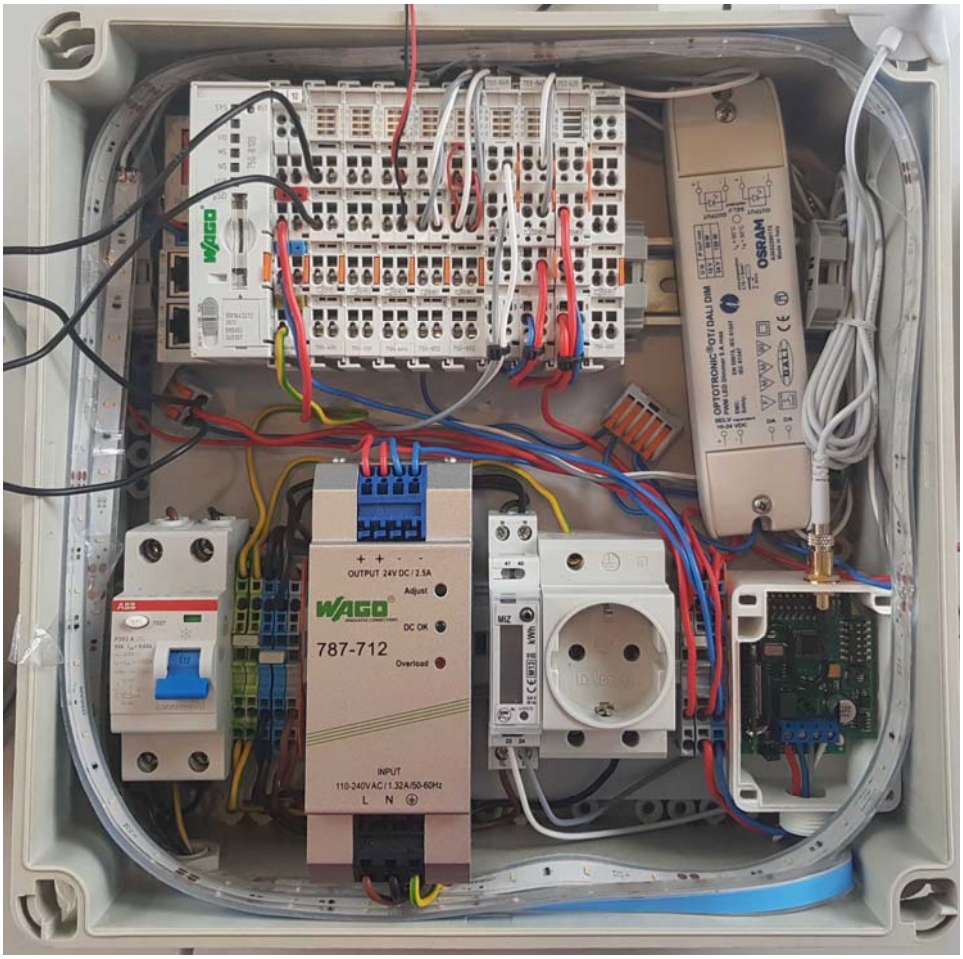
Hochschule **Rosenheim**
University of Applied Sciences

WAGO PFC100 mit e!Cockpit



Tutorial für das Praktikum „Gebäudeautomation“

Version 01c, 01. Juli 2021

Hochschule Rosenheim · Hochschulstrasse 1 · 83024 Rosenheim
www.fh-rosenhheim.de · *michael.kroedel@fh-rosenheim.de*

WAGO PFC100-Tutorial

Inhaltsverzeichnis

Dieses Tutorial wurde am Institut für Gebäudetechnologie unter Leitung von
Herrn Prof. Dr. Michael Krödel erstellt.

Mitgewirkt haben:

- Hr. Alexander Knebel (Ersterstellung 2018)
- Hr. Thomas Bogner (Tests und kleinere Updates 2021)

WAGO PFC100-Tutorial

1 Inhalt des Dokuments

Das Ziel dieses Dokuments ist die Beschreibung der Inbetriebnahme eines PFC100-Controllers der Firma WAGO Kontakttechnik GmbH & Co. KG mit Hilfe der Programmiersoftware „e!Cockpit“. Hierbei wird neben einer einleitenden Vorstellung der Software e!Cockpit, die Einrichtung des Controllers, die Einbindung von Klemmen und die grundlegende Vorgehensweise von der Erstellung bis zum Start des fertigen Programmcode beschrieben.

Das Dokument beschreibt weiter in konkreten Anwendungsbeispielen, die Integration von binären Signalen, die Auswertung von PT1000-Sensoren, die Verwendung von EnOcean-Geräten, die Einbindung einer Wetterstation über RS485, die Nutzung des M-Bus und des DALI-Bus, sowie die Erstellung einer webbasierten Visualisierung. Abschließend wird eine Anbindung des PFC100 mit der WAGO Cloud Data Control gezeigt.

2 Beschreibung der Praktikumsanordnung

Für die im Tutorial verwendeten Anwendungsbeispiele wird der in Abbildung 1 gezeigte Aufbau (Montage aller Komponenten innerhalb eines kompakten Kleinverteilers) eingesetzt.

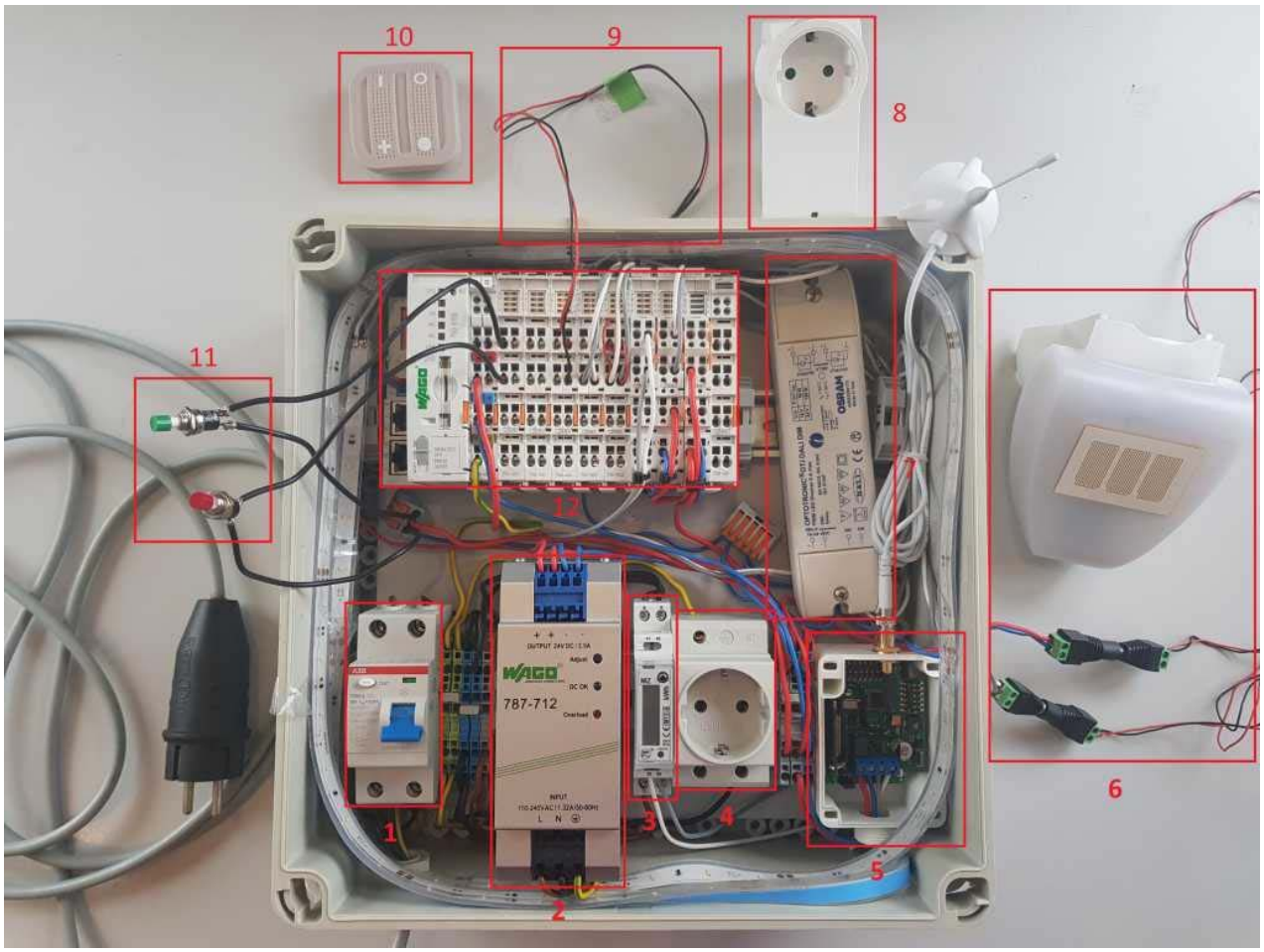


Abbildung 1: Controllerbox

1	RCD (FI-Schutzschalter)	7	DALI-LED-Treiber
2	24V Spannungsversorgung	8	EnOcean-SmartPlug
3	M-Bus-Zähler	9	PT 1000 Temperaturfühler
4	Steckdose	10	EnOcean-Taster
5	Thermokon-Gateway mit Antenne	11	Taster
6	Wetterstation inkl. Anschlüsse	12	PFC100-Controller mit Klemmen

Tabelle 1: Aufbau der Controllerbox

Position 1	750-8100: PFC100	Position 6	750-652: RS485 serielle Klemme
Position 2	750-602: Potentialeinspeisung	Position 7	750-652: RS485 serielle Klemme
Position 3	750-430: Digitale Eingangsklemme	Position 8	753-649: M-Bus Masterklemme
Position 4	750-530: Digitale Ausgangsklemme	Position 9	753-647: DALI Masterklemme
Position 5	750-464: RTD Analoge Eingangsklemme	Position 10	753-620: DALI DC-DC Konverter

Tabelle 2: PFC100-Controller mit eingesetzten Klemmen von links nach rechts

3 Inbetriebnahme des Controllers unter e!Cockpit

3.1 Allgemeine Einführung in die PFC-Steuerungen von WAGO sowie e!Cockpit

Nach dem Start der Anwendung e!Cockpit erscheint ein Auswahlfenster. Hier können unter anderem Bibliotheken verwaltet, die Klemmen der 750er-Produktserie eingesehen und neue Projekte angelegt werden.

Beim Erstellen eines neuen Projekts erscheint der nachfolgende Bildschirm, dessen Aufbau in Abbildung 2 beschrieben ist.

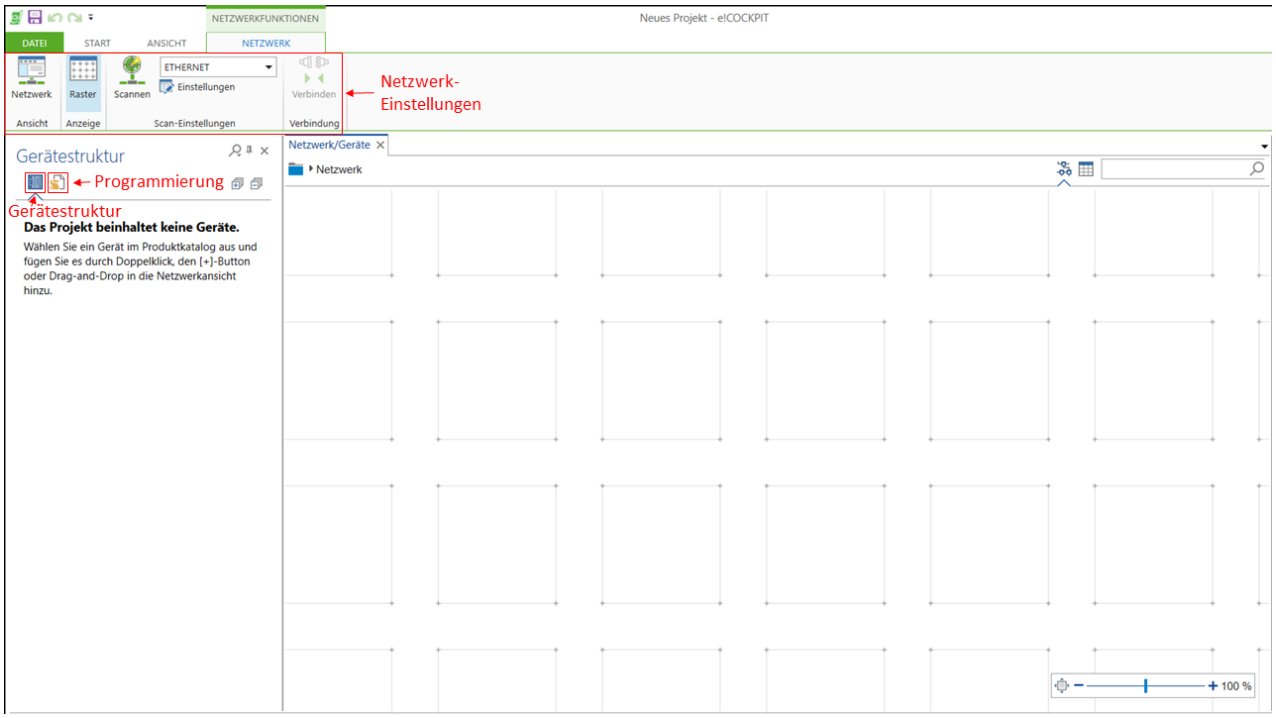


Abbildung 2: Übersicht e!Cockpit

Über die Buttons „Gerätestruktur“ und „Programmierung“ kann zwischen den Hardware-Einstellungen, wie Netzwerk- und Klemmenkonfigurationen und zwischen den Softwareprogrammen der Steuerung gewählt werden.

Zum grundsätzlichen Setup gehört das Konfigurieren des Knotens inkl. Netzwerkkonfiguration. Diese kann über zwei unterschiedliche Arten durchgeführt werden:

- USB-Service-Kabel (empfohlen!)
- BootP

3.1.1 IP-Adresse über USB Service Kabel

Die Adresszuweisung über das USB Servicekabel ist die Standardvariante. Der Controller ist über das USB-Service-Kabel mit einem PC zu verbinden (am Controller auf die Programmierschnittstelle unter der Klappe stecken).

Das Aufstecken oder Abziehen des Servicekabels muss im spannungsfreien Zustand des Controllers durchgeführt werden!

Hinweis: Der Betriebsartenschalter am Controller (unter der kleinen Plastiklappe) muss auf RUN, d.h. der obersten Position stehen.

Anschließend kann in e!Cockpit in den Netzwerkeinstellungen der COM-Port des Servicekabels gewählt und der Scan gestartet werden.

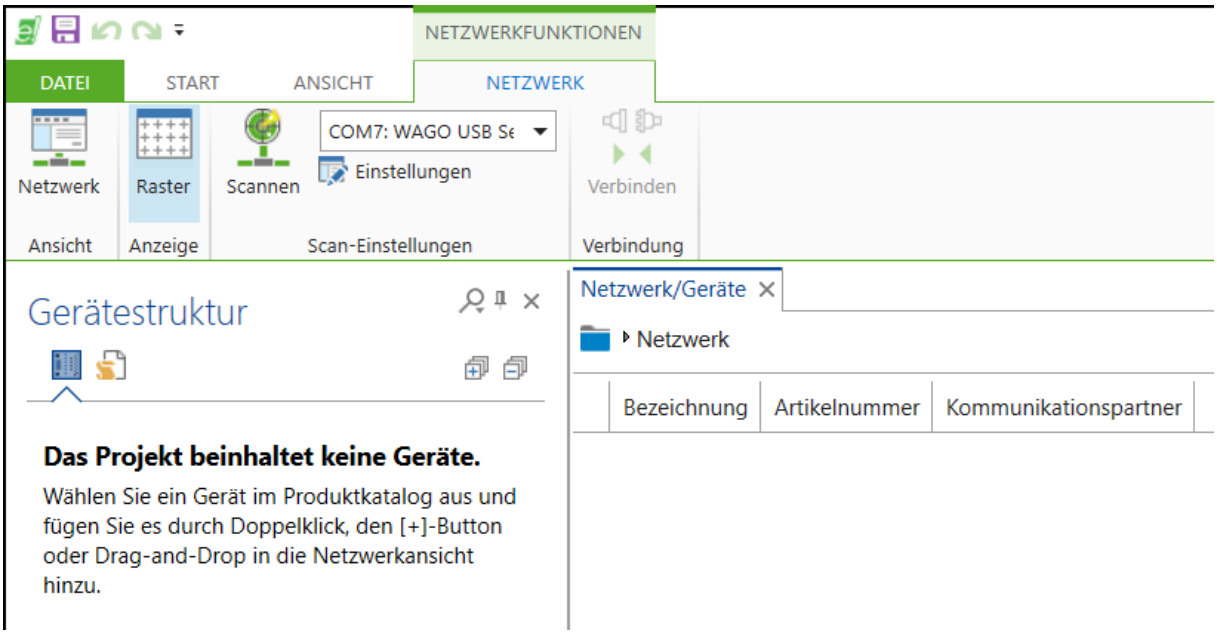


Abbildung 3: Hinzufügen neuer Controller

Nach dem Scan erscheint der angeschlossene Controller im Scan-Ergebnis.

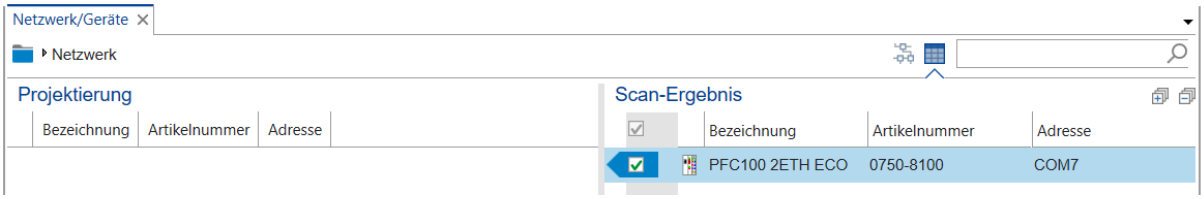


Abbildung 4: Erfolgreiches Scannen

Dieser wird über den Button im linken oberen Bildschirm „Auswahl übernehmen“ bestätigt.

Sofern die auf dem Controller aktuell installierte Version nicht mit der aktuellen in der Softwareversion des e!Cockpit übereinstimmt erscheint ein Fenster, welches auf eine veraltete Firmware des Controllers hinweist (siehe auch Kapitel 3.2). Über den Button „hinzufügen“ kann der Controller in die Gerätestruktur hinzugefügt werden.

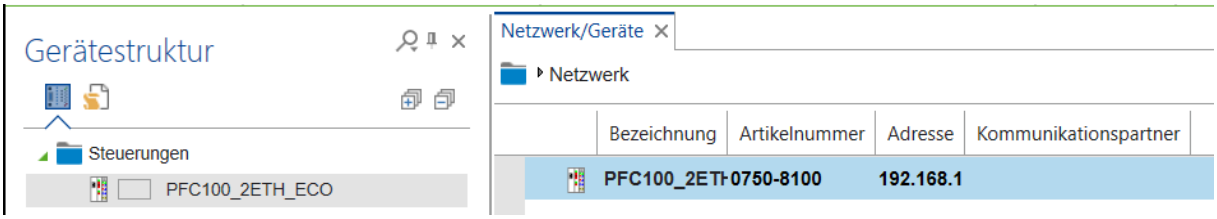


Abbildung 5: Gerätestruktur

Im nächsten Schritt werden die Netzwerkeinstellungen des Controllers an das Netzwerk angepasst, damit eine Programmierung via Ethernet erfolgen kann.

Um dies zu realisieren, wird der Controller ausgewählt und über Auswahl des Buttons „Einstellungen“ auf der rechten Seite, das Programm „WAGO Ethernet Settings“ automatisch aufgerufen und der Controller eingelesen. Über den Reiter „Netzwerk“ kann dem Controller die statische IP-Adresse **192.168.178.210** zugewiesen und anschließend über den Button „Schreiben“ in den Speicher des PFC100 geschrieben werden.

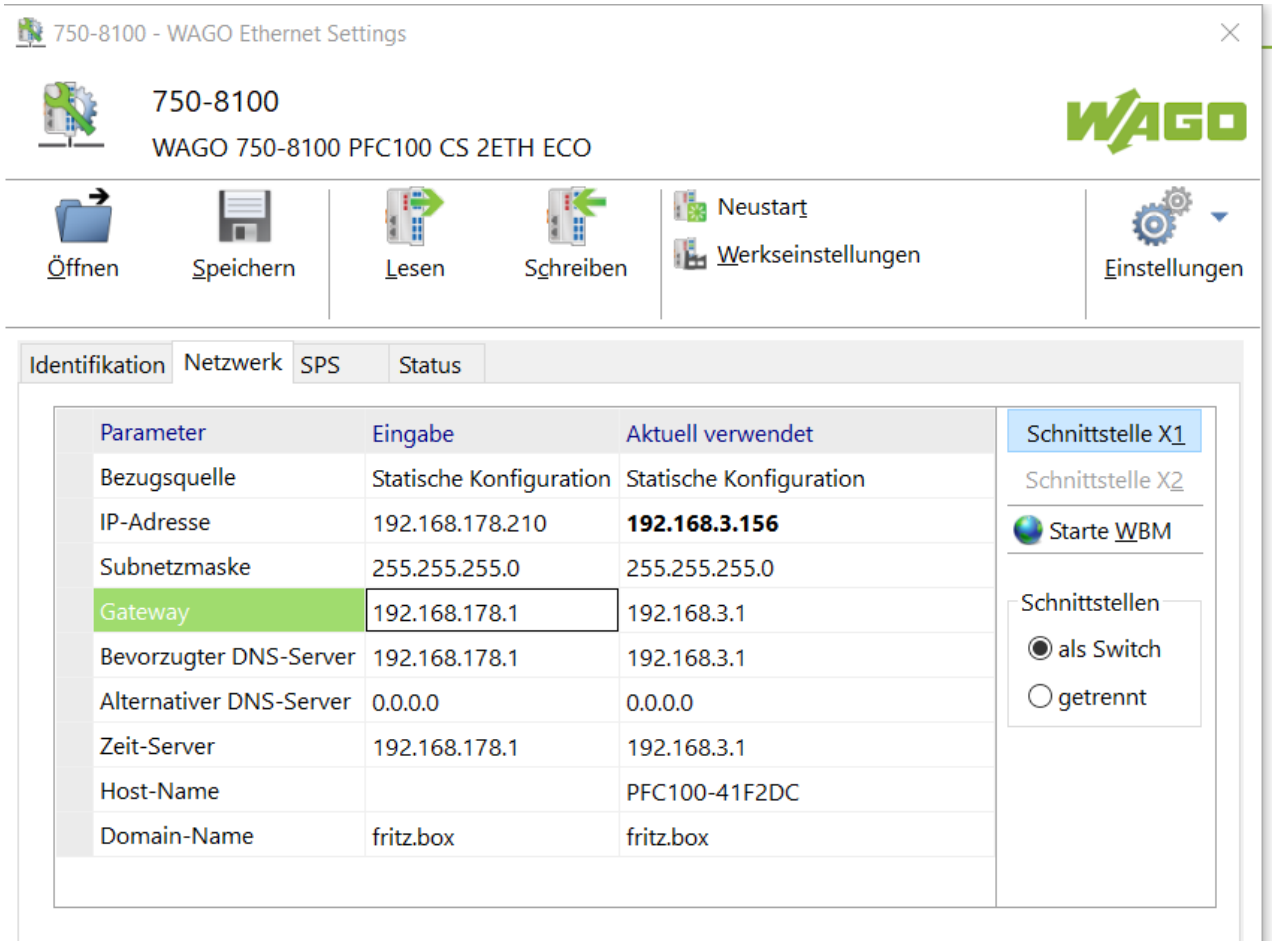


Abbildung 6: Ethernet Settings anpassen

Die Kommunikation zum Controller kann anschließend getestet werden über:

- Start eines normalen Browsers und Eingabe der IP-Adresse in die URL-Zeile
  → Die Webseite des Controllers sollte erscheinen.

Die Konfiguration über USB-Service Kabel ist damit abgeschlossen.

3.1.2 IP-Adresse über DHCP

Sofern der Controller die Möglichkeit der IP-Zuweisung über DHCP freigeschaltet hat (werksseitig bereits voreingestellt!), kann dieser nach Anschluss an das LAN gescannt werden.

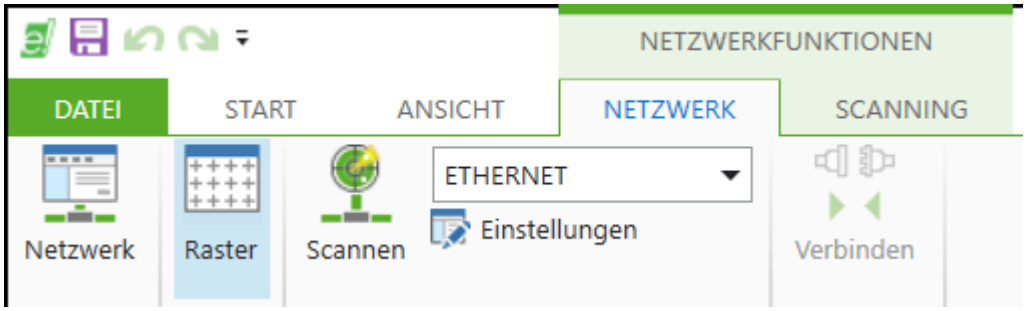


Abbildung 7: Ethernet-Scan

Im Vorfeld zum Scannen sollten dabei die Einstellungen der Scan-Parameter angepasst werden

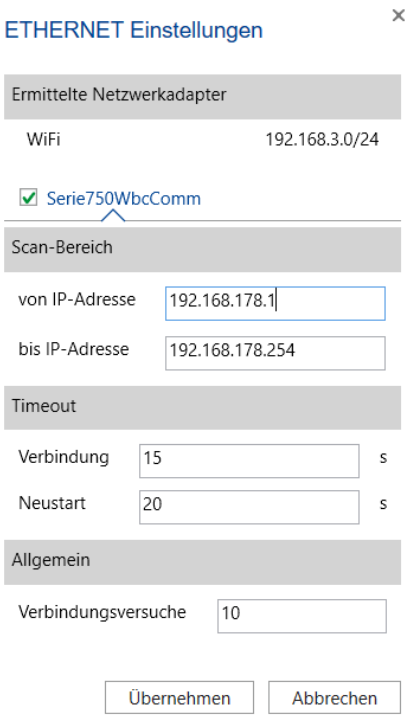


Abbildung 8: Scan-Parameter

Nach einem erfolgreichen Scan kann der Controller ausgewählt werden. Durch den Button „Auswahl übernehmen“ wird dieser übernommen.

