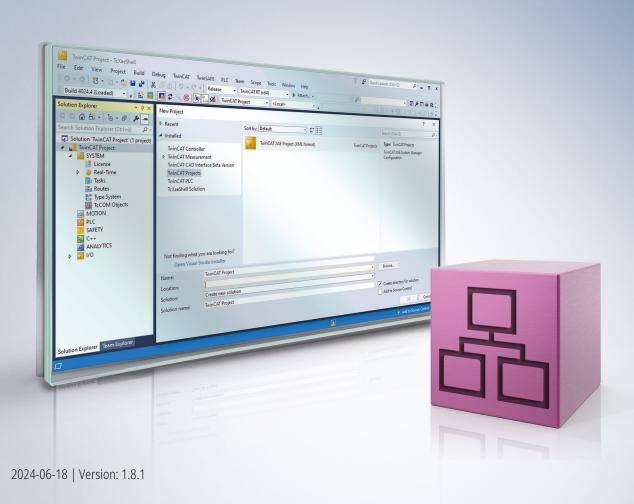
# **BECKHOFF** New Automation Technology

Manual | EN

# TF6311

TwinCAT 3 | TCP/UDP Realtime





# **Table of contents**

1	Fore	word		5
	1.1	Notes	on the documentation	5
	1.2	For you	ur safety	5
	1.3	Notes	on information security	7
2	Ove	rview		8
	2.1	Compa	arison TF6310 TF6311	8
	2.2	Restric	ctions	9
3	Insta	allation /	/ Licensing	10
			S	
•	4.1		Start (PLC / UDP)	
	4.2		Start (C++ / UDP)	
	4.3		Start (C++ / TCP Client)	
5			on	
J	5.1	_	e network cards	
	5.2		sk access to a network card	
_				
6	<b>Exar</b> 6.1	-	Simple TCP Client (PLC / C++)	
	0.1	6.1.1	S01: Simple TCP Client (C++)	
		6.1.2	S01: Simple TCP Client (C++)	
	6.2	_	JDP Client Server (PLC/C++)	
	0.2	6.2.1	S02: UDP Demo (PLC)	
		6.2.2	S02: UDP Demo (C++)	
		6.2.3	Test client	
	6.3		RP PING Demo (C++)	
	6.4		CP Echo Server (PLC / C++)	
		6.4.1	S04: TCP Server Demo (PLC)	
		6.4.2	S04: TCP Server Demo (C++)	49
7	Proc	ırammer	r's reference	51
	7.1		P: ITcloUdpProtocol(Recv)	
		7.1.1	Method ITcloUdpProtocolRecv:ReceiveData	
		7.1.2	Method ITcloUdpProtocol:SendData	53
		7.1.3	Method ITcloUdpProtocol:CheckReceived	54
		7.1.4	Method ITcloUdpProtocol:RegisterReceiver	54
		7.1.5	Method ITcloUdpProtocol:UnregisterReceiver	55
	7.2	TCP/U	IDP RT TcCom Parameter	55
	7.3	TCP/U	IDP RT TcCom diagnostics	57
	7.4	TCP/IF	P: ITcloTcpProtocol(Recv)	58
		7.4.1	Method ITcloTcpProtocolRecv:ReceiveData	59
		7.4.2	Method ITcIoTcpProtocolRecv:ReceiveEvent	
		7.4.3	Method ITcloTcpProtocol:AllocSocket	
		7.4.4	Method ITcloTcpProtocol:FreeSocket	
		7.4.5	Method ITcloTcpProtocol:Connect	
		7.4.6	Method ITcloTcpProtocol:IsConnected	61

Version: 1.8.1



	0.4	ADC Dot	urn Codos	70
9	Appe	ndix		70
	8.1	Start-up:	Ip Stack ADS 1823 / 0x71f	69
8	Fault	analysis		69
	7.6	Return v	alues	68
		7.5.7	Method ITcloArpPingProtocol:CheckReceived	68
		7.5.6	Method ITcloArpPingProtocol:UnregisterReceiver	67
		7.5.5	Method ITcloArpPingProtocol:RegisterReceiver	67
		7.5.4	Method ITcloArpPingProtocol:ArpRequest	66
		7.5.3	Method ITcIoArpPingProtocol:PingRequest	66
		7.5.2	Method ITcIoArpPingProtocolRecv:ArpReply	65
		7.5.1	Method ITcIoArpPingProtocolRecv:PingReply	65
	7.5	ARP/Pin	g: ITcloArpPingProtocol(Recv)	64
		7.4.13	Method ITcloTcpProtocol:GetFreeSendDataSize	
		7.4.12	Method ITcloTcpProtocol:GetRemotelpAddr	63
		7.4.11	Method ITcloTcpProtocol:CheckReceived	63
		7.4.10	Method ITcloTcpProtocol:SendData	63
		7.4.9	Method ITcloTcpProtocol:Accept	62
		7.4.8	Method ITcloTcpProtocol:Listen	62
		7.4.7	Method ITcloTcpProtocol:Close	62



### 1 Foreword

#### 1.1 Notes on the documentation

This description is intended exclusively for trained specialists in control and automation technology who are familiar with the applicable national standards.

For installation and commissioning of the components, it is absolutely necessary to observe the documentation and the following notes and explanations.

The qualified personnel is obliged to always use the currently valid documentation.

The responsible staff must ensure that the application or use of the products described satisfies all requirements for safety, including all the relevant laws, regulations, guidelines, and standards.

#### **Disclaimer**

The documentation has been prepared with care. The products described are, however, constantly under development.

We reserve the right to revise and change the documentation at any time and without notice.

No claims to modify products that have already been supplied may be made on the basis of the data, diagrams, and descriptions in this documentation.

#### **Trademarks**

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered and licensed trademarks of Beckhoff Automation GmbH.

If third parties make use of designations or trademarks used in this publication for their own purposes, this could infringe upon the rights of the owners of the said designations.

#### **Patents**

The EtherCAT Technology is covered, including but not limited to the following patent applications and patents:

EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702 and similar applications and registrations in several other countries.



EtherCAT® is registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany

#### Copyright

© Beckhoff Automation GmbH & Co. KG, Germany.

The distribution and reproduction of this document as well as the use and communication of its contents without express authorization are prohibited.

Offenders will be held liable for the payment of damages. All rights reserved in the event that a patent, utility model, or design are registered.

### 1.2 For your safety

#### Safety regulations

Read the following explanations for your safety.

Always observe and follow product-specific safety instructions, which you may find at the appropriate places in this document.



#### **Exclusion of liability**

All the components are supplied in particular hardware and software configurations which are appropriate for the application. Modifications to hardware or software configurations other than those described in the documentation are not permitted, and nullify the liability of Beckhoff Automation GmbH & Co. KG.

#### Personnel qualification

This description is only intended for trained specialists in control, automation, and drive technology who are familiar with the applicable national standards.

#### Signal words

The signal words used in the documentation are classified below. In order to prevent injury and damage to persons and property, read and follow the safety and warning notices.

#### Personal injury warnings

#### **▲ DANGER**

Hazard with high risk of death or serious injury.

#### **⚠ WARNING**

Hazard with medium risk of death or serious injury.

#### **A CAUTION**

There is a low-risk hazard that could result in medium or minor injury.

#### Warning of damage to property or environment

#### NOTICE

The environment, equipment, or data may be damaged.

#### Information on handling the product



This information includes, for example: recommendations for action, assistance or further information on the product.



### 1.3 Notes on information security

The products of Beckhoff Automation GmbH & Co. KG (Beckhoff), insofar as they can be accessed online, are equipped with security functions that support the secure operation of plants, systems, machines and networks. Despite the security functions, the creation, implementation and constant updating of a holistic security concept for the operation are necessary to protect the respective plant, system, machine and networks against cyber threats. The products sold by Beckhoff are only part of the overall security concept. The customer is responsible for preventing unauthorized access by third parties to its equipment, systems, machines and networks. The latter should be connected to the corporate network or the Internet only if appropriate protective measures have been set up.

In addition, the recommendations from Beckhoff regarding appropriate protective measures should be observed. Further information regarding information security and industrial security can be found in our https://www.beckhoff.com/secquide.

Beckhoff products and solutions undergo continuous further development. This also applies to security functions. In light of this continuous further development, Beckhoff expressly recommends that the products are kept up to date at all times and that updates are installed for the products once they have been made available. Using outdated or unsupported product versions can increase the risk of cyber threats.

To stay informed about information security for Beckhoff products, subscribe to the RSS feed at <a href="https://www.beckhoff.com/secinfo">https://www.beckhoff.com/secinfo</a>.



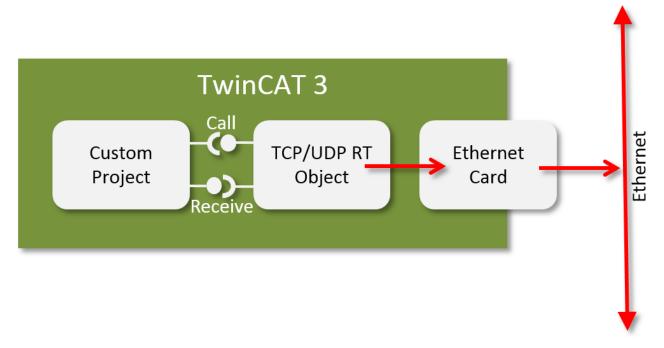
### 2 Overview

The "TCP/UDP Realtime" Function (TF6311) offers direct access to network cards from the real-time environment. Access can be either from the PLC (61131-3) or C++.

The following protocols are supported:

- TCP/IP
- UDP/IP
- · ARP / Ping

This section describes the concept of <u>interfaces as API [▶ 51]</u>. An introduction is provided by means of <u>sample programs [▶ 38]</u>.



Regardless of the protocol, the communication between the project using the protocol and TwinCAT is realized with a pair of interfaces:

- · An Interface Pointer provides support for sending data and establishing connections etc.
- The implementation of a receiver interface provides feedback for the project in the form of events or data, based on callbacks.

The communication partner of these interface pairs is a "TCP/UDP RT" TcCom object, which is instantiated and configured with the network card.

- Depending on the protocol, the Quickstarts [ 11] provide a good introduction.
- The configuration process is documented under Configuration [ 33].
- The interfaces are described in the <u>Programmer's reference</u> [▶ <u>51</u>] and illustrated through <u>samples</u> [▶ <u>38</u>].

### **2.1 Comparison TF6310 TF6311**

The products TF6310 "TCP/IP" and TF6311 "TCP/UDP Realtime" offer similar functionality.

This page provides an overview of similarities and differences of the products:



	TF 6310	TF 6311
TwinCAT	TwinCAT 2 / 3	TwinCAT 3
Client/Server	Both	Both
Large / unknown networks	++	+
Determinism	+	++
High-volume data transfer	++	+
Programming languages	PLC	PLC and C++
Operating system	Win32/64, CE5/6/7	Win32/64, CE7
UDP-Mutlicast	Yes	No
Trial license	Yes	Yes
Protocols	TCP, UDP	TCP, UDP, Arp/Ping
Hardware requirements	Variable	TwinCAT-compatible network card
Socket configuration	See operating system (WinSock)	TCP/UDP RT TcCom Parameter
		[ <u>&gt; 55]</u>

The Windows firewall cannot be used, since the TF6311 is directly integrated in the TwinCAT system. In larger / unknown networks we recommend using the TF6310.

### 2.2 Restrictions

The following limitations exist for the product:

- No local communication in real-time or between real-time and Windows operating system. (Alternative: communication via a second network interface.)
- · Multicast is not supported.
- The EL6601 and EL6614 cannot be used for TF6311 TCP/UDP real-time.
- If breakpoints are used, we strongly advise to use different network interfaces, since a breakpoint stops parts of the TwinCAT systems, which may be relevant for the communication with Engineering.



## 3 Installation / Licensing

The Function TF6311 requires no separate installation; all software components are available once TwinCAT 3 has been installed.

- A "TC3 TCP UDP RT" license is required.
   The dependence is entered by adding the "TCP/UDP RT" object to the project as a license. It can also be specified manually.
- A trial license can be created and used.



### 4 Quick Starts

This section contains detailed step-by-step instructions for some protocols.

They illustrate the use of the product in a simple manner. The samples are intended to facilitate understanding; they do not provide comprehensive implementation instructions. At the application level, the handling must be programmed in detail (e.g. the behavior on arrival of corresponding TCP events).

The function TF6311 "TCP/UDP real-time" has extensive capabilities:

- · different protocols (TCP, UDP, ARP/Ping)
- · different programming languages (PLC / C++) and
- · communication directions (client / server)

Step-by-step instructions are not available for all combinations. Once the basic <u>concept [ 8]</u> has been understood, further implementations can be derived in conjunction with the existing step-by-step instructions and samples [ 838].

### 4.1 Quick Start (PLC / UDP)

The sample implements an "echo service": A UDP server is started on a port (default: 10000). If this server receives a UDP packet, it returns the content to the sender (with same IP and same port).

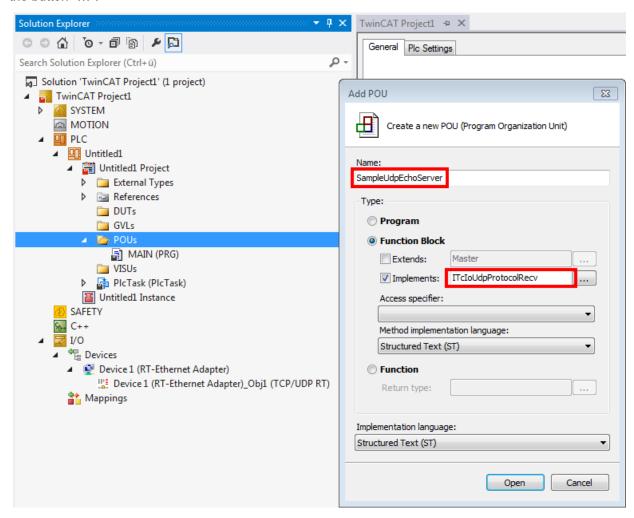
The sample is also available for download under <u>Sample 02 [ 40]</u>. In addition to the Quick Start, the download contains extended code, which does not affect the basic functionality.

#### Implementation of the UDP echo server in a PLC project

- ✓ A TwinCAT solution was generated
- 1. If no PLC project exists in the TwinCAT solution, you have to create one.
- A function block is generated, which implements the interface "ItcloUdpProtocolRecv". This creates a
  method, which is called when UDP packets arrive.
   By right-clicking on the node "POU" in the PLC project you can allocate names in the Popup window,



activate "SampleUdpEchoServer" and "Implements" by ticking, and select the interface mentioned with the button "...":



The declaration part of the function block contains several variables in the declaration:

- · Oid: Configurable reference to the TCP/UDP RT module
- · ipUdp: Interface pointer to the UdpProtocol, which is implemented by the TCP/UDP RT module
- · udpPort: Port used for receiving
- 3. The declaration part is created in this way:

```
{attribute 'c++ compatible'}
FUNCTION BLOCK SampleUdpEchoServer IMPLEMENTS ITCIOUdpProtocolRecv
VAR INPUT
END_VAR
VAR OUTPUT
END VAR
VAR
    {attribute 'TcInitSymbol'}
    oid:
                   OTCID;
    ipUdp:
                      ITcIoUdpProtocol;
                      UINT := 10000;
    nUdpPort:
    nReceivedPakets: UINT;
    hrInit :
                      HRESULT;
    hrSend :
                      HRESULT;
END VAR
```

The CheckReceive() method of the TCP/UDP RT module must be called in the body of the function block.

4. The body is created in this way:

```
IF ipUdp <> 0 THEN
    ipUdp.CheckReceived();
END IF
```

The method "ReceiveData", which was created through implementation of the interface, will be called repeatedly via "CheckReceived": one call for each packet received in the meantime.

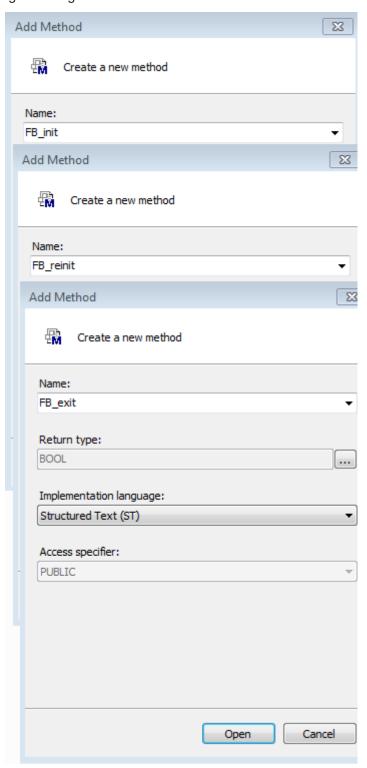


5. The method has both sender information and data as input parameters. In this sample, the "SendData" method returns an incoming packet as response (with sender/receiver reversed). The implementation is done as follows:

```
nReceivedPakets := nReceivedPakets+1;
IF ipUdp <> 0 THEN
    hrSend := ipUdp.SendData(ipAddr, udpSrcPort, udpDestPort, nData, pData, TRUE, 0); // send
data back
END_IF
```

During start and finish, a reference to the "UdpProtocol" interface must be set from the configured OID; corresponding approvals should be taken care of during shutdown.

6. The function block requires the methods "FB\_init", "FB\_reinit" and "FB\_exit", which can be created by right-clicking on the function block "Add..." method:





Appropriate signatures are generated automatically, so that only the actual body has to be realized. Of particular significance is the "RegisterReceiver" call, which opens a UDP port for reception.

7. The "FB\_init" method requires two local variables:

```
VAR
ipSrv: ITComObjectServer;
END VAR
```

8. The "FB\_init" method is implemented as follows:

```
IF NOT bInCopyCode THEN // no online change
IF ipUdp = 0 AND oid <> 0 THEN
hrInit := FW ObjMgr GetObjectInstance(oid:=oid, iid:=TC GLOBAL IID LIST.IID ITcIoUdpProtocol,
pipUnk:=ADR(ipUdp));
IF SUCCEEDED (hrInit) THEN
IF SUCCEEDED(ipUdp.RegisterReceiver(nUdpPort, THIS^)) THEN //open port
FB init := TRUE;
ELSE
FB init := FALSE;
FW_SafeRelease(ADR(ipUdp));
END_IF
END IF
ELSIF oid = 0 THEN
FB_init := FALSE;
hrInit := ERR_INVALID_PARAM;
END_IF
END IF
```

In the "FB\_reinit" method, which is executed during an OnlineChange, the TCP/UDP RT object must be supplied with the new address for the callbacks.

9. The "FB\_reinit" method is implemented as follows:

```
IF (ipUdp <> 0) THEN
ipUdp.RegisterReceiver(nUpdPort, THIS^);
FB_reinit := TRUE:
END_IF
```

The port must be closed again during shutdown (but not during OnlineChange, cf. blnCopyCode).

10. The "FB\_exit" method is implemented as follows:

```
IF (NOT bInCopyCode AND ipUdp <> 0) THEN //Shutdown
    ipUdp.UnregisterReceiver(nUpdPort);
    FW_SafeRelease(ADR(ipUdp));
    FB_exit := TRUE;
ELSE
    FB_exit := FALSE;
END_IF
```

11. Finally, the function block must be called:

```
PROGRAM MAIN

VAR

udp1 : SampleUdpEchoServer;

END_VAR

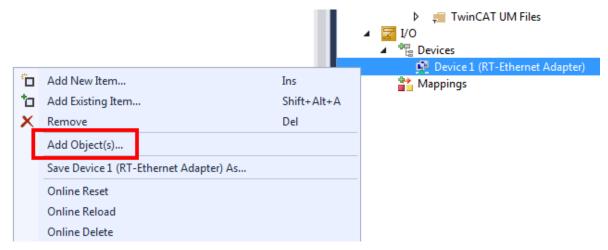
udp1();
```

"TCP/UDP RT" module configuration

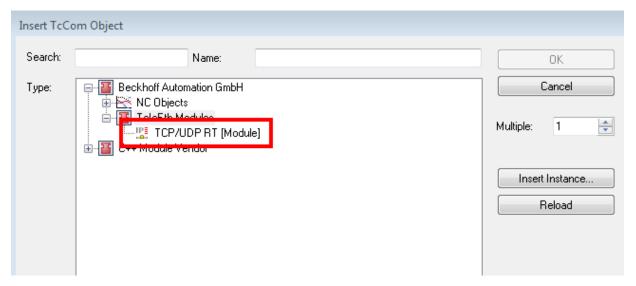
Notice Variable names relating to TCP are used here. They have to be substituted accordingly.



1. Create the "TCP/UDP RT" module under the RT Ethernet adapter by selecting "Add Object(s)..." in the context menu.



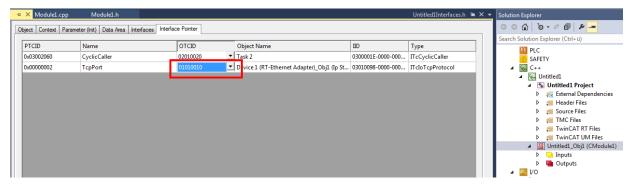
2. Then select the "TCP/UDP RT" module:



⇒ The TCP/UDP RT object is created under the adapter.

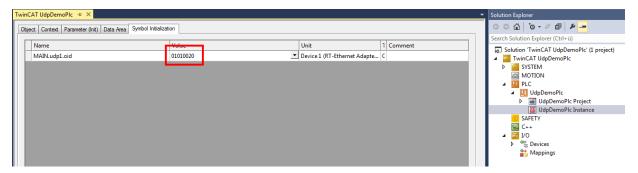


3. Parameterize the previously created instance of the module (here: Module1) under "Interface Pointer" "TcpProt" with the OID of the created "TCP/UDP RT" object:





4. For PLC projects this configuration is also done at the instance, under the tab "Symbol Initialization":



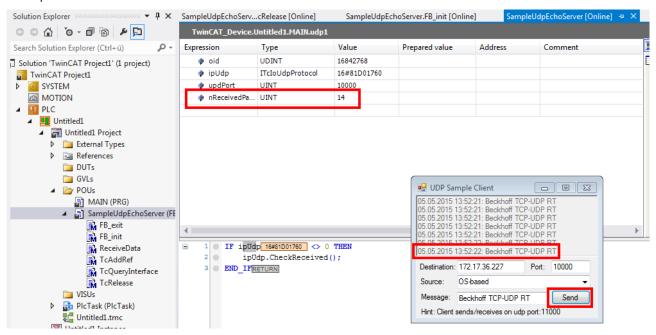
⇒ The configuration is thus completed

#### Disconnection by the operating system in Promiscuous mode

If Promiscuous mode is active at the RT Ethernet adapter in the "Adapter" tab, any TCP connection attempts are blocked by the operating system, since it does not recognize a port opened in the TCP/UDP RT object.

#### **Testing**

Once the configuration has been activated, a UDP packet can be sent to the PLC via the <u>UDP Sample Client</u> [<u>▶ 44</u>]. It can be observed that each call increments the counter. The client displays the returned packets at the top.



#### No local communication

The UDP sample client must run on a different computer than the PLC with the TCP/UDP RT object, because no local communication between the Windows operating system and the real-time is available.

Alternatively, a so-called "loop cable" can be used to connect two network ports. The UDP sample client can be forced to use a specific port by selecting the source (dropdown menu "Source").

### 4.2 Quick Start (C++ / UDP)

The example implements an "echo service": A UDP server is started on a port (default: 10000). If this server receives a UDP packet, it returns the content to the sender (with same IP and same port).

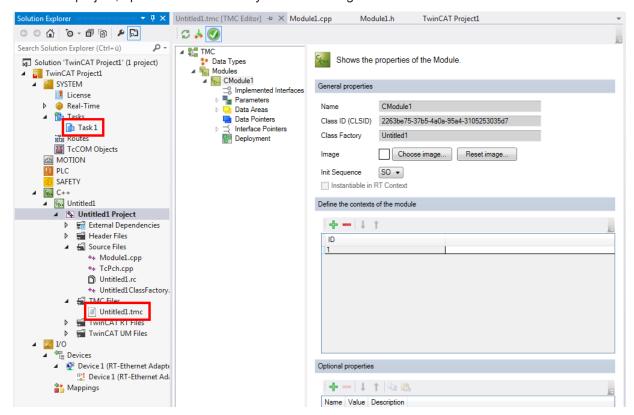
The engineering system must meet the requirements for TwinCAT 3 C++.



