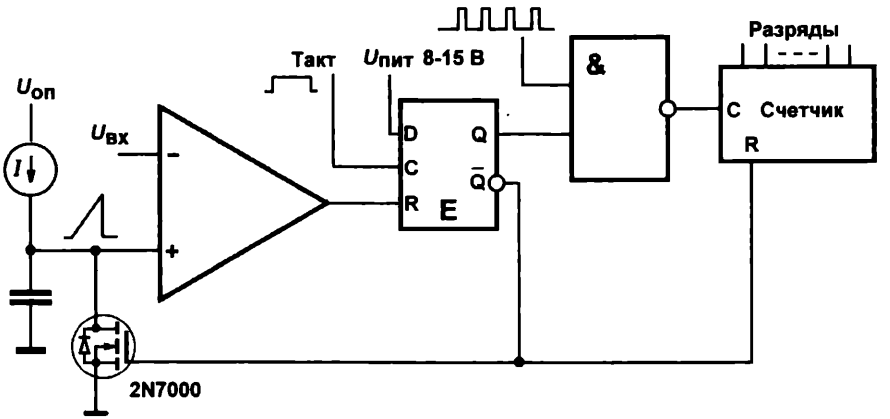


Рис. 17.4. АЦП однократного интегрирования

Источник тока вместе с конденсатором в данном случае образуют так называемый *ГЛИН* — генератор линейно изменяющегося напряжения. Схему можно упростить, если вместо источника тока поставить простой резистор, питающийся от стабильного источника напряжения, но так как форма кривой нарастания напряжения при этом не линейная, а экспоненциальная (см. рис. 5.7 в *главе 5*), то приходится ограничиться небольшим диапазоном входных напряжений, где форма кривой еще близка к прямой линии. Однако на практике так часто и поступают, поэтому источник тока я подробно не рисовал.

Если все же задаться целью расширения входного диапазона вплоть до значений, близких к напряжению питания, то придется делать «нормальный» источник тока. Использование простого полевого транзистора, как мы делали в схеме лабораторного источника питания (см. рис. 9.12), не выход, т. к. он в общем-то является достаточно грубым источником. С другой стороны, чем городить источник тока (например, по варианту, представленному на рис. 12.5, *г*), проще вообще построить ГЛИН по-иному, в виде обычного интегратора по рис. 12.5, *б*, только добавив к нему ключ для сброса по окончании преобразования.

Подробности

Для сброса можно использовать вместо полевого обычный маломощный *n-p-n*-транзистор, но диапазон входного напряжения будет тогда ограничен еще и снизу значением напряжения на коллекторе открытого транзистора (примерно 0,3 В). Поэтому при

снижении питания до 5 В лучше для сброса взять электронный ключ вроде 561КТ3. При конструировании таких схем на микроконтроллерах (см. далее) для сброса конденсатора можно применить тот же вывод порта, который является входом компаратора, если его переключать на вход в рабочем цикле и на выход с нулевым уровнем для сброса.

У схемы на рис. 17.4 единственное достоинство — простота, и куча недостатков. При взгляде на нее непонятно, чего это я ранее распинался насчет выдающихся характеристик интегрирующих АЦП. Результат преобразования здесь зависит от всего на свете: от стабильности источника тока и самого ГЛИН (и каждого его элемента в отдельности, в первую очередь — конденсатора), от стабильности порога компаратора, от неидеальности ключа для сброса и т. п. Еще хуже то, что схема в таком варианте срабатывает от мгновенного значения входного сигнала и потому весьма восприимчива к его дребезгу и вообще к любым помехам. А если тактовая частота случайно окажется кратной частоте помехи (в первую очередь сетевой с частотой 50 Гц), то мы вообще можем получать каждый раз значения, весьма далекие от истины[1]. Поэтому такая схема годится лишь для измерения сигналов постоянного тока — для контроля напряжения батареек или чего-нибудь в этом роде (подобная схема, например, ранее применялась в компьютерном игровом порту для измерения положения привязанного к движку потенциометра управляющего рычага джойстика).

В то же время преобразование длится все равно достаточно долго, поскольку обычные значения тактовой частоты, при которых схема еще работает приемлемо, лежат в диапазоне максимум десятков килогерц (если, конечно, специально не использовать быстродействующие компараторы и логику), т. е. для достижения разрешающей способности в восемь разрядов (больше все равно не выжмешь) частота отсчетов составит в лучшем случае 100 Гц, на практике же еще меньше. Может быть, использовать этот факт и измерять не мгновенное, а среднее значение сигнала за время преобразования?

Сделать это несложно — достаточно подать измеряемое напряжение на вход ГЛИН, а опорное — на компаратор. Тогда сигнал станет интегрироваться за время преобразования, причем интегрироваться очень точно, и мы станем получать истинное среднее арифметическое значение сигнала за это время. Но легко увидеть, что сама функция преобразования при этом окажется обратной, — т. е. время заряда (и значение выходного кода на счетчике) окажется *обратно* пропорциональным значению входного напряжения. Это неудобно, т. к. сильно усложняет обработку результата. Можно применить какой-нибудь хитрый метод деления частоты с использованием реверсивного счетчика, можно также попробовать инвертировать входной сигнал и затем сдвинуть его в положительную область, но все это приводит к усложнению схемы, причем неоправданному, — сама по себе точность преобразования в любом случае не увеличится, избавляемся мы только от помехи.

---

[1] На практике добиться полной некратности частоты измерения и помехи можно, только если сделать тактовую частоту изменяющейся по случайному закону. Так как отношения обычных чисел всегда образуют периодическую дробь, то на выходе мы получим биения выходной величины с частотой повторения периода этой дроби.

По всем этим причинам АЦП с однократным интегрированием, несмотря на его простоту, в настоящее время не употребляют вообще и даже не выпускают в виде специализированных микросхем. Единственная область, где можно было бы рекомендовать такой метод, — использование микропроцессоров, имеющих встроенный компаратор. В этом случае с помощью одного внешнего резистора и конденсатора можно получить простейший преобразователь аналогового сигнала в код. Но и эта рекомендация потеряла в настоящее время всякий смысл, т. к. доступны микроконтроллеры со встроенными «нормальными» АЦП без всяких внешних элементов, причем мультиканальными, с гарантированной точностью и разрешением до 10 и даже 12 разрядов, чего для большинства практических нужд более чем достаточно.

Пожалуй, рассказ об АЦП однократного интегрирования получился чересчур затянутым, но это оправданно, т. к. мы теперь знаем, к чему нам стремиться. И я предвкушаю изумление читателя, когда он узнает, как можно преодолеть чуть ли не все перечисленные здесь недостатки, как говорится, одним махом, и притом не слишком усложняя схему. Интегрирующие АЦП не получили бы такого распространения и заслуженной репутации «самых стабильных», если бы не это обстоятельство.

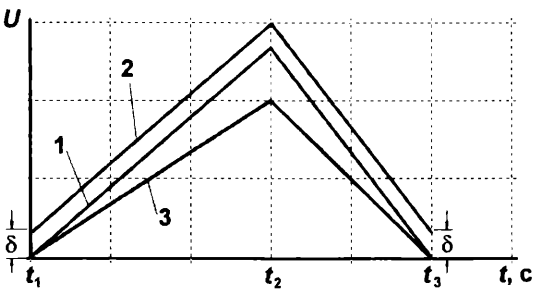


Рис. 17.5. Цикл работы АЦП двойного интегрирования: 1 — идеальный случай; 2 — при сдвиге порога компаратора; 3 — при изменении емкости конденсатора

Идея метода, который называется *двойным* или *двухстадийным* интегрированием, показана на рис. 17.5. Посмотрим сначала на график, обозначенный цифрой 1. В первую часть цикла работы за фиксированное время такта $t_2 - t_1$ конденсатор интегратора заряжается током, который определяется входным (измеряемым) напряжением $U_{вх}$. Во второй части этот конденсатор разряжается точно известным током, определяющимся опорным напряжением $U_{оп}$, до момента равенства напряжения нулю (t_3). Чем больше входное напряжение, тем до большей величины зарядится конденсатор в первой части и тем дольше он будет разряжаться во второй. Легко показать, что отношение интервала времени $t_3 - t_2$ к известному времени такта $t_2 - t_1$ будет равно отношению входного напряжения $U_{вх}$ к опорному $U_{оп}$. Таким образом, измерив полученный интервал времени $t_3 - t_2$ обычным методом с помощью счетчика, как это сделано в схеме на рис. 17.4, мы получим на выходе код, пропорциональный входному напряжению.

На самом деле напряжение, до которого разряжается конденсатор, задается порогом компаратора и может в общем случае быть отличным от нуля на величину δ за

счет «гуляния» порога, например, при изменении температуры. Но так как в начале цикла измерения напряжение определялось тем же значением порога, то, как вы видите из графика 2 на рис. 17.5, в данном случае имеет значение только изменение порога за время преобразования. А оно даже в самых «неповоротливых» АЦП такого типа не превышает долей секунды, потому это изменение можно не принимать в расчет. На результате не скажется и изменение емкости конденсатора, поскольку при этом наклон прямой и заряда и разряда изменится в одинаковой степени (график 3).

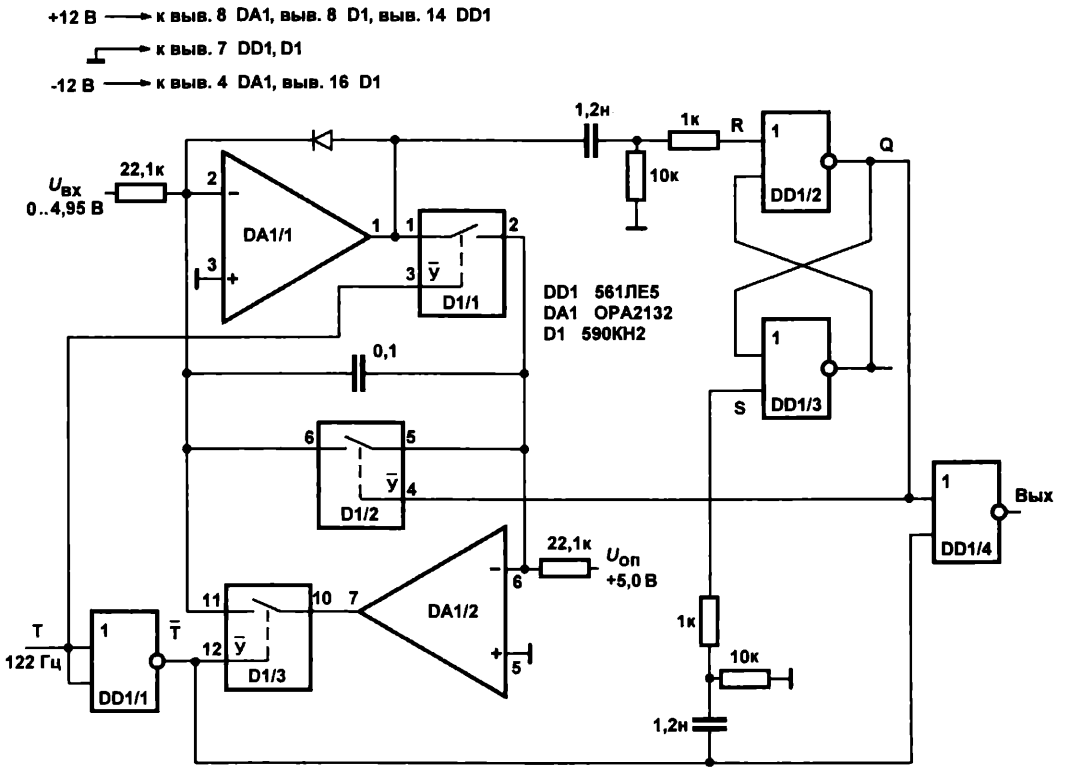
В самых точных АЦП такого типа дополнительно проводят цикл *автокоррекции нуля*, когда на вход подают нулевое напряжение и результат потом вычитают из значения кода, полученного в рабочем цикле. Мало того, здесь даже не требуется «кварцованная» частота, и всю схему можно заводить от любого RC-генератора при условии, что время такта $t_2 - t_1$ и частота заполнения «ворот» для подсчета длительности результирующего интервала $t_3 - t_2$ задаются от одного и того же генератора.

Но чудес не бывает — точность и стабильность преобразования здесь полностью определяются точностью и стабильностью значения $U_{оп}$. От этого никуда не денешься, и, как мы говорили, это общее условие для всех без исключения конструкций АЦП и ЦАП. Между прочим, обратите внимание, что $U_{вх}$ и $U_{оп}$ образуют в совокупности нечто вроде неинвертирующего и инвертирующего входа ОУ. Эта аналогия куда более полная, чем кажется, и, манипулируя этими величинами, можно выделывать с выходным кодом всякие штуки, в частности, подгонять масштаб преобразования к нужному диапазону. Другое облегчение, которое можно получить от этой связи, заключается в возможности проведения относительных измерений, когда входное и опорное напряжения получаются от одного источника и тем самым имеют одинаковую относительную погрешность (получается нечто вроде явления ослабления синфазного сигнала в ОУ).

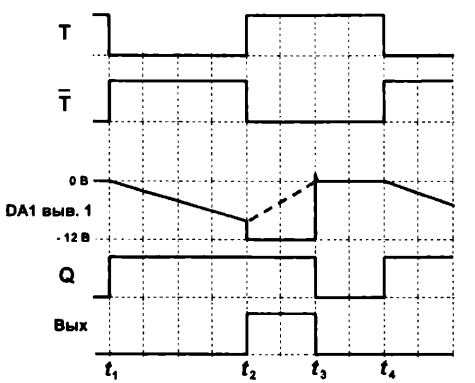
Кстати, в интегрирующих АЦП такого рода для более полного подавления помех нужно делать первую часть цикла интегрирования именно кратным периоду помехи. Тогда в цикле укладывается целое число периодов помехи, и она усредняется. Практически наибольшее влияние оказывает сетевая помеха частотой 50 Гц, поэтому частоту циклов стараются делать в круглых числах.

Простой вариант практической схемы АЦП двойного интегрирования (преобразователя напряжение-время, ПНВ) приведен на рис. 17.6, *а* (счетная часть на схеме не показана). Для понимания того, как работает схема, следует обратить внимание, что управляющий вход Y у ключей типа 590КН2 инверсный, т. е. при низком уровне на управляющем входе ключ распахнут, а при высоком — заперт.

Рассмотрим диаграмму работы этой схемы (рис. 17.6, *б*). В момент отрицательного перепада на тактовом входе T RS-триггер устанавливается в единицу по выходу Q. Так как на входе T в этот момент отрицательный уровень, ключ D1/1 открывается, остальные ключи заперты. Конденсатор подключается в обратную связь верхнего ОУ (DA1/1), и начинается цикл интегрирования входного напряжения (напряжение на конденсаторе возрастает по абсолютной величине, т. е. на выходе DA1/1 падает,



а



б

Рис. 17.6. *а* — простой вариант АЦП двойного интегрирования (ПНВ); *б* — диаграммы напряжений в различных точках схемы

поскольку интегратор инвертирующий). В момент окончания отрицательного полупериода тактовой частоты ключ D1/1 запирается, а ключ D1/3 открывается, заряженный конденсатор оказывается подключенным в обратную связь второго ОУ (DA1/2). Начинается цикл интегрирования опорного напряжения (изменение напряжения на конденсаторе показано на диаграмме пунктирной линией). Поскольку обратная связь в первом ОУ теперь отсутствует, то он сработает как компаратор — сначала на его выходе установится напряжение, равное отрицательному питанию (или близкое к нему), а в момент равенства напряжения на конденсаторе нулю выход резко устремится от отрицательного к положительному питанию (но его ограничит на уровне примерно +0,6 В включенный в обратную связь диод, который нужен для того, чтобы не затягивать переходный процесс). Положительный перепад передастся на обнуляющий вход RS-триггера и установит его выход Q в состояние логического нуля. При этом откроется ключ D1/2 и закоротит конденсатор, прерывая таким образом процесс интегрирования. На входе верхнего ОУ установится напряжение, равное нулю, а на выходе, вообще говоря (т. к. обратная связь по-прежнему отсутствует), оно станет неопределенным, и на диаграмме показано условно в виде нулевого уровня.

Это состояние длится до конца периода тактовой частоты, а с отрицательным перепадом на входе Т ключи D1/3 и D1/2 закроются, и все начнется сначала. На выходе схемы образуется положительный импульс напряжения, длительность которого $t_3 - t_2$ пропорциональна входному напряжению, согласно соотношению, сформулированному ранее.

Схема рассчитана для получения разрешающей способности 12 разрядов или 4096 градаций. Стабильность схемы напрямую зависит от стабильности резисторов, поэтому их нужно выбирать с точностью не хуже 0,1%, — в этом случае абсолютная точность может достигнуть 10 разрядов без дополнительной калибровки. Однако $U_{оп}$ тоже должно иметь не меньшую стабильность, поэтому для его получения следует использовать прецизионные источники опорного напряжения. В данном случае подойдет микросхема MAX875, дающая на выходе 5 В с точностью 0,04%. Подробный анализ всех погрешностей этой схемы, в том числе температурных, занял бы слишком много места, поэтому рассмотрим еще только принцип выбора частоты преобразования и требования к элементам.

Максимальная частота отсчетов может быть подсчитана из следующих соображений. Так как мы имеем дело с КМОП, то максимальную частоту счетных импульсов примем равной 1 МГц. Нам требуется обеспечить 12 разрядов, т. е. число импульсов за время «ворот» при максимально возможном входном напряжении, равном опорному, должно составить как минимум 4096 штук. Поделив 1 МГц на это число, мы получим частоту около 244 Гц, однако ее надо еще уменьшить вдвое, поскольку у нас в рабочем периоде должно быть два таких такта: прямого и обратного интегрирования. Итого получаем 122 Гц, что и есть максимальная частота при выбранной элементной базе. Исходя из этого подобраны величины сопротивлений и емкость конденсатора. При указанных на схеме их величинах, напряжение на выходе интегратора при входном напряжении 5 В достигнет примерно 9 В за время интегрирования, равное половине периода частоты 122 Гц.

Входное напряжение ограничено для этой схемы диапазоном от нуля до примерно 4,95 В. Напряжение выше этого значения расстроит работу схемы, потому что импульс обнуления за счет RC-цепочки все еще будет длиться, когда придет импульс установки. Импульс обнуления можно было бы сократить, например, за счет введения «корректной» дифференцирующей цепочки (см. рис. 16.7, *а*), но к ограничению уровня входного напряжения ведет и другое обстоятельство, а именно — конечное время разряда конденсатора через ключ при приведении схемы в исходное состояние. При использованных на схеме элементах и при условии достаточно полного разряда оно составит не менее 20–30 микросекунд (сопротивление ключа около 50 Ом), т. е. до 1% от максимальной длительности, что и ограничивает время рабочего импульса и максимальное напряжение примерно на ту же величину. Избавиться от этого можно только усложнением схемы и введением дополнительного интервала специально для обнуления — в серийных АЦП так и поступают.

О выборе элементов. При указанных частотах скорость нарастания сигнала на выходе верхнего по схеме ОУ, служащего компаратором, должна быть такой, чтобы сигнал изменялся от напряжения насыщения до нуля не более чем в пределах одного импульса счетной частоты, длящегося 1 мкс. То есть скорость нарастания должна быть не меньше 10 В/мкс, иначе мы получим ошибку за счет неточного определения момента окончания интегрирования (то же требование справедливо и для скорости срабатывания ключей). Второе требование к ОУ — для более точного интегрирования желателен достаточно малый входной ток смещения, не более нескольких наноампер. Он рассчитывается исходя из величины максимального тока интегрирования, в данном случае около 250 мкА, деленного на ту же величину в 12 разрядов, т. е. 4096. Входной ток ОУ должен удовлетворять условию «много меньше», чем полученная величина около 60 нА.

Если принять во внимание допустимое напряжение питания (не менее 12 В), то не так уж и много ОУ удовлетворят указанным требованиям. Микросхема OPA2132 (два OPA132 в одном корпусе DIP-8) фирмы Texas Instruments представляет собой прецизионный ОУ с высоким быстродействием (полоса 8 МГц, скорость нарастания до 20 В/мкс), очень малым входным током смещения (50 пикоампер) и высоким допустимым напряжением питания до ±18 В. Из классических отечественных ОУ в коридор требований с некоторой натугой влезет 544УД2 или некоторые ОУ серии 574. Впрочем, номенклатуру пригодных чипов можно значительно расширить, если снизить напряжение питания до ±5 В (при этом допустимый диапазон входного напряжения необязательно снизится, т. к. оно может превышать напряжение питания, просто манипулировать многими питаниями неудобно) и/или уменьшить частоту счета, например, до 100 кГц (частота отсчетов снизится до 12 Гц, а требования к быстродействию ОУ соответственно упадут). Все это иллюстрирует сложности, которые приходится преодолевать разработчикам при проектировании подобных АЦП в интегральном исполнении, и объясняет, почему интегрирующие АЦП обычно работают так медленно — у большинства прецизионных АЦП частота отсчетов не превышает величины несколько десятков или сотен герц.

Сконструированное нами АЦП относится к типу ПНВ — преобразователей напряжение-время. Ранее широко использовались ПНЧ — преобразователи напряжение-

частота, однако большинство их реализаций обладает тем же недостатком, что и однократный интегратор, т. е. в них точность зависит от качества компонентов напрямую. Радиолюбители почему-то обожают изобретать ПНЧ (точнее, ГУН — генератор, управляемый напряжением) на основе микросхемы 555 (см. *главу 16*), но если вдруг придется заниматься ПНЧ, учтите, что это крайне неудачный вариант. Имеется достаточно специализированных микросхем для этой цели, которые выполняют ту же задачу гораздо лучше (например, LM331).

Сейчас мы рассмотрим интегрирующий преобразователь, который также использует двойное интегрирование, но на выходе его получается не интервал времени, который еще нужно сосчитать, а число-импульсный код, т. е. сразу число импульсов за определенный промежуток времени, пропорциональное входному напряжению. Это не частота, как можно бы подумать, точнее, не совсем частота.

АЦП такого типа (преобразователи напряжение-код, ПНК) называются еще *дельта-сигма-преобразователями* или АЦП с уравновешиванием заряда. Они широко распространены в интегральном исполнении, большинство наиболее высокоразрядных АЦП построены именно так. Я не буду рисовать подробную схему с указанием типов компонентов и разводкой выводов, потому что принципы подбора комплектующих сильно зависят от необходимой точности и разрешающей способности (разрядности), а самостоятельно строить такие схемы нет особого резона.

Принципиальная схема ПНК показана на рис. 17.7. Работает она следующим образом. Как только напряжение на выходе интегратора DA1 становится меньше нуля, компаратор D1 переключается, и тактовые импульсы начинают поступать на вход счетчика и одновременно на ключ, коммутируя источник опорного тока к суммирующей точке интегратора. Входной ток $I_{вх}$ и опорный $I_{оп}$ имеют разные знаки и опорный больше по величине, поэтому с каждым тактовым импульсом напряжение на конденсаторе будет уменьшаться, а на выходе интегратора — стремиться к нулю. Как только оно опять сравняется с нулем, компаратор переключится, и тактовые импульсы перестанут поступать на счетчик и на ключ. Заряд, который сообщается интегратору за каждый тактовый импульс, строго одинаков, поэтому количество таких тактовых импульсов в единицу времени N, необходимых для уравновешивания заряда, сообщаемого источником входного напряжения, будет в точности пропорционально входному напряжению. Разумеется, токозадающие

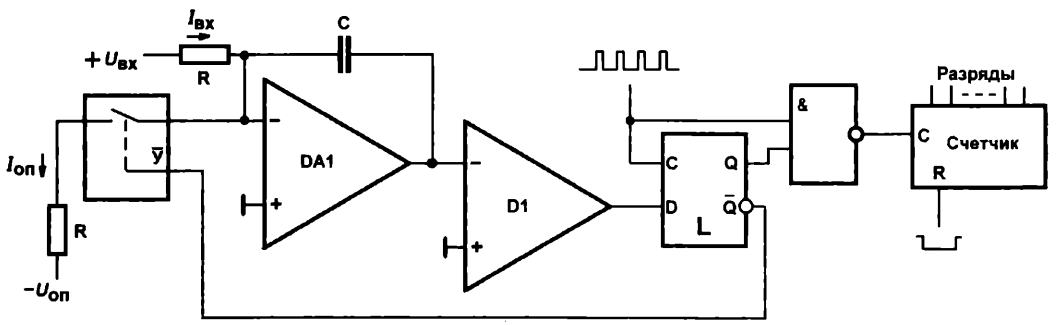


Рис. 17.7. Принцип работы АЦП с уравновешиванием заряда

резисторы в цепи входного и опорного напряжения вовсе не обязаны быть равны друг другу, но в любом случае число N будет пропорционально входному току и обратно пропорционально опорному, если соблюдается соотношение $I_{оп} \geq I_{вх}$. При их равенстве число импульсов N за секунду будет равно тактовой частоте. Манипулируя величиной $U_{оп}$ и номиналами резисторов, можно получать различный масштаб. Отметьте, что импульсы на входе счетчика, представляющие число N, могут быть неравномерно распределены во времени — этим ПНК отличается от ПНЧ.

Здесь точность преобразования зависит практически только от стабильности $I_{оп}$ ($U_{оп}$) — при условии, конечно, выбора остальных компонентов по быстродействию в соответствии с рекомендациями для АЦП двойного интегрирования. Автор этих строк строил схему подобного ПНК на самых что ни на есть рядовых элементах: ключах 590КН2, ОУ 544УД1 и КМОП 561-й серии, в качестве источника тока использовалась схема по типу рис. 12.5, *г* на ОУ 140УД20 и стабилитроне КС170. Тем не менее, при тактовой частоте 2048 Гц (т. е. разрешающей способности 11 разрядов при времени измерения 1 с) стабильность схемы составляла не хуже 3 единиц кода (0,15%) в диапазоне от –18 до +40 градусов! А если тщательно проработать вопрос стабильности и быстродействия элементов, то можно получить нечто вроде MAX1400 — прецизионного 18-разрядного АЦП с быстродействием 4800 отсчетов в секунду.

Конструируем цифровой термометр

Цифровые термометры конструировать самостоятельно имеет смысл по крайней мере потому, что рынок подобных бытовых устройств достаточно беден. Фирменные приборы для расположения на стенке комнаты или офиса обычно имеют невзрачный дизайн с корпусами белого или «компьютерного» серого цвета и с ЖК-индикаторами, которые из-за их «слепоты» я бы категорически не рекомендовал применять в бытовых приборах, особенно тех, что предназначены для разглядывания издалека. Терпеливый радиолюбитель вполне может сделать конструкцию куда лучше фирменной — удобную, красивую и приспособленную под свои нужды, и, самое главное, показывающую правильную температуру, а не «погоду на Марсе». А «приставив» к такому термометру измерители влажности, давления и еще чего угодно, получить настоящую метеостанцию, — вопрос только денег, и мы еще этим будем заниматься.

Но сначала поговорим об одной из самых популярных микросхем АЦП, специально приспособленной для конструирования таких приборов, как цифровые измерители или мультиметры. Впервые обе ее разновидности выпущены более четверти века назад и до сих пор не потеряли своего значения — значительная часть мультиметров, поступающих в продажу, изготовлена на таких микросхемах или их современных аналогах.

АЦП 572ПВ2 и ПВ5

Основой принципиальной схемы нашего термометра станет микросхема 572ПВ2 (ICL7107), которая представляет собой АЦП двойного интегрирования с выходом

в параллельном семисегментном коде с расчетом на 3,5 десятичных разряда. Что означает цифра 3,5 — не может же использоваться полразряда? Действительно, при использовании полного выходного диапазона этой микросхемы, который составляет число ±1999, нужно подключать 4 индикатора, однако последний (старший) из них будет служить только для индикации цифры 1, и, при необходимости, знака минус. Число 3,5 и означает, что старший разряд используется не полностью (бывают и более заковыристые обозначения, вроде 3¾ разряда, но их оставим на совести авторов). Заметим, что разрешающая способность (а при соблюдении некоторых требований — и точность) этого АЦП эквивалентна приблизительно 11 двоичным разрядам, т. е. приведенная погрешность составит 0,05%, что очень и очень неплохо.

Основная (типовая) схема включения микросхемы 572ПВ2 показана на рис. 17.8. Микросхема имеет два собственных питания: положительное 5 В (от 4,5 до 6 В) и отрицательное, которое может варьироваться в довольно большом диапазоне от −9 до −3,5 В. Это обстоятельство позволяет при необходимости использовать для отрицательного питания не слишком стабильные преобразователи-инверторы, о чем далее.

Семисегментные LED-индикаторы можно подключать напрямую, без каких-либо дополнительных резисторов (ток через сегмент при этом равен 5–8 мА). Управле-

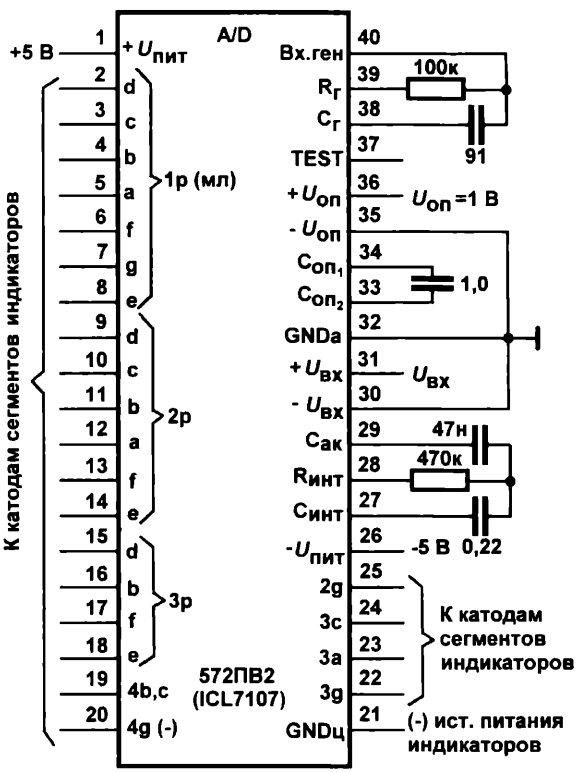


Рис. 17.8. Типовое включение микросхемы 572ПВ2 (ICL7107) в корпусе DIP-40

ние индикаторами осуществляется коммутацией на «землю», поэтому нужен индикатор с общим анодом, который целесообразно подключать к отдельному источнику питания, чтобы не вносить лишние помехи. Однако выходы управления индикатором не являются выходами с открытым коллектором (точнее — истоком), а есть обычный комплементарный КМОП-выход (см. схему инвертора на рис. 15.1, *справа*). Вытекающий ток в состоянии логической единицы может составить примерно 0,5 мА, а в состоянии логического нуля — примерно 5–8 мА (для вывода 19, который управляет одновременно двумя сегментами при засветке 1 в старшем разряде, этот ток составляет 10–16 мА).

Заметки на полях

Это обстоятельство можно использовать для управления индикаторами через внешние ключи. Дело в том, что для питания LED, потребляющих достаточно большой ток (при максимальном количестве зажженных сегментов, т. е. при индикации −1888, он может составить от 120 до 200 мА), естественно было бы использовать нестабилизированное повышенное напряжение, например, от входа стабилизатора положительного напряжения. Это особенно актуально при подключении крупных индикаторов с повышенным падением напряжения, поскольку при напряжении 5 В они будут светиться очень тускло (если загорятся вообще). Однако ставить более 20 штук ключей не очень хочется, если конструкция не слишком капитальная. К сожалению, в технической документации ни один из производителей не упоминает о возможности подключения LED-индикатора к повышенному напряжению. Можно ожидать, что при пиковом значении напряжения питания, не превышающем суммы основного питания (5 В) и падения напряжения на индикаторе (1,8–2 В для обычных и 3,5–4 В для крупных индикаторов), микросхеме ничего не грозит. В крайнем случае, можно поставить небольшие резисторы, ограничивающие ток через защитные диоды. Автор этих строк на свой страх и риск провел долгосрочный эксперимент по питанию LED-индикатора высотой 1 дюйм от пульсирующего напряжения 6,5–7 В с амплитудным значением, соответственно, около 9–10 В (от отдельной обмотки трансформатора через один диод в качестве выпрямителя). Опыт показал, что такой режим микросхема вполне выдерживает. При этом из-за «однополупериодности» напряжения средний ток через сегменты примерно в полтора раза ниже номинального, чего вполне достаточно для нормальной яркости свечения. Здесь мы также применим этот режим питания, однако в некоторых случаях это неудобно, и приходится ставить отдельный мощный стабилизатор, как и положено.

Выпускается и совершенно идентичная по функциональности и разводке выводов микросхема 572ПВ5 (ICL7106), которая отличается только тем, что она предназначена для управления ЖК-индикаторами, а не светодиодными, так что, если есть нужда в малом потреблении, можно почти без изменений основной схемы использовать такой вариант. Просто заменить LED-индикатор на ЖК и наоборот, как мы уже говорили, нельзя, потому что для управления ЖК-индикаторами требуется переменное напряжение, иначе отключенные сегменты «зависнут» в поглощающем свет состоянии. Поэтому при замене ПВ2 на ПВ5 отличие в схеме заключается в том, что вывод 21 представляет собой не «цифровую землю» (GNDц), а подсоединяется к общему выводу ЖК-индикатора. При этом отдельное питание, естественно, не требуется. Управление ЖК-сегментами происходит так: на общем выводе 21 все время присутствует меандр, а на тот сегмент, который нужно засветить, подается точно такой же меандр, но в противофазе. При отключении сегмента фаза на выводе его управления меняется на противоположную и становится такой же, как на выводе 21, поэтому постоянное напряжение на сегмент никогда не подается.

Отдельный вопрос представляет засветка запятой, если ее по ходу дела надо гасить. В LED-варианте это несложно (можно просто засветить постоянно или через какой-то ключ), а для ЖК-варианта нужно для нее также обеспечить подобный режим управления. Иначе при подаче постоянного напряжения она просто засветится навсегда (и будет светиться еще долго после выключения питания) и к тому же станет резко выделяться бóльшим контрастом. Разработчики рекомендуют использовать для этой цели отдельный логический инвертор, подключенный к выходу 21. При этом (как и в случае подключения внешнего тактового генератора, см. далее) в качестве «цифровой земли» в 572ПВ5 следует использовать вывод 37 (TEST).

Ввиду отсутствия у микросхемы ПВ5 «цифровой земли» как таковой, эту микросхему можно питать от одного источника, напряжение которого может составлять от 9 до 15 В (что эквивалентно диапазону от ±4,5 до ±7,5 В). Только при этом не следует забывать, что для обеих микросхем опорное и входное напряжения не должны выходить за пределы, на 1 В отступающие от потенциалов $+U_{пит}$ и $-U_{пит}$. Для микросхемы ПВ2, вообще говоря, требуется двухполярное питание во всех случаях, т. к. «цифровая земля» GNDц должна иметь общую точку с аналоговой частью для внутреннего согласования уровней управляющих сигналов. Однако можно обойтись одним питанием +5 В (подсоединив вход $-U_{пит}$ к «земле»), если, в соответствии с ранее сказанным, опорное и измеряемое напряжения по абсолютной величине не превышают 1,5 В, причем эта величина должна отсчитываться от середины $U_{пит}$.

Есть и более современные варианты этих преобразователей — например, с очень малым потреблением, но параметры рассмотренных микросхем и так достаточно хороши — при тактовой частоте 50 кГц время преобразования составляет 0,32 с (16 000 периодов тактовой частоты), а потребление при этом не превышает 0,6 мА (не считая, конечно, потребления индикаторов в LED-варианте).

Удобство микросхем ПВ2 и ПВ5 заключается и в том, что они оперируют с двухполярными входными напряжениями, автоматически определяя и высвечивая знак. Диапазон входного измеряемого напряжения определяется опорным, с помощью которого и задается масштаб, при этом опорное должно находиться в пределах 0,1–1 В, а измеряемое может по абсолютной величине превышать его, в соответствии с разрешающей способностью, ровно в два раза. Если, например, опорное напряжение равно 1 В, то измеряемое может быть в пределах ±2 В (точнее ±1,999 В), а в общем случае выходной код определяется выражением $N = 1000 \cdot \frac{U_{вх}}{U_{оп}}$. При превышении значением входного напряжения предела $+2U_{оп}$ младшие три разряда гаснут, а при снижении ниже $-2U_{оп}$ — гаснет все, кроме знака минус.

На схеме рис. 17.8 показан именно такой вариант включения с общими «землями». Однако оба входных напряжения — опорное и измеряемое — могут быть и «плавающими», без общей «земли», единственное требование — чтобы их значения не выходили за пределы питания (а по абсолютной величине они, естественно, должны соответствовать указанным ранее требованиям). В этом случае вывод 32 («аналоговая земля») не используется. На этом выводе тогда присутствует напряжение, равное ($U_{+пит}$ – 2,8) В. Если очень надо, его можно использовать в качестве опорно-

го (не само напряжение относительно «земли», которая в данном случае есть довольно условное понятие, а именно разность между положительным питанием и выводом 32). Однако стабильность этого напряжения невелика, и так рекомендуется поступать только в уж очень экономичных схемах. Особенно это плохо в случае ПВ2, в которой выходные каскады за счет большого тока сильно (и неравномерно по времени из-за разного количества подключенных сегментов) нагревают кристалл, и напряжение это начинает «плавать». Ошибка при этом может составить до 0,5%, т. е. точность снижается до 9 разрядов вместо 11.

Тактовую частоту микросхем рекомендуется выбирать из ряда 200, 100, 50 и 40 кГц, при этом частота помехи 50 Гц уложится в длительность фазы интегрирования входного напряжения (см. далее) целое число раз, и такая помеха будет интегрироваться полностью. Тактовую частоту можно задавать тремя способами: с помощью *RC*-цепочки, как показано на рис. 17.8, с помощью кварца, подключаемого к выводам 39 и 40, а также внешним генератором, выход которого подключается в выводу 40 (в ПВ2 при этом в качестве общего провода используется вывод 21 «цифровая земля», а в ПВ5 — вывод 37 «TEST»). На практике чаще всего используется первый способ, при этом частота будет равна примерно $0{,}45R_{\text{г}}C_{\text{г}}$. В фирменной документации на этот счет есть некоторая неясность, т. к. рекомендуется выбирать $R_{\text{г}} = 100$ кОм при $C_{\text{г}} = 100$ пФ, и тогда согласно формуле частота должна составить 45 кГц. Это далеко и от 40, и от 50 кГц, рекомендуемых для частоты помехи 50 Гц, и не вполне совпадает с 48 кГц, рекомендуемыми для помехи 60 Гц. Все отечественные описания микросхем ПВ2 и ПВ5 изящно обходят этот вопрос, просто повторяя фирменные рекомендации. Думается, что составители документации имели в виду все же 60-герцевую помеху (т. е. тактовую частоту 48 кГц), поэтому в отечественных пенатах следует снизить емкость $C_{\text{г}}$ до 91 пФ — так будет корректнее. Вообще, ошибка в ±5%, конечно, тут вполне допустима.

Из особенностей внутреннего функционирования этих микросхем нам интересен еще один момент. Цикл работы ПВ2 и ПВ5 состоит из трех фаз, первые две из которых идентичны циклу работы ПНВ по рис. 17.5. После окончания фазы интегрирования опорного напряжения и формирования собственно измерительного интервала начинается последняя (или первая для следующего измерения) часть цикла, носящая название *фазы автокоррекции*. В этой фазе происходит не только сброс интегрирующей емкости (который у нас в схеме на рис. 17.6 занимал некоторое время из отведенного для фазы интегрирования), но и, кроме этого, на конденсаторе $C_{\text{ак}}$ происходит накопление напряжения смещения всех участвующих в процессе ОУ и компараторов. В рабочих циклах это напряжение учитывается. Но для нас еще интереснее, что в фазе автокоррекции одновременно происходит заряд емкости $C_{\text{оп}}$ до значения опорного напряжения, и последующее интегрирование в рабочем цикле оперирует именно с этой величиной, а вход опорного напряжения при этом отключается. Собственно, сделано это для того, чтобы была возможность автоматического внутреннего инвертирования опорного напряжения при смене знака измеряемого. Однако для нас это важно, потому что позволяет сгладить наличие высокочастотных помех на входе опорного напряжения. К сожалению, длительность фазы автокоррекции является неопределенной (т. к. она занимает всю оставшуюся часть фазы интегрирования опорного напряжения, к которому прибавляется

фиксированный интервал времени в 4000 периодов тактовой частоты), и низкочастотная помеха при этом интегрируется плохо.

Номиналы емкостей и резисторов на рис. 17.8 приведены для случая опорного напряжения, равного 1 В, и тактовой частоты 50 кГц. При опорном напряжении 0,1 В емкость $C_{ак}$ нужно увеличить до 0,47 мкФ, $C_{инт}$ уменьшить до 0,1 мкФ, а $R_{инт}$ уменьшить до 47 кОм. В остальных случаях эти номиналы должны быть изменены в указанных пределах примерно пропорционально изменению опорного напряжения.

К выбору типов компонентов следует подходить весьма тщательно, от этого сильно зависит в первую очередь линейность преобразования. Резисторы все могут быть типа МЛТ, хотя при наличии стоит предпочесть С2-29В. Конденсатор тактового генератора $C_{ген}$ может быть керамическим (типа КМ73-10, КМ-5, КМ-6). Остальные конденсаторы ($C_{инт}$, $C_{оп}$ и $C_{ак}$) должны иметь органический диэлектрик, лучше всего подойдут фторопластовые (К72П-6, К72-9) или полистироловые (К71-4, К71-5), но сойдут и полиэтилентерефталатные (К73-16, К73-17). Эти конденсаторы могут ужаснуть вас своими размерами, но ничего не поделаешь — такова плата за стабильность. Высокие конденсаторы (как К73-17) следует устанавливать лежа — хотя при этом площадь платы увеличивается, но зато конденсаторы не торчат над всеми остальными компонентами. Это, кроме всего прочего, повышает надежность монтажа, ибо меньше вероятность выломать конденсатор с корнем, случайно положив поверх платы каталог продукции фирмы MAXIM.

Практическая схема термометра

Теперь, вооружившись всеми этими знаниями, приступим, наконец, к нашему термометру. И сначала нам надо будет посчитать — что мы имеем на входе и что хотим при этом получить на выходе?

Начнем с выхода — температура традиционно демонстрируется в виде «XX,X». Таким образом, мы должны использовать только три младших разряда, при этом диапазон температур получится от −99,9 до +99,9 °С. Собственно говоря, такой диапазон чересчур широкий, практически для «погодного» термометра хватило бы и диапазона от −50 до +50 °С. В чем и состоит, как мы уже говорили, недостаток использования готовых микросхем — мы вынуждены устанавливать диапазон в соответствии с возможностями отображения чисел с помощью ПВ2. И при этом мы теряем ровно два двоичных разряда, ужимая диапазон в 4 раза. Никто нам, конечно, не запретит подключить все четыре индикатора и демонстрировать температуру от −199,9 до 199,9 °С. Но если диапазон выше 100 °С еще может пригодиться в быту (скажем, признаком готовности варенья служит температура 105–106 °С), то отрицательный диапазон аж до −200 °С вряд ли потребуется даже для самых специфических производственных нужд, а для научных задач такие температуры измеряются своими способами. Но, конечно, никто не запрещает вам использовать, например, половину диапазона со сдвигом, от −50 до +150 °С — все будет определяться соотношением резисторов, как мы увидим, и наличием индикаторов. Калибровку для простоты будем производить от 0 до 50 градусов, полагая (и это оправдывается

на практике), что термодатчик при не слишком большом углублении в отрицательную область ведет себя линейно.

О выборе датчиков мы говорили в *главе 13*. Так как мы собираемся делать более-менее точный прибор, то выберем не полупроводниковый, а медный резистивный датчик и прикинем, какое было бы желательно иметь его сопротивление. Обычные токи через датчик должны составлять порядка 1–3 мА, иначе медная катушка приемлемых размеров будет сама нагреваться. Проще всего в качестве датчика использовать обмотку малогабаритного реле из серий, например, РЭС-60, РЭС-80, РЭС-79 или РЭС-49 — какое окажется под рукой, и чем старше возрастом, тем лучше, т. к. медь при хранении стабилизирует свои характеристики. Нет проблем использовать и любое другое реле, только крупные конструкции будут иметь значительную тепловую инерцию, к тому же многие, особенно старые, реле не герметизированы. Указанные мной типы имеют полностью герметизированный металлический корпус, остается только изолировать от внешней среды выводы. У меня «под рукой» оказалось реле типа РЭС-60 с обмоткой 800±120 Ом (паспорт РС4.569.435-01). Изменения на диапазон 100 °С составят в среднем 320 Ом (напомним, что у меди температурный коэффициент сопротивления равен 0,4 %/°). Выберем $U_{оп}$ = 0,5 В, тогда ток через датчик-обмотку должен составить 0,5 В/320 Ом ≈ 1,5 мА. Так как рабочие напряжения здесь не превышают по абсолютной величине 0,5 В, то мы сможем обойтись для АЦП одним питанием +5 В, только надо будет максимально приблизить эти напряжения к середине питания.

Общая схема термометра приведена на рис. 17.9. Рассмотрим сначала включение датчика. Для того чтобы при нуле градусов термометр показывал 0, нужно на вход АЦП подавать разность текущего напряжения на датчике и значения его при нулевой температуре. В данном случае это делается с помощью мостовой схемы. Два идентичных источника тока 1,5 мА (ОУ DA1, транзисторы VT1–VT2 и резисторы R16–R19) образуют верхнюю половину моста, а нижняя состоит из датчика температуры Rt и опорного резистора R20, сопротивление которого равно сопротивлению датчика при 0 °С. Разность этих напряжений подается на АЦП в качестве входного напряжения. Фильтр R22–C6 нужен для лучшего сглаживания помех (конденсатор C6 может быть керамическим). ОУ MAX478, как указывалось в *главе 12*, можно заменить, например, на OP293 (или, с небольшой переработкой схемы, на счетверенный OP493). Так как в этой схеме общее питание не превышает 5 В, то выбор ОУ с хорошими характеристиками несколько расширяется (OP296, AD8607, AD8616 и др.).

Обратим теперь внимание на хитрую схему включения самого датчика, которая носит название *трехпроводной*. Такая схема позволяет избежать влияния соединительных проводов и, главное, помех, которые наводятся на них. Сами по себе провода влияют слабо, т. к. в данном случае достаточно, чтобы они имели сопротивление, меньшее, чем 1/2000 сопротивления датчика[1], что составляет примерно 0,4 Ом. Это вполне обеспечит провод МГТФ-0,35, если его суммарная длина не

[1] На самом деле это даже слишком жесткое требование, т. к. погрешность вносит не само сопротивление подводящих проводов, а только его температурные изменения.

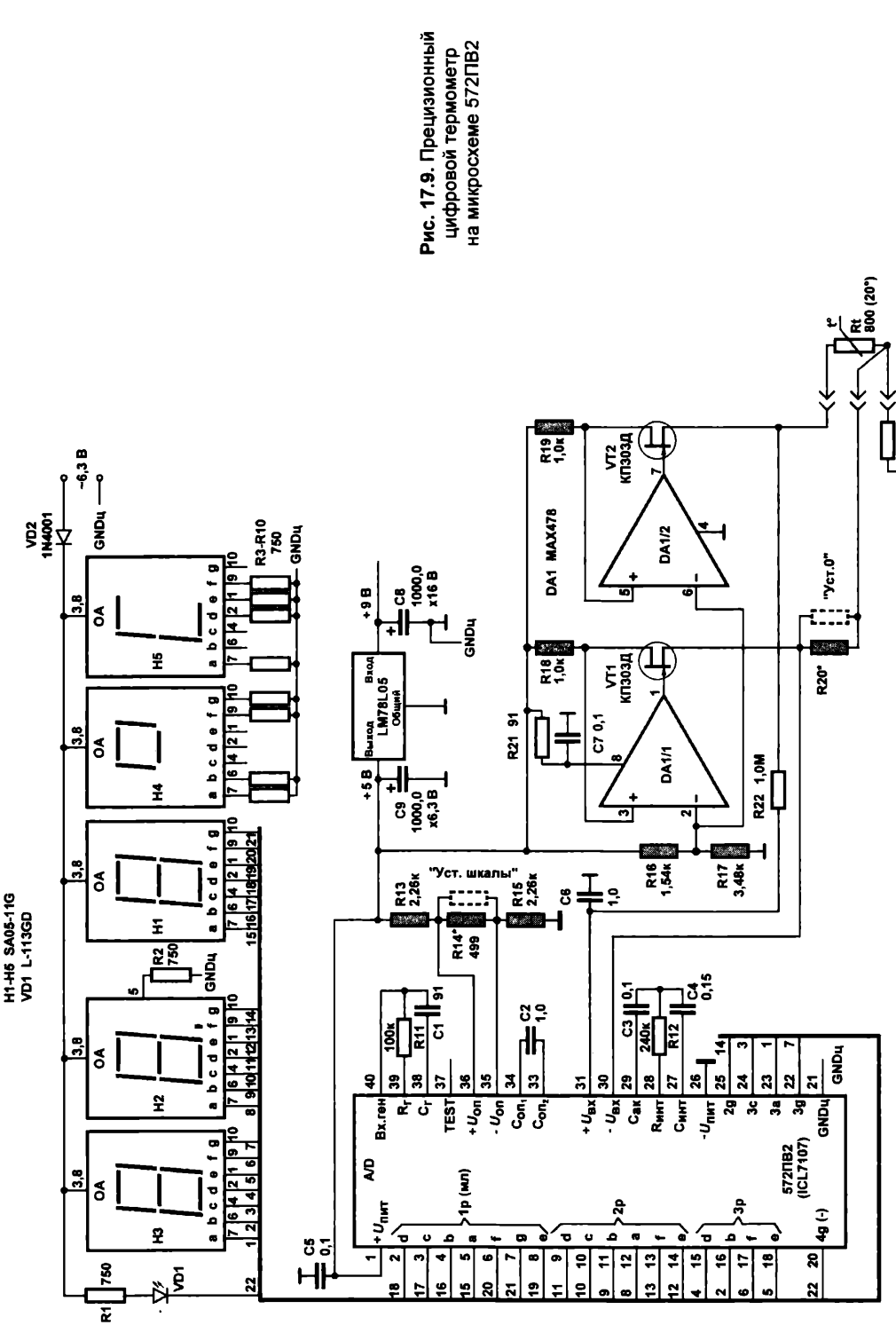


Рис. 17.9. Прецизионный цифровой термометр на микросхеме 572ПВ2

превысит 40 м. Однако в этой схеме и столь малые изменения нивелируются тем, что два одинаковых провода, соединяющие опорный резистор с датчиком и датчик с источником тока, оказываются включенными в разные плечи моста, потому их изменения взаимно компенсируются. Наведенные на этих проводах помехи ведут себя точно так же. А третий провод, соединяющий датчик с «землей», оказывается включенным в оба плеча сразу и создает чисто синфазную помеху, которая игнорируется преобразователем. Дополнительный резистор R23, включенный в этот провод, «подтягивает» напряжение разбаланса моста к середине напряжения питания (падение напряжения на R23 составляет около 1 В). При возможном изменении напряжения питания опорное напряжение и сигнал с выхода моста будут меняться пропорционально, поэтому ошибки не возникнет.

Цепочка R21–C7 есть дополнительный фильтр по питанию ОУ в источниках тока. Остальные компоненты схемы вызвать вопросов не должны. Все резисторы, выделенные темным, должны быть с точностью не хуже 1% — например, типа С2–29В. Номиналы их, естественно, необязательно должны быть именно такими, как указано на схеме, и могут меняться в очень широких пределах, но соотношения должны быть выдержаны точно. При ином сопротивлении датчика соотношения этих резисторов, а также сопротивления резисторов R20 и R23, придется пересчитать, при этом желательно приблизительно сохранить значения напряжений в схеме, особенно это касается близости к середине напряжения питания.

Индикаторная часть также не должна вызвать вопросов. Питание индикаторов в этой схеме обязательно должно осуществляться от отдельной обмотки трансформатора. Индикаторы зеленого свечения (с буквой G) можно заменить любыми другими, по вкусу. Так как мы четвертый разряд не используем, то не имеет смысла ставить целый индикатор для одного только знака минус, и его индикация производится с помощью одного плоского светодиода. Они бывают разных размеров, и чтобы прибор выглядел красиво, следует подогнать светящуюся полоску по ширине сегментов индикатора. В нашем случае светодиод L113 имеет размеры 5×2 мм, но сегменты заметно у́же, поэтому часть торцевой поверхности нужно аккуратно закрасить любой непрозрачной краской. Залить такой краской следует и боковые поверхности светодиода, иначе вместо минуса вы получите неопределенное светящееся пятно.

Если яркость минуса, запятой (вывод 5 индикатора H2) и индикаторов H4–H5, постоянно демонстрирующих знак «°С», будет отличаться от яркости основных разрядов, нужно подобрать резисторы R1–R10. Источник питания нужно рассчитывать на 250 мА по напряжению ~6,3 В (по напряжению +5 В потребление не достигает и 10 мА). Конечно, в целях экономии места, стоимости и потребления тока индикаторы H4–H5 можно исключить.

Датчик (рис. 17.10) можно изготовить следующим образом: берется трубка (лучше пластмассовая) длиной примерно 10 см и такого диаметра, чтобы все выводы реле, в том числе выводы обмотки с припаянными проводами, свободно помещались внутри. Места пайки на всякий случай следует изолировать термоусадочным кембриком. Затем нужно пропустить провода через трубку и обязательно в месте выхода из трубки также надеть на них отрезок кембрика, чтобы ограничить радиус

перегиба (позиция 5 на рис. 17.10). Потом залепить пластилином щели между корпусом реле и торцом трубки и залить ее внутренность эпоксидной смолой. Пластилин удаляется потом начисто с помощью бензина. Если датчик будет расположен снаружи помещения, его лучше покрыть атмосферостойким лаком или краской.

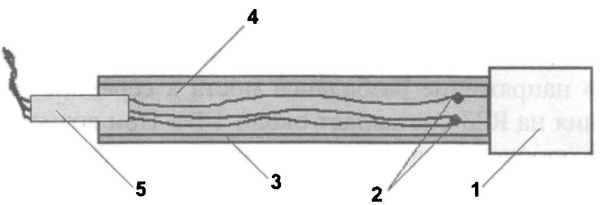


Рис. 17.10. Конструкция датчика на основе реле РЭС-49, РЭС-60, РЭС-79, РЭС-80 и аналогичных: 1 — реле; 2 — места пайки выводов; 3 — пластмассовая трубка; 4 — эпоксидная смола; 5 — кембрик

Схему следует собрать всю сразу (проверив отдельно, конечно, источник питания). Измерив величину сопротивления датчика при комнатной температуре, следует рассчитать необходимую величину резистора R20 (на схеме дана его величина, исходя из сопротивления датчика ровно 800 Ом при 20 °С). Затем на место калибровочных резисторов R14 и R20 нужно впаять резисторы большего номинала, а параллельно им — переменные резисторы с таким значением сопротивления, чтобы вместе они составляли номинал примерно на 5–10% больший расчетного. Наладку надо начинать с того, что проверить правильность разводки индикаторов. Для этого вывод «TEST» следует замкнуть с напряжением питания — индикаторы должны загореться все, показав значение «888».

Затем можно приступать к процедуре калибровки. Набейте термос толченым льдом (зимой лучше использовать для этой цели снег) пополам с водой — это будет первая калибровочная точка. Вторая может быть обеспечена просто теплой водой с температурой от 40 до 60 градусов, причем поддерживать точную температуру необязательно, только за ней нужно все время следить (хотя, разумеется, наличие термостата предпочтительнее). Поместив датчик в смесь льда и воды, с помощью резистора R20 устанавливают нулевые показания термометра. Затем датчик помещают в теплую воду вместе с образцовым термометром и с помощью резистора R14 устанавливают показания, соответствующие показаниям этого термометра. В обоих случаях размещать датчик нужно так, чтобы и он, и эталонный термометр не касались стенок, причем воду и смесь в термосе при этом следует обязательно перемешивать. Не забывайте, что каждый раз датчик следует выдерживать при соответствующей температуре не менее нескольких минут — до установления показаний.

Так как у нас при 0° мост находится в равновесии, то корректировки нуля и крутизны в целом независимы, и одной итерации достаточно, но на всякий случай следует несколько раз перенести датчик из нулевой температуры в теплую воду и обратно и при необходимости подкорректировать показания. Окончательно переменные резисторы заменяют на постоянные, которые подпаивают прямо к выводам основных (на схеме они показаны пунктиром). Эти дополнительные резисторы могут быть типа МЛТ — при условии, что основной резистор не слишком отличается от окон-

чательного номинала. Если все сделано аккуратно, то погрешность такого термометра не превысит приблизительно 0,2 °С во всем диапазоне от −50 до +50 °С.

При использовании иных типов датчиков, например полупроводниковых, отрицательное напряжение питания в 572ПВ2 может все же понадобиться. Лучший способ, безусловно, — сделать нормальный двухполярный источник ±5 В. На рис. 17.11 приведены различные варианты паллиативных решений. Первый вариант (рис. 17.11, *а*) представляет собой однополярный источник +10 В с искусственным расщеплением. Расщепитель организован на основе повторителя напряжения на ОУ с умощненным выходом на комплементарных транзисторах. Обратите внимание на схему включения стабилизатора LM78L09, которая позволяет получить напряжение несколько большее, чем номинальное. Излишне предупреждать, что питание индикаторов при такой схеме обязательно должно осуществляться от отдельного источника.

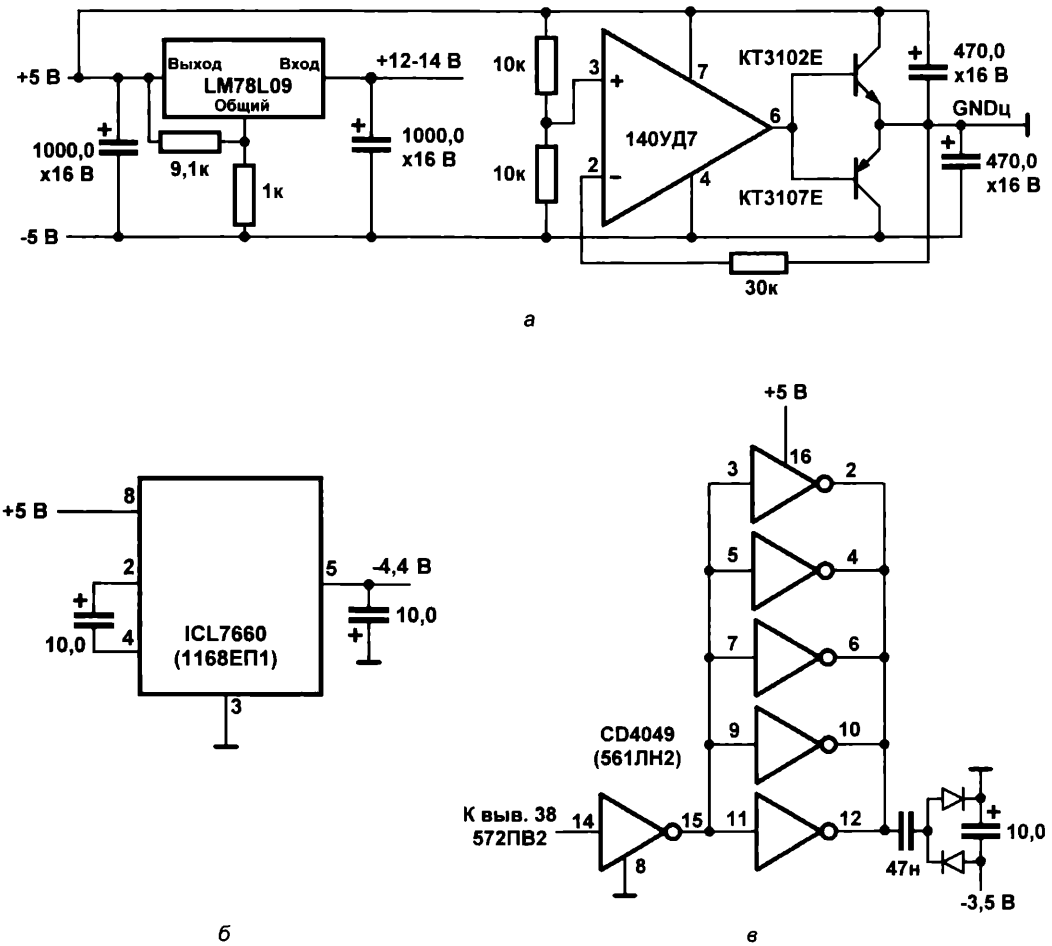
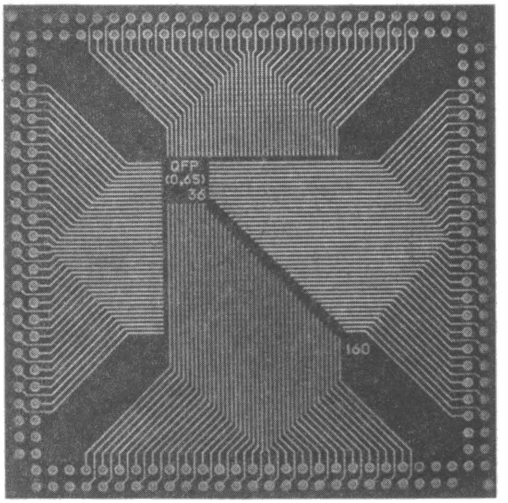


Рис. 17.11. Варианты организации отрицательного напряжения питания для 572ПВ2

Вторая схема (рис. 17.11, *б*) представляет собой инвертор-преобразователь положительного питания +5 В в отрицательное. Различных типов таких инверторов выпускается очень много, здесь выбран простейший нестабилизированный вариант. Преимущество преобразователя 1168ЕП1 (ICL7660, MAX1044) — в простоте включения, недостаток — высокое выходное сопротивление, так что при входном напряжении +5 В уже при потребляемом токе 20 мА отрицательное выходное напряжение снижается по абсолютной величине до 4,0 В (величина −4,4 В показана условно). Однако для нужд микросхемы 572ПВ2 этого вполне достаточно. Это вполне иллюстрирует рис. 17.11, *в*, на котором приведена схема инвертора напряжения, рекомендуемая самими разработчиками АЦП. Это на самом деле просто ухудшенный вариант того же инвертора, только работающий от тактовой частоты АЦП. Микросхема CD4049 может быть заменена на 561ПУ4 (CD4050) или на 561ЛН2 с соответствующей коррекцией разводки выводов. Подобные схемы могут особенно пригодиться для ПВ5 в случае батарейного питания — ведь обеспечить напряжение порядка 12–14 В, требующееся для нормального двухполярного источника ±5 В, в этом случае непросто.



ЧАСТЬ IV

Микроконтроллеры

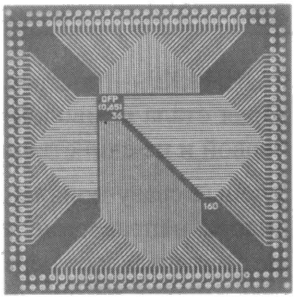
Глава 18. Начала микроэлектроники
Микропроцессоры, память и микроконтроллеры

Глава 19. Персональный компьютер вместо паяльника
О программировании МК на языке ассемблера

Глава 20. Основы Arduino
Контроллеры, среда и примеры программирования

Глава 21. Компоненты для Arduino
Как на Arduino делать устройства лучше фирменных

Глава 22. Применения Arduino
Избранные возможности платформы



ГЛАВА 18

Начала микроэлектроники

Микропроцессоры, память и микроконтроллеры

Людовик XIV поглотил все мелкие созвездия своего двора, затмив их своим ослепительным сиянием.

А. Дюма. «Три мушкетера»

Электронные устройства на дискретных элементах и тем более на микросхемах могут выполнять в автоматическом режиме довольно сложные функции. Устройства управления военной техникой в сороковые-шестидесятые годы XX века так и делали — для них строили специальные схемы на каждый раз, для каждой конкретной задачи, иногда очень «навороченные» и весьма остроумно придуманные. Эти схемы объединяли цифровые и аналоговые узлы, реализовывавшие различные функции, вплоть до решения в реальном времени сложнейших дифференциальных уравнений. Вы только представьте сложность задачи управления межконтинентальной баллистической ракетой, которая даже в те времена, когда не было ни спутников наведения, ни систем глобального позиционирования, обеспечивала точность попадания в радиусе нескольких десятков-сотен метров на расстоянии в тысячи километров!

Характерная черта таких устройств — они построены из одних и тех же основных элементов. Особенно это касается цифровой техники — со времен Клода Шеннона известно, что любая цифровая функция может быть реализована всего на нескольких базовых «кирпичиках», и мы видели в предыдущих главах, как на основе таких «кирпичиков» — логических элементов — строятся последовательно все более сложные устройства, вплоть до сумматоров и многофункциональных счетчиков, которые затем уже могут комбинироваться в схемы любой степени сложности. Возникает естественная мысль — а нельзя ли соорудить универсальное устройство, которое бы могло выполнять любые подобные функции, раз в какой-то глубинной основе своей они похожи?

К этой мысли человечество двигалось и с другой стороны, связанной с никогда не покидавшей ученых мечтой о построении искусственного разума. Через арифмометр Паскаля, аналитическую машину Бэббиджа, математическую логику Буля, теоретические построения Тьюринга и Шеннона, через первые электромеханические компьютеры Конрада Цузе, Эйкена и Атанасова, этот путь воплотился

в ЭНИАКе — построенной в 1946 году электронной вычислительной машине, которая стала символом начала компьютерной эпохи (хотя, добавим, была не самой первой и не единственной даже в те времена).

Ученые сразу поняли, каковы потенциальные возможности этого устройства: зародилось направление «искусственного интеллекта», стали обсуждаться проблемы автоматического перевода, шахматного компьютера, распознавания образов — увы, многие из них, несмотря на то, что мощность компьютеров возросла в миллионы раз, так и не решены до сих пор и вряд ли будут решены в ближайшее время[1].

Компьютер и есть то самое универсальное электронное устройство, которое может выполнить любую задачу: от наведения баллистической ракеты на цель до банального переключения режимов стиральной машины — надо только иметь соответствующую программу.

Заметки на полях

Для понимания того, как работают микропроцессорные системы, нужно очень твердо усвоить, что программирование процессора и составление логических схем есть в полном смысле слова один и тот же процесс, только выраженный на разных языках: либо в виде последовательности команд процессора, либо в виде схемы. Грубо говоря, при переходе на микроконтроллеры вы заменяете паяльник средствами программирования, причем программировать много проще, потому что гораздо легче поправить ошибку, а результат оказывается дешевле, надежнее и компактнее. Принцип эквивалентности можно проиллюстрировать таким примером: на процессоре 8086 операции с действительными числами выполнялись с помощью подпрограмм, но выполнение программы всегда медленнее, чем работа «железок». Поэтому к нему сначала добавили арифметический сопроцессор (8087), а потом (начиная с 486-х) и вовсе интегрировали блок обработки чисел «с плавающей точкой» внутрь процессора. В результате программы упростились, а процессор усложнился, но с точки зрения пользователя ничего (кроме ускорения работы) не произошло. Но потом процессоры стали быстрее, а количество необходимых функций возросло, поэтому, начиная с какого-то момента, их опять стали реализовывать в виде подпрограмм, только уже «зашитых» прямо в процессор. И опять с точки зрения пользователя ничего не произошло — просто процессор стал «умнее».

Однако принцип эквивалентности «железа» и программ, благодаря работам Шеннона ставший понятным ученым и инженерам еще задолго до эпохи всеобщей компьютеризации, дошел до практики далеко не сразу — «железо» резко отставало от нужд практики. Первые ЭВМ были огромными, потребляли энергии, как небольшой завод, требовали непрерывного обслуживания (плановое ежесуточное время работы первых советских ЭВМ составляло 16 часов, а остальное занимали ремонт и профилактика). Кому в те времена могла прийти мысль даже о том, чтобы дать компьютер каждому в персональное пользование, не то что пристроить его к управлению стиральной машиной, правда? Революция произошла лишь с изобре-

[1] В 1950 году Алан Тьюринг опубликовал работу «Вычислительные машины и интеллект», в которой предположил, что «думающий» компьютер, который нельзя было бы отличить по поведению от человека, должен иметь объем памяти примерно в 10^{10} битов — чуть больше гигабайта. В компьютере, на котором набирается этот текст, памяти в восемь раз больше...

тением микропроцессора в фирме Intel в 1971 году. С этого момента инженерам-электронщикам пришлось учить программирование.

Первый микропроцессор

Первоначально корпорация Intel не помышляла ни о каких процессорах и занималась разработкой и продажами микросхем памяти, на которые тогда как раз начиналось увеличение спроса. В 1969 году в Intel приехали несколько человек из Busicom — молодой японской компании, занимающейся производством калькуляторов. Им требовался набор из 12 интегральных схем в качестве основного элемента нового дешевого настольного калькулятора.

Проект был разработан Масатоши Шима (Masatoshi Shima), который и представлял японскую сторону. Тед Хофф (Marcian E. «Ted» Hoff, p. 1937), руководитель отдела, занимавшегося вопросами применений для продукции Intel, ознакомившись с проектом, понял, что вместо того чтобы создать калькулятор с некоторыми возможностями программирования, можно поступить наоборот — создать компьютер, программируемый для работы в качестве калькулятора. Развивая идею, в течение осени 1969 года Хофф определился с архитектурой будущего микропроцессора. Весной в отдел Хоффа пришел (все из той же уже известной нам Fairchild) новый сотрудник Федерико Фэггин (Federico Faggin), который и придумал название для всей системы: *семейство 4000*. Семейство состояло из четырех 16-выводных микросхем: 4001 содержала ROM на 2 килобайта, 4002 содержала RAM с 4-битным выходным портом для загрузки программ, 4003 представляла собой 10-битный расширитель ввода/вывода с последовательным вводом и параллельным выводом для связи с клавиатурой, индикатором и другими внешними устройствами, наконец, 4004 (рис. 18.1) была 4-битным ЦПУ (центральным процессорным устройством). Оно содержало 2300 транзисторов и работало с максимальной тактовой частотой 740 кГц[1]. 15 ноября 1971 года было объявлено о создании первого микропроцессора. Busicom приобрела разработку, заплатив Intel 60 тыс. долларов. Но в Intel решили возвратить Busicom эти деньги, чтобы вернуть себе права на микропроцессор.

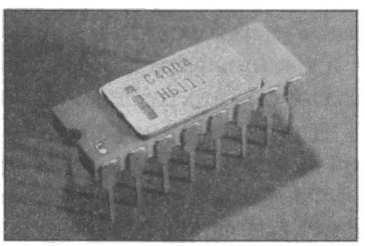


Рис. 18.1. Микропроцессор Intel 4004

i4004 обладал вычислительной мощностью, сравнимой с первым электронным компьютером ENIAC, — по скорости выполнения операций превосходил примерно на порядок, но, разумеется, отставал в разрядности (ENIAC имел 20-разрядные команды). Свое первое практическое применение 4004-й нашел в таких системах, как устройства

[1] Согласно документации на ЦПУ 4004, размещенной в музее Intel, один период тактовой частоты (clock period) может быть в пределах от 1,35 до 2 мкс (из первой цифры и получаем максимально возможную частоту около 740 кГц). При этом один машинный цикл занимает 8 периодов тактовой частоты, т. е. 10,8 мкс. Из последней цифры, вынесенной в заголовок документа, видимо, и взялась часто приводящаяся, но ошибочная цифра тактовой частоты 108 кГц. Очевидно, что любые другие цифры (например, «Википедия» дает почему-то разброс 92,6–200 кГц) также являются плодом фантазии или невнимательности авторов.

управления дорожными светофорами и анализаторы крови. Говорят, что он был использован в бортовой аппаратуре межпланетного зонда Pioneer-10, который поставил рекорд долгожительства среди подобных аппаратов (он был запущен NASA в 1972 году, а к 1 сентября 2001 года удалился от Земли на 11,78 млрд км и все еще работал), но по последним данным, это всего лишь легенда.

Как работает микропроцессор?

Для того чтобы понять, как работает микропроцессор, зададим себе вопрос — а как он должен работать? Есть теория (в основном созданная постфактум — после того, как первые ЭВМ были уже построены и функционировали), которая указывает, как именно строить алгоритмы, и что процессор в соответствии с ними должен делать. Мы, естественно, углубляться в это не будем, просто констатируем, что любой *алгоритм* есть последовательность неких действий, записанных в виде набора последовательно выполняемых команд (инструкций, операторов). При этом среди таких команд могут встречаться команды перехода, которые в некоторых случаях нарушают исходную последовательность выполнения операторов строго друг за другом. Среди прочих должны быть также команды ввода и вывода данных (программа должна как-то общаться с внешним миром?), а также команды выполнения арифметических и логических операций.

Команды должны где-то храниться, поэтому неотъемлемой частью всей системы должно быть устройство памяти программ. Где-то надо складывать и данные, как исходные, так и результаты работы программы, поэтому должно иметься и устройство памяти данных. Так как команды и данные, в конечном счете, все равно есть числа, то память может быть общая, только надо уметь отличать, где именно у нас команды, а где — данные. Это есть один из принципов фон Неймана, хотя и в микроконтроллерах, о которых мы будем говорить в дальнейшем, традиционно используют не фон-неймановскую, а так называемую *гарвардскую* архитектуру, когда памяти данных и программ разделены (это разделение, впрочем, может в определенных пределах нарушаться). Процессор, построенный по фон Нейману, более универсален — например, он позволяет без особых проблем наращивать память, строить ее иерархически и более эффективно ее перераспределять прямо по ходу работы. Так, в системе Windows всегда предполагается, что компьютер имеет практически неограниченный объем памяти (измеряемый в терабайтах), а если ее реально не хватает, к делу подключается своп-файл (так называемый *файл подкачки*) на жестком диске. В то же время микроконтроллерам подобная гибкость не особенно требуется — на их основе, как правило, строятся узлы, выполняющие узкую задачу и работающие по конкретной программе, так что нужную конфигурацию системы ничего не стоит предусмотреть заранее.

МП и МК

Кстати, а почему мы все время говорим то *микропроцессоры* (МП), то *микроконтроллеры* (МК)? Микроконтроллер отличается от микропроцессора тем, что он предназначен для управления другими устройствами, и поэтому имеет встроенную развитую систему ввода/вывода, но, как правило, относительно более слабое АЛУ. Микроконтроллерам очень хорошо подходит английский термин «computer-on-chip», однокристальный компьютер. В самом деле, для построения простейшего вычислительного

устройства, которое могло бы выполнять что-то полезное, обычный микропроцессор, от i4004 до Pentium и Core, приходится дополнять памятью, ПЗУ с записанной BIOS, устройствами ввода/вывода, контроллером прерываний, тактовым генератором с таймерами и т. п. — всем тем, что сейчас стало объединяться в так называемые *чипсеты*. «Голый» МП способен только на одно — правильно включиться, ему даже программу загрузки неоткуда взять.

В то же время для МК микропроцессор — это только ядро, даже не самая большая часть кристалла. Для построения законченной системы на типовом МК не требуется вообще ничего, кроме источника питания и периферийных исполняющих устройств, которые позволяли бы человеку определить, что система работает. Обычный МК может без дополнительных компонентов общаться с другими МК, внешней памятью, специальными микросхемами (вроде часов реального времени или флэш-памяти), управлять небольшими (а иногда — и большими) матричными панелями, к нему можно напрямую подключать датчики физических величин (в том числе — чисто аналоговые, АЦП тоже часто входят в МК), кнопки, клавиатуры, светодиоды и индикаторы, короче — в микроконтроллерах сделано все, чтобы приходилось как можно меньше паять и задумываться над подбором элементов. За это приходится расплачиваться пониженным быстродействием (которое, впрочем, не так-то уж и важно в типовых задачах для МК) и некоторым ограничением в отдельных функциях — по сравнению с универсальными, но в сотни раз более дорогими и громоздкими системами на «настоящих» МП. Вы можете мне не поверить, но рекордные по производительности процессоры для персональных компьютеров (ПК), о которых мы столько слышим, занимают в общем количестве выпускаемых процессоров лишь 5–6%, — остальные составляют микроконтроллеры различного назначения. В последнее время в таких устройствах, как планшеты и особенно смартфоны, грань между МК и МП все больше размывается: МП для мобильных устройств интегрируют в себе многие функции, характерные для МК.

В соответствии с изложенным, основной цикл работы процессора должен быть таким: выборка очередной команды (из памяти), если необходимо — выборка исходных данных для нее, выполнение команды, размещение результатов в памяти (опять же если это необходимо). Вся работа в этом цикле должна происходить автоматически по командам некоторого устройства управления, содержащего тактовый генератор — системные часы, по которым все синхронизируется. Кроме того, где-то это все должно происходить: складирование данных, кода команды, выполнение действий и т. п., так что процессор должен содержать некий набор рабочих регистров (по сути — небольшую по объему сверхбыструю память), определенным образом связанных как между собой, так и с устройством управления и АЛУ, которое неизбежно должно присутствовать.

Решающую роль в работе процессора играет *счетчик команд*. Он автоматически устанавливается на нуль в начале работы, что соответствует первой команде, и автоматически же инкрементируется (т. е. увеличивается на единицу) с каждой выполненной командой. Если по ходу дела порядок следования команд нарушается — например, встречается команда перехода (ветвления), то в счетчик загружается соответствующий адрес команды — ее номер от начала программы. Если это не просто ветвление, а выполнение подпрограммы, которое предполагает в дальнейшем возврат к основной последовательности команд (к следующей команде после вызова подпрограммы), то перед переходом к выполнению подпрограммы текущее значение счетчика команд сохраняется в специально отведенной для этой цели области памяти — *стеке*. По команде окончания подпрограммы сохраненный адрес из-

влекается из стека, и выполнение основной программы продолжается. К счастью, нам самим не придется иметь дело со счетчиком команд, потому что все указания на этот счет содержатся в командах, и процессор все делает автоматически.

Блок-схема простейшего МК, содержащего процессорное ядро и минимум компонентов для «общения» с внешней средой, показана на рис. 18.2. Здесь мы включили в состав системы память программ, которая у ПК-процессоров находится отдельно на жестком диске (если не считать относительно небольшого объема ПЗУ, содержащего так называемую BIOS, т. е. базовые процедуры для запуска и обмена с внешней средой) — сами знаете, какой объем программ бывает в персональных компьютерах. В большинстве современных микроконтроллеров постоянное запоминающее устройство (ПЗУ) для программ входит в состав чипа и обычно составляет от 1–2 до 8–32 Кбайт, хотя есть модели и с 256 килобайтами встроенной памяти. 2–8 Кбайт для подавляющего большинства применений вполне достаточно — при условии, что вы создаете программы прямо в командах контроллера, на языке ассемблера. Применение языка высокого уровня (обычно одного из вариантов язы-

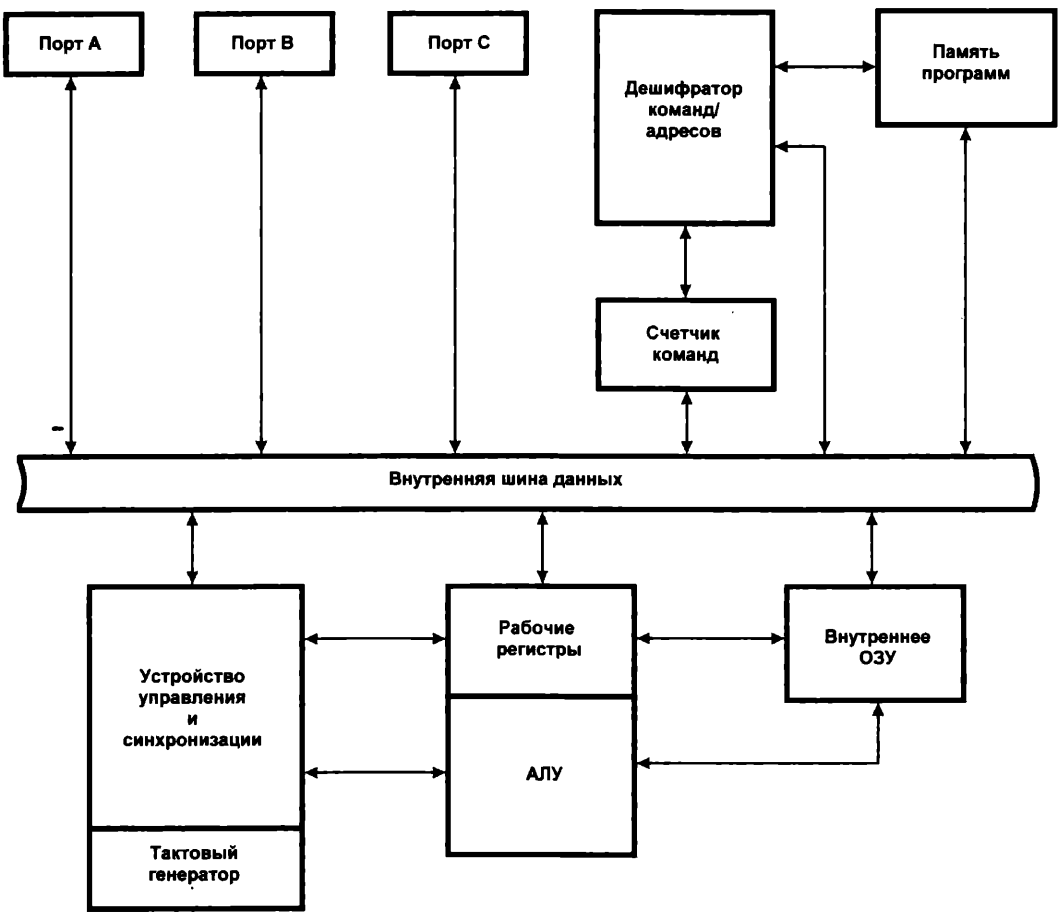


Рис. 18.2. Блок-схема простейшего микроконтроллера

ка С), что становится все более популярным из-за удобства работы с ним, как мы увидим, существенно повышает требования к объему памяти программ.

Встроенное оперативное запоминающее устройство (ОЗУ) для хранения данных в том или ином объеме также имеется во всех современных микроконтроллерах, типичный размер такого ОЗУ: от 128–256 байтов до 1–4 Кбайт. В большинстве универсальных контроллеров есть и некоторое количество встроенной энергонезависимой памяти для хранения констант — обычно столько же, сколько и ОЗУ данных. Но к памяти мы еще вернемся в этой главе, а пока продолжим про процессоры.

Подробности

В первых моделях микропроцессоров (включая и интеловские процессоры для ПК — от 8086 до 80386) процессор выполнял команды строго последовательно: загрузить команду, определить, что ей нужны операнды, загрузить эти операнды (по адресу регистров, которые их должны содержать, — адреса эти, как правило, хранятся сразу после собственно кода команды или определены заранее), потом проделать нужные действия, складировать результаты... До нашего времени дошла архитектура суперпопулярных еще недавно микроконтроллеров 8051, выпускающихся и по сей день различными фирмами (Atmel, Philips), которые выполняли одну команду аж за 12 тактов (в некоторых современных аналогах, впрочем, это число меньше). Для ускорения работы стали делить такты на части (например, срабатывать по переднему и заднему фронтам), но действительный прорыв произошел с внедрением *конвейера*. Со времен Генри Форда известно, что производительность конвейера зависит только от времени выполнения самой длинной операции, — если поделить команды на этапы и выполнять их одновременно разными аппаратными узлами, то можно добиться существенного ускорения (хотя и не во всех случаях). В рассматриваемых далее микроконтроллерах Atmel AVR конвейер двухступенчатый: когда очередная команда загружается и декодируется, предыдущая уже выполняется и пишет результаты. В AVR это позволило выполнять большинство команд за один такт (кроме команд ветвления программы и некоторых других).

Главное устройство в МП, которое связывает все узлы в единую систему, — *внутренняя шина данных*. По ней все устройства обмениваются сигналами. Например, если МП требуется обратиться к внешней дополнительной памяти, то при исполнении соответствующей команды на шину данных выставляется нужный адрес, от устройства управления поступает через нее же запрос на обращение к нужным портам ввода/вывода. Если порты готовы, адрес поступает на выходы портов (т. е. на соответствующие выводы контроллера), затем по готовности принимающий порт выставляет на шину принятые из внешней памяти данные, которые загружаются в нужный регистр, после чего шина данных свободна. Для того чтобы все устройства не мешали друг другу, все это строго синхронизированно, при этом каждое устройство имеет, во-первых, собственный адрес, во-вторых, может находиться в трех состояниях: работать на ввод, на вывод или находиться в третьем состоянии, не мешая другим работать.

Под *разрядностью МП* обычно понимают разрядность чисел, с которыми работает АЛУ, соответственно, такую же разрядность имеют и рабочие регистры. Например, все ПК-процессоры от i386 до последних инкарнаций Pentium были 32-разрядными, большинство последних моделей от Intel и AMD относятся уже к 64-разрядным.

Большинство микроконтроллеров общего назначения — 8-разрядные, но все большую популярность приобретают 32-разрядные, обладающие принципиально большими возможностями при практически той же цене. Интересно, что развитие промежуточной ветви 16-разрядных контроллеров практически остановилось ввиду нецелесообразности.