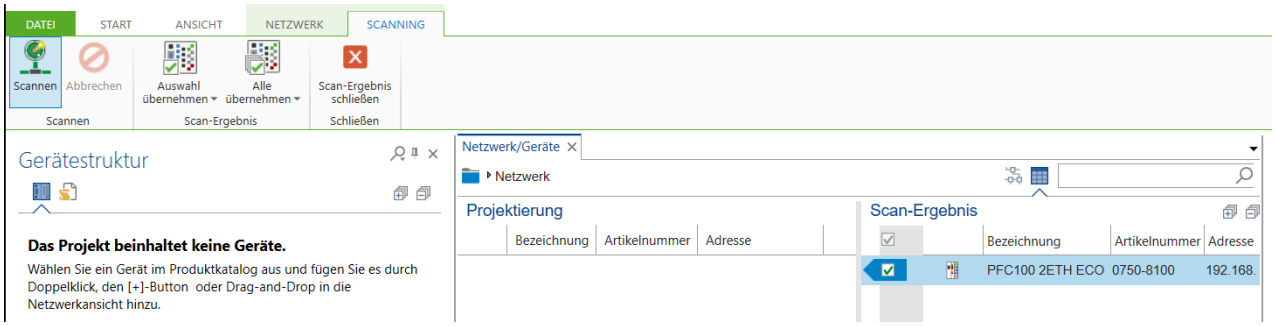


Abbildung 9: Controller übernehmen

Nach dem Übernehmen öffnet sich ein Fenster, in welchem der Button „hinzufügen“ angeklickt wird. Nun wird als nächstes der Login abgefragt. Dieser geschieht über durch Eingabe des Nutzernamens **„admin“** und dem Passwort **„wago“** (sofern das Passwort noch nicht geändert wurde). Die Steuerung befindet sich nun in der Gerätestruktur.

3.2 Update der Firmware

Sofern sich auf dem Controller eine veraltete Firmware befindet, sollte diese upgedatet werden, damit die aktuellsten in e!Cockpit verwendeten Klemmen und Bibliotheken auch korrekt verwendet werden können.

Das Update muss manuell erfolgen und erfordert dafür eine Firmware-Datei, welche von WAGO anzufordern und über eine SD-Karte und das WBM des Controllers installiert werden muss.

Nähere Informationen und das Prozedere des Updates sind in einem VideoTutorial der Fa. WAGO gezeigt: https://www.youtube.com/watch?v=Ks1ak1zYVUg

3.3 Einlesen der Klemmenkonfiguration:

Nach dem Login ist die Steuerung als hellgraue Kachel bei Netwerk/Geräte zu sehen. Durch Rechtsklick auf diese und Auswahl des Buttons „Scannen“ öffnet sich ein neues Fenster, in dem die Klemmeneinstellungen ausgelesen werden können.

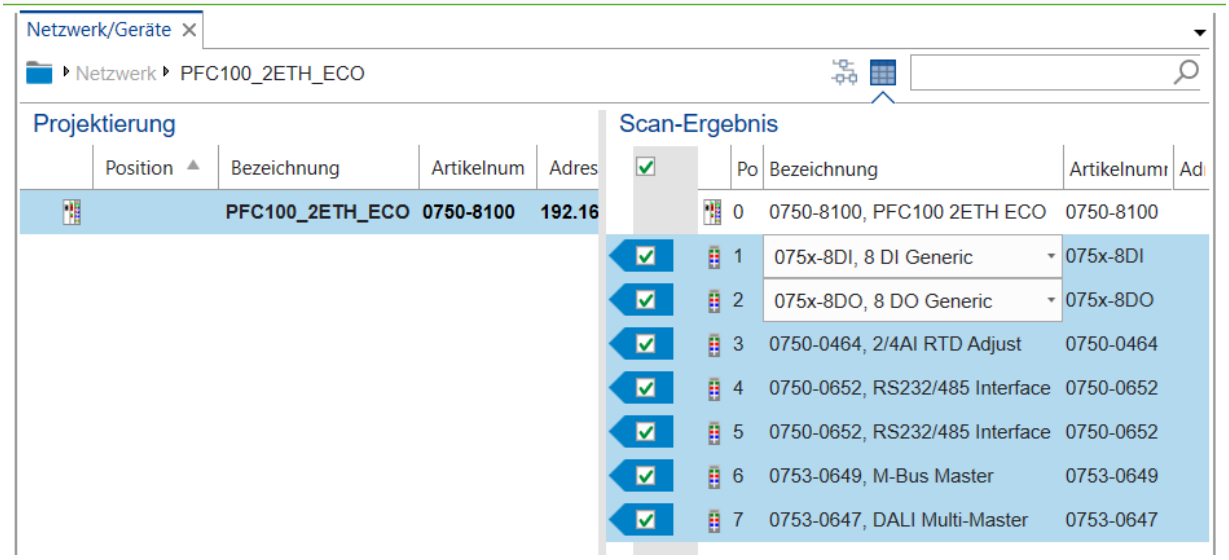


Abbildung 10: Scan der angeschlossenen Klemmen

Es werden nun die Klemmen des Scan-Ergebnisses auswählt und erneut über den Button „Auswahl hinzufügen“ in die Gerätestruktur geschrieben. Über den neuen Reiter „Gerät“ kann über den Button „Verbinden“ eine Verbindung mit der Steuerung hergestellt werden. Da der Lokalbus noch nicht aktiv ist, kann dieser über einen Rechtsklick auf die Steuerung und „Start“ gestartet werden.

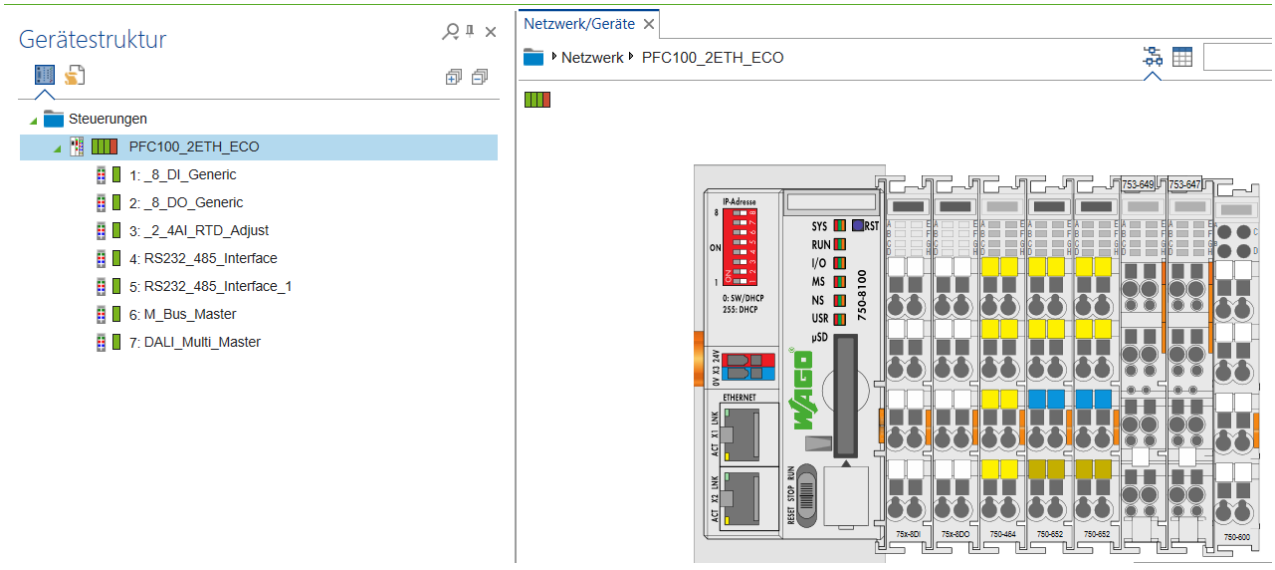


Abbildung 11: Klemmenkonfiguration im aktiven Zustand

Die Klemmenkonfiguration ist damit beendet. Durch Doppelklick auf eine Klemme wird die Steuerung angezeigt und die Klemmen können eingesehen werden. Dadurch können Datenströme oder Zustände der Klemmen angezeigt oder im rechten Bereich die Einstellungen der Klemme konfiguriert werden.

Zusätzliche Komponenten:

Der PFC-Controller vom Typ 750-8100 verfügt nicht über eine integrierte Potentialeinspeiseklemme. Die am Controller angeschlossene Spannung versorgt ausschließlich den Controllerkopf - nicht jedoch weitere am Controllerkopf angeschlossene Klemme. Daher muss eine zusätzliche Potentialeinspeiseklemme eingesetzt werden.

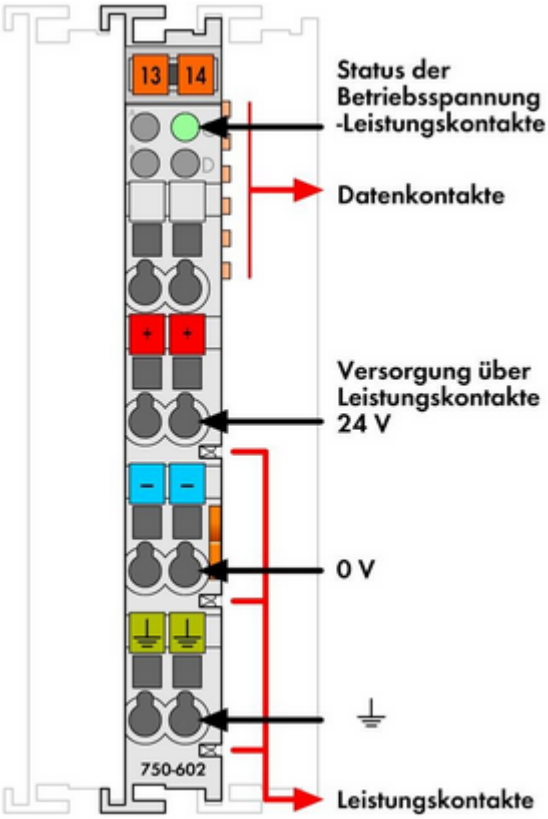


Abbildung 12: Potentialeinspeiseklemme (750-602)

Wichtig: Um die Funktion des Feldbuses (Funktionalitäten der Klemmen) zu aktivieren MUSS eine Potentialeinspeiseklemme (z.B.: 750-602) direkt vor die Klemmen und hinter den PFC-Controller auf die Hutschiene eingebunden werden. Die Spannungsversorgung für diese Klemmen ist anschließend an den Port „+“ und„-“ anzuschließen.

4 Einfaches Beispielprogramm

Anhand eines Demo-Programms soll nachfolgend gezeigt werden, wie Klemmen im Programmcode angesprochen werden, wie Programme erstellt, Programme simuliert und Bibliotheken eingebunden werden. Das Ziel dieses Beispiels liegt darin, über virtuelle Variablen den Ausgang der DO-Klemme zwischen TRUE und FALSE umzuschalten.

4.1 Einstellung der Variablendeklaration der Klemme

Nach erfolgreichem Auslesen der Klemmenkonfigurationen können den IO-Adressen Variablen zugeordnet werden. Dies sind „globale Variablen“ und dienen dem Zweck, sie einfacher im Programmcode aufzurufen zu können.

Netzwerk/Geräte

Netzwerk ▸ PFC100_2ETH_ECO

Lokalbus E/A-Abbild

Suchen | Filter Alle anzeigen

Variable	Mapping	Kanal	Adresse	Typ	Standardwert	Einheit	Beschreibung
		_OUT	%QB24	BYTE			Ausgangskanäle
DO_1		_OUT	%QX24.0	BOOL	FALSE		digitaler Ausgang
DO_2		_OUT	%QX24.1	BOOL	FALSE		digitaler Ausgang
DO_3		_OUT	%QX24.2	BOOL	FALSE		digitaler Ausgang
DO_4		_OUT	%QX24.3	BOOL	FALSE		digitaler Ausgang
DO_5		_OUT	%QX24.4	BOOL	FALSE		digitaler Ausgang
DO_6		_OUT	%QX24.5	BOOL	FALSE		digitaler Ausgang
DO_7		_OUT	%QX24.6	BOOL	FALSE		digitaler Ausgang
DO_8		_OUT	%QX24.7	BOOL	FALSE		digitaler Ausgang

Abbildung 13: Vergabe von Variablen einer DO-Klemme

4.2 Erstellung eines Programms:

Im Anschluss an die Benennung der IO-Adressen der Klemme wird über die Auswahl des Buttons „Programmierung“ in ein neues Fenster gewechselt. Hier befinden sich nun die Steuerung und Einstellungen des Projekts, wie der Bibliotheksverwalter, die Taskkonfiguration und auch Programme wie die PLC_PRG.

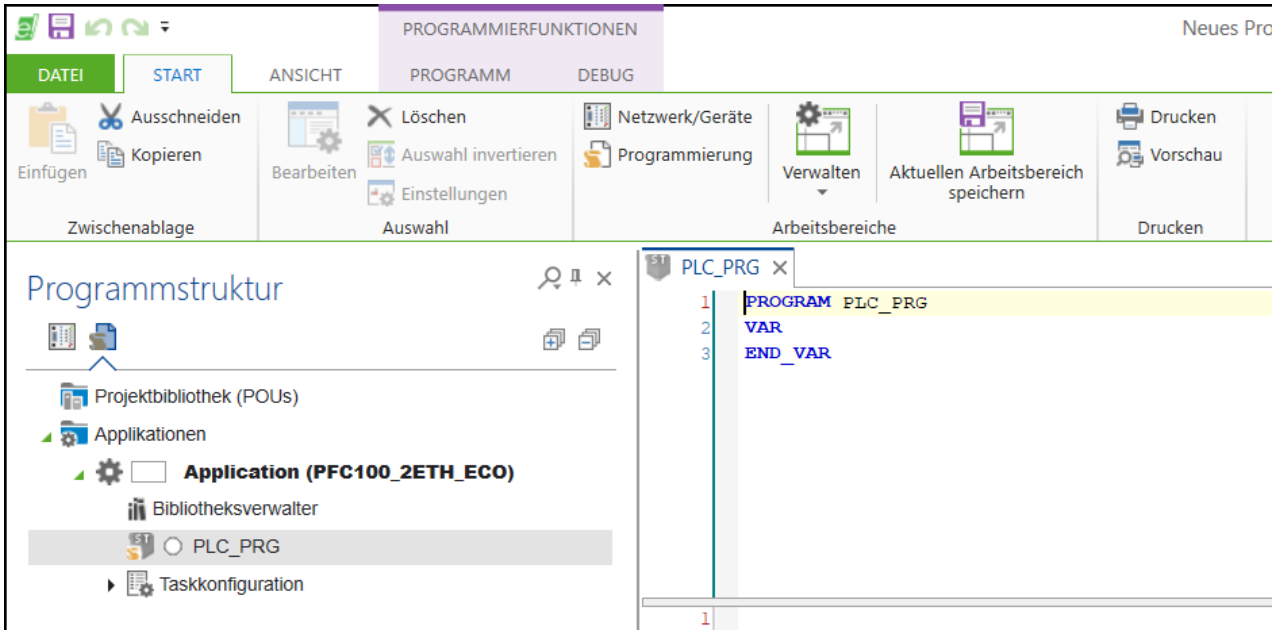


Abbildung 14: Programmierung der Steuerung

Ein neues Programm wird über Rechtsklick auf „Application (PFC100_2ETH_ECO)“ und Einfügen eines neuen „POU“ hinzugefügt. Hierzu wird das neue Programm mit dem Namen „Tutorial_Beispiel_1“ in der Implementierungssprache „Funktionsbausteinsprache“ (FUP) erstellt.

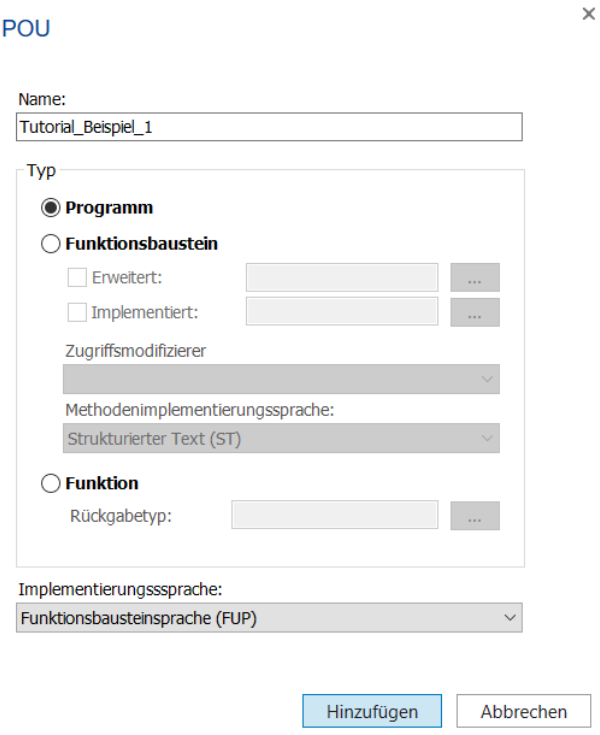


Abbildung 15: Erstellung eines Programms

4.3 Bibliotheksverwaltung

Im neu erstellten Programm kann nun mit der Programmierung begonnen werden. Zuvor sollte aber die Bibliotheksverwaltung geprüft werden. In der Bibliotheksverwaltung werden alle für die Applikation benötigten Bibliotheken dargestellt. Durch Doppelklick auf „Bibliotheksverwalter“ erscheint ein neues Fenster, in welchem der Nutzer die aktiven Bibliotheken einsehen, neue Bibliotheken hinzufügen oder löschen kann. Selbst Codesys 2.3-Bibliotheken können hier ausgewählt, konvertiert und in der Applikation eingesetzt werden.

Für das Umschalten der Variable der DO-Klemme soll ein RS-Flipflop verwendet werden. Dieser Funktionsblock kann, wie in Abbildung 16 gezeigt, in der Standard-Bibliothek gefunden werden.

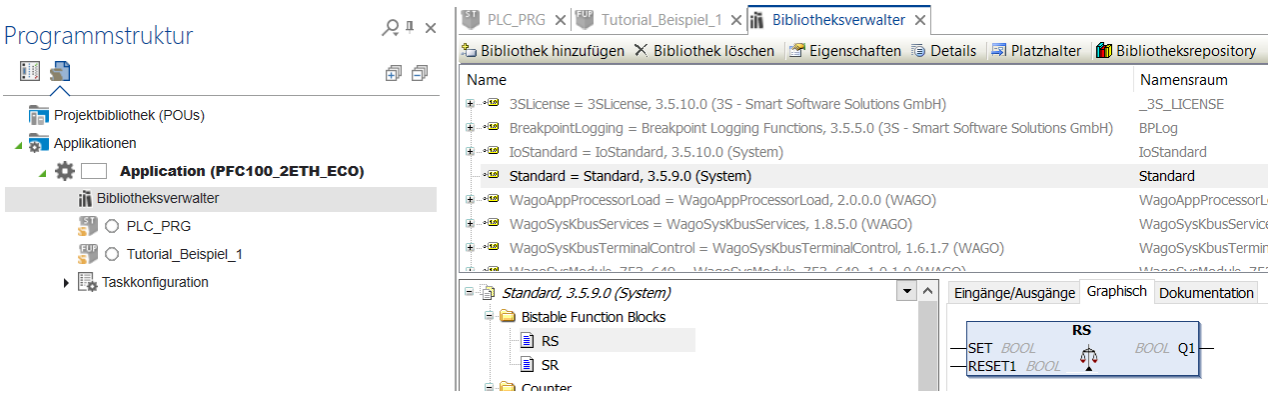


Abbildung 16: Betrachtung der Standard-Library im Bibliotheksverwalter

4.4 Programmierung des Beispiels

Nachdem geprüft wurde, ob die benötigte Bibliothek aktiv ist, wird im Programm „Tutorial_Beispiel_1“ ein Funktionsbaustein hinzugefügt und als RS-Glied deklariert. Außerdem werden die Variablen zum Setzen und Zurücksetzen eingefügt und die Ausgangsvariable dem ersten Port der DO-Klemme „DO_1“ zugewiesen.

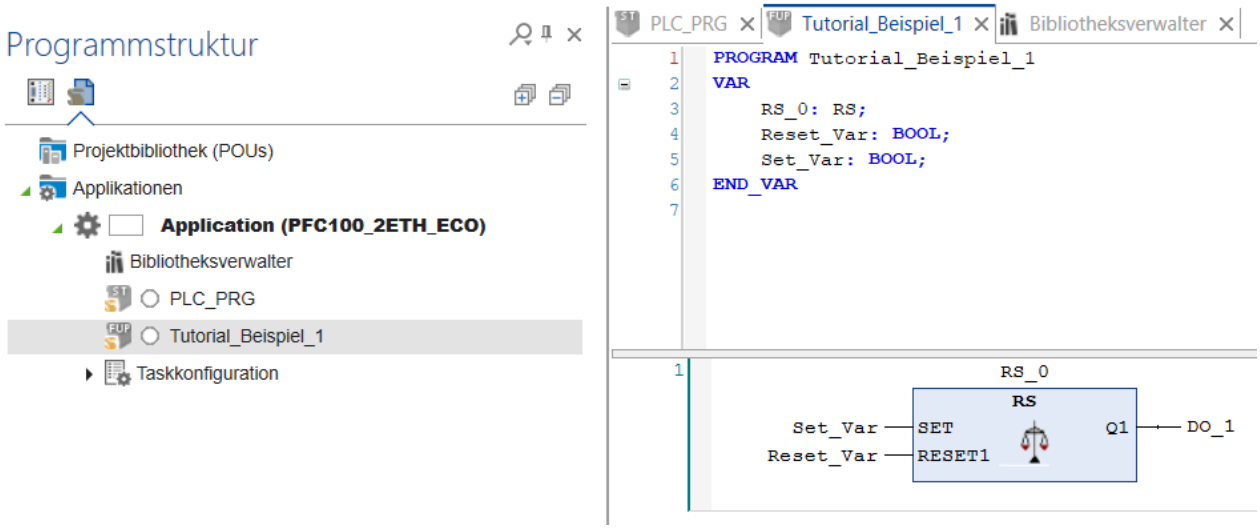


Abbildung 17: Fertiges Programm zu Beispiel 1

Anschließend muss im Programm PLC_PRG noch der Programmaufruf von „Tutorial_Beispiel_1“ eingetragen werden.

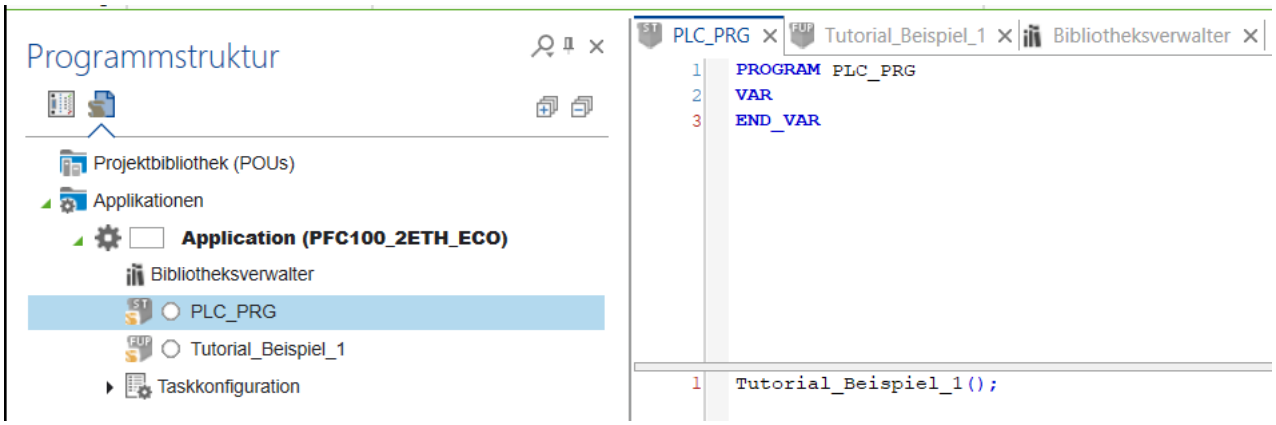


Abbildung 18: Programmaufruf Beispiel 1 in PLC_PRG

4.5 Programm testweise übersetzen

Um mögliche Fehler im Programm zu finden, kann man die programmierten Zeilen einmal „Übersetzen“ lassen. Dabei werden Syntaxfehler mit einem roten Kringel unter dem Programm angezeigt und die auftretenden Fehler werden in der Meldungsansicht angezeigt.

