Содержание

Об авторе ... 10

О рецензентах ... 11

Предисловие ... 12

Часть I. ОСНОВЫ ПРОГРАММИРОВАНИЯ ВСТРОЕННЫХ СИСТЕМ И РОЛЬ C++ ... 17

Глава 1. Что такое встроенные системы ... 18

Разнообразие встраиваемых систем ... 18
Микроконтроллеры ... 20
TMS 1000 ... 20
Intel MCS-48 ... 22
Intel MCS-51 ... 24
PIC ... 27
AVR ... 33
M68k и микроконтроллеры на основе Z80 ... 38
ARM Cortex-M ... 38
H8 (SuperH) ... 39
ESP8266/ESP32 ... 39
Другие микроконтроллеры ... 41
Особенности ... 42
Одноплатный компьютер, или система на кристалле ... 42
Особенности ... 43
Резюме ... 44

Глава 2. C++ как язык программирования встроенных систем ... 45

Связь C++ с C ... 45
C++ как язык программирования встроенных систем ... 47
Функциональные возможности языка C++ ... 50
Пространства имен ... 50
Строгая типизация ... 51
Преобразование типов ... 52
Классы ... 52
Наследование ... 55
Виртуальные базовые классы ... 56
Встроенные функции ... 57
Информация о типах во время выполнения ... 58
Обработка исключений ... 58

Шаблоны ... 58
Стандартная библиотека шаблонов ... 59
Удобство сопровождения ... 59
Резюме ... 60

Глава 3. Разработка для встроенной ОС Linux и подобных систем ... 61

Встроенные операционные системы ... 61
Операционные системы реального времени ... 64
Специализированные периферийные устройства и драйверы ... 65
Добавление часов реального времени ... 65
Специализированные драйверы ... 67
Ограничение ресурсов ... 68
Пример: мониторинг клубного зала ... 69
Аппаратные устройства ... 69
Реализация ... 77
Конфигурация сервиса ... 101
Права доступа ... 102
Окончательные результаты ... 102
Пример: простой медиа-плеер ... 103
Резюме ... 105

Глава 4. Встроенные системы с ограниченными ресурсами ... 106

Общий обзор применения малых систем ... 106
Пример: устройство управления лазерным резаком ... 108
Функциональная спецификация ... 110
Проектные требования ... 111
Варианты выбора реализации ... 112
Интегрированные среды разработки и рабочие среды для встроенных систем ... 118
Программирование микроконтроллеров ... 119
Программирование памяти и отладка устройства ... 120
Загрузчик ... 124
Управление памятью ... 124
Стек и динамически распределяемая память ... 126
Прерывания, ESP8266 IRAM_ATTR ... 127
Параллельный режим выполнения ... 129
Разработка для AVR с использованием Nodate ... 130
Вводная информация о Nodate ... 131
Пример: инструмент тестирования интегральной микросхемы CMOS ... 132
Практическое использование ... 137
Разработка для ESP8266 с использованием Sming ... 141
Разработка для микроконтроллеров ARM ... 142
Использование операционной системы реального времени ... 142
Резюме ... 143

Глава 5. Пример: монитор влажности почвы с использованием протокола Wi-Fi 145
Уход за растениями 145
Предлагаемое решение 147
Аппаратура 148
Специализированное программное обеспечение 152
Настройка рабочей среды Sming 152
Код модуля для растения (plant) 154
Компиляция и запись в ПЗУ 182
Первоначальное конфигурирование 183
Использование системы 184
Дальнейшие действия 184
Сложности 185
Резюме 185

Часть II. ТЕСТИРОВАНИЕ, МОНИТОРИНГ 186

Глава 6. Тестирование приложений, предназначенных для конкретных ОС 187
Почему следует избегать разработки на реальной аппаратуре 187
Кросс-компиляция для одноплатных компьютеров 189
Комплексный тест для сервиса управления состоянием клубного помещения 190
Имитация или реальная аппаратура 198
Тестирование с использованием Valgrind 200
Многоцелевая система сборки 201
Удаленное тестирование на реальной аппаратуре 210
Резюме 212

Глава 7. Тестирование платформ с ограниченными ресурсами 213
Снижение степени износа оборудования 213
Планирование проектного решения 214
Системы сборки, независимые от платформы 215
Использование кросс-компиляторов 216
Локальная отладка и отладка на микросхеме 216
Пример: комплексный тест ESP8266 217
Сервер 218
Узел 239
Сборка проекта 265
Резюме 266

Глава 8. Пример: информационно-развлекательная система на основе ОС Linux 268
Одно устройство выполняет все задачи 268
Необходимая аппаратура 269

Требования к программному обеспечению ... 270
Bluetooth-источники и приемники аудио ... 271
Организация потока в режиме онлайн ... 272
Управляемый голосом пользовательский интерфейс ... 273
Использование сценариев ... 273
Исходный код ... 274
Сборка проекта ... 282
Расширение системы ... 283
Резюме ... 284

Глава 9. Пример: мониторинг и управление внутренним микроклиматом в здании ... 285

Растения, помещения и прочее ... 285
История разработки ... 286
Функциональные модули ... 288
Исходный код специализированного ПО ... 288
Ядро ... 288
Модули ... 289
Сервер управления и контроля ... 318
Инструментальное средство администрирования ... 329
Система кондиционирования воздуха ... 329
База данных InfluxDB для записи показаний датчиков ... 332
Вопросы обеспечения безопасности ... 334
Дальнейшие разработки ... 335
Резюме ... 335

Часть III. ИНТЕГРАЦИЯ С ДРУГИМИ ИНСТРУМЕНТАЛЬНЫМИ СРЕДСТВАМИ И РАБОЧИМИ СРЕДАМИ ... 337

Глава 10. Разработка встроенных систем с использованием Qt ... 338

Главное преимущество правильно выбранной рабочей среды ... 338
Использование Qt для приложений с интерфейсом командной строки ... 339
Приложения с использованием графического пользовательского интерфейса Qt ... 341
Qt для встроенных систем ... 344
Графические пользовательские интерфейсы с использованием таблиц стилей ... 345
QML ... 345
3D Designer ... 345
Пример добавления графического пользовательского интерфейса в информационно-развлекательную систему ... 346
Основной файл исходного кода (main) ... 347
QmlInterface ... 347
Резюме ... 364

Глава 11. Разработка для гибридных систем SoC/FPGA 365
Организация исключительно параллельного выполнения 365
Языки описания аппаратуры 367
Архитектура ППВМ 368
Гибридные микросхемы FPGA/SoC 368
Пример: простой осциллограф 369
Аппаратура 370
Код VHDL 371
Код C++ 376
Сборка проекта 379
Резюме 379

Приложение А. Эффективные практические методики 380
Тщательно продуманные планы 380
Работа с аппаратурой 381
Огромный мир периферийных устройств 381
Изучайте свои инструментальные средства 382
Выбор асинхронных методов 382
Изучение спецификаций 383
Обеспечение краткости обработчиков прерываний 383
8 бит означает 8 бит 383
Не следует заново изобретать колесо 384
Подумайте, прежде чем начать оптимизацию 384
Требования – это основа, а не дополнение 384
Документация жизненно важна 385
Тестирование кода означает попытку нарушить его выполнение 386
Резюме 386

Предметный указатель 387

Об авторе

Майа Пош (Maya Posch) – ведущий разработчик на языке C++, обладающий более чем 15-летним опытом практической работы. Открыв для себя сначала столь увлекательную область деятельности, как программирование, а немного позже не менее увлекательную электронику, Майа всегда проявляла живой интерес к технологиям и охотно разделяла свое страстное увлечение с другими.

Сама Майа называет себя разработчиком на языке C, который со временем увлекся языками C++ и Ada. Ей нравится работать в условиях, ограничивающих объем программного кода и возможности аппаратуры до минимума, и создавать в этих условиях новые, эффективные и удобные системы.

Майа также активно интересуется разработками с использованием программируемых пользователем вентильных матриц (ППВМ – FPGA), разработками в области искусственного интеллекта и исследованиями в сфере робототехники, а кроме того, обладает талантами литератора, музыканта и живописца.

О рецензентах

Франс Фаазе (Frans Faase) изучал информационные технологии в университете Твенте (Нидерланды) и получил степень магистра в области проектирования компиляторов и формальных методов. Принимал участие в нескольких научных исследовательских проектах, но в основном работал как инженер по программному обеспечению в промышленной сфере. Франс обладает почти 20-летним опытом практического использования C++ как профессиональный разработчик и исследователь, а кроме того, C++ – это его хобби. Франс принимал активное участие в нескольких проектах с открытым исходным кодом. В последнее время он также приобрел некоторый практический опыт разработки ПО для микроконтроллеров, в основном с использованием среды Arduino.

Патрик Минтрэм (Patrick Mintram) – инженер по программному обеспечению, который начинал профессиональную карьеру как техник по электронному оборудованию. В течение длительного времени работал со встроенными системами, в том числе занимался разработкой и тестированием в средах с повышенными требованиями к безопасности, а также линейным сопровождением и восстановлением систем. Женат, имеет двух кошек Дьюка и Дэзи, в свободное время – «домашний мастер на все руки» и бег трусцой.

Предисловие

Язык программирования C++ не добавляет никаких излишеств, расширяет возможности сопровождения и предлагает многочисленные преимущества над прочими языками программирования, следовательно, представляет собой удачный выбор для разработки встроенных систем. Если вам необходимо создавать автономные или сетевые встроенные системы, обеспечивать их безопасность и рациональное использование памяти, то из этой книги вы точно узнаете, как это делается. Также вы узнаете, как работает C++, будет проведено сравнение с другими языками, используемыми для разработки встроенных систем. Кроме того, описывается методика создания удобных графических интерфейсов пользователя (GUI) для встроенных систем, проектирование привлекательных и функциональных пользовательских интерфейсов, а также методы интеграции проверенных стратегий в конкретные проекты для достижения оптимальной производительности аппаратуры.

В предлагаемой книге подробно рассматриваются разнообразные аппаратные платформы для встроенных систем, поэтому вы получаете возможность выбрать наилучший вариант для своего конкретного проекта. Вы научитесь решать сложные архитектурные задачи после внимательного изучения всех тщательно проработанных программных шаблонов, представленных в этой книге.

Для кого предназначена эта книга

Если вы начинающий разработчик программ на C++ для встроенных систем, то эта книга предназначена для вас. Для полного понимания всех тем книги требуется хорошее знание конструкций языка C++. Какие-либо предварительные знания в области встроенных систем не требуются.

Краткое содержание книги

Глава 1 «Что такое встроенные системы» знакомит читателя со встроенными системами в целом. Обзор разнообразных категорий и примеров встроенных систем дает общее представление о том, что означает термин «встроенный», а также о разнообразии значений этого термина. Рассматриваются многочисленные ранние и современные доступные модели микроконтроллеров и решения систем на одном чипе (на одной плате), которые вы можете обнаружить в существующих системах, а также новые проектные решения.

Глава 2 «C++ как язык программирования встроенных систем» объясняет, почему C++ так же хорош, как C и другие подобные языки. Вообще говоря, C++ не только так же быстр, как C, но еще и не содержит каких-либо излишеств и предлагает многочисленные преимущества, связанные с парадигмами кодирования и удобством сопровождения.

В главе 3 «Разработка для встроенной ОС Linux и подобных систем» рассматриваются методы разработки для встроенных систем на основе ОС Linux и родственных систем на одноплатных компьютерах с учетом различий между разработкой для Linux-подобных систем и систем, основанных на PC.

В главе 4 «Встроенные системы с ограниченными ресурсами» обсуждается планирование эффективного использования ограниченных ресурсов. Основное внимание уделяется правильному выбору микроконтроллера для нового проекта, добавлению периферийных устройств и определению требований к Ethernet-интерфейсам и последовательному интерфейсу в проекте. Также рассматривается пример проекта с использованием микроконтроллера AVR, методы разработки для других архитектур микроконтроллеров и возможности использования операционной системы реального времени.

В главе 5 «Пример: монитор влажности почвы с использованием протокола Wi-Fi» описывается процесс создания системы наблюдения за влажностью почвы с применением протокола беспроводной связи Wi-Fi с дополнительными возможностями управления приводом водяного насоса или аналогичного механизма. При наличии встроенного веб-сервера можно воспользоваться его пользовательским интерфейсом на основе браузера для наблюдения и управления или же интегрировать веб-сервер в более крупную систему, применив REST API.

В главе 6 «Тестирование приложений, предназначенных для конкретных ОС» демонстрируются методы разработки и тестирования приложений, предназначенных для встроенных операционных систем. Вы узнаете, как установить и использовать рабочую среду и конвейер инструментов для кросс-компиляции, как выполнять отладку в удаленном режиме с помощью отладчика GDB, как описать процедуру сборки системы.

