

**Citect for Windows**  
**Driver Specification**  
**TwinTCP**  
**for Beckhoff TwinCAT devices**

## Driver version history

Version	Date	Modified By	Details
1.0.0.0.A1	2016-02-25	Tomas Rook	Main development
1.0.0.0.B1	2018-04-22	Tomas Rook	String did not work (read/write)
1.0.0.1	2018-10-31	Tomas Rook	Upgraded to VS2010, no Protection (Andreas Mogren EI & Automation had problem with Citect license failure within a couple of days)
1.0.0.2	2018-11-14	Tomas Rook	Connect socket was allocated once again if connection failed, causing resources leakage
1.0.0.3	2019-05-04	Tomas Rook	Major updates, improved efficiency, added Unit-based parameters etc (transferred stuff from the ADS driver)
1.0.0.4	2019-08-26	Tomas Rook	Added protection project (lifted to VS2010)

## Driver documentation version history

Date	Modified By	Details
2016-02-25	Tomas Rook	Preliminary
2019-05-04	Tomas Rook	New parameters added

## Contents

<b>1. QA</b>	<b>5</b>
1.1 Introduction	5
1.2 Procedure for generating a new driver	5
<b>2. TARGET DEVICE(S) AND PROTOCOL</b>	<b>6</b>
2.1 Introduction	6
2.2 Device Manufacturer	6
2.3 Device Definition	6
2.4 Communications Methods	6
2.5 Communications Configuration	6
2.6 Contacts	6
<b>3. PROTOCOL REQUIREMENTS</b>	<b>7</b>
3.1 Introduction	7
3.2 TWINTCP.DBF	7
3.3 Supported variables/parameters	8
<b>4. USER INTERFACE</b>	<b>9</b>
4.1 Introduction	9
4.2 Driver Name	9
4.3 Boards Form	9
4.4 Ports Form	9
4.5 IO Devices Form	10
4.5.1 Address	11
4.5.2 Protocol	11
4.6 Pulldown lists Help	11
4.7 PROTDIR.DBF	11
4.8 Parameters and INI options	12
4.8.1 Standard Parameters	12
4.8.2 Driver Specific Parameters	12
4.9 Debug Messages	14
4.10 Stats Special Counters	14
4.11 Setup AMS and TCP/IP router communication	15
4.12 Network Address Translation (NAT) this driver	16
<b>5. BASIC TESTING</b>	<b>17</b>
5.1 Introduction	17

5.2 Procedure	17
<b>6. PERFORMANCE TESTING</b>	<b>19</b>
6.1 Introduction	19
6.2 Calculating the Blocking Constant – Not applicable	19
<b>7. REFERENCES</b>	<b>20</b>
7.1 References	20
<b>8. TCP COMMUNICATION FAIL DEBUGGING</b>	<b>21</b>
8.1 Communication problems	21
8.2 Using ADSConfig to check communication	21

## 1. QA

### 1.1 Introduction

This document follows the development of the new driver. It serves as a functional specification, design specification and test specification.

### 1.2 Procedure for generating a new driver

The following check list defines the QA steps for generating a new driver. This procedure must be followed for drivers to be integrated into Citect.

	Description	Person	Date
1	This specification document is written.	TR	2016-02-25
2	Specification reviewed and accepted by R&D department.		n/a
3	Driver coded.	TR	2016-02-25
4	Code and specification reviewed and accepted by R&D department.		n/a
5	Testing with connection project, and performance test.	TR	
6	Driver integrated into Citect source and built.		n/a
7	Documentation is written (HLP or MVB files)		n/a
	<b>At this checkpoint coding is done and the driver is available as a beta.</b>		
8a	Full testing is carried out.	TR	
8b	Performance testing is carried out.	TR	
8c	Specification and documentation updated from testing/performance tests	TR	
	<b>At this checkpoint the testing is complete.</b>		
9a	Review for completeness by developer, tester, documentor and R&D staff		n/a
9b	Add driver to install disks		n/a
9c	Add driver to protocols database		n/a
9d	Support notified of new driver for training purposes		n/a
10	Sales notified of new driver		n/a
	<b>The driver is now finished.</b>		

The hand over of a driver requires that all the above steps are completed and checked off.