„Übersetzen“ lässt Sich das Programm im Reiter „Programm“ unter „Quellcode“.

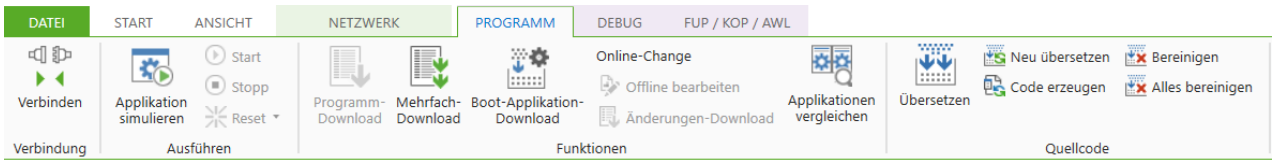


Abbildung 19: Programm testweise übersetzen

4.6 Programm starten und Variable setzen

Sofern keine Syntaxfehler angezeigt werden, kann die Applikation auf den Controller geladen werden. Hierzu wird, wie in Abbildung 20 gezeigt, der Button „Verbinden“ ausgewählt.

Sofern ein Offline-Test simuliert werden soll, kann dieser ausgewählt werden, das Programm wird anschließend nicht auf die Steuerung geladen, sondern in e!Cockpit simuliert.

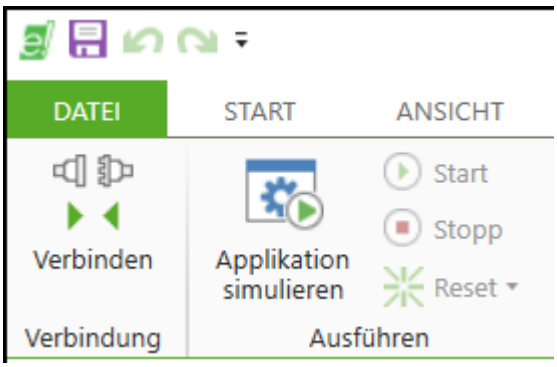


Abbildung 20: Verbindungs- und Steueroptionen des Controllers

Nach der Betätigung des Buttons „Verbinden“ wird von e!Cockpit nach den Logindaten gefragt.

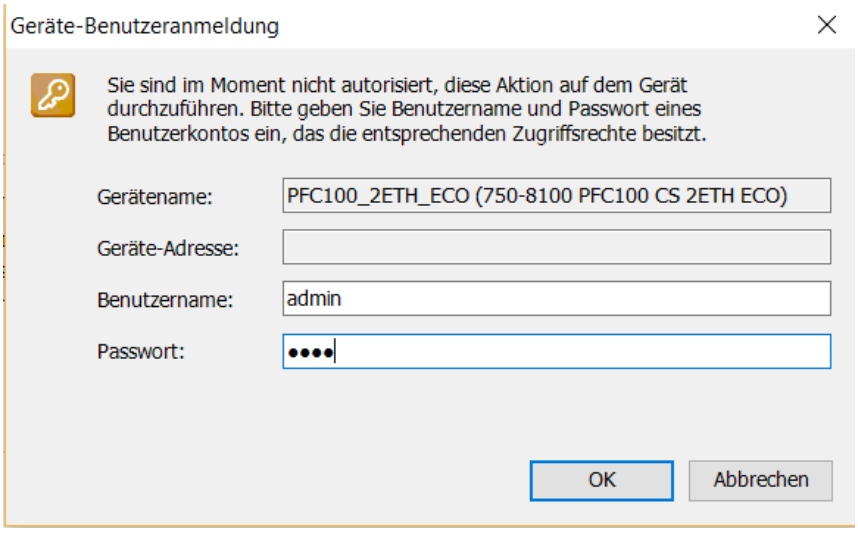


Abbildung 21: Login-Fenster der Applikation

Nachdem der Benutzername „admin“ mit Passwort „wago“ eingegeben wurde (sofern das Passwort nicht geändert wurde), erscheint ein Fenster, in welchem danach gefragt wird, ob die Applikation auf den Controller geladen werden soll. Nach der Bestätigung dieses Fensters beginnt der Upload des Programms auf den Controller. Dieser befindet sich noch im angehaltenen Zustand und muss erst über den Button „Start“ aktiviert werden.

Im gestarteten Zustand wird nun im Programm „Tutorial_Beispiel_1“ die Set-Variable des RS-Glieds mit Doppelklick angewählt und mit F7 geschrieben.

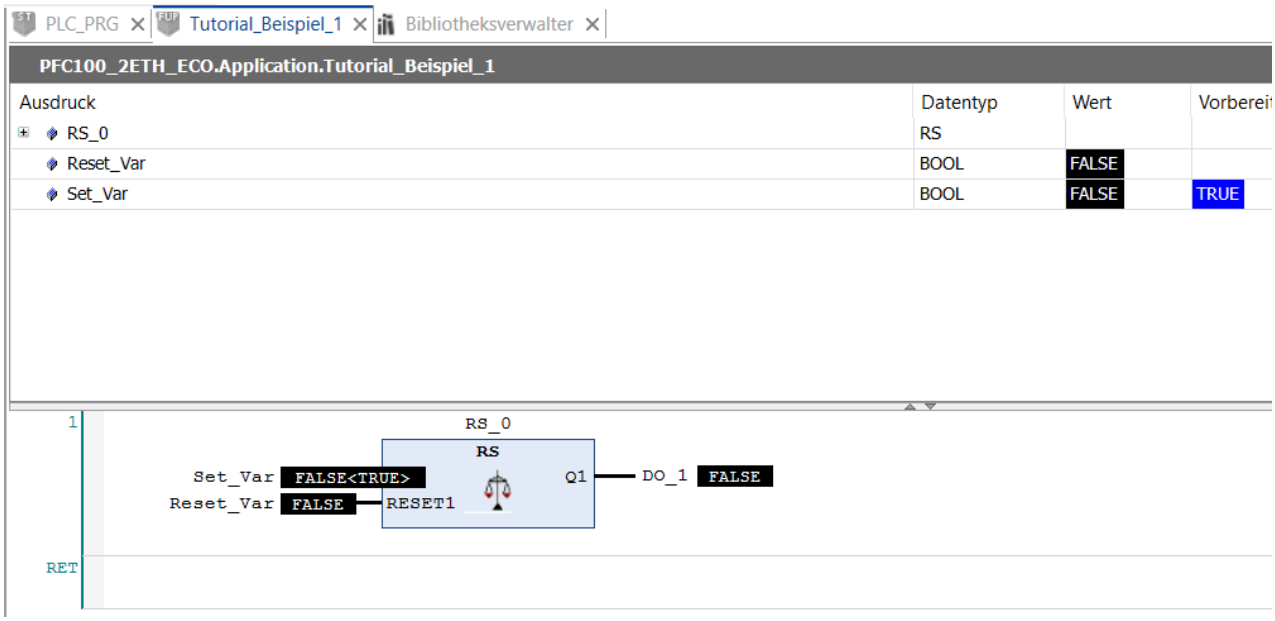


Abbildung 22: Variablen setzen

Der Zustand des DO_1 ändert sich nun auf TRUE.

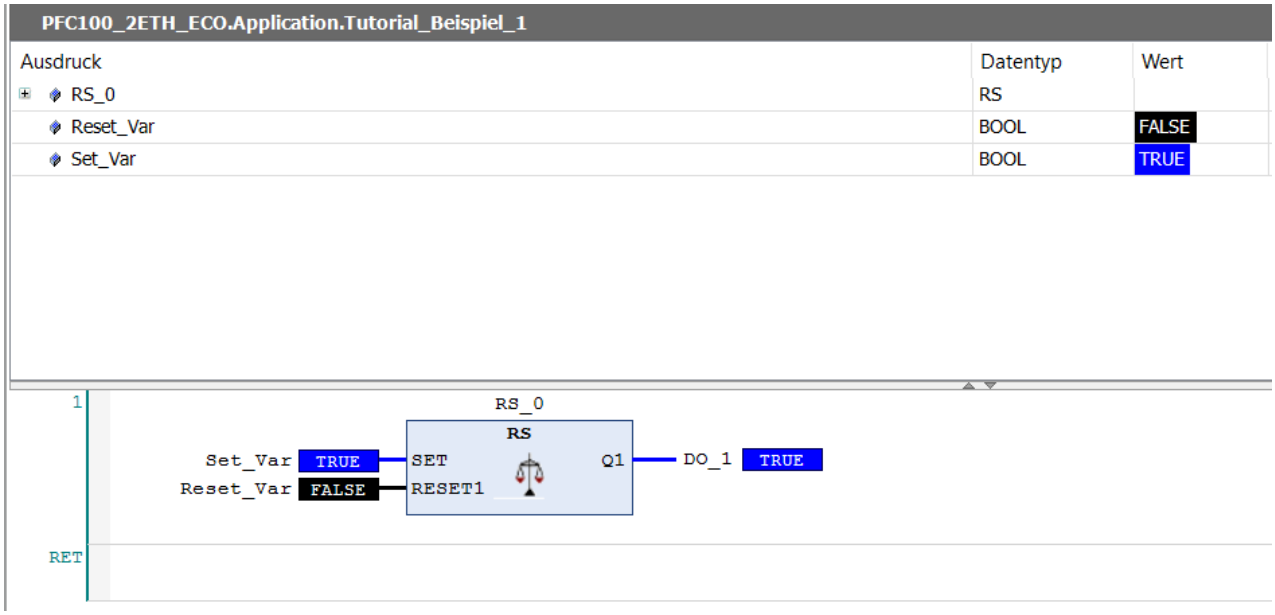
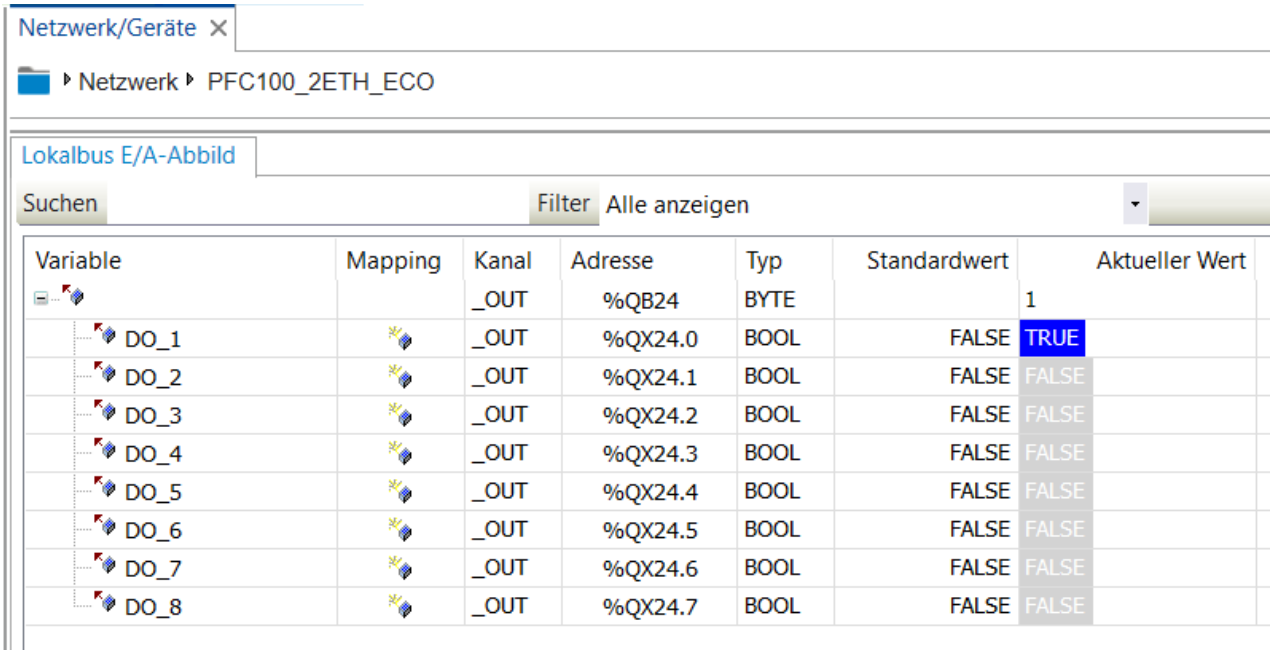


Abbildung 23: Gesetzter Output des RS-Glieds

Im Bereich Netzwerk/Geräte kann man nun erkennen, dass der erste Port der Klemme TRUE gesetzt ist und auch die LED des ersten Ports der Klemme sollte aktiv sein.



Variable	Mapping	Kanal	Adresse	Typ	Standardwert	Aktueller Wert
		_OUT	%QB24	BYTE		1
DO_1		_OUT	%QX24.0	BOOL	FALSE	TRUE
DO_2		_OUT	%QX24.1	BOOL	FALSE	FALSE
DO_3		_OUT	%QX24.2	BOOL	FALSE	FALSE
DO_4		_OUT	%QX24.3	BOOL	FALSE	FALSE
DO_5		_OUT	%QX24.4	BOOL	FALSE	FALSE
DO_6		_OUT	%QX24.5	BOOL	FALSE	FALSE
DO_7		_OUT	%QX24.6	BOOL	FALSE	FALSE
DO_8		_OUT	%QX24.7	BOOL	FALSE	FALSE

Abbildung 24: Zustand der DO-Klemme

Durch Setzen der Reset-Variable kann anschließend der Output wieder FALSE geschrieben werden.

5 Einbindung weiterer Klemmen im Projekt

In diesem Kapitel soll die Einbindung des in Abbildung 1 gezeigten kompletten Testaufbaus in e!Cockpit betrachtet werden.

5.1 Einbindung der RTD-Messklemme

Die Verwendung einer widerstandsbasierten Temperaturmessung zeigt deutliche Vorteile beim Einsatz in Applikationen, in denen die Messwerte schnell, hochauflösend und regelmäßig zur Verfügung gestellt werden müssen. Vor allem durch den Einsatz in sehr hohen und bei sehr niedrigen Umgebungstemperaturen liefern widerstandsbasierte Temperaturfühler gute Resultate. Bei höheren Leitungslängen oder hohen Störeinflüssen durch elektromagnetische Quellen ist der Einsatz eines 3-Draht oder 4-Draht Messprinzips für eine bessere Zuverlässigkeit der Messwerte empfehlenswert.

5.1.1 Anschluss

Für den Einsatz eines analogen Temperatursensors an der Klemme wird ein PT1000-Fühler im 2-Draht-Messprinzip an Port 5 und 6 angeschlossen. Die verwendete Klemme ist für den Einsatz verschiedener Pt- und Ni-Temperaturfühler konzipiert und je nach eingesetzten Fühlertyp muss eine individuelle Parametrierung des Kanals dafür erfolgen.

5.1.2 Parametrierung der Klemme und Fühler

Wie in Abbildung 25 gezeigt, werden für den Kanal 2 der Klemme, an welchen der Pt1000-Fühler angeschlossen ist, die entsprechenden Einstellungen und eine anschließende Kalibrierung durchgeführt,

um anschließend den Rohwert (der gemessene Widerstand des Fühlers) in einen Temperaturwert in °C umzuwandeln und als globale Variable AI_RTD_2 auszugeben.

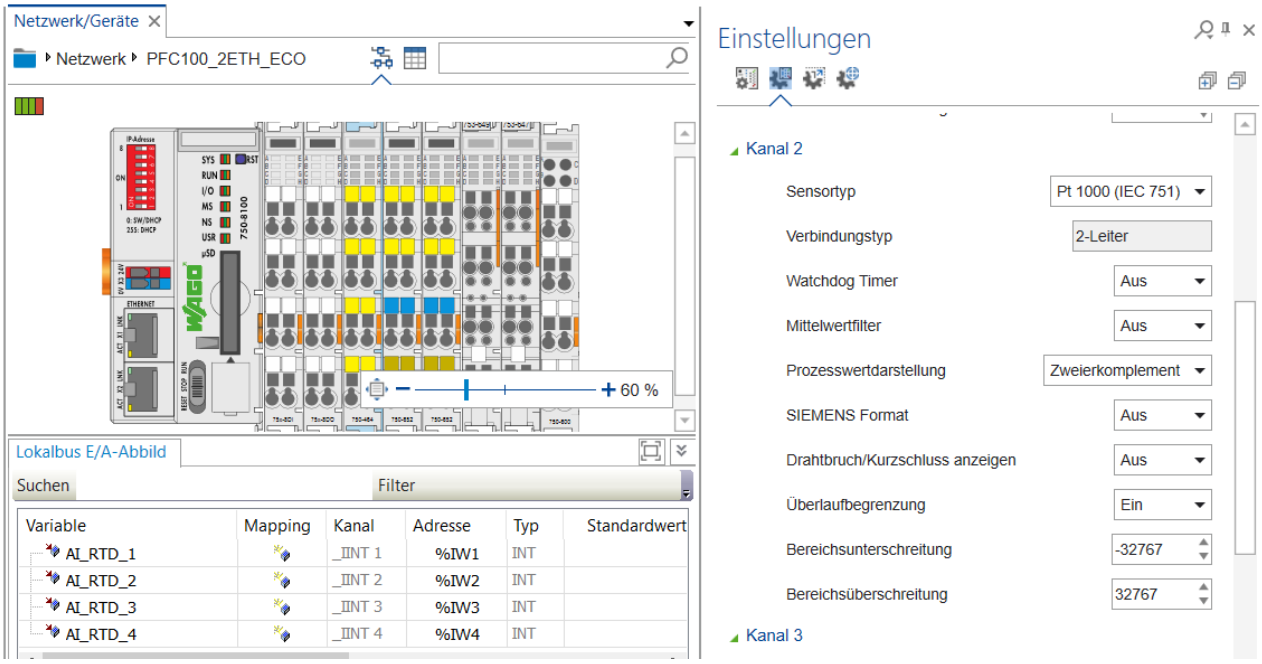


Abbildung 25: Parameteinstellungen der Klemme

5.1.3 Programmbeispiel zur Einbindung eines Pt1000-Temperaturfühlers

Im nächsten Schritt soll der gemessene Temperaturwert des Fühlers dazu verwendet werden, um eine einfache Temperaturregelung zu realisieren.

Hierfür wird an erster Stelle ein neues Programm mit dem Namen „Tutorial_Beispiel_2“ in der Implementierungssprache FUP erstellt und die in Abbildung 26 gezeigten Funktionsblöcke und Variablen eingesetzt. Nach Einfügen des Programmaufrufs in der PLC_PRG, dem Upload des Programms auf den Controller und dem Start des Programms wird anschließend der Temperaturwert der Klemme ausgelesen und mit einem Temperaturband zwischen 20°C und 23°C verglichen und bei Unter bzw. Überschreitung ein Befehl zum Heizen oder Kühlen ausgeführt.

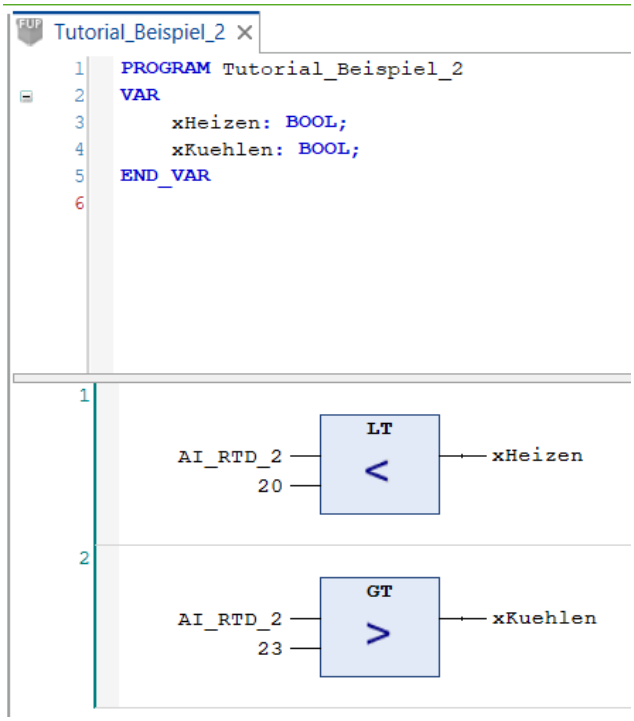


Abbildung 26: Programm einer einfachen Temperaturregelung

5.2 Einbindung von EnOcean-Sensoren über das Thermokon-Gateway

Bei EnOcean handelt es sich um einen in der Gebäudeautomation etabliertes Funksystem. Das System zeichnet sich durch Reichweiten in Gebäuden von ca. 30 Metern aus, wobei die Energie der Sensoren meist über „Energy Harvesting“ (Gewinnung von Energie aus der Umwelt) gewonnen wird. Vor allem Nachrüstungen von Bestandsinstallation können nicht nur schnell und einfach sondern auch wartungsfrei realisiert werden. Bei größeren Installationen ist darauf zu achten, dass ein EnOcean-Netzwerk sich nach einer Baumstruktur aufbaut. So werden Funksignale von EnOcean-Geräten an Gateways gesendet, welche die Daten auf z.B. eine serielle Schnittstelle umwandeln und somit zur weiteren Verarbeitung an z.B. DDC-Controller weiterleiten. Sollte die Reichweite eines Geräts nicht ausreichen, können zusätzliche Repeater in diese Baumstruktur eingebaut werden.

5.2.1 Anschluss

Zur Einbindung von EnOcean Sensoren und Aktoren wird ein EnOcean-Gateway der Firma Thermokon über RS485 an eine serielle Klemme angebunden. Hierfür muss die Verdrahtung, entsprechend Abbildung 27 aufgebaut sein, um die korrekte Funktion zu gewährleisten. Im Falle von Fehlern (Rückmeldung über LEDs der Klemme/des Gateways) sollte zuerst der richtige Anschluss geprüft werden.

Wichtig: Die gezeigten Brücken von Port 1 auf 2 und Port 5 auf 6 müssen immer berücksichtigt sein. Bei Kabellängen zwischen Gateway und Klemme unter einem Meter DARF kein Endwiderstand gesetzt sein. Bei größeren Entfernungen sollte dieser berücksichtig jedoch werden.

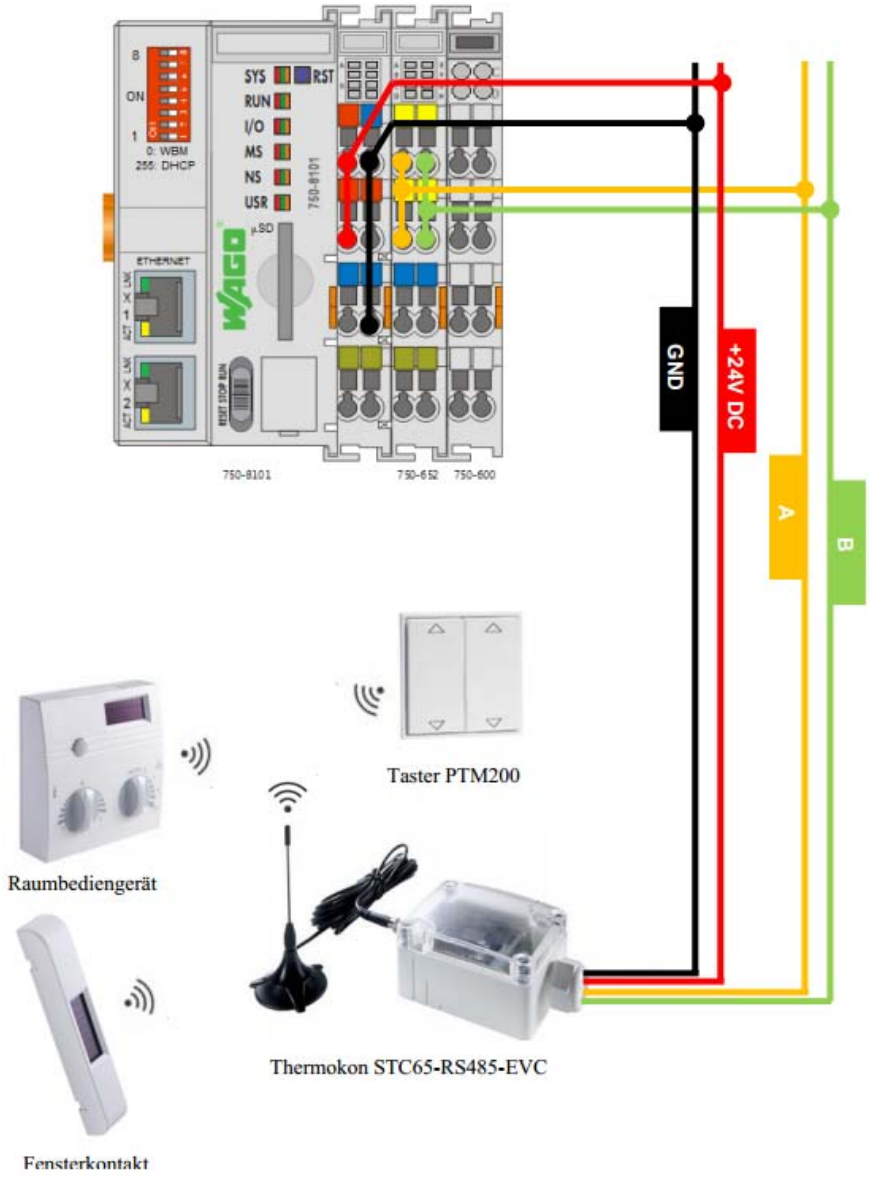


Abbildung 27: Verbindungsplan Thermokon-Gateway über RS485 (Quelle: WAGO)

5.2.2 Parametrierung der Klemme

Nachdem die Verdrahtung auf Richtigkeit geprüft wurde, sollte als nächstes die Parametereinstellung der Klemme geprüft werden. Dies geschieht, wie in Kapitel 3.3. beschrieben, im nicht verbundenen Zustand, durch Auswahl der entsprechenden 750-652 Klemme und Anklicken des Buttons „Einstellungen“ im rechten Bereich des Bildschirms. Es öffnet sich ein Fenster zur Konfiguration der Klemme. Hier müssen die in Abbildung 28 gezeigten Werte eingestellt sein.