The example is also available for download under Sample 02 [ 40].

#### Implementation of the UDP echo server in a C++ project

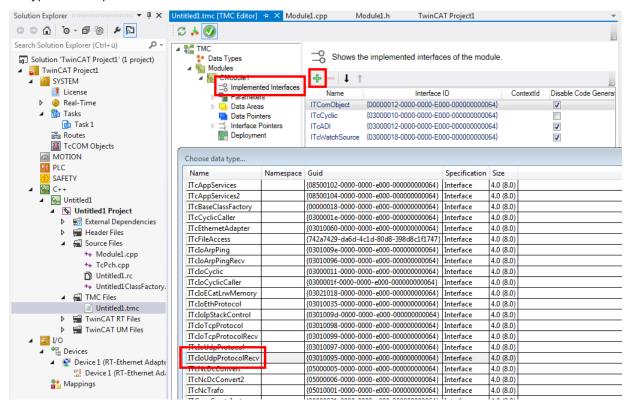
- ✓ A TwinCAT solution was generated
- 1. If no C++ project exists in the TwinCAT solution, you have to create one. Please use the template for "TwinCAT Module Class with Cyclic IO".
- 2. Create a task. Under System / Tasks right-click and select "Add new Item..." A normal task (without image) is sufficient.
- 3. In the C++ project, open the TMC editor by double-clicking on the TMC file.



The module must implement the *ITcloUdpProtocolRecv*. This creates a method, which is called when UDP packets arrive.

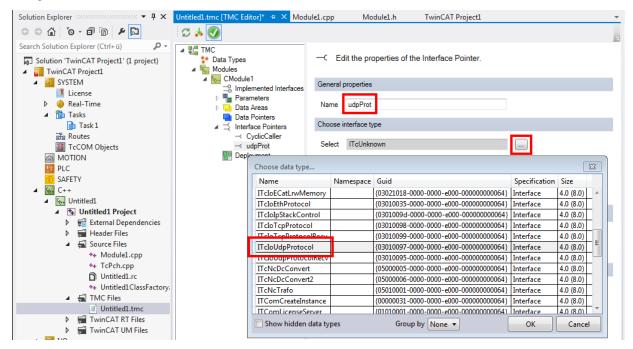


4. In the TMC editor select "Implemented interfaces" and create them with "+". A dialog appears, in which the type *ITcloUdpProtocolRecv* is selected:



The module requires an interface pointer to *ITcloUdpProtocol*, which contains the reference to the TCP/UDP RT object.

5. In the TMC editor select "Interface Pointer" and press "+". An interface is created, which can be opened by double-clicking. Assign a name "UdpProt" and set the pointer type with "..." and the selection in the dialog:



The TMC code generator is started once. Right-click on the C++ project and select "TMC Code Generator" in the context menu.

The CheckReceived() method of the TCP/UDP RT module must be called in the CycleUpdate() method in the CPP file of the module (Module1.cpp). As a result, arriving UDP packets are transferred to the implemented method ReceiveData() via callback.



7. The CycleUpdate() method is implemented as follows

```
///<AutoGeneratedContent id="ImplementationOf_ITcCyclic">
HRESULT CModule1::CycleUpdate(ITcTask* ipTask, ITcUnknown* ipCaller, ULONG_PTR context)
{
HRESULT hr = S_OK;
    m_counter+=m_Inputs.Value;
    m_Outputs.Value=m_counter;
    m_spUdpProt->CheckReceived(); // ADDED
    return hr;
}
```

The method "ReceiveData", which was created through implementation of the interface, will be called repeatedly via CheckReceived(): one call for each packet received in the meantime.

8. The method ReceiveData has both sender information and data as input parameters. In this sample, the SendData method returns an incoming packet as response (with sender/receiver reversed). The implementation is done as follows:

During start and finish, a reference to the "UdpProtocol" interface must be set from the configured OID; corresponding approvals should be taken care of during shutdown.

9. The start is triggered in the transition from SafeOp to Op. During this process, RegisterReceiver is of particular interest: It opens a UDP port for reception.

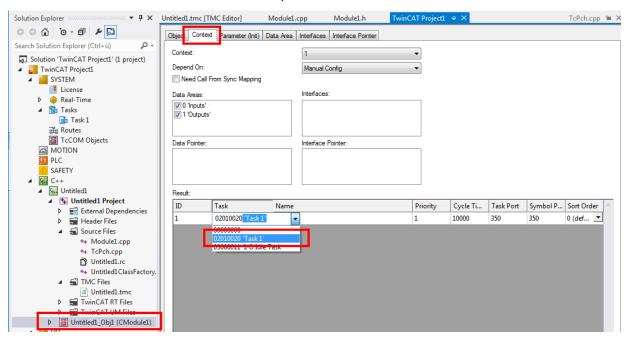
```
HRESULT CModule1::SetObjStateSO()
   HRESULT hr = S OK;
    //START EDITING
   if (SUCCEEDED(hr) && m_spUdpProt.HasOID())
       m Trace.Log(tlInfo, FLEAVEA "Register UdpProt");
       if (SUCCEEDED DBG(hr = m spSrv->TcQuerySmartObjectInterface(m spUdpProt)))
            m Trace.Log(tlInfo, FLEAVEA "Server: UdpProt listen to Port: %d", 10000);
            if (FAILED(hr = m spUdpProt->RegisterReceiver(10000,
THIS CAST(ITcIoUdpProtocolRecv))))
                m Trace.Log(tlError, FLEAVEA "Server: UdpProtRegisterReceiver failed on Port:
%d", 10000);
               m spUdpProt = NULL;
    // If following call is successful the CycleUpdate method will be
called,
    // eventually even before method has been left.
   hr = FAILED(hr) ? hr : AddModuleToCaller();
   // Cleanup if transition failed at some stage
   if (FAILED(hr))
        if (m spUdpProt != NULL)
           m spUdpProt->UnregisterReceiver(10000);
       m spUdpProt = NULL;
       RemoveModuleFromCaller();
   m Trace.Log(tlVerbose, FLEAVEA "hr=0x%08x", hr);
    return hr;
```

10. The stop takes place in the Op to SafeOp transition. The UDP port is closed again:



Finally, the module has to be instantiated and configured

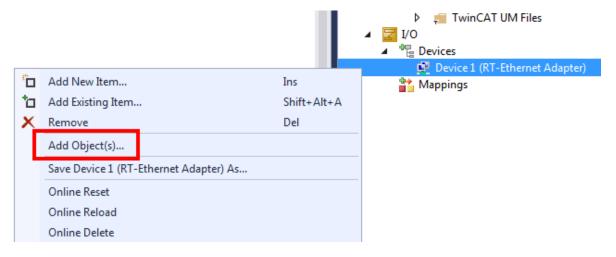
- 11. Build the project once. Right-click on the module select "Build"
- 12. Creating an instance of the module. Right-click on the project to open "Add new item...". Select the appropriate module here.
- 13. Double-click on the module instance to enable parameterization. First select the task in the "Context" tab.



#### "TCP/UDP RT" module configuration

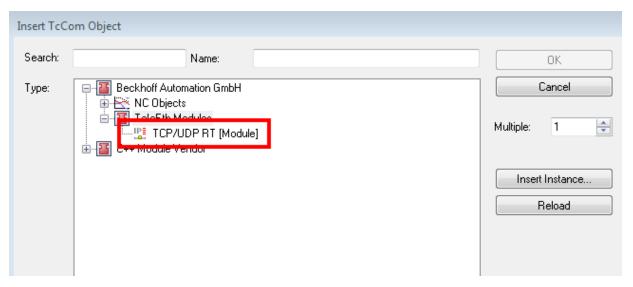
Notice Variable names relating to TCP are used here. They have to be substituted accordingly.

1. Create the "TCP/UDP RT" module under the RT Ethernet adapter by selecting "Add Object(s)..." in the context menu.





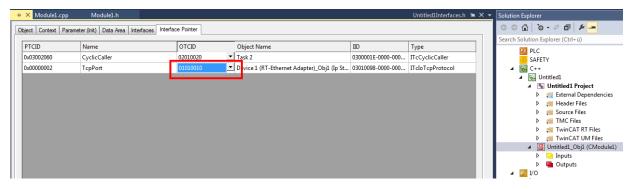
2. Then select the "TCP/UDP RT" module:



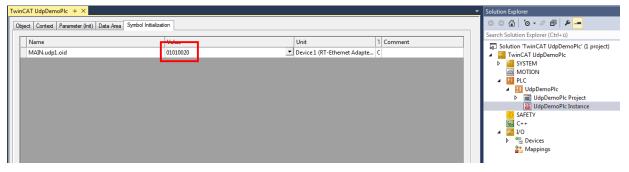
⇒ The TCP/UDP RT object is created under the adapter.



3. Parameterize the previously created instance of the module (here: Module1) under "Interface Pointer" "TcpProt" with the OID of the created "TCP/UDP RT" object:



4. For PLC projects this configuration is also done at the instance, under the tab "Symbol Initialization":



⇒ The configuration is thus completed

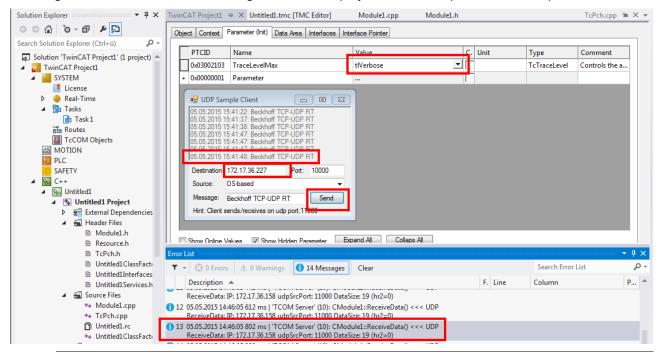
### Disconnection by the operating system in Promiscuous mode

If Promiscuous mode is active at the RT Ethernet adapter in the "Adapter" tab, any TCP connection attempts are blocked by the operating system, since it does not recognize a port opened in the TCP/UDP RT object.



#### **Testing**

Once the configuration has been enabled, a UDP packet can be sent to the C++ module via the <u>UDP Sample Client [\* 44]</u>. By activating the corresponding TraceLevel (here at least tllnfo; see C++ Tracing), an output can be generated in the Visual Studio log. The client displays the returned packets at the top.



#### No local communication



The UDP sample client must run on a different computer than the PLC with the TCP/UDP RT object, because no local communication between the Windows operating system and the real-time is available.

Alternatively, a so-called "loop cable" can be used to connect two network ports. The UDP sample client can be forced to use a specific port by selecting the source (dropdown menu "Source").

### 4.3 Quick Start (C++ / TCP Client)

This Quick Start shows the implementation of a TCP client as a TwinCAT 3 C++ project.

The engineering system must meet the requirements for TwinCAT 3 C++.

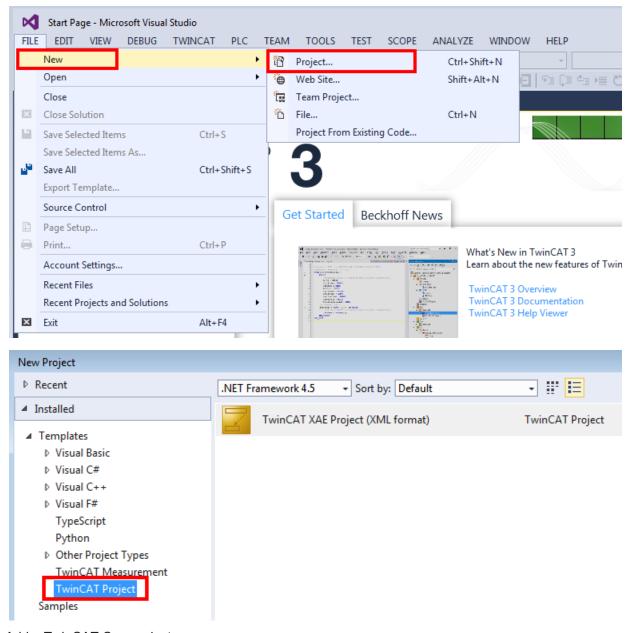
The example is also available for download under <u>Sample 01 [\* 38]</u>.

#### Creating a TwinCAT C++ project

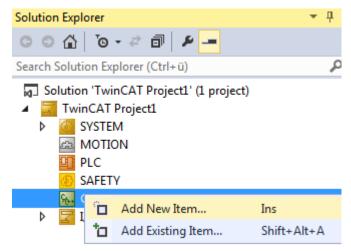
In this step, a new TwinCAT 3 C++ project is created.



1. Create a new TwinCAT project

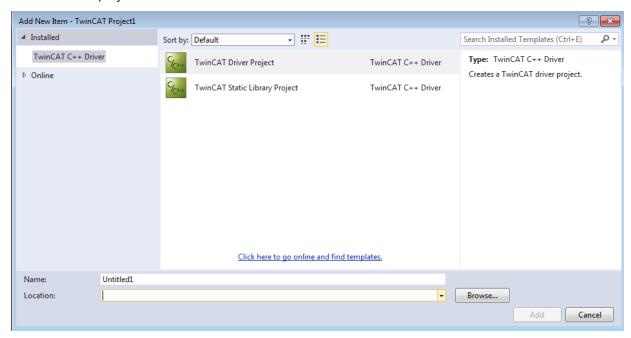


2. Add a TwinCAT C++ project

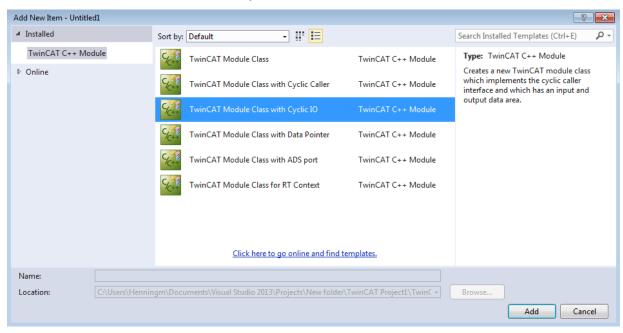




3. Select a Driver project

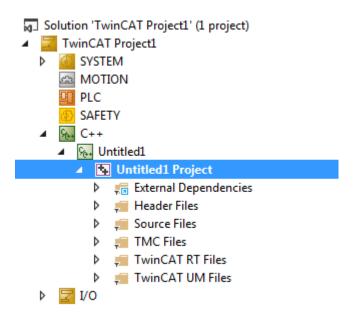


4. Use the wizard for a module class with "Cyclic IO" as the basis for the TCP client.





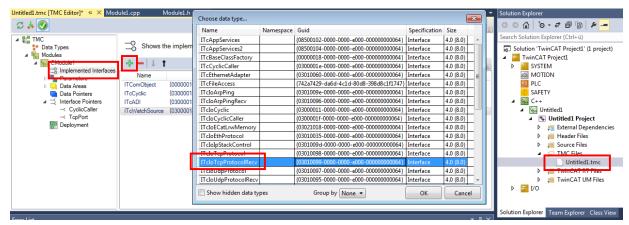
⇒ The result is a complete TwinCAT C++ project.



#### TMC editor for creating interfaces, pointers and parameters

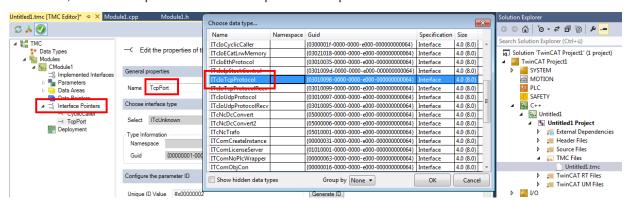
After creating the project, the next step involves implementation of the C++ TCP client.

 The module created by the wizard must implement the interface "ITcloTcpProtocolRecv". Open the TMC editor by double-clicking on the TMC file for the project. Add the interface to the module under "Implemented Interfaces".



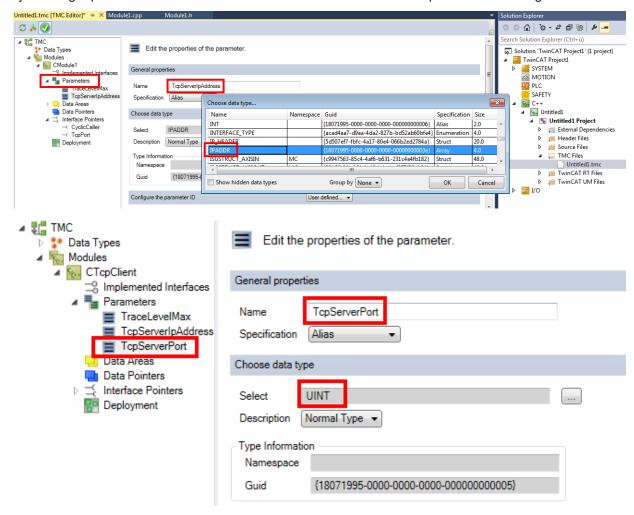
Under "Implemented Interfaces" open a selection of the available interfaces by clicking on the "+" button. Select "ITcloTcpProtocolRecv".

2. In addition, an "ITcIOTcpProtocol" interface pointer is required.

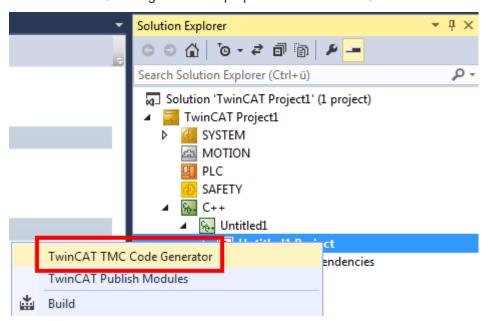




3. By creating a parameter the server IP address to be contacted and the port become configurable.



4. Now use the TMC code generator to prepare the code of the C++ module.



Start the TMC code generator by selecting the appropriate menu item in the context menu (right-click) of the C++ project.

⇒ All steps in the TMC editor are now completed.



#### Implement TCP client

Create two member variables in the module header file (here: Modul1.h).

```
ULONG m_SockId;
BOOL m_bSendRequest; //set by debugger for sending a http command
ULONG m_connections; //count number of connection attempts
HRESULT m hrSend; //Last hr of SendData
```

2. These are initialized in the Constructor (Module1.cpp).

3. The interface pointer m\_spTcpProt is now initialized in the Transition SO (i.e. in method SetObjStateSO).

4. In the Transition OS (i.e. method SetObjStateOS) a connection that may exist is closed, and the socket is released.

```
// State transition from OP to SAFEOP
HRESULT CTcpClient::SetObjStateOS()
   //start added code
   m Trace.Log(tlVerbose, FENTERA);
   HRESULT hr = S_OK;
   if ( m SockId != 0 )
      if (m spTcpProt->IsConnected(m SockId) == S OK)
          m spTcpProt->Close(m SockId);
          m_spTcpProt->CheckReceived();
    m spTcpProt->FreeSocket(m SockId);
    m SockId = 0;
   RemoveModuleFromCaller();
   m Trace.Log(tlVerbose, FLEAVEA "hr=0x%08x", hr);
   return hr;
   //end added code
```

5. The actual process is implemented in the "CycleUpdate" method, which is called cyclically. Establishes a TCP connection to a server (address is provided in parameters "m\_TcpServerlpAddress" and "m\_TcpServerPort"). The connection handle is stored in the member variable "m\_SockId". The connection is used to issue a simple http GET request.



6. The module implements the interface "ITcloTcpProtocolRecv", as a result of which the TMC code generator created a "ReceiveEvent" method. This is called when an event is received and must therefore be able to deal with a wide range of event types.

```
HRESULT CTcpClient::ReceiveEvent(ULONG socketId, TCPIP EVENT tcpEvent)
//start added code
m Trace.Log(tlInfo, FLEAVEA "Receive TCP Event: SocketId: %d Event: %d \n", socketId, tcpEvent);
    switch (tcpEvent)
   case TCPIP EVENT ERROR:
   case TCPIP_EVENT_RESET:
   case TCPIP EVENT TIMEOUT:
   m Trace.Log(tlInfo, FLEAVEA "Connection to remote server failed!\n");
          m SockId = 0;
       break;
   case TCPIP EVENT CONN CLOSED:
       m Trace.Log(tlInfo, FLEAVEA "Close connection: SocketId: %d \n", socketId);
         m SockId = 0;
       break;
   case TCPIP EVENT CONN INCOMING:
   case TCPIP_EVENT_KEEP_ALIVE:
   case TCPIP EVENT CONN IDLE:
   case TCPIP EVENT DATA SENT:
   case TCPIP_EVENT_DATA_RECEIVED:
       break;
   default:
       break;
   return S OK;
    //end added code
```

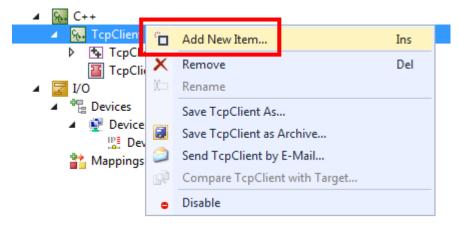
7. Analogous to the "ReceiveEvent" method, a "ReceiveData" method was created from the "ITcloTcpProtocolRecv" interface. It is responsible for receiving the data and is implemented as follows:

```
HRESULT CTcpClient::ReceiveData(ULONG socketId, ULONG nData, PVOID pData)
{
//start added code
    HRESULT hr = S_OK;
    PCHAR pResponse = new CHAR[100];
    memset(pResponse, 0, 100);
    memcpy(pResponse, pData, min(100, nData));
    m_Trace.Log(tlInfo, FLEAVEA "Receive answer w/ length %d : first 100 chars:'%s'", nData,
pResponse);
    return hr;
//end added code
}
```

8. The module is now ready and can be compiled. (Right-click on "Build" project).



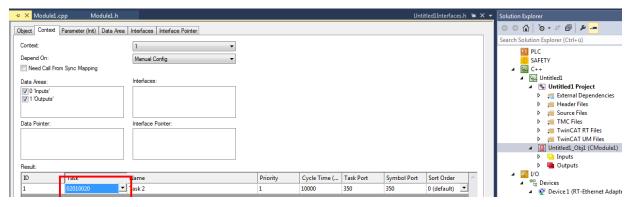
9. An instance of the module is created: Right-click on the C++ project



and select the module



⇒ The instance is associated with a task, so that the "CycleUpdate" method is called.



#### Preparing the network card

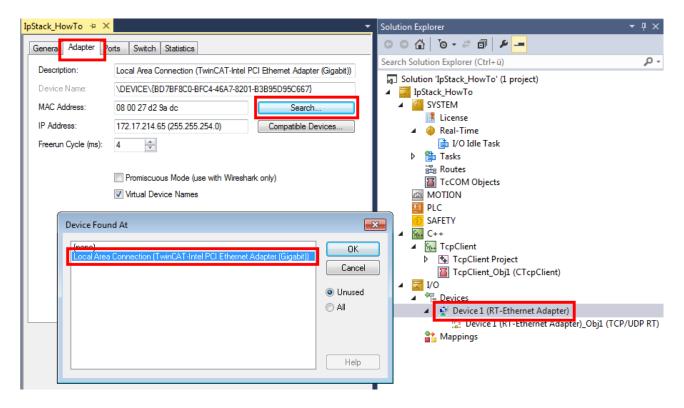
For the TCP/UDP RT module, make sure that the RT Ethernet adapter in the TwinCAT solution is connected with the correct network card (with TwinCAT driver).

### Local configuration only

ĺ

Installation of the driver on compatible network cards via the button "Compatible Devices" always takes place locally. On a controller with TwinCAT XAR, the program TcRteInstall.exe can be used. It is included in the installation (usually under C:TwinCAT\3.1\System).

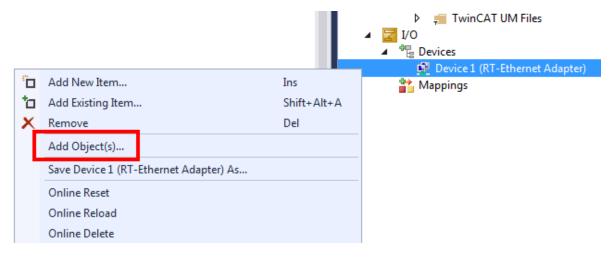




#### "TCP/UDP RT" module configuration

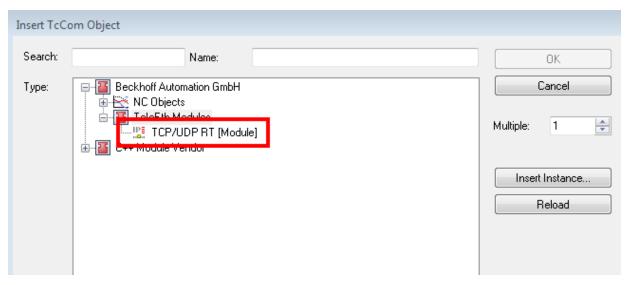
Notice Variable names relating to TCP are used here. They have to be substituted accordingly.

