Application Note

Subject:	**Example how to read and write via CANopen**
SICK Product:	RFH6xx
Department:	Flexible Identification
Creation Date:	2023/10/13

Version history

Version	Date	Author	Remarks
1	2023/10/13	AE-FI	Initial Version
2	2024/09/02	AE-FI	Added how to trigger device by toggling the corresponding index

Table of content

1. About this document

This application note describes a possible way how to set up a connection between the RFH6xx and a PC via CANopen.

First of all the used Software and Hardware is described. Afterwards there are examples how to communicate via CANopen with the device. Such as read and write data to a transponder or how to trigger the device.

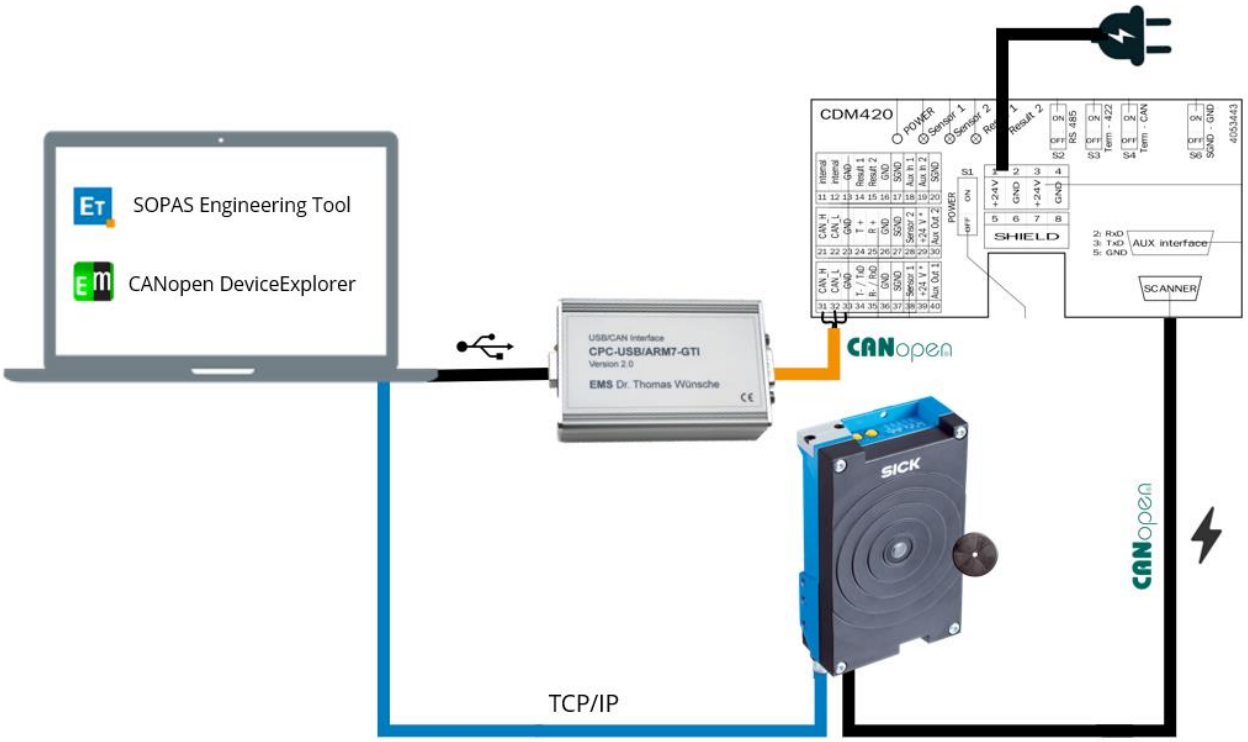
2. Used setup

The setup used for this document is only an example and a possibility to establish the connection via CANopen.

2.1. Hardware

The used RFH63x is connected to a connection box CDM420. The CDM420 is powered via a power supply unit and supplies the RFH63x with energy. At the same time, communication via CANopen takes place via this connection.

A CAN-USB adapter is used to pick up the CANopen signal at the CDM420 and pass it on to a PC via the USB interface. In addition, the RFH63x is connected to the PC via an Ethernet cable. The following figure shows a rough sketch of the setup.



2.2. Software

To configure the RFH63x it's necessary to use the SOPAS Engineering Tool, which can be found on sick.com.

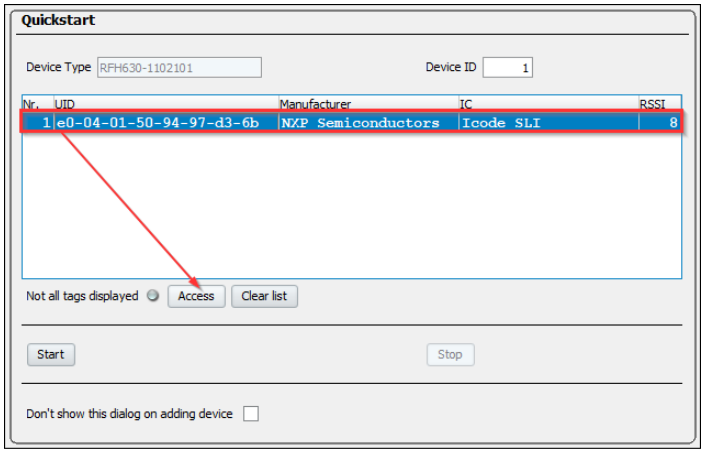
For the communication described in this document the program „CANopen DeviceExplorer" from emotas is used. This program offers to communicate via CANopen. It is free available and can be used for free with a one-hour trial license, that can be refreshed repeatedly.

3. Sopas configuration

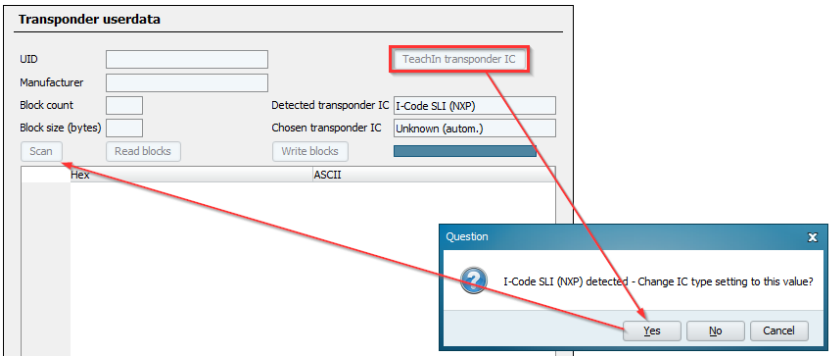
Open the SOPAS Engineering Tool and search for connected devices. Open the configuration page by double clicking on the device. Log in with the userlevel Service to have access to all parameters. The password is “servicelevel”.

3.1. Transponder TeachIn

Put a transponder on top of the RFH6xx and start reading at the Quickstart page. Stop the reading, click on the detected transponder and click on the Access-button afterwards.



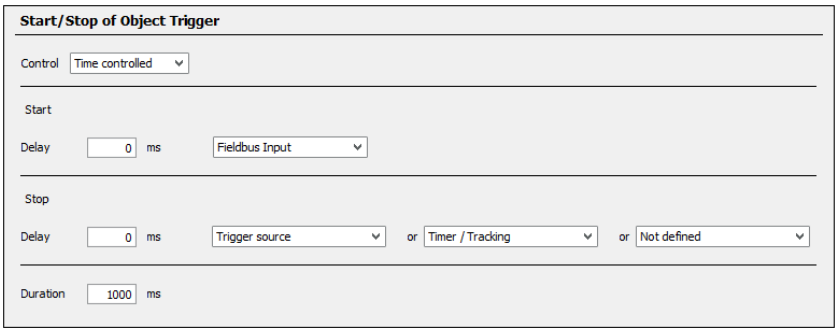
Click on the TeachIn-button and scan afterwards.



The transponder type is now taught in.

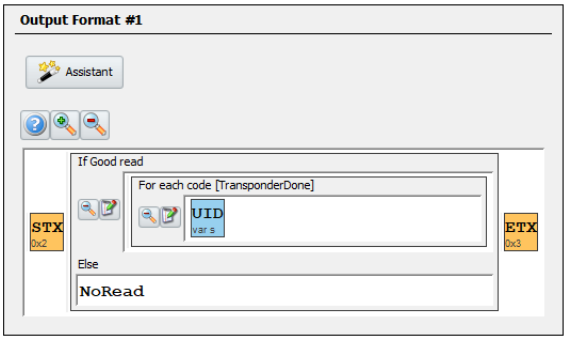
3.2. Trigger

Next step is to set up the trigger conditions. If the device is triggered via a PDO it is necessary to set “Fieldbus Input” as start condition for the trigger. If you trigger the device via SOPAS command choose “Command”. For the first example we start with this settings:



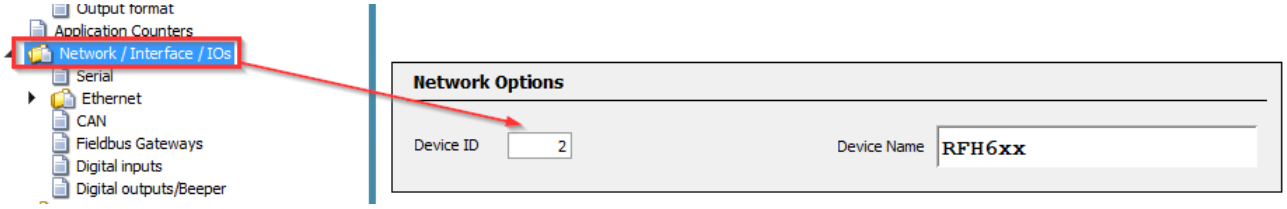
3.3. Output Format

The Output Format defines the output that will be send out from reader after a triggered reading gate. Here you can choose what kind of data you want to output. We start with the default Output Format 1.

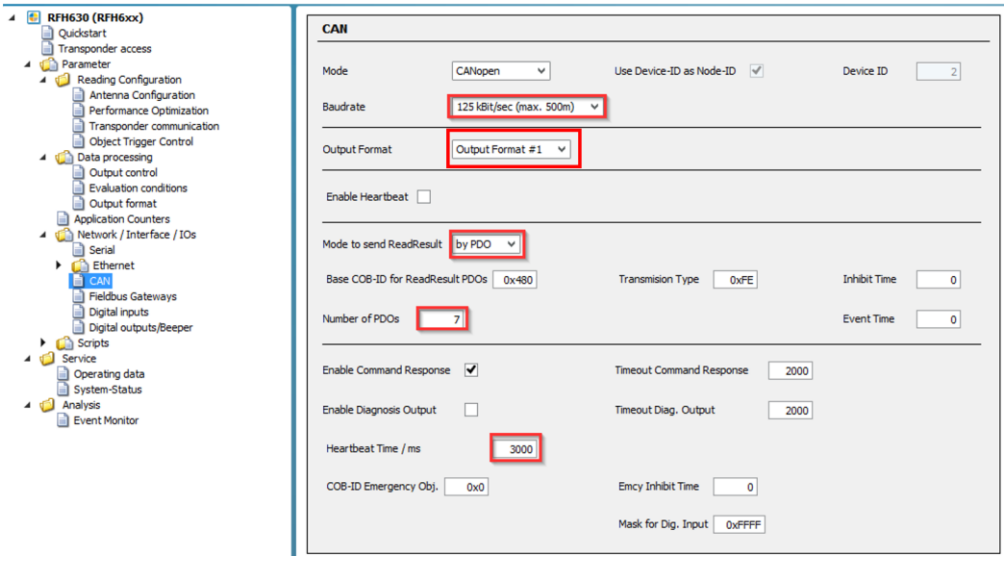


3.4. CANopen configuration

The Node-ID of the device is the same as the Device-ID. This ID can be modified at the “Network / Interface / IOs” page.



