Содержание

Предисловие

Linux – исключительно гибкая и мощная операционная система, и мне кажется, что мы еще не в полной мере осознали, какие преимущества сулит ее встраивание. Одна из возможных причин – многогранность этой темы и сложность ее изучения.

Конечно, можно самостоятельно пролагать себе путь в мире встраиваемых Linux-систем, как я и поступал в течение десяти лет, но отрадно видеть, что такие люди, как Крис, пишут книги, помогающие другим познакомиться со многими полезными вещами. Уж я-то точно нашел бы подобной книжке применение, если бы она существовала, когда я только начинал!

Понятно, что у меня есть личный интерес к проекту Yocto Project, поскольку это мое основное занятие и попытка внести свой вклад в мир встраиваемых Linux-систем. Одна из его основных целей – облегчить жизнь тем, кто создает такие системы. И мы добились кое-каких успехов, но есть области, требующие дальнейшей работы. Мы неустанно стремимся снять барьеры на пути начинающих и расширить круг посвященных, сделать технологию более доступной и распространенной.

В своей книге Крис преследует те же цели. Надеюсь, что вам понравится и книга, и сама система Linux и что в конечном итоге вы вольетесь в одно из активных сообществ, сформировавшихся вокруг проектов с открытым исходным кодом, стоящих за многими компонентами, о которых пойдет речь.

Ричард Пэрдай,
архитектор Yocto Project, член фонда Linux Foundation

Об авторе

Крис Симмондс – консультант по программному обеспечению и преподаватель, проживает в южной части Англии. С конца 1990-х годов занимается использованием Linux для создания встраиваемых систем и за это время успел поработать над многими интересными проектами, например: стереоскопическая камера, «умные» весы, различные абонентские приставки и домашние маршрутизаторы и даже большой шагающий робот.

Он часто выступает на конференциях по программному обеспечению с открытым исходным кодом и по встраиваемым системам, в том числе Embedded Linux Conference, Embedded World и Android Builders' Summit. С 2002 года читает курсы и проводит семинары по встраиваемым Linux-системам, а с 2010 – по встраиванию Android. Провел сотни презентаций во многих хорошо известных компаниях. Познакомиться с его работами можно в блоге «Inner Penguin» по адресу `www.2net.co.uk`.

Я благодарю своего редактора Саманту Гонсалвес, которая без устали следила за ходом работы и не давала мне сбиться с пути. Также выражаю благодарность всем, кто тратил время на чтение черновых вариантов, продираясь сквозь мои путаные словеса к сути. Спасибо Бихану Уэбстеру (Behan Webster), Клаасу ван Генду (Klaas van Gend), Тиму Бэрду (Tim Bird), Роберту Бергеру (Robert Berger), Матье Дешампу (Mathieu Deschamps) и Марку Фурману (Mark Furman). И конечно, я благодарен своей супруге Ширли Симмондс, которая всегда поддерживала меня и понимала, что я никак не мог помочь ей с ремонтом дома, потому что писал книгу.

О рецензентах

Роберт Бергер с 1993 года занимается проектированием, разработкой и управлением проектами в области встраиваемых систем как с жесткими требованиями к работе в режиме реального времени, так и без оных. Начиная с начала XXI века, он использует GNU/Linux на персональных компьютерах и серверах, но в основном для целей встраивания (в автомобильные системы, средства управления производственными процессами, робототехнику, телекоммуникационное оборудование, бытовую электронику и т. д.). Он регулярно присутствует на таких международных форумах, как Embedded World, Embedded Software Engineering Congress, Embedded Systems Conference и Embedded Linux Conference, в качестве эксперта и докладчика. Специализируется главным образом на преподавании, но также оказывает консультационные услуги (на английском и немецком языках) в различных странах. В сферу профессиональной компетенции Роберта входят как совсем небольшие системы реального времени (FreeRTOS), так и комплексы с несколькими процессорами или ядрами со встроенной системой GNU/Linux (уровень пространства ядра или пользователя, драйверы устройств, интерфейсы с оборудованием, отладка, многоядерная разработка, проект Yocto Project) с упором на свободное ПО и ПО с открытым исходным кодом. Он много путешествует по свету. Является исполнительным директором и специалистом по встраиваемому ПО в компании Reliable Embedded Systems со штаб-квартирой в городе Санкт-Барбата, Австрия. Проживает с семьей в Афинах. Связаться с Робертом можно через его сайт по адресу http://www.ReliableEmbeddedSystems.com.

