RTCU IEX

Version 1.00

logic IO
The M2M Enabler

RTCU IEX

1 RTCU IEX

1.1 Introduction

The **RTCU Instrumented Execution (IEX)** stands as a pivotal extension to the **RTCU Integrated Development Environment (IDE)**, significantly amplifying the platform's capabilities for advanced runtime analysis and in-depth debugging of applications deployed on NX32L-based devices.

RTCU IEX empowers developers with comprehensive monitoring and real-time instrumentation of executing code, leveraging a sophisticated hardware virtualization layer that uniquely facilitates dynamic and granular switching between physical and virtual hardware components. This allows for precise observation and control, for example, by virtualizing I/O interfaces while the core application logic interacts directly with the physical processor.

The seamlessly integrated "**Insight!**" tool provides an unprecedented level of visibility into application behavior, enabling real-time observation and on-the-fly manipulation of all global variables and thread states. Architected for both rigorous development-time debugging and critical in-field analysis of elusive and intermittent issues, **RTCU IEX** represents a substantial technological leap beyond the **RTCU Emulator**. By executing directly on the target hardware while offering the flexibility of shadowed virtual peripherals, **RTCU IEX** delivers far more accurate and representative insights into an application's true operational characteristics.

RTCU IEX - key features:

- Achieve emulator-like debugging power directly on physical **NX32L** devices for accurate real-world behavior analysis.
- Seamlessly integrates with the **RTCU IDE** for a streamlined development and debugging workflow.
- Supported by all **NX32L** capable devices.
- Comprehensive logging functionality.
- Local cable connection. Remote access is under development.
- **Insight!**
  - Live view/editor for all global variables.
  - Effortlessly pause and resume threads at any point to precisely analyze execution flow and identify timing-related issues.
  - Set coded breakpoints to automatically pause thread execution at specific points of interest for targeted debugging.
  - Inspect and even modify the call stack of paused threads to understand execution history and experiment with potential fixes.
  - Automatically resolves and displays PTR, ACCESS, and global variables for easier understanding of memory and data structures.
- **Virtual hardware:**
  - Digital and Analog input/outputs.
  - Power and battery.

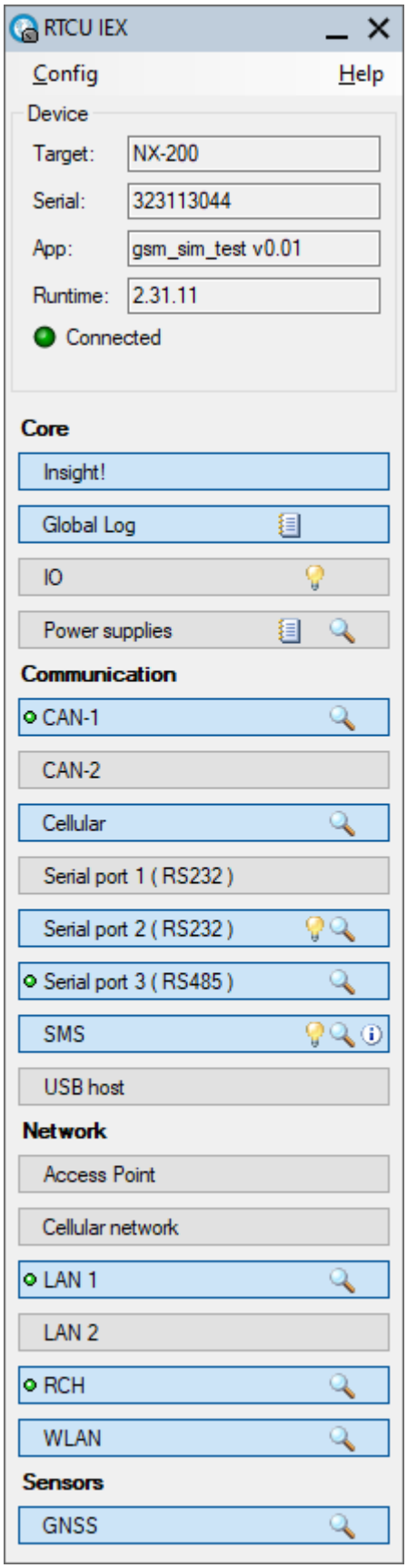
- CAN buses.
- GNSS / GPS.
- RS232 ports.
- RS485 ports.
- Modbus I/O Extension.
- SMS / PDU.
- Network interface state.

- **Monitoring**:
  - Detailed cellular information.
  - Detailed WLAN information.
  - RCH information
  - Access Point information.
  - USB host port information.

After years of meticulous development and drawing upon more than two decades of invaluable experience in the field, we are proud to offer **RTCU IEX** to support your diverse development requirements. This robust tool provides comprehensive instrumented execution, enabling efficient monitoring and debugging. Should you have any support inquiries or questions as you integrate **RTCU IEX** into your workflow, please do not hesitate to contact our support team at support@logicio.com.

1.2 Control Panel

The control panel is the main window of the program.
It shows the overall status of the instrumented execution and shows a list of the different modules that can be used.



Menu

Config

- Reset Layout: Moves all the open windows to just beside the control panel.
- Save Config: Saves the current window layout and configuration to a separate file. Can be used to have different configurations for different scenarios.
- Load Config: Load the window layout and configuration from a file.
- Disable
  - All: Disables logging, background monitoring and virtual for all modules.
  - Logging: Disables logging for all modules.
  - Monitor: Disables background monitoring for all modules.
  - Virtual: Disables virtual for all modules.

Help

- View Help: Shows this help file
- About: Shows the about dialog.

Device panel

The device panel shows the status of the device connection and information about the connected device.
The connection status icon can have three colors:

- Red: The MQTT broker is not connected.
- Orange: The MQTT broker is connected. The device can connect if it is configured to do so.
- Green: The device is connected.

Modules

Each module is represented by the name of the module with a possible status icon to the left of the name and a number of status icons to the right of the name:

Clicking the name of a module will show the window for the module, unless the module is already the active module, in which case the window will be closed.
See Modules for a list of the modules.

Right-clicking a module will show a context menu with the following content:

- Disable logging
- Disable virtual
- Disable monitor

These can be used to disable features for a specific module.

Control panel status icons

Status icon

To the left side of some modules, a colored dot is shown. This is used to show the status of the module, e.g. if it is turned on and if it is connected. This is only updated if monitoring is enabled

for the module. See the details for each module for details about what the color of the icon means.

To the right side, the following icons might appear:

Log

This indicates that logging is enabled for the module and that messages from it will be added to the global log.

Virtual

This indicates that the module is using at least one virtual feature instead of the physical feature.

Monitoring

This indicates that the module is currently monitoring what the device is doing. This is either because the module is open or because background monitoring is enabled.

Pending notification

This icon indicates that an event has occurred in the module while it did not have focus. This could e.g. be that it has received a message or that the status has changed. See the details for each module for details about what can trigger the notification.

1.3 Modules

The different modules have their own windows for managing the instrumentation.
The module windows can be closed with ctrl+w, clicking the X in the corner, or by clicking the button in the Control panel.
The modules are sorted by area for a better overview.

- Core
  - Insight
  - Global Log
  - IO
  - Power supplies
- Communication
  - CAN
  - Cellular Status
  - Serial Ports
  - SMS
  - USB Host
- Network
  - Access point
  - Cellular Network
  - LAN
  - RCH
  - WLAN
- Sensors
  - GNSS

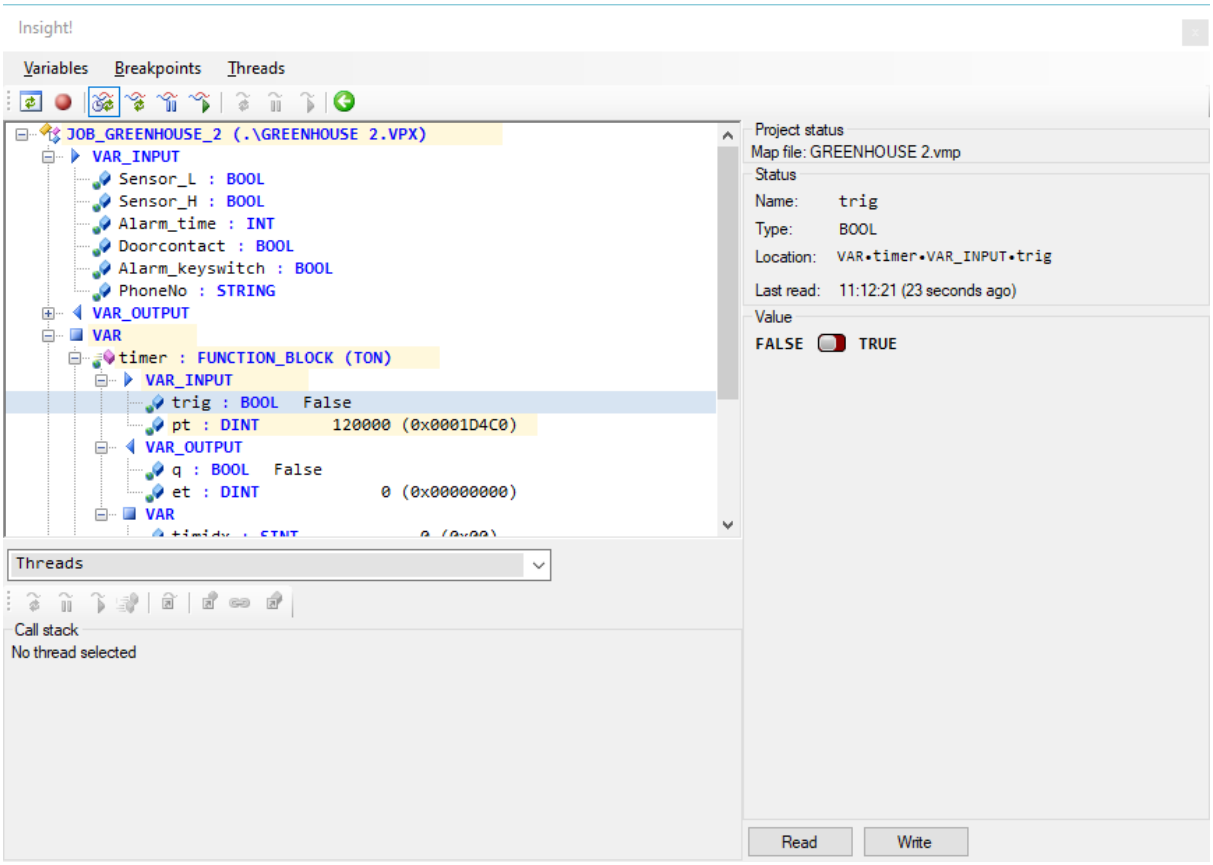
1.3.1 Core

The Core area contains the functionality that is common for all RTCU devices, which does not belong in other areas.

Modules:
- Insight
- Global Log
- IO
- Power supplies

1.3.1.1 Insight

The insight window provides a powerful window into the VPL application itself, giving access to view and modify variables and examine threads.



The Insight window consists of a menu with toolbar and three different areas: The global code view, the thread view and the details view.

Menu

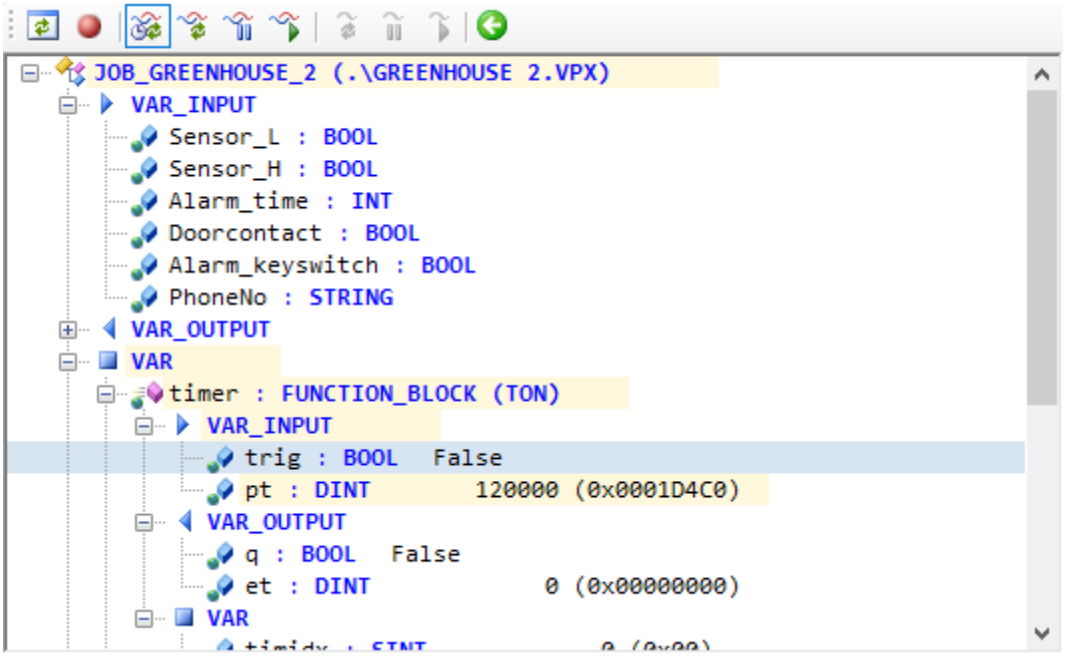
The menu contains the following items:

- Variables
  - Reload: Reloads the symbols from the map file. Must be pressed e.g. after rebuilding the application.
  - Read
  - Edit
  - Find
- Breakpoints
  - Enable breakpoints: When checked, any thread that hits the BREAKPOINT instruction will be paused automatically.
- Threads
  - Auto update: When checked, the thread status will automatically be refreshed every 5s.
  - Update All: Refreshes the status of all the threads.
  - Pause All: Pauses all the threads.
  - Resume All: Resumes all the threads.
  - Update current thread
  - Pause current thread
  - Resume current thread

Toolbar

The toolbar contains buttons for some of the features in the menu.
The toolbar additionally contains the back button, which jumps to the previously selected node. This can also be activated by pressing the back button on the mouse.

Global code view



The global code view shows an overview of all the global variables and other structures in the application. Selecting a node in the tree will show info about the node in the details view on the right (see below).
If the value of a variable or the status of a thread is known, it is shown to the right of the node type.
The background color indicates if the node or its children has changed:

- A yellow background indicates that the newly read value is different than the previous value.
- A light blue background indicates that the variable has local changes that have not been written yet.

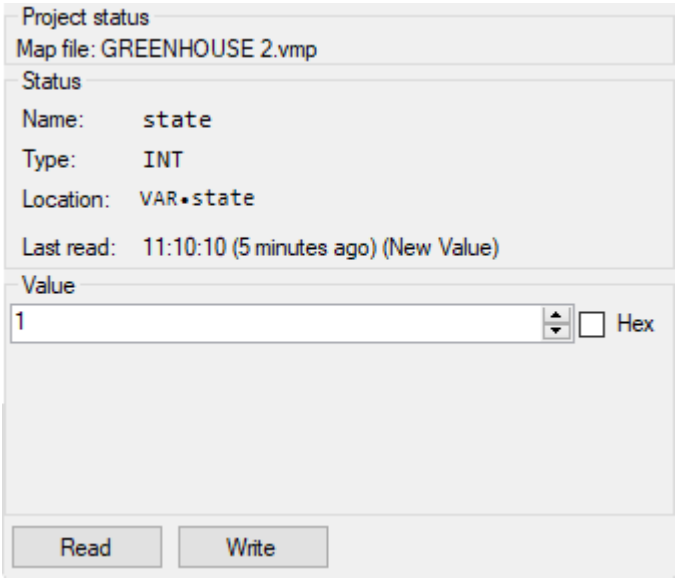
Right-clicking on a node will show a context menu that can contain the following items depending on the type of node:
Variables:

- Read value
- Edit value
- Follow reference
- Collapse Children

Threads:

- Get thread status
- Pause Thread
- Resume Thread
- Show Thread: Finds the selected thread in the thread list and shows it in the thread view below.
- Get Stack: Refreshes the call stack for a paused thread
- Collapse Children

Details view



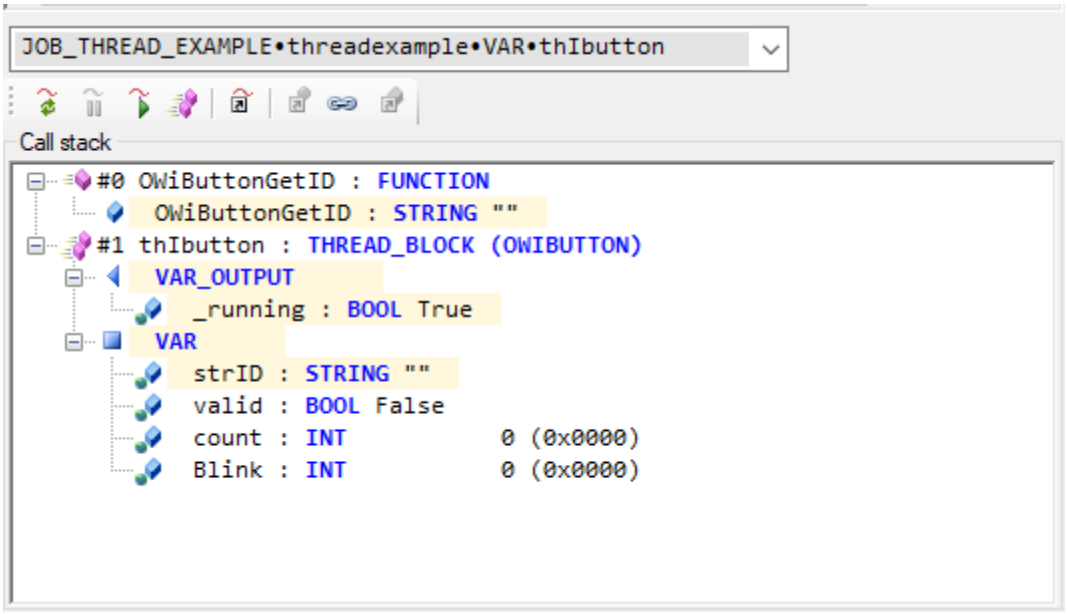
The details view shows the status of the current project and of the currently selected node.

Project status shows the name of map file used for the current project, which contains the metadata needed to show the variables, etc.
The project status will also show if the map file does not match the application running in the device, if e.g. it has been rebuilt since it was transferred to the device, or if the device is running a different application.

Status shows details about the currently selected node.
If the selected node is a variable, it might be possible to view and change the value of it in the Value panel.

The changed value is not written to the device until the write button is pressed or if the edit dialog is used.

Thread View



The drop down list contains all the threads in the application. Selecting a thread will show the status of it below.
The toolstrip below the thread list contains the same options as are available from the thread and variable context menus with one addition:

If the thread is paused, the call stack of the thread is shown as a tree. The call stack will have the most recent call at the top and the calling thread block, program or other global structure at the bottom of the call stack.

Selecting a node will show the details for it in the details view to the right.

In addition to reading and writing global variables in thread blocks, function blocks, etc. it is also possible to read and write local variables of a function on the call stack.

Nested elements, such as a function block inside a thread block can not be edited from the call stack, but must instead be edited from the global variables view. Find global variable or "Synchronize Variables" can be used to find the global variable to edit.

When "Synchronize Variables" is checked, selecting a global variable in the thread call stack will also select it in the code view above, making it easy to e.g. edit variables in nested structures.

