Операционная система QNX4

Руководство пользователя

Оглавление

Об этой книге

Руководство пользователя ОС QNX 4.25 предназначается как для администраторов системы, так и для конечных пользователей. Оно содержит информацию о том, как устанавливать, конфигурировать и сопровождать операционную систему QNX.

Следующая таблица может ускорить поиск необходимой информации:

Если вы хотите:	**Смотрите главу:**
Проверить необходимые условия для установки программного обеспечения	**Базовая установка**
Установить программное обеспечение на жесткий диск	"
Модифицировать процедуру начальной загрузки QNX	"
Установить начальное рабочее окружение	"
Узнать о модулях QNX и услугах системы	**Построение образа ОС**
Назначить начальные приоритеты системным процессам	"
Выбрать и изменить сервисные средства, встраиваемые в ОС	"
Построить пользовательский образ ОС	"
Монтировать и демонтировать разделы файловой системы	**Установка файловой системы**
Удалить блок-ориентированные устройства и повторно установить драйверы файловой системы	"
Использовать префиксы или символические связи	"
Установить файловую систему DOS	"
Запустить драйверы последовательных и параллельных устройств	**Подключение символьных устройств**
Назначить порт параллельному или последовательному драйверу	"
Сконфигурировать адреса и вектора прерываний аппаратуры ввода/вывода	"
Сконфигурировать модем, чтобы отвечать на поступающие вызовы	"
Решить задачу тестирования последовательного соединения	"
Получить информацию об установленных лицензиях	**Лицензирование**
Добавить и активизировать новые лицензии	"
Перенести лицензию из одного узла в другой	"
Спланировать свою сеть	**Создание сети**
Установить сетевые платы и драйверы	"
Назначить логические идентификаторы узлу и сети	"
Сконфигурировать сервер начальной загрузки	"
Сконфигурировать компьютер, оснащенный ПЗУ	"

Если вы хотите:	**Смотрите главу:**
начальной загрузки	
Сконфигурировать компьютер, чтобы начальная загрузка выполнялась с собственного жесткого диска	"
Выполнить диагностику сети	"
Добавить новых пользователей в систему	**Настройка учетных записей пользователей**
Установить рабочее окружение для пользователей	"
Управлять доступом к ресурсам системы	"
Поддерживать учетные записи пользователей	"
Изменить информацию в базе данных паролей	"
Просмотреть и изменить разрешения пользователя	"
Установить систему учета информации о системных ресурсах и регистрации событий	"
Выбрать тип терминала	**Установка терминалов**
Изменить характеристики терминалов	"
Создать пользовательский файл terminfo	"
Использовать совместно ресурсы принтера	**Печать со спулингом**
Установить и управлять сервисные функции спулера	"
Установить средства организации очередей данных	"
Сконфигурировать очереди, фильтры и получателей	"
Запустить, запросить, приостановить или аннулировать задание в спулере	"
Спланировать свой алгоритм резервирования	**Создание резервных копий**
Выбрать необходимые средства резервирования и архивных форматов	"
Сжать и поместить в архив файлы	"
Извлечь файлы из архива	"
Поддерживать файлы	**Восстановление дисков и файлов**
Проверить целостность данных	"
Восстановить данные	"
Настроить систему, которая не может выполнить начальную загрузку	"
Сообщить о замеченных ошибках	**Как связаться с QSSL**
Зарегистрировать свой сервис-план и программное обеспечение	"
Обновить свое программное обеспечение	"
Получить свободно-распространяемое программное обеспечение	"
Получить техническую поддержку	"
Найти информацию об обучении	"
Найти информацию о Мировой сети продаж	"

Глава 1. Базовая установка

Этот раздел охватывает следующие темы:

- Установка QNX на жесткий диск
- Установка дополнительного программного обеспечения
- Файл инициализации системы
- Использование файла инициализации системы
- Часовые пояса и часы реального времени

Установка QNX на жесткий диск

Перед тем, как вы приступите к установке, обязательно прочитайте инструкции по установке, а также *Release Notes* (Редакционные замечания), которые поступили с вашим программным обеспечением. *Release Notes* содержат важную информацию, которая может повлиять на ваш выбор способа установки программного обеспечения.

Вам также следует ознакомиться с книгой *Системная архитектура*, которая объясняет основы построения и функционирования QNX. Мы считаем, что вы имеете представление об операционной системе на данном уровне.

Требуемые ресурсы

Потребность в ресурсах зависит от того, какие продукты вы желаете устанавливать. Например, для того, чтобы установить базовую ОС для системы разработки, понадобится, по меньшей мере, 14 Мбайт свободного пространства на диске. Для более полной информации обратитесь к инструкции по установке, которая поставляется с каждым продуктом.

QNX поддерживает широкий спектр жестких дисков, графических адаптеров и т.д. Подробную информацию можно найти на QUICS (используя grep -i *шаблон* find-ls). Про QUICS можно прочитать в главе **Как связаться с QSSL.**

Примечание. Ваша целевая система, вероятно, потребует гораздо меньше дискового пространства (если она вообще будет иметь диск!), ОЗУ и т.д., чем ваша система разработки. Так как QNX является модульной ОС, вы можете легко сконфигурировать вашу целевую систему так, чтобы она содержала только действительно необходимые на стадии выполнения модули.

Дистрибутивные носители

Продукты QNX могут поставляться на одном из двух видов носителей:

- на дискетах;

- на CD-ROM.

В обоих случаях для установки QNX на жесткий диск вы будете использовать специальную программу установки. Эта программа создает структуру каталога, копирует программное обеспечение на жесткий диск и строит загрузочный образ ОС в соответствии с заданной в процессе установки конфигурацией.

Во время процедуры установки вам может быть предложено, чтобы вы подтвердили вашу аппаратную конфигурацию. Прежде чем начать установку, убедитесь, что вы располагаете достоверной информацией относительно ваших аппаратных средств:

- объем жесткого диска и его раздела (разделов);
- тип контроллера жесткого диска (IDE, Adaptec 2940 SCSI и т.п.);
- тип и производитель сетевой платы (например, NE2000, Ethernet Novell).

Дистрибутив на CD-ROM

Если ваше программное обеспечение QNX было поставлено на CD, то процедура установки будет совсем несложной:

- вставьте загрузочную дискету QNX в дисковод;
- вставьте CD в привод CD-ROM;
- перезапустите компьютер.

После того, как ваш компьютер выполнит начальную загрузку, следуйте инструкциям на вашем экране.

Дистрибутив на дискетах

Чтобы установить QNX с дискет, выполните следующие шаги:

1. Поместите загрузочную дискету QNX в дисковод и перезагрузите компьютер.

   Вы должны видеть вращающуюся стрелку или ряд точек в верхнем левом углу экрана, затем графические символы "QNX", приветственное сообщение и приглашение (промпт) командной строки.

Примечание. Если вы захотите получить информацию о возможных опциях утилиты install, то, прежде чем вы продолжите, введите на приглашение командной строки: use install.

2. Чтобы перенести программное обеспечение с дискет на жесткий диск, выполните следующую команду:

   ```
   install
   ```

Следуйте инструкциям на экране, чтобы подготовить жесткий диск к загрузке QNX;

3. Произведите перезагрузку вашего компьютера с загрузкой с жесткого диска.
   Если все файлы с дискет были установлены, то следует вынуть из дисковода дискету и перезагрузить компьютер с жесткого диска. Теперь должна загрузиться QNX. Вам следует войти в систему как root;

4. Теперь вы готовы к установке дополнительного программного обеспечения.

Установка дополнительного программного обеспечения

Существует три возможных способа установки дополнительных программных продуктов QSSL:

- с CD-ROM;
- с дискет;
- с QUICS.

Если с CD-ROM ...

Если вы имеете продукты QNX на CD-ROM, то вы можете установить любой продукт, находящийся на CD-ROM, при наличии у вас соответствующей лицензии.

Во время данной процедуры вы должны будете ввести информацию о лицензии, которая включена в ранее приобретенный программный продукт.

Примечание. Чтобы выполнить установку с CD-ROM, вы должны запустить Photon.

1. Вставьте компакт-диск в привод CD-ROM;
2. Запустите установку пакета (программ);
3. Выберите (мышью) из списка доступных программных продуктов те, которое вы хотите установить;
4. После появления соответствующих запросов вводите соответствующую лицензионную информацию для каждого продукта, который вы устанавливаете;
5. Следуйте инструкциям на экране.

Если с дискет ...

Все дополнительные программные продукты QNX, поставляемые QSSL, поступают в сжатом формате. Вы можете установить эти продукты с помощью утилиты /etc/install, поставляемой с ОС, - следуйте инструкциям, указанными ниже.

Вы можете также использовать этот метод для того, чтобы установить пакеты от третьих фирм, но всегда, прежде чем приступить к установке, ознакомьтесь с инструкциями, которые поступают с каждым пакетом.

Примечание. Перед тем как приступить к установке программного обеспечения, вы должны войти в систему как суперпользователь (root) и должен быть запущен драйвер гибких магнитных дисков. Чтобы проверить, что драйвер запущен, введите команду:

```
sin ver
```

Если "Floppy" отсутствует в столбце NAME, то введите:

```
Fsys.floppy &
```

Процедура установки

Чтобы установить дополнительное программное обеспечение, выполняйте следующие шаги:

1. вставьте первую дискету программного продукта в дисковод;
2. в командной строке наберите:

```
cd /
/etc/install [drive]
```, где

drive - устройство, с которого будет производиться установка. По умолчанию это локальный гибкий диск (/dev/fd0).

Следуйте инструкциям на экране, и программное обеспечение будет установлено на жестком диске.

Если из QUICS ...

Обновленные версии большинства программных продуктов помещаются на QUICS. Для установки обновленных версий продуктов: перепишите требуемый архив программного обеспечения, затем, используя утилиту /etc/install (поставляемую с ОС), распакуйте и установите архив на жесткий диск:

```
install -u update
```, где

update - имя архивного файла.

Более подробная информация об утилите **install** содержится в книге *Описание утилит*.

Файл инициализации системы

Что происходит при загрузке QNX

При загрузке QNX в основную оперативную память загружается образ, состоящий из нескольких процессов QNX. Первый процесс в образе - это процесс начальной загрузки boot, который выполняет инициализацию в реальном режиме и затем переводит компьютер в защищенный режим.

Второй процесс в образе - Менеджер процессов (Proc32), который содержит Микроядро. Менеджер процессов выполняет инициализацию процессора, а затем выполняет диспетчеризацию выполнения всех включенных в состав образа процессов.

Последним процессом в образе является утилита sinit. Утилита sinit инициирует вторую фазу системной инициализации, запуская командный интерпретатор, который выполняет команды из файла. Этот файл - файл инициализации системы (sysinit) - содержит команды, которые устанавливают сервисные средства для данного компьютера. Он является стандартным командным файлом командного интерпретатора, который выполняется подобно любому другому командному файлу с той лишь разницей, что его нельзя прервать.

Возможность запуска системных сервисных процессов после начальной загрузки системы является одним из преимуществ модульной архитектуры QNX. Загружаемый образ обычно содержит только несколько обязательных важнейших сервисных процессов, необходимых для запуска всех остальных требуемых сервисных процессов.

Во время выполнения утилита sinit сначала определяет, был ли образ (частью которого является сама утилита sinit) загружен с диска или по сети. Если образ был загружен с диска, то sinit выясняет, является ли начальная загрузка основной или альтернативной. Для получения подробной информации смотри главу **Построение образа ОС.**

Примечание. Вы можете, при желании, выбрать альтернативную начальную загрузку путем нажатия клавиши **Esc** после появления на экране соответствующей подсказки начального загрузчика.

Во время основной загрузки sinit пытается загрузиться из файла /etc/config/sysinit.*узел*. Во время альтернативной начальной загрузки (только с локального диска) sinit пытается выполнить файл /etc/config/altsysinit. Если же выполнить любой из этих файлов невозможно, то sinit пытается выполнить файл /etc/config/sysinit.

Если sinit не может найти или открыть файл инициализации системы, то она завершается без инициализации системы. Если же файл успешно открыт, то sinit заменяется командным интерпретатором (/bin/sh) и передает ему имя открытого файла.

Настройка файла инициализации системы

Любой файл инициализации системы является простым ASCII-файлом, который можно редактировать, используя текстовый редактор (например, vi). Каждый из трех вышеупомянутых типов файлов имеет определенное назначение:

- `sysinit.`*узел* - файл для конкретного узла (где *узел* - значение между 1 и числом узлов в вашей сети);
- `sysinit` - файл по умолчанию, если sinit не может найти файл sysinit.*узел*;
- `altsysinit` - файл, который выполняется при загрузке альтернативной ОС с локального (местного) диска.

Файл sysinit.*узел*

Именно этот файл нужно редактировать, если вы желаете настроить вашу систему. Он содержит заказные команды, необходимые для того, чтобы установить окружение и сервисные средства на конкретном компьютере. Каждый узел в сети может иметь собственную настройку.

Файл sysinit.*узел* всегда создается программой install. Первоначальное содержание файла отражает параметры, определенные в процессе установки системы и дополнительного программного обеспечения с помощью утилиты install.

Настройка узлов

Чтобы настроить компьютеры (рабочие станции), вам необходимо иметь файл sysinit.*узел* на загрузочном сервере, который выполняет загрузку настраиваемого узла.

Вы можете для начала взять файл sysinit.*узел* для другого узла с аналогичной конфигурацией, либо вы можете просто сделать копию файла по умолчанию sysinit, который затем можно модифицировать:

```
cp /etc/config/sysinit /etc/config/sysinit.узел
```

Запомните, что суффикс *узел* должен быть логическим номером (ID) настраиваемого узла.

Примечание. Если вы измените логический ID узла, то компьютер будет искать файл sysinit.*узел*, который соответствует его новому логическому ID.

Файл sysinit

Файл sysinit выполняется при отсутствии файла sysinit.*узел*.

Этот файл автоматически создается в процессе установки ОС. Он поставляется, как обобщенный файл, который должен быть способен загрузить *любую* машину, и поэтому не рекомендуется делать какие-либо изменения в этом файле sysinit file.

Файл altsysinit

Этот файл служит страховкой на тот случай, когда в результате изменений в файле sysinit.*узел* вход пользователя в систему стал невозможен.

Файл altsysinit выполняется только в том случае, если вы выбрали альтернативную загрузку при загрузке с локального диска (то есть нажали клавишу **Esc**, когда на экране появилось соответствующее приглашение).

Файл altsysinit должен всегда содержать последнюю рабочую копию файла sysinit.*узел* для компьютера, который загружается с локального жесткого диска. Поэтому, прежде чем сделать любые изменения в рабочем файле sysinit.*узел*, которые могли бы помешать загрузке с жесткого диска, вам следует скопировать файл /.boot в файл /.altboot и скопировать файл sysinit.*узел* в файл altsysinit:

```
cp /.boot /.altboot
cp /etc/config/sysinit.узел /etc/config/altsysinit
```

Рекомендуется обновлять файлы /.altboot и altsysinit после успешного изменения файла sysinit.*узел*.

Использование файла инициализации

Базовые сервисные средства

Содержимое файла sysinit каждого компьютера отражает состав аппаратных средств этого компьютера и сервисные функции, которые он должен обеспечивать. Ниже описан основной состав сервисных средств, которые имеются в большинстве файлов sysinit.

Примечание. Если различные компьютеры будут использовать общий набор команд, то вы можете поместить команды в отдельном командном файле и вызывать его выполнение с помощью директивы "." (точка) командного интерпретатора. Например, вы создали файл с именем techies, содержащий команды, которые должны использоваться всеми компьютерами в техническом отделе вашего предприятия. Этот файл может быть вызван из файла sysinit.*узел* каждого компьютера в этом отделе следующей командой:

```
. /etc/config/techies
```

Команды, специфичные для данного узла, добавляют после этой "точечной" строки. Дополнительную информацию о "точечной команде" смотрите в описании командного интерпретатора sh в книге *Описание утилит*.

Установка часовых поясов

Следующие командные строки позволяют установить часовой пояс (в этом случае EST) и получить время с часов реального времени. Эти две строки должны быть первыми в файлах инициализации компьютеров, которые загружаются с жестких дисков. Имейте в виду, что строка rtc является необязательной для компьютера, загружаемого через сеть

```
export TZ=EST5EDT
rtc hw
```

Для получения более полной информации смотрите в конце этой главы раздел "**Часовые пояса и часы реального времени**"

Запуск драйверов устройств

Следующие командные строки запускают Менеджер устройств (Dev) и консольный драйвер (Dev.ansi) с восемью виртуальными консолями, затем дают указание командному интерпретатору снова начать стандартный ввод/вывод на новом консольном устройстве:

```
Dev &
Dev.ansi -n 8 &
reopen //0/dev/con1
```

Следующие строки запускают последовательный драйвер Dev.ser, который ищет COM1 и COM2, и параллельный драйвер Dev.par, который ищет LPT1 и LPT2. Эти драйверы завершаются, если они не могут найти необходимые аппаратные средства.

```
Dev.ser &
Dev.par &
```

Если вы запустили Dev.ser, то вы можете, используя утилиту stty, изменить конфигурацию последовательного порта. Например, следующая строка изменяет скорость передачи в бодах на 57600:

```
stty baud=57600 </dev/ser1
```

Запуск эмулятора операций с плавающей запятой

Если ваши программы используют операции с плавающей запятой, а компьютер не имеет устройства, выполняющего операции с плавающей запятой (FPU), то вам необходимо запустить эмулятор операций с плавающей запятой (эмулятор математического сопроцессора типа 80x87):

```
emu87 &
```

Загрузка карты сети

При загрузке узла сети необходимо выполнить утилиту netmap, которая загружает карту сети, информируя Менеджер сети (Net) о соответствии физических идентификаторов узлов логическим номерам узлов и сетей. Следующую команду следует включить в файл sysinit.*узел*, даже если узел в настоящий момент не работает в сети (она не влияет на работу автономного компьютера):

```
netmap -f
```

Имейте в виду, что эта команда включается в стандартные файлы инициализации системы, поставляемые фирмой QSSL.

Запуск сервера имен

Каждый компьютер в сети должен иметь доступ к серверу глобальных имен. Вы должны запустить сервер имен на том компьютере, который не перезагружается слишком часто:

```
nameloc &
```

Утилита nameloc периодически связывается с каждым узлом в непрерывном цикле, начиная с узла 1 и заканчивая узлом *n*, где *n* - число установленных сетевых лицензий. Хорошо бы запустить nameloc на двух узлах, чтобы обеспечить избыточность на случай, если один узел, выполняющий nameloc, выйдет из строя.

Примечание. Не рекомендуется запускать nameloc на каждом узле в сети, так как это вызовет излишнюю нагрузку на сеть. Для получения большей информации смотри описание утилиты nameloc в книге *Описание утилит*.

Запуск "демона" терминала

Следующая командная строка запускает login на первой консоли и инициализирует все другие консоли:

```
tinit -T /dev/con* -t /dev/con1 &
```

Несмотря на то, что она не является обязательной, почти при всех установках используется утилита tinit. Стандартно поставляемый файл sysinit уже содержит вызов утилиты tinit.

Дополнительные сервисные средства

Вы можете добавить несколько других сервисных функций в ваш файл sysinit. Вам следует добавлять эти команды только перед строкой, содержащей команду tinit. Следующие примеры иллюстрируют использование дополнительных сервисных средств. Имейте в виду,

что эти утилиты обычно поддерживают опции командной строки, изменяющие их поведение (эти опции описаны для каждой утилиты в книге *Описание утилит*).

Установка переменных окружения

Процессы, запущенные в файле sysinit, наследуют свои переменные окружения. Общий синтаксис определения переменной окружения следующий:

`export` *`var=value`*, где

var - имя переменной окружения (такой, как **TERM** для типа терминала или **TZ** для часового пояса), и *value* - устанавливаемое значение. Команда export описана в описании утилиты sh в книге *Описание утилит*.

Запуск драйвера дисковода гибких дисков

Чтобы запустить локальный драйвер гибких дисков, сначала запускают Менеджер файловой системы Fsys, а затем непосредственно драйвер, как показано ниже. Если файловая система QNX уже запущена локально (это будет иметь место, если компьютер выполнил начальную загрузку с диска), то Менеджер файловой системы будет уже запущен, и вам не нужно включать первую строку:

```
Fsys &
Fsys.floppy &
```

Если вы желаете иметь доступ к гибкому диску как к файловой системе QNX, то вы должны монтировать его так:

```
mount /dev/fd0 /fd0
```

Запуск файловой системы CD-ROM

Чтобы иметь доступ к файлам на CD, вам нужно запустить менеджер файловой системы Iso9660fsys, который поддерживает стандартную файловую систему CD-ROM так же, как расширение Rock Ridge:

```
Iso9660fsys &
```

Запуск файловой системы DOS

Если вам необходимо иметь доступ к гибким дискам DOS и разделам жесткого диска, вам нужно запустить файловую систему DOS (Dosfsys):

```
Dosfsys &
```

Запуск TCP/IP

Если вы хотите связаться с компьютером, не имеющим ОС QNX (например, доступ в Интернет), вы должны будете запустить администратор сокетов TCP/IP. Для получения подробной информации обратитесь к документации на пакет **TCP/IP для QNX**.

Запуск сервера cron

Если компьютер будет редко перезагружаться, то рассмотрите возможность запуска на нем сервера cron:

```
cron &
```

Запуск драйвера "мыши" (для консоли)

Некоторые текстовые приложения (например, vedit) позволяют вам использовать "мышь". Если используются эти приложения на консоли в текстовом режиме (то есть без Photon), то вы должны будете запустить менеджер Mouse. Поддерживается несколько типов "мышей". Следующая команда обнаружит и корректно запустит драйвер "мыши":

```
mousetrap start
```

Запуск Photon microGUI

В некоторых системах файл инициализации системы запускает для графических приложений оконную систему Photon. Для получения более полной информации смотрите главу по Photon в книге *Системная архитектура* QNX.

Поддержка национальных клавиатур

По умолчанию, QNX драйвер консоли работает с 101-клавишной клавиатурой в стандарте США. Тем не менее, в QNX поддерживается целый ряд конфигураций, соответствующих клавиатурам, наиболее часто используемых в мире. Если вам требуется выбрать конфигурацию с альтернативной клавиатурой, то используйте утилиту kbd.

Если вы выбрали не совместимую со стандартом США клавиатуру, то при установке QNX на компьютере, который загружается из собственного жесткого диска, соответствующая команда kbd будет добавлена в файл sysinit.*узел* компьютера. Если вы потом устанавливаете другие узлы, то вы должны будете включить эту команду в их файлы sysinit.*узел*.

Имейте в виду, что утилитой kedit вы тоже можете сгенерировать заказное (пользовательское) расположение символов на клавиатуре.

Часовые пояса и часы реального времени

Важно, чтобы при инициализации была установлена правильная дата, время и информация о часовом поясе. Эти установки должны быть сделаны в начале вашего файла sysinit. Программа install считает, что аппаратные часы выдают правильную дату и время, и запрашивает только информацию о часовом поясе.

Допустимые даты в системе QNX изменяются от января 1970 до января 2038. Внутреннее представление даты и времени достигает максимума при значении в 2038. Если ваша система будет эксплуатироваться после 2038, и если тогда для системы еще не будет иметься способа, чтобы расширить или изменить этот предел, то вы должны будете взять на себя специальную заботу о датах системы (для справки по этому вопросу свяжитесь с техническим персоналом QSSL).

Некоторые компьютерные BIOS и некоторые более ранние версии утилиты rtc QNX имеют трудности при обработке перехода между годом 1999 и годом 2000. Обратите внимание также, что при использовании этого способа при форме ввода в программу года двумя цифрами, старые программы могут прекратить работать в 2000 году и позже.

Для более полной информации относительно проблем 2000 года для текущих и предыдущих версий QNX найдите "2000 год" в QUICS.

В QNX используется Скоординированное Всемирное Время (UTC), которое часто называют временем по Гринвичу. Приложения и утилиты преобразовывают его в локальное время, используя информацию о часовом поясе из переменной окружения (**TZ**).

Примечание. Если в переменной окружения **TZ** значение часового пояса не установлено, то QNX считает, что ваше местное время - то же самое, что и Стандартное Восточное Время в Северной Америке с текущим Летним временем (EST5EDT). Если вы захотите передать файлы в другую систему, находящуюся в другом часовом поясе, то в этом случае даты в файлах окажутся смещенными на разницу между двумя часовыми поясами.

Установка часового пояса

Информация о часовом поясе должна быть введена перед установкой текущей даты и времени. Если часы реального времени в вашем компьютере были установлены на местное время, то QNX необходима информация о часовом поясе для того, чтобы получить UTC.

В следующем примере часовой пояс, а также правила изменения времени, устанавливаются для Восточного Стандартного Времени в Северной Америке:

```
export TZ=EST5EDT4,M4.1.0/3,M10.5.0/3
```

где:

`export` – команда командного интерпретатора, которая устанавливает переменную

окружения;
`TZ` – имя переменной;
`EST` – Восточное Стандартное Время;
`5` – UTC — 5;
`EDT` – Летнее время;
`4` – UTC — 4;
`M4.1.0/3` – первое воскресенье апреля в 3 часа до полудня;
`M10.5.0/3` – последнее воскресенье октября в 3 часа до полудня.

Для подробной информации относительно правил установки часового пояса смотри техническое замечание timezone в QUICS.

Получение даты и времени с часов реального времени

Если вы загружаетесь из диска, то вы должны выполнять правила установки часового пояса в вашем файле sysinit утилитой rtc, чтобы установить текущую дату и время из таймера. Следующие две командные строки выполняют это:

```
export TZ=EST5EDT4,M4.1.0/3,M10.5.0/3
rtc hw
```

Имейте в виду, что имеются два возможных способа установки часов реального времени в вашем компьютере:

- установка часов реального времени по UTC;
- установка часов реального времени по местному времени.

Мы рекомендуем, чтобы вы устанавливали время в часах реального времени по UTC. Но, если вы также работаете с операционными системами, которые устанавливают часы реального времени по местному времени (например, DOS), то вам следует использовать утилиту rtc с опцией **-l**:

```
rtc -l hw
```

Этот вызов rtc указывает, что часы реального времени устанавливаются по местному времени. Имейте в виду, что если вы используете местное время в часах реального времени, то вы должны вручную изменить значение часов реального времени при переходе из месного времени на летнее время и обратно.

Примечание. Если время на ваших аппаратных часах неправильное (возможно, была заменена батарея), то вам следует установить системное время, используя утилиту date, затем установить часы реального времени, используя утилиту rtc с опцией **-s**. Дополнительную информацию об этих утилитах смотрите в книге *Описание утилит*.

Если вы загружаетесь через сеть ...

Если вы загружаетесь через сеть, то компьютер унаследует время по UTC и значение часового пояса, хранящееся в переменной окружения, из его сервера начальной загрузки. Следовательно, вам не нужно вводить эту информацию в ваш файл sysinit.

Глава 2. Построение образа ОС

Этот раздел охватывает следующие темы:

- Введение
- Создание файла построения
- Где хранятся исполняемые файлы
- Выбор процессов для образа

Введение

QNX - модульная операционная система, состоящая из Микроядра и одного или более процессов, которые обеспечивают системные услуги. Например, процесс с именем Fsys обеспечивает обслуживание файловой системы, а процесс с именем Net обеспечивает обслуживание сети и т.п.

Когда вы формируете образ ОС, вы должны выбрать сервисные функции, которые вы хотите иметь доступными немедленно после загрузки образа, и включить в образ процессы, которые обеспечивают эти функции в построенной на заказ ОС. Этот образ создается утилитой buildqnx. Образ может быть загружен с диска программой-загрузчиком раздела QNX или загружен через сеть утилитой netboot.

Создание файла построения

Утилита buildqnx выдает двоичный файл образа, содержащий несколько процессов, которые перечислены во входном текстовом файле, называемом файлом "построения". Файлы построения содержатся в каталоге /boot/build, а файлы образа содержатся в каталоге /boot/images.

Вы можете создать образ для любого узла, вызывая непосредственно buildqnx или используя утилиту make и Makefile в каталоге /boot. Например, вы можете ввести любую из следующих команд, чтобы создать файл образа, именуемый hard.1, из образца файла построения hard.1:

```
cd /boot
buildqnx build/hard.1 images/hard.ata.1
```

ИЛИ

```
cd /boot
make b=hard.ata.1
```

Формат файла построения

Каждая программа, которую вы хотите включить в создаваемый образ, предусматривает использование трех строк в файле построения:

- первая строка - путь к программе, которую вы хотите включить;
- вторая строка начинается с символа $, за ним - начальное значение (по умолчанию - 1) размера динамической области памяти ("кучи"), а за размером "кучи" следует команда;
- третья строка *должна* быть пустой, чтобы отделить одну запись от другой.

Примечание. Соответствующие имена пути должны предполагать запуск в /boot.

Вот образец файла построения hard.ata.1:

```
sys/boot
$ 1 boot -v

sys/Proc32
$ 1 Proc32 -l 1

sys/Slib32
$ 1 Slib32

sys/Slib16
$ 1 Slib16

/bin/Fsys
$ 1 Fsys

/bin/Fsys.ata
$ 1 Fsys.ata

/bin/mount
$ 1 mount -p /dev/hd0 /dev/hd0t77 /

/bin/sinit
$ 1 sinit TERM=ansi
```

Установка размеров "кучи"

Число после символа $ - начальный размер "кучи". Значение "1" заставит размер "кучи" динамически расти до необходимой величины.

Установка приоритетов

Вы можете установить начальный приоритет процесса посредством задания вслед за значением размера "кучи" необязательного аргумента **:***nn* - нужный приоритет, задаваемый в диапазоне от 1 (самый низкий) до 30 (самый высокий). Например, следующая команда запустит драйвер гибких дисков (Fsys.floppy) с приоритетом 8:

```
/bin/Fsys.floppy
$ 1:8 Fsys.floppy
```

Где хранятся исполняемые файлы

Большинство исполняемых файлов системы QNX хранится в каталоге /bin. Некоторые элементы пригодны для использования только как часть образа начальной загрузки; они хранятся в подкаталогах в каталоге /boot.

Следующая таблица подводит итог этих условных обозначений:

Чтобы найти:	**Смотри в:**
Системные исполняемые файлы	/bin
Makefile образа ОС	/boot
Файлы построения для создания образа (их читает утилита make)	/boot/build
Файл образа ОС	/boot/images
Процессы системы, необходимые во время начальной загрузки	/boot/sys

Для получения более полной информации о том, где хранятся файлы в QNX, см. **Приложение 2**.

Выбор процессов для образа

Процессы, которые вы включаете в образ ОС, определяются по нескольким факторам. Вы можете сгруппировать образы в три класса:

- образы, загружаемые с диска;
- образы, загружаемые из сети;
- образы, загружаемые во встраиваемых системах.

Процесс boot (/boot/sys/boot) всегда автоматически включается в первую строку сгенерированного образа, даже если он явно не определен. Если вам необходимо задать опции для sys/boot, то вы должны добавить строку "sys/boot" в файл построения как первый процесс.

Для образов, которые загружаются с диска или через сеть, вы можете запустить большинство процессов после загрузки, размещая их вызов в файле инициализации системы, который QNX выполняет после загрузки (смотрите подраздел "Файл инициализации системы" в разделе "**Базовая установка**"). Это позволит вам сохранять образ загрузки небольшим (< 512K) и простым.

Примечание. Файл построения обычно не содержит Менеджера устройств /bin/Dev. Менеджер устройств и его драйверы обычно запускаются в файле инициализации системы *после* начальной загрузки системы.

Если файл инициализации системы не выполнен, то Менеджер устройств *не* будет запущен. В результате этого ваша клавиатура и системная консоль не будут функционировать.

Обязательные процессы

Когда вы строите образ, помните, что есть два обязательных процесса:

- Менеджер процессов/Микроядро (/boot/sys/Proc32);
- системная совместно-используемая (разделяемая) библиотека (/boot/sys/Slib32).

Чтобы запустить 16-битовые программы, вы должны включить 16-битовую совместно-используемую библиотеку /boot/sys/Slib16.

Дисковые образы

Для загрузки с жесткого диска вы должны включить:

- два обязательных системных процесса (первый - Proc32, а второй - Slib32);
- Менеджер файловой системы (Fsys);
- драйвер, необходимый для доступа к накопителю (Fsys.*driver*).

Примечание. Для получения дополнительной информации смотрите buildqnx, Fsys*, Proc32, Slib* и sinit в книге *Описание утилит*.

Файл Makefile для загрузки с диска

Файл Makefile для создания образов начальной загрузки (в каталоге /boot) содержит входные данные для создания настраиваемого образа загрузки с жесткого диска. Если вы строите образ ОС, вызывая утилиту make, и при этом не вводите новые значения параметров, будут использоваться значения, задаваемые по умолчанию.

Вот пример использования make, в котором создается образ загрузки с именем /boot/images/hard.1, используя файл построения с именем /boot/build/hard.1:

```
cd /boot
make b=hard.1
```

Имейте в виду, что вы можете создавать образы статически, вызывая утилиту buildqnx, или динамически, выполняя утилиту netboot. Для получения более подробной информации смотрите эти утилиты в книге *Описание утилит*.

Вот пример вызова buildqnx "вручную", в котором создается образ с именем /boot/images/ws32.ether1000 из файла построения с именем /boot/build/ws32.ether1000:

```
cd /boot
buildqnx build/ws images/ws32.ether1000
```

Копирование образа в /.boot

Построенный вами образ не станет образом новой начальной загрузки до тех пор, пока вы не скопируете его в файл /.boot. Однако, прежде чем вы это сделаете, вам следует сохранить текущий файл /.boot, скопировав его в файл /.altboot:

```
cp /.boot /.altboot
```

Примечание. Если по какой-либо причине ваш новый образ не работает правильно, вы можете нажать **Esc** после подсказки во время процесса начальной загрузки и загрузить файл /.altboot вместо файла /.boot.

