PERLE Specialix

Release Note

RIO for Linux v2.2.x

Release v1.0

5600038-12 Specialix International Limited December 1999

1. Introduction

This release note covers the installation and configuration of the RIO Driver for Linux. The driver was developed, written and tested on Linux kernel version 2.2.13 and above. It will install on versions 2.2.11 and later, but has not been tested on them.

Before proceeding, the User should make sure that both a kernel patch file and utilities RPM are available. These can be found either on the HandyWEB CDROM (in the drivers/rio/linux directory) or from the Specialix website (http://www.perlespecialix.com).

2. Driver Installation

The kernel patch file is named: **rio.patch-<driver vers>-<kernel vers>.gz**

e.g. **rio.patch-13-2.2.14pre14.gz**

Copy this file to a temporary directory and uncompress it. The patch can then be applied to the Linux kernel as follows:

```
cd /usr/src/linux <cr>
patch -p1 </tmp/rio.patch-15-2.2.14pre14 <cr>
```

If the patch is successful the driver will need to be enabled using the **make config** or **make xconfig** kernel utility. The RIO driver appears in the "Character devices" section and is labeled "Specialix RIO system support". Set this to 'm' for modules.

Note: You will need to enable "Non-standard serial port support" in order to select the RIO Driver.

Now rebuild the kernel and modules.

Note: Please consult your Linux distribution documentation on how to rebuild and install a new kernel and modules.

3. Hardware Installation

The RIO driver for Linux supports up to 4 host cards of either ISA or PCI bus types. Under most circumstances the driver will not need to be configured to recognise cards of either type.

Once the host card has been installed and the machine has been powered on the driver will automatically find any PCI cards installed in the system and register the correct interrupt that has been assigned by the BIOS. This is a feature of PCI Plug & Play.

In order to determine if any ISA cards are present, the driver will search for RIO cards at 3 *well known* ISA bus addresses (**0xC0000, 0xD0000, 0xE0000**). It is recommended that you leave your RIO ISA card at the factory default of 0xD0000, as this is the most common free ISA address slot on the majority of machines. If, for some reason your ISA card cannot be left at one of these addresses then you will need to modify the sources of the driver and recompile the rio

module for inclusion into your system. You will need to modify the following line in the **/usr/src/linux/drivers/char/rio/rio_linux.c** file:

```
int rio_probe_addrs[]= {0xc0000, 0xd0000, 0xe0000};
```

Change one of the addresses to the value you have set your ISA card rotary switches to e.g. If you have set your rotary switches to F, F (reading from left to right as you look at the switches with the ISA bus edge connector closest to you). Then you will need to set the address line to:

```
int rio_probe_addrs[]= {0xc0000, 0xd0000, 0xff0000};
```

Once you have made this change you will need to recompile the kernel modules and install the new module on your system.

Note: Please consult your Linux distribution documentation on how to rebuild and install a new kernel and modules.

RIO ISA card types are always set to polled mode i.e. they do not use up scarce interrupt request line resources. This does not adversely affect the performance of the card or the operating system.

4. Utilities Installation

The utilities file is named: **specialix_riotools-<maj ver>-<min ver>.i386.rpm**

e.g. **specialix_riotools-4-1.i386.rpm**

Copy this file to a temporary directory and install the utilities using the following command:

```
rpm -i specialix_riotools-4-1.i386.rpm
```

The utilities are now installed in the **/usr/sbin** directory on your machine.

The RIO configuration files are in the **/etc/rio** directory. A sample RIO configuration file (rio.cf.sample) is installed in this directory for you to use as a template.

The files rta.bin and host.bin should not be touched. These files contain the device firmware for the host card and the RTA and are loaded into the driver when the rioboot command is run.

Note: The FTP site also contains a RPM for the sources to the utilities. These sources are available under the GPL license.

5. Quick Setup

After you have installed the new kernel, modules & hardware, and have rebooted the machine. The following procedure can be followed to create devices for the attached ports.

1. Load the RIO driver module using **`modprobe rio`**.

2. Verify the module has successfully loaded using the **`lsmod`** command. The output should contain references to the rio module.