Заметки на полях

Обычно тактовая частота универсальных МК невелика (хотя инженеру 1980-х, когда ПК работали на частотах не выше 6 МГц, она показалась бы огромной) — порядка 8–16 МГц, иногда до 20 МГц или несколько более. И это всех устраивает — дело в том, что обычные МК и не предназначены для разработки быстродействующих схем. Если требуется быстродействие, то используется другой класс интегральных схем — ПЛИС, «программируемые логические интегральные схемы» (английское название самой популярной сейчас их разновидности — FPGA, field-programmable gate array). Простейшая ПЛИС представляет собой набор никак не связанных между собой логических элементов (наиболее сложные из них могут включать в себя и некоторые законченные узлы, вроде триггеров и генераторов), которые в процессе программирования такого чипа соединяются в нужную схему. Комбинационная логика работает гораздо быстрее тактируемых контроллеров, и для построения сложных логических схем в настоящее время применяют только ПЛИС, от использования дискретных элементов («рассыпухи») в массовых масштабах уже давно отказались. Еще одно преимущество ПЛИС — статическое потребление энергии для некоторых серий составляет единицы микроватт, в отличие от МК, которые во включенном состоянии потребляют достаточно много (если не находятся в режиме энергосбережения). В совокупности с более универсальными и значительно более простыми в обращении, но менее быстрыми и экономичными микроконтроллерами, ПЛИС составляют основу большинства массовых электронных изделий, которые вы видите на прилавках. В этой книге мы, конечно, рассматривать ПЛИС не будем — в любительской практике, в основном из-за дороговизны соответствующего инструментария и высокого порога его освоения, они не используются, а для конструирования одиночных экземпляров приборов даже для профессиональных применений их использовать нецелесообразно.

Если подробности внутреннего функционирования МП нас волнуют не очень (центральный узел — АЛУ — мы уже «изобретали» в *главе 15*, и этого достаточно, чтобы понимать, что именно происходит внутри процессорного ядра), то обмен с внешней средой нас как раз интересует во всех деталях. Для этого служат *порты ввода/вывода* (I/O-port, от **I**nput/**O**utput). В этом термине имеется некоторая неопределенность, т. к. те, кто программировал для ПК на ассемблере, помнят, что в ПК портами ввода/вывода (ПВВ) назывались регистры для управления всеми устройствами, кроме непосредственно процессорного ядра. В микроконтроллерах то же самое называют *регистрами ввода/вывода* (РВВ) — это регистры для доступа ко встроенным компонентам контроллера, внешним по отношению к вычислительному ядру. А это все узлы, которыми непосредственно управляет пользователь: от таймеров и последовательных портов до регистра флагов и управления прерываниями. Кроме ОЗУ, доступ к которому обеспечивается специальными командами, все остальное в контроллере управляется через РВВ, и путать с портами ввода/вывода их не следует.

ПВВ в МК служат для обмена с «окружающей средой» (управляются они, естественно, тоже внутренними регистрами ввода/вывода). На схеме рис. 18.2 показано 3 ПВВ: A, B и C, в реальных МК их может быть и больше, и меньше. Еще важнее

число выводов этих портов, которое чаще всего совпадает с разрядностью процессора (но не всегда, как это было у 8086, который имел внутреннюю 16-разрядную структуру, а внешне выглядел 8-разрядным). Если мы заставим 8-разрядные порты «общаться», например, с внешней памятью, то на двух из них можно выставить 16-разрядный адрес, а на оставшемся — принимать данные. А как быть, если портов два или вообще один? (К примеру, в микроконтроллере ATtiny2313 портов формально два, но один усеченный, так что общее число линий составляет 15). Для того чтобы даже в такой ситуации это было возможно, все внешние порты в МП всегда двунаправленные. Скажем, если портов два, то можно сначала выставить адрес, а затем переключить порты на вход и принимать данные. Естественно, для этого порты должны позволять работу на общую шину — т. е. либо иметь третье состояние, либо выход с общим коллектором для объединения в «монтажное ИЛИ».

Варианты для обоих случаев организации выходной линии порта показаны на рис. 18.3, где приведены упрощенные схемы выходных линий микроконтроллеров семейства 8048 — когда-то широко использовавшегося предшественника популярного МК 8051 (например, 8048 был выбран в качестве контроллера клавиатуры в IBM PC). В современных МК построение портов несколько сложнее (в частности, вместо резистора там полевой транзистор), но для уяснения принципов работы это несущественно.

По первому варианту (рис. 18.3, *а*) в МК 8048 построены порты 1 и 2. Когда в порт производится запись, то логический уровень поступает с прямого выхода защелки на статическом D-триггере на вход схемы «И», а с инверсного — на затвор транзистора VT2. Если этот уровень равен логическому нулю, то транзистор VT1 заперт, а VT2 открыт, на выходе также логический ноль. Если уровень равен логической единице, то на время действия импульса «Запись» транзистор VT1 открывается, а транзистор VT2 запирается (они одинаковой полярности). Если на выходе присутствует емкость (а она всегда имеется в виде распределенной емкости проводников и емкости входов других компонентов), то через открытый VT1 протекает достаточно большой ток заряда этой емкости, позволяющий сформировать хороший фронт перехода из 0 в 1. Как только импульс «Запись» заканчивается, оба транзистора отключаются, и логическая единица на выходе поддерживается резистором R1. Выходное сопротивление открытого транзистора VT1 примерно 5 кОм, а резистора — 50 кОм. Любое другое устройство, подключенное к этой шине, при работе на выход может лишь либо поддержать логическую единицу, включив свой подобный резистор параллельно R1, либо занять линию своим логическим нулем, — это, как видите, и есть схема «монтажное ИЛИ». При работе на вход состояние линии просто считывается во время действия импульса «Запись» со входного буфера (элемент **В** на рис. 18.3, *а*).

Второй вариант (рис. 18.3, *б*), по которому устроен порт 0, есть обычный выходной каскад КМОП с третьим состоянием, т. е. такой порт может работать на выход, только полностью занимая линию, остальные подключенные к линии устройства при этом должны смиренно внимать монополисту, воспринимая сигналы. Это обычно не создает особых трудностей и схемотехнически даже предпочтительно ввиду симметрии выходных сигналов и высокого сопротивления для входных.

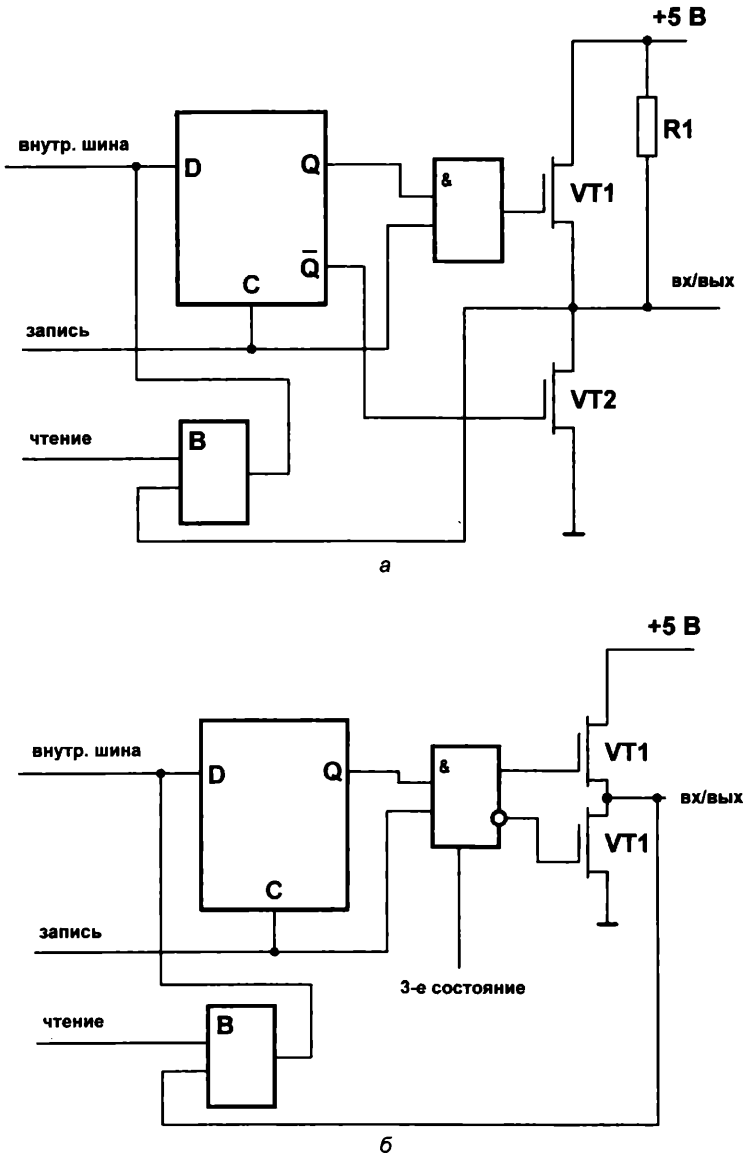


Рис. 18.3. Упрощенные схемы портов ввода/вывода МК 8048: *а* — портов 1 и 2; *б* — порта 0

Единственная сложность возникает при сопряжении такого порта с линией, работающей по первому варианту, т. к. при логической единице на выходе могут возникнуть электрические конфликты, если кто-то попытается выдать в линию логический ноль (ток от источника пойдет через два распахнутых транзистора).

Для обеспечения работы трехстабильного порта по схеме «монтажное И» (см. рис. 15.4) применяют хитрый прием: всю линию «подтягивают» к напряжению питания с помощью внешнего резистора (во многих МК существует встроенный отключаемый резистор, установленный аналогично R1 в схеме на рис. 18.3, *а*),

и нормальное состояние всех участвующих трехстабильных портов — работа на вход в третьем состоянии. В этом режиме на линии всегда будет логическая единица. На выход же линию переключают только, когда надо выдать логический ноль. В этом случае, даже при одновременной активности нескольких портов, конфликтов не возникнет.

Лечение амнезии

В 1965 году в Иллинойсском университете был запущен один из самых передовых компьютеров по тому времени — ILLIAC-IV. Он стал первым компьютером, в котором использовалась быстрая память на микросхемах, — каждый чип (производства Fairchild Semiconductor) имел емкость 256 битов, а всего было набрано 1 Мбайт. Стоимость этой памяти составила ощутимую часть от всей стоимости устройства, обошедшегося заказчику — NASA — в $31 млн. Через 10 лет один из первых персональных компьютеров Altair 8800 (1975 год), продававшийся в виде набора «сделай сам», при стоимости порядка $500 имел всего 256 байтов (именно байтов, а не килобайтов) памяти. В том же году для распространения языка Basic for Altair Биллом Гейтсом и Полом Алленом была создана фирма, получившая первоначальное название Micro-Soft. Одна из самых серьезных проблем, которую им пришлось решать, — нехватка памяти, потому что созданный ими интерпретатор Basic требовал аж 4 Кбайт!

Проблема объемов памяти и ее дороговизна преследовала разработчиков достаточно долго — еще в конце 1990-х стоимость памяти для ПК можно было смело прикидывать из расчета 1 доллар/Мбайт, что при требовавшихся уже тогда для комфортной работы объемах ОЗУ порядка 128–256 Мбайт могло составлять значительную часть стоимости устройства. Сейчас 4 гигабайта памяти в настольном ПК или ноутбуке уже стали фактическим стандартом. Это привело, в частности, к кардинальным изменениям в самом подходе к программированию — если еще при программировании под DOS о компактности программ и экономии памяти в процессе работы нужно было специально заботиться, то теперь это практически не требуется.

Но в программировании для микроконтроллеров это все еще не так. Хотя гейтсовский интерпретатор Basic влезет в большинство современных однокристальных МК, но экономная программа легче отлаживается (а значит, содержит меньше ошибок) и быстрее выполняется. Три-четыре потерянных на вызове процедуры такта могут стать причиной какой-нибудь трудновылавливаемой ошибки времени выполнения — например, если за это время произойдет вызов прерывания. Поэтому память в МК стоит экономить, даже если вы располагаете заведомо достаточным ее объемом. Этот призыв, конечно, пропадает впустую, когда мы сталкиваемся с программированием на языках высокого уровня, где от нас зависит гораздо меньше, чем от разработчиков компиляторов.

Далее мы рассмотрим основные разновидности памяти, используемые как в составе микроконтроллеров, так и во внешних узлах. И начнем с того, что попробуем сами сконструировать устройство долговременной памяти — ПЗУ (постоянное запоми-

нающее устройство). Как мы увидим, любая память в принципе есть не что иное, как преобразователь кодов.

Изобретаем простейшую ROM

Всем известно сокращение ROM — **R**ead-**O**nly **M**emory — английское название постоянного запоминающего устройства, ПЗУ. На самом деле это название (*память только для чтения*) не очень точно характеризует суть дела, отечественное название есть более корректный термин, самое же правильное называть такую память *энергонезависимой*. Ведь ПЗУ отличается от других типов памяти не тем, что его можно только читать, а записывать нельзя (практически все современные устройства ROM имеют возможность записи), а тем, что информация в нем не пропадает при выключении питания.

Тем не менее, первыми разновидностями ПЗУ, изобретенными еще в 1956 году, были именно нестираемые кристаллы, которые носят наименование OTP ROM — **O**ne-**T**ime **P**rogrammable **ROM**, однократно программируемое ПЗУ. До недавнего времени на них делали память программ МК для удешевления серийных устройств — вы отлаживаете программу на перезаписываемой памяти, а в серию пускаете приборы с «прожигаемой» OTP ROM. И лишь в последние годы «прожигаемая» память стала постепенно вытесняться более удобной flash-памятью, поскольку последняя подешевела настолько, что смысл в использовании одноразовых кристаллов пропал.

Мы сконструируем подобие «прожигаемого» ПЗУ с помощью диодов. Простейший вариант такого ПЗУ показан на рис. 18.4. В данном случае он представляет собой не что иное, как преобразователь из десятичного кода в семисегментный. Если на входе поставить дешифратор типа 561ИД1, переводящий двоичный код в десятичный, то мы получим аналог микросхемы 561ИД5.

Чтобы понять, как это работает, представьте себе, что первоначально на всех пересечениях между строками и столбцами диоды присутствовали — это аналог незаполненной памяти, в которой записаны все единицы. Затем мы взяли и каким-то образом (например, подачей высокого напряжения) разрушили те диоды, которые нам не нужны, в результате чего получили нужную конфигурацию.

Эта схема не содержит активных элементов, и потому возможности ее ограниченны, — например, выходы устройства, подающего активный высокий уровень по входным линиям, должны «тащить» всю нагрузку по зажиганию сегментов. Обычная микросхема ПЗУ построена на транзисторных ячейках, вследствие чего безо всяких хитростей принимает и выдает обычные логические уровни. К тому же она включает в себя и дешифрирующую логику, поэтому на вход подается двоичный, а не десятичный код.

Постойте, а причем тут ПЗУ вообще? Дело в том, что входной код здесь можно рассматривать как адрес ячейки, а выходной — как ее содержимое. И любое ПЗУ можно представить, как универсальный преобразователь кодов. Причем удобство состоит в том, что изначально в ПЗУ не записано ничего (одни нули или единицы), и мы может реализовать на нем любую логическую функцию — все зависит только

от его емкости. В том числе, такую простую, как преобразователь кодов десятичный-семисегментный, или же такую сложную, как операционная система Windows. Последнее мы каждый раз и делаем, когда устанавливаем Windows на компьютер, причем в качестве ПЗУ выступает жесткий диск. Из этого примера отчетливо видно, что каким бы сложным ни был алгоритм, он все равно в конечном итоге сводится к совокупности однозначных логических уравнений, которые можно реализовать как через ПЗУ с записанной программой, так и с помощью цифрового устройства любого другого типа.

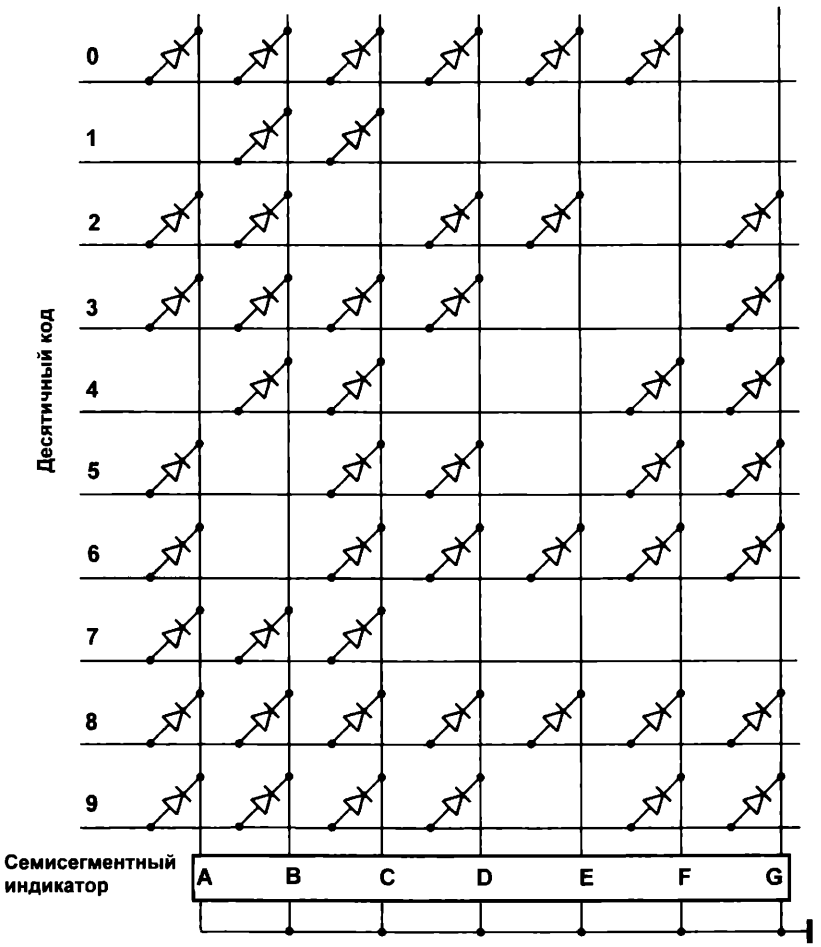


Рис. 18.4. Простейшее ПЗУ — преобразователь кода

Общее устройство памяти

Общее устройство фрагмента памяти любого типа показано на рис. 18.5. Из рисунка видно, что память всегда представляет собой матричную структуру. В нашем случае матрица памяти имеет 8×8 = 64 однобитных ячейки. Рис. 18.5 демонстрирует, как производится вывод и загрузка информации в память с помощью мульти-

плексоров/демультиплексоров (вроде 561КП2, см. *главу 15*). Код, поступающий на мультиплексор слева (x_3–x_5) подключает к строке с номером, соответствующим этому коду, активирующий уровень напряжения (это может быть логическая единица, как показано на рисунке, или ноль, неважно). Код на верхнем демультиплексоре (x_0–x_2) выбирает столбец, в результате к выходу этого демультиплексора подключается ячейка, стоящая на пересечении выбранных строки и столбца.

Легко заметить, что сама по себе организация матрицы при таком однобитном доступе для внешнего мира не имеет значения. Если она будет построена как 4×16, или 32×2, или даже вытянута в одну линеечку 64×1 — в любом случае код доступа (он называется *адресным* кодом) будет 6-разрядным, а выход один-единственный. Поэтому всем таким ЗУ приписывается организация N×1 битов, где N — общее число битов. Для того чтобы получить байтную организацию, надо просто взять 8 таких микросхем и подать адресный код на них параллельно, — тогда на выходах получим параллельный восьмибитный код, соответствующий байту. Общая емкость такой памяти составит 64×8 = 512 битов или 64 байта. У нас получается хорошая модель типового модуля памяти, вроде тех, что используются в компьютерном ОЗУ. Большинство выпускаемых интегральных ЗУ также сложены из таких отдельных однобитных модулей (только в наше время уже значительно большей емкости) и имеют 8, 16, 32 или большее количество параллельных выходов, но бывают кристаллы и с последовательным (побитным) доступом.

В качестве примера можно привести, скажем, ПЗУ с организацией 64К×16 типа AT27C1024 фирмы Atmel (рис. 18.6). Это однократно программируемое КМОП

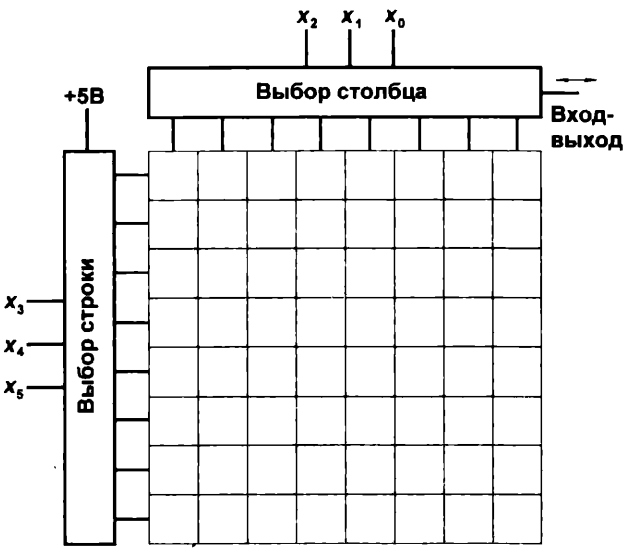


Рис. 18.5. Схематическое устройство ЗУ с однобитным последовательным выходом

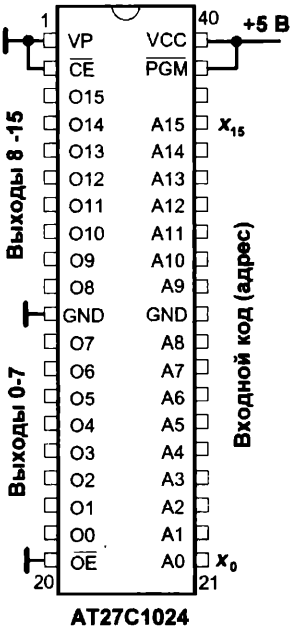


Рис. 18.6. Разводка выводов AT27C1024

ПЗУ с напряжением питания 5 В и емкостью 1024 Мбит, что составляет 128 Кбайт или 64 К двухбайтных слов. Следует отметить, что в области микросхем памяти сложилась хорошая традиция, когда все они, независимо от производителя и даже технологии, совпадают по выводам, разводка которых зависит только от организации матрицы (даже, как правило, не от объема!) и, соответственно, от применяемого корпуса (в рассматриваемом случае — DIP-40). Для разных типов (RAM, ROM, EEPROM и т. д.) разводка различается в части выводов, управляющих процессом программирования, но можно спокойно заменять одну микросхему на другую (с той же организацией и, соответственно, в таком же корпусе) без переделки платы.

RAM

Традиционное название энергозависимых типов памяти, как и в случае ROM, следует признать довольно неудачным. RAM значит **R**andom **A**ccess **M**emory, т. е. *память с произвольным доступом*, или, по-русски — ЗУПВ, *запоминающее устройство с произвольной выборкой*. Главным же признаком класса является не «произвольная выборка», а то, что при выключении питания память стирается. EEPROM (о которой далее), к примеру, тоже допускает произвольную выборку и при записи, и при чтении. Но так сложилось исторически, и не нам разрушать традиции.

Подавляющее большинство производимых микросхем ЗУПВ относится к динамическому типу. В них информация хранится в виде заряда на конденсаторе, который имеет привычку быстро утекать, и потому такая память требует периодической регенерации (раз в несколько миллисекунд). Зато она дешева (каждая ячейка состоит из одного конденсатора и одного транзистора) и упаковывается с высокой плотностью элементов, поэтому динамическое ЗУ (DRAM) является основным видом компьютерных ОЗУ.

Статическое ОЗУ (SRAM), ячейка которого представляет собой один из вариантов рассмотренных в *главе 16* триггеров, устроено сложнее, имеет меньшую плотность упаковки (т. е. при тех же габаритах меньшую емкость) и стоит гораздо дороже. Главное ее преимущество, кроме того, что она не требует регенерации, — высокое быстродействие и отсутствие потребления в статическом режиме. Выпускаются отдельные микросхемы SRAM, как простые (например, UT62256 с организацией 32K×8), так и довольно «навороченные», — микросхема M48T35, например, кроме собственно массива памяти (32K×8) содержит на кристалле часы реального времени, монитор питания и, главное, имеет встроенную литиевую батарейку, которая позволяет сохранять информацию при отключении питания. Но с распространением энергонезависимой flash-памяти, о которой будет рассказано далее, такие применения SRAM почти потеряли актуальность[1], и за ней остались главные области, где она незаменима: это регистры и кэш-память в микропроцессорах, а также ОЗУ данных в микроконтроллерах и ПЛИС.

---

[1] «Почти» — потому, что SRAM на триггерах является самой быстродействующей разновидностью памяти, и в этом ее преимущество перед медленной EEPROM, несмотря на дороговизну и неудобства, связанные со встроенной резервной батарейкой.

По счастью, с DRAM нам в схемотехническом плане иметь дело не придется, а SRAM мы увидим только в составе микроконтроллеров. Поэтому рассмотрим подробнее более актуальные для пользователя разновидности ROM.

EPROM, EEPROM и flash-память

На заре возникновения памяти, сохраняющей данные при отключении питания (EPROM, Erasable Programmable ROM, стираемая/программируемая ROM, или по-русски — ППЗУ, программируемое ПЗУ), основным типом ее была память, стираемая ультрафиолетом: UV-EPROM (Ultra-Violet EPROM, УФ-ППЗУ). Причем часто приставку UV опускали, т. к. всем было понятно, что EPROM — это стираемая ультрафиолетом, а ROM (или ПЗУ) просто, без добавлений — это однократно программируемые кристаллы OTP-ROM. Микроконтроллеры с УФ-памятью программ были распространены еще в середине 1990-х. В рабочих образцах устройств с УФ-памятью кварцевое окошечко, через которое осуществлялось стирание, заклеивали кусочком черной липкой ленты, т. к. информация в UV-EPROM медленно разрушается и на солнечном свету.

На рис. 18.7 показано устройство элементарной ячейки EPROM, которая лежит в основе всех современных типов flash-памяти. Если исключить из нее то, что обозначено надписью «плавающий затвор», мы получим самый обычный полевой транзистор — точно такой же входит в ячейку DRAM. Если подать на управляющий затвор этого транзистора положительное напряжение, то он откроется, и через него потечет ток (что считается состоянием логической единицы). На рис. 18.7 вверху и изображен такой случай, когда плавающий затвор не оказывает никакого влияния на работу ячейки, — например, подобное состояние характерно для чистой flash-памяти, в которую еще ни разу ничего не записывали.

Если же мы каким-то образом (каким — поговорим отдельно) ухитримся разместить на плавающем затворе некоторое количество зарядов — свободных электронов, которые показаны на рис. 18.7, *внизу* в виде темных кружочков со значком минуса, то они будут экранировать действие управляющего электрода, и такой транзистор вообще перестанет проводить ток. Это состояние логического нуля. Поскольку плавающий затвор потому так и называется, что он «плавает» в толще изолятора (двуокиси кремния), то сообщенные ему однажды заряды в покое никуда деваться не могут. И записанная таким образом информация может храниться десятилетиями (до последнего времени производители обычно давали гарантию на 10 лет, но на практике в обычных условиях время хранения значительно больше).

> ***Заметки на полях***
>
> Строго говоря, в NAND-чипах (о которых далее) логика обязана быть обратной. Если в обычной EPROM запрограммированную ячейку вы не можете открыть подачей считывающего напряжения, то там наоборот — ее нельзя запереть снятием напряжения. Поэтому, в частности, чистая NAND-память выдает все нули, а не единицы, как EPROM. Но это нюансы, которые не меняют сути дела.

Осталось всего ничего — придумать, как размещать заряды на изолированном от всех внешних влияний плавающем затворе. И не только размещать — ведь иногда

память и стирать приходится, потому должен существовать способ их извлекать оттуда. В UV-EPROM слой окисла между плавающим затвором и подложкой был достаточно толстым (если величину 50 нанометров можно охарактеризовать словом «толстый», конечно), и работало все это довольно грубо. При записи на управляющий затвор подавали достаточно высокое положительное напряжение — иногда до 36–40 В, а на сток транзистора — небольшое положительное. При этом электроны, которые двигались от истока к стоку, настолько ускорялись полем управляющего электрода, что просто перепрыгивали барьер в виде изолятора между подложкой и плавающим затвором. Такой процесс называется еще *инжекцией горячих электронов*.

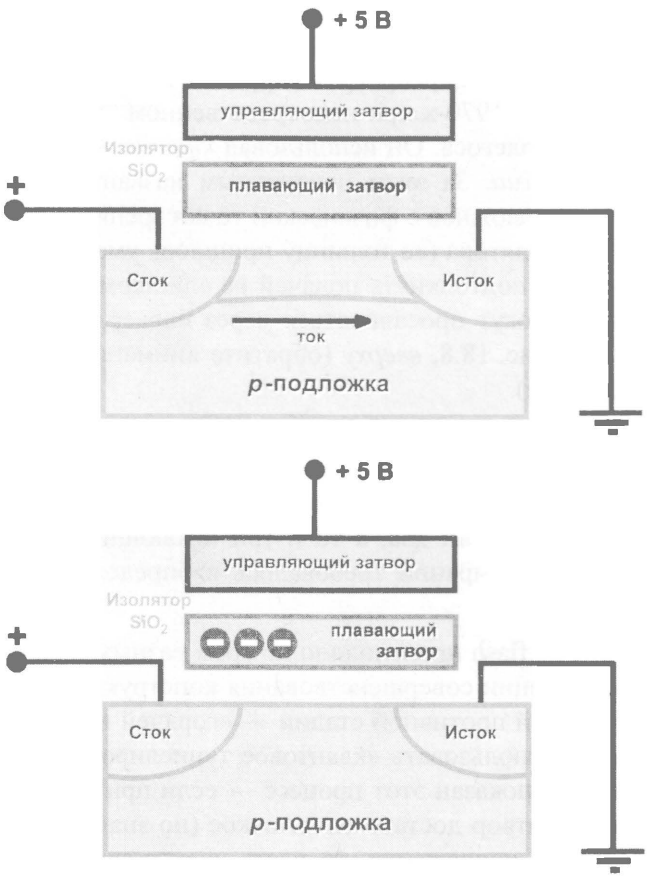


Рис. 18.7. Устройство элементарной ячейки EPROM

Ток заряда при этом достигал миллиампера — можете себе представить, каково было потребление всей схемы, если в ней одновременно программировать хотя бы несколько тысяч ячеек. И хотя такой ток требовался на достаточно короткое время (впрочем, с точки зрения быстродействия схемы не такое уж и короткое — миллисекунды), но это было крупнейшим недостатком всех старых образцов EPROM-

памяти. Еще хуже другое — и изолятор, и сам плавающий затвор такого издевательства долго не выдерживали и постепенно деградировали, отчего количество циклов стирания/записи было ограничено несколькими сотнями, максимум — тысячами. Во многих образцах flash-памяти более позднего времени даже была предусмотрена специальная схема для хранения карты «битых» ячеек — в точности так, как это делается для жестких дисков. В современных моделях с миллионами ячеек такая карта тоже имеется — однако число циклов стирания/записи теперь возросло до сотен тысяч. Как этого удалось добиться?

Сначала посмотрим, как осуществлялось в этой схеме стирание. В UV-EPROM при облучении ультрафиолетом фотоны высокой энергии сообщали электронам на плавающем затворе достаточный импульс для того, чтобы они «прыгнули» обратно на подложку самостоятельно, без каких-либо электрических воздействий. Первые образцы электрически стираемой памяти (EEPROM, Electrically Erasable Programmable ROM, электрически стираемое перепрограммируемое ПЗУ, ЭСППЗУ) были созданы в компании Intel в конце 1970-х при непосредственном участии будущего основателя Atmel Джорджа Перлегоса. Он использовал *квантовый эффект туннелирования Фаулера — Нордхейма*. За этим непонятным названием кроется довольно простое по сути (но очень сложное с физической точки зрения) явление — при достаточно тонкой пленке изолятора (ее толщину пришлось уменьшить с 50 до 10 нм) электроны, если их слегка подтолкнуть подачей не слишком высокого напряжения в нужном направлении, могут просачиваться через барьер, не перепрыгивая его. Сам процесс показан на рис. 18.8, *вверху* (обратите внимание на знак напряжения на управляющем электроде).

Старые образцы EEPROM именно так и работали: запись производилась «горячей инжекцией», а стирание — «квантовым туннелированием». Оттого они были довольно сложны в эксплуатации — разработчики со стажем помнят, что первые микросхемы EEPROM требовали два, а то и три питающих напряжения, причем подавать их при записи и стирании требовалось в определенной последовательности.

Превращение EEPROM во flash происходило по трем разным направлениям. В первую очередь — в направлении совершенствования конструкции самой ячейки. Для начала избавились от самой противной стадии — «горячей инжекции». Вместо нее при записи стали также использовать «квантовое туннелирование», как и при стирании. На рис. 18.8, *внизу* показан этот процесс — если при открытом транзисторе подать на управляющий затвор достаточно высокое (но значительно меньшее, чем при «горячей инжекции») напряжение, то часть электронов, двигающихся через открытый транзистор от истока к стоку, «просочится» через изолятор и окажется на плавающем затворе. Потребление тока при записи снизилось на несколько порядков. Изолятор, правда, пришлось сделать еще тоньше, что обусловило довольно большие трудности с внедрением этой технологии в производство.

Второе направление — ячейку сделали несколько сложнее, пристроив к ней второй транзистор (обычный, не двухзатворный), который разделил вывод стока и считывающую шину всей микросхемы. Благодаря всему этому удалось добиться значи-

тельного повышения долговечности — до сотен тысяч циклов записи/стирания (миллионы циклов, характерные для флэш-карточек, получаются, если добавить схемы коррекции ошибок). Кроме того, схемы формирования высокого напряжения и соответствующие генераторы импульсов записи/стирания перенесли внутрь микросхемы, отчего пользоваться этими типами памяти стало несравненно удобнее — они стали питаться от одного напряжения (5, 3,3 или даже 1,8 В).

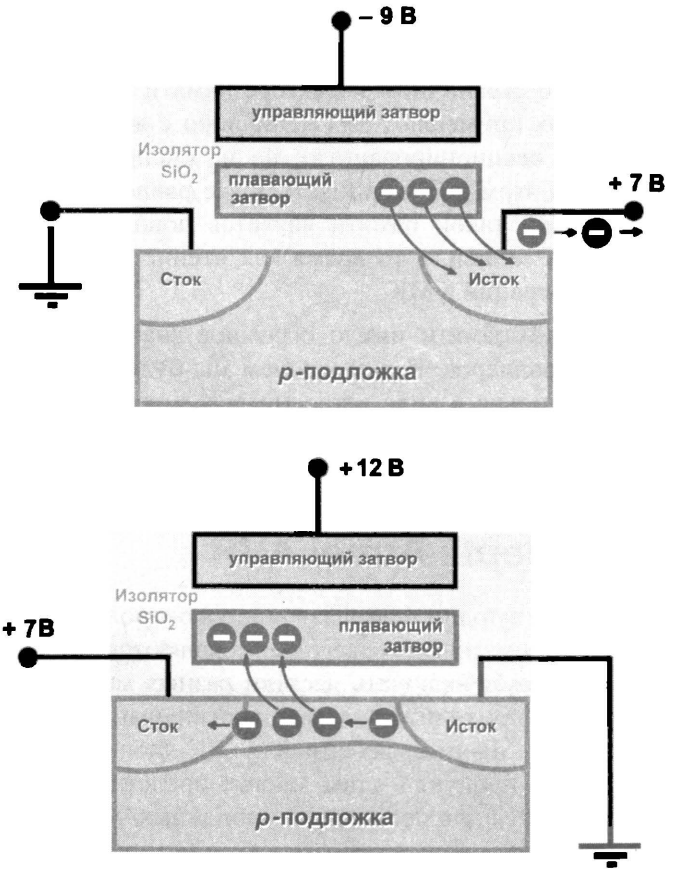


Рис. 18.8. Процессы стирания в элементарной ячейке EPROM

И, наконец, третье, едва ли не самое главное, усовершенствование заключалось в изменении организации доступа к ячейкам на кристалле, вследствие чего этот тип памяти и заслужил наименование — flash (т. е. «молния»), ныне известное каждому владельцу цифровой камеры или карманного MP3-плеера. Так в середине 1980-х назвали разновидность EEPROM, в которой стирание и запись производились сразу целыми блоками — страницами. Процедура чтения из произвольной ячейки, впрочем, по понятным причинам замедлилась — для его ускорения приходится на кристаллах flash-памяти располагать промежуточную (буферную) SRAM. Для флэш-накопителей это не имеет особого значения, т. к. там все равно данные читаются и пишутся сразу большими массивами, но для использования в микроконтроллерах

это может оказаться неудобным. Тем более в МК неудобно задействовать самый быстродействующий вариант flash-технологии — так называемую *память типа NAND* (от наименования логической функции «И-НЕ»), где читать и записывать память в принципе возможно только блоками по 512 байтов (это обычная величина сектора на жестком диске, также читаемого и записываемого целиком за один раз, — отсюда можно понять основное назначение NAND).

В МК обычно используют традиционную (типа NOR) flash-память программ, в которой страницы относительно невелики по размерам — порядка 64–256 байтов. Впрочем, если пользователь сам не берется за создание программатора для такой микросхемы, он может о страничном характере памяти и не догадываться. А для пользовательских данных применяют EEPROM либо с возможностью чтения произвольного байта, либо секционированную, но на очень маленькие блоки — например, по 4 байта. При этом для пользователя все равно доступ остается побайтным. Характерной чертой такой памяти является довольно медленная (порядка миллисекунд) процедура записи, в то время как чтение протекает ничуть не медленнее любых других операций в МК.

Развитие технологий flash-памяти имело огромное значение для удешевления и доступности микроконтроллеров. В дальнейшем мы будем иметь дело с энергонезависимой памятью не только в виде встроенных в микроконтроллер памяти программ и данных, но и с отдельными микросхемами, позволяющими записывать довольно большие объемы информации.

Микроконтроллеры Atmel AVR

Общее количество существующих семейств микроконтроллеров оценивается приблизительно в 100 с лишним, причем ежегодно появляются все новые и новые. Каждое из этих семейств может включать десятки разных моделей. Причем львиная доля выпускаемых чипов приходится на специализированные контроллеры — например, для управления USB-интерфейсом или ЖК-дисплеями. Иногда довольно трудно классифицировать продукт — так, многие представители семейства ARM, которое широко применяется для построения мобильных устройств, с точки зрения развитой встроенной функциональности относятся к типичным контроллерам, но в то же время достаточно мощное ядро позволяет отнести их и к классу микропроцессоров.

Из семейств универсальных 8-разрядных микроконтроллеров, так сказать, «на все случаи жизни», еще недавно были наиболее распространены три: контроллеры классической архитектуры *x*51 (первый контроллер семейства 8051 был выпущен фирмой Intel еще в середине 1980-х), контроллеры PIC фирмы Microchip (применялись для проектирования несложных устройств, особенно предназначенных для тиражирования) и рассматриваемые нами Atmel AVR. В настоящее время из этих трех семейств для любительской практики фактически имеют значение только Atmel AVR. Несомненно, истинным подарком для фирмы Atmel стал поступок итальянских инженеров, выбравших в 2004 году AVR для любительской платформы Arduino, отчего популярность этого семейства быстро выросла, и за его буду-

щее можно не беспокоиться. Об Arduino мы будем подробно говорить в последних главах этой книги.

Заметки на полях

В 1995 году два студента Норвежского университета науки и технологий в г. Тронхейме, Альф Боген и Вегард Воллен, выдвинули идею 8-разрядного RISC-ядра, которую предложили руководству Atmel. Имена разработчиков вошли в название архитектуры AVR: **A**lf + **V**egard + **R**ISC. В Atmel идея настолько понравилась, что в 1996 году был основан исследовательский центр в Тронхейме, и уже в конце того же года начат выпуск первого опытного микроконтроллера новой серии AVR под названием AT90S1200. Во второй половине 1997 года корпорация Atmel приступила к серийному производству семейства AVR.

Почему AVR?

У AVR-контроллеров «с рождения» есть несколько особенностей, которые отличают это семейство от остальных МК, упрощают его изучение и применение. Одним из существенных преимуществ AVR стало использование конвейера. В результате для AVR не существует понятия машинного цикла: большинство команд, как мы говорили, выполняется за один такт (для сравнения отметим, что еще недавно пользовавшиеся большой популярностью МК семейства PIC выполняют команду за 4 такта).

Правда, при этом пришлось немного пожертвовать простотой системы команд, есть некоторые сложности и в области операций с битами. Тем не менее, это не приводит к заметным трудностям при изучении AVR-ассемблера — наоборот, программы получаются короче и больше напоминают программу на языке высокого уровня (отметим, что AVR проектировались специально в расчете на максимальное приближение к структуре языка C).

Другое огромное преимущество AVR-архитектуры — наличие 32 оперативных регистров, не во всем равноправных, но позволяющих в простейших случаях обходиться без обращения к оперативной памяти и, что еще важнее, без использования стека — главного источника ошибок у начинающих программистов (мало того, в младших моделях AVR стек даже недоступен для программиста). Для AVR не существует понятия «аккумулятора», ключевого для ряда других семейств. Это еще больше приближает структуру ассемблерных программ для AVR к программам на языке высокого уровня, где операторы работают не с ячейками памяти и регистрами, а с абстрактными переменными и константами.

Но это, конечно, не значит, что AVR — однозначно лучшее в мире семейство МК в принципе. У него есть и ряд недостатков (например, несовершенная система защиты энергонезависимой памяти данных — EEPROM, некоторые вопросы с помехоустойчивостью, излишние сложности в системе команд и структуре программ и т. п.). А учитывая, что любые универсальные современные МК позволяют делать одно и то же, вопрос выбора платформы — вопрос в значительной степени предпочтений и личного опыта разработчика. Однако возникновение платформы Arduino, с самого начала ориентировавшейся на семейство AVR, выдвинуло эти

контроллеры на передовые позиции, и в своем классе 8-разрядных МК общего назначения семейство AVR теперь можно считать несомненным лидером.

Classic, Mega и Tiny

Линейка универсальных контроллеров AVR общего назначения делится на семейства: Classic, Mega и Tiny (есть и новейшее семейство Xmega, содержащее весьма «навороченные» кристаллы). МК семейства Classic (они именовались как AT90S<*марка контроллера*>) ныне уже не производятся, однако все еще распространены в литературе, т. к. для них наработано значительное количество программ. Чтобы пользователям не пришлось переписывать все ПО, фирма Atmel позаботилась о преемственности — большинство МК семейства Classic имеет функциональные аналоги в семействе Mega, например, AT90S8515 — ATmega8515, AT90S8535 — ATmega8535 и т. п. (только AT90S2313 имеет аналог в семействе Tiny — ATtiny2313).

Полная совместимость обеспечивается специальным установочным битом (из набора так называемых *Fuse-битов*), при программировании которого Mega-контроллер начинает функционировать, как Classic (подробнее об этом рассказано в *главе 19*). Для вновь разрабатываемых устройств обычно нет никакого смысла в использовании их в режиме совместимости, однако такой прием в ряде случаев может оказаться полезным для начинающих, поскольку программы для МК Classic устроены проще и часто встречаются в литературе.

Семейство Tiny (что в буквальном переводе означает «крохотный») предназначено для наиболее простых устройств. Часть МК этого семейства не имеет возможности программирования по последовательному интерфейсу, и потому мы их, за исключением ATtiny2313, в этой книге рассматривать не будем (это не значит, что остальных Tiny следует избегать — среди них есть очень удобные и функциональные микросхемы, нередко вообще не имеющие аналогов). У составляющего исключение МК ATtiny2313 отсутствует бит совместимости с «классическим» аналогом AT90S2313, одним из самых простых и удобных контроллеров Atmel, но при внимательном рассмотрении оказывается, что они и без такого бита совместимы «снизу вверх», — программы для «классического» 2313 практически полностью подходят и для Tiny2313 (см. *главу 19*).

Структура МК AVR

Общая структура внутреннего устройства МК AVR приведена на рис. 18.9. Здесь показаны все основные компоненты AVR (за исключением некоторых специализированных) — в отдельных моделях некоторые компоненты могут отсутствовать или различаться по характеристикам, неизменным остается только общее 8-разрядное процессорное ядро (GPU, General Processing Unit). Кратко рассмотрим наиболее важные компоненты, с большинством из которых мы познакомимся в дальнейшем подробнее.

Начнем с памяти. В структуре AVR имеются три разновидности памяти: flash-память программ, ОЗУ (SRAM) для временного хранения данных и энергонезави-

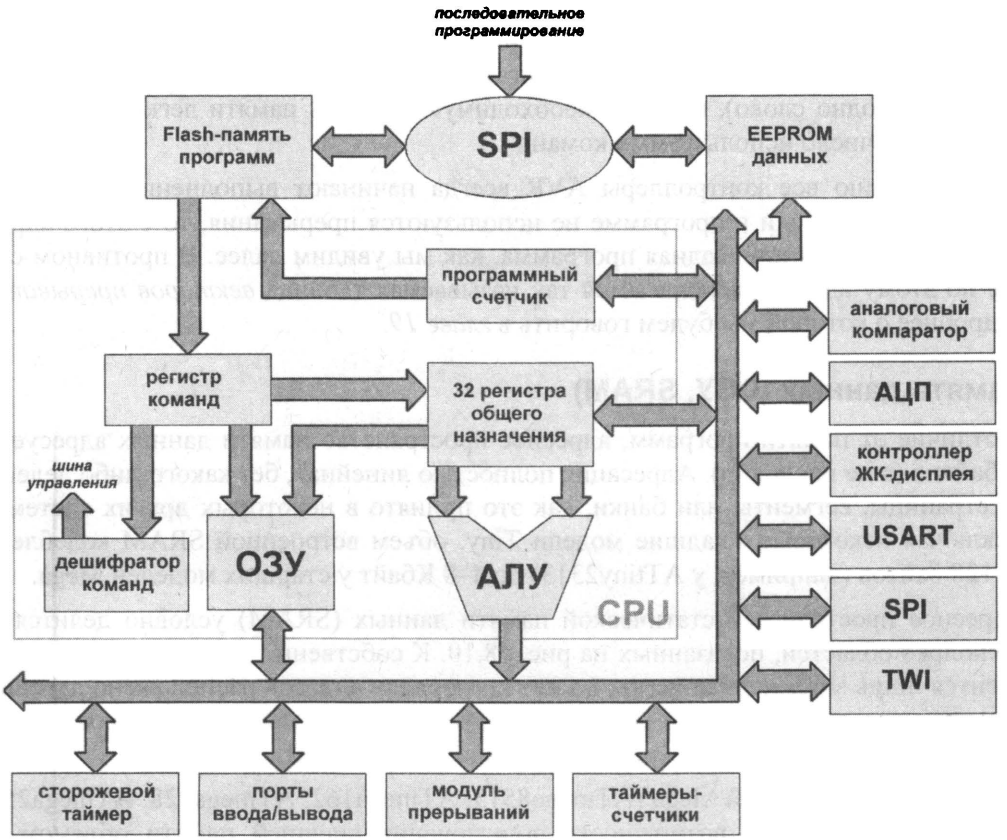


Рис. 18.9. Общая структурная схема микроконтроллеров AVR

симая память (EEPROM) для долговременного хранения констант и данных. Рассмотрим их по отдельности.

Память программ

Встроенная flash-память программ в AVR-контроллерах имеет объем от 1 Кбайт у ATtiny11 до 256 Кбайт у ATmega2560. Первое число в наименовании модели содержит величину этой памяти в килобайтах из ряда: 1, 2, 4, 8, 16, 32, 64, 128 и 256 Кбайт. Так, ATtiny2313 имеет 2 Кбайт памяти, а ATmega8535 — 8 Кбайт.

С точки зрения программиста память программ можно считать построенной из отдельных ячеек — слов по два байта каждое. Устройство памяти программ (и только этой памяти!) по двухбайтовым словам — очень важный момент, который ассемблерному программисту нужно твердо усвоить. Такая организация обусловлена тем, что любая команда в AVR имеет длину ровно 2 байта. Исключение составляют команды `jmp`, `call` и некоторые другие (например, `lds`), которые оперируют с адресами 16-разрядной и более длины. Длина этих команд составляет 4 байта, и они используются лишь в моделях с памятью программ более 8 Кбайт, поэтому в этом

разделе книги вы их не встретите. Arduino основано на AVR-контроллерах с бóльшим объемом памяти, но там нам об этих тонкостях знать необязательно. Во всех остальных случаях счетчик команд сдвигается при выполнении очередной команды на 2 байта (одно слово), поэтому необходимую емкость памяти легко подсчитать, зная просто число используемых команд.

По умолчанию все контроллеры AVR всегда начинают выполнение программы с адреса `$0000`[1]. Если в программе не используются прерывания, то с этого адреса может начинаться прикладная программа, как мы увидим далее. В противном случае по этому адресу располагается так называемая таблица *векторов прерываний*, подробнее о которой мы будем говорить в *главе 19*.

Память данных (ОЗУ, SRAM)

В отличие от памяти программ, адресное пространство памяти данных адресуется побайтно (а не пословно). Адресация полностью линейная, без какого-либо деления на страницы, сегменты или банки, как это принято в некоторых других системах. Исключая некоторые младшие модели Tiny, объем встроенной SRAM колеблется от 128 байтов (например, у ATtiny2313) до 4–8 Кбайт у старших моделей Mega.

Адресное пространство статической памяти данных (SRAM) условно делится на несколько областей, показанных на рис. 18.10. К собственно встроенной SRAM относится лишь затемненная часть, до нее по порядку адресов расположено адресное пространство регистров, где первые 32 байта занимает массив регистров общего назначения (РОН), еще 64 — регистров ввода/вывода (РВВ).

Для некоторых моделей Mega (ATmega8515, ATmega162, ATmega128, ATmega2560 и др.) предусмотрена возможность подключения внешней памяти объемом до 64 Кбайт. Отметим, что адресные пространства РОН и РВВ не отнимают пространство у ОЗУ данных — так, если в конкретной модели МК имеется 512 байтов SRAM, а пространство регистров занимает первые 96 байтов (до адреса `$5F`), то адреса SRAM займут адресное пространство от `$0060` до `$025F` (т. е. от 96 до 607 ячейки включительно). Конец встроенной памяти данных обозначается константой `RAMEND`. Следует учесть, что последние адреса SRAM, как минимум, на четыре-шесть ячеек от конца (в зависимости от количества вложенных вызовов процедур — для надежности лучше принять это число равным десяти или даже более) занимать данными не следует, т. к. они при использовании подпрограмм и прерываний заняты под стек.

Операции чтения/записи в память одинаково работают с любыми адресами из доступного пространства, и потому при работе с SRAM нужно быть внимательным, — вместо записи в память вы легко можете «попасть» в какой-нибудь регистр. Для обращения к РОН, как к ячейкам памяти, можно в качестве адреса подставлять но-

---

[1] Напомним, что в ассемблере AVR можно обозначать шестнадцатеричные числа в «паскалевском» стиле, предваряя их знаком $, при этом стиль языка C (0x00) тоже действителен, а вот интеловский способ (00h) не работает (подробнее об обозначениях чисел различных систем счисления говорилось в *главе 14*).

мер регистра, а вот при обращении к PBB таким же способом к номеру последнего нужно прибавлять `$20`. Следует также помнить, что по умолчанию при включении питания все PBB устанавливаются в нулевое состояние во всех битах (единичные исключения все же имеются, поэтому в критичных случаях надо смотреть документацию), а вот РОН и ячейки SRAM могут принимать произвольные значения.

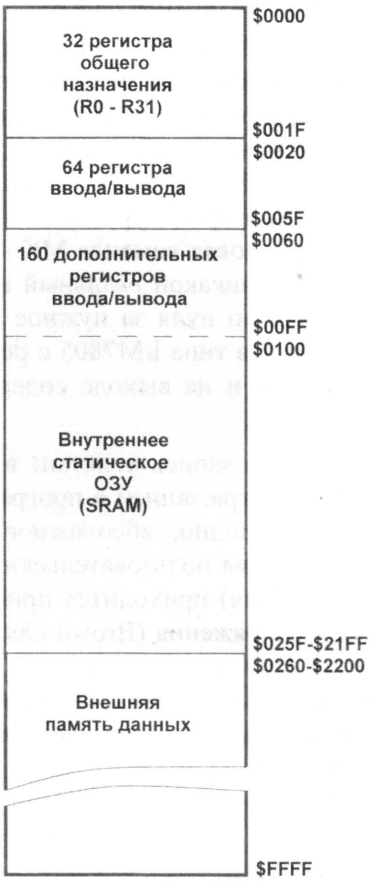


Рис. 18.10. Адресное пространство статической памяти данных (SRAM) микроконтроллеров AVR

Энергонезависимая память данных (EEPROM)

Все модели МК AVR (кроме снятого с производства ATtiny11) имеют встроенную EEPROM для хранения констант и данных при отключении питания. В разных моделях объем ее варьируется от 64 байтов (ATtiny1*x*) до 4 Кбайт (старшие модели Mega). Число циклов перепрограммирования EEPROM может достигать 100 тысяч.

Напомним, что EEPROM отличается от flash-памяти возможностью выборочного программирования побайтно (вообще-то, даже побитно, но эта возможность скрыта от пользователя). Чтение из EEPROM осуществляется с такой же скоростью, как и чтение из РОН, — в течение одного машинного цикла (правда, на практике оно растягивается на 4 цикла, но программисту следить за этим специально не требуется). А вот запись в EEPROM протекает значительно медленнее и к тому же с неоп-

ределенной скоростью — цикл записи одного байта может занимать от 2-х до 4-х и более миллисекунд. Процесс записи регулируется встроенным RC-генератором, частота которого нестабильна (при низком напряжении питания можно ожидать, что время записи будет больше). За такое время при обычных тактовых частотах МК успевает выполнить несколько тысяч команд, поэтому программирование процедуры записи требует аккуратности — например, нужно следить, чтобы в момент записи не «вклинилось» прерывание (подробнее об этом далее).

Главная же сложность при использовании EEPROM — то, что при недостаточно быстром снижении напряжения питания в момент выключения содержимое ее может быть испорчено. Обусловлено это тем, что при снижении напряжения питания ниже некоторого порога (ниже порога стабильной работы, но недостаточного для полного выключения) и вследствие дребезга МК начинает выполнять произвольные команды, в том числе может выполнить и процедуру записи в EEPROM, если она имеется в программе. Если учесть, что типовая команда МК AVR выполняется за десятые доли микросекунды, то ясно, что никакой реальный источник питания не может обеспечить снижение напряжения до нуля за нужное время. По опыту автора при питании от обычного стабилизатора типа LM7805 с рекомендованными значениями емкости конденсаторов на входе и на выходе содержимое EEPROM будет испорчено примерно в половине случаев.

Этой проблемы не должно существовать, если запись констант в EEPROM производится при программировании МК, а процедура записи в программе отсутствует. Во всех же остальных случаях (а их, очевидно, абсолютное большинство — EEPROM чаще всего используется для хранения пользовательских установок и текущей конфигурации при выключении питания) приходится принимать специальные меры. Встроенный детектор падения напряжения (Brown-Out Detection, BOD), имеющийся практически во всех моделях Tiny и Mega, обычно с этим не справляется. Наиболее кардинальной из таких мер является установка внешнего монитора питания, удерживающего МК при снижении напряжения питания ниже пороговой величины в состоянии сброса (подробности см. в [20]).

Способы тактирования

Каноническим способом тактирования МК является подключение кварцевого резонатора к соответствующим выводам (рис. 18.11, *а*). Емкость конденсаторов C1 и C2 в типовом случае должна составлять 22–36 пФ (о включении кварцев см. *главу 15*). В большинстве моделей Tiny и Mega имеется специальный конфигурационный бит CKPOT, который позволяет регулировать потребление. Если он установлен в единицу (незапрограммирован), — размах колебаний уменьшается, однако при этом сужается возможный диапазон частот и общая помехоустойчивость, поэтому использовать этот режим не рекомендуется (см. далее). Может быть также использован низкочастотный кварцевый резонатор (например, «часовой» 32 768 Гц), при этом конденсаторы C1 и C2 можно не устанавливать, т. к. при установке CKPOT в значение 0 подключаются имеющиеся в составе МК внутренние конденсаторы 36 пФ.

Вместо кварцевого можно применить керамический резонатор. Автору этих строк удавалось запускать МК на нестандартных частотах, используя вместо кварца в том

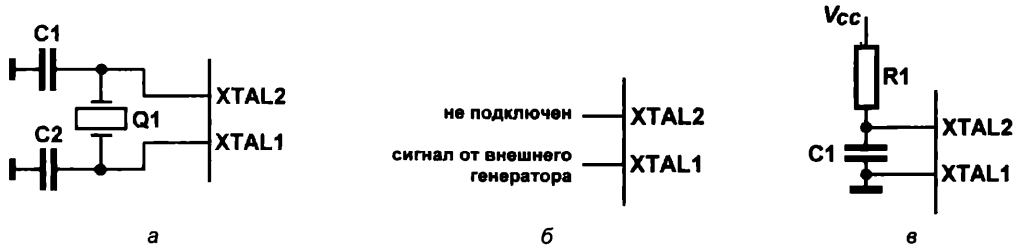


Рис. 18.11. Способы тактирования МК AVR с использованием: *а* — кварцевого резонатора; *б* — внешнего генератора; *в* — *RC*-цепочки

же подключении миниатюрную индуктивность (при ее значении в 4,7 мкГ и емкостях конденсаторов 91 пФ частота получается около 10 МГц).

Естественно, тактировать МК можно и от внешнего генератора (рис. 18.11, *б*). Особенно это удобно, когда требуется либо синхронизировать МК с внешними компонентами, либо иметь очень точную частоту тактирования при использовании соответствующих генераторов (например, серии SG-8002 фирмы Epson). Некоторые внутрисхемные программаторы (см. *главу 19*) используют эту возможность при загрузке программ в микроконтроллер, чтобы не зависеть от установленного режима тактирования.

Наоборот, когда точность не требуется, можно задействовать внешнюю *RC*-цепочку (рис. 18.11, *в*). В этой схеме емкость C1 должна быть не менее 22 пФ, а резистор R1 выбирается из диапазона 3,3–100 кОм. Частота при этом определяется по формуле $F = 2/3\ RC$. C1 можно не устанавливать вообще, если записать логический ноль в конфигурационную ячейку CKPOT, подключив тем самым внутренний конденсатор 36 пФ.

Наконец, можно обойтись вообще без каких-то внешних компонентов — использовать встроенный *RC*-генератор, который может работать на четырех частотах, приблизительно равных 1, 2, 4 и 8 МГц. К этой возможности наиболее целесообразно обратиться в младших моделях Tiny, выпускающихся в 8-контактном корпусе, — тогда выводы, предназначенные для подключения резонатора или внешнего генератора, можно задействовать для других целей, как обычные порты ввода/вывода. Семейство Classic встроенного *RC*-генератора не имело.

По умолчанию МК семейств Tiny и Mega установлены в состояние для работы со встроенным генератором на частоте 1 МГц (CKSEL = 0001), поэтому для работы в других режимах нужно соответствующим образом установить конфигурационные ячейки CKSEL (табл. 18.1). Как это осуществить на практике, будет рассказано в *главе 19*. Рекомендуемое значение этих ячеек для обычных резонаторов от 1 МГц и более: все единицы в ячейках CKSEL, и ноль в CKPOT.

Подробности

При установке ячеек следует учитывать, что состояние CKSEL = 0000 (зеркальное по отношению к наиболее часто употребляемому значению для кварцевого резонатора 1111) переводит МК в режим тактирования от внешнего генератора, и в этом состоянии его нельзя даже запрограммировать без подачи внешней частоты. Также, если вы

попытаетесь установить режим с низкочастотным резонатором, то от высокочастотного МК уже не запустится, а далеко не все программаторы могут работать при таких низких частотах тактирования. Поэтому при манипуляциях с ячейками, и не только CKSEL, нужно быть крайне осторожным и хорошо представлять, что именно вы устанавливаете. Подробнее об этом говорится в *главе 19*.

Таблица 18.1. *Установка конфигурационных ячеек CKSEL в зависимости от режимов тактирования*

CKSEL3...0	CKPOT	Источник тактирования	Частота
0000	1	Внешняя частота	0... 16 МГц
0001	1	Встроенный *RC*-генератор	1 МГц
0010	1	Встроенный *RC*-генератор	2 МГц
0011	1	Встроенный *RC*-генератор	4 МГц
0100	1	Встроенный *RC*-генератор	8 МГц
0101	x	Внешняя *RC*-цепочка	< 0,9 МГц
0110	x	Внешняя *RC*-цепочка	0,9–3,0 МГц
0111	x	Внешняя *RC*-цепочка	3,0–8,0 МГц
1000	x	Внешняя *RC*-цепочка	8,0–12 МГц
1001	x	Низкочастотный резонатор	32 768 кГц
101x	1	Керамический резонатор	0,4–0,9 МГц
110x	1	Керамический резонатор	0,9–3,0 МГц
111x	1	Кварцевый резонатор	3,0–8,0 МГц
1xxx	0	Кварцевый резонатор	≥1,0 МГц

Параллельные порты ввода/вывода

Портов ввода/вывода (повторим, что их не следует путать ни с регистрами ввода/вывода, ни с последовательными портами МК для обмена информацией с внешними устройствами) в разных моделях может быть от 1 до 7. Номинально порты 8-разрядные, в некоторых случаях разрядность ограничена числом выводов корпуса и может быть меньше восьми. Порты обозначаются буквами A, B, C, D и т. д., причем необязательно по порядку, — в младших моделях могут наличествовать, например, только порты B и D (как в ATtiny2313) или вообще только один порт B (как в ATtiny1*x*).

Для сокращения числа контактов корпуса в подавляющем большинстве случаев внешние выводы, соответствующие портам, кроме своей основной функции (двунаправленного ввода/вывода) несут также и дополнительную. Отметим, что кроме как для вывода Reset, если он может работать в альтернативном режиме, никакого специального переключения выводов портов не требуется. Если вы, к примеру, в своей программе инициализируете последовательный порт UART, то соответст-

вующие выводы порта (например, в ATmega8335 это выводы порта PD0 и PD1) будут работать именно в альтернативной функции — как ввод и вывод UART. При этом в промежутках между таким специальным использованием выводов их можно задействовать в качестве обычных двунаправленных выводов. На практике приходится применять схемотехнические меры для изоляции функций друг от друга, поэтому злоупотреблять этой возможностью не рекомендуется.

Типичным примером многообразия функций портов может служить базовый для Arduino ATmega168/328. В нем три порта (B, C и D), но все восемь линий доступны только для порта D, порты B и C — лишь частично, так что программисту доступны максимум 20 цифровых линий. Шесть из них могут быть также использованы как аналоговые входы встроенного АЦП (на платах Arduino помечены буквой A), а некоторая часть остальных также задействована под различные альтернативные функции (последовательные порты, прерывания), применяемые по мере надобности.

Выводы портов в достаточной степени автономны, и их режим может устанавливаться независимо друг от друга. По умолчанию при включении питания все дополнительные устройства отключены, а порты работают на вход, причем находятся в третьем состоянии с высоким *импедансом* (т. е. с высоким входным сопротивлением). Работа на выход требует специального указания, для чего в программе нужно установить соответствующий нужному выводу бит в регистре направления данных (этот регистр обозначается `DDRx`, где `x` — буква, обозначающая конкретный порт, например для порта A это будет `DDRA`). Если бит сброшен (т. е. равен логическому нулю), то вывод работает на вход (как по умолчанию), если установлен (т. е. равен логической единице) — то на выход.

Для установки выхода в состояние единицы нужно отдельно установить соответствующий бит в регистре данных порта (обозначается `PORTx`), а для установки в 0 — сбросить этот бит. Направление работы вывода (вход/выход, регистр `DDRx`) и его состояние (0–1, `PORTx`) путать не следует.

Регистр данных `PORTx` фактически есть просто выходной буфер — все, что в него записывается, тут же оказывается на выходе. Но если установить вывод порта на вход (т. е. записать в регистр направления `DDRx` логический ноль), как это сделано по умолчанию, то регистр данных `PORTx` будет играть несколько иную роль — установка его разрядов в ноль означает, что вход находится в третьем состоянии с высоким сопротивлением, а установка в единицу подключит к выводу «подтягивающий» (pull-up) резистор сопротивлением 20–50 кОм (в семействе Classic оно составляло 35–120 кОм).

Заметки на полях

Встроенного pull-up-резистора в большинстве случаев оказывается недостаточно для надежной работы — из-за наводок МК может сбоить, и параллельно этому внутреннему лучше устанавливать дополнительный внешний резистор с сопротивлением от 1 до 5 кОм (в критичных по отношению к потреблению случаях его величину можно увеличить до 20–30 кОм). Например, если вы подключаете ко входу выносную кнопку с двумя выводами, которая коммутируется на «землю», или вывод работает на «общую

шину» с удаленными (находящимися на другой плате) устройствами, или вывод осуществляет функцию внешнего прерывания (см. далее), то такой дополнительный резистор следует подключать обязательно.

Процедура чтения уровня на выводе порта, если он находится в состоянии работы на вход, не совсем тривиальна. Возникает искушение прочесть данные из регистра данных PORTx, но это ничего не даст — вы прочтете только то, что там записано вами же ранее. А для чтения того, что действительно имеется на входе (непосредственно на выводе микросхемы), предусмотрена другая возможность. Для этого нужно обратиться к некоторому массиву, который обозначается PINx. Обращение осуществляется так же, как и к отдельным битам обычных РВВ (см. *главу 19*), но PINx не есть регистр — это просто некий диапазон адресов, чтение по которым предоставляет доступ к информации из буферных элементов на входе порта. Записывать что-либо по адресам PINx, естественно, нельзя.

Прерывания

Как и в ПК, прерывания (interrupts) в микроконтроллерах бывают двух видов. Но если в ПК прерывания делятся на аппаратные (например, от таймера или клавиатуры) и программные (фактически не прерывания, а подпрограммы, записанные в BIOS, — с распространением Windows это понятие почти исчезло из программистской практики), то в МК, естественно, все прерывания — аппаратные, а делятся они на внутренние и внешние. Любое прерывание отдельно, а также вообще возможность их возникновения, требуют предварительного специального разрешения.

Следует твердо усвоить, что для инициализации любого прерывания надо в программе сделать четыре действия: разрешить прерывания вообще (по умолчанию они запрещены), затем разрешить это конкретное прерывание, установить для него один из доступных режимов и, наконец, установить *вектор прерывания* — указатель на метку, по которой расположена процедура подпрограммы-обработчика прерывания. И, конечно, после этого надо написать сам обработчик, иначе все это будет происходить вхолостую. Подробнее об этом также говорится в *главе 19*.

Внутренние прерывания могут возникать от любого устройства, которое является дополнительным по отношению к ядру системы: от таймеров, от аналогового компаратора, от последовательного порта и т. д. Внутреннее прерывание — это событие, которое возникает в системе и прерывает выполнение основной программы. Система внутренних прерываний в AVR довольно разветвленная и представляет собой основную систему взаимодействия устройств с ядром системы, и к этому вопросу мы еще будем неоднократно возвращаться.

Внешних прерываний у МК AVR как минимум два: INT0 и INT1 (у большинства Mega есть еще третье — INT2, а у основы Arduino Mega, контроллера ATmega2560, их целых семь). Кроме основных внешних прерываний, в ряде моделей (включая и применяющиеся в Arduino ATmega328/2560) есть еще внешние прерывания типа PCINT, на которых мы здесь не будем останавливаться. Внешнее прерывание — событие, которое возникает при появлении сигнала на одном из входов, специально предназначенных для этого. Различаются три вида событий, вызывающих преры-

вание, и их можно устанавливать в программе: это может быть низкий уровень напряжения, а также положительный или отрицательный фронт на соответствующем выводе. Любопытно, что прерывания по всем этим событиям выполняются, даже если соответствующий вывод порта сконфигурирован на выход.

Кратко рассмотрим особенности использования этих режимов. *Прерывание по низкому уровню* (режим установлен по умолчанию, для его инициализации достаточно разрешить соответствующее прерывание) возникает всякий раз, когда на соответствующем входе присутствует низкий уровень. «Всякий раз» — это значит, что действительно всякий, т. е. если отрицательный импульс длится какое-то время, то процедура обработки прерывания, закончившись, повторится снова и снова, не давая основной программе работать. Поэтому обычная схема использования этого режима внешнего прерывания — сразу же по возникновении его запретить (процедура обработки при этом, раз уж началась, один раз выполнится до конца) и разрешить опять только тогда, когда внешнее воздействие должно уже закончиться (например, если это нажатие кнопки, то его стоит опять разрешить по таймеру через одну-две секунды).

В отличие от этого, *прерывания по фронту или спаду* выполняются один раз. Конечно, от дребезга контактов там никакой защиты нет и быть не может, потому что МК не способен отличить дребезг от серии коротких импульсов. Если это критично, нужно либо принимать внешние меры по защите от дребезга, либо использовать тот же способ, что и для прерывания по уровню, — внутри процедуры обработчика прерывания первой же командой запретить само прерывание, а через некоторое время в другой процедуре (по таймеру или по иному событию) опять его разрешить (этот способ «антидребезга» фактически идентичен применению одновибратора, см. *главу 15*).

Подробности

У внимательного читателя возникает законный вопрос — а зачем вообще нужен режим внешнего прерывания по уровню? Дело в том, что оно во всех моделях выполняется асинхронно — в тот момент, когда низкий уровень появился на выводе МК. Конечно, обнаружение прерывания может произойти только по окончании текущей команды, так что очень короткие импульсы могут и пропасть. Но прерывания `INT0` и `INT1` в режиме управления по фронту у большинства моделей определяются наоборот, только синхронно — в момент перепада уровней тактового сигнала контроллера, поэтому длительность импульса не должна быть короче одного периода тактового сигнала. Но это не самое главное — по большому счету разницы в этих режимах никакой бы не было, если бы не то обстоятельство, что синхронный режим требует непременно наличия этого самого тактового сигнала. Потому асинхронное внешнее прерывание, соответственно, может «разбудить» контроллер, находящийся в одном из режимов глубокого энергосбережения, когда тактовый генератор не работает, а синхронное — нет. И старые МК, вроде AT90S8515 семейства Classic (но не его mega-аналога!), могли выводиться из глубокого «сна» только внешним прерыванием по уровню, которое не всегда удобно использовать. У большинства же моделей семейства Mega (из младших моделей — кроме ATmega8 и, кстати, «ардуиновских» 168/328) имеется еще одно прерывание `INT2`, которое происходит только по фронтам (по уровню не может), но, в отличие от `INT0` и `INT1`, асинхронно. В ATtiny2313 (но не в его «классическом» аналоге!) такое асинхронное прерывание может происходить по сигналу с любого из 8 выводов порта B. Асинхронно обнаруживаются и имеющиеся во

многих моделях прерывания типа PCINT, на которых мы обещали здесь не останавливаться. Это значительно повышает удобство пользования контроллером в режиме энергосбережения.

Таймеры-счетчики

В большинстве МК AVR присутствуют два или три таймера-счетчика, один из которых — 16-разрядный, а остальные — 8-разрядные (в старших моделях Mega общее число счетчиков может быть до 6). Все счетчики имеют возможность предварительной загрузки значений и могут работать непосредственно от тактовой частоты (СК) процессора или от нее же, поделенной на 8, 64, 256 или 1024 (в отдельных случаях еще на 16 и 32), а также от внешнего сигнала. В целом устройство таймеров в МК, как мы говорили, похоже на счетчики 561ИЕ11/14 (см. *главу 15*), только функциональность их значительно расширена.

В архитектуре AVR 8-разрядным счетчикам-таймерам присвоены номера 0 и 2, а 16-разрядным — 1, 3 и далее. Некоторые 8-разрядные счетчики (обычно Timer 2, если их два) могут работать в асинхронном режиме от отдельного тактового генератора, причем продолжать функционировать даже в случае «спящего» состояния всей остальной части МК, что позволяет использовать их в качестве часов реального времени.

При использовании счетчиков-таймеров как обычных счетчиков внешних импульсов (причем возможна реакция как по спаду, так и по фронту импульса) частота подсчитываемых импульсов не должна превышать половины частоты тактового генератора МК (причем при несимметричном внешнем меандре инструкция рекомендует еще меньшее значение предельной частоты — 0,4 от тактовой). Это обусловлено тем, что при счете внешних импульсов их фронты обнаруживаются синхронно (в моменты положительного перепада тактового сигнала). Кроме того, стоит учитывать, что задержка обновления содержимого счетчика после прихода внешнего импульса может составлять до 2,5 периода тактовой частоты.

Это довольно сильные ограничения, поэтому, например, использовать МК в качестве универсального частотомера не очень удобно — быстродействующие схемы лучше проектировать на соответствующей комбинационной логике или на ПЛИС (программируемых логических интегральных схемах).

При наступлении переполнения счетчика возникает событие, которое может вызывать соответствующее прерывание. 8-разрядный счетчик Timer 0 в ряде случаев этой функцией и ограничивается. Счетчик Timer 2, если он имеется, может также вызывать прерывание по совпадению подсчитанного значения с некоторой заранее заданной величиной. 16-разрядные счетчики — более «продвинутые» и могут вызывать прерывания по совпадению с двумя независимо заданными числами A и B. При этом счетчики могут обнуляться или продолжать счет, а на специальных выводах при этом могут генерироваться импульсы (аппаратно, без участия программы).

Кроме того, 16-разрядные счетчики могут осуществлять «захват» (capture) внешних одиночных импульсов на специальном выводе. При этом может вызываться прерывание, а содержимое счетчика помещается в некий регистр. Сам счетчик при этом

может обнуляться и начинать счет заново или просто продолжать счет. Такой режим удобно использовать для измерения периода внешнего сигнала или для подсчета неких нерегулярных событий (вроде прохождения частиц в счетчике Гейгера). Немаловажно, что источником таких событий может быть также встроенный аналоговый компаратор, который тогда используется как формирователь импульсов.

Все счетчики-таймеры могут работать в так называемых *режимах PWM*, т. е. в качестве 8-, 9-, 10- или 16-битных *широтно-импульсных модуляторов* (ШИМ), причем независимо друг от друга, что позволяет реализовать многоканальный ШИМ. В технической документации режимам PWM, в силу их сложности, многовариантности и громоздкости, посвящено много страниц. Простейший вариант использования этих режимов — воспроизведение звука. Их также можно задействовать для регулирования мощности или тока (например, при зарядке аккумуляторов), управления двигателями, выпрямления сигнала, при цифроаналоговом преобразовании. Такие применения МК AVR значительно упростились с появлением платформы Arduino, и им посвящено множество доступных интернет-ресурсов. Мы кратко рассмотрим возможности PWM-режима в Arduino в *главе 22*.

Кроме таймеров-счетчиков, во всех без исключения AVR-контроллерах есть сторожевой (Watchdog) таймер. Он предназначен в основном для вывода МК из режима энергосбережения через определенный интервал времени, но может использоваться и для аварийного перезапуска МК. Например, если работа программы зависит от прихода внешних сигналов, то при их потере (например, из-за обрыва на линии) МК может «повиснуть», а Watchdog-таймер выведет его из этого состояния (см. также *главу 22*).

Последовательные порты

Последовательные порты для обмена данными с внешними устройствами — важнейшая составляющая любого МК, без них его «общение» с внешним миром резко ограничено. *Последовательными* их называют потому, что в них в каждый момент времени передается только один бит (в некоторых случаях возможна одновременная передача и прием, но все равно только по одному биту за раз). Самое главное преимущество последовательных портов перед параллельными (когда одновременно производится обмен целыми байтами или полубайтами-тетрадами) — снижение числа соединений. Но оно не единственное — как ни парадоксально, но последовательные интерфейсы дают значительную фору параллельным на высоких скоростях, когда на надежность передачи начинают влиять задержки в линиях. Последние невозможно сделать строго одинаковыми, и это одна из причин того, что последовательные интерфейсы в настоящее время начинают доминировать (типовые примеры — USB и FireWire вместо LPT и SCSI, или Serial ATA вместо IDE).

В микроконтроллерных устройствах с нашими объемами данных, конечно, скорость передачи нас волнует во вторую очередь, но вот количество соединительных проводов — очень критичный фактор. Поэтому все внешние устройства, которые мы далее станем рассматривать, будут иметь последовательные интерфейсы (кроме

дисплеев для отображения информации, для которых, увы, последовательные интерфейсы встречаются лишь в моделях достаточно высокого уровня).

Практически любой последовательный порт можно имитировать программно, используя обычные выводы МК. Когда-то так и поступали даже в случае самого популярного из таких портов — UART. Однако с тех пор МК обзавелись аппаратными последовательными портами, что, впрочем, не означает необходимости их непременного использования. Легкость программной имитации последовательных портов — еще одно их достоинство.

Из всех разновидностей портов, которые могут наличествовать в МК AVR, мы особенно обратим внимание на UART (**U**niversal **A**synchronous **R**eceiver-**T**ransmitter, универсальный асинхронный приемопередатчик). UART есть основная часть любого устройства, поддерживающего протокол RS-232, но и не только его (недаром он «универсальный») — например, промышленные стандарты RS-485 и RS-422 также реализовываются через UART, т. к. они отличаются от RS-232 только электрическими параметрами и допустимыми скоростями, а не общей логикой построения. В персональных компьютерах есть COM-порт, который работает по тому же протоколу RS-232, и узел UART точно так же является его базовой частью.

Поэтому UART служит основным способом обмена данными МК с компьютером. Отметим, что отсутствие COM-порта в большинстве современных моделей ПК не является препятствием — существуют переходники USB-COM, а в настольную модель можно вставить дополнительную карту с COM-портами. О том, как обращаться с UART на практике, рассказывается в *главах 21* и *22* применительно к платформе Arduino — программировать такой обмен на ассемблере гораздо сложнее (хотя и надежнее, см. далее). В *главе 21* мы увидим, что существуют простые и при этом достаточно надежные способы организовать передачу через последовательный порт по радиоканалу, что позволяет обойтись вообще без проводов. Существуют и Arduino-модули, позволяющие подключить такой порт к сети через Wi-Fi — например, из-за своей простоты и универсальности приобретает растущую популярность китайская разработка ESP8266 (она может использоваться и как самостоятельный контроллер для несложных задач).

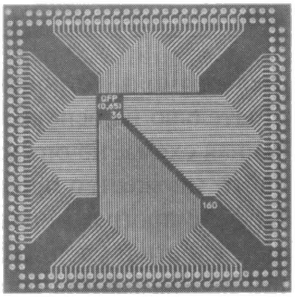
Кроме UART, почти все МК AVR содержат самый простой из всех последовательных портов — SPI (**S**erial **P**eripheral **I**nterface, последовательный периферийный интерфейс). Об устройстве SPI упоминалось в *главе 16*. Его принципиальная простота сыграла отчасти дурную роль — трудно встретить два устройства, где протоколы SPI полностью совпадают, обычно обмен по этому порту сопровождается теми или иными «наворотами». Следует отметить, что программирование AVR также осуществляется через SPI, однако в общем случае этот интерфейс и SPI для обмена данными — разные вещи, хотя в большинстве случаев выводы у них одни и те же. Кстати, всем знакомые карты памяти («флэшки») также адресуются через протокол, очень близкий к SPI.

Кроме этих портов, часто применяется очень простой аппаратно, но более сложный с программной точки зрения и довольно медленный интерфейс I^2C. В терминологии Atmel AVR он называется TWI (**T**wo-**W**ire **I**nterface, двухпроводной интер-

фейс). С его помощью можно общаться со многими устройствами: часами реального времени, компасами, датчиками, некоторыми разновидностями памяти. Мы рассмотрим его опять же в главах, посвященных Arduino.

В AVR имеется 10-разрядный АЦП последовательного приближения (см. *главу 17*). Непосредственная работа с ним на ассемблерном уровне имеет довольно много нюансов, и далее вы увидите, насколько Arduino упрощает этот процесс.

И вообще, некоторые другие узлы МК семейства AVR мы рассмотрим по ходу изложения конкретных схем — так будет нагляднее. Сейчас же мы закончим затянувшееся знакомство с микроконтроллером и обратимся к вопросу о том, как его программировать. Следующие две главы мы посвятим элементарным сведениям о программировании МК на ассемблере, а далее перейдем к языкам высокого (и даже сверхвысокого) уровня. Так вы сможете наглядно сравнить и даже при желании «пощупать руками» преимущества и недостатки того и иного подхода и границы их применимости.



ГЛАВА 19

Персональный компьютер вместо паяльника

О программировании МК на языке ассемблера

— Чтобы найти дорогу в Лондон, надо уметь говорить по-английски. По-моему, дело это очень трудное.

А. Дюма. «Три мушкетера»

Внедрение любой новой технологии требует начальных затрат. Не составляет исключения и микропроцессорная технология. В данном случае прямые затраты будут состоять в том, что вам, во-первых, придется приобрести программатор, и, во-вторых, компьютер — если по какой-то непостижимой случайности у вас его до сих пор нет. Сразу скажем, что мобильные ПК, под которыми ныне подразумеваются планшеты и смартфоны, можно приспособить к этой задаче только теоретически — на практике потребуется настольный компьютер или ноутбук под управлением Windows или, в крайнем случае, Linux (за применимость Mac OS я не отвечаю, хотя некоторые программаторы и декларируют свою применимость в этой среде). Причем компьютер для наших целей сгодится совершенно любого поколения, подойдет даже самый древний (есть программаторы, которые работают и с DOS). Это будет неудобно только потому, что вы не сможете параллельно пользоваться современными программами для отладки, потому предпочтительно все-таки иметь систему под Windows, лучше, если версии не позднее «семерки». Затраты будут сведены к минимуму, если программатор приобрести специализированный для AVR-контроллеров, он стоит на порядок меньше универсальных. Большинство современных программаторов общаются с компьютером через универсальный порт USB (создавая через него виртуальный COM-порт), так что в этом отношении проблем не ожидается.

Железо

Раз уж мы начали с потребного «железа», то закончим эту тему, а потом перейдем к собственно программированию. Программатор, о котором идет речь, называется *ISP-программатором* (**I**n **S**ystem **P**rogramming, т. е. программирование осуществля-

ется прямо в устройстве пользователя). Программатор можно соорудить и самостоятельно на основе любого AVR-контроллера — на сайте Atmel есть рекомендованная схема, но она уже очень устарела. Это не проблема, т. к. в Интернете можно найти множество предложений самодельных программаторов такого рода (ибо интерфейс программирования AVR не составляет секрета), но их функциональность, надежность и удобство пользования часто оставляют желать лучшего[1]. Учтите, что ISP-программатор, особенно совместимый с официальным протоколом Atmel (т. е. со средами программирования для AVR, включая и Arduino), предполагает в своей основе контроллер с программой загрузки (bootloader), т. е. для его изготовления в свою очередь нужен хоть какой-нибудь программатор (см. также описание проблемы и рекомендации в статье «Программаторы для AVR-микроконтроллеров» на сайте **ph0en1x.net**[2]).

Хлопоты по приобретению программатора можно дополнительно минимизировать, если воспользоваться платформой Arduino, в которую USB-программатор попросту встроен, как неотъемлемая часть (а программа загрузки уже имеется в контроллере). Но эти возможности, о которых мы расскажем в последующих главах, по умолчанию предполагают использование языка высокого уровня и наличие фирменных плат Arduino (или какой-либо из ее клонов), причем загрузчик уже занимает определенное место в памяти, и его наличие замедляет запуск контроллера. Однако программное обеспечение Arduino IDE включает утилиту AVRDUDE, позволяющую программировать МК AVR напрямую, не пользуясь ни средой Arduino, ни заранее встроенным в контроллер загрузчиком. Информацию о том, как обращаться с AVRDUDE, можно найти на множестве ресурсов в Сети[3]. Эта утилита и связанные с ней программные средства (вроде WinAVR), в частности, позволяют применять простейший самодельный программатор, подключаемый к порту LPT.

Чтобы со всем этим не возиться и не разгребать проблемы, связанные с самодеятельными конструкциями, стоит сразу покупать фирменный программатор, хотя он может быть и дороже. Прежде чем покупать программатор, следует внимательно разобраться, с каким именно софтом он работает, потому что это определяет комфортность его использования — есть, например, программаторы, ориентированные на программу AVRReal, которая не имеет графического интерфейса и запускается из командной строки. Неплохой программатор под названием AVR-ISP500, совместимый с официальным протоколом, продается в «Чипе-Дипе» — он довольно дорогой, но зато «умеет» исправлять ошибки в установке fuse-битов (см. далее). Очень удобные программаторы — и тоже не очень дешевые — выпускает фирма Argussoft (AS-2/3/4), в настоящее время можно приобрести версию AS2M, работающую через COM-порт и AS3E и AS4E, работающие через USB.

Для того чтобы работать с ISP-программатором, естественно, его надо куда-то подключить. Для этого на программируемой плате специально устанавливают про-

---

[1] Конструкции различных самодельных программаторов можно найти, например, на сайте **http://avr.ru/ready/tools/prog**.

[2] **https://ph0en1x.net/73-avr-microcontroller-programmers-usbasp-isp-com-lpt.html**.

[3] Например, тут: **http://www.myrobot.ru/stepbystep/mc_programmer.php**.

граммирующий разъем. ISP-программаторы используют один и тот же тип разъема — игольчатый PLD (PL double, т. е. двухрядный), который хорошо знаком всем, кто когда-нибудь подсоединял жесткий диск с IDE-интерфейсом к материнской плате. Естественно, для ISP-программаторов требуется гораздо меньше контактов, чем для жесткого диска. Минимальное их количество равно 6 (именно столько их у программатора, рекомендуемого самой фирмой Atmel, такой же разъем предусмотрен на всех платах Arduino) — это выводы SPI-интерфейса программирования: MOSI, MISO и SLK, а также Reset и два вывода питания +5 В и «земля» (ISP-программаторы обычно питаются от программируемой схемы).