1. Create the "TCP/UDP RT" module under the RT Ethernet adapter by selecting "Add Object(s)..." in the context menu.





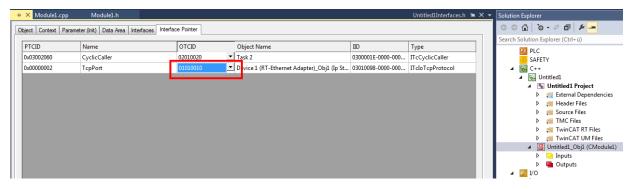
2. Then select the "TCP/UDP RT" module:



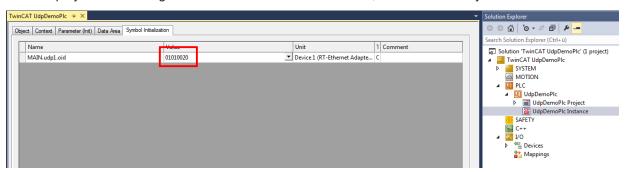
⇒ The TCP/UDP RT object is created under the adapter.



3. Parameterize the previously created instance of the module (here: Module1) under "Interface Pointer" "TcpProt" with the OID of the created "TCP/UDP RT" object:



4. For PLC projects this configuration is also done at the instance, under the tab "Symbol Initialization":



⇒ The configuration is thus completed

### Disconnection by the operating system in Promiscuous mode

If Promiscuous mode is active at the RT Ethernet adapter in the "Adapter" tab, any TCP connection attempts are blocked by the operating system, since it does not recognize a port opened in the TCP/UDP RT object.



#### Handling

1. The sample is ready to use once you have configured both the TcpServerlpAddress and the TcpServerPort at the module instance:



**Notice** Possible source of error: A test web server 62.159.14.51 is queried in the sample. A corresponding HTTP command is stored in the source code. IP address, port, and this HTTP command may have to be adjusted.

2. After activating the configuration you can see log messages (see source code) and the first 100 bytes of the response from the server in the output:

```
MSG | 1/22/2015 3:14:14 PM 671 ms | 'TCOM Server' (10): CTcpClient::ReceiveEvent() <<< Receive TCP Event: SocketId: 1 Event: 10

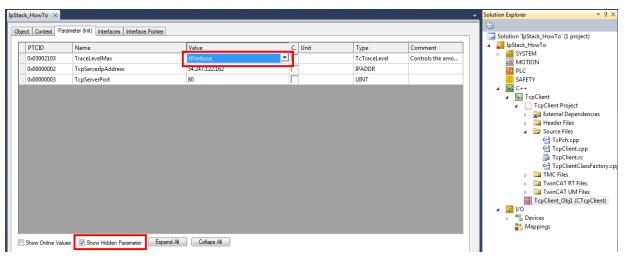
MSG | 1/22/2015 3:14:14 PM 671 ms | 'TCOM Server' (10): CTcpClient::ReceiveEvent() <<< Receive TCP Event: SocketId: 1 Event: 8

MSG | 1/22/2015 3:14:14 PM 671 ms | 'TCOM Server' (10): CTcpClient::ReceiveEvent() <<< Receive TCP Event: SocketId: 1 Event: 10

MSG | 1/22/2015 3:14:14 PM 671 ms | 'TCOM Server' (10): CTcpClient::ReceiveData() <<< Receive answer w/ length 5642 : 'HTTP/1.1 200 OK Cache-Control: private Content-Length: 5366
Content-Length: 5366
Content-Type: text/html
Server: Micr: So' |

MSG | 1/22/2015 3:14:15 PM 681 ms | 'TCOM Server' (10): CTcpClient::ReceiveEvent() <<< Receive TCP Event: SocketId: 1 Event: 7
```

3. To output these messages the "Tracelevel" can be configured (via Info):



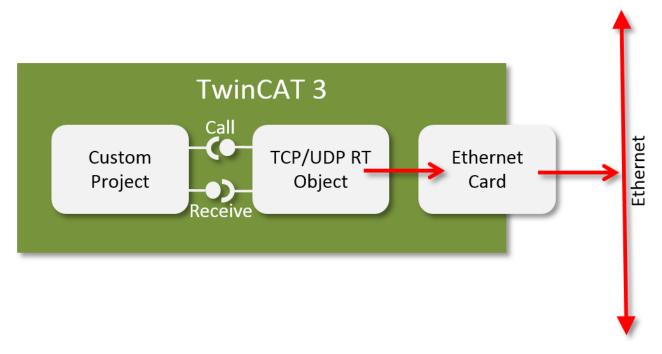
The procedure is carried out once when the program starts.

A new request is sent if "m\_bSendRequest" is set to TRUE (e.g. through TwinCAT Live Watch). The return of the SendData method is stored in hrSend. For the sample it can be monitored via the debugger.



## 5 Configuration

The integration and configuration of the "TCP/UDP RT" object is described here, starting from an existing TwinCAT project.



The "TCP/UDP RT" object is instantiated and configured. The configuration essentially consists of assigning the network card to be used.



#### **Windows Firewall**



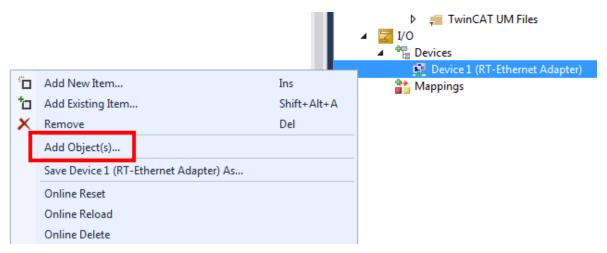
The Windows firewall cannot be used, since the TF6311 is directly integrated in the TwinCAT system.

The "TCP/UDP RT" object also contributes some parameters, which are documented <a href="here">here</a> [> 55].

#### "TCP/UDP RT" module configuration

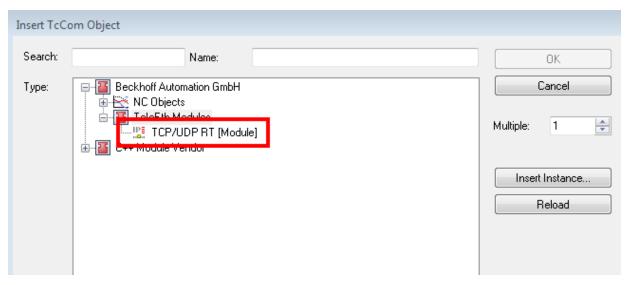
Notice Variable names relating to TCP are used here. They have to be substituted accordingly.

1. Create the "TCP/UDP RT" module under the RT Ethernet adapter by selecting "Add Object(s)..." in the context menu.

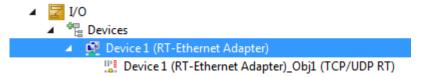




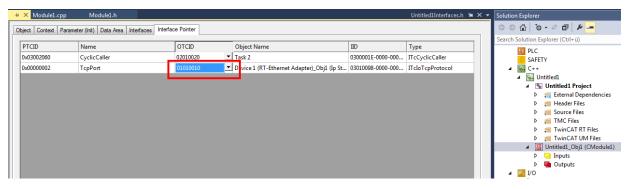
2. Then select the "TCP/UDP RT" module:



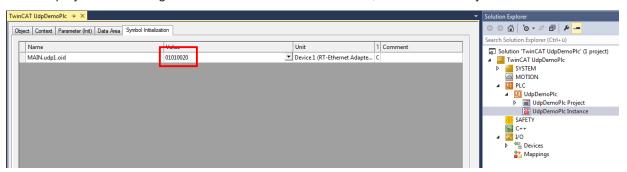
⇒ The TCP/UDP RT object is created under the adapter.



3. Parameterize the previously created instance of the module (here: Module1) under "Interface Pointer" "TcpProt" with the OID of the created "TCP/UDP RT" object:



4. For PLC projects this configuration is also done at the instance, under the tab "Symbol Initialization":



⇒ The configuration is thus completed

### Disconnection by the operating system in Promiscuous mode

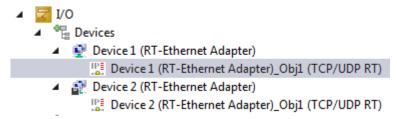
If Promiscuous mode is active at the RT Ethernet adapter in the "Adapter" tab, any TCP connection attempts are blocked by the operating system, since it does not recognize a port opened in the TCP/UDP RT object.



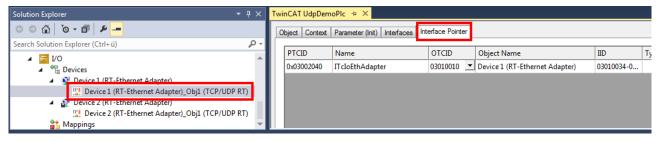
### 5.1 Multiple network cards

A TCP/UDP RT object is assigned to an RT Ethernet adapter by instantiating it under the objects, for example. A TCP/UDP RT object therefore always addresses precisely one network port of the controller via the RT Ethernet adapter.

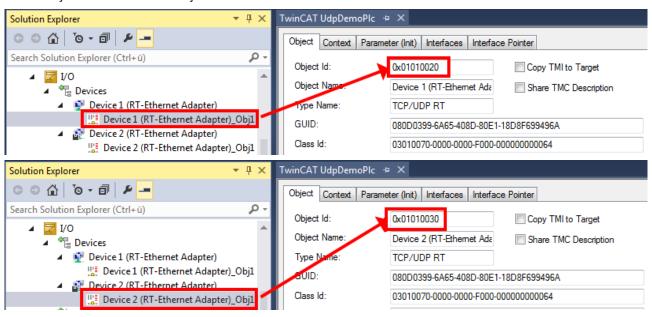
If several network ports are to be used, a TCP/UDP RT object is created for each RT Ethernet adapter:



The TCP/UDP RT objects relate to the higher-level RT Ethernet adapter, if no other configuration was specified manually:

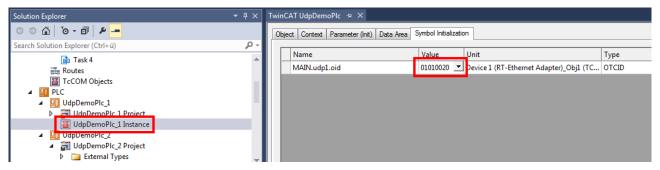


These objects have different object IDs:



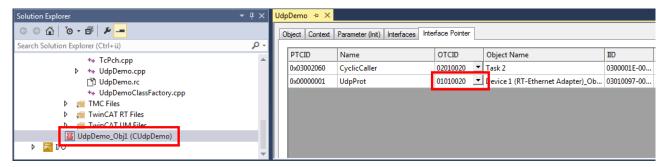
This object ID is used for referencing, as described above:

#### PLC:



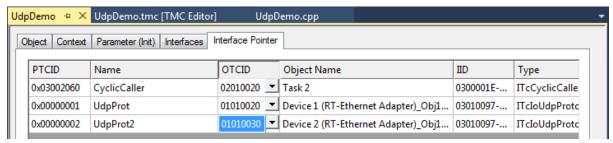


#### Or for a C++ module:



The use is highly dependent on the application. Some sample scenarios are provided below:

- A C++ module can be instantiated more than once. Each module can then communicate via a
  particular network card, based on the configuration with the corresponding object ID.
- Different PLC programs can be assigned separate TCP/UDP RT objects and thus act independently.
- A PLC or C++ program can address several TCP/UDP RT objects (and therefore several network cards), based on corresponding symbols (C++ is used as an example here):

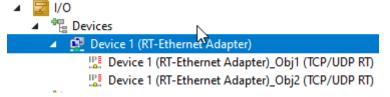


Object management must be implemented to suit the application. For example, the CheckReceived() calls must be applied to all objects. This also applies to calls for SendData() / RegisterReceiver() etc.

### 5.2 Multitask access to a network card

If a network card is to be used from several real-time contexts (tasks), it must be implemented as described here.

 A TCP/UDP RT object must be created for each real-time context (e.g. task) from which data is to be received or sent.



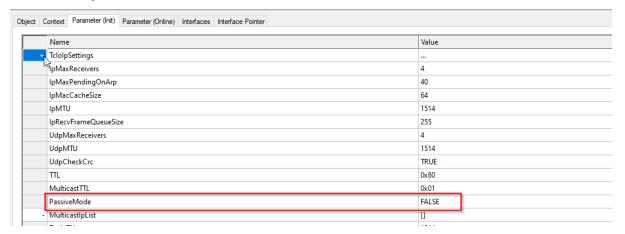
 The PassiveMode parameter on all TCP/UDP RT objects specifies whether or not these objects should fetch frames received from the RT Ethernet adapter. By default, PassiveMode is set to FALSE so that packets are fetched.

For multitask access, only one TCP/UDP RT object should fetch the data and all other objects should be configured with PassiveMode to TRUE.

Typically, this can be the object that receives packets in the fastest cycle. Where appropriate, a lower



priority can be used for this in order to make the real-time processes of other tasks more independent of the incoming frames.



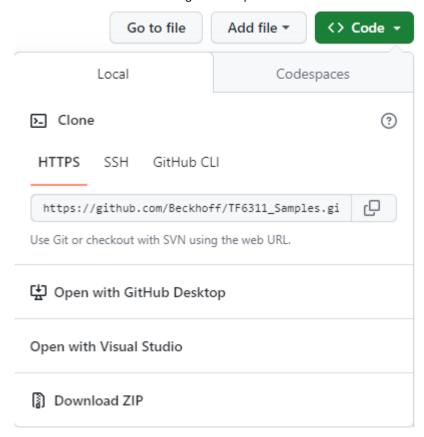
- The function block must call the RegisterReceiver() / Open() method in the same context as it calls the CheckReceived() method in the cyclic process.
- The callbacks via ReceiveData()/...Event() are called in the same context as the CheckReceived() from the function block of the application previously.



# 6 Examples

These examples provide easy-to-follow demonstrations for dealing with the TCP/UDP RT module.

Sample code and configurations for this product can be obtained from the corresponding repository on GitHub: https://github.com/Beckhoff/TF6311\_Samples. There you have the option to clone the repository or download a ZIP file containing the sample.



# 6.1 S01: Simple TCP Client (PLC / C++)

This sample shows the application of a TCP connection as client.

In this sample illustrates opening of a TCP connection with an IP address via port 80. The Beckhoff web server is used. The sample uses the connection to send an HTTP request to access a test website 62.159.14.51:80.

If the website does not fit into the receive buffer, the ReceiveData() method is called several times.

The client re-establishes a connection, if it was closed by the server, for example.

The sample is available for C++ and for the PLC.

# 6.1.1 S01: Simple TCP Client (C++)

This example implements a TCP client that issues a simple HTTP request and receives the response.

The download available here is preconfigured to call a test website 62.159.14.51:80.

### **Download**

Download the sample: https://github.com/Beckhoff/TF6311 Samples/tree/main/S01-lpStackTcpClient

- 1. Get the sample from GitHub, unzip the downloaded ZIP file if necessary
- 2. Open the project with TwinCAT XAE
- 3. Select your target system



- 4. Configure the network card (see below) for the target system
- 5. Build the **sample on your local machine** (e.g. Build->Build Solution)
- 6. Activate the configuration

## **Description**

The example is described in detail on the Quick Start [▶ 22] page.

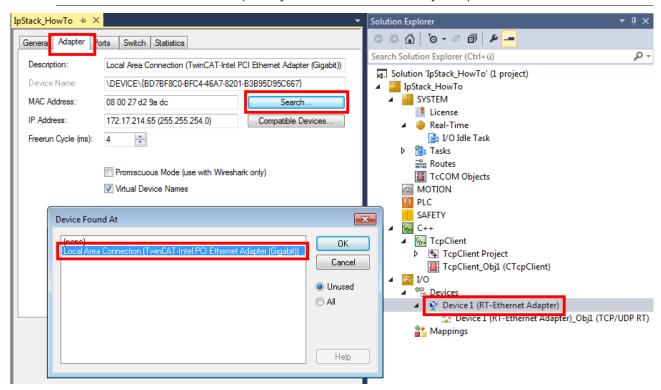
# Preparing the network card

For the TCP/UDP RT module, make sure that the RT Ethernet adapter in the TwinCAT solution is connected with the correct network card (with TwinCAT driver).

## Local configuration only



Installation of the driver on compatible network cards via the button "Compatible Devices" always takes place locally. On a controller with TwinCAT XAR, the program TcRteInstall.exe can be used. It is included in the installation (usually under C:TwinCAT\3.1\System).



# 6.1.2 S01: Simple TCP Client (PLC)

This sample implements a TCP client that issues a simple HTTP request and receives the response.

The download available here is preconfigured to call a test website 62.159.14.51:80.

#### **Download**

Download the sample: <a href="https://github.com/Beckhoff/TF6311\_Samples/tree/main/S01-lpStackTcpClientPlc">https://github.com/Beckhoff/TF6311\_Samples/tree/main/S01-lpStackTcpClientPlc</a>

- 1. Get the sample from GitHub, unzip the downloaded ZIP file if necessary
- 2. Open the project with TwinCAT XAE
- 3. Select your target system
- 4. Configure the network card (see below) for the target system
- 5. Build the **sample on your local machine** (e.g. Build->Build Solution)
- 6. Activate the configuration



### **Description**

After the startup, the PLC program can be used by setting the variable "bSend" to TRUE. The HTTP request (stored in "sMessage") is sent to the server, once the connection has been established. The first bytes of the incoming response are provided in "sLastReturnedMessage". The "sLastReturnedMessafeLength" indicates the whole length of the response.

The server address is defined in the FB\_init method.

The same sample is described in detail for C++ on the Quick Start [▶ 22] page.

### Preparing the network card

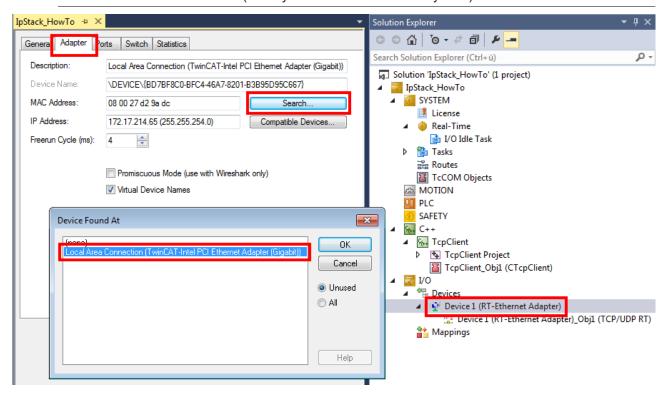
For the TCP/UDP RT module, make sure that the RT Ethernet adapter in the TwinCAT solution is connected with the correct network card (with TwinCAT driver).

# •

# Local configuration only



Installation of the driver on compatible network cards via the button "Compatible Devices" always takes place locally. On a controller with TwinCAT XAR, the program TcRteInstall.exe can be used. It is included in the installation (usually under C:TwinCAT\3.1\System).



# 6.2 S02: UDP Client Server (PLC/C++)

This example describes how a TwinCAT project can act as a UDP server. Thus, values can be delivered to the real-time or from the real-time on request.

The example implements an "echo service": A UDP server is started on a port (default: 10000). If this server receives a UDP packet, it returns the content to the sender (with same IP and same port). The example is available in  $PLC \ [\triangleright 40]$  and  $C++ \ [\triangleright 43]$ .

For testing purposes, a <u>UDP client [ 44]</u> (written in .NET) is also available.

The samples are also available in more detail as Quick Starts [▶ 11].

# 6.2.1 S02: UDP Demo (PLC)

This example describes a UDP server that is implemented in a PLC project.



It receives UDP packets and returns them to the sender ("echo server").

#### **Download**

Download the sample: https://github.com/Beckhoff/TF6311 Samples/tree/main/S02-UdpDemoPlc

- 1. Get the sample from GitHub, unzip the downloaded ZIP file if necessary
- 2. Open the project with TwinCAT XAE
- 3. Select your target system
- 4. Configure the network card (see below) for the target system
- 5. Build the sample on your local machine (e.g. Build->Build Solution)
- 6. Activate the configuration

### **Description**

The sample is also available in more detail as Quick Start.

The interface <u>ITcloUdpProtocolRecv</u> [▶ 52] is implemented and a pointer to a <u>ITcloUdpProtocol</u> [▶ 52] is used analogous to the <u>Quick Start</u> [▶ 22] in this sample.

To this end a PLC block is created, which implements the interface <u>ITcloUdpProtocolRecv</u> [▶ 52] ("Add POU" with "Implements"). It is important to realize the connection to the TCP/UDP RT object in the "FB\_init" and "FB\_exit" methods. This procedure is described in more detail in <u>Sample 11</u> of the C++ documentation.

The implementing function block (in sample UdpReceiver) calls the method "CheckReceived". In this way the IP stack is enabled to process incoming packets and transmit callbacks on the "ReceiveData" method of the function block.

The "ReceiveData" method uses the "SendData" method to return the data to the sender ("echo server").

### Understanding

Two methods are used to establish the communication between the function block and the TcCOM object "TCP/UDP RT":

- "FB init": This is executed automatically when the PLC is started
- · "FB exit": This is executed automatically when the PLC is stopped

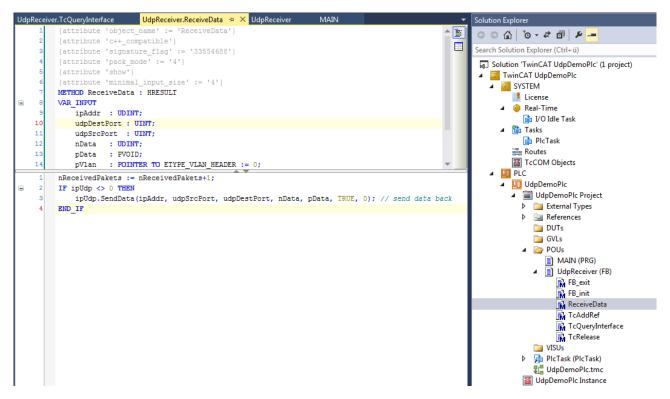
This initialization phase can largely be taken from the sample code.

Two methods are responsible for the actual UDP functionality in the PLC code:

- The "ReceiveData" method in the implemented function block receives the data.
- The "SendData" method in the ITcloUdpProtocol interface sends data.

In the sample, the "SendData" method is used in the "ReceiveData" method to return the received data:





The TcQueryInterface method must be implemented as follows to ensure that TwinCAT detects that the corresponding interface was implemented:

```
ipUdpRecv : ITcIoUdpProtocolRecv;
ipUnknown : ITcUnknown;
END_VAR
IF GuidsEqual(ADR(iid), ADR(TC GLOBAL IID LIST.IID ITcIoUdpProtocolRecv)) THEN
ipUdpRecv := THIS^; // cast to interface pointer
pipItf^ := ITCUNKNOWN_TO_PVOID(ipUdpRecv);
TcAddRef();
TcQueryInterface := S OK;
ELSIF GuidsEqual (ADR (iid), ADR (TC GLOBAL IID LIST.IID ITcUnknown)) THEN
ipUnknown := THIS^; // cast to interface pointer
pipItf^ := ITCUNKNOWN TO PVOID(ipUnknown);
TcAddRef();
TcQueryInterface := S OK;
ELSE
TcQueryInterface := E HRESULTAdsErr.NOINTERFACE ; //Call super if this fb extends some other
END IF
```

The additionally created methods

· TcAddRef / TcRelease

are inherited by the ITcUnknown interface and are not relevant in this context. For background information we suggest reading the chapter on the TcCOM module concept in the C++ domain.

### Preparing the network card

For the TCP/UDP RT module, make sure that the RT Ethernet adapter in the TwinCAT solution is connected with the correct network card (with TwinCAT driver).