В главе 7 «Тестирование платформ с ограниченными ресурсами» рассматриваются методы эффективной разработки для конкретных целевых микроконтроллеров. Также демонстрируется реализация интегрированной среды, позволяющей отлаживать приложения для микроконтроллеров со всеми удобствами, присущими настольной ОС, и с предоставляемым ею комплектом инструментальных средств.

В главе 8 «Пример: информационно-развлекательная система на основе ОС Linux» демонстрируется возможность относительно простого процесса создания информационно-развлекательной системы на одноплатном компьютере с использованием механизма преобразования голоса в текст для формирования пользовательского интерфейса, управляемого голосом. Также описаны возможности расширения системы с добавлением функциональности.

В главе 9 «Пример: мониторинг и управление микроклиматом в здании» рассматривается процесс разработки системы мониторинга и управления климатическими условиями внутри здания, отдельные компоненты этой системы, а также уроки, извлеченные во время разработки.

Глава 10 «Разработка встроенных систем с использованием библиотеки Qt» содержит описание многочисленных способов применения библиотеки и рабочей среды Qt для разработки встроенных систем. Выполняется сравнение с другими рабочими средами, рассматривается оптимизация Qt для встроенных платформ, а также пример использования предварительно разработанного графического пользовательского интерфейса на основе QML, который можно добавить в ранее созданную информационно-развлекательную систему.

В главе 11 «Разработка для гибридных систем SoC/FPGA» рассматривается организация обмена информацией на стороне программируемой пользователем вентильной матрицы (FPGA) в гибридной системе FPGA/SoC. Это помогает читателю понять разнообразные методы реализации алгоритмов в FPGA и применение их на стороне системы на кристалле (SoC). Также описана реализация простого осциллографа на основе гибридной системы FPGA/SoC.

В приложении А «Эффективные практические методики» кратко описан ряд проблем и затруднений, наиболее часто возникающих в процессе проектирования ПО для встроенных систем.

Максимально эффективное использование книги

Требуется практический опыт работы с семейством одноплатных компьютеров Raspberry Pi. Необходим компилятор C++, комплект инструментальных средств для организации (кросс)конвейера GCC ARM Linux, полный набор инструментальных средств AVR, рабочие среды Sming, Valgrind и Qt, а также интегрированная среда разработки (IDE) Lattice Diamond.

Получение файлов исходного кода примеров

Файлы исходного кода примеров из этой книги можно загрузить из вашей учетной записи на сайте **www.packtpub.com**. Если вы приобрели книгу в другом месте, то можно посетить страницу поддержки **www.packtpub.com/support** и зарегистрироваться, чтобы получить эти файлы по электронной почте. Загрузка файлов исходного кода примеров выполняется следующим образом:

1. Войти/зарегистрироваться на сайт/сайте **www.packtpub.com**.
2. Перейти на закладку **SUPPORT**.
3. Щелкнуть по пункту **Code Downloads & Errata**.
4. Ввести название книги в панели ввода **Search**, далее выполнять выводимые на экран инструкции.

После загрузки файла архива необходимо распаковать его, чтобы извлечь содержимое, используя самые свежие версии программ-упаковщиков:

- WinRAR/7-Zip для Windows;
- Zipeg/iZip/UnRarX для Mac;
- 7-Zip/PeaZip для Linux.

Комплект примеров исходного кода для этой книги размещен в репозитории GitHub **https://github.com/PacktPublishing/Hands-On-Embedded-Programming-with-CPP-17**. При любых изменениях в коде немедленно обновляются соответствующие файлы в существующем репозитории GitHub.

Кроме того, вы можете получить другие комплекты исходного кода из нашего обширного каталога книг и видеоматериалов по адресу **https://github.com/PacktPublishing/**. Рекомендуем регулярно проверять его содержимое.

Типографские соглашения, принятые в книге

В этой книге используется несколько стилей выделения некоторых элементов текста.

`Фрагмент кода в тексте` – ключевые слова, операторы, имена переменных и функций непосредственно в тексте. Пример: «Сам класс языка C++ реализован в языке C как структура `struct`, содержащая переменные этого класса».

Фрагмент кода отображается в следующем формате:

```
class B : public A {
   // Закрытые члены класса
public:
   // Дополнительные открытые члены класса
};
```

При необходимости привлечь внимание читателя к некоторой части фрагмента кода соответствующие строки или элементы строк выделяются полужирным шрифтом:

```
class B : public A {
   // Закрытые члены класса
public:
   // Дополнительные открытые члены класса
};
```

Ввод или вывод в командной строке отображается следующим образом:

```
sudo usermod -a -G gpio user
sudo usermod -a -G i2c user
```

Курсив – имена файлов, каталогов и прочих объектов.

Полужирный шрифт – важные (ключевые) слова, элементы пользовательского интерфейса или слова, которые выводятся на экран. Например, пункты меню или элементы диалоговых окон.

Этим значком обозначаются предупреждения или важные замечания.

Этим значком обозначаются советы, подсказки и полезные практические приемы.

Отзывы и пожелания

Мы всегда рады отзывам наших читателей. Расскажите нам, что вы думаете об этой книге – что понравилось или, может быть, не понравилось. Отзывы важны для нас, чтобы выпускать книги, которые будут для вас максимально полезны.

Вы можете написать отзыв прямо на нашем сайте **www.dmkpress.com**, зайдя на страницу книги, и оставить комментарий в разделе «Отзывы и рецензии». Также можно послать письмо главному редактору по адресу **dmkpress@gmail.com**, при этом напишите название книги в теме письма.

Если есть тема, в которой вы квалифицированы, и вы заинтересованы в написании новой книги, заполните форму на нашем сайте по адресу **http://dmkpress.com/authors/publish_book/** или напишите в издательство по адресу **dmkpress@gmail.com**.

Скачивание исходного кода примеров

Скачать файлы с дополнительной информацией для книг издательства «ДМК Пресс» можно на сайте **www.dmkpress.com** на странице с описанием соответствующей книги.

Список опечаток

Хотя мы приняли все возможные меры для того, чтобы удостовериться в качестве наших текстов, ошибки все равно случаются. Если вы найдете ошибку в одной из наших книг – возможно, ошибку в тексте или в коде, – мы будем очень благодарны, если вы сообщите нам о ней. Сделав это, вы избавите других читателей от расстройств и поможете нам улучшить последующие версии данной книги.

Если вы найдете какие-либо ошибки в коде, пожалуйста, сообщите о них главному редактору по адресу **dmkpress@gmail.com**, и мы исправим это в следующих тиражах.

Нарушение авторских прав

Пиратство в интернете по-прежнему остается насущной проблемой. Издательства «ДМК Пресс» и Packt очень серьезно относятся к вопросам защиты авторских прав и лицензирования. Если вы столкнетесь в интернете с незаконно выполненной копией любой нашей книги, пожалуйста, сообщите нам адрес копии или вебсайта, чтобы мы могли применить санкции.

Пожалуйста, свяжитесь с нами по адресу электронной почты **dmkpress@gmail.com** со ссылкой на подозрительные материалы.

Мы высоко ценим любую помощь по защите наших авторов, помогающую нам предоставлять вам качественные материалы.

Часть I

ОСНОВЫ ПРОГРАММИРОВАНИЯ ВСТРОЕННЫХ СИСТЕМ И РОЛЬ C++

В этой части читатель познакомится с разнобразными существующими встроенными аппаратными платформами, а также с практическим примером проекта простой встроенной системы.

Часть I включает следующие главы:

- глава 1 «Что такое встроенные системы»;
- глава 2 «C++ как язык программирования встроенных систем»;
- глава 3 «Разработка для встроенной ОС Linux и подобных систем»;
- глава 4 «Встроенные системы с ограниченными ресурсами»;
- глава 5 «Пример: монитор влажности почвы с использованием протокола Wi-Fi».

Глава 1

Что такое встроенные системы

По существу слово «встроенная» в термине «встроенная система» обозначает состояние включения (встраивания, интеграции) такой системы в более крупную систему. Встраиваемая система – это компьютерная система определенного вида, которая обладает одной или несколькими специализированными функциями в объемлющей системе, в отличие от компонентов общего назначения. Более крупная объемлющая система по своей природе может быть цифровой, механической или аналоговой, в то время как дополнительная интегрированная цифровая схема взаимодействует непосредственно с данными, передаваемыми и получаемыми от интерфейсов, сенсоров и устройств памяти, для реализации истинной функциональности системы.

В этой главе рассматриваются следующие темы:

- различные категории встраиваемых платформ;
- примеры встраиваемых платформ каждой категории;
- особенности и трудности разработки для каждой категории.

Разнообразие встраиваемых систем

Любая компьютеризованная функция в современных устройствах реализована с использованием одного или нескольких микропроцессоров. Это означает, что процессор (CPU – central processing unit) обычно содержит одну интегральную микросхему (IC – integrated circuit). Микропроцессор состоит, как минимум, из арифметико-логического устройства (АЛУ) и управляющей схемы, но если рассуждать логически, то необходимы также регистры и блоки ввода/вывода (I/O), а кроме того, более развитые функциональные возможности, специально предназначенные для определенной категории продукции (носимые устройства, низкочастотные (слаботочные) сенсоры, микшеры сигналов и т. п.) или ориентированные на широкий потребительский рынок (бытовая электроника, медицина, автомобили и т. п.).

В настоящее время во встроенных системах можно обнаружить почти все типы микропроцессоров. Даже у людей, которые предпочитают пользоваться настольным компьютером, ноутбуком, смартфоном или планшетом, количество встроенных микропроцессоров в домашнем хозяйстве значительно превышает количество микропроцессоров общего назначения.

Внутри ноутбука или настольного компьютера имеется несколько встроенных микропроцессоров, дополняющих центральный процессор общего назначения. Эти микропроцессоры выполняют такие задачи, как обработка ввода с клавиатуры и мыши или ввода с сенсорных экранов, преобразование потоков данных в пакеты Ethernet или формирование выходных потоков видео либо аудиоданных.

Даже в более старых системах, например в компьютере Commodore 64, можно обнаружить точно такие же компоненты: микросхема ЦПУ, звуковая микросхема, микросхема видео и т. д. Центральный процессор выполняет любой код, написанный разработчиком приложения, тогда как другие микросхемы в этой системе предназначены для весьма специализированных конкретных целей, вплоть до микросхемы контроллера, управляющего приводами жесткого или гибкого диска.

Встроенные микропроцессоры можно найти не только в компьютерах общего назначения, но практически везде, зачастую в виде даже еще более интегрированных микроконтроллеров (MCU – microcontroller unit). Они управляют кухонными устройствами, стиральными машинами, двигателями автомобилей, дополняя функции более высокого уровня и обработку информации, получаемой от сенсоров.

Первые микроволновые печи были аналоговыми устройствами с механическими таймерами и переменными резисторами (реостатами) для установки мощности и продолжительности, но в современных микроволновых печах содержится по меньшей мере один микроконтроллер, отвечающий за обработку пользовательского ввода, управление дисплеем определенного типа и конфигурирование внутренних систем микроволновой печи. Дисплей может иметь собственный микроконтроллер в зависимости от сложности общей конфигурации.

Еще более интересен тот факт, что встроенные системы также обеспечивают контроль, автоматизированное управление и безопасность самолетов, гарантируют, что управляемые снаряды и космические ракеты будут работать, как предполагалось, и предоставляют постоянно расширяющиеся возможности в таких областях, как медицина и робототехника. Авиационная электроника любого самолета постоянно отслеживает множество параметров, получаемых от огромного количества сенсоров, выполняя код в конфигурации с тройной избыточностью, чтобы своевременно выявлять любые возможные сбои и проблемы.

Крошечные, но мощные микропроцессоры обеспечивают быстрый анализ химических соединений, нитей ДНК и РНК, а раньше для этого требовалось громоздкое лабораторное оборудование. При постоянно прогрессирующих технологиях встроенные системы настолько уменьшились в размерах, что появилась возможность их ввода в организм человека для наблюдения за его здоровьем.

Не только на Земле, но и на Луне, на Марсе и на астероидах космические зонды и вездеходы ежедневно выполняют бесчисленное множество задач с помощью многократно проверенных и испытанных встроенных систем. Исследования Луны стали возможными благодаря первому экземпляру встроенной системы в форме компьютерной системы Apollo Guidance Computer. В 1966 году была создана встроенная система, состоящая из соединенных проводами печатных плат и заполненная логическими вентилями НЕ с тройными входами, специально предназначенная для обработки навигационной информации, управления и контроля командным модулем и лунным модулем, носителем для которых стали ракеты Saturn V.

Широкое распространение и универсальная природа встроенных систем сделали их неотъемлемой частью современной жизни.