Указанные выводы SPI-интерфейса присоединяются к одноименным выводам кристалла, которые есть у всех МК AVR, имеющих возможность SPI-программирования. У программаторов Argussoft традиционно нестандартный 10-контактный разъем для ISP-программирования, но комплект для изготовления адаптера к обычному 6-контактному разъему прилагается к программатору (вот его, кстати, и совсем нетрудно соорудить самостоятельно). Есть и другие программаторы, использующие этот вид ISP-разъема. Нумерация контактов того и другого разъема приведена на рис. 19.1, серыми линиями там показаны соединения для создания переходника от одного вида разъема к другому.

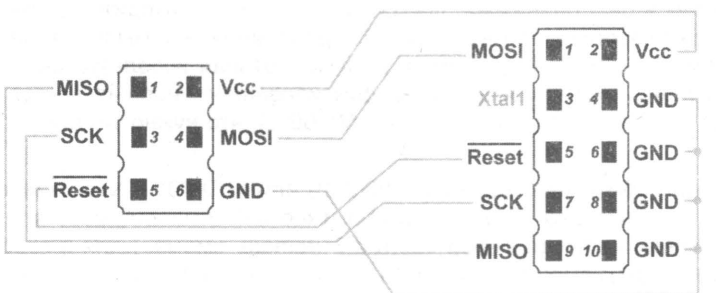


Рис. 19.1. Нумерация контактов разъемов для ISP-программирования: *слева* — стандартный 6-контактный; *справа* — 10-контактный

Подробности

На 10-контактном ISP-разъеме имеется вывод 3, который в обычном режиме никуда не подключается. Но некоторые программаторы (например, упомянутый AVR-ISP500) имеют возможность управлять через этот вывод тактовой частотой микроконтроллера, если его подключить к выводу Xtal1 внешней частоты (см. *разд. «Способы тактирования»* в *главе 18*). Это позволяет не зависеть от установленного режима тактирования, в том числе исправить неверно установленный режим с помощью fuse-битов (см. далее).

При отсчете выводов имейте в виду, что нумерация контактов разъемов типа PLD и IDC (см. *«Подробности»* далее) отличается от обычной и делается не в обход, как у микросхем, а следующим образом: если глядеть на штыри сверху (т. е. со стороны подсоединения ответной части), то вывод номер 1, как и положено, находится слева внизу, вывод 2 — над ним, вывод 3 — следующий за первым в нижнем ряду

и т. д., т. е. нижний (*левый* на рис. 19.1) ряд — это все нечетные, а верхний (*правый* на рис. 19.1) — все четные контакты. Это сделано потому, что такие разъемы могут иметь не строго фиксированное количество контактов, но при любом их количестве имеющиеся контакты будут нумероваться одинаково. Если не поняли, то поглядите на разводку любого PLD-разъема на материнской плате компьютера, которая обычно приведена прямо на ней (а также, как правило, имеется в руководстве).

Подробности

Так как игольчатые разъемы типа PLD и IDC с шагом 2,54 мм встречаются в практике изготовления микроэлектронных устройств довольно часто, то стоит разобраться в их маркировке. Начнем с того, что наименование IDC в случае штыревых разъемов для установки на плату относится к разъемам в кожухе с ключом (именно такие используются для подсоединения жесткого диска в ПК). Бескорпусные подобные разъемы носят название PLD для двухрядных (или PLS для однорядных) типов и более удобны в радиолюбительской практике, так как длинные разъемы легко отломать в нужном месте, чтобы обеспечить необходимое количество выводов (правда, при этом лучше все же обозначать на плате первый вывод, чтобы не перепутать ориентацию при включении). Разъемы типа PLD или PLS устанавливают в том числе и на платах Arduino. Следует учитывать, что обозначение PLD относится только к двухрядным штыревым разъемам, устанавливаемым на плату. Соответствующие розетки, устанавливаемые на плату, называются PBD, а устанавливаемые на кабель — BLD (разъемы IDC-типа все называются одинаково, см. далее). Разметка на плате для обоих типов разъемов (и с кожухом и без) одинакова, поскольку все равно приходится учитывать место, которое займет кабельная розетка при ее подсоединении, и мы в этой книге вперемешку упоминаем IDC и PLD. Разумеется, розетка, которая используется для установки на плоский кабель, может иметь только фиксированное число контактов (из ряда 6, 10, 14, 16, 20, 22, 24, 26, 30, 34, 36, 40, 44, 50, 60...), что нужно учитывать при проектировании.

Цифра после обозначения разъема (IDC-10 или PLD-10) — это количество контактов разъема, а следующая буква, если она имеется, символизирует его конфигурацию: M (male, «папа») для штыревой части и F (female, «мама») для гнездовой. Поскольку конфигурация разъемов PLD (PBD, BLD) уже заложена в их обозначение, то для этих типов буквы M и F в конце обозначения не указываются. Далее может следовать еще одна буква, которая обозначает ориентацию: S для прямых выводов (разъем перпендикулярен плате), R для повернутых под углом 90° (разъем параллелен плате). Таким образом, приведенное далее на схеме рис. 19.3 обозначение IDC-10MS означает штыревой («папа») разъем в кожухе с ключом и с 10-ю прямыми выводами. Соответствующая этому разъему кабельная часть обозначается как IDC-10F.

Вопрос, который при этом немедленно возникает: не жирно ли занимать как минимум три вывода портов (обычно это порт B контроллера) альтернативными функциями? Отвечаю: в большинстве случаев такие выводы можно использовать как обычные порты, программатору это не помешает. Единственная ситуация, в которой возникает конфликт между программатором и схемой, это если выводы MOSI, MISO и SLK используются как входные линии и подсоединены к активным выходам других микросхем, к коллектору открытого транзистора или к конденсатору большой емкости. Поэтому в схеме эти контакты лучше использовать в качестве выходных линий, а если хватает других портов — вообще оставить их только для программирования. На платах Arduino, где контакты штатного SPI-порта совпадают с портом программирования, они выведены как обычные линии (выводы D11,

D12 и D13[1]), и в случае их использования для программирования к ним тоже не рекомендуется подключать низкоомную нагрузку. Естественно, при работе схемы программатор лучше отключать вовсе, только учтите, что по окончании цикла программирования в любом случае схема заработает сразу с начала, как после включения питания.

Все эти проблемы несущественны, если работать с универсальным программатором, где используется параллельное программирование. Тогда при программировании МК приходится вставлять в специальную колодку на корпусе программатора, а в схеме, естественно, для него тогда должна быть предусмотрена установка на панельку. Преимущество универсальных программаторов в том, что можно легко переходить с одного семейства процессоров на другое, причем, независимо от типа корпуса, а также программировать любые микросхемы памяти. Не нужен также и программирующий разъем на плате и связанные с ним резисторы. Но в процессе отладки сложной программы пользоваться универсальным программатором крайне неудобно — МК все время приходится вынимать и вставлять, что сильно замедляет процесс, к тому же есть риск повредить выводы. Поэтому универсальный программатор следует рекомендовать в случаях, когда программа уже более или менее отлажена, — например, при тиражировании конструкции.

Софт

Если мы имеем какой-то программатор с прилагающимся к нему софтом, то, в сущности, нам нужна еще только одна специальная программа — ассемблер (assembler значит «сборщик»). Его можно бесплатно скачать с сайта Atmel в составе AVR Studio (папка **avrassembler**), файл носит название avrasm2.exe. Практически все существующие ассемблеры запускаются из командной строки (хотя могут быть и упакованы в оболочку с графическим интерфейсом). В качестве параметров для компилятора Avrasm2 указывается имя файла с исходным текстом программы и имена выходных файлов, главным из которых является файл с расширением hex. Чтобы каждый раз не вводить длинную командную строку, пишется соответствующий BAT-файл.

Подробности

Предположим, у вас файл *avrasm2.exe* находится в созданной вами папке *c:\avrtools*. Запустите Блокнот и введите следующий текст (соответственно измените путь, если папка другая):

```
c:\avrtools\avrasm2 -fI %1.asm
```

Строка эта может выглядеть и несколько иначе:

```
c:\avrtools\avrasm2 -e %1.eep -fI %1.asm
```

---

[1] Отметим, что на платах Arduino и в текстах программ цифровые выводы именуют просто номерами, тогда как аналоговые — с добавлением буквы A. В дальнейшем во избежание неопределенности мы к номеру цифрового вывода на схемах и в тексте книги будем добавлять привычную букву D, но следует помнить, что в тексте скетчей этого не требуется.

В этом случае в той же папке, что и HEX-файл, создастся файл с расширением *eep*, содержащий данные для загрузки в EEPROM. Причем работать это будет только, если в тексте программы есть соответствующая директива для создания такого файла, в противном случае результат будет одинаковым в обоих случаях (более подробно мы этот вопрос рассматривать не будем).

Сохраните созданный файл под названием, например, *avrasm.bat*. Пусть текст созданной вами программы находится в файле *programm.asm*, тогда достаточно в командной строке запустить *avrasm.bat* с параметром *programm* (если надо, то с путем к нему, а расширение добавится автоматически), и в той же папке, где находится последний, создастся файл *programm.hex*. При этом откроется DOS-окно, в котором будут проанализированы ошибки, если они есть (тогда выходной файл не создастся), а если все в порядке — указан объем полученной программы в двухбайтных словах (учтите, что размер HEX-файла ни о чем не говорит).

Полученный в результате ассемблирования HEX-файл с программой представляет собой текстовый файл (а не бинарный, как обычные исполняемые компьютерные файлы), но содержащий только числа в байтовом представлении в шестнадцатеричной записи. Он имеет строго определенную структуру, разработанную в свое время фирмой Intel. Этот HEX-файл и есть та программа в процессорных кодах, которую мы загружаем в МК с помощью различных программаторов (в том числе среда Arduino тоже создает такой файл). При этом программатор автоматически располагает ее в памяти программ МК, начиная с нулевого адреса.

Исходные тексты ассемблерных программ можно создавать в любом текстовом редакторе (разве что к результатам деятельности Microsoft Word следует относиться с осторожностью). Но, несмотря на широкий выбор, есть по крайней мере две причины, по которым лучше все же использовать редакторы специализированные. Первая причина — это так называемый highlighting или *подсветка синтаксиса* по-русски. Те, кто пользовался любыми средами высокоуровневого программирования (от Turbo Pascal до Delphi или Visual Basic), хорошо знают, что это такое — служебные слова, комментарии, разные типы выражений выделяются каждый своим цветом или шрифтом, что сильно облегчает чтение текста и служит заодно неплохим средством проверки правильности написания. Но если эту опцию предлагает множество фирменных и не очень редакторов, то вторая желательная функция есть лишь у считанных единиц. Я имею в виду возможность прямо из редактора с помощью горячих клавиш запускать процесс ассемблирования. В этом случае вы можете «не отходя от кассы», т. е. не покидая редактор, одним нажатием горячих клавиш сразу же ассемблировать написанный текст и ознакомиться с сообщениями об ошибках.

Еще одна причина для использования специализированных редакторов — они автоматически нумеруют строки. Причем пустые строки также входят в нумерацию — так проще считать. Если у вас есть ошибки в программе, то ассемблер укажет номер строки с ошибкой, так что нумерация строк принципиально важна. Один из рекомендуемых вариантов редакторов для AVR-ассемблера — «самопальный» редактор ASM Editor (не путать с Asmedit!), который сделан на удивление профессионально, хотя и не без некоторых досадных огрехов.

Все сказанное относится к программированию на языке ассемблера, потому что программы для AVR-контроллеров можно писать и на C, C++, ориентированном на

непрофессионалов языке Processing/Wiring (Arduino!) и многих других, включая даже специальные версии Pascal (mikroPascal for AVR[1]). Почему-то довольно популярен Bascom AVR, использующий Basic-подобный язык[2]. (Уж на что я не люблю С-подобные языки, и к тому же немало в свое время соорудил программ на самых разных версиях Бейсика, но хуже идеи, чем приспособить этот и без того противоестественный язык к контроллерам, придумать трудно.)

Для тех, кто уже владеет программированием на языках высокого уровня, их применение может показаться более удобным способом, и это действительно так во многих случаях. Только следует четко понимать, что, как и в «большом» программировании, каждый язык и даже каждая среда программирования — это своя отдельная экосистема. И применительно к AVR хорошо развиты только две из них, во многом совместимые, потому что ориентированы на близкородственные языки на основе С: это профессиональная среда, группирующаяся вокруг AVR Studio (поддерживающей несколько разных компиляторов с языка С), и любительская Arduino, также основанная на языке C/C++. Знакомиться со средой Arduino IDE мы еще будем в дальнейшем, и там вы сами сможете «пощупать руками» все ее достоинства и недостатки, а здесь важно подчеркнуть, что пройти мимо языка С вы не сможете.

Сейчас же я скажу только, что для углубленного изучения контроллеров я бы не рекомендовал начинать не только со среды Arduino, но и вообще с программирования на С и тем более других языках высокого уровня. Не столь важно то, что в результате на ассемблере получается более быстрый и компактный код, сколько то, что любой посредник (а компилятор С есть именно посредник) в этом деле только мешает понимать, что именно происходит в контроллере. Как мы увидим далее, программы на С, скомпилированные для контроллеров AVR, устроены и работают совсем иначе, чем ассемблерные — именно поэтому только последние могут полностью использовать все возможности платформы AVR. Конечно, если вы сможете одолеть непростую для освоения AVR Studio, то это совсем неплохо, потому что она предоставляет удобные и наглядные средства отладки программ, но даже в ней я рекомендую начинать серьезное изучение именно с ассемблера.

Следует, однако, учесть, что по мере усложнения программ и перехода к профессиональному программированию МК обойти освоение С уже невозможно, как минимум, по двум причинам. Во-первых, большинство профессиональных библиотек-подпрограмм существуют именно на С, и среди них есть вещи, которые повторить на ассемблере (как и на Bascom или mikroPascal, кстати) будет, как минимум, весь-

---

[1] Информацию о mikroPascal for AVR можно найти по ссылке: **http://we.easyelectronics.ru/AVR/mikropascal-for-avr-osobennosti-yazyka.html**. Белградская компания MikroElektronika за вполне вменяемые деньги (порядка 150–200 долларов) предлагает среды программирования для разных типов микроконтроллеров на разных языках. Среди них и есть и реализация microPascal для AVR: **http://www.mikroe.com/mikropascal/avr/**. Есть короткий курс по этой среде и на русском языке: **http://cxem.net/mc/mc261.php**. Отметим, что mikroPascal не единственная реализация Pascal для контроллеров Atmel AVR, — подробнее об этом см. по адресу: **http://en.wikibooks.org/wiki/Embedded_Systems/Atmel_AVR#Pascal**.

[2] **http://decada.org.ru/project/lessons/bascom_avr/**.

ма затруднительно. Во-вторых, потому что в ассемблере отсутствуют достаточно развитые средства структурирования, и отлаживать большую программу, содержащую более пары-другой тысяч ассемблерных операторов, — мучение. Ну а, конечно, когда углубленное изучение не предполагается, а стоит чисто утилитарная задача один раз воспроизвести ту или иную конструкцию — здесь такие среды, как Arduino, оказываются вне конкуренции.

Все, что сказано в этой главе далее, относится только к программированию МК AVR «в чистом виде», т. е. на ассемблере без дополнительных инструментов-посредников.

О конфигурационных битах

Эта напасть свалилась на нас с появлением семейств Tiny и Mega, в «классических» AVR ничего такого не было (точнее, было, но специально заботиться об установке этих битов не требовалось). В англоязычной инструкции конфигурационные биты называют *fuse-bits*. Их появление привело к многочисленным проклятиям на голову фирмы Atmel со стороны армии любителей, которые стали один за другим «запарывать» кристаллы при программировании. Положение усугублялось тем, что в описании этих сущностей используется извращенная логика, — как мы знаем, ячейки любой чистой EEPROM (по принципу ее устройства) содержат единицы, и слово «запрограммированный» по отношению к такой ячейке означает, что в нее записали логический ноль. Термины *запрограммированный* и *незапрограммированный* как раз и применяются в фирменных описаниях AVR, и оттуда перекочевали в ряд самодельных программаторов — готовьтесь к тому, что в некоторых программаторах отмеченный галочкой в меню программы бит означает его равенство логической единице, а в других — запрограммированное состояние, т. е. логический ноль. Поэтому разработчики программаторов AS из фирмы Argussoft даже специально предусмотрели в окне программирования конфигурационных ячеек памятку на этот счет (рис. 19.2).

В этом окне сейчас приведено безопасное рабочее состояние конфигурационных ячеек для ATmega8535, причем выпуклая кнопка означает единичное состояние ячейки, а нажатая — нулевое (и не путайтесь с этим самым «запрограммированным» состоянием!). Для разных моделей набор fuse-битов различный, но означают они одно и то же, потому мы рассмотрим типовое их состояние на этом примере. Перед первым программированием нового кристалла просто один раз установите эти ячейки в нужное состояние, дальше их уже менять не потребуется. Излишне говорить, что в среде Arduino ни о каких fuse-битах заботиться не приходится — все уже сделано за вас.

Самые необходимые пункты тут такие. По умолчанию любая микросхема семейств Mega или Tiny запрограммирована на работу от внутренней *RC*-цепочки, за что разработчикам большое спасибо, иначе было бы невозможным первичное программирование по SPI — только через параллельный программатор. Для работы с обычным «кварцем», присоединенным по типовой схеме, как мы говорили в *гла-*

ве 18, требуется установить все ячейки CKSEL0-3 в единицы (что согласно логике разработчиков означает *незапрограммированное* их состояние). Заметим, что это и ведет к критической ошибке — решив при поверхностном чтении написанного по-английски руководства, что установка всех единиц означает запрограммировать все ячейки, пользователь смело устанавливает их на самом деле в нули, отчего микросхема переходит в состояние работы от внешнего генератора, и разбудить ее через SPI-интерфейс уже невозможно. Легче всего в этом случае переустановить fuse-биты с помощью программатора, который это позволяет (см. *разд. «Железо»* ранее), либо, за неимением такового, попробовать-таки подключить внешний генератор, как описано в руководстве.

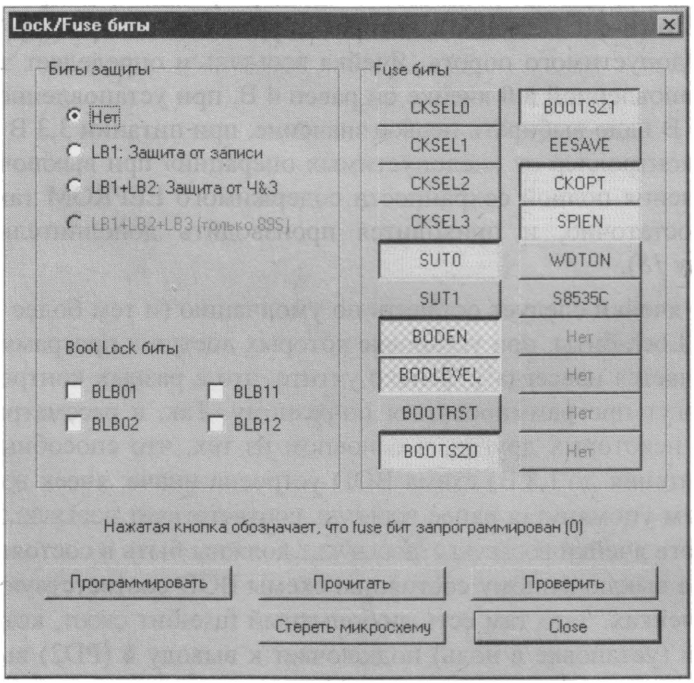


Рис. 19.2. Окно типового состояния конфигурационных ячеек в нормальном режиме работы ATmega8535

Это самая крупная ошибка, которую можно допустить, но есть и другие менее распространенные. Ячейка SPIEN разрешает/запрещает последовательное программирование по SPI и должна оставаться в нулевом состоянии, иначе МК не «разбудишь» никак, только с помощью параллельного программатора (говорят, правда, через SPI ее и отключить невозможно). Ячейка S8535C (в других моделях она будет иметь другое название или вовсе отсутствовать) — очень важна и определяет режим совместимости с семейством Classic (в данном случае с AT90S8535). Если ее установить в нулевое состояние, то МК семейства Mega перейдет в режим совместимости. В популярном ATtiny2313, потомке «классического» AT90S2313, такого бита совместимости нет (но, как мы увидим, это не очень существенно). При

использовании режима совместимости следует учесть, что состояния МК нельзя перемешивать: если fuse-бит совместимости запрограммирован (равен 0), то программа компилируется полностью, как для семейства Classic (в том числе с использованием соответствующего INC-файла, см. далее), иначе она может не заработать.

Еще одна важная ячейка — EESAVE, которая на рис. 19.2 установлена в единицу (режим по умолчанию), но ее целесообразно перевести в нулевое состояние — тогда при программировании памяти программ не будет стираться содержимое EEPROM. Ячейки SUT определяют длительность задержки сброса, и в большинстве случаев принципиального значения их состояние не имеет.

Наконец, для нас в дальнейшем будет иметь значение состояние ячеек BODEN и BODLEVEL. Первая, будучи установлена в ноль, разрешает работу так называемой *схемы BOD* (Brown-out Detection), которая сбрасывает контроллер при снижении питания ниже допустимого порога. Ячейка BODLEVEL и определяет этот самый порог — при установленной в 0 ячейке он равен 4 В, при установленной в 1 — 2,7 В. При питании 5 В надо выбирать первое значение, при питании 3,3 В — второе. Это предохраняет контроллер от «недопустимых операций» при выключении питания, но для обеспечения полной сохранности содержимого EEPROM таких мер может оказаться недостаточно, и приходится производить дополнительные действия (см. также *главу 18*).

Все остальные ячейки следует оставить по умолчанию (и тем более ни в коем случае не трогать Lock-биты, при установке которых доступ к программе и fuse-битам вообще отключается навсегда!). Только учтите, что в разных контроллерах одни и те же узлы могут программироваться по-разному. Так, в рассматриваемом далее ATtiny2313 (и некоторых других, в основном из тех, что способны работать при напряжении питания до 1,8 В) схема BOD устроена иначе: ячеек BODLEVEL там целых три, причем упомянутая ранее BODLEVEL соответствует BODLEVEL0, а при тех же значениях порога ячейки BODLEVEL2:BODLEVEL1 должны быть в состоянии 10. Ячейки BODEN там нет, а выключенному состоянию схемы BOD соответствуют все единицы во всех трех ячейках. Зато там есть любопытный fuse-бит CKOUT, который при программировании (установке в ноль) подключает к выводу 6 (PD2) выход тактового генератора, и еще бит CKDIV8, который в том же состоянии снижает тактовую частоту в восемь раз (например, при кварце 8 МГц МК заработает, как от 1 МГц, что снижает потребление). Поэтому для каждого конкретного контроллера следует сверяться с техническим описанием.

Пожалуй, это все, что нужно знать до того, как вы приступите к собственно программированию. Данные, относящиеся к конкретным контроллерам и программаторам, вы доберете из руководств к ним, которые придется изучать так или иначе.

Примеры программирования

Написание программ — целое искусство, поэтому здесь можно на нескольких примерах попытаться дать только общее представление о том, как это делается, — заостряя внимание на специфических особенностях программирования для МК, а

не программирования вообще. Полного обзора команд, правил обращения с ассемблером и даже комментариев по каждой встречающейся команде вы также здесь не увидите — это потребовало бы отдельной книги. См. [19], а также книгу автора [20] и многочисленные пособия в Сети, из которых, например, можно порекомендовать статьи автора под ником DI HALT из серии «AVR. Учебный курс» на сайте **easyelectronics.ru**. Довольно внятно основы ассемблера изложены в статье «Первая программа для AVR-микроконтроллера на ассемблере» на сайте **ph0en1x.net** и так далее — найти подобные ресурсы не представляет трудностей.

Заметки на полях

Хочу обратить внимание читателя на такую возможность AVR-ассемблера, как создание макросов. Макросы описаны во многих пособиях по AVR-ассемблеру (см. например, вот эту страничку: **http://cxem.net/mc/book24.php**), очень просты для понимания и выполнения. В любительской среде ими почему-то пользуются редко, а профессионалы — сплошь да рядом. Мы тоже макросы обходим в своем изложении, но забывать об этой возможности не следует. Механизм макросов позволяет создать фактически новый язык — надстройку над ассемблером, существенно сокращающую время программиста, но вместе с тем сохраняющую все преимущества ассемблера, как способа программирования, максимально приближенного к структуре контроллера.

Самая простая программа

Давайте напишем сначала самую простую программу, которая будет включать светодиод сразу при включении питания и больше ничего не делать. Для примера возьмем контроллер ATtiny2313, схема подключения которого вместе с программирующим разъемом показана на рис. 19.3.

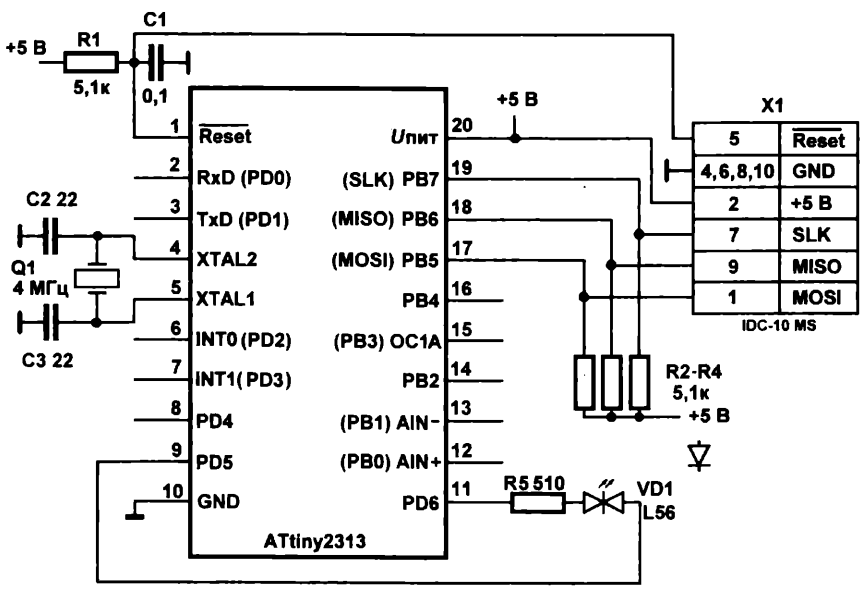


Рис. 19.3. Схема подключения ATtiny2313

Подробности

Обратим внимание на *RC*-цепочку (R1 и C1), которая подсоединена к выводу Reset. Она служит для более надежного сброса контроллера при не слишком качественном источнике питания, если установление напряжения затягивается. В техническом описании указано, что по выводу Reset уже имеется встроенный фильтр дребезга с «подтягивающим» резистором, так что эта цепочка оказывается вроде бы и ненужной (ее установка рекомендовалась для семейства Classic, где фильтра не было). Тем не менее, для более надежной работы МК рекомендуется во всех случаях устанавливать резистор R1 с номиналом 3–10 кОм (встроенный резистор имеет слишком большой номинал). Что касается конденсатора C1, то если для управления сбросом применяется внешний монитор питания, как описано в *главе 18*, то, разумеется, он будет только мешать, в остальных случаях он необязателен, но делает работу схемы более стабильной (есть подобная RC-цепочка и на платах Arduino). Резисторы R2–R4 при наличии программирующего разъема устанавливать желательно, в противном случае надежность схемы снижается (на платах Arduino их не устанавливают, видимо, полагаясь на усовершенствования в последних версиях контроллеров ATmega).

Заметим, что в схеме на рис. 19.3 показана только небольшая часть функциональности входных/выходных линий, которые все, включая даже Reset и выводы подсоединения кварца (кроме, конечно, выводов питания), имеют как минимум две, а большинство даже три функции. Подробно обо всех этих функциях можно узнать из технического описания, частично вы с ними познакомитесь из дальнейшего изложения.

Светодиод мы сразу взяли двухцветный — в целях дальнейшего усовершенствования схемы. Светодиод L56 (можно также взять L57 или малогабаритный L36) светится зеленым, если плюс питания находится со стороны ключа (скоса на корпусе), и красным — если наоборот. Здесь мы его подключили к выводам PD5 и PD6 порта D. Таким образом, если на этих выводах будет одинаковый уровень (неважно, единица или ноль), то светодиод погашен, если разный — то в зависимости от того, на каком выводе единица, светодиод будет гореть красным или зеленым. Токоограничивающий резистор R5 при логическом уровне на выходе порта примерно 4,5 В обеспечит ток через светодиод около 5 мА.

Пусть мы для начала хотим просто зажечь светодиод при включении питания, демонстрируя, что контроллер работает. Для определенности выберем зеленое свечение. Тогда нам необходимо создать программу, которая делает следующее:

1. Конфигурирует нужные выводы порта D (PD5 и PD6) на выход.
2. Устанавливает на выводе PD6 логическую единицу (логический ноль на выводе PD5 устанавливается по умолчанию).

И что, программа будет состоять всего из двух команд? Увы, не все так просто. В регистры портов записывать непосредственное значение нельзя, можно только командой `out` переносить информацию из какого-нибудь рабочего регистра. Поэтому добавится третья команда — сначала мы установим нужные биты в некоем рабочем регистре, специально выбранном для этих целей из числа регистров общего назначения (РОН), затем загрузим весь полученный байт в регистр порта. Причем установить непосредственное значение можно также не в любом из РОН, а только в регистрах с номерами с 16 по 31, поэтому выберем себе регистр r16 в качестве рабочего. Тогда последовательность команд будет выглядеть так:

```
ldi r16,0b01100000 ;устанавливаем биты номер 5 и 6 в регистре r16
out DDRD,r16 ;выводим это значение в регистр направления порта D
sbi PortD,6 ;устанавливаем в единицу бит 6 регистра данных порта D
```

Здесь `ldi` — команда загрузки непосредственного значения (load immediate), `out` — команда вывода в какой-либо регистр ввода/вывода (PBB), `sbi` (set bit input/output) — команда установки бита с выбранным номером в PBB. А что означает запись `0b01100000`? Это хорошо знакомое нам двоичное представление числа — `0b` впереди как раз и означает, что это именно двоичная запись (см. *главу 14*). Таким образом в нашем случае удобнее отсчитывать биты, но если мы запишем это же число в HEX-форме `0x60` (или `$60`), или даже просто десятичное `96`, ничего не изменится. Заметим, что способ установки значений PBB через команду `out` — не единственный, можно было бы установить биты 5 и 6 регистра `DDRD` двумя командами `sbi` (есть и другие способы).

Подробности

Обратите внимание на синтаксис команд — сначала пишется, *что* делать, затем *куда* писать, а потом, через запятую, *откуда* или *что* (так называемая «польская» или «прямая польская» нотация). Это справедливо для всех команд и для всех ассемблеров. Каждая команда пишется в отдельной строке. Весь текст после точки с запятой ассемблером игнорируется и представляет собой комментарии, которые нужно писать обязательно, иначе вы через пару месяцев и сами в своей программе не разберетесь. Если комментарий переходит на другую строку, то точку с запятой нужно ставить заново, в начале строки. В отличие от знака перевода строки, все пробелы и табуляции игнорируются (а там, где есть другой разделительный знак, в нашем случае — запятая, как видите, пробелов вообще может и не быть), так что украшать текст отступами можно, а для удобства чтения программы и нужно. Строчные и прописные буквы не различаются: записи `LDI`, `Ldi` и `ldi` означают одно и то же — и это отличие AVR-ассемблера от программных сред, основанных на C/C++. Вопреки распространяемым в Интернете сведениям (заимствованным из [21]), это правило соблюдается и для обновленных версий компилятора Avrasm2.

Ну, а теперь все? Можно скопировать это в Блокнот, сохранить с расширением asm и компилировать в HEX-файл? Ишь разбежались — нет, не все. Во-первых, никаких таких `DDRD` и `PortD` AVR-ассемблер не понимает. Если соответствия кодов команд и их мнемонических обозначений (`ldi`), а также обозначений рабочих регистров (`r16`) и их адресов, зашиты в ассемблере, то адреса PBB могут меняться от модели к модели, а их названия и вообще могут быть выбраны совершенно произвольным образом. Сам ассемблер понимает только конкретные числа, представляющие собой адреса этих регистров. Но писать программу без мнемонических обозначений было бы крайне неудобно, т. к. она оказалась бы совершенно нечитаемой (вторая команда, к примеру, тогда выглядела бы так: `out $11,r16`). Поэтому к нашей программе надо «пристегнуть» файл с мнемоническими обозначениями, который поставляется Atmel и в данном случае называется tn2313def.inc (при компиляции он должен находиться в одном каталоге с файлом программы). Это делается почти в точности, как в языке C, строкой:

```
.include "tn2313def.inc" ;точка впереди обязательна!
```

Замечание

Файлы макроопределений (с расширением *inc*), к сожалению, нельзя скачать с сайта Atmel в отдельности. Как и файл компилятора Avrasm2, эти файлы для каждого контроллера в отдельности проще всего добыть из пакета AVR Studio, где они находятся в папке *avrassembler/include*. Лучше, если это будет одна из последних версий AVR Studio, пятая или более поздняя, иначе при компилировании совместно с Avrasm2 возможны ошибки.

Если вы заглянете внутрь файла tn2313def.inc, то увидите, что он состоит из строк, начинающихся с директивы `.equ`. Мы могли бы не включать его в программу (в память контроллера он все равно не записывается, ибо не содержит команд, а только определения переменных), а лишь дописать к программе в начале текста строки:

```
.equ DDRD = $11
.equ PortD = $12
```

Мы часто будем применять директиву `.equ`, которая устанавливает соответствие между числами и их обозначениями, наряду с другой полезной директивой — `.def`. Если провести аналогии с языком Turbo Pascal или Turbo C, то директива `.equ` (от англ. *equal* — равно) полностью аналогична определению констант, а директива `.def` (от англ. *define* — определить) аналогична определению переменных, с единственным отличием: тип переменной здесь не указывается, ибо он один-единственный — число размером один байт. А вот в директиве `.equ` может быть указано число любого размера, а также и отрицательное, но, естественно, только целое.

Неудобно каждый раз писать `r16` и помнить, что это у нас рабочая переменная для всяких текущих надобностей, потому лучше дописать еще такую строку:

```
.def temp = r16 ;рабочая переменная, от слова temporary (временный)
```

Окончательно исходный текст программы будет выглядеть так:

```
;программа зажигания светодиода
;процессор Tiny2313, частота 4 МГц
.include "tn2313def.inc"
.def temp = r16 ;рабочая переменная
ldi temp,0b01100000 ;устанавливаем биты номер 5 и 6 в temp
out DDRD,temp ;выводим это значение в регистр направления порта D
sbi PortD,6 ;устанавливаем в единицу бит 6 регистра данных порта D
sleep
.exit
```

Программа займет в памяти программ контроллера ровно 8 байтов. Последняя команда `sleep` означает остановку процессора и выход в режим экономии — ведь должен процессор что-то делать по окончании программы? Директива `.exit` предназначена для ассемблера и означает конец программы, указывать ее необязательно.

А зачем мы в заголовочном комментарии указали тактовую частоту процессора? В данном случае она не имеет значения (лишь бы контроллер работал), но при использовании любых процедур, связанных со временем, это критично. И поскольку можно забыть, на какую частоту вы рассчитывали при написании программы, следует ее на всякий случай указывать в комментариях.

Заметки на полях

Программирование для контроллеров, как могли заметить грамотные читатели, имеет одну особенность, отличающую его от обычного программирования, например, под Windows. «Обычные» программисты как огня боятся замкнутых циклов, условие выхода из которых может не выполниться хотя бы теоретически. Бесконечное ожидание прихода символа с клавиатуры или из COM-порта, в котором не предусмотрено никаких альтернативных средств выхода из цикла, «повесит» не только саму программу, но и всю систему, если речь идет об однозадачной DOS или «кооперативной многозадачности» Windows 3.*x*, может доставить немало неприятностей в «компромиссных» Windows 9*x*, и даже в настоящих многозадачных ОС, по крайней мере, заставит обрывать работу программы принудительно. Потому в таких задачах программисты «на уровне подкорки» приучены применять средства, позволяющие выйти из такого цикла альтернативными методами: по таймеру, или какими-то пользовательскими действиями (нажатием клавиши <ESC> или кнопки **Cancel**). И любая «правильная» программа под Windows состоит из последовательных вызовов операторов `if <вызов функции> else <альтернативные действия>`, где функция при невозможности выполнения сама просигнализирует о том, что надо делать что-то другое.

В программировании для МК подход иной. И в фирменных «аппнотах», и в примерах в тексте технических описаний контроллеров, и в высокоуровневых программах для Arduino вы запросто можете встретить бесконечный цикл ожидания очистки какого-нибудь бита, и в теории, если этот бит никогда не очистится, МК так и «повиснет». ПК-программист от такой ситуации пришел бы в ужас. Конечно, есть средства вывести контроллер из этого состояния (см. *главу 22*), но незачем: скажем, если по ходу дела требуется послать что-то в компьютер, а связь оказывается неисправной, то, очевидно, функциональность системы и так окажется нарушенной, и тут уже выходом из «зависания» делу не поможешь. Тем не менее, в особо ответственных проектах, когда зависание может вызвать неприятные последствия (скажем, термостат, зависший при выводе информации на дисплей, может вскипятить или заморозить воду в аквариуме), следует этот недостаток иметь в виду и стараться изменять стандартные процедуры.

Таймер без прерываний

Давайте теперь заставим наш МК управлять этим светодиодом так, чтобы он мигал с частотой примерно один раз в секунду из красного в зеленый. И сначала сделаем это самым простым способом — так, как это делали в те времена, когда микропроцессоры еще не были микроконтроллерами и не содержали никаких дополнительных узлов вроде таймеров. Для отсчета времени тогда пользовались тем фактом, что команды выполняются строго определенное время. Причем в AVR этот способ применять особенно удобно, поскольку большинство команд занимают один такт, за исключением команд передачи управления. Этим способом часто пользуются и по сей день для отсчета программных задержек (не станешь же заводить таймер по каждому случаю?), потому урок окажется не совсем бесполезным. Заодно познакомимся с понятием процедур (подпрограмм) и с самими командами передачи управления.

Не вникая в подробности, сразу напишем «правильную» процедуру, позволяющую формировать заданные задержки без таймера. Назовем ее `Delay`, тогда она запишется так:

```
Delay:
 subi Razr0,1
 sbci Razr1,0
```

```
 sbci Razr2,0
 brcc Delay
ret
```

Здесь `Razr0—Razr2` — рабочие регистры. Отведем для них регистры r17, r18 и r19. В начало программы тогда следует внести их определения через команду `def` (по образцу `.def Razr0 = r17`). `Delay` с двоеточием — метка, в данном случае обозначающая начало процедуры, команда `ret` — выход из процедуры (зачем она нужна, пояснено далее). Команда `subi` вычитает из регистра константу, в данном случае единицу. А команды `sbci` работают хитрее — они также вычитают константу, но с учетом переноса. Если переноса нет, то они просто ничего не делают (ибо вычитаемое значение равно нулю). Перенос же возникает тогда, когда в результате предыдущей команды вычиталась единица из нуля. Тогда значение регистра меняется с нулевого на все единицы (255), а перенос записывается в специальный бит переноса и учитывается следующей командой `sbci`. В этой схеме легко узнать принцип работы соединенных между собой двоичных счетчиков из *главы 16*, в которых выход старшего разряда предыдущего счетчика соединен со входом переноса следующего. В рассматриваемом случае счетчик состоит из трех отдельных байтовых регистров, т. е. всего имеет 24 двоичных разряда.

Нам не надо, чтобы счет продолжался до бесконечности, и для этой цели служит команда `brcc`, которая относится к группе команд передачи управления (branch) и работает по очень простому правилу — если в момент ее исполнения бит переноса (он обозначается буквой C, от слова carry, что и значит «перенос») равен нулю (clear, т. е. очищен), то далее выполняется возврат к команде по метке `Delay`. То есть название команды (`brcc`) расшифровывается так: **br**anch if **c**arry **c**lear (перейти, если перенос очищен). В противном случае управление передается на следующую команду — выхода из процедуры `ret`, счет заканчивается.

Легко сообразить, что в момент выполнения команды `brcc` перенос станет равен единице только тогда, когда все регистры в предыдущем такте были равны нулю. Поэтому вся процедура работает так: перед ее началом в счетчики `Razr0—Razr2` записывается некое заданное число, которое в каждом такте уменьшается на единицу, и вся процедура заканчивается по достижении нулевого значения во всех разрядах. Отсюда, зная тактовую частоту МК и время выполнения команд (по такту на вычитание и два такта на возврат к начальной метке), легко вывести формулу: записываемое в счетчики число N, соответствующее нужному интервалу времени T (секунд) при тактовой частоте F (герц), рассчитывается, как $T \cdot F/(M+2)$, где M — число регистров-счетчиков, в нашем случае 3.

Число N в данном случае трехбайтовое. Старший байт записывается в `Razr2`, младший — в `Razr0`. При тактовой частоте 4 МГц мы можем получить задержку до 4,19 с, если запишем в регистры все единицы: число 16 777 215 = $FFFFFF. Если же нам требуется, например, задержка в 1 секунду, то придется записать число 800 000 или $0C3500, если в полсекунды — число 400 000 или $061A80.

Вооружившись этими знаниями, попробуем соорудить программу мигалки. Сначала давайте определимся с алгоритмом переключения из красного в зеленый. Наи-

более универсальный метод здесь такой — каждый раз будем определять, в каком состоянии в тот или иной момент находится светодиод (по состоянию какого-нибудь из задействованных выводов порта), и переключать его в противоположное. Это может выглядеть таким образом:

```
;начало мигания
 in temp,PortD ;загружаем в temp состояние PortD
 sbrc temp,5 ;если PD5 равен 0, след. команду пропускаем
 rjmp set_Red ;переход на установку красного
 sbi PortD,5 ;PD5=1, устанавливаем зеленый
 cbi PortD,6 ;PD6=0, сбрасываем красный
 rjmp mig_end ;переход на продолжение программы
set_Red: ;метка установка красного
 cbi PortD,5 ;PD5=0, сбрасываем зеленый
 sbi PortD,6 ;PD6=1, устанавливаем красный
mig_end: ;метка конец мигания
```

Не углубляясь в подробности, заметим, что такой алгоритм (он, конечно, не единственно возможный) будет устойчив к начальным условиям — в каком бы состоянии к моменту его начала ни находились биты 5 и 6 (даже если светодиод был совсем погашен), максимум через один цикл они придут в противоположное состояние, и мигание начнется. Теперь осталось соединить все это в законченную программу — повторять процедуру мигания через промежуток времени, определяемый процедурой `Delay`. Программа будет выглядеть таким образом:

```
;Программа мигания светодиода
;процессор Tiny2313, частота 4 МГц
.include "tn2313def.inc"
.def temp = r16 ;рабочий регистр
.def Razr0 = r17 ;разряды задержки
.def Razr1 = r18
.def Razr2 = r19
;========== code =========
ldi r16, low(RAMEND) ;только для 2313
out SPL,r16 ;установка указателя стека
ldi temp,0b01100000 ;устанавливаем биты номер 5 и 6 в temp
out DDRD,temp ;выводим это значение в регистр направления порта D
Mig_begin: ;начало мигания
 in temp,PortD ;загружаем в temp состояние PortD
 sbrc temp,5 ;если PD5 равен 0, след. команду пропускаем
 rjmp set_Red ;переход на установку красного
 sbi PortD,5 ;PD5=1, устанавливаем зеленый
 cbi PortD,6 ;PD6=0, сбрасываем красный
 rjmp mig_end ;переход на продолжение программы
set_Red: ;метка установка красного
 cbi PortD,5 ;PD5=0, сбрасываем зеленый
 sbi PortD,6 ;PD6=1, устанавливаем красный
```

```
mig_end: ;конец мигания
 ;пауза 0,5 с, N = $061A80h
 ldi Razr2,$06 ;начальное значение счетчиков
 ldi Razr1,$1A
 ldi Razr0,$80
rcall Delay; вызываем задержку
rjmp Mig_begin ;опять в начало
;============= end code =======
Delay: ;процедура задержки
 subi Razr0,1
 sbci Razr1,0
 sbci Razr2,0
 brcc Delay
ret
```

О назначении первых двух операторов программы мы поговорим позднее. `Rcall` — команда вызова процедуры (в данном случае `Delay`). После этой команды обязательно должна где-то встретиться команда `ret` (возврата из процедуры), иначе программа к основной последовательности операторов вернуться не сумеет. Сама программа представляет собой бесконечный цикл, т. к. заканчивается оператором безусловного перехода обратно в начало (`rjmp Mig_begin`). В данном случае мигание будет происходить с периодом в секунду (полсекунды красный, полсекунды зеленый), для другого периода нужно изменить записываемое в счетчики значение. Программа может служить хорошей основой для любимого развлечения радиолюбителей — конструирования елочных гирлянд.

Заметки на полях

Кстати, о любимом когда-то развлечении радиолюбителей — елочных гирляндах. На мой вкус, самый приличный вариант построения такой гирлянды, который к тому же не встретишь в продаже, — это обеспечить вместо раздражающего мигания медленное изменение из одного цвета в другой. Его несложно построить, используя явление *биений*: если сложить два почти совпадающих по частоте колебания через элемент, например, «Исключающее ИЛИ», то на выходе возникнет колебание с меняющейся скважностью (см. рис. 15.8, *г*, аналогичный эффект можно получить и с другими логическими функциями). Если напряжение такой формы подать на лампочку, то высокочастотные составляющие не будут заметны, и лампочка станет медленно зажигаться и гаснуть. При очень близких, но все-таки не равных частотах, период биений может составить минуты, а если взять две разноцветные лампочки или два светодиода и включить их в противофазе, то гирлянда начнет медленно менять цвет с одного на другой. Причем в качестве одной из частот удобно взять частоту электрической сети, которая никогда точно не равна 50 Гц.

Для осуществления этого проекта на дискетной логике пришлось бы городить довольно громоздкую схему на счетчиках с предзагрузкой, причем долго подгонять коэффициент деления так, чтобы смена цветов не была слишком быстрой. А вот с помощью контроллера такая задача решается, что называется, в пять секунд. Для одной лампочки или светодиода почти не нужно даже менять схему по сравнению с рис. 19.2 — достаточно подключить светоизлучающий элемент не непосредственно к выводу контроллера, а через ключевой транзистор. Коллекторную цепь его, в которую и будет включена лампочка или светодиод, при этом нужно питать от источника пульсирующего напряжения с частотой сети (от отдельной обмотки трансформатора через один

диод, без сглаживания). А частоту на выводе МК, управляющую открыванием транзистора, подогнать примерно под 50 Гц (т. е. длительность задержки должна составить примерно 0,01 с, для чего в счетчики придется записать число около 8000, или, в шестнадцатеричной системе, 1F40h — как видите, здесь можно сократить число регистров задержки до двух). Частоты станут складываться прямо на транзисторе — лампочка будет гореть только тогда, когда и на коллекторе, и на его базе одновременно присутствует высокий уровень напряжения. Чтобы сделать эффект более заметным, можно поиграть со значением скважности, т. е. ввести разные задержки для красного и зеленого состояний. Поскольку у нас уже имеется два управляемых в противофазе вывода, можно добиться переливания из одного цвета в другой, а если усложнить программу, подключив еще несколько выводов МК, то можно получить весьма сложные и красивые эффекты. И, что приятно, экспериментируя с такой схемой, почти не придется браться за паяльник.

Применение прерываний

Для того чтобы сделать все то же самое, но «по-настоящему», придется воспользоваться таймером, а значит — прерываниями. Заметим, что в большинстве проектов Arduino прерывания в явном виде не используются, что относится к одному из главных недостатков этой платформы. Наличие аппаратных прерываний — одно из главных преимуществ современных контроллеров, и если в Arduino мы этим подходом часто жертвуем ради простоты, то в обычном программировании такое недопустимо.

Если помните, я говорил, что при возникновении прерывания процессор обращается по некоторому фиксированному адресу. Для каждой модели МК количество и типы прерываний различаются, потому эти адреса фиксированы для каждой модели процессора. Это первые адреса памяти программ, начиная с адреса 0:0 и дальше, — столько адресов, сколько имеется прерываний.

В МК ATtiny2313 имеется 19 прерываний (в его классическом аналоге было всего 11), соответственно, они занимают первые 19 адресов памяти программ. Напомним: поскольку коды команд двухбайтовые, то память программ организована по 16-битовым словам, поэтому адреса в памяти программ означают адреса слов, а байтовый адрес будет в два раза больше (т. е. не 0, 1, 2,..., а 0, 2, 4,...). Это вас ни в коей мере не должно волновать, потому что абсолютные адреса вам отсчитывать не придется, — достаточно правильно оформить первые 19 строк кода, после секции всех определений. Поэтому использующая прерывания программа для ATtiny2313 всегда должна начинаться с последовательности команд, представляющих векторы прерываний, по следующему образцу:

```
.include "tn2313def.inc"
< секция определений>
;========= interrupts ==============
rjmp RESET ; Reset Handler
rjmp EXT_INT0 ; External Interrupt0 Handler
rjmp EXT_INT1 ; External Interrupt1 Handler
rjmp TIM1_CAPT ; Timer 1 Capture Handler
rjmp TIM1_COMPA ; Timer 1 CompareA Handler
rjmp TIM1_OVF ; Timer 1 Overflow Handler
```

```
rjmp TIM0_OVF ; Timer 0 Overflow Handler
rjmp USART0_RXC ; USART0 RX Complete Handler
rjmp USART0_DRE ; USART0,UDR Empty Handler
rjmp USART0_TXC ; USART0 TX Complete Handler
rjmp ANA_COMP ; Analog Comparator Handler
rjmp PCINT ; Pin Change Interrupt
rjmp TIMER1_COMPB ; Timer 1 Compare B Handler
rjmp TIMER0_COMPA ; Timer 0 Compare A Handler
rjmp TIMER0_COMPB ; Timer 0 Compare B Handler
rjmp USI_START ; USI Start Handler
rjmp USI_OVERFLOW ; USI Overflow Handler
rjmp EE_READY ; EEPROM Ready Handler
rjmp WDT_OVERFLOW ; Watchdog Overflow Handler
```

Подробности

Если сравнить таблицы прерываний «классического» AT90S2313 и ATtiny2313, то окажется, что первые 11 векторов у них полностью совпадают. Отсутствующие в «классическом» аналоге остальные 8 векторов можно просто проигнорировать: если соответствующие прерывания не задействованы, то к ним никогда не произойдет обращения. По этой причине программы, написанные для AT90S2313, почти без оговорок совместимы с ATtiny2313 (не требуется даже заменять файл определений констант *2313def.inc*). В дальнейшем мы будем без пояснений употреблять программы для обеих версий этого МК.

Здесь `rjmp` — знакомая нам команда безусловного перехода, а `RESET`, `EXT_INT0` и т. п. — метки в тексте программы, откуда начинается текст процедуры (подпрограммы) обработки прерывания. Метки могут быть обозначены и по-иному, тут полный произвол. Сам переход осуществляется автоматически, если в процессе выполнения программы возникнут условия для возникновения соответствующего прерывания, для чего и его отдельно, и прерывания вообще надо еще разрешить. Все прерывания и особенности их вызова мы рассматривать, конечно, не будем, а с некоторыми из них познакомимся далее на практике.

Сначала заметим, что подобным образом должна выглядеть программа, если вы используете все 19 прерываний, чего на самом деле, конечно, не бывает. Но каждой записи `rjmp <метка>` должно соответствовать наличие метки далее в тексте, иначе ассемблер укажет на ошибку (посылать по неизвестному адресу нехорошо!). Поэтому если некоторые прерывания не используются, то, вообще говоря, можно вместо безусловных переходов наставить в соответствующих строках команд `nop` (no operation — ее код равен просто нулям во всех разрядах), однако на практике вместо них чаще ставят команду `reti`, которая означает возврат из процедуры прерывания к выполнению основной программы, если вдруг оно возникнет.

Заметки на полях

На самом деле в общем случае не имеет значения, какую именно команду в этих строках ставить — выполняться они никогда не будут, если соответствующие прерывания не активированы, нам только нужно занять память, чтобы команда `rjmp` используемого нами прерывания оказалась по нужному адресу, — например, можно поста-

вить команду `rjmp $0000`. Есть и другие, более корректные способы размещения команд по конкретным адресам памяти (с помощью директивы `.org`), но не будем усложнять. Кстати, надо учесть, что данный способ относится, строго говоря, лишь к МК с памятью программ не более 8 Кбайт, уже для ATmega16 команды будут другими: вместо 2-байтовой `rjmp` там применяется 4-байтовая `jmp` (а вместо команды вызова подпрограммы `rcall` — `call`). Это правило действует и для сходных по построению контроллеров из одного ряда, отличающихся только количеством памяти: например, для ATmega48/88 с памятью 4 и 8 Кбайт в векторе прерываний подставляется `rjmp` (а инструкции `jmp` и `call` вообще для них не действительны), а для ATmega168/328 — `jmp` (в фирменном описании примеры приведены отдельно для всех случаев).

Особое место занимает вектор, расположенный по самому первому, нулевому адресу, у нас он именуется `RESET`. Вообще говоря, вектор по нулевому адресу — это даже не совсем прерывание, а так называемый *вектор сброса* — мы ведь неоднократно говорили, что МК начинает выполнение программы с нулевого адреса. Программа из предыдущего раздела с него прямо начиналась (т. е. при включении питания сразу выполнялся первый оператор, потом второй и т. д.), но если задействованы прерывания, мы не можем так поступить. Поэтому в самом первой строке программы обязательно должен стоять указатель на процедуру, которая осуществляется в самом начале работы, и обычно так и называется: `RESET`. Эта процедура должна начинаться со следующих обязательных строк:

```
RESET: ;Старт главной программы
 ldi r16,low(RAMEND) ;только для 2313
 out SPL,r16 ;установка указателя стека
 .
 <начальные установки и разрешение необходимых прерываний>
   .   .
 sei ;общее разрешение прерываний
```

Если указатель стека не установить, то прерывания не заработают. Установка указателя стека для ряда моделей Tiny (в которых отсутствует SRAM) не требуется, а вот для других, в том числе и всех Mega, где количество SRAM превышает 256 байтов, эта загрузка будет протекать иначе, т. к. константа `RAMEND` там размером больше байта:

```
RESET: ;для моделей с SRAM более 256 байт
 ldi  temp,low(RAMEND) ;загрузка указателя стека
 out  SPL,temp
 ldi  temp,high(RAMEND) ;загрузка указателя стека
 out  SPH,temp
```

Подробности

Как мы уже отмечали в *главе 18*, *стек* — область памяти, куда будут записываться адреса точек возврата при вызове подпрограмм по команде `rcall` или перехода к обработчикам прерываний. По окончании процедуры обработки прерывания (по команде `reti`) или обычной подпрограммы (по команде `ret`) этот адрес считывается в программный счетчик, и выполнение основной программы продолжается. Если во время выполнения обработчика данного прерывания происходит другое прерывание с боль-

шим приоритетом (это нужно специально разрешать, по умолчанию в AVR любое прерывание будет ожидать окончания обработки предыдущего), то в стек записывается также и этот текущий адрес команды, таким образом ошибиться при последовательном возврате невозможно. Стек устроен по принципу «последним вошел — первым вышел», т. е. извлекается всегда последнее записанное туда значение. Программист может использовать стек и для своих целей (командами `push` и `pop` — например, для сохранения текущего значения рабочих регистров), но неопытным программистам лучше активно этим способом не пользоваться — вероятность допустить труднообнаруживаемую ошибку значительно возрастает. Тем более, что в AVR и без того достаточно РОН (в отличие, например, от x51, где без стека обойтись практически невозможно), а при необходимости можно еще и хранить текущие значения в SRAM.

Прерывание таймера по переполнению

С учетом всего сказанного напишем программу, переключающую светодиод. В данном случае она будет это делать по событию переполнения таймера-счетчика Timer 1 (вектор у нас обозначен: `TIM1_OVF`). Так как счетчик 16-разрядный, то событие переполнения будет возникать при каждом 65536-м импульсе входной частоты. Если мы зададим коэффициент деления тактовой частоты на входе Timer 1 равным 64, то при 4 МГц частоты генератора мы получим примерно 1 Гц: 4000000/64/65536 = 0,953674 Гц.

Это не совсем то, что нам требуется, и к тому же частота неточно равна одному герцу. Для того чтобы светодиод переключался точно раз в полсекунды (т. е. период его был равен секунде), программу придется немного усложнить, загружая каждый раз в счетные регистры определенное значение, которое рассчитывается просто: если период одного тактового импульса таймера равен 16 мкс (частота 4 000 000/64), то для получения 500 000 микросекунд надо отсчитать таких импульсов 31 250. Так как счетчик суммирующий, а прерывание возникает при достижении числа 65 536, то нужно предварительно загружать в него необходимо число 65 536 − 31 250 = 34 286.

Это не единственный способ, но наиболее универсальный, годящийся для всех 16-разрядных таймеров. Кстати, именно таким способом реализован отсчет времени в Arduino (см. *главу 20*). Иной способ — использовать прерывание по достижению определенного числа, загруженного в регистр сравнения A или B. Как это делается, мы увидим далее в этой главе. Для того чтобы осуществить само переключение из красного в зеленый, нам придется поступить как раньше, т. е. по каждому событию переполнения перебрасывать два бита в регистре `PortD`.

Полностью программа тогда будет выглядеть так:

```
;программа мигания светодиода
;процессор Tiny2313, частота 4 МГц
.include "tn8535def.inc"
.def temp = r16 ;рабочий регистр
; ============== interrupts ==============
rjmp RESET ;Reset Handler
reti ;EXT_INT0 ; External Interrupt0 Handler
```

```
reti ;EXT_INT1 ; External Interrupt1 Handler
reti ;TIM1_CAPT ; Timer 1 Capture Handler
reti ;TIM1_COMPA ; Timer 1 CompareA Handler
rjmp TIM1_OVF ; Timer 1 Overflow Handler
reti ;TIM0_OVF ; Timer 0 Overflow Handler
reti ;USART0_RXC ; USART0 RX Complete Handler
reti ;USART0_DRE ; USART0,UDR Empty Handler
reti ;USART0_TXC ; USART0 TX Complete Handler
reti ;ANA_COMP ; Analog Comparator Handler
reti ;PCINT ; Pin Change Interrupt
reti ;TIMER1_COMPB ; Timer 1 Compare B Handler
reti ;TIMER0_COMPA ; Timer 0 Compare A Handler
reti ;TIMER0_COMPB ; Timer 0 Compare B Handler
reti ;USI_START ; USI Start Handler
reti ;USI_OVERFLOW ; USI Overflow Handler
reti ;EE_READY ; EEPROM Ready Handler
reti ;WDT_OVERFLOW ; Watchdog Overflow Handler
; ============== code ==============
TIM1_OVF: ;процедура обработки прерывания по переполнению таймера
;сразу перезагружаем таймер:
 ldi temp,high(34286)
 out TCNT1H,temp
 ldi r16,low(34286)
 out TCNT1L,temp
;начало мигания
 in temp,PortD ;загружаем в temp состояние PortD
 sbrc temp,5 ;если PD5 равен 0, след. команду пропускаем
 rjmp set_Red ;переход на установку красного
 sbi PortD,5 ;PD5=1, устанавливаем зеленый
 cbi PortD,6 ;PD6=0, сбрасываем красный
 rjmp mig_end ;переход на продолжение программы
set_Red: ;метка установка красного
 cbi PortD,5 ;PD5=0, сбрасываем зеленый
 sbi PortD,6 ;PD6=1, устанавливаем красный
reti ;выход из прерывания
RESET: ;начальная процедура после сброса
 ldi temp,low(RAMEND) ;загрузка указателя стека
 out SPL,temp
 ldi temp,0b01100000 ;устанавливаем бит 5 и 6
 out DDRD,temp ;выводим это значение в регистр направления порта D
 ldi temp,0b00000011 ;устанавливаем temp
 out TCCR1B,temp ;запускаем Timer 1 с делителем 1/64
 ldi temp,high(34286)
 out TCNT1H,temp
 ldi r16,low(34286)
 out TCNT1L,temp
 ldi temp,(1<<TOIE1) ;устанавливаем бит TOIE1
 out TIMSK,temp ;разрешаем прерывание Timer 1 по переполнению
```

```
sei ;общее разрешение прерываний
Cykle:
rjmp Cykle ;зацикливаем процессор
```

Я не буду комментировать подробно каждый оператор, т. к. это заняло бы слишком много места. После выполнения всех команд начальной установки МК зацикливается, но бесконечный цикл будет прерываться возникновением прерывания — здесь все аналогично операционной системе Windows, которая также представляет собой бесконечный цикл ожидания событий. Как вы узнаете из последующих глав, в Arduino такой цикл — одно из главных составляющих любой программы, как раз потому что прерывания там почти не используются. Внутрь бесконечного цикла здесь можно поставить знакомую команду `sleep`, без дополнительных настроек режима энергопотребления она будет экономить около 30% питания. А вот сэкономить еще больше просто так не получится, поскольку придется останавливать процессорное ядро, и таймер перестанет работать.

> ***Заметки на полях***
>
> Кстати, а как остановить запущенный таймер, если это потребуется? Очень просто: если обнулить регистр `TCCR1B` (тот, в котором задается коэффициент деления тактовой частоты), то таймер остановится. Чтобы запустить его опять с коэффициентом 1/64, нужно снова записать в этот регистр значение `0b00000011`.

Обратите внимание, что оператор `reti` (окончание обработки прерывания) при обработке прерывания таймера встречается дважды — это вполне нормальный прием, когда подпрограмма разветвляется. Можно, конечно, и пометить последний оператор `reti` меткой, и тогда текст процедуры стал бы неотличим от первого варианта без прерываний, но так будет корректнее.

Обратите также внимание на форму записи `ldi temp,(1<<TOIE1)`. Поскольку бит, обозначаемый как `TOIE1`, в регистре `TIMSK` имеет номер 7, то эта запись эквивалентна записи `ldi temp,0b10000000` — можно писать и так, и так, и еще кучей разных способов. Например, для запуска таймера с коэффициентом 1/64 требуется, как видно из текста программы, установить младшие два бита регистра `TCCR1B`. Здесь мы устанавливаем их в `temp` напрямую, но поскольку эти биты называются `CS11` и `CS10`, то можно записать так:

```
ldi temp,(1<<CS11)|(1<<CS10)
```

или даже так:

```
ldi temp,(3<<CS10)
```

Подробно этот способ записи приведен в описании AVR-ассемблера на сайте Atmel[1].

---

[1] Фирменное описание доступно по адресу: **http://www.atmel.com/ru/ru/images/doc1022.pdf**. Краткий перевод на русский можно найти, например, здесь: **http://microcon.euro.ru/app/books/Asm_AVR_rus.pdf**. Оба документа, к сожалению, устарели (они относятся к предыдущей версии avrasm32), хотя большая часть изложенных там сведений вполне пригодна и для современной версии. Подробное (хотя и не без досадных неточностей) описание современной версии AVR-ассемблера на русском языке можно найти в [21].

Подробности

В этой программе есть один тонкий момент, связанный с загрузкой счетных регистров таймера. При чтении и записи 16-разрядных регистров Timer 1 их содержимое может измениться в промежутке между чтением или записью отдельных 8-разрядных «половинок» (ведь, например, в данном случае таймер продолжает считать, пока идет обработка прерывания). Потому в 16-разрядных таймерах AVR предусмотрен специальный механизм чтения и записи таких регистров. При записи первым загружается значение старшего байта, которое автоматически помещается в некий (недоступный для программиста) буферный регистр. Затем, когда поступает команда на запись младшего байта, оба значения объединяются, и запись производится одновременно в обе «половинки» 16-разрядного регистра. Наоборот, при чтении первым должен быть прочитан младший байт, при этом значение старшего автоматически фиксируется помещением в тот же буферный регистр, и при следующей операции чтения старшего байта его значение извлекается оттуда. Таким образом, и при чтении значения оба байта соответствуют одному и тому же моменту времени.

Повторим: для того чтобы манипуляции со счетными регистрами были успешными, при чтении необходимо сначала прочесть младший байт `TCNTxL`, потом старший `TCNTxH`, при записи сначала записать старший байт `TCNTxH`, потом младший `TCNTxL`. Аналогичное правило действует для всех 16-разрядных регистров Timer 1, которые мы будем рассматривать далее, за исключением регистров управления `TCCR1A` и `TCCR1B`, которые по сути есть два раздельных регистра, а не один.

Напомним, что если вам попадется старый «классический» AT90S2313, то приведенную здесь программу можно использовать для него без изменений.

Прерывание таймера по сравнению

Способ отсчета времени с помощью прерывания таймера по сравнению более понятен и удобен, чем с предзагрузкой значений в счетный регистр, — хотя бы потому, что число, с которым сравнивается содержимое счетных регистров, можно загружать только один раз. Если потом запустить таймер, то больше об этом можно не думать — все будет происходить автоматически. Поскольку в Tiny2313 и большинстве моделей Mega (если не во всех) все таймеры, в том числе и 8-разрядные, имеют такой режим (в «классических» его имел только 16-разрядный Timer 1), то применение его тем более целесообразно.

Подробности

Прерывание по сравнению происходит в момент, когда счетчик досчитывает до наперед заданного значения, хранящегося в специальном регистре сравнения. Есть одна тонкость, связанная с этим режимом. Входной тактовый сигнал счетчика обычно делится в соответствии с заданным коэффициентом деления (в наших примерах это 1/64). Поэтому в нашем случае каждое состояние счетчика остается неизменным в течение 64 тактов процессора. Так в какой именно момент возникает прерывание — сразу, как только счетчик досчитал до заданного значения, или в момент, когда это заданное значение в счетчике должно уже смениться следующим? Если предположить, что в AVR все устроено, как на рис. 16.13, *в*, то имеет место первый случай — счетчик начинает новый отсчет с первого системного такта сразу после совпадения величин. Тогда состояния счетчика получаются неравноправными: все они длятся по 64 системных такта (в соответствии с выставленным коэффициентом предделителя), кроме последнего, который длится один системный такт независимо от коэффициента

деления. Так это было устроено в «классической» серии AVR. А вот для счетчиков семейств Mega и Tiny все иначе: там событие совпадения наступает по последнему такту при совпадении, в момент, когда состояние счетчика должно уже смениться. Поэтому если вы зададите в регистрах сравнения, к примеру, число 2, то при коэффициенте деления 1/64 Timer 1 в МК AT90S2313 отсчитает до возникновения прерывания 129 системных тактов (или примерно 2 такта входной частоты счетчика), а в МК ATtiny2313 — 192 системных такта (ровно 3 такта входной частоты). Таким образом, в первом случае коэффициент деления входной частоты таймера в режиме сравнения равен установленному числу *N* плюс один такт системного генератора, а во втором — числу *N* + 1.

Причем в этом режиме можно упростить программу предельно — специальный вывод контроллера (при прерывании с применением регистра сравнения A это OC1A — вывод 15 для 2313) можно включить в режим автоматического переключения в момент совпадения, без дополнительного программного управления. Если больше ничего в момент совпадения делать не требуется, не нужно даже будет включать прерывание — переключение вывода происходит аппаратно. Нам здесь это не подходит, т. к. светодиод двухцветный, и все равно необходимо переключать две линии одновременно, а вот в других случаях эту возможность можно использовать.

Для того чтобы установить режим сравнения, нужно вместо прерывания по переполнению разрешить прерывание по сравнению A (бит OCIE1A в регистре TIMSK), а также установить значение в регистрах сравнения (OCR1AH и OCR1AL), с которым будет сравниваться содержимое счетчика. Нетрудно догадаться, что для цикла счета в полсекунды оно равно вычисленному нами ранее значению 31 250. При записи в эти регистры нужно помнить то, что было сказано ранее про обращение с 16-разрядными регистрами в таймерах.

Есть в этом деле и еще один нюанс. Что будет происходить с таймером после того, как значения в счетных регистрах и регистрах сравнения станут одинаковыми (кроме того, что произойдет прерывание)? Ясно, что тут могут быть варианты: таймер может продолжить счет, обнулиться, установиться в какое-то наперед заданное значение и т. п. Это поведение настраивается — для выбора режима обнуления (чтобы после сравнения таймер пришел бы в исходное состояние) следует установить бит WGM12 (в «классической» версии МК он назывался CTC1) — бит номер 3 в регистре TCCR1B.

Программа с учетом всего сказанного будет выглядеть таким образом:

```
;программа мигания светодиода
;процессор Tiny2313, частота 4 МГц
.include "tn8535def.inc"
.def temp = r16 ;рабочий регистр
;========= interrupts ==============
rjmp RESET ;Reset Handler
reti ;EXT_INT0 ; External Interrupt0 Handler
reti ;EXT_INT1 ; External Interrupt1 Handler
reti ;TIM1_CAPT ; Timer 1 Capture Handler
rjmp ;TIM1_COMPA ; Timer 1 CompareA Handler
```

```
reti TIM1_OVF ; Timer 1 Overflow Handler
reti ;TIM0_OVF ; Timer 0 Overflow Handler
reti ;USART0_RXC ; USART0 RX Complete Handler
reti ;USART0_DRE ; USART0,UDR Empty Handler
reti ;USART0_TXC ; USART0 TX Complete Handler
reti ;ANA_COMP ; Analog Comparator Handler
reti ;PCINT ; Pin Change Interrupt
reti ;TIMER1_COMPB ; Timer 1 Compare B Handler
reti ;TIMER0_COMPA ; Timer 0 Compare A Handler
reti ;TIMER0_COMPB ; Timer 0 Compare B Handler
reti ;USI_START ; USI Start Handler
reti ;USI_OVERFLOW ; USI Overflow Handler
reti ;EE_READY ; EEPROM Ready Handler
reti ;WDT_OVERFLOW ; Watchdog Overflow Handler
;========= code ==============
TIM1_COMPA: ;процедура обработки прерывания по сравнению A
;начало мигания
 in temp,PortD ;загружаем в temp состояние PortD
 sbrc temp,5 ;если PD5 равен 0, след. команду пропускаем
 rjmp set_Red ;переход на установку красного
 sbi PortD,5 ;PD5=1, устанавливаем зеленый
 cbi PortD,6 ;PD6=0, сбрасываем красный
 rjmp mig_end ;переход на продолжение программы
set_Red: ;метка установки красного
 cbi PortD,5 ;PD5=0, сбрасываем зеленый
 sbi PortD,6 ;PD6=1, устанавливаем красный
reti ;выход из прерывания
RESET: ;процедура после сброса
 ldi temp,low(RAMEND) ;загрузка указателя стека
 out SPL,temp
 ldi temp,0b01100000 ;устанавливаем биты 5 и 6
 out DDRD,temp ;выводим это значение в регистр направления порта D
 ldi temp,high(31250) ;значение регистра сравнения для 0,5 сек
 out OCR1AH,temp ;старший байт
 ldi temp,low(31250)
 out OCR1AL,temp ;младший байт
 ldi temp,0b00001011
 out TCCR1B,temp ;запуск таймера с делителем 1/64
        ;и установка бита 3 «очищать счетчик при совпадении»
 ldi temp,(1<<OCIE1A) ;разрешение прерывания по сравнению A
 out TIMSK,temp
 sei ;общее разрешение прерываний
Cykle:
 rjmp Cykle ;зацикливаем процессор
```

Естественно, значение, загружаемое в регистры сравнения `OCR1AH:OCR1AL`, необязательно должно быть равно в точности 31 250. Это дает удобный способ для точной

подстройки интервала времени, который может иметь определенный разброс из-за неточностей используемого кварца.

Арифметика многоразрядных чисел на ассемблере

Этот раздел включен в новое издание книги потому, что многоразрядные числа в 8-разрядном контроллере требуют особого обращения, и далеко не всегда очевидно, как именно с ними манипулировать. Этой проблемы в Arduino и вообще для языков высокого уровня не существует вовсе: там достаточно задать тип данных и соблюдать правила обращения с ним, чтобы все происходило автоматически. Здесь же приходится все делать, как говорится, ручками. Примеры практического применения мы не приводим, но если вы поняли общие принципы программирования на ассемблере, использование приведенных процедур не должно представить трудностей.

Сложение и вычитание больших чисел в МК выполняется очень просто. Корректная операция сложения двух 16-разрядных чисел будет занимать две команды:

```
add RL1,RL2
adc RH1,RH2
```

Здесь переменные `RL1` и `RL2` содержат младшие байты слагаемых, а `RH1` и `RH2` — старшие. Если при первой операции результат превысит 255, то перенос запишется во все тот же флаг переноса C и учтется при второй операции. Общий результат окажется в паре `RH1:RL1`. Совершенно аналогично выглядят операция вычитания и операции с большим числом разрядов.

А вот с умножением и делением несколько сложнее. Выполнение типовых операций на AVR для чисел с различной разрядностью, вообще говоря, приводится в фирменных руководствах по применению: «аппнотах» (Application Notes, в данном случае номер 200[1]). Но эти процедуры для наших целей все равно придется творчески переработать. Поэтому мы не будем на них останавливаться, а сразу воспользуемся тем обстоятельством, что для контроллеров семейства Mega определены аппаратные операции умножения. Тогда алгоритм сильно упрощается и легко модифицируется для любого размера операндов и результата. Вот так выглядит алгоритм для перемножения двух 16-разрядных сомножителей с получением 24-разрядного результата (в названиях исходных переменных отражен факт основного назначения такой процедуры — для умножения неких данных на некий коэффициент[2]):

```
.def dataL = r4 ;младший байт данных
.def dataH = r5 ;старший байт данных
```

---

[1] Полный комплект Application Notes для AVR можно найти по адресу: **http://www.atmel.com/products/microcontrollers/avr/default.aspx**, вкладка **Documents**.

[2] Сокращения LSB и MSB означают least (most) significant bit — младший (старший) значащий разряд, по-русски МЗР и СЗР соответственно.

```
.def KoeffL = r2 ;младший байт коэффициента
.def koeffH = r3 ;старший байт коэффициента
.def temp = r16 ;байт 0 результата (LSB - младший разряд)
.def temp2 = r17 ;байт 1 результата
.def temp3 = r18 ;байт 2 результата (MSB - старший разряд)
. . . . .
;*****************************************************************
;умножение двух 16-разрядных величин, только для Mega
;исходные величины dataH:dataL и KoeffH:KoeffL
;результат 3 байта temp2:temp1:temp
;*****************************************************************
Mu16x16:
clr temp2 ;очистить старший
mul dataL,KoeffL ;умножаем младшие
mov temp,r0 ;в r0 младший результата операции mul
mov temp1,r1 ;в r01 старший результата операции mul
mul dataH,KoeffL ;умножаем старший на младший
add temp1,r0 ;в r0 младший результата операции mul
adc temp2,r1 ;в r01 старший результата операции mul
mul dataL,KoeffH ;умножаем младший на старший
add temp1,r0 ;в r0 младший результата операции mul
adc temp2,r1 ;в r01 старший результата операции mul
mul dataH,KoeffH ;умножаем старший на старший
add temp2,r0 ;4-й разряд нам тут не требуется, но он - в r01
ret
;*****************************************************************
```

Как видите, если нужно получить полный 32-разрядный диапазон, просто добавьте еще один регистр для старшего разряда (`temp3`, к примеру) и одну строку кода перед командой `ret`:

```
adc temp3,r01
```

Естественно, можно просто обозвать `r01` через `temp3`, тогда и добавлять ничего не придется.

Деление — значительно более громоздкая процедура, чем умножение, оно требует больше регистров и занимает больше времени. Операции деления двух чисел (и 8- и 16-разрядных) приведены в той же «аппноте» 200, но они не всегда удобны на практике: часто нам приходится делить результат какой-то ранее проведенной операции умножения или сложения, а он нередко выходит за пределы двух байтов.

Пусть нам потребуется вычислять среднее значение для уточнения результата измерения по сумме отдельных измерений. Если даже само измерение укладывается в 16 разрядов, то сумма нескольких таких результатов уже может занимать три байта. В то же время делитель — число измерений — будет относительно небольшим и укладывается в один байт. Но мы не будем здесь заниматься построением «настоящих» процедур деления (интересующихся отсылаю к моей книге [21], а также к пособию на сайте **cxem.net** по адресу **http://cxem.net/mc/book30.php**). Многие

подобные задачи на деление удается решить значительно более простым и менее громоздким методом, если заранее подгадать так, чтобы делитель оказался кратным степени 2. Тогда все деление сводится, как мы знаем, к сдвигу разрядов вправо столько раз, какова степень двойки.

Для примера предположим, что мы некую величину измерили 64 раза и хотим узнать среднее. Пусть сумма укладывается в 2 байта, тогда вся процедура деления будет такой:

```
;деление на 64
  clr count_data ;счетчик до 6
div64L:
  lsr dataH ;сдвинули старший
  ror dataL ;сдвинули младший с переносом
  inc count_data
  cpi count_data,6
  brne div64L
```

Не правда ли, гораздо изящнее и понятнее обычных делений столбиком? Попробуем от радости решить задачку, которая на первый взгляд требует по крайней мере знания высшей алгебры: умножить некое число на дробный коэффициент (вещественное число с «плавающей запятой»). Теоретически для этого требуется представить исходные числа в виде мантисса-порядок, сложить порядки и перемножить мантиссы. Насколько громоздкие процедуры при этом получаются, можно убедиться, если скачать библиотеку для работы с реальными числами, которую можно найти по этому адресу: **http://www.gaw.ru/data/soft/instr/avr/float.zip**. Нам же неохота возиться с этим представлением, т. к. мы не проектируем универсальный компьютер, и в подавляющем большинстве реальных задач все конечные результаты у нас есть целые числа: запятую при выводе на индикаторы, как мы помним, мы все равно устанавливаем вручную.

На самом деле эта задача решается очень просто, если ее свести к последовательному умножению и делению целых чисел, представив реальное число в виде целой дроби с оговоренной точностью. В десятичной форме это выглядит так: представим число 0,48576 как 48 576/100 000. И если нам требуется на такой коэффициент умножить, к примеру, число 976, то можно действовать, не выходя за рамки диапазона целых чисел: сначала умножить 976 на 48 576 (получится заведомо целое число 47 410 176), а потом поделить результат на 10^5, чисто механически перенеся запятую на пять разрядов. Получится 474,10176 или, если отбросить дробную часть, 474. Большая точность нам и не требуется, т. к. и исходное число было трехразрядным.

Улавливаете, к чему я клоню? Наше ноу-хау будет состоять в том, что мы для того, чтобы «вогнать» дробное число в целый диапазон в микроконтроллере, будем использовать не десятичную дробь, а двоичную — деление тогда сведется к той же самой механической процедуре сдвига разрядов вправо, аналогичной переносу запятой в десятичном виде.

Итак, чтобы умножить 976 на коэффициент 0,48576, следует сначала последний вручную умножить, например, на 2^{16} (65 536), и тем самым получить числитель со-

ответствующей двоичной дроби (у которой знаменатель равен 65 536) — он будет равен 31 834,76736, или, с округлением до целого, 31 835. Такой точности хватит, если исходные числа не выходят, как у нас, за пределы трех-четырех десятичных разрядов. Теперь мы в контроллере должны умножить исходную величину 976 на константу 31 835 (см. процедуру перемножения ранее), и полученное число 31 070 960 (оно оказывается 4-байтовым — $01DA1AF0, потому нашу процедуру `Mu16x16` придется чуть модифицировать, как сказано при ее описании) сдвинуть на 16 разрядов вправо:

```
;в ddHH:ddH:ddM:ddL число $01DA1AF0,
;его надо сдвинуть на 16 разрядов
  clr cnt
div16L: ;деление на 65536
  lsr ddHH ;сдвинули старший
  ror ddH ;сдвинули 3-й
  ror ddM ;сдвинули 2-й
  ror ddL ;сдвинули младший
  inc cnt
  cpi cnt,16
  brne div16L ;сдвинули-поделили на 2 в 16
```

В результате, как вы можете легко проверить, старшие байты окажутся нулевыми, а в `ddM:ddL` окажется число 474 — тот же самый результат. Но и это еще не все — такая процедура приведена скорее для иллюстрации общего принципа. Ее можно еще больше упростить, если обратить внимание на то, что сдвиг на восемь разрядов есть просто перенос значения одного байта в соседний (в старший, если сдвиг влево, и в младший — если вправо). Итого получится, что для сдвига на 16 разрядов вправо нам надо всего-навсего отбросить два младших байта и взять из исходного числа два старших `ddHH:ddH` — это и будет результат. Проверьте — $01DA и есть 474. Никаких других действий вообще не требуется!

Если степень знаменателя дроби, как в данном случае, кратна 8, то действительно никакого деления, даже в виде сдвига, не требуется, но чаще всего это не так. Однако даже в этом случае приведенный принцип может помочь: например, при делении на 2^{15} вместо пятнадцатикратного сдвига вправо результат можно сдвинуть на один разряд влево (умножив число на два), а потом уже выделить из него старшие два байта. Вот такая специальная арифметика в МК.

Операции с числами в формате BCD

О двоично-десятичных числах или числах в формате BCD было подробно рассказано в *главе 14*. Как ясно из сказанного там, упакованные BCD-числа удобны для хранения данных, но неудобны для отображения и для выполнения арифметических операций с ними. Поэтому перед отображением упакованные BCD-числа распаковывают, перемещая старший разряд в отдельный байт и заменяя в обоих байтах старшие полубайты нулями. А перед проведением арифметических действий их переводят в обычный формат, после чего опять преобразуют в упакованный фор-

мат BCD. Вот этими операциями мы и займемся. Следует отметить, что в системе команд 8051 (а также и знаменитого 8086) есть специальные команды десятичной коррекции, но в AVR их нет, и придется изобретать им замену самостоятельно.

В области двоично-десятичных преобразований (BCD-преобразований) есть три основные задачи:

- преобразование двоичного/шестнадцатеричного числа в упакованный BCD-формат;
- распаковка упакованного BCD-формата для непосредственного представления десятичных чисел с целью их вывода на дисплей;
- обратное преобразование упакованного BCD-формата в двоичный/шестнадцатеричный с целью, например, произведения арифметических действий над ним.

Некоторые процедуры для преобразования в BCD-формат приведены в фирменной Application notes 204. Приведем здесь вариант такой процедуры, более экономичный в части использования регистров. Исходное hex-число содержится в регистре `temp`, распакованный результат — в `temp1:temp`. Процедура довольно короткая:

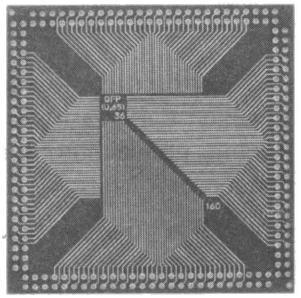
```
;преобразование 8-разрядного hex в неупакованный BCD
;вход hex= temp, выход BCD  temp1 - старший; temp - младший
;эта процедура работает только для исходного hex-числа от 0 до 99
bin2bcd8:
  clr temp1 ;очистить MSB
bBCD8_1:
  subi temp,10 ;input = input - 10
  brcs bBCD8_2 ;если установлен бит переноса, закончить
  inc temp1 ;увеличить MSB
  rjmp bBCD8_1 ;перейти на начало цикла
bBCD8_2:
  subi temp,-10  ;скомпенсировать лишнее вычитание
ret
```

Заодно приведем одно из решений обратной задачи — преобразования упакованного BCD в hex-число, после чего с ним можно производить арифметические действия. По сравнению с «фирменной» `BCD2bin8` эта процедура хоть и немного длиннее, но понятнее и более предсказуема по времени выполнения:

```
;на входе в temp упакованное BCD-значение
;на выходе в temp hex-значение
;temp1 - вспомогательный регистр для промежуточного хранения temp
;действительна только для семейства Mega
HEX_BCD:
  mov temp1,temp
  andi temp,0b11110000  ;распаковываем старшую тетраду
  swap temp ;старший в младшей тетраде
  mul temp,mult10 ;умножаем на 10, в r0 результат умножения
  mov temp,temp1 ;возвращаемся к исходному
  andi temp,0bQ0001111 ;младший
  add temp,r0 ;получили hex
ret
```

Преобразовывать BCD-числа, состоящие более чем из одного байта, обратно в hex-формат приходится крайне редко, зато задача прямого преобразования возникает на каждом шагу. Эта задача более громоздкая, но ее реализацию нет смысла разбирать подробно — она во всем аналогична рассмотренному случаю. Приведем только текст процедуры без дополнительных пояснений:

```
bin2BCD16: ;преобразование 16-разрядного hex в упакованный BCD
;input: hex value low=AregL hex value high = AregH
;output: BCD value digits 1 and 0 ResL
;BCD value digits 2 and 3(=0) ResH
       ldi    temp1,16       ;Init loop counter
       clr    ResH
       clr    ResL
       clr    ZH             ;clear ZH (not needed for AT90Sxx0x)
bBCDx_1:lsl   AregL          ;shift input value
       rol    AregH          ;through all bytes
       rol    ResL           ;
       rol    ResH
       dec    temp1          ;decrement loop counter
       brne   bBCDx_2        ;if counter not zero
       ret                   ;   return
bBCDx_2:ldi   r30,AtBCD2+1   ;Z points to result MSB + 1
bBCDx_3:
       ld     temp,-Z;get (Z) with pre-decrement
       subi   temp,-$03      ;add 0x03
       sbrc   temp,3 ;if bit 3 not clear
       st     Z,temp ;       store back
       ld     temp,Z ;get (Z)
       subi   temp,-$30      ;add 0x30
       sbrc   temp,7 ;if bit 7 not clear
       st     Z,temp ;       store back
       cpi    ZL,AtBCD0      ;done all three?
       brne   bBCDx_3        ;loop again if not
       rjmp   bBCDx_1
```



ГЛАВА 20

Основы Arduino

Контроллеры, среда и примеры программирования

— Но для путешествия в Лондон нужны деньги, — заметил Портос, — а у меня их нет.
— У меня тоже.
— И у меня.
— У меня они есть, — сказал д'Артаньян, вытаскивая из кармана свой клад и бросая его на стол.