Если вы выбираете образ альтернативной начальной загрузки, то обычная проверка файла /etc/config/sysinit.*узел* заменяется проверкой файла /etc/config/altsysinit. Вы должны гарантировать, что файл altsysinit содержит самую последнюю копию вашего работающего файла sysinit:

```
cp /etc/config/sysinit.узел /etc/config/altsysinit
```

Сетевые образы

Для загрузки через сеть вы должны включить:

- два обязательных системных процесса (первый - Proc32, а второй - Slib32);
- Менеджер сети (Net);
- драйвер, необходимый для доступа к аппаратным средствам сети (Net.*driver*).

Файл построения для сетевой загрузки

Следующий файл построения (/boot/build/ws.ether1000) содержит маркеры параметра, которые унаследуют их значения из сетевого окружения:

```
sys/Proc32
$ 1 Proc32 -l $(lnode) -D

sys/Slib32
$ 1 Slib32

sys/Slib16
$ 1 Slib16

/bin/Net
$ 1 Net -m $(netmap)
```

```
/bin/Net.ether1000
$ 1 Net.ether1000

/bin/sinit
$ 1 sinit -r //$(bnode)/  TERM=qnx TZ=$(TZ)

sys/Debugger32
$ 1 Debug
```

Маркер $(lnode) в команде Proc32 берет в качестве своего значения логический ID узла загружающегося компьютера. Логический ID узла определяется утилитой netboot на основе адреса Ethernet загружающегося компьютера и записи в файле /etc/config/netmap и затем передается утилите buildqnx.

Маркер $(netmap) в команде Net сообщает Net первоначальную карту сети сервера начальной загрузки с тем, чтобы загружающийся компьютер смог начать сетевую связь с ним.

Маркер $(bnode) в команде sinit представляет логический ID узла сервера начальной загрузки. Команда устанавливает корень файловой системы (/) загружающегося компьютера в корень сервера загрузки.

Значение маркера $(TZ) наследуется от сервера начальной загрузки. Команда устанавливает информацию зоны времени загружающегося компьютера, чтобы она совпадала с текущей установкой переменной окружения **TZ** сервера начальной загрузки.

Примечание. Если вы должны создать для узла предварительно построенный образ, то не используйте маркеры, а вместо этого жестко кодируйте требуемые значения непосредственно в пользовательском файле построения. Для более полной информации смотрите buildqnx в книге Описание утилит, а также главу **Создание сети** в этом руководстве.

Если вы загружаетесь через сеть, то вы имеете вариант загрузки либо предварительно созданного образа, либо образа, создаваемого на лету. Если вы создаете образ на лету, что рекомендуется, то вы не будете нуждаться в том, чтобы строить его вручную, как показано выше. Вариант определяется в файле /etc/config/netboot и документирован в описании утилиты netboot.

Примечание. Когда утилита netboot вызывает buildqnx, чтобы построить образ на лету, загрузочный модуль *не* записывается на диск.

Встраиваемые образы

Встраиваемые образы требуются для встраиваемых систем с центральным процессором. В то время как термин "встраиваемая система" имеет различные значения, QNX поддерживает две основных классификации:

- встраиваемые персональные компьютеры (ПК);
- заказные аппаратные средства.

Встраиваемые ПК

Это компьютерные контроллеры, которые ведут себя точно как ПК - они могут иметь клавиатуры, экраны, диски или сетевые платы. Они обычно используются в системах управления техническим процессом. Некоторые из этих систем имеют небольшой объем доступной флэш-памяти, либо встроенной в плату центрального процессора (типа Ziatech ZT8902), либо доступной в качестве сменной карты (типа Ampro MM/SSD).

Пакет программ QNX Embedded Kit поддерживает различные платы центрального процессора и платы флэш-памяти. Эта поддержка включает загрузку из платы флэш-памяти (обычно через некоторые ловушки BIOS) и использование флэш-памяти как файловой системы.

Заказные аппаратные средства

Эти системы разрабатываются на заказ для того, чтобы приспособить их к единичным специфическим условиям применения. Они обычно имеют флэш-память, однако может отсутствовать диск или сетевая карта. Большинство разновидностей этих систем (например, процессоры Intel 386EX и AMD SC400) не имеют клавиатуры или экрана. Пакет программ QNX Embedded Kit обеспечивает поддержку для некоторых из распространенных процессорных вычислительных плат.

Выбор процессов для встраиваемого образа

Вы можете модифицировать пакет программ Embedded Kit QNX, чтобы он поддерживал ваши особенные аппаратные средства. Ваш встраиваемый образ должен содержать *только* процессы, требующиеся для запуска системы. Затем этот образ должен быть помещен во встраиваемую систему (наиболее вероятно, в ПЗУ).

Глава 3. Установка файловой системы

Этот раздел охватывает следующие темы:

- Введение
- Разделение пространства имен путей
- Установка файловой системы DOS

Введение

Каждая файловая система имеет свой собственный *корневой каталог*. Этот корень находится в верхней части иерархии каталогов, от него QNX начинает поиск других каталогов и файлов. Типичное имя для корневого каталога файловой системы на жестком диске по умолчанию - слеш, наклонная черта вправо, (/).

Файловая система с корнем / может состоять из одной или более *физических файловых систем*, подсоединенных вместе. Физическая файловая система может быть на отдельном диске или разделе диска.

Если жесткий диск имеет более чем одну файловую систему, их имена могут быть следующими: /hd2, /hd3, /home2, /home3 и т.д.

Одна из физических файловых систем обычно назначается корнем (/), в то время как другие файловые системы монтируются как подкаталоги.

Дерево префиксов определяет принадлежность путей менеджерам ввода/вывода и определяет, к каким дискам и устройствам осуществляется доступ, когда процесс открывает файл. Во всех примерах этой главы дерево префиксов используется для разделения пространства имен. Для получения дополнительной информации относительно дерева префиксов смотрите главу **Пространство имен ввода/вывода** в книге *Системная архитектура*.

Разделение пространства имен путей

Утилита mount может использоваться, чтобы монтировать разделы диска как блок-ориентированные файлы и чтобы монтировать блок-ориентированные файлы как файловые системы QNX. Когда вы монтируете блок-ориентированный файл как файловую систему, ее позиция в пространстве имен путей называется *точкой монтирования*.

Например, задан раздел QNX (тип 77) на жестком диске 0 (/dev/hd0), следующая команда создаст блок-ориентированный файл /dev/hd0t77:

```
mount -p /dev/hd0
```

Следующая команда будет монтировать все разделы, обнаруженные на жестком диске 0, и установит раздел QNX как корень:

```
mount -p /dev/hd0 /dev/hd0t77 /
```

Примечание. Суперпользователь, владелец, или кто-нибудь другой, кто имеет разрешение записи на устройство (например, диск) или раздел, может монтировать или демонтировать это устройство. Например, хорошее правило - демонтировать накопитель на гибких дисках (флоппи-диск) или другие съемные носители информации перед тем, как вынуть диск из дисковода. Это поможет предотвратить необычные ошибки или возможную порчу файла. Для дополнительной информации смотрите описание **umount** в книге *Описание утилит*.

Следующие примеры должны помочь разъяснить, как пространство имен путей разбить на разделы. Рассмотрим эти конфигурации:

- жесткий и гибкий диск;
- два жестких диска;
- несколько разделов QNX на жестком диске;
- локальные (местные) и удаленные жесткие диски.

В этих примерах принимается, что Fsys и соответствующие драйверы запущены, и что на вашем жестком диске была выполнена команда mount -p.

Жесткий и гибкий диск

Жесткий диск монтируется как наклонная черта вправо - слеш (/), и формируется корень файловой системы. Гибкий диск монтируются как /fd0:

```
mount /dev/hd0t77 /
mount /dev/fd0 /fd0
```

Любое обращение с именем пути, начинающимся с /fd0, будет адресоваться к файловой системе QNX на гибком диске. Например, показать все файлы на гибком диске можно следующей командой:

```
ls -aR /fd0
```

Два жестких диска (на одном узле)

Первый жесткий диск монтируется как слеш (/) и образует корень файловой системы. Второй жесткий диск монтируется как /home2:

```
mount /dev/hd0t77 /
mount /dev/hd1t77 /home2
```

Любое обращение с именем пути, начинающимся с /home2, будет адресоваться к файловой системе QNX на втором жестком диске. Например, показать все файлы на втором жестком диске можно следующей командой:

```
ls -aR /home2
```

Несколько разделов QNX на жестком диске

Вы можете иметь до *четырех* разделов QNX на одном жестком диске. Первый раздел должен иметь тип 77 (смотрите утилиту fdisk в книге *Описание утилит*). Остальным разделам должны быть назначены типы 78, 79 и 80. Например:

```
mount /dev/hd0t77 /
mount /dev/hd0t78 /home2
```

Любое обращение с именем пути, начинающимся с /home2, будет адресоваться к файловой системе QNX во втором разделе жесткого диска. Например, показать все файлы во втором разделе жесткого диска можно следующей командой:

```
ls -aR /home2
```

Локальные и удаленные жесткие диски

В сети вы можете иметь диски с файловой системой QNX на более чем одном компьютере. Вы можете сконфигурировать пространства имен так, чтобы эти файловые системы были:

- независимыми;
- первичными/вторичными;
- связанными независимыми.

В следующих примерах локальные файловые системы ассоциируются с блок-ориентированными файлами (например, /dev/hd0t77), а удаленные файловые системы ассоциируются с отображениями префиксов имен путей. Это отображение префиксов переназначает запросы к удаленной файловой системе, которая будет ассоциирована с удаленным блок-ориентированным файлом.

Независимая

В этой конфигурации вы рассматриваете каждую машину как независимую автономную файловую систему. Для доступа к файлу на удаленной машине вы должны в начале имени пути указать номер узла удаленной машины. Например:

```
//10/etc/motd
```

Возможность указать файловую систему для конкретного узла будет работать всегда и являться наиболее общим механизмом для доступа к удаленным файлам.

Первичная/вторичная

В этой конфигурации вы рассматриваете одну файловую систему как первичную, и монтируете вторую файловую систему как подкаталог в первичной. Например, предположим, что узел 1 имеет первичную файловую систему, а узел 2 - вторичную, смонтированную как /home2. Узел 1 загружен с жесткого диска и имеет корень, установленный на его локальном диске утилитой mount, встроенной в образ операционной системы (с Fsys и драйвером). Его файл системной инициализации вызывает утилиту prefix, чтобы монтировать удаленную файловую систему следующим образом:

```
prefix -A /home2=//2/home2
```

Узел 2 загружается через сеть с узла 1 и имеет корень файловой системы, установленный /=//1/, используя утилиту sinit с опцией **-r**, встроенную в образ ОС. Его файл системной инициализации должен вызвать утилиту mount, чтобы монтировать локальную файловую систему следующим образом:

```
mount /dev/hd0t77 /home2
```

Менеджер файловой системы (Fsys) и его драйвер могут быть включены в образ, но более предпочтительно запускать их из файла системной инициализации. Другими словами, узел 2 загружается, подобно простой бездисковой машине (рабочей станции), которая после загрузки запускает свою файловую систему.

Оба узла, 1 и 2 будут иметь доступ к файловой системе на узле 1 как /, а на узле 2 - как /home2.

Связанная независимая

В этой конфигурации вы имеете дело с каждой машиной как с независимой автономной файловой системой, однако, вы частично связываете их вместе посредством утилиты prefix. Например, допустим, что файловая система на узле 1 имеет каталог /home1, а файловая система на узле 2 имеет каталог /home2. Вы можете отобразить каждый из каталогов home в область другой файловой системы следующим образом:

На узле 1:

```
prefix -A /home2=//2/home2
```

На узле 2:

```
prefix -A /home1=//1/home1
```

Кроме этой связи, каждая файловая система автономна со своими собственными копиями /bin и т.п. Преимущество - большая избыточность: если один отдел использует узел 2, а узел 1

отключается, то отдел, используя узел 2, может продолжать работу (кроме файлов из /home1, которые будут недоступны).

В качестве альтернативы, вы можете использовать символические связи, чтобы обращаться к файлам в другой файловой системе. Символическая связь создает специальный файл, который имеет имя пути как его данные. Чтобы создать символическую связь, задайте опцию **-s** для утилиты ln.

Допустим, на узле 1 в каталоге /home1 есть файл с именем file1. Следующая команда обеспечит отделу, использующему узел 2, доступ к file1 через символическую связь file2 в каталоге /home2.

```
ln -s /home1/file1 //2/home2/file2
```

Этим создается символическая связь //2/home2/file2, которая указывает на /home1/file1. Теперь можно отобразить на экране содержимое file1 из каталога /home2 на узле 2, используя утилиту less, следующим образом:

```
less file2
```

Для более полной информации о символических связях, смотри главу **Менеджер файловой системы** в книге *Системная архитектура*.

Удаление и повторная установка драйверов

Суперпользователь может удалять (обычно с помощью утилиты rm) блок-ориентированные устройства. Если все устройства, за которые отвечает драйвер, удаляются, то драйвер будет завершаться автоматически. Это позволяет пользователям с соответствующими привилегиями устанавливать, удалять и повторно устанавливать драйверы в работающей системе.

Примечание. Для получения информации о том, как изменять атрибуты доступа пользователя, смотри раздел "Атрибуты доступа файлов" в главе **Настройка учетных записей пользователей.**

Установка файловой системы DOS

Менеджер файловой системы Dosfsys обеспечивает доступ к файлам DOS и каталогам, которые находятся на дисках или разделах дисков с DOS. Dosfsys может поддерживать до восьми дисководов.

Вы можете создавать, читать, записывать и удалять файлы и каталоги на дисках DOS с использованием большинства программ QNX и стандартных утилит QNX (таких как mkdir, ls и rmdir). Большинство утилит QNX будут работать с файлами DOS при условии, что структура файлов DOS соответствует функциональным требованиям утилиты.

Ваши собственные Си-программы также смогут обрабатывать файлы DOS точно так же, как они обрабатывают файлы QNX, используя стандартные функции ввода/вывода, такие как *open()*, *read()*, *write()*, *close()*, *seek()* и т.л. Когда вы читаете каталоги DOS, они будут представлены вам в формате QNX.

Режимы вызова

Dosfsys имеет четыре режима вызова:

```
Dosfsys [-S|-s] [-e] [-m] [-t] [dos_drive=qnx_drive[,R]]... &

Dosfsys -i [-n node] [dos_drive_path]...

Dosfsys -o [-n node]

Dosfsys -x [-n node]
```

Опция **-i** позволяет вам получать информацию о накопителях DOS, подключенных в настоящее время. Опция **-o** позволяет вам увидеть имена любых файлов, открытых в настоящее время на каждом устройстве. Опция **-x** завершает работу сервера Dosfsys.

Если вы не задаете **-i**, **-o** или **-x**, то Dosfsys будет запускаться и пытаться подключить указанные дисководы.

Для того чтобы запустить сервер Dosfsys или завершить его работу, вы должны войти в систему как суперпользователь (root).

Запуск Dosfsys

Когда вы запускаете Dosfsys, он:

- открывает определенный дисковод(ы);
- регистрирует корневое имя DOS (/dos)
- регистрирует имя qnx/dosfsys у локального Менеджера процессов.

Если никакие опции не были определены, или если были заданы опции **-S** или **-s**, то Dosfsys просматривает каталог /dev для подключения допустимых дисководов DOS. Если в командной строке не задано иначе, то Dosfsys будет пытаться подключить до восьми дисководов.

Первичные разделы DOS (/dev/hd0t1, /dev/hd0t4, /dev/hd0t6 и т.п.) и расширенные разделы (/dev/hd0t1.1, /dev/hd0t1.2 и т.п.) монтируются, начиная с накопителя C (/dos/c).

Дисководы гибких дисков монтируются следующим образом:

```
/dev/fd0    -->  /dos/a
/dev/fd1    -->  /dos/b
```

Регистрация имен менеджером Dosfsys

Менеджер файловой системы Dosfsys может подключить до восьми дисководов. Как упоминалось выше, менеджер Dosfsys зарегистрирует имя /dos как системный префикс. Он также сможет управлять каждым специфическим именем дисковода, таким как /dos/a, /dos/b и т.п. Эти имена не регистрируются в дереве системных префиксов, но содержатся непосредственно в Dosfsys. Это могло быть прозрачно для пользователя, если бы не то обстоятельство, что пользователь не может создавать файлы или каталоги в корне /dos.

Устройства DOS

Устройство DOS может быть одним из следующих:

- раздел DOS на жестком диске;
- гибкий диск (дискета);
- образ раздела DOS или дискеты.

Для того чтобы создать образ дискеты DOS или раздела DOS, используйте утилиту QNX cp. Например, для того, чтобы скопировать образ дискеты DOS на ваш дисковод гибких дисков номер 0, вы можете использовать следующее:

```
cp /dev/fd0 /usr/qnx/dosa
```

и затем вызвать Dosfsys следующим образом:

```
Dosfsys a=/usr/qnx/dosa &
```

То же самое можно выполнить с разделом жесткого диска. Dosfsys будет обрабатывать эти образы так же, как будто это существующее устройство.

Для всех *несъемных* устройств Dosfsys при запуске считывает непосредственно блок параметров начальной загрузки DOS (Boot Parameter Block - BPB), а также часть таблицы распределения файлов (File Allocation Table - FAT). На *съемных* устройствах BPB и FAT будут считаны только тогда, когда устройство будет доступно.

Когда Dosfsys имеет открытое *несъемное* устройство, то устройство заблокировано для ЗАПИСИ так, что никакой другой процесс не сможет записывать на это устройство без взаимодействия с Dosfsys. Съемные устройства остаются открытыми и блокируются только во время доступа (например, во время чтения или записи на диск). Обратите внимание, что если вы не задали опцию **R**, то все дисководы имеют доступ для ЧТЕНИЯ/ЗАПИСИ.

Поддержка версий DOS

Менеджер Dosfsys поддерживает все форматы разделов, начиная с DOS версии 2.1 или выше, включая стандартные первичные разделы DOS, большие разделы DOS (DOS 4.0 и DOS 5.0 > 32 Мбайт) и расширенные разделы DOS (тип 5). Поддерживаются жесткие диски и гибкие диски размера $5^1/_4$" и $3^1/_2$".

Типы разделов DOS

Стандартные типы разделов жесткого диска DOS:

Тип раздела:	**Описание:**
1	первичный раздел DOS (12-битовые FAT)
4	первичный раздел DOS (16-битовые FAT; <= 32 Мбайт)
5	расширенный раздел DOS (DOS 3.3 или более поздней)
6	первичный раздел DOS (DOS 4.0 или более поздней; > 32 Мбайт)
11	32-битовые FAT DOS; разделы до 2047 Гбайт
12	то же самое, что и тип 11, но для увеличения объема используется Logical Block Address (LBA) и Int 13h
14	то же самое, что и тип 6, но для увеличения объема используется Logical Block Address (LBA) и Int 13h
15	то же самое, что и тип 5, но для увеличения объема используется Logical Block Address (LBA) и Int 13h

Текстовые файлы DOS

DOS использует структуру для текстовых файлов, которая отличается от структуры, используемой в QNX (текстовыми файлами мы называем строковые файлы, содержащие строки ASCII-текста, разделяемые последовательностями разделителей строк). В DOS каждая строка текстового файла завершается последовательностью возврат каретки/перевод строки (CR/LF); в QNX каждая строка завершается символом перевода строки (LF).

Менеджер Dosfsys не преобразует эти файлы. Все файлы обрабатываются "как есть". Поэтому вы должны использовать утилиту QNX textto или tr, чтобы преобразовывать ваши текстовые файлы перед копированием их на диски или с дисков QNX и DOS.

Обратите также внимание, что текстовые файлы, созданные некоторыми программами DOS, могут содержать символ SUB (^Z) как последний символ файла. Он тоже обрабатывается "как есть".

Двоичные файлы DOS

Поскольку Dosfsys не преобразует содержимое файлов, двоичные файлы могут быть скопированы из разделов QNX в разделы DOS и обратно "как есть".

Преобразование символов и имен из QNX в DOS

В DOS ваши имена файла ограничиваются в соглашении "8.3" (QNX не имеет такого ограничения). Кроме того, вы не можете включить любой из представленных ниже символов в имя файла (несмотря на то, что они вполне допустимы в QNX):

```
/  \  [  ]  :  *  |  +  =  ;  ,  ?
```

Вы не должны использовать эти имена файлов:

```
AUX
CLOCK$
COM1
COM2
COM3
COM4
CON
LPT1
LPT2
LPT3
NUL
PRN
```

Если вы пытаетесь в DOS создать файл, который содержит один из недопустимых символов DOS или имеет недопустимое имя, вам будет отказано в доступе.

Так как под QNX все имена файлов DOS и символы имен файлов разрешены, никакой проверки правильности для этих имен файлов не требуется.

DOS также преобразует все символы алфавита в верхний регистр, поэтому Dosfsys преобразует эти символы в верхний регистр при создании имени файла DOS; он преобразует имя файла в нижний регистр при возврате имени файла в приложение QNX.

Преобразование имен файлов QNX

Если при запуске **-t** вы задаете Dosfsys, то сервер будет преобразовывать и/или усекать любые имена файлов QNX, которые не соответствуют соглашению о присваивании имен в DOS. Следующая таблица описывает, как преобразовываются имена:

Если имя файла QNX:	**Dosfsys будет:**
Начинается с точки (.)	Изменять . на $

Если имя файла QNX:	**Dosfsys будет:**
Содержит несколько точек	Изменять все точки, кроме последней, на $
Имеет префикс длиннее восьми символов	Усекать префикс до восьми символов
Имеет суффикс длиннее трех символов	Усекать суффикс до трех символов

Примечание. В версии Windows 95 фирмы Microsoft добавлена поддержка длинных имен файлов, которые не ограничиваются традиционным соглашением 8.3 DOS об именах файлов. Dosfsys может считывать, но не создавать, длинные имена файлов Windows 95. Чтобы Dosfsys распознавала длинные имена файлов Windows 95, при запуске Dosfsys задайте опцию **-L**.

Здесь несколько примеров преобразования имен QNX в имена DOS:

Имя файла QNX:	**Имя файла DOS:**
.profile	$profile
a.b.c.d	abc.d
longfilename	longfile
longfilename.extension	longfile.ext
a..b..c.def.g.h	a$$b$$c$.h

Метки тома DOS

DOS использует понятие метки тома, которое является действующим элементом каталога в корне файловой системы DOS. Чтобы различать метку тома и действующий файл DOS, Dosfsys ставит знак равенства (=) в качестве первого символа имени метки тома. Dosfsys трактует этот элемент каталога как файл нулевой длины только для чтения, чьи атрибуты доступа не могут быть изменены.

Преобразование атрибутов доступа DOS/QNX

DOS не поддерживает все биты атрибутов доступа, которые использует QNX. DOS имеет следующие биты атрибутов:

READ_ONLY – только для чтения

HIDDEN – скрытый

SYSTEM – системный V

OLUME_LABEL – метка тома

DIRECTORY – каталог

ARCHIVE – архивный

Преобразование битов атрибутов доступа QNX в DOS

Dosfsys использует следующую логику соответствия, чтобы осуществить преобразование битов атрибутов QNX в DOS:

- если запись представляет собой каталог, то устанавливается бит файла DOS DIRECTORY;
- если запись представляет собой файл, и если все биты QNX WRITE выключены, то устанавливается бит DOS READ_ONLY.

Преобразование битов атрибутов доступа DOS в QNX

Чтобы осуществить преобразование битов атрибутов доступа DOS в QNX, используется следующая логика соответствия:

- устанавливаются биты атрибутов доступа QNX READ ("чтение") для пользователя, группы и для остальных;
- если запись не является меткой тома и если запись не только для чтения, то устанавливаются биты атрибутов доступа QNX WRITE ("запись") для пользователя, группы и для остальных;
- если запись представляет собой каталог, то устанавливаются биты QNX DIRECTORY ("каталог") и EXECUTE ("выполнить") для пользователя, группы и для остальных;
- если запись представляет собой файл, устанавливается бит QNX REGULAR FILE ("регулярный файл").

Примечание. Если файл записывается, то будет устанавливаться бит DOS ARCHIVE ("архивный").

Если Dosfsys запускается с опцией **-e**, все файлы и каталоги DOS будут иметь набор битов QNX для группа/владелец/остальные.

Принадлежность файла

Хотя файловая система DOS не поддерживает идентификаторы пользователей и идентификаторы групп, Dosfsys не будет возвращать код ошибки при попытке изменить идентификатор группы или идентификатор пользователя утилитой chown или библиотечной функцией *chown()*. Ошибка не возвращается, потому что ряд утилит использует библиотечную функцию *chown()*, что может привести к большому количеству отображаемых сообщений об ошибках.

Все файлы под Dosfsys, принадлежащие суперпользователю (uid=0, group=0), доступны для всех.

Завершение Dosfsys

Опция **-x** прекращает действия сервера Dosfsys. Если вы задаете **-x**, то никакие новые запросы функций *open()* не будут приниматься и сервер закончит работу, как только все активные файлы (то есть файлы все еще открытые) закроются.

Коды ошибок, возвращаемые Dosfsys

Если запрос, сделанный к Dosfsys, не поддерживается, то приложению, делающему запрос, будет возвращен код ошибки EOPNOTSUPP. Примеры запросов, не поддерживаемых Dosfsys:

```
LINK

BLOCK_READ

BLOCK_WRITE

MOUNT_PARTITION

MOUNT_RAMDISK  PIPE

DISK_GET_ENTRY

RECORD LOCKING

SYMBOLIC LINK
```

Если Dosfsys обнаруживает испорченную файловую систему, то будет возвращена ошибка EBADFSYS, которая указывает, что вам необходимо выполнить утилиту CHKDSK под DOS, чтобы устранить проблему.

Структура файловой системы DOS такая, что размер корневого каталога фиксируется во время форматирования и не может изменяться. Если корневой каталог переполняется, то выдается ошибка (ENOSPC).

Глава 4. Подключение символьных устройств

Эта глава охватывает следующие разделы:

- Запуск драйверов устройств
- Параллельные устройства
- Последовательные устройства
- Псевдотерминальные устройства
- Консольные устройства

Запуск драйверов устройств

Система QNX, обычно, содержит одно или более *символьных устройств*. Все эти устройства управляются процессом Dev. Сначала должен быть запущен этот процесс (как root), затем можно запускать любые драйверы устройств:

```
/bin/Dev &
```

Как только Dev запущен, можно запустить один или более следующих драйверов устройств (как root):

- `/bin/Dev.con` (драйвер консольного устройства QNX);
- `/bin/Dev.par` (драйвер параллельного печатающего устройства);
- `/bin/Dev.ser` (драйвер последовательного устройства);
- `/bin/Dev.pty` (драйвер псевдотерминала).

Каждый из этих драйверов подробно описан в книге *Описание утилит*.

Примечание. Вам может понадобиться модифицировать команду Dev в вашем файле sysinit.*узел* с целью поддержки большего числа устройств, чем это принято по умолчанию. Например, если вы зададите:

```
Dev [другие_опции] -n 64 &
```

то Dev будет поддерживать 64 терминальных устройства (виртуальных консолей, последовательных терминалов и псевдотерминальных устройств). Для получения более полной информации смотри описание Dev в книге *Описание утилит* QNX 4.

Параллельные устройства

Параллельные порты обычно используются для взаимодействия с параллельными принтерами. До подключения принтера к компьютеру, кроме запуска драйвера, вам следует выполнить еще некоторые действия. Если принтеры сетевые, то смотрите главу Печать со

спулингом, чтобы установить принтер как ресурс совместного (коллективного) использования.

Единственный параллельный порт

Если на компьютере доступен только один параллельный порт, то никакие параметры не нужны:

```
Dev.par &
```

Запущенный таким образом, параллельный драйвер создаст устройство с именем /dev/par1, которое соответствует первому параллельному порту, обнаруженному BIOS (LPT1).

Несколько параллельных портов

Если компьютер имеет более одного параллельного порта, то вам необходимо запустить дополнительные драйверы Dev.par для каждого дополнительного порта. К тому же вы должны обеспечить дополнительные устройства уникальными именами. Например:

```
Dev.par &
Dev.par -b 2 -N laser &
```

Эти команды создадут устройство с именем /dev/par1 на LPT1 и второе устройство с именем /dev/laser на LPT2.

Буфера вывода

При наличии памяти вы можете обнаружить, что при задании больших буферов вывода значительно сокращается длительность цикла обработки при передаче данных на ваш принтер. Вот пример параллельного устройства, созданного с буфером вывода в 30 КБайт:

```
Dev.par -O 30000 &
```

Тестирование параллельных драйверов

Если у вас проблемы с вашим принтером, то вы можете протестировать работоспособность драйвера, используя утилиту cat. Утилита cat записывает содержимое файла на стандартный вывод. Вы можете перенаправить вывод на принтер, чтобы определить, работает ли параллельный драйвер как ожидается:

```
cat myfile > /dev/par
```

Для многих принтеров, чтобы заставить страницу печататься, необходимо послать код перевода страницы (**Ctrl-L**).

Последовательные устройства

В этом разделе мы рассмотрим последовательные аппаратные средства, протокол RS-232 и различные проблемы конфигурации.

Аппаратные средства адаптеров

Адреса ввода/вывода

Драйвер Dev.ser может поддерживать один или более последовательных портов. Аппаратный интерфейс в компьютере состоит из универсального асинхронного приемо-передатчика (UART) для каждого последовательного порта. Драйвер поддерживает семейства последовательных контроллеров типа 8250, 16450 и 16550.

Каждый UART использует восемь адресов, лежащих подряд в адресном пространстве ввода/вывода компьютера. При запуске драйвер Dev.ser получает из аргументов командной строки информацию относительно адресного диапазона ввода/вывода для каждого UART.

Аппаратное прерывание

Таким же важным, как адрес ввода/вывода, является аппаратное прерывание, генерируемое каждым UART. Большинство компьютеров имеет на своей шине несколько сигналов аппаратного прерывания, обозначаемых от IRQ2 до IRQ15 (за исключением прерываний 8, 9 и 13, которые используются непосредственно материнской платой системы).

Эти сигналы прерывания возбуждаются высокими активными сигналами TTL-логики на ISA-шинах, а это означает, что вы можете подключить только одну плату адаптера к одному сигналу прерывания. Платы последовательных адаптеров бывают различных конфигураций. Адаптерные платы только с одним последовательным портом обычно предлагают только ограниченный набор вариантов адресов ввода/вывода и аппаратного прерывания. Их типовые комбинации предлагаются в следующей таблице, но рекомендуется внимательно прочитать документацию на аппаратные средства, чтобы выбрать комбинацию, допустимую для платы данного изготовителя:

Имя:	**Адрес:**	**Прерывание:**
COM1	3F8	IRQ4
COM2	2F8	IRQ3
COM3	3E8	выбираются
COM4	2E8	выбираются

Многопортовые последовательные адаптеры

Как правило, можно конфигурировать многопортовые последовательные адаптеры в

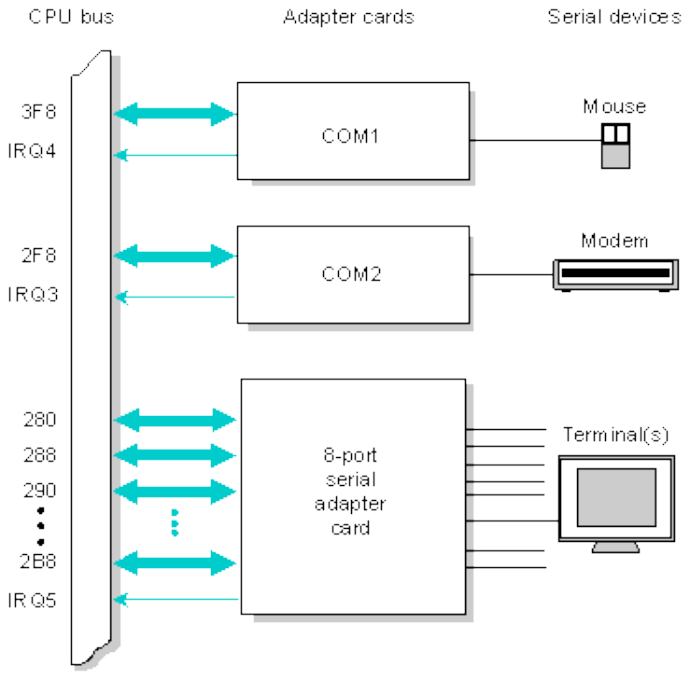
пределах широкого диапазона адресов ввода/вывода. Кроме того, адаптеры могут дать вам значительную гибкость в выборе аппаратных прерываний. Часто хорошим выбором могут быть адреса ввода/вывода в диапазоне от 280 до 2BF.

Примечание. Эта информация применима только к "неинтеллектуальным" последовательным платам. "Интеллектуальные" платы обычно поддерживаются драйверами, поставляемыми изготовителями аппаратных средств - обратитесь к документации, которая поставляется вместе с платой.

Вследствие ограниченного числа имеющихся в распоряжении аппаратных прерываний, для этих плат часто используется логическое сложение ("OR") линий прерывания от индивидуального UART в единственное прерывание, соединенное с шиной.

Примечание. QNX допускает совместное использование одного и того же прерывания большим количеством последовательных портов, так как Dev.ser проверит каждый UART, который использует это прерывание.

Типичная установка аппаратных средств



На следующем рисунке показана сложная конфигурация последовательных адаптерных плат в системе QNX.

