CG9103-2RS232-1USB
User Manual

Document Revision: 1.05
Document Date: 25 June 2016

WARRANTY:

BiPOM Electronics warrants CG9103-2RS232-1USB for a period of 3 years. If the board becomes defective during this period, BiPOM will at its option, replace or repair the board. This warranty is voided if the product is subjected to physical abuse or operated outside stated electrical limits. BiPOM Electronics will not be responsible for damage to any external devices connected to CG9103-2RS232-1USB. BiPOM Electronics disclaims all warranties express or implied warranties of merchantability and fitness for a particular purpose. In no event shall BiPOM Electronics be liable for any indirect, special, incidental or consequential damages in connection with or arising from the use of this product. BiPOM Electronics' liability is limited to the purchase price of this product.

Overview

CloudGate is an intelligent 3G or LTE M2M gateway and wireless Linux computer from Option®. CloudGate provides cell modem, LAN to WWAN routing and GPS functionality in a single basic unit for remote monitoring and control applications.

On top of the basic functionality, CloudGate can be tailored to meet specific industrial requirements by adding additional software and peripheral boards. CG9103-2RS232-1USB is one such peripheral board for CloudGate.

CG9103-2RS232-1USB is a miniature interface board, containing dual channel RS-232 line driver / receiver, and USB Host and OTG interface circuit. The combination of CG9103-2RS232-1USB and CloudGate offers unmatched connectivity to a wide variety of external devices. CG9103-2RS232-1USB is fully backed by a 3-year warranty, technical support and application assistance from BiPOM.

Specifications

- One 4-wire RS-232 port: TXD,RXD,CTS,RTS
- One 2-wire RS-232 port: TXD,RXD
- TIA / EIA –232-F compliant
- One USB Host interface with software controlled power management.
- One MicroSD socket with software controlled power management.
- 5VDC/1A power available to external devices
- Powers from CloudGate
- Low power consumption: Under 1mA when idle
- Protected against Electrostatic Discharge (ESD)
- FCC Certified

Target Applications:

- Oil and Gas Monitoring
- Irrigation Generator Set Monitoring
- Energy Monitoring
- Utility Meter Remote Monitoring
- Factory Automation
- Vibration Monitoring
- Sensor Networks
- Medical Devices
- Asset Tracking

Ordering Information

- CG9103-1USB (Single USB Host)
- CG9103-2RS232-1USB (Dual RS232 plus single USB Host)

Both models have microSD socket.

Board Layout

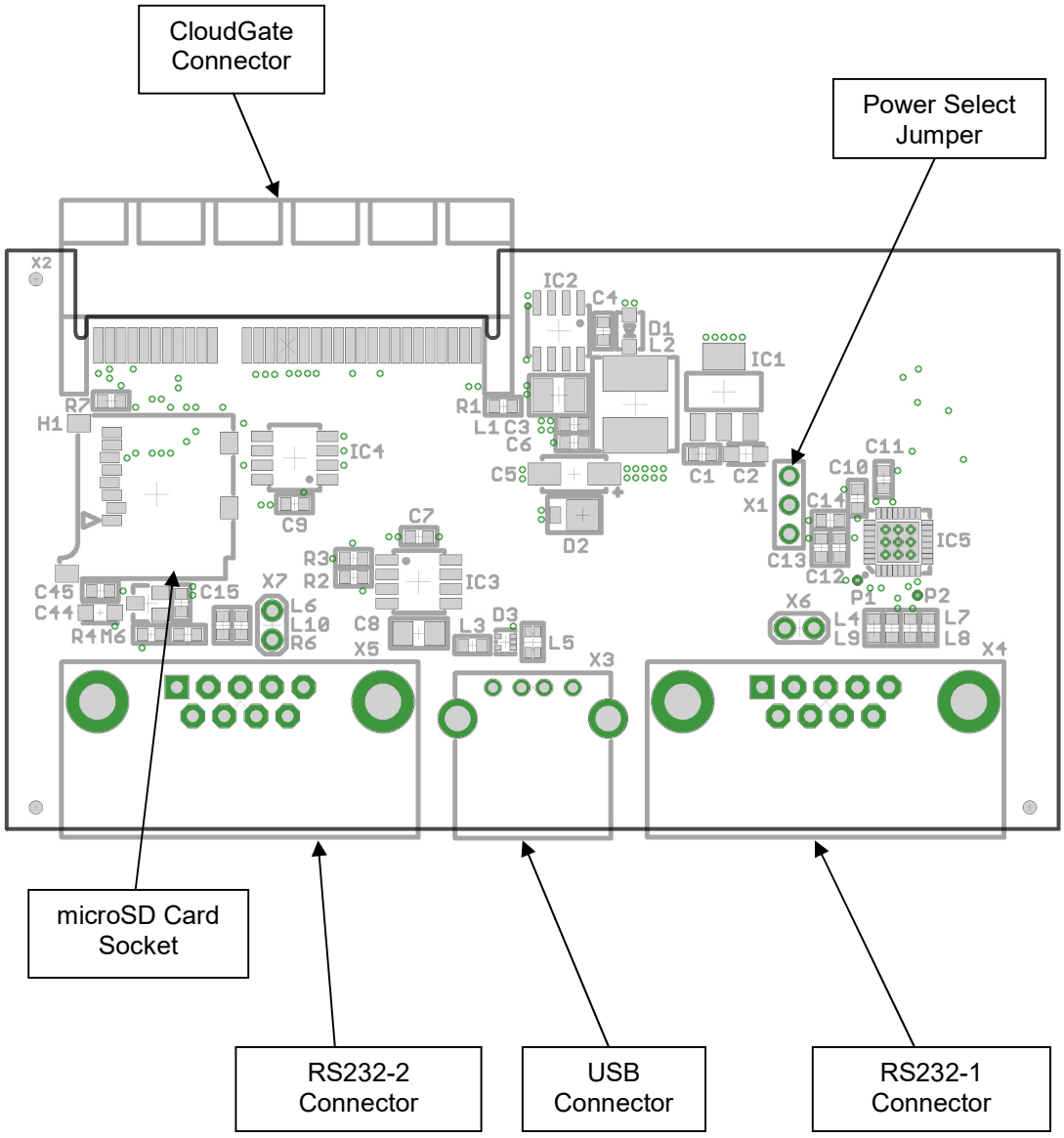


Figure 1

RS-232 ports and USB

The placement of USB and RS232 connectors at the front panel is shown on Figure 2:

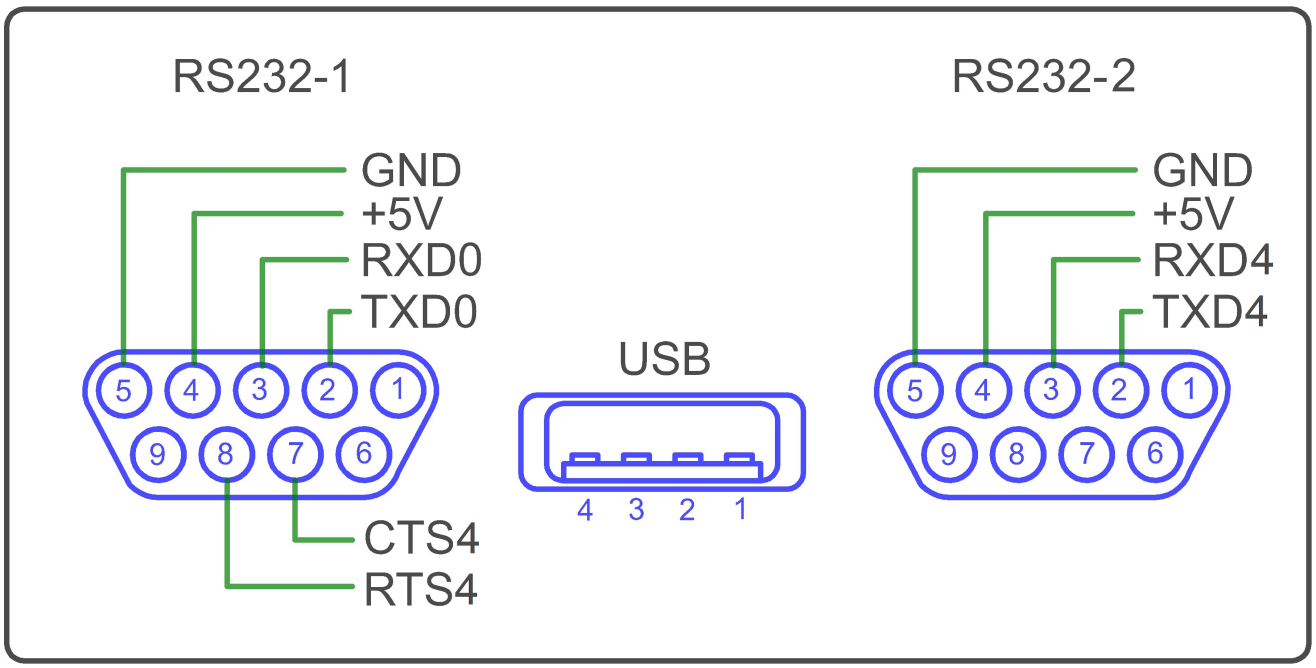


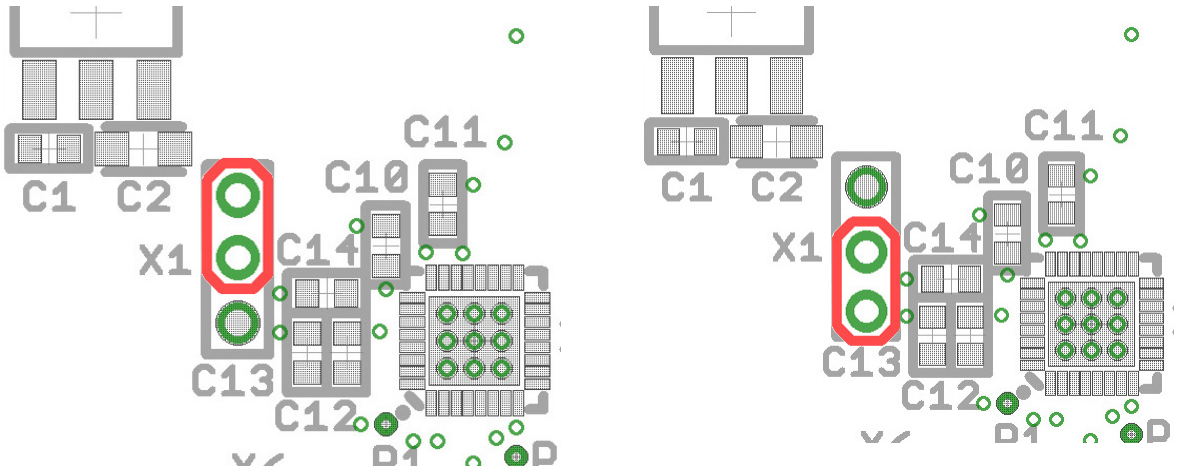
Figure 2

MicroSD Card Holder

CG9103-2RS232-1USB has a socket for a microSD card operation for data storage and other purposes. MicroSD card is accessible from CloudGate Linux.

Power Select Jumper

CG9103-2RS232-1USB can be powered either from its on-board power supply, or from CloudGate 3.4V. Figure 2 shows how Jumper X1 is used to switch between these two options:



Internal Power Supply

Power supply from CloudGate 3.4V

Figure 3

RS232 Power Jumpers

When installed, jumper X6 connects +5V power supply to Pin #4 (DTR) of RS232-0 connector.

When installed, jumper X7 connects +5V power supply to Pin #4 (DTR) of RS232-1 connector.

This feature can be used to power external RS232 devices from CloudGate, provided that the current consumption of the external RS232 device is sufficiently small (less than 1 Ampere combined on both RS232 ports).

Software

CG9103-2RS232-1USB can be accessed using the serial framework for CloudGate.

BiPOM also provides a test developer image that enables the Linux drivers for CG9103-2RS232-1USB. After installing the developer image through the CloudGate provisioning web interface:

1. From Linux command shell, microSD card can be mounted to /tmp/mmc using:

mkdir /tmp/mmc; mount /dev/mmcblk0 /tmp/mmc; cd /tmp/mmc

2. From Linux command shell, an external USB flash drive on USB port cane be mounted to /tmp/usb using:

mkdir /tmp/usb; mount /dev/sda /tmp/usb; cd /tmp/usb

3. A built-in terminal program allows RS232 loopback testing (without handshake lines) when a NULL modem cable is connected between the two RS232 connectors. Terminal can be started using:

For RS232-1 port:

/rom/mnt/cust/bin/terminal -d /dev/ttySP0

For RS232-2 port:

/rom/mnt/cust/bin/terminal -d /dev/ttySP4

4. A built-in terminal program allows RS232 loopback testing (using the handshake lines RTS and CTS on RS232-1 port) when a NULL modem cable is connected between the two RS232 connectors. Two separate sessions of the terminal program can be started in two separate Linux shell windows using:

/rom/mnt/cust/bin/terminal -d /dev/ttySP4

/rom/mnt/cust/bin/terminal_handshake -d /dev/ttySP0

If RTS-0 and CTS-0 are not connected, it is not possible to send data from SP0 to SP4.

SP4 can still send and SP0 will receive.

To send data from SP0 to SP4 necessary to connect RTS-0 and CTS-0 externally so handshake lines will be used to control RS232 traffic.

LuvitRED Support

First RS232 serial port is mapped to Linux serial port SP0 (**/dev/ttySP0)** and second RS232 serial port is mapped to Linux serial port SP4 (**/dev/ttySP4).**

RS232 ports can be configured using either the Basic Interface or Advanced Editor in LuvitRED.

Basic Interface

Go to the "Plugin" tab, under it one will find a sub-tab called "Serial and GPS settings" or "LuvitRED" (The name depends on the LuvitRED version being used):

Figure 4: Plugin tab, Serial and GPS settings.

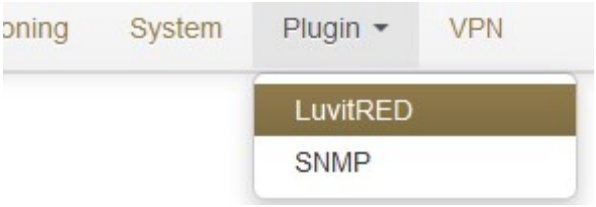


Figure 5: Plugin tab, LuvitRED

Without any configuration, the basic interface looks as follows:

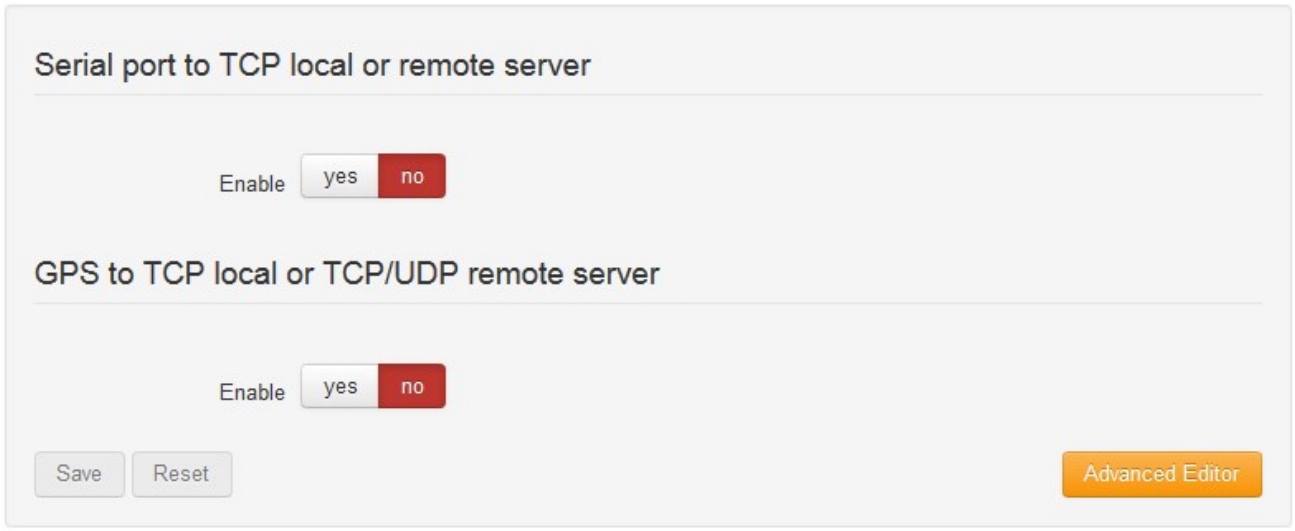


Figure 6: Basic Interface

First, we will focus on the section called "Serial port to TCP local or remote server". This section allows the configuration of a single serial port, the first RS232 (**SP0**), to be accessible remotely via a local TCP server running on the CloudGate (Figure 7) or a remote TCP server, running at another location (Figure 8).