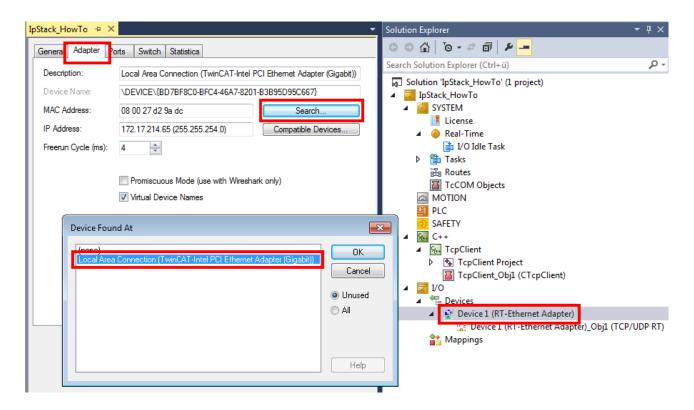


### Local configuration only



Installation of the driver on compatible network cards via the button "Compatible Devices" always takes place locally. On a controller with TwinCAT XAR, the program TcRteInstall.exe can be used. It is included in the installation (usually under C:TwinCAT\3.1\System).





# 6.2.2 S02: UDP Demo (C++)

This example describes a UDP server that is implemented in C++.

It receives UDP packets and returns them to the sender ("echo server").

#### **Download**

Download the sample: https://github.com/Beckhoff/TF6311\_Samples/tree/main/S02-UdpDemo

- 1. Get the sample from GitHub, unzip the downloaded ZIP file if necessary
- 2. Open the project with TwinCAT XAE
- 3. Select your target system
- 4. Configure the network card (see below) for the target system
- 5. Build the sample on your local machine (e.g. Build->Build Solution)
- 6. Activate the configuration

#### **Description**

The interface <u>ITcloUdpProtocolRecv</u> [▶ <u>52</u>] is implemented and a pointer to a <u>ITcloUdpProtocol</u> [▶ <u>52</u>] is used analogous to the <u>Quick Start</u> [▶ <u>22</u>] in this example.

Using "RegisterReceiver" in the Transition SO ensures that the module is registered for the transmitted port (default: 10000). A corresponding unregistration takes place in the Transition OS.

The "CheckReceived" method is called in the "CycleUpdate" method. In this way the TCP/UDP RT module is enabled to process incoming packets and transmit callbacks on the "ReceiveData" method to the module.

The "ReceiveData" method uses the "SendData" method to return the data to the sender ("echo server").

The sample is also available in more detail as Quick Start [ 16].

## Preparing the network card

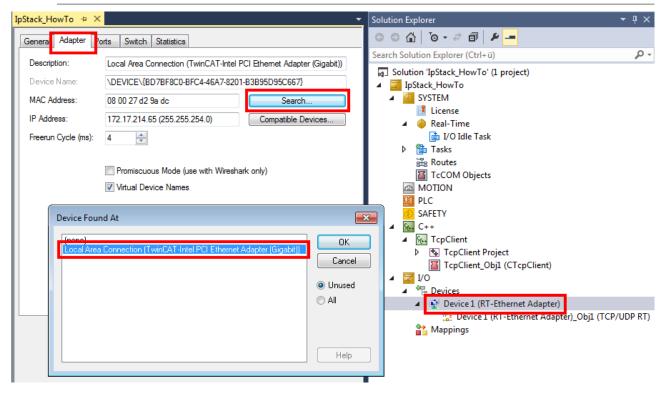
For the TCP/UDP RT module, make sure that the RT Ethernet adapter in the TwinCAT solution is connected with the correct network card (with TwinCAT driver).



# Local configuration only



Installation of the driver on compatible network cards via the button "Compatible Devices" always takes place locally. On a controller with TwinCAT XAR, the program TcRteInstall.exe can be used. It is included in the installation (usually under C:TwinCAT\3.1\System).



# 6.2.3 Test client

The test client is used to send and receive single UDP data packets to and from a UDP server.

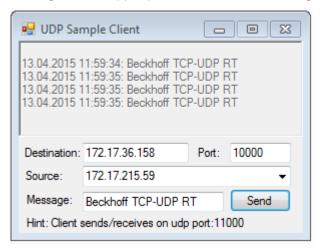
#### **Download**

Download the test client.

Unpack the ZIP file; the .exe file runs on a Windows system.

### **Description**

The client itself uses port 11000 for sending. At the same time it opens this port and displays received messages in the upper part of the interface as a log:





Together with the PLC / C++ samples, this results in an echo sample:

A UDP message is sent from the client port 11000 to the server port 10000, which sends the same data back to the sender.

The client can be configured via the interface:

- · Destination: Destination IP address
- · Port: The port that is addressed in the target
- Source: Sender network card (IP address).
   "OS-based" operating system deals with selection of the appropriate network card.
- Message

The TF6311 "TCP/UDP Realtime" does not allow local communication. However, for testing purposes a different network interface can be selected via "Source", so that the UDP packet leaves the computer through one network card and arrives on the other network card ("loop cable").

# 6.3 S03: ARP PING Demo (C++)

This example describes an ARP and PING client.

#### **Download**

Download the sample: https://github.com/Beckhoff/TF6311\_Samples/tree/main/S03-PingClient

- 1. Get the sample from GitHub, unzip the downloaded ZIP file if necessary
- 2. Open the project with TwinCAT XAE
- 3. Select your target system
- 4. Configure the network card (see below) for the target system
- 5. Build the sample on your local machine (e.g. Build->Build Solution)
- 6. Activate the configuration

### **Description**

The interface <u>ARP/Ping: ITcloArpPingProtocol(Recv) [ 64]</u> is implemented and a pointer to a <u>ARP/Ping: ITcloArpPingProtocol(Recv) [ 65]</u> is used analogous to the <u>Quick Start [ 22]</u> in this example.

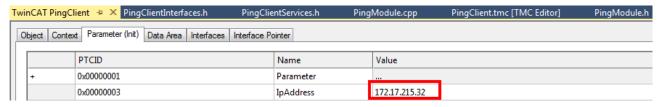
Using "RegisterReceiver" in the Transition SO ensures that the module is registered for receiving Arp and Ping messages. A corresponding unregistration takes place in the Transition OS.

The "CheckReceived" method is called in the "CycleUpdate" method. In this way the TCP/UDP RT module is enabled to process incoming packets and transmit callbacks on the "ArpReply" und "PingReply" methods to the module.

### Understanding

The procedure is carried out once when the program starts.

If "m\_bSendRequest" is set to TRUE (e.g. through TwinCAT Live Watch), a new request (ARP and Ping) is sent to the IP address defined here:



The output is in the messages:





To output these messages the "Tracelevel" can be configured (via Info).

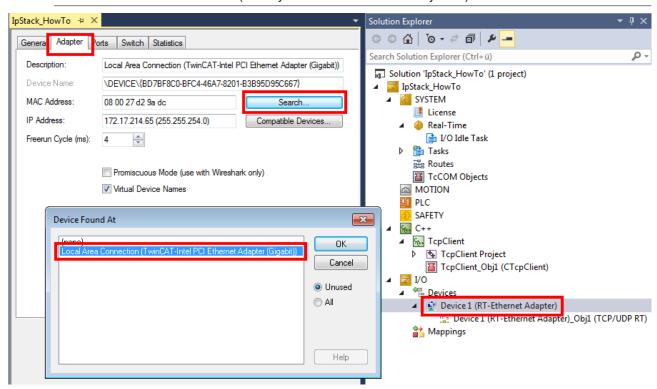
### Preparing the network card

For the TCP/UDP RT module, make sure that the RT Ethernet adapter in the TwinCAT solution is connected with the correct network card (with TwinCAT driver).

# Local configuration only



Installation of the driver on compatible network cards via the button "Compatible Devices" always takes place locally. On a controller with TwinCAT XAR, the program TcRteInstall.exe can be used. It is included in the installation (usually under C:TwinCAT\3.1\System).



# 6.4 S04: TCP Echo Server (PLC / C++)

This sample describes a TCP server accepting an income connection. Data sent to this server are simply returned as "echo".

The same sample is available for C++ and PLC. By default, the server runs on port 11000.

# Testing the sample

The sample can be tested via "telnet".

```
%>telnet 192.168.1.1 11000
```

If a character is sent via telnet, it is returned immediately. A picture similar to the following emerges:





# 6.4.1 S04: TCP Server Demo (PLC)

This sample describes a TCP server that is implemented in a PLC project.

It accepts a TCP connection, receives TCP packets and returns them to the sender ("echo server").

### **Download**

Download the sample: https://github.com/Beckhoff/TF6311\_Samples/tree/main/S04-TCPServerPlc

- 1. Get the sample from GitHub, unzip the downloaded ZIP file if necessary
- 2. Open the project with TwinCAT XAE
- 3. Select your target system
- 4. Configure the network card (see below) for the target system
- 5. Build the sample on your local machine (e.g. Build->Build Solution)
- 6. Activate the configuration

## Description

The interface <u>ITcloTcpProtocolRecv</u> [▶ <u>58</u>] is implemented and a pointer to a <u>ITcloTcpProtocol</u> [▶ <u>58</u>] is used analogous to the <u>Quick Start</u> [▶ <u>11</u>]s in this sample.

To this end a PLC block is created, which implements the interface <u>ITcloUdpProtocolRecv</u> [▶\_58] ("Add POU" with "Implements"). It is important to realize the connection to the TCP/UDP RT object in the "FB\_init" and "FB\_exit" methods. In particular, the Quick Starts illustrate how this OnlineChange can be implemented securely. The procedure is described in more detail in Sample 11 of the C++ documentation.

The implementing function block (in sample TCPServer) calls the method "CheckReceived". In this way the IP stack is enabled to process incoming packets and transmit callbacks relating to the "ReceiveData" and "ReceiveEvent" methods of the function block.

In order to take into account incoming connections, a port is opened in FB\_init via "AllocSocket" and "Listen". "Accept" is called in the "ReceiveEvent", if an event to establish a connection has occurred.

In this sample the "ReceiveData" method uses the "SendData" method to return the data to the sender ("echo server").



### Understanding

Two methods are used to establish the communication between the function block and the TcCOM object "TCP/UDP RT":

- "FB init": This is executed automatically when the PLC is started.
- "FB exit": This is executed automatically when the PLC is stopped.

This initialization phase can largely be taken from the sample code.

Two methods are responsible for the actual TCP functionality in the PLC code:

- The "ReceiveData" method in the implemented function block receives the data.
- The "ReceiveEvent" method indicates events occurring at the implemented function block.
- The "SendData" method in the ITcloTcpProtocol interface sends data.

In the sample, the "SendData" method is used in the "ReceiveData" method to return the received data: The TcQueryInterface method must be implemented as follows to ensure that TwinCAT detects that the corresponding interface was implemented:

```
ipTcpRecv : ITcIoTcpProtocolRecv;
ipUnknown : ITcUnknown;
END_VAR
IF GuidsEqual(ADR(iid), ADR(TC_GLOBAL_IID_LIST.IID_ITcIoTcpProtocolRecv)) THEN
ipTcpRecv := THIS^; // cast to interface pointer
pipItf^ := ITCUNKNOWN TO PVOID(ipUdpRecv);
TcAddRef();
TcQueryInterface := S OK;
ELSIF GuidsEqual (ADR (\overline{1}id), ADR (TC GLOBAL IID LIST.IID ITCUNknown)) THEN
ipUnknown := THIS^; // cast to interface pointer
pipItf^ := ITCUNKNOWN TO PVOID(ipUnknown);
TcAddRef();
TcQueryInterface := S_OK;
ELSE
TcQueryInterface := E HRESULTAdsErr.NOINTERFACE ; //Call super if this fb extends some other
```

The additionally created methods

TcAddRef / TcRelease

are inherited by the ITcUnknown interface and are not relevant in this context. For background information we suggest reading the chapter on the TcCOM module concept in the C++ domain.

### Preparing the network card

For the TCP/UDP RT module, make sure that the RT Ethernet adapter in the TwinCAT solution is connected with the correct network card (with TwinCAT driver).

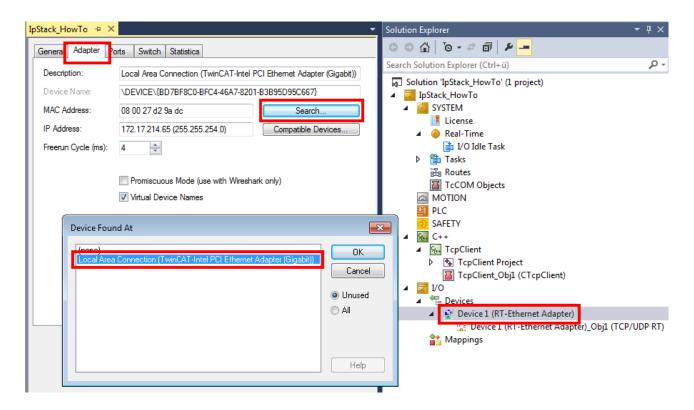


## Local configuration only



Installation of the driver on compatible network cards via the button "Compatible Devices" always takes place locally. On a controller with TwinCAT XAR, the program TcRteInstall.exe can be used. It is included in the installation (usually under C:TwinCAT\3.1\System).





# 6.4.2 **S04: TCP Server Demo (C++)**

This sample describes a TCP server that is implemented in C++.

It accepts a TCP connection, receives TCP packets and returns them to the sender ("echo server").

### **Download**

Download the sample: https://github.com/Beckhoff/TF6311\_Samples/tree/main/S04-TCPServer

- 1. Get the sample from GitHub, unzip the downloaded ZIP file if necessary
- 2. Open the project with TwinCAT XAE
- 3. Select your target system
- 4. Configure the network card (see below) for the target system
- 5. Build the sample on your local machine (e.g. Build->Build Solution)
- 6. Activate the configuration

#### **Description**

The interface ITcIoTcpProtocolRecv is implemented and a pointer to a ITcIoTcpProtocol is used analogous to the Quick Starts in this sample.

The "CheckReceived" method is called in the "CycleUpdate" method. In this way the TCP/UDP RT module is enabled to process incoming packets and transmit callbacks on the "ReceiveEvent" und "ReceiveData" methods to the module.

In order to take into account incoming connections, a port is opened in "CycleUpdate" via "AllocSocket" and "Listen". "Accept" is called in the "ReceiveEvent", if an event to establish a connection has occurred.

In this sample the "ReceiveData" method uses the "SendData" method to return the data to the sender ("echo server").

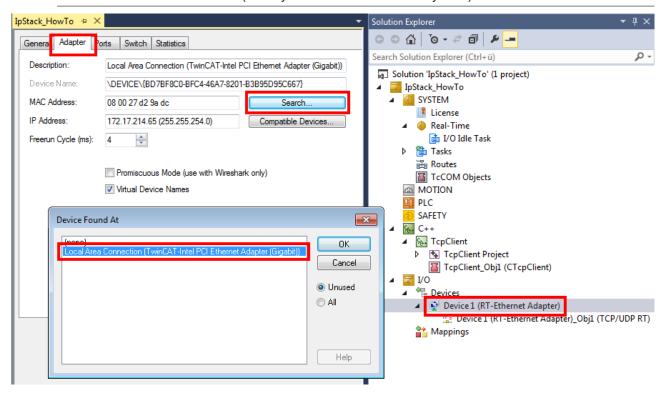
### Preparing the network card

For the TCP/UDP RT module, make sure that the RT Ethernet adapter in the TwinCAT solution is connected with the correct network card (with TwinCAT driver).



# Local configuration only

Installation of the driver on compatible network cards via the button "Compatible Devices" always takes place locally. On a controller with TwinCAT XAR, the program TcRteInstall.exe can be used. It is included in the installation (usually under C:TwinCAT\3.1\System).





# 7 Programmer's reference

The programmer's reference provides an overview of the different parameters, interfaces and their methods.

These include:

• <u>TCP/UDP RT TcCOM Parameters</u> [▶ <u>55]</u>: The parameters of the actual TCP/UDP RT module enable the configuration.

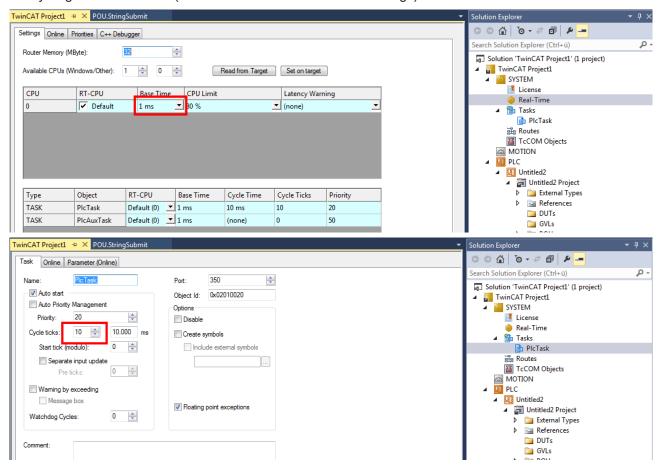
The TCP/UDP RT module can be used by different protocols. An InterfacePointer and an interface to be implemented always go hand in hand:

- <u>ITcloTcpProtocol(Recv)</u>: [▶ <u>58</u>] TCP/IP protocol
- ITcloUdpProtocol(Recv) [▶ 52]: UDP/IP protocol
- ITcloArpPingProtocol(Recv) [▶ 64]: ARP/Ping protocol

For all uses of IP addresses (e.g. "IpAddr"), the most significant elements are displayed in the last position. (Example: 192.168.2.1 -> 01 02 A8 C0)

### **Performance**

The TCP/UDP RT TcCOM object runs in real-time. Thus, the module is also directly dependent on the cycling of the real-time. The frequency with which data can be communicated can therefore be influenced by the cycling of the task used (and therefore also the real-time settings):



Communication via the network interface depends on this cycle. A corresponding call to the CheckReceived() methods (see <u>API documentation [> 51]</u>) must be made in each cycle.



### Incoming data: CheckReceived()

# Context of the incoming data

The customer must ensure that the method CheckReceived is called cyclically. Samples illustrate the procedure in PLC and C++

The CheckReveived() method is called cyclically in order to ensure that the data can be provided in the same context as the client project. The protocol-dependent Receive() methods of the customer project are called within this method call, if data have been received.

# Disconnection of Engineering connection on breakpoints

If breakpoints are used, we strongly advise to use different network interfaces, since a breakpoint stops parts of the TwinCAT systems, which may be relevant for the communication with Engineering.

# 7.1 UDP/IP: ITcloUdpProtocol(Recv)

The ITcloUdpProtocol and ITcloUdpProtocolRecv interfaces enable UDP/IP communication from the real-time environment.

A project that uses this interface contains a pointer to an ITcloUdpProtocol object and implements ITcloUdpProtocolRecv itself. ITcloUdpProtocolRecv serves as callback interface for receiving data from the TCP/UDP RT module within the application.

# Multiple calls of Receive()

1

During the implementation it should be noted that CheckReceived() will result in the callback to Receive() occurring several times within a cycle, if multiple packets have arrived between the cycles.

A buffer in the form a queue may therefore have to be provided.

### ITcloUdpProtocolRecv methods:

Name	Description
ReceiveData [▶ 53]	Is called by the TCP/UDP RT module as a callback to transfer data

# ITcloUdpProtocol methods:

Name	Description
SendData [▶ 53]	Sends data
CheckReceived [▶ 54]	Must be called cyclically. ReceiveData is used as callback in the context of this method (server and client functionality).
RegisterReceiver [ > 54]	Registering at the TCP/UDP RT module for receiving data.
UnregisterReceiver [▶ 55]	Unregistering at the TCP/UDP RT module for receiving UDP data.

The client and server implementation process is briefly described here. Only an overview is provided; the samples illustrate the application.

## Implementation of a UDP sender / receiver

Name	Description	
RegisterReceiver [ > 54]	Opens a port for incoming data packets.	
ReceiveData [▶ 53]	Is called when data packets arrive.	
SendData [▶ 53]	Can be used to send data.	
UnregisterReceiver [▶ 55]	For logout from (closing of) the port, e.g. during shutdown.	



To receive UDP data, registration is required by calling RegisterReceiver. This can be done in SetObjStateSO or FB init.

Data is provided by a callback of method ReceiveData from ITcloUdpProtocolRecv.

While TwinCAT switches from RUN mode to Config mode, all modules should unregister via UnregisterReceiver. This can be done in SetObjStateOS() or FB\_exit.

### NOTICE

### OnlineChange security

For OnlineChange security, RegisterReceiver should be called again.

# 7.1.1 Method ITcloUdpProtocolRecv:ReceiveData

Is called by the TCP/UDP RT module as a callback to transfer data.

### **Syntax**

HRESULT TCOMAPI ReceiveData(ULONG ipAddr, USHORT udpDestPort, USHORT udpSrcPort, ULONG nData, PVOID pData, ETYPE\_VLAN\_HEADER\* pVlan=0)

## Return value

Туре	Description
HRESULT	Indicates success and must be provided accordingly by the implemented module.

#### **Parameter**

Name	Туре	Description
ipAddr	ULONG	The IP address of the sender. IP addresses are displayed with the most significant element in the last position. (Example: 192.168.2.1 -> 01 02 A8 C0)
udpDestPort	USHORT	Port on which the data was received.
udpSrcPort	USHORT	Port of the sender.
nData	ULONG	Number of bytes received.
pData	PVOID	Pointer to the received data.
pVlan	ETYPE_VLAN_HEADER	ETYPE_VLAN_HEADER structure - see below.

The VLAN header represents information about the VLAN.

```
typedef struct _ETYPE_VLAN_HEADER
{
USHORT VLanType;
unsigned short VLanIdH : 4;
unsigned short reserved1 : 1;
unsigned short Priority : 3;
unsigned short VLanIdL : 8;
} ETYPE VLAN HEADER, *PETYPE VLAN HEADER;
```

# 7.1.2 Method ITcloUdpProtocol:SendData

Sends data.

### **Syntax**

HRESULT TCOMAPI SendData(ULONG ipAddr, USHORT udpDestPort, USHORT udpSrcPort, ULONG nData, PVOID pData, bool bCalcUdpCheckSum=0, ETYPE\_VLAN\_HEADER\* pVlan=0)



Туре	Description
HRESULT	Indicates success, see Return values [▶ 68].

#### **Parameter**

Name	Туре	Description
ipAddr	ULONG	The IP address of the receiver. IP addresses are displayed with the most significant element in the last position. (Example: 192.168.2.1 -> 01 02 A8 C0)
udpDestPort	USHORT	The port of the receiver.
udpSrcPort	USHORT	The port of the sender.
nData	ULONG	Number of date to be sent in bytes.
pData	PVOID	Pointer to the data to be sent.
bCalcUdpCheckSum	BOOL	Indicates whether the checksum should be calculated.
pVlan	ETYPE_VLAN_HEADER	ETYPE_VLAN_HEADER structure, see below.

# The VLAN header represents information about the VLAN.

```
typedef struct _ETYPE_VLAN_HEADER
{
USHORT VLanType;
unsigned short VLanIdH : 4;
unsigned short reserved1 : 1;
unsigned short Priority : 3;
unsigned short VLanIdL : 8;
} ETYPE_VLAN_HEADER, *PETYPE_VLAN_HEADER;
```

# 7.1.3 Method ITcloUdpProtocol:CheckReceived

Must be called cyclically; ReceiveData is used as callback in the context of this method (send and receive).

## **Syntax**

```
HRESULT TCOMAPI CheckReceived()
```

### Return value

Туре	Description
HRESULT	Indicates success, see Return values [▶ 68].

# 7.1.4 Method ITcloUdpProtocol:RegisterReceiver

Registering at the TCP/UDP RT module for receiving data.

### **Syntax**

HRESULT TCOMAPI RegisterReceiver(USHORT udpPort, ITcloUdpProtocolRecv\* ipRecv)

## Return value

Туре	Description
HRESULT	Indicates success, see Return values [▶ 68].



#### **Parameter**

Name	Туре	Description
udpPort	USHORT	Port on which the data is to be received.
ipRecv	ITcloUdpProtocolRecv*	Pointer to the receiver (Recv) interface.

# 7.1.5 Method ITcloUdpProtocol:UnregisterReceiver

Unregistering at the TCP/UDP RT module for receiving data.

### **Syntax**

HRESULT TCOMAPI UnregisterReceiver(USHORT udpPort)

## Return value

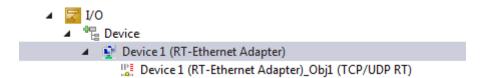
Туре	Description
HRESULT	Indicates success, see Return values [▶ 68].