The context menu for variables have the following items:

- Read value
- Edit value
- Follow reference
- Collapse Children
- Find global variable

1.3.1.1.1 Insight: Collapse Children

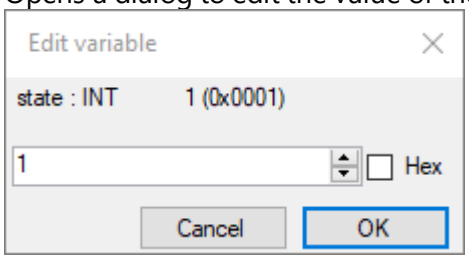
Collapses any open child nodes.

1.3.1.1.2 Insight: Read

Reads the value of the selected node and its children.
Large arrays are not read if they are child nodes.

1.3.1.1.3 Insight: Edit

Opens a dialog to edit the value of the variable:



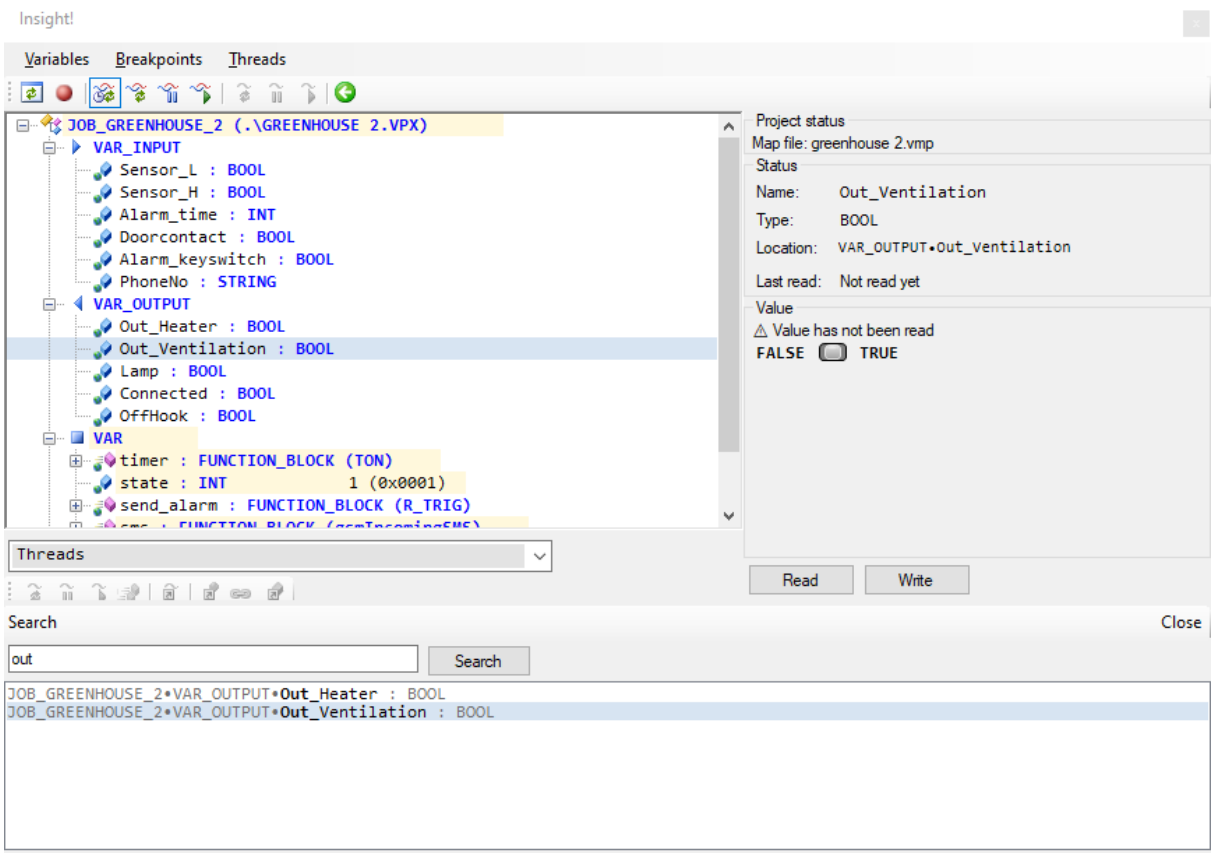
The dialog shows the name and type of the value as well as the current value, if it is known.

The field to enter the value depends on the type of variable.

Clicking OK will write the new value to the device.

1.3.1.1.4 Insight: Find

Opens the search panel below the thread view which is used to search for variables.



Enter a part of the name of the variable to search for in the text box.
Clicking Search or pressing enter will show a list of matching variables.
Double-clicking a result or pressing enter on it will select it in the global code tree.

To close the search panel, click the Close button or press escape while in the search field.

1.3.1.1.5 Insight: Follow reference

For reference types such as PTR and ACCESS variables, this will jump to the referenced variable.

1.3.1.1.6 Insight: Find global variable

Use this when a global variable is selected in the call stack to jump to the same node in the global code tree.
It can not be used on local variables inside functions.

1.3.1.1.7 Insight: Pause thread

Pauses the thread.
If the thread is busy in an internal call, it might take some time to change state to Paused, and will instead have the state Pausing.
Once a thread is paused, the call stack of the thread can be examined.

1.3.1.1.8 Insight: Resume thread

Resumes a thread that has a status of Paused or Pausing.

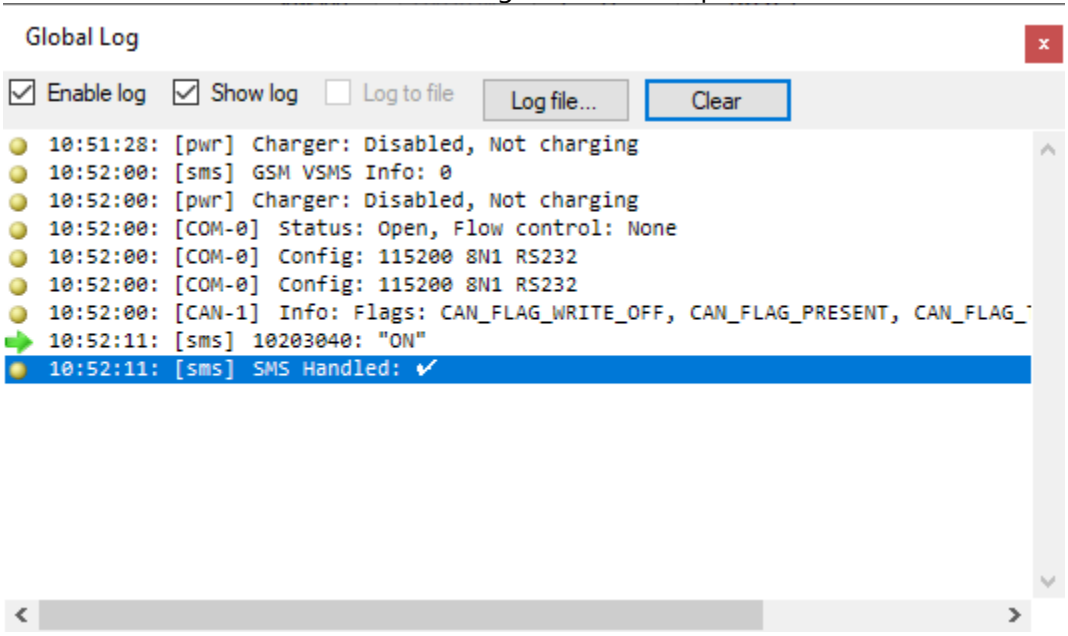
1.3.1.1.9 Insight: Update thread

Refreshes the status of a thread.

If Auto update is enabled, the status is automatically updated every 5 seconds.

1.3.1.2 Global Log

The global log window provides access to viewing an overview of log messages from other modules.
This can be used when the overall timing between multiple modules must be examined.



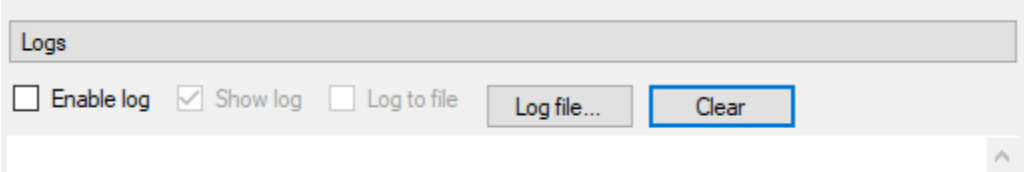
Global log showing log entries from Power supplies and SMS.

See Logs for details about how the control can be used.

1.3.1.2.1 Logs

Many modules contains a log panel, which can show status messages and other info from the module.

Toolbar



The log panel has a toolbar with the following options:

Enable log

Enable this to start logging the information.

Show log

Enable this to show the information in the data list.
It a lot of data is generated, it might be better to log the data to a file instead.

Log to file

Enable this to save the information to the file selected with the "Log File..." button.
If no file is selected, no data will be saved.

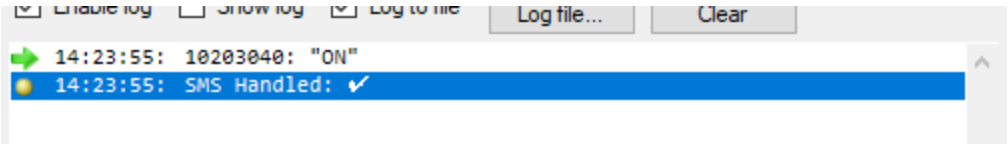
Log file...

Press this button to choose the file to log to.

Clear

Press this button to empty the list of logged data.

Data list



The data in the list can be copied to the clipboard using ctrl+C.
To select all the data, use ctrl+A.
By right-clicking on some types of log entries, it is possible to copy the data in special formats.

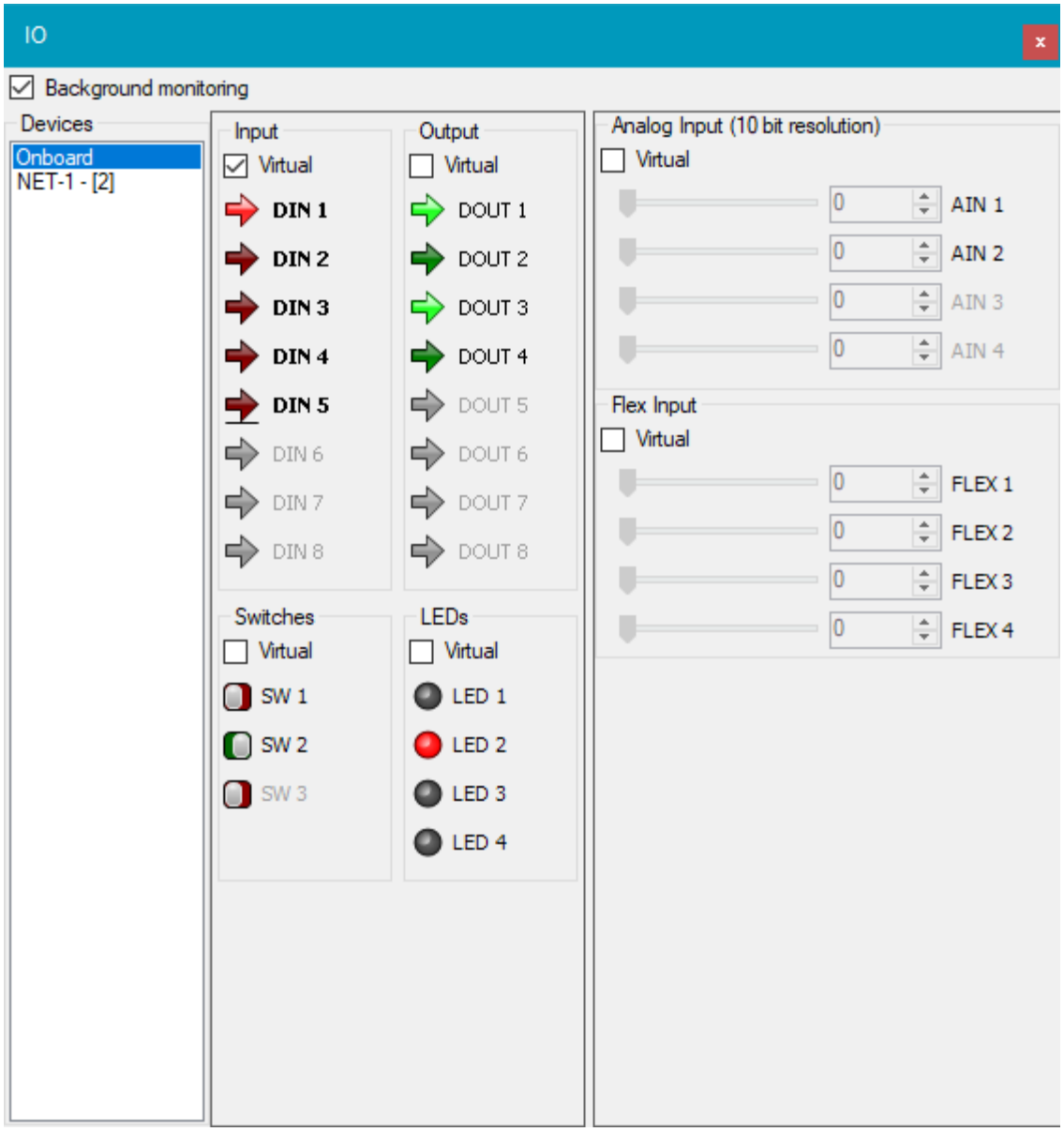
1.3.1.3 IO

The IO window provides access to viewing IO status.
The current IO status is automatically shown while the window is open.

By checking the Virtual check box, the IO can be made virtual:

Virtual inputs can be changed by clicking on the input for digital inputs and switches or by changing the value for analog inputs.
The ignition input is marked by a line under the arrow.

Virtual outputs are not connected to the physical output, meaning that the physical outputs are left at their previous state. This can be useful to prevent external HW from being affected while debugging.

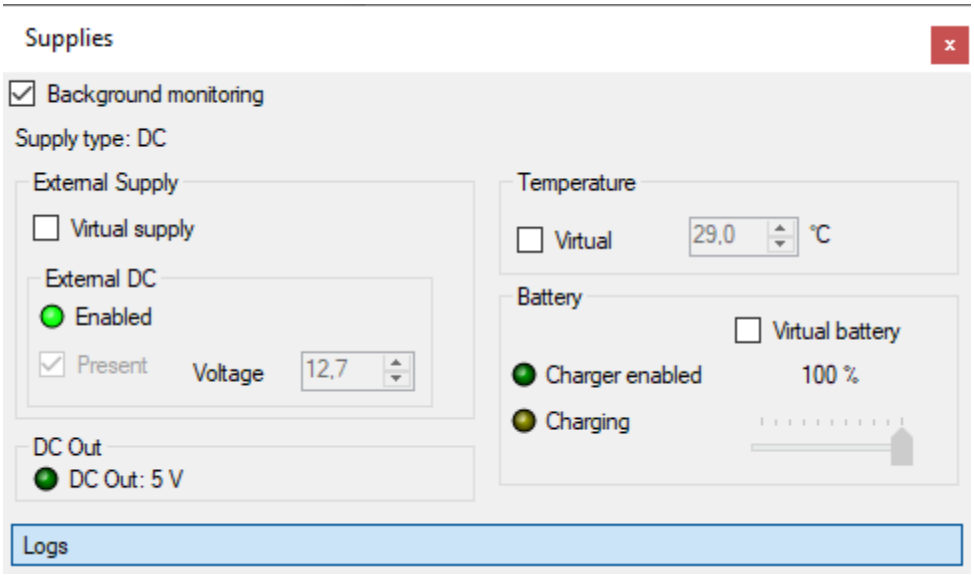


IO window with virtual onboard digital inputs enabled.

Background monitoring
If this is enabled, the data will keep being updated in the background if the window is closed.

1.3.1.4 Power supplies

The Power supplies window provides access to viewing and simulating power supplies and other board features.



The contents of this window depends on the connected device.

The current supply type is shown at the top.

Background monitoring
If this is enabled, the data will keep being updated in the background if the window is closed.

External supply
This groups shows the status of the external AC and/or DC supplies.
Enabling Virtual supply makes it possible to simulate that the external supply is missing and to simulate different voltages on external DC.

DC Out
This group shows the status of any DC out on the device.

Temperature
This shows the current temperature of the device.
Enabling Virtual makes it possible to simulate different device temperatures.

Battery
This shows the current status of the battery and the battery charger.

Enabling Virtual battery makes it possible to simulate a battery with a different charge state. The virtual battery will slowly charge if the charger is enabled.

Logs

Entries are added to the log when any of the values change, e.g. if the battery level or temperature changes or the charger turns on.

See Logs for details about how to use the log control.

Control panel status

Notification

When the supply type changes, a notification is created.

1.3.2 Communication

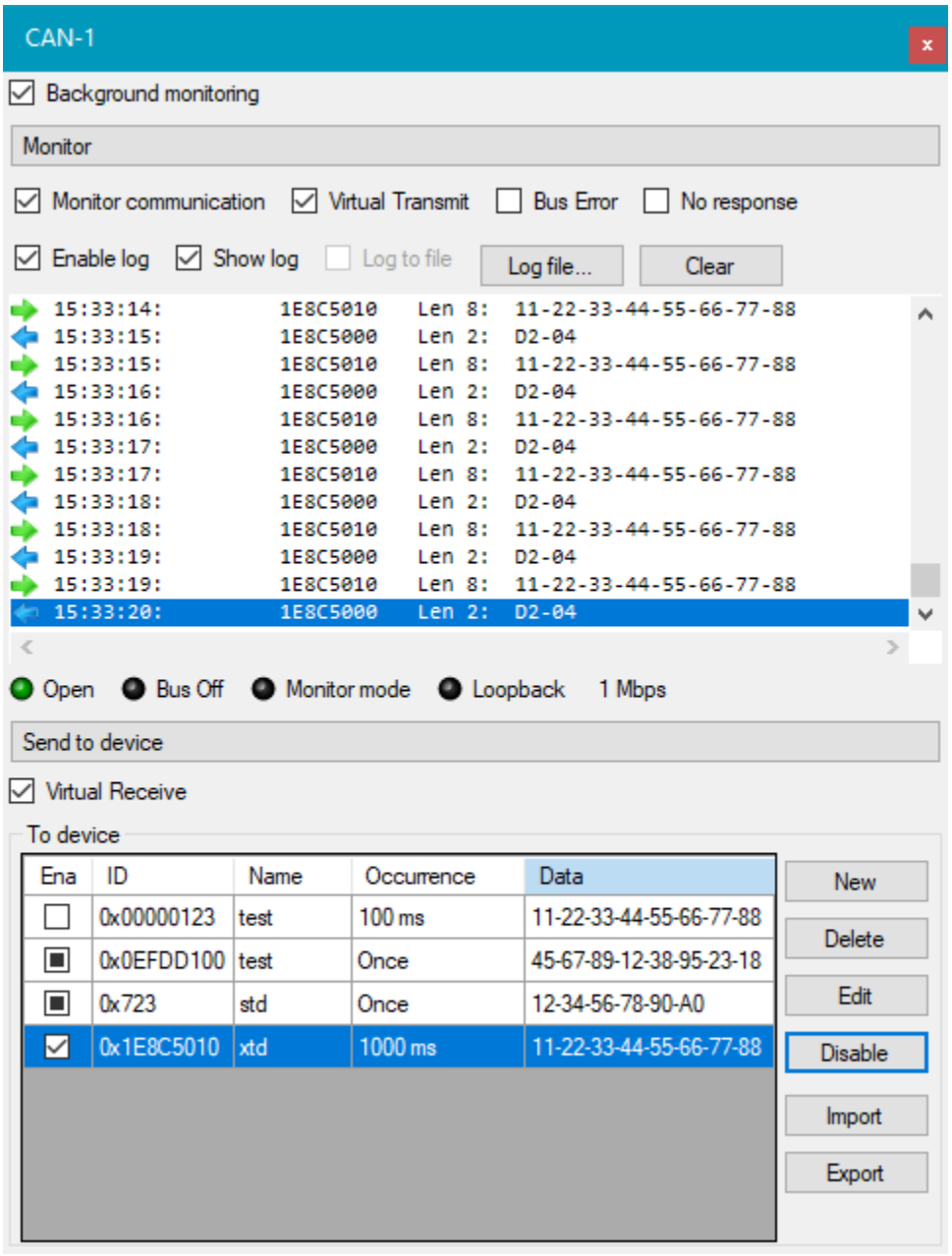
The Communication area contains functionality that is used to communicate over external interfaces.

