RBR*cervata* (pre 2025)

INSTRUMENT GUIDE

rbr-global.com

Table of Contents

1 Getting started . . . 3

1.1 General overview . . . 3

1.1.1 Data collection . . . 3

1.1.2 Storage format . . . 3

1.1.3 Removable storage status . . . 4

1.2 Software overview . . . 5

1.2.1 Diagnostic menu . . . 5

1.2.2 Configuration files . . . 6

1.2.3 Behaviour in an active deployment . . . 7

1.3 Hardware overview . . . 7

1.3.1 Opening the RBRcervata . . . 7

1.3.2 Inspecting O-rings . . . 8

1.3.3 Replacing an O-ring . . . 8

1.3.4 Replacing the desiccant canister . . . 9

1.3.5 Inserting/removing the micro SD card . . . 10

1.3.6 Installing the battery pack . . . 11

1.4 Quick start guide . . . 12

1.4.1 Before a deployment . . . 13

1.4.2 Data retrieval . . . 13

1.4.3 Terminal emulator . . . 14

1.5 Deployment calculator . . . 16

2 Reference . . . 16

2.1 Diagnostic menu reference . . . 16

2.1.1 Available commands . . . 16

2.2 Cervello configuration file . . . 25

2.2.1 Brief description . . . 27

2.2.2 Modifying configuration files on an RBRcervata . . . 29

2.3 Instrument configuration file . . . 29

2.3.1 Example files . . . 30

2.3.2 Modifying configuration files on an RBRcervata . . . 30

3 Revision history . . . 31

1 Getting started

1.1 General overview

The RBR*cervata* data controller acts as a removable storage and power device for conventional RBR data loggers. Upon connection to the data/power port of a logger, the RBR*cervata* provides external power and continuously downloads from the data logger's internal storage to the SDXC card internal to the data controller. Based on its own configuration, the data controller may also reconfigure the data logger as necessary. It was designed to be deployed adjacent to, and in support of, RBR*quartz*^{3} BPR and APT instruments. To facilitate easy bulk data retrieval from the system, the SD card can be removed from the data controller and connected to a desktop computer in order to retrieve the data.

An instrument can be connected to or disconnected from the RBR*cervata* at any time. Power delivery to an attached instrument is constant, regardless of the state of the data controller, including deep sleep.

1.1.1 Data collection

The RBR*cervata* will periodically download data from the attached instrument. And, when the instrument's internal storage gets too full, the data controller will stop and restart the deployment to minimize downtime.

Datasets are stored on the SD card in the `data/` directory. Datasets are stored in an RSK format compatible with Ruskin. Data within these RSKs is only minimally processed, and each dataset must be opened with Ruskin at least once before it can be opened with RSKtools.

1.1.2 Storage format

The data controller can use SD, SDHC, or SDXC memory cards which are formatted with ext2/3/4, or FAT32 filesystem. (exFAT is not supported.) The system ships with a 128GB, FAT32-formatted SDXC card installed, which is sufficient to store around three years of 10-channel, 32Hz data.

1.1.2.1 Energy conservation

The RBR*cervata* will enter a deep sleep mode whenever it is not downloading from the connected instrument. It will periodically wake itself, download any new data from the instrument, then return to sleep. The sleep duration can be configured to balance the trade-off between power consumption and the recency of data upon retrieval of the data controller.

1.1.2.2 Deployment management

The RBR*cervata* has an RS232 diagnostic port that can be accessed via the MINK connector for post-2021 models, for previous ones it can be accessed via the internal DB9 port. This port exposes a fully featured linux terminal using 'rbash'

with a set of scripts that enable the user to manage the behaviour of the data controller. The RBR*cervata* is able to alter instrument deployment parameters according to its internal configuration (e.g., sampling rate, bursting parameters). When an instrument's existing configuration doesn't match that which the data controller expects, it will stop and redeploy the instrument.

1.1.3 Removable storage status

Green LED	Red LED	Meaning
Off	Off	The system is powered down or is in deep sleep.
On	On	Immediately upon power-on, both LEDs are illuminated.
Off	Blinking	The data controller is writing to the SD card as it downloads from a connected instrument, or as it collects internal diagnostic information.
On	Off	The data controller is powered and running normally, and the SD card is not being actively used. It should enter deep sleep shortly after.

1.1.3.1 Support kit

The RBR*cervata* ships with a support kit that includes the following:

- 1x O-ring removal tool.
- 1x Silicone grease.
- 5x O-ring 2-243, RBR #0002408
- 5x Backing ring 8-243 RBR #0009115
- 1x Desiccant canister.
- 1x 128GB USB/USB-C drive.
- 1x USB/USB-C Card reader.

1.2 Software overview

The RBR*cervata* is capable of fully managing RBR loggers connected to it according to the settings stored in it. This includes downloading, clock syncing, dataset rollover and configuration. All of the user-accessible controls are available through a serial console on its Diagnostic Port (115200 8N1).

1.2.1 Diagnostic menu

This menu can be accessed via the diagnostic port using an RS232 interface (115200 8N1). This menu is a fully-featured Bash terminal (Restricted Bash) with 'tab' completion. The available commands are shown by executing the 'help' command. You will find all you need to set up a deployment using the RBR*cervata* and RBR loggers. As a quick in menu reference, each command prints a short manual by executing them with the '-h' or '--help' argument to it.

```
RBRcervello Menu.

These are the available commands to manage RBRcervello

   disable               Disable deployment.
   status                Provide a general overview or detailed information on
                         controller and instruments.
   diagnose              Diagnostics for MLM and telemetry.
   transparent-link      Establish a transparent link to an instrument
                         serial port.
   config-check          Print the current configuration of a specific RBR
                         subsystem.
   edit-config           Edit the current configuration of a specific
                         RBR subsystem.
   set-instrument-clock  RBR instrument clock sync.
   set-cervello-clock    RBRcervello clock setter.
   cervello-clock        RBRcervello clock checker.
   scrub-controller      Clear all data stored in the SD card.
   scrub-instrument      RBR instrument scrubber.
   enable                Enable deployment according to internal configuration
                         files.
   reboot-controller     Reboot system as if it had entered deep-sleep and
                         waken up.
   help                  Show this menu.

RBRcervello>
```

See the reference for more information: Diagnostic menu reference.