Рецензировал книгу Rudolf J. Streif «Embedded Linux Systems with the Yocto Project» (Prentice Hall Open Source Software Development Series).

Тим Бэрд работает старшим программистом в компании Sony Mobile Communications, где отвечает за модификацию ядра Linux для использования в продуктах Sony. Также является председателем архитектурной группы рабочей группы по бытовой электронике (CE Working Group) в фонде Linux Foundation. Занимается Linux свыше 20 лет. Стоял у истоков двух отраслевых объединений, связанных со встраиваемыми Linux-системами, и является основателем и бессменным руководителем конференции Embedded Linux Conference. В прошлом написал вместе с Тимом книгу «Using Caldera OpenLinux».

Матье Дешамп – основатель компании ScourGE (www.scourge.fr), предлагающей клиентам инновационные услуги в области оборудования и ПО с открытым исходным кодом. Компания занимает лидирующие позиции в сфере телекоммуникаций, мобильной связи, управления промышленными процессами и систем поддержки принятия решений.

Оказывает консультационные услуги в области исследований и разработок, преподает. С 2003 года в качестве технического руководителя принимал участие

во многих мелких и крупных проектах, так или иначе связанных с GNU/Linux, Android, разработкой встраиваемых систем и безопасностью.

Марк Фурман, автор книги «OpenVZ Essentials», в настоящее время занимает должность инженера-системотехника в компании Info-Link Technologies. В сфере ИТ работает свыше 10 лет, специализируется на Linux и других продуктах с открытым исходным кодом, а также балуется с Arduino, Python и Raspberry Pi в Knox Labs, клубе компьютерщиков-энтузиастов в округе Нокс, штат Огайо.

Клаас ван Генд окончил факультет системной инженерии и управления Эйндховенского технического университета в Нидерландах. Работал в различных компаниях, включая Philips, Siemens и Bosch, писал программное обеспечение для экспериментальных образцов принтеров, шифрования видео, автомобильных информационно-развлекательных систем, медицинского оборудования, средств автоматизации производства и навигационных систем. В 2004 году перешел в компанию MontaVista Software, являющуюся идером на рынке встраиваемых Linux-систем. В роли системного архитектора и консультанта оказывал услуги многим европейским компаниям, помогая встраивать Linux в разработанные ими продукты.

Последние несколько лет работает преподавателем и консультантом в компании Vector Fabrics, небольшом стартапе, специализирующемся на программировании многоядерных компьютеров и динамическом анализе ПО. Преподает многоядерное программирование на C и C++, помогает заказчикам улучшить программы за счет правильного использования аппаратных ресурсов. Созданный компанией Vector Fabrics комплект инструментов Pareon помогает автоматически находить трудно воспроизводимые ошибки, в том числе гонку за данные, переполнение буфера, использование уже освобожденных областей кучи и стека, утечки памяти.

Клаас – автор более сотни журнальных статей на темы Linux (в том числе встраиваемых систем), программирования, производительности, проектирования систем и компьютерных игр. Он является соучредителем конференции Embedded Linux Conference Europe и выступал в роли ведущего разработчика в нескольких проектах с открытым исходным кодом, включая UMTSmon для сотовых сетей и игру-головоломку на темы физики The Butterfly Effect.

На досуге Класс читает городское фэнтези или отрывается в аэроклубе Нистелроде, где летает на планерах.