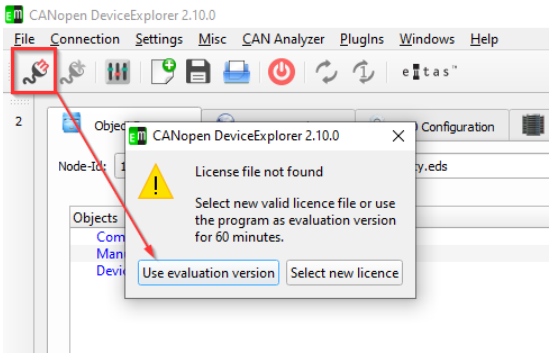
The CANopen configuration can be done on the “CAN” page. The free version of CANopen DeviceExplorer only supports a baudrate of 125 kBit/sec. It is possible to set a heartbeat time. We use a heartbeat time of 3000ms and start with the following configuration. Parameters that are changed from default are marked red.



⚠ Be aware that the maximum output size of a Readresult by PDO is 49 byte. Overlap will be cut.

4. Establish connection with CANopen DeviceExplorer

Make sure the CAN-USB-adapter is plugged into the PC and open the CANopen DeviceExplorer. Connect the device and use the evaluation version. This free trial license can be used for 60 minutes. It is possible to renew the license again and again.

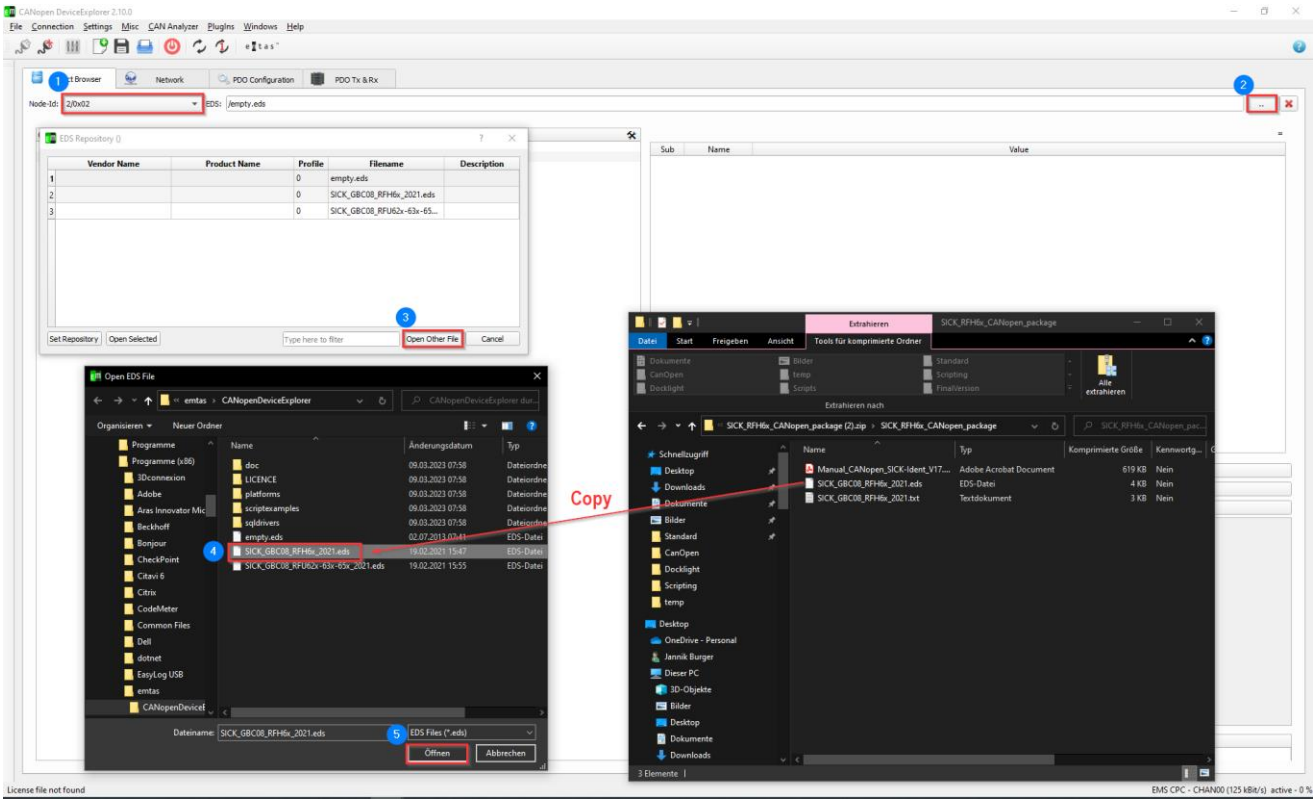


4.1. Load EDS file

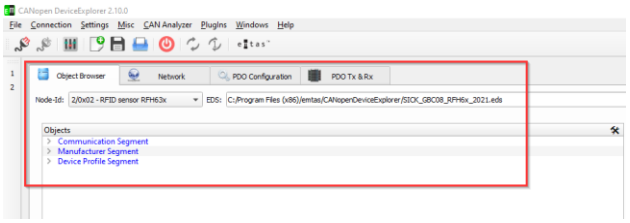
The program offers the possibility to upload an EDS file, which contains device specific objects. The EDS files can be found on the Support Portal: https://supportportal.sick.com/downloads/eds-files-can-open/

Download the EDS file of the RFH62x-63x and unzip the folder.

Choose the right Node-id of your device in the CANopen DeviceExporer and import the EDS file. Therefore it's recommended to copy the EDS file into the shown folder path:

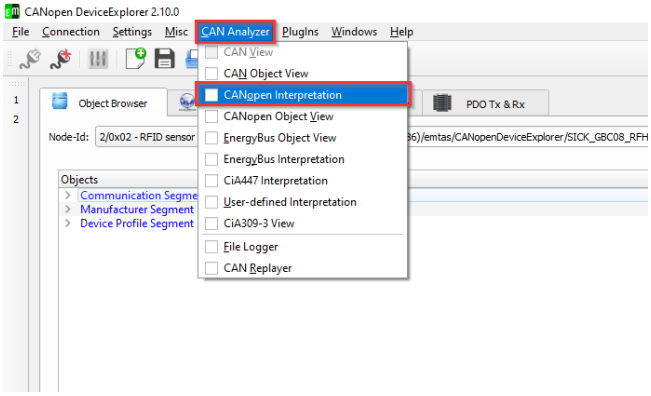


It should look like this now:

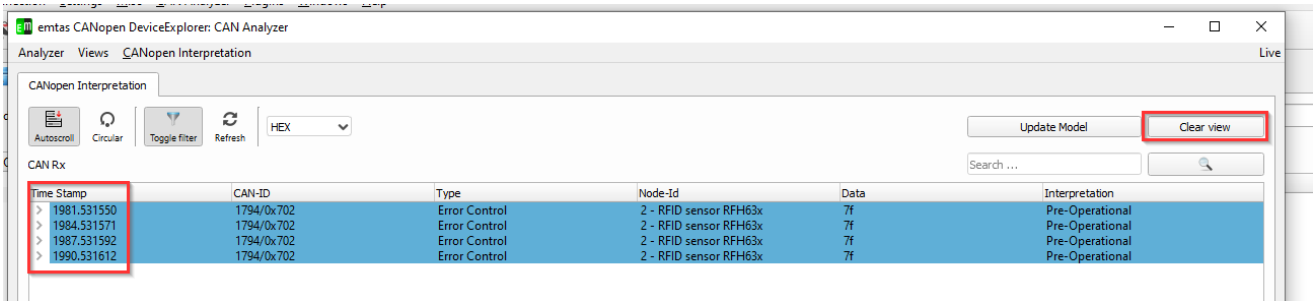


4.2. Open CANopen interpretation

Open the CANopen interpretation page to see which commands are sent and received.



Clear the view and check if the heartbeat is coming in every 3 seconds.

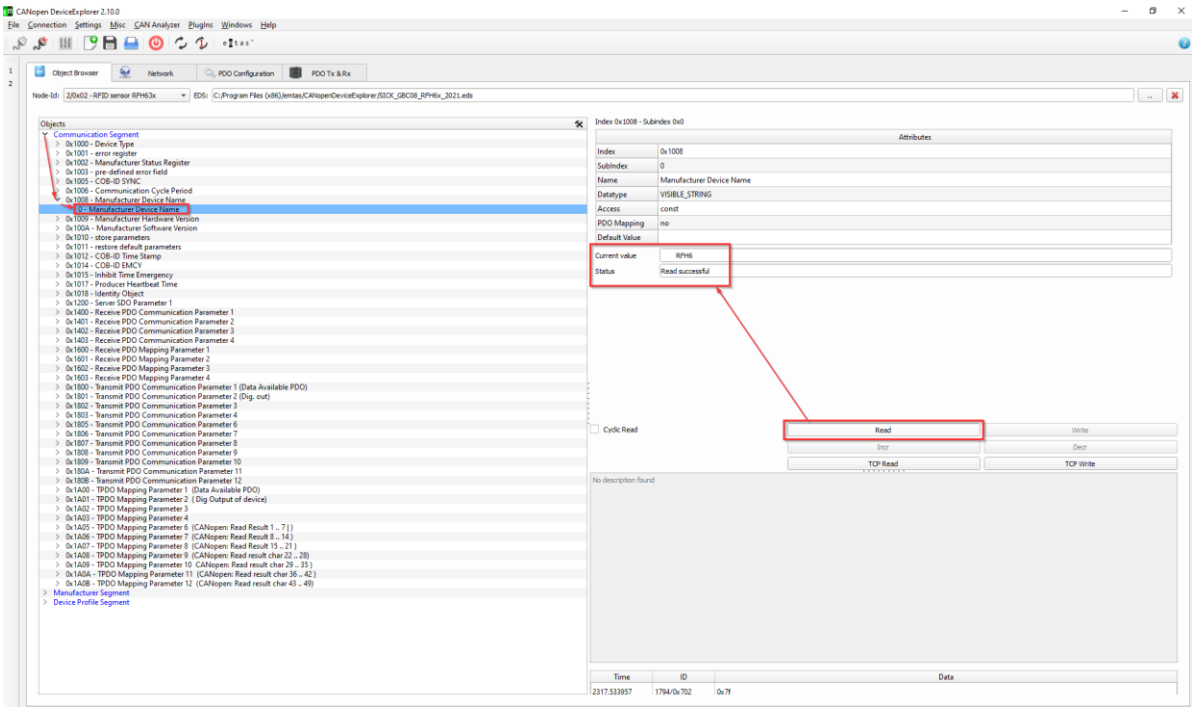


5. Communication examples

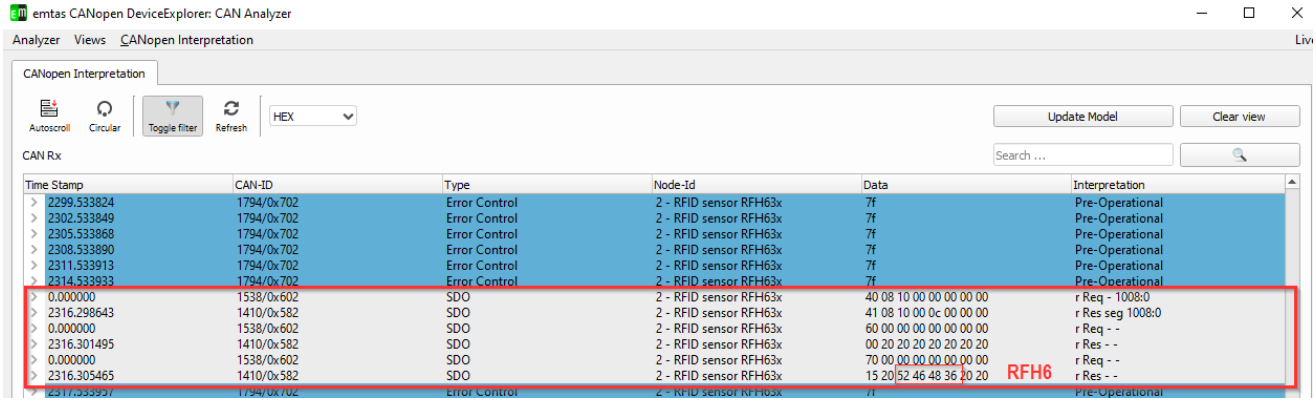
This chapter provides several Communication examples on how to read and write data.

