

Rexroth IndraLogic XLC IndraMotion MLC 13VRS HMI-Anbindung

R911336361
Ausgabe 02

Anwendungsbeschreibung



Titel Rexroth IndraLogic XLC
IndraMotion MLC 13VRS
HMI-Anbindung

Art der Dokumentation Anwendungsbeschreibung

Dokumentations-Type DOK-XLCMLC-HMI*****V13-AP02-DE-P

Interner Ablagevermerk RS-dcfa675221d6f7cb0a6846a00062316e-2-de-DE-24

Änderungsverlauf

Ausgabe	Stand	Bemerkung
Ausgabe 01	2012-04	Erstausgabe
Ausgabe 02	2014-07	Überarbeitung, mit BRWS-Treiber-Dokumentation

Schutzvermerk © Bosch Rexroth AG 2014

Alle Rechte bei Bosch Rexroth AG, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Verbindlichkeit Die angegebenen Daten dienen allein der Produktbeschreibung und sind nicht als zugesicherte Eigenschaften im Rechtssinne zu verstehen. Änderungen im Inhalt der Dokumentation und Liefermöglichkeiten der Produkte sind vorbehalten.

Redaktion Systementwicklung Automation Motion Logic Control, AHo (FrWe, vha)

Inhaltsverzeichnis

	Seite
1 Zu dieser Dokumentation.....	9
1.1 Gültigkeit der Dokumentation.....	9
1.2 Gebrauch der Sicherheitshinweise.....	9
1.2.1 Aufbau der Sicherheitshinweise.....	9
1.2.2 Erläuterung der Signalwörter und der Signalgrafik.....	10
1.2.3 Verwendete Symbole.....	10
1.3 Erforderliche und ergänzende Dokumentationen.....	11
1.4 Bezeichnungen und Abkürzungen.....	16
 2 Visualisierungsgeräte.....	 17
2.1 Übersicht.....	17
2.2 IndraControl VCP.....	17
2.2.1 Kurzbeschreibung.....	17
2.2.2 Technische Daten.....	19
2.2.3 Ausführungen.....	19
2.2.4 Zubehör.....	20
Verbindungskabel.....	20
Batterie-Kit.....	21
2.2.5 Dokumentation.....	21
2.3 IndraControl VCH.....	22
2.3.1 Kurzbeschreibung.....	22
2.3.2 Einsatzbereich.....	22
2.3.3 Technische Daten.....	22
2.3.4 Ausführungen.....	23
2.3.5 Zubehör.....	23
2.3.6 Dokumentation.....	23
2.4 IndraControl VR21.....	23
2.4.1 Kurzbeschreibung.....	23
2.4.2 Varianten.....	25
2.4.3 Technische Daten.....	26
VR 21 Singletouch.....	26
VR 21 Multitouch.....	26
2.4.4 Gehäuse - und Einbaumaße.....	27
2.4.5 Schnittstellen.....	28
Schnittstellenansicht.....	28
Schnittstellenübersicht.....	28
DC-24-V-Spannungsversorgung.....	28
USB-Schnittstellen X9, X10.....	29
Ethernet-Schnittstelle X5.....	29
Steckplatz für SD-Speicherkarte.....	29
2.4.6 Bestellinformationen.....	30
Zubehör- und Ersatzteile.....	30
Typenschlüssel VR21xx.01.....	31
2.4.7 Dokumentation.....	31

Inhaltsverzeichnis

	Seite
2.5 IndraControl VEP.....	31
2.5.1 Kurzbeschreibung.....	31
2.5.2 Einsatzbereich.....	32
2.5.3 Technische Daten.....	32
2.5.4 Verschleißteile.....	33
2.5.5 Ausführungen.....	33
2.5.6 Zubehör.....	34
2.5.7 Dokumentation.....	34
2.6 IndraControl VPB und VDP.....	34
2.6.1 Kurzbeschreibung.....	34
2.6.2 Einsatzbereich.....	35
2.6.3 Technische Daten.....	35
VPB 40.3.....	35
VDP 16.3/40.3/60.3.....	35
2.6.4 Verschleißteile.....	35
2.6.5 Ausführungen.....	35
2.6.6 Zubehör.....	37
Verbindungskabel (CDI-Interface).....	37
Zubehör für VPB 40.3.....	37
2.6.7 Dokumentation.....	37
2.7 IndraControl VPP.....	38
2.7.1 Kurzbeschreibung.....	38
2.7.2 Einsatzbereich.....	38
2.7.3 Technische Daten.....	38
2.7.4 Verschleißteile.....	39
2.7.5 Ausführungen.....	39
2.7.6 Zubehör.....	40
2.7.7 Dokumentation.....	40
3 Visualisierungen.....	41
3.1 Übersicht.....	41
3.2 VI-Logic.....	41
3.2.1 Einsatzgebiet.....	41
3.2.2 Eigenschaften der integrierten Visualisierung.....	41
3.2.3 Visualisierungselemente.....	42
3.2.4 Unterstützte Visualisierungsgeräte.....	43
3.2.5 Protokoll.....	43
3.2.6 Dokumentation.....	44
3.3 VI-Composer.....	44
3.3.1 Einsatzgebiet.....	44
3.3.2 Projektierbare Kleinbedienterminals.....	44
3.3.3 Kommunikationsprotokoll BRC-Symbolic.....	44
Kurzbeschreibung.....	44
Datenobjekte.....	44
Einzelvariablen.....	44
Stringvariablen.....	45

Inhaltsverzeichnis

	Seite
Tabellen.....	45
3.3.4 Verbindungseinstellungen.....	45
Protokollparameter für PROFIBUS-DP.....	45
Protokollparameter für Ethernet.....	45
3.3.5 Dokumentation.....	46
3.4 WinStudio.....	46
3.4.1 Übersicht.....	46
3.4.2 WinStudio-Elemente im Projektextplorer.....	47
3.4.3 Allgemeine WinStudio Einstellungen und Funktionen.....	47
3.4.4 WinStudio-Bilder.....	61
Erstellen und Bearbeiten von WinStudio-Bildern	61
Bildeigenschaften.....	62
Objekteigenschaften und Variablenauswahl.....	63
3.4.5 WinStudio-Variablen.....	63
3.4.6 Verlinkte Symbole.....	64
WinStudio-Bibliothek.....	64
Anwendersymbole.....	65
3.4.7 WinStudio-Tasks.....	67
Mathematikblätter.....	67
Alarmer.....	67
Rezepte.....	67
Reports.....	68
Trends.....	69
Scheduler.....	69
Externe Datenbanken.....	69
Skripte.....	70
Übersicht.....	70
Globale Prozeduren.....	70
Start-Skript.....	70
Grafik-Skript.....	70
Bild-Skripte.....	71
Anwender-Skripte.....	71
3.4.8 WinStudio-Werkzeuge.....	72
Anwendung verifizieren.....	72
Anwendung starten.....	72
3.4.9 Datenbank-Spion.....	73
3.4.10 LogWin.....	74
3.4.11 Debugger.....	74
3.4.12 Querverweissuche.....	75
3.4.13 Variablen-Importassistent.....	75
3.4.14 Export/Import.....	76
3.4.15 WinStudio-Lizenzierung.....	76
Einleitung.....	76
Übersicht der WinStudio-Lizenzen.....	76
Aktivierung einer WinStudio-Lizenz.....	77
Auswirkungen einer fehlenden oder zu geringen Lizenz.....	78

Inhaltsverzeichnis

	Seite
3.4.16	Mehrsprachigkeit..... 78
3.4.17	WinStudio-Applikationen..... 78
	Übersicht..... 78
	Erstellung einer WinStudio-Applikation..... 78
	Benutzerverwaltung..... 79
	Web-Funktionalität..... 79
	Übertragen der Applikation und Aktivierung auf dem Zielgerät..... 79
3.4.18	Weiterführende Informationen..... 81
4	Kommunikationsschnittstellen..... 83
4.1	Übersicht..... 83
4.2	BRWS-Treiber (Bosch-Rexroth-WinStudio Treiber)..... 83
4.2.1	Eigenschaften..... 83
4.2.2	Anwendungsgebiete..... 84
4.2.3	Konfiguration..... 84
4.2.4	Treiberblätter..... 85
4.2.5	Protokoll-Aufbau..... 87
4.2.6	Konventionen..... 88
4.2.7	Unterstützung SPS-Variablen und Parameter..... 88
	SPS-Variablen..... 88
	XLC/MLC Parameter..... 89
4.2.8	Fehlermeldungen..... 91
4.2.9	Erläuterungen der Meldungen..... 91
4.2.10	Dokumentation..... 92
4.3	OPC.lwSCP..... 92
4.3.1	Eigenschaften..... 92
4.3.2	SCP-OPC Konfigurator..... 93
	Allgemein..... 93
	Bedienung..... 93
	Gerätekonfiguration erstellen..... 93
	Gerätekonfiguration öffnen..... 94
	Geräte hinzufügen..... 94
	Geräteeeigenschaften anzeigen..... 95
	Geräte OPC Alias ändern..... 96
	Geräte entfernen..... 97
	Gerätekonfiguration speichern..... 97
	Gerätekonfiguration speichern unter..... 97
	Gerätekonfiguration aktivieren..... 97
	Anwendung beenden..... 98
4.3.3	lwSCP und OPC.lwSCP Items..... 98
	Allgemein..... 98
	IndraLogic Items..... 100
	Übersicht..... 100
	sercos Items..... 100
	Übersicht..... 100
4.3.4	Haftungsausschluss..... 101

Inhaltsverzeichnis

	Seite
4.4	IndraLogic OPC.DA..... 101
4.4.1	Eigenschaften..... 101
4.4.2	Konfiguration des IndraLogic OPC.DA Servers..... 101
	Allgemein..... 101
	OPC.DA Server Konfiguration..... 102
	Kommunikationsadresse ermitteln..... 102
	OPC.DA Server Konfigurator..... 103
	OPC.DA Server Konfigurationsdatei "OPCServer.ini"..... 105
	Ablageverzeichnis der OPC.DA Server Konfigurationsdatei..... 110
	Allgemeines..... 110
	Windows XP..... 110
	Windows 7..... 111
	Windows CE..... 111
	OPC.DA Server - Item Syntax..... 113
4.4.3	Haftungsausschluss..... 113
4.5	Konfigurieren einer OPC Clientanbindung mit WinStudio 113
4.6	OPC UA..... 117
4.6.1	Übersicht..... 117
4.6.2	Einführung in OPC Unified Architecture..... 117
	Allgemein..... 117
	Übersicht der Spezifikationen..... 117
	OPC UA Informationsmodell..... 118
	Serviceorientierte Architektur..... 118
4.6.3	Der Bosch Rexroth OPC UA Server..... 123
	Allgemeines..... 123
	Eigenschaften des Bosch Rexroth OPC UA Servers..... 123
	Konfiguration..... 124
	Konfiguration durch den Client..... 125
	Der Namensraum der SPS..... 127
4.6.4	Kommunikationsbeispiel über OPC UA..... 128
	Einleitung..... 128
	Voraussetzungen..... 128
	UaExpert..... 128
	Verbindungsaufbau..... 130
	Browsen..... 132
	Subscribe und Write..... 132
4.7	Konfigurieren einer OPC UA Client-Anbindung mit WinStudio..... 135
4.8	Modbus TCP..... 138
4.8.1	Einführung und Übersicht..... 138
5	Applikationsbeispiele..... 141
5.1	Übersicht..... 141
5.2	Symbolkonfiguration..... 141
5.2.1	Allgemein..... 141
5.2.2	Symbolkonfiguration mit Wizard..... 142
5.2.3	Symbolkonfiguration über Pragma..... 143

Inhaltsverzeichnis

	Seite
5.3	VI-Composer..... 144
5.4	WinStudio mit IndraLogic XLC..... 144
5.4.1	Allgemein..... 144
5.4.2	Technische Voraussetzungen..... 145
5.4.3	Erstellen des IndraWorks-Projekts..... 145
5.4.4	Einbinden der WinStudio-Visualisierung in IndraWorks..... 151
5.4.5	Erstellen der WinStudio-Visualisierung..... 158
5.4.6	Starten der Oberfläche..... 165
5.5	IndraLogic Visualisierung..... 167
5.6	HMI's von Fremdherstellern..... 167
5.6.1	OPC.DA Server..... 167
	Hilfsmittel zur Konfiguration des OPC.DA Server auf einem PC..... 167
	Hilfsmittel zum Übertragen der OPC.DA Server Konfiguration auf ein Visualisierungsgerät VEPxx. 4 mit WindowsCE..... 168
5.6.2	ModbusTCP..... 169
	Allgemein..... 169
	Projektierung mit Siemens WinCC-flexible 2007..... 171
	Projektierung mit Pro-face GP-Pro EX 2.2..... 174
6	Client-Entwicklung..... 183
6.1	OPC Client Visual Basic .NET..... 183
6.1.1	Allgemein..... 183
6.1.2	Visual Basic Projekt "OPC Client"..... 183
	Neues Projekt..... 183
	Verweise hinzufügen..... 184
	Windows Form und Steuerelement hinzufügen..... 185
	OPC Client Implementierung..... 187
	Allgemein..... 187
	Deklaration der Variablen..... 188
	OPC Server auflisten..... 189
	OPC Server Instanz erstellen und löschen..... 190
	OPC Subscription erstellen und löschen..... 191
	Steuerungsisems auflisten..... 194
	OPC Items hinzufügen und löschen..... 196
	Steuerungsdaten lesen..... 199
	Steuerungsdaten schreiben..... 202
	Zyklisches Lesen der Steuerungsdaten..... 204
	Fehlerbehandlung..... 206
6.2	OPC Client Visual C# .NET..... 206
6.2.1	Allgemein..... 206
6.2.2	Visual C# Projekt "OPC Client"..... 207
	Neues Projekt..... 207
	Verweise hinzufügen..... 207
	Windows Form und Steuerelement hinzufügen..... 208
	OPC Client Implementierung..... 210
	Allgemein..... 210

Inhaltsverzeichnis

	Seite
Deklaration der Variablen.....	211
OPC Server auflisten.....	212
OPC Server Instanz erstellen und löschen.....	212
OPC Subscription erstellen und löschen.....	215
Steuerungsitems auflisten.....	217
OPC Items hinzufügen und löschen.....	218
Steuerungsdaten lesen.....	221
Steuerungsdaten schreiben.....	223
Zyklisches Lesen der Steuerungsdaten.....	226
Fehlerbehandlung.....	227
7 Service und Support.....	229
Index.....	231

1 Zu dieser Dokumentation

1.1 Gültigkeit der Dokumentation

Zielgruppe Diese Dokumentation richtet sich an Anwender, die die unterstützten Visualisierungssysteme, der IndraLogic XLC / IndraMotion MLC, IndraMotion MTX oder IndraMotion MLD (IL 2G) anbinden wollen.

Diese Dokumentation unterstützt den Anwender in den Phasen

- Auswahl
- Engineering
- Inbetriebnahme

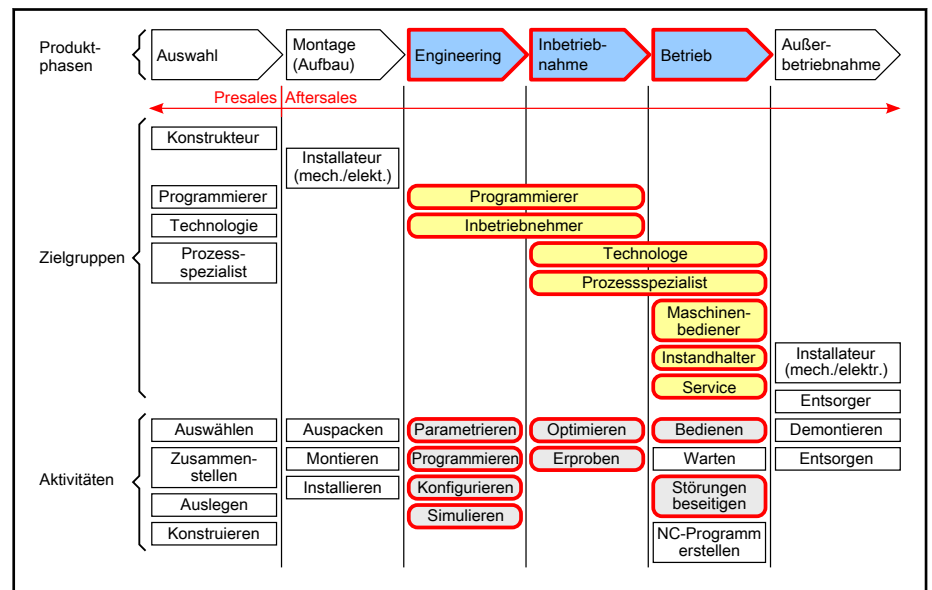


Abb. 1-1: Anwendungsphasen

Erforderliche Qualifikationen: Person, die aufgrund ihrer fachlichen Ausbildung, Kenntnisse und Erfahrungen sowie Kenntnis der einschlägigen Normen und Bestimmungen die ihr übertragenen Arbeiten beurteilen und mögliche Gefahren erkennen kann.

1.2 Gebrauch der Sicherheitshinweise

1.2.1 Aufbau der Sicherheitshinweise

Die Sicherheitshinweise sind wie folgt aufgebaut:

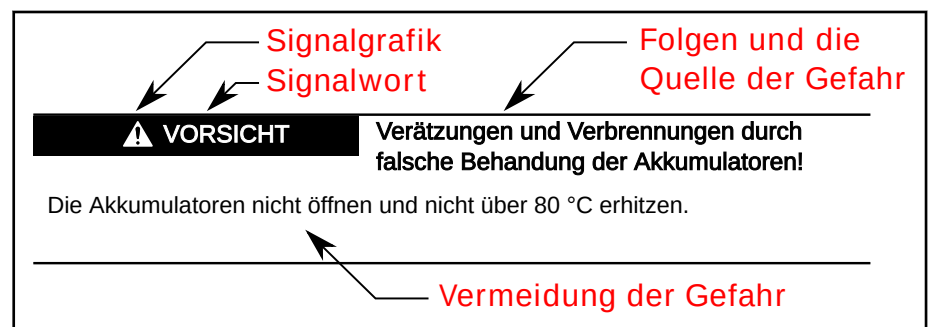


Abb. 1-2: Aufbau der Sicherheitshinweise

Zu dieser Dokumentation

1.2.2 Erläuterung der Signalwörter und der Signalgrafik

Die Sicherheitshinweise in der vorliegenden Dokumentation beinhalten bestimmte Signalwörter (Gefahr, Warnung, Vorsicht, Hinweis) und gegebenenfalls eine Signalgrafik (nach ANSI Z535.6-2006).

Das Signalwort soll die Aufmerksamkeit auf den Sicherheitshinweis lenken und bezeichnet die Schwere der Gefährdung.

Die Signalgrafik (Warndreieck mit Ausrufezeichen), welche den Signalwörtern Gefahr, Warnung und Vorsicht vorangestellt wird, weist auf Gefährdungen für Personen hin.

GEFAHR

Bei Nichtbeachtung dieses Sicherheitshinweises **werden** Tod oder schwere Körperverletzung eintreten.

WARNUNG

Bei Nichtbeachtung dieses Sicherheitshinweises **können** Tod oder schwere Körperverletzung eintreten.

VORSICHT

Bei Nichtbeachtung dieses Sicherheitshinweises können mittelschwere oder leichte Körperverletzung eintreten.

HINWEIS

Bei Nichtbeachtung dieses Sicherheitshinweises können Sachschäden eintreten.

1.2.3 Verwendete Symbole

Fingerzeige werden wie folgt dargestellt:



Dies ist ein Hinweis.

Tipps werden wie folgt dargestellt:



Dies ist ein Tipp.

1.3 Erforderliche und ergänzende Dokumentationen

Dokumentationstitel mit Typenschlüssel und Materialnummer

IndraWorks		MLC	XLC
/36/	Rexroth IndraWorks 13VRS Software-Installation DOK-IWORKS-SOFTINS*V13-CORS-DE-P, R911336879 Die Dokumentation beschreibt den Installationsprozess von IndraWorks.	X	X
/5/	Rexroth IndraWorks 13VRS Engineering DOK-IWORKS-ENGINEE*V13-APRS-DE-P, R911336869 Diese Dokumentation beschreibt die Verwendung von IndraWorks, in das sich die Rexroth Engineering-Tools integrieren. Beschrieben werden das Arbeiten mit IndraWorks und die Bedienung der Oszilloskop-Funktion.	X	X
/20/	Rexroth IndraMotion MLC 13VRS Funktionsbeschreibung DOK-MLC***-FUNC****V13-APRS-DE-P, R911336294 Die Dokumentation dient der Beschreibung der Wizards, Kontextmenüs, Dialoge, Steuerungsinbetriebnahme, Gerätekonfiguration, Funktionalitäten der IndraMotion MLC.	X	
/20/	Rexroth IndraLogic XLC 13VRS Funktionsbeschreibung DOK-XLC***-FUNC****V13-APRS-DE-P, R911336351 Die Dokumentation dient der Beschreibung der Wizards, Kontextmenüs, Dialoge, Steuerungsinbetriebnahme, Gerätekonfiguration, Funktionalitäten der IndraLogic XLC.		X
/7/	Rexroth IndraWorks 13VRS CamBuilder DOK-IWORKS-CAMBUIL*V13-APRS-DE-P, R911336290 Diese Dokumentation dient zur Beschreibung der Grundlagen und der Bedienung des CamBuilder, des Werkzeugs zur Kurvenscheibebearbeitung.	X	X
/37/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS Automation Interface DOK-XLCMLC-AUT*INT*V13-APRS-DE-P, R911336355 Diese Dokumentation beschreibt den skriptbasierten Zugriff auf IndraWorks-Projektdateien über die Schnittstelle des Automation Interfaces.	X	X
/38/	Rexroth IndraWorks 12VRS FDT-Container DOK-IWORKS-FDT*CON*V12-APRS-DE-P, R911334397 Diese Dokumentation beschreibt die FDT-Container-Funktionalität von IndraWorks. Beschrieben wird das Aktivieren der Funktionalität im Projekt und das Arbeiten mit DTMs.	X	X
/29/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS Projektkonvertierung DOK-XLCMLC-PROCONV*V13-APRS-DE-P, R911336365 Diese Dokumentation beschreibt die Projektkonvertierung von IndraLogic 04VRS und IndraMotion MLC04VRS auf IndraWorks Version 12 mit IndraLogic 2G. Es wird insbesondere auf die Änderungen im Bereich Motion und SPS eingegangen.	X	X
/28/	Rexroth IndraMotion MLC 13VRS Inbetriebnahme DOK-MLC***-STARTUP*V13-CORS-DE-P, R911336307 Das vorliegende Handbuch beschreibt die Inbetriebnahme- und Servicearbeitsschritte für das System IndraMotion MLC. Es beinhaltet Checklisten für häufig durchzuführende Arbeiten und eine ausführliche Beschreibung der Arbeitsschritte.	X	

Tab. 1-1: XLC/MLC-Dokumentationsübersicht

Zu dieser Dokumentation

Motion		MLC	XLC
/23/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS PLCopen-Bibliotheken DOK-XLCMLC-FUNLIB**V13-LIRS-DE-P, R911336359 Diese Dokumentation beschreibt die Funktionsbausteine, Funktionen und Datentypen der Bibliotheken RIL_CommonTypes, ML_Base und ML_PLCopen für die IndraLogic XLC/IndraMotion MLC. Die Dokumentation enthält ferner die Fehlerreaktion der Bausteine.	X	X
/27/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS Generic Application Template DOK-XLCMLC-TF*GAT**V13-APRS-DE-P, R911336369 Dieses Handbuch stellt dem IndraLogic SPS-Programmierer ein strukturiertes Template, welches zum Hinzufügen und Bearbeiten des SPS-Programmiercodes verwendet werden kann, zur Verfügung. Beschrieben werden das Template selbst, der Template Wizard und Beispielapplikationen.	X	X
/31/	Rexroth IndraMotion MLC 13VRS RCL-Programmieranleitung DOK-MLC***-RCL*PRO*V13-APRS-DE-P, R911336296 Das vorliegende Handbuch informiert über die Robot-Control V1. Die Programmiersprache RCL (Robot Control Language) steht dabei im Vordergrund. Beschrieben werden die Programmstruktur, Variablen, Funktionen, Bewegungsanweisungen sowie die notwendigen Parameter des Systems.	X	
/32/	Rexroth IndraMotion MLC 13VRS Robot-Control V2 DOK-MLC***-ROCO***V13-RE01-DE-P, R911341034 Die Dokumentation informiert über die Robot-Control V2. Die Programmierung mittels PLCopen steht hierbei im Vordergrund. Beschrieben werden die Programmstruktur, Variablen, Funktionen, Bewegungsanweisungen sowie die notwendigen Parameter des Systems. Die Dokumentation enthält ferner die Beschreibung der Bibliothek ML_Robot.	X	
/21/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS Parameter DOK-XLCMLC-PARAM***V13-RERS-DE-P, R911336363 Diese Dokumentation beschreibt die Parameter der Systeme XLC/MLC, sowie das Wechselspiel zwischen Parametrieren und Programmieren. Beschrieben werden die Achs-Parameter, die Steuerungs-Parameter, die Kinematik-Parameter, die Messtaster- und die Nockenschaltwerks-Parameter.	X	X
/10/	Rexroth IndraDrive Firmware für Antriebsregelgeräte MPH-, MPB-, MPD-, MPC-07 DOK-INDRV*-MP*-07VRS**-FKRS-DE-P, R911328669 Diese Dokumentation beschreibt alle funktionellen Eigenschaften der IndraDrive-Firmware in den Varianten MPH-07, MPB-07, MPD-07 und MPC-07.		
/11/	Rexroth IndraDrive MPx-17 Funktionen DOK-INDRV*-MP*-17VRS**-APRS-DE-P, R911331235 Diese Dokumentation beschreibt alle funktionellen Eigenschaften der IndraDrive-Firmware in den Varianten MPB-17, MPM-17, MPC-17 und MPE-17.		

Tab. 1-2: XLC/MLC-Dokumentationsübersicht

Zu dieser Dokumentation

Feldbusse		MLC	XLC
/4/	Rexroth IndraWorks 13VRS Feldbusse DOK-IWORKS-FB*****V13-APRS-DE-P, R911336871 Diese Dokumentation dient der Beschreibung der unterstützten Feldbusse und ihrer Diagnose-Funktionsbausteine.	X	X
/4a/	Rexroth IndraWorks 13VRS Feldbus-Bibliotheken DOK-IWORKS-FB*LIB**V13-LIRS-DE-P, R911337856 Das vorliegende Handbuch beschreibt die Bibliotheken der Feldbusse: RIL_ProfibusDP_02, RIL_ProfibusDP Slave, RIL_ProfinetIO, RIL_ProfinetIODevice, RIL_EtherNet/IP Adapter, RIL_MappingList, RIL_SERCOSIII, RIL_Inline, einschließlich deren Diagnosen, sowie die Fehlerreaktion der Bausteine.	X	X
/63/	sercos Systemhandbuch für I/O-Geräte DOK-CONTRL-ILS3*****-APRS-DE-P, R911333511 Diese Dokumentation beschreibt die Projektierung, Parametrierung, Inbetriebnahme und Diagnose von I/O-Geräten mit sercos Schnittstelle.	X	X

Tab. 1-3: XLC/MLC-Dokumentationsübersicht

HMI		MLC	XLC
/8/	Rexroth IndraWorks 13VRS HMI DOK-IWORKS-HMI*****V13-APRS-DE-P, R911336873 Diese Dokumentation beschreibt die Funktionen, die Konfiguration und die Bedienung der Bedienoberflächen IndraWorks HMI Engineering und IndraWorks HMI Operation.	X	X
/6/	Rexroth IndraWorks 13VRS WinStudio DOK-IWORKS-WINSTUD*V13-APRS-DE-P, R911336881 Diese Dokumentation beschreibt die Installation der Software, das Arbeiten mit WinStudio und das Erstellen und den Betrieb von Applikationen.	X	X
/50/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS HMI-Anbindung DOK-XLCMLC-HMI*****V13-APRS-DE-P, R911336361 Dieses Dokument beschreibt die von der IndraLogic XLC und IndraMotion MLC unterstützten Visualisierungssysteme sowie deren Anbindung.	X	X

Tab. 1-4: XLC/MLC-Dokumentationsübersicht

SPS		MLC	XLC
/3/	Rexroth IndraWorks 13VRS IndraLogic 2G Programmieranleitung DOK-IWORKS-IL2GPRO*V13-APRS-DE-P, R911336875 Diese Dokumentation beschreibt das SPS-Programmierungswerkzeug IndraLogic 2G und seine Verwendung. Beschrieben werden die Basisverwendung, Erste Schritte, die Visualisierung, Menüpunkte und die Editoren.	X	X
/33/	Rexroth IndraWorks 13VRS Basis-Bibliotheken IndraLogic 2G DOK-IL*2G*-BASLIB**V13-LIRS-DE-P, R911336284 Das vorliegende Handbuch beschreibt die systemübergreifenden SPS-Bibliotheken.	X	X

Tab. 1-5: XLC/MLC-Dokumentationsübersicht

Zu dieser Dokumentation

Technologie		MLC	XLC
/30/	Rexroth IndraMotion MLC 13VRS Technologie-Bibliotheken DOK-MLC***-TF*LIB**V13-LIRS-DE-P, R911336323 Diese Dokumentation beschreibt die Funktionsbausteine, Funktionen und Datentypen der Bibliotheken "ML_TechInterface.library", "ML_TechMotion.library", "RMB_TechCam.library" und "ML_TechBase.library" sowie Bibliotheken zur Wicklerfunktionalität, Registerreglerfunktionalität und Querschneiderfunktionalität.	X	
/60/	Rexroth IndraMotion MLC 13VRS RegisterControl (Bibliothek) DOK-MLC***-REGI*CO*V13-LIRS-DE-P, R911336305 Diese Dokumentation beschreibt die Ein- und Ausgänge der einzelnen Funktionsbausteine und gibt Hinweise für deren Verwendung.	X	
/62/	Rexroth IndraMotion MLC 13VRS RegisterControl (Anwendungsbeschreibung) DOK-MLC***-REGI*CO*V13-APRS-DE-P, R911336303 Dieses Dokument beschreibt die Anwendung der integrierten Registerregelung für eine Tiefdruckmaschine. Beschrieben werden die Komponenten des Markenstromsensors, die HMI-Anwendung und die Fehlerbehebung. Diese Anleitung dient dazu, die Registerregelung zu bedienen und auf Fehler zu reagieren sowie Diagnosen abzufragen. Die Zielgruppe für diese Dokumentation sind Maschineneinrichter und Maschinenbediener.	X	
/49/	Rexroth IndraMotion MLC 13VRS Anwendung Wicklerfunktionen DOK-MLC***-TF*WIND*V13-APRS-DE-P, R911336325 Diese anwendungsbezogene Systemdokumentation beschreibt die Anwendung der Wickler-Technologiefunktionen.	X	

Tab. 1-6: XLC/MLC-Dokumentationsübersicht

Hardware		MLC	XLC
/1/	Rexroth IndraControl L45/L65/L85 Steuerung DOK-CONTRL-ICL45L65L85-PRRS-DE-P, R911332115 Diese Dokumentation dient der Beschreibung der Steuerungen IndraControl L45/L65/L85.	X	X
/2/	Rexroth IndraControl L25 DOK-CONTRL-IC*L25*****-PRRS-DE-P, R911328473 Diese Dokumentation dient der Beschreibung der Steuerungen IndraControl L25.	X	X
/24/	Rexroth IndraControl Lxx 13VRS Funktionsmodule DOK-CONTRL-FM*LXX**V13-APRS-DE-P, R911336282 Diese Dokumentation dient der Beschreibung aller Funktionsmodule der Lxx Steuerungen inkl. Engineering und Diagnose.	X	X
/12/	Rexroth IndraDrive Antriebsregelgeräte MPx-02 bis MPx-07 DOK-INDRV*-GEN-**VRS*-PARS-DE-P, R911297316 Diese Dokumentation enthält die Beschreibung aller, in der Firmware für Antriebsregelgeräte der IndraDrive-Familie implementierten Parameter. Sie unterstützt die Parametrierung der Antriebsregelgeräte.		

Zu dieser Dokumentation

Hardware		MLC	XLC
/13/	Rexroth IndraDrive MPx-02 bis MPx-07 und HMV DOK-INDRV*-GEN-**VRS**-WARS-DE-P, R911297318 Diese Dokumentation enthält die Beschreibung aller in den folgenden Firmwares implementierten Diagnosen: - Antriebsregelgerät-Firmwares MPx-02 bis MPx-08 und - Firmwares der Versorgungsgeräte des Typs "HMV". Sie unterstützt den Maschinenbediener und den Anlagenprogrammierer bei der Fehlerbehebung.		
/35/	Rexroth IndraDrive Antriebsregelgeräte Steuerteile CSB01, CSH01, CDB01 DOK-INDRV*-CSH*****-PRRS-DE-P, R911295011 Diese Dokumentation ist eine Projektierungsbeschreibung zur Auswahl und Verwendung der Steuerteile CSB01, CDB01 und CSH01 für Antriebsregelgeräte der Produktfamilien Rexroth IndraDrive M und Rexroth IndraDrive C.		

Tab. 1-7: XLC/MLC-Dokumentationsübersicht

Diagnose und Service		MLC	XLC
/26/	Rexroth IndraWorks 13VRS IndraMotion Service Tool DOK-IWORKS-IMST****V13-RERS-DE-P, R911337706 Die Dokumentation beschreibt das IndraMotion Service Tool (IMST). Es ist ein webbasiertes Diagnosewerkzeug, das zum Zugriff auf die Steuerungssysteme IndraMotion MLC L25, L45 oder L65 sowie MLP über eine Ethernet-Hochgeschwindigkeitsverbindung dient. Das IMST ermöglicht OEMs, Endanwendern und Kundenbetreuern von überall auf ein System zuzugreifen und Ferndiagnosen zu stellen. Voraussetzung ist ein PC mit der Internet Explorer Version 6 oder 7.	X	X
/22/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS Diagnosen DOK-XLCMLC-DIAG****V13-RERS-DE-P, R911336357 Diese Dokumentation enthält alle in den Steuerungssystemen IndraLogic XLC und IndraMotion MLC implementierten Steuerungsparameter.	X	X

Tab. 1-8: XLC/MLC-Dokumentationsübersicht

Systemübersicht		MLC	XLC
/48/	Rexroth IndraMotion for Printing 13VRS Systemübersicht DOK-IM*PR*-SYSTEM**V13-PRRS-DE-P, R911336288 Das vorliegende Handbuch beschreibt das Produkt IndraMotion for Printing. Es stellt die Steuerungs-, Antriebs- und IO-Systeme vor und gibt eine Einführung in die Inbetriebnahme und Programmierung.	X	
/48/	Rexroth IndraMotion for Packaging 13VRS Systemübersicht DOK-IM*PA*-SYSTEM**V13-PRRS-DE-P, R911336286 Das vorliegende Handbuch beschreibt das Produkt IndraMotion for Packaging. Es stellt die Steuerungs-, Antriebs- und IO-Systeme vor und gibt eine Einführung in die Inbetriebnahme und Programmierung.	X	

Zu dieser Dokumentation

Systemübersicht		MLC	XLC
/9/	Rexroth IndraMotion MLC 13VRS Systemübersicht DOK-MLC***-SYSTEM**V13-PRRS-DE-P, R911336309 Diese Dokumentation stellt eine Übersicht der möglichen Hardware-/Softwarekomponenten des Automationssystems IndraMotion MLC in der genannten Version dar. Die Dokumentation dient als Auswahlhilfe zur Zusammenstellung eines Systems.	X	
/9/	Rexroth IndraLogic XLC 13VRS Systemübersicht DOK-XLC***-SYSTEM**V13-PRRS-DE-P, R911336353 Diese Dokumentation stellt eine Übersicht der möglichen Hardware-/Softwarekomponenten des Automationssystems IndraLogic XLC in der genannten Version dar. Die Dokumentation dient als Auswahlhilfe zur Zusammenstellung eines Systems.		X

Tab. 1-9: XLC/MLC-Dokumentationsübersicht

Erste Schritte		MLC	XLC
/25/	Rexroth IndraMotion MLC 13VRS Die ersten Schritte DOK-MLC***-F*STEP**V13-CORS-DE-P, R911336292 Diese Dokumentation beschreibt die "ersten Schritte" mit der IndraMotion MLC und RobotControl. Beschrieben werden die Hard- und Softwarevoraussetzungen und das Erstellen eines Projekts.	X	
/25/	Rexroth IndraLogic XLC 13VRS Die ersten Schritte DOK-XLC***-F*STEP**V13-CORS-DE-P, R911336349 Diese Dokumentation beschreibt die "ersten Schritte" mit der IndraLogic XLC. Beschrieben werden die Hard- und Softwarevoraussetzungen und das Erstellen eines Projekts.		X

Tab. 1-10: XLC/MLC-Dokumentationsübersicht

1.4 Bezeichnungen und Abkürzungen

Begriff	Erklärung
CANOpen	Feldbus
DeviceNet	Feldbus
Ethernet	Kommunikationsschnittstelle
IWE	IndraWorks Engineering
IWO	IndraWorks Operation
LWL	Lichtwellenleiter
NC	Numerical Control
ModbusTCP-Protokoll	Kommunikationsprotokoll, das auf einer Master/Slave- bzw. Client/Server-Architektur basiert (IEC 61158)
OEM	Original Equipment Manufacturer
PROFIBUS-DP	Feldbus
sercos	Sercos (SErial Realtime COmmunication System) interface ist eine weltweit genormte digitale Schnittstelle zur Kommunikation zwischen Steuerungen und Antrieben

Tab. 1-11: Verwendete Bezeichnungen und Abkürzungen

2 Visualisierungsgeräte

2.1 Übersicht

Rexroth IndraControl V ist das umfassende HMI-Geräteportfolio für individuelles Steuern, Bedienen und Beobachten in allen Industriebereichen. Mit skalierbarer Hard- und Software ist IndraControl V exakt auf Ihre maschinenspezifischen Anforderungen anpassbar.

IndraControl V integriert alle erforderlichen Funktionen für wirtschaftliches Automatisieren – angefangen vom komfortablen Bedienen und übersichtlichen Visualisieren über die integrierten Steuerungen bis hin zur Diagnose. In Verbindung mit den praxiserprobten Systemlösungen von Rexroth erhalten Sie so eine komplette Automatisierungslösung für:

- Transfermaschinen
- Druck- und Verarbeitungsmaschinen
- Nahrungsmittel- und Verpackungsmaschinen
- Umformmaschinen
- Holzbearbeitungsmaschinen
- Textilmaschinen
- Handling- und Montagesysteme

IndraControl V ist in folgenden Ausführungen erhältlich:

- controllerbasiert
 - [IndraControl VCP, Seite 17](#)
 - [IndraControl VCH, Seite 22](#)
 - [IndraControl VR21, Seite 23](#)
- Embedded-PC-basiert
 - [IndraControl VEP, Seite 31](#)
- PC-basiert
 - [IndraControl VPB und VDP, Seite 34](#)
 - [IndraControl VPP, Seite 38](#)

2.2 IndraControl VCP

2.2.1 Kurzbeschreibung

Die Kleinbedienfelder VCP sind Bedien- und Visualisierungsterminals zum Bedienen und zum Beobachten von Maschinen. Durch die kompakte Bauweise sind die Geräte besonders vielseitig einsetzbar. Die Kommunikation zu einer übergeordneten Steuerung erfolgt entweder über eine PROFIBUS DP- oder Ethernet-Schnittstelle.



Abb. 2-1: Kleinbedienfeld VCP 02

Visualisierungsgeräte



Abb. 2-2: Kleinbedienfeld VCP 05



Abb. 2-3: Kleinbedienfeld VCP 08

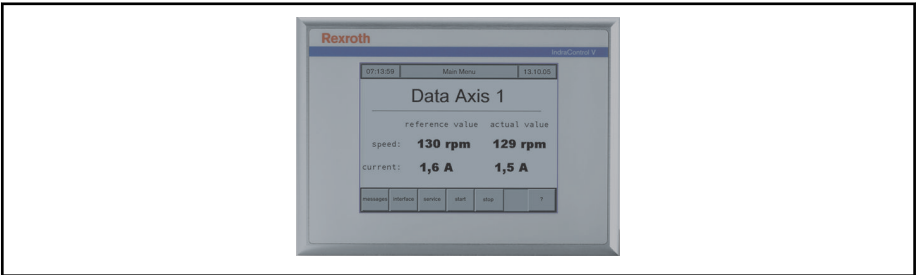


Abb. 2-4: Kleinbedienfeld VCP 11

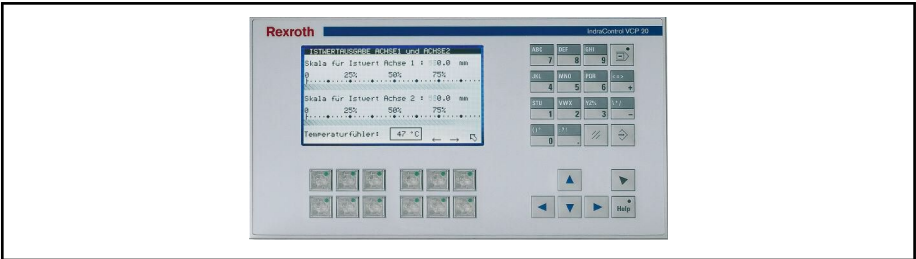


Abb. 2-5: Kleinbedienfeld VCP 20



Abb. 2-6: Kleinbedienfeld VCP 25

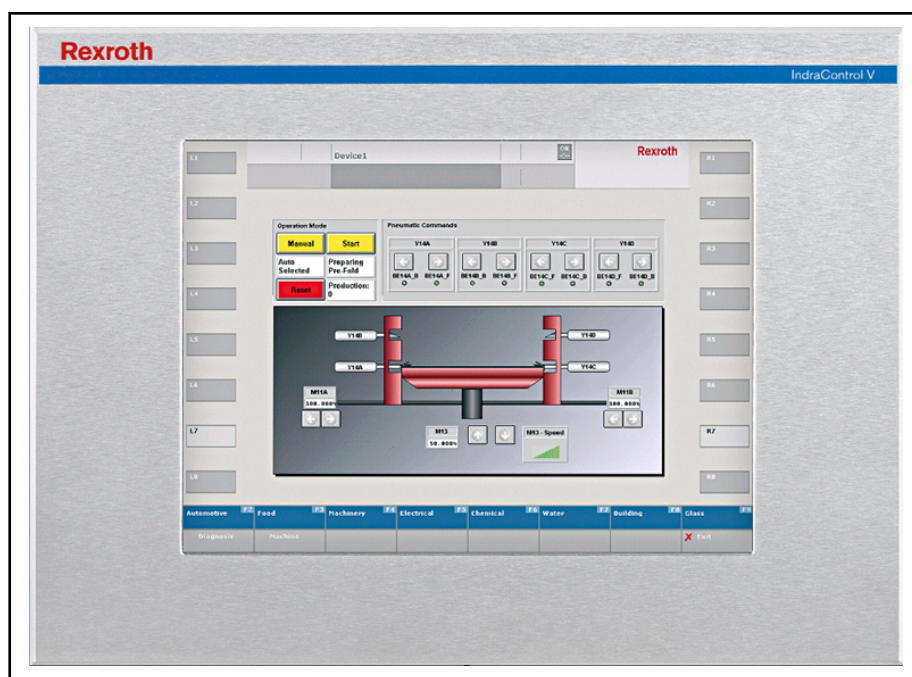


Abb. 2-7: Kleinbedienfeld VCP 35

2.2.2 Technische Daten

	VCP 02	VCP 05	VCP 08	VCP 11	VCP 20	VCP 25	VCP 35
Display	3" Grafikdisplay, 160 x 80 Pixel		3,8" Grafikdisplay, 320x240 Pixel	3,8" Grafikdisplay, 320 x 240 Pixel	5,7" Grafikdisplay, 320 x 240 Pixel	5,7" Grafikdisplay, 320 x 240 Pixel	10,4" Grafikdisplay, 640 x 480 Pixel
Frontplatte	100 x 148 mm	168 x 120 mm	159 x 209 mm	96 x 130 mm	160 x 300 mm	180 x 234	249 x 328 mm
Funktions-/Systemtasten	4 / 7	6 / 24	15/24	Touchscreen	16 / 22	Touchscreen	Touchscreen
Prozessor	ARM 200 MHz						
Applikationsspeicher	3 MB						
Schnittstellen	1x Ethernet (10/100 Base T) 2x USB Host 2.0 Profibus DP Interface RS232/RS485 (Option)						
Spannungsversorgung	24 V DC						

Tab. 2-1: Technische Daten Kleinbedienterminal VCP

2.2.3 Ausführungen

Die Kleinbedienfelder VCP sind in folgenden Ausführungen erhältlich:

Visualisierungsgeräte

Type	Bemerkungen
VCP02.2DRN-003-NN-NN-PW	Tasten, Kommunikation über Ethernet
VCP02.2DRN-003-PB-NN-PW	Tasten, Kommunikation über PROFIBUS DP
VCP02.2DRN-003-SR-NN-PW	Tasten, Kommunikation über seriellen Schnittstellen
VCP05.2DSN-003-NN-NN-PW	Tasten, Kommunikation über Ethernet
VCP05.2DSN-003-PB-NN-PW	Tasten, Kommunikation über PROFIBUS DP
VCP05.2DSN-003-SR-NN-PW	Tasten, Kommunikation über seriellen Schnittstellen
VCP08.2DTN-003-NN-NN-PW	Tasten, Kommunikation über Ethernet
VCP08.2DTN-003-PB-NN-PW	Tasten, Kommunikation über PROFIBUS DP
VCP08.2DTN-003-SR-NN-PW	Tasten, Kommunikation über seriellen Schnittstellen
VCP11.2DWN-003-NN-NN-PW	Touchscreen, Kommunikation über Ethernet
VCP11.2DWN-003-PB-NN-PW	Touchscreen, Kommunikation über PROFIBUS DP
VCP11.2DWN-003-SR-NN-PW	Tasten, Kommunikation über seriellen Schnittstellen
VCP11.2ECN-003-NN-NN-PW	TFT
VCP11.2ECN-003-PB-NN-PW	TFT mit PROFIBUS DP-Slave
VCP20.2DUN-003-NN-NN-PW	Tasten, Kommunikation über Ethernet
VCP20.2DUN-003-PB-NN-PW	Tasten, Kommunikation über PROFIBUS DP
VCP20.2DUN-003-SR-NN-PW	Tasten, Kommunikation über seriellen Schnittstellen
VCP25.2DVN-003-NN-NN-PW	Touchscreen, Kommunikation über Ethernet
VCP25.2DVN-003-PB-NN-PW	Touchscreen, Kommunikation über PROFIBUS DP
VCP25.2DVN-003-SR-NN-PW	Tasten, Kommunikation über seriellen Schnittstellen
VCP35.2ECN-003-NN-NN-PW	Touchscreen, Kommunikation über Ethernet
VCP35.2ECN-003-PB-NN-PW	Touchscreen, Kommunikation über PROFIBUS DP
VCP35.2ECN-003-SR-NN-PW	Tasten, Kommunikation über seriellen Schnittstellen

Tab. 2-2: Ausführungen VCP-Geräte

2.2.4 Zubehör

Verbindungskabel

Die Kommunikation der Kleinbedienfelder VCP mit den Steuerungen erfolgt entweder über PROFIBUS DP oder Ethernet. Für PROFIBUS DP ist folgendes Kabel erhältlich:

Visualisierungsgeräte

Type	Bemerkung
IKB0034/000,0	Verbindung zwischen Steuerung und VCP; Kabel einseitig konfektioniert, variable Länge

Tab. 2-3: PROFIBUS-Verbindungskabel

Für die Kommunikation über Ethernet sind folgende Kabel erhältlich:

Type	Bemerkung
RKB0008/002,5	Bus - Kabel, konfektioniert Ethernetkabel, 100-Base-T, CAT 6+ konfektioniert, UL, schleppkettenfähig Kabellänge 2,5 m
RKB0008/005,0	wie oben, Kabellänge 5 m
RKB0008/010,0	wie oben, Kabellänge 10 m
RKB0008/015,0	wie oben, Kabellänge 15 m
RKB0008/020,0	wie oben, Kabellänge 20 m
RKB0008/025,0	wie oben, Kabellänge 25 m
RKB0008/030,0	wie oben, Kabellänge 30 m
RKB0008/035,0	wie oben, Kabellänge 35 m
RKB0008/040,0	wie oben, Kabellänge 40 m
RKB0008/050,0	wie oben, Kabellänge 50 m

Tab. 2-4: Ethernet-Verbindungskabel

Batterie-Kit

Ein Batterie-Kit ist unter folgender Bezeichnung erhältlich:

Type	Bemerkung
VAS04.1-001-002-NN	Batterie-Kit

Tab. 2-5: Batterie-Kit für VCP-Geräte

2.2.5 Dokumentation

Für die ausführliche Beschreibung der Kleinbedienfelder VCP stehen folgende Dokumentationen zur Verfügung:

- DOK-SUPPL*-VCP*02.2***-PRxx-DE-P
- DOK-SUPPL*-VCP*05.2***-PRxx-DE-P
- DOK-SUPPL*-VCP*08.2***-PRxx-DE-P
- DOK-SUPPL*-VCP*11.2***-PRxx-DE-P
- DOK-SUPPL*-VCP*20.2***-PRxx-DE-P
- DOK-SUPPL*-VCP*25.2***-PRxx-DE-P
- DOK-SUPPL*-VCP*35.2***-PRxx-DE-P

xx - Ausgabe

Visualisierungsgeräte

2.3 IndraControl VCH

2.3.1 Kurzbeschreibung

Bedienen, Einrichten, Parametrieren und Diagnostizieren, ganz gleich von welchem Ort – IndraControl VCH 08 macht es über Ethernet TCP/IP möglich. Beim Anschließen und Abstecken wird die Stopp-Funktion automatisch überbrückt und gewährleistet störungsfreies Arbeiten.

Niedriges Gewicht und optimales Design ermöglichen ermüdungsfreies Arbeiten und hohen Bedienkomfort.

Der integrierte 3-stufige Zustimmschalter und der 2-kreisig ausgeführte Stopp-Taster werden durch die Anschaltbox VAC 30 eingeschleift und sorgen für maximale Sicherheit.

Besondere Merkmale:

- Sicherheitsfunktionen für Mensch und Maschine
- Haltegriffe mit integrierten Zustimmung- und Stopp-Tastern
- robuste Konstruktion für industriellen Einsatz
- ergonomisch optimiert für sicheres ermüdungsfreies Handling
- brillantes 3,8“-Grafikdisplay und eine universell nutzbare Folientastatur für komfortables Bedienen und Visualisieren



Abb. 2-8: IndraControl VCH 08

2.3.2 Einsatzbereich

Das VCH 08 ist für den Einsatz in Handlings-Applikationen konzipiert.

2.3.3 Technische Daten

	VCH 08
--	--------

Visualisierungsgeräte

Display	3,8" Graustufen, 320 x 240 Pixel
Ausführung Frontplatte	15 Funktionstasten 25 Systemtasten 40 Bedientasten
Zustimmtaster	Anzahl: 2 2-kreisig, 3-stufig
Stopptaster	Anzahl: 1 2-kreisig
Handräder	Anzahl: 1 (Option)
Override-Potentiometer	Anzahl 1: (Option)
Prozessor	CPU PXA 270/416 MHZ (Standard)
Speicher	64 MB (Anwendung) 64 MB (Flash)

Tab. 2-6: Technische Daten VCH 08

2.3.4 Ausführungen

Das VCH 08 ist in folgenden Ausführungen erhältlich:

Type	Bemerkungen
VCH08.1EAB-064ET-A1D-064-CS-E2-PW	mit Handrad (0,5 m Ethernet Steuerkabel)
VCH08.1EAB-064ET-A1D-064-FS-B2-PW	mit Nothalt-Taster (0,5 m Ethernet Steuerkabel)
VCH08.1EAB-064ET-A1D-064-CS-E4-PW	8 m Ethernet Steuerkabel
VCH08.1EAB-064ET-A1D-064-DS-E4-PW	mit Handrad (8 m Ethernet Steuerkabel)

Tab. 2-7: Ausführungen VCH 08

2.3.5 Zubehör

Für das VCH 08 ist folgendes Zubehör erhältlich:

Type	Bemerkungen
VAS01.1-002-NNN-NN	Wandhalterung VAS 01
VAC30.2N-NN	VAC 30

Tab. 2-8: Zubehör für VCH 08

2.3.6 Dokumentation

Für die ausführliche Beschreibung des Handbedienterminals VCH 08 steht die folgende Dokumentation zur Verfügung:

- DOK-SUPPL*-VCH*08.1***-PRxx-DE-P

xx - Ausgabe

2.4 IndraControl VR21

2.4.1 Kurzbeschreibung

Die Kleinbedienfelder VR21 sind Bedien- und Visualisierungsterminals zum Bedienen und zum Beobachten von Maschinen. Durch die kompakte Bauweise sind die Geräte besonders vielseitig einsetzbar. In Verbindung mit dem

Visualisierungsgeräte

Steuerungssystem XLC/MLC sind die Geräte mit einer Bildschirmdiagonale von 4,3", 7" und 9" verfügbar. Die Projektierung der Bedienbilder erfolgt mit Hilfe von Rexroth WinStudio. Die Kommunikation zu einer übergeordneten Steuerung erfolgt über eine Ethernet-Schnittstelle.

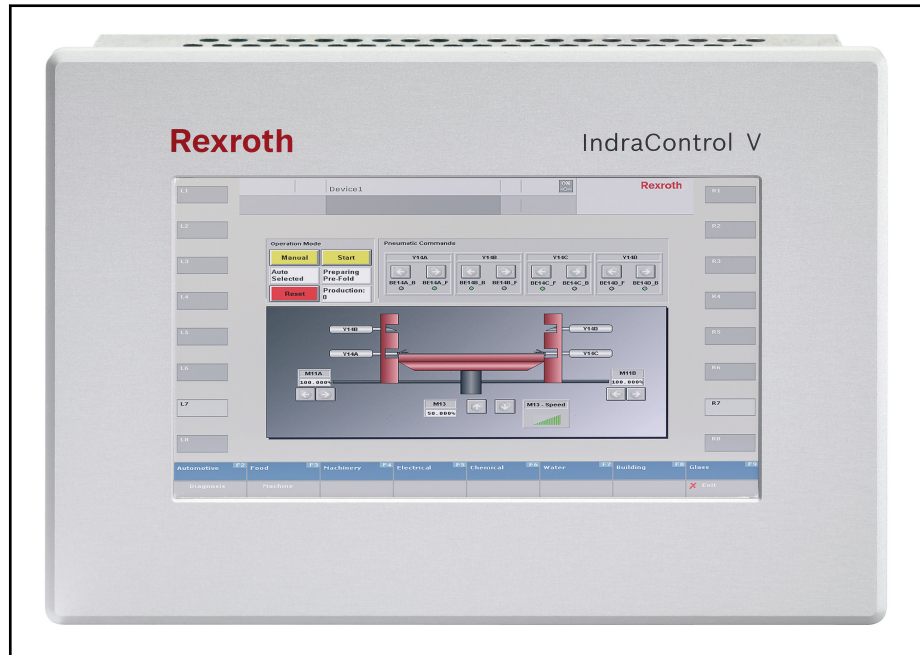


Abb. 2-9: Kleinbedienfeld VR2104

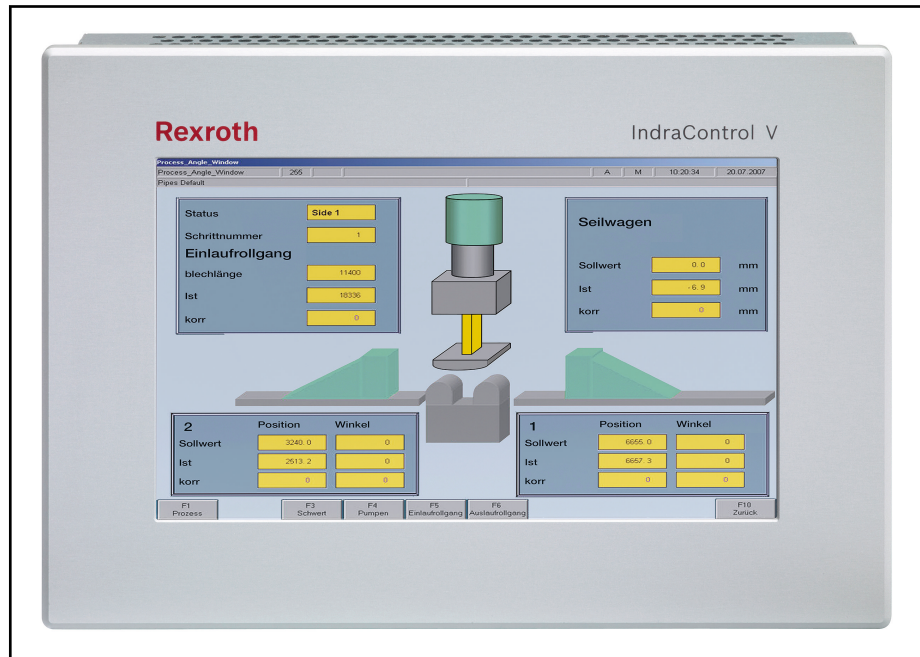


Abb. 2-10: Kleinbedienfeld VR2107

Visualisierungsgeräte

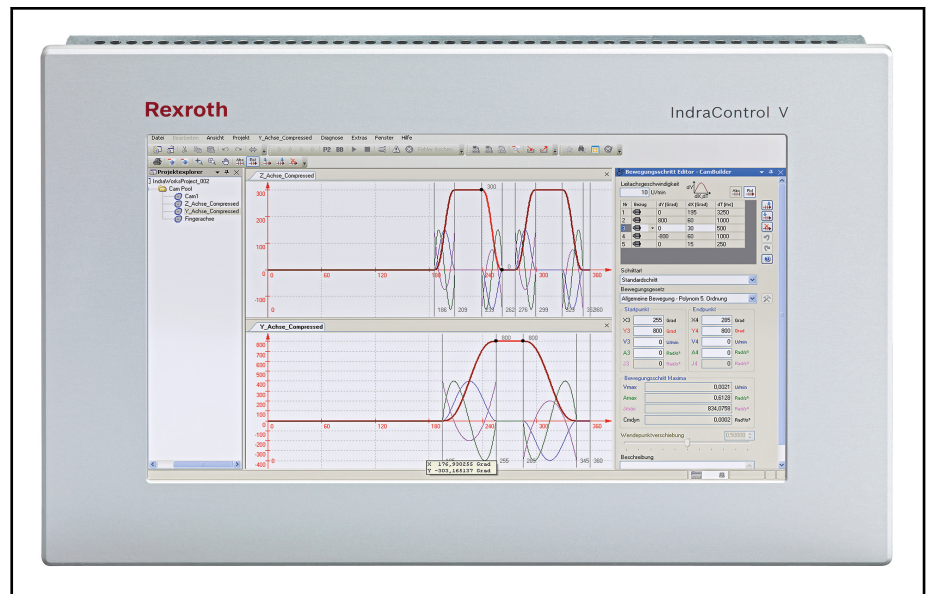


Abb. 2-11: Kleinbedienfeld VR2109

2.4.2 Varianten

Die IndraControl VR 21-Geräte sind in unterschiedlichen Varianten verfügbar. Die Geräte unterscheiden sich durch ihre Displaygröße und durch die Touch-Technologie (Single- und Multitouch).

Typenschlüssel	Display	Auflösung	Touch-Technologie
VR2104.01-00-01-N2-NNN-AA Mat.-Nr. R911340500	4,3"	480 × 272 Pixel	Resistiv, Singletouch
VR2107.01-00-01-N2-NNN-AA Mat.-Nr. R911340503	7"	800 × 480 Pixel	Resistiv, Singletouch
VR2107.01-00-01-N2-NNN-CA Mat.-Nr. R911340505	7"	800 × 480 Pixel	Kapazitiv, Multitouch
VR2109.01-00-01-N2-NNN-AA Mat.-Nr. R911340051	9"	800 × 480 Pixel	Resistiv, Singletouch
VR2109.01-00-01-N2-NNN-CA Mat.-Nr. R911340506	9"	800 × 480 Pixel	Kapazitiv, Multitouch

Tab. 2-9: Varianten des IndraControl VR 21

Visualisierungsgeräte

2.4.3 Technische Daten

VR 21 Singletouch

	VR2104-Singletouch	VR2107-Singletouch	VR2109-Singletouch
Display	107,95 mm TFT (4,3") 480 × 272 Pixel 65536 Farben Helligkeit 450 cd/m ² Anzeigefläche 53,8 × 95 (H × B)	177,8 mm TFT (7") 800 × 480 Pixel 262144 Farben Helligkeit 350 cd/m ² Anzeigefläche 91,4 × 152,4 (H × B)	228,6 mm TFT (9") 800 × 480 Pixel 16,77 Millionen Farben Helligkeit 360 cd/m ² Anzeigefläche 117 × 195 (H × B)
Touch-Technik	Analog resistiv, 4-Draht-Technik		
Touch-Aktivierungsdruck	15 g (Standard) mit R8 HS60 Silikonkautschuk		
Gehäuse	Stahlblech, verzinkt		
Frontplattenmaterial	Aluminium, gebürstet, natur eloxiert		
Schutzart	Frontplatte IP 65 nach DIN EN 60 529, Rückseite IP 20		
Zentraleinheit	ARM Cortex™-A8, 800 MHz mit Echtzeituhr		
Flash-Speicher	256 MByte		
LPDDR-Speicher	512 MByte		
Spannungsversorgung	DC 24 V (verwenden Sie ein 24-V-Industrienetzteil nach DIN EN 60742, Klassifikation VDE 0551, zum Beispiel das Netzteil VAP01.1H-W23-024-010-NN mit der Materialnummer R911171065)		
Stromaufnahme	0,2 A (typisch bei 24 V) 0,3 A (maximal)	0,3 A (typisch bei 24 V) 0,4 A (maximal)	0,6 A (typisch bei 24 V) 0,7 A (maximal)
Anschlusswert	4,8 W	7,2 W	14,4 W
Sicherung	Halbleitersicherung, selbstrückstellend		
Verpolschutz	Integriert		
USB	Pro USB-Buchse maximal 500 mA, Summenstrom an allen USB-Buchsen maximal 1 A		
Gewicht	ca. 0,6 kg	ca. 0,8 kg	ca. 1,3 kg

Tab. 2-10: Technische Daten des VR 21 Singletouch

VR 21 Multitouch

	VR2107-Multitouch	VR2109-Multitouch
Display	177,8 mm TFT (7") 800 × 480 Pixel 262144 Farben Helligkeit 350 cd/m ² Anzeigefläche 91,4 × 152,4 (H × B)	228,6 mm TFT (9") 800 × 480 Pixel 16,77 Millionen Farben Helligkeit 360 cd/m ² Anzeigefläche 117 × 195 (H × B)
Touch-Technik	Projiziert-kapazitiv, mit Finger bedienbar	
Touch-Aktivierungsdruck	15 g (Standard) mit R8 HS60 Silikonkautschuk	
Gehäuse	Stahlblech, verzinkt	
Frontplattenmaterial	Aluminium, gebürstet, natur eloxiert	

	VR2107-Multitouch	VR2109-Multitouch
Oberfläche der Frontplatte	Glas	
Schutzart	Frontplatte IP 65 nach DIN EN 60 529 Rückseite IP 20	
Zentraleinheit	ARM Cortex™-A8, 800 MHz mit Echtzeituhr	
Flash-Speicher	256 MByte	
LPDDR-Speicher	512 MByte	
Spannungsversorgung	DC 24 V (verwenden Sie ein 24-V-Industrienetzteil nach DIN EN 60742, Klassifikation VDE 0551, zum Beispiel das Netzteil VAP01.1H-W23-024-010-NN mit der Materialnummer R911171065)	
Stromaufnahme	0,4 A (typisch bei 24 V) 0,5 A (maximal)	0,7 A (typisch bei 24 V) 0,8 A (maximal)
Anschlusswert	9,6 W	16,8 W
Sicherung	Halbleitersicherung, selbstrückstellend	
Verpolschutz	Integriert	
USB	Pro USB-Buchse maximal 500 mA, Summenstrom an allen USB-Buchsen maximal 1 A	
Gewicht	ca. 0,8 kg	ca. 1,3 kg

Tab. 2-11: Technische Daten des VR 21 Multitouch

2.4.4 Gehäuse - und Einbaumaße

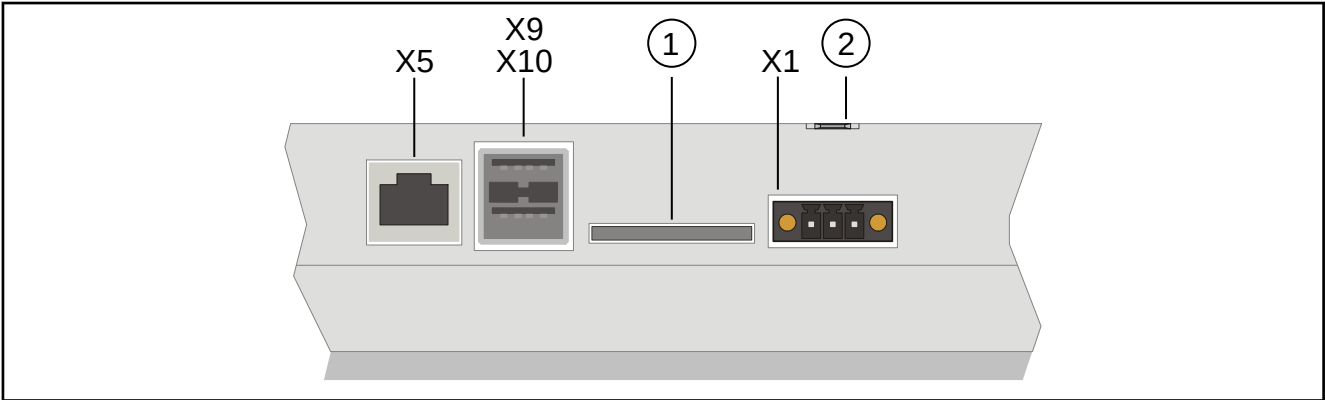
Variante	Display	Gerätehöhe	Gerätebreite	Einbautiefe	Montageausschnitt
VR2104	4,3 "	100 mm	140 mm	42 mm	H: 92 mm, B: 132 mm
VR2107	7 "	150 mm	211 mm	46,3 mm	H: 142 mm, B: 203 mm
VR2109	9 "	178 mm	263 mm	52,3 mm	H: 170 mm, B: 255 mm

Tab. 2-12: Gehäuse - und Einbaumaße der VR 21xx

Visualisierungsgeräte

2.4.5 Schnittstellen

Schnittstellenansicht



①SD-Speicherkarte

②Funktionserde

Abb. 2-12: Schnittstellen am IndraControl VR 21

Schnittstellenübersicht

Bezeichnung am Gehäuse	Anschluss typ	Steckertyp, eingebaut	Gegenstecker oder Leitung (von extern)
X1	DC-24-V-Spannungsversorgung	3-polig, Phoenix MINI COMBICON	3-polig, FK-MCP 1,5/3-ST-3,5
X9, X10 USB Host	2 USB-Schnittstellen	USB-Buchse, 4-polig, Typ A	USB-Stecker, 4-polig, Typ A
X5	Ethernet 10/100 Base-T	RJ45-Buchse	RJ45-Stecker
 Funktions- erde	Funktionserde	Flachsteckzunge	Flachsteckhülse
SD-Spei- cherkarte	Einschub für SD- und SDHC-Speicherkarte	-	

Tab. 2-13: Schnittstellen

HINWEIS

Funktionsstörungen durch mangelhafte Schirmung!

Verwenden Sie nur geschirmte Kabel und metallische oder leitende Stecker- oder Kupplungsgehäuse mit großflächiger Schirmauflage.

DC-24-V-Spannungsversorgung

An die Anschlussklemme können Sie Kabel mit einem Leiterquerschnitt von 0,75 bis 2,5 mm² anschließen.

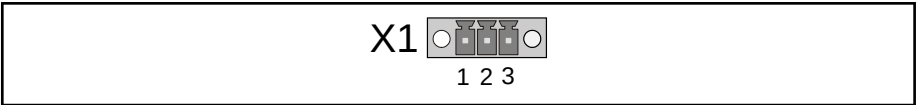


Abb. 2-13: Spannungsversorgungsstecker X1

Visualisierungsgeräte

Pin	Bezeichnung	Funktion
1		Fremdspannungsarme Erde, Funktionserde (FE)
2	0 V	Versorgungsspannung 0 V (GND)
3	24 VDC	Versorgungsspannung 24 VDC

Verwenden Sie für die Spannungsversorgung ein 24-V-Industrienetzteil nach DIN EN 60742, Klassifikation VDE 0551, zum Beispiel das VAP01.1H-W23-024-010-NN mit der Materialnummer R911171065.

USB-Schnittstellen X9, X10

Am Bediengerät sind zwei USB-Host-Schnittstellen vorhanden.

Betrieb der USB-Anschlüsse am Bediengerät



Bei Verwendung nicht industrietauglicher Eingabegeräte (z. B. Tastatur, Maus) muss mit einer eingeschränkten Betriebssicherheit gerechnet werden. Dazu zählen auch Eingabegeräte, die für den Heim- und Büroeinsatz deklariert sind.



Schließen Sie nur USB-Geräte an, die auch die USB2.0-Spezifikation erfüllen.



Die Spezifikation eines geeigneten Kabels entnehmen Sie der "Universal Serial Bus Specification Rev. 2.0".
Die maximale Kabellänge beträgt 2,5 m.



Nicht alle USB-Geräte werden erkannt.
Das Betriebssystem unterstützt nicht alle USB-Geräte. Geräte, die einen speziellen USB-Treiber benötigen, der nicht im System eingebunden ist, können an den USB-Schnittstellen nicht betrieben werden.

Ethernet-Schnittstelle X5

Am Bediengerät befindet sich eine 10/100Base-T Ethernet-Schnittstelle.



Verwenden Sie Twisted-Pair-Kabel der Kategorie 5 oder 6 (CAT 5 oder 6). Die maximale Kabellänge beträgt 100 m.

Ethernet-Schnittstelle – Status- und Diagnoseanzeigen

LED ACT/LNK (Grün)

- An: Verbindung zum Netzwerk vorhanden
- Aus: Keine Verbindung
- Blinkt: Kommunikation läuft

LED SPD 10/100 (Gelb)

- An: 100 MBit/s Modus
- Aus: 10 MBit/s Modus oder Verbindung getrennt

Steckplatz für SD-Speicherkarte

Auf der Unterseite des Bediengeräts können Sie eine SD-Karte einstecken.

Visualisierungsgeräte



Bei Verwendung nicht industrietauglicher Hardware (z. B. Tastatur, Maus, Speicherkarte) in industriellen Bereichen muss mit einer eingeschränkten Betriebssicherheit gerechnet werden. Dazu zählt auch Hardware, die für den Heim- und Büroeinsatz deklariert ist.

Einsetzen der Speicherkarte

Schieben Sie die Speicherkarte so ein, dass die Vorderseite oben ist. Führen Sie die Karte ein, bis die Speicherkarte einrastet.

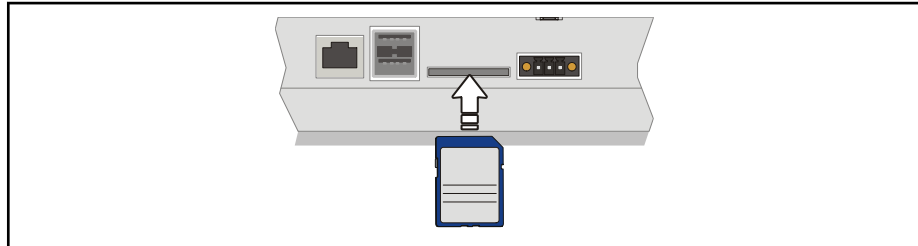


Abb. 2-14: Speicherkarte einföhren

Entfernen der Speicherkarte

Drücken Sie die Speicherkarte kurz bis zur Rastung in das Bediengerät. Entnehmen Sie die entriegelte Speicherkarte.

2.4.6 Bestellinformationen

Zubehör- und Ersatzteile

Bestellinformationen für Zubehör- und Ersatzteile finden Sie in Rexroth IndraControl VR 21 Bedienpanel, DOK-SUPPL*-VR21**..01**-ITRS-DE-P, R911339475.

Typenschlüssel VR21xx.01

Kurztext- spalte	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
Beispiel:	V	R	2	1	0	4	.	0	1	-	0	0	-	0	1	-	N	2	-	N	N	N	-	A	A	.	.	.
Gerätetyp 1																												
HMI..... = V																												
Gerätetyp 2																												
Box (vorkonfiguriert) = R																												
Leistungsklasse																												
Leistungsklasse S..... = 2																												
Prozessorvariante																												
Prozessorvariante 1..... = 1																												
Displaygröße (16:9)																												
4,3"..... = 04																												
7"..... = 07																												
9"..... = 09																												
Ausführung																												
HW-Ausführung..... = 01																												
Feldbus																												
Kein Feldbus..... = 00																												
Schnittstellen (Ethernet, USB...)																												
2 x USB Host + 1 x Ethernet 100 Mbit..... = 01																												
Remanenter Speicher																												
Nicht vorhanden..... = N																												
Massenspeicher (Festplatte, Flash, SSD)																												
Onboard Flash min.128 MB..... = 2																												
Hardware Konfiguration																												
Ohne Erweiterungsslot..... = NNN																												
Design & HMI-Displayeigenschaften																												
Single-Touch im Rexroth-Design = AA																												
Multi-Touch im Rexroth-Design = CA																												

Abb. 2-15: Typenschlüssel

2.4.7 Dokumentation

Für die ausführliche Beschreibung der Kleinbedienfelder VR21 steht folgende Dokumentationen zur Verfügung:

- DOK-SUPPL*-VR21**.01**-IT02-DE-P

2.5 IndraControl VEP

2.5.1 Kurzbeschreibung

Mit diesen Terminals bedienen Sie Ihre Maschine einfach und komfortabel über Touchscreen oder virtuelles Keyboard. In diesen PC-nahen Geräten werden ausschließlich Embedded-Komponenten für einen robusten Aufbau verwendet.

Mit der Vielzahl an Schnittstellen und Steckplätzen passen sich die IndraControl VEP-Geräte optimal an die Maschinen- und Anlagenerfordernisse an.

Besondere Merkmale:

- kompakte Geräteausführung für Schaltschrank- oder Tragarmmontage
- festplattenlose Hardware, ohne rotierende Medien
- Visualisieren, Bedienen und Beobachten mit durchgängiger Visualisierungssoftware WinStudio

Visualisierungsgeräte

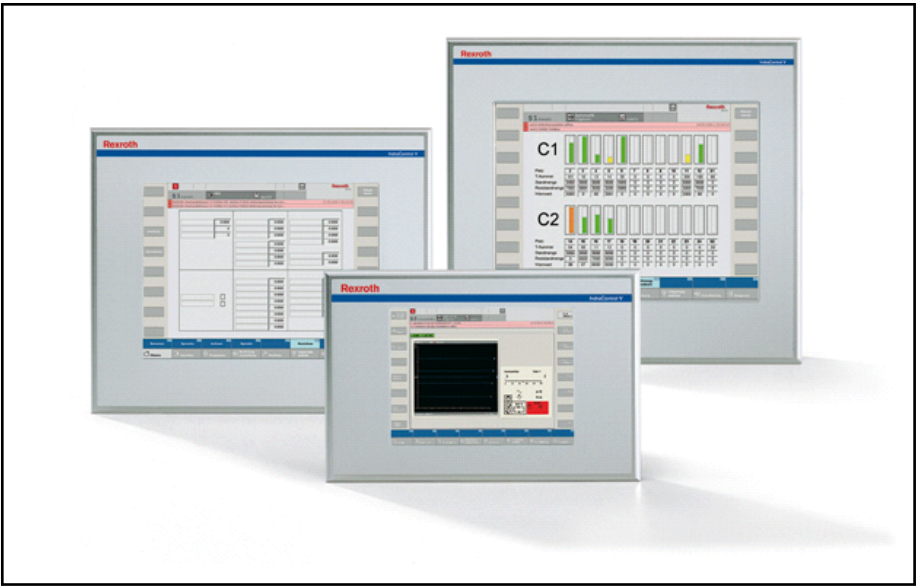


Abb. 2-16: IndraControl VEP-Varianten

2.5.2 Einsatzbereich

Die Standard-Industrie-Bedienterminals VEP werden vorwiegend für alle Aufgaben in der PC-basierten Automatisierung eingesetzt.

2.5.3 Technische Daten

	VEP 40.5	VEP 50.5
Display	12" Farbdisplay	15" Farbdisplay
Tastatur	über virtuelles Keyboard	
Modulplätze	--	
Schnittstellen	2 x USB 2.0 - Anschluss (Typ A) 2 x Ethernet-Anschluss (RJ 45, 10/100 Base-T)	
Feldbus	--	
Prozessor	Atom N455 1.66GHz	
Hauptspeicher	2 GB	
Compact-Flash	2 CF-Sockel / 8GB CF-Card bestückt	
Versorgungsspannung	24 V DC	
Betriebssystem	Windows XPe / Windows 7e	

Tab. 2-14: Technische Daten VEP 40.5 / 50.5

	VEP 30.4	VEP 40.4	VEP 50.4
Display	8,4" Farbdisplay	12" Farbdisplay	15" Farbdisplay
Tastatur	Touch-Bedienung	Touch-Bedienung oder Touch- und Tasten-Bedienung	Touch-Bedienung oder Touch- und Tasten-Bedienung
Modulplätze	- -		

Visualisierungsgeräte

Schnittstellen	2 x USB 2.0 - Anschluss (Typ A) 2 x (Intel Celeron) bzw. 1 x (Intel Atom) Ethernet-Anschluss (RJ 45, 10/100 Base-T)
Prozessor	Intel Celeron 600Mhz / Intel Atom 1,1 GHz
Hauptspeicher	512MB / 1 GB
Compact-Flash	2 CF-Sockel / 1GB CF-Card bestückt, oder 4 GB CF-Card bestückt
Versorgungsspannung	24 V DC
Betriebssystem	Windows CE 6.0 / Windows Xpe

Tab. 2-15: Technische Daten VEP 30.4 / 40.4 / 50.4

2.5.4 Verschleißteile

In den Standard-Industrie-Bedienterminals VEP sind Teile eingebaut, die eine begrenzte Lebensdauer besitzen und nicht der Gewährleistung unterliegen.

Es sind folgende Komponenten betroffen:

- CF Karte
- CMOS-Batterie
- Hintergrundbeleuchtung

Die Lebensdauer der einzelnen Komponenten sind in den Dokumentationen der Geräte beschrieben (siehe [Kapitel 2.5.7 Dokumentation, Seite 34](#)).

2.5.5 Ausführungen

Die Standard-Industrie-Bedienterminals VEP xx.5 sind in folgenden Ausführungen erhältlich:

Type	Bemerkungen
VEP40.5DBN-2G02E-A3D-NNN-NN-FW	Touchfront
VEP40.5EIN-2G02E-A3D-NNN-NN-FW	Kombifront
VEP50.5DFN-2G02E-A3D-NNN-NN-FW	Kombifront

Tab. 2-16: Ausführungen VEP xx.5

Die Standard-Industrie-Bedienterminals VEPxx.4 sind in folgenden Ausführungen erhältlich:

Type	Bemerkungen
VEP30.4EFN-512NN-A2D-NNN-NN-FW	Touchfront
VEP40.4DBN-512NN-A2D-NNN-NN-FW	Touchfront
VEP40.4BKN-512NN-A2D-NNN-NN-FW	Kombifront
VEP50.4DEN-512NN-A2D-NNN-NN-FW	Touchfront
VEP50.4BIN-512NN-A2D-NNN-NN-FW	Kombifront

Tab. 2-17: Ausführungen VEPxx.4

Visualisierungsgeräte

2.5.6 Zubehör

Für die VEP Geräte ist folgendes Zubehör erhältlich:

Type	Bemerkungen
Compact-Flash-Modul 1 GByte	CFM01.1-01G0-N-LBA-NN-NW
Compact-Flash-Modul 4 GByte	CFM01.1-04G0-N-LBA-NN-NW
Compact-Flash-Modul 8 GByte	CFM01.1-08G0-N-LBA-NN-NW
VAK40.1E-DE-U-MPNN	IndraControl VAK 40 (Einschubtastatur USB, DE)
VAK40.1E-EN-U-MPNN	IndraControl VAK 40 (Einschubtastatur USB, EN)

Tab. 2-18: Zubehör für VEP

2.5.7 Dokumentation

Für die ausführliche Beschreibung der Kompakt Industrie-PCs VEPxx.5 steht die folgende Dokumentation zur Verfügung:

- DOK-SUPPL*-VEP**.5****-PRxx-DE-P
xx - Ausgabe

Für die ausführliche Beschreibung der Kompakt Industrie-PCs VEPxx.4 steht die folgende Dokumentation zur Verfügung:

- DOK-SUPPL*-VEP**.4****-PRxx-DE-P
xx - Ausgabe

2.6 IndraControl VPB und VDP

2.6.1 Kurzbeschreibung

Der VPB 40.3 ist ein High End Industrie-PC, der in Verbindung mit den passiven Bediengeräten VDP 16.3, VDP 40.3 oder VDP 60.3 ein PC-basierendes Bedien- und Visualisierungsterminal mit hoher Industrietauglichkeit darstellt. Der VPB 40.3 ist für den Einbau in einen Schaltschrank vorgesehen. Die Bediengeräte sind für den Einbau in ein Bedienpult oder in eine Schaltschrankwand konzipiert. Die Verbindung zwischen VDPxx.3 und VPB 40.3 erfolgt über eine CDI-Schnittstelle.

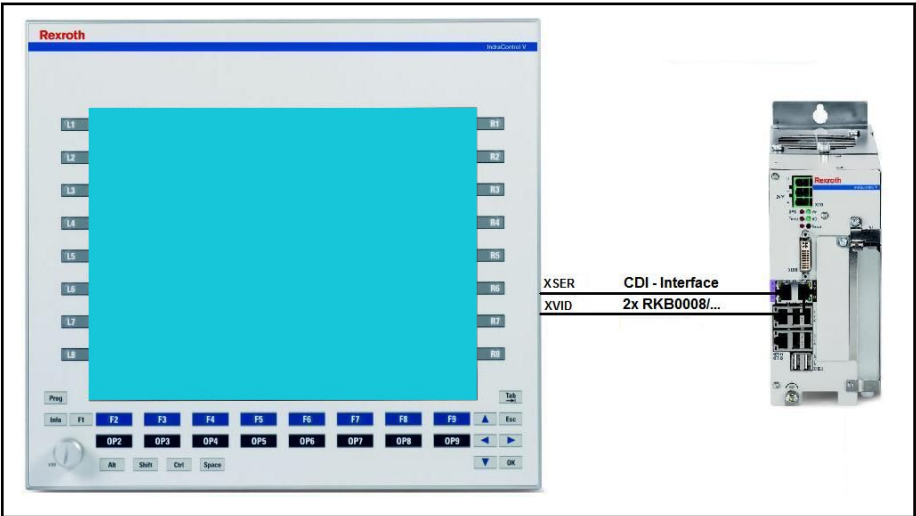


Abb. 2-17: High End Industrie-PC VPB 40.3 mit Bediengerät VDP 40.3

2.6.2 Einsatzbereich

Der Kompakt Industrie-PC VPB 40.3 wird in Industrieumgebungen mit erhöhten Anforderungen für Vibration und Schock eingesetzt.

2.6.3 Technische Daten

VPB 40.3

Bezeichnung	VPB40.3
PC-Box	1, 2 oder 4 Steckplätze
Prozessor	Intel Celeron™ P4500 1,86 GHz Intel Core i5™ 520M 2,4 GHz oder Intel Core i7™ 620M 2,66 GHz
Hauptspeicher	2 GB oder 4 GB RAM
Versorgungsspannung	24 V DC

Tab. 2-19: Technische Daten VPB 40.3

VDP 16.3/40.3/60.3

	VDP 16.3	VDP 40.3	VDP 60.3
Display	12" Farbdisplay	15" Farbdisplay	19" Farbdisplay
Ausführung Frontplatte	16 Maschinenfunktionstasten, Touchscreen bzw. kombiniert		
Versorgungsspannung	24 V DC		

Tab. 2-20: Varianten VDPxx.3

2.6.4 Verschleißteile

Im Kompakt Industrie-PC VPB 40.3 und im Bediengerät VDPxx.3 sind Teile eingebaut, die eine begrenzte Lebensdauer besitzen und nicht der Gewährleistung unterliegen.

Es sind folgende Komponenten betroffen:

- Festplatte
- CMOS-Batterie
- Lüfter
- Hintergrundbeleuchtung

Die Lebensdauer der einzelnen Komponenten sind in den Dokumentationen der Geräte beschrieben (siehe [Kapitel 2.6.7 Dokumentation, Seite 37](#)).

2.6.5 Ausführungen

Die High End Industrie-PCs VPB 40.3 sind in folgenden Ausführungen erhältlich:

Type	Slots	Bemerkungen
VPB40.3D1N-2G0NN-C3D-DN-NN-FW	1	Celeron P4500, 2GB RAM, 160GB HDD
VPB40.3D1N-2G0NN-D4D-DN-NN-FW	1	Core i5-520M, 2GB RAM, 160GB HDD
VPB40.3D1L-2G0NN-C3D-DN-NN-FW	2	Celeron P4500, 2GB RAM, 160GB HDD

Visualisierungsgeräte

Type	Slots	Bemerkungen
VPB40.3D1L-2G0NN-D4D-DN-NN-FW	2	Core i5-520M, 2GB RAM 160GB HDD
VPB40.3D1L-2G0NN-D4D-HN-NN-FW	2	Core i5-520M, 2GB RAM 100GB SSD
VPB40.3D1L-2G0NN-D4D-EE-NN-FW	2	Core i5-520M, 2GB RAM 2x160GB HDD (RAID1), DVD-Brenner
VPB40.3D1L-2G0NN-D4D-DE-NN-FW	2	Core i5-520M, 2GB RAM 160GB HDD, DVD-Brenner
VPB40.3D1L-2G0NN-D4D-EN-NN-FW	2	Core i5-520M, 2GB RAM, 2x160GB HDD (RAID1)
VPB40.3D1M-2G0NN-D4D-DE-NN-FW	2	Core i7-620M, 4GB RAM, 160GB HDD
VPB40.3D1L-4G0NN-D5D-DN-NN-FW	2	Core i7-620M, 4GB RAM, 2x160GB HDD (RAID1), DVD-Brenner
VPB40.3D1L-4G0NN-D5D-EE-NN-FW	2	Core i7-620M, 4GB RAM, 2x160GB HDD (RAID1)
VPB40.3D1L-4G0NN-D5D-HN-NN-FW	2	Core i7-620M, 4GB RAM, 100GB SSD
VPB40.3D1Q-4G0NN-D5D-DN-NN-FW	2	Core i7-620M, 4GB RAM, 160GB HDD
VPB40.3D1Q-4G0NN-D5D-EE-NN-FW	2	Core i7-620M, 4GB RAM, 2x160GB HDD (RAID1), DVD-Brenner
VPB40.3D1M-2G0NN-C3D-DE-NN-FW	4	Celeron P4500, 2GB RAM, 160GB HDD, DVD-Brenner
VPB40.3D1M-2G0NN-C3D-HN-NN-FW	4	Celeron P4500, 2GB RAM, 100GB SSD
VPB40.3D1M-2G0NN-D4D-DE-NN-FW	4	Core i5-520M, 2GB RAM, 160GB HDD, DVD-Brenner
VPB40.3D1M-2G0NN-D4D-EE-NN-FW	4	Core i5-520M, 2GB RAM, 2x160GB HDD (RAID1), DVD-Brenner
VPB40.3D1M-4G0NN-D5D-DE-NN-FW	4	Core i7-620M, 4GB RAM, 160GB HDD, DVD-Brenner
VPB40.3D1M-4G0NN-D5D-EE-NN-FW	4	CPU Core i7-620M, 4GB RAM, 2x160GB HDD (RAID1), DVD-Brenner
VPB40.3D1M-4G0NN-D5D-HN-NN-FW	4	CPU Core i7-620M, 4GB RAM, 100GB SSD

Tab. 2-21: Auswahl VPB 40.3

Die Bediengeräte VDPxx.3 sind in folgenden Ausführungen erhältlich:

Type	Bemerkungen
VDP16.3BKN-D1-NN-NN	30,5 cm (12") Display, Folientasten
VDP16.3DBN-D1-NN-NN	30,5 cm (12") Display, Touchscreen
VDP40.3DEN-D1-NN-NN	38,1 cm (15") Display, Touchscreen
VDP40.3BIN-D1-NN-NN	38,1 cm (15") Display, Folientasten
VDP40.3DFN-D1-NN-NN	38,1 cm (15") Display, Touchscreen, Folientasten

Type	Bemerkungen
VDP60.3FEN-D1-NN-NN	48,3 cm (19") Display, Touchscreen

Tab. 2-22: Auswahl VDPxx.3

2.6.6 Zubehör

Verbindungskabel (CDI-Interface)

Die Verbindung des Kompakt-PCs VPB 40.3 und eines Bediengerätes VDPxx.3 erfolgt über das CDI-Interface. Es sind folgende konfektionierte Kabel erhältlich:

Type	Bemerkung
RKB0008/002,5	Bus - Kabel, konfektioniert Ethernet-Kabel, 100-Base-T, CAT 6+ konfektioniert, UL, schleppkettenfähig Kabellänge 2,5 m
RKB0008/005,0	wie oben, Kabellänge 5 m
RKB0008/010,0	wie oben, Kabellänge 10 m
RKB0008/015,0	wie oben, Kabellänge 15 m
RKB0008/020,0	wie oben, Kabellänge 20 m
RKB0008/025,0	wie oben, Kabellänge 25 m
RKB0008/030,0	wie oben, Kabellänge 30 m
RKB0008/035,0	wie oben, Kabellänge 35 m
RKB0008/040,0	wie oben, Kabellänge 40 m
RKB0008/050,0	wie oben, Kabellänge 50 m

Tab. 2-23: Verbindungskabel VPB 40.3 - VDPxx.3



Für die Verbindung eines VPB 40.3 und eines VDPxx.3 sind 2 Verbindungskabel vom Typ RKB0008/... erforderlich.