## 2. Target Device(s) and Protocol

### 2.1 Introduction

This section defines the types of I/O Devices that are targeted by this driver.

### 2.2 Device Manufacturer

Beckhoff Automation Sverige AB  
Stenåldersgatan 2A 15  
SE - 213 76 Malmö  
Sweden

### 2.3 Device Definition

All Beckhoff PLC equipment capable of routed ADS communication (TwinCat2 & TwinCAT3 SoftPLC devices)

### 2.4 Communications Methods

The driver support the ADS protocol over TCP/IP.

The ADS protocol (*Automation Device Specification*) is a transport layer within the Beckhoff Twin-CAT system. It was developed for data exchange between the different software modules, for instance the communication between the NC and the PLC. The ADS protocol is used on top of TCP/IP. This means that in a networked system, all the data is accessible from any desired point.

### 2.5 Communications Configuration

The driver support only TCP. The driver connects directly to the *Message Router* using ADS packets wrapped in TCP/IP.

This method does not require any *TwinCAT* software whatsoever to be installed in the PC. The *Message Router* is built into the driver.

The driver is compatible with the Citect TWINCAT driver and support exactly the same memory/input/output variables.

### 2.6 Contacts

Driver Development	TroSoft AB, Tomas Rook	+46 (0)8 532 57262
Beckhoff Automation	Christer Wik	+46 (0)40 680 81 76

## 3. Protocol Requirements

### 3.1 Introduction

The driver is an asynchronous queued driver.

The driver was developed using DDK 5.5 and VS2010 sp1, and has been tested with Citect up to version 7.50.

### 3.2 TWINTCP.DBF

<i>Template</i>	<i>Unit type</i>	<i>Raw type</i>	<i>Bit width</i>	<i>Comment</i>
DEVICEVERSION	0x10000000	7	1024	TwinCAT I/O version
DEVICENAME	0x11000000	7	1024	Device name
IX%U(0,0,64511).%u(0,0,7)	0	0	8	InputBit
IB%U(0,0,64511).%u(0,0,7)	0	0	8	InputBit
IW%U(0,0,64510).%u(0,0,15)	0	0	8	InputBit
ID%U(0,0,64508).%u(0,0,31)	0	0	8	InputBit
IB%U(0,0,64511)	1	8	8	InputByte
IS%U(0,0,64511)	1	7	2040	InputString
IW%U(0,0,64510)%/2	2	1	16	InputWord
ID%U(0,0,64508)%/4	3	4	32	InputDoubleWord
IR%U(0,0,64508)%/4	3	2	32	InputReal
QX%U(0,0,64511).%u(0,0,7)	4	0	8	OutputBit
QB%U(0,0,64511).%u(0,0,7)	4	0	8	OutputBit
QW%U(0,0,64510).%u(0,0,15)	4	0	8	OutputBit
QD%U(0,0,64508).%u(0,0,31)	4	0	8	OutputBit
QB%U(0,0,64511)	5	8	8	OutputByte
QS%U(0,0,64511)	5	7	2040	OutputString
QW%U(0,0,64510)%/2	6	1	16	OutputWord
QD%U(0,0,64508)%/4	7	4	32	OutputDoubleWord
QR%U(0,0,64508)%/4	7	2	32	OutputReal
MX%U(0,0,64511).%u(0,0,7)	8	0	8	MemoryBit
MB%U(0,0,64511).%u(0,0,7)	8	0	8	MemoryBit
MW%U(0,0,64510).%u(0,0,15)	8	0	8	MemoryBit
MD%U(0,0,64508).%u(0,0,31)	8	0	8	MemoryBit
MB%U(0,0,64511)	9	8	8	MemoryByte
MS%U(0,0,64511)	9	7	2040	MemoryString
MW%U(0,0,64510)%/2	10	1	16	MemoryWord
MD%U(0,0,64508)%/4	11	4	32	MemoryDoubleWord
MR%U(0,0,64508)%/4	11	2	32	MemoryReal

### **3.3 Supported variables/parameters**

Memory, input and output of type digital, integer (16-bit), real (IEEE 32-bit), long (32-bit), byte (8-bit) and strings who are accessible using ADS.



## 4. User Interface

### 4.1 Introduction

This section defines how the user will see the driver. This relates directly to how the Citect forms need to be filled out and any special INI options. For the kernel, the debug trace messages and the *Stats.Special* counters are documented.

