

---

# Rivendell Interprocess Communication Protocol

Fred Gleason

## Table of Contents

Overview .....	1
Unprivileged Commands .....	1
Drop Connection .....	1
Send Password .....	2
Privileged Commands .....	2
Request User .....	2
Set User .....	2
RML Send .....	2
RML Echo .....	2
Get GPI Carts .....	3
Get GPO Carts .....	3
Get GPI States .....	3
Get GPO States .....	4
Get GPI Mask States .....	4
Get GPO Mask States .....	4
Reload GPI Table .....	5
Get OnAir Flag .....	5
Process Notification .....	5

## Overview

This defines the IP protocol used for communication between different modules of Rivendell and the **ripd(8)** daemon.

Connection to ripd is by means of a TCP SOCK\_STREAM connection to TCP port **5006**. The format of a message is as follows:

*cmd-code* [*arg*] [...]!

*cmd-code*                      A two letter command code, describing the generic action to be performed

*arg*                              Zero or more arguments, delimited by spaces or, if the last argument, by ! (see below)

!                                  The ASCII character 33, indicating the end of the command sequence.

## Unprivileged Commands

No authentication is required to execute these.

### Drop Connection

End the session and drop the TCP connection.

**DC!**

## Send Password

Send a password to the server for authentication.

**PW *passwd*!**

*passwd*            A password to be supplied before granting the client access.

**ripd**(8) will respond with PW +! or PW -!, indicating the success or failure of the authentication.

## Privileged Commands

A connection must be authenticated before these can be executed.

## Request User

Request the LOGIN\_NAME of the user currently logged in.

**RU!**

**ripd**(8) will respond with RU *user-name*!.

*user-name*    The LOGIN\_NAME of the user currently logged in.

## Set User

Login in a user.

**SU *user-name*!**

**ripd**(8) will respond with RU *user-name*!.

*user-name*    The LOGIN\_NAME of the user to log in.

## RML Send

Send an RML command to a specified host.

**MS *ip-addr echo rml*!**

*ip-addr*    IPv4 address of the destination, indotted-quad notation.

*echo*        1 = Request echo, 0 = Request no echo.

*rml*         The RML command to send.

## RML Echo

Echo an RML command to a specified host.

**ME *ip-addr echo rml*!**

*ip-addr* IPv4 address of the destination, indotted-quad notation.

*echo* 1 = Request echo, 0 = Request no echo.

*rml* The RML command to send.

## Get GPI Carts

Return current GPI line cart numbers.

**GC *matrix*!**

Request the list of macro carts currently assigned for *matrix*. The following record will be returned for each line in the matrix:

GC *matrix gpi-line off-cart-num on-cart-num*!

*matrix* The specified matrix number.

*gpi-line* The GPI line number.

*off-cart-num* The number for the cart to be activated when *gpi-line* transitions to an OFF state. 0 indicates that no cart is currently set.

*on-cart-num* The number for the cart to be activated when *gpi-line* transitions to an ON state. 0 indicates that no cart is currently set.

## Get GPO Carts

Return current GPO line cart numbers.

**GD *matrix*!**

Request the list of macro carts currently assigned for *matrix*. The following record will be returned for each line in the matrix:

GD *matrix gpo-line off-cart-num on-cart-num*!

*matrix* The specified matrix number.

*gpo-line* The GPO line number.

*off-cart-num* The number for the cart to be activated when *gpo-line* transitions to an OFF state. 0 indicates that no cart is currently set.

*on-cart-num* The number for the cart to be activated when *gpo-line* transitions to an ON state. 0 indicates that no cart is currently set.

## Get GPI States

Return current GPI states.

**GI *matrix*!**

Request the list of current GPI states for *matrix*. The following record will be returned for each line in the matrix:

*GI matrix gpi-line state mask!*

*matrix*      The specified matrix number.

*gpi-line*    The GPI line number.

*state*       1 = GPI is ON, 0 = GPI is OFF.

*mask*        1 = GPI is ENABLED, 0 = GPI is DISABLED.

## Get GPO States

Return current GPO states.

**GO *matrix*!**

Request the list of current GPO states for *matrix*. The following record will be returned for each line in the matrix:

*GI matrix gpo-line state mask!*

*matrix*      The specified matrix number.

*gpo-line*    The GPO line number.

*state*       1 = GPO is ON, 0 = GPO is OFF.

*mask*        1 = GPO is ENABLED, 0 = GPO is DISABLED.

## Get GPI Mask States

Return current GPI mask states.

**GM *matrix*!**

Request the list of current GPI mask states for *matrix*. The following record will be returned for each line in the matrix:

*GI matrix gpi-line mask!*

*matrix*      The specified matrix number.

*gpi-line*    The GPI line number.

*mask*        1 = GPI is ENABLED, 0 = GPI is DISABLED.

## Get GPO Mask States

Return current GPO mask states.

**GN *matrix*!**

Request the list of current GPO mask states for *matrix*. The following record will be returned for each line in the matrix:

*GN matrix gpo-line mask!*

*matrix*     The specified matrix number.

*gpo-line*   The GPO line number.

*mask*       1 = GPO is ENABLED, 0 = GPO is DISABLED.

## Reload GPI Table

Reload the GPI table in **ripd**(8).

**RG!**

## Get OnAir Flag

Request the state of the OnAir flag.

**TA!**

The following record will be returned:

*TAstate!*

*state*   1 = Active, 0 = Disabled.

## Process Notification

Send or Receive an Object Notification

**ON *notify*!**

Send the *notify* to all other active Rivendell modules.

### Note

See the Rivendell Notification Protocol API document for a description of the contents of *notify*.