Бехан Уэбстер двадцать лет работал в различных технических отраслях, включая телекоммуникации, передачу данных, оптику, встраиваемые системы и автомобильную промышленность. Писал код для самых разных устройств – от крохотных до сверхбольших. Обладает опытом программирования ядра Linux, разработки встраиваемых систем и подготовки печатных плат к производству. В настоящее время занимает должность ведущего консультанта в компании Converse in Code Inc, где занимается встраиваемыми Linux-системами, а также руководит проектом LLVMLinux и преподает в интересах фонда Linux Foundation.

Вступление

Встраиваемая система – это устройство, содержащее внутри себя компьютер, но не выглядящее как компьютер. Стиральные машины, телевизоры, принтеры, автомобили, роботы – все они управляются каким-то компьютером, а иногда и не одним. Устройства становятся все сложнее, мы ожидаем от них все большего, а значит, растут требования к управляющей ими операционной системе. И все чаще такой системой становится Linux.

Богатство возможностей Linux проистекает из модели открытых исходных текстов, поощряющей совместное владение кодом. Это означает, что инженеры с разным образованием и опытом работы, зачастую работающие в конкурирующих компаниях, могут объединить усилия для создания ядра операционной системы, поддержания его актуальности и соответствия новейшим разработкам в области оборудования. Общая кодовая база позволяет поддержать самые разные устройства – от мощнейших суперкомпьютеров до наручных часов. ОС Linux – лишь один компонент, а для создания работающей системы нужно еще много чего: от базовых инструментов типа командной оболочки до графического интерфейса пользователя, который позволяет взаимодействовать с вебом и облачными службами. Ядро Linux в сочетании с обширным набором других открытых компонентов становится основой для системы, способной исполнять различные роли.

Однако гибкость – палка о двух концах. Да, у проектировщика системы появляется широкий спектр возможных решений задачи, но вместе с ним и проблема выбора оптимального решения. Цель этой книги – подробно рассказать о том, как сконструировать встраиваемую Linux-систему из свободных программ с открытым исходным кодом, получив в результате надежный и эффективный продукт. В основу книги положен многолетний опыт работы автора в качестве консультанта и преподавателя.

Структура книги

Организационно книга устроена так же, как жизненный цикл типичного проекта встраиваемой Linux-системы. В первых шести главах изложено все, что нужно знать о подготовке проекта и устройстве системы на базе Linux, а завершается эта часть соображениями о выборе подходящей системы сборки Linux. Затем настает пора принятия ключевых решений об архитектуре и составных частях системы, в том числе флэш-памяти, драйверах устройств и системе инициализации. Следующий этап – написание приложений для работы на собранной встраиваемой платформе, этой теме посвящены две главы о процессах, потоках и управлении памятью. И наконец, в главах 12 и 13 обсуждаются отладка и оптимизация платформы. А в последней главе описана конфигурация Linux для выполнения приложений реального времени.

В главе 1 «Приступая к работе» описываются альтернативы, имеющиеся у проектировщика системы в самом начале проекта.

В главе 2 «О наборах инструментов» описаны компоненты набора инструментальных средств с упором на кросс-компиляцию. Рассказывается, где взять набор инструментов и как собрать его из исходного кода.

В главе 3 «Все о начальных загрузчиках» объяснена роль загрузчика ОС в инициализации оборудования, а в качестве примеров взяты загрузчики U-Boot и Barebox. Описывается также дерево устройств – способ представления конфигурации оборудования, используемый во многих встраиваемых системах.

Глава 4 «Портирование и конфигурирование ядра» содержит информацию о том, как выбрать ядро Linux для встраиваемой системы и настроить его для работы с оборудованием, находящимся внутри устройства. Здесь же рассказывается о портировании Linux на новое оборудование.

В главе 5 «Построение корневой файловой системы» мы познакомимся с пользовательским пространством встраиваемой Linux-системы и представим пошаговую инструкцию по конфигурированию корневой файловой системы.

В главе 6 «Выбор системы сборки» описаны две системы сборки встраиваемых Linux-систем, цель которых – автоматизация описанных ранее шагов. На этом завершается первая часть книги.

