

**Vergleich der Implementierung
der Sensorik und Aktorik
des KNX/EIB-Gebäudebussystems
in IEC 61131-basierte
Gebäudeautomatisierungssysteme
von WAGO und Beckhoff**

- Diplomarbeit -

Prüfer: Prof. Dr. Bernd Aschendorf

Zweitprüfer: Prof. Dr. Annette Zacharias

Christian Glesmann

Matrikel-Nr.: 7062767

Abgabetermin: 15.02.2008

Eidesstattliche Erklärung

Hiermit erkläre ich, dass die Diplomarbeit von mir selbständig verfasst wurde und nur die angegebenen Quellen und Hilfsmittel benutzt wurden.

Dortmund, den 14.02.2008

(Christian Glesmann)

Inhaltsverzeichnis

<i>Inhaltsverzeichnis</i>	<i>1</i>
<i>Abbildungsverzeichnis</i>	<i>3</i>
<i>1 Aufgabenstellung dieser Diplomarbeit</i>	<i>5</i>
<i>2 Allgemeine Vorgehensweise</i>	<i>7</i>
<i>3 Vorstellung der verwendeten Systeme</i>	<i>8</i>
3.1 <i>WAGO-I/O-System 750</i>	<i>8</i>
3.1.1 <i>WAGO Hardware Feldbus-Controller 750-841</i>	<i>8</i>
3.1.2 <i>WAGO Software WAGO-I/O-PRO CAA</i>	<i>9</i>
3.2 <i>Beckhoff Embedded-PC CX9000 mit K-Bus</i>	<i>11</i>
3.2.1 <i>Beckhoff Hardware CX9000 CPU-Grundmodul</i>	<i>11</i>
3.2.2 <i>Beckhoff Software TwinCAT</i>	<i>13</i>
3.3 <i>WAGO/Beckhoff Busklemmensystem</i>	<i>16</i>
3.4 <i>EnOcean</i>	<i>18</i>
3.5 <i>KNX/EIB</i>	<i>19</i>
3.5.1 <i>Grundlagen</i>	<i>19</i>
3.5.2 <i>Topologie</i>	<i>20</i>
3.5.3 <i>Adressierung</i>	<i>21</i>
3.5.4 <i>KNX/EIB Software ETS</i>	<i>22</i>
<i>4 Die verwendeten Komponenten</i>	<i>23</i>
4.1 <i>Die WAGO Komponenten</i>	<i>23</i>
4.2 <i>Die Beckhoff Komponenten</i>	<i>25</i>
4.3 <i>Die EnOcean Komponenten</i>	<i>27</i>
4.4 <i>Die KNX/EIB Komponenten</i>	<i>28</i>
<i>5 Herangehensweise zur Lösung der Aufgabenstellung</i>	<i>30</i>
5.1 <i>Pflichtenhefte der Grundfunktionen in einem Bürogebäude</i>	<i>30</i>
5.2 <i>Umsetzung der Funktionen mit dem WAGO Controller 750-841</i>	<i>34</i>
5.2.1 <i>Verbindungsaufbau zur WAGO-SPS</i>	<i>35</i>
5.2.2 <i>Allgemeine Programmierung der WAGO-SPS</i>	<i>37</i>
5.2.3 <i>Implementierung der EnOcean-Klemme von WAGO</i>	<i>46</i>
5.2.4 <i>Implementierung der KNX/EIB-Klemme von WAGO</i>	<i>48</i>
5.2.5 <i>Zwischenfazit zur WAGO KNX/EIB-Klemme</i>	<i>56</i>
5.3 <i>Umsetzung der Funktionen mit dem Beckhoff CX9001-1001</i>	<i>58</i>
5.3.1 <i>Verbindungsaufbau zur Beckhoff-SPS</i>	<i>58</i>
5.3.2 <i>Allgemeine Programmierung der Beckhoff-SPS</i>	<i>60</i>
5.3.3 <i>Implementierung der EnOcean-Klemme von Beckhoff</i>	<i>64</i>
5.3.4 <i>Implementierung der KNX/EIB-Klemme von Beckhoff</i>	<i>67</i>
5.3.5 <i>Zwischenfazit zur Beckhoff KNX/EIB-Klemme</i>	<i>72</i>
<i>6 Gegenüberstellung der KNX/EIB-Klemmen von WAGO und Beckhoff</i>	<i>73</i>

7	<i>Fazit</i>	75
8	<i>Möglichkeiten zur Umsetzung in großen Objekten</i>	77
	<i>Abkürzungsverzeichnis</i>	81
	<i>Danksagungen</i>	82
	<i>Literatur</i>	83
	<i>Programmcode des WAGO Controllers</i>	84
	<i>Programmcode des Beckhoff Controllers</i>	101

Abbildungsverzeichnis

<i>Abbildung 1: Feldbus-Controller 750-841</i>	<i>8</i>
<i>Abbildung 2: WAGO-I/O-PRO CAA</i>	<i>10</i>
<i>Abbildung 3: CX9000 CPU-Grundmodul</i>	<i>11</i>
<i>Abbildung 4: TwinCAT System Manager</i>	<i>14</i>
<i>Abbildung 5: TwinCAT PLC</i>	<i>15</i>
<i>Abbildung 6: KNX/EIB-Busklemmen WAGO 753-646 und Beckhoff KL6301</i>	<i>17</i>
<i>Abbildung 7: Batterielose Funktechnologie von EnOcean</i>	<i>18</i>
<i>Abbildung 8: KNX/EIB Topologie</i>	<i>20</i>
<i>Abbildung 9: ETS Professional 3.0d</i>	<i>22</i>
<i>Abbildung 10: Komponenten der WAGO-Anlage</i>	<i>23</i>
<i>Abbildung 11: Komponenten der Beckhoff-Anlage</i>	<i>25</i>
<i>Abbildung 12: EnOcean PTM 200</i>	<i>27</i>
<i>Abbildung 13: EIB-Schnittstellen</i>	<i>28</i>
<i>Abbildung 14: EIB-Schnittstellen</i>	<i>28</i>
<i>Abbildung 15: EIB-Schaltaktor</i>	<i>29</i>
<i>Abbildung 16: Siemens Raumtemperaturregler UP 231/2 DELTA profil</i>	<i>29</i>
<i>Abbildung 17: Siemens Raumtemperaturregler UP 231/2 DELTA profil</i>	<i>29</i>
<i>Abbildung 18: Funktionsanzeige des Demokoffers</i>	<i>33</i>
<i>Abbildung 19: WAGO Ethernet Settings</i>	<i>35</i>
<i>Abbildung 20: Einstellung der IP-Adresse des Controllers</i>	<i>36</i>
<i>Abbildung 21: CoDeSys „Zielsystemeinstellung“</i>	<i>37</i>
<i>Abbildung 22: CoDeSys „Neuer Baustein“</i>	<i>37</i>
<i>Abbildung 23: CoDeSys „Steuerungskonfiguration“</i>	<i>38</i>
<i>Abbildung 24: CoDeSys „I/O-Konfigurator“</i>	<i>38</i>
<i>Abbildung 25: Vergeben der Variablennamen</i>	<i>39</i>
<i>Abbildung 26: CoDeSys „Bibliotheksverwalter“</i>	<i>40</i>
<i>Abbildung 27: Übersichtlichkeit durch Ordner</i>	<i>41</i>
<i>Abbildung 28: CoDeSys „Eingabehilfe“</i>	<i>42</i>
<i>Abbildung 29: CoDeSys „Variablendeklaration“</i>	<i>42</i>
<i>Abbildung 30: Kopierter Programmbaustein</i>	<i>43</i>
<i>Abbildung 31: CoDeSys Hauptprogramm</i>	<i>44</i>
<i>Abbildung 32: CoDeSys „Kommunikationsparameter“</i>	<i>45</i>
<i>Abbildung 33: WAGO FbEnOceanReceive-Baustein</i>	<i>46</i>
<i>Abbildung 34: WAGO EnOcean Baustein 4-Kanal</i>	<i>46</i>
<i>Abbildung 35: WAGO FbKNX_Master_646-Baustein</i>	<i>48</i>
<i>Abbildung 36: WAGO KNX-DPT_Bool-Baustein</i>	<i>49</i>
<i>Abbildung 37: WAGO Funktionsbaustein KNX Jalousieaktor</i>	<i>50</i>
<i>Abbildung 38: WAGO FbSunblind_KNX im Bibliotheksverwalter</i>	<i>51</i>

<i>Abbildung 39: Abfolge zur Inbetriebnahme der WAGO KNX/EIB-Klemme</i>	<i>52</i>
<i>Abbildung 40 Erzeugen der Symboltabelle</i>	<i>53</i>
<i>Abbildung 41: Auswahl der SYM-XML Datei</i>	<i>54</i>
<i>Abbildung 42: WAGO-ETS3-PlugIn</i>	<i>55</i>
<i>Abbildung 43: TwinCAT System Manager „Add Route Dialog“</i>	<i>58</i>
<i>Abbildung 44: Einstellung der IP-Adresse des CX</i>	<i>59</i>
<i>Abbildung 45: Deklaration einer Eingangsvariable bei der Beckhoff-SPS</i>	<i>60</i>
<i>Abbildung 46: Auslesen der Klemmenkonfiguration in TwinCAT</i>	<i>61</i>
<i>Abbildung 47: Einstellen der Klemmen in TwinCAT</i>	<i>62</i>
<i>Abbildung 48: Variablenzuweisung im „TwinCat System Manager“</i>	<i>63</i>
<i>Abbildung 49: Beckhoff FB_EnOceanReceive-Baustein</i>	<i>64</i>
<i>Abbildung 50: Beckhoff FB_EnOceanPTM200</i>	<i>64</i>
<i>Abbildung 51: Verknüpfung der Variablen der KL6021-0023</i>	<i>66</i>
<i>Abbildung 52: Beckhoff Funktionsbaustein „KL6301“</i>	<i>67</i>
<i>Abbildung 53: Filtereinträge für die KL6301</i>	<i>68</i>
<i>Abbildung 54: Beckhoff 1-Bit Empfangs- und Sendebaustein</i>	<i>69</i>
<i>Abbildung 55: Verknüpfung der Variablen der KL6301</i>	<i>71</i>
<i>Abbildung 56: Konzept „SPS mit Buskopplern“</i>	<i>77</i>
<i>Abbildung 57: Konzept „Verteilte SPS-Controller“</i>	<i>79</i>
<i>Abbildung 58: Umsetzungsmöglichkeiten in großen Objekten</i>	<i>80</i>

1. Aufgabenstellung dieser Diplomarbeit

Die Anforderungen an die Gebäudeautomation haben sich in den letzten Jahren stark verändert und fordern eine deutlich höhere Komplexität der Anlagen. Eine Steuerung und Überwachung dieser Anlagen über das Ethernet und/oder das Internet sind mittlerweile zum Standard geworden.

Um die Datenflut großer Anlagen komfortabel verarbeiten zu können, werden Automatisierungssysteme benötigt, die sehr leistungsfähig und außerdem einfach zu programmieren und zu warten sind.

In der Gebäudeautomation existieren viele verschiedene Bussysteme, die aber meist nach einem eigenen Standard arbeiten.

Da diese Bussysteme jedoch inkompatibel zueinander sind, aber alle ihre Vor- und Nachteile in der Funktionalität, dem Installations- und Programmieraufwand, der Geschwindigkeit, der Visualisierung oder einfach nur in den Kosten haben, ist es nicht möglich, die Nachteile eines Systems durch die Vorteile eines anderen Systems zu kompensieren.

Einen wichtigen gemeinsamen Nachteil haben aber alle auf dem Markt existierenden Bussysteme. Sie stellen alle keine zufrieden stellenden Automatisierungsmöglichkeiten zur Verfügung.

Die Firmen WAGO und Beckhoff liefern seit einiger Zeit Busklemmen für Ihre Gebäudeautomatisierungssysteme, die es ermöglichen, eines der am weitesten verbreiteten Bussysteme, den KNX/EIB, einzubinden und zu automatisieren.

Bei den Gebäudeautomatisierungssystemen von WAGO und Beckhoff handelt es sich um leistungsfähige nach IEC 61131-3 programmierbare Speicherprogrammierbare Steuerungen (SPS) mit Ethernet-Anschluss.

Durch die hohe Anzahl der für den KNX/EIB verfügbaren Sensoren und Aktoren, sowie der leistungsfähigen und beinahe unbegrenzten Automatisierungsmöglichkeiten einer SPS mit schneller Ethernet-Anbindung, sind die Grundlagen für eine komfortable und leistungsfähige Gebäudeautomation gegeben.

Die KNX/EIB-Busklemmen der Gebäudeautomatisierungssysteme haben zwar dieselbe Bauform, sind aber trotzdem nicht mit dem System des jeweils anderen Herstellers verwendbar, da diese mit unterschiedlichen Konzepten genutzt werden.

Um die Unterschiede dieser Konzepte zu verdeutlichen, ergibt sich die folgende Aufgabenstellung:

***Vergleich der Implementierung der Sensorik und Aktorik des KNX/EIB-
Gebäudebussystems in IEC 61131-basierte Gebäudeautomatisierungssysteme von
WAGO und Beckhoff***

2. Allgemeine Vorgehensweise

Um die Implementierung der Sensorik und Aktorik des KNX/EIB über die EIB-Busklemmen in die Gebäudeautomatisierungssysteme von WAGO und Beckhoff zu vergleichen, wurden eine SPS von WAGO und eine SPS von Beckhoff mit verschiedenen Ein- und Ausgangsklemmen und je einer EIB-Busklemme in einen Musterkoffer eingebaut.

In diesem Musterkoffer werden die Grundfunktionen eines Bürogebäudes nachgebildet, bestehend aus zwei Büroräumen, die sich durch Öffnen einer Trennwand zu einem Konferenzraum verbinden lassen, und einem davor liegenden Flur.

Um die Anlage von WAGO zu programmieren wurde die Software „CoDeSys V2.3“ installiert und mit den von WAGO zur Verfügung stehenden Gebäude- und KNX/EIB-Bibliotheken versehen. Für die Programmierung der Anlage von Beckhoff wurde die „TwinCat v2.10.0“ installiert und ebenfalls mit den zur Verfügung stehenden Gebäude- und KNX/EIB-Bibliotheken versehen.

An die KNX/EIB-Klemmen wurden je eine KNX/EIB-Linie, bestehend aus Spannungsversorgung, einem Schaltaktor und zwei Tastern mit eingebautem Raumtemperaturregler, angebunden.

3. Vorstellung der verwendeten Systeme

3.1 WAGO-I/O-System 750

3.1.1 WAGO Hardware Feldbus-Controller 750-841

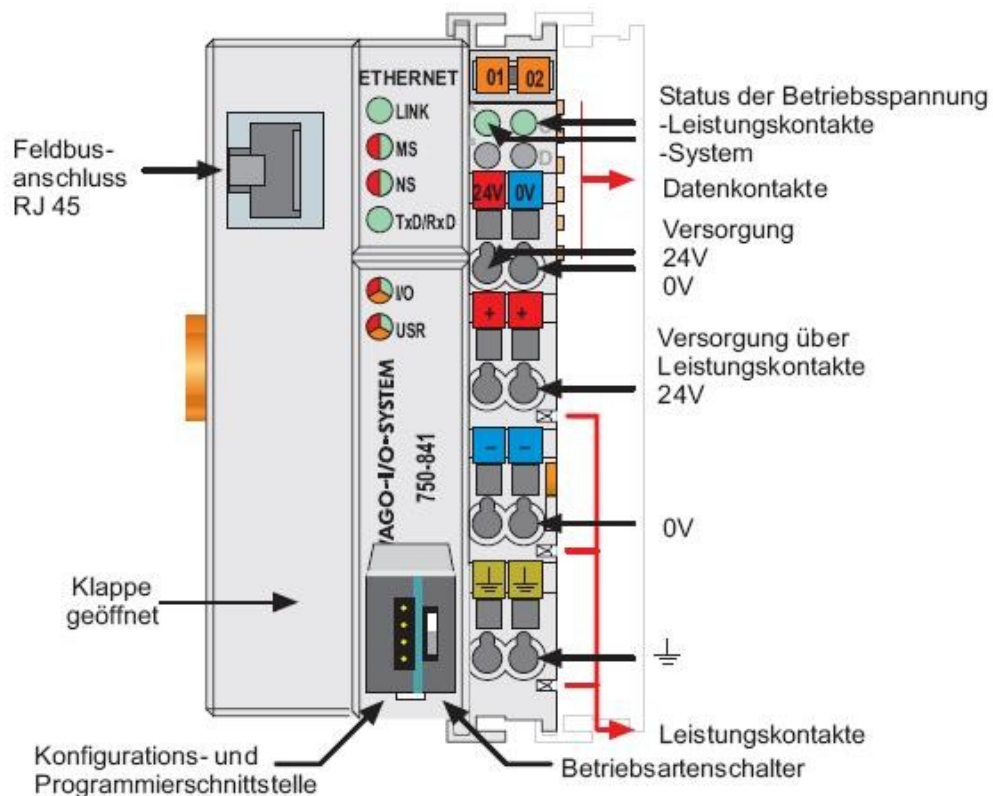


Abbildung 1: Feldbus-Controller 750-841

Der programmierbare Feldbus-Controller 750-841 kombiniert die Funktionalität eines Kopplers zur Anschaltung an den Feldbus ETHERNET mit der einer Speicherprogrammierbaren Steuerung (SPS). Der Controller basiert auf einer 32-Bit CPU und ist Multitasking fähig, d.h. mehrere Programme können quasi gleichzeitig ausgeführt werden.

In dem Controller werden sämtliche Eingangssignale der Sensoren zusammengeführt. Nach Anschluss des ETHERNET TCP/IP Feldbus-Controllers ermittelt der Controller alle in dem Knoten gesteckten I/O-Klemmen und erstellt daraus ein lokales

Prozessabbild. Hierbei kann es sich um eine gemischte Anordnung von analogen und digitalen Klemmen handeln. Das lokale Prozessabbild wird in einen Eingangs- und Ausgangsdatenbereich unterteilt.

Entsprechend der IEC 61131-3-Programmierung erfolgt die Bearbeitung der Prozessdaten vor Ort im Controller. Die daraus erzeugten Verknüpfungsergebnisse können direkt an die Aktoren ausgegeben oder über den Bus an die übergeordnete Steuerung übertragen werden. Wahlweise kann der ETHERNET-Controller 10/100 MBit/s über Ethernet „100BaseTX“ oder „10BaseT“ mit übergeordneten Systemen kommunizieren.

Für die IEC 61131-3 Programmierung stellt der Feldbus-Controller 750-841 512 KB Programmspeicher, 256 KB Datenspeicher und 24 KB Retainspeicher zur Verfügung. Der Anwender hat Zugriff auf alle Feldbus- und E/A-Daten.

3.1.2 WAGO Software WAGO-I/O-PRO CAA

Die Erstellung des Applikationsprogramms erfolgt mit WAGO-I/O-PRO CAA gemäß IEC 61131-3, wobei die Basis von WAGO-I/O-PRO CAA das Standard-Programmiersystem CoDeSys der Firma 3S ist, das mit den Target-Dateien für alle WAGO Controller spezifisch erweitert wurde.

Es werden die Programmiersprachen AWL, KOP, FUP, CFC, ST und AS unterstützt. Es kann somit für jeden Einsatzfall die optimale Programmiersprache gewählt werden.

Die steigenden Anforderungen in der Entwicklung von Steuerungsprogrammen, wie z.B. Wiederverwendbarkeit und Modularisierung, werden von der Software mit umfangreichen Programmierfunktionen wie z.B.

- Leistungsstarke Übersetzung zwischen den Programmiersprachen
- Automatische Deklaration der Variablen
- Bibliotheksverwaltung
- Steuerungskonfiguration

unterstützt.

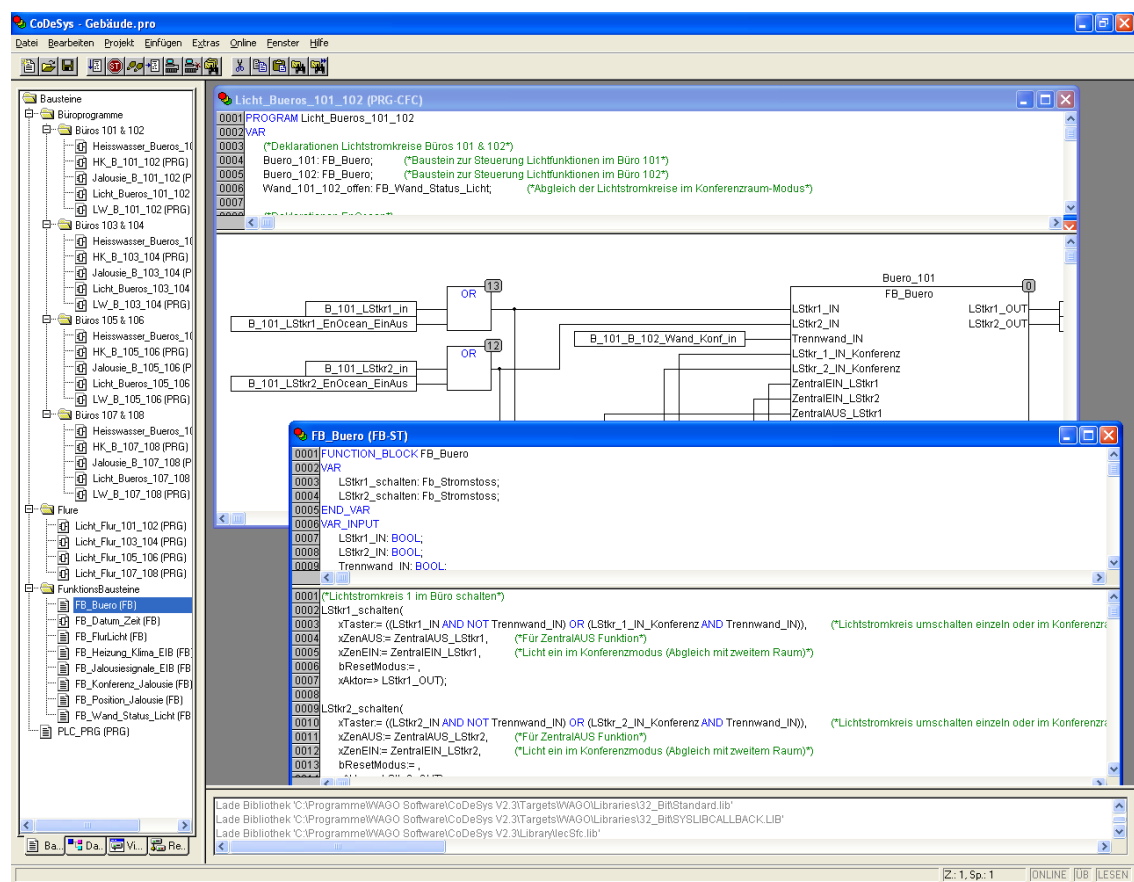


Abbildung 2: WAGO-I/O-PRO CAA

3.2 Beckhoff Embedded-PC CX9000 mit K-Bus

3.2.1 Beckhoff Hardware CX9000 CPU-Grundmodul

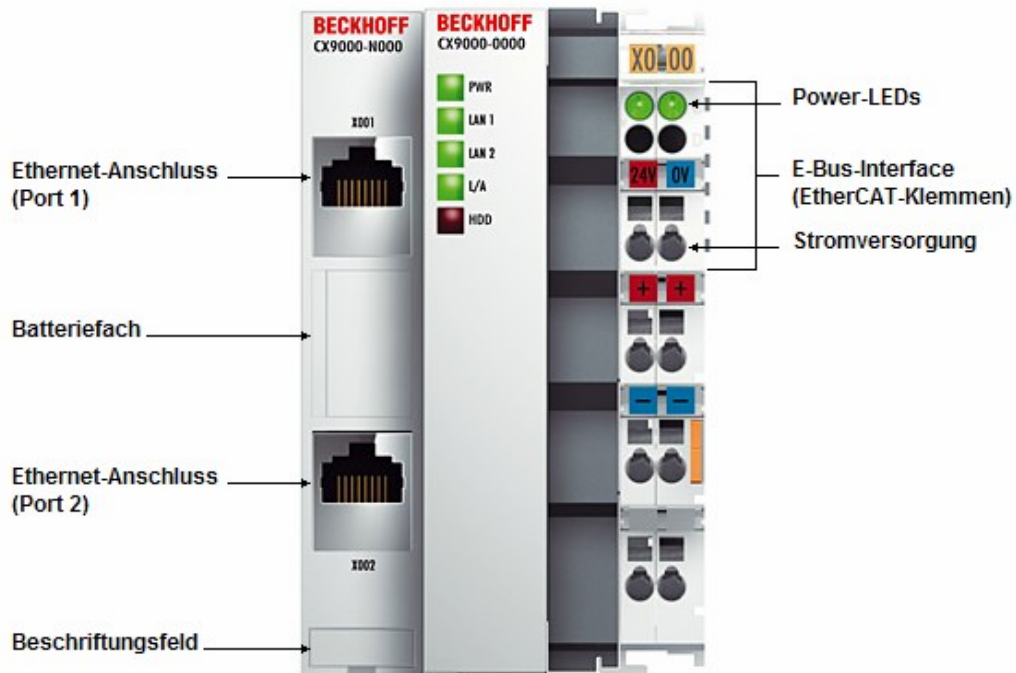


Abbildung 3: CX9000 CPU-Grundmodul

Der CX9000 ist eine kompakte, hutschienen-montierbare Ethernet-Steuerung mit direktem Anschluss an die Beckhoff-I/O-Systeme in IP 20. Der CX9000 umfasst die CPU, den internen Flash-Speicher in zwei Ausbaustufen, sowie den Arbeitsspeicher (RAM) in zwei wählbaren Größen und NOVRAM als nicht-flüchtigen Speicher. Zwei Ethernet-RJ-45-Schnittstellen gehören ebenfalls zur Basisausstattung. Diese Schnittstellen sind auf einen internen Switch geführt und bieten eine einfache Möglichkeit zum Aufbau einer Linientopologie ohne den zusätzlichen Einsatz von Ethernet-Switchen.

Der CX9000 ist mit einem stromsparenden Intel® IXP420 mit XScale® Technologie und 266 MHz ausgestattet.

Damit steht ausreichend Rechenleistung bereit, um auch komplexe Automatisierungsaufgaben zu übernehmen.

Für den Einsatz der CX9000-Familie sind keine externen Speichermedien erforderlich, das Gerät bootet das Betriebssystem aus dem internen Flash. Durch die niedrige Leistungsaufnahme wird im spezifizierten Betriebsbereich kein Lüfter benötigt. Die Embedded-PCs CX9000 kommen also ohne rotierende Bauteile aus.

Der mechanische Aufbau des Gerätes ist, wie bei der CX-Serie üblich, modular gestaltet und verfügt in der Grundausstattung über eine sehr kompakte Bauform von lediglich 47 x 100 x 91 mm.

Als optionales Modul steht ein Speichermedium im Compact-Flash-Format I und II zur Verfügung. Das Betriebssystem ist Microsoft Windows CE. Durch die Automatisierungssoftware TwinCAT wird das CX9000-System zu einer leistungsfähigen SPS und Motion-Control-Steuerung, die mit oder ohne Visualisierung eingesetzt werden kann. Weitere Systemschnittstellen können ab Werk an das CPU-Modul angeschlossen werden. Die Option CX9000-N010 bietet über die DVI- und USB-Schnittstellen den Anschluss an Beckhoff Control Panel oder marktübliche Monitore mit DVI- oder VGA-Eingang. An die USB-Schnittstellen vom Typ USB 2.0 können Geräte wie Drucker, Scanner, Maus, Tastatur, Massenspeicher, CD-RW usw., angeschlossen werden. Insgesamt zwei serielle RS232-Schnittstellen mit max. 115-kBaud-Übertragungsgeschwindigkeit bietet das Modul CX9000-N030. Diese zwei Schnittstellen können auch als RS422/RS485 ausgeführt werden; die Bezeichnung lautet dann CX9000-N031.

3.2.2 Beckhoff Software TwinCAT

Die Programmierung des CX9000/CX9010 als Automatisierungsgerät erfolgt, wie bei allen Beckhoff-Steuerungen, mit TwinCAT; auf dem Gerät selbst befindet sich die Laufzeitumgebung für SPS und Motion Control. Als Programmierschnittstelle dient eine der beiden Ethernet-Schnittstellen.

Microsoft Windows CE ermöglicht die Erstellung vollgrafischer Benutzerprogramme, die dank des im CX9000 integrierten Grafikchips auch höheren Ansprüchen genügen. Damit entsteht eine kompakte Ethernet-Steuerung, die in der Kombination mit EtherCAT-Klemmen kurze I/O-Zykluszeiten ermöglicht und mit Windows CE und TwinCAT eine leistungsfähige Softwareausstattung aufweist.

TwinCAT System Manager

Der TwinCAT System Manager ist die Konfigurationszentrale des Systems: Die Anzahl und Programme der SPS-Systeme, die Konfiguration der Achsregelung und die angeschlossenen E/A-Kanäle werden miteinander in Beziehung gebracht.

Der System Manager

- verbindet E/A-Geräte und Tasks variablenorientiert
- verbindet Tasks zu Tasks variablenorientiert
- unterstützt synchrone oder asynchrone Beziehungen
- ermöglicht den Austausch konsistenter Datenbereiche und Prozessabbilder

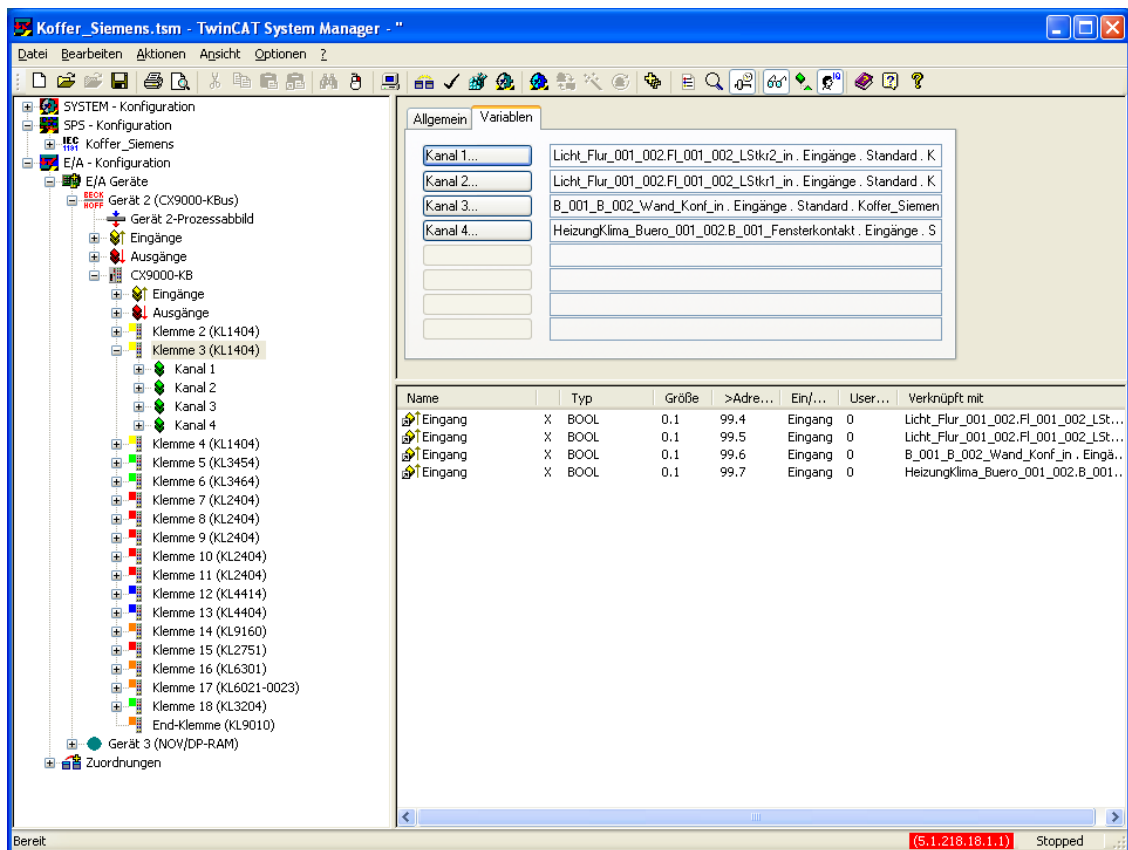


Abbildung 4: TwinCAT System Manager

TwinCAT PLC

Das SPS-System TwinCAT PLC wird herstellerunabhängig nach IEC 61131-3 programmiert. Online-Verbindung mit SPS-Laufzeitsystemen sind weltweit über TCP/IP oder über Feldbus auf dem IPC realisierbar.

TwinCAT PLC bietet alle definierten Sprachen der IEC 61131-3-Norm sowie eine leistungsfähige Entwicklungsumgebung für Programme, deren Codegröße und Datenbereiche weit über die Möglichkeiten herkömmlicher SPS-Systeme hinausgehen.

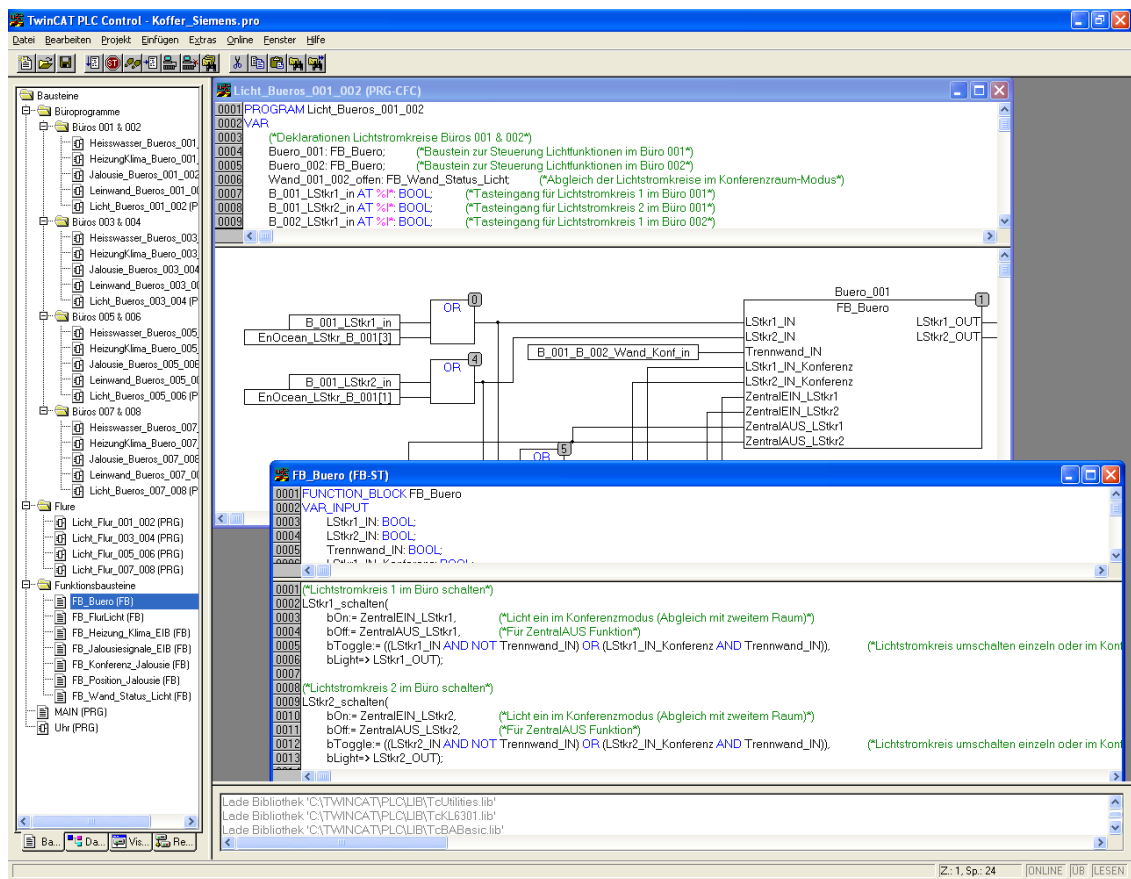


Abbildung 5: TwinCAT PLC

3.3 WAGO/Beckhoff Busklemmensystem

Die WAGO/Beckhoff Busklemme ist ein offenes und feldbusneutrales I/O-System, bestehend aus elektronischen Reihenklemmen.

