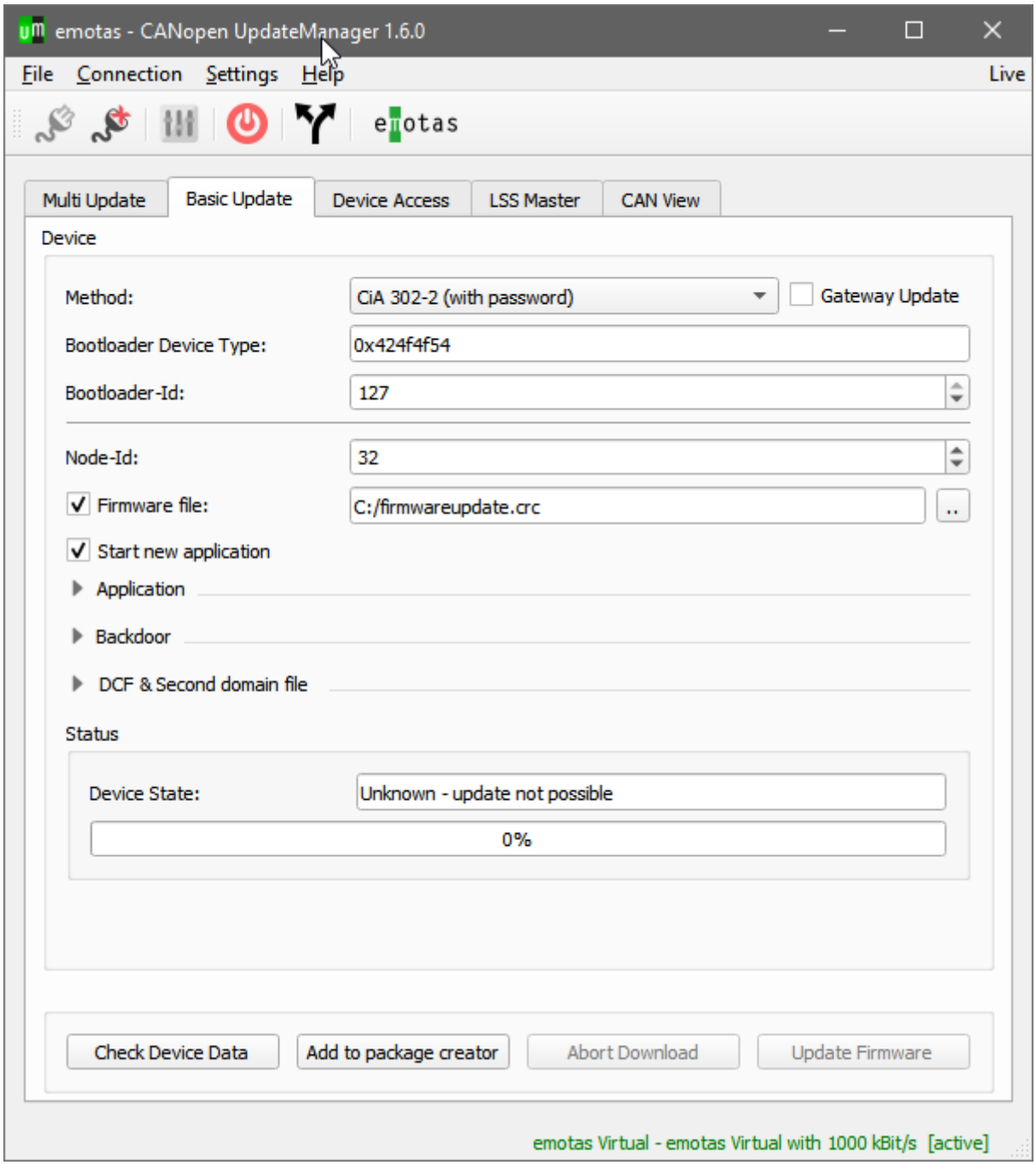
User Manual
CANopen UpdateManager



Version history

Version	Changes	date	editor
V1.0	First version	03.04.2012	ged
V1.3	Firmware comparison	2015/08/30	ri
V1.4	Some improvements	2017/05/10	ri
V1.5	Description of command line usage	2019/02/22	ri
V1.6	Edited for emotas	2019/10/07	ri

Disclaimer

Even if software from emotas embedded communication GmbH is produced on high reliability levels and tested on different environments there is no proof on absolute reliability. Therefor emotas embedded communication GmbH cannot give any liability on even software or documentation and cannot guarantee there functionality in use cases of our customers. Especially description and technical data are no granted properties of our products. As a consequence emotas embedded communication GmbH will not be responsible for any damages and losses in consequence of using our products.

emotas embedded communication GmbH may apply changes on products and documentation to improve reliability, stability and serve technical advantages. This changes may not be announced.

emotas embedded communication GmbH holds all rights on their products and documentation. Propagation to third parties even from parts of products or documentation may be permitted by emotas embedded communication GmbH. Copies of products or documentation may by established only on backup purposes. It is in the responsibility of our customers not to pass these copies to third parties. We appreciate very much all considerations of our customers on errors ore other aspects of improvement.

Copyright

Fritz-Haber-Str. 9
D-06217 Merseburg
Germany
Tel. +49 3461/79416-0
Fax. +49 3461/79416-10
service@emotas.de
http://www.emotas.de

Table of contents

1. Introduction

Thank your for using the CANopen UpdateManager. The program is for firmware updates of CANopen devices with CANopen bootloaders. The following manual explains the installation and use of the program.

2. Installation

2.1 Windows

To install the tool on Windows start the setup setup_canopen_updatemanager.exe and follow the instructions of the setup. The setup creates a shortcut to start the program.