5.1. Read device name

Due to the import of the EDS file the program offers an easy access to device specific object like the device name for example. This can be found at index 0x1008 and be read out as shown below.

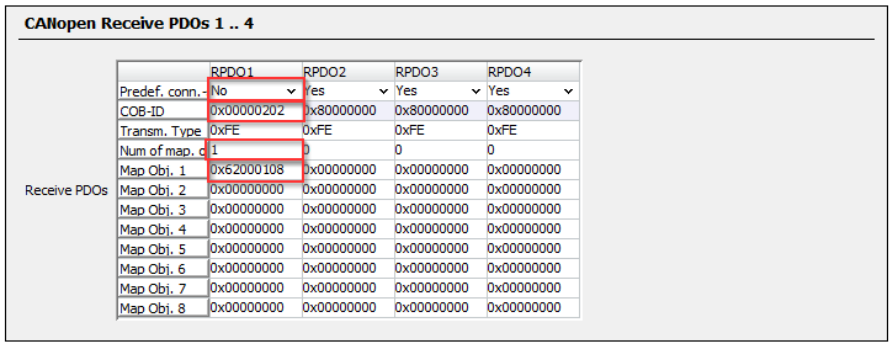


The command can be seen in the CANopen interpretation view.

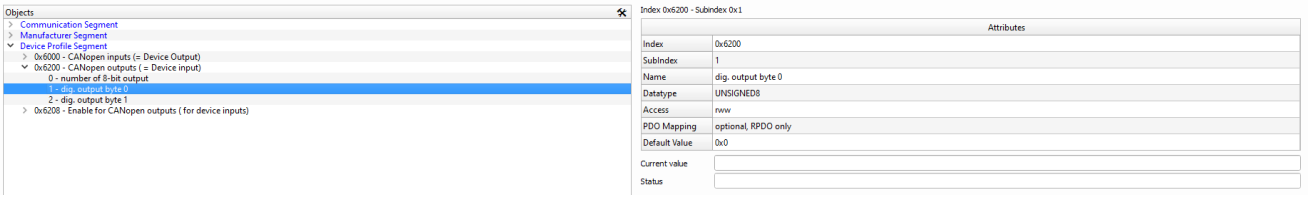


5.2. Trigger via PDO

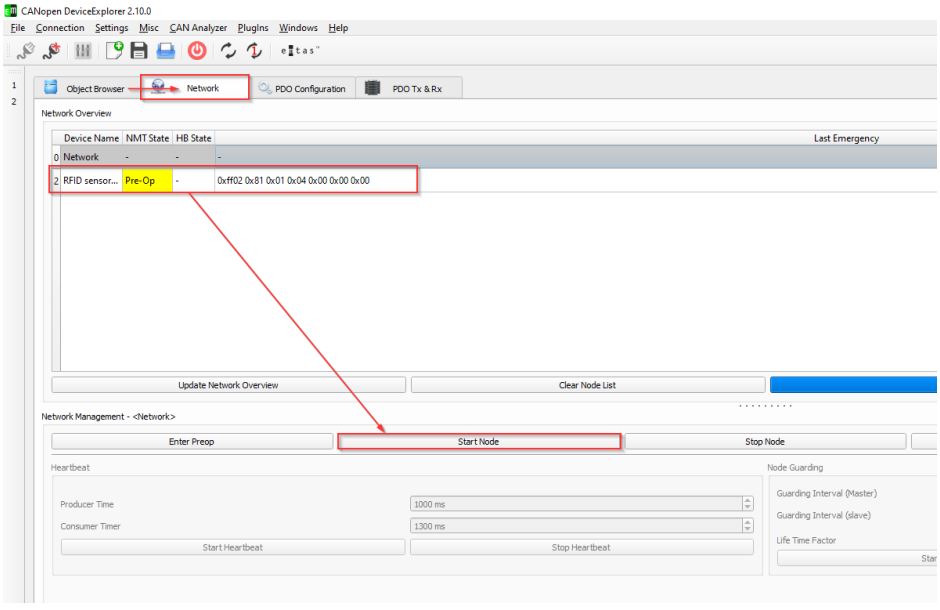
To trigger the device via PDO it is necessary to create a Receive PDO in the SOPAS configuration.



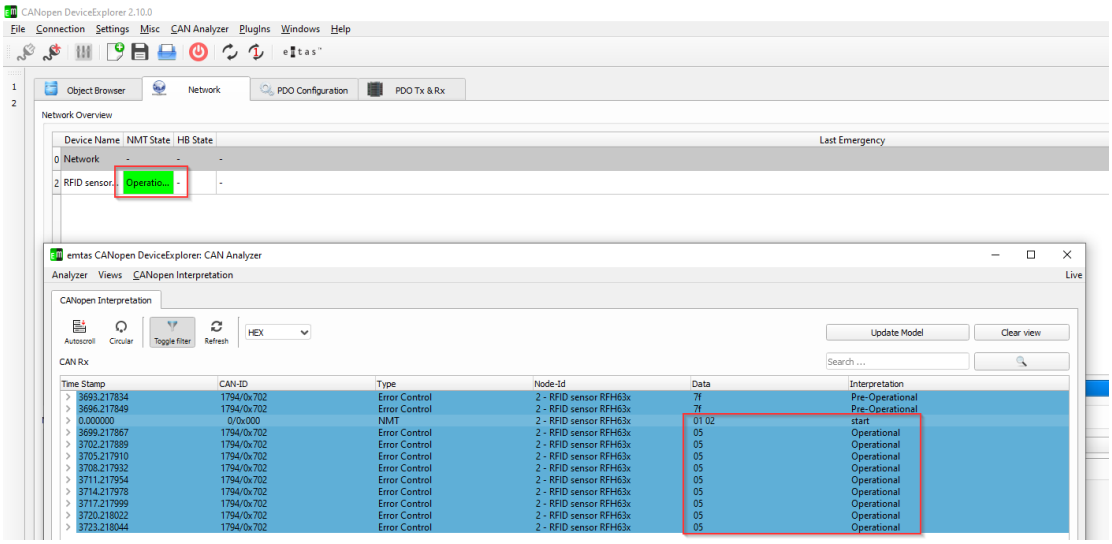
The created RPDO receives the low byte of the digital output object 0x6200/01 (Length = 08 bits). This RPDO is assigned to COB-ID 0x202. The last number (“2”) stands for the CAN ID.



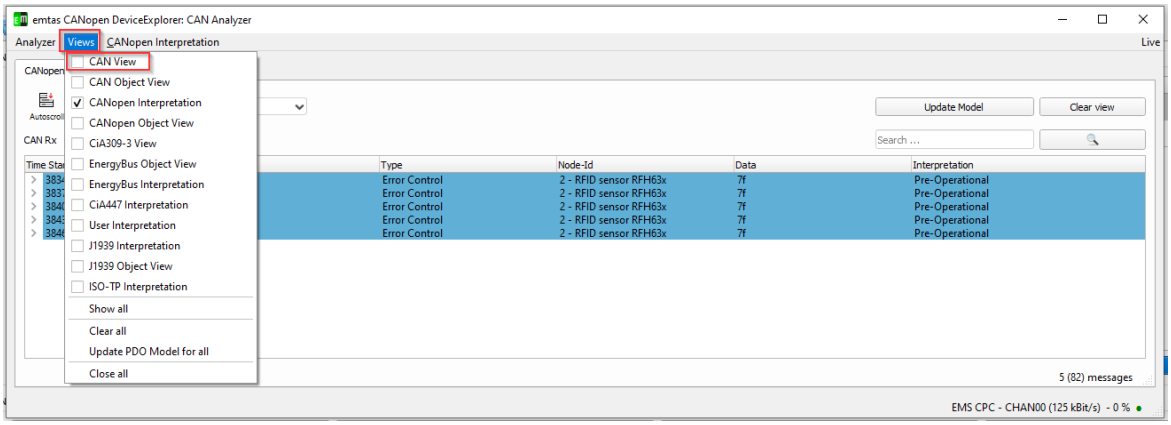
Next step is to set the device into operational mode. There are two ways to do that. First way is to switch to the network page and start the node:

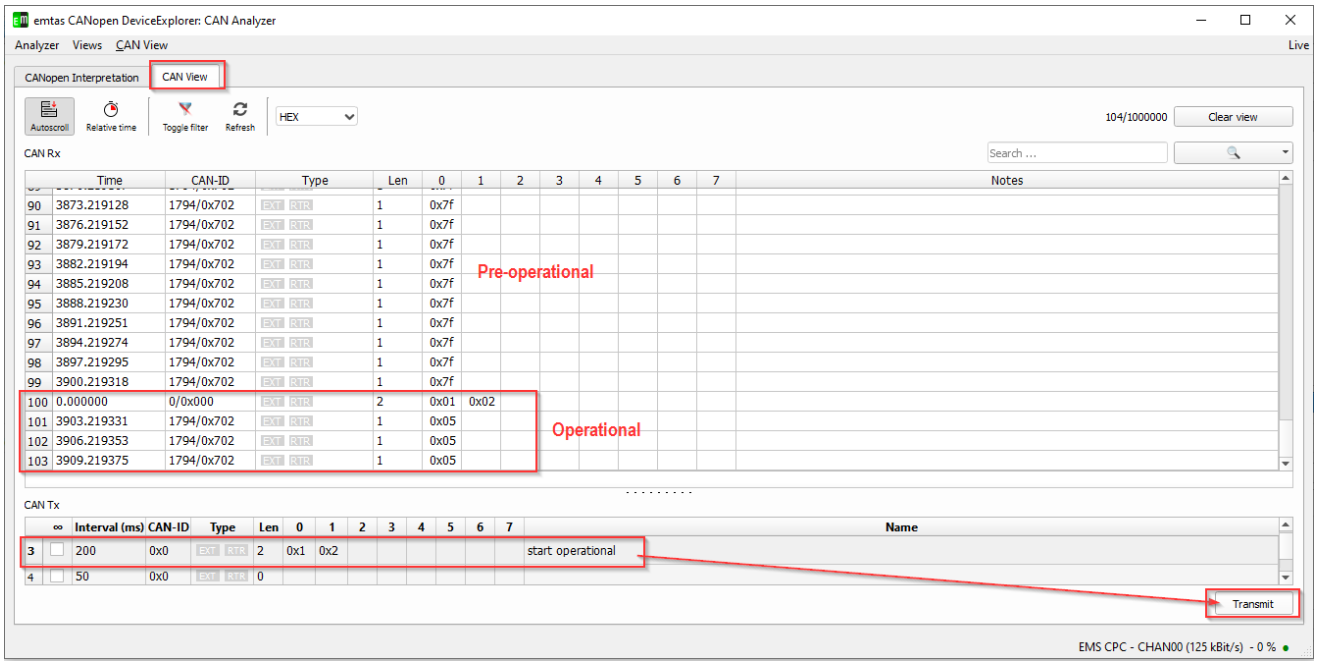


The device is now in operational mode.