Help prompt example

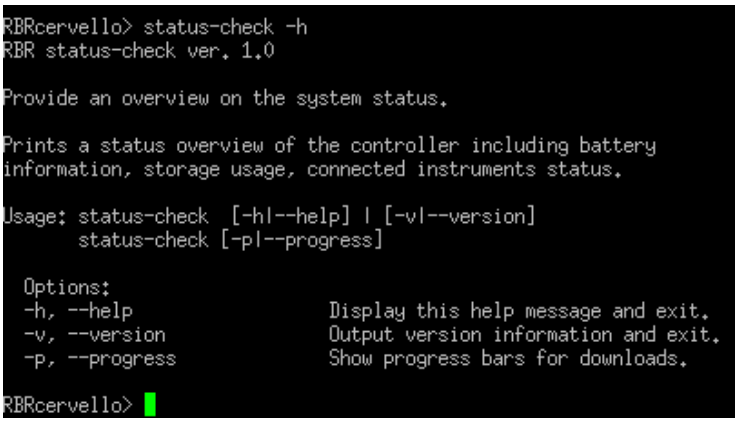


Fig. 1. 'status-check' help.

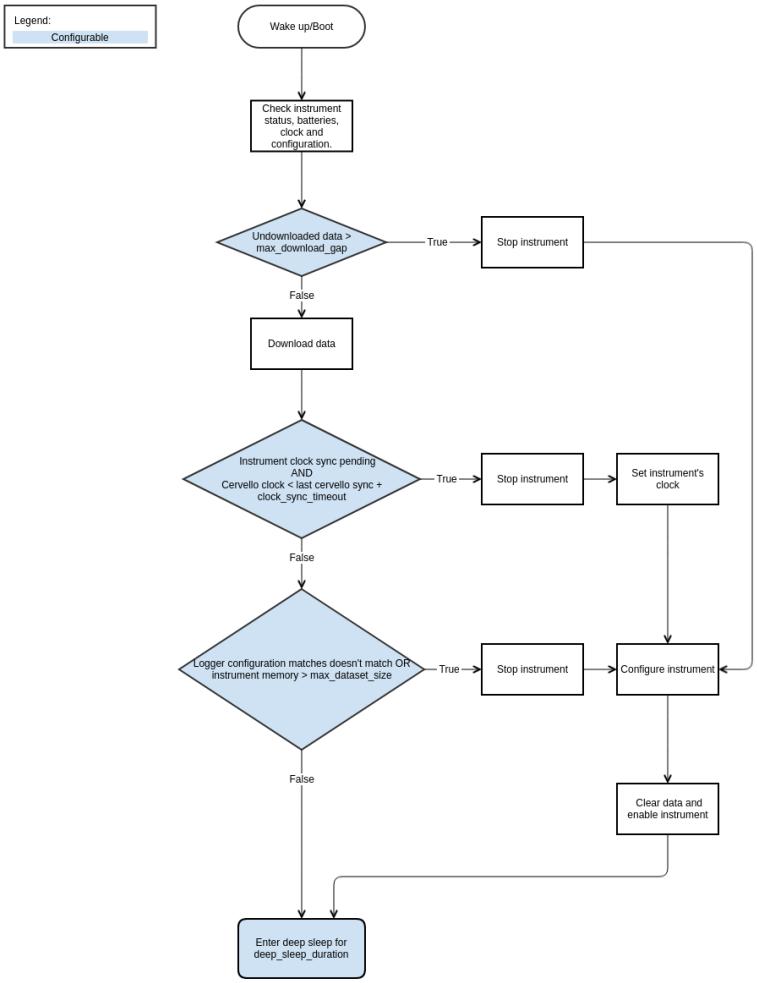
1.2.2 Configuration files

The RBR*cervata*'s behaviour can be described using its configuration files, Cervello configuration and Instrument configuration. The former defines how the RBR*cervata* is going to behave while the former dictates the configuration to be loaded into each instrument for each deployment.

See the reference for more information:

- Instrument configuration file
- Cervello configuration file

1.2.3 Behaviour in an active deployment



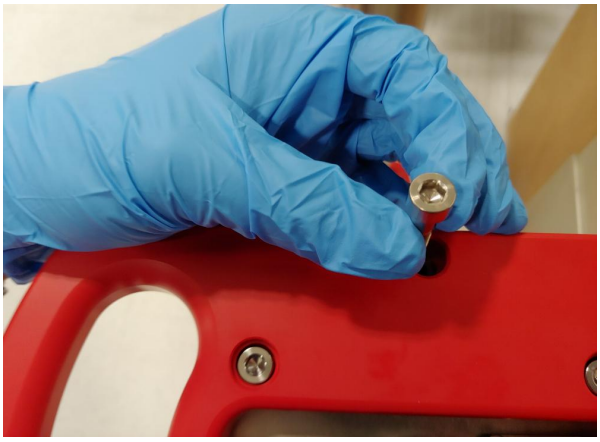
1.3 Hardware overview

- 5mm Hex/Allen key.
- Nitrile gloves.