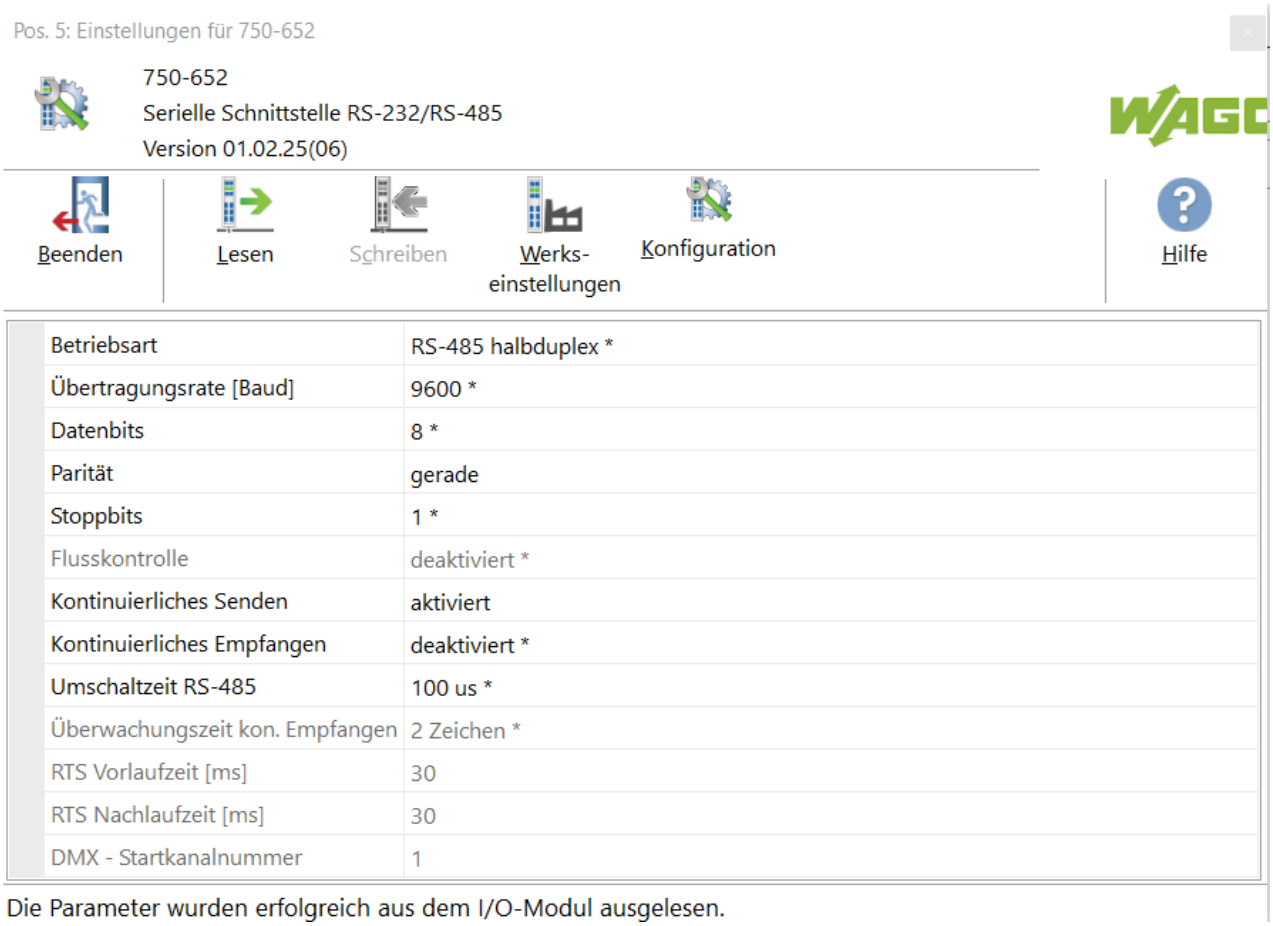


Abbildung 28: Parametereinstellungen 750-652 an Thermokon-Gateway

Sofern die Verkabelung stimmt und alle Parameter der Klemme richtig gesetzt sind, sollte das Thermokon-Gateway bei Betätigung eines EnOcean-Senders eine visuelle Rückmeldung der LEDs geben. Auch an den LEDs der 750-652 Klemme sollte dies durch Aufleuchten erkennbar sein.

5.2.3 Programmbeispiel zur Einbindung von EnOcean-Sensoren

Im nächsten Schritt soll anhand eines Beispiels die Ansteuerung eines EnOcean-SmartPlugs (ein schaltbarer Zwischensteckdosenstecker) über einen EnOcean-Taster gezeigt werden.

Hierzu wird ein neues Programm mit dem Namen „Tutorial_Beispiel_3“ in FUP erstellt und der Bibliotheksverwalter aufgerufen. Durch Auswahl des Buttons „Bibliothek hinzufügen“ erscheint ein Fenster, in welchem nach der Bibliothek „WAGOAppEnOcean“, wie in Abbildung 29 gezeigt, gesucht und anschließend eingefügt wird.

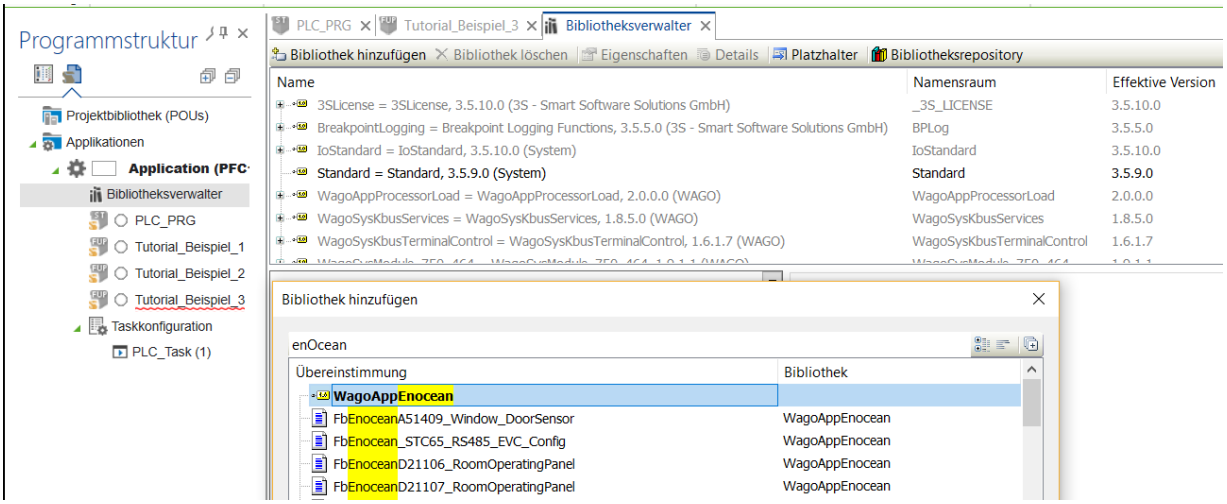


Abbildung 29: Hinzufügen der EnOcean-Bibliothek

Teil 1: Initialisieren

Nach dem Einfügen wird anschließend das Fenster des Programms geöffnet und die in Abbildung 30 gezeigten Bausteine mit den dazugehörigen Variablen eingefügt. Hierbei handelt es sich im Netzwerk 1 (dabei bezeichnet „Netzwerk“ einen waagrechten Bereich unterhalb der Variablendeklaration und ist mit Rechtsklick → „Netzwerk einfügen“ hinzuzufügen) um den Baustein zum Aufruf der seriellen Kommunikation mit dem Gateway, dieser muss immer zuerst im Programm aufgerufen werden. Wichtig ist hierbei, dass am Eingang „I_Port“ die richtige RS-485-Klemme angegeben ist. Diese muss den Namen, wie in der Gerätekonfiguration angegeben, besitzen (Achtung: Somit sollte der Name in Abbildung 30 „RS_232_RS_485_Interface_Adjustable_2“ eigentlich „RS232_485_Interface“ heißen, um mit der Gerätekonfiguration in Abbildung 11 übereinzustimmen). Der zweite (optinale) Funktionsbaustein zeigt ankommende EnOcean-SenderIDs an.

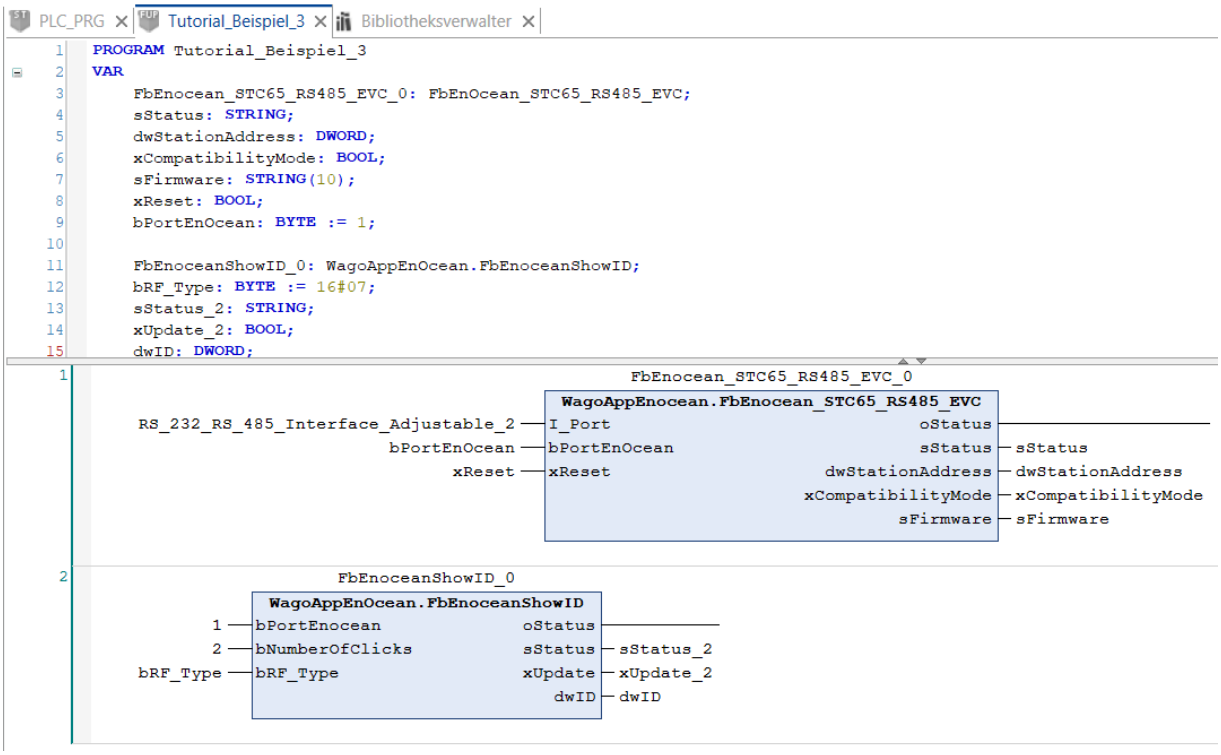


Abbildung 30: Netzwerke zum Klemmenaufruf und Auslesen der EnOcean-IDs

Teil 2: Geräte-IDs identifizieren:

Mit Hilfe des FbEnoceanShowID-Bausteins kann die ID eines Sensors angezeigt werden. Dazu ist der Eingang bRF_Type zu ändern. Für den eingesetzten Taster ist dieser F6, entsprechend der ersten Zahl seines EEP-Typs (**F6**-02-02). Dies ergibt den Hexadezimalwert 16#F6. Wird dieser am Eingang bRF_Type eingetragen (246 ist hierbei die Dezimalwertdarstellung), gibt der Baustein alle IDs von Geräten dieses bRF_Typs aus, wobei diese als Dezimalzahl ausgegeben werden. Im laufenden Programm kann der Typ geändert werden, indem dieser zuerst durch Doppelklick auf das orange Feld eingetragen wird und anschließend im Menüpunkt „Programmierfunktionen“ → „Debug“ → „Werte“ → „Schreiben“ als aktueller Wert gesetzt wird.

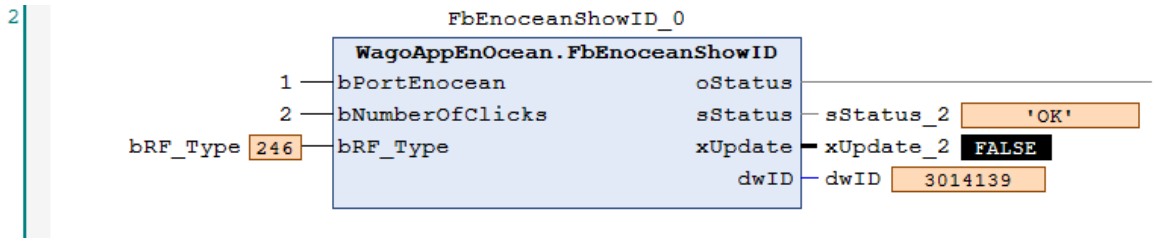


Abbildung 31: FbEnoceanShowID im aktiven Programm

Alternativ können ID’s über einen EnOcean-USB-Stick mit der Auswertungssoftware DolphinView ermitellt werden. Diese liefert das in Abbildung 32 gezeigte Ergebnis. Diese ID’s werden als Hexadezimalzahlen dargestellt werden, womit die ID 0x002DFDFB in DolphinView mit der ID 3014139 in e!Cockpit übereinstimmt.

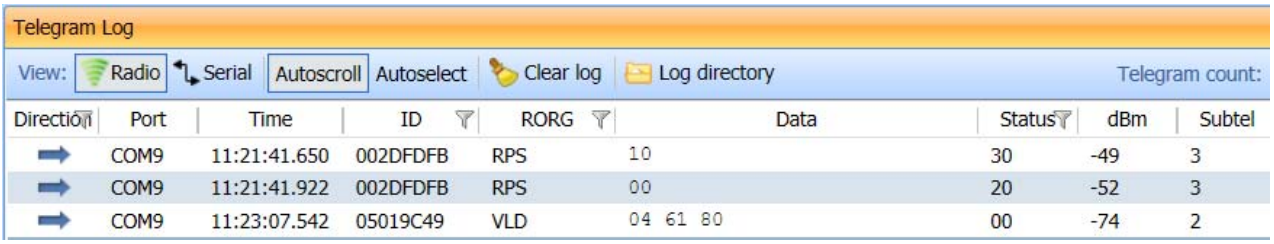


Abbildung 32: Telegram Log in DolphinView

Im Log wird festgehalten, wann welches EnOcean-Telegramm empfangen wurde. Hierbei ergibt sich für die ersten beiden Telegrame, welche vom EnOcean-Taster gesendet wurden, die **Device-ID: 002DFDFB** und für den SmartPlug, welcher als zweites betätigt wurde, die **Device-ID: 05019C49**. Diese IDs werden anschließend für den weiteren Verlauf der Programmierung benötigt.

Teil 3: Taster und Aktor einfügen

Um den EnOcean-Taster einbinden zu können, wird das Programm um einen weiteren Baustein erweitert. Da es sich um einen Taster des EnOcean Equipment Profile (EEP)-Typs F6-02-02 (Information aus Datenblatt entnehmen) handelt, wird in der Eingabehilfe zum Einfügen eines Funktionsbausteins, der in Abbildung 33 gezeigte Funktionsbaustein ausgewählt.

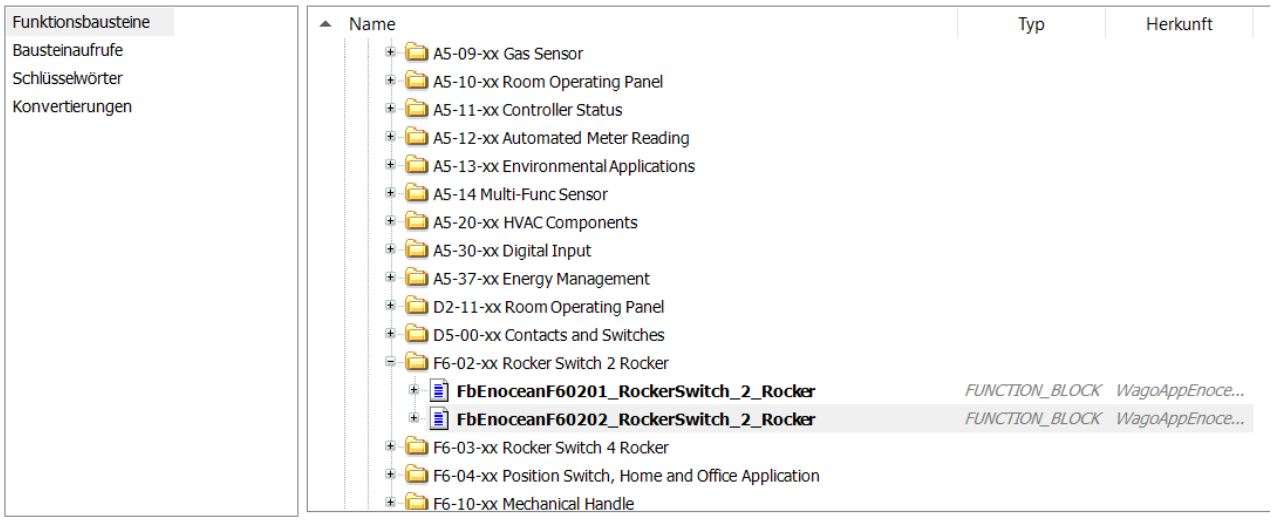


Abbildung 33: Eingabehilfe zum Einfügen eines Funktionsbausteins

Anschließend wird ein weiteres Netzwerk benötigt, in welchem der SmartPlug aktiviert/deaktiviert werden kann. Dazu wird der Baustein FbEnOcean_RPS_Send verwendet.

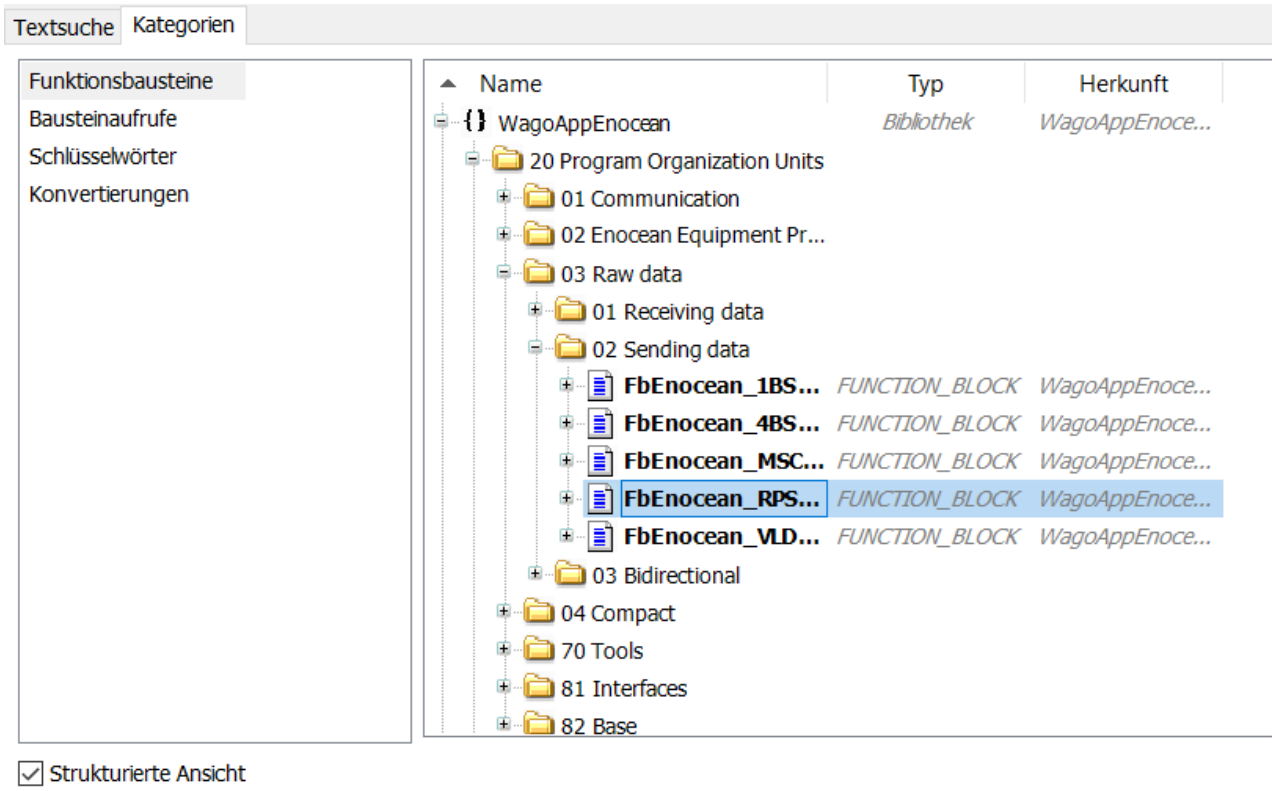


Abbildung 34: FbEnOcean_RPS_Send einfügen

Jetzt können die Variablen zugewiesen und die Geräte-IDs eingetragen werden. Abbildung 35 zeigt den fertigen Aufbau des Programms.

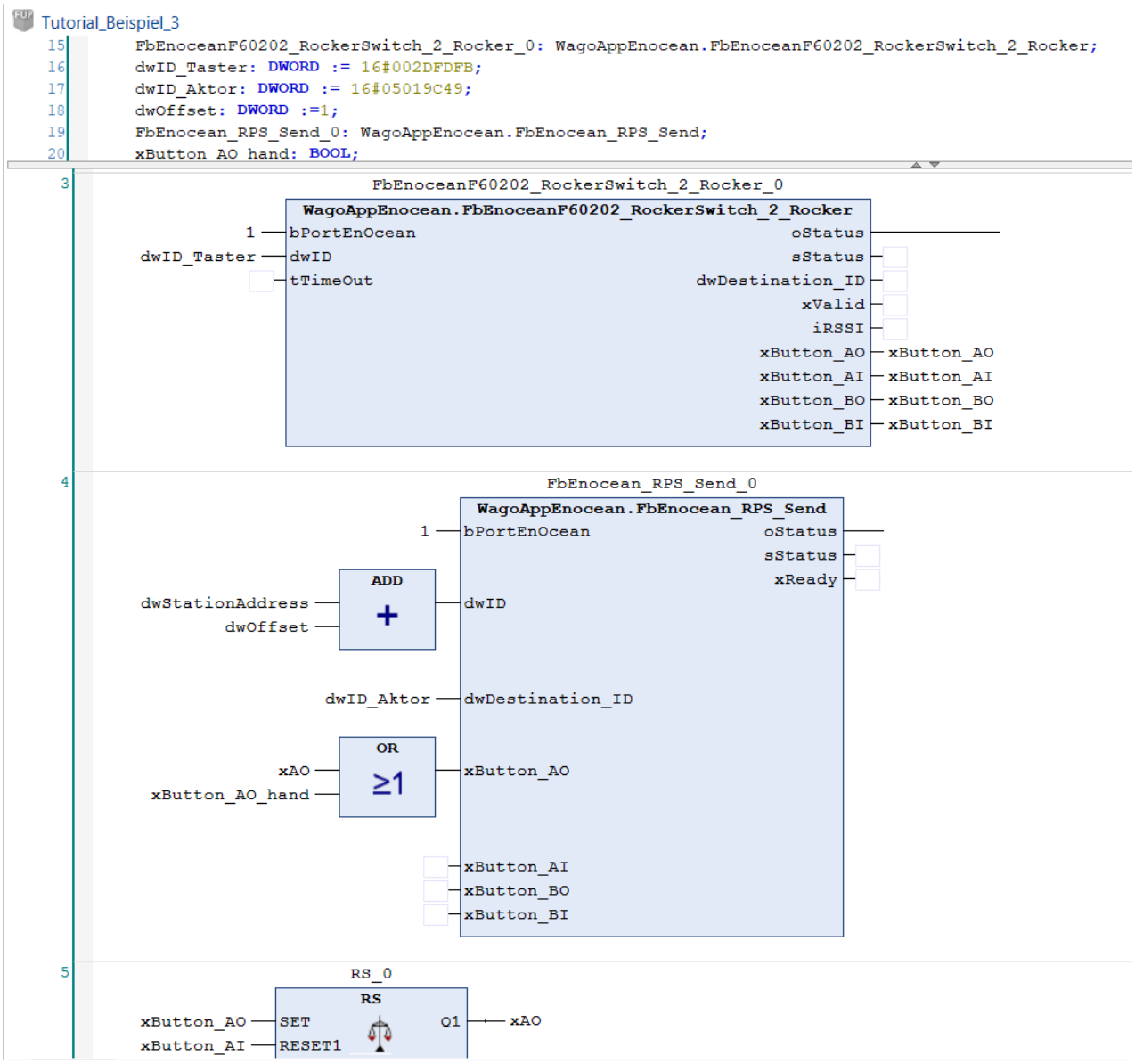


Abbildung 35: SmartPlug über Taster schalten

Um einen Zwischenstecker der Fa. Nodon („SmartPlug“) über den Controller steuern zu können muss das Gateway erst auf diesen Aktor eingelernt werden. Dazu steckt man den Aktor an eine Steckdose, so dass dieser mit Strom versorgt wird. Nun betätigt man den Button am Gerät für ca. zwei Sekunden. Die LED beginnt nun rot zu blinken – das Gerät befindet sich im Lernmodus und wird nun, durch beispielsweise Betätigen eines Tasters auf diesen eingelernt. Um mit dem Thermokon-Gateway den SmartPlug anzusteuern, muss bei laufender SPS das Gerät in den Lernmodus versetzt und anschließend der Button „xButton_AO_hand“ gesetzt werden. Der SmartPlug ist nun auf einen „virtuellen Button“ des Bausteins „FbEnocean_RPS_Send“ eingelernt. Durch Betätigen des Buttons AO des Tasters kann dieser den SmartPlug anschließend an und abschalten. Der Vorteil gegenüber dem direkten Einlernen mit dem Taster liegt im einfachen Austauschen oder Erweitern von Variablen im Programmcode.