Eine Busklemmenstation besteht aus einem Buskoppler und bis zu 64 elektronischen Reihenklemmen. Mit dem System der Klemmenbusverlängerung ist ein maximaler Ausbau von bis zu 255 räumlich verteilten Busklemmen bei Beckhoff und von bis zu 250 räumlich verteilten Busklemmen bei WAGO möglich.

An den Buskoppler werden die elektronischen Reihenklemmen angesteckt. Die Kontaktierung erfolgt beim Einrasten, ohne einen weiteren Handgriff. Dabei bleibt jede elektronische Reihenklemme einzeln austauschbar und lässt sich auf eine Normprofilschiene aufsetzen. Neben dem waagerechten Einbau sind alle anderen Einbauarten erlaubt.

Die Busklemmen passen sich mit ihrer Außenkontur technisch perfekt den Abmessungen von Klemmenkästen an. Eine übersichtliche Anschlussfront mit Leuchtdioden für die Statusanzeige, einsteckbare Kontaktbeschriftung und herausziehbare Beschriftungsfelder sorgen für Klarheit vor Ort. Die Dreileitertechnik, ergänzt durch einen Schutzleiteranschluss, ermöglicht die direkte Sensor-/Aktorverdrahtung.

Mit über 200 verschiedenen Busklemmen bei Beckhoff und über 100 verschiedenen Busklemmen bei WAGO ist das I/O-System sehr umfangreich. Die Komponenten erlauben dem Anwender, frei mischbare Signalzusammenstellungen je Station zu betreiben. Damit kann ein einziger dezentraler Ein-/Ausgangsknoten alle benötigten Signale abbilden.

Für alle in der Automatisierungswelt vorkommenden digitalen und analogen Signalformen stehen entsprechende Busklemmen bereit: für Ströme und Spannungen mit standardisierten Signalpegeln sowie für PT100- und Thermoelementsignale. Über

Busklemmen mit serieller Schnittstelle, gemäß RS232, RS485 oder 20 mA TTY, lassen sich intelligente Geräte anschließen.

Des Weiteren stehen Busklemmen für den EIB, LON, DALI und den MP-Bus zur Verfügung. Es ist somit möglich die am stärksten verbreiteten

Kommunikationsstandards in der Gebäudeautomatisierung als Subsystem zu integrieren.

Die feine Granularität der Busklemmen ermöglicht außerdem die bitgenaue Zusammenstellung der benötigten I/O-Kanäle. Die digitalen Busklemmen sind als 2-, 4-, und 8-Kanal-Klemmen ausgeführt. Die analogen Standardsignale $\pm 10\text{ V}$, $0 \dots 10\text{ V}$, $0 \dots 20\text{ mA}$ und $4 \dots 20\text{ mA}$ sind durchgängig als 1-, 2-, 4-, und 8-Kanal-Variante in einem Standardgehäuse erhältlich. Bei der 8-Kanal-Variante sind in einem Standardbusklemmengehäuse, auf einer Breite von nur 12 mm, analoge Ein- und Ausgangssignale ultrakompakt komprimiert. Das System ist damit hochmodular und kann, bis auf einen Kanal genau, kostengünstig projektiert werden.

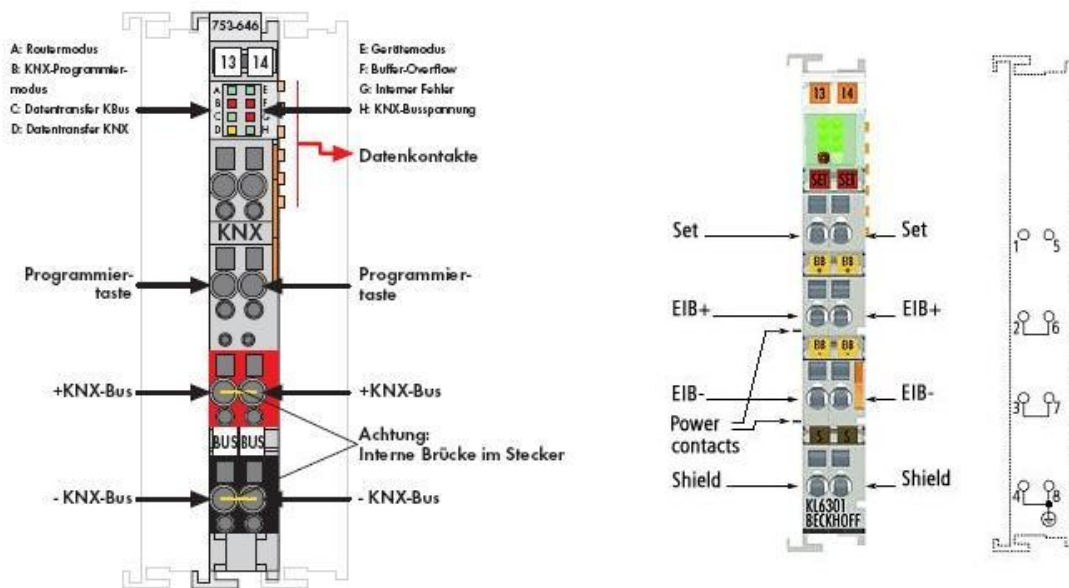


Abbildung 6: KNX/EIB-Busklemmen WAGO 753-646 und Beckhoff KL6301

Die Busklemmensysteme von WAGO und Beckhoff sind trotz der gleichen Bauform, aufgrund technischer Unterschiede der Klemmen, nicht untereinander kompatibel.

3.4 EnOcean

Die Grundidee für die innovative Technologie beruht auf einer einfachen Beobachtung: Dort, wo Sensoren Messwerte erfassen, ändert sich auch immer der Energiezustand. Ein Schalter wird gedrückt, die Temperatur ändert sich oder die Beleuchtungsstärke variiert. In diesen Vorgängen steckt genug Energie, um Funksignale über eine Entfernung von bis zu 300 Metern zu übertragen.

Die Stärke von EnOcean liegt in der Vereinigung dreier Schlüsseltechnologien:

- *Energiewandler*
Das Gewinnen von Energie aus der Umgebung (Energy Harvesting).
- *Energiemanagement & Funktechnologie*
Der Einsatz optimaler Techniken für das Energiemanagement, die gewährleisten, dass unsere Geräte mit winzigsten Energiemengen funktionieren, um Sensorinformationen per Funk zu übertragen.
- *Intelligente Software-Stacks*
Der Einsatz von intelligenten Software-Stacks, die eine modulare, vielseitige und benutzerfreundliche Integration in den Kundenapplikationen erlauben.

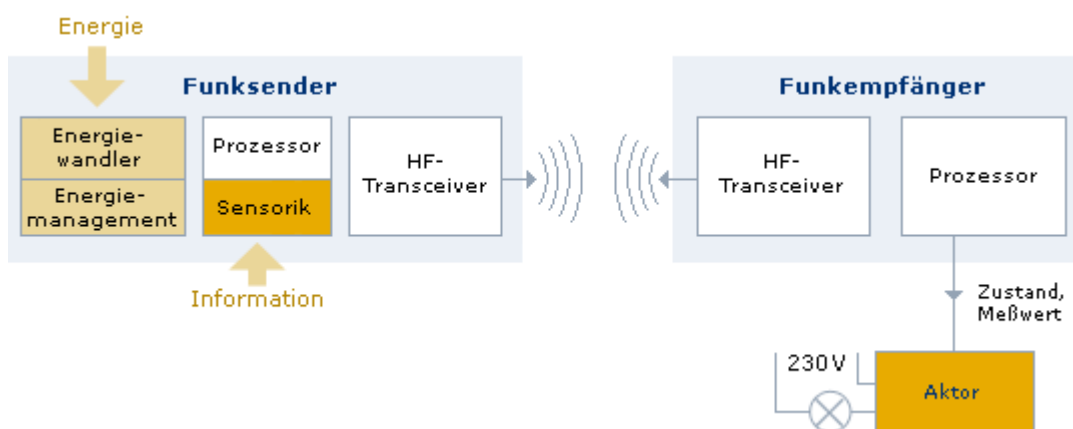


Abbildung 7: Batterie lose Funktechnologie von EnOcean

3.5 KNX/EIB

3.5.1 Grundlagen

Der EIB (European Installation Bus) ist ein dezentral aufgebautes Bussystem zur Gebäudeautomation, bei dem jeder Teilnehmer (ob Sensor oder Aktor) über einen eigenen Mikrocomputer und EEPROM verfügt.

Er ist bidirektional aufgebaut und benutzt in der ursprünglichen Variante neben der 400V/230V-Netzspannung zur Lastversorgung zusätzlich ein getrenntes, 4-adriges Twisted-Pair (TP) Kabel, zur Übertragung der Steuerbefehle. Es werden jedoch nur 2 Adern der Busleitung genutzt. Zum Einen zur Versorgung der Geräte mit einer Gleichspannung von 24V und zum Anderen zur Übertragung der Befehlstelegramme, die mit der Versorgungsspannung überlagert werden. Die beiden freien Adern sind als Reserve vorgesehen.

Die Verdrahtung der Busleitung kann in einer Linien- oder Baumstruktur erfolgen.

Eine Verdrahtung als Ring ist dagegen nicht zulässig.

Die Datenübertragungsgeschwindigkeit auf der Busleitung beträgt 9600Bit/s.

Der KNX-Standard ist eine Erweiterung des EIB-Standards und somit uneingeschränkt abwärtskompatibel. Es stehen mittlerweile vier Übertragungsmedien zur Verfügung (Twisted Pair (TP), Powerline (PL), Funk, Ethernet (KNX/IP)).

Da die KNX/EIB-Busklemmen von WAGO und Beckhoff nur Twisted Pair (TP) unterstützen, wird im Weiteren nur noch auf das Übertragungsmedium TP eingegangen.

3.5.2 Topologie

Im KNX/EIB ist die kleinste Einheit die Linie. Sie besteht aus einer Spannungsversorgung mit Drossel, Sensoren, Aktoren und der Busleitung. An einer Linie können maximal 64 Teilnehmer betrieben werden; durch Linienverstärker kann die Anzahl aber auf 256 Teilnehmer pro Linie erweitert werden.

Über Linienkoppler können maximal 15 Linien zu einem Bereich zusammengefasst werden. Es können in einer KNX/EIB-Anlage insgesamt 15 solcher Bereiche über Bereichskoppler zu einem Gesamtsystem zusammengefasst werden. Auf diese Weise können mehr als 58000 Teilnehmer über den Bus kommunizieren.

Die maximale Leitungslänge einer KNX/EIB-Linie beträgt 1000m. Dabei darf die maximale Leitungslänge zwischen zwei Teilnehmern nicht größer sein als 700m und der maximale Abstand eines Teilnehmers zur Spannungsversorgung darf 350m nicht überschreiten. Befinden sich mehrere Spannungsversorgungen in einer Linie müssen diese einen Mindestabstand von 200m haben.

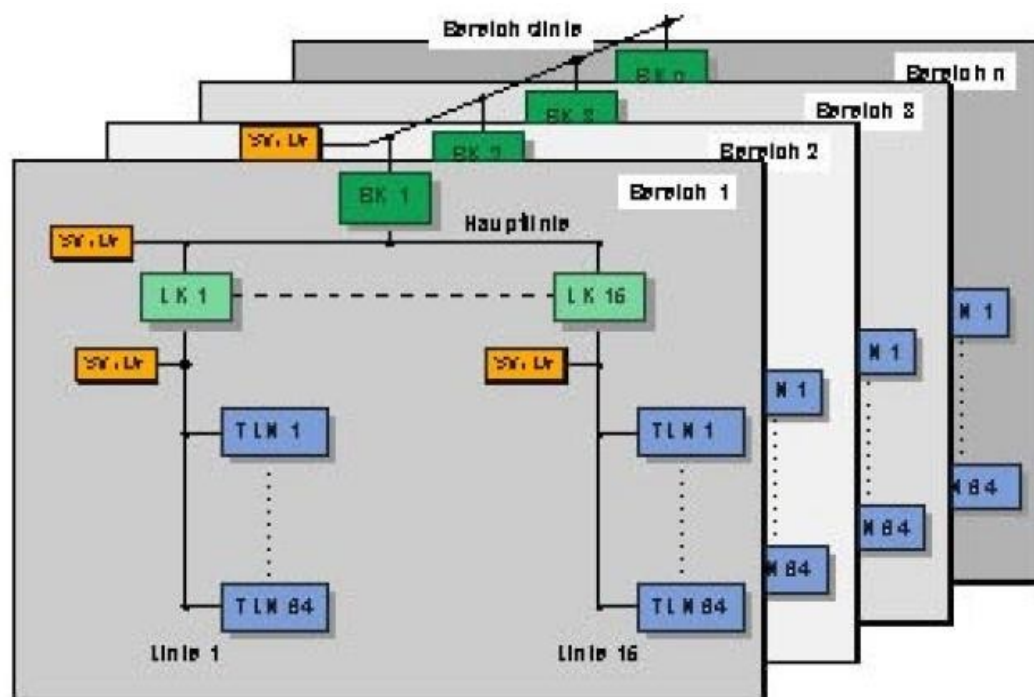


Abbildung 8: KNX/EIB Topologie

3.5.3 Adressierung

Die Adresse eines Teilnehmers wird in die physikalische Adresse und in die Gruppenadresse unterteilt.

Die physikalische Adresse gibt die Stelle in der Installation an, in der sich der Teilnehmer befindet.

Sie wird nach dem folgenden Schema zugeteilt: Bereich . Linie . Teilnehmer

Die Gruppenadresse ist eine virtuelle Adresse, die mit einem Schaltdraht vergleichbar ist. Wird zum Beispiel ein Sensor, der mit der Gruppenadresse 1/5/3 verknüpft ist, betätigt, werden alle Aktoren, die ebenfalls mit der Gruppenadresse 1/5/3 verknüpft sind, geschaltet.

Es gibt zwei Arten der Gruppenadressierung:

- 2 Ebenen bestehend aus Haupt- und Untergruppe (z.B.: 5/456)
- 3 Ebene bestehend aus Haupt-, Mittel- und Untergruppe (z.B.:3/1/58)

Bei der Gruppenadressierung im 2-Ebenen-Konzept stehen 16 Haupt- und 2048 Untergruppen und im 3-Ebenen-Konzept 16 Haupt-, 8 Mittel- und 256 Untergruppen zu Verfügung. Es sind somit 32768 Funktionen für mehr als 58000 Teilnehmer möglich.

3.5.4 KNX/EIB Software ETS

Die ETS (Engineering Tool Software) ist die herstellerunabhängige Software zur Planung und Konfiguration von Projekten mit dem KNX/EIB System.

Diese Software ist somit als ein Teil des KNX/EIB-Standards zu betrachten. Jeder KNX/EIB-Hersteller stellt für seine zertifizierten Produkte Datenbanken zur Verfügung, die in die ETS importiert werden können. Somit ist die problemlose Zusammenarbeit von Komponenten der mittlerweile über 100 Herstellern weltweit gesichert.

Die momentan aktuellste Software-Versionen ist die ETS 3.0e. Sie steht in drei verschiedenen Versionen zur Verfügung:

- ETS 3 Tester (zum Lernen)
- ETS 3 Starter (für kleine Projekte)
- ETS 3 Professional (für große und anspruchsvolle Projekte)

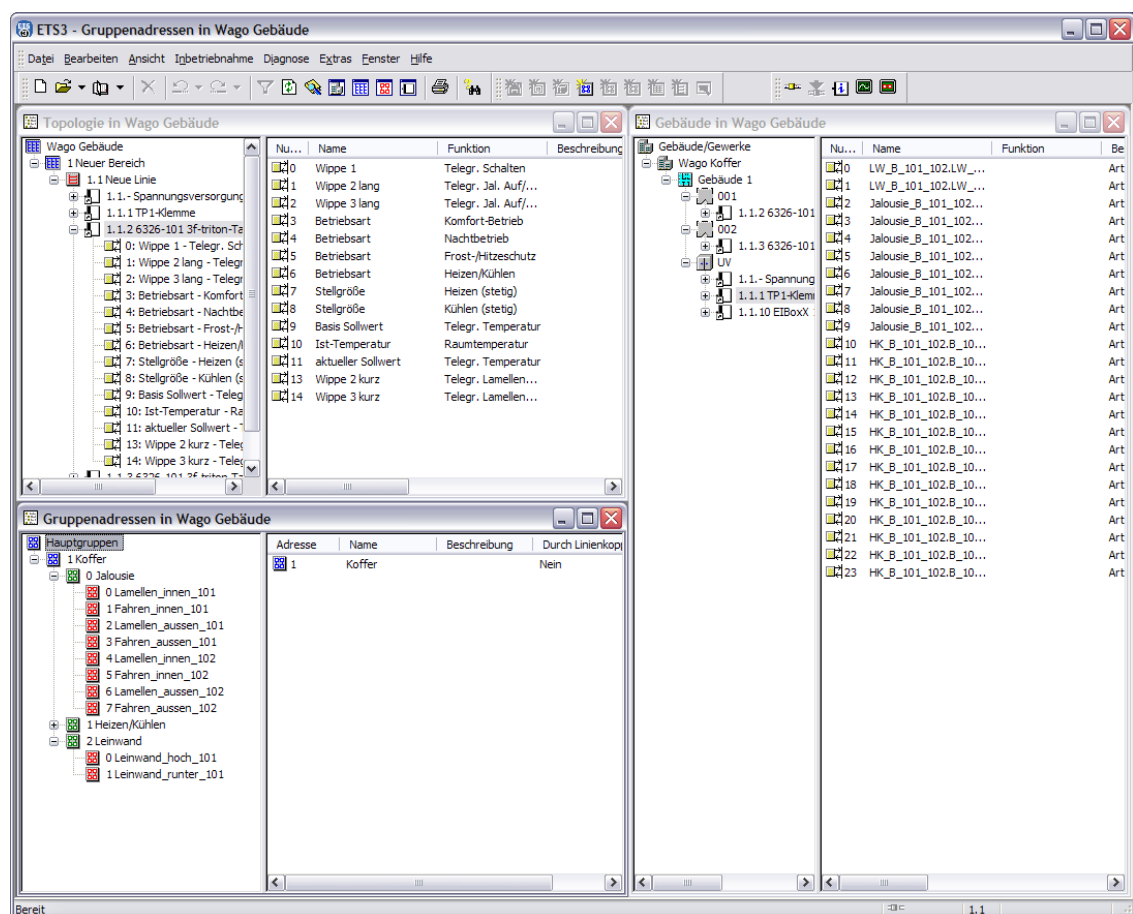


Abbildung 9: ETS Professional 3.0d

4. Die verwendeten Komponenten

4.1 Die WAGO Komponenten

Folgende Komponenten von WAGO standen in dieser Diplomarbeit zur Verfügung:

Hardware

- 1x WAGO Feldbus-Controller 750-841
- 2x 750-402, 4-Kanal Digital-Input (24V DC)
- 1x 750-430, 8-Kanal Digital-Input (24V DC)
- 2x 750-504, 4-Kanal Digital-Output (24V DC)
- 2x 750-530, 8-Kanal Digital-Output (24V DC)
- 1x 750-552, 2-Kanal Analog-Output (0-20mA)
- 1x 750-559, 4-Kanal Analog-Output (0-10V DC)
- 1x 750-452, 2-Kanal Analog-Input (0-20mA)
- 1x 750-454, 2-Kanal Analog-Input (4-20mA)
- 2x 750-467, 2-Kanal Analog-Input (0-10V DC)
- 1x 750-642, EnOcean/RF-Receiver
- 1x 753-646, KNX/EIB/TP1-Klemme
- 1x 750-600, Bus-Endklemme
- 1x Spannungsversorgung 24V DC Siemens Logo!Power

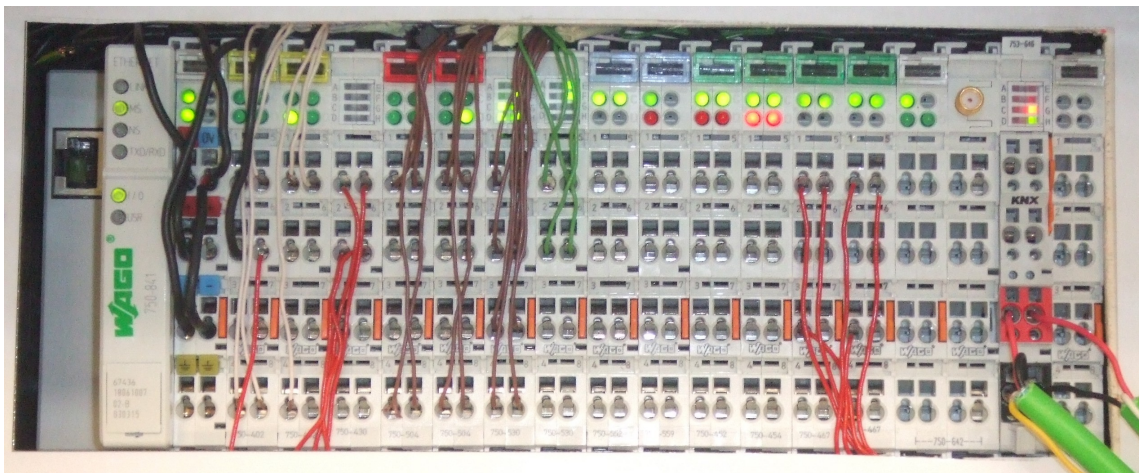


Abbildung 10: Komponenten der WAGO-Anlage

Software

- WAGO-I/O-PRO CAA
 - CoDeSys V2.3
- WAGO Ethernet Settings V4.0
- WAGO Bibliotheken für die Gebäudeautomation
 - Gebäude_allgemein.lib (Bausteine für allg. Gebäudefunktionen)
 - Building_HVAC_01.lib (Bausteine für HLK-Funktionen)
 - EnOcean04_d.lib (Bausteine für EnOcean Funkempfänger)
 - Scheduler_02.lib (Bausteine für Zeitschaltprogramme)
 - KNX_Standard_d.lib (Bausteine für KNX/EIB)
 - KNX_Applications_01_d.lib (Bausteine für KNX/EIB-Applikationen)
- WAGO-ETS3-Plug-In

4.2 Die Beckhoff Komponenten

Folgende Komponenten von Beckhoff standen in dieser Diplomarbeit zur Verfügung:

Hardware

- 1x Beckhoff CX9001-1001
- 3x KL1404, 4-Kanal Digital-Input (24V DC)
- 1x KL3454, 4-Kanal Analog-Input (4-20mA)
- 1x KL3464, 4-Kanal Analog-Input (0-10V DC)
- 5x KL2404, 4-Kanal Digital-Output (24V DC)
- 1x KL4414, 4-Kanal Analog-Output (0-20mA)
- 1x KL4404, 4-Kanal Analog-Output (0-10V DC)
- 1x KL9160, Einspeiseklemme mit LED (230V AC)
- 1x KL2751, 1-Kanal Universal-Dimmer (230V AC, 300W)
- 1x KL6301, EIB-Klemme
- 1x KL6021, EnOcean-Klemme
- 1x KL3204, 4-Kanal Analog-Input PT100 (RTD)
- 1x KL9010, Bus-Endklemme
- 1x Spannungsversorgung 24V DC Siemens Logo!Power

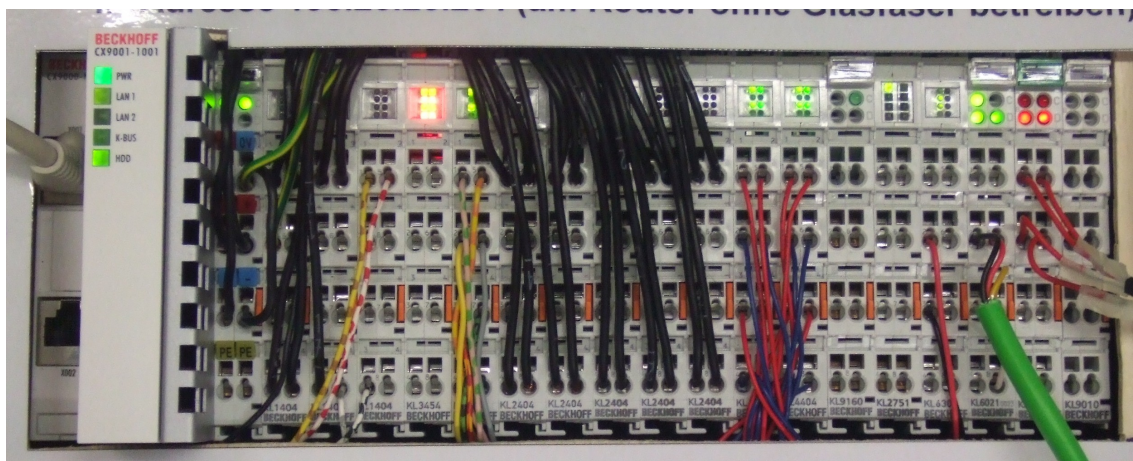


Abbildung 11: Komponenten der Beckhoff-Anlage

Software

- Beckhoff TwinCat Sytem
 - TwinCat System Manager
 - TwinCat PLC Control
- Beckhoff Bibliotheken für die Gebäudeautomation
 - TcBABasic.lib (Bausteine für allg. Gebädefunktionen)
 - TcPlcLibHVAC.lib (Bausteine für HLK-Funktionen)
 - TcEnOcean.lib (Bausteine für EnOcean Funkempfänger)
 - TcKL6301.lib (Bausteine für KNX/EIB)

4.3 Die EnOcean Komponenten

Folgende EnOcean Komponenten standen in dieser Diplomarbeit zur Verfügung:

Hardware

- 2x EnOcean PTM 200

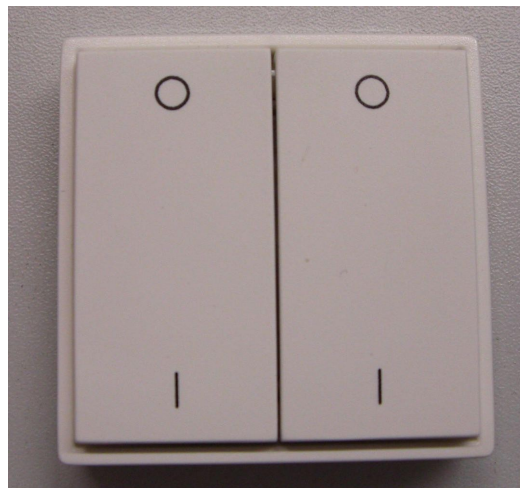


Abbildung 12: EnOcean PTM 200

4.4 Die KNX/EIB Komponenten

Folgende KNX/EIB Komponenten standen in dieser Diplomarbeit zur Verfügung:

Hardware

- 1x Siemens N 125/11 (230V/320mA) (Spannungsversorgung)
- 1x Walther EIBoxX REG-USB
- 1x Siemens N148 RS232 (RS232 Schnittstelle)
- 1x Walther EIBoxX 1602 REG (16-fach Schaltaktor)
- 2x Busch-Jaeger 6326-101 3-fach-Triton mit Raumtemperaturregler
- 2x Siemens Raumtemperaturregler UP 231/2 DELTA profil

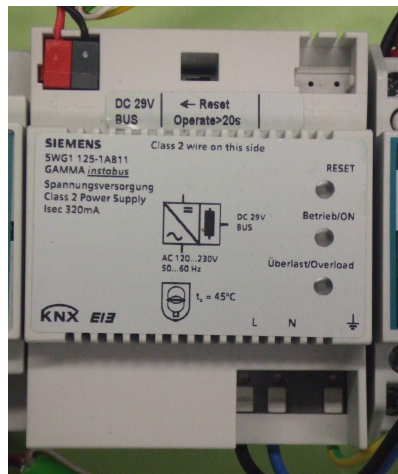


Abbildung 13: EIB-Spannungsversorgung

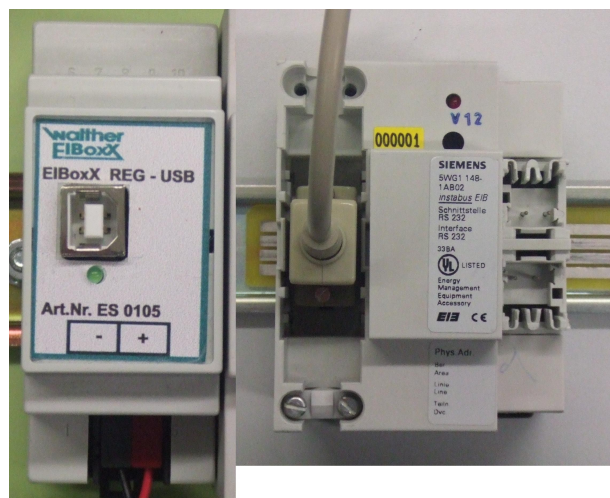


Abbildung 14: EIB-Schnittstellen

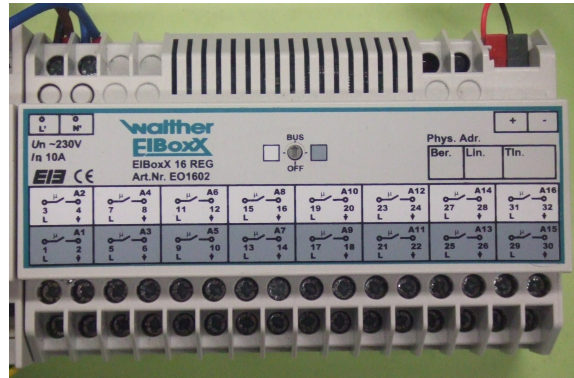


Abbildung 15: EIB-Schaltaktor



Abbildung 16: Busch-Jaeger 3-fach-Triton mit Raumtemperaturregler

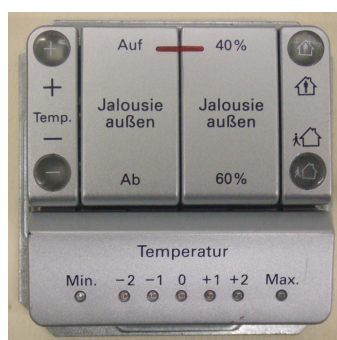


Abbildung 17: Siemens Raumtemperaturregler UP 231/2 DELTA profil

Software

- ETS2 Version 1.3
- ETS3 Professional Version 3.0d

5. Herangehensweise zur Lösung der Aufgabenstellung

5.1 Pflichtenhefte der Grundfunktionen in einem Bürogebäude

Funktionalität der Büroräume:

In jedem Büroraum sollen zwei Lichtstromkreise über je einen konventionellen Taster und einen EnOcean Funkschalter schaltbar sein. Der aktuelle Status der Lichtstromkreise soll in der Visualisierung darstellbar sein.

Die Innen- und Außenjalousie sollen über einen KNX/EIB-Tastsensor mit Raumtemperaturregler, mit den Funktionen „fahren“ und „Lamellenverstellung“, realisiert werden.

Im Falle eines zu starken Windes, soll die Außenjalousie hinauf gefahren und bis zum abschwächen des Windes verriegelt werden.

Bei erhöhter Sonneneinstrahlung dagegen, soll die Außenjalousie in eine vorgegebene Beschattungsposition gefahren werden. Die Beschattungsposition soll über eine Visualisierungsoberfläche veränderlich sein. Des Weiteren soll es möglich sein die aktuelle Position der Innen- und Außenjalousie in der Visualisierung anzuzeigen.

Ein im Büro vorhandenes Heisswassergerät soll über eine Zeitschaltuhr geschaltet werden. Die Schaltzeiten und Tage sollen über die Visualisierung vorgegeben und der Schaltstatus des Gerätes in der Visualisierung angezeigt werden.

Die Heizung und die Klimatisierung des Büroraumes sollen über den KNX/EIB-Tastsensor mit Raumtemperaturregler erfolgen. Der Benutzer soll am Tastsensor zwischen den Betriebsarten Standby und Komfort wählen können und die Basis-Temperatur des Raumes um $\pm 2K$ verändern können. Über einen Fensterkontakt soll die Heizung bei geöffnetem Fenster in die Betriebsart Frost-/Hitzeschutz gesetzt werden.

Die Nachtabenkung soll über die Leitebene geschaltet werden.

Die vom Temperatursensor des KNX/EIB-Tastsensors übermittelte Raumtemperatur, die aktuelle Solltemperatur und die aktuelle Betriebsart sollen ebenfalls in der Visualisierung angezeigt werden.

Zusammenfassende Kurzübersicht der Bürofunktionen:

- Lichtstromkreis 1 über konventionellen und EnOcean Taster schalten
- Lichtstromkreis 2 über konventionellen und EnOcean Taster schalten
- Innenjalousie über KNX/EIB-Taster fahren
- Innenjalousie über KNX/EIB-Taster verstellen
- Außenjalousie über KNX/EIB-Taster fahren
- Außenjalousie über KNX/EIB-Taster verstellen
- Außenjalousie bei starkem Wind in Sicherheitsposition fahren
- Außenjalousie bei hoher Sonneneinstrahlung in Beschattungsposition fahren
- Heisswassergerät über Zeitschaltuhr schalten
- Umschalten zwischen Standby- und Komforttemperatur über KNX/EIB-Taster
- Änderung der Solltemperatur um $\pm 2K$ über KNX/EIB-Taster
- Frost-/Hitzeschutz über Fensterkontakt setzen
- Nachtabsenkung über Leitebene setzen

Funktionalität zweier Büroräume als Konferenzraum:

Um dem Mieter eine möglichst hohe Flexibilität bei der Raumnutzung zu gewährleisten, befindet sich zwischen zwei Büroräumen eine flexible Trennwand, damit zwei Büroräume sowohl temporär als auch dauerhaft zu einem Konferenzraum umgerüstet werden können und somit diese Nutzungsänderung schnell und komfortabel gewährleistet ist.

Ein Endschalter an der Trennwand soll sowohl den Status der Nutzung in der Visualisierung anzeigen, als auch die Funktionen der beiden Büroräume zu einem Konferenzraum zusammenfügen.

Es müssen folglich beim Öffnen der Trennwand alle Funktionen des Konferenzraumes auf denselben Zustand angeglichen werden, damit der Konferenzraum von beiden Schaltstellen bedienbar ist. Hier ist besonders auf den Abgleich einer Veränderung der Raumtemperatur an den Schaltstellen zu achten.

Eine in die Decke eingelassene Leinwand für Präsentationen soll über einen KNX/EIB-Schaltaktor bei geöffneter Trennwand heruntergefahren werden können. Wird die Trennwand geschlossen, muss die Leinwand automatisch wieder hochfahren.

Zusammenfassende Kurzübersicht der Funktionen im Konferenzraum:

- Lichtstromkreis 1 über zwei konventionelle und zwei EnOcean Taster schalten
- Lichtstromkreis 2 über zwei konventionelle und zwei EnOcean Taster schalten
- Innenjalousie über zwei KNX/EIB-Taster fahren
- Innenjalousie über zwei KNX/EIB-Taster verstellen
- Außenjalousie über zwei KNX/EIB-Taster fahren
- Außenjalousie über zwei KNX/EIB-Taster verstellen
- Außenjalousie bei starkem Wind in Sicherheitsposition fahren
- Außenjalousie bei hoher Sonneneinstrahlung in Beschattungsposition fahren
- Heisswassergerät über Zeitschaltuhr schalten
- Umschalten zwischen Standby-/Komforttemperatur über zwei KNX/EIB-Taster
- Änderung der Solltemperatur um $\pm 2K$ über zwei KNX/EIB-Taster
- Frost-/Hitzeschutz über Fensterkontakt setzen
- Nachtabsenkung über Leitebene setzen
- Leinwand über konventionelle Taster fahren

Funktionalität des Büroflures:

Durch eine kurze Betätigung des Lichttasters soll das Flurlicht als Treppenhauslicht mit Ausschaltvorwarnung und bei einer langen Betätigung des Tasters als Dauerlicht (Reinigungsbeleuchtung) eingeschaltet werden.

