Применение программного обеспечения с открытым кодом для встраиваемых систем

Application of software with open code for integrated systems of situation centers

Ключевые слова: свободное ПО – free software; ПО с открытым кодом – software with open code; встраиваемые системы – built-in systems; Linux – Linux; операционные системы – operating systems.

Операционная система Linux, как и все сообщество открытого ПО, завоевывает все большую популярность в мире. Linux работает на множестве аппаратных платформ: от наладонных компьютеров до многопроцессорных мэйнфреймов. Тем не менее, многие разработчики боятся применять эту ОС в своих продуктах. В статье на примере ОС Linux перечисляются плюсы решений на основе открытого ПО. Также в статье рассматриваются специфические проблемы лицензирования, с которыми может столкнуться разработчик при применении открытого ПО в своих разработках.

Operating system Linux, as well the whole community of open Software, wins more and more popularity in the world. Linux operates with a great many of hardware platforms: from palm computers to multiprocessor mainframes. Nevertheless, many design engineers are afraid of using this Operating system in their products. Taking Operating system Linux as an example, the article considers positive decisions on the basis of the open Software. The article also covers specific problems of licensing, which can be faced by a design engineer when using open. Software in his or her developments.

ГУСАКОВ / GUSAKOV A.
Андрей Васильевич
аспирант кафедры безопасных информационных технологий, Санкт-Петербургский государственный университет информационных технологий, механики и оптики, Санкт-Петербург

Еще десять лет назад для встраиваемых систем разработчику приходилось разрабатывать свои продукты либо использовать значительно упрощенные варианты известных программ и алгоритмов. Связанно это было, в первую очередь, со значительными ограничениями ресурсов во встраиваемых системах. За последние годы цены на основные компоненты встраиваемых систем значительно снизились, появилась возможность строить такие системы на аппаратной базе, сравнимой по производительности с персональными компьютерами. Значительно расширился выбор всевозможного ПО и ОС, которые можно применять для систем с архитектурой, отличной от x86. Это, в первую очередь, заслуга большого количества энтузиастов, занимающихся портированием и адаптацией различных ОС и ПО с открытым кодом. Коммерческие решения для таких архитектур развиваются значительно хуже в силу ряда причин, связанных с защитой интеллектуальной собственности.

При выборе операционной системы (ОС) для той или иной встраиваемой системы обычно используется несколько критериев, в частности – доступность на рынке, требовательность к ресурсам, доступность программных средств, богатство функциональных возможностей, надежность и производительность. Решающим фактором может стать даже такой простой факт, что некая операционная система работает на выбранном микропроцессоре. Модификация ОС для нового процессора или процессорного модуля может оказаться слишком долгим делом, наличие же готовой операционной системы ведет к существенной экономии времени и средств.

При проектировании встраиваемых систем на первый план может выйти наличие или отсутствие необходимых ПО и драйверов в составе конкретной ОС. Например, для системы мониторинга транспорта с возможностью удаленного видеонаблюдения написание драйвера для обычной USB-камеры или платы захвата видео может вылиться в месяцы работы небольшого коллектива. Прибавьте к этому времени еще время на разработку и написание алгоритмов сжатия видео, алгоритмов определения движения

и других прикладных задач. Аналогичная ситуация – с различными интерфейсами и стеками протоколов для них.

Необходимость ускорения выпуска новой продукции на рынок и снижения общей стоимости систем подвигает многих разработчиков встраиваемых приложений на применение ОС Linux и других программных средств с открытыми исходными кодами. Подобное решение обусловлено тем, что используя открытое программное обеспечение, разработчики получают возможность сконцентрировать больше усилий на совершенствовании своей продукции. Однако встраиваемые системы обладают уникальным набором требований и ограничений: небольшие объемы памяти, загрузка из флэш-памяти, жесткие ограничения по энергопотреблению. Использование в таких приложениях операционной системы Linux связано с рядом дополнительных проблем.