#### **Parameter**

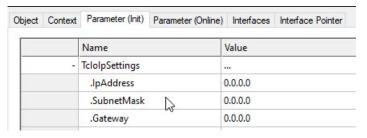
Name	Туре	Description
udpPort	USHORT	Port at which the data should no longer be received.

# 7.2 TCP/UDP RT TcCom Parameter

In addition to the interfaces, the TcCOM object "TCP/UDP RT" is the main component of the function. An instantiation usually takes place under the device:



Double-click to open the instance, so that the parameters documented below can be used:





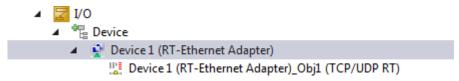
Name	Default value	Description
TclolpSettings.lpAddress	0.0.0.0	Own (local) IP address used for communication.
TclolpSettings.SubnetMask	0.0.0.0	Own subnet mask
TclolpSettings.Gateway	0.0.0.0	Gateway used to reach communication partners outside your own network.
TclolpSettings.DhcpEnable	FALSE	Not yet implemented.
TclolpSettings.ManualSettings	FALSE	Set to FALSE: The operating system uses the current IP configuration of the referenced adapter.
		Set to TRUE: Parameters of TclolpSettings* are used.
IpMaxReceivers	4	Maximum number of permitted IP-based protocols.
IpMaxPendingOnArp	40	Maximum number of entries in the ARP Request Table.
IpMacCacheSize	64	Number of entries in MAC cache, i.e. IP address to MAC address allocations. Caching is implemented as LRU.
IpMTU	1514	Not yet implemented. (Maximum transport unit size for IP packets)
IpRecvFrameQueueSize	255	Number of entries in the queue for receiving Udp packets.
UdpMaxReceivers	4	Maximum number of UDP receivers
UdpMTU	1514	From TwinCAT 3.1 Build 4026: Maximum Transport Unit size for UDP. Fragmentation is ready.
		In earlier versions (<= Build 4024) this parameter has no function
UdpCheckCrc	TRUE	Set to TRUE means that UDP packets with incorrect checksum are discarded.
TTL	0x80	TTL in the IP header of the frames to be sent.
MultiCastTTL	0x01	TTL of the MultiCast frames to be sent.
PassiveMode	FALSE	If TRUE, no frames are fetched from the RT network adapter frames by this instance. See Multitask access to a network card [▶ 36]
MulticastIpList		Multicast addresses for receiving MultiCast packets.
ТсрМТU	1514	Not yet implemented. (Maximum transport unit size for TCP)
TcpCheckCrc	TRUE	Incoming TCP frames are checked for valid checksum and discarded, if the checksum is incorrect.
TcpMaxSocketCount	32	Maximum number of sockets that are managed by the IP stack.
TcpReceiveBufferSize	16192	Number of received bytes that can be cached with a TCP connection.
TcpTransmitBufferSize	16192	Number of bytes to be sent that can be cached in the TCP stack with a connection.
TcpMaxRetry	5	Number of retries of TCP packets until the connection is terminated.
TcpTimeoutCon	5000	Timeout for TCP connection establishment and disconnection.
TcpTimeoutWait	60000	Timespan for storing handles internally after an unexpected termination of the connection.
TcpTimeoutIdle	1000	Time to callback (ReveiveEvent), if no response.
TcpRoundTripTime	3000	Start value for the timeout of data packets. Is adjusted dynamically depending on the connection quality (depending on the packet round-trip time).

Times are given in milliseconds.

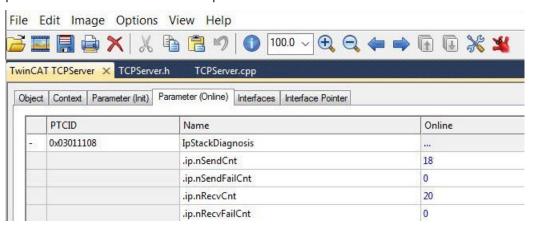


# 7.3 TCP/UDP RT TcCom diagnostics

The TcCOM object TCP/UDP RT represents the coupling of customer project with the hardware.



In addition to parameters, it therefore also contains diagnostic information, which is described here. Once Engineering can communicate with the target system and the program runs smoothly, various information is provided via the received and sent packets:



Name	Value	Description
IpStackDiagnosis		Diagnostic information of the IP stack
.ip.nSendCnt	18	Number of IP packets sent
.ip.nSendFailCnt	0	Number of IP packets not sent
.ip.nRecvCnt	20	Number of packets received
.ip.nRecvFailCnt	0	Number of packets not received
.arpRequest.nSendCnt	0	Arp-Requests: Number of packets sent
.arpRequest.nSendFailCnt	0	Arp-Requests: Number of packets not sent
.arpRequest.nRecvCnt	12	Arp-Requests: Number of packets received
.arpRequest.nRecvFailCnt	0	Arp-Requests: Number of packets not received
.arpReply.nSendCnt	12	Arp-Reply: Number of packets sent
.arpReply.nSendFailCnt	0	Arp-Reply: Number of packets not sent
.arpReply.nRecvCnt	0	Arp-Reply: Number of packets received
.arpReply.nRecvFailCnt	0	Arp-Reply: Number of packets not received
.pingRequest.nSendCnt	0	Ping-Request: Number of packets sent
.pingRequest.nSendFailCnt	0	Ping-Request: Number of packets not sent
.pingRequest.nRecvCnt	0	Ping-Request: Number of packets received
.pingRequest.nRecvFailCnt	0	Ping-Request: Number of packets not received
.pingReply.nSendCnt	0	Ping-Reply: Number of packets sent
.pingReply.nSendFailCnt	0	Ping-Reply: Number of packets not sent
.pingReply.nRecvCnt	0	Ping-Reply: Number of packets received
.pingReply.nRecvFailCnt	0	Ping-Reply: Number of packets not received
.nLinkStatusChangedCnt	1	Number of link changes
.nAllocFailCnt	0	Number of failed allocations
.nArpTimeoutFrames	0	Number of arp frames in the timeout
.nDroppedFrames	0	Number of discarded packages



# 7.4 TCP/IP: ITcloTcpProtocol(Recv)

The ITcloTcpProtocol and ITcloTcpProtocolRecv interfaces enable TCP/IP communication from the real-time environment.

A project that uses this interface contains a pointer to an ITcloTcpProtocol object and implements ITcloTcpProtocolRecv itself. ITcloTcpProtocolRecv serves as a callback interface for receiving data and events from the TCP/IP module within the application. The interfaces are based on a socket API. Before a socket can be used, it must be allocated with AllocSocket().

# ITcloTcpProtocolRecv methods:

Name	Description
ReceiveData [▶ 59]	Is called by the TCP/UDP RT module as a callback to transfer data.
ReceiveEvent [▶ 60]	Is called by the TCP/UDP RT module as a callback if an event has occurred.

## ITcloTcpProtocol methods:

Name	Description	
AllocSocket [▶ 60]	Allocates a socket.	
FreeSocket [ 61]	Enables a socket.	
Connect [▶ 61]	Establishes a connection to a remote terminal.	
IsConnected [▶ 61]	Indicates whether a socket is connected (for inbound and outbound connections).	
<u>Close [▶ 62]</u>	Closes a socket.	
<u>Listen [▶ 62]</u>	Opens a TCP port for incoming connections (see remarks).	
Accept [▶ 62]	For server functionality: Accepts incoming connections (see remarks).	
SendData [▶ 63]	Sends data (server and client functionality).	
CheckReceived [▶ 63]	Must be called cyclical; ReceiveEvent and ReceiveData are used as callback in the context of this method (server and client functionality).	
GetRemotelpAddr [▶ 63]	Returns the remote IP address of a communication partner.	
GetFreeSendDataSize [▶ 64]	Returns the number of free bytes in the TCP send buffer.	



Call CheckReceived() continuously.



Perhaps call AllocSocket() again in the event of an OnlineChange, in order to refresh the callback target.

The client and server implementation process is described here, independent of programming languages. Only an overview is provided; the samples illustrate the application.



# Implementation of an TCP server:

Name	Description
AllocSocket	Opens a socket.
[ <u><b>&gt;</b> 60]</u>	
<u>Listen [▶ 62]</u>	Opens a port on which connections are expected.
Accept [▶ 62]	Is called in the ReceiveEvent() method in order to accept a connection.
<u>ReceiveData</u>	Is called when data are received.
[ <u>&gt; 59</u> ]	
SendData [▶ 63]	Can be used to send data.
FreeSocket	On the Listen socket and all connection sockets for stopping.
[ <u><b>&gt;</b> 61]</u>	

## Code diagram for accepting a connection:

```
HRESULT CIpStackDemo::ReceiveEvent(ULONG socketId, TCPIP_EVENT tcpEvent)...
case TCPIP_EVENT_CONN_INCOMING:
m_spTcpProt->Accept(socketId);
break;
```

# Implementation of a TCP client:

Name	Description
AllocSocket	Opens a socket.
[ <u>\begin{align*}60]</u>	
Connect [▶ 61]	Starts connection establishment.
<u>IsConnected</u>	Checks whether the connection was established successfully.
[ <u><b>61</b></u> ]	
<u>ReceiveData</u>	Is called when data are received.
[ <u>&gt; 59</u> ]	
SendData [▶ 63]	Can be used to send data.
FreeSocket	On the Listen socket and all connection sockets for stopping.
[ <u>\beta 61]</u>	



# Disconnection by the operating system in Promiscuous mode



If Promiscuous mode is active at the RT Ethernet adapter in the "Adapter" tab, any TCP connection attempts are blocked by the operating system, since it does not recognize a port opened in the TCP/UDP RT object.

# 7.4.1 Method ITcloTcpProtocolRecv:ReceiveData

Is called by the TCP/UDP RT module as a callback to transfer data.

### **Syntax**

HRESULT TCOMAPI ReceiveData(ULONG socketId, ULONG nData, PVOID pData)

# Return value

Name	Туре	Description	
ReceiveData	HRESULT	Indicates success and must be provided accordingly by the implemented	
		module.	



#### **Parameter**

Name	Туре	Description
socketId	ULONG	The socket on which data was received.
nData	ULONG	Number of data received.
pData	PVOID	Pointer to the received data.

# 7.4.2 Method ITcloTcpProtocolRecv:ReceiveEvent

Is called by the TCP/UDP RT module as a callback if an event has occurred.

### **Syntax**

HRESULT TCOMAPI ReceiveEvent (ULONG socketId, TCPIP\_EVENT tcpEvent)

# Return value

Name	Туре	Description	
ReceiveEvent		Indicates success and must be provided accordingly by the implemented module.	

#### **Parameter**

Name	Туре	Description
socketId	ULONG	The socket on which data was received.
tcpEvent	TCP_EVENT	An element of the Enum.

The enumeration TCP EVENT refers to different events, which can occur with a TCP connection:

```
enum TCPIP_EVENT : ULONG {
TCPIP_EVENT_NONE = 0,
TCPIP_EVENT_ERROR = 1,
TCPIP_EVENT_RESET = 2,
TCPIP_EVENT_TIMEOUT = 3,
TCPIP_EVENT_CONN_ESTABLISHED = 4,
TCPIP_EVENT_CONN_INCOMING = 5,
TCPIP_EVENT_CONN_INCOMING = 6,
TCPIP_EVENT_CONN_IDLE = 7,
TCPIP_EVENT_DATA_RECEIVED = 8,
TCPIP_EVENT_DATA_SENT = 9,
TCPIP_EVENT_DATA_SENT = 10,
TCPIP_EVENT_LINKCONNECT = 11,
TCPIP_EVENT_LINKCONNECT = 12
};
```

An implementation of the method should provide a switch case over all elements, so that the system can respond according to the event.

The application of events for a TCP server is described in the interface overview.

# 7.4.3 Method ITcloTcpProtocol:AllocSocket

Allocates a socket.

## **Syntax**

HRESULT TCOMAPI AllocSocket(ITcIoTcpProtocolRecv\* ipRecv, ULONG& socketId)

# Return value

Туре	Description	
HRESULT	Indicates success, see Return values [▶ 68].	



### **Parameter**

Name	Туре	Description
ipRecv	ITcloTcpProtocolRecv	Pointer to the receiver (Recv) interface.
socketId	ULONG&	The generated socket.

# 7.4.4 Method ITcloTcpProtocol:FreeSocket

Enables a socket.

### **Syntax**

HRESULT TCOMAPI AllocSocket (ULONG socketId)

# Return value

Туре	Description
HRESULT	Indicates success, see Return values [▶ 68].

### **Parameter**

Name	Туре	Description
socketId	ULONG	The socket to be enabled.

# 7.4.5 Method ITcloTcpProtocol:Connect

Establishes a connection to a remote terminal.

### **Syntax**

HRESULT TCOMAPI Connect(ULONG socketId, ULONG ipRemoteAddress, USHORT tcpPort)

### Return value

Туре	Description
HRESULT	Indicates success, see Return values [▶ 68].

### **Parameter**

Name	Туре	Description	
socketId	ULONG	The socket to be used.	
ipRemoteAddress		IP address of the remote terminal to be contacted. IP addresses are displayed with the most significant element in the last position. (Example: 192.168.2.1 -> 01 02 A8 C0)	
tcpPort	USHORT	Port of the remote terminal to be contacted.	

# 7.4.6 Method ITcloTcpProtocol:IsConnected

Indicates whether a socket is connected (for inbound and outbound connections).

## **Syntax**

HRESULT TCOMAPI IsConnected (ULONG socketId)



Туре	Description	
HRESULT	Indicates success, see Return values [▶ 68].	

#### **Parameter**

Name	Туре	Description
socketld	ULONG	The socket to be used.

# 7.4.7 Method ITcloTcpProtocol:Close

Closes a socket.

# **Syntax**

HRESULT TCOMAPI Close (ULONG socketId)

# Return value

Туре	Description	
HRESULT	Indicates success, see Return values [▶ 68].	

### **Parameter**

Name	Туре	Description
socketld	ULONG	The socket to be closed.

# 7.4.8 Method ITcloTcpProtocol:Listen

Opens a TCP port for incoming connections. The application is described in the interface overview.

# **Syntax**

HRESULT TCOMAPI Listen (ULONG socketId, USHORT tcpPort)

## Return value

Туре	Description	
HRESULT	Indicates success, see Return values [▶ 68].	

# **Parameter**

Name	Туре	Description	
socketId	ULONG	The socket to be used.	
tcpPort	USHORT	The port which is scanned for incoming connections.	

# 7.4.9 Method ITcloTcpProtocol:Accept

Accepts income connections. The application is described in the interface overview.

# **Syntax**

HRESULT TCOMAPI Accept (ULONG socketId)



Туре	Description	
HRESULT	Indicates success, see Return values [▶ 68].	

#### **Parameter**

Name	Туре	Description
socketId	ULONG	The socket to be used.

# 7.4.10 Method ITcloTcpProtocol:SendData

Sends data (server and client functionality).

# **Syntax**

HRESULT TCOMAPI SendData(ULONG socketId, ULONG nData, PVOID pData, ULONG& nSendData)

# Return value

Туре	Description
HRESULT	Indicates success, see Return values [▶ 68].

#### **Parameter**

Name	Туре	Description
socketId	ULONG	The socket to be used.
nData	ULONG	Length of the data to be sent.
pData	PVOID	Pointer to the data to be sent.
nSendData		Returns the number of sent bytes. If this is smaller than nData, the data should be re-sent.

# 7.4.11 Method ITcloTcpProtocol:CheckReceived

Must be called cyclical; ReceiveEvent and ReceiveData are used as callback in the context of this method (server and client functionality).

### **Syntax**

HRESULT TCOMAPI CheckReceived()

### Return value

Туре	Description
HRESULT	Indicates success, see Return values [▶ 68].

### **Parameter**

-

# 7.4.12 Method ITcloTcpProtocol:GetRemotelpAddr

Returns the remote IP address of a communication partner.

## **Syntax**

HRESULT TCOMAPI GetRemoteIpAddr(ULONG socketId, ULONG& remoteIpAddr)



Туре	Description
HRESULT	Indicates success, see Return values [▶ 68].

#### **Parameter**

Name	Туре	Description
socketId	ULONG	The socket to be used.

# 7.4.13 Method ITcloTcpProtocol:GetFreeSendDataSize

Returns the number of free bytes in the TCP send buffer.

### **Syntax**

HRESULT TCOMAPI GetRemoteIpAddr(ULONG socketId, ULONG& nData)

# Return value

Туре	Description
HRESULT	Indicates success, see Return values [▶ 68].

#### **Parameter**

Name	Туре	Description
socketId	ULONG	The socket to be used.
nData	ULONG&	Returns the free bytes in the buffer.

# 7.5 ARP/Ping: ITcloArpPingProtocol(Recv)

The interfaces ITcloArpPingProtocol and ITcloArpPingProtocolRecv enable sending of ARP and Ping messages from the real-time environment.

A project that uses this interface contains a pointer to an ITcloArpPingProtocol object and implements ITcloArpPingProtocolRecv itself. ITcloArpPingProtocolRecv serves as callback interface for receiving data from the TCP/UDP RT module within the application.

## ITcloArpPingProtocolRecv methods:

Name	Description	
ArpReply [▶ 65]	Callback function that is invoked when an ArpReply message is received.	
PingReply [▶ 65]	Callback function that is invoked when an PingReply message is received.	

If these methods return S\_OK, the packet is regarded as processed and is not forwarded to the operating system. If necessary, S\_FALSE should be returned.



## ITcloArpPingProtocol methods:

Name	Description
ArpRequest [▶ 66]	Sends an ArpRequest
PingRequest [▶ 66]	Sends a PingRequest
RegisterReceiver [▶ 67]	Registering at the TCP/UDP RT module for receiving data.
UnregisterReceiver [▶ 67]	Unregistering at the TCP/UDP RT module for receiving data.
CheckReceived [▶ 68]	Must be called cyclically; ArpReply and PingReply are used as callback in the context of this method

To receive ARP or Ping data, registration is required by calling RegisterReceiver. This can be done in SetObjStateSO().

Data is provided by a callback of method ArpReceive or PingReceive from ITcloArpPingProtocolRecv.

During the shutdown, all modules should unregister via UnregisterReceiver. This can be done in SetObjStateOS().

# 7.5.1 Method ITcloArpPingProtocolRecv:PingReply

Callback function that is invoked when an PingReply message is received.

# **Syntax**

HRESULT TCOMAPI PingReply(ULONG ipAddr, ULONG nData, PVOID pData, ETYPE\_VLAN\_HEADER\* pVlan=0)

## Return value

Туре	Description
HRESULT	Indicates success and must be provided accordingly by the implemented module. If this is
	not S_OK, the response continues to be transferred to the operating system.

#### **Parameter**

Name	Туре	Description
ipAddr	ULONG	The IP address of the search.
nData	ULONG	Number of bytes received.
pData	PVOID	Pointer to the received data.
pVlan	ETYPE VLAN HEADER*	ETYPE_VLAN_HEADER structure, see below.

The VLAN header represents information about the VLAN.

```
typedef struct _ETYPE_VLAN_HEADER
{
   USHORT VLanType;
   unsigned short VLanIdH : 4;
   unsigned short reserved1 : 1;
   unsigned short Priority : 3;
   unsigned short VLanIdL : 8;
} ETYPE VLAN HEADER, *PETYPE VLAN HEADER;
```

# 7.5.2 Method ITcloArpPingProtocolRecv:ArpReply

Callback function that is invoked when an ArpReply message is received.

### **Syntax**

HRESULT TCOMAPI ArpReply(ULONG ipAddr, ETHERNET ADDRESS macAddr, ETYPE VLAN HEADER\* pVlan=0)



Туре	Description
	Indicates success and must be provided accordingly by the implemented module. If this is not S_OK, the response continues to be transferred to the operating system.

### **Parameter**

Name	Туре	Description
ipAddr	ULONG	The IP address of the search.
macAddr	ETHERNET_ADDRESS	Determined MAC address.
pVlan	ETYPE VLAN HEADER*	ETYPE_VLAN_HEADER structure, see below.

### The VLAN header represents information about the VLAN.

```
typedef struct _ETYPE_VLAN_HEADER
{
USHORT VLanType;
unsigned short VLanIdH : 4;
unsigned short reserved1 : 1;
unsigned short Priority : 3;
unsigned short VLanIdL : 8;
} ETYPE_VLAN_HEADER, *PETYPE_VLAN_HEADER;
```

# 7.5.3 Method ITcloArpPingProtocol:PingRequest

Sends a ping request.

### **Syntax**

HRESULT TCOMAPI PingRequest(ULONG ipAddr, ULONG nData=0, PVOID pData=0, ETYPE\_VLAN\_HEADER\* pVlan=0)

# Return value

Туре	Description
HRESULT	Indicates success, see Return values [▶ 68].

### **Parameter**

Name	Туре	Description
ipAddr	ULONG	The IP address of the target.
nData	ULONG	Number of bytes received.
pData	PVOID	Pointer to the received data.
pVlan	ETYPE_VLAN_HEADER*	ETYPE_VLAN_HEADER structure, see below.

# The VLAN header represents information about the VLAN.

```
typedef struct _ETYPE_VLAN_HEADER
{
USHORT VLanType;
unsigned short VLanIdH : 4;
unsigned short reserved1 : 1;
unsigned short Priority : 3;
unsigned short VLanIdL : 8;
} ETYPE_VLAN_HEADER, *PETYPE_VLAN_HEADER;
```

# 7.5.4 Method ITcloArpPingProtocol:ArpRequest

Sends an ARP request.

## **Syntax**

HRESULT TCOMAPI ArpRequest(ULONG ipAddr, ETHERNET\_ADDRESS\* macAddr=0, ETYPE\_VLAN\_HEADER\* pVlan=0)

66 Version: 1.8.1 TF6311



Туре	Description
HRESULT	Indicates success, see Return values [▶ 68].

#### **Parameter**

Name	Туре	Description
ipAddr	ULONG	The IP address of the target.
macAddr	ETHERNET_ADDRESS*	Restriction of the MAC address.
pVlan	ETYPE_VLAN_HEADER*	ETYPE_VLAN_HEADER structure, see below.

# The VLAN header represents information about the VLAN.

```
typedef struct _ETYPE_VLAN_HEADER
{
USHORT VLanType;
unsigned short VLanIdH : 4;
unsigned short reserved1 : 1;
unsigned short Priority : 3;
unsigned short VLanIdL : 8;
} ETYPE_VLAN_HEADER, *PETYPE_VLAN_HEADER;
```

# 7.5.5 Method ITcloArpPingProtocol:RegisterReceiver

Registering at the TCP/UDP RT module for receiving responses (ARP / Ping).

### **Syntax**

HRESULT TCOMAPI RegisterReceiver(ITcIoArpPingRecv\* ipRecv)

## Return value

Туре	Description
HRESULT	Indicates success, see Return values [▶ 68].

### **Parameter**

Name	Туре	Description
ipRecv	ITcIoArpPingRecv*	Pointer to the receiver (Recv) interface.

# 7.5.6 Method ITcloArpPingProtocol:UnregisterReceiver

Unregistering at the TCP/UDP RT module for receiving responses (ARP / Ping).

# **Syntax**

HRESULT TCOMAPI UnregisterReceiver(ITcIoArpPingRecv\* ipRecv)

# Return value

Туре	Description
HRESULT	Indicates success, see Return values [▶ 68].

#### **Parameter**

Name	Туре	Description
ipRecv	ITcloArpPingRecv	Reference to the receiver to be unregistered



# 7.5.7 Method ITcloArpPingProtocol:CheckReceived

Must be called cyclically; ArpReply and PingReply are used as callback in the context of this method.

# **Syntax**

HRESULT TCOMAPI CheckReceived()

# Return value

Туре	Description
HRESULT	Indicates success, see Return values [▶ 68].

