

# DNNF012

# DNNF020

TCP/IP- bzw. UDP-Einbindung von uniVision-Produkten in Steuerungen



Schnittstellenprotokoll

Inhaltsverzeichnis

1. Bestimmungsgemäße Verwendung ..... 3

2. Netzwerkübersicht ..... 4

3. Einstellungen in uniVision ..... 5

4. TIA-Beispielprogramm ..... 11

4.1 Prozessdaten vom Gerät TCP empfangen ..... 11

4.2 Prozessdaten vom Gerät UDP empfangen ..... 15

4.3 LIMA-Befehle über TCP/IP senden und LIMA-Antworten empfangen ..... 18

5. TwinCAT3-Beispielprogramme ..... 24

5.1 Prozessdaten vom Gerät TCP empfangen ..... 25

5.2 Prozessdaten vom Gerät UDP empfangen ..... 27

5.3 LIMA-Befehle über TCP/IP senden und LIMA-Antworten empfangen ..... 29

6. Rockwell-Beispielprogramme ..... 32

6.1 Prozessdaten vom Gerät TCP empfangen ..... 33

6.2 Prozessdaten vom Gerät UDP empfangen ..... 35

6.3 LIMA-Befehle über TCP/IP senden und LIMA-Antworten empfangen ..... 37

# 1. Bestimmungsgemäße Verwendung

Die Anleitung zeigt beispielhaft die Einbindung von uniVision-Produkten in verschiedenen Steuerungsumgebungen über die TCP/IP- bzw. die UDP-Schnittstelle. Sie ist ergänzend zu den Steuerungsbeispielprogrammen und zeigt u.a. welche Änderungen beispielsweise bei einer anderen Netzwerkkonfiguration oder bei einer anderen Anzahl an Zeichen, die per TCP/IP oder UDP übertragen werden, notwendig sind.

Folgende uniVision-Produkte können somit eingebunden werden:



Die Beispielprogramme sind für folgende Steuerungsumgebungen verfügbar:

- SPS S7-1200 von Siemens mit TIA Portal V15
- TwinCAT3 von Beckhoff
- SPS 1769-L18ERM-BB1B von Allen-Bradley mit Studio 5000 Logix Designer V32

Abhängig von der Steuerungsumgebung beinhaltet das Beispielprogramm einen unterschiedlichen Funktionsumfang. Allgemein sind folgende Funktionen in den Steuerungsbeispielprogrammen möglich:

- Prozessdaten vom Gerät TCP empfangen
- Prozessdaten vom Gerät UDP empfangen
- LIMA-Befehle (z.B. Triggerbefehle) über TCP/IP senden und die LIMA-Antworten empfangen

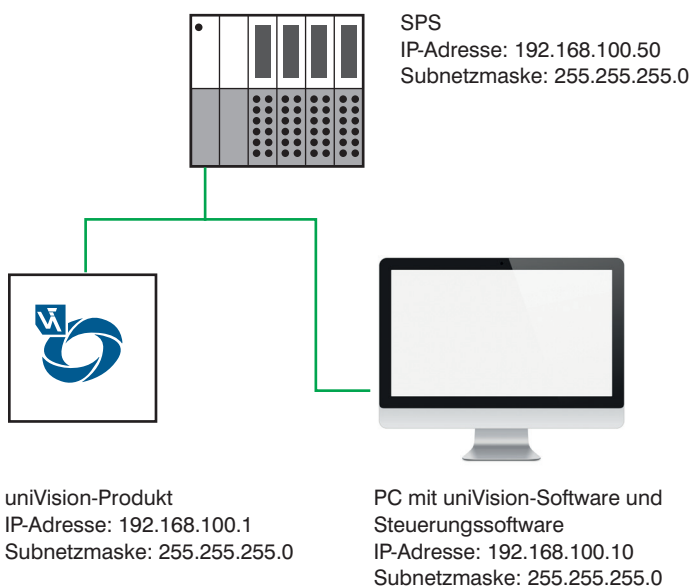


### HINWEIS!

Die Steuerungsbeispielprogramme werden ab der uniVision-Version 2.4.0 unterstützt.

## 2. Netzwerkübersicht

Das uniVision-Produkt, die SPS und der PC mit der uniVision-Software und der Steuerungssoftware müssen sich im selben Netz befinden. Im Beispielprogramm werden folgende Netzwerkeinstellungen verwendet.

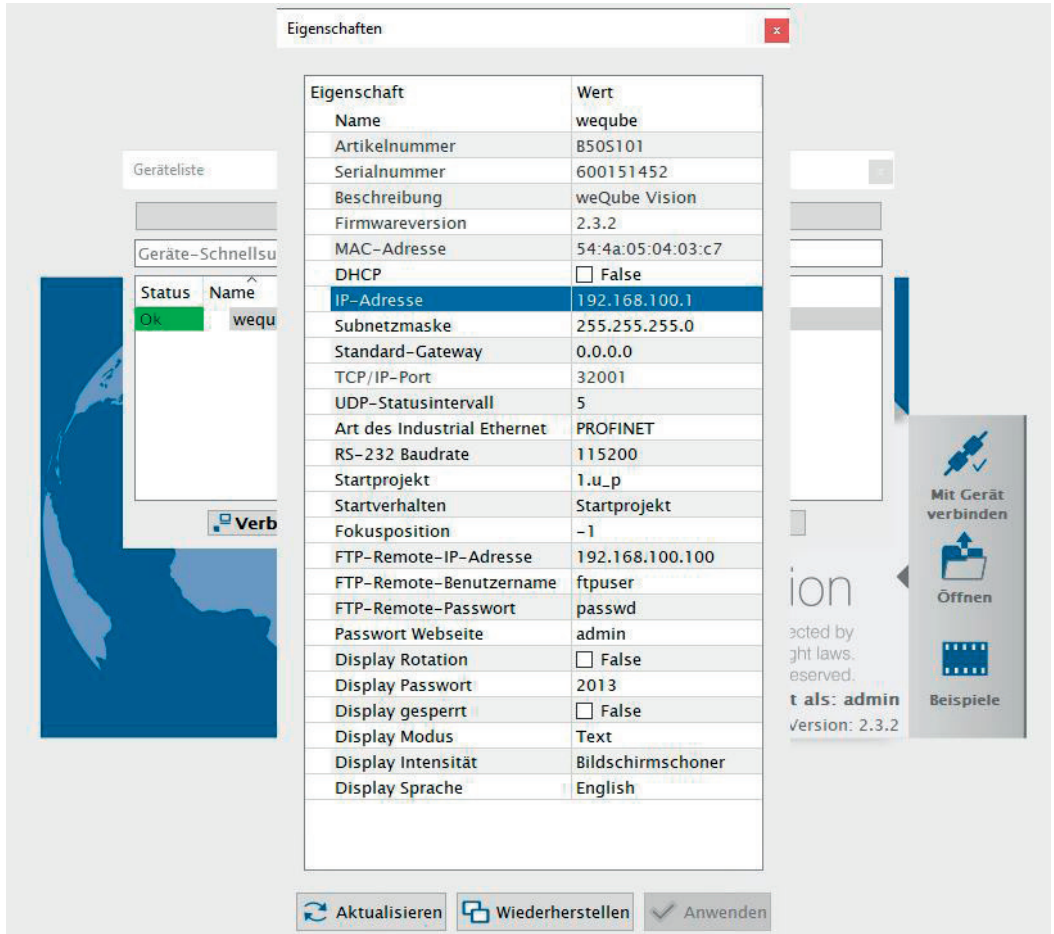




### 3. Einstellungen in uniVision

Folgende Schritte sind zunächst mit der uniVision-Software notwendig:

1. Installieren und öffnen Sie die uniVision-Software für Windows (Artikelnummer: DNNF020)
2. Richten Sie die Netzwerkkonfiguration des uniVision-Produkts über die uniVision-Software ein. Hierzu in der Geräteliste das uniVision-Produkt auswählen und auf Eigenschaften klicken.



The screenshot shows the 'Eigenschaften' (Properties) window in the uniVision software. The window displays a list of properties for a device named 'weqube'. The properties are organized into two columns: 'Eigenschaft' (Property) and 'Wert' (Value). The 'IP-Adresse' (IP Address) property is highlighted in blue.

Eigenschaft	Wert
Name	weqube
Artikelnummer	8505101
Serialnummer	600151452
Beschreibung	weQube Vision
Firmwareversion	2.3.2
MAC-Adresse	54:4a:05:04:03:c7
DHCP	<input type="checkbox"/> False
IP-Adresse	192.168.100.1
Subnetzmaske	255.255.255.0
Standard-Gateway	0.0.0.0
TCP/IP-Port	32001
UDP-Statusintervall	5
Art des Industrial Ethernet	PROFINET
RS-232 Baudrate	115200
Startprojekt	1.u_p
Startverhalten	Startprojekt
Fokusposition	-1
FTP-Remote-IP-Adresse	192.168.100.100
FTP-Remote-Benutzername	ftpuser
FTP-Remote-Passwort	passwd
Passwort Webseite	admin
Display Rotation	<input type="checkbox"/> False
Display Passwort	2013
Display gesperrt	<input type="checkbox"/> False
Display Modus	Text
Display Intensität	Bildschirmsschoner
Display Sprache	English

At the bottom of the window, there are three buttons: 'Aktualisieren' (Refresh), 'Wiederherstellen' (Restore), and 'Anwenden' (Apply).

3. Verbinden Sie sich über einen Doppelklick mit dem uniVision-Produkt und laden Sie ein Template auf dem Produkt.

4. Den Triggermodus auf Software bzw. Trigger stellen, um später über TCP/IP die LIMA-Schnittstelle zu nutzen und Triggerbefehle an das uniVision-Gerät zu schicken.

Projektbaum

Modul Applikation

Gerät Kamera

Modul Nachführung

Modul Region

Modul Schwellwert

Modul Zähler

Gerät E/A

TCP-IPGerät TCP

Modul hinzufügen

Assistent starten

Eigenschaft	Wert	
Beleuchtung intern	<input checked="" type="checkbox"/>	
Beleuchtung extern	<input type="checkbox"/>	
Eingangsbild rotieren	<input type="checkbox"/>	
HSV-Bild erstellen	<input checked="" type="checkbox"/>	
RGB-Bild erstellen	<input type="checkbox"/>	
RAW-Bild erstellen	<input type="checkbox"/>	
BGRA-Bild erstellen	<input type="checkbox"/>	
Belichtungszeit [us]	200	
Fokusposition [Schritte]	141	
Automatischer Fokus	<input type="checkbox"/>	
Beleuchtungsstrom [%]	20	
Beleuchtungsmodus	Blitzlicht	
Triggermodus	Trigger	

5. Um Prozessdaten über TCP/IP oder UDP zu verschicken, muss zudem Gerät TCP oder Gerät UDP im Projektbaum verfügbar sein und entsprechend konfiguriert werden.



**HINWEIS!**  
Im Template sind Gerät TCP und Gerät UDP zur Kommunikation mit der Steuerung bereits vorkonfiguriert. Alternativ kann auch ein neues Projekt erstellt werden und das Gerät TCP bzw. UDP manuell aus der Werkzeugleiste zum Projekt hinzugefügt werden.

4. Im Gerät TCP bzw. UDP können Zeichenanzahl, Präambel, Separator und Postambel beliebig eingerichtet werden. Zudem sollte der Ausgabemodus auf „Formatiert“ gestellt sein, um eine fixe Zeichenanzahl definieren zu können. Dies erleichtert das Auslesen der Prozessdaten auf der Steuerung.

**Projektbaum**

- Modul Applikation
  - Gerät Kamera
  - Modul Region
  - Modul Schwellwert
  - Modul Zähler
  - Gerät E/A
  - Gerät TCP**
  - Gerät UDP
  - Modul hinzufügen

Eigenschaft	Wert	
Prozesszeit [us]	1000	⚙
Modulstatus	0	⚙
Ausgang	+0027958,+0005748,+0016000,+0000000,+0035302,+0004300,+0001448,+0000000,+0006000;	
Präambel		⚙
Postambel	;	⚙
Seperator	,	⚙
Zeichenanzahl	9	⚙
Ausgabemodus	Formatiert	⚙
Fehlerbehandlung	Wertesetzung	⚙
Verbindungen	5	⚙
TCP-Port	32002	⚙
Sperrmodus	<input type="checkbox"/>	⚙

5. Ist der Ausgabemodus auf „Formatiert“ gestellt, so kann unter „Formatierungsoptionen“ die Zeichenanzahl für die verschiedenen Datentypen fix eingestellt werden.



**HINWEIS!**  
Im Beispiel werden für „Ganze Zahlen“ und „Fließkommazahlen“ insgesamt acht Zeichen verwendet (inkl. Vorzeichen und Komma). Für Ergebnisse des Datentyps Bool wird ein Zeichen verwendet.

Projektbaum

Modul Applikation

- Gerät Kamera
- Modul Region
- Modul Schwellwert
- Modul Zähler
- IO Gerät E/A
- Gerät TCP
  - Zeichenanzahl
  - Fehlerbehandlung
    - Formatierungsoptionen
      - Ganze Zahl
      - Fließkomma
      - Boolesch
- Gerät UDP
- Modul hinzufügen

Eigenschaft	Wert
Stellen vor Komma	4
Stellen nach Komma	2
Drucken +	<input checked="" type="checkbox"/>

6. Auch für den unter Fehlerbehandlung definierten Ersatzwert sollte die Zeichenanzahl passend gewählt werden. Im Beispiel werden ebenfalls acht Zeichen für den Fehler-Ersatzwert genutzt.