Modules:

- CAN
- Cellular status
- Serial Ports
- SMS
- USB Host

1.3.2.1 CAN

The CAN window provides access to viewing and simulating CAN communication.



Background monitoring

If this is enabled, the data will keep being updated in the background if the window is closed.

Monitor

The monitor panel is used to view the communication the device sees on the CAN bus and to manipulate what happens when it transmits.

The toolbar has the following options:

Monitor communication
Enable this to capture the messages on the bus and add the to the log. The log does not need to be enabled to be able to add it to the log.

Virtual transmit
When this is enabled, messages sent with canSendMessage are not sent to the real bus but are handled internally, with the option of simulating some error conditions:

Bus Error
Simulate a bus error when transmitting.

No response
Simulate that there is not response to the message.

Below the toolbar is a log showing the messages, see Logs for more details about how it can be used.

The status of the bus is shown below the log.

Send to device
Enabling Virtual Receive makes it possible to send virtual messages to the device.

To device
The "To device" group is used to manage the CAN messages that are available for being transmitted to the device.
An existing CAN message can be edited by double-clicking on it.

Columns:

- Ena

This column shows if automatic sending of the message is enabled.

- ID

This column contains the identifier of the message.
The identifier is displayed as a hexadecimal number.

- Name

This column contains the name of the message.
Naming the messages is not required but can make it easier to recognize what it represents.

- Occurrence

This column contains the frequency with which the message will be sent to the device once it has been enabled.

- Data

This column contains the data bytes of the message.
The bytes are displayed as hexadecimal numbers.

New
This button will create a new message.
The CAN message dialog (see below) is used to configure the new message.

Delete
This button will remove the selected message.

Send / Enable / Disable
This button will show a different title dependent on the occurrence of the selected message.

When the selected message has an occurrence of one, the button will show "Send", and the message will be sent to the device when pressed.

When the selected message is disabled, the button will show "Enable", and the message will be enabled when pressed. An enabled message will be sent to the device repeatedly with the configured interval.

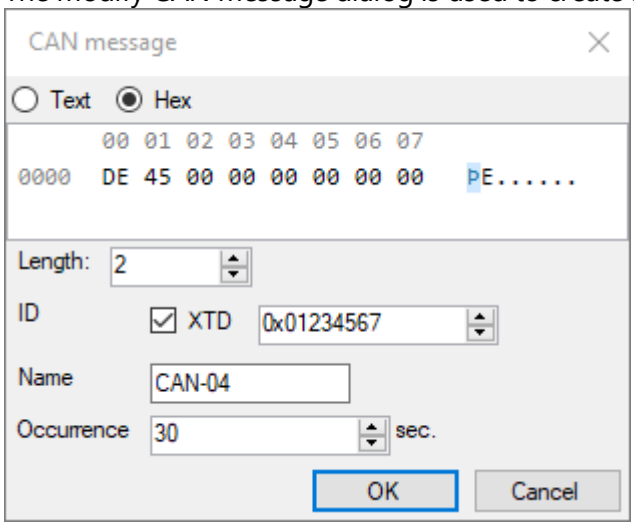
When the selected message is enabled, the button will show "Disable", and the message will be disabled when pressed. A disabled message will no longer be sent to the device.

Import
This button will import a previously exported collection of messages and add it to the list.

Export
This button will export the messages from the group as a collection of messages.

The CAN message collection is stored in a human readable XML file.

Modify CAN message
The modify CAN message dialog is used to create and modify CAN messages.



The data can be entered either as text or as hexadecimal data.

Length
The length of the data in bytes.

ID
This is the message identifier. If XTD is selected, it is an extended identifier.

Name
The name of the message. It is not required, but can be helpful to keep track of the messages.

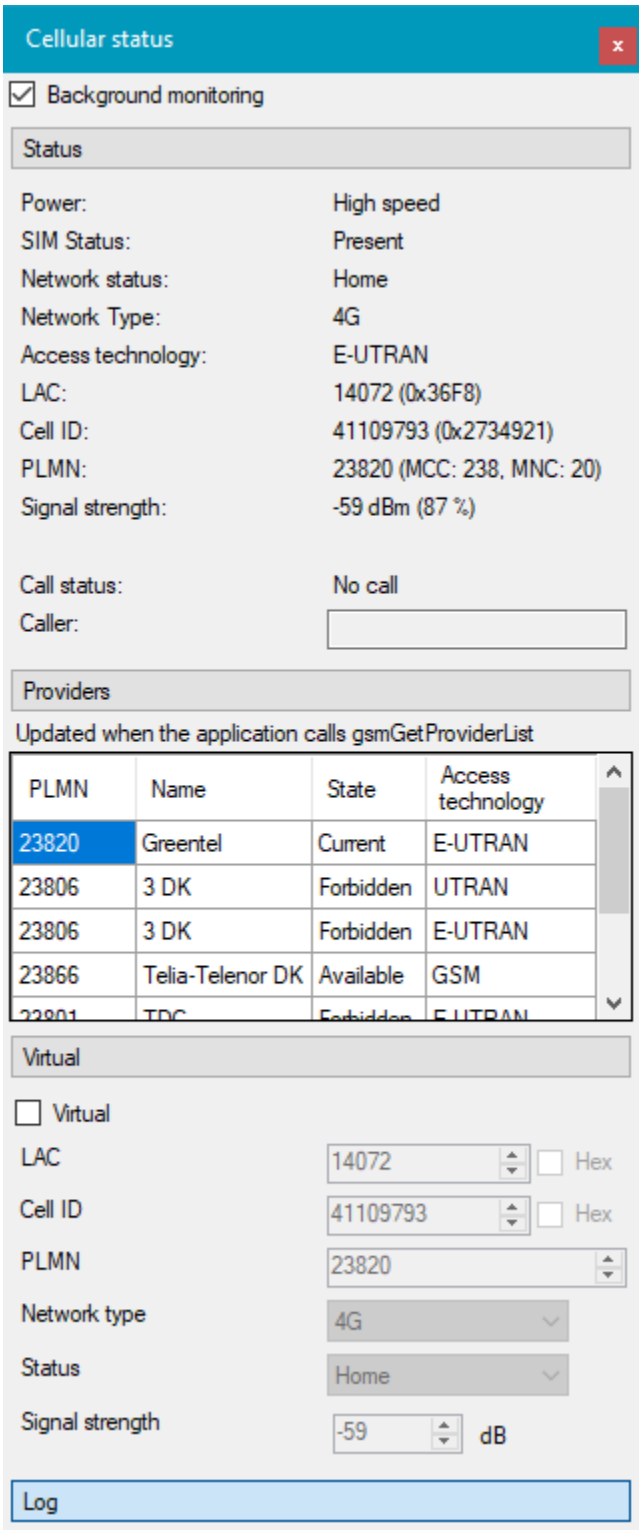
Occurrence
The occurrence determines if the message must be manually sent or if it automatically is sent at the specified interval, e.g. setting the occurrence to 120 seconds will send the message every two minutes.

Control panel status
Notification
When a message is sent or received, a notification is created.

Control panel status:
Green:	Port open
Red:	Bus off

1.3.2.2 Cellular Status

The Cellular window provides access to viewing and simulating the cellular status.



Background monitoring
If this is enabled, the data will keep being updated in the background if the window is closed.

Status

The status panel shows the status of the GSM module, the SIM card, the cellular network and the call status.

Providers

The provider panel shows the result of gsmGetProviderList.

Virtual

Enabling Virtual makes it possible to simulate different network types, signal strength and network IDs.
These virtual parameters will only be used while the real network is connected.
When virtual parameters are used, the network status will show "(Virtual)" after the status.

Log

Entries are added to the log when a change is detected.

See Logs for details about how to use the log control.

Control panel status

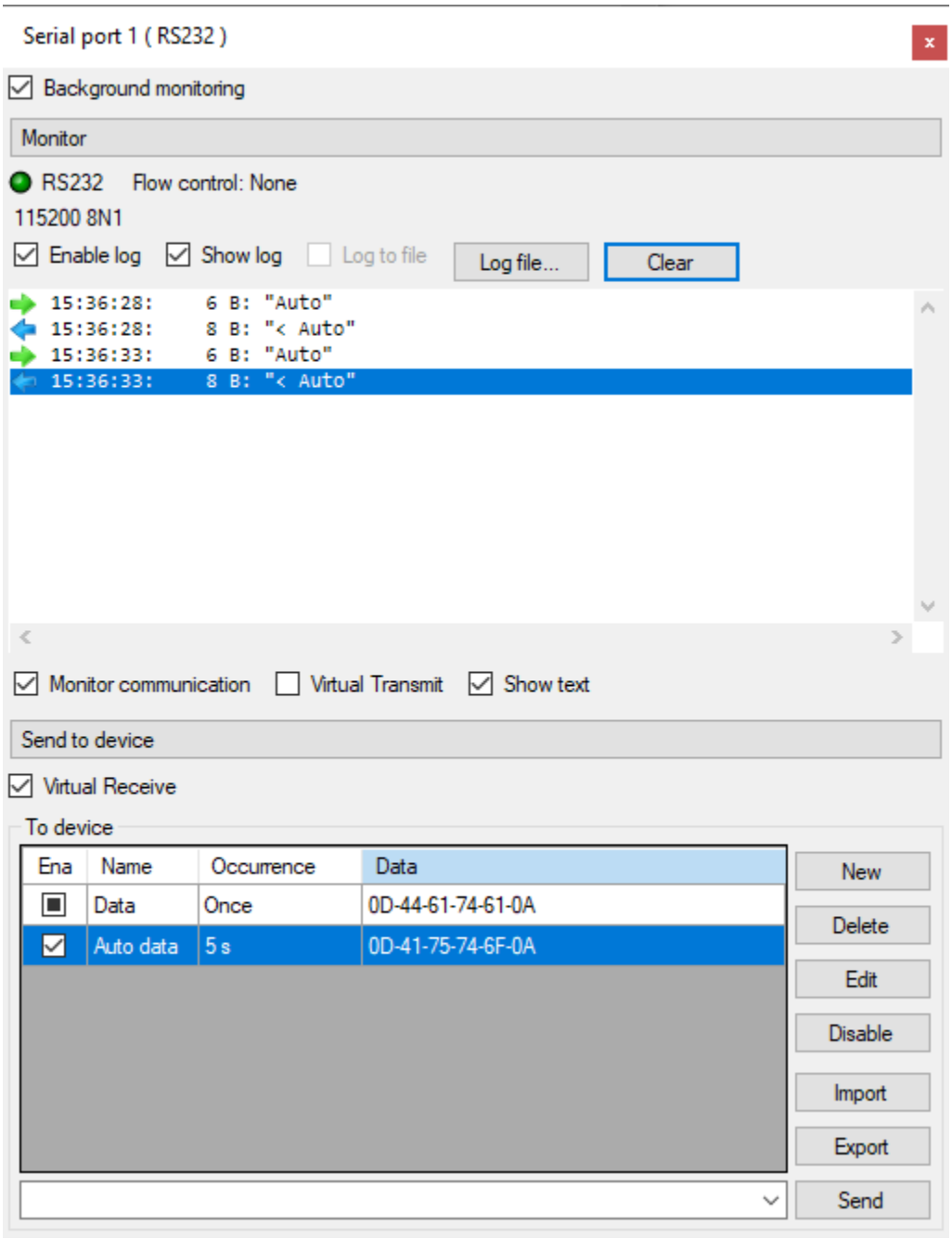
Control panel status:

Network status:

Green:	Home network
Blue:	Roaming
Yellow:	Searching or turning on
Red:	Denied

1.3.2.3 Serial ports

The Serial port window provides access to viewing and simulating serial communication using RS232 or RS485.



Background monitoring

If this is enabled, the data will keep being updated in the background if the window is closed.

Monitor

The monitor panel is used to view the communication that happens on the serial port.
The current status of the port is shown at the top.

Below the status is a log showing the sent and received packets, see Logs for more details about how it can be used.

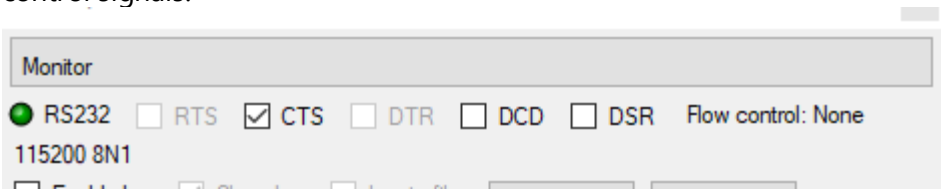
Below the log is a toolbar with the following options:

Monitor communication

Enable this to capture the data on the port and add the to the log. The log does not need to be enabled to be able to add it to the log.

Virtual control signals

Checking this on a port with controls signals, makes it possible to manually set the state of the control signals:



Virtual transmit

When this is enabled, data is not sent to the real port and is only added to the log.

Show text

When this is checked, it will try to format the data as text if possible.

Send to device

Enabling Virtual Receive makes it possible to send virtual data to the device.

To device

The "To device" group is used to manage the data that is available to be transmitted to the device.
An existing packet can be edited by double-clicking on it.

Columns:

- Ena

This column shows if automatic sending of the packet is enabled.

- Name

This column contains the name of the packet.
Naming the packets is not required but can make it easier to recognize what it represents.

- Occurrence

This column contains the frequency with which the packet will be sent to the device once it has been enabled.

- Data

This column contains the data of the packet.

New

This button will create a new packet.
The Packet dialog (see below) is used to configure the new packet.

Delete
This button will remove the selected packet.

Send / Enable / Disable
This button will show a different title dependent on the occurrence of the selected packet.

When the selected packet has an occurrence of one, the button will show "Send", and the packet will be sent to the device when pressed.

When the selected packet is disabled, the button will show "Enable", and the packet will be enabled when pressed. An enabled packet will be sent to the device repeatedly with the configured interval.

When the selected packet is enabled, the button will show "Disable", and the packet will be disabled when pressed. A disabled packet will no longer be sent to the device.

Import
This button will import a previously exported collection of packets and add it to the list.

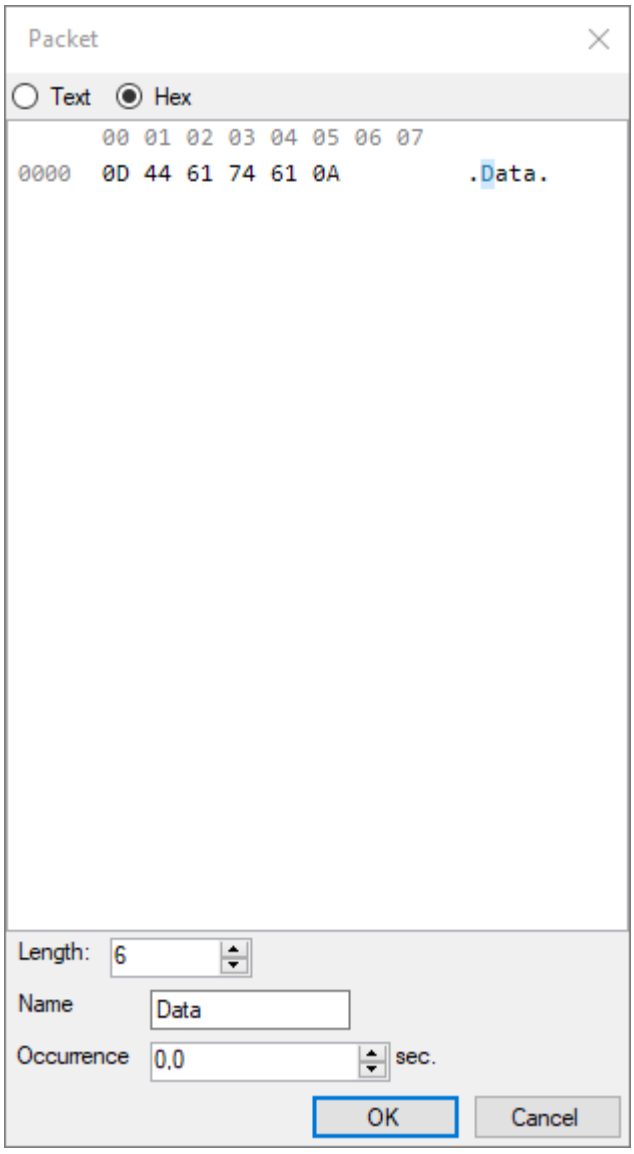
Export
This button will export the packets from the group as a collection of packets.

The packet collection is stored in a human readable XML file.

Sending text directly
The text box and the Send button are used to write data to the serial port without having to create a packet for it.
To write special characters, e.g. start- and end-of-frame characters, the following strings can be used:

String	Result
$$	$
$"	"
$L	Linefeed (value 10)
$P	Form feed (value 12)
$R	Carriage return (value 13)
$T	Tab (value 9)
$N	Newline (value 13, value 10)
$xx	Value xx (xx in hex, 00..FF)

Modify packet
The modify packet dialog is used to create and modify packets.



The data can be entered either as text or as hexadecimal data.

Length
The length of the packet in bytes.

Name
The name of the packet. It is not required, but can be helpful to keep track of the packets.

Occurrence
The occurrence determines if the packet must be manually sent or if it automatically is sent at the specified interval, e.g. setting the occurrence to 120 seconds will send the packet every two minutes.