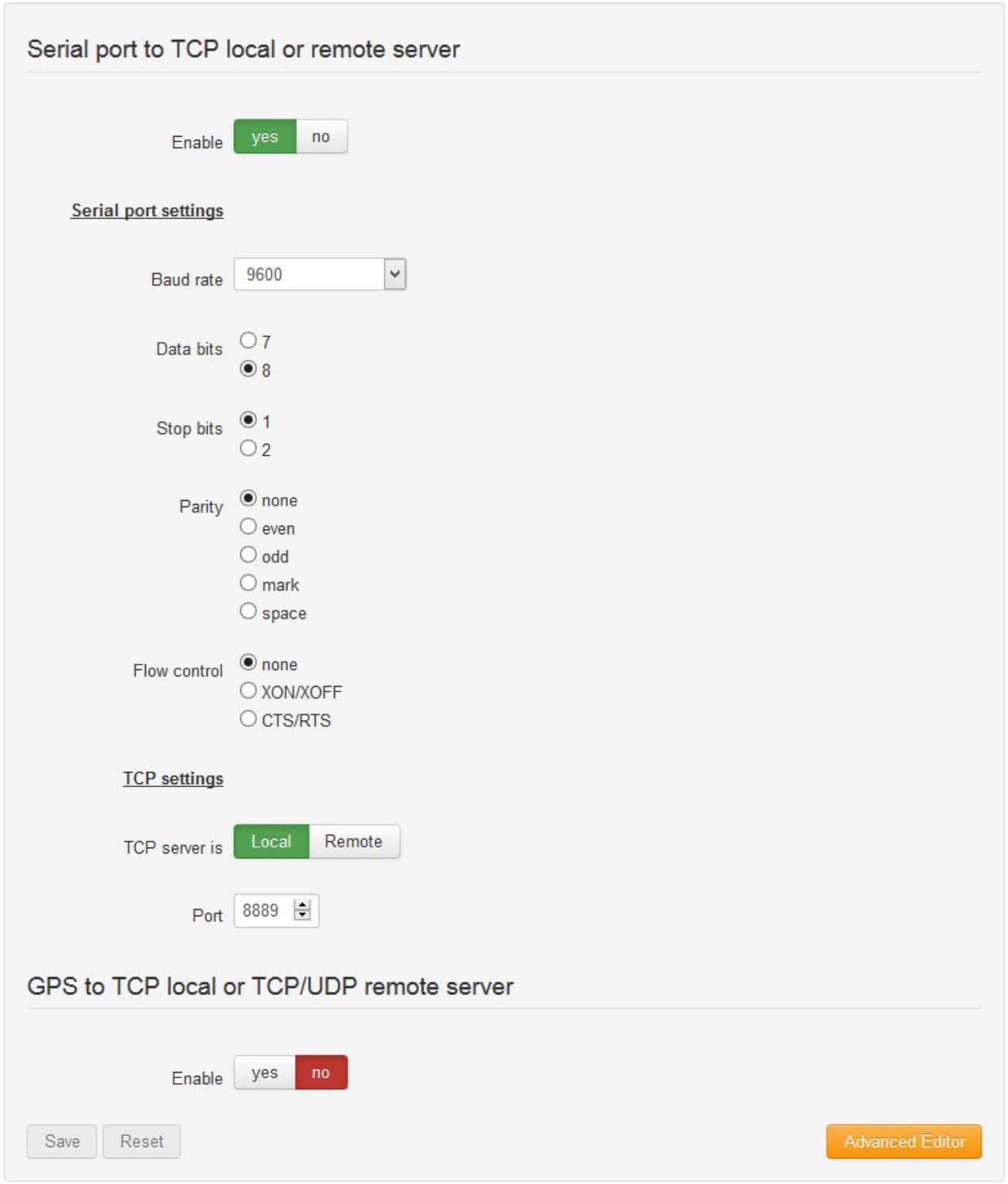


Figure 7: Serial to local TCP server

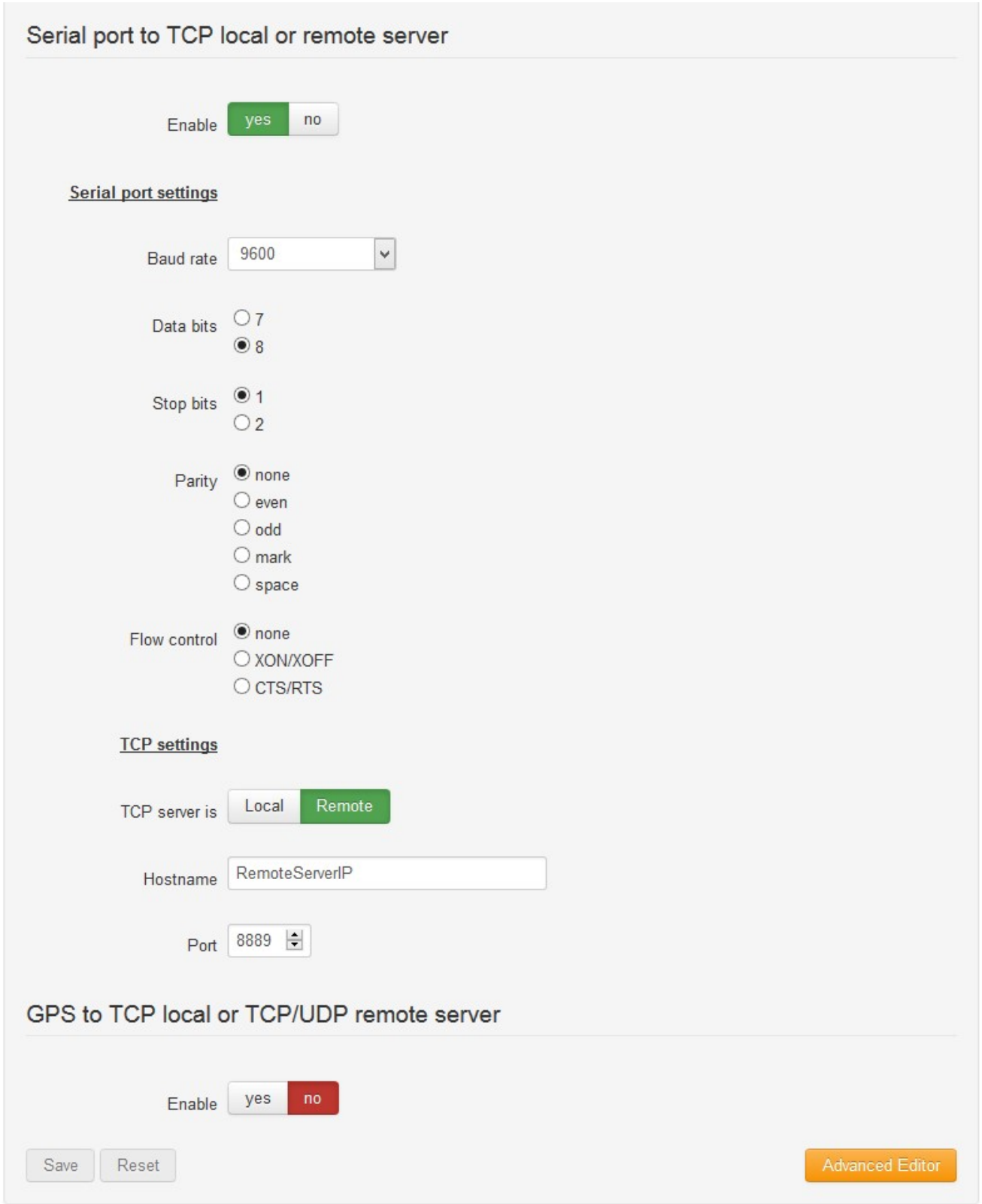


Figure 8: Serial to Remote TCP server

Both configurations allow the configuration of the serial interface (***Serial port settings***):

- Baud rate
- Data bits
- Stops bits
- Parity
- Flow control

These settings should match the settings of the device connected to the serial interface.