Projektbaum

Modul Applikation

Gerät Kamera

Modul Region

Modul Schwellwert

Modul Zähler

IO Gerät E/A

Gerät TCP

Zeichenanzahl

Fehlerbehandlung

Formatierungsoptionen

Ganze Zahl

Fließkomma

Boolesch

Gerät UDP

Modul hinzufügen

Eigenschaft

Wert

Ersetze *STRING*-Typen durch

Error###

uniVision Software

9

7. Unter „Ausgang“ im Gerät TCP bzw. UDP kann die gesamte Anzahl an Zeichen ermittelt werden, die über TCP bzw. UDP verschickt werden.

Projektbaum

Modul Applikation

Gerät Kamera

Modul Region

Modul Schwellwert

Modul Zähler

IO Gerät E/A

Gerät TCP

Gerät UDP

Modul hinzufügen

Eigenschaft	Wert
Prozesszeit [us]	1000
Modulstatus	0
Ausgang	+0027958,+0005748,+0016000,+0000000,+0035302,+0004300,+0001448,+0000000,+0006000;
Präambel	
Postambel	;
Seperator	,
Zeichenanzahl	9
Ausgabemodus	Formatiert
Fehlerbehandlung	Wertesetzung
Verbindungen	5
TCP-Port	32002
Sperrmodus	<input type="checkbox"/>

8. Das Projekt auf dem uniVision-Gerät abspeichern und in den Eigenschaften des Geräts als Startprojekt hinterlegen.

## 4. TIA-Beispielprogramm

Das TIA-Beispielprogramm ist erstellt mit einer SPS S7-1200 von Siemens mit TIA Portal V15. Es beinhaltet folgende Anwendungsfälle:

- Prozessdaten vom Gerät TCP empfangen
- Prozessdaten vom Gerät UDP empfangen
- LIMA-Befehle (z.B. Triggerbefehle) über TCP/IP senden und die LIMA-Antworten empfangen

### 4.1 Prozessdaten vom Gerät TCP empfangen

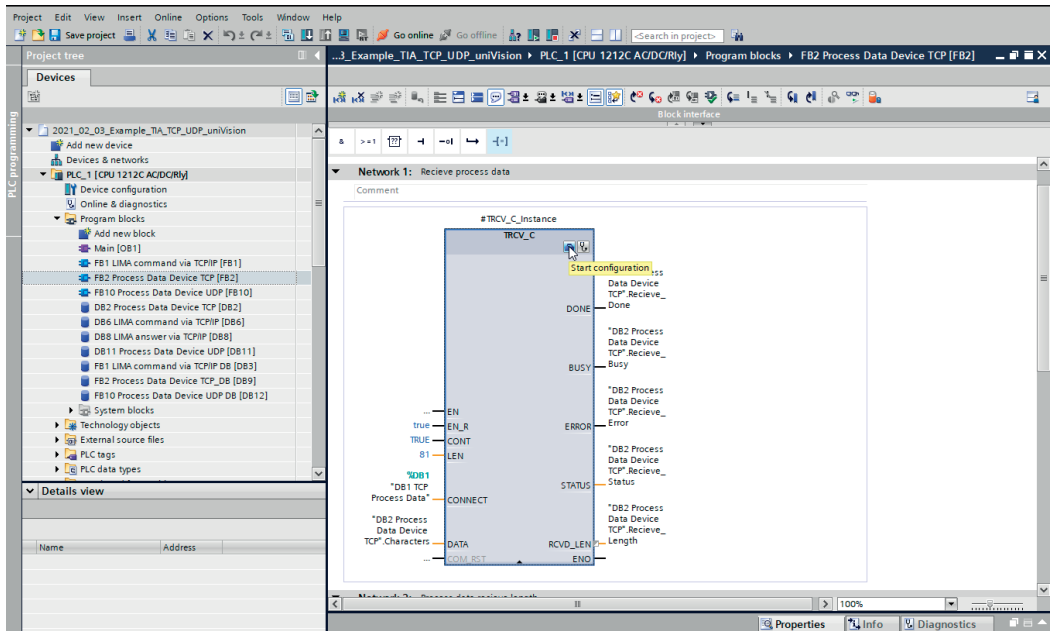
Das TIA-Beispielprogramm ist mit folgender Netzwerkeinstellung für das uniVision-Produkt erstellt:

- IP-Adresse: 192.168.100.1
- Subnetzmaske: 255.255.255.0

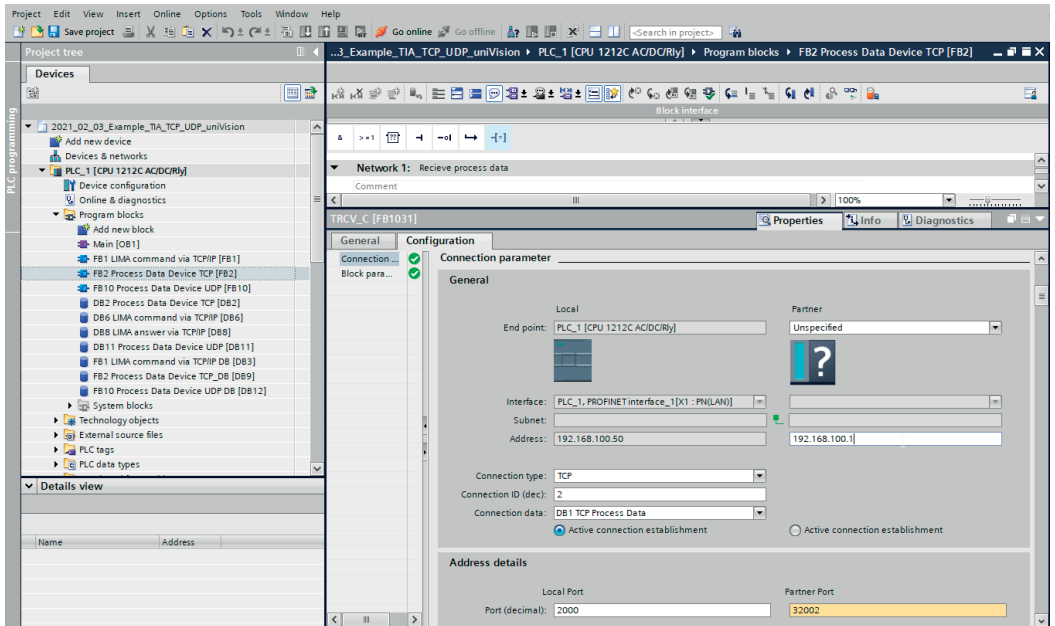
Die TCP-Prozessdaten werden standardmäßig über den Port 32002 verschickt.

Wird eine andere Netzwerkeinstellung oder ein anderer Port am uniVision-Produkt verwendet, so muss das Beispielprogramm entsprechend angepasst werden.

Hierfür den Funktionsbaustein „FB2 Process Data Device TCP“ öffnen und im Netzwerk 1 „Recieve process data“ auf „Start Configuration“ klicken.



Unter „Partner“ die IP-Adresse und den Port eingeben.



Das TIA-Beispielprogramm ist für Prozessdaten mit einer Länge von 81 Zeichen erstellt. Wird eine andere Zeichenanzahl benötigt, so muss das Beispielprogramm entsprechend angepasst werden.

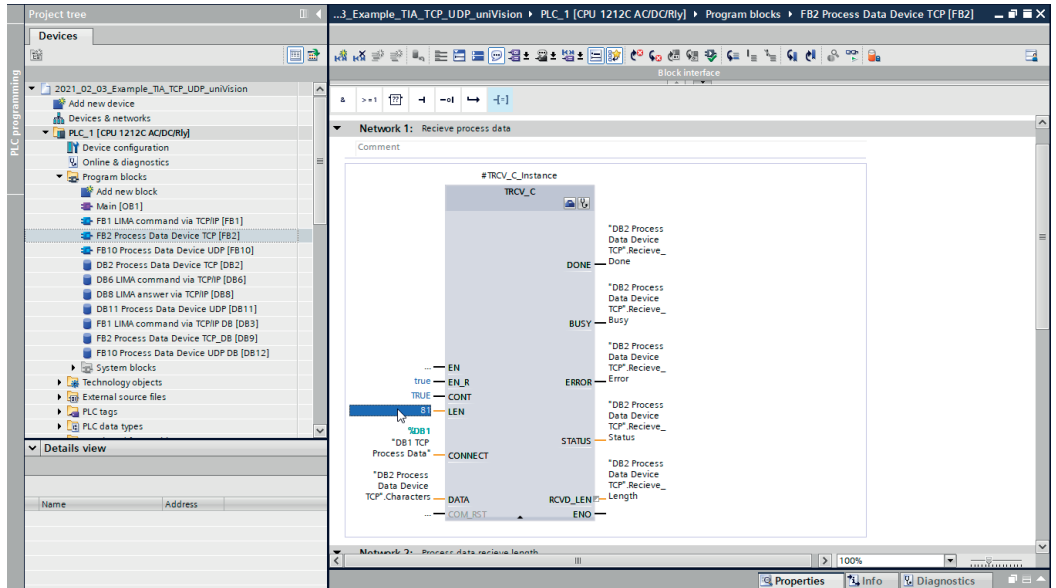
### HINWEIS!



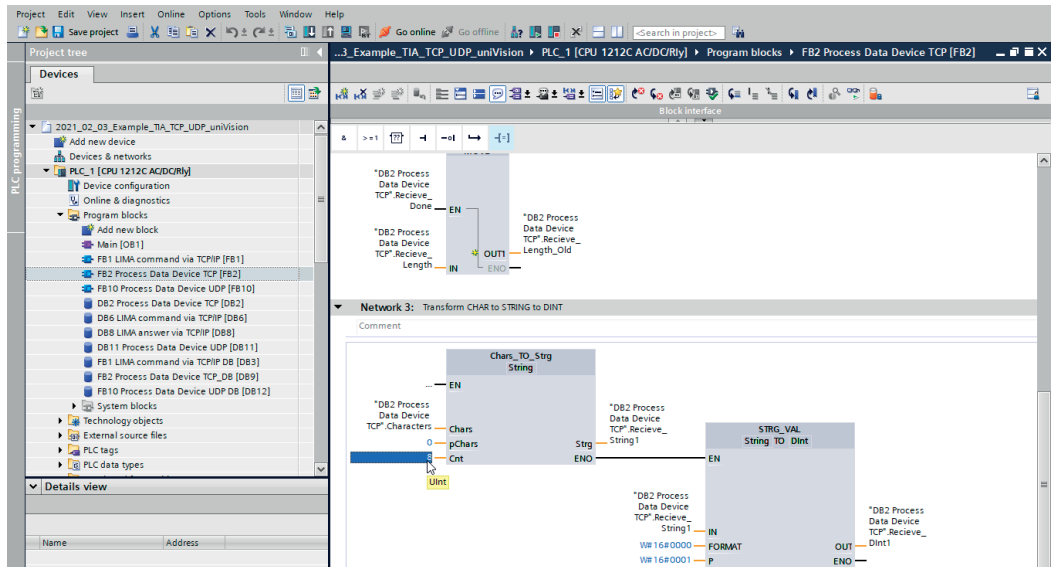
Die gesamte Anzahl an Zeichen, die als Prozessdaten über TCP verschickt werden, können in der uniVision-Software im Gerät TCP unter „Ausgang“ ermittelt werden. (siehe Kapitel 3. [Einstellungen in uniVision“ auf Seite 5](#)). Präambel, Separator und Postambel sowie Vorzeichen müssen bei der Zeichenanzahl mitgezählt werden!



Hierfür im Netzwerk 1 „Recieve process data“ die Zeichenanzahl unter „LEN“ anpassen.



Im Beispielprogramm ist für den ersten String auch die direkte Umwandlung der Zeichen in eine Zahl (DINT) enthalten. Die Anzahl an Zeichen bzw. der Datentyp für die erste Zahl können beliebig geändert werden.



Das Beispielprogramm kompilieren, auf die Steuerung laden und online verbinden.

Im Datenbank „DB2 Process Data Device TCP“ werden die Prozessdaten, die vom Gerät TCP verschickt werden, empfangen. Die Daten werden hierbei als einzelne Zeichen (Char) empfangen.

The screenshot shows the 'DB2 Process Data Device TCP' configuration in the TIA Portal. The 'Characters' array is set to 'Array[0..100]' and contains 29 Char elements. The 'Details view' shows the 'Characters' array with a data type of 'Array...' and 'Access...' set to 'True'.

Name	Offset	Data type	Access...
Characters		Array...	True
Receive_Done		Bool	True
Receive_Busy		Bool	True
Receive_Error		Bool	True

Für den ersten String wird die Umwandlung in einen anderen Datentyp beispielhaft am DINT gezeigt.

The screenshot shows the 'DB2 Process Data Device TCP' configuration in the TIA Portal. The 'Characters' array is set to 'Array[0..100] of Char'. The 'Details view' shows the 'Characters' array with a data type of 'Array[0..100] of Char' and 'Access...' set to 'True'. The 'Monitor value' column shows the first string '16#7D02' and the first DINT value '5081'.