3. Ensure all your RIO devices are attached and powered on.

4. Upload the host card firmware to the driver and boot the attached devices (ignore any warning or error messages): **`rioboot -f`**

5. Create a configuration file (/etc/rio/rio.cf) based on what is attached to your system:
   `rioboot -w > /etc/rio/rio.cf.new`

   ```
   HOST:d100000f::boot:
       RTA:9400073d: UNKNOWN RTA 1-1:-1:boot:
       RTA:94000731: UNKNOWN RTA 1-2:-1:boot:
       RTA:94000732: UNKNOWN RTA 1-3:-1:boot:
       RTA:94000733: UNKNOWN RTA 1-4:-1:boot:
   ```

6. Modify the new configuration file (/etc/rio/rio.cf.new) to give your RTA's names and assign ports to them. Then rename the file as /etc/rio/rio.cf.

   ```
   HOST:d100000f:RIO PCI 1:boot:
       RTA:9400073d:RTA16 Ports 0-15:0:boot:
       RTA:94000731:RTA16 Ports 16-31:16:boot:
       RTA:94000732:RTA16 Ports 32-47:32:boot:
       RTA:94000733:RTA16 Ports 48-63:48:boot:
   ```

7. Upload the changes to the driver: **`rioboot -u`**

8. Create device nodes for the attached RTA's: **`riomkdev`**

9. Your system now has a number of device nodes (/dev/ttySR*) relating to the ports on the attached RTA's.

Note: The above rio.cf file displays are examples only. Files generated on your system will contain different information.

6. Adding ports to the System

More ports, in the form of extra RTA devices, can be easily added to a live system following the guidelines below.

1. Attach the new RTA to the RIO network via its link cable and power the unit on.

2. Wait until the unit has booted. This can be determined by the fact that the "RUN" and "CON" LEDs will be on.

3. Write out the current device driver configuration: **`rioboot -w >/etc/rio/rio.cf.new`**

4. Modify the configuration file to give the new RTA (the one named "UNKNOWN RTA….") a name and allocate a starting port number to the device:

   Before modification:

```
HOST:d100000f:RIO PCI 1:boot:
    RTA:9400073d:RTA16 Ports 0-15:0:boot:
    RTA:94000731:RTA16 Ports 16-31:16:boot:
    RTA:94000732:UNKNOWN RTA 1-3:-1:boot:
```

   After modification:

```
HOST:d100000f:RIO PCI 1:boot:
    RTA:9400073d:RTA16 Ports 0-15:0:boot:
    RTA:94000731:RTA16 Ports 16-31:16:boot:
    RTA:94000732:RTA16 Ports 32-47:32:boot:
```

5. Syntax check the new configuration file: **`rioboot -s -c /etc/rio/rio.cf.new`**

6. Fix any errors indicated by the syntax check and then rename the configuration file to **`/etc/rio/rio.cf`**

7. Upload the changes to the driver: **`rioboot -u`**

8. Create device nodes for the attached RTA's: **`riomkdev`**

9. Your system now has a number of new device nodes (/dev/ttySR*) relating to the ports on the new RTA.

10. Perform a check to determine the port is present and available: stty -a </dev/ttySR32. The port name can be determined by the RTA offset in the rio.cf file added to the port offset on the RTA i.e. the 4 port on RTA "RTA16 Ports 16-31" is 16 + 4 or /dev/ttySR20.

7. Removing ports from a System

Groups of ports can be easily removed from a live system following the guidelines below.

1. Determine the name and/or unique number of the RTA you wish to remove from the system. This can be done either by disconnecting the RTA link and getting its name from the console messages that occur when an RTA is disconnected from the network or by using the **`rioidentify`** command to determine which RTA/name needs to be removed.

2. Power off and detach an RTA from the RIO network.

3. Delete the RTA entry from the device driver: **`riodelete <unique number|"name">`**

4. Edit the **`/etc/rio/rio.cf`** file to remove the RTA you have just powered off. Remove the RTA line that contains the unique number and/or name of the RTA just deleted.

5. Syntax check the new configuration file and fix any errors: **`rioboot -s -c /etc/rio/rio.cf`**

6. Upload the changes to the driver: **`rioboot -u`**

7. Delete the device nodes from the /dev directory: **`riomkdev -d <unique number|"name">`**

8. Loading the rio module at System boot time

In order to make use of the RIO devices the driver module must be loaded into the Kernel and the firmware must be downloaded onto the host card and RTA devices. This can be done manually every time the machine boots or configured in to the start-up files to ensure the driver is always loaded and initialised when the machine boots.