Zusammenfassende Kurzübersicht der Funktionen im Büroflur:

- Lichtstromkreis 1 über konventionellen Taster
 - kurzer Tastendruck als Treppenhauslicht mit Ausschaltvorwarnung
 - langer Tastendruck als Dauerlicht
- Lichtstromkreis 2 über konventionellen Taster
 - kurzer Tastendruck als Treppenhauslicht mit Ausschaltvorwarnung
 - langer Tastendruck als Dauerlicht

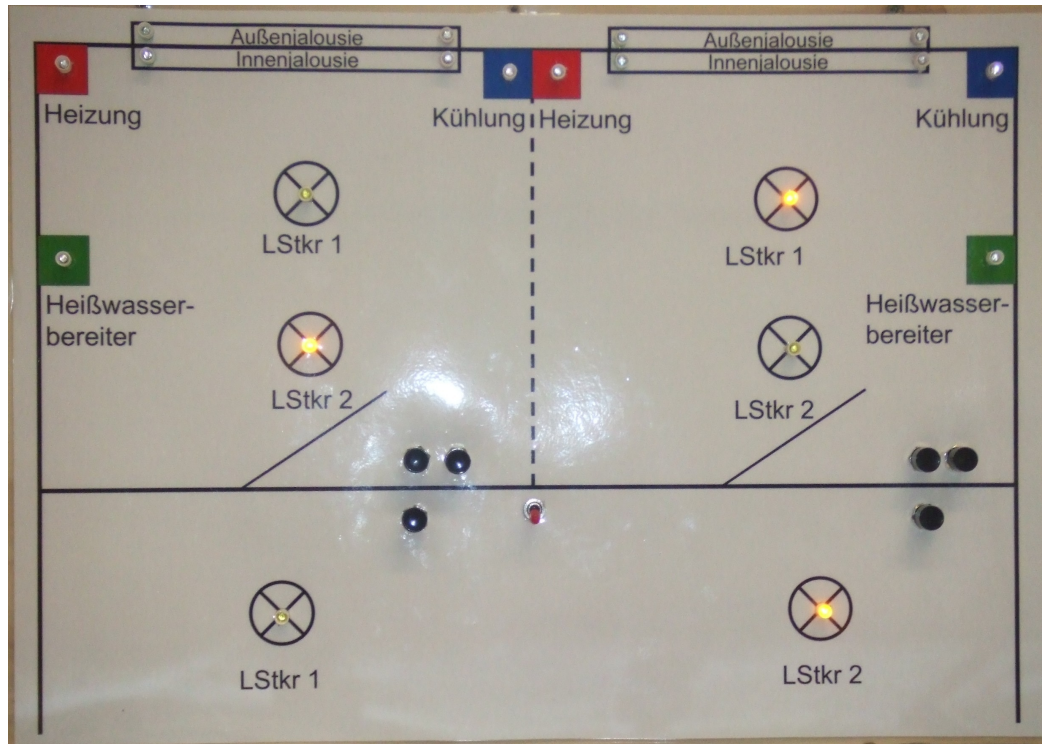


Abbildung 18: Funktionsanzeige des Demokoffers

5.2 Umsetzung der Funktionen mit dem WAGO Controller 750-841

Als erstes muss die „WAGO-I/O-Pro CAA Software (CoDeSys)“ und das Programm „WAGO Ethernet Settings“ installiert werden. Nach erfolgreicher Installation müssen noch alle benötigten Zusatzbibliotheken nach „C:\Programme\WAGO Software\CoDeSys V2.3\Targets\WAGO\Libraries\Building\German“ kopiert werden. Bei der KNX/EIB-Bibliothek ist dabei zu beachten, dass die „SysLibGetAddress.lib“ in die Verzeichnisse „C:\Programme\WAGO Software\CoDeSys V2.3\Targets\WAGO\Libraries\32_Bit“ und „C:\Programme\WAGO Software\CoDeSys V2.3\Targets\WAGO\Libraries\16_Bit“ kopiert werden.

Eine Realisierung von Gebäudeautomationsfunktionen ohne diese Bibliotheken wäre sehr zeitaufwändig, da selbst einfache und ständig benötigte Funktionen wie z.B. Stromstoßschalter, Auswertung auf kurzen oder langen Tastendruck, Treppenhauslicht und ähnliche Funktionen, jedes mal neu und relativ aufwendig realisiert werden müssten.

5.2.1 Verbindungsaufbau zur WAGO-SPS

Um die WAGO-SPS über die Ethernet-Schnittstelle programmieren zu können, muss zuerst über das beiliegende serielle Programmierkabel und das Programm „WAGO Ethernet Settings“ dem Controller eine IP-Adresse zugewiesen werden. Nach dem Start des Programms werden die aktuellen Einstellungen der angeschlossenen Anlage ermittelt.

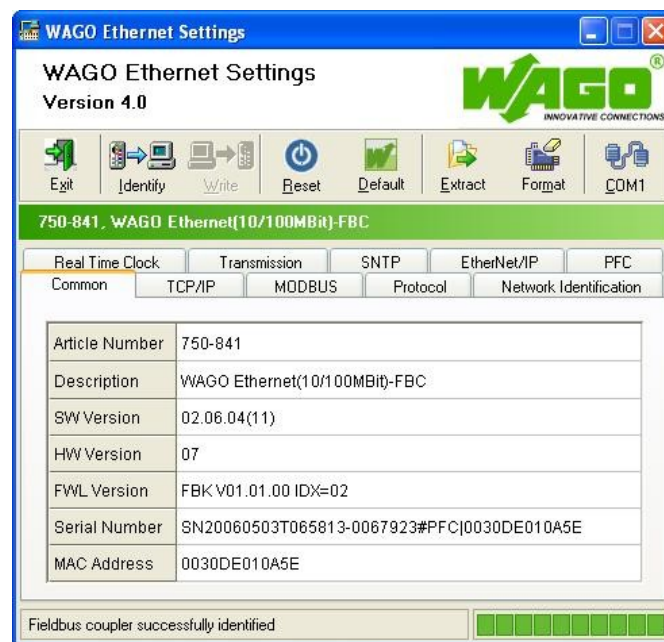


Abbildung 19: WAGO Ethernet Settings

Sollte ein Verbindungs-Timeout auftreten, kann über die Schaltfläche „Identify“ eine Aktualisierung der Anzeige erzwungen werden.

Über die Karteikarte „TCP/IP“ wird die IP-Adresse für die angeschlossene Anlage vergeben. Nachdem diese eingetragen wurden, muss die Änderung über die Schaltfläche „Write“ in die Anlage übertragen werden.



Abbildung 20: Einstellung der IP-Adresse des Controllers

Nun kann die Anlage mit dem Ethernet entweder direkt mit dem PC über ein Crossover-Kabel oder mit einem 1-zu-1 Patchkabel über einen Hub oder Switch verbunden werden.

5.2.2 Allgemeine Programmierung der WAGO-SPS

Nachdem die WAGO-SPS eine IP-Adresse bekommen hat, kann nun das Programm für die SPS geschrieben werden.

Sobald die WAGO-I/O-Pro CAA Software (CoDeSys) gestartet ist, wird über „Datei - Neu“ ein neues Projekt angelegt. Es öffnet sich nun automatisch das Fenster „Zielsystem Einstellung“. In diesem Dialogfenster wird das Zielsystem, in diesem Fall „WAGO_750-841“ ausgewählt und mit „OK“ bestätigt.

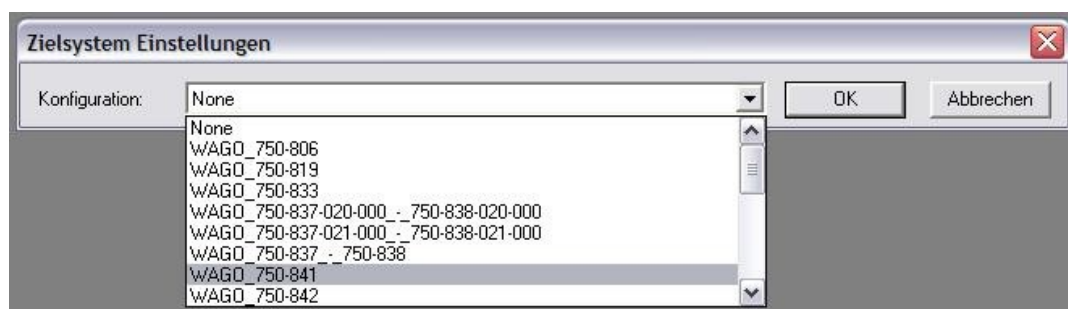


Abbildung 21: CoDeSys „Zielsystemeinstellung“

Im folgenden Dialogfenster wird der erste Programmbaustein angelegt. In diesem Fenster bekommt der Baustein einen Namen, seinen Typ und die Programmiersprache zugewiesen. Durch bestätigen mit „OK“ wird der Baustein in der CoDeSys angelegt.



Abbildung 22: CoDeSys „Neuer Baustein“

Bevor nun die Erstellung des Programmes beginnen kann, ist erst noch die Steuerungskonfiguration durchzuführen. Dazu ist im Register unten links „Ressourcen“ auszuwählen und dann „Steuerungskonfiguration“.

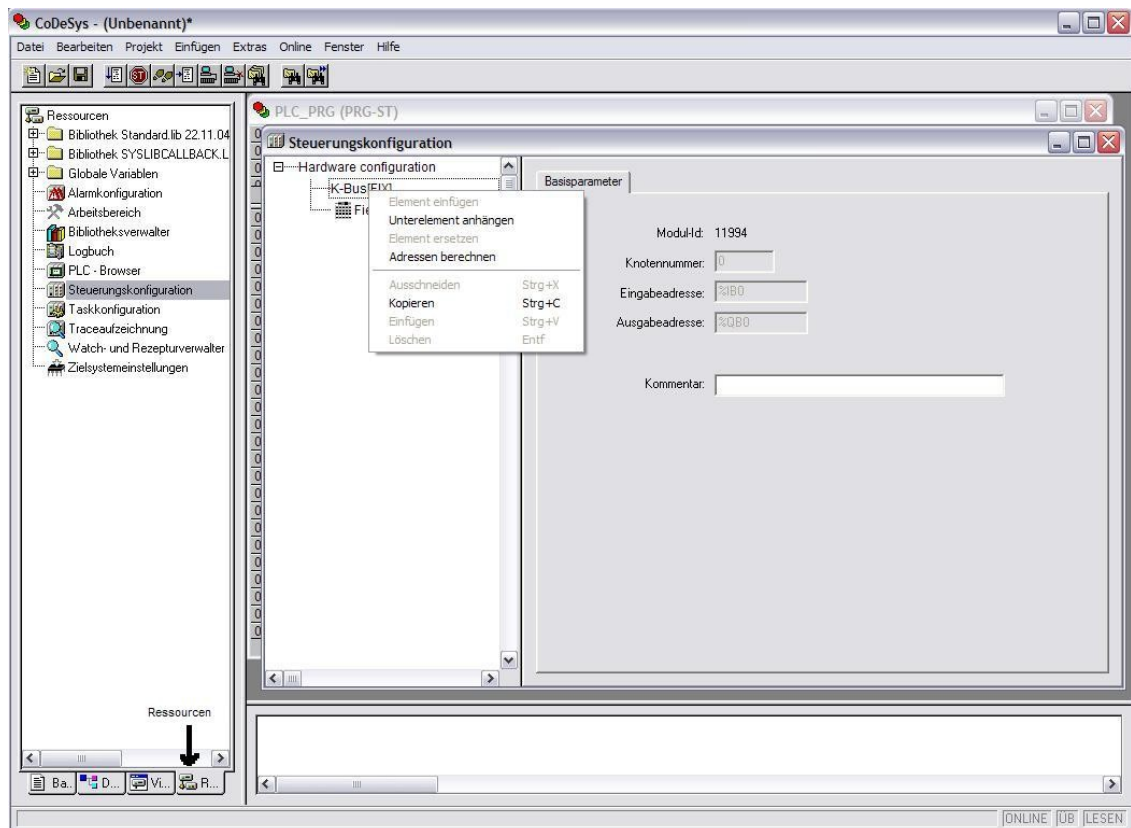


Abbildung 23: CoDeSys „Steuerungskonfiguration“

Über einen Rechtsklick auf den Eintrag „K-Bus[FIX]“ öffnet sich ein Kontextmenü, indem der Eintrag „Unterelement anhängen“ ausgewählt wird. Dadurch wird der „I/O-Konfigurator“ geöffnet.

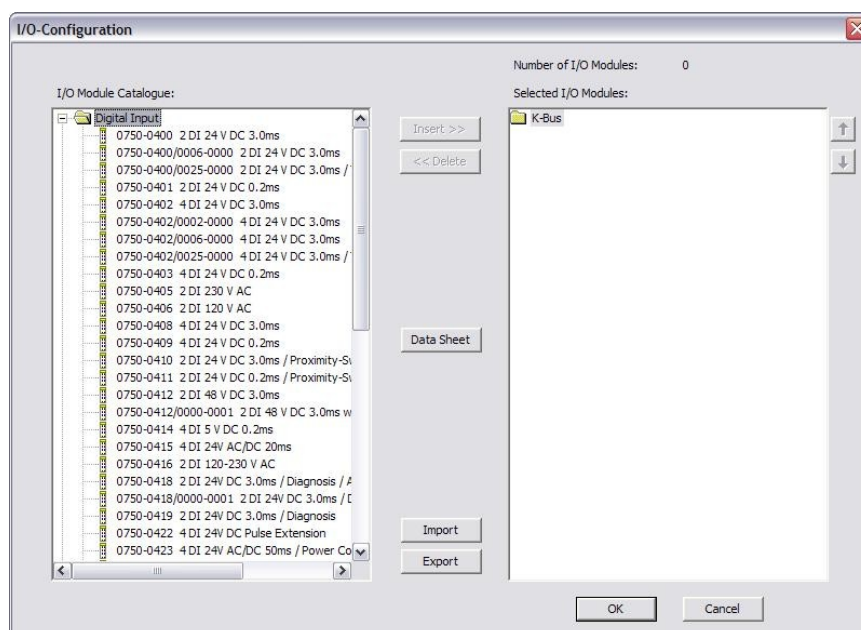


Abbildung 24: CoDeSys „I/O-Konfigurator“

Im „I/O-Konfigurator“ werden nun alle am Controller vorhandenen Klemmen in der gesteckten Reihenfolgen über „Insert“ hinzugefügt. Eine automatische Erkennung der Klemmen am Bus ist nicht möglich. Die Bus-Endklemme wird nicht angegeben.

Durch bestätigen mit „OK“ werden die ausgewählten Klemmen in die Steuerungskonfiguration übernommen.

Dieser zeigt nun die Adresse jedes Kanals auf der Klemme an. Wird nun mit der linken Maustaste vor das Wort „AT“ der Adressbezeichnung geklickt, öffnet sich ein Beschriftungsfeld, in das der Variablenname eingetragen werden kann. Hierbei ist zu beachten, dass Leerzeichen und Sonderzeichen in Variablennamen nicht zulässig sind.

Alle beschrifteten Ein- und Ausgänge stehen nun als System-Variablen global im Projekt zur Verfügung.

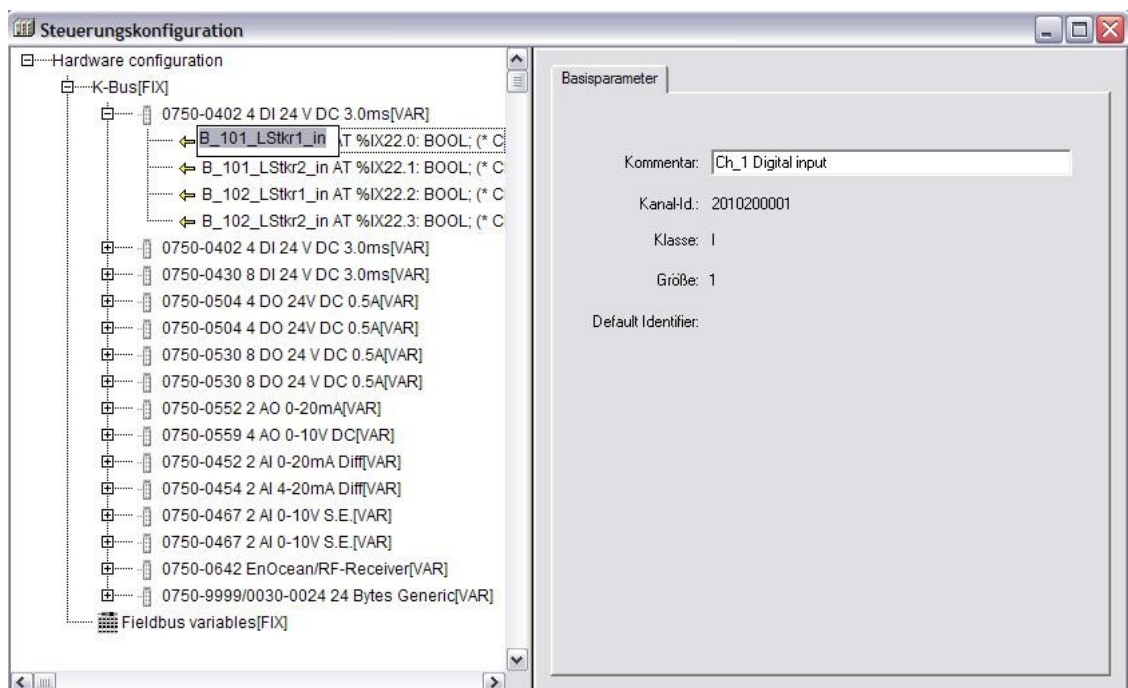


Abbildung 25: Vergeben der Variablennamen

Nun müssen nur noch die benötigten Zusatzbibliotheken über den „Bibliotheksverwalter“, der ebenfalls unter „Ressourcen“ zu finden ist, hinzugefügt werden. Mit einem rechten Mausklick in den Bereich oben links des „Bibliotheksverwalters“ öffnet sich das Kontextmenü, indem „Weitere Bibliothek ...“ ausgewählt werden muss. Im nun geöffneten Fenster kann die gewünschte Bibliothek ausgewählt und geöffnet werden.

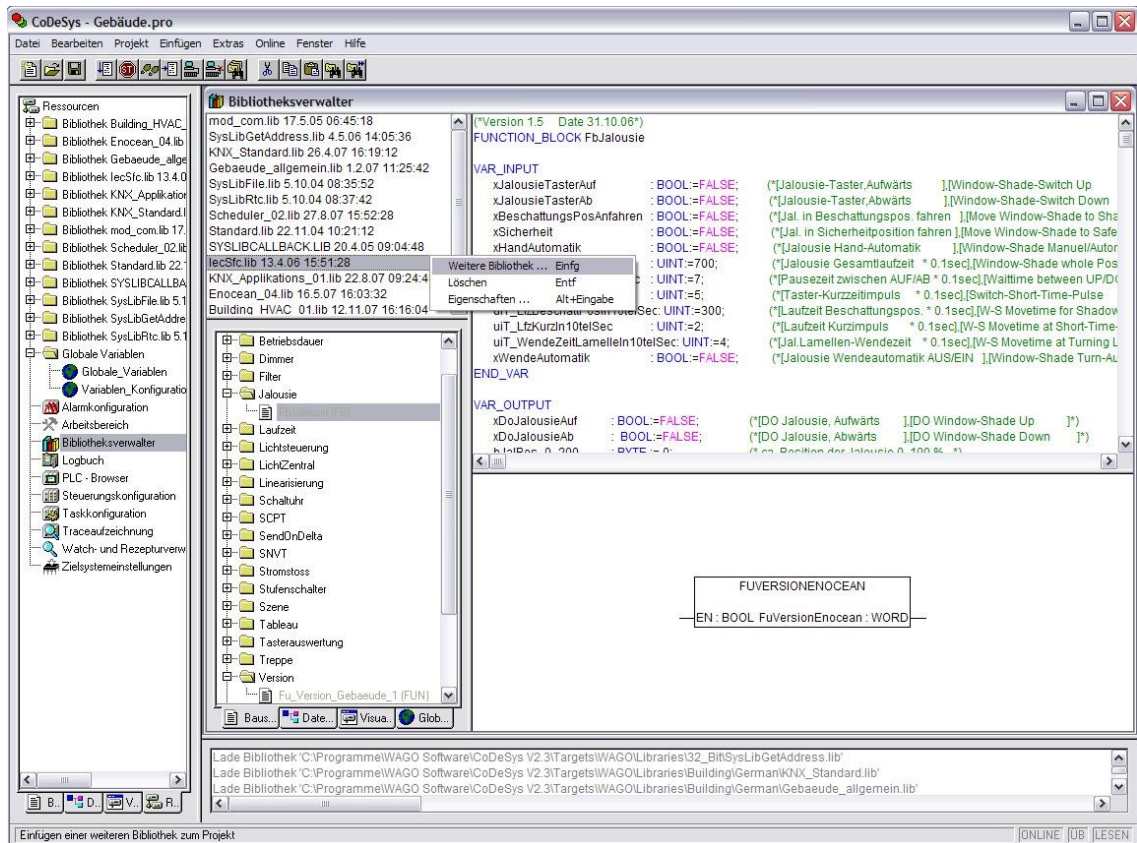


Abbildung 26: CoDeSys „Bibliotheksverwalter“

Nachdem die Steuerungskonfiguration durchgeführt und die Bibliotheken hinzugefügt sind, kann nun das SPS-Programm geschrieben werden. Dazu wechselt man wieder auf die Karteikarte „Bausteine“, in dem sich über einen Rechtsklick auf den Ordner „Bausteine“ das Kontextmenü öffnen lässt. Über dieses Kontextmenü lassen sich neue Bausteine anlegen und verwalten; außerdem können hier Ordner angelegt und benannt werden. Diese Ordner haben keinen Einfluss auf die Programmierung, sind aber sehr nützlich bei der Übersichtlichkeit eines größeren Programms (Abbildung 27).

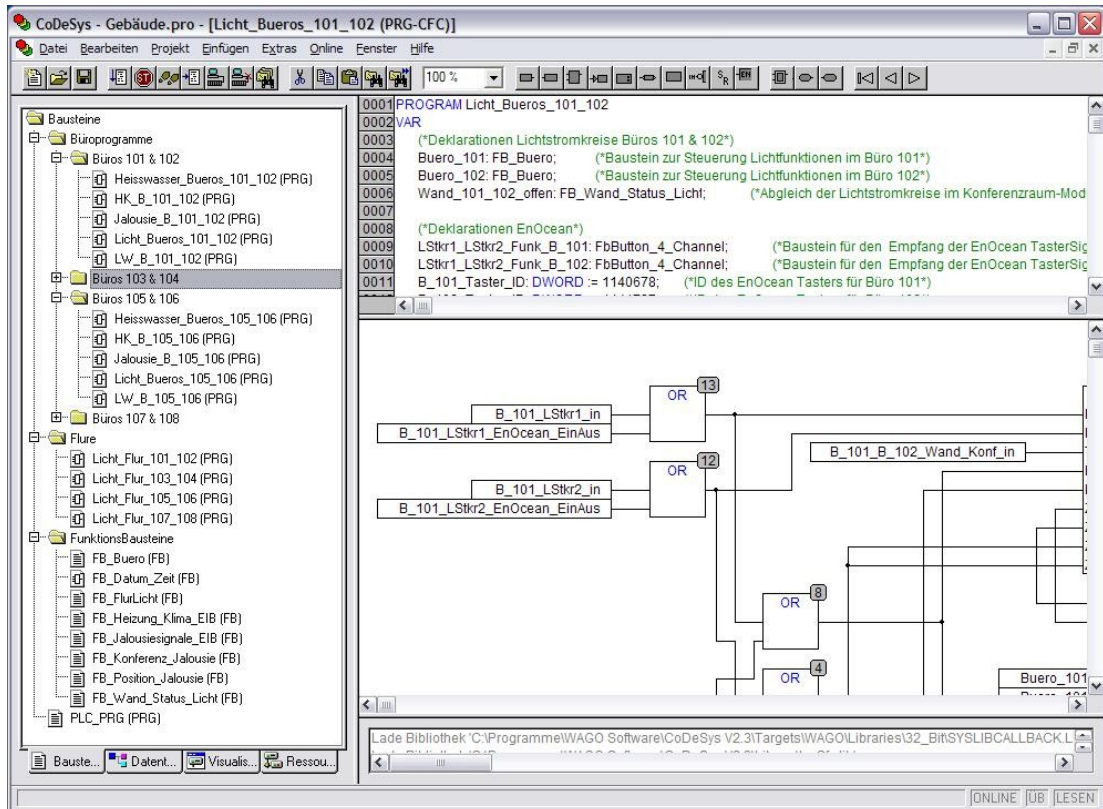


Abbildung 27: Übersichtlichkeit durch Ordner

Funktionalitäten, die in einem Projekt oft benötigt werden (z.B.: Abfrage von Zuständen, Jalousiesteuerung im Büroraum), sollten als Funktionsblock programmiert werden. Die Funktion muss somit nicht immer für jeden Raum neu programmiert werden und des Weiteren wird der Programmbaustein somit deutlich übersichtlicher für eine später eventuell notwendige Fehleranalyse. Anstatt die Funktionen jedesmal neu zu programmieren wird nur noch dieser Funktionsblock im Programmbaustein aufgerufen und kann dort mit den Ein-/Ausgangsvariablen und Parameter versehen werden, sowie mit weiteren Bausteinen erweitert werden.

In den grafischen Programmiersprachen müssen über das Kontextmenü zu erst Bausteine, Ein- und Ausgänge hinzugefügt werden. Um die Art eines Bausteins oder die Variable eines Ein-/Ausgangs zu ändern, muss der aktuelle Name des Bausteins bzw. des Ein-/Ausgangs markiert werden. Über die Taste „F2“ wird die Eingabehilfe aufgerufen, in der alle vorhandenen Funktionsblöcke, Variablen und Programme vorhanden sind, sodass nur noch der gewünschte Funktionsblock bzw. die Variable ausgewählt werden muss.

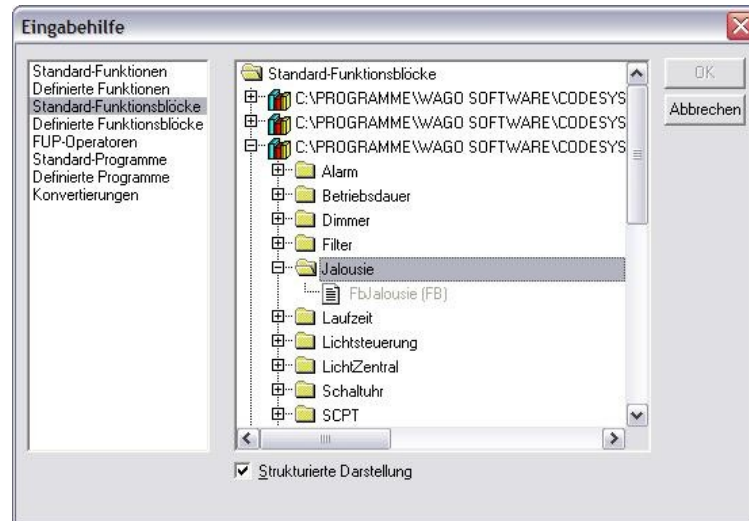


Abbildung 28: CoDeSys „Eingabehilfe“

Bei Variablen die noch nicht deklariert wurden, kann der Variablenname auch direkt in den Ein-/Ausgang geschrieben werden. Nach der Bestätigung des Variablennamens mit der Enter-Taste öffnet sich das Fenster „Variablendeklaration“, in dem alle Einstellungen für die Variable vorgenommen werden können.

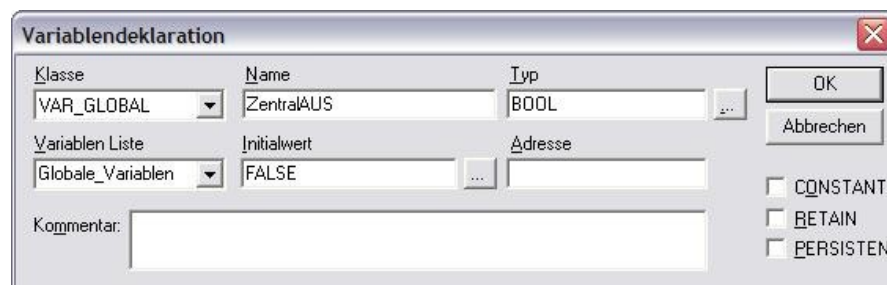


Abbildung 29: CoDeSys „Variablendeklaration“

Der fertige Programmbaustein kann jetzt über das Kontextmenü des Programmbausteins mit „Objekt kopieren“ kopiert werden, sodass identische Räume nicht einzeln programmiert werden müssen. Es müssen in den kopierten Räumen lediglich die Ein-/Ausgangsvariablen angepasst werden (Abbildung 30). Es ist somit möglich, viele identische Räume, wie in großen Bürogebäuden üblich, innerhalb kürzester Zeit zu programmieren.

Hierzu ist es wichtig, die Benennung der Einzelfunktionen (Datenpunkte) hinsichtlich der Gebäudetopologie sinnvoll zu wählen.

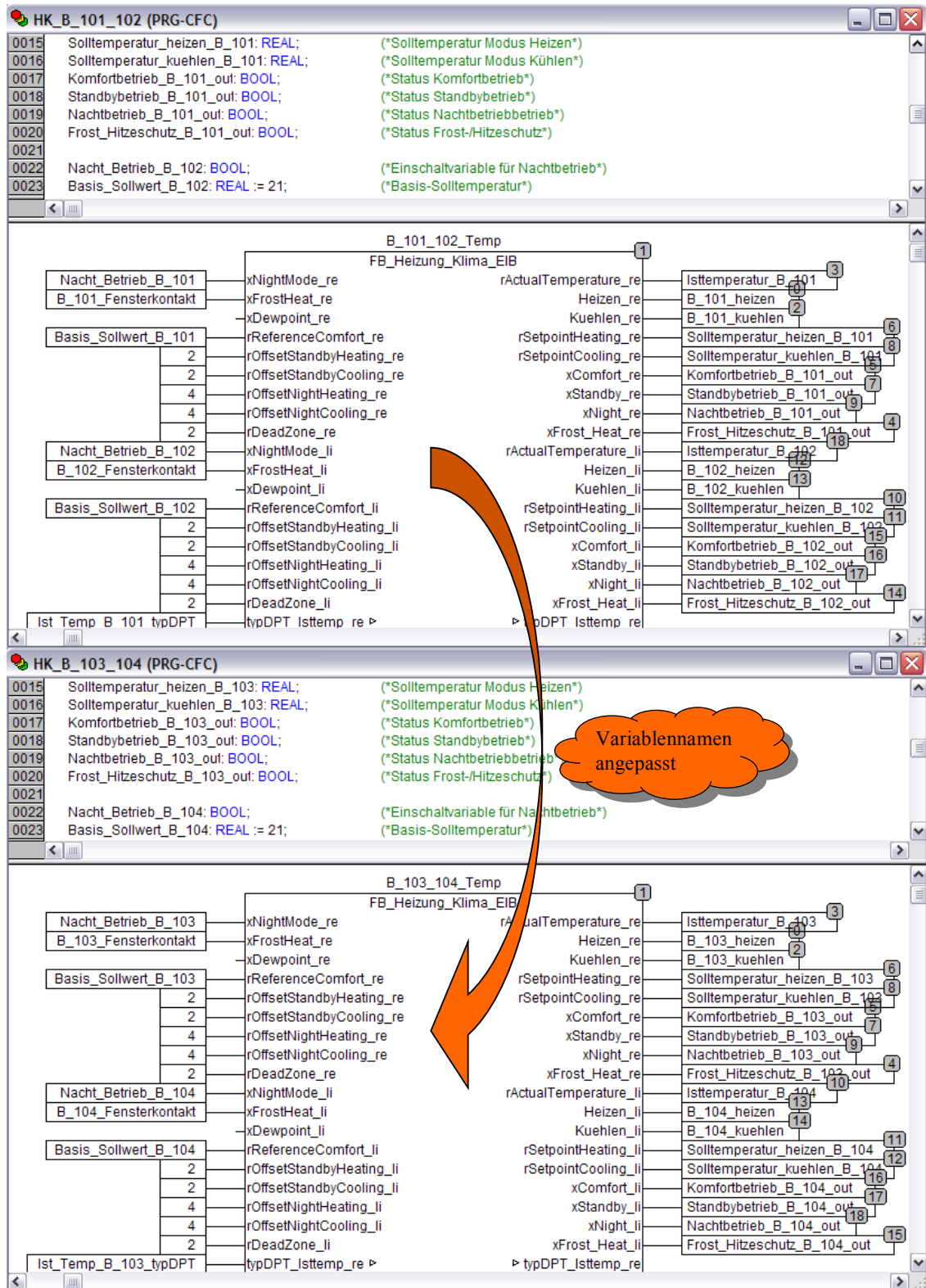


Abbildung 30: Kopierter Programmbaustein

Wenn alle Programmbausteine programmiert sind, müssen diese noch im
Programmbaustein „PLC_PRG“ aufgerufen werden.

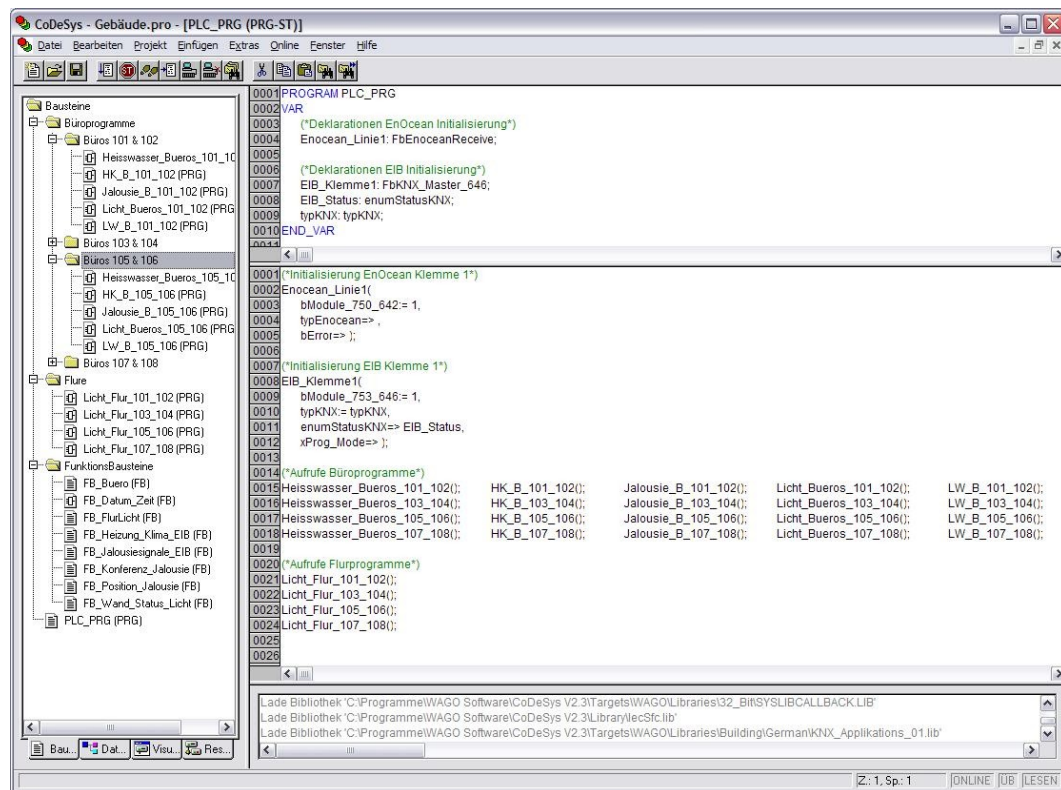


Abbildung 31: CoDeSys Hauptprogramm

Über den Menüpunkt „Projekt – Alles Übersetzen“ muss das gesamte Projekt übersetzt
werden. Danach kann das Projekt gespeichert werden.

Über den Menüpunkt „Online – Kommunikationsparameter...“ muss nun die IP-Adresse der Anlage hinzugefügt werden.

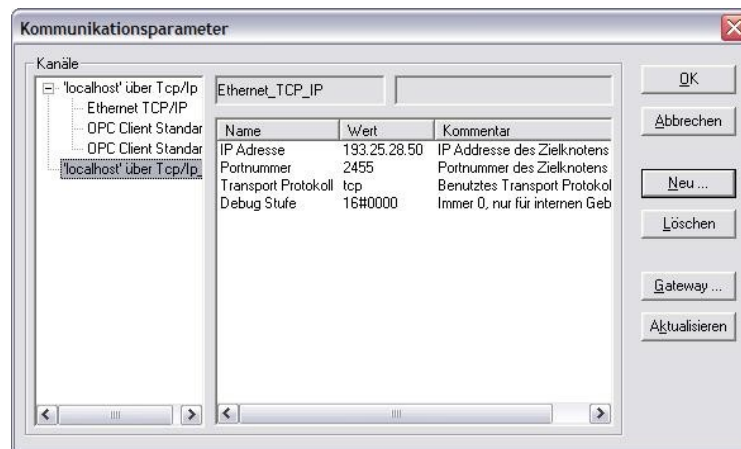


Abbildung 32: CoDeSys „Kommunikationsparameter“

Dazu muss im Fenster „Kommunikationsparameter“ über die Schaltfläche „Neu...“ ein neuer Kanal angelegt werden. Hier kann ein Name für die Anlage vergeben werden, um mehrere Anlagen unterscheiden zu können. Über „OK“ wird ein neuer Kanal im Fenster „Kommunikationsparameter“ angelegt. Jetzt muss noch die IP-Adresse des neuen Kanals eingetragen und mit „OK“ bestätigt werden.