Name	Data type	Start value	Monitor value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Co...
Static									
Characters	Array[0..100] of Char								
Receive_Done	Bool	false	FALSE						
Receive_Busy	Bool	false	TRUE						
Receive_Error	Bool	false	FALSE						
Receive_Status	Word	16#0	16#7D02						
Receive_Length	Int	0	0						
Receive_Length_Old	Int	0	81						
Receive_String1	String		"16#7D02"						
Receive_Dint1	Dint	0	5081						

## 4.2 Prozessdaten vom Gerät UDP empfangen

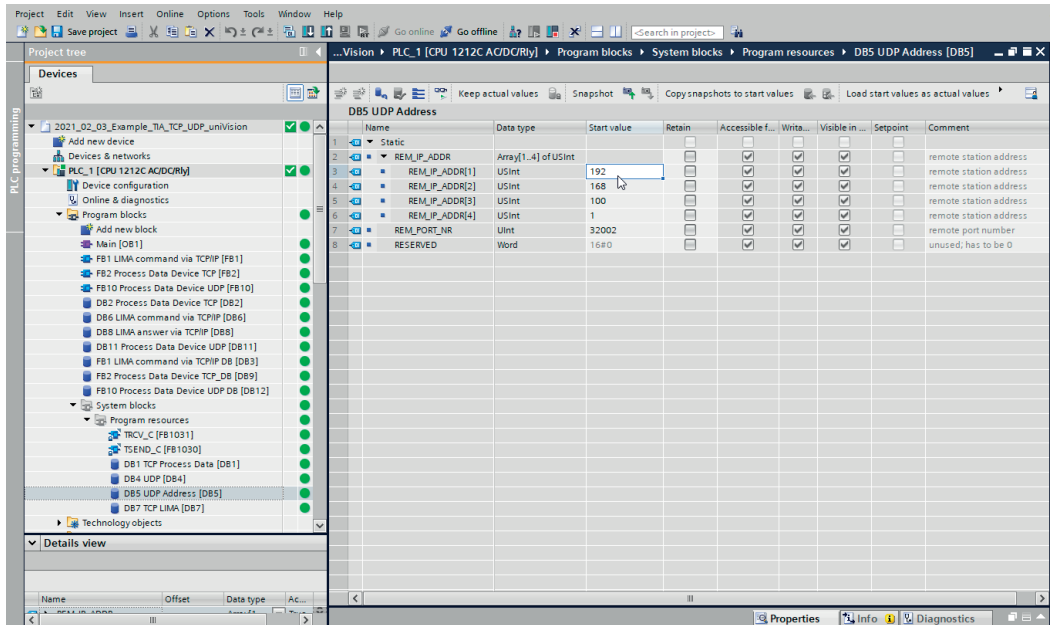
Das TIA-Beispielprogramm ist mit folgender Netzwerkeinstellung für das uniVision-Produkt erstellt:

- IP-Adresse: 192.168.100.1
- Subnetzmaske: 255.255.255.0

Die UDP-Prozessdaten werden über den Port 32002 verschickt.

Wird eine andere Netzwerkeinstellung am uniVision-Produkt verwendet, so muss das Beispielprogramm entsprechend angepasst werden.

Hierfür unter „System blocks“ den Datenblock „DB5 UDP Address“ öffnen und unter „REM\_IP\_ADDR“ die IP-Adresse des uniVision-Produkts eintragen.



Name	Data type	Start value	Retain	Accessible f...	Write...	Visible in ...	Setpoint	Comment
Static								
REM_IP_ADDR	Array[1..4] of USint							remote station address
REM_IP_ADDR[1]	USint	192						remote station address
REM_IP_ADDR[2]	USint	168						remote station address
REM_IP_ADDR[3]	USint	100						remote station address
REM_IP_ADDR[4]	USint	1						remote station address
REM_PORT_NR	UInt	32002						remote port number
RESERVED	Word	16#0						unused; has to be 0

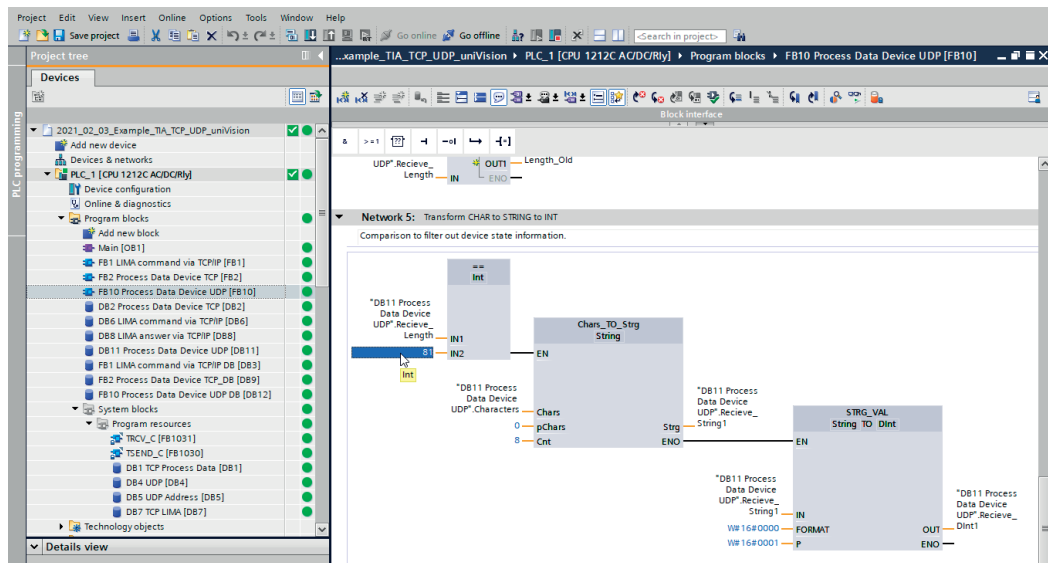
Das TIA-Beispielprogramm ist für Prozessdaten mit einer Länge von 81 Zeichen erstellt. Wird eine andere Zeichenanzahl benötigt, so muss das Beispielprogramm entsprechend angepasst werden.

### HINWEIS!

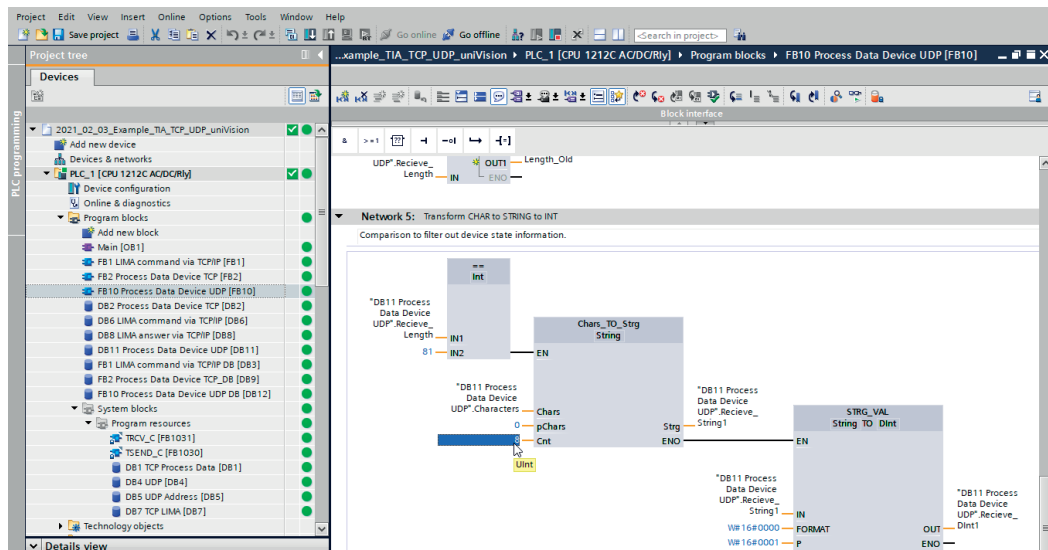


Die gesamte Anzahl an Zeichen, die als Prozessdaten über UDP verschickt werden, können in der uniVision-Software im Gerät UDP unter „Ausgang“ ermittelt werden. (siehe Kapitel 3. [Einstellungen in uniVision“ auf Seite 5](#)). Präambel, Separator und Postambel sowie Vorzeichen müssen bei der Zeichenanzahl mitgezählt werden!

Hierfür im Funktionsbaustein „FB10 Process Data Device UDP“ im Netzwerk 5 „Transform CHAR to STRING to INT“ die Zeichenanzahl unter „IN2“ anpassen.

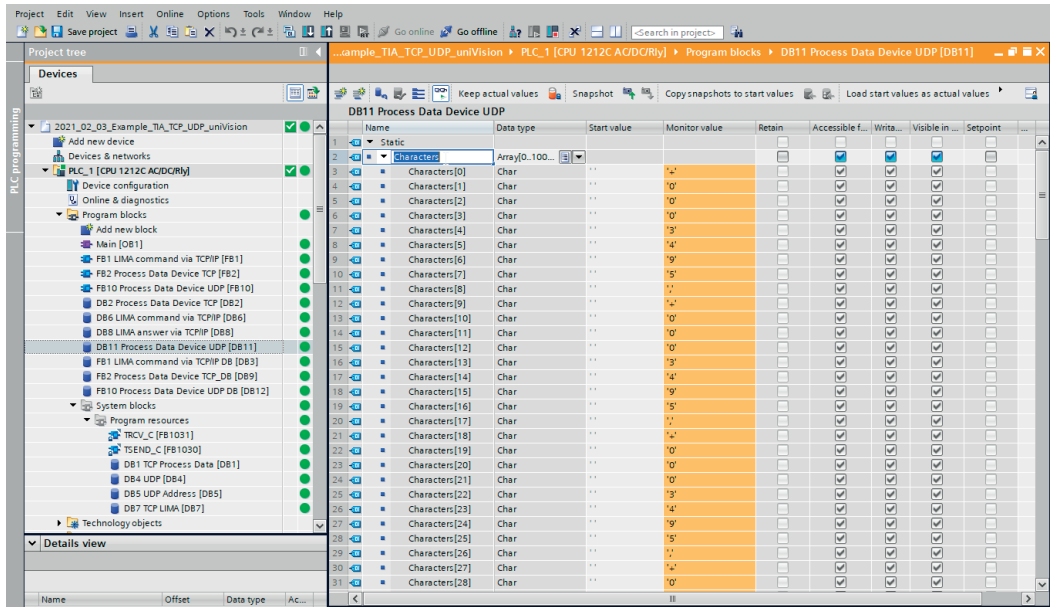


Im Beispielprogramm ist für den ersten String auch die direkte Umwandlung der Zeichen in eine Zahl (DINT) enthalten. Die Anzahl an Zeichen bzw. der Datentyp für die erste Zahl können beliebig geändert werden.



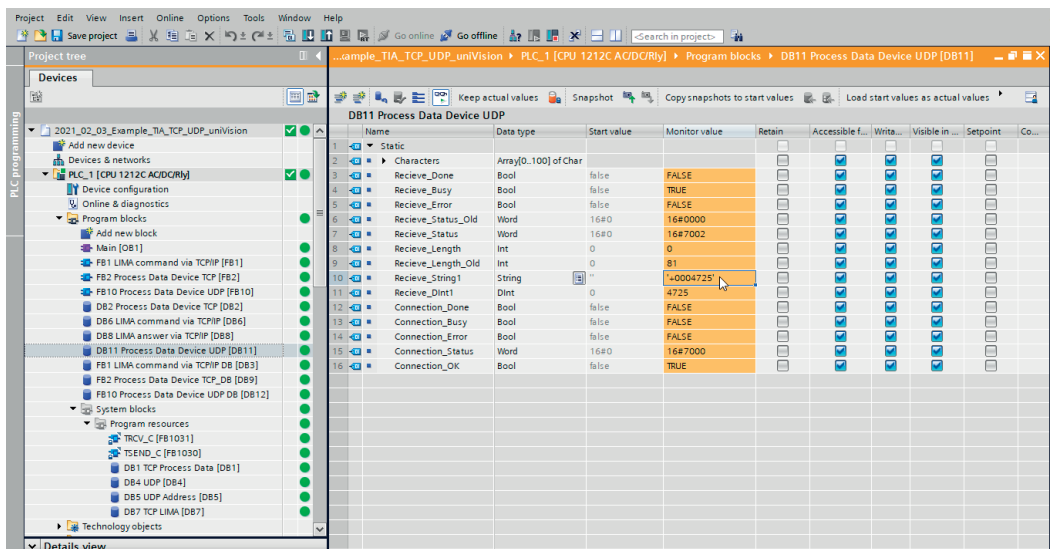
Das Beispielprogramm kompilieren, auf die Steuerung laden und online verbinden.

Im Datenblock „DB11 Process Data Device UDP“ werden die Prozessdaten, die vom Gerät UDP verschickt werden, empfangen. Die Daten werden hierbei als einzelne Zeichen (Char) empfangen.