Встроенные системы обычно классифицируют по следующим категориям:

- микроконтроллеры (MCU);
- система на кристалле (System-on-Chip – SoC), которую часто называют одноплатным компьютером (Single-Board Computer – SBC).

Микроконтроллеры

Одним из решающих факторов при инновациях в области встроенных систем является стоимость, поскольку встроенные системы чаще всего должны представлять собой широко распространенную дешевую потребительскую продукцию. Такой подход помогает получить полноценный микропроцессор, память, устройство хранения данных и периферийные устройства ввода/вывода в одной микросхеме, упрощая реализацию, исключая необходимость печатной платы, но при этом добавляя преимущества более производительного и простого проектного решения и высокоэффективного производства. В 1970-е годы это привело к разработке микроконтроллеров (MCU): компьютерных систем на одной микросхеме, которые можно добавлять в новое проектное решение с минимальными затратами.

С внедрением энергонезависимого электрически стираемого перепрограммируемого ПЗУ (EEPROM) в технологию производства микроконтроллеров в начале 1990-х годов сначала появилась возможность многократной перезаписи программной памяти микроконтроллеров без операции стирания содержимого памяти с помощью ультрафиолетового луча, направляемого через специальное кварцевое окно в корпусе микроконтроллера. Это позволило существенно упростить прототипирование и в дальнейшем снизить стоимость, а также разработать и внедрить внутрисхемное программирование.

В результате многие системы, которые ранее управлялись сложными механическими и аналоговыми устройствами (например, лифты и регуляторы температуры), были оснащены одним или несколькими микроконтроллерами, которые обеспечивали ту же функциональность при более низкой цене и более высокой надежности. Используя возможности, управляемые программно, разработчики стали беспрепятственно добавлять расширенные функции, такие как сложные предварительно устанавливаемые программы (для стиральных машин, микроволновых печей и т. п.) и упрощенное управление комплексными дисплеями для обеспечения обратной связи с пользователем.

TMS 1000

Первым коммерчески распространяемым микроконтроллером был TMS 1000 компании Texas Instruments, универсальная 4-битовая система на кристалле. Этот микроконтроллер впервые поступил в широкую продажу в 1974 году. Первый вариант модели имел 1 Кб ПЗУ (ROM), 64×4 бита ОЗУ (RAM) и 23 контакта ввода/вывода. Частота варьировалась от 100 до 400 КГц, при этом каждая инструкция выполнялась за шесть циклов таймера.

В более поздних моделях была возможность увеличения размеров ПЗУ и ОЗУ, но общее проектное решение в основном оставалось неизменным вплоть до прекращения производства этого микроконтроллера в 1981 году.

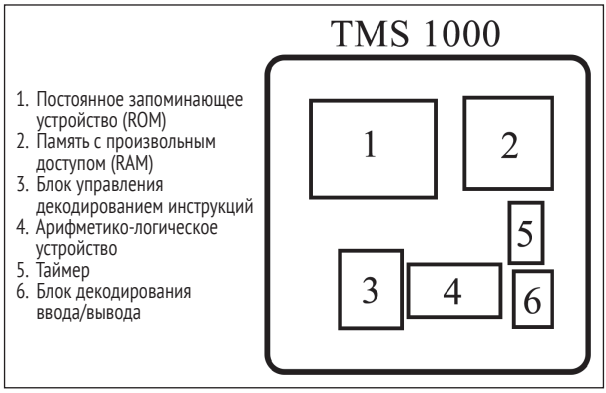

Рис. 1.1

Размер пластины этого микроконтроллера составлял приблизительно 5×5 мм, то есть был достаточно компактным для упаковки в корпус типа DIP (dual in-line package). В этом типе микроконтроллера использовалось маскируемое (программируемое при помощи маски) ПЗУ, то есть вы не могли получить «пустой» чип TMS 1000 и самостоятельно запрограммировать его. Вместо этого вы должны были передать отлаженную и адаптированную программу в компанию Texas Instruments для физического производства микросхемы с использованием фотолитографической маски, позволяющей создать металлический связующий мостик для каждого бита.

Это было достаточно примитивное проектное решение (по сравнению с более поздними микроконтроллерами), в котором отсутствовал стек и возможность обработки прерываний. Микроконтроллер работал с набором из 43 инструкций и имел два регистра общего назначения, что придавало ему некоторую схожесть с центральным процессорным устройством Intel 4004. Некоторые модели оснащались специализированными периферийными устройствами для управления вакуумно-люминесцентными (катодолюминесцентными) индикаторами (vacuum fluorescent displays – VFD) и для непрерывного считывания входных данных для обработки пользовательского ввода с клавиатуры без прерывания основной программы. Основная схема расположения контактов показана на рис. 1.2.

По рис. 1.2 становится понятно, что функции контактов являлись предшественниками функций контактов интерфейса ввода/вывода общего назначения (GPIO), хорошо известного нам сегодня, – контакты K могут использоваться только для ввода, в то время как контакты для вывода обозначены буквой O, а управляющие контакты помечены буквой R. Контакты OSC предназначены для соединения со схемой внешнего осциллографа. Как и в большинстве интегральных схем дискретной логики, контакт Init используется для инициализации микросхемы при подключении электроэнергии и должен сохранять высокую скорость выполнения, не более шести циклов, в то время как в более современные модели мик-

роконтроллеров интегрированы автоматические устройства реинициализации (power-on reset – POR), поэтому соответствующий контакт требует лишь наличия дискретного резистора или конденсатора.

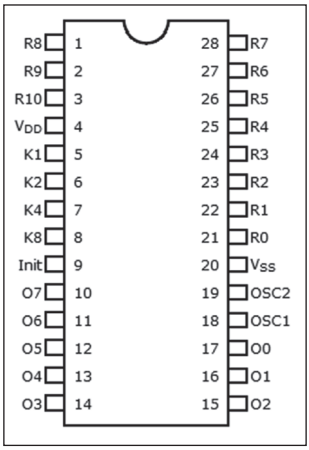

Рис. 1.2

В пресс-релизе компании Texas Instruments, выпущенном в 1974 году, указано, что этот микроконтроллер можно приобрести за 3 доллара, а при покупке крупной партии цена снижается. Предполагалось их использование не только в популярных электронных игрушках, таких как Speak and Spell, но почти везде, в том числе в бытовых приборах, автомобилях и научном оборудовании. До того, как производство TMS 1000 было прекращено в начале 1980-х годов, были проданы миллионы этих микроконтроллеров.

Также следует отметить, что стоимость однократно программируемых дешевых микроконтроллеров существенно снизилась, но этот тип продукции остается востребованным на рынке, например микроконтроллер Padauk PM150C сейчас можно приобрести за 0,03 долл., и хотя он предлагает 8-битовую архитектуру, 1 Кб слов ПЗУ и 64 байта ОЗУ кажутся подозрительно знакомыми.

Intel MCS-48

Ответом компании Intel на успешный микроконтроллер TMS 1000 компании Texas Instruments стала серия MCS-48, первые модели которой, 8048, 8035 и 8748, были выпущены в 1976 году. Модель 8048 имела 1 Кб ПЗУ и 64 байта ОЗУ. Это 8-битовое проектное решение с гарвардской архитектурой (раздельная память для кода/данных), в которой был введен внутренний 8-битовый размер слова и поддержка прерываний (на двух отдельных уровнях). Также обеспечивалась совместимость с микросхемами управления периферией 8080/8085, таким образом, MCS-48 представлял собой весьма универсальное семейство микроконтроллеров. Преимущество более развитых функций АЛУ и размера регистровых слов остается весьма заметным и по сей день, когда, например, 32-битовое расширение последовательно выполняется на 8-битовом микроконтроллере как группа 8-битовых расширений с соблюдением осторожности.

Для серии MCS-48 определено более 96 инструкций, большинство которых работают с данными длиной в один байт, а также допускается добавление расширенной памяти к внутреннему блоку ОЗУ. Благодаря усилиям сообщества вся доступная информация о семействе микроконтроллеров MCS-48 была собрана и опубликована на сайте **https://devsaurus.github.io/mcs-48/mcs-48.pdf**.

Здесь мы рассматриваем упрощенную функциональную блок-схему семейства микроконтроллеров MCS-48 и сравниваем ее с последующими семействами моделей.

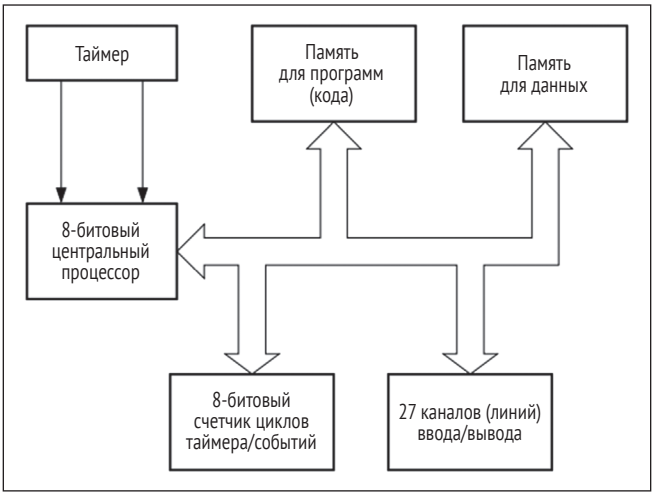

Рис. 1.3

Несмотря на то что это проектное решение было реализовано всего лишь несколькими годами позже TMS 1000, быстрое развитие архитектуры микроконтроллеров очевидно. Поскольку процесс проектирования микроконтроллеров происходил параллельно с проектированием широко распространенных в то время центральных процессоров, включая 16-битовую версию 6502, которая в конечном итоге стала основой семейства процессоров M68K, то в проектных решениях обнаруживается много похожих характеристик.

Благодаря гибкому и универсальному проектному решению это семейство микроконтроллеров оставалось широко распространенным и производилось до 1990-х годов, пока серия MCS-51 (8051) постепенно не заменила его. В следующем разделе микроконтроллер 8051 рассматривается более подробно.

Микроконтроллер MCS-48 использовался как контроллер клавиатуры в самой первой модели IBM PC. Он также применялся вместе с процессорами 80286 и 80386 как вентиль линии A20 и для выполнения функций рестарта для процессора 80286. В более поздних моделях PC эти функции были интегрированы в устройства Super I/O.

Еще один заслуживающий внимания вариант использования MCS-48 – игровая видеоконсоль Magnavox Odyssey и ряд моделей аналоговых синтезаторов Korg и Roland. Маскируемое ПЗУ (размером до 2 КБ) являлось дополнительной опцией для семейства MCS-48, но в микросхеме 87P50 уже использовался внешний

модуль ПЗУ для ее программирования, а микросхемы 8748 и 8749 имели до 2 Кб перезаписываемого ПЗУ (EPROM – Erasable Programmable Read Only Memory), позволяющего многократно перепрограммировать внутреннее содержимое микроконтроллера.

Как и для независимых модулей EPROM, для микроконтроллеров требовался специальный корпус со встроенным кварцевым окном, позволяющим ультрафиолетовому лучу света воздействовать на пластину микросхемы. Это кварцевое окно отчетливо видно на рис. 1.4 на фотографии микроконтроллера 8749 с модулем EPROM (автор фото Константин Ланзет (Konstantin Lanzet), лицензия CC BY-SA 3.0).

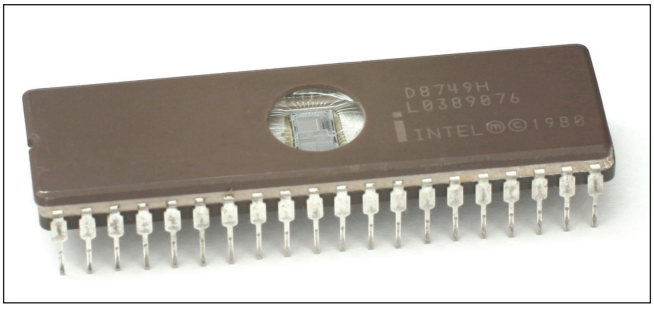

Рис. 1.4

Заряд ячеек EPROM, определяющий запись битов, уничтожается за 20–30 минут воздействия сильного ультрафиолетового излучения. Тот же эффект достигается при прямом воздействии солнечного света в течение нескольких недель. Цикл стирания обычно предполагает удаление корпуса и воздействие света непосредственно на стираемое устройство. После этого EPROM можно перепрограммировать заново. Спецификацией определен срок хранения данных в EPROM: приблизительно 10–20 лет при температуре около 85 °C, но поскольку срок хранения уменьшается экспоненциально при росте температуры, то при комнатной температуре можно рассчитывать на сохранность данных в течение 100 лет и более (для микросхемы 27C512A – 200 лет).