Глава 7 «Выбор стратегии хранения» посвящена обсуждению проблем, связанных с управлением флэш-памятью, в том числе неуправляемых микросхем флэш-памяти и карт типа MMC и eMMC для встраиваемых систем. Описаны файловые системы для каждой технологии. Рассматривается также вопрос об обновлении прошивки устройства пользователем.

В главе 8 «Введение в драйверы устройств» описывается, как находящийся в ядре драйвер устройства взаимодействует с оборудованием. Приведены примеры простых драйверов. Описаны также различные способы вызова драйвера из пользовательского адресного пространства.

В главе 9 «Инициализация системы – программа init» описано, как запускается первая программа, работающая в пользовательском пространстве, – `init`, и как она запускает все остальные программы в системе. Мы рассматриваем три варианта программы `init`, подходящие для встраиваемых систем разного типа: от самой простой, BusyBox `init`, до systemd.

Глава 10 «Процессы и потоки» посвящена рассмотрению встраиваемых систем с точки зрения прикладного программиста. В ней речь пойдет о процессах, потоках, межпроцессном взаимодействии и политиках планирования.

В главе 11 «Управление памятью» обсуждаются идеи, лежащие в основе механизма виртуальной памяти и отображения адресного пространства. Рассказано также о том, как узнать, какие части памяти используются, и обнаружить утечки памяти.

Глав 12 «Отладка в GDB» посвящена использованию отладчика GDB для интерактивной отладки кода, работающего в ядре и в пользовательском пространстве. Описан также отладчик ядра `kdb`.

В главе 13 «Профилирование и трассировка» рассматриваются методы измерения производительности системы, начиная с получения профиля работы системы в целом и заканчивая выявлением узких мест, приводящих к снижению производительности. Здесь же описана программа Valgrind, которая проверяет, правильно ли приложение синхронизирует потоки и управляет памятью.

В главе 14 «Программирование в режиме реального времени» вы найдете подробное руководство по созданию программ реального времени в Linux, включая настройку ядра и наложение на него заплат реального времени, а также описание инструментов для измерения задержек. Здесь же приводятся сведения о том, как уменьшить число страничных прерываний путем блокировки памяти.

Что необходимо для чтения этой книги

В книге используется только программное обеспечение с открытым исходным кодом, в большинстве случаев – последние версии, доступные на момент написания книги. Я старался описывать основные возможности, не привязываясь к конкретной версии, но некоторые команды могут не работать с более поздними версиями – от этого никуда не денешься. Надеюсь, впрочем, что сопроводительные описания достаточно информативны, так что вам не составит труда применить те же принципы к последующим версиям пакета.

В создании любой встраиваемой системы участвуют две системы: исходная – в которой кросс-компилируется программный код, и целевая – в которой этот код будет работать. В качестве исходной системы я использовал Ubuntu 14.04, но годится (возможно, с минимальной модификацией) почти любой дистрибутив Linux. Я вынужден был выбрать какую-то целевую систему для представления встраиваемой. В книге рассматриваются два варианта: BeagleBone Black и QEMU – эмулятор системы с процессором ARM. Второй вариант означает, что примеры можно выполнять, не тратясь на оборудование для реального целевого устройства. При этом ничто не мешает попробовать примеры на различных целевых платформах, изменив такие детали, как имена устройств и распределение памяти.

Версии основных пакетов для целевой системы: U-Boot 2015.07, Linux 4.1, Yocto Project 1.8 «Fido» и Buildroot 2015.08.

Предполагаемая аудитория

Эта книга рассчитана в первую очередь на разработчиков Linux и системных программистов, которые уже знакомы со встраиваемыми системами и хотят узнать, как создавать лучшие в своем классе устройства. Требуются понимание основ программирования на языке C и опыт системного программирования.

Графические выделения

В этой книге для выделения семантически различной информации применяются различные стили. Ниже приведены примеры стилей с пояснениями.

Фрагменты, имена функций, таблиц базы данных, папок и файлов, URL-адреса, адреса в Twitter и данные, вводимые пользователем, выглядят следующим образом: «Мы могли бы воспользоваться функциями потокового ввода-вывода `fopen(3)`, `fread(3)` и `fclose(3)`».

