Техника обновления прошивки устройств на базе ОС Linux

Виктор Полстюк, инженер-программист, дизайн-центр электроники Promwad

В последнее время на рынке встраиваемых систем сформировалось несколько заметных тенденций. Во-первых, стоит отметить рост популярности ОС Linux. Широкое применение этой операционной системы обусловлено появлением новых микропроцессоров, обладающих достаточной производительностью для работы на базе Linux при сохранении невысокой стоимости и энергопотребления (что типично для микроконтроллеров). Также использование Linux позволяет сократить время разработки устройства за счет использования программного обеспечения со свободной лицензией.

Второй заметной тенденцией стало стремление к постоянному развитию продукта даже после его продажи. Так, производители электроники разрабатывают новое программное обеспечение для ранее выпущенных устройств, позволяя пользователям самостоятельно обновить прошивку и таким образом добавить новый функционал.

Сегодня возможность обновления программного обеспечения электронного устройства превратилась из дополнительного преимущества в обязательное требование. Безусловно, распространение Linux повлияло и на развитие способов обновления программной части устройств. Поэтому целью данной статьи является общее описание механизма обновления прошивки на основе ОС Linux и приведение нескольких конкретных реализаций в качестве примеров.

Рассмотрим разделы памяти системы (рис. 1) и те части памяти, которые необходимо обновлять при переходе на новую версию ПО. Как правило, системы на базе Linux имеют следующую структуру энергонезависимой памяти: в первом ее разделе хранится программа-загрузчик ядра Linux, которая в свою очередь может выполняться в несколько этапов. Например, начальный загрузчик малого размера копируется во внутреннюю память процессора, проводит инициализацию внешнего ОЗУ и копирует загрузчик второго уровня во внешнее ОЗУ. Загрузчик второго уровня (например, U-Boot) копирует ядро Linux в ОЗУ и передает ему управление. В последнюю очередь происходит запуск пользовательских приложений, размещенных в последнем разделе флэш-памяти. Таким образом, становится очевидной необходимость обновления раздела памяти с пользовательскими приложениями и раздела с ядром ОС.

События для запуска процесса обновления ПО может служить:

- запуск системы (в этом случае программа обновления встраивается в начальный загрузчик либо запускается при инициализации ОС);
- подключение внешнего носителя (USB- или SD-карты) во время работы устройства;
- обнаружение на сервере обновления ПО более новой версии;
- действия пользователя.

Файлы для обновления могут быть скопированы при использовании подключаемых носителей информации либо получены по сети (при наличии в системе Ethernet или модуля Wi-Fi). Как правило, для этих целей используется протокол TFTP, широко поддерживаемый различными загрузчиками. В случае, когда обновлением ПО занимается запущенная в ОС программа, копирование необходимых файлов с сервера может происходить с помощью приложения wget либо с использованием библиотеки libcurl.

Важным моментом в обновлении ПО является проверка версии полученных файлов и их целостности. Для этого могут быть использованы такие алгоритмы, как MD5, CRC32 и др. Также желательно проверять совместимость новой прошивки с данным устройством (ревизия печатной платы).

Когда файлы для обновления получены и проверены, можно переходить к процессу их записи в ПЗУ системы. В зависимости от объема ПЗУ реализуются различные схемы обновления ПО.

Вариант 1: Если используется ПЗУ с объемом, достаточным для хранения старой и новой прошивок (рис. 2), реализуется схема с возможностью возврата к предыдущей версии ПО в случае неудачно завершившегося процесса обновления. Непосредственно перезапись памяти осуществляется

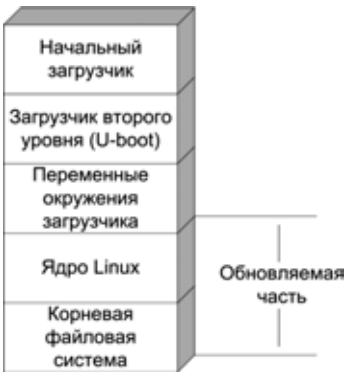

рис. 1. **Области флэш-памяти устройства на базе ОС Linux**

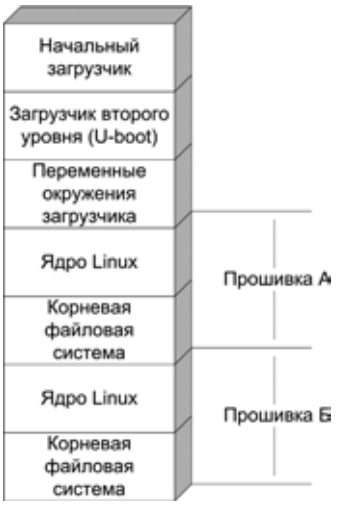

рис. 2. **Области флэш-памяти устройства с возможностью возврата к старой прошивке**

рис. 3. **Алгоритм обновления ПО с возможностью возврата к старой прошивке**

с помощью программ flash_eraseall (для стирания области памяти) и nandwrite (для записи образа в NAND-память), которые входят в состав пакета mtd-utils.