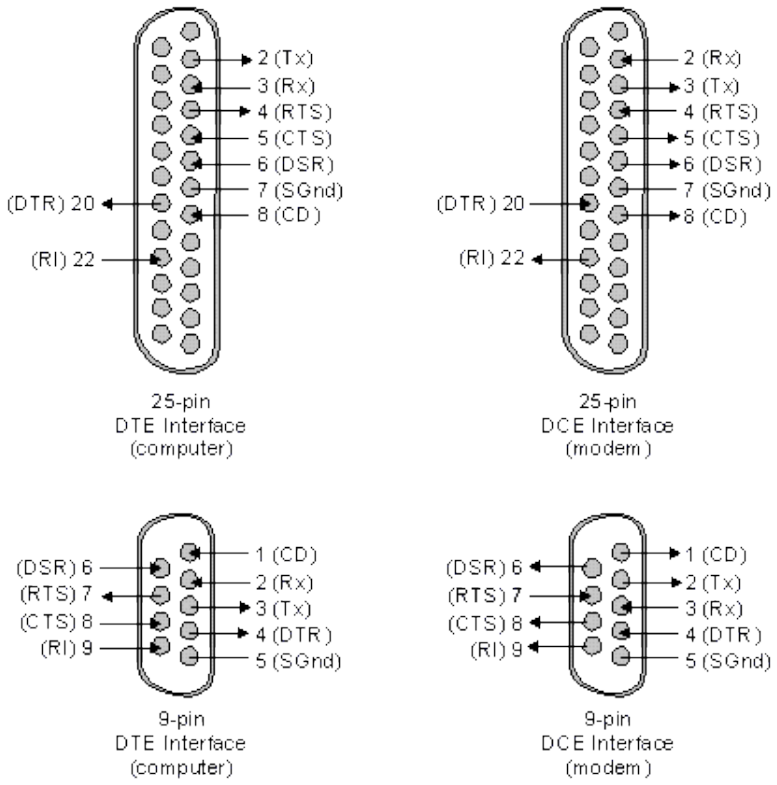
Примечание. Для правильного функционирования каждый последовательный канал должен иметь уникальный адрес ввода/вывода, а каждая плата адаптера должна использовать уникальное аппаратное прерывание.

Последовательный протокол RS-232

Протокол асинхронной связи RS-232C определяет электрический и физический интерфейс между терминальным оборудованием пользователя (DTE или *терминалами*) и аппаратурой передачи данных (DCE или *модемами*).

Электрический интерфейс

Следующий рисунок показывает кабельную разводку при подключении к стыку RS-232.



Ведущий компьютер обычно конфигурируется как DTE, выступая в роли терминального устройства. Предполагается, что компьютер будет соединен с модемом.

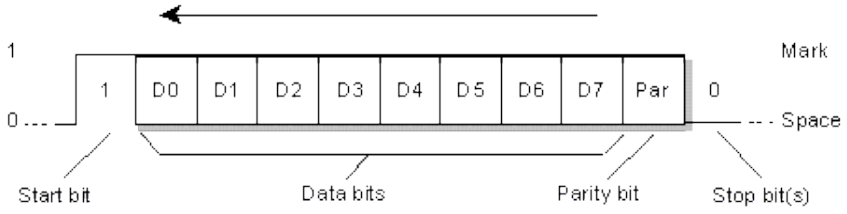
Сигналы RS-232 имеют следующие обозначения:

Сигнал:	**Назначение:**
Tx	передача данных
Rx	прием данных
RTS	запрос для посылки
CTS	очистка для посылки
DSR	готовность набора данных
DTR	готовность терминальных данных

Сигнал:	**Назначение:**
CD	обнаружение несущей
RI	индикатор кольца
SGnd	сигнальная земля

Последовательный протокол

Данные передается асинхронно, используя битовый протокол, как показано ниже:



Обычно линия данных RS-232 находится в состоянии SPACE (0). Переданный символ состоит из битов, передаваемых в следующем порядке:

- бит START (всегда 1);
- от 5 до 8 битов данных (младший разряд - первый);
- бит паритета (необязательный);
- один или более битов STOP (0).

Продолжительность каждого бита определяется *скоростью передачи данных*, которая указывает на число передаваемых бит в секунду.

Если вы используете бит паритета, то он может иметь следующие значения:

`odd` – сумма битов данных плюс бит паритета - нечетная
`even` – сумма битов данных плюс бит паритета - четная
`mark` – всегда 1
`space` – всегда 0

Управление сеансом

RS-232 использует линии DTR и DSR для управления сеансами связи. Терминал устанавливает уровень сигнала DTR высоким, когда он включен и доступен. Аналогично,

модем устанавливает уровень сигнала DSR высоким, когда он включен и доступен (но не обязательно *соединен* с удаленным модемом). Обмен данными не состоится, пока уровни обоих сигналов - DTR и DSR - не станут высокими.

Терминал, сбрасывая (понижая) уровень сигнала на линии DTR, сообщает, что он не желает далее передавать данные, что заставляет большинство модемов разъединить телефонную линию, таким образом, освобождая соединение. Модем, устанавливая уровень сигнала CD высоким, сообщает, что он установил соединение. Некоторые модемы также будут сообщать, поднимая RI, что они обнаружили (но еще не ответили) поступающий вызов.

Управление потоком данных

Линии RTS и CTS управляют потоком данных между терминалом и модемом. Терминал поднимает RTS, когда он способен принять данные по линии Rx. Аналогично, модем поднимает CTS, когда он может получать данные по линии Tx.

Конфигурирование последовательных портов

Используйте утилиту stty для установки четырех основных параметров, которые определяют связь RS-232.

Биты данных

QNX поддерживает четыре размера символа. Размер символа данных выбирается одной из следующих команд stty:

`stty bits=`*n*
где *n* может быть 5, 6, 7 или 8 (по умолчанию).

Этот параметр определяет, сколько битов, следующих за стартовым битом, будет использоваться для формирования символа.

Стоповый биты

Данные можно передать, если за ними следуют один или два стоповых бита. Два стоповых бита используются только для замедления передачи данных, только для того, чтобы удаленный конец не отставал. Используя stty, задайте одну из следующих команд:

```
stty stopb=1 (по умолчанию)
```

ИЛИ

```
stty stopb=2
```

Паритет

Для того чтобы отключить передачу битов паритета и запретить проверку (в аппаратных средствах) принимаемых битов паритета, задайте:

```
stty par=none
```

Проверка паритета по умолчанию отключена.

Если бит паритета нужно использовать, то задайте одно из следующих значений (эти значения были описаны выше):

```
stty par=odd
stty par=even
stty par=mark
stty par=space
```

Скорость передачи в бодах

Вы можете определить скорость передачи в бодах опцией **baud=***число* утилиты stty. Например:

```
stty baud=2400
```

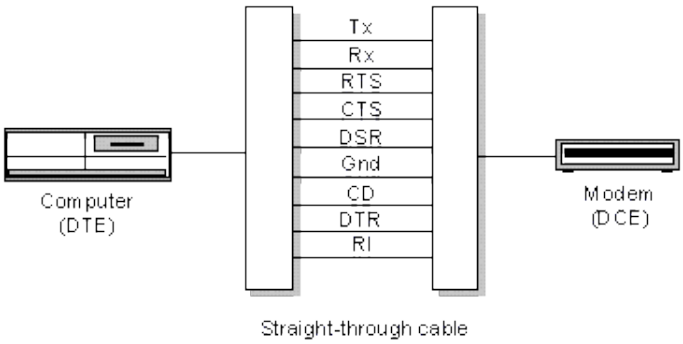
Драйвер Dev.ser устанавливает по умолчанию скорость, равную 9600 бод. Драйверы "третьих фирм" могут устанавливать по умолчанию различные значения скорости.

Соединение последовательных устройств

Высокоскоростные ECC-модемы

Высокоскоростные модемы с исправлением ошибок становятся очень сложными - они работают лучше всего, когда используются все аппаратные сигналы квитирования установления связи. Эти модемы часто взаимодействуют с ведущим компьютером на фиксированной высокой скорости передачи (например, 115200 или 57600 бод) и используют линии RTS/CTS для квитирования связи, чтобы регулировать фактический поток данных. QNX идеально подходит для работы с такими модемами.

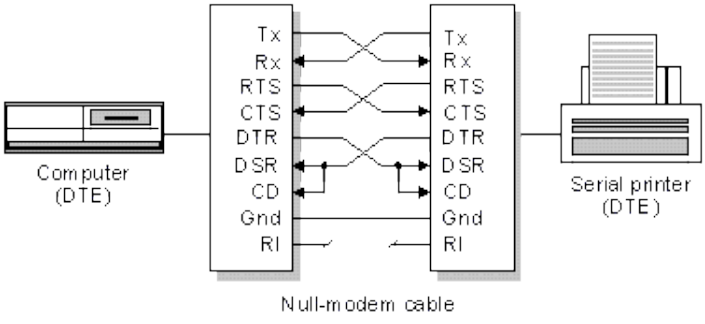
Чтобы соединить модем с компьютером, используется следующий кабель.



Последовательные принтеры

Последовательные принтеры - это обычно *двунаправленные* устройства. Основной поток данных передается из компьютера на принтер, но так как принтеры могут отставать от компьютера, чтобы отрегулировать поток данных, для вывода на последовательные принтеры часто используется программное управление потоком данных. Другими словами, они передают обратно на компьютер символы XON (старт) и XOFF (стоп). Некоторые принтеры для этой цели используют аппаратные линии квитирования связи, а некоторые поддерживают обе формы управления потоком данных.

Для надежности вам следует соединить все девять сигналов, хотя принтеры, которые поддерживают только программное управление потоком данных, могут функционировать так же хорошо с трехпроводным кабелем (Rx, Tx и Gnd). Кроме того, т.к. принтеры обычно сконфигурированы как DTE (Data Terminal Equipment), точно так же, как ведущий компьютер, то можно использовать кабель нуль-модема.



Примечание. Фактическое подключение контактов CD и RI может отличаться в зависимости от изготовителя кабеля.

Если принтер использует:	**Используйте:**
Программное управление потоком данных	stty +osflow </dev/ser1
Аппаратное управление потоком данных	stty +ohflow </dev/ser1
Программное и аппаратное управление	Обе опции stty

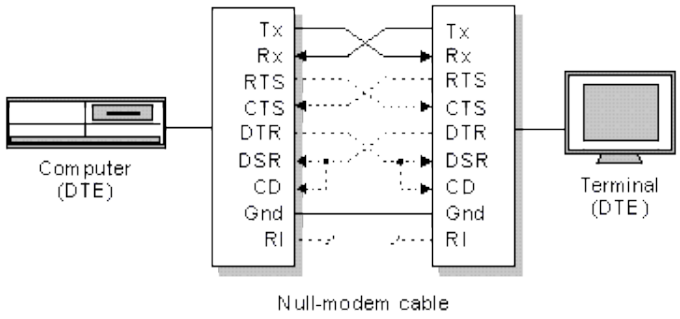
Если принтер использует:	**Используйте:**
потоком данных	

Примечание. Если ваш принтер сетевой, то чтобы установить принтер в качестве ресурса общего пользования, смотрите главу **Печать со спулингом.**

Терминалы

Терминалы работают с управлением или без управления потоком данных и с фиксированной скоростью передачи данных. В отличие от принтеров, терминалы обычно не отстают от скорости передачи данных, поддерживаемой ведущим компьютером. Для соединения простого трехпроводного кабеля может быть достаточно, но рекомендуется кабель с девятью проводами.

Подобно ведущему компьютеру, терминалы обычно сконфигурированы как DTE-устройства, так что обычно требуется кабель нуль-модема.



Если терминал поддерживает программное управление потоком данных (например, VT100), то рекомендуется включать управление для выходного потока данных и фиксировать его:

```
stty +osflow +lksflow </dev/ser1
```

Можно также использовать опцию **-x** при запуске Dev.ser.

Выбор конфигурации последовательных линий для терминалов и пользователей

QNX может использоваться как полнофункциональная система с разделением времени. Большое количество пользователей может быть подключено к некоторым или ко всем компьютерам в сети QNX через терминалы, в том числе и удаленные, использующие модемные линии связи.

Предположим, что терминал/модем правильно сконфигурированы и вам только осталось обеспечить механизм для "входа в систему" пользователя.

Вход в систему

Рассмотрим терминал, который правильно сконфигурирован и подключен к последовательному порту /dev/ser1. Самый простой способ разрешения входа в систему пользователю, сидящему за данным терминалом, - использовать команду:

```
on -t /dev/ser1 login
```

Пользователь сможет войти в систему и выполнять команды. Однако когда пользователь выйдет из системы, сеанс командного интерпретатора завершится, и пользователь не сможет снова войти в систему. Чтобы пользователю дать возможность снова войти в систему, задайте опцию **-t** в утилите tinit при инициализации терминала.

```
tinit -t /dev/ser1 /dev/ser2 &
```

Автоматизированный вход в систему

Можно использовать утилиту tinit, чтобы автоматизировать процесс входа в систему. Когда вы запускаете tinit с опцией **-T**, эта утилита будет ожидать символ, который нужно послать с терминала. Вы можете задать (с помощью опции **-c**) команду для tinit, чтобы она выполнялась при нажатии клавиши. По умолчанию tinit запускает утилиту login.

После того, как пользователь вышел из системы, tinit снова ожидает символ от терминала. Приведенная ниже команда после нажатия клавиши запускает login на двух последовательных устройствах (/dev/ser1 и /dev/ser2):

```
tinit -T /dev/ser1 /dev/ser2 &
```

Запуск пользовательских приложений

Вы можете, используя опцию **-c** в утилите tinit, запустить любую программу. Вы даже можете задать для каждого устройства разные программы. В некоторых средах "стандартное" приложение всегда запускается на заданном терминале.

Для получения дополнительной информации смотрите описание утилиты tinit в книге *Описание утилит.*

Доступ к модему

Утилита modem предусмотрена для того, чтобы использовать модемы. Используемая в сочетании с tinit, утилита modem может обеспечить отличные возможности вызова по номеру. Типичная система вызова по номеру, используемая QNX, могла бы иметь несколько последовательных портов (/dev/ser1, /dev/ser2 и т.п.) и могла бы использовать следующую команду, чтобы обеспечить доступ по номеру:

```
tinit -c "modem -b 38400 -L" -t /dev/ser1 &
```

Утилита tinit может запускать утилиту modem на каждую из последовательных линий. Когда связь установлена, modem выполнит следующее:

- ответит на телефонный вызов;
- определит и установит соответствующую скорость передачи в бодах;
- *exec* (запустит вместо себя) утилиту login

Когда пользователь либо выходит из системы, либо "дает отбой", tinit снова запустит новую утилиту modem, которая будет ожидать другой вызов. Для получения дополнительной информации смотрите утилиту modem в книге *Описание утилит*.

Монопольный доступ к последовательному устройству

Большинство процессов могут открывать одно и то же устройство; если они считывают из устройства, их считывание будет перемежаться (поочередным). Это, как правило, нежелательно. Чтобы уберечься от этого, QNX поддерживает "счетчик открытий."

Счетчик открытий, подобно блокировкам, является только консультативным. Программы, которые требуют монопольного использования устройства, должны перед чтением из устройства проверять счетчик открытий (с помощью функции *dev_info()*). Если счетчик открытий равен единице, то это указывает на то, что ваша программа только одна открывается устройством. Утилиты modem и qtalk используют это средство, так что вам не требуется выделять отдельную линию последовательной передачи для поступающих вызовов.

Эмулятор терминала qtalk проверяет, свободна ли линия, и если это так, то modem будет устанавливать связь, позволяя вам сделать исходящий вызов. Когда сеанс qtalk завершен, то modem снова получит управление, ожидая прибытия поступающих вызовов. Для получения дополнительной информации смотрите описание утилиты qtalk в книге *Описание утилит*.

Тестирование последовательных драйверов

Если у вас есть проблемы с линией последовательной передачи данных, то вы можете использовать утилиту qtalk, чтобы проверить, что последовательный драйвер работает правильно. Запуск qtalk позволит вам общаться с системой, к которой вы подключены с помощью последовательной линии. Когда утилита qtalk запущена, то вы можете передавать по линии последовательной связи вводимые символы.

Следующая команда будет запускать сеанс qtalk с системой, подключенной непосредственно

к устройству:

```
qtalk -m device
```

Теперь ваши нажатия клавиши qtalk будет посылать через последовательную линию, и данные, поступающие на последовательную линию, должны отображаться на вашем экране. Если это не происходит, то, возможно, имеет место проблема с конфигурацией последовательного порта. Чтобы протестировать выполняемые функции базовых последовательных портов, присоедините модем к линии последовательной передачи данных и попробуйте общаться с модемом в его командном режиме.

Проблемы поиска и устранения неисправностей последовательных устройств

Следующая таблица описывает действия, которые вы можете выполнять, если вы сталкиваетесь с некоторыми общими проблемами, имеющими место при подключении последовательных устройств.

Проблема:	**Возможная причина:**	**Выполнить действия:**
Не передаются и не принимаются данные	Поврежден кабель	Проверьте кабели; используйте кабель нуль-модема, если можно
	Неисправные порты ввода/вывода	Проверьте аппаратные установки и соответствующие параметры в Dev.ser
	Конфликт прерываний	Проверьте установку векторов прерывания на адаптерной плате
Данные принимаются и передаются, только когда используется другой последовательный порт	Конфликт прерываний	Проверьте аппаратные прерывания и параметры запуска Dev.ser; убедитесь, что два последовательных адаптера *не* используют один и тот же IRQ
Случайные потери данных	Управление потоком данных поддерживается, но устройство не настроено	Определите тип управления, потоком данных, поддерживаемый устройством, и настройте его утилитой stty (**ihflow**, **ohflow**, **isflow** и **osflow**)
	Управление потоком не поддерживается	Уменьшите скорость передачи данных и/или увеличьте количество стоповых бит; если теряются только

Проблема:	Возможная причина:	Выполнить действия:
		принимаемые данные, задайте более длинный входной буфер в Dev.ser (опцией **-I**)
	Проблемы с кабелем	Убедитесь, что кабель хорошо заземлен и не слишком длинный; также убедитесь, что все провода RS-232 в кабеле подключены
Не распознаются символы данных	Неправильно установлена скорость передачи данных или паритет	Используйте stty для установки правильной скорости передачи данных и/или паритета
Часть символов отображается нормально, а часть нет	Неправильно установлен паритет	Попробуйте изменить установку паритета, используя утилиту stty

Псевдотерминальные устройства

Менеджер Dev несет ответственность за управление всеми терминальными устройствами, включая "псевдотерминальные" устройства или ptys. Псевдоустройства размещаются парами; элементы каждой пары внутренне соединены друг с другом. Любые данные, записываемые на ведущей половине (например, /dev/ptyp0), будут доступны для чтения на ведомой половине (например, /dev/ttyp0).

Вы можете столкнуться с общей проблемой, когда Dev достигает своего максимального числа устройств и отвергает попытки регистрации Dev.pty. Если это случится, увеличьте значение опции **-n** менеджера Dev.

Например, создадим 16 пар устройств с именами от /dev/ptyq0 до /dev/ptyqf и от /dev/ttyq0 до /dev/ttyqf:

```
Dev.pty -n 16 -l q &
```

Консольные устройства

Адаптер монитора, монитор и системная клавиатура вместе рассматриваются в QNX как *консоль*. Чтобы позволить вам взаимодействовать сразу с несколькими приложениями, QNX разрешает нескольким сеансам выполняться на консолях одновременно посредством *виртуальных консолей*. Эти виртуальные консоли обычно имеют имена /dev/con1, /dev/con2 и т.п.

Dev управляет консолями через свои действующие совместно драйверы ввода/вывода Dev.con или Dev.ansi. Кроме того, сам менеджер Dev должен запускаться *раньше* любого из

его драйверов.

Чтобы установить максимальное число консолей для компьютера, используйте опцию **-n** консольного драйвера. Например, следующая команда запустит драйвер Dev.con максимум для 6-ти виртуальных консолей:

```
Dev.con -n 6 &
```

Вы можете переключиться с одной консоли на другую, используя эту простую комбинацию клавиш:

Ctrl-Alt-Enter

Для получения более подробной информации смотрите в **Приложении 1. Консоли QNX и клавиатурные соглашения**.

Глава 5. Лицензирование

Эта глава охватывает следующие темы:

- Лицензирование операционной системы
- Проверка лицензий
- Добавление лицензий

Лицензирование операционной системы

Для каждого компьютера, который работает под QNX, должна быть получена лицензия. Причем неважно, используете ли вы 10 автономных машин или 10 машин, объединенных в сеть.

В сети каждая машина рассматривается как "узел" и каждому узлу присваивается логический идентификатор (ID) узла. Логические ID узлов изменяются в пределах от 1 до общего числа узлов в сети. Например, сеть, имеющая лицензии для трех узлов, будет поддерживать логические ID узлов 1, 2 и 3.

Число узлов в сети определяется количеством лицензий, которые вы установили. Вы устанавливаете лицензии старого стиля, выполняя утилиту license, или нового стиля, добавляя их в файл /.licenses (см. раздел "**Добавление лицензий**").

Примечание. Чтобы лицензирование работало по сети, сервер загрузки (машина с лицензиями) должен выполнить утилиту nameloc. Если машина загружается с локального жесткого диска (с лицензиями), но не в сети - например, портативный компьютер, - то она также должна выполнить утилиту nameloc, чтобы работало лицензирование.

Лицензии старого и нового стиля

Данная версия ОС QNX поддерживает параллельно две схемы лицензирования: старого стиля (более ранние, чем QNX 4.23) и нового стиля (QNX 4.23 и более поздние). Все программные продукты QNX 4, выпущенные до QNX 4.23, используют схему лицензирования старого стиля.

Старый стиль

Лицензии старого стиля хранятся в каталоге /etc/licenses. Каждой лицензии соответствует файл, и они всегда распространяются на дискетах. Лицензии старого стиля могут устанавливаться и переноситься с диска на диск исключительно при использовании утилиты license.

Новый стиль

Лицензии нового стиля хранятся в файле /.licenses. Они могут поставляться на дискете, но в большинстве случаев номера лицензий записаны на бумаге как лицензионный сертификат. Лицензии нового стиля должны быть установлены на диск использованием утилиты license, которая добавляет их к файлу /.licenses. Вы можете добавлять лицензии нового стиля из сертификата в файл /.licenses, используя любой текстовый редактор. Файл /.licenses может быть перенесен с диска на диск с использованием команды cat (см. в этой главе раздел "**Перенос лицензии на другой узел**").

Файл /.licenses

Лицензии нового стиля сохраняются в файле /.licenses. Ниже приведен пример (не действующий) файла /.licenses. Он содержит лицензии для шести программных продуктов, включая ОС QNX:

```
  qnx.00090209-02lg-0947-48g2-00p7-0044 (4 node)
  qnx.00035882-02lg-0947-48g2-00p7-0044 (4 node)
  wcc.00375634-0l04-4k0l-0x6l-6112-5409 (4 node)
 phab.00006233-0040-0527-00l4-ji3g-1130 (4 node)
 phrt.00006932-007l-8070-g140-l410-84n3 (4 node)
 xrun.00004746-0l04-4l0k-0x7o-5514-8609 (4 node)
motif.00053489-00lk-0245-44e9-04i4-0004 (4 node)
```

Элемент *имя_продукта.nnnnnnnn* в начале каждой строки содержит серийный номер, принадлежащий указанному продукту.

Все лицензирование QNX выполнено на основе числа узлов. В вышеуказанном примере устанавливаются две лицензии QNX, и с каждой из них могут выполняться четыре параллельных сеанса QNX. Это значит, что одновременно можно запустить максимум восемь копий QNX. Если вы попытаетесь запустить девятый узел, то он не сможет взаимодействовать в сети.

Каталог etc/licenses

Лицензии старого стиля хранятся в каталоге /etc/licenses. Каждая лицензия представляет собой индивидуальный дисковый файл, имя которого можно увидеть, используя команду ls:

```
ls /etc/licenses
```

QNX сообщает лицензионную информацию старого стиля в следующем формате:

```
qnx0000178n004
```

где "qnx" - имя программного продукта, "0000178" - серийный номер, "n" - разделитель, а "004" означает, что файл - для четырех узлов. Если вы установили другие приложения QNX (например, Watcom C), то их лицензии тоже будут отображаться.

Проверка лицензий

Утилита licinfo позволяет суперпользователю вывести список всех установленных лицензий. Вы можете задать опции командной строки, чтобы отобразить информацию для узла или индивидуального программного продукта следующим образом:

`licinfo` (выводит все лицензии, используемые в сети);

`licinfo -a` (выводит все лицензии, независимо от того, используются они в сети или нет);

`licinfo -l wcc` (выводит все лицензии компилятора);

`licinfo -n 1` (выводит все лицензии на узле 1).

Добавление лицензий

В сети лицензии должны быть установлены на сервере начальной загрузки. Если вы хотите загрузить несколько компьютеров с их собственных жестких дисков, то вы должны устанавливать все лицензии на все те машины, которые будут загружаться с жесткого диска. Вновь установленные лицензии впоследствии должны быть активированы, чтобы QNX "узнала" о них (смотри ниже раздел "Активация лицензий").

Лицензии нового стиля

Когда вы приобретаете новое программное обеспечение, ваша лицензия (лицензии) поставляется или на дискете, или на лицензионном сертификате. Лицензии на дискете добавляются в файл /.licenses автоматически во время процесса установки. Если вы получили сертификат, то на нем имеется один или несколько лицензионных номеров нового стиля. Вы должны будете добавлять эти лицензионные номера в файл /.licenses вручную, используя текстовый редактор.

Примечание. Не забудьте активировать лицензии на вашем сервере начальной загрузки и на всех узлах, которые загружались с этого сервера, используя license -r.

Лицензии старого стиля

Приведенная ниже команда устанавливает лицензию с дискеты в 0-ом накопителе на жесткий диск. Обратите внимание, что предварительно должен быть запущен драйвер гибкого диска (Fsys.floppy):

```
license
```

Активация лицензий

После того, как вы установите лицензию (обычно на сервере начальной загрузки), то ее нужно активировать. Чтобы это сделать, запустите на узле (или сервере начальной загрузки) утилиту license **-r**, которая считывает все лицензии из /.licenses и /etc/licenses (если такой файл и каталог существуют) и помещает их в базу данных лицензий компьютера.

Узлы, которые загружаются через сеть, будут наследовать базу данных с лицензиями от своего сервера начальной загрузки. Вы можете выполнить следующую команду, чтобы обновить лицензионную информацию узла из базы данных лицензий сервера начальной загрузки:

```
license -R ID_узла_сервера_начальной_загрузки
```

Перенос лицензии на другой узел

Лицензии старого стиля можно переносить с одного узла на другой, используя команду license. Например, следующая команда перенесет все лицензии из каталога /etc/licenses на узле 61 в каталог /etc/licenses на узле 71:

```
license //61/etc/licenses //71/etc/licenses
```

Лицензии нового стиля хранятся в файле /.licenses. Чтобы копировать этот файл с узла на узел, используйте cat (а не cp) с тем, чтобы случайно не перезаписать содержимое файла. Например, следующая команда *добавит* содержимое файла /.licenses на узле 1 к содержимому файла /.licenses на узле 5:

```
cat //1/.licenses  >>//5/.licenses
```

Примечание. Вы должны защитить ваш файл /.licenses с тем, чтобы администратор системы был единственным пользователем с доступом для чтения/записи в файл.

Введите эту команду как root:

```
chmod 600 /.licenses
```

Глава 6. Создание сети

Данная глава охватывает следующие разделы:

- Введение
- Планирование сети
- Выбор конфигурации сервера начальной загрузки
- Загрузка узла из сети QNX
- Загрузка узла, используя BOOTP-сервер
- Загрузка узла из своего собственного жесткого диска
- Несколько серверов начальной загрузки
- Примеры сетей
- Диагностика сети

Введение

QNX - это сетевая распределенная операционная система. Каждый компьютер, на котором работает QNX, называется *узлом*. Одиночный компьютер можно рассматривать как сеть с одним узлом.

Менеджер сети QNX распознает широкий диапазон пакетов протоколов и передает их прозрачно соответствующим адресатам.

Примечание. Если вам нужна полностью базирующаяся на TCP/IP сетевая связь между машиной с QNX и машиной, на которой QNX не установлена, вам необходимо приобрести и установить TCP/IP для QNX.

Логические ID узлов

Каждому узлу QNX присваивается уникальный логический идентификатор (ID) узла, который представляет собой положительное целое число. Операционная система использует эти идентификаторы, чтобы взаимодействовать с другими узлами, связанными сетью.

Вам следует присваивать логические ID узлов в непрерывной последовательности, начиная с 1 (например, 1, 2, 3, 4, 5, ...). Для этого имеется две причины:

- получение лицензии для QNX основывается на общем числе логических узлов в сети. Пятиузловая сетевая лицензия позволяет взаимодействовать узлам с 1-го по 5-й. Узел 6 будет изолироваться и игнорироваться, даже если один из узлов с 1-го по 5-й был опущен;
- несколько важных утилит (например, nameloc) пытаются взаимодействовать с каждым узлом сети. Они делают это с помощью посылки сообщений, начиная с узла 1 и до узла *n*, где *n* - число установленных сетевых лицензий.

Логические ID сетей

Узлы связаны через одну или более физических сетей. Каждая сеть представляет собой отдельную коммуникационную связь. Сетям, подобно узлам, присваиваются логические ID сети, которые должны быть уникальными.

Единственная сеть (наиболее общий случай) по умолчанию имеет логический ID сети, равный 1. Если у вас есть вторая сеть, присвойте ей ID сети, равный 2.

Физические ID узлов

Хотя процессы QNX работают с логическими ID узлов, сами сетевые платы взаимодействуют, используя физические ID узлов. Эти физические ID обычно содержатся непосредственно в самой сетевой плате. Формат физического ID зависит от типа сети (например, Ethernet, Arcnet, Token Ring) и невидим для большинства приложений. Физические ID при установке сети должны быть уникальными в пределах сети, но они не должны быть уникальными для всех сетей.

Серверы начальной загрузки и загружаемые узлы

Локальная сеть QNX, по существу, представляет собой сеть с равноправными узлами, которую устанавливают в два этапа:

- выбирают конфигурацию сервера начальной загрузки;
- выбирают конфигурацию загружаемых узлов.

Любой узел QNX, имеющий жесткий диск, может загружаться со своего собственного жесткого диска. Сервер начальной загрузки - это просто узел, который обслуживает запросы загрузки от других узлов в сети. Обычно в качестве сервера загрузки выбирается узел, который редко перезагружается.

Когда узел выполняет начальную загрузку через сеть, то он получает образ ОС от сервера начальной загрузки. Этот образ начальной загрузки загружается в память выполняющей начальную загрузку машины, которая обычно наследует от загружаемого образа начальной загрузки следующие параметры:

- время;
- глобальный список символьных имен процессов (смотрите nameloc в книге *Описание утилит*);
- лицензионные возможности.

Вы можете устанавливать вашу сеть с несколькими серверами начальной загрузки, чтобы обеспечить отказоустойчивость (в том случае, если ваш первичный сервер начальной загрузки станет неработоспособен) и распределить ресурсы начальной загрузки так, чтобы

избежать образования узких мест.

Каким образом выполняется начальная загрузка сети

Сетевые платы в загружающемся узле выдают запросы начальной загрузки одним из двух способов:

- посылают запрос начальной загрузки в заданный сервер начальной загрузки

ИЛИ

- широковещательно посылают запрос начальной загрузки ко всем узлам и ожидают, что сервер начальной загрузки ответит на запрос.

Плата Arcnet QNX, имеющая энергонезависимое ОЗУ, в котором можно хранить ID узла сервера, использует точный метод начальной загрузки. Большинство других сетевых плат, таких как Ethernet и Token Ring, не имеют средств для хранения физического ID узла сервера, поэтому они должны использовать широковещательный метод.

Создание "мостов" между локальными сетями QNX

Любой узел сети QNX может действовать как мост между двумя различными сетями QNX, если они обе являются сетями стандарта IEEE 802. Например, создание "моста" QNX позволит присоединить сеть Ethernet к сети Token Ring или к сети FDDI. Это возможно, потому что QNX использует один и тот же формат пакета и протокол на всех сетях, основанных на стандарте IEEE 802. Все, что вы должны сделать, - это присоединить две различные сети к одному и тому же узлу. Узел может быть сервером начальной загрузки или просто загружающейся машиной.

Примечание. QNX не может создать "мост" к сети Arcnet.

Планирование сети

Одна сеть или больше?

Если вы устанавливаете небольшую сеть, то, вероятно, потребуется только один сервер начальной загрузки, обычно узел 1. Если же вы устанавливаете большую сеть, которая будет охватывать несколько отделов, следует устанавливать сервер начальной загрузки для каждого отдела. Пользователи каждого отдела смогут иметь доступ к файлам (при условии разрешения доступа к файлам) в других отделах.

Например, предположим, у вас есть три отдела: техотдел, отдел маркетинга и отдел продаж. Можно выполнять начальную загрузку в каждом отделе из его собственного сервера начальной загрузки, который мог бы также работать в качестве главного файлового сервера всех отделов.

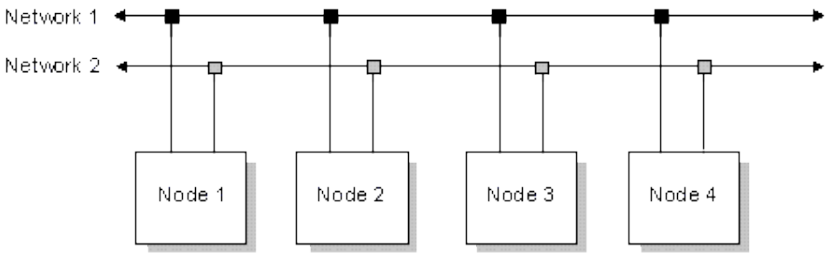
Такая установка создает отказоустойчивую среду - отказ в одном отделе не помешает другому отделу выполнить начальную загрузку. Для большого числа узлов это должно также уменьшить вероятность "узкого места" при начальной загрузке, когда служащие в офисе начинают свою работу с загрузки.