To manually load the module and firmware follow the steps below:

1. Load the RIO module into the kernel: **`modprobe rio`**

2. Load the RIO firmware into the driver: **`rioboot`**

3. The RTA's should now be booting and connection messages should appear on the console.

To automatically configure the start up files to load the rio module and firmware, copy the following script to the file **/etc/rc.d/init.d/rio**.

```
#!/bin/sh
#
# rio           This shell script takes care of starting and
stopping
#               the RIO services.
#
# probe: true

# Source function library.
. /etc/rc.d/init.d/functions

#
# Exit if the rio utilities are not present
#
[ -f /usr/sbin/rioboot ] || exit 0

# See how we were called.
case "$1" in
  start)
        # Start driver and load module.
        action "Starting RIO services: " /sbin/modprobe rio
        echo -n "Starting RIO Firmware: "
        #
        # If we have a rio.cf file then just rioboot
        #
        if [ -f /etc/rio/rio.cf ]; then
                daemon /usr/sbin/rioboot
        else
                daemon /usr/sbin/rioboot -f
        fi
        echo
        ;;
  stop)
        # Stop driver and unload module.
        action "Shutting down RIO services: " /sbin/modprobe -r rio
        echo
        ;;
  status)
        status rio
        ;;
  restart|reload)
        $0 stop
        $0 start
        ;;
  *)
        echo "Usage: rio {start|stop|restart|reload|status}"
        exit 1
esac
```

```
exit 0
```

You will then need to create symbolic links to the file in the relevant system start-up directories e.g. If you machine normally boots into run level 5 you will need to create the following symbolic links:

```
ln -s /etc/rc.d/init.d/rio /etc/rc.d/rc5.d/S50rio
ln -s /etc/rc.d/init.d/rio /etc/rc.d/rc5.d/K50rio
```

These links will ensure that the rio start-up script S50rio is run each time the machine is booted into run level 5 and that the kill script K50rio is run whenever the machine is shutdown.

9. Configuration Utilities

The configuration file - **`/etc/rio/rio.cf`** – contains all the information the utilities may require to carry out configuration operations on the driver.

A default **`rio.cf`** file is installed with driver, but does not contain any host card or RTA configuration data. The default port entry (DEFPORT) is defined. Please read the following pages detailing the various fields in the **`rio.cf`** file first.

The driver will be booted at boot time by the command **`rioboot –f`**, which forces the driver to boot even if no configuration file is present. When the system comes up you should run :

```
rioboot -w
```

to get the configuration details of your RIO installation, redirect the output to a file, (not **`rio.cf`**). Add PORT configuration information, if all your ports on an RTA are to be configured the same way, you only need to add one PORT entry for each RTA.

An example **`rio.cf`** file from a system with 2 host cards and an RTA on each using the default configuration for all ports is shown on the next page. Note that the RTA names have been changed from the defaults given by the driver when booted. Setting up your **`rio.cf`** file like this will get you started.

Use

```
rioboot –c file –s
```

to syntax check your config file. When you are happy with the content of your config file, copy it to **`/etc/rio/rio.cf`** and run

```
rioboot –u
```

to update the driver with your configuration details. Then run

```
riomkdev -f
```

to make all the tty device nodes, (you may need the –f flag to force any existing RIO tty device nodes to be unlinked if you have installed on a system previously hosting a RIO set up).

Your RIO system is now ready for use.

Example rio.cf file :

```
#
# Current RIO driver configuration.
#
# File Created : Fri Feb 26 08:31:01 1999
#
#
# Host entry format is:
# HOST:<unique id>:<Device Name>:<boot/noboot>:
#
# Rta entry format is:
# RTA:<unique id>:<Device Name>:<First Port>:<boot/noboot>:
#
# Port entry format is:
# PORT:<offset>:<tty dev>:<xp dev>:<xcps>:<xpon>:<xpoff>:ixany,ixon,lock,store,d
rain:
#
# Default port format is:
# DEFPORT:<offset>:<dev prfx>:<xp dev prfx>:<tty/modem>:<xcps>:<xpon>:<xpoff>:ix
any,ixon,lock,store,drain:
#
#

DEFPORT:0:/dev/ttySR%03d::tty:100:XPON:XPOFF:ixany:


HOST:d100000f:HOST 1:boot:
    RTA:94000944:RTA 1:0:boot:
        PORT:0:::tty:::::

HOST:da00000e:HOST 2:boot:
    RTA:94001a0b:RTA 2:16:boot:
        PORT:0:::tty:::::
```

NAME

rio.cf - file contains RIO device driver configuration information

DESCRIPTION

/etc/rio/rio.cf is an administrative file that contains information used by the RIO utilities to configure the RIO device driver and device nodes.