Zubehör für VPB 40.3

Type	Bemerkung
VAP01.1H-W23-024-010-NN	Netzteil 24V DC; 10A
VAU01.1U-024-024-240-NN	Unterbrechungsfreie Stromversorgung mit 24V Eingang / 24V Ausgang / 240W
RKB0019/...	USB Schnittstellenkabel zum Anschluss der unterbrechungsfreien Stromversorgung Längen: 0,5m; 1,0m; 3,0m; 5,0m

Tab. 2-24: Zubehör für VPB 40.3

2.6.7 Dokumentation

Für die ausführliche Beschreibung des High End Industrie-PCs VPB 40.3 stehen folgende Dokumentationen zur Verfügung:

- DOK-SUPPL*-VPB*40.3***-PRxx-DE-P
- DOK-SUPPL*-VAU*01.1U**-PRxx-DE-P

Visualisierungsgeräte

Für die ausführliche Beschreibung der Bediengeräte VDPxx.3 steht die folgende Dokumentation zur Verfügung:

- DOK-SUPPL*-VDP*XX.3***-PRxx-DE-P
- xx - Ausgabe

2.7 IndraControl VPP

2.7.1 Kurzbeschreibung

Die Bediengeräte VPP 16.3/40.3/60.3 sind aktive PC-basierende Bedien- und Visualisierungsterminals mit hoher Industrietauglichkeit. Sie werden vorwiegend in ein Bedienpult oder in eine Schaltschrankwand eingebaut.

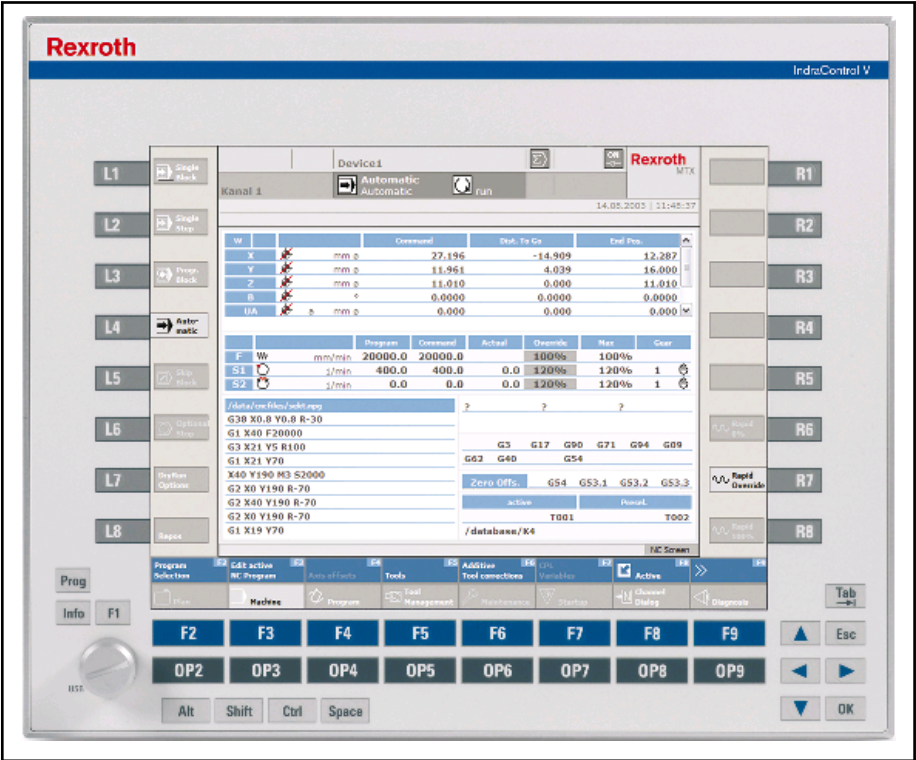


Abb. 2-18: Kompakt Industrie-PC VPP 40.3 mit 15" Farbdisplay

2.7.2 Einsatzbereich

Die Kompakt Industrie-PCs VPP 16.3/40.3/60.3 werden bei Standardanwendungen bis hin zu High-End-Anforderungen eingesetzt.

2.7.3 Technische Daten

	VPP 16.3	VPP 40.3	VPP 60.3
Display	12" Farbdisplay	15" Farbdisplay	19" Farbdisplay
Ausführung Frontplatte	16 Maschinenfunktionstasten		
PC-Box	1, 2 oder 4 Steckplätze		
Prozessor	Intel Celeron™ P4500 1,86 GHz Intel Core i5™ 520M 2,4 GHz Intel Core i7™620M 2,66 GHz		

Hauptspeicher	2 GB oder 4 GB RAM
Versorgungsspannung	24 V DC

Tab. 2-25: Technische Daten VPP 16.3/40.3

2.7.4 Verschleißteile

Im Kompakt Industrie-PC VPP 16.3/40.3 sind Teile eingebaut die eine begrenzte Lebensdauer besitzen und nicht der Gewährleistung unterliegen.

Es sind folgende Komponenten betroffen:

- Festplatte
- CMOS-Batterie
- Lüfter
- Hintergrundbeleuchtung

Die Lebensdauer der einzelnen Komponenten sind in den Dokumentationen der Geräte beschrieben (siehe [Kapitel 2.7.7 Dokumentation, Seite 40](#)).

2.7.5 Ausführungen

Die Kompakt Industrie-PCs VPP 16.3/40.3 sind in folgenden Ausführungen erhältlich:

Type	Slots	Bemerkungen
VPP16.3DBM-4G0NN-D5D-DN-NN-FW	4	30,5 cm (12")-Display, 160 GB HDD, Core i7-620M, 4 GB RAM
VPP40.3DEK-2G0NN-C3D-DN-NN-FW	1	38,1 cm (15")-Display, Touchscreen, Celeron P4500, 2GB RAM, 160GB HDD
VPP40.3BIK-2G0NN-C3D-DN-NN-FW	1	38,1 cm (15")-Display, Folientasten, Celeron P4500, 2GB RAM, 160GB HDD
VPP40.3BIM-2G0NN-D4D-DN-NN-FW	4	38,1 cm (15")-Display, Folientasten, Core i5-520M, 2GB RAM, 160GB HDD
VPP40.3DEM-2G0NN-D4D-HN-NN-FW	4	38,1 cm (15")-Display, Folientasten, Core i5-520M, 2GB RAM, 100GB SSD
VPP40.3DFL-2G0NN-D4D-HN-NN-FW	2	38,1 cm (15")-Display, Folientasten und Touchscreen, Core i5-520M, 2GB RAM, 100GB SSD
VPP40.3BIM-4G0NN-D5D-DN-NN-FW	1	38,1 cm (15")-Display, Folientasten, Core i7-620M, 4GB RAM, 100GB SSD
VPP40.3DEM-4G0NN-D5D-DN-NN-FW	4	38,1 cm (15")-Display, Touchscreen, Core i7-620M, 4GB RAM, 160GB HDD
VPP40.3BIK-4G0NN-D5D-HN-NN-FW	1	38,1 cm (15")-Display, Folientasten, Core i7-620M, 4GB RAM, 100GB SSD
VPP40.3DEN-4G0NN-D5D-DN-NN-FW	1	38,1 cm (15")-Display, Touchscreen, 160 GB HDD, Core i7-620M, 4 GB RAM

Visualisierungsgeräte

Type	Slots	Bemerkungen
VPP60.3FEL-2G0NN-D4D-DN-NN-FW	2	48,3 cm (19")-Display, Touchscreen, 160 GB HDD, Core i5-520M, 2 GB RAM
VPP60.3FEL-4G0NN-D5D-HN-NN-FW	2	48,3 cm (19")-Display, Touchscreen, 100 GB SSD, Core i7-620M, 4 GB RAM

Tab. 2-26: Ausführungen VPP 16.3/40.3

2.7.6 Zubehör

Für die VPP-Geräte ist folgendes Zubehör erhältlich:

Type	Bemerkungen
VAP01.1H-W23-024-010-NN	Netzteil 24 V DC; 10 A
VAU01.1U-024-024-240-NN	Unterbrechungsfreie Stromversorgung mit 24 V Eingang / 24 V Ausgang / 240 W
RKB0019/...	USB-Schnittstellenkabel zum Anschluss der unterbrechungsfreien Stromversorgung; Längen: 0,5 m; 1,0 m; 3,0 m; 5,0 m

Tab. 2-27: Zubehör für VPP 16.3/40.3

2.7.7 Dokumentation

Für die ausführliche Beschreibung der Kompakt Industrie-PCs VPP 16.3/40.3/60.3 stehen die folgenden Dokumentationen zur Verfügung:

- DOK-SUPPL*-VPP*XX.3***-PRxx-DE-P
- DOK-SUPPL*-VAU*01.1U**-PRxx-DE-P

xx - Ausgabe

3 Visualisierungen

3.1 Übersicht

Die Systeme IndraLogic XLC und IndraMotion MLC stellen mehrere Visualisierungen zur Verfügung. Jede Visualisierung hat ein spezielles Anwendungsgebiet. Die folgenden Visualisierungen werden unterstützt:

- **VI-Logic:** Diese Visualisierung ist vorrangig für die Entwicklungs- und Inbetriebnahmephase gedacht. Sie läuft direkt in der IndraWorks-Entwicklungsumgebung. Es werden sowohl Text, als auch graphische Elemente unterstützt.

Siehe [VI-Logic, Seite 41](#)

- **VI-Composer:** Diese Projektiersoftware dient der Projektierung aller Kleinbedienterminals der Bauart VCP und VCH. Der Funktionsumfang beschränkt sich auf Textdarstellung und einfache graphische Elemente.

Siehe [VI-Composer, Seite 44](#)

- **WinStudio:** Diese Visualisierungssoftware dient der Projektierung aller PC-basierten und Embedded Systeme. Es werden sowohl Text als auch graphische Elemente unterstützt.

Siehe [WinStudio, Seite 46](#)

3.2 VI-Logic

3.2.1 Einsatzgebiet

Integrierte Visualisierung für Test / Inbetriebnahme / Diagnose.



Die IndraLogic Inbetriebnahme-Visualisierung "VI-Logic Visualisierung" sollte nur zur Unterstützung bei der Maschineninbetriebnahme verwendet werden.



Die IndraLogic Inbetriebnahme-Visualisierung "VI-Logic Visualisierung" ist sowohl auf dem PC, als auch auf der Steuerung lauffähig. Damit sind auch unterschiedliche Features verbunden.

3.2.2 Eigenschaften der integrierten Visualisierung

- Innerhalb einer Visualisierung sind beliebige Ausdrücke, sogar Funktionsaufrufe, zulässig.
- Eine Visualisierung kann als Objekt unterhalb einer Applikation (oder im Ordner "Allgemeiner Baustein") angelegt werden.
- Im Gegensatz zu IndraLogic 1.x werden alle Visualisierungselemente, wie auch die Visualisierungsobjekte selbst als IEC61131-3 Funktionsbausteine implementiert. Somit können Visualisierungen in anderen Visualisierungen instanziiert, d. h. referenziert werden, und sie können verwendet werden, um andere Funktionsbausteine zu erweitern.
- Ein Platzhalter-Konzept ist vorhanden, das Platzhalter als Eingabe-Variablen einer Visualisierung behandelt.
- Schnittstellen können in einem Schnittstellen-Editor bearbeitet werden. Die Verwendung von Frame-Elementen erlaubt ein einfaches Wechseln der Anzeige von verschiedenen anderen Visualisierungen innerhalb eines Visualisierungsobjekts.

Visualisierungen

- Visualisierungen im Projektexplorer werden pro Applikation durch einen Visualisierungsmanager verwaltet, der gemeinsame Einstellungen für alle applikationsspezifischen Visualisierungen definiert.
- Einige allgemeine Einstellungen für alle Visualisierungen in einem Projekt (Dateipfade, Größenanpassung etc.) können in den Visualisierungsoptionen vorgenommen werden.
- Jede einzelne Visualisierung hat Eigenschaften wie beispielsweise ihre geplante Verwendung (als "Visualisierung", "Numpad/Keypad" oder "Dialog") oder die Anzeigegröße.
- Der Visualisierungs-Editor wird durch eine Werkzeuge-Box (Toolbox) ergänzt, die die verfügbaren Visualisierungselemente bereitstellt und durch einen Eigenschaften-Editor für die Konfiguration der eingefügten Elemente.
- Die Visualisierungselemente können auf einfache Weise angeordnet und gruppiert werden. Die Standardelemente werden über entsprechende Visualisierungsbibliotheken im Projekt bereitgestellt. IndraLogic 1.x Visualisierungen können importiert werden.
- Die Visualisierungselemente können über IndraLogic-Projektvariablen animiert werden, die direkt eingetragen werden oder über Ausdrücke, d. h. Kombinationen der Variablen mit Operatoren und Konstanten. Beispielsweise erlaubt dies ein Skalieren von Variablenwerten, so dass die Variablenwerte für die Verwendung in der Visualisierung geeignet sind.
- Sprachumschaltung (ANSI oder UNICODE) innerhalb einer Visualisierung ist durch die Verwendung von Textlisten möglich.
- Die Variablenwerte einer Applikation können über ein Textfeld-Element in der Visualisierung verändert und angezeigt werden. Die Formatierung dieser Ein- und Ausgaben basieren auf der Standardfunktion sprintf (C-Bibliothek).
- Externe Datenquellen können verwendet werden.
- Jedem Visualisierungselement kann ein Tooltip-Text zugeordnet werden.
- Neben der Zoom-Funktion ist auch automatisches Skalieren (Visualisierung passt sich der Bildschirmgröße an) möglich.
- Visualisierungen können in Bibliotheken gespeichert und somit in anderen Projekten verfügbar gemacht werden.
- Mehrsprachigkeit

Die VI-Logic Visualisierung unterstützt die Mehrsprachigkeit von statischen Texten über die "GlobaleTextListen" und die dynamischen Texte über die "TextListen". Bei dynamischen Texten wird in Abhängigkeit eines Variablenwerts ein anderer Text angezeigt.

3.2.3 Visualisierungselemente

Zur Erstellung von Inbetriebnahme-Visualisierungen stehen verschiedene Visualisierungselemente zur Auswahl, die in die Bereiche "Basis", "Allgemeine Steuerelemente", "Messgeräte", "Lampen/ Schalter/ Bilder" und "Spezielle Steuerelemente" unterteilt sind. Die folgenden Auflistungen enthalten die zur Verfügung stehenden Visualisierungsobjekte:

Visualisierungselemente

Basis	Allgemeine Steuerelemente	Messgeräte	Lampen/ Bilder	Schalter/	Spezielle Steuerelemente
Zeiger	Schaltfläche	Zeigerinstrument	Bildwechsler		Trace
Rechteck	Scrollbar	Histogramm	Lampe		ActiveX Element
Abgerundetes Rechteck	Tabelle	Balkenanzeige			
Ellipse	Textfeld	Hebelschalter			
Linie		Energieschalter			
Polygon		Drückschalter			
Linienzug		Drückschalter LED			
Bezierskurve		Kippschalter			
Bild		Drehschalter			
Frame					

Tab. 3-1: Visualisierungselemente nach Bereichen

Die Konfiguration der Visualisierungsobjekte erfolgt über statische und dynamische Eigenschaften sowie Ereignisbehandlungen für Benutzereingaben im Fenster "Eigenschaften".

In den Aktionen für die Benutzereingaben stehen vordefinierte Aktionen zur Auswahl, wie z.B. die Ausführung von eingeschränktem ST-Code (es sind nur Zuweisungen, jedoch keine Schleifen oder Verzweigungen erlaubt), Beeinflussung von Variablen, Bild- und Sprachumschaltungen, interne Kommandos sowie Aktionen zum Öffnen und Schließen von Dialogen.

3.2.4 Unterstützte Visualisierungsgeräte

Für die Visualisierung ist keine zusätzliche Hardware nötig, da Visualisierung und Engineering auf einem Gerät laufen.

Das Engineering wird vereinfacht, weil die Projektierung der Visualisierung und der SPS-Applikation in einer Oberfläche erfolgen.

3.2.5 Protokoll

Die integrierte Visualisierung kann direkt auf die Variablen aus der Steuerung zugreifen – extern erzeugte Listen von Variablen oder Symbolen sind nicht erforderlich.

Automatische Projektierung der Kommunikation – eine fehleranfällige Kommunikationskonfiguration ist überflüssig.



In der Visualisierung "VI-Logic" bzw. in den Visualisierungsobjekten stehen alle Variablen, die im SPS-Programm (Implementation) verwendet werden oder Variablen die mit dem Attribut "{attribute 'linkalways'}" deklariert wurden, für die Projektierung zur Verfügung.

D.h. Variablen, die nur in der Deklaration enthalten sind, werden vom Compiler aus dem Laufzeitcode heraus optimiert und können daher nicht visualisiert bzw. beeinflusst werden.

Visualisierungen

3.2.6 Dokumentation

Eine ausführliche Beschreibung der integrierten Visualisierungssoftware steht in der Online-Hilfe von IndraWorks zu Verfügung.

3.3 VI-Composer

3.3.1 Einsatzgebiet

Die Projektiersoftware VI Composer dient der einheitlichen Projektierung aller Kleinbedienterminals der Bauart VCP und VCH.



Die Projektiersoftware **VI Composer** ist nicht Bestandteil von IndraWorks und muss separat bestellt werden.

3.3.2 Projektierbare Kleinbedienterminals

Folgende Kleinbedienterminals können mit der Projektiersoftware VI-Composer projiziert werden:

- VCP:
 - IndraControl VCP 02
 - IndraControl VCP 05
 - IndraControl VCP 08
 - IndraControl VCP 11
 - IndraControl VCP 20
 - IndraControl VCP 25
 - IndraControl VCP 35
- VCH
 - IndraControl VCH 08

Siehe auch [IndraControl VCP](#), Seite 17.

3.3.3 Kommunikationsprotokoll BRC-Symbolic

Kurzbeschreibung

Das Protokoll bietet Ihnen wahlfreien Zugriff auf alle globalen Datenobjekte der Steuerung lesend und schreibend.

Die Projektiersoftware übernimmt die Datenobjekte der Symboldatei (Dateiname.SYM oder Dateiname.xml), die bei der Kompilierung des Steuerungsprojekts erzeugt werden.

Das angeschlossene Bediengerät greift über den symbolischen Namen auf ein Datenobjekt zu.

Datenobjekte

Grundsätzlich ist die Länge einer Variablen durch die Definition in der Projektiersoftware "IndraLogic" festgelegt.

Einzelvariablen

Sie können auf Variablen vom Typ BOOL, SINT, INT, DINT, BYTE, USINT, WORD, UINT, LWORD, DWORD, UDINT, REAL, STRING, WSTRING, LINT, ULINT, LREAL, BITORBYTE, ENUM, TIME, TOD, DATE und DT zugreifen.

Stringvariablen

Für Stringvariablen verwenden Sie den Variablentyp STRING(N) oder WSTRING(N), wobei N die Länge des Strings angibt.



Stringvariablen dürfen eine Länge von 64 Zeichen nicht überschreiten.

Tabellen

Für die Darstellung von ARRAY-Variablen in Tabellenfeldern müssen Sie den Datentyp ARRAY [1..N] verwenden. Das ARRAY [1..N] muss aus einem der folgenden Basistypen bestehen:

BOOL, BYTE, WORD, LWORD, DWORD, SINT, INT, DINT, USINT, UINT, UDINT, LINT, ULINT, LREAL, REAL, STRING, WSTRING, ENUM

3.3.4 Verbindungseinstellungen

Protokollparameter für PROFIBUS-DP

- Geräteadresse

Dieser Parameter gibt die Adresse des Bediengerätes an.

Einstellbare Werte	Standardwert
1 bis 126	2

Tab. 3-2: Geräteadresse

- Verzögerung bis Verbindungsaufbau.

Dieser Parameter gibt eine Wartezeit an, bis vom Bediengerät die Kommunikation gestartet wird.

Einstellbare Werte	Standardwert
5s bis 255s	5s

Tab. 3-3: Verzögerungszeit bei Verbindungsaufbau

- Reaktionszeit Slave

Dieser Parameter gibt an, wie lange das Bediengerät auf die Antwort der Steuerung wartet

Einstellbare Werte	Standardwert
10ms bis 65535ms	1000ms

Tab. 3-4: Reaktionszeit Slave

Protokollparameter für Ethernet

Verbindungstabelle:

In der Verbindungstabelle definieren Sie Verbindungen zu maximal 16 Steuerungen.

In die Spalte Verbindungsname werden die Verbindungsnamen des Hauptdialogs übernommen.

In der Spalte IP-Adresse der Steuerung zu jedem Verbindungsnamen eine gültige IP-Adresse eingeben.

Visualisierungen

3.3.5 Dokumentation

Für die ausführliche Beschreibung der Projektierungssoftware VI-Composer steht folgende Dokumentationen zur Verfügung:

DOK-SUPPL*-VIC*BED*0... de-DE

3.4 WinStudio

3.4.1 Übersicht

WinStudio ist eine durchgängige Visualisierungssoftware für alle PC-basierten und Embedded Systeme.

Das Produkt besteht aus zwei Teilen.

WinStudio Engineering

- Projektierungswerkzeug zur Erstellung einzelner HMI-Bilder bis hin zu kompletten Bedienoberflächen.
- Einmalige Investition (Lizenz) für beliebig viele WinStudio-Projekte verwendbar.
- Skalierbar
 - WinStudio Lite-Lizenz (funktional eingeschränkt; im Standardumfang der IW- Systeme enthalten).
 - Gestaffeltes Lizenzmodell nach Anzahl der verwendeten Tags (Prozessvariablen).
- Ausprägungen des WinStudio Engineering-Paketes.
 - Integraler Bestandteil in IndraWorks Engineering zur Erstellung von Visualisierungsanwendungen.
 - Ergänzung der IndraWorks HMI- Oberfläche (IndraWorks OPD) .
 - Erstellung von eigenständigen Visualisierungen (WinStudio Runtime).
 - Stand-alone Editor zur Erstellung von eigenständigen Visualisierungen (WinStudio Engineering stand-alone).

WinStudio Runtime

- Software auf den Visualisierungsgeräten
 - vorinstalliert bzw.
 - für die Installation vorbereitet (Software-Download).
- Skalierung
 - WinStudio Lite-Lizenz (funktional eingeschränkt; im Standardumfang der IW-Systeme enthalten).
 - Gestaffeltes Lizenzmodell nach Anzahl der verwendeten Tags (Prozessvariablen).
- Ausprägungen der WinStudio Runtime-Umgebung.
 - WinStudio in IndraWorks HMI-Oberfläche (IndraWorks OPD).
(Vorgefertigte Bedienoberfläche der Systeme MLC, MTX, IL unter Windows CE/Windows 7 embedded compact, Win XP/ Win XPe, Win 7/ Win 7e)
 - WinStudio Runtime (IndraWorks unabhängige Bedienoberfläche).

3.4.2 WinStudio-Elemente im Projektexplorer

Für die Erstellung von WinStudio-Bildern, sowie der Konfiguration von erweiterten Funktionen (z. B. Abläufe, Alarme) und der Kommunikationsanbindung stehen verschiedene Editoren und Dialoge zur Verfügung, die Sie über das Kontextmenü des Ordners "WinStudio" bzw. weiterer Unterordner öffnen können.

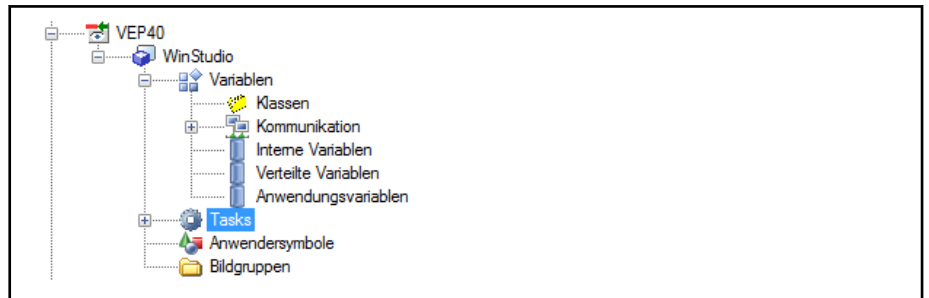


Abb. 3-1: WinStudio-Ordner

Unterhalb des WinStudio Knotens im Projektexplorer befinden sich folgende Unterordner:

- **Variablen**
In diesem Ordner finden Sie die Editoren für verschiedene Variablentypen (Anwendungsvariablen, Systemvariablen), Editoren zur Konfiguration der Prozesskommunikation (Treiberblätter) und zur Erstellung eigener Datentypen (Klassen).
- **Tasks**
In diesem Ordner können Sie erweiterte WinStudio-Objekte (z.B. Mathematikblätter, Alarmblätter) erstellen und editieren.
- **Anwendersymbole**
In diesem Ordner werden selbsterstellte WinStudio-Objekte angezeigt, die in WinStudio-Bildern verwendet werden können.
- **Bildgruppen**
In diesem Ordner können Sie Bildgruppen erstellen, die mehrere WinStudio-Bilder zu einem gemeinsamen Bild zusammenfassen.

Folgende WinStudio-Elemente stehen nur zur Verfügung, falls mindestens eine Lizenz "WinStudio-Engineering und Runtime 1.5k" freigeschaltet und in den Einstellungen des WinStudio-Projektes als Zielsystem "WinStudio 1.5k" oder höher ausgewählt ist.

- [Alarme, Seite 67](#)
- [Rezepte, Seite 67](#)
- [Reports, Seite 68](#)
- [Trends, Seite 69](#)
- TCP/IP-Treiberblätter
- DDE-Treiberblätter

Wenn Sie ein Projekt öffnen, in dem als Zielsystem "WinStudio 1.5k" oder höher ausgewählt ist, jedoch keine ausreichende WinStudio-Lizenz aktiv ist, werden die lizenzpflichtigen Elemente angezeigt, jedoch werden diese deaktiviert und können nicht bearbeitet werden.

3.4.3 Allgemeine WinStudio Einstellungen und Funktionen

Projekt-Eigenschaften

Visualisierungen

Wählen Sie "Projekt Einstellungen", um die gleichnamige Dialogbox zu öffnen, mit der Sie grundlegende Projekteinstellungen vornehmen können.

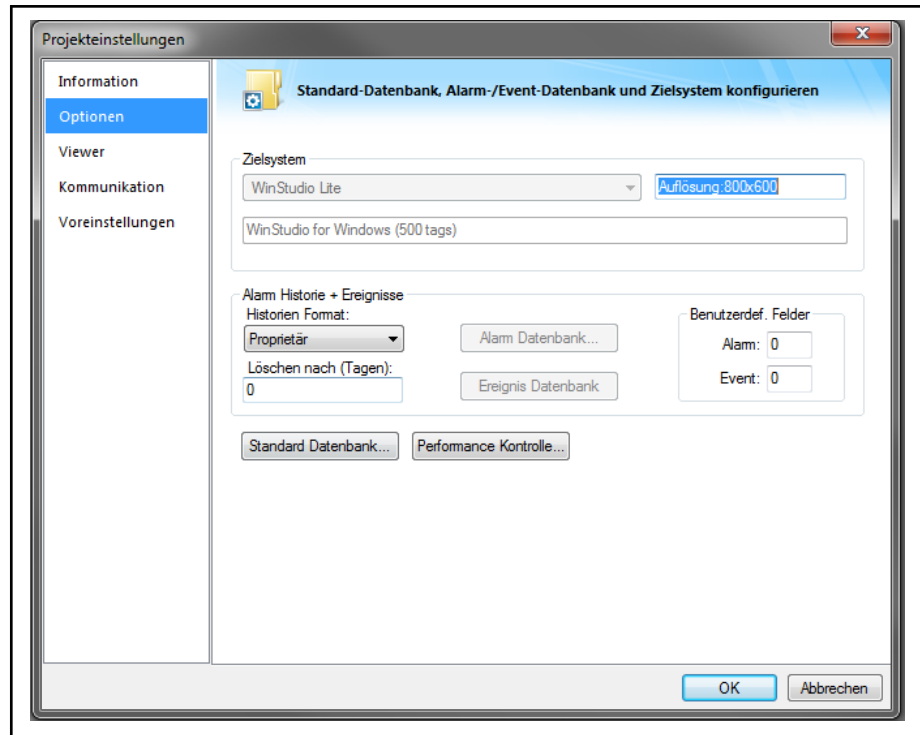


Abb. 3-2: Dialog: Projekteinstellungen

Nachfolgend werden die Registerkarten beschrieben:

- [Information, Seite 48](#)
- [Optionen, Seite 49](#)
- [Runtime Desktop, Seite 50](#)
- [Kommunikation, Seite 54](#)
- [Web, Seite 56](#)

Information

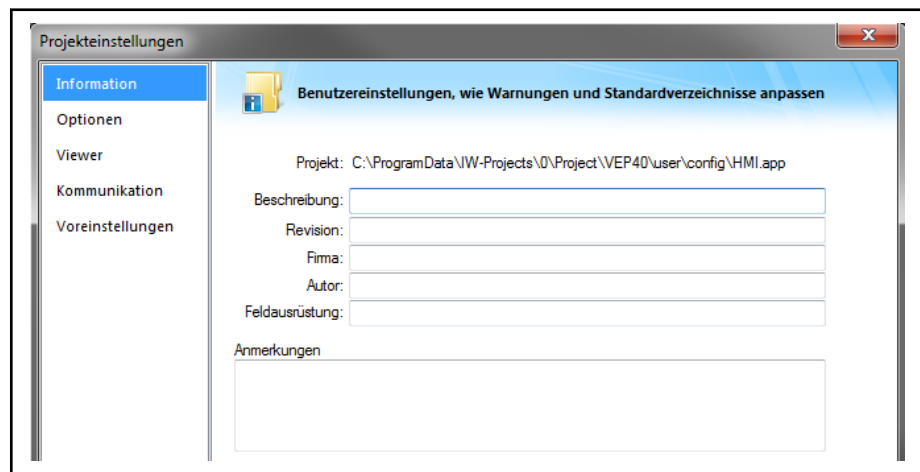


Abb. 3-3: Dialog: Projekteinstellungen (Registerkarte: Information)

Die Registerkarte im Dialog Projekteinstellungen dient der Beschreibung des Projektes zu Dokumentationszwecken.

Optionen

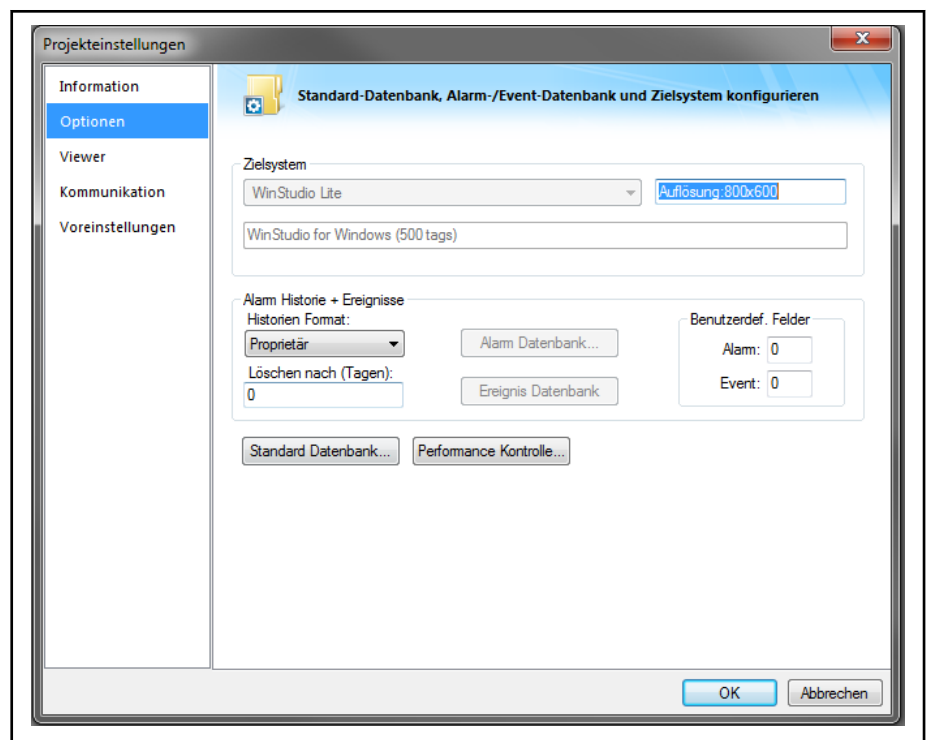


Abb. 3-4: Dialog: Projekteinstellungen (Registerkarte: Optionen)

Verwenden Sie diese Registerkarte, um allgemeine Projektparameter festzulegen:

- Zielsystem**

Über die Auswahl des Zielsystems legen Sie die Anzahl der verfügbaren WinStudio-Variablen, den verfügbaren Funktionsumfang, sowie den zum Zielsystem passenden Projekttyp (Win XP (e)/ Win 7(e) bzw. Windows CE/ Windows 7 embedded compact) fest. Als Voreinstellung ist das Zielsystem "WinStudio Lite" gewählt. Weitere Zielsysteme mit einer höheren Variablenanzahl und erweitertem Funktionsumfang sind nach Aktivierung der entsprechenden WinStudio-Engineering-Lizenz verfügbar.

Die Bildauflösung wird über das projektierte HMI Gerät festgelegt.



Wird eine Lizenz ausgewählt, die nicht mit der aktuellen Lizenz auf der Bedienstation übereinstimmt, wird die Anwendung u. U. nicht korrekt ausgeführt

- Alarm Historisch + Ereignisse**

Geben Sie einen Wert (in Tagen) in das Feld "Löschen nach (Tagen)" ein, um festzulegen, wie lange angelegte Alarm- und Ereignishistorie-Dateien zu Verfügung stehen sollen. Nach Ablauf dieser Zeit löscht WinStudio automatisch vorhandene Alarm- und Ereignishistorie-Dateien, die älter als der festgelegte Zeitraum sind.



Wird in dieses Feld der Wert 0 eingegeben, werden die Historie Dateien nicht automatisch gelöscht. In diesem Fall muss der Anwender sicher stellen, dass die Dateien gelöscht werden, da kontinuierlich Speicherplatz auf dem System belegt wird.

Visualisierungen

Mittels der Auswahl "Historisches Format" legen Sie fest wie und wo die Alarm Dateien gespeichert werden.

Eigentum	Speichert die Historie-Daten im Unterordner \Alarm der Anwendung (standardmäßig) in Textdateien unter Verwendung eines proprietären Formats von WinStudio.
Daten-bank	Speichert die Historie-Daten in der vom Benutzer festgelegten SQL Relationalen Datenbank unter Verwendung der integrierten ADO-Schnittstelle.
Binär	Speichert die Historie-Daten im Unterordner \Alarm der Anwendung (standardmäßig) in binär verschlüsselten Dateien unter Verwendung eines proprietären Formats von WinStudio.

- **Standard-Datenbank**

Ermöglicht es Ihnen, eine Standard-Datenbank zu konfigurieren, die von verschiedenen Tasks und Objekten genutzt werden kann.

Runtime Desktop

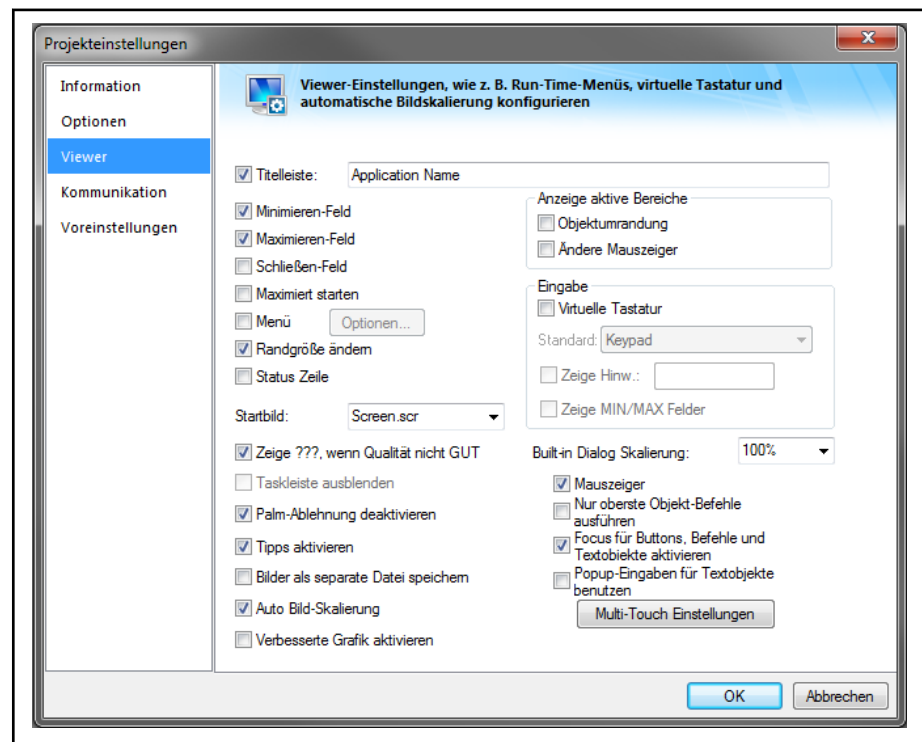


Abb. 3-5: Dialog: Projekteinstellungen (Registerkarte: Viewer)

Diese Registerkarte ermöglicht es, Runtime Desktop-Parameter zu aktivieren / deaktivieren:

- **Auswahlfeld und Textfeld Titel**
Legen sie den Standardtext für die Titelleiste des Viewer-Fensters fest.
- **Auswahlfelder Minimieren Feld, Maximieren Feld und Schließen**
Aktivieren Sie diese Optionen, um die entsprechenden Schaltflächen im Viewert Fenster einzublenden (aktivieren) oder auszublenden (deaktivieren).
- **Auswahlfeld Maximiert starten**
Die Anwendung wird maximiert ausgeführt.
- **Auswahlfeld Menü und Schaltfläche Optionen**

Aktivieren Sie das Auswahlfeld und die Schaltfläche Optionen, um festzulegen, welche Menüoptionen während der Laufzeit verfügbar sind. Im Dialog "Runtime Menü-Optionen" besteht die Möglichkeit, Menüoptionen einzublenden (aktivieren) oder auszublenden (deaktivieren).

- Auswahlfeld: **Randgröße**
Wird diese Option aktiviert, kann die Größe des Viewer-Fensters während der Laufzeit geändert werden.
- Auswahlfeld: **Statuszeile**
Um die Statuszeile in der Runtime-Anwendung anzuzeigen, kann diese Option aktiviert werden.
- Feld: **Startbild**
Geben Sie den Namen der WinStudio Bilddatei ein, die als Startbildschirm angezeigt werden soll, wenn Sie die Runtime-Anwendung öffnen.
- Auswahlfeld: **Zeige ???, wenn Qualität nicht GUT**
Wird diese Option aktiviert, so werden Fragezeichen (???) anstelle des Tag-Wertes angezeigt, falls zur Laufzeit Probleme bei der Verarbeitung der Date auftreten
- Auswahlfeld: **Taskleiste ausblenden**
Mittels dieser Option wird die Taskleiste ausgeblendet (nur Windows CE/ Windows 7 embedded compact)

Palm Ablehnung deaktivieren: Wählen Sie diese Option aus, um Palm Rejection während der Projektlaufzeit zu deaktivieren. Palm Rejection ist ein Merkmal von Windows-7-Touchscreen-Geräten; hierbei werden versehentliche Berührungen durch den Bediener erkannt und nicht als Signal angenommen. Dies kann jedoch das Ansprechverhalten des Touchscreens merklich verlangsamen.



Palm Rejection ist ausschließlich auf Windows-7-Touchscreen-Geräten verfügbar, daher hat die Auswahl von **Palm Rejection deaktivieren** keine Auswirkungen auf Projekte, die auf anderen Zielsystemen ausgeführt werden.

Visualisierungen



Windows-7-Touchscreen-Geräte verfügen auch über eine Option, mit der der Anwender durch längeres Berühren des Bildschirms einen Rechtsklick-Befehl ausführen kann („press and hold“). Diese Option ist Standardmässig aktiviert. So deaktivieren Sie die "press and hold" Funktion:

1. Klicken Sie auf die Schaltfläche Start und klicken Sie im Startmenü auf Systemsteuerung. Das Fenster Systemsteuerung wird angezeigt.
2. Doppelklicken Sie auf Eingabestift und Touch. Der Dialog Eingabestift und Touch wird angezeigt.
3. Klicken Sie auf die Registerkarte Touch.
4. Wählen Sie aus der Liste der Touch-Aktionen „Drücken und Halten“ und klicken Sie dann auf Einstellungen. Der Dialog Drücken und Halten – Einstellungen wird angezeigt.
5. Löschen Sie „Drücken und Halten für Rechtsklick aktivieren“.
6. Klicken Sie auf OK, um den Dialog „Drücken und Halten – Einstellungen“ zu schließen.
7. Klicken Sie auf OK, um den Dialog „Eingabestift und Touch“ zu schließen.

- **Auswahlfeld: Tipps aktivieren**

Mit dieser Option können Sie Windows Tooltips zur Laufzeit anzeigen. Die Tooltips können als Hinweis im Dialog Objekteigenschaften für jedes Objekt eingetragen werden.

- **Auswahlfeld: Auto Bild-Skalierung**

Mit Hilfe dieses Parameters können Sie die Bilder Ihrer Anwendung automatisch auf die Größe des Displays anpassen. Diese Funktion ist für lokale Anwendungen unter Windows XP/ Windows 7 (oder Web Thin Clients) verfügbar. Dieser Parameter existiert nicht für lokale Anwendungen unter Windows CE/Windows 7 embedded compact.



Es wird empfohlen die Auto Bild-Skalierung nur zu verwenden, wenn das Seitenverhältnis des Zieldisplays dem Seitenverhältnis der erstellten WinStudio-Bilder entspricht, da die Bildelemente ansonsten verzerrt dargestellt werden.

- **Abschnitt: Anzeige aktive Bereiche**

Aktivieren Sie die Optionen "Objektumrandung" und "Ändere Mauszeiger" in diesem Bereich, um die Objektkante und den Mauscursor zu ändern, wenn der Cursor über ein Objekt bewegt wird.

- **Abschnitt Eingabe**

Auswahlfeld: Virtuelle Tastatur

Aktivieren Sie diese Option, um den Mauscursor in der Runtime-Anwendung anzuzeigen.

- **Auswahlfeld Zeige Hinweise:** Ist diese Option aktiviert, wird ein Tipp in der Titelleiste der Tastatur angezeigt. Dieser Tipp wird für die einzelnen Objekte im Dialog Objekteigenschaften festgelegt. Für andere Fälle tragen Sie einen String-Wert in das Feld „Tipp anzeigen“ ein, der dann als Standardtipp dient.

- Auswahlfeld **Zeige Min/Max-Felder**: Wenn diese Option aktiviert ist, werden die erlaubten Mindest- und Höchstwerte am unteren Rand der Tastatur angezeigt. Diese Werte können im Dialog Objekteigenschaften für die einzelnen WinStudio Objekte festgelegt werden. Ansonsten werden die Eigenschaften Min und Max des zugehörigen Tags verwendet.



Die Felder Min und Max werden nur auf dem Tastaturtyp Tastenfeld angezeigt, und nur dann, wenn der zugeordnete Tag als Integer oder Real definiert ist. Wenn Min größer ist als Max, wird die Eingabe deaktiviert. Wenn das am Objekt eingestellte Paar Min/Max sich von den Eigenschaften Min/Max im Tag unterscheidet, dann wird versucht, die Größe der Eingabe entsprechend umzurechnen.

- Auswahlfeld **Mauszeiger**: Verwenden Sie diese Option, um zur Laufzeit den Mauszeiger anzuzeigen.



Die Option Mauszeiger wird unter Windows CE/ Windows 7 embedded compact, wenn es auf Armv4l-Prozessoren ausgeführt wird, nicht unterstützt.

- Auswahlfeld **Nur oberste Objekt-Befehle ausführen**: Diese Option legt fest, wie sich Ihr Projekt verhält, wenn der Benutzer in einen Bereich klickt, in dem sich zwei oder mehr Bildobjekte überschneiden. Wenn diese Option aktiviert wurde, werden nur die Befehle, welche dem Objekt auf oberster Bildebene zugeordnet sind, ausgeführt. Ist die Option inaktiv, dann werden die Befehle aller überlappenden Objekte ausgeführt.



Das Objekt mit der höchsten Objekt-ID gilt als das oberste. (Die ID eines Objekts wird in der Statuszeile am unteren Rand der Entwicklungsumgebung angezeigt.) Mithilfe der Werkzeuge Nach hinten/ Nach vorne verschieben können Sie die Reihenfolge verändern, in der die Objekte übereinander liegen.

- Auswahlfeld **Fokus für Buttons, Befehle und Textobjekte aktivieren**: Wird ein Klick auf eine Befehlsschaltfläche oder eine Texteingabe während der Laufzeit ausgeführt, so wird das entsprechende Objekt wie in der unten stehenden Illustration gezeigt fokussiert:

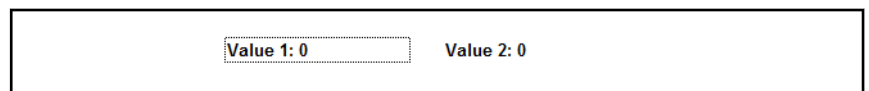


Abb. 3-6: Fokus auf Texteingabe (links), kein Fokus auf Texteingabe (rechts)

Danach können Sie mithilfe der **Tab**-Taste nacheinander zu allen fokussierbaren Objekten im Bild navigieren oder mit Hilfe der **Enter**-Taste das Objekt aktivieren, auf dem gerade der Fokus liegt. Diese Option ist standardmäßig aktiviert.

- Auswahlfeld **Popup-Eingabe für Textobjekte verwenden**: Wählen Sie diese Option, wenn für Texteingaben ein kleines Popup-Fenster angezeigt werden soll.

Visualisierungen

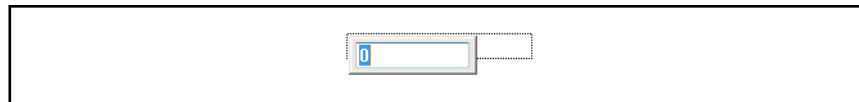


Abb. 3-7: Popup für Texteingaben



Wenn die (oben beschriebene) Option **Virtuelle Tastatur** ausgewählt ist, setzt sie diese Option außer Kraft.

Zudem gilt: Wenn die (oben beschriebene) Option **Fokus auf Schaltflächen, Befehle und Textobjekte aktivieren** "nicht" gesetzt ist, dann ist diese Option automatisch ausgewählt. Hiermit soll sichergestellt werden, dass es im Bild einen Hinweis gibt, welche Texteingabe jeweils aktuell aktiv ist.

- Dialogfeld **Multi-Touch Einstellungen** Falls Sie multitouchfähige Visualisierungshardware projektieren, können Sie mittels dieses Dialoges Multitouchfunktionen konfigurieren

Kommunikation

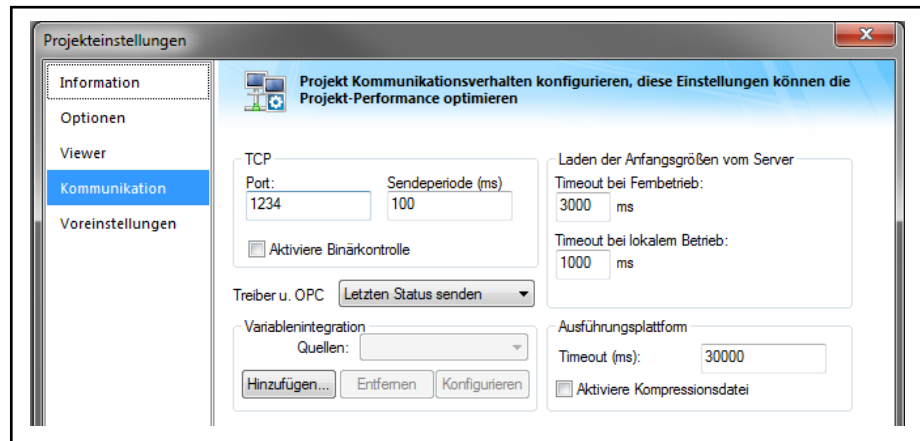


Abb. 3-8: Dialog: Projekteinstellungen (Registerkarte: Kommunikation)

Verwenden Sie diese Registerkarte, um allgemeine Kommunikationsparameter festzulegen:

- **Treiber und OPC**

Auswahl der Methode, die von allen Kommunikationstreibern und OPC-Client-Arbeitsblättern in der aktuellen Anwendung genutzt wird.

- **Jeden Status senden**

Ist diese Option aktiviert, wird jede Werteänderung eines Tags an die Steuerung übertragen.

- **Letzten Status senden**

Ist diese Option ausgewählt, dann wird nur der letzte Wert an die Steuerung übertragen.

- **TCP**

Konfigurationsdialog für die Einstellungen des TCP/IP Client und der TCP/IP Server Module

- **Port**

TCP Port, der vom TCP/IP Client und von den TCP/IP Server Modulen genutzt wird, die auf dem aktuellen Computer laufen. Wird dieser Wert in der lokalen Anwendung geändert, so muss sicher gestellt werden, dass dieser auch in der Remote-Anwendung, die mit der lokalen Anwendung kommuniziert, angepasst wird.

- **Sendeperiode (ms)**
Aktualisierungsintervall in Millisekunden, das zwischen zwei aufeinanderfolgenden Mitteilung liegt, die vom TCP/IP Client oder von den TCP/IP Server Modulen gesendet werden.
- **Aktiviere Binär Kontrolle**
Aktivieren Sie diese Option, um Informationen, die TCP/IP Client oder TCP/IP Server Modul an Remote-Station sendet, verschlüsselt zu übertragen. Eine Verschlüsselung erhöht die Systemsicherheit, senkt jedoch auch die Kommunikationsperformance. Wenn zwei Stationen über den TCP/IP Client und das TCP/IP Server Modul verbunden werden, muss sicher gestellt werden, dass die Einstellung bei beiden Anwendungen identisch ist.
- **Bereich Ausführungsplattform:** Konfigurieren Sie hier die Kommunikationseinstellungen für die Ausführungsplattform, das Werkzeug mittels dessen ihre Projektdateien auf das Zielsystem übertragen werden:
 - Auswahlfeld **Timeout (ms)**: Legt die Wartezeit (ms) fest, welche die Ausführungsplattform auf erfolgreichen Verbindungsaufbau mit dem Zielsystem wartet.
 - Auswahlfeld **Aktiviere Kompressionsdatei**: Wählen Sie diese Option aus, um die System- und Projektdateien zu komprimieren, bevor sie an das Zielsystem gesendet werden. Diese Option ist nur bei langsamen Verbindungen sinnvoll. Standardmäßig ist die Datenkomprimierung deaktiviert.
- **Bereich Laden der Ausgangsgrößen vom Server:** Zur Verbesserung der Leistung lädt der Viewer vor dem Anzeigen eines Bildes alle Server-Tags des anzuzeigenden Bildes in den Speicher. Konfigurieren Sie hier die Zeitüberschreitungseinstellungen für Remote- und lokale Viewer:
 - Feld **Timeout bei Fernbetrieb**: Legt die Zeit (ms) fest, die ein auf einer Remote-Station ausgeführter Secure Viewer oder Thin Client auf das Laden der Tags wartet.
 - Feld **Timeout bei lokalem Betrieb**: Legt die Zeit (ms) fest, die ein auf einer lokalen Station (d.h. auf dem Server) ausgeführter Viewer auf das Laden der Tags wartet.
- **Bereich Variablenintegration:** Hiermit können Sie automatisiert ein Treiberblatt generieren. Nähere Informationen hierzu finden Sie in der IndraWorks WinStudio Online-Hilfe "Rexroth WinStudio 7.3 Technische Referenz".

Visualisierungen

Web

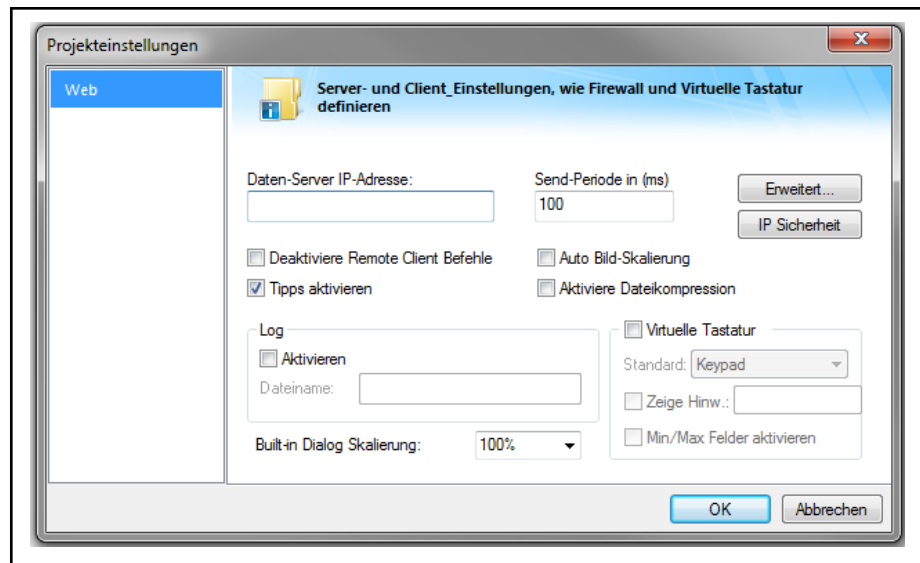


Abb. 3-9: Dialog: Projekteinstellungen (Registerkarte: Web)

Durch Aufruf der Registerkarte im Verzeichnis **WinStudio** ► **Werkzeuge** ► **Web-Einstellungen** werden die Parameter für einen Web Thin Client festgelegt:

- **Feld: Daten Server IP Adresse**
Eingabe der IP-Adresse (oder des Host-Namens) der Datenserverstation. Die Datenserverstation ist das Gerät, auf dem das WinStudio TCP/IP Server Modul läuft.
- **Feld: Send-Periode in ms**
Legen Sie hier die Aktualisierungsrate (ms) zwischen Server und Web Thin Client fest
Der Default-Wert ist 100 ms. Eine Verringerung des Wertes führt i. A. zu einer Erhöhung der Kommunikationslast im Netzwerk.
- **Auswahlfeld: Deaktiviere Remote Client Befehle**
Das Aktivieren dieser Option verhindert, dass ein Remote Client Kommandos vom Web Thin Client an den Server weitergibt. Wenn diese Option aktiviert ist, kann der Web Thin Client Daten vom Server lesen, aber keine Daten (Tag-Werte, Set-Points) an den Datenserver senden. Die Web Thin Client Station wird zu einer "Read Only Station".
- **Auswahlfeld: Tipps Aktivieren**
Auswahlfeld **Log**
Aktivieren und **DateiName**: Geben Sie einen Dateinamen im Textfeld ein, um eine Protokolldatei auf der Thin-Client-Station zu erzeugen. Diese Protokolldatei können Sie zu Fehlerbeseitigungszwecken einsetzen.
- **Bereich Virtuelle Tastatur**: Wird diese Option aktiviert, ist die virtuelle Tastatur für Thin Clients freigegeben. Mit Hilfe dieser Tastatur kann der Benutzer während der Laufzeit Daten auf Touchscreens eingeben – es wird keine Hardware-Tastatur benötigt.
Sie können für die virtuelle Tastatur folgende Einstellungen vornehmen:
 - **Standard**: Wählen Sie den Standard-Tastaturtyp, der in Ihrem Projekt verwendet werden soll, wenn durch das aufrufende Objekt bzw. die aufrufende Funktion kein Tastaturtyp festgelegt wird.

Visualisierungen

- **Skalierung:** Mit dieser Option können Sie die Tastatur so verkleinern bzw. vergrößern, dass sie sich der Größe des Zielbildschirms anpasst. Ein Wert von 100% steht für die Standardgröße des jeweiligen Tastaturtyps.
- **Zeige Hinweise:** Wenn diese Option aktiviert ist, wird ein Tipp in der Titelleiste der Tastatur angezeigt. Tragen Sie einen Text oder einen String-Tag in das Feld **Tipp anzeigen** ein, der dann als Standardtipp dient.
- **Min/Max-Felder aktivieren:** Die erlaubten Mindest- und Höchstwerte werden am unteren Rand der Tastatur angezeigt. Für Objekte werden diese Werte im Dialog Objekteigenschaften festgelegt. Ansonsten werden die Eigenschaften Min und Max des zugehörigen Tags verwendet.



Die Felder **Min** und **Max** werden nur auf dem Tastaturtyp **Tastentfeld** angezeigt. hierzu muss der zugeordnete Tag vom Typ Integer oder Real sein. Ist Min größer als Max, wird die Eingabe deaktiviert. Wenn die am Objekt eingestellten Min/Max Werte sich von den Min/Max Werten im Tag unterscheiden, versucht WinStudio, die Größe der Eingabe entsprechend umzurechnen.

- Das Aktivieren dieser Option ermöglicht es, Tooltips, die für die Objekte auf den Bildschirmen konfiguriert wurden, anzuzeigen, wenn diese auf dem Web Thin Client (Web-Browser) betrachtet werden.
- **Auswahlfeld: Auto Bild-Skalierung**
Aktivieren Sie diese Option, um die in einem Web-Browser-Fenster angezeigten Bildschirme automatisch in der Größe anzupassen. Wenn Sie diese Option verwenden, passt sich die Größe des Web-Browser-Fensters unabhängig von der Auflösung an den Bildschirm an.



Die Option Automatische Bild-Skalierung gilt nicht für Web-Browser, die unter Windows CE/ Windows 7 embedded compact laufen.

- **Auswahlfeld: Aktiviere Dateikompression**
Die im Ordner \Web der Anwendung gespeicherten Dateien werden bei Aktivierung dieser Option komprimiert. Dies ist sinnvoll, um die Download-Zeit bei langsamen Verbindungen zu reduzieren

Einstellungen zur IP-Sicherheit

IP-Sicherheit: Klicken Sie auf diese Schaltflächen, um den Dialog IP-Sicherheit zu öffnen.

Visualisierungen

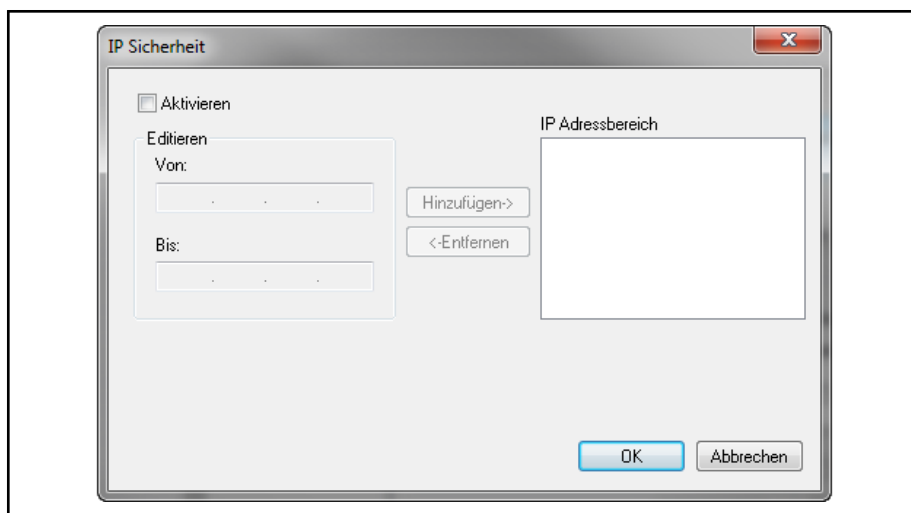


Abb. 3-10: Dialog IP-Sicherheit

Mit Hilfe der Parameter dieses Dialogs geben Sie den Bereich von IP-Adressen für die Rechner an, die als Thin Clients auf Ihr Projekt zugreifen dürfen.

Klicken Sie das Kontrollkästchen **Aktivieren** an, und wenn die Parameter des Bereichs **Editieren** aktiv werden, geben Sie die IP-Adressen in die Felder **Von** und **Bis** ein, um den IP-Adress-Bereich zu benennen. Mit den Schaltflächen **Hinzufügen** und **Entfernen** verschieben Sie die IP-Adressen in die Liste **IP-Adressen-Bereich**. Wenn ein Thin Client versucht sich mit dem Server zu verbinden, prüft dieser den definierten Adressbereich und verhindert eine Verbindung mit der Thin-Client-Station, sofern deren Adresse nicht vorhanden ist.



Standardmäßig gilt die IP-Sicherheit nur für Thin Clients, die sich mit dem Datenserver verbinden. Jedoch können Sie IP-Sicherheit auch für eine Datenbanksynchronisation von Projekten, die auf verschiedenen Stationen laufen implementieren. Fügen Sie hierfür den folgenden Parameter in Ihre Projektdatei (**project_name.app**) ein:

[TCP]

UseWebIPSecurity=1

Weitere Einstellungen

Erweitert: Klicken Sie auf diese Schaltfläche, um einen Dialog zu öffnen, in dem die weiteren Web-Einstellungen bearbeiten werden können. In den meisten Fällen müssen diese Einstellungen nicht geändert werden. Abhängig von der in Ihrem Projekt verwendeten Architektur haben Sie jedoch die Flexibilität, weitere Einstellungen zu konfigurieren.

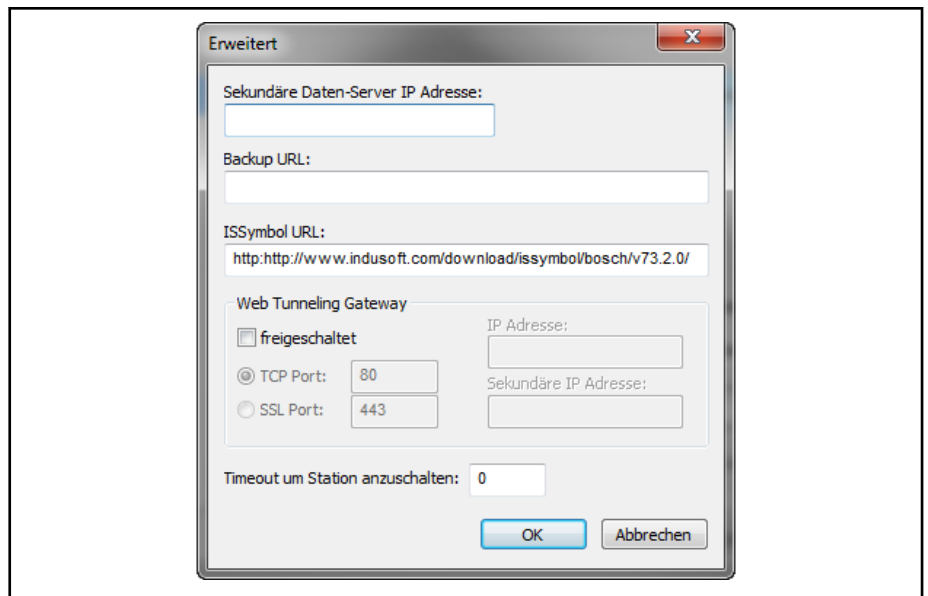


Abb. 3-11: Dialog Weitere Einstellungen

- **Feld IP-Adresse des sekundären Datenservers:** Tragen Sie hier die IP-Adresse (bzw. den Host-Namen) Ihrer sekundären Datenserverstation ein. Die Datenserverstation ist der Rechner bzw. das Gerät, auf dem das TCP/IP-Server-Modul von "WinStudio" ausgeführt wird. Dieses Feld muss ausgefüllt werden, wenn Sie von Thin Clients ausgehend redundante Datenserver verwenden. Wenn der primäre Datenserver ausfällt, versucht der Thin Client automatisch, sich mit dem sekundären Datenserver zu verbinden.
- **Feld Backup-URL:** Tragen Sie die URL desjenigen Orts ein, an dem die Web-Dateien gespeichert sind (Dateien aus dem "Web"-Unterordner Ihres Projektordners). Diese URL wird eingesetzt, um die Dateien vom sekundären Webserver herunterzuladen, wenn der primäre Webserver nicht verfügbar ist.
- **Feld ISSymbol-URL:** Wenn sich der Thin Client mit dem Server verbindet, versucht er, das Control ISSymbol zu laden. Wenn ISSymbol auf dem lokalen Rechner (Thin Client) nicht registriert ist, versucht der Browser, es von der in diesem Feld spezifizierten URL herunterzuladen. Die Standard-URL ist eine Website, auf der "Bosch Rexroth" die aktuellste Version von ISSymbol zum Herunterladen vorhält. Möglicherweise müssen Sie einen anderen Ort konfigurieren, vor allem wenn der Thin-Client-Rechner nicht mit dem Internet verbunden ist. Die Datei **ISSymbolVM.cab** (abgelegt im Unterverzeichnis **\BIN** von "WinStudio") muss bei der URL verfügbar sein, die in diesem Feld konfiguriert ist.



Wenn die Thin-Client-Stationen keinen Zugang zum Internet haben, wird empfohlen, die Datei **ISSymbolVM.cab** auf der Webserverstation verfügbar zu machen und die URL hierfür in diesem Feld zu konfigurieren.

Visualisierungen



Web-Browser, die unter dem Betriebssystem Windows CE laufen, sind nicht in der Lage, das Control ISSymbol (**ISSymbolCE.ocx**) automatisch von einer Remote-Stelle herunter zu laden. **ISSymbolCE.ocx** muss manuell in dem Windows-CE-Gerät registriert werden, das als Thin Client eingesetzt werden soll.

- **Web-Tunnel-Gateway:** Wählen Sie diese Option aus, um den Web-Tunnel-Gateway freizugeben. Je nach Architektur Ihres Projekt müssen Sie unter Umständen die Verbindung der Thin-Client-Rechner zum Daten-server über ein Web-Tunnel-Gateway leiten.
- **TCP-Port:** Wählen Sie diese Option aus, wenn Sie HTTP mit dem Webserver von Microsoft IIS verwenden. Sie können hier den von Ihrem HTTP-Webserver verwendeten TCP-Port angeben ("80" ist der Standard-TCP-Port für das HTTP-Protokoll).
- **SSL-Port:** Wählen Sie diese Option, wenn Sie SSL (Secure Socket Layer) mit dem Webserver von Microsoft IIS verwenden. Sie können hier den von Ihrem HTTPS-Webserver verwendeten TCP-Port angeben ("443" ist der Standard-TCP-Port für das HTTPS-Protokoll).
- **IP-Adresse:** Die IP-Adresse des Webserver-Rechners, auf dem der Web-Tunnel-Gateway ausgeführt wird. Hierbei muss es sich um die IP-Adresse des Webserver handeln, der von der/den Thin-Client-Station(en) aus zugänglich ist.
- **Sekundäre IP-Adresse:** Die IP-Adresse des Webserver-Rechners, auf dem der sekundäre Web-Tunnel-Gateway ausgeführt wird. Hierbei muss es sich um die IP-Adresse des sekundären Webserver handeln, der von der/den Thin-Client-Station(en) aus zugänglich ist. Dieses Feld muss konfiguriert werden, wenn Sie redundante Webserver einsetzen.



Die meisten Web-Einstellungen lassen sich dynamisch über die Funktion SetWebConfig ändern. Diese ist insbesondere dann nützlich, wenn Sie ein Projekt nur einmal anlegen und dann sicher gehen wollen, dass die Web-Einstellungen automatisch konfiguriert werden, wenn Sie Ihr Projekt auf verschiedenen Arbeitsstationen mit verschiedenen IP-Adressen laufen lassen.

Projekt-Status

Für eine korrekte Ausführung einiger WinStudio-Funktionen müssen die entsprechenden Laufzeitkomponenten aktiviert werden. Eine Übersicht aller WinStudio-Laufzeitkomponenten ist über **WinStudio ► Status...** aufrufbar.

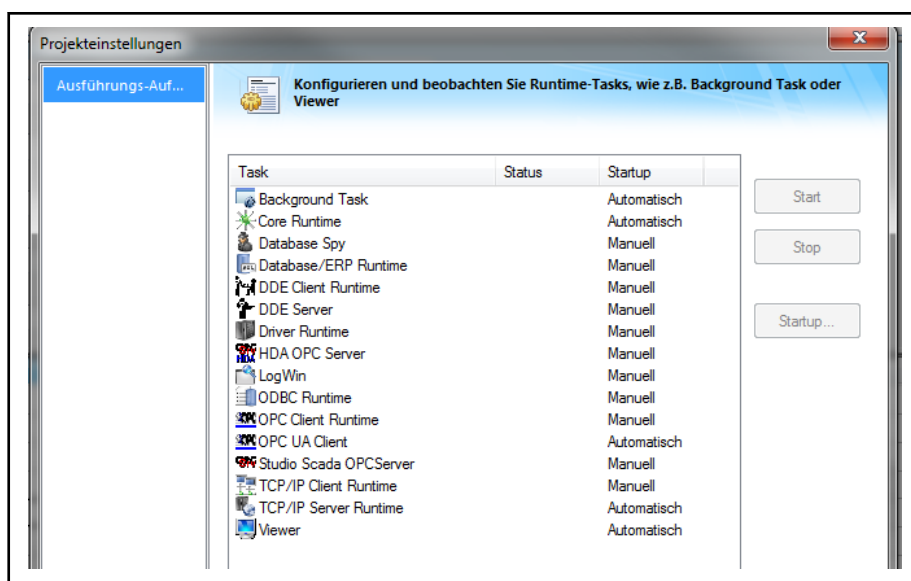


Abb. 3-12: Dialog: Projekteinstellungen (Registerkarte: Ausführungs-Aufrufe)

Im Statusdialog werden alle verfügbaren WinStudio-Laufzeitkomponenten angezeigt. Für jede Laufzeitkomponente wird der Status sowie der Startmodus angezeigt. Standardmäßig werden folgende Laufzeitkomponenten beim Start aktiviert:

- Background Task
- Core Runtime
- TCP/IP Server

Sie haben über Start, Stop und Startup die Möglichkeit, Tasks während der Laufzeit zu starten und zu stoppen, sowie festzulegen, welche Tasks automatisch bzw. manuell gestartet werden sollen.

WinStudio-Projekt laden

Das aktuelle WinStudio-Projekt inklusive aller zugehörigen WinStudio-Objekte kann durch ein anderes WinStudio-Projekt ersetzt werden, beispielsweise um ein vorhandenes WinStudio-Projekt als Vorlage zu verwenden. Um das aktuelle WinStudio-Projekt zu ersetzen, führen Sie die Funktion **WinStudio ► Projekt laden...** aus. Es öffnet sich ein Datei-Browser, mit dem Sie eine Projektdatei (Dateiendung "*.app") auswählen. Nach Bestätigung der Dateiauswahl werden alle aktuellen WinStudio-Elemente gelöscht und neue Elemente des gewählten Projekts eingefügt.

3.4.4 WinStudio-Bilder

Erstellen und Bearbeiten von WinStudio-Bildern

Um ein WinStudio-Bild zu erstellen, wählen Sie **WinStudio ► Neues Bild**. Es wird ein neues Element mit dem Namen "Screen" zum Ordner "WinStudio" hinzugefügt. Diesen Namen können Sie aktuell oder zu einem späteren Zeitpunkt ändern.

Um ein vorhandenes WinStudio-Bild zu bearbeiten, öffnen Sie mit einem Doppelklick auf das Element im Projektbaum den WinStudio-Bildeditor.



Einige Elemente können nur bei Aktivierung einer WinStudio-Engineering-Lizenz 1,5k oder höher genutzt werden. Lizenzabhängige Elemente sind bei nicht ausreichender Lizenz deaktiviert.

Beim Öffnen eines WinStudio-Bildes werden in der Voreinstellung zusätzliche Symbolleisten mit häufig benötigten Funktionen angezeigt. Sie können die

Visualisierungen

Voreinstellung über **Extras ► Anpassen...** ändern und die angebotenen Symbolleisten ausblenden oder weitere Symbolleisten hinzufügen. Alle Editierfunktionen eines WinStudio-Bildes sind zusätzlich über die Menüzeile von IndraWorks Engineering über **WinStudio-Bild** verfügbar.

Bildeigenschaften

In den Bildeigenschaften stehen weitere Optionen zur Änderung der Bildgröße und zur Verwendung einer Hintergrundgrafik zur Verfügung. Sie können den Bildeigenschaftendialog über das Kontextmenü des Bildeditors über den Menüpunkt **Bildeigenschaften** öffnen.

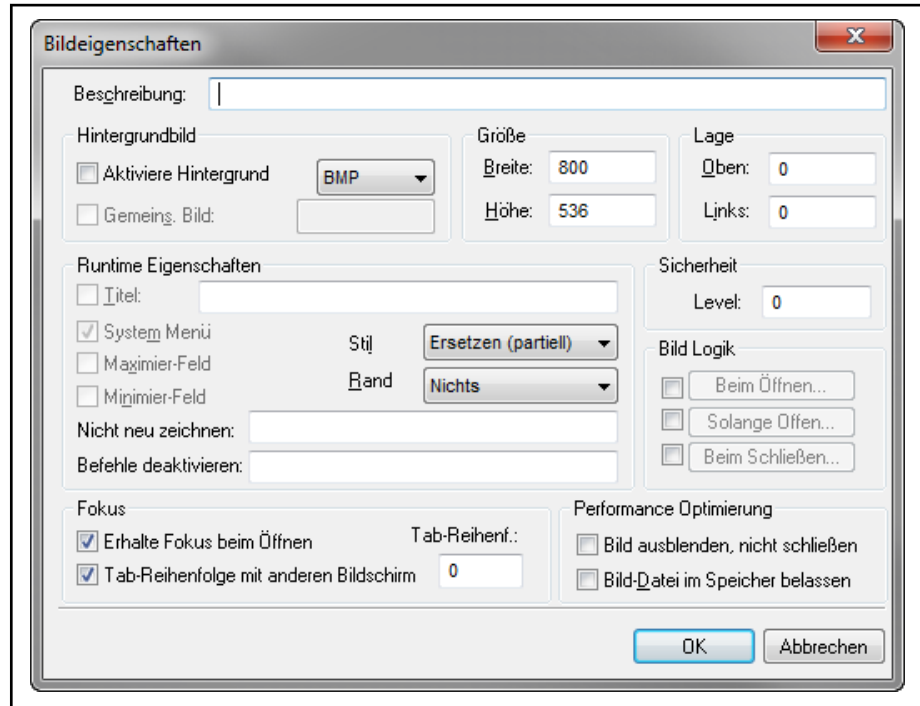


Abb. 3-13: Attribute eines WinStudio-Bildes

Dynamische Objekt-Eigenschaften

Einige Eigenschaften (Farbe, Größe, Position) von WinStudio-Objekten, die in ein WinStudio-Bild eingefügt wurden, können dynamisch geändert werden. Außerdem lassen sich Funktionen konfigurieren, die beim Aktivieren des WinStudio-Objekts in IndraWorks Operation ausgeführt werden.

Einem WinStudio-Objekt können mehrere dynamische Eigenschaften zugewiesen werden, indem Sie diese einzeln aktivieren. Folgende dynamische Eigenschaften stehen zur Verfügung:

- **Kommando:** Funktion, die beim Aktivieren des Elements ausgeführt werden soll
- **Hyperlink:** öffnet beim Aktivieren einen Hyperlink
- **Balkendiagramm:** stellt einen Variablenwert als Balkendiagramm dar
- **Textein-/ausgabe:** zeigt den Wert einer WinStudio-Variablen an oder übernimmt einen eingegebenen Wert
- **Farbumschlag:** ändert die Farbe in Abhängigkeit einer WinStudio-Variablen
- **Position:** ändert die X- und Y-Position des Objekts in Abhängigkeit einer WinStudio-Variablen
- **Größe:** ändert Breite/Höhe des Objekts in Abhängigkeit einer WinStudio-Variablen

- **Linienrotation:** rotiert eine Linie in Abhängigkeit einer WinStudio-Variablen

Objekteigenschaften und Variablenauswahl

Die Eigenschaften eines WinStudio-Objekts lassen sich mit Hilfe des Eigenschaftendialogs bearbeiten. Hier können grundlegende Eigenschaften (z.B. Linienbreite) editiert werden, wobei der Umfang der angebotenen Eigenschaften vom jeweiligen Objekttyp abhängt.

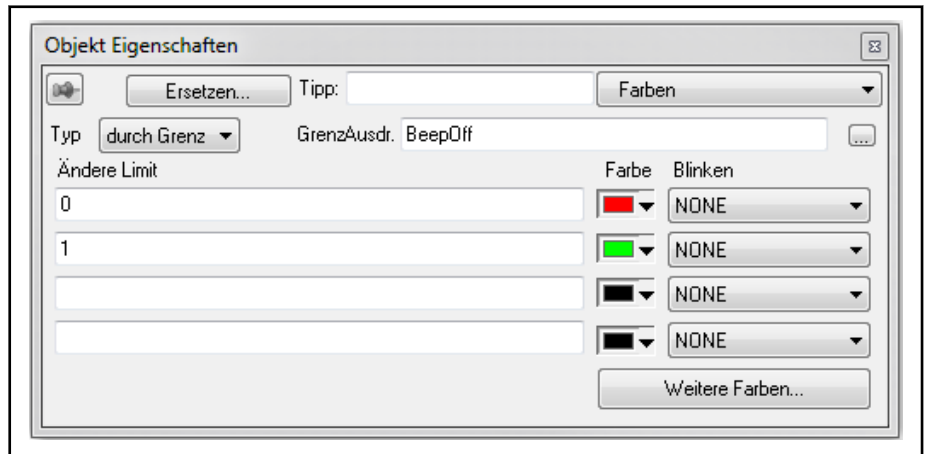


Abb. 3-14: Objekteigenschaften

Im Auswahldialog können Variablen aus folgenden Bereichen ausgewählt werden:

- **Projekt** - Im Rahmend der Projektierung angelegte Tags aus der Anwendungsvariablenbank
- **System** - WinStudio Systemvariablen (z.B. Minute) aus der internen Variablenbank
- **Verteilt** - Variablen aus automatischer Treiberblattgenerierung (falls konfiguriert) aus der verteilte Variablenbank

3.4.5 WinStudio-Variablen

WinStudio bietet die Möglichkeit, Aktionen (z.B. die Ausführung einer Berechnung) in Abhängigkeit einer Variablen ausführen zu lassen, Ergebnisse von Berechnungen Variablen zuzuweisen sowie WinStudio-Variablen mit Prozessvariablen zu verknüpfen.

Im Ordner "Variablen" stehen folgende Variablenkategorien zu Verfügung:

- **Interne Variablen** vordefinierte, nicht änderbare Variablen
- **Verteilte Variablen** Variablen, die mit einer Prozessvariablen verknüpft sind (automatische Treiberblattgenerierung)
- **Anwendungsvariablen** eigene, anwendungsspezifische Variablen

Klassen WinStudio stellt folgende Datentypen für eigene Variablen zur Verfügung:

- **Bool**
- **Integer** - 32 bit, ganzzahlig mit Vorzeichen
- **Real** - 64 bit, Gleitkomma mit Vorzeichen
- **String** - Zeichenkette bis zu 1024 Zeichen

Darüber hinaus lassen sich eigene Datentypen, sogenannte "Klassen" definieren, die sich aus den Standardtypen zusammensetzen.

Visualisierungen

	Name	Type	Description
1	 yPos	Integer	x-Position
2	 xPos	Integer	y-Position
3	 Title	String	Title
*		Integer	
*		Integer	
*		Integer	
*		Integer	
*		Integer	

Abb. 3-15: WinStudio-Klasseneditor

Im Klasseneditor können Sie in der Spalte "Name" beliebige Elementnamen hinzufügen und unter "Typ" einen Datentyp zuweisen. Nach dem Speichern der neuen Datenklasse können Sie diese für WinStudio-Anwendungsvariablen verwenden.

Kommunikation

WinStudio-Variablen können mit SPS-Variablen verknüpft werden, um den Wert der SPS-Variablen in einem WinStudio-Bild anzuzeigen, in Berechnungen zu verwenden oder zu verändern. Die Verknüpfung von WinStudio-Variablen mit SPS-Variablen erfolgt über sogenannte Treiberblätter. Ein Treiberblatt enthält neben der Variablenliste verschiedene Einstellungen, die für die Kommunikation mit der SPS benötigt werden. Die Treiberblattbezeichnung besteht aus einer eindeutigen Nummer und einem optionalen Beschreibungstext. Für die Kommunikation stehen sowohl OPC-DA, OPC-IwSCP, OPC-UA als auch BRWS-Treiber zur Verfügung.



Achten Sie darauf, dass die Wertebereiche der WinStudio-Variablen die Wertebereiche der Steuerungsvariablen, die Sie mit Ihrer Anwendung verknüpfen, abdecken

3.4.6 Verlinkte Symbole

WinStudio-Bibliothek

WinStudio bietet eine umfangreiche Bibliothek von vorgefertigten Symbolen, die in WinStudio-Bilder eingefügt werden können. Darüber hinaus können auch selbst definierte Symbole, sog. Anwendersymbole, erzeugt und wiederverwendet werden.

Die WinStudio-Bibliothek wird in einem Werkzeugfenster im unteren Bereich von IndraWorks Engineering dargestellt (siehe nachfolgende Grafik). Die Bibliothek besteht aus einer Anzahl von Ordnern, in denen die WinStudio-Symbole sortiert sind.

Symbol einfügen

Öffnen Sie zunächst das WinStudio-Symbolfenster über den WinStudio-Knoten **WinStudio ► Werkzeuge ► WinStudio-Bibliothek**. Um ein Symbol in ein WinStudio-Bild einzufügen, öffnen Sie zunächst das WinStudio-Bild. Anschließend wählen Sie ein Symbol, indem Sie es im Vorschaufenster selektieren. Mit einem Mausklick in den WinStudio-Bildeditor fügen Sie das Symbol an der gewünschten Position ein.

Symbol parametrieren

Jedes WinStudio-Symbol besitzt mehrere Parameter, die entsprechend anzupassen sind. Der Parametrierungsdialog öffnet sich durch einen Doppelklick auf das Symbol im WinStudio-Bild. Alle Parameter eines Symbols sind nach Kategorien sortiert und werden durch Anklicken der Kategorie angezeigt. Einzelne Parameter können Sie ändern, indem Sie in die Spalte "Wert" klicken und den gewünschten Wert eingeben. Sie können hier entweder einen festen Wert oder eine WinStudio-Variable eintragen.

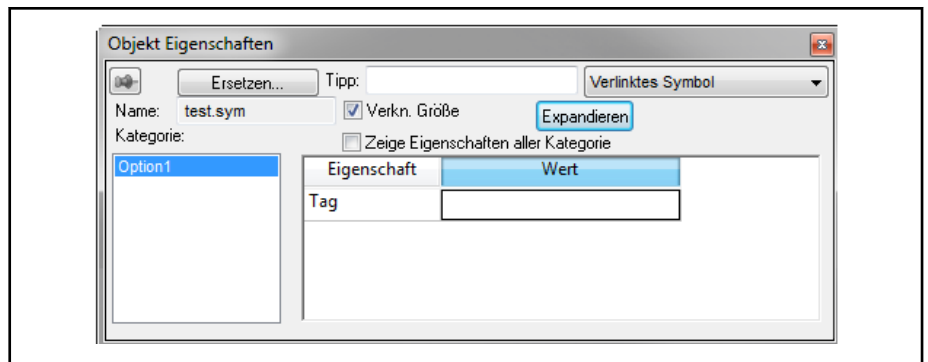


Abb. 3-16: Parametrierungsdialog eines WinStudio-Symbols

Anwendersymbole

Zusätzlich zu den vordefinierten WinStudio-Symbolen können Sie eigene Symbole erstellen. Die Erstellung eines Symbols ist insbesondere nützlich, wenn ein Objekt mehrfach in einem WinStudio-Projekt verwendet werden soll. Wenn Änderungen am Symbol durchgeführt werden, sind diese in allen WinStudio-Bildern wirksam, in denen das Symbol verwendet wird. Das Symbol muss also nicht erneut in die betroffenen WinStudio-Bilder eingefügt werden.

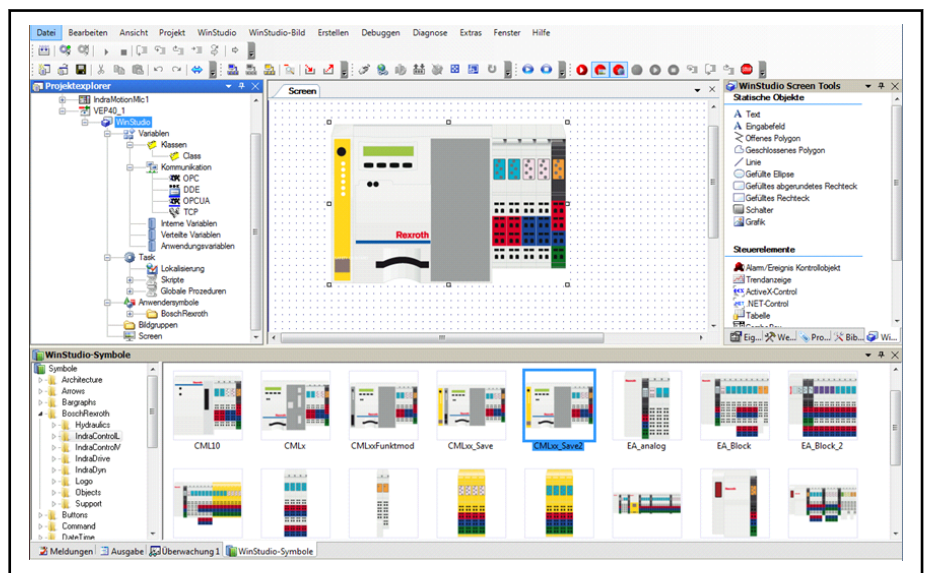


Abb. 3-17: IndraWorks WinStudio mit geöffnetem Symbolfenster

Symbol erstellen

Um ein Symbol zu erstellen, fügen Sie zunächst ein oder mehrere Objekte in ein WinStudio-Bild ein, markieren Sie diese und führen Sie im Kontextmenü die Funktion **Gruppe** aus. Anschließend führen Sie im Kontextmenü die Funktion **Erzeuge verlinktes Symbol** aus und geben Sie im anschließenden Dateidialog einen Symbolnamen ein.