Control panel status

Notification

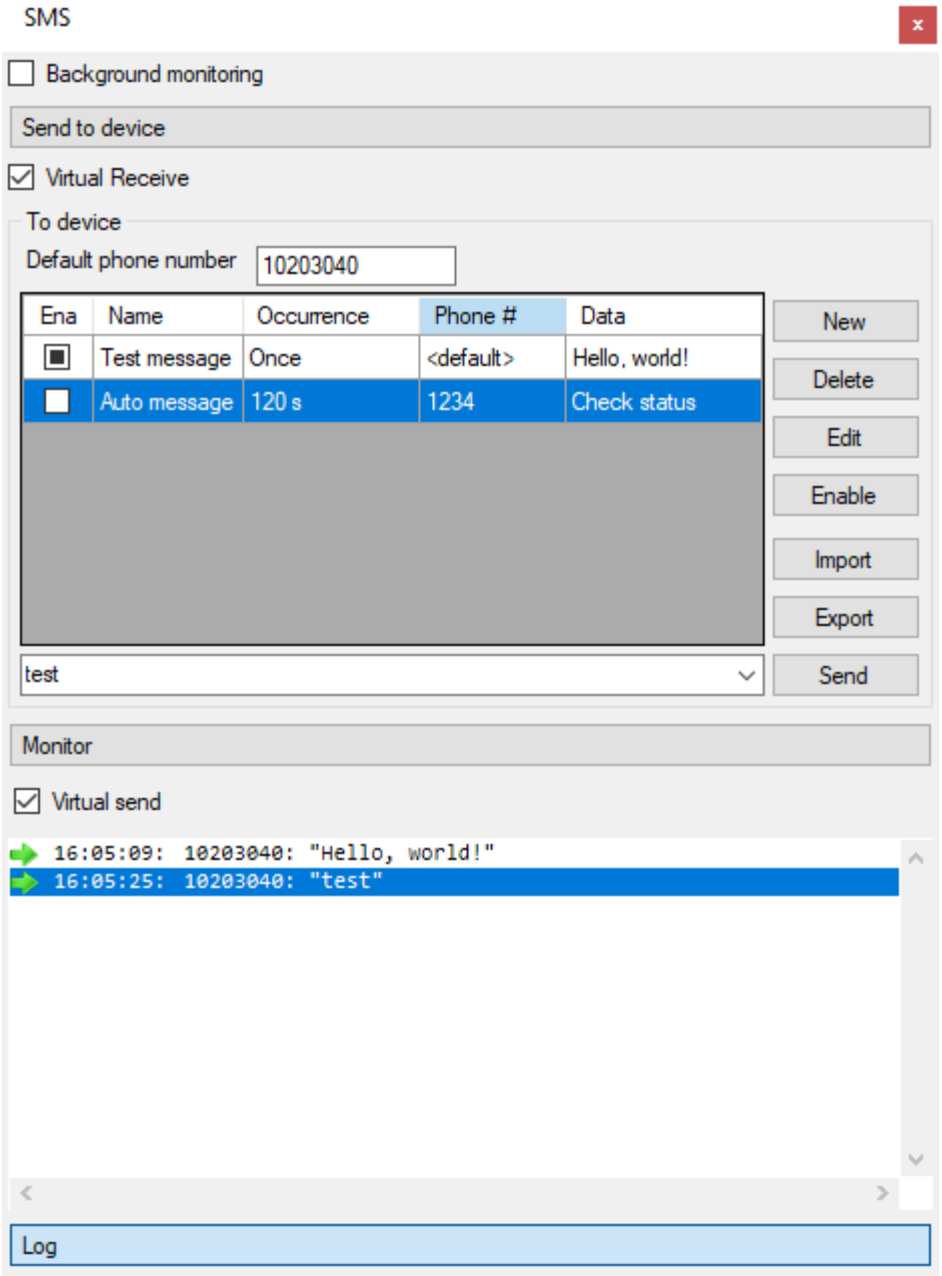
When data is sent or received, a notification is created.

Control panel status:

- Green: Port open

1.3.2.4 SMS

The SMS module is used to view SMS messages and to create virtual SMS messages.



Background monitoring

If this is enabled, the data will keep being updated in the background if the window is closed.

Send to device

Enabling Virtual Receive makes it possible to send virtual SMS messages to the device.

To device
The "To device" group is used to manage the messages that are available to be transmitted to the device.
An existing message can be edited by double-clicking on it.

Default phone number
This is the phone number to use by default.

Columns:
- Ena

This column shows if automatic sending of the message is enabled.
- Name

This column contains the name of the message.
Naming the messages is not required but can make it easier to recognize what it represents.
- Occurrence

This column contains the frequency with which the message will be sent to the device once it has been enabled.
- Phone #

The Phone number to use for the message.
- Data

This column contains the content of the message.

New
This button will create a new message.
The message dialog (see below) is used to configure the new message.

Delete
This button will remove the selected message.

Send / Enable / Disable
This button will show a different title dependent on the occurrence of the selected message.

When the selected message has an occurrence of one, the button will show "Send", and the message will be sent to the device when pressed.

When the selected message is disabled, the button will show "Enable", and the message will be enabled when pressed. An enabled message will be sent to the device repeatedly with the configured interval.

When the selected message is enabled, the button will show "Disable", and the message will be disabled when pressed. A disabled message will no longer be sent to the device.

Import
This button will import a previously exported collection of messages and add it to the list.

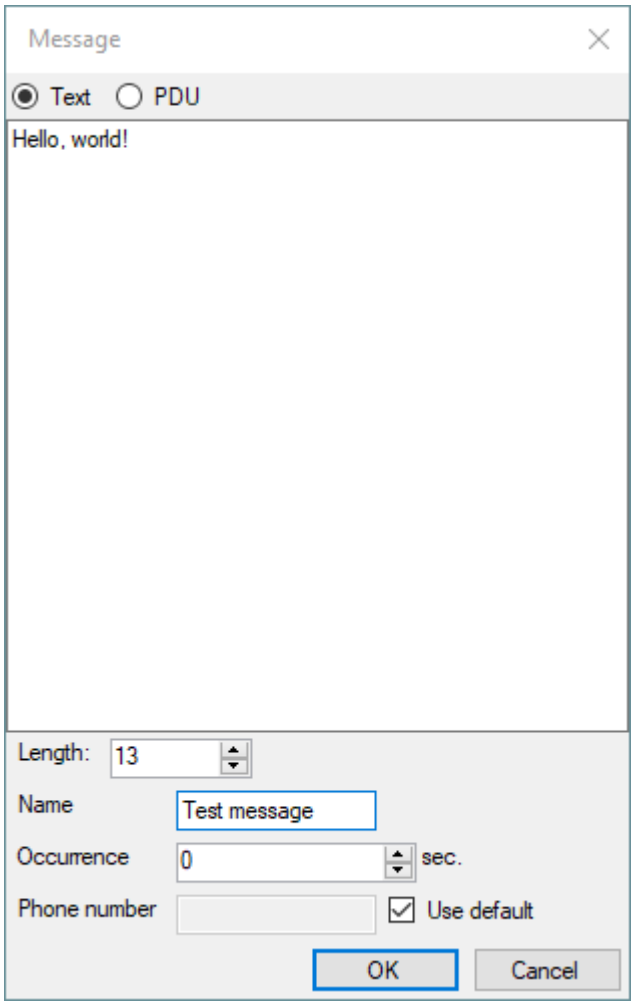
Export
This button will export the messages from the group as a collection of messages.

The message collection is stored in a human readable XML file.

Sending text directly
The text box and the Send button are used to send an SMS message without having to first create a message.
To write special characters, e.g. start- and end-of-frame characters, the following strings can be used:

String	Result
$$	$
$"	"
$L	Linefeed (value 10)
$P	Form feed (value 12)
$R	Carriage return (value 13)
$T	Tab (value 9)
$N	Newline (value 13, value 10)
$xx	Value xx (xx in hex, 00..FF)

Modify message
The modify message dialog is used to create and modify messages.



The data can be entered either as text or as a binary PDU, represented as hexadecimal data.

Length
The length of the message.

Name
The name of the message. It is not required, but can be helpful to keep track of the messages.

Occurrence
The occurrence determines if the message must be manually sent or if it automatically is sent at the specified interval, e.g. setting the occurrence to 120 seconds will send the message every two minutes.

Phone number

If the "Use default" check box is checked, the messages will be sent from the default phone number provided on the "To device" panel, otherwise the phone number provided in the "Phone number" text box will be used.

Monitor

The monitor panel is used to view the sent and received SMS messages.

Virtual send

When this is enabled, the device will not really send SMS messages, but just add them to the log.

The log shows the sent and received messages, see Logs for more details about how it can be used.

Log

Entries are added to the log when an SMS messages is sent or received.

See Logs for details about how to use the log control.

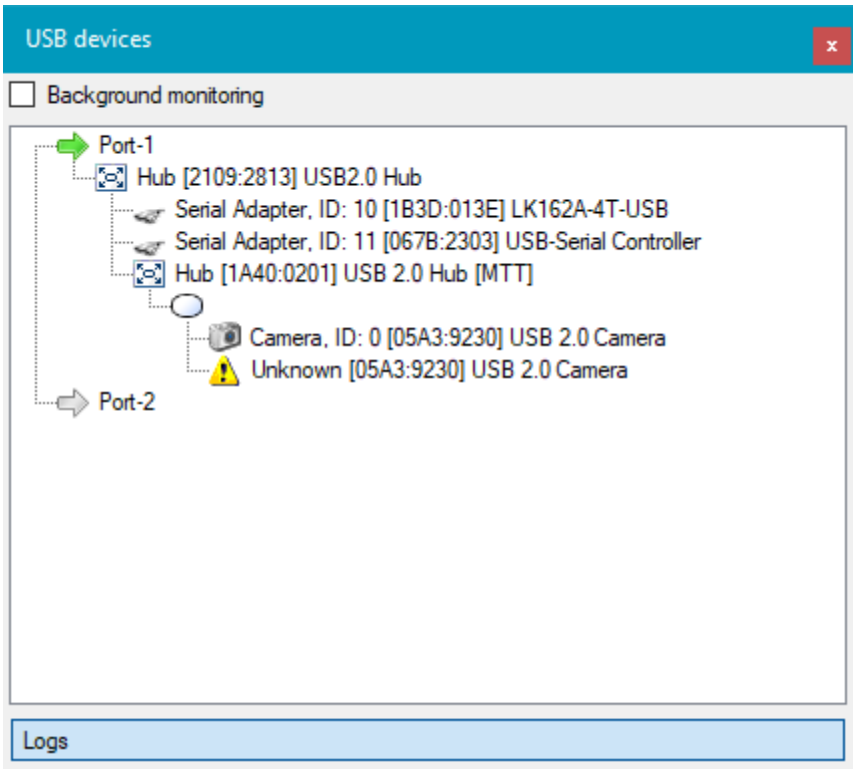
Control panel status

Notification

When an SMS is sent or received, a notification is created.

1.3.2.5 USB host

The USB host window provides access to viewing the status of the USB ports and any connected devices.



Background monitoring
If this is enabled, the data will keep being updated in the background if the window is closed.

Ports
Each USB Host port is represented by an arrow. If the port is powered on, the arrow is green, otherwise it is gray.

The hierarchy of the connected devices is show under each port.
Known devices are indicated with an icon depending on the type of device.
USB Hubs and composite devices are also shown.

Double-clicking on a device will show further details about the device.

Logs
Entries are added to the log when a USB devices are found or removed.

See Logs for details about how to use the log control.

Control panel status
Control panel status:
- Green: A port is open

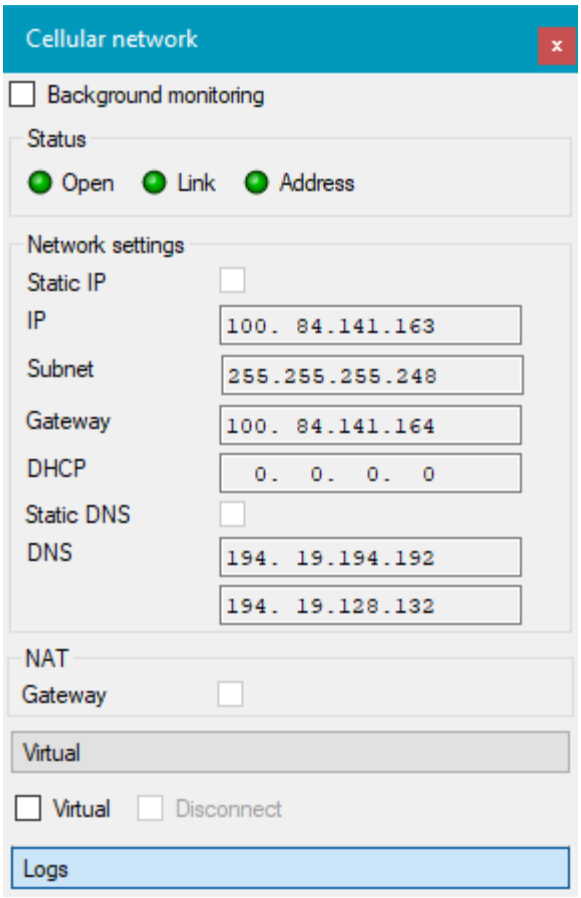
1.3.3 Network

The Network area contains functionality to work with the network interfaces.

Modules:
- Cellular Network
- LAN
- WLAN
- Access Point
- RCH

1.3.3.1 Cellular Network

The Cellular network window provides access to viewing the status of the cellular network and to simulate disconnection.



Background monitoring
If this is enabled, the data will keep being updated in the background if the window is closed.

Status

This section shows the current network status.

Network Settings

This section shows the current IP and DNS status.

NAT

This section shows if this interface is configured as NAT gateway.

Virtual

Enable Virtual and Disconnect to simulate that the network is disconnected.

Logs

Entries are added to the log when the status changes.

See Logs for details about how to use the log control.

Control panel status

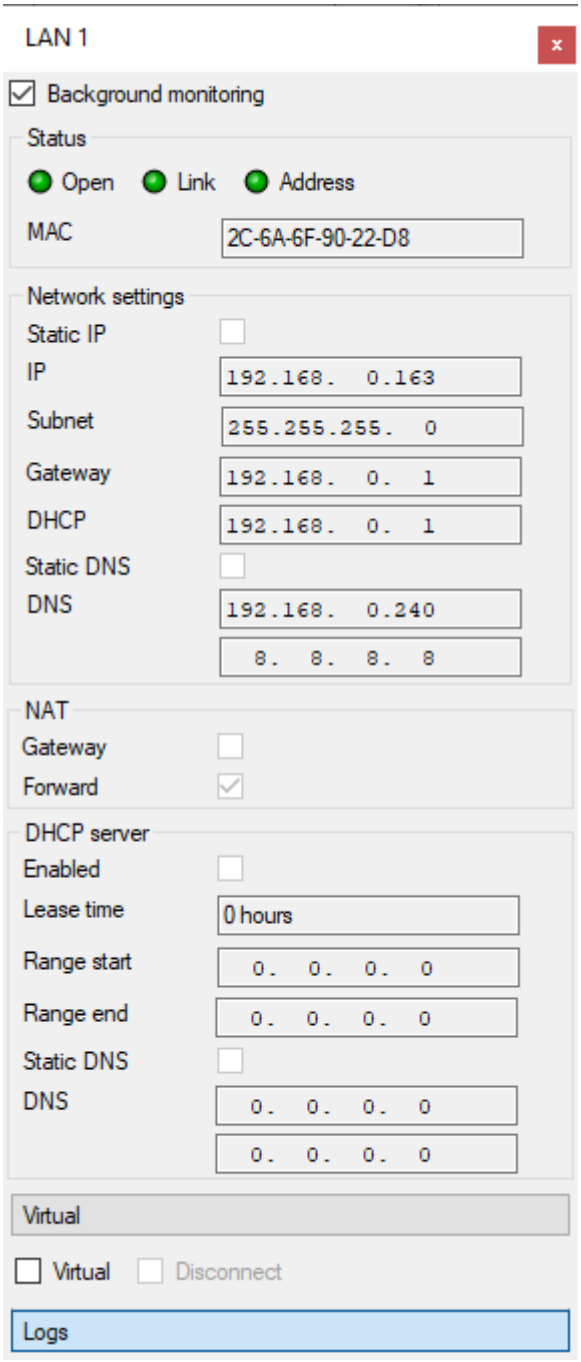
Control panel status:

Network status:

Red:	Network open
Yellow:	Link established
Green:	Connected

1.3.3.2 LAN

The LAN window provides access to viewing the status of the LAN networks and to simulate disconnection.
LAN 1 is used for the onboard LAN interface and LAN 2 is used for the optional USB LAN interface.



Background monitoring

If this is enabled, the data will keep being updated in the background if the window is closed.

Status

This section shows the current network status and MAC address.

Network Settings

This section shows the current IP and DNS status.

NAT

This section shows if this interface is configured as NAT gateway or if the interface is forwarded to the NAT gateway interface.

DHCP server

This section shows the configuration of the DHCP server for the interface.

Virtual

Enable Virtual and Disconnect to simulate that the network is disconnected.

Logs

Entries are added to the log when the status changes.

See Logs for details about how to use the log control.

Control panel status

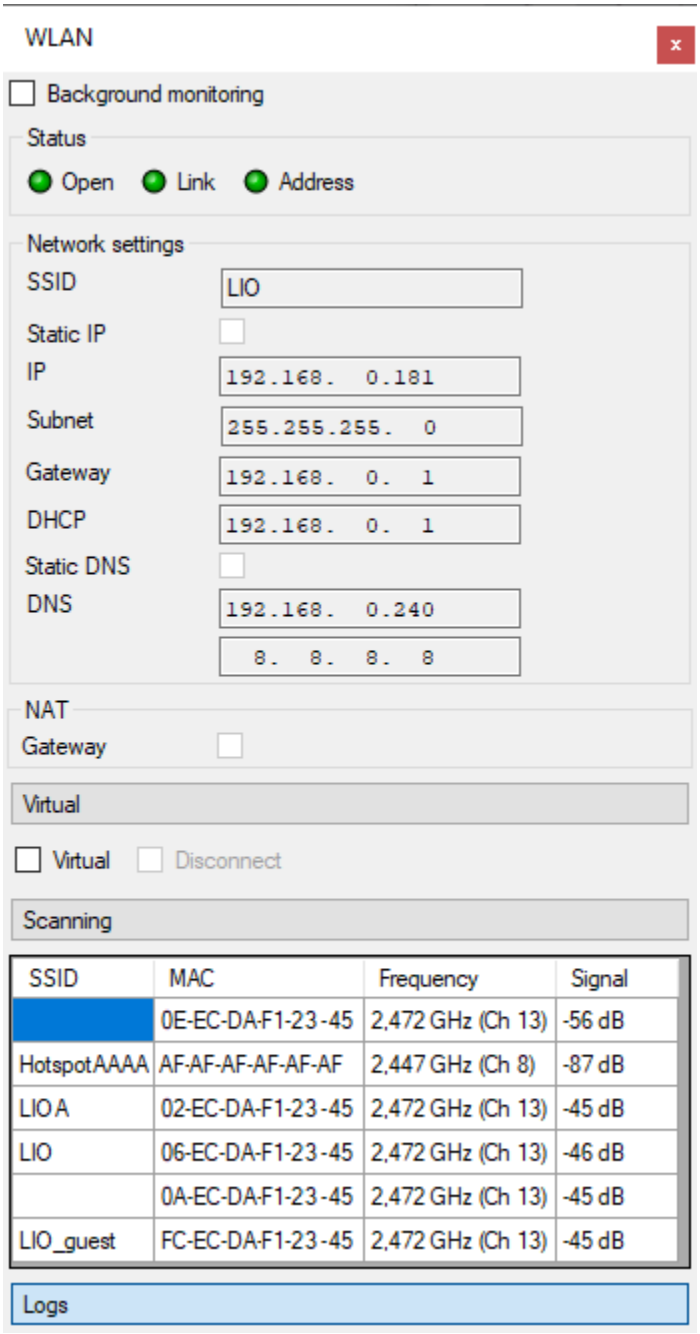
Control panel status:

Network status:

Red:	Network open
Yellow:	Link established
Green:	Connected

1.3.3.3 WLAN

The WLAN window provides access to viewing the status of the WLAN network interface, the available SSIDs and to simulate disconnection.



Background monitoring

If this is enabled, the data will keep being updated in the background if the window is closed.

Status

This section shows the current network status.

Network Settings

This section shows the current IP and DNS status.

NAT

This section shows if this interface is configured as NAT gateway.

Virtual

Enable Virtual and Disconnect to simulate that the network is disconnected.