Несколько сетевых связей

Как описано в книге *Системная архитектура*, наличие больше чем одной сетевой связи на компьютере обеспечивает отказоустойчивость и увеличивает производительность.

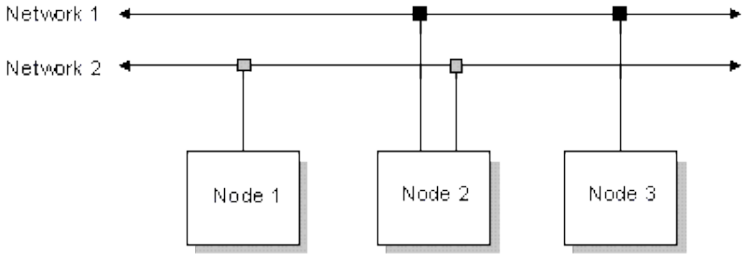
Обмен информацией через сети

В QNX узел может обмениваться информацией одновременно со многими сетями. Рассмотрите следующую диаграмму, где четыре компьютера связываются через две сети:



В этой отказоустойчивой конфигурации, каждый из четырех узлов может напрямую обмениваться информацией с любым другим узлом через сеть 1 или сеть 2.

Сравните это со следующей установкой, где узлы 1 и 3 подключены к различным сетям, а узел 2 является общим для обеих сетей. Узлы 1 и 3 напрямую не соединяются друг с другом, но они оба могут обращаться к узлу 2.

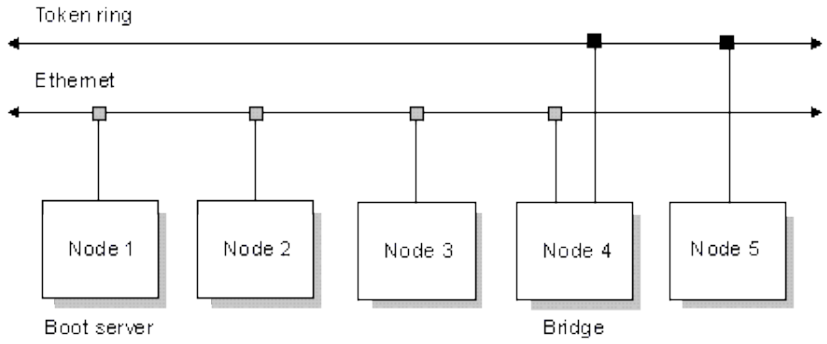


Если узел 2 был соединен с двумя сетями стандарта IEEE 802, то сетевой "мост" QNX будет автоматически связывать две сети через узел 2.

Соображения по установке

Возможно, вам понадобится подготовить схему сети, подобную приведенной ниже. На этой примерной схеме показана сеть, состоящая из пяти узлов. Один из узлов - сервер начальной

загрузки, а три - узлы, которые будут загружаться и получать файлы с сервера начальной загрузки. Узел 5 загружается из своего собственного жесткого диска.



Присвоим логические ID каждому узлу; серверу начальной загрузки - узел 1. Обратите внимание, что узел 4 будет действовать как "мост" между сетью Ethernet (логический ID сети - 1) и сетью Token Ring (логический ID сети - 2).

Примечание. Вы не можете производить начальную загрузку через "мост". В предыдущей диаграмме, например, узел 5 не может быть загружен с узла 1.

Сетевые платы и драйверы

Установка сети обычно включает установку сетевой платы в компьютер и ее соединение кабелем с другими компьютерами, которые имеют тот же самый тип сетевой платы. Ethernet, Arcnet и Token Ring - это самые популярные типы сетей.

Кроме сетевой платы, необходим поддерживающий ее сетевой драйвер QNX. Все сетевые драйверы QNX описаны в книге *Описание утилит*. Их имена начинаются с префикса "Net.".

Платы от различных поставщиков для одних и тех же сетевых сред (например, Ethernet) могут потребовать свои собственные специальные драйверы. Но часто драйвер может поддерживать платы более чем одной компании, если платы поддерживают аналогичный аппаратный интерфейс.

Примечание. Для получения информации относительно аппаратной установки/конфигурации сетевой платы и физического соединения между сетевыми платами, смотрите документы, поставляемые с платой.

Определение физических ID узлов

Метод, используемый для определения физического ID узла сетевой платы зависит от типа имеющейся платы. В большинстве случаев вы найдете этикетку с адресом платы на упаковке. Иногда физический ID узла сетевой платы задается переключателями или перемычками на плате.

Ethernet и Token Ring

Каждая плата Ethernet или Token Ring поставляется со своим собственным физическим ID узла, встроенным в плату. Этот ID, который является совершенно уникальным, имеет длину 48 битов и соответствует стандарту IEEE 802.

Для плат Ethernet можно найти значение ID где-нибудь на самой плате или на упаковке. Формат адреса ID может меняться. Например, все ID, указанные ниже, одинаковые:

```
0000c0 4a9330
0000c04a9330
00 00 c0 4a 93 30
00:00:c0:4a:93:30
0000 c04a 9330
```

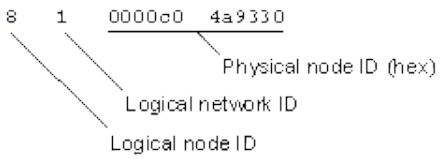
Arcnet

Каждая плата Arcnet требует, чтобы вы запрограммировали физический ID узла на плате. В зависимости от изготовителя, это делается посредством DIP-переключателя или записью в энергонезависимое ОЗУ с помощью интерфейсного меню во время начальной загрузки.

Физический ID имеет длину 8 битов. Нельзя использовать физический ID, равный 0, который зарезервирован. При выборе физического ID узла рекомендуется для единственной сети Arcnet установить физический ID таким же, как логический ID узла. Это упростит вашу работу.

Присвоение логических ID узлам и сетям

Так как процессы QNX используют логические ID, Менеджер сети (Net) должен установить соответствие между логическими ID и физическими ID, используемыми аппаратными средствами, - карту сети. Эта карта сети задается в файле /etc/config/netmap. Каждая строка в этом файле определяет соответствие для одного узла: первыми в строке задаются логические ID узла и сети, за ними следует физический ID узла. Например, следующая строка устанавливает соответствие логического узла 8 в логической сети 1 48-битному физическому ID 0000c4a9330:



Вы можете разделить логический и физический ID символами пробела или табуляции. Логический ID - в десятичный, а физический ID - 16-ричный, если ему не предшествует "t", - в этом случае он также десятичный.

Например, в случае использования Arcnet более удобно представлять физический ID узла в десятичном виде:

```
15   1   t15
```

Физический ID не может превышать 48 битов. Требуемое количество битов определяется сетевым аппаратным обеспечением.

Выбор конфигурации сервера начальной загрузки

В этом разделе показано, как устанавливать одиночный сервер начальной загрузки. Если одиночный сервер установлен, можно устанавливать загружающиеся узлы. Если требуются дополнительные серверы начальной загрузки, читайте раздел "**Несколько серверов начальной загрузки**".

Примечание. Эта процедура допускает, что вы уже сделали базовую установку QNX на жестком диске машины, как это указано в главе "**Базовая установка**". Следовательно, ваша машина уже была сконфигурирована для того, чтобы загрузиться со своего собственного жесткого диска.

Превращение узла QNX в сервер начальной загрузки включает в себя следующие шаги:

- установка сетевой платы (плат);
- установка ваших сетевых лицензий;
- запуск Менеджера сети и сетевого драйвера (драйверов);
- запуск утилиты nameloc;
- запуск утилиты netboot;
- модификация файла sysinit.*узел*;
- модификация файла netmap;
- модификация файла netboot.

Шаг 1. Установка сетевой платы (плат)

Выключите компьютер, затем установите сетевую плату согласно инструкциям, прилагаемым к плате. Если возможно, определите физический ID платы. Когда закончите, перезагрузите компьютер.

Примечание. Если вы не смогли найти физический ID узла при установке платы, или если его не было на плате, то запустите (после запуска сетевых драйверов) утилиту netmap без каких-либо опций, и она отобразит физический ID узла.

Установка нескольких плат

Когда вы устанавливаете две одинаковые платы в одной и той же машине, не полагайтесь на автоматическое обнаружение по умолчанию портов ввода/вывода и т.п. Непременно

проверьте уникальность установочных параметров - можно определить кое-что из этой информации, используя опции командной строки в сетевом драйвере. Платы следует сконфигурировать таким образом, чтобы они не имели никаких аппаратных конфликтов.

Прежде чем установить плату (платы), прочитайте технические замечания для типа устанавливаемой платы (плат). Файлы в каталоге /etc/readme/technotes содержат полезные советы, типа того, как установить аппаратные прерывания, чтобы избежать возможных конфликтов.

Шаг 2. Установка ваших сетевых лицензий

Вам нужна сетевая лицензия для каждого узла сети. Когда вы первоначально установили QNX на жестком диске, то была установлена лицензия, по крайней мере, для отдельного узла. Можно проверить, сколько лицензий было установлено, используя команду licinfo. Для получения дополнительной информации относительно установки и активизации лицензий читайте главу "**Лицензирование**".

Шаг 3. Запуск Менеджера сети и сетевого драйвера(ов)

Если вы сообщили install, что данный узел должен быть сервером начальной загрузки, то соответствующие сетевые сервисные процессы должны быть запущены. Пропустите этот шаг и идите далее.

Запуск Менеджера сети и одного сетевого драйвера

Менеджер сети (Net) и сетевой драйвер вашей сети должны быть запущены, чтобы иметь доступ к логической сети 1.

Имена для всех сетевых драйверов имеют форму Net.*xxx*, где *xxx* представляет собой соответствующий драйвер. Например, если вы имеете плату Ethernet, совместимую с NE2000, то ваш ввод будет похож на следующий:

```
Net &
Net.ether1000 &
```

Чтобы выяснить, какие сетевые драйверы доступны в вашей системе, введите команду:

```
ls /bin/Net.*
```

Запуск двух или более сетевых драйверов

Если вы установили две сетевые платы, необходимо запустить драйвер для второй сети.

По умолчанию все сетевые драйверы связаны с логическим ID сети, равным 1. При создании нескольких сетей, вы должны задать опцию **-l** в командной строке запуска драйвера, чтобы иметь драйвер, связанный с сетью иной, чем логическая сеть 1.

Например, следующие три команды в файле sysinit.*узел* сервера начальной загрузки запустят Менеджер сети, драйвер Ethernet для логической сети 1, а драйвер Token Ring для логической сети 2:

```
Net &
Net.ether1000 &
Net.tr164a -l 2 &
```

Заметьте, что Менеджер сети должен запускаться с опцией **-d** всякий раз, когда запускаются три или большее число сетевых драйверов.

Эта команда заставляет nettrap просканировать машину, на которой запущена, и найти сетевые платы:

```
nettrap
```

Примечание. Утилита nettrap считывает и записывает порты ввода/вывода и память. Это может вызвать неумышленные побочные эффекты в других аппаратных средствах, размещенных в проверяемой памяти и адресах ввода/вывода. Если ваша система установлена для управления внешним оборудованием, отключите эти устройства перед выполнением nettrap.

Утилита nettrap также может быть использована для запуска Net вместе с соответствующими сетевыми драйверами (и netmap -f):

```
nettrap start
```

Шаг 4. Запуск утилиты nameloc

Утилита nameloc выполняется в фоновом режиме и обеспечивает в масштабах сети систему идентификации имен для всех процессов, выполняющихся под ОС. Для обеспечения работы лицензионных продуктов, утилита nameloc должна быть запущена, как минимум, на одной машине в сети, даже если это *единственная* машина в автономной "сети":

```
nameloc &
```

Запустившись, nameloc опрашивает каждый узел в сети для составления списка глобальных символических имен процессов. Для получения дополнительной информации смотрите описание утилиты nameloc в книге *Описание утилит*.

Примечание. Машина, выполняющая nameloc, *должна* быть способна взаимодействовать с

любой другой машиной в сети. Убедитесь, что компьютер имеет *полный* файл netmap. Если nameloc выполняется на машине, не имеющей полного доступа в сеть, то вы встретитесь со всеми разновидностями проблем - неверной информацией о лицензиях, неполным списком глобальных имен, неправильной работой сетевых утилит и т.п.

Шаг 5. Запуск утилиты netboot

Когда загружающийся узел запускается, его ПЗУ начальной загрузки посылает запрос о загрузке на сервер начальной загрузки или он пересылает запрос о загрузке на любой сервер начальной загрузки, сконфигурированный для ответа. Когда Менеджер сети на сервере начальной загрузки получает запрос о загрузке, он передает этот запрос утилите netboot. Чтобы запустить netboot, введите:

```
netboot &
```

При ответе на запрос о начальной загрузке утилита netboot обращается к файлу /etc/config/netboot, чтобы определить, какой из файлов построения может использоваться, чтобы сгенерировать образ ОС для запрашивающего узла.

Примечание. Выполнение netboot с опцией **-v** ("многословный" режим) может быть полезно для поиска неисправностей. Многократное применение опции **v** увеличит уровень информированности - при этом вы можете направлять вывод в файл или вывести его на своей собственной консоли.

Вы можете запустить утилиту netboot с опцией **-a**, чтобы помочь автоматизировать создание карты сети (смотри Шаг 7):

```
netboot -a &
```

В том случае, если файл /etc/config/netmap не содержит соответствующих установок для запрашивающего узла, то netboot автоматически запишет необходимую строку карты сети в файл /etc/config/netmap, используя ближайший доступный логический ID узла и логический ID сети, и запрашивающий узел подключится.

Шаг 6. Модификация файла sysinit.*узел*

Теперь, когда запущены все требуемые сетевые драйверы и сервисные процессы, узел может функционировать как сервер начальной загрузки. Если добавить команды запуска сети в файл sysinit.*узел* сервера начальной загрузки, они будут автоматически запускаться всякий раз, когда узел перезагружается.

Сначала скопируйте файл sysinit.*узел* сервера начальной загрузки (например, sysinit.1) в altsysinit прежде, чем вы продолжите:

```
cp /etc/config/sysinit.1 /etc/config/altsysinit
```

Это сохранит альтернативную копию файла инициализации сервера начальной загрузки. Теперь добавьте требуемые записи в /etc/config/sysinit.*узел*. Для сервера начальной загрузки требуются записи, похожие на следующую:

```
Net &
Net.ether1000 &
nameloc &
netboot &
```

Шаг 7. Модификация файла netmap

Файл /etc/config/netmap является по умолчанию файлом карты сети ID узлов и сетей, и используется утилитами netboot и netmap. Этот файл устанавливает соответствие физического ID узла, логического ID узла и логического ID сети для каждого узла.

Образец файла поставляется с QNX. Используемый файл, должен содержать логические ID узлов и сетей для сервера начальной загрузки и для всех сетевых узлов, взаимодействующих с сервером начальной загрузки. Соответствующие записи в файле выглядят примерно так:

ID узла	ID сети	Адрес сетевой платы
1	1	*Физичечкий_адрес_загрузочного_сервера*
2	1	*Физический_адрес_узла2*
3	1	*Физический_адрес_узла3*
4	1	*Физический_адрес_узла4*

Заметьте, что когда имеется единственная сеть, логический ID сети обычно равен "1". Все узлы сети должны иметь свои собственные уникальные логические ID узлов. Следует выбирать логический ID узла, равный 1, для вашего первого (первичного) сервера начальной загрузки. Для узла, который взаимодействует с двумя или более сетями, один тот же логический ID узла будет иметь две записи, каждая из которых будет иметь свой уникальный логический сетевой идентификатор.

Файл netmap можно редактировать вручную, или, если запущена утилита netboot с опцией **-a**, то система будет автоматически записывать данные об узле в файл netmap, как только вы включаете любой узел. Узлу, который будет включен первым, будет присвоен следующий свободный логический ID узла, правильный логический ID сети и правильный физический адрес. Если задается опция **-A**, то новому узлу будет присваиваться наименьший свободный логический ID узла и правильный физический адрес.

Если имеются узлы, загружающиеся с их собственных жестких дисков, то можно сгенерировать модифицированный "главный" файл /etc/config/netmap на узле 1 следующим образом:

1. На всех машинах, за исключением узла 1, создайте локальный файл netmap, содержащий карту сети логических и физических ID для данного узла и узла 1
2. На всех узлах, за исключением узла 1, выдайте следующую команду, чтобы откорректировать сохраняемую в памяти карту сети на узле 1:

```
sin -n1
```

3. Чтобы сохранить из памяти карту сети на диске узла 1, наберите на клавиатуре:

```
netmap > /etc/config/netmap
```

4. Чтобы перенести карту сети на другие узлы, можно скопировать главный файл с узла 1. Например, чтобы скопировать главный файл на узел 2, наберите на клавиатуре:

```
cp //1/etc/config/netmap //2/etc/config/netmap
```

Карта сети для нескольких сетей

Для того чтобы определить узел, имеющий более одной сетевой платы, добавьте одну запись, чтобы соответствовать каждой сети. Помните, логические сетевые ID должны быть уникальны. Например, если сеть имела пятый узел, соединенный с двумя различными сетями, надо добавить еще две записи в файл netmap:

ID узла	ID сети	Адрес сетевой платы
5	1	*Физический_адрес_платы1_узла5*
5	2	*Физический_адрес_платы2_узла5*

Для получения дополнительной информации о файле netmap смотрите утилиту netmap в книге ***Описание утилит***. Примеры, описывающие карту сети для нескольких сетей, найдете в разделе "**Примеры сетей**" в этой главе.

Примечание. Каждый раз при редактировании файла netmap следует также откорректировать сохраняемые в памяти сетевые таблицы:

```
netmap -f
```

Шаг 8. Модификация файла netboot

Когда сервер начальной загрузки получает запрос о начальной загрузке от узла, Менеджер сети отправляет запрос вместе с физическим ID узла и логическим ID сети утилите netboot.

Имея эту информацию, утилита qqqЦ2netbootqqqЦ0 обращается к файлу карты сети (/etc/config/netmap), чтобы определять логический идентификатор узла (загружаемой машины). Затем утилита обращается к файлу /etc/config/netboot, чтобы определить, какой файл построения можно использовать, чтобы сгенерировать образ ОС для узла.

Если используются платы, совместимые с NE2000, ваш файл /etc/config/netboot в сервере начальной загрузки может содержать строку, аналогичную следующей:

```
f=build/ws.ether1000
```

Если вы первоначально задали в install, что данный узел должен быть сервером начальной загрузки, то файл netboot уже будет содержать то, что вам нужно.

Вы можете изменить эту запись f=, если хотите, чтобы QNX использовал специальный файл построения. Например, если вы скопировали один из файлов построения, поставляемых с QNX, переименовали его в ws.mybuild, а затем определили дополнительные сервисные средства в данном заказном файле построения, то вы можете изменить запись так, чтобы она выглядела следующим образом:

```
* f=build/ws.mybuild
```

Если некоторое количество узлов оборудовано одним и тем же видом сетевой платы, для всех этих узлов может быть использована одна запись. В приведенном выше примере записи все (*) узлы на сети будут загружаться из одного и того же файла построения (ws.mybuild).

Если сервер начальной загрузки будет обслуживать запросы о начальной загрузке от двух или более различных узлов и/или сетей, можно задавать различные заказные файлы построения для некоторых из ваших узлов. В каталоге /boot/build предусмотрены файлы построения образов ОС и для других сетевых сред, например, таких как Token Ring.

Например, давайте представим, что имеется восьмиузловая сеть Ethernet, совместимая с NE2000, и что эти узлы загружают из netboot запись f=build/ws.ether1000. Если добавить еще три узла, которые будут загружаться из того же самого сервера, используя заказные файлы построения, можно изменить /etc/config/netboot таким образом, что он будет выглядеть примерно так:

```
9     f=build/ether2100.node9
10    f=build/ws.tr16a
11    f=build/mybuild.node11
*     f=build/ws.ether1000
```

В вышеуказанном примере узлы 9, 10 и 11 используют заказные файлы построения, а все другие узлы используют по умолчанию сетевой файл построения ws.ether1000.

Примечание. Позиция строки с * важна! Удостоверьтесь, что строка, начинающаяся с *, является *последней* строкой в /etc/config/netboot, поскольку обработка файла прекратится, когда будет обнаружено первое соответствие.

Установка для широковещательной загрузки

При широковещательной загрузке с ПЗУ утилита netboot должна знать, в качестве какого сервера она выступает по отношению к каждому узлу (первичного, вторичного или третичного). Утилита выясняет это, считывая необязательный параметр режима сервера в файле netboot.

Общий синтаксис записей netboot следующий:

```
логический_ID_узла f=файл_построения | F=файл_образа [режим_сервера]
```

где *режим_сервера* определяет роль, которую утилита netboot должна играть по отношению к узлу. Если *режим_сервера* не задан, то netboot будет действовать для данного узла как первичный сервер начальной загрузки.

Примечание. Опция F=*файл_образа* позволяет указать имя предварительно построенного заказного образа ОС, который будет загружен в узел из сети. По умолчанию, QNX строит образ "на лету", используя заданный файл построения. Дополнительную информацию о передаче образа ОС загружающемуся узлу, вместо именованного, читайте в разделе "**Загрузка узла из сети**".

Возможные значения для *режим_сервера*:

```
A
```

первичный сервер (по умолчанию)

```
B
```

вторичный сервер

```
C
```

третичный сервер

В следующем примере задается, что netboot должен быть первичным сервером начальной загрузки для всех узлов, за исключением узла 4, для которого он будет вторичным сервером начальной загрузки:

```
4  f=build/ws.tr16a      B
*  f=build/ws.ether1000  A
```

Дополнительную информацию о выполнения netboot на нескольких серверах начальной загрузки читайте в разделе "**Несколько серверов начальной загрузки**".

Загрузка узла из сети QNX

Когда узел, загружаемый из сети QNX, включается, то сначала выполняется программа начальной загрузки QNX, находящаяся в ПЗУ на сетевой плате машины. При этом либо устанавливается связь с конкретным сервером начальной загрузки (если это плата QNX Arcnet), либо делается широковещательный запрос о начальной загрузке.

Процесс netboot на ответившем сервере начальной загрузки генерирует и загружает образ начальной загрузки в ОЗУ сетевой платы загружающегося узла. Затем запускается программа sinit и начнется запуск сервисных средств, определенных в файле sysinit.*узел* для данного узла.

Конфигурирование узла, загружаемого из сети QNX, включает следующие шаги:

- установка ПЗУ начальной загрузки QNX;
- установка сетевой платы;
- создание файла построения;
- модификация файла sysinit.*узел*;
- загрузка узла.

Шаг 1. Установка ПЗУ начальной загрузки

Чтобы иметь возможность загрузки QNX с плат Ethernet или Token Ring, следует установить ПЗУ начальной загрузки QNX в гнездо ПЗУ на плате, как это описано в документации на плату.

Обратите внимание, что платы QNX Arcnet поставляются с уже установленным специальным ПЗУ начальной загрузки QNX. Это ПЗУ начальной загрузки не делает широковещательный запрос начальной загрузки; вместо этого оно входит в контакт с заданным сервером начальной загрузки.

Все другие ПЗУ начальной загрузки QNX используют метод широковещательной начальной загрузки. Когда ПЗУ загрузки QNX начинает загружаться, оно передает запрос о загрузке в сервер, который имеет символическое имя "A" (первичный сервер); если оно не получает никакого ответа, ПЗУ начнет транслировать запрос о загрузке в сервер "B" (вторичный сервер), и, наконец, в сервер "C" (третичный сервер).

Если в узле устанавливаются две сетевых платы с ПЗУ начальной загрузки, то при выполнении последовательности самотестирования, выполняемой BIOS, сетевая плата, чье ПЗУ размещается в *старшем* адресе памяти, обнаруживается последней. Обычно это та плата, которая будет использоваться вашим компьютером для загрузки через сеть.

Шаг 2. Установка сетевой платы

Если вы этого еще не сделали, прочитайте техническое замечание, относящееся к типу

устанавливаемой платы (смотрите /etc/readme/technotes).

Следуя установочным инструкциям изготовителя, не забудьте обратить внимание на физический ID узла платы (плат). При загрузке ПЗУ начальной загрузки QNX будет печатать физический ID узла.

Шаг 3. Создание файла построения

Вы можете настроить сервисные средства, которые наследует каждый узел с помощью файла построения. Каждый узел может иметь отличный от других файл построения, который либо заранее конфигурируется и сохраняется на сервере начальной загрузки (в каталоге /boot/build) либо создается "на лету" с помощью утилиты buildqnx.

По умолчанию образы базируются на файлах построения, заданных в файле netboot сервера начальной загрузки и создаются "на лету". Когда вы создаете заказной образ операционной системы, то запись F=*файл_образа* должна заменить стандартную запись f=*файл_построения* в файле netboot сервера начальной загрузки.

Примечание. Net и вспомогательный драйвер должны запускаться в файле построения загружающихся узлов. Если узлу будет нужно более двух сетевых драйверов, то в файле построения для Net следует задать опцию -d. Для получения более подробной информации обратитесь к документации Net в книге ***Описание утилит***.

Если вы желаете изменить файл построения узла, то вам следует сделать копию его текущей версии и задать копии значащее имя, например, такое, как ws.eth1000_tr16a для файла построения, который содержит сетевой драйвер Ethernet 1000 и сетевой драйвер Token Ring, а потом модифицировать новый файл.

Шаг 4. Модификация файла sysinit.*узел*

Если вы хотите настроить сервисные средства, доступные на узле, то создайте для узла файл sysinit.*узел* на сервере начальной загрузки и добавьте в него требуемые сервисные средства (см. раздел "Файл инициализации системы" в главе "**Базовая установка**").

Помните, суффикс *узел* должен быть логическим ID узла машины, которую вы настраиваете.

Шаг 5. Загрузка узла

Узел теперь будет загружаться из сети. Если у вас имеются какие-либо проблемы, то в файле построения загружающегося узла задайте опцию **-v** в Net. Это позволит отобразить сетевые ошибки на консоли. При этом хорошо бы использовать опцию **-v** для netboot и для сетевых драйверов (Net.*).

Загрузка узла с использованием BOOTP-сервера

Чтобы загружать систему QNX, используя протокол начальной загрузки BOOTP Интернет, необходимо иметь BOOTP-ПЗУ для клиента QNX и BOOTP-сервер (bootpd) для вашего сервера. Так как протокол TFTP используется для перемещения образа от сервера до пользователя, то вам также будет нужен TFTP-сервер, который обычно предоставляется BOOTP-сервером на большинстве систем (UNIX и Windows 95/NT). Рабочий пакет QNX 4 TCP/IP включает сервер bootpd и сервер tftpd.

QSSL поддерживает ПЗУ от Lanworks Technologies (www.lanworks.com). QSSL протестировала Lanworks-ПЗУ с BOOTP-сервером Win32 от Weird Solutions (www.weird-solutions.com) и сервером bootpd QNX 4.

При использовании Lanworks BOOTP-ПЗУ образ должен помещаться в память по адресу 0x10000, так что нужно будет запустить buildqnx с опцией **-b**, чтобы задать адрес в ОЗУ:

```
buildqnx -b 0x10000
```

Для получения дополнительной информации относительно установки BOOTP-сервера QNX 4 смотрите описание bootpd в книге *Руководство пользователя TCP/IP* для QNX.

Загрузка узла со своего собственного жесткого диска

Ваша сеть может включать узел, который будет загружаться со своего собственного жесткого диска. Если это так, то прежде чем загружать узел, нужно сделать следующее:

- установить сетевую плату, как это объяснено ранее;
- отредактировать файл построения узла;
- скопировать главный файл netmap на узел;
- установить или перенести лицензию на жесткий диск узла.

Редактирование файла построения узла

Первые записи в файле построения запускают Менеджер процессов (например, sys/Proc32) и устанавливают опции, которые будет передаваться Proc32. Опция **-l** для Proc32 присваивает логический ID узла.

Когда вы устанавливали QNX на жестком диске данного узла, то предполагалось, что логический ID узла равен 1. Следует изменить логический ID узла, задаваемый опцией **-l**, чтобы он соответствовал суффиксу *узел* файла sysinit.*узел* узла.

Для получения дополнительной информации относительно построения пользовательского образа для узла смотрите раздел "Сетевые образы" в главе "**Построением образа ОС**".

Копирование файла netmap в узел

Машина, загружаемая со своего собственного жесткого диска, нуждается в файле netmap, содержащем карту сети, устанавливающей соответствие между логическими ID и физическими ID, для *всех* узлов, с которыми эта машина должна взаимодействовать.

Можно использовать модифицированный "главный" файл /etc/config/netmap сервера начальной загрузки, как это описано в Шаге 7 раздела "**Выбор конфигурации сервера начальной загрузки**".

Установка или перенос лицензий

Чтобы узел надлежащим образом загрузился с собственного жесткого диска, на жестком диске должна иметься лицензия. Если вы установили лицензию QNX нового стиля с дистрибутива, то лицензия для узла будет установлена в файле /.licenses. Если вы получили лицензионный сертификат, то следует добавить номер лицензии в файл /.licenses. Не забудьте активизировать новые лицензии (используя license -r). Для получения дополнительной информации смотрите главу " **Лицензирование**".

Несколько серверов начальной загрузки

Так как при широковещательном методе загрузки доступны все узлы в сети, то вы можете запускать netboot на нескольких узлах, создавая несколько серверов начальной загрузки. Это позволит вам задавать разным группам узлов разные серверы начальной загрузки.

Хотя каждый процесс netboot обычно получает запрос широковещательной загрузки, но он реагирует только в соответствии с его уникальным файлом /etc/config/netboot, который определяет отношение к каждому запрашивающему узлу (например, "Я - вторичный сервер загрузки для узла 5").

Никакие два сервера начальной загрузки не могут иметь один и тот же *режим_сервера* для данного логического ID узла. Если это правило не выполняется, то каждый сервер загрузки с одним и тем же *режим_сервера* для данного логического ID узла ответит на запрос загрузки этого узла.

Давайте теперь рассмотрим 15-узловую сеть Ethernet, в которой узлы с 1 по 9 принадлежат к техническому отделу, а узлы с 10 по 15 принадлежат к отделу маркетинга. Если вы захотели сделать узел 1 первичным сервером начальной загрузки для техотдела, а узел 10 первичным сервером начальной загрузки для отдела маркетинга, то вам следует запустить утилиту netboot на обоих узлах, но каждую с ее собственным конфигурационным файлом netboot:

На узле 1 `(//1/etc/config/netboot)`:

```
2  f=build/a1.1000       A
```

```
3  f=build/pat.1000      A
4  f=build/celeste.1000  A
5  f=build/robin.1000    A
6  f=build/vanessa.1000  A
7  f=build/sandy.1000    A
8  f=build/john.1000     A
9  f=build/bubba.1000    A
*  f=build/ws.ether1000  B
```

На узле 10 `(//10/etc/config/netboot)`:

```
10  f=build/andrew.1000  A
11  f=build/bob.1000     A
12  f=build/liz.1000     A
13  f=build/fred.1000    A
14  f=build/joe.1000     A
15  f=build/betty.1000   A
*  f=build/ws.ether1000  B
```

В этом случае вы также сделали узел 10 вторичным сервером начальной загрузки для техотдела, а узел 1 - вторичным сервером начальной загрузки для отдела маркетинга.

Примечание. Удостоверьтесь, что строка, начинающаяся с *, является *последней* строкой в файле /etc/config/netboot, поскольку файл прекратится обрабатываться, как только будет обнаружено первое соответствие.

Еще помните, что запись *режим_сервера* в строке * должна указывать иную роль, нежели все другие записи для всех других netboot, запущенных на всех остальных узлах в сети. network.

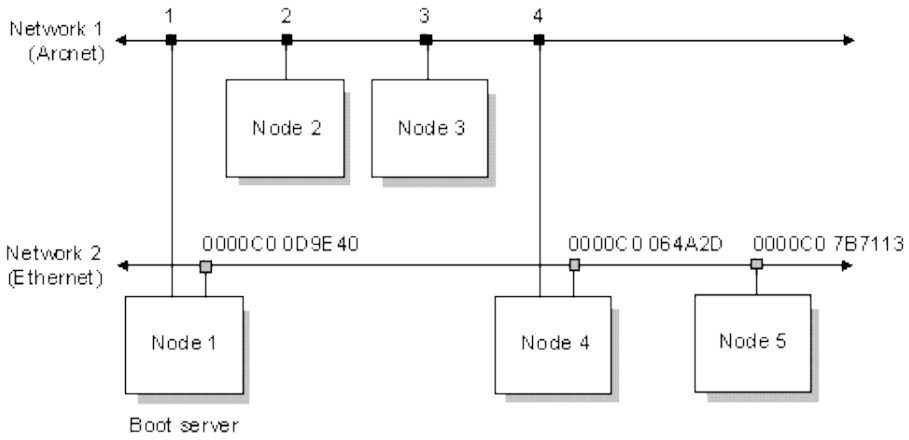
Примеры сетей

Теперь, когда мы обсудили, что входит в установку нескольких сетевых соединений, давайте рассмотрим, как:

- добавить несколько узлов сети Ethernet к сети Arcnet;
- установить отказоустойчивую сеть Ethernet;
- установить частную сетевую связь.

Добавление нескольких узлов сети Ethernet к сети Arcnet

Допустим, имеется следующая компоновка сети:



Как вы помните, формат файла /etc/config/netmap выглядит следующим образом:

логический_ID_узла логический_ID_сети физический_ID_узла

Для данного примера файл netmap будет содержать следующие записи:

```
1           1           t1
1           2           0000C0 0D9E40
2           1           t2
3           1           t3
4           1           t4
4           2           0000C0 064A2D
5           2           0000C0 7B7113
```

Имейте в виду, что логические ID узлов должны быть уникальны для обеих взаимосвязанных сетей.

