# Daisy Chain feature

Before you familiarize yourself with the details of Daisy Chain, let us consider simple example.

Suppose you need to connect the radio modem embedded into your receiver and a device (PC, handheld, etc.) through which you are going to configure the modem. These devices are not directly connected but are both connected to your receiver via different serial ports. In this way, you are able to establish a communication link between the modem and a control device through the receiver using Daisy Chain mode. After you set up Daisy Chain between the two receiver's ports connected with the modem and the control device, respectively, your control device will be able to manage the modem. For working examples of how to set up a daisy chain configuration and then return to normal operation, see items A and B below.

In this section, the notation [port] is used to denote an input port of the receiver's, for example, *dev/ser/a*, *dev/modem/a* or *cur/term*.

You can make your TPS receiver redirect data from a port to an "output stream". We will call this receiver feature "data echoing". "Output stream" may be either a port (e.g. */dev/ser/a*) or the current log-file (*/cur/log*). Each receiver port has a set of parameters through which it can be controlled. One of these parameters is named */par/[port]/echo* (or, for short, "echo parameter"). A port's echo parameter defines the output stream into which you are going to redirect input data thus activating data echoing. Specifically, this parameter contains the name of a valid output stream (e.g. */dev/ser/a* or the string */dev/null* (the latter indicates that data echoing is disabled for the corresponding port). The default value is */dev/null* for all of the receiver's ports.

Whatever the value of */par/[port]/echo*, the receiver will handle input data on the port in accordance with its input mode. Recall that you select among the port input modes via the parameter */par/[port]/imode*. If you want to disable interpretation of input data on *[port]*, you should set the port's input mode to ***echo***.

After a port's input mode is switched to ***echo*** and the corresponding echo parameter is set to a value other than */dev/null*, incoming data will be redirected to the specified output stream without interpreting them.

If a port's input mode is switched to ***echo***, but the corresponding echo parameter is set to */dev/null*, then incoming data will still be interpreted as commands (note that these data are redirected nowhere in this case). This situation is equivalent to running this port in ***cmd*** mode.

There are two ways to disable data echoing and return to ***cmd*** mode.

The most straightforward, yet not always a feasible solution, is to reset the port's echo parameter back to */dev/null* by issuing an appropriate command via this or another port.

The other way is based on using a special sequence of characters ("echo-off sequence") that are sent to the port to disable data echoing.

When do we have to use the echo-off sequence to disable data echoing for *[port]*? We have to use this technique if

- */par/[port]/echo* ≠ */dev/null*
- */par/[port]/imode* = ***echo***
- none of the ports are running in command mode.

The receiver provides a “watchdog mechanism” that will force the echo parameter */par/[port]/echo* to */dev/null* as soon as the echo-off sequence is received and identified. Note that the input mode remains unchanged for *[port]*.

Bear in mind that the echo-off sequence will get to the output stream before the watchdog mechanism goes off. Suppose that the port you redirect data to is connected to a device that interprets the output stream somehow. What happens if the device mistakes the echo-off sequence (or part of it) for one of its own control sequences (commands)?

Obviously, this can cause problems. Therefore, the echo-off sequence must meet the following two requirements:

1) The echo-off sequence must be as distinctive from other possible character combinations on the port as possible. This will minimize the probability of “false alarm”.
2) Neither the echo-off sequence as a whole nor any part of it should be coincident with any of the external device's control sequences.

Every port has its own special sequence to turn off data echoing. This sequence is stored in the parameter */par/[port]/eoff*. The default value is #OFF# for all of the receiver's ports. This sequence is safe as long as you connect TPS receivers to each other (note that #OFF# will be interpreted as a comment for a pure TPS daisy chain configuration).

While data echoing is on, incoming data are redirected into an output stream unchanged and without delay. There are applications, however, where this is unacceptable. Suppose you have two or more ports concurrently redirecting their input data into the same output stream. Generally speaking, you may not be able to distinguish among data from different ports unless special measures are taken.

TPS receivers provide a special feature, called “wrapped echoing”, allowing you to effectively mix data from different sources without corrupting data integrity in the output stream. To activate “wrapped echoing”, set the parameter */par/[port]/ewrap* to “on”.

When “wrapped echoing” is turned on, the receiver packs individual characters on the port into a separate message in JPS format. Note that this message also contains the identifier of the corresponding port, which allows you to easily distinguish data from different sources in the output stream.

Once the new message is packed up, it is redirected to the output stream. How does the receiver know when to send off this message? The message is sent off as soon as either the message buffer is full or the time passed since some data was last received on the port exceeds the timeout threshold. Note that the message has the header [>>].

Examples.

A. Suppose you are going to use a control device (PC, hand-held, etc.) to handle the modem through your TPS receiver's ports. Assume that your modem is on port C whereas the control device is connected to another port. Also assume that the echo-off sequence should be set to QUIT for the current terminal.

To set up the required daisy chain configuration, run the following commands (for brevity, we omit here such commands as “set baud rate”, etc.):

| | |
|---|---|
| *set,dev/ser/c/echo,/cur/term* | Enables data echoing from port C to the current terminal |
| *set,dev/ser/c/imode,echo* | Disables input data interpretation on port C |
| *set,cur/term/eoff,QUIT* | Defines an echo-off sequence for the current terminal |
| *set,cur/term/echo,/dev/null* | Disables data echoing on the current terminal |

| | |
|---|---|
| *set,cur/term/imode,echo* | Disables input data interpretation on the current terminal |
| *set,cur/term/echo,/dev/ser/c* | Enables data echoing from the current terminal to port C |

The above settings will instruct the receiver to redirect all input from port C to the current terminal and to redirect input data from port C to the current terminal.

| | |
|---|---|
| *QUIT* | Turns off data echoing for the current terminal. Enable input data interpretation on the current terminal. |
| *set,cur/term/imode,cmd* | Returns the current terminal to command mode |
| *set,dev/ser/c/echo,/dev/null* | Disables data echoing on port C |
| *set,dev/ser/c/imode,cmd* | Returns port C to command mode |

B. Suppose you want to monitor commands sent to port B by using the current terminal. It can be done with the following command:
*set,dev/ser/b/echo,/cur/term*
To turn off data echoing for port B, run
*set,dev/ser/b/echo,/dev/null*