ПРИМЕНЕНИЕ ПАКЕТОВ ПОДДЕРЖКИ ПЛАТ В КАЧЕСТВЕ ИНСТРУМЕНТА АДАПТАЦИИ СПЕЦИАЛИЗИРОВАННЫХ ОПЕРАЦИОННЫХ СИСТЕМ ДЛЯ ФУНКЦИОНИРОВАНИЯ НА ПЛАТФОРМЕ «МУЛЬТИКОР»

А.В.Сеньков

ООО "СВД Встраиваемые Системы", a.senkov@kpda.ru

Операционные системы, предназначенные для функционирования на специализированных аппаратных платформах, по своим характеристикам значительно отличаются от операционных систем общего назначения не только способностью функционировать в условиях ограниченных ресурсов в реальном масштабе времени, но и наличием средств и механизмов адаптации для различных целевых платформ. К таким средствам относятся пакеты поддержки плат (BSP – Board Support Package) операционной системы QNX Neutrino.

В состав BSP входят программные модули, обеспечивающие старт ядра системы, а также модули поддержки периферийных устройств, таких как последовательные порты, flash-память, сеть и т.д. Компоненты BSP могут использоваться для обеспечения альтернативных способов загрузки операционной системы, например, через последовательный канал или из flash-памяти процессорной платы.

Принципиальным отличием подхода, применяющегося при адаптации ОС QNX Neutrino для целевой платформы по сравнению с большинством распространенных ОС, является отсутствие необходимости в модификации ядра системы. Это достигается за счет таких характеристик ОС QNX Neutrino как возможности гибкого масштабирования и архитектуры на основе микроядра. Микроядерная архитектура предполагает наличие в ОС небольшого по размеру и функционально детерминированного ядра («истинного» микроядра), а также программной шины, обеспечивающей межзадачное взаимодействие остальных модулей в системе, как это показано на рисунке 1.

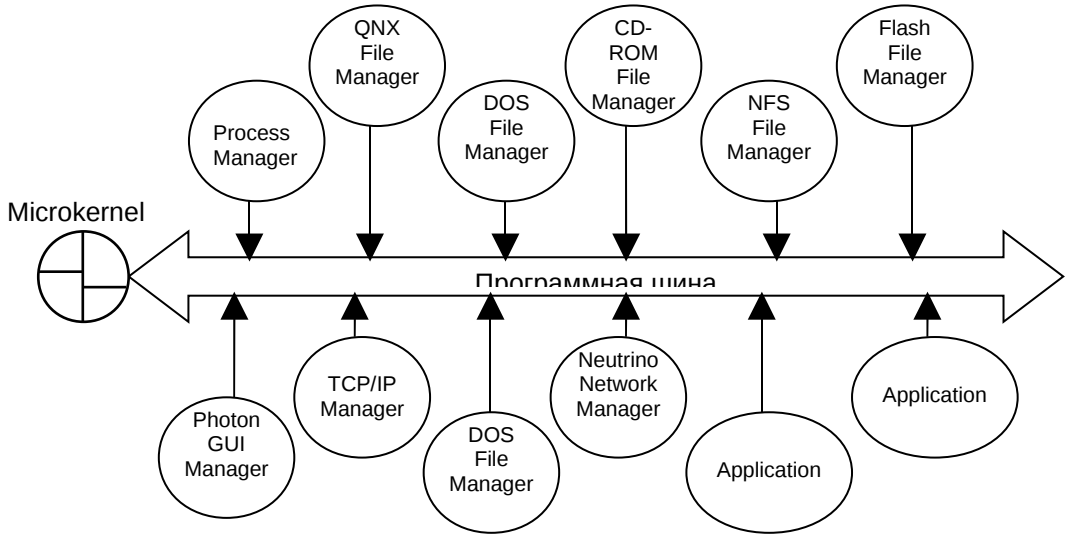


Рис. 1. Межзадачное взаимодействие в микроядерной ОС QNX Neutrino

Микроядерная архитектура ОС QNX Neutrino предполагает равнозначность системных и пользовательских процессов. Все процессы выполняются в контексте пользователя и переключаются в контекст ядра во время системных вызовов. Основным механизмом межзадачного взаимодействия является обмен сообщениями. С точки зрения системы, сообщение QNX – это неструктурированный массив байт, содержимое которого понятно только процессу-получателю и процессу-передатчику. Прозрачная передача сообщений в сети QNX обеспечивает сетевую прозрачность операционной системы и, соответственно, возможности построения распределенных систем, т.е. масштабирование вверх.