А. Дюма. «Три мушкетера»

Возникновение платформы Arduino стало закономерным ответом индустрии на запрос со стороны пользователей электронных приборов, не желающих тратить кучу времени на поиск нужного (и, возможно, отсутствующего) устройства на рынке, а имеющих намерение сделать его своими руками, причем, по возможности, с наименьшей затратой сил, средств и времени. Развитие микроэлектроники в последние десятилетия подготовило все условия для решения такой задачи, тем самым переведя любительство в этой области на принципиально иной уровень.

Переворот, который совершила Arduino в области любительского конструирования электронной техники, можно сравнить в революцией в фотографии, наступившей с появлением цифровых камер. Если еще лет двадцать назад увлеченному радиолюбителю, как и фотографу, приходилось заводить дома целую лабораторию, то теперь на все про все достаточно одного настольного компьютера. Своим возникновением Arduino создала новую категорию любителей и целую отрасль промышленности, направленную на их обеспечение нужными комплектующими. Вы берете платы из коробки, доставленной курьером, соединяете их в нужном порядке, и готовый прибор работает, даже если вы в жизни ни разу не прикасались к паяльнику.

Но не следует думать, что таким способом можно овладеть всеми тонкостями ремесла. Как грамотному фотографу по-прежнему необходимо знание многих нюансов из области теории цвета и оптики (а необходимость освоения основ химии ему теперь заменили основы компьютерных наук), так и любителю Arduino, если он не собирается ограничиваться повторением чужих схем неизвестного качества, а хочет создавать и совершенствовать что-то свое, придется изучать контроллеры «изнутри». Именно поэтому я подчеркивал в *главе 19*, что если вы желаете овладеть микроэлектроникой по-настоящему, то начинать следует с программирования про-

стых конструкций на ассемблере, а не на языке С и, тем более, не в среде Arduino. Переход к языкам высокого уровня целесообразен тогда, когда вы понимаете, что именно происходит в контроллере, и в случае надобности можете управлять этим процессом.

Такое мое убеждение, однако, не исключает того факта, что в качестве элементарного введения в предмет Arduino подойдет очень неплохо. О недостатках этой платформы мы еще поговорим в самом конце книги, а в оставшихся ее главах покажем, как с минимальной затратой сил с помощью Arduino делать настоящие электронные приборы, которые будут работать лучше покупных, иметь больше функций и обойдутся при этом, как минимум, не дороже тех, что имеются на прилавках.

При этом ограниченный объем книги не позволяет мне остановиться на многих интересных темах: например, совсем несложно пристегнуть к Arduino модуль GPS и построить свой собственный навигатор, превратить Arduino в управляющий узел квадрокоптера и даже создать на его основе автономный Web-сервер, который будет передавать погоду в Интернет. По необходимости мы также оставим в стороне работу в Arduino со звуком и лишь вскользь коснемся одного из главных направлений применения этой платформы в области конструирования роботов. Хочу еще обратить ваше внимание на открытый проект Accessory Development Kit компании Google — он позволяет устройствам на Android обеспечивать двусторонний обмен данными с Arduino через USB или Bluetooth. Здесь же мы сосредоточимся на измерительной технике, различных датчиках, вопросах взаимодействия с компьютером и вывода информации на дисплеи различной конструкции, что даст хорошее и обстоятельное введение в платформу и позволит конструировать практически полезные вещи.

Многие из упоминаемых здесь и далее комплектующих можно приобрести в интернет-магазине «Амперка» (**http://amperka.ru**), сотрудники которого оказали автору неоценимую помощь в написании этого раздела книги. Советую также заглянуть в их вики-раздел, где собрано большое количество сведений о применении различных компонентов Arduino.

Что такое Arduino?

Платформа Arduino возникла в среде сотрудников учреждения Interaction Design Institute (что можно перевести, как «Институт конструирования взаимодействий»), расположенного в итальянском городке Ивреа, и получила свое почти толкиеновское название от имени реально существовавшего короля Ардуина, правившего этой местностью в начале прошлого тысячелетия. Arduino выросла из задачи научить студентов непрофильных специальностей создавать электронные устройства, причем быстро и, желательно, без опоры на углубленное изучение электроники, электротехники и программирования.

В конце концов группа, руководимая программистом Массимо Банци, создала универсальную аппаратную платформу на основе дешевых и доступных микрокон-

троллеров Atmel AVR и решила ее распространять на принципах Open Source. Такие свободные лицензии, как знаменитая GPL, разработанная применительно к софту, для «железа» напрямую не годится, потому создатели взяли за основу пакет лицензий Creative Commons для творческих продуктов. Лицензия Arduino запрещает использование этой торговой марки для каких-то сторонних продуктов, кроме расширений основного проекта. Это привело к тому, что от Arduino стали отпочковываться аналогичные проекты, совместимые с ним, но желающие иметь иные названия — например, такие, как Freeduino, Craftduino, Carduino и многие другие.

Сама компания Arduino LLC, основанная Массимо Банци в 2004 году в США, изначально лишь проектировала платы контроллеров Arduino и разрабатывала программную среду разработки Arduino IDE. Родственная итальянская компания Smart Projects SRL (в 2014 году переименована в Arduino SRL) производила оригинальные платы Arduino. К сожалению, как это часто бывает, в 2014 году между сооснователями платформы произошел раскол, завершившийся возникновением двух независимых ветвей развития и продаж под одной торговой маркой: одна на сайте **arduino.cc**, другая на **arduino.org**. Arduino SRL даже начала выпускать собственную среду разработки Arduino IDE (начав с номера версии 1.7, тогда как оригинал на тот момент имел номера, начинающиеся с 1.6).

Юридический спор за владение торговой маркой Arduino между соперничающими ветвями был завершен в середине 2017 года, когда обе компании слились в одной Arduino AG. Сайт **arduino.org** закрылся, его посетителей переадресовывают на **arduino.cc**. Еще примерно за год до этого на обоих сайтах появилась единая Arduino IDE, нумерация версий которой начиналась с 1.8. Причем, что интересно, в перечне предыдущих версий, которые также традиционно доступны для скачивания, версии с номерами 1.7 отсутствуют — они все равно не сыскали популярности, пользователи по всему миру продолжали пользоваться версиями 1.6 с сайта **arduino.cc**.

В мире насчитывается более двухсот дистрибьюторов продукции Arduino, включая довольно крупные торговые фирмы. Контроллеров Arduino создано уже около 15 версий, причем некоторые из последних — на 32-разрядных AVR или даже на ARM-процессорах. Популярные платы контроллеров стоят приблизительно от 10 (Arduino Mini) до 30 долларов (Arduino Uno), и любая из них может быть изготовлена самостоятельно — документация доступна всем желающим (кроме официального сайта **arduino.cc**, см. также русскоязычный сайт **arduino.ru** или его украинского собрата **arduino.ua**). Поэтому встречающиеся в интернет-магазинах китайские клоны основных плат Arduino в принципе ничуть не хуже оригинальных, поскольку схемотехника, программирование и главный узел — собственно контроллер от фирмы Atmel — у них одни и те же (хотя определенный риск в таком приобретении, конечно, имеется).

Бесплатно распространяется и среда программирования, основанная на адаптированной под непрофессионалов версии C/C++. При желании платы Arduino можно программировать и напрямую на низком уровне или из других сред программирования, т. е. так, как описано в предыдущих главах этой книги, — на многих платах

Arduino предусмотрен для этой цели ISP-разъем (или имеются контактные площадки для его установки). Кстати, Atmel добавила поддержку загрузочного модуля Arduino в свою фирменную AVR Studio. Появился даже такой пакет, как Visual Micro — расширение микрософтовской Visual Studio (если кто не знает — это такая универсальная среда программирования для Windows), ориентированное на Arduino.

В основе платформы лежат несколько типовых плат-модулей, в современной версии большей частью построенных на контроллере ATmega328. Этот контроллер имеет 32 килобайта памяти программ, чего достаточно для загрузки даже столь объемных загрузочных файлов, какие получаются при компилировании в среде Arduino IDE. Базовые модули Arduino в целом следуют структуре типового AVR, описанной в *главе 18*, но дополнительно содержат стабилизаторы питания, несколько светодиодов и других компонентов, и, главное — встроенный загрузчик с преобразователем USB/UART, позволяющим и программировать контроллер через последовательный порт, и организовать «общение» программы с компьютером.

Для этой цели в контроллер на платах Arduino заранее записывается программа-загрузчик. Если вы будете программировать Arduino напрямую, через обычный ISP-программатор, то загрузчик, естественно, окажется испорченным. Однако его всегда можно восстановить с помощью среды Arduino IDE и совместимого ISP-программатора (см. *главу 19*), потому любые эксперименты не приведут к фатальным последствиям. С другой стороны, на некоторых платах Arduino контроллер установлен на панельку, что позволяет применять плату совместно со средой программирования, как удобный программатор для МК AVR, которые потом можно устанавливать в другие схемы.

Основные платы Arduino

Ранее платы Arduino использовали контроллер ATmega168 (а еще ранее — самый первый представитель этой линейки ATmega8). И до сих пор на многих ресурсах в Сети можно встретить ссылки на принципиальные схемы и обозначения выводов плат, где показан именно ATmega168. Это не должно вас смущать, потому что контроллеры ATmega168 и ATmega328 устроены одинаково, и различаются лишь объемом памяти. Так что схемы с обозначениями выводов на основе любой из этих разновидностей AVR-контроллеров пригодны для использования.

Мы для начального изучения в основном воспользуемся одним из самых популярных модулей под названием **Arduino Uno** (рис. 20.1). Arduino Uno — основная (базовая) модель Arduino, на нее равняются все остальные. Плата очень хорошо приспособлена для макетирования совместно с беспаечными макетными платами и различными платами расширений — «шилдами» (shields). «Шилды», содержащие дополнительные компоненты или гнезда для их подключения, могут устанавливаться на плату Arduino Uno «в этажерку» друг над другом. Ее размер (54×69 мм, без учета разъемов, выступающих за пределы платы) и конфигурация контактов стали стандартом, на который равняется множество электронных изделий, включая даже далекие от собственно Arduino.

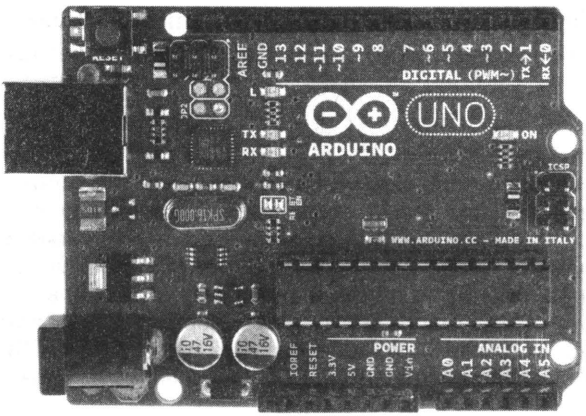


Рис. 20.1. Arduino Uno — плата одной из базовых моделей платформы

Гнезда на плате Uno совпадают по типоразмеру с гнездами на беспаечной макетной плате (см. рис. 3.3). Другие платы и внешние датчики подключаются к Arduino Uno с помощью либо специальных проводочков или плоских кабелей со штекерами — обычно через макетную плату для «размножения» контактов. Существуют разновидности плат Uno, на которых специально установлены дополнительные контактные штыри или гнезда таким образом, что можно подключать различную периферию, не пользуясь макетной платой в качестве разветвителя.

Рассмотрим кратко основные компоненты платы Arduino Uno. Контроллер ATmega328, как видно из рис. 20.1, здесь в корпусе DIP и установлен на панельку, что позволяет извлекать его и использовать отдельно в различных схемах. Таким путем Arduino Uno можно использовать в качестве программатора для этой микросхемы. Практически на всех платах Arduino имеется отдельный программирующий 6-контактный ISP-разъем, описанный в *главе 19* (на рис. 20.1 он расположен справа вверху от контроллера).

На другом торце платы Arduino Uno размещен, во-первых, основной коммуникационный USB-разъем для подключения к компьютеру. По традиции он самого большого размера из всех многочисленных разновидностей USB (такие разъемы устанавливают сейчас только на крупногабаритной технике, например, на принтерах или сканерах). Потому к нему придется приобретать отдельный кабель, называемый USB-кабелем типа AB.

Во-вторых, на той же стороне платы размещено гнездо для подключения внешнего источника питания. При подключении к компьютеру через USB-разъем плата будет получать питание 5 вольт через него, таким образом при программировании отдельного внешнего источника обычно не требуется. Если же в гнездо питания вставить штекер от сетевого адаптера, то тогда уже питания от USB не требуется, и плата может работать отдельно от компьютера. Напряжение, которое выдает адаптер, может быть нестабилизированным и должно находиться в пределах 7–12 вольт. Теоретически преобразователь питания, встроенный в Arduino, позволяет подавать внешнее питание до 20 вольт, но он не установлен на радиатор и может перегреваться при значительной разнице напряжений между входом и выходом. Потому

в описаниях Arduino обычно ограничивают питание значением 12 вольт, хотя при небольшой нагрузке платы можно подавать питание и немного выше этого значения.

Подробности

Кстати, если внешний источник стабилизированный, то нижнее значение в 7 вольт также соблюдать необязательно — при необходимости Arduino Uno отлично работает от 5-вольтовых адаптеров или от батарейных блоков с напряжением даже ниже 5 вольт. При покупке адаптера надо учитывать, что диаметр штекера разъема питания составляет 5,5 мм, и более тонкие разъемы (например, от мобильных зарядников) сюда не подойдут. При необходимости подключения нестандартных источников или батарейных блоков можно приобрести универсальный разъем 5,5 мм, в котором установлена колодка для непосредственного подключения проводов от источника «под винт».

На гнезда боковых разъемов в основном выведены напрямую выводы самого контроллера (схему соответствия выводов контроллера и платы можно скачать на любом из официальных сайтов со странички описания Arduino Uno). Назначения выводов в случае фирменной платы Uno подписаны на плате, а дополнительные функции, кроме того, прямо на разъеме. Часть из этих выводов обозначается просто цифрами от 0 до 13 — это цифровые порты ввода/вывода. На схемах и в описаниях мы договорились для ясности добавлять к номеру порта букву D (от слова digital), но в тексте программы это все равно будет просто цифра 3 или 12.

Выводы номер 0 и 1 представляют собой выводы RX и TX последовательного (serial) порта UART. Последовательный порт используется почти в каждой программе (хотя бы на стадии ее загрузки), и в качестве цифровых портов 0 и 1 практически не используют. Потому на платах Mini и Nano (о которых далее) эти два вывода обозначены только как выводы последовательного порта (RX и TX), а нумерация цифровых выводов начинается с двойки.

К выводу номер 13 традиционно подключен светодиод с токоограничивающим резистором 1 кОм (он размещен на плате недалеко от этого вывода и обозначен буквой L). Его можно использовать для проверки работы различных программ, причем работе вывода при другом его использовании этот светодиод не мешает. Шесть выводов с буквой A в обозначении могут работать в качестве аналоговых входов — к ним подключен аналого-цифровой преобразователь (АЦП). Заметим, что аналоговые порты, если они не использованы по назначению, могут также служить в виде обычных цифровых портов ввода/вывода с номерами от 14 до 19. Есть и другие важные функции отдельных выводов, о которых мы поговорим при рассмотрении различным схем.

Кроме портов ввода/вывода, на боковые разъемы с другой стороны платы выведены многочисленные выводы, связанные с питанием, назначение которых понятно из надписей рядом. Напряжение 3,3 вольта получается из отдельного преобразователя, установленного на плате и часто бывает необходимо для питания внешних датчиков (заметим, что у Nano такой преобразователь отсутствует, см. далее). Напряжение Vin повторяет то, что подается на разъем внешнего питания (можно подавать внешнее питание сюда вместо внешнего разъема). Особую функцию выполняет вывод, обозначенный как AREF — в необходимых случаях на него подается

отдельное опорное напряжение для АЦП, в обычном режиме на него никаких напряжений подавать нельзя.

В реальных устройствах Uno целесообразно заменять на почти во всем аналогичную ей, но меньшую по размерам **Arduino Nano** (рис. 20.2, *а*), а также на совсем миниатюрную **Arduino Mini** (20.2, *б* — не путать с Arduino Micro!). Эти платы в целом повторяют устройство Arduino Uno. Поэтому программы, отлаженные на Uno, обычно можно без переделок загружать в Nano и Mini. Только расположены эти версии Arduino на более миниатюрных платах и используют самые малогабаритные корпуса контроллеров. Потому они не могут устанавливаться «в этажерку» стандартных Arduino-габаритов, а контроллер из них извлечь нельзя. Размещение контактов по сторонам плат совместимо с однорядными разъемами с шагом 2,54 мм типа PLS или PBS (см. «Подробности» в *разд. «Железо» главы 19*).

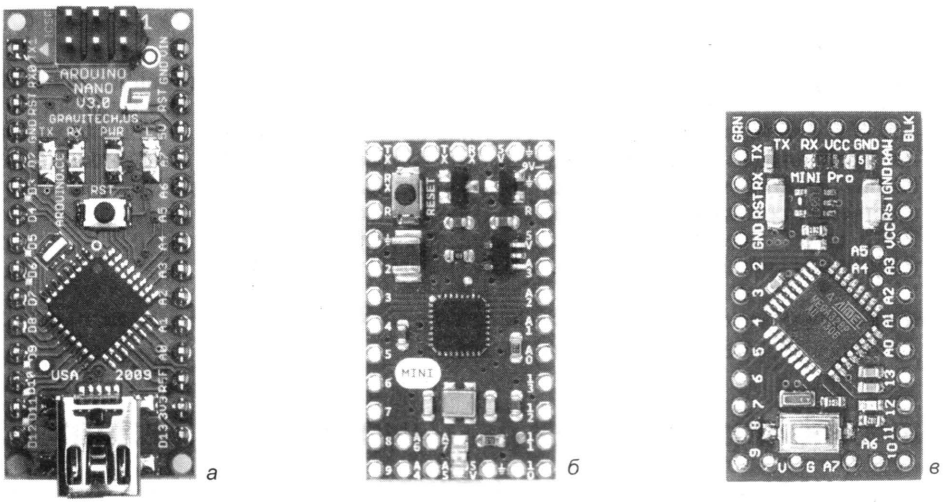


Рис. 20.2. Платы Arduino Nano (*а*), Arduino Mini (*б*) и Arduino Mini Pro (*в*)

Arduino Mini, самый миниатюрный из этих вариантов, лишен USB-разъема и может программироваться или через специальный отдельный адаптер USB-UART, или с помощью платы Arduino Uno (см. *приложение б*). Есть и другие мелкие отличия — так, у Mini отсутствует вывод AREF для подключения внешнего опорного напряжения АЦП, зато есть стабилизатор питания 3,3 вольта для внешних датчиков, как у Uno. У Nano, который почти идентичен Uno по всем остальным компонентам платы, такой стабилизатор отсутствует: видимый на рисунке вывод 3,3 вольта работает только при питании от USB. Нет у этих плат и отдельного разъема для подключения внешнего питания — его следует подключать к контакту Vin. Как вариант, можно подавать от внешнего адаптера стабилизированное напряжение 5 вольт или ниже на вывод с таким же обозначением. Но в этом случае нельзя одновременно подключать плату к USB, потому лучше таким способом не пользоваться — как мы говорили, Arduino прекрасно работает от 5-вольтового адаптера, включенного в качестве внешнего питания Vin.

На платах Mini и Nano имеются дополнительные аналоговые входы A6 и A7, которые обусловлены бóльшим количеством контактов корпуса примененной миниа-

тюрной версии контроллера ATmega328. В отличие от обычных аналоговых входов A0–A5, входы A6 и A7 не могут служить в качестве цифровой линии общего назначения.

В интернет-магазинах очень часто можно встретить так называемый **Arduino Pro Mini** (см. рис. 20.2, *в*). Плата разработана компанией SparkFun Electronics и отличается от обычного Mini наличием дополнительных контактов по коротким сторонам платы, большей частью повторяющих уже имеющиеся — их удобно применять для программирования платы, установленной в схему, когда к боковым контактам доступ затруднен (см. *приложение 6*). По контактам на длинных сторонах платы Pro Mini и Mini полностью совместимы. Arduino Pro Mini обычно продается в виде набора — плата и отдельно PBS-разъемы к ней. Разработчик должен сам запаять те разъемы, которые необходимо. Если все разъемы запаять заранее, плату Mini Pro может быть трудно установить на беспаечную макетную плату.

Подробности

Платы Mini и Mini Pro в силу своих небольших размеров и отсутствия дополнительных микросхем, влияющих на потребление тока, неплохо подходят для устройств с автономным питанием на батарейках. Но у них есть серьезный недостаток в этом отношении: светодиод, сигнализирующий о включении питания. Он потребляет всего несколько миллиампер тока, но при питании экономичного прибора от батареек это существенно сокращает время работы. Поэтому при использовании плат Mini или Mini Pro в таком качестве этот светодиод лучше удалить, аккуратно отпаяв его с платы. Работе плат по прямому назначению это никак не помешает.

Для начала работы со всем этим хозяйством необходимо установить и настроить среду Arduino IDE, чем мы сейчас и займемся.

Установка среды программирования Arduino

Среда программирования Arduino или Arduino IDE (**I**ntegrated **D**evelopment **E**nvironment, интегрированная среда разработки) отличается от других подобных продуктов простотой и компактностью. Можно вообще ее не устанавливать автоматически — просто скачайте ZIP-архив с официального сайта (**http://www.arduino.cc/en/Main/Software**) и распакуйте его на компьютере в любую папку, учитывая при этом, что размещать среду предпочтительно не в привычной Program Files (или в Program Files (x86) для 64-разрядных Windows), а в отдельном каталоге вне системных папок — иначе придется возиться с правами доступа (см. далее).

Проще, конечно, воспользоваться автоматическим установщиком (Installer), причем если у вас русская версия Windows, то вы получите среду сразу на русском языке. Будут также установлены все нужные драйверы. Если у вас имелась старая версия, которая была установлена тем же «инсталлером», то ее лучше предварительно удалить (запустив uninstall.exe из папки Arduino), сохранив предварительно копию папки с библиотеками (Arduino\libraries), которую вы неизбежно дополняли сторонними библиотеками, и папку с проектами (о ней далее). В процессе установки вам, разумеется, предложат установить среду в системный каталог Program Files — повторяю, что предпочтительно указать отдельную папку вне системных каталогов. Запускать нужно файл Arduino.exe из папки Arduino (при автоматической установке на рабочем столе возникнет соответствующий значок).

Для проверки подсоедините любую имеющуюся плату Arduino к порту USB компьютера. Для Uno потребуется обычный AB-кабель USB, для Nano — с разъемом USB-мини. На плате должен при этом загореться светодиод, сигнализирующий о наличии питания (на рис. 20.1 он расположен над контроллером у правого края платы и помечен надписью ON). В диспетчере устройств (**Панель управления | Диспетчер устройств**) в разделе **Порты (COM и LPT)** появится название платы — например, **Arduino UNO (COMxx)**. На случай, если при подключении плат Arduino начнутся какие-то недоразумения с драйверами, то нужный вам драйвер (файл arduino.inf) находится в каталоге Arduino\Drivers (будьте внимательны: именно в папке Drivers, а не в подпапке FTDI USB Drivers). Драйвер можно обновить прямо из диспетчера устройств через пункт контекстного меню **Обновить драйвер**. Можно также действовать через апплет **Панель управления | Устройства и принтеры** — установка или обновление тогда делается через контекстное меню нужного устройства: **Свойства | Оборудование | Свойства | Драйвер | Обновить**. После этого выберите ручной поиск драйверов и укажите упомянутую ранее папку Drivers. В диспетчере устройств и в окне **Устройства и принтеры** после этого возникнет соответствующее устройство с указанием номера привязанного к нему виртуального COM-порта — например, **Arduino Uno (COM8)**.

На рис. 20.3 показано окно Arduino IDE после компилирования демонстрационного примера из коллекции, скачанной с сайта «Амперки», и представляющего собой вывод через последовательный порт традиционного «Здравствуй, Мир!» и производства некоторых других действий по обмену информации с компьютером, о которых далее[1]. Для компиляции с целью проверки загруженного текста надо выбрать пункт меню **Эскиз | Проверить/Компилировать** (или нажать комбинацию клавиш <Ctrl>+<R>), а для его загрузки в контроллер Arduino — пункт **Файл | Загрузить** (или нажать комбинацию клавиш <Ctrl>+<U>). Перед загрузкой файл компилируется заново, потому проверенные программы можно не компилировать отдельно. При сохранении вновь созданной или измененной программы каждый новый проект вас заставят сохранять в отдельной папке, имя которой должно совпадать с именем файла (в общем-то, разумный подход, с точки зрения «чайника»).

Подробности

Имейте в виду, что добавление в папку скетча почти любых других файлов — например, поясняющих текстов, PDF с техническими характеристиками, картинок со схемами и пр. никак не влияет на работу с программой (исключение составляют другие файлы **.ino* или файлы библиотек с расширениями *h* или *cpp*). Что очень удобно, ибо позволяет хранить все, относящееся к проекту, в одном месте. Присутствие других файлов **.ino*, кроме единственного, имя которого совпадает с именем папки, здесь не допускается: они просто не будут открываться. Если вы захотите переименовать программу, то это надо делать не вручную в Проводнике, а путем создания копии с другим именем через меню **Файл | Сохранить как**, после чего папку со старым названием вместе с файлом **.ino* можно удалить вручную.

---

[1] Архив с этими примерами можно скачать с сайта автора по адресу: **http://revich.lib.ru/AVR/amperka-primery.zip**.

А вот скомпилированный HEX-файл, если он вдруг вам понадобится (его можно ведь загружать обычным программатором, без среды Arduino), придется поискать. Результаты деятельности Arduino IDE размещаются в недрах папки *Пользователи\<имя пользователя>\AppData\Local\Temp* (не путайте AppData с системной *Application Data*, куда вас, скорее всего, не пустят). Там вы найдете кучу папок с расширением *tmp*, название которых начинается с *build* (например, *build2903884968954626 56.tmp*) — внутри одной из них и находится искомый HEX-файл, имя которого должно совпадать с именем файла программы.

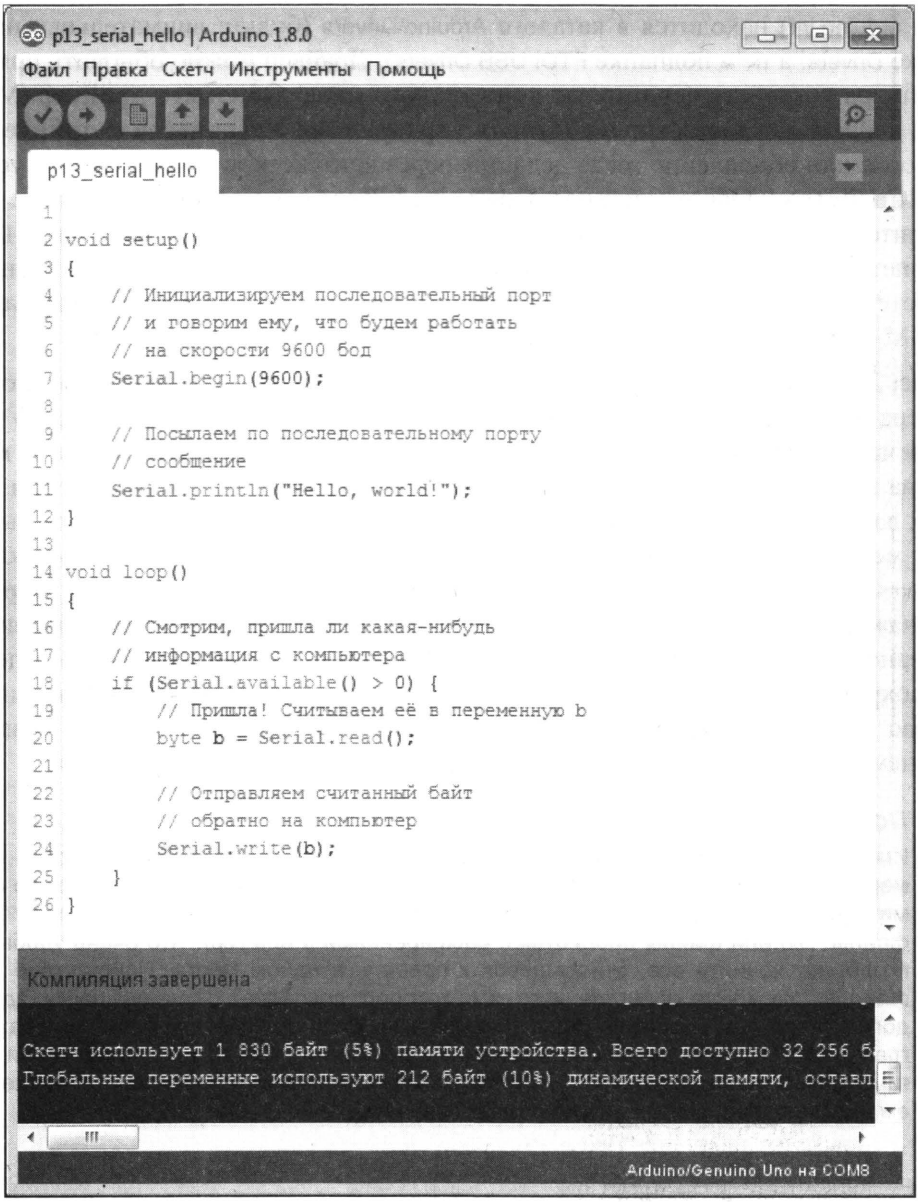


Рис. 20.3. Главное окно Arduino IDE

Скриншот окна Arduino IDE на рис. 20.3 хорошо иллюстрирует главный недостаток программирования микроконтроллеров на высокоуровневом языке, таком как Processing, — программа, содержащая всего несколько строк, в памяти контроллера займет почти два килобайта (см. сообщение внизу). И хотя к этим нескольким строкам следовало бы добавить пару-тройку сотен строк библиотеки Serial (эта библиотека стандартная и подключается автоматически), все равно для такой простой программы это очень много — почти тысяча команд AVR-контроллера, которые с трудом влезут в память, например, знакомого нам ATtiny2313. Аналогичная программа на «голом» AVR-ассемблере заняла бы несколько десятков операторов и спокойно влезла бы в любой контроллер, имеющий встроенный последовательный порт UART (т. е. в любой AVR, за исключением младших представителей семейства Tiny). Такова цена за удобство и скорость разработки — написание и отладка подобной программы на ассемблере у не очень опытного программиста запросто может занять целый день, а в среде Arduino неопытный любитель создаст ее с нуля от силы за час, который в основном потребуется для макетирования схемы с целью проверки функционирования.

> ***Скетч***
>
> Среда программирования Arduino основана на языке Processing, разработанном изначально для художников и дизайнеров, т. е. людей с гуманитарным складом ума. Потому исходные тексты программ в нем называются несерьезным словом «скетчи» (от англ. *sketch* — эскиз, набросок). Обратите внимание, что пункт меню также называется «Скетч» (в некоторых версиях IDE его взялись было переводить, называя «Эскиз», но потом эту практику прекратили). Придется привыкнуть!

Еще больше преимуществ, как мы увидим, такой язык в сравнении с ассемблером дает при выполнении операций с многобайтовыми числами или числами с плавающей запятой и производстве некоторых других подобных действий (например, форматированного вывода чисел на дисплей). Эффективность труда программиста возрастает на много порядков, а платой за это служит снижение быстродействия и увеличение объема необходимой памяти.

Среда Arduino сама не найдет устройство. Даже если оно подключено, но по каким-то причинам связь с компьютером нарушена, то при попытке загрузки программы возникнет сообщение на красном фоне: **Проблема загрузки в плату**, или в окне ниже загадочное: **avrdude: stk500_get sync(): not in sync: resp = 0x00** красным шрифтом. Чтобы этих угрожающих сообщений не возникало, следует после подключения платы и запуска arduino.exe сразу установить нужный COM-порт через меню **Инструменты | Последовательный порт**. Тип подключенной платы также, как правило, не определяется автоматически или определяется неверно (например, при подключении через отдельный адаптер таких плат, как Arduino Mini, не имеющих встроенного USB-порта, драйвер определяет устройство вообще не как Arduino, а как адаптер). Поэтому тип платы также обязательно нужно выбрать отдельно через меню **Инструменты | Плата**. Тогда в нижнем правом углу окна программы появится надпись, соответствующая типу платы и подключенному порту. Можно подключить одновременно и более одной платы — им тогда будут соответствовать разные номера COM-портов. В этом случае надо не забывать менять номер порта и тип платы при переходе от одного скетча к другому.

В процессе отладки коммуникационных функций по последовательному порту вам понадобится подключать устройство Arduino к USB-порту и отключать его. Если вы используете стороннюю коммуникационную программу и забудете ее закрыть перед программированием, то порт может оказаться недоступным для Arduino IDE. Arduino IDE может также запутаться в портах, если вы переходили с одного скетча на другой в другом окне Arduino, а загружали их в один и тот же контроллер.

Для исправления положения прежде всего закройте стороннюю коммуникационную программу или второе окно Arduino и попробуйте загрузить программу в плату заново — скорее всего, дело только в этом. Но при многих включениях и отключениях платы Arduino и переходах между скетчами драйвер может окончательно запутаться, в результате чего последовательный порт окажется совсем недоступен. Чтобы восстановить работоспособность порта, необязательно перезагружать компьютер. Найдите устройство Arduino в диспетчере устройств Windows и в контекстном меню разыщите пункт **Отключить**. Отключите устройство и сразу же включите опять (в Windows 7, 8 и 10 пункт меню будет называться **Задействовать**). После этого порт должен заработать, как надо.

Настройки Arduino IDE

После загрузки драйвера первым делом проверьте пункт **Файл | Настройки** (**File | Preferences** для англоязычной версии). Там вы можете поменять язык самой программы (и, кстати, также и язык сообщений об ошибках), отказаться от проверки наличия обновлений (иначе при каждом запуске будете получать назойливые предложения сменить версию) и, главное, поменять размещение текстов ваших программ (скетчей), заданное по умолчанию.

Во всех последних версиях Windows подобные среды программирования предлагают разместить папку с проектами где-то в недрах папки Users (Пользователи). Способ неудобный (проще хранить среду и привязанные к ней документы в одном каталоге) и опасный (потому что потерять пользовательские папки при переустановке системы — как два байта переслать), но вынужденный — по умолчанию писать в системный каталог Program Files пользовательским программам во всех версиях Windows после XP запрещено. Поэтому я и рекомендовал не устанавливать среду в системный каталог — если вы захотите создать в нем пользовательскую папку с проектами, то придется долго и мучительно возиться с правами доступа. А если вся среда размещена отдельно, то просто создайте внутри каталога Arduino папку, с названием, например, Projects, и укажите его в самом первом пункте настроек через кнопку **Выбрать**.

Много разнообразных настроек доступны через файл preferences.txt (его размещение указано внизу окна настроек). Для последних версий среды туда вмешиваться нет никакой необходимости, но если вы захотите это сделать, то учтите, что Arduino IDE перед внесением изменений должна быть закрыта, иначе в момент закрытия они вернутся к первоначальному состоянию.

Программы для Arduino

Программы для Arduino (скетчи) пишутся на варианте языка Processing/Wiring, специально разработанном для этой среды, потому правильнее всего его называть просто «язык Arduino». Как и многие другие языки, он основан на языке C/C++, потому в случае затруднений в правилах синтаксиса можете смело обращаться к любому сетевому справочнику по функциям этих популярных языков. В среде Arduino работает большинство стандартных функций языка C, так что проблема будет не в том, чтобы найти способ осуществления какого-либо действия (такого, как извлечение корня или преобразование числа в строку и наоборот), а в том, чтобы выбрать подходящий способ из всего многообразия, которым почему-то так гордятся приверженцы этого языка.

Основные обозначения арифметических и логических операций неплохо изложены в соответствующем разделе классического учебника Герберта Шилдта [22]. Там же можно найти справку по большинству функций, пришедших из языка C. Что же касается функций, специфических для Arduino, то они изложены в разделе **Программирование** любого официального сайта Arduino, в том числе и в переводе на русский (автор предпочитает версию на украинском сайте **arduno.ua**, как более проработанную и корректную). Хорошее введение в программирование Arduino на языке C дает учебник [23] — в части описания языка он просто повторяет то, что написано на официальных сайтах, причем даже в более сжатом виде, а вот в конкретных примерах использования эта книга может быть весьма полезной. Для уже немного освоившихся в этой области читателей можно порекомендовать пособие Саймона Монка [24], только стоит учитывать допущенные автором и переводчиками ошибки (см. по этому поводу публикацию автора [25]).

Подробности

Кстати, опыт показывает, что язык Arduino поддерживает многие специфические выражения, не указанные в официальных руководствах, но характерные для C/C++: например, упомянутые там целые типы данных `byte` и `unsigned int` (беззнаковое целое) могут записываться сокращенно: как `uint8_t` (то же самое, что `byte`) и `uint16_t` (собственно 16-разрядное «беззнаковое целое», `unsigned int`). В стандартных библиотечных функциях такие обозначения встречаются гораздо чаще, чем приведенные в справочном разделе сайта. Заметим, что в C есть и еще много других типов, наименование которых образовано по тому же шаблону, но именно эти применительно к Arduino встречаются довольно часто, потому нельзя о них не упомянуть. Поддерживается и ассемблерное представление двоичного числа, как 0bxxxxxxxx (см. *главу 14*), причем не только в пределах 8-ми разрядов (хотя бóльшие, чем байт, числа в этом случае и не требуются). На официальных сайтах указано его нестандартное и потому неудобное для запоминания двоичное представление с префиксом в виде заглавной B, причем действующее только в пределах байта.

Если вы с языком C до сих пор не знакомы, то не ждите, что я здесь вам кратенько изложу все его особенности. Тем более, что наслоения C++, правил компилятора AVRGCC и сверху еще Processing/Wiring вместе с особенностями собственно языка Arduino (предпринятыми как бы «в целях упрощения») превратили изначально не очень строгую структуру языка в полную кашу (для примера см. *Подробности* ранее). Фактически в среде Arduino действует несколько языков, которые могут

незаметно перетекать один в другой, и в некоторой степени объединены они лишь общими правилами синтаксиса. В результате логики и стройности в языке Arduino немного, зато очень много лишнего, непонятного и рассчитанного на тупое запоминание без попытки понять «почему так?». Но не унывайте: чтобы овладеть основами Arduino, изучать язык досконально не требуется — Processing и был придуман для тех, кто не хочет углубляться в программирование. Нам сейчас будет достаточно следующих элементарных сведений.

Любая программа в среде Arduino состоит из трех основных блоков: блока определений, функции установок `setup()` и бесконечного цикла `loop()`, который и составляет собственно программу. Эти блоки полностью аналогичны структуре нашей ассемблерной программы (см. *главу 19*), где с блока определений начиналась программа, функция установок у нас следовала за меткой `Reset`, а бесконечный цикл заключал текст, который выполняется вне прерываний (у нас — то, что между меткой `Cykle:` и оператором `rjmp Cykle`). Явное использование прерываний в программах Arduino — прерогатива достаточно «продвинутых» пользователей, в простых скетчах вы прерываний не встретите, что относится к числу недостатков этой платформы (и мы еще будем об этом говорить).

Подробности

Но было бы ошибкой считать, что прерывания в элементарном Arduino не используются вовсе. Например, в Arduino отсчет времени реализован совершенно так же, как мы делали в *главе 19*, только не с помощью `Timer1`, как у нас, а через восьмиразрядный `Timer0`. Здесь тоже устанавливается прерывание таймера по переполнению и тоже с коэффициентом предделителя 64. При обычной тактовой частоте Arduino, равной 16 МГц, прерывания переполнения восьмиразрядного таймера происходят каждые (64/16)*256 = 1024 микросекунды, что позволяет реализовать такие функции, как `millis()` или `delay()`. Самый частый отсчет возможен при таком коэффициенте каждые 4 микросекунды, что обуславливает приведенное в справочнике по функциям Arduino максимальное разрешение функции отсчета микросекунд `micros()`. Любопытно, что задержка в микросекундах (т. е. функция `delayMicroseconds()`) при этом реализована не на прерываниях, а в виде простой программной задержки, как мы делали в первом примере *главы 19*. Функции коммуникационного порта в стандартной библиотеке Serial (см. далее) также основаны на прерываниях.

Блок определений содержит обычные для почти любого языка программирования определения переменных с указанием их типа, например:

```
int i; //переменная i типа int - 16-разрядный счетчик
byte temp = 0; //рабочая переменная типа byte
float temperature; //переменная - действительное число
//для значения температуры
```

Определение наименований выводов как констант:

```
#define dataPin  16 //dataPin — цифровой вывод 16[1]
//(т.е. вывод A2 платы, см. далее)
```

[1] Напомним, что на платах Arduino и в текстах программ цифровые выводы именуют просто номерами, но на схемах мы к номеру цифрового вывода для определенности будем добавлять привычную букву D.

Выводы можно определять и как константы целого типа `byte` (или `uint8_t`):

```
const byte ledPin = 3;    //цифровой выход управления светодиодом
```

При рассмотрении примеров из Сети и других источников, вы можете заметить, что часто употребляется способ обозначения номеров выводов через обычные переменные, причем типа `int`, занимающего два байта в памяти. Это дважды ошибочно: во-первых, номер вывода в принципе не может занять более одного байта, во-вторых, номер — не переменная, а константа. Так что такой способ означает расходование памяти впустую и к тому же приводит к замедлению работы программы из-за необходимости каждый раз извлекать установленное значение из памяти. Ввод обозначений выводов лучше всего производить через один из двух способов объявления констант в языке Arduino, показанных здесь. В обоих случаях во время компилирования все имена констант автоматически будут просто преобразованы в соответствующие числа и никаких дополнительных действий программе производить не придется.

Ссылка на внешнюю библиотеку вставляется по следующему образцу (обычно в самом начале программы):

```
#include <LiquidCrystal.h> //подключаем библиотеку для работы
//со строчным ЖК-индикатором
```

Название файла библиотеки заключается в угловые скобки, если его надо искать в системных папках библиотек Arduino. Встречается также запись имени файла в двойных прямых кавычках:

```
#include "LiquidCrystal.h"
```

Она обычно применяется в случае, когда файл располагается в одной папке с проектом. Хотя если применить такую запись к обычной библиотеке, то ошибки это не вызовет: не найдя файл в папке проекта, Arduino IDE продолжит его искать в системных папках.

При этом библиотека должна находиться в одной из предопределенных папок, обычно это Arduino\libraries (не путать с каталогом Arduino\LIB!). Это требование, кстати, необязательное, о чем догадываются немногие: библиотеку можно разместить где угодно, если в кавычках указать полный путь к папке. Однако произвольное размещение чревато: в большинстве библиотек имеются ссылки на системные библиотеки Arduino и AVR, и при компиляции они могут не найтись.

Подробности

В этой книге для каждого конкретного устройства указывается ссылка, откуда можно скачать библиотеку, если ее она не входит в установочный комплект Arduino IDE. Скачанная библиотека оказывается в архиве типа ZIP. Распаковать и установить ее в Arduino IDE можно через пункт меню **Скетч | Подключить библиотеку | Добавить ZIP-библиотеку**. Но не сложнее это сделать и вручную: просто скопировать из архива всю имеющуюся там папку библиотеки в каталог *Arduino\libraries*. При этом имя папки обычно совпадает с именем библиотеки, но необязательно: так, скачанные с репозитория **github.com** библиотеки почему-то в конце названия папки обязательно содержат слово «master». Эту добавку можно убрать и, вообще, именовать папку как заблагорассудится — принципиальное значение имеют только имена файлов и собственно

название, которое содержится в одном или нескольких из них. Поэтому папку переименовать просто (это ни на что не повлияет), а вот саму библиотеку пытаться переименовывать я бы вам не посоветовал, пока не наберетесь достаточного опыта — слишком много там всего завязано на установленные названия файлов, самой библиотеки и содержащихся в ней объектов.

Заметим, что вместо *Arduino\libraries* можно копировать новые библиотеки в папку *libraries*, которую Arduino IDE создает автоматически в папке с вашими проектами. Хранить там постоянно используемые библиотеки неудобно, особенно тогда, когда папка с проектами расположена в пользовательских папках, а каталог Arduino — в *Program Files*, как это делается по умолчанию. Библиотеку следует скопировать в эту внутреннюю папку, когда в нее необходимо внести изменения — среда Arduino в первую очередь просматривает именно ее, и конфликта со старым вариантом не возникнет, несмотря на одинаковые названия самой библиотеки и ее функций.

После перезапуска среды Arduino следует проверить доступность новой библиотеки, разыскав ее в списке меню **Скетч | Подключить библиотеку**. Отметим, что в этом списке новые установленные вами библиотеки будут отделены от имеющихся по умолчанию, что облегчает поиск.

Строчные и заглавные буквы в языке Arduino различаются, что доставляет кучу неудобств привыкшим к другим языкам высокого уровня или AVR-ассемблеру. Например, `string()` и `String()` — это разные функции (см. справочник по языку на официальном сайте). В языке С любые определения можно делать в любом месте программы (и даже прямо при вызове функции), выносить их в начало необязательно. Только стоит учесть, что, например, вызов переменной, определенной внутри некоей функции (*локальная переменная*), в другой функции вызовет сообщение об ошибке. Для того чтобы переменная действовала для всей программы, она должна быть определена именно в начале, до всех функций в разделе установок (*глобальная переменная*). Нюанс заключается в том, что глобальная переменная займет ресурсы контроллера на все время работы программы, тогда как локальная освободит их по окончании действия функции. В условиях ограниченных ресурсов МК это может оказаться существенным фактором, влияющим на скорость выполнения программы.

Подробности

Есть один важный момент, касающийся некоторых особенностей компилятора языка С в отношении глобальных и локальных переменных. Если у вас в программе есть глобальные переменные, хранящие значения, которые могут быть изменены в коде обработчика прерывания, то такие переменные должны быть объявлены с ключевым словом `volatile`. Компилятор ничего о таких сущностях, как прерывания, не знает, потому полагает, что значение глобальной переменной, однажды присвоенное в основном цикле программы, должно сохраняться до следующего такого изменения. И если оно было внезапно изменено где-то в параллельно исполнявшемся прерывании таймера, то может возникнуть трудно обнаруживаемая ошибка времени выполнения. А объявление типа `volatile` даст понять компилятору, что такая переменная может быть изменена в любой момент[1]. В программах для микроконтроллеров такие ситуа-

[1] На самом деле собственно компилятор такой дурацкой ошибки не сделает. Но в Arduino разрешена оптимизация кода при компиляции, и вот оптимизатор может натворить в том числе и такое. Объявление типа `volatile` явным образом запретит ему это делать.

ции встречаются сплошь да рядом. Простой пример дает цикл ожидания изменения какой-либо переменной через прерывание:

```
byte Flag = FALSE;
interrupt void rx_isr(void) //прерывание приема порта Serial
{
    ...
    if (ETX == rx_char)
    {
        Flag = TRUE;
    }
    ...
} // end прерывание

void loop ()
{
    ...
    while (!Flag)
    {
        //ожидаем
    }
    ...
}//end loop
```

В таком виде цикл ожидания `while (!Flag)` может не выполниться никогда, и вы так и не узнаете, почему так случилось. Здесь объявление переменной `Flag` необходимо обязательно дополнить типом `volatile`:

```
volatile byte Flag = FALSE;
```

Другие примеры подобного объявления можно увидеть в следующей главе. Есть и иные случаи необходимости применения ключевого слова `volatile`, но в наших программах они не встречаются.

Наша процедура `Reset` (блок установок из *главы 19*) здесь выглядит как функция `setup`:

```
void setup()
{
< операторы >
}
```

Следует заметить, что в языке C служебное слово `void` («пустота») обозначает, что за ним последует то, что в более строгих языках программирования носит название «процедура» — т. е. функция, не возвращающая никакого значения. Между фигурными скобками здесь размещаются те операторы, которые должны выполняться при запуске программы один раз. Потому пусть простят меня ревнители языка C за то, что я буду употреблять термины «процедура» и «функция» вперемешку.

После `setup` обычно идет функция (на самом деле тоже процедура) бесконечного цикла, которая обозначается словом `loop` («петля»):

```
void loop()
{
< операторы >
}
```

Кроме этих двух обязательных функций, программа для Arduino может включать в себя любое количество других функций (или процедур), определяемых пользователем, и примеры этого мы увидим далее.

Следует учесть, что числа в Arduino имеют ограниченный диапазон: 2 байта (16 двоичных разрядов, т. е. значения от –32 768 до 32 767) для типа `int` и 4 байта (32 разряда или от –2 147 483 648 до 2 147 483 647) для `longint`. Для вещественных чисел этот диапазон составляет от $-3{,}4028235 \cdot 10^{+38}$ до $3{,}4028235 \cdot 10^{+38}$, т. е. точность не более 6–7 знаков после запятой (о представлении действительных чисел в контроллерах см. *главу 14*). Причем здесь типы `float` и `double`, в отличие от стандартного C, идентичны по размеру, и большей точности в операциях с вещественными числами не добиться.

В заключение нашего суперкраткого обзора программирования для Arduino стоит напомнить про некоторые особенности логических операций в языке C, которые почти не играют роли в обычном программировании, но в приложении к микроконтроллерам имеют важное значение. Это касается выполнения базовых логических функций «И», «ИЛИ» и «НЕ», которые мы изучали в *главе 14* (см. также [22]). В языке C имеются две разновидности логических операций: обычные («логическое И» &&, «логическое ИЛИ» ||, «логическое НЕ» !) и поразрядные битовые («поразрядное И» &, «поразрядное ИЛИ» |, «поразрядное НЕ» ~). Теперь вы можете с полным пониманием отнестись к этому разделению: обычные логические операции относятся к булевым переменным (т. е. таким, которые принимают только два значения: «ноль»/«не ноль», «ложь»/«правда»), а поразрядные — к числовым переменным, т. е. попросту к нашим родным регистрам контроллера.

В условных операторах (`if`) должна присутствовать чисто логическая операция с бинарным исходом («правда» — «ложь»), потому там надо ставить символы логических операций, а вот в операциях с числами и регистрами — поразрядных. Например, значок неравенства в языке C запишется как «!=» (буквально и значит «не равно»), а запись «~=» будет бессмысленной. Но одинарные символы (& вместо положенного && или | вместо положенного ||) все равно часто ставят в условном операторе `if`, потому что там обычно фигурируют бинарные операции, вроде операций сравнения («больше», «меньше», «равно», «не равно» и т. п.), которые сами по себе в результате дают логическое значение. То есть они фактически состоят из одного двоичного разряда, и применение к ним побитовой операции даст ровно тот же результат, что и обычной логической.

Для новичков в нотации языка C укажем также, что стандартные битовые операции здесь могут выполняться точно так же, как на ассемблере (см. примеры в предыдущих главах), но их синтаксис несколько другой:

```
DDRA |= (1<<1)|(1<<4); //выводы 1 и 4 порта A - в единицу
DDRA &= ~((1<<2)|(1<<3)); //выводы 2 и 3 порта A - в ноль
//биты COM1A1(бит 7) и WGM10 (бит 0) р-ра TCCR0A - в единицу:
TCCR1A|= (1<<COM1A1)|(1<<WGM10));
```

Можно и так, что уже гораздо ближе к языку высокого уровня, чем к ассемблеру:

```
bitSet(TCCR1B, WGM12); //бит WGM12 (бит 3) р-ра TCCR1B - в единицу;
bitClear(TCCR0A, COM0A1); бит COM0A1 (бит 7) р-ра TCCR0A - в ноль;
```

И тот и другой способ можно произвольно смешивать в тексте программы.

Хочется обратить ваше внимание на стиль написания этих примеров: в языке Arduino, как в других вариантах языков высокого уровня для AVR, возможно непосредственное управление портами и регистрами по их названиям, подобно тому, как это делается в ассемблере[1]. Иногда это более целесообразно, чем применять высокоуровневые функции. Например, строка

```
DDRB = B00000100;
```

равносильна записи

```
pinMode(10, OUTPUT);
```

Вывод Arduino D10 подключен к третьему контакту PB2 порта B контроллера (см. схему соответствия портов контроллера и платы Arduino, о которой говорилось ранее), а регистр DDRB как раз и управляет направлением вывода. Потому обе записи устанавливают цифровой вывод 10 на выход, но при этом первая выполняется гораздо быстрее.

Очень наглядный пример представляет функция `digitalWrite()`. Так, `digitalWrite(10, HIGH);` выполняется в разы и даже в десятки раз медленнее, чем прямая запись в порт `PORTB = B00000100;` (подробнее об этом см. в *разд. «О преимуществах и недостатках Arduino»* в конце *главы 22*).

Подобные приемы нередко используются, например, при оформлении прерываний (см. далее *разд. «Правильная мигалка на Ardiuno»*). В обычной любительской практике этим заниматься, конечно, не имеет смысла — преимущества высокоуровневого языка Arduino при этом теряются, и разобраться в такой программе куда сложнее.

Зависания программ

Как и на любом компьютере, программы на контроллере Arduino могут зависать. Причем это явление в случае обычной микропрограммы для AVR, тем более написанной на ассемблере прямо в кодах контроллера (как в предыдущих главах книги), наблюдается гораздо реже, чем для скетчей на высокоуровневом языке Arduino, в которых каждая строчка есть фактически обращение к той или иной библиотечной функции или макросу, иногда весьма большого объема и с большим количеством условных переходов. Для таких несложных устройств, как схемы на контроллерах, есть свои эффективные способы преодоления подобных исключительных ситуаций. На них мы остановимся в конце *главы 21* в *разд. «О режиме энергосбережения, Watchdog-таймере и питании метеостанции»*.

---

[1] Есть в языке Arduino, разумеется, и способ прямого включения ассемблерного кода в текст программы (т. н. inline-ассемблер), но результат его применения мало чем отличается от изложенного способа прямого обращения к регистрам, потому используют его редко.

Примеры программирования

Приведенные далее примеры программирования представляют собой иллюстрацию к применению некоторых несложных действий, доступных Arduino. Некоторые из этих примеров, впрочем, вполне можно использовать в качестве готового проекта или его части для создания устройств бытовой автоматики. Начнем мы с освоения работы с коммуникационной программой Монитор порта, которая нам потом поможет при отладке наших скетчей.

Обмен через последовательный порт

Для лучшего понимания того, что именно мы будем делать дальше, кратко рассмотрим устройство последовательного порта. Сначала разберемся в терминах, которые имеют отношение к предмету разговора. В компьютерах ранее всегда присутствовал COM-порт, часто кратко называемый портом RS-232. Правильнее сказать так: COM-порт передает данные, основываясь на стандарте последовательного интерфейса RS-232. UART (Universal Asynchronous Receiver-Transmitter, универсальный асинхронный приемопередатчик) есть основная часть любого устройства, поддерживающего RS-232. Соответственно, UART как составная часть входит практически во все универсальные микроконтроллеры, в том числе и в ATmega328, лежащий в основе Arduino. В контроллере ATmega2560 (Arduino Mega) таких портов даже целых три.

Кроме UART в порт RS-232 (в том числе в COM-порт ПК) входит схема преобразования логических уровней в уровни RS-232, где биты передаются разнополярными уровнями напряжения, притом инвертированными относительно UART. В UART действует положительная логика с обычными логическими уровнями, где логическая единица есть высокий уровень (+3 или +5 В), а логический ноль — низкий уровень (0 В). У RS-232 логическая единица передается отрицательным уровнем от –3 до –12 В, а логический ноль — положительным уровнем от +3 до +12 В. Преобразователь уровня в контроллер, естественно, не входит, так что для стыковки с компьютером придется его приобретать или изобретать самостоятельно.

В Arduino все эта история, однако, завернута гораздо круче. Дело в том, что COM-порт был стандартным оборудованием любого ПК или ноутбука вплоть до 1999 года, а в новом тысячелетии от него, как и от порта LPT, отказались. Поэтому для совместимости приходится приобретать специальные USB-кабели или вставные платы, организующие такой порт на стороне компьютера. В Arduino от такого внешнего решения сразу отказались: т. к. уровни UART все равно приходится преобразовывать в уровни RS-232, так почему бы не поставить преобразователь сразу в USB?

Потому плата Arduino работает через USB, но со стороны компьютера выглядит как COM-порт, о чем мы уже говорили. Этот порт используется и для программирования контроллера, и для обмена данными при условии, что в программе таковой обмен предусмотрен. Напомним, что для обмена данными библиотека Serial задействует цифровые порты платы Arduino 0 (RX) и 1 (TX). Разумеется, если вы обра-

щаетесь к функциям Serial, то нельзя одновременно с этим использовать порты D0 и D1 для других целей, — обратите внимание, что во всех наших проектах они останутся свободными.

Рассмотрим работу с последовательным портом на примере пробной программы, которую мы загрузили для образца (рис. 20.3). Она выполняет следующие действия: сначала посылает через последовательный порт в компьютер традиционное сообщение «Hello, world!», что означает «Здравствуй, мир!». После его посылки она переходит к ожиданию сообщений со стороны компьютера через тот же порт. Каждый из принятых символов контроллер будет тут же отсылать обратно.

Еще раз напомним, что последовательный (serial) порт в данном случае — тот же самый USB, через который осуществлялось программирование. Для взаимодействия с ним нужна какая-то коммуникационная программа со стороны компьютера. Для несложных надобностей, в том числе и для отладки скетчей, удобнее всего пользоваться простейшим Монитором порта, входящим в состав пакета программирования Arduino.

Для его запуска обратитесь к меню **Инструменты | Монитор порта**. В момент вызова контроллер будет перезагружен и отработает с самого начала заново. Вручную перезагрузку можно осуществить нажатием кнопочки Reset, которая имеется на всех платах Arduino (на Uno она размещается рядом с USB-разъемом, см. рис. 20.1).

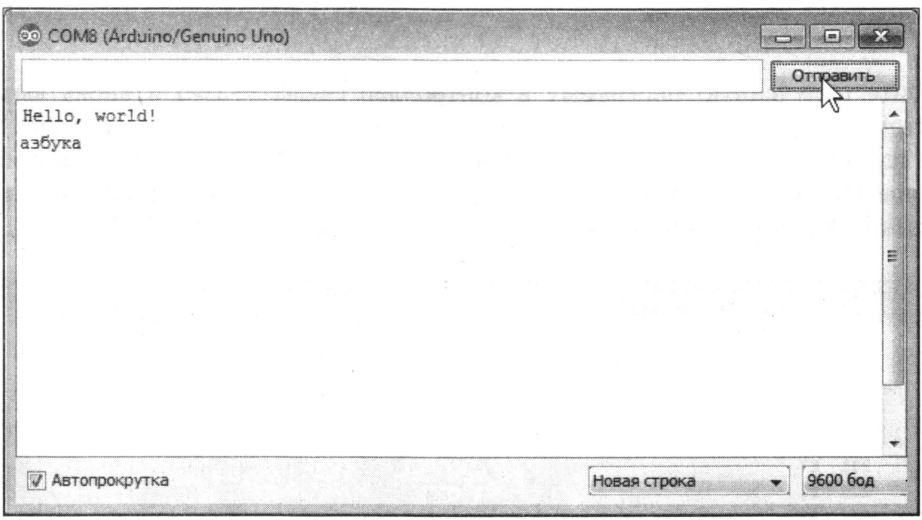


Рис. 20.4. Монитор порта Arduino IDE

В результате после вызова у вас откроется окно, показанное на рис. 20.4. Через пару секунд (когда отработает процедура перезагрузки) в окне появится надпись «Hello, world!» (см. верхнюю строку в окне на рисунке). Это означает, что программа в контроллере работает как надо. Чтобы проверить остальную часть, установите курсор в строку ввода вверху окна и наберите в нем строку по-русски, например: `азбука`. Перед тем, как посылать ее в контроллер, через выпадающий список внизу окна выберите пункт **Новая строка** (иначе новые принятые символы

выстроятся вслед за принятыми ранее без какого-либо промежутка). Теперь нажмите на кнопку **Отправить**. Результат вы видите на том же рисунке — набранная строка без изменений возвратилась в компьютер.

А что, собственно, посылается через последовательный порт? Ясно, что числа, но их представление может быть самым разным: так, тип `char`, как и положено по названию, может означать символ, а может число, причем со знаком. Давайте немного разберемся, как именно интерпретируются символы при передаче в Монитор порта.

Для этого немного изменим текст нашего пробного скетча «Hello, word!». Обратная отсылка принятых символов (обозначенных в программе переменной под названием `b` типа `byte`) осуществляется в строке программы, где написано `Serial.write(b)`. Самостоятельно замените эту единственную строку в тексте скетча на следующую последовательность строк (и не забывайте ставить точку с запятой после каждого оператора):

```
. .
Serial.println();
Serial.write(b);
Serial.print("  ");
Serial.print(b, DEC);
Serial.print("  ");
Serial.print(b, HEX);
```

Загрузите полученную программу в контроллер (<Ctrl>+<U>) и вновь вызовите Монитор порта. Убедитесь, что строка `Hello, world!` принимается как надо. После этого верните пункт в выпадающем списке внизу окна Монитора порта в исходное значение **Нет конца строки** и наберите в строке ввода то же самое слово `азбука` по-русски. Результат приведен на рис. 20.5. Что же мы там получили?

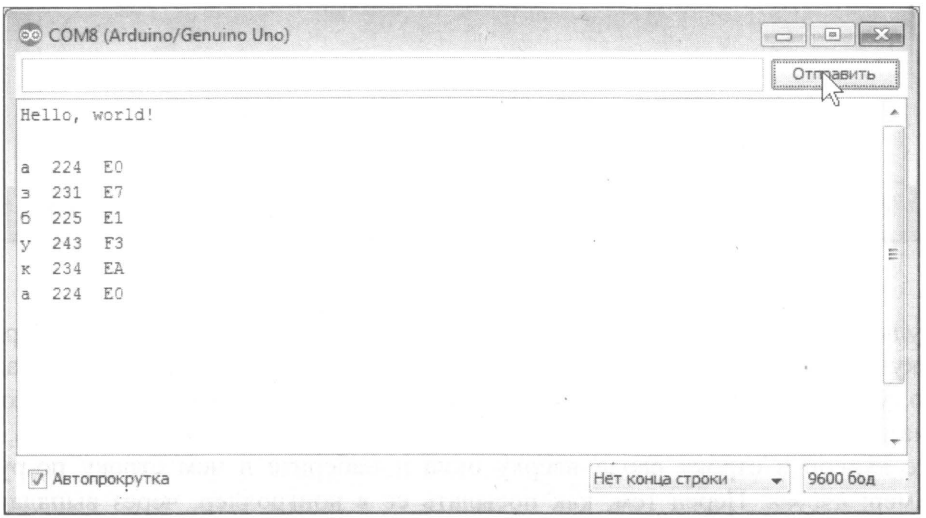


Рис. 20.5. Интерпретация различных способов представления числа в Мониторе порта

В первой внесенной нами строке стоит функция `Serial.println()`, которая посылает единственный невидимый символ — символ конца строки. Он предназначен для того, чтобы новая посылка начиналась с новой строки (т. е. для того же, для чего ранее мы выбирали пункт меню **Новая строка**). Следующая функция `Serial.write(b);` оставлена от исходной программы — она передает посланный в программу символ обратно без изменений. Функция `Serial.print(" ");` (в тексте она повторяется дважды) передает строку из двух разделяющих пробелов для удобства чтения выведенного текста. И наконец, функции `Serial.print(b, DEC);` и `Serial.print(b, HEX);` посылают в компьютер не сам символ, как это делает функция `Serial.write`, а его десятичный (DEC) и шестнадцатеричный (HEX) коды.

Таким образом, в каждой строке ответа на посылку располагается сам посланный символ, его десятичный код, отделенный пробелами, и, наконец, шестнадцатеричный код. Аналогично можно вывести двоичное (BIN) представление символов (попробуйте добавить вариант `Serial.print(b, BIN)`). Таким способом можно посылать числа в различном представлении, что очень помогает при отладке программ.

> ***Подробности***
>
> При этом следует учитывать, что ни функция `Serial.print()`, ни функция `Serial.write()` не «понимают» русских строк и отдельных символов, введенных в программе напрямую в двойных кавычках. Результат действия оператора `Serial.print("Привет, мир!")` отобразится в Мониторе порта в виде абракадабры («РџСЂРёРІРµС„, РјРёСЂ!»). Это происходит потому, что редактор Arduino IDE работает с двухбайтовыми символами в кодировке UTF-8 (как это и положено в современных версиях Windows). При пересылке из компьютера этот двухбайтовый код Монитором порта автоматически конвертируется в однобайтовую кодировку ANSI и далее все происходит как обычно, потому в том случае русские символы интерпретировались правильно. А функция `Serial.print()`, конечно, никаких конвертаций не производит. Подробнее на эту тему мы будем говорить при рассмотрении вывода информации на различные типы дисплеев в *главе 21*.

Для более «продвинутых» читателей вместо Монитора порта я рекомендую свою программу-монитор под названием Com2000 (ее можно скачать с сайта автора по адресу **http://revich.lib.ru/comcom.zip**) — она отличается тем, что позволяет организовать обмен в любом удобном формате (численном в десятичной или шестнадцатеричной форме, а также в текстовом). Входит подобная программа, как составная часть, и в утилиту X-CTU для настройки радиомодулей по протоколу Xbee, которыми мы будем заниматься в следующей главе.

Термостат на Arduino

Давайте соорудим для начала на Arduino что-нибудь простенькое. В *главе 12* мы уже изобретали термостаты на чисто аналоговых компонентах. Теперь посмотрим, как можно привлечь к этому полезному в хозяйстве делу цифровую технику.

Мы уже упоминали (см. *главу 18*), что в состав AVR-контроллеров входит 10-разрядный многоканальный АЦП. На платах Arduino его выводы специально помечены как аналоговые входы (буквой A с цифрами от нуля до пяти). Один из этих вхо-

дов мы как раз и задействуем для измерения температуры, а управлять подключением нагрузки будем с одного из цифровых выходов.

Итого нам понадобятся:

- плата Arduino Uno (годится и любая другая из упомянутых ранее);
- термистор в качестве датчика температуры. Подойдет, например, B57164-K 103-J с номинальным сопротивлением 10 кОм при 25 °С — именно его характеристики приведены в *главе 13* в качестве иллюстрации к свойствам термисторов;
- переменный резистор 10 кОм, постоянный резистор 620 Ом;
- исполнительное реле — электромагнитное (обязательно с усилительным транзисторным ключом, см. далее) или твердотельное.

В продаже имеются модули на основе 5-вольтовых электромагнитных реле, специально подогнанных под управление от выходов Arduino. Электромагнитные реле сами по себе требуют довольно большого тока управления (и он тем больше, чем мощнее реле, которое к тому же может вызывать помехи в работе контроллера — см. *главу 7*), потому во всех подобных релейных модулях обязательно имеется транзисторный усилительный ключ[1].

Если вас электромагнитное реле не устраивает, и вы стремитесь к предельной простоте схемы, то можно поискать твердотельные реле — подойдут, например, CX240D5R фирмы Crydom или аналогичные с напряжением срабатывания 3–15 В. У них ток управления составляет около 15 мА при 5 вольтах на входе, что допустимо для AVR, потому их управляющий вход можно подключать к цифровому выводу Arduino напрямую. Правда, при напряжении 220 вольт коммутировать нагрузку мощностью больше киловатта CX240D5R не может, но нам в рассматриваемой задаче больше и не требуется.

Схема термостата на Arduino Uno показана на рис. 20.6. На схеме управление реле K1 от вывода D2 условно показано в виде обычной «обмотки», но для электромагнитного реле это будет вход управляющего транзисторного ключа, а для оптоэлектронного — выводы управляющего светодиода с резистором. K1 должно иметь нормально разомкнутые контакты. Для контроля состояния схемы одновременно с нагревателем срабатывает светодиод, подключенный к выводу D3. Программа термостата в соответствии с подобной схемой крайне проста:

```
#define termPin = A0; //Аналоговый вход термистора
#define relayPin = 2; //Цифровой выход управления реле
#define ledPin = 3;   //Цифровой выход управления
     //сигнальным светодиодом
void setup()
{
    pinMode(relayPin, OUTPUT); //оба цифровых вывода - на выход
```

[1] Хотя продавцы этот факт почему-то тщательно замалчивают — отсутствие документации или хотя бы исчерпывающих технических характеристик готовых комплектующих изделий для Arduino часто перерастает в серьезную проблему для желающего ими воспользоваться.

```
    pinMode(ledPin, OUTPUT);
}
void loop()
{
    //Считываем показания термистора и сравниваем
    //их с пороговыми значениями в середине диапазона АЦП
if (analogRead(termPin) > 510) {
       //Если температура низкая (сопротивление термистора высокое),
       //включаем нагреватель и светодиод
        digitalWrite(relayPin, HIGH);
        digitalWrite(ledPin, HIGH);
    }
if (analogRead(termPin) < 490)  {
        //Если температура высокая - выключаем
        digitalWrite(relayPin, LOW);
        digitalWrite(ledPin, LOW);
    }
}
```

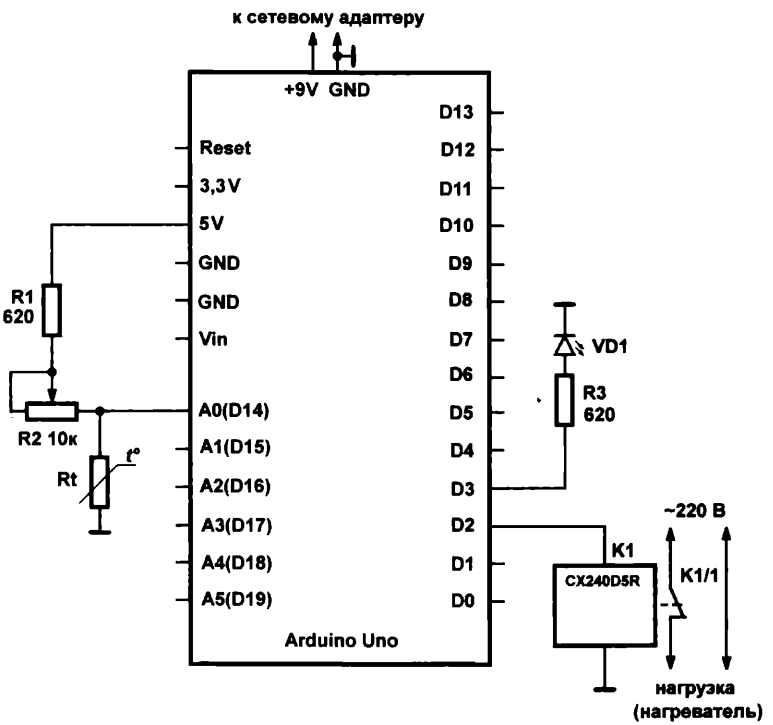


Рис. 20.6. Схема термостата на Arduino Uno

Величины резисторов подогнаны под указанный термистор B57164-K с номинальным сопротивлением 10 кОм при 25 °C (тип 103-J). В соответствии с программой срабатывание реле будет происходить вблизи значения на выходе АЦП, равного

500. Это составляет примерно середину 10-разрядного диапазона (вся шкала — 1024 градации), т. е. такое значение установится при приблизительном равенстве верхнего и нижнего сопротивлений относительно входа A0 (напряжение на этом входе тогда составит примерно 2,5 вольта).

Обратите внимание, что обе функции `if` не заканчиваются привычным `else`. Для предотвращения дребезга в программу введен гистерезис: реле включается при превышении значения кода 510, а выключается при снижении до значения 490. В промежутке оно будет сохранять предыдущее состояние. Двадцать единиц кода (то, что в *главе 12* мы называли *зоной нечувствительности*) соответствуют примерно 10 милливольтам, т. е. гистерезис при температуре в пределах 30–40 градусов составит чуть меньше одной десятой градуса (проверьте сами с помощью табл. 13.1 из *главы 13*).

Установка температуры срабатывания с помощью резистора R2 при таких параметрах возможна в пределах примерно от 22 до 96 °C. Разумеется, на практике такой широкий диапазон регулировки не требуется, потому целесообразно номинал R2 уменьшить. Величина R1 подбирается так, чтобы R1 и номинальное значение R2 в сумме составляли сопротивление термистора при нижнем значении желаемого диапазона температур (в соответствии с табл. 13.1). Для более точной подгонки можно провести калибровку и изменить пороговые значения в программе, измеряя установившуюся температуру обычным термометром.

Если вы примените в этой схеме другие датчики, то не забудьте про знак температурного коэффициента. Обычный диод или транзистор в диодном включении (как в схемах из *главы 13*) также имеют отрицательный наклон характеристики, потому для них в программе придется поменять только числовые значения порога срабатывания. А вот полупроводниковые датчики типа TMP35 (см. *главу 13*) или просто металлические термометры сопротивления (как в конструкции из *главы 17*) имеют положительный температурный коэффициент, поэтому условия срабатывания придется изменить на обратные. Причем не просто поменять «больше» на «меньше» и наоборот, а изменить и соотношение порогов для гистерезиса — в новой ситуации нагреватель должен будет включаться, если значение меньше меньшего порога, а выключаться — если больше большего.

Как видите, обращаться с Arduino не просто, а очень просто. Работа с АЦП относится к базовым функциям платформы и не требует даже подключения отдельных библиотек. Оцените, насколько облегчили создатели платформы жизнь разработчику: вызову функции `analogRead()` соответствуют операции установки режима АЦП, тактовой частоты его работы, выбора канала и пр.

Правильное подключение кнопки

Управлять действиями контроллера с помощью кнопки сложнее, чем кажется. Причину этого мы уже рассматривали не раз: в момент и нажатия, и отпускания большинство кнопок дребезжит, некоторое время находясь в неопределенном состоянии. Контроллёр — достаточно быстрое устройство, чтобы отрабатывать такой дребезг. Следовательно, от дребезга надо избавляться, иначе результат нажатия или

отпускания кнопки также будет неопределенным. Но вот вопрос — а во всех ли случаях это необходимо?

Устройства, в которых применяется кнопка, делятся на две категории. В одних важен сам факт того, что кнопка пребывает в отпущенном или нажатом состоянии. К таким случаям, например, относится управление сигнальными лампочками: кнопка нажата — лампа горит, кнопка отпущена — гаснет (или меняет цвет). Легко сообразить, что здесь дребезг не играет никакой роли. В ряде случаев, однако, требуется другой алгоритм действий — когда что-то должно сработать в момент нажатия кнопки, и больше не реагировать до следующего нажатия (типичный пример — счетный триггер, см. *главу 16*). Вот здесь обязательно нужно избавляться от дребезга контактов, иначе схема будет просто неработоспособна.

Рассмотрим оба случая отдельно. Для этого возьмем плату Uno и подключим любую двухвыводную кнопку к выводу 3 и к «земле» (GND). Двухцветный двухвыводной светодиод (BL-L317, L-117 или аналогичный) подключим через резистор 330 Ом к выводам 12 и 13 (к последнему, напомним, уже подключен светодиод L, он имеется на любой плате из перечисленных в начале главы). Все это показано на рис. 20.7, где в стиле всех этих интернет-ресурсов категории «Arduino для чайников» вместо схемы приведен фрагмент платы Uno с подключенными компонентами (рисунок исполнен во Fritzing, см. *главу 3*).

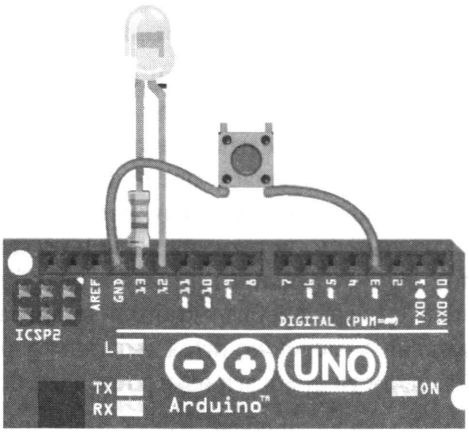


Рис. 20.7. Подключение кнопки и светодиода к плате Arduino Uno

Текст примера с переключением цветов по нажатию кнопки будет таким:

```
#define BUTTON_PIN 3 //вывод подключения кнопки
#define LED_RED_PIN 13 //вывод управления красным светодиодом
#define LED_GREEN_PIN 12 //вывод управления зеленым светодиодом
void setup() {
    pinMode(LED_RED_PIN, OUTPUT); //вывод красного - на выход
    pinMode(LED_GREEN_PIN, OUTPUT); //вывод зеленым - на выход
    pinMode(BUTTON_PIN, INPUT_PULLUP); //вывод кнопки на вход
                                       //с подтягивающим резистором
}
```

```
void loop() {
   if (digitalRead(BUTTON_PIN) == HIGH) {  //кнопка не нажата
    //включаем зеленый светодиод
    digitalWrite(LED_GREEN_PIN, HIGH);
    digitalWrite(LED_RED_PIN, LOW);
  }
  else { //кнопка нажата
    //включаем красный светодиод
    digitalWrite(LED_GREEN_PIN, LOW);
    digitalWrite(LED_RED_PIN, HIGH);
  }
}
```

Обратите внимание на последнюю строку в процедуре `setup`: вывод, к которому подключена кнопка, подключается на вход, т. е. на прием сигналов (`INPUT`), что можно было бы вообще не указывать — в исходном состоянии все выводы контроллера подключены именно для работы на вход. Однако мы не должны оставлять «висящим в воздухе» вывод кнопки, подключенный в разомкнутом виде к выводу в третьем состоянии с входным сопротивлением, измеряемым гигаомами. Можно было бы подключить обычный внешний резистор (в пределах 1–10 кОм), подтянутый к питанию, что было бы надежней в случае наличия возможных помех, но в рядовом случае можно воспользоваться встроенным подтягивающим резистором. Константа `INPUT_PULLUP` и указывает функции `pinMode()` на его подключение.

После выполнения такой команды в разомкнутом состоянии кнопки на выводе `BUTTON_PIN` все время будет «висеть» логическая единица. Так что наша программа обязана отслеживать момент, когда на нем окажется логический ноль: это и будет означать, что кнопка нажата. Это делает последовательность команд в главном цикле программы: в течение времени нажатия кнопки цвет свечения светодиода меняется с зеленого на красный (и, кстати, параллельно зажигается желтый светодиод L на плате Arduino). При отпускании он меняется обратно на зеленый (а светодиод L гаснет). В этом случае дребезг кнопки не играет никакой роли: если светодиод в момент нажатия и переключится несколько раз, мы этого просто не заметим, поскольку длительность дребезга измеряется миллисекундами.

Теперь рассмотрим случай, когда дребезг имеет значение: например, пусть при нажатии кнопка каждый раз меняет цвет светодиода на противоположный (получим нечто подобное упомянутому счетному триггеру, делящему частоту входных импульсов на два, см. *главу 16*).

Подробности

Есть несколько алгоритмов разной степени надежности, позволяющих избавиться от дребезга кнопок. Самый надежный заключается в имитации схемы с RS-триггером (см. рис. 16.3, *а*), т. е. в использовании трехвыводной кнопки, подключенной сразу к двум выходам контроллера (если на одном высокий уровень, на другом должен быть низкий и наоборот). На практике он малоприменим из-за отсутствия достаточного выбора трехвыводных кнопок, поэтому чаще применяют разные модификации способа с задержками (т. е. имитацию действия одновибратора, см. *разд. «Одновибраторы»* в *главе 16*). Канонический алгоритм состоит в том, что кнопку подключают к одному из

выходов, где действует внешнее прерывание (у Arduino это выводы 2 и 3). При возникновении такого прерывания, например, по спаду импульса (т. е. при переключении входа из высокого уровня в низкий) в обработчике прерывания его первой же командой запрещают, а включают обратно по таймеру через некоторый промежуток времени, гарантирующий, что к этому моменту кнопка уже будет отпущена.

В Arduino чаще применяют упрощенную (и менее надежную) модификацию этого способа, когда задержка обеспечивается просто функцией `delay()`. Вариант такого алгоритма будет приведен в *разд. «Подключение тастатуры 3×4» главы 22*. Но опыт показал, что в рядовом случае здесь проще и эффективней применять алгоритм с многократной проверкой состояния кнопки, как описано далее в этом разделе — он вполне надежен и к тому же практически не тормозит программу. Совсем избавиться от тормозящих программу задержек можно, если объединить способ многократной проверки с прерыванием, но здесь мы такими наворотами заниматься не будем.

Итак, принцип выбранного способа заключается в том, чтобы несколько раз определять состояние кнопки в течение заданного промежутка времени — в этом случае факт нажатия можно установить весьма точно. Опыт показывает, что для достаточно надежной работы хватает одной повторной проверки примерно через 0,01 секунды (при нужде программу легко модифицировать и для нескольких таких проверок). Схему мы используем ту же самую, что в предыдущем примере, изменения коснутся только программы. Пусть цвет светодиода меняется именно при нажатии, т. е. при возникновении на выводе 3 перепада из высокого уровня в низкий. Программа тогда может быть такой:

```
#define BUTTON_PIN 3 //вывод подключения кнопки
#define LED_RED_PIN 13 //вывод управления красным светодиодом
#define LED_GREEN_PIN 12 //вывод управления зеленым светодиодом
boolean LEDState = true; //состояние светодиода
boolean buttonOldState; //логическая переменная - предыдущее
                        //состояние кнопки
void setup() {
    pinMode(LED_RED_PIN, OUTPUT); //вывод красного - на выход
    pinMode(LED_GREEN_PIN, OUTPUT); //вывод зеленым - на выход
    pinMode(BUTTON_PIN, INPUT_PULLUP); //вывод кнопки на вход
                                       //с подтягивающим резистором
    buttonOldState = digitalRead(BUTTON_PIN); //реальное состояние кнопки
                                       //в момент запуска
}
void loop() {
  boolean buttonIsState = digitalRead(BUTTON_PIN); //текущее состояние кнопки
  //отслеживаем момент смены состояния:
if (buttonOldState && !buttonIsState) { //был перепад из 1 в 0
    delay(10); //ждем 10 миллисекунд и повторяем
    buttonIsState = digitalRead(BUTTON_PIN); //состояние кнопки
    if (!buttonIsState) { //если все еще нулевое значение,
  LEDState = !LEDState;   //то меняем значение переменной LEDState
                          //на противоположное
    }
  }
```

```
if (LEDState) { //если состояние светодиода = true
//то светим зеленым:
    digitalWrite(LED_GREEN_PIN, HIGH);
    digitalWrite(LED_RED_PIN, LOW);
} else {//иначе красным
    digitalWrite(LED_RED_PIN, HIGH);
    digitalWrite(LED_GREEN_PIN, LOW);
}
buttonOldState = buttonIsState; //запоминаем к следующему разу
} //end loop
```

Здесь мы вводим логические переменные, которые отражают состояние кнопки (локальная `buttonIsState` — текущее, и глобальная `buttonOldState` — предыдущее). Сложное условие `(buttonOldState && !buttonIsState)` отражает возникновение ситуации, когда состояние меняется именно с высокого уровня в низкий. Если низкий уровень не изменился через 10 мс, то меняем состояние светодиода на противоположное через переключение логической переменной `LEDState`. Изменение производится отдельно в конце цикла в соответствии со значением этой переменной.

Переделать приведенный выше алгоритм на отпускание несложно. Для этого нужно только изменить логику условия для реакции на перепад не из единицы в ноль, а наоборот, из нуля в единицу. Исправленный алгоритм будет выглядеть так:

```
. . . . .
if (!buttonOldState && buttonIsState) { //был перепад из 0 в 1
    delay(10); //ждем 10 миллисекунд и повторяем
    buttonIsState = digitalRead(BUTTON_PIN);//состояние кнопки
    if (buttonIsState) { //если все еще единичное значение
  LEDState = !LEDState; //меняем состояние светодиода
//на противоположное
    }
  }
```

Попробуем на радостях решить задачку посложнее — использовать в применении к Arduino методы работы с прерываниями, которые мы уже изучали в предыдущей главе.

Правильная мигалка на Ardiuno

Здесь попробуем решить стандартную задачу из «песочницы» — помигать светодиодом. Но не традиционным простейшим методом — мы здесь воспользуемся тем, что уже довольно много знаем про контроллеры Atmel, и сделаем «правильную» процедуру. Отличается наш таймер от обычных тем, что однажды включенный, он будет работать совершенно независимо от основной программы до тех пор, пока мы его сами не выключим.

Вы правильно догадались, что для этого придется использовать прерывания таймера. У Arduino один восьмиразрядный таймер (Timer0) занят в функциях `delay()`, `millis()` и `micros()`, потому мы его трогать не будем, а привлечем к делу Timer2.

Пример очень хорошо иллюстрирует, как можно в Arduino IDE перемешивать ассемблерный код с высокоуровневым.

Наш таймер рассчитан два светодиода (красный и зеленый), подключенные к портам D18 и D19. На плате Arduino эти порты обозначаются, как A4 и A5, что указывает на возможность их использования в качестве аналоговых входов АЦП, однако это вполне обычные цифровые выводы портов (номер 4 и 5 порта C). Мы именно эти выводы подключили здесь потому, что они обычно используются только, как выводы интерфейса TWI (I^2C), таким образом наш таймер можно будет безболезненно подключать почти к любой программе, где этот интерфейс не используется, без перенумерации выводов. При необходимости освободить эти выводы, в программе нужно поменять только значения переменных `ledRedPin` и `ledGreenPin` в секции объявлений переменных (см. текст программы далее) — отсюда понятно, почему всегда следует заменять конкретные номера на мнемонические обозначения.