Name	Data type	Start value	Monitor value	Retain	Accessible f...	Write...	Visible in ...	Setpoint
Static	Array[0..100]							
Characters[0]	Char	"						
Characters[1]	Char	"						
Characters[2]	Char	"						
Characters[3]	Char	"						
Characters[4]	Char	"						
Characters[5]	Char	"						
Characters[6]	Char	"						
Characters[7]	Char	"						
Characters[8]	Char	"						
Characters[9]	Char	"						
Characters[10]	Char	"						
Characters[11]	Char	"						
Characters[12]	Char	"						
Characters[13]	Char	"						
Characters[14]	Char	"						
Characters[15]	Char	"						
Characters[16]	Char	"						
Characters[17]	Char	"						
Characters[18]	Char	"						
Characters[19]	Char	"						
Characters[20]	Char	"						
Characters[21]	Char	"						
Characters[22]	Char	"						
Characters[23]	Char	"						
Characters[24]	Char	"						
Characters[25]	Char	"						
Characters[26]	Char	"						
Characters[27]	Char	"						
Characters[28]	Char	"						

Für den ersten String wird die Umwandlung in einen anderen Datentyp beispielhaft am DINT gezeigt.

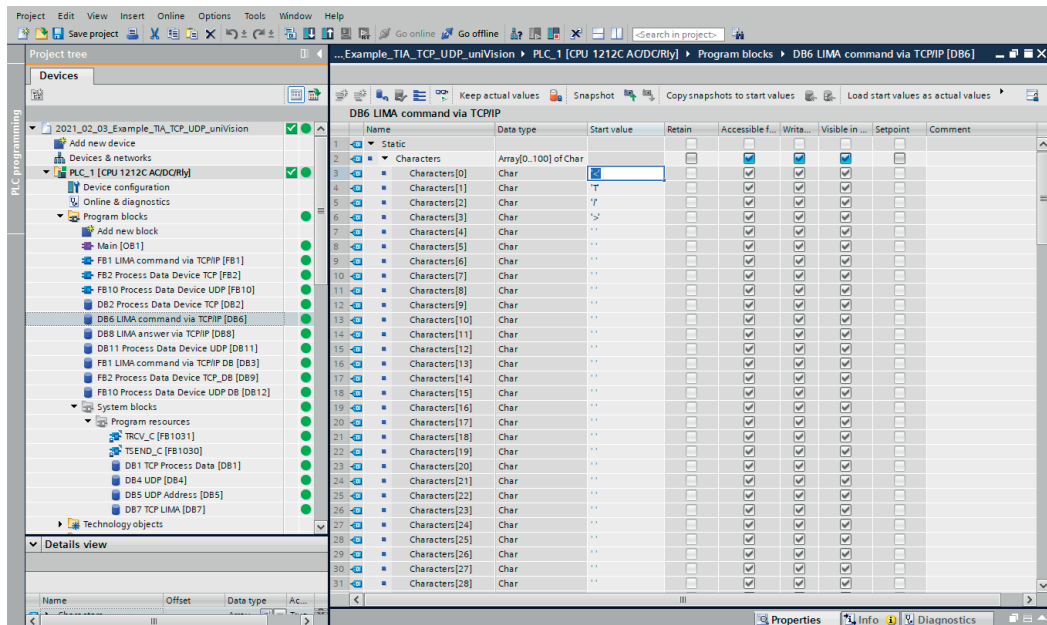


Name	Data type	Start value	Monitor value	Retain	Accessible f...	Write...	Visible in ...	Setpoint
Static	Array[0..100] of Char							
Receive_Done	Bool	false	FALSE					
Receive_Busy	Bool	false	TRUE					
Receive_Error	Bool	false	FALSE					
Receive_Status_Old	Word	16#0	16#0000					
Receive_Status	Word	16#0	16#7002					
Receive_Length	Int	0	0					
Receive_Length_Old	Int	0	81					
Receive_String1	String	"	"0004725"					
Receive_Dint1	Dint	0	4725					
Connection_Done	Bool	false	FALSE					
Connection_Busy	Bool	false	FALSE					
Connection_Error	Bool	false	FALSE					
Connection_Status	Word	16#0	16#7000					
Connection_OK	Bool	false	TRUE					

### 4.3 LIMA-Befehle über TCP/IP senden und LIMA-Antworten empfangen

Über die TCP/IP-Schnittstelle können LIMA-Befehle verschickt werden. Im Beispielprogramm wird ein Triggerbefehl an das uniVision-Produkt geschickt, der eine Bild- bzw. Profilausnahme auslöst. Details zu den verfügbaren Befehlen befinden sich im LIMA-Schnittstellenprotokoll. Es ist verfügbar im Downloadbereich der Produktdetailseite von uniVision (<https://www.wenglor.com/product/DNNF020>).

Der LIMA-Befehl muss hierfür mit einzelnen Zeichen unter „DB6 LIMA command via TCP/IP“ eingetragen werden. Für den Triggerbefehl muss hierfür <T/> geschickt werden.

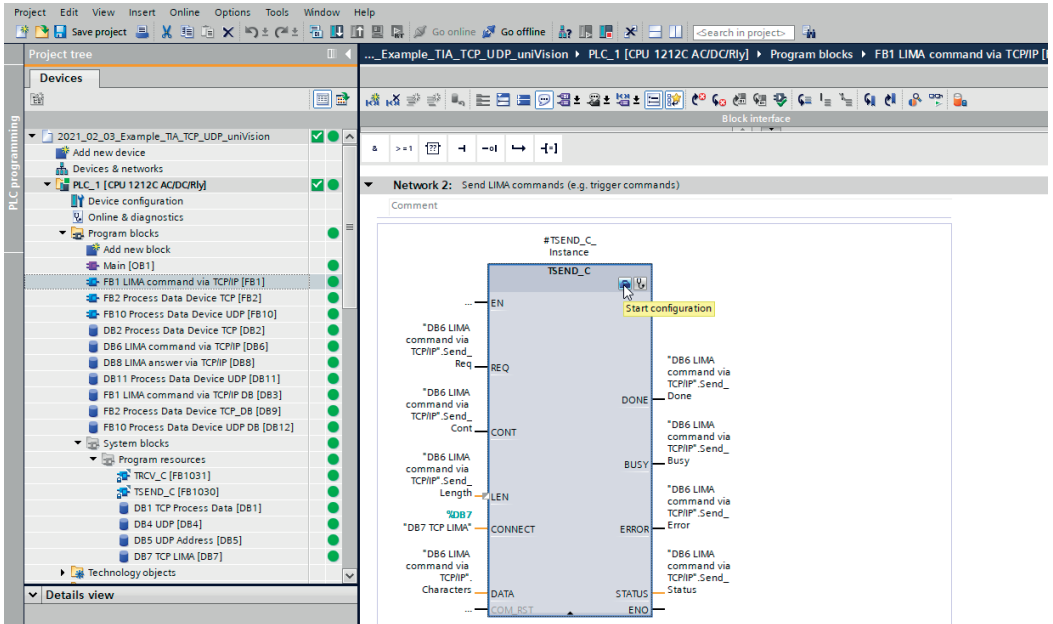


Das TIA-Beispielprogramm ist mit folgender Netzwerkeinstellung für das uniVision-Produkt erstellt:

- IP-Adresse: 192.168.100.1
  - Subnetzmaske: 255.255.255.0
- LIMA-Befehle werden über den Port 32001 verschickt.

Wird eine andere Netzwerkeinstellung am uniVision-Produkt verwendet, so muss das Beispielprogramm entsprechend angepasst werden.

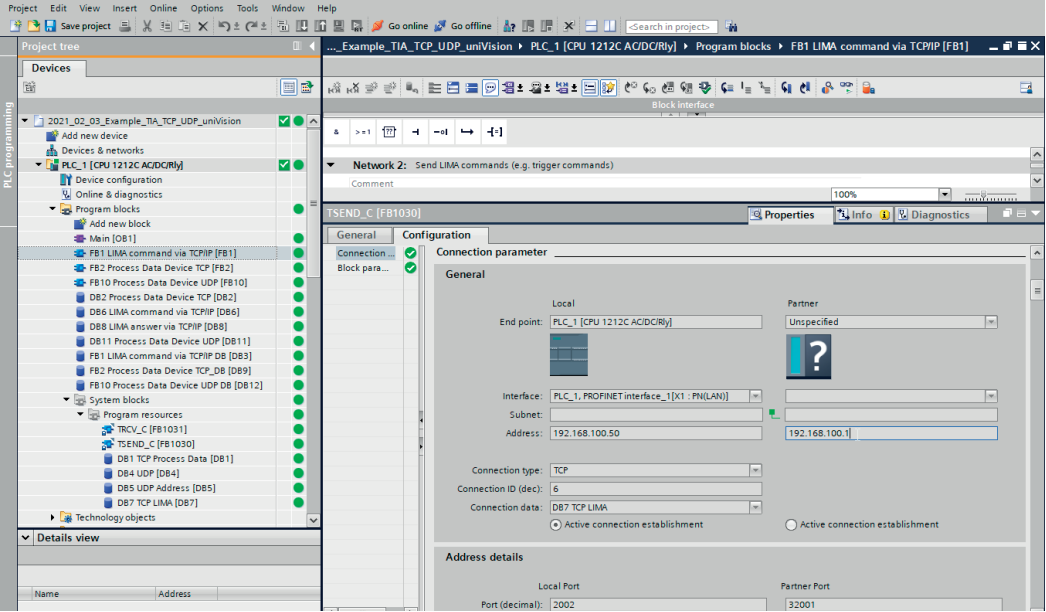
Hierfür den Funktionsbaustein „FB1 LIMA command via TCP/IP“ öffnen und im Netzwerk 2 „Send LIMA commands (e.g. trigger commands)“ auf „Start Configuration“ klicken.



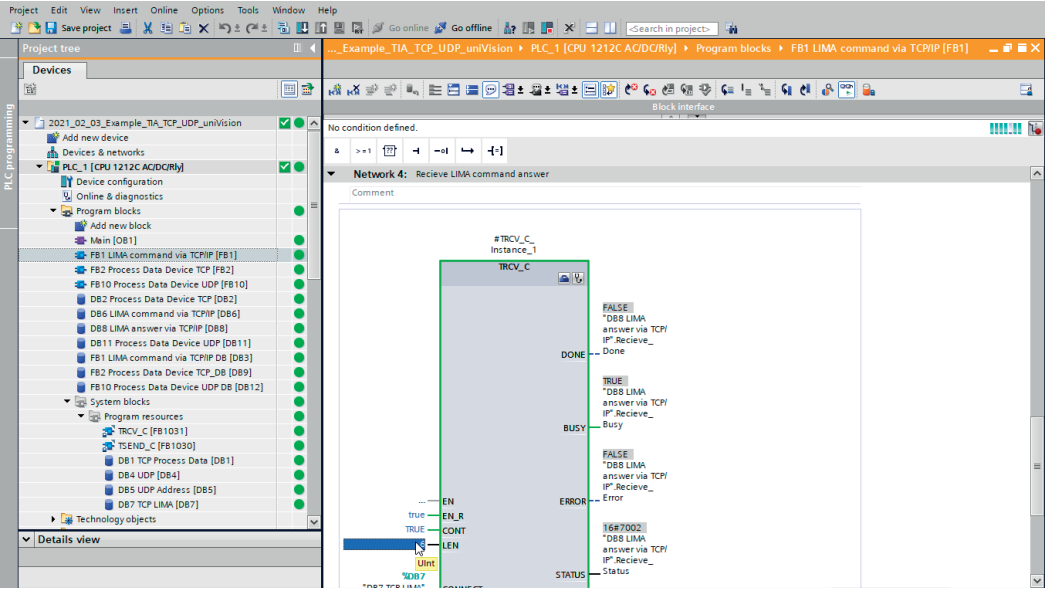
The screenshot displays the uniVision software interface. On the left, the 'Project tree' shows the project structure under '2021\_02\_03\_Example\_TIA\_TCP\_UDP\_uniVision'. The 'Program blocks' section is expanded, showing various function blocks. The 'FB1 LIMA command via TCP/IP' block is selected. On the right, the 'Network 2: Send LIMA commands (e.g. trigger commands)' is shown. The network contains a function block call for 'TSEND\_C' (Instance). The 'Start configuration' button is highlighted. The network diagram shows the following connections:

- EN** (Enable) input is connected to the 'REQ' (Request) input of the 'TSEND\_C' block.
- REQ** (Request) input is connected to the 'REQ' input of the 'TSEND\_C' block.
- CONT** (Continue) input is connected to the 'CONT' input of the 'TSEND\_C' block.
- LEN** (Length) input is connected to the 'LEN' input of the 'TSEND\_C' block.
- CONNECT** input is connected to the 'CONNECT' input of the 'TSEND\_C' block.
- DATA** (Data) input is connected to the 'DATA' input of the 'TSEND\_C' block.
- STATUS** (Status) output is connected to the 'STATUS' output of the 'TSEND\_C' block.
- ERROR** output is connected to the 'ERROR' output of the 'TSEND\_C' block.
- DONE** (Done) output is connected to the 'DONE' output of the 'TSEND\_C' block.
- BUSY** (Busy) output is connected to the 'BUSY' output of the 'TSEND\_C' block.
- ERROR** output is connected to the 'ERROR' output of the 'TSEND\_C' block.
- STATUS** output is connected to the 'STATUS' output of the 'TSEND\_C' block.

Unter „Partner“ die IP-Adresse und den Port 32001 eingeben.