5.3 Einbindung der Elsner Wetterstation

Beim Einsatz einer Wetterstation der Firma Elsner gilt zu beachten, dass zwei unterschiedliche Typen der Wetterstation „P03/3“ angeboten werden. Der eine Typ verfügt über eine Modbus-Schnittstelle und wird von e!Cockpit mit der Bibliothek „WAGOSolElsner“ unterstützt. Der andere Typ verfügt über eine RS485-Schnittstelle. Diese wird nicht von der Standard-Bibliothek unterstützt. Auf Anfrage wurde von WAGO hierzu jedoch einen Funktionsbaustein bereitgestellt, welcher das Auslesen der Messwerte einer P03/3 RS485 übernimmt. Die Einbindung dieses Bausteins und die Auswertung der Messwerte wird im nachfolgenden gezeigt.

5.3.1 Anschluss

Zu Beginn sollte ein entsprechender Anschluss nach Abbildung 36 geprüft werden. Ähnlich wie beim Thermokon-Gateway kann bei fehlerhaftem Anschluss, zu langen Leitungslängen (Notwendigkeit eines Abschlusswiderstands) oder Drahtbruch eine Kommunikation fehlschlagen. Bei einer Neu- oder Umverdrahtung ist vor allem auf den korrekten Anschluss der 24V-Leitung zu achten, da die Wetterstation nicht verpolungssicher ist und bei falschem Anschluss zerstört werden kann. Nähere Informationen enthält das Datenblatt.

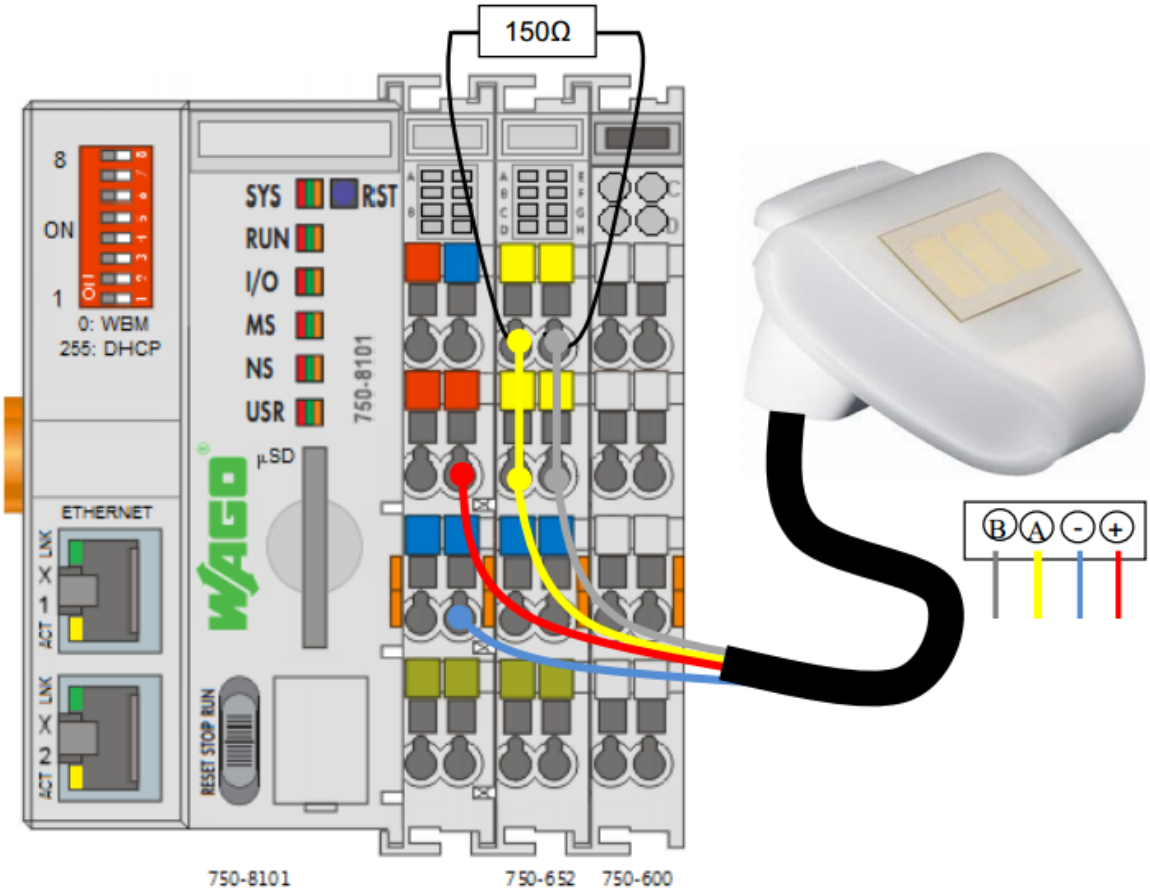


Abbildung 36: Verbindungsplan Wetterstation über RS485 (Quelle: WAGO)

5.3.2 Parametrierung der Klemme

Ähnlich wie bei der Kommunikation mit dem Thermokon-Gateway gilt es auch, die Kommunikationsparameter der Klemme anzupassen, damit die Daten der Wetterstation empfangen werden können. Sofern nicht voreingestellt, sollten diese über den IO-Check, entsprechend Abbildung 37 angepasst und in die Klemme geschrieben werden.

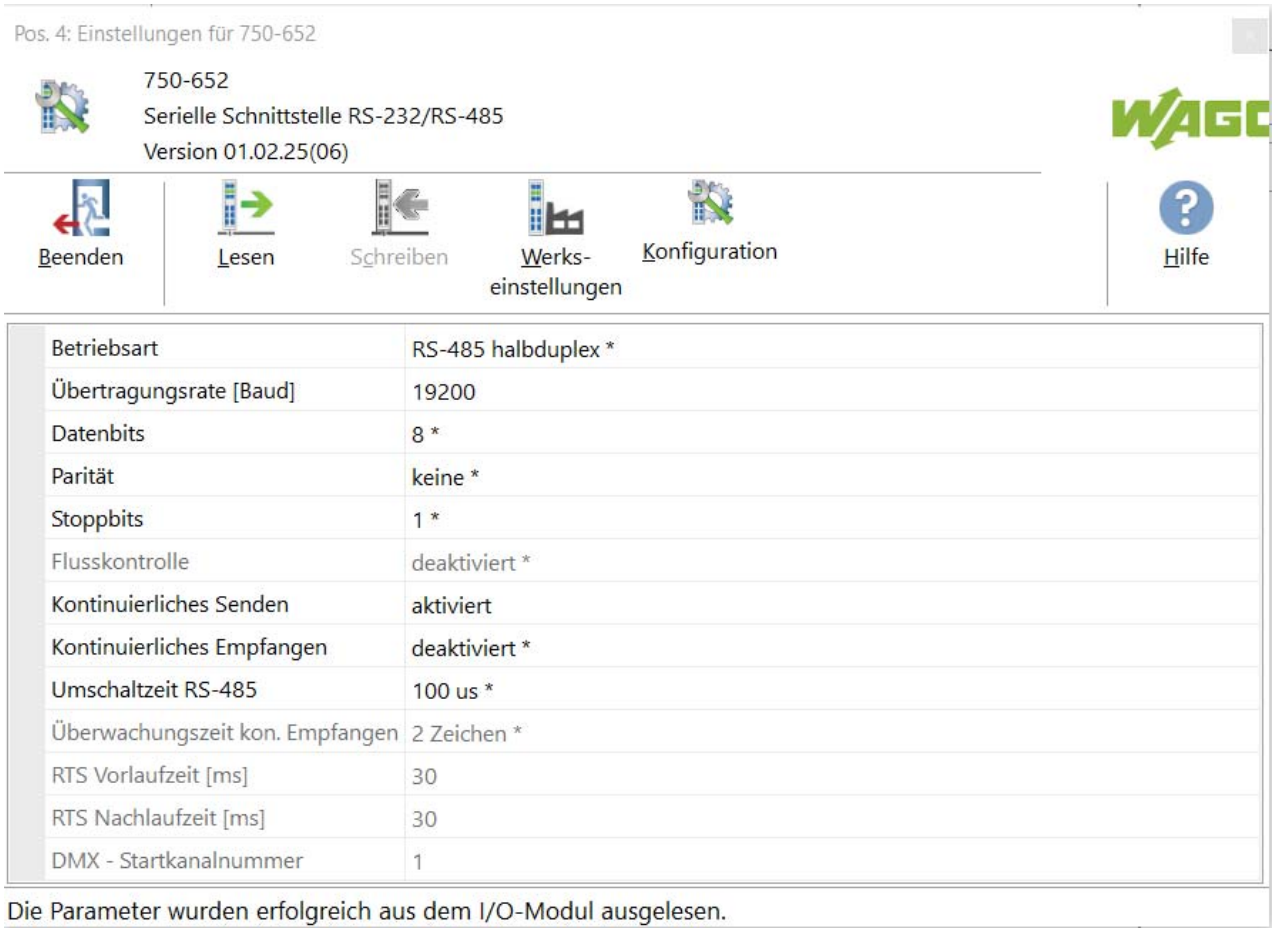


Abbildung 37: Parametereinstellungen 750-652 an Elsner Wetterstation

5.3.3 Programmbeispiels zur Einbindung der Wetterstation

Nachdem die Verkabelung und die Klemmeneinstellungen geprüft und ggf. angepasst wurden, kann mit der programmierseitigen Einbindung begonnen werden.

Hierzu wird ein neues Programm mit dem Namen „Tutorial_Beispiel_4“ in der Implementierungssprache FUP erstellt. Parallel dazu wird eine von WAGO bereitgestellten Datei mit dem Namen „MeteorologicalStation.export“ geöffnet und der Funktionsbaustein „MeteorologicalStation“ kopiert und in die Applikation des PFC100 eingefügt. Dieser Baustein kann wie gewöhnlich im Programm „Tutorial_Beispiel_4“ aufgerufen und die Variablen deklariert werden. Abbildung 38 zeigt den eingesetzten Baustein MeterologicalStation im Programm.

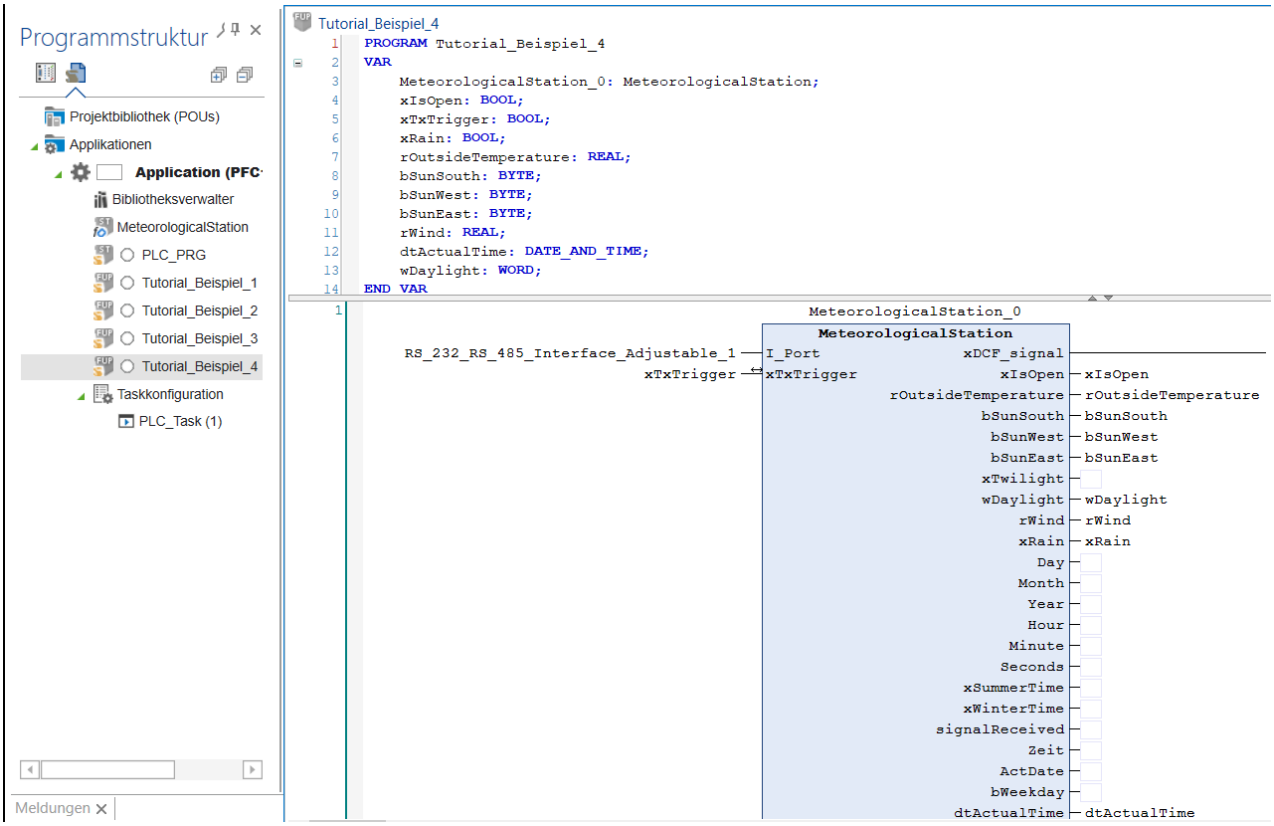


Abbildung 38: Fertiges Programm zu Beispiel 4

Der eingefügte Funktionsbaustein benötigt eine zusätzliche Bibliothek „WAGOAppCom“, welche wie in Abbildung 39 gezeigt, in die Bibliotheksverwaltung hinzugefügt werden muss.

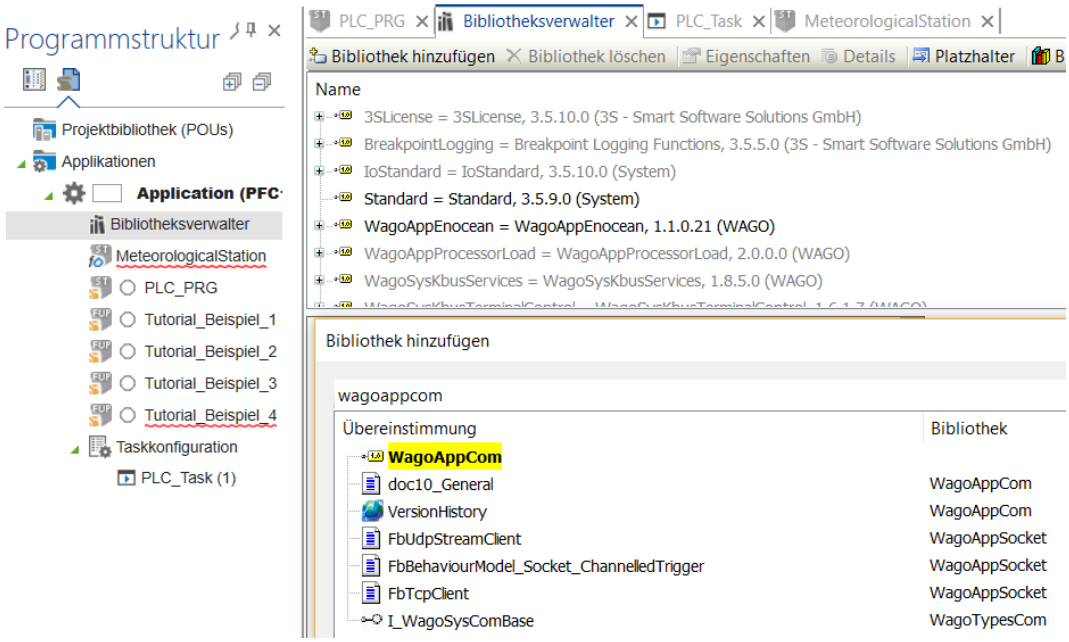


Abbildung 39: WAGOAppCom einfügen

Nachdem der Programmaufruf **Tutorial_Beispiel_4();** in das Programm PLC_PRG eingetragen wurde, kann die Applikation auf den Controller geladen und gestartet werden. Der GPS-Empfänger benötigt je nach Beschaffenheit der Umgebung (Indoor- oder Outdooreinsatz der Wetterstation) ein paar Minuten, bis dieser ein stabiles Signal für die Zeitangabe gefunden hat.

5.4 Einbindung der M-Bus-Klemme

Als nächstes wird beschrieben, nach welchem Vorgehen einer oder mehrere M-Buszähler ausgelesen werden können.

In einem M-Bus-Netzwerk können bis zu 256-Teilnehmer kommunizieren. Dabei besitzt jeder Teilnehmer eine eigene Adresse; diese kann für gewöhnlich direkt am Zähler ausgelesen bzw. eingestellt werden. Der in diesem Tutorial verwendete Elektrozähler besitzt die Adresse 2.

5.4.1 Anschluss

Der Anschluss des M-Bus erfolgt mittels einer M-Bus Masterklemme 753-649, wie in Abbildung 40 gezeigt. Bei der Verkabelung werden lediglich zwei Drähte verwendet, welche in einer beliebigen Topologie an die Zähler angeschlossen werden. Dabei gilt es nicht zu berücksichtigen welches Datenkabel auf welchem Port am Zähler anliegt.

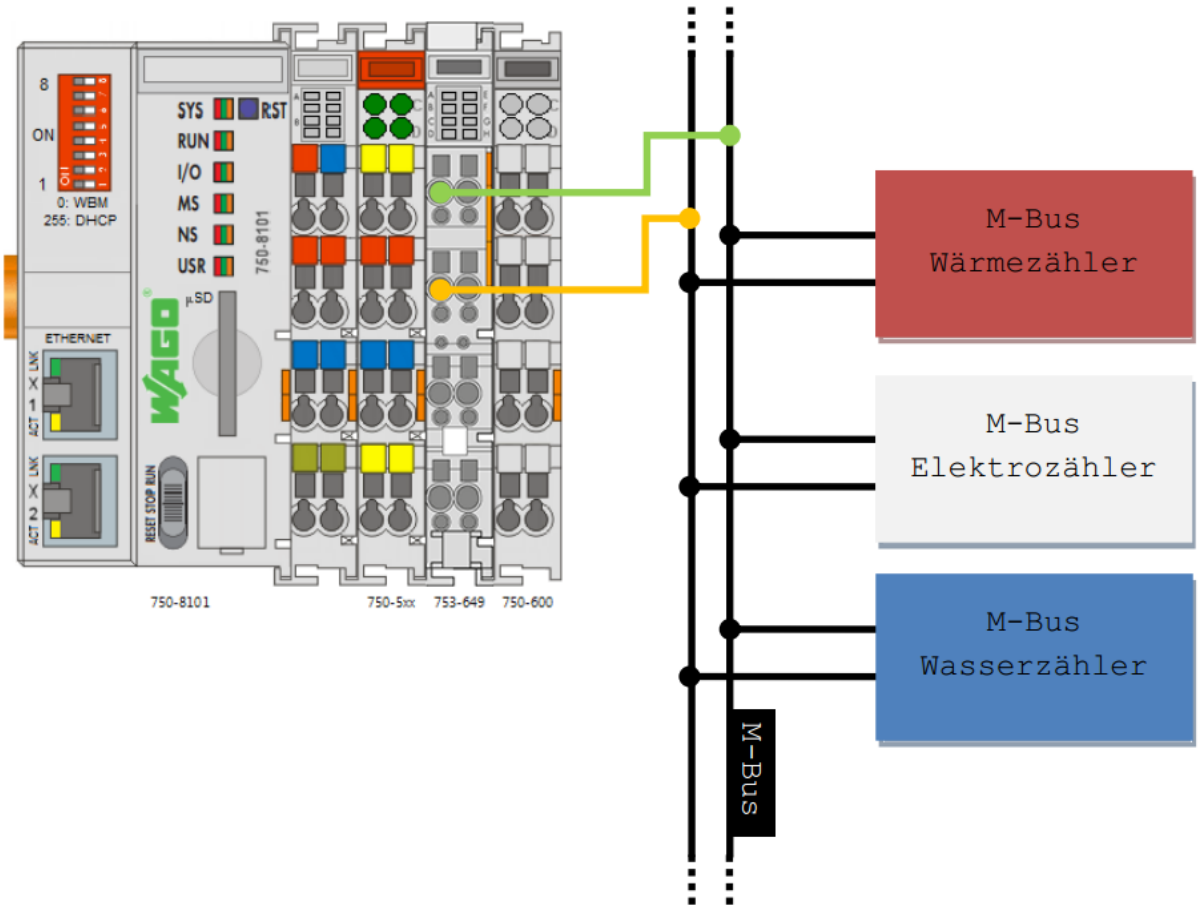


Abbildung 40: Verbindungsplan M-Bus (Quelle: WAGO)

5.4.2 M-Bus-Teilnehmer suchen

Zuerst wird der Klemme der Name „M_Bus“ gegeben. Eine Parametrierung der Klemme ist nicht notwendig. Es können, sofern keine Applikation auf dem Controller aktiv ist, die angeschlossenen M-Bus-Teilnehmer über den M-Bus Sheet ausgelesen werden. Hierzu wird wie für das Einstellen einer Klemme der der IO-Check aktiviert - dieser ruft anschließend das in Abbildung 41 gezeigte Programm auf.

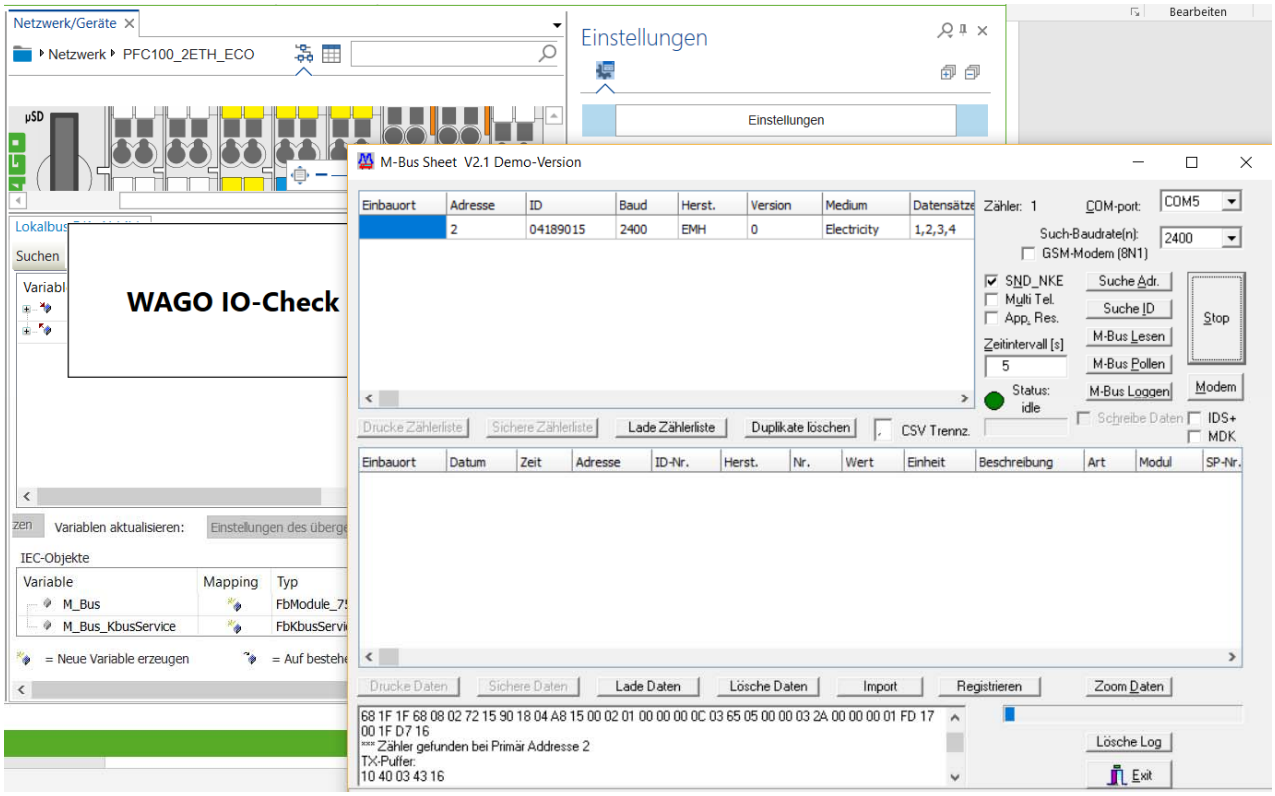


Abbildung 41: M-Bus mit externen Programm scannen

Das Ergebnis dieses Scans zeigt die Liste aller angeschlossenen Zähler.