Отдельно стоящие фрагменты кода набраны так:

```
static struct mtd_partition omap3beagle_nand_partitions[] = {
  /* Размеры разделов задаются в терминах размера блока NAND-памяти */
  {
    .name = "X-Loader",
    .offset = 0,
    .size = 4 * NAND_BLOCK_SIZE,
    .mask_flags = MTD_WRITEABLE, /* только для чтения */
  }
}
```

Чтобы привлечь внимание к части кода, строки или отдельные слова выделяются полужирным шрифтом:

```
static struct mtd_partition omap3beagle_nand_partitions[] = {
  /* Размеры разделов задаются в терминах размера блока NAND-памяти */
  {
    .name = "X-Loader",
    .offset = 0,
    .size = 4 * NAND_BLOCK_SIZE,
    .mask_flags = MTD_WRITEABLE, /* только для чтения */
  }
}
```

Текст, который вводится на консоли или выводится на консоль, напечатан следующим образом:

```
# flash_erase -j /dev/mtd6 0 0
# nandwrite /dev/mtd6 rootfs-sum.jffs2
```

Новые термины и важные слова набраны полужирным шрифтом. Также выделяются элементы интерфейса, например пункты меню и поля в диалоговых окнах: «Во второй строке на консоль выводится сообщение **Please press Enter to activate this console**».

Предупреждения и важные замечания оформлены так.

Советы и рекомендации выглядят так.

Отзывы

Мы всегда рады отзывам читателей. Расскажите нам, что вы думаете об этой книге – что вам понравилось или, быть может, не понравилось. Читательские отзывы

важны для нас, так как помогают выпускать книги, из которых вы черпаете максимум полезного для себя.

Чтобы отправить обычный отзыв, просто пошлите письмо на адрес feedback@packtpub.com, указав название книги в качестве темы.

Если вы являетесь специалистом в некоторой области и хотели бы стать автором или соавтором книги, познакомьтесь с инструкциями для авторов по адресу www.packtpub.com/authors.

Поддержка клиентов

Счастливым обладателям книг Packt мы можем предложить ряд услуг, которые позволят извлечь из приобретения максимум пользы.

Загрузка кода примеров

Вы можете скачать код примеров ко всем приобретенным вами книгам издательства Packt в своем личном кабинете на сайте http://www.PacktPub.com. Если книга была куплена в другом месте, зайдите на страницу http://www.PacktPub.com/support, зарегистрируйтесь, и мы отправим файлы по электронной почте.

Опечатки

Мы проверяли содержимое книги очень тщательно, но какие-то ошибки все же могли проскользнуть. Если вы найдете в нашей книге ошибку, в тексте или в коде, пожалуйста, сообщите нам о ней. Так вы избавите других читателей от разочарования и поможете нам сделать следующие издания книги лучше. При обнаружении опечатки просьба зайти на страницу http://www.packtpub.com/submit-errata, выбрать книгу, щелкнуть по ссылке **Errata Submission Form** и ввести информацию об опечатке. Проверив ваше сообщение, мы поместим информацию об опечатке на нашем сайте или добавим ее в список замеченных опечаток в разделе **Errata** для данной книги.

Список подтвержденных опечаток можно просмотреть, введя название книги в поле поиска на странице https://www.packtpub.com/books/content/support. Информация появится в разделе **Errata**.

Нарушение авторских прав

Незаконное размещение защищенного авторским правом материала в Интернете – проблема для всех носителей информации. В издательстве Packt мы относимся к защите прав интеллектуальной собственности и лицензированию очень серьезно. Если вы обнаружите незаконные копии наших изданий в любой форме в Интернете, пожалуйста, незамедлительно сообщите нам адрес или название вебсайта, чтобы мы могли предпринять соответствующие меры.

Просим отправить ссылку на вызывающий подозрение в пиратстве материал по адресу copyright@packtpub.com.