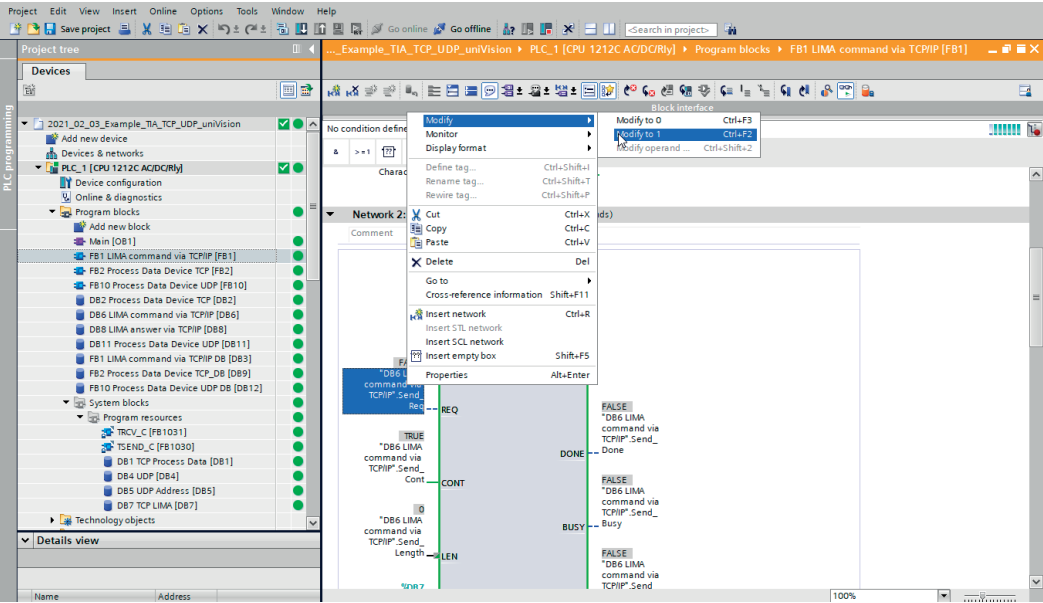
Analog dazu im Netzwerk 4 „Recieve LIMA command answer“ ebenfalls auf „Start Configuration“ klicken und erneut die IP-Adresse und den Port 32001 eingeben. Zusätzlich muss im Netzwerk 4 unter „LEN“ die Zeichenanzahl der LIMA-Antwort eingetragen werden. Die Antwort des Triggerbefehls beinhaltet 6 Zeichen (<TOK/>).







Der LIMA-Befehl wird durch das Setzen von REQ auf 1 an das uniVision-Gerät gesendet.

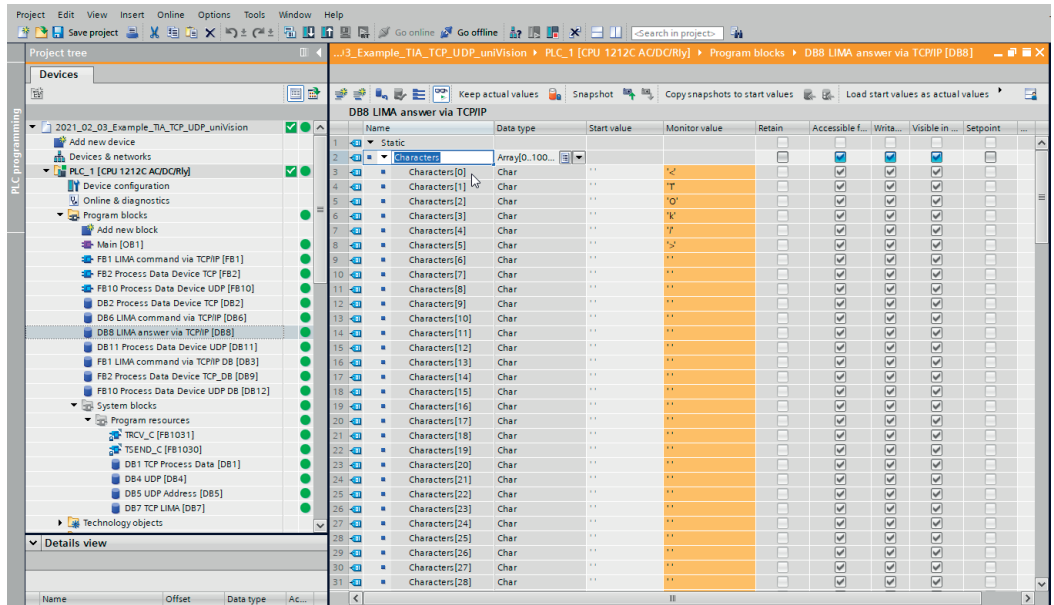


HINWEIS!



Der LIMA-Befehl wird im Beispielprogramm unmittelbar nach dem Senden wieder zurückgesetzt, damit nur ein Bild bzw. Profil vom uniVision-Produkt aufgenommen wird. Die zugehörigen Ergebnisse zum Trigger können über die Prozessdaten empfangen werden. Dabei kann beispielsweise über den Ausführezähler geprüft werden, wann die Ergebnisse verfügbar sind.

Die LIMA-Antwort kann im Datenblock „DB8 LIMA answer via TCP/IP“ empfangen werden. Für den Triggerbefehl wird <Tok/> vom uniVision-Produkt als Antwort für eine erfolgreiche Ausführung des Triggerbefehls verschickt.



The screenshot displays the SIMATIC Manager interface with the Ladder Logic editor open for the network 'DB8 LIMA answer via TCP/IP'. The left sidebar shows the project tree with the following structure:

- 2021\_02\_03\_Example\_TIA\_TCP\_UDP\_uniVision
  - Devices & networks
    - PLC\_1 [CPU 1212C AC/DC/Rly]
      - Device configuration
        - Online & diagnostics
          - Program blocks
            - Main [OB1]
              - FBI LIMA command via TCP/IP [FB1]
                - FBI Process Data Device TCP [FB2]
                  - FBI0 Process Data Device UDP [FB10]
                    - DB2 Process Data Device TCP [DB2]
                      - DB8 LIMA command via TCP/IP [DB8]
                        - DB11 Process Data Device UDP [DB11]
                          - FBI LIMA command via TCP/IP DB [DB3]
                            - FBI Process Data Device TCP DB [DB9]
                              - FBI0 Process Data Device UDP DB [DB12]
                                - System blocks
                                  - Program resources
                                    - TRCV\_C [FB1031]
                                      - TSEND\_C [FB1030]
  - DB1 TCP Process Data [DB1]
  - DB4 UDP [DB4]
  - DB5 UDP Address [DB5]
  - DB7 TCP LIMA [DB7]
  - Technology objects
  - Details view
  - Name
  - Offset
  - Data type
  - Ac...

The main editor shows a table of variables for 'DB8 LIMA answer via TCP/IP':

| Name           | Data type       | Start value | Monitor value | Retain | Accessible f... | Write... | Visible in ... | Setpoint |
|----------------|-----------------|-------------|---------------|--------|-----------------|----------|----------------|----------|
| Static         | Array[0..100... |             |               |        |                 |          |                |          |
| Characters[0]  | Char            |             | '2'           |        |                 |          |                |          |
| Characters[1]  | Char            |             | 'T'           |        |                 |          |                |          |
| Characters[2]  | Char            |             | '0'           |        |                 |          |                |          |
| Characters[3]  | Char            |             | 'K'           |        |                 |          |                |          |
| Characters[4]  | Char            |             | '7'           |        |                 |          |                |          |
| Characters[5]  | Char            |             | '2'           |        |                 |          |                |          |
| Characters[6]  | Char            |             |               |        |                 |          |                |          |
| Characters[7]  | Char            |             |               |        |                 |          |                |          |
| Characters[8]  | Char            |             |               |        |                 |          |                |          |
| Characters[9]  | Char            |             |               |        |                 |          |                |          |
| Characters[10] | Char            |             |               |        |                 |          |                |          |
| Characters[11] | Char            |             |               |        |                 |          |                |          |
| Characters[12] | Char            |             |               |        |                 |          |                |          |
| Characters[13] | Char            |             |               |        |                 |          |                |          |
| Characters[14] | Char            |             |               |        |                 |          |                |          |
| Characters[15] | Char            |             |               |        |                 |          |                |          |
| Characters[16] | Char            |             |               |        |                 |          |                |          |
| Characters[17] | Char            |             |               |        |                 |          |                |          |
| Characters[18] | Char            |             |               |        |                 |          |                |          |
| Characters[19] | Char            |             |               |        |                 |          |                |          |
| Characters[20] | Char            |             |               |        |                 |          |                |          |
| Characters[21] | Char            |             |               |        |                 |          |                |          |
| Characters[22] | Char            |             |               |        |                 |          |                |          |
| Characters[23] | Char            |             |               |        |                 |          |                |          |
| Characters[24] | Char            |             |               |        |                 |          |                |          |
| Characters[25] | Char            |             |               |        |                 |          |                |          |
| Characters[26] | Char            |             |               |        |                 |          |                |          |
| Characters[27] | Char            |             |               |        |                 |          |                |          |
| Characters[28] | Char            |             |               |        |                 |          |                |          |

## 5. TwinCAT3-Beispielprogramme

Die TwinCAT3-Beispielprogramme für UDP und TCP beinhalten folgende Anwendungsfälle:

- Prozessdaten vom Gerät TCP empfangen (im Beispielprogramm TCP)
- Prozessdaten vom Gerät UDP empfangen (im Beispielprogramm UDP)
- LIMA-Befehle (z.B. Triggerbefehle) über TCP/IP senden und die LIMA-Antwort empfangen (im Beispielprogramm TCP)

Beim Beispiel wird folgende Netzwerkkonfiguration verwendet:

- PC mit TwinCAT3:
  - IP-Adresse: 192.168.100.181
  - Subnetzmaske: 255.255.255.0
- uniVision-Produkt:
  - IP-Adresse: 192.168.100.1
  - Subnetzmaske: 255.255.255.0



### HINWEIS!

Hierfür muss die aktuellste TwinCAT3 Version installiert werden inklusive dem Modul TF6310 TC3 TCP/IP. Details hierzu erhalten Sie von Ihrem Beckhoff-Support.

## 5.1 Prozessdaten vom Gerät TCP empfangen

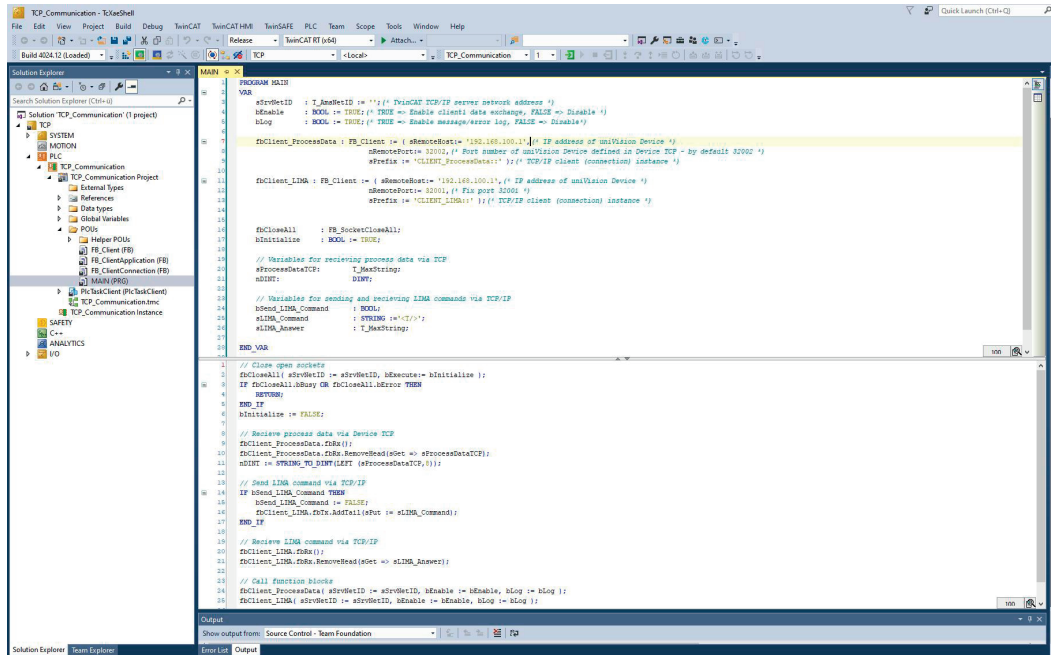
Das Beispielprogramm ist mit folgender Netzwerkeinstellung für das uniVision-Produkt erstellt:

- IP-Adresse: 192.168.100.1
- Subnetzmaske: 255.255.255.0

Die TCP-Prozessdaten werden standardmäßig über den Port 32002 verschickt.

Wird eine andere Netzwerkeinstellung oder ein anderer Port am uniVision-Produkt verwendet, so muss das Beispielprogramm entsprechend angepasst werden.