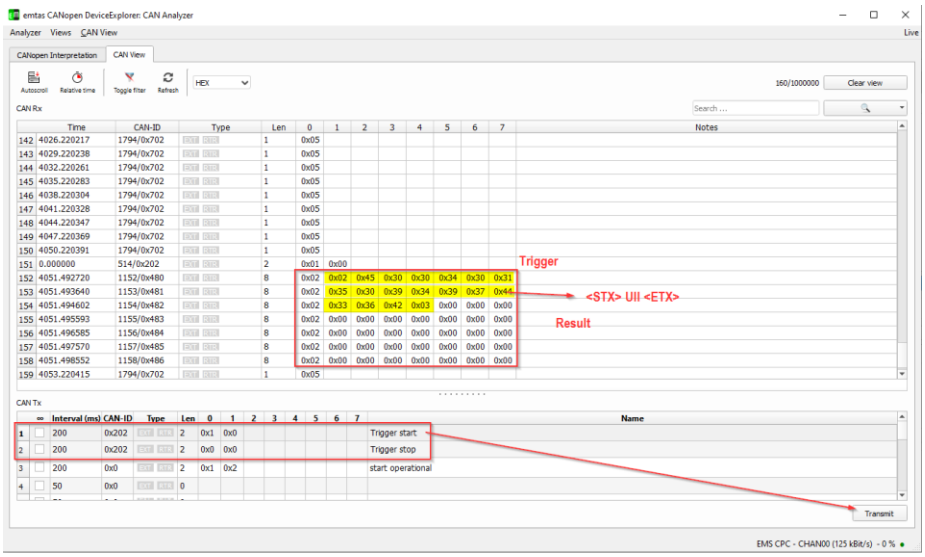
The second way is to send a PDO command to start the operational mode. Therefore switch to the CAN viewer and send following command:





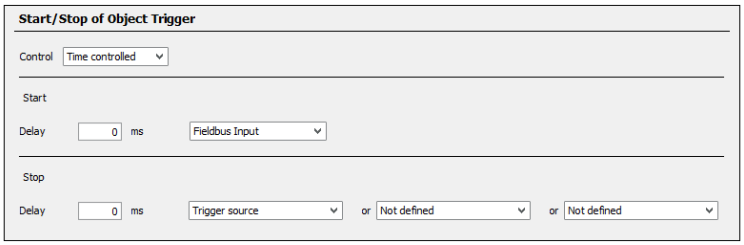
Now make sure the trigger start condition in SOPAS is set to “Fieldbus Input”.

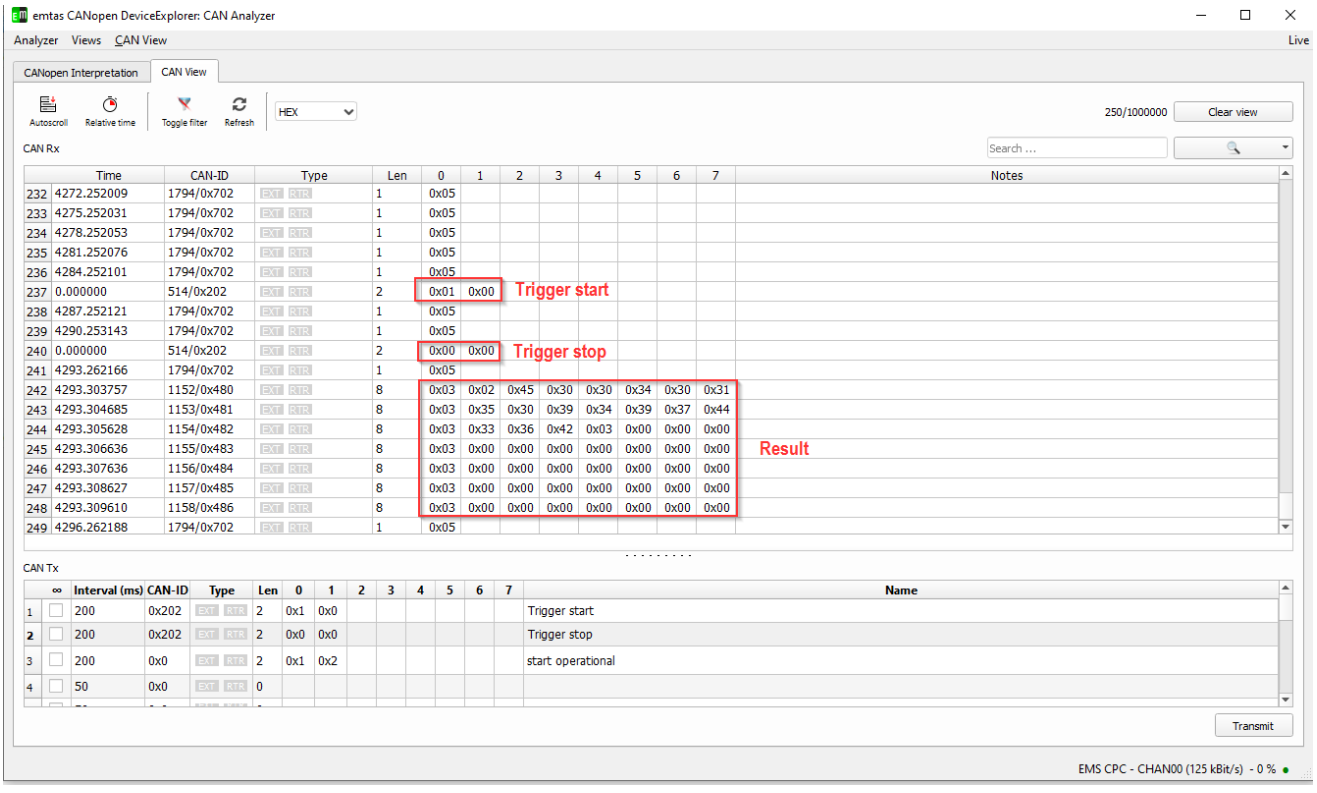
Create following PDO commands to trigger the device:



It is always necessary to send trigger stop (toggle Index 0x6200 sub 1) by toggling back from 1 to 0. This is also necessary if trigger stop by GoodRead is selected!

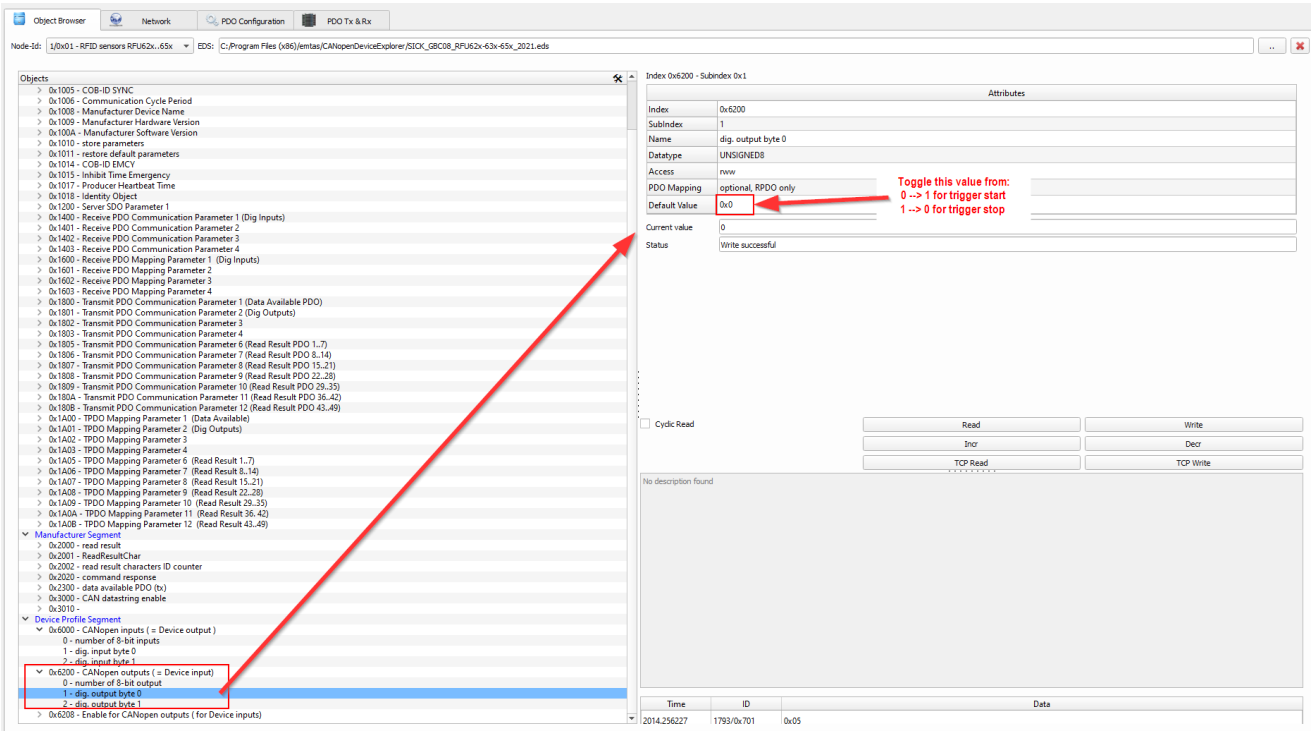
Difference between trigger stop by “GoodRead” and trigger stop by “Trigger source” is the output time of the result. If “GoodRead is selected the result is output as soon as a GoodRead occurs. With trigger stop by “Trigger source” the result will be output after the trigger stop command (toggling back to 0)





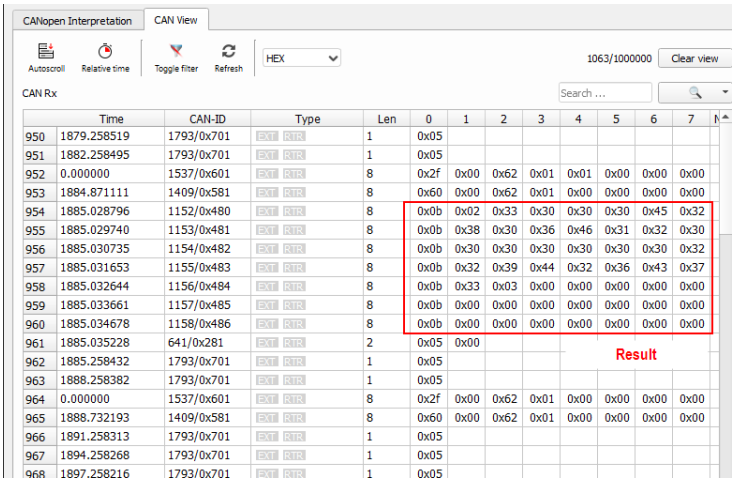
The Output Format can be freely adjusted. So it's possible to read UIDs and content of the UMEM.