Самая распространенная сегодня версия Linux – это, безусловно, версия для совместимых с ПК компьютеров, созданных на базе процессоров с архитектурой x86. Процессоры семейства x86 применяются не только в персональных, но и во многих встраиваемых системах. Существует множество моделей x86-процессоров – от чрезвычайно быстродействующих до самых экономичных. Полностью совместимый x86 компьютер может уместиться на один кристалл. Однако модификации ОС Linux существуют для многих типов встраиваемых микропроцессоров, более приспособленных для применения во встраиваемых решениях. Обновленные Linux-дистрибутивы для иных, нежели x86, платформ появляются регулярно. О поддержке версий операционной системы Linux для различных процессорных платформ заявляет все большее число производителей программного обеспечения. Впрочем, основной движущей силой являются пользователи. Растущий спрос на приложения, стеки протоколов и собственно Linux-версии для различных процессоров существенно ускоряет процесс создания соответствующих модификаций этой ОС. Также играет роль наличие открытых кодов для необходимого приложения – необязательно ждать, пока разработчик произведет портирование на необходимую платформу – это можно сделать самому. Кроме того, большинство прикладных программ полностью платформонезависимы, т.е. достаточно просто скомпилировать ПО под нужную архитектуру, не внося никаких изменений в код.

Решение о применении ОС Linux автоматически накладывает некоторые ограничения на выбор возможного CPU для разработки. Так, автоматически отметаются почти все микроконтроллеры, ядра которых не имеют MMU (блока управления памятью). Теоретически можно применить специальную версию Linux для контроллеров без MMU – μClinux, но с отказом от всех преимуществ защищенной памяти. В этом случае ошибка в любом прикладном приложении может стать причиной краха всей системы. Выбрав в качестве ОС Linux, стоит использовать все ее преимущества.

Следующее ограничение – это ОЗУ и ПЗУ. В большинстве современных контроллеров недостаточно «набортной» (встроенной в тот же чип, что и CPU) памяти для работы Linux. Применение внешней (по отношению к CPU) памяти обычно вызывает яростное сопротивление со стороны схемотехника и трасировщика ПП, так как разводка такой памяти требует соблюдения ряда противоречивых требований. Зачастую в качестве ППЗУ во встраиваемых системах используется накопитель на флэш-памяти. Однако применение флэш-памяти имеет некоторые ограничения, связанные, в первую очередь, с ограниченным ресурсом перезаписи такой памяти. В состав Linux входит драйвер ОЗУ диска используемой для имитации дискового накопителя в определенной части оперативной памяти. Такой диск может использоваться для хранения файлов, которые не потребуются после перезагрузки системы, что значительно снижает нагрузку на флэш-память. Для работы с флэш-памятью файловая система обязана иметь механизм контроля износа, сводящий число операций стирания флэш-блоков к разумному минимуму. Для этих целей разработаны особые файловые системы (JFFS, UBIFS и т.д.). Для повышения надежности системы зачастую используется одновременно несколько файловых систем: все системные файлы находятся в одном разделе, к которому разрешен доступ только на чтение, настройки и «черный ящик» хранятся в независимом разделе с возможностью записи, а все временные файлы – на RAM-диске.

Для нормальной работы любой операционной системы требуются определенные ресурсы, главным из которых является память. Одним ОС необходимо больше памяти, другим – меньше. Применения операционных систем с повышенными требованиями к памяти во встраиваемых приложениях, где и ПЗУ, и ОЗУ на вес золота, следует избегать. По умолчанию Linux необходимы довольно значительные объемы памяти, что неприемлемо для большинства встраиваемых систем. Как правило, дистрибутив ОС Linux занимает несколько сотен мегабайтов. В несжатом состоянии стандартное Linux-ядро занимает около полутора мегабайтов, а требуемый этим ядром объем ОЗУ составляет более 4 Мбайт. Кроме того, необходима память для

поддержания файловой системы, а если система работает в бездисковом режиме – память для ОЗУ дисков.