Scanning

This section contains a list of SSIDs found in by the device.
The list is updated automatically.

Logs

Entries are added to the log when the status changes.

See Logs for details about how to use the log control.

Control panel status

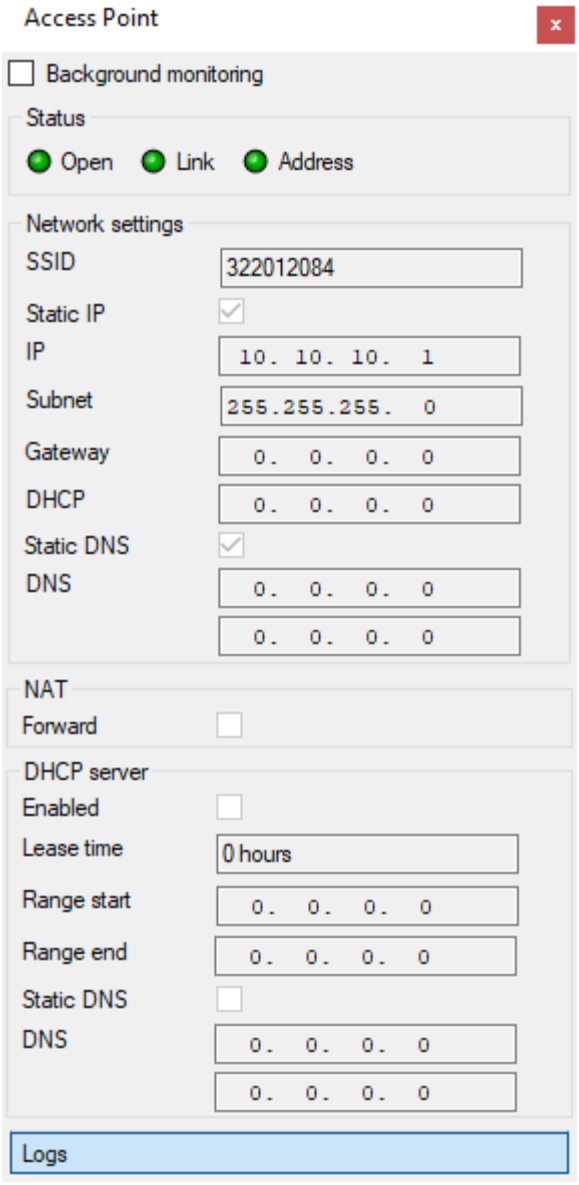
Control panel status:

Network status:

Red:	Network open
Yellow:	Link established
Green:	Connected

1.3.3.4 Access Point

The Access Point window provides access to viewing the status of the access point interface.



Background monitoring

If this is enabled, the data will keep being updated in the background if the window is closed.

Status

This section shows the current network status and MAC address.

Network Settings

This section shows the current IP and DNS status.

NAT

This section shows if this interface is configured to be forwarded to the NAT gateway interface.

DHCP server

This section shows the configuration of the DHCP server for the interface.

Virtual

Enable Virtual and Disconnect to simulate that the network is disconnected.

Logs

Entries are added to the log when the status changes.

See Logs for details about how to use the log control.

Control panel status

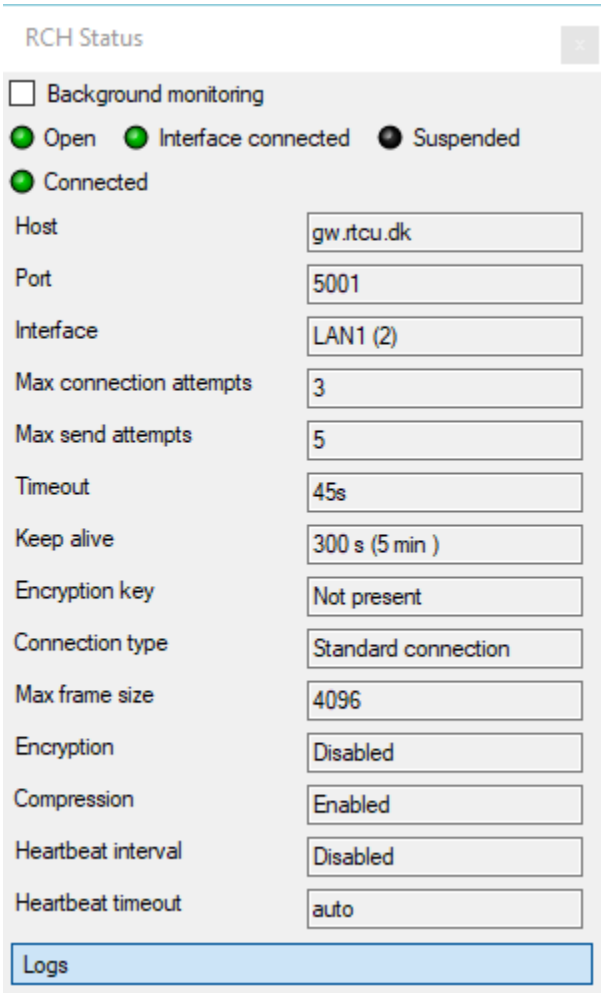
Control panel status:

Network status:

Red:	Network open
Yellow:	Link established
Green:	Connected

1.3.3.5 RCH

The RCH window provides access to viewing the status of the connection to the RTCU Communication Hub.



Background monitoring

If this is enabled, the data will keep being updated in the background if the window is closed.

The connection status is shown at the top.
The currently used connection parameters are shown below.

Logs

Entries are added to the log when the status changes.

See Logs for details about how to use the log control.

Control panel status

Control panel status:

Red:	Interface not available
Yellow:	Interface connected
Green:	Connected
Blue:	Connection is suspended.

1.3.4 Sensors

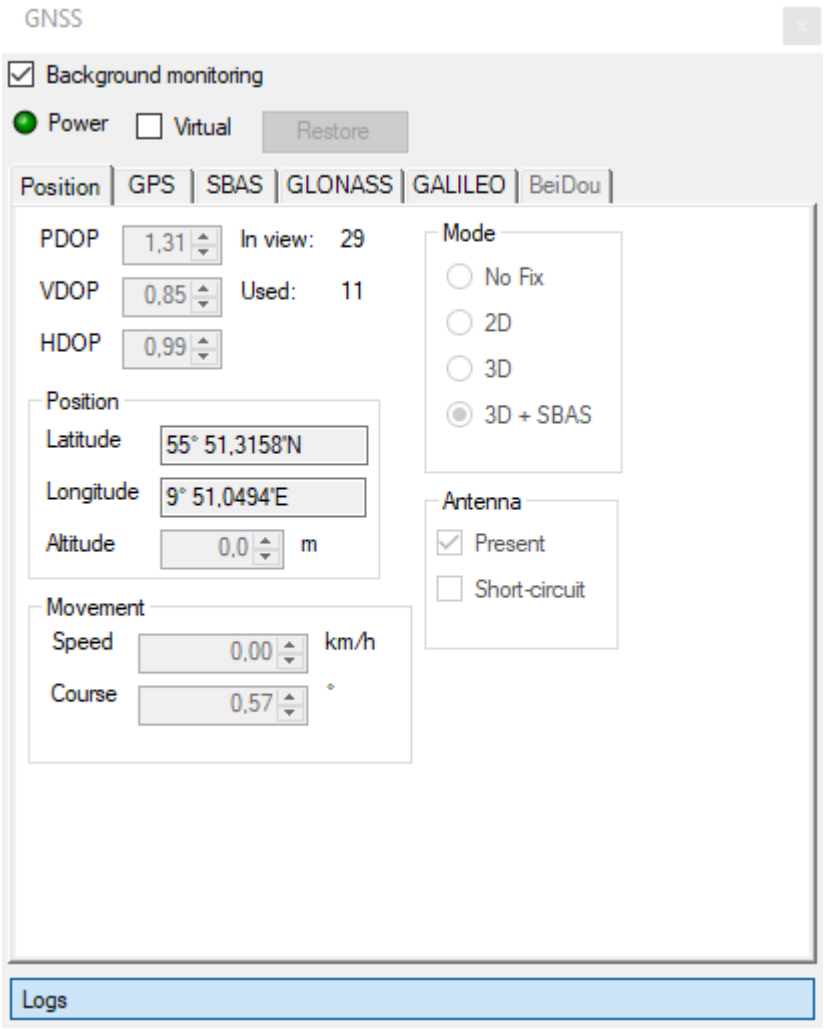
The Sensors area contains functionality to work with sensors that fetch data from the environment.

Modules:

- GNSS

1.3.4.1 GNSS

The GNSS window provides access to viewing and simulating the GNSS status.



Background monitoring
If this is enabled, the data will keep being updated in the background if the window is closed.

Status

The top of the window provides the current status of the GNSS module.

When virtual is enabled, the device will be using the position, mode and other data specified in the tabs below, instead of using the real data, making it possible to simulate different conditions without having to move the device.

The restore button is used to set the virtual values to a reasonable default.

Position tab

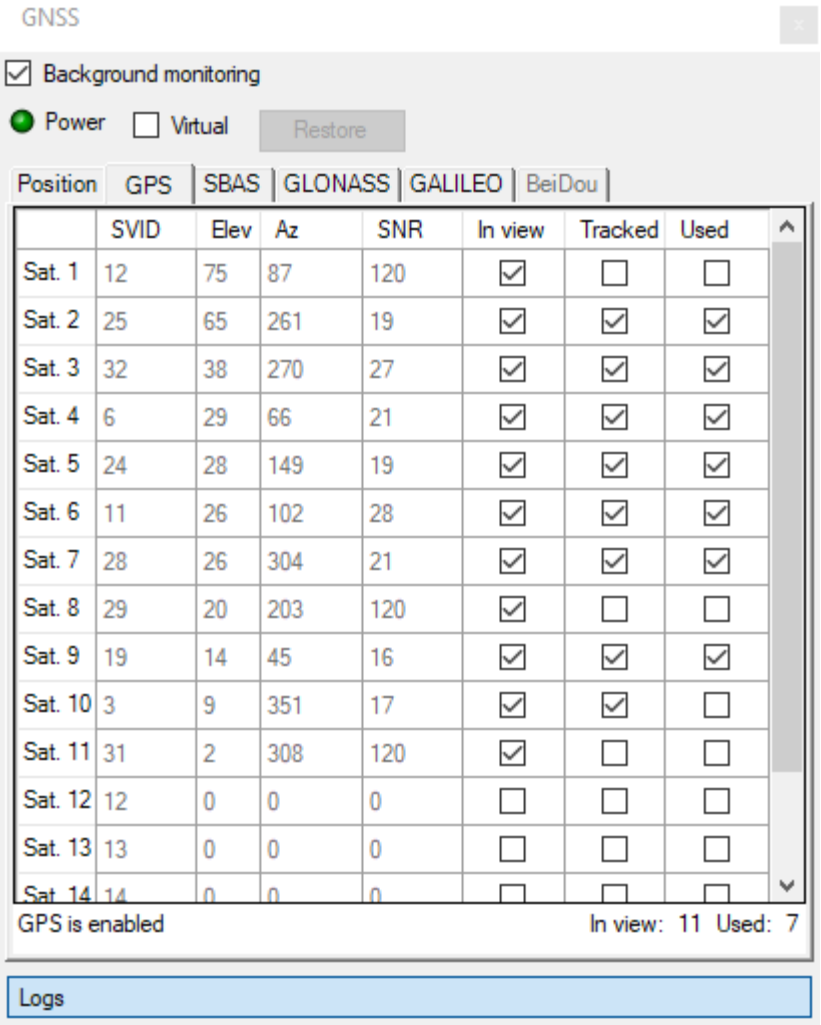
The position quality is available in the PDOP, VDOP and HDOP fields.
The "in view" and "used" numbers are based on the reported satellites as shown on the satellite tabs.
The position is available in the Latitude, Longitude and Altitude fields.
The movement speed and direction are available in the Speed and Course fields.

When virtual is enabled, the fields can be edited, allowing a virtual position to be specified with the wanted accuracy and movement.

The mode status reports the current fix mode. When virtual is enabled, this can be used to specify which mode the device should report.

The antenna status reports the status of the antenna. When virtual is enabled, this can be used to test short-circuit and a missing antenna.

Satellite tabs



The satellite tabs provide access to viewing the first 16 visible satellites for the different satellite systems and SBAS.
When virtual is enabled, it is possible to manually specify the position of the satellite, the signal-to-noise ratio and if it is visible, tracked and used.
The position does not affect the quality of the fix and it is not necessary to specify any used satellites to be able to report a valid virtual position.

Logs

Entries are added to the log when the antenna status changes on supported devices.
If "Log NMEA sentences" is enabled, the raw NMEA sentences will be added to the log for further analysis.

See Logs for details about how to use the log control.

Control panel status

Notification

When the antenna status changes, a notification is created.

Control panel status:

- Red: No fix
- Yellow: 2D fix
- Green: 3D fix
- Blue: SBAS or DR fix

Tutorial

2 Tutorial

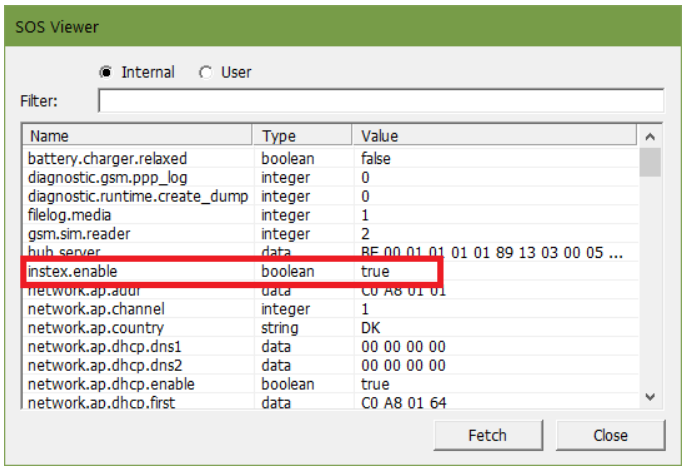
This tutorial provides a quick tour of some of the features of the RTCU IEX program.
It uses the well-known Greenhouse 2 example project which is included with the RTCU IDE as basis.

The tutorial consists of the following chapters:

2.1 Chapter 1: Setup

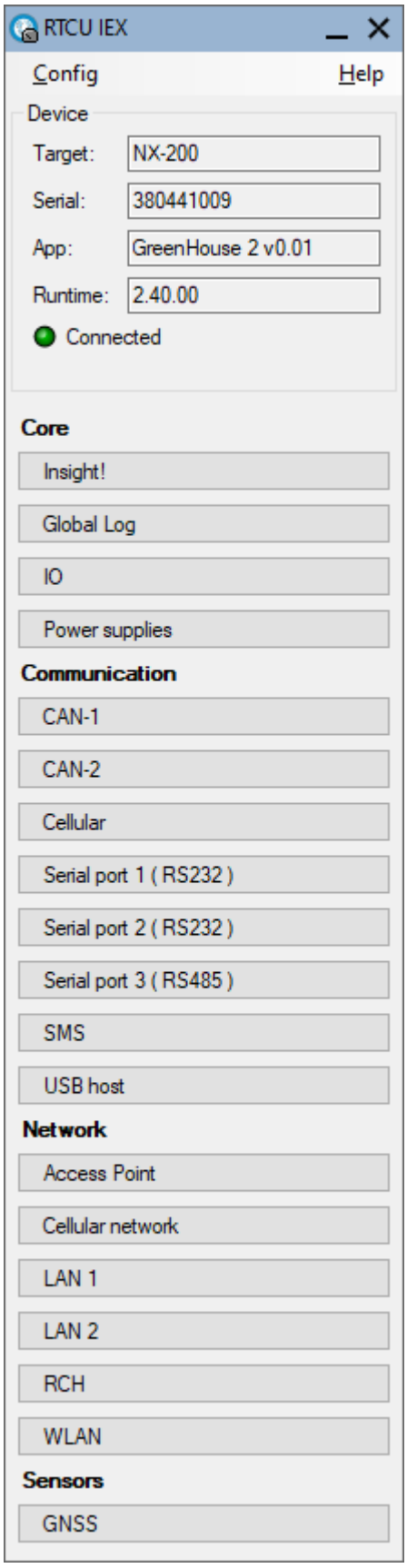
This tutorial assumes that both the **RTCU IDE** and **RTCU IEX** are already installed and that an RTCU device that is running firmware V2.40.00 or newer is connected to the **RTCU IDE**. The device must contain a valid SIM card to be able to establish cellular connection.

The device must be configured to use IEX
To do so, please go to [Device]-[Data]-[SOS Viewer] and locate the "instex.enable" variable. Set it to TRUE:



The device will be reset later, when the application has been transferred, which will make the setting take effect.

1. Open the "Greenhouse 2" example project with the RTCU IDE.
   It can normally be found at "C:\Users\<user>\Documents\RTCU Projects\Examples\Greenhouse 2\GreenHouse 2.prj".
2. Change the project settings:
   a. "Architecture" must be set to NX32L.
   b. "Debug" must be enabled.
   c. "Insight!" must be enabled.
2. Please make sure that the PhoneNo Job variable is set to the correct phone number.
3. Build and transfer the project to the device.
4. Press the toolbar Launch button and await the the **RTCU IEX** to start with the status "Waiting for device".
5. Press the toolbar Connect button and wait until the **RTCU IEX** reports "Connected" and the control panel unfolds.
   The top of the RTCU IEX control panel shows information about the device and the application:

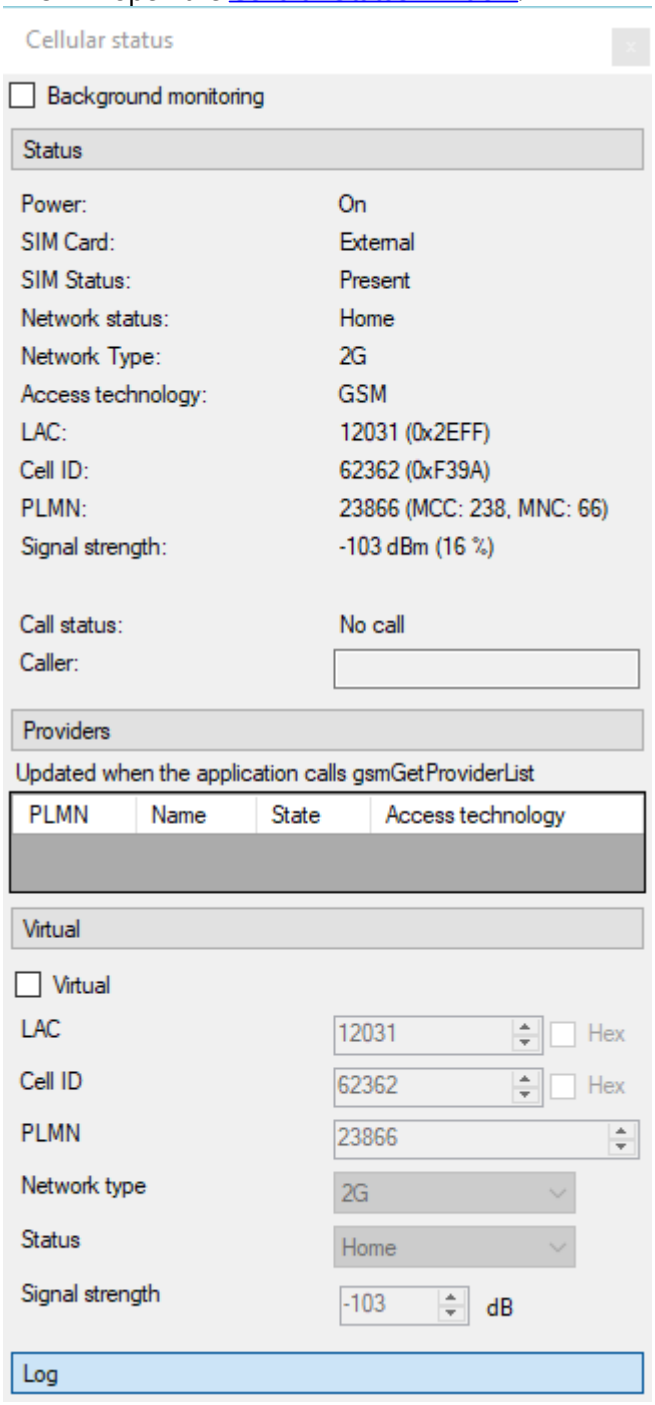


We are now ready for the next step: Chapter 2: Examining the device status.