# 7.6 Return values

The interface functions have HRESULT as return values. The returned values are derived from the <u>ADS Return Codes [▶ 70]</u>. Their meaning for TF6311:

Value (Enum)	Value (Numeric)	Description	
ADS_E_INVALI DPARM	0x9811070B	Socket not allocated/known, transferred pointer NULL	
ADS_E_NOMO REHDLS	0x98110716	No free sockets available. Default: 32 see TCP/UDP RT TcCom Parameter [▶ 55]	
ADS_E_INCOM PATIBLE	0x9811070E	Socket in wrong state. E.g. Connect() attempt, if a socket was previous used with Listen(); Close() without previous connection; Send() without connection; Socket Listen(), if a Listen() call was already issued.	
ADS_E_INVALI DSTATE	0x98110712	TCP/UDP RT object is not in OP mode	
ADS_E_INVALI DDATA	0x98110706	Problem with parameter. E.g. pData==NULL for SendData	
ADS_E_EXIST S	0x9811070F	Port already used otherwise	
ADS_E_PENDI NG	0x9811071E	Not all data were sent (SendData)	
S_OK	0x0	Call successful. IsConnected(): Connection exists	
S_FAIL	0x1	Call not successful, general error IsConnected(): Connection does not exist	

The values from the range 0x9811 are defined in the enumeration "E\_HRESULTAdsErr" (PLC) and corresponding ADS\_E\_\* (C++) "defines".

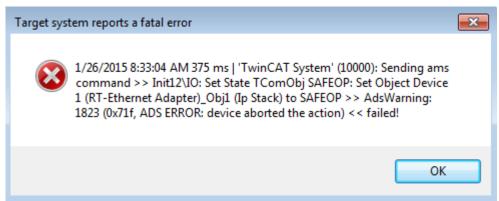


# 8 Fault analysis

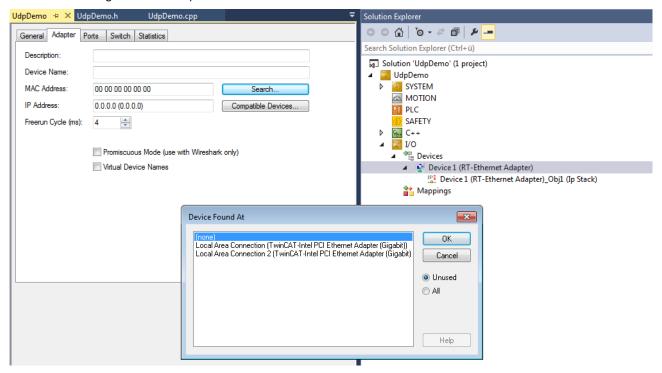
At this point, it is usual practice to list problems or situations in connection with handling the product, together with an error description.

# 8.1 Start-up: Ip Stack ADS 1823 / 0x71f

If ADS error 1823 (0x71f) occurs when an IP stack TcCOM object is started, the configuration of the network card is probably incorrect.



Check the settings under "Adapter" for the network card in the Solution:



The configuration of the network card for the TCP/UDP RT module is documented in more detail <a href="here">here</a> [> 14].



# 9 Appendix

# 9.1 ADS Return Codes

Grouping of error codes:

Global error codes: ADS Return Codes [▶ 70]... (0x9811\_0000 ...)
Router error codes: ADS Return Codes [▶ 70]... (0x9811\_0500 ...)
General ADS errors: ADS Return Codes [▶ 71]... (0x9811\_0700 ...)
RTime error codes: ADS Return Codes [▶ 73]... (0x9811\_1000 ...)

### Global error codes

Hex	Dec	HRESULT	Name	Description
0x0	0	0x98110000	ERR_NOERROR	No error.
0x1	1	0x98110001	ERR_INTERNAL	Internal error.
0x2	2	0x98110002	ERR_NORTIME	No real time.
0x3	3	0x98110003	ERR_ALLOCLOCKEDMEM	Allocation locked – memory error.
0x4	4	0x98110004	ERR_INSERTMAILBOX	Mailbox full – the ADS message could not be sent. Reducing the number of ADS messages per cycle will help.
0x5	5	0x98110005	ERR_WRONGRECEIVEHMSG	Wrong HMSG.
0x6	6	0x98110006	ERR_TARGETPORTNOTFOUND	Target port not found – ADS server is not started, not reachable or not installed.
0x7	7	0x98110007	ERR_TARGETMACHINENOTFOUND	Target computer not found – AMS route was not found.
0x8	8	0x98110008	ERR_UNKNOWNCMDID	Unknown command ID.
0x9	9	0x98110009	ERR_BADTASKID	Invalid task ID.
0xA	10	0x9811000A	ERR_NOIO	No IO.
0xB	11	0x9811000B	ERR_UNKNOWNAMSCMD	Unknown AMS command.
0xC	12	0x9811000C	ERR_WIN32ERROR	Win32 error.
0xD	13	0x9811000D	ERR_PORTNOTCONNECTED	Port not connected.
0xE	14	0x9811000E	ERR_INVALIDAMSLENGTH	Invalid AMS length.
0xF	15	0x9811000F	ERR_INVALIDAMSNETID	Invalid AMS Net ID.
0x10	16	0x98110010	ERR_LOWINSTLEVEL	Installation level is too low –TwinCAT 2 license error.
0x11	17	0x98110011	ERR_NODEBUGINTAVAILABLE	No debugging available.
0x12	18	0x98110012	ERR_PORTDISABLED	Port disabled – TwinCAT system service not started.
0x13	19	0x98110013	ERR_PORTALREADYCONNECTED	Port already connected.
0x14	20	0x98110014	ERR_AMSSYNC_W32ERROR	AMS Sync Win32 error.
0x15	21	0x98110015	ERR_AMSSYNC_TIMEOUT	AMS Sync Timeout.
0x16	22	0x98110016	ERR_AMSSYNC_AMSERROR	AMS Sync error.
0x17	23	0x98110017	ERR_AMSSYNC_NOINDEXINMAP	No index map for AMS Sync available.
0x18	24	0x98110018	ERR_INVALIDAMSPORT	Invalid AMS port.
0x19	25	0x98110019	ERR_NOMEMORY	No memory.
0x1A	26	0x9811001A	ERR_TCPSEND	TCP send error.
0x1B	27	0x9811001B	ERR_HOSTUNREACHABLE	Host unreachable.
0x1C	28	0x9811001C	ERR_INVALIDAMSFRAGMENT	Invalid AMS fragment.
0x1D	29	0x9811001D	ERR_TLSSEND	TLS send error – secure ADS connection failed.
0x1E	30	0x9811001E	ERR_ACCESSDENIED	Access denied – secure ADS access denied.

## **Router error codes**



Hex	Dec	HRESULT	Name	Description	
0x500	1280	0x98110500	ROUTERERR_NOLOCKEDMEMORY	Locked memory cannot be allocated.	
0x501	1281	0x98110501	ROUTERERR_RESIZEMEMORY	The router memory size could not be changed.	
0x502	1282	0x98110502	ROUTERERR_MAILBOXFULL	The mailbox has reached the maximum number of possible messages.	
0x503	1283	0x98110503	ROUTERERR_DEBUGBOXFULL	The Debug mailbox has reached the maximum number of possible messages.	
0x504	1284	0x98110504	ROUTERERR_UNKNOWNPORTTYPE	The port type is unknown.	
0x505	1285	0x98110505	ROUTERERR_NOTINITIALIZED	The router is not initialized.	
0x506	1286	0x98110506	ROUTERERR_PORTALREADYINUSE	The port number is already assigned.	
0x507	1287	0x98110507	ROUTERERR_NOTREGISTERED	The port is not registered.	
0x508	1288	0x98110508	ROUTERERR_NOMOREQUEUES	The maximum number of ports has been reached.	
0x509	1289	0x98110509	ROUTERERR_INVALIDPORT	The port is invalid.	
0x50A	1290	0x9811050A	ROUTERERR_NOTACTIVATED	The router is not active.	
0x50B	1291	0x9811050B	ROUTERERR_FRAGMENTBOXFULL	The mailbox has reached the maximum number for fragmented messages.	
0x50C	1292	0x9811050C	ROUTERERR_FRAGMENTTIMEOUT	A fragment timeout has occurred.	
0x50D	1293	0x9811050D	ROUTERERR_TOBEREMOVED	The port is removed.	