Решение проблемы требуемой для операционной системы Linux памяти заключается в тщательной настройке ядра ОС. Ядро имеет модульную структуру, что позволяет манипулировать неиспользуемыми в конкретных приложениях функциональными возможностями. Выбор соответствующих модулей и их конфигурирование позволяют значительно снизить потребность операционной системы в ОЗУ и ПЗУ. Однако последнее время стоимость микросхем ОЗУ и ПЗУ непрерывно снижается и даже наблюдается интересная ситуация, когда микросхемы меньшего объема стоят дороже микросхем большего объема. Это связано с тем, что микросхемы малого объема производятся по остаточному принципу в малых количествах для поддержания старых проектов. Наличие памяти «с запасом» опять же позволяет разработчику сконцентрироваться на улучшении своей продукции, а не на подгонке ПО под жесткие требования аппаратной платформы.

Как правило, встраиваемые системы работают в безоператорном режиме. Это означает, что для их запуска и нормальной работы участие оператора не требуется. Операционная система Linux разрабатывалась на основе модели UNIX и задумывалась как ОС для настольных компьютеров. В традиционных версиях Linux для взаимодействия с оператором имеется алфавитно-цифровая консоль, позволяющая управлять загрузкой операционной системы, осуществлять ввод команд и контролировать работу ОС. Однако встроенным системам оператор, как правило, не нужен, поэтому во встраиваемой Linux подобное взаимодействие предусматриваться не должно.

Создание безоператорной версии операционной системы Linux не составляет особого труда: достаточно заменить консоль фиктивным устройством. Если вместо выполнения процедуры регистрации стартовый скрипт будет сразу осуществлять запуск прикладных программ, взаимодействия с оператором удастся избежать. Мониторинг при необходимости можно выполнять удаленно, что обеспечивается возможностью дистанционного входа в систему и назначением псевдоустройству TTY роли консоли. В процессе создания встраиваемых приложений разработчикам приходится применять различные программные средства типа компиляторов и отладчиков, причем, как правило, из-за недостатка памяти и производительности выполнение этого ПО на целевой платформе оказывается невозможным. Большинство Linux-программ пишется, запускается и отлаживается в одной и той же Linux-системе. После того как программа будет отлажена, ее работоспособный исполняемый образ (или содержащий такой образ пакет) может быть скопирован в другие Linux-машины. Аналогичным образом можно создавать программное обеспечение для встраиваемой системы при том условии, что в этой системе имеется оперативная и дисковая память достаточного для работы инструментальных средств объема. На практике такая ситуация довольно редка, поэтому у программиста возникает необходимость использования средств кросс-разработки.

Для кросс-разработки вполне подойдут широкораспространенные GNU-компиляторы. Необходимо также определить метод загрузки образа ядра и файловой системы из инструментальной системы в целевую. Если целевая система способна работать с каким-либо дисковым накопителем и загружаться с этого диска, можно подключить этот диск к инструментальной машине, записать на него образ ядра и файловую систему, затем подключить его к целевой платформе и осуществить загрузку. В этом случае необходимо чтобы ОС на инструментальной и целевой машине поддерживали этот тип файловой системы. В том случае, когда загрузка ОС в целевую систему может осуществляться по сети, на инструментальном компьютере достаточно запустить файловый сервер.

Кросс-разработка предполагает поддержку и кросс-отладку. В среде кросс-разработки отладчик исполняется на инструментальной системе. В Linux-дистрибутивы входит GNU-отладчик GDB, который может использоваться для отладки программ, написанных как в кодах инструментальной платформы, так и в кодах целевого компьютера. Разумеется, отладка Linux-ядра и драйверов устройств не может осуществляться так же, как и отладка пользовательского приложения. Установка контрольной точки в ядре или иная остановка исполняющегося ядра процесса способна привести к остановке самого агента. Большая часть настольных дистрибутивов Linux не обеспечивает функций, необходимых для отладки ядра и драйверов. Однако рассчитанные на встраиваемые системы дистрибутивы такими функциями обладают. Обычно отладчик GDB взаимодействует с агентом отладки ядра по низкоуровневому последовательному соединению. Такое решение позволяет осуществлять пошаговое исполнение ядра или драйвера и даже ставить контрольные точки в процедурах обслуживания прерываний. В редких случаях может понадобиться аппаратный отладчик, который подключается непосредственно к целевому процессору и взаимодействует с ним