Hierfür im MAIN der TCP\_Communication unter fbClient\_ProcessData die IP-Adresse des uniVision-Produkts bei „sRemoteHost“ und den Port unter „nRemotePort“ eintragen.



```

PROGRAM MAIN
VAR
  sRemoteID : T_AnnetID := ''; (* TwinCAT TCP/IP server network address *)
  bEnable   : BOOL := TRUE; (* TRUE => Enable client's data exchange, FALSE => Disable *)
  bLog      : BOOL := TRUE; (* TRUE => Enable message/error log, FALSE => Disable *)

  fbClient_ProcessData : FB_Client := ( sRemoteHost := '192.168.100.1'; (* IP address of uniVision Device *)
                                         sRemotePort := 32002; (* Port number of uniVision Device defined in Device TCP - by default 32002 *)
                                         sPrefix := 'CLIENT_ProcessData'; ) (* TCP/IP client (connection) instance *)

  fbClient_LING : FB_Client := ( sRemoteHost := '192.168.100.1'; (* IP address of uniVision Device *)
                                sRemotePort := 32001; (* File port 32001 *)
                                sPrefix := 'CLIENT_LING'; ) (* TCP/IP client (connection) instance *)

  fbCloseAll : FB_SocketCloseAll;
  Initialize : BOOL := TRUE;

  // Variables for receiving process data via TCP
  sProcessDataTCP : T_String;
  sUDIP : UDIP;

  // Variables for sending and receiving LING commands via TCP/IP
  sSend_LING_Command : T_STRING := '';
  sLing_Command : T_STRING := '';
  sLing_Answer : T_String;
END VAR

// Close open sockets
fbCloseAll(sRemoteID := sRemoteID, bEnable := Initialize);
IF fbCloseAll.bBusy OR fbCloseAll.bError THEN
  RETURN;
END IF
Initialize := FALSE;

// Receive process data via Device TCP
fbClient_ProcessData.fbx(sRemoteID := sRemoteID, bEnable := bEnable, bLog := bLog);
nUDIP := sProcessDataTCP;

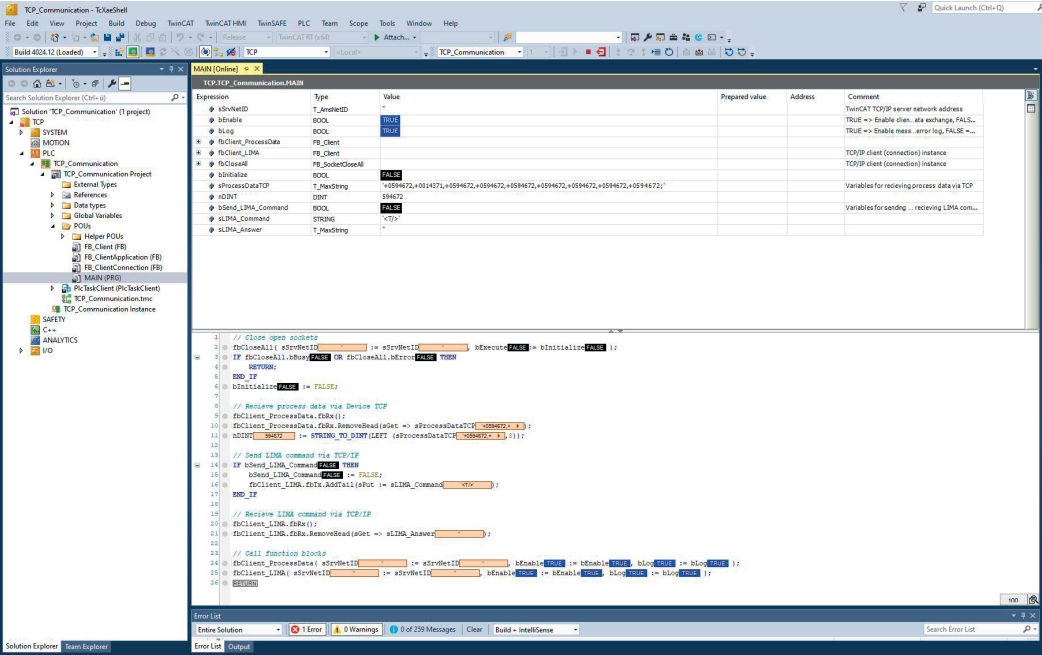
// Send LING command via TCP/IP
IF sSend_LING_Command THEN
  fbClient_LING.fbx(sRemoteID := sRemoteID, bEnable := bEnable, bLog := bLog);
  fbClient_LING.fbx(sRemoteID := sRemoteID, bEnable := bEnable, bLog := bLog);
END IF

// Receive LING command via TCP/IP
fbClient_LING.fbx(sRemoteID := sRemoteID, bEnable := bEnable, bLog := bLog);
fbClient_LING.fbx(sRemoteID := sRemoteID, bEnable := bEnable, bLog := bLog);

// Call function blocks
fbClient_ProcessData(sRemoteID := sRemoteID, bEnable := bEnable, bLog := bLog);
fbClient_LING(sRemoteID := sRemoteID, bEnable := bEnable, bLog := bLog);
  
```

Im Beispielprogramm ist für den ersten String auch die direkte Umwandlung der ersten acht Zeichen in eine Zahl (DINT) enthalten. Die Anzahl an Zeichen bzw. der Datentyp können beliebig geändert werden.

Das Beispielprogramm aktivieren, einloggen und starten. Die Prozessdaten, die vom Gerät TCP verschickt werden, erscheinen unter der Variablen „sProcessDataTCP“. Die Daten des ersten DINT erscheinen unter „nDINT“.



## 5.2 Prozessdaten vom Gerät UDP empfangen

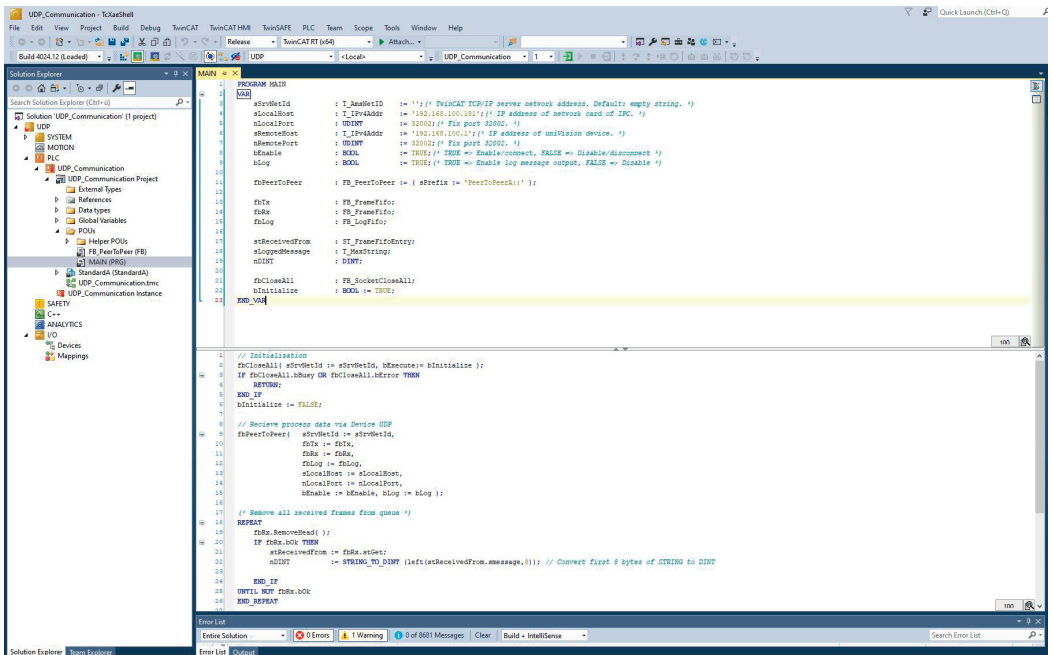
Das Beispielprogramm ist mit folgender Netzwerkeinstellung für das uniVision-Produkt erstellt:

- IP-Adresse: 192.168.100.1
- Subnetzmaske: 255.255.255.0

Die UDP-Prozessdaten werden über den Port 32002 verschickt.

Wird eine andere Netzwerkeinstellung am uniVision-Produkt verwendet, so muss das Beispielprogramm entsprechend angepasst werden.

Hierfür im MAIN der UDP\_Communication bei der Variablen „sRemoteHost“ die IP-Adresse des uniVision-Produkts eingeben.



```

PROGRAM MAIN
END

: T_DeviceID := "" (* TrinCAT TCP/IP server network address. Default: empty string. *)
: T_IPv4Addr := "192.168.100.1" (* IP address of network card of PLC. *)
: T_PORT := 32002 (* Fix port 32002. *)
: sRemoteHost := "192.168.100.1" (* IP address of uniVision device. *)
: sRemotePort := 32002 (* Fix port 32002. *)
: bEnable := TRUE (* TRUE => Enable connect, FALSE => Disable/disconnect *)
: bLog := TRUE (* TRUE => Enable log message output, FALSE => Disable *)

fbPeerToPeer := ( sPrefix := "PeerToPeer!" );

fbTx := FB_FrameInfo;
fbRx := FB_FrameInfo;
fbLog := FB_LogInfo;

sReceivedFrom := ST_FrameInfoEntry;
sLogMessage := T_String;
nDINT := DINT;

fbCloseAll := FB_SocketCloseAll;
bInitialize := BOOL := TRUE;

END_MAIN

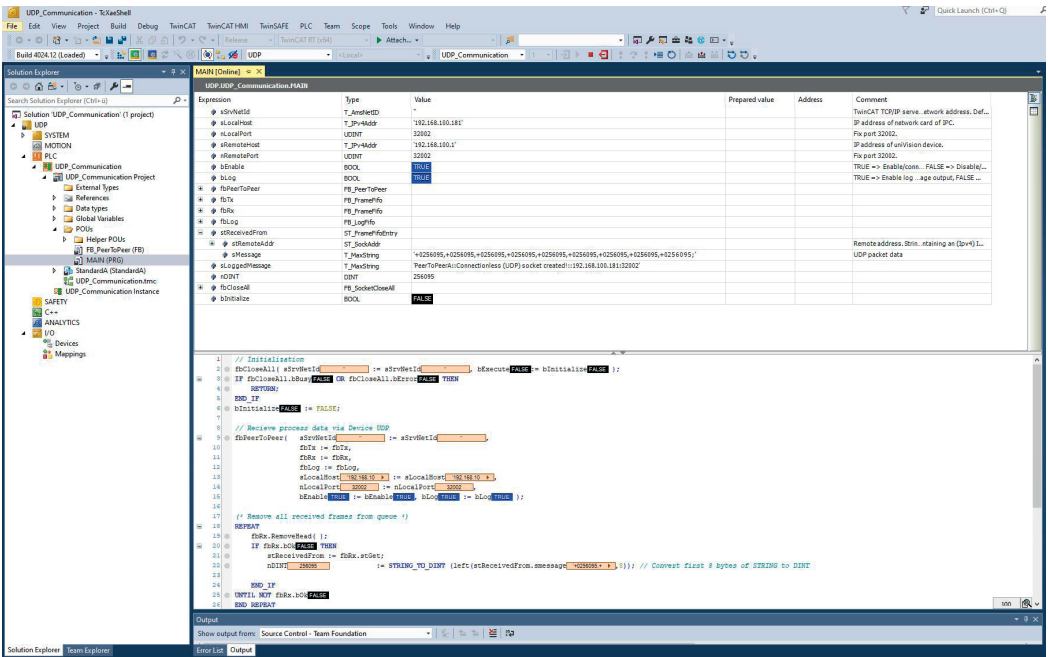
// Initialization
fbCloseAll sReceivedFrom := sDeviceID, sRemoteHost := bInitialize;
IF fbCloseAll.bBusy ON fbCloseAll.bError THEN
  RETURN;
END_IF
bInitialize := FALSE;

// Receive process data via Device UDP
fbPeerToPeer sDeviceID := sDeviceID,
fbTx := fbTx,
fbRx := fbRx,
fbLog := fbLog,
sLocalHost := sLocalHost,
sLocalPort := sLocalPort,
bEnable := bEnable, bLog := bLog;

(* Show all received frames from queue *)
REPEAT
  fbRx.RemoveRead();
  IF fbRx.bOK THEN
    sReceivedFrom := fbRx.sGet;
    nDINT := STRING_TO_DINT (left(sReceivedFrom, message, 1)); // Convert first 8 bytes of STRING to DINT
  END_IF
UNTIL NOT fbRx.bOK
END_REPEAT
  
```

Im Beispielprogramm ist für den ersten String auch die direkte Umwandlung der ersten acht Zeichen in eine Zahl (DINT) enthalten. Die Anzahl an Zeichen bzw. der Datentyp können beliebig geändert werden.

Das Beispielprogramm aktivieren, einloggen und starten. Die Prozessdaten, die vom Gerät UDP verschickt werden, erscheinen unter der Variablen „stReceivedFrom“ -> „sMessage“. Die Daten des ersten DINT erscheinen unter „nDINT“.







Das Beispielprogramm ist mit folgender Netzwerkeinstellung für das uniVision-Produkt erstellt:

- IP-Adresse: 192.168.100.1
- Subnetzmaske: 255.255.255.0

LIMA-Befehle werden über den Port 32001 verschickt.

Wird eine andere Netzwerkeinstellung am uniVision-Produkt verwendet, so muss das Beispielprogramm entsprechend angepasst werden.

Hierfür bei fbClient\_LIMA unter „sRemoteHost“ die IP-Adresse des uniVision-Produktes eintragen.

Das Beispielprogramm aktivieren, einloggen und starten.