Из-за дополнительных затрат на создание кварцевого окна и вмонтирование его в корпус однократно программируемые EPROM использовались в течение некоторого времени, что позволяло легко программировать EPROM, но при помещении программируемой пластины микросхемы в непрозрачный корпус возможность перепрограммирования исчезла. В конце концов, в начале 1980-х годов стали доступными модули EEPROM (Electrically Erasable Programmable Read Only Memory), которые почти полностью заменили EPROM. Модули EEPROM можно перезаписывать около миллиона раз, прежде чем начнут возникать проблемы с сохранностью записанных в них данных. Надежность хранения данных почти ничем не отличается от модулей EPROM.

Intel MCS-51

Ряд недавно выпущенных микросхем – от Cypress CY7C68013A (контроллер периферийных USB-устройств) до Ti CC2541 (Bluetooth-система на кристалле) – содер-

жит ядра 8051, что свидетельствует о сохранении широкой распространенности архитектуры семейства Intel MCS-51 в наши дни. Существует огромное количество производных модификаций, в том числе и от других производителей, несмотря на то что компания Intel прекратила производство этой серии микроконтроллеров в марте 2007 года. Этот 8-битовый микроконтроллер, впервые появившийся в 1980-х годах, похож на серию 8048, но со значительным расширением набора функциональных возможностей.

Функциональная блок-схема, взятая из спецификации Intel 80xxAH, показана на рис. 1.5.

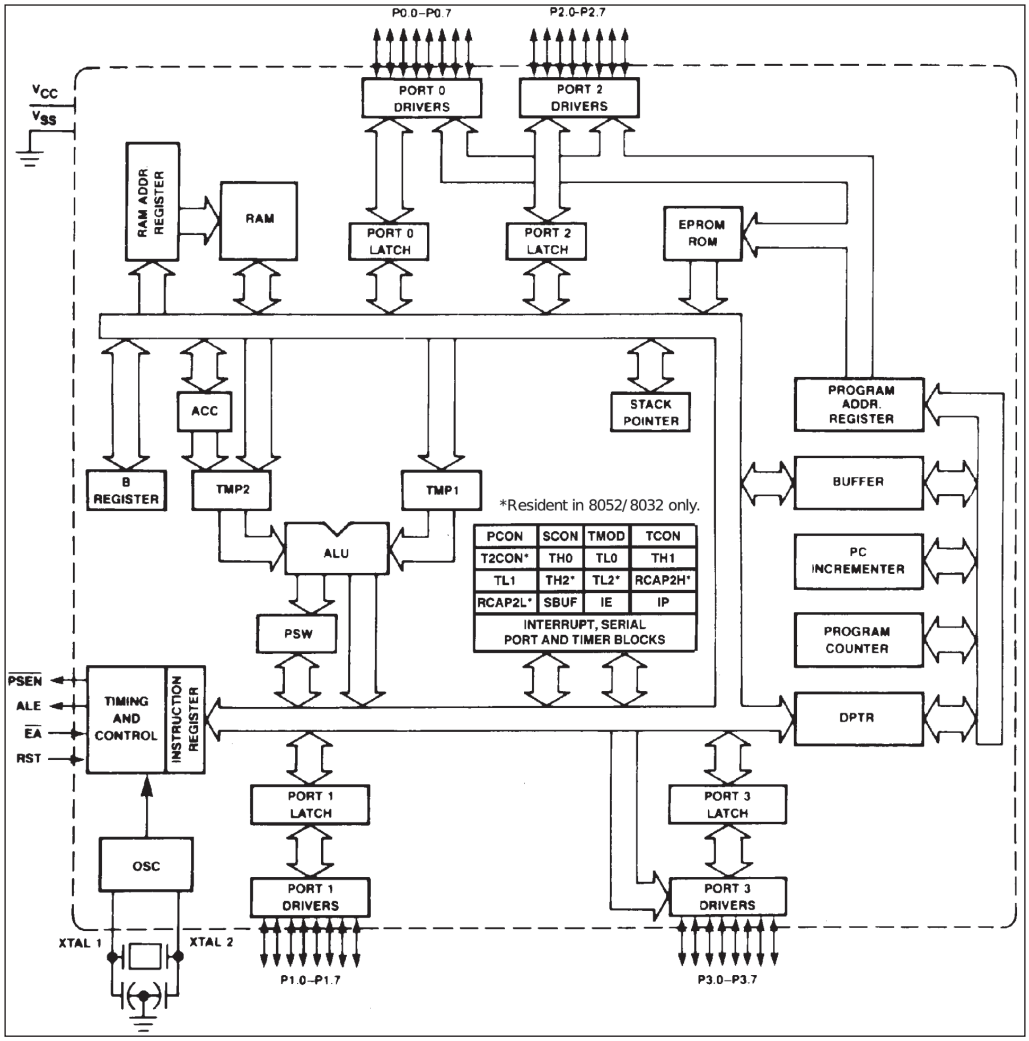

Рис. 1.5

Схема очень похожа на Atmel (сейчас это микрочип) AT89S51, который выпускается и в наши дни.

В спецификации определены общие рабочие характеристики в списке Features, часть которого, относящаяся к AT89S51, приведена ниже:

- 4 Кб программируемой в эксплуатируемой системе (in-system programmable – ISP) флеш-памяти – долговечность (выносливость): 10 000 циклов записи/стирания (ранее 1 000 000 для EEPROM);
- режим работы от 4,0 до 5,5 В;
- полностью статическая операция: от 0 Гц до 33 МГц (ранее 12 МГц);
- трехуровневая блокировка программной памяти;
- внутреннее ОЗУ (RAM) 128×8 бит;
- 32 программируемых канала (линии) ввода/вывода.

Затем в этом же списке перечисляются современные улучшенные характеристики: ядро, периферия, низкое энергопотребление и функциональные возможности, улучшающие удобство использования:

- два 16-битовых таймера/счетчика;
- шесть источников прерываний;
- полнодуплексный последовательный канал UART;
- режимы ожидания (Idle) и «сна» с пониженным энергопотреблением;
- прерывание для восстановления из режима «сна»;
- сторожевой таймер;
- двойной указатель на данные;
- флаг отключения электропитания;
- возможность быстрого программирования;
- гибкое программирование в режиме эксплуатации системы (ISP) в побайтовом или постраничном режиме.

За последние десятилетия в архитектуре 8051 значительным изменением стал только переход от первоначальной технологии транзисторов NMOS (n-type metal oxide semiconductor) к технологии CMOS (complementary MOS), обычно обозначаемый как 80C51, а также более позднее добавление интерфейсов USB, I2C и SPI и усовершенствованное управление энергопотреблением. Кроме того, были добавлены отладочные интерфейсы, которые стали вездесущими с начала XXI века. В примечаниях к спецификации Amtel 3487A не приводится даже краткое объяснение смысла буквы S, тем не менее ниже можно особо выделить новую функцию последовательного программирования микросхемы в эксплуатируемой системе (ISP).

Схема расположения контактов (ножек) микросхемы AT89S51 определяет назначение контактов SPI (Serial Peripherial Interface) (MOSI – Master Output Slave Input; MISO – Master Input Slave Output; SCK – Serial Clock), как показано на рис. 1.6.

Помимо автономных микроконтроллеров, ядра 8051 также включаются в состав более крупных систем, где простые микроконтроллеры с низким энергопотреблением предназначены для разнообразных задач, связанных с мониторингом операций ввода/вывода как с низкими, так и с высокими скоростями и даже в реальном времени. Широкий спектр микросхем от аналогов Ti CC2541 (Bluetooth-система на кристалле с низким энергопотреблением) до Cypress CY7C68013A (FX2LP™ контроллер периферийных USB-устройств) подчеркивает полезность и надежность архитектуры 8051 и в наши дни.

В процессе разработки программируемой пользователем вентильной матрицы (field-programmable gate array – FPGA) и интегральной схемы специального назна-

чения (application specific integration circuit – ASIC) процессоры типа 8051 широко использовались в качестве программируемых ядер. Этот тип микросхемы также адаптирован и добавлен в проекты VHDL и Verilog HDL для задач, которые проще решаются с помощью последовательного выполнения при отсутствии жестких требований к увеличенной пропускной способности (производительности) или ограничений по времени. Последнее, но не менее важное замечание: преимущество программируемых ядер заключается в их способности полноценного использования инструментальных средств разработки и отладки с обеспечением тесной интеграции при сохранении неизменности проектного решения аппаратной части. Всего лишь несколько сотен байтов программного кода, выполняемого программируемым ядром, могут успешно заменять крупные механизмы управления состояниями, большие блоки памяти, многочисленные счетчики и поддерживать логику уровня АЛУ. В результате возникает вопрос: какая реализация проще с точки зрения проверки и сопровождения?

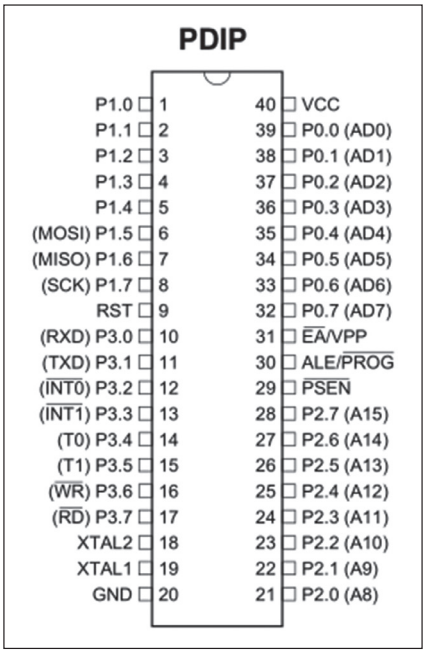

Рис. 1.6

PIC

Семейство микроконтроллеров PIC было выпущено в 1976 году компанией General Instrument, которая использовала для этих микроконтроллеров свой новый 16-битовый процессор CP1600. Этот центральный процессор был почти совместимым по набору инструкций с серией процессоров PDP-11 (компании Digital Equipment Corporation, DEC).

В 1987 году компания General Instrument расширила и укрепила свой отдел микроэлектроники, создав крупное подразделение Microchip Technology, которое

стало независимой компанией в 1989 году. Компания Microchip Technology производит новые модификации PIC и сейчас. Вместе с развитием ядер и периферии PIC разработка памяти, интегрированной непосредственно в микросхему, позволила начать внедрение слабосвязанной инкапсулированной памяти EPROM для однократно программируемого чипа, а в дальнейшем – EEPROM для обеспечения возможностей многократного перепрограммирования в период эксплуатации микроконтроллера. Как и большинство микроконтроллеров, PIC имеет гарвардскую архитектуру. В наше время выпускается линия микроконтроллеров PIC от 8-битовых до 32-битовых с широким спектром функциональных возможностей. Во время написания данной книги микроконтроллер PIC был представлен несколькими семействами, описанными в табл. 1.1.

Таблица 1.1

Семейство	Контакты	Память	Дополнительные характеристики
PIC10	6–8	384–896 байт ПЗУ, 64–512 байт ОЗУ	8-битовый, 8–16 МГц, модифицированная гарвардская архитектура
PIC12	8	2–16 Кб ПЗУ, 256 байт ОЗУ	8-битовый, 16 МГц, модифицированная гарвардская архитектура
PIC16	8–64	3,5–56 Кб ПЗУ, 1–4 Кб ОЗУ	8-битовый, модифицированная гарвардская архитектура
PIC17	40–68	4–16 Кб ПЗУ, 232–454 байта ОЗУ	8-битовый, 33 МГц, вытеснен моделью PIC18, хотя продолжают выпускаться клоны сторонних производителей
PIC18	28–100	16–128 Кб ПЗУ, 3728–4096 байт ОЗУ	8-битовый, модифицированная гарвардская архитектура
PIC24 (dsPIC)	14–144	64–1024 Кб ПЗУ, 8–16 Кб ОЗУ	16-битовый, в микроконтроллеры DsPIC (dsPIC33) встроены периферийные устройства для цифровой обработки сигналов
PIC32MX	64–100	32–512 Кб ПЗУ, 8–32 Кб ОЗУ	32-битовый, 200 МГц MIPS M4K с режимом MIPS16e, выпущен в 2007 году
PIC32MZ EC PIC32MZ EF PIC32MZ DA	64–288	512–2048 Кб ПЗУ, 256–640 Кб статического ОЗУ (32 Мб DDR2 DRAM)	32-битовые, MIPS ISA (2013), версия PIC32MZ DA (2017) содержит графическое ядро. Тактовые частоты ядер 200 МГц (ec, DA) и 252 МГц (EF)
PIC32MM	20–64	16–256 Кб ПЗУ, 4–32 Кб ОЗУ	32-битовый microMIPS, 25 МГц, вариант, оптимизированный по низкой цене и низкому энергопотреблению
PIC32MK	64–100	512–1024 Кб ПЗУ, 128–256 Кб ОЗУ	32-битовый, 120 МГц, MIPS ISA, вариант, созданный в 2017 году. Ориентирован на управление в производстве и другие формы глубоко интегрированных приложений

Семейство PIC32 интересно тем, что основано на ядре процессора MIPS и использует соответствующую архитектуру набора инструкций (Instruction Set Architecture – ISA) вместо PIC ISA, который применялся во всех прочих микроконтроллерах PIC. Используемая во всех микроконтроллерах этого семейства модель ядра процессора – M4K, 32-битовое ядро MIPS32 компании MIPS Technology. Различия между семействами PIC легко обнаружить, изучая блок-схемы из соответствующих спецификаций.