Please note: Drivers for CAN interfaces are required and need to be installed separately.

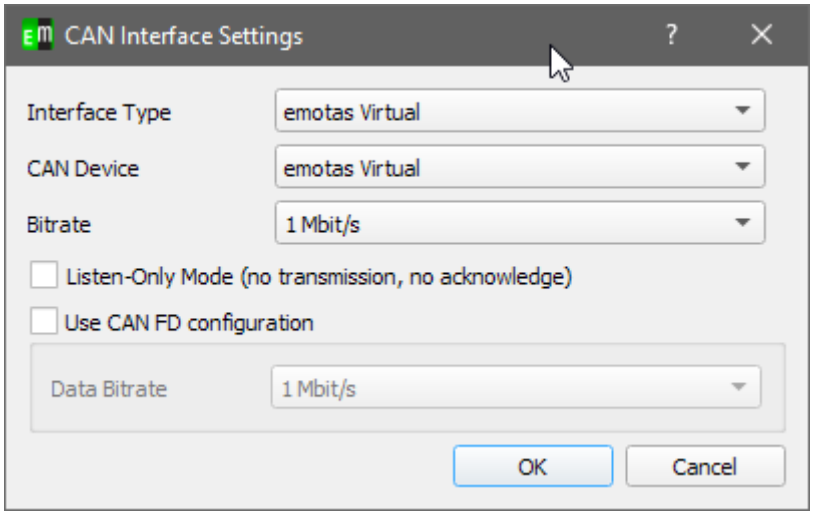
2.2 Linux

To install the tool in Linux just unzip the ZIP file setup_canopen_updatemanager.zip into a directory. To start the program run the script CANopenMasterManager.sh in this directory.

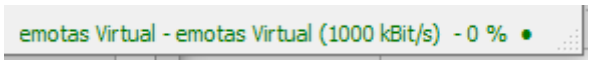
In Linux the CANopen UpdateManager uses the SocketCAN-API which is supported by many manufacturers of CAN interfaces. Make sure that the SocketCAN devices is active and correct bit rates are set before starting the tool.

3. First Steps

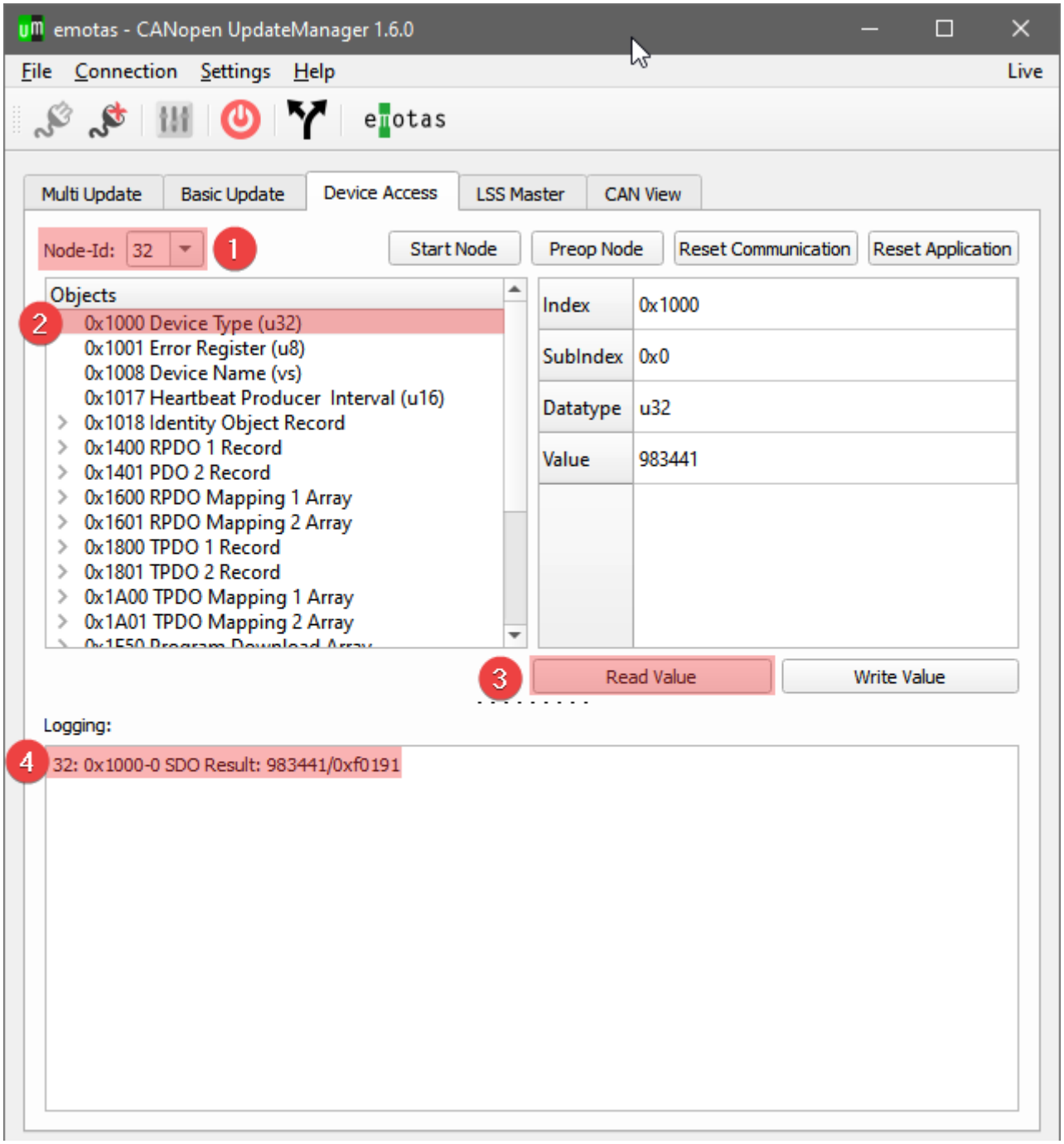
The first step at the very first start of the tool is the configuration of the CAN interface. Open CAN interface settings at the menu entry "Connection → CAN Interface Settings". The following mask appears



Choose the type of the CAN interface, the name of the CAN device and the bit rate in the CANopen network and confirm the settings with OK.
Connect now the CANopen UpdateManager with the CAN interface via "Connection → Connect". In the status bar you can see now "Connected to " with the name of the CAN device and the current bit rate.