Для начальной загрузки узлов утилите netboot требуется, чтобы каждый узел имел образ ОС или связанный с ним файл построения. Узлы, загружаемые через сеть Arcnet, не используют файлы построения узлов, загружаемых через Ethernet. Следовательно, файл /etc/config/netboot будет содержать следующие записи:

```
2   f=build/ws.arcnet
3   f=build/ws.arcnet
4   f=build/ws.arc_ether1000
5   f=build/ws.ether1000
```

Узел 1 не имеет своей записи, т.к. он загружается со своего собственного жесткого диска. Однако, из его файла sysinit.*узел* должен запускаться как драйвер Arcnet, так и драйвер Ethernet.

Можно сконфигурировать узел 4, чтобы загружаться или из Arcnet, или из Ethernet.

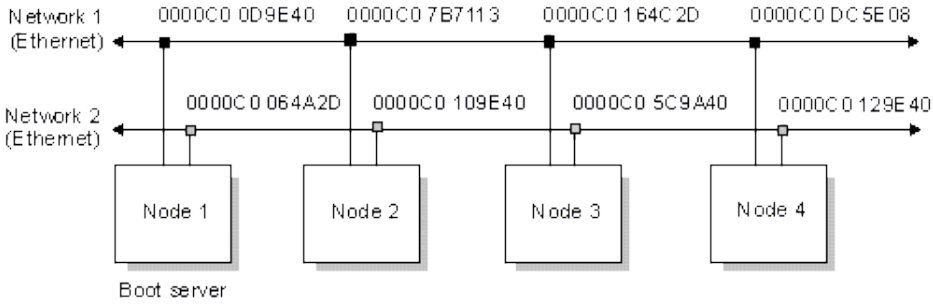
Примечание. Выбор сети, через которую загружается узел 4, зависит от того, какая сетевая плата обнаружена *последней* во время сканирования ПЗУ при включении питания; при этом плата с самым старшим адресом всегда обнаруживается последней и она будет использоваться компьютером для начальной загрузки.

В этом примере будем считать, что узел 4 конфигурируется для загрузки через Arcnet путем установки самого старшего адреса для ПЗУ начальной загрузки Arcnet. Поскольку узел 4 находится как в сети Arcnet, так и в сети Ethernet, то файл построения узла запускает Net и оба драйвера - Net.ether1000 и Net.arcnet. Это обеспечит передачу сервером начальной загрузки файла sysinit.*узел* по каждой сети.

Большинство файлов построения должны запускать Менеджер сети (Net) и связанные с ним сетевые драйверы. Если требуется более двух драйверов сети, следует задать опцию **-d** в Net.

Установка отказоустойчивой сети Ethernet

Допустим, имеется следующая компоновка сети:



Соответствующий файл /etc/config/netmap будет содержать следующие записи:

```
1           1           0000C0 0D9E40
1           2           0000C0 064A2D
2           1           0000C0 7B7113
2           2           0000C0 109E40
3           1           0000C0 164C2D
3           2           0000C0 5C9A40
4           1           0000C0 DC5E08
4           2           0000C0 129E40
```

Соответствующий файл /etc/config/netboot будет содержать следующую запись:

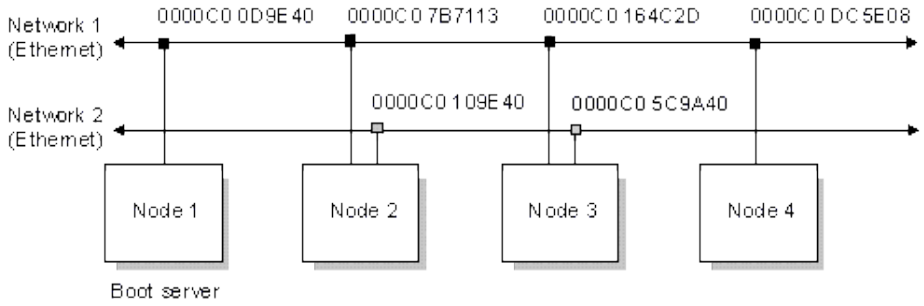
```
* f=build/ws.eth1000x2
```

Так как требуется дополнительный драйвер Ethernet, то следует создать заказной файл

построения (например, ws.eth1000x2) и добавить в файл вторую копию драйвера Net.ether1000, задавая опцию **-l 2**.

Установка частной сетевой связи

Если между определенными узлами сети существует напряженный трафик, можно объединить эти узлы частной сетью, чтобы разгрузить коммуникационный поток основной сети, как это сделано с узлами 2 и 3 в этом примере:



Примечание. Может понадобиться "снизить" скорость передачи для драйвера Ethernet, использующуюся между узлами 2 и 3 на локальной сети 1. Делается это так:

На узле 2 и на узле 3 запустите драйвер для локальной сети 1 с **-r** *скорость_носителя*, установленной в 100000 (по умолчанию — 10000000):

```
Net.ether1000 -l1 -r 100000 &
Net.ether2100 -l2 &
```

Диагностика сети

Менеджер сети при выполнении поддерживает кольцевой буфер значимых сетевых событий. Можно выполнить утилиту netinfo, чтобы отобразить самый последний буфер. Вывод включает время события в сети, номер связанного с ним узла, числовой код события и краткое описание этого события.

Для получения дополнительной информации относительно этих утилит смотрите книгу ***Описание утилит***.

Глава 7. Настройка учетных записей пользователей

Эта глава охватывает следующие темы:

- Запуск сеанса пользователя
- Безопасность
- Файл регистрации
- Другие файлы регистрации

Запуск сеанса пользователя

Утилита login управляет доступом пользователя ко всем ресурсам QNX и системным функциям, когда пользователь делает попытку входа в систему. Пользователь с терминала /dev/ser1 может получить доступ в систему, введя правильные ответы для следующей команды:

```
on -t /dev/ser1 /bin/login
```

Вызываемая без опций, утилита login запрашивает у пользователя имя и пароль. Можно автоматизировать процесс входа в систему с помощью утилиты tinit. Чтобы tinit автоматически вызывала login при запуске системы, следует добавить следующую запись в файл sysinit.*узел* узла:

```
tinit -t /dev/ser1 &
```

Инициализация командного интерпретатора

После успешного входа пользователя в систему, на терминале запускается командный интерпретатор (shell). Запускаемый при входе в систему командный интерпретатор (например, /bin/sh) выполняет сначала файл /etc/profile. Посредством этого файла могут быть установлены локальные соглашения.

Затем выполняется специальный пользовательский файл установки начальных параметров, обычно это $HOME/.profile. В этот файл можно добавить любые нужные вам экспортируемые переменные окружения. Например, если вы хотели бы сохранять ваши настройки запускаемого при входе в систему командного интерпретатора всякий раз, когда процесс запускает от вашего имени новый командный интерпретатор, то нужно определить и экспортировать переменную окружения **ENV** в вашем файле .profile. Например:

```
export ENV=$HOME/.kshrc
```

Эта запись гарантирует, что каждый раз при запуске нового командного интерпретатора будет выполняться файл $HOME/.kshrc. Чтобы отобразить значения всех экспортируемых

переменных, напечатайте export -p.

Установка переменных окружения

Когда пользователь начинает сеанс, то начинает работу программа запускаемого при входе в систему командного интерпретатора (например, /bin/sh). Смотрите описание утилит login и sh в книге *Описание утилит*, если хотите получить дополнительную информацию о переменных командного интерпретатора и о функционировании по умолчанию.

Дополнительные установочные параметры берутся из пользовательского файла .profile, если он существует. Командный интерпретатор может автоматически добавить некоторые из требуемых значений в файл .profile пользователя. Например, файл .profile пользователя может содержать установочные параметры для следующих переменных окружения:

- **TERM –** определяет тип терминала пользователя (например, qnx или qansi - перечень допустимых значений смотрите в главе "**Установка терминалов**")
- **TMPDIR –** задает местоположение временного рабочего пространства, которое может быть использовано утилитами и программами (например, /tmp)
- **TZ –** устанавливает зону времени (например, TZ=EST5EDT4)

Команда set может устанавливать (+) или не устанавливать (-) эти опции командного интерпретатора. Чтобы отобразить список переменных, которые могут быть определены для вашего командного интерпретатора, напечатаете set после приглашения командного интерпретатора. Чтобы увидеть значение одной переменной, напечатаете echo $*имя_переменной* после приглашения командного интерпретатора (например, echo $WORKDIR). Для получения дополнительной информации об этих командах командного интерпретатора смотрите утилиту sh в книге *Описание утилит*.

Безопасность

QNX предоставляет механизмы по управлению доступом к ресурсам и критическим системным функциям. Эти механизмы базируются на способности системы определять конкретного пользователя.

Когда система загружается в первый раз (сразу после начальной установки), вы автоматически регистрируетесь в ней как суперпользователь (root). Это имя пользователя имеет доступ к любой части системы и не требует пароля.

Чтобы предотвратить несанкционированный доступ пользователей в систему, следует задать для root пароль. Вы можете создать для себя новую учетную запись пользователя, чтобы использовать ее впоследствии для ежедневной работы. Утилита passwd позволяет изменять пароль входа в систему; вы можете использовать утилиту, чтобы создавать новое имя пользователя для себя и для других пользователей.

Утилиты управления доступом

Утилиты управления доступом в QNX:

- `login` – вход в систему;
- `su` – временное получение прав другого пользователя;
- `passwd` – создание и ведение учетных записей и изменения паролей пользователей.

Утилита login

Утилита login может быть запущена с помощью tinit на всех устройствах tty. Утилита запрашивает имя пользователя и пароль и сверяет их с базой данных пользователей /etc/passwd перед предоставлением доступа к системе. Если комбинация имя пользователя/пароль неправильная, login выдает сообщение об этом и завершается. Если пользователь вводит правильную комбинацию, то login запускает командный интерпретатор и загружает окружение для данного пользователя.

Утилита su

Утилита su позволяет пользователю временно иметь привилегии другого пользователя. Если идентификатор (ID) пользователя не введен в командную строку при запуске su, утилита запрашивает пароль, который должен соответствовать паролю суперпользователя. Ввод правильной комбинации доступа заставляет su обратиться в базу данных пользователя /etc/passwd и запустить командный интерпретатор, который имеет все права и привилегии соответствующего пользователя. Выход из командного интерпретатора возвратит вас к своему настоящему ID пользователя.

Утилита passwd

Утилита passwd может использоваться для изменения пароля или ввода нового пользователя в систему. Любой пользователь может изменить свой собственный пароль, но только суперпользователь (root) имеет право создавать или изменять другие пароли. В обоих случаях утилита passwd блокирует доступ к файлу паролей пользователей /etc/passwd, чтобы предотвратить любые другие попытки доступа к файлу во время его модификации.

Если вы:	**Используйте команду:**
пользователь	passwd для изменения *вашего* пароля
root	passwd *имя пользователя* для создания или изменения пароля

Прежде, чем начать создание ID пользователя, вы имеете возможность изменить режим работы passwd. Параметры установки в файле /etc/default/passwd определяют, какой локальный алгоритм будет осуществлять утилита passwd, например, обязательность паролей,

диапазон изменений ID пользователя и т.п. Для получения дополнительной информации об опциях, которые вы можете задавать, смотрите утилиту passwd в книге *Описание утилит*.

Суперпользователь может добавить новую учетную запись, вызвав passwd с новым *именем пользователя*. Утилита запрашивает информацию, которая будет управлять окружением нового пользователя, включая права доступа. Учетная запись включает в себя ID группы, к которой принадлежит пользователь, и атрибуты доступа файла.

Например, чтобы создать новую учетную запись с именем пользователя jsmith, суперпользователь должен ввести:

```
passwd jsmith
```

По умолчанию, passwd будет затем запрашивать следующую информацию:

- ID пользователя;
- ID группы, к которой принадлежит пользователь;
- реальное имя пользователя;
- исходный (домашний) каталог пользователя;
- начальная команда (например, /bin/sh);
- начальный пароль.

Номер ID пользователя

Обычно учетные записи пользователя имеют номер ID пользователя >= 100. Номера ID пользователя ниже 100 часто используются процессами системы. Диапазон для ID пользователя устанавливается в файле /etc/default/passwd.

ID группы, к которой принадлежит пользователь

Если ID группы не существует, система выведет напоминание, что нужно добавить группу в файл /etc/group.

Файл /etc/group предоставляет привилегии всем пользователям, являющимся членами определенной группы. Например, отделу "Финансы" разрешен доступ к финансовым записям компании, а группе "Поддержка Заказчика" - нет. Вы можете установить ID группы "Финансы" с требуемыми правами доступа и затем назначать пользователей в этой группе согласно привилегиям, которых они требуют.

Файл /etc/group содержит список пользователей в группе. Каждая строка имеет следующий общий синтаксис:

имя группы::*ID_группы*:*пользователь1*, *пользователь2*, *пользователь3*, ..., *пользовательN*

Например, следующий ввод определяет группу "maestri" с ID группы "123" и членами "alvivaldi", "gfhandel", "gptelemann" и "jsbach":

```
maestri::123:alvivaldi, gfhandel, gptelemann, jsbach
```

Сначала вы будете добавлять группы, которые не имеют никаких пользователей, как в следующем примере:

```
docs::0:
techies::1:
support::2:
marketing::3:
finance::4:
```

Когда пользователь входит в систему, значение в базе данных паролей /etc/passwd определит ID группы, который был присвоен пользователю (если имеется). Пользователям, которые принадлежат к этой группе, позволено подключаться к привилегиям этой группы всякий раз, когда они используют команду newgrp.

Исходный каталог, начальная команда и начальный пароль

Исходный каталог, начальная команда и начальный пароль - значения необязательные. Если вы не определяете эти значения, то будут использоваться следующие значения по умолчанию:

- *исходный каталог* – используется каталог /home/*имя_пользователя* (создается, если он не существует);
- *начальная команда* – команда запуска оболочки по умолчанию (/bin/sh), выполняемая немедленно после входа пользователя в систему
- *пароль* – учетная запись пользователя создается без пароля

Удаление учетной записи пользователя

Чтобы удалить учетную запись, вы должны удалить *все*, что указано ниже:

- запись в базе данных пользователя из файла /etc/passwd;
- запись с именем пользователя из соответствующей ID группы принадлежности в файле /etc/group;
- соответствующую закодированную запись пароля входа в систему из файла /etc/shadow;
- (необязательно) исходный каталог пользователя;
- (необязательно) "почтовый ящик" пользователя (то есть /usr/spool/mail/*имя_пользователя*).

Подробнее о ID пользователей и групп

После регистрации в системе пользователю присваиваются ID пользователя и ID группы.

Они известны как *реальный* ID пользователя и *реальный* ID группы.

ID пользователя должен быть уникален - никакие два пользователя не должны совместно использовать одинаковые ID. Это правило соблюдается утилитой passwd, однако суперпользователь может обойти это правило, редактируя непосредственно файл пароля.

ID группы позволяет нескольким пользователям объединиться в группу. Этот механизм позволяет группе пользователей совместно использовать общие ресурсы, не делая их доступными для всех остальных пользователей. Для получения дополнительной информации о файле /etc/group и его содержимом смотрите утилиту newgrp в книге *Описание утилит*.

Эффективные идентификаторы пользователя и группы

Процессы имеют два класса ID пользователя и группы: *реальные* и *эффективные*. Когда процесс создается, то он автоматически наследует эти четыре ID от своего родителя (который является обычно запускаемым при входе в систему командным интерпретатором):

- реальный ID пользователя;
- реальный ID группы;
- эффективный ID пользователя;
- эффективный ID группы.

Эффективные ID пользователя и группы используются для проверки прав доступа. Процесс может изменить эффективные ID пользователя или группы, или оба ID так, чтобы они отличались от своих реальных аналогов. Обычно это делается для того, чтобы получить доступ к ресурсам, которые не доступны для реальных ID пользователя и группы.

Чтобы изменить эффективные ID группы и пользователя:	**Используйте:**
Из программ Си	Функции *seteuid()* и *setegid()*
Из командного интерпретатора	Утилитами su и newgrp
Для исполняемого модуля на диске	Утилиты chgrp и chmod (например, чтобы изменить группу, к которой принадлежит пользователь, и биты атрибутов доступа для программы mailx: chgrp mail /usr/bin/mailx chmod g+s /usr/bin/mailx)

Для поучения большего количества информации о битах setuid и setgid, смотри раздел "**Атрибуты доступа файлов**".

Утилита newgrp

Утилита newgrp запускает новый командный интерпретатор с различными реальными и эффективными ID группы. Если она вызывается без параметров, то newgrp подключает ID группы идентифицированного в базе данных пароля для текущего пользователя. База данных пароля управляет группами, к которым может подключиться конкретный пользователь.

Атрибуты доступа файлов

В QNX каждый файл имеет связанный набор атрибутов доступа, называемые *битами режима*. Эти биты режима, в сочетании с *владельцем* и *группой* файла, управляют доступом к файлу.

Когда процесс создает файл, эффективный ID пользователя и эффективный ID группы процесса определяют владельца и группу, которым будет принадлежать файл. Файлу также назначается набор битов режима (это описано в разделе "**Как устанавливаются биты режима**").

Имеются три класса битов режима:

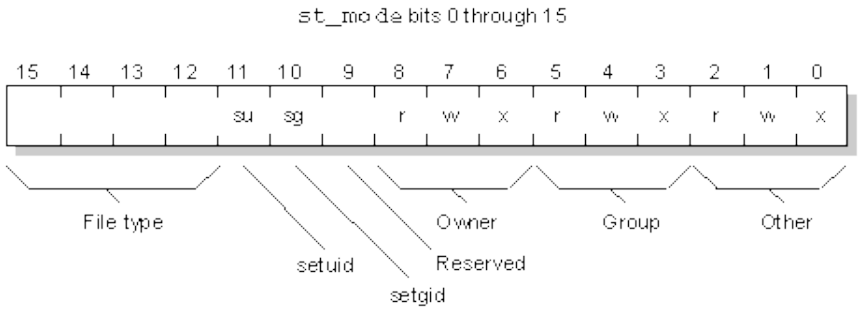
- владелец;
- группа;
- остальные.

Когда процесс пробует обратиться к файлу, Менеджер файловой системы обслуживает соответствующий класс битов режима, выполняя несколько сравнений в определенном порядке:

1. эффективный ID пользователя запрашивающего процесса сравнивается с владельцем файла. Если они совпадают, то далее рассматриваются биты режима *владелец*;
2. если сравнение с владельцем файла терпит неудачу, то эффективный ID группы запрашивающего процесса сравнивается с ID группы файла. Если ID этих групп соответствуют, то рассматриваются биты режима *группа*;
3. если и сравнение ID владельца, и сравнение ID группы терпят неудачу, то рассматриваются биты режима *остальные*.

Примечание. Как суперпользователь root (uid 0) вы можете обращаться к любому файлу для чтения/записи независимо от атрибутов доступа, и вы можете выполнить любой файл, для которого установлен хотя бы один бит режима разрешения выполнения.

Биты режима файла сохраняются в 16-битовом поле st_mode элемента каталога для файла; доступ предоставляется, если установлен бит режима.



Следующая таблица показывает различие воздействия атрибутов доступа на регулярные файлы и каталоги:

Бит атрибутов доступа:	**Воздействие на файл:**	**Воздействие на каталог:**
r	Чтение файла	Проверка имен файлов в каталоге (например, с помощью ls)
w	Запись в файл	Создание, удаление и переименование файлов внутри каталога, включая подкаталоги
x	Выполнение файла	Поиск каталога. Это означает, что вы можете изменить ваш текущий рабочий каталог на этот каталог. Кроме того, вы можете включить этот каталог в имя пути (например, в *open()*, *stat()* и т.п.).

Примечание. Бит атрибутов доступа - "разрешение выполнения" имеет смысл только для каталогов и регулярных файлов; он не применяется к другим типам файла.

Просмотр атрибутов доступа

Вы можете использовать команду ls -l, чтобы просмотреть атрибуты доступа файла, владельца и группу. Следующий вывод ls -l для файла alpha.c показывает, что владелец и любой пользователь из группы techies может читать и записывать в файл, в то время как остальные пользователи могут только читать из него:

```
-rw-rw-r--  1 mplanck   techies   8475 Apr 1  1997 alpha.c
```

В следующем примере вывод ls -ld для каталога /bin показывает, что владелец и члены группы могут читать, записывать и вести поиск в каталоге, в то время как остальные пользователи не имеют разрешения записи и, следовательно, не могут добавлять какие-либо новые файлы в каталог:

```
drwxrwxr-x  2 root     techies   2048 Sep 19  1990 bin
```

Но обратите внимание, что остальные пользователи все же могут просматривать или выполнять файлы в /bin.

Это возможно - хотя и необычно - иметь разрешение читать каталог, но не иметь разрешения поиска в нем. Например, ls может работать (в зависимости от опций, которые вы задаете ей), но вы не сможете обратиться к любому файлу в каталоге или выполнить команду cd в каталоге.

Как устанавливаются биты режима

Атрибуты доступа файла устанавливаются, как комбинация текущей маски *umask* создающего процесса и режима доступа, запрашиваемого функциями *open()*/*creat()*.

Маска битов атрибутов доступа - *umask* - указывает, какие биты атрибутов доступа будут "сброшены", если процесс определяет по умолчанию атрибуты доступа при создании файла, каталога или очереди (смотрите umask в книге *Описание утилит*)

Маска представляет собой логическое "И" режима и дополнения (обратного кода) *umask*. Например, если запрашивается режим с разрешением чтения/записи для всех:

Двоичный вид:	**Восьмеричный вид:**	**Символьный вид:**
110 110 110	0666	rw- rw- rw-

а *umask* указывает разрешение записи для группы и остальных:

Двоичный вид:	**Восьмеричный вид:**	**Символьный вид:**
000 010 010	0022	--- -w- -w-

то в результате владелец получит разрешение чтения/записи, а группа и остальные получат разрешение чтения:

Двоичный вид:	**Восьмеричный вид:**	**Символьный вид:**
110 100 100	0644	rw- r-- r--

Изменение атрибутов доступа

Вы можете изменить атрибуты доступа файла следующими образом:

Чтобы:	**Используйте:**
Сменить владельца и группу	Утилиты chown и chgrp или функцию *chown()* (в программах Си)
Изменить атрибуты доступа и биты режима setuid и setgid	Утилиту chmod или функцию *chmod()* (в программах Си)
Установить маску режима создания файла	Утилиту umask

Примечание. Если вы желаете сменить владельца или изменить атрибуты доступа к файлу, то ваш эффективный ID пользователя должен совпадать с ID владельца файла. Если вы сменяете владельца файла на отличного от вас, то вы больше не сможете изменять разрешения и право собственности на файл. Суперпользователь может модифицировать биты режима любого файла, независимо от его владельца.

Биты режима setuid и setgid

Чтобы выполнить определенные функции, пользователь должен иногда выполнить команду, на уровне привилегий более высоком, чем он имеет. Выполнять команду на более высоком уровне позволяют пользователю следующие два бита режима файла:

- setuid (установить ID пользователя) - например, необходимо, чтобы утилита passwd вела себя так, как будто она является root для того, чтобы можно было изменить файлы passwd и shadow в интересах пользователя, который обычно не имеет разрешения записи в эти файлы. (Биты режима для этой команды будут выглядеть следующим образом: -rwsrwxr-x);
- setgid (установить ID группы) - например, необходимо, чтобы команда mailx позволила пользователю временно "принадлежать" группе mail для того, чтобы разрешить пользователю записывать в файлы "почтовых ящиков" (mailboxes) других пользователей, которые обычно имеют разрешения: -rw-rw ---- *имя_пользователя* mail. (Биты режима для этой команды будут выглядеть следующим образом: -r-xr-sr-x).

Если эти биты установлены, то выполняемый файл будет выполняться с привилегиями его владельца и группы, а не с привилегиями процесса, который вызывает выполнение такого файла. Эти биты режима применяются только к выполняемым регулярным файлам.

Если файл загружается для выполнения и бит режима setuid установлен, то эффективным ID пользователя нового процесса становится таким, как у владельца файла. Аналогично, если бит режима setgid установлен, то эффективным ID группы процесса становится таким, как у группы файла.

Для того чтобы закрыть потенциальную лазейку в безопасности, биты setuid и setgid обнуляются всякий раз, когда изменяется владелец или группа файла. Установка бита setuid защищает от записи пользователями выполняемого файла на диск и, таким образом, изменяя владельца файла на root, пользователь получает неограниченные системные полномочия.

Утилиты passwd, login, su и newgrp - все имеют владельца root и бит setuid; следовательно, эти программы запускаются с правами доступа суперпользователя.

Принято, что root - единственный пользователь, имеющий ID пользователя равный 0, который обеспечивает статус суперпользователя. При управлении правами доступа, пользователь должен гарантировать, что можно доверять установку setuid как root только надежным и совершенно надежным программам. Никаких специальных привилегий при установке setgid как root программа не получает.

Примечание. Так как все программы при установке setuid как root наследуют возможности суперпользователя, вы должны удостовериться, что они *не* имеют всеобщих разрешений записи, чтобы только суперпользователь был способен модифицировать программы.

База данных пароля

Эти файлы формируют общую базу данных пароля:

```
/etc/passwd
/etc/group
/etc/shadow
/etc/.pwlock
```

Атрибуты доступа к этим файлам должны устанавливаться следующим образом:

Файл:	**Владелец:**	**Группа:**	**Атрибуты доступа:**
/etc/passwd	root	root	rw- r-- r--
/etc/group	root	root	rw- r-- r--
/etc/shadow	root	root	rw- --- ---
/etc/.pwlock	root	root	rw- r-- r--

Файл /etc/passwd

Файл /etc/passwd содержит набор строк следующего формата:

username:*haspw*:*userid*:*group*:*comment*:*homedir*:*shell*

где:

- *username* – имя пользователя для входа в систему;
- *haspw* – если пусто, то пользователь не имеет никакого пароля; если x, то пароль пользователя находится в файле /etc/shadow. Пустое значение или значение x - единственные разрешенные значения;
- *userid* – числовой ID пользователя;

- *group* – числовой ID группы;
- *comment* – поле комментария свободного формата; не должно содержать ":";
- *homedir* – исходный (домашний) каталог этого пользователя (по умолчанию - /);
- *shell* – начальная команда (и аргументы), запускаемая после login (по умолчанию - /bin/sh).

Вот пример строки из файла /etc/passwd:

```
bubba:x:290:120:Bubba L. Jones:/home/bubba:/bin/sh
```

Файл /etc/group

Файл /etc/group содержит строки в следующем формате:

```
groupname::group_ID:[user_ID[,user_ID]...]
```

где:

- *groupname* – имя группы;
- *group_ID* – числовой ID группы;
- *user_ID* – один или более ID пользователя, принадлежащих этой группе.

Вот пример строки из файла /etc/group:

```
techies::123:bubba,charlie,sue,jake
```

Файл /etc/shadow

Файл /etc/shadow содержит строки в следующем формате:

```
user_ID:password:0:0:0
```

где:

user_ID – имя пользователя;
password – зашифрованный пароль пользователя.

Вот пример строки из файла /etc/shadow:

```
bubba:SRjg51|weIJ[H:819388588:0:0
```

Файл /etc/.pwlock

Файл /etc/.pwlock создается утилитой passwd, чтобы указывать другим запускаемым

утилитам passwd, что файл пароля в данный момент времени изменяется. Когда утилита passwd завершается, файл блокировки удаляется.

Вы могли заметить, что файл /etc/passwd доступен для чтения любому пользователю. Это обеспечивает стандартными утилитами простой механизм поиска информации о пользователях. Поскольку этот файл читается, зашифрованный пароль здесь не хранится.

Зашифрованный пароль хранится в файле /etc/shadow, который читается только суперпользователем. Это должно предотвращать несанкционированные попытки дешифрования паролей. Чтобы обеспечить безопасность вашего сообщества пользователей, вы должны обеспечить поддержание этих разрешений.

QNX поставляется с базой данных паролей, которая по умолчанию включает /etc/passwd и /etc/group. Файл /etc/shadow не поставляется, потому что учетные записи пользователей первоначально не имеют паролей.

Примечание. Если вы захотели отредактировать файл /etc/passwd на работающей системе (что обычно не рекомендуется!), то следуйте этим предосторожностям:

1. Выполните touch для файла, чтобы изменить время доступа;
2. Выполните ls -l, чтобы проверить, что вы являетесь владельцем файла;
3. Произведите редактирование;
4. Удалите файл /etc/.pwlock.

Обратите внимание, что если кто-нибудь еще (или некоторая утилита, подобная login и su) пробует обратиться к базе данных пароля в то время, когда вы редактируете файл /etc/passwd, то это может закончиться разрушением базы данных пароля!

Также обратите внимание, что если вы удалите пароль пользователя, то вы должны также удалить соответствующую запись из файла /etc/shadow.

Файлы пароля по умолчанию

По умолчанию файл /etc/passwd, поставленный с вашей системой QNX, содержит следующее:

```
root::0:0::/:/bin/sh
```

Файл /etc/group по умолчанию содержит следующее:

```
root::0:root
```

Файл регистрации

Утилита login обновляет системную учетную информацию, которая регистрируется в файле

/etc/acclog. Если этот файл не существует, то вся учетная информация будет игнорироваться - это нормальный режим работы после установки QNX.

Для большинства систем, работающих в реальном масштабе времени, рекомендуется этот, устанавливаемый по умолчанию режим работы, когда учетная информация не записывается. Если компьютер имеет линию телефонного входа, или если вы запускаете QNX в сети с большим количеством пользователей, то вы можете включить регистрацию, создав пустой файл /etc/acclog.

Включение регистрации

Чтобы включить регистрацию, вы должны создать пустой файл /etc/acclog. Его можно создать, используя утилиту touch:

```
touch /etc/acclog
chmod g=,o= /etc/acclog
```

Если этот файл создан, то учетная информацию будет регистрироваться в нем.

Обратите внимание, что только суперпользователь (ID пользователя root) может создавать и изменять этот файл.

Формат записи

Каждая запись в файле /etc/acclog имеет форму:

```
ttttttttt cc data...
```

где *ttttttttt* - время в секундах, начиная с 1970 (в десятичном исчислении). За ним всегда следует один пробел. За временем следует двухсимвольный код *cc*. За кодом через пробел следуют данные, специфические для каждого кода. Каждая строка завершается символом перевода строки. Следующие утилиты производят запись в файл /etc/acclog:

Утилиты:	**Назначение:**	**Запись:**
login	Вход пользователя в систему	*ttttttttt* LO *устройство uid gid uname*
login	Неудачный вход	*ttttttttt* LF *устройство uname*
modem	Модемная связь	*ttttttttt* MO *устройство baud*
su	Переключение пользователя	*ttttttttt* SU *устройство uid gid uname*
tinit	Запуск команды	*ttttttttt* TS *устройство command*

Утилиты:	**Назначение:**	**Запись:**
tinit	Управление с устройства	*ttttttttt* TA *устройство*

где: *uname* - имя пользователя.

Типичный учетный файл может выглядеть следующим образом:

```
670464500 TS //1/dev/ser1 modem -b 19200 -L
670464545 MO //1/dev/ser1 2400
670464550 LO //1/dev/ser1 100 101 steve
670465824 TS //1/dev/ser1 modem -b 19200 -L
```

Эта записи показывают, что tinit, запущенная с программой modem, ждет вызова. Получен вызов и выдан ответ на скорости в 2400 бод, и получатель с ID пользователя steve вошел в систему. Обратите внимание, что в файле регистрации не показывается выход из системы. Можно только косвенно определить выход из системы, так как после выхода из системы tinit снова запускается с программой modem.

Общая продолжительность сеанса пользователя (при успешном входе в систему) может быть вычислена следующим образом:

```
670465824 - 670464550 = 1274 секунды
```

В сильно загруженной системе записи от большого количества устройств будут постоянно поступать в учетный файл. Для того чтобы соотнести зафиксированные события с каждым устройством, вы можете воспользоваться номером узла, который указан для каждого устройства и позволяет отследить регистрационные записи для всех устройств сети в едином регистрационном файле.

Имеются несколько типичных последовательностей событий:

Последовательность событий:	**Значение:**
TS --> LO --> TS	Начало и конец сеанса по выделенной линии.
TS --> MO --> LO --> TS	Начало и конец сеанса по коммутируемой телефонной линии.
TA --> TS --> LO --> TA	Начало и конец сеанса на выделенной линии, ожидающей нажатия клавиши.
TS --> TS	Неудачная попытка входа на выделенную линию.
TS --> MO --> TS	Неудачный вход по коммутируемой телефонной линии.

Очистка регистрационного файла

Как только вы создаете файл /etc/acclog, он начинает увеличиваться, так как в него будут постоянно добавляться записи. Если этим процессом не управлять, то данный файл может занять значительное дисковое пространство, поэтому вам следует регулярно распечатывать или архивировать информацию из этого файла. Вы можете даже решить автоматизировать эту процедуру, используя утилиту cron (смотри книгу *Описание утилит*).

В следующем примере регистрационный файл переименовывается в файл, название которого содержит год и месяц, и создается новый пустой регистрационный файл:

```
mv /etc/acclog /etc/acclogs/9607
touch /etc/acclog
chmod g=,o= /etc/acclog
```

Сжатие файла регистрации

Так как данные в этом файле очень повторяющиеся, вы можете использовать утилиту сжатия gzip, которая обеспечивает очень высокий коэффициент сжатия файла. Это может значительно уменьшить требования к дисковому пространству, если вы сохраняете текущие файлы регистрации в работающей системе или записываете их на дискету.

Примечание. Не забывайте переименовывать файл *перед* его сжатием. Никогда не сжимайте непосредственно файл /etc/acclog.

Вот пример рекомендуемой процедуры сжатия:

```
mv /etc/acclog /etc/logs/9607
touch /etc/acclog
chmod g=,o= /etc/acclog
gzip /etc/logs/9607
```

Имейте в виду, что и другие утилиты (возможно третьих фирм) могут добавлять свои собственные регистрационные записи в файл /etc/acclog.