Вероятнее всего, историю развития линии микроконтроллеров PIC в течение нескольких десятилетий лучше всего изучать по функциональным блок-схемам, поэтому начнем с модели PIC10 (рис. 1.7).

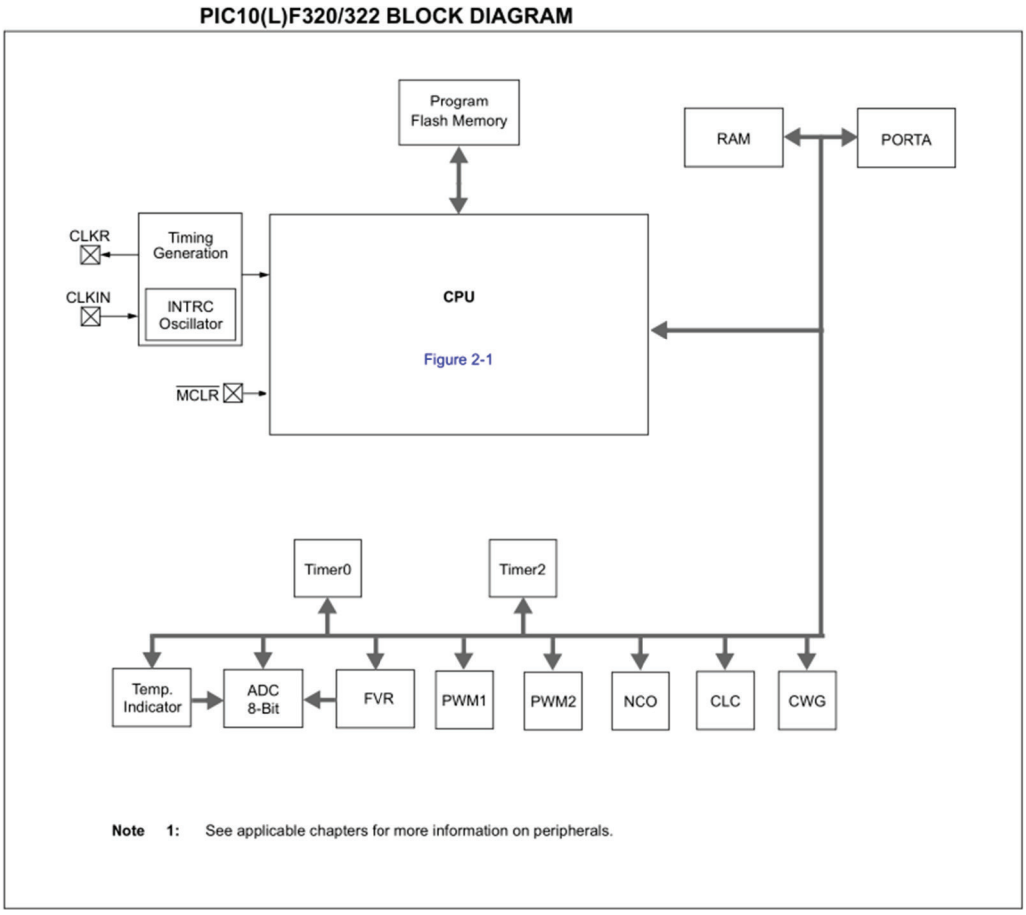

Рис. 1.7

Это очень маленькие микроконтроллеры с практически любыми периферийными устройствами, расположенными вокруг процессорного ядра, не описываются здесь более подробно, а в справочной таблице указаны только характеристики памяти. Количество портов ввода/вывода минимально, а интерфейсы I2C и UART, широко известные сегодня, не реализованы как периферийная логика. Для выбора следующего микроконтроллера этого семейства обратим внимание на то, что спецификация PIC16F84 весьма подробно описывает архитектуру процессора и сообщает, что добавлены схемы управления включением электропитания и перезагрузкой. Кроме того, расширены функции GPIO и добавлена память EEPROM для упрощения интеграции энергонезависимого устройства хранения данных. Последовательные периферийные устройства, размещенные непосредственно на пластине микроконтроллера, пока еще отсутствуют.

На рис. 1.8 показана функциональная блок-схема микроконтроллера PIC18.

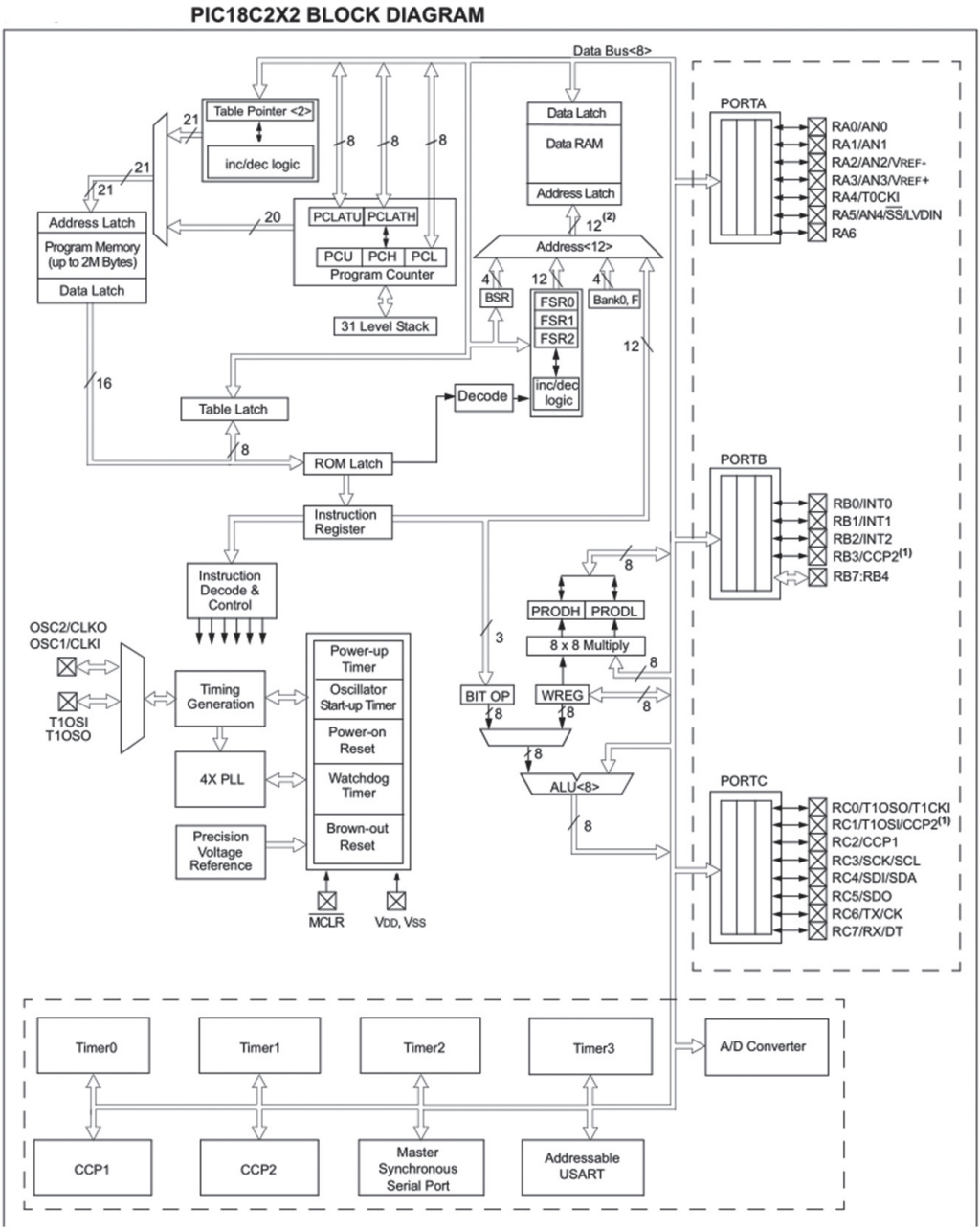

Рис. 1.8

Семейство PIC18 представляет самую последнюю версию 8-битовой архитектуры PIC, которая охватывает широкий диапазон приложений. Эта модель обеспе-

чивает значительно больше функций ввода/вывода по сравнению с семействами PIC10, PIC12 и PIC16, а кроме того, предлагает больше вариантов расширений ПЗУ и ОЗУ и содержит универсальный синхронный и асинхронный приемопередатчик (USART) в сочетании с синхронизированным последовательным портом для 4-проводного внешнего последовательного интерфейса (SPI). Также следует отметить, что теперь порты имеют изменяемые функции контактов, но маршруты от периферийных модулей к контактам и соответствующие конфигурационные регистры здесь не показаны для упрощения блок-схемы.

Теперь перейдем от ядра микроконтроллера к рассмотрению функциональных возможностей портов и периферийных модулей на функциональной блок-схеме PIC24 (рис. 1.9).

dsPIC33EPXXXGP50X, dsPIC33EPXXXMC20X/50X AND PIC24EPXXXGP/MC20X BLOCK DIAGRAM

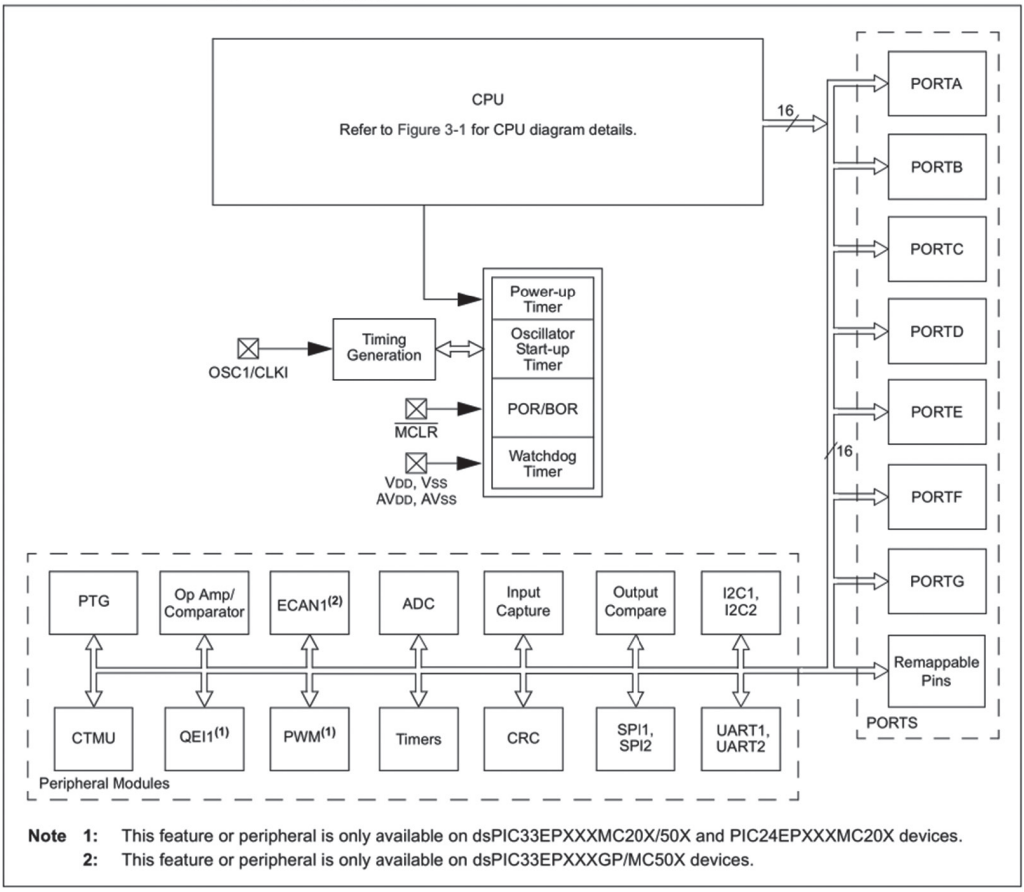

Рис. 1.9

Эта блок-схема похожа на схему для PIC10, здесь центральное процессорное устройство показано условно как единый блок по отношению к остальным компонентам микроконтроллера. Каждый из блоков портов имеет собственный на-

бор контактов ввода/вывода, но на этой блок-схеме невозможно показать все возможные функции контактов.