The file contains lines that are either comments or begin with one of a pre-defined set of keywords. Each line contains fields separated by the '**:**' character. A line beginning with the '**#**' character is defined as a comment.

The **rio.cf** file contains lines of the following types:

HOST A line beginning with this mnemonic is used to define a host card entry. The line contains the following fields:

unique num This field contains the unique number identifying the host card in the system.

name A string defining the name associated with this host card.

bootflag This field contains the string **boot** or **noboot**. If the user sets this field to **noboot** then the device driver will not attempt to download the host card with its binary file. If this field is left blank then the default value is **boot**.

RTA A line beginning with this identifier is used to define an RTA device. The line contains the following fields:

unique num This field contains the unique number identifying the RTA in the system.

name This field contains a string defining the name associated with this RTA.

sysport This field contains the system port number to be assigned to the first port on the RTA. A system port number is any number from 0 to 504 and must be divisible by 8 leaving no remainder i.e. 0, 8, 16, 24 are all valid system port numbers, 43 is an invalid system port number. System port numbers are allocated in groups of 8 and a 16 port RTA consumes 2 consecutive system port groups. If this field is left blank then no system ports will be allocated to this device.

bootflag This field contains the string **boot** or **noboot**. If set to **noboot** then the RTA is not booted by the device driver and no ports on this system are assigned to it. The default for this field if left blank is **boot**.

DEFPORT A line beginning with this identifier is used to define the default values for the system ports. The line contains the following fields:

offset This field contains the numerical offset added to each system port when defining the default name for a device in the **riomkdev** command. The default for this field if left blank is **1**.

device prefix Not used on Linux.

xprint prefix Not used on Linux.

device type This field contains the string **tty** or **modem** and defines the type of default device that should be created. A device of type of **tty** is a normal device and ignores the state of any

modem signals during an open call and while the port is open. A device of type **modem** asserts the **DTR** output signal on an open and does not return from the open until the **DCD** input signal has been asserted. A **SIGHUP** signal is generated and sent to the controlling tty process when the **DCD** signal is de-asserted if the port is open. This is equivalent to the Linux callout device. If this field is left blank the default is **tty**.

xprint cps Not used on Linux.

xpoff string Not used on Linux.

xpon string Not used on Linux.

flags This field contains a combination of any of the following flags. Each flag must be separated by a '**,**' and the field must be ended with a '**:**'.

ixany Always open the port with the **termio** flag, *ixany* set.

ixon Always open the port with the **termio** flag, *ixon* set.

lock Lock the port settings. When this flag is set the RIO driver disallows any further attempts to configure a port using **termio**. See the **riolock** command for a more detailed description.

store Store the port settings across closes. When this flag is set the RIO driver keeps a copy of the port configuration and resets the port to this on each subsequent open. See the **riostore** command for a more detailed description.

PORT A line beginning with this identifier is used to define a unique set of port attributes to an individual port. This line must be preceded by a line of type **RTA** and the port referred to is assumed to be one of the ports connected to this RTA. If a port is multiply defined then the last entry in the configuration file will take precedence.

offset This field contains the numerical offset indicating which port on the previously specified RTA that the following attributes refer to. It is an error to leave this field blank.

device prefix This field contains a string uniquely identifying this port device. It is used by the **riomkdev** command to create or delete the device. If this field is left blank then the default tty device is created.

xprint prefix Not used on Linux.

device type This field contains string **tty** or **modem** and defines the type of tty device that should be created. See the **DEFPORT** description for more details.

xprint cps Not used on Linux.

xpoff string Not used on Linux.

xpon string Not used on Linux.

flags This field contains a combination of any of the flags **ixany**, **ixoff**, **lock** & **store**. See the **DEFPORT** description for more details. If this field is left blank then the default is assumed.