С другой стороны, ОС QNX Neutrino сохраняет свои ключевые свойства даже в минимальных конфигурациях, что позволяет встраивать ее в специализированные целевые

платформы, обладающие ограниченными ресурсами и особыми требованиями к габаритам, энергопотреблению, конструктивному исполнению, а также к вибро-, термо-, и удароустойчивости.

Несмотря на широкую распространенность в нашей стране аппаратной платформы x86, многие специализированные промышленные процессоры выполняются на платформах отличных от x86. Это может быть связано с повышенными требованиями к энергопотреблению и радиационной стойкости. В связи с этим, операционная система должна поддерживать основные аппаратные платформы (MIPS, ARM, PowerPC, x86 и др.) и обладать возможностью адаптации для функционирования на специализированной оборудовании.

В качестве примера рассмотрим методологию адаптации ОС QNX Neutrino для функционирования на платформе «Мультикор» МС-12, являющейся серией отечественных однокристальных сигнальных контроллеров. ИМС контроллера имеет двуядерную архитектуру на базе MIPS-совместимого RISC-ядра и оригинального масштабируемого DSP-ядра [1]. Отладочный комплект МС-12ЕМ предназначен для освоения аппаратно-программных средств сигнального микроконтроллера 1892ВМ3Т (МС-12) и содержит следующую аппаратуру:

- Микросхему МС-12 в корпусе PQFP-240;
- Статическую память (SRAM) ёмкостью 1Мбайт;
- Два банка динамической памяти (SDRAM) каждый ёмкостью по 64Мбайт;
- Встроенный источник питания;
- Приёмопередатчик канала RS-232, подсоединенный к порту UART МС-12;
- Адаптер связи параллельного порта IBM PC (EPP) с JTAG портом МС-12.

Наличие отладочного комплекта позволяет эффективно производить адаптацию ОС QNX Neutrino для функционирования на платформе «Мультикор». Конечной целью этого процесса является создание BSP – пакета поддержки платы, который должен обеспечить старт системы на целевой платформе «Мультикор» и предоставить модули поддержки внешнего оборудования, например, последовательного порта UART. Схема разработки и отладки BSP для платы МС-12ЕМ представлена на рисунке 2.

Рис. 2. Схема разработки и отладки BSP для платы МС-12ЕМ

В приведенной схеме используется механизм кросс-платформенной разработки программного обеспечения для ОС QNX. В настоящий момент для QNX Neutrino версии 6.3, существует возможность ведения разработки под следующими операционными системами:

- Windows;
- Linux;
- Solaris;
- QNX.

Это позволяет использовать многочисленный инструментарий, разработанный для этих операционных систем (редакторы, файловые менеджеры, JTAG-средства, интегрированные среды и т.д.).

В качестве операционной системы для инструментальной машины выбрана ОС Linux, поскольку для нее существует инструментальное средство mdb для работы с JTAG и комплект разработчика QNX Momentics Professional Edition Linux Host. Для организации последовательной консоли может использоваться как инструментальный ПК, так и отдельный ПК, с установленной терминальной программой (minicom, qtalk и др.). Основными задачами, выполняемыми на инструментальной системе, являются:

- Компиляция и сборка необходимых модулей BSP;
- Низкоуровневая отладка и загрузка образа ОС на целевую платформу МС-12 через JTAG;
- Отладка модулей BSP на уровне ОС.

Структура типичного пакета поддержки платы представлена на рисунке 3. Комплект BSP представляет собой упорядоченное дерево каталогов и файлов и содержит следующие компоненты:

- Исходные тексты модулей BSP;

– Файлы построения образов ОС;
– Необходимые библиотеки и заголовочные файлы;
– Документацию.