Visualisierungen

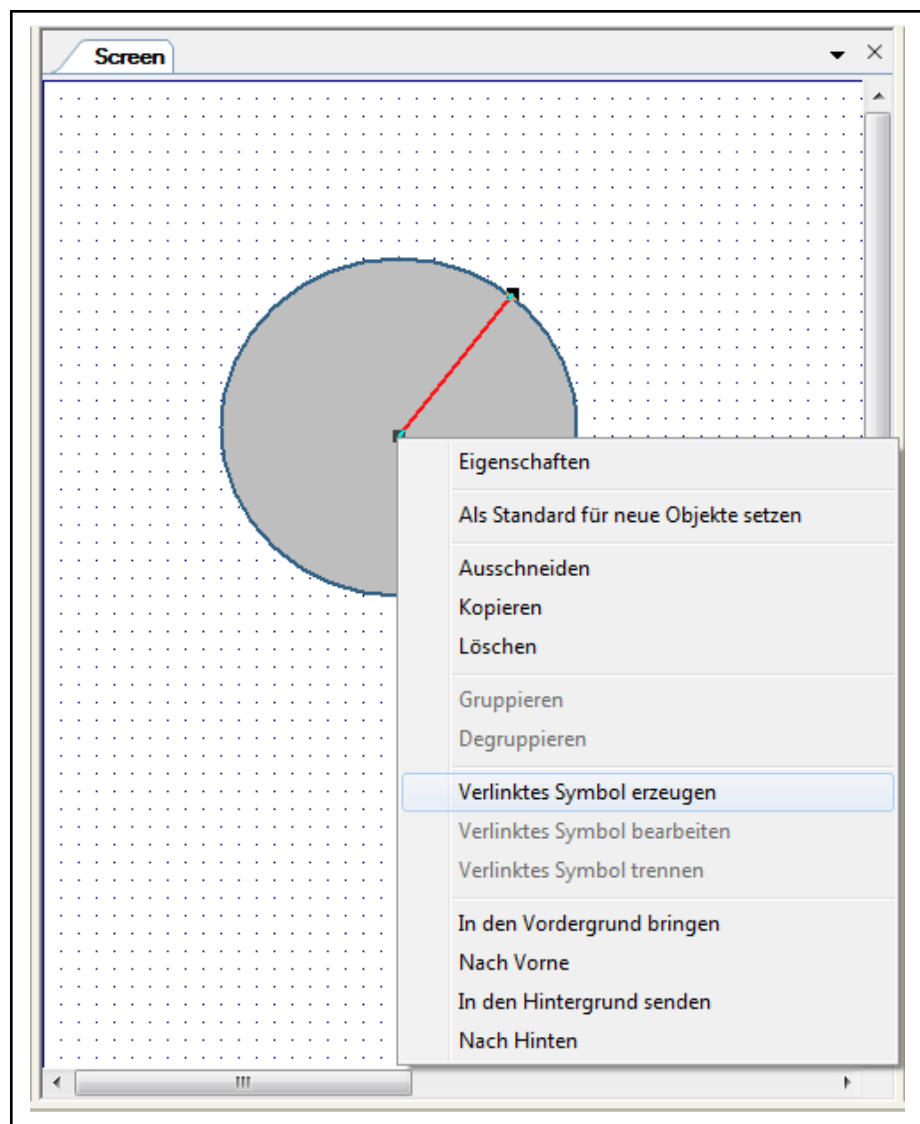


Abb. 3-18: Erstellung eines Anwendersymbols

Symbol in Bibliothek übernehmen

Ein erstelltes WinStudio-Symbol ist zunächst nur im selben Projekt verfügbar. Wenn Sie dieses auch in anderen Projekten verwenden möchten, können Sie dieses in die WinStudio-Bibliothek übernehmen, indem Sie das Element im Projektbaum unter **WinStudio ► Anwendersymbole** anklicken und im Kontextmenü die Funktion **In WinStudio-Bibliothek übernehmen** ausführen. Das Symbol wird hierbei in den Ordner "User Objects" der WinStudio-Bibliothek kopiert.

Symbol mit Passwort schützen

Sie können eigene Anwendersymbole mit einem Passwort versehen, so dass es nicht von unberechtigten Personen bearbeitet werden kann. Ein geschütztes Anwendersymbol kann somit in WinStudio-Bildern verwendet und parametrisiert werden, ohne die interne Funktionalität einsehen zu können.

Wenn Sie ein geschütztes Anwendersymbol öffnen, erscheint ein Eingabedialog für das zugehörige Passwort. Nur wenn Sie das korrekte Passwort eingeben, öffnet sich der Editor. Weitere Informationen zum Passwortschutz von Anwendersymbolen finden Sie in der IndraWorks Online-Hilfe "Rexroth WinStudio 7.3 Technische Referenz".

3.4.7 WinStudio-Tasks

Mathematikblätter

Ein WinStudio-Mathematikblatt dient zur Programmierung von Abläufen, Variablenzuweisungen und der Ausführung sonstiger WinStudio-Funktionen.

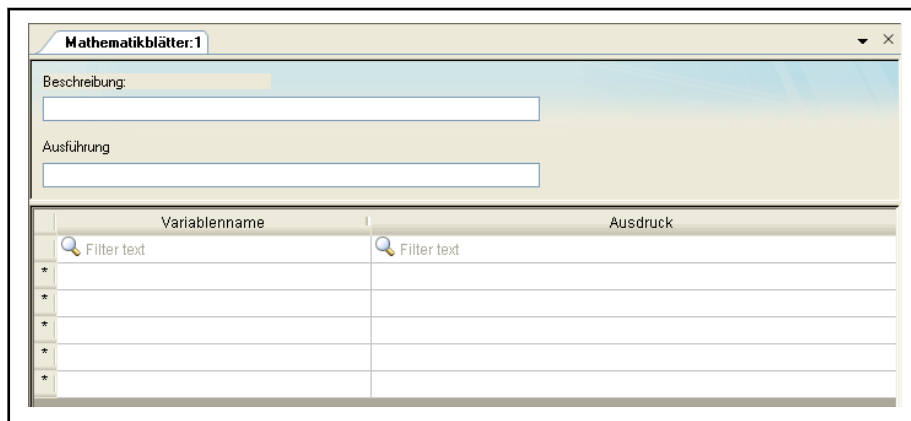


Abb. 3-19: WinStudio-Mathematikblatt

Das Mathematikblatt wird zyklisch ausgeführt, wenn die Variable oder der Ausdruck im Feld "Ausführung" den Wert TRUE besitzt.

Weitere Informationen über Mathematikblätter finden Sie in der IndraWorks Online-Hilfe "Rexroth WinStudio 7.3 Technische Referenz"

Alarme

Mit Hilfe von WinStudio-Alarmen können Sie Meldungen definieren, die ausgegeben werden, wenn eine Variable einen Wert über- oder unterschreitet.

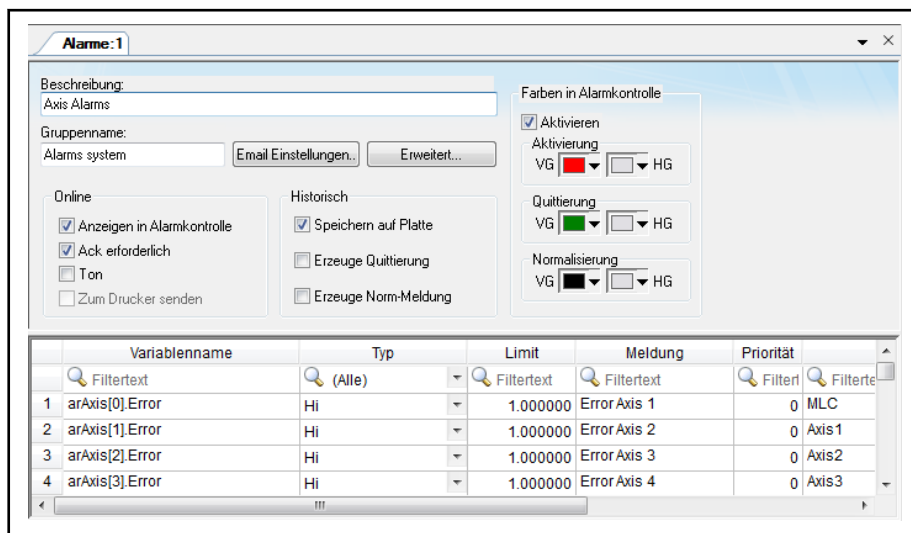


Abb. 3-20: WinStudio-Alarmeditor

Alarmmeldungen lassen sich mit Hilfe des Alarm-Steuerelements in WinStudio-Bildern anzeigen.

Details zur Konfiguration von Alarmen finden Sie in der IndraWorks Online-Hilfe "Rexroth WinStudio 7.3 Technische Referenz".

Rezepte

WinStudio ermöglicht das Speichern von Variablenwerten in Dateien sowie das Laden von Variablenwerten aus Dateien mit Hilfe sogenannter Rezepte.

Visualisierungen

Ein WinStudio-Rezept stellt somit eine Liste von Variablen und der Zuordnung von Werten dar.

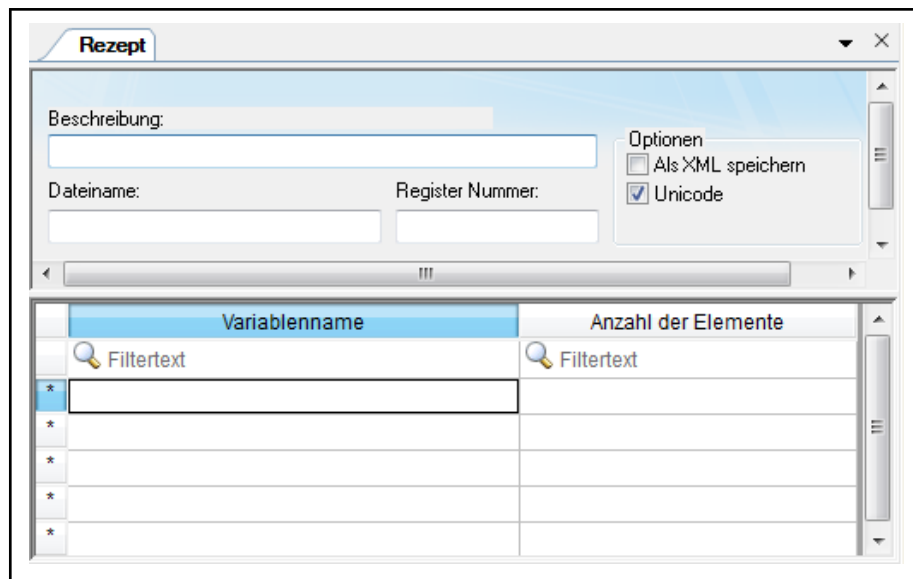


Abb. 3-21: WinStudio-Rezepteditor

Im Rezepteditor können in der Spalte "Variablenname" die Namen der Variablen eingetragen werden, deren Werte gespeichert oder geladen werden sollen. Falls es sich bei einer Variablen um ein Array handelt, ist in der Spalte "Anzahl der Elemente" die jeweilige Elementanzahl einzutragen.

Details zur Konfiguration von Rezepten finden Sie in der IndraWorks Online-Hilfe Rexroth "WinStudio 7.3 Technische Referenz".

Reports

Ein WinStudio-Report dient zur Erstellung einer Dokumentation, die beliebigen Text, sowie aktuelle Werte von WinStudio-Variablen enthalten kann. Der Report kann sowohl in eine Datei als auch auf einem Drucker ausgegeben werden.

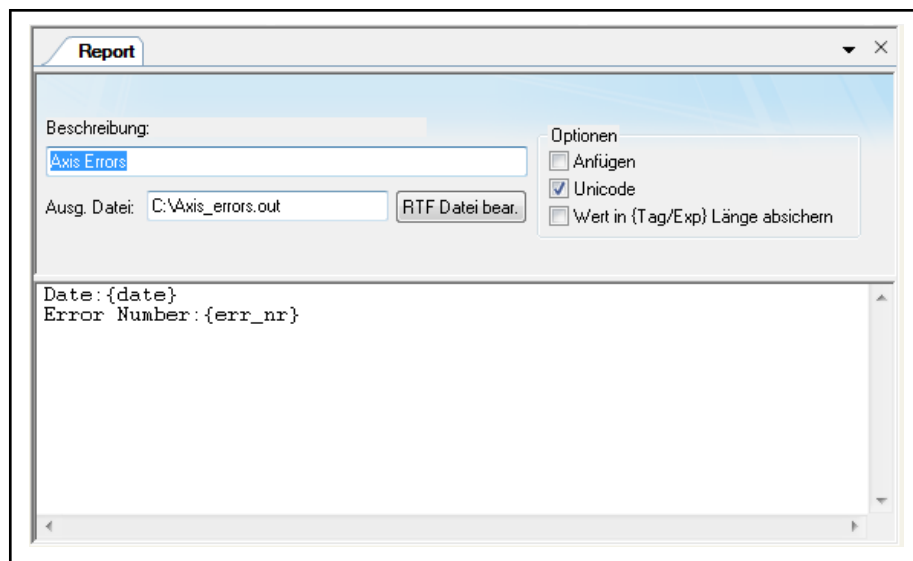


Abb. 3-22: WinStudio-Reporteditor

Weitere Informationen finden Sie in der IndraWorks Online-Hilfe "WinStudio 7.3 Technische Referenz".

Trends

WinStudio-Trenddaten dienen zur Aufzeichnung von WinStudio-Variablen. Diese Variablen können mit Hilfe des Trendobjekts in einem WinStudio-Bild dargestellt werden, um den zeitlichen Verlauf bestimmter Variablen zu beobachten.

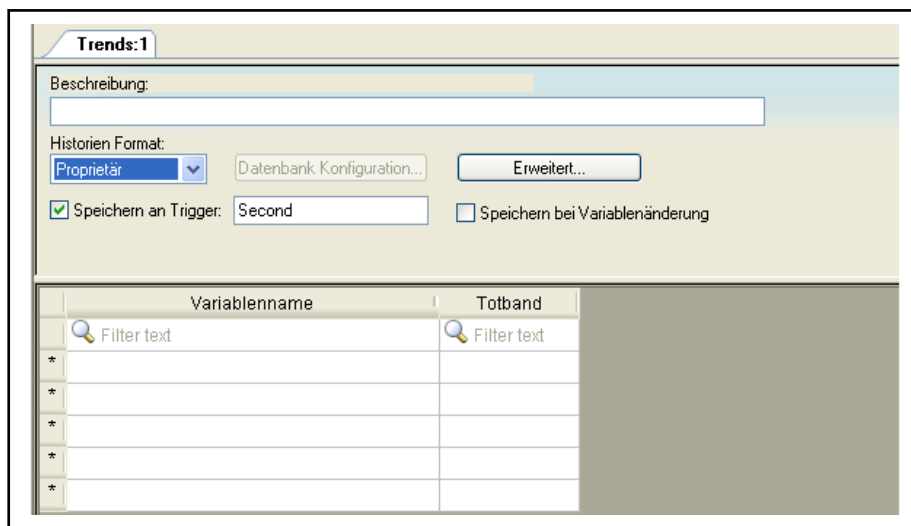


Abb. 3-23: WinStudio-Trendeditor

Details zur Konfiguration von Trend-Datenobjekten und zur Visualisierung von Trend-Daten entnehmen Sie der WinStudio-Online-Hilfe "WinStudio 7.3 Technische Referenz".

Scheduler

Mit Hilfe von WinStudio-Schedules können Sie Aktionen zeit- bzw. ereignis-gesteuert ausführen. Beispielsweise ist es möglich, jeden Tag automatisch einen Report auszugeben oder alle 30 Sekunden eine Variable neu zu initialisieren.

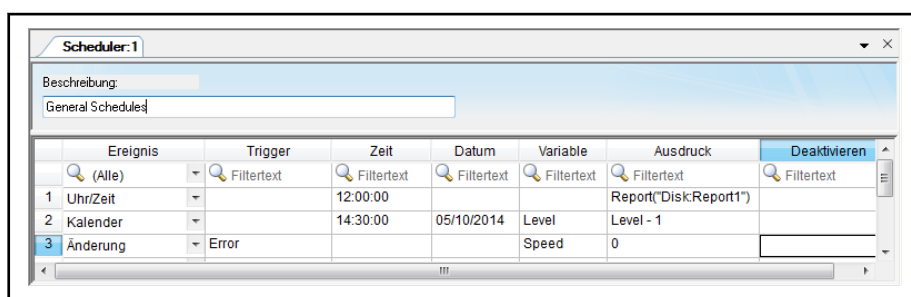


Abb. 3-24: WinStudio Schedule-Editor

Eine detaillierte Beschreibung zur Arbeit mit dem Scheduler finden Sie in der IndraWorks Online-Hilfe "WinStudio 7.3 Technische Referenz".

Externe Datenbanken

Mit Hilfe von Externen Datenbanken lassen sich historische Daten, welche z.B. über Alarmer, Ereignisse oder Trends erzeugt wurden mit externen Datenbanken verknüpfen. Hier werden verschiedene Datenbankschnittstellen unterstützt (ODBC, Microsoft SQL Server, Microsoft Access, etc.)

Eine detaillierte Beschreibung zur Konfiguration einer solchen Schnittstelle finden Sie in der IndraWorks Online-Hilfe "WinStudio 7.3 Technische Referenz".

Visualisierungen

Skripte

Übersicht

WinStudio unterstützt die Erstellung und Ausführung eigener Funktionen in VisualBasic-Skripten. Die VisualBasic-Skriptsprache stellt ein einfaches und flexibles Werkzeug für die Programmierung von Funktionen und Abläufen dar. Eine Einführung in VB (VisualBasic) und eine Sprachreferenz finden Sie im MSDN (MicroSoft Developers Network).

Weitere Informationen zu VB-Skripten und der VB Skriptsprache finden Sie auch in der IndraWorks Online-Hilfe "WinStudio 7.3 Technische Referenz".

Globale Prozeduren

Eigene VB-Funktionen können Sie als "Globale Prozedur" deklarieren und somit eine Funktionsbibliothek erstellen. Funktionen, die unter "Globale Prozeduren" deklariert sind, können in allen WinStudio-Objekten mit VB-Unterstützung (z.B. als Kommando auf Schaltflächen) aufgerufen werden.

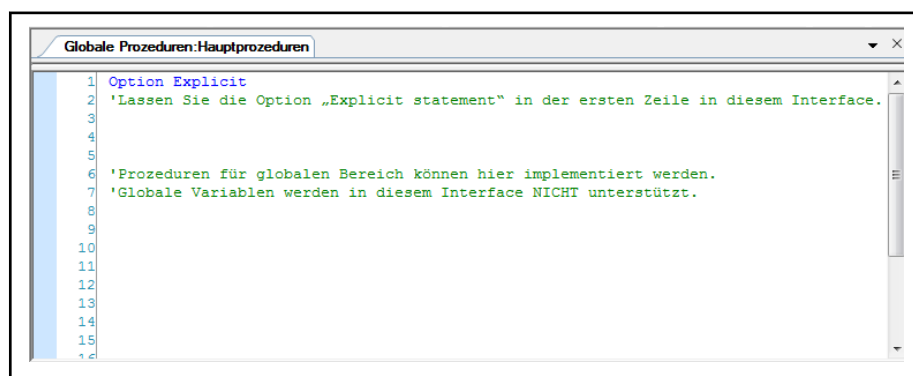


Abb. 3-25: Dialog: Globale Prozeduren (Editor)

Im Editor für globale Prozeduren können Sie beliebig viele Funktionen und Unterfunktionen deklarieren.

Start-Skript

Im Element "Start-Skript" können Sie VB-Funktionen und eigene globale Prozeduren aufrufen, die einmalig beim Start der Applikation ausgeführt werden sollen. Das Start-Skript eignet sich besonders für die Initialisierung von Variablen.

Grafik-Skript

Das VB-Skript "Grafik-Skript" dient zur einmaligen Ausführung von VB-Anweisungen beim Starten oder Beenden des WinStudio-Grafik-Moduls oder zur kontinuierlichen Ausführung während der Laufzeit des Grafik-Moduls. Das Grafik-Skript enthält bereits die notwendigen Unterfunktionen, in die Sie eigenen VB-Code einfügen können. Für eine korrekte Ausführung der Funktionen im Grafik-Skript dürfen die Funktionsnamen nicht geändert werden. Sie können jedoch weitere Funktionen hinzufügen, die in den vordefinierten Grafik-Funktionen aufgerufen werden können.

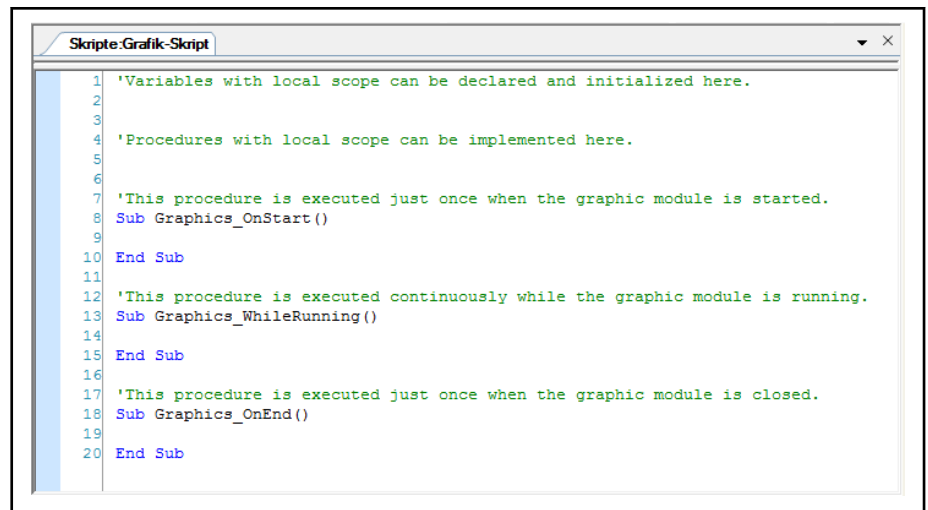


Abb. 3-26: Dialog: Grafik-Skript (Editor)

Bild-Skripte

Analog zum Grafik-Skript lassen sich auch für jedes WinStudio-Bild VB-Anweisungen programmieren, die einmalig beim Öffnen bzw. Schließen des Bildes oder kontinuierlich ausgeführt werden, während das WinStudio-Bild geöffnet ist. Die Bild-Skripte enthalten bereits die notwendigen Funktionsrumpfe, in die Sie eigenen VB-Code einfügen können. Die Namen der vordefinierten Funktionen dürfen nicht verändert werden.

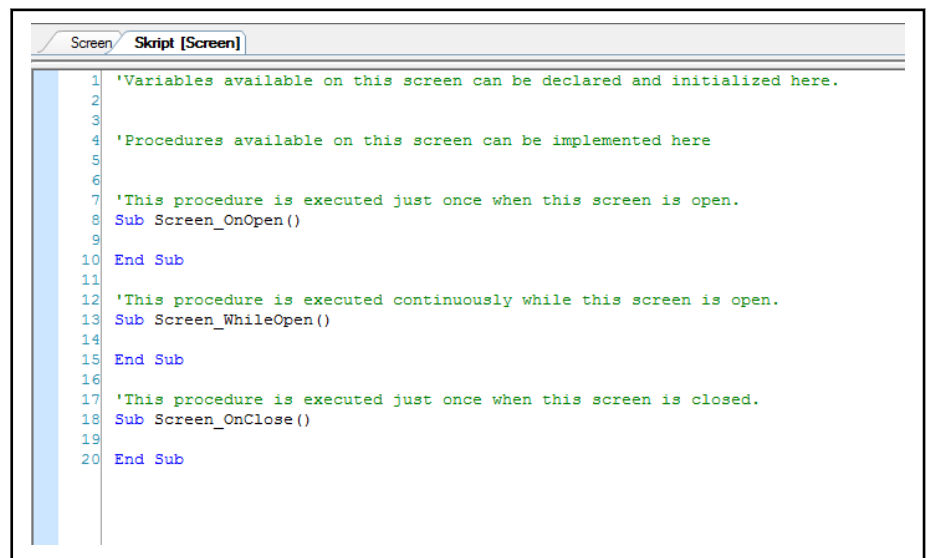


Abb. 3-27: Dialog: Bild-Skript (Editor)

Zu jedem WinStudio-Bild existiert jeweils ein VB-Skript, das Sie im Kontextmenü des geöffneten Bild-Editors über **Bild-Skript** öffnen und bearbeiten können.

Anwender-Skripte

Zusätzlich zu den vordefinierten VB-Skripten ("Start-Skripte", "Grafik-Skript" und "Bild-Skript") können Sie unter dem Knoten "Skripte" beliebig viele weitere VB-Skripte hinzufügen, in denen Sie VB-Anweisungen einfügen können. In Anwender-Skripten ist es jedoch nicht möglich Funktionen zu deklarieren.

Anwender-Skripte erhalten eine fortlaufende Nummer, die die Ausführungsreihenfolge der Anwender-Skripte festlegt. Um ein Skript zur Laufzeit von der Ausführung auszuschließen, tragen Sie im Feld "Ausführung" eine WinStu-

Visualisierungen

dio-Variable oder einen WinStudio-Ausdruck ein. In diesem Fall wird das Skript nur ausgeführt, wenn die Variable bzw der Ausdruck einen Wert größer Null besitzt.

3.4.8 WinStudio-Werkzeuge

Anwendung verifizieren

In allen WinStudio-Elementen lassen sich Variablen verwenden, die für die jeweilige Funktion benötigt werden. Während der Konfiguration können jedoch durch Löschen oder Umbenennen von Variablen Inkonsistenzen auftreten, so dass die betroffenen WinStudio-Funktionen und -Elemente nicht korrekt funktionieren.

Mit Hilfe der Funktion "Anwendung verifizieren" lässt sich überprüfen, ob in WinStudio-Elementen (Mathematikblätter, WinStudio-Bilder, Treiberblätter, Skripte, etc.) Variablennamen verwendet werden, die nicht in der Variablen-datenbank existieren, bzw. Variablen in der Variablen-datenbank deklariert wurden, die in WinStudio Elementen nicht verwendet werden.

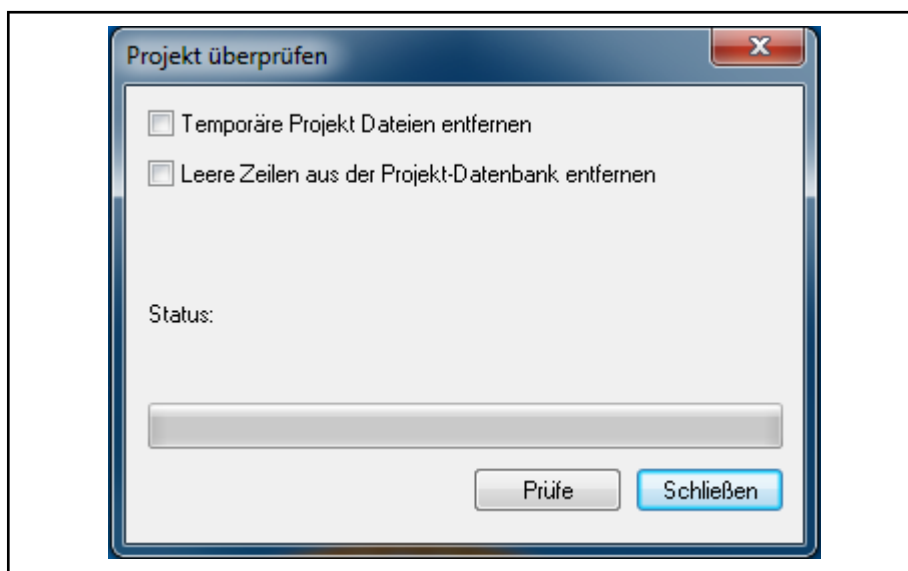


Abb. 3-28: Anwendung verifizieren

Nach dem Schließen des Verifizierungsdialogs werden im Meldungs-fenster von IndraWorks Engineering ggf. Informationen, Warnungen und Fehler-meldungen ausgegeben.

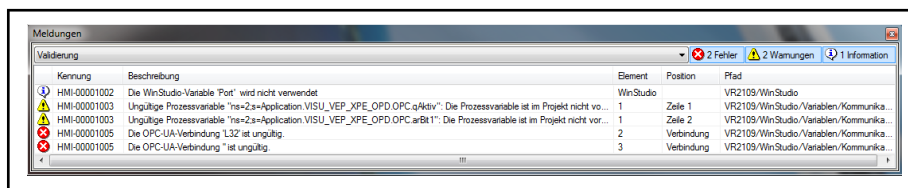


Abb. 3-29: Ausgabe von Meldungen der Funktion "Anwendung verifizieren"

Dort werden Zeilenweise Detailinformationen zu jeder einzelnen Meldung ausgegeben. Durch einen Doppelklick auf die Meldungszeile öffnet sich das entsprechende Werkzeug, mit dem die Meldungsursache korrigiert werden kann. Auf diese Weise können Sie sukzessive alle Inkonsistenzen beheben und ggf. die Funktion "Anwendung verifizieren" erneut starten.

Anwendung starten

Die Funktion einzelner WinStudio-Bilder können Sie mit der Funktion "Anwendung starten" überprüfen, bevor Sie das komplette Projekt auf das Ziel-

Visualisierungen

gerät übertragen und aktivieren. Das Testfenster lässt sich über **WinStudio ► Anwendung starten** öffnen.

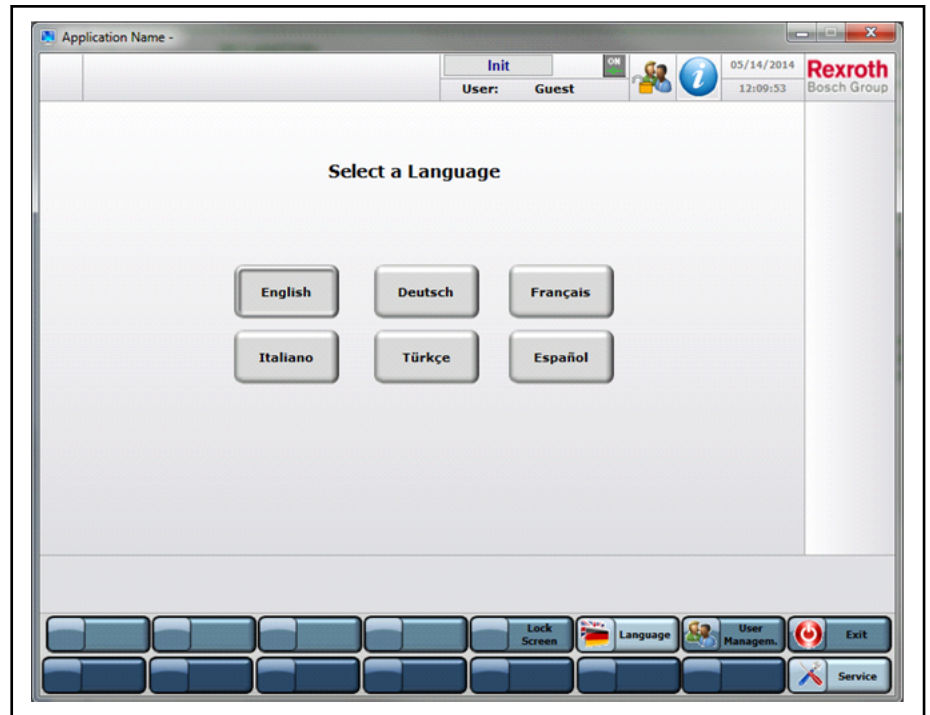


Abb. 3-30: Testen von WinStudio-Bildern

Nach Ausführung von "Anwendung Starten" öffnet sich die WinStudio Viewer Task, sowie entsprechende Hintergrundtasks und Sie können die gesamte Applikation, also das Anzeigen von Variablen und Bildern, Kommandos auf Schaltflächen und die Ausführung der WinStudio Tasks testen. Achten Sie darauf, dass Sie die Applikation über "Anwendung beenden" wieder beenden, da über das alleinige Beenden der Viewer-Task Hintergrundprozesse nicht mit beendet werden.

3.4.9 Datenbank-Spion

Mit Hilfe des WinStudio-Datenbank-Spions lassen sich der Wert und Status einzelner WinStudio-Variablen einer aktiven WinStudio-Applikation überwachen. Im Datenbank-Spion können Sie im Eingabefeld den Namen einer WinStudio-Variablen eintragen und mit der Schaltfläche **Hinzufügen** in die Liste der zu überwachende Variablen übernehmen. In der Variablenliste wird zu jeder Variable der aktuelle Wert sowie der Verbindungsstatus angezeigt.

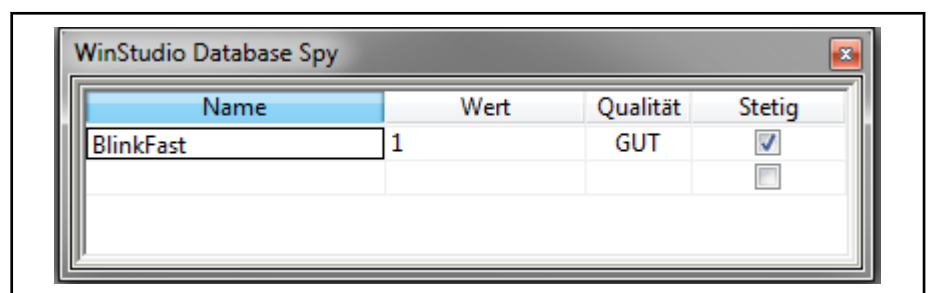


Abb. 3-31: WinStudio-Datenbankspion

Hierbei können Sie zwei Varianten des Datenbankspions verwenden:

1. **Datenbankspion lokal:** verwenden Sie den lokalen Datenbankspion um auf eine lokal laufende WinStudio-Anwendung zuzugreifen

Visualisierungen

2. **Datenbankspion Remote:** Verwenden Sie den Remote-Datenbankspion um auf eine WinStudio Anwendung innerhalb Ihres Netzwerks zuzugreifen

3.4.10 LogWin

Zur Analyse von Abläufen in einer WinStudio-Applikation und insbesondere für die Fehlersuche stellt WinStudio-LogWin ein nützliches Werkzeug dar.

In LogWin lassen sich z.B. Meldungen der Kommunikationsanbindung (z.B. BRWS, OPC), eigene Trace-Meldungen und die Ausführung von Rezepten protokollieren.

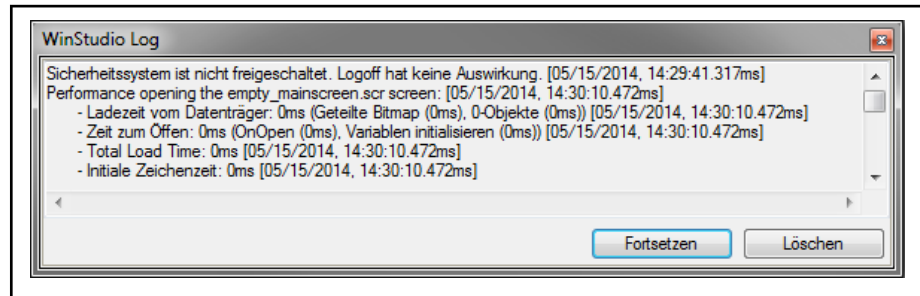


Abb. 3-32: Überwachung von WinStudio-Variablen in LogWin

Im Kontextmenü des LogWin-Fensters können Sie über **Einstellungen** einen Konfigurationsdialog öffnen, mit dem sich die aufzuzeichnenden Meldungen auswählen lassen. Außerdem können WinStudio-Variablen ausgewählt werden bei deren Änderung eine Meldung mit dem aktuellen Wert ausgegeben werden soll.

Weitere Informationen zur Verwendung des LogWin finden Sie in der IndraWorks Online-Hilfe "WinStudio 7.3 Technische Referenz".

3.4.11 Debugger

Zur Analyse von Abläufen und insbesondere für die Fehlersuche innerhalb der Skript Logik einer WinStudio-Applikation stellt WinStudio mit dem Debugger ein weiteres nützliches Werkzeug bereit. Unterstützt werden hierbei Globale Prozeduren, Grafik-, Start-, Anwender- und Bildskripte. Zum Starten einer Debug Session öffnen sie zunächst das betreffende Skript im Skript-Editor und setzen Sie mittels Rechtsklick an den linken, hellblauen Rand des Skriptfensters einen Breakpoint. Schliessen sie nun den Editor und rufen Sie anschliessend im Projektknoten **WinStudio** ▶ **Anwendung Debuggen** auf. Mittels der WinStudio Debug-Leiste, die Sie unter **Ansicht** ▶ **Symbolleiste** ▶ **Debug** finden, können Sie nun das laufende Projekt debuggen.

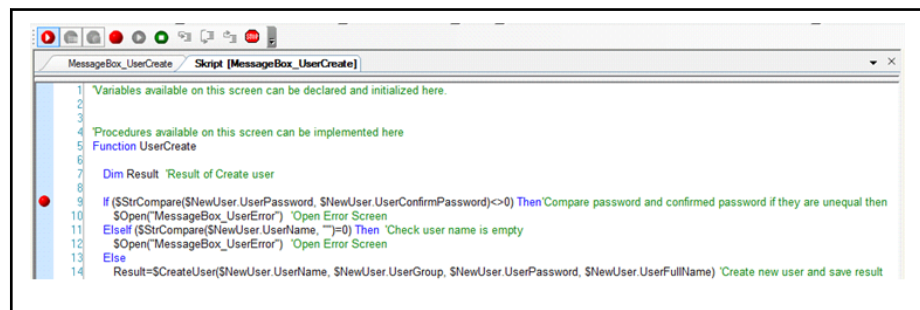
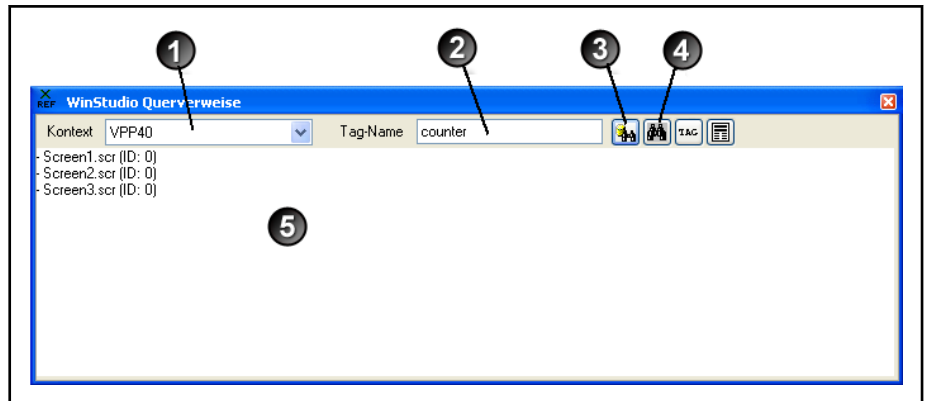


Abb. 3-33: WinStudio Debug Funktionalität

Weitere Informationen zur Verwendung des Debuggers finden Sie in der IndraWorks Online-Hilfe "WinStudio 7.3 Technische Referenz".

3.4.12 Querverweissuche

Um die Verwendung einer bestimmten WinStudio-Variablen zu finden, können Sie die Querverweissuche verwenden.



- 1 Suchumgebung
- 2 Variable
- 3 Suche in Datei
- 4 Suche im Projekt
- 5 Anzeige

Abb. 3-34: Querverweissuche von WinStudio-Variablen

3.4.13 Variablen-Importassistent

Häufig ist es hilfreich, Variablenlisten aus einer anderen WinStudio-Applikation oder einer Datenbank zu übernehmen, so dass diese nicht erneut manuell konfiguriert werden müssen. Die Übernahme von Variablen wird durch den Importassistenten unterstützt.

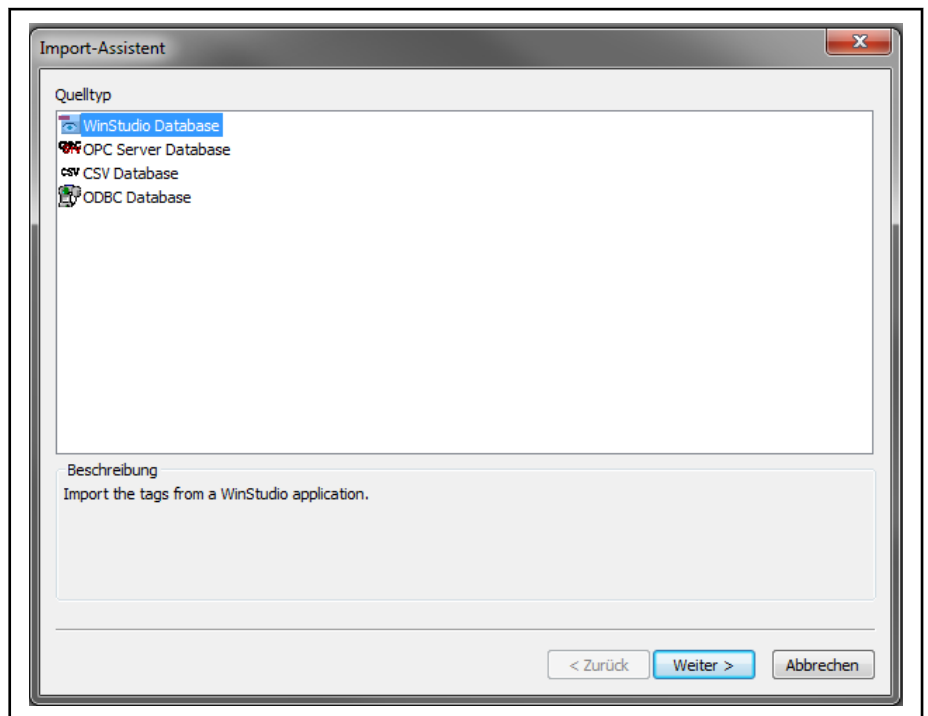


Abb. 3-35: Auswahl einer Datenquelle im WinStudio-Importassistenten

Visualisierungen

3.4.14 Export/Import

Der Ordner "WinStudio" sowie alle zugehörigen Unterelemente unterstützen die Export/Import-Funktionalität von IndraWorks. Diese ermöglicht Ihnen das vollständige WinStudio-Projekt oder auch nur Teile davon zu exportieren und in ein anderes IndraWorks-Projekt zu importieren.

Bitte beachten Sie bei Anwendung der Export/Import-Funktion auf WinStudio-Elemente folgende Besonderheiten:

- Für jedes WinStudio-Element werden die darin verwendeten WinStudio-Variablen ermittelt und automatisch mit exportiert. Beim Import dieses WinStudio-Elements werden somit auch die benötigten WinStudio-Variablen wieder eingefügt. Wenn die zu importierende Variable im neuen Projekt bereits existiert, werden alle zugehörigen Eigenschaften der Variable (z.B. Beschreibungstext, Startwert) überschrieben. Falls sich jedoch der Variablentyp unterscheidet, bleibt die Variable unverändert. Wenn Sie die Variablen dennoch importieren möchten, ändern Sie zunächst den Typ der Variablen oder benennen Sie die Variable um und wiederholen Sie den Importvorgang.

Wenn eine vorhandene Variable beim Import überschrieben wurde oder die neue Variable nicht importiert werden konnte, zeigt der Import-Dialog eine entsprechende Meldung an.

- Wenn es sich beim Typ einer zu exportierenden WinStudio-Variablen um eine selbstdefinierte WinStudio-Datenklasse handelt, muss auch die zugehörige Klassendefinition exportiert werden, damit die Variable beim Import korrekt erstellt werden kann.

3.4.15 WinStudio-Lizenzierung

Einleitung

Das abgestufte Lizenzmodell erlaubt eine optimale Anpassung an die erforderliche Variablenanzahl und den notwendigen Funktionsumfang. Dabei deckt das größere Lizenzpaket den Funktionsumfang des kleineren Paketes ab. Ein Upgrade auf die nächst größere Lizenz ist problemlos möglich.

Das WinStudio-Lizenzierungsverfahren wurde mit der Einführung von WinStudio 7.x (ab IndraWorks 09VRS) geändert. Im Folgenden finden Sie die Beschreibung der neuen Lizenzverwaltung.

Übersicht der WinStudio-Lizenzen

Folgende Lizenzen stehen zur Verfügung:

- **WinStudio Lite**
Diese Lizenz ist Bestandteil der IndraWorks-Installation und ermöglicht die Erstellung und Ausführung von WinStudio-Projekten mit maximal 500 Variablen (Tags). Darüber hinaus sind alle CE-Geräte mit einer Lite-Lizenz zur Ausführung von Lite-Runtimeprojekten ausgestattet. Bei dieser Lizenz werden Alarmer, Rezepte, Reporte, Trends sowie TCP/IP- und DDE-Treiberblätter nicht ausgeführt und können nicht erstellt oder bearbeitet werden.
- **WinStudio Runtime 1K5 / 4K / 64K / 512K (RUN-Lizenzen)**
Für die Ausführung von WinStudio-Runtimeprojekten stehen Lizenzen mit maximal 1.500, 4.000, 64.000 und 512.000 WinStudio-Variablen (Tags) zur Verfügung.
- **WinStudio Engineering und Runtime 1K5 / 4K / 64K / 512K (RUD-Lizenzen)**

Diese Lizenz ermöglicht die Erstellung und Ausführung von WinStudio-Projekten mit maximal 1.500, 4.000, 64.000 bzw. 512.000 WinStudio-Variablen (Tags)

- **WinStudio Demo-Lizenz**

Diese Lizenz ermöglicht es den vollen WinStudio Funktionsumfang innerhalb eines Zeitraums von 30 Tagen zu testen. Nach Ablauf des Testzeitraums ist die zuletzt gültige Lizenz (z.B. Lite-Lizenz) aktiv

Die Demo-Lizenz kann nur einmal aktiviert werden.



Für die Aktivierung auf einem IndraControl VEx- Gerät stehen ausschließlich Lite, 1K5 und 4K Runtime-Lizenzen zur Verfügung.

Aktivierung einer WinStudio-Lizenz

Die einzelnen "WinStudio Runtime"- und "WinStudio Engineering und Runtime"-Lizenzen sind kostenpflichtig und müssen durch Eingabe eines Lizenzierungsschlüssels aktiviert werden. Ihren persönlichen Lizenzschlüssel finden Sie auf dem Lizenzblatt.

Um eine Lizenz zu aktivieren, öffnen Sie in IndraWorks Engineering über **Extras ▶ Optionen ▶ Allgemein ▶ Softwarelizenzen** den Lizenzierungsdialog und wählen Sie den Eintrag "WinStudio" aus. Mit der Schaltfläche **Lizenzieren...** können Sie den Aktivierungsdialog öffnen und den Aktivierungsschlüssel eingeben.

Nach der Aktivierung der Lizenz ist es ggf. notwendig, WinStudio-Engineering oder IndraWorks Operation neu zu starten.

Lizenz auf Dongle speichern

Der Aktivierungsschlüssel ist immer nur für einen PC gültig. Um eine Lizenz nacheinander auf mehreren PCs nutzen zu können, gibt es die Möglichkeit die Lizenz auf einen Dongle zu übertragen. Die Lizenz ist dann solange gültig wie der Dongle an dem USB-Anschluss des Computers eingesteckt ist. Windows CE-Lizenzen (VEx) können nicht auf einen Dongle übertragen werden.

Lizenzaktivierung auf VEx-Geräten

Um die WinStudio-Runtime-Lizenz auf einem Visualisierungsgerät mit dem Betriebssystem Windows XP (embedded)/ Windows 7(embedded) zu aktivieren, öffnen Sie auf dem Gerät im Windows Startmenü **Programme ▶ Rexroth ▶ IndraWorks ▶ Common Tools ▶ License Manager** den Lizenzierungsdialog und geben Sie den Aktivierungsschlüssel ein. Auf neuen VEx-Geräten ist die Lizenzierungssoftware bereits vorinstalliert. Falls Sie bereits ein VEx-Geräte ohne vorinstallierte Lizenzierungssoftware besitzen, installieren Sie zunächst die IndraWorks-Visualisierungssoftware auf dem VEx-Gerät und aktivieren Sie danach die WinStudio-Lizenz.

Falls Sie ein VEx Gerät mit dem Betriebssystem Windows CE besitzen, aktivieren Sie den Lizenzmanager über die Desktop-Verknüpfung **Licenselmpor.CE**

Verwenden Sie ein VRxx-Gerät, so benutzen sie die Remote-Lizenzierung in der IndraWorks Engineering Umgebung. Sie erreichen diese über das Kontextmenü des jeweiligen Geräteknotens **Lizenzieren**. Beachten Sie hierbei, dass Sie mit dem Gerät verbunden sein müssen.



Die Lizenzaktivierung über den WinStudio-Remote-Agent ist mit der Einführung des neuen Lizenzierungs-Verfahrens in WinStudio 7.x nicht mehr möglich. Verwenden Sie zur Lizenzaktivierung den IndraWorks-Lizenzierungsdialog (wie oben beschrieben).

Visualisierungen



Eine Übertragung des Lizenzschlüssels über die integrierte WinStudio-Funktion ist mit der Einführung des neuen Lizenzierungs-Verfahrens in WinStudio 7.x nicht mehr möglich. Der Lizenzschlüssel muss direkt am VEx-Gerät eingegeben werden.

Auswirkungen einer fehlenden oder zu geringen Lizenz

Wenn Sie ein Projekt mit dem WinStudio-Editor öffnen, das auf einem anderen Computer mit einer höheren WinStudio-Lizenz erstellt wurde, sind die lizenzpflichtigen Elemente in WinStudio-Engineering deaktiviert, d.h. eine Erstellung oder Bearbeitung dieser Elemente ist in diesem Fall nicht möglich.

Beim Start von IndraWorks Operation erscheint eine Hinweismeldung, falls die vorhandene WinStudio-Lizenz für das aktuelle Projekt nicht ausreicht. Bei unzureichender oder fehlender Lizenz werden keine WinStudio-Bilder angezeigt.



Falls Sie das aktuelle Projekt dennoch nutzen möchten, können Sie die Demo-Lizenz aktivieren. Verwenden Sie zur Lizenzaktivierung den IndraWorks-Lizenzierungsdialog (wie oben beschrieben).

3.4.16 Mehrsprachigkeit

WinStudio-Anwenderbilder können mehrsprachig projiziert werden, so dass alle Texte in WinStudio-Bildern zur Laufzeit in der jeweils eingestellten Projektsprache angezeigt werden.

Dazu ist es erforderlich, über den Knoten **WinStudio** ► **Tasks** ► **Lokalisierung** eine Übersetzungsdatei zu erstellen. Hierbei wird, ausgehend von einer Quellsprache, für jede Zielsprache (die Sprache, in welche die Projektttexte übersetzt werden sollen) eine neue Spalte erzeugt (Mehrsprachigkeitsfunktion).

Weitere Informationen über die Mehrsprachigkeitsfunktion von WinStudio finden Sie in der WinStudio Online-Hilfe "Rexroth WinStudio 7.3 Technische Referenz".

3.4.17 WinStudio-Applikationen

Übersicht

In IndraWorks können Sie alternativ auch Applikationen erstellen, die ausschließlich auf der WinStudio-Laufzeitumgebung basieren. Dies ist beispielsweise notwendig, wenn Sie die Web-Funktionalität von WinStudio nutzen möchten. Die Erstellung und Aktivierung von WinStudio-Applikationen unterscheidet sich in einigen Punkten von der zuvor beschriebenen Verwendung innerhalb der IndraWorks-Laufzeitumgebung. Diese Unterschiede werden im folgenden aufgezeigt.

Erstellung einer WinStudio-Applikation

Um eine WinStudio-Applikation zu erstellen, fügen Sie ein VPP- oder VEx-Gerät aus der Gerätebibliothek in ein IndraWorks-Projekt ein. Wählen Sie im HMI-Wizard den Applikationstyp "WinStudio-Applikation" aus. Nach Bestätigung der Eingaben mit der Schaltfläche **Fertigstellen** werden die zugehörigen WinStudio-Elemente im Projektbaum angelegt. Elemente wie F-, M- und OP-Panels, Bedienbilder, etc., die die IndraWorks-Laufzeitumgebung voraussetzen, werden hier nicht angeboten.

Benutzerverwaltung

In WinStudio-Applikationen können Sie die WinStudio-Benutzerverwaltung verwenden, um verschiedene WinStudio-Funktionen zu schützen. Die Konfiguration der WinStudio-Benutzerverwaltung erfolgt über die Funktion **WinStudio ► Werkzeuge ► Sicherheitseinstellungen**. In dem Konfigurationsdialog können Sie das Sicherheitssystem aktivieren sowie Gruppen und Benutzer erstellen und die entsprechenden Rechte zuweisen.

Weitere Informationen zur Verwendung der WinStudio-Benutzerverwaltung entnehmen Sie bitte der WinStudio-Online-Hilfe "Rexroth WinStudio 7.3 Technische Referenz"



Benutzer und Gruppen, die ggf. in der IndraWorks-Benutzerverwaltung projektiert wurden, sind in der WinStudio-Benutzerverwaltung nicht verfügbar. Bei Bedarf müssen diese in der WinStudio-Benutzerverwaltung erneut erstellt werden.

Web-Funktionalität

WinStudio bietet die Möglichkeit, zur Laufzeit WinStudio-Applikationen mit Hilfe eines Web-Browsers zu visualisieren (Web-Anwendung). Voraussetzung ist hierbei, dass die Applikation im HTML-Format gespeichert wurde.

Um ein WinStudio-Bild im HTML-Format zu speichern, öffnen Sie das zugehörige Kontextmenü des WinStudio-Bildes und führen Sie die Funktion "Als HTML speichern" aus. Alternativ können Sie die Funktion "Alle Dokumente als HTML speichern" verwenden, um alle bereits vorhandenen WinStudio-Bilder im HTML-Format zu speichern.

Nach Abschluss der Projektierung und Speicherung aller benötigten WinStudio-Bilder im HTML-Format muss der Web-Server konfiguriert werden. Öffnen Sie dazu die WinStudio-Projekteinstellungen und wählen das Register "Web" aus.

Informationen zu Konfiguration und Verwendung des WinStudio-Web-Servers finden Sie in der WinStudio-Online-Hilfe "Rexroth WinStudio 7.3 Technische Referenz".



Die Funktionen "Als HTML speichern" und "Alle Dokumente als HTML speichern" sowie das Register "Web" in den WinStudio-Projekteinstellungen sind nur bei einer WinStudio-Lizenz "1,5 k" oder höher verfügbar.

Übertragen der Applikation und Aktivierung auf dem Zielgerät

Um eine WinStudio-Applikation auf ein Zielgerät zu übertragen, starten Sie auf dem Zielgerät zunächst den WinStudio Remote-Agent.

Bei Geräten mit Betriebssystem Windows CE / Windows 7 embedded compact befindet sich die Desktop-Verknüpfung "WinStudio"CE-Server" auf dem Desktop, über die der Remote Agent gestartet werden kann. Bei Geräten mit Windows 7 embedded / Windows XP embedded starten Sie den Remote Agent über **Start ► Programme ► Rexroth ► Indraworks ► WinStudio ► Remote Agent**

Visualisierungen

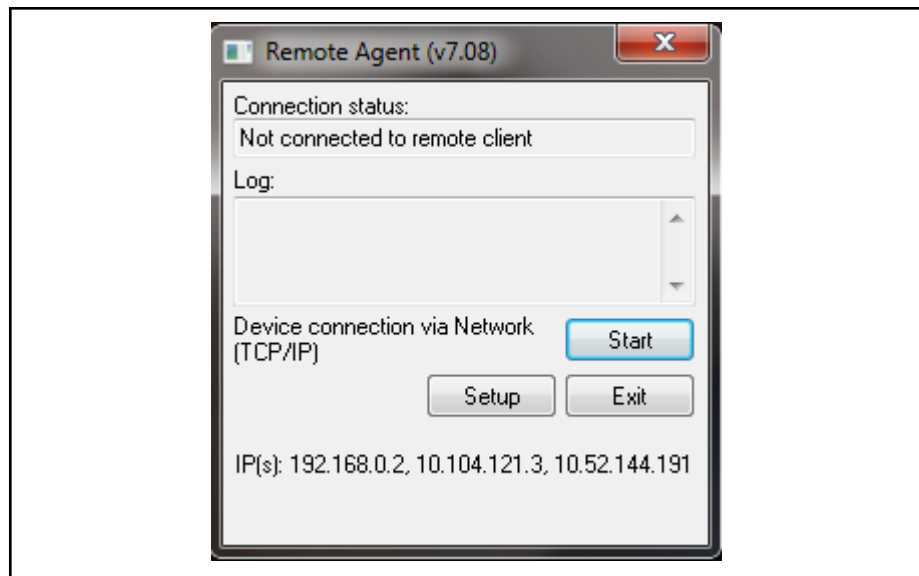


Abb. 3-36: WinStudio Remote Agent auf dem Zielgerät

Anschließend können Sie über das Kontextmenü des WinStudio-Knotens in IndraWorks **WinStudio Zielplattform** den Dialog Verwaltung Remote öffnen. In diesem Dialog können Sie im Reiter "Ziel" auswählen, mit welchem Zielgerät Sie sich verbinden möchten. Bei Einstellung der Zielstation "Localhost" können Sie sich mit dem Remote Agent, der auf Ihrem lokalen Rechner gestartet wurde, verbinden. Um sich mit dem Remote Agent, der auf einem Gerät in Ihrem Netzwerk läuft zu verbinden, tragen Sie die entsprechende IP-Adresse des Gerätes ein. Anschließend stellen Sie mit der Schaltfläche **Verbinden** eine Verbindung mit dem Zielgerät her.

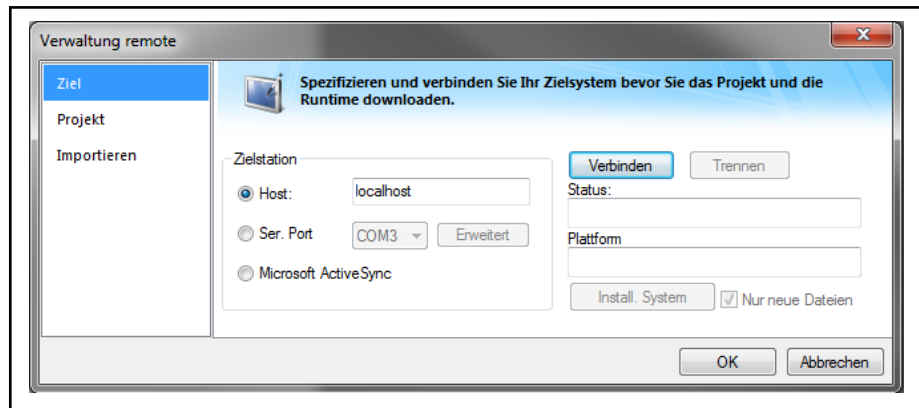


Abb. 3-37: WinStudio Remote-Verwaltung

Wählen Sie im Register "Projekt" den Zielpfad, in den die Projektdateien auf das Zielgerät übertragen werden sollen.



Achten Sie darauf, dass Sie vor Anwahl des Registers mit dem Gerät verbunden sind, dann wird Ihnen der Zielpfad vom System angeboten.

Beachten Sie, dass beim Übertragen von Projektdateien das Zielverzeichnis zuvor gelöscht wird. Dies können Sie unterbinden, wenn Sie die Option "behalten Benutzerdateien" auswählen.

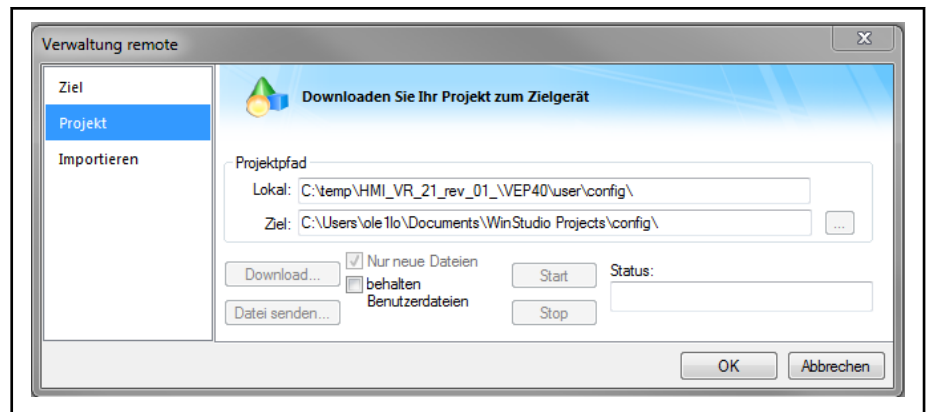


Abb. 3-38: WS Remote-Verwaltung: Register "Projekt"

Mit der Schaltfläche **Download** starten Sie die Übertragung der Projektdateien. Sie können das Projekt nach erfolgter Übertragung mit den Schaltflächen **Start** und **Stop** Starten und Stoppen

3.4.18 Weiterführende Informationen

WinStudio bietet umfangreiche Funktionen zur Erstellung von Anwenderbildern. Der volle Funktionsumfang ist in der Dokumentation WinStudio 7.3 Technische Referenz in der IndraWorks-Online-Hilfe beschrieben.

4 Kommunikationsschnittstellen

4.1 Übersicht

Die folgende Tabelle gibt einen Überblick über die Kombinationsmöglichkeiten von Visualisierungsgeräten, Visualisierungen und Kommunikationsprotokollen.

Gerät und Visualisierung	Kommunikationsprotokoll					
	OPC.DA	OPC.UA	OPC.lwSCP	BRWS	ModbusTCP	IndraLogic Gateway
VCP/VCH						
VI-Composer				x		
VEP/VEH (WinXPe/Win7)						
WinStudio	X	X	X*	X		
Fremd-Visu	X*	X**	X*		X	
VEP/VEH (WinCE)						
WinStudio	X	X		X		
Fremd-Visu					X	
VPx						
WinStudio	X*	X	X*	X		
VI-Logic						X
Fremd-Visu	X*	X**	X*		X	
VR21						
WinStudio		X		X		

* nach Installation der Installationsvariante "Communication - IndraLogic, IndraMotion MLC" durch den IndraWorks-Installer verfügbar
 ** sofern die Visualisierungssoftware über einen OPC UA Client verfügt

Tab. 4-1: Übersicht der Kombinationsmöglichkeiten der Visualisierungen
 Die Steuerungen der Familie XLC/MLC lassen hierbei jeweils die folgende Anzahl von Kommunikationskanälen zu:

- XLC/MLC L25: 4
- XLC/MLC L45: 8
- XLC/MLC L65: 16

4.2 BRWS-Treiber (Bosch-Rexroth-WinStudio Treiber)

4.2.1 Eigenschaften

Der BRWS Treiber ist ab WinStudio Version 6.2 integriert und bietet für PC-basierte und Embedded-PC-basierte Visualisierungsgeräte die Möglichkeit, auf Variablen und Parameter der Systeme XLC/MLC auf Ethernet-Basis zuzugreifen.

Hierzu nutzt der Treiber die Möglichkeiten des Kommunikationsservers (KOM-Server).

Kommunikationsschnittstellen

Das BRWS Protokoll arbeitet ausschließlich mit synchronen Datendiensten. Der Client stellt eine Anfrage und der Server antwortet unmittelbar auf diese.

4.2.2 Anwendungsgebiete

4.2.3 Konfiguration

Die Konfiguration des BRWS Treibers erfolgt in der WinStudio Entwicklungs-umgebung.

Im IndraWorks-Projektexplorer das Kontextmenü des WinStudio- Kommunikationsknotens öffnen und **Treiber einfügen** auswählen. In dem sich öffnenden Kommunikationstreiber-Dialog kann nun der Treiber (BRWS Verbindung mit Kommunikationsserver) ausgewählt werden und erscheint dann im unteren Fenster des Dialoges.

Nach Bestätigung mit **OK** wird im Arbeitsbereich von WinStudio ein neuer Ordner "BRWS" eingerichtet. Dort kann der Treiber konfiguriert, und neue Treiberblätter angelegt werden.

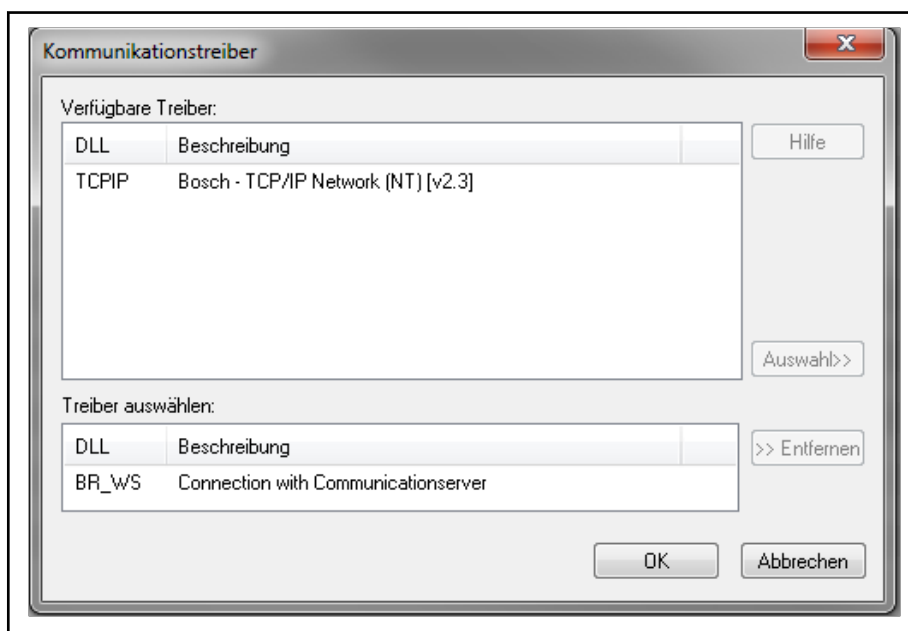


Abb. 4-1: Treiberauswahl

Im Treiber-Einstellungsdialog wird die TCP/IP-Adresse der Zielsteuerung angegeben. Als "Serial Abkapselung" ist "Nichts" einzustellen.

In das Feld "Timeout" kann eine Zeitüberschreitung in Millisekunden angegeben werden. Die eingestellte Zeit wird zur Zeitüberwachung der Kommunikationsverbindungen verwendet. Der Timeout-Wert ist ausschlaggebend, wenn eine Visualisierung mit zwei oder mehreren Steuerungen verbunden ist und eine Steuerung ausfällt oder nicht erreichbar ist.

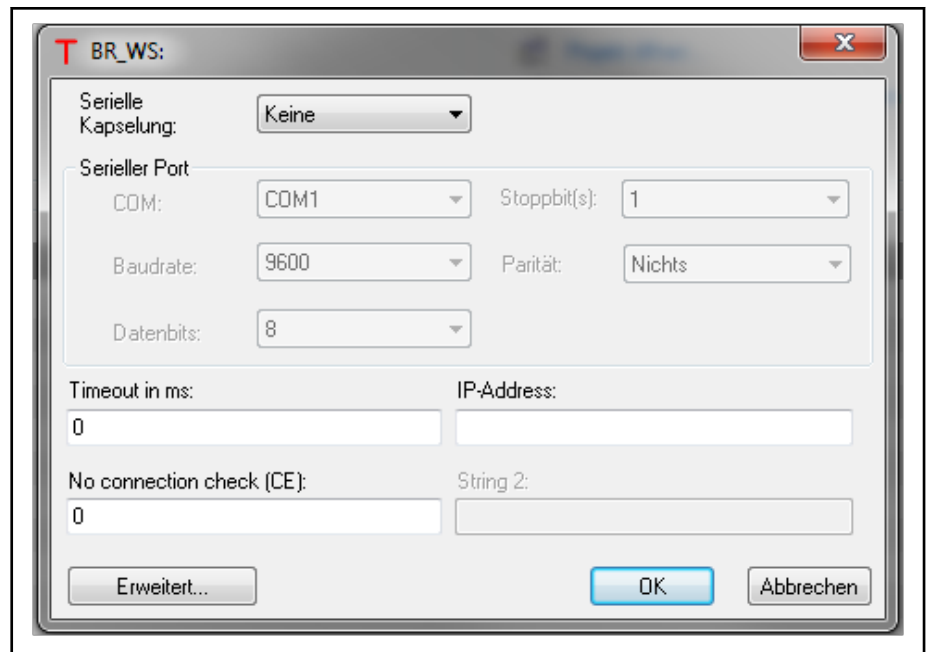


Abb. 4-2: Treibereigenschaften



Bei schlechten LAN-Verbindungen kann es bei XP/Win7-Geräten eventuell zu kurzzeitigen Netzausfällen kommen, die mit der Fehlernummer 1005 protokolliert werden. In diesem Fall muss die Timeout-Zeit auf 300 - 500 Millisekunden angehoben werden!

Kommunizieren mehrere Embedded-PC-basierten (CE6.0) Visualisierungsgeräte mit einer Steuerung, und es kommt zu kurzzeitigen Netzausfällen (1005), sollte die Verbindungsprüfung ausgeschaltet werden, um den Kommunikationskanal zu entlasten.

Empfehlung: Werden Projekte mit großen Symboltabellen, z.B. während der Projektierungsphase mittels "Online Change" in die Steuerung geladen, empfehlen wir eine Timeout-Zeit von mindestens 1000 bis 2000 Millisekunden!

4.2.4 Treiberblätter

In den WinStudio Treiberblättern werden im wesentlichen die Daten festgelegt, welche zwischen Steuerung und HMI kommuniziert werden sollen. Darüber hinaus kann definiert werden, wie die Daten kommuniziert werden sollen.

Durch einen rechten Mausklick auf den BRWS Ordner können neue Treiberblätter erstellt werden. Diese werden als Elemente unter dem Knoten BRWS angezeigt und können dort geöffnet und bearbeitet werden.

Die Treiberblätter unterteilen sich in zwei Bereiche:

- **Kopfbereich:**

Im Kopfbereich wird festgelegt, zu welchen Zeiten und mit welcher Priorität gelesen oder geschrieben werden soll.

- **Listenbereich:**

Im Listenbereich wird die Beziehung der Variablen in der Anwendung zu den Variablen in der Steuerung definiert.

Kommunikationsschnittstellen

Abb. 4-3: BRWS Treiberblatt

Kopfbereich der Treiberblätter:

- **Beschreibung**
Beschreibung des Treiberblattes zu Dokumentationszwecken.
- **Checkbox: Erhöht die Priorität**
Das aktivieren der Checkbox erhöht die Priorität für das Lesen und Schreiben der Variablen dieses Treiberblattes.
- **Lese Trigger**
Variablen des Treiberblattes werden bei Änderung des eingetragenen Tags gelesen.
- **Lesen wenn Untätig**
Wird in dieses Feld eine Konstante oder ein Tag ungleich Null eingegeben werden die Variablen des Treiberblattes kontinuierlich gelesen.
- **Lesen beendet**
Eingetragener Tag wird gesetzt wenn der Lesebefehl abgeschlossen ist.
- **Lese Status**
Der Status des letzten Lesebefehls wird an den eingetragenen Tag übergeben.
- **Schreibe Trigger**
Variablen des Treiberblattes werden bei Änderung des eingetragenen Tags geschrieben.
- **Schreiben bei Änderung**
Wird in dieses Feld eine Konstante oder ein Tag ungleich Null eingegeben, werden die Variablen ständig auf Änderungen überprüft. Tritt eine Änderung ein werden die Variablen des Treiberblattes geschrieben.
- **Schreiben beendet**
Ein eingetragener Tag wird gesetzt wenn der Schreibbefehl abgeschlossen ist.
- **Schreibe Status**
Der Status des letzten Schreibbefehls wird an den eingetragenen Tag übergeben.
- **Station**
Das Feld erweitert die Möglichkeiten des Treibers in Bezug auf die Kommunikation zu anderen Steuerungen.

Kommunikationsschnittstellen

Bleibt es leer, wird die IP-Adresse der vorher bei der Treiberkonfiguration (siehe [Treibereigenschaften](#), Seite 85) eingestellten Zielsteuerung verwendet, um Daten zu lesen oder zu schreiben.

Wird in die Station die IP-Adresse einer Steuerung eingegeben, wird mit dieser Steuerung kommuniziert. Die IP-Adresse kann zur Laufzeit beliebig geändert werden. Das bedeutet, dass immer die aktuell eingetragene IP-Adresse angesprochen wird. Die angesprochenen Steuerungen müssen die verwendeten Symbole geladen haben und auch physikalisch über TCP/IP ansprechbar sein, sonst ist keine Kommunikation möglich.

Damit ist es möglich, die verwendete Visualisierung auf verschiedenen Steuerungen zu verwenden, ohne das Projekt anzupassen.

- **Kopfzeile**

Es ist notwendig eine eindeutige Identifikation (ID) in die Kopfzeile einzugeben.

Dabei spielt es keine Rolle, wie die ID aufgebaut ist. Es können Zahlen und Buchstaben verwendet werden. Sie muss nur eindeutig sein, d. h., innerhalb eines Projektes darf eine ID nur einmal vorkommen. Wird die Nummer des Treiberblattes mit in der ID verwendet, sind keine doppelten IDs möglich.



Vermeiden Sie – sofern möglich – ein gleichzeitiges "Lesen wenn untätig" und "Schreiben bei Änderung"! Das Lesen von Variablen sollte über den "Lese Trigger" erfolgen.

Listenbereich der Treiberblätter:

- **Variablenname**

Hier werden die WinStudio Variablen eingetragen, die mit dem Wert der Variablen aus der Spalte Adresse beschrieben werden sollen (lesend), oder deren Wert auf die Variable in der Spalte Adresse geschrieben werden soll (schreibend).

- **Adresse**

In diesem Feld wird der Name und der Typ der Variable der Steuerung angegeben.

Der Variablenname besteht aus <Programmbausteinname>.<Variablenname>;<Typ>

– **Beispiel:** "Globale Variable"

.Glob_int;T_INT16

.Glob_write;T_UTF8STR

– **Beispiel:** "Lokale Variable im Programmbaustein PLC_PRG"

PLC_PRG.Byt;T_INT8

PLC_PRG.Ind;T_INT16

4.2.5 Protokoll-Aufbau

Jedem Protokoll wird die Anfragenlänge im UTF-8 Format vorangestellt. Die Anfragenlänge ist nicht identisch mit der in der Datensicherungsschicht (Transport/Network), wenn die Anfrage beim Transport fragmentiert wird und/oder mehrere Anfragen gleichzeitig übertragen werden.

Header	Body
UTF-8 Datenlänge 0-7FFF FFFF (1-6 Byte)	Anfragen / Antworten

Kommunikationsschnittstellen

4.2.6 Konventionen

Der BRWS Treiber kann maximal mit 20 unterschiedlichen Steuerungen (IP-Adressen) gleichzeitig kommunizieren.

Allgemeine Treiberblatt-Informationen:

- maximal 512 Items pro Treiberblatt
- es dürfen keine Leerzeilen im Treiberblatt vorhanden sein
- alle Variablen müssen in der Steuerung vorhanden sein
- Groß-, bzw. Kleinschreibung im Adressfeld muss identisch zur Schreibweise in der Symboldatei sein
- maximal 960 Zeichen je Adressfeld

Informationen zum EIS:

- Beim EIS wird die Anzahl der für die Kommunikation zur Verfügung stehenden Handles auf 10000 beschränkt (DeviceXCC.xml bzw. DevicePlc.cfg).
- Die Anzahl der benötigten Handles setzt sich wie folgt zusammen:
Jede angemeldete SPS Variable benötigt ein Handle. Zusätzlich werden für jedes Treiberblatt ggf. mehrere Gruppenhandles angemeldet.
- Die TCP Windowsize im MLC BSP ist auf 1460 Byte (1 Telegramm) gesetzt. D.h. ist die Datenmenge eines Treiberblattes größer als 1460 Byte muss es aufgeteilt werden, was zu Einbußen in der Performance führt.

4.2.7 Unterstützung SPS-Variablen und Parameter

SPS-Variablen

- Item Syntax
 - 'POE_NAMESPACE'.PlcVariable';BRWS_DataType'

Bsp.: Application.PLCPROG.Value;T_INT16

Hier wird die Variable "Value" aus der POU "PLCPROG" des Namensraumes "Application" adressiert. Diese ist vom Datentyp INT

SPS Datentyp	BRWS Datentypen	WinStudio
BOOL	T_BOOL	Boolean
BYTE	T_UINT8	Integer
WORD	T_UINT16	Integer
DWORD	T_UINT32	Integer
LWORD	T_UINT64	'nicht unterstützt' Variable mit BRWS Datentyp "T_UTF8STR" konfigurieren.
SINT	T_INT8	Integer
INT	T_INT16	Integer
TIME	T_UINT32	Integer
LTIME	T_UTF8STR	String
DATE	T_UINT32	Integer
DATE_AND_TIME	T_UINT32	Integer
TIME_OF_DAY	T_UINT32	Integer

Kommunikationsschnittstellen

DINT	T_INT32	Integer
LINT	T_INT64	'nicht unterstützt' Variable mit BRWS Datentyp "T_UTF8STR" konfigurieren.
USINT	T_UINT8	Integer
UINT	T_UINT16	Integer
UDINT	T_UINT32	Integer
ULINT	T_UINT64	'nicht unterstützt' Variable mit BRWS Datentyp "T_UTF8STR" konfigurieren.
REAL	T_FLOAT32	Real
LREAL	T_FLOAT64	Real
STRING	T_UTF8STR	String
WSTRING	T_UTF8STR	String
TIME	T_UINT32	Integer
LTIME	T_UTF8STR	String
DATE	T_UINT32	Integer
DATE_AND_TIME	T_UINT32	Integer
TIME_OF_DAY	T_UINT32	Integer

Tab. 4-2: Zuordnung SPS Datentypen



Die Konvertierung in einem Datentyp mit größerem Wertebereich ist möglich.

Z.B. "T_UINT8 ⇒ T_UINT16 ⇒ T_INT32 ⇒ T_UTF8STR"

XLC/MLC Parameter

- Item Syntax

'CtrlNo','AxisNo','SercosParameter'. 'SercosElement','BRWS_DataType'

Bsp.: 0,15,A-0-0002.7;T_UTF8STR

Hier wird mittels des Achsparameters A-0-0002 der Name der Achse mit der Achsnummer 15 (A-0-0001) adressiert. Die Achse wurde in der Steuerung mit der Steuerungsnummer 0 (C-0-0031) angelegt. Es wird hierbei auf das Element 7 (Data) des Parameters A-0-0002 zugegriffen. Dieses Element ist eine STRING-Variable.

BRWS Item Element	Wert	Beschreibung	
CtrlNo	0	Lokale Steuerung	optional
	1 - 999	Steuerungsadresse im Verbund	
AxisNo	0	Steuerungsparameter	
	1 - 999	Achsparameter	

Kommunikationsschnittstellen

BRWS Item Element	Wert	Beschreibung	
SercosParameter	C	Systemparameter	
	K	Kinematic	
	M	Touch Probe	
	N	Cam Switch	
	O	Oscilloscope	
	A	Axis specific parameter	
	S	Standard drive parameter	
	P	Product drive parameter	
SercosElement	'Empty'	Data	
	1	sercos ID	
	2	Description	
	3	Attribute	
	4	Unit	
	5	Min Value	
	6	Max Value	
	7	Data	
	8	Data status (only Drives)	

Tab. 4-3: Aufbau BRWS Item für Parameter

Parameter Typ	SPS Datentyp	BRWS Datentypen	WinStudio
BINARY, 2BYTE	WORD	T_UINT16	Integer
UINT_DEC, 2BYTE	UINT	T_UINT16	Integer
INT_DEC, 2BYTE	INT	T_INT16	Integer
UINT_HEX, 2BYTE		T_UINT16	Integer
IDN, 2BYTE		T_UINT16	Integer
BINARY, 4BYTE	DWORD	T_UINT32	Integer
UINT_DEC, 4BYTE	UDINT	T_UINT32	Integer
INT_DEC, 4BYTE		T_INT32	Integer
UINT_HEX, 4BYTE	UDINT	T_UINT32	Integer
MLC_IDN, 4BYTE		T_UINT32	Integer
FLOAT, 4BYTE	REAL	T_FLOAT32	Real
TEXT, 1BYTE_VAR	STRING	T_UTF8STR	String
BINARY, 8BYTE	LWORD	T_UINT64	'nicht unterstützt' Variable mit BRWS Datentyp "T_UTF8STR" konfigurieren.
UINT_HEX, 8BYTE	ULINT	T_UINT64	'nicht unterstützt' Variable mit BRWS Datentyp "T_UTF8STR" konfigurieren.

Kommunikationsschnittstellen

DOUBLE, 8BYTE	LREAL	T_FLOAT64	Real
MLC_IDN, 8BYTE	MB_IDN (LINT)	T_UINT64	'nicht unterstützt' Variable mit BRWS Datentyp "T_UTF8STR" konfigurieren.

Tab. 4-4: Zuordnung Parameter Datentypen



Die Konvertierung in einem Datentyp mit größerem Wertebereich ist möglich.

Z.B. "T_UINT8 ⇒ T_UINT16 ⇒ T_INT32 ⇒ T_UTF8STR"

4.2.8 Fehlermeldungen

Mit der Entwicklungsumgebung von WinStudio kann über das Remote LogWin der Zustand der Kommunikation überwacht und / oder in einer Datei protokolliert werden.

Folgende Meldungen werden dabei vom Bosch-Rexroth-WinStudio-Treiber ausgegeben:

-
- 0 ; OK
 - 1003 ; Keine sinnvolle IP-Adresse zur Datenquelle vorhanden
 - 1004 ; Anmeldung am Kom-Server fehlgeschlagen
 - 1005 ; Kommunikationsfehler
(Datenquelle nicht vorhanden, Verbindung zur Steuerung falsch konfiguriert, keine physikalische Verbindung zur Steuerung, Netzwerkprobleme usw.)
 - 1006 ; Zeitüberschreitung der Antwort auf eine Anfrage am Kom-Server (> 30 Sekunden)
 - 1008 ; Ungültiges Handle
(Variable im Treiberblatt nicht in den Symbolen der Steuerung, Symboldatei geladen?, Programm nachgeladen, falsche Symboldatei)

Tab. 4-5: Meldungen des Bosch-Rexroth-WinStudio-Treiber

Weitere Informationen über das Remote LogWin finden Sie in der IndraWorks Online-Hilfe "Rexroth WinStudio 7.3 Technische Referenz".

4.2.9 Erläuterungen der Meldungen

- 0 Keine Fehler. Kommunikation ist online.
- 1003 Bei der Konfiguration des Treibers wird die IP-Adresse der Zielsteuerung eingetragen. Ist dieser Eintrag leer oder falsch, kann sich der Treiber nicht mit dem Kommunikationsserver auf der Zielsteuerung verbinden.
- 1004 Login am Kom-Server fehlgeschlagen. Die grundsätzliche Verbindung mit der Steuerung wurde aufgenommen, auf die Login-Anfragen wird aber negativ oder gar nicht geantwortet. Entweder ist in der Steuerung kein oder ein alter Stand des Kom-Server installiert oder die Steuerung ist in einem undefinierten Zustand und die Kom-Server-Task kommt nicht mehr zur Ausführung.
- 1005 Der Kommunikationsfehler wird immer dann angezeigt, wenn das Senden und Empfangen von Kom-Server-Daten gestört ist.
- 1006 Der Kom-Server reagiert nicht auf Anfragen des Clients. Die Steuerung ist überlastet oder befindet sich in einem Ausnahmezustand.

Kommunikationsschnittstellen

- 1008** Ungültiges Handle. Diese Meldung wird vom Kommunikationsserver gesendet, wenn eine Variable vom Client angefordert wird, die nicht auf der Steuerung vorhanden ist. Im Normalbetrieb sollte dieser Fall nicht vorkommen, da bei korrekt ausgefüllten Treiberblättern (mit der "Check"-Funktion des Browsers überprüft werden kann ob alle Variablen auf den Treiberblättern ein Gegenstück in der Steuerung besitzen.

Wird das Programm nachgeladen, verwirft der Kommunikationsserver alle bisher angemeldeten Handles. Der Client erkennt anhand der Rückmeldung 1008 des Kommunikationsservers, dass sein Handle nicht mehr gültig ist. Die Kommunikation wird vom Client neu angesetzt. Alle Handles werden neu angemeldet. Das bedeutet, beim Nachladen des Programms meldet der Client für einige Sekunden einen 1008 Fehler, bis sich der Client wieder synchronisiert hat.

Sollte wiederholt der Fehler "1008" anstehen, gibt es drei prinzipielle Fehlerursachen:

1. Mindestens ein Treiberblatt wurde nicht korrekt ausgefüllt und enthält Variablen-Bezeichnungen, die in der Symboldatei der Steuerung nicht vorhanden sind.

Lösung: Die Treiberblätter mit der "Check"-Funktion des Browsers überprüfen.

2. Eine falsche oder unvollständige Symboldatei wurde auf die Steuerung geladen.

Lösung: Überprüfen, ob der Download des Projektes mit Symbolinformationen gemacht wurde und ob die korrekte Symboldatei geladen wurde.

3. Der Kom-Server hat nicht genügend Handles.

4.2.10 Dokumentation

Für die ausführliche Beschreibung des BRWS Treibers (Bosch-Rexroth-Win-Studio Treiber) steht folgende Dokumentation zur Verfügung:

DOK-CONTRL-IL2GPRO*Vxx-APyy-DE-P

xx - Version

yy - Ausgabe

4.3 OPC.lwSCP

4.3.1 Eigenschaften

Der OPC.lwSCP ist der Bosch-Rexroth eigene Standard-OPC-Server.

Er wird zur Kommunikationsanbindung von IndraWorks-Projekten in Verbindung mit der Standard-Bedienoberfläche "IndraWorks Operation" auf pc-basierenden Hardwaresystemen eingesetzt (siehe [Visualisierungsgeräte, Seite 17](#)).

Die Verwendung für Winstudio- und "Third Party"-Applikationen ist ebenfalls möglich. Voraussetzung hierfür ist jedoch die Installation des IndraWorks Com-Pakets.

Wird der OPC.lwSCP ohne IndraWorks Operation Desktop eingesetzt, muss der OPC.lwSCP für die Geräte, mit denen er sich verbinden soll konfiguriert werden.

Nur für diesen Anwendungsfall ist die nachfolgende Dokumentation [SCP-OPC Konfigurator, Seite 93](#), relevant.



Detaillierte Informationen zur OPC-Kommunikation, den verwendbaren Items und einer eventuell benötigten Client-Programmierung sind in der Dokumentation DOK-IM*ML*-OPC*COM*Vxx-APxx-DE-P verfügbar

4.3.2 SCP-OPC Konfigurator

Allgemein

Der SCP-OPC Konfigurator dient zur Deklaration der Geräte, die über OPC von SCP bereit gestellt werden. Mit Hilfe des SCP-OPC Konfigurators können Sie Konfigurationsdateien erstellen und als aktive Konfiguration kennzeichnen. Beim Start des OPC Servers wird die aktive Konfiguration ausgewertet und die entsprechenden SCP-Geräte werden erzeugt. Selektierte Geräte sind blau hinterlegt. Änderungen an einer aktiven Konfiguration wirken sich erst beim nächsten Neustart des OPC Servers aus.

Im Hauptfenster der Anwendung zeigt Ihnen der SCP-OPC Konfigurator die Liste von Geräten, die in einer SCP-OPC Konfigurationsdatei enthalten sind.

Es werden folgende Geräteeigenschaften angezeigt:

Spalte	Beschreibung
Gerät	SCP-Gerätename, nicht editierbar
Type	SCP-Gerätetyp, nicht editierbar
OPC Alias	OPC Alias-Name des Geräts, editierbar, optional

Tab. 4-6: Angezeigte Geräteeigenschaften

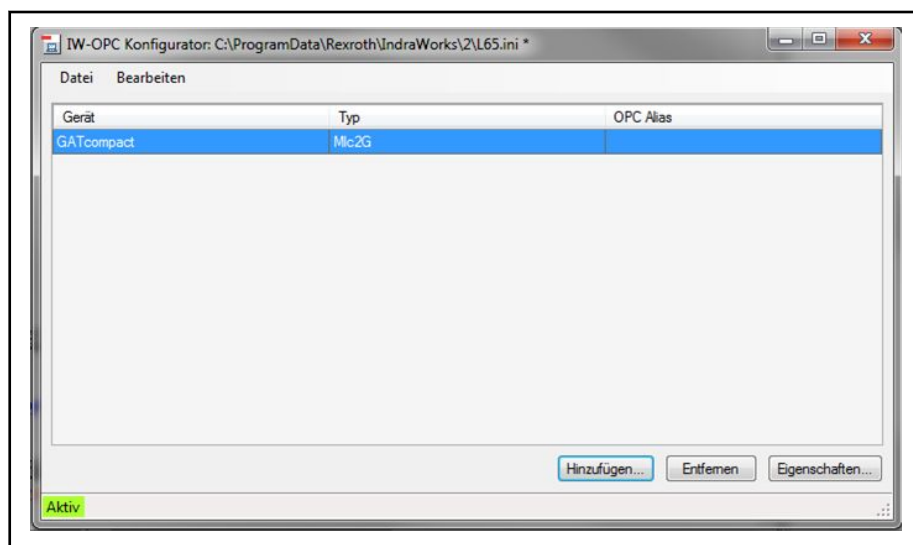


Abb. 4-4: IW-OPC Konfigurator: Geräteeigenschaften

Bedienung

Gerätekonfiguration erstellen

Mit Hilfe der Hauptmenüoption **Datei ► Neu** erstellen Sie eine neue SCP-OPC Konfigurationsdatei. Den Dateinamen geben Sie beim Speichern der Konfiguration an. Liegen beim Beenden der Anwendung noch ungespeicherte Konfigurationsdaten vor, werden Sie zum Speichern der Daten aufgefordert.

Kommunikationsschnittstellen



Die Titelzeile der Anwendung zeigt den Pfad und Dateinamen der gegenwärtig geöffneten Datei an. Wurde eine Datei seit dem Öffnen bzw. seit dem letzten Speichern geändert, wird in der Titelzeile des Dialogs das Symbol "*" am Ende des Dateinamens angezeigt.

Gerätekonfiguration öffnen

Mit Hilfe der Hauptmenüoption **Datei ► Öffnen** öffnen Sie eine neue SCP-OPC Konfigurationsdatei. In einem Dateiauswahldialog wählen Sie das Verzeichnis und die Konfigurationsdatei aus, die Sie öffnen möchten. Es werden ausschließlich Dateien mit der Dateierweiterung ".ini" angezeigt. Die SCP-OPC Konfiguration wird als Liste von Geräten im Hauptfenster der Anwendung angezeigt. Ist die geöffnete Datei die gegenwärtig aktive SCP-OPC Konfiguration, wird in der Statuszeile der Anwendung ein hellgrünes Feld mit der Aufschrift "aktiv" angezeigt.