EXAMPLES

If you have a system with 2 host cards and 2 RTA's attached to each host the configuration file might look something like this:

```
#
# Example config file
#
DEFPORT:0:/dev/ttySR%d::tty::::ixany:

HOST:d11000011:First Host:boot:
    RTA:e30000d0:Software:0:boot:
        PORT:0:/dev/printer1::::::
    RTA:e30000d1:Hardware:8:boot:
        PORT:0:/dev/printer2::::::
HOST:c11000012:Second Host:boot:
    RTA:e30000d2:Marketing:16:boot:
        PORT:0:/dev/printer3::::::
    RTA:9400002d:Sales:24:boot:           # A 16 port RTA
        PORT:9:/dev/printer4::::::
```

The default setting for this site would produce tty devices starting at /dev/ttySR0 and ending at /dev/ttySR39. There are no modem devices in the system. There are 3 printers connected to port 0 of each of the 8 port RTA's and one printer connected to port 9 of the 16 port RTA. The printers have their own unique device names. The names of each of the RTA's are "Software", "Hardware", "Marketing" and "Sales".

If you have 3 modems connected to an RTA the configuration file might contain:

```
RTA:e30000d0:Software:0:boot:
    PORT:0:/dev/cusr0::modem:::
    PORT:1:/dev/cusr1::modem:::
    PORT:2:/dev/cusr2::modem:::
```

FILES

/etc/rio/rio.cf	RIO configuration file

NOTES

When the **rioadopt** command is used write this file, any comment lines in an existing file will be lost. The existing file will be moved to [filespec]**.old**.

SEE ALSO

rioboot, **riomkdev**, **rioreboot**, **rioshow, riolock, riounlock, riostore.**

NAME

rioboot - Initialise, update or save the RIO driver configuration.

SYNOPSIS

rioboot [-c *file*] [-h] [-v] [-s]

rioboot [-c *file*] [-h] [-v] -u

rioboot [-c *file*] [-h] [-f] [-v] -w

DESCRIPTION

During system start-up the RIO driver scans the hardware for available RIO host cards. It performs a comprehensive RAM test on all host cards that it finds and builds an internal table of the ones that pass their RAM test. The driver then exits its start-up routine leaving the host cards in a reset state.

Before any serial or transparent print ports can be accessed by the system the RIO driver must be initialised using the **rioboot** command. This command reads in the RIO configuration file and passes this information to the driver. When used to initialise the driver it performs three distinct functions:

1. Parses the RIO configuration file and initialises the driver with the host card and rta information found in this file. This includes the device unique number and name etc.
2. Reads in the RIO download code files and requests the driver download each of the host cards and any attached RTA's.
3. Parses the RIO configuration file and sets up the RIO ports with either the default settings or any custom settings defined within this file.

Once the **rioboot** command has initialised the RIO driver the attached serial and transparent print ports become accessible as long as the RTA they are connected to is booted.

The command **rioboot -u** can be used to update the driver with any changes that have been made to the configuration file after the driver has been initialised.

The command **rioboot -w** can be used to output the current configuration of the driver in the format of the configuration file. The output is the amalgamation of the current topology as supplied by the Driver and the current configuration file **(/etc/rio/rio.cf**). The output is sent to the standard output file descriptor and can be redirected using Shell commands. The command can be used to generate a new configuration file after adding a new host card and/or RTA's. If the **-c** option is used in conjunction with this option the file specified is the current configuration file.

Normal operation is for the command to execute silently and any errors are passed back through the exit code. The **-v** option, however, can be used to execute the command in *verbose* mode and any anomalies or errors will be reported to the standard error file descriptor.

The **-s** option can be used to perform a syntax check on the configuration file. This option is useful if you have made changes to the file and want to check that they are valid before committing them to the Driver. When this option is selected the **-v** option is forced on to give as much information as possible about the file syntax.

The **-c** option can be used to specify an alternate configuration file.

The **-f** option can be used to force the Driver to attempt to boot even if no configuration file exists.

The **-h** option prints a usage message.

DIAGNOSTICS

The exit codes for **rioboot** are the following:

0	successful completion of task
1	Failed to initialise driver
2	Failed to download device binaries
3	Failed to set-up ports
4	Failed to save configuration
5	Terminated by signal

EXAMPLES

You could use rioboot to replace a malfunctioning RTA, (normally done using rioadopt) :

- Power off the malfunctioning RTA.
- Add the new RTA and power it on.
- Once the RTA has booted type "**rioshow**" and make a note of the old and new RTA unique numbers.
- Edit the file **/etc/rio/rio.cf** and change the unique number entry corresponding to the old RTA to the new RTA's unique number.
- Execute **rioboot -u**

FILES

/etc/rio/rio.cf	RIO configuration file
/etc/rio/rta.bin	Host card download code
/etc/rio/host.bin	Eight port RTA download code

NOTES

Creation of a configuration file will not generate any custom port settings.