It is also possible to trigger the device via the corresponding index:



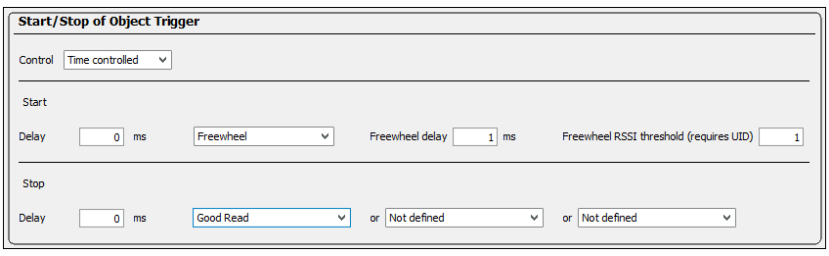
Trigger start: toggle from 0 → 1

Trigger stop: toggle from 1 → 0

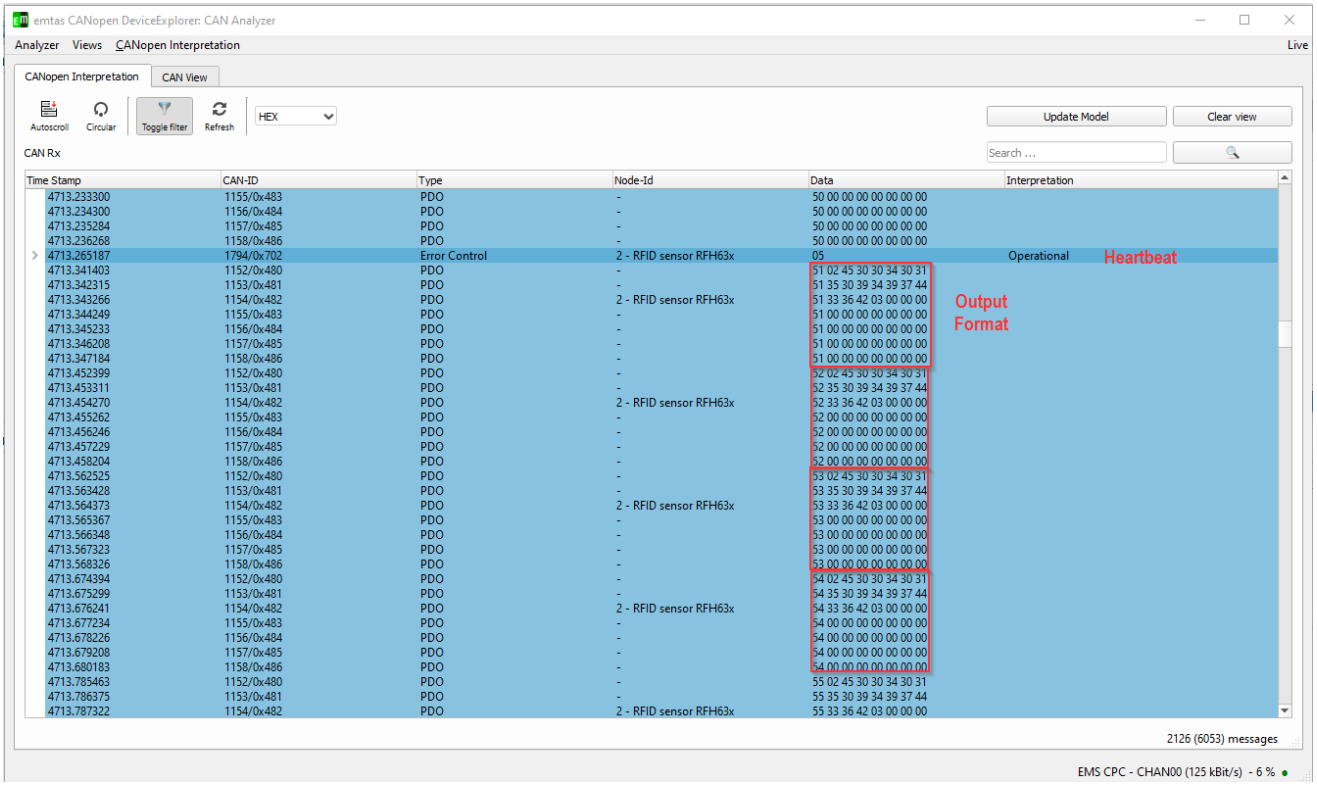


5.3. Use of freewheel and receiving data

It is also possible to run the RFH6xx in freewheel mode and receive the defined Output Format via CANopen. Therefore activate the freewheel mode in SOPAS.



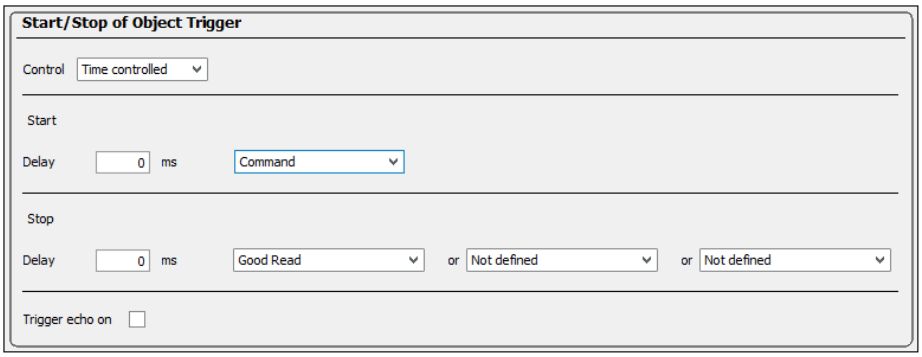
You will get the data via CANopen.



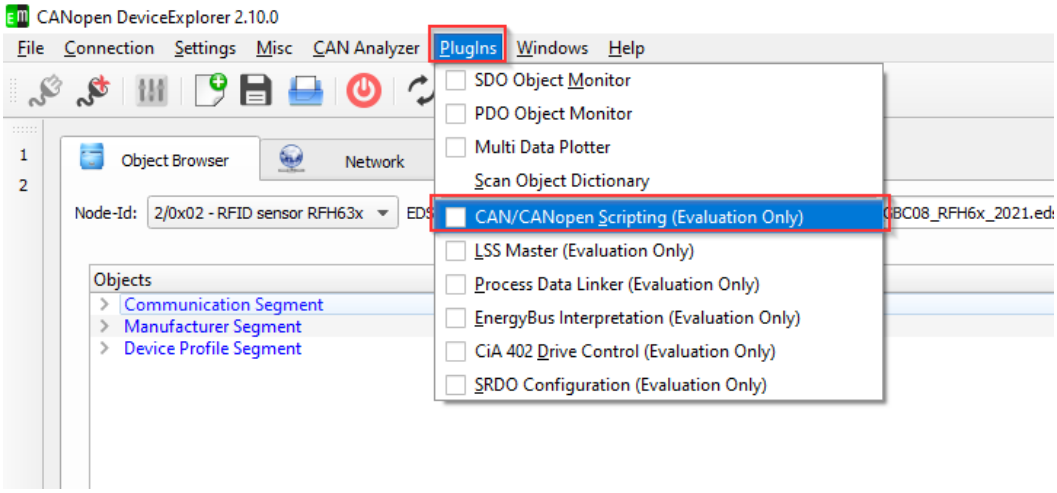
5.4. Trigger via SOPAS command

It is also possible to trigger via a SOPAS command by SDO communication. Therefore you have to handle a SDO protocol. This can be done by using the provided script.

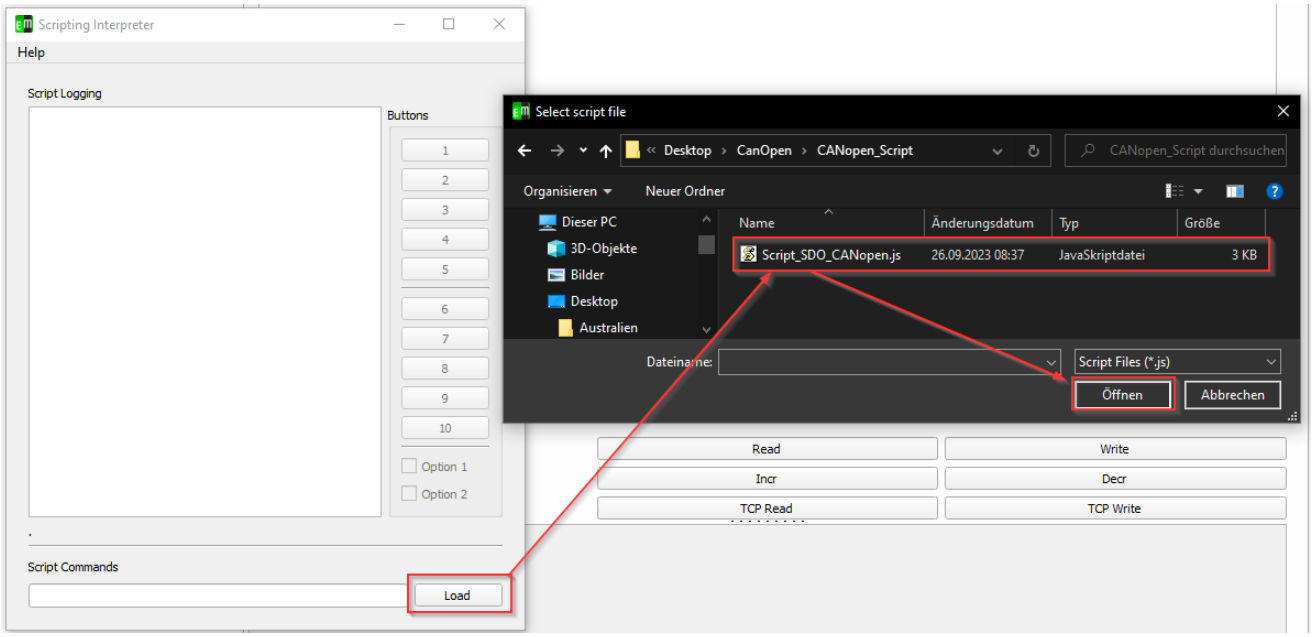
To trigger via SOPAS command it is necessary to change the trigger start condition to “command”.



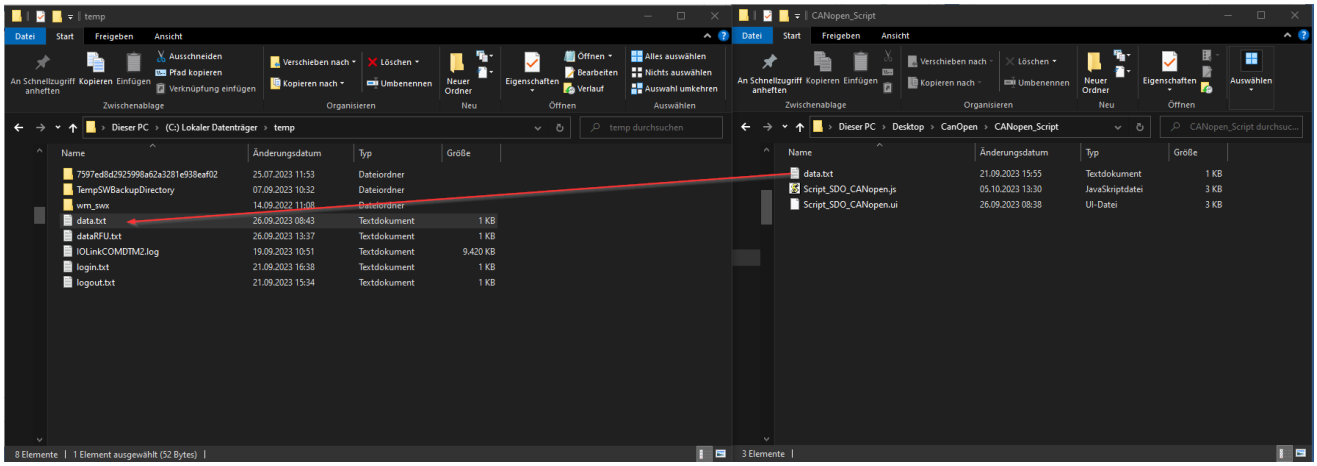
Open the CANopen DeviceExplorer and connect to the device. Open the CANopen Interpretation and the CAN/CANopen Scripting.



Now load the provided script.