To test the functionality of the CAN connection, select the "Device Access" tab. After that configure the node-Id of the device(1) and click in the object tree at the object 0x1000(2). Press "Read Value"(3) afterwards. Now the object $0x1000_1$ is read and the result is displayed at "Logging"(4)



If a valid value has been read, this indicates that the node Id is correct and the CAN connection works. If you do not receive a response, but a SDO timeout, please check the node Id and the CAN bit rate.

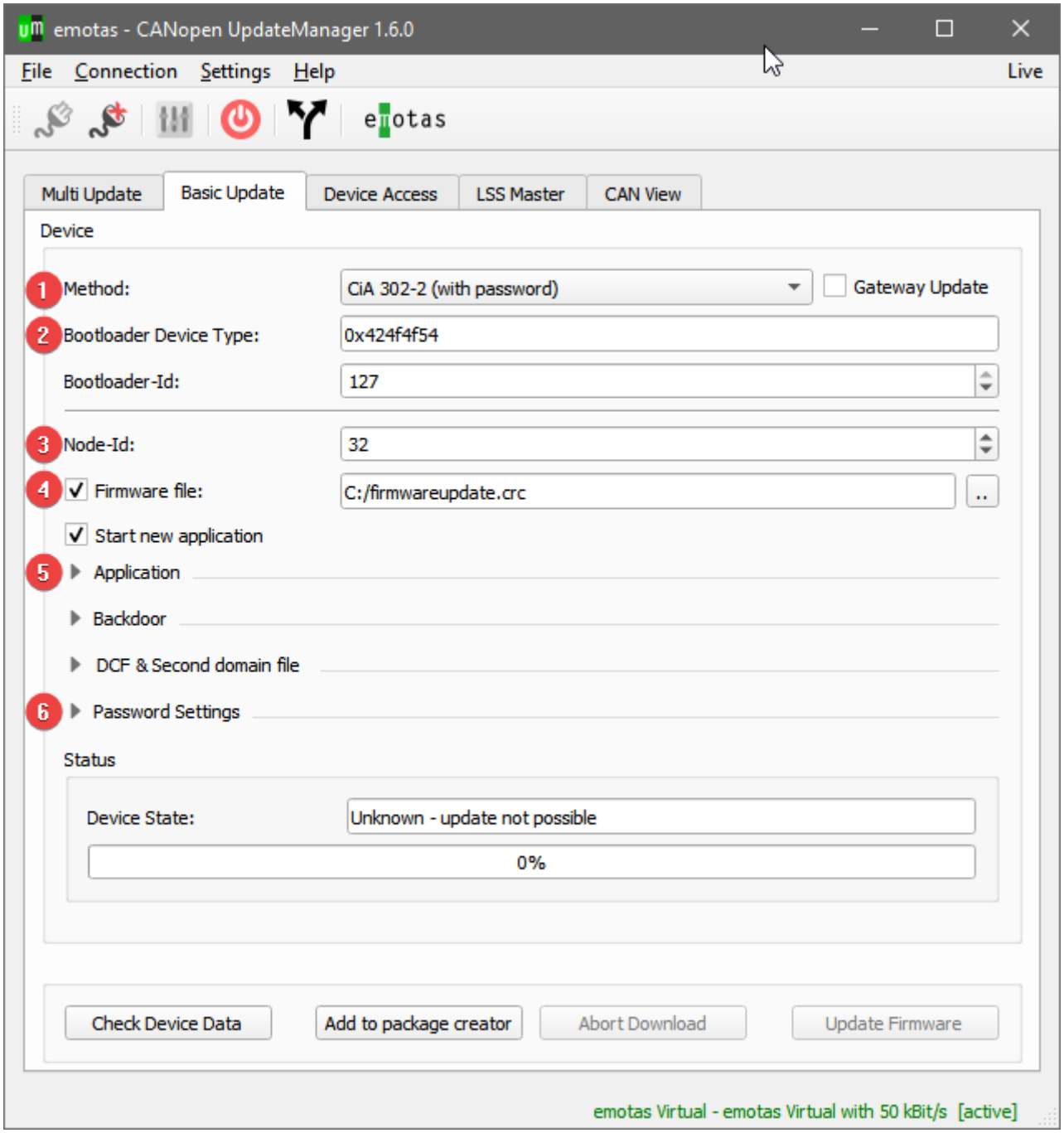
With an established CAN connection a firmware update as described in chapter 4 „**Tabs → Basic Update**“ can be done.

Unless switched off, the tool saves all settings when quiting the tool by default, so at next start you can start with the same settings

4. Tabs

4.1 Basic Update

With the „Basic Update“ tab the firmware of each CANopen node can be updated by their bootloader. Different methods are supported including the switching from application to bootloader. The settings are explained on the following page:



1. Method

   Selection of the update method (e.g. if device is CANopen or not)

2. Bootloader features

   Node ID of the bootloader and the device type are defined here. The node ID of the bootloader can be identical with that of the application or be different from it.

3. Node Id of the device (application)

   Definition of the node ID of the application.