Die Titelzeile der Anwendung zeigt den Pfad und Dateinamen der gegenwärtig geöffneten Datei an. Wurde eine Datei seit dem Öffnen bzw. seit dem letzten Speichern geändert, wird in der Titelzeile des Dialogs das Symbol "*" am Ende des Dateinamens angezeigt.

Geräte hinzufügen

Mit Hilfe der Hauptmenüoption **Bearbeiten ► Hinzufügen**, der Kontextmenüoption **Hinzufügen** oder der Schaltfläche **Hinzufügen** erhalten Sie nach Angabe der gewünschten Datenquelle einen Auswahldialog, in dem Sie die Geräte auswählen, die Sie Ihrer Konfiguration hinzufügen möchten. Geräte, die in der Konfiguration bereits enthalten sind, werden im Auswahldialog ausgegraut dargestellt. Als Datenquellen stehen Ihnen IndraWorks-Projekte, SCP-OPC Konfigurationsdateien oder eine aktive SCP-OPC Konfiguration zur Verfügung. Sowohl im Auswahldialog als auch im Hauptfenster des Konfigurator können Sie zu jedem Gerät über die Option "Eigenschaften" einen Dialog zur Anzeige der Eigenschaften eines SCP-Geräts aufschalten.



Innerhalb einer Konfiguration müssen die enthaltenen Geräte eindeutig sein. Werden im Auswahldialog Geräte ausgewählt, die bereits in der Konfiguration enthalten sind (grau dargestellt), werden die bereits enthaltenen Geräte nicht mehr eingefügt.

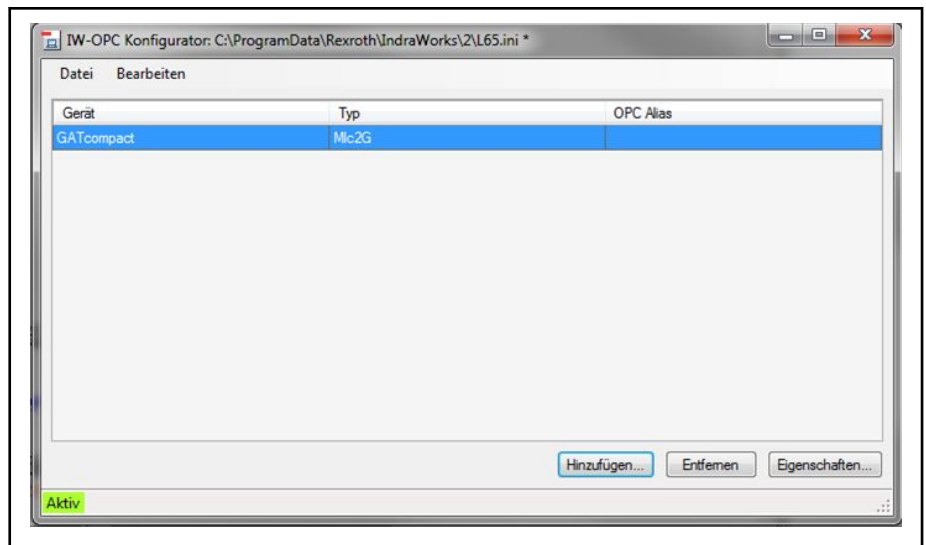


Abb. 4-5: Dialog Geräte auswählen

Geräteeigenschaften anzeigen

Mit Hilfe der Hauptmenüoption **Bearbeiten ► Eigenschaften**, der Kontextmenüoption **Eigenschaften** oder der Schaltfläche **Eigenschaften** wird Ihnen der Eigenschaftsdialog des selektierten Gerätes der Geräteliste, mit allen Eigenschaften, angezeigt. Diese Option ist nur dann aktiviert, wenn genau ein Gerät der Liste selektiert ist.



Den Geräteeigenschaftendialog können Sie sowohl aus dem Hauptfenster als auch aus dem Geräteauswahldialog aufrufen.

Kommunikationsschnittstellen

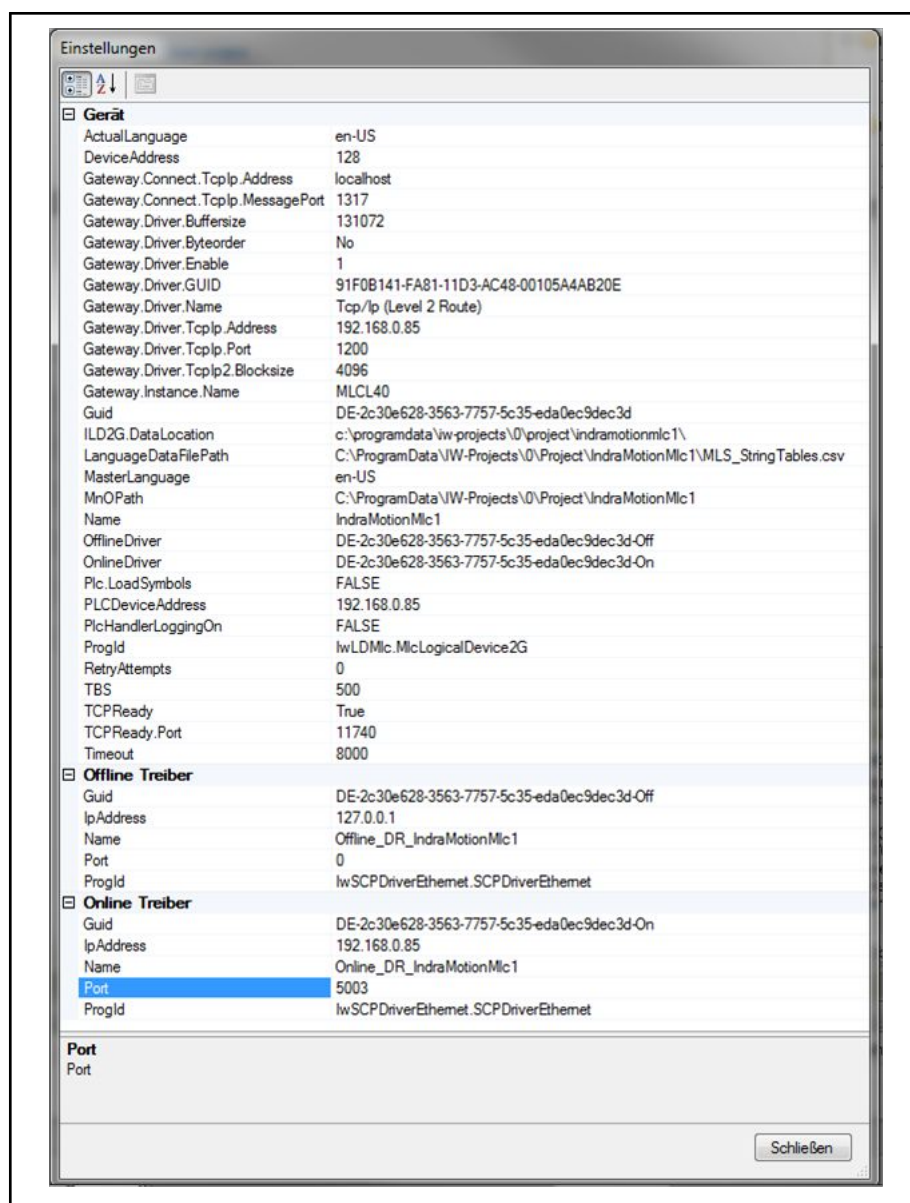


Abb. 4-6: Dialog Einstellungen

Geräte OPC Alias ändern

Der Gerätenamen und der Gerätetyp kann durch den Anwender nicht geändert werden. Diese Felder werden als nicht editierbare Felder mit grauem Hintergrund angezeigt. Sie können jedoch den OPC Alias-Namen eines Geräts ändern, indem Sie das gewünschte Gerät auswählen und mit der Maus das OPC Alias-Namensfeld anklicken. Den Editiermodus beenden Sie durch <Return> oder durch Auswahl eines anderen Felds.



Der OPC Alias-Name darf nur folgende Zeichen enthalten:

"a-z, A-Z, 0-9, _, -".

Bei Verletzung der Namenskonvention erhalten Sie eine Nachricht; der Hintergrund des Felds wird rot dargestellt und innerhalb des Felds wird ein Fehlersymbol angezeigt. Eine Beschreibung zum vorliegenden Fehler finden Sie auch im Tooltip des als fehlerhaft gekennzeichneten Felds.

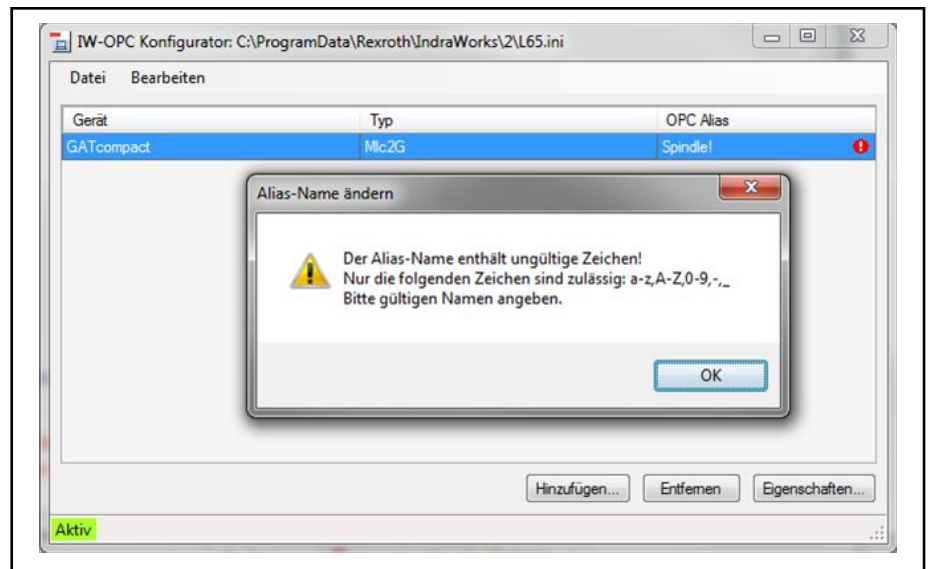


Abb. 4-7: Dialog: Alias-Name ändern

Geräte entfernen

Mit Hilfe der Hauptmenüoption **Bearbeiten ► Entfernen**, der Kontextmenüoption **Entfernen** oder der Schaltfläche **Entfernen** entfernen Sie, nach Bestätigen einer Sicherheitsabfrage, die selektierten Geräte aus der Geräteliste.



Diese Option ist nur dann aktiviert, wenn mindestens ein Gerät der Geräteliste selektiert ist.

Gerätekonfiguration speichern

Mit Hilfe der Hauptmenüoption **Datei ► Speichern** speichern Sie die gegenwärtig bearbeitete Konfiguration unter dem in der Titelzeile der Anwendung angegebenen Verzeichnis und Dateinamen.



Diese Option ist nur dann verfügbar, wenn Sie, seit dem Öffnen bzw. letzten Speichern der Daten, Änderungen vorgenommen haben.

Gerätekonfiguration speichern unter

Mit Hilfe der Hauptmenüoption **Datei ► Speichern unter** speichern Sie die gegenwärtig bearbeitete Konfiguration unter einem anderen Dateinamen. In einem Dateiauswahldialog wählen Sie das gewünschte Verzeichnis aus und geben den Dateinamen an, unter dem Sie die Datei speichern möchten. Existiert die Datei bereits im Zielverzeichnis, müssen Sie das Überschreiben der Datei explizit bestätigen.



Bei neu erstellten Konfigurationen ist die Option "Speichern" noch nicht verfügbar, da noch kein Dateiname vergeben wurde. Die Option "Speichern unter" ist immer verfügbar, falls eine Konfigurationsdatei geöffnet ist.

Gerätekonfiguration aktivieren

Mit Hilfe der Hauptmenüoption **Datei ► Aktivieren** legen Sie fest, dass die gegenwärtig geöffnete Datei als aktive Konfiguration verwendet wird. Das bedeutet, dass diese Konfiguration beim nächsten Neustart des OPC Servers zur Erstellung der Geräteliste des OPC Servers verwendet wird. Es sind aus-

Kommunikationsschnittstellen

schließlich die Geräte über den OPC Server verfügbar, die in dieser Konfigurationsdatei definiert sind.

Ist die geöffnete Datei die gegenwärtig aktive SCP-OPC Konfiguration, wird in der Statuszeile der Anwendung ein hellgrünes Feld mit der Aufschrift "aktiv" angezeigt.



Eine Konfiguration kann nur dann als aktive Konfiguration gekennzeichnet werden, wenn die Konfiguration zuvor als Datei gespeichert worden ist.

Anwendung beenden

Mit Hilfe der Hauptmenüoption **Beenden** oder über das Systemmenü **x** beenden Sie die Anwendung. Liegen bisher noch nicht gespeicherte Daten vor, werden Sie aufgefordert, die Daten zu speichern.

4.3.3 IwSCP und OPC.IwSCP Items

Allgemein

In diesem Kapitel erhalten Sie einen Überblick über die für den OPC Client relevanten Items, die im "IwSCP Server" zur Verfügung stehen.

Bei der Verwendung der Items im OPC Client ist zu beachten, dass die beschriebene Item-Syntax für den "IwSCP Server" um die Angabe des "angeforderten Datentyps" und den "Steuerungsnamen oder Alias" erweitert wird.

Für den direkten Vergleich ist im Folgenden zuerst die Item-Syntax für den "IwSCP" Server und anschließend für den "OPC.IwSCP" Server dargestellt.

IwSCP Server

- SPS Variablen
"[IwSCP Dienst],[voll qualifizierender Variablenname]"
Beispiel:
"Plc.PVL,Application.HMIVARS.bLifeSign_hmi"
- Sercos Parameter
 - MLC/MLP
"[IwSCP Dienst],[Antriebs Adresse],[Parameter Nummer],[Elementkodierung]"
Beispiel:
"SPA1,0,C-0-0001,D"
"SPA1,0,C-0-0001,40"
 - MLD
"[IwSCP Dienst],[Parameter Nummer],[Elementkodierung]"
Beispiel:
"SPA2,S-0-0030,D"
"SPA2,S-0-0030,40"

OPC.IwSCP Server

- SPS Variablen
"! [angeforderter Datentyp],[Steuerungsname oder Alias],[IwSCP Dienst],[voll qualifizierender Variablenname]"
Beispiel:
"!Bool,ImMlc2G,Plc.PVL,Application.HMIVARS.bLifeSign_hmi"
- sercos Parameter

Kommunikationsschnittstellen

– MLC/MLP

"! [angeforderter Datentyp], [Steuerungsname] oder

"Alias], [lwSCP Dienst], [Antriebs Adresse], [Parameter Nummer],
[Elementkodierung]"**Beispiel:**

"!UI4,ImMlc2G,SPA1,0,C-0-0001,D"

"!UI4,ImMlc2G,SPA1,0,C-0-0001,40"

– MLD

"! [angeforderter Datentyp], [Antriebsname oder Alias], [lwSCP
Dienst], [Parameter Nummer], [Elementkodierung]"**Beispiel:**

"!BSTR,Drive01,SPA2,S-0-0030,D"

"!BSTR,Drive01,SPA2,S-0-0030,40"

Für den Parameter "angeforderter Datentyp" können die in der folgenden Tabelle aufgelisteten "OPC.lwSCP Datentypen" verwendet werden.

OPC.lwSCP Datentypen	IEC Datentypen	Wertebereich Speicherplatz
BOOL	BOOL	TRUE (1) und FALSE (0) 8 Bit
I1	SINT	-128 bis 127 8 Bit
I2	INT	-32768 bis 32767 16 Bit
I4	DINT	-2147483648 bis 2147483647 32 Bit
I8	LINT	-2 ⁶³ bis 2 ⁶³ -1 64 Bit
UI1	USINT	0 bis 255 8 Bit
UI2	UINT	0 bis 65535 16 Bit
UI4	UDINT DATE TIME DATE_AND_TIME DT TIME_OF_DAY TOD	0 bis 4294967295 32 Bit
UI8	ULINT LTIME	0 bis 2 ⁶⁴ -1 64 Bit

Kommunikationsschnittstellen

OPC.lwSCP Datentypen	IEC Datentypen	Wertebereich Speicherplatz
R4	REAL	$\pm(1.175494351e-38 \text{ bis } 3.402823466e+38)$ 32 Bit
R8	LREAL	$\pm(2.2250738585072014e-308 \text{ bis } 1.7976931348623158e+308)$ 64 Bit
BSTR	STRING WSTRING	Beliebige Zeichenkette mit einer Länge von 80 Zeichen, wenn die Deklaration in der SPS ohne Längenangabe erfolgt.

Tab. 4-7: OPC.lwSCP Datentypen

IndraLogic Items

Übersicht

In der nachfolgenden Liste sind die für den OPC Client relevanten IndraLogic Items aufgelistet.

Item-Überblick

- **Plc.State**
Auslesen der aktuellen Zustandsinformation der IndraLogic Runtime.
- **Plc.PVT**
Liest die Symbolinformation von dem SPS Projekt.
- **Plc.PVF**
Liest die Symbolinformation einer Variablen.
- **Plc.PVL**
Liest und schreibt Variablenwerte.
- **Plc.PVBS**
Zugriff auf SPS Variablen (Strukturen, Arrays) als Bytestream.
- **Plc.PVLK**
Zugriff auf SPS Variablen (Strukturen, Arrays)

sercos Items

Übersicht

In der nachfolgenden Liste sind die für den OPC Client relevanten sercos Items aufgelistet.

Item-Überblick

- **SPH1**
Liest und schreibt die sercos Phase.
- **SPA1**
Liest und schreibt Elemente der sercos Parameter.
- **SPA2**
Liest und schreibt Elemente der sercos Parameter eines Antriebs.

4.3.4 Haftungsausschluss

WARNUNG

Keine Haftung für Schäden durch Kunden- OPC-Clients

Über OPC haben Sie uneingeschränkten Zugriff auf alle Geräte-Daten und die gesamte Geräte-Funktionalität.

Für Schäden und Datenverlust, die auf eine unsachgemäße Verwendung von OPC in "Third Party" OPC-Client Applikationen und WinStudio-Bedienbilder oder auf eine Umgehung bzw. Nichtbeachtung von gerätespezifischen Sicherheitsrichtlinien zurückzuführen sind, übernimmt Bosch Rexroth keine Haftung. Dies betrifft insbesondere das Schreiben gerätesensibler und sicherheitsrelevanter Daten.

4.4 IndraLogic OPC.DA

4.4.1 Eigenschaften

Der IndraLogic OPC-Server OPC.DA ist ein OPC-Server für den Zugriff auf das SPS-System der Steuerung.

Dabei kann der IndraLogic OPC.DA für PC-basierte und embedded-PC-basierte Visualisierungen eingesetzt werden.

Für den Betrieb des IndraLogic OPC.DA-Server ist immer eine Konfiguration des Server notwendig. Damit wird festgelegt, mit welchen Geräten sich der OPC.DA-Server beim Start verbinden soll (siehe [Konfiguration des IndraLogic OPC.DA Servers, Seite 101](#)).



Detaillierte Informationen zur OPC-Kommunikation, den verwendbaren Items und einer eventuell benötigten Client-Programmierung sind in der Dokumentation DOK-IM*ML*-OPC*COM*Vxx-APxx-DE-P verfügbar

4.4.2 Konfiguration des IndraLogic OPC.DA Servers

Allgemein

Dieses Kapitel beschreibt die Konfiguration des IndraLogic OPC.DA Servers "IndraLogic.OPC.DA" für PC Systeme (Windows XP/Windows 7) und CE Systeme (Windows CE 6) für die Kommunikation mit IndraLogic 2G Steuerungssystemen.

Die Konfiguration des OPC.DA Servers erfolgt über die Datei "OPCServer.ini" im Verzeichnis "%CommonAppData%\Rexroth\IndraWorks":

Beispiel:

Pfade:

"C:\ProgramData\Rexroth\IndraWorks"(Windows7) oder "C:\Documents and Settings\All Users\Application Data\Rexroth\IndraWorks" (WindowsXP) auf PC Systemen bzw. im Verzeichnis "\StorageCard\Rexroth\Controls\OPCServer" auf CE Systemen.

Die "OPCServer.ini" wird auf PC-basierten Bedienfeldern mit einem Konfigurator erstellt. Siehe auch [OPC.DA Server Konfiguration, Seite 102](#).

Die manuelle Konfiguration der "OPCServer.ini" (bspw. für WindowsCE basierte Visualisierungsgeräte notwendig, falls kein Konfigurator im Rahmen ei-

Kommunikationsschnittstellen

ner Installation von Indraworks Communication verfügbar ist) wird unter [OPC.DA Server Konfigurationsdatei OPCServer.ini, Seite 105](#), beschrieben.



Damit eine Kommunikation mit der Steuerung möglich ist, muss eine Symbolkonfiguration auf der Steuerung vorhanden sein. Siehe auch [Symbolkonfiguration, Seite 141](#).



Der IndraLogic OPC.DA Server unterstützt die Kommunikation zu den IndraLogic 2G Steuerungssystemen.

Dieses Kapitel beschreibt nur die relevanten Einträge in der OPC.DA Server Konfigurationsdatei "OPCServer.ini" für die Kommunikation mit IndraLogic 2G Steuerungssystemen.

OPC.DA Server Konfiguration

Kommunikationsadresse ermitteln

Der OPC.DA Server bzw. die unterlagerte Kommunikationskomponente "GatewayService" benötigen für die Kommunikation mit der Steuerung eine "Kommunikationsadresse", die sich unter anderem aus der IP-Adresse und der SubNet-Mask der Steuerung berechnet.

Damit die Kommunikationsadresse als Fehlerquelle in der Konfigurationsdatei ausgeschlossen werden kann, ist diese immer über einen Verbindungstest mit der Steuerung zu ermitteln.

In der Folgenden Abbildung wurde die Kommunikationsadresse "0000.c0a8.0001" für das Steuerungssystem IndraMotion MLC mit der IP-Adresse "192.168.0.1" und der SPS Kommunikationsoption UDP ermittelt. Die Kommunikationsadresse wird nach erfolgreichem Verbindungstest in der Zeile "Adresse:" im Bereich "SPS-Kommunikation erfolgreich:" von dem Feld "Ergebnis Verbindungstest" als hexadezimale Ziffernfolge angezeigt und für die Konfiguration des OPC.DA Servers auch in dieser Form verwendet.

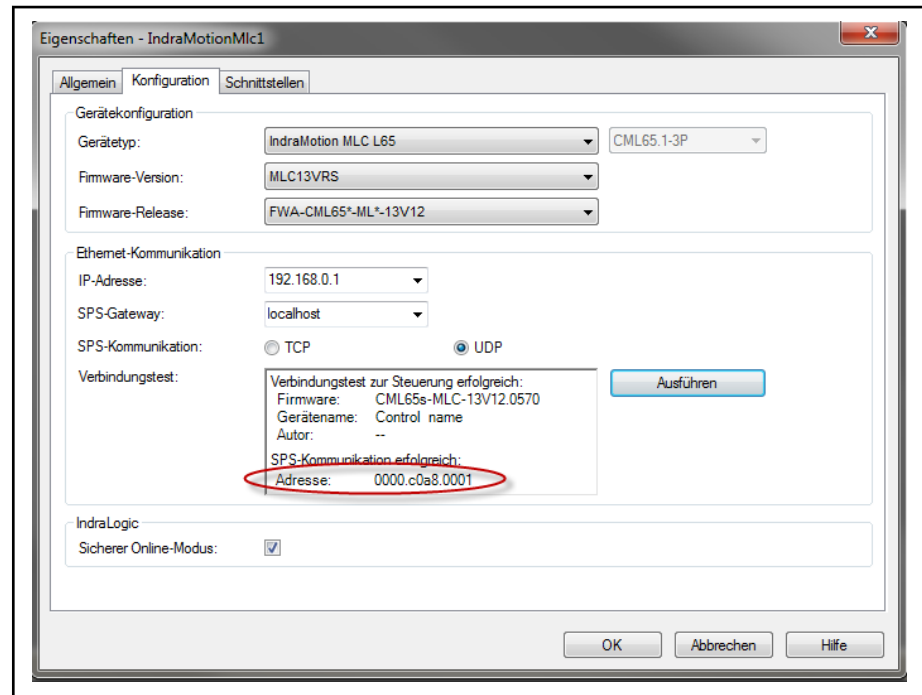


Abb. 4-8: Eigenschaftsdialog IndraMotion MLC - Kommunikationsadresse ermitteln

OPC.DA Server Konfigurator

Für die Konfiguration des OPC.DA Servers steht auf PC-basierten Visualisierungsgeräten ein Konfigurator zur Verfügung

Der Aufruf erfolgt über das Windows Startmenü der installierten IndraWorks-Version

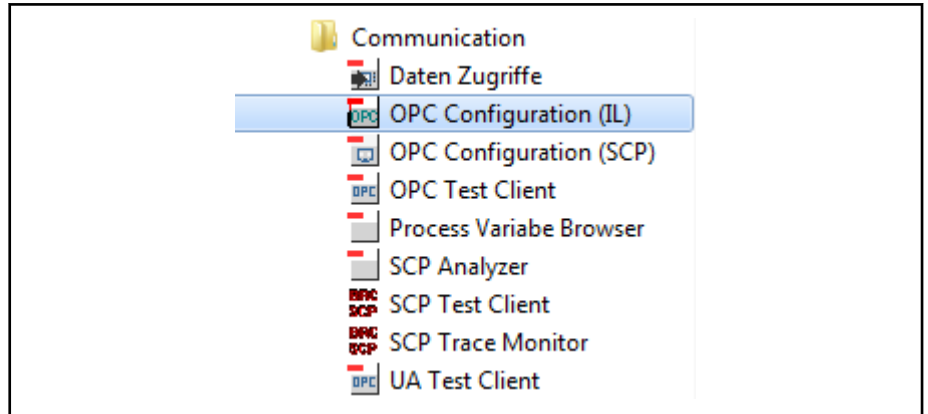


Abb. 4-9: Aufruf OPC.DA Server Konfigurator

Das Anlegen einer neuen Konfiguration erfolgt über Aufruf des Kontextmenüs und Anhängen einer neuen PLC.

- Das Logging sollte für den "normalen Betrieb" deaktiviert werden, da es lediglich während der Inbetriebnahme interessant ist.
- Die Standard Update-Rate von 200ms für die Anmeldung von Gruppen und Items am OPC.DA Server wird empfohlen. Höhere Update-Raten sind in den meisten Anwendungsfällen nicht erforderlich.



Eine detaillierte Beschreibung der Konfigurationsparameter finden Sie im [OPC.DA Server Konfigurationsdatei - Server'-Sektion \[Server\]](#), Seite 106.

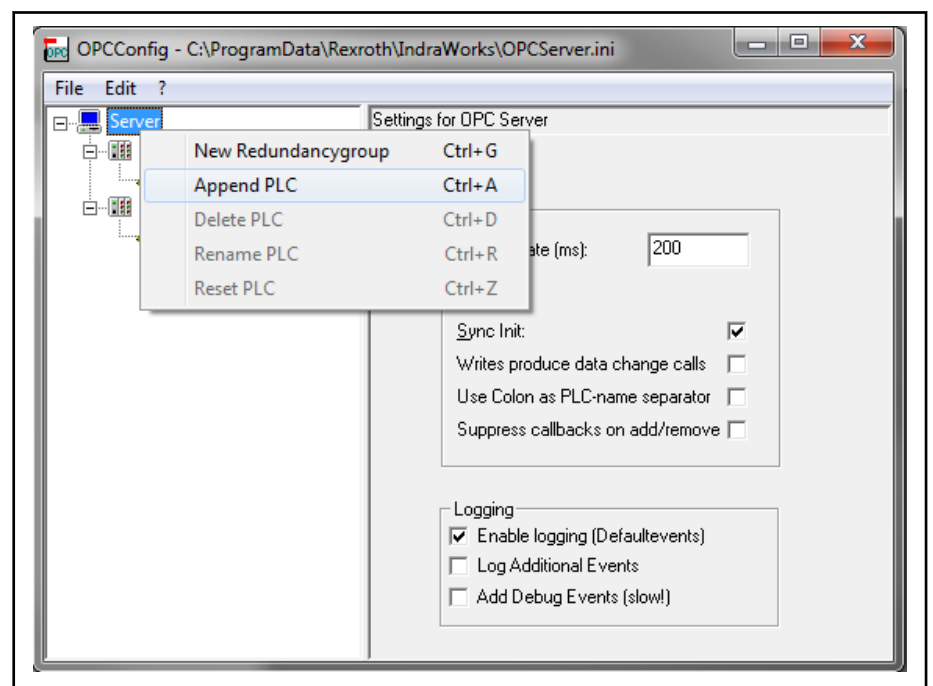


Abb. 4-10: Anlegen einer neuen Steuerung innerhalb der OPC.DA Server Konfiguration

Kommunikationsschnittstellen

Für die Kommunikation des OPC.DA Servers mit 2G Steuerungssystemen muss ein Interface "GATEWAY3" (= Default) ausgewählt werden, über das die Verbindung zur Steuerung aufgebaut werden soll

Abschließend sind noch die Verbindungseinstellungen vorzunehmen, wo die IP-Adresse der 2G Steuerung in HEX einzugeben ist.

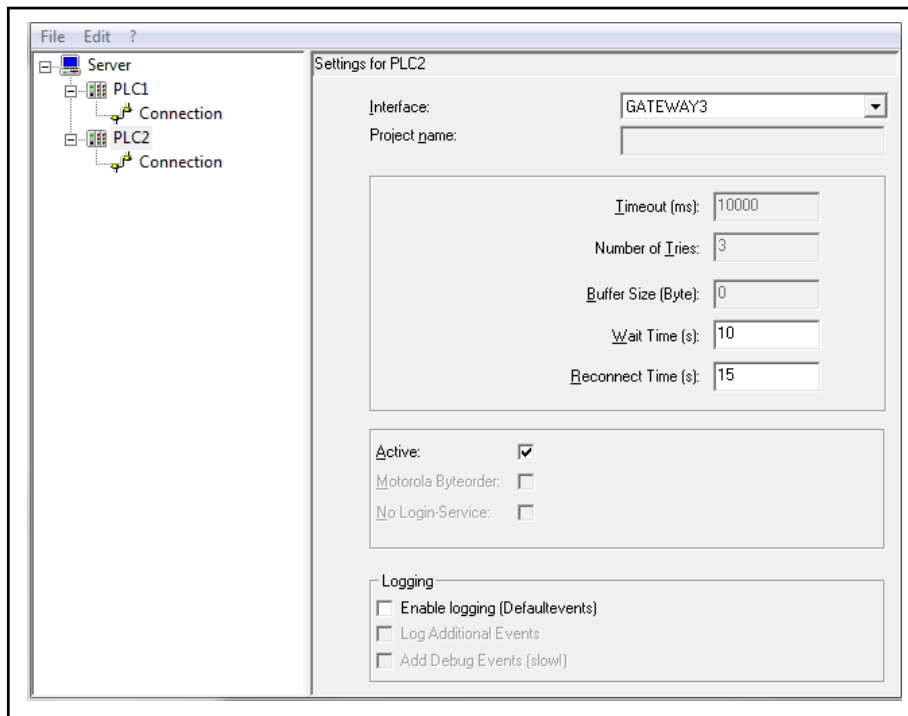


Abb. 4-11: Konfiguration des Interfaces (Gateway) für die Kommunikation im OPC.DA Server Konfigurator

Im beschriebenen Fall wurde die Adresse verwendet, wie sie unter [Kommunikationsadresse ermitteln](#), Seite 102, in IndraWorks Engineering ermittelt wurde.

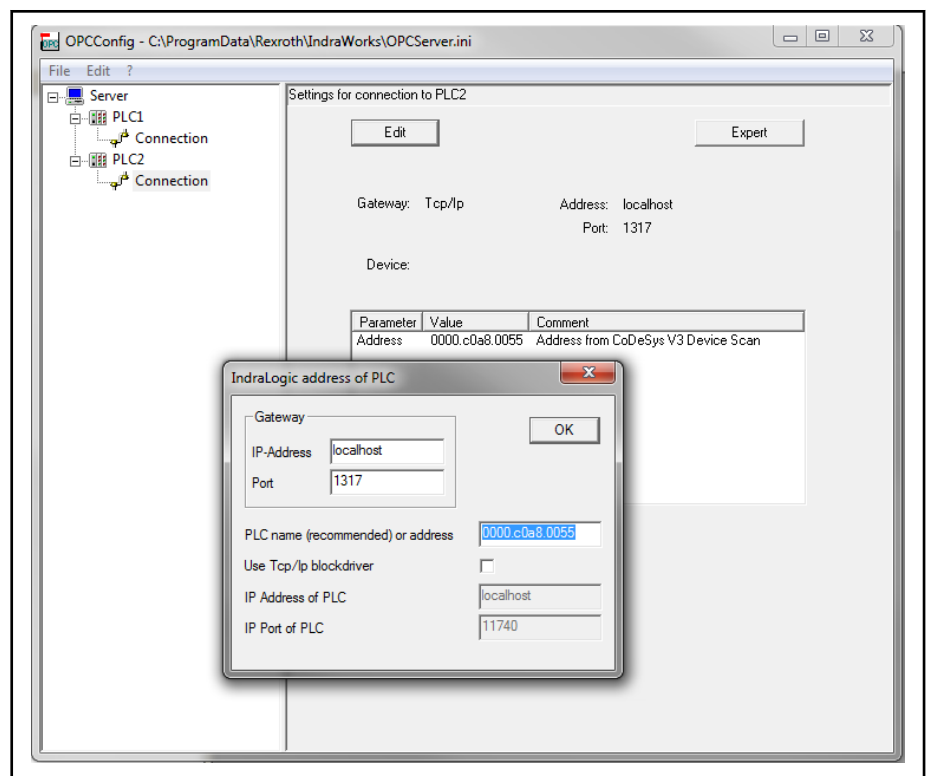


Abb. 4-12: Konfiguration der Verbindungseinstellungen im OPC.DA Server Konfigurator

Zum Abschluss muss die Konfiguration noch unter "OPCServer.ini" abgespeichert werden, damit sie beim nächsten Start des OPC.DA Servers wirksam wird

Zu Ablageorten der Konfiguration siehe [Kap. "Allgemein" auf Seite 101](#)

OPC.DA Server Konfigurationsdatei "OPCServer.ini"

Die OPC.DA Server Konfigurationsdatei "OPCServer.ini" hat den Aufbau einer "INI-Datei", d. h., sie beinhaltet Sektionen und Parameter mit Namen und Werten. Mit dem Zeichen ";" (Semikolon) können Kommentarzeilen zu Dokumentationszwecke eingefügt oder Parameter ausgeblendet werden.

Zum Erstellen einer OPC.DA Serverkonfiguration, kann die Vorlagendatei "OPCServer.ini.template" verwendet werden. Die Vorlagendatei wird mit jeder IndraWorks Suite "Engineering" oder "Communication" Installationsvariante im Verzeichnis "%CommonProgramFiles%\Rexroth\" abgelegt.

Der Platzhalter oder die Umgebungsvariable "%CommonProgramFiles%" wird vom Betriebssystem in Abhängigkeit der installierten Betriebssystemsprache festgelegt und kann mit der Eingabe der Zeile "echo %CommonProgramFiles%" in einer Eingabeaufforderung erfragt werden. Auf PCs mit einem deutschen Betriebssystem wird der Pfad "C:\Programme\Gemeinsame Dateien" und bei englischen oder MUI (Multilingual User Interface) Betriebssystemen der Pfad "C:\Program Files\Common Files" ausgegeben.

In welchem Verzeichnis die Konfigurationsdatei "OPCServer.ini" für den OPC.DA Server liegen muss, wird in den Abschnitten [Windows XP, Seite 110](#), und [Ablageverzeichnis der OPC.DA Server Konfigurationsdatei, Seite 110](#) für PC Systeme und in [Windows CE, Seite 111](#) für CE Systeme beschrieben.

Kommunikationsschnittstellen

```

[[Server]
Version=0x305040A
updaterate=200
logevents=1
logfilter=15
syncinit=1
ReadCyclesAfterWrite=0
UseColonAsPlcDevId=0
GroupUpdateBehaviour=0
PLCs=1
PLC0=PLC1

[PLC:PLC1]
interfacetype=GATEWAY3
reconnecttime=15
waittime=10
active=1
logevents=1
logfilter=15
gateway=Tcp/Ip
gatewayaddress=localhost
gatewayport=1317
parameters=1
parameter0=Address
value0=0000.c0a8.0001

[Redundancy]
NumOfGroups=0

```

Abb. 4-13: Konfigurationsdatei Vorlage des IndraLogic OPC.DA Server

Die Konfigurationsdatei besteht immer aus einer Serversektion und n Steuerungssektionen. In den folgenden Tabellen [OPC.DA Server Konfigurationsdatei - Server'-Sektion \[Server\]](#), Seite 106, und [OPC.DA Server Konfigurationsdatei - Steuerungs-Sektion \[PLC:...\]](#), Seite 107, sind die wesentlichen Konfigurationsparameter für die Server- bzw. Steuerungssektion der Datei "OPC-Server.ini" beschrieben.

Parametername und Parameterwert	Beschreibung
[Server]	Sektionsbezeichner der "Server" Sektion
updaterate=200	Aktualisierungsrate des OPC.DA Servers in Millisekunden. In diesem Intervall erfolgt die Aktualisierung des internen OPC Cache. Die Einstellung für die OPC Client Aktualisierung wird durch die Eigenschaften der OPC Gruppe bzw. Subscription festgelegt. Abhängig von der Anzahl der angemeldeten Variablen ist es notwendig die Aktualisierungsrate zu ändern, damit für die anderen laufenden Prozesse noch genügend Ressourcen von dem PC vorhanden sind. Auch sollte das Aktualisierungsintervall von 200 ms nicht unterschritten werden.
logevents=1	Einschalten des Log-Mechanismus. Es werden alle Server Ereignisse entsprechend der Einstellung von dem Parameter "logfilter" in die "OPCServer.log" Datei geschrieben. Die Log-Datei befindet sich im gleichen Verzeichnis, wie die "OPCServer.ini" Datei. Bei der Inbetriebnahme bzw. bei Tests kann der Parameter "logevents" auf "1" gesetzt werden. Im Betrieb ist das Logging wieder zu deaktivieren, d.h. "0".

Kommunikationsschnittstellen

Parametername und Parameterwert	Beschreibung
logfilter=15	Filter Definition für die Server Ereignisse. <i>Mögliche Werte sind:</i> 15 0x0000 0000 0000 000F: Standard Logging 47 0x0000 0000 0000 002F: Standard Logging und zusätzliche Einträge 63 0x0000 0000 0000 003F: Standard Logging, zusätzliche Einträge und Debug-Einträge
syncinit = 1	Ist diese Option aktiviert, wird eine synchrone Verbindung initialisiert.
ReadCyclesAfterWrite=0	Nach jedem geschriebenen Item sendet der OPC-Server einen Callback an den Client. Früher konnte es im Fall von sich schnell ändernden Werten dazu kommen, dass nicht für jede Änderung ein Callback gesendet und somit der Wert im Client nicht aktualisiert wurde. Um in jedem Fall eine Aktualisierung zu gewährleisten, kann die Einstellung „ReadCyclesAfterWrite“ verwendet werden. Sie gibt die Anzahl von Lese-Zyklen nach dem letzten Schreiben an, nach denen die Werte im Client auf jeden Fall aktualisiert werden sollen, auch wenn sie sich nicht geändert haben. (Standard: Callback wird nach 0 Lesezyklen erzeugt; eine andere gewünschte Zyklusanzahl muss händisch in die ini-Datei eingetragen werden)
UseColonsAsPlcDevider=0	Sofern dieser Wert nicht gleich 0 ist, wird der erste „:“ (z.B. der Punkt nach dem SPS-Namen) durch einen Doppelpunkt ersetzt.
GroupUpdateBehaviour=0	In diesem Fall werden Data-changed-Callbacks (Rückruffunktion bei Datenänderungen) für das aktivierte/deaktivierte Item erzeugt. Ist der Eintrag auf 0, dann werden Rückruffunktionen für alle Items in der Gruppe erzeugt, immer wenn ein (oder mehrere) Item(s) innerhalb der Gruppe aktiviert oder deaktiviert werden
PLCs=1	Anzahl der konfigurierten Steuerungen.
PLC0=PLC1	Auflistung aller konfigurierten Steuerungen in der Konfigurationsdatei.
PLC...	Hierbei ist zu beachten, dass der Zähler Index (Parametername) fortlaufend, und dass für alle gelisteten Steuerungen (Parameterwert z.B. "PLC1") eine eindeutige Steuerungssektion vorhanden ist!

Tab. 4-8: OPC.DA Server Konfigurationsdatei - "Server"-Sektion [Server]

Parametername und Parameterwert	Beschreibung
[PLC:PLC1]	Sektionsbezeichner der Steuerungssektion "PLC1". Soll eine Verbindung beim Start des OPC.DA Servers zu dieser Steuerung hergestellt werden, muss diese in der Auflistung der Steuerungen von der Server Sektion in einem der Parameter "PLC..." (z.B. "PLC0=PLC1") als Wert vorhanden sein.
interfacetype=GATEWAY3	Definition des Steuerungstyps. Dieser Parameter definiert das Steuerungssystem, mit dem kommuniziert wird. Für IndraLogic 2G Steuerungssysteme wird GATEWAY3 verwendet
reconnecttime=15	Zeitintervall in Sekunden, in dem versucht wird, die Verbindung wieder aufzubauen.
waittime=10	Wartezeit in Sekunden der unterlagerten Kommunikationsschicht "PLC Handler". Das ist die maximale Wartezeit, um eine Verbindung zur Steuerung herzustellen.

Kommunikationsschnittstellen

Parametername und Parameterwert	Beschreibung
active=1	Steuerung aktivieren. Über diesen Parameter kann die Kommunikation zur Steuerung aktiv oder inaktiv geschaltet werden. Z.B. kann die Konfiguration einer Anlage bereits erstellt und Module, die nicht vorhanden sind deaktiviert werden.
logevents=1	Einschalten des Log-Mechanismus. Es werden alle Steuerungsereignisse entsprechend der Einstellung von dem Parameter "logfilter" in die "OPCServer.log" Datei geschrieben. Die Log-Datei befindet sich im gleichen Verzeichnis wie die "OPCServer.ini" Datei. Bei der Inbetriebnahme bzw. bei Tests kann der Parameter "logevents" auf "1" gesetzt werden. Im Betrieb ist das Logging wieder zu deaktivieren, d.h. "0".
logfilter=15	Filter Definition für die Server Ereignisse. <i>Mögliche Werte sind:</i> • 15 0x0000 0000 0000 000F: Standard Logging • 47 0x0000 0000 0000 002F: Standard Logging und zusätzliche Einträge • 63 • 0x0000 0000 0000 003F: Standard Logging, zusätzliche Einträge und Debug-Einträge
gateway=Tcp/Ip	Kommunikationseinstellungen vom OPC.DA Server zum "GatewayService" • gateway – Protokoll, welches für die Kommunikation verwendet wird. • gatewayaddress – IP4 Adresse oder Hostname zu dem Gateway, welches die Verbindung zur Steuerung herstellt. • gatewayport – Netzwerkport an dem der "GatewayService" eingehende Verbindungen annimmt. Diese Einstellung muss mit der Gateway Konfiguration übereinstimmen.
gatewayaddress=localhost	
gatewayport=1317	
parameters=1	Anzahl der verwendeten Parameter. In Abhängigkeit der Konfiguration kann die Anzahl der verwendeten Parameter variieren
parameter0=Address	Definiton der Kommunikationsadresse der Steuerung. Diese wird, wie unter Kommunikationsadresse ermitteln, Seite 102 , beschrieben, für den Parameter "value0" übernommen.
value0=0000.c0a8.0001	
Erweiterte Parametereinstellungen	
parameter1=Port	Port des TCP/IP Blocktreibers (In der Standardkonfiguration nicht verwendet)
value1=1234	Über den zugehörigen Parameter "value1" wird die Adresse übernommen
parameter1=DirectAdressesOnly	Für IndraLogic 2G Steuerungen nicht verwendet

Kommunikationsschnittstellen

Parametername und Parameterwert	Beschreibung
parameter2=CheckData-ChangeOnPlc	Wenn dieser Parameter aktiv ist ("value<Nummer>" = 1) liefert die Steuerung nur die geänderten Werte. Die Option benötigt mehr Ressourcen auf der Steuerung und kann ggf. zu mehr Kommunikationslast führen. Sie sollte nur in geschlossenen Systemen mit wohlbekannten Applikationsdaten verwendet werden
parameter3=ClientAddress-Resolution	Für IndraLogic 2G Steuerungen nicht verwendet
parameter4=DontExpand-SimpleTypeArrays	Während des Verbindungsaufbaus werden alle in der Symbolkonfiguration der Steuerung vorhandenen Datentypen geladen. Anschliessend werden komplexe Datentypen (Arrays, Strukturen) expandiert um alle Elementaren Datentypen in der Symbolkonfiguration darzustellen. Für Intelligente Client Applikationen, welche die Elementardatentypen selber expandieren können, kann dieses Feature unterdrückt werden ("Value<Nummer>" = 1)
parameter5=WriteThrough-ReadCache	Standardmäßig beeinflussen alle synchronen Schreibbefehle an die Steuerung die zyklisch zu lesenden Daten nicht Wenn ein Wert gleichzeitig vom Client (Visualisierung) und von der Steuerung geschrieben wird, kann es zu folgendem Effekt kommen: Abhängig vom Client kann nach dem Schreibvorgang ein kurzes Flackern des Variablenwertes mit dem vorherigen Wert der Variable auf der Steuerung auftreten. Wenn nicht garantiert werden kann, dass Variablen niemals gleichzeitig von Client und Steuerung geschrieben werden, kann in der Steuerung bei Jedem Schreibauftrag geprüft werden, ob die zu schreibenden Variablen auch Teil der zyklischen Leseliste sind. Falls dies der Fall ist, wird der Cachewert der betroffenen Variablen auf den geschriebenen Wert gesetzt und der Datachange Callback wird für Jede Liste aufgerufen. Es muss hierbei beachtet werden, dass ein registrierter Callbacks häufiger aufgerufen wird. Dieses Feature nimmt während jedes Schreibvorgangs und proportional zur Anzahl der zu schreibenden Variablen als auch zur Anzahl der Variablen pro zyklischer Leseliste zusätzliche Rechenzeit in Anspruch. Zur Aktivierung dieser Option muss "Value<Nummer>" = 1 gesetzt werden
parameter6=AllowAlternate-Symbolfile	Diese Option wird nicht unterstützt
parameter7=DontLoadSymbolsFromPlc	Wenn der Client alle Symbolnamen und Typen kennt, so dass ein Browsing entfallen kann, kann diese Option aktiviert werden (value<Nummer> = 1)
parameter8=DontExpand-ComplexTypeArrays	Diese Option kann verwendet werden um die Expansion von Arrays komplexer Datentypen (Strukturen) zu unterdrücken. Die Option DontExpandComplexTypeArrays kann typischerweise nur in geschlossenen Systemen angewendet werden, da der Client das Alignment der Member kennen muss. Falls die Optionen DontExpandComplexTypeArrays und DontExpandSimpleTypeArrays gleichzeitig aktiv sind, werden nur die top level nodes der GVLs bzw. POU's zurückgeliefert. Zur Aktivierung dieser Option muss "Value<Nummer>" = 1 gesetzt werden

Tab. 4-9: OPC.DA Server Konfigurationsdatei - "Steuerungs"-Sektion [PLC:...]

Im Folgenden ist eine IndraLogic OPC.DA Server Konfigurationsdatei abgebildet, die auf PC und CE Systemen mit der Steuerungs IP-Adresse "192.168.0.1" eine Kommunikationsverbindung aufbaut.

Kommunikationsschnittstellen

```

[Server]
Version=0x305040A
updateRate=200
logEvents=1
logFilter=15
syncInit=1
ReadCyclesAfterWrite=0
UseColonAsPlcDevId=0
GroupUpdateBehaviour=0
PLCs=1
PLC0=PLC1

[PLC:PLC1]
interfaceType=GATEWAY3
reconnectTime=15
waitTime=10
active=1
logEvents=1
logFilter=15
gateway=TCP/IP
gatewayAddress=localhost
gatewayPort=1317
parameters=1
parameter0=Address
value0=0000.c0a8.0001

[Redundancy]
NumOfGroups=0

```

Abb. 4-14: Konfigurationsdatei des IndraLogic OPC.DA Server - "OPCServer.ini"



Die OPC.DA Server Konfigurationsdatei "OPCServer.ini" ist für PC und CE Systeme in Verbindung mit IndraLogic 2G Steuerungssystemen gleich.

Ablageverzeichnis der OPC.DA Server Konfigurationsdatei

Allgemeines



Die OPC.DA Server Konfigurationsdatei "OPCServer.ini" von Windows XP und Windows CE sind erst ab der IndraWorks Version 12 kompatibel.

Windows XP

Die OPC.DA Server Konfigurationsdatei "OPCServer.ini" muss im Verzeichnis

C:\Dokumente und Einstellungen\All Users\Anwendungsdaten\Rexroth\IndraWorks bei einer deutschen Betriebssystem Installation oder im Verzeichnis

C:\Documents and Settings\All Users\Application Data\Rexroth\IndraWorks bei einer englischen bzw. MUI Betriebssystem Installationen liegen.

In der folgenden Grafik ist das Ablageverzeichnis für eine deutsche Betriebssystem Installation dargestellt:

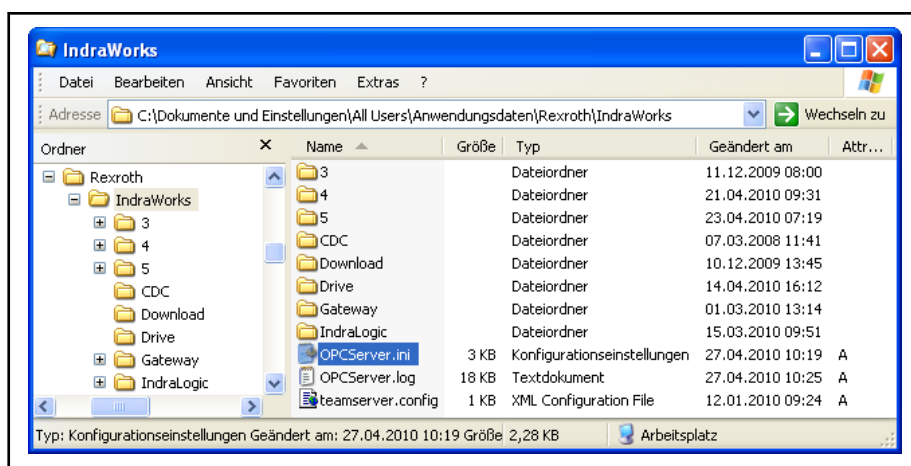


Abb. 4-15: Ablageverzeichnis der Konfigurationsdatei "OPCServer.ini" auf PC Systemen mit Windows XP

Windows 7

Die OPC.DA Server Konfigurationsdatei "OPCServer.ini" muss im Verzeichnis C:\ProgramData\Rexroth\IndraWorks abgelegt werden.

In der folgenden Grafik ist das Ablageverzeichnis dargestellt:

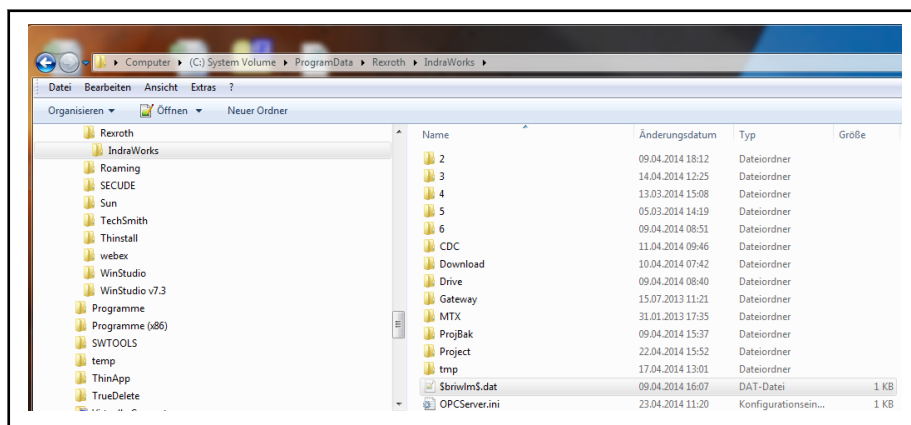


Abb. 4-16: Ablageverzeichnis der Konfigurationsdatei "OPCServer.ini" auf PC Systemen mit Windows 7

Windows CE

Bei den Windows CE Geräten muss die OPC.DA Server Konfigurationsdatei "OPCServer.ini" für das Visualisierungsgerät "VEPxx.4" in das Verzeichnis \StorageCard\Rexroth\Controls\OPCServer abgelegt werden.

In den Abbildungen [Abb. 4-17 "Ablageverzeichnis der Konfigurationsdatei OPCServer.ini auf CE Systemen - CE Explorer"](#) auf Seite 112 und [Abb. 4-18 "Ablageverzeichnis der Konfigurationsdatei OPCServer.ini auf CE Systemen - XP Explorer über FTP Verbindung"](#) auf Seite 112 ist das Ablageverzeichnis mit dem "CE Windows Explorer" und dem "XP Windows Explorer" für ein "VEPxx.4" Visualisierungsgerät dargestellt.

Kommunikationsschnittstellen

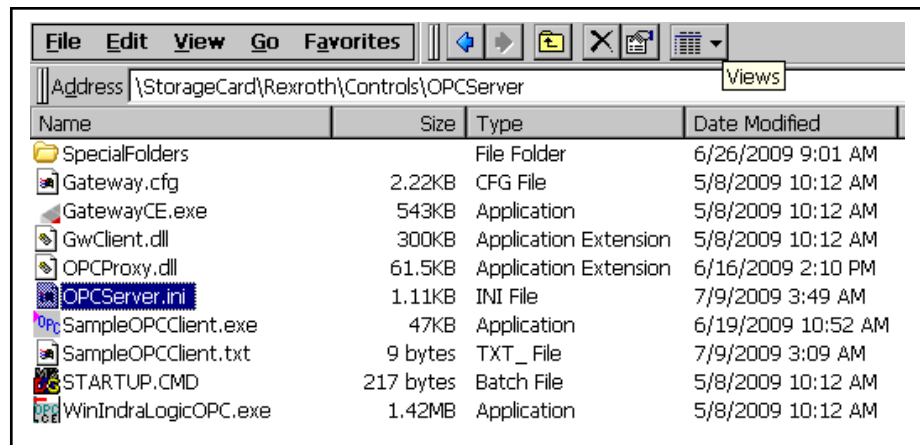


Abb. 4-17: Ablageverzeichnis der Konfigurationsdatei "OPCServer.ini" auf CE Systemen - CE Explorer

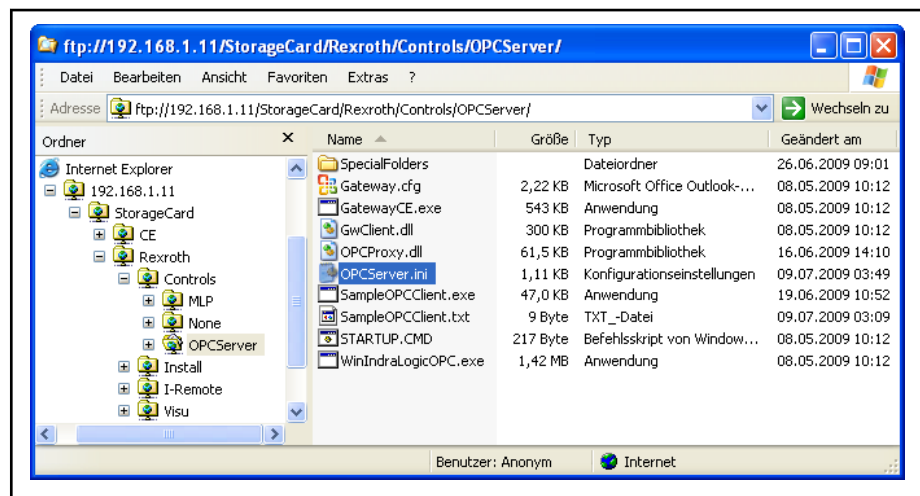


Abb. 4-18: Ablageverzeichnis der Konfigurationsdatei "OPCServer.ini" auf CE Systemen - XP Explorer über FTP Verbindung

Zum Aktivieren der OPC.DA Server Kommunikation, ist in den "CE Rexroth Settings" (Verknüpfung "RxSettings") im Gruppierungsfeld "Control" von dem Register "Application Settings" der "OPCServer" auszuwählen. Die Auswahl in der Gruppierung "Visualization" wird in Abhängigkeit des eingesetzten Visualisierungssystems vorgenommen. Siehe Abb. 4-19 "Dialog: CE Rexroth Settings (Konfiguration einer OPC Kommunikation)" auf Seite 112.

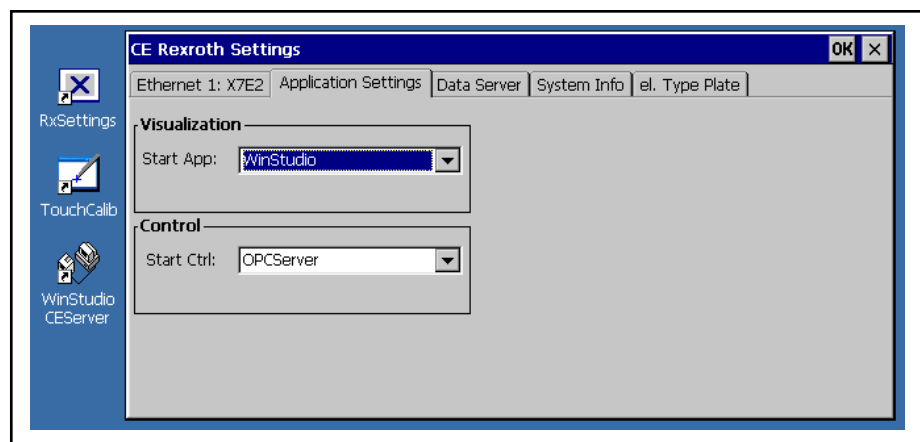


Abb. 4-19: Dialog: CE Rexroth Settings (Konfiguration einer OPC Kommunikation)

OPC.DA Server - Item Syntax

Der IndraLogic OPC.DA Server kann auf alle SPS Variablen zugreifen, die im IndraWorks Projekt für den Zugriff in der Symbolkonfiguration oder über Pragmas konfiguriert wurden. Die Item Syntax setzt sich hierbei aus dem "Steuerungsnamen" (Konfigurationsdatei "OPCServer.ini") und den "voll qualifizierenden Variablennamen" zusammen. D.h., dem Variablennamen werden alle Namensräume vorangestellt, in denen die Variable deklariert wurde.

Als Trennzeichen für Steuerungsname und voll qualifizierender Variablenname wird der Punkt "." verwendet.

Siehe folgende Zusammenfassung:

- Item Syntax
 - "[Steuerungsname].[voll qualifizierender Variablenname]"

Beispiel:

"PLC_IL2G.Application.HMIVARS.bLifeSign_hmi"



Bei der Verwendung der IndraLogic Funktionalität zur Namensraumersetzung per Pragma

{attribute 'namespace' := 'NewNamespace'}

oder über die Symbolkonfiguration, erfolgt der Zugriff über den neu definierten Namensraum.

Beispiel:

"PLC_IL2G.NewNamespace.bLifeSign_hmi"

4.4.3 Haftungsausschluss

WARNUNG

Keine Haftung für Schäden durch Kunden-OPC-Clients

Über OPC haben Sie uneingeschränkten Zugriff auf alle Geräte-Daten und die gesamte Geräte-Funktionalität.

Für Schäden und Datenverlust, die auf eine unsachgemäße Verwendung von OPC in "Third Party" OPC-Client Applikationen und WinStudio-Bedienbilder oder auf eine Umgehung bzw. Nichtbeachtung von gerätespezifischen Sicherheitsrichtlinien zurückzuführen sind, übernimmt Bosch Rexroth keine Haftung. Dies betrifft insbesondere das Schreiben gerätesensibler und sicherheitsrelevanter Daten.

4.5 Konfigurieren einer OPC Clientanbindung mit WinStudio

Zur Konfiguration einer Verbindung zu einem OPC Server (OPC.lwSCP, OPC.DA) werden in WinStudio entsprechende Treiberblätter verwendet. Sie können für jeden Konfigurierten Server mehrere Treiberblätter anlegen. Hierbei können Sie für jedes angelegte Treiberblatt die Kommunikationsparameter (Aktualisierungsraten, zu kommunizierende Variablen, etc.) individuell festlegen.

Zum Anlegen einer solchen Verbindung fügen Sie im Projektextplorer im Knoten **WinStudio ▶ Variablen ▶ Kommunikation ▶ OPC** des jeweiligen Visualisierungsprojektes ein neues OPC-Treiberblatt ein.

Kommunikationsschnittstellen

Abb. 4-20: OPC-Treiberblatt

Verwenden Sie die folgenden Parameter für die Konfiguration eines OPC-Treiberblatts:

- **Beschreibung:** Geben Sie zu Dokumentationszwecken eine Beschreibung für das OPC-Arbeitsblatt ein.
- **Server-Auswahl:** Wählen sie hier, je nach vorausgegangener Konfiguration, den OPC.LWSCP bzw. den IndraLogic OPC.DA Server für IndraLogic 2G Steuerungen aus.
- **Deaktivieren:** Geben Sie einen Tag oder eine Konstante mit einem Wert ungleich "0" ein, um die Kommunikation mit dem OPC-Server zu deaktivieren. Geben Sie "0" ein oder lassen Sie das Feld leer, um die Kommunikation zu aktivieren.
- **Aktualisierungsrate Lesen:** Legen Sie fest, wie oft der Server diese Gruppe aktualisieren soll (in Millisekunden). Bei Eingabe von "0" verwendet der Server die schnellste sinnvolle Rate.
- **Prozent Totband** (nur für analoge Elemente gültig): Legen Sie fest, bei welcher prozentualen Änderung eines Elementwerts eine Benachrichtigung durch den Server erfolgt.
- **Status:** Geben Sie den Namen eines Tags ein, um den Status der Verbindung zu erhalten. Der Status "Gut" entspricht dem Wert 1.
- **Remote-Server-Name:** Knotenname oder IP-Adresse des Servers im Knotennetzwerk.
- **Lesen vor Schreiben:** Bei Aktivierung dieser Option werden direkt vor dem Schreiben von neuen Variablenwerten auf den Server die Ursprungswerte vom OPC-Server gelesen. Das Projekt puffert dazu erst die neuen zu schreibenden Werte und liest dann die Ursprungswerte vom Server. Erst nach der Synchronisation des Projekts mit dem Server werden die neuen Werte vom Puffer zum Server geschrieben.
- **Lesen nach Schreiben:** Bei Aktivierung dieser Option werden direkt nach dem Schreiben von neuen Variablenwerten auf den Server die Werte vom OPC-Server zurückgelesen. Weitere Informationen über diese Einstellung finden Sie in der IndraWorks Onlinehilfe "Rexroth Win-Studio 7.3 Technische Referenz".
- **Annehmen Variablen-Name in Item-Spalte:** Wenn diese Option aktiviert ist, wird der in geschweiften Klammern eingeschlossene Text im Feld "Element" als Tag-Name (von Typ String) interpretiert. In diesem Fall

Kommunikationsschnittstellen

wird der Wert dieses Tags als Name des Elements vom OPC-Server verwendet. Dadurch können Sie während der Laufzeit die Verknüpfung eines Tag mit OPC-Elementen ändern, indem Sie den Wert des konfigurierten Tags ändern. Wenn die Option **Annehmen Variablen-Namen in Item-Spalte** nicht aktiviert ist, werden alle in der Spalte "Element" konfigurierten Zeichen als Bestandteil des Elementnamens betrachtet (einschließlich der Zeichen in geschweiften Klammern).

- **Variablenname:** Geben Sie die Namen der mit den Serverelementen zu verknüpfenden Tags ein.
- **Element:** Geben Sie die Itemstrings der Variablen, die Sie mit der Steuerung austauschen möchten, an.

1. Mit Hilfe des WinStudio OPC-Browsers : Hierbei muss eine Verbindung zur Steuerung vorhanden sein

Führen Sie einen Rechtsklick auf das Feld **Element** aus und wählen Sie die Option **OPC-Browser**. Es werden die Steuerungen aus der jeweiligen OPC Konfiguration dargestellt. Die in der Symbolkonfiguration der jeweiligen Steuerung vorhandenen Variablen werden in einer Baumstruktur dargestellt.

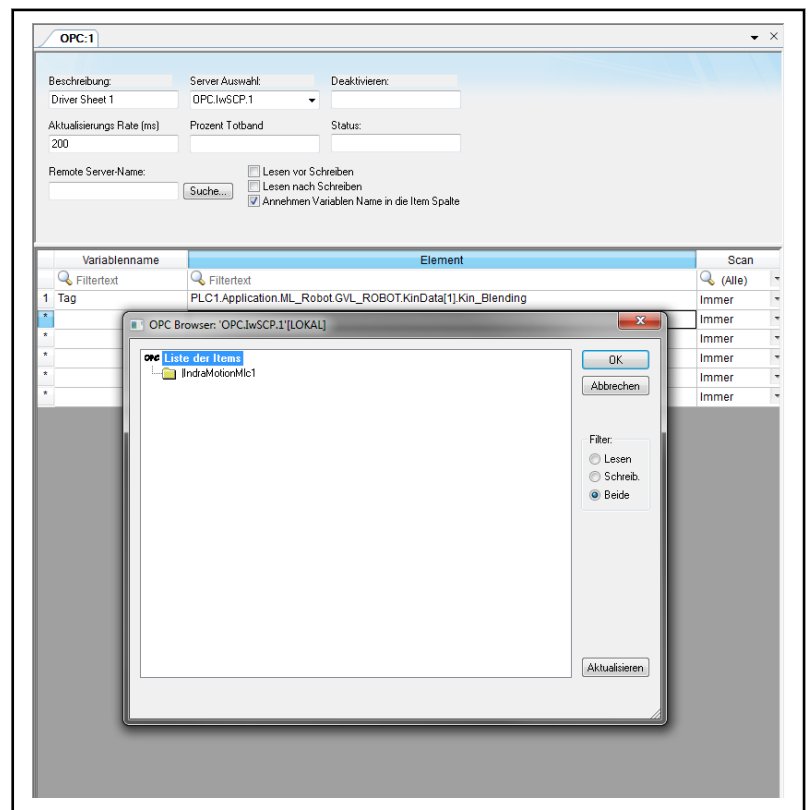


Abb. 4-21: OPC-Browser mit OPC Treiberblatt. Hier ist der OPC.IwSCP Server ausgewählt

2. Mit Hilfe des IndraWorks Werkzeugfensters für Prozessvariablen: Hier ist keine Verbindung zur Steuerung notwendig

Aktivieren Sie das Werkzeugfenster über **Ansicht ► Andere Fenster ► Prozessvariablen** in der IndraWorks-Werkzeugeiste.

Wurde die Konfiguration des OPC-Treiberblattes gespeichert, so erscheinen hier die Symbolvariablen der im Projekt vorhandenen Steuerungen. Diese können Sie bequem mittels Drag&Drop in die entsprechende Treiberblattzeile ziehen. Falls sich die Steuerung nicht im Projekt befindet, können Sie die Symbolkonfigurationsda-

Kommunikationsschnittstellen

tei der externen Steuerung laden und somit auch diese in ihrem HMI-Projekt projektieren. Weitere Informationen über das Werkzeugfenster für Prozessvariablen finden Sie in der IndraWorks Online-Hilfe

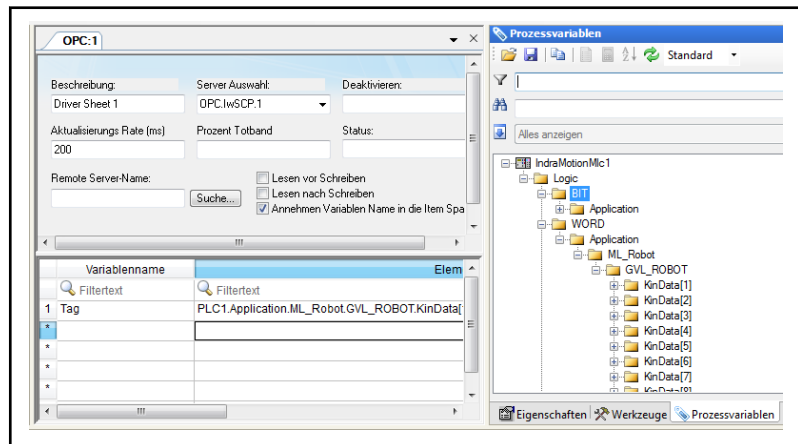


Abb. 4-22: OPC Treiberblattkonfiguration und Werkzeugfenster für Prozessvariablen

- **Scan:** Legt die Bedingungen fest, unter denen der Tag-Wert vom Remote-Gerät oder -Server gelesen und dann in der Projektdatenbank aktualisiert wird. Dabei gelten die folgenden Optionen:
 - **Immer** bedeutet, dass der Tag bei jedem Scan des Kommunikationsarbeitsblatts gelesen und aktualisiert wird, ungeachtet dessen, ob der Tag in anderen WinStudio-Bildern, Skripten oder Arbeitsblättern verwendet wird. Diese Option wird empfohlen für Tags, die kontinuierlich im Hintergrund überwacht werden müssen, wie z.B. Tags zum Auslösen von Alarmen, in Rezepturen verwendete Tags, in der historischen Datenbank erfasste Tags usw.
 - **Bild** bedeutet, dass der Tag nur dann gelesen und aktualisiert wird, wenn er in mindestens einem geöffneten Bild verwendet wird, entweder lokal oder auf einer anderen Client-Station. Diese Option wird für Tags empfohlen, die ausschließlich in Bildern verwendet werden, da diese nicht immer aktualisiert werden müssen. Durch Auswahl dieser Option kann die Leistungsfähigkeit des Projekts verbessert werden.
 - **Auto** bedeutet, dass das Projekt automatisch entweder **Immer** oder **Bild** wählt, je nach dem, ob der Tag in Ihrem Projekt verwendet wird. Wenn der Tag ausschließlich in einem WinStudio-Bild verwendet wird, wird der Scan standardmäßig auf **Bild** zurückgesetzt. Wenn der Tag aber in anderen Arbeitsblättern verwendet wird (z.B. Skript, Berechnung, Alarm, Trend, Rezeptur, Bericht, Scheduler), wird zur Laufzeit der Scan auf **Immer** gesetzt.

Wenn Sie nicht sicher sind, welche Option zu wählen ist, wählen Sie **Immer**. Dadurch wird sichergestellt, dass der Tag gelesen und aktualisiert wird.

- **Div:** Festlegen der Divisionskonstante, wenn eine Anpassung der Skalierung erforderlich ist. Dieser Wert ist ein Divisionsfaktor in einer Lese-Operation und ein Multiplikationsfaktor in einer Schreib-Operation.
- **Hinzufügen:** Festlegen der Additionskonstante, wenn eine Anpassung der Skalierung erforderlich ist. Dieser Wert ist ein Additionsfaktor in einer Lese-Operation und ein Subtraktionsfaktor in einer Schreib-Operation.

Um die OPC-Client-Task zur Laufzeit des Projekts auszuführen, öffnen Sie die Task-Konfiguration (**Werkzeuge** ► **Status** im Kontextmenü des Ordners "WinStudio") und wählen Sie für "OPC-Client Runtime" die Starteinstellung "Automatisch". In diesem Dialog können Sie zum Testen den Task durch Klicken auf **Start** auch manuell starten. Nach Ausführung dieses Programms wird ein kleines Symbol im Infobereich angezeigt.

4.6 OPC UA

4.6.1 Übersicht

Der OPC UA Server steht ab der Version 12VRS in den Systemen "IndraMotion MLC" und "IndraLogic XLC" als Kommunikationsserver für die Anbindung von HMIs, SCADA und Leitstandssysteme zur Verfügung.

Er erlaubt den Zugriff auf alle SPS Variablen, die in der IndraLogic Symbolkonfiguration von der jeweiligen Steuerung für den Zugriff freigegeben wurden.

Die Rexroth Visualisierung "WinStudio" besitzt einen OPC UA Client für die Kommunikationsanbindung an den OPC UA Server in den Steuerungssystemen.

Sie erhalten einen Überblick über

- die OPC UA Architektur
[Einführung in OPC Unified Architecture, Seite 117](#)
- den Bosch Rexroth OPC UA Server
[Der Bosch Rexroth OPC UA Server, Seite 123](#)
- das Erstellen einer Kommunikationsverbindung zum Bosch Rexroth OPC UA Server
[Kommunikationsbeispiel über OPC UA, Seite 128](#)
- das Erstellen einer Kommunikationsverbindung zum Bosch Rexroth OPC UA Server mittels WinStudio



Zur Diagnostizierung von Verbindungsproblemen zwischen einem "OPC UA Client" und dem "Bosch Rexroth OPC UA Server" wird die Software "Wireshark" in der Version 1.6 oder höher empfohlen.

4.6.2 Einführung in OPC Unified Architecture

Allgemein

OPC UA ist die Weiterentwicklung des OPC Industriestandards. Die serviceorientierte Architektur gewährleistet durch den Verzicht auf eine DCOM Basis Plattformunabhängigkeit, Skalierbarkeit und Hochverfügbarkeit. OPC UA ermöglicht eine lückenlose vertikale Integration von der Leitebene bis hin zur Automatisierungskomponente – unabhängig von der Programmiersprache oder dem Betriebssystem. OPC UA ist ein Client / Server-System. Mehrere Clients können gleichzeitig auf einen Server zugreifen. Ein Client kann auf mehrere Server zugreifen.

Übersicht der Spezifikationen

OPC Unified Architecture wurde von der OPC Foundation als Multipart-Spezifikation in 13 Teilen <https://opcfoundation.org/developer-tools/specifications-unified-architecture/> veröffentlicht. Der Download ist jedoch nur für Mitglieder der OPC-Foundation möglich.

Kommunikationsschnittstellen

Part	Spezifikation
Part 1	OPC UA Part 1 - Overview and Concepts 1.01 Specification
Part 2	OPC UA Part 2 - Security Model 1.01 Specification
Part 3	OPC UA Part 3 - Address Space Model 1.01 Specification
Part 4	OPC UA Part 4 - Services 1.01 Specification
Part 5	OPC UA Part 5 - Information Model 1.01 Specification
Part 6	OPC UA Part 6 - Mappings 1.00 Specification
Part 7	OPC UA Part 7 - Profiles 1.00 Specification
Part 8	OPC UA Part 8 - Data Access 1.01 Specification
Part 9	OPC UA Part 9 - Alarms and Conditions 1.00 Specification
Part 10	OPC UA Part 10 - Programs 1.00 Specification
Part 11	OPC UA Part 11 - Historical Access 1.00 Specification
Part 12	OPC UA Part 12 - Discovery
Part 13	OPC UA Part 13 - Aggregates

OPC UA Informationsmodell

Das OPC UA Informationsmodell basiert auf Nodes, die einen objektorientierten Kontext beschreiben. Ein Node kann aus Attributen, Methoden und Ereignissen (Events) bestehen. Der Inhalt ist abhängig von der NodeClass. Das Typmodell erlaubt die Darstellung sämtlicher Typen von Nutzdaten bis hin zu Metadaten. Jeder Node hat eine eindeutige NodeId. Die Verbindungen zwischen Nodes werden durch References beschrieben. Die Semantik der Reference wird durch die ReferenceType festgelegt.

Das OPC UA Informationsmodell wird von anderen Organisationen, wie z. B. PLCOpen, als Basis verwendet und als eigenes Profile erweitert. Weitere Informationen über diese Erweiterungen findet man unter dem Stichwort Companion Specification bei den jeweiligen Organisationen bzw. bei der OPC Foundation. Jeder OPC UA Server teilt auf Anfrage mit, welche Profiles er unterstützt.

Serviceorientierte Architektur

Die Architektur von OPC UA ist in logischen Ebenen aufgeteilt.

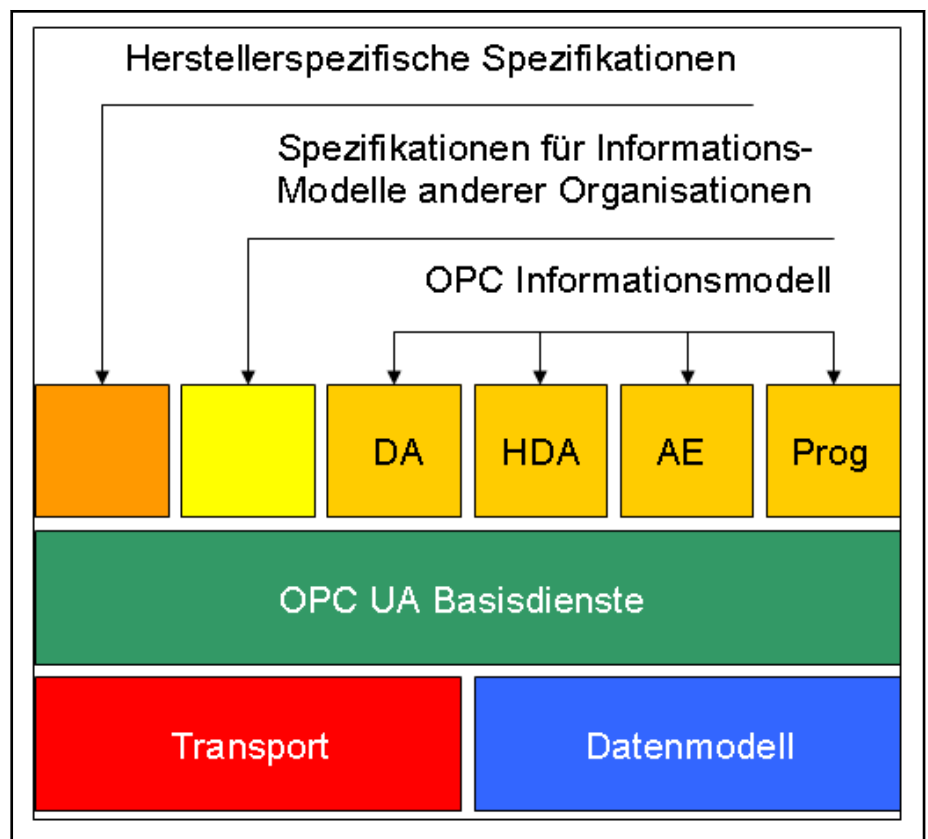


Abb. 4-23: OPC UA Architektur

Transportebene

Die Ebene Transport serialisiert und deserialisiert die Daten und versendet bzw. empfängt sie. Übertragen werden die Daten alternativ mittels XML-Streams über das HTTP-HTTPS Protokoll oder über ein performantes binäres TCP-Protokoll.

OPC UA Basisdienste

Alle OPC UA Basisdienste sind abstrakte Methodenbeschreibungen. Sie sind unabhängig vom Protokoll und sind Basis für alle OPC UA Funktionen. Die Methoden werden in Service Sets zusammengefasst.

Die wichtigsten Service Sets sind im folgenden Abschnitt erwähnt:

Discovery Service Set

- Der Discovery Service Set definiert Dienste mit denen ein Client die Endpoints eines Servers und deren Sicherheitskonfiguration feststellen kann. Ein Endpoint versteht sich als Variante, bestehend aus Kommunikationsprotokoll, Hostadresse, Portnummer und Securityeinstellung.

Kommunikationsschnittstellen

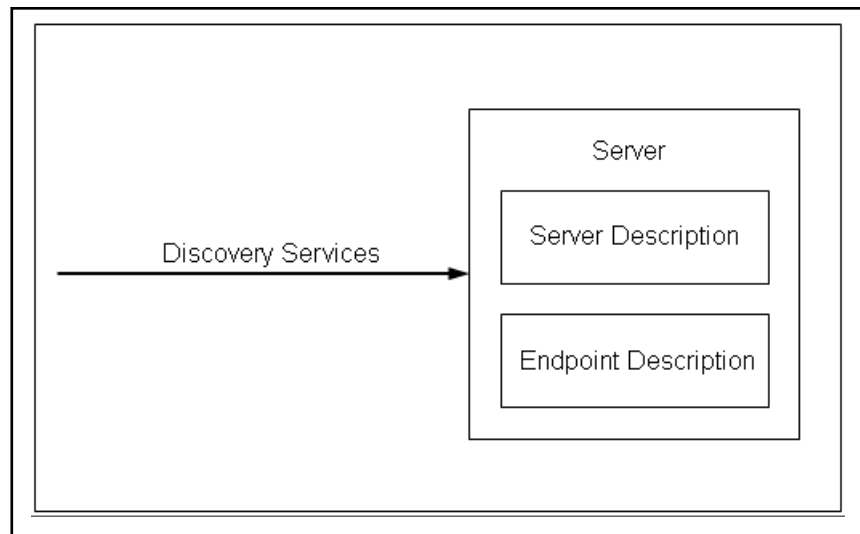


Abb. 4-24: Discovery Service Set

- Session Service Set**
- Eine Session stellt eine logische Verbindung auf Applikationsebene zwischen Client und Server dar. Sie wird durch den Client erstellt und kann clientspezifische Nutzer- und Spracheinstellungen enthalten.

Der Aufbau einer Session erfolgt über den Service CreateSession. Der Client erhält dabei eine Session-Id. Über diese Session-Id kann der Client nachfolgend Datendienste (Read, Write, Subscribe etc.) aufrufen.

Eine Session wird nach Anforderung durch den Client (CloseSession) bzw. nach Ablauf eines Timeout automatisch durch den Server beendet.

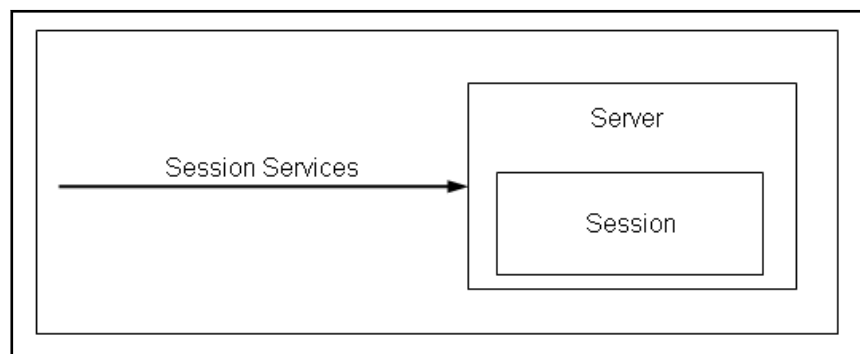


Abb. 4-25: Session Service Set

- View Service Set**
- Der View Service Set ermöglicht dem Client durch eingeschränkte Adressräume (den sogenannten Views) des OPC UA Servers zu browsen. Der Adressraum dient dazu, die verfügbaren Informationen des OPC UA Servers einem Client zu präsentieren. Er wird im wesentlichen aus einer Menge von Nodes gebildet. Nodes sind typisierte Objekte und dienen weitestgehend der Abbildung von "Real World" - Objekten. Die wichtigsten Nodeklassen sind Variable, Object und Method.

Verbindungen zwischen den Nodes werden über Referenzen hergestellt. Man unterscheidet zwischen hierarchischen und nichthierarchischen Referenzen. Hierarchische Referenzen werden z.B. verwendet um eine strukturelle Ordnung der Nodes darzustellen. Nichthierarchische Referenzen werden z.B. verwendet um auf die Typdefinition einer Node zu verweisen.

Das Typsystem von Nodes und Referenzen umfasst Typen für Objekte, Referenzen, Variablen, Events und Daten. Dieses kann durch Ableitung

Kommunikationsschnittstellen

bestehender Typen erweitert und so mit einer tieferen Semantik versehen werden.

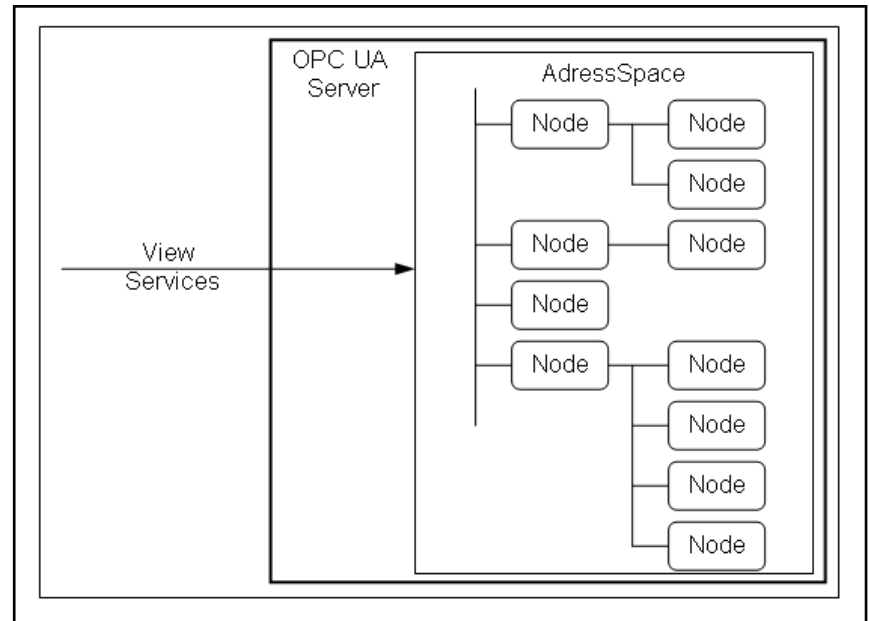


Abb. 4-26: View Service Set

- Attribute Service Set**
- Eine Node besteht aus mehreren Attributen. Diese Attribute dienen der näheren Beschreibung einer Node. Welche Attribute eine Node besitzt, wird durch die Nodeklasse festgelegt. So enthalten nur Nodes der Nodeklasse Variable das Attribut Value.

Alle Nodeklassen haben ein einheitliches Set von Basisattributen. Diese umfassen u.a. die NodeId zur eindeutigen Adressierung einer Node, die NodeClass und den BrowseName.

Mittels des Attribute Service Set kann auf diese Attribute lesend und schreibend zugegriffen werden.

Kommunikationsschnittstellen

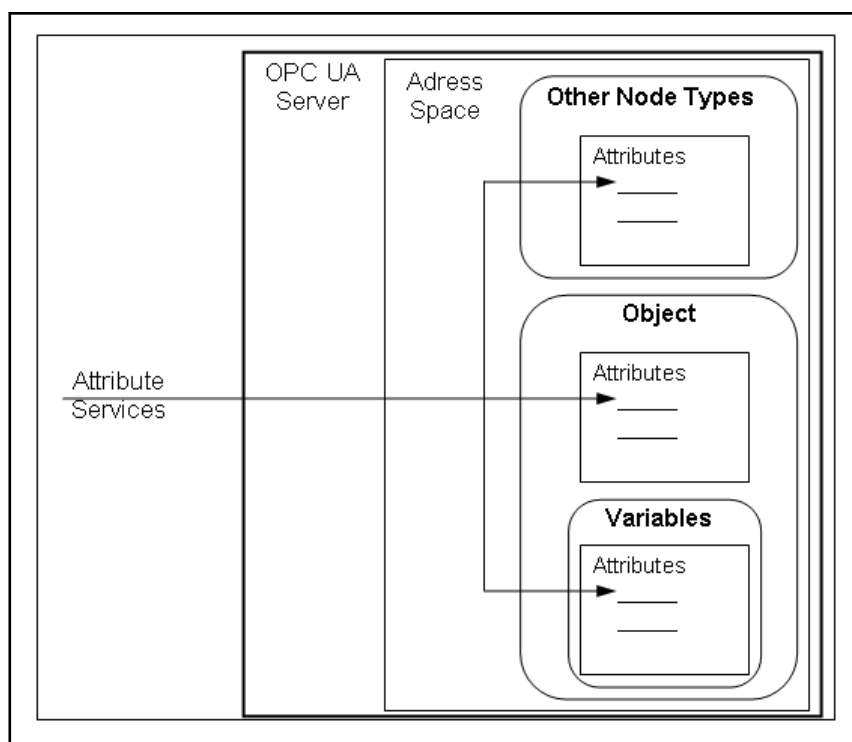


Abb. 4-27: Attribute Service Set

- Subscription Service Set**
- Subscriptions stellen in OPC UA eine effiziente Methode dar, um sich vom Server über Wertänderungen im OPC UA Server informieren zu lassen. Mit dem Subscription Service Set können Clients Subscription (Abonnements) anlegen, modifizieren und löschen.