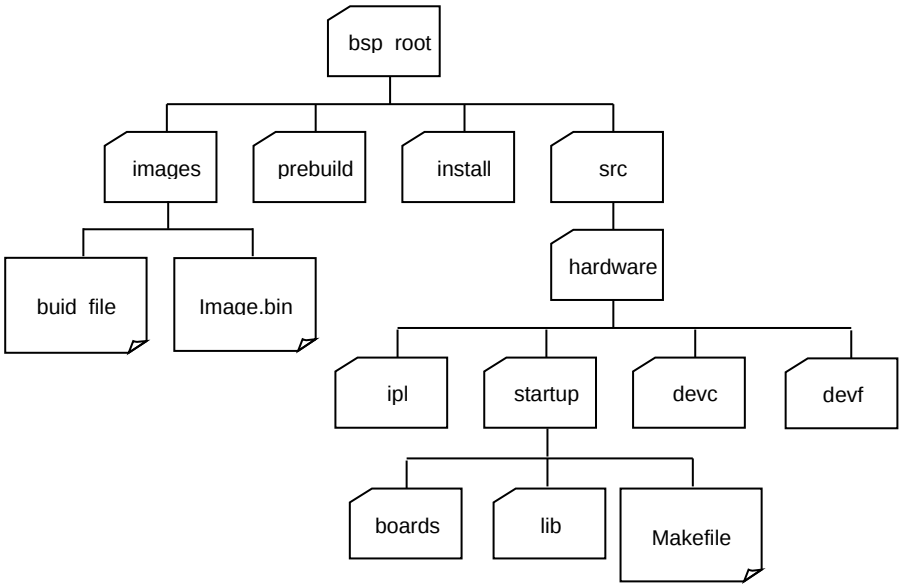


Рис. 3. Структура типичного пакета поддержки платы BSP

Важнейшими модулями BSP, обеспечивающими старт образа ОС QNX Neutrino на целевой платформе, являются модуль первоначального загрузчика IPL и стартовая программа образа startup. Задачей модуля IPL является загрузка образа ОС и передача управления первой программе образа startup. При этом, могут обеспечиваться альтернативные варианты загрузки, например, загрузка по последовательному каналу, загрузка из on-board flash или через сеть Ethernet по протоколу tftp. Реализация модуля IPL может различаться для «холодного» и «горячего» старта системы. В случае «холодного» старта первая инструкция IPL находится по вектору сброса процессора. При этом IPL должен произвести начальную инициализацию оборудования, и лишь затем приступать к загрузке образа ОС. При «горячем» старте первой программой, получающей управление, является ROM-монитор или BIOS (в случае x86). Эта программа выполняет основную инициализацию оборудования и предоставляет дополнительные сервисы, например, загрузка образа в ELF-формате в память ОЗУ. В этом случае, QNX-загрузчик IPL может быть упрощенным или вовсе отсутствовать.

Большая часть кода IPL – это вызовы функций библиотеки IPL Library. В общем случае, IPL выполняет следующие действия:

– Производит начальную инициализацию оборудования;
– Загружает образ ОС QNX Neutrino (обычно несколько вариантов загрузки);
– Осуществляет поиск загруженного образа по сигнатуре;
– Осуществляет подготовку к старту образа, копируя модуль startup в ОЗУ;
– Передает управление модулю startup.

Модуль startup – это самая первая программа в образе ОС QNX Neutrino. Она получает управление от IPL и выполняет следующие действия:

– Производит дополнительную инициализацию оборудования;
– Заполняет специальную структуру данных – системную страницу QNX (system page);
– Реализует в своем коде специальные процедуры ядра callout-ы;
– Передает управление микроядру и менеджеру процессов – модулю procnto.

Код модуля startup базируется на вызовах функций библиотеки Startup Library и, также как IPL, реализуется на высокоуровневом языке С. Это существенно упрощает задачу системным программистам, адаптирующим ОС QNX Neutrino для целевой платформы. Модуль startup должен заполнить системную страницу QNX, содержащую информацию о конфигурации оборудования. Эта информация в дальнейшем используется ядром операционной системы, а также системными и прикладными задачами. Структура системной страницы может различаться в зависимости от аппаратной платформы, и, как правило, содержит следующие данные:

– Информация об отладочных устройствах, например, в качестве таких устройств могут использоваться последовательные порты;
– Информация о конфигурации памяти в системе (физические адреса блоков ОЗУ и др.);

– Информация о системных таймерах;
– Информация о контроллерах прерываний (номер каскадного прерывания и др.);
– Информация о характеристиках ЦПУ (частота, работа с кэш и др.).