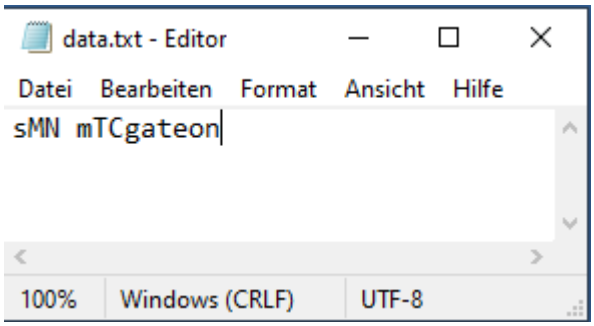


It is important to copy the data.txt into the C:\temp folder:



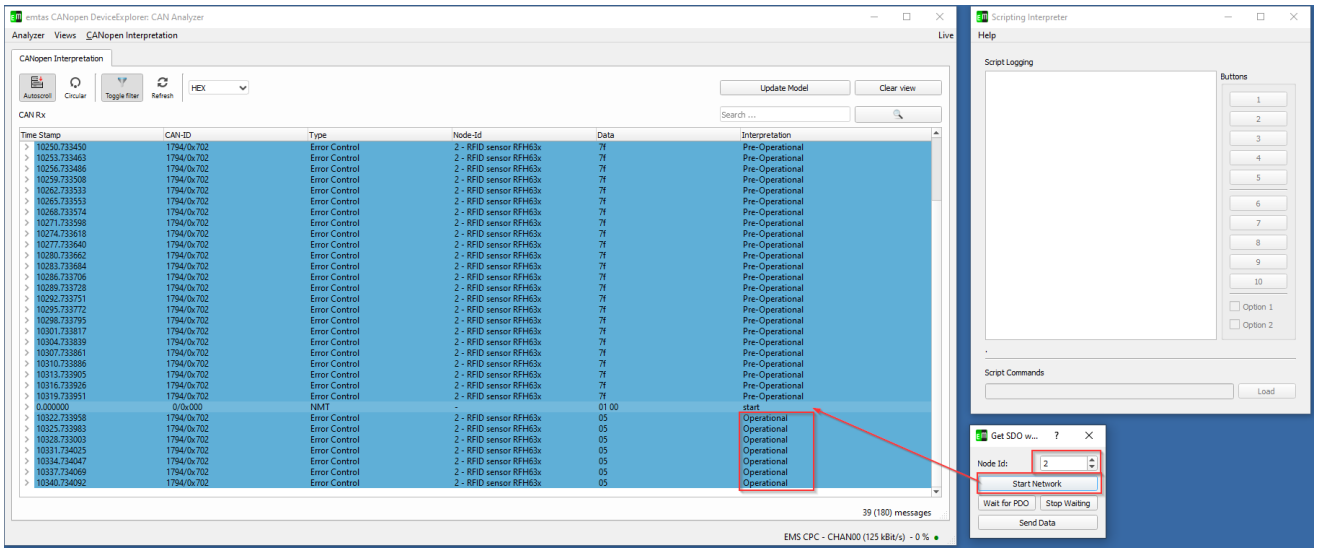
Open the data.txt file and insert the following trigger-command:

sMN mTCgateon

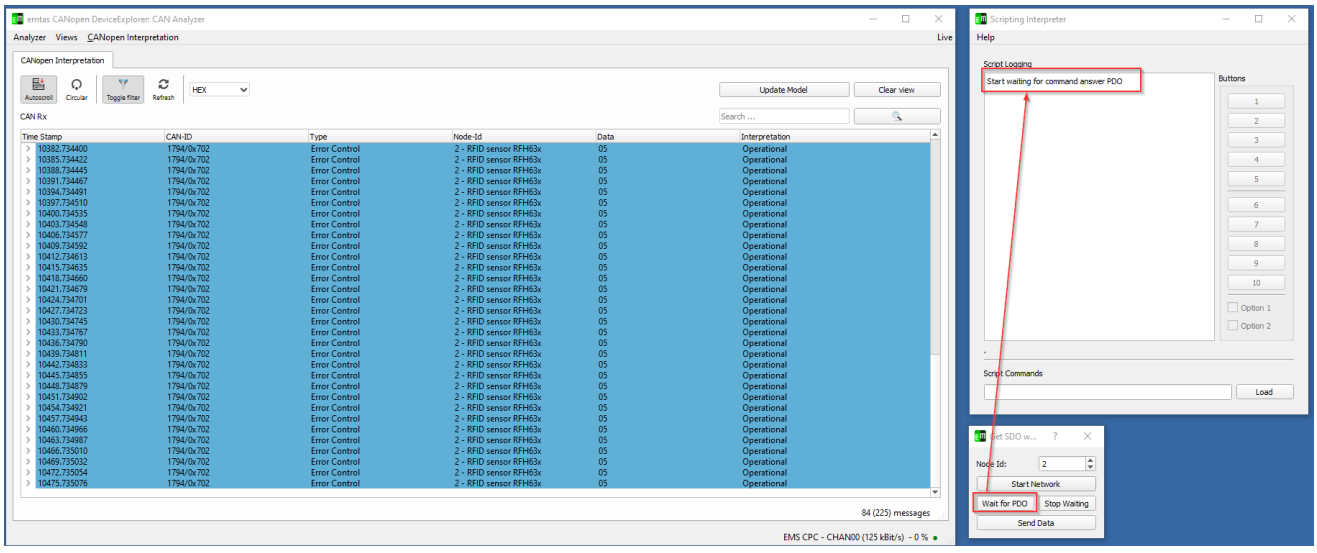


And save the data.txt file.

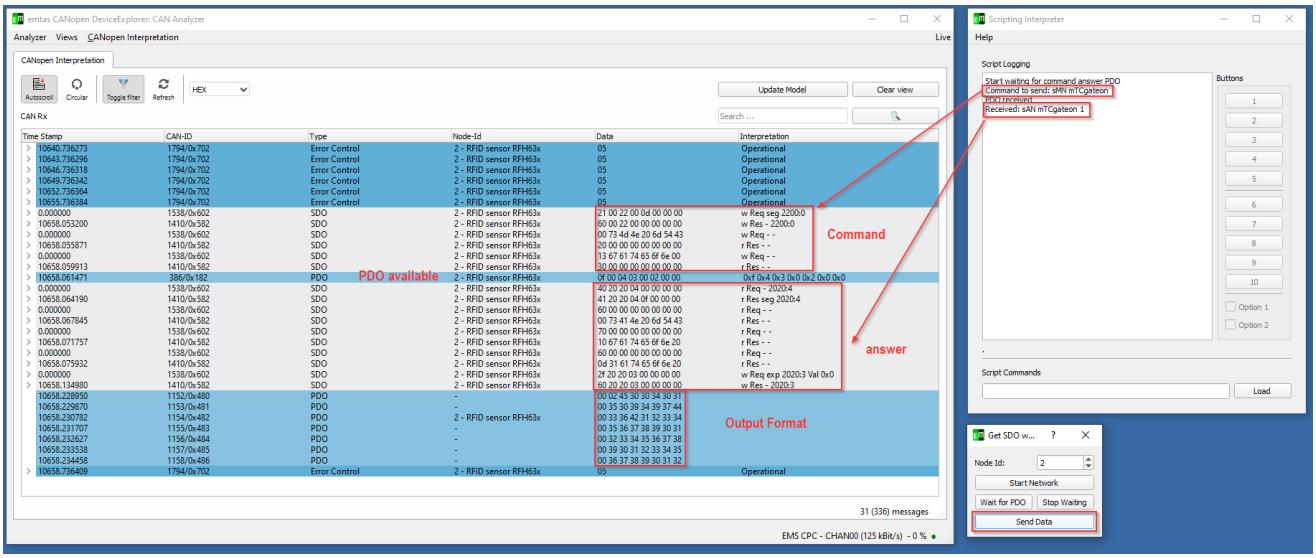
Now choose the Node id you are using in the GUI of the Script. And start the network which means to set the device into operational mode.



Click on the “Wait for PDO” button to listen for available answers and pick them up if possible.



Now click on “Send data” to send the content of data.txt as command to the device via SDO.



In the CANopen Interpretation you can see the commands and results in HEX. The Scripting Interpreter shows the sent command and the received answers in ASCII.

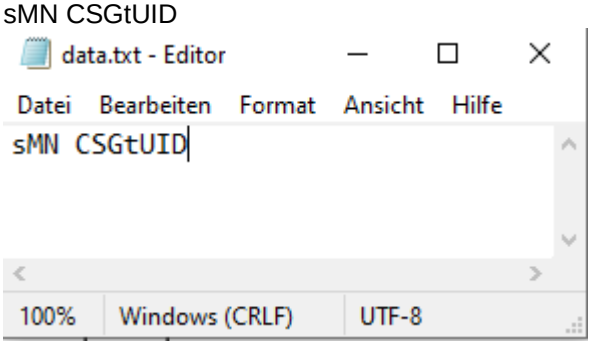
5.5. Read transponder via SOPAS command

It is also possible to read transponder data by using SOPAS commands. Here are examples of how to read UID and block content.

5.5.1. Read UID via SOPAS command

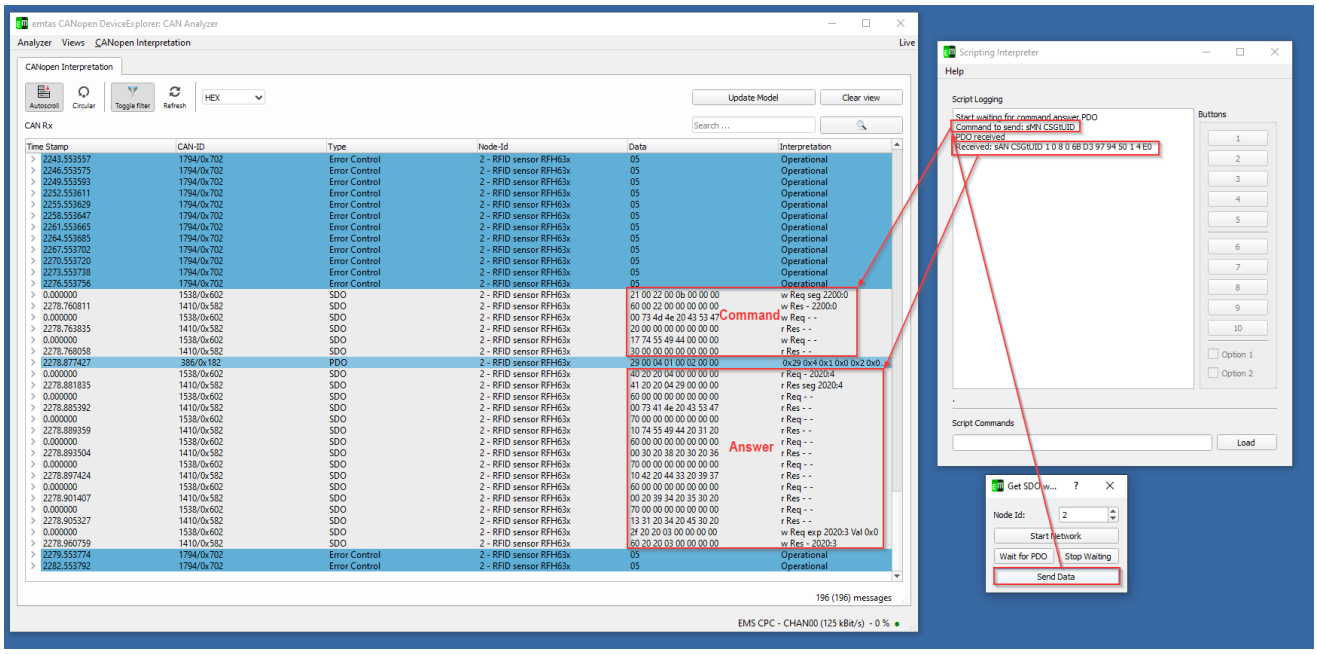
To read the UID of a transponder copy following command into the data.txt file in the C:\temp folder and save the file.

sMN CSGtUID



(Open the Script if it's not already running, start the Network and wait for PDO. If the Scrip is not running have a look at chapter 5.4)

Send the command to read the UID by clicking on "send data".