When the **rioboot -w** option is executed any blank fields in custom port definitions will be expanded.

SEE ALSO

rioreboot, **rioshow**, **riomkdev**

NAME

rioadopt - Replace a malfunctioning RTA and update configuration automatically.

SYNOPSIS

rioadopt [-c *file*] [-h] [-v] -r

rioadopt [-c *file*] [-h] [-v] -i

DESCRIPTION

Updates configuration file and driver when a new RTA replaces one that is no longer connected. System operation continues without requiring a reboot.

May be invoked in “interactive” mode if the user requires the option of committing or rejecting the “adoption”.

May be invoked in “report only” mode if required.

The current content of the configuration file is compared with the configuration reported by the driver. If the driver reports that only one RTA is missing and only one new one is present then, unless invoked in “report only” mode, the new RTA will be adopted automatically if it is connected to the same host card as the missing one. If invoked in interactive mode, missing and new RTA unique Ids are displayed and the user is required to confirm or otherwise that the adoption should be committed to the driver.

The following are illegal combinations for RTA adoption and will force “report only” mode :

- New RTA(s) detected when none reported missing - add new RTA(s) to rio.cf and use rioboot -u.
- Missing RTA(s) reported and no new one to adopt.
- Missing RTA(s) reported and more than one new RTA detected - edit rio.cf and use rioboot -u.

If more than one RTA is reported missing and one new RTA is detected then, unless “interactive” mode is specified, “report only” mode is forced. This enables rioadopt to be invoked at boot time without the possibility of the boot process halting because rioadopt is waiting for user input. A report will displayed and the system administrator should login and run rioadopt interactively to select the missing RTA which the new one should take the place of.

When an RTA adoption takes place the configuration file is automatically updated, the existing configuration file is moved to [filespec].old. Future system reboots will automatically configure the adopted RTA.

Once an RTA has been adopted it assumes the name of the one it replaced. Any ports configured on the old RTA retain their device names and node numbers and continue to function exactly as before the adoption took place.

The **-c** option can be used to specify an alternate configuration file - **for read and write**.

The **-r** option selects “report only” mode.

The **-i** option causes rioadopt to operate interactively.

The **-v** Verbose mode. Error messages and warnings are sent to the standard error file descriptor.

The **-h** option prints a usage message.

DIAGNOSTICS

The exit codes for **rioadopt** are the following:

0	successful completion of task
1	Failed to initialise driver
4	Failed to save configuration

EXAMPLES

To replace a malfunctioning RTA :

- Power off the malfunctioning RTA.
- Add the new RTA and power it on.
- Once the RTA has booted type **rioadopt**.

FILES

/etc/rio/rio.cf	RIO configuration file

NOTES

When the **rioadopt** command is used to a new write the file, any comment lines in an existing file will be lost. Comments may be retrieved from the existing file which will be moved to [filespec]**.old**.

SEE ALSO

rioboot, rioreboot, **rioshow**

NAME

rioreboot - Reboot an RTA

SYNOPSIS

rioreboot [-h] [-v] *name*|*unique num*

rioreboot [-v] -n

DESCRIPTION

The **rioreboot** command requests the RIO driver to issue a reboot command packet to the specified RTA. The RTA can be specified by either its *name* or its *unique number*.

The **rioshow** command can be used to display the current machine configuration and the device names and unique numbers.

The normal operation of this command is silent but the **-v** option can be used to give a more verbose operation. All messages are sent to the standard error file descriptor.

The command **rioreboot -n** can be used to reboot any *network interconnected* RTA's present on the network. Network interconnected RTA's are ones that have been booted by an alternate host card to the one they are currently connected. This command is normally run after swapping out a host card. When used in conjunction with the **-v** option the command will display information about its current progress. The command attempts to reboot all the currently interconnected RTA's and will then wait until they have re-appeared on the network. It will then search for any more interconnected RTA's and issue reboot commands to any found. This process is repeated until there are no more interconnected RTA's or until a number of re-tries is exceeded.