### 4.2 Driver Name

TWINTCP

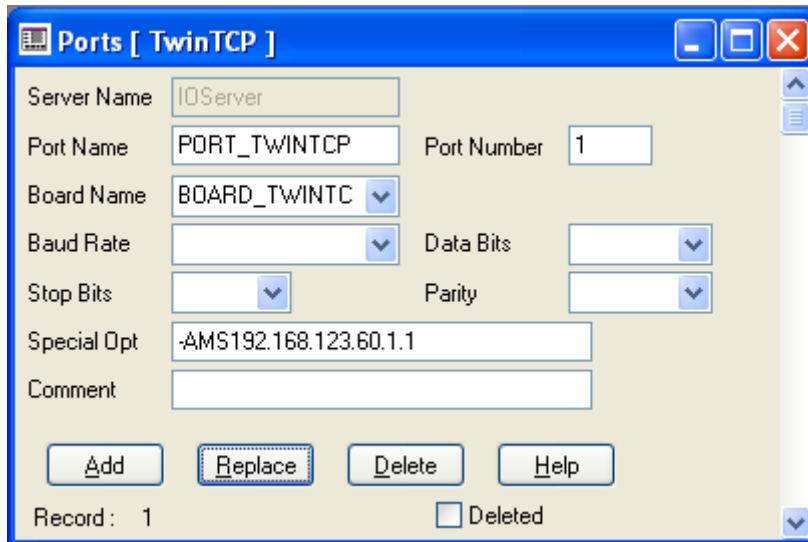
### 4.3 Boards Form

The screenshot shows a Windows-style dialog box titled "Boards [ TwinTCP ]". It contains several input fields and buttons. The "Server Name" field is set to "IOServer". The "Board Name" field is set to "BOARD\_TWINTCP". The "Board Type" field is a dropdown menu set to "TWINTCP". The "Address" field is a dropdown menu set to "0". The "I/O Port" and "Interrupt" fields are also dropdown menus. The "Special Opt" and "Comment" fields are text boxes. At the bottom, there are four buttons: "Add", "Replace", "Delete", and "Help". Below the buttons, it says "Record: 1" and there is a "Deleted" checkbox.

The driver is a board driver, *Board Type* should be set to *TWINTCP*

Enter *0* as address.

### 4.4 Ports Form



Here is where the AMS Net Id should be entered in the Special Opt field. The ID number has the following format,

*x.x.x.x.y.z* This number refers to the AMS Net Id specified in the TwinCAT System Properties. If the TwinCAT is on the local computer the *Local Computer AMS Net Id* should be used, for TwinCAT systems in a network the *AMS Net Id* specified in the *Remote Computers* should be used.

For easy to understand configuration in TwinCAT the first four digits *x.x.x.x* in the *AMS Net Id* can be the TCP/IP address of the local/remote computer, i.e. *192.168.123.230*. The two last digits *y.z* is information used in the *AMS Net Id* by the TwinCAT system.

These two digits have to be the same as configured in the *AMS Net Id* of the local/remote computer. If the *AMS Net Id* for the local/remote computer is set to *192.168.123.230.1.1*, the *Special Option* field has to be set according to the above example.

*x, y, z should be a decimal value between 0 – 255*

Example,

*-ams192.168.123.230.1.1*

*-AMS* could be lower or upper case.

See both *UseAMSAddressAsDefaultIP* later on in the *Parameters* section

## 4.5 IO Devices Form

#### 4.5.1 Address

In the each TwinCAT system you can have up to 4 PLC's, each PLC will get a consecutive *Port Number* as follows. *PLC1 -> 801, PLC2 -> 811, PLC3 -> 821 and PLC4 -> 831 (851..854 for a TwinCAT3 system)*. These unique *Port Numbers* will automatically be assigned to each PLC in the TwinCAT system. The address specified in the address field has to be one of these numbers as the driver uses the address when communicating to the TwinCAT system. This means that every local/remote TwinCAT system will have 1 to 4 PLC's which Citect can read.

Which PLC is read on the local/remote TwinCAT system by Citect is decided by the *IO Device Address*. Which computer the PLC is located on is decided by the *AMS Net Id (Special Opt in the Ports form)*.