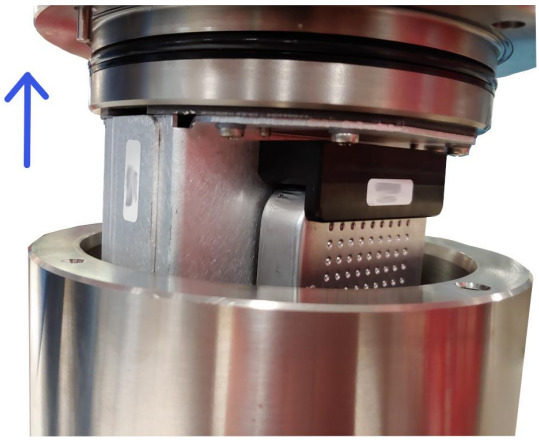
1.3.1 Opening the RBR*cervata*

The RBR*cervata* is serviceable by opening the "battery end-cap". The battery end-cap will be the side with the connectors on it. Only the battery end cap needs to be opened to access the controller, desiccant and battery.

1. Remove the two opening screws from the end with connectors using a 5mm Hex/Allen key.



2. Pull carefully, but do not twist.



1.3.2 Inspecting O-rings

Pay attention to the following areas:

- The surface of the O-ring itself
- The mating surface on the inside of the case between the threads and the open end
- The inner surfaces of the groove in the end cap where the O-ring sits

Any dirt present should be removed by wiping thoroughly with a soft, lint-free cloth. When cleaning, never use any material or tool which could scratch the O-ring or any of its mating surfaces. If any dirt is present in the O-ring groove, then the O-ring should be removed as described below to allow thorough cleaning of the groove. If an O-ring needs to be removed for any reason, it should be replaced.

If the O-ring is scratched, cut, distorted, perished or defective in any other way it must be replaced. If any of the surfaces of the O-ring groove are scratched, pitted or otherwise damaged, the logger may need to be returned to RBR for refurbishment: please contact us for advice.

1.3.3 Replacing an O-ring

1. Lever the O-ring from its groove. Use a soft plastic or wood tool; do not use a metal screwdriver or any other tool which may scratch the surfaces of the O-ring groove, doing so will probably render the end cap useless. Slide the O-ring out of its groove and off the logger. The O-ring may need to stretch quite a bit as it is pushed off; this requires some effort but can be done by hand.
2. Thoroughly clean the O-ring groove, taking great care not to scratch or damage it in any way. Inspect it carefully,

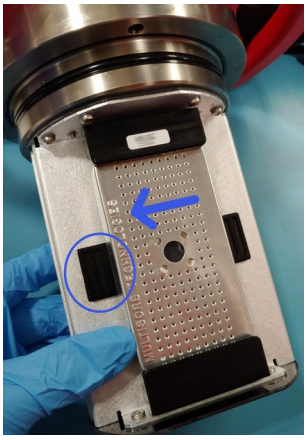
and contact us for advice if it appears damaged. Apply a light film of silicone grease to all three inside surfaces of the groove, being careful not to trap any dirt, hairs or lint.

3. Select the proper O-ring, see part numbers below, and make sure it is not damaged. Lubricate it with a very light film of silicone grease to ease its installation.
4. Install the new O-ring by pushing it into place and popping it into its groove.
5. When the new O-ring is in place, inspect it once more for scratches and dirt, and wipe away any silicone grease deposited on the end cap.

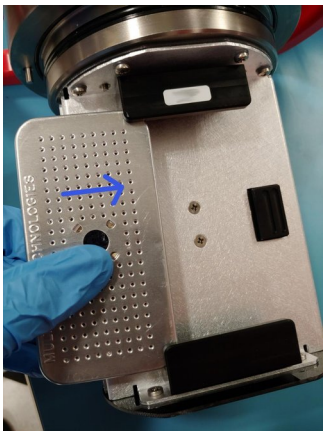
1.3.4 Replacing the desiccant canister

Replacing the desiccants is recommended before each deployment. The use of fresh desiccants will keep the RBR*cervata* compartment dry and prevent the electronics from being damaged due to condensation inside the controller. As a preventative measure, RBR recommends always use fresh desiccant packs and, when possible, service the instrument in a cool, dry place.

1. Remove the previous canister by pressing the black release mechanism and sliding the canister as shown below.



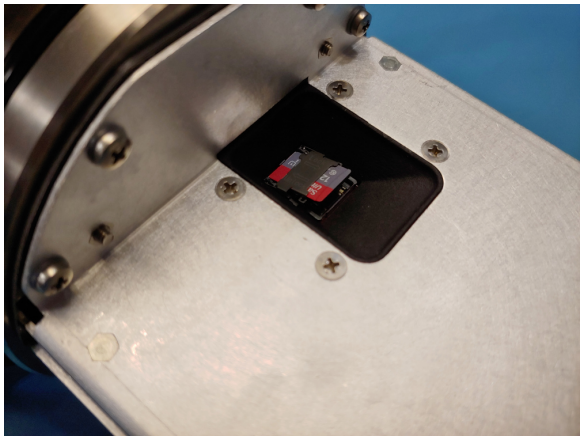
2. Insert the new canister by pressing the release mechanism with the canister itself and slide it in until it clicks into place.



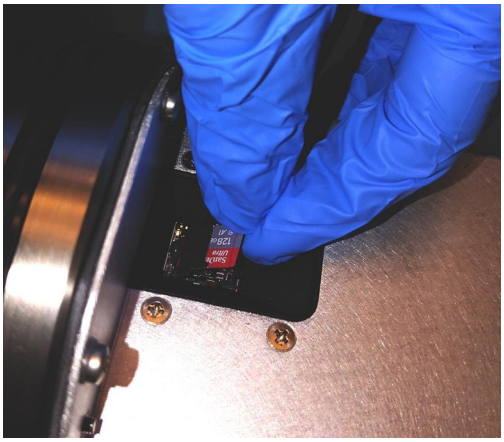
1.3.5 Inserting/removing the micro SD card

Please ensure that the RBR*cervata* is fully powered down and is not connected to internal or external power. The SD card is held by a metal carrier that keeps it in the correct position.