The **-h** option prints a usage message.

EXAMPLES

If you have an RTA named "Ports 0-7" and want to reboot it then type:

rioreboot "Ports 0-7"

If you have a number of RTA's you want to reboot and you know either their unique numbers or their names then type:

rioreboot e3000ccf e00037da "Ports 0-7" 9400001b

DIAGNOSTICS

The exit codes for **rioreboot** are the following:

0 successful completion of task
1 Failed to reboot RTA.

NOTES

Data loss may be experienced when rebooting an active RTA.

If a network interconnected RTA is connected to more than one host card issuing a reboot will result in the RTA being rebooted but still network interconnected.

SEE ALSO

rioboot, **rioshow**, **riomkdev**

NAME

rioshow - Display current driver device configuration information.

SYNOPSIS

rioshow [-s] [-h] [-v]

DESCRIPTION

The **rioshow** command displays information about the current RIO driver configuration. The output displays the devices attached to the RIO driver one line at a time giving information about their unique number, driver id, system ports and name.

The **-v** option causes the command to print error messages to the standard error file descriptor.

The **-s** option inhibits the display of the column headings.

The **-h** option prints a usage message.

DIAGNOSTICS

The exit codes for **rioshow** are the following:

0 successful completion of task

1 Failed to retrieve information from RIO device driver.

EXAMPLES

A typical output from the rioshow command executed without any arguments should be:

Unique Num	**ID**	**Ports**	**Name**
d1000011	**0**	**-1 - -1**	**PCI Host card**
940000ab	**1**	**0 - 7**	**Ports 0 - 15**
e3000ccf	**3**	**16 - 23**	**Ports 16-23**

NOTES

The device driver does not have to have been initialised using the **rioboot** command before attempting to get the host card information.

SEE ALSO

rioboot, **rioreboot**, **riomkdev**

NAME

riomkdev - Create or delete RIO tty and transparent print devices.

SYNOPSIS

riomkdev [-c *file*] [-f] [-h] [-r*name*|*unique num*] [-v]

riomkdev [-c *file*] [-f] [-h] [-r*name*|*unique num*] [-v] -x

riomkdev [-c *file*] [-f] [-h] [-r*name*|*unique num*] [-v] -d

DESCRIPTION

The command **riomkdev** should be used to create and delete RIO specific device nodes.

The command parses the RIO configuration file **/etc/rio/rio.cf** and looks for lines beginning with **RTA:**, **DEFPORT:** or **PORT:** (see **rio.cf**). These lines are then used to define the device node name and the devices are created or deleted depending on the command line options.

The device major number is determined using the RTA sysport field and the associated host card control device major number. The device minor number is determined by the RTA sysport field.

The device node name is determined either by a **PORT:** entry for the corresponding device or it is deduced from the device's sysport number and the port name string in the **DEFPORT:** line.

Options

-c*file*	Specify an alternative configuration file.
-d	Delete devices. All specified devices shall be deleted if they exist.
-f	Force operation. Any existing device nodes shall be unlinked before the new devices are created.
-h	Print usage message and exit.
-r*name*|*unique num*	Specify the specific RTA whose devices should be created/deleted.
-v	Verbose mode. Error messages and warnings shall be sent to the standard error file descriptor.
-x	Not used on Linux.

DIAGNOSTICS

The exit codes for **riomkdev** are the following:

0 successful completion of task
1 Failed to create/delete devices

FILES

/etc/rio/rio.cf	RIO configuration file

NOTES

The command makes no attempt to determine if a particular device node is currently open or in use before deleting it.

SEE ALSO

rioboot, **rioreboot**, **rioshow**

NAME

riodelete - Remove an RTA entry from the RIO driver tables.

SYNOPSIS

riodelete [-f] [-h] [-v] *name|unique num*

DESCRIPTION

The command **riodelete** requests the RIO driver to remove the specified RTA device from its internal tables. The RTA can be specified by either its *name* or its *unique number*.

The **rioshow** command can be used to display the current RIO driver configuration including the devices names and unique numbers.

If an RTA is currently attached and booted then the command will not delete it unless the user has specified the **-f** option. This option will force the deletion of the device by removing it from the network and then requesting the RIO driver to delete its entry. The RTA is removed from the network by placing the RTA software in an infinite loop.