### HINWEIS!



Der Verbindungsaufbau von der Steuerung zum uniVision-Produkt ist nur möglich, wenn der Port 32001 für die Steuerung verfügbar ist. Je nach Produkt bzw. Betriebsmodus der uniVision-Software wird der Port 32001 auch von der uniVision-Software benötigt (z.B. im Bearbeitungsmodus). In diesem Fall muss die Verbindung durch die uniVision-Software ggf. getrennt werden, damit die Verbindung über die Steuerung aufgebaut werden kann.

Der LIMA-Befehl wird durch das Setzen von „bSend\_LIMA\_Command“ auf TRUE an das uniVision-Produkt geschickt. Der Befehl darf nur einmal geschickt werden und darf nicht permanent anliegen, damit auch nur ein Bild bzw. Profil aufgenommen wird. Ein neuer Befehl darf erst geschickt werden, wenn die LIMA-Antwort zum vorherigen Befehl erhalten ist.

Die LIMA-Antwort ist in „sLIMA-Answer“ enthalten. Für den Triggerbefehl wird <Tok/> vom uniVision-Produkt als Antwort auf eine erfolgreiche Ausführung des Triggerbefehls verschickt. Zudem sind nach der Datenaufnahme und -auswertung ebenfalls die neuen Prozessdaten über TCP unter „sProcessDataTCP“ verfügbar. Über den Ausführzähler kann beispielsweise geprüft werden, wenn neue Ergebnisse vorhanden sind.

[illegible]

## 6. Rockwell-Beispielprogramme

Die Rockwell-Beispielprogramme für Prozessdaten und LIMA beinhalten folgende Anwendungsfälle:

- Prozessdaten vom Gerät TCP empfangen (im Beispielprogramm „Example\_Rockwell\_ProcessData.ACD“)
- Prozessdaten vom Gerät UDP empfangen (im Beispielprogramm „Example\_Rockwell\_ProcessData.ACD“)
- LIMA-Befehle (z. B. Triggerbefehle) über TCP/IP senden und die LIMA-Antwort empfangen (im Beispielprogramm „Example\_Rockwell\_LIMA.ACD“)

Beim Beispiel wird folgende Netzwerkkonfiguration verwendet:

- SPS:
  - IP-Adresse: 192.168.100.70
  - Subnetzmaske: 255.255.255.0
- uniVision-Produkt:
  - IP-Adresse: 192.168.100.1
  - Subnetzmaske: 255.255.255.0



### HINWEIS!

Das Beispielprogramm ist mit der SPS 1769-L18ERM-BB1B von Allen-Bradley mit Studio 5000 Logix Designer V32 erstellt.

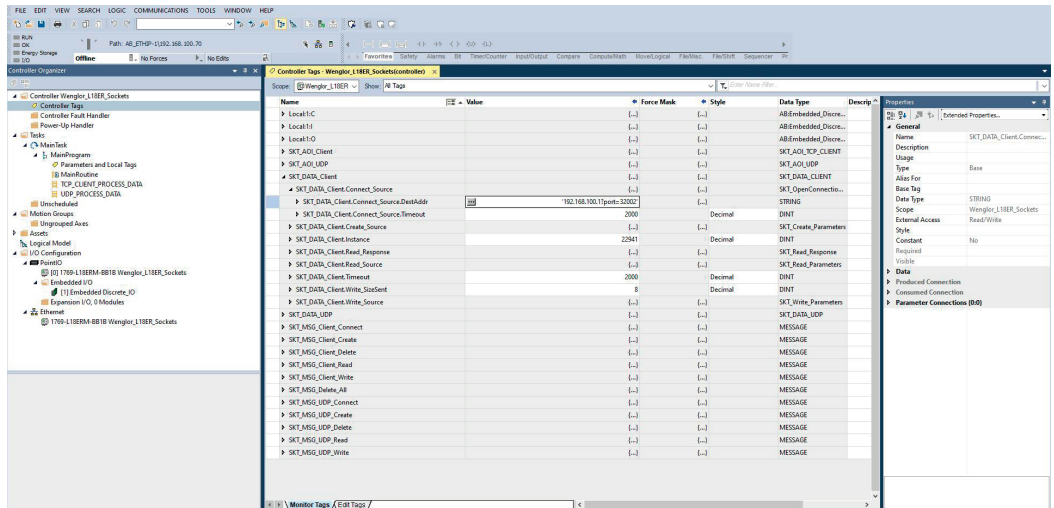
## 6.1 Prozessdaten vom Gerät TCP empfangen

Das Beispielprogramm „Example\_Rockwell\_ProcessData.ACD“ ist mit folgender Netzwerkeinstellung für das uniVision-Produkt erstellt:

- IP-Adresse: 192.168.100.1
- Subnetzmaske: 255.255.255.0

Die TCP-Prozessdaten werden standardmäßig über den Port 32002 verschickt.

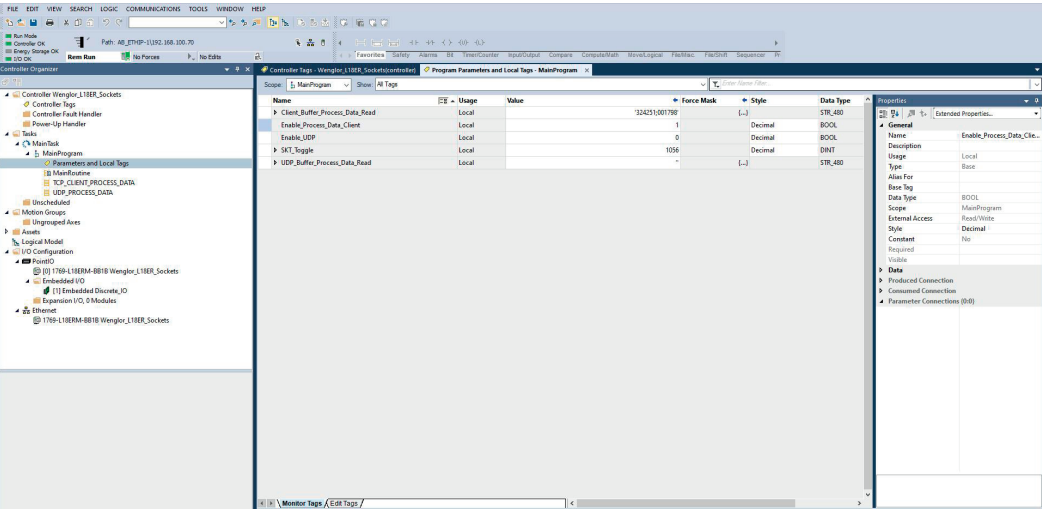
Wird eine andere Netzwerkeinstellung oder ein anderer Port am uniVision-Produkt verwendet, so muss das Beispielprogramm entsprechend angepasst werden. Hierfür die „Controller Tags“ öffnen und unter „SKT\_DATA\_Client.Connect\_Source.DestAddr“ die IP-Adresse und den Port eintragen.



The screenshot displays the 'Controller Tags' window in the Wenglor uniVision software. The 'SKT\_DATA\_Client.Connect\_Source.DestAddr' tag is selected, and its value is set to '192.168.100.11:port:32002'. The interface shows various other tags and their properties, including 'LocalIC', 'LocalH', 'LocalIO', 'SKT\_AOI\_Client', 'SKT\_AOI\_UDP', 'SKT\_DATA\_Client', 'SKT\_DATA\_Client.Connect\_Source', 'SKT\_DATA\_Client.Connect\_Source.Timeout', 'SKT\_DATA\_Client.Create\_Source', 'SKT\_DATA\_Client.Instance', 'SKT\_DATA\_Client.Read\_Response', 'SKT\_DATA\_Client.Read\_Source', 'SKT\_DATA\_Client.Timeout', 'SKT\_DATA\_Client.Write\_Source', 'SKT\_DATA\_UDP', 'SKT\_MSG\_Client.Connect', 'SKT\_MSG\_Client.Create', 'SKT\_MSG\_Client.Delete', 'SKT\_MSG\_Client.Read', 'SKT\_MSG\_Client.Write', 'SKT\_MSG\_Delete\_AB', 'SKT\_MSG\_UDP.Connect', 'SKT\_MSG\_UDP.Create', 'SKT\_MSG\_UDP.Delete', 'SKT\_MSG\_UDP.Read', and 'SKT\_MSG\_UDP.Write'.

Das Beispielprogramm an die Steuerung übertragen und online gehen.

Die TCP-Verbindung wird aufgebaut, indem unter „Parameters und Local Tags“ der Wert „Enable\_Process\_Data\_Client“ aktiviert wird. Die Prozessdaten, die vom Gerät TCP verschickt werden, erscheinen unter „Client\_Buffer\_Process\_Data\_Read“.



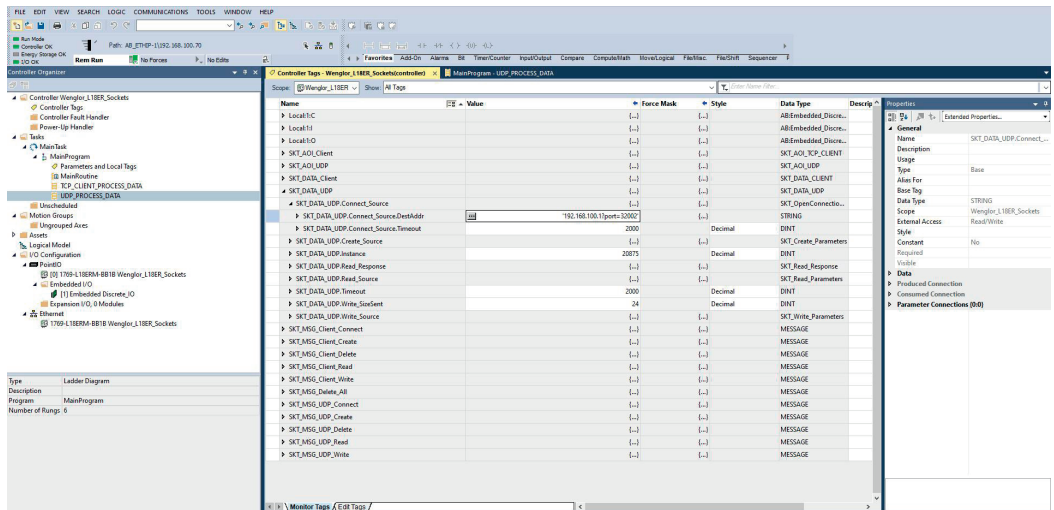
## 6.2 Prozessdaten vom Gerät UDP empfangen

Das Beispielprogramm „Example\_Rockwell\_ProcessData.ACD“ ist mit folgender Netzwerkeinstellung für das uniVision-Produkt erstellt:

- IP-Adresse: 192.168.100.1
- Subnetzmaske: 255.255.255.0

Die UDP-Prozessdaten werden über den Port 32002 verschickt.

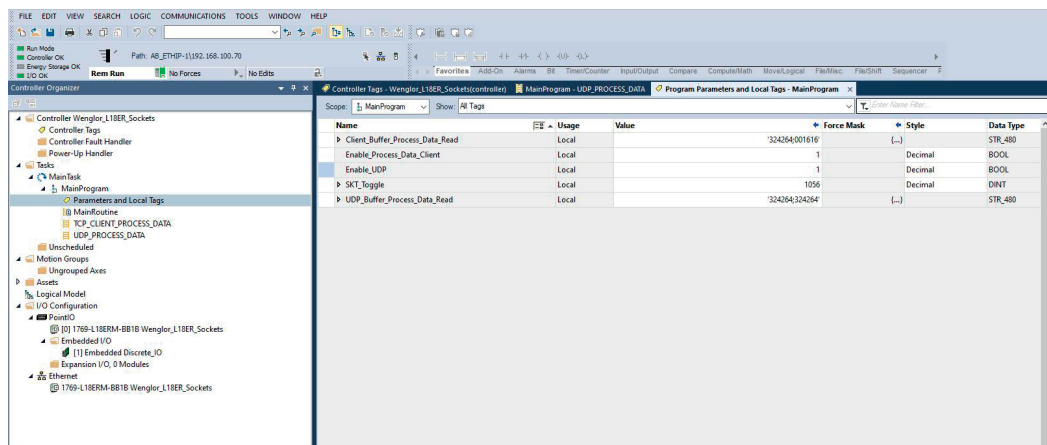
Wird eine andere Netzwerkeinstellung am uniVision-Produkt verwendet, so muss das Beispielprogramm entsprechend angepasst werden. Hierfür die „Controller Tags“ öffnen und unter „SKT\_DATA\_UDP.Connect\_Source.DestAddr“ die IP-Adresse eintragen.