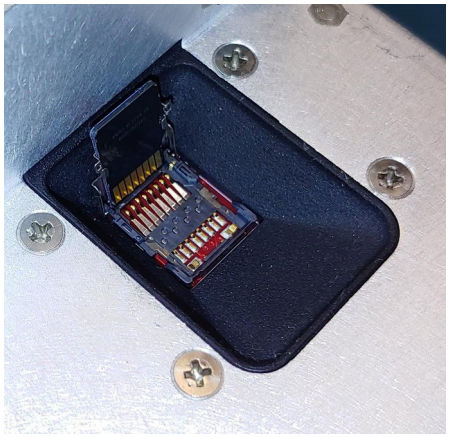
1. Slide the metal carrier on top of the micro SD card to the "Unlock" position, as stated in its engravings. The SD card should pop up as shown below.



2. Grab the SD card by carefully sliding it away from the metal carrier.



3. After retrieving the data from it, insert it back by sliding it into the metal carrier. Do not put the SD card directly over the contacts, it should be installed in the metal carrier before locking it.



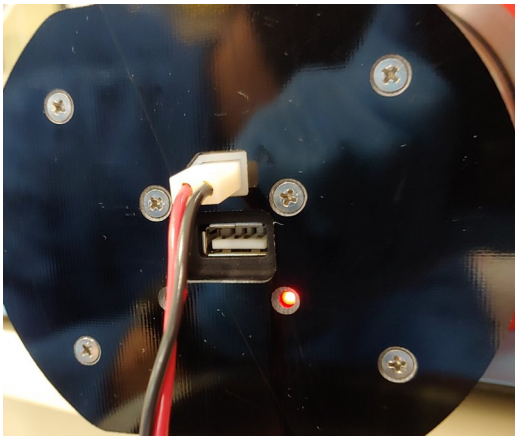
4. Lower the metal carrier with the SD card in it, and slide it into the "Lock" position.

1.3.6 Installing the battery pack

1. Remove battery top foam.

2. Insert the battery pack and add battery top foam passing the battery cable through it as shown below.

3. Connect the battery to the electronics and check for LEDs.



4. Insert the battery end-cap making sure the battery cable is not interfering and using the pin on the end cap for alignment.

5. Tighten the opening screws to 1.7Nm.

Fig. 2. Opening screws, torqued to 1.7Nm

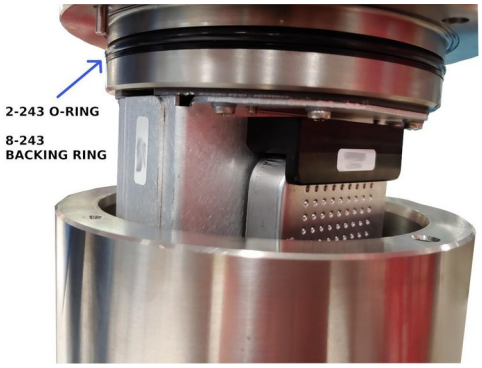


Fig. 3. O-rings

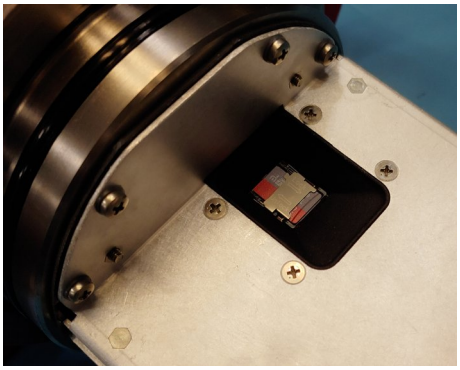


Fig. 4. SD card holder

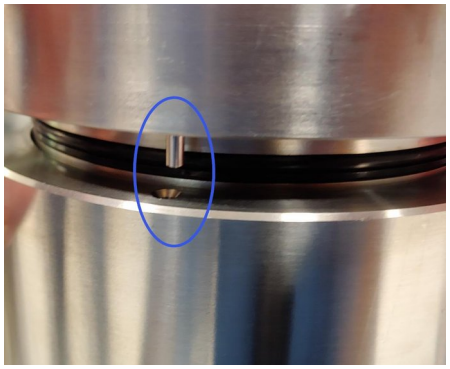


Fig. 5. Alignment pin

1.4 Quick start guide

- A computer with an RS232 adapter and a terminal client software such as *Putty*.
- An RS232 cable as appropriate to connect to RBR*cervata*'s diagnostic port.
- A suitable cable to connect the instrument to the RBR*cervata*

1.4.1 Before a deployment

In preparation for future deployment or an imminent one, follow the steps below.

1. Install the battery pack in the RBR*cervata*.
2. Connect the RS232 cable to the "diagnostic port" and a computer.
3. Open a terminal on the corresponding serial port with a 115200 Baud rate, 8 bits, no parity bit, 1 stop bit.
4. Press enter and wait until you see the prompt "RBRcervello>".
5. Type "stop-deployment" and press enter.
6. Use the "edit-config -c cervello" to edit the RBR*cervata* configuration.
7. Use the "edit-config -c instrument01" to edit the instrument 01 configuration. Add the -n flag to start from an empty file.
8. Use "scrub-instrument" to erase any data that might be stored in the instrument(s).
9. Execute "scrub-controller" to erase any data that might be stored in the RBR*cervata*.
10. Synchronize the clock of the RBRcervata using "set-cervello-clock *YYYY-MM-DD*T*HH*:*MM*:*SS*" with the appropriate time, preferably UTC.
11. Send the "set-instrument-clock -a" command to synchronize the clock of the instruments with the RBR*cervata*.
12. Use "status-check" to verify that the instruments and the controller and in the right state, in this case, stopped.
13. (Optional) issue a "start-deployment" command to initiate the deployment and enable the instruments right away.
14. (Optional) issue a "status-check" to verify the instruments are logging.

If the deployment is stopped, the RBR*cervata* will enter deep sleep until it receives data through its diagnostic port again. In this state, the instruments, as well as the RBRcervata, will enter an ultra-low power state.