Другие файлы регистрации

Для того чтобы QNX начала записывать события в регистрационный файл, вы просто создайте пустой регистрационный файл в соответствующем каталоге. Вы должны использовать имена файлов, приведенные ниже. QNX начнет регистрировать системные события, как только соответствующий регистрационный файл будет создан.

Файл /tmp/syslog

Утилиты QNX регистрируют проблемы или непредвиденные события в файле /tmp/syslog.

Файл /usr/adm/syslog

Утилита logger добавляет строки в файл /usr/adm/syslog. Пока программа имеет доступ записи в этот файл, она может выполнять утилиту logger, как процесс setuid, принадлежащий adm:adm.

Файл /usr/adm/lastlog

Утилита login записывает информацию в файл /usr/adm/lastlog (при условии, что он существует). Файл может использоваться, чтобы проследить, где и когда пользователь в последний раз вошел в систему. Утилита finger может вывести на экран информацию о пользователе из файла lastlog.

Файл $HOME/.lastlogin

Файл $HOME/.lastlogin создается утилитой login. Он содержит информацию о том, когда и с какого устройства последний раз пользователь входил в систему. Содержимое этого файла отображается, когда пользователь регистрируется в системе. Если пользователь не входил в систему с тех пор, как было последнее обновление "сообщения дня" (motd), то на терминал пользователя выводится содержимое файла /etc/motd (при условии, что он существует).

Файл /etc/nologin

Присутствие файла /etc/nologin запрещает кому-либо регистрацию в системе.

Файл /etc/wtmp

Файл /etc/wtmp поддерживает список тех, кто в настоящее время находится в системе, и то, как эти пользователи вошли в систему. Эту информацию использует TCP/IP для QNX.

Файл /etc/utmp

Файл /etc/utmp обновляется утилитой login. Он содержит информацию о входе в систему в двоичном формате и поддерживает хронологию всех изменений в /etc/wtmp.

Глава 8. Установка терминалов

Эта глава охватывает следующие темы:

- Характеристики терминалов
- Установка типа терминала
- База данных termcap
- База данных terminfo
- Расширенные характеристики для событий мыши

Характеристики терминалов

Два вида файлов с характеристиками терминалов дают программистам свободу для написания терминально-независимых программ. Более старые полноэкранные программы полагались на коды управления в файле с характеристиками терминала /etc/termcap, чтобы использовать возможности терминала. Более новые программы и утилиты запрашивают базу данных terminfo (в каталоге /usr/lib/terminfo).

Установка типа терминала

Переменная окружения **TERM** определяет тип терминала. Сначала программы запрашивают переменную окружения **TERM**, чтобы определить, с каким типом терминала они выполняются. После определения типа терминала программа находит характеристики терминала либо в файле termcap, либо в соответствующем файле terminfo.

Установка типа терминала обычно наследуется из окружения от sinit немедленно после начальной загрузки:

```
sinit -r //$(bnode)/ TERM=qnx
```

Как альтернативный вариант, тип терминала может определяться в пользовательском файле .profile.

Для стационарных терминалов, которые выполняют заранее известные прикладные программы, вы можете предварительно установить переменную окружения **TERM** при запуске приложения:

```
TERM=vt100 on -t /dev/ser1 заказная_прикладная_программа
```

Но вы не можете сделать это при входе в систему пользователей с помощью login и tinit. Если тип терминала известен, то можно заставить tinit устанавливать переменную окружения **TERM** перед запуском login:

```
tinit -c login -t /dev/ser1 TERM=ansi &
```

Замечание о доступе по коммутируемой линии

Если вы не ограничиваете доступ по коммутируемой линии одним конкретным типом терминала, то для входящих через модем пользователей нужно будет выполнять утилиту termdef, чтобы запросить тип терминала. Утилита termdef может отображать все поддерживаемые типы терминалов и запросить пользователя выбирать один из них:

```
tinit -c "modem -c termdef" -t /dev/ser1 &
```

Если тип терминала выбран, то termdef устанавливает, соответственно, переменную окружения **TERM** и запускает login для пользователя.

База данных termcap

Коды управления в файле termcap QNX могут определять входные и выходные характеристики ряда типов терминалов, в том числе:

- универсальный терминал стандарта ANSI (ansi);
- QNX консоль (qnx);
- QNX ANSI (qansi, qansi-g, qansi-t, qansi-m или qansi-w);
- QNX терминал (qnxt);
- QNX терминал с событиями "мыши" (qnxm);
- QNX Windows (qnxw);
- QNX монохромный терминал или консоль(qnxtmono);
- QNX 2.15 последовательный терминал (qnxt2 или qnxs2);
- VT100 с Auto-margin (vt100);
- VT102 (vt102 или vt102-plus);
- Эмулятор терминала X-term (xterm, xterms, xterm-m или xterm-q).

Полный список типов терминала смотри в /etc/termcap.

Каждый элемент в файле termcap начинается со строки псевдонимов терминала. Остальные строки с кодами управления представляет собой список терминальных характеристик.

Каждая строка характеристики может идентифицироваться именем из двух символов. В общем, строка описывает поведение терминала, когда пользователь нажимает клавишу, или визуальный эффект по команде программы. Если имеется более чем одна запись для одного и того же типа терминала, то используется первая запись.

База данных terminfo

Подкаталоги terminfo - /usr/lib/terminfo/... - содержат двоичные файлы. Каждый файл в базе данных terminfo содержит описание характеристик, связанное с одним типом терминала. QNX поставляется с несколькими такими файлами, включая:

`/usr/lib/terminfo/q/qnx` (QNX консоль)

`/usr/lib/terminfo/v/vt100` (VT100 терминал)

`/usr/lib/terminfo/a/ansi` (ANSI терминал)

Соответствующие исходные тексты для всех QNX-поддерживаемых типов терминала содержатся в /usr/lib/terminfo/terminfo.src.

Исходные тексты файлов termcap и terminfo схожи в том, что они оба содержат описания характеристик терминала. Однако, имена характеристик в файле terminfo могут иметь длину от двух до пяти символов. И так как файлы terminfo компилируются, то они загружаются и выполняются быстрее.

Если вам нужно преобразовать из terminfo в termcap (в случае, если имеются какие-то старые программы), утилита infocmp имеет опцию **-C**, которая генерирует запись termcap из существующего файла terminfo. Если имеется более чем одна запись для одного и того же терминала в файле terminfo, то используется последняя запись.

Если вы не имеете файла terminfo для вашего типа терминала, то вы можете либо считать файл terminfo из QUICS, либо найти соответствующий файл terminfo в любой системе UNIX, либо создать новый файл terminfo с помощью утилит infocmp и tic.

Создание файла terminfo

Чтобы просматривать или изменять информацию в существующем файле terminfo, вы должны сначала декомпилировать файл с помощью утилиты infocmp. Чтобы использовать измененный исходный файл, вы должны скомпилировать его в двоичную форму, используя утилиту tic, компилятор для terminfo. Прикладные программы могут извлекать характеристики, в которых они нуждаются для данного терминала, прямо из компилированной версии.

Примечание. Вы можете получить характеристики из библиотеки curses более высокого уровня.

Расширенные характеристики для событий "мыши"

Файлы termcap и terminfo QNX поддерживают множество стандартных характеристик терминала. В дополнение к ним, QNX поддерживает следующие расширения строковых и

числовых характеристик для обработки событий "мыши".

Файл termcap:	**Файл terminfo:**	**Характеристики:**
ZZ	*mcud1*	micro_down. Не возвращать нажатие.
Za	*mcub1*	micro_left. Возвращать перемещение "мыши".
Zb	*mcuf1*	micro_right. Не возвращать перемещение "мыши".
Zd	*mcuu1*	micro_up. Не возвращать отпускание.
Zf	*mcud*	parm_down_micro. Возвращать нажатие ADJUST.
Zg	*mcub*	parm_left_micro. Возвращать нажатие SELECT.
Zh	*mcuf*	parm_right_micro. Возвращать нажатие MENU.
Zi	*mcuu*	parm_up_micro. Извещать об отпускании кнопки.
ZJ	*smicm*	enter_micro_mode. Извещать о изменении размера экрана.
ZT	*rmicm*	exit_micro_mode. Не извещать о изменении размера экрана.
Yd	*maddr*	max_micro_address. Максимальное значение в адресе micro_..._ (числовая характеристика).

Дополнительная информация

Полный список имен характеристик и их значений смотри в файле технических замечаний для terminfo в QUICS. Для получения более подробной информации о termcap и terminfo рекомендуется книга ***termcap & terminfo***, Strang, Mui, & O'Reilly (ISBN 0-937175-22-6).

Глава 9. Печать со спулингом

Эта глава охватывает следующие темы:

- Введение
- Использование утилит спулера
- Архитектура спулера
- Файл установки спулера
- Использование файлов установки
- Примеры файлов установки
- Организация доступа к спулерам и очередям

Введение

Совместное использование ресурсов в сети

QNX поддерживает сетевое распределение ресурсов с незначительными искусственными ограничениями. Каждое устройство (диск, модем, принтер и т.п.), подключенное к компьютеру, по умолчанию является *совместно-используемым* ресурсом. Таким образом, программа, работающая на любом компьютере, имеет равный доступ к любому устройству в сети, независимо от того, подключено устройство к тому же самому компьютеру (локальное) или к другой машине (удаленное).

Хотя пользователь, в принципе, может получить доступ к любым ресурсам в сети, системный администратор, вероятно, должен принять меры, чтобы контролировать доступ к определенным типам ресурсов. Принтеры и модемы, например, могут использоваться в один и тот же момент времени только одним пользователем и, следовательно, требуется организовать формирование очереди, чтобы избежать конфликтов. Для обеспечения удобного доступа к принтерам в системе QNX имеется набор сервисных функций *спулинга*.

Спулеры

Спулер это просто механизм, который принимает запросы на ресурс и затем распределяет использование ресурса согласно набору заданных правил.

Для того чтобы понять, каким образом использовать спулер, давайте рассмотрим, как он управляет доступом к принтеру. Принтер должен быть постоянно доступен пользователям, хотя он может выполнять одновременно только одно задание. Если имеется спулер, то пользователи могут послать данные через спулер, а не непосредственно на принтер. Получая данные, предназначенные для принтера, спулер записывает эти данные во временный файл, а не передает их немедленно на принтер. Позже, когда принтер становится доступным, спулер передает данные на принтер. Таким образом, несколько пользователей могут свободно

подавать задания на печать для одного физического принтера.

QNX осуществляет спулинг с помощью именованных очередей, на которые можно ссылаться набором утилит "lp"; эти очереди находятся в файловом пространстве имен в каталоге /dev/spool. Данные, записываемые в очередь, будут помещаться во внутренний список и, в конечном счете, посылаться на определенное устройство вывода.

Сервер спулинга QNX (lpsrvr) может поддерживать большое количество различных очередей спулера. С очередями спулера взаимодействуют следующие утилиты:

- `lp` – помещает файлы в очередь спулера
- `lprm` – удаляет задание из очереди спулера
- `lpc` – управляет очередью спулера
- `lpq` – отображает состояние очереди спулера

Подробную информацию об этих утилитах можно получить в книге *Описание утилит*. Для получения информации об установке заданий в очередь печати на принтер в сети TCP/IP, смотри главу "Удаленная печать" в книге *Руководство пользователя* по TCP/IP для QNX.

Использование утилит спулера

Запуск спулера

Прежде, чем в системе QNX выполнится печать со спулингом, вы должны запустить его сервер lpsrvr:

```
lpsrvr &
```

Для того чтобы определить, какие ресурсы доступны, и как управлять ими, утилита lpsrvr сначала ищет файл установки с именем /etc/config/lpsrvr.*узел* (где *узел* - идентификатор узла, на котором выполняется lpsrvr). Если файл установки с расширением *.узел* не обнаружен, то lpsrvr будет использовать файл /etc/config/lpsrvr.

Передача заданий спулеру

Следующая команда lp поместит файл report в принимаемую по умолчанию очередь спулера и, в итоге, распечатает его:

```
lp report
```

Подробную информацию об очереди по умолчанию, смотри в разделе "**Файл установки спулера**" этой главы.

В системах, где доступна более чем одна очередь спулера, вы можете определить имя

очереди. Следующая команда вставляет report в очередь спулера с именем txt:

```
lp -P txt report
```

Можно также использовать команду, которая записывает непосредственно в файл очереди:

```
cp report /dev/spool/txt
```

Запрос заданий очереди

Для того чтобы проверить очередь буферизации, можно использовать утилиту lpq. Ниже приведен пример вывода информации утилитой lpq:

```
1:  fred   [job #39]   1400 bytes  lalist.doc

2:  wilma  [job #42]   2312 bytes  netdrvr.c
```

Эта утилита позволит вам определить, когда были завершены поданные на выполнение задания; она также позволит определить ID задания спулера для использования его другими утилитами lp.

Отмена заданий спулера

Утилита lprm позволяет удалять задания из очереди спулера. Вы можете удалить задание явно, указав его номер ID. Для очереди, показанной выше, задание fred (#39) может быть отменено следующей командой:

```
lprm 39
```

Если задание #39 в данный момент времени обрабатывается, то оно будет отменено. Успешность отмены текущих заданий спулера может зависеть от типа используемого устройства вывода - некоторые принтеры имеют большие внутренние буфера.

Суперпользователь может также удалить все задания, принадлежащие конкретному пользователю. Например, все задания fred могут быть отменены следующей командой:

```
lprm fred
```

Управление очередями спулера

Утилита lpc - системное административное средство для управления спулерами. Она позволяет выполнять несколько управляющих функций, такие как, открытие или закрытие очереди. Обеспечиваются следующие базовые функции:

- приостановить/возобновить включение заданий в очередь;
- приостановить/возобновить выборку заданий из очереди;
- приостановить/возобновить выполнение текущего задания;
- удалить текущее задание;
- переставить задания в очереди;
- переместить задания в другую очередь;
- отобразить состояние очередей.

Имейте в виду, что функциональные возможности lpc перекрывают возможности lpq и lprm. Это удобно, т.к. в отличие от lpq и lprm, lpc может использоваться в диалоговом режиме.

Архитектура спулера

В основе спулинга QNX лежат два объекта: *очереди* и *адресаты* (targets). Они работают друг с другом, обеспечивая гибкий метод управления преобразованием данных и организацией очередей.

Очередь - это внутренний список ждущих данных, которые нужно послать адресату. Как отмечалось ранее, каждая очередь получает имя, которое пользователи указывают при запуске заданий.

Адресат связан с физическим устройством вывода (например, принтером) и удаляет задания из очередей (после их отработки). Можно соединить выход очереди с одним или более адресатами или соединить несколько очередей с единственным адресатом. Однако, можно соединить выход адресата только с одним устройством.

Очереди могут иметь дополнительные атрибуты, называемые *фильтрами*, которые имеют два типа:

- входные фильтры, copy-in, (ci), которые выполняют операции с данными прежде, чем они скопируются в очередь (например, форматирование);
- выходные фильтры, copy-out, (co), которые оперируют с данными после их удаления из очереди.

Например:

```
ci=a2ps -H"$(file)" | awk '/%%EndProlog/ { print "<< /Duplex \
    true >> setpagedevice"; } { print $0; }'
co=echo $(username)"\n" "put" $(spfile) | SOCK=666 /usr/ucb/ftp net_printer
```

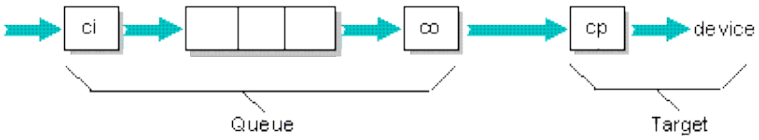
Адресаты могут иметь дополнительную *управляющую программу* (cp), которая позволяет, если потребуется, инициализировать устройство вывода между заданиями. Например, если выполнение задания отменено, и устройство вновь требуется для выполнения другого задания, то программа, управляющая устройством, должна обнаружить SIGTERM и определить, какое действие необходимо выполнить, чтобы перевести устройство в

устойчивое состояние.

Следующие схемы иллюстрируют, каким образом могут быть скомпонованы и работать друг с другом очереди, фильтры и адресаты.

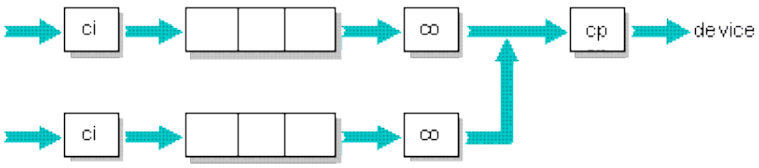
Одна очередь соединена с одним адресатом

Следующая конфигурация может использоваться там, где есть один принтер и требуется (или не требуется) преобразование данных:



Несколько очередей соединены с одним адресатом

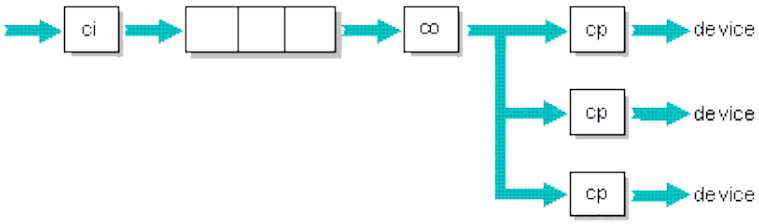
Если несколько очередей соединены с одним адресатом, то в этом случае адресат выберет из всех заданий в очередях подходящее задание, основываясь на приоритете очереди и времени ожидания (то есть самое задержанное задание).



В конце этой главы вы найдете несколько примеров файлов установки. Вышеуказанная конфигурация используется в примере, в котором одна очередь преобразует ASCII в PostScript, а другая - просто очередь PostScript. Обе очереди связаны с одним и тем же адресатом, который посылает данные на принтер PostScript.

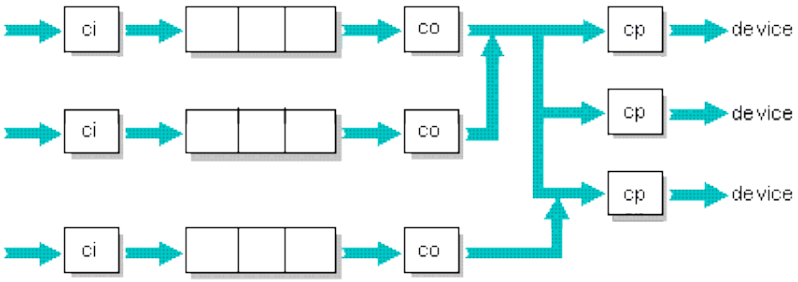
Одна очередь связана с несколькими адресатами

Если выход очереди передается на несколько адресатов, то спулер выберет любого доступного адресата. Следующая конфигурация используется в том случае, если имеется три принтера и не имеет значения, какой из них печатает ваше задание.



Несколько очередей связаны с несколькими адресатами

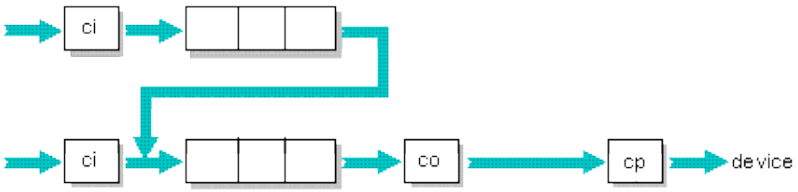
Следующая конфигурация - комбинация двух предыдущих примеров. Третья очередь имеет отдельный канал для одного из адресатов. Этот канал может быть использован, чтобы гарантировать передачу заданий всегда на третий принтер (например, принтер имеет цветные возможности).



Формирование цепочки очередей

Выход очереди обычно соединяется с одним или более адресатом, однако существует возможность соединения в *цепочку* ее выхода с другой очередью.

При формировании цепочки выход одной очереди подключается непосредственно к входу другой очереди, в обход любых возможных входных фильтров для этой очереди. Если последняя очередь в цепи имеет выходной фильтр, то он будет использоваться.



Файл установки спулера

После запуска спулер обращается к файлу, чтобы получить информацию о конфигурации. Если никакой файл не задан в командной строке, то спулер использует файл /etc/config/lpsrvr. Этот файл определяет очереди, адресатов и взаимосвязи между ними.

Синтаксис определений

Очереди и адресаты имеют символические имена, а также набор атрибутов. Каждый элемент файла установки имеет следующий формат:

```
[имя]
      атрибут
      атрибут
      .
      .
      .
```

Определение очереди или адресата начинается с директивы [*имя*] и состоит из всех допустимых значений атрибутов, следующих до следующей директивы [*имя*]. Все начальные пробелы игнорируются. Строки комментариев начинаются со знака фунта (#).

Чтобы продолжить один атрибут на другой строке, перед символом перехода на новую строку должен быть поставлен символ обратный слеш (\).

Имя может иметь до 48 символов и может содержать только алфавитно-цифровые символы.

Если описываемый объект - адресат, то перед именем должно быть тире (-). Тире не считается частью имени.

Каждый атрибут состоит из двухсимвольного ключа (key) в одной из следующих форм:

- *key* (логический);
- *key#число* (числовой);
- *key=string* (символьный).

Все *числа* считаются десятичными, если они не начинаются с нуля (означающего восьмеричное представление) или с комбинации 0x (означающей шестнадцатеричное представление).

Все *строки* содержат печатные символы. Обратная косая черта (\) - "специальный" символ. Он может использоваться для интерпретации других символов. Например, "настоящая" обратная косая черта должна быть представлена как: \\.

Следующая таблица описывает все определенные ключи, включая значения по умолчанию для каждого ключа, если его соответствующий атрибут не задан. В колонке Использование имеются следующие обозначения: "Q" - "очередь," "T" - "адресат" и "G" - "глобальный"

Ключ:	**Описание:**	**Умолчание:**	**Исполь-зование:**
ab=*string*	Выполняется, когда адресат по любой причине отменяет команду	Пустой указатель; никакая команда не выполняется	T
af=*string*	Файл регистрации, который могут использовать команды	Регистрация не выполняется (выполняемую lpsrvr	Q

Ключ:	**Описание:**	**Умолчание:**	**Исполь-зование:**
ab=*string*	Выполняется, когда адресат по любой причине отменяет команду	Пустой указатель; никакая команда не выполняется	T
		регистрацию не затрагивает)	
cd=*string*	Каталог, где находятся временные файлы спулера	/usr/spool/lp	G
ci=*string*	Команда copy-in; используется перед размещением задания в очереди	Двоичное копирование задания; без преобразования	Q
co=*string*	Команда copy-out; используется, когда задание удаляется из очереди	Двоичное копирование задания; без преобразования	Q
cp=*string*	Программа управления устройством; используется адресатом	Двоичное копирование задания; без преобразования	T
dv=*string*	Устройство, которое использует адресат	Направляется на стандартный вывод lpsrvr	Q, T
mn#*numeric*	Минимальное число заданий в очереди до начала вывода	0; задания выводятся из очереди как можно быстро	Q
mx#*numeric*	Максимальное число заданий в очереди	нет ограничения; ограничения для очереди определяются размерами памяти и диска	Q
na=*string*	Имя; строка, которая описывает очередь или адресата	Пустая строка	Q, T
ok=*string*	Команда, выполняемая при нормальном завершении адресата	Пустой указатель; не выполняются никакие действия	T
pr#*numeric*	Приоритет выполнения данной очереди (1-100); 100 - самый высокий	50	Q
qn=*string*	Очередь (в цепочке), в которую помещается задание после извлечения из спулера	Пустой указатель; задание из спулера удаляется	Q
re#*numeric*	Число повторов в случае неудачного выполнения	0	Q

Ключ:	Описание:	Умолчание:	Исполь-зование:
ab=*string*	Выполняется, когда адресат по любой причине отменяет команду	Пустой указатель; никакая команда не выполняется	T
	задания		
rt#*numeric*	Время повтора; число секунд	Не ограничено	Q
sp=*string*	Регистрируемое имя спулера, поддерживающего эту очередь	/qnx/spooler; регистрируется только, если не задано имя очереди	G
ta=*string*	Адресат, связанный с этой очередью	Передается на стандартный вывод lpsrvr	Q
wa#*numeric*	Ожидает заданное число секунд перед извлечением из очереди каждого задания	0; задания будут извлекаться без задержки, как можно быстро	Q

Так как ключи чувствительны к регистру, все ключи, состоящие из двух букв нижнего регистра, являются зарезервированными. Вы можете безопасно применять заказные расширения, используя верхний регистр или комбинацию верхнего и нижнего регистра. Утилиты спулера игнорируют любые опции, которые они не понимают.

Глобальные ключевые слова

Два ключевых слова, sp и cd, могут определять глобальную информацию для спулера. Для того чтобы указать, что они применяются глобально, перед этими ключевыми словами должна стоять пара пустых квадратных скобок ([ ]).

Регистрация имен - sp

Так как спулер всегда настроен на пространство имен файла /dev/spool, то на каждом узле может работать только один спулер. Вы можете запустить несколько спулеров в сети, если каждый спулер будет регистрироваться под другим глобальным именем. В противном случае, каждый спулер будет пытаться регистрироваться под принятым по умолчанию одним и тем же глобальным именем: /qnx/spooler. При наличии нескольких спулеров можно использовать такие имена, как, например, /qnx/spooler2, /qnx/spooler3 и так далее.

Для того чтобы указать глобальное имя, которое нужно регистрировать для спулера, необходимо в файле установки использовать команду sp. Имя всегда должно начинаться с наклонной черты вправо (/). Если ключевое слово sp не определено, то по умолчанию регистрируется глобальное (т.е. для всей сети) имя /qnx/spooler.

Определение временного каталога для файлов спулера - cd

Ключевое слово cd позволяет определить каталог, используемый для создания временных файлов спулера. По умолчанию, временные файлы создаются в каталоге /usr/spool/lp. Эти файлы удаляются, когда они больше не требуются.

Любой путь, определенный в ключевом слове cd, должен начинаться с ведущей наклонной черты вправо. Имейте в виду, что задаваемый каталог должен существовать и иметь соответствующие права доступа.

В следующем примере в файле установки регистрируется имя спулера /qnx/spooler2, он также помещает временные файлы в каталог /tmp/spool2:

```
# Глобальные переменные спулера

[ ]
    sp=/qnx/spooler2
    cd=/tmp/spool2

# Текстовая очередь
[txt]....
```

Переменные

Спулер будет устанавливать надлежащим образом следующие переменные, когда он встретит их:

Переменная:	**Описание:**
$(file)	Имя подаваемого на вход файла или "--standard input--", если имя не определено
$(fname)	Полное имя пути подаваемого на вход файла или "--standard input--", если имя не определено
$(spfile)	Имя файла данных спулера
$(username)	Имя, под которым вошел в систему пользователь, запустивший задание
$(userid)	Числовой ID пользователя, запустившего задание
$(queue)	Имя очереди, в которой находится текущее задание
$(target)	Имя адресата, в который помещается текущее задание
$(device)	Имя устройства, для которого запланировано выполнение задания
$(ncopies)	Число копий, заказанное пользователем
$(jobid)	Числовой ID этого задания
$(cifile)	Имя временного файла, которое используется программой copy-in

Кроме того, на все определенные выше ключи можно ссылаться, как на переменные. Например, *$(ci)* может раскрываться в имя команды входного фильтра: ci=*строка*.

Действия по умолчанию

Команда cat используется по умолчанию в качестве фильтрующих команд входного и выходного фильтров. К тому же, если не указано иначе в файле установки, то стандартный ввод и стандартный вывод команд фильтра автоматически подключаются к файлам или процессам по умолчанию. Значение по умолчанию для входного фильтра следующее:

```
cat < $(fname) > $(spfile)
```

Существуют два возможных значения по умолчанию для выходного фильтра, в зависимости от того, определена ли *управляющая программа* (т.е. cp):

```
cat < $(spfile) > $(device)
cat < $(spfile) | $(cp) > $(device)
```

Например, файл установки вида:

```
[txt]
    ta=lpt
    ci=pr -f -h
    co=txt2ps
    cp=init_printer

[-lpt]
    dv=/dev/par
```

в результате дает следующие подстановки:

```
pr -f -h < $(fname) > $(spfile)
txt2ps < $(spfile) | init_printer > /dev/par
```

Использование файлов установки

Очереди и адресаты

Когда создается файл установки, нужно определить каждую очередь, давая ей, имя и дополнительный список параметров; определяйте каждый адресат, начиная его имя с тире (-). Для того, чтобы проиллюстрировать это, рассмотрим очень простой файл установки:

```
[txt]
    ta=lpt
```

```
[-lpt]
    dv=/dev/par
```

В этом файле имеется очередь с именем txt и адресат с именем lpt. Когда данные посылаются в очередь спулера txt, то они сохраняются во временном файле спулера и называются "заданием." Когда спулер удаляет задание из очереди, то задание передается адресату lpt, который затем направляет данные в параллельный порт /dev/par.

Фильтры

Очень часто необходимо фильтровать данные спулинга, для чего lpsrvr имеет механизмы *входного* и *выходного* фильтрования. Как уже упоминалось, входное фильтрование выполняется перед помещением данных в очередь, а выходное - после того, как данные извлекаются из очереди.

Имейте в виду, что, если несколько очередей передают данные одному адресату и одна из очередей имеет выходной фильтр, который требует много времени для выполнения, то, если такая очередь будет выбрана, адресат будет временно недоступен для остальных очередей. (Для получения дополнительной информации смотрите раздел "**Примеры файлов установки**").

Примечание. Хорошей практикой считается заключать переменные в двойные кавычки, чтобы гарантировать что они "безопасны" для команды.

Использование входного фильтра

Ниже приведен пример использования входного фильтра при передаче данных с помощью pr для разбиения на страницы:

```
[txt]
    ci=pr -f -h "$(file)"
    ta=lpt

[-lpt]
    dv=/dev/par
```

Использование выходного фильтра

Можно использовать выходной фильтр, например, для одновременного вывода на два принтера PostScript и HP. Два выходных фильтра могут сгенерировать соответствующие выходные форматы. Допустим, имеется две программы: txt2ps, которая генерирует коды для PostScript и txt2hpgl, которая генерирует коды для HPGL:

```
[ps]
    ta=lpt1
```

```
    co=txt2ps

[hp]
    ta=lpt2
    co=txt2hpgl

[-lpt1]
    dv=/dev/ser

[-lpt2]
    dv=/dev/par
```

Примечание. Не кодируйте жестко устройство вывода при определении выходного фильтра; используйте для этой цели адресатов. Адресаты должны иметь *исключительное* право доступа к устройству и должны быть единственными средствами доступа в пределах системы спулинга.

Формирование цепочки очередей

Очереди могут быть сцеплены, это позволяет перемещать задания из очереди в очередь. Ниже приведен простой пример формирования цепочки с очередью с именем tmp, которая посылает данные в адресат lpt:

```
[txt]
    qn=tmp

[tmp]
    ta=lpt

[-lpt]
    dv=/dev/par
```

Регистрация информации

Ключевое слово af позволяет определить имя файла, к которому команды могут получить доступ через $(af) и, таким образом, записать любую информацию, которую требуется зарегистрировать.

Ошибки ввода

При вызове утилиты lp будут сообщаться любые ошибки, которые появляются во время ввода данных в очередь. Если вы подаете данные путем записи непосредственно в файл спулера в каталоге /dev/spool, то ошибки будут появляться в том случае, если что-нибудь препятствует успешному копированию в очередь.

Ошибки вывода

Можно использовать ключевое слово ab, чтобы определять команду для выполнения в случае, если адресат "отказывается" от задания. В следующем примере пользователь будет извещаться об ошибке сообщением электронной почты:

```
ab=echo lpsrvr print job $(jobid), file $(file) failed \
| mailx $(username)
```

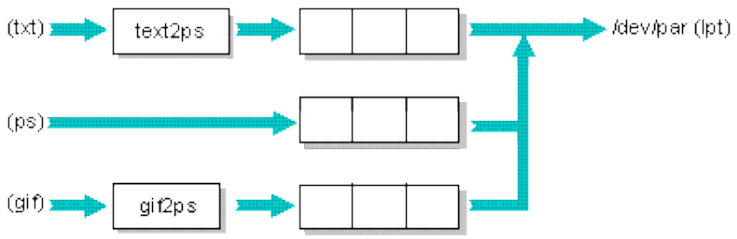
Примеры файлов установки

Несколько очередей связаны с одним адресатом

В следующем примере показан набор очередей, которые совместно используют одного общего адресата. Три очереди названы:

- `txt` (текстовые файла типа ASCII);
- `ps` (файлы PostScript);
- `gif` (файлы графического формата типа GIF).

Конфигурация выглядит следующим образом:



Файл, определяющий эту установку, имеет следующий вид:

```
# ASCII в очередь PostScript:
[txt]
    ta=lpt
    ci=text2ps
    pr#50

# прямо в очередь PostScript:
[ps]
    ta=lpt
    pr#60

# GIF в очередь PostScript:
[gif]
    ta=lpt
    ci=gif2ps
    pr#5
```

```
# принтер-адресат:
[-lpt]
    dv=/dev/par
```

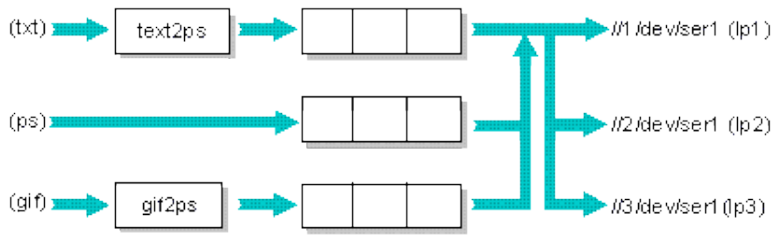
В этом примере пользователи, используя утилиту lp, посылают файлы в соответствующую очередь с входным фильтром, который преобразовывает файл (за исключением очереди ps). Затем очередь передает преобразованные данные адресату /dev/par.