4. Path to firmware file

   Set the path to the new firmware file here. The button right of the input field opens a dialog to choose the path/file.

5. Identity data of the application

   The three fields contain the data of identity of the device. They are read from object 0x1018. Comparing this data one can make sure that the correct device was chosen.

6. Update-method

   Choose the update-method. Following methods are supported:

   7. CiA 302-2 (standard CANopen)
   8. CiA 417 (CANopen Lift)

9. Password Object and password

   The access to the bootloader is saved by a password object of the data type UNSIGNED32 in many cases. The object, the SubIndex and the value of the password can be defined. The password is send to the application to activate the bootloader.

10. Checking of device data

    With a click on button „Check Device Data“ device type and identity data of the device are read and it is checked whether the bootloader is already running or still the application. After a successful check the button „Update Firmware“is active.

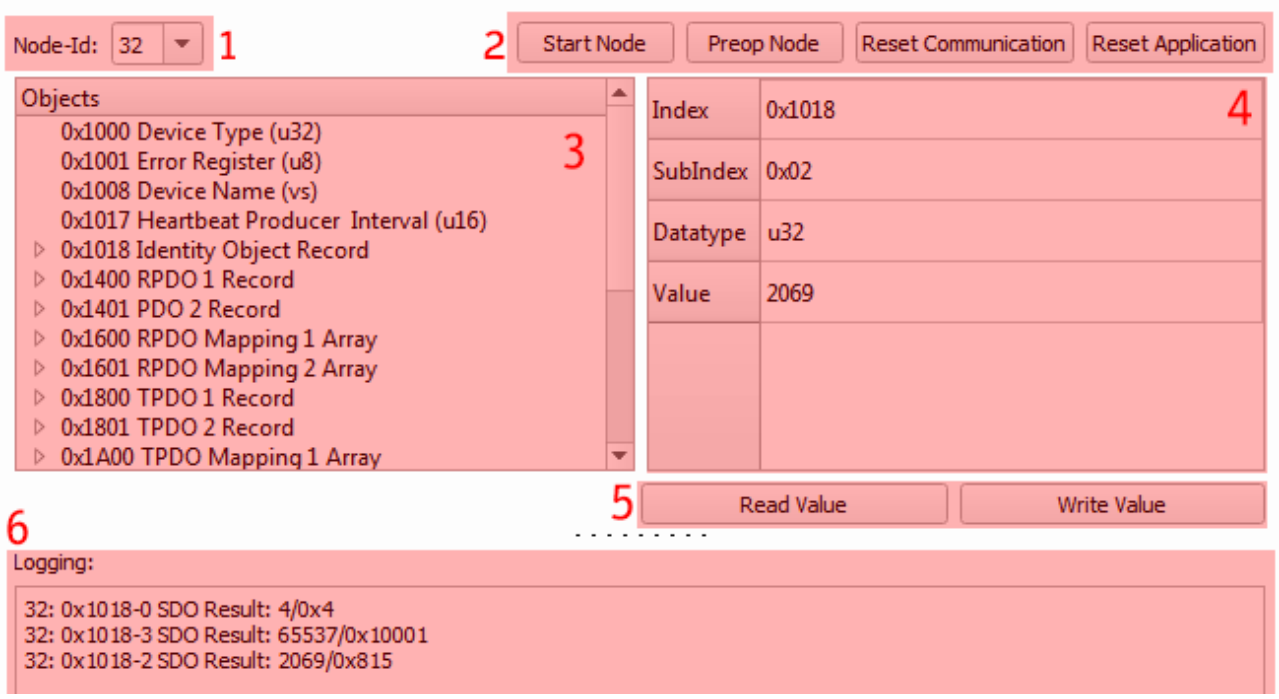
11. Starting an update

    Activates the bootloader and the transmission of the firmware to the bootloader starts. The progress is shown in a bar.

4.2 Device Access

The „Device Access“ tab allows reading and writing access to the object dictionary of CANopen devices. And it allows to send NMT commands to devices.

The structure of the tab is as follows:



1. Choose node ID (Node-Id)

   In this drop down box the node ID can be chosen. All SDO accesses and NMT commands are sent to the device.

2. NMT-command buttons

   With the NMT command buttons following NMT commands can be send to the CANopen device:

   - Start Node
   - Enter PreOperational
   - Reset Communication
   - Reset Application

3. Object tree

   The Object tree contains a preselection of general objects relevant for bootloaders. A click on an object shows the attributes in the entry mask (4). A double click on an objects and the object is read out via SDO.

4. Object entry mask

   Index, subindex and data type of any object can be entered into the entry mask to read or write from the objects. A decimal (e.g. 4096) or hexadecimal (e.g. 0x1000) notation with a leading 0x is possible for index,, subindex and value.

   Values of the data types are limited as follows:D

   - u8 .... UNSIGNED8
   - u16 .... UNSIGNED16
   - u32 .... UNSIGNED32
   - i8 .... INTEGER8
   - i16 .... INTEGER16
   - i32 .... INTEGER32
   - d .... DOMAIN
   - vs ... VISIBLE_STRING

   The use of the data type DOMAIN(d) requires a file name in the field „Value". A click on the right mouse button opens a dialog to select a file.