1.4.2 Data retrieval

When the RBR*cervata* is retrieved after a deployment the following steps should be followed:

1. Connect the RS232 cable to the "diagnostic port" and a computer.
2. Apply power using the RS232 cable as the battery pack might be depleted.
3. Open a terminal on the corresponding serial port with a 115200 Baud rate, 8 bits, no parity bit, 1 stop bit.
4. Press enter and wait until you see the prompt "RBRcervello>".
5. Type "stop-deployment" and press enter.
6. Synchronize the clock of the RBRcervata using "set-cervello-clock *YYYY-MM-DD*T*HH*:*MM*:*SS*" with the appropriate time, preferably UTC. This will allow for timestamp correction after retrieving the data.
7. Use "status-check" to verify that the instruments (if any) and the controller and in the right state, in this case, stopped.
8. Open the RBR*cervata* and retrieve the micro SD card according to the instructions provided in the Hardware overview. Make sure both LEDs are off before unplugging the battery and the RS232 cable.
9. The RSK files will be in the 'data' folder of the SD card, copy them over to the computer using the included SD card reader.

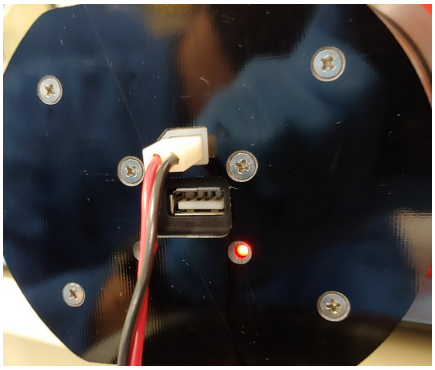


Fig. 6. Battery connection

Fig. 7. Micro SD card reader

1.4.3 Terminal emulator

- Baud rate: 115200 *baud/s.*
- 8bit, no parity-bit, 1 stop-bit: 8N1.
- No flow control.

A terminal emulator is used to communicate with the RBRcervata through its diagnostic port.

1.4.3.1 Windows

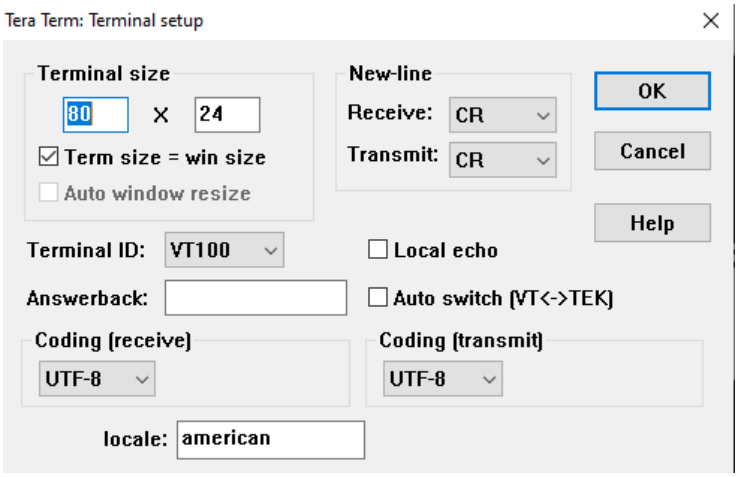
Teraterm

Teraterm is an open-source terminal emulator that lets you connect to hosts via serial port or the network. The default settings should work with the RBR*cervata* interface. Nevertheless, below you'll find the configuration file that can be loaded using the *Setup→Restore Setup…* menu.

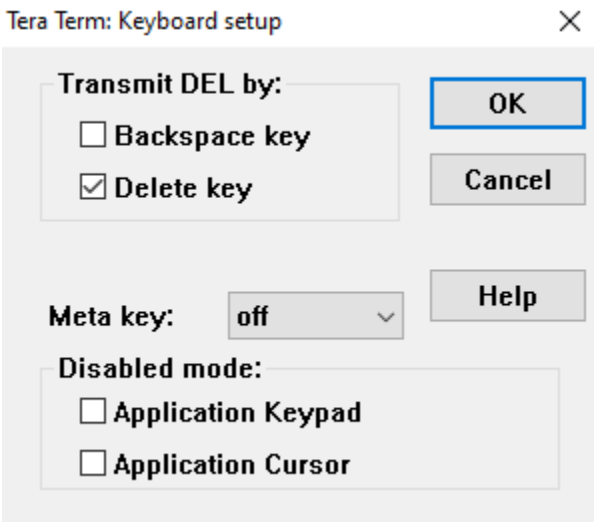
Teraterm_RBRcervata.INI

Alternatively, you could configure the settings manually as follows:

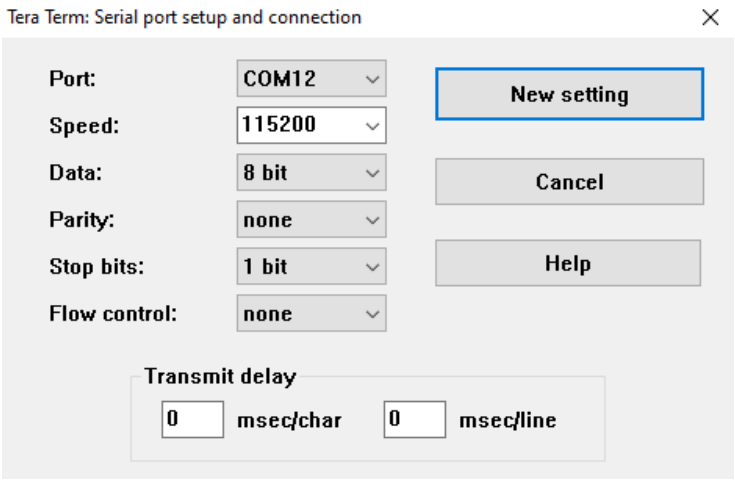
- In *Setup→Terminal…*:



- In *Setup→Keyboard…*



- In *Setup→Serial Port…*:



Highlighting text on the terminal automatically copies it to the clipboard and right-clicking on the terminal pastes text on it.