On Figure 7, CloudGate is running a local TCP server that will listen for incoming connections and forward these connections to the serial port. The Port number of the TCP server is **8889** by default but this can be changed if needed.

If access from the WAN interface (internet) is needed, an appropriate firewall rule needs to be in place to allow the connection to the port:

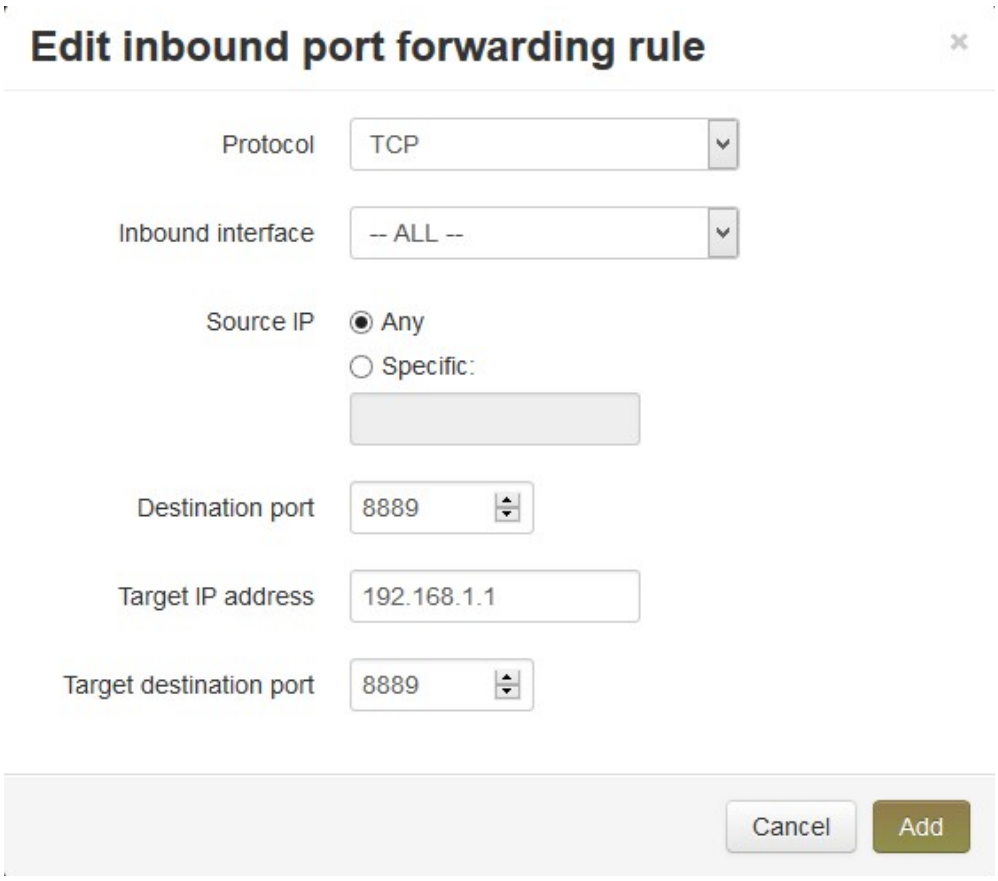


Figure 9: Inbound port forwarding rule

NOTE: Recent versions of LuvitRED already open a firewall hole to allow remote access from the WAN interface. This can be verified only under the Advanced Editor, not on the Basic Interface.

In Figure 8, CloudGate will connect to a remote TCP server running on the specified port and send all the information that arrives from the device connected to the serial interface.

Advanced Editor

After configuring the serial port under the Basic Interface, one can go to the Advance Editor and further edit the configuration. The configuration made on the Basic Interface will be reflected under the Advanced Editor in the following way:

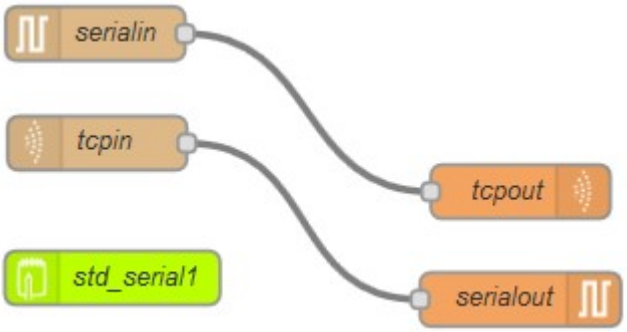


Figure 10: Same configuration under Advanced Editor

Using the second RS232

Double click on the ***serialin*** node:

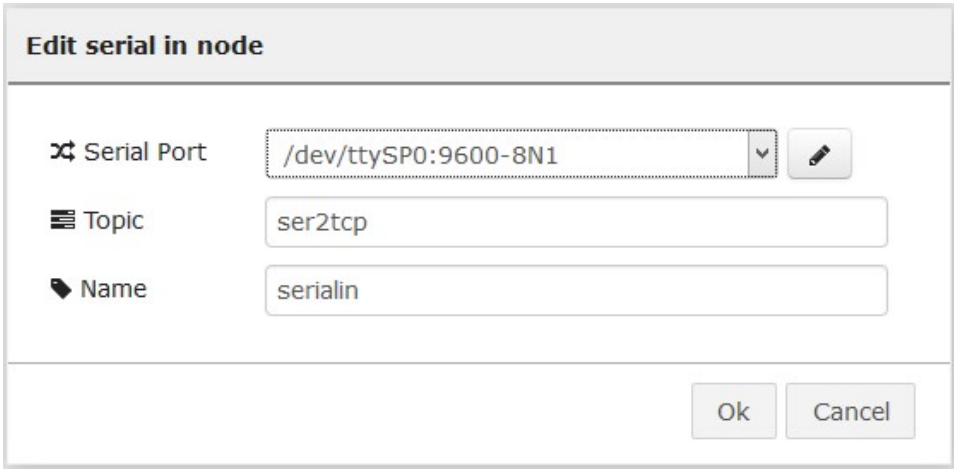


Figure 11: Serial node general configuration

Click on the pencil icon:

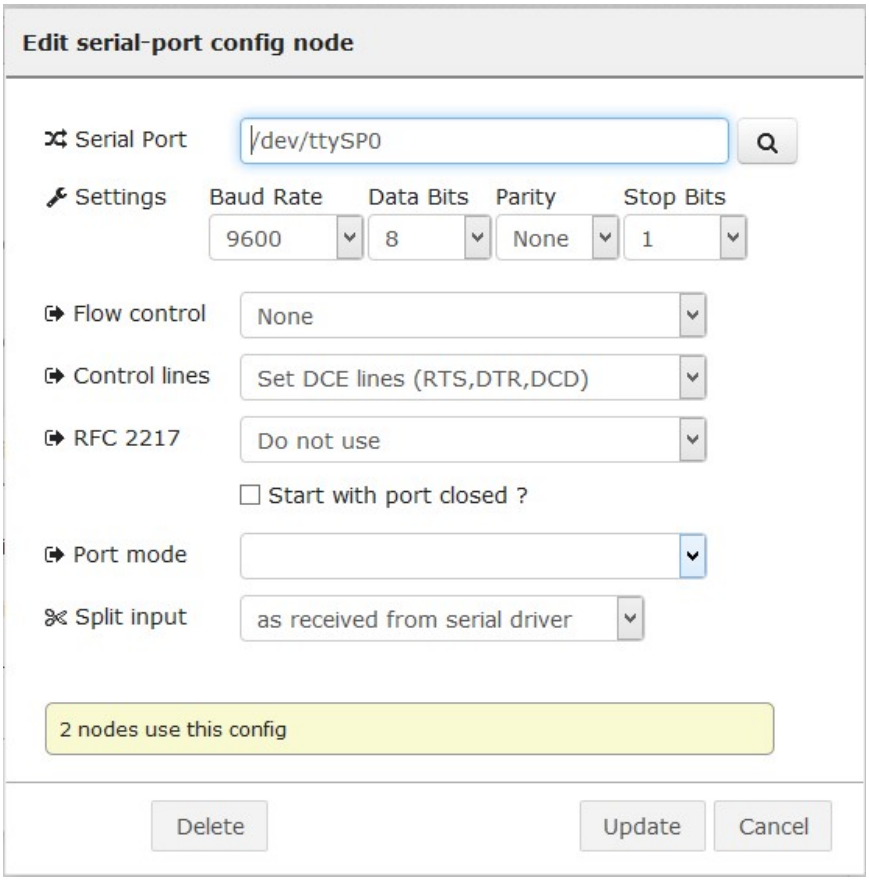


Figure 12: Serial interface configuration

Clicking on the magnifying glass will give access to the interface selection:

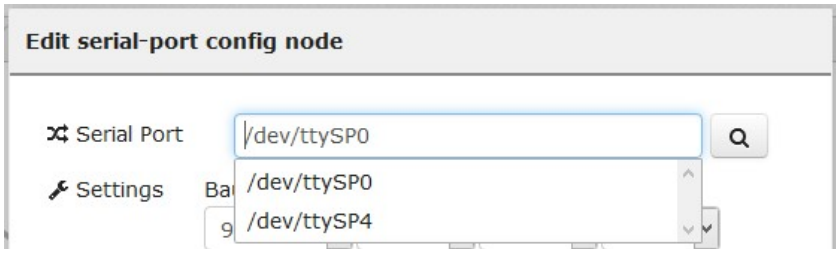


Figure 13: Interface selection

Selecting the ***/dev/ttySP4*** interface will modify the configuration to work with the second RS232 interface instead of the first RS232 interface. Click Update, then click OK and then click Deploy (at the top right corner).

Verifying firewall hole

Double click on the ***tcpin*** node:

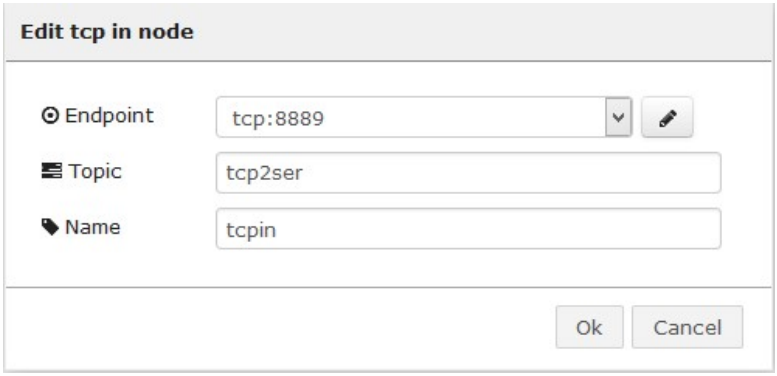


Figure 14: Tcpin node general configuration

One can access the "Endpoint" configuration by clicking on the pencil icon:

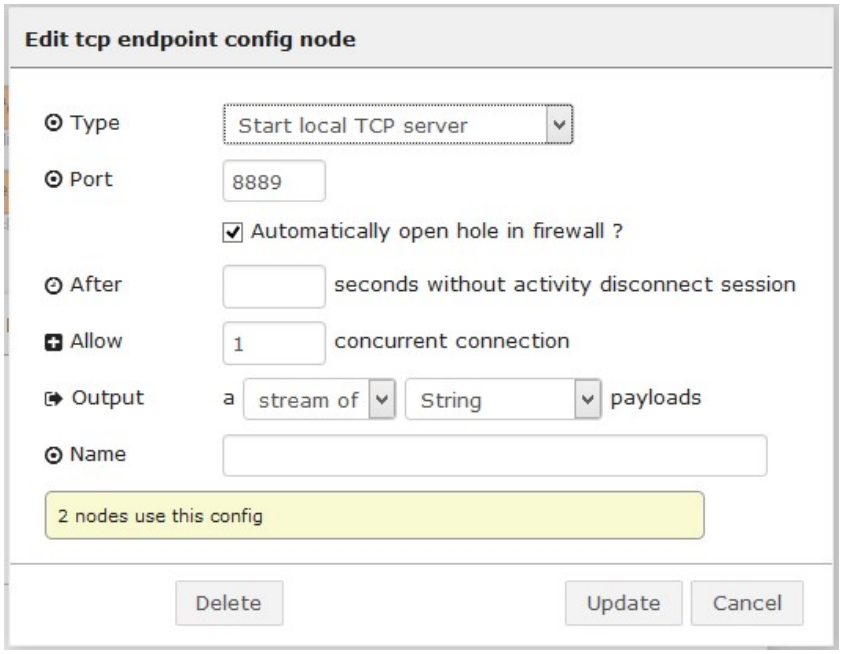


Figure 15: Endpoint configuration

Check if the configuration item called **"Automatically open a hole in firewall?"** is checked or modify it as needed.

Inactivity Timeout on the TCP node

Open the "Endpoint" configuration again. Add a timeout, in seconds (30 on the example below), on the **"After ___ seconds without activity disconnect session"** configuration item so that it closes any open connection that is not generating traffic:

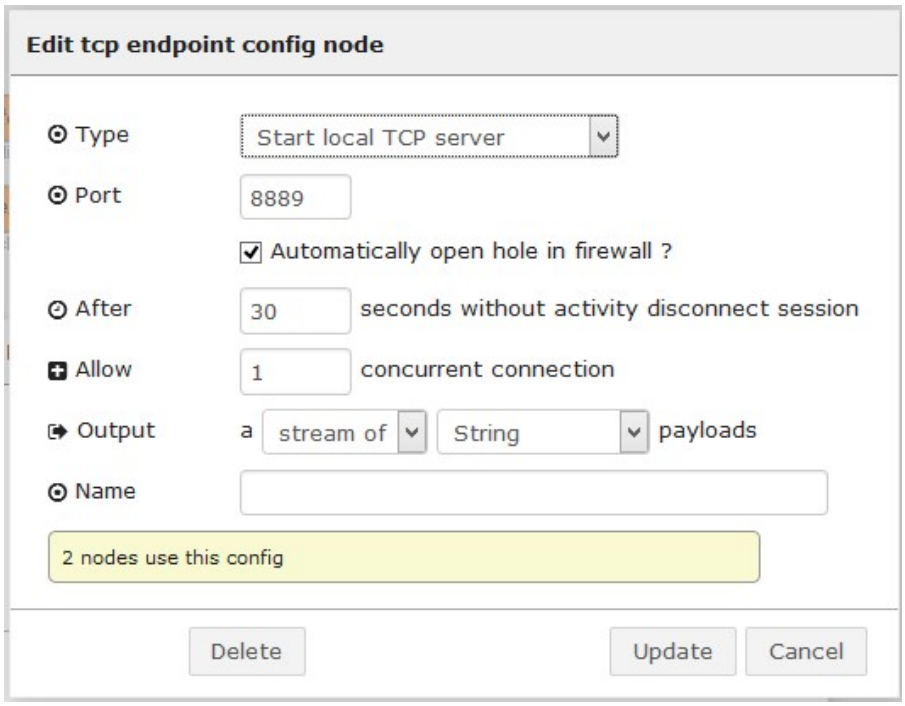


Figure 16: Adding connection timeout

Advanced Editor - Both RS232 ports at the same time

To reduce deployment time, let's start from the configuration created using the Basic Interface of LuvitRED for the RS232 interface:

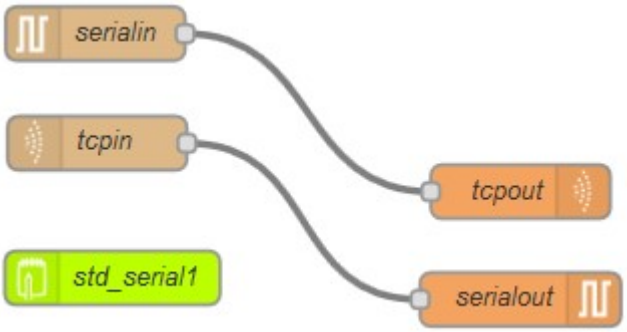


Figure 17: Basic configuration (single RS232)

Click on the green ***std_serial1*** node and delete the node by using the delete key on the keyboard and then click on Deploy:

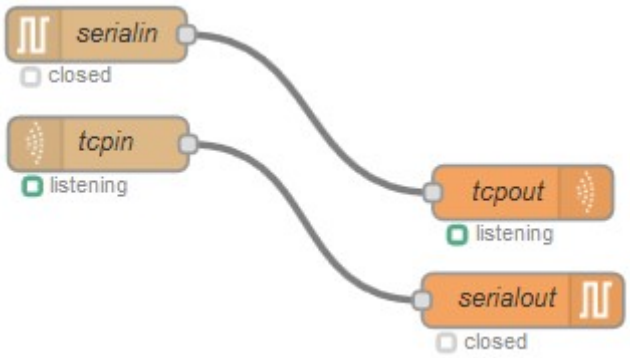


Figure 18: RS232 configuration without link to basic interface

Select all nodes using the mouse pointer and then copy the node using CTRL+C and paste them again using CTRL+V into the Editor section of LuvitRED:

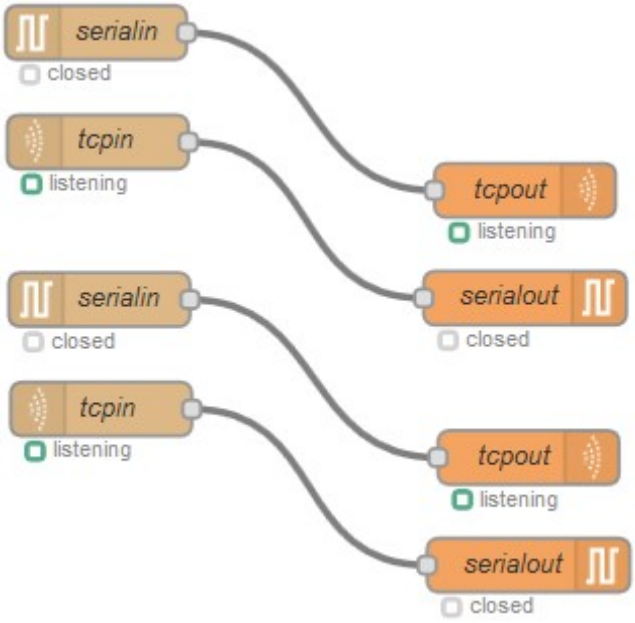


Figure 19: Copy and Paste the flow

Let's modify the name of the new nodes to avoid problems when editing them. Simply double click on the node to rename and change the configuration item called "Name":

Figure 20: Renaming the tcpin node

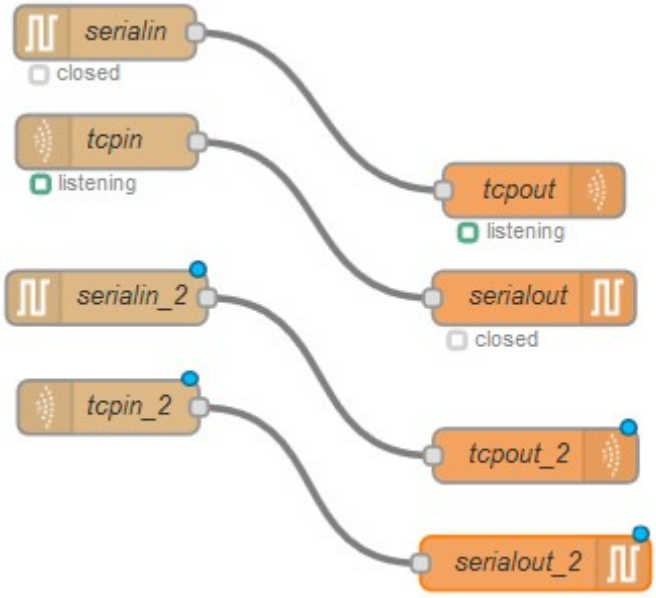


Figure 21: Renamed nodes

Double click on the ***serialin_2*** node to edit the node and select "Add new serial-port", and then click on the pencil icon:

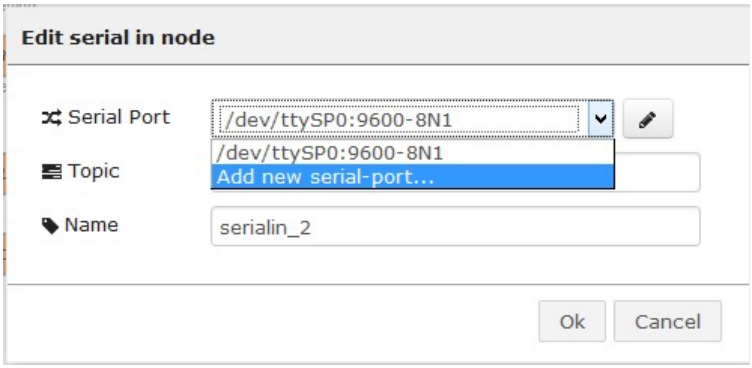


Figure 22: Adding a new serial port

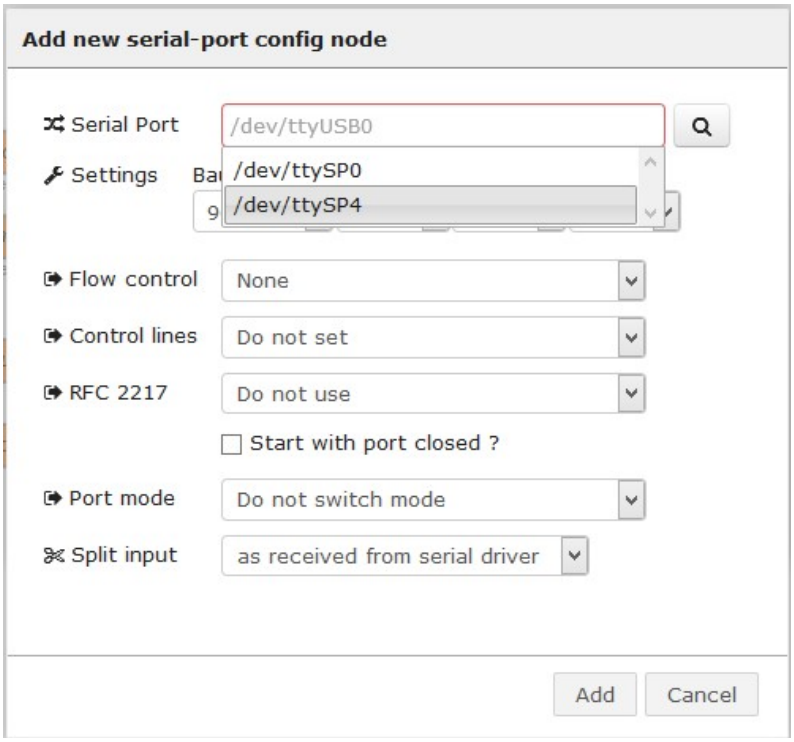


Figure 23: Adding the second RS232 port

Edit the second RS232 as needed, then click "Add" and then "OK".