5. Logging

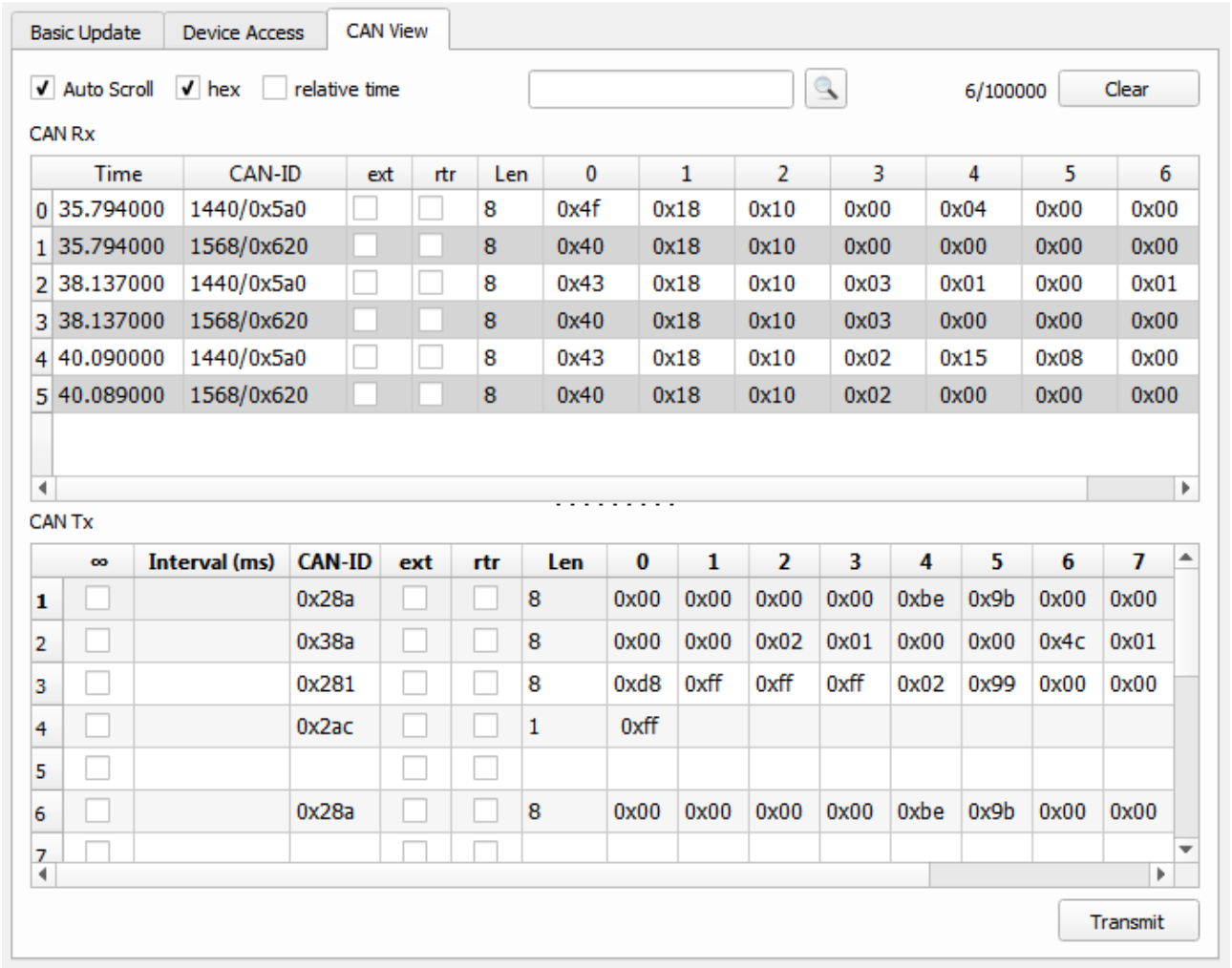
   Logging lists the results of all SDO accesses and sent NMT commands. It is not the logging of CAN messages.

6. SDO activity

   The buttons „Read Value" and „Write Value" start the reading or writing SDO access. With „Write Value" the value in the field „Value" (4) will be written to the device.

4.3 CAN message view

The CAN-Raw-View shows all sent and received CAN messages with time stamp. This tab also allows to send any CAN message.



In exported text files the recorded CAN message has this form:

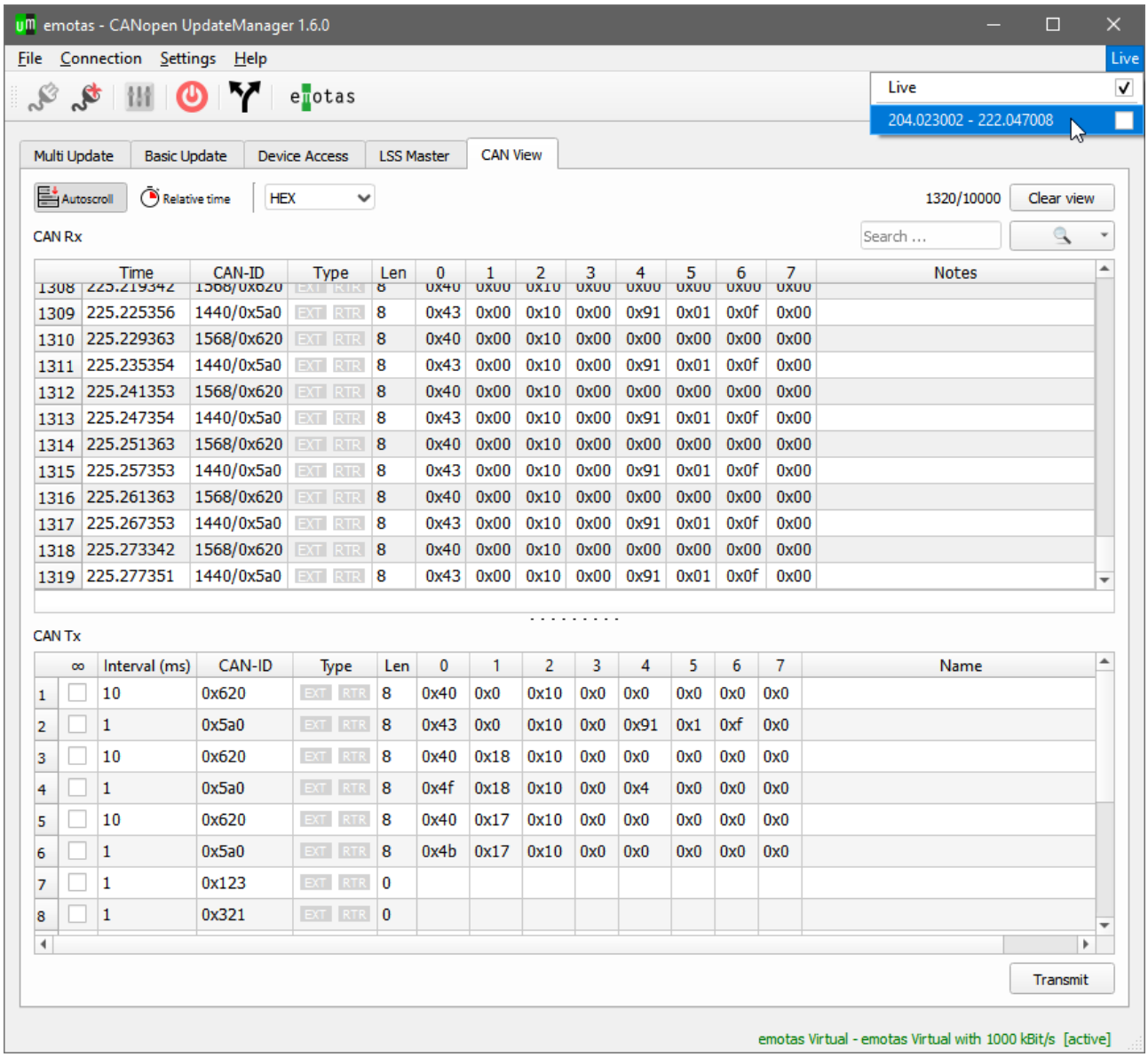
```
3.652109 0x5a0/1440 (8): 43 18 10 02 15 00 08 00
3.654306 0x620/1568 (8): 40 18 10 04 00 00 00 00
3.653302 0x5a0/1440 (8): 43 18 10 03 00 00 00 00
Zeitstempel
           CAN-ID
                    DLC
                        Daten im hexadezimalen Format
```

How exact the time stamp is depends on the operating system and the CAN interface. The time stamp for sending is not for all CAN interfaces available.

4.4 Message Handling

The received messages are stored as message package to the hard disc drive if the "Maximal message count" (configurable in the options) is reached and removed from the internal buffer.

The saved messages could be viewed via the menu in the upper right corner of the window.



Each message package holds the timestamp of the first and of the last message as name.

If a message package is selected newly received message are still being processed in the background and could be viewed by switching back to "Live".

5. Settings

5.1 CAN settings

The dialog CAN interface settings supports the configuration of the CAN interface.

- **Interface Type**

  Choose the type of CAN interface. With Linux SocketCAN is supported at the moment. With Windows PCANBasic (PEAK USB-CAN-Interface with PCAN-Basic-API) and CANfox by Sontheim Industrie Elektronik are supported, now.

- **CAN-Device**

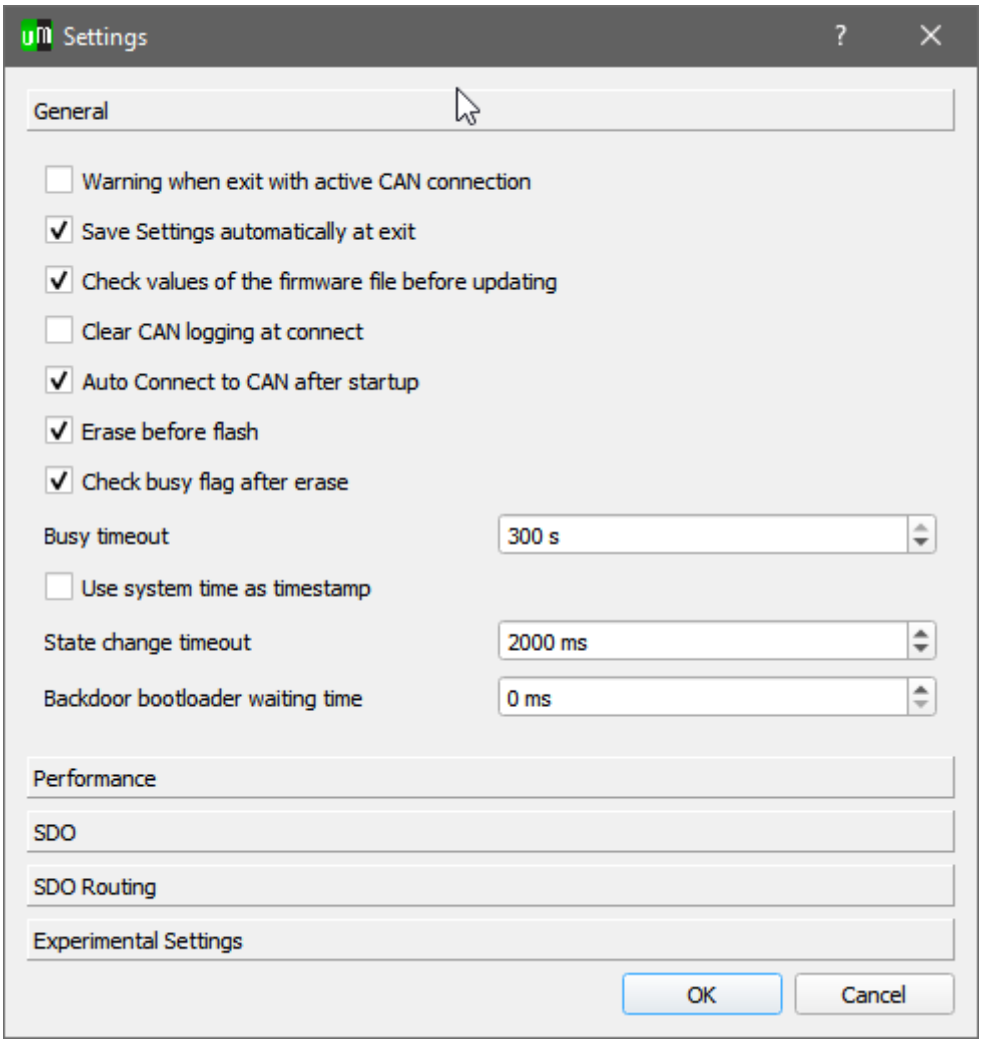
  Depending on the interface type the name of the CAN device can be chosen here. Usual names for SocketCAN are can0, can1, ... vcan0 und bei PCANBasic usb1 ... usb8.