2.2 Chapter 2: Examining the device status

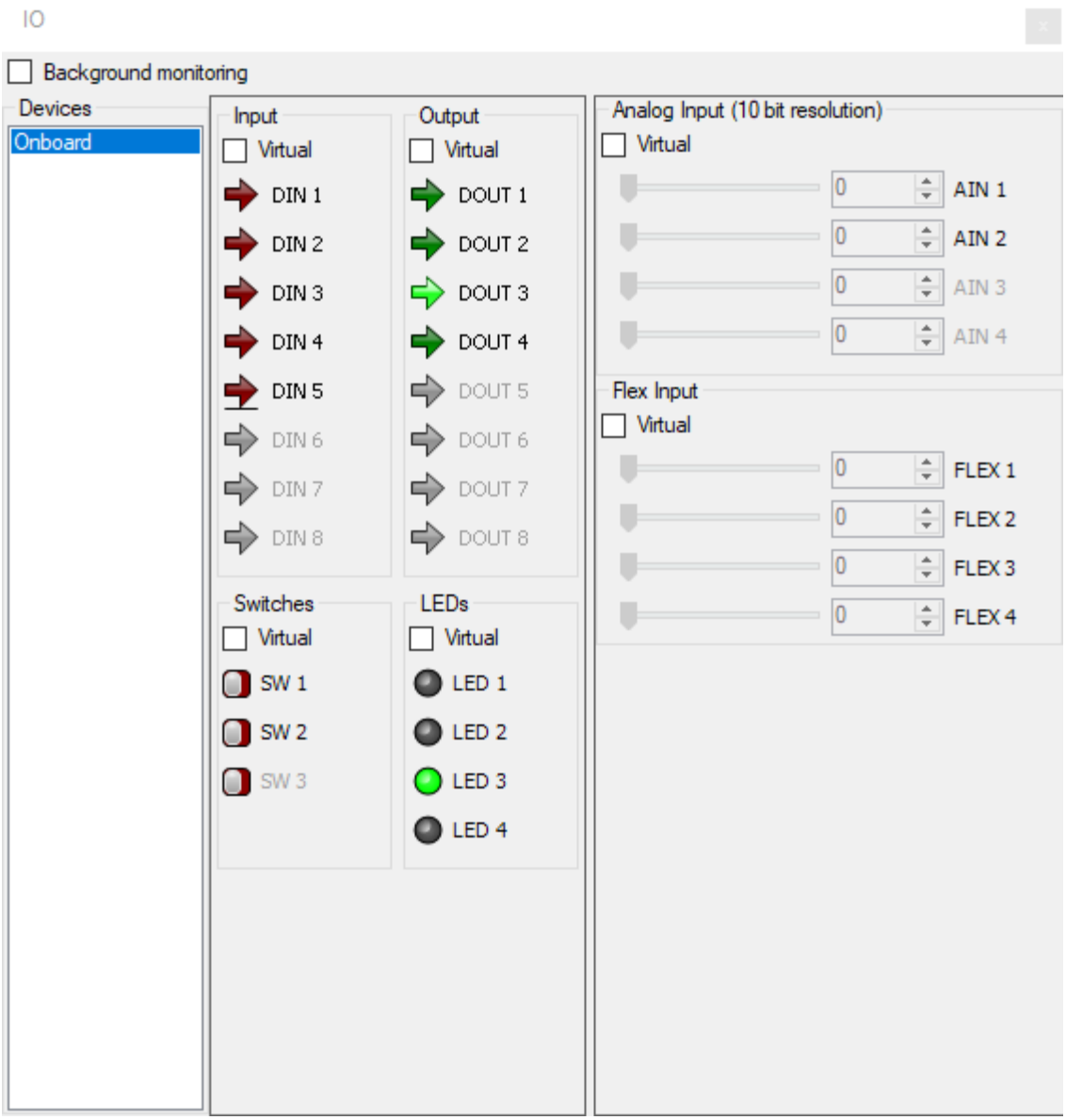
The RTCU IEX is now connected to the device and can be used to examine what it is doing.

We will start by clicking the Cellular button in the Communication section on the control panel. This will open the Cellular Status window:



This window shows the status of the cellular connection. We can see that it is turned on because the application called gsmPower() and we can see the connection status.

We will now open the IO window by clicking on the IO button in the control panel:



This window provides access to viewing the value of inputs and outputs and changing the value of the inputs.

We can see that DOUT 3 is active, because it is mapped to the Lamp variable which is on by default.

We can also see that DIN 1 and DIN 2 which are mapped to Sensor_L and Sensor_H are both off because there is nothing connected to the inputs.

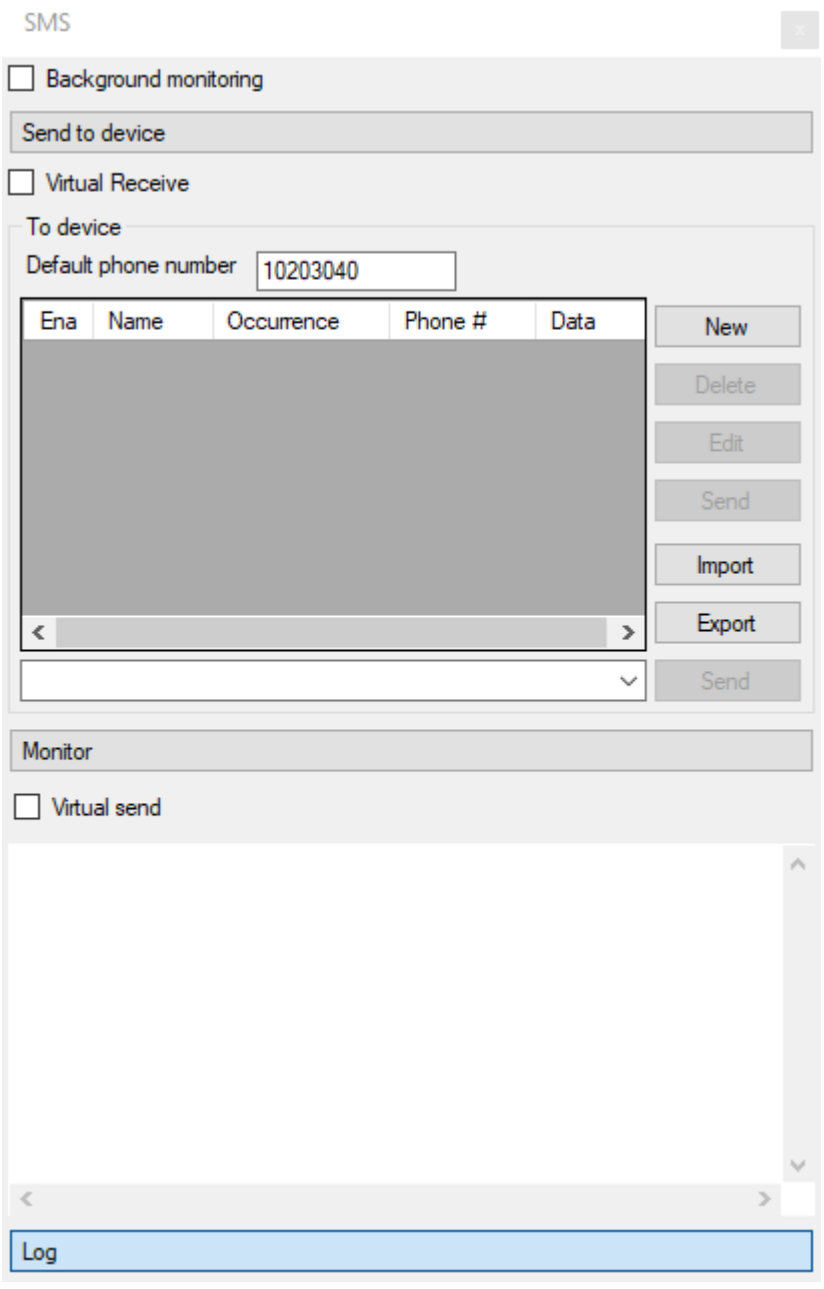
LED3 is ON as it is mapped to the Connected variable, indicating that the cellular network is connected.

We are now ready to start controlling the device in the next step: Chapter 3: Virtual IO and receiving SMS

2.3 Chapter 3: Virtual IO and receiving SMS

In this chapter we will start using virtual SMS messages and virtual IO to interact with the application.

Start by opening the SMS window:



We start by configuring the SMS features:

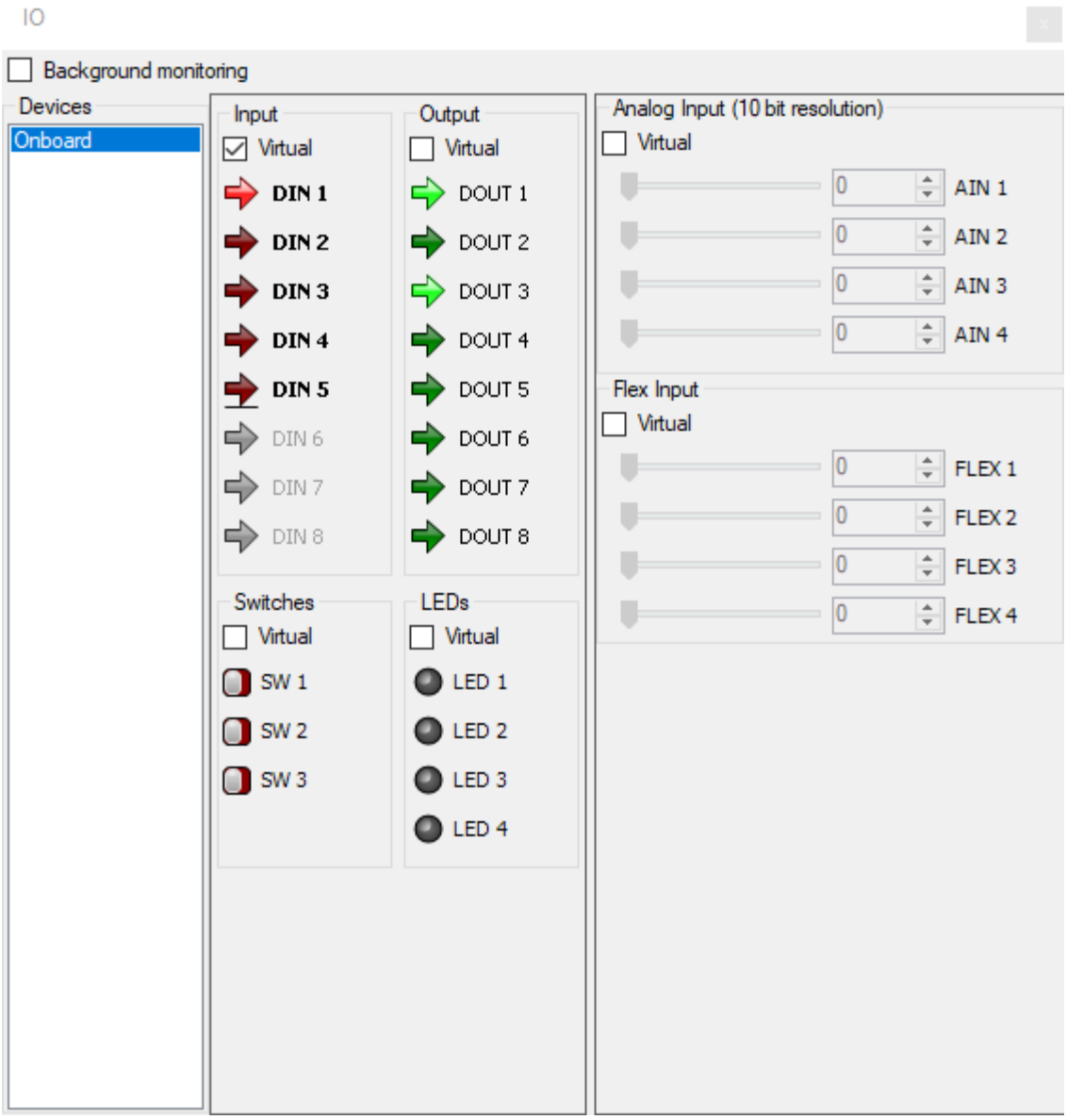
1. Enable "Background monitoring" to make it keep monitoring if the window is closed by mistake.
2. Enable Virtual Receive, to be able to send virtual SMS messages to the device.

3. Enable Virtual send to prevent the device from sending actual SMS messages.

The application will send an SMS if it detects that the temperature is too low or high for too long, so we will simulate this to make the device send an SMS:

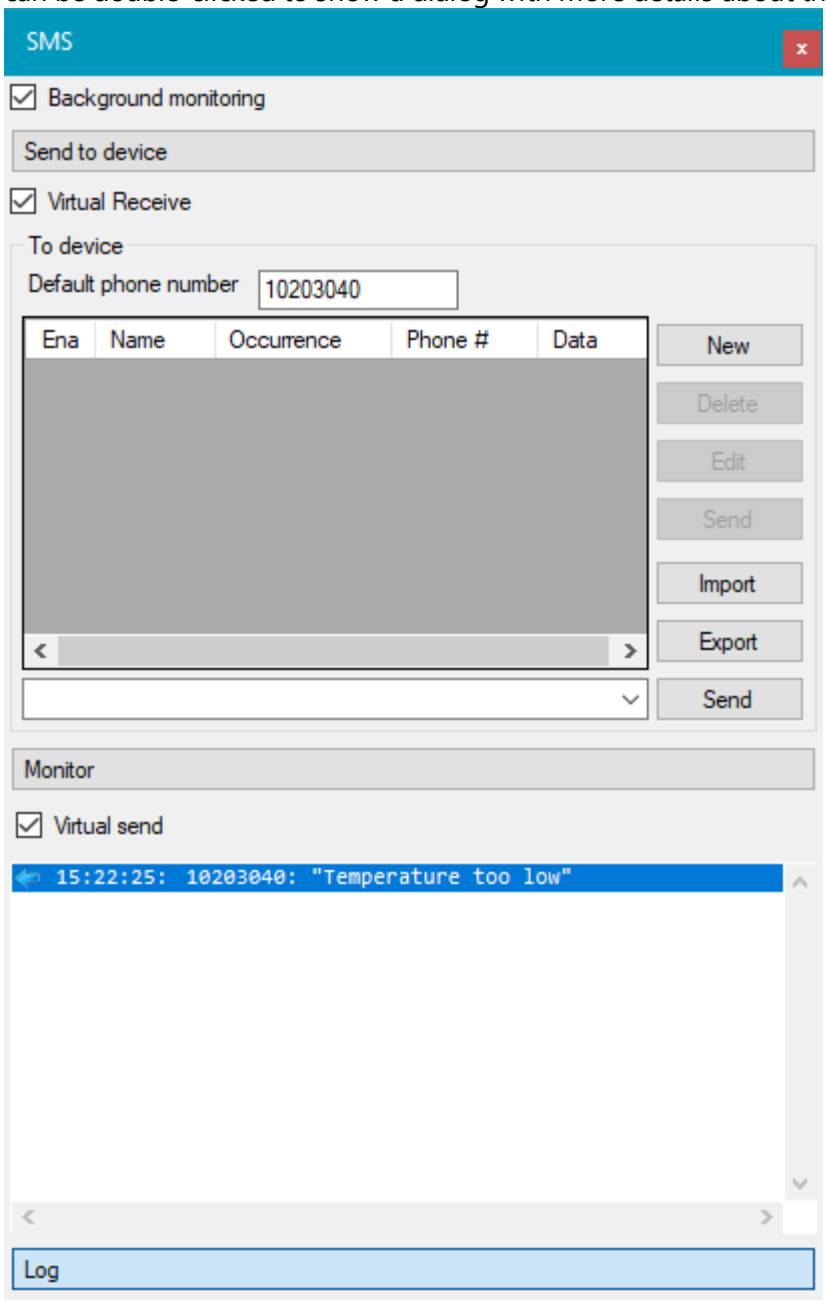
We start by enabling virtual digital input in the IO window, which allows us to control what value the application sees on the inputs.
We can also enable virtual digital output to disconnect the physical outputs from the application, so we e.g. do not have to listen to clicking relays or worry about activating external hardware that is connected to the outputs.

Click on DIN 1 to report a low temperature. This will turn on the heater on DOUT 1 and start the timer which will eventually send an SMS once the delay configured with Alarm_time has elapsed.



Virtual DIN enabled. DIN 1 is active which turned on DOUT 1.

After about 2 minutes, the SMS monitor will show that it has received the SMS. The SMS entry can be double-clicked to show a dialog with more details about the message.



Turn off DIN 1 which will then turn off DOUT 1.

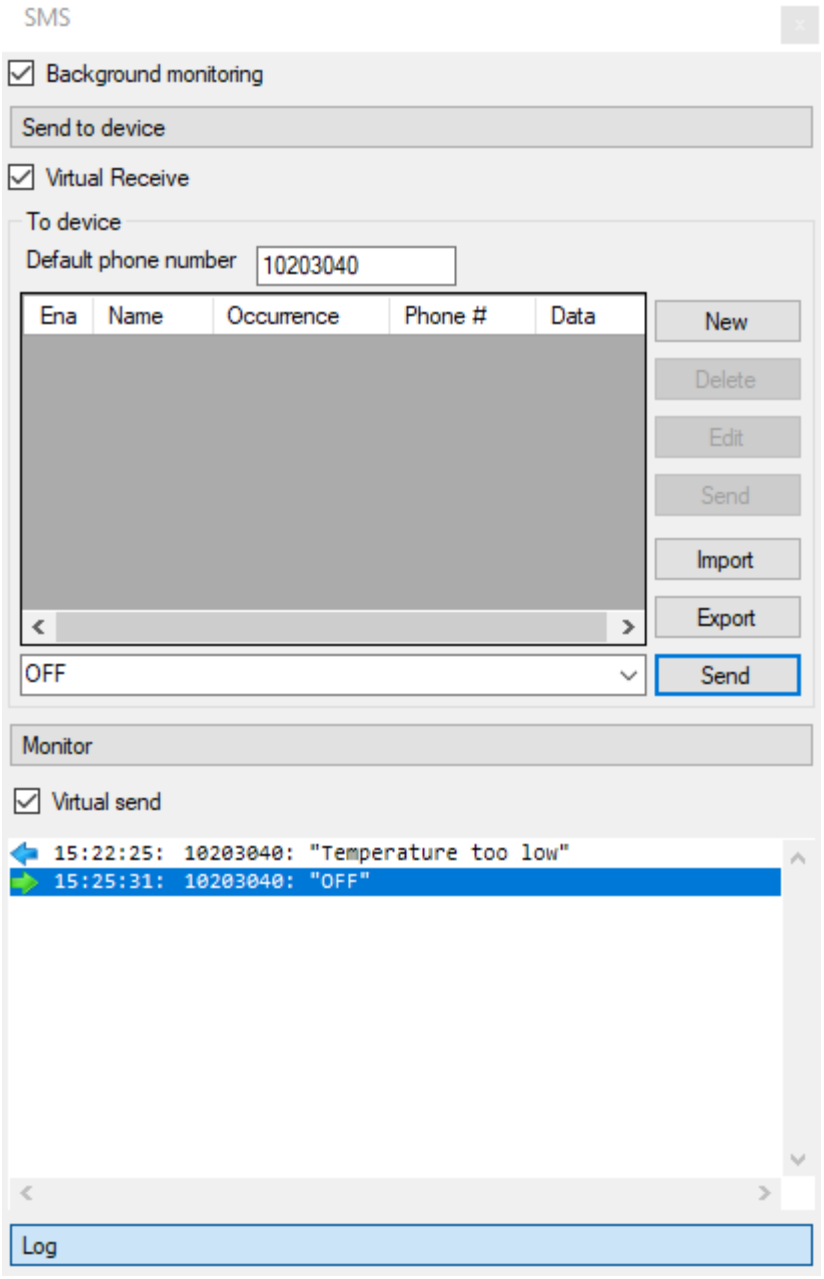
We are now ready for the next step: Chapter 4: Sending Virtual SMS

2.4 Chapter 4: Sending Virtual SMS

In this chapter we will continue working with SMS and IO, by examining the different ways we can send virtual SMS messages to the application.