Как уже указывалось, адаптация ОС QNX Neutrino для функционирования на различных аппаратных платформах достигается без модификации ядра операционной системы. В этом состоит основное отличие QNX от других распространенных ОС, обладающих монолитным ядром, например, ОС Linux, в которой для обеспечения запуска требуется серьезная модификация и пересборка ядра. Для того чтобы адаптировать ОС QNX Neutrino достаточно, чтобы ядро поддерживало аппаратную архитектуру целевой платформы. В случае контроллера «Мультикор» МС-12, такой архитектурой является MIPS32 Little Endian. Функционирование ядра ОС на специфической аппаратуре достигается за счет использования технологии callout-ов.

Callout-ы микроядра – это специальные процедуры, вызываемые микроядром ОС QNX Neutrino, для обеспечения функционирования операционной системы на целевой платформе с определенной аппаратной конфигурацией. Основные характеристики callout-ов:

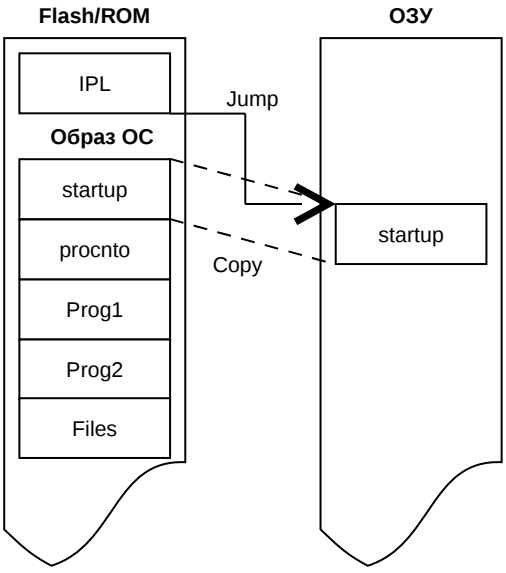
– Определяются в коде модуля startup;
– Код реализуется на языке ассемблера целевой платформы;
– Код callout-ов должен быть перемещаемым в памяти (Position-independent), так как ядро копирует эти процедуры в ОЗУ.

ОС QNX Neutrino поддерживает множество типов callout-ов, с помощью которых можно учесть практически все особенности оборудования. Некоторые из них являются общими для целевой аппаратной платформы, другие требуют специфической реализации. Наиболее распространенны следующие типы callout-ов:

– Callout-ы отладочных интерфейсов – позволяет организовать вывод отладочной текстовой информации, например, на консоль или в последовательный порт;
– Callout-ы, обеспечивающие интерфейс доступа к аппаратным таймерам - используется ядром для отсчета системного времени и обслуживания программных таймеров QNX;
– Callout-ы, обеспечивающие интерфейс к аппаратным контроллерам прерываний – реализуют основные функции работы с аппаратными прерываниями (маскирование, демаскирование, определение источника прерываний и др.);
– Callout-ы, управляющие кэш ЦПУ – используются для оптимизации производительности процессора;
– Callout-ы, обеспечивающие управление энергопотреблением (Power Management) – используются для контроля энергопотребления в системах, критичных к этому показателю.

Процесс загрузки ОС QNX Neutrino различается в зависимости от того, где располагается образ ОС, в устройстве с линейно-адресуемой памятью или в устройстве со страничной адресацией, так как это показано на рисунке 4.

Загрузка образа из линейно-адресуемого пространства (Flash/ROM etc)



Загрузка образа со странично-адресуемого пространства (HDD, Serial Port, Ethernet etc)

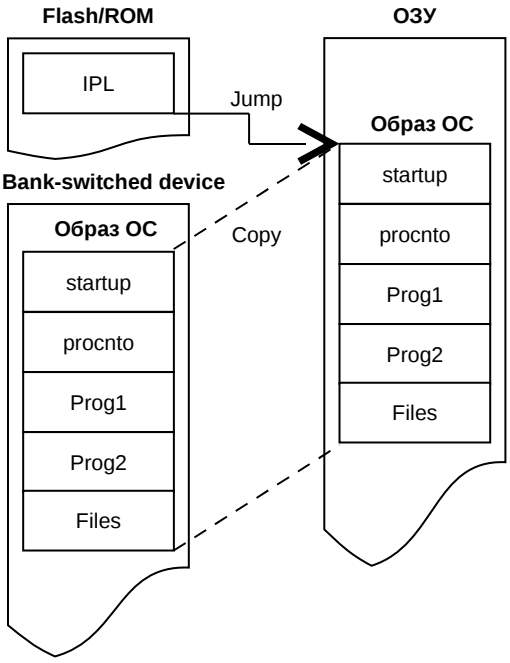


Рис. 4. Загрузка образа ОС QNX Neutrino из устройств с линейной и страничной адресацией.