- **Bit Rate**

  Configuration of the bit rate within the CAN network. Choose the bit rate that is also used for all other devices within the CAN network. Please note the the bit rate must be set before starting the program when used with SocketCAN.

5.2 Program settings

The setting dialog gives access to different program settings.



General

- Warning when exit with active CAN connection

  Configures whether a warning is shown when the program is left with active CAN connection.

- Save Settings automatically at exit

  Configures whether the settings are saved automatically with exit of the program.

- Check values of the firmware file before updating

  Configures whether values of the bootloader should be compared with values of the firmware file. See chapter Fehler: Verweis nicht gefunden Fehler: Verweis nicht gefunden

- Clear CAN logging at connect

  Configures whether the logging of the CAN messages is deleted at connection.

- Auto Connect to CAN after startup

  Automatic connection to the CAN interface when starting the program

SDO – settings

- SDO Timeout for normal SDO access (ms)

  The SDO Timeout indicates the time to wait for a response of the devices after a SDO message.

- SDO Timeout for Flash access (ms)

  The SDO Timeout indicates the time to wait for a response of the device after a SDO message. The time needed for a response with flash access is longer therefore this timeout is separately configured.

6. Menu

The menu offers access to different functions and settings of the CANopen UpdateManager.

File

- **Export CAN Logging**

  Export of the logged CAN messages into a ASCII text file.

- **Load EDS File**

  Loads an EDS file with the object dictionary of the device for the 'Device Access' tab.

- **Quit Ctrl+Q**

  Quits the program. Depending on the configuration all settings are saved . The CAN connection is disconnected.

Connection

- **CAN-Interface-Settings**

  Dialog to configure the CAN interfaces and the used bit rates for the CAN network. Which CAN interface types are available depends on the operating system.

- **Connect**

  Setting up of a CAN connection by using the configured CAN interfaces.

- **Disconnect**

  Disconnects active CAN connections.

Settings

- **Options**

  Opens the dialog to configure program settings (not CAN settings).

  The automatic saving of program settings at closing the program can be disabled here.

- **Save**

  Saves the recent program settings. This functions is useful in case of disabled auto save function.

- **Export Settings**

Export of recent settings into a configuration file. Later import of configuration files is possible.

- **Import Settings**

  Import of a configuration file to restore the saved settings.

- **Update License File**

  Dialog to choose a (new) license file. Content of recent and new license file are shown and copying of the new file is possible.

- **Check for Updates**

  Inquiry to the web server for new versions of the CANopen Update Manager. Only the IP Address is transmitted, no further data.

Help

- **Manual**

  Shows the manual in HTML format.

- **Changelog**

  Shows the changelog.

- **About**

  Shows the „About-dialog“ with license information.

- **About Qt**

  Information about the Qt framework with license information of the used Qt components.

7. Automated firmware download via command line

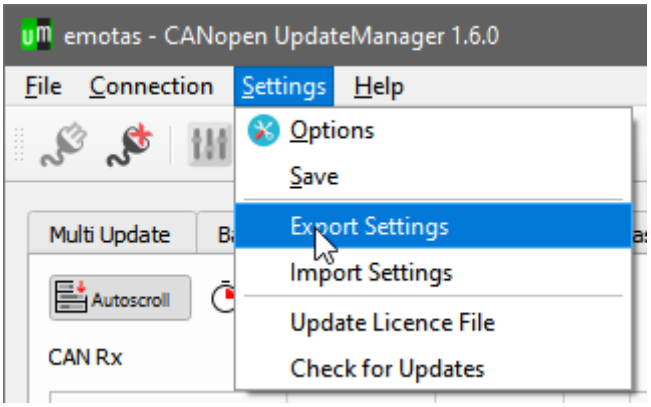
The CANopen Update Manager can be started via the command line for automated updates.

7.1 Creating and loading of a configuration file

The CANopen UpdateManager always starts with its old settings. To use specific settings the tool offers the option to export the current settings and load it at start via command line.

Export a configuration file

To export a configuration file you have to select “Setting -> Export Settings” within the menu.



Now all settings from the “Basic Update” and the CAN interface settings are exported.

Load the configuration file via command line

To load the configuration file you have to start the CANopen UpdateManage with the parameter “--loadConfig filepath”.

7.2 Autostart of the update

The CANopen UpdateManager starts the update automatically if "--autoStart" is passed as parameter.

In combination with the parameter "--loadConfig" an individual update can be executed automatically.