Каждый контакт ввода/вывода может иметь жестко закрепленную функцию (связанную с периферийным модулем) или переназначаемую функцию (изменение маршрута на аппаратном уровне или перепрограммирование). В общем случае чем более сложным является микроконтроллер, тем более вероятно, что его контакты ввода/вывода имеют обобщенные (изменяемые), а не жестко определенные функции.

Последней рассматриваемой здесь моделью является PIC32 (рис. 1.10).

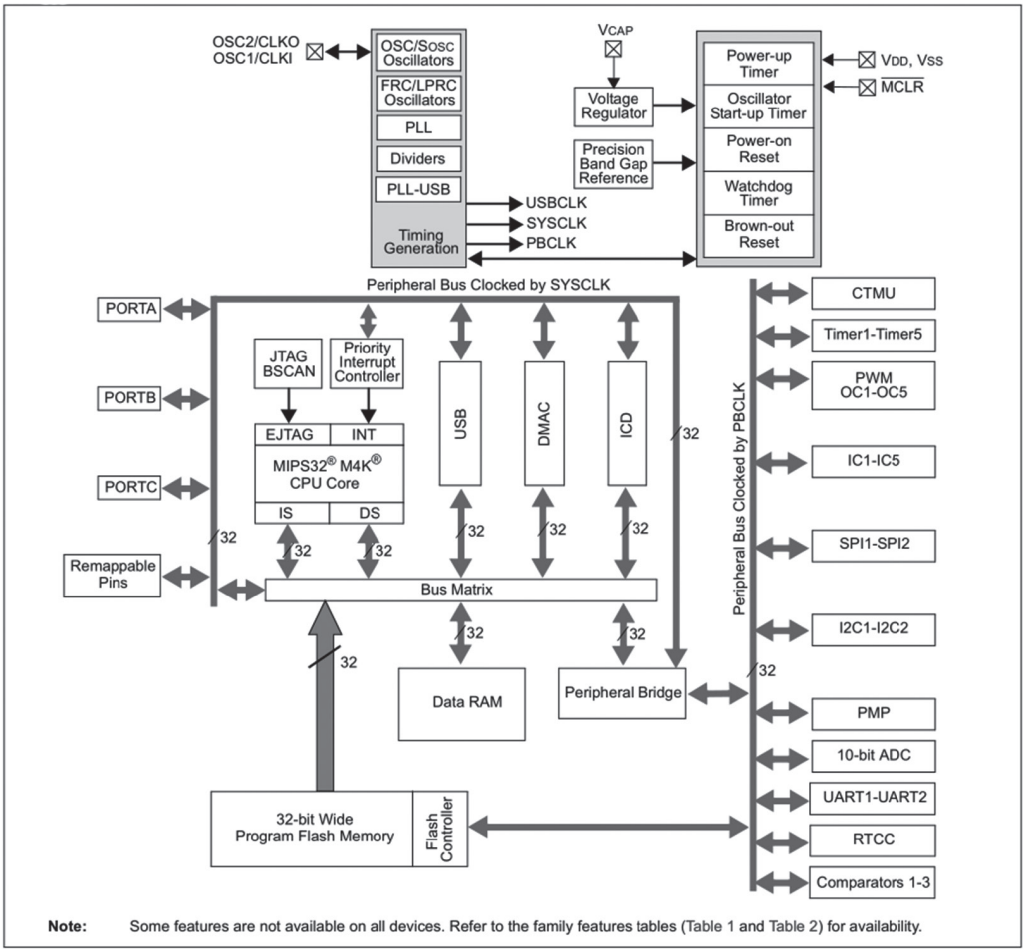

Рис. 1.10

На рис. 1.10 показана блок-схема для устройств PIC32MX1XX/2XX из семейства PIC32MX. Обычно эти устройства работают на тактовой частоте 50 МГц.

Интересным свойством архитектуры PIC32 является эффективное преобразование гарвардской архитектуры ЦПУ M4K MIPS в архитектуру, более похожую на

фон-неймановскую (John von Neumann), с передачей программных инструкций и данных по системной матрице шин (System Bus Matrix). Отметим, что блок, обозначающий единственный регистр процессора на блок-схеме PIC10, теперь абстрактно отображает сложный периферийный модуль, цифровой или со смешанным сигналом, или мощный стандартный аппаратный интерфейс тестирования и отладки (JTAG), размещенный непосредственно на панели микроконтроллера.

AVR

Архитектура AVR была разработана двумя студентами Норвежского технологического института, а самый первый микроконтроллер AVR был разработан в компании Nordic VLSI (теперь Nordic Semiconductor). Сначала микроконтроллер назывался µRISC и для обеспечения лицензионной защиты технологии был продан компании Atmel. Первый микроконтроллер Atmel AVR был выпущен в 1997 году.

В настоящее время можно вспомнить некоторые модели из многочисленных семейств 8-битовых микроконтроллеров AVR (табл. 1.2).

Таблица 1.2

Семейство	Контакты	Память	Дополнительные характеристики
ATtiny	6–32	0,5–16 Кб ПЗУ, 0–2 Кб ОЗУ	1,6–20 МГц, компактный микроконтроллер с низким энергопотреблением и ограниченными периферийными модулями
ATmega	32–100	4–256 Кб ПЗУ, 0,5–32 Кб ОЗУ	
ATxmega	44–100	16–384 Кб ПЗУ, 1–32 Кб ОЗУ	32 МГц, самый крупный микроконтроллер AVR с расширяемыми периферийными модулями и функциональными возможностями, улучшающими производительность, такими как DMA

Также использовалась 32-битовая архитектура AVR32, но не была принята компанией Atmel, так как компания перешла на другую 32-битовую архитектуру ARM (SAM). Немного больше об архитектуре SAM можно узнать из подраздела «Микроконтроллеры на основе ARM». Но самая подробная информация содержится в соответствующем руководстве «Product Selection Guide».

Кроме того, компания Atmel применяла так называемые контроллеры с программируемой пользователем на системном уровне интегральной микросхемой (Field Programmable System Level Integrated Circuit – FPSLIC): гибрид систем AVR/FPGA. По существу, такой вариант позволяет пользователю добавлять собственные периферийные модули и функциональность непосредственно в аппаратуру микроконтроллера AVR.

Рассмотрим подробнее семейство микроконтроллеров ATtiny. На рис. 1.11 показана блок-схема серии ATtiny212/412.

Эта серия микроконтроллеров ATtiny может работать на тактовых частотах до 20 Мгц, иметь до 4 Кб флеш-ПЗУ и 256 байт статической оперативной памяти (SRAM), а также до 128 байт EEPROM. Все это размещено в корпусе с 8 контактами. Несмотря на малый размер, микроконтроллер содержит множество периферий-

ных модулей, которые могут быть соединены с любым поддерживаемым контактом, как показано на рис. 1.12.

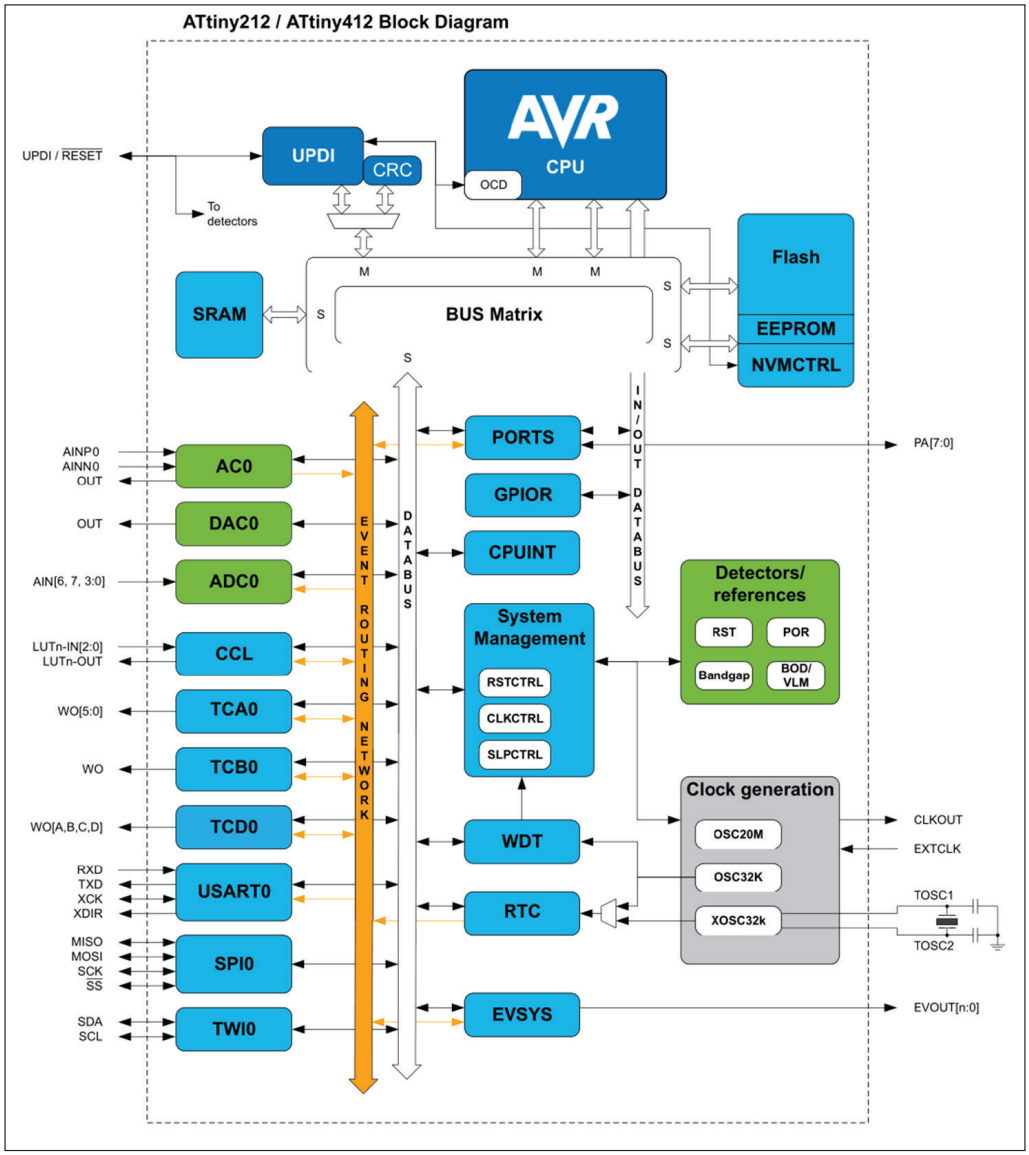

Рис. 1.11

Рис. 1.12

Для сравнения в табл. 1.3 приведены характеристики широко распространенного микроконтроллера ATmega2560 и некоторых других представителей этого семейства.

Таблица 1.3

Устройство	Флеш-память (Кб)	EEPROM (Кб)	ОЗУ (Кб)	Контакты ввода/вывода общего назначения	16-битовые каналы широтно-импульсной модуляции	UART	Каналы аналого-цифрового преобразования
ATmega640	64	4	8	86	12	4	16
ATmega1280	128	4	8	86	12	4	16
ATmega1281	128	4	8	54	6	2	8
ATmega2560	256	4	8	86	12	4	16
ATmega2561	256	4	8	54	6	2	8

В этом семействе микроконтроллеров насчитывается несколько десятков контактов интерфейса ввода/вывода общего назначения (GPIO), поэтому блок-схема существенно усложняется и содержит гораздо больше блоков портов для контактов ввода/вывода, как показано на рис. 1.13.