В некоторых случаях схема загрузки ОС QNX Neutrino может отличаться от представленной на рисунке 4. Так при использовании отладочного комплекта МС-12ЕМ платформы «Мультикор», целесообразно загрузку образа в ОЗУ производить через JTAG-интерфейс с использованием программного средства mdb. При этом отсутствует загрузчик IPL, а образ в ELF-формате загружается через JTAG непосредственно в ОЗУ платы, затем управление передается программе startup, которая отвечает за инициализацию оборудования.

Кроме различных вариантов загрузки, ОС QNX Neutrino поддерживает компрессию и декомпрессию образа. Сжатие образа существенно уменьшает его размер, декомпрессию должна осуществлять программа startup. Также поддерживаются различные форматы образа ОС: ELF, SREC и двоичный (binary). Образ ОС QNX Neutrino можно рассматривать как файловую систему «только для чтения» (read only), к объектам которой могут обращаться обычные утилиты, например, ls, cat и др [2]. Конфигурирование образа ОС осуществляется с помощью специального файла построения (buildfile). Этот файл содержит всю необходимую информацию о способе загрузки и типе образа, о включаемых процессах, динамических библиотеках и файлах [3].

Многие промышленные процессорные платы имеют на своем борту flash-носитель, который предназначен для хранения программ (в т.ч. операционной системы) и данных (файлов). Поэтому важным элементом построения встраиваемых систем является обеспечение работы ОС с flash-носителем. Полноценную работу с flash должен обеспечивать драйвер flash файловой системы. Структура драйвера flash файловой системы ОС QNX Neutrino представлена на рисунке 5.

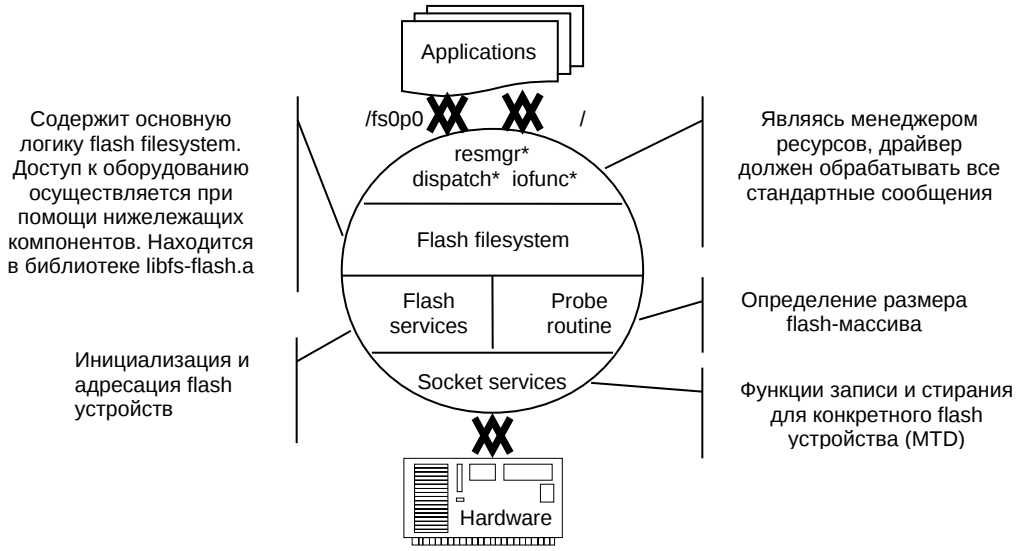


Рис. 5. Структура драйвера flash файловой системы ОС QNX Neutrino

После того как мы проанализировали необходимые действия, связанные с запуском ОС QNX Neutrino на целевой платформе, рассмотрим инструментальные средства разработки и отладки, применяющиеся для построения встраиваемых систем на базе ОС QNX Neutrino. Перечень этих средств достаточно широк и включает в себя:

- Инструментарий кросс-платформенной разработки, включающий компилятор GCC, набор утилит, работы с образами ОС и другие средства;
- Пакеты поддержки плат – Board Support Packages, BSP’s;
- Комплекты разработки драйверов – Driver Development Kits, DDK’s, позволяющие вести разработку драйверов различных устройств;
- Комплекты разработки технологий – Technology Development Kits, TDK’s, позволяющие вести разработку приложений на основе базового набора инструментов или модифицировать ПО под конкретные задачи;
- Интегрированная среда разработки – Integrated Development Environment (IDE) QNX Momentics.