The answer sAN CSGtUID 1 0 8 0 6B D3 97 94 50 1 4 E0 means the following:

sAN: indicates a SOPAS command answer

CSGtUID: name of the SOPAS command

1: Length of error

0: no error occurred

8: RSSI value

0: DSFID

6B D3 97 94 50 1 4 E0: UID backwards

5.5.2. Read one block of UMEM via SOPAS command

Use following command to read one block of the UMEM:

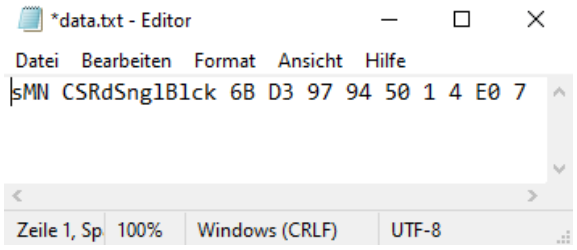
sMN CSRdSnglBlck 6B D3 97 94 50 1 4 E0 7

red: UID of transponder backwards. Ending with E0 (0 0 0 0 0 0 0 0 will read unaddressed)

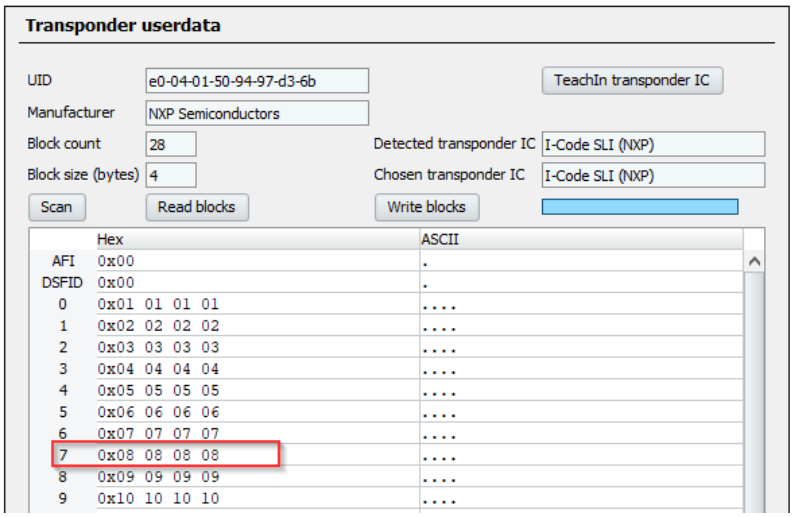
blue: the number of the block that shall be read

With this command we will read out the block content of the seventh block of the mentioned transponder.

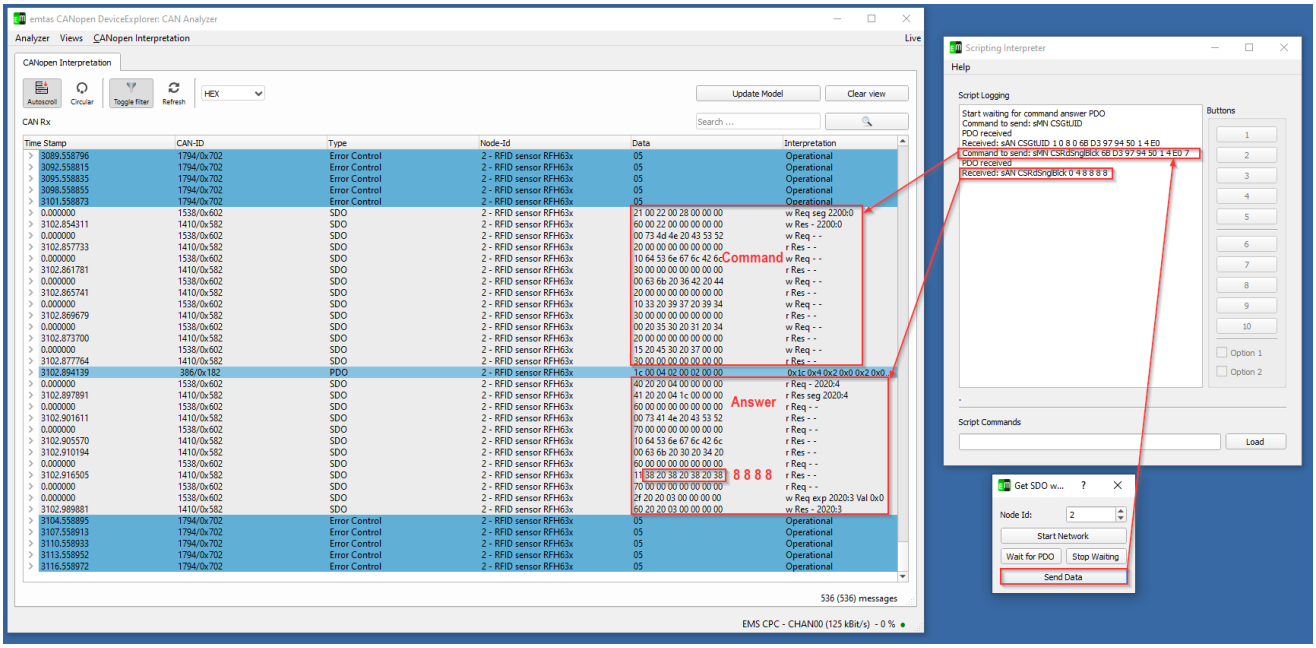
Write the command into the data.txt file at C:\temp and save the file.



According to the SOPAS Transponder access the content we expect to read is 0x08 08 08 08



After sending the data we get the following answer.



The answer sAN CSRdSnglBlck 0 4 8 8 8 8 means the following:

sAN: indicates a SOPAS command answer

CSRdSnglBlck: name of the SOPAS command

0: no error occurred

10: Length of data that has been read in HEX.

8 8 8 8: data of the read block

5.5.3. Read multiple blocks of UMEM via SOPAS command

Use following command to read multiple blocks of the UMEM:

sMN CSRdMltBlck 6B D3 97 94 50 1 4 E0 2 3

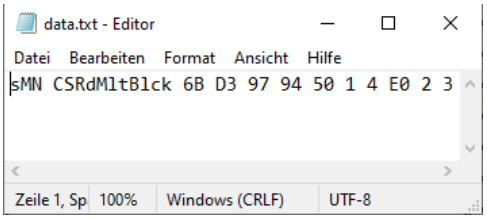
red: UID of transponder backwards. Ending with E0 (0 0 0 0 0 0 0 0 will read unaddressed)

blue: number of first block to read, block count starts at block 0

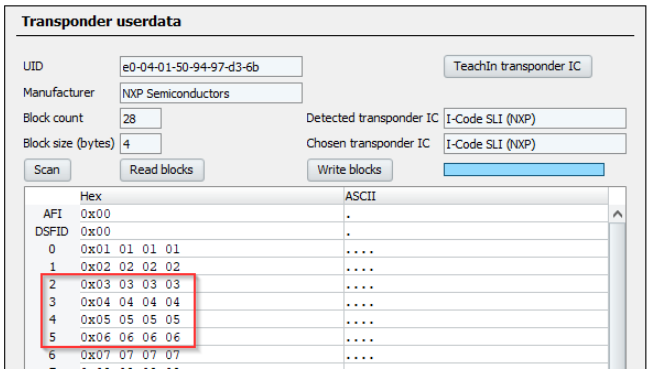
green: amount of blocks that shall be read. Enter number of blocks minus one! So 3 will read 4 blocks

This command will read 4 blocks starting from block 2 of the mentioned transponder.

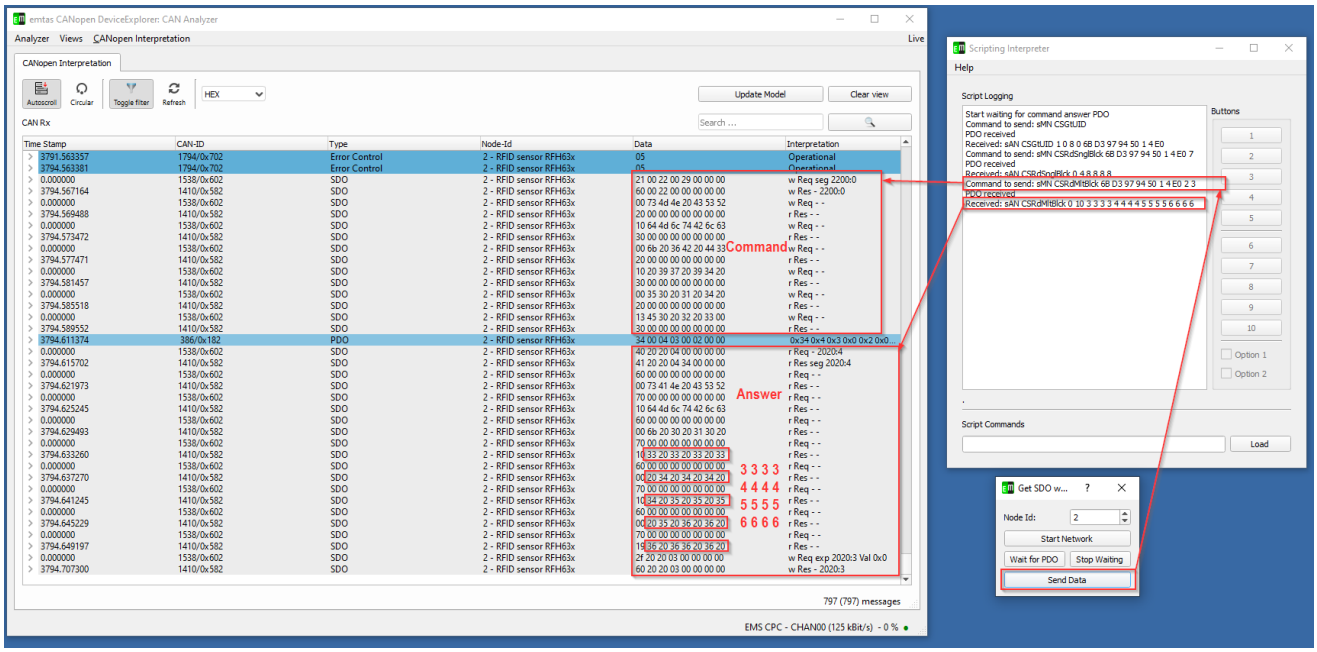
Write the command into the data.txt file at C:\temp and save the file.



According to the SOPAS Transponder access the content we expect to read is 0x03 03 03 03 04 04 04 04 05 05 05 05 06 06 06 06



After sending the data we get the following answer.



The answer sAN CSRdMltBlck 0 10 3 3 3 3 4 4 4 4 5 5 5 5 6 6 6 6 means the following:

sAN: indicates a SOPAS command answer

CSRdMltBlck: name of the SOPAS command

0: no error occurred

10: Length of data that has been read in HEX. So 16 DEC byte

3 3 3 3 4 4 4 4 5 5 5 5 6 6 6 6: data of the read blocks

5.6. Write UMEM via SOPAS command

It is also possible to write data to tag by using SOPAS commands.