Das Projekt kann nun über „Online – Einloggen“ in die Anlage geladen werden. Wenn das Projekt fertig geladen ist, muss die Anlage über „Online – Start“ gestartet werden. Um die Anlagenkonfiguration bei einem Spannungsausfall nicht zu verlieren, muss über „Online – Bootprojekt erzeugen“ noch das Bootprojekt der Anlage erzeugt werden.

5.2.3 Implementierung der EnOcean-Klemme von WAGO

Zur Kommunikation mit EnOcean Geräten über die EnOcean-Funk-Klemme 750-642 werden in der Bibliothek „EnOcean04_d.lib“ spezielle Funktionsbausteine zur Verfügung gestellt.

Jede am Klemmenbus verfügbare EnOcean-Funk-Klemme benötigt zur Kommunikation mit der IEC 61131-Applikation einen Baustein, der die gesamte Kommunikation verwaltet.

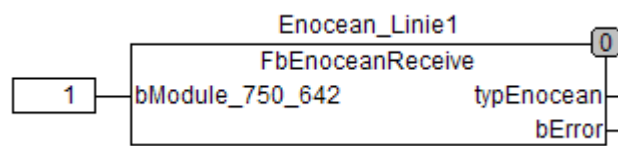


Abbildung 33: WAGO FbEnOceanReceive-Baustein

Am Eingang „bModule_750_642“ des Bausteins „FbEnOceanReceive“ wird der Klemmenindex der gesteckten EnOcean-Funk-Klemme als Konstante eingetragen (erste gesteckte EnOcean-Klemme => 1, zweite EnOcean-Klemme => 2 usw.). Pro gesteckter EnOcean-Funk-Klemme darf dieser Baustein nur einmal benutzt werden.

Der Baustein „FbEnOceanReceive“ empfängt alle gesendeten Signale und leitet diese über den Ausgang „typEnOcean“ an die EnOcean-Funktionbausteine weiter.

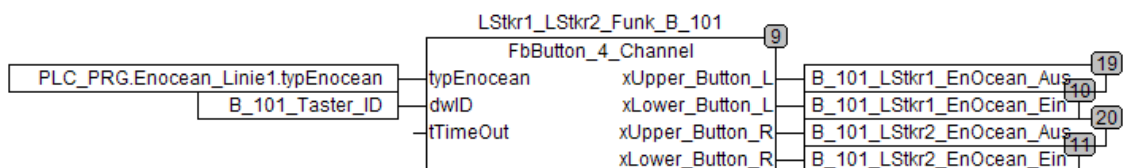


Abbildung 34: WAGO EnOcean Baustein 4-Kanal

Für alle verfügbaren EnOcean Sensoren stehen Funktionsbausteine zur Verfügung. Diese Bausteine werden über den Eingang „typEnOcean“ mit dem Ausgang „typEnOcean“ des zugehörigen „FbEnOceanReceive“-Bausteins verbunden, um die Signale zu empfangen. Am Eingang „dwID“ wird die Transmitter ID des EnOcean-Sensors eingetragen.

Sollte die Transmitter ID des Sensors nicht bekannt sein, stellt die „EnOcean04_d.lib“ einen Baustein zur Verfügung, mit der die ID ermittelt werden kann.

An den Ausgängen des Bausteins werden die Signale des Sensors in Abhängigkeit der gedrückten Taste auf TRUE gesetzt.

In der Dokumentation der EnOcean Bibliothek sind alle verfügbaren Bausteine detailliert beschrieben. Diese sollte daher bei der Programmierung genutzt werden, um Fehler bei der Deklaration der Variablen zu vermeiden.

5.2.4 Implementierung der KNX/EIB-Klemme von WAGO

Um über die KNX/EIB/TP1-Klemme 753-646 mit dem KNX/EIB kommunizieren zu können, werden spezielle Funktionsbausteine benötigt, welche die Schnittstelle zwischen den KNX/EIB-Objekten und den IEC-Variablen darstellen. Jede am Klemmenbus verfügbare KNX/EIB-Klemme benötigt zur Kommunikation zwischen der Klemmen und der IEC-61131-Applikation einen FbKNX_Master_646-Baustein.

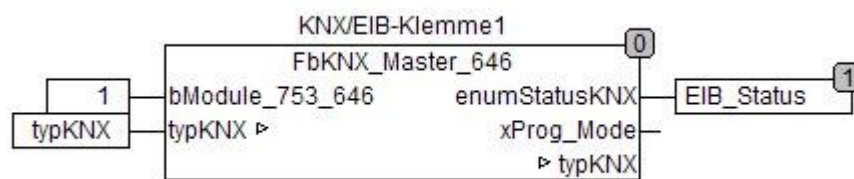


Abbildung 35: WAGO FbKNX_Master_646-Baustein

Der KNX-Master-Baustein erfasst alle anstehenden Kommandos der weiteren KNX-Baustein im Programm und sorgt für deren Ausführung. Alle Kommandos werden über die Ein-/Ausgangsvariable „typKNX“ als Datentabelle zur Verfügung gestellt. Am Eingang „bModule_753_646“ muss der entsprechende Klemmenindex der gesteckten Klemmen als Konstante eingetragen werden (erste KNX-Klemme => 1, zweite KNX-Klemme => 2 usw.).

Am Ausgang „enumStatusKNX“ wird der aktuelle Betriebszustand der KNX/EIB-Klemme angezeigt.

Dieser Baustein darf nur einmal pro gesteckter KNX/EIB-Klemme benutzt werden.

Für den Zugriff auf die KNX/EIB-Gruppenadressen stehen für jeden im KNX/EIB verfügbaren Data-Point-Typ (DPT) ein entsprechender Funktionsbaustein zur Verfügung. Die DPT-Funktionsbausteine stellen die Kommunikationsobjekte zum KNX/EIB dar. Sie können sowohl zum Senden, als auch zum Empfangen von Objektwerten verwendet werden.

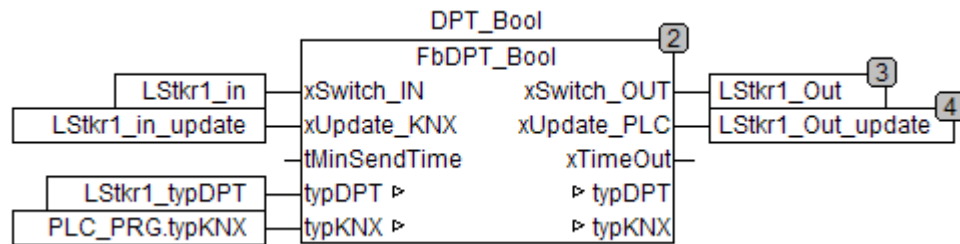


Abbildung 36: WAGO KNX-DPT_Bool-Baustein

Über „xSwitch_IN“ des FbDPT_BOOL kann ein Wert auf den KNX/EIB-Bus gesendet werden. Für den Fall, dass derselbe Wert noch einmal gesendet werden soll, kann man über „xUpdate_KNX“ ein Telegrammupdate initiieren. Um eine zu hohe Busbelastung bei sich häufig ändernden Signalen zu verhindern, kann über „tMinSendTime“ der minimale Zeitabstand für das Senden dieses Bausteins bestimmt werden. Der Eingang „typDPT“ des Bausteins muss zwingend beschaltet werden. Dieser Eingang erlaubt die Deklaration der Objektvariablen als RETAIN, wodurch der Baustein auf den letzten Wert vor einem Spannungsausfall bzw. Reset gesetzt wird. Um mit dem KNX/EIB-System kommunizieren zu können, muss der Ein-/Ausgang „typKNX“ mit dem Ein-/Ausgang „typKNX“ des zugehörigen KNX-Master-Bausteins verbunden sein.

Über die Ausgänge „xSwitch_OUT“ und „xUpdate_PLC“ werden der Objektwert und die Meldung einer Telegrammänderung empfangen. Bei aktiver Timeout-Überwachung eines KNX-Objekts wird die Überschreitung der Überwachungszeit am Ausgang „xTimeOut“ angezeigt.

Alle weiteren DPT-Bausteine sind ähnlich aufgebaut und detailliert in der beiliegenden Dokumentation beschrieben. Der Master-Baustein und alle DPT-Bausteine sind in der WAGO KNX_Standard_d Bibliothek zu finden.

Des Weiteren stellt WAGO eine Bibliothek zur Verfügung, die Applikations-Bausteine der meist genutzten KNX/EIB-Geräte beinhaltet. Es handelt sich bei diesen Bausteinen um virtuelle KNX/EIB-Geräte, die mit Ein-/Ausgängen der WAGO-SPS simuliert werden. Diese Bausteine sollen das Nachstellen von KNX/EIB-Geräten in der IEC-61131 vereinfachen und sind in der Bibliothek KNX_Applikations_01.lib zu finden.

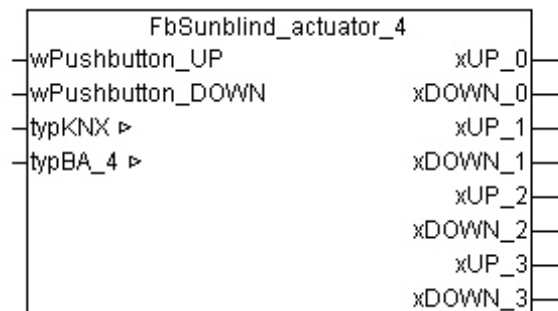


Abbildung 37: WAGO Funktionsbaustein KNX Jalousieaktor

Zu jedem dieser Bausteine verfügt die Bibliothek auch über ein Visualisierungselement mit dem die Parameter des Bausteins komfortabel eingestellt werden können.

Diese Applikationen sind aber nur einsetzbar, wenn eine direkte Sensor-Aktor Verbindung erstellt wird, z.B. ein KNX/EIB-Jalousieaktor wird direkt über einen KNX/EIB-Taster gesteuert, nur das hier der KNX/EIB-Jalousieaktor ein virtuelles Geräte in der IEC 61131 ist, dass sich wie ein Standard-KNX/EIB-Jalousieaktor verhält und die Ausgänge der WAGO-SPS ansteuert.

Für den Fall, dass zwischen dieser Sensor-Aktor-Verbindung eine Automatisierungsschicht, wie im Pflichtenheft dieses Projektes gefordert, eingefügt werden muss, ist dies mit den KNX-Applikationen nicht möglich, da der Schaltbefehl erst am Ausgang und nicht zwischen Sensorik und Aktorik abgefangen und durch weitere Bedingungen erweitert werden kann.

Hier müssen die Werte somit über die DPT-Bausteine gelesen und gesendet werden und mit herkömmlichen Bausteinen weiter verarbeitet werden.

Die KNX_Applikations_01.lib stellt hier zwei weitere Bausteine mit den Namen „FbDimmerDoubleSwitch_KNX“ und „FbSunblind_KNX“ zur Verfügung.

Diese Bausteine befinden sich in einem Ordner „Private“ der Bibliothek und sind in der zugehörigen Bibliothekenbeschreibung leider nicht dokumentiert. Die Belegung der Ein- und Ausgänge dieser beiden Bausteine konnten nur an Hand der im „Bibliotheksverwalter“ vorhandenen Variablenkommentare zu den einzelnen Ein- und Ausgangsvariablen der Bausteine erörtert werden (Abbildung 38).

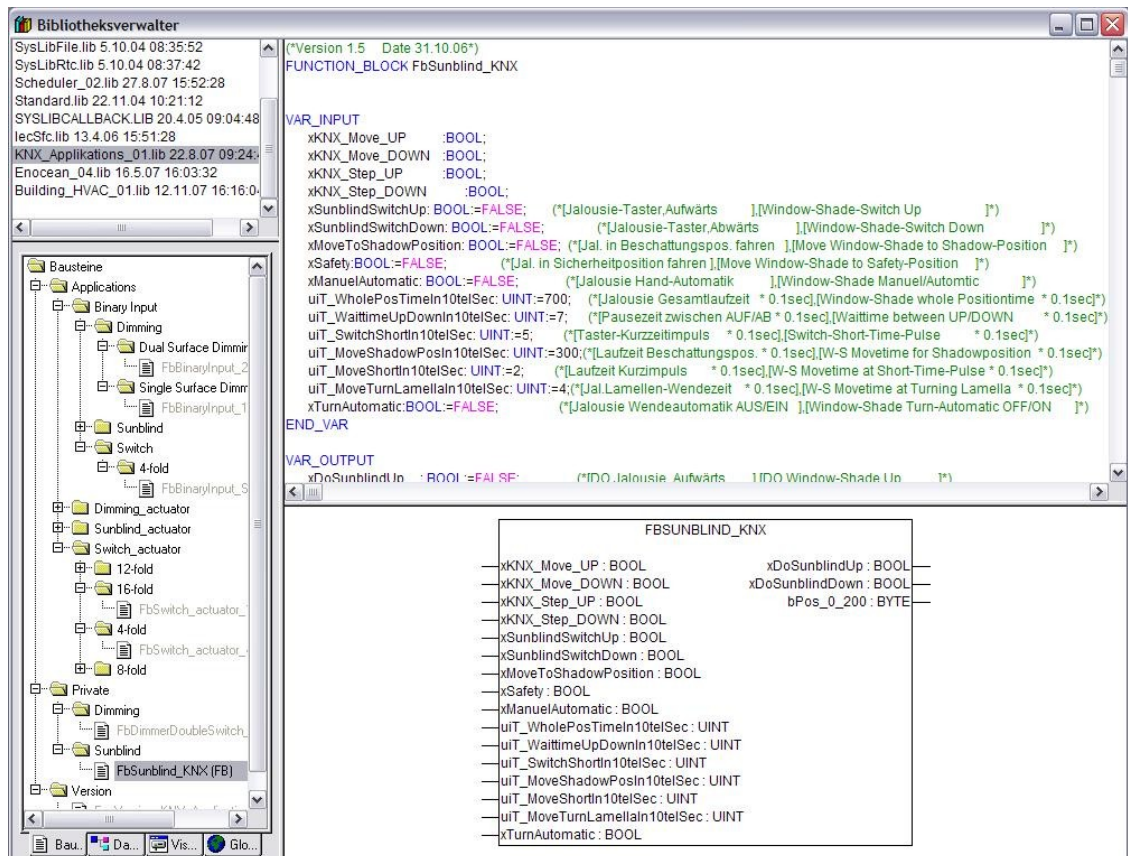


Abbildung 38: WAGO FbSunblind_KNX im Bibliotheksverwalter

Diese beiden Bausteine werden aber dringend benötigt, um Jalousie- und Dimmfunktionen über die DPT-Bausteine zu realisieren, da die Jalousie- und Dimmerbaustein der Bibliothek „Gebäude_allgemein.lib“ nicht die erforderlichen Eingänge zur Verfügung stellen. Die Aufnahme und Erläuterung dieser Bausteine in die Dokumentation ist wünschenswert.

Nachdem die gewünschten Funktionalitäten programmiert sind und im „PLC_PRG“ aufgerufen, muss nun die KNX/EIB-Klemme konfiguriert werden. Dies geschieht nach dem in Abbildung 39 dargestellten Ablauf, der im folgenden detailliert erläutert wird.

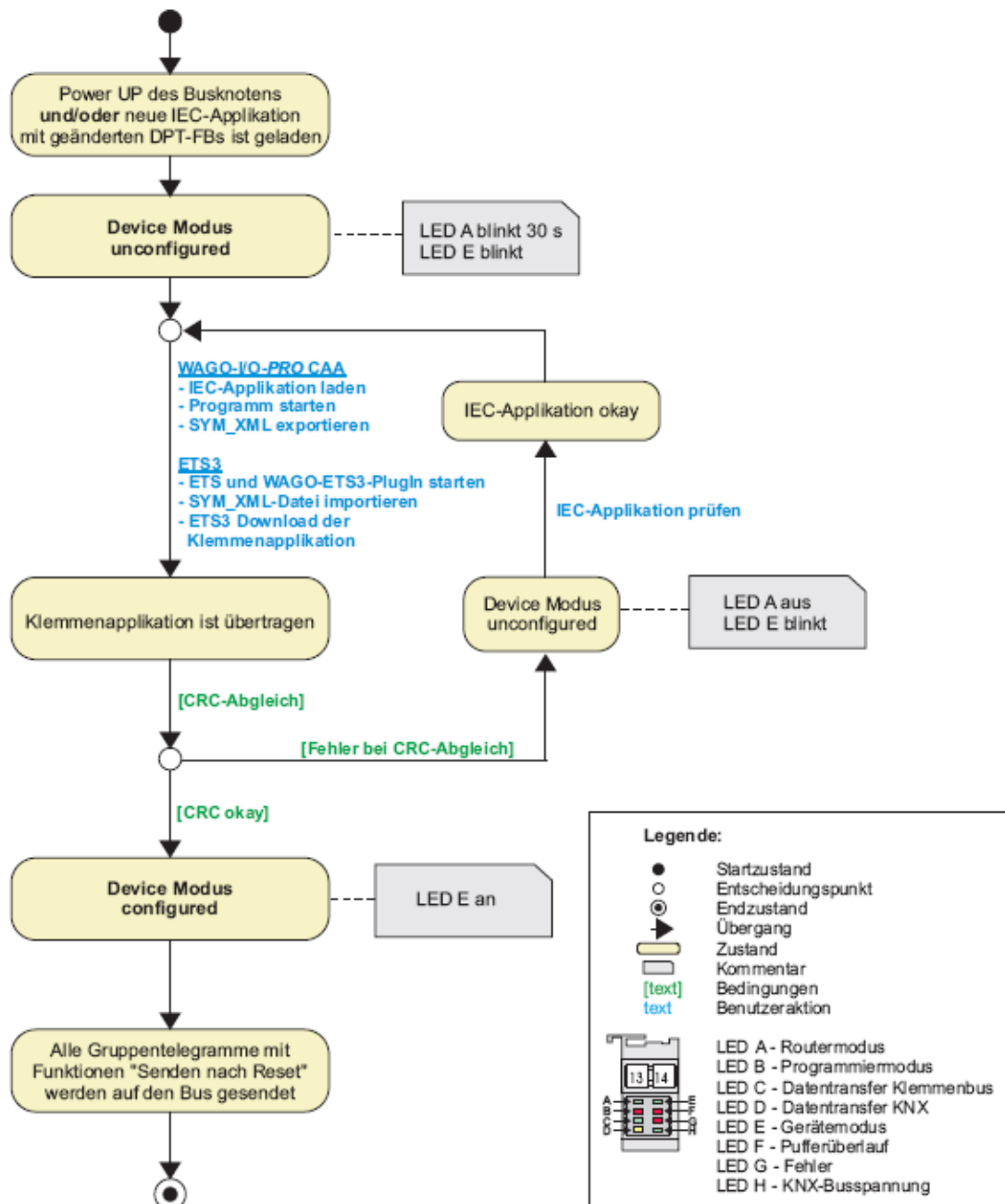


Abbildung 39: Abfolge zur Inbetriebnahme der WAGO KNX/EIB-Klemme

Dazu muss als erstes die Symboldatei mit den KNX-DPT-Variablen für das WAGO-ETS3-PlugIn erstellt werden. Hierzu ist die Konfiguration der CoDeSys im Menü „Projekt – Optionen...“ erforderlich.

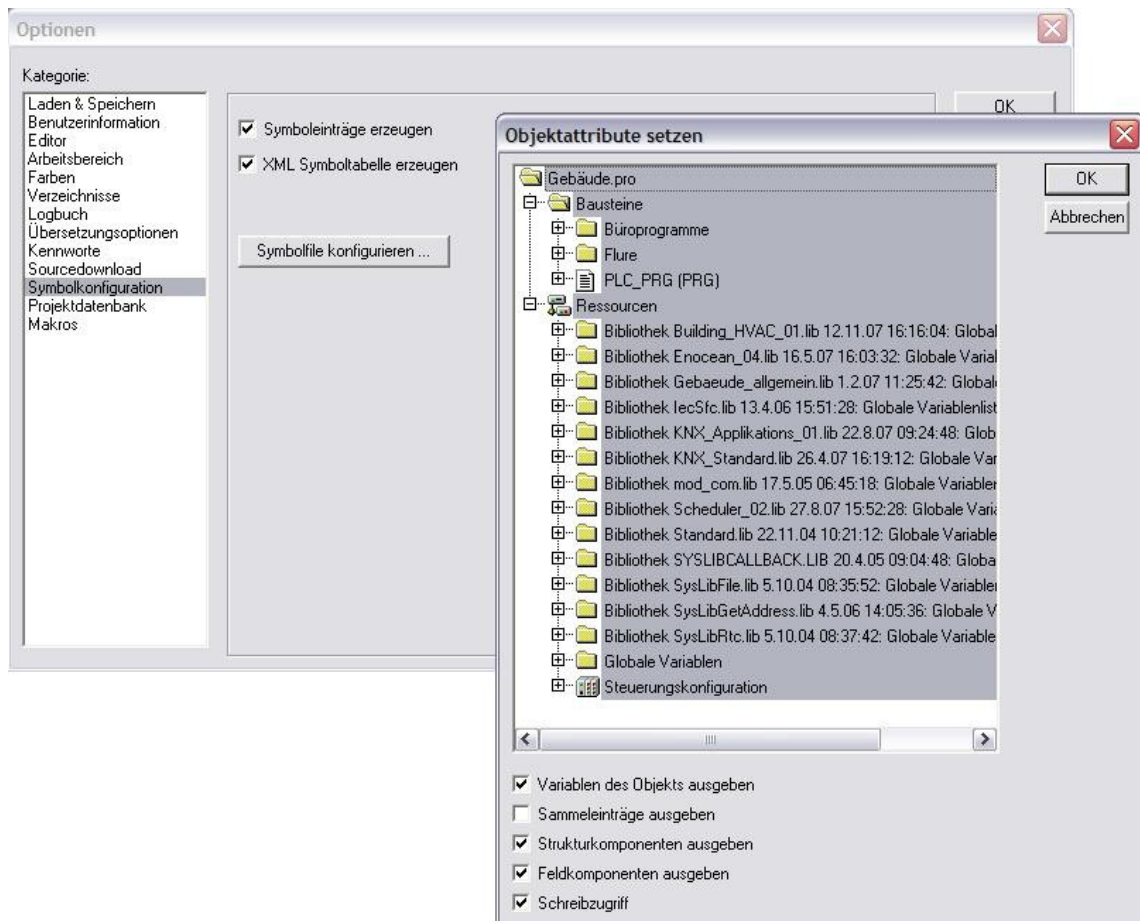


Abbildung 40: Erzeugen der Symboltabelle

In der Kategorie „Symbolkonfiguration“ muss die Option „XML Symboltabelle erzeugen“ ausgewählt werden. Mit einem Klick auf die Schaltfläche „Symbolfile konfigurieren...“ öffnet sich das Fenster „Objektattribute setzen“. In diesem Fenster können alle Projektbausteine oder gezielt Variableneinträge ausgewählt werden. Außerdem muss der Haken „Variablen des Objekts ausgeben“ gesetzt werden. Sollte dieser grau hinterlegt sein, dann sollte der Haken deaktiviert und neu aktiviert werden. Das Projekt muss nun übersetzt und gespeichert werden. Dabei wird eine .SYM_XML Datei erzeugt.

Das fertige Programm sollte jetzt in die Anlage geladen und gestartet werden.

Die programmierten KNX/EIB-Kommunikationsobjekte können jetzt mit den KNX/EIB-Gruppenadressen verknüpft werden. Hierzu ist die ETS3.0d oder höher und das WAGO-ETS3-PlugIn notwendig.

Nachdem die ETS3 gestartet ist, wird das WAGO-ETS3-PlugIn standardmäßig wie eine Herstellerdatenbank in die ETS3 importiert. Die KNX/EIB-Klemme steht jetzt als Gerät in der ETS3 zur Verfügung und kann in das ETS-Projekt eingefügt werden.

Während die KNX/EIB-Klemme dem ETS-Projekt hinzugefügt wird, startet automatisch die Installationsroutine des WAGO-ETS3-PlugIn. Sobald das Gerät hinzugefügt wurde, kann es wie jedes andere KNX/EIB-Gerät über „Parameter bearbeiten...“ parametrierbar werden. Es öffnet sich nun das Fenster „SYM-XML Datei für das Gerät auswählen“.

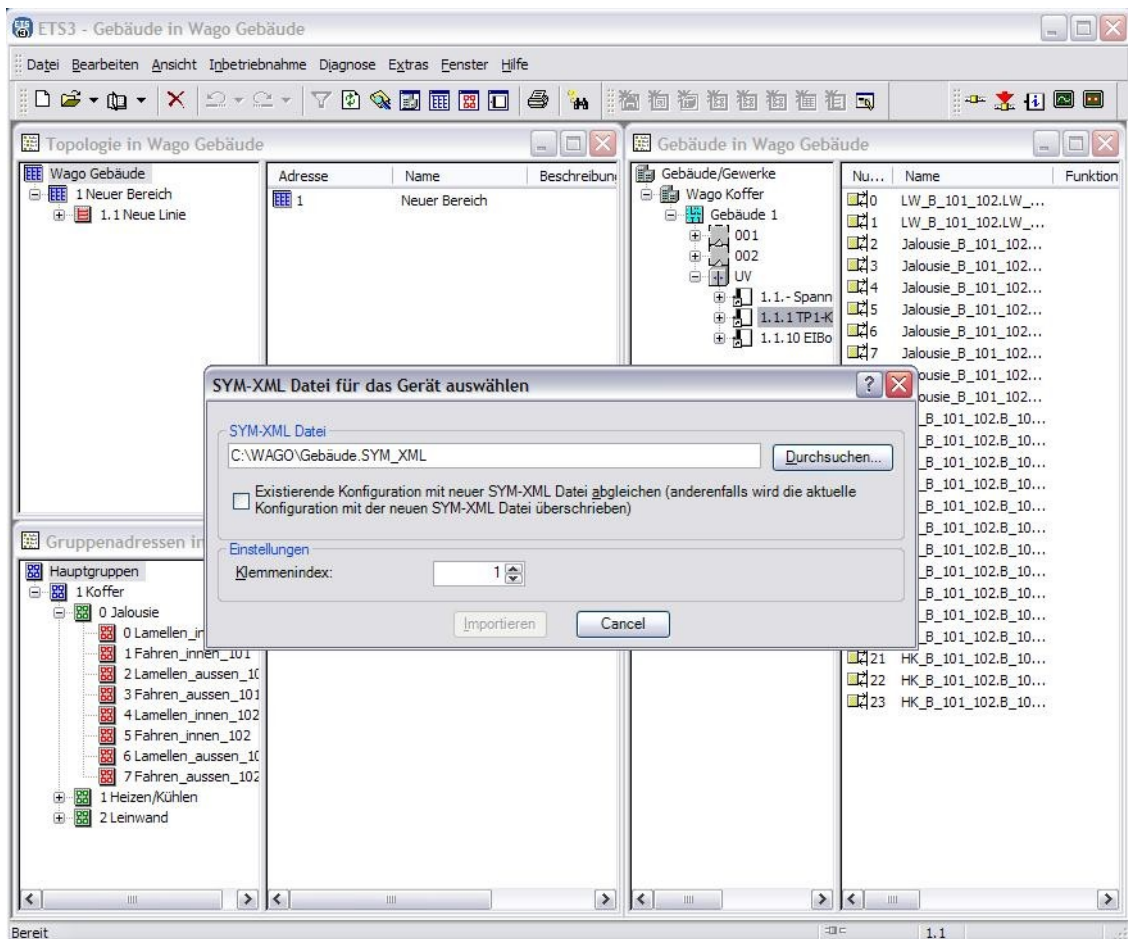


Abbildung 41: Auswahl der SYM-XML Datei

In diesem Fenster wird die zuvor erstellte SYM-XML Datei des Projektes angegeben und über die Schaltfläche „Importieren“ an das WAGO-ETS3-PlugIn übergeben.

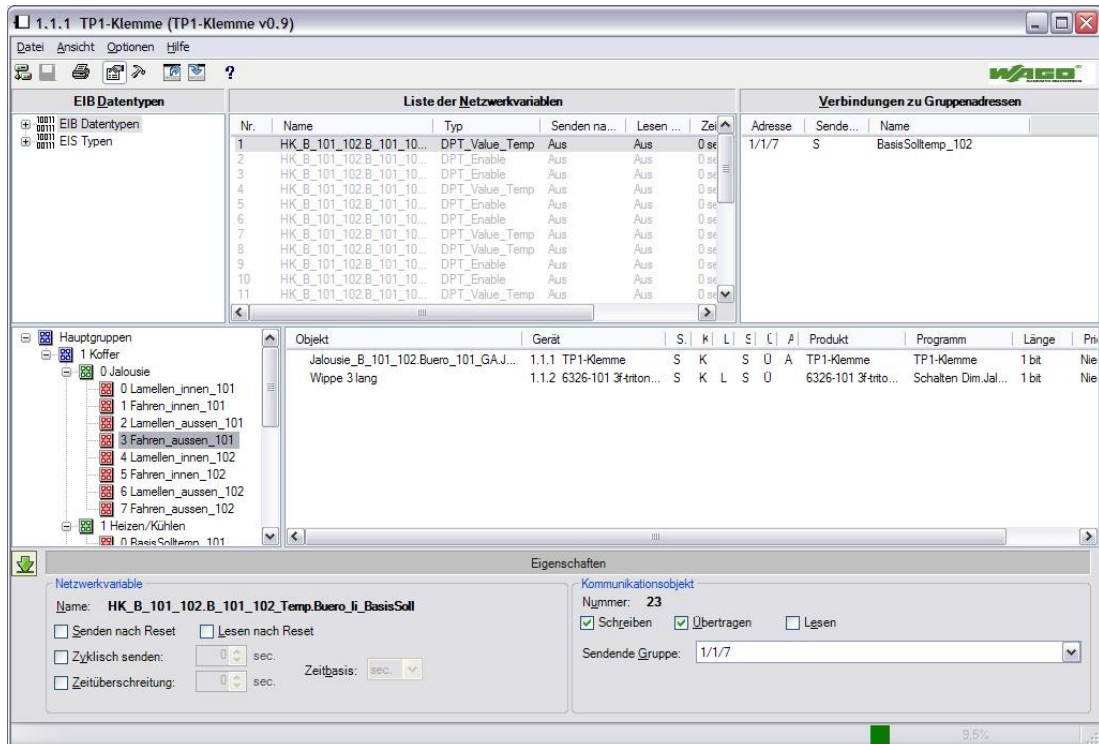


Abbildung 42: WAGO-ETS3-PlugIn

In diesem Fenster erscheinen jetzt alle DPT-Variablen aus dem IEC-61131 Projekt und können mit den KNX/EIB-Gruppenadressen verknüpft werden. Außerdem können hier noch die Eigenschaften der Objekte geändert werden. Sind alle Objekt mit den Gruppenadressen verknüpft wird die Konfiguration über „Datei- Speichern“ gesichert und das PlugIn beendet. Die KNX/EIB-Klemme wird nun mit den anderen KNX/EIB-Geräten aus der ETS3 heraus programmiert.

Nachdem die KNX/EIB-Klemme erfolgreich programmiert wurde, erlischt die rote LED an der KNX/EIB-Klemme. Sollte dies nicht der Fall sein, so stimmt eventuell das Projekt in der Anlage mit der SYM-XML Datei der KNX/EIB-Klemme nicht überein.

Nach der Inbetriebnahme wurden die programmierten Funktionalitäten nun ausgiebig getestet.

Hierbei stellte sich heraus, dass das Objekt „aktueller Sollwert“ des Busch-Jaeger Triton sich nicht von der WAGO-SPS beschreiben lässt. Außerdem war festzustellen, dass der Status eines KNX/EIB-Objektes, wenn dieses durch die SPS verändert wurde, nicht mit dem aktuellen Wert zurückgesendet wird, wodurch in der SPS ein falscher Statuswert des KNX/EIB-Objektes vorliegt.

5.2.5 Zwischenfazit zur WAGO KNX/EIB-Klemme

Die im Pflichtenheft definierten Bürofunktionalitäten Beleuchtung, Jalousiesteuerung, Heißwasserbereiter, Leinwand im Konferenzraummodus und die Flurbeleuchtung ließen sich mit den vorhandenen Bausteinen in den WAGO-Bibliotheken problemlos und komfortabel umsetzen. Die vorhandenen Dokumentationen sind bis auf die beiden bereits erwähnten KNX-Bausteine übersichtlich und verständlich geschrieben und daher sehr hilfreich. Dies ist aber auf Grund der großen Anzahl Bausteine auch zwingend notwendig.

Die Umsetzung der Raumtemperaturregelung war jedoch nicht wie gefordert möglich. Dies liegt daran, dass in den WAGO KNX-Bibliotheken kein Baustein zur Verfügung steht, der es ermöglicht eine KNX/EIB-Gruppenadresse zu zwingen seinen momentanen Status zu senden.

Wenn sich der verwendete Busch-Jaeger Triton mit Raumtemperaturregler im Komfortbetrieb befindet und zur vorgegeben Zeit von der SPS in den Nachtbetrieb geschaltet wird, musste dieser vorher von der SPS in den Standbybetrieb gesetzt werden. Wird der Nachtbetrieb wieder beendet, werden KNX/EIB-Sensoren mit Raumtemperaturregler automatisch in den Standbybetrieb gesetzt. Diese senden aber ihren neuen Betriebsstatus nicht auf den Bus, da dieser Wert über den Bus empfangen und nicht am KNX/EIB-Sensor verändert wurde. Da der DPT-Baustein aber immer noch den Wert des zuletzt gesendeten Telegramms hat, also Komfortbetrieb, stimmen die Werte am KNX/EIB-Sensor und in der SPS nicht überein.

Dieses Problem kann aber gelöst werden, wenn von WAGO ein Baustein zur Verfügung gestellt wird, der eine KNX/EIB-Gruppenadresse zwingt, ihren momentanen Wert auf den Bus zu senden, so dass der Wert des entsprechende DPT-Baustein aktualisiert wird.

Außerdem war es nicht möglich im Konferenzraum-Modus eine Erhöhung der Raumtemperatur an einem Busch-Jaeger Triton in den zweiten im Raum vorhandenen Busch-Jaeger Triton zu schreiben, da das Objekt „aktueller Sollwert“ des Triton nicht über den Bus beschrieben werden kann.

Es handelt sich hierbei aber um ein Problem des Triton und ist somit auch mit keinem anderen Automatisierungssystem lösbar.

Dieses Problem ist nur mit einer Änderung des Applikationsprogramms des Triton zu lösen.

Mit dem Siemens Raumtemperaturregler UP 231/2 DELTA profil wäre es zwar möglich die Temperaturerhöhung in den zweiten vorhandenen Siemens UP 231/2 DELTA profil zu schreiben, dieser kann aber nicht zusammen mit der WAGO-SPS betrieben werden, da sich die vorhandenen Siemens UP 231/2 DELTA profil nicht mit der ETS3.0d programmieren lassen.

5.3 Umsetzung der Funktionen mit dem Beckhoff CX9001-1001

Zuerst muss das „Beckhoff TwinCat System “ installiert werden. Nach erfolgreicher Installation müssen noch alle benötigten Zusatzbibliotheken nach „C:\TwinCAT\Plc\Lib” kopiert werden.

5.3.1 Verbindungsaufbau zur Beckhoff-SPS

Um die Beckhoff-SPS programmieren zu können, muss der SPS eine IP-Adresse zugewiesen werden. Dies sollte möglichst eine statische IP-Adresse sein, damit die Anlagen nach einem Reset immer über die gleiche IP-Adresse angesprochen werden kann.

Zuerst muss die SPS aber an einen DHCP-Server angeschlossen werden, um eine IP-Adresse zu bekommen. Mit dem „TwinCat System Manager“ kann nun über „Zielsystem wählen...“ im Fenster „Wähle Zielsystem“ über die Schaltfläche „Suchen (Ethernet)“ im Fenster „Add Route Dialog“ nach allen im Ethernet verfügbaren Anlagen gesucht werden.