При разработке пакета поддержки платы зачастую бывает необходимо реализовать драйверы устройств, находящихся на борту платы. Комплекты DDK’s позволяют эффективно разрабатывать драйверы устройств из широкого спектра:

- Network DDK – создание драйверов сетевых устройств;
- Graphics DDK – создание драйверов для видеоадаптеров;
- Audio DDK – создание драйверов для аудио-устройств;
- Character DDK – создание драйверов для символьных устройств (последовательные, параллельные порты и др.);
- Input DDK – создание драйверов для устройств ввода (клавиатуры, мыши, touch-screens и др.);
- USB DDK – создание драйверов USB-устройств;
- Printer DDK – создание драйверов для принтеров.

Каждый DDK содержит примеры драйверов в исходных текстах, необходимые библиотеки, заголовочные файлы и подробную документацию. По сути, DDK предоставляет программный каркас абстрактного драйвера устройства, а задачей программиста является реализация аппаратно-зависимой части драйвера.

Пакеты разработки технологий TDK's предоставляют необходимый инструментарий для разработки и модификации программного обеспечения в рамках предлагаемых технологий и включают в себя следующие комплекты:

- «Flash File System & Embedding Kit» - файловая система во флэш-памяти и инструменты встраивания;
- «Extended Networking» - расширенные сетевые технологии;
- «Symmetric Multiprocessing (SMP)» - разработка приложений для симметричной мультипроцессорной обработки данных;
- «3D Graphics» - разработка трехмерных графических приложений;
- «Multi-media framework» - комплект для разработки мультимедийных средств;
- «Web Browser» - разработка приложений для веб-браузера;
- «Critical Process Monitoring» - мониторинг ключевых процессов;

Интегрированная среда разработки IDE Momentics базируется на открытой платформе Eclipse и является полнофункциональной масштабируемой инструментальной средой, обладающей следующими возможностями:

- Организация ресурсов (проекты, каталоги, файлы);
- Редактирование ресурсов;
- Совместная работа над проектом;
- Компиляция, выполнение и отладка программ на целевой платформе;
- Построение образов ОС QNX Neutrino для встраиваемых систем;
- Анализ производительности и профилирование программных систем;
- Расширение функциональности за счет добавления собственных плагинов и перспектив.

Весьма полезным свойством среды IDE Momentics является возможность организации прозрачного взаимодействия инструментальной и целевой платформы на этапах разработки и отладки программного обеспечения. При этом, на физическом уровне, соединение может быть организовано как через сеть Ethernet, так и через последовательный канал.

Резюмируя вышесказанное можно отметить, что применение пакетов поддержки плат QNX BSP позволяет выполнять адаптацию операционной системы на целевые платформы без модификации ядра. При этом использование особенностей аппаратуры достигается реализацией специальных процедур – callout-ов. Поддержка ядром QNX архитектуры MIPS32 позволяет осуществить адаптацию операционной системы для платформы «Мультикор» МС-12 с применением стандартных средств разработки и отладки. Существующий широкий спектр кросс-платформенных инструментальных средств, включающий комплекты разработки драйверов, пакеты поддержки плат, интегрированную среду разработки IDE, а также наличие прототипных или отладочных плат платформы «Мультикор» позволяют значительно сократить сроки разработки конечных приборов.

ЛИТЕРАТУРА

1. Солохина Т.В., Петричкович Я.Я. Микросхемы базовых серий «Мультикор». Сигнальный микроконтроллер 1892ВМ2Т (МС-24) // Chip News Инженерная Микроэлектроника – М., 2005 - №2. С. 20 -30.
2. Rob Krten. The QNX Cookbook. Recipes for Programmers. – PARSE Software Devices, edited by C. Herborth. – 2003. – P. 335-371.

3. Зыль С.Н. Операционная система реального времени QNX: от теории к практике. – СПб., БХВ-Петербург, 2004. – С. 192.