Так как гипотетическим программам-фильтрам text2ps и gif2ps может потребоваться относительно большое количество времени для обработки задания, то чаще используются входные фильтры, нежели выходные фильтры. Выходной фильтр будет задерживать готовность адресата, тем самым, задерживая извлечение других заданий из очереди, пока работает фильтр. Выбранная конфигурация позволяет посылать другие задания адресату, пока генерируется перевод в PostScript.

Кроме того, так как графические файлы типа GIF обычно имеют большой размер, то им назначен более низкий приоритет, чем другим.

Несколько очередей связаны с тремя адресатам

Следующий пример является дальнейшим усовершенствованием рассмотренной выше установки с некоторыми дополнительными особенностями, необходимыми для расширенной конфигурации. Теперь имеется три принтера PostScript, расположенные в разных частях здания (подключенные к //1/dev/ser1, //2/dev/ser1 и //3/dev/ser1). Конфигурация будет выглядеть следующим образом:



Файл для такой установки имеет вид:

```
# ASCII в очередь PostScript:
[txt]
    ta=lp1,lp2,lp3
    ci=text2ps
    pr#50

# прямо в очередь PostScript:
[ps]
    ta=lp1,lp2,lp3
    pr#60
```

```
# GIF в очередь PostScript:
[gif]
    ta=lp1,lp2,lp3
    ci=gif2ps
    pr#5

# принтеры-адресаты:
[-lp1]
    dv=//1/dev/ser1
    ok=echo file $(fname) sent to $(target) \
    | mailx $(username)
    ab=echo file $(fname) did not get printed \
    | mailx $(username)

[-lp2]
    dv=//2/dev/ser1
    ok=echo file $(fname) sent to $(target) \
    | mailx $(username)
    ab=echo file $(fname) did not get printed \
    | mailx $(username)

[-lp3]
    dv=//3/dev/ser1
    ok=echo file $(fname) sent to $(target) \
    | mailx $(username)
    ab=echo file $(fname) did not get printed \
    | mailx $(username)
```

Вышеуказанная конфигурация использует те же три очереди (txt, ps и gif), описанные раньше, но теперь они связаны с тремя отдельными адресатами. Спулер выбирает первый доступный адресат из набора адресатов (lp1, lp2, lp3) и посылает данные на соответствующий принтер.

Выбор из очереди производится сначала по приоритету, затем по времени нахождения задания в очереди.

В этом примере инициатору задания посылается почтовое сообщение, информируя его о нормальном или ненормальном завершении задания.

Доступ к спулерам и очередям

В сети могут работать несколько спулеров, и каждый из них может поддерживать несколько очередей. Для того, чтобы связаться с любым заданным спулером, утилита lp сначала находит спулер по уникальному глобальному имени, которое он зарегистрировал (смотрите раздел "**Глобальные ключевые слова**").

Используя любую утилиту lp, можно:

- опускать имена спулера и очереди;

- задавать только имя спулера;
- задавать только имя очереди;
- задавать имена спулера и очереди.

Если вы не задаете в командной строке имя спулера, то утилита lp проверяет переменную окружения **LPSRVR**, которая содержит имя спулера, принятое по умолчанию. Однако, если qqqЦ2LPSRVRqqqЦ0 не определена, то утилита lp будет использовать спулер, который зарегистрирован по умолчанию под глобальным именем /qnx/spooler:

Если утилита lp успешно нашла спулер, но вы не задали в командной строке имя очереди, то утилита lp проверит переменную окружения **LPDEST**, которая содержит имя очереди по умолчанию. Однако, если **LPDEST** не определена, то утилита lp будет использовать первую очередь в файле установки спулера.

Переменные LPSRVR и LPDEST

Переменные окружения **LPSRVR** и **LPDEST** используются, когда информация командной строки, передаваемая утилитам lp, полностью не определяет, какой спулер или очередь использовать.

Переменная LPSRVR

Переменная **LPSRVR** определяет имя спулера, принимаемое по умолчанию. Следующая установка **LPSRVR** определяет как используемый по умолчанию спулер с именем /qnx/spooler2:

```
export LPSRVR=/qnx/spooler2
```

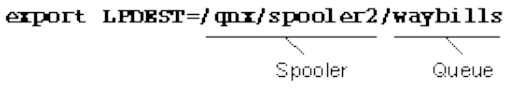
Имя, определенное в **LPSRVR**, должно всегда начинаться с ведущей наклонной черты вправо. Переменная **LPSRVR** не должна содержать имя очереди.

Переменная LPDEST

Переменная **LPDEST** определяет имя очереди, принимаемое по умолчанию. Вы можете использовать **LPDEST** двумя способами. Вы можете определить только очередь:

```
export LPDEST=waybills
```

Или вы можете определить спулер и очередь:



Если вы определяете спулер в переменной **LPDEST**, то переменная **LPSRVR** игнорируется, даже если она и определена.

Примечание. Если **LPDEST** не представлена, то для совместимости утилиты lp используют переменную окружения **PRINTER**.

Примеры

Здесь приведено несколько простых примеров, которые показывают некоторые способы задания спулеров и очередей. Для этих примеров предполагается, что в сети имеется два спулера, каждый с двумя очередями.

Первый спулер использует по умолчанию глобальное имя /qnx/spooler; его очереди имеют имена txt и ps. Второй спулер использует имя /qnx/spooler2; его очереди имеют имена checks и waybills.

Первый спулер:	**Второй спулер:**
/qnx/spooler/txt	/qnx/spooler2/checks
/qnx/spooler/ps	/qnx/spooler2/waybills

Имена спулера и очереди не заданы

Допустим, введена следующая команда lp, в которой не задано ни имя спулера, ни имя очереди:

```
lp test.dat
```

Сначала утилита будет пытаться определить спулер, проверяя **LPSRVR**. Если эта переменная не определена, то утилита определит для спулера глобальное имя /qnx/spooler, принятое по умолчанию.

Затем утилита будет пытаться определить имя очереди, проверяя **LPDEST**. Если эта переменная не определена, то утилита выберет первую очередь, определенную в файле установки спулера.

Задано только имя спулера

Если определяется строка, которая начинается с ведущей наклонной черты вправо, то строка всегда считается именем спулера. Таким образом, если вы введете следующую команду, то утилита рассмотрит имя /qnx/spooler2 как имя спулера:

```
lp -P /qnx/spooler2 test.dat
```

Так как второй спулер использует имя /qnx/spooler2, то этот спулер и будет установлен. Затем утилита будет пытаться получить имя очереди из переменной **LPDEST**. Если **LPDEST** не определена, то задание передается в очередь checks, так как она является первой очередью в файле установки /qnx/spooler2.

Задано только имя очереди

Если определяется строка, которая *не* начинается с ведущей наклонной черты вправо, то строка всегда считается именем очереди. Таким образом, если вы введете следующую команду, то утилита lp рассмотрит txt как очередь:

```
lp -P txt test.dat
```

Утилита lp сначала будет пытаться определить спулер, проверяя **LPSRVR**. Если эта переменная не определена, то утилита lp будет использовать первый спулер, так как этот спулер зарегистрирован под глобальным именем /qnx/spooler, принятым по умолчанию.

Заданы имена спулера и очереди

В следующем примере спулеру и очереди определены имена. Так как строка начинается с ведущей наклонной черты вправо, то утилита lp сначала будет пытаться обнаружить спулер с именем /qnx/spooler2/waybills.

```
lp -P /qnx/spooler2/waybills test.dat
```

Однако, так как спулера с именем /qnx/spooler2/waybills не существует, то поиск потерпит неудачу, и утилита будет рассматривать данную строку как имя спулера, за которым следует имя очереди. При этом будет определен спулер с именем /qnx/spooler2, и задания будут передаваться очереди waybills в этом спулере.

Файлы инициализации

Полезно сконфигурировать несколько спулеров в качестве используемых по умолчанию. Если, например, отделы маркетинга, продаж и техотдел имеют свои спулеры:

`LPSRVR=/qnx/spooler` (отдел маркетинга)
`LPSRVR=/qnx/spooler2` (отдел продаж)
`LPSRVR=/qnx/spooler3` (техотдел)

Обычно системные переменные инициализируются заранее в файлах инициализации системы. Для этого можно использовать один из следующих файлов:

`/etc/config/sysinit.node` (выполняется при загрузке)
`/etc/default/login` (выполняется при входе в систему)

`/etc/profile` (выполняется при каждом запуске командного интерпретатора).

Глава 10. Создание резервных копий

Эта глава охватывает следующие темы:

- Введение
- Когда создавать резервные копии
- Форматы резервных копий
- Носители резервных копий
- Сжатие данных
- Примеры создания архивов

Введение

В этой главе рассматривается вопрос создания копий данных для того, чтобы избежать потери данных при сбоях аппаратных средств, программного обеспечения или ошибках оператора, которые могут разрушить оригинал. Если ваши данные важны для вас, то вам следует регулярно выполнять процедуры резервного копирования, что позволит вам восстановить потерянную информацию с минимальными затратами времени и денег.

Помните: жесткие диски *могут* выйти из строя, а люди *могут* ошибаться, а также поздно начинать изготовление резервных копий - после того, как ваши данные уже потеряны!.

Вы можете копировать целые файловые системы или только их части. Пользователи могут решить копировать только свои собственные файлы, обычно на дискеты. Для того чтобы копировать большие части файловой системы с файлами, принадлежащими многим пользователям, вам понадобится разрешение на чтение этих файлов. Суперпользователь (root) имеет такие привилегии.

Когда создавать резервные копии

Вам следует копировать данные так часто, чтобы можно было восстановить данные, которые все еще являются текущими или на основе которых могут быть восстановлены текущие данные с минимальными затратами. В группе разработки программного обеспечения частота копирования может изменяться в пределах от дня до недели. Каждый день старения копии обычно будет стоить вам дня работы. Если вы сохраняете финансовые или кассовые данные, резервное копирование выполняется, как правило, ежедневно или даже дважды в день. Рекомендуется вести архив не на рабочей установке.

Форматы резервных копий

QNX поддерживает целый ряд форматов резервных копий, которые могут быть распределены на две группы:

- архивы;

- регулярные файловые системы.

Архив состоит из одного или более файлов, объединенных в единый модуль с собственным каталогом содержимого. Архив может быть сохранен или в регулярном файле QNX, или на низком уровне на блок-ориентированном устройстве, например дискете или ленте.

Сохранение в *регулярной файловой системе* заключается просто в копировании файлов. В этом случае адресатом должно быть устройство с установленной на нем файловой системой.

Архивные копии

QNX поддерживает три основные утилиты архивирования:

- cpio;
- tar;
- pax.

Утилиты cpio и tar связаны с утилитой pax, которая может читать и записывать в обоих форматах, cpio и tar. Утилиты cpio и tar являются стандартом для систем UNIX. Утилита pax включает cpio так и tar, таким образом, она не поддерживает собственный архивный формат. По умолчанию pax будет использовать формат tar при создании архива.

Утилита pax определит, когда будет достигнут конец диска или ленты для тома, и подскажет, чтобы вы вставили следующий том, который нужно использовать для сохранения. Результат представляет собой резервную копию, распределенную по нескольким носителям (дискетам, лентам и т.п.).

Присваивание имен томам

К сожалению, формат tar/cpio не присваивает метки тома носителю. Если вы перепутаете ваш носитель или вставите его не в нужном порядке, это приведет к восстановлению ошибочных данных.

Для того чтобы исключить эту возможность, в QNX предусмотрена утилита vol, которая метит каждый дисковый том последовательным номером и, следовательно, предотвращает установку носителей не в том порядке.

Примечание. Утилита vol может использоваться с дискетами или сменными дисками. Эта утилита не может использоваться с ленточными накопителями.

Утилита vol будет, по умолчанию, пропускать первый блок носителя. Это важно для дискет и кассетных дисков, которые содержат QNX-сигнатуру в первом блоке. Эта сигнатура содержит размер дискеты (360K, 1.2M и т.п.) и обеспечивает автоматическое перемонтирование сменного носителя файловой системы.

QSSL поставляет свои дистрибутивные дискеты, используя pax для создания архива, freeze - для сжатия данных и vol - для записи на гибкие диски.

Примечание. Если вы хотите сохранить данные для их последующего восстановления в UNIX-системе, то не используйте утилиты freeze или vol, так как вы не найдете там утилит, которые выполнят восстановление. Вместо них используйте утилиту pax для сохранения и восстановления непосредственно на носителе.

Создание резервных копий файловой системы

Существует возможность сохранения файловой системы посредством копирования файлов утилитами cp или cpio **-p**. Если ваш носитель расположен на гибком диске, то утилита cp подскажет вам о необходимости установки дискеты, но помните, что размер файлов не должен превышать емкость дискеты. Если вы хотите поддерживать копии на гибких дисках, то рекомендуется использовать одну из утилит, предназначенных для создания архивов.

Носители резервных копий

Ваш выбор носителя резервных копий будет определяться доступным аппаратным обеспечением и его стоимостью. Имеется несколько вариантов выбора:

- гибкий диск;
- магнитная лента;
- сменный диск;
- жесткий диск.

QNX имеет драйверы для нескольких типов SCSI-контроллеров. Эти драйверы определяют и поддерживают жесткие диски, накопители на магнитных лентах с последовательным доступом, устройства WORM (однократная запись/многократное чтение), дисководы CD-ROM и оптические дисководы.

Гибкие диски

Гибкие диски - наиболее доступные носители для персональных резервных копий. Их основной недостаток - ограниченный объем. Так как в QNX утилиты pax и vol позволяют распределить копию по носителям, то вам нужно только обеспечить установку нескольких дискет в накопитель. Если вы имеете дело с более чем четырьмя или пятью дискетами, то процедура копирования станет достаточно неприятной, что влечет за собой уменьшение частоты копирования. Вы можете использовать сжатие ваших данных, как описано ниже.

Для того чтобы скопировать/восстановить с дискеты, вы должны убедиться, что запущен драйвер гибких дисков Fsys.floppy (см. описание Fsys.floppy в книге *Описание утилит*).

Следующая командная строка запустит драйвер (предполагается, что Fsys уже запущен):

```
Fsys.floppy &
```

По умолчанию, Fsys.floppy создает блок-ориентированное устройство для дисковода гибких дисков номер 0 (/dev/fd0). Если ваш компьютер имеет более чем один дисковод для гибких дисков, то Fsys.floppy создает блок-ориентированное устройство для каждого накопителя. Например, дисковод номер 1 будет иметь имя /dev/fd1.

Архивирование данных на дискетах

Когда вы используете архивную утилиту для ваших копий, то нужно помнить, что утилита считывает и записывает данные, начиная прямо с первого блока. Утилита tar также пишет архивные данные в первый блок, поэтому первый блок на дискете будет содержать 512 байтов исходных данных. Это может привести к проблеме для Fsys.floppy, который пытается читать первый блок на всех дискетах как QNX-сигнатуру.

Чтобы исключить непредсказуемое поведение при архивировании или восстановлении архивированных данных из гибкого диска, прежде чем вы начнете копировать, запустите утилиту lockfd. Утилита lockfd заставляет драйвер Fsys.floppy зафиксировать специфический размер носителя и подавляет попытку драйвера читать информацию сигнатуры из первого блока при восстановлении данных из дискеты.

Копирование файлов на гибкий диск

Вы должны отформатировать и инициализировать каждую неформатированную дискету прежде, чем вы начнете копировать на нее файлы. В следующем примере показано, как инициализировать гибкий диск высокой плотности $3^1/_2$":

```
fdformat -s 1.4m /dev/fd0
dinit /dev/fd0
```

Далее вы должны смонтировать файловую систему на блок-ориентированном устройстве:

```
mount /dev/fd0 /fd0
```

Теперь можно рассматривать дискету как файловую систему QNX, смонтированную как /fd0. Для копирования файлов на дискету используйте утилиту cp.

Магнитная лента

Чтобы поддерживать сохранение и восстановление копий на лентах, вы должны удостовериться, что необходимый драйвер запущен. Обычно это один из SCSI-драйверов, описанных в книге ***Описание утилит***.

Например, если вы имеете SCSI-контроллер, совместимый с Adaptec 1540, и подключенный ленточный накопитель, то следующая команда запустит драйвер и зарегистрирует ленту с

последовательным доступом как /dev/tape1:

```
Fsys.aha4scsi fsys -n 1=tape
```

Утилиты архивирования будут читать и записывать непосредственно на ленту блок-ориентированные файлы. Вы не можете монтировать файловую систему с этим типом блок-ориентированного файла.

Когда драйвер AHA 4 получает запрос на чтение или запись, он начинает выполнение операции, начиная с текущей позиции ленты. Если на ленте необходимо начать записывать новую копию, то вы должны будете стереть данные с ленты и перемотать ленту на начало.

Поиск номера позиции на ленте и функции управления обеспечиваются утилитой tape, которая описана в книге *Описание утилит*. Например, следующая команда перемотает и сотрет ленту, затем перемотает ленту на начало и подготовит ее к процедуре архивирования:

```
tape rewind erase
```

Сменные диски

В настоящее время жесткие сменные диски распространены как в магнитном так, и в оптическом формате. Все большее количество модулей в компьютерах используют интерфейс SCSI, так что вы можете рассмотреть возможность установки жесткого несъемного диска с SCSI-дисководом (тогда вы не будете нуждаться во втором контроллере и драйвере).

В отличие от дискет и лент, жесткий сменный диск позволяет вам избегать использования утилит архивирования, таких как tar. Вместо этого вы будете, наверное, использовать утилиту cp или pax -rw для копирования ваших данных на файловую систему сменного диска. Это позволит вам восстанавливать одиночные файлы очень легко и быстро.

Жесткий диск

Вы можете установить в компьютере второй жесткий диск для копирования в режиме online. Как вариант, вы могли бы рассмотреть создание копий на жестком диске в другой машине в сети. Однако, копирование через сеть медленнее, чем копирование на локальный жесткий диск. Если имеется жесткий дополнительный диск, то можно спланировать все операции копирования как задание для утилиты cron, которая выполнит их, например, в ночное время.

Сжатие данных

Вы можете использовать утилиту сжатия, чтобы уменьшить размер дискового пространства, требуемого для сохранения данных. Степень сжатия будет зависеть от характера данных, которые вы сохраняете. Некоторые базы данных, содержащие большое количество повторяемых данных, могут быть сжаты до 90%. Отдельные же данные могут быть сжаты

менее чем на 10%.

Примечание. Хотя сжатие и сохраняет свободное пространство носителя, но имеются два нежелательных побочных эффекта:

- сжатие данных требует значительного количества вычислений, что может замедлять процесс сохранения данных;
- вы не сможете восстановить сжатые данные в случае возникновения сбоя во время процесса сжатия данных (и хранения носителя). Есть вероятность, что один дефектный блок данных может повлиять на остальные данные (т.е. увеличивается вероятность потери всех данных в архивном файле).

Вы можете использовать утилиты gzip или freeze, чтобы сжать ваши данные, и утилиты gunzip или melt, чтобы восстановить их. Обе утилиты работают как с потоком данных, так и с файлами. Эта способность работать, как фильтр позволяет вам присоединять их к одному из стандартных архиваторов через конвейер. Например, QSSL распространяет операционную систему QNX в сжатой форме на дискетах, используя pax, freeze и vol.

Примеры создания архивов

Сжатые данные в архиве на дискетах

Собрать файлы в каталоге /home в архивном формате tar, сжать архив и записать его на такое количество дискет, которое потребуется, проставляя порядковые номера на дискеты:

```
pax -w -x ustar /home | freeze | vol -w /dev/fd0
```

Считать данные с дискет, распаковать данные и восстановить файлы:

```
vol -r /dev/fd0 | melt | pax -r
```

UNIX-совместимые архивы на дискетах

Сохраните файлы в каталоге /home/dino в архивном формате tar для восстановления в системе UNIX:

```
pax -w -t/dev/fd0 -xustar /home/dino
```

Сохраните файлы в каталоге /home/dino в архивном формате cpio для восстановления в системе UNIX:

```
pax -w -xcpio -t/dev/fd0 /home/dino
```

Восстановите данные в архивном формате tar или cpio из другой системы UNIX и поместите все файлы в каталог /usr/unix:

```
pax -r -s -t/dev/fd0  ",/,/usr/unix/,"
```

Архив на ленте

Начните на ленте новый архив и сохраните все файлы на ленте:

```
tape rewind erase    (стереть данные на ленте и перемотать ее)
pax -w -t/dev/tp0 /
```

Добавьте файлы, которые изменились позднее даты последней модификации файла lastsave, в конец существующей архивной ленты. После сохранения обновите время последней модификации lastsave:

```
tape forward
touch newsave
find / -newer lastsave | pax -w -t/dev/tp0
mv newsave lastsave
```

Восстановите все файлы с ленты в каталоге /home/dino:

```
tape rewind
pax -r "/home/dino/*" -t/dev/tp0
```

Сменные магнитные/оптические диски

Скопируйте все файлы с файловой системы на узле 1 на файловую систему на узле 2:

```
cp -Rp //1/ //2/
```

В следующем примере на узле 2 очень большой оптический диск. Создание полной копии выполняется каждую пятницу, а создание частичной копии измененных файлов может выполняться каждый день недели:

```
cp -Rp //1/ //2/fri
cp -Rp -a date //1/ //2/mon
cp -Rp -a date //1/ //2/tue
.
.
.
```

Вы можете также выполнять это с помощью cpio **-p**.

Глава 11. Восстановление дисков и файлов

Эта глава охватывает следующие темы:

- Введение
- Создание восстановительной дискеты
- Краткий обзор структуры диска QNX
- Утилиты сопровождения файла
- Процедуры восстановления дисков
- Что делать, если система больше не загружается
- Восстановление потерянных файлов и каталогов

Введение

Файловая система QNX имеет высокую производительность и надежность. Хотя файловая система разрабатывалась в соответствии с принципом обеспечения максимальной живучести, тем не менее, на практике всегда могут возникнуть ситуации отказа диска. Могут выйти из строя аппаратные средства, может произойти сбой по питанию и т.п.

Файловая система QNX спроектирована так, что может устоять при возникновении подобных ситуаций. Она базируется на принципе обеспечения целостности и непротиворечивости файловой системы в целом в любой момент времени. В то время как большинство данных содержится в буферном кэше и записывается только после короткой задержки, критические данные файловой системы записываются немедленно. Изменения в каталогах, индексных дескрипторах (inode), блоках экстентов и битовых картах немедленно записываются на диск, чтобы гарантировать, что структура файловой системы на диске никогда не разрушится (то есть данные на диске никогда не будут внутренне противоречивыми).

Если происходит сбой, можно использовать следующие утилиты сопровождения и восстановления:

- `fdisk`
- `dinit`
- `chkfsys`
- `dcheck`
- `zap`
- `spatch`

Эти утилиты позволят определить, было ли какое-то повреждение в файлах, которые были открыты на запись в момент возникновения сбоя. Эти же самые утилиты могут исправить такие повреждения и, во многих случаях, полностью восстановить файловую систему.

Иногда повреждения могут быть более серьезными. Например, на жестком диске может появиться плохой блок в середине файла или, еще хуже, - в середине каталога или какого-то другого критического блока.

И в этом случае предоставляемые утилиты могут помочь определить степень такого повреждения. Может потребоваться перестроить файловую систему таким образом, чтобы избежать поврежденных областей. В этом случае некоторые данные будут потеряны, но, приложив некоторые усилия, большую часть используемых данных можно восстановить.

Создание восстановительной дискеты

Всегда надо иметь под рукой *восстановительную дискету*, если, по любой причине, машина не будет загружаться с жесткого диска.

Примечание. Эта процедура выполняется только для систем QNX, поставляемых на дискетах. Если ваша система поставлена на CD-ROM, то следует обратиться к техническим замечаниям в файле /etc/readme/technotes/qnx_install, где задокументирован сценарий для создания загрузочной дискеты.

Прежде, чем начать создание восстановительной дискеты, удостоверьтесь, что вы зарегистрированы как root и что Fsys.floppy выполняется.

Теперь выполним следующие шаги:

1. Вставьте загрузочную дискету QNX в ваш дисковод гибких дисков.

2. Скопируйте образ во временный файл на вашем жестком диске:

```
dd if=/dev/fd0 of=/tmp/floppy_image
```

3. Вставьте пустую дискету в дисковод. Отформатируйте дискету:

```
fdformat -k0 -z2 /dev/fd0
```

4. Скопируйте образ (из вашего временного файла) на дискету:

```
dd if=/tmp/floppy_image of=/dev/fd0
```

5. Выполните dcheck, чтобы проверить новую дискету:

```
dcheck /dev/fd0
```

   Если произошел сбой, повторите шаги 3 и 4 (fdformat и dd); если и во второй раз произошел сбой, попробуйте новую дискету.

6. Смонтируйте файловую систему дисковода гибких дисков:

```
mount /dev/fd0 /fd
```

7. Чтобы освободить немного места на дискете, удалите следующее:

- утилиту disktrap (14K):

```
rm /fd/bin/disktrap
```

- все драйверы диска из /fd/bin, которые не нужны для этой машины;

8. Теперь скопируйте эти полезные утилиты в /fd/bin:

```
cp /bin/sin /fd/bin/sin
cp /bin/zap /fd/bin/zap
cp /bin/rm /fd/bin/rm
cp /bin/ls /fd/bin/ls
cp /bin/spatch /fd/bin/spatch
cp /bin/chkfsys /fd/bin/chkfsys
cp /usr/bin/elvis /fd/bin/elvis
```

9. Создайте каталог /etc:

```
mkdir /fd/etc
```

10. Скопируйте файл termcap:

```
cp /etc/termcap /fd/etc/termcap
```

    Также необходимо отредактировать файл termcap и удалить ненужные записи. Единственной нужной является запись для QNX;

11. Теперь создайте две связи:

```
cd /fd/bin
ln -s elvis vi
ln -s fcat melt
```

12. И в заключение, следует изменить файл инициализации системы (/fd/etc/config/sysinit) с тем, чтобы он теперь содержал эти строки:

```
Dev -n 10 &
Dev.con -n 4 -O 256 &
reopen /dev/con1
export PATH=/ram:.:/bin:/usr/bin
export HOME=/
dinit /dev/ram
mount /dev/ram /ram
prefix -A /pipe=/ram
prefix -A /tmp=/ram
fcat /util.tar.F | pax -vr
cp /bin/esh /ram/sh
melt -z </etc/logo.F
rtc hw
echo Welcome to QNX 4.25
```

```
ontty /dev/con1 /bin/sh
ontty /dev/con2 /bin/sh
```

Вот и все! Храните вашу восстановительную дискету в безопасном месте. Если когда-нибудь вам понадобится использовать ее, просто вставьте дискету в выключенную машину и включите питание - машина будет загружать QNX с дискеты.

Краткий обзор структуры диска QNX

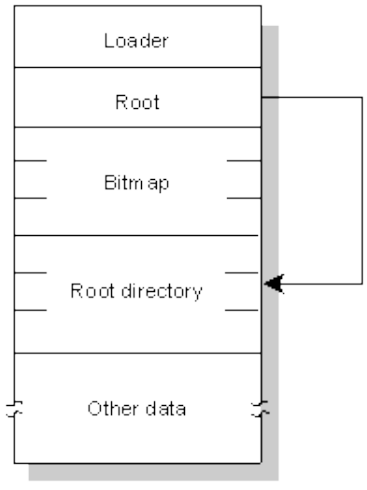
В этом разделе описано, как файловая система QNX сохраняет данные на диске. Чтение этого раздела поможет распознать и, возможно, исправить повреждения файловой системы, если вам когда-то придется ее восстанавливать.

Если у вас имеется пакет разработки Си, то в заголовочном файле <sys/fsys.h> содержатся определения всех терминов, используемых в этом разделе.

Полное описание файловой системы QNX содержится в главе "**Менеджер файловой системы**" книги *Системная архитектура*.

Структура раздела

Файловая система QNX может занимать целый диск (в случае дискет) или один из разделов жесткого диска. В пределах дискового раздела, файловая система QNX содержит следующие компоненты:



В дисковом разделе QNX указанные ниже блоки всегда находятся в таком порядке:

- загрузчик;
- корень;
- битовая карта;

- корневой каталог;
- другие данные.

Блок загрузчика

Блок загрузчика - первый блок раздела QNX. Он содержит программу начальной загрузки, которая загружает ОС QNX в память.

Корневой блок

Корневой блок - второй блок раздела QNX. Он содержит элемент каталога для корня (/), inode-элементы для файла inode и поле меток.

Блоки битовой карты

За корневым блоком следует несколько блоков битовой карты. Они образуют битовую карту для раздела QNX. Каждому блоку раздела соответствует один бит, таким образом один блок битовой карты будет использоваться для каждых 4096 дисковых блоков (что соответствует 2 Мб дискового пространства).

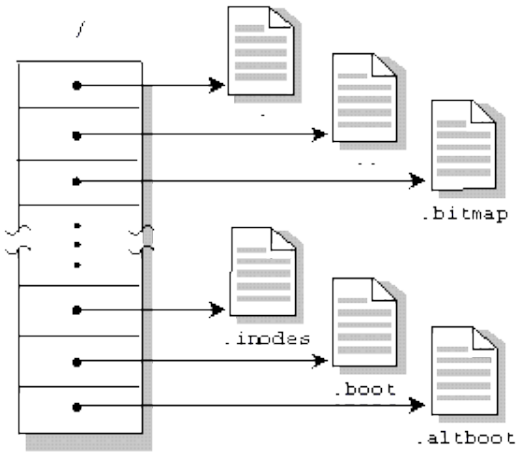
Если значение бита нулевое, то соответствующий ему блок не использован. Неиспользуемые биты в конце последнего блока битовой карты (для которых нет соответствующих дисковых блоков) имеют значение единица.

Присвоение битов начинается с младшего бита байта 0 первого блока битовой карты, который соответствует блоку 1 раздела QNX.

Корневой каталог

Корневой каталог следует за блоками битовой карты. Корневой каталог является "нормальным" каталогом (смотрите ниже "**Каталоги**") и первоначально создается утилитой dinit с объемом, достаточным для 32 элементов каталога (4 блока).

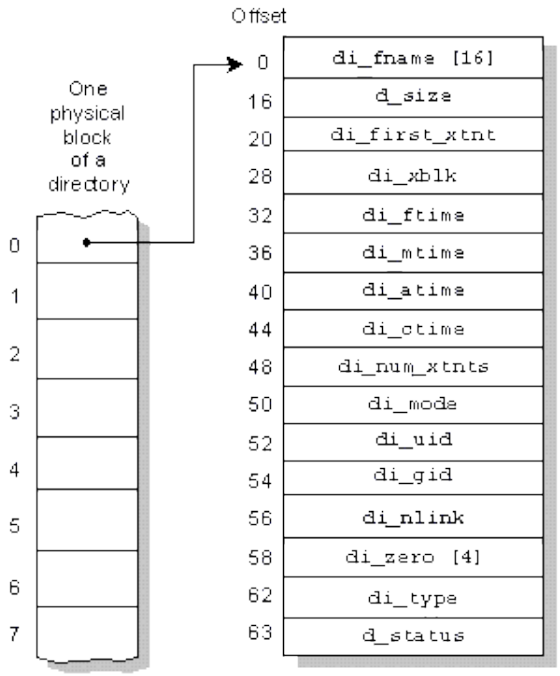
Как показано на следующей иллюстрации, корневой каталог (/) содержит элементы каталогов для различных специальных файлов, которые всегда существуют в файловой системе QNX. Утилита dinit создает эти файлы при инициализии файловой системы.



Файл:	**Описание:**
/.	Ссылка на каталог /
/..	Тоже ссылка на каталог /
/.bitmap	Файл "только для чтения", состоящий из блоков битовой карты.
/.inodes	Нормальный файл, занимающий, по крайней мере, один блок на дискете или ОЗУ-диске и 16 блоков на других дисках, /.inodes - это набор inode-элементов. Первый элемент резервируется и используется в качестве области сигнатуры. Первые байты файла .inode - "IamTHE.inodeFILE".
/.boot	Файл образа ОС, который будет загружаться в память при *стандартной* начальной загрузке. Этот файл будет иметь нулевую длину, если файла начальной загрузки не существует.
/.altboot	Файл образа ОС, который будет загружаться в память при *альтернативной* начальной загрузке. Этот файл имеет нулевую длину, если файла альтернативной начальной загрузки не существует.

Каталоги

Каталог - это просто файл, который имеет специальное значение для файловой системы. Этот файл содержит совокупность элементов каталога, как показано на следующем рисунке:



Тип элемента каталога определяется битами в поле d_status следующим образом:

Бит 3 (_FILE_LINK)	**Бит 0 (_FILE_USED)**	**Комментарий:**
0	0	Неиспользованный элемент каталога
0	1	Нормально использованный элемент каталога
1	0	Ссылка на элемент в /.inodes (который должен использоваться)
1	1	Недопустимо

Первый элемент каталога всегда принадлежит файлу "." и включает сигнатуру каталога ("I[символ_сердце]QNX"). Шестнадцатеричный эквивалент символа [символ_сердце] - 0x03. Этот элемент ссылается на собственный каталог, указывая на элемент в родительском каталоге, описывающий этот каталог.

Второй элемент всегда принадлежит файлу "..". Этот элемент ссылается на родительский каталог, указывая на его первый блок.

Каждый элемент каталога или определяет файл, или ссылается на запись в файле /.inodes. Inode-строки используются, когда имя файла превышает 16 символов или когда один файл имеет два или более имени.