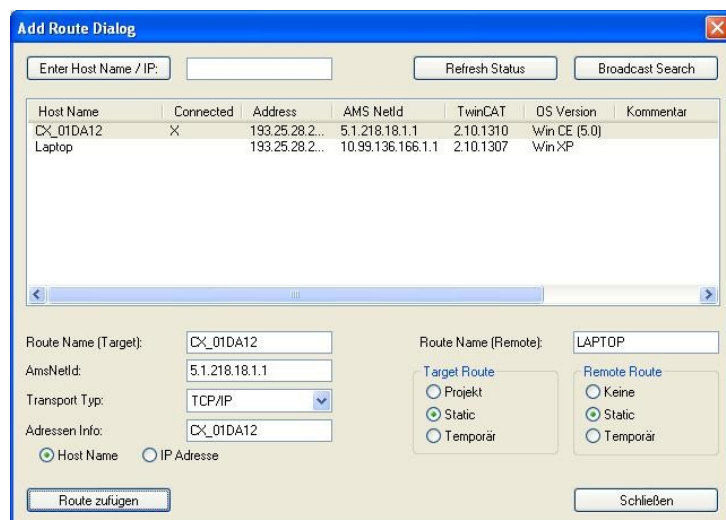


Abbildung 43: TwinCAT System Manager „Add Route Dialog“

Nach Betätigung der Schaltfläche „Broadcast Search“ sucht die Software nach Anlagen, die dann über „Route zufügen“ mit der Software verbunden werden.

Im Fenster „Wähle Zielsystem“ kann nun die neue Anlage für das Projekt ausgewählt werden.

Im TwinCAT System Manager kann jetzt über die Karteikarte „CX Settings“ der SYSTEM-Konfiguration eine statische IP-Adresse vergeben werden und über die Schaltfläche „Übernehmen“ in die Anlage geschrieben werden.

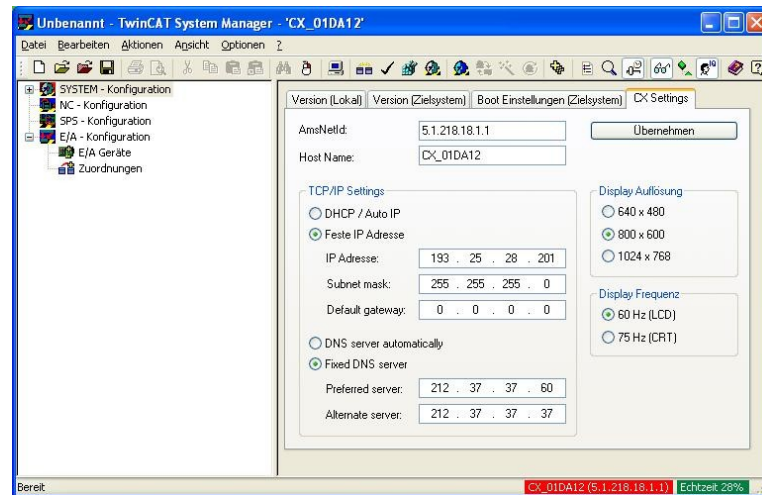


Abbildung 44: Einstellung der IP-Adresse des CX

Die SPS ist nun immer unter dieser IP-Adresse im Ethernet verfügbar.

5.3.2 Allgemeine Programmierung der Beckhoff-SPS

Nachdem die Beckhoff-SPS eine IP-Adresse bekommen hat, kann nun das Programm für die SPS geschrieben werden.

Um das SPS-Programm zu schreiben muss die „TwinCat PLC Control“ gestartet werden. Da die TwinCat Software auf der CoDeSys Software basiert, ist die Erstellung eines Projektes mit „TwinCat PLC Control“ in wesentlichen identisch zur „WAGO-I/O-Pro CAA Software (CoDeSys)“. Da die Steuerungskonfiguration der Beckhoff-SPS später mit dem „TwinCat System Manager“ durchgeführt wird, kann in der „Steuerungskonfiguration“ auf der Karteikarte „Ressourcen“ lediglich die Art der Zielplattform verändert werden. Alle zusätzlichen Bibliotheken werden wie bereits bei der WAGO-SPS beschrieben mit dem „Bibliotheksverwalter“ hinzugefügt.

Der einzige Unterschied bei der Programmierung ist die Definition der Ein-/Ausgangsvariablen der Klemmen. Während bei der WAGO-SPS die Variablennamen in der „Steuerungskonfiguration“ direkt an den Klemmen vergeben wurden, werden die Variablen der Ein-/Ausgänge bei Beckhoff im Programm deklariert und im Fenster „Variablendeklaration“ im Feld „Adresse“ mit „%I*“ für die Eingänge und „%Q*“ für die Ausgänge versehen. Nach der Fertigstellung des Programms werden die Kanälen der Klemmen im „TwinCat System Manager“ mit den Variablen verknüpft.

Daher ist es bei der Beckhoff-SPS möglich, die Ein- und Ausgangsvariablen sowohl als globale Variablen für den Zugriff im gesamten Projekt, als auch als lokale Variablen zur Verwendung in nur einem Teilprogramm, zu deklarieren.



Abbildung 45: Deklaration einer Eingangsvariable bei der Beckhoff-SPS

Um eine Kommunikation mit der Anlage aufzubauen, geschieht dies bei der Beckhoff-SPS nicht über „Online – Kommunikationsparameter...“, sondern über „Online – Auswahl des Zielsystem...“. Im Fenster „Zielsystem Auswahl“ muss nur die zu programmierende SPS ausgewählt werden. Hier stehen alle Analgen zur Verfügung die vorher mit dem „TwinCAT System Manager“ verbunden wurden.

Das fertig Programm muss nun über „Projekt – Alles Übersetzen“ übersetzt und gespeichert werden.

Über den „TwinCAT System Manager“ kann nun die Klemmen-Konfiguration der Beckhoff-SPS ausgelesen werden. Dazu muss die SPS über „Aktionen – Starten/Restarten von TwinCAT in Konfig-Modus“ in den Konfigurationsmodus gesetzt werden.

Über einen Rechtsklick auf „E/A Geräte“ unter „E/A-Konfiguration“ kann im Kontextmenü „Geräte Suchen...“ die Anlage nach den gesteckten Klemmen untersucht werden. Im Gegensatz zu „WAGO-I/O-Pro CAA Software (CoDeSys)“ geschieht dies im „TwinCAT System Manager“ automatisch.

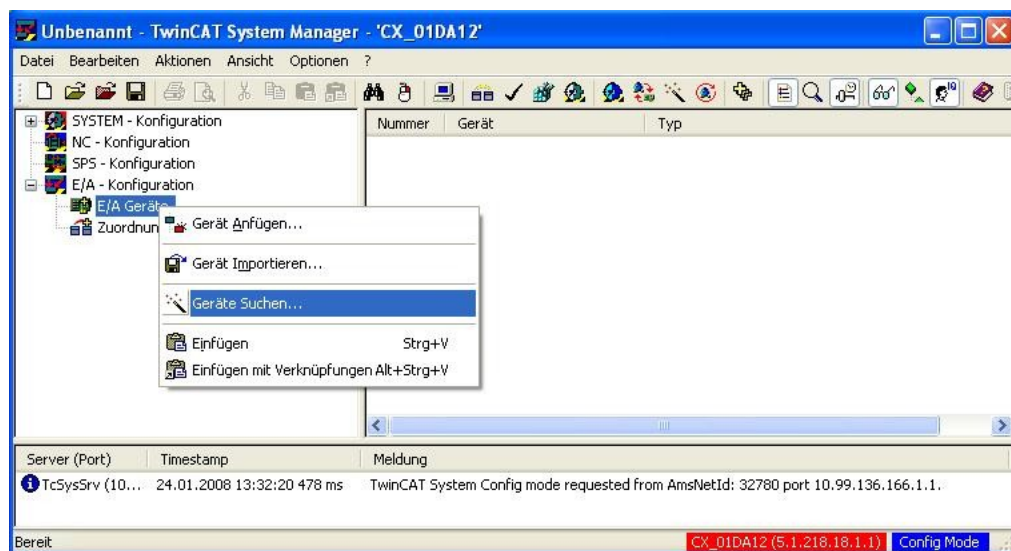


Abbildung 46: Auslesen der Klemmenkonfiguration in TwinCAT

Nachdem die Software die gesteckten Klemmen automatisch ausgelesen hat, müssen einige Klemmen eventuell noch in einen kompatiblen Typ geändert werden. Dies geschieht über das Kontextmenü der jeweiligen Klemme.

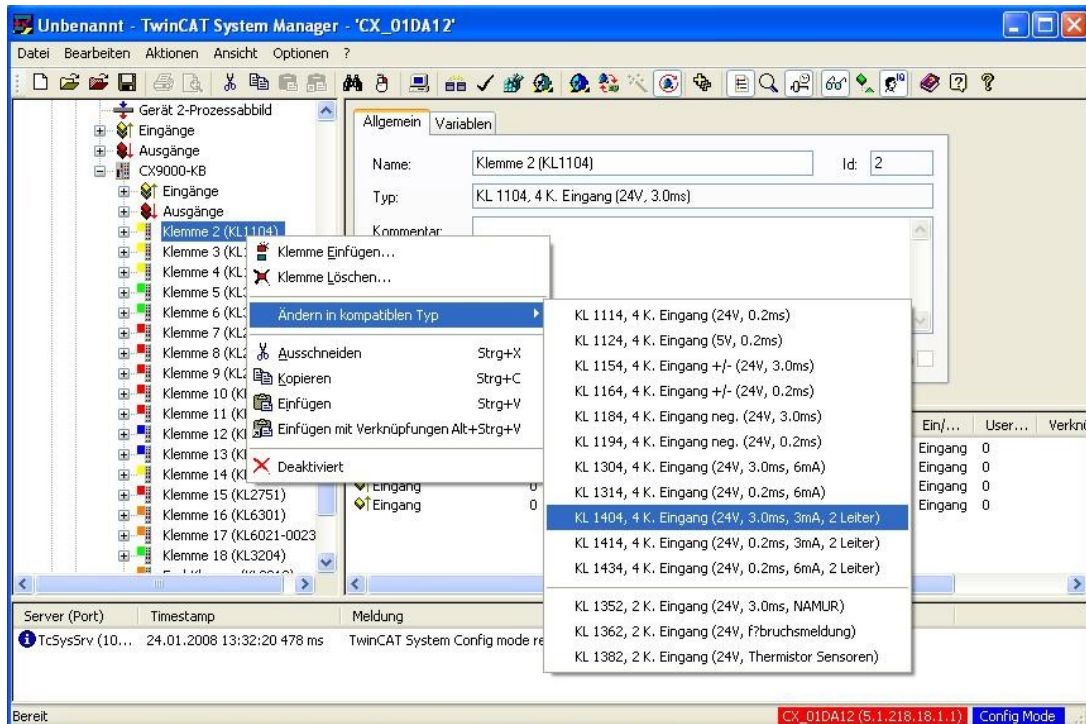


Abbildung 47: Einstellen der Klemmen in TwinCAT

Unter „Ändern in kompatiblen Typ“ kann aus einer Liste eine passende Klemme mit demselben Profil ausgewählt werden. Somit ist gewährleistet, dass keine falschen Klemmen eingefügt werden können und dass diese auch an der richtigen Stelle in der Anlagenkonfiguration stehen.

Die Konfiguration der Klemmen kann aber auch vollständig manuell erfolgen.

Die Konfiguration sollte jetzt gespeichert werden.

Als nächstes werden noch die Ein-/Ausgangsvariablen mit den Klemmen verknüpft. Hierzu muss im „TwinCat System Manager“ unter „SPS-Konfiguration“ und „SPS Projekt Anfügen...“ das mit „TwinCat PLC Control“ geschriebene Projekt eingefügt werden.

Nun können an den Klemmen die Variablen des SPS-Projektes den Kanälen zugewiesen werden.

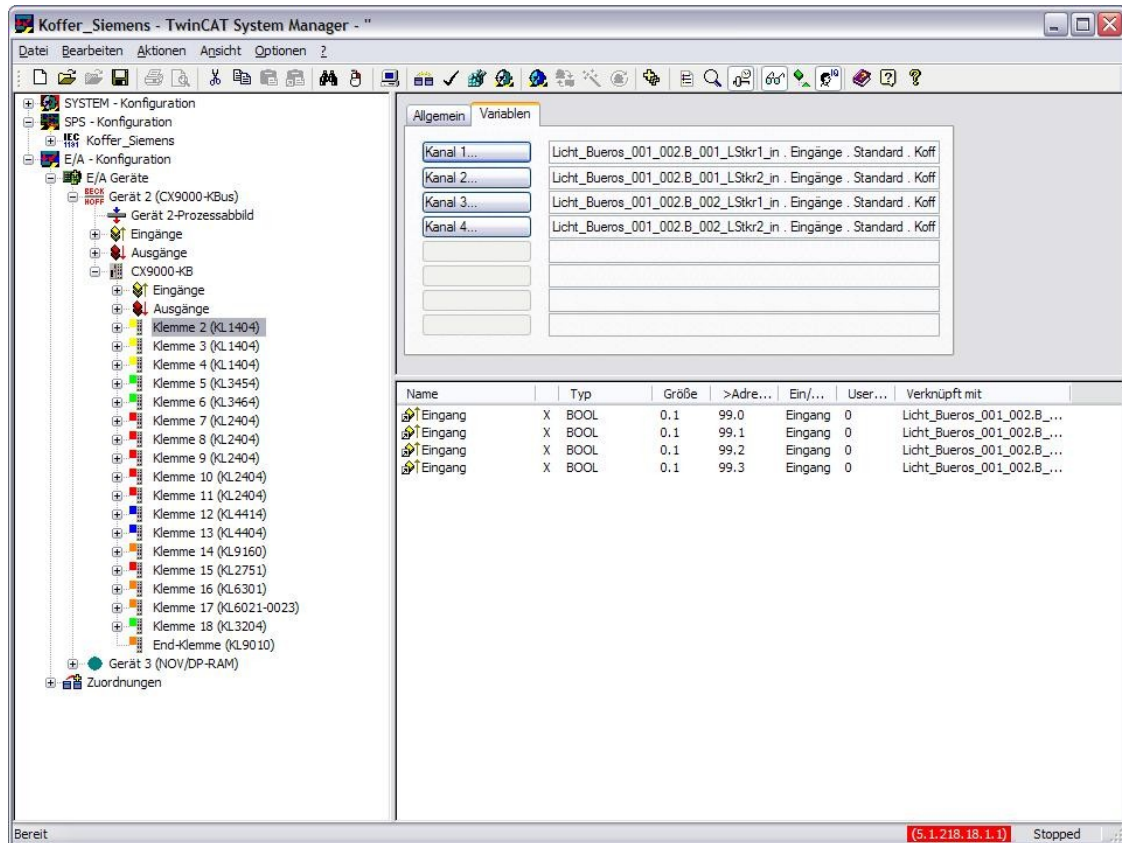


Abbildung 48: Variablenzuweisung im „TwinCat System Manager“

Die Anlage muss anschließend über den Menüpunkt „Aktionen – Starten/Restarten von TwinCAT in Konfig-Modus“ in den Konfigurationsmodus versetzt werden.

Über den Menüpunkt „Aktionen“ – „Zuordnungen erzeugen“ müssen jetzt die Zuordnungen erstellt werden und durch „Aktionen“ – „Überprüfen der Konfiguration“ sollte die Konfiguration auf mögliche Fehler überprüft werden. Hier kann es passieren das nicht genutzte Geräte als Fehler interpretiert werden. Diese sollten daher über einen Rechtsklick auf das betreffende Gerät deaktiviert werden.

Danach kann über „Aktionen“ – „Aktiviert Konfiguration...“ die Konfiguration aktiviert und die Anlage neu gestartet werden.

Da die Variablen den Ein-/Ausgängen jetzt zugeordnet sind, kann mit „TwinCAT PLC Control“ über den Menüpunkt „Online – Einloggen“ das fertige Projekt in die Anlage geladen werden. Wie bei der WAGO-SPS muss die Anlage ebenfalls gestartet und ein Bootprojekt geladen werden.

5.3.3 Implementierung der EnOcean-Klemme von Beckhoff

Zur Kommunikation mit EnOcean-Geräten über die EnOcean-Funk-Klemme KL6021-0023 mit Wireless-Adapter KL6023 werden in der Bibliothek „TcEnOcean.lib“ spezielle Funktionsbausteine zur Verfügung gestellt. Jede am Klemmenbus verfügbare EnOcean-Funk-Klemme benötigt zur Kommunikation mit der IEC 61131-Applikation einen Baustein der die gesamte Kommunikation verwaltet.

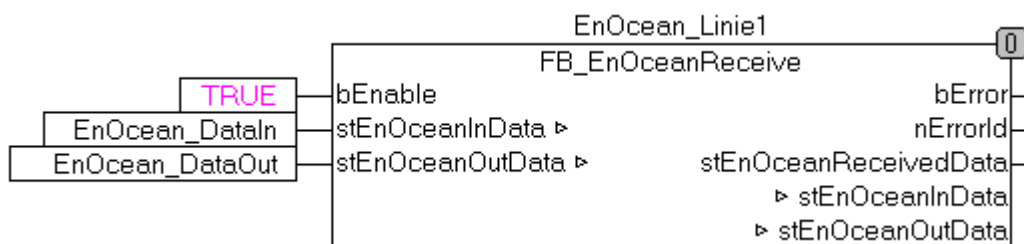


Abbildung 49: Beckhoff FB_EnOceanReceive-Baustein

Der Eingang „bEnable“ des Bausteins „FB_EnOceanReceive“ muss auf TRUE gesetzt werden, um den Baustein und somit die Kommunikation mit der Klemme zu aktivieren. An den Eingängen „stEnOceanInData“ und „stEnOceanOutData“ werden die Ein- und Ausgangsvariable angelegt, über die das IEC-Programm mit der EnOcean-Funk-Klemme kommuniziert. Diese Variablen werden später im „TwinCAT System Manager“ mit der Klemme verknüpft.

Der Baustein „FB_EnOceanReceive“ empfängt alle gesendeten Signale und leitet diese über den Ausgang „stEnOceanReceivedData“ an die EnOcean-Funktionbausteine weiter.

Pro gesteckter EnOcean-Funk-Klemme darf dieser Baustein nur einmal verwendet werden.

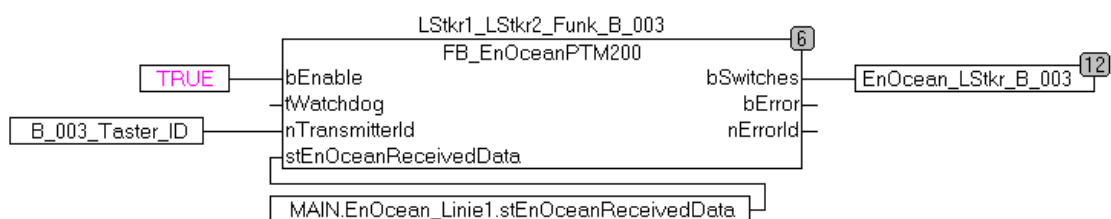


Abbildung 50: Beckhoff FB_EnOceanPTM200

Zur Auswertung der EnOcean-Sensoren stehen Funktionsbausteine zur Verfügung. Der Funktionsbaustein muss über den Eingang „bEnable“ aktiviert und über den Eingang „stEnOceanReceivedData“ mit dem Ausgang „stEnOceanReceivedData“ des zugehörigen „FbEnOceanReceive“-Bausteins verbunden werden, um die Signale zu empfangen.

Am Eingang „nTransmitterId“ wird die Transmitter ID des EnOcean Sensors eingetragen.

Am Ausgang des Bausteins werden die Signale des Sensors als Array vom Typ BOOL ausgegeben und können so problemlos im Projekt weiter verarbeitet werden.

Die verfügbaren Bausteine sind alle detailliert in der Dokumentation der EnOcean-Funk-Klemme beschrieben.

Um die EnOcean-Funk-Klemme in Betrieb nehmen zu können, müssen im „TwinCAT System Manager“ noch die Ein- und Ausgangsvariable des „FB_EnOceanReceive“-Bausteins mit der Klemme verknüpft werden. Dazu muss das Projekt neu eingelesen werden, damit die Variablen für die EnOcean-Klemme zur Verfügung stehen.

Um die Eingangsvariable mit der „KL6021-0023“ zu verknüpfen, muss unter „SPS-Konfiguration“ die Liste der Variablen geöffnet werden. Sobald die Liste sichtbar ist, muss die Eingangsvariable für die „KL6021-0023“ markiert werden. Über den Menüpunkt „Verknüpfung Ändern...“ des Kontextmenü oder über die Schaltfläche „Verknüpft m.“ auf der Karteikarte „Variable“ öffnet sich das Fenster „Variablenverknüpfung ...“.

Um die Variable verknüpfen zu können, muss erst noch ein Haken vor „Alle Typen“ unter „Zeige Variablen Typen“ und ein Haken vor „Kontinuierlich“ unter „Offsets“ gesetzt werden.

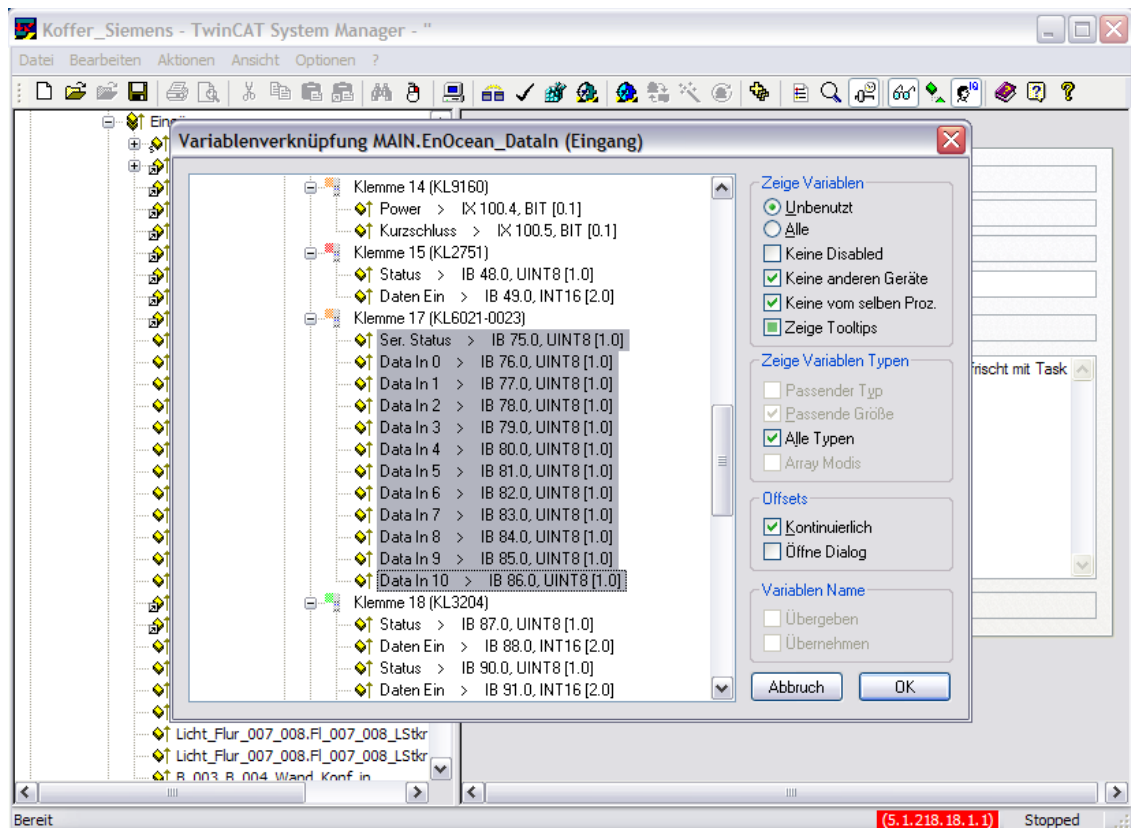


Abbildung 51: Verknüpfung der Variablen der KL6021-0023

Nun müssen noch alle Eingänge der KL6021-0023 („Ser. Status „ bis „Data In 10“) markiert werden. Über die Schaltfläche „OK“ wird die Variablenverknüpfung hinzugefügt.

Für die Ausgangsvariable muss dieses Vorgehen auf dieselbe Art und Weise noch einmal wiederholt werden.

Die Anlage muss nun wieder in den Konfigurationsmodus versetzt und die neue Konfiguration aktiviert werden. Danach kann die Anlage wieder in der Run-Modus gesetzt und mit „TwinCAT PLC Control“ das Projekt wie üblich in die Anlage geladen und gestartet werden.

5.3.4 Implementierung der KNX/EIB-Klemme von Beckhoff

Zur Kommunikation mit dem KNX/EIB-Bussystem über die KNX/EIB-Klemme KL6301 von Beckhoff sind die von Beckhoff in der Bibliothek TcKL6301.lib zur Verfügung gestellten Funktionsbausteine notwendig. Für jede vorhandene KNX/EIB-Klemme muss ein Funktionsbaustein „KL6301“ aufgerufen werden. Dieser Baustein konfiguriert, startet, überwacht und übernimmt die Kommunikation mit der KNX/EIB-Klemme.

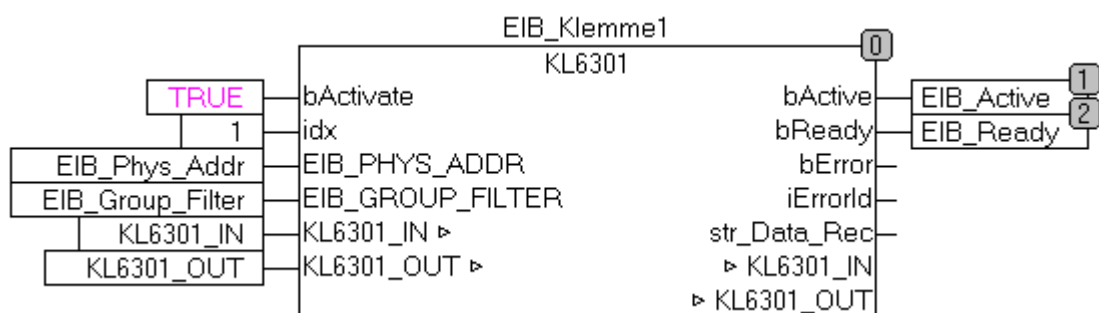


Abbildung 52: Beckhoff Funktionsbaustein „KL6301“

Sollten mehrere KNX/EIB-Klemmen an einer Anlage verfügbar sein, muss für jede Klemme ein KL6301-Baustein aufgerufen werden und mit einer Klemmen-Nummer am Eingang „idx“ versehen werden (erste Klemme => 1, zweite Klemme => 2 usw.).

Es sind jedoch maximal 64 KNX/EIB-Klemmen in einem Projekt möglich.

Um mit dem KNX/EIB-Bus kommunizieren zu können, benötigt die KNX/EIB-

Klemme, da sie am KNX/EIB-Bus ein KNX/EIB-Gerät ist, wie jedes andere

KNX/EIB-Gerät eine physikalische Adresse, die im KNX/EIB-Netzwerk einmalig sein muss.

Diese physikalische Adresse wird am Eingang „EIB_PHYS_ADDR“ des Bausteins über eine Variable vorgegeben und nicht mit der ETS programmiert. Die Beckhoff KNX/EIB-Klemme ist somit mit allen KNX/EIB-Geräten nutzbar, unabhängig davon mit welcher ETS Version sie programmiert wurden.

Bevor die KNX/EIB-Klemme jedoch mit dem Datenaustausch am KNX/EIB-Bus

beginnen kann, müssen EIB-Gruppenfilter parametrisiert werden. Die KNX/EIB-

Klemme empfängt später nur die Telegramme der KNX/EIB-Gruppenadressen, die in diesen Filtern eingetragen sind.

Alle weiteren Telegramme werden von der KNX/EIB-Klemme ignoriert.

Es stehen maximal vier Filter zur Verfügung, mit einer Länge der Filtereinträge von 0 bis 63. Es können pro Filter somit 64 Gruppenadressen freigeschaltet werden, womit bei vier Filtern eine Anzahl von 256 Gruppenadressen freigeschaltet werden können. Hierbei ist zu beachten, dass jeder Filter immer nur als Block, z.B.: 1/1/0...63 oder 4/2/0...63, freigeschaltet werden kann und nicht 64 einzelne Gruppenadressen aus unterschiedlichen Haupt- oder Mittelgruppen.

Die Vergabe der Gruppenadressen im ETS-Projekt sollte auf Grund dieser Beschränkung von Anfang an gut geplant werden.

Um die Filter zu aktivieren, legt man am Eingang „EIB_GROUP_FILTER“ eine Variable an und parametriert diese im Fenster „Variablen Deklaration“.

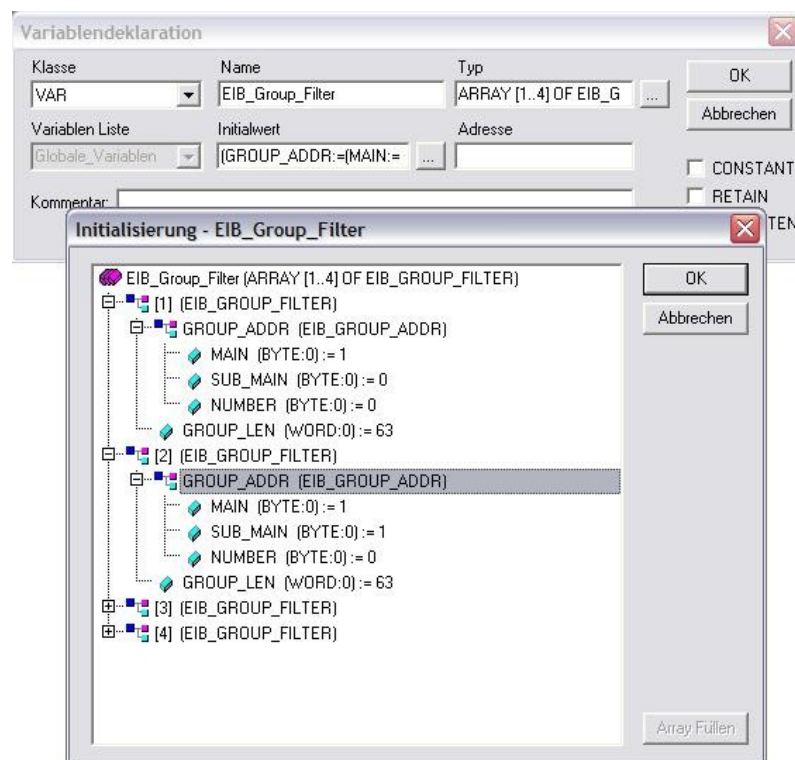


Abbildung 53: Filtereinträge für die KL6301

Nachdem im Feld „Typ“ „ARRAY [1..4] OF EIB_GROUP_FILTER“ eingetragen wurden, kann über die Schaltfläche „...“ neben dem Feld „Initialwert“ das Fenster „Initialisierung – EIB_Group_Filter“ geöffnet werden. In diesem Fenster lassen sich die Filter komfortabel eintragen.

Die „TcKL6301.lib“ stellt außerdem noch ein Baustein „EIB_READ_SEND“ zur Verfügung, der es ermöglicht, eine KNX/EIB-Gruppenadresse aufzufordern ihren aktuellen Wert zu übermitteln. Dies wird zum Beispiel dringend benötigt, um die Betriebsart eines KNX/EIB-Raumtemperaturreglers abzufragen, da dieser eine Statusänderung vom Komfort- in den Standbybetrieb, herbeigeführt durch die Nachtabsenkung, nicht auf den KNX/EIB-Bus sendet.

Alle Bausteine können somit wie Ein- und Ausgänge behandelt werden und problemlos im Programm mit allen Funktionsbaustein kombiniert werden.

Eine ausführliche Beschreibung aller Bausteine, die Bibliothek „TcKL6301.lib“, sowie Beispiele sind in der Dokumentation der KNX/EIB-Klemme zu finden.

Diese Beschreibung ist für die Nutzung der Bibliothek auch unbedingt erforderlich.

Um die KNX/EIB-Klemme in Betrieb nehmen zu können, müssen im „TwinCAT System Manager“ noch die Ein- und Ausgangsvariable des „KL6301“-Bausteins mit der Klemme verknüpft werden. Dazu muss das Projekt neu eingelesen werden, damit die Variablen für die KNX/EIB-Klemme zur Verfügung stehen.

Um die Eingangsvariable mit der „KL6301“ zu verknüpfen, muss unter „SPS-Konfiguration“ die Liste der Variablen geöffnet werden. Sobald die Liste sichtbar ist, muss die Eingangsvariable für die „KL6301“ markiert werden. Über den Menüpunkt „Verknüpfung Ändern...“ des Kontextmenü oder über die Schaltfläche „Verknüpft m.“ auf der Karteikarte „Variable“ öffnet sich das Fenster „Variablenverknüpfung ...“. Um die Variable verknüpfen zu können, muss erst noch ein Haken vor „Alle Typen“ unter „Zeige Variablen Typen“ und ein Haken vor „Kontinuierlich“ unter „Offsets“ gesetzt werden.

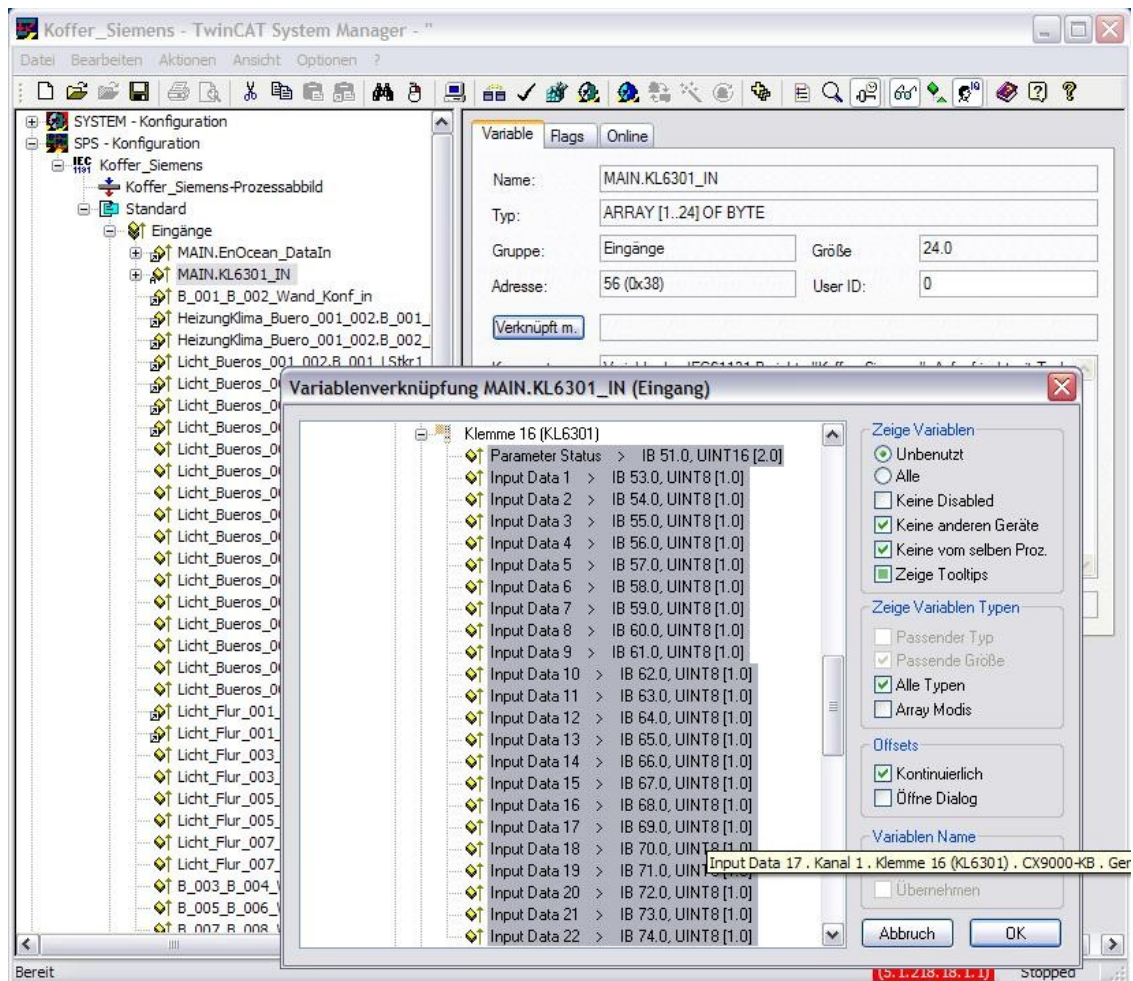


Abbildung 55: Verknüpfung der Variablen der KL6301

Nun müssen noch alle Eingänge der KL6301 („Parameter Status „ bis „Input Data 22“) markiert werden. Über die Schaltfläche „OK“ wird die Variablenverknüpfung hinzugefügt.