The following example shows a complete call:

```
coum.exe --autoStart --loadConfig C:\Daten\UpdateSettings.umconf
```

7.3 Return value

The called CANopen UpdateManager returns one of the following error codes:

Rückgabewert	Beschreibung
0 – No error	No error - update successful
1 – Update status uncertain	There was no error, but it could not be confirmed that the update was successful.
2 – No bootup of the application	Firmware downloaded, but no bootup of the application could be received.
3 – No bootup of the bootloader	The update could not be started because no bootup of the bootloader was received.
4 – Application not running	The update could not be started because no bootup of the bootloader was received.
5 – Wrong password	The password is not correct
6 – SDO error 0x1F50	SDO error at object 0x1F50
7 – SDO error 0x1F51	SDO error at object 0x1F51
8 – SDO error 0x1F57	SDO error at object 0x1F57
9 – General SDO error	A general SDO error occured
10 – LSS error	A LSS error occured
11 – Bootloader error	The bootloader has not sent a specific CAN ID.
12 – Seriell connection error	An error occurred when reconnecting to the serial interface.
13 – Invalid Vendor ID	The device has an invalid vendor ID.
14 – No CAN connection	There is no active CAN connection.
15 – Update canceled	The update was cancelled by the user.
16 – INI file error	The INI file does not match device properties
17 – SDO error	An SDO error occurred while reading or writing.
18 – Flash Erase Timeout	The flash could not be erased within the time period.
19 – Applikation ID error	Object 0x156F has an invalid value
20 – Firmware file not found	The firmware file could not be found.
21 – Config file not found	The configuration file could not be found.

Complete call with return value

The following call is possible via Windows command line to query the return value:

Input:

```
start /wait coum.exe --autoStart --loadConfig C:\Daten\UpdateSettings.umconf
echo %errorlevel%
```

Output:

```
0
```

8. Comparison between bootloader objects and firmware file

Within the UpdateManager it is possible to compare values from the bootloader of the device with values of the firmware file. For this feature the UpdateManager tries to read a .ini file with the same name as the firmware file in the directory of the firmwarefile or a file called “compare.ini” in the install directory.

If one of this succeeds the UpdateManager shows the values of the bootloader and the firmware file so the user can decide to continue the update or cancel it.

8.1 Structure of the ini file

The ini file can hold multiple entries. Every entry starts with a corner bracket which holds the name of the entry, e.g. [Vendor Id].

Afterwards there have to be five properties for each entry. Those properties are described in the table below.

	Description	Example
Index	Index of object	0x1018
SubIndex	Subindex of object	1
DataType	Datatype of the object	u16, u8, INTEGER8 etc.
DataOffset	The offset for the entry at the firmware file in bytes as decimal number.	8
CompOp	The comparison operator describes in which case the comparison will be true. The value of the firmware file is mathematically left of the operator. If the comparison is true the line of the compare dialogue will be displayed green otherwise it will be red.	„>“, „>=“, „<“, „<=“, „=“, „!=“

8.2 Example

The image shows the first part of a firmware file.

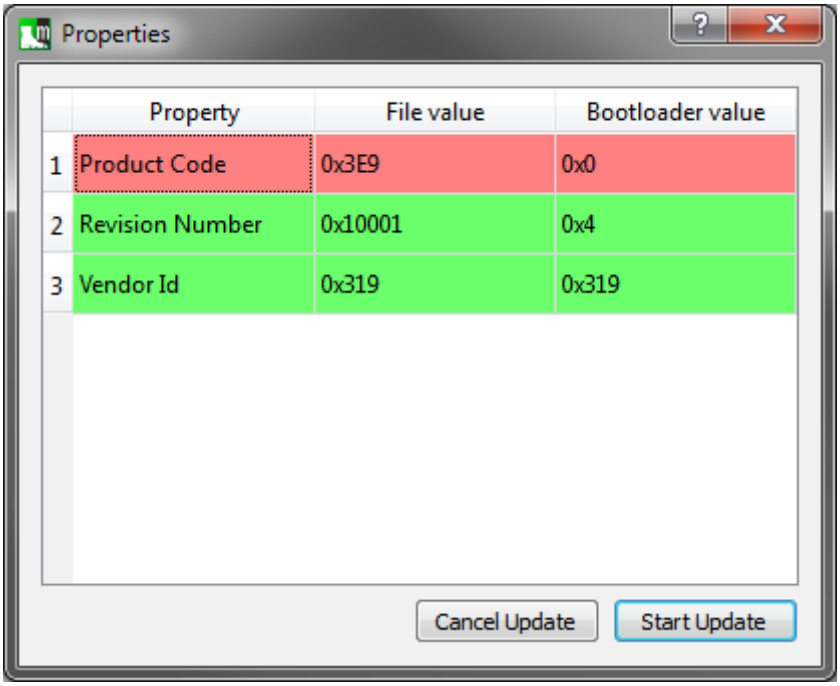
```
Offset(d)  00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
00000000   2D 01 00 00 82 D3 FF FF 19 03 00 00 E9 03 00 00  -...,Óÿÿ....é...
00000016   01 00 01 00 00 01 03 00 FF FF FF FF FF FF FF FF  ........ÿÿÿÿÿÿÿÿ
00000032   FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
```

This is like a .ini file may look.

```
[Vendor Id]
Index=1018
SubIndex=1
DataType=u32
DataOffset=8
CompOp="="

[Product Code]
Index=1018
SubIndex=2
DataType=u32
DataOffset=12
CompOp="<"

[Revision Number]
Index=1018
SubIndex=3
DataType=u32
DataOffset=16
CompOp=">"
```



9. Support & Contact

On all questions and upcoming problems on CANopen UpdateManager you may contact us via email (support@emotas.de) or by phone +49(0)3461/794160. If a CANopen device does not react as expected, a logging of the CAN communication is useful for the analysis. Please send us your current CAN logging by email, ideally also before you contact us by phone.