Open the ***serialout_2*** node and select the newly created configuration for the second RS232 interface:

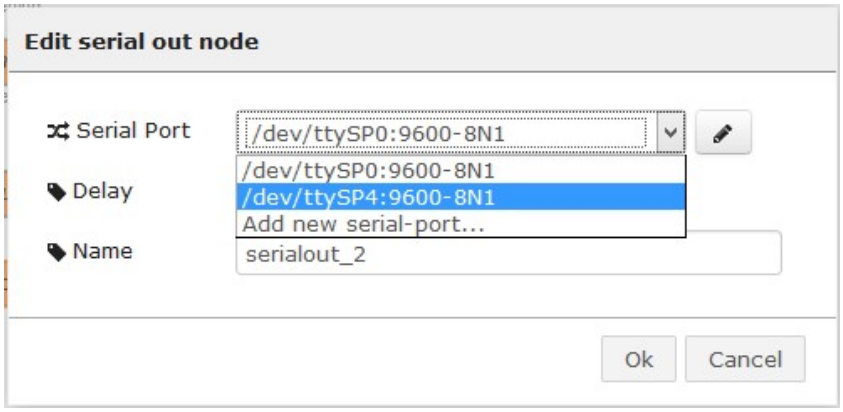


Figure 24: Select the configuration for the second RS232 interface

Click "OK".

Open ***tcpin_2*** and select "Add a new tcp endpoint", then click the pencil icon:

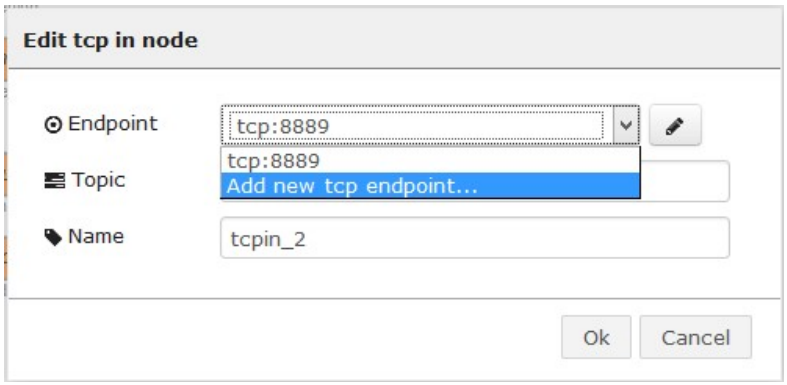


Figure 25: Adding a new tcp endpoint

Click "OK".

Start a local TCP server, but on the port, make sure a different port number than the one used on the first RS232 configuration is selected:

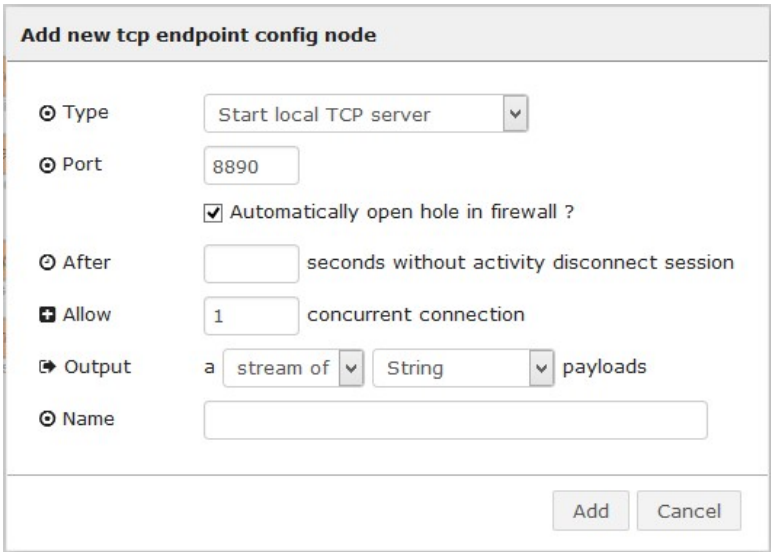


Figure 26: Local TCP server on port 8890

Click "Add" and then "OK".

Open the ***tcpout_2*** node and select the newly created TCP endpoint running on port 8890:

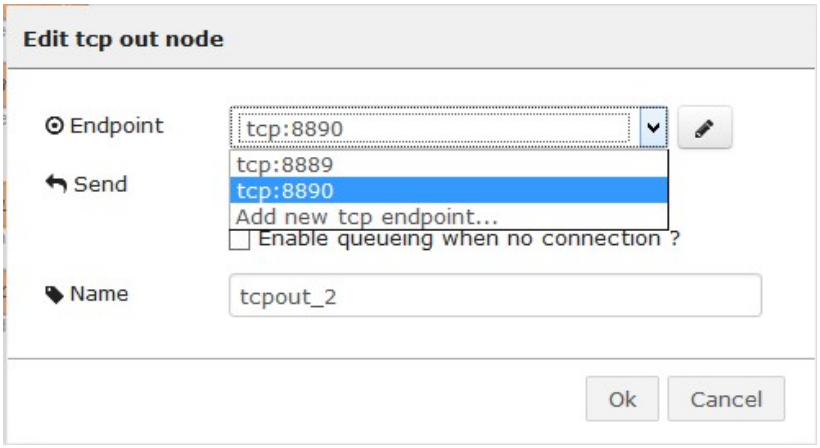


Figure 27: Select the new endpoint

Press "OK" and Deploy the configuration.

When looking at the "Configuration nodes" list one should see that four configuration nodes are available and that each of those are used by two nodes:

Figure 28: Configuration nodes view

If the configuration was done correctly, access to the serial ports should be granted on TCP port 8889 for the first RS232 port and TCP port 8890 for the second RS232 port.

Using the USB port for storage

Starting with CloudGate firmware version 1.46.0/2.46.0, automount for USB and SD mass storage devices (FAT file systems only) is supported on the CloudGate hardware.

Any new FAT formatted drive will be mounted under the ***/mnt/*** directory. Normally these drives are mounted as ***sdX#***:

```
admin@cgate:~$ ls /mnt/
base_cfg  cust      cust_cfg  sda1      sdb1
admin@cgate:~$
```

Figure 29: Example of two FAT drives mounted on the Linux system (sda1 and sdb1)

In this example, a drives is mounted on the system as ***sda1***. The ***sda1*** drive is connected to the USB port.

Under the "Advanced editor" of LuvitRED, there are some nodes that are in charge of data storage and others for parsing data:

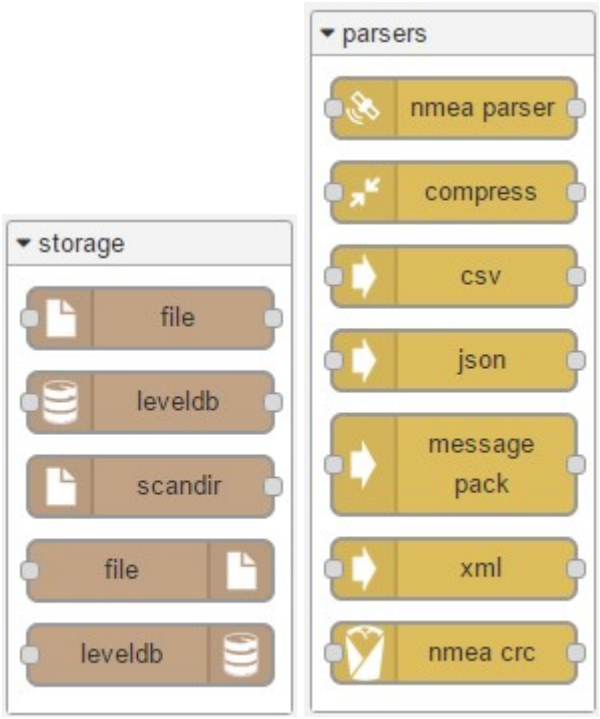


Figure 30: Storage and parsing nodes

Writing data to Mass Storage Device

Let's say we want to write a file to the ***sda1*** drive which currently looks like this:

```
admin@cgate:~$ ls /mnt/sda1/
1_DATA        HBCD          drivers       home
CLEAN         autorun.inf   grldr         menu.lst
admin@cgate:~$
```

Figure 31: Current view of ***sda1***

We can drop a file out node:

Figure 32: File out node

Configure the File out node like this:

1. Add the file name to write using the full location: ***/mnt/sda1/file.txt***

2. Choose an action for the node (append to file in our case), there are three actions available:

- Append to file
- Overwrite file
- Delete file

3. Choose if you want to add a ***newline*** character at the end of every line written to the file.

4. Change the node's name.

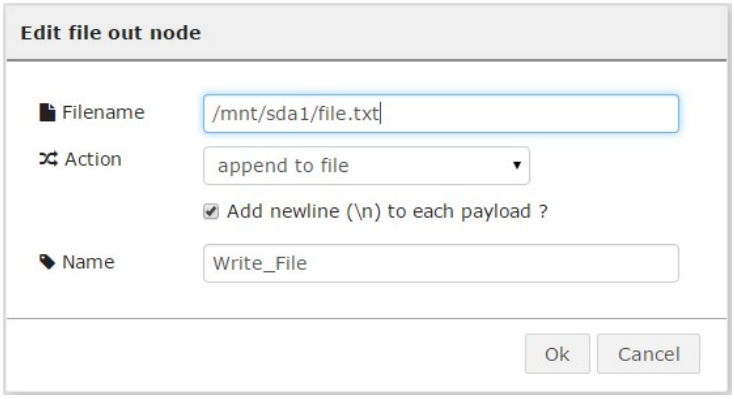


Figure 33: File out node configuration

Now, let's drop an inject node to send some data to the file out node. In this case the Inject node is configured to send a string **"write test"** every time we press the inject button:

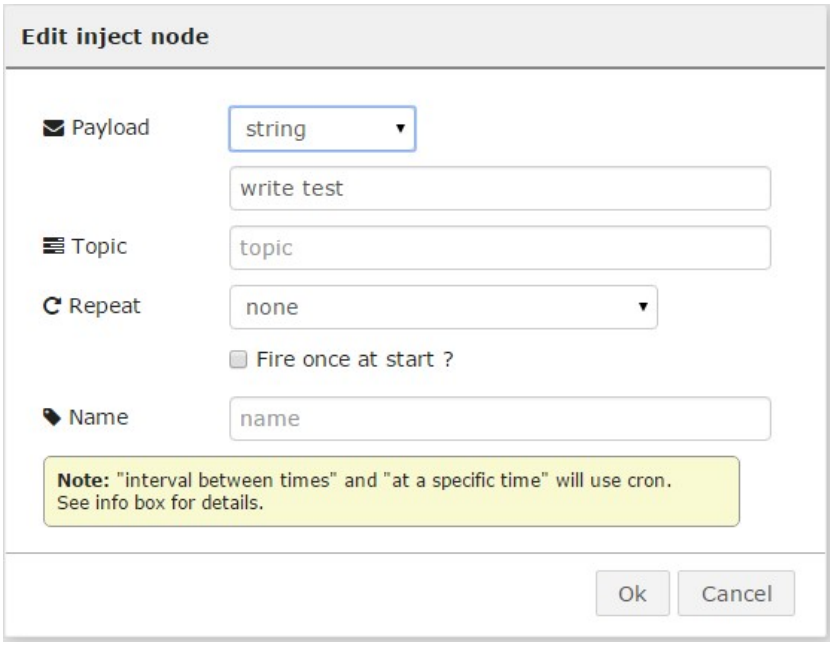


Figure 34: Inject node configuration

Now, connect both nodes together the following way:

Figure 35: Write flow

Deploy the configuration.

After pressing the inject button next to the inject node a few times (3 times in this example), the file contains the following information when reading it on a SSH session:

Figure 36: File containing new information on the ***sda1*** drive

Reading data from the Mass Storage Device

Now that we have written information to a file in the ***sda1*** drive, we want to read it back.

For reading the ***/mnt/sda1/file.txt*** we need first to drop a file in node:

Figure 37: File in node

Configure the file in node like this:

1. Add the filename to read using the full location: ***/mnt/sda1/file.txt***

2. Choose a Read file action for the node (once per message in our case), there are two actions available:

- Once per message
- Continuously

3. Choose if you want to delete the file after a successful read (leave it blank for our example).

4. Change the node's name.

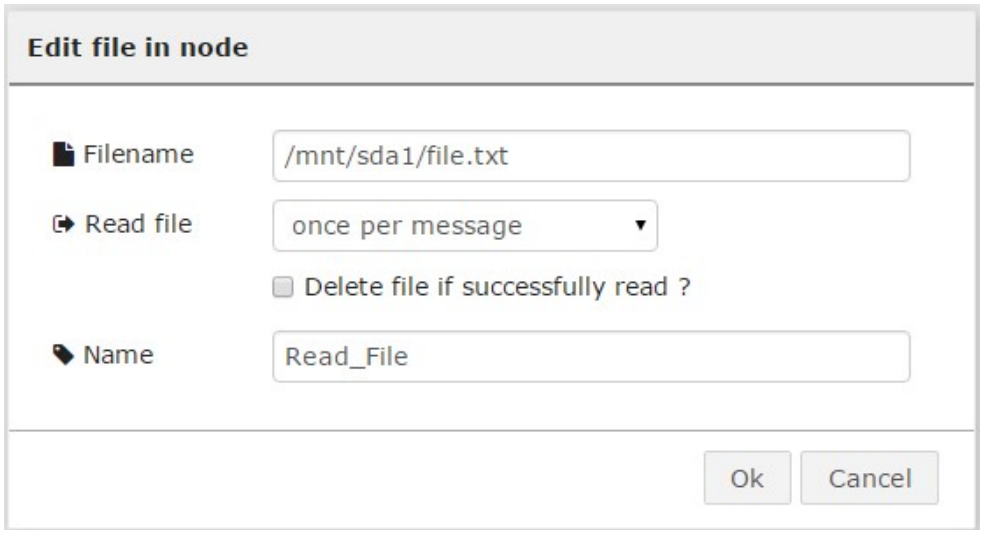


Figure 38: File out node configuration

Now, let's drop an inject node to trigger the file in node to read (this will be the message that the node is waiting for reading "once per message"). In this case the Inject node is configured with its default values, so no change on its configuration:

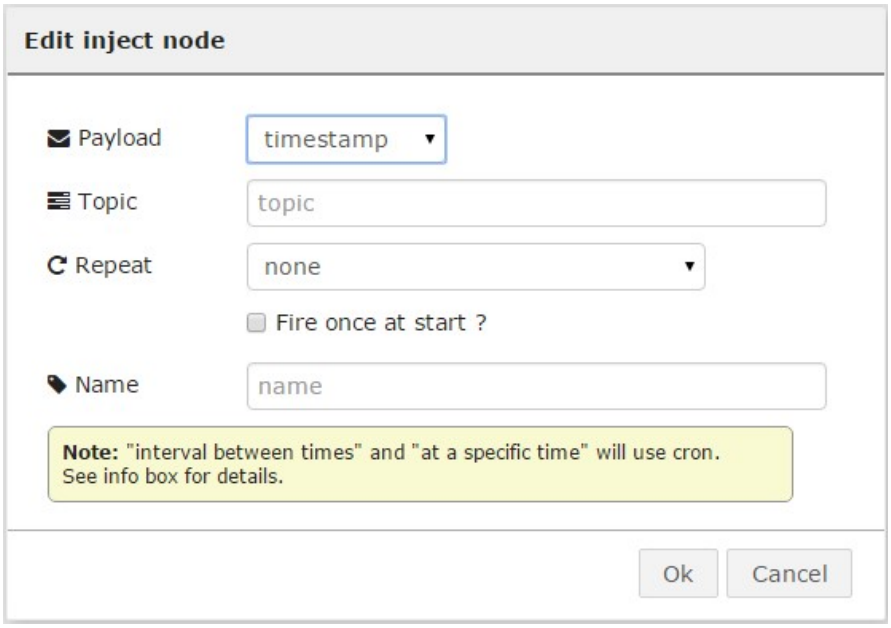


Figure 39: Inject node configuration

Let's also drop a debug node to view the result of reading the file. Connect the three nodes together the following way:

Figure 40: Read flow

Deploy the configuration.

After pressing the inject button next to the inject node, the debug node should print the reading made by the file in node and print the result on the debug tab:

Figure 41: Debug node printing the result of reading the file

Of course, printing the file to debug might not be something useful, but it is a good step to show that the file was correctly read. Instead of a debug node, or together with it, one could place other kind of nodes, for example, to send the file to a remote server, or make it available for a remote incoming connection.

Example Application: Flow Monitoring and Control