Für die Ausgangsvariable muss dieses Vorgehen auf die elbe Art und Weise noch einmal wiederholt werden.

Die Anlage muss nun wieder in den Konfigurationsmodus versetzt und die neuen Konfiguration aktiviert werden. Danach kann die Anlage wieder in der Run-Modus gesetzt und mit „TwinCAT PLC Control“ das Projekt wie üblich in die Anlage geladen und gestartet werden.

5.3.5 Zwischenfazit zur Beckhoff KNX/EIB-Klemme

Mit der Beckhoff KNX/EIB-Klemme ließen sich alle im Pflichtenheft definierten Bürofunktionalitäten mit dem Siemens Raumtemperaturregler UP 231/2 DELTA profil vollständig und problemlos umsetzen.

Mit dem Busch-Jaeger Triton mit Raumtemperaturregler war, auf Grund der bereits beschriebenen Probleme mit dem Triton, an der Beckhoff KNX/EIB-Klemme ebenfalls keine vollständige Umsetzung des Pflichtenheftes möglich.

6. Gegenüberstellung der KNX/EIB-Klemmen von WAGO und Beckhoff

Um die WAGO KNX/EIB-Klemme in Betrieb zu nehmen, ist eine ETS in der Version 3.0d oder höher notwendig. Für die Beckhoff KNX/EIB-Klemme dagegen ist keine ETS notwendig. Es ist daher im Gegensatz zur WAGO KNX/EIB-Klemme möglich, diese in älteren KNX/EIB-Anlagen die mit der ETS2 programmiert wurden zu integrieren. Die WAGO KNX/EIB-Klemme ist nur in neuen KNX/EIB-Anlagen, die mit der ETS3d oder höher programmiert wurden, einsetzbar.

Durch die Verknüpfung der KNX/EIB-Gruppenadressen mit der WAGO KNX/EIB-Klemme in der ETS3 können die KNX/EIB-Gruppenadressen hier etwas flexibler vergeben werden als bei der Beckhoff KNX/EIB-Klemme, da hier nur vier Filter zur Verfügung stehen, die als Block von je 64 KNX/EIB-Gruppenadressen genutzt werden müssen.

Für den Fall, dass nicht alle KNX/EIB-Gruppenadressen in der SPS benötigt werden, ist eine genaue Planung der KNX/EIB-Gruppenadressen erforderlich, um eine vollständige Ausschöpfung der möglichen KNX/EIB-Gruppenadressen zu erreichen. Außerdem ist man bei der Beckhoff KNX/EIB-Klemme auf das 3-Ebenen-Gruppenadressenkonzept der ETS festgelegt, da es bei der Klemme keine Möglichkeit gibt zwischen 2 oder 3 Ebenen zu wählen.

Wird der KNX/EIB, wie in diesem Projekt, aber nur als reines Subsystem genutzt, werden somit auch alle KNX/EIB-Gruppenadressen in der SPS benötigt. Da an jeder SPS bis zu 64 KNX/EIB-Klemmen eingesetzt werden können, können somit 16384 KNX/EIB-Gruppenadressen pro SPS genutzt werden. Hier sollten somit keine Engpässe auf Grund nicht ausreichend verfügbarer KNX/EIB-Gruppenadressen entstehen.

Da die Nutzung so vieler KNX/EIB-Gruppenadressen an einer SPS wieder zu einem großen KNX/EIB-Netz und der damit hohen Busbelastung führen würde, ist es somit nicht sinnvoll diese hohe Anzahl KNX/EIB-Gruppenadressen an einer Anlage zu nutzen. Es sollte daher kein Problem darstellen, ein sinnvolles Konzept für die Vergabe der KNX/EIB-Gruppenadressen zu entwickeln.

Bei beiden KNX/EIB-Klemmen muss ein Master-Baustein zur Kommunikation zwischen SPS-Programm und KNX/EIB-Klemme im SPS-Programm deklariert werden.

Das Lesen und Schreiben der KNX/EIB-Gruppenadressen erfolgt dann in beiden Systemen über vorgefertigte Funktionsbaustein, die den Programmieraufwand enorm reduzieren.

Der aktuelle Wert einer KNX/EIB-Gruppenadresse konnte jedoch nur mit der Beckhoff-SPS ausgelesen werden, da in der KNX/EIB-Bibliothek von WAGO kein Baustein hierfür zur Verfügung stand. Ein solcher Baustein muss von WAGO dringend in der KNX/EIB-Bibliothek aufgenommen werden.

7. Fazit

Mit den KNX/EIB-Klemmen von WAGO und Beckhoff lässt sich das KNX/EIB-Gebäudebussystem komfortabel in eine schnelle SPS-Umgebung integrieren. Die Funktionalität des KNX/EIB-Gebäudebussystems kann weiterhin vollständig genutzt werden. Der Automatisierung eines Objektes sind nun über die Anbindung des KNX/EIB-Bussystems an die SPS kaum noch Grenzen gesetzt.

Des Weiteren ist es nun auch problemlos möglich, die Vorteile mehrerer auf dem Markt befindlicher Bussysteme zu kombinieren, da WAGO und Beckhoff auch Klemmen für LON, DALI, MP-Bus und EnOcean anbieten und somit mehrere Bussysteme über die SPS miteinander kommunizieren können.

Da praktisch an jeder SPS eine eigene kleine KNX/EIB-Welt oder ein Teil einer KNX/EIB-Welt angebunden ist, die die Größe eines Bereiches nicht übertrifft, ist ein überlasteter KNX/EIB-Bus somit auf Grund der geringeren Anzahl der Geräte in einer KNX/EIB-Welt prinzipiell fast nicht mehr möglich. Die Anzahl der KNX/EIB-Geräte in einem Objekt ist dagegen fast unbegrenzt, und dennoch kommt es auf Grund der schnellen Kommunikation zwischen den SPS-Anlagen über Ethernet nicht zu Engpässen bei der Verarbeitung der Daten.

Die Anbindung des KNX/EIB-Gebäudebussystems an die Gebäudeautomatisierungssysteme von WAGO und Beckhoff unterscheiden sich zwar in der Art der Anbindung des KNX/EIB und der Inbetriebnahme der KNX/EIB-Klemme deutlich, die Programmierung in der IEC 61131 über die Funktionsbausteine ist aber trotzdem recht ähnlich.

Das einzige unüberwindbare Hindernis, das auch nicht durch eine SPS gelöst werden konnte, sind die vom Hersteller eines KNX/EIB-Gerätes im Applikationsprogramm festgelegten Funktionalitäten eines KNX/EIB-Gerätes. Ist hier ein Objekt, wie z.B. das Objekt „aktueller Sollwert“ in diesem Projekt, nicht über den Bus beschreibbar, kann die gewünschte Funktion auch nicht realisiert werden.

Hier muss somit vor der Nutzung eines KNX/EIB-Gerätes beim Hersteller eines KNX/EIB-Gerätes eine Bestätigung eingeholt werden, dass das Gerät die gewünschte Funktionalität zur Verfügung stellt.

Zur Anbindung der Automatisierungsebene stellen die Firmen WAGO und Beckhoff eigene OPC Server zur Verfügung.

Mit der Kombination einer schnellen SPS mit einer nach IEC 61131 genormten Automatisierungsplattform und einem am Markt weit verbreiteten Gebäudebussystems, wie dem KNX/EIB, ist es möglich, eine komfortable und leistungsfähige Gebäudeautomation in allen Ebenen der Automatisierungspyramide zu realisieren.

8. Möglichkeiten zur Umsetzung in großen Objekten

Zur Umsetzung möchte ich an dieser Stelle zwei Möglichkeiten vorschlagen.

1. Vorschlag zur Umsetzung:

In einem Haustechnikraum auf jeder Etage wird eine SPS installiert. Die SPS kann bei Beckhoff mit mehreren Buskoppler mit integriertem 2-Port Switch in Form einer Linientopologie oder mit einer Ethernet-Schnittstelle über einen Switch miteinander verbunden. Bei den WAGO Buskoppler können diese nur über einen Switch miteinander verbunden werden.

Die Buskoppler reichen die empfangenen Daten hierbei nur zwischen der SPS und den angesteckten Klemmen weiter, ohne sie selber zu verarbeiten. Die Installation der Buskoppler könnte in einer Zwischendecke oder einem Doppelboden realisiert werden. Auf diese Weise sind kurze Leitungswege zwischen Klemmen und Geräten möglich. Die Kommunikation mit dem KNX/EIB-Bus sollte sinnvoller Weise direkt an der SPS im Haustechnikraum erfolgen.

Es handelt sich hierbei um ein sehr zentrales Konzept, dass einfach zu installieren und verhältnismäßig preisgünstig ist. Bei einem Ausfall der SPS wäre die Funktionalität einer gesamten Etage dann aber nicht mehr gegeben.

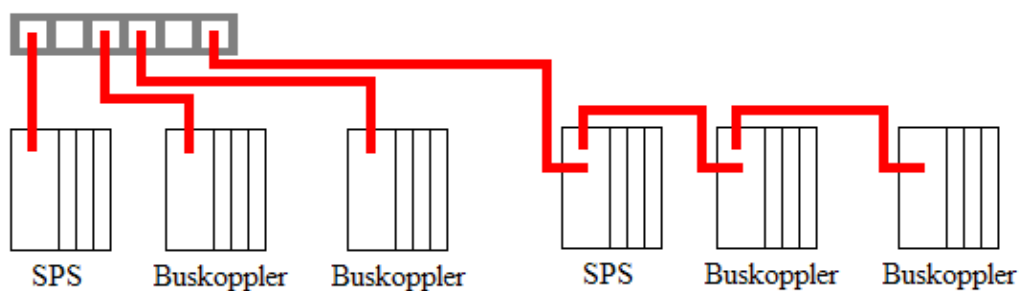


Abbildung 56: Konzept „SPS mit Buskopplern“

Benötigte Komponenten für eine Büroetage mit 50 Büros nach dem Ausstattungsgrad dieses Projektes:

- 1x SPS (z.B.: Beckhoff CX9000 oder WAGO 750-841)
- 5x Buskoppler (z.B.: Beckhoff BK9100 oder WAGO 750-341)
- 176x Digitaleingang (2 Taster Beleuchtung je Büro, 25 Trennwandschalter, 50 Taster Beleuchtung Flur und 1x Taster Beleuchtung Haustechnikraum)
- 501x Digitalausgang (9 Ausgänge je Büro, 50 Ausgänge Flurbeleuchtung und 1x Beleuchtung Haustechnikraum)
- 50x KNX/EIB-Sensor (1x je Büro (z.B.: Siemens RTR UP 231/2 DELTA profil))

2. Vorschlag zur Umsetzung:

Über die gesamte Etage werden anstelle der Buskoppler eigenständige SPS-Controller installiert, die bei Beckhoff je nach Modell über einen Switch (Abbildung 56 rechts) oder in Form einer Linientopologie (Abbildung 56 links) miteinander verbunden werden können; bei WAGO jedoch nur über einen Switch (Abbildung 56 rechts). Die KNX/EIB-Linie kann problemlos an allen Controllern angeklemmt werden, sodass jeder Controller mit den für ihn relevanten KNX/EIB-Gruppenadressen kommunizieren kann.

Bei diesem Konzept sind ebenfalls kurze Leitungswege möglich. Ein Totalausfall der Anlage ist durch mehrere verteilte Systeme jedoch fast auszuschließen.

Diese Konzept ist durch die höhere Anzahl intelligenter Systeme aber auch teuer.

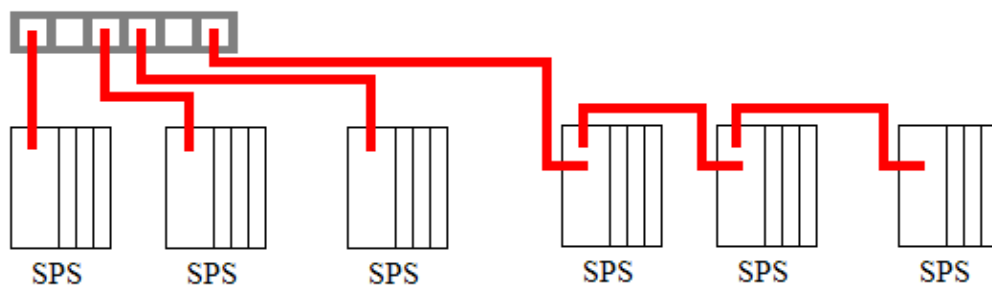


Abbildung 57: Konzept „Verteilte SPS-Controller“

Benötigte Komponenten für eine Büroetage mit 50 Büros nach dem Ausstattungsgrad dieses Projektes:

- 5x SPS (z.B.: Beckhoff CX9000 oder WAGO 750-841)
- 175x Digitaleingang (2 Taster Beleuchtung je Büro, 25 Trennwandschalter und 50 Taster Beleuchtung Flur)
- 500x Digitalausgang (9 Ausgänge je Büro und 50 Ausgänge Flurbeleuchtung)
- 50x KNX/EIB-Sensor (1x je Büro (z.B.: Siemens RTR UP 231/2 DELTA profil))

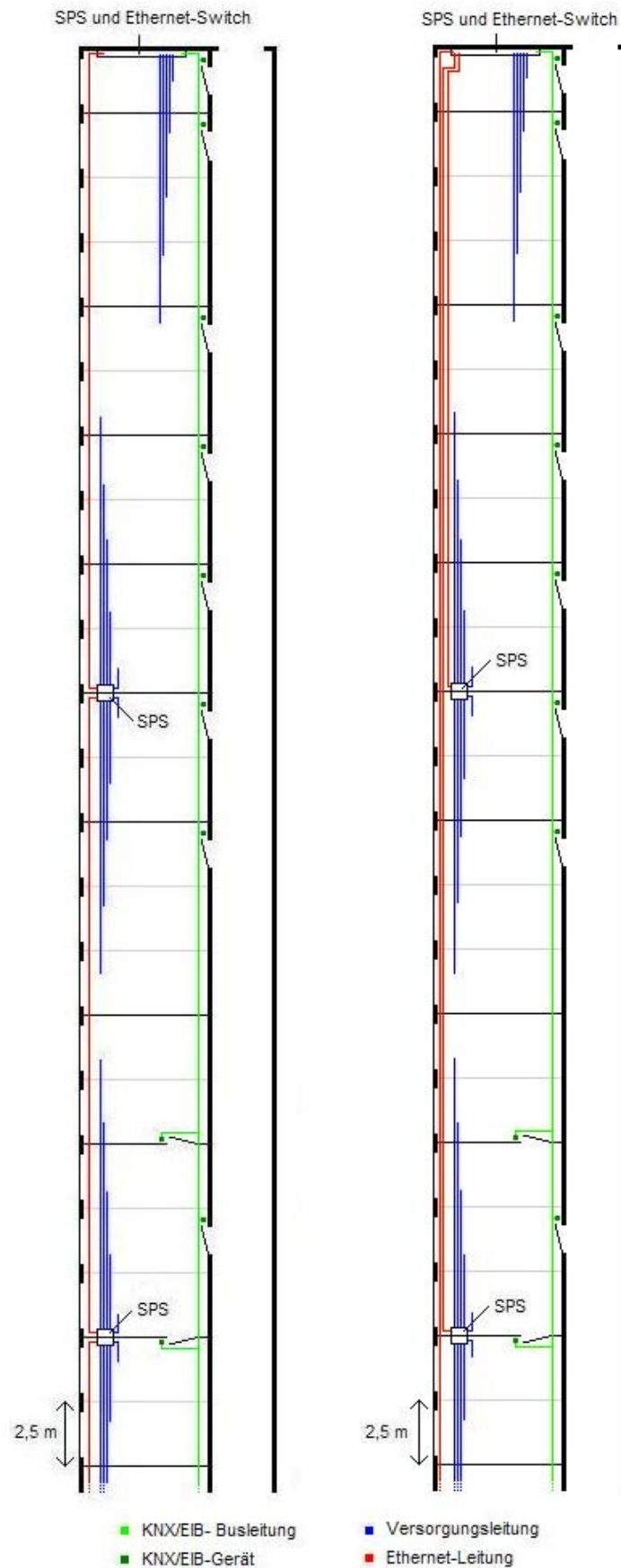


Abbildung 58: Umsetzungsmöglichkeiten in großen Objekten

Abkürzungsverzeichnis

AS	Ablaufsprache
AWL	Anweisungsliste
CFC	Continuous Function Chart (Frei grafischer Funktionsplan)
d.h.	das heißt
DALI	Digital Addressable Lighting Interface
DPT	Data Point Typ (Datenpunkttyp)
DVI	Digital Visual Interface
E/A	Ein-/Ausgabe
EIB	European Installation Bus (Europäischer Installationsbus)
ETS	Engineering Tool Software
FUP	Funktionsplan
IEC	International Electrotechnical Commission
I/O	Input/Output
KNX	Konnex Association
KOP	Kontaktplan
LED	Light Emitting Diode (lichtemittierende Diode)
LON	Local Operating Network
MP-Bus	Belimo Master-Slave Bus
RTR	Raumtemperaturregler
SPS	Speicherprogrammierbare Steuerung
ST	Structured Text (Strukturierter Text)
TCP/IP	Transmission Control Protocol/Internet Protocol
TP	Twisted-Pair
USB	Universal Serial Bus
usw.	und so weiter
VGA	Video Graphics Array
XML	Extensible Markup Language (erweiterbare Auszeichnungssprache)
z.B.	zum Beispiel

Danksagungen

An dieser Stelle möchte ich mich bei Herrn Prof. Dr. Bernd Aschendorf und Herrn Dipl.-Ing. Dieter König für die freundliche Unterstützung während meiner Diplomarbeit bedanken.

Des Weiteren danke ich den Firmen WAGO und Beckhoff für den freundlichen Support und die zur Verfügung gestellten Unterlagen und Hilfsmittel.

Literatur

- Internetseite der Beckhoff Automation GmbH, <http://www.beckhoff.de>,
Stand: Januar 2008
- Beckhoff Information System, <http://infosys.beckhoff.com/>,
Stand: August 2007
- Internetseite der WAGO Kontakttechnik GmbH & Co. KG,
<http://www.wago.com/>, Stand: Januar 2008
- Internetseite der EnOcean GmbH, <http://www.enocean.com/de/>,
Stand: Januar 2008
- Rainer Scherg: EIB planen, installieren und visualisieren, Vogel Buchverlag,
ISBN: 978-3-8343-3055-0

Programmcode des WAGO Controllers

Der gesamte Programmcode befindet sich auch im Inhalt der beigelegten CD.

Funktionsbausteine

```
FUNCTION_BLOCK FB_Buero
VAR
    LStkr1_schalten: Fb_Stromstoss;
    LStkr2_schalten: Fb_Stromstoss;
END_VAR
VAR_INPUT
    LStkr1_IN: BOOL;
    LStkr2_IN: BOOL;
    Trennwand_IN: BOOL;
    LStkr_1_IN_Konferenz: BOOL;
    LStkr_2_IN_Konferenz: BOOL;
    ZentralEIN_LStkr1: BOOL;
    ZentralEIN_LStkr2: BOOL;
    ZentralAUS_LStkr1: BOOL;
    ZentralAUS_LStkr2: BOOL;
END_VAR
VAR_OUTPUT
    LStkr1_OUT: BOOL;
    LStkr2_OUT: BOOL;
END_VAR

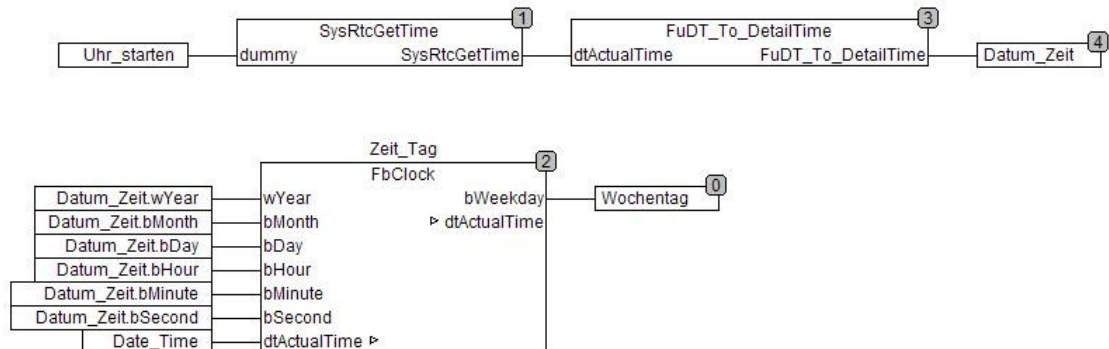
(*Lichtstromkreis 1 im Büro schalten*)
LStkr1_schalten(
    xTaster:= ((LStkr1_IN AND NOT Trennwand_IN) OR (LStkr_1_IN_Konferenz AND Trennwand_IN)),
    (*Lichtstromkreis umschalten einzeln oder im Konferenzraummodus kompletten LStkr1*)
    xZenAUS:= ZentralAUS_LStkr1,          (*Für ZentralAUS Funktion*)
    xZenEIN:= ZentralEIN_LStkr1,          (*Licht ein im Konferenzmodus (Abgleich mit zweitem Raum)*)
    bResetModus:= ,
    xAktor=> LStkr1_OUT);

LStkr2_schalten(
    xTaster:= ((LStkr2_IN AND NOT Trennwand_IN) OR (LStkr_2_IN_Konferenz AND Trennwand_IN)),
    (*Lichtstromkreis umschalten einzeln oder im Konferenzraummodus kompletten LStkr2*)
    xZenAUS:= ZentralAUS_LStkr2,          (*Für ZentralAUS Funktion*)
    xZenEIN:= ZentralEIN_LStkr2,          (*Licht ein im Konferenzmodus (Abgleich mit zweitem Raum)*)
    bResetModus:= ,
    xAktor=> LStkr2_OUT);
```

```

FUNCTION_BLOCK FB_Datum_Zeit
VAR_INPUT
END_VAR
VAR_OUTPUT
    Wochentag: BYTE;
    Datum_Zeit: typDetailTime;
END_VAR
VAR
    Zeituhr: FB_Time;
    Uhr_starten: BOOL := TRUE;
    Zeit_Tag: FbClock;
    Date_Time: DT;
END_VAR

```



```

FUNCTION_BLOCK FB_FlurLicht
VAR_INPUT
    Taster: BOOL;
END_VAR
VAR_OUTPUT
    Lampe: BOOL;
END_VAR
VAR
    LStkr_lang: Fb_KurzLang;
    LStkr: Fb_Treppe2;
    LStkr_Dauer: Fb_Stromstoss;
END_VAR

```

(*Auswertung langer oder kurzer Tastendruck*)

```

LStkr_lang(
    xTaster:= Taster,
    uiT_10tel_s:= ,
    uiTL_10tel_s:= ,
    uiTK_10tel_s:= ,
    xLang=> LStkr_Dauer.xTaster,      (*langer Tastendruck Dauerlicht*)
    xKurz=> LStkr.xTaster);          (*kurzer Tastendruck Treppenhauslicht*)

```

(*Lichtstromkreis Flur umschalten (Dauerlicht über langen Tastendruck)*)

```

LStkr_Dauer(
    xTaster:= ,
    xZenAUS:= ,
    xZenEIN:= ,
    bResetModus:= ,
    xAktor=> LStkr.xHand);

```

(*Lichtstromkreis Flur Treppenhauslicht über kurzen Tastendruck*)

```

LStkr(
    xTaster:= ,
    xHand:= ,
    dwT_10tel_s:= 100,
    dwTv_10tel_s:= 50,
    xAktor=> Lampe);

```

```
FUNCTION_BLOCK FB_Heizung_Klima_EIB
VAR
    Stoptimer_re: TP;
    Stoptimer_li: TP;
    Einzelraumregler_Buero_rechts: Fb_PidHeatingCooling;
    Buero_re_Isttemp: FbDPT_Value_Temp;
    Buero_re_Solltemp: FbDPT_Value_Temp;
    Buero_re_BasisSoll: FbDPT_Value_Temp;
    Buero_re_Komfort: FbDPT_Enable;
    Buero_re_Nacht: FbDPT_Enable;
    Buero_re_Frost_Hitze: FbDPT_Enable;
    Buero_re_Heiz_Kuehl: FbDPT_Enable;
    SetpointCorrection_re: REAL;
    Einzelraumregler_Buero_links: Fb_PidHeatingCooling;
    Buero_li_Isttemp: FbDPT_Value_Temp;
    Buero_li_Solltemp: FbDPT_Value_Temp;
    Buero_li_BasisSoll: FbDPT_Value_Temp;
    Buero_li_Komfort: FbDPT_Enable;
    Buero_li_Nacht: FbDPT_Enable;
    Buero_li_Frost_Hitze: FbDPT_Enable;
    Buero_li_Heiz_Kuehl: FbDPT_Enable;
    SetpointCorrection_li: REAL;
END_VAR
VAR_INPUT
    xNightMode_re: BOOL;
    xFrostHeat_re: BOOL;
    xDewpoint_re: BOOL;
    rReferenceComfort_re: REAL;
    rOffsetStandbyHeating_re: REAL;
    rOffsetStandbyCooling_re: REAL;
    rOffsetNightHeating_re: REAL;
    rOffsetNightCooling_re: REAL;
    rDeadZone_re: REAL;
    xNightMode_li: BOOL;
    xFrostHeat_li: BOOL;
    xDewpoint_li: BOOL;
    rReferenceComfort_li: REAL;
    rOffsetStandbyHeating_li: REAL;
    rOffsetStandbyCooling_li: REAL;
    rOffsetNightHeating_li: REAL;
    rOffsetNightCooling_li: REAL;
    rDeadZone_li: REAL;
END_VAR
VAR_IN_OUT
    typDPT_Isttemp_re: typDPT;
    typDPT_Solltemp_re: typDPT;
    typDPT_BasisSolltemp_re: typDPT;
    typDPT_Komfort_re: typDPT;
    typDPT_Nacht_re: typDPT;
    typDPT_Frost_Hitze_re: typDPT;
    typDPT_heizen_kuehlen_re: typDPT;
    typKNX: typKNX;
    typDPT_Isttemp_li: typDPT;
    typDPT_Solltemp_li: typDPT;
    typDPT_BasisSolltemp_li: typDPT;
    typDPT_Komfort_li: typDPT;
    typDPT_Nacht_li: typDPT;
    typDPT_Frost_Hitze_li: typDPT;
    typDPT_heizen_kuehlen_li: typDPT;
END_VAR
VAR_OUTPUT
    rActualTemperature_re: REAL;
    Heizen_re: BOOL;
    Kuehlen_re: BOOL;
    rSetpointHeating_re: REAL;
    rSetpointCooling_re: REAL;
    xComfort_re: BOOL;
    xStandby_re: BOOL;
    xNight_re: BOOL;
    xFrost_Heat_re: BOOL;
    rActualTemperature_li: REAL;
    Heizen_li: BOOL;
    Kuehlen_li: BOOL;
    rSetpointHeating_li: REAL;
    rSetpointCooling_li: REAL;
```

```
xComfort_li: BOOL;
xStandby_li: BOOL;
xNight_li: BOOL;
xFrost_Heat_li: BOOL;
END_VAR

(*-----Büro rechts-----*)
(*Einzelraumreglungs-Baustein*)
Einzelraumregler_Buero_rechts(
    rRoomTemperature:= Buero_re_Isttemp.rValue_OUT,
    rSetpointCorrection:= SetpointCorrection_re,
    xComfortStandby:= Buero_re_Komfort.xSwitch_OUT,
    xNightMode:= xNightMode_re,
    xFrostHeat:= xFrostHeat_re,
    xDewpoint:= xDewpoint_re,
    rReferenceComfort:= rReferenceComfort_re,
    rOffsetStandbyHeating:= rOffsetStandbyHeating_re,
    rOffsetStandbyCooling:= rOffsetStandbyCooling_re,
    rOffsetNightHeating:= rOffsetNightHeating_re,
    rOffsetNightCooling:= rOffsetNightCooling_re,
    rDeadZone:= rDeadZone_re,
    rOffset:= ,
    rKpHeating:= ,
    rTnHeating:= ,
    rTdHeating:= ,
    rKpCooling:= ,
    rTnCooling:= ,
    rTdCooling:= ,
    rActualTemperature=> rActualTemperature_re,
    rY_Heating=> ,
    rY_Cooling=> ,
    wY_Heating=> ,
    wY_Cooling=> ,
    xHeating=> ,
    xCooling=> ,
    rSetpointHeating=> rSetpointHeating_re,
    rSetpointCooling=> rSetpointCooling_re,
    rComfortHeating=> ,
    rComfortCooling=> ,
    rStandbyHeating=> ,
    rStandbyCooling=> ,
    rNightHeating=> ,
    rNightCooling=> ,
    rSetpointFrost=> ,
    rSetpointHeat=> ,
    xComfort=> xComfort_re,
    xStandby=> xStandby_re,
    xNight=> xNight_re,
    xFrost_Heat=> xFrost_Heat_re);

(*Istwert Temperatur*)
Buero_re_Isttemp(
    rValue_IN:= ,
    xUpdate_KNX:= ,
    rSendOnDelta:= ,
    tMinSendTime:= ,
    typDPT:= typDPT_Isttemp_re,
    typKNX:= typKNX,
    rValue_OUT=> ,
    xUpdate_PLC=> ,
    xTimeOut=> );

(*Sollwert Temperatur*)
Buero_re_Solltemp(
    rValue_IN:= ,
    xUpdate_KNX:= ,
    rSendOnDelta:= ,
    tMinSendTime:= ,
    typDPT:= typDPT_Solltemp_re,
    typKNX:= typKNX,
    rValue_OUT=> ,
    xUpdate_PLC=> ,
    xTimeOut=> );
```

(*Basis Sollwert Temperatur*)

```
Buero_re_BasisSoll(  
    rValue_IN:= Einzelraumregler_Buero_rechts.rReferenceComfort,  
    xUpdate_KNX:= ,  
    rSendOnDelta:= ,  
    tMinSendTime:= ,  
    typDPT:= typDPT_BasisSolltemp_re,  
    typKNX:= typKNX,  
    rValue_OUT=> ,  
    xUpdate_PLC=> ,  
    xTimeOut=> );
```

(*Funktion Komfort/Standby-Betrieb*)

```
Buero_re_Komfort(  
    xSwitch_IN:= ,  
    xUpdate_KNX:= ,  
    tMinSendTime:= ,  
    typDPT:= typDPT_Komfort_re,  
    typKNX:= typKNX,  
    xSwitch_OUT=> ,  
    xUpdate_PLC=> ,  
    xTimeOut=> );
```

(*Funktion Nacht-Betrieb*)

```
Buero_re_Nacht(  
    xSwitch_IN:= Einzelraumregler_Buero_rechts.xNight,  
    xUpdate_KNX:= ,  
    tMinSendTime:= ,  
    typDPT:= typDPT_Nacht_re,  
    typKNX:= typKNX,  
    xSwitch_OUT=> ,  
    xUpdate_PLC=> ,  
    xTimeOut=> );
```

(*Funktion Frost/Hitze-Schutz*)

```
Buero_re_Frost_Hitze(  
    xSwitch_IN:= Einzelraumregler_Buero_rechts.xFrost_Heat,  
    xUpdate_KNX:= ,  
    tMinSendTime:= ,  
    typDPT:= typDPT_Frost_Hitze_re,  
    typKNX:= typKNX,  
    xSwitch_OUT=> ,  
    xUpdate_PLC=> ,  
    xTimeOut=> );
```

(*Funktion Umschaltung Heizen/Kühlen*)

```
Buero_re_Heiz_Kuehl(  
    xSwitch_IN:= ,  
    xUpdate_KNX:= ,  
    tMinSendTime:= ,  
    typDPT:= typDPT_heizen_kuehlen_re,  
    typKNX:= typKNX,  
    xSwitch_OUT=> ,  
    xUpdate_PLC=> ,  
    xTimeOut=> );
```

(*Umschaltung Heizen/Kühlen senden*)

```
IF (Einzelraumregler_Buero_rechts.xHeating = TRUE) THEN  
    Buero_re_Heiz_Kuehl.xSwitch_IN := FALSE;  
END_IF
```

```
IF (Einzelraumregler_Buero_rechts.xCooling = TRUE) THEN  
    Buero_re_Heiz_Kuehl.xSwitch_IN := TRUE;  
END_IF
```

```
IF (Einzelraumregler_Buero_rechts.xHeating = FALSE AND Einzelraumregler_Buero_rechts.xCooling = FALSE) THEN  
    Buero_re_Heiz_Kuehl.xSwitch_IN := FALSE;  
END_IF
```

(*Berechnung der Sollwertverschiebung in K im Heizen-Modus*)

```
Stoptimer_re(IN:= Buero_re_Komfort.xUpdate_PLC, PT:= T#5s, Q=> , ET=> );
```

```
IF (Stoptimer_re.Q = FALSE) THEN  
    IF (Buero_re_Heiz_Kuehl.xSwitch_IN = FALSE) THEN  
        IF (Einzelraumregler_Buero_rechts.xComfort = TRUE) THEN
```

```
SetpointCorrection_re := (Buero_re_Solltemp.rValue_OUT -
Einzelraumregler_Buero_rechts.rReferenceComfort);
END_IF
IF (Einzelraumregler_Buero_rechts.xStandby = TRUE) THEN
SetpointCorrection_re := (Buero_re_Solltemp.rValue_OUT -
(Einzelraumregler_Buero_rechts.rReferenceComfort - Einzelraumregler_Buero_rechts.rOffsetStandbyHeating));
END_IF
IF (Einzelraumregler_Buero_rechts.xNight = TRUE) THEN
SetpointCorrection_re := (Buero_re_Solltemp.rValue_OUT -
(Einzelraumregler_Buero_rechts.rReferenceComfort - Einzelraumregler_Buero_rechts.rOffsetNightHeating));
END_IF
END_IF

(*Berechnung der Sollwertverschiebung in K im Kühlen-Modus*)
IF (Buero_re_Heiz_Kuehl.xSwitch_IN = TRUE) THEN
IF (Einzelraumregler_Buero_rechts.xComfort = TRUE) THEN
SetpointCorrection_re := (Buero_re_Solltemp.rValue_OUT -
(Einzelraumregler_Buero_rechts.rReferenceComfort + Einzelraumregler_Buero_rechts.rDeadZone));
END_IF
IF (Einzelraumregler_Buero_rechts.xStandby = TRUE) THEN
SetpointCorrection_re := (Buero_re_Solltemp.rValue_OUT -
(Einzelraumregler_Buero_rechts.rReferenceComfort + Einzelraumregler_Buero_rechts.rDeadZone +
Einzelraumregler_Buero_rechts.rOffsetStandbyCooling));
END_IF
IF (Einzelraumregler_Buero_rechts.xNight = TRUE) THEN
SetpointCorrection_re := (Buero_re_Solltemp.rValue_OUT -
(Einzelraumregler_Buero_rechts.rReferenceComfort + Einzelraumregler_Buero_rechts.rDeadZone +
Einzelraumregler_Buero_rechts.rOffsetNightCooling));
END_IF
END_IF
END_IF

(*Ausgang heizen schalten*)
IF (Einzelraumregler_Buero_rechts.wY_Heating > 0) THEN
Heizen_re := TRUE;
ELSE
Heizen_re := FALSE;
END_IF

(*Ausgang kühlen schalten*)
IF (Einzelraumregler_Buero_rechts.wY_Cooling > 0) THEN
Kuehlen_re := TRUE;
ELSE
Kuehlen_re := FALSE;
END_IF

(*-----Büro links-----*)
(*Einzelraumreglungs-Baustein*)
Einzelraumregler_Buero_links(
rRoomTemperature:= Buero_li_Isttemp.rValue_OUT,
rSetpointCorrection:= SetpointCorrection_li,
xComfortStandby:= Buero_li_Komfort.xSwitch_OUT,
xNightMode:= xNightMode_li,
xFrostHeat:= xFrostHeat_li,
xDewpoint:= xDewpoint_li,
rReferenceComfort:= rReferenceComfort_li,
rOffsetStandbyHeating:= rOffsetStandbyHeating_li,
rOffsetStandbyCooling:= rOffsetStandbyCooling_li,
rOffsetNightHeating:= rOffsetNightHeating_li,
rOffsetNightCooling:= rOffsetNightCooling_li,
rDeadZone:= rDeadZone_li,
rOffset:= ,
rKpHeating:= ,
rTnHeating:= ,
rTdHeating:= ,
rKpCooling:= ,
rTnCooling:= ,
rTdCooling:= ,
rActualTemperature=> rActualTemperature_li,
rY_Heating=> ,
rY_Cooling=> ,
wY_Heating=> ,
wY_Cooling=> ,
xHeating=> ,
xCooling=> ,
```

```
rSetpointHeating=> rSetpointHeating_li,  
rSetpointCooling=> rSetpointCooling_li,  
rComfortHeating=> ,  
rComfortCooling=> ,  
rStandbyHeating=> ,  
rStandbyCooling=> ,  
rNightHeating=> ,  
rNightCooling=> ,  
rSetpointFrost=> ,  
rSetpointHeat=> ,  
xComfort=> xComfort_li,  
xStandby=> xStandby_li,  
xNight=> xNight_li,  
xFrost_Heat=> xFrost_Heat_li);
```