The *Address* field could be any TwinCAT2 or TwinCAT3 address, ex *801, 821, 851* etc.

Example,

801

#### 4.5.2 Protocol

TWINTCP

### 4.6 Pulldown lists Help

The following entries should be included in the Citect Help.DBF spec file.

TYPE	DATA	FILTER
PROTOCOL	<b>TWINTCP</b>	

### 4.7 PROTDIR.DBF

The following entries should be included in the Citect Protdir.DBF spec file.

TAG	FILE	BIT_BLOCK	MAX_LENGTH	OPTIONS
TWINTCP	TWINTCP	256	1920	0xcf

## 4.8 Parameters and INI options

### 4.8.1 Standard Parameters

Block	256	
Delay	20	ms
MaxPending	30	
Polltime	0	ms
Timeout	10000	ms
Retry	0	(Sets the complete packet retries, should be 0)
WatchTime	30	s

### 4.8.2 Driver Specific Parameters

All TWINTCP specific parameters are located in the section 'TWINTCP'

Most parameters may also be defined per port specific (which usually means per PLC). The driver will default to the *Default* value below. Port specific values are set in section(s) named *[TWINTCP.<PortName>]* and always override any settings in the *[TWINTCP]* section.

Ex we like to have a transmit delay of 50 msec for port *P101\_C52026\_01P*

```
[TWINTCP. P101_C52026_01P]
Delay=50
```

This means port *P101\_C52026\_01P* will have a transmit delay of 50 msec and the other ports will have a transmit delay of 20 msec (as default).

Parameters that could be set port specific are marked in the PP-column below.

*OBS! In a redundant system the port names usually differs between the primary and standby server, ex for a standby server in the above example the section would be named ex [TWINTCP. P101\_C52026\_01S]*

PARAMETER	DEFAULT	UNIT	PP	DESCRIPTION
Delay	20	msec	✓	Each request cycle, the driver will scan through internal queues for work to do (create packets, update statistics, cleanup memory, return new values from buffers etc). Performance is linear to this time, but also CPU load. Basically this is the time the driver will pause between each packet to the PLC.
TimeOut	5000	msec	✓	Response timeout for the device. The timeout counter is reset each time a packet fragment is received, hence it will take a lot longer for the driver to timeout if the connection is very slow.
MaxPending	30			Number of DCBs the driver may handle simultaneously. Actually the default is quite low for this driver, but Citect is not capable of handling an unlimited number of outstanding DCBs. This number must be multiplied by number of channels to get the total DCBs for Citect.
WatchTime	30	sec		When Citect send DCB requests to the driver, it have to respond within this time. This is also the cycle time for init and status requests for offline devices. The driver will try to reply

				Citect DCB's within 3/4 of this time.
MaxReadsPerRequest	30		✓	<p>Number of read requests handled in one packet. Packets are bundled in a per-SoftPLC fashion, ex if address 801 and 802 exist on the same <i>AMS Net Id</i>, and Citect simultaneously request 5 values from each SoftPLC it will result in two requests with 5 variables in each.</p> <p>TwinCAT versions before v2.10 (build 1324) does not support multiple read requests. The driver automatically determine if multiple requests could be used. To force single reads either use the optional <i>-s</i> parameter in the <i>Ports form</i> of the affected SoftPLC's or set both <i>MaxReadsPerRequest</i> and <i>MaxWritesPerRequest</i> to 1 to affect all SoftPLCs.</p>
MaxWritesPerRequest	30		✓	<p>Number of write requests handled in one packet. Packets are bundled in a per-SoftPLC fashion, ex if address 801 and 802 exist on the same <i>AMS Net Id</i>, and Citect simultaneously request 5 values from each SoftPLC it will result in two requests with 5 variables in each.</p> <p>TwinCAT versions before v2.11 (build 1550) does not support multiple write requests. The driver automatically determine if multiple requests could be used. To force single reads either use the optional <i>-s</i> parameter in the <i>Ports form</i> of the affected SoftPLC's or set both <i>MaxReadsPerRequest</i> and <i>MaxWritesPerRequest</i> to 1 to affect all SoftPLCs.</p> <p><i>OBS! Not yet implemented. This parameter has no effect since the driver always issues single writes.</i></p>
IgnoreStartupErrors	0			<p>If set to 1 no error message will be displayed if the driver detects some problem during startup.</p> <p>If Citect is running as service it will just hang the IOserver process if an error message is presented during startup. The error box is not visible to the user, hence the <i>Ok</i> button could not be pressed.</p>
TraceToLogFile	0		✓	<p>If set to 1 all debug data that would appear in Kernel using the <i>debug &lt;portname&gt; all</i> command will be logged to file <i>TwinTCP.log</i> in the TwinTCP subfolder of Citect's data folder. The file could grow very big, but its size will be limited by <i>TraceFileLimit</i>.</p>
TraceFileLimit	100	MB's		<p>The file size limit of the <i>TwinTCP.log</i> file. This is the maximum file size in megabytes. If the file exceed this size, the logging will be stopped, however to continue logging just change the file name or delete it and logging will automatically resume (no restart is required)</p>
TraceByteLimit	64	bytes		<p>Limit the number of data bytes presented in logfile and Kernel debug (per packet presented hex data)</p>
TraceLevel	15		✓	<p><i>TraceLevel</i> sets the tracing to <i>Kernel</i> and <i>ADS.log</i></p> <ul style="list-style-type: none"> <li>1 Trace errors</li> <li>2 Trace Reads</li> <li>4 Trace Writes</li> <li>8 Trace Generic Info</li> <li>16 (not used)</li> <li>32 Trace Connection stuff</li> <li>64 (not used)</li> <li>128 Trace Communications <sup>1</sup></li> </ul>