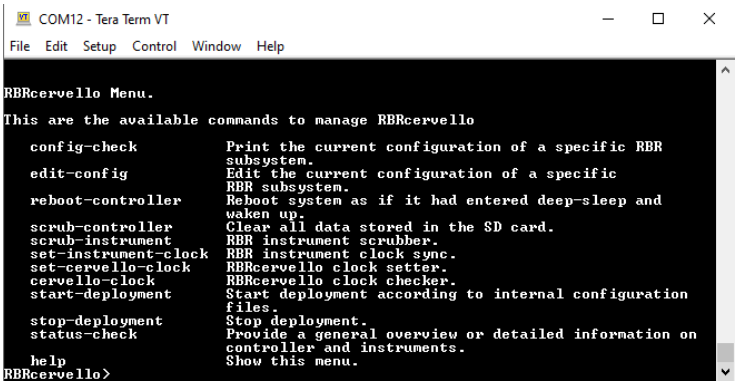


Fig. 8. Teraterm

1.4.3.2 Linux

On a Linux-based workstation, any terminal emulator console that supports a serial port can be used.

Minicom

You could either use *minicom -s* to set up the terminal with the right port and settings or execute it with explicit settings at runtime with the correct device:

```
minicom -b 115200 -D /dev/ttyUSB0
```

To copy and paste text from and to a Linux terminal, Ctrl+Shift+C and Ctrl+Shift+V should be used.

Fig. 9. Minicom

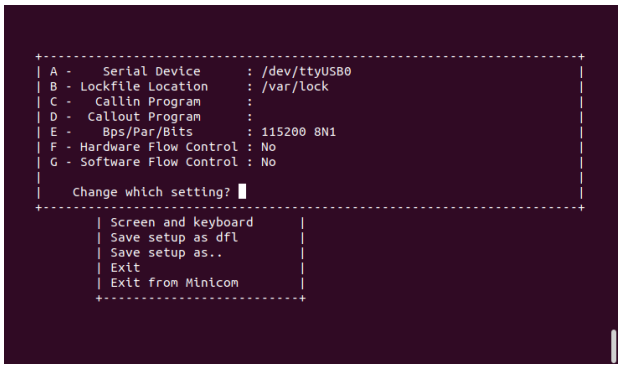


Fig. 10. Minicom serial settings

1.5 Deployment calculator

The deployment calculator is only visible online when viewing this page through the Viewport. It is currently located here:

https://docs.rbr-global.com/cervello/getting-started/deployment-calculator/

2 Reference

2.1 Diagnostic menu reference

2.1.1 Available commands

Brief description of the commands that a user can execute manually from a serial terminal on an RBR*cervello.*

```
RBRcervello Menu.

These are the available commands to manage RBRcervello

   disable               Disable deployment.
   status                Provide a general overview or detailed information on
                         controller and instruments.
   diagnose              Diagnostics for MLM and telemetry.
   transparent-link      Establish a transparent link to an instrument
                         serial port.
   config-check          Print the current configuration of a specific RBR
                         subsystem.
   edit-config           Edit the current configuration of a specific
                         RBR subsystem.
```

```
set-instrument-clock  RBR instrument clock sync.
set-cervello-clock    RBRcervello clock setter.
cervello-clock        RBRcervello clock checker.
scrub-controller      Clear all data stored in the SD card.
scrub-instrument      RBR instrument scrubber.
enable                Enable deployment according to internal configuration
                      files.
reboot-controller     Reboot system as if it had entered deep-sleep and
                      waken up.
help                  Show this menu.
```

2.1.1.1 status

Name

status - Status checker for RBRcervello and RBRcervata

Synopsis

Provide an overview on the system status.

status [-h|--help] | [-v|--version]

status [-p|--progress]

Description

Prints a status overview of the controller including battery information, storage usage, connected instruments status.

Options

-h, --help

Display this help message and exit.

-v, --version

Output version information and exit.

-p, --progress

Show progress bars for downloads.

Examples

status

```
RBRcervata
Serial number: 234567
Part number: 0010423
Cervello firmware version: 1.014

Storage used : 33MB
Storage remaining: 120GB
Storage size : 120GB

Cervello status: logging
Cervello internal battery (V): 11.95
```

```
Cervello external battery (V): 9.05

Instrument 01: logging
Instrument 01 internal battery (V): 6.00
Instrument 01 external battery (V): 11.49

Instrument 02: logging
Instrument 02 internal battery (V): 6.00
Instrument 02 external battery (V): 11.52

Download remaining: 1716B
```

status -p

```
RBRcervata
Serial number: 234567
Part number: 0010423
Cervello firmware version: 1.014

Storage used : 33MB
Storage remaining: 120GB
Storage size : 120GB

Cervello status: logging
Cervello internal battery (V): 11.95
Cervello external battery (V): 9.05

Instrument 01: logging
Instrument 01 internal battery (V): 6.00
Instrument 01 external battery (V): 11.49

Instrument 02: logging
Instrument 02 internal battery (V): 6.00
Instrument 02 external battery (V): 11.52

Download remaining: 1556B

Instrument 01: waiting
Remaining= 0B
```

[==========] 100%

```
Instrument 02: downloading
Remaining= 770B
```

[==========----------] 50%

2.1.1.2 config-check

Name

config-check - RBR configuration checker.

Synopsis

RBR configuration checker.

config-check [-h|--help] | [-v|–version]

config-check [-c|--config-file] [device]

Description

Prints the current configuration of a specific RBR subsystem.

Options

-h, --help

Display this help message and exit.

-v, --version

Output version information and exit.

-c, --config-file [*device*]

Show configuration file.

Where 'device' is the device of which configuration will be shown.

