Содержание

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.

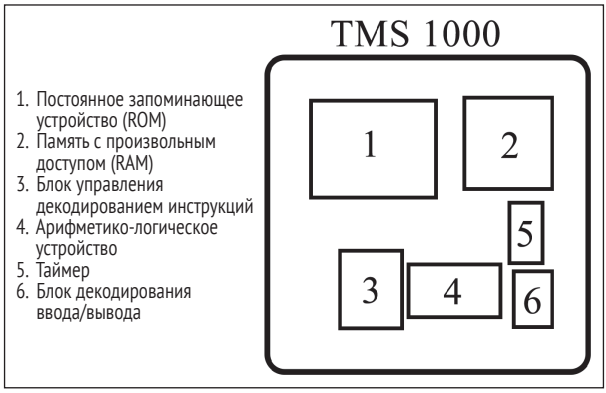

Рис. 1.1

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.

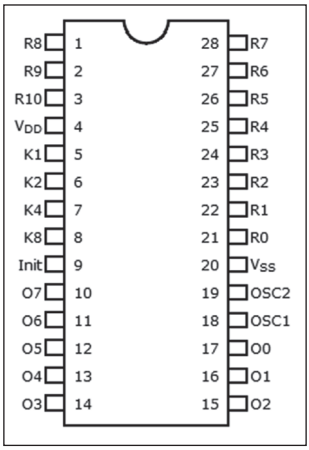

Рис. 1.2

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.

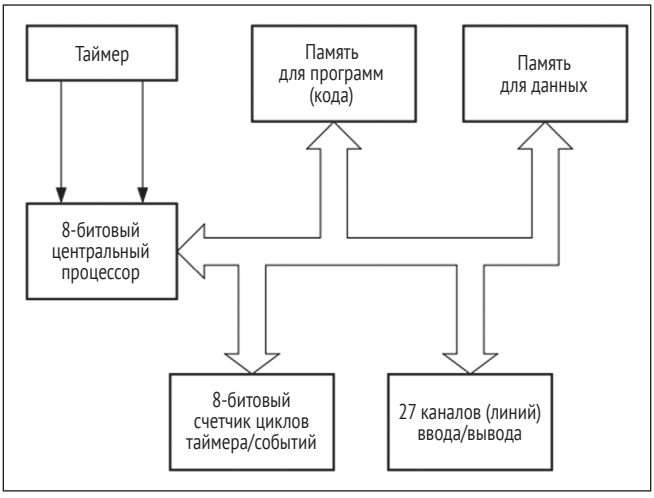

Рис. 1.3

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