| Name                                 | Value               | Force Mask | Style | Data Type             | Describe | Priority |
|--------------------------------------|---------------------|------------|-------|-----------------------|----------|----------|
| Local:IC                             |                     | [-]        | [-]   | ABEmbedded_Discrete   |          |          |
| Local:II                             |                     | [-]        | [-]   | ABEmbedded_Discrete   |          |          |
| Local:IO                             |                     | [-]        | [-]   | ABEmbedded_Discrete   |          |          |
| SKT_AIO_Client                       |                     | [-]        | [-]   | SKT_AIO_CLIENT        |          |          |
| SKT_AIO_UDP                          |                     | [-]        | [-]   | SKT_AIO_UDP           |          |          |
| SKT_DATA_Client                      |                     | [-]        | [-]   | SKT_DATA_CLIENT       |          |          |
| SKT_DATA_UDP                         |                     | [-]        | [-]   | SKT_DATA_UDP          |          |          |
| SKT_DATA_UDP.Connect_Source          |                     | [-]        | [-]   | SKT_OpenConnection... |          |          |
| SKT_DATA_UDP.Connect_Source.DestAddr | 192.168.100.1:32002 |            |       | STRING                |          |          |
| SKT_DATA_UDP.Connect_Source.Timeout  | 2000                |            |       | Decimal               |          |          |
| SKT_DATA_UDP.Create_Source           |                     | [-]        | [-]   | SKT_Create_Parameters |          |          |
| SKT_DATA_UDP.Instance                | 20875               |            |       | Decimal               |          |          |
| SKT_DATA_UDP.Read_Response           |                     | [-]        | [-]   | SKT_Read_Response     |          |          |
| SKT_DATA_UDP.Read_Source             |                     | [-]        | [-]   | SKT_Read_Parameters   |          |          |
| SKT_DATA_UDP.Timeout                 | 2000                |            |       | Decimal               |          |          |
| SKT_DATA_UDP.Write_SrcInst           | 24                  |            |       | Decimal               |          |          |
| SKT_DATA_UDP.Write_Source            |                     | [-]        | [-]   | SKT_Write_Parameters  |          |          |
| SKT_MSG_Client.Connect               |                     | [-]        | [-]   | MESSAGE               |          |          |
| SKT_MSG_Client.Create                |                     | [-]        | [-]   | MESSAGE               |          |          |
| SKT_MSG_Client.Delete                |                     | [-]        | [-]   | MESSAGE               |          |          |
| SKT_MSG_Client.Read                  |                     | [-]        | [-]   | MESSAGE               |          |          |
| SKT_MSG_Client.Write                 |                     | [-]        | [-]   | MESSAGE               |          |          |
| SKT_MSG_Delete_All                   |                     | [-]        | [-]   | MESSAGE               |          |          |
| SKT_MSG_UDP.Connect                  |                     | [-]        | [-]   | MESSAGE               |          |          |
| SKT_MSG_UDP.Create                   |                     | [-]        | [-]   | MESSAGE               |          |          |
| SKT_MSG_UDP.Delete                   |                     | [-]        | [-]   | MESSAGE               |          |          |
| SKT_MSG_UDP.Read                     |                     | [-]        | [-]   | MESSAGE               |          |          |
| SKT_MSG_UDP.Write                    |                     | [-]        | [-]   | MESSAGE               |          |          |

Das Beispielprogramm an die Steuerung übertragen und online gehen.

Zum Empfangen der UDP-Prozessdaten unter „Parameters und Local Tags“ den Wert „Enable\_UDP“ aktivieren. Die Prozessdaten, die vom Gerät UDP verschickt werden, erscheinen unter „UDP\_Buffer\_Process\_Data\_Read“.

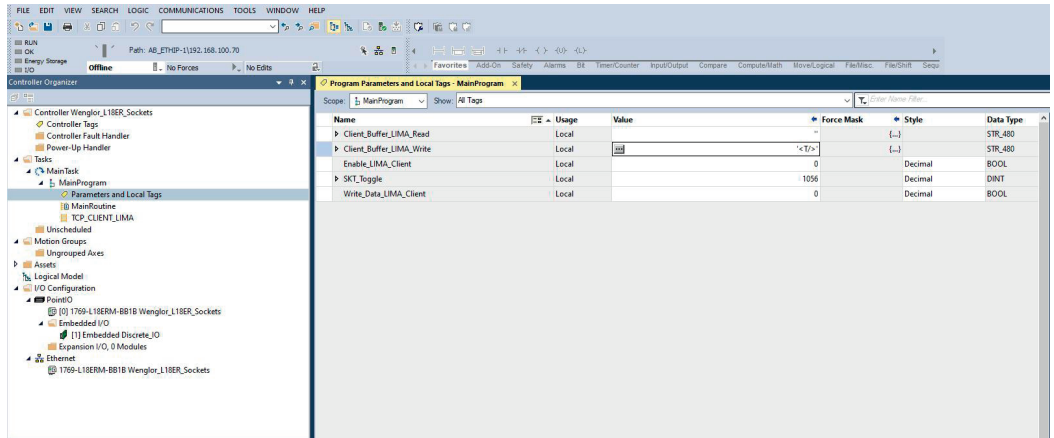




### 6.3 LIMA-Befehle über TCP/IP senden und LIMA-Antworten empfangen

Über die TCP/IP-Schnittstelle können LIMA-Befehle verschickt werden. Im Beispielprogramm „Example\_Rockwell\_LIMA.ACD“ wird ein Triggerbefehl an das uniVision-Produkt geschickt, der eine Bild- bzw. Profilaufnahme auslöst. Details zu den verfügbaren Befehlen befinden sich im LIMA-Schnittstellenprotokoll. Es ist verfügbar im Downloadbereich der Produktdetailseite von uniVision (<https://www.wenglor.com/product/DNNF020>).

Der LIMA-Befehl muss bei „Parameters and Local Tags“ unter „Client\_Buffer\_LIMA\_Write“ eingetragen werden. Für den Triggerbefehl muss hierfür <T/> geschickt werden.



Das Beispielprogramm ist mit folgender Netzwerkeinstellung für das uniVision-Produkt erstellt:

- IP-Adresse: 192.168.100.1
- Subnetzmaske: 255.255.255.0

LIMA-Befehle werden über den Port 32001 verschickt.

Wird eine andere Netzwerkeinstellung am uniVision-Produkt verwendet, so muss das Beispielprogramm entsprechend angepasst werden. Hierfür die „Controller Tags“ öffnen und unter „SKT\_DATA\_Client\_LIMA.Connect\_Source.DestAddr“ die IP-Adresse eintragen.

The screenshot shows the 'Controller Tags' window in the UniView software. The window is titled 'Program Parameters and Local Tags - Controller Tags - Wenglor\_L18ER\_Sockets(controller)'. It displays a list of tags for the 'Wenglor\_L18ER' controller. The tag 'SKT\_DATA\_Client\_LIMA.Connect\_Source.DestAddr' is highlighted, showing its value as '192.168.100.11ports:32001'. The table below shows the details of the tags.

| Name   | Force Mask | Style | Data Type            | Description |
|--|------------|-------|----------------------|-------------|
| Local:I:C                                    | (...)      | (...) | ABEmbedded_Discr...  |             |
| Local:I:I                                    | (...)      | (...) | ABEmbedded_Discr...  |             |
| Local:I:O                                    | (...)      | (...) | ABEmbedded_Discr...  |             |
| SKT_AOI_Client_LIMA                          | (...)      | (...) | SKT_AOI_TCP_CLIENT   |             |
| SKT_DATA_Client_LIMA                         | (...)      | (...) | SKT_DATA_CLIENT      |             |
| SKT_DATA_Client_LIMA.Connect_Source          | (...)      | (...) | SKT_OpenConnectio... |             |
| SKT_DATA_Client_LIMA.Connect_Source.DestAddr | (...)      | (...) | STRING               |             |
| SKT_DATA_Client_LIMA.Connect_Source.Timeout  | 2000       | (...) | Decimal              | DINT        |
| SKT_DATA_Client_LIMA.Create_Source           | (...)      | (...) | Decimal              | DINT        |
| SKT_DATA_Client_LIMA.Instance                | 29941      | (...) | Decimal              | DINT        |
| SKT_DATA_Client_LIMA.Read_Response           | (...)      | (...) | SKT_Read_Response    |             |
| SKT_DATA_Client_LIMA.Read_Source             | (...)      | (...) | SKT_Read_Parameters  |             |
| SKT_DATA_Client_LIMA.Timeout                 | 2000       | (...) | Decimal              | DINT        |
| SKT_DATA_Client_LIMA.Write_SizeSent          | 8          | (...) | Decimal              | DINT        |
| SKT_DATA_Client_LIMA.Write_Source            | (...)      | (...) | SKT_Write_Parameters |             |
| SKT_MSG_Client_Connect_LIMA                  | (...)      | (...) | MESSAGE              |             |
| SKT_MSG_Client_Create_LIMA                   | (...)      | (...) | MESSAGE              |             |
| SKT_MSG_Client_Delete_LIMA                   | (...)      | (...) | MESSAGE              |             |
| SKT_MSG_Client_Read_LIMA                     | (...)      | (...) | MESSAGE              |             |
| SKT_MSG_Client_Write_LIMA                    | (...)      | (...) | MESSAGE              |             |
| SKT_MSG_Delete_All                           | (...)      | (...) | MESSAGE              |             |

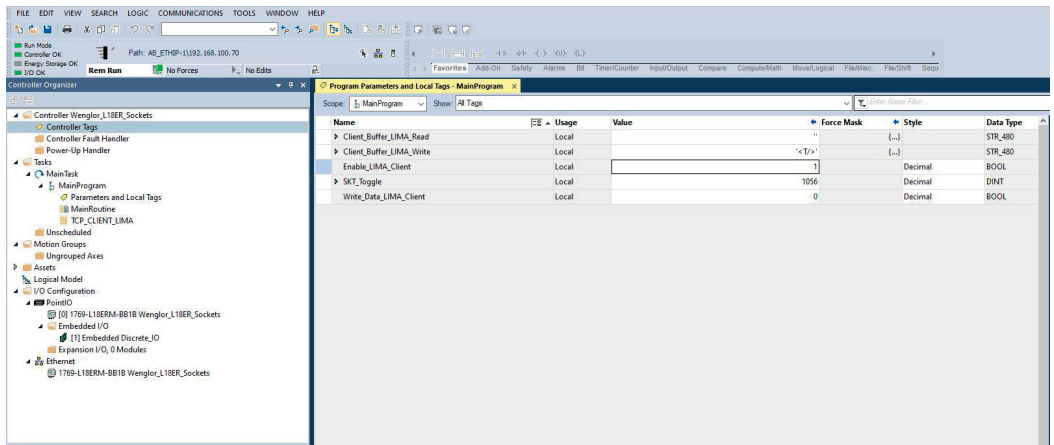
Das Beispielprogramm an die Steuerung übertragen und online gehen.

## HINWEIS!

Der Verbindungsaufbau von der Steuerung zum uniVision-Produkt ist nur möglich, wenn der Port 32001 für die Steuerung verfügbar ist. Je nach Produkt bzw. Betriebsmodus der uniVision-Software wird der Port 32001 auch von der uniVision-Software benötigt (z. B. im Bearbeitungsmodus). In diesem Fall muss die Verbindung durch die uniVision-Software ggf. getrennt werden, damit die Verbindung über die Steuerung aufgebaut werden kann.

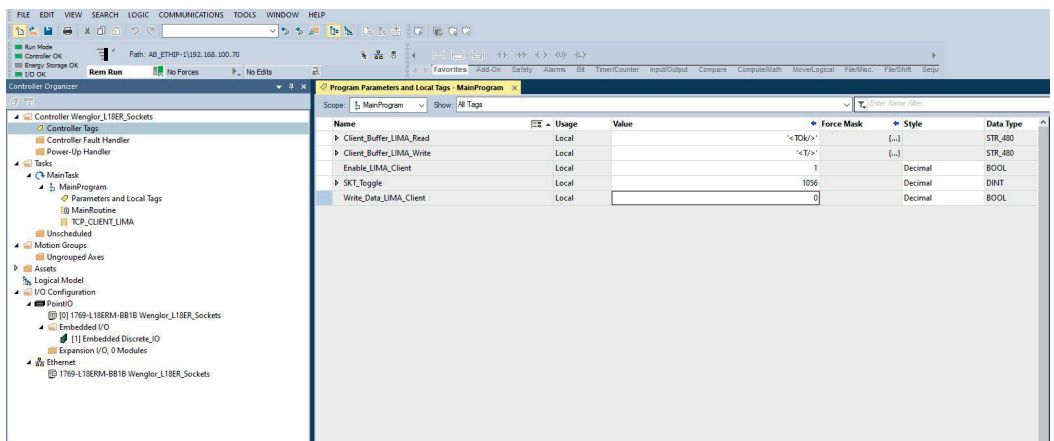


Die TCP-Verbindung wird aufgebaut, indem unter „Parameters und Local Tags“ der Wert „Enable\_LIMA\_Client“ aktiviert wird.



Der LIMA-Befehl wird durch Aktivieren von „Write\_Data\_LIMA\_Client“ an das uniVision-Produkt geschickt. Der Befehl darf nur einmal geschickt werden und darf nicht permanent anliegen, damit auch nur ein Bild bzw. Profil aufgenommen wird. Ein neuer Befehl darf erst geschickt werden, wenn die LIMA-Antwort zum vorherigen Befehl erhalten ist.

Die LIMA-Antwort ist unter „Client\_Buffer\_LIMA\_Read“ enthalten. Für den Triggerbefehl wird <TOK/> vom uniVision-Produkt als Antwort auf eine erfolgreiche Ausführung des Triggerbefehls verschickt.



Zudem sind nach der Datenaufnahme und -auswertung ebenfalls die neuen Prozessdaten über TCP verfügbar (s. Beispielprogramm „Example\_Rockwell\_ProcessData.ACD“). Über den Ausführzähler kann beispielsweise geprüft werden, wenn neue Ergebnisse vorhanden sind.