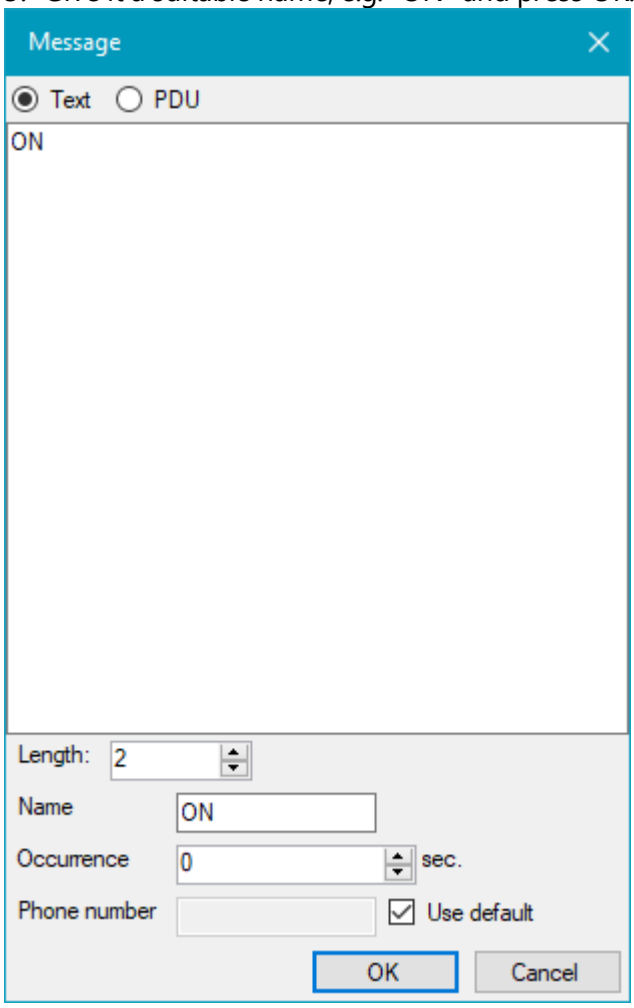
We will now send an "OFF" SMS to the device to make it turn off the lamp output:

1. Write "OFF" in the text field next to the Send button and click the Send button or press enter to send the SMS.
2. The Monitor panel now shows that the device has received an SMS from the default phone number with the text "OFF".
3. The IO window now shows that DOUT 3 has turned off, as the device reacted to the SMS.

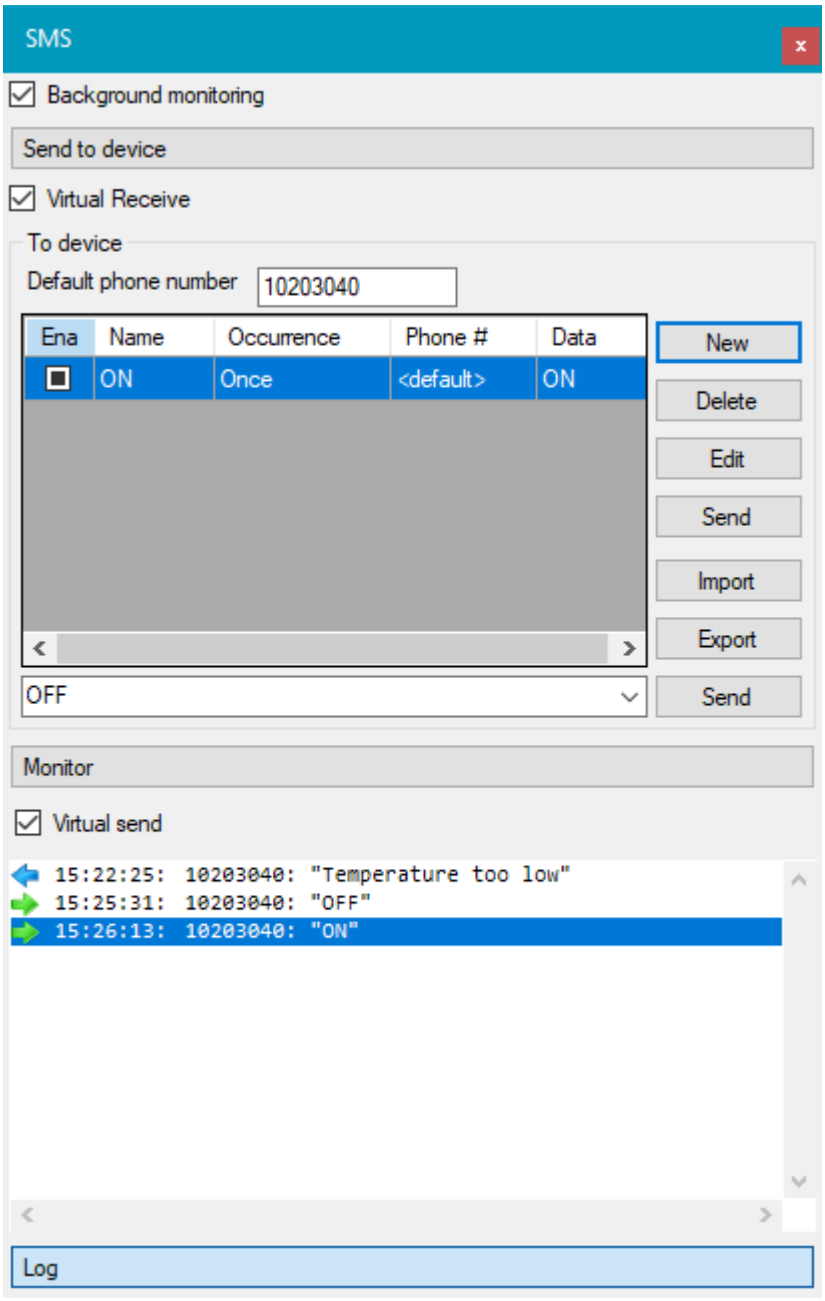


Instead of having to type out the SMS to send every time, it is possible to create a list with saved SMS messages that can be send with the push of a button or automatically.

1. Click the "New" button
2. A new dialog is shown. Select Text and write "ON" in the text field. The length will automatically be set to 2.
3. Give it a suitable name, e.g. "ON" and press OK.



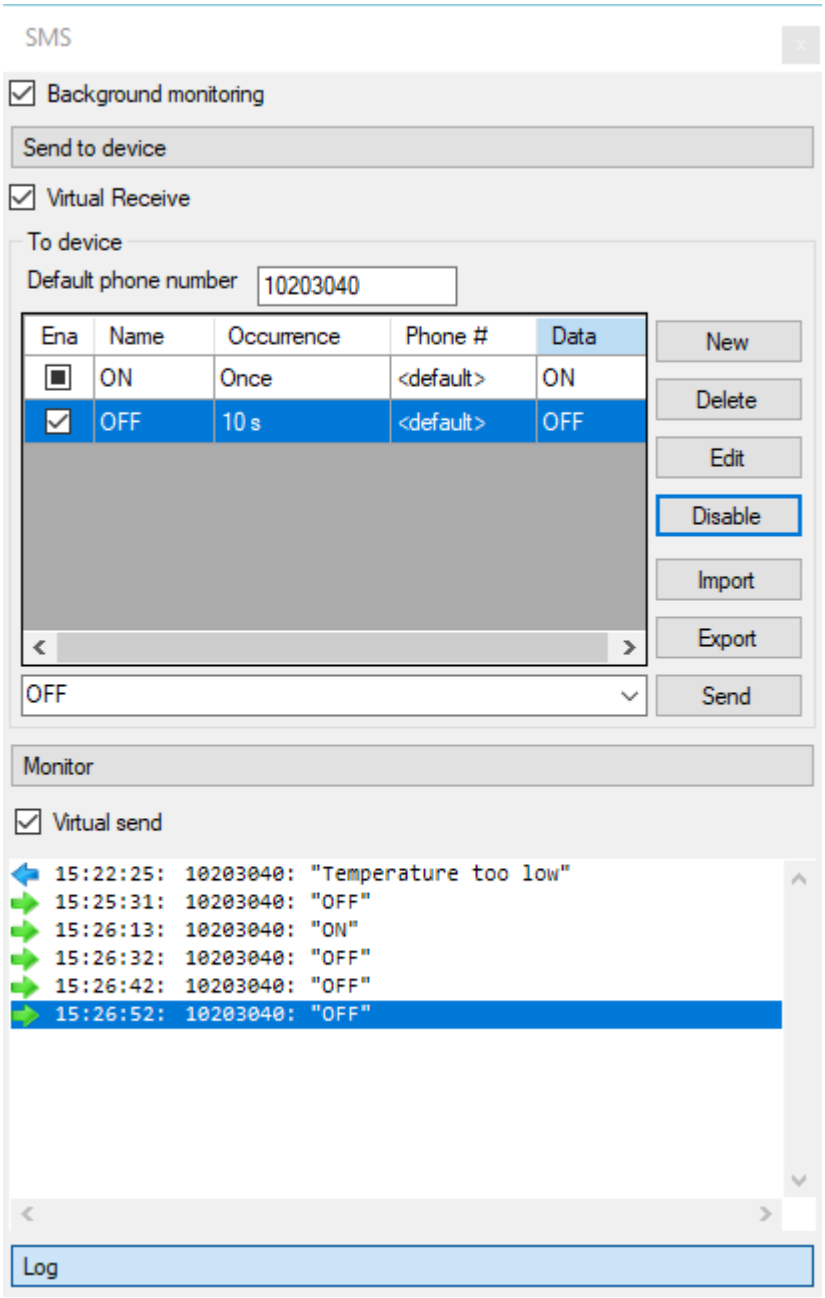
The list now contains the "ON" message. Select it and click Send to send it and turn the lamp back on.



We will now create an SMS that is automatically sent periodically to turn the lamp back off.

1. Click the "New" button
2. A new dialog is shown. Select Text and write "OFF" in the text field. The length will automatically be set to 3.
3. Give it a suitable name, e.g. "OFF".
4. Set Occurrence to 10 seconds and click OK.

The list now also contains the "OFF" message. Select it and click "Enable" to make it start sending the "OFF" SMS every 10 seconds.



Select the "ON" SMS and click send: It will now turn the lamp on for a few seconds before it is turned off again when the next "OFF" SMS is received.

The functionality to create saved messages that can be sent automatically is also available for the CAN and serial port interfaces.

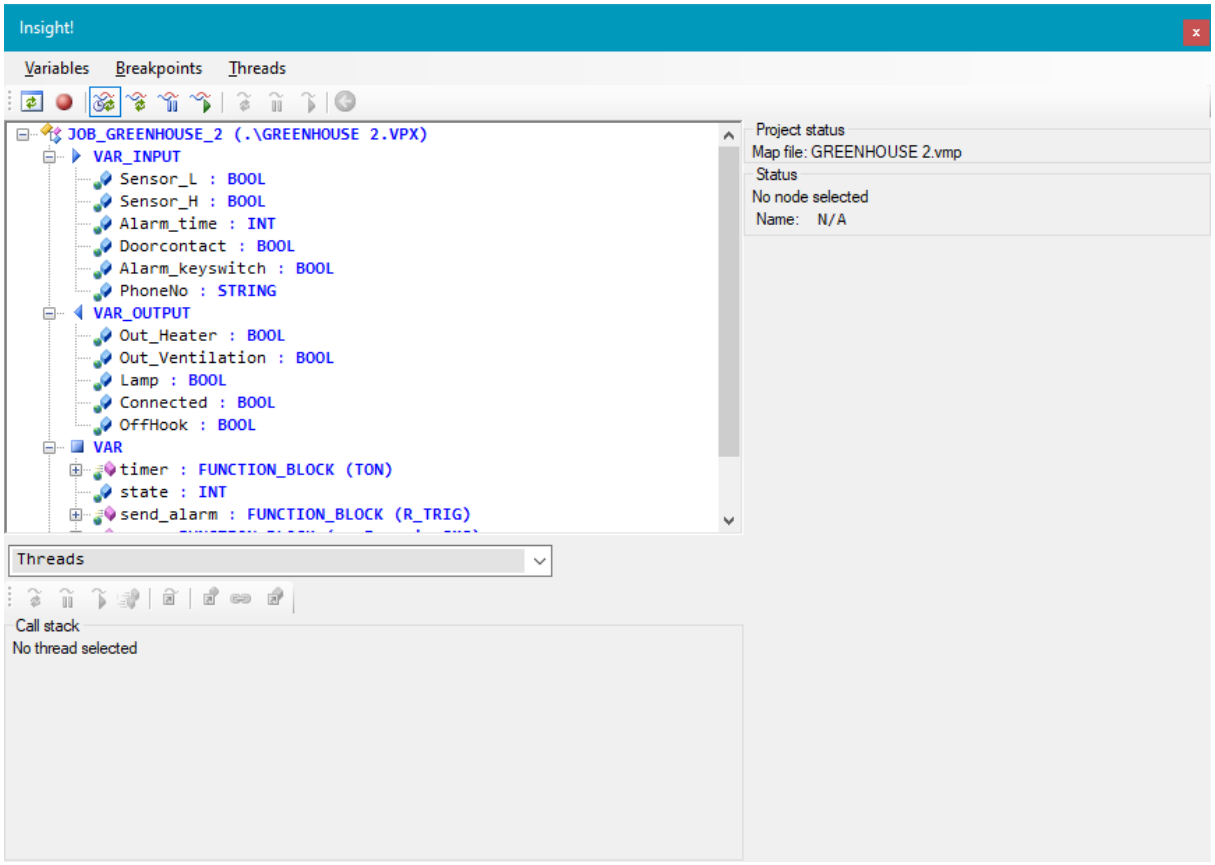
Disable the "OFF" message when done experimenting with it.

We are now ready for the next step, where we step deeper into the application with: Chapter 4: Insight

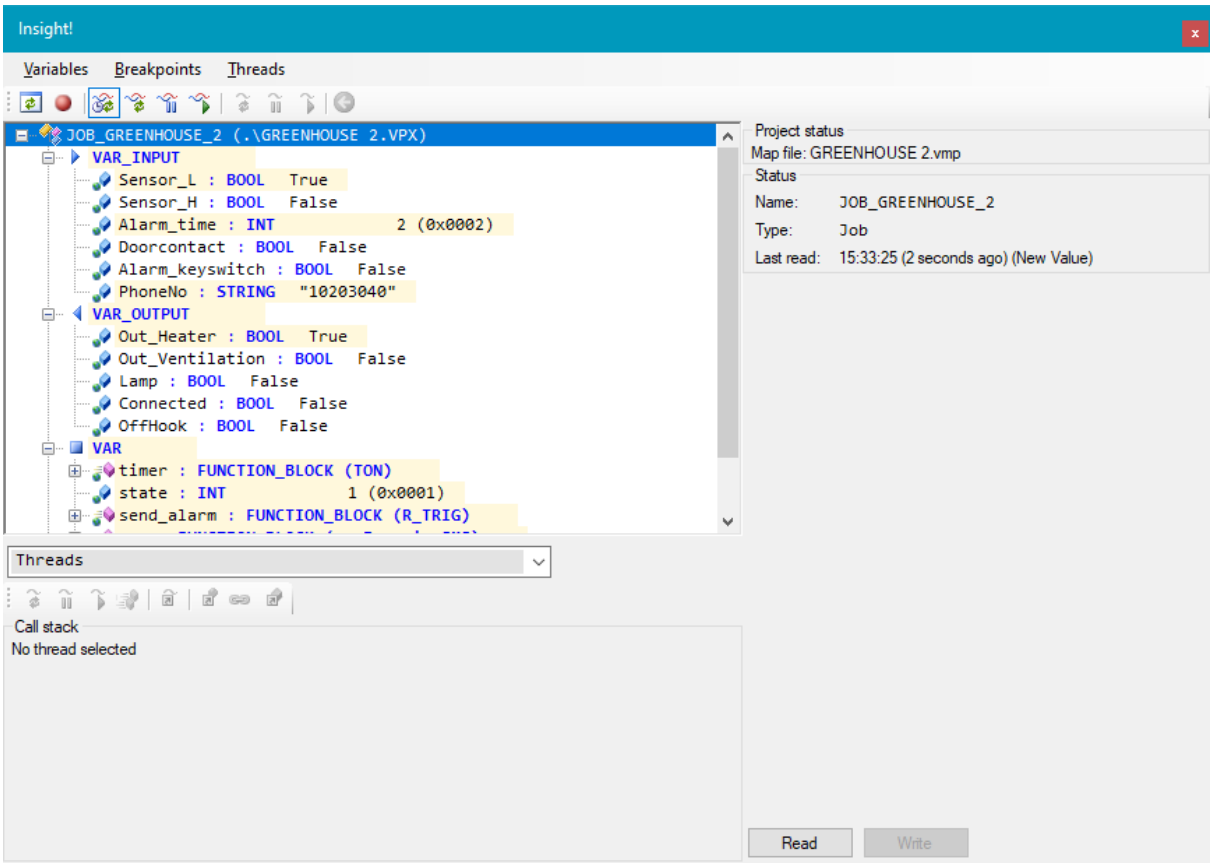
2.5 Chapter 5: Insight

The insight support opens up a view into the core of the VPL application itself.