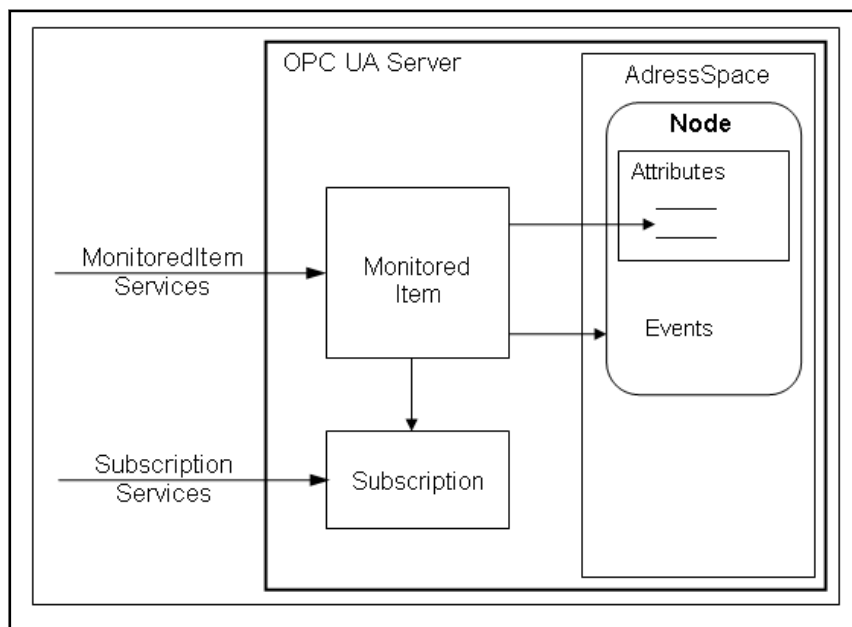


Abb. 4-28: Subscription Service Set

- MonitoredItem Service Set**
- Als MonitoredItem bezeichnet man ein Value Attribut, eine Aggregation eines oder mehrerer Value Attribute oder ein Event, das in den Kontext einer Subscription gebracht wird. Ändert sich der Wert dieser Value Attribute oder wird ein entsprechendes Event geworfen, wird dieser automatisch an den Client übertragen.

Mit dem MonitoredItem Service Set können Clients MonitoredItems im OPC UA Server anlegen, modifizieren und löschen.

4.6.3 Der Bosch Rexroth OPC UA Server

Allgemeines

Der Bosch Rexroth OPC UA Server ist eine in die Firmware integrierte Komponente. Sie ermöglicht eine sichere Verbindung über ein standardisiertes Kommunikationsprotokoll für den Zugriff auf die Steuerung. Clientseitige Anpassungen bzw. Treiberentwicklungen sind nicht erforderlich. Der OPC UA Server wird per Default beim Hochlauf der Steuerung initialisiert und gestartet. Wird dieser nicht benötigt, so kann er deaktiviert werden um die CPU-Last zu verringern.

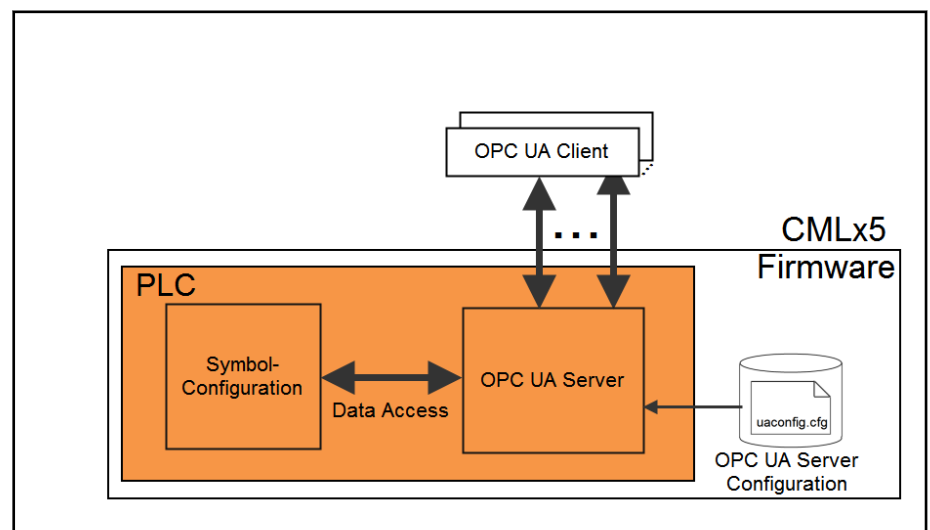


Abb. 4-29: Integration OPC UA Server

Eigenschaften des Bosch Rexroth OPC UA Servers

Im Folgenden werden die Eigenschaften des Bosch Rexroth OPC UA Servers dargestellt.

EndpointURL	Die EndpointURL ist Bestandteil des Endpoints und setzt sich aus dem verwendeten Kommunikationsprotokoll und der Hostadresse zusammen. Als Kommunikationsprotokoll definiert OPC UA ein performantes Binärprotokoll (opc.tcp://) und ein XML basierte Protokoll (http://). Der Bosch Rexroth OPC UA Server unterstützt das Binärprotokoll und verwendet per Default den Port 4840. Aufbau der EndpointURL: opc.tcp://<hostadresse>:4840
MessageSecurityMode	Über den MessageSecurityMode wird die Sicherheitsstufe festgelegt. <i>Mögliche Werte sind hier:</i> • Sign: Diese Sicherheitsstufe gewährleistet die Integrität der Nachrichten. Durch das Hinzufügen einer Signatur wird der Nachrichteninhalte gegen das Ändern durch Dritte geschützt. • SignAndEncrypt: Zusätzlich zur Sicherheitsstufe Sign wird hiermit die Vertraulichkeit der Nachrichten sichergestellt. Mittels Kryptoalgorithmen werden die Nachrichten verschlüsselt und sind damit durch Dritte nicht lesbar. • None: Die Nachrichten werden weder signiert noch verschlüsselt. Der Bosch Rexroth OPC UA Server unterstützt nur den MessageSecurityMode None. Einträge unter SecurityPolicy werden ignoriert.

Kommunikationsschnittstellen

SecurityPolicy	Die SecurityPolicy beschreibt die Art der Verschlüsselung von Nachrichten. OPC UA definiert die Policies Basic128Rsa15 und Basic256. Die SecurityPolicy wird nicht unterstützt.
PublishingInterval	Das PublishingInterval definiert den Zyklus, in dem eine Subscription versucht eine Notification an den Client zu senden. Der Wert für das PublishingInterval wird beim Erstellen einer Subscription zwischen dem Client und dem Server festgelegt. Der Bosch Rexroth OPC UA Server unterstützt die folgenden PublishingIntervalle [ms]: 1, 2, 5, 10, 50, 100, 500, 1000, 2000, 5000
SamplingInterval	Das SamplingInterval definiert den Zyklus, in dem der Server die unterlagerten Datenquellen abtastet. Der Wert für das SamplingInterval wird zwischen Client und Server beim Anlegen eines MonitoredItem festgelegt. Der Bosch Rexroth OPC UA Server unterstützt die folgenden SamplingIntervalle [ms]: 1, 2, 5, 10, 50, 100, 500, 1000, 2000, 5000

Tab. 4-10: Eigenschaften des Bosch Rexroth OPC UA Servers



PublishingInterval und SamplingInterval sind vom Client vorgegebene Werte, die sich auf die Steuerungsauslastung durch den OPC UA Server auswirken. Besonders zu klein gewählte Zeitintervalle durch den Client können Rückwirkungen auf das Gesamtverhalten der Steuerung haben!

Konfiguration

Der Opc UA Server ist mittels der in Tabelle Konfiguration aufgeführten Parameter konfigurierbar.

Die Parameter sind dazu in einer Datei abgelegt und können über diese geändert werden.

Der Dateiname ist mit "uaconfig.cfg" festgelegt. Die Datei ist in Abhängigkeit vom verwendeten Steuerungssystem unter folgenden Pfaden zu finden:

- MTX: /plc
- XLC/MLC: /ata0b bzw. USER

Konfiguration

Parameter	Beschreibung
TaskRun	Dieser Parameter dient der Aktivierung bzw. Deaktivierung des OPC UA Servers. Zulässige Wert sind: 0 : der Server ist nicht aktiv 1 : der Server ist aktiv

Kommunikationsschnittstellen

ShortestSamplingInterval	Über diesen Parameter werden der minimale Wert für das PublishingInterval der Subscriptions und das SamplingInterval der MonitoredItems festgelegt. Es definiert somit das schnellste Intervall, in dem Wertänderungen detektiert werden können. Zu beachten ist, dass kleinere Werte zu einer höheren Systembelastung führen. Andererseits können bei einem zu hoch eingestellten Parameter Wertänderungen evtl. nicht mehr detektiert werden. Zulässige Werte sind: 1, 2, 5, 10, 50, 100, 500[ms] Um eine Überlastung der Steuerung zu vermeiden, wird dieser Wert in Abhängigkeit vom verwendeten Steuerungstyp begrenzt. Der kleinste zulässige Wert ist im Konfigurationsfile der Indralogic, Sektion "OPCUAStack" mit dem Eintrag "ShortestSamplingInterval" festgelegt und kann nicht geändert werden.
TcpPort	Über diesen Parameter wird die Portnummer des OPC UA Servers festgelegt. Für OPC UA Kommunikation ist die Portnummer 4840 standardisiert.
Logging	Dieser Parameter ist für zukünftige Erweiterungen reserviert.
LoggingLevel	Dieser Parameter ist für zukünftige Erweiterungen reserviert.

Tab. 4-11: Konfigurationsparameter

Die Konfigurationsparameter werden bei Hochlauf der Steuerung übernommen. Parameteränderungen wirken somit erst nach einem Reboot der Steuerung.

Konfiguration durch den Client

Im OPC UA Server gibt es Eigenschaften, die durch den Client konfiguriert werden. Welche Eigenschaften durch den Client konfiguriert werden können, hängt von der Implementierung des Clients ab. Nachfolgend werden mögliche konfigurierbare Eigenschaften dargestellt.

Durch den Client konfigurierbare Eigenschaften

Parameter	Beschreibung
Sampling Interval	Das SamplingInterval definiert den Zyklus, in dem der Server die unterlagerten Datenquellen abtastet.
Publishing Interval	Das Publishing Intervall beschreibt das Intervall, in dem ein oder mehrere Wertaenderungen für ein oder mehrere Monitored Items an den Client übertragen werden.

Kommunikationsschnittstellen

Queue Size / Discard Oldest	Die Größe der Queue gibt an, wieviele Werte des MonitoredItems in einer Subscription gespeichert werden können. Der maximale Wert beträgt 1000. Über den Parameter Discard Oldest wird festgelegt, wie ein neuer Wert bei einer vollen Queue eingetragen wird. <i>Mögliche Werte für Discard Oldest sind hier:</i> • TRUE: Der älteste Wert in der Queue wird durch den neuen Wert ersetzt. • FALSE: Der neueste Wert in der Queue wird durch den neuen Wert ersetzt.
Filter	Es können unterschiedliche Filtertypen eingestellt werden. <i>Mögliche Werte sind hier:</i> • DataChangeFilter • EventFilter • AggregateFilter Die Bedeutung der Typen sind der jeweiligen Client-Dokumentation oder der OPC UA-Spezifikation zu entnehmen.
Monitoring Mode	Der Monitoring Mode gibt an, ob die Werte eines Monitored Item in die Queue eingetragen und reportet werden. <i>Mögliche Werte sind hier:</i> • Disabled: Der Wert des Monitored Item wird nicht abgetastet und somit nicht in die Queue eingetragen. Eine Notification durch dieses Item findet nicht statt. • Sampling: Der Wert des Monitored Item wird abgetastet und in die Queue eingetragen. Eine Notification durch dieses Item findet nicht statt. • Reporting: Der Wert des Monitored Item wird abgetastet und in die Queue eingetragen. Eine Notification durch dieses Item findet statt.
Enable Publishing	Dieser Parameter legt fest, ob die Daten der Subscription an den Client übertragen werden oder nicht.

Tab. 4-12: Mögliche Konfigurationsparameter durch den Client

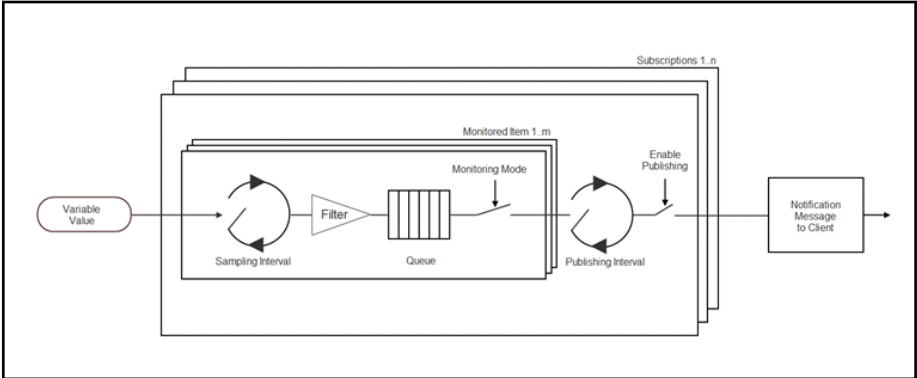


Abb. 4-30: Vereinfachte Darstellung einer Subscription

Der Namensraum der SPS

Der OPC UA Server unterstützt den clientseitigen Zugriff auf Datenpunkte der SPS. Über die Read- und Writeservices können so SPS-Variablen gelesen und geändert werden. Der Dienst Publish ermöglicht einen effizienten Mechanismus zur Benachrichtigung eines Clients über Werteänderung von SPS Variablen. Der Namensraum der SPS kann mittels der Browserservices untersucht werden. Zu beachten ist, dass der Namensraum der SPS auf der Symbolkonfiguration des SPS-Programms beruht. Es werden nur symbolisch exportierte Datenpunkte im Namensraum des OPC UA Servers abgebildet (siehe Symbolkonfiguration der IndraLogic Dokumentation).

Zentraler Eintrittspunkt in den Namensraum der SPS ist der Knoten Logic

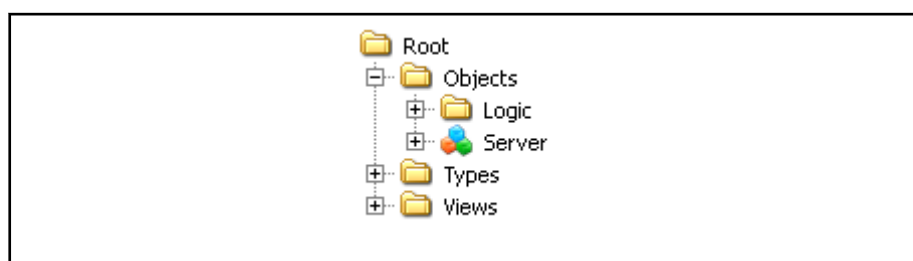


Abb. 4-31: Root-Knoten Logic

Die unter diesem Knoten aufgespannte Struktur folgt der Spezifikation "PLCOpen: OPC UA For IEC 61131-3 1.00 Companion Specification" (erhältlich unter <http://www.opcfoundation.org>). Sie entspricht weitestgehend der Strukturierung der SPS-Applikation.

OPC UA und die PLCOpen verwenden ein unterschiedliches Basisdatentypsystem. Das Typemapping erfolgt anhand folgender Tabelle.

Typemapping

IEC 61131-3	OPC UA "build in" Datentyp	erweiterte OPC UA Datentypen durch "OPC UA For IEC 61131-3 1.00 Companion Specification"	Kommentar
BOOL	Boolean		8 bit Datentyp
SINT	SByte		8 bit Datentyp
USINT	Byte		8 bit Datentyp
INT	Int16		16 bit Datentyp, vorzeichenbehaftet
UINT	UInt16		16 bit Datentyp, vorzeichenlos
DINT	Int32		32 bit Datentyp, vorzeichenbehaftet
UDINT	UInt32		32 bit Datentyp, vorzeichenlos
LINT	Int64		64 bit Datentyp, vorzeichenbehaftet
ULINT	UInt64		64 bit Datentyp, vorzeichenlos
BYTE	Byte	BYTE	BYTE ist ein durch die PLCOpen spezifizierter und von Byte abgeleiteter Datentyp mit der Bedeutung Bit-String der Länge 8.
WORD	UInt16	WORD	WORD ist ein durch die PLCOpen spezifizierter und von UInt16 abgeleiteter Datentyp mit der Bedeutung Bit-String der Länge 16.

Kommunikationsschnittstellen

IEC 61131-3	OPC UA "build in" Datentyp	erweiterte OPC UA Datentypen durch "OPC UA For IEC 61131-3 1.00 Companion Specification"	Kommentar
DWORD	UInt32	DWORD	DWORD ist ein durch die PLCOpen spezifizierter und von UInt32 abgeleiteter Datentyp mit der Bedeutung Bit-String der Länge 32.
LWORD	UInt64	LWORD	LWORD ist ein durch die PLCOpen spezifizierter und von UInt64 abgeleiteter Datentyp mit der Bedeutung Bit-String der Länge 8.
REAL	Float	-	32 bit Floating Point
LREAL	Double	-	64 bit Floating Point
STRING	String	STRING	STRING ist ein durch die PLCOpen spezifizierter und von String abgeleiteter Datentyp. Er stellt einen Ein-Byte Character String variabler Länge dar.
WSTRING	String	-	ein UTF 8 String variabler Länge
DT DATE_AND_TIME	DateTime	-	64 Bit Datentype welcher die Zeit in 100ns Intervallen seit 1. Januar 1601 repräsentiert.
DATE	DateTime	DATE	DATE ist ein von der PLCOpen spezifizierter und von DateTime abgeleiteter 64 Bit Datentyp. Er repräsentiert ein Datum
TOD TIME_OF_DAY	DateTime	TOD	TOD ist ein von der PLCOpen spezifizierter und von DateTime abgeleiteter 64 Bit Datentyp. Er repräsentiert eine Zeit.
TIME	Double	Duration	Duration ist ein von der PLCOpen spezifizierter und von Double abgeleiteter 64 Bit Datentyp. Er repräsentiert ein Zeitintervall in Millisekunden.

Tab. 4-13: Typemapping IEC 61131-3 OPC UA

4.6.4 Kommunikationsbeispiel über OPC UA

Einleitung

Im Folgenden wird gezeigt, wie man mit einfachen Mitteln auf den Bosch Rexroth OPC UA Server zugreifen kann.

Es wird beschrieben, wie eine Verbindung zu parametrieren ist und wie die unterschiedlichen Services des Servers genutzt werden können.

Voraussetzungen

Um dieses Kommunikationsbeispiel selbst nachvollziehen zu können, benötigen Sie den UaExpert der Unified Automation GmbH.

Der UaExpert ist für die Plattformen Win32 und Linux implementiert und kann frei über die Seite <http://www.unified-automation.com> bezogen werden.

Diese Dokumentation basiert auf der Version 1.2.0 132 des UaExpert.

UaExpert

Der UaExpert ist ein universeller OPC UA Test-Client. Mit ihm ist es möglich den Adressraum eines OPC UA Servers zu untersuchen und einfache Tests mit dem OPC UA Server durchzuführen.

Der UaExpert unterstützt

- das Browsen von Adressräumen,
- das Lesen und Schreiben von Nodes (Value Attribut),
- das Beobachten von Wertänderungen und Events (Subscription),
- den Aufruf von Methoden,
- und das Lesen von historischen Daten

Der UaExpert ist in mehrere Fenster unterteilt.

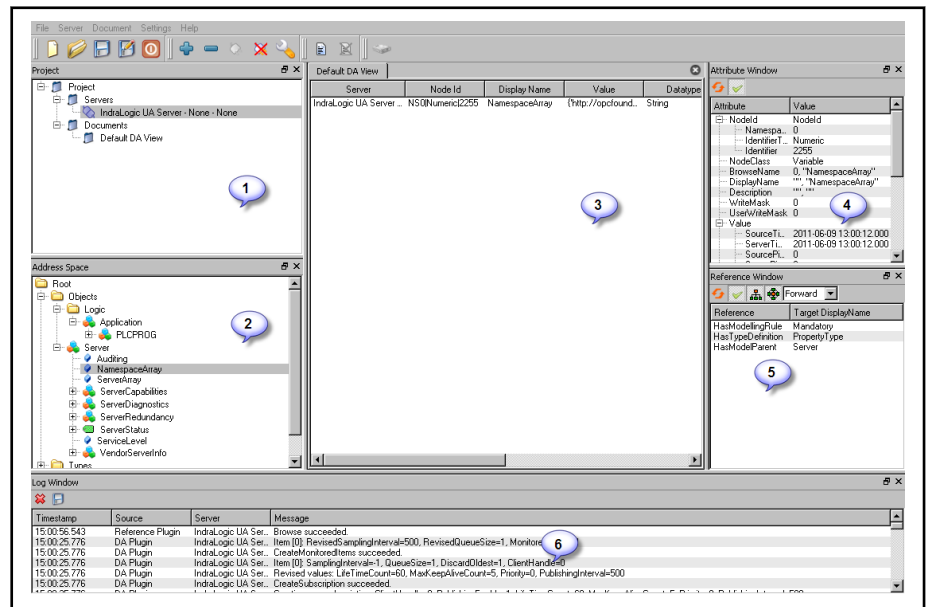


Abb. 4-32: UaExpert

Die Subfenster haben folgende Bedeutung:

- ① Project Window. Hier werden die Serververbindungen und die Dokumente verwaltet.
- ② Address Space Window. Das Address Space Window ermöglicht das Navigieren und Browsen im Adressraum des im Project Window angewählten Servers.
- ③ Data View Window. Das Data View Window dient der Darstellung der Serverdaten.

Es werden folgende Views unterstützt:

– DataAccess View

Der DataAccess View beschreibt den Inhalt einer Subscription. Es werden alle Nodes, die auf Wertänderung beobachtet werden, aufgelistet.

– Alarms&Condition View

Im Alarms&Condition View können von einem Server ausgelöste Events und Alarme konfiguriert und angezeigt werden. Alarme und Events werden in der aktuellen Bosch Rexroth OPC UA Server Version nicht unterstützt.

– HistoricalTrend View

Im HistoricalTrend View kann die zeitliche Wertänderung für eine oder mehrere Nodes in Form eines Diagramms angezeigt bzw. parametrisiert werden.

– Performance View

Kommunikationsschnittstellen

Mit dem Performance View kann ein OPC UA Server hinsichtlich seiner Performance untersucht werden. Es werden die Services Read, Write, CreateMonitoredItem und DeleteMonitoredItem unterstützt.

- ④ Attribute Window

Im Attribute View werden alle Attribute, der im Address Space Window angewählten Node angezeigt.

- ⑤ Reference Window

Das Reference Window zeigt alle Referenzen für die im Address Space Window angewählte Node an. Es wird zwischen Forward Referenzen, Referenzen die von dieser Node aus wegführen, und Inversen Referenzen, die zu dieser Node zeigen, unterschieden.

- ⑥ Log Window

Das Log Window dient der Diagnose von Kommunikationsproblemen.

Verbindungsaufbau

Der Verbindungsaufbau zu einem OPC UA Server erfordert die Angabe eines Endpoints. Der Endpoint besteht aus dem verwendeten Kommunikationsprotokoll, der Hostadresse und den Securityeinstellungen.

Um einen Endpoint für einen OPC UA Server im UaExpert zu definieren, wählen sie im Menü **Server ► Add....** Im erscheinenden Dialog aktivieren Sie den Reiter "Discovery" und wählen in der Baumdarstellung unter "Custom Discovery" den Eintrag "<Double click to Add Server...>".

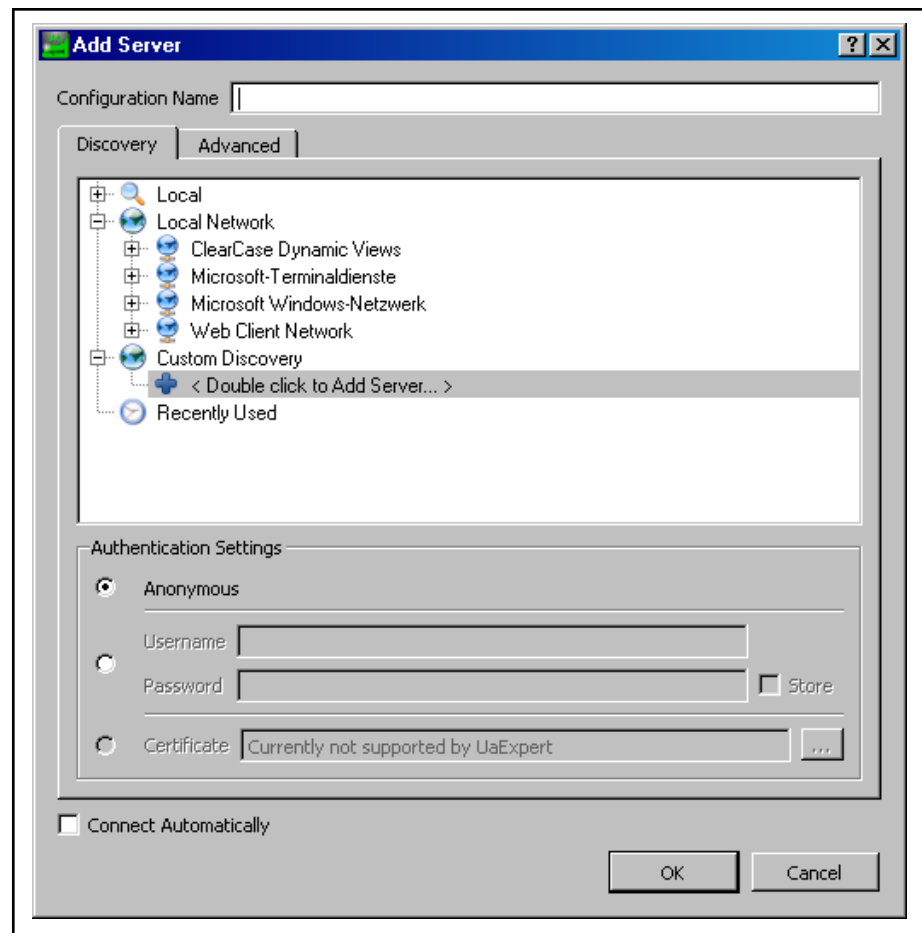


Abb. 4-33: Dialog: Add Server

Kommunikationsschnittstellen

Über einen Doppelclick auf diesen Eintrag gelangen Sie zur Eingabemaske des Kommunikationsprotokolls und der Hostadresse. In der Eingabezeile ist das Kommunikationsprotokoll durch den Eintrag "opc.tcp://" als UA-Binary Protokoll schon vorgelegt. Ergänzen Sie die IP-Adresse oder den Hostnamen für ihren OPC UA Server. Optional kann hier auch die Portnummer, getrennt durch einen Doppelpunkt, mit angegeben werden. Der Bosch Rexroth OPC UA Server verwendet den Port 4840.

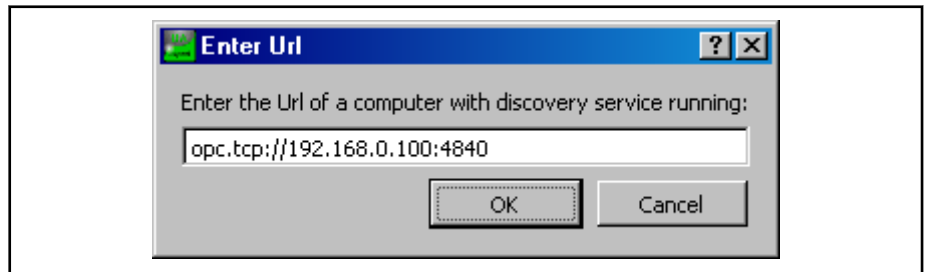


Abb. 4-34: Eingabe der EndpointUrl

Mit **Ok** kommen Sie in den vorherigen Dialog zurück.

Unter dem Eintrag "Custom Discovery" ist nun die EndpointUrl sichtbar.

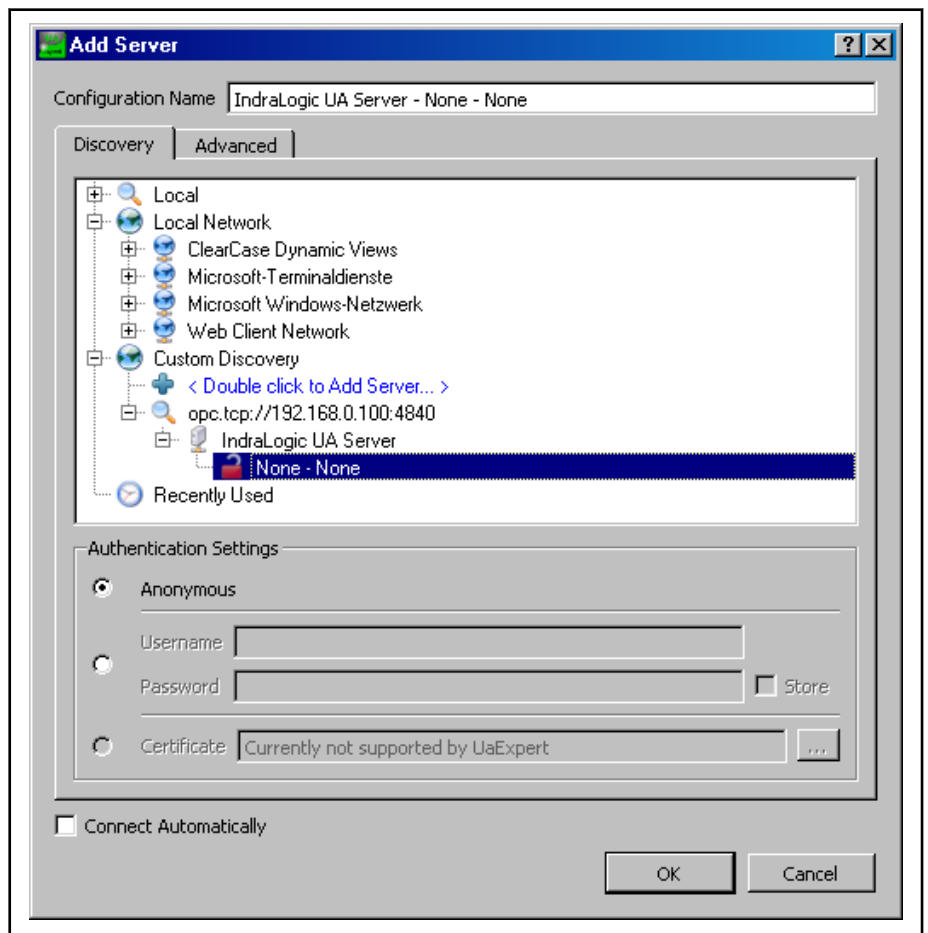


Abb. 4-35: Auswahl "No Security Endpoint"

Soweit Verbindung zum Server vorhanden ist, kann der Eintrag als Baum expandiert werden. Es sollte nun die ServerUrl "IndraLogic UA Server" und unterhalb dieser die Security Endpoints sichtbar sein. Der Bosch Rexroth OPC UA Server unterstützt keine Security. Deshalb ist hier nur der Eintrag "None - None" vorhanden. Klicken Sie nun diesen Eintrag an und schließen Sie den Dialog.

Kommunikationsschnittstellen

Browsen

Im Project Window ist unterhalb des Knotens "Server" die ServerUrl für die neu parametrisierte Verbindung ersichtlich. Über das Kontextmenü dieses Eintrages bzw. über den Menüpunkt "Server-Connect" kann nun eine Verbindung mit diesem OPC UA Server aufgebaut werden.

Nach erfolgreichen Verbindungsaufbau wird der Adressraum des OPC UA Servers im Address Space Window dargestellt.

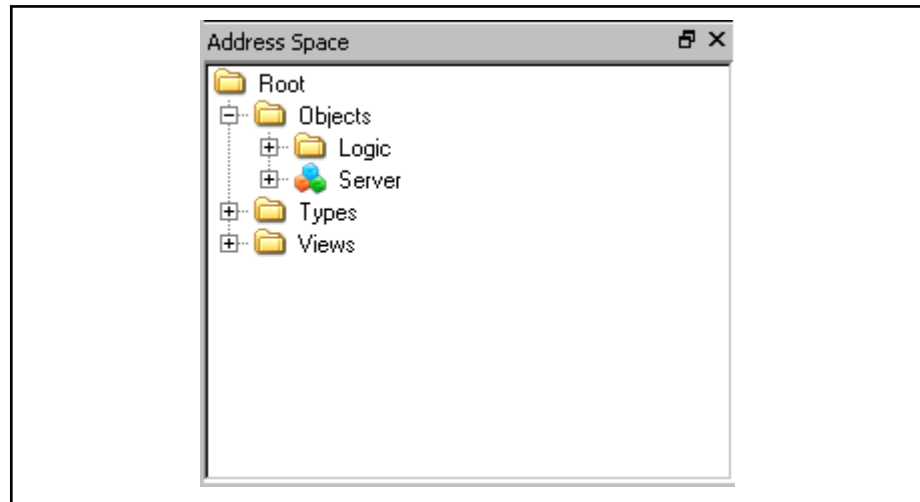


Abb. 4-36: Address Space Window

Die Einträge in der Baumdarstellung sind die OPC UA Nodes. Diese Nodes können nun expandiert werden und ermöglichen so das Navigieren im Adressraum des OPC UA Servers. Es werden alle Nodes dargestellt, die der Adressraum beinhaltet.

Mit Selektion einer Node durch Mausklick werden alle Attribute dieser Node im Attribute Window mit ihren Werten aufgelistet. Die Werte sind in diesem Fenster readonly, d.h. ein Client kann diese Werte nicht ändern.

Im Reference Window können alle von dieser Node ausgehenden Referenzen (Forward) und alle auf diese Node zeigende Referenzen (Inverse) abgelesen werden.

Subscribe und Write

Der UaExpert unterstützt den Mechanismus zur Beobachtung von Datenänderungen (Subscription) im Data View Window. Sie können einen oder mehrere dieser Views über das Menü **Document - Add** und der Auswahl von "Data Access Classic View" in dem "Add Document" - Dialog erzeugen.

Als Montitored Item werden Value Attribute bezeichnet, die durch den Server auf Wertänderung beobachtet werden. Um ein Monitored Item zu erzeugen, müssen Sie die betreffende Node im Address Space Window mittels Drag and Drop in das Data View Window ziehen.

Kommunikationsschnittstellen

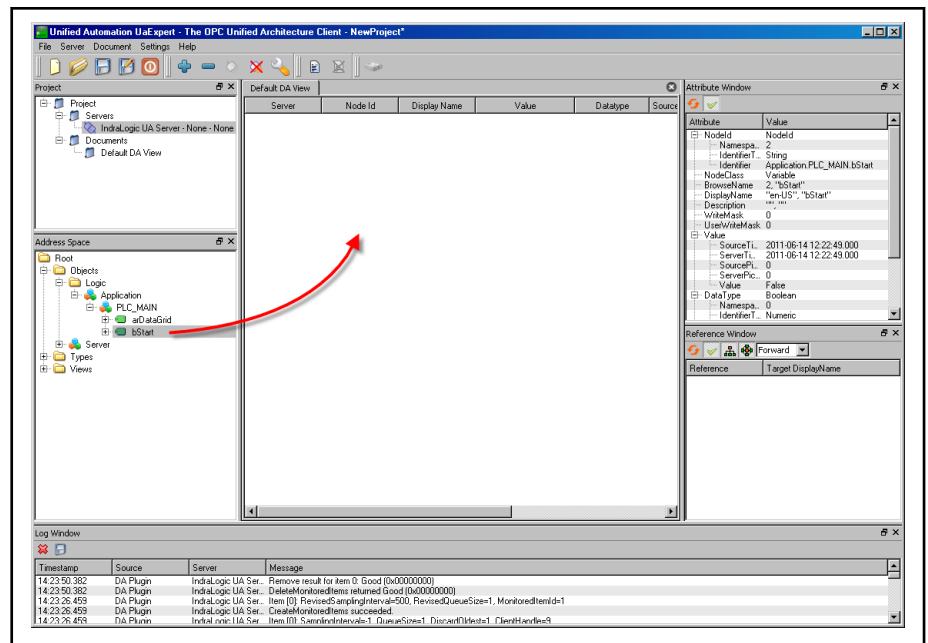


Abb. 4-37: Create Monitored Item

Vom UaExpert wird automatisch das Attribut Value als Monitored Item verwendet. Die Nodes müssen also der Nodeclass Variable angehören.

Im Data View Window wird nun der Wert der Node angezeigt und regelmäßig aktualisiert. Ein "Data Access Classic View" kann als eine Subscription angesehen werden. Der Dialog zur Einstellung der Eigenschaften der Subscription ist über das Kontextmenü dieses Fensters erreichbar. Dazu müssen sie im Data View Window die rechte Maustaste klicken und den Eintrag "Subscription Settings..." wählen.

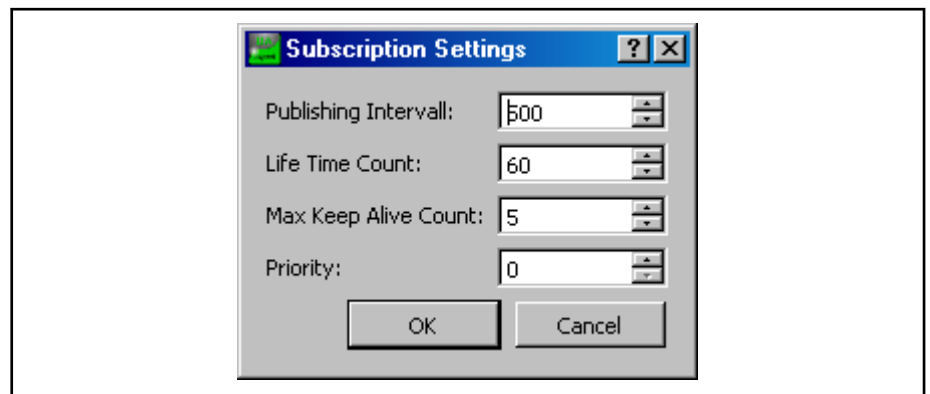


Abb. 4-38: Dialog: Subscription Settings

In dem Dialog können sie folgende Einstellungen vornehmen. Zu beachten ist, dass alle Werte zwischen Client und Server ausgehandelt werden.

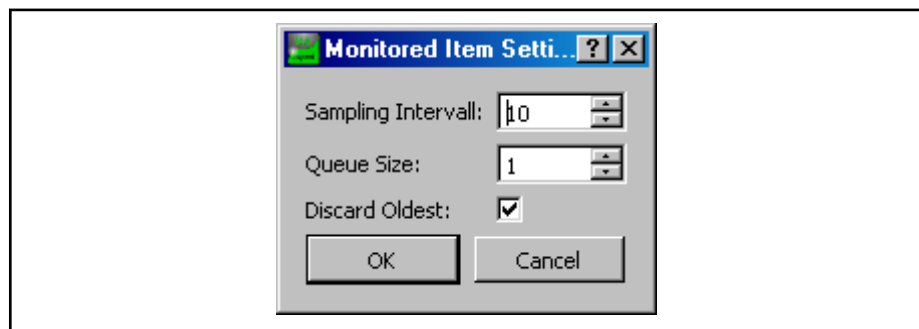
Kommunikationsschnittstellen

Einstellwerte für eine Subscription

Publishing Intervall	Das Publishing Intervall beschreibt das Intervall in Millisekunden, in dem die Werte der enthaltenen Monitored Items bei Änderungen an den Client übertragen werden. Ändern sich die Werte der Monitored Items innerhalb des Publishing Intervalls nicht, so werden auch keine Daten zum Client übertragen. Der Bosch Rexroth OPC UA Server verwendet die festen Publishing Intervalle: 10 ms, 100 ms, 200 ms, 500 ms, 1.0 s, 2.0 s, 5.0 s.
Life Time Count	Der Life Time Count dient der Überwachung der Verbindung zwischen Client und Server. Er gibt an, wie oft das Publishing Intervall ohne Werteübertragung an den Client vergehen kann, bis im Server alle Monitored Items dieser Subscription und die Subscription selbst gelöscht werden. Der Life Time Count muss mindestens 3 mal so groß sein wie Max Keep Alive Count
Max Keep Alive Count	Der Max Keep Alive Count gibt an, wie oft das Publishing Intervall vergehen kann, ohne dass Werteänderungen vom Server übertragen werden. Der Server sendet an den Client eine "leere" Nachricht, anhand derer der Client feststellen kann, dass die Subscription im Server noch aktiv ist.
Priority	Über diesen Wert kann einer Subscription eine Priorität (höherer Wert entspricht höherer Priorität) zugeordnet werden. Sind in einem Server mehrere Subscriptions aktiv, so werden Subscriptions mit höherer Priorität bevorzugt behandelt.

Tab. 4-14: Subscription settings

Weitere Einstellmöglichkeiten betreffen die Monitored Items selbst. Erreichbar sind diese durch einen rechten Mausklick auf das Monitored Item und der Auswahl von "Monitored Items Settings...". Auch hier werden die Werte zwischen Client und Server ausgehandelt.

*Abb. 4-39: Dialog: Monitored Item Settings*

Einstellwerte für Monitored Items

Sampling Intervall	Das Sampling Intervall gibt an, in welchem Intervall der Server die unterlagerte Datenquelle abtasten soll, um einen Wert für das Monitored Item zu ermitteln. Wird der Wert auf -1 gesetzt, so wird der Wert auf das Publishing Intervall gesetzt.
Queue Size	Über diesen Eintrag kann die Größe der Queue für das Monitored Item eingestellt werden. Es werden nur bei Wertänderungen diese Werte in die Queue eingetragen. Alle Werte der Queue werden im Publishing Intervall an den Client übertragen.
Discard Oldest	Discard Oldest beschreibt die Regel, wie der Server bei gefüllter Queue vorgehen soll. Ist der Wert gesetzt, so wird der älteste Wert der Queue mit dem neuen Wert ersetzt. Ansonsten wird der neueste Wert ersetzt. Ist die Queue Size gleich 1, so ist dieser Parameter ohne Bedeutung.

Tab. 4-15: Monitored Items settings

Um einen Wert zu schreiben, müssen Sie die Value Spalte eines Monitor Items per Doppelklick auswählen. Sie können nun den Wert editieren. Nach Bestätigung mit der Enter-Taste wird der neue Wert in den Server geschrieben.

4.7 Konfigurieren einer OPC UA Client-Anbindung mit WinStudio

Zur Konfiguration einer Verbindung zu einem OPC UA Server werden in WinStudio entsprechende Treiberblätter verwendet. Treiberblätter enthalten im wesentlichen die zu kommunizierenden Variablen und Kommunikationsparameter. Sie können für jeden in WinStudio konfigurierten Server mehrere Treiberblätter anlegen. Hierbei können Sie für jedes angelegte Treiberblatt die Kommunikationsparameter (Aktualisierungsraten, zu kommunizierende Variablen, etc.) individuell festlegen. Jedes Treiberblatt stellt eine Subscription dar.

Zum Projektieren einer OPC UA Verbindung zu einem OPC UA Client sind 2 Stufen notwendig:

1. Es muss zunächst die OPC-UA Verbindung konfiguriert werden
2. Anschliessend muss ein Treiberblatt konfiguriert werden. Hierbei können, wie bereits erwähnt, auch mehrere Treiberblätter angelegt werden.

A) Konfigurieren einer OPC-UA-Verbindung

Um mit einem Gerät mittels des OPC-UA-Servers zu kommunizieren muss die Verbindung zu diesem Server zunächst konfiguriert werden. Gehen Sie dazu wie folgt vor:

1. Öffnen Sie das Kontextmenü des Ordners "OPC-UA" und wählen Sie den Eintrag "OPC-UA-Verbindungen konfigurieren".

Es erscheint ein Dialog, der alle bisher erstellten Konfigurationen anzeigt. Um eine neue Verbindung zu einem OPC-UA-Server zu erstellen, drücken Sie die Schaltfläche "Hinzufügen".

Der Dialog "UA-Server-Konfiguration" wird angezeigt.

Kommunikationsschnittstellen

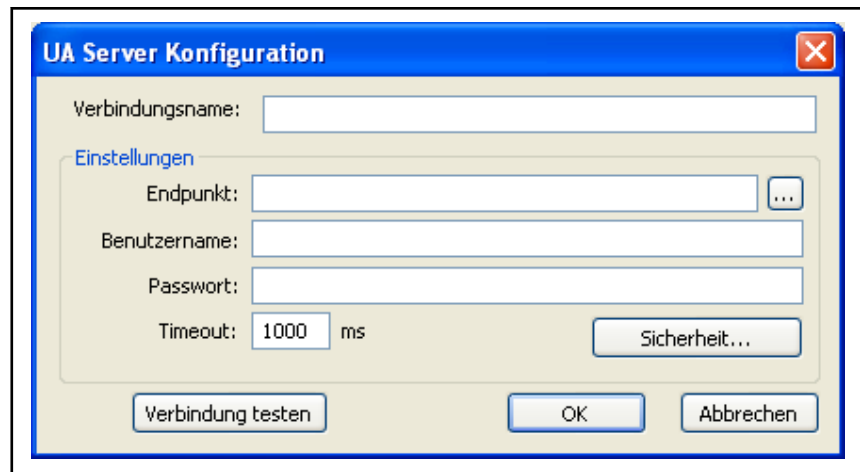


Abb. 4-40: Dialog UA-Server-Konfiguration

2. Tragen Sie im Feld **Verbindungsname** einen Namen für die Verbindung ein.
3. Geben Sie im Feld **Endpoint** die URL des OPC-UA-Servers ein, mit dem Sie sich verbinden möchten. Hierzu wählen Sie über die Schaltfläche ... den UA Discovery Server Dialog aus.

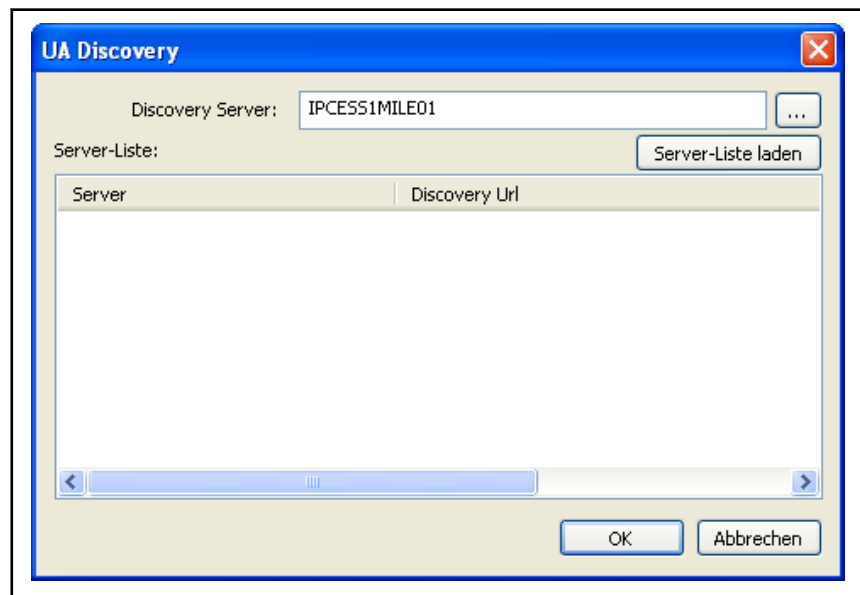


Abb. 4-41: Dialog UA Discovery

Verwenden Sie diesen Dialog zum Auffinden des sog. "Discovery-Servers", der eine Liste mit OPC-UA-Servern im Netzwerk veröffentlicht und wählen Sie dann den Zielserver für Ihre Verbindung aus. Zur Verbindung mit einer Rexroth-Steuerung tragen Sie hier die IP-Adresse der Steuerung ein. Mit der Schaltfläche **Server-Liste laden** aktualisieren Sie nun die Serverliste. Jetzt können Sie die Steuerung aus der Serverliste auswählen und mit **OK** bestätigen.

B) Konfigurieren eines OPC-UA-Treiberblatts

Um ein neues Treiberblatt für den OPC-UA-Client zu erstellen, gehen Sie folgendermaßen vor:

1. Öffnen Sie das Kontextmenü des Knotens **WinStudio** ► **Variablen** ► **Kommunikation** ► **OPC-UA** und wählen Sie "Neues Treiberblatt"

Kommunikationsschnittstellen

Ein neues Element wird im Ordner OPC-UA hinzugefügt. Öffnen Sie nun mit einem Doppelklick den Treiberblatteditor:

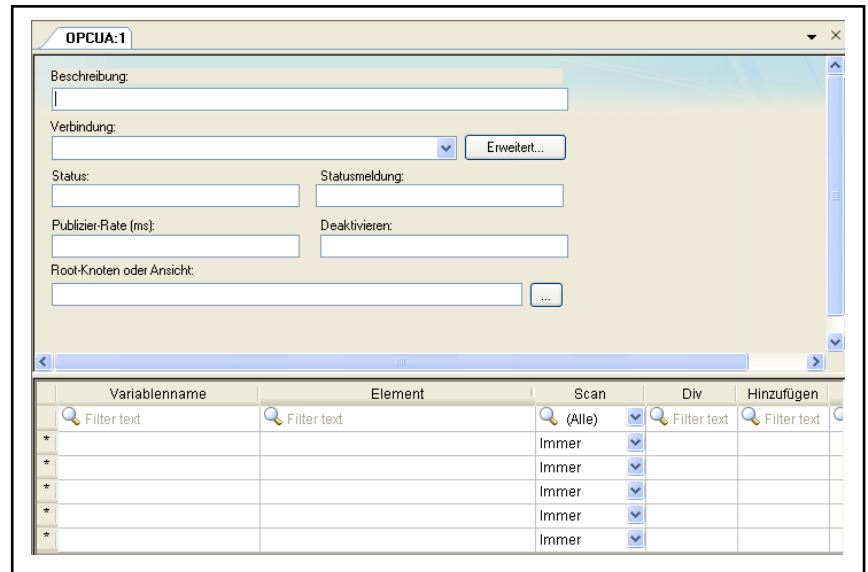


Abb. 4-42: Arbeitsblatt OPC UA Client

2. Tragen Sie im Feld **Beschreibung** zu Dokumentationszwecken eine Beschreibung für das Arbeitsblatt ein. Dies dient nur zu Dokumentationszwecken
3. Wählen Sie aus der Liste **Verbindung** die in A) konfigurierte Serververbindung und speichern Sie das Treiberblatt.
4. Stellen Sie die **Publizier-Rate** ein. In diesem Zeitintervall werden die Variablen des Treiberblattes zur Laufzeit aktualisiert
5. Nun können Sie die entsprechenden Variablen, die kommuniziert werden sollen, auswählen.

Hier zu stehen ihnen zwei Möglichkeiten zur Auswahl

- Mit Hilfe des WinStudio OPC UA Browsers : Hierbei muss eine Verbindung zur Steuerung vorhanden sein

Führen Sie einen Rechtsklick auf das Feld **Element** aus und wählen Sie die Option **OPC-Browser**. Es werden die Steuerungen aus der jeweiligen OPC Konfiguration dargestellt. Die in der Symbolkonfiguration der jeweiligen Steuerung vorhandenen Variablen werden in einer Baumstruktur dargestellt.

- Mit Hilfe des IndraWorks Werkzeugfensters für Prozessvariablen: Hier ist keine Verbindung zur Steuerung notwendig

Aktivieren Sie das Werkzeugfenster über **Ansicht ► Andere Fenster ► Prozessvariablen** in der IndraWorks-Werkzeugleiste.

Wurde die Konfiguration des OPC-Treiberblattes gespeichert, so erscheinen hier die Symbolvariablen der im Projekt vorhandenen Steuerungen. Diese können Sie bequem mittels Drag&Drop in die entsprechende Treiberblattzeile ziehen. Falls sich die Steuerung nicht im Projekt befindet, können Sie die Symbolkonfigurationsdatei der externen Steuerung laden und somit auch diese in ihrem HMI-Projekt projektieren. Weitere Informationen über das Werkzeugfenster für Prozessvariablen finden Sie in der IndraWorks Online-Hilfe

Kommunikationsschnittstellen

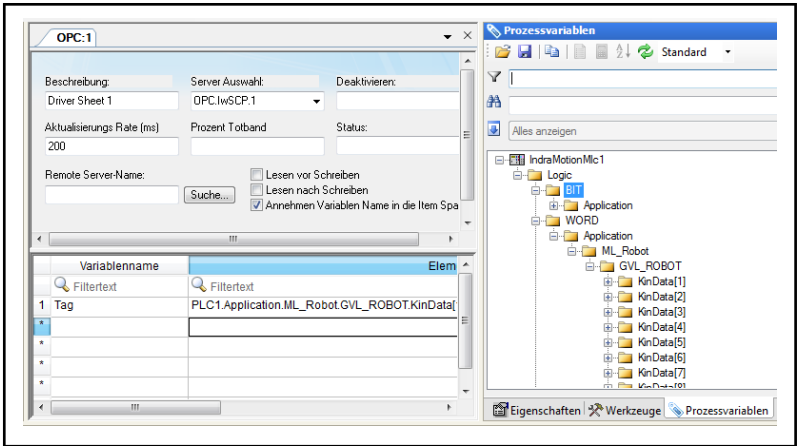


Abb. 4-43: OPC Treiberblattkonfiguration und Werkzeugfenster für Prozessvariablen

Weitere Verbindungsoptionen, sowie eine detaillierte Beschreibung der erweiterten Einstellungen entnehmen Sie der IndraWorks Online-Hilfe "Rexroth WinStuido 7.3 Technische Referenz"

4.8 Modbus TCP
4.8.1 Einführung und Übersicht

Modbus TCP dient dem Datenaustausch zwischen Ethernet-Geräten, die das Modbus-Protokoll unterstützen. Es eignet sich insbesondere dazu, mehrere HMI-Geräte mit einer Steuerung zu verbinden. Die Steuerung bildet hierbei den Modbus-Slave (Server), auf den sich die Modbus-Master (Clients) verbinden.

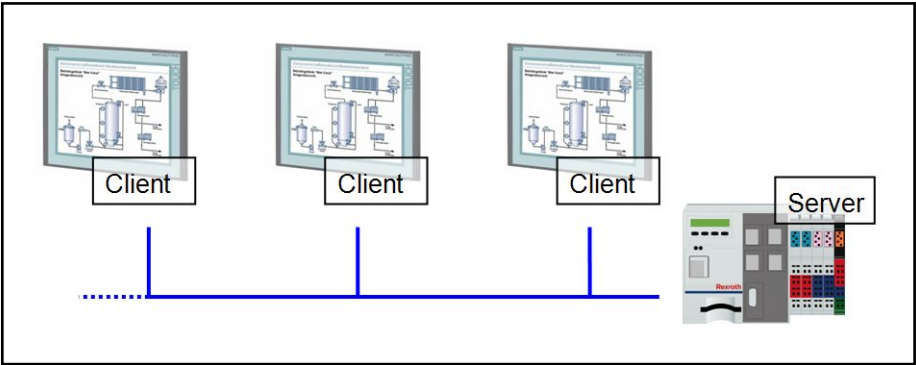


Abb. 4-44: Verbindung mehrerer HMI-Geräte (Modbus-Clients) an eine Steuerung (Modbus-Server)

Modbus TCP unterscheidet vier verschiedene Datentypen:

- Discrete Input: bitadressierter binärer Eingang (array of bool)
- Coil: bitadressierter binärer Ausgang (array of bool)
- Input Register: Word-basierter Ausgang (array of word)
- Holding Register: Word-basierter Ausgang (array of word)

Übertragungsmedium	Ethernet
Protokoll	Modbus TCP auf TCP-Port 502
max Anz. HMI-Geräte	16 pro Modbus TCP-Server

Kommunikationsschnittstellen

Übertragungsleistung	1000 I/O-Operationen pro Sekunde bei 1ms Zykluszeit
I/O-Bereich	65536 Bit Discrete Input 65536 Bit Coil 65536 Word Input Register 65536 Word Holding Register

Tab. 4-16: *Eigenschaften von Modbus TCP*

Die Steuerungen IndraMotion MLC und IndraLogic XLC unterstützen die folgenden Modbus TCP-fähigen HMI-Geräte:

- Siemens WinCC-flexible 2007
- Pro-face GP-Pro EX2.2

Geräte anderer Hersteller werden unter Umständen unterstützt, wurden jedoch nicht getestet.

Ein Modbus TCP-Server ist jeweils in den Funktionsbausteinen IL_ModBusTCPServer und IL_ModBusTCPServerType02 implementiert, die in der RIL_ModbusTCP.library enthalten sind.

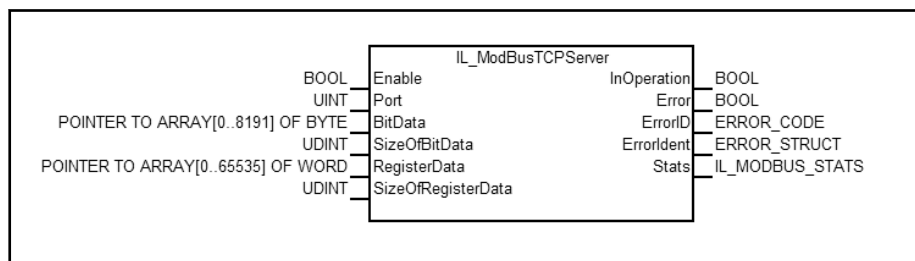


Abb. 4-45: *Funktionsbaustein IL_ModBusTCPServer*

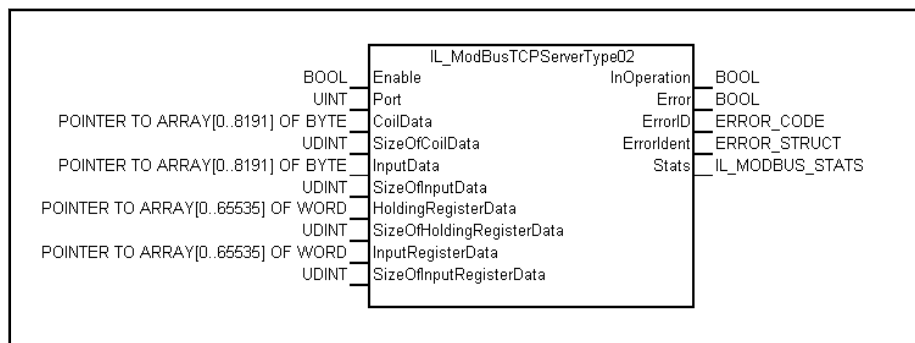


Abb. 4-46: *Funktionsbaustein IL_ModBusTCPServerType02*



Die beiden Funktionsbausteine IL_ModBusTCPServer und IL_ModBusTCPServerType02 unterscheiden sich wie folgt:

Entgegen der Modbus-Spezifikation sind beim Funktionsbaustein "IL_ModBusTCPServer" die Datenbereiche für "Coil Data" und "Discrete Input Data" unter "BitData", sowie "Holding Register" und "Input Register" unter "RegisterData" zusammengefasst.

Eine genaue Beschreibung der Modbus TCP-Funktionalität finden Sie in der Dokumentation

- Rexroth IndraWorks xxVRS Basis-Bibliotheken IndraLogic 2G

xx = Version

5 Applikationsbeispiele

5.1 Übersicht

Dieses Kapitel beschreibt einfache Beispielapplikationen für die einzelnen Visualisierungsarten. Die Beispiele sind bewusst knapp gehalten. Für tiefer gehende Informationen sollten die entsprechenden Dokumentationen der Visualisierungssysteme heran gezogen werden.

5.2 Symbolkonfiguration

5.2.1 Allgemein

Damit ein OPC Client auf die SPS-Variablen einer Steuerung zugreifen kann, müssen die vom OPC Client benötigten bzw. die vom SPS-Projekt für den Zugriff bereitgestellten Variablen in der Symbolkonfiguration der IndraLogic konfiguriert sein.

Die Variablen können mit den folgenden Zugriffsberechtigungen konfiguriert werden:

- Kein Zugriff
- Lesend
- Schreibend
- Lesend und Schreibend

Die Symbolkonfiguration kann durch einen Wizard über das Fenster "Symbolkonfiguration" oder über "Pragma Direktiven" im Variablen Deklarationsbereich der POE bzw. über Variablen Listen erfolgen.

Voraussetzung für die Symbolkonfiguration ist das Vorhandensein des SPS-Objekts "Symbolkonfiguration" unterhalb der SPS-Applikation im IndraWorks Projektbaum. Hierzu wird das in der folgenden Tabelle gelistete Objekt aus dem Bibliotheksfenster "SPS Objekte / Allgemeine Objekte" unterhalb des Geräteknotens "Logic / Application / Hmi" per Drag&Drop hinzugefügt. Die nachfolgende Abbildung zeigt den Projektexplorer nach dem Hinzufügen des Objekts.

Objekt-Typ	Name	Beschreibung
Symbolkonfiguration	"_"	Konfiguration der Variablen für den Zugriff über den OPC Server mit Zugriffsrechten.

Tab. 5-1: SPS-Bibliotheksojekt "Symbolkonfiguration" hinzufügen

Applikationsbeispiele

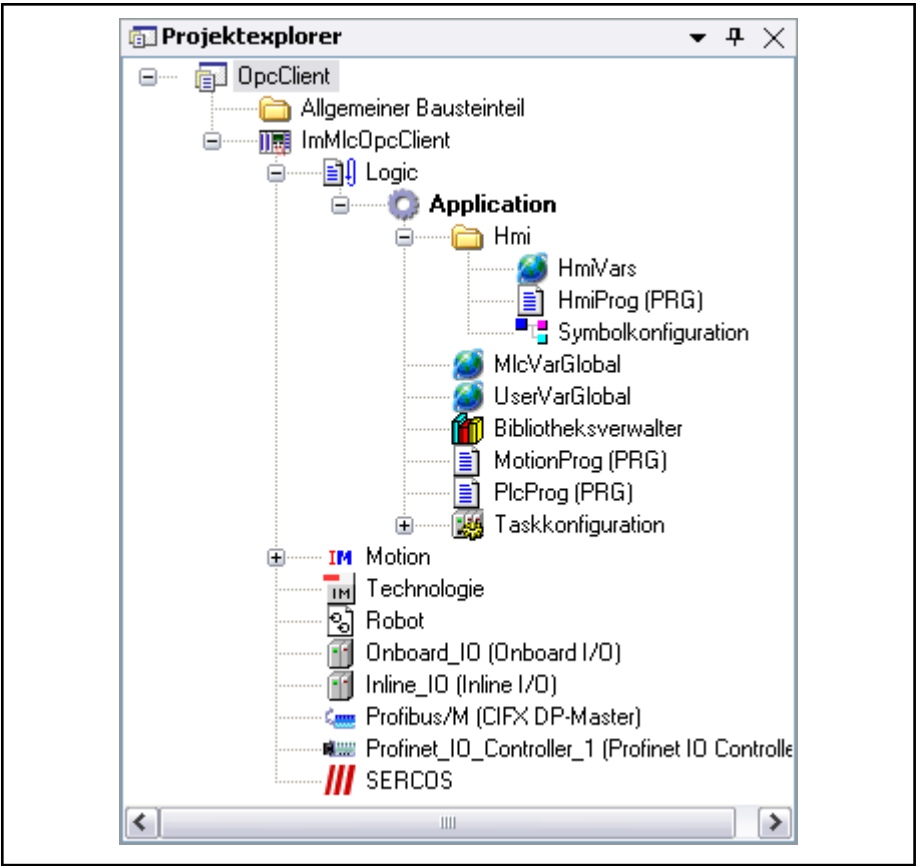


Abb. 5-1: *Hinzugefügtes Bibliotheksobjekt "Symbolkonfiguration"*

5.2.2 **Symbolkonfiguration mit Wizard**

Der Wizard für die Symbolkonfiguration wird über den neu erstellten Knoten "Symbolkonfiguration" im IndraWorks Projektexplorer geöffnet.

Das Fenster der Symbolkonfiguration ist in zwei Bereiche gegliedert:

Verfügbare Variablen	Auflistung aller Variablen, die im SPS-Programm vorhanden sind, mit der jeweiligen maximalen Zugriffsberechtigungsstufe. Werden keine Variablen aufgelistet, wird über den Link "Aktualisieren" eine Aktualisierung ausgelöst. Hierbei ist zu beachten, dass das Projekt ohne Fehler übersetzbar sein muss.
Ausgewählte Variablen	Auflistung aller Variablen, die für den Zugriff über den OPC Server konfiguriert wurden, mit der jeweiligen konfigurierten Zugriffsberechtigung. Mit Betätigen des Links "Ungültige Variablen anzeigen" werden alle Variablen, die nicht mehr im SPS-Projekt vorhanden sind, hervorgehoben.

Tab. 5-2: *Aufbau des Wizards für die "Symbolkonfiguration"*

Die Auswahl bzw. das Löschen von Variablen erfolgt per Doppelklick auf das Element oder über die Schaltflächen ">" und "<". Hierbei wird das selektierte Element und alle untergeordneten Elemente ausgewählt bzw. entfernt.

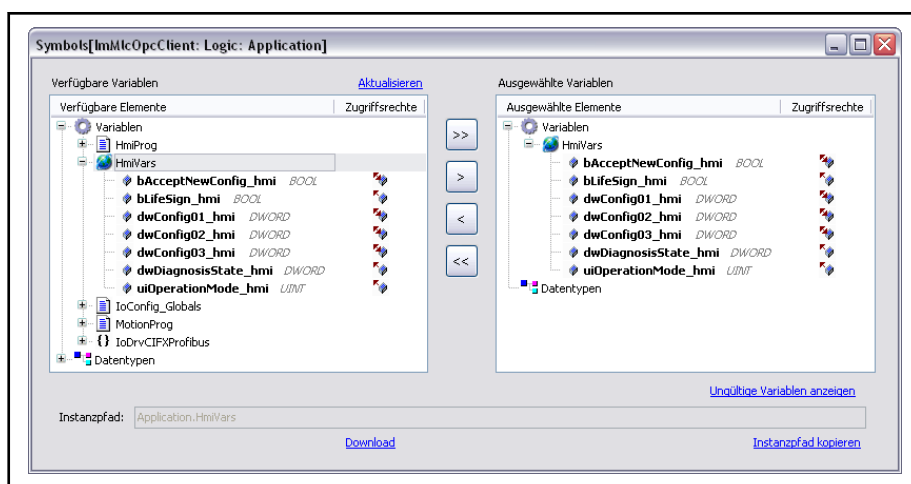


Abb. 5-2: Wizard Symbolkonfiguration



Im Fenster "Symbolkonfiguration" werden nur Variablen aufgelistet, die im SPS-Programm (POE) verwendet werden, wenn dieses in einer Task aufgerufen wird. D.h. Variablen die nicht verwendet werden, sind nicht im Projektbaum vorhanden.

Die Zugriffsrechte für die jeweilige Variable kann über einen Mausklick auf das Symbol in der Spalte "Zugriffsrechte" im Bereich "Ausgewählte Variablen" geändert werden. Das Zugriffsrecht wird in der folgenden Reihenfolge weitergeschaltet:

-  Lese und Schreibzugriff
-  Schreibzugriff
-  Lesezugriff

5.2.3 Symbolkonfiguration über Pragma

Die Symbolkonfiguration über "Pragma Direktiven" erfolgt direkt im Deklarationsbereich der Variablen, z. B. in Variablen Listen oder in POEs.

Für die Pragma Konfiguration steht das Attribut {attribute 'symbol'} mit den Parametern für die Zugriffsberechtigung zur Verfügung. Die möglichen Konfigurationen sind in der folgenden Abbildung dargestellt.

Pragma-Direktive	Beschreibung
{attribute 'symbol' := 'none'}	Variablen sind für den Zugriff gesperrt und werden nicht in der Browsing-Liste des OPC Servers angezeigt.
{attribute 'symbol' := 'read'}	Variablen sind nur für den lesenden Zugriff freigegeben und werden in der Browsing-Liste des OPC Servers angezeigt.
{attribute 'symbol' := 'write'}	Variablen sind nur für den schreibenden Zugriff freigegeben und werden in der Browsing-Liste des OPC Servers angezeigt.
{attribute 'symbol'} {attribute 'symbol' := 'readwrite'}	Variablen sind für den lesenden und schreibenden Zugriff freigegeben und werden in der Browsing-Liste des OPC Servers angezeigt.

Tab. 5-3: Pragma Direktiven für die "Symbolkonfiguration"

Applikationsbeispiele

In Abhängigkeit der Position bzw. der Zeile in der das Pragma geschrieben wird, hat es für alle Variablen im Deklarationsbereich oder nur für eine Variable Gültigkeit.

- Damit das Pragma nur auf eine einzelne Variable wirkt, muss es in der Zeile vor der Variable geschrieben werden.
- Soll es auf alle Variablen im Deklarationsbereich wirken, muss es in der ersten Zeile der Deklaration stehen. In diesem Fall können noch einzelne Variablen mit einer anderen Zugriffsberechtigung versehen werden.

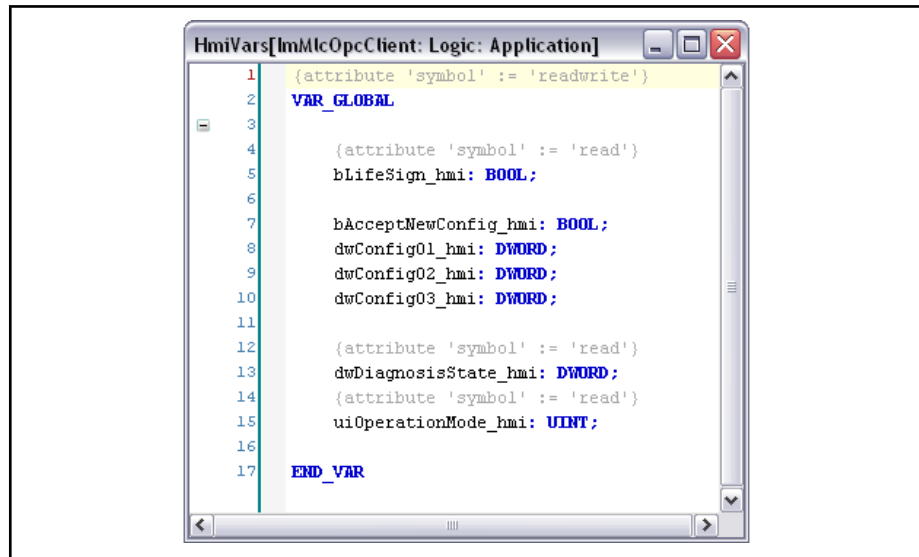


Abb. 5-3: HMI Variablen-Deklaration mit Pragma



Erfolgt die Symbolkonfiguration über Pragma Direktiven, werden diese Symbole **nicht** im Symbolkonfigurationsdialog angezeigt.

Mit dem Download der SPS-Applikation wird die Symbolinformation als XML-Datei im IndraWorks Projektunterverzeichnis "IndraLogic" erstellt und als Symbolapplikation in die Steuerung geladen.

5.3 VI-Composer

In Vorbereitung.

5.4 WinStudio mit IndraLogic XLC

5.4.1 Allgemein

Das Ziel ist es, eine WinStudio-Oberfläche zu erstellen, mit der ein Antrieb gesteuert und seine Statusinformationen angezeigt werden können.

Dieses Projekt ist ein Einstieg in die Anwendung von "WinStudio" in Verbindung mit der IndraLogic XLC bzw. IndraMotion MLC. Die Erstellung einer Oberfläche mit "WinStudio" wird Schritt für Schritt durchgeführt.

Teilaufgaben

Es ergeben sich folgende Aufgaben, die entsprechend dieser Kurzbeschreibung gelöst werden:

- Erstellen eines IndraWorks Projektes, in dem ein Antrieb betrieben wird,
- Einbindung von WinStudio in IndraWorks,

- Erstellen der WinStudio-Anwendung zur Visualisierung der Daten aus dem Steuerungsprojekt (Bedienfeld auf der Steuerung).
- Starten der Oberfläche.

5.4.2 Technische Voraussetzungen

Voraussetzungen

- PC mit installiertem IndraWorks ab Version 13,
- IndraLogic XLC oder IndraMotion MLC als Steuerung mit sercos Interface,
- IndraDrive C (MPx-07VRS) als Antrieb,
- WinStudio V 7.3 unter IndraWorks,
- IndraWorks Engineering 13Vxx.

Die Schaffung dieser Voraussetzungen sind beschrieben in

- Rexroth IndraLogic XLC 13VRS-Funktionsbeschreibung, DOK-IM*ML*-SYSTEM**V13-RExx-DE-P und
- IndraWorks Online-Hilfe, **System IndraLogic XLC / IndraMotion MLC ▶ XLC/MLC-Funktionsbeschreibung ▶ Inbetriebnahme und Bedienung**,

HMI, Typ VEP xx.4/5 oder VPP xx.3

- die aktuelle Firmware ist geladen,
- die IP-Adresse ist bekannt,
- die Steuerung und die HMI ist über Ethernet mit dem Programmiergerät (PC) verbunden.

Weitere Voraussetzungen:

- Ein gebrückter sercos Bus, wenn kein Antrieb vorhanden ist. Verwenden Sie in diesem Fall eine virtuelle Achse anstelle der realen Achse.

5.4.3 Erstellen des IndraWorks-Projekts

Starten Sie die Engineering-Umgebung von IndraWorks mittels Icon auf dem Desktop oder der entsprechenden Auswahl im Programmordner von Windows.

Um ein neues IndraWorks-Projekt zu erstellen, wählen Sie

Datei ▶ Neu ▶ Projekt....

Es erscheint der Dialog "Neues IndraWorks Projekt erstellen".

Geben Sie hier den Namen „WinStudio“ (dies ist nur ein Vorschlag, natürlich können Sie jeden beliebigen Namen, der von IndraWorks akzeptiert wird, verwenden) für das Projekt an und wählen Sie das Verzeichnis, in dem das Projekt abgelegt werden soll.

Fügen Sie die Steuerung „IndraLogic XLC Lxx“ in das Projekt ein, indem Sie das Gerät aus der Bibliothek in das Projekt ziehen. Beachten Sie dabei, dass Sie das Gerät nur direkt auf den Projektordner ziehen können.

Applikationsbeispiele

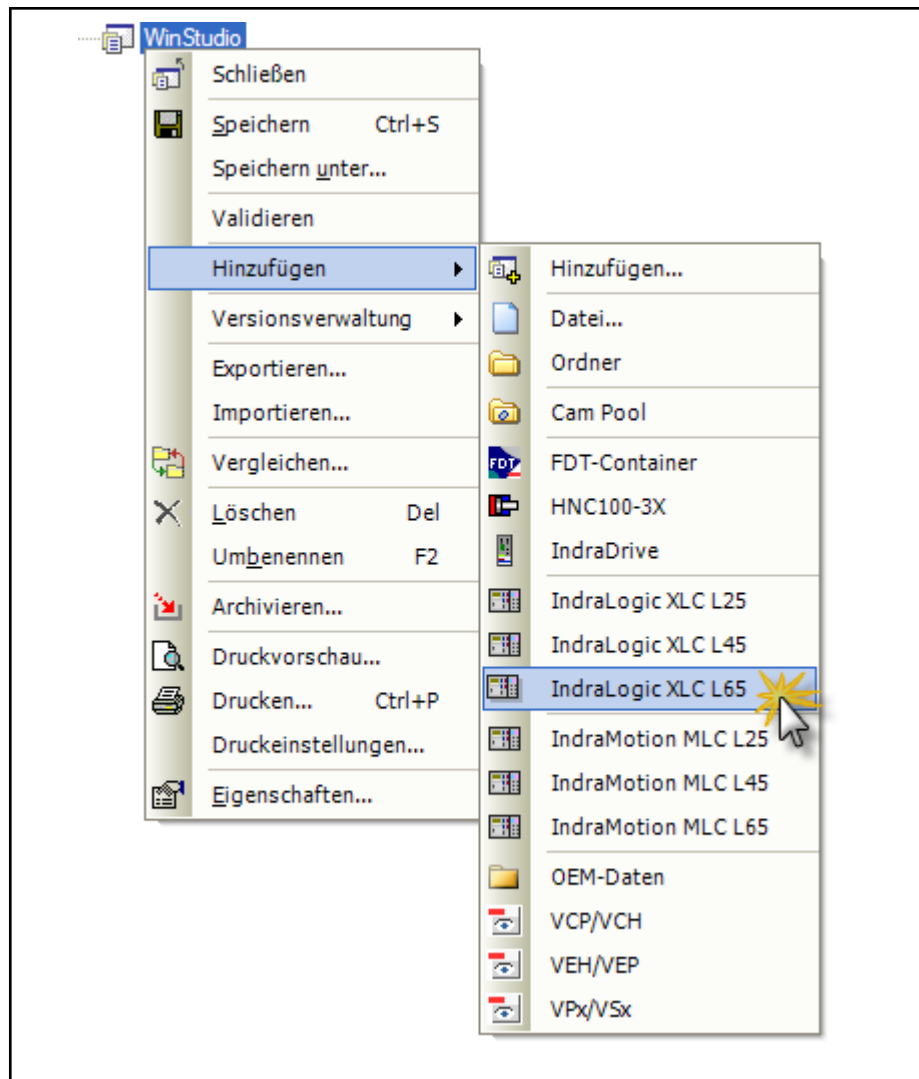


Abb. 5-4: Auswahl der Steuerung IndraLogic XLC L65

Sie können nun einen Gerätenamen Ihrer Wahl vergeben (IndraWorks meldet, falls der Name Zeichen enthält, die nicht erlaubt sind) oder den vorgegebenen verwenden.

Klicken Sie auf "Weiter", um die Kommunikationseinstellungen vorzunehmen und somit die Verbindung mit der Steuerung aufzunehmen.

Die Einstellungen unter den Punkten Firmware-Version und Firmware-Release müssen der Steuerung angepasst werden.

Bei Gerätetyp muss die **Option "CML65.1-3P"** ausgewählt werden.

In das Feld **IP-Adresse** tragen Sie die IP-Adresse Ihrer Steuerung ein. Die IP-Adresse der Steuerung sehen Sie auf der IndraLogic XLC VPP unter "Shortcut to RxSettings".

Ist die Option "Sicherer Online-Modus" aktiviert, erhalten Sie vor sicherheitsrelevanten Aktivitäten eine zusätzliche Abfrage.

Danach sollte ein **Verbindungstest** zur Steuerung durchgeführt werden.

Ist der Verbindungstest positiv verlaufen, ermitteln Sie, welche Antriebe real an Ihrer Steuerung laufen.

Öffnen Sie dazu das Kontextmenü des Ordners "SERCOS", **Buskonfiguration scannen**.

Applikationsbeispiele

Im nachstehenden Bild wurde der "Antrieb1" erkannt.

Durch Doppelklick auf das Symbol in der Spalte IPO (1) wurde entschieden, dass die Bearbeitung der Motion-Funktionalität in der Steuerung erfolgt.



Interpolation in Steuerung  oder im Antrieb  :

- Umschalten durch Doppelklick auf das Icon
- aus dem IndraDrive (Interpolation im Antrieb) wird ein SercosDrive (Interpolation in der Steuerung)

Sein Name wurde in "RA1" geändert (2).

"Antriebe anlegen" (3) platziert den Antrieb im Projektexplorer.

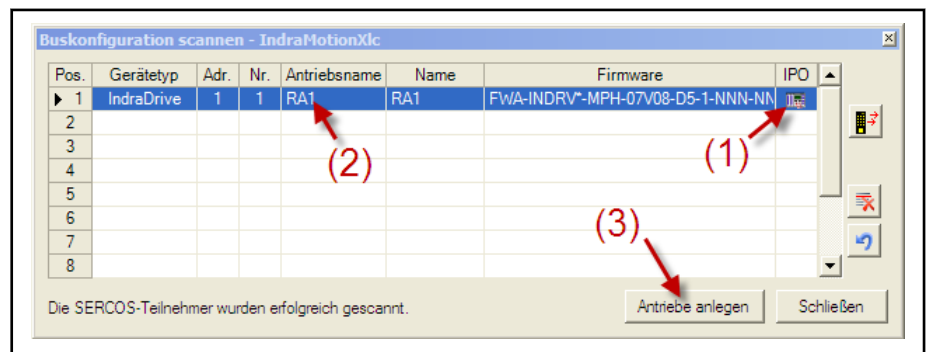


Abb. 5-5: Ergebnis des Antriebs-Scannens

Sind Steuerung und Antrieb korrekt in das Projekt eingefügt worden, sollte Sie unten stehende Ansicht in IndraWorks bekommen.

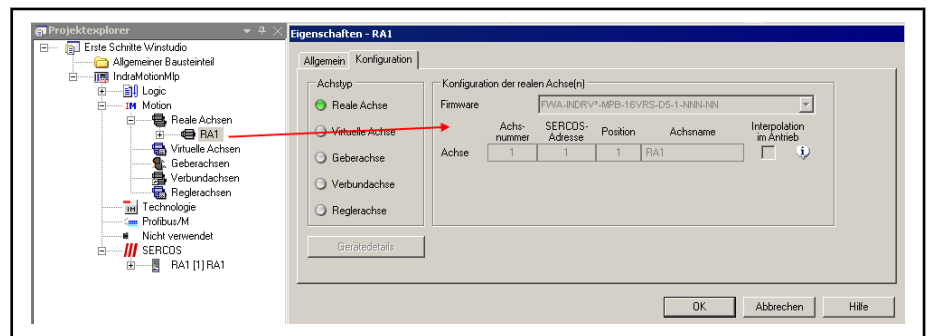


Abb. 5-6: IndraWorks – Steuerung und Antrieb korrekt projektiert

Erstellen Sie unter **Logic ► Application ► MotionProg (PRG)** ein Steuerungsprogramm in strukturiertem Text. Mit einem Doppelklick auf die Datei "MotionProg" öffnet sich automatisch IndraLogic im Mittelteil von IndraWorks.

Applikationsbeispiele

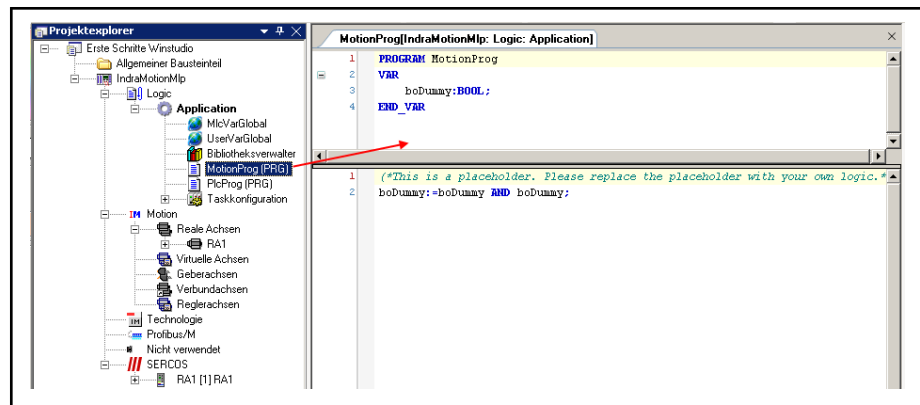


Abb. 5-7: IndraLogic – MotionProg

Den in der Datei vordefinierten Code löschen Sie aus dem Editor.

Fügen Sie die Bausteine MC_Power, MC_MoveVelocity, MC_Stop und MC_Reset in das Programm ein.

Beginnen Sie mit MC_Power. Verwenden Sie die F2-Eingabehilfe.

Geben Sie im Deklarationsteil den Instanznamen (PowerRA1) ein und wählen danach in der F2-Eingabehilfe den Typ (**Benutzerdefinierte Typen** ► **ML_PLCopen** ► **POUs** ► **Motion** ► **MC_Power**) aus.

Schalten Sie die Option "mit Argumenten einfügen" aus.

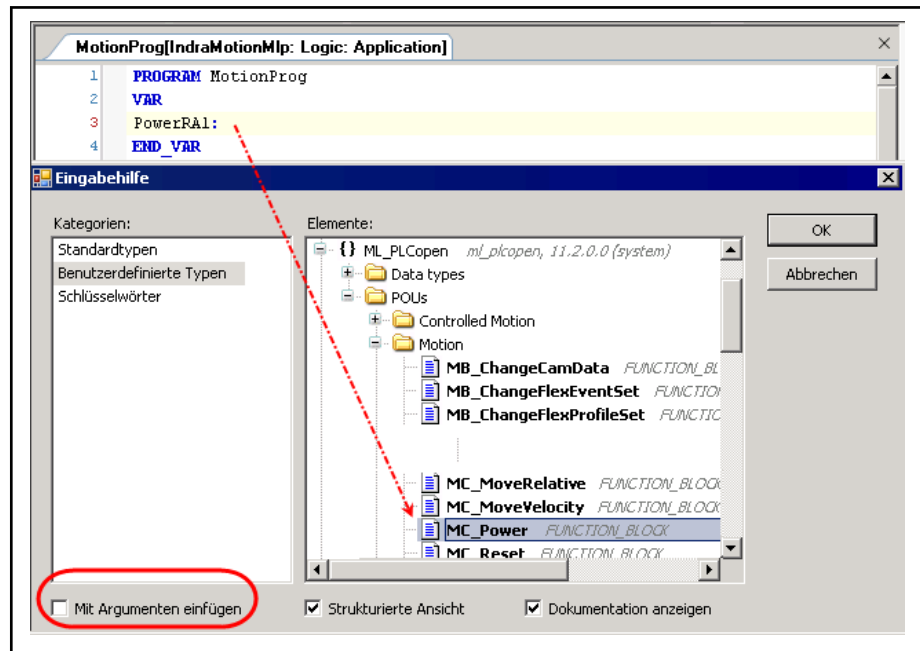


Abb. 5-8: F2-Eingabehilfe, Deklaration, Auswahl des FB-Typs (ohne Argument)

Komplettern Sie die Deklarationszeile mit ";".

Gehen Sie nun in den Implementationsteil und öffnen erneut die F2-Eingabehilfe (**Instanzaufrufe** ► **PowerRA1**).

Schalten Sie die Option "mit Argumenten einfügen" ein.

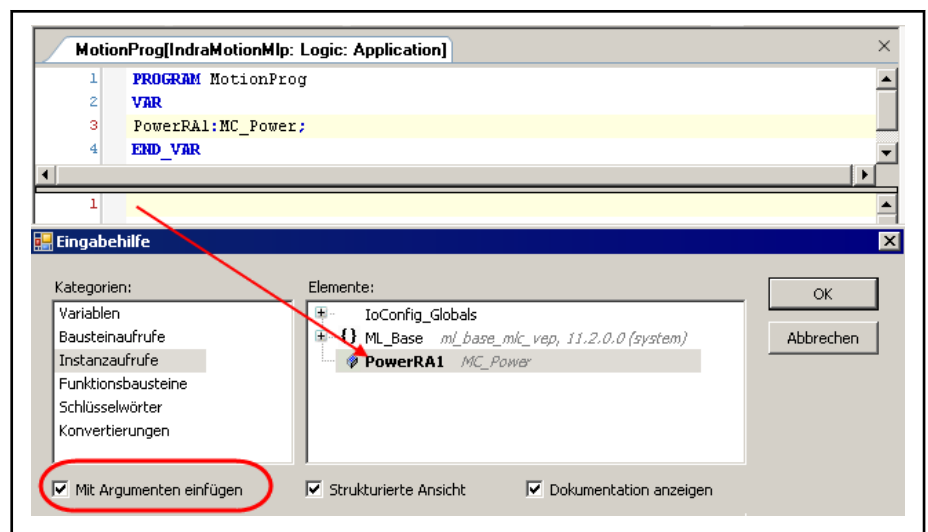


Abb. 5-9: F2-Eingabehilfe, Implementation, Auswahl der FB-Instanz (mit Argumenten)

Sie erhalten das Template des Bausteins zur Beschaltung.

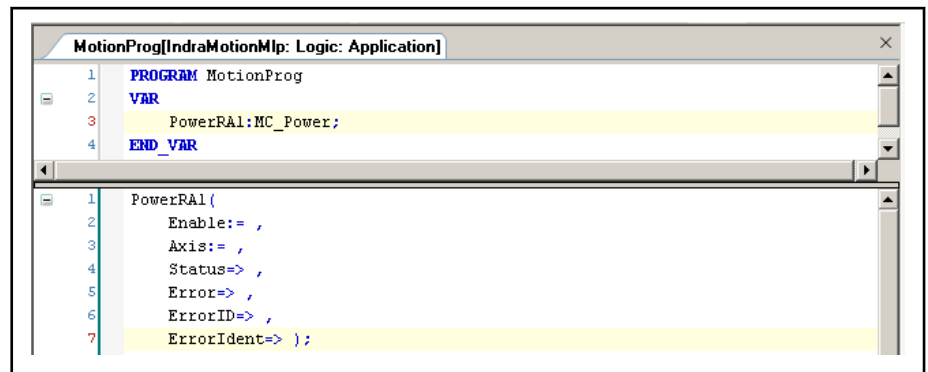


Abb. 5-10: Implementation, Template des Bausteins PowerRA1: MC_Power

"Enable" ordnen Sie bitte "en_PowerRA1" zu. Beim Verlassen der Zeile bietet Ihnen der Editor das Deklarationsfenster zur Bestätigung an.

"Axis" ordnen Sie bitte die entsprechende Achsreferenz zu. Verwenden Sie dazu erneut die F2-Eingabehilfe.