(*Istwert Temperatur*)

```
Buero_li_Isttemp(  
  rValue_IN:= ,  
  xUpdate_KNX:= ,  
  rSendOnDelta:= ,  
  tMinSendTime:= ,  
  typDPT:= typDPT_Isttemp_li,  
  typKNX:= typKNX,  
  rValue_OUT=> ,  
  xUpdate_PLC=> ,  
  xTimeOut=> );
```

(*Sollwert Temperatur*)

```
Buero_li_Solltemp(  
  rValue_IN:= ,  
  xUpdate_KNX:= ,  
  rSendOnDelta:= ,  
  tMinSendTime:= ,  
  typDPT:= typDPT_Solltemp_li,  
  typKNX:= typKNX,  
  rValue_OUT=> ,  
  xUpdate_PLC=> ,  
  xTimeOut=> );
```

(*Basis Sollwert Temperatur*)

```
Buero_li_BasisSoll(  
  rValue_IN:= Einzelraumregler_Buero_rechts.rReferenceComfort,  
  xUpdate_KNX:= ,  
  rSendOnDelta:= ,  
  tMinSendTime:= ,  
  typDPT:= typDPT_BasisSolltemp_li,  
  typKNX:= typKNX,  
  rValue_OUT=> ,  
  xUpdate_PLC=> ,  
  xTimeOut=> );
```

(*Funktion Komfort/Standby-Betrieb*)

```
Buero_li_Komfort(  
  xSwitch_IN:= ,  
  xUpdate_KNX:= ,  
  tMinSendTime:= ,  
  typDPT:= typDPT_Komfort_li,  
  typKNX:= typKNX,  
  xSwitch_OUT=> ,  
  xUpdate_PLC=> ,  
  xTimeOut=> );
```

(*Funktion Nacht-Betrieb*)

```
Buero_li_Nacht(  
  xSwitch_IN:= Einzelraumregler_Buero_links.xNight,  
  xUpdate_KNX:= ,  
  tMinSendTime:= ,  
  typDPT:= typDPT_Nacht_li,  
  typKNX:= typKNX,  
  xSwitch_OUT=> ,  
  xUpdate_PLC=> ,  
  xTimeOut=> );
```

(*Funktion Frost/Hitze-Schutz*)

```
Buero_li_Frost_Hitze(  
  xSwitch_IN:= Einzelraumregler_Buero_links.xFrost_Schutz,  
  xUpdate_KNX:= ,  
  tMinSendTime:= ,  
  typDPT:= typDPT_Frost_Hitze_li,  
  typKNX:= typKNX,  
  xSwitch_OUT=> ,  
  xUpdate_PLC=> ,  
  xTimeOut=> );
```

```
xSwitch_IN:= Einzelraumregler_Buero_links.xFrost_Heat,
xUpdate_KNX:= ,
tMinSendTime:= ,
typDPT:= typDPT_Frost_Hitze_li,
typKNX:= typKNX,
xSwitch_OUT=>,
xUpdate_PLC=>,
xTimeOut=> );

(*Funktion Umschaltung Heizen/Kühlen*)
Buero_li_Heiz_Kuehl(
    xSwitch_IN:= ,
    xUpdate_KNX:= ,
    tMinSendTime:= ,
    typDPT:= typDPT_heizen_kuehlen_li,
    typKNX:= typKNX,
    xSwitch_OUT=>,
    xUpdate_PLC=>,
    xTimeOut=> );

(*Umschaltung Heizen/Kühlen senden*)
IF (Einzelraumregler_Buero_links.xHeating = TRUE) THEN
    Buero_li_Heiz_Kuehl.xSwitch_IN := FALSE;
END_IF

IF (Einzelraumregler_Buero_links.xCooling = TRUE) THEN
    Buero_li_Heiz_Kuehl.xSwitch_IN := TRUE;
END_IF

IF (Einzelraumregler_Buero_links.xHeating = FALSE AND Einzelraumregler_Buero_links.xCooling = FALSE) THEN
    Buero_li_Heiz_Kuehl.xSwitch_IN := FALSE;
END_IF

(*Berechnung der Sollwertverschiebung in K im Heizen-Modus*)
Soptimer_li(IN:= Buero_li_Komfort.xUpdate_PLC, PT:= T#5s, Q=>, ET=> );

IF (Soptimer_li.Q = FALSE) THEN
    IF (Buero_li_Heiz_Kuehl.xSwitch_IN = FALSE) THEN
        IF (Einzelraumregler_Buero_links.xComfort = TRUE) THEN
            SetpointCorrection_li := (Buero_li_Solltemp.rValue_OUT -
Einzelraumregler_Buero_links.rReferenceComfort);
        END_IF
        IF (Einzelraumregler_Buero_links.xStandby = TRUE) THEN
            SetpointCorrection_li := (Buero_li_Solltemp.rValue_OUT -
(Einzelraumregler_Buero_links.rReferenceComfort - Einzelraumregler_Buero_links.rOffsetStandbyHeating));
        END_IF
        IF (Einzelraumregler_Buero_links.xNight = TRUE) THEN
            SetpointCorrection_li := (Buero_li_Solltemp.rValue_OUT -
(Einzelraumregler_Buero_links.rReferenceComfort - Einzelraumregler_Buero_links.rOffsetNightHeating));
        END_IF
    END_IF
END_IF

(*Berechnung der Sollwertverschiebung in K im Kühlen-Modus*)
IF (Buero_li_Heiz_Kuehl.xSwitch_IN = TRUE) THEN
    IF (Einzelraumregler_Buero_links.xComfort = TRUE) THEN
        SetpointCorrection_li := (Buero_li_Solltemp.rValue_OUT -
(Einzelraumregler_Buero_links.rReferenceComfort + Einzelraumregler_Buero_links.rDeadZone));
    END_IF
    IF (Einzelraumregler_Buero_rechts.xStandby = TRUE) THEN
        SetpointCorrection_li := (Buero_li_Solltemp.rValue_OUT -
(Einzelraumregler_Buero_links.rReferenceComfort + Einzelraumregler_Buero_links.rDeadZone +
Einzelraumregler_Buero_links.rOffsetStandbyCooling));
    END_IF
    IF (Einzelraumregler_Buero_rechts.xNight = TRUE) THEN
        SetpointCorrection_li := (Buero_li_Solltemp.rValue_OUT -
(Einzelraumregler_Buero_links.rReferenceComfort + Einzelraumregler_Buero_links.rDeadZone +
Einzelraumregler_Buero_links.rOffsetNightCooling));
    END_IF
END_IF
END_IF

(*Ausgang heizen schalten*)
IF (Einzelraumregler_Buero_links.wY_Heating > 0) THEN
    Heizen_li := TRUE;
ELSE
```

```
        Heizen_li := FALSE;
    END_IF

(*Ausgang kühlen schalten*)
IF (Einzelraumregler_Buero_links.wY_Cooling > 0) THEN
    Kuehlen_li := TRUE;
ELSE
    Kuehlen_li := FALSE;
END_IF

FUNCTION_BLOCK FB_Jalousiesignale_EIB
VAR_INPUT
END_VAR
VAR_OUTPUT
    Up: BOOL;
    Down: BOOL;
    StepUp: BOOL;
    StepDown: BOOL;
END_VAR
VAR
    Jal_Fahren: FbDPT_UpDown;
    Jal_Lamellen: FbDPT_UpDown;
END_VAR
VAR_IN_OUT
    typDPT_fahren: typDPT;
    typDPT_Step: typDPT;
    typKNX: typKNX;
END_VAR

(*Funktionsbaustein EIB_Gruppenadresse "Fahren"*)
Jal_Fahren(
    xSwitch_IN:= ,
    xUpdate_KNX:= ,
    tMinSendTime:= ,
    typDPT:= typDPT_fahren,
    typKNX:= typKNX,
    xSwitch_OUT=> ,
    xUpdate_PLC=> ,
    xTimeOut=> );

(*Funktionsbaustein EIB_Gruppenadresse "Lamellen"*)
Jal_Lamellen(
    xSwitch_IN:= ,
    xUpdate_KNX:= ,
    tMinSendTime:= ,
    typDPT:= typDPT_Step,
    typKNX:= typKNX,
    xSwitch_OUT=> ,
    xUpdate_PLC=> ,
    xTimeOut=> );

(*Jalousie Fahrbetrieb hoch*)
IF (Jal_Fahren.xSwitch_OUT AND Jal_Fahren.xUpdate_PLC) THEN
    Up := TRUE;
ELSE
    Up := FALSE;
END_IF

(*Jalousie Fahrbetrieb runter*)
IF (NOT Jal_Fahren.xSwitch_OUT AND Jal_Fahren.xUpdate_PLC) THEN
    Down := TRUE;
ELSE
    Down := FALSE;
END_IF

(*Jalousie Schrittbetrieb hoch*)
IF (Jal_Lamellen.xSwitch_OUT AND Jal_Lamellen.xUpdate_PLC) THEN
    StepUp := TRUE;
ELSE
    StepUp := FALSE;
END_IF
```

```
(*Jalousie Schrittbetrieb runter*)
IF (NOT Jal_Lamellen.xSwitch_OUT AND Jal_Lamellen.xUpdate_PLC) THEN
    StepDown := TRUE;
ELSE
    StepDown := FALSE;
END_IF
```

```
FUNCTION_BLOCK FB_Konferenz_Jalousie
VAR_INPUT
```

```
    Jalousie_re_Up: BOOL;
    Jalousie_re_Down: BOOL;
    Jalousie_re_StepUp: BOOL;
    Jalousie_re_StepDown: BOOL;
    Trennwand_IN: BOOL;
    Jalousie_li_Up: BOOL;
    Jalousie_li_Down: BOOL;
    Jalousie_li_StepUp: BOOL;
    Jalousie_li_StepDown: BOOL;
```

```
END_VAR
```

```
VAR_OUTPUT
```

```
    Jalousie_rechts_Up: BOOL;
    Jalousie_rechts_Down: BOOL;
    Jalousie_rechts_StepUp: BOOL;
    Jalousie_rechts_StepDown: BOOL;
    Konferenz_Up: BOOL;
    Konferenz_Down: BOOL;
    Konferenz_StepUp: BOOL;
    Konferenz_StepDown: BOOL;
    Jalousie_links_Up: BOOL;
    Jalousie_links_Down: BOOL;
    Jalousie_links_StepUp: BOOL;
    Jalousie_links_StepDown: BOOL;
```

```
END_VAR
```

```
VAR
```

```
END_VAR
```

```
IF (Trennwand_IN = TRUE) THEN
```

```
(*Jalousien im Konferenzraum-Modus hochfahren*)
```

```
    IF (Jalousie_re_Up OR Jalousie_li_Up) THEN
        Konferenz_Up := TRUE;
    ELSE
        Konferenz_Up := FALSE;
    END_IF
```

```
(*Jalousien im Konferenzraum-Modus runterfahren*)
```

```
    IF (Jalousie_re_Down OR Jalousie_li_Down) THEN
        Konferenz_Down := TRUE;
    ELSE
        Konferenz_Down := FALSE;
    END_IF
```

```
(*Jalousien im Konferenzraum-Modus Schritt hoch*)
```

```
    IF (Jalousie_re_StepUp OR Jalousie_li_StepUp) THEN
        Konferenz_StepUp := TRUE;
    ELSE
        Konferenz_StepUp := FALSE;
    END_IF
```

```
(*Jalousien im Konferenzraum-Modus Schritt runter*)
```

```
    IF (Jalousie_re_StepDown OR Jalousie_li_StepDown) THEN
        Konferenz_StepDown := TRUE;
    ELSE
        Konferenz_StepDown := FALSE;
    END_IF
```

```
END_IF
```

```
(*Zuweisung der Eingänge auf die Ausgänge im Büro-Modus*)
```

```
IF (Trennwand_IN = FALSE) THEN
    Jalousie_rechts_Up:= Jalousie_re_Up;
    Jalousie_rechts_Down:= Jalousie_re_Down;
    Jalousie_rechts_StepUp:= Jalousie_re_StepUp;
```

```
Jalousie_rechts_StepDown:= Jalousie_re_StepDown;
Jalousie_links_Up:= Jalousie_li_Up;
Jalousie_links_Down:= Jalousie_li_Down;
Jalousie_links_StepUp:= Jalousie_li_StepUp;
Jalousie_links_StepDown:= Jalousie_li_StepDown;
END_IF

FUNCTION_BLOCK FB_Position_Jalousie
VAR_INPUT
    Position_Jalousie_re: BYTE;
    Trennwand_IN: BOOL;
    Position_Jalousie_li: BYTE;
END_VAR
VAR_OUTPUT
    Jalousie_Position: BOOL;
END_VAR
VAR
    Wand_offen: R_TRIG;
    Mittelwert_re_li: BYTE;
END_VAR

Wand_offen(CLK:= Trennwand_IN);

(*wenn die Trennwand geöffnet wird, fahren die Jalousien auf eine gemittelte Position*)
IF (Wand_offen.Q) THEN
    IF (Position_Jalousie_re > 0 OR Position_Jalousie_li > 0) THEN
        Mittelwert_re_li := ((Position_Jalousie_re + Position_Jalousie_li)/2);
        Jalousie_Position := TRUE;
    END_IF
END_IF

(*Jalousien auf gleicher Position dann Stop*)
IF (Position_Jalousie_re = Mittelwert_re_li AND Position_Jalousie_li = Mittelwert_re_li) THEN
    Jalousie_Position := FALSE;
END_IF

FUNCTION_BLOCK FB_Wand_Status_Licht
VAR_INPUT
    Buero_re_LStkr1: BOOL;
    Buero_re_LStkr2: BOOL;
    Trennwand_IN: BOOL;
    Buero_li_LStkr1: BOOL;
    Buero_li_LStkr2: BOOL;
END_VAR
VAR_OUTPUT
    ZentraleIN_LStkr1: BOOL;
    ZentraleIN_LStkr2: BOOL;
END_VAR
VAR
    Wand_offen: R_TRIG;
END_VAR

Wand_offen(CLK:= Trennwand_IN, Q=> );

(*wird die Trennwand geöffnet, dann Abgleich der Lichtstromkreise 1*)
IF ((Buero_re_LStkr1 OR Buero_li_LStkr1) AND Wand_offen.Q) THEN
    ZentraleIN_LStkr1 := TRUE;
ELSE
    ZentraleIN_LStkr1 := FALSE;
END_IF

(*wird die Trennwand geöffnet, dann Abgleich der Lichtstromkreise 2*)
IF ((Buero_re_LStkr2 OR Buero_li_LStkr2) AND Wand_offen.Q) THEN
    ZentraleIN_LStkr2 := TRUE;
ELSE
    ZentraleIN_LStkr2 := FALSE;
END_IF
```

Programme

PROGRAM Heisswasser_Bueros_101_102

VAR

Zeit_Wochentag: FB_Datum_Zeit;

(*Deklarationen Heisswasser Büro 101*)

Heisswasser_B_101: FbSchaltuhr;

HW_Ein_Stunde_B_101: BYTE;

HW_Ein_Minute_B_101: BYTE;

HW_Aus_Stunde_B_101: BYTE;

HW_Aus_Minute_B_101: BYTE;

HW_Mo_B_101: BOOL;

HW_Di_B_101: BOOL;

HW_Mi_B_101: BOOL;

HW_Do_B_101: BOOL;

HW_Fr_B_101: BOOL;

HW_Sa_B_101: BOOL;

HW_So_B_101: BOOL;

(*Deklarationen Heisswasser Büro 102*)

Heisswasser_B_102: FbSchaltuhr;

HW_Ein_Stunde_B_102: BYTE;

HW_Ein_Minute_B_102: BYTE;

HW_Aus_Stunde_B_102: BYTE;

HW_Aus_Minute_B_102: BYTE;

HW_Mo_B_102: BOOL;

HW_Di_B_102: BOOL;

HW_Mi_B_102: BOOL;

HW_Do_B_102: BOOL;

HW_Fr_B_102: BOOL;

HW_Sa_B_102: BOOL;

HW_So_B_102: BOOL;

END_VAR

(*Zeitschaltbaustein Büro 101*)

(*Einschaltstunde für Heißwasserbereiter Büro 101*)

(*Einschaltminute für Heißwasserbereiter Büro 101*)

(*Ausschaltstunde für Heißwasserbereiter Büro 101*)

(*Ausschaltminute für Heißwasserbereiter Büro 101*)

(*-----*)

(* Auswahl der *)

(* Einschalttage *)

(* Büro 101 *)

(* -----*)

(*Zeitschaltbaustein Büro 102*)

(*Einschaltstunde für Heißwasserbereiter Büro 102*)

(*Einschaltminute für Heißwasserbereiter Büro 102*)

(*Ausschaltstunde für Heißwasserbereiter Büro 102*)

(*Ausschaltminute für Heißwasserbereiter Büro 102*)

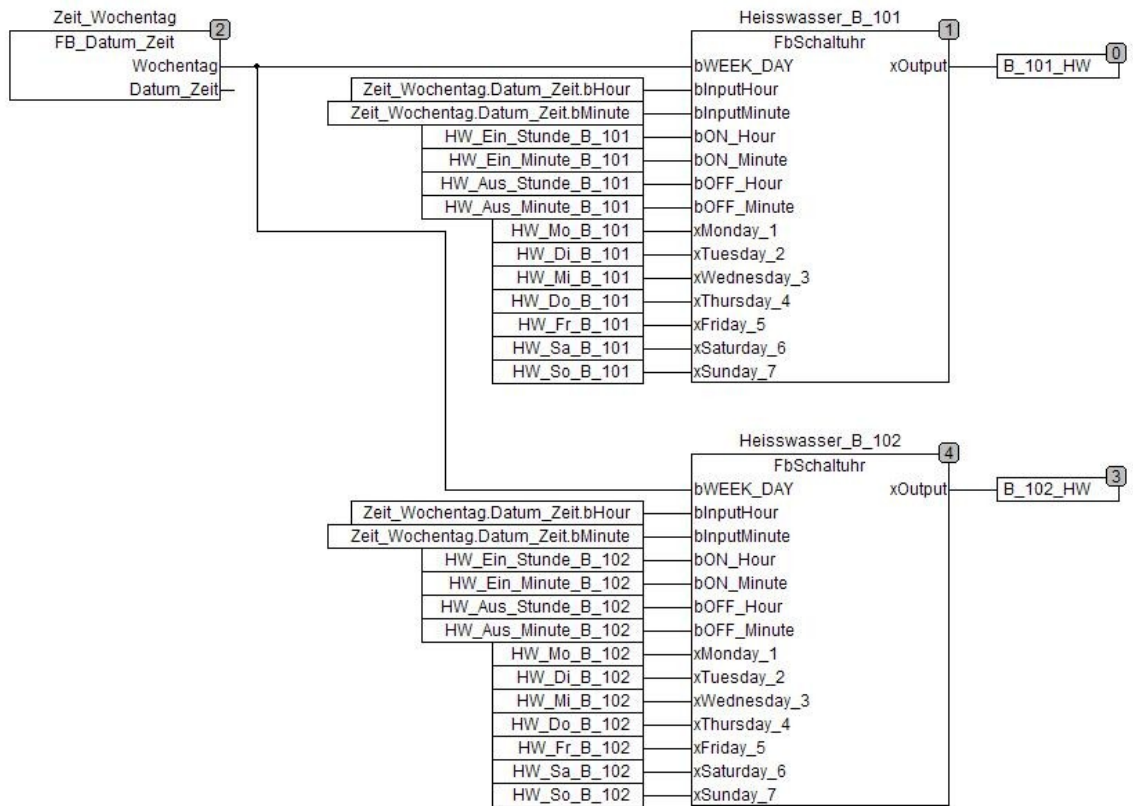
(*-----*)

(* Auswahl der *)

(* Einschalttage *)

(* Büro 102 *)

(* -----*)



Vergleich der Implementierung der Sensorik und Aktorik des KNX/EIB-Gebäudebussystems in
IEC 61131-basierte Gebäudeautomatisierungssysteme von WAGO und Beckhoff

PROGRAM HK_B_101_102

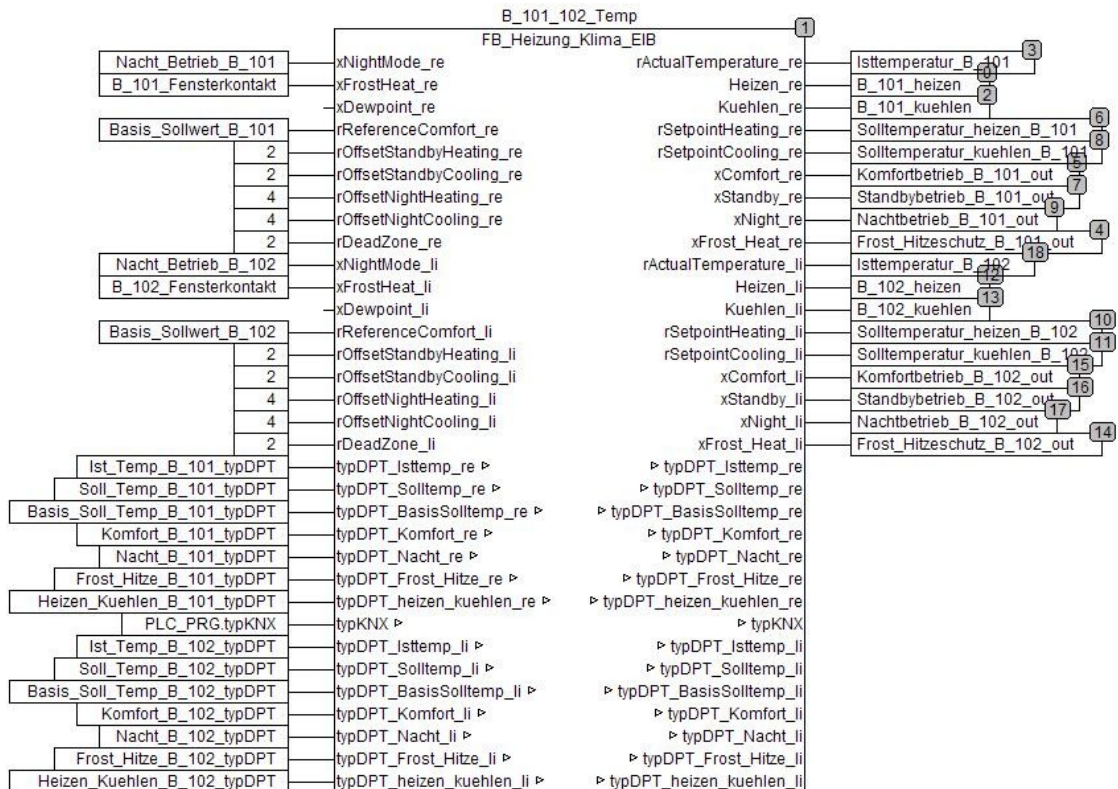
VAR

B_101_102_Temp: FB_Heizung_Klima_EIB;

Nacht_Betrieb_B_101: BOOL;	(*Einschaltvariable für Nachtbetrieb*)
Basis_Sollwert_B_101: REAL := 21;	(*Basis-Solltemperatur*)
Ist_Temp_B_101_typDPT: typDPT;	
Soll_Temp_B_101_typDPT: typDPT;	
Basis_Soll_Temp_B_101_typDPT: typDPT;	
Komfort_B_101_typDPT: typDPT;	
Nacht_B_101_typDPT: typDPT;	
Frost_Hitze_B_101_typDPT: typDPT;	
Heizen_Kuehlen_B_101_typDPT: typDPT;	
Isttemperatur_B_101: REAL;	(*Isttemperatur*)
Solltemperatur_heizen_B_101: REAL;	(*Solltemperatur Modus Heizen*)
Solltemperatur_kuehlen_B_101: REAL;	(*Solltemperatur Modus Kühlen*)
Komfortbetrieb_B_101_out: BOOL;	(*Status Komfortbetrieb*)
Standbybetrieb_B_101_out: BOOL;	(*Status Standbybetrieb*)
Nachtbetrieb_B_101_out: BOOL;	(*Status Nachtbetriebsbetrieb*)
Frost_Hitzeschutz_B_101_out: BOOL;	(*Status Frost-/Hitzeschutz*)

Nacht_Betrieb_B_102: BOOL;	(*Einschaltvariable für Nachtbetrieb*)
Basis_Sollwert_B_102: REAL := 21;	(*Basis-Solltemperatur*)
Ist_Temp_B_102_typDPT: typDPT;	
Soll_Temp_B_102_typDPT: typDPT;	
Basis_Soll_Temp_B_102_typDPT: typDPT;	
Komfort_B_102_typDPT: typDPT;	
Nacht_B_102_typDPT: typDPT;	
Frost_Hitze_B_102_typDPT: typDPT;	
Heizen_Kuehlen_B_102_typDPT: typDPT;	
Isttemperatur_B_102: REAL;	(*Isttemperatur*)
Solltemperatur_heizen_B_102: REAL;	(*Solltemperatur Modus Heizen*)
Solltemperatur_kuehlen_B_102: REAL;	(*Solltemperatur Modus Kühlen*)
Komfortbetrieb_B_102_out: BOOL;	(*Status Komfortbetrieb*)
Standbybetrieb_B_102_out: BOOL;	(*Status Standbybetrieb*)
Nachtbetrieb_B_102_out: BOOL;	(*Status Nachtbetriebsbetrieb*)
Frost_Hitzeschutz_B_102_out: BOOL;	(*Status Frost-/Hitzeschutz*)

END_VAR



Vergleich der Implementierung der Sensorik und Aktorik des KNX/EIB-Gebäudebussystems in IEC 61131-basierte Gebäudeautomatisierungssysteme von WAGO und Beckhoff

PROGRAM Jalousie_B_101_102

VAR

Buero_101_Jalousie_aussen: FbSunblind_KNX; (*Jalousiebaustein aussen*)
Buero_102_Jalousie_aussen: FbSunblind_KNX; (*Jalousiebaustein aussen*)
Buero_101_102_Konferenz_aussen: FB_Konferenz_Jalousie; (*Baustein Konferenzabfrage aussen*)
Wand_101_102_offen_Jal_aussen: FB_Position_Jalousie; (*Baustein Positionsabgleich im Konferenzraum-

Modus*)

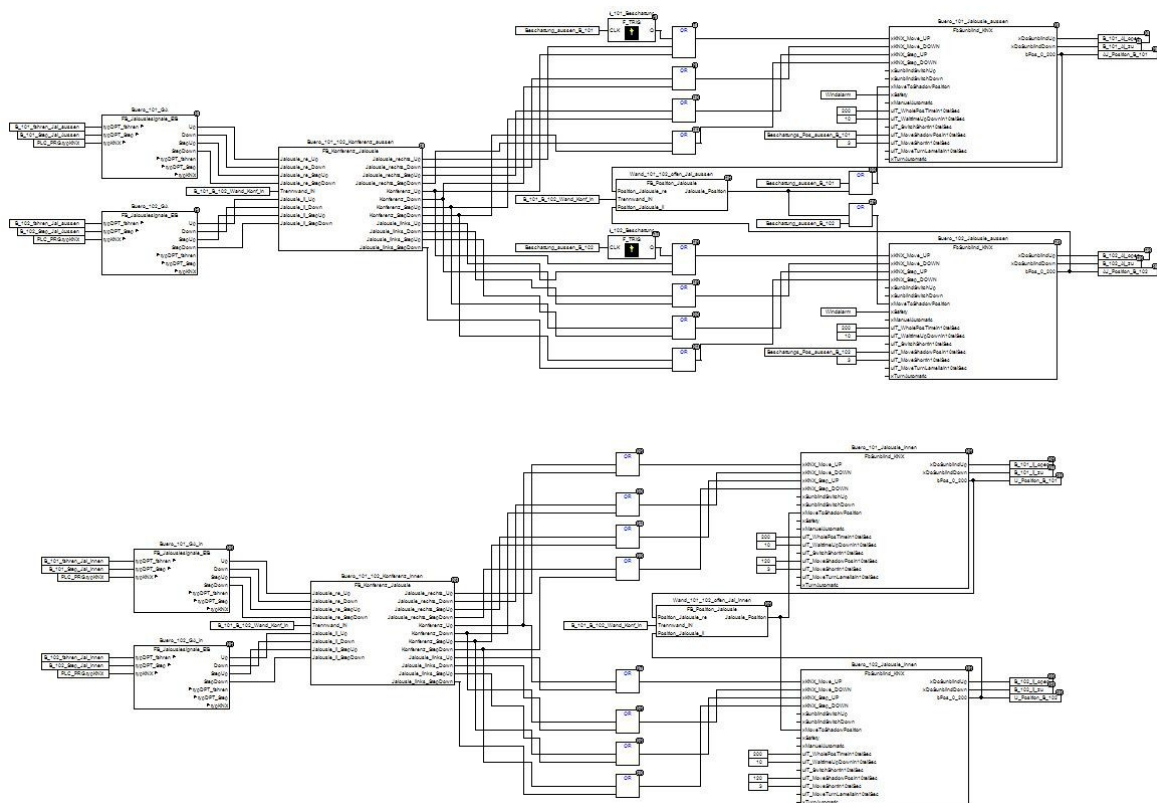
Buero_101_GA: FB_Jalousiesignale_EIB; (*Baustein zur Auswertung der EIB-Signale*)
Buero_102_GA: FB_Jalousiesignale_EIB; (*Baustein zur Auswertung der EIB-Signale*)
B_101_fahren_Jal_aussen: typDPT;
B_101_Step_Jal_Aussen: typDPT;
B_102_fahren_Jal_aussen: typDPT;
B_102_Step_Jal_Aussen: typDPT;
B_101_Beschattung: F_TRIG;
B_102_Beschattung: F_TRIG;
Beschattung_aussen_B_101: BOOL; (*Variable für Beschattungsposition B101*)
Beschattung_aussen_B_102: BOOL; (*Variable für Beschattungsposition B102*)
Beschattungs_Pos_aussen_B_101: UINT := 120; (*Beschattungsposition B101*)
Beschattungs_Pos_aussen_B_102: UINT := 120; (*Beschattungsposition B102*)
AJ_Position_B_101: BYTE; (*Positionsbestimmung der Aussenjalousie*)
AJ_Position_B_102: BYTE; (*Positionsbestimmung der Aussenjalousie*)

Buero_101_Jalousie_innen: FbSunblind_KNX; (*Jalousiebaustein innen*)
Buero_102_Jalousie_innen: FbSunblind_KNX; (*Jalousiebaustein innen*)
Buero_101_102_Konferenz_innen: FB_Konferenz_Jalousie; (*Baustein Konferenzabfrage innen*)
Wand_101_102_offen_Jal_innen: FB_Position_Jalousie; (*Baustein Positionsabgleich im Konferenzraum-

Modus*)

Buero_101_GA_in: FB_Jalousiesignale_EIB; (*Baustein zur Auswertung der EIB-Signale*)
Buero_102_GA_in: FB_Jalousiesignale_EIB; (*Baustein zur Auswertung der EIB-Signale*)
B_101_fahren_Jal_innen: typDPT;
B_101_Step_Jal_innen: typDPT;
B_102_fahren_Jal_innen: typDPT;
B_102_Step_Jal_innen: typDPT;
IJ_Position_B_101: BYTE; (*Positionsbestimmung der Aussenjalousie*)
IJ_Position_B_102: BYTE; (*Positionsbestimmung der Aussenjalousie*)

END_VAR



Vergleich der Implementierung der Sensorik und Aktorik des KNX/EIB-Gebäudebussystems in IEC 61131-basierte Gebäudeautomatisierungssysteme von WAGO und Beckhoff

PROGRAM Licht_Bueros_101_102

VAR

(*Deklarationen Lichtstromkreise Büros 101 & 102*)

Buero_101: FB_Buero; (*Baustein zur Steuerung Lichtfunktionen im Büro 101*)

Buero_102: FB_Buero; (*Baustein zur Steuerung Lichtfunktionen im Büro 102*)

Wand_101_102_offen: FB_Wand_Status_Licht; (*Abgleich der Lichtstromkreise im Konferenzraum-Modus*)

(*Deklarationen EnOcean*)

LStkr1_LStkr2_Funk_B_101: FbButton_4_Channel; (*Baustein für den Empfang der EnOcean TasterSignale für Büro 101*)

LStkr1_LStkr2_Funk_B_102: FbButton_4_Channel; (*Baustein für den Empfang der EnOcean TasterSignale für Büro 102*)

B_101_Taster_ID: DWORD := 1140678; (*ID des EnOcean Tasters für Büro 101*)

B_102_Taster_ID: DWORD := 1141737; (*ID des EnOcean Tasters für Büro 102*)

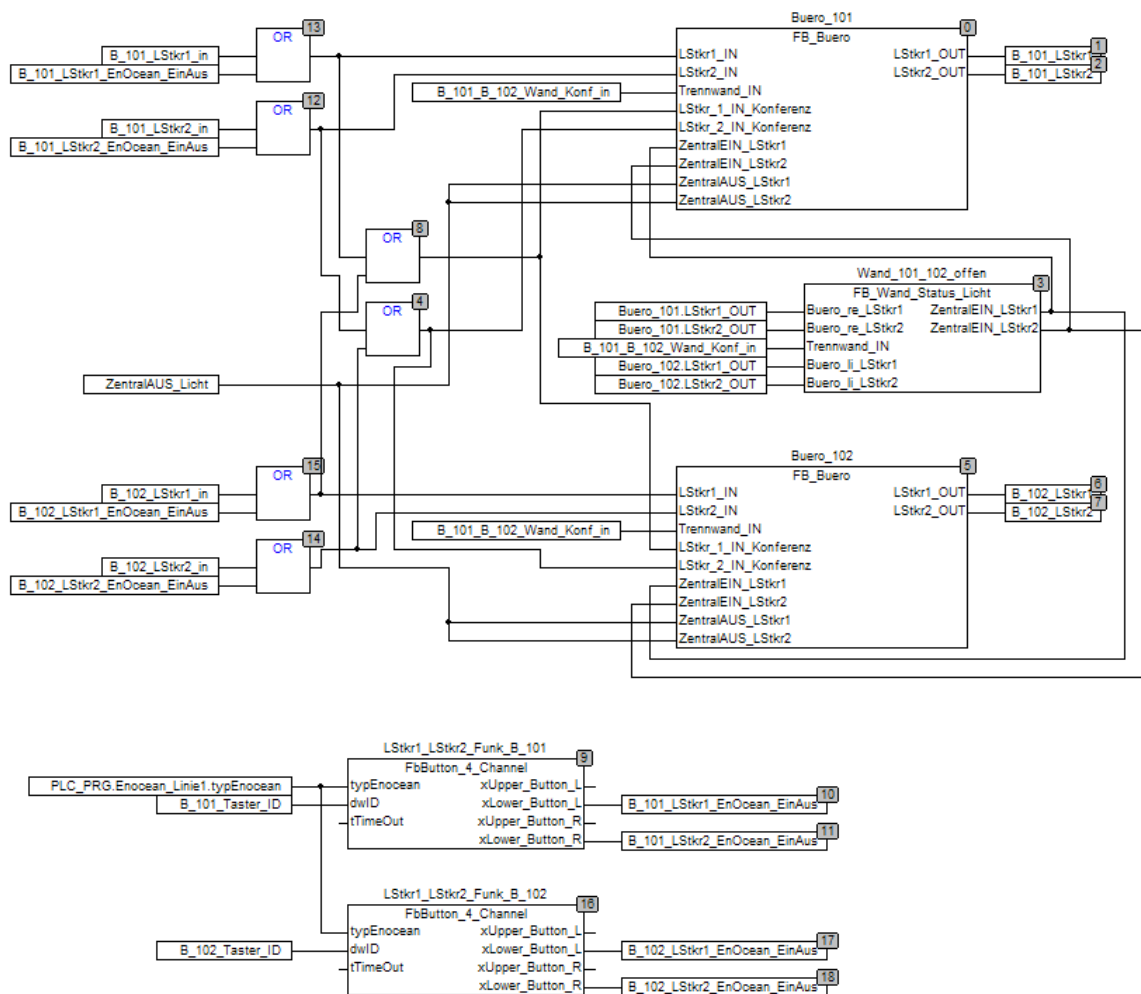
B_101_LStkr1_EnOcean_EinAus: BOOL; (*Tasten des EnOcean Tasters für Büro 101*)