5.6.1. Write one block

Use following command to write one block of the UMEM:

sMN CSWrtSnglBlck 6B D3 97 94 50 1 4 E0 1 4 AA BB CC DD

red: UID of transponder backwards. Ending with E0 (0 0 0 0 0 0 0 0 will write unaddressed)

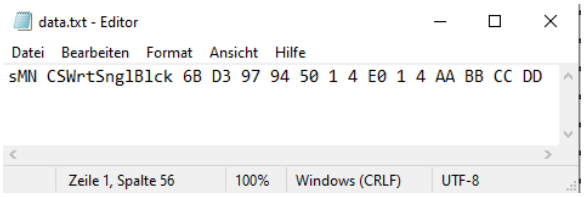
blue: number of block to write (Block count starts at 0)

green: Length of the data that shall be written (block size in Byte)

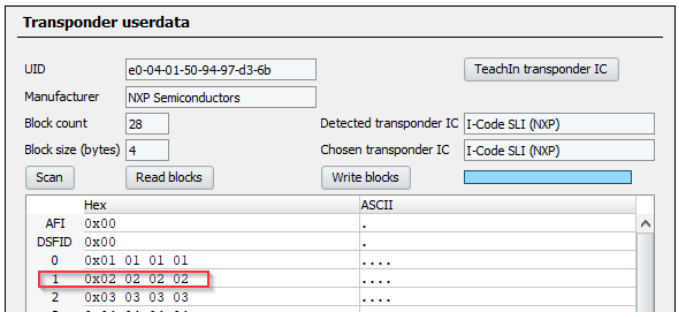
orange: data that shall be written

This command will write the data AA BB CC DD into the first block of the mentioned transponder.

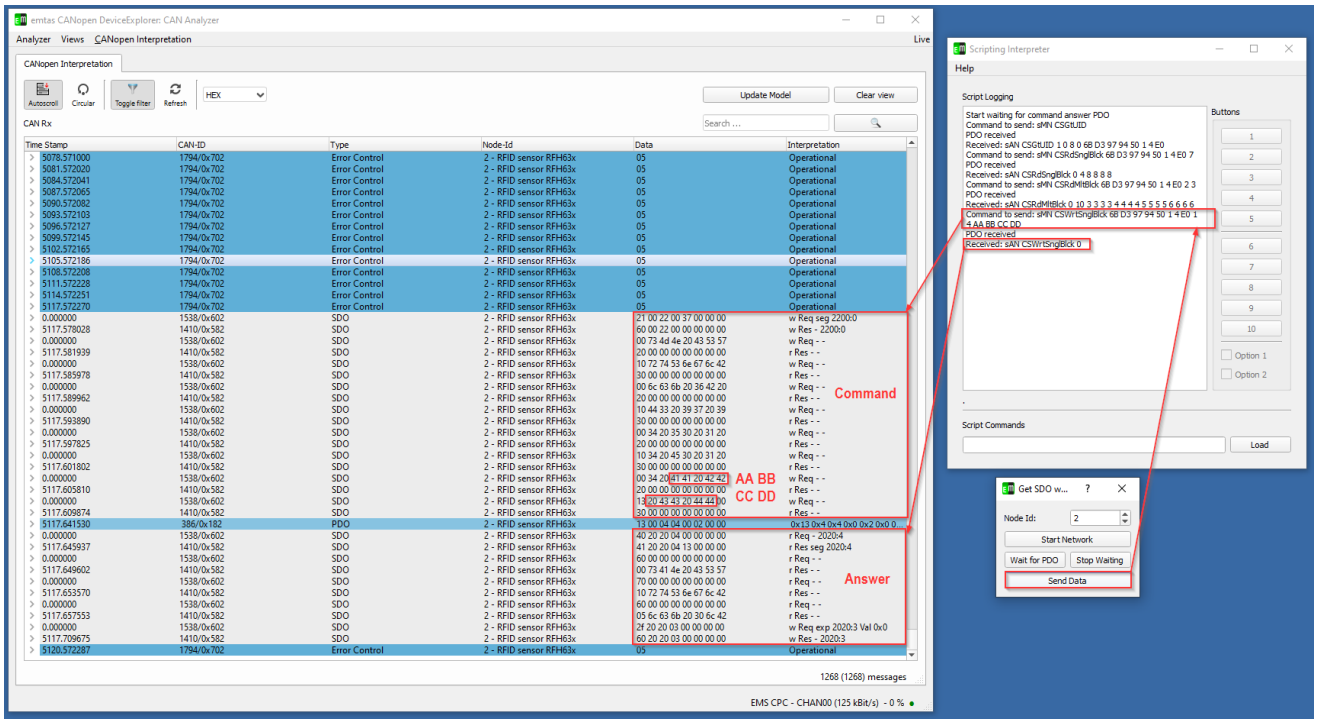
Write the command into the data.txt file at C:\temp and save the file.



According to SOPAS Transponder access the content that is at the first block before writing is 0x02 02 02 02.

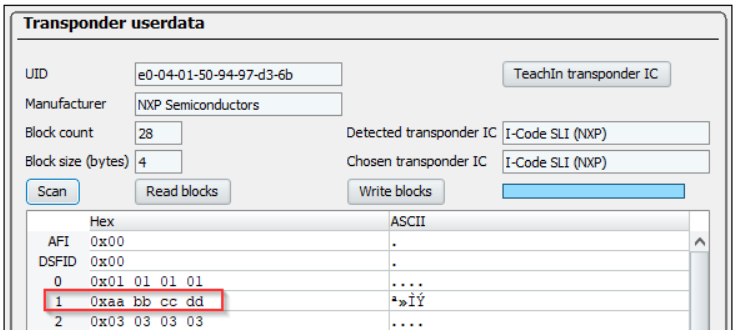


After sending the data we get the following answer.



The answer sAN … 0 indicates that the write process was successful.

Check at SOPAS Transponder access if the content has been written.



5.6.2. Write multiple blocks

sMN WrtMltBlck 6B D3 97 94 50 1 4 E0 0 2 +12 A A A A B B B B C C C C

red: UID of transponder backwards. Ending with E0 (0 0 0 0 0 0 0 0 will write unaddressed)

blue: number of first block to read, block count starts at block 0

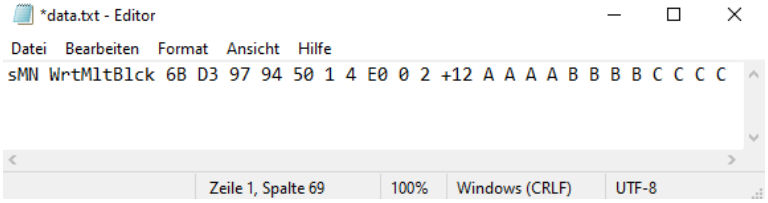
green: amount of blocks that shall be written. Enter number of blocks minus one! So 2 will read 3 blocks

purple: Length of the data that shall be written (block size in Byte * number of blocks). +12 is a DEC number for C in HEX. C could also be used here

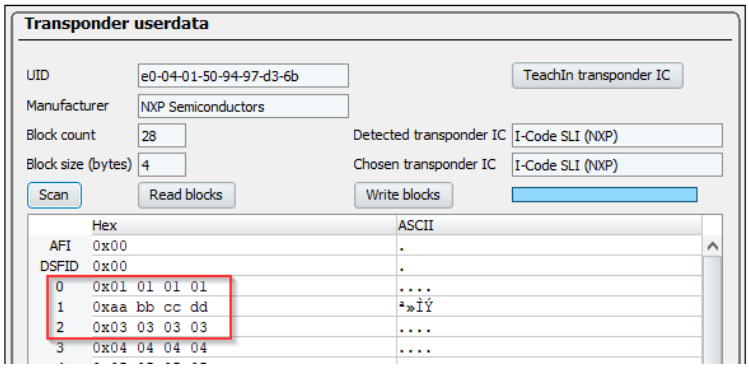
orange: data that shall be written

This command will write the data A A A A into block zero, B B B B into first block and C C C C into second block of the mentioned transponder.

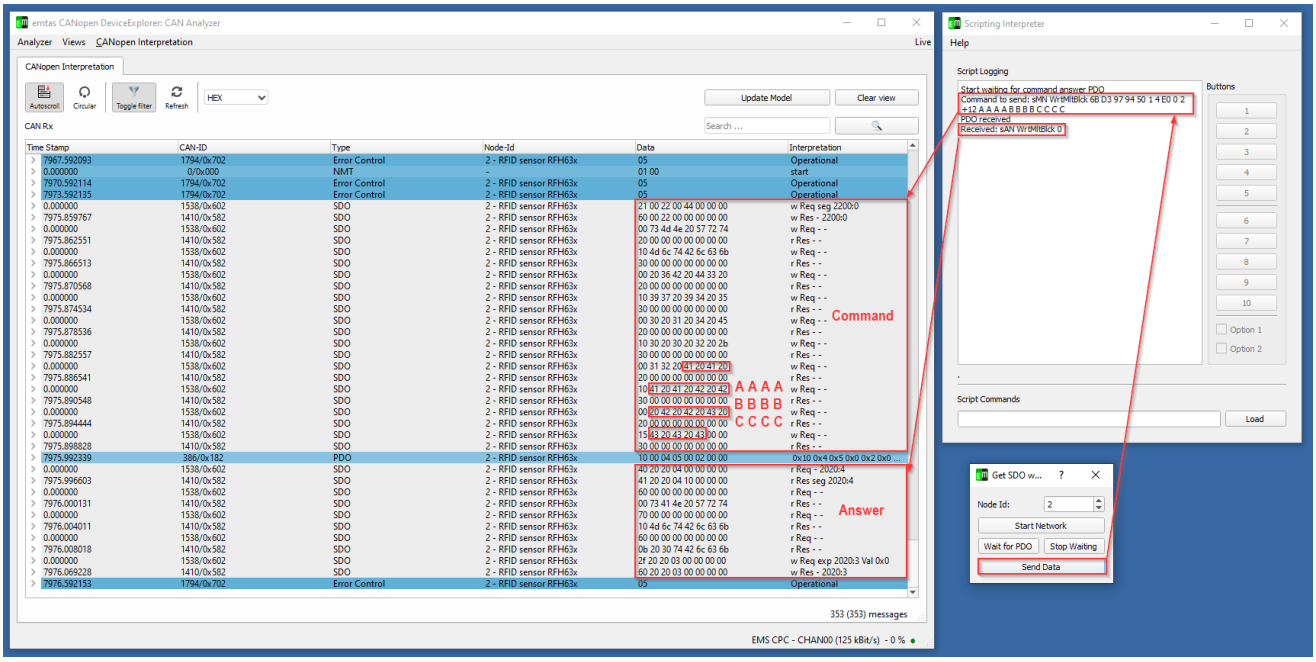
Write the command into the data.txt file at C:\temp and save the file.



According to SOPAS Transponder access the content in those blocks is the following:

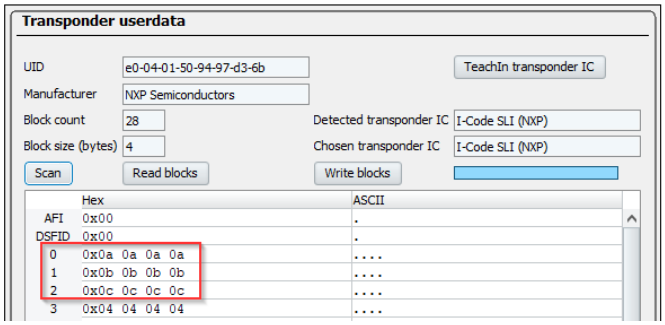


After sending the data we get the following answer.



The answer sAN … 0 indicates that the write process was successful.

Check at SOPAS Transponder access if the content has been written.



5.7. Change CAN ID via SDO

Use SDO communication as described in the previous examples and send following SOPAS commands to the device:

1. Login Command
   sMN SetAccessMode 4 81BE23AA

2. Command to change CAN ID
   sWN NWDevID **1**
   (red number stands for the CAN ID you want to use)

3. Command to save all parameters (to avoid loosing the configuration after reboot of the device)
   sMN mEEwriteall

4. Logout command
   sMN Run

Be aware that the device will be only accessible with the new ID from now on.