				<p>256 Trace Citect requests (DCB's) <sup>1</sup></p> <p>512 (not used)</p> <p><sup>1</sup> will cause much data</p> <p>Values may be bit-weighted together, ex if <i>TraceLevel=6</i> will enable both <i>Read</i> and <i>Write</i> Traces</p> <p><i>TraceLevel=-1</i> will enable all traces</p> <p>The tracelevel may also be set using + to set multiple traces in a more convenient way, ex:</p> <p><i>TraceLevel=1+2+4+8</i></p>
LocalAmsAddress			✓	<p>The local <i>AMS Net Id</i> (and its port). Ex:</p> <p><i>192.168.123.80.1.1</i></p> <p><b>Warning!</b> If TwinCAT is installed on the Citect server, the <i>LocalAmsAddress</i> should not be the same as TwinCAT is using if it will be run when Citect is running, because PLC's won't allow two connections simultaneous from the same <i>AMS Net Id</i>. Use another <i>LocalAmsAddress</i> (ex ending with 2, ex <i>192.168.123.80.1.2</i>) and set up appropriate router lines in the PLC's</p>
AMS...=IP			✓	<p>Router table. Remote <i>AMS Net Id's</i> (as specified in the <i>Ports form Special Opt field</i>) and their physical IP..</p> <p>Ex:</p> <p><i>ams 192.168.123.230.1.1=192.168.123.230</i></p> <p><i>ams 192.168.123.231.1.1=192.168.123.231</i></p> <p><i>ams 192.168.123.232.1.1=192.168.123.232</i></p>
UseAMSAddress- sAsDefaultIP	0		✓	<p>If set to 1 the driver will automatically use the left part of the <i>Ports form Special Opt field</i> <i>AMS-netid</i> as physical IP if a corresponding router table translation is missing in <i>Citect.ini</i>, ex in <i>Ports form Special Opt field</i> <i>AMS netid</i> is - <i>ams 192.168.123.230.1.1</i> and it is missing in <i>Citect.ini</i> then the driver will use <i>192.168.123.230</i> as physical IP instead of indicating an error (<i>Missing remote Message Router IP in Citect.INI</i>).</p>
LocalNetworkAd- dress			✓	<p>Network address (network card) to be used (bind) if the server have more than one network card or IP-address. Generally the OS selects the most appropriate card for the local endpoint. Ex <i>192.168.123.80</i></p>

## 4.9 Debug Messages

Shows the outgoing/incoming data packets. Header in verbose format and data/value in hex.