The supported devices are the following:

instrument[##] prints its corresponding deployment.json

cervello prints user.ini

Examples

config-check -c cervello

```
[metadata]
# The user.ini specification version to which this file will conform.
version = 2.0

[controller]
# How frequently to record the data controller battery level.
battery_period = 600 # 10 minutes
# Whether the data controller should enter deep sleep whenever it's finished
# downloading from the logger.
deep_sleep = true
# How long to stay in deep sleep before checking the logger for data again.
deep_sleep_duration = 21600 # 6 hours
#The number of seconds after the cervello clock sync under which the cervello clock
is
#considered good to be used as a synchronization source .
clock_sync_timeout = 43200 # 12 hours

[gps]
# Whether position retrieval via GPS is enabled.
enabled = false

...

```

```
[processing]
# Whether data processing for telemetry is enabled or not.
enabled = false

[telemetry]
# Whether telemetry is enabled or not.
enabled = false
```

2.1.1.3 scrub-instrument

Name

scrub-instrument - memory eraser for instruments

Synopsis

RBR instrument scrubber.

scrub-instrument [*instrument-number*]

scrub-instrument [-h] | [-a]

Description

scrub-instrument provides the means for clearing the internal memory of an instrument connected to the system.

Options

-h, --help

Display this help message and exit.

-v, --version

Output version information and exit.

instrument-number

Numeric id of the instrument to be erased.

-a

Erase all instruments.

Examples

scrub-instrument 01

Clearing memory…

Instrument 01: success

scrub-instrument -a

Clearing memory…

Instrument 01: success

Instrument 02: success

2.1.1.4 scrub-controller

Name

scrub-controller - clears the system's storage.

Synopsis

Clear all data stored in the SD card.

scrub-controller [-h] | [-f|--format]

Description

Stops any running service, clears the memory and restarts all the services.

Options

-h, --help

Display this help message and exit.

-v, --version

Output version information and exit.

-f| --format

(Not available for Cervata)

Formats the SD card in the proper ext4 filesystem

and initializes the folder structure.

2.1.1.5 set-cervello-clock

Name

set-cervello-clock - sets internal clock.

Synopsis

RBRcervello clock setter.

set-cervello-clock [*string-formatted-date*]

set-cervello-clock [-h]

Description

set-cervello-clock sets instrument clock and saves a previous and

new time in every Ruskin Database file involved, system's and intrument's

Options

-h, --help

Display this help message and exit.

-v, --version

Output version information and exit.

Examples

set-cervello-clock '2014-12-25T12:34:56'

Previous internal time: '2014-12-25T10:12:00'

New internal time: '2014-12-25T12:34:56'

2.1.1.6 set-instrument-clock

Name

set-instrument-clock - set instrument clock based on internal clock.

Synopsis

Set instrument clock with the system's time as reference.

set-instrument-clock [-h] | [-a|--all]

set-instrument-clock [*instrument-number*]

Description

set-instrument-clock sets instrument clock and saves a previous and

new time in every Ruskin Database file involved, system's and intrument's

Options

-h, --help

Display this help message and exit.

-v, --version

Output version information and exit.

-a |--all

Set clock on all connected instruments.

instrument-number

Numeric id of the instrument to be time synchronized.

Examples

set-instrument-clock 01

2.1.1.7 enable

Name

enable

Synopsis

Enable deployment according to internal configuration files.

enable [-h]

Description

Start deployment, configure instruments and enable them to start logging. Deletes any stopping flag for the deployment, whenever the system reboots, the deployment behaviour will continue.

Options

-h, --help

Display this help message and exit.

-v, --version

Output version information and exit.

2.1.1.8 disable

Name

disable

Synopsis

Disable deployment.

disable [-h]

Description

Stop deployment, disable instruments download their data and generate a persistent flag to prevent the deployment to start in future boot cycles.

Options

-h, --help

Display this help message and exit.

-v, --version

Output version information and exit.

2.1.1.9 reboot-controller

Name

reboot-controller - Gracefully reboot the system

Synopsis

Reboot system as if it had entered deep-sleep and waken up.

reboot-controller [-h]

Description

Gracefully stop all internal services and reboot the system. Useful

for testing deployment behaviour.

Options

-h, --help

Display this help message and exit.

-v, --version

Output version information and exit.

2.1.1.10 edit-config

Name

edit-config - edit configuration files.

Synopsis

Edit the current configuration of a specific RBR device.

edit-config [-n|–new-file] | [-c|--config-file] *device*

edit-config [-h]

Description

Edit the current configuration of a specific RBR device. It uses 'nano' as its editor.

Options

-h, --help

Display this help message and exit.

-v, --version

Output version information and exit.

-c, --config-file [*device*]

Where 'device' is the device of which configuration will be shown.

The supported devices are the following:

- instrument[##] prints its corresponding deployment.json

eg. "instrument01"

- cervello prints user.ini

-n, --new-file

Sets an empty file for copy/paste a new config file.

Examples

edit-config -c instrument01

edit-config -c cervello

2.1.1.11 **cervello-clock**

Name

cervello-clock - check Cervello clock

Synopsis

RBRcervello clock checker

cervello-clock [-h|--help]

Description

Check system's clock and its latest synchronization

Options

-h, --help

Display this help message and exit.

-v, --version

Output version information and exit.