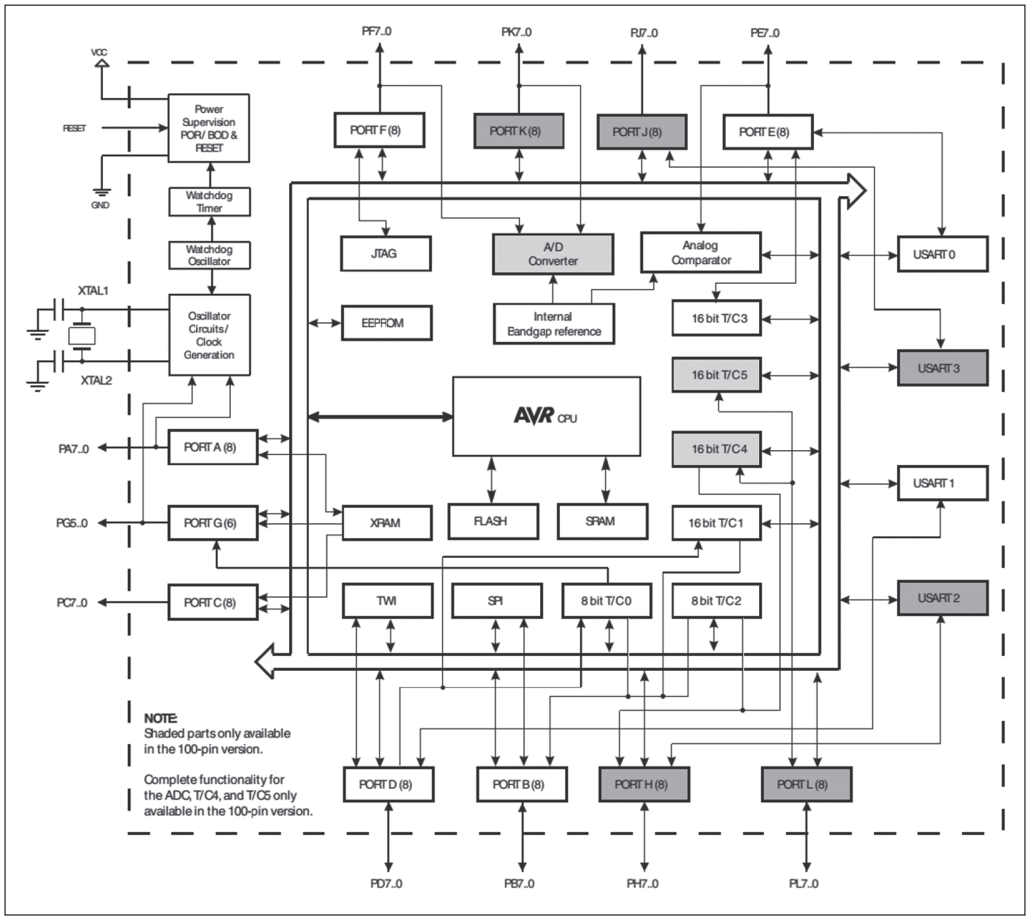

Рис. 1.13

На этой схеме все входящие и исходящие стрелки обозначают один контакт или блок контактов, большинство из которых имеет общее назначение. Из-за большого количества контактов больше нет смысла использовать на практике формат корпуса с расположением контактов в ряд (DIP, SOIC и проч.) для физической микросхемы.

Для моделей ATmega640, 1280 и 2560 используется корпус TQFP со 100 контактами, а функциональность каждого контакта описана в соответствующей спецификации (рис. 1.14).

Семейство ATxmega очень похоже на семейство ATmega с аналогичным расположением контактов, но различия в основном относятся к изменениям архитектуры и к дополнительной оптимизации. Кроме того, модели ATxmega оснащены ПЗУ и ОЗУ большего размера, а также более развитыми вариантами периферийных модулей.

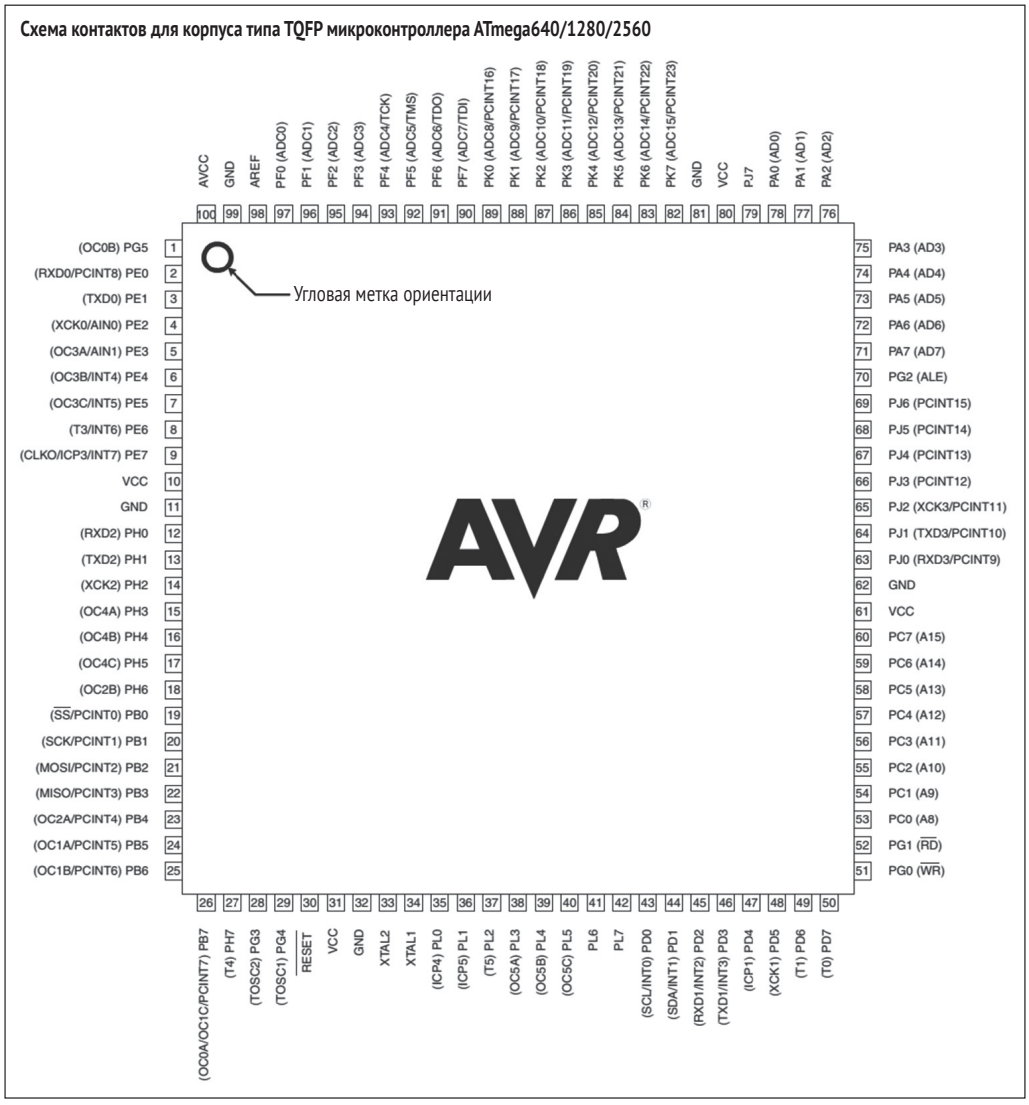

Рис. 1.14

Выбор микроконтроллера ATtiny, ATmega или ATxmega прежде всего зависит от требований конкретного проекта, в особенности от требований к вводу и выводу, от необходимых типов периферийных устройств (последовательные устройства, SPI, I2C, CAN и т. д.), а также от размера программного кода и от размера ОЗУ, требуемого для выполнения этого кода.

M68k и микроконтроллеры на основе Z80

8-битовый процессор Zilog Z80 совместим с процессором Intel 8080, который конкурировал с другими микропроцессорами на протяжении 1980-х годов. Борьба шла за преобладание в домашних компьютерах и в игровых системах, в том числе в Nintendo Game Boy, Sega Master System, Sinclair ZX80/ZX81/Spectrum, MSX и Tandy TRS-80.

Компания Zilog представила микроконтроллер Z380 на основе микропроцессора Z80 в 1994 году и в течение нескольких лет обновляла модели, в том числе Z8, eZ80 и др. Кроме того, широко распространены клоны Z80.

Еще одним известным в 1980-е годы микропроцессором являлась модель Motorola 68k (или 68000). АЛУ и внешняя шина данных этого микропроцессора 16-битовые, но внутренняя шина данных и регистры 32-битовые. После появления M68k в 1979 году его архитектура не изменялась и используется до настоящего времени – компания Freescale Semiconductor (сейчас NXP) продолжает производство большого количества микропроцессоров 68k.

Компания Motorola создавала множество вариантов микроконтроллеров на основе архитектуры 68k, в том числе контроллер коммуникационной связи (обмена данными) MC68320 в 1989 году. В настоящее время к проектным решениям микроконтроллеров на основе 68k относится модель ColdFire, полностью 32-битовая микросхема.

ARM Cortex-M

Весьма распространенным типом 32-битовых микроконтроллеров является семейство ARM Cortex-M. К нему относятся модели M0, M0+, M1, M3, M4, M7, M23 и M33, большинство из которых предоставляет возможность использования сопроцессора операций с плавающей точкой (floating point unit – FPU) для улучшения производительности математических операций.

Это семейство используется не только в качестве автономных микроконтроллеров, но часто модели интегрируются в устройства типа «система на кристалле» (System-on-Chip – SoC) для обеспечения особой функциональности, например для управления сенсорным экраном, датчиками-сенсорами или энергопотреблением. Поскольку компания Arm Holdings сама не производит различные варианты и модификации микроконтроллеров, многие сторонние производители приобрели лицензии на эти проектные решения и иногда вносят собственные изменения и усовершенствования.

В табл. 1.4 приведены некоторые характеристики микроконтроллеров семейства ARM Cortex-M.

Таблица 1.4

Ядро	Год анонсирования	Архитектура	Набор инструкций
M0	2009	Armv6-M	Thumb-1, частично Thumb-2
M0+	2012	Armv6-M	Thumb-1, частично Thumb-2
M1	2007	Armv6-M	Thumb-1, частично Thumb-2
M3	2004	Armv7-M	Thumb-1, Thumb-2
M4	2010	Armv7-M	Thumb-1, Thumb-2, дополнительно FPU
M7	2014	Armv7E-M	Thumb-1, Thumb-2, дополнительно FPU
M23	2016	Armv8-M	Thumb-1, частично Thumb-2
M33	2016	Armv8-M	Thumb-1, Thumb-2, дополнительно FPU

Thumb – это компактный набор 16-битовых инструкций, практически идеальный для встроенных систем с ограниченными ресурсами. Другие семейства микропроцессоров ARM также поддерживают набор инструкций Thumb в дополнение к основному набору 32-битовых инструкций.

H8 (SuperH)

Семейство микроконтроллеров H8 широко использовалось в 8-, 16- и 32-битовых вариантах. Это семейство в начале 1990-х годов начала выпускать компания Hitachi, а новые проектные решения продолжала создавать компания Renesas Technology до недавнего времени, хотя в последние годы рекомендуется использовать новейшие решения – семейства RX (32-битовое) и RL78 (16-битовое). Заслуживает внимания использование микроконтроллера H8 в контроллере Lego Mindstorms RCX, где применяется модель H8/300.

ESP8266/ESP32

ESP – это семейство 32-битовых микроконтроллеров, выпускаемых компанией Espressif Systems, в которые включена функциональная поддержка протоколов Wi-Fi (в обе модели) и Bluetooth (ESP32).

Микроконтроллер ESP8266 появился в 2014 году и сразу был продан стороннему производителю – компании Ai-Thinker в виде модуля (ESP-01), который мог использоваться другим микроконтроллером или системой на основе микропроцессоров для обеспечения функциональной поддержки протокола Wi-Fi. Модуль ESP-01 содержал встроенное программное обеспечение для этой цели, которое позволяло работать с модулем, применяя команды в стиле модемов семейства Hayes.

Ниже приведены некоторые системные характеристики модуля ESP-01:

- 32-битовый микропроцессор Tensilica Xtensa Diamond Standard L106;
- тактовая частота ЦПУ 80–160 МГц;
- до 50 Кб ОЗУ, доступного пользовательским приложениям (с загруженным стеком Wi-Fi);
- ПЗУ внешнего последовательного интерфейса SPI ROM (от 512 Кб до 16 Мб);
- поддержка Wi-Fi для 802.11 b/g/n.

Поскольку 32-битовый микроконтроллер на модуле ESP-01 обладал гораздо большей функциональностью, чем простейший модем, вскоре он стал использоваться для более общих задач, тем более что появились обновленные модули ESP8266 (со встроенной микросхемой EEPROM), а также переходные платы (адаптеры). Не так давно тип адаптера NodeMCU стал весьма распространенным, несмотря на то что другие производители создали свои версии переходных плат в разнообразных форм-факторах и с различной функциональностью.

На рис. 1.15 показана упрощенная блок-схема микроконтроллера ESP8266EX.

После невероятно успешной модели ESP8266 компания Espressif Systems разработала модель ESP32, в которой использовался обновленный двухъядерный центральный процессор, а также были внесены другие изменения. На рис. 1.16 показана блок-схема ESP32.