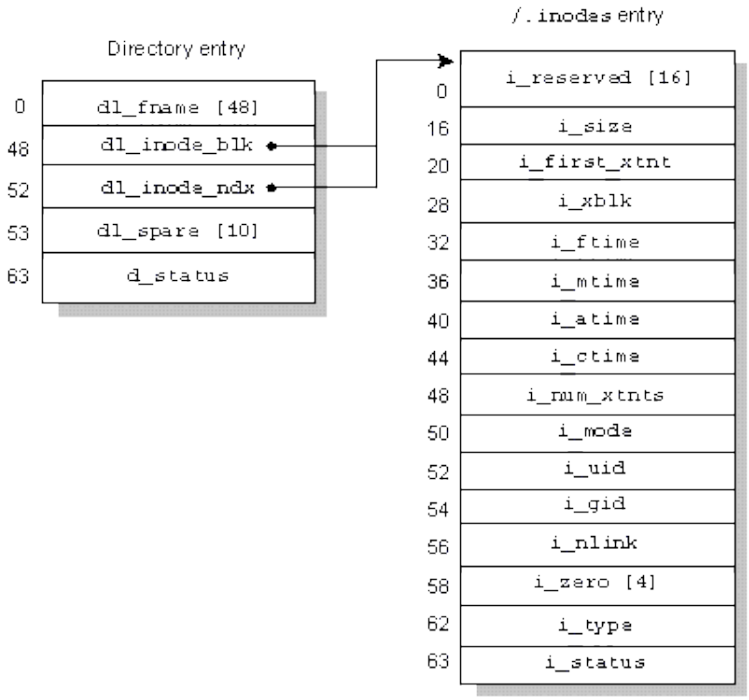
Первый экстент (если он существует) файла описывается в элементе каталога или inode-файле. Дополнительные экстенты файла требуют связный список (т.е. список с

использованием указателей на предыдущий/следующий элемент списка) блоков экстентов, начало которого также находится в элементе каталога или inode-файла. Каждый блок экстентов в цепочке определяет от 1 до 60 экстентов.

Связи

Файлы с именами, имеющими более 16 символов, и *связи* с другими файлами реализованы специальной формой строки каталога. Эти строки определяются битом _FILE_LINK (0x08) поля d_status.

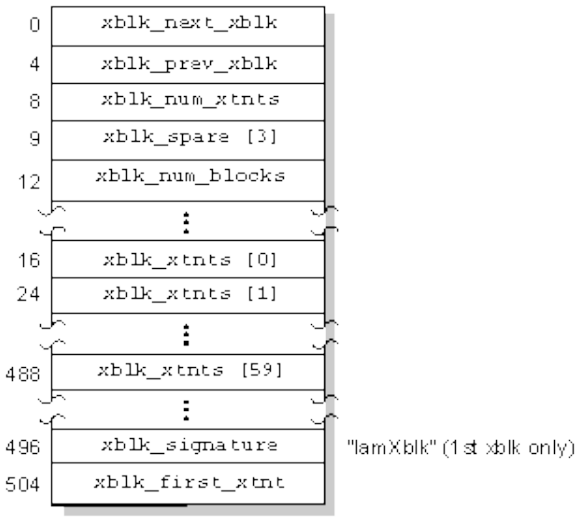
Для этих файлов часть строки каталога перемещается в файл /.inodes.



Блоки экстентов

Блоки экстентов используются для любого файла, который имеет более одного экстента. Элемент каталога di_xblk *указывает* на первый блок экстентов, который, в свою очередь, определяет, где должны находиться второй и последующие экстенты.

Блок экстентов соответствует одному дисковому блоку в 512 байт и имеет следующий формат:



Каждый блок экстентов содержит:

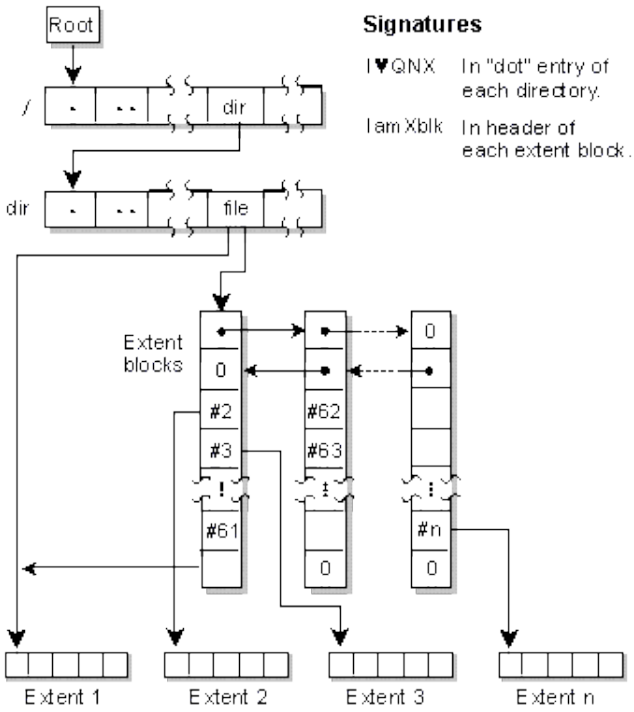
- прямые/обратные указатели;
- количество экстентов;
- количество всех блоков во всех экстентах, определяемых этим блоком экстентов;
- указатели и количество блоков для каждого экстента;
- сигнатуру ("IamXblk").

Первый блок экстентов также содержит резервный указатель на первый экстент файла (который описан еще и в элементе каталога или inode-файла). Это позволяет восстанавливать все данные в файле, располагая только этим блоком.

Файлы

Файлы или файловые экстенты - это группы блоков, описываемые элементами каталогов и inode-файлов; у них нет жесткой структуры в файловой системе QNX.

Большинство файлов в QNX имеет следующую общую структуру:



Утилиты сопровождения файла

Утилита fdisk

Утилита fdisk создает и поддерживает *блок раздела* на жестком диске. Этот блок совместим с другими операционными системами и может быть поддержан версиями fdisk других ОС (хотя наша имеет преимущество при распознавании QNX-специфической информации). Если загрузчик раздела отсутствует или поврежден, fdisk может создать его.

Примечание. Рекомендуется сохранить распечатку таблицы разделов для каждого диска в вашей сети.

Утилита dinit

Утилита dinit создает (но Fsys поддерживает) следующее:

- блок начальной загрузки;
- корневой блок;
- блоки битовой карты;
- корневой каталог;
- файл /.inodes.

Утилита chkfsys

Утилита chkfsys - основное средство сопровождения файловой системы. Эта утилита:

- проверяет структуру каталога всего раздела диска, сообщает о любом несоответствии и исправляет его, если возможно;
- выполняет полную проверку распределения блоков диска;
- записывает новый файл /.bitmap при вашем согласии.

Утилита chkfsys работает при условии, что корневой блок не нарушен. Если корневой блок нарушен, то утилита chkfsys сообщит об этом и прекратит выполнение. В этом случае попытайтесь восстановить его утилитой dinit.

Утилита dcheck

Утилита dcheck проверяет, правильно ли отформатирован диск, пытаясь считать каждый блок на дисководе. Если задана опция **-m**, то dcheck удаляет все плохие блоки из битовой карты (/.bitmap).

Если найден файл /.bad_blks, то dcheck будет корректировать битовую карту и переписывать файл /.bad_blks. Можно выполнить dcheck несколько раз, чтобы увеличить шансы распознать плохие блоки и добавить их в файл /.bad_blks.

Утилита zap

Утилита zap позволяет root удалять файлы или каталоги из файловой системы, не возвращая используемые блоки в свободный список. Это может понадобиться по многим причинам, включая следующие:

- повреждены элементы каталога;
- два файла занимают одно и то же пространство на диске (ошибочно).

Восстановление файла, удаленного утилитой zap

Если вы по ошибке удалили файл утилитой zap, то этот файл иногда возможно восстановить, используя утилиту zap с опцией **-u** *немедленно* после стирания. Можно восстановить такой файл, используя zap, при следующих условиях:

элемент каталога для этого (теперь удаленного) файла *не* был повторно использован;
блоки диска, ранее использовавшиеся файлом, *не* переназначались другому файлу.

Утилита spatch

Утилита spatch позволяет просматривать диск на низком уровне и устранять некоторые проблемы. Иногда можно устранить кратковременные сбои диска, считывая и перезаписывая неисправный блок посредством утилиты spatch.

Процедуры восстановления диска

Использование утилиты chkfsys

Утилита chkfsys - это главное средство для проверки и восстановления поврежденной файловой системы. С ее помощью можно определить и исправить большинство небольших проблем, а также проверить целостность дисковой системы в целом.

Обычно для утилиты chkfsys требуется, чтобы файловая система была не задействована и чтобы не было файлов, открытых на данном устройстве. Должны быть завершены все процессы, которые открыли файлы или могут их открыть, пока выполняется chkfsys.

Чтобы выполнить chkfsys на точке монтирования, просто введите:

```
chkfsys /
```

Утилита просматривает весь дисковый раздел от корня вниз, создавая внутреннюю копию битовой карты, и проверяет непротиворечивость всех файлов и каталогов, которые она находит в процессе.

После завершения обработки всех файлов chkfsys сравнит внутреннюю битовую карту с битовой картой на диске. Если они совпадают, то выполнение chkfsys завершается. При обнаружении несоответствия утилита chkfsys перепишет (с вашего согласия) битовую карту в соответствии с данными о файлах, которые она обнаружила и проверила.

Кроме того, проверяя распределение блоков (битовую карту), chkfsys пытается фиксировать любые обнаруженные проблемы. Например, chkfsys может:

- "освободить" файлы, которые записывались в момент краха системы;
- откорректировать размер файла в элементе каталога, чтобы он соответствовал реальным данным.

Когда выполнять chkfsys

Рекомендуется выполнять chkfsys как часть регулярных плановых процедур сопровождения - это позволит проверять целостность данных на диске. Например, возможен вариант выполнения chkfsys на ваших сетевых серверах каждый раз, когда они загружаются.

Автоматизированная проверка в файловой системе во время начальной загрузки гарантирует, что chkfsys будет пытаться устранить любые проблемы, которые она найдет во время просмотра. Чтобы автоматизировать этот процесс, добавьте chkfsys в файл sysinit.*узел* сервера.

Особенно важно выполнять chkfsys после системного отказа, сбоя по питанию или непредвиденной перезагрузки системы, чтобы определить поврежденные файлы. Утилита chkfsys проверяет флажок "чисто" ("clean") на диске, чтобы определить, была ли система в тот момент в устойчивом состоянии.

Флажок "чисто" сохраняется на диске и поддерживается системой. Флажок сбрасывается всякий раз, когда файл открывается для модификации, и устанавливается после того, как все открытые файлы закроются и все связанные с ними данные будут сброшены из кэша на диск. Когда флажок "чисто" установлен, chkfsys считает, что файловая система не повреждена. Если chkfsys обнаруживает флажок очистки сброшенным, она пробует устранить проблему.

Утилита chkfsys поддерживает опцию **-u**, которая позволяет игнорировать флаг "чисто" и вызывает безусловное выполнение chkfsys. Потребность игнорировать флажок "чисто" может возникнуть, когда:

- dcheck обнаруживает дефектные блоки;
- файлы были преднамеренно стерты или удалены утилитой zap;
- вы хотите сделать общую проверку исправности.

Использование chkfsys в работающей системе

Утилита chkfsys обычно требует эксклюзивного использования файловой системы для обеспечения *всесторонней* проверки диска.

Примечание. Имеется некоторый риск при запуске утилиты chkfsys на работающей системе - и chkfsys, и файловая система читают и, возможно, записывают на диске одни и те же блоки. Кроме того, файловая система имеет внутренние кэшированные данные о файлах и каталогах, которые не могут быть откорректированы, когда chkfsys делает изменение. Но статические изменения в файлах или каталогах, которые не открыты Fsys в настоящее время, *вероятно*, не вызовут проблем.

Если вы выполняете приложение, которое не может прерваться, или вы не смогли выполнить chkfsys, потому что были открытые файлы, попробуйте выполнить chkfsys с опцией **-f**:

```
chkfsys -f /dev/hd0t77
```

Это вызывает специальный режим "только для чтения" chkfsys. Он даст вам представление об общем состоянии файловой системы.

Восстановление дефектного блока в середине файла

На жестких дисках со временем могут появляться дефектные блоки. В некоторых случаях можно восстановить большую часть или даже все данные файла, содержащего дефектный блок.

Некоторые дефектные блоки являются результатом сбоев по питанию или низкого качества носителя жесткого диска. В этих случаях просто чтение и последующая перезапись блока "восстановит" блок на короткий период времени. Это позволит скопировать весь файл куда-нибудь еще, прежде чем блок испортится снова. Данная процедура, несомненно, не причинит вреда и часто стоит того, чтобы ее выполнили.

Для того чтобы проверить блоки в файле, используют утилиту spatch. При чтении дефектного блока spatch сообщает об ошибке, но в действительности, возможно, была прочитана часть "хороших" байтов из этого блока. Запись того же блока обратно часто достигает цели.

В то же самое время, spatch перезапишет правильную CRC (контрольную сумму избыточного циклического кода), что вновь сделает блок хорошим (но, возможно, с неправильными данными).

Вы можете затем скопировать весь файл куда-нибудь еще и потом утилитой zap удалить поврежденный файл. Для того чтобы завершить процедуру, необходимо пометить сбойный блок как дефектный (добавить его в файл /.bad_blks), затем восстановить остальные хорошие блоки, используя утилиту chkfsys.

Если эта процедура не поможет, следует использовать утилиту spatch, чтобы скопировать по возможности большую часть файла в другой файл, и затем утилитой zap удалить дефектный файл и выполнить chkfsys.

Что делать, если ваша система больше не загружается

Если ранее работавшая система QNX внезапно прекратила работу и больше не загружается, то, возможно, произошло следующее:

- отказали аппаратные средства или были повреждены данные на жестком диске;
- кто-то изменил/переписал файл начальной загрузки или изменил файл системной инициализации - это два наиболее типичных случая.

Следующие шаги могут помочь идентифицировать проблему. Там, где это возможно, рекомендуются корректирующие действия.

Шаг 1 - Попробуйте загрузиться из дискеты или через сеть

Если у вас имеется сеть для загрузки, попытайтесь загрузить вашу машину по сети. Если

машина загружается, следует войти в систему как root, а затем запустить локальную файловую систему:

```
Fsys &
```

Если у вас нет сети, следует загружаться с вашей восстановительной дискеты (описанной ранее в этом разделе) или с загрузочной дискеты QNX, которая использовалась для установки системы на жесткий диск. В данном случае файловая система будет уже выполнена, и вы будете зарегистрированы как root.

Шаг 2 - Запустите драйвер жесткого диска

Теперь нужно запустить подходящий драйвер жесткого диска. Например, чтобы запустить драйвер для адаптера Adaptec 4 SCSI, напечатайте:

```
Fsys.aha4scsi &
```

Если вы используете другой тип драйвера, введите другое имя вместо этого.

Эта команда должна создать блок-ориентированный файл с именем /dev/hd0, который представляет весь жесткий диск.

Шаг 3 - Выполните fdisk

Выполнение утилиты fdisk немедленно даст полезную информацию о состоянии жесткого диска.

Утилита fdisk может сообщить об одной из нескольких типов проблем:

Проблема:	**Вероятная причина:**	**Действия:**
Ошибка чтения блока 1.	Произошел отказ либо дискового контроллера, либо жесткого диска.	Если диск исправен, то, заменив плату контроллера, *можно* продолжить использование диска. В противном случае следует заменить жесткий диск, вновь установить QNX и восстановить ваши файлы с копии.
Неправильные дисковые параметры.	Аппаратные средства, вероятно, "потеряли" информацию об этом жестком диске. Наиболее вероятная причина состоит в том, что разрядилась батарея памяти CMOS.	Перезапустите процедуру аппаратной установки *setup* (или процедуру выбора программируемых опций на PS/2). Конечно, замена батареи более надежна.
Информация о	Если дисковый размер	Используйте fdisk, чтобы вновь

Проблема:	**Вероятная причина:**	**Действия:**
дефектном разделе.	правильно определен утилитой fdisk, но информация о разделе некорректна, то это означает, что данные в блоке 1 физического диска были повреждены.	создать правильную информацию о разделе. Рекомендуется записывать или распечатывать правильную информации о разделе на случай, если вам когда-либо придется выполнять этот шаг.

Шаг 4 - Монтируйте раздел и файловую систему

Вы проверили, что аппаратные средства работают (по крайней мере для блока 1) и что для QNX определен правильный раздел. Теперь следует создать блок-ориентированный файл непосредственно для раздела QNX и монтировать блок-ориентированный файл как файловую систему QNX:

```
mount -p /dev/hd0 /dev/hd0t77 /hd
```

Эта команда должна создать том, называемый /dev/hd0t77. В зависимости от состояния раздела QNX, mount может или не может потерпеть неудачу. Если информация раздела правильная, проблем не должно возникать. Поскольку корень (/) уже существует (на дискете или на удаленном диске на сети), то мы монтируем локальный раздел жесткого диска как файловую систему с именем /hd.

Теперь ваша цель будет состоять в том, чтобы выполнить утилиту chkfsys, чтобы исследовать - и, возможно, исправить - файловую систему.

Примечание. Если вы загрузились с дискеты и не знаете, есть ли какое-либо повреждения файловой системы на жестком диске (например, система не в состоянии загрузиться из-за простой ошибки в файле загрузки или файле инициализации системы), то теперь можно изменить корневой префикс на раздел жесткого диска следующей командой, что возобновит нормальную работу системы:

```
/hd/bin/prefix -R /=/hd/
```

Если выполняется эта команда, можно пропустить остальную часть этого раздела.

Если команда mount не выполняется...

Если команда mount не выполняется, то, вероятно, повреждена первая часть раздела QNX (так как Fsys не будет монтировать поврежденную, по его мнению, файловую систему).

В этом случае вы можете использовать утилиту dinit для записи достаточного количества хорошей информации на диск, чтобы удовлетворить Fsys:

```
dinit -hr /dev/hd0t77
```

Опция **-r** сообщает утилите dinit, чтобы она перезаписала:

- корневой блок;
- битовую карту (с распределением *всех* блоков);
- постоянные части корневого каталога.

Вам следует теперь повторно выполнить команду mount и еще раз пробовать создать точку монтирования для файловой системы QNX с именем /hd.

После этого вы должны будете восстановить битовую карту с помощью утилиты chkfsys даже для хорошего раздела.

Шаг 5. Запуск chkfsys

По крайней мере часть вашей файловой системы QNX теперь должна быть доступна. Вы можете использовать утилиту chkfsys, чтобы проверить файловую систему и восстановить, по возможности, большую часть данных.

Если жесткий диск смонтирован как /hd (например, компьютер загружается с дискеты), то введите:

```
/hd/bin/chkfsys /hd
```

Если жесткий диск смонтирован, как / (например, сетевая начальная загрузка), то введите:

```
chkfsys /
```

В каждом случае вам следует взять на заметку все возникающие проблемы и позволить утилите chkfsys исправить все, что возможно. Ваши дальнейшие действия зависят от результата выполнения утилиты chkfsys.

Если неисправен диск

Если по некоторой причине ваш диск полностью неисправен, читайте следующий раздел "**Восстановление потерянных файлов и каталогов**". В некоторых случаях вам, вероятно, придется вновь установить QNX и восстановить ваш диск с ваших резервных копий файлов.

Если значительная часть файловой системы повреждена или потеряны важные файлы, то восстановление с резервных копий может быть наилучшей альтернативой.

Если файловая система не повреждена

Если ваша файловая система не повреждена, однако компьютер еще отказывается загружаться с жесткого диска, то, вероятно, повреждено что-либо из следующего:

- загрузчик раздела в физическом блоке 1;
- загрузчик QNX в первом блоке раздела QNX.

Чтобы перезаписать загрузчик раздела для диска, используйте fdisk:

```
fdisk /dev/hd0 loader
```

Чтобы перезаписать загрузчик QNX, используйте dinit:

```
dinit -b /dev/hd0t77
```

Теперь вы сможете загрузить вашу систему.

Восстановление потерянных файлов и каталогов

Иногда вы можете обнаружить, что файлы или каталоги были полностью потеряны вследствие повреждения диска. Если после выполнения утилиты chkfsys вы узнаете, что определенные ключевые файлы или каталоги *не* были восстановлены, то вы можете использовать утилиту spatch, чтобы восстановить некоторые или все данные.

Перед этой попыткой вы должны сначала ознакомиться с организацией файловой системы QNX (см. в этой главе "**Краткий обзор структуры диска QNX**"). Следует также изучить описание утилиты spatch в книге ***Описание утилит***.

Приложение 1. Консоль и клавиатурные соглашения QNX

Это приложение охватывает следующие темы:

- Ввод строковых данных
- Повторение команд
- Переключение виртуальных консолей
- Использование нескольких консолей
- Изменение консольных шрифтов
- Приостановка и продолжение вывода данных
- Прекращение процесса
- Вызов системного отладчика
- Перезагрузка
- Национальные клавиатуры
- Клавиатура с одного взгляда

Это приложение описывает стандартную консоль QNX и клавиатурные соглашения *в текстовом режиме*. Для получения дополнительной информации относительно графической среды Photon смотрите в книге ***Руководство пользователя*** Photon.

Примите к сведению, что некоторые клавиши могут повести себя не так, как здесь описано; это зависит от того, как сконфигурирована ваша система.

Для получения подробной информации об утилитах, управляющих консолями QNX, смотрите в книге ***Описание утилит*** следующее:

- Dev;
- Dev.ansi;
- Dev.con;
- sh;
- stty.

Ввод строковых данных

Клавиши редактирования строк

Многие приложения выполняются в режиме *редактирования*. Если приложение выполняется в этом режиме, можно использовать следующие клавиши для ввода данных в виде строк:

Если хотите:	**Нажмите клавишу:**
Переместить курсор влево на одну	<-- (клавиша со стрелкой влево)

Если хотите:	**Нажмите клавишу:**
позицию	
Переместить курсор вправо на одну позицию	--> (клавиша со стрелкой вправо)
Переместить курсор в начало строки	**Home**
Переместить курсор в конец строки	**End**
Удалить символ слева от позиции курсора	**Backspace** или **Rubout** или <-- (клавиша со стрелкой). Заметьте, что при нажатии этой клавиши генерируется шестнадцатеричный код 7F (ASCII Rubout), а не 08.
Удалить символ в текущей позиции курсора	**Del**
Удалить все символы в текущей строке	**Ctrl-U**
Переключать между режимом вставки и режимом замены (по умолчанию - режим вставки)	**Ins** Заметьте, что если вы находитесь в режиме замены, то после ввода строки будете возвращены в режим вставки.
Ввести строку	**Enter**

Примечание. Командный интерпретатор QNX имеет дополнительные команды редактирования входных данных. Для получения дополнительной информации смотрите утилиту sh в книге ***Описание утилит***.

Обратите внимание, что клавиатура может работать не так, как указано, если:

- вы работаете с приложением, которое предъявляет сложные требования к диалогу с пользователем - приложение может управлять работой клавиатуры;
- вы работаете на подключенном терминале - терминал может иметь клавиатурные ограничения.

Максимальная длина входной строки

Максимальная длина входной строки - 256 символов. Приложения могут снижать этот предел.

Ввод длинных входных строк

Если вводится одна строка, превышающая размеры экрана, то она отображается на экране, как несколько экранных строк. Строка будет восприниматься как единственная входная строка в том случае, если используются стандартные клавиши редактирования строк.

Для получения дополнительной информации о режиме редактирования смотрите в книге *Системная архитектура* главу "Менеджер устройств".

Повторение команд

Командный интерпретатор позволяет повторно вызывать команды, которые были введены раньше, и затем вновь выполнять их. Эти команды сохраняются командным интерпретатором в буфере.

Если хотите переместиться в буфере:	**Нажмите эту клавишу:**
Назад	/\ (клавиша со стрелкой вверх)
Вперед	\/ (клавиша со стрелкой вниз)

Переключение виртуальных консолей

Адаптер дисплея, экран и системная клавиатура в целом представляют собой *консоль*. Чтобы взаимодействовать с несколькими приложениями одновременно, QNX позволяет войти в систему с так называемых *виртуальных консолей*. Эти виртуальные консоли обычно именуются /dev/con1, /dev/con2 и т.д..

На каждой виртуальной консоли может выполняться свое приложение, которое использует весь экран. Клавиатура подключается к виртуальной консоли, которая в настоящий момент видима. Можно переключиться с одной виртуальной консоли на другую и, таким образом, из одной программы на другую, вводя следующую комбинацию клавиш:

Если хотите увидеть:	**Нажмите:**
Следующую активную консоль	**Ctrl-Alt-Enter** *или* **Ctrl-Alt-+** (клавиша "плюс" вспомогательной клавиатуры)
Предыдущую активную консоль	**Ctrl-Alt--** (клавиша "минус" вспомогательной клавиатуры)

Можно также "прыгнуть" на определенную консоль, используя комбинацию клавиш **Ctrl-Alt-*n***, где *n* - цифра, которая соответствует номеру виртуальной консоли.

Если хотите увидеть:	**Нажмите:**
/dev/con1	**Ctrl-Alt-1**
/dev/con2 (если имеется)	**Ctrl-Alt-2**
...	...
/dev/con10 (если имеется)	**Ctrl-Alt-0**

Вы можете заблокировать переключение клавиатуры на консоли, введя команду stty +noswitch.

Для получения дополнительной информации о консолях QNX смотрите в книге *Системная архитектура* главу "Менеджер устройств".

Использование нескольких консолей

Системный администратор может задать количество виртуальных консолей, которые будут поддерживаться в машине, указав при запуске драйвера консолей Dev.con соответствующий аргумент.

Администратор может также определить программу (если такая есть), которая первоначально запускается на каждой консоли. По умолчанию утилита инициализации терминала (tinit) запускает login только на первой консоли, но "готова" запустить login на любой другой консоли, на которой вы нажмете клавишу. Это означает, что в то время как консоль 1 постоянно доступна, любая другая консоль не будет использоваться до тех пор, пока вы специально не переключитесь на нее и не нажмете клавишу.

Чтобы запустить login на неиспользуемой консоли:

Переключитесь на неиспользуемую консоль клавишами **Ctrl-Alt-*n***;
Нажмите любую клавишу, и login будет запущена.
Теперь можно получить доступ к консоли через любую из комбинаций клавиш циклического переключения консолей (например, **Ctrl-Alt-+**), описанных в предыдущем разделе.

Когда вы завершаете сеанс, печатая logout или exit, или нажимая **Ctrl-D**, консоль снова будет незанятой. Она теперь не появится, если использовать любую из комбинаций клавиш циклического переключения консоли. Исключением является консоль 1, на которой система обычно перезапускает login.

Изменение консольных шрифтов

В зависимости от видеоаппаратных средств, драйвер консоли (Dev.con) может использовать различные экранные шрифты. Имеющиеся в распоряжении шрифты пронумерованы от 0 до *n*. Когда QNX загружается, он устанавливает шрифт 0, как текстовый шрифт 25x80. Если используется видеоадаптер EGA или VGA, QNX определяет шрифт 1 как текстовый шрифт 43x80 (EGA) или текстовый шрифт 50x80 (VGA).

Чтобы информировать Dev.con о новых шрифтах или переопределить существующие шрифты, используйте утилиту cfont. Можно использовать эту утилиту, чтобы обеспечить работу со шрифтами различных размеров или шрифтами, которые содержат дополнительные наборы символов.

Для того чтобы изменить шрифт на текущей консоли, можно использовать следующие комбинации клавиш:

Если хотите выбрать:	**Нажмите:**
Следующий шрифт (до *n*)	**Ctrl-Alt->**
Предыдущий шрифт (до 0)	**Ctrl-Alt-<**

QNX отслеживает шрифты, используемые каждой консолью. Все консоли первоначально отображают шрифт 0. Вы можете заблокировать изменение шрифта с клавиатуры командой stty +noresize.

Задержка и продолжение вывода данных

Если хотите:	**Нажмите:**
Приостановить отображение вывода данных	**Ctrl-S**
Продолжить отображение вывода данных	**Ctrl-Q**

Прекращение процесса

Если необходимо прекратить процесс, протекающий в настоящий момент на консоли, нажмите **Ctrl-C** или **Ctrl-Break**. Система попытается прекратить процесс.

Вызов системного отладчика

QNX снабжен системным отладчиком низкого уровня, который позволяет устанавливать контрольные точки в программе, отображать и редактировать память, дисассемблировать код и проверять порты ввода/вывода. Если такой отладчик был включен в вашу операционную систему, его можно вызвать нажатием следующей комбинации клавиш:

Ctrl-Alt-Esc

Эту комбинацию клавиш можно заблокировать командой `stty +nodebug`.

Примечание. Не используйте этот отладчик в многопользовательской среде, так как он блокирует прерывания и замораживает всю систему - он предназначен только для отладки системы на низком уровне. Для получения дополнительной информации об этом отладчике смотрите Debugger в *Описании утилит* QNX. Информацию об обычном отладчике смотрите в книге *Руководство пользователя* по Отладчику Watcom.

Перезагрузка

Чтобы перезагрузить компьютер, используйте комбинацию клавиш:

Ctrl-Alt-Shift-Del

Эту комбинацию клавиш можно заблокировать командой `stty +noboot`.

Примечание. Перед вводом этой команды убедитесь, что на компьютере нет программ или утилит, находящихся в работе. В противном случае, некоторые файлы могут остаться открытыми. Кроме того, если вы производите перезагрузку в момент, когда проводится

критическая коррекция данных, возможно, что файловой системе потребуется восстановление (смотрите в книге *Системная архитектура* раздел "Устойчивость к ошибкам файловой системы" и главу "Восстановление файлов и дисков" этого руководства).

Национальные клавиатуры

Некоторые клавиатуры, например, французские и немецкие, используют клавиши, которые сами по себе не генерируют символы. QNX рассматривает эти клавиши, как "вспомогательные". Нажатие вспомогательной клавиши и следом второй клавиши модифицирует вторую клавишу, создавая составной символ. Например, для того, чтобы создать символ ü (символ U с умлаутом), нажмите, ", затем - **Shift-U**.

Такая обработка вспомогательных клавиш обеспечивает операторам знакомый метод ввода символов.

Заметьте, что вы можете также сгенерировать составные символы следующим образом:

1. нажать и отпустить клавишу **Alt**;
2. набрать на клавиатуре два символа.

Клавиатура с одного взгляда

Если хотите:	**Нажмите:**
Переместить курсор влево	<--
Переместить курсор вправо	-->
Переместить курсор в начало строки	**Home**
Переместить курсор в конец строки	**End**
Удалить символ слева от курсора	**Backspace** или <-- (на цифровой/ "серой" клавиатуре)
Удалить символ в позиции курсора	**Del**
Удалить все символы в строке	**Ctrl-U**
Переключаться между режимами вставки/замены	**Ins**
Ввести строку	**Enter**
Повторно вызвать команду	/\ или \/
Переключиться на следующую виртуальную консоль	**Ctrl-Alt-Enter** *или* **Ctrl-Alt-+**
Переключиться на предыдущую виртуальную консоль	**Ctrl-Alt--**
Переключиться на определенную виртуальную консоль	**Ctrl-Alt-*n***
Выбрать следующий шрифт	**Ctrl-Alt->**
Выбрать предыдущий шрифт	**Ctrl-Alt-<**
Приостановить вывод данных на экран	**Ctrl-S**
Возобновить вывод данных на экран	**Ctrl-Q**

Если хотите:	**Нажмите:**
Попытаться прекратить процесс	**Ctrl-C** или **Ctrl-Break**
Указать конец ввода данных (EOF)	**Ctrl-D**
Вызвать системный отладчик	**Ctrl-Alt-Esc**
Перезагрузить компьютер	**Ctrl-Alt-Shift-Del**

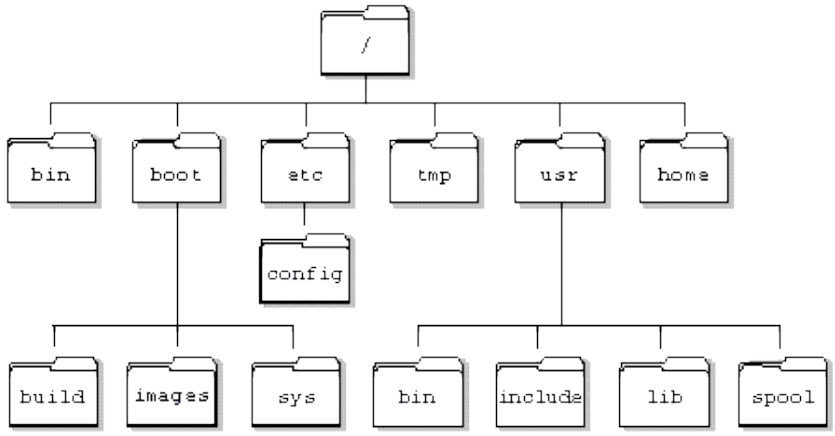
Приложение 2. Где находятся файлы QNX

Это приложение охватывает следующие темы:

- Типичная структура каталога
- Краткий обзор расположения файлов

Типичная структура каталога

Модули QNX хранятся в зависимости от вида услуг, которые они предоставляют. Следующая диаграмма дает краткий обзор общей структуры каталога при типичной установке QNX:



Краткий обзор расположения файлов

Чтобы найти:	**Смотрите:**
Системные исполняемые файлы	/bin
Makefile образа ОС	/boot
Файлы построения для создания образов (их читает утилита make)	/boot/build
Файлы образа ОС	/boot/images
Системные процессы, необходимые при начальной загрузке	/boot/sys
Файлы инициализации и другие файлы	/etc
sysinit и файлы конфигурации	/etc/config
Файлы информации о версии программного обеспечения	/etc/readme
Технические замечания	/etc/readme/technotes
Принятое по умолчанию местоположение для временных файлов	/tmp
Другие исполняемые файлы	/usr/bin

Чтобы найти:	**Смотрите:**
Заголовочные (*.h) файлы для Си-компилятора	/usr/include
Библиотеки для Си-компилятора	/usr/lib
Файлы характеристик терминала	/usr/lib/terminfo
Рабочие файлы для системных спулеров и очередей	/usr/spool
Исходный (домашний) каталог пользователя	/home/*имя_пользователя*