## 4.10 Stats Special Counters

Number	Label	Purpose/Meaning of this counter
0	DCB Requests	Total number of driver requests

1	Phys read time	Physical read time in millisecs
2	Values per sec (R/W)	Number of read and written values per sec (based on a ten second sample period)
3	Requests per sec	Packet requests per sec (based on a ten second sample period)
4	Tx bytes	Transmitted bytes
5	Rx bytes	Received bytes
6	Tx packets	Transmitted packets
7	Rx packets	Received packets
8	Failed conn attempts	Succeeded TCP connection attempts
9	Failed conn attempts	Succeeded TCP connection attempts
10	Request timeouts	Timed out requests (will cause the connection to be reestablished)

## 4.11 Setup AMS and TCP/IP router communication

The *ADS Message Router* is built into the driver and need to be configured in the *Citect.ini* [TWINTCP] section as:

[twintcp]

localamsaddress=192.168.0.132.1.1

; May use DNS name(s) instead of IP

ams192.168.123.230.1.1=192.168.123.230

ams192.168.123.235.1.1=192.168.123.235

The driver has an own local *AMS Net Id* defined by parameter *LocalAMSAddress*. Use the Citect server's IP address followed by 1.1 as above to avoid confusion. The port defaults to 65534 (same as TwinCAT uses) but can be changed, ex:

LocalAMSAddress=192.168.0.132.1.1:65530

In the remote device's *TwinCAT Message Router* table you must also define the Citect driver's *LocalAMSAddress* (and its IP-address), even if communication takes place in same TCP connection the driver established. If you fail to do this, the device will not answer.

To define your own router table you enter the text 'AMS' followed by the *remote message router AMS Net Id's* and the IP-address. To avoid confusion try to use same *AMS Net Id* as the IP (as in the above examples). The IP-address can be exchanged to *DNS addresses*, ex

ams192.168.123.230.1.1=softplc1.trosoft.se

The port is by default 48898 but may be changed, ex:

ams192.168.123.230.1.1=softplc1.trosoft.se:10025

The *AMS Net Id's* must be used in the *Ports form* to associate the *IO Devices* with the correct IP-address. It would of course be more logical to place also the IP-address in the *Special Opt's* field in the *Ports form*, but it has not enough field length.

If the *AMS Net Id* usually has the device real IP-address as beginning you may set the *UseAMSAddressAsDefaultIP* parameter, then the driver will automatically parse the IP from the *AMS Net Id* if it is not found in the *[TwinTCP]* section.

See also the *AllowIPAddressInCommentField* parameter.

#### **4.12 Network Address Translation (NAT) this driver**

Communication to NAT networks behind firewalls is possible because communication use a TCP/IP connection (not UDP). Obviously the remote devices public IP-address must be static or registered dynamically using ex *DynDNS*.

The public IP of the Citect installation must however always be static because the *Remote Message router* table only allows IP-addresses.



## 5. Basic Testing

### 5.1 Introduction

The programmer will perform a minimum level of testing which is outlined here.

A sample Project is available which can be used as a starting point for the programmers test Project. When the programmer has completed basic testing and debugging this Project should be backed up and supplied to the Citect Testing department.

### 5.2 Procedure

The following are points should be covered by basic testing.

- On startup the IO Device comes online without errors.
  - Ok
- The driver supports IO Devices of addresses as documented in the specification.
  - Ok
- The driver reports the IO Device offline when the IO Device is a) powered down, b) disconnected.
  - Ok
- The driver will re-establish communication with the IO Device after a) power cycle, b) disconnection/ reconnection.
  - Ok
- Confirm that retries (if supported) and error reporting operate correctly.
  - Ok (Statistics report)
- The driver reads all the device data types documented as readable in this specification.
  - Ok
- The driver writes to all the device data types documented as write-able in this specification.
  - Ok
- The driver reads and writes all data formats supported by the protocol, ie DIGITAL, INT, LONG, REAL, STRING.
  - Ok
- Test the limit of the IO Devices request size, this should be done for at least DIGITAL and an INT data formats.
  - Not applicable to this driver.

- Let the driver run over night and check that no retries or other errors have occurred.
  - Ok
- If a multidrop or network protocol and if the hardware is available then the protocol should be tested with more than one IO Device connected.
  - Ok

## 6. Performance Testing

### 6.1 Introduction

Tests which give some indication of the drivers performance. The programmer needs to perform these tests since the results feed back into the Constants structure and the PROTDIR.DBF.

### 6.2 Calculating the Blocking Constant – Not applicable

Because this driver is a hybrid of the 'Front-End-Back-End' type, and due to the internal structure the blocking constant is 256 bits.