Схему мы рисовать не будем ввиду ее крайней простоты. Код демонстрационного примера следующий:

```
const byte ledRedPin = 19;  //Led красный
const byte ledGreenPin = 18;  //Led зеленый
byte countT=0;//счетчик для таймера
unsigned char flagKey=0; //вспомогательная переменная - флаг
void setup() {
  pinMode (ledRedPin, OUTPUT);
  pinMode (ledGreenPin, OUTPUT);
}

ISR(TIMER2_OVF_vect) //начало прерывания
{//мигалка
countT--;
if (countT!=0) return;
if (flagKey != 0)
{digitalWrite(ledRedPin,LOW);
 digitalWrite(ledGreenPin, !digitalRead(ledGreenPin)); //включаем/выключаем
                                                                    зеленый
} else
{digitalWrite(ledGreenPin,LOW);
 digitalWrite(ledRedPin, !digitalRead(ledRedPin)); //включаем/выключаем красный
}
} //end прерывание

void Timer2start ()
{ //пуск Timer2
TCCR2B = (1<<CS22)|(0<<CS21);
//Timer 2 счет 1/256
    bitSet(TIMSK2, TOIE2);//разрешаем прерывания overflow
}
```

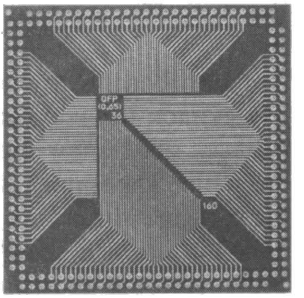
```
void Timer2stop() //стоп Timer2
{bitClear(TIMSK2, TOIE2);//запрещаем прерывания overflow
  digitalWrite(ledRedPin,LOW); //оба светодиода погашены
  digitalWrite(ledGreenPin,LOW);
 }

void loop() { //основной цикл программы
  Timer2start(); //запускаем Timer2
  flagKey=0; //мигаем красным 5 сек
  delay(5000);
  flagKey=1; //мигаем зеленым 5 сек
  delay(5000);
  Timer2stop();//останавливаем - оба темные
  delay(5000); //на 5 сек
}
```

Здесь восьмиразрядный `Timer2` включается на переполнение со входной частотой 1/256 от тактовой. Переполнение наступает, когда таймер отсчитает 256 входных импульсов. Дополнительно мы вводим переменную `countT` типа `byte` (если ориентироваться на обычный язык C, то тип может быть `unsigned char` или `uint8_t` — то же самое, что `byte`). Переменная инициализируется нулем, в первом же прерывании она окажется равной 0 – 1 = 255 (тип `byte` не может быть больше 255), и затем в каждом прерывании уменьшается на 1, пока опять не дойдет до нуля (в следующем прерывании она опять сразу окажется равной 255). В момент равенства нулю мы переключаем светодиод из зажженного состояния в погашенное или наоборот.

Частота переключений окажется равной почти 1 герцу (16 МГц, трижды деленное на 256). Частоту можно увеличить в два раза, если в установках `TCCR2B` вместо `CS21` поставить `CS20`, или в четыре, если оставить только один `CS22`, меняя тем самым входную частоту таймера на 1/128 или, соответственно, 1/64 от тактовой (другие сочетания установочных битов см. в описании регистра `TCCR2B` в документации на ATmega328, ссылка на которую имеется на всех официальных сайтах Arduino).

Согласно программе, на 5 секунд включится красный светодиод, затем на 5 секунд зеленый, потом на 5 секунд таймер остановится, и все начнется заново. Вместо двух светодиодов или одного трехвыводного двухцветного, можно подключить двухцветный с двумя выводами, если оба вывода подключить к портам 18 и 19. Специально для этого случая перед заданием цвета вывод, отвечающий за противоположный цвет, устанавливается в нулевой уровень. Если светодиод нужен только один, то из программы убирается все, что относится к переменой `flagKey` и установкам второго порта. Например, если оставляем только красный светодиод на порту 19, то все строки, где присутствует `ledGreenPin`, удаляются, а в прерывании после проверки счетчика оказывается только одна строка `digitalWrite(ledRedPin, !digitalRead(ledRedPin))`.

А теперь перейдем к более подробному изложению способов использования различных компонентов Arduino — именно в их многообразии заключено все удобство этой платформы.



ГЛАВА 21

Компоненты для Arduino

Как на Arduino делать устройства лучше фирменных

> Вдруг из темноты выступила какая-то фигура, очертания которой показались д'Артаньяну знакомыми, и привычный его слуху голос сказал:
>
> — Я принес ваш плащ, сударь: сегодня прохладный вечер.
>
> *А. Дюма. «Три мушкетера»*

В этой главе мы рассмотрим ряд компонентов — модулей для Arduino, которые выпускаются различными фирмами для упрощения и удешевления разработки и сборки готовых изделий. Отбор компонентов, по традиции, сложившейся в предыдущих изданиях, будет подчинен основной цели создания домашней метеостанции. Многообразие компонентов таково, что от рассмотрения законченного проекта автор отказался — проще предоставить читателю выбор и рекомендации по сборке отдельных модулей в единое целое. Кроме того, опыт показал, что готовый достаточно крупный проект воспроизвести «один в один» не только сложно, но и не всегда хочется: вы просто чего-то не достанете, или можете предпочесть простоту и дешевизну красоте и функциональности, или, наоборот, захотите ориентироваться на самые дешевые решения с целью реализовать проект попроще, исключив часть компонентов. Вот выбор для подобных вариантов и предоставляется в этой главе.

Здесь мы рассмотрим три основных направления: датчики различных величин, дисплеи для отображения информации и модули для беспроводной передачи данных по радиоканалу, а также некоторые сопутствующие темы (например, тему энергосбережения). Очевидно, что рассматриваемые компоненты можно задействовать далеко не только в измерениях метеорологических параметров — это всего лишь хороший иллюстрационный материал по их использованию. Тем не менее, нельзя не заметить, что именно домашняя метеостанция — один из самых оправданных радиолюбительских проектов с практической точки зрения. Как мы уже говорили, бытовые метеостанции, имеющиеся в продаже, не выдерживают никакой критики — ни с точки зрения удобства пользования и дизайна, ни с точки зрения метрологических качеств. Через руки автора этих строк прошло не менее десятка «фирменных» моделей бытовых метеостанций, и ни у одной из них работу нельзя было признать удовлетворительной.

Arduino позволяет подойти к конструированию приборов «по-взрослому» — не делая скидок на любительское происхождение при выборе функциональности. Поскольку мы станем ориентироваться на ассортимент готовых модулей, то результат окажется, как минимум, не дороже тех убогих произведений, которыми переполнены интернет-магазины домашней техники. При этом мы легко сможем реализовать дополнительные функции, которые либо присущи очень дорогим моделям, либо отсутствуют в промышленных образцах вовсе. Самое трудное, как всегда в таких случаях, — оформление конечного результата так, чтобы его было не стыдно повесить на стенку, но тут все в ваших руках. Еще и по этой причине в данной главе я отказался от описания законченной конструкции — читатель уже обладает достаточной квалификацией, чтобы самостоятельно собрать готовый прибор из тех компонентов, которые он предпочтет, придав ему внешний вид по собственному вкусу и карману.

Техническое задание

Перед тем как приступать к проектированию, надо точно определить, к чему мы стремимся. Давайте сформулируем, что должна «уметь» и из каких основных узлов состоять домашняя метеостанция.

- **Главный модуль** — измерение внутренней температуры (в месте установки), влажности и атмосферного давления. Питание от сети.
- **Выносной датчик внешней (уличной) температуры и влажности** — связь с главным модулем по радиоканалу, питание от батарейки (а значит, функции энергосбережения). Связь должна надежно работать как минимум через оконный стеклопакет, а лучше — через бревенчатую (в идеале — кирпичную или бетонную) стенку, на расстоянии не менее 5–7 метров.
- **Часы реального времени с календарем** (в составе главного модуля) — должны иметь автономное питание и возможность автоматической/полуавтоматической коррекции хода.
- **Дисплей главного модуля** — внешняя температура/влажность, внутренняя температура/влажность, атмосферное давление, время, дата, день недели. Он может быть как ЖК-типа (проще работать и цена гораздо меньше), так и более эстетично выглядящим, но и существенно более дорогим светящимся экраном на основе OLED. Для полноты картины также покажем, как можно реализовать функции отображения некоторых параметров наиболее простым способом — на семисегментных и малогабаритных матричных индикаторах.
- **Метрологические требования** — при установке выносного датчика рядом с главным модулем расхождение показаний по температуре желательно не более 0,5 °С, по влажности — не более 2%. Абсолютное значение ошибки измерения температуры вблизи нуля градусов — не более 0,5 °С. Погрешность влажности не нормируем — проверить ее мы все равно не сможем, главное, чтобы показатели выглядели правдоподобно и совпадали во всем диапазоне у разных датчиков. Отметим, что погрешность барометра также можно не нормировать — его

показания в любом случае придется подгонять «по месту» по причинам, о которых далее.

О выборе компонентов

Прежде чем выбирать компоненты, следует внимательно изучить каталоги основных продавцов, торгующих электронной продукцией необходимого направления (Iarduino.ru, «Чип-Дип», «Амперка», «Терраэлектроника» и др.). Иначе вы можете упереться в то, что выбранного компонента в данный момент нет в продаже, или производитель выпускает обновленную версию, или имеется похожая, но с другой разводкой выводов и т. п. Можно покупать и на Aliexpress, где дешевле и «все есть», но документация и точные характеристики там, как правило, не прилагаются, потому обязательно сначала узнайте точное наименование того компонента, который вам нужен. Проверьте также, доступны ли соответствующие библиотеки, причем, желательно, не только те, что от торгующей организации, но и независимая альтернатива. Все это необходимо выяснить, как минимум, до составления полной схемы, иначе можно потратить кучу времени и денег совершенно зря.

Независимо от реализации управления готовым прибором, вам необходимо иметь базовую Arduino Uno версии R3, причем оригинального дизайна, т. е. с USB-чипом от Atmel или совместимым (FTDI232), а также контроллером в DIP-корпусе, установленным на панельку (как показано на рис. 20.1 из *главы 20*). Этот контроллер пригодится, во-первых, для макетирования узлов, и во-вторых, как программатор в случае, если вы с целью снижения потребления захотите использовать его отдельно от платы (см. далее *разд. «О режиме энергосбережения, Watchdog-таймере и питании метеостанции»*). Отдельный контроллер можно заменить на плату Arduino Mini (удалив предварительно с нее светодиод по питанию).

В основном модуле для управления дисплеями отображения метеорологических параметров целесообразно применить Arduino Nano и, может быть, еще один отдельный контроллер Nano или Mini для управления часами и календарем. Все зависит от количества дисплеев, которыми приходится управлять: если их больше двух, то лучше их поделить между двумя контроллерами. Во-первых, так меньше нагрузка на каждый из контроллеров и вероятность сбоев, во-вторых, раздельное размещение программ удобнее. Программу для часов вы один раз отладите и, может быть, будете перезагружать раз в год, когда часы «уйдут». А вот программу приема и обработки не исключено, что придется править не раз, подгоняя различные параметры.

Начнем мы с того, что подробнее рассмотрим основной интерфейс, по которому обычно подключаются датчики и некоторые другие компоненты станции — I^2C (TWI).

Интерфейс TWI (I^2C)

Двухпроводной интерфейс TWI (Two-Wire Interface) широко известен также под названием I^2C. Чтобы не путаться в названиях, следует усвоить, что TWI и I^2C синонимы: первое название чаще используется в применении к AVR-контроллерам

(в том числе и Arduino), а второе — к различной периферии. TWI широко используется для подключения различных датчиков и удобен тем, что у него на одни и те же два сигнальных провода теоретически можно «нанизать» сколько угодно разных устройств (на самом деле не более 127 штук). На практике по некоторым причинам (см. *«Подробности»* далее) число устройств обычно ограничено тремя-четырьмя, но и этого для обычных нужд более чем достаточно.

Конечно, на самом деле проводников, соединяющих TWI-устройства, не два, а, как минимум, три или даже четыре: к двум сигнальным добавляются линии «земли» (обязательно!) и питания (необязательно — питание у каждого устройства может быть свое). Сигнальные линии TWI чаще всего носят название SDA (Serial DAta) и SCL (Serial CLock), но у некоторых устройств могут называться иначе, например, просто Data и CLK. Как следует из названий, по первой из них передаются данные, по второй — тактовые импульсы.

В AVR-контроллерах имеется встроенный (аппаратный) порт TWI. Его контакты соответствуют выводам плат Uno, Nano и Mini A4 (SDA) и A5 (SCL). В последних реализациях платы Uno ее разработчики, наконец, прониклись популярностью TWI среди производителей оборудования для Arduino и вывели линии SDA и SCL на отдельные контакты. На рис. 20.1 — это крайние контакты слева в верхнем ряду (их названия на рисунке не видны, они подписаны на разъеме сбоку). Отметим, что эти выводы SDA и SCL просто дублируют контакты A4 и A5, потому TWI-устройства можно подключать к любой из этих групп контактов. Это позволяет подключить к плате Uno сразу два устройства, не прибегая к внешним разветвлениям проводников.

Так как выходы устройств на шине I^2C есть выходы с открытым коллектором, то при подключении TWI-устройств требуется устанавливать «подтягивающие» резисторы R1 и R2 на каждой из линий (рис. 21.1). Достаточно одной пары резисторов на все подключенные устройства. Величина этих резисторов, вообще говоря, должна рассчитываться для каждого конкретного варианта схемы, но на практике огра-

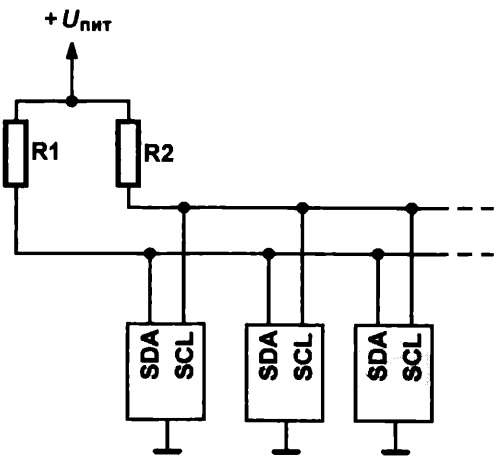


Рис. 21.1. Подключение нескольких устройств к шине I^2C/TWI

ничиваются величинами от 4,7 до 10 кОм. При этом исходят из предположения, что длина проводников, соединяющих устройства, лежит в пределах 10–30 см. Резисторы R1 и R2 обычно уже установлены в модулях устройств, работающих по TWI-протоколу, и в Arduino-схемах самостоятельно заботиться о них не требуется.

Библиотека для TWI (Wire.h) входит в основной комплект среды Arduino и отдельной установки не требует.

Подробности

О подтягивающих резисторах надо заботиться только в том случае, если вы разрабатываете собственную схему на «голых» компонентах, без использования готовых модулей. Однако в этом-то и состоит проблема: минимально допустимое значение сопротивлений R1 и R2 равно 1,5 кОм. И если в готовых модулях стоят резисторы по 4,7 кОм, то они оказываются включенными параллельно, и предельное значение достигается уже при трех-четырех модулях, подключенных к одной шине. Кроме этой неприятности, в огромном «зоопарке» датчиков для Arduino изредка встречаются несовместимые разновидности, которые просто не будут работать, если их подключить к одному порту TWI. В подобных случаях модули можно подключать к другим цифровым выводам Arduino, используя вместо аппаратного TWI его программную имитацию, хотя найти такую альтернативную библиотеку бывает непросто. В любом случае перегружать шину TWI не стоит — если есть альтернатива, то лучше использовать ее.

Заметим также, что при обычном порядке работы с датчиками сигнал SCL для контроллера всегда является выходным, а т. к. у AVR обычный симметричный КМОП-выход (а не с открытым коллектором), то подтягивающий резистор на шине SCL не требуется. Конструкторы модулей его обычно устанавливают (на AVR свет клином не сошелся), но при самостоятельном подключении «голого» датчика его можно не ставить (см. далее о датчике SHT75).

Различают два типа устройств, работающих по протоколу TWI. Они называются, соответственно, Master (ведущий) и Slave (ведомый). На практике ведущим (Master) всегда является контроллер, который посылает команды всем ведомым одновременно. Так как устройств на шине много, а ведущий один, то каждая команда сопровождается индивидуальным адресом ведомого устройства, и реагировать будет только то устройство, которое имеет указанный адрес. Адрес представляет собой двоичное число длиной 7 битов, потому и теоретическое ограничение на количество одновременно подключенных устройств с разными адресами равно 127 (от 1 до 2^7-1).

Адреса устройств обычно присваиваются еще конструкторами при проектировании и различаются для модулей разных типов. Следует учесть, что подключать два одинаковых устройства к одной шине, как правило, нельзя — контроллер их не сможет различить, если только в них не предусмотрен специальный механизм, который позволяет менять адрес в установленных пределах с помощью перемычек (далее мы познакомимся с этим механизмом на примере I^2C-конвертера для управления дисплеями).

Кроме двухпроводного интерфейса TWI, в различных датчиках употребляется близкий к нему по принципам работы однопроводной интерфейс 1-Wire. Как следует из названия, он для подключения требует всего одного сигнального проводника. Так как аппаратный однопроводной интерфейс в контроллере отсутствует, то

подключать датчик с таким интерфейсом можно к любому цифровому выводу Arduino, достаточно указать номер вывода в программе. Мы встретимся с таким вариантом подключения на примере датчиков температуры и влажности DHT11/DHT22.

Теперь перейдем к рассмотрению отдельных узлов, и начнем с общего обзора датчиков.

Датчики метеорологических параметров

Датчиков таких параметров, как температура, влажность и атмосферное давление, ныне предлагается уйма. Неопытный человек просто потеряется в этом разнообразии, не в силах понять критерии выбора. Сказывается и тотальная безграмотность в метрологических (не путать с метеорологическими!) вопросах даже среди опытных пользователей, тщательно поддерживаемая производителями, которым это крайне выгодно. Можно в описании, например, обещать, что Operating Range датчика для влажности составляет 0–100%, а пользователь потом обнаружит, что выше 87–89% датчик не показывает влажность вовсе, тогда как это основной диапазон для загородных условий на улице вечером и ночью.

В этом разделе мы представляем общий обзор различных датчиков и модулей, с кратким описанием некоторых особенностей применения каждого типа. Практическое использование некоторых из этих датчиков мы впоследствии рассмотрим подробнее, когда будем говорить об отображении информации на дисплеях и других функциях законченной метеостанции.

Датчики температуры и влажности

Для начала укажем на то, что стоит и что не стоит применять и при каких условиях. Во-первых, не нужно ни при каких вариантах использовать в реальных конструкциях датчик DHT11 — он завоевал популярность своей дешевизной, но реально мерять он ничего не может, и индивидуальная градуировка ему не помогает. Вот его родного брата DHT22 вполне можно использовать, по крайней мере, для комнатных условий.

DHT11 традиционно имеет чехол, окрашенный в голубой цвет, а DHT22 — в белый. Способы подключения и готовые библиотеки для этих датчиков одни и те же. Окончательный тип датчика может выбираться при подключении библиотеки в тексте скетча. С учетом этого обстоятельства, все многочисленные примеры использования DHT11, которые можно встретить в Интернете, пригодны для обоих типов датчиков.

Все типы DHT подключаются по однопроводному интерфейсу, что удобно, когда нужно подключить много разных датчиков — они гарантированно не будут мешать друг другу. Схема подключения модуля DHT11/DHT22 настолько проста, что рисовать мы ее не будем — на рис. 21.2 показана разводка выводов для двух вариантов: оформленного модуля на плате (*слева*) и отдельного датчика (*справа*). Как видите, «голый» датчик отличается только отсутствием подтягивающего резистора, кото-

рый придется устанавливать самостоятельно. На плате готового модуля (как и вообще на всех модулях для Arduino) разводка обычно прямо обозначена надписями напротив разъемов.

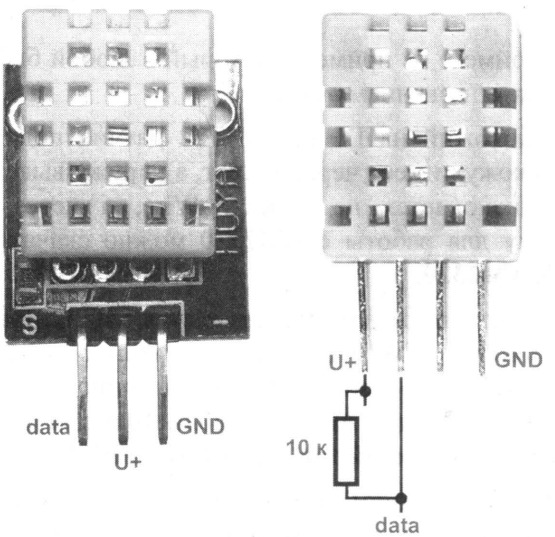


Рис. 21.2. Разводка выводов модуля DHT11/DHT22 на плате (*слева*) и отдельного датчика (*справа*)

Библиотек для работы с этими датчиками тьма. Выбирать стоит те, в которых температура выводится, как минимум, с десятыми градуса после запятой (с ними при попытке подключения DHT11 вы сразу поймете, почему я его не рекомендовал). Укажем на ту, которой пользуюсь сам: это облегченный вариант популярной библиотеки от фирмы Adafruit, в котором спрятано все лишнее. Его можно скачать со странички примера подключения датчика по адресу **http://e-device.in.ua/dht11_arduino**. Называется эта библиотека просто DHT. Напомню, что скачанную библиотеку надо разархивировать в одну из предопределенных папок libraries, после чего Arduino IDE обнаружит библиотеку самостоятельно (подробности см. в *разд. «Программы для Arduino» главы 20*).

Вывод, обозначенный на нашем рисунке как «data», подключается к любому цифровому выводу Arduino, его только следует указать при инициализации библиотеки, примерно так:

```
#include "DHT.h" //подключаем библиотеку
#define DHTTYPE DHT22   //датчик DHT 22
#define DHTPIN 2 //вывод, к которому подключается контакт data
DHT dht(DHTPIN, DHTTYPE); //задаем параметры библиотеки
. . . . .
void setup() {
      . .
     dht.begin(); //инициализация
```

```
//читаем данные:
    int hum = dht.readHumidity(); //влажность сразу целое число
    float t = dht.readTemperature(); //температура
.
}
```

Остальное вы легко поймете из примера, который в любой библиотеке традиционно размещается в папке с названием examples.

В продаже имеется похожий на DHT11/DHT22 датчик под названием AM2320, только его защитный кожух имеет черный цвет, а наружу выведены не три, а четыре контакта. Он подключается не через 1-Wire, а через стандартный TWI-интерфейс. Библиотеку для работы с AM2320 можно скачать по адресу: **https://github.com/thakshak/AM2320**.

Еще один вариант представляют датчики семейства SHT1*x*. Например, SHT10 также работает через интерфейс TWI/I^2C, но его библиотеки традиционно применяют не аппаратный порт TWI, а его программную имитацию, и их можно подключить к двум любым цифровым выводам Arduino. Библиотеку для работы с SHT10 можно скачать отсюда: **https://github.com/practicalarduino/SHT1x**.

Все перечисленные варианты (кроме аутсайдера DHT11) примерно равны друг другу по метрологическим характеристикам, и для комнатных условий можно применять любой из них. Есть еще подобные модули, совмещенные с барометром, но такой универсальный вариант опять же можно рассматривать только для комнаты. Для расширенного уличного диапазона они все одинаково плохи прежде всего потому, что не имеют индивидуальной калибровки, и их показания будут разъезжаться, отличаясь и друг от друга, и от истинных значений на целые градусы и вплоть до десятка единиц влажности. Причем если подправить значения температуры, в общем, не представляет трудностей (см. *главу 13*), то калибровку измерителя влажности воздуха в любительских условиях провести практически невозможно — разве что вам удастся арендовать на время профессиональный измеритель влажности (гигрометр). Приобрести его в собственность нереально потому, что цена его заметно превысит стоимость всего остального оборудования вашей домашней лаборатории.

Заметки на полях

Кроме совмещенных датчиков температуры и влажности в виде модулей, можно, конечно, рассмотреть и подключение обычных датчиков без встроенного цифрового интерфейса. Для температуры вполне подойдет упомянутый в *главе 13* полупроводниковый датчик TMP36, только его надо обязательно калибровать, не полагаясь на фирменные данные, и при желании получить хорошую разрешающую способность придется подбирать значение опорного напряжения АЦП (подробнее см. книгу автора [26]). Впрочем, подключение любого другого аналогового датчика температуры к AVR не представляет трудностей. Есть достаточный ассортимент и датчиков влажности, не стоит только попадаться на очень дешевые решения типа резистивного датчика SYH-2RNC. Реально в серьезных целях можно использовать относительно дорогие датчики фирмы Honeywell HIH-40*xx*, калибруемые на производстве, но их фабричная калибровка все-таки требует уточнений (подобно датчику TMP36) и тем самым делает столь дорогую покупку нецелесообразной.

Если вы хотите иметь все-таки измерительный прибор, а не игрушку, остается только поискать действительно калиброванный датчик с фирменной гарантией. Такие находятся, например, в составе семейства SHT7*x* и называются SHT71 и SHT75 (последний предпочтительнее из-за меньшей погрешности). Стоит такой датчик (рис. 21.3) существенно дороже остальных, и по-хорошему тоже требует индивидуальной калибровки, но и прямо «из коробки» показывает величины, которым хочется верить (и, кстати, через него можно калибровать и обычные датчики, упомянутые ранее). При подключении к контроллеру не забудьте установить подтягивающий резистор 10 кОм между выводами питания и SDA (для SCL, согласно документации, такого резистора не требуется).

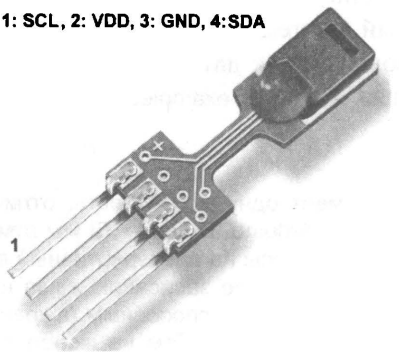


Рис. 21.3. Датчик SHT71/75 и разводка его выводов

Работать с SHT71/75 через библиотеку SHT1*x* нельзя, потому что в ней отсутствуют нужные калибровочные коэффициенты. Фирма Sensirion выпустила для него специальную библиотеку, которую можно найти по адресу: **https://github.com/domhardt/ArduinoLibraries/tree/master/Sensirion**. Она также рассчитана на программную имитацию TWI, потому датчик можно подключать к любым цифровым выводам, указав их в программе.

Подробности

Конструкция SHT71/75 имеет крупный недостаток: металлическая подложка платы отлично проводит тепло, потому датчик обязательно следует целиком выносить за пределы корпуса прибора и соединять со схемой жгутом из как можно более тонких проводников (плоским кабелем или МГТФ-0,35). Иначе наличия одного только «голого» контроллера типа ATmega328 со стабилизатором питания в замкнутом пространстве корпуса достаточно, чтобы через подложку платы подогреть датчик на 2–5 градусов даже в случае, если его головка вынесена наружу.

Барометры

Ассортимент датчиков атмосферного давления гораздо меньше, чем температуры/влажности, и по сути сводится лишь к решениям на основе микросхем BMP180 и ее модификаций (BMP085, BMP280), а также LPS331AP. Последних гораздо меньше, библиотеку к ним найти труднее, а видимых преимуществ они не имеют, потому стоит рассматривать только варианты BMP180. С метрологией там никаких

проблем — датчик и «из коробки» показывает все, как надо. Такой небольшой ассортимент обусловлен сложностью производства чувствительных элементов — датчики изготовляются по технологии МЭМС. Расшифровывается это как «микроэлектромеханические системы» — так называют микросхемы, объединяющие в себе микроэлектронные и микромеханические компоненты.

Библиотеку для модулей на основе BMP180 можно скачать, например, отсюда: **https://github.com/sparkfun/BMP180_Breakout_Arduino_Library**. Подойдут и более простая: **https://github.com/adafruit/Adafruit_BMP085_Unified**, а также много других, отличающихся только удобством вывода. Подключаются датчики давления обычно по стандартному TWI, т. е. параллельно часам и другим подобным датчикам. Все датчики атмосферного давления всегда содержат и встроенный датчик температуры, используемый для температурной компенсации давления, потому при нужде их можно использовать и как датчик этой величины. Примеры использования можно найти, как всегда, в папках examples библиотек.

Заметки на полях

Библиотека от Adafruit имеет одну особенность, отличающую ее от многих других стандартных библиотек для Arduino. В *главе 19* мы отмечали, что в программах для МК бессмысленно тратить ресурсы на альтернативный выход из программы в случае, если какая-то функция привела к ее зависанию из-за неисправности в схеме — устройство все равно окажется неработоспособным (в крайнем случае можно применить метод, описанный в конце *главы 22*). Тем не менее в данной библиотеке функция инициализации `begin()` возвращает значение, свидетельствующее об успешном ее завершении. В примере BMP085test (см. папку examples библиотеки) это используется для вывода сообщения и последующего намеренного зацикливания программы:

```
if (!bmp.begin()) {
Serial.println("Could not find a valid BMP085 sensor, check wiring!");
while (1) {}
}
```

Не иначе, как программу составлял какой-то чересчур грамотный специалист по «большому» программированию. Мы уже говорили, что подобное действие бессмысленно: появление далеких от истины показаний или зависание программы и без того сигнализируют, что с датчиком непорядок. Так что без таких усложнений можно обойтись, просто вызвав фунцию `begin()` без обработки возвращаемого значения.

Библиотеки для барометров, как водится, выдают давление в паскалях, гектопаскалях или в миллибарах (практически равных гектопаскалям). Поэтому при выводе на дисплей их стоит перевести в привычные миллиметры ртутного столба, которые к тому же компактнее миллибаров на один десятичный знак. Учитывая, что 1000 гПа практически точно равны 750 мм рт. ст., получаем формулы перевода:

```
mmHg = int(pressure*0.0075); //для Па
mmHg = int(pressure*0.75); //для гПа и мбар
mmHg = int(pressure*750); //для бар
```

Здесь мы применяем явное преобразование типов — результат умножения переменной `pressure` типа `float` на дробный коэффициент сразу приводим к целому виду типа `int`. Это делается потому, что атмосферное давление выводить с десяты-

ми не имеет смысла, а вывод целого числа компактнее — переводить реальное число в строку куда более громоздкая процедура, как мы увидим далее.

Другие метеорологические датчики

Разумеется, в современной электронике есть готовые решения на все случаи жизни, в том числе имеются они и для метеорологических станций. Для любительской станции, кроме перечисленных в предыдущих разделах, интерес представляют датчики дождя и ветра.

Простейший датчик дождя продается на каждом углу, и его нетрудно соорудить самостоятельно. Он содержит компаратор LM393 и простую платку с сеткой электродов, сопротивление между которыми меняется при увлажнении. Недостаток его очевиден: при прекращении осадков трудно ожидать, что влажность снизится сразу настолько, чтобы поверхность сенсора высохла за приемлемое время (скорее наоборот — если солнце не выглядывает, то после дождя влажность может быть даже выше, чем во время него), и настройка чувствительности тут не поможет. Такой датчик либо после однократного намокания в пасмурную погоду будет показывать дождь сутками, либо упускать момент начала осадков. Другим недостатком его является примитивная конструкция чувствительного элемента с негерметичным разъемом, который сложно уберечь от намокания.

Более интересный вариант датчика дождя представляет собой RG-11, аналогичный по принципу работы тем, что применяются в автомобилях для автоматического включения дворников (см. например, **http://rainsensors.com**). Он основан на принципе отражения инфракрасного света и существенно более дорогой, хотя приобрести его не проблема. Принцип устройства еще одной интересной конструкции приведен на вот этой фотографии: **https://en.wikipedia.org/wiki/File:Interior_tipping_bucket.JPG**. Подобные датчики выдают частотный сигнал, зависящий от скорости наполнения водой углублений на балансире, перекидывающемся с одной стороны на другую. То есть, кроме факта осадков, он показывает еще и их интенсивность, т. е. может быть использован в качестве осадкомера (плювиометра). Подобный готовый датчик приобрести сложнее остальных, и он довольно дорог, но можно попробовать по этому принципу соорудить что-то самостоятельно.

С датчиками ветра сложнее. Для измерения скорости ветра (анемометрии) существует миллион способов, главные из которых такие:

- — термоанемометрический;
- — механический: с пропеллером (точнее, импеллером) или чашечной горизонтальной крыльчаткой (классический чашечный анемометр). Измерение скорости в этих случаях эквивалентно измерению частоты вращения оси, на которой закреплены пропеллер или крыльчатка;
- — а также ультразвуковой, объединяющий измерения скорости и направления.

Для измерения направления способов меньше:

- — упомянутый ультразвуковой;
- — механический флюгер с электронным съемом угла поворота.

Как мы видим, полный датчик ветра, включающий и скорость и направление, может быть создан только в рамках ультразвукового принципа. Реализация его весьма сложна, и только некоторые конструкции профессиональных метеостанций снабжены таким датчиком — остальные обходятся раздельными анемометром и флюгером.

Заметки на полях

А почему, кстати, ультразвуковой датчик скорости ветра так сложен в реализации? В принципе достаточно взять ультразвуковой дальномер (которых выпускается масса для тех же Arduino), разнести приемную и передающую части на некоторое расстояние и при нахождении в потоке воздуха померить разность времен прохождения сигнала «туда и обратно». Практически же подобных конструкций мало по той же причине, почему не существует ультразвуковых рулеток для более-менее точного измерения расстояния. Скорость звука в воздухе подвержена слишком большому количеству случайных факторов — даже чтобы определить расстояние с приемлемой точностью, и тем более на фоне этого шума сложно уверенно отделить полезный сигнал от сноса звуковой волны ветром.

Теоретически методы такого выделения, конечно, секрета не представляют, но совокупность конструктивных, технологических, схемотехнических и математических приемов, позволяющую сделать это с достаточной точностью, представляет собой тщательно оберегаемое ноу-хау каждой фирмы, выпускающей такие датчики и практически исключает возможность повторения этого способа на любительском уровне. Автор этих строк убил пару месяцев и кучу денег на такие попытки и, когда осознал необходимость консультации у грамотного акустика и расчета специальной конструкции акустической антенны, оставил это занятие. Но если у кого получится, буду только рад, пишите.

Что же касается раздельных конструкций, то фирменные датчики также для любителей почти недоступны. После долгих поисков в одном из западных интернет-магазинов был обнаружен термоанемометр для Arduino за приемлемые деньги, но в наших торговых сетях он отсутствует. Еще сложнее с флюгером, потому что это большей частью не электронная, а механическая конструкция, и датчики получаются неоправданно дорогими.

Так что если хотите снабдить свой загородный дом подобными прибамбасами, придется изобретать их на коленке. Один из вариантов конструкции такой метеостанции читатель может узнать из статьи [27].

Особенности калибровки цифровых датчиков

На особенностях проведения калибровки обычных аналоговых датчиков мы уже останавливались в *главе 13* и других главах. Фактически она сводится к построению кривой или таблицы зависимости выходного сигнала от входного. В случае полностью аналогового устройства в соответствии с полученными данными изменяются параметры измерительной схемы, в случае цифрового — параметры входного усилителя и коэффициенты пересчета в цифру. В случае применения контроллеров, в том числе и платформы Arduino, где настройки аналогово-цифровых компонентов скрыты от пользователя, все это сводится к выбору режимов АЦП и программному воспроизведению формулы пересчета из кодов АЦП в реальные

физические величины в том или ином формате (см. пример далее в *разд. «Простой электронный термометр»* этой главы).

Полностью цифровые датчики уже выдают готовые физические величины, которые, однако, чаще всего требуется корректировать «по месту». Надо учесть, что имеются датчики, в которых эти готовые физические величины есть результат пересчета выходного сигнала такими же формулами пересчета «сырых» данных в нужную величину, только реализованный в фирменных библиотеках (пример — датчики серии SHT*x*). Есть и такие, в которых калибровочная формула «зашита» в сам датчик, потому до «сырых» данных все равно не добраться. И очень редко встречаются разновидности, которые предоставляют разработчику возможность оперировать и пересчитанными, и, при желании что-то подправить, исходными данными встроенного АЦП (пример — некоторые магнитометрические компасы или датчики положения в пространстве).

В этих случаях не стоит копаться в библиотеках и портить их лишним вмешательством. Проще принять готовые данные датчика за исходную величину, и составить формулу пересчета их с учетом поправки по опытным данным. Это делается точно так же, как описано в *разд. «Регрессия и метод наименьших квадратов» главы 13*, с помощью, например, программы автора RegrStat (**http://revich.lib.ru**, раздел «Программы»). При этом составляется табличка, где показания датчика принимаются за независимую переменную x, а показания образцового прибора — за y, и по ней рассчитываются коэффициенты корректирующего уравнения.

Цифровые датчики стараются делать уже линеаризованные, потому для такого пересчета обычно достаточно простого линейного уравнения с двумя коэффициентами: нулевой точкой A0 и крутизной преобразования A1, т. е. вида $y = A0 + A1 \cdot x$. Иногда его бывает удобнее представить в виде $y = a1(x + a0)$, где a1 равно полученному из расчета по методу наименьших квадратов A1, а для определения a0 значение A0 надо поделить на A1. Но здесь нет особых проблем для реализации и более сложных зависимостей, включая и зависимости коэффициентов уравнения от третьего фактора — скажем, для температурных изменений коэффициентов пересчета влажности.

Только следует учесть, что такой вторичный пересчет данных, уже один раз пересчитанных из исходных данных датчика, умножает все ошибки, и статистическая погрешность может существенно увеличиваться. Поэтому при возможности нужно стараться не нагромождать одно калибровочное уравнение на другое, а определять исходные «сырые» данные с датчика и уже по ним составлять новый исправленный алгоритм пересчета. Если имеется исходное уравнение пересчета, то такие «сырые» данные можно определить обратным пересчетом из показаний датчика. Но в случае готовых цифровых датчиков это не всегда возможно, потому для измерителей, от которых ждут заведомо достоверных показаний (профессиональных научных, для нужд производства или образцовых для поверки других датчиков), готовые цифровые модули применять не следует.

Часы

С часами (RTC, real time clock — часы реального времени), наоборот, все просто так, что проще некуда. Даже единственный серьезный вопрос, возникающий у любителей при попытке прочесть значения из микросхемы RTC — касательно обращения с упакованным BCD-кодом — в Arduino-библиотеках решен прозрачно для пользователя, и перед ним не возникает вовсе. Более того, даже самую сложную задачу начальной установки часов здесь удается решить таким образом, что пользователю совсем не требуется вникать в вопросы представления времени в различных электронных системах (см. далее).

Изначально были самые простые часы бывшей фирмы Dallas (ныне подразделение Maxim) под названием DS1307. Они идеально приспособлены к работе с контроллерами вообще и Arduino в частности: имеют двухпроводной интерфейс I^2C и почти никаких лишних функций, способны идти от резервной литиевой «таблетки» лет десять до полного ее истощения. Несмотря на то, что DS1307 уже приближается к своему тридцатилетнему юбилею, она до сих пор востребована и, по крайней мере для Arduino, стала этаким стандартом, на который равняются все остальные.

Есть модули RTC более современные. При прочих равных некоторые из них следует предпочесть стандартному модулю DS1307, тем более что основные функции у них полностью совпадают, и потому для стандартных действий не нужны специальные библиотеки. Прежде всего нужно упомянуть микросхему DS3231, модули на основе которой имеют точность более ±1 минуты в год. Причем, что интересно, при практически той же стоимости, что и обычная DS1307, которая при неудачном экземпляре кварца и неаккуратной разводке платы способна на «уход» до минут в сутки.

Не забудем еще, что микросхема DS1307, как и большинство более современных, имеет отдельный управляемый выход SQW точной частоты, которая может программироваться в диапазоне 1–8192 Гц (разумеется, для использования этой функции нужна специальная библиотека и модуль, где вывод SQW специально выведен наружу).

Остановимся еще на вопросе установки часов и коррекции хода — вопрос, который по какому-то недоразумению не решен в большинстве бытовых приборов. В устройствах, ориентированных на практическое применение, заставлять пользователя жать на кнопочки после каждого сбоя в питании — признак либо лени, либо крайней безграмотности разработчика. Если еще можно понять разработчиков часов, встроенных в газовую плиту, то в случае мобильников ручная коррекция выглядит безобразно: сотовые телефоны по своей природе работают в среде точного времени, и вопрос автоматической его коррекции при желании решается с полпинка даже в самых дешевых конструкциях.

Автоматическая коррекция может проводиться несколькими способами. Существует модуль Arduino DCF77 radio clock receiver, ориентированный на прием радиосигналов служб точного времени (известных как DCF77, информация о них есть на официальном сайте **arduino.cc**). Способ имеет тот недостаток, что работает не везде, — так, модули, ориентированные на немецкий передатчик во Франкфурте-

на-Майне, глохнут уже километрах в ста к востоку от Москвы. Наиболее приемлемым методом для Arduino будет использование приемников GPS, но ставить отдельный (дорогой, весьма потребляющий и работающий не везде) спутниковый модуль только для этой цели, конечно, нецелесообразно.

Потому коррекцию здесь можно делать полуавтоматически, через подключение к компьютеру. Обновить время, раз в полгода подключив станцию к любому компьютеру, совсем несложно, тем более что никаких дополнительных аппаратных средств для этого не понадобится. При использовании большей части обычных библиотек RTC для этой цели нужна программа со стороны компьютера (а в библиотеке — процедура, которая способна с ней взаимодействовать), но есть библиотеки, где этот вопрос решается иначе. Так, библиотека RTClib предоставляет специальную функцию `adjust()`, входными параметрами которой служат значения даты и времени, извлекаемые из компьютера в момент компиляции скетча. Помещается она внутри функции `setup()`, и ее использование выглядит следующим образом:

```
. .   . .
RTC.adjust(DateTime(__DATE__, __TIME__));
 if (! RTC.isrunning()) {
 Serial.print("RTC NOT running!");
} else <все нормально, продолжаем программу>
```

Вот здесь проверка на корректность запуска часов выглядит вполне к месту: часы могут не запуститься, например, от того, что села резервная батарейка, и мы никогда об этой причине не узнаем, просто разглядывая плату. Разумеется, если последовательный порт в программе не используется, обрабатывать ошибку следует иным образом.

Но пользоваться функцией `adjust()` следует только один раз, иначе время будет устанавливаться на одну и ту же величину при каждом включении контроллера. Потому поступают так: функцию `adjust()` вставляют в процедуру `setup()` скетча и оставляют закомментированной. Если после загрузки программы часы не запускаются или выявляются какие-то несуразности в выдаваемых значениях времени, то функцию раскомментируют и загружают программу повторно. Затем функцию нужно снова закомментировать и опять загрузить программу. Точно так же следует поступать при коррекции текущего значения времени. Для проведения раз в полгода-год процедура не очень сложная, зато не перегружает программу и не требует лишних сущностей.

Простейшие дисплеи

Дисплеем называется устройство, на которое выводится информация для непосредственного восприятия человеком. Как мы знаем из *главы 7*, дисплеи различаются технологией изготовления: самостоятельно светящиеся матрицы светодиодов или жидкокристаллические матрицы, которые требуют отдельной подсветки в темноте.

Светодиодные называются сокращением LED (Light Emission Diode), а жидкокристаллические LCD (Liquid Crystal Display, ЖК-дисплей).

К LED-дисплеям в том числе относятся и уже хорошо знакомые нам цифровые сегментные индикаторы (они рассматривались в *главах 7, 14* и *17*). В этом разделе мы познакомимся с наборами таких цифровых LED-индикаторов, объединенных в дисплей, состоящий из четырех цифр и управляющийся специальным встроенным контроллером.

Заметки на полях

Как мы увидим, с помощью контроллеров управляться с семисегментными индикаторами оказывается намного проще, и вот почему. Для любительской схемотехники, основанной на микросхемах малой и средней степени интеграции, семисегментные индикаторы находятся практически на пределе достижимого порога сложности — взгляните на рис. 17.9, где показана схема цифрового термометра с четырьмя разрядами. Даже такое количество соединений развести вручную затруднительно, а ведь матричные дисплеи могут содержать намного больше соединений. И то мы здесь использовали микросхему средней степени интеграции 572ПВ2, в которой функции АЦП и драйвера-преобразователя двоичного кода в код сразу для четырех семисегментных индикаторов объединены в одном корпусе. Представьте себе схему, в которой все это реализовано на отдельных микросхемах с выполнением преобразования интервала времени (выхода АЦП двойного интегрирования, описанного в *главе 17*) в двоичный код на счетчиках с временным хранением результата на статических триггерах, и дешифратором его в код управления четырехразрядным семисегментным индикатором — там одних цифровых микросхем потребуется десятка два.

Можете сравнить все это со схемой точно такого же по функциональности термометра (приведенной в этой главе далее, см. *разд. «Простой электронный термометр»*), построенного на контроллере с управлением дисплеем всего по двум проводникам, и содержащую ровно три компонента (не считая источника питания). Для контроллерной схемотехники дисплеи на основе семисегментных индикаторов, наоборот, представляют простейший случай. С помощью контроллера можно столь же просто управлять и куда более сложными конструкциями дисплеев — все нужные соединения сделаны за вас в готовом модуле дисплея, а трудности управления переносятся в программную область и обычно скрыты за интерфейсом готовых библиотек. Далее мы рассмотрим несколько вариантов таких конструкций на основе разновидностей дисплеев, доступных для любительского творчества.

После такого простейшего варианта мы перейдем к более сложным и имеющим больше возможностей. Сначала мы рассмотрим для примера управление ЖК-матрицей 128×64 точки. Что же касается светодиодных аналогов, то с ними сложнее — это относительно недавний продукт, и управление ими еще не успело устояться (особенно это касается воспроизведения кириллицы). Из более-менее традиционных в продаже имеются светодиодные матрицы размерами 5×7 или 8×8 точек. Из-за сложности управления и относительной дороговизны они не получили распространения в радиолюбительской среде, но тем не менее мы рассмотрим далее управление подобной матрицей 5×7 для вывода отдельных знаков. В принципе, из таких матриц можно составлять и достаточно крупные панели, но они будут слишком велики по физическим размерам, потому годится такое решение только для настенных панелей и требует специальных средств по управлению ими, потому мы его не рассматриваем.

А небольшие по физическому размеру строчные и графические светодиодные дисплеи, как мы упоминали в *главе 7*, производят на основе органических светодиодов (OLED). Управляются дисплеи на основе OLED теми же способами, что и ЖК-разновидности, и программные библиотеки для них часто одинаковые, составленные на основе стандартной LiqudCrystal, входящей в комплект Arduino. Потому рассмотрение ЖК-версий мы опустим и сразу перейдем к OLED-вариантам в их строчной и графической разновидностях.

Но начнем, как и говорили, с простейшего случая — управления наборами семисегментных индикаторов.

Подключение цифрового 4-разрядного дисплея к Arduino

Начнем с того, что цифровые дисплеи на основе семисегментных индикаторов выпускаются во множестве вариантов. Их всех объединяют похожие принципы работы, основанные на динамической индикации с помощью сдвиговых регистров. Наиболее часто в продаже встречаются разновидности на основе одного или двух собственно сдвиговых регистров 74HC595 или на основе специальной микросхемы-драйвера TM1637. Кроме того, инициативные китайские товарищи наплодили кучу вариантов, в которых часть функций отсутствует: встречаются дисплеи с присутствующими на плате, но отключенными десятичными точками, отключенным разделительным двоеточием и т. п. Все указанные разновидности могут быть совершенно неотличимы друг от друга по внешнему виду, потому необходимо внимательно смотреть, что именно вы приобретаете.

Мы будем рассматривать наиболее полный вариант (т. е. с задействованными десятичными точками и наличием работающего двоеточия) на основе микросхемы TM1637 — обращение с ней проще и занимает меньше ресурсов контроллера, чем работа с «голым» сдвиговым регистром. Изображен подобный дисплей на рис. 21.4. На том же рисунке на всякий случай показана разводка выводов для него, но нужно привыкать, что обычно она показана прямо на плате.

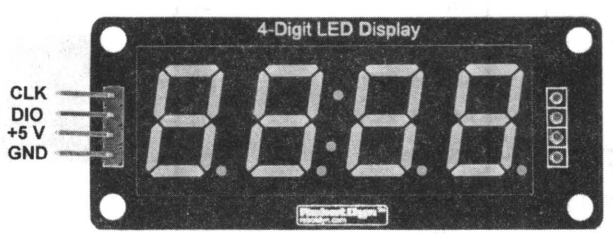


Рис. 21.4. Цифровой 4-разрядный дисплей на основе семисегментных индикаторов

Из-за равных промежутков между разрядами такой дисплей одинаково годится для часов и для обычных датчиков, в последнем случае двоеточие просто не задействуется. Всего четыре знакоместа, надо сказать, сильно ограничивает возможности: в случае отрицательной температуры, например, на цифры остается всего три раз-

ряда и значок градуса уже демонстрировать негде. Но зато все очень просто и без особых проблем, если вы не будете придираться к частностям. А мы придираться будем, но, как вы увидите далее, эти частности успешно преодолеем.

Для работы с такими дисплеями чаще всего употребляют библиотеку, которая по названию управляющей микросхемы-драйвера так и называется TM1637. Оригинал этой библиотеки можно скачать, например, по адресу: **http://wiki.seeedstudio.com/wiki/File:DigitalTube.zip**. Кроме цифр, библиотека выводит буквы A-F (коды 10–15), разделительное двоеточие, десятичную точку в каждой позиции и может очищать позицию при выводе кода 127. Если хотите демонстрировать значок градуса (например, для комнатного термометра), то подправленный вариант можно скачать с сайта автора по адресу: **http://revich.lib.ru/AVR/TM1637.zip**, где значок градуса выводится вместо буквы «A», во всем остальном они идентичны. Учтите, что на работу с этой библиотекой проверены только самые распространенные типы 4-разрядных дисплеев на основе TM1637, но не исключено, что другие конструкции могут потребовать своих библиотек — стандартов здесь никаких не существует.

Часы на 4-разрядном дисплее

Начнем с часов. Подключим к любой из основных плат Arduino модуль часов на основе DS1307 или ее аналогов и 4-разрядный дисплей так, как показано на рис. 21.5.

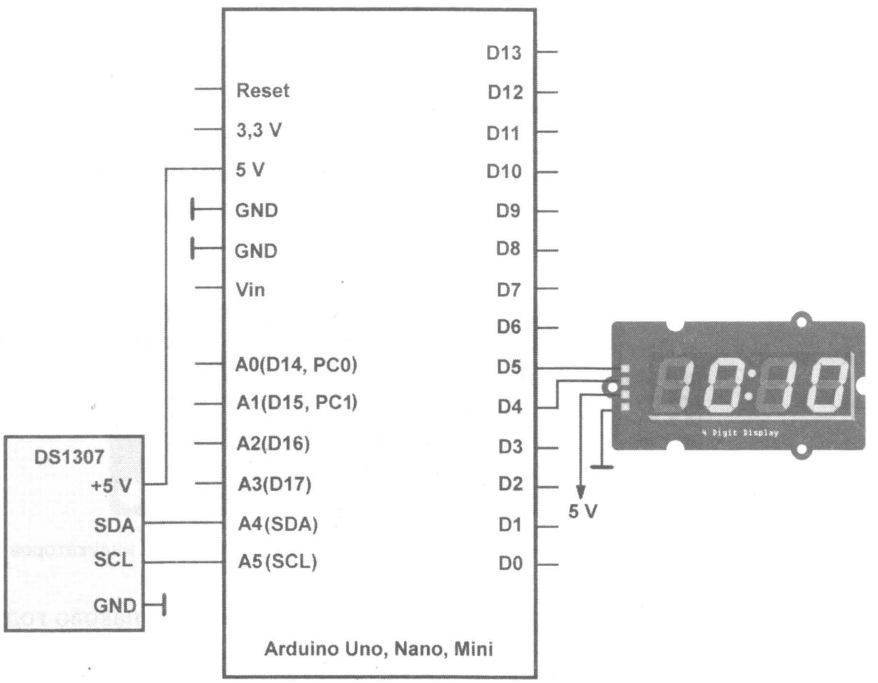


Рис. 21.5. Часы на 4-разрядном дисплее

В установочной части скетча объявим все библиотеки и необходимые переменные:

```
#include <Wire.h> //библиотека для TWI
#include <RTClib.h> //библиотека для часов
#include <TM1637.h> //библиотека для дисплея
#define CLK 5 //тактовый вывод дисплея к выводу 5
#define DIO 4 //вывод данных дисплея к выводу 4
TM1637 disp4(CLK,DIO); //запускаем библиотеку для индикатора
RTC_DS1307 RTC; //запускаем библиотеку для работы с DS1307
DateTime clock ; //массив значений типа DateTime
byte old_second; //сохраненное значение секунд
int8_t values[4] = {14,14,14,14}; //массив для индикатора
```

Байтовый массив `values[]` необходим для поразрядного вывода цифр в нужные позиции дисплея. Вывод целого числа на такой дисплей можно, конечно, организовать и напрямую, но формат вывода при этом устанавливается автоматически, а нам необходимо выводить не только значения часов:минут в нужные позиции, но и заполнить пустые разряды ведущими нулями (по образцу 01:02, как это принято при демонстрации времени). Число 14, которым заполняются разряды по умолчанию, в библиотеке TM1637 означает вывод буквы «E».

В секции `setup()`, кроме инициализации библиотек, мы воспроизведем описанную ранее функцию установки часов из компьютера и закомментируем ее (подробнее см. *разд. «Часы»* в этой главе ранее). При неустановленных часах будет возникать ошибка, которая приведет к индикации буквы «E» во всех разрядах и последующему зацикливанию программы (оператор `while(1)` есть распространенный прием организации бесконечного цикла без возможности выхода). Если ошибки нет, программа обновляет текущее время и выводит его на дисплей:

```
void setup() {
  disp4.init();//инициализация библиотеки дисплея
  disp4.clearDisplay(); //очистка дисплея
  disp4.set(BRIGHT_TYPICAL); //устанавливаем среднюю яркость
  Wire.begin(); //библиотека для TWI
  RTC.begin(); //часы
//строка для настройки времени и даты из компьютера:
//RTC.adjust(DateTime(__DATE__, __TIME__));
 if (! RTC.isrunning()) {
   disp4.display(values); //выводим "EEEE" при ошибке
  while (1); //зацикливаем программу при ошибке
  } else { //если ошибки нет
    clock = RTC.now(); //обновляем время из часов
   display_time (); //и выводим текущее время
 }
}// end setup
```

Отметьте, что библиотека TM1637 имеет функцию `set()`, которая позволяет управлять яркостью свечения индикаторов. Всего предлагается 7 градаций, но практическое значение имеют лишь три. Создатели библиотеки TM1637 предусмотрели для

этих значений константы, которые можно подставлять в функцию `set()` вместо чисел: `BRIGHT_TYPICAL` (2), `BRIGHT_DARKEST` (0), `BRIGHTEST` (7). Подключив к любому аналоговому порту фоторезистор (см. *разд. «Датчик освещенности» главы 22*), с помощью этой функции можно реализовать удобную функцию изменения яркости индикаторов в зависимости от внешнего освещения.

Далее напишем отдельную процедуру вывода времени:

```
void display_time () {
  if (!clock.second()%10){ //каждую десятую секунду
  values[0]=clock.day()/10; //старшая цифра даты
  values[1]=clock.day()%10; //младшая цифра даты
  values[2]=clock.month()/10; //старшая цифра месяца
  values[3]=clock.month()%10; //младшая цифра месяца
    } else { //остальное время:
  values[0]=clock.hour()/10; //старшая цифра часов
  values[1]=clock.hour()%10; //младшая цифра часов
  values[2]=clock.minute()/10; //старшая цифра минут
  values[3]=clock.minute()%10; //младшая цифра минут
}
  if (clock.second()%2)
    disp4.point(POINT_ON); else disp4.point(POINT_OFF); //мигаем
  disp4.display(values); //вывод массива в дисплей
  old_second=clock.second();
} //end display_time
```

Значок `%`, который здесь неоднократно повторяется, означает операцию, которая в математике (и нормальных языках программирования тоже) называется `mod` — определение остатка от деления. То есть оператор `clock.second()%2` будет принимать значение 0 («ложь»), если текущее значение четное, и 1 («правда») — если нечетное. Итого, согласно следующей строке программы двоеточие будет показываться (`POINT_ON`) и гаснуть (`POINT_OFF`) каждую секунду.

В начале процедуры вы видите подобный прием определения остатка только при делении на 10. Соответственно, оператор `clock.second()%10` будет принимать значение 0 («ложь») каждую десятую секунду (любое число, отличное от нуля, интерпретируется, как «правда»), а его инверсия, наоборот, каждую десятую секунду принимать значение «правда». В эту секунду на дисплей выводится значение дата:месяц, остальные девять секунд — часы:минуты. В этих строках операция обычного (целочисленного, т. к. участвующие числа — целые) деления на 10 выделяет старший разряд из двухразрядной цифры, операция остатка от деления на 10 — младший, которые заполняют соответствующие позиции в массиве `values`. При этом автоматически получается заполнение старших незначащих разрядов нулями.

В последней строке текущее значение секунд сохраняется в переменной `old_second`. Она используется в главном цикле программы, чтобы вызывать обновление дисплея каждую секунду:

```
void loop() {
  clock = RTC.now(); читаем время из часов
  if (clock.second()!=old_second)
  {
    display_time ();
  }
} // end loop
```

В качестве домашнего задания советую поразмыслить, как можно изменить соотношение времен демонстрации даты и времени, не изменяя общего периода в десять секунд.

Простой электронный термометр

Как и в примере из *главы 17*, для измерения температуры мы применим полупроводниковый датчик TMP36. Здесь его можно включать напрямую в один из аналоговых входов платы Arduino. Соответствующая схема (рис. 21.6) настолько проста, можно было бы ее не рисовать.

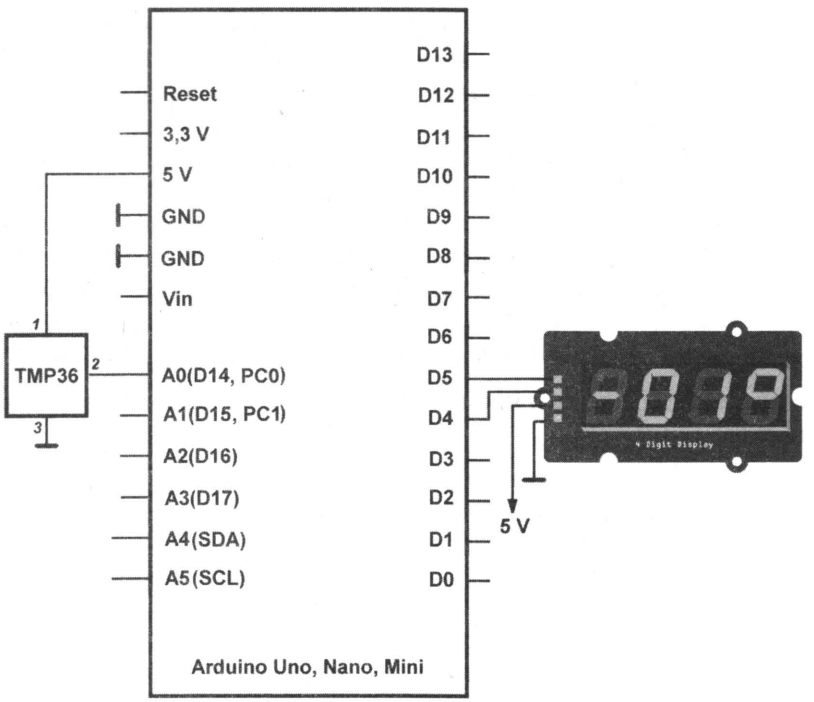


Рис. 21.6. Простой электронный термометр на 4-разрядном дисплее

Термометр будет демонстрировать значок градуса и значение температуры со знаком, но без десятых градуса. От значка градуса можно отказаться и выводить температуру с десятыми, как положено, и такой вывод будет организовать, как ни странно, проще, мы его кратко рассмотрим далее. А форматированный вывод со

значком хорошо иллюстрирует различные приемы, которые приходится применять при использовании дисплея. Напомню только, что для вывода значка градуса необходимо скачать подправленную библиотеку TM1637 (см. ссылку в начале раздела).

С использованием встроенного АЦП применительно к датчику TMP36 связан один нюанс. Если мы заглянем в описание датчика TMP36, то узнаем, что его выходной сигнал меняется на 10 мВ при изменении температуры на один градус. Тогда, если опорное напряжение АЦП равно напряжению питания 5 вольт, как по умолчанию, то цена одной единицы 10-разрядного кода в величинах напряжения будет равна примерно 5 милливольт. Иными словами, разрешающая способность такого термометра будет около половины градуса. В принципе, для данного случая (когда мы десятые градуса не демонстрируем) это приемлемо, но лучше сделать универсальную процедуру перевода милливольт в градусы, которая годится для любого варианта вывода на дисплей. В AVR это делается просто: для этого надо в качестве опорного напряжения выбрать не напряжение питания, а внутренний опорный источник ATmega, напряжение которого равно приблизительно 1,1 вольта. Это повысит разрешающую способность примерно в нужные нам пять раз.

Из того же описания узнаем, что при нуле градусов выходной сигнал будет равен примерно 0,5 вольта. Из этих данных можно составить предварительное калибровочное уравнение, которое (не забудьте!) потом придется подправлять по опытным данным. Величине кода 1023 соответствует 1,1 вольта, потому одна единица кода равна 1100/1024 = 1,075 милливольта. Значение при нуле градусов — 500 мВ или 500/1,075 = 465 единиц кода. Итого, в калибровочном уравнении $T = a1(N - a0)$, где T — температура в градусах, N — величина кода АЦП, a1 будет равно 0,1075, а a0 = 465. Заметьте, что для того, чтобы в программе получить целые градусы, величина a1 здесь должна подставляться в десять раз меньше, чем расчетная, иначе вы получите целое значение температуры, выраженное в масштабе десятых градуса. При этом значение кода, равное нулю, будет при температуре $465 \cdot 1{,}075 = 500$ десятых градуса ниже нуля, или –50,0 °C. А максимальное значение кода 1023 будет соответствовать $(1023 - 465) \cdot 1{,}075 = 599$ десятых градуса, или +59,9 °C.

Установочная секция скетча и процедура `setup()` при этих исходных данных будут выглядеть так:

```
#include <TM1637.h> //подключаем библиотеку
#define CLK 5 //тактовый вывод дисплея к выводу 5
#define DIO 4 //вывод данных дисплея к выводу 4
#define minus 16 //код минуса 16
#define degr 10 //код градуса 10 - вместо буквы A
#define clr 0x7F //код очистки разряда 127
TM1637 disp4(CLK,DIO); //создаем рабочий экземпляр библиотеки
                       //под именем disp4
 int8_t values[4] = {0,0,0,0}; //массив, начальные значения 0000
 int t; //температура
 int t_code=0; //код температуры с АПЦ
 int a0 = 465; //код нулевого значения температуры
 float a1 = 0.1075; //крутизна зависимости кода от температуры
 byte i=0; //рабочая переменная - счетчик
```

```
void setup() {
  analogReference(INTERNAL);//внутр. источник опорного напряжения
  disp4.init();//инициализация библиотеки
  disp4.clearDisplay(); //очистка дисплея
  disp4.set(BRIGHT_TYPICAL); //устанавливаем оптимальную яркость
  values[3]=degr;//в четвертом разряде всегда знак градуса
}// end setup
```

В последней строке мы навсегда устанавливаем в последней позиции значок градуса. Итого на температуру остается три позиции — две на цифры значения градусов и одна на знак минус, если он имеется. Вывод осуществляется в главном цикле программы и получается довольно навороченный:

```
void loop() {
  t_code = t_code+analogRead(0);//читаем код датчика и суммируем с предыдущими
  delay (500); //каждые полсекунды
  i++;
  if (i>15) { //каждый 16-й раз
    t_code = t_code/i; //усредняем за 16 отсчетов
    t = abs(a1*(t_code-a0)); //преобразовываем его в значение
                             //целых градусов без знака
    if (t<=99){
     values[0]=clr; //очищаем старший разряд
      //t - двузначное десятичное число, выводим его
      //во второй и третий разряды:
     values[2] = t % 10; //младший разряд = остаток от деления на 10
     values[1] = t/10; //старший разряд целочисленное деление на 10
     if (values[1]==0) i=1; else i=0; //старший значащий разряд
//выводим знак в нужный разряд или очищаем его, если знак плюс:
      if (t_code<a0) {values[i]=minus;} else {values[i]=clr;}
      disp4.display(values); //вывод массива в дисплей
      i=0; //обнуляем счетчик
      t_code = 0; //обнуляем значение кода
    } //конец t<=99
  } //конец i>15
}// end loop
```

Здесь мы вводим программный фильтр дребезга показаний — осреднение значения кода за каждые 16 отсчетов, с паузой в 0,5 секунды между ними. Таким образом, сразу после первого запуска программы дисплей будет пустым, а затем его обновление будет происходить каждые 8 секунд. Остальное должно быть понятно из комментариев.

А вот обещанный вариант процедуры вывода градусов с десятыми (значок градуса исключается, а температура должна быть действительным числом):

```
. . . .
float t; //температура
```

```
void loop() {
  t_code = t_code+analogRead(0);//читаем код датчика
  delay (500); //каждые полсекунды
  i++;
  if (i>15) { //каждый 16-й раз
    t_code = t_code/i; //усредняем за 16 отсчетов
    t = (a1*(t_code-a0)); //преобразовываем его в значение градусов
    if (abs(t)<=99){ //значение от -99.0 до 99.0
      disp4.display(t);//выводим число
      i=0; //обнуляем счетчик
      t_code = 0; //обнуляем значение кода
    } //конец t<=99
  } //конец i>15
}// end loop
```

Обратите внимание, что мы ограничили вывод числами в диапазоне от –99,0 до 99,0, иначе при сбое дисплей будет показывать невесть что. Как видите, процедура проще предыдущей, потому что действительное число со знаком можно прямо указывать в функции вывода, но красивого вывода в нужные позиции здесь не обещается (так, число «10,5» сдвинется влево относительно числа «–10,5» а число «1,5» будет размещено еще левее). Подумайте, как и в этом случае обеспечить красивый форматированный вывод в нужные позиции разрядов.

Перед тем как мы перейдем к более информативным дисплеям — матричным и многострочным, рассмотрим еще один случай дисплеев на основе обычных светодиодов: матричные дисплеи 5×7 точек.

Arduino и поразрядные матричные индикаторы

Платформа Arduino предлагает широкие возможности для конструирования всяческих измерителей, но, как мы уже говорили, вопрос часто упирается в вывод показаний так, чтобы это было эстетично и удобно. Семисегментные светодиодные индикаторы всем хороши в качестве дисплеев часов или измерителей каких-либо величин, но, во-первых, уже навязли в зубах, во-вторых, недостаточно гибкие. Практически они ограничены узкой областью отображения цифр, в крайнем случае еще нескольких сопутствующих знаков, но даже знак «+» они отображать не могут. Вообще, ассортимент достаточно эстетично выглядящих светодиодных конструкций на рынке куда у́же всех остальных разновидностей. Не размахиваться же на графический OLED-экран, когда нужно отобразить всего несколько символов, правда? Есть, конечно, сегментные дисплеи с бóльшим количеством сегментов, чем семь, но стандартное управление к ним подобрать непросто, а самостоятельные решения резко увеличивают стоимость и сложность разработки.

Здесь я предлагаю обратить свой взор в сторону простейших матричных индикаторов 5×7 точек (рис. 21.7, *справа*). Они выпускаются самых разных размеров и не-

скольких цветов, могут отображать любые символы, хоть китайские иероглифы. Сложность состоит в надлежащей организации управления большим количеством светодиодов: только один разряд содержит их 35 штук, значит, приемлемые для практических нужд линейки из четырех таких разрядов будут содержать 5×7×4 = = 140 точек. Разумеется, индивидуальное управление каждой точкой здесь невозможно и не предполагается: выводы таких индикаторов объединены по столбцам и строкам (рис. 21.7, *слева*).

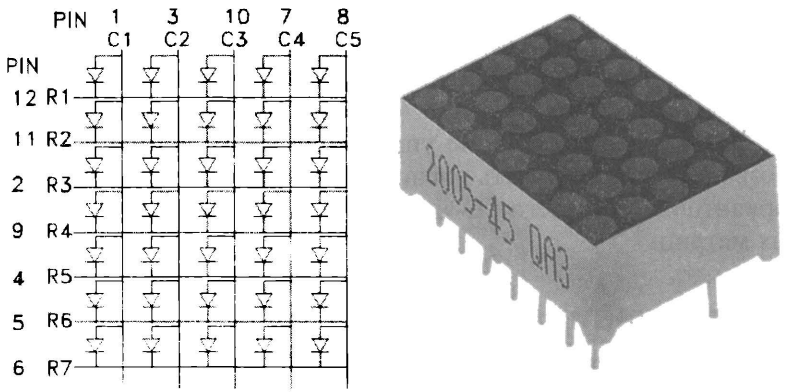


Рис. 21.7. Разряд матричного индикатора 5x7

Чтобы засветить все нужные точки независимо друг от друга, приходится городить довольно сложные алгоритмы управления. Все эти алгоритмы (например, метод «мультиплексирования Чарли»[1], Charlieplexing) предполагают динамическое управление отдельными столбцами в матрице, т. е. существенное снижение коэффициента заполнения (относительного времени в процентах, когда каждый светодиод оказывается включен) — легко подсчитать, что при 20 столбцах (четырехразрядный индикатор) этот коэффициент заполнения составит всего 5%. Учитывая, что у матриц и без того не слишком большая яркость (например, 18-миллиметровая зеленая TA07-11GWA имеет силу света 8 мКд при токе 10 мА — вдвое хуже, чем самые тусклые дискретные светодиоды), такое падение яркости нельзя назвать хоть сколько-нибудь приемлемым. Приходится искать решения, позволяющие увеличить импульсный ток через каждую точку матрицы хотя бы до 15–20 мА и одновременно получить максимально возможный коэффициент заполнения. Простые решения, как со сдвиговым регистром в случае семисегментных индикаторов, здесь не работают.

Видимо, эти обстоятельства и обусловили малую применимость точечных матриц в качестве индикаторов и в любительских, и в простых профессиональных применениях — их привычнее наблюдать в крупных табло типа «бегущая строка». Однако есть готовое решение, причем очень удобное как раз для простых приборов: это драйверы матричного дисплея конфигурации 5×7 точек × 4 разряда под названием

[1] См. например: **http://www.rlocman.ru/shem/schematics.html?di=56563**.

MAX6952 и MAX6953, где эти проблемы решены. Русское описание обеих микросхем можно найти на сайте: **www.gaw.ru**. MAX6952/6953 может обеспечить ток (устанавливаемый и программно, и схемотехнически) через точку матрицы вплоть до 48 мА, обладает встроенной таблицей шрифтов из 104 символов, дополняемых 24 пользовательскими, и управляется через скоростной SPI-интерфейс (MAX6952) или I^2C (MAX6953). Их недостаток — достаточно высокая стоимость и редкость в отечественной розничной продаже.

Сразу отметим, что решение с этими драйверами можно масштабировать практически неограниченно, получая линейки из четырех, восьми, двенадцати и т. д. разрядов. Причем без каких-то усложнений, кроме увеличения числа микросхем и роста общего потребления. Для SPI-варианта со стороны контроллера при этом увеличится лишь число управляющих выводов «выбор кристалла» соответственно числу микросхем, а для I^2C-версии в схеме управления не изменится вообще ничего — MAX6953 позволяет устанавливать внешней коммутацией соответствующих выводов до 16 вариантов внутреннего I^2C-адреса, т. е. управлять табло, составленным из 64 отдельных матриц.

Схема подключения драйвера MAX6953 с I^2C-интерфейсом

Особенности MAX6953, как имеющего минимальное число внешних соединений, мы здесь и рассмотрим. К сожалению, готовых модулей такого рода не выпускается, потому придется повозиться с довольно громоздкой платой. Дело осложняется еще и тем, что эта микросхема чаще всего представлена в продаже разновидностью в планарном корпусе SSOP с шагом 0,8 мм (в DIP-корпусе она существует только теоретически), потому переходник для нее изобретать придется даже при макетировании.

Схема подключения линейки матричных индикаторов типа TA07-11, рассчитанных на работу в качестве дисплея электронных часов, к драйверу MAX6953 показана на рис. 21.8. Всего схема имеет пять внешних соединений: питание +5 В и «земля», два вывода управляющего I^2C-интерфейса (SDA и SCL), а для часового дисплея — также и объединенные катоды светодиодов разделительного двоеточия, обозначенные на схеме как Blink. Этот вывод подключается к любому из свободных портов Arduino и управляется независимо от микросхемы.

Рассмотрение схемы начнем с вопроса о подборе величин резистора R3 и конденсатора C1, от которых зависит как максимально возможный ток через светодиоды матрицы, так и внутренняя тактовая частота драйвера. От тактовой частоты зависит частота мигания дисплея при включении режима BLINK через драйвер. Мы такой режим не используем, потому частота (f_{OSC}) нас волнует во вторую очередь (документация рекомендует выбирать ее от 1 до 8 МГц), а вот ток через сегменты (I_{SEG}) очень важен. Согласно документации на MAX6953, эти величины определяются формулами (обозначения далее — в соответствии со схемой рис. 21.8):

- $I_{SEG} = 2144/R3$ mA
- $f_{OSC} = 6003/(R3 \times (C1 + C0))$ MHz

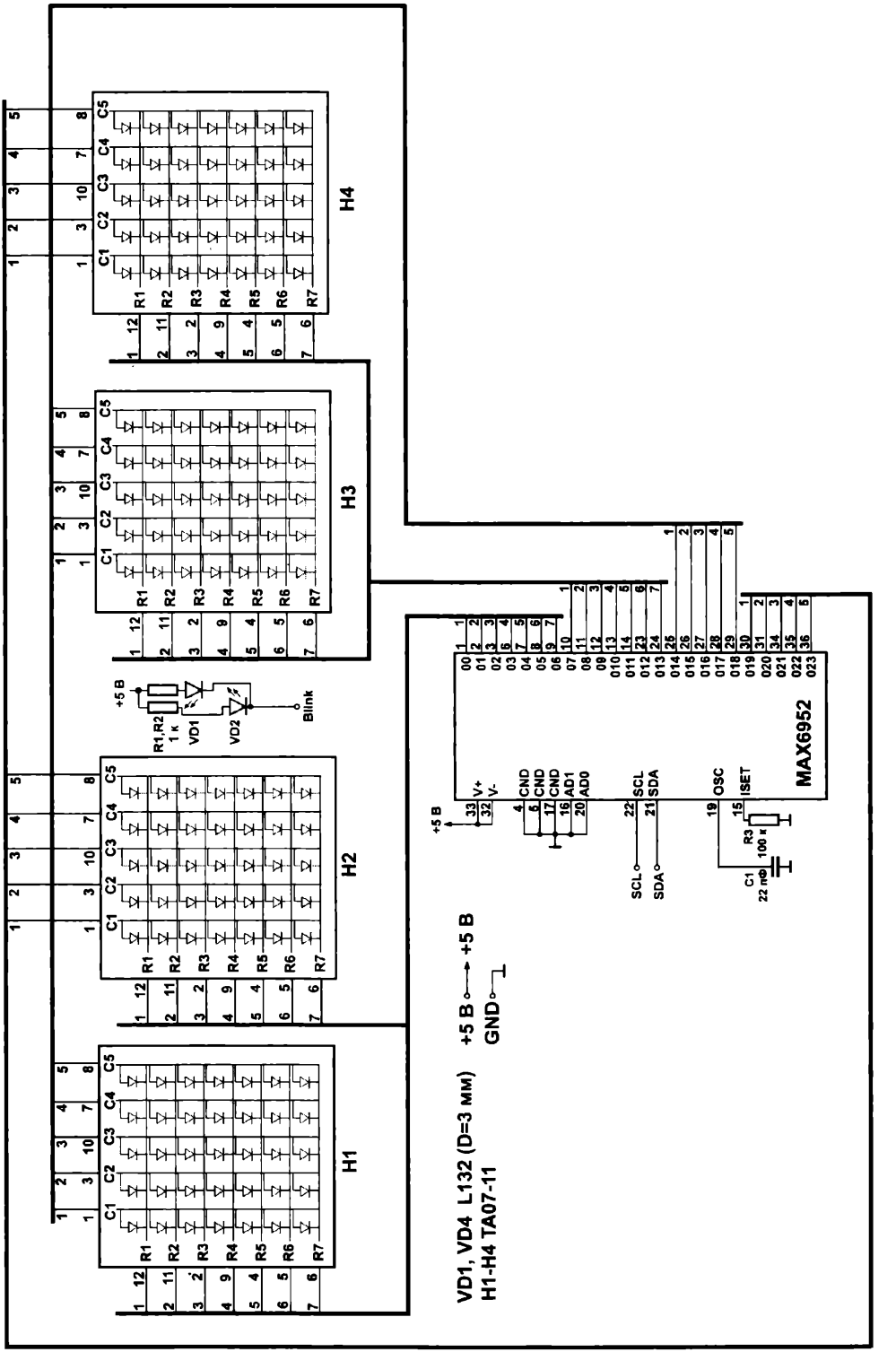


Рис. 21.8. Схема подключения 4-х матричных индикаторов 5×7 точек к драйверу MAX6953

Здесь R3 подставляется в килоомах, C1 — в пикофарадах, C_0 — паразитная емкость монтажа (типовое значение около 2 пФ).

По умолчанию документация рекомендует для R1 и C3 нестандартные значения 53,6 кОм и 26 пФ, что дает частоту f_{OSC} в 4 МГц, и максимальный ток через сегмент I_{SEG} равный 40 мА. Разобраться в том, что в документации понимается под «сегментом», непросто, потому я поясню, к чему относится эта величина тока: в каждый момент времени включены два из четырех разрядов, и для расчета общего среднего тока потребления величину I_{SEG} надо умножать не на 20 (5×4 — общее число столбцов), а на 10. Это равносильно коэффициенту заполнения 50%, т. е. реальный средний ток через столбец матрицы будет вдвое меньше I_{SEG}, следовательно, под сегментом тут понимаются два столбца в разрядах, горящих одновременно. То есть при таком токе вся конструкция будет потреблять около 415 мА, считая собственное потребление драйвера. Однако при напряжении питания 5 В выделяющаяся мощность может превысить возможности микросхемы, потому документация рекомендует ставить дополнительный резистор по питанию. Мы в эти тонкости вникать не хотим, потому просто увеличим номинал R3 до 100 кОм, тем самым снизив пиковый ток через светодиоды матрицы до 20 мА. Но имейте в виду, что у вас есть некоторый запас по яркости, и если вы, например, поставите матрицу побольше размерами, то можете увеличить яркость, просто уменьшив номинал R3 (только внимательно изучите документацию перед этим).

Второе соображение относится к источнику питания. При питании Arduino от USB-порта его номинальных возможностей (500 мА максимум) может не хватить, особенно если к Arduino подключены еще какие-то устройства. При питании от адаптера 7–9 В встроенный в Arduino Uno стабилизатор NCP1117 имеет достаточно высокую нагрузочную способность (до 1 А), однако надо учитывать, что он не установлен на радиатор, и при большом токе и высоком напряжении адаптера может перегреваться. По всем этим причинам питать нашу схему от платы Arduino необходимо с оглядкой. А на период отладки настоятельно рекомендую включать ее от отдельного источника 5 В — если вы что-то не так соедините, какие-то выводы MAX6953 окажутся висящими в воздухе или перемкнуты между собой (что случается при любительском монтаже микросхем в корпусах с мелким шагом), то ток потребления может резко возрасти, и лучше пожертвовать одной микросхемой MAX6953, чем еще и платой Arduino.

В остальном схема столь тонких расчетов не требует. Токоограничивающие резисторы для светодиодов разделительного двоеточия выбраны опытным путем, чтобы подогнать яркость под точки матрицы. Тип трехмиллиметровых светодиодов L132 выбран по длине волны (т. е. их цвету свечения) соответствующему индикатору TA07-11 и минимальной яркости среди других типов. Рисовать подключение к Arduino не будем: контакты MAX6953 21 (SDA) и 22 (SCL) подключаются к штатным выводам интерфейса I^2C Arduino A4 и A5, миганием двоеточия (вывод Blink) можно управлять через любой вывод Arduino.

Программа

Готовые библиотеки, напрямую пригодные для нашей цели, отсутствуют, потому придется все изобретать самим на основе стандартной библиотеки TWI-порта Wire.h. Рассмотрим построение программы управления драйвером MAX6953 от Arduino. Схема на рис. 21.8 предполагает I^2C-адрес устройства по умолчанию (0x50). О том, какие выводы надо переключить для изменения адреса, если это необходимо, читайте документацию. Запуск драйвера предполагает довольно громоздкую процедуру инициализации всех нужных регистров, которую следует размещать в секции `setup`. Предварительно в программе следует подключить библиотеку wire.h и установить необходимые переменные:

```
#include <Wire.h>
#define deviceaddress 0x50 //адрес по умолчанию MAX6953
int ledPin = 14; //A0 - мигалка разделительной точки
//определяем переменные для разрядов дисплея:
int dig3 = 0x23; //номера регистров взяты из документации
int dig2 = 0x22;
int dig1 = 0x21;
int dig0 = 0x20;

void setup()
{
//====инициализация матрицы=====
Wire.begin(); //подключаем i2c bus
Wire.beginTransmission(deviceaddress); //запись в устройство
Wire.write(0x04); //регистр конфигурации
//MAX6953 Table 6 - номер таблицы в описании
Wire.write(0x01); //выключаем режим shutdown;
Wire.endTransmission();
//MAX6953 Table 23 - следующий шаг:
Wire.beginTransmission(deviceaddress);
Wire.write(0x01); //Интенсивность разрядов 0 и 2
Wire.write(0xFF); //все сегменты на максимум = 20 мА
Wire.endTransmission();
//MAX6953 Table 24 - следующий шаг:
Wire.beginTransmission(deviceaddress);
Wire.write(0x02); //Интенсивность разрядов 1 и 3
Wire.write(0xFF); //все сегменты на максимум = 20 мА
Wire.endTransmission();
//Включаем test mode для проверки: все светодиоды горят
//MAX6953 Table 22
Wire.beginTransmission(deviceaddress);
Wire.write(0x07);
Wire.write(0x01);
Wire.endTransmission();
delay(1000); //секундная задержка
```

```
//выключаем test mode
//MAX6953 Table 22
Wire.beginTransmission(deviceaddress);
Wire.write(0x07);
Wire.write(0x00);
Wire.endTransmission();
//=====конец инициализации =======
pinMode(ledPin, OUTPUT); //управление разделительным двоеточием
}
```

Далее отдельно вставляем текст функции для вывода символа в определенный разряд. Автор по мере возможности всегда исправляет перечеркнутый ноль на дисплеях, который по инерции задержался в таблицах шрифтов для подобных индикаторов с доисторических времен господства АЦПУ и алфавитно-цифровых терминалов, но в современном антураже выглядит довольно дико. Потому функция заодно заменяет цифру «0» на букву «О». Если вас это не волнует, удалите функцию `if..else`, оставив от нее только строку `Wire.write(value)`:

```
//запись символа value в разряд disp
void writeChar(byte value,byte disp)
{
Wire.beginTransmission(0x50);
Wire.write(disp);
if (value == '0') Wire.write('O'); //ноль заменяем на букву O!!
else Wire.write(value);
Wire.endTransmission();
}
```

И наконец, собственно проба вывода символов в функции `Loop`:

```
void loop()
{
 //матрица:
writeChar('0',dig3);
writeChar('2',dig2);
writeChar('0',dig1);
writeChar('3',dig0);
delay(500);
 digitalWrite(ledPin, LOW);
writeChar('6',dig3);
writeChar('0',dig2);
writeChar('8',dig1);
writeChar('0',dig0);
delay(500);
 digitalWrite(ledPin, HIGH);
 }
```

Эта программа выводит на дисплей попеременно строку цифр «30:20» и «08:06» с одновременным подмигиванием двоеточием. Вы можете поэкспериментировать,

подставляя в функции `writeChar` разные другие символы. Придется, однако, ограничиться только английским алфавитом и рядом специальных знаков (см. таблицу в документации). Среди этих знаков есть символы доллара, евро и иены, что позволяет на основе MAX6953 делать, например, табло валютных обменников.

Внешний вид метеостанции с часами на матричных индикаторах показан на рис. 21.9. Обратите внимание, что, в отличие от семисегментных, в этих индикаторах отсутствует разделительное поле справа и слева, они рассчитаны на возможность составлять непрерывные линейки матриц большего размера. Это, однако, приводит к тому, что символы шрифта 5×7 сливаются, потому индикаторы установлены с некоторым промежутком. Для реализации таких часов к Arduino надо по тому же I^2C подключить любой из описанных ранее модулей часов на основе DS1307 или ее аналогов. Других соединений, кроме управления двоеточием (вывод Blink), тут не требуется, и остальные порты контроллера можно использовать под другие нужды.

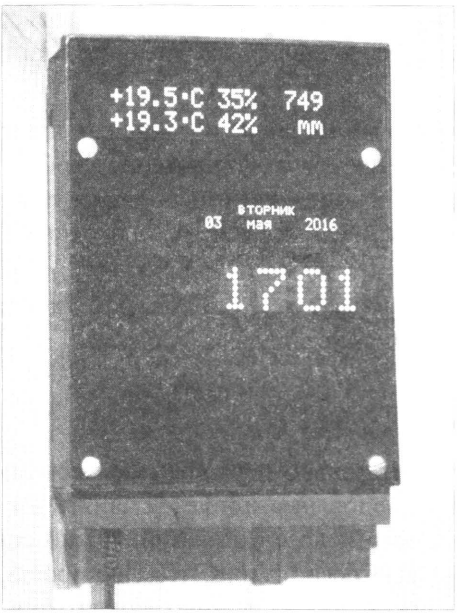


Рис. 21.9. Внешний вид метеостанции с часами на матричных индикаторах

Полную программу для такого варианта часов можно скачать с сайта автора по ссылке: **http://revich.lib.ru/AVR/RTC_matrix.zip**. Двоеточие в ней управляется через вывод A0 (цифровой вывод 14). Она использует старую и когда-то популярную библиотеку чтения часов с простым названием DS1307.h, которая не позволяет автоматически устанавливать точное время. Самостоятельная переделка кода под более современные библиотеки (в том числе и замена архаичной функции `decToBcd` для распаковки значений часов и минут на более простые методы) для читателя, без сомнения, уже не составит труда и будет полезной тренировкой — все нужное описано в этой главе ранее.