Start by opening the "Insight!" window.
It will show an overview of the global variables in the application:



Select the job JOB_GREENHOUSE_2 and press F5, click the Read button or use the menu [Variables]-[Read] to read the current value of all the variables in the Job:

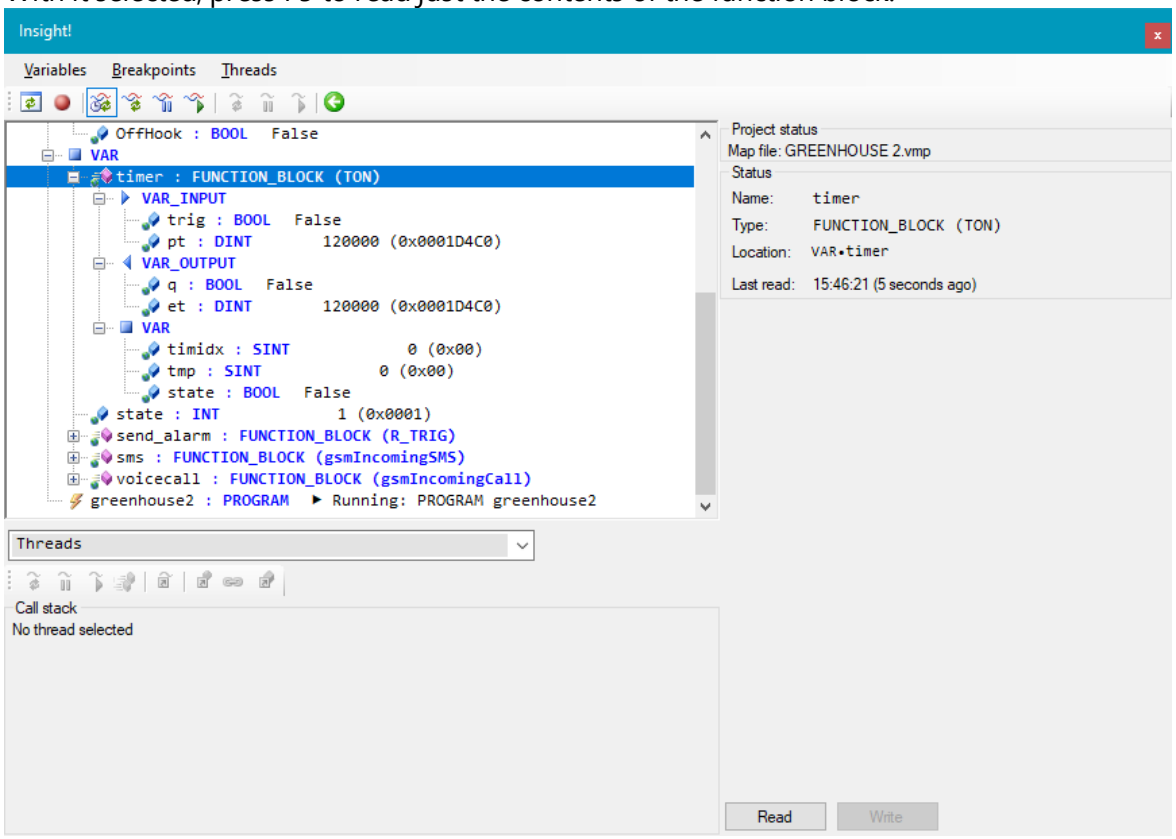


Values that have changed since they were last read are highlighted with yellow.

We will now take a closer look at the `timer` function block, which is used to detect when the temperature has been out of range for too long.

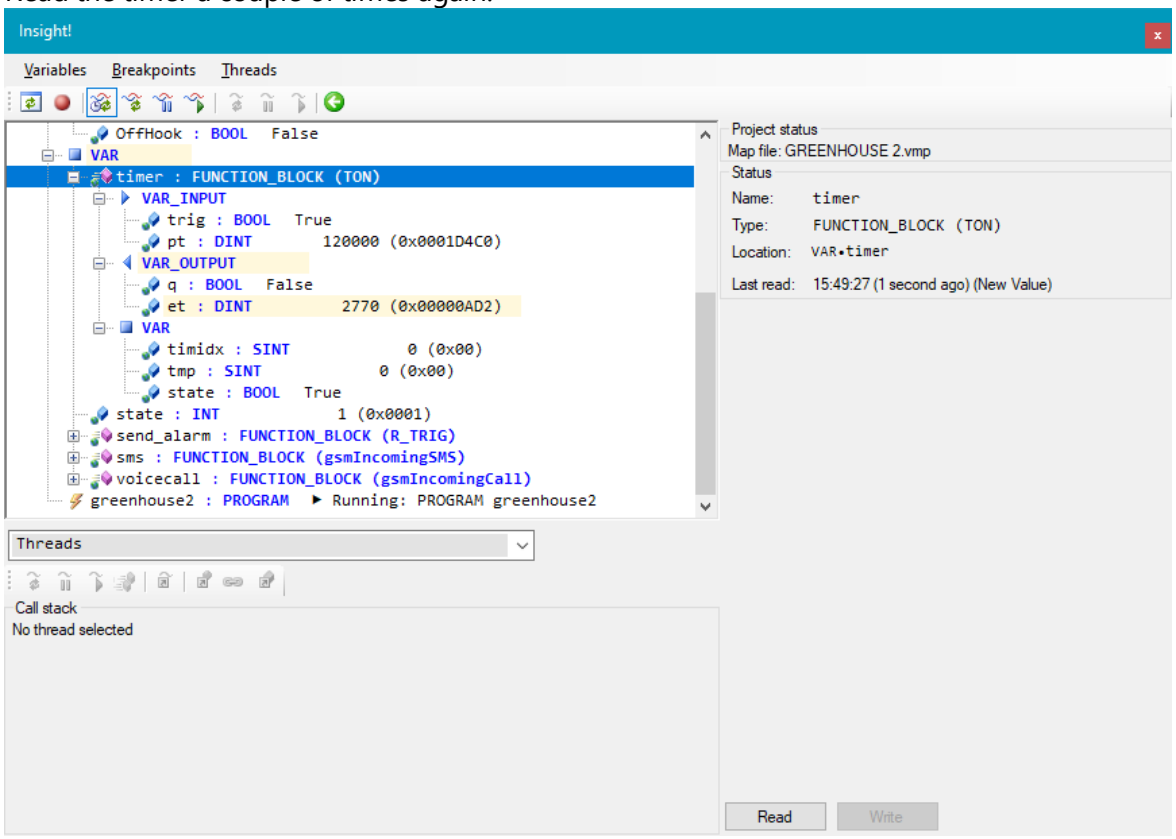
1. Scroll down and double-click the `timer` function block to expand it.

2. With it selected, press F5 to read just the contents of the function block:



3. Turn DIN 2 on in the IO window, to represent a too high temperature.

4. Read the timer a couple of times again.



5. It can be seen that the `et` variable is now counting up, eventually stopping once it reaches the value of the `pt` variable.
6. Disable DIN 2 again when done.

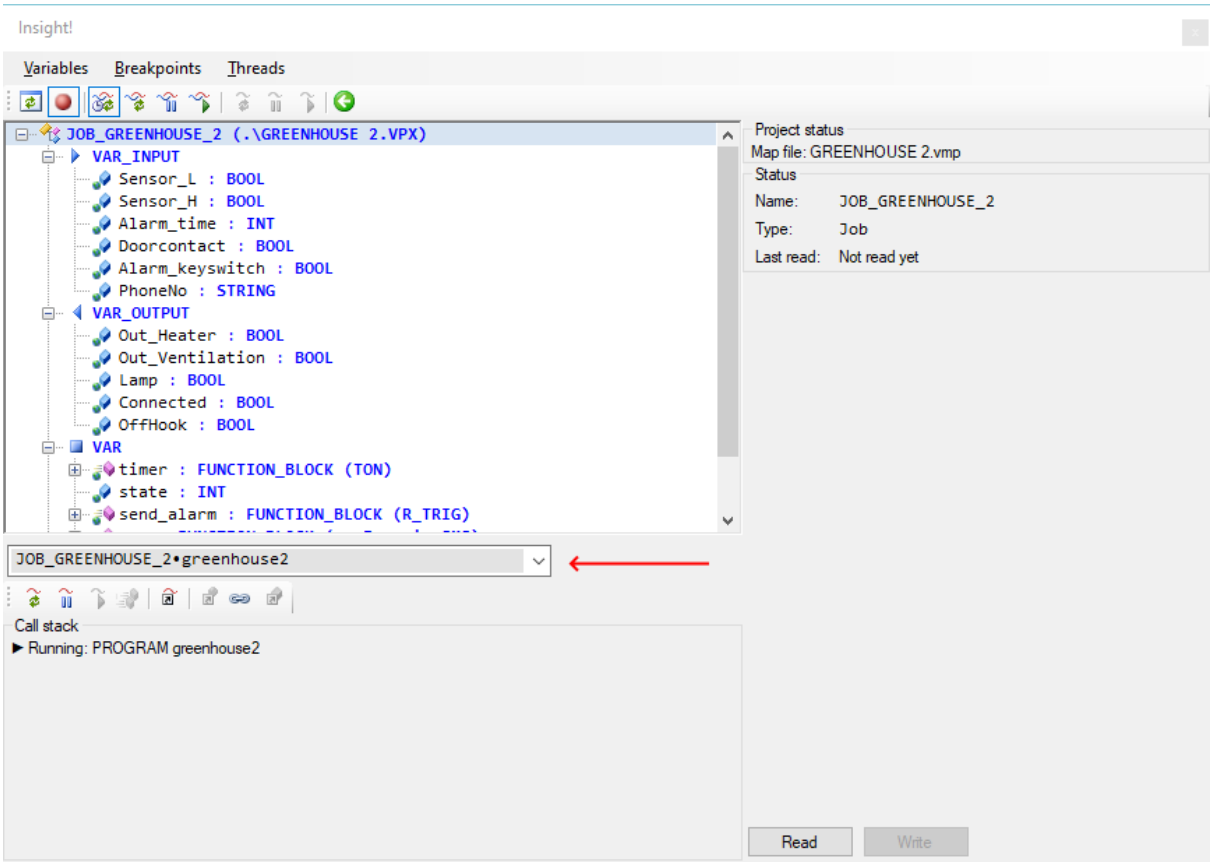
We will now change the code to add a breakpoint, allowing us to pause the execution of a thread once it reaches a specific line of code.

1. In the function SMSLamp starting on line 90, add a new line with the `BREAKPOINT` keyword between the lines with strRemoveSpaces and strCompare:

```
tmpstr:=strRemoveSpaces(str:=sms.message);
BREAKPOINT;
IF strCompare(str1:=tmpstr, str2:="ON")=0 THEN
```

2. Build and transfer the application to the device.
3. In the insight window, enable breakpoints by pressing or via the menu [Breakpoints]-[Enable breakpoints].

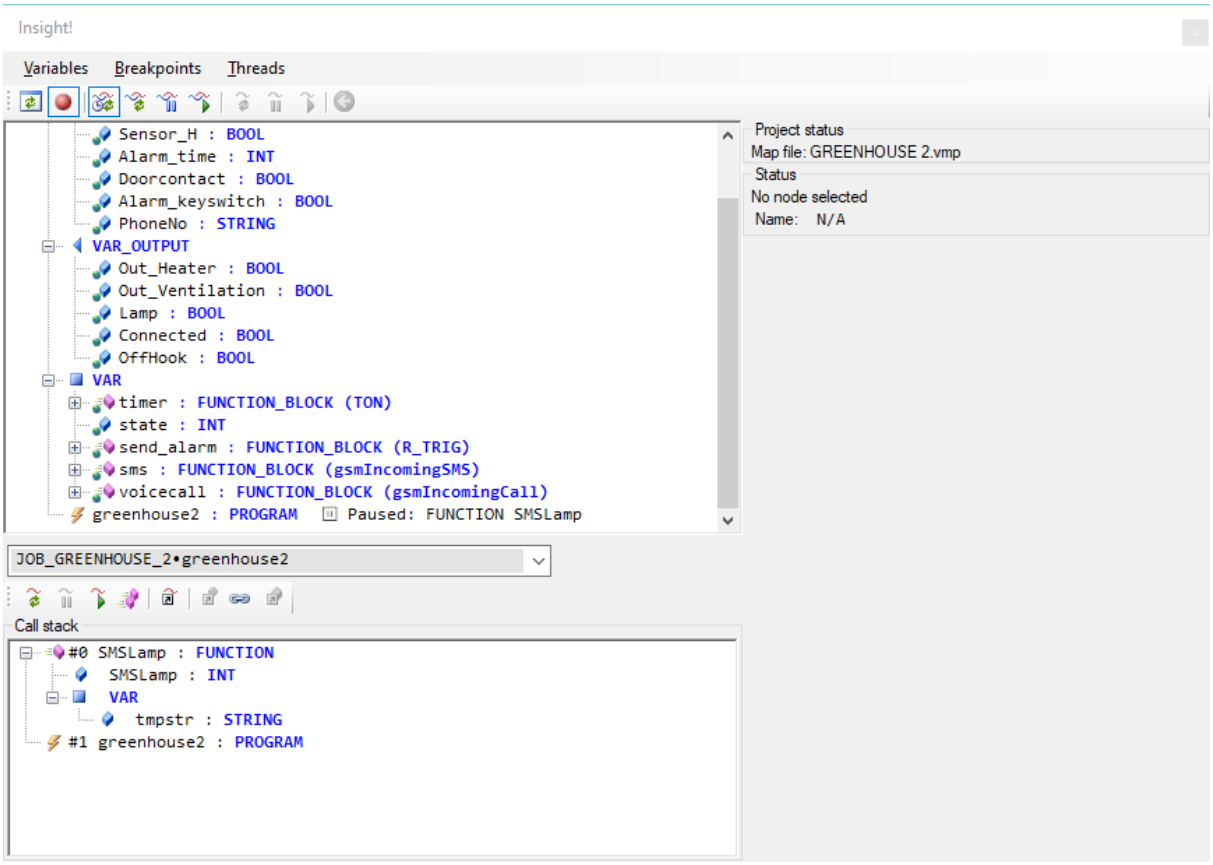
When this is enabled, any thread that reaches the `BREAKPOINT` keyword will be paused.
In the insight window, use the drop down to select the greenhouse2 PROGRAM thread:



The status shows that the thread is running.

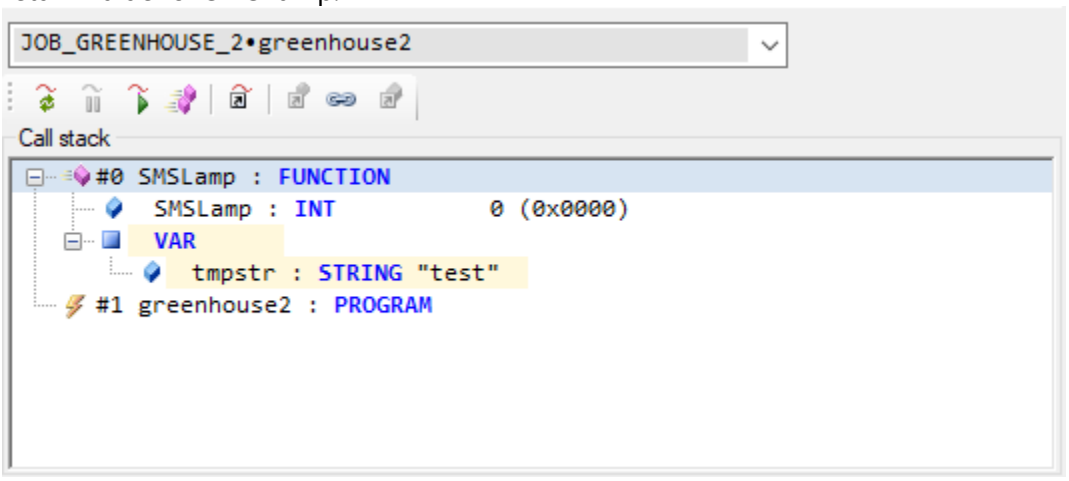
Use the SMS window to send an SMS with the message "test" to trigger the breakpoint.

Because automatic thread updates are enabled, the thread will update after a few seconds to show the stack:

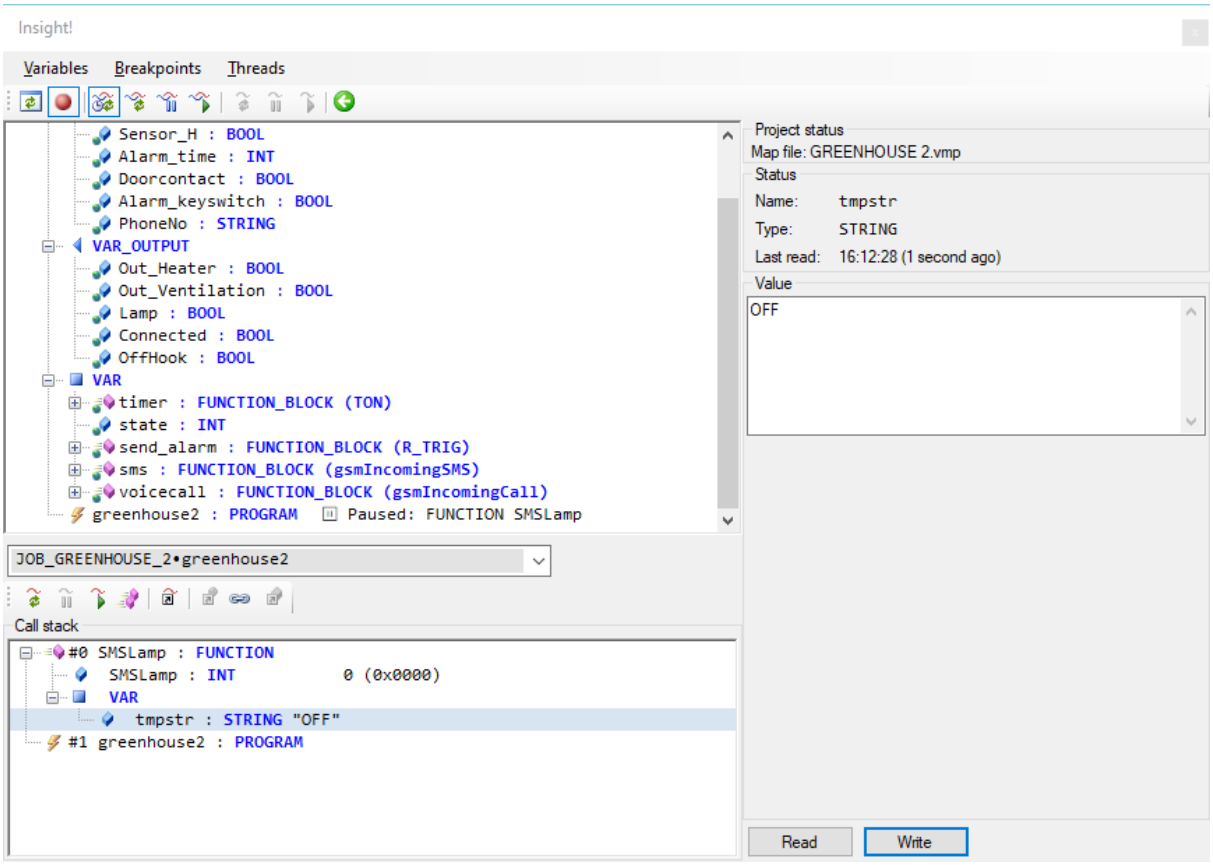


It shows that the thread is paused in the SMSLamp function, called from the greenhouse2 program.

Select the SMSLamp function and press F5 to read the value of the local variable tmpstr and the return value for SMSLamp:



Select the tmpstr variable and change the value in the value panel to "OFF" and press write to update the variable in the device:



Press "Resume thread" to let the thread continue, turning the lamp off.

Instead of pausing a thread with a breakpoint, a single thread can be paused by selecting it in the drop down and pressing "Pause thread" and all the threads can be paused by pressing "Pause all threads". This will e.g. show if a thread is stuck in a function call.

This completes the tutorial.