Examples

cervello-clock

Last time sync: '2021-06-28 15:31:00'

Current internal time: '2021-06-29 11:54:24'

2.2 Cervello configuration file

The Cervello configuration is a JSON file stored in the RBRcervata. It defines user-adjustable settings, such as sampling rates for batteries, deep sleep periods, and general behaviours for controlling loggers. The JSON documentation can be found here.

```
{
 "file": {
 "date": "20230509000000",
 "version": "3.1.0",
 "schema": "0011713",
 "customer": "ONC"
     },
 "controller": {
 "battery_period": 600,
 "deep_sleep": {
 "enabled": true,
 "deep_sleep_duration": 600
        },
 "clock_sync_timeout": 43200
    },
 "gps": {
 "enabled": false
    },
 "instrument": [
        {
 "enabled": true,
 "label": "APT",
 "type": "RBR",
 "config_file": "deployment01.json",
 "port": 1,
 "ssm": 0,
 "battery_period": 1200,
 "can_be_stopped": true,
 "max_stopped_download_gap": 2000000000,
 "enforce_max_dataset_size": true,
 "max_dataset_size": 580608000
        },
        {
 "enabled": true,
 "label": "BPR",
 "type": "RBR",
 "config_file": "deployment02.json",
 "port": 2,
 "ssm": 0,
 "battery_period": 1200,
 "can_be_stopped": true,
 "max_stopped_download_gap": 2000000000,
 "enforce_max_dataset_size": true,
 "max_dataset_size": 414720000
        }
    ],
 "streaming": {
 "enabled": false
    },
 "processing": {
 "enabled": false
    },
 "telemetry": {
 "enabled": false
    }
}
```

2.2.1 Brief description

Key	Description	Units	Default Value
controller			
battery_period	How frequently to record the data controller battery level.	seconds	600
deep_sleep	Whether the data controller should enter deep sleep whenever it's finished downloading from the logger.	boolean	false
deep_sleep_duration	How long to stay in deep sleep before checking the logger for data again.	seconds	300
clock_sync_timeout	The number of seconds allowed to pass after setting the flag for the instrument clock sync to happen. It is meant to provide a safeguard against synchronizing the instrument clock with a stale and possibly inaccurate Cervello clock.	seconds	43200
gps			
enabled	Whether position retrieval via GPS is enabled.	boolean	false
instrument			
enabled	Whether communication with/downloading from an RBR data logger is enabled.	boolean	true
label	Human readable descriptor of the specific unit (eg. Bottom CTD)	string	""
type	Type of logger, defines driver to be used.	string	"RBR"

Key	Description	Units	Default Value
config_file	Configuration file for this instrument.	string	"deployment01.json"
port	Port number to which it is connected.	integer	1
ssm	If the logger is connected to the data controller via an MLM, the serial number of the SSM connected to the instrument.	integer	0
battery_period	How frequently to record the logger battery levels.	seconds	600
can_be_stopped	Whether the logger can be stopped and restarted when there is too much data in memory to download in a reasonable amount of time (see max_stopped_download_gap), or when the logger's configuration doesn't match the configuration found in `deployment.json`.	boolean	true
max_stopped_download_gap	If the logger will be stopped or is stopped and the difference between the amount of data stored on the data controller and recorded in logger memory exceeds this amount, the logger will be restarted and its memory reset.	bytes	3000000
enforce_max_dataset_size	Whether we should enforce a maximum dataset size. Useful for forcing the dataset to roll over in conjunction with the cervata.	boolean	false

Key	Description	Units	Default Value
max_dataset_size	The maximum permissible size of the dataset in logger memory. If it grows past this size, the logger should be restarted.	bytes	262144000
streaming			
enabled	Whether streaming data via the serial debug port is enabled or not.	boolean	false
processing			
enabled	Whether data processing for telemetry is enabled or not.	boolean	false
telemetry			
enabled	Whether telemetry is enabled or not.	boolean	false

2.2.2 Modifying configuration files on an RBRcervata

A command called "edit-config" is available within the diagnostic menu of the RBRcervata for editing the configuration files. For modifying a cervello configuration file you can issue the following command:

```
1 edit-config -c cervello
```

If you would like to have an empty file as a starting point add "-n" or "--new-file" parameter to the command:

```
1 edit-config -n -c cervello
```

You'll be greeted with a file opened in *nano* for ease of use in the terminal environment. If you chose to start from an empty file, you can copy/paste the entire text on it.

After you finish editing it, press Ctrl+o, enter and then Ctrl+x. It will ask for confirmation in case something went wrong, press 'y' if you want the file to be permanently saved or anything else to cancel the edition.

2.3 Instrument configuration file

The deployment configuration or instrument is a JSON file stored in the RBRcervata. It contains instrument deployment settings and is used by the controller to reset the instrument to a known state before starting the deployment.

The file/version tag should reflect the specification version to which the configuration file will conform. For RBRcervatas with firmware version 1.014, it would be version "3.000".

Only a section labelled "deployment" which has no parents is used in the file. These commands are parsed as the following; the nodes hierarchically below "deployment" are designated as the [command] and the key-value pairs below that are then set to the [attribute] and [value] respectively. For each line a command is synthesized which has the format '[command] [attribute] = [value]'. So the following example section:

```
{
     "file": {
          "version": 3.000
     },
     "deployment": {
          "sampling": {
               "mode": "continuous",
               "period": 125,
          },
          "deployment": {
               "starttime": 20000101000000,
               "endtime": 20990101000000,
          },
.....
}
```

Would issue the following commands to the logger:

```
sampling mode = continuous
sampling period = 125
deployment starttime = 20000101000000
deployment endtime = 20990101000000
```

Anything in the "settings" section of the file would trigger a corresponding permit command to be issued first. See L3 command reference for command information.

2.3.1 Example files

BPR|zero example.json

APT example.json

2.3.2 Modifying configuration files on an RBRcervata

A command called "edit-config" is available within the diagnostic menu of the RBRcervata for editing the configuration files. For modifying an instrument configuration file you can issue the following command:

```
edit-config -c instrument01
```

If you would like to have an empty file as a starting point add "-n" or "--new-file" parameter to the command:

```
edit-config -n -c instrument01
```

You'll be greeted with a file opened in *nano* for ease of use in the terminal environment. If you chose to start from an empty file, you can copy/paste the entire text on it.

After you finish editing it, press Ctrl+o, enter and then Ctrl+x. It will ask for confirmation in case something went wrong, press 'y' if you want the file to be permanently saved or anything else to cancel the edition.

3 Revision history

Earlier revisions of this document were issued as PDF editions and are available on request.

Revision No.	Release date	Notes
C	14 September 2026	Migrated. Reformatted and reviewed; no technical changes.