Теперь перейдем, наконец, к матричным (графическим) и строчным (текстовым) индикаторам, имеющим куда более широкие возможности.

Работа с текстом на графическом дисплее MT-12864J

Для начала мы рассмотрим популярный отечественный графический ЖК-модуль MT-12864J фирмы МЭЛТ. Автор уже неоднократно говорил о своей нелюбви к ЖК-дисплеям, однако такая конструкция может быть полезной, например, при работе от батареек. Хотя при этом надо учесть, что даже без подсветки, согласно документации, экран потребляет 4 мА, и круглосуточная работа при таком потреблении, как минимум, затруднительна. А вот для приборов вроде тестеров, включаемых на время, MT-12864J — идеальный вариант даже с включенной подсветкой.

На сайте «Амперки» в разделе **Вики** легко разыскать статью «Работа с ЖК-матрицей 128×64», рассказывающую о подключении этого модуля в стандартном графическом режиме с помощью библиотеки GLCD. Здесь мы подробнее остановимся на некоторых нюансах практического применения ЖК-матриц на основе контроллера ks0108, а также рассмотрим вывод текста с помощью готовых шрифтов из обновленной библиотеки openGLCD и вопросы их модернизации для вывода кириллических символов.

Для большинства графических и строчных матричных дисплеев имеется стандартный контроллер, по образцу которого строится управление любой аналогичной матрицей. Для символьных строчных экранов, о которых далее, стандартный контроллер называется HD44780, а для графических ту же роль играет ks0108 — с ним совместимы почти все графические экраны небольшого размера. Матрицей 128×64 управляют два таких контроллера — каждый своей половиной дисплея 64×64 точки, отсюда у дисплея два вывода «выбор кристалла»: CS1 и CS2.

Подключение MT-12864J

К сожалению, контроллер ks0108 имеет только параллельный восьмибитовый интерфейс, и с учетом управляющих выводов нам придется занять аж 13 функциональных контактов платы Arduino Uno. Число соединений можно сократить, если подключить ЖК-модуль через сдвиговый регистр или, что еще проще, дешифратор двоичного кода. В первом случае число линий данных сократится с восьми до одной плюс линия импульсов сдвига, во втором — до трех. Но в таких случаях о стандартной библиотеке придется забыть, и создавать свои схемы подключения и писать свои процедуры вывода.

Поэтому мы прибегнем к стандартному способу, но внесем в него некоторые изменения. Использовать именно те выводы Arduino, что предусмотрены библиотекой GLCD по умолчанию[1], при таком их количестве не только неудобно, но часто просто невозможно. Не забудем, что ведь мы еще подключаем к контроллеру всякие

[1] См. таблицу в описании библиотеки на официальном сайте Arduino, где матрице MT-12864J соответствует вариант «Pinout A»: **http://www.arduino.cc/playground/Code/GLCDks0108**. Описания этой библиотеки на русском языке, к сожалению, не имеется.

другие устройства по стандартным коммуникационным портам, и стоит постараться не занимать их выводы по максимуму.

Модернизированная с учетом этого обстоятельства схема подключения MT-12864J к Arduino Uno приведена на рис. 21.10. Как видите, у нас оставлены свободными контакты D1 и D2 (RX и TX последовательного порта) и контакты A4–A5, которые, как мы знаем, участвуют в обмене через порт I^2C с различными датчиками.

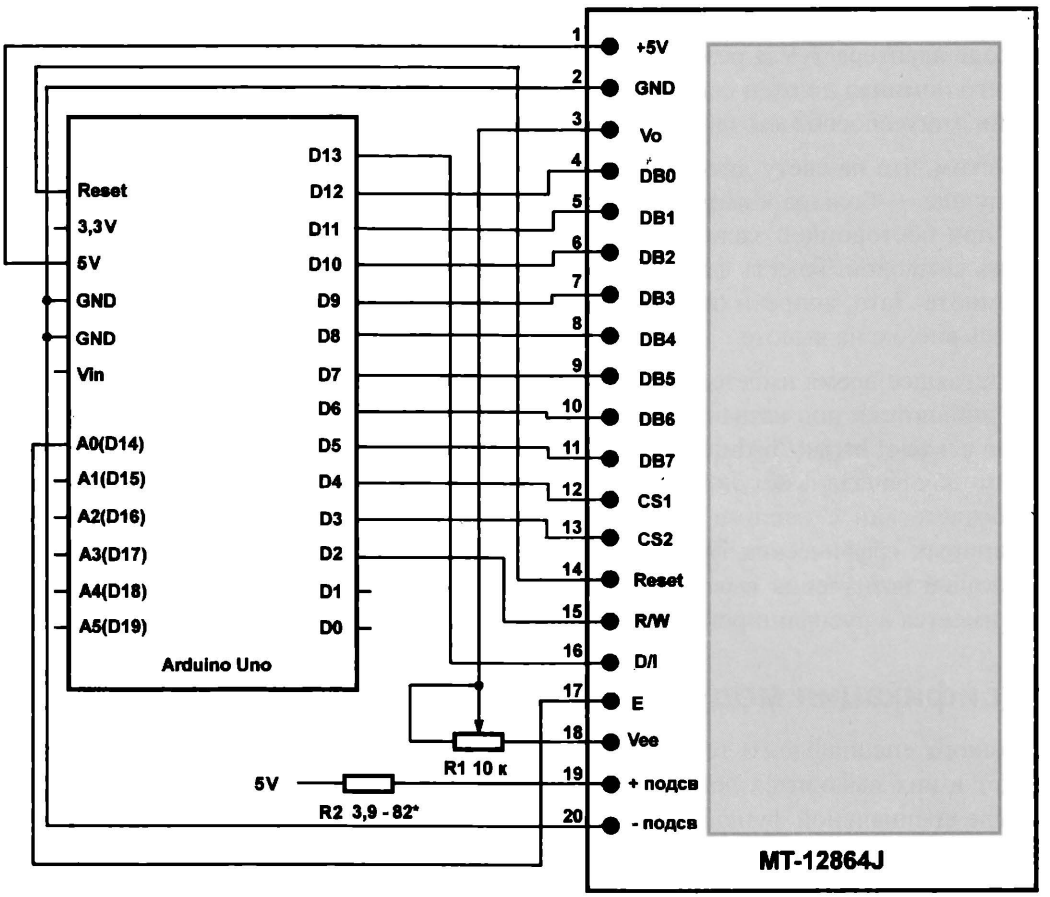


Рис. 21.10. Схема подключения MT-12864J к Arduino Uno

Обратим также внимание на резисторы R1 и R2, о которых почему-то авторы статей о графических матрицах часто забывают упомянуть. Переменный или подстроечный резистор R1 служит для регулировки контрастности индикатора. На схеме он подключен так, как рекомендуется в документации фирмы МЭЛТ, однако может подключаться и по схеме потенциометра: одним концом к «земле», другим — к питанию 5 В, а средним выводом — к контакту Vo индикатора. Номинал его выбирается в пределах 10–20 кОм, и в конечном продукте R1 можно заменить на постоянный резистор подобранного номинала.

Резистор R2 — токоограничивающий для LED-подсветки. Его можно не устанавливать вовсе (и в документации он не упоминается). Согласно документации МЭЛТ, ток подсветки при отсутствии этого резистора составит около 64 мА. Установить токоограничивающий резистор R2 следует в двух случаях: если вы хотите уменьшить ток (и, соответственно, пожертвовать яркостью подсветки) или если подсветка питается от отдельного источника с повышенным напряжением. Последний вариант обычно реализуется при смешанном питании, когда подсветка подключается к нестабилизированному напряжению на выходе сетевого адаптера, а при переключении на батарейку отключается вовсе. При обычном напряжении на выходе адаптера 7–9 В резистор R2 должен погасить лишние 2–4 В, соответственно, его номинал должен составлять от 39 до 62 Ом. На других подробностях реализации этого способа мы здесь останавливаться не будем

Отметим, что на свету дисплей MT-12864J при отсутствии подсветки выглядит даже лучше — больше контраст и углы обзора, а сама подсветка настолько тусклая, что при посторонней засветке экрана только снижает контраст, ухудшая различимость символов. То есть фактически она требуется только при эксплуатации экрана в темноте. Зато, вопреки ожиданиям, качество дисплея в отсутствие подсветки оказалось вполне на высоте.

В настоящее время имеется модернизованный под новые версии Arduino IDE вариант библиотеки под названием openGLCD. Скачать его можно с официального сайта по ссылке: **https://bitbucket.org/bperrybap/openglcd**. Так как мы меняли контакты, то для начала файл библиотеки, где обозначены выводы, придется «причесать» в соответствии с нашими потребностями — странно, но увлекшись реализацией различных графических функций, автор библиотеки так не дополнил ее простой функцией назначения выводов. Описывать, как это делать, я здесь не буду — все это имеется в русифицированном варианте библиотеки, о котором далее.

Русификация модуля MT-12864J

Никакого специального текстового режима в модулях MT-12864J не существует. Текст в них выводится просто как картинка, для чего имеются таблицы шрифтов в виде специальной функции, вызываемой через оператор `SelectFont`. Чтобы русифицировать этот индикатор, для него придется создать шрифт с русскими символами. Причем создать «руками» — рекомендуемый в описании библиотеки конвертер шрифтов Windows, к сожалению, не понимает никаких символов, кроме служебных и английских. Это объясняется тем, что в UTF-8, принятой в качестве кодировки для текстовых файлов Windows (в том числе и для Arduino IDE), только эти символы из стандартной таблицы ASCII однозначно переводятся в однобайтовую кодировку. Нет никаких сомнений, что авторы конвертера могли бы с этой проблемой справиться, — но представьте, какой объем работы им пришлось бы провернуть, чтобы охватить всего десяток-другой самых популярных языков?

Но еще важнее то, что со времен создания первых версий библиотеки и конвертера Arduino IDE успела перейти в отношении всех не-ASCII-символов на стандартную многобайтовую кодировку UTF-8. Итого простой правки таблиц шрифтов, как это

предлагалось в предыдущих изданиях этой книги, оказывается недостаточно — поверх нее требуется еще отдельная функция перекодировки из двухбайтовой UTF-8.

На имеющийся в составе библиотеки шрифт cp437font8x8 мы обращать внимания не будем — он хотя и имеет символы кириллицы, но слишком крупные, с ошибками, и к тому же непонятно в какой кодировке (cp437 в названии шрифта вообще к кириллице отношения не имеет). Мы модернизируем имеющийся в комплекте библиотеки GLCD английский шрифт 5×7 точек, размещающийся в файле SystemFont5x7.h (папка fonts). Никакие особые инструменты для этого не нужны — немного поразмыслив над приведенными там кодами (для наглядности каждый байт следует разложить в двоичное представление и записать все пять штук один над другим), вы легко разберетесь в принципе устройства таблицы и сможете ее менять и дополнять без какого-либо визуального редактора.

Опять же на подробностях этого процесса я здесь не останавливаюсь: уже исправленный шрифт SystemFont5x7R.h имеется в русифицированной библиотеке (см. далее). Кроме добавленных русских букв, в шрифте переопределен значок градуса. В оригинальном шрифте System5x7.h значок градуса вынесен в последнюю строку таблицы, занимая символ 0x80, который у нас уже относится к кириллице. Чтобы не нарушать порядок построения кириллической таблицы в соответствии с UTF-8, эта строка выброшена из файла. А значок градуса пристроен вместо символа ASCII «~» (номер 0x7E), который в шрифте все равно не использовался по назначению. Зато такая замена позволяет вводить значок градуса в тексте скетча прямо с клавиатуры в виде символа «~».

Но этого, как мы видели, недостаточно — нужна еще функция-перекодировщик из двухбайтового шрифта. Чтобы не лопатить всю библиотеку, создана надстройка в виде отдельной (не библиотечной) функции `outstr()`, которая перебирает все элементы входной строки, пропуская их через оператор выбора `switch`. Он вылавливает из строки кириллические символы и заменяет их однобайтовыми кодами, соответствующими модернизированному файлу шрифта System5x7R.h.

Например, для заглавной русской буквы «Ф» строка замены будет такой:

```
case 'Ф': GLCD.PutChar(0xA4); break;
```

Здесь `0xA4` — младший байт кодировки буквы «Ф» в UTF-8, соответствии с этой кодировкой и составлен новый файл шрифта System5x7R.h.

Подправленную библиотеку можно скачать с сайта автора по адресу: **http://revich.lib.ru/AVR/openGLCD.zip**. Пример Proba_cyr_openGLCD находится в том же архиве отдельно (не в папке examples библиотеки). Результат его выполнения показан на рис. 21.11.

Если вы рассмотрите исходный код указанного примера, то поймете, как обращаться с текстом при выводе через функцию `GLCD.print()` или `outstr()`. Текстовая зона с данным шрифтом содержит 8 строк по 21-му символу в каждой. Если при выводе строка оказывается длиннее 21 символа, ее конец автоматически перейдет на другую строку. Для принудительного перевода строки можно, как обычно, использовать функцию `GLCD.println()`. Чтобы правильно позиционировать вывод текста,

следует иметь в виду, что библиотечная функция `CursorToXY()` рассчитана на графический экран 128×64 точки. По этой причине при выводе текста указывать положение курсора удобно с учетом того, что символ занимает 6 точек по ширине, а строка — 8 точек по высоте. Поскольку позиции в строке и сами строки нумеруются с нуля, то вывод символа в предпоследнюю (20-ю, т. е. номер 19) позицию пятой (т. е. номер 4) строки предваряем оператором `CursorToXY(19*6,4*8)`.

Рис. 21.11. Результат выполнения тестовой программы для дисплея MT-12864J

Обратите внимание, что вывод на такой дисплей всегда должен начинаться с функции установки курсора на определенную позицию — чтобы выводимые символы заменяли старые на том же месте. Иначе в следующем цикле функции `loop()` строки быстро поползут вверх, не давая разглядеть результата.

С помощью дисплея MT-12864J можно быстро и дешево создать метеостанцию, подключив к оставшимся входным портам Arduino часы и подходящие датчики из тех, что рассматривались ранее. Внешний датчик температуры/влажности можно для простоты подключить кабелем. Образец размещения данных на дисплее такой метеостанции показан на рис. 21.12.

Рис. 21.12. Отображение результатов работы метеостанции на ЖК-дисплее

Строчные OLED-дисплеи

Как мы уже говорили, для строчных матричных экранов стандартный контроллер носит название HD44780. Для работы с ним имеется стандартная библиотека LiquidCrystal, входящая в состав Arduino IDE. С командами HD44780 совместим интерфейс любых подобных конструкций, как ЖК-, так и OLED-разновидностей, однострочных или многострочных. Потому то, что сказано далее о строчных дисплеях в отношении работы с LiquidCrystal или ее русифицированной версией LiquidCrystalRus, в основном относится к любым их разновидностям. Причем в случае ЖК-дисплеев оригинальные библиотеки работают без проблем, а вот в случае OLED требуют некоторой модернизации. Мы здесь будем говорить только об OLED-разновидностях фирмы Winstar.

Заметки на полях

Конечно, есть другие производители подобных дисплеев (Midas, Newhaven), но в Сети давно заметили, что ничем от Winstar их продукция не отличается, и выпускаются эти дисплеи явно на одном конвейере. Непосредственная проверка дисплея от Newhaven подтвердила, что управляется он теми же самыми функциями, что и Winstar. При этом в документации от Newhaven вместо указания типа контроллера написано просто «LCD-comparable controller» — не могут же они прямо признаться, что продают китайскую разработку под своим брендом.

В OLED-версии дисплеев отсутствует вывод управления контрастом Vo (вывод 3 индикатора не подключается), и также ни к чему не подсоединяются выводы 15 и 16, в ЖК-версии управляющие подсветкой. Правда, некоторые сетевые источники утверждают, что функциональность этих выводов можно восстановить путем перестановки некоторых перемычек на плате, и таким образом управлять яркостью свечения точек, но не очень понятно, зачем. Индикаторы WEH001602, которые мы рассматриваем далее, могут работать как от питания 5 В, так и от питания 3,3 В, и именно от напряжения питания зависит яркость. Опыт показал, что нормальной яркости свечения дисплей достигает уже при 3,3 В. При питании 5 В яркость в нормальных условиях явно избыточна, но предполагается, что передняя панель будет выполняться из дымчатого пластика, затемняющего «потроха» прибора (см. фото на рис. 21.9), так что в готовом изделии яркость окажется в самый раз.

Потребление индикатора, например, WEH001602B при питании 5 В, согласно фирменной документации, составит 43 мА. Обратите внимание, что это почти в полтора раза меньше, чем потребление ЖК-панели MT-12864J с включенной подсветкой. На самом деле потребление должно быть еще ниже — цифра в документации указывает на случай, когда засвечены все точки матрицы, в реальной жизни такого, конечно, не случается.

Подробности

Несколько слов об обозначениях дисплеев Winstar. Первые буквы означают технологическую разновидность: WEH означает, что это OLED-тип. Цифры — конфигурацию (например, 1602 означает строчный дисплей 16 символов в 2 строки, 10016 — графический 100×16 точек). Лишние два нуля перед этими цифрами ничего не обозначают, и мы далее их часто будем опускать. А первая буква после цифр (A, B, C) — физическую конфигурацию, т. е. размеры матрицы, расположение и тип разъема. Для опти-

мальных с точки зрения удобства строчных дисплеев 16×2 тип A означает верхнее расположение разъема, тип B — нижнее, тип C — боковой разъем в два ряда. При этом тип 1602B обладает самой крупной из всех остальных разновидностей матрицей с физическим размером символа 5×9 мм. Для других сочетаний символов и строк та же буква может означать другую конфигурацию.

Контроллер WS0010 и библиотека LiquidCrystal

Модернизированный вариант стандартного контроллера HD44780 от Winstar носит незамысловатое наименование WS0010. Главное отличие его от стандартного заключается в наличии нескольких встроенных таблиц шрифтов, из-за чего управление этим дисплеем усложняется, и приходится модернизировать стандартную библиотеку LiquidCrystal (она входит в поставку Arduino). Эта проблема за нас уже решена — имеется, как и отмечалось уже ранее, проверенный русскоязычный вариант LiquidCrystalRus, который можно скачать по адресу: **https://github.com/mk90/LiquidCrystalRus**. Но проблема заключается не в одних только шрифтах — как водится, что-то улучшив, разработчики что-то и ухудшили, существенно изменив порядок инициализации дисплея и очень неуверенно описав его в документации. Учтите, что в этом разделе и далее (там, где о графическом режиме) речь идет только о четырехпроводном (четырехбитовом) включении дисплеев (см. рис. 21.13 далее) — кажется, восьмипроводной в практических схемах никто никогда и не пробовал.

Самый крупный недостаток этих дисплеев — ни в традиционном HD44780, ни в новом WS0010 не предусмотрено наличие аппаратного Reset (в этом их отличие от рассмотренного выше ks0108). Его наличие решило бы множество проблем, но в этих конструкциях он заменен специальной последовательностью команд, которые нужно выполнять при каждом запуске. Для разных разновидностей контроллеров и дисплеев они разные Поэтому приходится править библиотеку, чтобы избавиться от «глюков» при запуске OLED-разновидностей. Со стандартной LiquidCrystal после перепрограммирования на экране появляется мусор, от которого удается избавиться лишь выключением/включением питания (а еще лучше переключением на более мощный адаптер). Однако при этом могут произвольно меняться местами верхняя/нижняя строка, и от этого недостатка избавиться бывает очень сложно.

Основных направлений модернизации библиотеки LiquidCrystal четыре:

1. В контроллере WS0010 целых четыре кодовых таблицы для разных языков. По умолчанию включена ENGLISH_JAPANESE, и для включения русских символов нужно ее переключить на ENGLISH_RUSSIAN. Так как русско-английская таблица в WS0010 имеет номер 2, то для переключения на нее нужно два младших бита FT1 и FT0 в команде `FUNCTION SET` установить в состояние 10 (0x02). (В ЖК-дисплеях с одной кодовой таблицей эти биты, кстати, никак не используются.) Для этого в файлах библиотеки нужно исправить выражение для установки значения переменной `displayfunction`.
2. Далее нужно исправить задержку инициализации после включения питания. Для HD44780 она должна быть не более 40 мс. В библиотеке для этого используется

функция `delayMicroseconds(50000)`. Для контроллера WS0010 нужно иметь задержку в десять раз больше — не менее 500 мс. Требование это так тщательно спрятано (файлы с англоязычной документацией по дисплеям Winstar «потеряли шрифт» как раз в этой части), что о нем, кажется, до сих пор мало кто задумывался. Поэтому мы заменяем эту строку на 32 повтора задержек по 16 мс каждая:

```
for (int i = 0; i <=31; i++) delayMicroseconds(16000);
```

3. Кроме того, после этой задержки при четырехпроводном включении нужно пять раз подряд подать пустую команду (`0x00`). Так как Arduino существенно быстрее контроллера дисплея, команды следует подавать с промежуточной задержкой.
4. Как показал опыт, для окончательного избавления от «глюков» всего этого недостаточно. Для избавления от путаницы между строками необходимо еще провести команды выключения дисплея, переключения режима и очистки и последующего включения заново:

```
command(0x08);//выключили экран
    delayMicroseconds(100);
command(0x17);//переключение в текстовый режим
    delayMicroseconds(100);
command(0x01);//очистили от мусора ОЗУ
    delayMicroseconds(100);
command(0x04 | 0x08);//включили экран
    delayMicroseconds(100);
```

В исправленной библиотеке LiquidCrystalRus_OLED (см. далее) этими командами заменено содержимое функции `clear()`. В скетчах, использующих этот вариант библиотеки, в начале `setup()` обязательно надо вызывать функцию `clear()`, при этом она заодно и почистит экран.

Пишем по-русски

Проблему подключения кириллической таблицы мы решили, но русский язык остался доступен нам только в принципе. Если вы рассмотрите таблицу ENGLISH_RUSSIAN (она имеется в полном описании любого из дисплеев Winstar, а также в руководстве по контроллеру WS0010), то увидите, что там применен экономичный метод кодирования — вводятся только русские буквы, не совпадающие с английскими по начертанию. Можно просто вводить их в виде восьмеричных кодов, руководствуясь табл. 21.1. Но это неудобно и долго: например, слово «суббота» тогда будет выглядеть, как «су\262\262о\277а».

Поэтому проще прибегнуть к русифицированной версии библиотеки под названием LiquidCrystalRus, которую, как уже отмечалось, можно скачать по адресу **https://github.com/mk90/LiquidCrystalRus**. Чтобы она работала с OLED-индикаторами, эту библиотеку необходимо также доработать в части процедур инициализации, как рассказано выше. Такую доработанную библиотеку LiquidCrystalRus_OLED можно скачать с сайта автора по адресу: **http://revich.lib.ru/AVR/LC_OLED.zip**. Напомню, что все это касается OLED-дисплеев, для обычных ЖК можно применять неисправленные версии библиотек (или, возможно, исправленные, но иначе).

Таблица 21.1. *Коды кириллических символов и значка градуса для контроллера WS0010 (таблица ENGLISH_RUSSIAN, код 0x02)*

Русский символ	Код		Русский символ	Код	
	восьмеричный	шестнадцатеричный		восьмеричный	шестнадцатеричный
Б	240	A0	д	343	E3
Г	241	A1	е	265	B5
Д	340	E0	ж	266	B6
Ё	242	A2	з	267	B7
Ж	243	A3	и	270	B8
З	244	A4	й	271	B9
И	245	A5	к	272	BA
Й	246	A6	л	273	BB
Л	247	A7	м	274	BC
П	250	A8	н	275	BD
У	251	A9	п	276	BE
Ф	252	AA	т	277	BF
Ц	341	E1	ф	344	E4
Ч	253	AB	ц	345	E5
Ш	254	AC	ч	300	C0
Щ	342	E2	ш	301	C1
Ъ	255	AD	щ	346	E6
Ы	256	AE	ъ	302	C2
Э	257	AF	ы	303	C3
Ю	260	B0	ь*	304	C4
Я	261	B1	э	305	C5
б	262	B2	ю	306	C6
в	263	B3	я	307	C7
г	264	B4	°(градус)	337	DF

* прописной мягкий знак — английское «b».

Серым цветом в таблице отмечены символы, которые в некоторых библиотеках надо вводить напрямую, с помощью восьмеричного кода в строке (например, «всё» — «вс\265», «22,5°» — «22,5\337»). В общем случае это касается только значка градуса (точнее, верхней точки, которой здесь удобно заменять этот символ — нормального значка градуса в библиотеке не имеется). Обычно обе буквы «Ё» и «ё» выводятся в виде букв аналогично остальным, но есть одно исключение, о котором далее. Кстати, LiquidCrystalRus выводить такой значок градуса не позволит (код «\337» будет проигнорирован), и в доработанной LiquidCrystalRus_OLED

для этого пришлось исправить условие в сложной функции перекодировки из UTF-8 в табличные коды.

Подключение строчных дисплеев Winstar

Кое-как разобравшись с программированием, займемся, наконец, схемотехникой рассматриваемых дисплеев. Подключение согласно упомянутой четырехпроводной схеме выглядит так, как представлено на рис. 21.13 (заодно на ней показаны наши любимые часы, чтобы потом не рисовать схему заново). На схеме обозначен дисплей 16 символов на 2 строки (а также графический 100×16 точек, о котором далее), но точно так же можно подключать дисплей любой другой конфигурации на основе контроллера WS0010 (12×2, 12×4, 20×2 и т. п.), а также аналогичные дисплеи Newhaven и, вероятно, Midas, о которых упоминалось ранее. Исключение

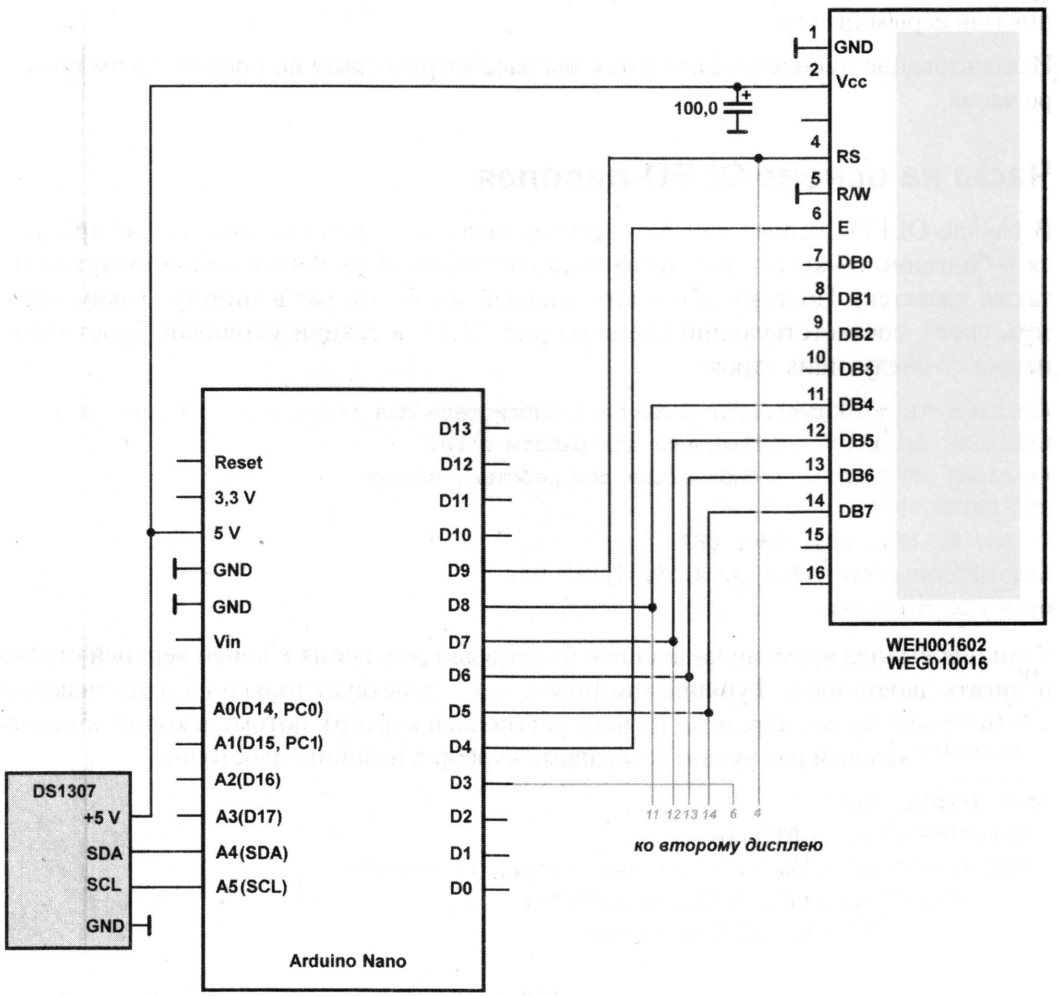


Рис. 21.13. Подключение дисплеев Winstar к Arduino по четырехпроводной схеме

составляет дисплей WEG010032, т. е. графический диплей 100×32 точки (о котором далее). В зависимости от конфигурации будет меняться только задание типа дисплея в строке инициализации.

К счастью, выводы для подключения основных линий здесь можно также задавать в процессе инициализации, и их можно произвольно менять. Соответствующая рис. 21.13 строка будет выглядеть так:

```
// RS, E, DB4, DB5, DB6, DB7
LiquidCrystalRus OLED1(9, 4, 8, 7, 6, 5);
```

Обратите внимание, что вывод R/W дисплея подключается к «земле», потому что чтение из дисплея в наших программах не используется. При необходимости подключения второго и других дисплеев, они подключаются параллельно первому, за исключением вывода E, который здесь играет ту же роль, что и вывод «выбор кристалла» CS у микросхем. На рисунке вариант подключения второго дисплея показан серым цветом.

Использование дисплея и библиотек мы рассмотрим сразу на практическом примере часов.

Часы на основе OLED-дисплея

В случае OLED-дисплея каждую секунду выводить время на дисплей не требуется — мигание точки тут обеспечивается специальной функцией (ЖК-вариантов это также касается). Поэтому обновлять дисплей мы будем раз в минуту. Таким образом, скетч, соответствующий схеме на рис. 21.13, в секции установок будет начинаться со следующих строк:

```
#include <LiquidCrystalRus_OLED.h> //библиотека для текстового OLED-дисплея
#include <Wire.h> //библиотека для работы с TWI
#include <RTClib.h> //библиотека для работы с часами
RTC_DS1307 RTC;
// RS, E, DB4, DB5, DB6, DB7
LiquidCrystalRus OLED1(9, 4, 8, 7, 6, 5);
byte old_minutes;
```

Функция вывода времени на дисплей будет выводить время в конец верхней строки и мигать двоеточием. Функция мигания `blink()` действует только на одно знакоместо (и только до тех пор, пока на него установлен курсор), потому в конце обновления экрана каждый раз нужно возвращать курсор в позицию двоеточия:

```
void display_time () {
  DateTime clock = RTC.now();
  OLED1.setCursor(11,0); //верхняя строка, 11 позиция
  if (clock.hour()<10) OLED1.print("0");
  OLED1.print(clock.hour()); //часы
  OLED1.print(":");
  if (clock.minute()<10) OLED1.print("0");
  OLED1.print(clock.minute()); //минуты
```

```
  old_minutes=clock.minute(); //запоминаем текущую минуту
  OLED1.setCursor(13,0); //верхняя строка, 13 позиция ":"
  OLED1.blink(); //мигаем двоеточием
}//конец display_time
```

Мы здесь намеренно сместили время до упора вправо. В начало верхней строки можно вывести что-то еще — температуру, например, или какую-то информацию, но длиной не более 12 символов, включая разделительный пробел в конце. Если хотите поставить время посередине, нужно изменить установку курсора: в функции `display_time()` строку `OLED1.setCursor(11,0)` надо заменить на `OLED1.setCursor(6,0)`, а `OLED1.setCursor(13,0)` (для мигания) на `OLED1.setCursor(8,0)`.

Отдельно напишем функцию вывода календаря. Она разрастается в сравнении с той, что была ранее, потому что здесь мы получили возможность для вывода дня недели, причем словами. К сожалению, при 16 символах в строке после вывода даты (с усеченным до двух цифр годом) на день недели остается максимум 7 символов, потому длинные «понедельник» и «воскресенье» приходится урезать. Для дисплея с конфигурацией 20 символов в строке этого делать бы не пришлось, но, как мы говорили, только у дисплея WEH1602B символы высотой 9 мм, у остальных они мельче (у WEH2002A 5,5 мм). В случае, если вас размеры символов не волнуют, можете выбрать дисплей с 20 символами и исправить эти строки, только не забудьте исправить задание конфигурации дисплея в процедуре `setup()` далее.

```
void display_date_dayofweek () {
  OLED.clear(); \\очистка дисплея
  DateTime clock = RTC.now();
  OLED1.setCursor(0,1); //нижняя строка, нулевая позиция
  if (clock.day()<10) OLED1.print("0");
  OLED1.print(clock.day()); //дата день
  OLED1.print(".");
  if (clock.month()<10) OLED1.print("0");
  OLED1.print(clock.month()); //дата месяц
  OLED1.print(".");
  int year2=clock.year()-2000; //две цифры года
  OLED1.print(year2); //год
  OLED1.print(" ");
   int dayofweek = clock.dayOfTheWeek();
   switch(dayofweek){
     case 1:
     OLED1.print("понед");
     break;
     case 2:
     OLED1.print("вторник");
     break;
     case 3:
     OLED1.print("среда");
     break;
```

```
    case 4:
    OLED1.print("четверг");
    break;
    case 5:
    OLED1.print("пятница");
    break;
    case 6:
    OLED1.print("суббота");
    break;
    case 0:
    case 7:
    OLED1.print("воскрес");
    break;
   delay(1000);
  } //конец switch
} //конец display_date_dayofweek
```

Обратите внимание на идущие подряд операторы `case 0` и `case 7`. В большинстве микросхем RTC и библиотек для них воскресенье объявлено седьмым днем (как, вообще-то, и положено), но встречается и вариант нулевого дня. Чтобы не гадать, какой случай перед нами, оставлено оба варианта, это ничему не помешает.

Чтобы курсор остался в позиции мигающего двоеточия (см. конец функции `display_time()`), а очистка происходила перед заполнением новыми данными (см. функцию `clear()` в начале `display_date_dayofweek ()`), вызывать функции нужно в обратном порядке: сначала выводим дату в нижнюю строку, потом время в верхнюю:

```
void setup() {
delay (500);
  OLED1.begin(16,2); //16 символов 2 строки
  OLED1.clear();
 Wire.begin();
 RTC.begin();
/* строка для настройки времени и даты из компьютера:*/
// RTC.adjust(DateTime(__DATE__, __TIME__));
 if (! RTC.isrunning()) {
 OLED1.setCursor(0,0); //верхняя строка, нулевая позиция
 OLED1.print("RTC NOT running!");
 while (1); //зацикливаем программу при ошибке
} else {
  display_date_dayofweek(); //именно в порядке дата потом время
  display_time();
  }
} //конец setup
```

Наконец, в главном цикле мы проверяем текущую минуту, сравниваем ее с предыдущей и обновляем дисплей:

```
void loop() {
  DateTime clock = RTC.now();
  if (clock.minute()!=old_minutes) //обновление каждую минуту
  {
   display_date_dayofweek();
   display_time ();
  }
}
```

Результат работы программы показан на рис. 21.14.

Рис. 21.14. Часы на дисплее WEH1602B

При использовании дисплея для вывода метеорологических данных (см. рис. 21.9 или рис. 21.17 далее), на него приходится выводить действительные числа с десятичной точкой. Чтобы они занимали строго свои места, следует, во-первых, не забывать задавать в операторе `print()` количество знаков после запятой, во-вторых, отрабатывать ситуацию, когда количество знаков перед запятой меняется, включая и изменение знака числа. Обычный вывод предполагает, что знак плюс у положительного числа не выводится, и ведущие нули отбрасываются, поэтому значение «1.1» займет три позиции, а значение «–10.1» — пять, и эти изменения будут мешать при выводе других величин. Чтобы этого не случалось, температура выводится по такому образцу:

```
OLED1.setCursor(0,0); //нулевая позиция, верхняя строка
if (tmpr<10.0) OLED1.print(" "); //если меньше 10, то пробел
                                 //вместо первого знака
if (tmpr>0.0) OLED1.print("+"); //если больше 0, то знак плюс
OLED1.print(tmpr,1); //температура с одним знаком после запятой
OLED1.print("\337C "); //значок градуса
```

При такой организации вывода дисплей можно никогда не очищать целиком, а просто выводить новые величины в заданные позиции.

Графические дисплеи Winstar

Winstar выпускает графические дисплеи на основе тех же матриц, что и строчные, при этом у них один и тот же управляющий контроллер WS0010. Легко подсчитать, что в дисплее с 20 символами в строке, если каждый имеет размер 5×7 пикселов, по горизонтали умещается ровно 100 пикселов. В 2-строчных дисплеях на каждый символ по вертикали отводится 7 пикселов, а матрица имеет в высоту 16 пикселов, итого в текстовом режиме в каждой строке остается лишний ряд незадействован-

ных точек. Иными словами, графические дисплеи Winstar отличаются от текстовых только отсутствием физических промежутков между символами.

100×16 — предельный размер массива пикселов, которым может управлять WS0010. Поэтому дисплеи 20×4 управляются двумя такими контроллерами, как и соответствующий им графический дисплей 100×32. Только в текстовом режиме управление двумя контроллерами для пользователя происходит скрыто (он просто задает нужную строку), а в графическом ему приходится «рулить» ими самостоятельно. Поэтому графический дисплей 100×32 управляется несколько иначе, чем другие, и мы его здесь не рассматриваем. А вот дисплей 100×16 может легко управляться теми же библиотеками и функциями, что и строчные разновидности, с некоторыми добавками, которые мы сейчас и рассмотрим.

Поскольку русский нам тут не нужен, можно без опаски применять стандартную LiquidCrystal прямо из поставки Arduino, которая, как выяснилось, в графическом режиме работает безупречно. К ней только нужно добавить функцию переключения в графический режим `setGraphicMode()`. Библиотеку на такой мелкий случай перелопачивать не станем, а просто будем вставлять эту функцию в каждый скетч:

```
. . .  .
#define LCD_SETGRAPHICMODE 0x1f
LiquidCrystal lcd(9, 4, 8, 7, 6, 5);
void setGraphicMode(){
  lcd.command(LCD_SETGRAPHICMODE);
}
```

Как вы можете заметить, инициализация библиотеки соответствует схеме на рис. 21.13. И действительно, на том рисунке не зря указан дисплей WEH10016 — любой графический дисплей может переключаться в текстовый режим, и наоборот. Впрочем, как мы увидим далее, использование их не по назначению дает не слишком красивые результаты.

Для того чтобы пользоваться графическим режимом, потребуется функция установки графического курсора:

```
void setGraphicCursor( uint8_t x, uint8_t y ){
    if( 0 <= x && x <= 99 ){
      lcd.command(LCD_SETDDRAMADDR | x);
    }
    if( 0 <= y && y <= 1 ){
      lcd.command(LCD_SETCGRAMADDR | y);
    }
}
```

Константа `LCD_SETDDRAMADDR` определена в библиотеке LiquidCrystal. Дисплей 100x16, как и текстовый, делится на две строки: 0 и 1 (по 8 пикселов в высоту каждая), потому *y* здесь может принимать только два значения. А горизонтальная координата *x* варьируется от 0 до 99.

Далее по установленной координате командой `lcd.write()` посылается байт, отдельные биты которого определяют светящиеся позиции вертикальной линии высотой в 8 точек. Крайняя левая позиция в верхней строке имеет координаты 0,0, крайняя правая в нижней — 99,1. Причем верхней точке будет соответствовать младший бит, а нижней точке — старший.

Попробуем закодировать вывод каких-нибудь символов произвольного начертания. Для полных таблиц шрифтов, конечно, целесообразно применять специальные редакторы (которых не меньше миллиона разной степени самодеятельности), но 10 цифр с нужным порядком битов быстрее обработать вручную, тем более что автоматически создаваемые шрифты часто все равно приходится дорабатывать руками. Для удобства кодирования расчертите на бумаге в клетку табличку, соответствующую размеру шрифта (например, 10×16), и закрасьте области, которые должны светиться, а потом запишите это в виде кода по столбцам (рис. 21.15).

	0	1	2	3	4	5	6	7	8	9
0		x	x	x	x	x	x	x	x	
1	x	x	x	x	x	x	x	x	x	x
2	x	x							x	x
3									x	x
0									x	x
1								x	x	
2							x	x		
3						x	x			
0					x	x				
1				x	x					
2			x	x						
3		x	x							
0	x	x								
1	x	x								
2	x	x	x	x	x	x	x	x	x	x
3	x	x	x	x	x	x	x	x	x	x
стр.0	0x06	0x07	0x03	0x03	0x03	0x83	0xc3	0x63	0x3f	0x1e
стр.1	0xf0	0xf8	0xcc	0xc6	0xc3	0xc1	0xc0	0xc0	0xc0	0xc0

Рис. 21.15. Таблица кодирования цифры 2 в графическом режиме

Результат кодирования записывается в двумерный массив:

```
const byte Data2[2][10]={{0x06,0x07,0x03,0x03,0x03,0x83,0xc3,0x63,0x3f,0x1e},
{0xf0,0xf8,0xcc,0xc6,0xc3,0xc1,0xc0,0xc0,0xc0,0xc0}};
```

Для каждой цифры 0–9 создается отдельный такой массив Data0, Data1, Data2 и т. д. Для часов, кроме цифр, потребуется еще двойная точка. Ее можно сделать покороче:

```
const byte DataDP[2][2]={{0x70,0x70},
                         {0x1c,0x1c}};//двойная точка
```

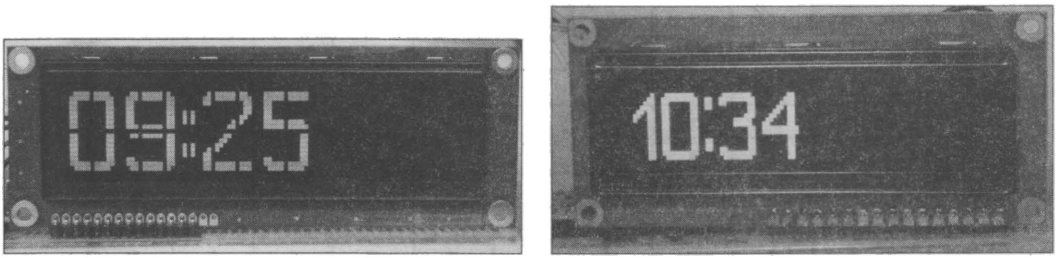
Так как в графическом режиме функция `blink()` не работает, то мигать двоеточием придется программно. Гасить двойную точку можно и просто выводом нулей в соответствующие позиции, но для единообразия я сделал отдельный массив для гашения:

```
const byte DataDPclr[2][2]={{0x00,0x00},
                            {0x00,0x00}};//очистка дв. точки
```

Для вывода каждой цифры (и отдельно для двойной точки) пишется функция, в которой задается горизонтальная позиция `x` в пикселах от левого края экрана:

```
void draw2 (byte x) //вывод "2"
{
 for (byte i = x; i<x+10; i++){
 setGraphicCursor(i, 0);
 lcd.write(Data2[0][i-x]);
 setGraphicCursor(i, 1);
 lcd.write(Data2[1][i-x]);}
}
```

Результат работы программы часов с символами, построенными по такому принципу, на текстовом дисплее WEH1602B в графическом режиме показан на рис. 21.16, *а*. Полный текст скетча можно скачать отсюда: **http://revich.lib.ru/AVR/WEH1602_Graf.zip**.



а *б*

Рис. 21.16. Отображение часов на текстовом дисплее WEH1602B в графическом режиме (*а*) и на графическом дисплее WEH10016 (*б*)

Как видим, получилось не очень красиво — мешают разделительные полосы между текстовыми символами. Потому следует подключить все-таки графический дисплей WEH10016, тогда картинка будет нормальная (рис. 21.16, *б*).

С этими картинками связан один нюанс: у графического дисплея WEH10016A разъем расположен вверху экрана, а у текстового WEH1602B — внизу. Поэтому при простой замене надпись перевернется. И тем не менее, на картинках вы видите одинаковое нижнее расположение разъема, т. е. на дисплее 100×16 картинка перевернута. В чем тут секрет? Здесь такой трюк сделан специально, чтобы проиллюстрировать возможности управления графическим режимом. Чтобы перевернуть картинку и тем самым расположить разъем в удобном месте, не надо переписывать

массивы, кодирующие изображения, — для этого достаточно в функциях вывода «отзеркалить» горизонтальную координату и поменять порядок вывода битов. Скетч часов, использующий такой прием, можно также скачать с сайта автора: **http://revich.lib.ru/AVR/100x16_Clock.zip**.

Кстати, нет проблем запустить в текстовом режиме и графический дисплей. Но при этом символы, лишенные вертикального разделителя, сольются, и будет это выглядеть еще хуже, чем так, как показано на рис. 21.16, *а*.

На рис. 21.17 показан дисплей метеостанции с датчиком скорости и направления ветра, который использует два графических дисплея: один для часов (вторая строка снизу), второй (выше первого) для отображения данных датчиков ветра. Как видите, графические дисплеи предоставляют бóльшую гибкость в части выбора шрифтов и объединения их с картинками на одном дисплее.

Рис. 21.17. Дисплей метеостанции с датчиком скорости и направления ветра

Заметки на полях

При применении графических дисплеев для отображения символов надо учесть, что разместить полный шрифт в памяти контроллера будет затруднительно. Десять цифр шрифта 10×16 по 20 байтов займут в памяти 200 байтов — около 10% ее объема (а более широкие шрифты еще больше). Полный одноязычный шрифт такого размера вместе с цифрами, без учета всяческих знаков препинания и спецсимволов, содержит от 62 (английский) до 74 (русский без Ё) символов и займет почти половину оперативной памяти ATmega328. Очевидно, одноязычного шрифта не хватит, а значит, не хватит и памяти. Потому манипуляции с массивами и функциями вывода раздельно для каждого символа придется отменить, и делать, как положено. То есть шрифты оставлять в программной памяти и загружать через специальную директиву `PROGMEM` (как это сделано в библиотеке openGLCD), все рисунки глифов оформлять в виде единого массива шрифта и загружать для вывода по номеру символа в единой таблице. В противном случае не только памяти не хватит, но и код программы раздуется до неуправляемого объема. Здесь мы на этом останавливаться не будем, т. к. это отдельная и довольно громоздкая задача.

I²C-интерфейс для дисплеев Winstar

При рассмотрении дисплея MT-12864J, который занимает 14 (из имеющихся 19) цифровых выводов стандартного Arduino, мы упоминали, что можно сократить число соединений, если использовать сдвиговые регистры или дешифраторы двоичного кода и, соответственно, самостоятельно переписать имеющиеся библиотеки. Для случая дисплеев Winstar, однако, есть готовое изящное решение, которое позволяет сократить число необходимых выводов с шести до двух. Это так называемый *расширитель* (экспандер) портов на основе шины I²C под названием PCF8574.

Вообще-то микросхема PCF8574 может быть приспособлена к большому количеству применений (считывание кнопок, засвечивание светодиодов и т. п.). В режиме записи микросхема при этом напрямую транслирует значение битов в переданном по шине I²C байте в состояние восьми выходов (при чтении, наоборот, состояние восьми линий передается в передаваемый байт). I²C-адрес в PCF8574 может меняться установкой уровня на трех входах задания адреса, так что с помощью одной шины по двум проводам можно устанавливать или читать состояние до 64 линий.

Одно из самых распространенных применений этой микросхемы — управление ЖК- или OLED-дисплеями по шине I²C. Для этого выпускаются специальные модули (рис. 21.18), на одной стороне которых установлен штыревой разъем типа PLS, и разводка которого позволяет напрямую стыковать такой модуль с ЖК- или OLED-дисплеем. При этом вывод номер 1 (как вы видите на рисунке, отсчет ведется от входного разъема) должен совпадать с выводом 1 модуля дисплея (см. рис. 21.13).

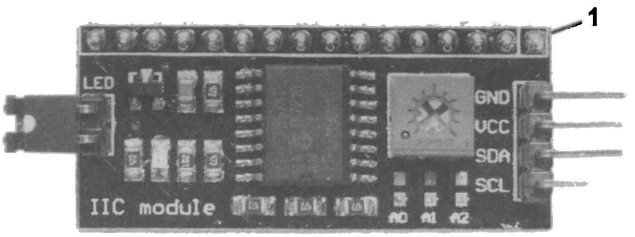


Рис. 21.18. Модуль для управления дисплеями по шине I²C

Разводка выводов для случая OLED-дисплея приведена в табл. 21.2. Если вы хотите вместо готового модуля самостоятельно подключить микросхему (что позволит уменьшить габариты), но собираетесь использовать описанную далее библиотеку, то по этой таблице можно определить правильное подключение микросхемы.

Отметим, что фактически здесь используется четырехпроводная схема подключения и биты DB0–DB3 не подключены никуда, поэтому обозначены серым цветом. Подстроечный резистор, который виден на плате модуля (см. рис. 21.18), предназначен для управления подсветкой в ЖК-дисплеях (выводы 15 и 16) и в нашем случае не задействуется.

Под этим резистором на плате имеются контакты A0, A1 и A2 для задания младших битов I²C-адреса. По умолчанию они подключены к высокому уровню, поэтому

Таблица 21.2. Разводка выводов при подключении PCF8574 и модуля на ее основе к OLED-дисплею

Дисплей		Модуль	PCF8574 (DIP16, SO16)	
GND	1	1	8	
Vcc (5V)	2	2	16	
–	3	3		
RS	4	4	4	P0
R/W	5	5	5	P1
E	6	6	6	P2
DB0	7	7		
DB1	8	8		
DB2	9	9		
DB3	10	10		
DB4	11	11	9	P4
DB5	12	12	10	P5
DB6	13	13	11	P6
DB7	14	14	12	P7

адрес имеет наибольшее возможное значение из заданного диапазона. Микросхемы PCF8574 выпускаются в нескольких модификациях, отличающихся этим диапазоном. Для PCF8574 без буквенного индекса (или PCF8574T) адрес по умолчанию будет равен 0x27, и может меняться в меньшую сторону до 0x20. У PCF8574A адрес по умолчанию равен 0x3F и меняется до 0x38. Следует заметить, что в принципе могут встречаться I^2C-устройства, у которых установленный на производстве адрес может совпадать со значениями из этих диапазонов, так что перед подключением еще каких-то устройств на шину необходимо проверять значения их адресов.

Для работы с дисплеем, подключенным через PCF8574 по I^2C-интерфейсу, имеется рекомендованная библиотека под названием LiquidCrystal_I2C (см. официальный сайт **arduino.cc**). Разумеется, как и оригинальная LiquidCrystal, она работает только с английским языком. Русскоязычных версий ее не имеется (по крайней мере, таких, которые бы уверенно работали в современных версиях Arduino IDE), и вариант для работы с OLED-дисплеем тоже отсутствует. Поэтому автор взял на себя труд доработки, приняв за исходный самый простой из вариантов LiquidCrystal_I2C (**https://github.com/marcoschwartz/LiquidCrystal_I2C**).

Исправленную и дополненную версию под названием LiquidCrystal_I2C_OLED можно скачать отсюда: **http://revich.lib.ru/AVR/LC_i2cOLEDRus.zip**. В ней библиотека дополнена функцией `outStr()`, которая принимает русские символы в строке. Если строка не содержит русских букв, то ее следует выводить обычной функцией `print()`, которая работает быстрее. Именно для этой библиотеки в табл. 21.1 не только значок градуса, но и буквы «ё» и «Ё» помечены серым цветом — их вы-

водить можно только указанием восьмеричных кодов. Вот так выводится пример с русскими и английскими символами вперемешку, а также со значком градуса:

```
OLED1.setCursor(7, 0); //середина верхней строки
OLED1.outStr("Проба\337C");//вывод вперемешку, символ '°C' - английский
OLED1.setCursor(7, 1);//середина нижней строки
OLED1.print("-22,3\337C"); //"-22,3°C"
```

Примеры вывода имеются в папке examples (не забудьте сменить адрес микросхемы PCF8574, если у вас версия, отличная от PCF8574A). Подчеркнем, что эта русифицированная библиотека (как и те, что указаны ранее) предназначена для работы в последних версиях Arduino IDE (начиная примерно с 1.6.1 и далее). В среде Arduino IDE 1.0, а также в других редакторах, работающих в однобайтовой кодировке Win1251 (ANSI), эти библиотеки прямой русский текст в строке будут выводить с ошибками. В этом случае следует пользоваться функцией `print()` с указанием восьмеричных кодов русских букв, согласно табл. 21.1.

Передача данных по радиоканалу

Из всех довольно многочисленных способов дистанционной передачи данных мы рассмотрим два: с помощью радиомодулей Xbee фирмы Digi, работающих на частоте 2,4 гигагерца, и посредством простых передатчиков/приемников диапазона 433 МГц. Оба они имеют свои достоинства и недостатки: Xbee-модули дороги, капризны к питанию, потребляют большой ток, довольно сложны в обращении и к тому же в стандартном варианте действуют на меньшем расстоянии, чем передатчик/приемник 433 МГц. К тому же передача через Xbee-модуль занимает последовательный порт контроллера, что усложняет загрузку программ.

В довершение модуль Xbee работает от напряжения 2,7–3,3 вольта, потому при совместном применении с платами Arduino Nano и Mini придется озаботиться сопряжением логических уровней, а также отдельным стабилизатором. Зато собственно процесс связи приборов друг с другом здесь предельно упрощен: посылка и прием данных осуществляется через стандартную библиотеку Serial, входящую в Arduino IDE по умолчанию, потому тут доступны любые форматы и объемы данных.

В отличие от этого способа, пара передатчик/приемник 433 МГц предельно дешева, очень проста в подключении, экономична и работает от любого напряжения питания вплоть до 12 В. При максимальном напряжении питания проверенная в опыте дальность уверенной передачи на открытом пространстве не меньше 100 метров и мало зависит от наличия не слишком плотных преград — стекла или стенок из дерева, гипсолита, сухой штукатурки и т. п. (кусты и кроны деревьев в этом диапазоне не мешают вовсе). Эта простота оборачивается довольно сложными алгоритмами фильтрации шума, и, как следствие, необходимостью ограничений по форматам и объему передаваемых данных.

Рассмотрим оба эти способа по очереди.

Подключение и настройка Xbee-модулей

Подключение Xbee-модулей осваивается далеко не с полпинка. Главная трудность в правильном проведении их предварительной настройки, а также в том, что для коммуникации с контроллером они используют тот же последовательный порт, что и USB-соединение с компьютером (собственно, Xbee-модуль представляет собой как бы продолжение UART в замену проводному кабелю). Поэтому Xbee-модуль будет мешать не только коммуникации с компьютером, но и процессу программирования платы, отчего перед каждой модификацией программы его придется извлекать из схемы и потом вставлять заново.

> ***Подробности***
>
> Извлечения Xbee-модуля при каждой загрузке программы можно, конечно, избежать, если использовать Arduino Mega, где UART-портов несколько. Но в реальных проектах, где стараются размеры модуля контроллера минимизировать, это малоприменимо.
>
> Из ситуации можно вывернуться и еще одним способом — попросту организовать программный UART на свободных выводах Arduino Uno. О том, как это делается, см. **http://arduino.ua/ru/prog/SoftwareSerial**. Вариант этого способа — применить для общения датчика со станцией на частоте 2,4 ГГц не Xbee, а, например, беспроводной модуль на основе nRF24L01+[1]. Он предназначен в принципе для тех же целей, но управляется не через UART, а через SPI. Как видите, платформа Arduino дает большое разнообразие способов решения для любых пришедших в голову идей.
>
> Кстати, если приобрести третий Xbee-модуль, настроить его на совместную работу с остальными и подключать к компьютеру через специальный Xbee-адаптер, то через него можно не только устанавливать часы, как через обычный UART, но даже и программировать контроллер по беспроводному каналу. Базовый материал для этой функции есть в статье о настройке Xbee, которая упомянута в *разд. «Подключение и настройка Xbee-модулей»* далее.

Для настройки пары Xbee-модулей на совместную работу их нужно прошить через фирменную программу. Для этого можно использовать специальную плату адаптера XBee-USB или универсальную Wireless Shield. Подробное описание последней можно найти на украинском сайте Arduino[2]. Там же имеются и рекомендации по подключению Xbee-модулей. Обычно контроллер из платы Arduino перед прошивкой Xbee-модулей через Wireless Shield рекомендуют извлекать, но можно поступить иначе — в упомянутом разделе украинского сайта приведена программа-заглушка, которую перед настройкой следует закачать в Arduino, чтобы контроллер не мешал прошивке модулей. В таком случае ничего извлекать не придется. Программа очень проста и состоит из двух строк:

```
void setup() { }
void loop() { }
```

Напомним, что в обратной ситуации (т. е. при прошивке Arduino в присутствии Xbee-модуля) столь простого решения не существует — при необходимости изме-

---

[1] Об этом модуле см., например, тут: **http://mk90.blogspot.ru/2013/12/freeduino-wireless-2.html**.

[2] См. **http://arduino.ua/ru/hardware/WirelessShield**.

нить программу Xbee-модуль придется извлекать каждый раз. В этом неудобство применения таких модулей — они-то прошиваются один раз, а рабочую программу приходится долго отлаживать. Потому стоит постараться отладить все возможное до установки Xbee-модуля, и напоследок оставить только беспроводные функции.

Отдельная от контроллера прошивка с применением адаптера XBee-USB описана в разделе Wiki сайта **Amperka.ru** в статье «Настройка пары модулей XBee Series 2 для коммуникации друг с другом»[1]. Там же изложен порядок настройки с помощью фирменной программы (а не подачи AT-команд вручную). Статья довольно старая, потому следует иметь в виду, что все сказанное далее в этой главе относится к модулям Series 2 (на плате помечены буковками S2), а, возможно, еще встречающиеся в некоторых магазинах более дешевые модули Series 1 настраиваются несколько иначе и несовместимы с Series 2.

Из статьи узнаем, что специальную программу X-CTU надо скачать с сайта производителя. Отправившись по указанной ссылке, скачиваем последнюю версию (на момент написания этих строк: X-CTU ver. 5.2.8.6) и устанавливаем ее. Запускаем X-CTU, первым делом выбираем COM-порт, соответствующий Arduino, и для проверки жмем кнопку **Test/Query** (напоминаю, что в Arduino перед этим уже должна быть загружена программа-заглушка или контроллер извлечен из платы). Далее переходим на вкладку **Modem Configuration** и скачиваем свежие версии прошивок через кнопку **Download new versions** (что будет выполняться довольно долго — программа скачивает обновления для всех устройств, которые в ней предусмотрены).

И, наконец, выполняем, как описывается в статье, собственно прошивку для двух Xbee-модулей: одного в режиме «координатора», другого — «роутера». Сначала выбираем тип модема, определенный программой через **Test/Query** (в нашем случае **XB24-B**), затем в поле **Function Set** находим пункт **ZNET 2.5 COORDINATOR AT** (не ошибитесь! Там много пунктов с похожими названиями). Кроме этого, устанавливаем идентификатор сети в графе **PAN ID** (рис. 21.19). Он должен быть одинаковым для обоих модулей, пусть у нас он будет равен **0FFF**. Можно установить скорость обмена в пункте **Serial interfacing/BD-BaudRate**, однако по умолчанию там и без того установлено значение 3 (9600). Потом на всякий случай поставьте отметку в пункте **Always update firmware** (при повторной прошивке этого можно не делать) и, наконец, нажмите кнопку **Write**.

Для второго модуля в поле **Function Set** находим **ZNET 2.5 ROUTER/END DEVICE AT** и делаем все то же самое, только еще в пункте **Sleep Modes/SM-Sleep Mode** выбираем значение **1 - PIN HIBERNATE** (это нам понадобится для установки режима энергосбережения в выносном датчике). Кроме этого, советуют в пункте **Serial Interfacing/D7-DIO7 Configuration** установить значение 0 (**CTS flow control disabled**). После этого устройство конфигурируется именно как END DEVICE, и нам станет доступен Sleep-режим, который активируется установкой логической 1 на выводе 9 модуля.

---

[1] См. **http://wiki.amperka.ru/беспроводная-связь:настройка-xbee-series-2?s[]=xbee**.

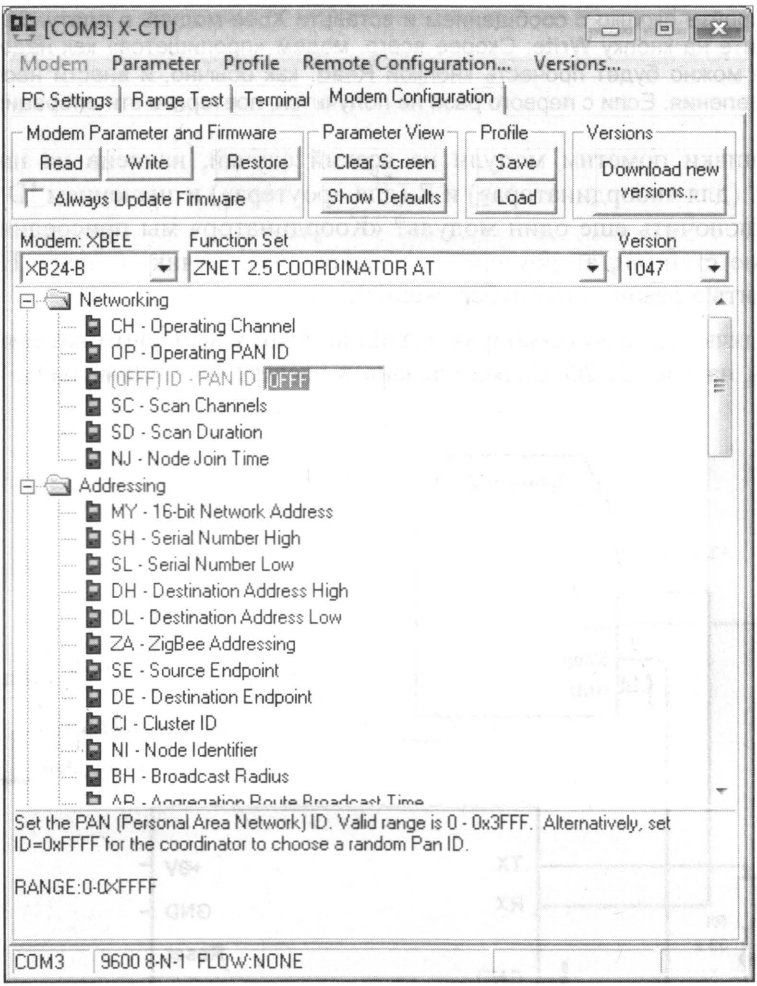


Рис. 21.19. Установка параметров Xbee-модуля в программе X-CTU

Обязательно сохраните обе конфигурации (**Profile | Save**). При повторном подключении модулей можно прочесть зашитую в них конфигурацию, нажав кнопку **Read**, и в случае необходимости что-то подправить.

Совет

Xbee-модули фирмы Digi имеют весьма капризный характер. Если у вас в процессе настройки модуль перестал откликаться на запросы программы X-CTU, то не кидайтесь сразу выбрасывать довольно дорогой девайс, полагая его испорченным. Помогает следующая шаманская последовательность действий, проверенная мной на практике:

1. Извлеките из платы Wireless Shield (или из адаптера XBee-USB, если вы используете его) «неисправный» Xbee-модуль и подключите ее к компьютеру.
2. Запустите X-CTU, сразу перейдите на вкладку **Modem Configuration**, загрузите любую из сохраненных конфигураций (**Profile | Load**) и попробуйте загрузить ее в устройство через кнопку **Write**. Естественно, вы получите сообщение о том, что никакого модема не обнаружено.

3. Закройте окошко с сообщением и вставьте Xbee-модуль в плату. Затем снова нажмите на кнопку **Write**. Скорее всего, модем «пропишется» как надо. После этого его можно будет прочесть кнопкой **Read**, как обычно, и внести необходимые исправления. Если с первого раза не получится, повторите эти операции.

После прошивки пометим модули на всякий случай, наклеив на них «лейблы» с буквами C (для «координатора») и R (для «роутера») и значением ID — вдруг мы захотим подключить еще один модуль? «Координатор» мы присоединим к основному модулю станции, а «роутер» — к выносному датчику (только END DEVICE можно вводить в режим энергосбережения).

Вариант подключения Xbee-модуля к Arduino Mini в выносном датчике метеостанции показан на рис. 21.20. Схема сделана в расчете на минимальное потребление

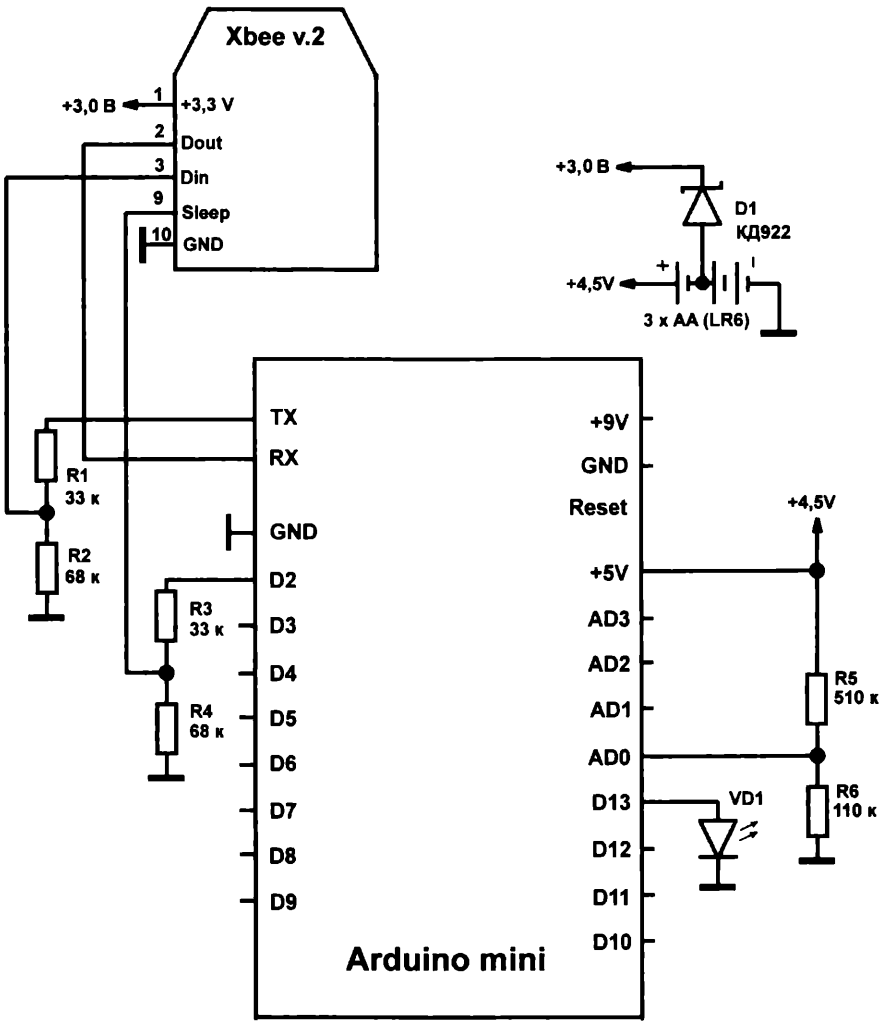


Рис. 21.20. Экономичное подключение Xbee-модуля к Arduino Mini

и отличается от обычного подключения наличием линии Sleep (контакт 9 платы Xbee-модуля), по которой модуль «загоняется» в режим низкого потребления в паузах между измерениями. Выходы Arduino подключены к модулю через согласующие делители R1/R2 и R3/R4 с довольно большим сопротивлением, — без согласования ток через эти выводы резко возрастет, что нехорошо и для контроллера, и для датчика, и с точки зрения потребления энергии. О других особенностях этой схемы (в частности, о хитром включении батареек) мы поговорим далее в *разд. «О режиме энергосбережения, Watchdog-таймере и питании метеостанции»*.

Как уже говорилось, никаких специальных библиотек для модуля Xbee не требуется — и посылка и чтение данных осуществляются так, как будто это происходит через последовательный порт, при этом доступны все функции библиотеки Serial.

Укажем, как правильно организовать в приемнике (например, главном модуле станции) прием данных с ожиданием. Для установления факта приема служит функция `Serial.available()`, которая выдает отличное от нуля значение, если в буфере последовательного порта появились данные. Если разновидностей данных много и они поступают в разное время (или из разных источников — из Xbee-модуля или обычного порта), то каждую из них следует пометить своей сигнатурой. В простейшем случае сигнатурой может служить буквенный символ английского алфавита или цифра. Тогда прием с ожиданием в главном цикле программы `loop()` будет выглядеть как последовательность операторов `if...else`:

```
if (Serial.available())
     {
     command = Serial.read();
     if (command == 79) //символ O - прием с Xbee
{
<принимаем данные с беспроводного канала>
     } else if (command == 68) //символ D - установка часов
       {
         clock_USB_set(); //функция установки часов
     } else if (command == 82) //символ R - вывод времени
       {
       printTime(); //выводим время через последовательный порт
     } // end if command
 } //end Serial.available
```

Поскольку прием относительно редкое событие, то вне этого цикла можно сделать еще много разных вещей — расшифровку и пересчет полученных данных, вывод их на дисплеи и т. п.

Подключение передатчика и приемника 433 МГц

Здесь мы остановимся на самом простом и дешевом способе связи — передатчиках и приемниках диапазона 433 МГц. Комплект не имеет сложившегося общего наименования (каждая фирма называет его по-своему) и выглядит примерно так, как показано на рис. 21.21 (*слева* — передатчик, он меньше по размеру и отличается характерной «таблеткой» частотного фильтра, *справа* — приемник). Изредка встре-

чаются модификации на 315 МГц, внешне они не отличаются, но требуют антенны другой длины (см. далее). Модули могут иметь и другое оформление, но принцип работы у них один и тот же — это простые аналоговые схемы радиодиапазона, не содержащие никаких управляющих контроллеров и других «умных» микросхем. Сами по себе эти модули могут только одно: зафиксировать приемником перепад логического уровня, который был подан на вход передатчика.

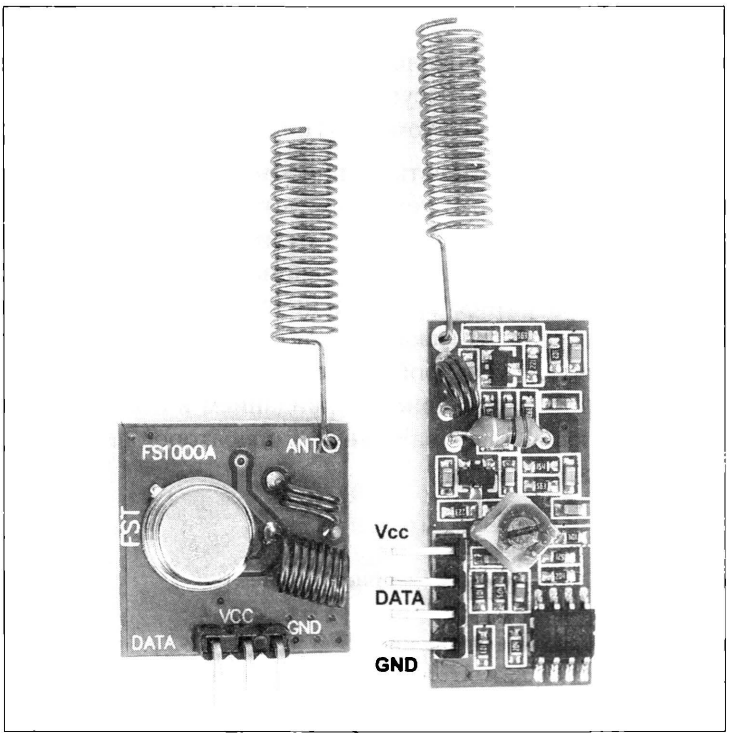


Рис. 21.21. Передатчик (*слева*) и приемник (*справа*) диапазона 433 МГц

Это простое действие очень сильно осложняется помехами в эфире. Если вы попробуете просто подключить передатчик, например, к мультивибратору, который периодически выдает прямоугольные импульсы, то в приемнике ничего даже близко похожего не получите: на его выходе будет более-менее равномерный шум с отдельными малозаметными всплесками. Потому в рассматриваемых нами простейших модулях на 433 МГц вся тяжесть по обеспечению надежной передачи ложится на сложные программные алгоритмы, заложенные в библиотеках.

Кроме того, модули 433 МГц «заточены» под передачу битовых последовательностей и изначально не понимают никаких других форматов. Потому библиотеки для работы с ними просто формально разбивают полученную последовательность битов на байты, а извлечение нужных форматов данных из такой последовательности ложится на плечи пользователя. Иными словами, единственным форматом, которым могут оперировать передатчик и приемник, служит байтовый массив, и любые другие форматы должны быть приведены к этому виду.

Подключим передатчик и приемник к разным платам Arduino и попробуем что-нибудь передать и принять. Максимальная скорость передачи таким способом может составлять несколько тысяч битов в секунду, но мы будем устанавливать не слишком высокую — от этого зависит дальность и надежность передачи.

Подробности

На рис. 21.21 передатчик и приемник снабжены фирменными антеннами, которые обычно продаются отдельно. Антенну нужно подключать обязательно — дальность передачи возрастает радикально. Но совсем необязательно покупать фирменные — антенны несложно соорудить самостоятельно из отрезка одножильного провода длиной 17 см, считая от места пайки. Для диапазона 315 МГц такая же антенна должна иметь длину 24 см. Если хотите обеспечить максимальную дальность передачи, то лучше их не скручивать в спираль (для точной подгонки нужно знать диаметр и шаг витков), а оставить прямой отрезок. Причем предпочтительно, чтобы у передатчика и приемника эти отрезки были ориентированы в пространстве параллельно друг другу.

Передатчик работает от питания 3–12 вольт, и чем выше напряжение, тем мощнее сигнал. Поэтому его следует подключать непосредственно к нестабилизированному напряжению внешнего источника (контакт Vin платы Arduino) с напряжением не менее 9–12 вольт. Как это сделать наиболее экономичным способом — см. далее *разд. «О режиме энергосбережения, Watchdog-таймере и питании метеостанции»*.

Приемник питается от напряжения 5 вольт, но ни в коем случае не следует питать приемник от внутреннего стабилизатора Arduino! В различных описаниях обмена данными по каналу 433 МГц для простоты приемник подключается к 5-вольтовому выходу Arduino, но это грубая ошибка: уверенный прием в таких случаях возможен на расстоянии нескольких метров в пределах прямой видимости. Поскольку приборы эти чисто аналоговые, то питание приемника должно быть очень хорошо очищено от любых посторонних пульсаций. Установка для приемника отдельного маломощного 5-вольтового стабилизатора (LM2931, LM2950 или аналогичного) непосредственно поблизости от его выводов, с правильными цепями фильтрации на входе и выходе, радикально повышает дальность и надежность передачи.

Схемы мы рисовать не будем ввиду их тривиальности — вывод DATA и у приемника, и у передатчика подключается к любому свободному цифровому порту соответствующего контроллера. Почему-то у приемника обычно этих выводов два, и хотя они чаще всего соединены между собой, правильно подключаться к тому, который ближе в контакту питания Vcc. В программах далее подключение передатчика предполагается к выводу D10, а приемника — D2.

Для работы с RF link мы воспользуемся самой распространенной библиотекой для этой цели под названием Virtual Wire. Ссылку на архив с последней версией можно найти на странице по адресу: **http://www.airspayce.com/mikem/arduino/VirtualWire**.

Для примера попробуем передать число 2731 (целое число, в десять раз большее, чем температура нуля градусов по Цельсию, выраженная в градусах Кельвина). Текст программы передатчика выглядит следующим образом:

```
#include <VirtualWire.h>
char msg[4]; //байтовый массив для передачи
```

```
int xx=2731; //целое число
void setup() {
  vw_setup(1200); //скорость соединения VirtualWire
  vw_set_tx_pin(10);   //вывод передачи VirtualWire
}
void loop() {
  itoa(xx, msg, 10); //преобразование числа в массив char
  vw_send((uint8_t *)msg, strlen(msg)); //передача сообщения
  vw_wait_tx(); //Ждем завершения передачи
  delay(2000); //каждые две секунды
}
```

Здесь непонятное выражение `(uint8_t *)msg` означает, что по адресу массива-строки `msg` извлекается по очереди столько байтов (т. е. величин типа `uint8_t`), сколько насчитает функция определения длины строки `strlen(msg)`. В принципе число типа `int` и занимает четыре байта в памяти. Но здесь будет проще преобразовать число в строку алфавитных символов (функция `itoa`), переслать его в таком виде, а потом на приемном конце совершить обратное преобразование. Это проще, нагляднее, и быстрее выполняется, чем разборка числа на отдельные байты и последующая его сборка из таких «осколков». Кстати, если вы объявите число элементов массива больше реально необходимого (например, `char msg[5]`), то хуже не будет — главное, на приемном конце извлечь именно те байты, которые нужны, и опять составить из них исходное число.

Программа каждые две секунды будет передавать в окружающий мир четырехзначное десятичное число, в данном случае 2731. Скорость передачи выбрана 1200 битов в секунду, что не очень большая величина — нам торопиться некуда, зато ошибочных сообщений будет меньше. В приемнике нужно будет обязательно установить такую же величину скорости:

```
#include <VirtualWire.h>
char str[5]; //исходная строка + нулевой символ
void setup() {
  vw_setup(1200); //скорость соединения VirtualWire
  vw_rx_start();  //готовность приема
  vw_set_rx_pin(2); //вывод приемника VirtualWire
  Serial.begin(9600); //Serial-порт для контроля
}
void loop() {
  uint8_t buf[VW_MAX_MESSAGE_LEN]; //переменная для принятых данных
  uint8_t buflen = VW_MAX_MESSAGE_LEN;  //max длина принятых данных
  if (vw_get_message(buf, &buflen)) //если данные приняты
  {
        for (byte i=0;i<4;i++)  //получить четыре байта
                str[i]= buf[i]; // и упаковать их в строку
      int value=atoi(str); //преобразовать в целое число
      Serial.println(value,1); //выводим
  }
}
```

Здесь полученная символьная строка снова преобразовывается в число (функция atoi). Для проверки отправляем его через последовательный порт на компьютер. Кстати, а что за величина VW_MAX_MESSAGE_LEN? По смыслу названия она обозначает максимально возможную длину сообщения в байтах, на которую мы закладываемся при приеме массива данных, чтобы ничего не потерять по дороге. На самом деле это не совсем так: константа VW_MAX_MESSAGE_LEN равна 30, а максимально допустимая длина сообщения (она носит название VW_MAX_PAYLOAD) на три байта меньше, и равна 27. Три лишних байта, которые присоединяются к любому сообщению, несут в себе служебную информацию.

Учтите, что переменная buflen должна представлять собой локальную переменную, и строка с присваиванием ей максимально возможной длины (uint8_t buflen = VW_MAX_MESSAGE_LEN) обязана присутствовать в каждом цикле приема. В противном случае из-за приема испорченных сообщений значение buflen будет каждый раз укорачиваться, пока вы не начнете получать всякую чушь вместо данных.

При передаче нескольких чисел разного формата лучше всего единообразно приводить их к целым, преобразовывать в строку символов и посылать соединенные строки вместе. Тут может помочь тип данных String, который умеет автоматически преобразовывать любые типы данных и легко соединять их в одну строку. Кроме того, в реальной передаче/приеме длинные сообщения следует предварять сигнатурой, которая дополнительно позволит отсеять ошибочно принятые сообщения. Пусть мы читаем данные из датчика DHT22 (см. *разд. «Датчики температуры и влажности»* в этой главе), тогда их посылка через передатчик может выглядеть следующим образом:

```
. . .
int hum = dht.readHumidity(); //влажность сразу целое число
float t = dht.readTemperature(); //температура
int tmpr = t*10+2731; //температуру с десятыми
                      //в целое положительное число
String strMsg="DAT"; //сигнатура - данные
strMsg+=tmpr; //присоединяем температуру
strMsg+=hum; //присоединяем влажность
strMsg.toCharArray(msg,10); //переводим строку в массив,
               //10 - общее количество знаков с запасом
vw_send((uint8_t *)msg, strlen(msg)); //передача
vw_wait_tx(); //ждем завершения передачи
```

В приемнике необходимо будет провести обратное преобразование, которое проще по сути, но длиннее по количеству команд:

```
. .   . .
char str[5]; //строка для приема чисел
float tempr; //температура
int hh; //влажность
```

```
  if (vw_get_message(buf, &buflen))     //если данные приняты
  {
    vw_rx_stop();
        for (byte i=0;i<3;i++)  //получить первые три байта
                str[i]= buf[i]; //
                str[3]='\0';
      Serial.println(str); //выводим для контроля
      if((str[0]=='D')&&(str[1]=='A')&&(str[2]=='T')) {
//если сигнатура верная, то принимаем:
      for (byte i=3;i<7;i++)  //получить четыре байта температуры
                str[i-3]= buf[i]; //упаковать их в строку
      int value=atoi(str); //преобразовать в целое число
      value=value-2731; //вычесть 2731 - в Цельсия
      tempr=(float(value)/10); //с десятыми
      Serial.print(tempr,1); //выводим температуру
      Serial.print("\260C "); //градусы Цельсия
       for (byte i=0;i<4;i++) str[i]=0; //очищаем строку
        for (byte i=7;i<9;i++)  //получить два байта влажности
                str[i-7]= buf[i]; //упаковать их в строку
      hh=atoi(str); //преобразовать в целое число
      Serial.print(hh,1); //выводим влажность
      Serial.println("% "); //пробел
      }//end if str=DAT
  }//end vw_get_message
```

Здесь мы последовательно разбираем полученный массив байтов. В нем первые три байта (с нулевого по второй) — код сигнатуры «DAT», байты с третьего по шестой — температура, седьмой и восьмой — влажность. Последний байт — запасной, в принципе он не нужен. Если рассмотрите, как формируется посылаемая строка в передатчике, увидите, что этот байт всегда будет равен нулю (как ячейка чистой памяти), потому он может служить в качестве конца строки, если это потребуется.

Обратите внимание на строку `Serial.print("\260C ")`. «\260» — это восьмеричный код символа градуса в кодировке Unicode (шестнадцатеричное `0xB0`, см. Таблицу символов Windows). В результате вы будете получать в окне Монитора порта строки вида:

```
24.0°C 30%
```

О конструктивном оформлении устройств на Arduino

Вопрос об изготовлении корпусов мы достаточно обсудили в *главе 3*, а вот с монтажом готовых и отлаженных устройств придется повозиться. Платформа Arduino идеально приспособлена для макетирования, но довольно слабо — для изготовления законченных приборов. Не следует применять для этой цели беспаечные

макетные платы, даже если это кем-то рекомендуется — они крайне ненадежны при долговременной работе. Но не собирать же конструкцию «паучком», кое-как запихав в корпус модули и соединяющие их провода? Давайте рассмотрим другие способы.

Большинство компонентов для платформы Arduino обычно продается в комплекте с соединительными кабелями, а если они и отсутствуют, то несложно приобрести соединители нужной конфигурации отдельно (исключение составляют разве что дисплеи с большим количеством контактов, для которых кабели приходится изготавливать самостоятельно). Для таких модулей множеством фирм, торгующих компонентами для Arduino, предлагаются переходные и расширительные платы-«шилды» (Shields), которые по идее должны позволить смонтировать любой прибор, не прикасаясь к паяльнику. За время существования платформы сложилось несколько стандартов подключения периферии к Arduino, но далеко не всегда можно подобрать нужные компоненты именно того стандарта, на который рассчитан данный комплект приобретенных «шилдов». Кроме того, полученная таким способом «этажерка» плат не позволяет мечтать о минимальных габаритах, эргономичном дизайне и об оптимизации таких функций, как энергосбережение. Потому этот подход годится только для самых простых устройств и еще может подойти для первичного макетирования или экспериментирования во время обучения.

Оптимальным (в том числе и с точки зрения стоимости) путем для изготовления приборов в единичных экземплярах будет расположение всех компонентов в нужном порядке на универсальной макетной плате «под пайку» (см. рис. 3.4, *слева*), или нескольких таких платах, установленных друг над другом с помощью стоек, и соединение их проводами с помощью пайки, а там, где это возможно — шлейфами из плоского кабеля. Такой подход дает возможность без проблем совмещать в одном приборе отдельные дискретные компоненты, вроде стабилизаторов питания или микросхем — расширителей портов, готовые модули для Arduino и дисплеи, которые все равно никогда не влезают в готовые «шилды» без доработки. Есть и компоненты с экзотическими или нигде больше не встречающимися разъемами — например, SD-карты, — которые, наоборот, удобнее всего подключать через предназначенные для них «шилды». Но в большинстве случаев можно обойтись самыми дешевыми и простыми переходниками, не затрудняя себя поиском конфигурации нужного стандарта.