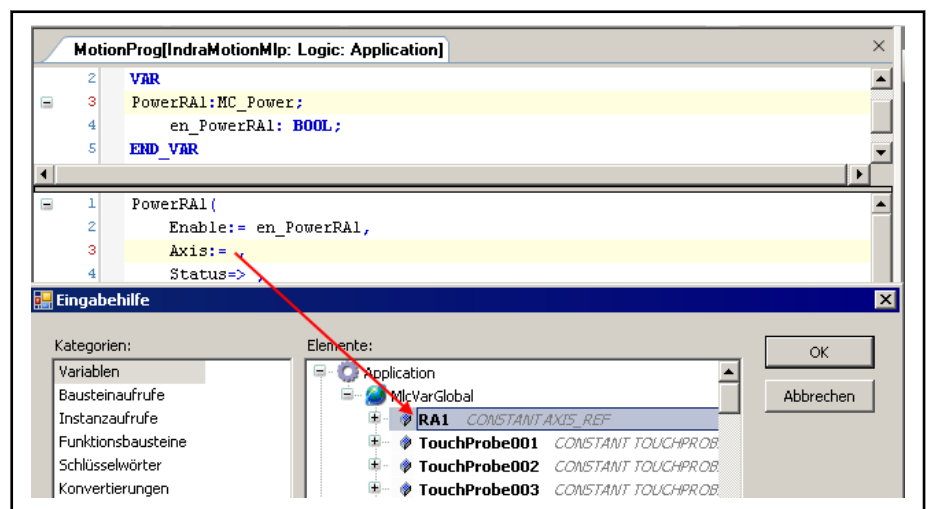


Abb. 5-11: F2-Eingabehilfe, Bereich der auszuwählenden Globalen Konstanten

Applikationsbeispiele

Die Achsreferenz wird als Globale Variable (Konstante) mit dem Namen der Variablenliste und der Achsbezeichnung eingetragen.

Kompletieren Sie die übrigen Variablen.

```

MotionProg[IndraMotionMlp: Logic: Application]
1  PROGRAM MotionProg
2  VAR
3      PowerRA1:MC_Power;
4      en_PowerRA1: BOOL;
5      st_PowerRA1: BOOL;
6      er_PowerRA1: BOOL;
7      ID_PowerRA1: ERROR_CODE;
8      Ident_PowerRA1: ERROR_STRUCT;
9  END_VAR

1  PowerRA1(
2      Enable:= en_PowerRA1,
3      Axis:= MlcVarGlobal.RA1,
4      Status=> st_PowerRA1,
5      Error=> er_PowerRA1,
6      ErrorID=> ID_PowerRA1,
7      ErrorIdent=> Ident_PowerRA1);
8

```

Abb. 5-12: PowerRA1: MC_Power: Komplett in Deklaration und Implementation.

Fügen Sie die Bausteine MC_MoveVelocity, MC_Stop und MC_Reset in das Programm ein.

```

1  // Leistung RA1 einschalten
2  PowerRA1(
3      Enable:= en_PowerRA1,
4      Axis:= MlcVarGlobal.RA1,
5      Status=>st_PowerRA1 ,
6      Error=> er_PowerRA1 ,
7      ErrorID=>ID_PowerRA1 ,
8      ErrorIdent=> Ident_PowerRA1);
9  // RA1 auf vordefinierte Geschwindigkeit bringen
10 MoveVelRA1(
11     Execute:= ex_MoveVelRA1,
12     Velocity:= 300,
13     Acceleration:= 20,
14     Deceleration:= 20,
15     Axis:= MlcVarGlobal.RA1,
16     InVelocity=> in_MoveVelRA1,
17     Active=> ac_MoveVelRA1,
18     CommandAborted=> ca_MoveVelRA1,
19     Error=> er_MoveVelRA1,
20     ErrorID=> ID_MoveVelRA1,
21     ErrorIdent=> Ident_MoveVelRA1);
22 // RA1 stoppen mit vorgegebener Verzögerung
23 StopRA1(
24     Execute:= ex_StopRA1,
25     Deceleration:= 20,
26     Axis:= MlcVarGlobal.RA1,
27     Done=> do_StopRA1,
28     Active=> ,
29     CommandAborted=> ,
30     Error=> er_StopRA1,
31     ErrorID=> ID_StopRA1,
32     ErrorIdent=> Ident_StopRA1);
33 // RA1 Fehler löschen
34 ResetRA1(
35     Execute:= ex_ResetRA1,
36     Axis:= MlcVarGlobal.RA1,
37     Done=> do_ResetRA1,
38     Error=> er_ResetRA1,
39     ErrorID=> ID_ResetRA1,
40     ErrorIdent=> Ident_ResetRA1);

```

Abb. 5-13: MotionProg – Strukturierter Text

Der Deklarationsteil, der die Variablen enthält, füllt sich während der Erstellung des Programms.

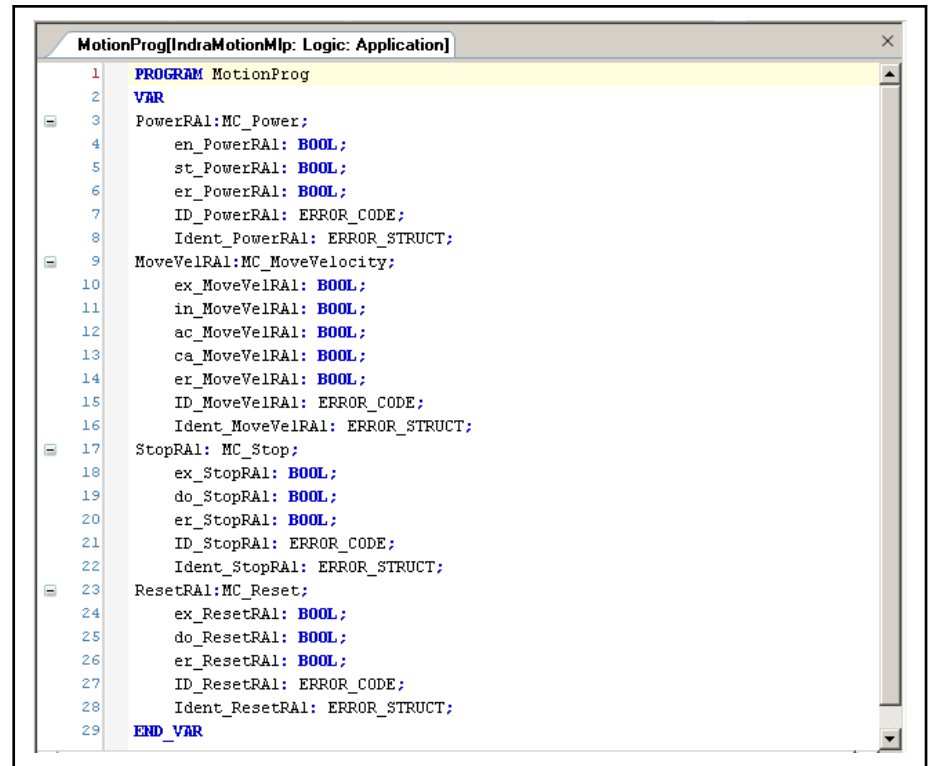


Abb. 5-14: MotionProg – Variablen

Wurde das Steuerungsprogramm erstellt, sollten Sie das Programm übersetzen:

1. **Erstellen ► Application übersetzen**, um zu sehen, ob der Quellcode fehlerfrei ist.
2. Ist die Übersetzung erfolgreich durchgeführt worden (siehe Aufgabenliste), schalten Sie die Steuerung online (**rechte Maustaste auf die Steuerung ► Online schalten** oder Icon auf Symbolleiste betätigen).
3. Danach ist eine Phasenschaltung von P2 nach BB erforderlich.
Wählen Sie im Kontextmenü des SERCOS-Knotens **SERCOS-Zustand ► P4**.
4. Zum Abschluss schalten Sie die Application online (das in IndraLogic erstellte Programm wird in die SPS geladen, **Debuggen ► Einloggen**) und starten das Programm (**Debuggen ► Start**).

Nun ist die Erstellung des Steuerungsprogramms abgeschlossen. In IndraLogic sehen Sie die Variablen, die später auch über die WinStudio-Oberfläche angesprochen werden sollen.

5.4.4 Einbinden der WinStudio-Visualisierung in IndraWorks

Um von extern auf Programmvariablen zugreifen zu können (wie das bei WinStudio der Fall ist, externe Anwendung) müssen Symbole mit bestimmten Zugriffsrechten definiert werden. Dies passiert über die Symbolkonfiguration (**Application ► Symbolkonfiguration**)

Ist die Symbolkonfiguration unter "Application" eingefügt, werden nach der Anwahl der Symbolkonfiguration und nach Klicken auf Aktualisieren (1), die verfügbaren Variablen links angezeigt. Mittels Pfeil-Tasten (2) können die Variablen in den Bereich der **"Ausgewählten Variablen"** bewegt werden. Die so

Applikationsbeispiele

ausgewählten Variablen können dann in einer externen Anwendung angesprochen werden. Wurden die Variablen aus dem **MotionProg** selektiert, ergibt sich nachfolgende Abbildung:

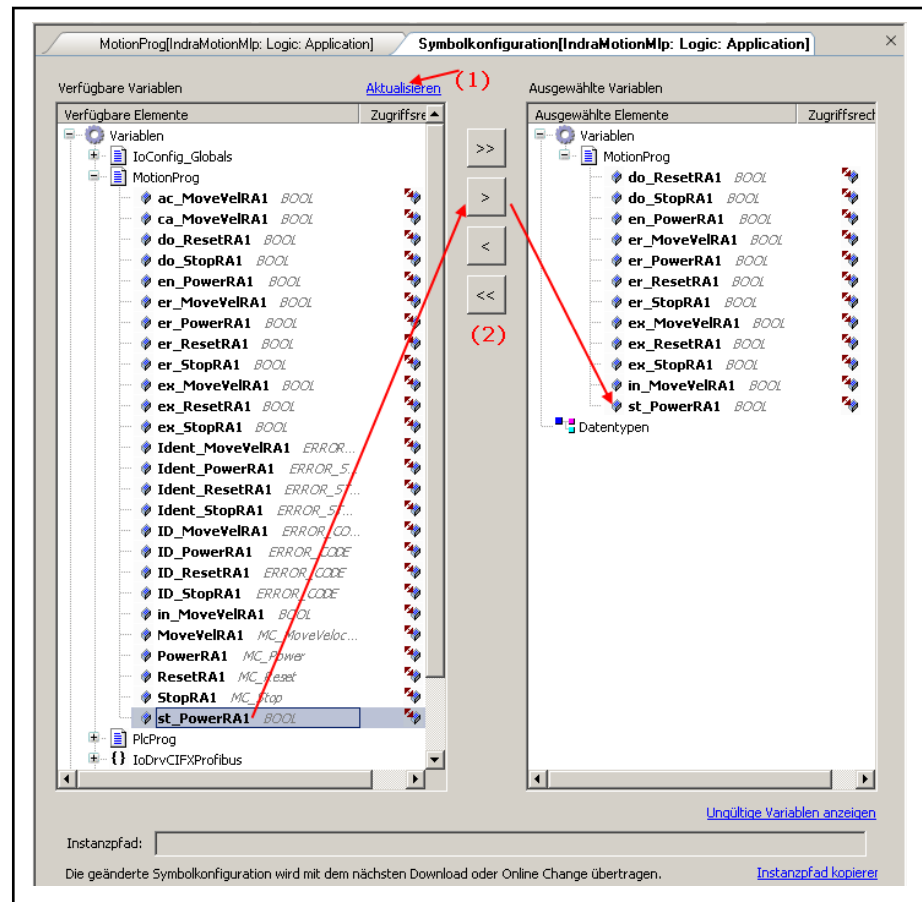


Abb. 5-15: Symbolkonfiguration – Variablen ausgewählt

Drei Schritte sind erforderlich, um die Symbolkonfiguration korrekt in die SPS zu schreiben.

1. Zum Einen ist die Option **Erstellen ► Alles bereinigen** auswählen, damit werden alte Daten der SPS gelöscht.
2. Danach muss die Applikation neu übersetzt werden, das Programm hat sich geändert (**Erstellen ► Übersetzen**).
3. Als letzten Schritt schalten Sie die Application online, um das geänderte Programm auf die SPS zu schreiben (**Debuggen ► Einloggen**).

Nun können von einer externen Anwendung (hier WinStudio) die ausgewählten Variablen von der SPS gelesen werden.

Ziehen Sie die Visualisierung "VEH/VEP" aus der Bibliothek in den Projektknoten.

In diesem Fenster wählen Sie den Gerätetypen, vergeben einen Name für die Visualisierung, tragen die IP-Adresse der Steuerung ein und wählen den Typ der Applikation aus.

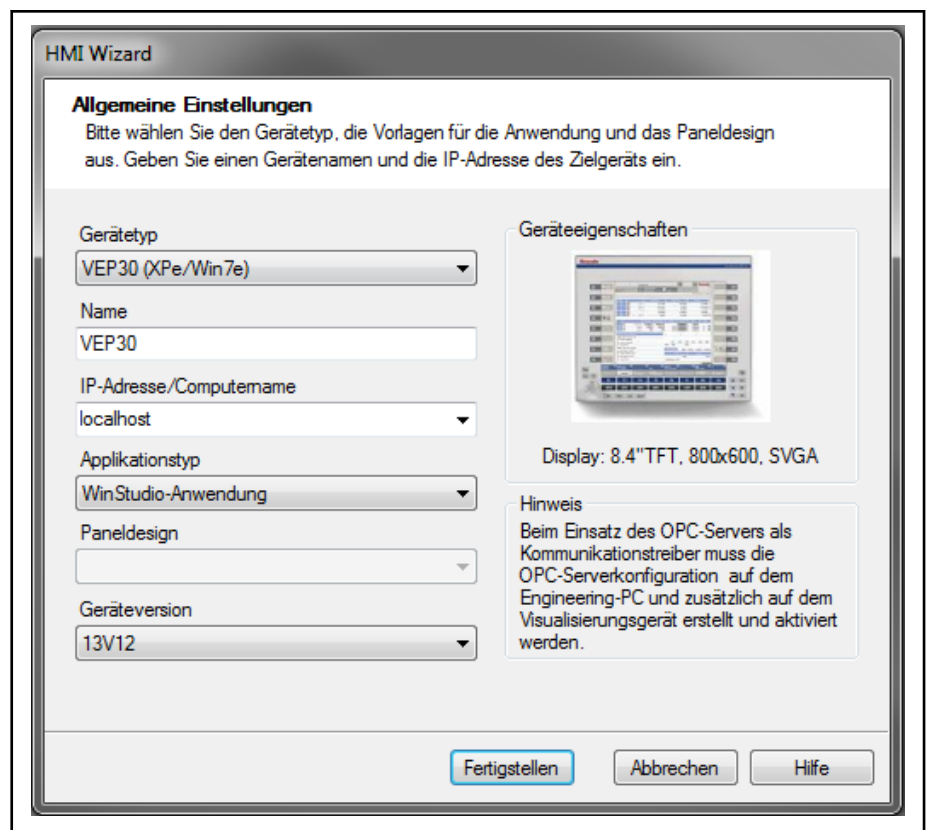


Abb. 5-16: Einfügen Visualisierung VEP – Allgemeine Einstellungen

- Der **Gerätetyp** ist nach der zu verwendeten HMI festzulegen (in diesem Fall **VEP30 (XPe/Win7e)**).
- Im Feld **Name** ist der Name einzutragen, unter welchem die Visualisierung im Projekt geführt wird (in diesem Fall **VEP30**).

Sie können einen eigenen vergeben (IndraWorks meldet, falls dieser ungültige Zeichen enthalten sollte) oder Sie verwenden den vorgegebenen.

- Unter **IP-Adresse** tragen Sie die IP des VEP30 ein.
- Als **Applikationstyp** ist eine WinStudio-Anwendung zu wählen.

Als nächstes muss die Kommunikation zwischen WinStudio und der SPS hergestellt werden.

Sie öffnen WinStudio, indem Sie auf das Objekt "WinStudio" im Projektbaum unter der Visualisierung klicken.

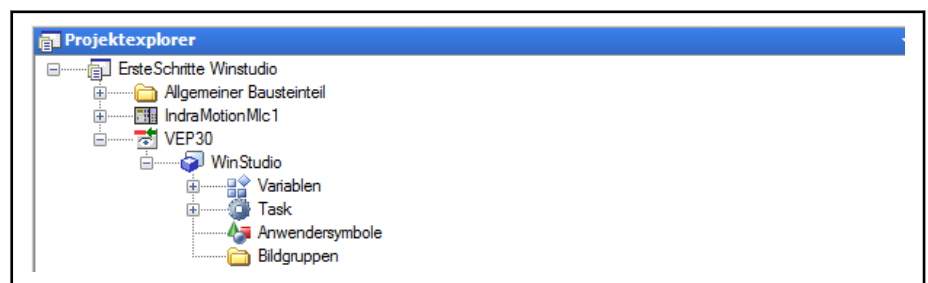


Abb. 5-17: Projektexplorer mit WinStudio Knoten

Um die Variablen aus der Steuerung lesen zu können, muss ein spezielles Treiberblatt in WinStudio eingebunden werden. In diesem Beispielprojekt wird der BRWS-Treiber verwendet. Binden Sie hierzu zunächst über

Applikationsbeispiele

VEP30 ► WinStudio ► Variablen ► Kommunikation ► Treiber hinzufügen/Entfernenden BRWS Treiber in das Projekt ein. Markieren Sie den BRWS Treiber im Auswahlfeld "Verfügbare Treiber" durch einen Rechtsklick und betätigen Sie anschließend das Schaltfeld "Auswählen". Der Treiber erscheint nun im Feld "Treiber auswählen". Bestätigen Sie mit "OK"

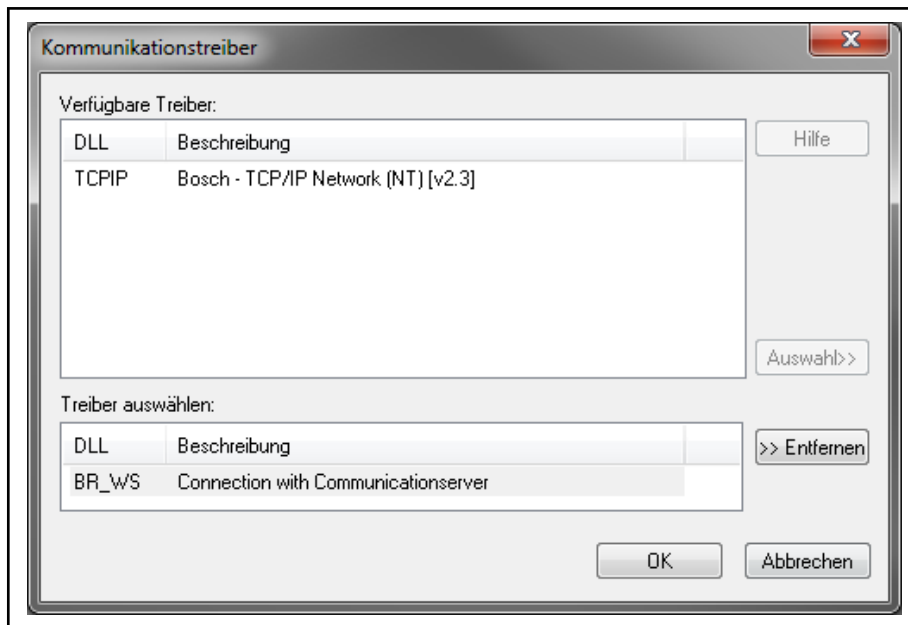


Abb. 5-18: Der BRWS Treiber wurde über das Schaltfeld "Auswahl" eingebunden

Das Treiberblatt erzeugen Sie im Arbeitsbereich unter **VEP30 ► WinStudio ► Variablen ► Kommunikation ► BRWS ► Neues BR_WS-Treiberblatt**.

Durch Klicken mit der rechten Maustaste auf BRWS-Treiber, erscheint das Kontextmenü "Neues BR_WS-Treiberblatt".

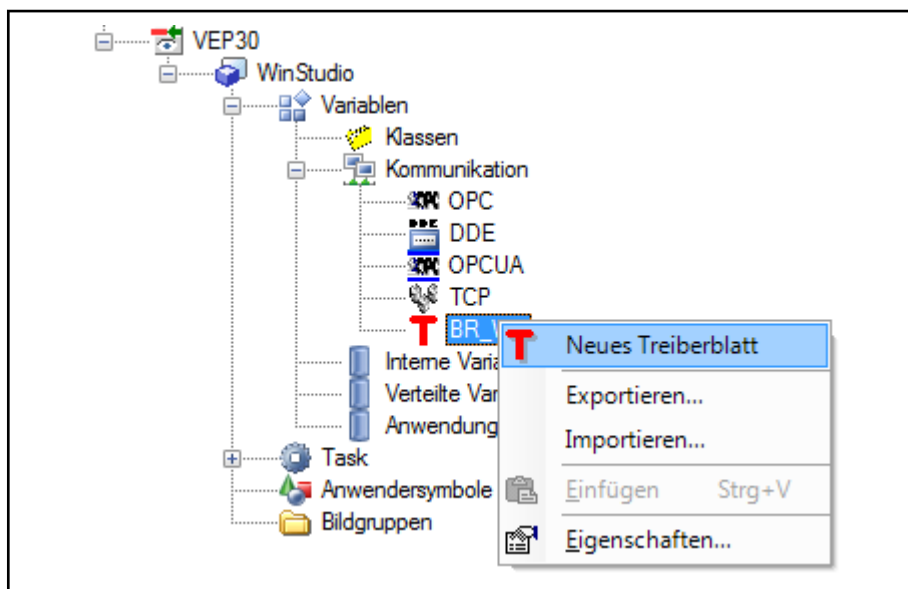


Abb. 5-19: Neues BR_WS-Treiberblatt anlegen

Der Treiber muss nun speziell konfiguriert werden.

Wählen Sie den Ordner "BRWS" mit der rechten Maustaste an. Wieder erscheint ein Kontextmenü, in dem Sie **Eigenschaften** wählen müssen.

In dem sich öffnenden Fenster tragen Sie die IP-Adresse der Steuerung ein

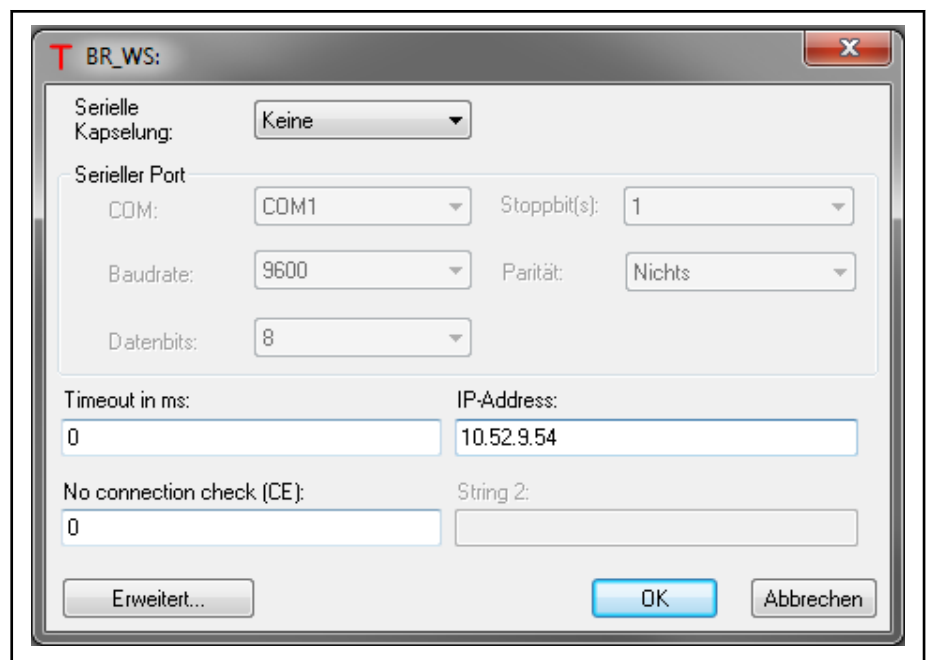


Abb. 5-20: BRWS-Eigenschaften, hier IP-Adresse

Im folgenden Schritt werden die SPS-Variablen der WinStudio-Umgebung bekannt gemacht.

Hierzu klicken Sie auf das Treiberblatt "1". Daraufhin wird rechts neben dem Projektbaum ein neues Fenster geöffnet.

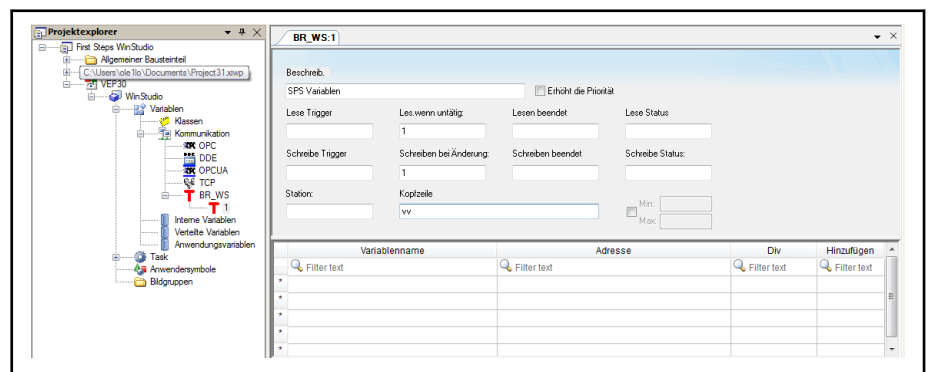


Abb. 5-21: BRWS-Treiberblattkonfiguration, Blatt "1"

Füllen Sie nun die Felder aus, wie oben zu sehen.

Die Felder **Lesen, wenn untätig** und **Schreiben bei Änderung** müssen Sie mit dem Wert 1 versehen, damit man mit den Variablen arbeiten kann.

Im Feld **Kopfzeile** muss ein für jedes Treiberblatt eindeutiger Eintrag erstellt werden (hier wird beispielhaft "vv" verwendet, der Eintrag ist frei wählbar). Speichern Sie nun das Treiberblatt (z.B. in dem Sie mit Strg+S das Projekt speichern). Die Treiberblattkonfiguration ist nun im System bekannt.

Der nächste Schritt ist, die SPS-Variablen, die in der Symbolkonfiguration freigegeben wurden, in WinStudio einzulesen. Hierzu stehen Ihnen zwei Möglichkeiten zur Verfügung:

1. Mit Hilfe des IndraWorks-Prozessvariablen-Browsers:

Applikationsbeispiele

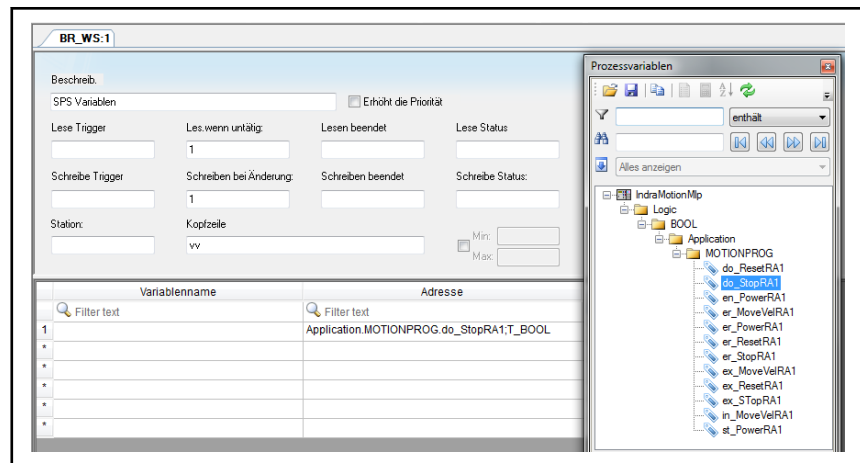


Abb. 5-22: BRWS Treiberblatt und IndraWorks-Prozessvariablen-Browsers

Öffnen Sie hierzu zunächst über **Ansicht ► Andere Fenster ► Prozessvariablen** das Werkzeugfenster für Prozessvariablen. Nach Aktualisie-

ren des Prozessvariablen-Browsers mit  sehen Sie die in der Symbolkonfiguration der Steuerung verfügbaren SPS-Variablen. Diese Können Sie nun mittels Drag&Drop in die Spalten des Treiberblattes ziehen. Der Variablenname mit Zugriffspfad wird mit ";" gefolgt vom Typ der Variablen "T_BOOL" angegeben.

2. Mittels des BRWS Variablen-Browsers

Setzen Sie hierzu den Cursor der Maus in den Tabellenbereich des Treiberblattes (z.B. in die Spalte "Adresse") und Führen Sie einen Rechtsklick aus, durch diesen Vorgang öffnet sich ein neues Fenster, der Variablenbrowser.

In diesem Fenster wählen Sie "Variablen lesen (online)".

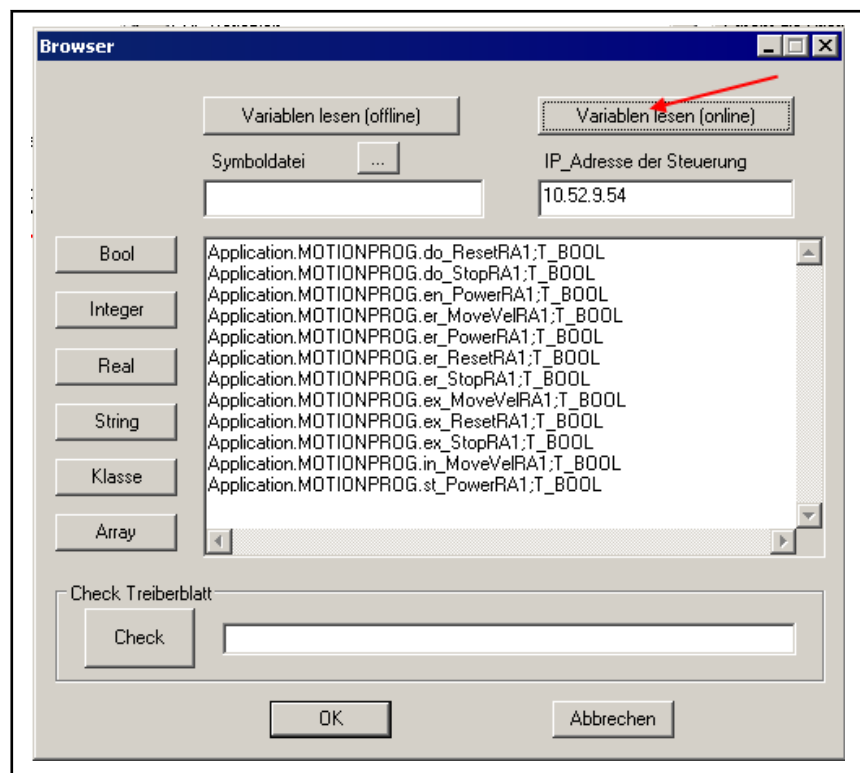


Abb. 5-23: Browser, Variablen Online-lesen

Applikationsbeispiele

Im Browser sind alle SPS-Variablen zu sehen, die zuvor unter der **Sym-bolkonfiguration** freigegeben wurden und die über WinStudio angesprochen werden können.

Markieren und kopieren Sie die Variablen. Schließen Sie das Fenster mit "OK"

Kopieren Sie diese Variablen in die Spalte "Adresse" (1), oberste Zelle.

WinStudio baut einen eigenen Namensraum auf. Dies ist notwendig, da praktisch die Visualisierung unabhängig und möglicherweise vor dem SPS-Programm erstellt werden soll.

Die beiden Namenslisten werden Variable für Variable einander zugeordnet (2).

Die **Variablen aus der Sicht der WinStudio Anwendung** befinden sich unter **Anwendungsvariablen** (3).

Sie können die WinStudio-Liste entweder direkt (Liste Anwendungsvariablen) oder wie nachfolgend gezeigt ausfüllen.

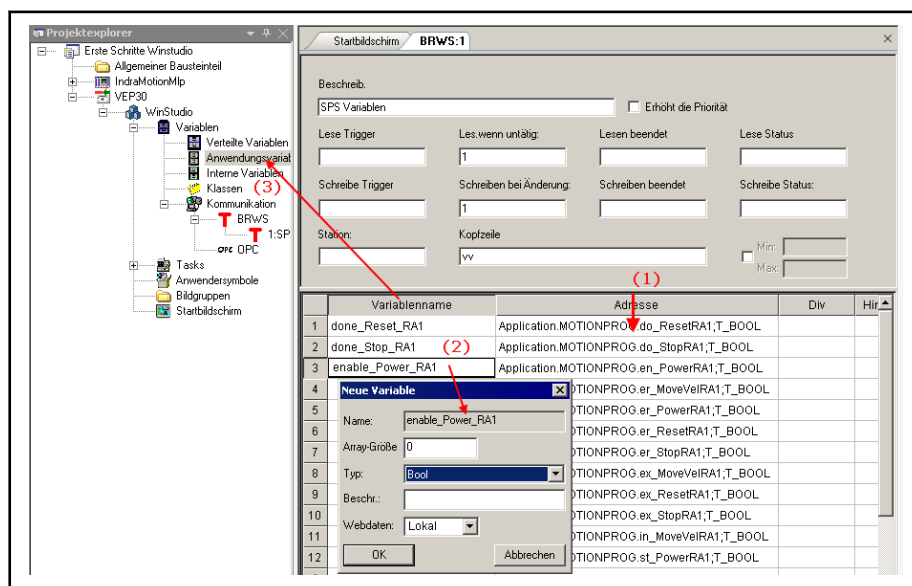


Abb. 5-24: BRWS-Eigenschaften, hier Blatt 1, mit Variablenamen und Adressen

Startbildschirm BRWS:1 Anwendungsvariablen					
	Name	Größe	Typ	Beschreibung	Bereich:
1	done_Reset_RA1	0	Bool		Lokal
2	done_Stop_RA1	0	Bool		Lokal
3	enable_Power_RA1	0	Bool		Lokal
*			Integer		Server
*			Integer		Server
*			Integer		Server
*			Integer		Server
*			Integer		Server

Abb. 5-25: WinStudio: Anwendungsvariablen, Zwischenstand

Applikationsbeispiele

	Variablenname	Adresse	Div	Hinzufüg
1	done_Reset_RA1	Application.MOTIONPROG.do_ResetRA1;T_BOOL		
2	done_Stop_RA1	Application.MOTIONPROG.do_StopRA1;T_BOOL		
3	enable_Power_RA1	Application.MOTIONPROG.en_PowerRA1;T_BOOL		
4	error_MoveVelocity_RA1	Application.MOTIONPROG.er_MoveVelRA1;T_BOOL		
5	error_Power_RA1	Application.MOTIONPROG.er_PowerRA1;T_BOOL		
6	error_Reset_RA1	Application.MOTIONPROG.er_ResetRA1;T_BOOL		
7	error_Stop_RA1	Application.MOTIONPROG.er_StopRA1;T_BOOL		
8	execute_MoveVelocity_RA1	Application.MOTIONPROG.ex_MoveVelRA1;T_BOOL		
9	execute_Reset_RA1	Application.MOTIONPROG.ex_ResetRA1;T_BOOL		
10	execute_Stop_RA1	Application.MOTIONPROG.ex_StopRA1;T_BOOL		
11	in_Velocity_RA1	Application.MOTIONPROG.in_MoveVelRA1;T_BOOL		
12	status_Power_RA1	Application.MOTIONPROG.st_PowerRA1;T_BOOL		
*				

Abb. 5-26: BRWS-Eigenschaften, hier Blatt 1, mit Variablennamen und Adressen

Wurden die SPS-Variablen korrekt eingebunden, sind alle Variablen in WinStudio ► Variablen ► Anwendungsvariablen aufgelistet

Startbildschirm BRWS:1 Anwendungsvariablen					
	Name	Größe	Typ	Beschreibung	Bereich:
1	done_Reset_RA1	0	Bool		Lokal
2	done_Stop_RA1	0	Bool		Lokal
3	enable_Power_RA1	0	Bool		Lokal
4	error_MoveVelocity_RA1	0	Bool		Lokal
5	error_Power_RA1	0	Bool		Lokal
6	error_Reset_RA1	0	Bool		Lokal
7	error_Stop_RA1	0	Bool		Lokal
8	execute_MoveVelocity_RA1	0	Bool		Lokal
9	execute_Reset_RA1	0	Bool		Lokal
10	execute_Stop_RA1	0	Bool		Lokal
11	in_Velocity_RA1	0	Bool		Lokal
12	status_Power_RA1	0	Bool		Lokal
*			Integer		Server

Abb. 5-27: Kontrolle der Übernahme: WinStudio – Anwendungsvariablen

5.4.5 Erstellen der WinStudio-Visualisierung

Nachdem die Variablen eingebunden wurden und nun als Anwendervariablen zu Verfügung stehen, kann jetzt mit der graphischen Oberfläche begonnen werden.

Schließen Sie das aktuelle Fenster. Wählen Sie den Ordner "WinStudio" mit der rechten Maustaste an. Im nächsten Schritt fügen Sie ein neues Bild ein.

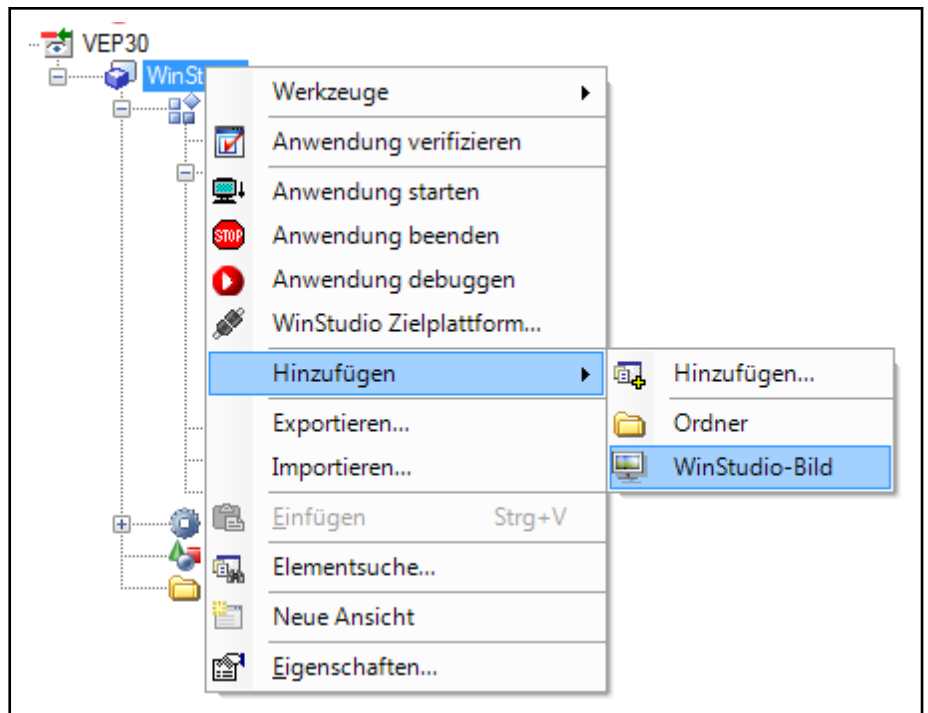


Abb. 5-28: WinStudio – Einfügen Bild

Sie können den Namen des Bildes frei wählen: hier "Startbildschirm".

Öffnen Sie das Bild. Nach einem Klick mit der rechten Maustaste in das Bild wird "Bildeigenschaften" gewählt.

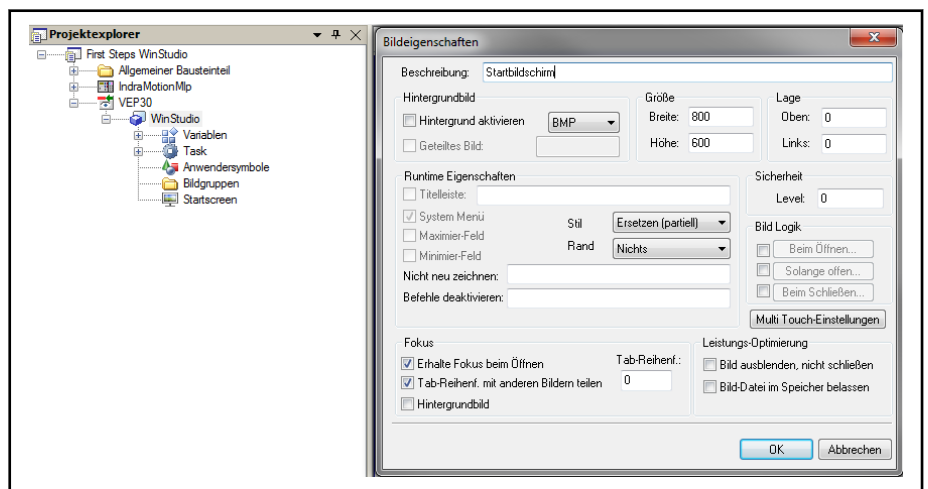


Abb. 5-29: WinStudio – Bild/ Einstellungen

Ebenso öffnet sich die Arbeitsfläche rechts neben dem Arbeitsbaum. Nun können Sie mit den einzelnen Werkzeugleisten, am rechten Rand von WinStudio, Objekte in das Bild einfügen.

Applikationsbeispiele

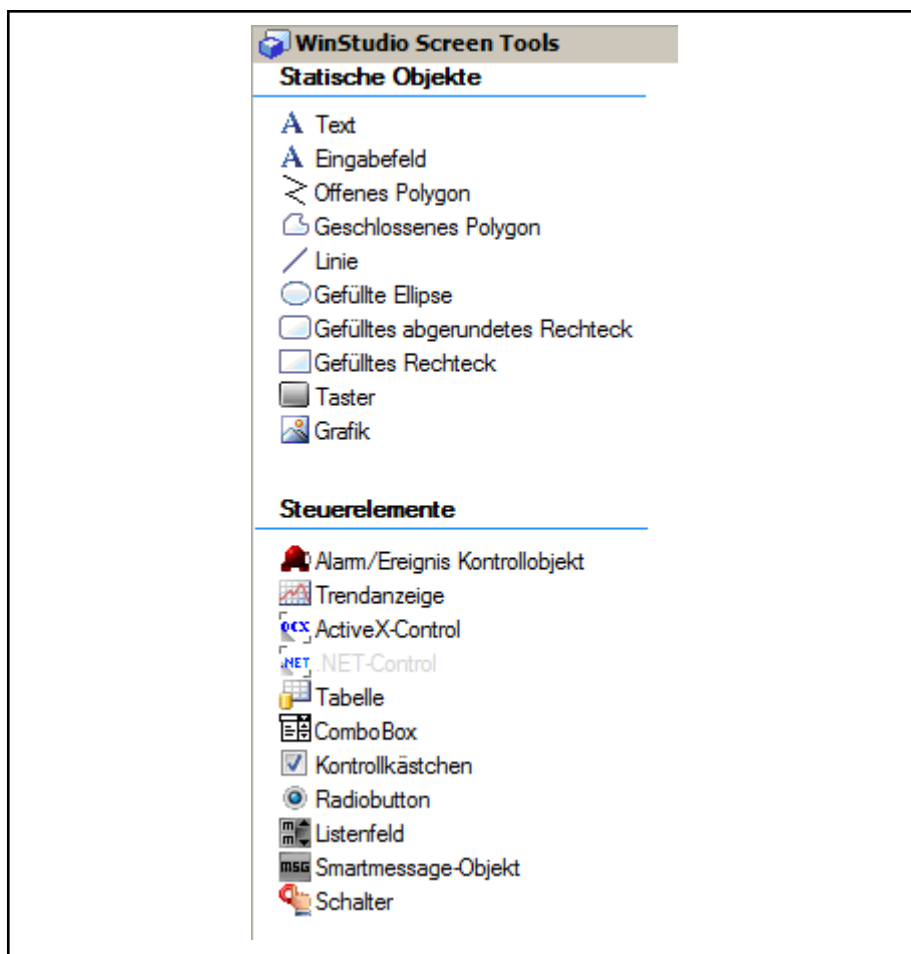


Abb. 5-30: WinStudio – Screen Tools

Nun folgt die Zusammenstellung von Objekten zu einer kleinen Demo-Oberfläche. Hierzu verändern Sie mit der rechten Maustaste die Hintergrundfarbe der späteren Oberfläche.

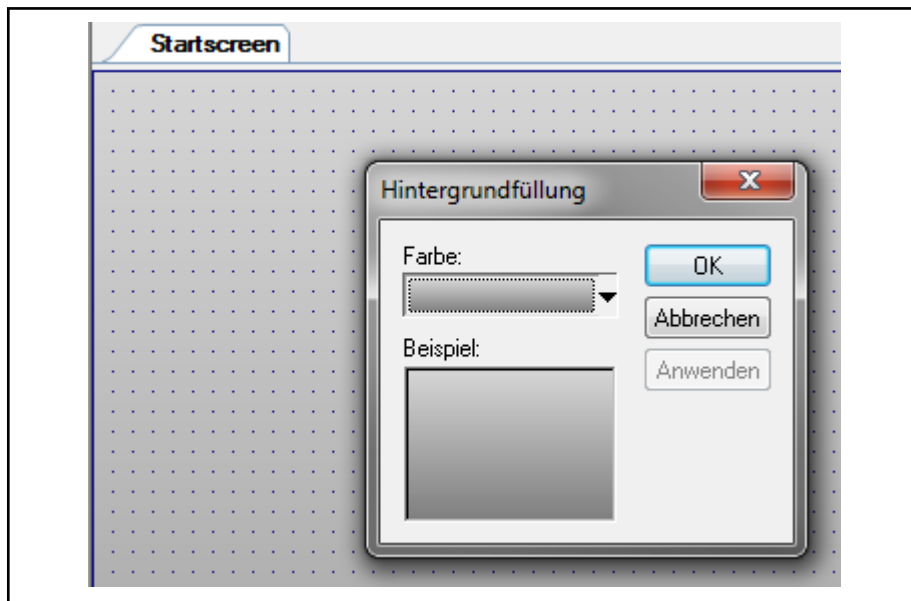


Abb. 5-31: Bild "Startscreen" – Hintergrundfarbe ändern

Applikationsbeispiele

Um die Oberfläche optisch zu unterteilen, fügen Sie das Objekt "Gefülltes Rechteck" in das Bild ein. Dieses Rechteck soll das obere Viertel der Arbeitsfläche ausfüllen.

Das Objekt kann noch spezieller angepasst werden, wenn Sie mit der rechten Maustaste das Kontextmenü öffnen und dort "Eigenschaften" auswählen.

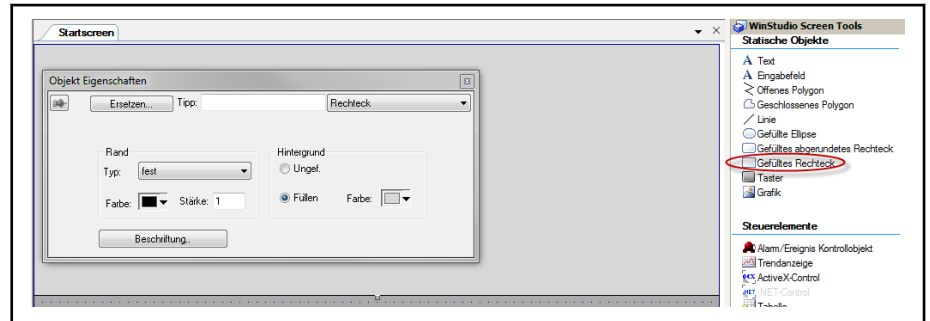


Abb. 5-32: WinStudio – Objekt "Gefülltes Rechteck" eingefügt

Wie zu ersehen, wurde die Füllfarbe angepasst und das Rechteck auf die komplette Länge des Bildes zugeschnitten.

WinStudio unterstützt den Anwender mit einer Bibliothek, die u.a. eine Reihe von Symbolen enthält.

Die Bibliothek ist entweder über den Hauptmenüpunkt "WinStudio" oder über das Kontextmenü des WinStudio-Ordners erreichbar.

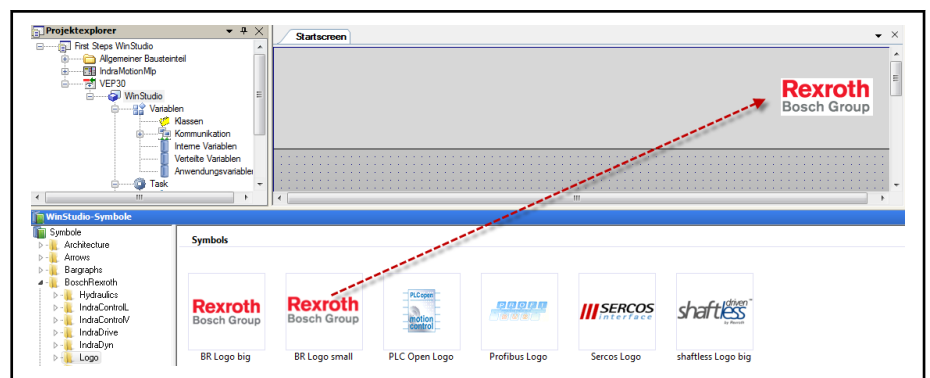


Abb. 5-33: WinStudio – Bibliothek geöffnet, Logo übernommen

Das gewünschte Symbol, hier "BR Logo small" wird per Drag&Drop eingefügt.

Nun fügen Sie ein "Text"-Objekt aus den WinStudio ScreenTools in das Bild ein. Formatierungen können über die Einstellungen des Objektes (**rechte Maustaste ► Einstellungen**) vorgenommen werden.

Applikationsbeispiele

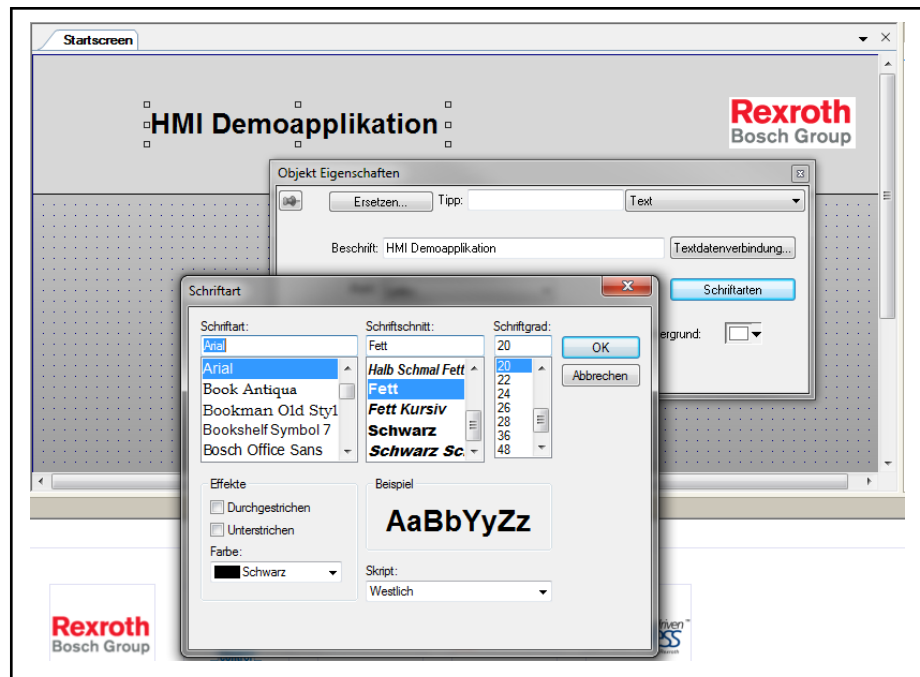


Abb. 5-34: WinStudio – Textobjekt eingefügt

Im nächsten Schritt werden die Steuerelemente in das Bild eingefügt, hier wird exemplarisch der Switch (Steuerelement "Switch") aus den WinStudio ScreenTools verwendet.

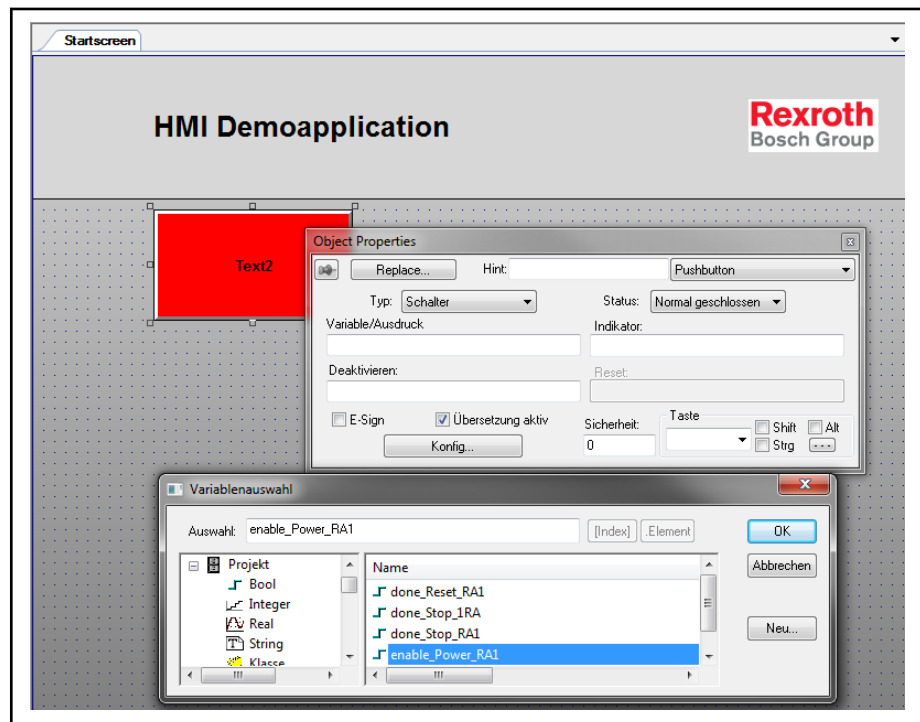


Abb. 5-35: WinStudio – "Switch" eingefügt

Bei den Steuerelementen müssen Sie verschiedene Einstellungen durchführen.

Wie im Bild angezeigt, müssen verschiedene Felder gesetzt werden.

Im speziellen konfigurieren Sie den Switch in den Objekteigenschaften als Typ "Schalter".

Applikationsbeispiele

In das Feld **Var/Ausdruck** fügen Sie durch Doppelklick ins Feld "Variable/ Ausdruck" die Variable ein, die durch das Objekt Switch angesprochen werden soll

(in diesem Fall ist das **enable_Power_RA1**).

Bei diesem Vorgang öffnet sich die **"Variablenauswahl"** in welchem die Variable ausgewählt werden kann.

Unter **Konfig..** stellen Sie das Verhalten bei offenem und geschlossenem Schalter ein.



Die Einstellungen "Farbwahl", "Status" (normal geschlossen/ normal offen) und die "Beschriftung" sind anwenderspezifisch zu belegen.

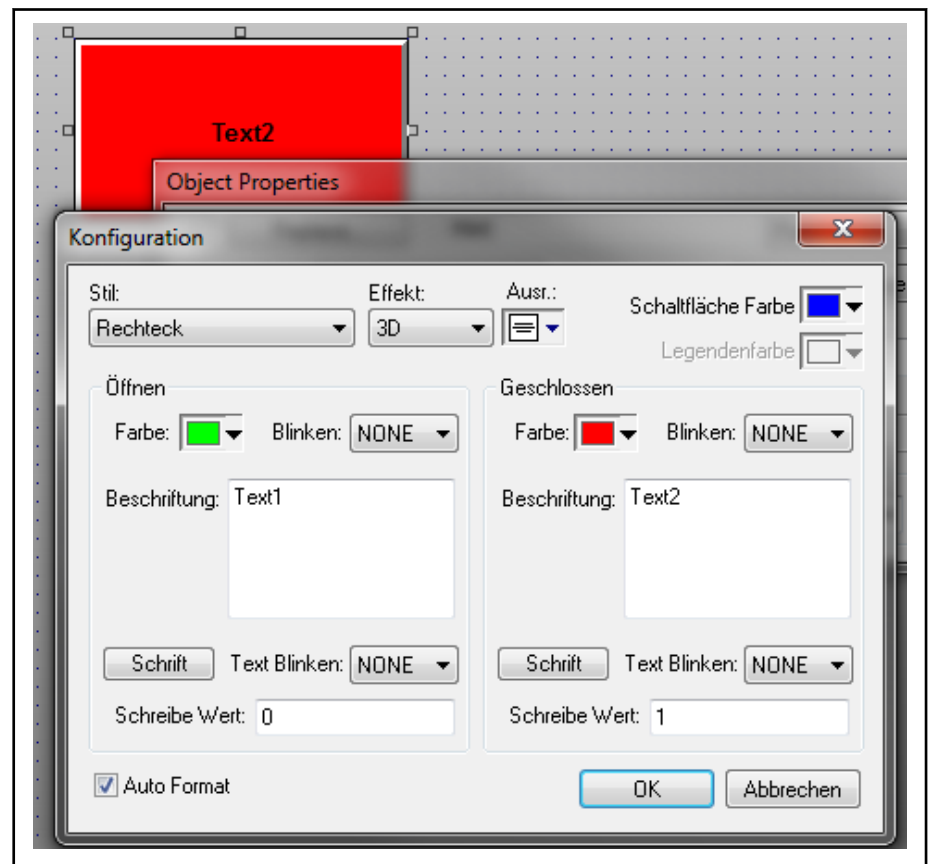


Abb. 5-36: Konfigurationsdialog des Switch Objektes

Realisiert wurden die vier Schalter mit nachstehenden Konfigurationen:

Leistung zuschalten

- Typ: Schalter
- Status: Normal geschlossen
- Variable: enable_Power_RA1
- Beschriftung (öffnen): Leistung EIN / grün
- Beschriftung (geschlossen): Leistung AUS / grau

Reset Antrieb

- Typ: Schalter
- Status: Normal geschlossen

Applikationsbeispiele

- Variable: execute_Reset_RA1
- Beschriftung (öffnen): Reset Antrieb / grün
- Beschriftung (geschlossen): Reset AUS / grau

Antrieb Einschalten

- Typ: Schalter
- Status: Normal geschlossen
- Variable: execute_MoveVelocity_RA1
- Beschriftung (öffnen): Antrieb RUN / grün
- Beschriftung (geschlossen): Antrieb EIN / grau

Antrieb Anhalten

- Typ: Schalter
- Status: Normal geschlossen
- Variable: execute_Stop_RA1
- Beschriftung (öffnen): Antrieb AUS / grün
- Beschriftung (geschlossen): Antrieb STOP / grau.

Um den momentanen Status des Antriebs anzuzeigen, fügen Sie Kontrollfelder in die Oberfläche ein.

Hier sind die Einstellungen **"Beschreibung"** und **"Variable"** vorzunehmen.

Im Folgenden die Konfiguration aller Kontrollfelder:

Status Leistung

- Beschreibung: Leistung EIN
- Variable: status_Power_RA1

Status Leistung (Fehler)

- Beschreibung: Leistung FEHLER
- Variable: error_Power_RA1

Status Antrieb

- Beschreibung: Antrieb EIN
- Variable: in_Velocity_RA1

Status Antrieb (Fehler)

- Beschreibung: Antrieb FEHLER
- Variable: error_MoveVelocity_RA1

Status Reset Antrieb

- Beschreibung: Reset EIN
- Variable: done_Reset_RA1

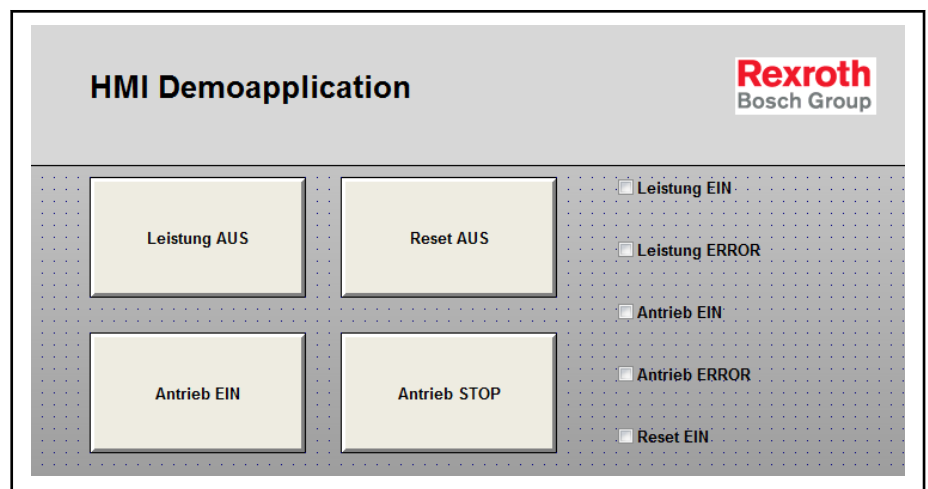


Abb. 5-37: WinStudio – Fertige Oberfläche

5.4.6 Starten der Oberfläche

Es sind nur noch wenige Schritte erforderlich, bis die Oberfläche auf der Steuerung läuft.

Im nächsten Schritt wird der Startbildschirm über die Projekteinstellungen festgelegt.

Unter dem Pfad **WinStudio ► Eigenschaften** wählen Sie die Registerkarte **"Viewer"** und tragen im Fenster **"Startbild"** den von Ihnen erstellten Startbildschirm ein (notwendig, sonst wird die Oberfläche im Viewer nicht geladen). Alternativ können Sie die Eigenschaft auch über das Kontextmenü des entsprechenden Bildknotens im Projektexplorer anwählen).

In diesem Reiter können weitere Darstellungsoptionen gewählt werden, z.B. das Anzeigen der WinStudio-Menüleiste.

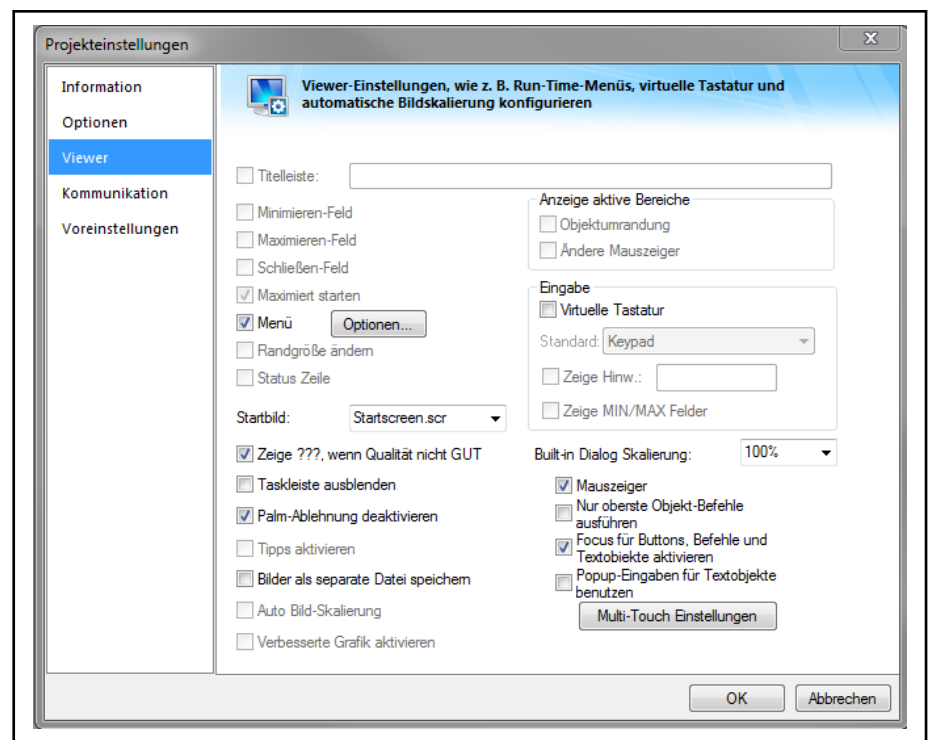


Abb. 5-38: Winstudio Projekteinstellungen, Registerkarte "Viewer"

Die vorgenommenen Einstellungen speichern Sie nun im Projekt ab.

Applikationsbeispiele

Überführen Sie jetzt Ihr SPS-Projekt in einen arbeitsbereiten Zustand:

1. Gehen Sie mit der Steuerung Online.
2. Loggen Sie ein.
3. Starten Sie Ihre Applikation.
4. Schalten Sie den sercos Bus auf Phase 4 (P4).

Nun müssen Sie das WinStudio-Projekt zur Zielplattform übertragen.

Wählen Sie den Kontextmenüpunkt "WinStudio Zielplattform" auf dem WinStudio Knoten

Verbinden Sie sich nun mit der HMI (1). Bei Geräten mit Betriebssystem Windows CE6.0 bzw. Windows 7 embedded compact muss jetzt das Laufzeitsystem auf das HMI-Gerät übertragen werden (2). Bei Geräten mit Windows 7 embedded bzw. Windows XP embedded ist Schritt(2) nicht notwendig.

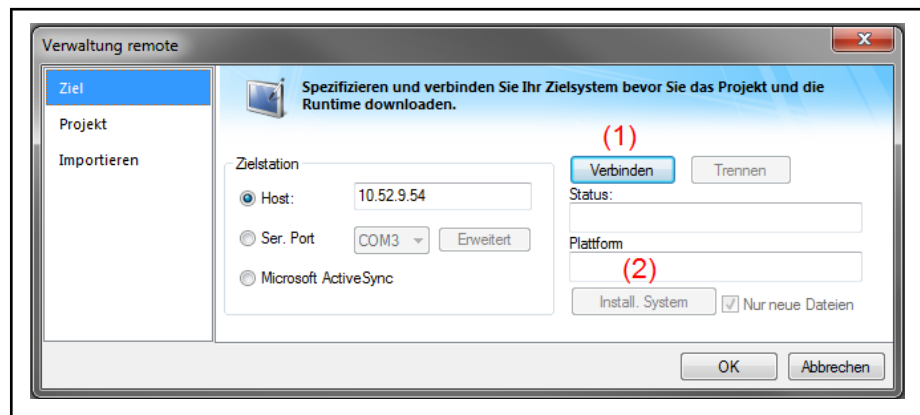


Abb. 5-39: Ausführungsplattform: Übertragen der Projektdaten

Danach legen Sie den Zielpfad für die Projektdaten auf dem Zielgerät fest (1). Folgende Default-Pfade werden für die verschiedenen Zielgeräte verwendet:

- **VEPxx.4/5, Windows Xp embedded/ Windows 7 embedded:**
D:\Documents\Rexroth\My Documents\WinStudio for IndraWorks Projects\config
- **VR21xx, Windows 7 embedded compact:**
\\FlashDrv\Rexroth\WinStudio\config\
- **VEPxx.4, Windows CE6.0:**
\\StorageCard\Rexroth\Visu\WinStudio\config\

Wenn Sie sich mit dem jeweiligen Gerät verbunden haben, wird ihnen der Pfad automatisch angeboten. Vergewissern Sie sich dennoch, dass die Zielpfadangabe (1) korrekt eingetragen ist und korrigieren Sie diese ggf. mittels der Schaltfläche ...

Achten Sie weiterhin darauf, dass die Option "**behalten Benutzerdateien**" aktiviert ist. Somit vermeiden Sie, dass Dateien, die sich zuvor im angegebenen Zielpfad befunden haben, beim Übertragen der Projektdateien gelöscht werden.

Starten Sie nun den Download der Projektdateien mittels der Schaltfläche **Download(2)**(der Ladevorgang ist im Fenster des Remote Agents auf dem Zielgerät sichtbar).

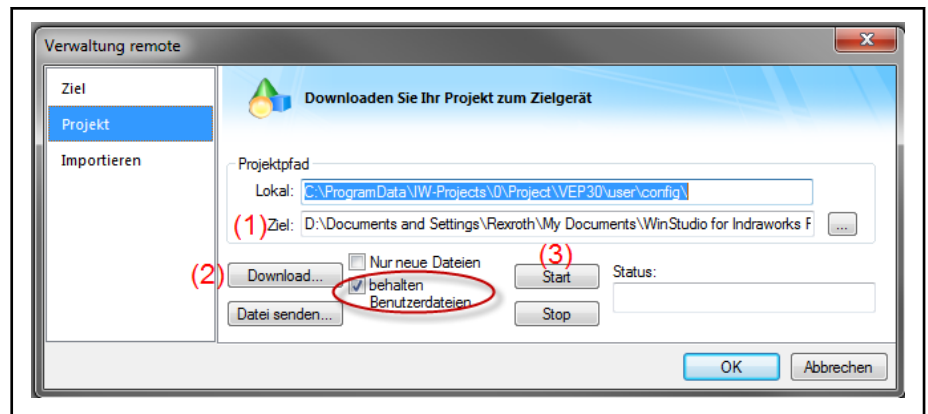


Abb. 5-40: Anwendung laden und starten

Starten Sie nun die Anwendung (3).

Oberfläche beim Booten der
Steuerung automatisch starten

Damit die HMI-Applikation beim Booten von Windows XP/ Windows 7 embedded mit gestartet wird, gehen Sie folgendermaßen vor:

Öffnen Sie **Start ► Programs ► Rexroth ► IndraWorks ► WinStudio**.

Öffnen Sie das Kontextmenü des "Startup" - Dienstes und erzeugen Sie eine Verknüpfung. Fügen Sie die Verknüpfung nun in den Windows Autostart-Ordner ein.

5.5 IndraLogic Visualisierung

Ein Applikationsbeispiel für die VI-Logic Visualisierung ist in "Rexroth IndraLogic XLC 13VRS Die ersten Schritte", DOK-XLC***-F*STEP**V13-CO01-DE-P enthalten.

5.6 HMIs von Fremdherstellern

5.6.1 OPC.DA Server

Hilfsmittel zur Konfiguration des OPC.DA Server auf einem PC

Für die einfachere Handhabung der OPC.DA Server Konfigurationsdatei "OPCServer.ini" sowie der Log Datei "OPCServer.log" auf dem PC, können die beiden Programmlistings "Stapelverarbeitungsprogramm zum Öffnen und Editieren der Datei OPCServer.ini" auf Seite 167 und "Stapelverarbeitungsprogramm zum Öffnen der Datei OPCServer.log" auf Seite 168 verwendet werden. Sie ermitteln sich über die Registry und den Umgebungsvariablen die Pfade zu den Dateien und öffnen diese im "Notepad" Editor ("SET EDITOR_FILE=notepad.exe").

Stapelverarbeitungsprogramm zum Öffnen und Editieren der Datei "OPCServer.ini"

```
@ECHO OFF
REM
REM Open IndraLogic OPC Server Configuration File.
REM If it does not exist, copy it!
REM

SET EDITOR_FILE=notepad.exe
SET REGKEY=HKLM\Software\Rexroth\IndraWorks
SET REGKEYNAME=CurrentIndraLogicOPCDAIniFileName

REM 'delims' is set with TAB and SPACE
FOR /F "tokens=2* delims=" "%a" IN ('REG QUERY "%REGKEY%" /v %REGKEYNAME%') DO SET OPCFILE="%%b"

IF EXIST "%OPCFILE:="%" GOTO fileopen

COPY "%CommonProgramFiles%\Rexroth\OPCServer.ini.template" "%OPCFILE:="%
```

Applikationsbeispiele

```
:fileopen
START %EDITOR_FILE% "%OPCFILE:="%
```

Abb. 5-41: Stapelverarbeitungsprogramm zum Öffnen und Editieren der Datei "OPCServer.ini"

Stapelverarbeitungsprogramm zum Öffnen der Datei "OPCServer.log"

```
@ECHO OFF
REM
REM Open IndraLogic OPC Server Log File.
REM

SET EDITOR_FILE=notepad.exe
SET REGKEY=HKLM\Software\Rexroth\IndraWorks
SET REGKEYNAME=CurrentIndraLogicOPCDALogFileName

REM 'delims' is set with TAB and SPACE
FOR /F "tokens=2* delims=" " %a IN ( 'REG QUERY "%REGKEY%" /v %REGKEYNAME%' ) DO SET OPCFILE="%b"

IF EXIST "%OPCFILE:="%" GOTO fileopen

GOTO :EOF
```

```
:fileopen
START %EDITOR_FILE% "%OPCFILE:="%
```

Abb. 5-42: Stapelverarbeitungsprogramm zum Öffnen der Datei "OPCServer.log"

Hilfsmittel zum Übertragen der OPC.DA Server Konfiguration auf ein Visualisierungsgerät VEPxx.4 mit WindowsCE

Befindet sich das Visualisierungsgerät ("VEPxx.4") im gleichen Netzwerk wie der Entwicklungs-PC, können Sie mit Hilfe des Skriptes "Stapelverarbeitungsprogramm zum Kopieren der Datei OPCServer.ini auf ein VEPxx.4" auf Seite 168, die auf den PC aktive OPC.DA Server Konfigurationsdatei "OPC-Server.ini" übertragen.

Passen Sie hierzu die IP-Adresse in der folgenden Zuweisungszeile an:

"SET IPADDR_VEP=192.168.1.11"

Stapelverarbeitungsprogramm zum Kopieren der Datei "OPCServer.ini" auf ein "VEPxx.4"

```
@ECHO OFF
CLS
REM
REM Copy IndraLogic OPC Server Configuration File.to VEPxx.4
REM
REM
REM Set IP Address of the VEP Device
SET IPADDR_VEP=192.168.1.11

SET REGKEY=HKLM\Software\Rexroth\IndraWorks
SET REGKEYNAME=CurrentIndraLogicOPCDALogFileName

REM 'delims' is set with TAB and SPACE
FOR /F "tokens=2* delims=" " %a IN ( 'REG QUERY "%REGKEY%" /v %REGKEYNAME%' ) DO SET OPCFILE="%b"

IF EXIST "%OPCFILE:="%" GOTO filecopy

ECHO.
ECHO !!! ERROR !!!
ECHO - File
ECHO %OPCFILE%
ECHO does not exist!
ECHO.

PAUSE
GOTO :EOF

:filecopy
```

```

REM Create FTP command file
SET FTPCMDFILE="%~dp0FtpCmd.txt"
SET VEPDSTDIR=/StorageCard/Rexroth/Controls/OPCServer

ECHO open %IPADDR_VEP%> %FTPCMDFILE%
ECHO anonymous>> %FTPCMDFILE%
ECHO 123@456.com>> %FTPCMDFILE%
ECHO cd %VEPDSTDIR%>> %FTPCMDFILE%
ECHO ascii>> %FTPCMDFILE%
ECHO put "%OPCFILE:"=">> %FTPCMDFILE%
ECHO close>> %FTPCMDFILE%
ECHO bye>> %FTPCMDFILE%

ECHO.
ECHO Copy file
ECHO %OPCFILE%
ECHO.
ECHO.
CALL ftp -s:%FTPCMDFILE%

PAUSE

```

Abb. 5-43: Stapelverarbeitungsprogramm zum kopieren der Datei "OPCServer.ini" auf ein "VEPxx.4"

5.6.2 ModbusTCP

Allgemein

Bei der Projektierung kann es sinnvoll sein, erste Tests mit Modbus Test-Clients zu realisieren.

Beispiele für frei im Internet erhältliche Test-Clients sind (Stand 09.2008):

- IPC@CHIP Modbus Client V1.4
- Modbus TCP/IP Request Tool Pro

In den folgenden Kapiteln wird auf die Details der Projektierung einer Modbus TCP Anbindung mit

Details über die Projektierung einer Modbus TCP Anbindung werden in den folgenden Kapiteln erklärt:

- [Projektierung mit Siemens WinCC-flexible 2007, Seite 171](#) und
- [Projektierung mit Pro-face GP-Pro EX 2.2, Seite 174](#)

Adressierung der Daten aus dem SPS-Projekt

Im SPS-Projekt werden die Datenbereiche für Register-Daten und Bit-Daten mittels Pointer an die Instanz des Funktionsbaustein IL_ModbusTCPServer übergeben.



Der Zugriff auf die Datenbereiche des Modbus TCP Server erfolgt über folgende Adressierung durch die Modbus-Clients

Register-Daten-Bereich:

Adressierung der "Holding Register"-Daten mittels **%MW** (lesend und schreibend) oder der "Input Register"-Daten mittels **%IW** (nur lesend).

Bit-Daten-Bereich:

Adressierung der "Coil"-Daten mittels **%M** (lesend und schreibend) oder der "Discret Input"-Daten mittels **%I** (nur lesend).

Bit-Daten

Der Bit-Daten-Bereich wurde im Testprojekt als Array of Byte angelegt, aus dem fortlaufend jedes einzelne Bit adressiert werden kann.

Applikationsbeispiele



Für den Bit-Datenbereich kann kein ARRAY OF BOOL verwendet werden, da in IndraLogic eine Variable vom Typ BOOL 8 Bit Speicher belegt.

Die einzelnen Bits aus einem Byte können aber auf Variablen vom Typ BOOL kopiert werden (und umgekehrt)

Die Adresse einer Variablen aus dem Bit-Daten-Bereich entspricht der Bit-Nummer aus dem Byte-Array, das für die Bit-Daten verwendet wird.

Beispiel:

Bit-Daten (%M)

Adresse **%M0** entspricht der Variablen **BitData[0].0**

Adresse **%M7** entspricht der Variablen **BitData[0].7**

Adresse **%M8** entspricht der Variablen **BitData[1].0**

Adresse **%M12** entspricht der Variablen **BitData[1].4**

Register-Daten

Der Register-Daten-Bereich kann bis zu 65536 Worte groß sein und durch den Anwender beliebig organisiert werden.

Um eine bessere Übersicht über die Register-Daten zu erhalten, wurde im Testprojekt für den Register-Daten-Bereich eine Struktur verwendet, welche Arrays der verschiedenen SPS-Datentypen beinhaltet. Diese Struktur ist so aufgebaut, dass die einzelnen Datentypen-Bereiche immer 1000 Worte lang sind und somit an einer definierten Wort-Adresse beginnen.

Beispiel einer Datenstruktur für RegisterData

Programm:

```
TYPE REGISTER_DATA :
STRUCT
  arWord:   ARRAY[0..999]OF WORD;      (* %MW0000 - %MW0999 *)
  arInt:    ARRAY[0..999]OF INT;        (* %MW1000 - %MW1999 *)
  arUInt:   ARRAY[0..999]OF UINT;       (* %MW2000 - %MW2999 *)
  arDint:   ARRAY[0..499]OF DINT;       (* %MW3000 - %MW3999 *)
  arUdint:  ARRAY[0..499]OF UDINT;      (* %MW4000 - %MW4999 *)
  arReal:   ARRAY[0..499]OF REAL;       (* %MW5000 - %MW5999 *)
  arString: ARRAY[0..99]OF STRING(19); (* %MW6000 - %MW6999 *)
  (* String(19): because 19 byte string + 1 byte end sign *)
  (* together 7000 WORDS *)
END_STRUCT
END_TYPE
```

Text-String

Über den Register-Daten-Bereich können auch Texte (Strings) übertragen werden.



In IndraLogic beträgt die Länge eines Strings (in Byte) immer die angegebene Länge + 1 Byte Ende-Kennung

Um die Strings mit ganzer Wortlänge zu definieren, muss die Länge also immer mit einer ungeraden Byte-Anzahl angegeben werden

(z.B. String(19) => 19 Byte String + 1Byte Endekennung = 20 Byte = 10 Worte)

Die Adresse einer Variablen aus dem Register-Daten-Bereich ergibt sich aus dem Aufbau der Struktur, die für die Register-Daten verwendet wird.

Legt man das zuvor gezeigte "[Beispiel einer Datenstruktur für RegisterData](#)" auf Seite 170 zu Grunde, ergeben sich folgende Adressen:

Beispiel:

Register-Daten (%MW)

Adresse **%MW0** entspricht der Variablen **RegisterData.arWord[0]**

Adresse %MW1000 entspricht der Variablen **RegisterData.arInt[0]**
 Adresse %MW1001 entspricht der Variablen **RegisterData.arInt[1]**
 Adresse %MW5000 entspricht der Variablen **RegisterData.arReal[0]**
 Adresse %MW5002 entspricht der Variablen **RegisterData.arReal[1]**
 Adresse %MW6000 entspricht der Variablen **RegisterData.arString[0]**
 Adresse %MW6010 entspricht der Variablen **RegisterData.arString[1]**

Projektionierung mit Siemens WinCC-flexible 2007

Kommunikationseinstellungen

Zur Konfiguration der Verbindung in WinCC-flexible sind folgende Einstellungen notwendig:

1. Im Projektbaum des Bediengeräts öffnen Sie das Menü **Communication** ► **Connections**
2. Geben Sie den Namen der Verbindung (und optional Kommentar) ein und wählen Sie Communication driver "**Modicon MODBUS TCP/IP**" aus
3. Im darauf erscheinenden Dialog müssen Sie nun die Kommunikationsparameter konfigurieren

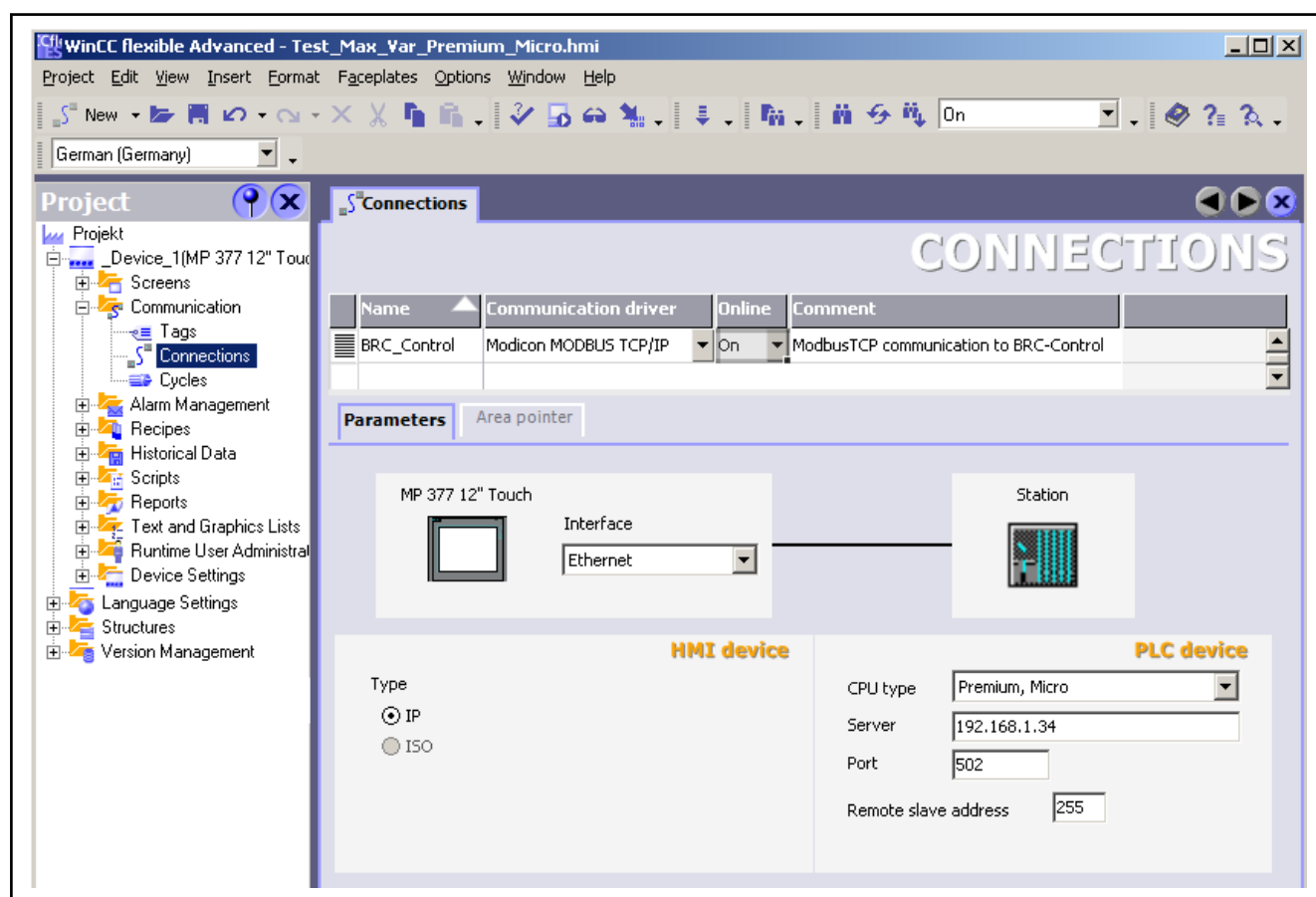


Abb. 5-44: WinCC-flexible Modbus TCP Kommunikationsparameter

Sektion	Name	Wert	Beschreibung
HMI device	Interface	Ethernet	Auswahl der Kommunikationsschnittstelle
	Type	IP	Nur IP unterstützt

Applikationsbeispiele

Sektion	Name	Wert	Beschreibung
PLC device	CPU type	Premium, Micro	Schneider Modicon Steuerungstyp der ohne Adressoffset und mit üblicher Bit-Zählweise arbeitet (siehe Siemens Benutzerhandbuch "WinCC-flexible 2007 Kommunikation Teil 2 - Kapitel 6")
	Server	"192.168.1.34"	IP - Adresse der Steuerung
	Port	"502"	Port der Kommunikationsverbindung (Port der Instanz des Funktionsbausteins IL_ModbusTCPServer)
	Remote slave address	"255"	Standardeinstellung beibehalten (255) ! für Modbus TCP nicht relevant - siehe "MODBUS Messaging on TCP/IP Implementation Guide V1.0b" unter www.Modbus-IDA.org

Tab. 5-4: WinCC-flexible Modbus TCP Kommunikationsparameter

Variablendeklaration

Nachdem die Verbindung konfiguriert wurde, können Sie nun die Variablen deklarieren. Hierzu öffnen Sie im Projektbaum des Bediengeräts das Menü **Communication ▶ Tags**.
Beispiel einer Variablendeklaration in WinCC-flexible 2007:

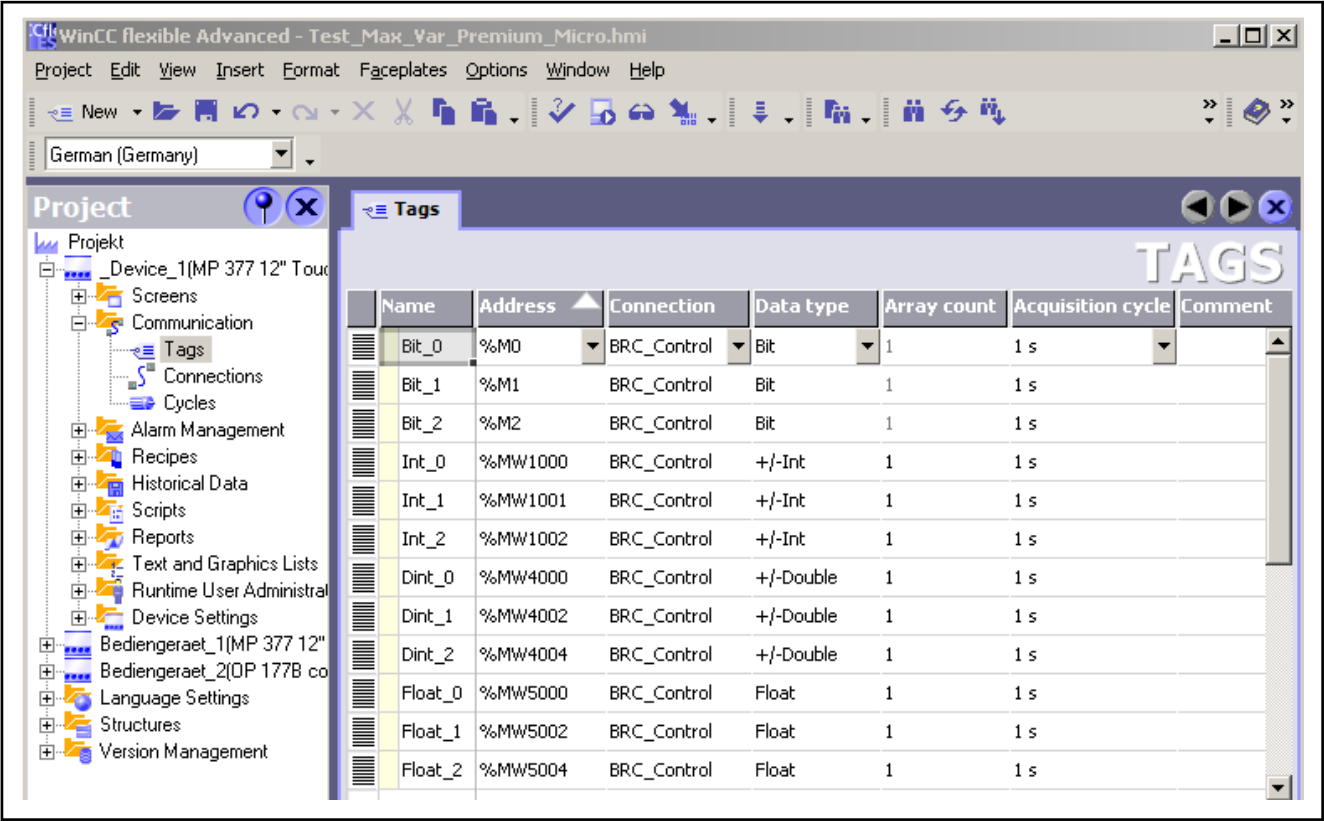


Abb. 5-45: WinCC-flexible Variablendeklaration - Beispiel

WinCC - flexible Variablendeklaration

Eingabefeld	Beispiel	Beschreibung
Name	"Bit_0"	Name der Variable
Address	"%M0"	Modbus Adresse der Variable ("Adressierung der Daten aus dem SPS-Projekt" auf Seite 169)

Applikationsbeispiele

Eingabefeld	Beispiel	Beschreibung
Connection	"BRC_Control"	Name der Verbindung
Data type	"Bit"	WinCC-flexible Datentyp der Variable (Tab. 5-6 "WinCC-flexible Datentypen / IndraLogic Datentypen" auf Seite 173)
Array count	"1"	Anzahl der Array-Elemente falls die Variable als Array angelegt werden soll
Acquisition cycle	"1s"	Aktualisierungsrate in der der Wert der Variablen aus der Steuerung gelesen werden soll

Tab. 5-5: WinCC - flexible Variablendeklaration

In WinCC-flexible 2007 stehen verschiedene Datentypen zur Verfügung. Die entsprechenden IndraLogic-Datentypen sind nachfolgend aufgelistet.