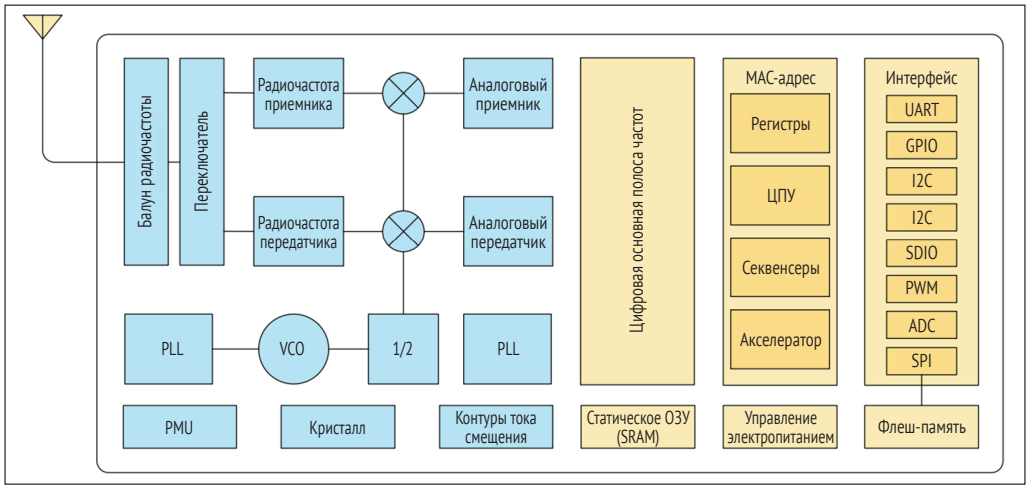

Рис. 1.15

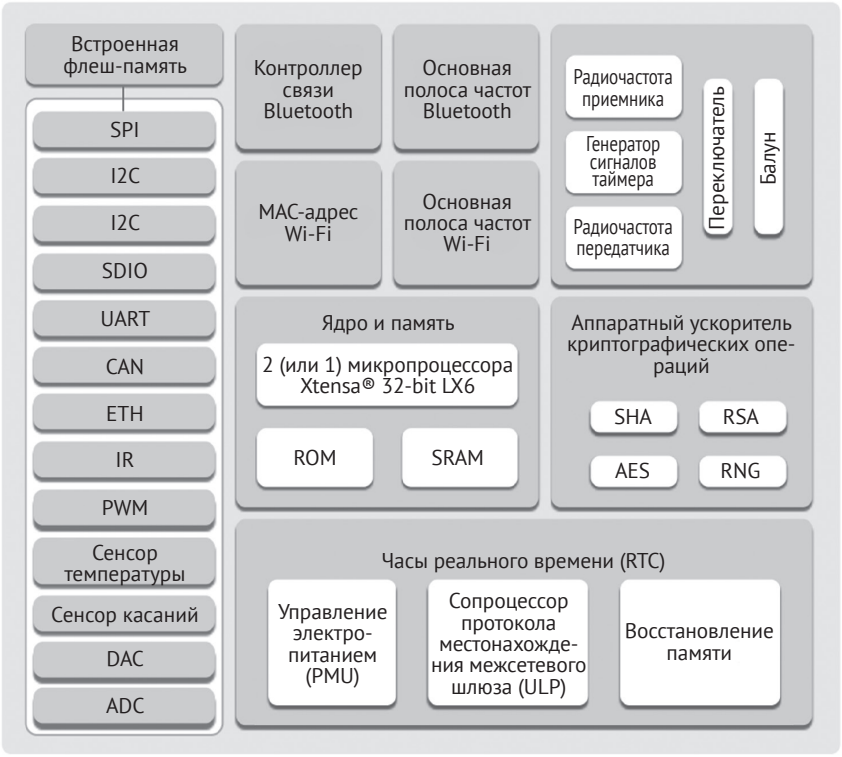

Рис. 1.16

Ниже перечислены некоторые технические характеристики ESP32:

- ❍ микропроцессор Xtensa 32-bit LX6 (двухъядерный);
- ❍ тактовая частота ЦПУ 160–240 МГц;

- 520 Кб статического ОЗУ (SRAM);
- поддержка Wi-Fi для 802.11 b/g/n;
- поддержка Bluetooth v4.2 и BLE (с низким энергопотреблением).

Обе модели ESP8266 и ESP32 массово продаются как полнофункциональные модули с микроконтроллером, модулем внешней памяти ROM и антенной Wi-Fi, либо смонтированной непосредственно на плате, либо во внешнем исполнении (рис. 1.17).

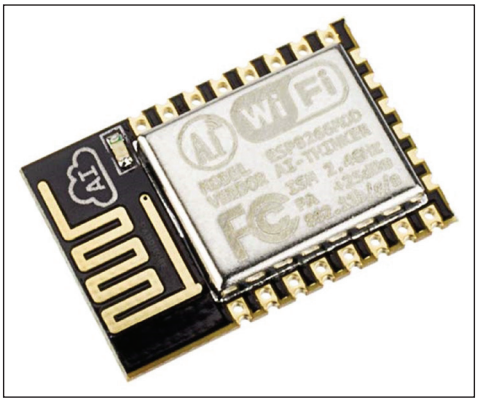

Рис. 1.17

Возможен вариант с металлическим корпусом, защищающим плату от электромагнитного воздействия и улучшающим характеристики трансивера Wi-Fi (и Bluetooth в модели ESP32), но в целом такое проектное решение со смонтированной на плате антенной и жестко определенной геометрией требует сертификации FCC и последующего использования как апробированный и санкционированный модуль. Несанкционированное подключение внешней антенны может нарушать местные законы. Поэтому необходимо получение идентификатора FCC ID, для того чтобы устройство, содержащее такой модуль, можно было применять в коммерческих целях.

Другие микроконтроллеры

Кроме описанных выше микроконтроллеров, существует множество микроконтроллеров с другими архитектурами от различных производителей. Некоторые из них, например Propeller компании Parallax, имеют многоядерную архитектуру и в определенной степени уникальны, тогда как в большинстве микроконтроллеров реализована обычная архитектура с одноядерным ЦПУ, набором периферийных модулей, ОЗУ и внутренним или внешним ПЗУ.

Кроме физических микросхем компании Altera (сейчас Intel), Lattice Semiconductor и Xilinx предлагают так называемые программные ядра, представляющие собой микроконтроллеры, работающие на программируемой пользователем вентильной матрице (FPGA), либо как независимые компоненты, либо как часть более крупного проектного решения на этой матрице FPGA. Для таких решений также существуют соответствующие компиляторы C/C++.

Особенности

Основные особенности и затруднения разработки для микроконтроллеров заключаются в относительно ограниченных доступных ресурсах. Особенно это касается небольших микроконтроллеров с малым количеством внешних контактов (ножек), для которых необходимо принять правильное проектное решение о том, сколько ресурсов (циклов ЦПУ, ОЗУ и ПЗУ) потребует конкретный блок кода и насколько реальным является добавление некоторой особенной функциональной возможности.

Кроме того, это означает, что при выборе правильного микроконтроллера для конкретного проекта нужны не только технические знания, но и практический опыт. Теоретические знания требуются для выбора микроконтроллера, полностью соответствующего задаче. Практический опыт очень полезен для подбора оптимальной модели (варианта) микроконтроллера, к тому же помогает сэкономить время на этапе выбора.

Одноплатный компьютер, или система на кристалле

Система на кристалле (System-on-Chip – SoC) похожа на микроконтроллер, но отличается от этих типов встроенных систем определенным уровнем интеграции при сохранении требований к наличию некоторого количества внешних компонентов для обеспечения функциональности. Системы на кристалле широко используются как часть реализации одноплатного компьютера (Single Board Computer – SBC), в том числе по стандарту PC/104, а также для более современных форм-факторов, таких как Paspberry Pi и производных от этой модели плат (см. рис. 1.18).

Схема на рис. 1.18 взята с сайта **https://xdevs.com/article/rpi3_oc/**. На ней четко показана компоновка одноплатного компьютера, в данном случае Raspberry Pi 3. Микросхема BCM2837 – это система на кристалле на основе ARM, предоставляющая ядро ЦПУ и основные периферийные модули (в основном описанные в соответствующих отдельных разделах заголовков). Вся оперативная память (ОЗУ) представляет собой внешний модуль, как и периферийные модули Ethernet и Wi-Fi. ПЗУ представлено в виде твердотельной (флеш-) карты, которая также предоставляет возможность хранения данных.

Большинство систем на кристалле создаются на основе микросхем ARM (семейство Cortex-A), хотя не менее часто применяются чипы MIPS. Системы на кристалле широко используются в промышленности.

Другими вариантами являются платы массового потребления, например для смартфонов, для которых не установлен предопределенный форм-фактор, тем не менее производители следуют некоторому шаблону системы на кристалле с учетом необходимости внешнего ПЗУ, ОЗУ и устройства хранения данных, а также разнообразных периферийных устройств. Такой подход противоположен проектным решениям микроконтроллеров, описанных в предыдущем разделе, которые всегда способны самостоятельно выполнять свои функции, за исключением разве что выполнения некоторых требований к внешним ПЗУ.

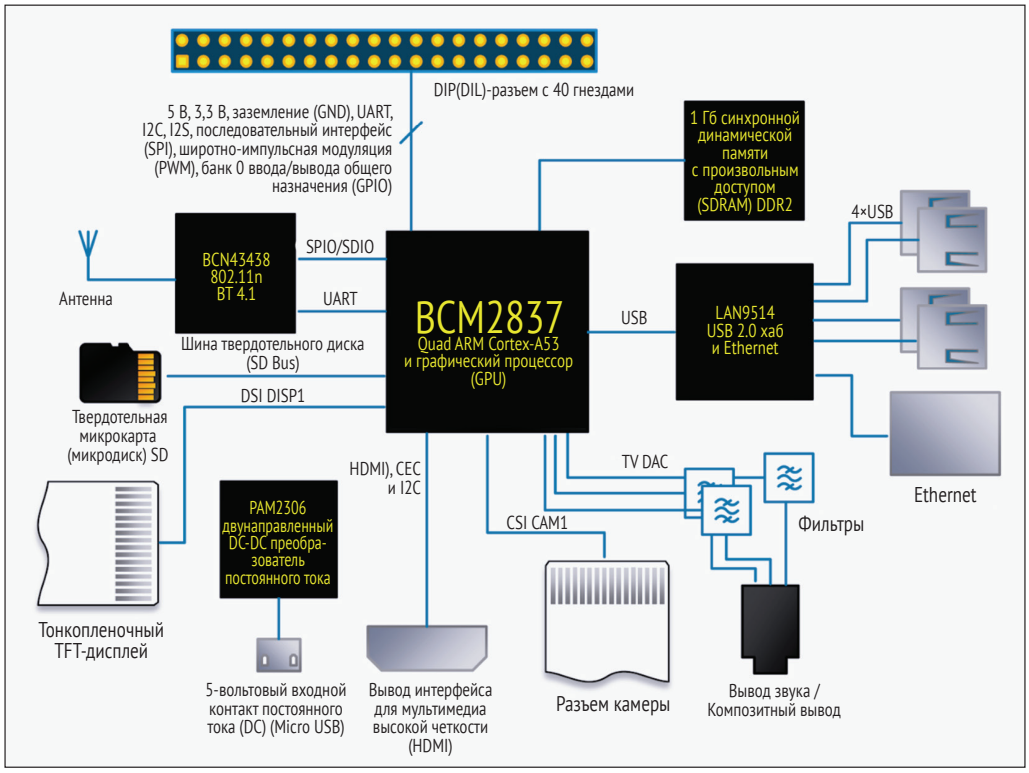

Рис. 1.18

Особенности

По сравнению с микроконтроллерами особенности и трудности разработки для систем на кристалле не столь многочисленны. Некоторые системы на кристалле предоставляют открытый интерфейс, который можно даже использовать для разработки непосредственно на устройстве, то есть с выполнением циклов компиляции на самом устройстве без кросс-компиляции на настольном компьютере и копирования (переноса) бинарных файлов. Кроме того, это позволяет установить и использовать полноценную операционную систему вместо разработки непосредственно на «голой» аппаратуре.

Очевидным недостатком является увеличение сложности при наращивании функциональных возможностей, в результате приводящее к дополнительным затруднениям, таким как обеспечение работы с учетными записями пользователей, установка прав доступа, управление драйверами устройств и т. п.

Резюме

В этой главе были подробно рассмотрены компоненты встроенных систем. Были наглядно продемонстрированы различия между разнообразными типами встроенных систем, а также определены основные принципы выбора правильного микроконтроллера или системы на кристалле для конкретного проекта.

После ознакомления с этой главой читатель более уверенно будет разбираться в спецификациях на микроконтроллеры и системы на кристалле, определять различия между ними и выбирать необходимые модели и варианты для своих проектов.

В следующей главе объясняется, почему язык C++ является наиболее подходящим вариантом выбора для программирования встроенных систем.