Разумеется, использовать в качестве управляющего контроллера Arduino Uno при размещении схемы на общей плате нецелесообразно — при необходимости частого перепрограммирования употребляют Nano, в противном случае Mini или Mini Pro (напомним, что у последних целесообразно удалить неотключаемый светодиод по питанию). Для подключения этих контроллеров, как и для большинства дисплеев, на плату устанавливают однорядные разъемы-гнезда типа PBS.

Большей миниатюризации и лучшего энергосбережения можно достичь, если использовать «голый» контроллер ATmega328 в DIP-корпусе, который программируется отдельно через плату Uno и устанавливается на панельку. Соответствие выводов микросхемы ATmega328 и Arduino, как мы уже говорили, можно найти в описании Uno на любом официальном сайте. Отдельная микросхема контроллера с уже

записанным загрузчиком Arduino (ATmega328P-PU with bootloader Arduino Uno) продается, например, в «Чипе-Дипе». Только не нужно забывать, что при таком подходе, кроме стабилизатора питания, придется также разместить на плате необходимые элементы обвязки — прежде всего кварц на 16 МГц с конденсаторами к нему, а также RC-цепочку для надежного запуска при включении (см. *главу 19*). Все это подключается точно так же, как показано на рис. 19.3, только выводы для подключения кварца у ATmega328 будут другими. Кроме всего прочего, комплект из микросхемы с загрузчиком и необходимых дополнительных компонентов обойдется от двух до пяти раз дешевле фирменной платы Arduino.

Единственная трудность, которая подстерегает вас на этом пути, — наличие в вашем проекте микросхем или модулей в корпусах, отличных от DIP. Для небольших планарных микросхем переходники можно вырезать из макетной платы, подобной показанной на рис. 3.4, *справа*, а для модулей с уменьшенным шагом, вроде Xbee, можно либо вырезать переходник из подходящего «шилда», либо изготовить его самостоятельно, приобретя нужные разъемы. Но встречаются и случаи, когда готовую макетку с подходящей для переходника конфигурацией подобрать очень трудно — как, например, для драйверов MAX695*x* в планарном корпусе, содержащих большое количество выводов с шагом 0,8 мм. Вот в этом случае действительно придется заказывать плату либо пытаться изготовить ее самостоятельно. Но все равно, изготовить или заказать один переходник будет гораздо дешевле и проще, чем целую плату всего прибора.

О режиме энергосбережения, Watchdog-таймере и питании метеостанции

В контроллерах ATmega есть несколько режимов энергосбережения («сна»), разница между которыми применительно к Arduino, нас, по большому счету, не волнует. Мы будем применять единственный (самый «крутой») режим Power Down. Гораздо больше тут вопросов возникает по части правильного обращения с внешними портами, питанием и дополнительным оборудованием на плате. Сначала о собственно включении режима «сна».

Для этого достаточно подключить библиотеку sleep.h, которую почему-то нужно вызывать с указанием папки, где она находится:

```
#include  <avr/sleep.h>
```

Собственно включение состояния «сна» заключается в установке режима и вызове функции запуска:

```
set_sleep_mode(SLEEP_MODE_PWR_DOWN); //режим сна
sleep_mode();                        //система засыпает
```

После выхода из сна будет выполнена первая команда после `sleep_mode()`. Заметим, что вызывать `sleep_mode()` можно только из главного цикла программы (или, разумеется, из функций, которые там вызываются). Попытка обратиться к `sleep_mode()` в процедуре обработки прерывания ни к чему не приведет.

Заметки на полях

В сети полно примеров, в которых фигурируют отличающиеся способы запуска с использованием других функций библиотеки sleep.h. Во избежание недоразумений и вопросов («а почему у Пети не так, как у Васи?»), замечу, что последовательность функций `sleep_enable(); sleep_cpu(); sleep_disable()` делает практически то же самое, что единственная функция `sleep_mode()`. Практически потому, что между `set_sleep_mode` и реальным уходом в сон по команде `sleep_mode` может вклиниться какое-нибудь аппаратное прерывание, и уход в сон будет, как минимум, отложен. Чтобы этого не случалось, руководство рекомендует загонять в «сон» следующей последовательностью функций:

```
cli(); //запрещаем прерывания
sleep_enable(); //разрешаем режим сна
sei(); //разрешаем прерывания
sleep_cpu(); //уходим в сон
sleep_disable();//при выходе запрещаем режим сна
```

Так как в Arduino прерывания широко не используются, а немногочисленные действующие (вроде тиков системного таймера) уход в сон надолго не отложат, то мы можем на это не обращать внимания. Заметим также, что иногда встречающаяся последовательность команд `sleep_mode(); sleep_disable();` бессмысленна, потому что запрещение режима сна уже выполняется в первой команде.

Чтобы обеспечить минимум потребления, целесообразно повыключать в контроллере все, что можно. На практике, говорят, достаточно выключить АЦП, поэтому лучше оформить вызов режима сна в виде отдельной функции, которая потом вызывается из главного цикла:

```
void system_sleep() {
  ADCSRA &= ~(1 << ADEN);//выключим АЦП
  set_sleep_mode(SLEEP_MODE_PWR_DOWN); //режим сна
  sleep_mode();//система засыпает
    ADCSRA |= (1 << ADEN);//включаем АЦП
}
```

Как видите, с вызовом «сна» все просто. Сложнее понять, что делать потом — контроллер уйдет в сон навечно, и из этого состояния его надо выводить. Для такого действия есть два способа (не считая, конечно, перезапуска вручную или по таймеру выключением/включением питания). Выход из режима сна может быть осуществлен, прежде всего, по внешнему прерыванию INT0 или INT1. Обычно они вызываются либо кнопкой, либо сигналом какого-то внешнего таймера — в частности, чтобы разбудить контроллер в точно определенный момент времени, удобно использовать выход SQW часов реального времени (см. *разд. «Часы»* в этой главе). Этот способ мы рассматривать здесь подробно не будем, т. к. в Сети полно соответствующих рекомендаций.

Watchdog Timer

Подробнее мы рассмотрим другой способ — с помощью входящего в контроллер устройства, называемого *сторожевым таймером* (Watchdog Timer, WDT). Заметим, что в многочисленных описаниях использование Watchdog применительно

к Arduino обычно сильно переусложнено. Извиняет авторов то, что описание Watchdog — самый запутанный раздел руководств по AVR-контроллерам, и разобраться в нем бывает нелегко даже опытным людям. В реальности пользоваться этой возможностью, как вы сейчас увидите, совсем несложно.

Watchdog Timer представляет собой независимый от остальных устройств контроллера простой двоичный счетчик, который, будучи запущен, работает от собственного генератора частоты и по достижении заданного числа отсчетов может выполнять два действия: либо вызывать связанное с ним прерывание (прерывание `wdt`), либо сразу принудительно перезапускать контроллер так, как будто он включился заново. В первом случае обнуления регистров не происходит, и контроллер начинает выполнять программу с того места, на котором остановился. Поэтому режим с прерыванием используют в основном для выхода из «сна», а режим с полной перезагрузкой — в случае зависания программы.

Случай принудительной перезагрузки мы сначала и рассмотрим, т. к. он проще в реализации. Для этого надо лишь подключить библиотеку `wdt.h` (также с указанием папки: `#include <avr/wdt.h>`), которая содержит ровно три функции: `wdt_enable(timeout)`; `wdt_reset()` и `wdt_disable()`. Первая функция, как ясно из названия, разрешает таймер, который после это начинает непрерывно «молотить», никак не влияя на остальные действия контроллера. Отвлекается он от этого увлекательного занятия только тогда, когда счет дойдет до заданного в параметре `timeout` числа. Число может принимать одно из десяти установленных значений из ряда: `WDTO_15MS`, `WDTO_30MS`, `WDTO_60MS`, `WDTO_120MS`, `WDTO_250MS`, `WDTO_500MS`, `WDTO_1S`, `WDTO_2S`, `WDTO_4S`, `WDTO_8S` (эти константы могут быть заменены просто числами от 0 до 9). То есть задержка срабатывания Watchdog-таймера может быть установлена от 15 миллисекунд до 8 секунд, но только из указанного ряда. Так как генератор таймера представляет собой несложный RC-мультивибратор, то эти значения приблизительные и зависят от питания — указанные величины характерны для напряжения питания 5 вольт, с его снижением реальные задержки увеличиваются.

Простейший порядок использования таймера в режиме принудительной перезагрузки получается, таким образом, следующим: в процедуре `setup()` мы разрешаем таймер с нужной задержкой (например, `wdt_enable(WDTO_2S)`), а в главном цикле `loop()` периодически его сбрасываем с помощью `wdt_reset()`, заставляя начать отсчет заново. Задержка должна заведомо превышать время выполнения главного цикла в самом медленном случае. Если программа зависнет на выполнении какой-то операции, главный цикл прервется, таймер не обнулится вовремя и перезагрузит контроллер.

На этот простейший порядок действий могут накладываться разные усложнения: например, если надо иногда выполнить какую-то длительную процедуру (передать через последовательный порт большое количество данных с ожиданием ответа), то на время ее выполнения сторожевой таймер можно отключать функцией `wdt_disable()`, а потом включать заново. Можно внутри подобной процедуры отдельно периодически вызывать функцию обнуления и т. п. Во всех таких случаях при нормальном выполнении программы наличие включенного сторожевого таймера никак на ней не сказывается.

Заметки на полях

В старых реализациях плат Arduino в загрузчике (bootloader), который всегда присутствует в памяти контроллеров, была допущена ошибка, которая заключалась в уходе контроллера вместо нормального перезапуска в бесконечный цикл перезагрузки, если ее источником был WDT, а не просто включение питания. До сих пор во многих статьях про Watchdog Timer значительное место занимают рекомендации по решению этой проблемы. В платах Arduino (по крайней мере, в оригинальных, а не в клонированных копиях), выпущенных примерно после 2013 года, эти проблемы решены.

Но если вы хотите на всякий случай проверить имеющийся у вас экземпляр Arduino на эту ошибку, то это сделать просто. Составьте проверочный скетч, который в процедуре `setup()` включает сторожевой таймер на максимальное время 8 секунд. В главном цикле вы периодически, например, каждые 4 секунды (с обычной задержкой через функцию `delay()`) посылаете в Монитор порта какое-нибудь сообщение и заодно для наглядности мигаете светодиодом. Если срабатывание таймера приводит к зависанию, то после пары сообщений вывод в Монитор порта остановится, а светодиод либо погаснет, либо начнет быстро-быстро мигать. В этом случае надо отдельным программатором заменить bootloader на корректную версию, причем начальная задержка в 8 с после включения даст вам достаточно времени на проведение этой операции. Подробности здесь излагать нет смысла — они достаточно хорошо описаны во множестве источников.

Второй способ — с вызовом прерывания вместо перезагрузки — несколько сложнее, но только потому, что для него нет готовых библиотек (точнее, есть, конечно, но использовать их нет особого смысла, как вы сейчас увидите). Для этого пишем отдельную функцию, которая запускает Watchdog в режиме вызова прерывания и сразу с нужной задержкой:

```
void setup_watchdog (int timeout) {
  byte bb;
  if (timeout > 9 ) timeout = 9;
  bb= timeout & 7;
  if (timeout > 7) bb|= (1<<5); //в bb - код периода
  MCUSR = 0; //очистка флагов, требуется руководством
  WDTCSR |= (1<<WDCE) | (1<<WDE); // запуск таймера
  WDTCSR = bb; //установка периода срабатывания
  WDTCSR |= (1<<WDIE); //разрешаем прерывание WDT
}
```

Эта функция вызывается либо один раз в процедуре `setup()`, либо после каждой остановки таймера. Для удобства такой остановки и, главное, периодического сброса таймера в начале программы ставим ссылку на библиотеку `<avr/wdt.h>`. В этом случае значение `timeout` можно вводить с помощью тех же констант, что и ранее, а периодический сброс или остановку — с помощью уже знакомых `wdt_reset()` и `wdt_disable()`.

В обработчике прерывания `wdt` надо обязательно что-то сделать, иначе контроллер уйдет в перезагрузку, как и в предыдущем случае. Проще всего выполнить в нем либо сброс, либо, если надо, остановку WDT (только тогда не забыть потом его включить обратно):

```
//прерывание watchdog
ISR (WDT_vect)
{
   wdt_reset();  //можно заменить на wdt_disable()
}  // end WDT_vect
```

Выполнив прерывание, контроллер перейдет к выполнению команды, следующей после той, на которой он остановился перед срабатыванием таймера.

Вот теперь мы можем создать законченный алгоритм ухода в «сон» с периодическим «пробуждением» по сторожевому таймеру. Тестовая программа будет выглядеть так:

```
#include  <avr/sleep.h>
#include  <avr/wdt.h>

const byte LED = 13; \\светодиод
void setup_watchdog (int timeout) {
  byte bb;
  if (timeout > 9 ) timeout = 9;
  bb= timeout & 7;
  if (timeout > 7) bb|= (1<<5); //в bb - код периода
  MCUSR = 0; //очистка флагов, требуется руководством
  WDTCSR |= (1<<WDCE) | (1<<WDE); //запуск таймера
  WDTCSR = bb; //установка периода срабатывания
  WDTCSR |= (1<<WDIE); //разрешаем прерывание WDT
}
void system_sleep() {
  ADCSRA &= ~(1 << ADEN);//cbi(ADCSRA,ADEN); выключим АЦП
  set_sleep_mode(SLEEP_MODE_PWR_DOWN); //режим сна
  sleep_mode();                        //система засыпает
    ADCSRA |= (1 << ADEN);//sbi(ADCSRA,ADEN); включаем АЦП
}

// watchdog interrupt
ISR (WDT_vect)
{
   wdt_reset();  // disable watchdog
}  // end of WDT_vect

void setup () {
  setup_watchdog(WDTO_4S); //4 c
  wdt_reset();
} // end setup

void loop ()
{
  pinMode (LED, OUTPUT);
  digitalWrite (LED, HIGH);
```

```
  delay (50);
  digitalWrite (LED, LOW);
  pinMode (LED, INPUT);
  system_sleep();
} // end loop
```

По этой программе контроллер мигнет светодиодом и уйдет в сон на 4 секунды, после чего все повторится заново. Вы легко можете дополнить этими процедурами программу, которая, например, обрабатывает уличный датчик температуры/влажности и посылает его данные через радиоканал. Получите экономичный выносной датчик, работающий от батареек. Только, как вы сейчас увидите, одного лишь объявления режима энергосбережения недостаточно — чтобы он работал, как надо, следует еще принять продуманные схемотехнические и программные меры. И вот это как раз бывает самое сложное.

О мерах по снижению энергопотребления

Если вы загрузите приведенную тестовую программу в Arduino Uno или Nano, то получите снижение потребления раза в два (примерно с 22 до 11 мА). Это, конечно, здорово, но не то, чего мы добиваемся — режимом энергосбережения это назвать трудно. Надо прежде всего избавиться от всяческой неотключаемой обвязки, что можно выполнить путем применения Arduino Mini или Arduino Mini Pro, особенно если удалить с их платы неуправляемый светодиод, который сигнализирует о подаче питания. Потому в простейшем случае для устройств, работающих от батареек, следует употреблять именно Mini.

> ***Совет***
>
> Но следите за тем, что приобретаете! В оригинальных платах Mini или Mini Pro ставят стабилизаторы MIC5205 или LP2985, собственное потребление которых измеряется микроамперами. А в клонах запросто могут поставить более мощный, но совсем не экономичный AMS1117, который сам потребляет миллиампер 5–10.

Радикально решает проблему собственного потребления контроллера использование «голой» микросхемы ATmega328, запрограммированной отдельно, как о том говорилось ранее. Совместно с ней следует применять малопотребляющий стабилизатор питания (см. *главу 9*). Учтите, что импульсные стабилизаторы здесь не пригодны — так, упомянутый в *главе 9* LM2596 эффективен при конверсии относительно высокого входного напряжения (как, например, от 12-вольтового аккумулятора) в низкое питание контроллера, но собственный ток покоя у него далек от идеала, как и у всех остальных импульсных стабилизаторов вообще.

Если все сделано правильно, то от контроллера можно добиться потребления порядка единиц микроампер (десятков-сотен микроампер, если принимать во внимание стабилизатор питания). Но это еще не все: необходимо проследить, чтобы все внешние порты оказались отключенными. Иначе можно оставить какой-нибудь горящий светодиод, который через включенный на выход порт сожрет всю батарейку. В сложных схемах можно, не глядя, устанавливать все порты в третье состояние (на вход и без «подтягивающих» резисторов), но при выходе из «сна» все равно при-

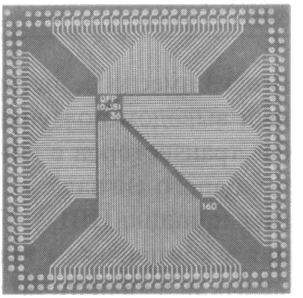
дется их устанавливать по очереди в нужные режимы. Потому организация правильного режима энергосбережения может оказаться очень муторным делом — особенно, учитывая тот факт, что и все остальные компоненты устройства тоже необходимо либо устанавливать в режим энергосбережения, либо отключать у них питание в случае, если они потребляют более нескольких сотен микроампер. Общее среднее потребление (включая время пребывания в активном режиме) уже в 3–4 мА у постоянно включенного прибора следует признать не режимом энергосбережения, а издевательством — не напасетесь батареек каждый месяц менять.

В целях снижения потребления и повышения срока службы батареек можно обойтись вообще без стабилизаторов. Для примера рассмотрим схему на рис. 21.20, на которой хитрое включение батарейного питания ориентировано на достижение энергосбережения в максимальной степени. От трех элементов AA (реальное напряжение около 4,5–4,8 В) питается плата Arduino Mini, а от отвода между вторым и третьим — модуль Xbee (напряжение 2,8–3,2 В). Диод D1 типа КД922 (с переходом Шоттки, т. е. с малым падением напряжения) развязывает источники питания 4,5 и 3,3 В, чтобы они по каким-то причинам не начали работать друг на друга. Если бы мы подключили обычное питание 7–9 В к стабилизатору платы, а модуль Xbee через какой-нибудь отдельный стабилизатор, то теряли бы питание не только на самих стабилизаторах, но и за счет их собственного потребления.

Подробности

Недостаток такой схемы в том, что из-за высокого потребления Xbee-модуля (до 40 мА в момент передачи у обычной, не умощненной версии) «верхняя» батарейка будет расходоваться куда медленнее двух «нижних». На схеме показан высокоомный делитель, подключенный к аналоговому порту A0, который позволяет контролировать через АЦП напряжение питания и посылать его значение вместе с другими данными. Xbee-датчик может работать, согласно документации, при снижении напряжения до 2,2 вольта, т. е. измеренное общее напряжение питания может снижаться примерно до 3,6 вольта. Когда оно снижается ниже этого порога, главный модуль сигнализирует о снижении питания. Тогда для экономии батареек можно временно поменять местами одну из «нижних» батареек с «верхней», что даст дополнительно еще неделю-другую работы.

Если добиться малого потребления от схемы постоянно включенного уличного датчика не удается по принципиальным соображениям, то можно питать ее от аккумулятора, что сократит расходы, но не избавит от хлопот, связанных с перезарядкой. А самым радикальным решением вопроса будет питание такого датчика от комплекта: солнечная батарея — контроллер-зарядник — аккумулятор. Для правильного подбора такого комплекта учтите, что номинальная мощность солнечной батареи в ваттах должна превышать мощность потребления не менее, чем в 6–8 раз, а емкость аккумулятора в ампер-часах в 4 раза численно превышать ту же самую мощность потребления в ваттах. При небольших мощностях потребления в единицы ватт, характерных для электронных устройств, такой комплект может обойтись совсем недорого (см. [27]).



ГЛАВА 22

Применения Arduino

Избранные возможности платформы

> При выходе из комнаты отца юноша увидел свою мать, ожидавшую его с рецептом пресловутого бальзама, применять который, судя по приведенным выше отцовским советам, ему предстояло часто.
>
> *А. Дюма. «Три мушкетера»*

Платформа Arduino может использоваться в самых различных областях. В этой главе рассмотрим подключение к модулям Arduino некоторых популярных внешних устройств. И начнем с темы, которую мы ранее игнорировали: аналоговое управление внешними устройствами, которое в случае Arduino осуществляется методом *широтно-импульсной модуляции* (ШИМ). Оно часто применяется для управления, например, приводами колесных тележек различных роботов или двигателями квадрокоптеров.

Аналоговое управление внешними устройствами (ШИМ)

Метод широтно-импульсной модуляции (ШИМ, pulse-wide modulation, PWM) заключается в очень простом принципе изменения ширины (скважности) прямоугольных импульсов в зависимости от управляющего сигнала. Управление нагрузкой достигается за счет изменения количества энергии: чем шире импульс, тем больше энергии подводится к нагрузке — фактически это равносильно изменению среднего значения питающего напряжения. Нагрузка при этом играет роль этакого фильтра низкой частоты: она реагирует не на каждый импульс, а на некое среднее значение напряжения (и, соответственно, тока) во всей последовательности импульсов. Отметим, что в радиотехнике аналогичный принцип называется фазовой модуляцией, но используется несколько иначе, чем в нашем случае, потому здесь мы ограничимся лишь его упоминанием — называть ШИМ фазовой модуляцией и наоборот было бы неправильно.

Без контроллеров и программирования ШИМ с ручным управлением очень просто реализовать, используя схему транзисторного мультивибратора (см. рис. 16.1, *б*).

В него вместо резисторов R2 и R3 добавляется переменный резистор, подключенный выводом ползунка к питанию, а крайними выводами — к точкам соединения баз транзисторов с конденсаторами. Передвигая ползунок, вы меняете скважность выходного сигнала, не меняя его периода, т. е. делаете именно то, что и предполагает принцип ШИМ. Пределы регулирования тут довольно широкие (см. требования к компонентам такого мультивибратора в *главе 16*), но ограниченны в области малых и, наоборот, больших значениях скважности.

Принцип ШИМ-регулирования

Простая модель ШИМ-регулирования показана на рис. 22.1, *а*. На нем показано, как можно получить изменяющуюся скважность с помощью компаратора и затем обратно выделить модулирующий сигнал из полученного и усиленного ШИМ-сигнала. На один из входов компаратора поступает сигнал опорной частоты пилообразной формы, на второй — модулирующий аналоговый сигнал. На выходе компаратора получается сигнал с изменяющейся скважностью — чем выше значение модулирующего сигнала, тем шире импульс на выходе компаратора. Пропустив ШИМ-сигнал через простой ключевой усилитель и отфильтровав затем высокочастотную составляющую, можно получить исходный модулирующий сигнал, причем усиленный по отношению ко входному — так работают т. н. усилители класса D (они входят практически во все малогабаритные звуковые устройства: плееры, планшеты/смартфоны и т. д.). В сравнении с обычными аналоговыми звуковыми усилителями (см. *главы 8* и *11*) усилители класса D намного экономичнее, т. к. усилитель мощности работает в ключевом режиме, и потери происходят только за счет неидеальности ключей-транзисторов. Отметим, что в правильном усилителе типа D опорная частота имеет порядок сотен килогерц, и никак не влияет на качество звука, но в восьмиразрядном контроллере это, конечно, недостижимый идеал.

Городить ШИМ-управление на микросхемах малой степени интеграции (на том же таймере 555, например) в современных условиях нет никакого смысла — они получатся переусложненными и все равно с досадными ограничениями выходных параметров. Исключение представляют, естественно, специализированные микросхемы такого рода, но все равно дешевле и проще, чем на микроконтроллере, где режим ШИМ уже реализован за вас, все равно не получится.

В цифровом виде тот же принцип реализован во всех таймерах-счетчиках МК AVR серии Mega. На рис. 22.1, *б* показана реализация ШИМ на примере таймера-счетчика Timer 1 любого из младших контроллеров этой серии. Таймер-счетчик вводится в специальный режим fastPWM и начинает работать, как непрерывный реверсивный счетчик от значения 0 до заданного максимума и обратно. Обычно ограничиваются максимумом в 256 градаций (8 битов). Если в регистр сравнения OCR1A заносится некоторое число в этих же пределах, то в момент совпадения содержимого счетчика с ним при счете на увеличение на выходе OC1A появится высокий уровень, который сбросится при том же значении при обратном счете. Иными словами, если на регистр сравнения OCR1A подавать некую числовую последовательность, соответствующую оцифрованному модулирующему сигналу, то на

выходе счетчика получим тот же результат, что на выходе компаратора на схеме по рис. 22.1, *а* — модулированный ШИМ-сигнал.

Очень важно то, что сигнал этот выдается аппаратно и независимо от работы остальных систем контроллера, т. е. в промежутках между изменением состояния регистра сравнения контроллер может успеть выполнить множество других действий, а ШИМ-последовательность на выходе при этом прерываться не будет. Максимальная тактовая частота ШИМ при 8-битовом счете равна $f_{такт}/510$, т. е. для контроллеров AVR Mega с тактовой частотой 16 МГц составит 32 кГц. Этот факт надо учитывать — такую частоту вполне можно использовать для генерации цифрового звука. Грамотный способ извлечения звука с помощью счетчиков-таймеров Arduino довольно сложен (не выходя за пределы высокоуровневого языка среды программирования, его реализовать не получится). Он приведен, например, в статье Андрея Шаройко «Arduino. Генератор сигналов»[1].

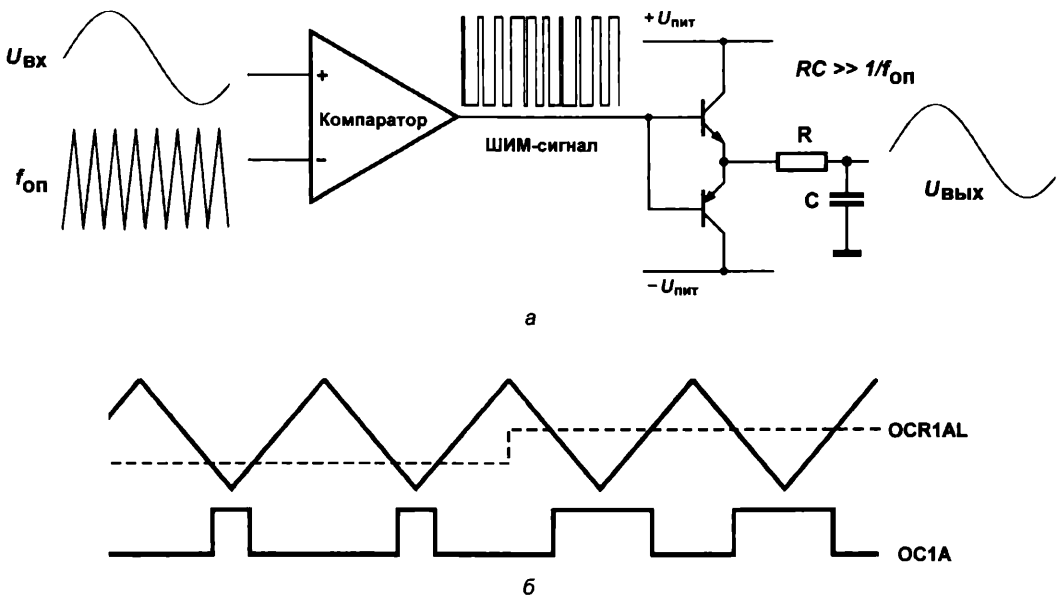


Рис. 22.1. Принцип широтно-импульсной модуляции: *а* — простая модель ШИМ; *б* — реализация ШИМ в цифровом виде в МК AVR

ШИМ и Arduino

ШИМ-режим, доступный в стандартных функциях среды Arduino, гораздо медленнее. Для разных плат Arduino он несколько различается для разных выводов. В самом популярном Arduino Uno, который мы здесь используем, функция работает на выводах 3, 5, 6, 9, 10 и 11, причем на всех выводах тактовая частота ШИМ составляет около 490 Гц, за исключением 5 и 6 выводов, где она равна примерно 980 Гц.

[1] См. **https://sites.google.com/site/vanyambauseslinux/arduino/arduino-generator-signalov**.

Последние предпочтительно не использовать, т. к. сигнал ШИМ на них может искажаться из-за того, что Timer 0 задействован также в функциях `millis()` и `delay()`.

Преимущество высокоуровневого языка Arduino здесь встает в полный рост. Для вывода ШИМ-последовательности на нужном выводе достаточно написать всего одну строку в тексте скетча: `analogWrite (pin, value)`, где `pin` — номер вывода (3, 5, 6, 9, 10 или 11), а `value` — значение в диапазоне от 0 до 255. И все! Для вызова `analogWrite()` нет даже необходимости устанавливать направление выбранного вывода на выход — все произойдет автоматически. Обратите внимание, что, как подчеркивается в официальном руководстве, «функция `analogWrite` никак не связана с аналоговыми входами и с функцией `analogRead`».

Можете попробовать поэкспериментировать с ШИМ-режимом, подключив к выбранному выводу светодиод и меняя его яркость подачей разных значений `value`. Можно к другому выводу из числа аналоговых входов (т. е. выводов A0–A5) подключить потенциометр и считывая напряжение на нем функцией `analogRead()`, непосредственно подавать полученную величину на ШИМ-выход, таким образом превратив Arduino в простейший регулятор яркости свечения. И не думайте, что такое его применение будет избыточным (целый мини-компьютер для того, что можно выполнить одним простым переменником?) — вместо маломощного светодиода можно подключить устройство-драйвер для управления мощными светодиодными лампами, и таким образом решить не самую простую задачу эффективного регулирования яркости светодиодного освещения (см., например, в Интернете схемы на популярной микросхеме-драйвере HV9910/ HV9911).

А мы сейчас займемся другим вопросом: попробуем с помощью ШИМ регулировать обороты двигателя, что является одной из ключевых задач, например, в робототехнике. Сразу заметим, что двигатели бывают очень разные (и управлять ими тоже следует, соответственно, по-разному), и для начала мы ограничимся простейшим вариантом маломощного коллекторного двигателя постоянного тока.

Для подключения двигателя выход контроллера придется умощнить. Обычно это делают либо специальными драйверами (о которых далее), либо просто мощным полевым транзистором. Так как мы сейчас рассматриваем идею управления, то ограничимся последним методом, как наиболее наглядным. MOSFET-транзисторов для этой цели фирма International Rectifier — основной производитель силовых полевиков в мире (ныне объединилась с компанией Infineon) — наплодила неисчислимое количество, потому разобраться в них без опыта непросто. Для наших целей годится в принципе любой мощный *n*-канальный MOSFET-транзистор, у которого пороговое напряжение затвора (Gate Threshold Voltage) составляет 2–4 вольта, не более. Все такие транзисторы обычно рассчитаны на токи в десятки ампер, но это не должно нас смущать: в отличие от биполярных, полевики спокойно будут работать в самом широком диапазоне нагрузок (типовой ток утечки сток-исток для запертого мощного MOSFET-транзистора составляет десятки микроампер, в отличие от миллиампер для биполярных сравнимой мощности). Я выбрал старинный IRFZ44, он может быть заменен, например, на IRFZ46/48, IRF530 и еще много других типов — хотя, как вы увидите, это не оптимальный выбор для данного кон-

кретного случая, но для этого класса транзисторов есть множество рекомендованных применений, потому его стоит рассмотреть.

Простейшая схема ШИМ-управления скоростью вращения двигателя постоянного тока приведена на рис. 22.2, *а*. Двигатель здесь любой коллекторный с мощностью до 1–2 Вт (например, серии R140 на напряжение 5–6 В из детского конструктора). Такой двигатель в установившемся режиме потребляет от нескольких десятков до сотен миллиампер тока. Обратите внимание, что двигатель подключен к напряжению Vin, т. е. ко входному напряжению стабилизатора платы Arduino — не следует перегружать этот стабилизатор стартовым током двигателя, который может быть в несколько раз выше номинального.

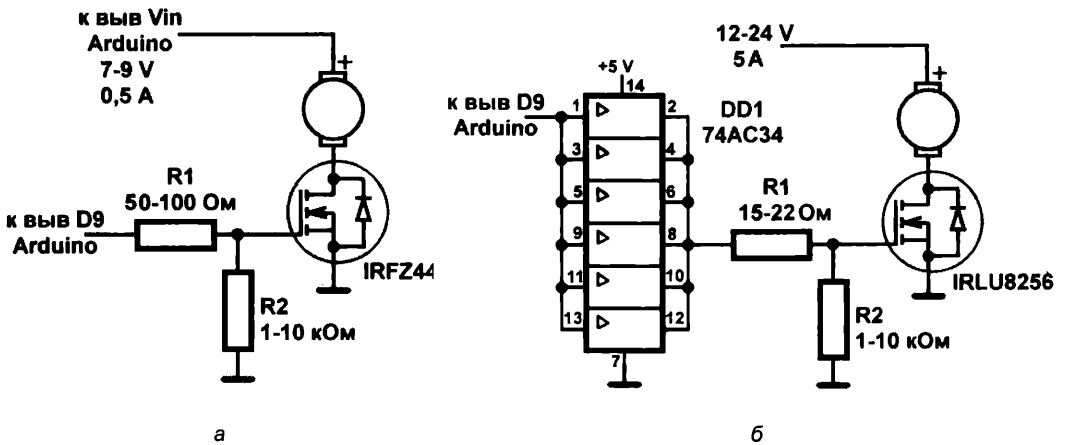


Рис. 22.2. *а* — простейшая схема ШИМ-управления скоростью вращения двигателя постоянного тока; *б* — вариант ШИМ-управления мощной нагрузкой

Сначала отключите двигатель, подключите к Arduino USB-кабель и закачайте в контроллер следующую программу:

```
int p_gate = 9; //pin 9 управление затвором транзистора
int n_speed = 0; //начальное значение ШИМ =0
int speed_step = 5; //шаг изменения скорости
void setup()
{
 pinMode(p_gate, OUTPUT); //на всякий случай
}
void loop()
{
 //устанавливаем значение ШИМ (скорость вращения)
 analogWrite(p_gate, n_speed);
 //увеличиваем текущее значение скорости вращения
 n_speed = n_speed + speed_step;
 //если значение ШИМ=255 или 0
 if (n_speed == 0 || n_speed == 255)
```

```
 {
 speed_step = -speed_step ; //начинаем снижать/повышать
 }
 delay(50); //пауза 50 миллисекунд
}
```

Отключите плату от USB, подключите двигатель и затем запустите схему от внешнего источника 7–9 В с током не менее 1–2 А. Согласно программе, двигатель должен в течение около 2,5 с набирать обороты до максимума, затем за то же время плавно снижать их до нуля. Цикл будет продолжаться до бесконечности.

Теперь обдумаем, что же мы тут натворили. Во-первых, укажем, что именно так следует запускать любой двигатель во всех случаях — пусть он плавно набирает обороты, а не запускается рывком с места. Одному моему младшему родственнику подарили детский автомобиль, в котором конструкторы забыли про плавный запуск двигателя. В результате родители всерьез озаботились состоянием шейных позвонков ребенка, и автомобиль пришлось до некоторого возраста просто запретить. Кроме того, таким образом гораздо меньше насилуется источник питания: ток потребления, минуя экстремальные пусковые значения, плавно нарастает от нуля до номинального значения (это особенно существенно, если источник питания — литиевые аккумуляторы, которые не любят экстремальных значений тока потребления). Необязательно запускать его так долго — шаг нарастания в 50 мс мы здесь выбрали просто для наглядности, для нормальной работы достаточно значения 15–20 мс, чтобы максимум достигался за время 0,7–1 с.

Во-вторых, ровно по той же причине останавливать двигатель стоит по такому же *алгоритму: не выключая питание мгновенно, а постепенно, в течение примерно* 0,5 с снижая его до минимума (на всякий случай можно предусмотреть и аварийный останов мгновенным выключением питания и, если надо, еще и приложение тормозящего усилия).

По такой примитивной схеме с управлением затвором MOSFET-транзистора напрямую от выхода микроконтроллера можно управлять только маломощными двигателями с током потребления не более 0,5 ампера при напряжениях питания двигателя до 10–12 В. Но прежде, чем мы разберемся с вопросами управления MOSFET-транзисторами, стоит понять, как же грамотно их выбирать под конкретную задачу.

Подбор MOSFET-ключей и драйверов для мощной нагрузки

Мы здесь ориентировались на традиционный тип MOSFET-транзисторов с большим запасом по мощности и напряжению, просто, чтобы понять, как они работают и что с ними делать в настоящих задачах. Мощные полевые транзисторы, как мы выяснили в *главе 6*, — довольно капризная штука. Напряжение на затворе, при котором транзистор IRFZ44 и аналогичные полностью открываются и сопротивление канала становится равным сотым долям ома — величина весьма условная, в реаль-

ности она зависит от тока стока и напряжения на нем. Как правильно оценить работу ключа в условиях нашей задачи?

На рис. 22.3, *слева* показана зависимость тока стока (*Ic*) от напряжения затвор-исток (*Uзи*) для IRFZ44, взятая из фирменной документации на этот транзистор. Пусть ток в нагрузке при открытом ключе должен быть равен 5 А, напряжение питания 12 В. Проведем на графике прямую через две точки, соответствующие полностью открытому ключу (ток стока *Ic* = 5 А, напряжение сток-исток близко к нулю) и полностью закрытому (ток *Ic* близок к нулю, напряжение сток-исток равно напряжению питания *Vcc* = 12 В) — серая линия на графике. Найдем точку пересечения этой прямой с характеристикой, соответствующей напряжению на затворе 5 В (жирная серая точка). Спроецировав ее на оси координат, мы найдем, что ток при этом реально будет равным примерно 4,5 А, а напряжение на стоке около 0,22 В. Проверим: сопротивление ключа равно примерно 49 миллиом, что лишь ненамного больше указанного для этого типа наилучшего значения в 28 миллиом, так что все правильно.

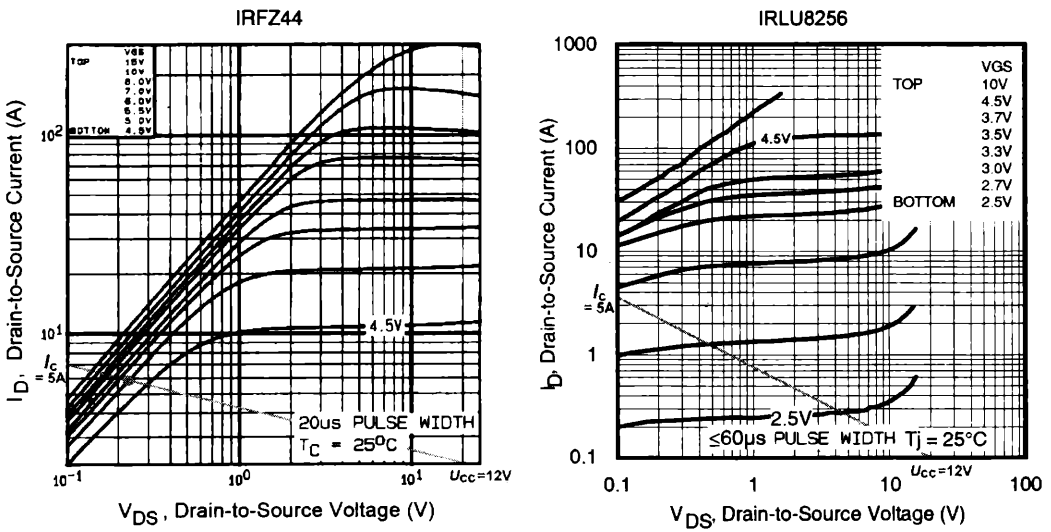


Рис. 22.3. Выходные характеристики IRFZ44 и IRLU8256
(I_D — ток стока, V_{DS} — напряжение сток-исток)

Чтобы получить в нагрузке реальные 5 А, придется подать на затвор напряжение, соответствующее одной из кривых, находящихся выше нашей прямой — для данного типа это значения от 10 вольт и выше. То есть для управления от контроллера с пятивольтовым питанием IRFZ44 безоговорочно годится лишь при малых токах стока (менее 0,2 А, см. график). Больше 12 ампер при таком напряжении на затворе этот транзистор вообще выдать не сможет (см. горизонтальный участок кривой при напряжении 5 В). При снижении управляющего напряжения до 4,5 вольт, что реально происходит в начале процесса открывания, пока заряжается емкость затвор-канал, ток упадет еще больше. При увеличении напряжения питания картина

не будет меняться кардинально — хотя график ограничен значениями ниже 12–14 В, в начальном участке прямая при более высоких напряжениях питания пройдет практически так же.

Отметим, что для рекомендуемых во многих проектах транзисторов IRF530 все еще хуже: для них при 4,5–5 вольтах на затворе напряжение на стоке составит не менее 0,3–0,4 вольта. Более подходящий тип транзистора для данной задачи — например, IRLU8256 в корпусе TO-251 (как TO-126 на рис. 6.12, но короче и без отверстия). Он имеет номинальное пороговое напряжение 1,5–2,5 вольта, и, как следует из фирменного описания, при 4,5–5 вольтах на затворе уже запросто пропускает через себя ток в десятки ампер при сопротивлении канала в 7–8 тысячных ома. Если для IRLU8256 провести тот же анализ, что для IRFZ44, то прямая от *Ic* = 5 А до Vcc = 12 В пройдет ниже почти всех кривых для различного напряжения управления (см. рис. 22.3, *справа*). Уже при 3-х вольтах пересечение с нашей прямой окажется левее области графика, т. е. напряжение на стоке окажется менее 0,1 В, а при 4,5 вольтах на затворе оно будет исчезающее мало (минимальное сопротивление канала, согласно документации, равно 5,7 миллиома).

Правда, для большого тока здесь все равно потребуется умощнение выхода контроллера с целью ускорения процесса открывания-закрывания (емкость затвора у IRLU8256 и у IRFZ44 одного порядка — 1470 и 1900 пФ соответственно). Такой вариант схемы ШИМ-управления нагрузкой с током до 5–6 А при напряжениях до 12–24 вольт показан на рис. 22.2, *б*. Здесь для умощнения выхода Arduino используются быстродействующие буферные повторители AC-серии, каждый из которых способен без нарушения уровней напряжения на выходе тянуть до 24 мА (74AC34 можно заменить на 74HC4050, но она вдвое медленнее). Соединив параллельно все шесть штук, входящих в микросхему 74AC34, мы получим на затворе транзистора неискаженные импульсы с фронтами длительностью порядка десятков наносекунд, т. е. на уровне быстродействия самого транзистора IRLU8256. Это значительно снизит потери мощности в этой схеме (также как, кстати, и в преобразователе из *главы 9* — см. рис. 9.19). Резистор R1, ограничивающий ток в цепи затвора, здесь служит для той же цели, что и раньше: уберечь выход микросхемы от перегрузки, только величина его снижена до 15–20 Ом (что примерно соответствует максимальному току через вывод, равному 50 мА). Без него схема сразу не сгорит, конечно, но для надежности его следует установить.

Конечно, можно обойтись и без самодеятельности: для управления двигателями (как и трансформаторами, кстати — см. все тот же импульсный преобразователь-инвертор в *главе 9*) проще использовать готовые драйверы, которые выпускаются специально для управления мощной индуктивной нагрузкой. Для не слишком мощных двигателей с током 1–4 ампера при напряжении питания от 5 до 24 вольт (такие обычно ставят в роботы-модели) подойдет драйвер на основе микросхем L293/298 — такие, в частности, установлены на платах расширения типа Motor Shield (см. ее описание и примеры использования на сайте **Amperka.ru**). Для более мощных двигателей, например для больших электромобилей, имеется также свой ассортимент драйверов, которые несложно разыскать в Интернете.

Заметки на полях

В связи с удешевлением микросхем-драйверов и мощных MOSFET-ключей ШИМ-управление мощностью в различных устройствах вошло, как говорится, в моду. В связи с этим следует предостеречь читателя от попыток замены традиционных редукторов на ШИМ-регуляторы — они абсолютно не взаимозаменяемы. Вспомните коробку передач автомобиля: переключение на низкие/высокие обороты с помощью редуктора преследует цель не только и не столько снижения/увеличения собственно оборотов, сколько изменения крутящего момента, т. е. усилия, подводимого к колесам. Если вы помните школьную физику, то знаете, что то же самое правило рычага относится ко всем редукторам: чем ниже обороты, тем выше передаваемое усилие. А ШИМ-регулирование скорости вращения делает ровно обратную операцию: при снижении скорости снижается и мощность, следовательно, крутящий момент падает. Как приспособление для плавного запуска, например, электродрели — при постепенном нажатии клавиши включения — ШИМ-регулятор как раз очень удобная штука, но для точной установки числа оборотов в подобных механизмах он далеко не лучший вариант. Есть типы двигателей (например, шаговые), скорость вращения которых зависит от частоты, что позволяет менять скорость в широких пределах, но при этом сохраняя постоянной и мощность на валу привода, однако ШИМ-управление к этому не имеет отношения.

В заключение было бы неправильно не упомянуть, что в «настоящих» схемах регулирования мощности двигателей часто стоит не один транзисторный ключ, а два или даже четыре. Это зависит как от конструкции драйвера, так и от особенностей управления различных типов двигателей. Кроме того, микросхемы-драйверы обычно имеют не один канал управления двигателем, а также два или иногда четыре — для одновременного управления, например, всеми колесами тележки-робота.

Запись на SD-карту

Возможность сохранения произвольных данных на SD-карте — пример операций на Arduino, которые на ассемблере или низкоуровневом C, без привлечения готовых библиотек, повторить практически невозможно (другой такой пример дает обработка кодированных данных приемника 433 МГц, описанная в предыдущей главе).

Первая проблема заключается в подключении SD-карты к схеме. Конструкция SD-карт предполагает упаковку их в корпуса различных габаритов, потому приходится подбирать стандартные «шилды» именно под тот корпус, который имеется, или запасаться переходниками. Решение это довольно громоздкое, потому, если вы не найдете готовой малогабаритной платы, можно сделать самодельный переходник по схеме, показанной на рис. 22.4. Здесь контакт «выбор кристалла» (CS) подключен к выводу 4 платы Arduino (в расчете на стандартную библиотеку SD, которая входит в пакет Arduino), но, разумеется, можно задействовать и любой другой, который позволяет выбранная библиотека.

Для беспрепятственной вставки и извлечения карты, вообще говоря, необходимо специальное гнездо — если вы посмотрите на изображения карт разных типов на рис. 22.4, *справа*, то увидите, что у многих контакты размещены тем или иным нестандартным способом (и частично с нестандартным шагом — у карт обычного размера, например, шаг 2,5, а не обычные 2,54 мм). Если специального гнезда под рукой нет, то под SD- или MMC-карты стандартного (большого) размера можно по-

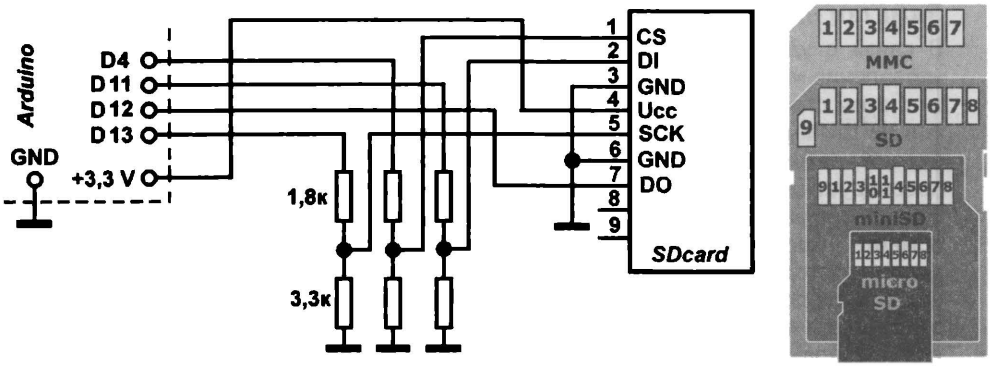


Рис. 22.4. Схема переходника для SD-карты и нумерация выводов для карт разных типов

пробовать приспособить разъемы-«слоты» типа CE, как для компьютерных ISA-карт. Только у них придется немного подпилить перегородки между контактами с «нерабочей» стороны — стандартная толщина карты 2,1 мм, а в CE-разъеме расстояние между перегородками 1,9 мм. Остальные разновидности карт (самый ходовой — micro-SD) можно к тому же разъему подключать через адаптер.

Для отладочных целей можно такой разъем сделать из двухрядного PLD с шагом 2,54 мм, если аккуратно подогнуть его выводы друг к другу (см. рис. 22.5). Контакт 8, который, как можно видеть на рис. 22.4, размещен к соседнему ближе остальных не понадобится.

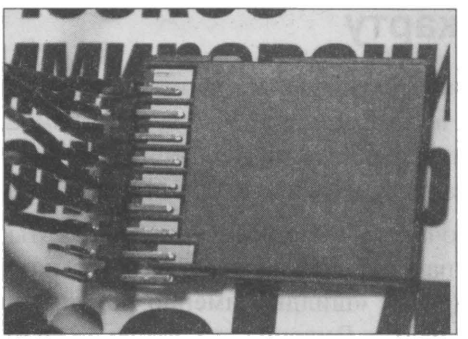


Рис. 22.5. Самодельный отладочный разъем для SD-карты

Отдельный вопрос — о питании карт. Они требуют 3,3 вольта (от 2,7 до 3,3), и на платах Arduino Uno и Mini есть отдельный стабилизатор именно для таких целей (на популярной Nano его нет, учтите). В описаниях этих плат пишут, что максимальный ток по выходу 3,3 В равен 50 мА, чего для SD-карт, вообще говоря, недостаточно. Эксперимент, однако, показывает, что карты от этого питания нормально работают. И действительно, если верить принципиальной схеме с официального сайта Arduino, то в этих платах стоит стабилизатор LP2985-3.3, который, согласно его документации, тянет до 150 мА. Если не хотите заморачиваться этим вопросом, берите клон Arduino с более мощным выходом 3,3 вольта или ставьте отдельный стабилизатор специально для питания карты.

Сигнальные выходы платы Arduino, работающей от напряжения 5 В, придется развязать от входов карты делителями, как показано на рис. 22.4. Для них годятся любые пары резисторы в пределах 1–10 килоом, в соотношении верхнего плеча к нижнему примерно 1:2.

Тестовые программы для записи на SD-карту можно найти в папке examples библиотеки SD. Действия при записи на карту, в общем, похожи на создание и запись файла в программировании под Windows. Учтите, что библиотека не отрабатывает все возможные ошибочные ситуации. Если вы, например, вытащите карту и затем подключите ее обратно (при желании считать накопленные данные), то карта станет недоступной и чисто программными методами из этого вывернутся не получится — только перезагрузкой контроллера. Кроме того, во избежание неприятностей с записанными файлами при случайном выключении питания или извлечении карты, не следует держать файл открытым. При периодической записи необходимо файл каждый раз закрывать по окончании очередного сеанса записи и в начале следующего открывать заново:

```
dataFile=SD.open("data.txt", FILE_WRITE); //создать файл или открыть
                                          //имеющийся для записи
. . . . .
<запись в файл>
    . . .
    dataFile.close();     //закрыть файл
```

Подключение тастатуры 3×4

Тастатурой называют маленькую цифровую клавиатуру. На английский любая ее разновидность переводится словом keypad, буквально «вспомогательная клавиатура». Однако в русскоязычной технической литературе по традиции тастатурой называют только телефонный кнопочный номеронабиратель — тот, что пришел на смену дисковому на телефонном аппарате. Тастатуру удобно использовать для ввода пин-кодов, паролей и прочих цифровых данных. Мы для определенности будем применять тастатуру типа АК-304 (в «Чипе-Дипе» она продается под названием TF-0286), но можно использовать любую другую на 12 или 16 кнопок, принцип подключения у них один и тот же.

Пример подключения подобной тастатуры показан на рис. 22.6. В качестве контроллера подойдет любая из стандартных плат Arduino. Аналоговые порты A0–A5 в данном случае используются не по назначению, а просто как цифровые входы и выходы (в скетчах обозначения A0–A5 лучше заменить на показанные в скобках номера 14–19, чтобы не путаться).

Даже и не думайте опрашивать клавиатуру в основном цикле программы, как это предлагается во многих примерах[1]! Для одной кнопки это еще худо-бедно приемлемый метод, но правильно выбрать из 12 кнопок одну нажатую, да еще и в про-

[1] См., например, тут: **http://wiki.amperka.ru/конспект-arduino:кнопочный-переключатель**.

грамме, созданной на языке сверхвысокого уровня, просто невозможно — она потеряется в собственном дребезге и у вас будут сплошные пропуски, перемешанные с многократными ложными срабатываниями. Мы в предыдущих главах неоднократно обращались к проблеме дребезга кнопок, и не нужно думать, что тастатура — исключение. Да, чаще всего тастатуры выпускаются на основе резиновых клавиш, где дребезга почти не наблюдается, но кто будет гадать, на какой основе создан именно ваш экземпляр? Как мы говорили в *главе 20*, есть довольно много случаев, когда без разницы, сколько раз сработает нажатая кнопка, однако здесь мы с первого импульса должны четко и без ошибок зафиксировать введенный код.

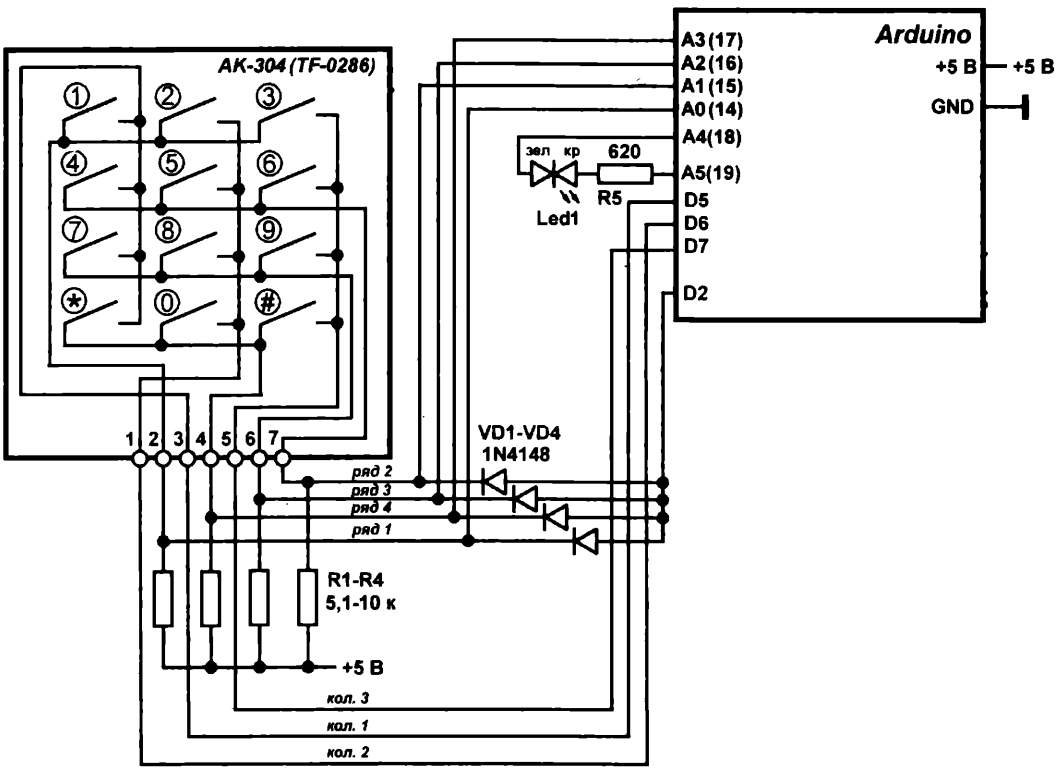


Рис. 22.6. Подключение тастатуры к Arduino

Здесь мы применим метод, отличный от использовавшегося в *главе 20*, который несколько напоминает работу кнопки совместно с одновибратором (см. *разд. «Одновибраторы» главы 16*) и состоит в следующем. Кнопка заводится на вывод внешнего прерывания (INT0 или INT1), который настраивается на срабатывание по определенному перепаду (из 1 в 0 или наоборот, смотря, как подключена кнопка). При возникновении прерывания в его обработчике первой же строкой мы запрещаем его дальнейшее возникновение (чтобы прерывание не сработало вновь сразу по выходу из обработчика). После этого в нем анализируется состояние нажатой кнопки и производятся остальные необходимые действия. По выходу из обработчика вклю-

чается любой из таймеров на время примерно 0,2–0,5 с, по его окончании вновь разрешается прерывание от кнопки[1].

Перед разрешением внешнего прерывания в случае подключения такого ненадежного устройства, как кнопка, обязательно необходимо сбросить регистр флагов внешних прерываний (в ATmega328 он носит название EIFR) — потому что даже при запрещенном прерывании произошедшее, но не обработанное событие фиксируется в этом регистре. Оно обработается сразу, как только прерывание будет разрешено, даже если это произошло через час или через сутки после самого события. Регистр флагов сбрасывается записью единиц в нужные разряды, для чего служит команда `EIFR=0xff;` (здесь на всякий случай сбрасываются все биты, хотя к прерыванию INT0 и INT1 относятся только нулевой и первый бит). Отсутствием такой команды перед вызовом функции `attachInterrupt()` грешат даже авторы руководств по программированию на официальных сайтах Arduino, но не будем следовать их примеру.

Эта простая последовательность действий может обрасти кучей нюансов в зависимости от конструкции кнопки, но в случае мало подверженной дребезгу тастатуры она работает нормально. Таймер для задержки у нас уже имеется усилиями разработчиков Arduino, так что оформить придется только прерывание от кнопки. Постойте, но ведь у нас не одна кнопка, а двенадцать — как их все собрать на одно прерывание, да еще и так, чтобы они не мешали друг другу? Этому служит конструкция из резисторов и диодов (см. рис. 22.6). Порты 5, 6 и 7, связанные с колонками (столбцами) в матрице кнопок, здесь подключены на выход, а порты 14–17 (выводы A0–A3 Arduino), связанные с рядами (строками) — на вход.

Работает эта конструкция следующим образом. В начальном состоянии у нас на выходе портов 5, 6 и 7 логический ноль, а на входах 14–17 (A0–A3) — логические единицы за счет резисторов R1–R4, диоды VD1–VD4 заперты, на входе 2 (связанном с прерыванием INT0) логическая единица за счет включенного подтягивающего резистора. Если нажата какая-то кнопка, то она закорачивает соответствующий резистор на низкий уровень с выхода порта 5, 6 или 7, на этой строке возникает низкий уровень, соответствующий диод открывается, а на входе 2 возникает логический ноль.

Перепад из единицы в ноль на выводе 2 вызывает прерывание INT0. В нем мы первым делом его самого запрещаем (функция `detachInterrupt(0)`), затем устанавливаем на всех портах 5–7 логическую единицу и, устанавливая по очереди эти порты в ноль, опрашиваем выводы A0–A3 (14–17). Тот, на котором окажется логический ноль — и определяет нажатую кнопку. Если таковая нашлась, то некая служебная переменная `flag` устанавливается в 1, чтобы по выходу остальная программа знала, что сканирование закончено. На этом цикл обработки не заканчивается: закончив сканирование кнопок, мы опять устанавливаем порты 5–7 в низкий уровень для следующего нажатия.

---

[1] Отметим, что специальная библиотека Bounce (см. **http://greenoakst.blogspot.ru/2012/06/arduino-bounce.html**) дает еще более совершенный метод отслеживания кнопки, основанный на подсчете времени, но мы здесь обойдемся простейшим.

В основном цикле программы действуем так: если переменная `flag` в некий момент оказалась отличной от нуля, то делаем паузу в полсекунды (обычной функцией `delay()`), после чего не забываем сбросить регистр флагов прерываний EIFR и опять разрешаем прерывание INT0.

Эта общая схема реализована в следующем алгоритме (использование ключевого слова `volatile` обусловлено тем, что здесь глобальные переменные используются и в теле программы, и в обработчике прерывания, см. *главу 20*):

```
volatile int PinOut[3] {5, 6, 7}; //выходы
volatile int PinIn[4] {14, 15, 16, 17}; //входы
volatile unsigned char flag=0; //вспомогательная переменная - флаг
volatile char Nbutt; //символ кнопки из массива
const byte ledPin = 19; //светодиод нажатия кнопки
const char value[4][3]
{ {'1', '2', '3'},
 {'4', '5', '6'},
 {'7', '8', '9'},
 {'*', '0', '#'}
};
void setup() {
for (int i = 1; i <= 3; i++)
{
 pinMode (PinOut[i - 1], OUTPUT); //на выход
 digitalWrite(PinOut[i - 1], LOW); //с нулевым уровнем
}//end set output

for (int i = 1; i <= 4; i++)
{
 pinMode (PinIn[i - 1], INPUT); //на вход
 digitalWrite(PinIn[i - 1], HIGH);//подтягивающий резистор
}//end set input
 pinMode (ledPin, OUTPUT);
 pinMode (18, OUTPUT)//это если Led1 двухцветный, см. схему
 digitalWrite(18, HIGH); //включаем зеленый
 EIFR=0xff; //очищаем регистр прерываний
 attachInterrupt(0, matrix, FALLING); //вкл. прерыв. int 0 1->0
 Serial.begin(9600); //открываем Serial-порт
}//end setup

void matrix (){//обработка прерывания
 detachInterrupt(0); //запрещаем прерывание int 0
 digitalWrite(18, LOW); //выключаем зеленый
 digitalWrite(ledPin, HIGH); //включаем красный
 for (int i = 1; i <= 3; i++) //все столбцы в 1
 {
 digitalWrite(PinOut[i - 1], HIGH); //
 }
```

```
 for (int i = 1; i <= 3; i++) //0 по столбцам по очереди
 {
 digitalWrite(PinOut[i - 1], LOW); //
 for (int j = 1; j <= 4; j++) //ищем 0 по строкам
 {
 if (digitalRead(PinIn[j - 1]) == LOW) //если вход равен 0
 {
 Nbutt = value[j-1][i-1] ;//то кнопка найдена
 flag = 1;
 }
 }
 digitalWrite(PinOut[i - 1], HIGH); //для сканирования следующего столбца
 }
 for (int i = 1; i <= 3; i++) //закончили, все столбцы в 0
 {
 digitalWrite(PinOut[i - 1], LOW); //
 }
}//end прерывания matrix

void loop() {
if (flag!=0) //если было прерывание
{
 flag = 0; //флаг обнуляем снова
 Serial.println(Nbutt);
 delay(500); //ждем полсекунды
 digitalWrite(ledPin, LOW); //выключаем красный
 digitalWrite(18, HIGH); //включаем зеленый
 EIFR=0xff; //очищаем регистр прерываний
 attachInterrupt(0, matrix, FALLING); //вкл. прерыв. int 0 1->0
}
} // end loop
```

Здесь о текущем состоянии схемы сигнализирует светодиод Led1, подключенный к порту 19 (A5): он загорается в момент срабатывания прерывания кнопки и гаснет по истечении выдержки времени, когда схема готова к обработке следующего нажатия. Кроме того, здесь в последовательный порт для контроля выводится определенный программой символ нажатой кнопки. Двухцветный светодиод Led1 может быть заменен обычным, подключенным к выводу 19 и «земле» (без переделки программы).

Эта программа приводится как демонстрационная к примеру подключения тастатуры и может быть использована для проверки правильности подключения или как часть любого проекта, где подобная клавиатура требуется.

Измерение частоты в Arduino

Измерение частоты внешних импульсов — довольно часто встречающаяся задача. В электронике испокон веков существует два метода ее решения, которые отличаются диапазонами частот, в которых они применяются. Высокие частоты измеряются с помощью формирования промежутка времени («временны́х ворот») точно известной длительности, в течение которых подсчитывается число пришедших внешних импульсов. Если длительность таких «ворот» равна одной секунде, то число импульсов равно частоте сигнала в герцах. Наоборот, при низких частотах «ворота» формируют из периода (или нескольких периодов) входного сигнала и заполняют их импульсами точно известной частоты. Получают длительность периода, и частоту приходится вычислять. Если частота заполнения равна 1 МГц, то длительность периода получается в микросекундах, и для вычисления частоты в герцах нужно 10^6 поделить на полученное значение.

Подчеркнем, что в первом способе разрешение принципиально получается также в один герц — чтобы получить десятые доли герца, надо увеличивать длительность ворот в десять раз, сотые — в сто раз, а такие времена ожидания результата и неудобное, и некрасивое решение. На самом деле таким способом измеряют частоты от сотен герц и выше, когда разрешение в один герц вполне приемлемая величина. А для частот ниже одного герца, а также в единицы и десятки герц применяют второй способ. Он представлял некоторые трудности в реализации на основе отдельных микросхем-счетчиков (из-за сложностей с проведением операции деления многозначных чисел), но в микроконтроллерах, да еще и программируемых на языке высокого уровня, это проблемы не представляет.

Есть много способов для реализации этих алгоритмов на основе Arduino, в том числе очень точных, исключающих вероятность выпадения отдельных входных импульсов за счет относительной медленности контроллера. Есть даже специальные библиотеки FreqMeasure и FreqPeriod, чересчур переусложненные, с практическим отсутствием документации и неработающими примерами, потому отметаем их сходу. Мы здесь рассмотрим реализацию обоих методов одним из самых простых способов — на основе внешнего прерывания. Программы, которые мы сейчас создадим, имеют вполне приемлемый уровень погрешности, и не сложнее тех, которые можно было бы соорудить более примитивными способами. Заметим, что любой метод придется резко усложнять, если речь будет идти об импульсах, получаемых с механических контактов — там придется внедрять защиту от дребезга (см. *разд. «Правильное подключение кнопки» главы 20*). Здесь мы такого «наворота» не рассматриваем, предполагая, что входные импульсы идут от электронного датчика, в котором дребезг отсутствует.

Метод первый — измерение частоты

Схему рисовать не будем ввиду ее тривиальности: источник частотных импульсов подключается ко входу D2 Arduino, т. е. тому, на котором действует внешнее прерывание INT0. При необходимости (если выход источника с открытым коллекто-

ром, как, например, у процессорного вентилятора) туда же можно подключить внешний подтягивающий резистор, что, как мы говорили, надежнее встроенного.

Библиотек никаких тут не потребуется, потому в начале только объявляем переменную-счетчик `count`. Тип ее следует прикинуть, исходя из диапазона внешней частоты при представлении в герцах. Предположим, что нам хватит диапазона `unsigned int` (65 тыс. импульсов), но не забудем, что изменяться этот счетчик будет в прерывании, потому к нему нужно обязательно приставить ключевое слово `volatile` (см. *главу 20*):

```
volatile unsigned int count=0;
```

Далее в установках включаем последовательный порт для вывода данных и подключаем прерывание INT0:

```
void setup()
{
   Serial.begin(9600);
   attachInterrupt(0, frq, RISING); //срабатывание по фронту
}
```

Обработчик прерывания `frq` делает единственное действие — увеличивает на единицу число `count`:

```
void frq(){
count++;
}
```

Основной цикл также будет крайне простым:

```
void loop() {
count = 0; //сбрасываем счетчик
delay(1000); //и ждем 1 секунду
Serial.print(count); //выводим результат
Serial.println(" Гц");
}
```

Поигравшись с длительностью задержки и емкостью счетчика `count`, можно получить выходную величину в самых разных пределах. Верхний предел измеряемой частоты зависит от скорости обработки прерывания, и подсчитать его тут сложно — вероятно, он будет не менее 1–1,5 МГц. Для такого интервала в принципе нужно трехбайтовое число (т. е. тип `long` для `count`), но его можно сократить, если измерять число импульсов не за секунду, а за миллисекунду — в этом случае частота получится в килогерцах.

Метод второй — измерение периода

В Arduino есть функция `pulseIn()`, которая измеряет длительность импульса, т. е. время от фронта до спада сигнала на входе. Для того чтобы приспособить эту функцию к измерению периода (от фронта до фронта или от спада до спада), можно ее немного модернизировать и обозвать новым именем. Но эксперимент пока-

зал, что таким способом мы практически ничего не выигрываем в сравнении с простым и очевидным методом, аналогичным приведенному ранее способу измерения частоты. Мы точно так же объявляем внешнее прерывание INT0 по определенному перепаду, только теперь не подсчитываем число таких прерываний за единицу времени, а наоборот, отсчитываем системное время между двумя (или больше) прерываниями.

Время отсчитывается с помощью функции `micros()`, выдающей число микросекунд, и для операций с ними потребуются числа типа `unsigned long`. Это равносильно заполнением «ворот» частотой 1 МГц, как говорилось ранее, и чтобы получить целую частоту в герцах, нужно поделить 1 000 000 на полученное число. Надо принимать во внимание, что разрешение функции `micros()` равно 4 микросекундам, т. е. на выходе всегда будет число, кратное четырем. С точки зрения точности измерения нас это совершенно не волнует в области низких частот, для которых мы все это затеваем, т. к. число знаков после запятой у значений, выраженных в герцах, и без того будет избыточным. Но может приводить в недоумение, потому что целый результат измерения периода будет колебаться в пределах ±4 последних знаков.

Тестовый скетч может выглядеть следующим образом:

```
#define IN_PIN 2 //вход обнаружения частоты

volatile unsigned long ttime = 0; //время конца ворот
volatile unsigned long time_old = 0; //время начала ворот
volatile uint8_t flag=0; //флаг занятости

void setup() {
  attachInterrupt(0, impuls, RISING); //Прерывание по фронту INT0
  Serial.begin(9600);
}

void impuls(){
    if(flag!=1) ttime = micros()-time_old; //длительность периода
    time_old = micros(); //запоминаем время окончания периода
}

void loop() {
      flag=1; //чтобы ttime не изменилось в процессе вывода
        Serial.println(ttime);
       if (ttime!=0) {//на случай отсутствия частоты
       float f = 1000000/float(ttime); //Вычисляем частоту в Гц
        Serial.println(f,1);}
      flag=0;
      delay(500);
}
```

Обратите внимание на применение флага занятости. Методически наш способ не симметричен изложенному ранее способу измерения частоты, потому что там от-

счет времени производился остановкой контроллера на время измерения. А здесь отсчет производится каждый период непредсказуемой длительности и независимо от других действий. Поэтому при обработке результатов необходимо приостанавливать измерения, чтобы полученная длительность внезапно не изменила свою величину. Это и осуществляется с помощью флага `flag`. В прерывании при этом все равно фиксируется время конца очередного периода, так что первое после паузы измерение не будет ошибочным.

Заметки на полях

В приведенном методе предполагается, что какая-то частота на входе всегда присутствует. Если импульсов нет и ожидание прерывания затягивается, то переменная `ttime` будет либо равна 0 (если ни одного законченного импульса с начала программы не проскакивало), либо в ней будет содержаться результат последнего корректного измерения (если частота внезапно пропала). Последнюю ситуацию невозможно отличить от случая, когда частота просто не меняется, и факт пропадания частоты требует отдельной отработки. Предоставляю читателю при необходимости самостоятельно доработать этот момент: обычно это делается введением тайм-аута на ожидание прерывания, по истечении которого значение периода обнуляется.

Пироэлектрический датчик

Сам эффект появления зарядов на поверхности некоторых кристаллических материалов при изменении температуры известен очень давно — говорят, о нем знали еще древние греки. Но практическое применение этого эффекта, как и многих похожих (пьезоэффект, фотоэффект, термоэлектричество и др.) стало возможным в широких масштабах только с развитием полупроводниковых технологий. Пироэлектрический датчик движения — одно из замечательных изобретений последних десятилетий развития электроники.

Пироэлектрический датчик (PIR-датчик) реагирует на возмущения теплового поля вокруг него. Типичный источник такого возмущения — человеческое тело. Проходящий мимо человек вызовет срабатывание датчика на расстоянии до 7 метров, а движение руки датчик регистрирует на расстоянии 2–3 метров, так что спрятаться от него очень сложно. Работа датчика не зависит от температуры окружающего воздуха, поэтому их можно устанавливать на улице и в неотапливаемых помещениях. PIR-датчики очень часто применяют для управления освещением подъездов, для управления автоматическими дверями в универсамах и офисах, в охранных системах. Другие их применения — в качестве датчика огня в противопожарных системах и для измерения температуры пламени.

Типовой пироэлектрический датчик изображен на рис. 22.7. В зависимости от конструкции он может иметь несколько регулировок. У края платы размещены один или два подстроечных резистора «под отвертку», с помощью которых регулируется чувствительность и время реакции на возмущение теплового поля. Сбоку может располагаться штырьковый разъем с переставляемой перемычкой, которая имеет два положения: H и L. В данном случае разъем отсутствует, а перемычка дорожкой на плате замкнута в положении H. Буквы означают режим одновибратора: режим H — с перезапуском, т. е. датчик будет все время держать активный (высокий)

уровень на выходе, пока вблизи него перемещаются тепловыделяющие объекты и на вход поступают все новые импульсы. В режиме L датчик выдаст однократный импульс заданной длительности, после чего наступит пауза. Чаще всего употребляют режим H, который удобен, например, для включения света в подъезде до тех пор, пока там двигается человек, именно поэтому многие платы PIR-датчиков выпускаются с постоянной перемычкой, запаянной в положении H.

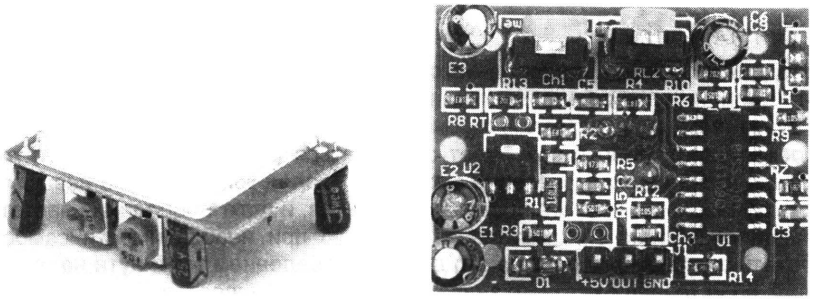


Рис. 22.7. Типовой пироэлектрический датчик

Подробности

Пироэлектрические датчики обладают фантастической чувствительностью — они способны отреагировать на изменение температуры в одну миллионную градуса. Казалось бы, с такой чувствительностью в реальном мире им делать нечего: при любых мельчайших колебаниях температуры основанный на этом эффекте прибор просто зашкалит, и больше ничего не будет происходить. Секрет тут заключается в слове «изменение»: чувствительный кристалл датчика реагирует не на саму величину температуры, а на ее производную, т. е. на достаточно быстрые отклонения от текущего значения, поэтому датчик одинаково хорошо работает и на морозе –40°, и на жаре в +50°. Любое достаточно быстрое изменение температуры окружающей среды ведет к появлению короткого импульса на выходе сенсора, который усиливается и подается на вход одновибратора, выход которого и служит выходом всего датчика. Чувствительность электронной схемы, время работы одновибратора и его режим могут подстраиваться пользователем.

Как уже было сказано, PIR-датчики реагируют на движение. Поэтому, если человек в зоне действия датчика на некоторое время замрет в неподвижности, датчик может не вовремя выключить, например, освещение. Чтобы этого не происходило, и служит регулировка времени срабатывания. Установив ее на достаточно длительный срок, можно добиться пребывания освещения во включенном состоянии, даже если движения какое-то время не происходит.

Кстати, белый полипропиленовый колпачок на PIR-датчике установлен не просто для красоты. Внутри он имеет конфигурацию линзы Френеля, в данном случае не для видимых, а для инфракрасных лучей. Колпачок собирает тепловое излучение в угловой зоне 100–110 градусов, и концентрирует его на установленном внутри датчике.

Давайте соорудим простейший датчик присутствия человека. PIR-датчик имеет всего один вывод данных, на котором при срабатывании возникает уровень логической единицы и никаких библиотек для работы с ним не требуется. Подключите

выход датчика, например, к выводу 2 Arduino, к выводу 13 — пьезоизлучатель звука, к выводу 12 — светодиод.

Программа для проверки PIR-датчика очень простая:

```
#define tonePin 13  //выход для пищалки
#define ledPin 12  //выход для светодиода
#define inputPin 2  //сигнал от датчика
void setup() {
pinMode(ledPin, OUTPUT);  //светодиод на выход
pinMode(inputPin, INPUT_PULLUP);  //вход датчика с подтяжкой
}
void loop(){
if (digitalRead(inputPin) == HIGH) {  //датчик сработал
  tone(tonePin, 1000);  //включаем пищалку
  digitalWrite(ledPin, HIGH);  //включаем светодиод
} else {
  noTone(tonePin); //выключаем пищалку
  digitalWrite(ledPin, LOW); //выключаем светодиод
}
}//конец loop
```

На основе такой программы, практически ничего не меняя, можно сделать много всяких интересных вещей. Это и простейшая охранная сигнализация, и управление освещением и включением различных бытовых приборов, или, например, автоматическое включение видеонаблюдения в присутствии людей перед входной дверью. Для того чтобы управлять с помощью этой схемы различными устройствами, достаточно заменить светодиод на релейный модуль или прямо подключить мощное электронное реле, как это делалось в схеме термостата (см. *главу 20*). А уже контактами реле можно управлять любой нагрузкой: мощной звуковой сиреной, включением прожектора, двигателем привода дверей или штор и т. д.

По адресу: **http://blog.arduinosensors.nl/2016/11/04/arduino-pir-motion-alarm-4x4-keypad** можно найти очень интересный и хорошо продуманный проект охранной сигнализации на основе PIR-датчика. Сигнализация снабжена матричной клавиатурой для ввода пин-кода отключения и двухстрочным текстовым ЖК-дисплеем для вывода сообщений. Красный светодиод «тревога» в ней можно заменить на звуковой излучатель. Автор — по национальности голландец, потому сообщения, выводимые на дисплей, в тексте скетча можно заменить на русские, заменив рекомендуемую автором библиотеку LiquidCrystal на LiquidCrystalRus (см. *разд. «Работа с текстом на графическом дисплее MT-12864J» главы 21*).

Управляем с ИК-пульта от телевизора

Если вы хотите дистанционно включать освещение или управлять тележкой робота, то совершенно незачем изобретать какие-то специальные пульты. Лучше и удобнее, чем фирменные ИК-пульты для управления бытовой техникой, вы все равно не сделаете. А таких пультов в каждом современном доме не один и не два, и в них

уже заложено огромное количество различных команд. Ничто не мешает применить их для своих целей.

Для пробы нам понадобится модуль приемника и какой-нибудь пульт. Можно брать ИК-пульт любой фирмы от любого устройства. Как вы увидите, единственное ограничение будет в том, что нам придется настроить наш приемник под конкретный пульт, но ничто не мешает потом расширить программу на несколько типов пультов, чтобы можно было бы управлять любым из них. Но эту задачу я предлагаю вам попробовать решить самостоятельно, а для начала мы займемся очень простым делом: посмотрим, что именно посылают пульты разных фирм.

Определение и применение кодов команд с ИК-пульта

Любой ИК-пульт посылает в инфракрасном диапазоне сигнал, представляющий собой последовательность битов, следующих со стандартной частотой 38 кГц. На приемном конце, таким образом, последовательность нужно распознать, расшифровать и преобразовать в число — код посылаемой команды.

Для решения такой задачи прежде всего нужен приемник инфракрасных сигналов. Конечно, можно было бы взять любой ИК-фотодиод или фототранзистор и приспособить его в качестве приемника. Но принимать и распознавать коды в таком случае нам придется самостоятельно, а из-за неизбежного наличия шумов и помех это не самое простое дело. Потому проще воспользоваться готовой конструкцией ИК-приемника, благо их выпускается не меньше десятка разновидностей. Причем лучше приобрести не собственно приемник (микросхему в корпусе с окошком и тремя выводами), а готовый модуль с выводами для подключения к Arduino. В таком модуле уже установлены все необходимые дополнительные элементы, и вам не придется лазать по «даташитам» и искать рекомендованные схемы включения.

Нам подойдет модуль ИК-приемника на основе любой из множества микросхем, предназначенных для этой цели. Для примера на рис. 22.8 изображен модуль на основе VS1838b. Как и другие такие модули, он имеет три вывода: два для подключения питания и один сигнальный. Проверьте по описанию, что допустимое питание вашего модуля составляет 5 вольт — обычно они все допускают от 2,7 до 5,5 вольт как минимум, но убедиться в этом все-таки следует.

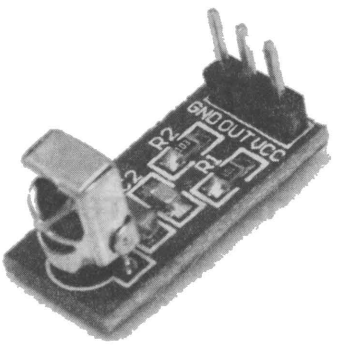


Рис. 22.8. Модуль ИК-приемника на основе микросхемы VS1838b

Для программы расшифровки кодов схему рисовать не потребуется: просто подключите вывод данных модуля (Out) к выводу 11 любой платы Arduino, а также не забудьте подключить к модулю питание +5 В и GND.

Скачайте библиотеку Arduino-IRremote по адресу: **https://github.com/z3t0/Arduino-IRremote**. Следует отметить, что в комплект Arduino последних версий входит библиотека RobotIRremote, которая пересекается с Arduino-IRremote по названиям файлов и вызывает ошибку компиляции. Чтобы они не мешали друг другу, автор библиотеки Arduino-IRremote советует папку RobotIRremote из arduino\libraries перенести в другое место за пределы доступа среды Arduino или попросту удалить.

Далее создайте новый скетч (назовите его IR_code) и впишите туда следующий текст простенькой программы:

```
#include <IRremote.h>
#define RemotePin 11 //прием на 11 вывод
IRrecv irrecv(RemotePin); //вывод приемника
decode_results results;
void setup()
{
  Serial.begin(9600); //Запускаем COM-порт
  irrecv.enableIRIn(); //Запускаем прием
}
void loop() {
  if (irrecv.decode(&results)) //если команда пришла
  {
    Serial.println(results.value, HEX); //выводим полученный код
    irrecv.resume(); //готовность к приему следующей команды
  }
} //конец loop
```

Запаситесь пультами, которые есть у вас дома. Загрузите программу в контроллер и запустите Монитор порта. Направляя пульт на приемник и нажимая различные клавиши, вы увидите на экране их коды в шестнадцатеричной форме. В них придется разбираться — во-первых, при каждом нажатии код, скорее всего, будет повторяться несколько раз, во-вторых, на одну клавишу могут выдаваться два и более разных кодов подряд.

Коды пультов одного производителя будут одинаковые, а вот у разных фирм они могут сильно различаться — никакого стандарта на этот счет не существует, и каждый делает так, как ему удобнее. Самые простые и короткие у фирмы Sony: например, код клавиши увеличения громкости `0x490`, а снижения громкости `0xC90`. При каждом нажатии на соответствующую клавишу такой код выдается от двух до четырех раз.

Другой пример — фирма Pioneer. У нее код при каждом нажатии состоит из двух длинных частей. Клавиша увеличения громкости выдаст два числа подряд: `0xD52A34CB` и `0xF50A4FB0`. Причем при каждом нажатии этот код повторится дважды. Если вы теперь нажмете на клавишу уменьшения громкости, то получите `0xD52A34CB` и `0xF50ACF30`. Как можно заключить из этих последовательностей, пер-

вое число здесь код пульта, он одинаков для всех клавиш. А вот второе число как раз и есть код нажатой клавиши.

Другие фирмы представляют еще большее разнообразие разновидностей кодов, поэтому мы не будем в них разбираться досконально. Кстати, для этого в составе библиотеки Arduino-IRremote есть специальные функции, которые разбирают код на составляющие и сами распознают производителя — это потребуется, если вы захотите создать, например, свой собственный универсальный пульт для управления техникой. Нам этого всего не нужно, потому достаточно знания кодов нескольких определенных клавиш.

Предположим, что мы ориентируемся на фирму Pioneer. Создадим программу, которая с помощью клавиши громкости ее пульта будет включать и выключать имеющийся на плате Arduino светодиод, подключенный к выводу 13. Скетч назовем IR_proba:

```
#include <IRremote.h>
#define ledPin 13 //светодиод
#define RemotePin 11 //вывод приемника
IRrecv irrecv(RemotePin);
decode_results results;
void setup()
{
  irrecv.enableIRIn(); //Запускаем прием
  pinMode(ledPin, OUTPUT);  //светодиод на выход
}
void loop() {
  if (irrecv.decode(&results)) //Если данные пришли
  {
    if(results.value==0xF50A4FB0)//Если нажата кнопка "+"
      digitalWrite(ledPin, HIGH);  //включаем светодиод
    if(results.value==0xF50ACF30)//Если нажата кнопка "-"
      digitalWrite(ledPin, LOW);  //выключаем светодиод
    irrecv.resume(); //Принимаем следующую команду
  }
}//конец loop
```

Эта программа при нажатии клавиши увеличения громкости будет включать светодиод на плате и выключать его при нажатии клавиши уменьшения громкости. Для замены на другой пульт или другие клавиши достаточно определить их коды с помощью скетча IR_Code и заменить коды, имеющиеся в программе. Если же заменить выключение и включение светодиода на приведение в действие исполнительных механизмов, на основе этой простой программы можно создавать довольно навороченные устройства управления различными конструкциями: автоматическими шторами, дверями, замками, сервоприводами видеокамер и робототехники и т. п.

Двухкнопочный плавный регулятор с запоминанием состояния

В этом устройстве мы применим ранее полученные сведения о ШИМ-управлении мощностью (см. начало этой главы) и заодно поучимся обращению с памятью EEPROM, о который мы до сих пор говорили только мельком. В результате мы должны получить устройство плавного управления нагрузкой двумя кнопками «больше»-«меньше». Для дистанционного управления их можно заменить на команды, читаемые с IR-приемника, или вообще применить радиомодули на 433 МГц. Так что здесь мы излагаем только основу — такой простейший кнопочный регулятор, подобрав к нему нужный исполнительный модуль, можно встроить, например, в настольную лампу, люстру или ночник.