WinCC - flexible Datentypen

WinCC - flexible	IndraLogic	Beschreibung
Int	UINT	Unsigned Integer Länge 16 Bit Wertebereich: 0 bis 65535
+/-Int	INT	Signed Integer Länge 16 Bit Wertebereich: -32768 bis 32767
Double	UDINT	Unsigned Double Integer Länge 32 Bit Wertebereich: 0 bis 4294967295
+/-Double	DINT	Signed Double Integer Länge 32 Bit Wertebereich: -2147483648 bis 2147483647
Float	REAL	Floating Point Länge 32 Bit Wertebereich: 1.175494351e-38F bis 3.402823466e+38F
Bit	einzelnes Bit eines Byte	Single Bit Länge 1 Bit ("Bit-Daten" auf Seite 169)
16 Bit Group	WORD	WORD Länge 16 Bit Achtung: Bei 16-Bit-Gruppen ist die Bitzählweise in WinCC-flexible umgekehrt zur Zählweise in IndraLogic
ASCII	STRING	Die Länge des Strings (in Byte) muss in WinCC-flexible über die Tag Properties konfiguriert werden. ("Text-String" auf Seite 170)

Tab. 5-6: WinCC-flexible Datentypen / IndraLogic Datentypen

Die deklarierten Variablen können Sie nun für die verschiedenen Funktionalitäten von WinCC-flexible verwenden.

Applikationsbeispiele

Beispiel:

Anzeige und Eingabe von verschiedenen Variablen

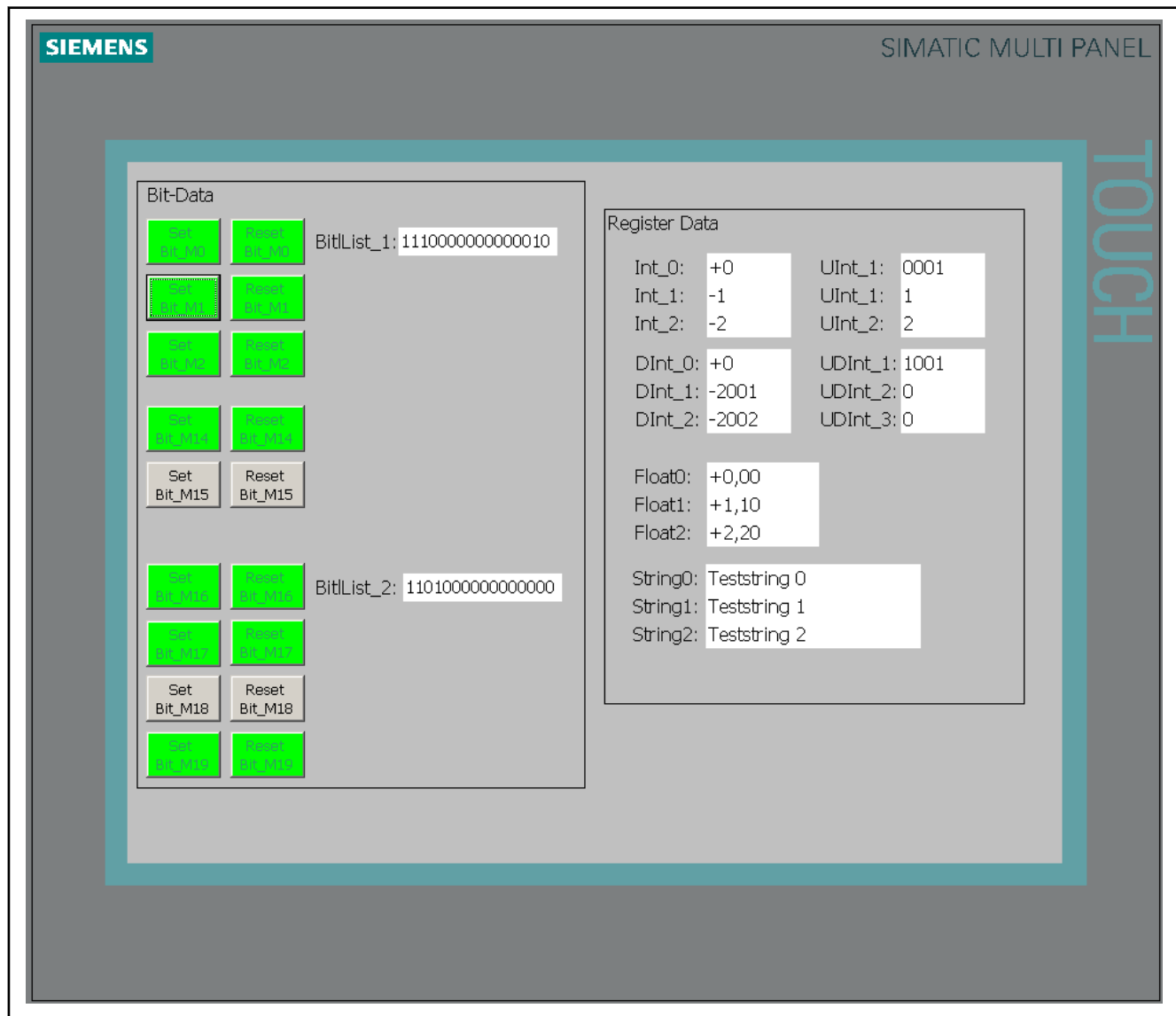


Abb. 5-46: WinCC-flexible - Visualisierungs Beispiel

Projektierung mit Pro-face GP-Pro EX 2.2

Kommunikationseinstellungen

Zur Konfiguration der Verbindung in Pro-face GP-Pro EX 2.2 sind folgende Einstellungen notwendig:

1. Wählen Sie im Menü **System Settings** den Eintrag **"Device/PLC"**
2. Führen Sie im Fenster des Arbeitsbereichs den Link **"Add Device/PLC"** aus
3. Nehmen Sie im darauf erscheinenden Dialog "Add Device/PLC" die folgenden Einstellungen vor:

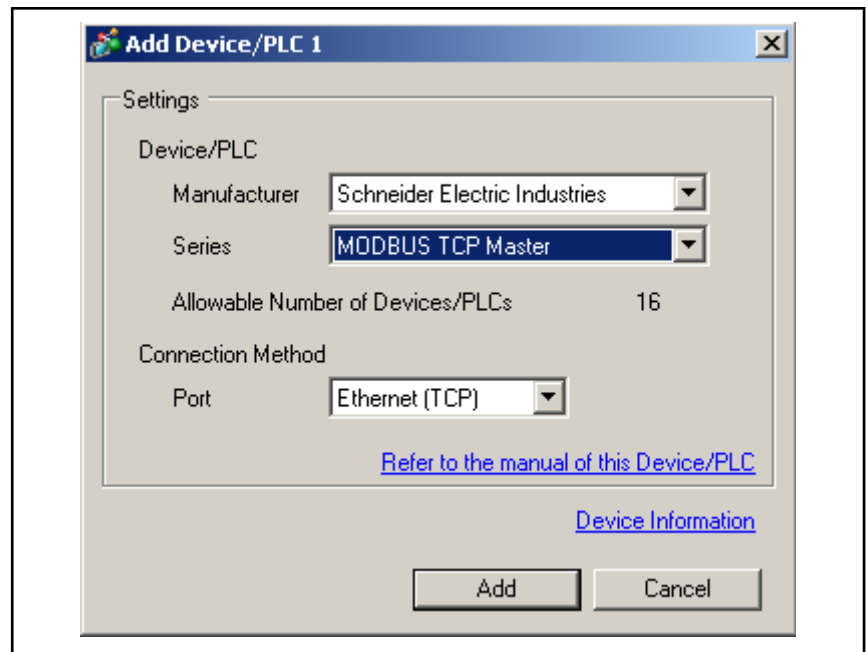


Abb. 5-47: GP-Pro EX Kommunikationseinstellungen - Geräteauswahl

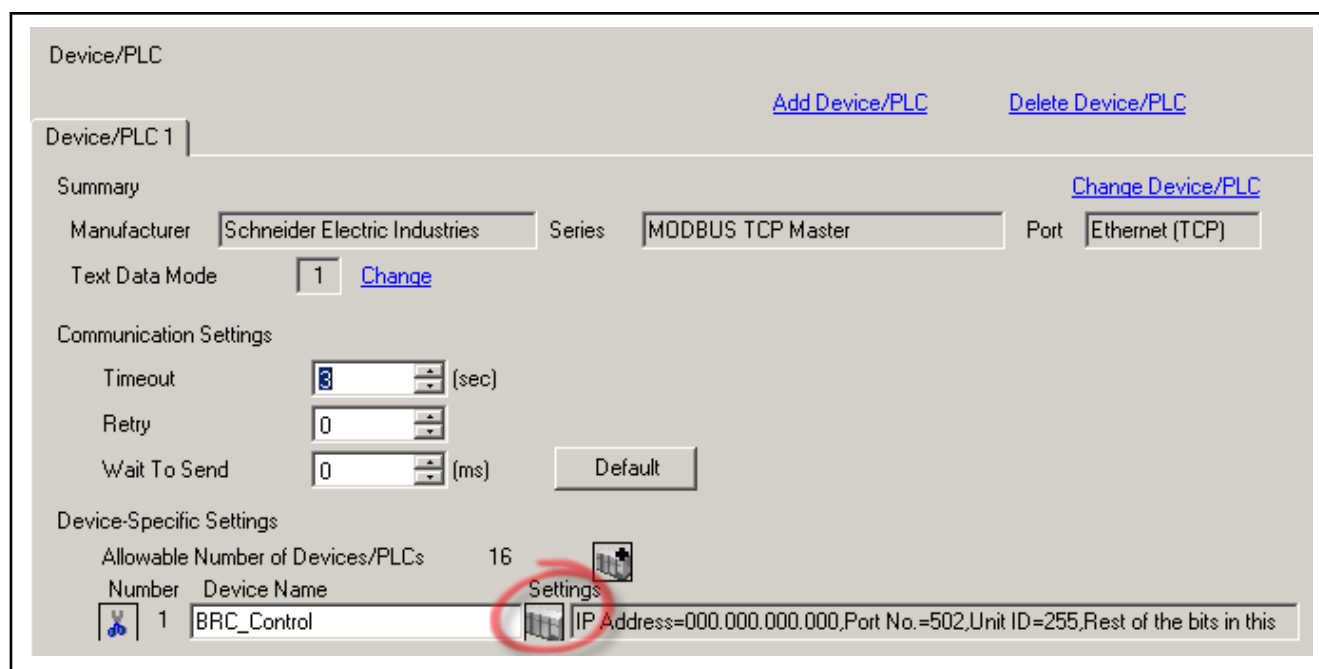
Auswahlfeld	Auswahl
Manufacturer	Schneider Electric Industries
Series	MODBUS TCP Master
Port	Ethernet (TCP)

Tab. 5-7: Kommunikationseinstellungen - Geräteauswahl

Bestätigen Sie die Auswahl mit **Add**

4. Tragen Sie nun im erweiterten Fenster des Arbeitsbereichs unter "Device Name" den Namen für die Verbindung ein und betätigen die Schaltfläche  **"Settings"**

Applikationsbeispiele



Device/PLC

[Add Device/PLC](#) [Delete Device/PLC](#)

Device/PLC 1

Summary [Change Device/PLC](#)

Manufacturer: Schneider Electric Industries Series: MODBUS TCP Master Port: Ethernet (TCP)

Text Data Mode: 1 [Change](#)

Communication Settings

Timeout: 3 (sec)

Retry: 0

Wait To Send: 0 (ms) [Default](#)

Device-Specific Settings

Allowable Number of Devices/PLCs: 16

Number	Device Name	Settings
1	BRC_Control	Settings IP Address=000.000.000.000,Port No.=502,Unit ID=255,Rest of the bits in this

Abb. 5-48: GP-Pro EX Kommunikationseinstellungen - Arbeitsbereich

- Im anschließend erscheinenden Dialog "Individual Device Settings" nehmen Sie im Register "Equipment Configuration" die folgenden Einstellungen vor:

Abb. 5-49: GP-Pro EX Kommunikationseinstellungen - Geräteparameter

Sektion	Name	Wert	Beschreibung
Equipment Address	IP Address	"192.168.1.34"	IP - Adresse der Steuerung
	Type	"502"	Port der Kommunikationsverbindung (Port der Instanz des Funktionsbausteins IL_ModbusTCPServer)
	Unit ID	"255"	Standardeinstellung beibehalten (255) ! für Modbus TCP nicht relevant - siehe "MODBUS Messaging on TCP/IP Implementation Guide V1.0b" unter www.Modbus-IDA.org
Bit manipulation to Holding Register	Reset of the bits in this word	"Do not clear"	Standardeinstellung beibehalten (Do not clear)

Applikationsbeispiele

Sektion	Name	Wert	Beschreibung
IEC61131 Syntax	Address Mode	"0-based (Default)"	IEC61131 Syntax aktivieren und für Address Mode die Standardeinstellung beibehalten (0-based)
Variables	Double Word word order	"Low word first (L/H)"	Anordnung der Worte innerhalb eines Doppelwortes

Tab. 5-8: GP-Pro EX Kommunikationseinstellungen - Geräteparameter

Im Register "MaxQuery" kann die maximale Anzahl der Daten, die in eine Modbus-Anfrage gepackt werden, definiert werden. Hier können Sie die Standardeinstellungen beibehalten.

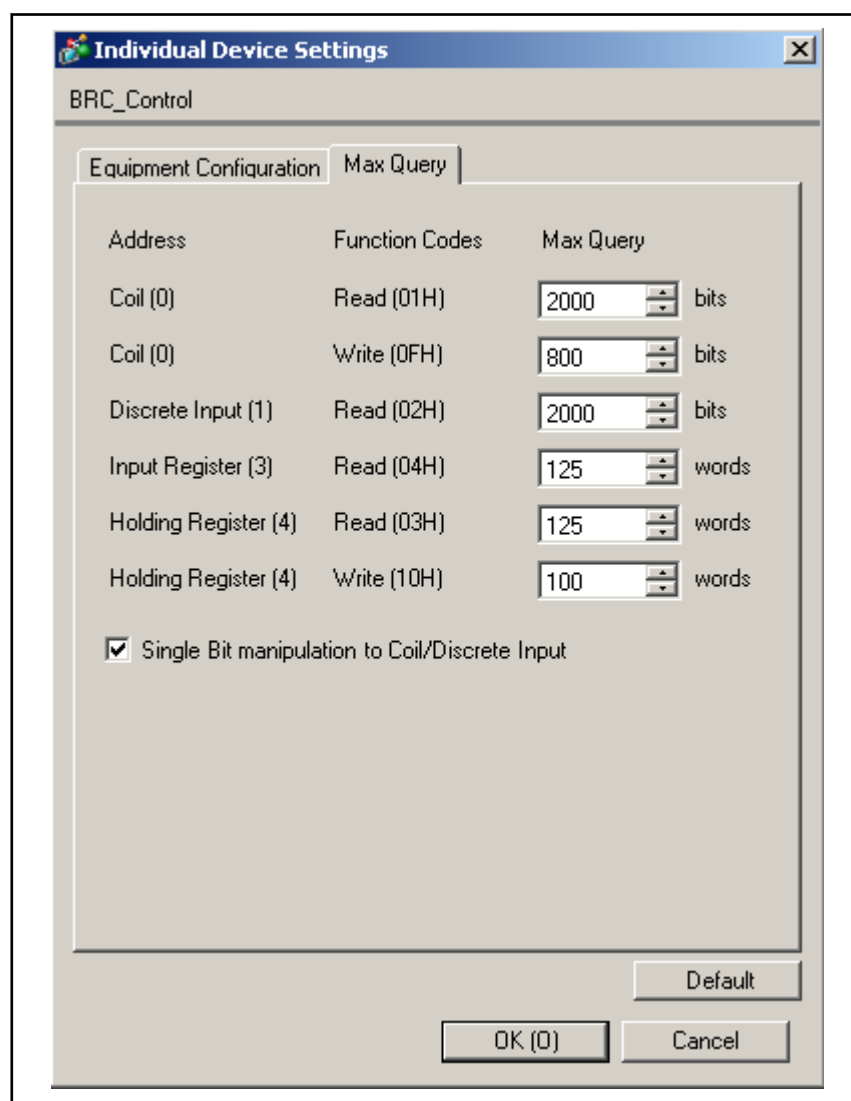


Abb. 5-50: GP-Pro EX Kommunikationseinstellungen - Geräteparameter

Auswahl mit <OK> bestätigen.

- Nun müssen Sie noch den "Text Data Mode" zur Interpretation von Texten anpassen. Hierzu betätigen Sie im Fenster des Arbeitsbereichs den Link **"Change"**.

Device/PLC

[Add Device/PLC](#) [Delete Device/PLC](#)

Device/PLC 1

Summary [Change Device/PLC](#)

Manufacturer: Schneider Electric Industries Series: MODBUS TCP Master Port: Ethernet (TCP)

Text Data Mode: 2 [Change](#)

Communication Settings

Timeout: 3 (sec)

Retry: 0

Wait To Send: 0 (ms) [Default](#)

Device-Specific Settings

Allowable Number of Devices/PLCs: 16

Number	Device Name	Settings
1	BRC_Control	IP Address=000.000.000.000,Port No.=502,Unit ID=255,Rest of the bits in this

Abb. 5-51: GP-Pro EX Kommunikationseinstellungen - Arbeitsbereich
Selektieren Sie im darauf erscheinenden Dialog "Change Text Data Mode" die Auswahl 2 und bestätigen Sie sie mit <Change>.

Change Text Data Mode

Select a text data mode from the following list.

Data in Device Addresses	Bytes LH/HL storage	Double-word LH/HL storage	Text Data Mode	Select
Store from Top Data	LH Order	LH Order	4	☐
		HL Order	2	☒
	HL Order	LH Order	5	☐
		HL Order	1	☐
Store from Last Data	LH Order	LH Order	6	☐
		HL Order	7	☐
	HL Order	LH Order	8	☐
		HL Order	9	☐

[Change](#) [Cancel](#)

Abb. 5-52: Dialog: Change Text Data Mode (GP-Pro EX Kommunikationseinstellungen)

Verwendung der Adressen

Nachdem die Verbindung konfiguriert wurde, können Sie nun in den verschiedenen Funktionalitäten von GP-Pro EX die Adressen direkt verwenden, siehe "Adressierung der Daten aus dem SPS-Projekt" auf Seite 169.

Applikationsbeispiele

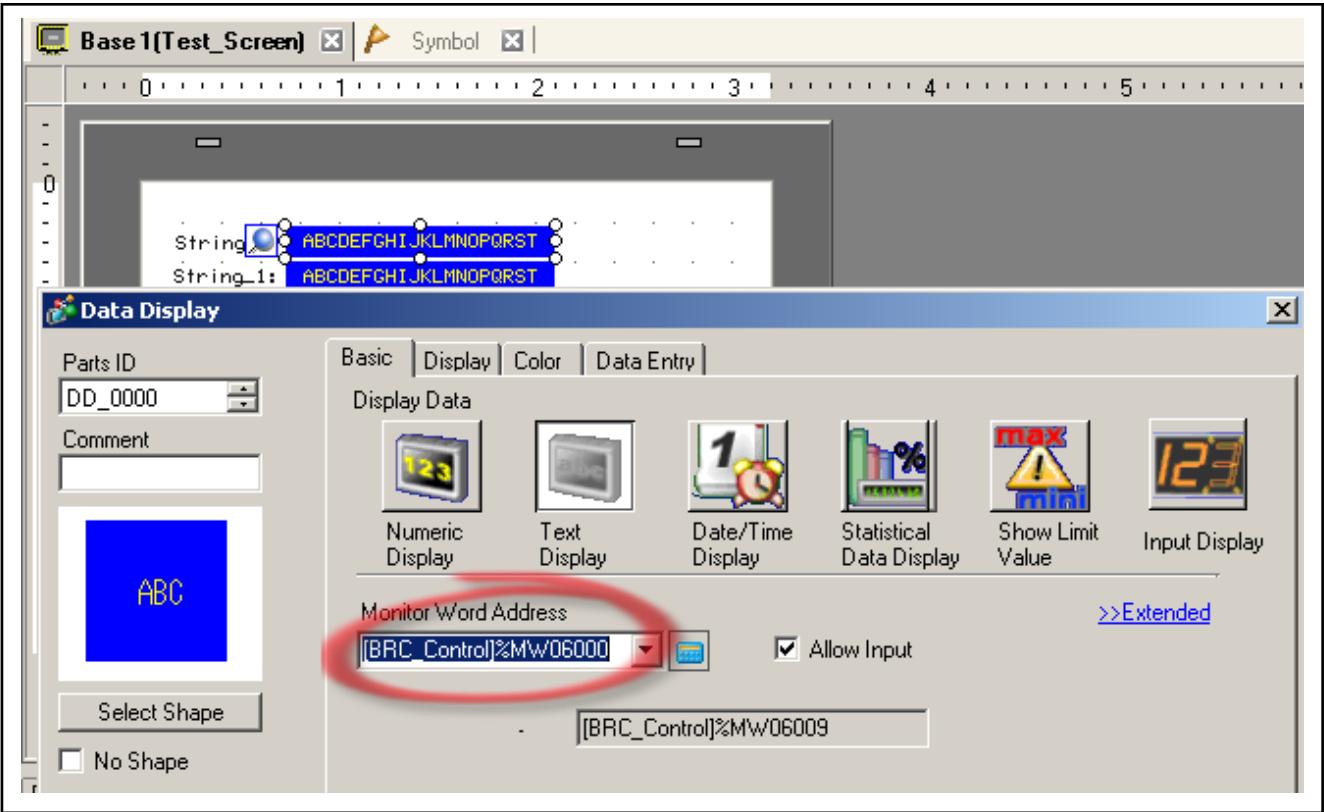


Abb. 5-53: GP-Pro EX - direkte Verwendung der Adressen

Variablendeklaration

Alternativ besteht auch die Möglichkeit symbolische Variablen vom Typ Bitaddress und Wordaddress zu deklarieren, siehe ["Adressierung der Daten aus dem SPS-Projekt" auf Seite 169](#). Über diese symbolischen Variablen können Sie dann die Inhalte der Adressen in den verschiedenen Funktionalitäten von GP-Pro EX verwenden.

Öffnen Sie über das Menü **Common Settings ▶ Symbol Variables** im Arbeitsbereich das Fenster zur Deklaration von Symbol Variablen:

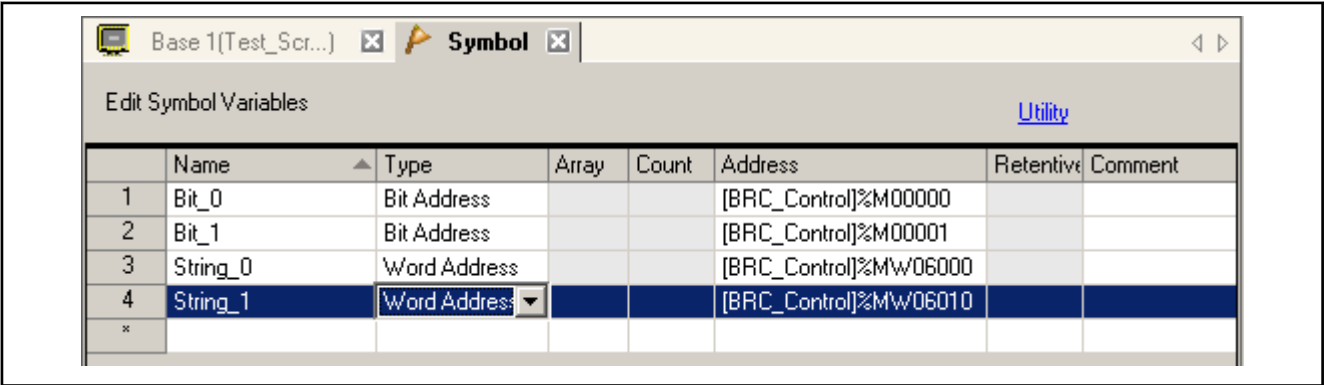


Abb. 5-54: GP-Pro EX - Symbol Variable anlegen

Nachdem die Symbol-Variable angelegt wurde können Sie diese als "Platzhalter" für die Adresse verwenden.



Abb. 5-55: GP-Pro EX - Verwendung der Symbol Variablen



Die Anzahl der zu interpretierenden Register wird über den Datentyp bestimmt, der in den einzelnen Anzeigefeldern verwendet wird.

Bei Anzeigefeldern vom Typ "Text Display" wird die Anzahl der zu interpretierenden Register über die Anzahl darzustellender Zeichen bestimmt.

Beispiel:

Anzeige und Eingabe von verschiedenen Variablen

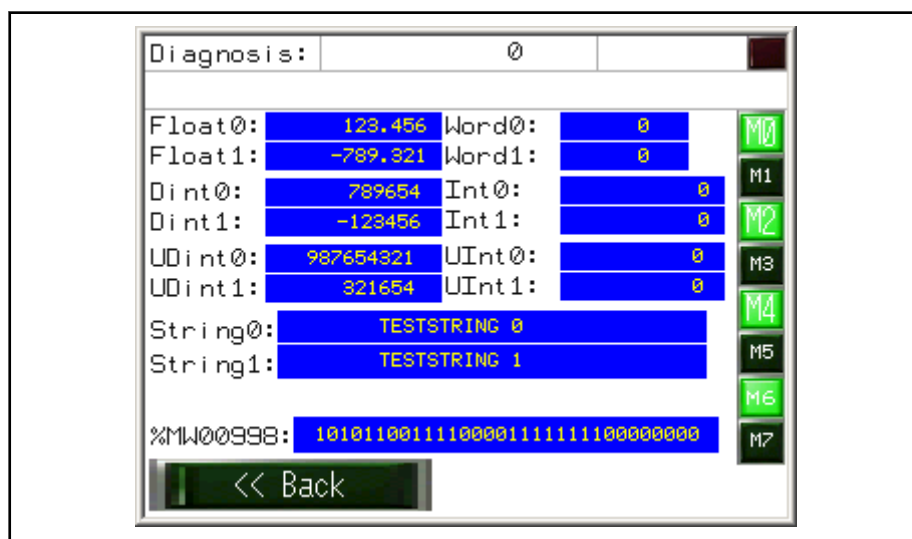


Abb. 5-56: Variablenanzeige

6 Client-Entwicklung

6.1 OPC Client Visual Basic .NET

6.1.1 Allgemein

Dieser Abschnitt beschäftigt sich mit der Entwicklung des "Visual Basic .NET" OPC Clients mit dem "Microsoft Visual Studio .NET 2005".

Er umfasst folgende Themen:

- Projekt erstellen
- Verweise hinzufügen
- Windows Forms und Steuerelemente hinzufügen
- OPC Client Implementierung

Die folgende Abbildung zeigt den OPC Client als aktive Windows Applikation.

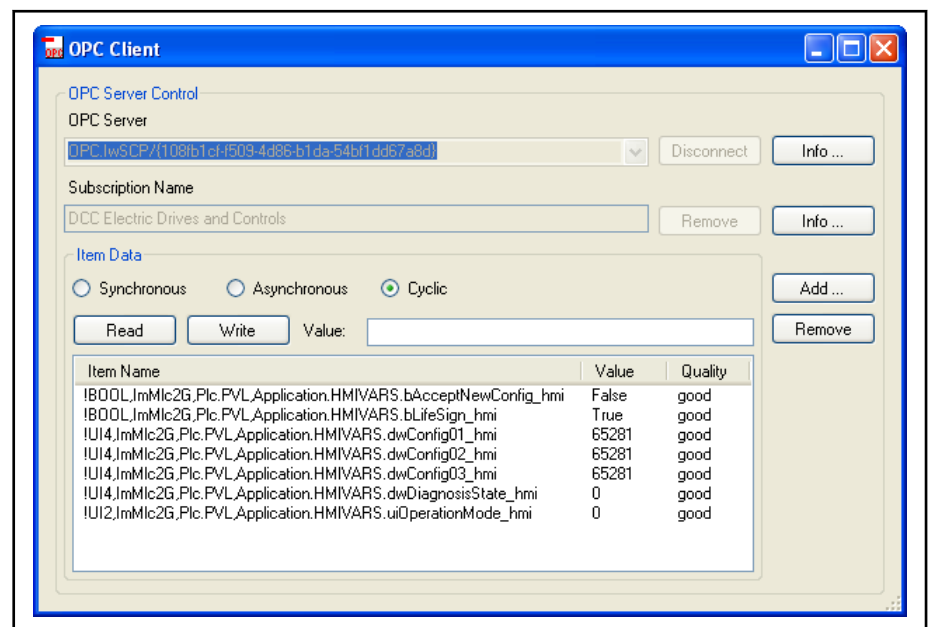


Abb. 6-1: OPC Client Oberfläche

6.1.2 Visual Basic Projekt "OPC Client"

Neues Projekt

Um ein neues Projekt in der Entwicklungsumgebung "Microsoft Visual Studio .NET 2005" zu erstellen, betätigen Sie entweder direkt auf der Startseite den Link "Neues Projekt" oder klicken Sie im Menü **Datei** ► **Neu** auf den Menüpunkt **Projekt**....

Es erscheint der Dialog "Neues Projekt", wie in nachstehender Abbildung dargestellt.

Hier wählen Sie Projekttyp und Vorlage aus:

- "Andere Sprachen \ Visual Basic \ Windows" oder
- "Visual Basic \ Windows" und als Vorlage
- "Windows Application"

Geben Sie als Projektname "OPCCient" ein und passen Sie den vorgeschlagenen Speicherort entsprechend an.

Client-Entwicklung

Mit Betätigen der Schaltfläche **OK** wird das Projekt erstellt.

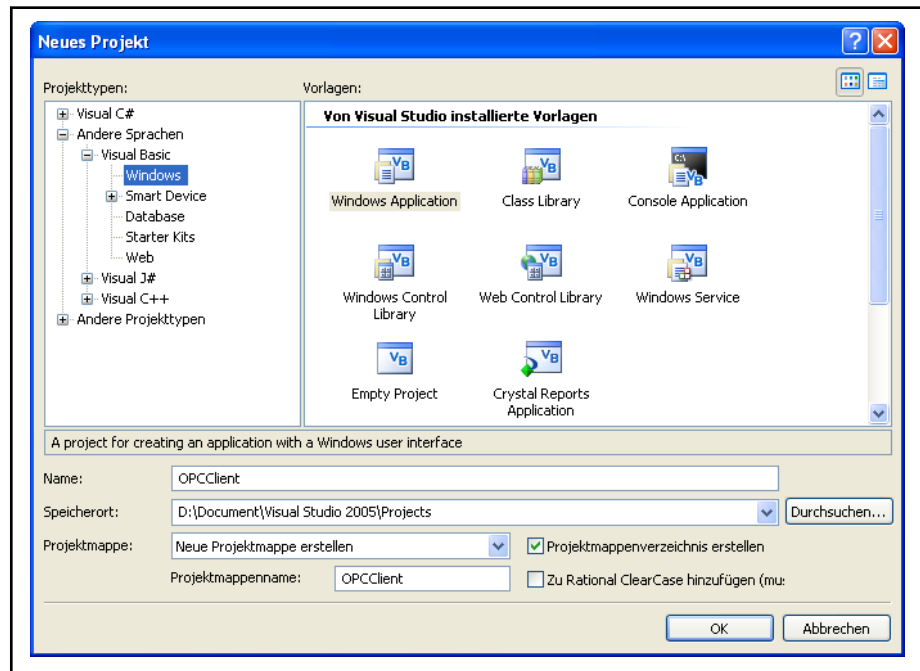


Abb. 6-2: Dialog: Neues Projekt

Verweise hinzufügen

Damit Sie auf die Funktionalitäten der OPC .NET API zugreifen können, muss ein Verweis auf die "OpcNetApi.dll" und "OpcNetApi.Com.dll" im Projekt vorhanden sein.

Die Verwaltung von "Referenzen" ist auf der "Referenz"-Seite der Projekt-Eigenschaften möglich. Das Hinzufügen von weiteren Abhängigkeiten erfolgt über den Dialog "Referenz hinzufügen", der mit dem Betätigen der "Add..."-Schaltfläche aufgerufen wird. Im Dialog "Referenz hinzufügen" fügen Sie die beiden Dateien "OpcNetApi.dll" und "OpcNetApi.Com.dll" auf der Registerkarte "Durchsuchen" dem Projekt hinzu.

Beide Dateien werden bei der Installation der "IndraWorks Suite" mitgeliefert und sind im Installationsordner (z.B. "**C:\Programme\Rexroth\IndraWorks**") gespeichert.

Mit Öffnen der Dateien werden diese in der Liste "Referenzen:" übernommen.

In der Liste "Imported namespaces:" sind alle aktuell zur Verfügung stehenden Namensräume für das Projekt aufgelistet.

Alle ausgewählten Namensräume können Sie im Projekt ohne die explizite Deklaration mittels "Imports"-Statement verwenden.

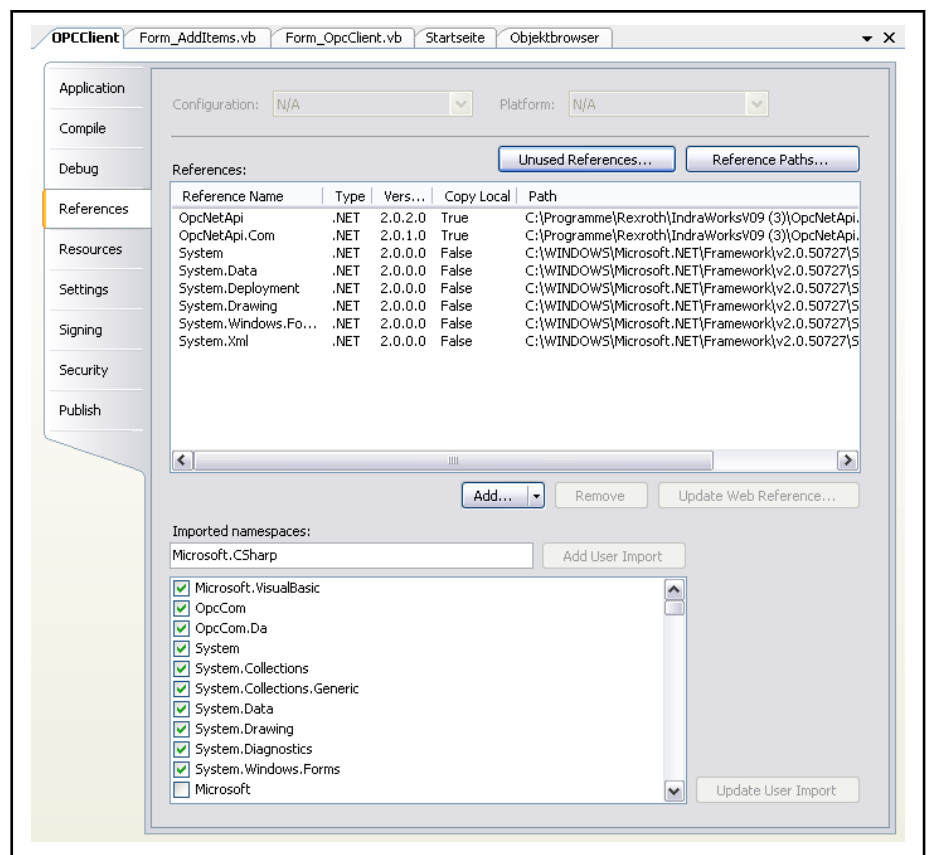


Abb. 6-3: Projekt-Eigenschaften - Seite Referenz

Windows Form und Steuerelement hinzufügen

Das Hinzufügen einer Windows Form in das Startprojekt erfolgt über den Menüpunkt **Projekt ► Windows Form hinzufügen....**

Im Dialog "Neues Element hinzufügen - OPCClient" ist die Vorlage "Windows Form" bereits ausgewählt und Sie müssen nur noch den Namen der Form eingeben.

Bei allen im OPC Client Beispiel erstellten Windows Forms wurde dem Namen der Bezeichner "Form_" vorangestellt.

Die nachfolgende Abbildung zeigt eine Übersicht des Projekts "OPC Client" im "Projektmappen-Explorer".

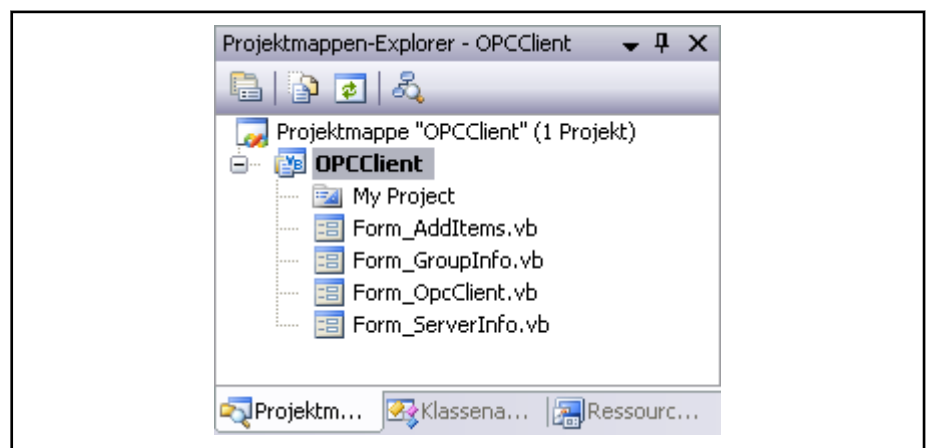


Abb. 6-4: Projektübersicht

Client-Entwicklung

Das Entwerfen von "Windows Forms" führen Sie mit Hilfe des "Windows Forms Designer" und den in der "Toolbox" zur Verfügung stehenden Steuerelementen durch. Hierbei wählen Sie die Steuerelemente per Mausklick aus und platzieren sie auf der Form.

Die Konfiguration der Steuerelemente für das ausgewählte Objekt erfolgt im Fenster "Eigenschaften".

Außerdem können Sie die Größe und Position der Steuerelemente mit der Maus im Windows Forms Designer zu verändern.

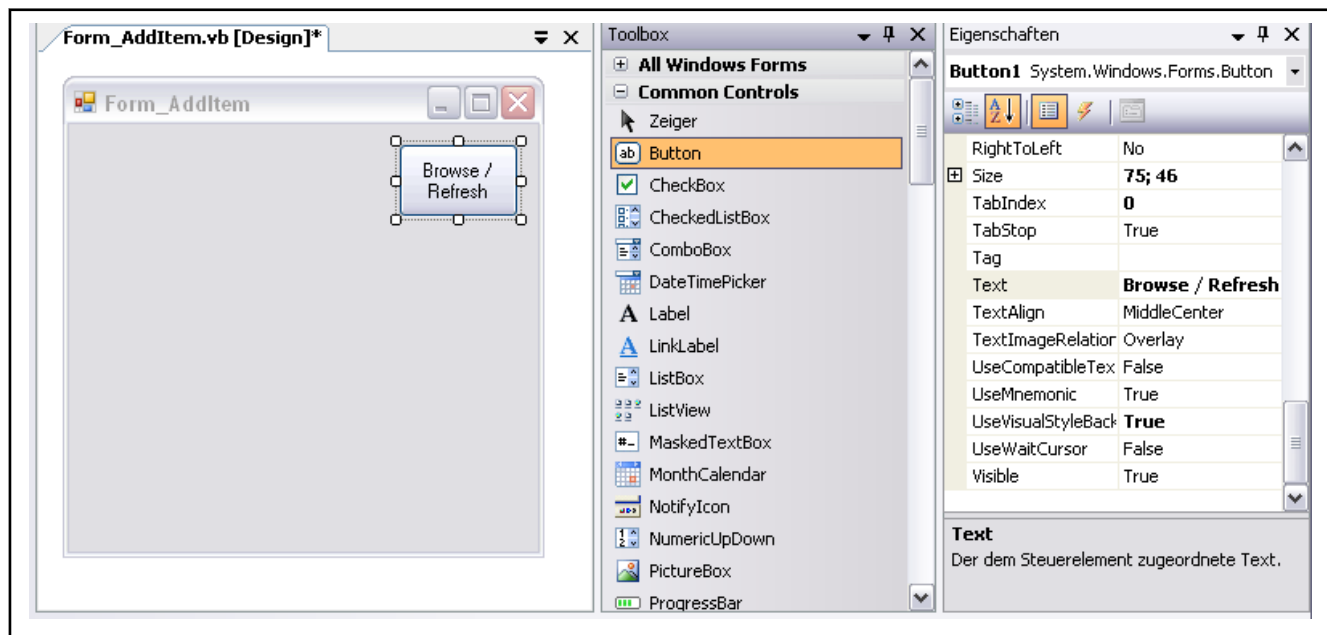


Abb. 6-5: Hinzufügen und Konfigurieren von Steuerelementen

Die nachstehende Abbildung zeigt die Form "OPC Client" im "Windows Forms Designer".

Auf eine detaillierte Beschreibung der Form und deren einzelne Steuerelemente wird aus Gründen der Übersichtlichkeit verzichtet. Alle konfigurierten Eigenschaften können dem Beispielprojekt entnommen werden.

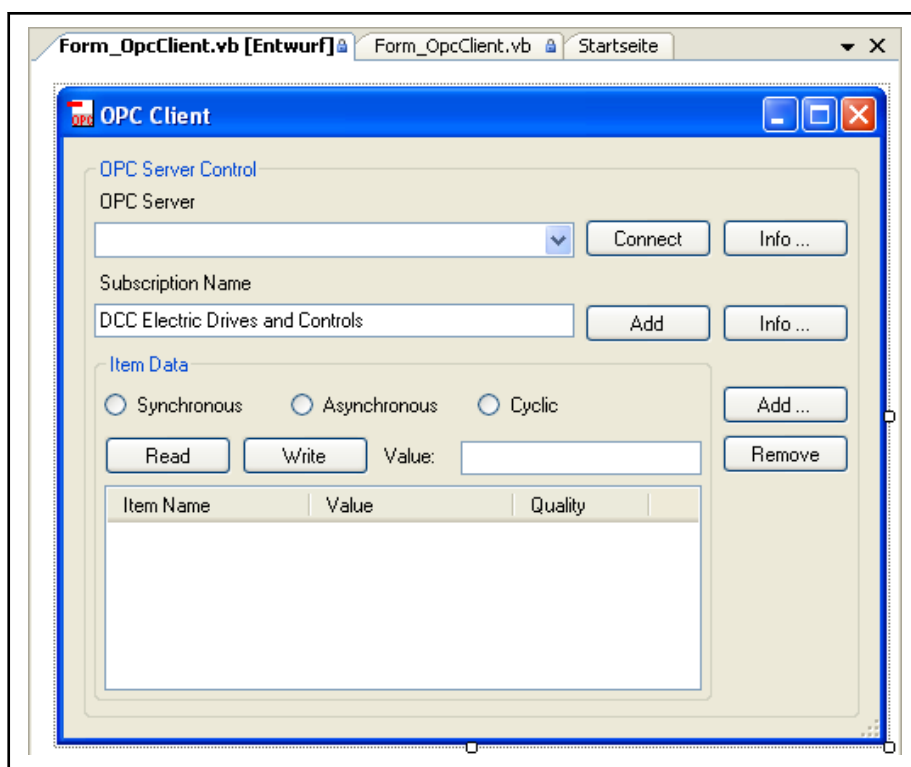


Abb. 6-6: Dialog: OPC Client (Windows Forms Designer)

OPC Client Implementierung

Allgemein

Im folgenden Abschnitt wird die Vorgehensweise bei der Programmierung des OPC Clients beschrieben.

Zur besseren Übersicht im Programm, wurden einzelne Bereiche in der Klasse mit der `"#Region"`-Direktive zusammengefasst. Damit können Sie diese Bereiche bei der Programmerstellung zur besseren Übersichtlichkeit auf-, bzw. zugeklappen.

Die nachstehende Abbildung zeigt am Beispiel der Form "Form_OpcClient" die definierten Bereiche.

Bereich	Beschreibung
"#Region" - "#End Region"	
"Declaration"	Enthält die Deklaration der Klasse (Member Variablen und Konstanten).
"Form Events"	Prozeduren zur Behandlung der von der Form und den enthaltenen Steuerelementen ausgelösten Ereignisse.
"OPC Events"	Prozeduren zur Behandlung der vom OPC Server ausgelösten Ereignisse.
"Additional Internal Functions"	Zusätzliche interne Definitionen, Strukturen und Funktionen zur Optimierung der Benutzerführung.

Tab. 6-1: Programm-Strukturierung mittels `"#Region"`-Direktiven

Auch die relevanten Programmzeilen im Beispielprojekt sind mit dem Schlüsselwort **"Step:"** und einer fortlaufenden Nummerierung gekennzeichnet. D.h., mit der im Visual Studio enthaltenen Suche "In Dateien Suchen" (<Strg>

Client-Entwicklung

+<Umschalt>+<F>) kann durch die Eingabe des Suchbegriffs, z.B. **"Step: 2"**, direkt zu der beschriebenen Implementierung verzweigt werden.

Die im weiteren beschriebene Implementierung wurde in Anlehnung an die Bedienreihenfolge der Beispielapplikation "OPC Client" dokumentiert.

Die Gliederung ist in der folgenden Abbildung dargestellt:

Kapitelüberschrift	Beschreibung
Kap. "Deklaration der Variablen" auf Seite 188	Deklaration der Variablen für die Verwaltung des OPC Servers und der Windows Form.
Kap. "OPC Server auflisten" auf Seite 189	Auflistung aller auf dem lokalen PC installierten OPC Server.
Kap. "OPC Server Instanz erstellen und löschen" auf Seite 190	Erstellen und Löschen der OPC Server Instanz und Anzeige der OPC Server Information.
Kap. "OPC Subscription erstellen und löschen" auf Seite 191	Erstellen und Löschen einer OPC Subscription und Anzeige der Subscription Information.
Kap. "Steuerungselemente auflisten" auf Seite 194	Lesen und Anzeigen aller Items der Steuerung (browsen).
Kap. "OPC Items hinzufügen und löschen" auf Seite 196	Hinzufügen und Löschen von OPC Items.
Kap. "Steuerungsdaten lesen" auf Seite 199	Lesen der Steuerungsdaten "Synchron" und "Asynchron".
Kap. "Steuerungsdaten schreiben" auf Seite 202	Schreiben der Steuerungsdaten "Synchron" und "Asynchron".
Kap. "Zyklisches Lesen der Steuerungsdaten" auf Seite 204	Zyklisches Lesen der Steuerungsdaten.

Tab. 6-2: Übersicht zur Gliederung der Implementation

Deklaration der Variablen

Im Bereich Deklaration **"Step: 1"** sind die Variablen der "Form_OpcClient" deklariert. Für die Namen wurde die Syntax so gewählt, dass alle Klassenvariablen mit dem Präfix "m_" ("Member Variablen") und die Konstanten-Definitionen in "GROßBUCHSTABEN" geschrieben werden.

Bei den Variablen handelt es sich um

- das Objekt für die OPC Server Instanz und
- Variablen für die Verwaltung der Form.

Die Zeile **"Dim m_OpcServer As Opc.Da.Server = Nothing"** ist die für das Beispiel relevante Deklaration. In der Variablen "m_OpcServer" wird das erstellte OPC Server Objekt gehalten.

```

' Step: 1
'
' declaration of members used for opc administration
#Region " Declaration "
'
' Step: 1
'
' declaration of members used for opc server administration
Dim m_OpcServer As Opc.Da.Server = Nothing
'
'
' members to store configuration
Dim m_DialogAddItems As OPCClient.Form_AddItems
Dim m_InternalStateControl As InternalStateControl
'
'
' constant values
Const BUTTON_DESCRIPTION_SERVER_CONNECT As System.String = "Connect"
Const BUTTON_DESCRIPTION_SERVER_DISCONNECT As System.String = "Disconnect"
Const BUTTON_DESCRIPTION_GROUP_ADD As System.String = "Add"
Const BUTTON_DESCRIPTION_GROUP_REMOVE As System.String = "Remove"
Const MAX_ITEMS As System.Int32 = 10
'
#End Region

```

Abb. 6-7: Deklaration der Klassenvariablen

OPC Server auflisten

Die Auflistung aller lokal auf dem PC installierten OPC Server erfolgt im "Load Ereignis" **"Step: 2"** der Form "Form_OpcClient". Es werden alle OPC Server über die Methode "opcDiscovery.GetAvailableServers()" der Discovery Schnittstelle "Opc.IDiscovery" des Dienstes "OpcEnum.exe" abgefragt, die die Spezifikation "Data Access 2.0" unterstützen. Welche OPC Spezifikationen zur Auswahl stehen, ist in der Struktur "Opc.Spezifikation" definiert.

Die durch das Discovery Interface bereitgestellte Liste enthält den vollständigen URL-Pfad mit ProgID (ohne Versionsindex) und GUID zur eindeutigen Identifikation des Servers.

```

' Step: 2
'
' browse for available opc servers on this system
Private Sub FormOpcClient_Load(ByVal sender As Object, ByVal e As System.EventArgs)
'
'
' Step: 2
'
' browse for available opc servers on this system
opcDiscovery = New OpcCom.ServerEnumerator
opcServers = opcDiscovery.GetAvailableServers( _
    Opc.Specification.COM_DA_20)
'
' copy opc server to ...
Me.ComboBox_Server.BeginUpdate()
'
For Each opcServer In opcServers
' add all server to the combo box
Me.ComboBox_Server.Items.Add(opcServer.Url.Path)
Next
'
Catch ex As Exception
'

```

Abb. 6-8: Installierte OPC Server auflisten

OPC Server Instanz erstellen und löschen

Allgemein

In diesem Abschnitt wird das Erzeugen und Löschen der OPC Server Instanz sowie das Auslesen der Server Informationen beschrieben. Die Verwaltung der OPC Server Instanz erfolgt im Ereignishandler "Button_Connect_Disconnect_Click" der Schaltfläche "**Connect**" bzw. "**Disconnect**" und das Auslesen und Anzeigen der Server Informationen in der Funktion "Button_ServerInfo_Click" der Schaltfläche "**Info ...**".

Die relevanten Codesegmente im OPC Client Beispielprojekt sind:

- **"Step: 3"** - OPC Server Instanz erzeugen
- **"Step: 4"** - OPC Server Instanz löschen
- **"Step: 5"** und **"Step: 21"** - Server Informationen lesen

OPC Server Instanz erzeugen - Connect

Das Erzeugen einer OPC Server Instanz und das Herstellen der Verbindung ist im **"Step: 3"** beschrieben (siehe auch nachfolgende Grafik).

Im ersten Schritt wird eine OPC Server Instanz mit "New Opc.Da.Server" und Übergabe des Parameters "OpcCom.Factory" sowie der "Opc.URL" erzeugt.

Die "Opc.URL" enthält Informationen über die verwendete OPC Spezifikation "Opc.UrlScheme.DA", den PC-Namen "localhost" oder die IP-Adresse auf dem der Server gestartet werden soll, sowie die Serverkennung.

Im Beispiel wird die Serverkennung aus dem Kombinationslistenfeld "Combo-Box_Server" ausgelesen.

Für die Angabe der Serverkennung sind die folgenden Varianten möglich:

- **"OPC.lwSCP"**
ProgID ohne Versionsindex
- **"OPC.lwSCP.1"**
ProgID mit Versionsindex
- **"OPC.lwSCP/{108fb1cf-f509-4d86-b1da-54bf1dd67a8d}"**
ProgID ohne Versionsindex und GUID
- **"OPC.lwSCP.1/{108fb1cf-f509-4d86-b1da-54bf1dd67a8d}"**
ProgID mit Versionsindex und GUID

Mit dem Aufruf der Funktion "Me.m_OpcServer.Connect()" wird der OPC Server instanziiert.

OPC Server Instanz löschen - Disconnect

Das Löschen der OPC Server Instanz ist im **"Step: 4"** dargestellt.

Mit dem Aufruf der Funktion "Me.m_OpcServer.Disconnect()" erfolgt die Abmeldung am OPC Server. Ist zu diesem Zeitpunkt kein weiterer Client mit dem Server verbunden, wird dieser beendet.

Anschließend erfolgt der Aufruf der Funktion "Me.m_OpcServer.Dispose()" zur Freigabe von Ressourcen die in der OPC Server Instanz gehalten werden sowie das Löschen der Instanz durch "Me.m_OpcServer = Nothing".

```

' show hour glas
Me.Cursor = Cursors.WaitCursor

If Me.Button_Connect_Disconnect.Text = BUTTON_DESCRIPTION_SERVER_CONN
Then
    ' Step: 3
    '
    ' select opc server and
    Me.m_OpcServer = New Opc.Da.Server(New OpcCom.Factory, _
    New Opc.URL(Opc.UrlScheme.DA + "://localhost/" + _
    Me.ComboBox_Server.Text))

    ' connect with ...
    Me.m_OpcServer.Connect()

    ' Enable or disable controls
    Me.ComboBox_Server.Focus()
    Me.SetInternalStateControl(InternalStateControl.Connected)

ElseIf Me.Button_Connect_Disconnect.Text = BUTTON_DESCRIPTION_SERVER_
Then
    ' Step: 4
    '
    ' disconnect from server and clean up
    Me.m_OpcServer.Disconnect()
    Me.m_OpcServer.Dispose()
    Me.m_OpcServer = Nothing

    ' Enable or disable controls
    Me.SetInternalStateControl(InternalStateControl.Disconnected)

    ' clear browse info
    If Not Me.m_DialogAddItems Is Nothing Then Me.m_DialogAddItems.Cle
End If

```

Abb. 6-9: OPC Server Instanz erzeugen und löschen

Server Informationen lesen

Die durch den OPC Server bereitgestellten Informationen werden im "Load Ereignis" "Step: 21" der Form "Form_ServerInfo" ausgelesen und zur Anzeige gebracht.

Der Aufruf der Form erfolgt im "Step: 5" (ohne Abbildung).

```

' Step: 21
'
' read opc server information
Private Sub Form_ServerInfo_Load(ByVal sender As System.Object, ByVal e As :
Try
    '
    If (Me.m_OPCTServer Is Nothing) Then Exit Sub

    ' Step: 21
    ' read opc server information
    Me.Label_Name.Text = m_OPCTServer.Name
    Me.Label_URL.Text = m_OPCTServer.Url.Path
    Me.Label_Status.Text = m_OPCTServer.GetStatus.StatusInfo
    Me.Label_Vendor.Text = m_OPCTServer.GetStatus.VendorInfo

Catch ex As Exception

```

Abb. 6-10: Auslesen der OPC Server Informationen

OPC Subscription erstellen und löschen

Allgemein

Dieser Abschnitt beschreibt das An- und Abmelden einer "Subscription" (Gruppe) am OPC Server. Die Verwaltung der Subscription erfolgt im Ereignis

Client-Entwicklung

nishandler "Button_Add_Remove_Group_Click" der Schaltfläche **"Add"** bzw. **"Remove"** und das Auslesen und Anzeigen der Subscriptioninformationen in der Funktion "Button_GroupInfo_Click" der **"Info ..."**-Schaltfläche.

Die relevanten Codesegmente im OPC Client Beispielprojekt sind:

- **"Step: 6"** - Subscription erstellen
- **"Step: 7"** - Subscription löschen
- **"Step: 8"** und **"Step: 22"** - Subscription Informationen anzeigen

Subscription erstellen - CreateSubscription

Mit der Funktion "Me.m_OpcServer.CreateSubscription()" und Übergabe des Parameters "Opc.Da.SubscriptionState" wird eine Subscription in der Collection vom OPC Server erstellt (**"Step: 6"**, siehe [Abb. 6-11 "OPC Subscription erstellen" auf Seite 193](#)).

Die Elemente der Klasse "Opc.Da.SubscriptionState" dienen zur eindeutigen Identifikation der Subscription und zur Konfiguration bzgl. dem Verhalten bzw. der Bearbeitung durch den OPC Server.

Eine Beschreibung der Klassenelemente ist in der folgenden Tabelle dargestellt.

Klassenelement	Beschreibung
Active	Die Items der Subscription werden zyklisch von der Steuerung gelesen und im Cache vom Server gespeichert.
ClientHandle	"Unique Identifier" (wird vom Client vergeben).
Deadband	Kleinstmögliche prozentuale Änderung eines Wertes, damit ein Ereignis zum Client ausgelöst wird.
KeepAlive	Größtmöglicher Zeitabstand zwischen zwei Ereignissen zum Client (Lebenszeichen).
Locale	Definition der Lokalisierung im Format "[language Code]-[country/region code]", z.B. "de-DE", "en-US" ...
Name	"Unique Name" (wird vom Client vergeben). Wird durch das Textfeld "TextBox_Group.Text" vorgegeben.
ServerHandle	"Unique Identifier" (wird vom Server vergeben).
UpdateRate	Zeitabstand, in dem der Server die Daten prüft und ein Ereignis an den Client sendet.

Tab. 6-3: Elemente der Klasse "Opc.Da.SubscriptionState"

```

' Step: 6 / 7
'
' create and cancel subscription
Private Sub Button_Add_Remove_Group_Click(ByVal sender As System.Object, By

    Dim subSubscriptionState As Opc.Da.SubscriptionState
    Dim iSubscription As Opc.Da.ISubscription

    Try

        ' show hour glas
        Me.Cursor = Cursors.WaitCursor

        If Me.Button_Add_Remove_Group.Text = BUTTON_DESCRIPTION_GROUP_ADD Then

            ' Step: 6
            '
            ' create subscription
            subSubscriptionState = New Opc.Da.SubscriptionState
            subSubscriptionState.Active = True ' activate cyclic communication
            subSubscriptionState.ClientHandle = System.Guid.NewGuid().ToString()
            subSubscriptionState.Deadband = 0.0F
            subSubscriptionState.Name = TextBox_Group.Text
            subSubscriptionState.UpdateRate = 1000

            Me.m_OpcServer.CreateSubscription(subSubscriptionState)

            ' Enable or disable controls
            Me.SetInternalStateControl(InternalStateControl.SubscriptionSet)

        ElseIf Me.Button_Add_Remove_Group.Text = BUTTON_DESCRIPTION_GROUP_REM

            ' Step: 7

```

Abb. 6-11: OPC Subscription erstellen

Subscription löschen - CancelSubscription

Zum Löschen einer Subscription wird die Funktion "Me.m_OpcServer.CancelSubscription()" mit Übergabe des Parameters "Opc.Da.ISubscription" aufgerufen ("Step: 7").

Nach dem Löschen der Subscription aus der Collection des OPC Servers ist es wichtig, die Funktion "iSubscription.Dispose()" aufzurufen, damit die von der Subscription belegten Ressourcen wieder freigegeben werden. Durch die Zuweisung "iSubscription = Nothing" wird die Referenz der Schnittstellenvariablen gelöscht.

Die zum Beginn des Codesegments durchgeführte lokale Zuweisung "iSubscription = Me.m_OpcServer.Subscription()" ist notwendig, da nach dem Aufruf der Funktion "Me.m_OpcServer.CancelSubscription()" die Subscription aus der Collection des OPC Servers entfernt wurde und nicht mehr zur Verfügung steht.

Client-Entwicklung

```

ElseIf Me.Button_Add_Remove_Group.Text = BUTTON_DESCRIPTION_GROUP_REM
    ' Step: 7
    ' cancel subscription and clean up
    iSubscription = Me.m_OpcServer.Subscriptions(0)
    Me.m_OpcServer.CancelSubscription(iSubscription)
    iSubscription.Dispose()
    iSubscription = Nothing

    ' Enable or disable controls
    Me.SetInternalStateControl(InternalStateControl.Connected)

End If

Catch ex As Exception

```

Abb. 6-12: OPC Subscription löschen

Subscription Informationen anzeigen

Die von der Subscription bereitgestellten Informationen werden im "Load Ereignis" ("Step: 22") der Form "Form_GroupInfo" ausgelesen und zur Anzeige gebracht.

Der Aufruf der Form erfolgt im "Step: 8" (ohne Abbildung).

```

If (Me.m_OPCTServer Is Nothing) Then Exit Sub

' Step: 22
' read opc subscription information
subScripton = m_OPCTServer.Subscriptions(0)
Me.Label_Name.Text = subScripton.Name
Me.Label_Deadband.Text = _
    subScripton.GetState().Deadband.ToString + " %"
Me.Label_UpdateRate.Text = _
    subScripton.GetState().UpdateRate.ToString + " ms"

Catch ex As Exception

```

Abb. 6-13: Auslesen der OPC Subscription Informationen

Steuerungsitems auflisten**Allgemein**

Dieser Abschnitt beschreibt das Auslesen aller in der Steuerung für den Zugriff über den OPC Server konfigurierten Items.

Die relevanten Codesegmente im OPC Client Beispielprojekt sind:

- "Step: 20", "Step: 20.1" und "Step: 20.2" - Steuerungsitems lesen

Steuerungsitems lesen

Mit dem Betätigen von "Add ..." der Form "Form_OpcClient" wird die Form "Form_AddItems" aufgerufen. Da das Lesen der Items unter Umständen lange dauern kann, erfolgt das Einlesen der am OPC Server verfügbaren Items nicht im "Load Ereignis" der Form, sondern über die Funktion "Button_BrowseItems_Click", die durch einen Klick auf "Browse / Refresh" ausgelöst wird.

Zum Auslesen der Items werden die beiden Klassen "Opc.ItemIdentifier" und "Opc.Da.BrowseFilters" der Funktion "Me.m_OPCTServer.Browse()" als Parameter übergeben ("Step: 20.2").

Hierbei haben die Klassen folgende Funktion:

Opc.ItemIdentifier Mit der Klasse "Opc.ItemIdentifier" wird der "Startpunkt" der Suche in der "Baum Hierarchie" der Items vorgegeben. Das heißt, wurde z.B. eine Suche abgebrochen, kann die Suche wieder an diesem Punkt fortgesetzt werden.

Opc.Da.BrowseFilter Die Klasse "Opc.Da.BrowseFilter" dient als Filterkriterium für das Ergebnis des Funktionsaufrufs. Hier besteht die Möglichkeit, nur Variablen zu lesen, die z.B. mit einer bestimmten Zeichensequenz beginnen.

Da es keine Vorgaben gibt, in welcher Form ein OPC Server die "Browsing" Information ausliest, wurde das Lesen der Steuerungsisems in zwei Funktionen ("Step: 20.1" und "Step: 20.2") aufgeteilt. Mit dieser Aufteilung können flache Hierarchien (Baumstruktur ohne Unterknoten) und strukturierte Hierarchien (Baumstruktur mit Unterknoten) ausgelesen werden (siehe nachfolgende Abbildungen).

```

' clear list, stop updating
Me.ListBox_Available_Items.BeginUpdate()
Me.ListBox_Available_Items.Items.Clear()

' Step: 20.1
'
' browse items
filters.ElementNameFilter = ""
filters.BrowseFilter = Opc.Da.browseFilter.all
filters.ReturnAllProperties = True
filters.ReturnPropertyValues = True
itemID.ItemName = ""
Me.BrowseItems(itemID, filters)

Catch ex As Exception

```

Abb. 6-14: Steuerungsisems lesen - Filter Konfiguration

```

' Step: 20.2
'
' browse items
elements = m_OPCTServer.Browse(itemID, Filters, position)

' adds each element to the list
For Each element In elements
'
' avoid errors while using .NET API 1.30
If Not element.ItemName Is Nothing Then
    itemName = element.ItemName
Else
    itemName = element.Name
End If
Me.ListBox_Available_Items.Items.Add(itemName)

If (element.HasChildren = True) Then
' check on subordinate items

' set branch to search
item.ItemName = itemName & element.ItemName
item.ItemPath = element.ItemPath
Me.BrowseItems(item, Filters)

End If

Next

```

Abb. 6-15: Steuerungsisems lesen - Rekursion

Client-Entwicklung

OPC Items hinzufügen und löschen

Allgemein

In diesem Abschnitt wird das Hinzufügen und Entfernen von Items zu einer Subscription beschrieben.

Die relevanten Code-Segmente im OPC Client Beispielprojekt sind:

- **"Step: 9", "Step: 9.1" und "Step: 9.2"** - Items hinzufügen
- **"Step: 10"** - Items löschen

Items hinzufügen - AddItems

In dem Form-Ereignis "Button_Add_Item_Click" (**"Step: 9"**) erfolgt der Aufruf der Form "Form_AddItems" und das Hinzufügen der in der Form ausgewählten Items.

Die Variablen können im Textfeld der Form "Add Items" durch ";" getrennt eingegeben oder im Listenfeld ausgewählt werden. Bei der Eingabe der Variablen ist die Item-Syntax des jeweiligen OPC Servers zu beachten.

Im folgenden ist die Item-Syntax für den "OPC.lwSCP" Server und dem "IndraLogic OPC" Server dargestellt.

"OPC.lwSCP" Server

- **SPS Variablen**
"! [angeforderter Datentyp], [Steuerungsname oder Alias], [lwSCP Dienst], [voll qualifizierender Variablenname]"
Beispiel:
"!Bool,ImMlc2G,Plc.PVL,Application.HMIVARS.bLifeSign_hmi"
- **sercos Parameter**
"! [angeforderter Datentyp], [Steuerungsname oder Alias], [lwSCP Dienst], [Antriebs Adresse], [Parameter Nummer], [Elementkodierung]"
Beispiel:
"!UI4,ImMlc2G,SPA1,0,C-0-0001,D"
"!UI4,ImMlc2G,SPA1,0,C-0-0001,40"



Der "OPC.lwSCP" Server ermöglicht den Zugriff auf Parameter und SPS-Variablen!



Weitere Informationen zu der Item-Syntax des "OPC.lwSCP" Servers finden Sie in [Kap. 4.3.3 "lwSCP und OPC.lwSCP Items" auf Seite 98](#).

"IndraLogic OPC" Server

- **SPS Variablen**
"[Steuerungsname].[voll qualifizierender Variablenname]"
Beispiel:
"PLC_IL2G.Application.HMIVARS.bLifeSign_hmi"



Mit dem "IndraLogic OPC" Server ist der Zugriff auf SPS-Variablen möglich.

Die Beschreibung der "Browse"-Funktionalität ist in [Kap. "Steuerungsitems auflisten" auf Seite 194](#) beschrieben.

Wurden die Items in der Form "Add Items" eingegeben bzw. ausgewählt, wird für jedes Item im **"Step: 9.1"** eine Klasse "Opc.Da.Item" erstellt und initialisiert. Der Name des Items wird in der Zeile "opcItems(index).ItemName = items(index)" zugewiesen.

Die im Beispiel verwendeten Klassenelemente der Klasse "Opc.Da.Item" sind in der folgenden Tabelle dargestellt:

Klassenelement	Beschreibung
Active	Für das Item werden "DataChanged" Ereignisse an den Client gesendet.
ActiveSpecified	Die Eigenschaft "Active" wurde definiert.
ItemName	Voll qualifizierter Item-Name. Die Syntax ist abhängig vom OPC Server.
ItemPath	Voll qualifizierter Item-Pfad. Diese Eigenschaft wird vom "OPC.lwSCP" nicht ausgewertet.
MaxAge	Beim Lesen des Items aus dem Cache des OPC Servers darf der Wert nicht älter sein, als in "MaxAge" definiert.
MaxAgeSpecified	Die Eigenschaft "MaxAge" wurde definiert. Ist diese Eigenschaft auf "True" gesetzt, wird in Abhängigkeit von "MaxAge" der Wert aus dem "Cache" des OPC Servers oder vom "Device" gelesen. Mit dem Setzen der Eigenschaft auf "False" wird immer vom "Device" gelesen.

Tab. 6-4: Elemente der Klasse "Opc.Da.Item"

Nachdem alle OPC Items erstellt wurden, wird das OPC Item-Array mit dem Funktionsaufruf "Me.m_OpcServer.Subscriptions(0).AddItems()" in der Subscription am OPC Server angemeldet (**"Step: 9.2"**).

Client-Entwicklung

```

' Step: 9
'
' add items to the subscription
ReDim opcItems(items.Length - 1)
For index = 0 To items.Length - 1

    ' Step: 9.1
    '
    ' creating new item
    opcItems(index) = New Opc.Da.Item
    ' set item properties
    Randomize()
    opcItems(index).ClientHandle = CInt(Int((1000 * Rnd()) + 1))
    opcItems(index).Active = True
    opcItems(index).ActiveSpecified = True
    opcItems(index).ItemName = items(index)
    opcItems(index).ItemPath = ""
    opcItems(index).MaxAge = -1
    'opcItems(index).MaxAgeSpecified = False ' read from device
    opcItems(index).MaxAgeSpecified = True ' read from cache

Next

' Step: 9.2
'
' add the item to the subscription and to the list view
itemResults = Me.m_OpcServer.Subscriptions(0).AddItems(opcItems)
For index = 0 To itemResults.Length - 1

    If (itemResults(index).ResultID.Succeeded = True) Then
        listViewItem = _
            Me.ListView_ItemData.Items.Add(opcItems(index).ItemName)
        listViewItem.SubItems.Add("") ' sub item 1
        listViewItem.SubItems.Add("") ' sub item 2
    End If

Next

```

Abb. 6-16: Items einer Subscription hinzufügen

Items löschen - RemoveItems

Zum Löschen der Items aus der OPC Subscription werden diese in der Listenansicht ausgewählt und mit Betätigen der **"Remove"**-Schaltfläche entfernt. In dem daraufhin ausgelösten Form-Ereignis "Button_Remove_Item_Click" wird die Funktion "Me.m_OpcServer.Subscriptions(0).RemoveItems()" aufgerufen, die das als Parameter übergebene OPC Item entfernt (**"Step: 10"**).

```

Me.RadioButton_Sync.Checked = True

' set array size for opc item list
selectedItems = Me.ListView_ItemData.SelectedItems.Count - 1
ReDim opcItemIds(selectedItems)

' Step: 10
'
' remove selected items from opc subscription and from the list
For index = 0 To selectedItems
    For Each opcItem In Me.m_OpcServer.Subscriptions(0).Items
        If Me.ListView_ItemData.SelectedItems(index).Text.Equals(opcItem.ItemName) Then
            opcItemIds(index) = opcItem
            Exit For
        End If
    Next
Next
' remove items
opcIdentResult = Me.m_OpcServer.Subscriptions(0).RemoveItems(opcItemIds)

' clean up list view control
For index = 0 To opcIdentResult.Length - 1
    If (opcIdentResult(index).ResultID = ResultID.S_OK) Then
        For Each listViewItem In Me.ListView_ItemData.SelectedItems
            If (opcIdentResult(index).ItemName.Equals(listViewItem.Text)) Then
                Me.ListView_ItemData.Items.Remove(listViewItem)
            End If
        Next
    End If
Next
End If
Next

```

Abb. 6-17: Items von einer Subscription entfernen



Das Entfernen von Items aus einer Gruppe sollte immer für eine gesamte Liste von Items durchgeführt werden, damit im OPC Server die Verwaltungsarbeiten zum Ab- und Anmelden der Gruppe nur einmalig entstehen.

Steuerungsdaten lesen

Allgemein

Dieser Abschnitt beschreibt den lesenden Zugriff auf die Steuerungsdaten. Hierbei wird zwischen einem

- "synchronen" und
- "asynchronen" Zugriff unterschieden.

Das Lesen der Steuerungsdaten erfolgt im Ereignishandler "Button_Read_Click" der **"Read"**-Schaltfläche.

Die relevanten Codesegmente sind:

- **"Step: 11"** - synchrones Lesen
- **"Step: 12"** - asynchrones Lesen
- **"Step: 13"** - Daten auswerten

Synchrones Lesen

Bei einem "synchronen" Zugriff wartet der Client solange, bis der Server die Daten bereitgestellt hat. In Abhängigkeit des im "Opc.Da.Item" definierten Verhalten, werden die Daten aus dem internen "Cache" des OPC Servers oder von der Steuerung "Device" gelesen. Der Client nimmt in diesem Zeitraum **keine weiteren Bedienhandlungen an**, d.h. der "synchrone" Aufruf ist blockierend.



Das Lesen von Daten aus dem "Cache" ist in der Regel immer schneller, als das Lesen vom "Device". Dieses Verhalten sollte bei der Konzeption eines OPC Clients immer berücksichtigt werden.

Client-Entwicklung

Das "synchrone" Lesen ("**Step: 11**") von Daten einer Subscription erfolgt über den Funktionsaufruf `opcValueResults = Me.m_OpcServer.Subscriptions(0).Read()` und der Übergabe der zu lesenden Items `"Me.m_OpcServer.Subscriptions(0).Items"` als Parameter.

In unserem Beispiel wird die gesamte Gruppe gelesen und das Ergebnis an die Funktion `"OnReadComplete"` zur Verarbeitung weitergegeben.

Die Beschreibung der Funktion `"OnReadComplete"` folgt in [Kap. "Daten auswerten"](#) auf Seite 201.

Asynchrones Lesen

Bei einem "asynchronen" Zugriff wird der Auftrag an den Server übergeben und der Funktionsaufruf kehrt sofort zurück. Der Server bearbeitet anschließend den Auftrag und meldet das Ergebnis der Auftragsbearbeitung an den Client über einen zuvor definierten "Delegate" (Ereignishandler `"OnReadComplete"`) zurück, d.h., der Client kann während der Auftragsbearbeitung durch den Server weitere **Bedienkommandos annehmen und verarbeiten**.

Das "asynchrone" Lesen ("**Step: 12**") erfolgt wie beim "synchrone" Lesen über den Funktionsaufruf `opcValueResults = Me.m_OpcServer.Subscriptions(0).Read()`.

Der Aufruf unterscheidet sich nur in der Anzahl der zu übergebenden Parameter. Die Übergabeparameter und ihre Beschreibung sind in der folgenden Tabelle aufgelistet.

Übergabeparameter	Beschreibung
<code>Opc.Da.Item()</code>	Item-Collection aller zu lesenden Items. Wie beim synchronen Lesen.
<code>Object</code>	Vom Client übergebene ID die dem "Delegate" bei der Verarbeitung der Ergebnisse mitgegeben wird (optional "Nothing").
<code>ReadCompleteEventHandler</code>	"Delegate" für die Verarbeitung der Ergebnisse der Anfrage. Der "Delegate" muss mit der Anweisung <code>"New ReadCompleteEventHandler(AddressOf OnReadComplete)"</code> an den Funktionsaufruf übergeben werden.
<code>IRequest</code>	Handle zum Abbrechen der Anforderung über den Funktionsaufruf <code>"Subscription.Cancel()"</code> (optional 'Nothing').

Tab. 6-5: Übergabeparameter für das asynchrone Lesen

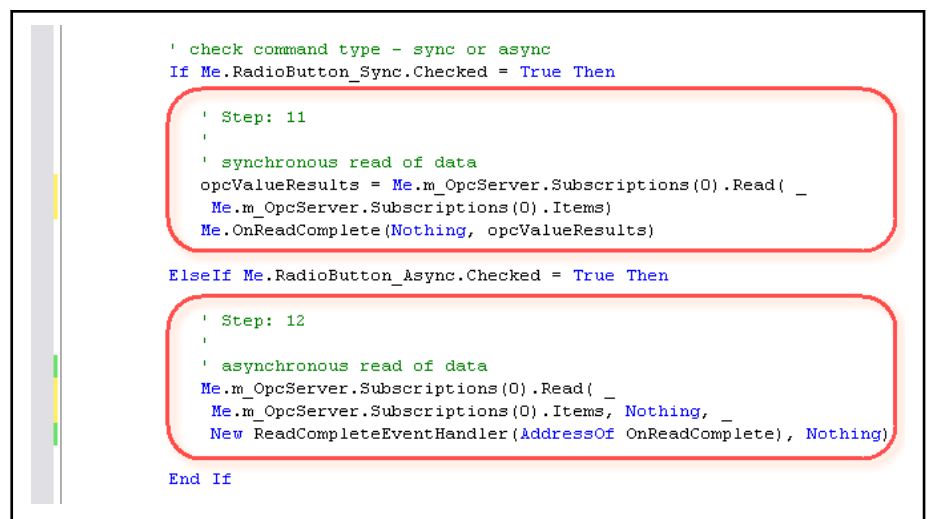


Abb. 6-18: Synchrones und asynchrones Lesen von Steuerungsdaten

Daten auswerten

Die Funktion "OnReadComplete" zum Auswerten der vom OPC Server zurückgelieferten Daten wird nach dem "synchronen" Lesen vom OPC Client und als "Ereignishandler" ("Delegate") vom OPC Server, mit der Bereitstellung der Daten nach der "Asynchronen" Anforderung, aufgerufen.

Da Steuerelemente nicht "threadsicher" sind, d.h. der Zugriff auf diese Objekte nur von einem Thread erfolgen kann, muss geprüft werden, ob der Funktionsaufruf "OnReadComplete" im gleichen Thread läuft wie der Thread der Form. Diese Überprüfung erfolgt im "Step: 13.1".

Wurde die Funktion nicht im Thread der Form aufgerufen, wird der Aufruf durch die Funktion "BeginInvoke()" synchronisiert.

Im "Step: 13.2" werden die Ergebnisse ausgewertet und in der Listenansicht angezeigt.

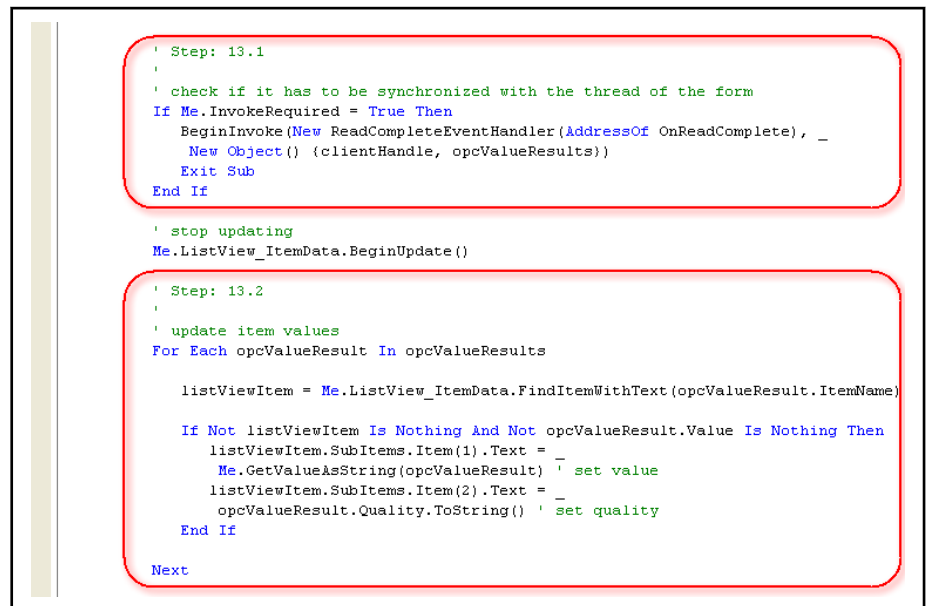


Abb. 6-19: Daten auswerten - lesend

Steuerungsdaten schreiben

Allgemein

Dieser Abschnitt beschreibt den schreibenden Zugriff auf die Steuerungsdaten. Hierbei wird zwischen einem

- "synchronen" und
- "asynchronen" Zugriff unterschieden.

Das Schreiben der Steuerungsdaten erfolgt im Ereignishandler "Button_Write_Click" der "Write"-Schaltfläche.

Die relevanten Code-Segmente im OPC Client Beispielprojekt sind:

- **"Step: 14"** - synchrones Schreiben
- **"Step: 15"** - asynchrones Schreiben
- **"Step: 16"** - Daten auswerten

Synchrones Schreiben

Bei einem "synchronen" Zugriff wartet der Client solange, bis der Server den Schreibauftrag beendet hat.

Das Schreiben der Daten erfolgt im "Cache" und im "Device" des OPC Servers, wenn die Subscription und das Item aktiv ist. Der Client nimmt in diesem Zeitraum **keine weiteren Bedienhandlungen an**, d.h. der "synchrone" Aufruf ist blockierend.

Das "synchrone" Schreiben (**"Step: 14"**) von Daten einer Subscription erfolgt über den Funktionsaufruf "opcResults = Me.m_OpcServer.Subscriptions(0).Write()" und der Übergabe der zu schreibenden "Opc.Da.ItemValue" ("opcValues") als Parameter.

In unserem Beispiel wird nur ein Item von der gesamten Subscription geschrieben und das Ergebnis des Schreibauftrages an die Funktion "OnWriteComplete" zur Verarbeitung weitergegeben.

Die Beschreibung der Funktion "OnWriteComplete" folgt in [Kap. "Daten auswerten" auf Seite 203](#).

Asynchrones Schreiben

Bei einem "asynchronen" Zugriff wird der Auftrag an den Server übergeben und der Funktionsaufruf kehrt sofort zurück.

Der Server bearbeitet anschließend den Auftrag und meldet das Ergebnis der Auftragsbearbeitung an den Client über einen zuvor definierten "Delegate" (Ereignishandler "OnWriteComplete") zurück. D.h., der Client kann während der Auftragsbearbeitung durch den Server weitere **Bedienkommandos annehmen und verarbeiten**.

Das "asynchrone" Schreiben (**"Step: 15"**) erfolgt wie beim "synchronen" Schreiben über den Funktionsaufruf

"opcResults = Me.m_OpcServer.Subscriptions(0).Write()".

Der Aufruf unterscheidet sich nur in der Anzahl der zu übergebenden Parameter. Die Übergabeparameter und ihre Beschreibung sind in der folgenden Tabelle aufgelistet.

Übergabeparameter	Beschreibung
Opc.Da.ItemValue()	Item-Collection aller zu schreibenden Items mit den Werten. Wie beim synchronen Schreiben.
Object	Vom Client übergebene ID die dem "Delegate" bei der Verarbeitung der Ergebnisse mitgegeben wird (optional "Nothing").
WriteCompleteEventHandler	"Delegate" für die Verarbeitung der Ergebnisse des Auftrags. "Delegate" muss mit der Anweisung "New WriteCompleteEventHandler(AddressOf OnWriteComplete)" an den Funktionsaufruf übergeben werden.
IRequest	Handle zum Abbrechen des Auftrags über den Funktionsaufruf "Subscription.Cancel()" (optional "Nothing").

Tab. 6-6: Übergabeparameter für das asynchrone Schreiben

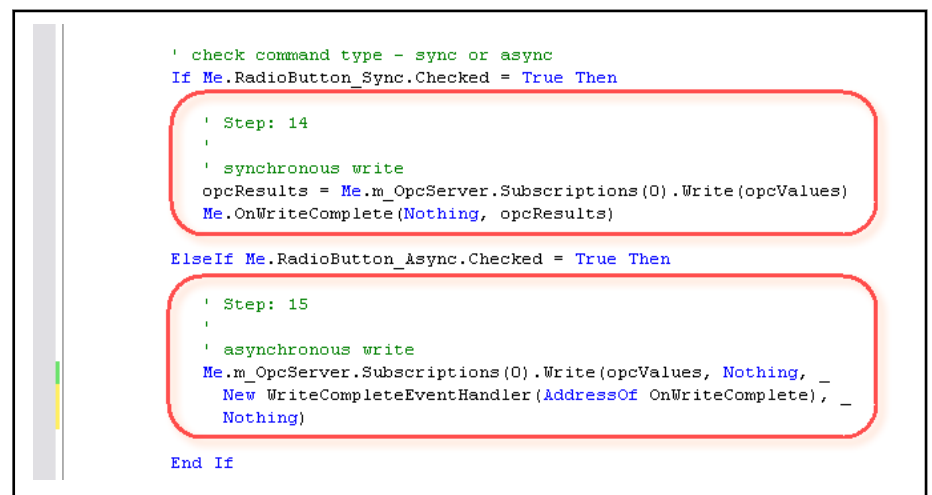


Abb. 6-20: Synchrones und asynchrones Schreiben von Steuerungsdaten

Daten auswerten

Die Funktion "OnWriteComplete" dient der Auswertung der vom OPC Server zurückgelieferten Ergebnisse.

Sie wird nach dem "synchronen" Schreiben vom OPC Client und als "Ereignishandler" ("Delegate") vom OPC Server, mit dem Abschluss der Auftragsbearbeitung nach dem "Asynchronen" Kommando, aufgerufen.

Da Steuerelemente nicht "threadsicher" sind, d.h. der Zugriff auf diese Objekte nur von einem Thread erfolgen kann, muss geprüft werden, ob der Funktionsaufruf "OnWriteComplete" im gleichen Thread läuft wie der Thread der Form. Diese Überprüfung erfolgt im **"Step: 16.1"**. Wurde die Funktion nicht im Thread der Form aufgerufen, wird der Aufruf durch die Funktion "BeginInvoke()" synchronisiert.

Im **"Step: 16.2"** kann eine Auswertung der Ergebnisse des Schreibauftrags erfolgen. Diese Auswertung kann in einer "Log-Datei" oder z.B. in einem "StatusStrip"-Steuerelement der Applikation erfolgen. Der Beispiel Client enthält keine Auswertung der Ergebnisse des Schreibauftrags.

Client-Entwicklung

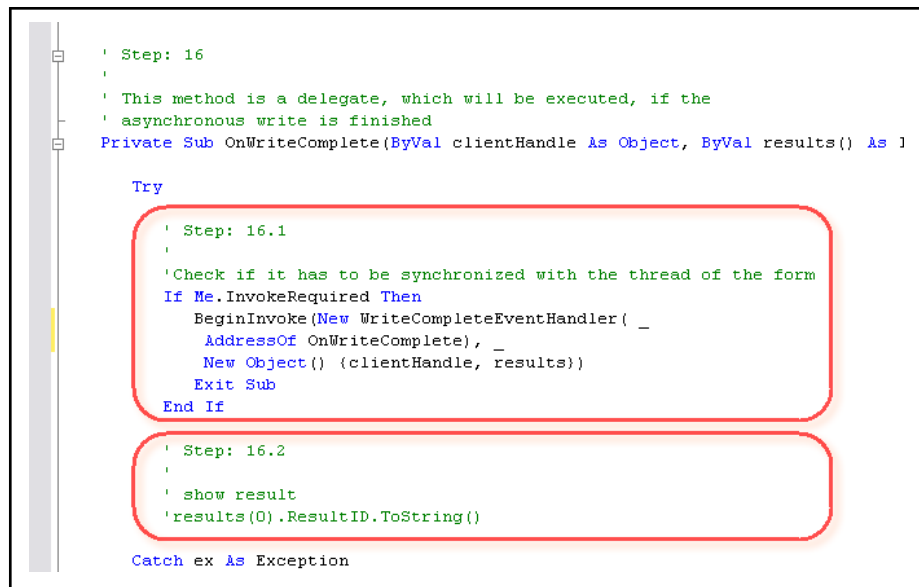


Abb. 6-21: Daten auswerten - schreibend

Zyklisches Lesen der Steuerungsdaten

Allgemein

In diesem Abschnitt wird das zyklische Lesen von Steuerungsdaten beschrieben.

Voraussetzung für das zyklische Lesen von Werten ist, dass

- die Subscription und
- das Item aktiv, sowie
- ein Eventhandler an der Subscription angemeldet ist.

Das Aktivieren und Deaktivieren des zyklischen Lesens wird über das Ereignis "RadioButton_CheckedChanged" der "Radio Buttons" "Synchronous", "Asynchronous" und "Cyclic" gesteuert.

Die relevanten Codesegmente im OPC Client Beispielprojekt sind:

- **"Step: 17"** - zyklisches Lesen aktivieren
- **"Step: 18"** - zyklisches Lesen deaktivieren
- **"Step: 19"** - Daten auswerten

Zyklisches Lesen aktivieren

Da die Voraussetzungen der aktiven Subscription und aktiven Items bereits erfüllt ist, muss nur noch der Eventhandler "DataChanged" zu der Subscription Liste "iSubscription.DataChanged" hinzugefügt werden (**"Step: 17"**). Mit dem Hinzufügen sind alle Voraussetzungen erfüllt und der Client erhält die "DataChanged" Ereignisse.

Zyklisches Lesen deaktivieren

Zum Deaktivieren des zyklischen Lesens wird der Eventhandler

"DataChanged" wieder aus der Subscription Liste

"iSubscription.DataChanged" entfernt. Nach dem Entfernen werden keine weiteren Werte-Änderungen mehr an den Client gemeldet.