Далее необходимо установить переменные окружения загрузчика U-Boot для загрузки новой прошивки. Для чтения и записи переменных окружения U-Boot из Linux используются команды fw_printenv и fw_setenv (исходные коды входят в дистрибутив U-Boot). Кроме типичных параметров, необходимых для запуска ОС (расположение образа ядра в ПЗУ, название раздела, содержащего корневую файловую систему и др.), в переменных окружения можно хранить число неудачных загрузок и при достижении определенного порога возвращаться на предыдущее ПО (рис. 3).

Вариант 2: При малом объеме ПЗУ ядро Linux и приложения, а также необходимые библиотеки должны быть скопированы в ОЗУ. После чего может быть проведено стирание и перезапись флэш-памяти. Очевидно, что данная схема не позволяет вернуться к предыдущей прошивке в случае сбоя в процессе обновления.

В загрузчике U-Boot реализована встроенная функция проверки целостности загружаемого образа ядра Linux (управляется переменной verify), что позволяет еще до загрузки ОС проверить ее целостность и с помощью hush-скриптов определить альтернативные источники для загрузки ПО (старая прошивка или TFTP) [1].

Рассмотрим способы реализации механизма обновления ПО в нескольких открытых проектах, таких как OpenWrt, Openmoko и OpenInkpot.

OpenWrt

В этом проекте по созданию свободного ПО для сетевых роутеров OpenWrt [2] прошивка может обновляться из различных источников: веб-интерфейс или TFTP.

Наиболее простым для пользователя является обновление ПО через веб-страницу. При этом пользователь посредством веб-браузера заходит на страницу настройки устройства (раздел обновления ПО) и копирует новое ПО с помощью специальной HTML-формы.

Второй вариант получения обновлений — это загрузка по TFTP. Такой способ хорошо подходит для тех пользователей, которые самостоятельно создали образ для прошивки. В случае возникновения проблем они смогут вернуть старую прошивку обратно (с помощью TFTP). Процедура происходит следующим образом: при включении устройства запускается загрузчик, который выполняет инициализацию системы, а также проверяет и загружает исполняемый код. Если прошивка не прошла проверку на целостность, загрузчик будет считать ее поврежденной и автоматически перейдет в режим ожидания загрузки прошивки по сети. После этого пользователь может загружать новую прошивку с компьютера по TFTP.

При необходимости, загрузка ПО с TFTP-сервера может быть инициирована вручную. При этом требуется подключится к консоли устройства CFE (Common Firmware Environment) по последовательному порту и прервать процедуру обычной инициализации устройства. Далее необходимо сконфигурировать сеть, скопировать новую прошивку с TFTP и записать ее во флэш-память устройства.

Openmoko

В качестве примера обновления ПО устройства через USB можно привести проект по созданию свободной прошивки для смартфонов под названием Openmoko [3]. Механизм обновления реализован в соответствии со спецификацией USB Device Class Specification for Device Firmware Upgrade [4]. Целью данного документа является создание универсального механизма обновления ПО устройства, оснащенного USB, т.е. механизма, который не зависит от производителя и конкретной аппаратной платформы. Это позволяет использовать одну программу в составе операционной системы для установки ПО на разные устройства, используя соответствующие образы прошивки. Кроме записи новой прошивки упомянутая спецификация описывает процедуру считывания текущей прошивки на PC.

В Openmoko DFU является частью модифицированной версии загрузчика U-Boot. Также в рамках проекта разрабатывается утилита DFU-Util, которая позволяет передавать и записывать ПО во внутреннюю NAND-память устройства и записывать программу в его оперативную память. Последняя функция может быть использована для отладки низкоуровневого кода (например, ядра) без перезаписи флэш-памяти. А предусмотренная в DFU-Util возможность считывания содержания NAND-памяти позволяет создавать резервную копию прошивки устройства.

OpenInkpot

Для обновления ПО электронной книги в проекте OpenInkpot используется карта памяти SD [5]. Для этого новое ПО в виде файла .oifw помещается в корневой каталог SD-карты. При подаче питания модифицированный загрузчик U-boot проверяет наличие прошивки на карте памяти и проводит проверку ее целостности, используя CRC32. При возможности обновления он запрашивает у пользователя подтверждение выполнения операции перезаписи флэш-памяти устройства. Далее происходит обновление ядра и корневой файловой системы либо полная переустановка ПО, включая начальный загрузчик.

Стоит заметить, что инженеры не ограничиваются одним способом обновления ПО устройств на базе ОС Linux, а реализуют одновременную поддержку нескольких. Это связанно с тем, что при разработке устройства необходимо иметь инструмент обновления с множеством функций, а при эксплуатации устройства — максимально простой и надежный.

СПИСОК СЕТЕВЫХ РЕСУРСОВ:

1. *http://www.denx.de/wiki/DULG/CommandLineParsing*
2. *http://wiki.openwrt.org/doc/howto/installing*
3. *http://wiki.openmoko.org/wiki/USB_DFU*
4. *http://www.usb.org/developers/devclass_docs/DFU_1.1.pdf*
5. *http://openinkpot.org/wiki/0.2/InstallationGuide*