# **General ADS error codes**



0x700         1792         0x88110700         ADSERR DEVICE_RROR         General device error.           0x701         1793         0x98110702         ADSERR DEVICE_RVALIDEP         Service is not supported by the server.           0x702         1794         0x98110702         ADSERR DEVICE_INVALIDOR PINVALIDER CONTROL PROVIDED TO SERVICE INVALIDANCESS         Invalid index offset.           0x704         1796         0x98110703         ADSERR DEVICE_INVALIDANCESS         Reading or witing not permitted.           0x705         1797         0x98110705         ADSERR DEVICE_INVALIDANCE         Parameter size not correct.           0x707         1799         0x98110707         ADSERR_DEVICE_INVALIDORAT         Invalid data values.           0x709         1801         0x98110708         ADSERR_DEVICE_INVALIDCONTEXT         Invalid operating system context. This can result from use of ADS blocks in additional part of the proper system context. This can result from use of ADS blocks in additional part of ADS bl	Hex	Dec	HRESULT	Name	Description	
0.7071         1793         No98110701         ADSERR, DEVICE, SRYNOTSUPP         Service is not supported by the server.           0x702         1795         0x88110703         ADSERR, DEVICE, INVALIDOFFSET         Invalid index offset.           0x703         1796         0x88110703         ADSERR, DEVICE, INVALIDOFFSET         Invalid index offset.           0x704         1798         0x98110705         ADSERR, DEVICE, INVALIDOFFSET         Invalid index offset.           0x706         1798         0x98110706         ADSERR, DEVICE, INVALIDOATA         Invalid data values.           0x707         1799         0x98110709         ADSERR, DEVICE, INVALIDOATA         Device is not ready to operate.           0x708         1800         0x98110709         ADSERR, DEVICE, INVALIDOATEXT         Invalid operating system context. This can result from use of ADS blocks in different tasks. It may be possible to reserve bits bitrough multitasking synchronization in the PLC.           0x704         1802         0x98110700         ADSERR, DEVICE, INVALIDARAM         Invalid operating system context. This can result from use of ADS blocks in different tasks. It may be possible to reside bits bitrough multitasking synchronization in the PLC.           0x704         1802         0x98110700         ADSERR, DEVICE, INVALIDARAM         Invalid parameter values.           0x705         1803         0x98110700         ADSERR, DEVICE,					· · · · · · · · · · · · · · · · · · ·	
0x702         1794         0x88110702         ADSERR_DEVICE_INVALIDGRP         Invalid index group.           0x704         1798         0x88110703         ADSERR_DEVICE_INVALIDACCESS         Roading or writing not permitted.           0x704         1797         0x98110704         ADSERR_DEVICE_INVALIDACCESS         Roading or writing not permitted.           0x706         1798         0x98110705         ADSERR_DEVICE_INVALIDACE         Parameter size not correct.           0x707         1799         0x98110707         ADSERR_DEVICE_BUSY         Device is not ready to operate.           0x708         1800         0x98110709         ADSERR_DEVICE_BUSY         Device is not ready to operate.           0x709         1801         0x98110709         ADSERR_DEVICE_INVALIDCONTEXT         Invalid operating system context. This can result from use of ADS blocks in different tasks. It may be possible to resolve this through multitasking synchronization in the PLC.           0x700         1802         0x98110700         ADSERR_DEVICE_INOMEMORY         Invalid operating system context. This can result from use of ADS blocks in different tasks. It may be possible to resolve this through multitasking synchronization in the PLC.           0x701         1802         0x98110700         ADSERR_DEVICE_NOMEMORY         Invalid operating system context. This can result from use of ADS blocks in different tasks. It may be possible to resolve this through multitasking synchronization in th		-	-			
0.7036         17956         0.988110703         ADSERR DEVICE INVALIDOFSET         Invalid Index offset.           0.7704         7796         0.988110705         ADSERR DEVICE INVALIDACESS         Reading or willing not parmitted.           0.7705         7798         0.989110705         ADSERR DEVICE INVALIDATA         Invalid data values.           0.7707         1798         0.989110707         ADSERR DEVICE INVALIDATA         Invalid data values.           0.7707         1798         0.989110708         ADSERR DEVICE INVALIDATA         Device is not ready to operate.           0.7708         1800         0.989110709         ADSERR DEVICE INVALIDATA         Device is not ready to operate.           0.7708         1800         0.989110709         ADSERR DEVICE INVALIDARY         Device is busy.           0.7708         1801         0.989110709         ADSERR DEVICE INVALIDARY         Invalid perameter values.           0.7708         1803         0.989110700         ADSERR DEVICE INCOMPATIBLE         ON Dipotal device values.           0.7709         1805         0.989110700         ADSERR DEVICE INCOMPATIBLE         O) Dipota de not match.           0.7701         1807         0.989110711         ADSERR DEVICE EXISTS         O) Operation of the values.           0.7712         1810         0.9891						
0.776         1797         0.88110703         ADSERR DEVICE INVALIDACESS         Reading or writing not permitted.           0.7705         1797         0.88110705         ADSERR DEVICE INVALIDACES         Parameter size not correct.           0.7705         1799         0.881107070         ADSERR DEVICE INVALIDATA         Invalid data values.           0.7070         1799         0.88110709         ADSERR DEVICE INVALIDATA         Invalid data values.           0.7070         1801         0.88110709         ADSERR DEVICE BUSY         Device is busy.           0.7070         1801         0.88110700         ADSERR DEVICE INVALIDORNTEXT         Invalid operating system context. This can result from use of ADS blocks in different tasks. It may be possible to resolve this through multitasking synchronization in the PLC.           0.7070         1802         0.88110700         ADSERR DEVICE INVALIDORATM         Invalid operating system context. This can result from the PLC.           0.7070         1803         0.98110700         ADSERR DEVICE INVALIDATE         Invalid operating system context. This can result from use of ADS blocks in different tasks. It may be possible to resolve this through multitasking synchronization in the PLC.           0.7070         1803         0.98110700         ADSERR DEVICE INVALIDATE         Invalid operating system context.           0.7070         1805         0.98110701         ADSERR					<u> </u>	
0x706         1797         0x88110705         ASSERR DEVICE, INVALIDSIZE         Parameter size not correct.           0x707         1799         0x88110707         ADSERR DEVICE INVALIDATA         Invalid data values.           0x708         1800         0x88110709         ADSERR DEVICE INVALIDATA         Device is busy.           0x709         1800         0x98110709         ADSERR DEVICE INVALIDEONTEXT         Invalid operating system context. This can result from use of ADS blocks in different stakes. It may be possible to ranche that through multitasking synchronization in the PLC.           0x70A         1802         0x98110700         ADSERR DEVICE INVALIDEARIM Invalid operating system context. This can result from use of ADS blocks in different stakes. It may be possible to ranche that through multitasking synchronization in the PLC.           0x70B         1803         0x98110700         ADSERR DEVICE NOMEMORY         Invalid operating system context. This can result from use of ADS blocks in different stakes. It may be possible for ranche that through multitasking synchronization in the PLC.           0x70B         1803         0x98110700         ADSERR DEVICE NOTEOUND         Not found (files).           0x70C         1805         0x98110701         ADSERR DEVICE SYNTEXA         Syntax error in file or command.           0x710         1807         0x98110711         ADSERR DEVICE, SYMBOLNOTEOND         Optical stake yours of the purpose of the purpose of the pur	0x704	1796	0x98110704		Reading or writing not permitted.	
0x706         17980         0x881107076         ADSERR DEVICE_INVALIDATA         Invalid data values.           0x707         1799         0x881107076         ADSERR_DEVICE_BUSY         Device is not ready to operate.           0x709         1801         0x88110708         ADSERR_DEVICE_INVALIDCONTEXT         Invalid operating system context. This can result from use of ADS blocks in different tasks. It may be possible to resolve this through multitasking synchronization in the PLC.           0x70A         1802         0x88110708         ADSERR_DEVICE_INVALIDCANTEXT         Invalid operating system context. This can result from use of ADS blocks in different tasks. It may be possible to resolve this through multitasking synchronization in the PLC.           0x70B         1803         0x88110700         ADSERR DEVICE_INVALIDEARM         Invalid operating system context. This can result from the PLC.           0x70B         1806         0x88110700         ADSERR DEVICE_INVALIDEARM         Invalid parameter values.           0x70F         1806         0x88110701         ADSERR DEVICE_SYMED.         Not found (files).           0x710         1806         0x88110710         ADSERR_DEVICE_SYMED.         Opice and not match.           0x711         1809         0x88110711         ADSERR_DEVICE_SYMED.         Opice and not found.           0x711         1810         0x88110711         ADSERR_DEVICE_TRANSMODENTSUP.	0x705	1797	0x98110705		0 0 1	
	0x706	1798	0x98110706	<del> </del>	Invalid data values.	
0x700         8801         0x98110709         ADSERR_DEVICE_INVALIDCONTEXT         Invalid operating system context. This can result possible to resolve this through multitasking synchronization in the PLC.           0x70A         1802         0x9811070A         ADSERR_DEVICE_NOMEMORY         Insufficient memory.           0x70B         1803         0x9811070B         ADSERR_DEVICE_INVALIDPARM         Invalid parameter values.           0x70D         1805         0x9811070C         ADSERR_DEVICE_SYNTAX         Syntax error in file or command.           0x70E         1805         0x9811070F         ADSERR_DEVICE_SYNTAX         Syntax error in file or command.           0x70E         1808         0x9811070F         ADSERR_DEVICE_SYNTAX         Syntax error in file or command.           0x710         1807         0x9811070F         ADSERR_DEVICE_ENCENDATBILE         Objects do not match.           0x711         1808         0x98110711         ADSERR_DEVICE_EXISTS         Object afready exists.           0x712         1810         0x98110711         ADSERR_DEVICE_EXISTS         Object afready exists.           0x712         1810         0x98110714         ADSERR_DEVICE_INVALIDISTATE         Device (server) is in invalid state.           0x713         1811         0x98110715         ADSERR_DEVICE_INVALIDISTATE         Device (server) is in invalid. </td <td>0x707</td> <td>1799</td> <td>0x98110707</td> <td>ADSERR_DEVICE_NOTREADY</td> <td>Device is not ready to operate.</td>	0x707	1799	0x98110707	ADSERR_DEVICE_NOTREADY	Device is not ready to operate.	
	0x708	1800	0x98110708	ADSERR_DEVICE_BUSY	Device is busy.	
0x70B         1803         0x9811070B         ADSERR DEVICE_INVALIDPARM         Invalid parameter values.           0x70D         1804         0x9811070C         ADSERR DEVICE_NOTFOUND         Not found (files,),           0x70B         1806         0x9811070E         ADSERR DEVICE_SYNTAX         Syntax error in file or command.           0x70F         1807         0x9811070F         ADSERR_DEVICE_SYMBOLNOTFOUND         Objects do not match.           0x710         1808         0x98110711         ADSERR_DEVICE_SYMBOLNOTFOUND         Symbol not found.           0x711         1809         0x98110711         ADSERR_DEVICE_SYMBOLNOTFOUND         Invalid symbol version. This can occur due to an online change. Create a new handle.           0x712         1810         0x98110713         ADSERR_DEVICE_INVALIDSTATE         Device (server) is in invalid state.           0x713         1811         0x98110713         ADSERR_DEVICE_CLIENTUNKNOWD         Notification nadie is invalid.           0x714         1812         0x98110714         ADSERR_DEVICE_INVALIDWATCHSIZE         No further handle is invalid.           0x716         1814         0x98110717         ADSERR_DEVICE_INVALIDWATCHSIZE         No further handle available.           0x717         1815         0x98110718         ADSERR_DEVICE_INVALIDWATCHSIZE         No further handle available.	0x709		0x98110709	ADSERR_DEVICE_INVALIDCONTEXT	from use of ADS blocks in different tasks. It may be possible to resolve this through multitasking	
0x70C         1804         0x9811070C         ADSERR DEVICE NOTFOUND         Not found (files,)           0x70D         1805         0x9811070D         ADSERR DEVICE SYNTAX         Syntax eror in file or command.           0x70E         1807         0x9811070F         ADSERR DEVICE EXISTS         Object do not match.           0x701         1808         0x98110710         ADSERR DEVICE SYMBOLNOTFOUND         Symbol not found.           0x711         1809         0x98110712         ADSERR DEVICE SYMBOLVERSIONINVALD.         Invalid symbol not found.           0x711         1810         0x98110712         ADSERR DEVICE SYMBOLVERSIONINVALD.         Invalid symbol version. This can occur due to an online change. Create a new handle.           0x712         1810         0x98110713         ADSERR DEVICE SYMBOLVERSIONINVALD.         ADSERRE DEVICE SYMBOLVERSIONINVALD.         Notification client or tegistered.           0x714         1812         0x98110713         ADSERR DEVICE NOTIFIEYHDINVALID         Notification client to registered.           0x715         1813         0x98110718         ADSERR DEVICE CLIENTALINKNOWN         Notification size too large.           0x717         1815         0x98110718         ADSERR DEVICE INVALIDWATCHSIZE         Notification size too large.           0x718         1816         0x98110714         ADSERR DEVIC	0x70A		0x9811070A	ADSERR_DEVICE_NOMEMORY	,	
0x70D         1805         0x9811070D         ADSERR_DEVICE_SYNTAX         Syntax error in file or command.           0x70F         1807         0x9811070E         ADSERR_DEVICE_INCOMPATIBLE         Objects do not match.           0x701         1808         0x9811070F         ADSERR_DEVICE_SYSTS         Object sizes develop exists.           0x710         1808         0x98110710         ADSERR_DEVICE_SYMBOLNOTFOUND         Symbol not found.           0x711         1809         0x98110711         ADSERR_DEVICE_SYMBOLNERSIONINVALID         Invalid symbol version. This can occur due to an online change. Create a new handle.           0x712         1810         0x98110712         ADSERR_DEVICE_INVALIDSTATE         Device (server) is in invalid state.           0x713         1811         0x98110714         ADSERR_DEVICE_INTANSMODENTSUPP         AGSTRANGHO ent surpreded.           0x716         1813         0x98110715         ADSERR_DEVICE_CLENTUNKNOWN         Notification size too large.           0x716         1814         0x98110718         ADSERR_DEVICE_INVALIDWATCHSIZE         Notification size too large.           0x716         1815         0x98110718         ADSERR_DEVICE_INVALIDWATCHSIZE         Notification size too large.           0x716         1816         0x98110719         ADSERR_DEVICE_INVALIDWATCHSIZE         Notification size too larg	0x70B		0x9811070B	ADSERR_DEVICE_INVALIDPARM		
0x70E         1806         0x9811070E         ADSERR_DEVICE_INCOMPATIBLE         Objects do not match.           0x701         1807         0x9811070F         ADSERR_DEVICE_EXISTS         Object already exists.           0x710         1808         0x98110710         ADSERR_DEVICE_SYMBOLNOTFOUND         Symbol not found.           0x711         1809         0x98110711         ADSERR_DEVICE_INVALIDSTATE         Device (server) is in invalid state.           0x712         1810         0x98110713         ADSERR_DEVICE_TRANSMODENOTSUPP         ADSERR_DEVICE_NOTIFYHOLINVALID           0x713         1811         0x98110714         ADSERR_DEVICE_NOTIFYHOLINVALID         Notification nandle is invalid.           0x714         1812         0x98110715         ADSERR_DEVICE_NOMERHOLING         Notification nandle is invalid.           0x716         1814         0x98110715         ADSERR_DEVICE_NOMERHOLING         No further handle available.           0x717         1815         0x98110717         ADSERR_DEVICE_NOMERHOLING         No further handle available.           0x717         1816         0x98110718         ADSERR_DEVICE_NOMINER         No further handle available.           0x719         1817         0x98110718         ADSERR_DEVICE_NOMINER         No further handle available.           0x719         1821		-	0x9811070C		` ' '	
0x70F         1807         0x9811070F         ADSERR_DEVICE_EXISTS         Object already exists.           0x711         1809         0x98110710         ADSERR_DEVICE_SYMBOLVERSIONINVALID         Symbol not found.           0x712         1810         0x98110712         ADSERR_DEVICE_SYMBOLVERSIONINVALID         Invalid symbol version. This can occur due to an online change. Create a new handle.           0x713         1810         0x98110713         ADSERR_DEVICE_INVALIDSTATE         Device (server) is in invalid state.           0x713         1811         0x98110713         ADSERR_DEVICE_TANAMODENOTSUPP         AdsTransMode not supported.           0x716         1812         0x98110715         ADSERR_DEVICE_NOTIFYHNDINVALID         Notification handle is invalid.           0x716         1813         0x98110716         ADSERR_DEVICE_NOMOREHDLS         No further handle available.           0x716         1814         0x98110717         ADSERR_DEVICE_NOMOREHDLS         No further handle available.           0x716         1816         0x98110718         ADSERR_DEVICE_TIMEOUT         Device not initialized.           0x719         1817         0x98110714         ADSERR_DEVICE_TIMEOUT         Device has a timeout.           0x710         1820         0x98110714         ADSERR_DEVICE_INVALIDORID         Object ID is invalid.		-	0x9811070D		•	
0x710         1808         0x98110710         ADSERR DEVICE_SYMBOLNOTFOUND         Symbol not found.           0x711         1809         0x98110711         ADSERR_DEVICE_SYMBOLVERSIONINVALID.         Invalid symbol version. This can occur due to an online change. Create a new handle.           0x712         1810         0x98110712         ADSERR_DEVICE_INVALIDSTATE         Device (server) is in invalid state.           0x714         1812         0x98110714         ADSERR_DEVICE_CLIENTUNKNOWN         AdsTransMode not supported.           0x714         1813         0x98110715         ADSERR_DEVICE_CLIENTUNKNOWN         Notification client not registered.           0x716         1813         0x98110716         ADSERR_DEVICE_NOMOREHDLS         No further handle available.           0x717         1815         0x98110718         ADSERR_DEVICE_NOTINIT         Device not initialized.           0x718         1816         0x98110718         ADSERR_DEVICE_NOTINIT         Device has a timeout.           0x719         1817         0x98110719         ADSERR_DEVICE_INVALIDINTERFACE         Interface query failed.           0x710         1820         0x98110716         ADSERR_DEVICE_INVALIDORSID         Class ID is invalid.           0x710         1821         0x98110712         ADSERR_DEVICE_INVALIDARIA         Class ID is invalid.		-	0x9811070E	ADSERR_DEVICE_INCOMPATIBLE	,	
0x711         1809         0x98110711         ADSERR_DEVICE_SYMBOLVERSIONINVALID         Invalid symbol version. This can occur due to an online change. Create a new handle.           0x712         1810         0x98110712         ADSERR_DEVICE_INVALIDSTATE         Device (server) is in invalid state.           0x713         1811         0x98110714         ADSERR_DEVICE_INTANSMODENOTSUPP         AdSTransMode not supported.           0x714         1812         0x98110715         ADSERR_DEVICE_INTANSMODENOTSUPP         AdSTransMode not supported.           0x715         1813         0x98110716         ADSERR_DEVICE_CIENTUNKNOWN         Notification client not registered.           0x716         1814         0x98110716         ADSERR_DEVICE_NOMOREHDLS         No further handle available.           0x717         1815         0x98110711         ADSERR_DEVICE_INVALIDWATCHSIZE         Notification size too large.           0x718         1816         0x98110719         ADSERR_DEVICE_INVALIDWATCHSIZE         Notification size too large.           0x719         1817         0x98110719         ADSERR_DEVICE_INVALIDWATCHSIZE         Notification size too large.           0x710         1820         0x98110710         ADSERR_DEVICE_INVALIDATERFACE         Urong interface requested.           0x710         1821         0x98110710         ADSERR_DEVICE_INVALIDATERFACE <td></td> <td>_</td> <td>0x9811070F</td> <td></td> <td>Object already exists.</td>		_	0x9811070F		Object already exists.	
		1808	0x98110710	ADSERR_DEVICE_SYMBOLNOTFOUND		
0x713         1811         0x98110713         ADSERR_DEVICE_TRANSMODENOTSUPP         AdsTransMode not supported.           0x714         1812         0x98110714         ADSERR_DEVICE_CNOTIFYHNDINVALID         Notification client not registered.           0x716         1814         0x98110715         ADSERR_DEVICE_CIENTUNKNOWN         Notification client not registered.           0x716         1814         0x98110717         ADSERR_DEVICE_INVALIDWATCHSIZE         Not further handle available.           0x717         1815         0x98110718         ADSERR_DEVICE_INVALIDWATCHSIZE         Notification size too large.           0x718         1816         0x98110719         ADSERR_DEVICE_TONINIT         Device not initialized.           0x719         1817         0x98110719         ADSERR_DEVICE_INVALIDONITERFACE         Interface query failed.           0x710         1820         0x98110719         ADSERR_DEVICE_INVALIDONITERFACE         Wrong interface requested.           0x710         1821         0x98110710         ADSERR_DEVICE_INVALIDONITERFACE         Wrong interface requested.           0x710         1821         0x98110711         ADSERR_DEVICE_PRONING         Request pending.           0x711         1822         0x98110712         ADSERR_DEVICE_PRONING         Request pending.           0x720         1824<	0x711	1809	0x98110711	ADSERR_DEVICE_SYMBOLVERSIONINVALID		
0x714         1812         0x98110714         ADSERR_DEVICE_NOTIFYHNDINVALID         Notification handle is invalid.           0x715         1813         0x98110715         ADSERR_DEVICE_CLIENTUNKNOWN         Notification client not registered.           0x716         1814         0x98110716         ADSERR_DEVICE_NOMOREHDLS         No further handle available.           0x717         1815         0x98110718         ADSERR_DEVICE_INVALIDWATCHSIZE         Notification size too large.           0x718         1816         0x98110719         ADSERR_DEVICE_TIMEOUT         Device not initialized.           0x718         1818         0x98110719         ADSERR_DEVICE_TIMEOUT         Device has a timeout.           0x71A         1818         0x9811071B         ADSERR_DEVICE_INVALIDITERFACE         Interface query falled.           0x71B         1819         0x9811071D         ADSERR_DEVICE_INVALIDOSID         Class ID is invalid.           0x71D         1821         0x9811071D         ADSERR_DEVICE_INVALIDOSID         Object ID is invalid.           0x71F         1822         0x9811071E         ADSERR_DEVICE_NERNDING         Request pending.           0x720         1824         0x98110720         ADSERR_DEVICE_WARNING         Signal warning.           0x721         1825         0x98110722         ADSERR	0x712	1810	0x98110712	ADSERR_DEVICE_INVALIDSTATE	Device (server) is in invalid state.	
0x715         1813         0x98110715         ADSERR_DEVICE_CLIENTUNKNOWN         Notification client not registered.           0x716         1814         0x98110716         ADSERR_DEVICE_NOMOREHDLS         No further handle available.           0x717         1815         0x98110718         ADSERR_DEVICE_NOTINIT         Device not initialized.           0x718         1816         0x98110714         ADSERR_DEVICE_TIMEOUT         Device has a timeout.           0x714         1818         0x98110714         ADSERR_DEVICE_INVALIDINTERFACE         Interface query failed.           0x714         1818         0x98110714         ADSERR_DEVICE_INVALIDINTERFACE         Wrong interface requested.           0x716         1820         0x98110710         ADSERR_DEVICE_INVALIDOBUTE         Class ID is invalid.           0x710         1821         0x98110710         ADSERR_DEVICE_INVALIDOBUTE         Class ID is invalid.           0x711         1822         0x98110710         ADSERR_DEVICE_PENDING         Request pending.           0x711         1823         0x98110712         ADSERR_DEVICE_WARNING         Signal warning.           0x720         1824         0x98110720         ADSERR_DEVICE_WARNING         Signal warning.           0x721         1825         0x98110723         ADSERR_DEVICE_LICENSENOTFOUND <td>0x713</td> <td>1811</td> <td>0x98110713</td> <td>ADSERR_DEVICE_TRANSMODENOTSUPP</td> <td>AdsTransMode not supported.</td>	0x713	1811	0x98110713	ADSERR_DEVICE_TRANSMODENOTSUPP	AdsTransMode not supported.	
0x716         1814         0x98110716         ADSERR_DEVICE_NOMOREHDLS         No further handle available.           0x717         1815         0x98110717         ADSERR_DEVICE_INVALIDWATCHSIZE         Notification size too large.           0x718         1816         0x98110718         ADSERR_DEVICE_INVALIDWATCHSIZE         Notification size too large.           0x719         1817         0x98110719         ADSERR_DEVICE_TIMEOUT         Device has a timeout.           0x71A         1818         0x9811071A         ADSERR_DEVICE_INVALIDINTERFACE         Interface query failed.           0x71B         1820         0x9811071B         ADSERR_DEVICE_INVALIDINTERFACE         Wrong interface requested.           0x71D         1821         0x9811071D         ADSERR_DEVICE_INVALIDINTERFACE         Wrong interface requested.           0x71D         1821         0x9811071D         ADSERR_DEVICE_INVALIDORSID         Object ID is invalid.           0x71E         1822         0x9811071F         ADSERR_DEVICE_ABORTED         Request pending.           0x71E         1822         0x98110721         ADSERR_DEVICE_ABORTED         Request pending.           0x721         1825         0x98110722         ADSERR_DEVICE_ACCESSEDENTED         Request is aborted.           0x720         1824         0x98110722	0x714	1812	0x98110714		Notification handle is invalid.	
0x717         1815         0x98110717         ADSERR_DEVICE_INVALIDWATCHSIZE         Notification size too large.           0x718         1816         0x98110718         ADSERR_DEVICE_NOTINIT         Device not initialized.           0x719         1817         0x98110719         ADSERR_DEVICE_TIMEOUT         Device has a timeout.           0x71A         1818         0x98110714         ADSERR_DEVICE_NOINTERFACE         Interface query failed.           0x71D         1820         0x98110710         ADSERR_DEVICE_INVALIDOBJID         Class ID is invalid.           0x71D         1821         0x98110710         ADSERR_DEVICE_INVALIDOBJID         Object ID is invalid.           0x71E         1822         0x98110710         ADSERR_DEVICE_PENDING         Request pending.           0x71F         1823         0x98110712         ADSERR_DEVICE_WARNING         Signal warning.           0x721         1824         0x98110721         ADSERR_DEVICE_INVALIDARRAYIDX         Invalid array index.           0x722         1826         0x98110721         ADSERR_DEVICE_ACCESSDENIED         Access denied.           0x722         1826         0x98110723         ADSERR_DEVICE_LICENSEXPIRED         License expired.           0x725         1827         0x98110725         ADSERR_DEVICE_LICENSEXPIRED         License	0x715	1813	0x98110715	ADSERR_DEVICE_CLIENTUNKNOWN	Notification client not registered.	
0x718         1816         0x98110718         ADSERR_DEVICE_NOTINIT         Device not initialized.           0x719         1817         0x98110719         ADSERR_DEVICE_TIMEOUT         Device has a timeout.           0x71A         1818         0x98110718         ADSERR_DEVICE_INVALIDINTERFACE         Interface query failed.           0x71C         1820         0x9811071B         ADSERR_DEVICE_INVALIDINTERFACE         Wrong interface requested.           0x71D         1821         0x9811071C         ADSERR_DEVICE_INVALIDOBJID         Class ID is invalid.           0x71E         1822         0x9811071E         ADSERR_DEVICE_INVALIDOBJID         Object ID is invalid.           0x71E         1822         0x9811071E         ADSERR_DEVICE_PRDING         Request pending.           0x71E         1823         0x9811071F         ADSERR_DEVICE_MARNING         Signal warning.           0x721         1824         0x98110721         ADSERR_DEVICE_INVALIDARRAYIDX         Invalid array index.           0x722         1826         0x98110722         ADSERR_DEVICE_ACCESSDENIED         Access denied.           0x723         1827         0x98110723         ADSERR_DEVICE_LICENSENTOTOUND         Missing license.           0x725         1829         0x98110725         ADSERR_DEVICE_LICENSENTOTOUND         M	0x716	1814	0x98110716	ADSERR_DEVICE_NOMOREHDLS	No further handle available.	
0x719         1817         0x98110719         ADSERR_DEVICE_TIMEOUT         Device has a timeout.           0x71A         1818         0x9811071A         ADSERR_DEVICE_NOINTERFACE         Interface query failed.           0x71C         1820         0x9811071B         ADSERR_DEVICE_INVALIDINTERFACE         Wrong interface requested.           0x71D         1821         0x9811071D         ADSERR_DEVICE_INVALIDCISID         Class ID is invalid.           0x71E         1822         0x9811071D         ADSERR_DEVICE_INVALIDCISID         Object ID is invalid.           0x71E         1822         0x9811071D         ADSERR_DEVICE_PENDING         Request pending.           0x71E         1823         0x9811071F         ADSERR_DEVICE_BADRTED         Request pending.           0x721         1825         0x98110721         ADSERR_DEVICE_WARNING         Signal warning.           0x721         1825         0x98110722         ADSERR_DEVICE_INCE_INCALIDARRAYIDX         Invalid array index.           0x722         1826         0x98110723         ADSERR_DEVICE_LICENSENOTFOUND         Missing license.           0x723         1827         0x98110724         ADSERR_DEVICE_LICENSEXPIRED         License expired.           0x725         1830         0x98110725         ADSERR_DEVICE_LICENSEXPIRED         Licens		1815	0x98110717	ADSERR_DEVICE_INVALIDWATCHSIZE	Notification size too large.	
0x71A         1818         0x9811071A         ADSERR_DEVICE_NOINTERFACE         Interface query failed.           0x71B         1819         0x9811071B         ADSERR_DEVICE_INVALIDINTERFACE         Wrong interface requested.           0x71C         1820         0x9811071C         ADSERR_DEVICE_INVALIDING         Class ID is invalid.           0x71D         1821         0x9811071D         ADSERR_DEVICE_PENDING         Request pending.           0x71F         1822         0x9811071F         ADSERR_DEVICE_BABORTED         Request is aborted.           0x720         1824         0x98110721         ADSERR_DEVICE_WARNING         Signal warning.           0x721         1825         0x98110721         ADSERR_DEVICE_INVALIDARRAYIDX         Invalid array index.           0x722         1826         0x98110722         ADSERR_DEVICE_SYMBOLNOTACTIVE         Symbol not active.           0x723         1827         0x98110723         ADSERR_DEVICE_ACCESSDENIED         Access denied.           0x724         1828         0x98110724         ADSERR_DEVICE_LICENSENOTFOUND         Missing license.           0x725         1829         0x98110725         ADSERR_DEVICE_LICENSENOTED         License exceeded.           0x721         1831         0x98110725         ADSERR_DEVICE_LICENSENOTIMELIMIT         Licens		1816	0x98110718	ADSERR_DEVICE_NOTINIT	Device not initialized.	
0x71B         1819         0x9811071B         ADSERR_DEVICE_INVALIDINTERFACE         Wrong interface requested.           0x71C         1820         0x9811071C         ADSERR_DEVICE_INVALIDCLSID         Class ID is invalid.           0x71D         1821         0x9811071D         ADSERR_DEVICE_INVALIDOBJID         Object ID is invalid.           0x71E         1822         0x9811071E         ADSERR_DEVICE_INVALIDOBJID         Object ID is invalid.           0x721         1823         0x9811071E         ADSERR_DEVICE_ABORTED         Request is aborted.           0x720         1824         0x98110721         ADSERR_DEVICE_WARNING         Signal warning.           0x721         1825         0x98110722         ADSERR_DEVICE_SYMBOLNOTACTIVE         Symbol not active.           0x722         1826         0x98110723         ADSERR_DEVICE_ACCESSDENIED         Access denied.           0x723         1827         0x98110723         ADSERR_DEVICE_LICENSEENOTFOUND         Missing license.           0x724         1828         0x98110723         ADSERR_DEVICE_LICENSEEXCEEDED         License expired.           0x725         1829         0x98110725         ADSERR_DEVICE_LICENSEINVALID         Invalid license.           0x726         1830         0x98110728         ADSERR_DEVICE_SICENSEINTIMELIMIT <t< td=""><td>0x719</td><td>1817</td><td>0x98110719</td><td>ADSERR_DEVICE_TIMEOUT</td><td>Device has a timeout.</td></t<>	0x719	1817	0x98110719	ADSERR_DEVICE_TIMEOUT	Device has a timeout.	
0x71C         1820         0x9811071C         ADSERR_DEVICE_INVALIDCLSID         Class ID is invalid.           0x71D         1821         0x9811071D         ADSERR_DEVICE_INVALIDOBJID         Object ID is invalid.           0x71E         1822         0x9811071E         ADSERR_DEVICE_PENDING         Request pending.           0x71F         1823         0x98110721         ADSERR_DEVICE_MARNING         Signal warning.           0x720         1824         0x98110721         ADSERR_DEVICE_INVALIDARRAYIDX         Invalid array index.           0x721         1825         0x98110722         ADSERR_DEVICE_SYMBOLNOTACTIVE         Symbol not active.           0x722         1826         0x98110722         ADSERR_DEVICE_ACCESSDENIED         Access denied.           0x723         1827         0x98110723         ADSERR_DEVICE_LICENSENOTFOUND         Missing license.           0x724         1828         0x98110724         ADSERR_DEVICE_LICENSEEXPIRED         License excreeded.           0x725         1829         0x98110725         ADSERR_DEVICE_LICENSEEXCEEDED         License exceeded.           0x720         1831         0x98110728         ADSERR_DEVICE_LICENSESYSTEMID         License problem: System ID is invalid.           0x721         1832         0x98110728         ADSERR_DEVICE_LICENSETOTIMELIMIT <td></td> <td>_</td> <td>0x9811071A</td> <td></td> <td></td>		_	0x9811071A			
0x71D         1821         0x9811071D         ADSERR_DEVICE_INVALIDOBJID         Object ID is invalid.           0x71E         1822         0x9811071E         ADSERR_DEVICE_PENDING         Request pending.           0x71F         1823         0x9811071F         ADSERR_DEVICE_ABORTED         Request is aborted.           0x720         1824         0x98110720         ADSERR_DEVICE_WARNING         Signal warning.           0x721         1825         0x98110721         ADSERR_DEVICE_INVALIDARRAYIDX         Invalid array index.           0x722         1826         0x988110722         ADSERR_DEVICE_SYMBOLNOTACTIVE         Symbol not active.           0x723         1827         0x98110723         ADSERR_DEVICE_LICENSENOTFOUND         Missing license.           0x724         1828         0x98110724         ADSERR_DEVICE_LICENSEEXPIRED         License expired.           0x725         1829         0x98110725         ADSERR_DEVICE_LICENSESYSTEMID         License expired.           0x727         1831         0x98110728         ADSERR_DEVICE_LICENSESYSTEMID         License problem: System ID is invalid.           0x729         1833         0x98110728         ADSERR_DEVICE_LICENSESYSTEMID         License not limited in time.           0x720         1833         0x98110729         ADSERR_DEVICE_LICENSETIMETO		-		<del></del>	,	
0x71E         1822         0x9811071E         ADSERR_DEVICE_PENDING         Request pending.           0x71F         1823         0x9811071F         ADSERR_DEVICE_ABORTED         Request is aborted.           0x720         1824         0x98110720         ADSERR_DEVICE_WARNING         Signal warning.           0x721         1825         0x98110721         ADSERR_DEVICE_INVALIDARRAYIDX         Invalid array index.           0x722         1826         0x98110722         ADSERR_DEVICE_SYMBOLNOTACTIVE         Symbol not active.           0x723         1827         0x98110723         ADSERR_DEVICE_ACCESSDENIED         Access denied.           0x724         1828         0x988110724         ADSERR_DEVICE_LICENSENOTFOUND         Missing license.           0x725         1829         0x98110725         ADSERR_DEVICE_LICENSEEXCEEDED         License exceeded.           0x727         1831         0x98110726         ADSERR_DEVICE_LICENSEINVALID         Invalid license.           0x728         1832         0x98110728         ADSERR_DEVICE_LICENSESYSTEMID         License problem: System ID is invalid.           0x729         1833         0x98110728         ADSERR_DEVICE_LICENSETIMETOLONG         License period too long.           0x720         1835         0x98110728         ADSERR_DEVICE_LICENSETIMETOLONG <td></td> <td>-</td> <td></td> <td></td> <td></td>		-				
0x71F         1823         0x9811071F         ADSERR_DEVICE_ABORTED         Request is aborted.           0x720         1824         0x98110720         ADSERR_DEVICE_WARNING         Signal warning.           0x721         1825         0x98110721         ADSERR_DEVICE_SYMBOLNOTACTIVE         Symbol not active.           0x722         1826         0x98110722         ADSERR_DEVICE_SYMBOLNOTACTIVE         Symbol not active.           0x723         1827         0x98110722         ADSERR_DEVICE_ACCESSDRIED         Access denied.           0x724         1828         0x98110724         ADSERR_DEVICE_LICENSENOTFOUND         Missing license.           0x725         1829         0x98110725         ADSERR_DEVICE_LICENSEEXPIRED         License expired.           0x726         1830         0x98110726         ADSERR_DEVICE_LICENSEINVALID         Invalid license.           0x727         1831         0x98110728         ADSERR_DEVICE_LICENSESYSTEMID         License problem: System ID is invalid.           0x728         1832         0x98110728         ADSERR_DEVICE_LICENSETUTELISMIT         License problem: time in the future.           0x729         1833         0x98110724         ADSERR_DEVICE_LICENSETUTELISMIT         License proid too long.           0x720         1835         0x98110725         ADSERR_DEVIC			1 1 1 1		,	
0x720         1824         0x98110720         ADSERR_DEVICE_WARNING         Signal warning.           0x721         1825         0x98110721         ADSERR_DEVICE_INVALIDARRAYIDX         Invalid array index.           0x722         1826         0x98110722         ADSERR_DEVICE_SYMBOLNOTACTIVE         Symbol not active.           0x723         1827         0x98110723         ADSERR_DEVICE_ACCESSDENIED         Access denied.           0x724         1828         0x98110724         ADSERR_DEVICE_LICENSENOTFOUND         Missing license.           0x725         1829         0x98110725         ADSERR_DEVICE_LICENSEEXPIRED         License expired.           0x726         1830         0x98110726         ADSERR_DEVICE_LICENSEEXCEEDED         License exceeded.           0x727         1831         0x98110728         ADSERR_DEVICE_LICENSESYSTEMID         License problem: System ID is invalid.           0x728         1832         0x98110728         ADSERR_DEVICE_LICENSENOTIMELIMIT         License not limited in time.           0x729         1833         0x98110724         ADSERR_DEVICE_LICENSETUTUREISSUE         Licensing problem: time in the future.           0x728         1835         0x98110724         ADSERR_DEVICE_LICENSEDUPLICATED         License period too long.           0x720         1836         0x98110725<						
0x721         1825         0x98110721         ADSERR_DEVICE_INVALIDARRAYIDX         Invalid array index.           0x722         1826         0x98110722         ADSERR_DEVICE_SYMBOLNOTACTIVE         Symbol not active.           0x723         1827         0x98110723         ADSERR_DEVICE_ACCESSDENIED         Access denied.           0x724         1828         0x98110724         ADSERR_DEVICE_LICENSEEXOFFOUND         Missing license.           0x725         1829         0x98110725         ADSERR_DEVICE_LICENSEEXPIRED         License exceeded.           0x726         1830         0x98110726         ADSERR_DEVICE_LICENSEEXCEEDED         License exceeded.           0x727         1831         0x98110726         ADSERR_DEVICE_LICENSEINVALID         Invalid license.           0x728         1832         0x98110728         ADSERR_DEVICE_LICENSESYSTEMID         License problem: System ID is invalid.           0x729         1833         0x98110729         ADSERR_DEVICE_LICENSETUTUREISSUE         Licensing problem: time in the future.           0x72A         1834         0x9811072A         ADSERR_DEVICE_LICENSETIMETOLONG         License period too long.           0x72B         1835         0x9811072B         ADSERR_DEVICE_LICENSEDUPLICATED         License file read twice.           0x72C         1836         0x9811						
0x722         1826         0x98110722         ADSERR_DEVICE_SYMBOLNOTACTIVE         Symbol not active.           0x723         1827         0x98110723         ADSERR_DEVICE_ACCESSDENIED         Access denied.           0x724         1828         0x98110724         ADSERR_DEVICE_LICENSENOTFOUND         Missing license.           0x725         1829         0x98110725         ADSERR_DEVICE_LICENSEEXPIRED         License expired.           0x726         1830         0x98110726         ADSERR_DEVICE_LICENSEEXCEEDED         License exceeded.           0x727         1831         0x98110727         ADSERR_DEVICE_LICENSEINVALID         Invalid license.           0x728         1832         0x98110728         ADSERR_DEVICE_LICENSESYSTEMID         License problem: System ID is invalid.           0x729         1833         0x98110729         ADSERR_DEVICE_LICENSESYSTEMID         License not limited in time.           0x729         1833         0x98110729         ADSERR_DEVICE_LICENSETUTUREISSUE         Licensing problem: time in the future.           0x729         1835         0x9811072B         ADSERR_DEVICE_LICENSETIMETOLONG         License period too long.           0x720         1836         0x9811072C         ADSERR_DEVICE_LICENSEDUPLICATED         License file read twice.           0x720         1831		-				
0x723         1827         0x98110723         ADSERR_DEVICE_ACCESSDENIED         Access denied.           0x724         1828         0x98110724         ADSERR_DEVICE_LICENSENOTFOUND         Missing license.           0x725         1829         0x98110725         ADSERR_DEVICE_LICENSEEXPIRED         License expired.           0x726         1830         0x98110726         ADSERR_DEVICE_LICENSEEXCEEDED         License exceeded.           0x727         1831         0x98110727         ADSERR_DEVICE_LICENSEINVALID         Invalid license.           0x728         1832         0x98110728         ADSERR_DEVICE_LICENSESYSTEMID         License problem: System ID is invalid.           0x729         1833         0x98110729         ADSERR_DEVICE_LICENSETUTUREISSUE         Licensing problem: time in the future.           0x72A         1834         0x9811072A         ADSERR_DEVICE_LICENSETIMETOLONG         License period too long.           0x72B         1835         0x9811072B         ADSERR_DEVICE_EXCEPTION         Exception at system startup.           0x72C         1836         0x9811072C         ADSERR_DEVICE_SIGNATUREINVALID         Invalid signature.           0x72E         1838         0x9811072E         ADSERR_DEVICE_CERTIFICATEINVALID         Invalid certificate.           0x72F         1839         0x981107						
0x724         1828         0x98110724         ADSERR_DEVICE_LICENSENOTFOUND         Missing license.           0x725         1829         0x98110725         ADSERR_DEVICE_LICENSEEXPIRED         License expired.           0x726         1830         0x98110726         ADSERR_DEVICE_LICENSEEXCEEDED         License exceeded.           0x727         1831         0x98110727         ADSERR_DEVICE_LICENSEINVALID         Invalid license.           0x728         1832         0x98110728         ADSERR_DEVICE_LICENSESYSTEMID         License problem: System ID is invalid.           0x729         1833         0x98110729         ADSERR_DEVICE_LICENSENOTIMELIMIT         License not limited in time.           0x72A         1834         0x9811072A         ADSERR_DEVICE_LICENSETUTREISSUE         Licensing problem: time in the future.           0x72B         1835         0x9811072B         ADSERR_DEVICE_LICENSETIMETOLONG         License period too long.           0x72C         1836         0x9811072C         ADSERR_DEVICE_EXCEPTION         Exception at system startup.           0x72D         1837         0x9811072D         ADSERR_DEVICE_LICENSEDUPLICATED         License file read twice.           0x72F         1838         0x9811072F         ADSERR_DEVICE_SIGNATUREINVALID         Invalid signature.           0x72F         1839 <td></td> <td>-</td> <td></td> <td></td> <td></td>		-				
0x725         1829         0x98110725         ADSERR_DEVICE_LICENSEEXPIRED         License expired.           0x726         1830         0x98110726         ADSERR_DEVICE_LICENSEEXCEEDED         License exceeded.           0x727         1831         0x98110727         ADSERR_DEVICE_LICENSEINVALID         Invalid license.           0x728         1832         0x98110728         ADSERR_DEVICE_LICENSESYSTEMID         License problem: System ID is invalid.           0x729         1833         0x98110729         ADSERR_DEVICE_LICENSETOTUREISSUE         License not limited in time.           0x72A         1834         0x9811072A         ADSERR_DEVICE_LICENSETIMETOLONG         License period too long.           0x72B         1835         0x9811072B         ADSERR_DEVICE_EXCEPTION         Exception at system startup.           0x72C         1836         0x9811072C         ADSERR_DEVICE_LICENSEDUPLICATED         License file read twice.           0x72D         1837         0x9811072B         ADSERR_DEVICE_SIGNATUREINVALID         Invalid signature.           0x72F         1839         0x9811072F         ADSERR_DEVICE_CERTIFICATEINVALID         Invalid certificate.           0x730         1840         0x98110730         ADSERR_DEVICE_LICENSEOEMNOTFOUND         Public key not known from OEM.           0x731         1841 <td></td> <td></td> <td>-</td> <td></td> <td></td>			-			
0x72618300x98110726ADSERR_DEVICE_LICENSEEXCEEDEDLicense exceeded.0x72718310x98110727ADSERR_DEVICE_LICENSEINVALIDInvalid license.0x72818320x98110728ADSERR_DEVICE_LICENSESYSTEMIDLicense problem: System ID is invalid.0x72918330x98110729ADSERR_DEVICE_LICENSENOTIMELIMITLicense not limited in time.0x72A18340x9811072AADSERR_DEVICE_LICENSEFUTUREISSUELicensing problem: time in the future.0x72B18350x9811072BADSERR_DEVICE_LICENSETIMETOLONGLicense period too long.0x72C18360x9811072CADSERR_DEVICE_EXCEPTIONException at system startup.0x72D18370x9811072DADSERR_DEVICE_LICENSEDUPLICATEDLicense file read twice.0x72E18380x9811072EADSERR_DEVICE_SIGNATUREINVALIDInvalid signature.0x72F18390x9811072FADSERR_DEVICE_CERTIFICATEINVALIDInvalid certificate.0x73018400x98110730ADSERR_DEVICE_LICENSEOEMNOTFOUNDPublic key not known from OEM.0x73118410x98110731ADSERR_DEVICE_LICENSEDEMODENIEDDemo license prohibited.0x73318430x98110733ADSERR_DEVICE_INVALIDFNCIDInvalid function ID.0x73418440x98110734ADSERR_DEVICE_OUTOFRANGEOutside the valid range.0x73518450x98110735ADSERR_DEVICE_INVALIDALIGNMENTInvalid alignment.		-				
0x72718310x98110727ADSERR_DEVICE_LICENSEINVALIDInvalid license.0x72818320x98110728ADSERR_DEVICE_LICENSESYSTEMIDLicense problem: System ID is invalid.0x72918330x98110729ADSERR_DEVICE_LICENSENOTIMELIMITLicense not limited in time.0x72A18340x9811072AADSERR_DEVICE_LICENSEFUTUREISSUELicensing problem: time in the future.0x72B18350x9811072BADSERR_DEVICE_LICENSETIMETOLONGLicense period too long.0x72C18360x9811072CADSERR_DEVICE_EXCEPTIONException at system startup.0x72D18370x9811072DADSERR_DEVICE_LICENSEDUPLICATEDLicense file read twice.0x72E18380x9811072EADSERR_DEVICE_SIGNATUREINVALIDInvalid signature.0x72F18390x9811072FADSERR_DEVICE_CERTIFICATEINVALIDInvalid certificate.0x73018400x98110730ADSERR_DEVICE_LICENSEOEMNOTFOUNDPublic key not known from OEM.0x73118410x98110731ADSERR_DEVICE_LICENSERESTRICTEDLicense not valid for this system ID.0x73218420x98110732ADSERR_DEVICE_LICENSEDEMODENIEDDemo license prohibited.0x73318430x98110733ADSERR_DEVICE_INVALIDFNCIDInvalid function ID.0x73418440x98110734ADSERR_DEVICE_OUTOFRANGEOutside the valid range.0x73518450x98110735ADSERR_DEVICE_INVALIDALIGNMENTInvalid alignment.					·	
0x72818320x98110728ADSERR_DEVICE_LICENSESYSTEMIDLicense problem: System ID is invalid.0x72918330x98110729ADSERR_DEVICE_LICENSENOTIMELIMITLicense not limited in time.0x72A18340x9811072AADSERR_DEVICE_LICENSETUTUREISSUELicensing problem: time in the future.0x72B18350x9811072BADSERR_DEVICE_LICENSETIMETOLONGLicense period too long.0x72C18360x9811072CADSERR_DEVICE_EXCEPTIONException at system startup.0x72D18370x9811072DADSERR_DEVICE_LICENSEDUPLICATEDLicense file read twice.0x72E18380x9811072EADSERR_DEVICE_SIGNATUREINVALIDInvalid signature.0x72F18390x9811072FADSERR_DEVICE_CERTIFICATEINVALIDInvalid certificate.0x73018400x98110730ADSERR_DEVICE_LICENSEDEMNOTFOUNDPublic key not known from OEM.0x73118410x98110731ADSERR_DEVICE_LICENSERESTRICTEDLicense not valid for this system ID.0x73218420x98110732ADSERR_DEVICE_LICENSEDEMODENIEDDemo license prohibited.0x73318430x98110733ADSERR_DEVICE_INVALIDFNCIDInvalid function ID.0x73418440x98110734ADSERR_DEVICE_OUTOFRANGEOutside the valid range.0x73518450x98110735ADSERR_DEVICE_INVALIDALIGNMENTInvalid alignment.		-	-			
0x72918330x98110729ADSERR_DEVICE_LICENSENOTIMELIMITLicense not limited in time.0x72A18340x9811072AADSERR_DEVICE_LICENSETUTUREISSUELicensing problem: time in the future.0x72B18350x9811072BADSERR_DEVICE_LICENSETIMETOLONGLicense period too long.0x72C18360x9811072CADSERR_DEVICE_EXCEPTIONException at system startup.0x72D18370x9811072DADSERR_DEVICE_LICENSEDUPLICATEDLicense file read twice.0x72E18380x9811072EADSERR_DEVICE_SIGNATUREINVALIDInvalid signature.0x73D18400x9811073DADSERR_DEVICE_CERTIFICATEINVALIDInvalid certificate.0x73118410x98110731ADSERR_DEVICE_LICENSEOEMNOTFOUNDPublic key not known from OEM.0x73218420x98110732ADSERR_DEVICE_LICENSERESTRICTEDLicense not valid for this system ID.0x73318430x98110733ADSERR_DEVICE_LICENSEDEMODENIEDDemo license prohibited.0x73418440x98110734ADSERR_DEVICE_INVALIDFNCIDInvalid function ID.0x73518450x98110735ADSERR_DEVICE_OUTOFRANGEOutside the valid range.0x73518450x98110735ADSERR_DEVICE_INVALIDALIGNMENTInvalid alignment.		_				
0x72A18340x9811072AADSERR_DEVICE_LICENSETUTUREISSUELicensing problem: time in the future.0x72B18350x9811072BADSERR_DEVICE_LICENSETIMETOLONGLicense period too long.0x72C18360x9811072CADSERR_DEVICE_EXCEPTIONException at system startup.0x72D18370x9811072DADSERR_DEVICE_LICENSEDUPLICATEDLicense file read twice.0x72E18380x9811072EADSERR_DEVICE_SIGNATUREINVALIDInvalid signature.0x72F18390x9811073FADSERR_DEVICE_CERTIFICATEINVALIDInvalid certificate.0x73018400x98110730ADSERR_DEVICE_LICENSEOEMNOTFOUNDPublic key not known from OEM.0x73118410x98110731ADSERR_DEVICE_LICENSEDEMODENIEDLicense not valid for this system ID.0x73218420x98110732ADSERR_DEVICE_LICENSEDEMODENIEDDemo license prohibited.0x73318430x98110733ADSERR_DEVICE_INVALIDFNCIDInvalid function ID.0x73418440x98110734ADSERR_DEVICE_OUTOFRANGEOutside the valid range.0x73518450x98110735ADSERR_DEVICE_INVALIDALIGNMENTInvalid alignment.		-	1	<del></del>		
0x72B18350x9811072BADSERR_DEVICE_LICENSETIMETOLONGLicense period too long.0x72C18360x9811072CADSERR_DEVICE_EXCEPTIONException at system startup.0x72D18370x9811072DADSERR_DEVICE_LICENSEDUPLICATEDLicense file read twice.0x72E18380x9811072EADSERR_DEVICE_SIGNATUREINVALIDInvalid signature.0x72F18390x9811072FADSERR_DEVICE_CERTIFICATEINVALIDInvalid certificate.0x73018400x98110730ADSERR_DEVICE_LICENSEOEMNOTFOUNDPublic key not known from OEM.0x73118410x98110731ADSERR_DEVICE_LICENSERESTRICTEDLicense not valid for this system ID.0x73218420x98110732ADSERR_DEVICE_LICENSEDEMODENIEDDemo license prohibited.0x73318430x98110733ADSERR_DEVICE_INVALIDFNCIDInvalid function ID.0x73418440x98110734ADSERR_DEVICE_OUTOFRANGEOutside the valid range.0x73518450x98110735ADSERR_DEVICE_INVALIDALIGNMENTInvalid alignment.						
0x72C18360x9811072CADSERR_DEVICE_EXCEPTIONException at system startup.0x72D18370x9811072DADSERR_DEVICE_LICENSEDUPLICATEDLicense file read twice.0x72E18380x9811072EADSERR_DEVICE_SIGNATUREINVALIDInvalid signature.0x72F18390x9811072FADSERR_DEVICE_CERTIFICATEINVALIDInvalid certificate.0x73018400x98110730ADSERR_DEVICE_LICENSEOEMNOTFOUNDPublic key not known from OEM.0x73118410x98110731ADSERR_DEVICE_LICENSERESTRICTEDLicense not valid for this system ID.0x73218420x98110732ADSERR_DEVICE_LICENSEDEMODENIEDDemo license prohibited.0x73318430x98110733ADSERR_DEVICE_INVALIDFNCIDInvalid function ID.0x73418440x98110734ADSERR_DEVICE_OUTOFRANGEOutside the valid range.0x73518450x98110735ADSERR_DEVICE_INVALIDALIGNMENTInvalid alignment.						
0x72D18370x9811072DADSERR_DEVICE_LICENSEDUPLICATEDLicense file read twice.0x72E18380x9811072EADSERR_DEVICE_SIGNATUREINVALIDInvalid signature.0x72F18390x9811072FADSERR_DEVICE_CERTIFICATEINVALIDInvalid certificate.0x73018400x98110730ADSERR_DEVICE_LICENSEOEMNOTFOUNDPublic key not known from OEM.0x73118410x98110731ADSERR_DEVICE_LICENSERESTRICTEDLicense not valid for this system ID.0x73218420x98110732ADSERR_DEVICE_LICENSEDEMODENIEDDemo license prohibited.0x73318430x98110733ADSERR_DEVICE_INVALIDFNCIDInvalid function ID.0x73418440x98110734ADSERR_DEVICE_OUTOFRANGEOutside the valid range.0x73518450x98110735ADSERR_DEVICE_INVALIDALIGNMENTInvalid alignment.						
0x72E18380x9811072EADSERR_DEVICE_SIGNATUREINVALIDInvalid signature.0x72F18390x9811072FADSERR_DEVICE_CERTIFICATEINVALIDInvalid certificate.0x73018400x98110730ADSERR_DEVICE_LICENSEOEMNOTFOUNDPublic key not known from OEM.0x73118410x98110731ADSERR_DEVICE_LICENSERESTRICTEDLicense not valid for this system ID.0x73218420x98110732ADSERR_DEVICE_LICENSEDEMODENIEDDemo license prohibited.0x73318430x98110733ADSERR_DEVICE_INVALIDFNCIDInvalid function ID.0x73418440x98110734ADSERR_DEVICE_OUTOFRANGEOutside the valid range.0x73518450x98110735ADSERR_DEVICE_INVALIDALIGNMENTInvalid alignment.		<b>-</b>	-			
0x72F18390x9811072FADSERR_DEVICE_CERTIFICATEINVALIDInvalid certificate.0x73018400x98110730ADSERR_DEVICE_LICENSEOEMNOTFOUNDPublic key not known from OEM.0x73118410x98110731ADSERR_DEVICE_LICENSERESTRICTEDLicense not valid for this system ID.0x73218420x98110732ADSERR_DEVICE_LICENSEDEMODENIEDDemo license prohibited.0x73318430x98110733ADSERR_DEVICE_INVALIDFNCIDInvalid function ID.0x73418440x98110734ADSERR_DEVICE_OUTOFRANGEOutside the valid range.0x73518450x98110735ADSERR_DEVICE_INVALIDALIGNMENTInvalid alignment.		-				
0x73018400x98110730ADSERR_DEVICE_LICENSEOEMNOTFOUNDPublic key not known from OEM.0x73118410x98110731ADSERR_DEVICE_LICENSERESTRICTEDLicense not valid for this system ID.0x73218420x98110732ADSERR_DEVICE_LICENSEDEMODENIEDDemo license prohibited.0x73318430x98110733ADSERR_DEVICE_INVALIDFNCIDInvalid function ID.0x73418440x98110734ADSERR_DEVICE_OUTOFRANGEOutside the valid range.0x73518450x98110735ADSERR_DEVICE_INVALIDALIGNMENTInvalid alignment.						
0x73118410x98110731ADSERR_DEVICE_LICENSERESTRICTEDLicense not valid for this system ID.0x73218420x98110732ADSERR_DEVICE_LICENSEDEMODENIEDDemo license prohibited.0x73318430x98110733ADSERR_DEVICE_INVALIDFNCIDInvalid function ID.0x73418440x98110734ADSERR_DEVICE_OUTOFRANGEOutside the valid range.0x73518450x98110735ADSERR_DEVICE_INVALIDALIGNMENTInvalid alignment.						
0x732         1842         0x98110732         ADSERR_DEVICE_LICENSEDEMODENIED         Demo license prohibited.           0x733         1843         0x98110733         ADSERR_DEVICE_INVALIDENCID         Invalid function ID.           0x734         1844         0x98110734         ADSERR_DEVICE_OUTOFRANGE         Outside the valid range.           0x735         1845         0x98110735         ADSERR_DEVICE_INVALIDALIGNMENT         Invalid alignment.		_				
0x733         1843         0x98110733         ADSERR_DEVICE_INVALIDFNCID         Invalid function ID.           0x734         1844         0x98110734         ADSERR_DEVICE_OUTOFRANGE         Outside the valid range.           0x735         1845         0x98110735         ADSERR_DEVICE_INVALIDALIGNMENT         Invalid alignment.		-				
0x734     1844     0x98110734     ADSERR_DEVICE_OUTOFRANGE     Outside the valid range.       0x735     1845     0x98110735     ADSERR_DEVICE_INVALIDALIGNMENT     Invalid alignment.						
0x735 1845 0x98110735 ADSERR_DEVICE_INVALIDALIGNMENT Invalid alignment.	0x734	1844	0x98110734	ADSERR_DEVICE_OUTOFRANGE	Outside the valid range.	
0x736 1846 0x98110736 ADSERR_DEVICE_LICENSEPLATFORM Invalid platform level.	0x735	1845	0x98110735	ADSERR_DEVICE_INVALIDALIGNMENT	Invalid alignment.	
	0x736	1846	0x98110736	ADSERR_DEVICE_LICENSEPLATFORM	Invalid platform level.	