на аппаратном уровне. Применение такого отладчика обычно ограничивается отладкой процедуры начальной инициализации аппаратной платформы и во время написания прикладного ПО не требуется. С программной точки зрения Linux является UNIX-подобной операционной системой. Одно из основных объяснений большого числа Linux-программ – такие же, как и в UNIX, API-интерфейсы, а также форматы объектных и исполняемых файлов. Значительная часть написанного за последние три десятка лет для ОС UNIX общедоступного программного обеспечения может без каких-либо изменений использоваться в Linux.

Благодаря популярности ОС Linux уже появилось несколько поставщиков программных средств, специализирующихся на версиях этой ОС для встраиваемых систем. Как правило, такие поставщики предлагают также проблемно-ориентированную техническую поддержку, включающую исправление ошибок в устаревших Linux-версиях и помощь в вопросах переноса ПО на встраиваемые платформы. Надежность и производительность – две очень важные характеристики операционной системы, которые невозможно улучшить ни прикладными, ни библиотечными программами. Надежная программа в ненадежной операционной системе сама становится ненадежной. Надежность ОС определяется лежащими в ее основе базовыми принципами, а также особенностями архитектуры и программирования. Многие встраиваемые приложения характеризуются гораздо более высокими требованиями к безотказности работы операционной системы, чем настольные компьютеры, и выбор действительно надежной ОС приобретает в этой связи огромное значение. Надежность и производительность Linux можно оценить, ознакомившись с результатами независимых тестирований.

Несмотря на весьма веские причины выбора Linux в качестве ОС для встраиваемой системы или встраиваемого устройства, в области лицензирования ей свойственны определенные проблемы. Операционная система Linux – это не общедоступное ПО, ее лицензирование осуществляется согласно требованиям лицензии GPL (GNU Public License), устанавливающей строгий набор правил пользования. Что значит лицензия GPL для разработчика встраиваемых систем, имеющего дело с ОС Linux? Если разработчик каким-либо образом модифицирует Linux-ядро или утилиту, перенесет их на другую платформу или расширит набор их функциональных возможностей, он будет обязан опубликовать эти изменения в Интернете и предоставлять их бесплатно каждому, кто их запросит. При не вполне аккуратном поведении можно запросто лишиться прав на собственные программные средства, поэтому по всем вопросам, касающимся охраны авторства в области открытых исходных текстов, имеет смысл советоваться с опытным юристом.

Включать ОС Linux в состав продаваемой аппаратуры считается вполне нормальным. В случае каких-либо модификаций ядра, библиотек или утилит Linux соответствующие исходные тексты должны стать общедоступными и распространяться бесплатно. Исполняющиеся в среде Linux прикладные программы могут при этом оставаться запатентованными, необходимо лишь следить за тем, чтобы в состав этих программ не входил никакой подпадающий под действие лицензии GPL (а равно и любой другой лицензии) код. Драйверы также могут быть запатентованными, при этом они должны быть отделены от ядра Linux. Одним из простейших способов является написание драйвера как загружаемого модуля ядра. Если же драйвер компонуется как часть базового ядра, вопрос о распространении на эту часть действия GPL-лицензии является весьма непростым, и лучшим решением здесь будет проконсультироваться с грамотным юристом, занимающимся вопросами охраны интеллектуальной собственности. Несмотря на то, что изначально ОС Linux не была рассчитана на использование во встраиваемых устройствах, постоянное развитие этой ОС и постоянная работа сообщества сделали ОС Linux очень привлекательной и в данном сегменте.

Литература

1. *Таненбаум Э.* Современные операционные системы. – СПб.: «Питер», 2002.

2. Successfully using Linux and open software in an Embedded System Design. Greg Rose. «LynuxWorks». 2005. – www.lynuxworks.com /products /whitepapers/usinglinux.php3.

3. *Yaghmour K., Masters J.* Building Embedded Linux Systems Second Edition. – «Gilad Ben-Yossef & Phillipe Gerum», «O'Reilly», 2007.