## 7. References

### 7.1 References

n/a.

## 8. TCP Communication fail debugging

### 8.1 Communication problems

If the unit will not come online, ensure following are ok (TCP mode):

- *LocalAMSAddress* is correct
- *TwinCAT System Manager* in the PLC is in Run mode
- *TwinCAT PLC Control* in the PLC is in Run mode
- The PLC has a program
- The device address (usually 801) is ok
- The AMS router entries in Citect.ini matches the Ports form
- The router table in the PLC contains a valid entry for the Citect server's IP and *LocalAMSAddress*
- The logged in user (or the user in the Citect service) has *Full control* in the Citect's *Data* folder
- Parameters are not misspelled

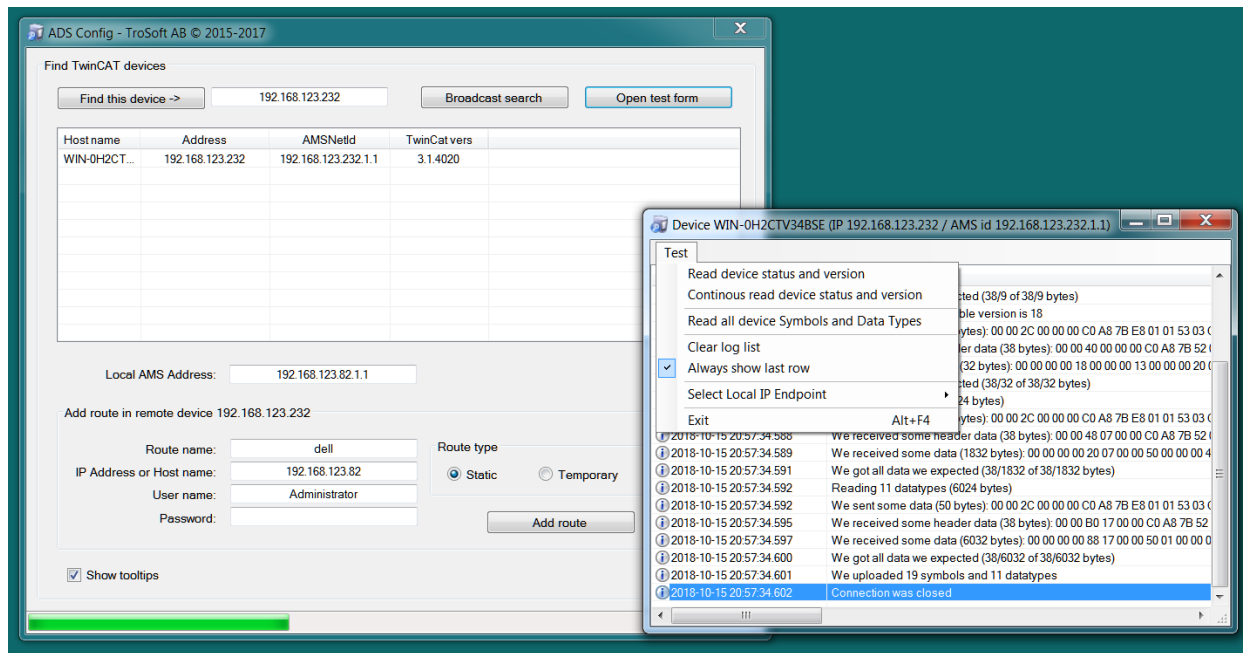
Use the *TraceToLogFile* and *TraceOptions* parameters to create an *TwinTCP.log* file.

If the driver cannot connect there is usually some problem with the router entry in the PLC and Citect.ini.

If the driver can connect there usually are some addressing issues, or the PLC is not online.

### 8.2 Using ADSConfig to check communication

There is a simple tool that may be used to check communication and create router entries remote in the PLC. It does not require TwinCAT to be installed in the server.



Sample of how the tool can be used to scan for devices on the local subnet using broadcast search and to test communication. In the above example the router line to the is ok and we get a proper response from the device.

This indicates that the Citect driver probably will work fine on this server.

In the *ADSCConfig.ini* file there are some setting to fine tune the behaviour, ex if using a TwinCAT 3 system the port should be changed to 851.