5.4.3 Programm-Beispiels zur Einbindung des M-Bus

Sofern ein ordnungsgemäßer Anschluss des Zählers vorliegt und dieser über das Display keinen Fehler ausgibt, kann mit der Programmierung begonnen werden.

Es wird ein neues Programm mit dem Namen „Tutorial_Beispiel_5“ in der Implementierungssprache FUP angelegt. Als nächstes wird im Bibliotheksverwalter die Bibliothek „WAGOAppM_Bus“ der Applikation hinzugefügt, im Programm der Baustein „FbMBusMaster“ aufgerufen und die Ein- und Ausgänge entsprechend Abbildung 42 deklariert.

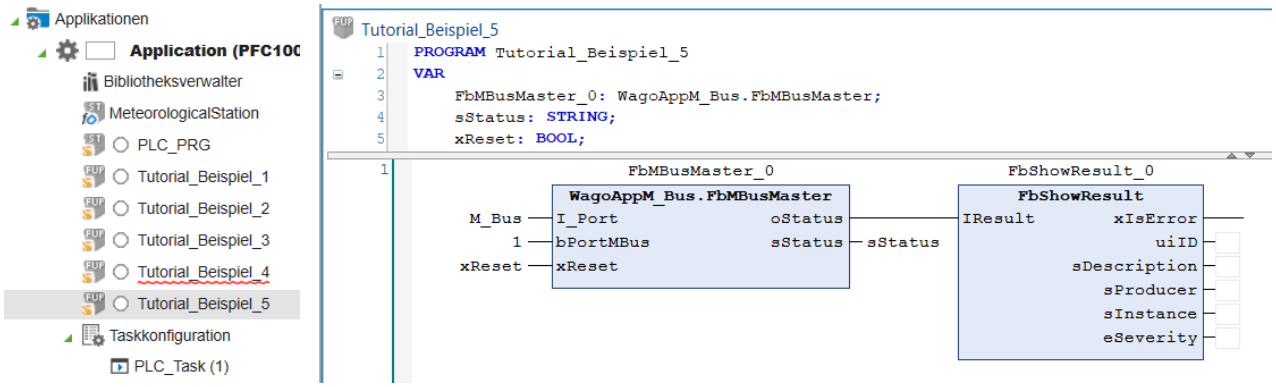


Abbildung 42: M-Bus-Master Bausteinaufruf

Im nächsten Schritt wird, wie in Abbildung 43 gezeigt, ein Netzwerk mit einer Einschaltverzögerung eingefügt, ein weiteres Netzwerk mit dem Funktionsbaustein „FbMBusElectricity“ angelegt und mit den gezeigten Ein- und Ausgängen deklariert. Hierbei ist es wichtig die „dwAdress“ entsprechend der Adresse des Elektrozählers zu vergeben. Die Einschaltverzögerung wird im laufenden Programm zyklisch den Elektrozähler abfragen und den Baustein „FbMBusElectricity“ regelmäßig updaten. Der Baustein „FuUnitConvert“ dient lediglich dem Sinn die Variable typEnergy in kWh umzuwandeln und diese als Real-Wert auszugeben.

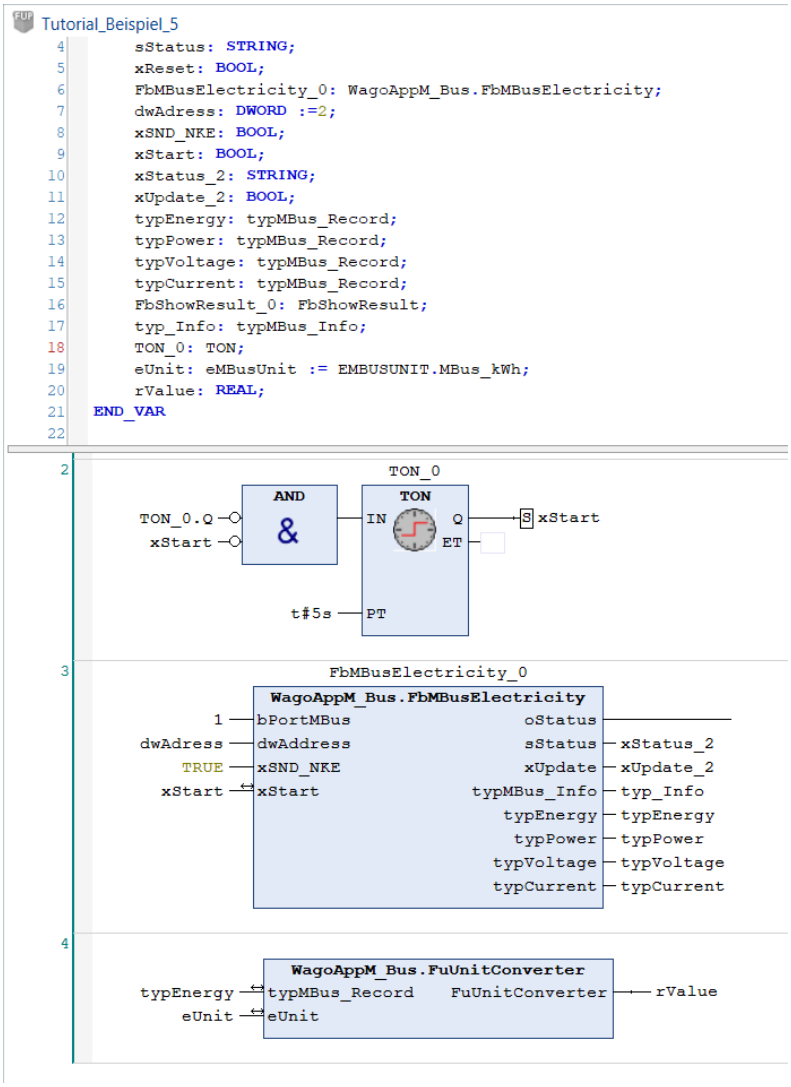


Abbildung 43: Daten des M-Buszählers abrufen

Nach diesem Vorgehen können nun weitere Zählerwerte des Elektrozählers ausgelesen und umgewandelt werden oder weitere Zähler mit verschiedenen Adressen eingefügt werden. Für Wärmemengen- und Wasserzähler gibt es hierfür wiederum andere Bausteine.

5.5 Einbindung der DALI-Klemme

Der DALI-Bus dient der Ansteuerung von elektrischen Vorschaltgeräten (EVGs) oder LED-Treibern für DALI-fähige Beleuchtungskomponenten und ermöglicht zugleich die Einbindung von Multisensoren (z.B. Helligkeit, Präsenz und Remote-Control) in ein und denselben Bus. Die Leuchten (maximal 64 Leuchten pro Multi-Masterklemme möglich) verfügen dabei über ihre eigene Spannungsversorgung und werden lediglich über ein Zweidraht-Buskabel in beliebiger Topologie verbunden. Sensoren (maximal 16 Multisensoren pro Multi-Masterklemme) werden über den Bus mit Betriebsspannung versorgt.

Wichtig: Die bisherige DALI-Klemme 750-641 wird vom PFC100 nichtmehr unterstützt. Es kann lediglich die neuere DALI-Multi-Masterklemme 753-647 verwendet werden.

5.5.1 Anschluss

Für den Einsatz des DALI-Bus ist der in Abbildung 44 dargestellte Anschluss der DALI-Multi-Masterklemme zu berücksichtigen. Die Datenkabel müssen lediglich an die Ports DA+ und DA- und an die Anschlüsse des LED-Treibers beliebiger angeschlossen werden. Die Stromversorgung an U+ und U- muss 18V betragen, dazu ist der Einsatz eines DC-DC-Wandlers oder der in diesem Beispiel verwendeten DALI-DC-Converter-Klemme 753-620 nötig.

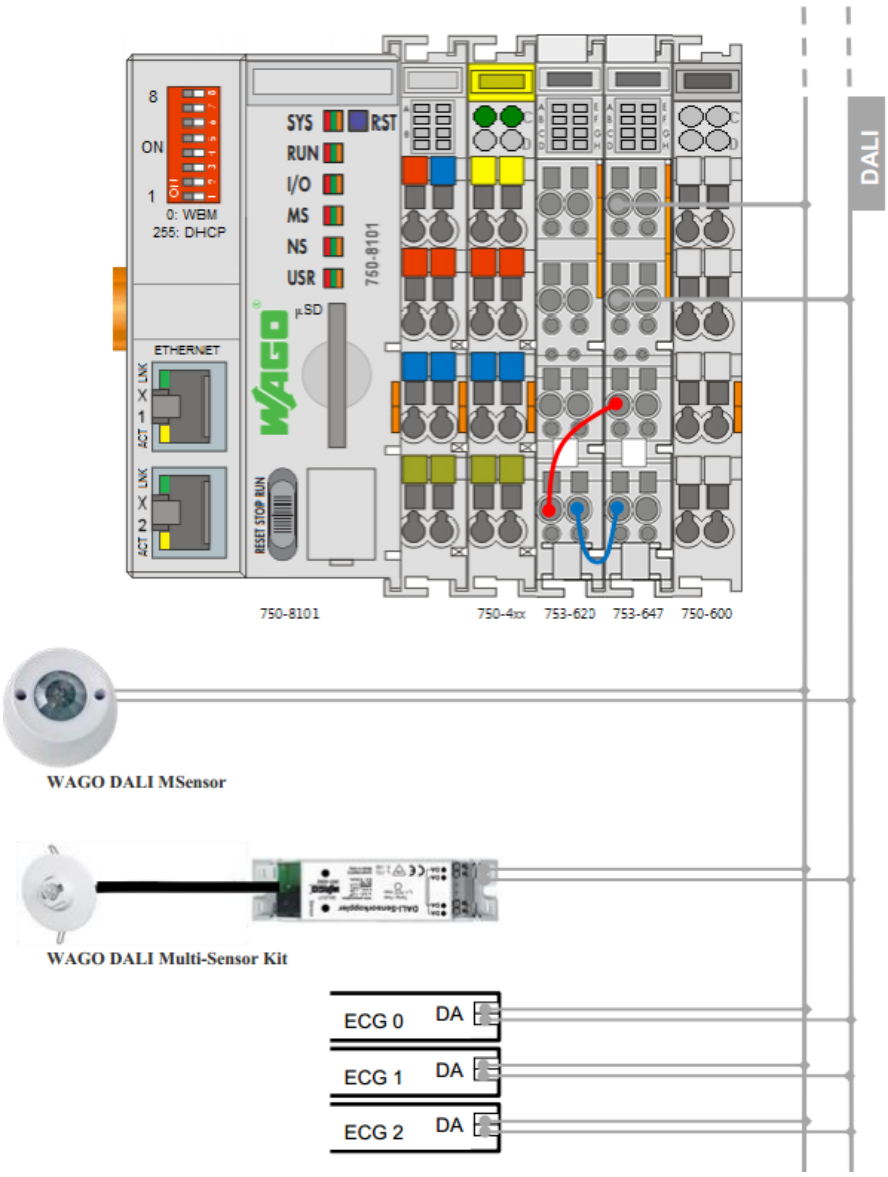


Abbildung 44: Anschlussdiagramm der DALI-Multimaster-Klemme (Quelle: WAGO)

5.5.2 DALI-Bus-Teilnehmer scannen

Eine Parametrierung ist nicht möglich. Jedoch kann, um festzustellen, welche Geräte am Bus angeschlossen sind, ein zusätzliches Softwaretool „WAGO DALI Configurator“ verwendet werden.

Im WAGO DALI Configurator wird über den Button „Einstellungen“ die Kommunikationseinstellung aufgerufen und die IP-Adresse des Controllers eingegeben und bestätigt.

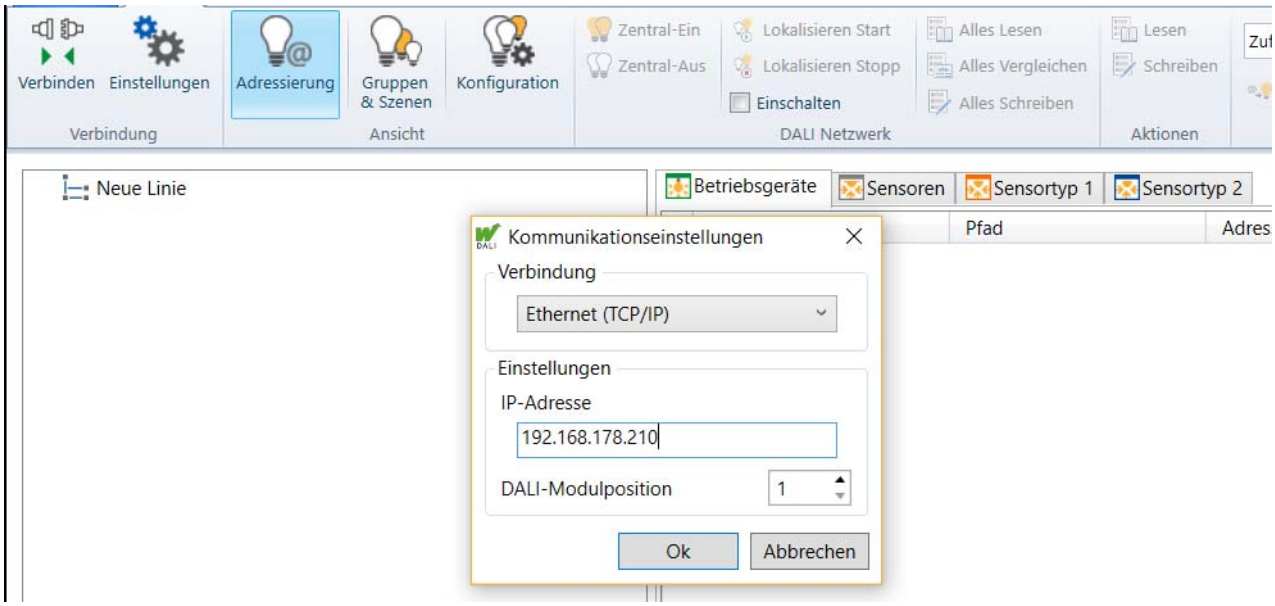


Abbildung 45: Verbindung mit DALI-Klemme über WAGO DALI Configurator herstellen

Anschließend wird über den Button „Verbinden“ die Verbindung zur Klemme hergestellt und über den Button „Lesen“ werden die angeschlossenen DALI-EVGs ausgelesen. Mit Rechtsklick auf ein beim Scan gefundenes EVG kann es umbenannt werden (Abbildung 46).

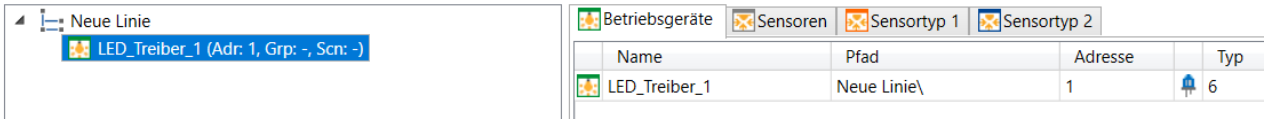


Abbildung 46: Ausgelesene DALI-Teilnehmer umbenennen

5.5.3 Programmbeispiel zur Einbindung des DALI-Bus

Es wird ein neues Programm mit dem Namen „Tutorial_Beispiel_6“ in der Implementierungssprache FUP angelegt. Als nächstes wird im Bibliotheksverwalter die Bibliothek „WAGOAppDALI“ der Applikation hinzugefügt und im Programm der Baustein „FbDALIMaster“ aufgerufen und die Ein- und Ausgänge entsprechend Abbildung 47 benannt. Zu beachten ist hierbei, dass die Deklaration des typBallast des LED_Treiber_1 korrekt vergeben wird, da dieser die anzusprechende DALI-Leuchte angibt. Über den im Programm „Tutorial_Beispiel_3“ verwendeten EnOcean-Taster kann nun der LED-Treiber angesteuert, ein- und ausgeschaltet, sowie hoch und runter gedimmt werden. Alternativ kann dies über die Variablen xOn und xOff über das Programm auch manuell erfolgen. Der Programmaufruf in PLC_PRG ist zu beachten.

WAGO PFC100-Tutorial

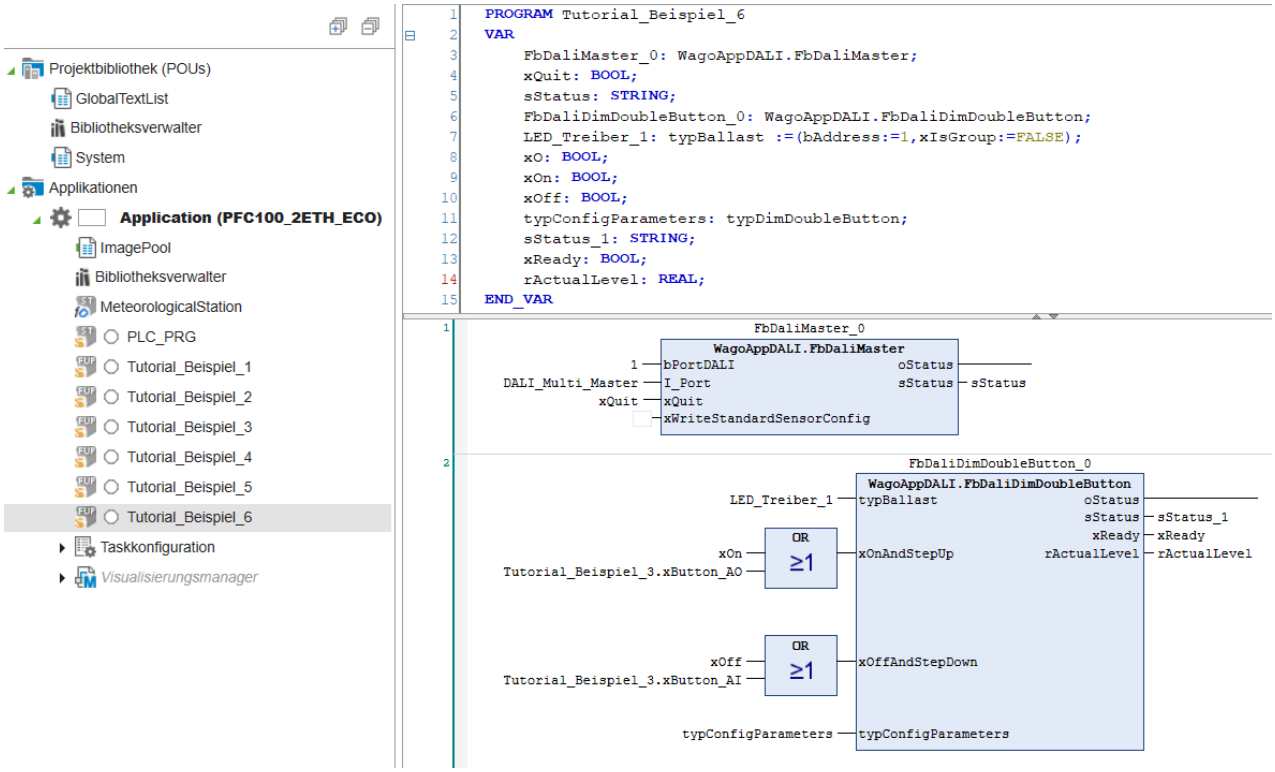


Abbildung 47: Programmaufbau zur Steuerung einer LED-Leiste über den DALI-Bus

6 Visualisierung in e!Cockpit

Für die in Kapitel 5.3 beschriebene Wetterstation soll im Nachfolgenden eine Visualisierungsseite, für die Darstellung der Messwerte, erstellt werden. Diese kann anschließend über den Browser eines PCs oder eines mobilen Gerätes aufgerufen werden.

6.1 Erstellung einer Visualisierung

Hierzu wird im ersten Schritt eine neue Visualisierung durch Rechtsklick auf Application hinzugefügt.

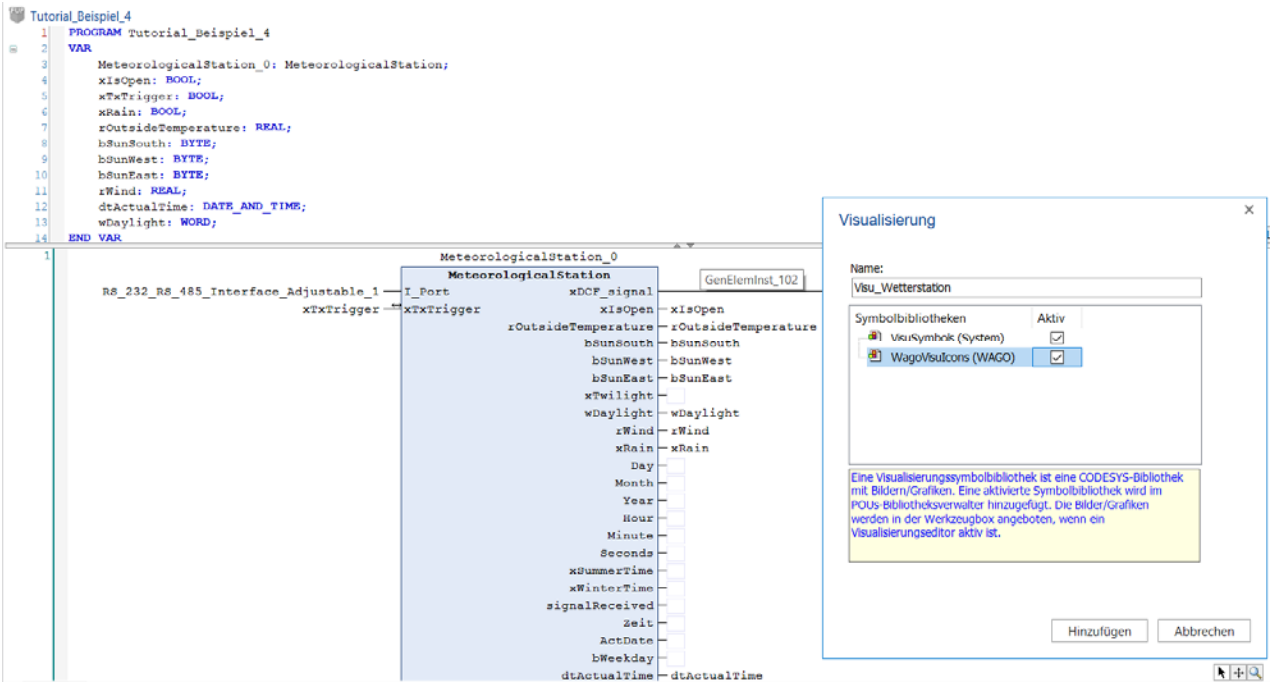


Abbildung 48: Visualisierung hinzufügen

In e!Cockpit werden anschließend automatisch neue Dateien eingefügt. VISU_TASK ist dabei für den Aufruf der Visualisierung verantwortlich. Ähnlich wie PLC_PRG ist sie essenziell für den Aufruf der Visualisierungen. In „WebVisu“ können zusätzlich Anpassungen an die Darstellung geändert werden (Abbildung 49).

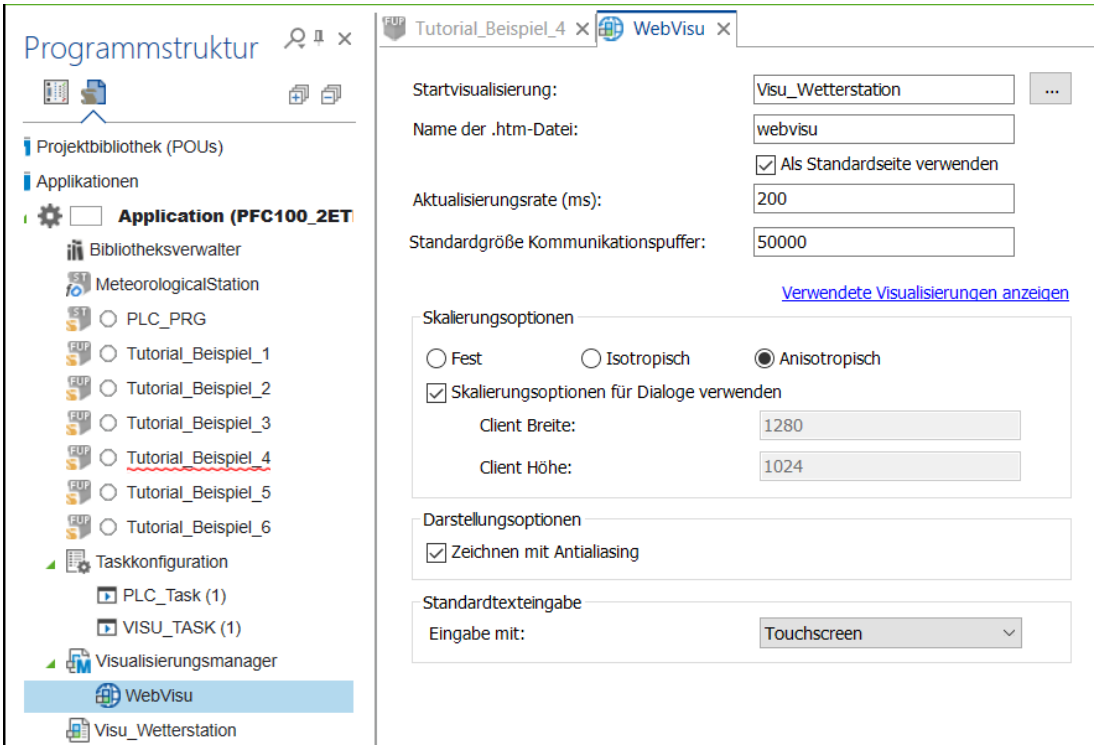


Abbildung 49: WebVisu

In „Visu_Wetterstation“ kann mit der Visualisierung der Wetterstation begonnen werden. Hierzu werden die Werkzeuge „Beschriftung“ und „Rechteck“ in die Visualisierung eingefügt.