Чтобы не запутаться в кнопках, не поленимся и нарисуем принципиальную схему макета такого регулятора (рис. 22.9), с помощью которого будет производиться управление свечением светодиода повышенной яркости. Назначение кнопок Кн1 и Кн2 очевидно из схемы, а вот зачем здесь трехконтактный штыревой разъем «нач. установка», средним выводом подключенный к выводу 2, — пока не ясно. Разъем представляет собой «гребенку» типа PLS, а установленная на нем перемычка в рабочем состоянии подключает вывод 2 к напряжению питания. Рассмотрим сначала его назначение.

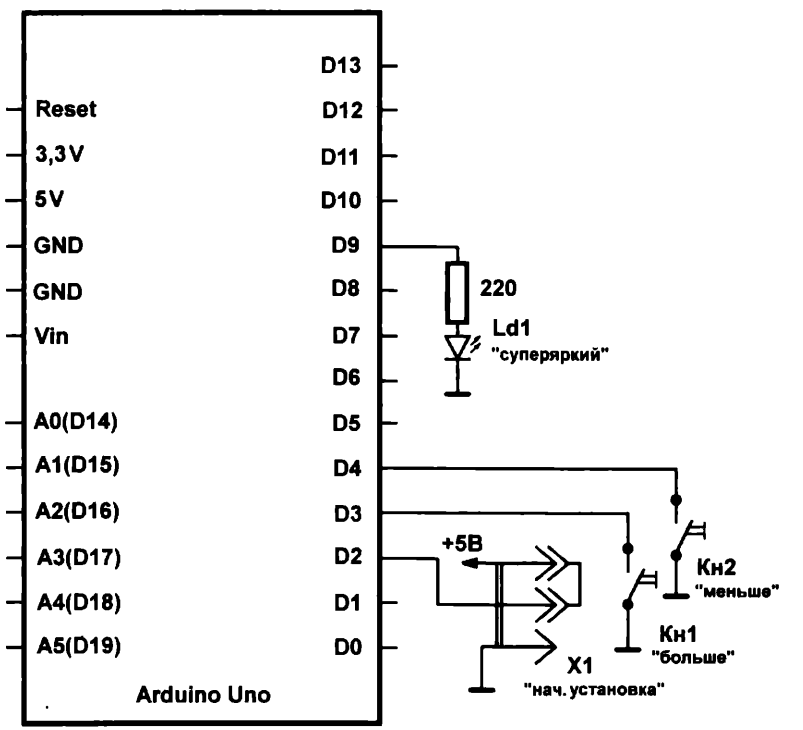


Рис. 22.9. Принципиальная схема плавного двухкнопочного регулятора

В этом примере мы собираемся запоминать установленное значение яркости так, чтобы оно сохранялось даже при выключении питания. При последующем включении оно будет извлекаться из памяти, и нам не придется каждый раз устанавливать подобранную яркость заново. Для примера вспомните, например, управление телевизором, который при включении восстанавливает все ранее установленные параметры: яркость и контрастность экрана, насыщенность цветов, громкость звука, а также настройки каналов — только представьте себе, что было бы, если бы все это пришлось бы устанавливать заново при каждом включении? Именно исходя из подобных соображений, автор в *главе 21* осуждал разработчиков бытовой техники с часами, которые нужно заново устанавливать при каждом включении — для современной электроники это, как минимум, стыдно. Но и простой ночник у постели больного, который при включении вспыхивает на полную и заставляет каждый раз спешно его пригашать — тоже решение из позапрошлого века.

Для реализации функции запоминания состояния здесь надо просто сохранять число в энергонезависимой памяти. Для запоминания каких-то чисел так, чтобы они сохранялись при выключении питания, в микроконтроллерах ATmega и существует EEPROM (более подробное ее описание см. в *главе 18*). Зачем же нам здесь понадобился вывод начальной установки? Дело в том, что сразу при загрузке программы в процедуре `setup()` мы будем читать значение, сохраненное в памяти EEPROM при прошлом включении пульта. А EEPROM — не всегда надежная штука. Во всех официальных руководствах по микроконтроллерам Atmel даже присутствует раздел под названием «Preventing EEPROM Corruption», что означает «предотвращение повреждения EEPROM». И хотя в платах Arduino приняты предлагаемые в этом разделе меры, тем не менее порча хранящихся в EEPROM данных не исключена.

Потому, несмотря на то, что начальное значение, записанное в ячейках памяти EEPROM должно равняться 256 (0xFF), там может оказаться любое число и с самого начала, и в процессе многократных выключений и включений устройства. Чтобы не сбить работу кнопок «больше»-«меньше» в таких случаях, требуется процедура установки определенного начального значения, которую мы и будем осуществлять через вывод `Set_pin`. На практике такая процедура, вероятно, потребуется очень редко (может даже не понадобится вообще), потому на схеме вместо еще одной кнопки к выводу номер 2 подключен контакт штыревого разъема. По умолчанию он подключен перемычкой к напряжению питания, а для установки начального значения перемычку следует удалить и кратковременно замкнуть средний вывод к «земле» той же перемычкой или просто пинцетом. После чего перемычку следует снова установить в прежнее положение: прямое замыкание вывода на напряжение питания дает практически стопроцентную гарантию, что вывод не переключится случайно от каких-либо наводок.

Программа регулятора

В стандартной поставке среды программирования должна быть библиотека для работы с EEPROM, наименование которой записывается заглавными буквами: EEPROM.h. Для уточнения ее наличия загляните в папку Ardino\libraries, где должна

находиться папка EEPROM, или проверьте через пункт меню **Скетч | Подключить библиотеку**. В папке EEPROM во вложенном каталоге с именем examples можно найти примеры использования библиотеки. Будьте внимательны: на некоторых сайтах примеры даны для библиотеки eeprom.h (маленькими буквами), что относится к несколько другому случаю.

Библиотека EEPROM.h обычно имеет только две минимально необходимых функции: чтения `read` и записи `write`. В некоторых современных ее версиях содержится больше разных функций, но здесь они нам не понадобятся. Стоит обратить внимание, что чтение и запись функциями `EEPROM.read` и `EEPROM.write` производится отдельными байтами, потому запись чисел большего размера (типа `int`, `long` или, например, `float`) осуществляется гораздо сложнее — их нужно разобрать на байты, а потом собрать снова (см. аналогичную задачу в случае приемопередатчиков 433 МГц, описанных в *главе 21*).

Создадим новый скетч и в самом начале объявим ссылку на библиотеку EEPROM.h. Далее в секции определений объявим вывод, управляющий яркостью, вывод с перемычкой для начальной установки, две кнопки «больше»-«меньше», переменную для хранения значения яркости и адрес в EEPROM, по которому будет сохраняться ее текущее значение:

```
#include <EEPROM.h>
#define LED_pin 9 //вывод для подключения светодиода
#define Set_pin 2 //вывод начальной установки
#define Button_pin1 3 //вывод подключения кнопки 1 "больше"
#define Button_pin2 4 //вывод подключения кнопки 2 "меньше"

byte bright; //яркость
int addr = 0; //пишем и читаем в EEPROM по адресу 0
```

В секции `setup()` устанавливаем направление работы выводов (на вход с подтяжкой) и читаем из EEPROM предыдущее сохраненное значение яркости:

```
void setup() {
    pinMode(Button_pin1, INPUT_PULLUP); //вывод кнопки 1 на вход
    pinMode(Button_pin2, INPUT_PULLUP); //вывод кнопки 2 на вход
    pinMode(Set_pin, INPUT_PULLUP); //вывод начальной установки
 //для функции analogWrite вывод 9 устанавливать на выход необязательно
 bright = EEPROM.read(addr); //читаем сохраненное ранее значение яркости
 analogWrite(LED_pin, bright); //выводим его в порт LED_pin
}
```

По первому разу после загрузки программы из EEPROM должно прочитаться значение 255 (максимальное значение яркости). Но бывает всякое, для чего мы и выдумывали специальную процедуру установки начального значения. Здесь она выполняется прямо в главном цикле среди прочих процедур. В результате главный цикл состоит из трех условий (для двух кнопок и вывода начальной установки), почти одинаковых по структуре:

```
void loop() {
   if (digitalRead(Set_pin) == LOW) {  //вывод установки на "землю"
       if (EEPROM.read(addr) != 255) {
        EEPROM.write(addr, 255); //начальное значение в EEPROM
        bright = EEPROM.read(addr); //читаем записанное значение
        analogWrite(LED_pin, bright); //устанавливаем LED_pin
       }
       }//конец Set_pin1

   if (digitalRead(Button_pin1) == LOW) {  //кнопка "больше" нажата
    if (bright != 255) { //верхний предел кнопки "больше"
    bright = bright + 15;   //прибавляем шаг = 15
    analogWrite(LED_pin, bright); //устанавливаем значение LED_pin
    EEPROM.write(addr, bright); //запоминаем в EEPROM
    delay(300);  //пауза 0,3 сек
    }
   } //конец Button_pin1

   if (digitalRead(Button_pin2) == LOW) {  //кнопка "меньше" нажата
    if (bright != 0) {//нижний предел кнопки "меньше"
    bright = bright - 15;   //вычитаем шаг=15
    analogWrite(LED_pin, bright); //устанавливаем значение LED_pin
    EEPROM.write(addr, bright); //запоминаем в EEPROM
    delay(300);  //пауза 0,3 сек
   }
   }//конец Button_pin2
}
```

По этой программе контроллер все время проверяет состояние выводов `Set_pin`, `Button_pin1`, `Button_pin2`, и если какой-либо из них замкнут на «землю», производит соответствующие действия. Для кнопок это уменьшение либо увеличение яркости с шагом 15 единиц (с запоминанием по ходу дела текущего значения в EEPROM). Для `Set_pin` сначала производится проверка записанного в EEPROM значения. Если оно не равно максимальному, то устанавливается начальное значение 255 также с его сохранением в EEPROM. Если в процессе нажатия на кнопки установилось крайнее значение (0 для «меньше» и 255 для «больше»), то далее нажатие на ту же кнопку не вызывает никакой реакции.

Подробности

Здесь учитывается факт, что запись в ячейки EEPROM может производиться ограниченное число раз. Хотя это число достаточно велико (руководства фирмы Atmel гарантируют миллион записей), без нужды насиловать EEPROM повторными перезаписями не следует. Потому запись начального значения при замыкании `Set_pin` начинается с проверки уже записанной величины, и затем произведется только один раз, если это необходимо.

Конечно, мы здесь получили только модель настоящего пульта, на которой можно отработать алгоритм. Чтобы получить настоящий пульт, нужно, во-первых, чтобы

он мог управлять нагрузкой помощнее, чем сигнальный светодиод. Для этого к выводу 9 можно подключить затвор мощного MOSFET-транзистора, его исток подключить к «земле», а к стоку — нагрузку, которую следует запитать от внешнего источника соответствующей мощности. Учтите, однако, что так можно управлять лампочками накаливания или «голыми» светодиодами, даже довольно мощными осветительными. А вот готовыми светодиодными или люминесцентными лампами со встроенными схемами управления таким простым способом управлять не получится (см. также на эту тему *разд. «Устройство плавного включения ламп накаливания» главы 10*).

О преимуществах и недостатках Arduino (вместо послесловия)

Как мы видим, проектировать и изготавливать конструкции с помощью Arduino гораздо проще, чем обычным дедовским способом, из отдельных компонентов. Но за эту простоту приходится платить. В некотором смысле ситуация с Arduino напоминает историю персональных компьютеров — как известно, самым первым продуктом компании Microsoft, созданной в 1976 году, была реализация языка Бейсик на компьютере Altair, для которой требовалось аж целых 4 килобайта памяти. Андрес Хейлсберг тоже создавал свою первую версию Pascal на чистом ассемблере, получив файл объемом 31 Кбайт. Современные среды программирования (в том числе и для тех же самых языков) занимают гигабайты, но при этом работают на гигагерцевых компьютерах медленнее, чем первые продукты Гейтса и Хейлсберга на машинах того времени с тактовой частотой и объемом памяти в тысячи раз меньшими. Подобно им и AVR-контроллеры, запрограммированные в среде Arduino, оказываются далеки от своих потенциальных возможностей.

Я не ставлю перед собой задачу как-то принизить значение Arduino и отговорить читателей от работы с этой платформой. Наоборот, я всячески приветствую ее энтузиастов и распространителей. Хочется только, чтобы натолкнувшись в Сети на ее критику, неискушенный читатель не впадал в уныние, а хорошо представлял себе, как говорится, «на каком свете он находится».

Для начала рассмотрим одно принципиальное отличие ассемблерных программ конкретно для AVR-платформы от таких же программ, созданных на языках высокого уровня. Это касается не только Arduino, но и любых других сред программирования, в том числе и профессионального языка C от IAR Systems. Любой текст программы на любом языке высокого уровня включает в себя определения констант и переменных, для которых заранее резервируется место в оперативной памяти. На нюансы этого процесса, разные для разных языков, сейчас не будем обращать внимание — нам важен факт, что в этих случаях программа принципиально ориентирована на работу с памятью. Это происходит потому, что языки высокого уровня создавались в расчете на фоннеймановскую архитектуру, где между памятью данных и памятью программ нет различий — все происходит в одном и том же линейном пространстве общей памяти.

В AVR-контроллерах это не так. Память программ (flash-память) и память данных (оперативная память SRAM) у нее разделены и (с некоторыми исключениями) не могут работать совместно без выполнения специальных операций. Если команда извлекается из памяти программ и выполняется автоматически в течение (в идеале) одного такта, то данные из оперативной памяти предварительно необходимо перенести в рабочие регистры (РОН). В случае, если все предварительные данные размещаются именно в пространстве оперативной памяти, почти каждое действие сопровождается такими операциями переноса, следовательно, ни о каком одном такте на операцию уже говорить не приходится.

В чем же тут отличие от программ на ассемблере? А в том, что в примерах в *главе 19* ни разу не упомянуты ни оперативная память, ни эти самые команды переноса. Все действия в простейших примерах программ в этой главе происходили исключительно в рабочих регистрах. Их там аж 32 штуки — в этом отличие AVR от подавляющего большинства других платформ, где рабочих регистров гораздо меньше. И даже с учетом того, что часть регистров бывает задействована в текущих операциях, десятка-другого оставшихся вполне хватает для размещения всех необходимых переменных в не слишком сложной программе. А если и не хватит, то всегда можно напрячь голову и задействовать в разных процедурах одни и те же регистры, сохраняя их значения в стеке. Это будет все равно менее накладно по времени выполнения, чем каждый раз обращаться к оперативной памяти. В несложных программах резервировать место в SRAM имеет смысл для значений переменных или констант только в том случае, если они имеют большой объем: типичный пример — табличное вычисление какой-либо функции.

В этом стилевом отличии ассемблерных программ от скомпилированных с любого языка высокого уровня причина того, что только ассемблер может позволить использовать все особенности архитектуры полностью, типичные примеры: оптимизация операций с дробными числами путем преобразования их в целые или деление сдвигом вместо обычных последовательных вычитаний, т. е. приемы, которые дают наибольший эффект только при ручной их подгонке «по месту» (см. *главу 19*). Это вообще касается программ на любом ассемблере, но именно для AVR из-за его уникальной регистровой памяти эта особенность выглядит особенно выпукло. Для обычных компьютерных процессоров язык С действительно лучше всех остальных приспособлен к написанию программ системного уровня, а в случае AVR это его преимущество нивелируется.

Теперь непосредственно об Arduino. Простота Arduino во многом обусловлена тем, что практически все действия в программе осуществляются в ее главном цикле. Но такая простота оборачивается недостаточной надежностью работы — «правильно» запрограммированный контроллер должен работать через прерывания. Например, неверно заставлять программу отслеживать нажатие кнопки в главном цикле и убирать дребезг путем простых временны́х задержек, как это делается в распространенном примере для начинающих. Когда контроллер основное время занят последовательным отслеживанием происходящих событий, он запросто может потерять какое-то из них. Так поступали в семидесятые годы, когда контроллеры были намного примитивнее сегодняшних. «Правильные» процедуры работы с кнопками

обсуждались в этой главе и в *главе 20*. Только так эти действия не могут помешать никаким другим процедурам в программе.

В Arduino эксплуатируется факт, что современные микроконтроллеры работают очень быстро, но, по мере усложнения программы, вы довольно скоро упретесь в порог этого быстродействия и не будете понимать, как из этой ситуации вывернуться. На **Habrahabr.ru** один критик платформы писал, что «вы можете всю жизнь формировать задержки с помощью delay-функций и не иметь простейшего представления, как работает таймер на микроконтроллере».

Впечатляют и размеры программ, получающихся после компилирования скетчей в среде Arduino IDE. Программа метеостанции с ЖК-дисплеем займет почти 20 килобайт — около 10 тыс. AVR-команд. Это непредставимо большая величина для таких устройств, и неудивительно, что в Arduino, начав с mega8 с памятью программ 8 килобайт, в конце концов стали ставить контроллер mega328 с памятью аж на 32 килобайта. Для сравнения — самая большая программа, написанная автором на ассемблере, содержала примерно 1200 команд, или около 2,5 Кбайт. При этом программа была весьма навороченная — прием и отправка данных по портам RS-232 и I^2C, обслуживание аналоговых датчиков, обработка многоразрядных чисел, управление многопозиционным индикатором, запись результатов в отдельную энергонезависимую память и т. п.

В то же время Arduino-программа, состоящая всего из двух операторов: `digitalWrite(HIGH)` и `digitalWrite(LOW)`, переменно переключающих внешний вывод без искусственных задержек, при проверке на осциллографе покажет меандр с частотой 70–80 кГц — это в контроллере, работающем на частоте 16 МГц! Простая замена этих функций на непосредственное управление портом, даже без выхода за пределы среды Arduino (по типу, показанному в конце *разд. «Программы для Arduino» главы 20*), ускоряет выполнение операций переключения порта примерно в 50 раз — частота переключения будет почти 4 мегагерца, как и положено (см. [24]).

Хорошей иллюстрацией к расточительности языка служит также пример пустой программы из двух строк, которую употребляют в качестве заглушки, если необходимо общаться с каким-то устройством на последовательном порту, не задевая контроллера (например, при программировании Xbee-модуля — см. *главу 21*):

```
void setup() { }
void loop() { }
```

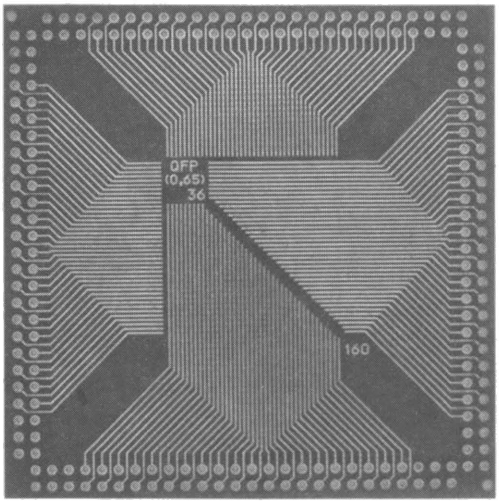
Ее размер после компиляции в последних версиях Arduino IDE составит целых 450 байтов — с помощью ассемблера в такой объем можно запросто втиснуть небольшую программку ориентирования по звездам для орбитального аппарата (реальный случай с одним программистом 60-х годов прошлого века из НАСА, который упаковал такую программу в остававшиеся свободными 256 байтов памяти бортовой ЭВМ спутника).

Заметки на полях

Очень сказывается то, что значительная доля проектов для Arduino в Интернете публикуется людьми, мало понимающими и в электронике, и в программировании. Среди

этих проектов встречаются истинные шедевры, но в общем случае в каждом втором встреченном мной примере содержались грубые схемотехнические или программные ошибки и недоработки, вплоть до полной неработоспособности схемы. А в каждом первом (без преувеличения!) — те или иные методические. Спешу заметить, что это, конечно, не всегда касается официальных примеров, сопровождающих библиотеки или документацию к «шилдам», а также проектов с некоторых особо «продвинутых» сайтов (например, **geektimes.ru** или раздела «вики» на сайте «Амперки»). Но в общем случае к встреченному скетчу необходимо отнестись со всей возможной критичностью: прежде всего, выяснить все непонятные моменты и тщательно разобраться, как он работает.

Как мы видели, нет особых проблем применять к разработке программ для Arduino все возможности МК AVR, включая и прерывания, но при этом среда Arduino теряет свою простоту и идеальную приспособленность к нуждам любителей. Уже из примера с мигалкой в этой главе видно, что делать грамотно — означает ковырять-ся в англоязычных «даташитах», изучать регистры, прерывания и таймеры, вникать в тонкости программирования той или иной процедуры. В конце концов вы быстро придете к выводу, что Arduino IDE вместе с языком Processing только мешают — придется переходить на обычный C или на ассемблер. К этому выводу в конце концов приходят все, кто старается двигаться дальше. Но не унывайте: Arduino дает отличный старт!



Приложения

Приложение 1. Резисторы и конденсаторы

Приложение 2. Стандартные обозначения, размеры и характеристики некоторых гальванических элементов

Приложение 3. Соответствие наименований и функциональности некоторых зарубежных и отечественных цифровых микросхем

Приложение 4. Словарь часто встречающихся аббревиатур и терминов

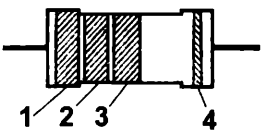
Приложение 5. Единицы измерения и обозначения

Приложение 6. Программирование Arduino Mini и Pro Mini

ПРИЛОЖЕНИЕ 1

Резисторы и конденсаторы

Международная цветная маркировка резисторов (с допуском 5 и 10%)



1 — первая цифра; 2 — вторая цифра; 3 — множитель; 4 — допуск (красный — 2%, золотистый — 5%, серебряный — 10%, отсутствует — 20%)

Цифра 1 и 2	Цвет	Множитель
	Серебряный	0,01 Ом
	Золотистый	0,1 Ом
0	Черный	1 Ом
1	Коричневый	10 Ом
2	Красный	100 Ом
3	Оранжевый	1 кОм
4	Желтый	10 кОм
5	Зеленый	100 кОм
6	Голубой	1 МОм
7	Фиолетовый	10 МОм
8	Серый	
9	Белый	

Пример: красно-желто-оранжево-золотистый — резистор на 24 кОм с допуском 5%

Таблицы номиналов резисторов и конденсаторов

Далее приведены множители для номиналов резисторов и конденсаторов (см. *главу 5*) с допуском 5% (ряд E24) и 10% (ряд E12, выделен жирным). Из этих значений формируются стандартные номиналы резисторов путем их умножения на степень десяти — например: 1,8 Ом, 18 Ом, 180 Ом, 1,8 кОм, 18 кОм, 180 кОм и т. д. Для SMD-резисторов (чип-резисторов) цветной код не применяют, а маркируют тремя цифрами. Первые две цифры обозначают мантиссу (см. таблицу далее), а последняя цифра — показатель степени по основанию 10 для определения номинала в омах (или, что то же самое, число нулей, которое нужно прибавить к первым двум цифрам, чтобы получить сопротивление в омах). Например, маркировка 242 означает, что чип-резистор имеет номинал 24×10^2 Ом = 2,4 кОм.

Емкость керамических конденсаторов также обозначается на их корпусе двузначным числом из той же таблицы, с добавлением третьей цифры. В отличие от резисторов, последняя цифра означает степень десяти, на которую нужно умножить двузначное число, выраженное в пикофарадах (или, что то же самое, число нулей, которое нужно прибавить к первым двум цифрам, чтобы получить емкость в пикофарадах). Так, надпись 474 для конденсатора расшифровывается как $47\cdot10^4\cdot10^{-12}$ = = $0{,}47\cdot10^{-6}$ фарады или 0,47 мкФ. Значение множителя 9 в последней позиции означает, что перед нами емкость в доли пикофарады.

10	16	**27**	43	**68**
11	**18**	30	**47**	75
12	20	**33**	51	**82**
13	**22**	36	**56**	91
15	24	**39**	62	

Резисторы с допуском 1% (ряд E96) имеют следующие множители для номиналов:

100	140	196	274	383	536	750
102	143	200	280	392	549	763
105	147	205	287	402	562	787
107	150	210	294	412	576	806
110	154	215	301	422	590	825
113	158	221	309	432	604	845
115	162	226	316	442	619	866
118	165	232	324	453	634	887
121	169	237	332	464	649	903
124	174	243	340	475	665	931
127	178	249	348	487	681	953
130	182	255	357	499	698	976
133	187	261	365	511	715	
137	191	267	374	523	732	

ПРИЛОЖЕНИЕ 2

Стандартные обозначения, размеры и характеристики некоторых гальванических элементов

В табл. П2.1 представлены стандартные разновидности бытовых одноразовых электрохимических элементов питания. Элементы одного типоразмера взаимозаменяемы, а также могут быть заменены NiMH-аккумуляторами аналогичных типоразмеров (подробности описаны в *главе 9*). Щелочные элементы (с буквой L в обозначении или надписью «alkaline» на корпусе) приблизительно в три раза превышают по энергоемкости «обычные» солевые (последние часто маркируются надписью GENERAL PURPOSE, то есть «общего назначения»). Кроме стандартных щелочных, существуют также элементы Super (Ultra) Alkaline, емкость которых еще приблизительно на треть-четверть выше.

Таблица П2.1. *Стандартные разновидности одноразовых элементов питания*

Напряжение, В	Обозначение (типоразмер)			Размеры, мм		Примерная емкость, мА·ч**
	Торговое (ANSI)	Международное (МЭК)*	Отечественное (ГОСТ)	Диаметр (высота× ширина)	Длина (толщина)	
4, 5	–	3R12, 3LR12	3336	62×67	22	6000
9	1604	6LR6 (щелочная), 6F22 (солевая, литиевая)	«Крона»	17,5×26	48	600 750***
1,5	AAA	R03, LR03, FR03	286	10.5	45	1200
1,5	AA	R6, LR6, FR6	316	14,5	50,5	2500
1,5	C	R14, LR14	343	26,2	50	7500
1,5	D	R20, LR20	373	34,2	61,8	15 000–18 000

* Буква L означает щелочной элемент, F — литий-железо-дисульфидный (Li/FeS_2).

** Для щелочных элементов.

*** Литиевый элемент 1604LC.

Аккумуляторы NiMH тех же типоразмеров имеют близкую емкость (в среднем на 10–20% ниже), за исключением типоразмера «Крона» — емкость аккумулятора в этом исполнении не превышает 200 мА·ч.

Кроме указанных в таблице элементов марганец-цинковой системы (к которым относятся и щелочные, и солевые батарейки), в последние годы появились литий-железо-дисульфидные в типоразмерах AA и AAA со стандартным номинальным напряжением питания 1,5 В (в отличие от обычных литий-марганцевых, имеющих напряжение 3–3,6 В). Они отличаются повышенной емкостью при больших токах разряда, и при токе порядка 0,5 А превышают емкость щелочных примерно вдвое. В маломощных приборах при комнатной температуре выигрыша в емкости литиевые элементы почти не дают. Их высокая стоимость оправдывается долгим сроком хранения (15 лет при комнатной температуре), низким выходным сопротивлением, а также намного большей стойкостью к низким температурам — так, фирма Energizer гарантирует работу своих FR6-элементов до –40 °С. Основная особенность литиевых, отличающая их от других типов, — они держат напряжение «до последнего», после чего оно быстро падает до нуля (см. разрядные кривые на рис. 9.3 в *главе 9*).

Для питания маломощных электронных устройств широко применяются «таблетки»: серебряно-окисные (SR) или литиевые (CR) малогабаритные элементы. Литиевые имеют номинальное напряжение 3 В, «серебряные» — 1,55 В, причем размеры корпусов для тех и других отличаются, литиевые больше диаметром и обычно тоньше («монетки»). Емкость ходовых литиевых элементов (CR2032) достигает 225 мА·ч при токе разряда не более 0,2 мА, у «серебряных» она ниже, и для самых крупных (SR44) не превышает 200 мА·ч. В тех же габаритах, что и «серебряные», изредка можно встретить щелочные «таблетки» (LR) с близким номинальным напряжением (1,5 В), их емкость примерно на 30% ниже, а в остальном они взаимозаменяемы. Система обозначений «таблеток» и «монеток» очень проста: первые две цифры обозначают диаметр, третья и четвертая — толщину (например, 2032 — «монетка» 20 мм в диаметре и толщиной 3,2 мм, 1225 — «таблетка» диаметром 12 мм и толщиной 2,5 мм).

ПРИЛОЖЕНИЕ 3

Соответствие наименований и функциональности некоторых зарубежных и отечественных цифровых микросхем

В табл. П3.1 приведены основные микросхемы «классической» КМОП-серии, их отечественные аналоги, полные или функциональные аналоги из серии 74, включающей в себя как быстродействующие КМОП-микросхемы (с буквой С в наименовании), так и ТТЛ, а также отечественные функциональные ТТЛ-аналоги из серий 155 (133), 555, 533, 1555, 1533 и др. Следует отметить, что всего в серии 74 представлено около полутора десятков различных технологий, но одинаковые по функциональности микросхемы в различных сериях называются одинаково. Не все микросхемы имеются во всех исполнениях — так, полные аналоги 4049/4050 имеются только в серии НС, в то же время дешифратор 74*xx*247(ИД18) существует только в ТТЛ-исполнении. CD4056 (561ИД5) не является его полным аналогом, так как имеет дополнительные выводы для управления ЖКИ или люминесцентного индикатора переменным напряжением и т. д.

Кроме приведенных в таблице, и 4000-я серия, и, тем более, разновидности 74-й, включают в себя еще очень много типов микросхем, здесь приведены только те, которые имеют аналоги в отечественной «классической» 561-й серии (за исключением ряда микросхем, выполняющих арифметические функции, ввиду потери ими актуальности). Часто приводимая в справочниках серия 54 есть та же 74-я, но в военном исполнении, с повышенной надежностью и расширенным температурным диапазоном. О соответствии отечественных и импортных КМОП-серий говорилось в *главе 15*.

Прочерк в таблице для микросхем ТТЛ и 74-й серии означает, что для данной микросхемы «классической» КМОП-серии прямых функциональных ТТЛ-аналогов не существует. Однако это вовсе не означает, что микросхем, выполняющих аналогичные функции, в ТТЛ и быстродействующей КМОП вообще нет. Так, в этих сериях нет микросхемы, аналогичной CD4000 (два элемента «3ИЛИ-НЕ» и один элемент «НЕ»), но есть микросхема ЛЕ4 (74*xx*27), содержащая три элемента

«3ИЛИ-НЕ», которые могут выполнять те же функции. Для некоторых типов, как для упомянутой CD4056, функциональный аналог приведен неполный: ПУ4 и 74HC4050 имеют обычный симметричный выход, а ЛП4 и 74*xx*17 — с открытым коллектором, еще сложнее ситуация с некоторыми счетчиками. Поэтому при замене «классических» КМОП-микросхем на их быстродействующие аналоги и, тем более, на ТТЛ-микросхемы всегда следует сверяться с техническими описаниями.

CD4*xxx* — наименование серии 4000, принятое в фирме Fairchild Semiconductor, MC14*xxx* — в фирме Motorola. У других фирм могут быть свои префиксы, например, SN4*xxx* — у Texas Instruments, HCC или HCF — у фирмы ST Microelectronics, HEF — у Philips и NXP. Микросхемы серии 74HC Texas Instruments выпускает также с префиксом SN, Fairchild — с префиксом MM, остальные с префиксом M или вовсе без него. Еще разнообразнее правила в отношении 74AC, кроме того, все эти правила действуют не всегда, — так, имеются микросхемы с префиксом CD, но производства не Fairchild, а других фирм (например, Texas Instruments или Philips). Отечественное НПО «Интеграл» выпускает лицензионные микросхемы 74-й серии с префиксом IN. Сами обозначения микросхем в серии остаются одинаковыми независимо от префикса.

Таблица П3.1. *КМОП-микросхемы, их аналоги и функции*

4000	КМОП (561)	74xx	ТТЛ	Функции (цифра обозначает число входов элемента)
CD4000	ЛП4	–	–	Два элемента «3ИЛИ-НЕ» и один элемент «НЕ»
CD4001	ЛЕ5	02	ЛЕ1	Четыре логических элемента «2ИЛИ-НЕ»
CD4002	ЛЕ6	25	ЛЕ3	Два логических элемента «4ИЛИ-НЕ»
CD4008	ИМ1		ИМ3	4-разрядный сумматор
CD4011	ЛА7	00	ЛА3	Четыре логических элемента «2И-НЕ»
CD4012	ЛА8	20	ЛА1	Два логических элемента «4И-НЕ»
CD4013	ТМ2	74	ТМ2	Два D-триггера (динамических)
CD4015	ИР2		ИР46	Два 4-разрядных сдвигающих регистра
CD4017	ИЕ8	–	–	Счетчик-делитель на 10
CD4018	ИЕ19	–	–	5-разрядный счетчик с предустановкой
CD4019	ЛС2	–	–	Четыре логических элемента «И-ИЛИ»
CD4020	ИЕ16	–	–	14-разрядный двоичный счетчик
CD4022	ИЕ9	–	–	Счетчик-делитель на 8
CD4023	ЛА9	10	ЛА4	Три логических элемента «3И-НЕ»
CD4025	ЛЕ10	27	ЛЕ4	Три логических элемента «3ИЛИ-НЕ»
CD4027	ТВ1	72	ТВ1	Два JK-триггера
CD4028	ИД1	42	ИД6, ИД10	Двоично-десятичный дешифратор
CD4029	ИЕ14	168	ИЕ16	4-разрядный двоично-десятичный реверсивный счетчик

Таблица П3.1 *(окончание)*

4000	КМОП (561)	74xx	ТТЛ	Функции (цифра обозначает число входов элемента)
CD4030, CD4070	ЛП2	86	ЛП5	Четыре логических элемента «исключающее ИЛИ»
CD4034	ИР6	198	ИР13	8-разрядный регистр сдвига
CD4035	ИР9	195	ИР12	4-разрядный регистр сдвига
CD4039	РП1	170	РП1	Буферное ЗУ
CD4042	ТМ3	175	ТМ8	Четыре D-триггера («защелки»)
CD4043	ТР2	279	ТР2	Четыре RS-триггера
CD4046	ГГ1	124	ГГ1	Генератор с фазовой автоподстройкой частоты
CD4049	ЛН2	04, 74HC4049	ЛН1	Шесть инверторов
CD4050	ПУ4	17, 74AC34, 74HC4050	ЛП4	Шесть буферных повторителей
CD4051	КП2	152	КП5	Аналоговый 8-канальный мультиплексор (КМОП), 8-канальный мультиплексор (ТТЛ, серия 74)
CD4052	КП1	153	КП2	Два аналоговых 4-канальных мультиплексора (КМОП), два 4-канальных мультиплексора (ТТЛ, серия 74)
CD4056	ИД5	247	ИД18	Дешифратор для управления сегментными индикаторами
CD4059	ИЕ15	–	–	Программируемый счетчик-делитель
CD4061	РУ2		РУ2	ОЗУ 256 бит со схемами управления
CD4066	КТ3**	–	–	Четыре 2-направленных переключателя
CD4093	ТЛ1	132	ТЛ3	Четыре триггера Шмидта «2И-НЕ»
–	1564ТЛ2	14	ТЛ2	Шесть триггеров Шмидта с инверсией***
CD4098	АГ1	123	АГ3	Два мультивибратора/одновибратора
CD40107	ЛА10	22	ЛА7	Два элемента «2И-НЕ» с открытым истоком (коллектором)
CD40109	ПУ6	–	–	Четыре преобразователя уровня ТТЛ-КМОП с индивидуальным стробированием
MC14502	ЛН1	365	ЛП10	Шесть стробируемых инверторов
MC14516	ИЕ11	169	ИЕ17	4-разрядный двоичный реверсивный счетчик
MC14520	ИЕ10	393	ИЕ19	Два 4-разрядных двоичных счетчика

* В быстродействующей КМОП-версии не существует. Рекомендуется заменять на 514ИД1/2 (MSD047/MSD101).

** Для аналоговых сигналов предпочтительнее использовать 590КН2/КН5/КН13.

*** В стандартной КМОП не существует.

ПРИЛОЖЕНИЕ 4

Словарь часто встречающихся аббревиатур и терминов

Разработчики электронных приборов — большие любители сокращений, которые нередко приводят без дополнительных пояснений. Некоторые из часто встречающихся английских аббревиатур расшифровываются в табл. П4.1.

***Таблица П4.1.** Английские аббревиатуры*

Термин	Расшифровка
AC (alternating current)	Переменный ток
ADC (analog-to-digital converter)	Аналого-цифровой преобразователь, АЦП
AF (audio frequency)	Звуковая частота
AM (amplitude modulation)	Амплитудная модуляция, АМ
ATM (asynchronous transfer mode)	Асинхронный режим передачи
CLK (clock)	Тактовый сигнал
CMOS (complementary metal-oxide semiconductor)	Комплементарная структура металл-окисел-полупроводник, КМОП
CPU (central processing unit)	Центральный процессор, ЦП, ЦПУ
DAC (digital-to-analog converter)	Цифроаналоговый преобразователь, ЦАП
DC (direct current)	Постоянный ток
EIA/TIA (Electronics Industry Association/Telecommunications Industry Association)	Ассоциация электронной промышленности/ Ассоциация телекоммуникационной промышленности
EEPROM (electrically erasable programmable read-only memory)	Электрически стираемое программируемое постоянное запоминающее устройство, ЭСППЗУ
EPROM (erasable programmable read-only memory)	Стираемое программируемое постоянное запоминающее устройство, СППЗУ
FET (field-effect transistor)	Полевой транзистор
FLOP (floating octal points)	Плавающая восьмеричная точка
FM (frequency modulation)	Частотная модуляция, ЧМ

Таблица П4.1 (продолжение)

Термин	Расшифровка
FPGA (field-programmable gate array)	Логическая матрица, программируемая пользователем, ПЛИС (программируемая интегральная схема)
GND (ground)	«Земля», корпус, общий
GPU (general processing unit)	Главный процессорный модуль
IC (integrated circuit)	Интегральная схема, ИС
IEC (International Electrotechnical Commission)	Международная электротехническая комиссия
IR (infrared)	Инфракрасный, ИК
ISO (International Standards Organization)	Международная организация по стандартизации
ITU (International Telecommunication Union)	Международный телекоммуникационный союз
LCD (liquid-crystal display)	Жидкокристаллический индикатор, ЖКИ
LED (liquid emitting diode)	Светодиод
LPF (lowpass filter)	Фильтр низких частот, ФНЧ
MODEM (modulator/demodulator)	Модулятор/демодулятор
MOSFET (metal-oxide semiconductor field-effect transistor)	Полевой транзистор структуры металл-окисел-полупроводник, МОП-транзистор
MPU (microprocessor unit)	Микропроцессорный модуль
MUX	Мультиплексор
NC (not connected)	Свободный, не подсоединенный вывод
NR (noise reduction)	Шумоподавление
PA (power amplifier)	Усилитель мощности, УМ
PCB (printed circuit board) или **PWB** (printer wiring board)	Печатная плата, ПП
PCI (peripheral component interconnect)	«Соединение периферийных компонентов», стандарт передачи данных
PM (phase modulation)	Фазовая модуляция, ФМ
ppb (parts per billion)	Частей на миллиард
ppm (parts per million)	Частей на миллион
PWM (pulse width modulation)	Широтно-импульсная модуляция (ШИМ)
RAM (random access memory)	Запоминающее устройство с произвольной выборкой (ОЗУ)
ROM (read only memory)	Постоянное запоминающее устройство (ПЗУ)
RX (receiver/receive)	Приемник/прием, ПРМ
TCR (temperature coefficient of resistance)	Температурный коэффициент сопротивления, ТКС
T/R (transmit/receive)	Прием/передача

Таблица П4.1 (окончание)

Термин	Расшифровка
TTL (transistor-transistor logic)	Транзисторно-транзисторная логика, ТТЛ
TX (transmit/transmitter)	Передача/передатчик, ПРД
V-REF (voltage reference)	Опорное напряжение
Vss (voltage super source)	Напряжение питания

Далее приведен перевод некоторых терминов, часто встречающихся в технической документации. Термины, вошедшие в русский язык в оригинальном звучании или близком к нему (transistor, resistor, logic, timer, emitter и т. п.) и потому понятные без перевода, за некоторыми исключениями не приводятся. Не приводятся также термины и сокращения, подробно рассмотренные в тексте соответствующих глав (SRAM, DRAM, EEPROM и т. п.).

Соответствие терминов на русском их переводу на английский

- **Блок** (узел, устройство) — unit
  Центральный процессорный блок — central processor unit, CPU
- **Внешний** — external
- **Внутренний** — internal
- **Восьмеричный** — octal
- **Вход** — input
- **Вывод** (компонента) — pin, lead
- **Выпрямитель** — rectifier
- **Выход** — output
- **Вычитание** — subtraction
- **Генератор тактирующих импульсов** — clock
- **Данные** data
- **Двоичный** — binary
- **Действующий, ее** (значение, напряжение) — effective
- **Деление** — division
  Делитель — divisor
- **Десятичный** — decimal
- **Диапазон** — range, scale
- **Дистанционное управление** — remote control
- **Доступ** — access
- **Дрейф** — drift
- **Емкость** — capacity, capacitance
- **Задержка** — delay
- **Заряд** — charge
- **Затвор** — gate
- **Земля** — ground
- **Измерение** — measuring
- **Индуктивность** (катушка индуктивности) — coil
- **Исток, источник** — source
- **Канал** — channel
  Канал передачи данных — data transfer channel
- **Кнопка** — button, key
- **Конденсатор** — capacitor
- **Корпус** — case, package
- **Коэффициент усиления** — gain
  Коэффициент усиления по напряжению — voltage gain
- **Мост** — bridge
  Мост выпрямительный — rectifier bridge
- **Мощность** — power
- **Набор** — **kit**
- **Напряжение** — voltage
  Высокий уровень напряжения — high voltage
  Низкий уровень напряжения — low voltage
  Напряжение питания — supply voltage
  Напряжение смещения — bias voltage
  Падение напряжения — dropout voltage
  Опорное напряжение — reference voltage

- **Ноль** — zero
- **Объединение** (каналов) — multiplex
- **Отношение** — ratio
- **Пайка** — soldering
- **Память** — memory
- **Панель** (для микросхем) — socket
- **Параллельный** — parallel
- **Передатчик** — transmitter
- **Переключатель** — switch
- **Период** (импульсов) — cycle
- **Питание** — power

  Источник питания — power supply
- **Плата** — board
- **Поддержка** — support
- **Показатель** — rate
- **Полоса** (частот) — band

  Ширина полосы — bandwidth
- **Полупроводник** — semiconductor
- **Поправка** — correction
- **Последовательный** — serial
- **Предел** — limit
- **Преобразователь** — converter

  Аналого-цифровой преобразователь — analog-to-digital converter, ADC
- **Приемник** — receiver
- **Проверка, контроль** — check
- **Провод** — wire

  Гибкий провод (шнур) — cord
- **Проводник** — conductor
- **Произвольный** — random
- **Прямой** — direct
- **Регулировать** — adjuist, control

  Регулировка — adjuistment
- **Режим (работы)** — mode
- **Синхронизация** — clock
- **Сложение** — adding
- **Смещение** — offset
- **Соединение** — connect
- **Соединитель** (разъем) — connector
- **Состояние** — state
- **Стирание** — erase
- **Сток** — drain
- **Сторожевой** (таймер) — watchdog
- **Схема** — circuit
- **Счетчик** — counter
- **Ток** — current

  Ток базы — base current

  Втекающий ток — sink current

  Вытекающий ток — source current

  Ток насыщения — saturation current

  Переменный ток — alternating current, AC

  Постоянный ток — direct current, DC

  Ток покоя — quiescent current

  Ток нагрузки — load current

  Ток смещения — bias current

  Сила тока — amperage
- **Точность** (погрешность) — accuracy
- **Умножение** — multiplication

  Умножитель — multiplier
- **Управление** — control

  Центральное устройство управления — mean control unit, MCU
- **Усилитель** — amplifier
- **Установка** — set

  Начальная установка (переустановка) — reset
- **Устройство** — device
- **Утечка** — leakage
- **Хранение** — storage
- **Частота** — frequency
- **Шестнадцатеричный** — hexadecimal
- **Шина** — bus
- **Элемент** (гальванический) — cell, battery

Соответствие терминов на английском их переводу на русский

- **AC** (alternating current) — переменный ток
- **Access** — доступ
- **Accuracy** — точность (погрешность)
- **ADC** (analog-to-digital converter) — аналого-цифровой преобразователь
- **Adding** — сложение
- **Adjuist** — регулировать
- **Adjuistment** — регулировка
- **Amperage** — сила тока
- **Amplifier** — усилитель
- **Band** — полоса (частот)
- **Bandwidth** — ширина полосы
- **Battery** — элемент (гальванический)
- **Bias** — смещение; напряжение смещения
- **Billion** — миллиард
- **Binary** — двоичный
- **Board** — плата
- **Bridge** — мост

  Rectifier Bridge — выпрямительный мост
- **Bus** — шина
- **Button** — кнопка, клавиша
- **Capacitor** — конденсатор
- **Capacity**, **capacitance** — емкость
- **Case** — корпус
- **Cell** — ячейка, элемент (гальванический)
- **Channel** — канал

  Data transfer channel — канал передачи данных
- **Charge** — заряд
- **Check** — проверка, контроль
- **Circuit** — схема
- **Clock** — синхронизация; генератор тактирующих импульсов
- **Coil** — индуктивность (катушка индуктивности)
- **Conductor** — проводник
- **Connect** — соединение
- **Connector** — соединитель (разъем)
- **Control** — управлять, регулировать, управление
- **Converter** — преобразователь
- **Cord** — гибкий провод (шнур)
- **Correction** — поправка
- **Counter** — счетчик
- **CPU** (central processor unit) — центральный процессорный блок
- **Current** — ток

  Base current — ток базы

  Bias current — ток смещения

  Quiescent current — ток покоя

  Load current — ток нагрузки

  Saturation current — ток насыщения

  Sink current — втекающий ток

  Source current — вытекающий ток
- **Cycle** — период (импульсов)
- **Data** — данные
- **DC** (direct current) — постоянный ток
- **Decimal** — десятичный
- **Delay** — задержка
- **Device** — устройство
- **Direct** — прямой
- **Division** — деление
- **Divisor** — делитель
- **Drain** — сток
- **Drift** — дрейф
- **Effective** — действующий (значение, напряжение)
- **Erase** — стирание
- **External** — внешний
- **Frequency** — частота
- **Gain** — коэффициент усиления
- **Gate** — затвор (полевого транзистора); логический элемент, вентиль (AND gate)
- **Ground** — земля
- **Hexadecimal** — шестнадцатеричный
- **Input** — вход
- **Internal** — внутренний
- **Key** — кнопка
- **Kit** — набор
- **Lead** — вывод (компонента)
- **Leakage** — утечка
- **Limit** — предел
- **Loop** — контур обратной связи; цикл (в программе)

- **MCU** (mean control unit) — центральное устройство управления
- **Measuring** — измерение
- **Memory** — память
- **Mobile** — мобильный
- **Mode** — режим (работы)
- **Mount** — монтировать
- **Multiplex** — объединение (каналов)
- **Multiplication** — умножение
- **Multiplier** — умножитель
- **Octal** — восьмеричный
- **Offset** — смещение
- **Output** — выход
- **Package** — корпус
- **Parallel** — параллельный
- **Pin** — вывод (компонента)
- **Power** — мощность, питание

  Power supply — источник питания
- **Random** — произвольный, случайный
- **Range** — диапазон
- **Rate** — показатель
- **Ratio** — отношение
- **Receiver** — приемник
- **Rectifier** — выпрямитель
- **Remote** — удаленный

  Remote control — дистанционное управление
- **Reset** — переустановка; начальная установка
- **Scale** — диапазон
- **Semiconductor** — полупроводник
- **Serial** — последовательный
- **Set** — установка
- **Socket** — панель (для микросхем)
- **Soldering** — пайка
- **Source** — исток, источник
- **State** — состояние
- **Storage** — хранение
- **Subtraction** — вычитание
- **Support** — поддержка
- **Switch** — переключатель
- **Transmitter** — передатчик
- **Unit** — блок (узел, устройство)
- **Value** — значение
- **Voltage** — напряжение

  Dropout voltage — падение напряжения

  High voltage — высокий уровень напряжения

  Low voltage — низкий уровень напряжения

  Supply voltage — напряжение питания

  Voltage gain — коэффициент усиления по напряжению
- **Watchdog** — сторожевой (таймер)
- **Wire** — провод
- **Zero** — ноль

ПРИЛОЖЕНИЕ 5

Единицы измерения и обозначения

Физические величины и их единицы измерения по умолчанию

- I — ток, ампер (А)
- U — напряжение, вольт (В, V)
- R — электрическое сопротивление, ом (Ом, Ohm)
- E — энергия, джоуль (Дж)
- N, P — электрическая мощность, ватт (Вт, W)
- W — тепловая мощность, ватт (Вт, W)
- C — электрическая емкость, микрофарада (мкФ, μF)
- Q — заряд, кулон (Кл)
- t — время, секунда (с)
- T, θ — температура, градус Цельсия (°С)
- L — длина, метр (м)
- S — площадь ($м^2$)
- ρ — удельное сопротивление ($ом \times м/мм^2$).

Примечание

Все наименования единиц измерения, которые названы по фамилиям ученых, в сокращении пишутся с большой буквы (вольт — В, ом — Ом, ватт — Вт, ампер — А), а все остальные — с маленькой (секунда — с, метр — м).

Приставки и множители для образования десятичных кратных и дольных единиц

Приставка			Множитель
Наименование	Обозначение		
	русское	международное	
экса	Э	E	10^{18}
пета	П	P	10^{15}
тера	Т	T	10^{12}
гига	Г	G	10^{9}
мега	М	M	10^{6}
кило	к	k	10^{3}
гекто	г	h	10^{2}
дека	да	da	10
деци	д	d	10^{-1}
санти	с	c	10^{-2}
милли	м	m	10^{-3}
микро	мк	μ	10^{-6}
нано	н	n	10^{-9}
пико	п	p	10^{-12}
фемто	ф	f	10^{-15}
атто	а	a	10^{-18}

Некоторые буквенные обозначения в электрических схемах

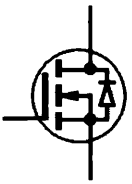
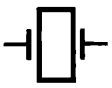
- R — резистор
- C — конденсатор
- VT — транзистор
- VD — диод
- H — цифровой индикатор
- K — тумблер, кнопка
- P — реле
- DA — аналоговая микросхема

- DD — цифровая микросхема
- D — микросхема, содержащая аналоговые и цифровые узлы
- Q — кварцевый резонатор
- L — лампа (газоразрядная или накаливания)

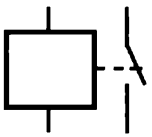
Некоторые символические обозначения в электрических схемах

Изображение	Элемент
	Постоянный резистор
	Переменный резистор
	Подстроечный резистор
	Резистор с зависимостью от температуры (терморезистор, термистор)
	Конденсатор
	Полярный (электролитический) конденсатор
	Диод
	Стабилитрон
	Двусторонний стабилитрон
	Тиристор
	Светодиод

(продолжение)

Изображение	Элемент
	Фотодиод
	Биполярный транзистор (n-p-n)
	Биполярный транзистор (p-n-p)
	Полевой транзистор с pn-переходом и n-каналом
	Полевой транзистор с pn-переходом и p-каналом
	Полевой транзистор с изолированным затвором и n-каналом (n-МОП)
	Полевой транзистор с изолированным затвором и p-каналом (p-МОП)
	Мощный полевой транзистор с изолированным затвором (MOSFET)
	Кварцевый или керамический резонатор
	Нормально замкнутый контакт (реле, тумблера, кнопки)
	Нормально разомкнутый контакт (реле, тумблера, кнопки)

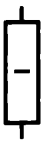
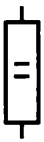
(окончание)

Изображение	Элемент
	Реле с нормально разомкнутым контактом
	Лампа накаливания
	Лампа газоразрядная (неоновая)
	Динамическая звуковая головка
	Общий провод
	«Земля»

Символические обозначения мощности резисторов на схемах

Изображение	Резистор
	Резистор 0,125 Вт
	Резистор 0,25 Вт

(окончание)

Изображение	Резистор
	Резистор 0,5 Вт
	Резистор 1 Вт
	Резистор 2 Вт
	Резистор 10 Вт

ПРИЛОЖЕНИЕ 6

Программирование Arduino Mini и Pro Mini

Как уже говорилось в *главе 20*, платы Arduino Mini и Pro Mini не имеют встроенного преобразователя для подключения к компьютеру. За счет этого они меньше потребляют энергии от источника питания и в целом проще и компактнее. Из-за этих свойств Arduino Mini удобно применять в автономных устройствах на батарейках. Однако встает вопрос о том, как их программировать. Не изучать же специально для этой цели довольно муторный способ программирования через разъем ISP, правда?

Есть два основных способа программирования Arduino Mini и Pro Mini. Первый из них заключается в использовании платы Uno в качестве переходника, второй — в применении различных адаптеров USB-UART. Причем второй встречается в двух разновидностях: с ручным и автоматическим сбросом контроллера. Рассмотрим эти способы по порядку в применении к Arduino Mini, а также посмотрим, что изменится в отношении Pro Mini, имеющего дополнительные выводы.

Программирование Mini с помощью платы Arduino Uno

Это наиболее простой способ. Единственная его сложность заключается в том, что из платы Uno перед программированием необходимо извлечь контроллер. В плате Uno контроллер ATmega328 не впаян в плату, а установлен на панельку, потому ее и удобно использовать. Мы как бы подменяем один контроллер на другой, установленный на плате Mini. Неудобство в том, что для этого нужно под рукой иметь свободную плату Uno, что оказывается довольно накладно.

Для использования Uno в качестве адаптера сначала нужно удалить имеющуюся на нем микросхему контроллера. При последующей вставке ее обратно будьте внимательны с ориентацией: микросхема размещается «вверх ногами» относительно надписей на плате (см. рис. 20.1).

После извлечения контроллера следует соединить следующие пять контактов платы Uno с *одноименными* выводами Mini: питание 5V, общий провод GND, выход

данных TX, вход данных RX и вывод сброса Reset. Последний на плате Mini может называться просто буквой R, а на Pro Mini — RST, см. внешний вид обеих плат на рис. 20.2, *б* и *в*.

После этого подключите плату Uno к USB, как обычно, и в среде Arduino IDE найдите соответствующий ей COM-порт (**Инструменты | Порт**). Обязательно нужно указать плату Arduino Mini в пункте **Инструменты | Плата**. Arduino Pro Mini можно программировать точно так же, как Mini, только обязательно нужно указать правильный тип платы (**Инструменты | Плата | Arduino Pro or Pro Mini**). При этом вывод Reset платы Uno можно подключать, как сказано ранее, к выводу R (RST) платы Pro Mini, а можно к выводу GRN (DTR), о котором далее.

Программирование Mini через адаптер USB-UART

Эта также несложная задача, которая затрудняется поистине необъятным количеством разновидностей переходников USB-UART, имеющихся в продаже. К тому же они могут продаваться под различными названиями: USB-UART, USB-TTL, USB-Serial и т. д. Они могут встречаться и в виде небольшой платки с разъемом USB на одной стороне (причем разъемы могут быть различных типов), и в виде законченного кабеля.

Кабели мы тут подробно рассматривать не будем, т. к. они встречаются существенно реже, а принципы обращения с ними такие же, как с адаптерами в виде плат. Плата адаптера должна иметь, как минимум, четыре контакта: два вывода данных: RX и TX и два вывода питания: +5 и GND. Иногда дополнительно выводится питание 3,3 вольта для плат с таким напряжением питания.

Описываемый здесь способ универсален и годится для адаптера USB-Serial абсолютно любой конструкции, лишь бы он имел указанные выводы. Один из простейших вариантов такой разновидности адаптеров показан на рис. П6.1.

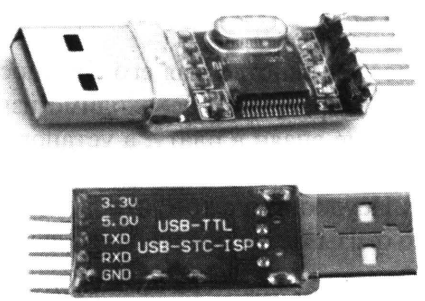


Рис. П6.1. Простой адаптер USB-UART

Четыре указанных вывода подключаются к соответствующим контактам платы Mini или Pro Mini, только надо учесть, что в отличие от соединения через Uno, выводы последовательного порта здесь подключаются *в перекрестном порядке*: RX

адаптера — к TX контроллера, а TX адаптера — к RX котроллера. И никак иначе! В Интернете можно найти упоминания о существовании плат адаптеров, где обозначения RX и TX перепутаны местами, но за такие изыски производителей автор этих строк не отвечает.

Вывод Reset контроллера здесь остается неподключенным. Плату Pro Mini может быть удобнее подключать через торцевой разъем, в отверстия которого впаивается колодка с контактами, параллельными плате. На этом разъеме продублированы выводы питания и обмена данными RX и TX. После соединения платы с контроллером подключите ее к компьютеру и укажите в меню **Инструменты | Порт** появившийся порт адаптера. Не забудьте указать тип платы (Arduino Mini или Pro Mini).

После этого можно приступать к загрузке скетча, которая здесь имеет одну неприятную особенность. Так как вывод Reset никуда не подключен, то сброс придется производить вручную. Для этого надо запустить процесс загрузки, а затем успеть нажать на кнопку Reset платы Mini в момент, когда в среде Arduino надпись «Компиляция скетча...» сменится надписью «Загрузка...». Кнопка расположена в углу платы Mini около разъема (см. внешний вид плат на рис. 20.2, *б* и *в*). Период ожидания сброса контроллера длится довольно долго, потому промахнуться трудно, но определенные неудобства такой способ все-таки доставляет.

Программирование Arduino Mini через адаптер USB-UART с автоматическим сбросом

Автоматический сброс в процессе программирования гораздо удобнее ручного. При загрузке программы через плату Uno такой сброс осуществляется через адаптер USB, имеющийся на плате. А при программировании через внешний переходник USB-UART способ сброса приходится изобретать отдельно.

Заметки на полях

В Сети можно найти огромное количество рекомендаций по вопросу программирования Mini и Pro Mini. Все эти советы, как правило, страдают одним и тем же недостатком: они применимы только к какому-то конкретному виду платы или кабеля USB-UART. Поэтому они часто противоречат друг другу и разобраться бывает очень непросто. Секрет тут исключительно в отсутствии устоявшейся терминологии, из-за чего одни и те же контакты могут называться совершенно по-разному. Такое встречается даже на платах Pro Mini, у которых хорошо известен разработчик оригинальной схемы, не говоря уж о неисчислимом разнообразии плат преобразователей USB-UART. Поэтому вывод, который нужно подключать к контакту Reset платы Arduino, приходится поискать.

На эти несуразности со стороны производителей накладывается тотальное непонимание назначения выводов RTS и DTR последовательного порта со стороны пользователей. Однако, если разобраться, все оказывается довольно просто.

Для сброса контроллера перед программированием через последовательный порт можно использовать дополнительные сигналы порта, которые обеспечиваются

микросхемами адаптеров. Среди них есть два сигнала, которые выставляются в момент готовности компьютера к закачке данных: это сигналы RTS (Request To Send) и DTR (Data Terminal Ready). Название первого из них переводится как «запрос передачи данных», второго — как «готовность терминала к обмену данными».

Любой из этих сигналов можно приспособить к сбросу контроллера, с одним небольшим нюансом. Если просто подключить Reset контроллера к какому-нибудь из этих выводов, то он так и останется в состоянии сброса, и не сможет принимать данные — загрузка зависнет, не начавшись. Потому на практике поступают очень просто: между соответствующим выводом адаптера и контактом Reset платы Arduino устанавливают конденсатор емкостью 0,1 мкФ. Через него проходит только перепад импульса сброса, а в остальное время на входе Reset контроллера нормальный рабочий уровень.

Зная это, нам осталось разобраться в том, где искать на платах адаптеров вывод RTS или DTR. Собственно, в этом и заключается основная сложность, потому что на некоторых платах он просто отсутствует (см. изображение простейшего адаптера в предыдущем разделе), и тут уж ничего не поделаешь. А на тех платах, где такой вывод имеется, он может называться очень по-разному.

Давайте сначала рассмотрим канонический способ программирования Mini, который предлагается на некоторых официальных сайтах Arduino. Там подразумевается плата адаптера под названием USB Serial Light, которая промаркирована логотипом Arduino, т. е. также может считаться образцовой. Внешний вид этой платы показан на рис. П6.2. Как видите, вывод для сброса здесь обозначен понятной надписью Ext Reset (т. е. «внешний сброс»), его и нужно подключать к контакту сброса платы Reset через конденсатор 0,1 мк. Все остальное производится так же, как в предыдущем способе, за одним исключением: кнопку Reset теперь нажимать руками не нужно.

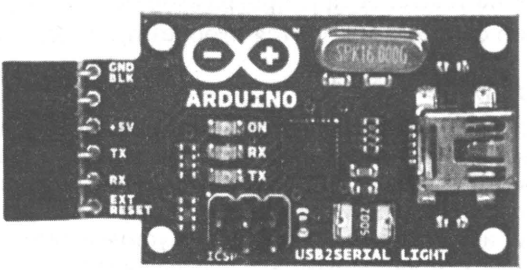


Рис. П6.2. Фирменный адаптер USB-UART от Arduino

Этот способ полностью годится и для случая Pro Mini, не следует только забывать устанавливать правильное название платы перед программированием. Однако здесь можно все еще больше упростить. В дополнительном ряду выводов на торце платы у Pro Mini крайний вывод специально предназначен для подачи сигнала сброса при программировании. На изображении Pro Mini (см. рис. 20.2, *в*) этот вывод находится в левом верхнем углу и носит название GRN. Это крайне неудачное сокращение придумала компания SparkFun, которая выпускает Arduino Pro Mini, и его все время

путают с GND. На некоторых платах Pro Mini он носит более внятное название DTR, аналогично дополнительному выходу порта.

Но вся эта путаница не должна вас смущать — вывод Arduino Pro Mini, который называется DTR (GRN), предназначен именно для подключения сигнала DTR или RTS от платы адаптера USB-UART. На плате Pro Mini этот вывод уже подключен к выводу Reset контроллера через конденсатор 0,1 мкФ, т. е. в случае его использования конденсатор ставить не нужно. Кстати, на загадочное название противоположного вывода разъема BLK вообще обращать внимания не стоит: на плате он соединен с соседним выводом «земли», и на некоторых платах они оба так и называются GND.

Различные названия выводов, пригодных для сброса плат Mini и Pro Mini, иллюстрируются двумя моделями адаптеров, изображенных на рис. П6.3. Осталось разобраться с многочисленными их вариантами, которых еще больше, чем для Pro Mini. Для примера рассмотрим вариант «продвинутых» вариантов адаптеров, два образца которых на разных микросхемах преобразователей как раз на рис. П6.3 и показаны. Отметим, что среди боковых выводов на этих платах (на картинках видны отверстия сверху и снизу платы) имеется и вывод DTR. Но для подключения сброса контроллера к нему подпаиваться необязательно — на основном разъеме имеется вывод RTS, который вполне работает в нужном качестве без всяких доработок платы.

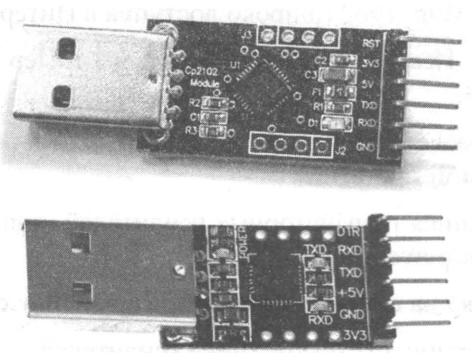


Рис. П6.3. Расширенные адаптеры USB-UART

Кроме того, здесь надо учитывать, что на некоторых «расширенных» платах питание, которое подается на Arduino (вывод тогда обозначается VCCIO), может переключаться перемычкой-джампером. Обычно она желтого цвета, а ее положения понятны из подписей на плате.

Литература и источники

1. Элементарный учебник физики, тома II, III. / под ред. Г. С. Ландсберга. — М.: Наука, 1971.
2. Физические величины: справочник. / под ред. И. С. Григорьева, Е. З. Мейлихова. — М.: Энергоатомиздат, 1991.
3. Ан Пей. Сопряжение ПК с внешними устройствами. Пер. с англ. — М.: ДМК Пресс, 2003.
4. Титце У., Шенк К. Полупроводниковая схемотехника: Справочное руководство. — М.: Мир, 1982 (широко доступна в Интернете в формате Djvu).
5. Хоровиц П., Хилл У. Искусство схемотехники, в 3 т. Пер. с англ. — М.: Мир, изд. 1983, 2001, 2003.
6. Крылов Г. Пример расчета усилителя мощности (**http://geokrilov.com/design.html**).
7. Крылов Г. Схемотехника транзисторных усилителей мощности (**http://geokrilov.com/powersch.html**).
8. Крылов Г. Мой взгляд на субъективизм (**http://geokrilov.com/notes.html**).
9. Ежков Ю. С. Справочник по схемотехнике усилителей.— 2-е изд. — М.: РадиоСофт, 2002. — 272 с.
10. Куликов Г. В. Бытовая аудиотехника (устройство и регулировка). — М.: ПрофОбрИздат, 2001.
11. Гутников В. С. Интегральная электроника в измерительных устройствах. — 2-е изд. — Л.: Энергоатомиздат. Ленингр. отд., 1988.
12. Гнеденко Б. В., Хинчин А. Я. Элементарное введение в теорию вероятностей. — М.: Едиториал УРСС, 2003.
13. Вентцель Е. С. Теория вероятностей. — 7-е изд., стереотипное. — М.: Высшая школа, 2001.
14. Вентцель Е. С., Овчаров Л. А. Теория вероятностей и ее инженерные приложения. — 2-е изд. — М.: Высшая школа, 2000.

15. Горелова Г. В., Кацко И. А. Теория вероятностей и математическая статистика в примерах и задачах с применением Excel. Учебное пособие для вузов. — 2-е изд., исп. и доп. — «Феникс», 2005.

16. Петцольд Ч. Код. — М.: ИТД «Русская редакция», 2001.

17. Андреева Е., Фалина И. Информатика. Системы счисления и компьютерная арифметика. // серия: Информатика — М.: Лаборатория Базовых Знаний, 1999.

18. Шило В. Л. Популярные цифровые микросхемы: Справочник. — М.: Радио и связь, 1987, 1988, 1989.

19. Евстифеев А. В. Микроконтроллеры AVR семейств Tiny и Mega фирмы ATMEL. — М.: Издательский дом «Додэка-XXI», 2008.

20. Ревич Ю. Практическое программирование микроконтроллеров Atmel AVR на языке ассемблера. — СПб.: БХВ-Петербург. — 2-е изд. — 2011, 3-е изд. — 2014.

21. Зубарев А. А. Ассемблер для микроконтроллеров AVR: Учебное пособие. — Омск: Изд-во СибАДИ, 2007. — 112 с. (**http://bek.sibadi.org/fulltext/ED1519.pdf**).

22. Шилдт Г. Полный справочник по C. — 4-е изд. — М.: Вильямс, 2004. (**http://cpp.com.ru/shildt_spr_po_c/index.html**, раздел по элементарным операциям: **http://cpp.com.ru/shildt_spr_po_c/02/0210.html**).

23. Соммер У. Программирование микроконтроллерных плат Arduino/Freeduino. — СПб: БХВ-Петербург, 2012. — 256 с.

24. Монк С. Программируем Arduino. Профессиональная работа со скетчами. — СПб.: Питер, 2017. — 272 с.

25. Ревич Ю. О книге Саймона Монка «Программируем Arduino. Профессиональная работа со скетчами». Портал Geektimes (**https://geektimes.ru/post/285780/**).

26. Ревич Ю. Азбука электроники. Изучаем Arduino. — М.: Изд-во АСТ: Кладезь, 2017. — 224 с.

27. Ревич Ю. Станция для измерения скорости и направления ветра. Портал Geektimes (**https://geektimes.ru/post/289835/**).

Предметный указатель

A

Alkaline 175, 176, 641
Arduino 503–510, 633
◊ матричные индикаторы 558
◊ программирование 515, 522
◊ среда программирования 510
◊ ШИМ 607
ASM Editor 474
AVR
◊ прерывание 462
◊ процедуры деления 497
◊ процедуры умножения 496
◊ семейства 454

B

BOD 458, 478

C

CMOS *См.* Микросхема:КМОП

E

EEPROM 450, 457, 629
EPROM 448

F

Flash-память 450, 634
Fuse-биты 476

I

I^2C *См.* TWI

P

PWM 465

R

RAM 447
ROM 444
RS-232 522
RTC 548, 552, 576

S

SD-карта 613
SRAM 456, 634

T

TTL *См.* Микросхема:ТТЛ
TWI 537

U

UART 466, 522

W

Watchdog 599

X

Xbee-модуль 586, 587
◊ подключение 587

А

Аккумулятор 73, 179, 642
Алгебра Буля 311
АЛУ 437
Ампер 23
Амперметр 29, 35
Амплитуда 74
Анод 110
Антидребезг 358, 378, 386
Аппроксимация 304
Ассемблер 473
АЦП 408
◊ двойного интегрирования 419
◊ интегрирующие 410
◊ однократного интегрирования 410
◊ параллельного действия 408
◊ последовательного приближения 409
◊ с двойным интегрированием 413
◊ с уравновешиванием заряда 418

Б

База 112
Бит 323, 327
Бокорезы 50

В

Ватт 77
Величина переменная 74
Вольт 23
Вольтметр 29, 36
Выпрямитель двухполупериодный 187

Г

Гальванический элемент 73
Генератор 33
◊ аналоговый 265
◊ кварцевый 371
◊ прямоугольных импульсов 363
◊ релаксационный 221
◊ цифровой лабораторный 396
Герц 74
Гистерезис 276

Д

Датчик
◊ атмосферного давления 543
◊ ветра 545
◊ дождя 545
◊ температуры 272, 274, 285, 295, 425, 555
◊ температуры и влажности 540
◊ трехпроводная схема включения 425
Демонтаж 63
Децибел 83
Диод 109
◊ вольтамперная характеристика 110
◊ германиевый 110
◊ защитный 240
◊ кремниевый 110
◊ характеристики 111
◊ Шоттки 110
Дисперсия 302
Дисплей 549
◊ I^2C-интерфейс 584
◊ LCD 566
◊ OLED 571
◊ графический 579
◊ жидкокристаллический 570
◊ кириллица 569, 573
◊ строчный 571–76
◊ текстовый режим 568
◊ цифровой 551
Доверительная вероятность 303
Доверительный интервал 303
Дрейф 298
Дроссель 230

Е

Елочная гирлянда 486
Емкость
◊ определение 93
◊ распределенная 97

З

Заземление 213, 230
Закон
◊ Джоуля — Ленца 77
◊ Мерфи 47
◊ Ома 24
◊ трансформатора 185
Заряд 23
Земля 213
ЗУПВ 447

И

ИК-пульт 625
Импульс управляющий 219
Индикаторы
◊ жидкокристаллические 150
◊ семисегментные 149

Индуктивность 105, 230
Интегратор 263
Источник
◊ тока 30, 265
◊ напряжения 30
◊ питания 38, 183
  ▫ бестрансформаторный 183, 215
  ▫ вторичный 183
  ▫ двухполярный 190, 192
  ▫ импульсный 205
  ▫ нестабилизированный 187
  ▫ простейший 187
  ▫ регулируемый 195
  ▫ стабилизированный 190
  ▫ трансформаторный 183

К

Калибровка 546
Каскад дифференциальный 123
Катод 110
Кварц 370
Класс точности 306
Колебания
◊ амплитуда 74
◊ суммирование 76
Коллектор 112
Компаратор 272, 279
Комплементарный 131
Конденсатор 92
◊ в цепи переменного тока 102
◊ емкость 93
◊ заряд и разряд 94
◊ неполярный 99
◊ номинальные значения 87
◊ параллельное и последовательное соединение 101
◊ развязывающий 163, 212
◊ фильтрующий 188, 191
◊ электролитический 98
Конденсаторы: номинальные значения 640
Конфигурационные биты 476
Корпус
◊ изготовление в домашних условиях 66
◊ нанесение надписей 66
Коэффициент
◊ диэлектрической проницаемости 97
◊ мощности 107
◊ ослабления синфазного сигнала (КООС) 84
◊ передачи 162
◊ передачи в системе с обратной связью 255
◊ сопротивления температурный, ТКС 27
◊ температурный резисторов 88
◊ удельного сопротивления 27
◊ усиления 161
◊ усиления ОУ 255

Л

Лейденская банка 92
Литиевый элемент 176
Логика 309
Логические
◊ микросхемы 341, 349
◊ переменные 311, 312
◊ схемы на транзисторах 318
Логическое
◊ отрицание 311
◊ сложение 311
◊ умножение 311

М

Магнитопровод 186
Математическая статистика 297
Математическое ожидание 302
Меандр 81
Метеостанция 535
Метод наименьших квадратов 305
Мигалка
◊ на Arduino 532
◊ на таймере 490
◊ простейшая 484
Микроконтроллер 436, 452
◊ прерывания 487
◊ программирование 469
◊ таймеры-счетчики 490
Микропроцессор 436
Микросхема
◊ КМОП 341, 344
◊ корпус 241
◊ с открытым коллектором (истоком) 351
◊ ТТЛ 341, 342
◊ цифровая 341
Микросхемы 233
◊ защита от статики 239
◊ схемотехника 237
◊ эксплуатация 239
Монтаж 62
Мост
◊ выпрямительный 187, 224, 226
◊ Уитстона 290
Мощность 77
◊ звукового усилителя 163
◊ регулирование 218, 221
◊ резисторов номинальная 88
◊ управление 218

Мультивибратор 366
Мультиметр 34
Мультиплексор 361

Н

Напряжение 23, 24
◊ действующее значение 79, 219
◊ измерение 219
◊ несинусоидальное 79
◊ определение 23
◊ переменное 74
◊ постоянное 74
◊ прямое падение 110
◊ сетевое 44, 82, 183, 215
◊ синусоидальное 75
◊ среднеамплитудное значение 79
◊ среднее значение 79
◊ фазное 213
◊ эмиттерного перехода 112
Нормальное распределение вероятностей 301
◊ характеристики 302

О

Обратная связь 161, 250
◊ отрицательная 120, 195, 250, 254
▫ принцип действия 251
◊ положительная 252, 276
Одновибратор 376
Олигархи 33
Ом 26
Оптопара 144, 221
Оптрон 144, 224
◊ диодный 224
Осциллограф 41
Отсчет 401
Оцифровка 401
Ошибки измерения
◊ систематические 299
◊ случайные 299
◊ случайные, оценка 299

П

Память
◊ данных (SRAM) 456
◊ программ 455
Паяльник 51
Передатчик/приемник 433 МГц 586, 591
Перерегулирование 270
Период 74
◊ измерение 620
ПЗУ 444
Пинцет 50
Пироэлектрический датчик 623
Питание сетевое 82
Платы печатные 60
Плотность вероятностей 300
Погрешность 298
◊ абсолютная 306
◊ относительная 306
▫ приведенная 306
Полупроводники 109
Помехи 229
Порт ввода/вывода 460
Последовательный порт 465
Постоянная времени
◊ RC-цепи 96
◊ тепловая 270
Правила де Моргана 313
Преобразование
◊ напряжения в ток 90
◊ тока в напряжение 90
Преобразователь
◊ аналого-цифровой, АЦП 408
◊ дельта-сигма 418
◊ инвертор 430
◊ напряжение-время 417
◊ цифроаналоговый, ЦАП 404
Прерывание 462
◊ Reset 489
◊ таймера 1 490
Прерывания 487
Провода 63
◊ обмоточные 64
Программатор 469
Программирование 469, 479
Программная задержка 483

Р

Радиатор 245
Радиаторы 200
◊ методика расчета 200
◊ монтаж 202
Разбаланс 107
Разводка питания 211
Разрешающая способность 298
РВВ (регистр ввода/вывода) 456
Регистр 389
◊ защелка 389
◊ последовательный 389
◊ сдвиговый 390
Регрессия 304
◊ уравнение 305

Резистор 27, 85
◊ допустимое напряжение 88
◊ маркировка 86
◊ МЛТ 86
◊ мощность 88
◊ номинальные значения 86
◊ параллельное соединение 91
◊ переменный 28, 89
◊ подстроечный 90
◊ последовательное соединение 91
◊ прецизионный 88
◊ С1-4 86
◊ самодельный 37
Резисторы 639
◊ номинальные значения 640
Реле 135, 216
◊ герконовые 136
◊ напряжение срабатывания 139
◊ с самоблокировкой 140
◊ ток срабатывания 140
◊ электромагнитное 135
◊ электронные 145, 221, 274
РОН (регистр общего назначения) 456

С

Светодиод 143, 146
◊ особенности включения 148
◊ падение напряжения 149
Сигнал 81
Симистор 218, 226
Синхронизация луча осциллографа 43
Система счисления 320
◊ двоичная 323
◊ перевод 324
◊ позиционная 321
◊ шестнадцатеричная 323
Скважность 81
COM-порт 522
Сопротивление 24
◊ внутреннее 30
◊ определение 26
◊ реактивное 102, 105
◊ резисторов 87
◊ удельное 27
Среднеквадратическое отклонение 303
Стабилизатор 190
◊ интегральный 193
Стабилитрон 141, 198
◊ вольт-амперная характеристика 142
◊ двусторонний 142
Сумматор 353
◊ аналоговый 263
◊ двоичный 353
◊ одноразрядный 356
◊ полный 356
Схема
◊ антидребезга 358, 378, 386
◊ АЦП с двойным интегрированием 414
◊ АЦП с однократным интегрированием 410
◊ АЦП типа 418
◊ включения интегрального стабилизатора 194
◊ включения светодиода 148
◊ включения тиристора 216
◊ выделения фронтов 375
◊ генератора аналогового 265
◊ генератора релаксационного 364
◊ генератора цифрового 397
◊ двоичного полусумматора 356
◊ двухполярного нестабилизированного источника 190
◊ дешифратора семисегментного 444
◊ дифференциального каскада 123
◊ дифференциального усилителя 260
◊ задержки импульса 376
◊ защиты входа микрофонного усилителя 142
◊ защиты выводов микросхем 240
◊ звонка 140
◊ звуковой сигнализации 368
◊ идеального источника тока 265
◊ измерения сопротивления 291
◊ измерения тока 37
◊ инвертирующего усилителя с высоким коэффициентом усиления 258
◊ инвертора напряжения 430
◊ интегратора 263
◊ исключающее ИЛИ 355
◊ кварцевого генератора 371
◊ компаратора с гистерезисом 276
◊ лабораторного источника питания 195
◊ логическая диодно-транзисторная 318
◊ логическая на выключателях 314
◊ логическая на реле 315
◊ микропроцессора 438
◊ мостика Уитстона 290
◊ мультивибратора 366
◊ мультивибратора с управлением 367
◊ одновибратора 377
◊ плавного включения ламп 228
◊ повторителя 256
◊ повторителя/инвертора 263
◊ подключения переменного резистора 90
◊ портов ввода/вывода 441

◊ предустановки счетчика 395
◊ проверка закона Ома 24
◊ простейшего источника питания 187
◊ разводки питания 211
◊ расщепителя напряжения 429
◊ реле с самоблокировкой 140
◊ ручного регулятора мощности 221, 225
◊ с общим коллектором 119
◊ с общим эмиттером 117
◊ сетевого фильтра 230
◊ системы с отрицательной обратной связью 254
◊ стабилизатора двухполярного 192
◊ стабилизатора на стабилитроне 190
◊ стабилизатора на транзисторе 191
◊ стабилизатора параметрического 190
◊ структурная термостата 268
◊ сумматора аналогового 263
◊ сумматора двоичного 356
◊ счетчика асинхронного 390
◊ счетчика реверсивного 393
◊ счетчика синхронного 392
◊ терморегулятора 272
◊ термостата для нагревания воды 277
◊ термостата на Arduino 526
◊ токового зеркала 237
◊ транзистора в ключевом режиме 114
◊ транзистора Дарлингтона 116
◊ транзисторного УМЗЧ 160
◊ триггера RS 385
◊ триггера динамического типа D 388
◊ триггера статического типа D 388
◊ триггера счетного 389
◊ УМЗЧ на микросхемах 243, 244, 246, 247
◊ управления реле 140
◊ усилительного каскада 120
◊ усилителя инвертирующего 257
◊ усилителя неинвертирующего 256
◊ фазового компаратора 360
◊ фильтра высоких частот 103
◊ фильтра низких частот 104
◊ формирователя 374
◊ ЦАП 405, 407
◊ цепи с двумя резисторами 28
◊ цепочки R—2R 407
◊ электромагнитного реле 136
◊ электронного термометра 294, 425
◊ элемента КМОП 342
◊ элемента ТТЛ 342
◊ эмиттерного повторителя 119
Счетчик 390
◊ двоичный 392
◊ реверсивный 393

T

Таймеры-счетчики 464
Тактирование 458
Тастатура 615
Температура 284
Теорема Котельникова 402
Теория вероятностей 297
Термистор 527
Термометр
◊ медный 288, 425
◊ металлический 285
◊ платиновый 288
◊ полупроводниковый 288
◊ ртутный 289
◊ сопротивления 288
◊ цифровой 419, 424
◊ цифровой, калибровка 428
◊ электронный 294, 404
Терморегулятор 272, 277
Термостат 268
◊ на Arduino 525
Тестер 34
Тетрада 326
Тиристор 216
Ток 23, 24
◊ зависимость от напряжения (закон Ома) 24
◊ измерение 35
◊ измерение вольтметром 37
◊ определение 23
Токовое зеркало 237
Точность 298
Транзистор 111, 131
◊ MOSFET 125, 208, 610
◊ биполярный 111
◊ включение с общим коллектором 119
◊ включение с общим эмиттером 115
◊ Дарлингтона 114, 116
◊ дифференциальное включение 123
◊ ключевой режим 113, 114
◊ комплементарный 131, 164
◊ коэффициент усиления по току 114, 117
◊ критерии выбора 129, 131
◊ маломощный 131
◊ мощность 130
◊ мощный 132
◊ полевой 124
◊ полевой с p-n-переходом 124
◊ типа n-p-n 112
◊ типа p-n-p 112
◊ усилительный режим 117
◊ характеристики 130

Трансформатор 175, 183, 187
◊ импульсный 226
◊ определение количества витков 185
◊ расчет 185
Трансформаторы 77
Триак 218, 226
Триггер 375, 384
◊ динамический типа D 388
◊ статический типа D 387
◊ счетный 389
◊ типа RS 384
Трубка кембриковая термоусадочная 65

У

УМЗЧ 160, 243
◊ маломощный 247
Усилитель 168
◊ дифференциальный 260
◊ звуковой 159
◊ звуковой на микросхемах 243
◊ инвертирующий 257
◊ инструментальный 262
◊ мощность 163, 166, 172
◊ неинвертирующий 256
◊ операционный 195, 199
◊ операционный, использование 249
◊ операционный, определение 249
◊ переменного тока 122
◊ режим АВ 166

Ф

Фаза 75, 219
Фарада 94
Фильтр
◊ высоких частот 103
◊ нижних частот 104
◊ развязывающий 213, 230
Флюс
◊ активный 54
◊ спирто-канифольный 53

Ц

ЦАП 404
Цепь
◊ дифференцирующая 103
◊ интегрирующая 104
Цифровой звук 403

Ч

Частота
◊ измерение 620
◊ определение 74
Чип 233

Ш

ШИМ 605

Щ

Щелочной элемент 176, 641

Э

Электронно-лучевая трубка 42
Электрохимический элемент 73, 175, 641
Электроэнергия 82
Элемент
◊ литиевый 176
◊ солевой 178
◊ щелочной 176
Эмиттер 112
Эмиттерный повторитель 119, 161, 162
Энергосбережение 598, 603
ЭСЛ 341

Электроника

занимательная ЭЛЕКТРОНИКА

«Электроника — это искусство, которое основано на нескольких основных законах и включает в себя большое количество практических правил и приемов.»

П. Хоровиц, У. Хилл.

Автор — инженер-электронщик с многолетним стажем. Занимался автоматизацией производств, конструированием измерительных приборов для изучения океана и других научных исследований. В настоящее время журналист и писатель. Основной круг интересов — информационные технологии, их влияние на современное общество, история компьютеров и электронных приборов. Имеет несколько сотен публикаций в журналах, газетах и сетевых изданиях. Автор 10 книг, среди которых «Практическое программирование микроконтроллеров Atmel AVR на языке ассемблера», «Информационные технологии в СССР. Создатели советской вычислительной техники» и др.

За последние несколько десятков лет в мире произошла революция. Мы и не успели опомниться, как компьютеры, мобильные телефоны, «умные» электронные игрушки стали частью нашего быта. В том, что эти чаяния фантастов прошлого стали явью, главную роль сыграла электроника — искусство конструирования и производства электронных схем. Автор вводит читателя в увлекательный мир электроники и схемотехники, на простых примерах показывая, что проектировать и изготавливать электронные приборы самостоятельно — вовсе не такая сложная наука, как кажется. Основной упор при изложении материала делается на творческий подход — так, чтобы читатель всегда знал, как можно улучшить ту или иную конструкцию или включить ее в свой собственный проект. В новом издании существенно расширен раздел про основу современной любительской электроники — платформу Arduino. Книга рассчитана на старших школьников, учащихся техникумов, студентов и широкий круг любителей электроники.

5-е издание

bhv®

БХВ-ПЕТЕРБУРГ
191036, Санкт-Петербург,
Гончарная ул., 20
Тел.: (812) 717-10-50,
339-54-17, 339-54-28
E-mail: mail@bhv.ru
Internet: www.bhv.ru

ISBN 978-5-9775-3961-6