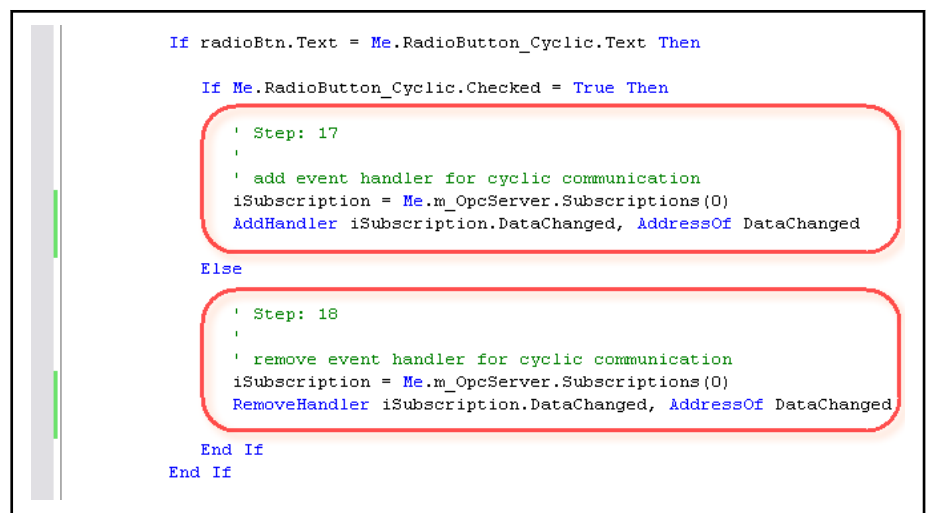


Abb. 6-22: Ereignisbehandlung hinzufügen und entfernen

Daten auswerten

Die Funktion "DataChanged" wird vom OPC Server zyklisch in den Zeitabständen aufgerufen, wie es im Element "UpdateRate" der Klasse "Opc.Da.SubscriptionState" festgelegt wurde.

Es werden alle Items mit den zugehörigen Werten der "DataChanged" Funktion übergeben, wenn

- sich der Wert geändert hat oder
- das maximale "Alter" des Wertes überschritten ist.

Das heißt, in Abhängigkeit der Konfiguration der Elemente "MaxAge" und "MaxAgeSpecified" der Klasse "Opc.Da.Item".

Da Steuerelemente nicht "threadsicher" sind, d.h. der Zugriff auf diese Objekte nur von einem Thread erfolgen kann, muss geprüft werden, ob der Funktionsaufruf "DataChanged" im gleichen Thread läuft wie der Thread der Form. Diese Überprüfung erfolgt im "Step: 19.1".

Wurde die Funktion nicht im Thread der Form aufgerufen, wird der Aufruf durch die Funktion "BeginInvoke()" synchronisiert.

Im "Step: 19.2" werden die Ergebnisse, wie beim synchronen und asynchronen Lesen ausgewertet und in der Listenansicht angezeigt.

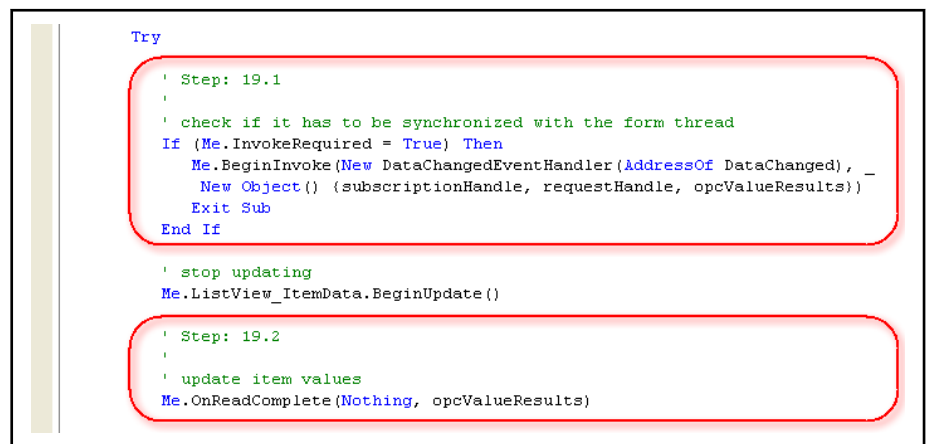


Abb. 6-23: Daten auswerten - zyklisches Lesen

Client-Entwicklung

Fehlerbehandlung

Mit dem Beispiel OPC Client wurde ein Minimum an Fehlerhandling implementiert. Bei einem an Produktionsmaschinen eingesetzten OPC Client sollten auftretende Fehler detaillierter ausgewertet und protokolliert werden.

Der OPC Client beschränkt sich daher auf die Überprüfung von Variablen und ein einfaches Exception-Handling in der Form von

- "If"-Abfragen und
- Exception Handling mit "Try...Catch..."-Anweisungen

"If"-Abfragen

Mit der "If"-Abfrage werden die für die Funktionsausführung notwendigen Parameter geprüft. Können die vorgegebenen Definitionen nicht erfüllt werden, wird die Funktion an dieser Stelle abgebrochen.

Exception Handling mit "Try...Catch..."-Anweisungen

Da die meisten Komponenten einen Fehler in der Funktionsausführung dem Aufrufer über das "Ausgeben" einer Ausnahmemeldung mitteilen, sollte mindestens eine "Try...Catch..."-Anweisung in jeder Funktion/Methode enthalten sein. Im Beispiel OPC Client wurde, bedingt durch den Umfang der einzelnen Funktionen, nur eine "Try...Catch..."-Anweisung programmiert.

6.2 OPC Client Visual C# .NET

6.2.1 Allgemein

Dieser Abschnitt beschäftigt sich mit der Entwicklung des Visual C# .NET OPC Clients mit dem "Microsoft Visual Studio .NET 2005".

Er umfasst folgende Themen:

- Projekt erstellen
- Verweise hinzufügen
- Windows Forms und Steuerelemente hinzufügen
- OPC Client Implementierung.

Die folgende Abbildung zeigt den OPC Client als aktive Windows Applikation.

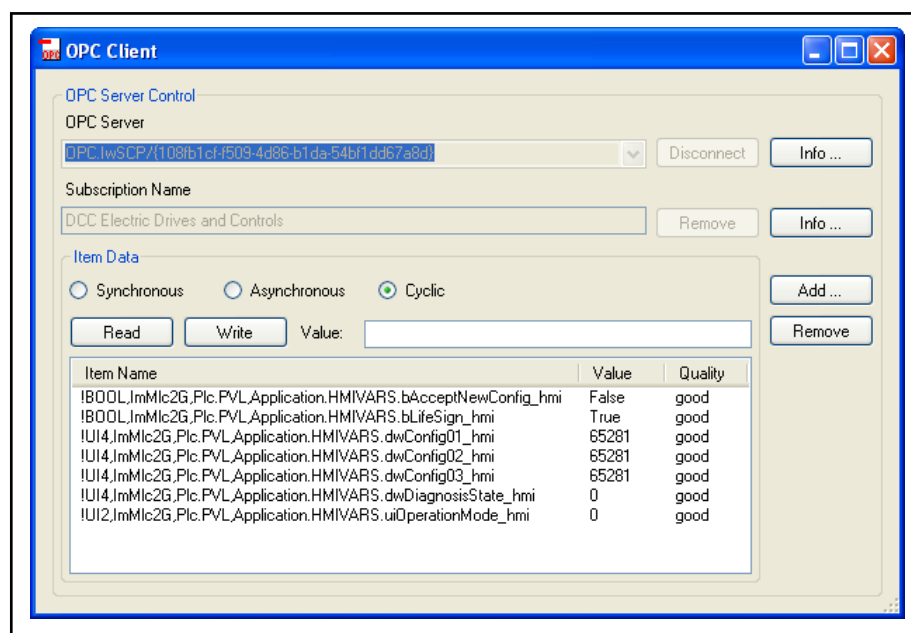


Abb. 6-24: OPC Client Oberfläche

6.2.2 Visual C# Projekt "OPC Client"

Neues Projekt

Um ein neues Projekt in der Entwicklungsumgebung "Microsoft Visual Studio .NET 2005" zu erstellen, betätigen Sie entweder direkt auf der Startseite den Link "Neues Projekt" oder klicken Sie im Menü **Datei ► Neu** auf den Menüpunkt **Projekt....**

Es erscheint der Dialog "Neues Projekt", wie in nachstehender Abbildung dargestellt.

Hier wählen Sie Projekttyp und Vorlage aus:

- "Andere Sprachen \ Visual C# \ Windows" oder
- "Visual C# \ Windows" und als Vorlage
- "Windows Application"

Geben Sie als Projektname "OPCClient" ein und passen Sie den vorgeschlagenen Speicherort entsprechend an.

Mit Betätigen der Schaltfläche **OK** wird das Projekt erstellt.

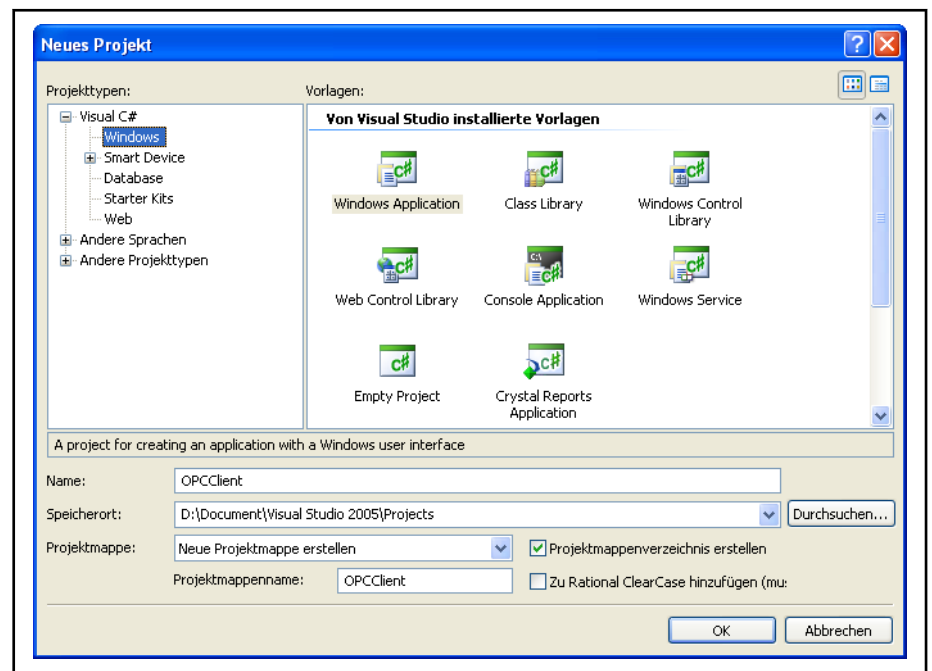


Abb. 6-25: Neues Projekt erstellen

Verweise hinzufügen

Damit auf die Funktionalitäten der OPC .NET API zugegriffen werden kann, muss ein Verweis auf die "OpcNetApi.dll" und "OpcNetApi.Com.dll" im Projekt vorhanden sein.

Die Verwaltung der "Projektreferenzen" erfolgt im Projektmappen-Explorer auf dem Knoten "Referenzen" von dem Projekt "OPCCClient" bzw. über die bereits hinzugefügten Referenzen.

Das Hinzufügen von Referenzen wird über den Menüpunkt **Verweis hinzufügen** ... von dem zugehörigen Kontextmenü oder über das Menü **Projekt** vorgenommen. Im Dialog "Referenz hinzufügen" können die beiden Dateien "OpcNetApi.dll" und "OpcNetApi.Com.dll" auf der Registerkarte "Durchsuchen" dem Projekt hinzugefügt werden.

Client-Entwicklung

Beide Dateien werden bei der Installation der "IndraWorks Suite" mitgeliefert und sind im Installationsordner (z.B. "C:\Programme\Rexroth\IndraWorks") gespeichert.

Mit Öffnen der Dateien werden diese in die Auflistung des Knotens "Referenzen:" übernommen.

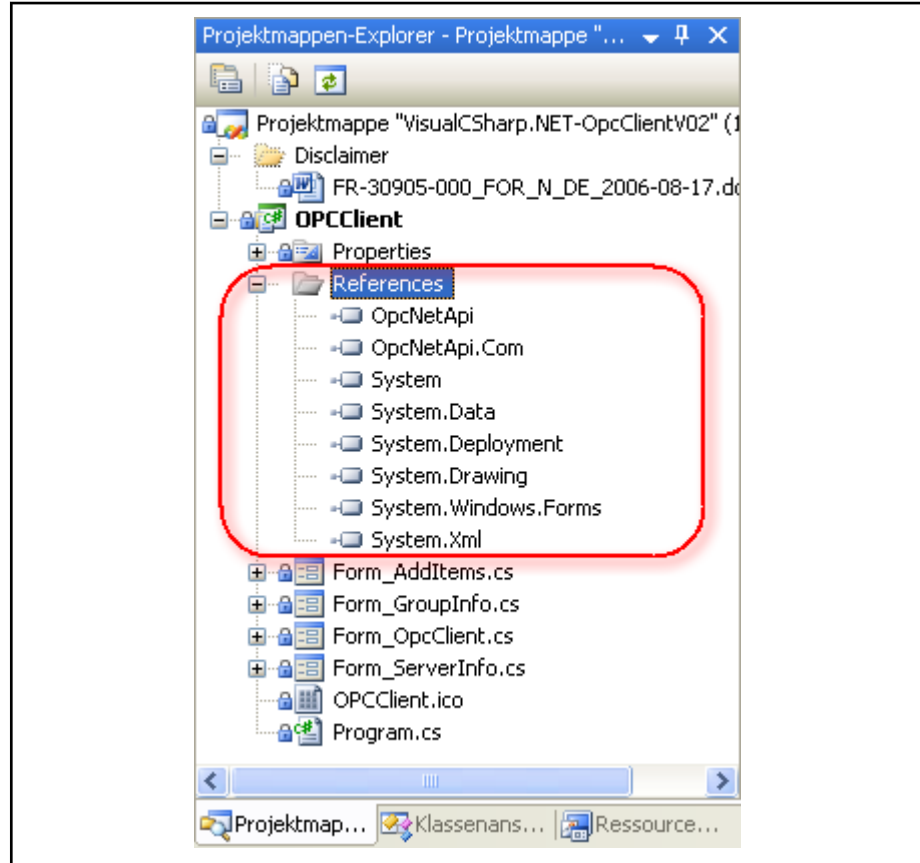


Abb. 6-26: Projekt Referenzen

Windows Form und Steuerelement hinzufügen

Das Hinzufügen einer Windows Form in das Startprojekt erfolgt über den Menüpunkt **Projekt ► Windows Form hinzufügen...**

Im Dialog "Neues Element hinzufügen - OPCClient" ist die Vorlage "Windows Form" bereits ausgewählt und muss nur noch der Name der Form eingegeben werden.

Bei allen im OPC Client Beispiel erstellten Windows Forms wurde dem Namen der Bezeichner "Form_" vorangestellt.

Die nachfolgende Abbildung zeigt eine Übersicht des Projekts "OPC Client" im "Projektmappen-Explorer".

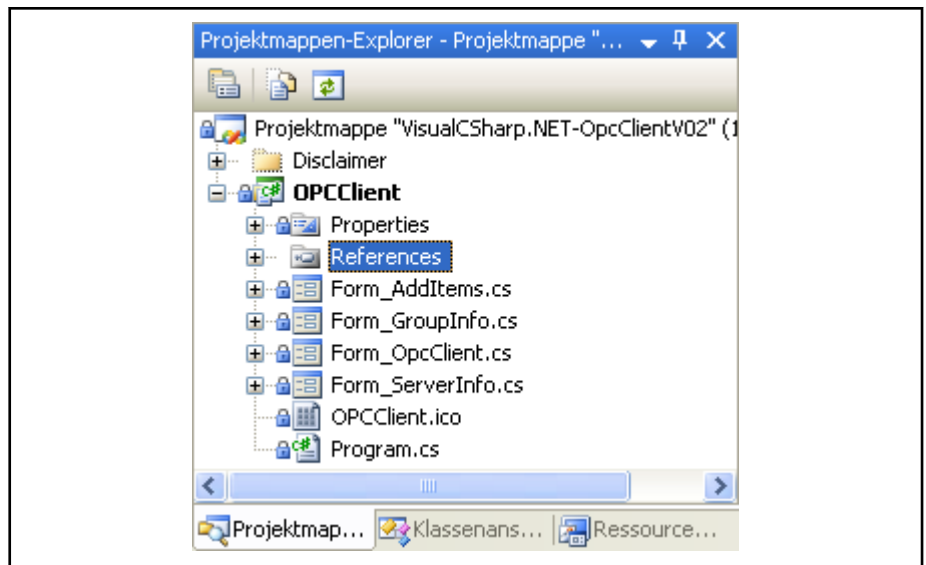


Abb. 6-27: Projektübersicht

Das Entwerfen von "Windows Forms" führen Sie mit Hilfe des "Windows Forms Designer" und den in der "Toolbox" zur Verfügung stehenden Steuerelementen durch. Hierbei wählen Sie die Steuerelemente per Mausklick aus und platzieren sie auf der Form.

Die Konfiguration der Steuerelemente für das ausgewählte Objekt erfolgt im Fenster "Eigenschaften".

Außerdem können Sie die Größe und Position der Steuerelemente mit der Maus im Windows Forms Designer zu verändern.

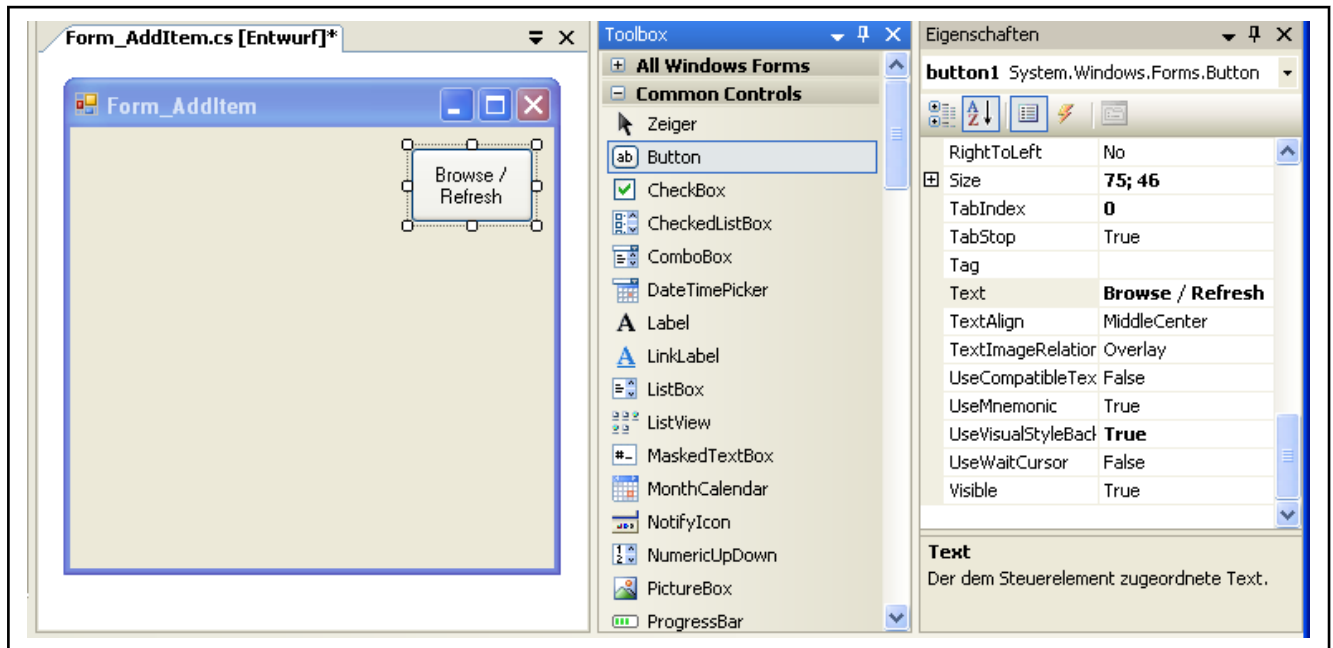


Abb. 6-28: Hinzufügen und Konfigurieren von Steuerelementen

Die nachstehende Abbildung zeigt die Form "OPC Client" im "Windows Forms Designer".

Auf eine detaillierte Beschreibung der Form und deren einzelne Steuerelemente wird aus Gründen der Übersichtlichkeit verzichtet. Alle konfigurierten Eigenschaften können dem Beispielprojekt entnommen werden.

Client-Entwicklung

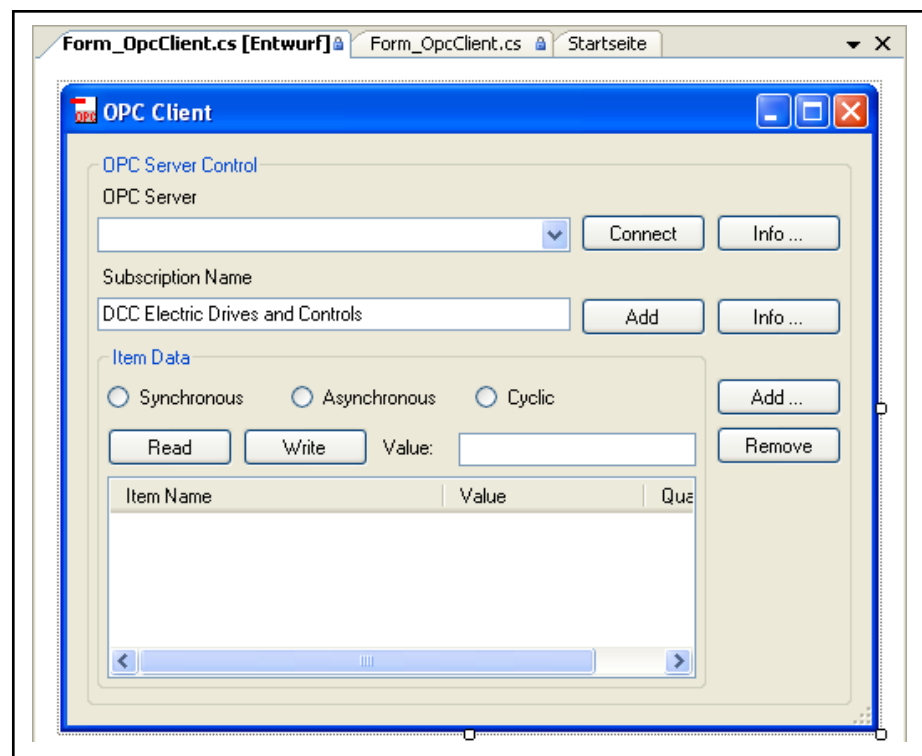


Abb. 6-29: Windows Forms Designer "OPC Clients"

OPC Client Implementierung

Allgemein

Im folgenden Abschnitt wird die Vorgehensweise bei der Programmierung des OPC Clients beschrieben.

Damit die Übersicht im Programm besser gegeben ist, wurden einzelne Bereiche in der Klasse mit der "#region"-Direktive zusammengefasst. Damit können Sie diese Bereiche bei der Programmerstellung zur besseren Übersichtlichkeit auf-, bzw. zugeklappen.

Die nachstehende Abbildung zeigt am Beispiel der Form "Form_OpcClient" die definierten Bereiche.

Bereich	Beschreibung
"#region" -"#endregion"	
"Declaration"	Enthält die Deklaration der Klasse (Member Variablen und Konstanten).
"Form Events"	Prozeduren zur Behandlung der von der Form und den enthaltenen Steuerelementen ausgelösten Ereignisse.
"OPC Events"	Prozeduren zur Behandlung der vom OPC Server ausgelösten Ereignisse.
"Additional Internal Functions"	Zusätzliche interne Definitionen, Strukturen und Funktionen zur Optimierung der Benutzerführung.

Tab. 6-7: Programm-Strukturierung mittels "#region"-Direktiven

Auch sind die relevanten Programmzeilen im Beispielprojekt mit dem Schlüsselwort **"Step:"** und einer fortlaufenden Nummerierung gekennzeichnet. Das heißt, mit der im Visual Studio enthaltenen Suche "In Dateien Suchen"

(<Strg>+<Umschalt>+<F>) kann durch die Eingabe des Suchbegriffs, z.B. **"Step: 2"**, direkt zu der beschriebenen Implementierung verzweigt werden.

Die im weiteren beschriebene Implementierung wurde in Anlehnung an die Bedienreihenfolge der Beispielapplikation "OPC Client" dokumentiert.

Die Gliederung ist in der folgenden Abbildung dargestellt.

Kapitelüberschrift	Beschreibung
Kap. "Deklaration der Variablen" auf Seite 211	Deklaration der Variablen für die Verwaltung des OPC Servers und der Windows Form.
Kap. "OPC Server auflisten" auf Seite 212	Auflistung aller auf dem lokalen PC installierten OPC Server.
Kap. "OPC Server Instanz erstellen und löschen" auf Seite 212	Erstellen und Löschen der OPC Server Instanz und Anzeige der OPC Server Information.
Kap. "OPC Subscription erstellen und löschen" auf Seite 215	Erstellen und Löschen einer OPC Subscription und Anzeige der Subscription Information.
Kap. "Steuerungsisitems auflisten" auf Seite 217	Lesen und Anzeigen aller Items der Steuerung (browsen).
Kap. "OPC Items hinzufügen und löschen" auf Seite 218	Hinzufügen und Löschen von OPC Items.
Kap. "Steuerungsdaten lesen" auf Seite 221	Lesen der Steuerungsdaten "Synchron" und "Asynchron".
Kap. "Steuerungsdaten schreiben" auf Seite 223	Schreiben der Steuerungsdaten "Synchron" und "Asynchron".
Kap. "Zyklisches Lesen der Steuerungsdaten" auf Seite 226	Zyklisches Lesen der Steuerungsdaten.

Tab. 6-8: Übersicht zur Gliederung der Implementation

Deklaration der Variablen

Im Bereich Deklaration **"Step: 1"** sind die Variablen der "Form_OpcClient" deklariert.

Für die Namen wurde die Syntax so gewählt, dass alle Klassenvariablen mit dem Präfix "m_" ("Member Variablen") und die Konstanten-Definitionen in "GROßBUCHSTABEN" geschrieben werden.

Bei den Variablen handelt es sich um

- das Objekt für die OPC Server Instanz und
- Variablen für die Verwaltung der Form.

Die Zeile **"private Opc.Da.Server m_OpcServer = null;"** ist die für das Beispiel relevante Deklaration. In der Variablen "m_OpcServer" wird das erstellte OPC Server Objekt gehalten.

Client-Entwicklung

```

// Step: 1
//
// declaration of members used for opc administration
#region " Declaration "

// Step: 1
//
// declaration of members used for opc server administration
private Opc.Da.Server m_OpcServer = null;

//
// members to store configuration
private OPCClient.Form_AddItems m_DialogAddItems = null;
private InternalStateControl m_InternalStateControl;

//
// constant values
private readonly String BUTTON_DESCRIPTION_SERVER_CONNECT = "Connect";
private readonly String BUTTON_DESCRIPTION_SERVER_DISCONNECT = "Disconnect";
private readonly String BUTTON_DESCRIPTION_GROUP_ADD = "Add";
private readonly String BUTTON_DESCRIPTION_GROUP_REMOVE = "Remove";
private readonly Int32 MAX_ITEMS = 10;

#endregion

```

Abb. 6-30: Deklaration der Klassenvariablen

OPC Server auflisten

Die Auflistung aller lokal auf dem PC installierten OPC Server erfolgt im "Load Ereignis" **"Step: 2"** der Form "Form_OpcClient".

Es werden alle OPC Server über die Methode "opcDiscovery.GetAvailableServers()" der Discovery Schnittstelle "Opc.IDiscovery" des Dienstes "OpcEnum.exe" abgefragt, die die Spezifikation "Data Access 2.0" unterstützen. Welche OPC Spezifikationen zur Auswahl stehen, ist in der Struktur "Opc.Spezifikation" definiert.

Die durch das Discovery Interface bereitgestellte Liste enthält den vollständigen URL-Pfad mit "ProgID" (ohne Versionsindex) und "GUID" zur eindeutigen Identifikation des Servers.

```

// Step: 2
//
// browse for available opc servers on this system
private void Form_OpcClient_Load ( object sender, EventArgs e )
{
    try
    {
        // Step: 2
        //
        // browse for available opc servers on this system
        Opc.IDiscovery opcDiscovery = new OpcCom.ServerEnumerator ();
        Opc.Server[] opcServers = opcDiscovery.GetAvailableServers (
            Opc.Specification.COM_DA_20);

        // copy opc server to ...
        this.ComboBox_Server.BeginUpdate ();

        foreach ( Opc.Server opcServer in opcServers )
        {
            this.ComboBox_Server.Items.Add ( opcServer.Url.Path );
        }
    }
    catch ( Exception except )
    {
    }
}

```

Abb. 6-31: Installierte OPC Server auflisten

OPC Server Instanz erstellen und löschen**Allgemein**

In diesem Abschnitt wird das Erzeugen und Löschen der OPC Server Instanz sowie das Auslesen der Server Informationen beschrieben. Die Verwaltung

der OPC Server Instanz erfolgt im Ereignishandler "Button_Connect_Disconnect_Click" der Schaltfläche "**Connect**" bzw. "**Disconnect**" und das Auslesen und Anzeigen der Server Informationen in der Funktion "Button_ServerInfo_Click" der Schaltfläche "**Info ...**".

Die relevanten Codesegmente im OPC Client Beispielprojekt sind:

- "**Step: 3**" - OPC Server Instanz erzeugen
- "**Step: 4**" - OPC Server Instanz löschen
- "**Step: 5**" und "**Step: 21**" - Server Informationen lesen

OPC Server Instanz erzeugen - Connect

Das Erzeugen einer OPC Server Instanz und das Herstellen der Verbindung ist im "**Step: 3**" beschrieben.

Im ersten Schritt wird eine OPC Server Instanz mit "new Opc.Da.Server" und Übergabe des Parameters "OpcCom.Factory" sowie der "Opc.URL" erzeugt.

Die "Opc.URL" enthält Informationen über die verwendete OPC Spezifikation "Opc.UrlScheme.DA", den PC-Namen "localhost" oder die IP-Adresse auf dem der Server gestartet werden soll sowie die Serverkennung. Im Beispiel wird die Serverkennung aus dem Kombinationslistenfeld "ComboBox_Server" ausgelesen.

Für die Angabe der Server Kennung sind die folgenden Varianten möglich:

- "**OPC.lwSCP**"
ProgID ohne Versionsindex
- "**OPC.lwSCP.1**"
ProgID mit Versionsindex
- "**OPC.lwSCP/{108fb1cf-f509-4d86-b1da-54bf1dd67a8d}**"
ProgID ohne Versionsindex und GUID
- "**OPC.lwSCP.1/{108fb1cf-f509-4d86-b1da-54bf1dd67a8d}**"
ProgID mit Versionsindex und GUID

Mit dem Aufruf der Funktion "this.m_OpcServer.Connect()" wird der OPC Server instanziiert.

OPC Server Instanz löschen - Disconnect

Das Löschen der OPC Server Instanz ist im "**Step: 4**" dargestellt.

Mit dem Aufruf der Funktion "this.m_OpcServer.Disconnect()" erfolgt die Abmeldung am OPC Server. Ist zu diesem Zeitpunkt kein weiterer Client mit dem Server verbunden, wird dieser beendet.

Anschließend erfolgt der Aufruf der Funktion "this.m_OpcServer.Dispose()" zur Freigabe von Ressourcen, die in der OPC Server Instanz gehalten werden sowie das Löschen der Instanz durch "this.m_OpcServer = null".

Client-Entwicklung

```

try
{
    // show hour glas
    this.Cursor = Cursors.WaitCursor;

    if ( this.Button_Connect_Disconnect.Text.Equals (BUTTON_DESCRIPTION
    {
        // Step: 3
        //
        // select opc server and
        this.m_OpcServer = new Opc.Da.Server ( new OpcCom.Factory(),
            new Opc.URL(Opc.UriScheme.DA + "://localhost/" +
                this.ComboBox_Server.Text) );

        // connect with ...
        this.m_OpcServer.Connect();

        // Enable or disable controls
        this.ComboBox_Server.Focus();
        this.SetInternalStateControl(InternalStateControl.Connected);
    }
    else if ( this.Button_Connect_Disconnect.Text.Equals (BUTTON_DESCRI
    {
        // Step: 4
        //
        // disconnect from server and clean up
        this.m_OpcServer.Disconnect();
        this.m_OpcServer.Dispose();
        this.m_OpcServer = null;

        // Enable or disable controls
        this.SetInternalStateControl(InternalStateControl.Disconnected);

        // clear browse info
        if ( this.m_DialogAddItems is Form_AddItems )
            this.m_DialogAddItems.Clear();
    }
    }
}

```

Abb. 6-32: OPC Server Instanz erzeugen und löschen

Server Informationen lesen

Die durch den OPC Server bereit gestellten Informationen werden im "Load Ereignis" **"Step: 21"** der Form "Form_ServerInfo" ausgelesen und zur Anzeige gebracht.

Der Aufruf der Form erfolgt im **"Step: 5"** (ohne Abbildung).

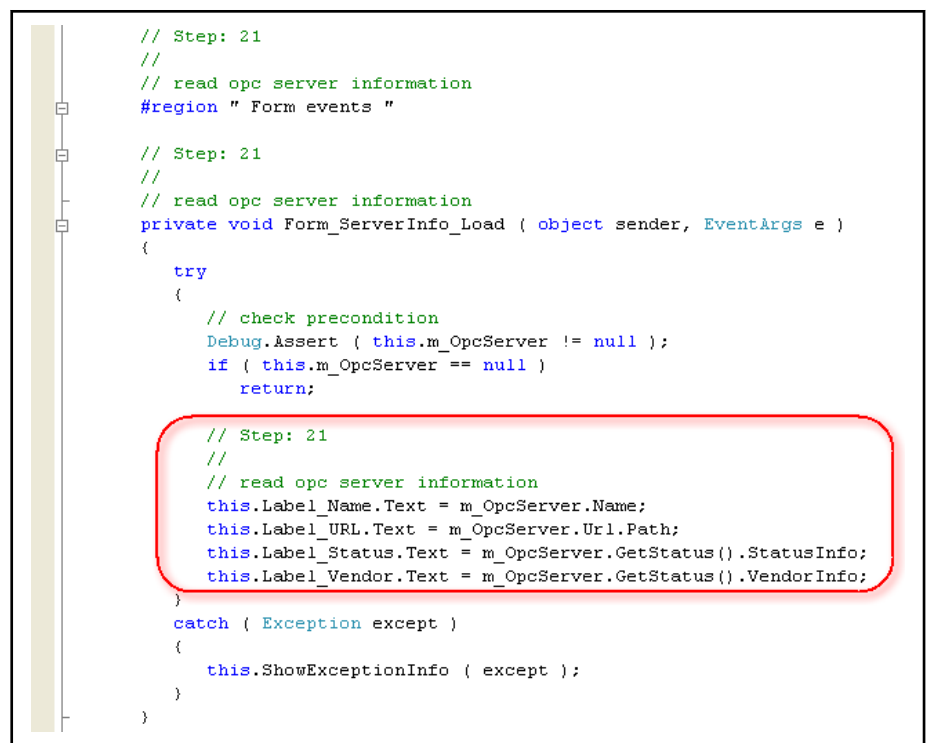


Abb. 6-33: Auslesen der OPC Server Informationen

OPC Subscription erstellen und löschen

Allgemein

Dieser Abschnitt beschreibt das An- und Abmelden einer "Subscription" (Gruppe) am OPC Server. Die Verwaltung der Subscription erfolgt im Ereignishandler "Button_Add_Remove_Group_Click" der Schaltfläche **"Add"** bzw. **"Remove"** und das Auslesen und Anzeigen der Subscriptioninformationen in der Funktion "Button_GroupInfo_Click" der **"Info ..."**-Schaltfläche.

Die relevanten Code-Segmente im OPC Client Beispielprojekt sind:

- **"Step: 6"** - Subscription erstellen
- **"Step: 7"** - Subscription löschen
- **"Step: 8"** und **"Step: 22"** - Subscription Informationen anzeigen

Subscription erstellen - CreateSubscription

Mit der Funktion "this.m_OpcServer.CreateSubscription()" und Übergabe des Parameters "Opc.Da.SubscriptionState" wird eine Subscription in der Collection vom OPC Server erstellt (**"Step: 6"**, siehe [Abb. 6-34 "OPC Subscription erstellen" auf Seite 216](#)).

Die Elemente der Klasse "Opc.Da.SubscriptionState" dienen zur eindeutigen Identifikation der Subscription und zur Konfiguration bzgl. dem Verhalten bzw. der Bearbeitung durch den OPC Server.

Eine Beschreibung der Klassenelemente ist in der folgenden Tabelle dargestellt.

Klassenelement	Beschreibung
Active	Die Items der Subscription werden zyklisch von der Steuerung gelesen und im Cache vom Server gespeichert.
ClientHandle	"Unique Identifier" (wird vom Client vergeben).

Client-Entwicklung

Klassenelement	Beschreibung
Deadband	Kleinstmögliche prozentuale Änderung eines Wertes, damit ein Ereignis zum Client ausgelöst wird.
KeepAlive	Größtmöglicher Zeitabstand zwischen zwei Ereignissen zum Client (Lebenszeichen).
Locale	Definition der Lokalisierung im Format "[language Code]-[country/region code]", z.B. "de-DE", "en-US" ...
Name	"Unique Name" (wird vom Client vergeben). Wird durch das Textfeld "TextBox_Group.Text" vorgegeben.
ServerHandle	"Unique Identifier" (wird vom Server vergeben).
UpdateRate	Zeitabstand in dem der Server die Daten prüft und ein Ereignis an den Client sendet.

Tab. 6-9: Elemente der Klasse "Opc.Da.SubscriptionState"

```

// Step: 6 / 7
//
// create and cancel subscription
private void Button_Add_Remove_Group_Click ( object sender, EventArgs e )
{
    try
    {
        // show hour glas
        this.Cursor = Cursors.WaitCursor;

        if ( this.Button_Add_Remove_Group.Text.Equals ( BUTTON_DESCRIPTION_GROUP_ADD ) )
        {
            // Step: 6
            //
            // create subscription
            Opc.Da.SubscriptionState subSubscriptionState = new Opc.Da.SubscriptionState();
            subSubscriptionState.Active = true; // activate cyclic communication
            subSubscriptionState.ClientHandle = System.Guid.NewGuid().ToString();
            subSubscriptionState.Deadband = 0.0F;
            subSubscriptionState.Name = TextBox_Group.Text;
            subSubscriptionState.UpdateRate = 1000;

            this.m_OpcServer.CreateSubscription(subSubscriptionState);

            // Enable or disable controls
            this.SetInternalStateControl ( InternalStateControl.SubscriptionSet );
        }
        else if ( this.Button_Add_Remove_Group.Text.Equals ( BUTTON_DESCRIPTION_GROUP_RE
    {

```

Abb. 6-34: OPC Subscription erstellen

Subscription löschen - CancelSubscription

Zum Löschen einer Subscription wird die Funktion

"this.m_OpcServer.CancelSubscription()" mit Übergabe des Parameters "Opc.Da.ISubscription" aufgerufen (**"Step: 7"**).

Nach dem Löschen der Subscription aus der Collection des OPC Servers ist es wichtig, die Funktion "iSubscription.Dispose()" aufzurufen, damit die von der Subscription belegten Ressourcen wieder freigegeben werden. Durch die Zuweisung "iSubscription = null" wird die Referenz der Schnittstellenvariablen gelöscht.

Die zum Beginn des Code-Segments durchgeführte lokale Zuweisung "iSubscription = this.m_OpcServer.Subscription[0]" ist notwendig, da nach dem Aufruf der Funktion "this.m_OpcServer.CancelSubscription()" die Subscription aus der Collection des OPC Servers entfernt wurde und nicht mehr zur Verfügung steht.

```

    }
    else if ( this.Button_Add_Remove_Group.Text.Equals ( BUTTON_DESCRIPTION_GROU
    {
        // Step: 7
        //
        // cancel subscription and clean up
        Opc.Da.ISubscription iSubscription = this.m_OpcServer.Subscriptions[0];
        this.m_OpcServer.CancelSubscription(iSubscription);
        iSubscription.Dispose();
        iSubscription = null;

        // Enable or disable controls
        this.SetInternalStateControl ( InternalStateControl.Connected );
    }
    }
    catch ( Exception except )

```

Abb. 6-35: OPC Subscription löschen

Subscription Informationen anzeigen

Die von der Subscription bereitgestellten Informationen werden im "Load Ereignis" ("Step: 22") der Form "Form_GroupInfo" ausgelesen und zur Anzeige gebracht.

Der Aufruf der Form erfolgt im "Step: 8" (ohne Abbildung).

```

// check precondition
Debug.Assert ( this.m_OpcServer != null );
if ( this.m_OpcServer == null )
    return;

// Step: 22
//
// read opc subscription information
Opc.Da.Subscription subSubscription = m_OpcServer.Subscriptions[0];
this.Label_Name.Text = subSubscription.Name;
this.Label_Deadband.Text = subSubscription.GetState ( ).Deadband.ToString ( ) + " %";
this.Label_UpdateRate.Text = subSubscription.GetState ( ).UpdateRate.ToString ( ) + " ms";
}
catch ( Exception except )

```

Abb. 6-36: Auslesen der OPC Subscription Informationen

Steuerungsisems auflisten

Allgemein

Dieser Abschnitt beschreibt das Auslesen aller in der Steuerung für den Zugriff über den OPC Server konfigurierten Items.

Die relevanten Codesegmente im OPC Client Beispielprojekt sind:

- "Step: 20", "Step: 20.1" und "Step: 20.2" - Steuerungsisems lesen

Steuerungsisems lesen

Mit dem Betätigen des Buttons "Add ..." der Form "Form_OpcClient" wird die Form "Form_AddItems" aufgerufen. Da das Lesen der Items unter Umständen lange dauern kann, erfolgt das Einlesen der am OPC Server verfügbaren Items nicht im "Load Ereignis" der Form, sondern über die Funktion "Button_BrowseItems_Click", die durch einen Klick auf die Schaltfläche "Browse / Refresh" ausgelöst wird.

Zum Auslesen der Items werden die beiden Klassen "Opc.ItemIdentifier" und "Opc.Da.BrowseFilters" der Funktion "this.m_OPCTServer.Browse()" als Parameter übergeben ("Step: 20.2").

Hierbei haben die Klassen folgende Funktion:

Opc.ItemIdentifier

Mit der Klasse "Opc.ItemIdentifier" wird der "Startpunkt" der Suche in der "Baum Hierarchie" der Items vorgegeben. D.h. wurde z.B. eine Suche abgebrochen, kann die Suche wieder an diesem Punkt fortgesetzt werden.

Opc.Da.BrowseFilter

Die Klasse "Opc.Da.BrowseFilter" dient als Filterkriterium für das Ergebnis des Funktionsaufrufs. Hier besteht die Möglichkeit nur Variablen zu lesen, die z.B. mit einer bestimmten Zeichensequenz beginnen.

Client-Entwicklung

Da es keine Vorgaben gibt, in welcher Form ein OPC Server die "Browsing" Information ausliest, wurde das Lesen der Steuerungsisems in zwei Funktionen ("Step: 20.1" und "Step: 20.2") aufgeteilt. Mit dieser Aufteilung können flache Hierarchien (Baumstruktur ohne Unterknoten) und strukturierte Hierarchien (Baumstruktur mit Unterknoten) ausgelesen werden (siehe nachfolgende Abbildungen).

```
// clear list, stop updating
this.ListBox_Available_Items.BeginUpdate ( );
this.ListBox_Available_Items.Items.Clear ( );

// Step: 20.1
//
// browse items
Opc.Da.BrowseFilters filters = new Opc.Da.BrowseFilters ( );
filters.ElementNameFilter = "";
filters.BrowseFilter = Opc.Da.BrowseFilter.all;
filters.ReturnAllProperties = false;
filters.ReturnPropertyValues = false;

Opc.ItemIdentifier itemID = new Opc.ItemIdentifier ( );
itemID.ItemName = "";

this.BrowseItems ( itemID, filters );

}
catch ( Exception except )
```

Abb. 6-37: Steuerungsisems lesen - Filter Konfiguration

```
// Step: 20.2
//
// browse items
opcBrowseElements = this.m_OpcServer.Browse ( itemID, filters, out opcBrowsePosition );

// adds each element to the list
foreach (Opc.Da.BrowseElement opcBrowseElement in opcBrowseElements)
{
    //
    // avoid errors while using .NET API 1.30
    if ( opcBrowseElement.ItemName != null )
        itemName = opcBrowseElement.ItemName;
    else
        itemName = opcBrowseElement.Name;

    this.ListBox_Available_Items.Items.Add ( itemName );

    // check on subordinate items
    if (opcBrowseElement.HasChildren == true)
    {
        // set branch to search
        opcItemID.ItemName = itemName; // opcBrowseElement.ItemName;
        this.BrowseItems ( opcItemID, filters );
    }
}
```

Abb. 6-38: Steuerungsisems lesen - Rekursion

OPC Items hinzufügen und löschen

Allgemein

In diesem Abschnitt wird das Hinzufügen und Entfernen von Items zu einer Subscription beschrieben.

Die relevanten Codesegmente im OPC Client Beispielprojekt sind:

- "Step: 9", "Step: 9.1" und "Step: 9.2" - Items hinzufügen
- "Step: 10" - Items löschen

Items hinzufügen - AddItems

In dem Form-Ereignis "Button_Add_Item_Click" ("Step: 9") erfolgt der Aufruf der Form "Form_AddItems" und das Hinzufügen der in der Form ausgewählten Items.

Die Variablen können im Textfeld der Form "Add Items" durch ";" getrennt eingegeben oder im Listenfeld ausgewählt werden. Bei der Eingabe der Variablen ist die Item-Syntax des jeweiligen OPC Servers zu beachten.

Im folgenden ist die Item-Syntax für den "OPC.lwSCP" Server und dem "IndraLogic OPC" Server dargestellt.

"OPC.lwSCP" Server

- **SPS Variablen**

"![angeforderter Datentyp],[Steuerungsname oder Alias],[lwSCP Dienst],[voll qualifizierender Variablenname]"

Beispiel:

"!Bool,ImMlc2G,Plc.PVL,Application.HMIVARS.bLifeSign_hmi"

- **sercos Parameter**

"![angeforderter Datentyp],[Steuerungsname oder Alias],[lwSCP Dienst],[Antriebs Adresse],[Parameter Nummer],[Elementkodierung]"

Beispiel:

"!UI4,ImMlc2G,SPA1,0,C-0-0001,D"

"!UI4,ImMlc2G,SPA1,0,C-0-0001,40"



Der "OPC.lwSCP" Server ermöglicht den Zugriff auf Parameter und SPS Variablen!



Weitere Informationen zu der Item-Syntax des "OPC.lwSCP" Servers finden Sie in [Kap. 4.3.3 "lwSCP und OPC.lwSCP Items" auf Seite 98](#).

"IndraLogic OPC" Server

- **SPS Variablen**

"[Steuerungsname].[voll qualifizierender Variablenname]"

Beispiel:

"PLC_IL2G.Application.HMIVARS.bLifeSign_hmi"



Mit dem "IndraLogic OPC" Server ist der Zugriff auf SPS Variablen möglich.

Die Beschreibung der "Browse"-Funktionalität ist in [Kap. "Steuerungselemente auflisten" auf Seite 217](#) beschrieben.

Wurden die Items in der Form "Add Items" eingegeben bzw. ausgewählt, wird für jedes Item im **"Step: 9.1"** eine Klasse "Opc.Da.Item" erstellt und initialisiert. Der Name des Items wird in der Zeile "opclItems[index].ItemName = items[index]" zugewiesen.

Die im Beispiel verwendeten Klassenelemente der Klasse "Opc.Da.Item" sind in der folgenden Tabelle dargestellt:

Klassenelement	Beschreibung
Active	Für das Item werden "DataChanged" Ereignisse an den Client gesendet.
ActiveSpecified	Die Eigenschaft "Active" wurde definiert.
ItemName	Voll qualifizierter Item-Name. Die Syntax ist abhängig vom OPC Server.
ItemPath	Voll qualifizierter Item-Pfad. Diese Eigenschaft wird vom "OPC.lwSCP" nicht ausgewertet.

Client-Entwicklung

Klassenelement	Beschreibung
MaxAge	Beim Lesen des Items aus dem Cache des OPC Servers darf der Wert nicht älter sein, als in "MaxAge" definiert.
MaxAgeSpecified	Die Eigenschaft "MaxAge" wurde definiert. Ist diese Eigenschaft auf "True" gesetzt, wird in Abhängigkeit von "MaxAge" der Wert aus dem "Cache" des OPC Servers oder vom "Device" gelesen. Mit dem Setzen der Eigenschaft auf "False" wird immer vom "Device" gelesen.

Tab. 6-10: Elemente der Klasse "Opc.Da.Item"

Nachdem alle OPC Items erstellt wurden, wird das OPC Item-Array mit dem Funktionsaufruf "this.m_OpcServer.Subscriptions[0].AddItems()" in der Subscription am OPC Server angemeldet ("Step: 9.2").

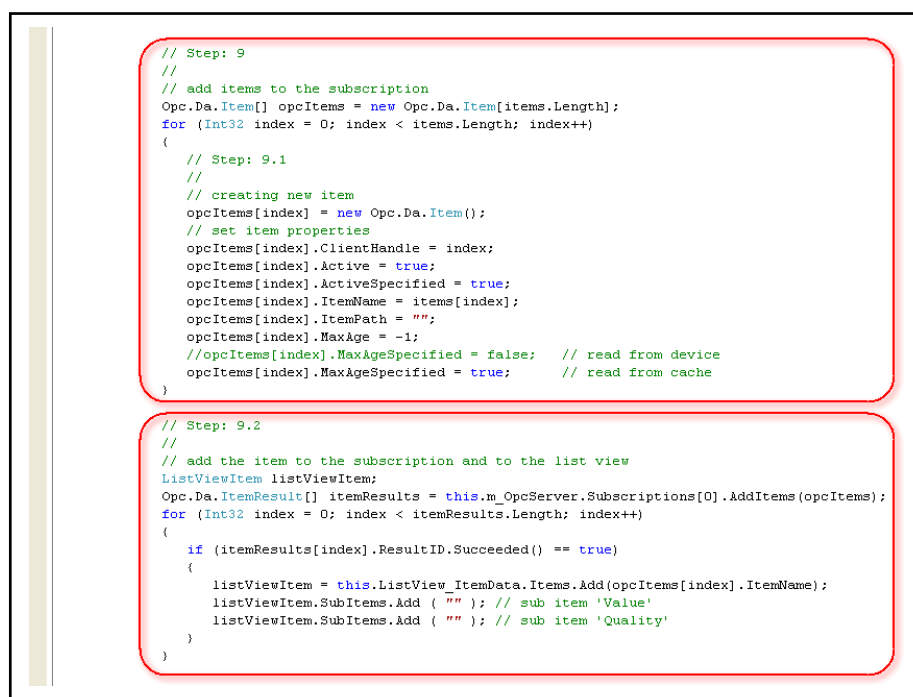


Abb. 6-39: Items einer Subscription hinzufügen

Items löschen - RemoveItems

Zum Löschen der Items aus der OPC Subscription werden diese in der Listenansicht ausgewählt und mit Betätigen der "Remove"-Schaltfläche entfernt.

In dem daraufhin ausgelösten Form-Ereignis "Button_Remove_Item_Click" wird die Funktion "this.m_OpcServer.Subscriptions[0].RemoveItems()" aufgerufen, die das als Parameter übergebene OPC Item entfernt ("Step: 10").

```

// Step: 10
//
// remove selected items from opc subscription and from the list
for ( Int32 index = 0; index < selectedItems; index++ )
{
    foreach (Opc.Da.Item opcItem in this.m_OpcServer.Subscriptions[0].Items)
    {
        if ( this.ListView_ItemData.SelectedItems[index].Text.Equals ( opcItem.ItemName ) )
        {
            opcItemIds[index] = opcItem;
            break;
        }
    }
}
// remove items
Opc.IdentifiedResult[] opcIdentResult =
this.m_OpcServer.Subscriptions[0].RemoveItems ( opcItemIds );

// clean up list view control
for ( Int32 index = 0; index < opcIdentResult.Length; index++ )
{
    if ( opcIdentResult[index].ResultID == ResultID.S_OK )
    {
        foreach ( ListViewItem listViewItem in this.ListView_ItemData.SelectedItems )
        {
            if ( opcIdentResult[index].ItemName.Equals ( listViewItem.Text ) )
            {
                this.ListView_ItemData.Items.Remove ( listViewItem );
                break;
            }
        }
    }
}

```

Abb. 6-40: Items von einer Subscription entfernen



Das Entfernen von Items aus einer Gruppe, sollte immer für eine gesamte Liste von Items durchgeführt werden, damit im OPC Server die Verwaltungsarbeiten zum ab- und anmelden der Gruppe nur einmalig entstehen.

Steuerungsdaten lesen

Allgemein

Dieser Abschnitt beschreibt den lesenden Zugriff auf die Steuerungsdaten.

Hierbei wird zwischen einem

- "synchrone" und
- "asynchrone" Zugriff unterschieden.

Das Lesen der Steuerungsdaten erfolgt im Ereignishandler "Button_Read_Click" der "Read"-Schaltfläche.

Die relevanten Codesegmente sind:

- "Step: 11" - synchrones Lesen
- "Step: 12" - asynchrones Lesen
- "Step: 13" - Daten auswerten

Synchrones Lesen

Bei einem "synchrone" Zugriff wartet der Client solange, bis der Server die Daten bereitgestellt hat. In Abhängigkeit des im "Opc.Da.Item" definierten Verhalten, werden die Daten aus dem internen "Cache" des OPC Servers oder von der Steuerung "Device" gelesen. Der Client nimmt in diesem Zeitraum **keine weiteren Bedienhandlungen an**, d.h. der "synchrone" Aufruf ist blockierend.



Das Lesen von Daten aus dem "Cache" ist in der Regel immer schneller als das Lesen vom "Device". Dieses Verhalten sollte bei der Konzeption eines OPC Clients immer berücksichtigt werden.

Das "synchrone" Lesen ("**Step: 11**") von Daten einer Subscription erfolgt über den Funktionsaufruf
 "opcValueResults = this.m_OpcServer.Subscriptions[0].Read()" und der

Client-Entwicklung

Übergabe der zu lesenden Items "this.m_OpcServer.Subscriptions[0].Items" als Parameter.

In unserem Beispiel wird die gesamte Gruppe gelesen und das Ergebnis an die Funktion "OnReadComplete" zur Verarbeitung weitergegeben.

Die Beschreibung der Funktion "OnReadComplete" folgt in [Kap. "Daten auswerten" auf Seite 223](#).

Asynchrones Lesen

Bei einem "asynchronen" Zugriff wird der Auftrag an den Server übergeben und der Funktionsaufruf kehrt sofort zurück. Der Server bearbeitet anschließend den Auftrag und meldet das Ergebnis der Auftragsbearbeitung an den Client über einen zuvor definierten "Delegate" (Ereignishandler "OnReadComplete") zurück, d.h. der Client kann während der Auftragsbearbeitung durch den Server weitere **Bedienkommandos annehmen und verarbeiten**.

Das "asynchrone" Lesen ("**Step: 12**") erfolgt wie beim "synchrone" Lesen über den Funktionsaufruf "opcValueResults = this.m_OpcServer.Subscriptions[0].Read()".

Der Aufruf unterscheidet sich nur in der Anzahl der zu übergebenden Parameter. Die Übergabeparameter und ihre Beschreibung sind in der folgenden Tabelle aufgelistet.

Übergabeparameter	Beschreibung
Opc.Da.Item[]	Item-Collection aller zu lesenden Items. Wie beim synchronen Lesen.
Object	Vom Client übergebene ID die dem "Delegate" bei der Verarbeitung der Ergebnisse mitgegeben wird (optional "null").
ReadCompleteEventHandler	"Delegate" für die Verarbeitung der Ergebnisse der Anfrage. Der "Delegate" muss mit der Anweisung "new ReadCompleteEventHandler(OnReadComplete)" an den Funktionsaufruf übergeben werden.
IRequest	Handle zum Abbrechen der Anforderung über den Funktionsaufruf "Subscription.Cancel()" (optional "null").

Tab. 6-11: Übergabeparameter für das asynchrone Lesen

```
// check command type - sync or async
if (this.RadioButton_Sync.Checked == true)
{
    // Step: 11
    //
    // synchronous read of data
    Opc.Da.ItemValueResult[] opcValueResults = this.m_OpcServer.Subscriptions[0].Read(
        this.m_OpcServer.Subscriptions[0].Items);
    this.OnReadComplete(null, opcValueResults);
}
else if (this.RadioButton_Async.Checked == true)
{
    // Step: 12
    //
    // asynchronous read of data
    Opc.IRequest opcRequest;
    this.m_OpcServer.Subscriptions[0].Read ( this.m_OpcServer.Subscriptions[0].Items,
        null, new Opc.Da.ReadCompleteEventHandler ( OnReadComplete ), out opcRequest );
}
```

Abb. 6-41: Synchrones und asynchrones Lesen von Steuerungsdaten

Daten auswerten

Die Funktion "OnReadComplete" zum Auswerten der vom OPC Server zurückgelieferten Daten wird nach dem "synchronen" Lesen vom OPC Client und als "Ereignishandler" ("Delegate") vom OPC Server, mit der Bereitstellung der Daten nach der "Asynchronen" Anforderung, aufgerufen.

Da Steuerelemente nicht "threadsicher" sind, d.h. der Zugriff auf diese Objekte nur von einem Thread erfolgen kann, muss geprüft werden, ob der Funktionsaufruf "OnReadComplete" im gleichen Thread läuft wie der Thread der Form. Diese Überprüfung erfolgt im **"Step: 13.1"**. Wurde die Funktion nicht im Thread der Form aufgerufen, wird der Aufruf durch die Funktion "this.BeginInvoke()" synchronisiert.

Im **"Step: 13.2"** werden die Ergebnisse ausgewertet und in der Listenansicht angezeigt.

```
try
{
    // Step: 13.1
    //
    // check if it has to be synchronized with the thread of the form
    if ( this.InvokeRequired == true )
    {
        this.BeginInvoke ( new Opc.Da.ReadCompleteEventHandler ( OnReadComplete ),
            new Object[] { clientHandle, opcValueResults } );
        return;
    }

    // stop updating
    this.ListView_ItemData.BeginUpdate();

    // Step: 13.2
    //
    // update item values
    ListViewItem listViewItem;
    foreach (Opc.Da.ItemValueResult opcValueResult in opcValueResults)
    {
        listViewItem = this.ListView_ItemData.FindItemWithText ( opcValueResult.ItemName );

        if ( ( listViewItem != null ) && ( opcValueResult.Value != null ) )
        {
            listViewItem.SubItems[1].Text = this.GetValueAsString ( opcValueResult );
            listViewItem.SubItems[2].Text = opcValueResult.Quality.ToString ();
        }
    }
}
catch ( Exception except )
```

Abb. 6-42: Daten auswerten - lesend

Steuerungsdaten schreiben

Allgemein

Dieser Abschnitt beschreibt den schreibenden Zugriff auf die Steuerungsdaten.

Hierbei wird zwischen einem

- "synchronen" und
- "asynchronen" Zugriff unterschieden.

Das Schreiben der Steuerungsdaten erfolgt im Ereignishandler "Button_Write_Click" der **"Write"**-Schaltfläche.

Die relevanten Code-Segmente im OPC Client Beispielprojekt sind:

- **"Step: 14"** - synchrones Schreiben
- **"Step: 15"** - asynchrones Schreiben
- **"Step: 16"** - Daten auswerten

Synchrones Schreiben

Bei einem "synchronen" Zugriff wartet der Client solange, bis der Server den Schreibauftrag beendet hat. Das Schreiben der Daten erfolgt im "Cache" und im "Device" des OPC Servers, wenn die Subscription und das Item aktiv ist.

Client-Entwicklung

Der Client nimmt in diesem Zeitraum **keine weiteren Bedienhandlungen an**, d.h. der "synchrone" Aufruf ist blockierend.

Das "synchrone" Schreiben ("**Step: 14**") von Daten einer Subscription erfolgt über den Funktionsaufruf "opcResults = this.m_OpcServer.Subscriptions[0].Write()" und der Übergabe der zu schreibenden "Opc.Da.ItemValue" ("opcValues") als Parameter.

In unserem Beispiel wird nur ein Item von der gesamten Subscription geschrieben und das Ergebnis des Schreibauftrages an die Funktion "OnWriteComplete" zur Verarbeitung weitergegeben.

Die Beschreibung der Funktion "OnWriteComplete" folgt in [Kap. "Daten auswerten" auf Seite 225](#).

Asynchrones Schreiben

Bei einem "asynchronen" Zugriff wird der Auftrag an den Server übergeben und der Funktionsaufruf kehrt sofort zurück. Der Server bearbeitet anschließend den Auftrag und meldet das Ergebnis der Auftragsbearbeitung an den Client über einen zuvor definierten "Delegate" (Ereignishandler "OnWriteComplete") zurück. D.h. der Client kann während der Auftragsbearbeitung durch den Server weitere **Bedienkommandos annehmen und verarbeiten**.

Das "asynchrone" Schreiben ("**Step: 15**") erfolgt wie beim "synchrone" Schreiben über den Funktionsaufruf "opcResults = this.m_OpcServer.Subscriptions[0].Write()".

Der Aufruf unterscheidet sich nur in der Anzahl der zu übergebenden Parameter. Die Übergabeparameter und ihre Beschreibung sind in der folgenden Tabelle aufgelistet.

Übergabeparameter	Beschreibung
Opc.Da.ItemValue()	Item-Collection aller zu schreibenden Items mit den Werten. Wie beim synchronen Schreiben.
Object	Vom Client übergebene ID die dem "Delegate" bei der Verarbeitung der Ergebnisse mitgegeben wird (optional "null").
WriteCompleteEventHandler	"Delegate" für die Verarbeitung der Ergebnisse des Auftrags. "Delegate" muss mit der Anweisung "new WriteCompleteEventHandler(OnWriteComplete)" an den Funktionsaufruf übergeben werden.
IRequest	Handle zum Abbrechen des Auftrags über den Funktionsaufruf "Subscription.Cancel()" (optional "null").

Tab. 6-12: Übergabeparameter für das asynchrone Schreiben

```

// check command type - sync or async
if ( this.RadioButton_Sync.Checked == true )
{
    // Step: 14
    //
    // synchronous write
    Opc.IdentifiedResult[] opcIdentResults =
        this.m_OpcServer.Subscriptions[0].Write ( opcItemValues );
    this.OnWriteComplete( null, opcIdentResults );
}
else if ( this.RadioButton_Async.Checked == true )
{
    // Step: 15
    //
    // asynchronous write
    Opc.IRequest opcRequest;
    this.m_OpcServer.Subscriptions[0].Write ( opcItemValues, null,
        new Opc.Da.WriteCompleteEventHandler ( OnWriteComplete ), out opcRequest );
}

```

Abb. 6-43: Synchrones und asynchrones Schreiben von Steuerungsdaten

Daten auswerten

Die Funktion "OnWriteComplete" dient der Auswertung der vom OPC Server zurückgelieferten Ergebnisse. Sie wird nach dem "synchrone" Schreiben vom OPC Client und als "Ereignishandler" ("Delegate") vom OPC Server, mit dem Abschluss der Auftragsbearbeitung nach dem "asynchronen" Kommando, aufgerufen.

Da Steuerelemente nicht "threadsicher" sind, d.h. der Zugriff auf diese Objekte nur von einem Thread erfolgen kann, muss geprüft werden, ob der Funktionsaufruf "OnWriteComplete" im gleichen Thread läuft, wie der Thread der Form. Diese Überprüfung erfolgt im **"Step: 16.1"**.

Wurde die Funktion nicht im Thread der Form aufgerufen, wird der Aufruf durch die Funktion "this.BeginInvoke()" synchronisiert.

Im **"Step: 16.2"** kann eine Auswertung der Ergebnisse des Schreibauftrags erfolgen. Diese Auswertung kann in einer "Log-Datei" oder z.B. in einem "StatusStrip"-Steuerelement der Applikation erfolgen. Der Beispiel Client enthält keine Auswertung der Ergebnisse des Schreibauftrags.

```

// Step: 16
//
// This method is a delegate, which will be executed, if the
// asynchronous write is finished
private void OnWriteComplete ( Object clientHandle, Opc.IdentifiedResult[] opcIdentResults )
{
    try
    {
        // Step: 16.1
        //
        // Check if it has to be synchronized with the thread of the form
        if ( this.InvokeRequired == true )
        {
            this.BeginInvoke ( new Opc.Da.WriteCompleteEventHandler ( OnWriteComplete ),
                new Object[] { clientHandle, opcIdentResults } );
            return;
        }

        // Step: 16.2
        //
        // show result
        Boolean error = false;
        StringBuilder errorInfo = new StringBuilder ( );
        errorInfo.Remove ( 0, errorInfo.Length );
        errorInfo.AppendLine ( "The following variables could not be written in case of error!" );
        foreach ( Opc.IdentifiedResult opcIdentResult in opcIdentResults )
        {
            if ( opcIdentResult.ResultID != Opc.ResultID.S_OK )
            {
                error = true;
                errorInfo.AppendFormat ( "Item Name: {0}(1)Error Info: {2}(3)Error Number: {4}(5)",
                    opcIdentResult.ItemName, Environment.NewLine,
                    opcIdentResult.DiagnosticInfo, Environment.NewLine,
                    opcIdentResult.ResultID, Environment.NewLine );
            }
        }
        if ( error )
            MessageBox.Show ( errorInfo.ToString ( ), "Error while writing variables",

```

Abb. 6-44: Daten auswerten - schreibend

Zyklisches Lesen der Steuerungsdaten

Allgemein

In diesem Abschnitt wird das zyklische Lesen von Steuerungsdaten beschrieben.

Voraussetzung für das zyklische Lesen von Werten ist, dass

- die Subscription und
- das Item aktiv sowie
- ein Eventhandler an der Subscription angemeldet ist.

Das Aktivieren und Deaktivieren des zyklischen Lesens wird über das Ereignis "RadioButton_CheckedChanged" der "Radio Buttons" "Synchronous", "Asynchronous" und "Cyclic" gesteuert.

Die relevanten Code-Segmente im OPC Client Beispielprojekt sind:

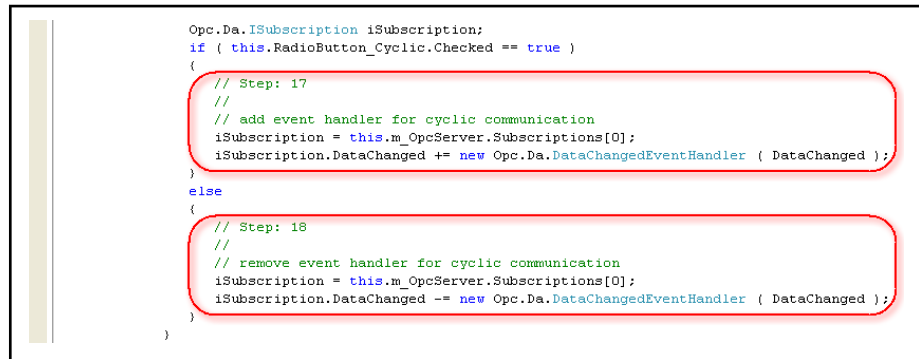
- "Step: 17" - zyklisches Lesen aktivieren
- "Step: 18" - zyklisches Lesen deaktivieren
- "Step: 19" - Daten auswerten

Zyklisches Lesen aktivieren

Da die Voraussetzungen der aktiven Subscription und aktiven Items bereits erfüllt ist, muss nur noch der Eventhandler "DataChanged" zu der Subscription Liste "iSubscription.DataChanged" hinzugefügt werden ("Step: 17"). Mit dem Hinzufügen sind alle Voraussetzungen erfüllt und der Client erhält die "DataChanged" Ereignisse.

Zyklisches Lesen deaktivieren

Zum Deaktivieren des zyklischen Lesens wird der Eventhandler "DataChanged" wieder aus der Subscription Liste "iSubscription.DataChanged" entfernt. Nach dem Entfernen werden keine weiteren Werte-Änderungen mehr an den Client gemeldet.



```

Opc.Da.ISubscription iSubscription;
if ( this.RadioButton_Cyclic.Checked == true )
{
    // Step: 17
    //
    // add event handler for cyclic communication
    iSubscription = this.m_OpcServer.Subscriptions[0];
    iSubscription.DataChanged += new Opc.Da.DataChangedEventHandler ( DataChanged );
}
else
{
    // Step: 18
    //
    // remove event handler for cyclic communication
    iSubscription = this.m_OpcServer.Subscriptions[0];
    iSubscription.DataChanged -= new Opc.Da.DataChangedEventHandler ( DataChanged );
}
    
```

Abb. 6-45: Ereignisbehandlung hinzufügen und entfernen

Daten auswerten

Die Funktion "DataChanged" wird vom OPC Server zyklisch in den Zeitabständen aufgerufen, wie es im Element "UpdateRate" der Klasse "Opc.Da.SubscriptionState" festgelegt wurde.

Es werden alle Items mit den zugehörigen Werten der "DataChanged" Funktion übergeben, wenn

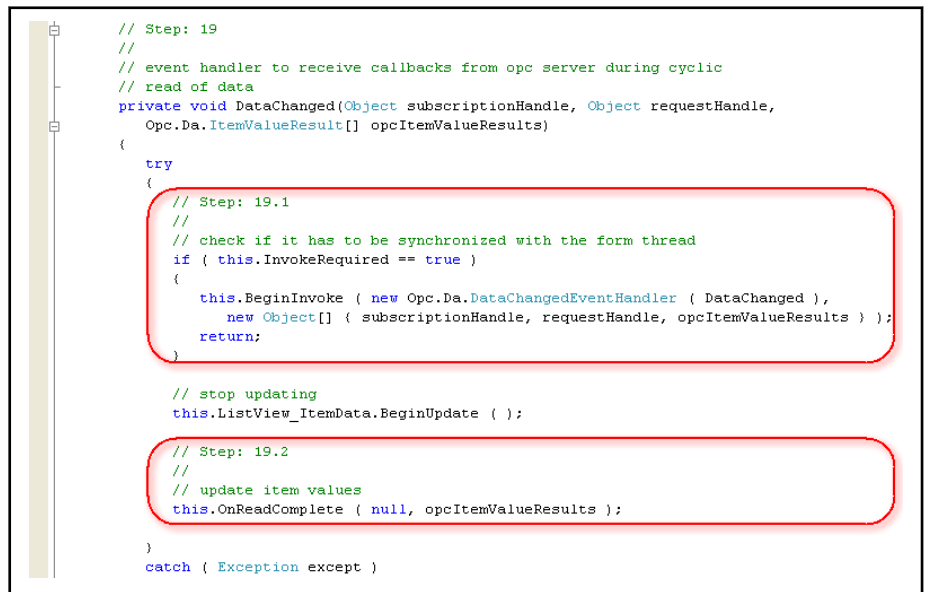
- sich der Wert geändert hat oder
- das maximale "Alter" des Wertes überschritten ist.

Das heißt in Abhängigkeit der Konfiguration der Elemente "MaxAge" und "MaxAgeSpecified" der Klasse "Opc.Da.Item".

Da Steuerelemente nicht "threadsicher" sind, d.h. der Zugriff auf diese Objekte nur von einem Thread erfolgen kann, muss geprüft werden, ob der Funktionsaufruf "DataChanged" im gleichen Thread läuft wie der Thread der Form. Diese Überprüfung erfolgt im **"Step: 19.1"**.

Wurde die Funktion nicht im Thread der Form aufgerufen, wird der Aufruf durch die Funktion "this.BeginInvoke()" synchronisiert.

Im **"Step: 19.2"** werden die Ergebnisse, wie beim synchronen und asynchronen Lesen ausgewertet und in der Listenansicht angezeigt.



```
// Step: 19
//
// event handler to receive callbacks from opc server during cyclic
// read of data
private void DataChanged(Object subscriptionHandle, Object requestHandle,
    Opc.Da.ItemValueResult[] opcItemValueResults)
{
    try
    {
        // Step: 19.1
        //
        // check if it has to be synchronized with the form thread
        if ( this.InvokeRequired == true )
        {
            this.BeginInvoke ( new Opc.Da.DataChangedEventHandler ( DataChanged ),
                new Object[] { subscriptionHandle, requestHandle, opcItemValueResults } );
            return;
        }

        // stop updating
        this.ListView_ItemData.BeginUpdate ( );

        // Step: 19.2
        //
        // update item values
        this.OnReadComplete ( null, opcItemValueResults );
    }
    catch ( Exception except )
    {
    }
}
```

Abb. 6-46: Daten auswerten - zyklisches Lesen

Fehlerbehandlung

Mit dem Beispiel OPC Client wurde ein Minimum an Fehlerhandling implementiert. Bei einem an Produktionsmaschinen eingesetzten OPC Client sollten auftretende Fehler detaillierter ausgewertet und protokolliert werden.

Der OPC Client beschränkt sich daher auf die Überprüfung von Variablen und ein einfaches Exception-Handling in der Form von

- "If"-Abfragen und
- Exception Handling mit "try...catch..."-Anweisungen

"If"-Abfragen

Mit der "If"-Abfrage werden die für die Funktionsausführung notwendigen Parameter geprüft. Können die vorgegebenen Definitionen nicht erfüllt werden, wird die Funktion an dieser Stelle abgebrochen.

Exception Handling mit "try...catch..."-Anweisungen

Da die meisten Komponenten einen Fehler in der Funktionsausführung dem Aufrufer über das "Ausgeben" einer Ausnahmemeldung mitteilen, sollte mindestens eine "try...catch..."-Anweisung in jeder Funktion/Methode enthalten sein. Im Beispiel OPC Client wurde bedingt durch den Umfang der einzelnen Funktionen nur eine "try...catch..."-Anweisung programmiert.

7 Service und Support

Für Ihre schnelle und optimale Unterstützung verfügen wir über ein dichtes weltweites Servicenetz. Unsere Experten stehen Ihnen mit Rat und Tat zur Seite. Sie erreichen uns täglich **rund um die Uhr - auch am Wochenende und an Feiertagen**.

Service Deutschland Unser technologieorientiertes Competence Center in Lohr deckt alle Belange rund um den Service für elektrische Antriebe und Steuerungen ab.

Die **Service Helpdesk & Hotline** erreichen Sie unter:

Telefon: **+49 9352 40 5060**
Fax: **+49 9352 18 4941**
E-Mail: service.svc@boschrexroth.de
Internet: <http://www.boschrexroth.com>

Auf unseren Internetseiten finden Sie ergänzende Hinweise zu Service, Reparatur (z.B. Anlieferadressen) und Training.

Service weltweit Außerhalb Deutschlands nehmen Sie bitte zuerst Kontakt mit Ihrem Ansprechpartner auf. Die Hotline-Rufnummern entnehmen Sie bitte den Vertriebsadressen im Internet.

Vorbereitung der Informationen Wir können Ihnen schnell und effizient helfen, wenn Sie folgende Informationen bereithalten:

- detaillierte Beschreibung der Störung und der Umstände
- Angaben auf dem Typenschild der betreffenden Produkte, insbesondere Typenschlüssel und Seriennummern
- Ihre Kontaktdaten (Telefon-, Faxnummer und E-Mail-Adresse)

Index

A

Abkürzungen.....	16
Ablageverzeichnis - OPC.DA Server.....	110
Windows 7.....	111
Windows CE.....	111
Windows XP.....	110
Anschlussfeld.....	28
Anwendung beenden.....	98
Applikationsbeispiele.....	141
HMIs von Fremdherstellern.....	167
IndraLogic Visualisierung.....	167
Symbolkonfiguration.....	141
Symbolkonfiguration über Pragma.....	143
Übersicht.....	141
VI-Composer.....	144
WinStudio mit IndraLogic XLC.....	144

B

Bestellinformationen.....	30
Bildgröße.....	62
BRWS Treiber	
Anwendungsgebiete.....	84
Datenmengen.....	88
Dokumentation.....	92
Eigenschaften.....	83
Erläuterungen der Meldungen.....	91
Fehlermeldungen.....	91
Konfiguration.....	84
Protokoll-Aufbau.....	87
Treiberblätter.....	85
Unterstützung SPS-Variablen und Parameter.....	88
BRWS-Treiber.....	83

D

Darstellung von Informationen	
Bezeichnungen und Abkürzungen.....	16
DC-24-V-Spannungsversorgung.....	28
Diagnoseanzeigen.....	29

E

Einführung in OPC Unified Architecture	
Allgemein.....	117
OPC UA Informationsmodell.....	118
Serviceorientierte Architektur.....	118
Übersicht der Spezifikationen.....	117
Ethernet-Schnittstelle.....	29

F

Fehlermeldungen Bosch-Rexroth-WinStudio-Treiber.....	91
--	----

G

Gehäuse - und Einbaumaße der VR 21xx.....	27
---	----

Geräte entfernen.....	97
Geräte hinzufügen.....	94
Geräte OPC Alias ändern.....	96
Geräteeigenschaften anzeigen.....	95
Gerätekonfiguration aktivieren.....	97
Gerätekonfiguration erstellen.....	93
Gerätekonfiguration speichern.....	97
Gerätekonfiguration speichern unter.....	97

I

IndraControl VCH.....	22
IndraControl VCH 08 – kompaktes Handbediengerät für mobilen Einsatz	
Ausführungen.....	23
Dokumentation.....	23
Einsatzbereich.....	22
Kurzbeschreibung.....	22
Technische Daten.....	22
Zubehör.....	23
IndraControl VCP.....	17
IndraControl VEP 30.3 / 40.3 / 50.3 / VEP 30.4 / 40.4 / 50.4 – Embedded-PC-basierte Bedienterminals	
Kurzbeschreibung.....	31
IndraControl VEP 30.5 / 40.5 / 50.5 / VEP 30.4 / 40.4 / 50.4 – Embedded-PC-basierte Bedienterminals	
Ausführungen.....	33
Einsatzbereich.....	32
Technische Daten.....	32
Verschleißteile.....	33
IndraControl VEP 40.5 / 50.5 / VEP 30.4 / 40.4 / 50.4 – Embedded-PC-basierte Bedienterminals	
Dokumentation.....	34
Zubehör.....	34
IndraControl VPB und VDP.....	34
IndraControl VPP.....	38
IndraControl VR21.....	23
IndraLogic Items.....	100
IndraLogic OPC.DA.....	101
Eigenschaften.....	101
Haftungsausschluss.....	113
Item Syntax.....	113
lwSCP Items.....	98
lwSCP Server.....	98

K

Kleinbedienfeld VCP	
Ausführungen.....	19
Dokumentation.....	21, 31
Kurzbeschreibung.....	17
Technische Daten.....	19
Zubehör.....	20
Kleinbedienfeld VR21xx	

Index

Kurzbeschreibung.....	23
Kommunikationsadresse ermitteln.....	102
Kommunikationsbeispiel über OPC UA	
Browsen.....	132
Einleitung.....	128
Subscribe und Write.....	132
UaExpert.....	128
Verbindungsaufbau.....	130
Voraussetzungen.....	128
Kompakt Industrie-PC VPB 40.3 mit Bedien- gerät VDP 08.3 / 16.3 / 40.3	
Kurzbeschreibung.....	34
Kompakt Industrie-PC VPB 40.3 mit Bedien- gerät VDP xx.3	
Ausführungen.....	35
Dokumentation.....	37
Kompakt Industrie-PC VPB 40.3 mit Bedien- gerät VDPxx.3	
Einsatzbereich.....	35
Technische Daten.....	35
Verschleißteile.....	35
Zubehör.....	37
Kompakt Industrie-PC VPP 16.3 / 40.3 / 60.3	
Einsatzbereich.....	38
Kurzbeschreibung.....	38
Technische Daten.....	38
Kompakt Industrie-PC VPP xx.3	
Ausführungen.....	39
Verschleißteile.....	39
Zubehör.....	40
Kompakt Industrie-PC VPPxx.3	
Dokumentation.....	40
Konfiguration des IndraLogic OPC.DA Server	
Ablageverzeichnis.....	110
Ablageverzeichnis Windows 7.....	111
Ablageverzeichnis Windows CE.....	111
Ablageverzeichnis Windows XP.....	110
Item Syntax.....	113
Konfiguration des IndraLogic OPC.DA Servers	101
Kommunikationsadresse ermitteln.....	102
Konfiguration Server.....	102
OPCServer.ini.....	105
Server Konfigurator.....	103
Konfigurieren einer OPC Serveranbindung mit WinStudio	113
Konfigurieren einer OPC-UA Client-Anbin- dung mit WinStudio	135

M

Modbus TCP.....	138
Einführung und Übersicht.....	138
ModbusTCP.....	169
Allgemein.....	169
Projektierung mit Pro-face GP-Pro EX 2.2... ..	174
Projektierung mit Siemens WinCC-flexi- ble 2007.....	171

O

OPC Client Implementierung.....	187
OPC Client Visual Basic .NET.....	183
Exception Handling mit Try...Catch...- Anweisungen.....	206
If-Abfragen.....	206
Visual Basic Projekt OPC Client.....	183
OPC Client Visual C# .NET.....	206
Asynchrones Lesen.....	222
Asynchrones schreiben.....	224
Daten auswerten.....	223, 225, 226
Deklaration der Variablen.....	211
Exception Handling mit try...catch...-An- weisungen.....	227
Fehlerbehandlung.....	227
If-Abfragen.....	227
IndraLogic OPC Server.....	219
Items hinzufügen - AddItems.....	218
Items löschen - RemoveItems.....	220
Neues Projekt.....	207
OPC Client Implementierung.....	210
OPC Items hinzufügen und löschen.....	218
OPC Server auflisten.....	212
OPC Server Instanz erstellen und löschen..	212
OPC Server Instanz erzeugen - Connect....	213
OPC Server Instanz löschen - Disconnect...	213
OPC Subscription erstellen und löschen....	215
Opc.Da.BrowseFilter.....	217
Opc.ItemIdentifier.....	217
OPC.lwSCP Server.....	219
Server Informationen lesen.....	214
Steuerungsdaten lesen.....	221
Steuerungsdaten schreiben.....	223
Steuerungsisems auflisten.....	217
Steuerungsisems lesen.....	217
Subscription erstellen - CreateSubscription..	215
Subscription Informationen anzeigen.....	217
Subscription löschen - CancelSubscription..	216
Synchrones Lesen.....	221
Synchrones schreiben.....	223
Verweise hinzufügen.....	207
Visual C# Projekt OPC Client.....	207
Windows Form und Steuerelement hinzu- fügen.....	208
Zyklisches Lesen aktivieren.....	226
Zyklisches Lesen deaktivieren.....	226
Zyklisches Lesen der Steuerungsdaten.....	226
OPC UA.....	117
Der Bosch Rexroth OPC UA Server.....	123
Einführung in OPC Unified Architecture.....	117
Kommunikationsbeispiel über OPC UA.....	128
Übersicht.....	117
OPC.DA Server.....	167
Konfiguration.....	167
Konfiguration auf ein Visualisierungsge- rät VEPxx.4 mit WindowsCE übertragen....	168
OPC.lwSCP.....	92

Eigenschaften.....	92
Haftungsausschluss.....	101
OPC.lwSCP Items.....	98
OPC.lwSCP Server.....	98
OPCServer.ini.....	105

P

PROFIBUS DP.....	20
------------------	----

S

Schnittstellen.....	28
Ethernet-Schnittstelle.....	29
X9, X10 USB-Schnittstellen.....	29
SCP-OPC Konfigurator.....	93
Allgemein.....	93
Anwendung beenden.....	98
Bedienung.....	93
Geräte entfernen.....	97
Geräte hinzufügen.....	94
Geräte OPC Alias ändern.....	96
Geräteeigenschaften anzeigen.....	95
Gerätekonfiguration aktivieren.....	97
Gerätekonfiguration erstellen.....	93
Gerätekonfiguration öffnen.....	94
Gerätekonfiguration speichern.....	97
Gerätekonfiguration speichern unter.....	97
SD-Speicherkarte.....	29
sercos Items.....	100
Sicherheitshinweise.....	9, 10
Signalgrafik.....	10
Signalwörter.....	10
Spannungsversorgung.....	28
Spannungsversorgung (DC 24 V).....	28
Speicherkarte.....	29
Statusanzeigen.....	29
Steckplatz Speicherkarte.....	29
Support	
siehe Service-Hotline.....	229
Symbole.....	10
Symbolkonfiguration	
Allgemein.....	141
Symbolkonfiguration mit Wizard.....	142

T

Technische Daten.....	26
Spannungsversorgung.....	28
Typenschlüssel.....	31

U

Übersicht.....	17
USB-Schnittstellen.....	29

V

Varianten des VR 21.....	25
VDP 16.3.....	34
VDP 40.3.....	34
VDP 60.3.....	34

VI-Composer

Dokumentation.....	46
Einsatzgebiet.....	44
Kommunikationsprotokoll BRC-Symbolic.....	44
Projektierbare Kleinbedienterminals.....	44
Verbindungseinstellungen.....	45

VI-Logic

Dokumentation.....	44
Eigenschaften der integrierten Visualisierung	41
Einsatzgebiet.....	41
Protokoll.....	43
Unterstützte Visualisierungsgeräte.....	43
Visualisierungselemente.....	42

Visual Basic Projekt OPC Client.....	183
Asynchrones Lesen.....	200
Asynchrones Schreiben.....	202
Daten auswerten.....	201, 203, 205
Deklaration von Variablen.....	188
Fehlerbehandlung.....	206
IndraLogic OPC Server.....	196
Items hinzufügen - AddItems.....	196
Items löschen - RemoveItems.....	198
Neues Projekt.....	183
OPC Client Implementierung.....	187
OPC Server auflisten.....	189
OPC Server Instanz erzeugen - Connect... ..	190
OPC Server Instanz löschen - Disconnect... ..	190
OPC Subscription erstellen - Create-Subscription.....	192
OPC Subscription Löschen - Cancel-Subscription.....	193
Opc.Da.BrowseFilter.....	195
Opc.ItemIdentifier.....	195
OPC.lwSCP Server.....	196
Server Informationen lesen.....	191
Steuerungsdaten lesen.....	199
Steuerungsdaten schreiben.....	202
Steuerungsisitems auflisten.....	194
Steuerungsisitems lesen.....	194
Subscription Informationen anzeigen.....	194
Synchrones Lesen.....	199
Synchrones Schreiben.....	202
Verweise hinzufügen.....	184
Windows Form und Steuerelement hinzufügen.....	185
Zyklisches Lesen aktivieren.....	204
Zyklisches Lesen deaktivieren.....	204
Zyklisches Lesen der Steuerungsdaten.....	204
VPB 40.....	34

W

Windows Form und Steuerelement hinzufügen	185
WinStudio.....	46
Aktivierung einer WinStudio-Lizenz.....	77
Alarmer.....	67
Allgemeine WinStudio Einstellungen und Funktionen.....	47

Index

Anwender-Skripte.....	71	Z	
Anwendersymbole.....	65	Zu dieser Dokumentation.....	9
Anwendung starten.....	72	Gültigkeit der Dokumentation.....	9
Anwendung verifizieren.....	72		
Auswirkungen einer fehlenden oder zu geringen Lizenz.....	78		
Benutzerverwaltung.....	79		
Bild-Skripte.....	71		
Bildeigenschaften.....	62		
Datenbank-Spion.....	73		
Debugger.....	74		
Erstellen und Bearbeiten von WinStudio-Bildern.....	61		
Erstellung einer WinStudio-Applikation.....	78		
Export/Import.....	76		
Externe Datenbanken.....	69		
Grafik-Skript.....	70		
LogWin.....	74		
Mathematikblätter.....	67		
Mehrsprachigkeit.....	78		
Objekteigenschaften und Variablenauswahl..	63		
Querverweissuche.....	75		
Reporte.....	68		
Rezepte.....	67		
Schedules.....	69		
Skripte.....	70		
Start-Skript.....	70		
Trends.....	69		
Übersicht.....	46, 78		
Übersicht der WinStudio-Lizenzen.....	76		
Übertragen der Applikation und Aktivierung auf dem Zielgerät.....	79		
Variablen-Importassistent.....	75		
Verlinkte Symbole.....	64		
Web-Funktionalität.....	79		
Weiterführende Informationen.....	81		
WinStudio-Applikationen.....	78		
WinStudio-Bibliothek.....	64		
WinStudio-Bilder.....	61		
WinStudio-Elemente im Projektexplorer.....	47		
WinStudio-Lizenzierung.....	76		
WinStudio-Tasks.....	67		
WinStudio-Variablen.....	63		
WinStudio-Werkzeuge.....	72		
WinStudio mit IndraLogic			
Einbinden der WinStudio-Visualisierung in IndraWorks.....	151		
Erstellen der WinStudio-Visualisierung.....	158		
IndraWorks-Projekt erstellen.....	145		
Starten der Oberfläche.....	165		
Technische Voraussetzungen.....	145		
X			
X5.....	29		
X9.....	29		
X10.....	29		

Notizen

Bosch Rexroth AG
Electric Drives and Controls
Postfach 13 57
97803 Lohr, Deutschland
Bgm.-Dr.-Nebel-Str. 2
97816 Lohr, Deutschland
Tel. +49 9352 18 0
Fax +49 9352 18 8400
www.boschrexroth.com/electrics



R911336361

DOK-XLCMLC-HMI*****V13-AP02-DE-P