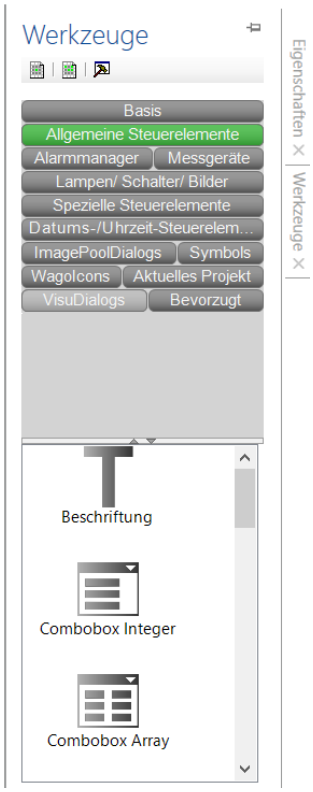


Abbildung 50: Visualisierungs-Werkzeuge

Für das Werkzeug „Rechteck“ wird anschließend die Toolbox „Eigenschaften“ geöffnet und die in Abbildung 51 markierten Einstellungen für jede Variable vorgenommen. Dies wird nun wie gezeigt für die notwendigen Ausgangs-Variablen des Programms „Tutorial_Beispiel_4“ durchgeführt.

Hinweis: Um eine externe Bilddatei in das Projekt einzubinden, wird mit Rechtsklick auf Application ein neuer „ImagePool“ erstellt und die Bilddateien per Drag and Drop eingefügt. PNG-Dateien schienen hierbei in der späteren Visualisierung nicht zu funktionieren. Das JPG-Format sollte hierfür gewählt werden.

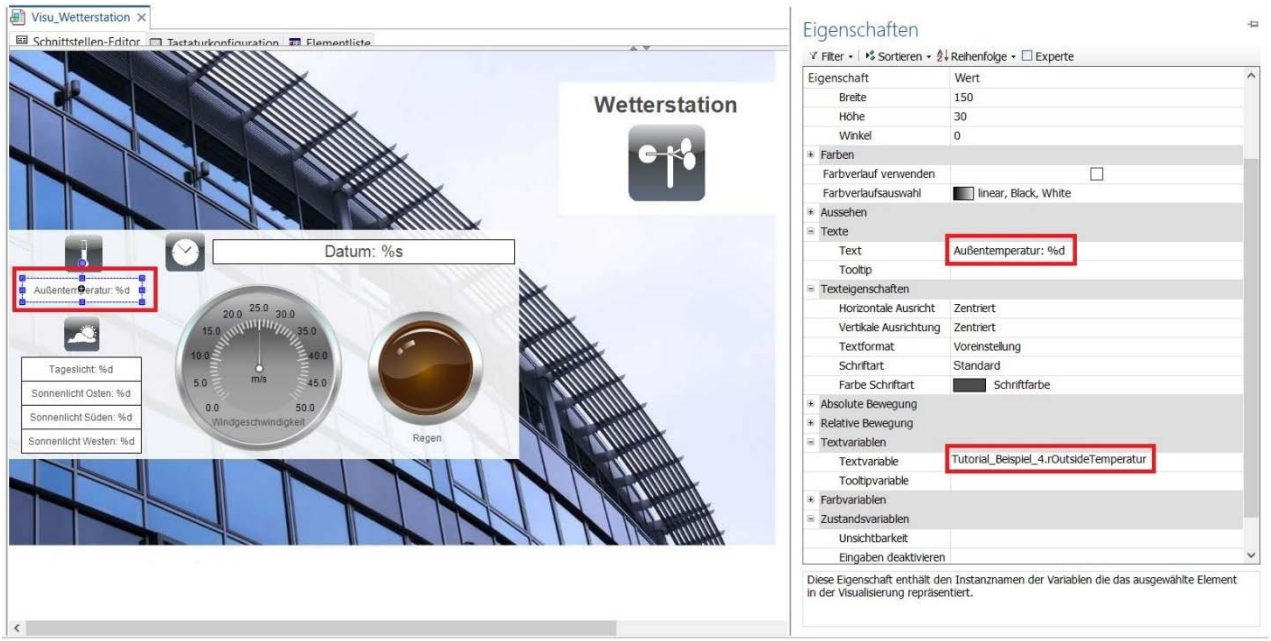


Abbildung 51: Bearbeitung der Visualisierung

6.2 Inbetriebnahme der WebVisu

Nachdem die Visualisierung fertig gestaltet wurde, sollte vor dem Start des Programms geprüft werden, ob im WBM des Controllers, wie in Abbildung 52 gezeigt, der Webserver aktiviert ist oder dieser, wenn nötig, aktiviert werden.

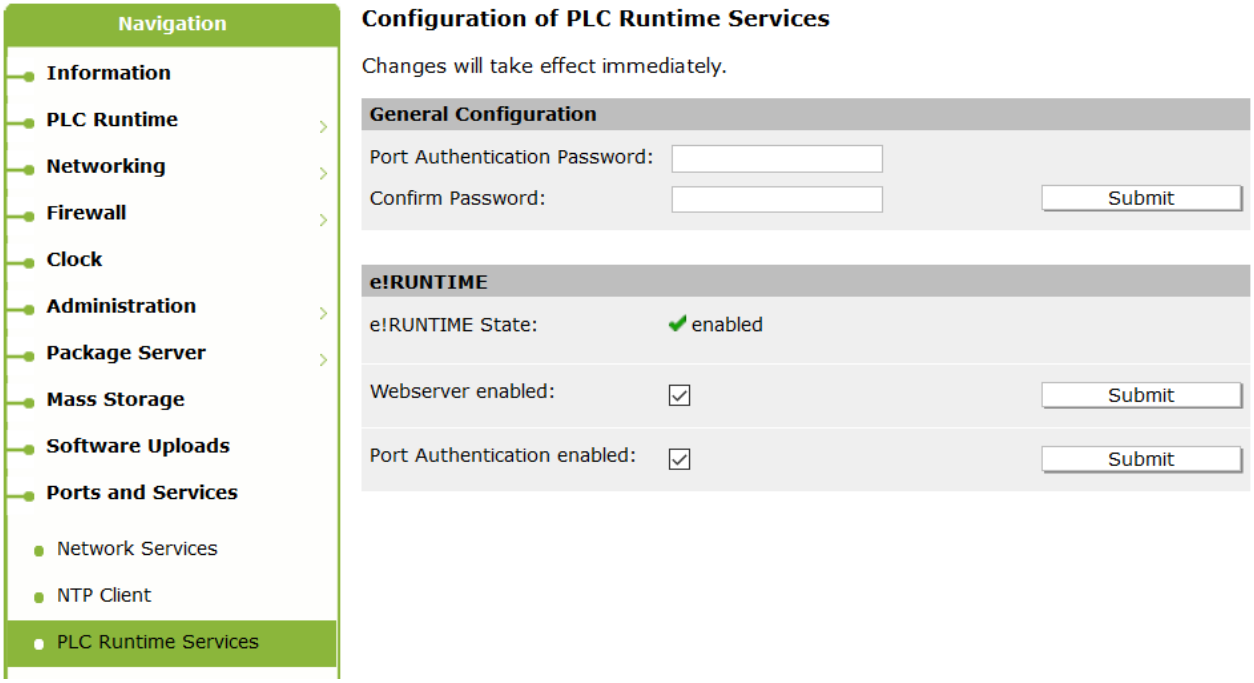


Abbildung 52: Webvisu über das WBM aktivieren

Es wird im WBM der Reiter PLC Runtime und der Punkt WebVisu aufgerufen. Hier kann anschließend „Open WebVisu in new window“ aufgerufen werden.

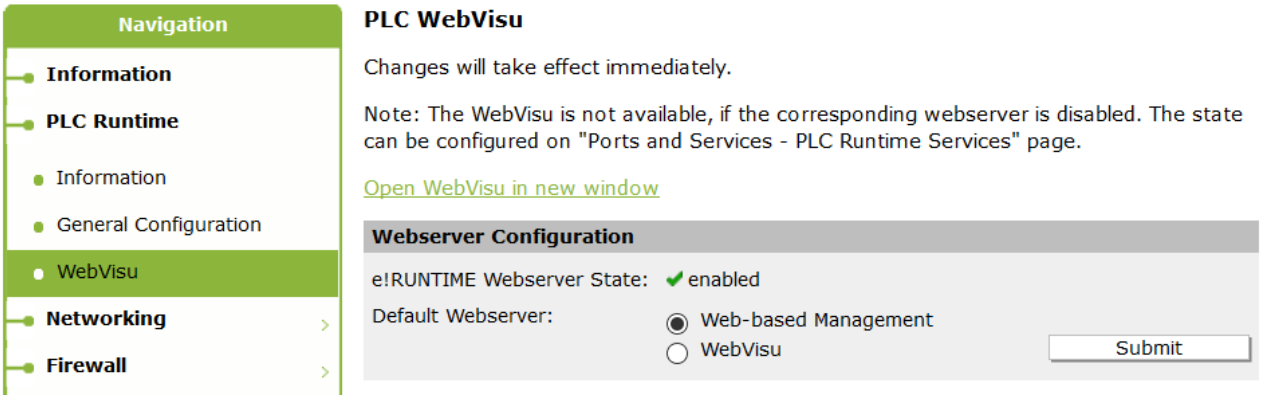


Abbildung 53: WebVisu über das WBM starten

In einem neuen Fenster öffnet sich nun die zuvor eingestellte Webvisualisierung der Wetterstation als HTML5-Seite, wie in Abbildung 54 gezeigt.

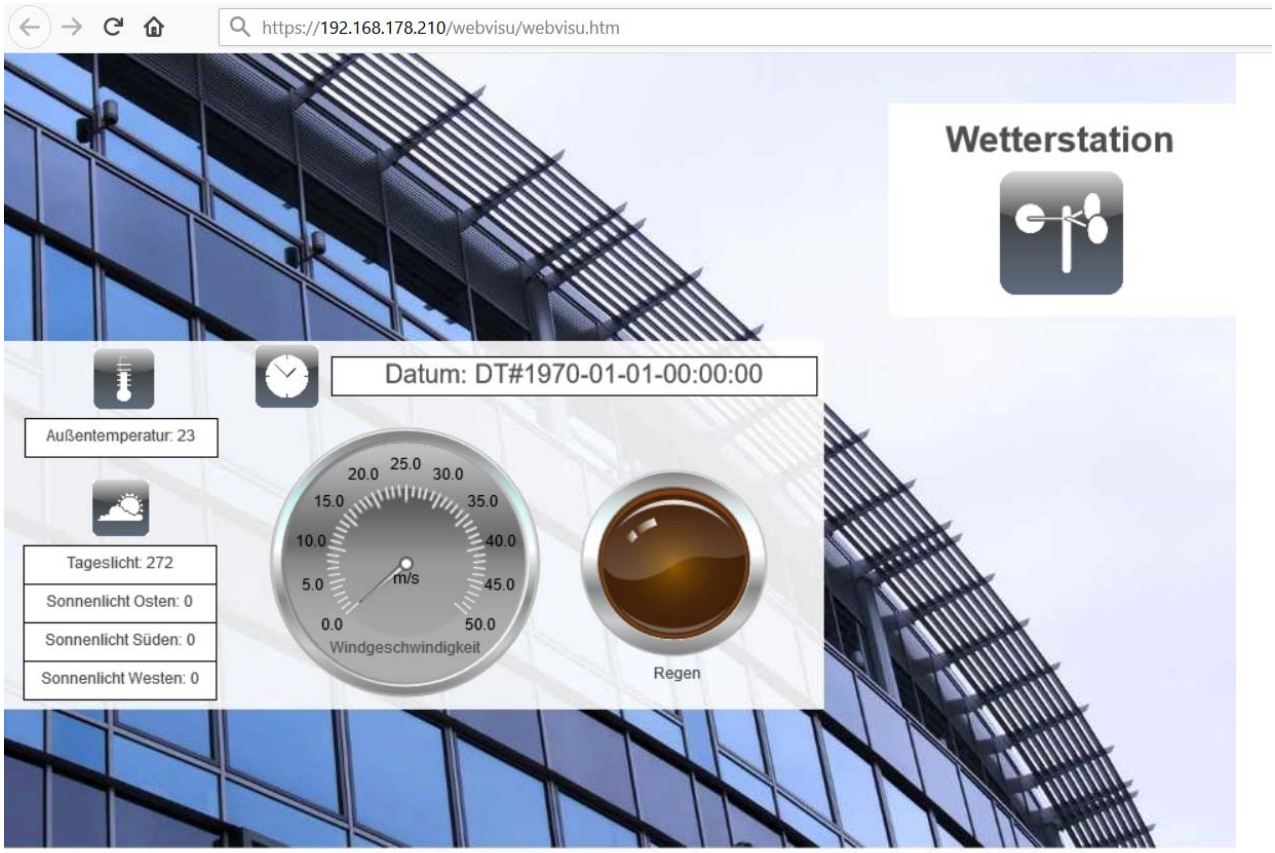


Abbildung 54: Browserdarstellung der WebVisu

7 Kommunikation mit externen Systemen

Durch Verwendung der PFC-Familie von WAGO ist es deutlich einfacher geworden Daten an eine cloudbasierte Applikation zu senden. Hierzu bietet WAGO eine eigene Cloud, aber eine Anbindung des PFC an gängige Clouds, wie Microsoft Azure, IBM Bluemix, Amazon Webservices usw. ist ebenso möglich. Hierzu finden sich weiterführende Dokumentationen im Internet. Im Nachfolgenden soll die Anbindung an die WAGO Cloud „Data Control“ gezeigt werden.

7.1 Anbindung an WAGO Cloud Data Control

Unter dem Link https://www.wago.com/de/automatisierungstechnik/wago-cloud muss sich der Nutzer registrieren und wird anschließend in einer Email der Firma WAGO über die Freischaltung benachrichtigt.

Nach dem Login unter dem Link www.cloud.wago.com wird ein neues Projekt mit dem Namen „Tutorial_Beispiel_7“ erstellt.

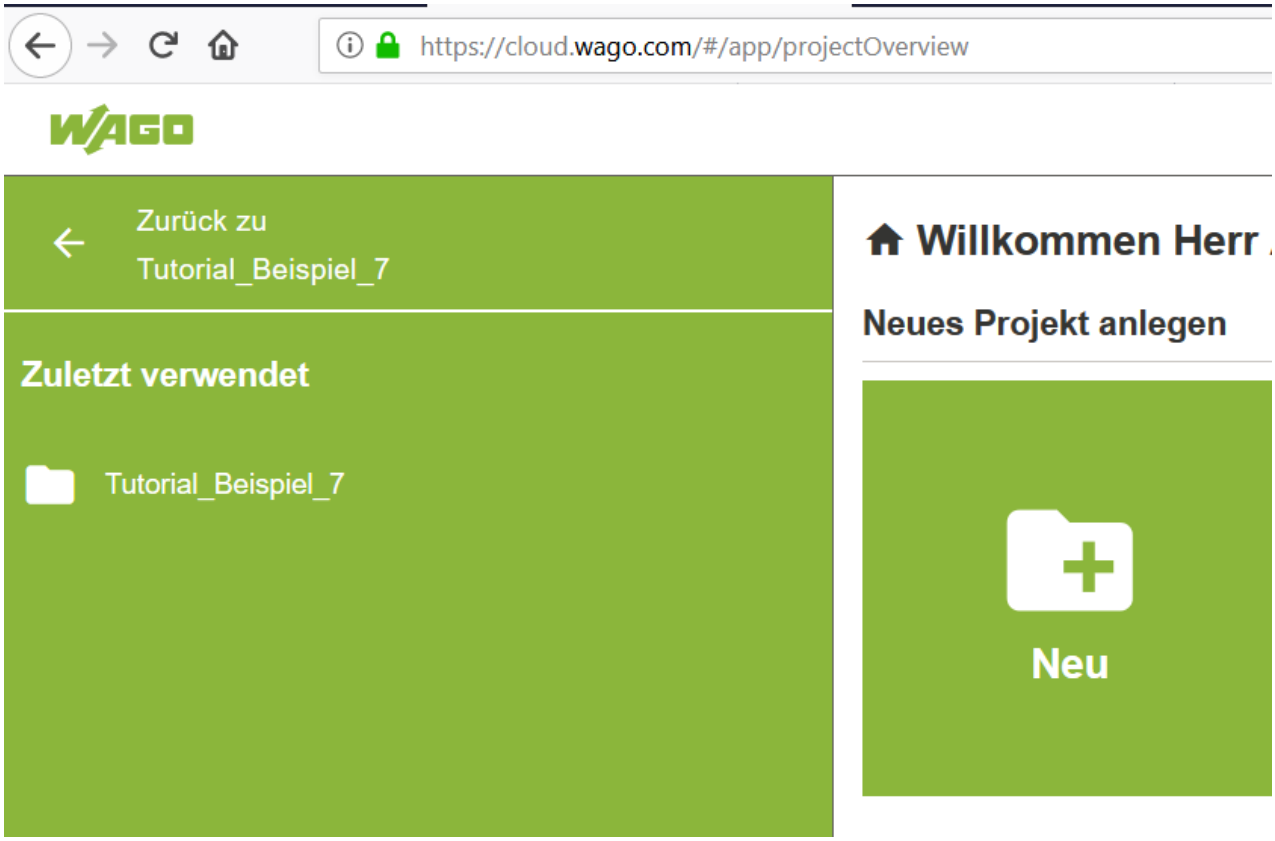


Abbildung 55: Erstellung eines neuen Projekts in der Cloud

Im Menüpunkt Steuerungen wird eine neue Steuerung mit dem Namen „PFC100“ hinzugefügt. Diese neue Steuerung erhält eine eigene Device-ID und einen eigenen Aktivierungsschlüssel und gibt den aktuellen Verbindungsstatus etc. aus.

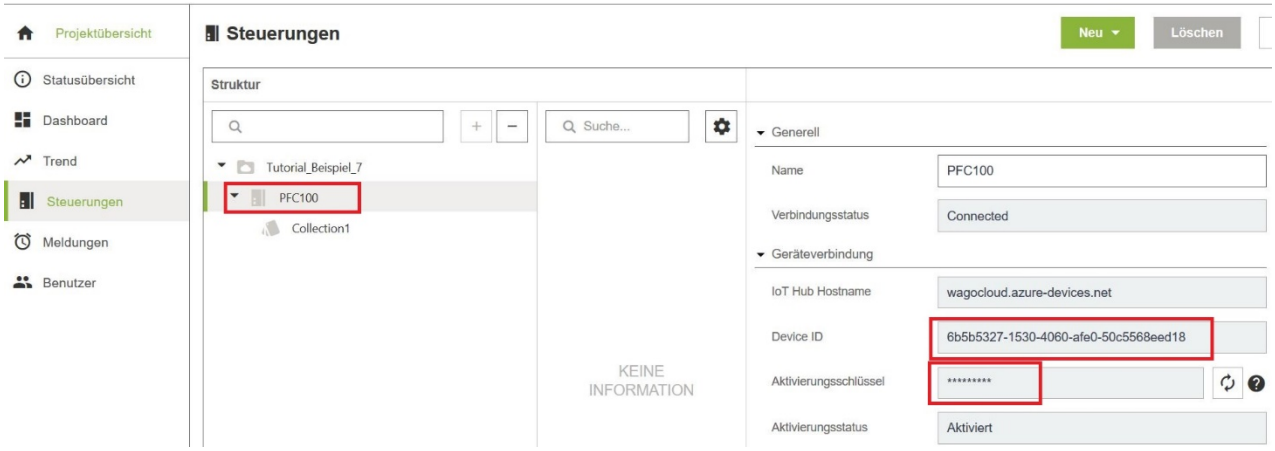


Abbildung 56: Device ID und Aktivierungsschlüssel des neu erstellten Controllers auslesen

Mit der Device-ID und dem Aktivierungsschlüssel der Cloud wird nun im WBM des Controllers der Menüpunkt Cloud Connectivity aufgerufen. Und diese Einstellungen mit der individuellen Device-ID und dem Aktivierungsschlüssel in die Steuerung geschrieben. Anschließend ist ein Neustart der Steuerung (stromlos schalten) nötig. Über den Status des WBM wird eine erfolgreiche Anbindung dargestellt. Im nächsten Schritt muss ein Datenaustausch mit der Cloud über das Programm des Controllers erfolgen.

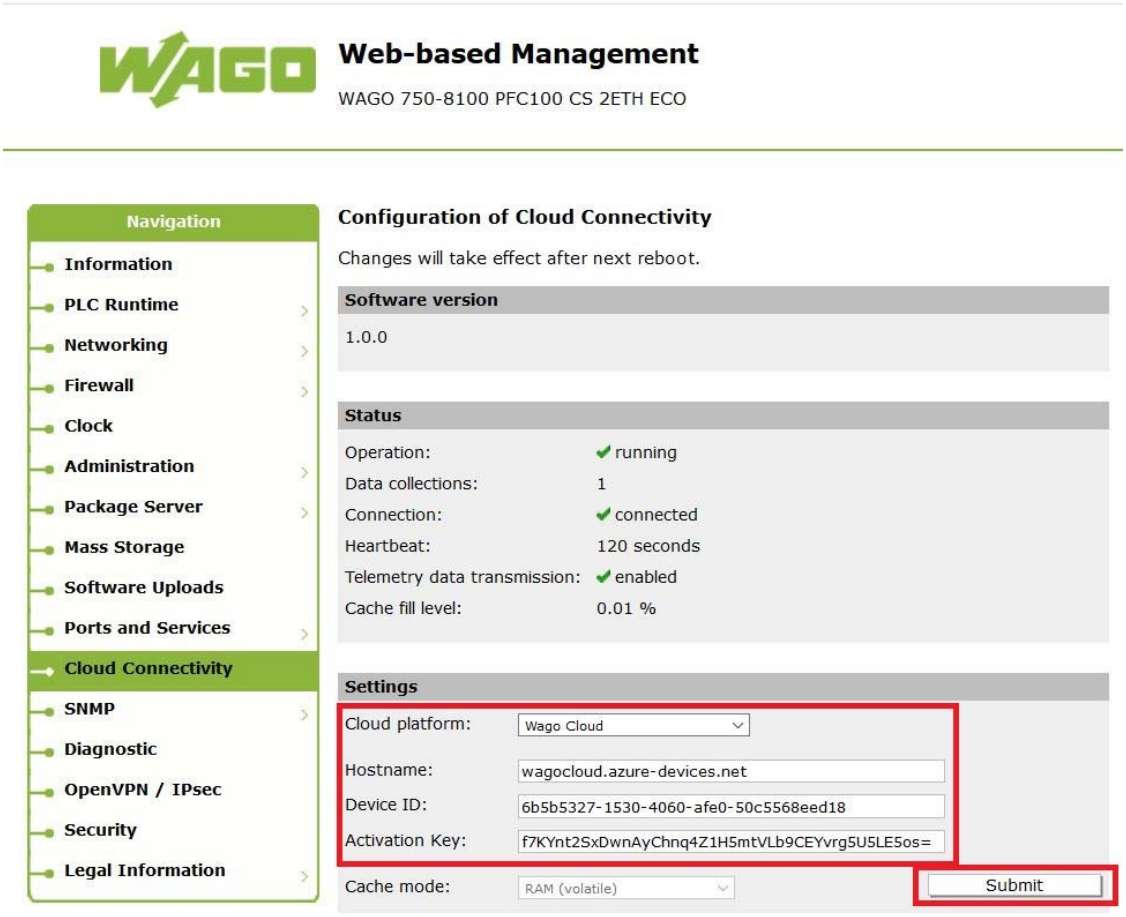


Abbildung 57: Einstellungen der Cloud Konnektivität anpassen

Hinweis: Zusätzlich zu den dargestellten Einstellungen kann es sein, dass die Realtime Clock (RTC) des Controllers nicht stimmt. Diese sollte über den Menüpunkt „Clock“ geprüft und eingestellt werden.

7.1.1 Werte an die Cloud senden

In diesem Beispiel sollen Daten an die Cloud gesendet werden. Dabei soll das Vorgehen zur Darstellung von Daten der Wetterstation in einem einfachen Dashboard in der WAGO Cloud gezeigt werden.

Um die Cloud-Funktionalität zu verwenden muss an erster Stelle die Bibliothek „WAGOAppCloud“ in der Bibliotheksverwaltung hinzugefügt werden. Anschließend wird ein neues Programm mit dem Namen „Tutorial_Beispiel_7“ in der Implementierungssprache ST erstellt und der in Abbildung 58 gezeigte Inhalt eingefügt. Der Sinn dieses Programms ist die Erstellung eines Collection-Arrays, welche die Variablen rOutsideTemperatur, rWind und xRain mit ihren entsprechenden Descriptions beinhaltet und über den Funktionsblock „FBCollectionLogger“ an die Cloud sendet.