The normal operation of this command is silent but the **-v** option can be used to give a more verbose operation. All messages are sent to the standard error file descriptor.

The **-h** option prints a usage message to the standard error file descriptor.

DIAGNOSTICS

The exit codes for **riodelete** are the following:

0	successful completion of task
1	Command was unsuccessful

EXAMPLES

To remove an RTA named “Ports 0-7” type:

riodelete “Ports 0-7”

To force the removal of the RTA with unique number e3000045, type:

riodelete -f e3000045

NOTES

This command removes the RTA device from the driver tables but does not delete its entry from the RIO configuration file **/etc/rio/rio.cf**.

SEE ALSO

rioreboot, **rioshow**, **riomkdev, rioidentify**

NAME

rioidentify - Display current driver version information.

SYNOPSIS

rioidentify [-h] [-v] name|unique num

DESCRIPTION

The **rioidentify** command causes an RTA, specified either by its name or unique number, to flash its link LEDs amber. The operation continues until such time as the user terminates the program by pressing <return> in response to an appropriate prompt.

The **-v** option causes the command to print error messages to the standard error file descriptor.

The **-h** option prints a usage message.

DIAGNOSTICS

The exit codes for **rioidentify** are the following:

0 successful completion of task

1 Failed to retrieve information from RIO device driver.

NOTES

This command does not affect normal operation.

NAME

rioversion - Display current driver version information.

SYNOPSIS

rioversion [-h] [-v]

DESCRIPTION

The **rioversion** command displays the current version of the RIO device.

The **-v** option causes the command to print error messages to the standard error file descriptor.

The **-h** option prints a usage message.

DIAGNOSTICS

The exit codes for **rioversion** are the following:

0 successful completion of task

1 Failed to retrieve information from RIO device driver.

NOTES

The device driver does not have to have been initialised using the **rioboot** command before attempting to get the version information.

NAME

riostats - Display statistics from the RIO device driver.

SYNOPSIS

riostats [-h] [-v] [-e] ttyname

riostats [-h] [-v] [-d] ttyname

DESCRIPTION

Retrieves and displays device driver statistics on a per port basis.

The driver does not update its internal data until statistics gathering is “enabled”.

The driver may be inhibited from updating statistics information by “disabling” statistics gathering.

By default the output has column header information. This may be switched off,. For example, the riostats command may be invoked in a loop from a script which displays statistics information for all ports, (where the header information is only required to be output for the first port – the other port’s statistics being displayed directly underneath).

The **-e** enables driver statistics gathering..

The **-d** disables driver statistics gathering..

The **-v** Verbose mode. Error messages and warning are sent to the standard error file descriptor.

The **-h** option prints a usage message.

DIAGNOSTICS

The exit codes for **riostats** are the following:

0 successful completion of task

1 ‘stats’ ioctl failed.

EXAMPLES

To enable and start statistics gathering for the device /dev/ttyr000 :

- Type, riostats –e /dev/ttyr000.

To see statistics information for the device /dev/ttyr000 :

- Type, riostats /dev/ttyr000.

To disable statistics gathering for the device /dev/ttyr000 :

- Type, riostats –d /dev/ttyr000.

NOTES

Invoking riostats with the –d option does not reset the driver’s internal statistics data.

10. Dual-Host Fail-Safe

This appendix gives an example of how to set up Dual-Host Fail-Safe and subsequently swap RTAs from MASTER host to SLAVE host control. The scenario is the most basic Dual-Host set-up possible – two systems, each with one host card, both connected to one RTA.

1. Install RIO on both systems, leave the RTA switched off and reboot both systems as required following an installation.

2. As described in Appendix C, set up the RIO configuration file - /etc/rio/rio.cf – on both systems.

3. On the system which is to be the 'slave', set the `'bootflag'` field, for the host card only, to 'noboot'. Run rioboot –u again on the slave system.

4. Switch on the RTA, it should establish connections on the master system only. The slave system will report 'HOST 1 (A) connected to another network'.

5. To test Dual-Host Fail-Safe, remove the link between the RTA and the Master system's host card.

6. On the slave system, change the `'bootflag'` field back to 'boot' and run `rioboot –u`.

7. To get the slave system to take over the RTA, run `rioreboot –n,` (note that this command may take a while to complete, especially if there are many tiers of RTAs).