Hex	Dec	HRESULT	Name	Description	
0x737	1847	0x98110737	ADSERR_DEVICE_FORWARD_PL	Context – forward to passive level.	
0x738	1848	0x98110738	ADSERR_DEVICE_FORWARD_DL	Context – forward to dispatch level.	
0x739	1849	0x98110739	ADSERR_DEVICE_FORWARD_RT	Context – forward to real time.	
0x740	1856	0x98110740	ADSERR_CLIENT_ERROR	Client error.	
0x741	1857	0x98110741	ADSERR_CLIENT_INVALIDPARM	Service contains an invalid parameter.	
0x742	1858	0x98110742	ADSERR_CLIENT_LISTEMPTY	Polling list is empty.	
0x743	1859	0x98110743	ADSERR_CLIENT_VARUSED	Var connection already in use.	
0x744	1860	0x98110744	ADSERR_CLIENT_DUPLINVOKEID	The called ID is already in use.	
0x745	1861	0x98110745	ADSERR_CLIENT_SYNCTIMEOUT	Timeout has occurred – the remote terminal is not responding in the specified ADS timeout. The route setting of the remote terminal may be configured incorrectly.	
0x746	1862	0x98110746	ADSERR_CLIENT_W32ERROR	Error in Win32 subsystem.	
0x747	1863	0x98110747	ADSERR_CLIENT_TIMEOUTINVALID	Invalid client timeout value.	
0x748	1864	0x98110748	ADSERR_CLIENT_PORTNOTOPEN	Port not open.	
0x749	1865	0x98110749	ADSERR_CLIENT_NOAMSADDR	No AMS address.	
0x750	1872	0x98110750	ADSERR_CLIENT_SYNCINTERNAL	Internal error in Ads sync.	
0x751	1873	0x98110751	ADSERR_CLIENT_ADDHASH	Hash table overflow.	
0x752	1874	0x98110752	ADSERR_CLIENT_REMOVEHASH	Key not found in the table.	
0x753	1875	0x98110753	ADSERR_CLIENT_NOMORESYM	No symbols in the cache.	
0x754	1876	0x98110754	ADSERR_CLIENT_SYNCRESINVALID	Invalid response received.	
0x755	1877	0x98110755	ADSERR_CLIENT_SYNCPORTLOCKED	Sync Port is locked.	
0x756	1878	0x98110756	ADSERR_CLIENT_REQUESTCANCELLED	The request was cancelled.	

# **RTime error codes**

Hex	Dec	HRESULT	Name	Description	
0x1000	4096	0x98111000	RTERR_INTERNAL	Internal error in the real-time system.	
0x1001	4097	0x98111001	RTERR_BADTIMERPERIODS	Timer value is not valid.	
0x1002	4098	0x98111002	RTERR_INVALIDTASKPTR	Task pointer has the invalid value 0 (zero).	
0x1003	4099	0x98111003	RTERR_INVALIDSTACKPTR	Stack pointer has the invalid value 0 (zero).	
0x1004	4100	0x98111004	RTERR_PRIOEXISTS	The request task priority is already assigned.	
0x1005	4101	0x98111005	RTERR_NOMORETCB	No free TCB (Task Control Block) available. The maximum number of TCBs is 64.	
0x1006	4102	0x98111006	RTERR_NOMORESEMAS	No free semaphores available. The maximum number of semaphores is 64.	
0x1007	4103	0x98111007	RTERR_NOMOREQUEUES	No free space available in the queue. The maximum number of positions in the queue is 64.	
0x100D	4109	0x9811100D	RTERR_EXTIRQALREADYDEF	An external synchronization interrupt is already applied.	
0x100E	4110	0x9811100E	RTERR_EXTIRQNOTDEF	No external sync interrupt applied.	
0x100F	4111	0x9811100F	RTERR_EXTIRQINSTALLFAILED	Application of the external synchronization interrupt has failed.	
0x1010	4112	0x98111010	RTERR_IRQLNOTLESSOREQUAL	Call of a service function in the wrong context	
0x1017	4119	0x98111017	RTERR_VMXNOTSUPPORTED	Intel VT-x extension is not supported.	
0x1018	4120	0x98111018	RTERR_VMXDISABLED	Intel VT-x extension is not enabled in the BIOS.	
0x1019	4121	0x98111019	RTERR_VMXCONTROLSMISSING	Missing function in Intel VT-x extension.	
0x101A	4122	0x9811101A	RTERR_VMXENABLEFAILS	Activation of Intel VT-x fails.	

# Specific positive HRESULT Return Codes:

HRESULT	Name	Description
0x0000_0000	S_OK	No error.
0x0000_0001	S_FALSE	No error. Example: successful processing, but with a negative or incomplete result.
0x0000_0203	S_PENDING	No error. Example: successful processing, but no result is available yet.
0x0000_0256	S_WATCHDOG_TIMEOUT	No error. Example: successful processing, but a timeout occurred.

# **TCP Winsock error codes**



Hex	Dec	Name	Description	
0x274C	10060	WSAETIMEDOUT	A connection timeout has occurred - error while establishing the connection, because the remote terminal did not respond properly after a certain period of time, or the established connection could not be maintained because the connected host did not respond.	
0x274D	10061	WSAECONNREFUSED  Connection refused - no connection could be established becautarget computer has explicitly rejected it. This error usually resulatempt to connect to a service that is inactive on the external has a service for which no server application is running.		
0x2751	10065	WSAEHOSTUNREACH	No route to host - a socket operation referred to an unavailable host.	
More Winsock error codes: Win32 error codes				

More Information: www.beckhoff.com/tf6311

Beckhoff Automation GmbH & Co. KG Hülshorstweg 20 33415 Verl Germany Phone: +49 5246 9630 info@beckhoff.com www.beckhoff.com