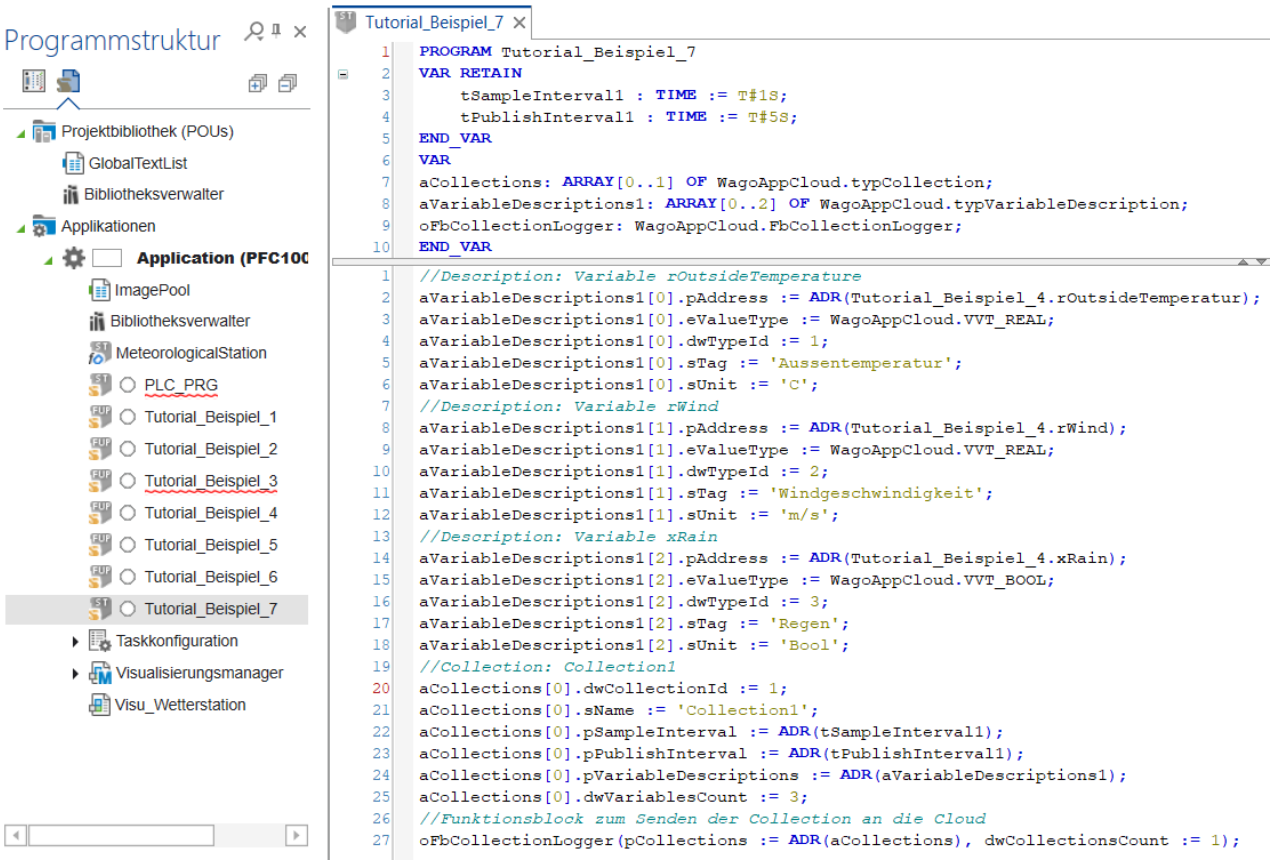


Abbildung 58: Programmcode zum Senden von Daten an die Cloud

Nachdem der Programmaufruf in PLC_PRG eingetragen wurde, kann sich mit der Steuerung verbunden und das Programm gestartet werden.

In der Cloud sollte nun die Collection der erstellten Steuerung, wie in Abbildung 59 gezeigt, hinzugefügt worden sein.

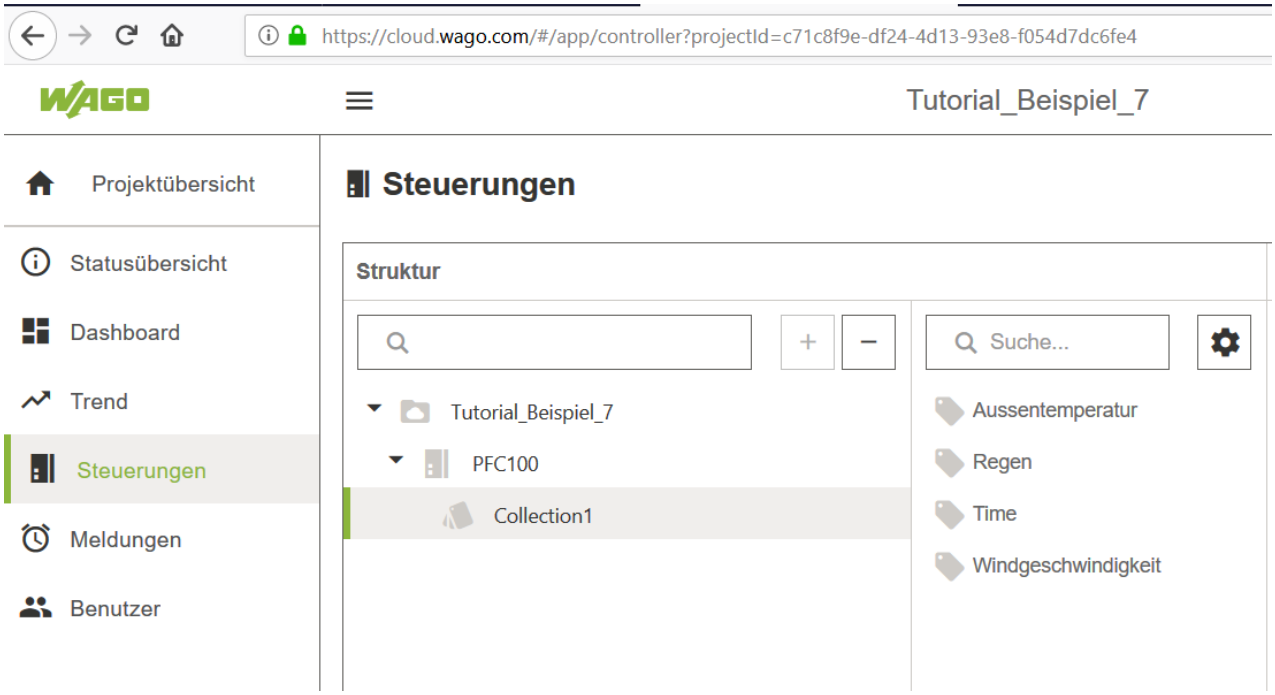


Abbildung 59: Gesendete Daten des PFC100 in der Cloud

Auf Basis dieser Collection wird über den Menüpunkt „Trend“ ein Trendverlauf der Aussentemperatur erstellt. Über den Menüpunkt „Dashboard“ wird anschließend ein neues Dashboard mit dem Namen „Tutorial_Beispiel_7“ hinzugefügt und im Editiermodus, wie in Abbildung 60 dargestellt, eine Liste mit den Variablenwerten und das Trenddiagramm eingefügt.

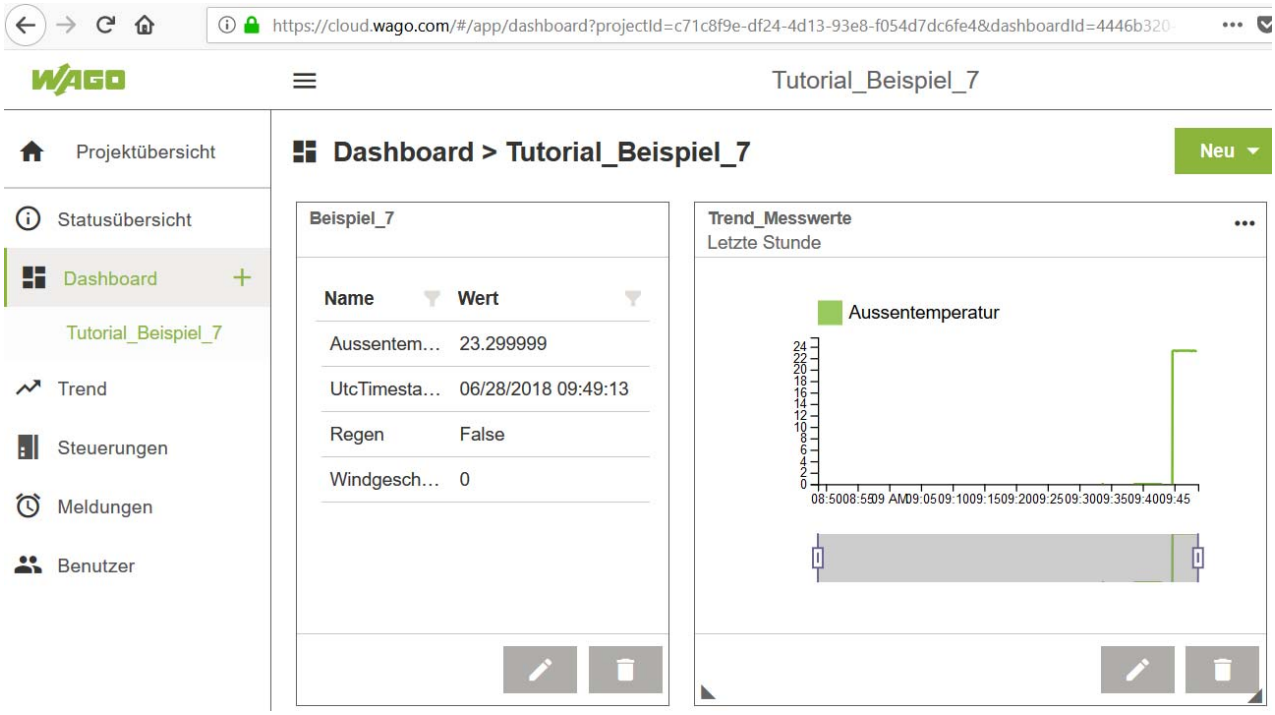


Abbildung 60: Dashboard-Visualisierung der gesendeten Daten

7.1.2 Werte aus der Cloud senden

In diesem Beispiel soll gezeigt werden, wie Daten aus der Cloud an die Steuerung gesendet werden können. Hierbei soll die Aufgabe darin liegen, über Buttons eines Dashboards in der Cloud, den SmartPlug zu steuern.

Zuerst wird ein neues Programm mit dem Namen „Tutorial_Beispiel_8“ in der Implementierungssprache ST erstellt und der Abbildung 61 bis Abbildung 63 gezeigte Programmcode eingetragen. Der Sinn dieses Programms ist das Empfangen eines Real-Werts und eines Strings über einen Command „MyCommand0“ mit dem Funktionsblock „FbCommandListener“. Das Programm sendet außerdem eine Antwort mit dem gesendeten Command mit dem Funktionsblock „FbCommandResponder“ zurück an die Cloud.

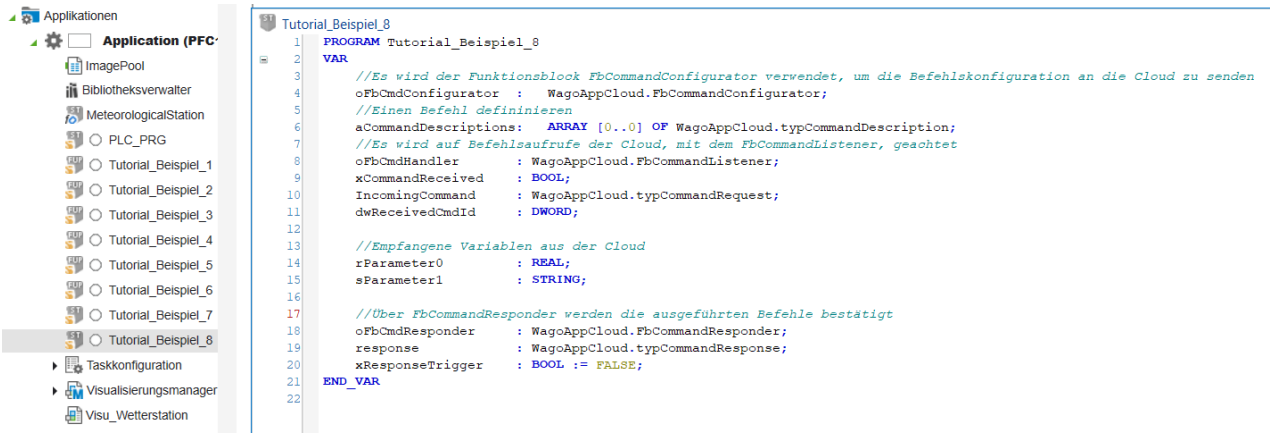


Abbildung 61: Variablendeklaration des Programms zum Empfangen von Daten aus der Cloud

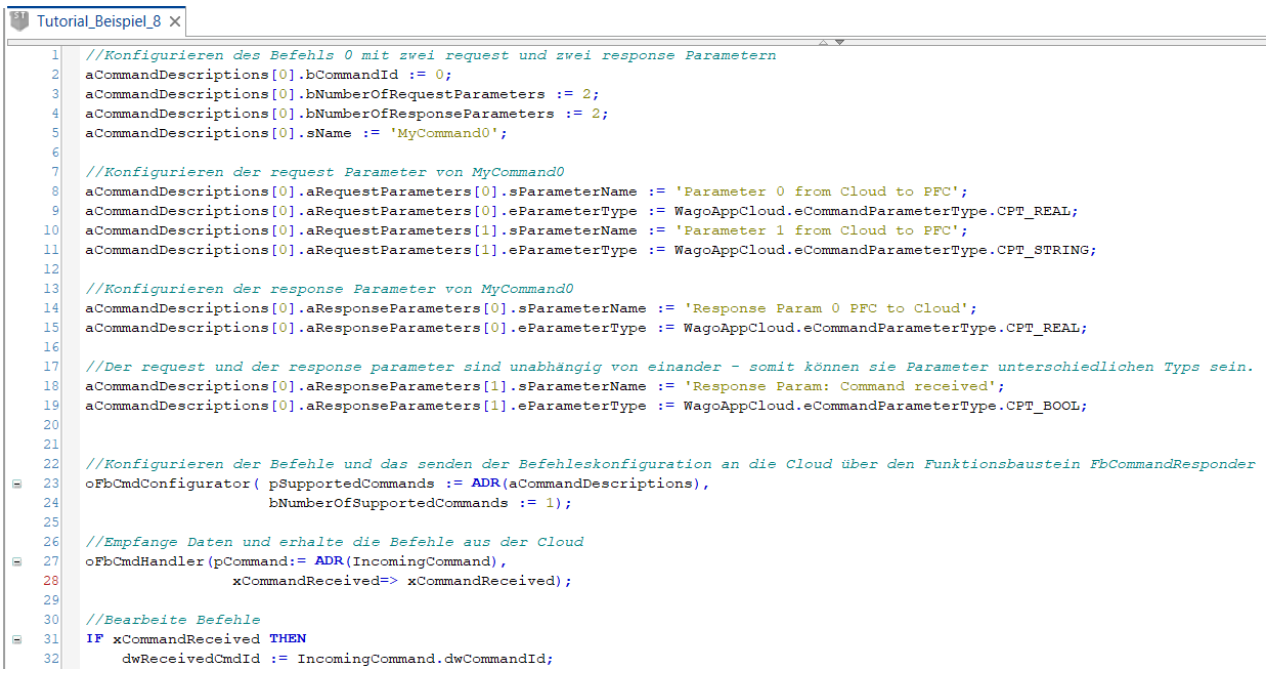


Abbildung 62: Erster Teil des Programmcodes zum Empfangen von Daten aus der Cloud

```
33
34      CASE dwReceivedCmdId OF
35          0:
36              //Erhaltene Befehle mit BefehlsID 0
37              rParameter0 := STRING_TO_REAL(IncomingCommand.aRequestParameters[0].sParameterValue);
38              sParameter1 := IncomingCommand.aRequestParameters[1].sParameterValue;
39
40              //Hier können die erhaltenen Befehle ausgeführt werden. In diesem Beispiel senden wir die erhaltenen Befehle zurück an die Cloud.
41              //Konfigurieren der response Antwort an die Cloud
42              response.dwCommandId:= dwReceivedCmdId;
43              response.dwInvokeId := IncomingCommand.dwInvokeId;
44              response.bNumberOfResponseParameters := 2;
45              response.aResponseParameters[0].eParameterType := aCommandDescriptions[0].aResponseParameters[0].eParameterType;
46              response.aResponseParameters[0].sParameterName := aCommandDescriptions[0].aResponseParameters[0].sParameterName;
47              response.aResponseParameters[0].sParameterValue := REAL_TO_STRING(rParameter0);
48
49              response.aResponseParameters[1].eParameterType := aCommandDescriptions[0].aResponseParameters[1].eParameterType;
50              response.aResponseParameters[1].sParameterName := aCommandDescriptions[0].aResponseParameters[1].sParameterName;
51              response.aResponseParameters[1].sParameterValue := BOOL_TO_STRING(TRUE);
52
53              xResponseTrigger := TRUE;
54
55          (*1:
56              // In diesem Beispiel ist nur ein Befehl mit der ID 0 konfiguriert
57          *)
58      END_CASE
59  END_IF
60
61  //Senden der Antwort an die Cloud
62  oFbCmdResponder(pCommand := ADR(response),
63                  xTrigger := xResponseTrigger);
```

Abbildung 63: Zweiter Teil des Programmcodes zum Empfangen von Daten aus der Cloud

In einem neuen Dashboard mit dem Namen „Tutorial_Beispiel_8“ werden nun ein Button zum Einschalten und ein Button zum Ausschalten des in „Tutorial_Beispiel_3“ verwendeten SmartPlugs eingefügt. Für den An-Button werden die in Abbildung 64 und Abbildung 65 gezeigten Einstellung vorgenommen, für den Aus-Button wird in die Text-Box „Steckdose_Off“ eingetragen und der Parameter 0 auf den Wert 0 gesetzt, sowie der Parameter 1 mit dem String „Deaktivieren“ beschrieben.

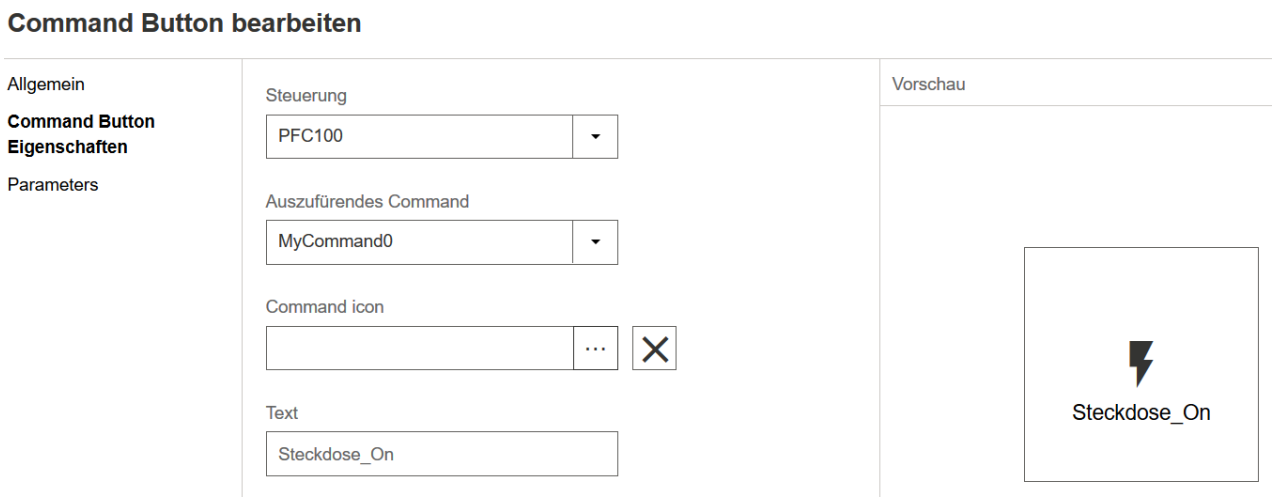


Abbildung 64: Erstellen eines neuen Buttons in der WAGO Cloud

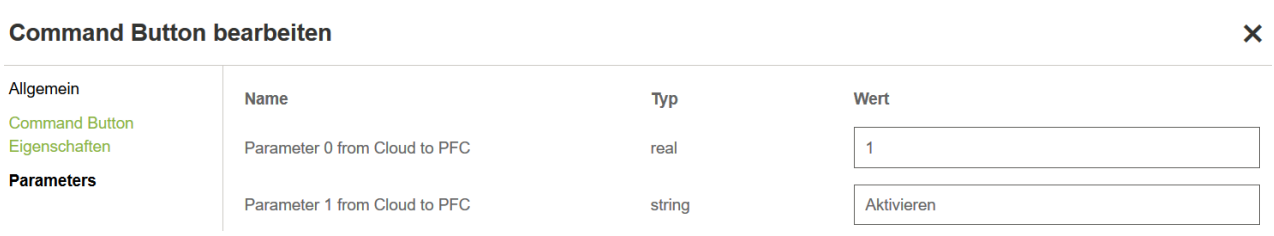


Abbildung 65: Button bearbeiten

Um den SmartPlug ansteuern zu können muss das RS-Glied von Tutorial_Beispiel_3 um die in Abbildung 66 gezeigten Variablen und Funktionsblöcke erweitert werden.

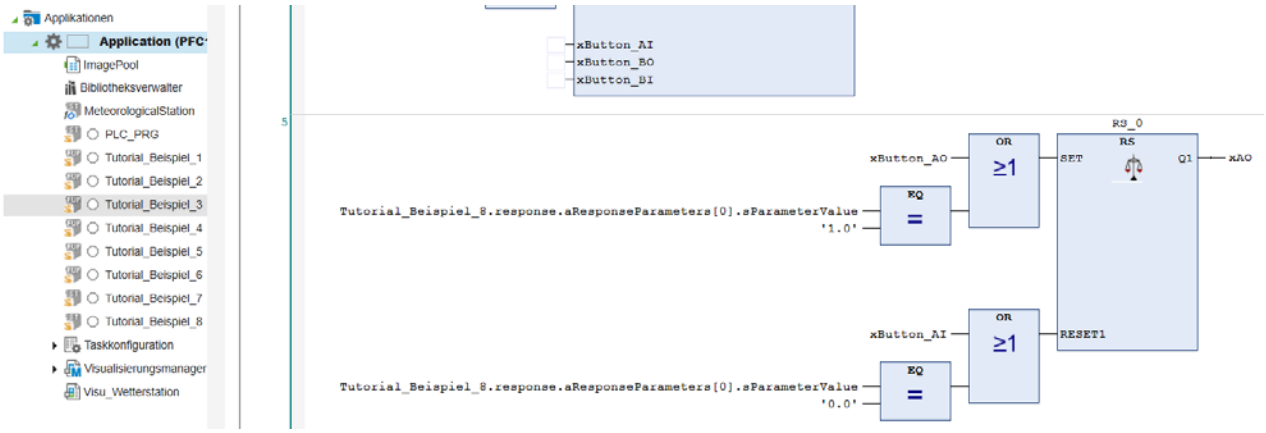


Abbildung 66: Programm von Beispiel_3 erweitern

Wie in Abbildung 67 gezeigt, vergleicht des Programm nun die an die Steuerung gesendeten Variablenwerte aus Parameter 0 des Commands mit den von den Buttons des Dashboards gesendeten Werten mit 0 oder 1 und setzt den Ausgang xAO, welcher den SmartPlug ansteuert.

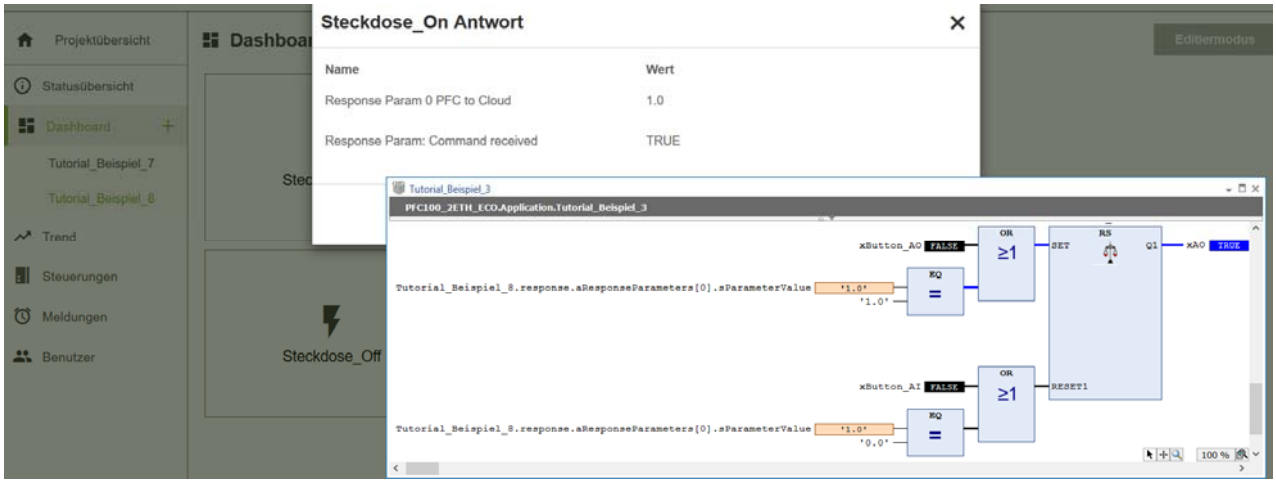


Abbildung 67: Steuerung von Aktoren über die WAGO Cloud

8 Sicherheitshinweise

- **Alle Verdrahtungsarbeiten müssen im spannungsfreien Zustand durchgeführt werden!**
    - Anschlussleitung nicht vor Inbetriebnahme einstecken
    - Sicherung – Schalter auf die „0“ – Position stellen
- Inbetriebnahme nur nach Abnahme der Aufsichtsperson (Dozent)
- An der Sicherung mit Anschlussleitung dürfen keine Veränderungen vorgenommen werden!
- An allen Tastern und Sensoren werden keine Veränderungen vorgenommen!
- An den gesamten Leuchten und Vorschaltgeräten werden keine Veränderungen vorgenommen!