B_101_LStkr2_EnOcean_EinAus: BOOL; (*Tasten des EnOcean Tasters für Büro 101*)

B_102_LStkr1_EnOcean_EinAus: BOOL; (*Tasten des EnOcean Tasters für Büro 102*)

B_102_LStkr2_EnOcean_EinAus: BOOL; (*Tasten des EnOcean Tasters für Büro 102*)

END_VAR



Vergleich der Implementierung der Sensorik und Aktorik des KNX/EIB-Gebäudebussystems in IEC 61131-basierte Gebäudeautomatisierungssysteme von WAGO und Beckhoff

PROGRAM LW_B_101_102

VAR

Leinwand_101_102: FbSunblind_KNX;

(*Baustein zur Steuerung der Leinwand*)

Leinwand_zu_101_102: F_TRIG;

(*Hochfahren der Leinwand

wenn Trennwand zu*)

LW_Up_101_102: FbDPT_Bool;

(*Baustein zum

Senden des hochfahren Befehls an den EIB*)

LW_Down_101_102: FbDPT_UpDown;

(*Baustein zum Senden des runterfahren

Befehls an den EIB*)

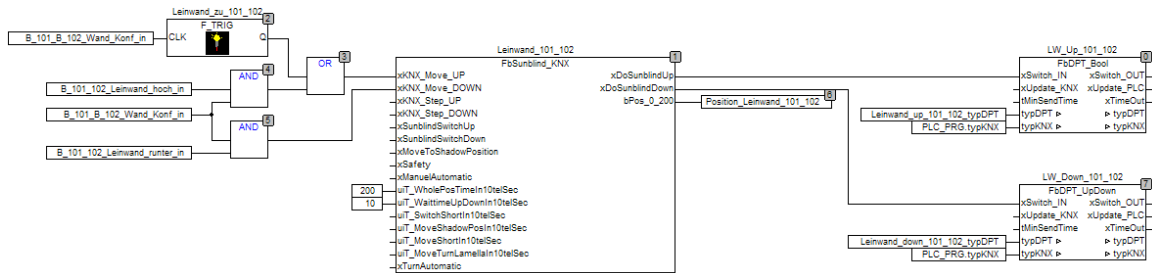
Leinwand_up_101_102_typDPT: typDPT;

Leinwand_down_101_102_typDPT: typDPT;

Position_Leinwand_101_102: BYTE;

(*Positionsangabe der Leinwand*)

END_VAR



PROGRAM Licht_Flur_101_102

VAR

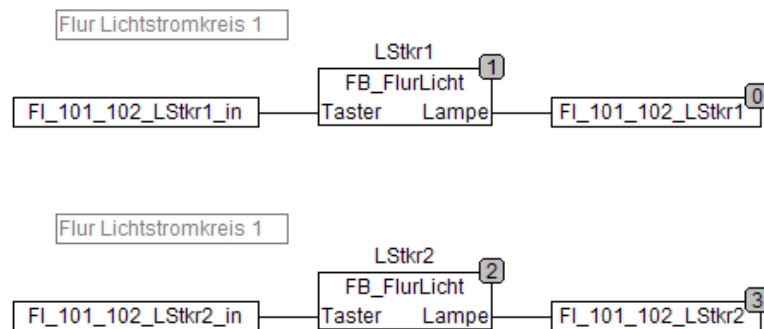
LStkr1: FB_FlurLicht;

(*Baustein zur Steuerung des Lichtstromkreises 1 im Flur 001 002*)

LStkr2: FB_FlurLicht;

(*Baustein zur Steuerung des Lichtstromkreises 2 im Flur 001 002*)

END_VAR



PROGRAM PLC_PRG

VAR

(*Deklarationen EnOcean Initialisierung*)
EnOcean_Linie1: FbEnOceanReceive;

(*Deklarationen EIB Initialisierung*)
EIB_Klemme1: FbKNX_Master_646;
EIB_Status: enumStatusKNX;
typKNX: typKNX;

END_VAR

(*Initialisierung EnOcean Klemme 1*)

EnOcean_Linie1(
 bModule_750_642:= 1,
 typEnOcean=> ,
 bError=>);

(*Initialisierung EIB Klemme 1*)

EIB_Klemme1(
 bModule_753_646:= 1,
 typKNX:= typKNX,
 enumStatusKNX=> EIB_Status,
 xProg_Mode=>);

(*Aufrufe Büroprogramme*)

Heisswasser_Bueros_101_102(); HK_B_101_102(); Jalousie_B_101_102(); Licht_Bueros_101_102(); LW_B_101_102();
Heisswasser_Bueros_103_104(); HK_B_103_104(); Jalousie_B_103_104(); Licht_Bueros_103_104(); LW_B_103_104();
Heisswasser_Bueros_105_106(); HK_B_105_106(); Jalousie_B_105_106(); Licht_Bueros_105_106(); LW_B_105_106();
Heisswasser_Bueros_107_108(); HK_B_107_108(); Jalousie_B_107_108(); Licht_Bueros_107_108(); LW_B_107_108();

(*Aufrufe Flurprogramme*)

Licht_Flur_101_102();
Licht_Flur_103_104();
Licht_Flur_105_106();
Licht_Flur_107_108();

Programmcode des Beckhoff Controllers

Der gesamte Programmcode befindet sich auch im Inhalt der beigegeführten CD.

Funktionsbausteine

```
FUNCTION_BLOCK FB_Buero
VAR_INPUT
    LStkr1_IN: BOOL;
    LStkr2_IN: BOOL;
    Trennwand_IN: BOOL;
    LStkr1_IN_Konferenz: BOOL;
    LStkr2_IN_Konferenz: BOOL;
    ZentralEIN_LStkr1: BOOL;
    ZentralEIN_LStkr2: BOOL;
    ZentralAUS_LStkr1: BOOL;
    ZentralAUS_LStkr2: BOOL;
END_VAR
VAR_OUTPUT
    LStkr1_OUT: BOOL;
    LStkr2_OUT: BOOL;
END_VAR
VAR
    LStkr1_schalten: FB_Light;
    LStkr2_schalten: FB_Light;
END_VAR

(*Lichtstromkreis 1 im Büro schalten*)
LStkr1_schalten(
    bOn:= ZentralEIN_LStkr1,          (*Licht ein im Konferenzmodus (Abgleich mit zweitem Raum)*)
    bOff:= ZentralAUS_LStkr1,         (*Für ZentralAUS Funktion*)
    bToggle:= ((LStkr1_IN AND NOT Trennwand_IN) OR (LStkr1_IN_Konferenz AND Trennwand_IN)),
    (*Lichtstromkreis umschalten einzeln oder im Konferenzraummodus kompletten LStkr1*)
    bLight=> LStkr1_OUT);

(*Lichtstromkreis 2 im Büro schalten*)
LStkr2_schalten(
    bOn:= ZentralEIN_LStkr2,          (*Licht ein im Konferenzmodus (Abgleich mit zweitem Raum)*)
    bOff:= ZentralAUS_LStkr2,         (*Für ZentralAUS Funktion*)
    bToggle:= ((LStkr2_IN AND NOT Trennwand_IN) OR (LStkr2_IN_Konferenz AND Trennwand_IN)),
    (*Lichtstromkreis umschalten einzeln oder im Konferenzraummodus kompletten LStkr2*)
    bLight=> LStkr2_OUT);
```

```
FUNCTION_BLOCK FB_FlurLicht
VAR_INPUT
    Taster: BOOL;
    ZentralAUS: BOOL;
END_VAR
VAR_OUTPUT
    Lampe: BOOL;
END_VAR
VAR
    LStkr_lang: FB_ShortLongClick;
    LStkr_Dauer: FB_Light;
    LStkr: FB_StairwellLight;
END_VAR

(*Auswertung langer oder kurzer Tastendruck*)
LStkr_lang(
    bSwitch:= Taster,
    tSwitchTime:= T#300ms,
    bShortClick=> LStkr.bSwitch ,
    bLongClick=> LStkr_Dauer.bToggle);
(*kurzer Tastendruck Treppenhauslicht*)
(*langer Tastendruck Dauerlicht*)

(*Lichtstromkreis Flur Treppenhauslicht über kurzen Tastendruck*)
LStkr(
    bSwitch:= ,
    bOff:= ,
    tLightDuration:= T#10s,
    tPreWarningStart:= T#8s,
    tPreWarningDuration:= T#200ms,
    bLight=> );

(*Lichtstromkreis Flur umschalten (Dauerlicht über langen Tastendruck)*)
LStkr_Dauer(
    bOn:= ,
    bOff:= ZentralAUS,
    bToggle:= ,
    bLight=> );

(*Ausgang TRUE oder FALSE*)
IF (LStkr.bLight OR LStkr_Dauer.bLight) THEN
    Lampe := TRUE;
ELSE
    Lampe := FALSE;
END_IF
```

```
FUNCTION_BLOCK FB_Heizung_Klima_EIB
VAR_INPUT
    strData_Rec: EIB_REC;
    Group_Address_heizen_re: EIB_GROUP_ADDR;
    Group_Address_kuehlen_re: EIB_GROUP_ADDR;
    Group_Address_Nachtbetrieb_re: EIB_GROUP_ADDR;
    Group_Address_Komfort_Betrieb_re: EIB_GROUP_ADDR;
    Group_Address_Standby_Betrieb_re: EIB_GROUP_ADDR;
    Group_Address_Frost_Hitze_Schutz_re: EIB_GROUP_ADDR;
    Group_Address_Sollwert_re: EIB_GROUP_ADDR;
    Group_Address_Sollwert_Verschiebung_re: EIB_GROUP_ADDR;
    Group_Address_Istwert_re: EIB_GROUP_ADDR;
    Group_Address_Basis_Sollwert_re: EIB_GROUP_ADDR;
    Nachtbetrieb_re: BOOL;
    Frost_Hitze_Schutz_re: BOOL;
    Group_Address_heizen_li: EIB_GROUP_ADDR;
    Group_Address_kuehlen_li: EIB_GROUP_ADDR;
    Group_Address_Nachtbetrieb_li: EIB_GROUP_ADDR;
    Group_Address_Komfort_Betrieb_li: EIB_GROUP_ADDR;
    Group_Address_Standby_Betrieb_li: EIB_GROUP_ADDR;
    Group_Address_Frost_Hitze_Schutz_li: EIB_GROUP_ADDR;
    Group_Address_Sollwert_li: EIB_GROUP_ADDR;
    Group_Address_Sollwert_Verschiebung_li: EIB_GROUP_ADDR;
    Group_Address_Istwert_li: EIB_GROUP_ADDR;
    Group_Address_Basis_Sollwert_li: EIB_GROUP_ADDR;
    Nachtbetrieb_li: BOOL;
    Frost_Hitze_Schutz_li: BOOL;
    Trennwand_IN: BOOL;
    Basis_Solltemp_Komfort: REAL;
END_VAR
VAR_OUTPUT
    heizen_re: BOOL;
    kuehlen_re: BOOL;
    Status_Komfort_Betrieb_re: BOOL;
    Status_Standby_Betrieb_re: BOOL;
    Istwert_re: REAL;
    Sollwert_re: REAL;
    heizen_li: BOOL;
    kuehlen_li: BOOL;
    Status_Komfort_Betrieb_li: BOOL;
    Status_Standby_Betrieb_li: BOOL;
    Istwert_li: REAL;
    Sollwert_li: REAL;
END_VAR
VAR
    EIB_Heizen_Rec_re: EIB_BIT_REC;
    EIB_Kuehlen_Rec_re: EIB_BIT_REC;
    EIB_Komfort_Rec_re: EIB_BIT_REC;
    EIB_Standby_Rec_re: EIB_BIT_REC;
    EIB_Sollwert_Rec_re: EIB_2OCTET_FLOAT_REC;
    EIB_Istwert_Rec_re: EIB_2OCTET_FLOAT_REC;
    EIB_Sollwert_Verschiebung_Rec_re: EIB_2OCTET_FLOAT_REC;
    EIB_Nachtbetrieb_Send_re: EIB_BIT_SEND;
    EIB_Basis_Sollwert_Send_re: EIB_2OCTET_FLOAT_SEND;
    EIB_Komfort_Send_re: EIB_BIT_SEND;
    EIB_Standby_Send_re: EIB_BIT_SEND;
    EIB_Frost_Hitze_Schutz_Send_re: EIB_BIT_SEND;
    EIB_Komfort_Status_aktualisieren_re: EIB_READ_SEND;
    EIB_Standby_Status_aktualisieren_re: EIB_READ_SEND;
    EIB_Sollwert_Verschiebung_Send_re: EIB_2OCTET_FLOAT_SEND;

    EIB_Heizen_Rec_li: EIB_BIT_REC;
    EIB_Kuehlen_Rec_li: EIB_BIT_REC;
    EIB_Komfort_Rec_li: EIB_BIT_REC;
    EIB_Standby_Rec_li: EIB_BIT_REC;
    EIB_Sollwert_Rec_li: EIB_2OCTET_FLOAT_REC;
    EIB_Istwert_Rec_li: EIB_2OCTET_FLOAT_REC;
    EIB_Sollwert_Verschiebung_Rec_li: EIB_2OCTET_FLOAT_REC;
    EIB_Nachtbetrieb_Send_li: EIB_BIT_SEND;
    EIB_Basis_Sollwert_Send_li: EIB_2OCTET_FLOAT_SEND;
    EIB_Komfort_Send_li: EIB_BIT_SEND;
    EIB_Standby_Send_li: EIB_BIT_SEND;
    EIB_Frost_Hitze_Schutz_Send_li: EIB_BIT_SEND;
    EIB_Komfort_Status_aktualisieren_li: EIB_READ_SEND;
    EIB_Standby_Status_aktualisieren_li: EIB_READ_SEND;
```

```
EIB_Sollwert_Verschiebung_Send_li: EIB_OCTET_FLOAT_SEND;

Komfort_Betrieb_re: BOOL;
Komfort_Status_aktualisieren_re: BOOL;
Komfort_DataReceive_re: BOOL;
Standby_Betrieb_re: BOOL;
Standby_Status_aktualisieren_re: BOOL;
Standby_DataReceive_re: BOOL;
Solltemp_re: REAL;
Solltemp_Verschiebung_re: REAL;
Sollwert_Verschiebung_re: REAL;
Sollwert_Verschiebung_DataReceive_re: BOOL;
Sollwert_DataReceive_re: BOOL;
Komfort_Betrieb_li: BOOL;
Komfort_DataReceive_li: BOOL;
Komfort_Status_aktualisieren_li: BOOL;
Standby_Betrieb_li: BOOL;
Standby_DataReceive_li: BOOL;
Standby_Status_aktualisieren_li: BOOL;
Solltemp_li: REAL;
Solltemp_Verschiebung_li: REAL;
Sollwert_Verschiebung_li: REAL;
Sollwert_Verschiebung_DataReceive_li: BOOL;
Sollwert_DataReceive_li: BOOL;

Wand_offen: R_TRIG;
Wand_zu: F_TRIG;
Wand_ist_offen: BOOL;
Wand_ist_geoeffnet: TON;
END_VAR

(*Heizen*)
EIB_Heizen_Rec_re(Group_Address:= Group_Address_heizen_re, strData_Rec:= strData_Rec, bDataReceive=> , bData=>
heizen_re); (*EIB lesen*)
EIB_Heizen_Rec_li(Group_Address:= Group_Address_heizen_li, strData_Rec:= strData_Rec, bDataReceive=> , bData=>
heizen_li); (*EIB lesen*)

(*Kühlen*)
EIB_Kuehlen_Rec_re(Group_Address:= Group_Address_kuehlen_re, strData_Rec:= strData_Rec, bDataReceive=> , bData=>
kuehlen_re); (*EIB lesen*)
EIB_Kuehlen_Rec_li(Group_Address:= Group_Address_kuehlen_li, strData_Rec:= strData_Rec, bDataReceive=> , bData=>
kuehlen_li); (*EIB lesen*)

(*Istwert Temperatur*)
EIB_Istwert_Rec_re(Group_Address:= Group_Address_Istwert_re, strData_Rec:= strData_Rec, bDataReceive=> , rData=>
Istwert_re); (*EIB lesen*)
EIB_Istwert_Rec_li(Group_Address:= Group_Address_Istwert_li, strData_Rec:= strData_Rec, bDataReceive=> , rData=>
Istwert_li); (*EIB lesen*)

(*Basis Sollwert Temperatur*)
EIB_Basis_Sollwert_Send_re( (*EIB schreiben*)
    Group_Address:= Group_Address_Basis_Sollwert_re,
    rData:= Basis_Solltemp_Komfort,
    str_Rec:= strData_Rec,
    bError=> ,
    iErrorID=> );

EIB_Basis_Sollwert_Send_li( (*EIB schreiben*)
    Group_Address:= Group_Address_Basis_Sollwert_li,
    rData:= Basis_Solltemp_Komfort,
    str_Rec:= strData_Rec,
    bError=> ,
    iErrorID=> );

(*Sollwert Temperatur*)
EIB_Sollwert_Rec_re(Group_Address:= Group_Address_Sollwert_re, strData_Rec:= strData_Rec, bDataReceive=>
Sollwert_DataReceive_re, rData=> Sollwert_re); (*EIB lesen*)
EIB_Sollwert_Rec_li(Group_Address:= Group_Address_Sollwert_li, strData_Rec:= strData_Rec, bDataReceive=>
Sollwert_DataReceive_li, rData=> Sollwert_li); (*EIB lesen*)

(*Sollwert Verschiebung*)
EIB_Sollwert_Verschiebung_Rec_re(Group_Address:= Group_Address_Sollwert_Verschiebung_re, strData_Rec:= strData_Rec,
bDataReceive=> Sollwert_Verschiebung_DataReceive_re, rData=> Sollwert_Verschiebung_re); (*EIB lesen*)
EIB_Sollwert_Verschiebung_Rec_li(Group_Address:= Group_Address_Sollwert_Verschiebung_li, strData_Rec:= strData_Rec,
bDataReceive=> Sollwert_Verschiebung_DataReceive_li, rData=> Sollwert_Verschiebung_li); (*EIB lesen*)
```

```
EIB_Sollwert_Verschiebung_Send_re(                                     (*EIB schreiben*)
    Group_Address:= Group_Address_Sollwert_Verschiebung_re,
    rData:= Solltemp_Verschiebung_re,
    str_Rec:= strData_Rec,
    bError=> ,
    iErrorID=> );

EIB_Sollwert_Verschiebung_Send_li(                                     (*EIB schreiben*)
    Group_Address:= Group_Address_Sollwert_Verschiebung_li,
    rData:= Solltemp_Verschiebung_li,
    str_Rec:= strData_Rec,
    bError=> ,
    iErrorID=> );

(*Funktion Komfort-Betrieb*)
EIB_Komfort_Rec_re(Group_Address:= Group_Address_Komfort_Betrieb_re, strData_Rec:= strData_Rec, bDataReceive=>
Komfort_DataReceive_re, bData=> Status_Komfort_Betrieb_re);          (*EIB lesen*)
EIB_Komfort_Rec_li(Group_Address:= Group_Address_Komfort_Betrieb_li, strData_Rec:= strData_Rec, bDataReceive=>
Komfort_DataReceive_li, bData=> Status_Komfort_Betrieb_li);          (*EIB lesen*)

EIB_Komfort_Send_re(                                                  (*EIB schreiben*)
    Group_Address:= Group_Address_Komfort_Betrieb_re,
    bData:= Komfort_Betrieb_re,
    str_Rec:= strData_Rec,
    bError=> ,
    iErrorID=> );

EIB_Komfort_Send_li(                                                  (*EIB schreiben*)
    Group_Address:= Group_Address_Komfort_Betrieb_li,
    bData:= Komfort_Betrieb_li,
    str_Rec:= strData_Rec,
    bError=> ,
    iErrorID=> );

EIB_Komfort_Status_aktualisieren_re(                                   (*EIB Status aktualisieren*)
    Group_Address:= Group_Address_Komfort_Betrieb_re,
    bRead:= Komfort_Status_aktualisieren_re,
    str_Rec:= strData_Rec,
    bError=> ,
    iErrorID=> );

EIB_Komfort_Status_aktualisieren_li(                                   (*EIB Status aktualisieren*)
    Group_Address:= Group_Address_Komfort_Betrieb_li,
    bRead:= Komfort_Status_aktualisieren_li,
    str_Rec:= strData_Rec,
    bError=> ,
    iErrorID=> );

(*Funktion Standby-Betrieb*)
EIB_Standby_Rec_re(Group_Address:= Group_Address_Standby_Betrieb_re, strData_Rec:= strData_Rec, bDataReceive=>
Standby_DataReceive_re, bData=> Status_Standby_Betrieb_re);          (*EIB lesen*)
EIB_Standby_Rec_li(Group_Address:= Group_Address_Standby_Betrieb_li, strData_Rec:= strData_Rec, bDataReceive=>
Standby_DataReceive_li, bData=> Status_Standby_Betrieb_li);          (*EIB lesen*)

EIB_Standby_Send_re(                                                  (*EIB schreiben*)
    Group_Address:= Group_Address_Standby_Betrieb_re,
    bData:= Standby_Betrieb_re,
    str_Rec:= strData_Rec,
    bError=> ,
    iErrorID=> );

EIB_Standby_Send_li(                                                  (*EIB schreiben*)
    Group_Address:= Group_Address_Standby_Betrieb_li,
    bData:= Standby_Betrieb_li,
    str_Rec:= strData_Rec,
    bError=> ,
    iErrorID=> );

EIB_Standby_Status_aktualisieren_re(                                   (*EIB Status aktualisieren*)
    Group_Address:= Group_Address_Standby_Betrieb_re,
    bRead:= Standby_Status_aktualisieren_re,
    str_Rec:= strData_Rec,
    bError=> ,
    iErrorID=> );

EIB_Standby_Status_aktualisieren_li(                                   (*EIB Status aktualisieren*)
    Group_Address:= Group_Address_Standby_Betrieb_li,
    bRead:= Standby_Status_aktualisieren_li,
```

```
    str_Rec:= strData_Rec,
    bError=> ,
    iErrorID=> );

(*Funktion Nacht-Betrieb*)
EIB_Nachtbetrieb_Send_re(                                     (*EIB schreiben*)
    Group_Address:= Group_Address_Nachtbetrieb_re,
    bData:= (Nachtbetrieb_re OR ((Nachtbetrieb_re OR Nachtbetrieb_li) AND Trennwand_IN)),
    str_Rec:= strData_Rec,
    bError=> ,
    iErrorID=> );
EIB_Nachtbetrieb_Send_li(                                     (*EIB schreiben*)
    Group_Address:= Group_Address_Nachtbetrieb_li,
    bData:= (Nachtbetrieb_li OR ((Nachtbetrieb_re OR Nachtbetrieb_li) AND Trennwand_IN)),
    str_Rec:= strData_Rec,
    bError=> ,
    iErrorID=> );

(*Funktion Frost/Hitze-Schutz*)
EIB_Frost_Hitze_Schutz_Send_re(                               (*EIB schreiben*)
    Group_Address:= Group_Address_Frost_Hitze_Schutz_re,
    bData:= (Frost_Hitze_Schutz_re OR ((Frost_Hitze_Schutz_re OR Frost_Hitze_Schutz_li) AND Trennwand_IN)),
    str_Rec:= strData_Rec,
    bError=> ,
    iErrorID=> );
EIB_Frost_Hitze_Schutz_Send_li(                               (*EIB schreiben*)
    Group_Address:= Group_Address_Frost_Hitze_Schutz_li,
    bData:= (Frost_Hitze_Schutz_li OR ((Frost_Hitze_Schutz_re OR Frost_Hitze_Schutz_li) AND Trennwand_IN)),
    str_Rec:= strData_Rec,
    bError=> ,
    iErrorID=> );

(*Änderungen Komfort/Standby-Betrieb durch Büro rechts*)
IF (Status_Komfort_Betrieb_re = TRUE AND Komfort_DataReceive_re = FALSE) THEN
    Komfort_Betrieb_re := TRUE;
    Standby_Betrieb_re := FALSE;
END_IF
IF (Status_Standby_Betrieb_re = TRUE AND Standby_DataReceive_re = FALSE) THEN
    Komfort_Betrieb_re := FALSE;
    Standby_Betrieb_re := TRUE;
END_IF

(*Statusabfrage nach Änderung Komfort/Standby-Betrieb durch Büro rechts*)
IF (Komfort_Betrieb_re = TRUE AND Status_Komfort_Betrieb_re = FALSE) OR (Komfort_Betrieb_re = FALSE AND
Status_Komfort_Betrieb_re = TRUE) THEN
    Komfort_Status_aktualisieren_re := TRUE;
    Standby_Status_aktualisieren_re := TRUE;
ELSE
    Komfort_Status_aktualisieren_re := FALSE;
    Standby_Status_aktualisieren_re := FALSE;
END_IF;
IF (Standby_Betrieb_re = TRUE AND Status_Standby_Betrieb_re = FALSE) OR (Standby_Betrieb_re = FALSE AND
Status_Standby_Betrieb_re = TRUE) THEN
    Komfort_Status_aktualisieren_re := TRUE;
    Standby_Status_aktualisieren_re := TRUE;
ELSE
    Komfort_Status_aktualisieren_re := FALSE;
    Standby_Status_aktualisieren_re := FALSE;
END_IF;

(*Änderungen Komfort/Standby-Betrieb durch Büro links*)
IF (Status_Komfort_Betrieb_li = TRUE AND Komfort_DataReceive_li = FALSE) THEN
    Komfort_Betrieb_li := TRUE;
    Standby_Betrieb_li := FALSE;
END_IF
IF (Status_Standby_Betrieb_li = TRUE AND Standby_DataReceive_li = FALSE) THEN
    Komfort_Betrieb_li := FALSE;
    Standby_Betrieb_li := TRUE;
END_IF

(*Statusabfrage nach Änderung Komfort/Standby-Betrieb durch Büro links*)
IF (Komfort_Betrieb_li = TRUE AND Status_Komfort_Betrieb_li = FALSE) OR (Komfort_Betrieb_li = FALSE AND
Status_Komfort_Betrieb_li = TRUE) THEN
```

```
Komfort_Status_aktualisieren_li := TRUE;
Standby_Status_aktualisieren_li := TRUE;
ELSE
    Komfort_Status_aktualisieren_li := FALSE;
END_IF;
IF (Standby_Betrieb_li = TRUE AND Status_Standby_Betrieb_li = FALSE) OR (Standby_Betrieb_li = FALSE AND
Status_Standby_Betrieb_li = TRUE) THEN
    Komfort_Status_aktualisieren_li := TRUE;
    Standby_Status_aktualisieren_li := TRUE;
ELSE
    Komfort_Status_aktualisieren_li := FALSE;
    Standby_Status_aktualisieren_li := FALSE;
END_IF;

(*Konferenzraum-Modus*)

(*Status Trennwand geöffnet*)
Wand_offen(CLK:= Trennwand_IN, Q=> );
Wand_zu(CLK:= Trennwand_IN, Q=> );
Wand_ist_geoeffnet(IN:= Trennwand_IN, PT:= T#16s, Q=> Wand_ist_offen, ET=> );

IF (Wand_offen.Q = TRUE) THEN                                     (*Abgleich der Werte beim öffnen der Trennwand*)
    IF (Status_Komfort_Betrieb_re = TRUE ) THEN
        Solltemp_Verschiebung_re := 0.02;
        Solltemp_Verschiebung_li := 0.02;
        Komfort_Betrieb_li := TRUE;
    END_IF
    IF (Status_Komfort_Betrieb_li = TRUE) THEN
        Solltemp_Verschiebung_re := 0.02;
        Solltemp_Verschiebung_li := 0.02;
        Komfort_Betrieb_re := TRUE;
    END_IF
    IF (Status_Komfort_Betrieb_re = TRUE AND Status_Komfort_Betrieb_li = TRUE OR Status_Komfort_Betrieb_re =
FALSE AND Status_Komfort_Betrieb_li = FALSE) THEN
        Solltemp_Verschiebung_re := 0.02;
        Solltemp_Verschiebung_li := 0.02;
        Komfort_Betrieb_re := FALSE;
        Komfort_Betrieb_li := FALSE;
    END_IF
END_IF

IF (Trennwand_IN = TRUE) THEN
    IF (Wand_ist_offen = TRUE) THEN

        (*Sollwertverschiebung im Konferenzraum-Modus*)
        IF (Sollwert_Verschiebung_DataReceive_re = FALSE) THEN
            Solltemp_Verschiebung_li := Sollwert_Verschiebung_re;
            Solltemp_Verschiebung_re := Sollwert_Verschiebung_re;
        END_IF
        IF (Sollwert_Verschiebung_DataReceive_li = FALSE) THEN
            Solltemp_Verschiebung_re := Sollwert_Verschiebung_li;
            Solltemp_Verschiebung_li := Sollwert_Verschiebung_li;
        END_IF

        (*Änderung Komfort/Standby Betrieb im Konferenzraum-Modus*)
        IF (Status_Komfort_Betrieb_re = TRUE AND NOT Komfort_DataReceive_re = TRUE) THEN
            Standby_Betrieb_li := FALSE;
            Komfort_Betrieb_li := TRUE;
        END_IF
        IF (Status_Standby_Betrieb_re = TRUE AND NOT Standby_DataReceive_re = TRUE) THEN
            Komfort_Betrieb_li := FALSE;
            Standby_Betrieb_li := TRUE;
        END_IF
        IF (Status_Komfort_Betrieb_li = TRUE AND NOT Komfort_DataReceive_li = TRUE) THEN
            Standby_Betrieb_re := FALSE;
            Komfort_Betrieb_re := TRUE;
        END_IF
        IF (Status_Standby_Betrieb_li = TRUE AND NOT Standby_DataReceive_li = TRUE) THEN
            Komfort_Betrieb_re := FALSE;
            Standby_Betrieb_re := TRUE;
        END_IF
    END_IF
END_IF
END_IF
```

```
(*Rücksetzen des Konferenzraum-Modus*)
IF (Wand_zu.Q = TRUE) THEN
    Solltemp_Verschiebung_re := 0.01;
    Solltemp_Verschiebung_li := 0.01;
END_IF

FUNCTION_BLOCK FB_Jalousiesignale_EIB
VAR_INPUT
    Group_Address_Up_Down: EIB_GROUP_ADDR;
    Group_Address_StepUp_StepDown: EIB_GROUP_ADDR;
    strData_Rec: EIB_REC;
END_VAR
VAR_OUTPUT
    Up: BOOL;
    Down: BOOL;
    StepUp: BOOL;
    StepDown: BOOL;
END_VAR
VAR
    Buero_Fahren: EIB_BIT_REC;
    Buero_Lamellen: EIB_BIT_REC;
END_VAR

(*Funktionsbaustein EIB_Gruppenadresse "Fahren"*)
Buero_Fahren(
    Group_Address:= Group_Address_Up_Down,
    strData_Rec:= strData_Rec,
    bDataReceive=> ,
    bData=> );

(*Funktionsbaustein EIB_Gruppenadresse "Lamellen"*)
Buero_Lamellen(
    Group_Address:= Group_Address_StepUp_StepDown,
    strData_Rec:= strData_Rec,
    bDataReceive=> ,
    bData=> );

(*Jalousie Fahrbetrieb hoch*)
IF (NOT Buero_Fahren.bData AND NOT Buero_Fahren.bDataReceive) THEN
    Up := TRUE;
ELSE
    Up := FALSE;
END_IF

(*Jalousie Fahrbetrieb runter*)
IF (Buero_Fahren.bData AND NOT Buero_Fahren.bDataReceive) THEN
    Down := TRUE;
ELSE
    Down := FALSE;
END_IF

(*Jalousie Schrittbetrieb hoch*)
IF (NOT Buero_Lamellen.bData AND NOT Buero_Lamellen.bDataReceive) THEN
    StepUp := TRUE;
ELSE
    StepUp := FALSE;
END_IF

(*Jalousie Schrittbetrieb runter*)
IF (Buero_Lamellen.bData AND NOT Buero_Lamellen.bDataReceive) THEN
    StepDown := TRUE;
ELSE
    StepDown := FALSE;
END_IF
```

```
FUNCTION_BLOCK FB_Konferenz_Jalousie
VAR_INPUT
    Jalousie_re_Up: BOOL;
    Jalousie_re_Down: BOOL;
    Jalousie_re_StepUp: BOOL;
    Jalousie_re_StepDown: BOOL;
    Trennwand_IN: BOOL;
    Jalousie_li_Up: BOOL;
    Jalousie_li_Down: BOOL;
    Jalousie_li_StepUp: BOOL;
    Jalousie_li_StepDown: BOOL;
END_VAR
VAR_OUTPUT
    Konferenz_Up: BOOL;
    Konferenz_Down: BOOL;
    Konferenz_StepUp: BOOL;
    Konferenz_StepDown: BOOL;
END_VAR
VAR
END_VAR

(*Jalousien im Konferenzraum-Modus hochfahren*)
IF (Trennwand_IN AND (Jalousie_re_Up OR Jalousie_li_Up)) THEN
    Konferenz_Up := TRUE;
ELSE
    Konferenz_Up := FALSE;
END_IF

(*Jalousien im Konferenzraum-Modus runterfahren*)
IF (Trennwand_IN AND (Jalousie_re_Down OR Jalousie_li_Down)) THEN
    Konferenz_Down := TRUE;
ELSE
    Konferenz_Down := FALSE;
END_IF

(*Jalousien im Konferenzraum-Modus Schritt hoch*)
IF (Trennwand_IN AND (Jalousie_re_StepUp OR Jalousie_li_StepUp)) THEN
    Konferenz_StepUp := TRUE;
ELSE
    Konferenz_StepUp := FALSE;
END_IF

(*Jalousien im Konferenzraum-Modus Schritt runter*)
IF (Trennwand_IN AND (Jalousie_re_StepDown OR Jalousie_li_StepDown)) THEN
    Konferenz_StepDown := TRUE;
ELSE
    Konferenz_StepDown := FALSE;
END_IF
```

```
FUNCTION_BLOCK FB_Position_Jalousie
VAR_INPUT
    Position_Jalousie_re: USINT;
    Trennwand_IN: BOOL;
    Position_Jalousie_li: USINT;
END_VAR
VAR_OUTPUT
    Jalousie_Position: BOOL;
    Jalousie_SetPosition: USINT;
END_VAR
VAR
    Wand_offen: R_TRIG;
END_VAR

Wand_offen(CLK:= Trennwand_IN);

(*wenn die Trennwand geöffnet wird, fahren die Jalousien auf eine gemittelte Position*)
IF (Wand_offen.Q) THEN
    IF (Position_Jalousie_re > 0 OR Position_Jalousie_li > 0) THEN
        Jalousie_SetPosition := ((Position_Jalousie_re + Position_Jalousie_li) / 2);
        Jalousie_Position := TRUE;
    END_IF
END_IF

(*Jalousien auf gleicher Position dann Stop*)
IF (Position_Jalousie_re = Position_Jalousie_li) THEN
    Jalousie_Position := FALSE;
END_IF
```

```
FUNCTION_BLOCK FB_Wand_Status_Licht
VAR_INPUT
    Buero_re_LStkr1: BOOL;
    Buero_re_LStkr2: BOOL;
    Trennwand_IN: BOOL;
    Buero_li_LStkr1: BOOL;
    Buero_li_LStkr2: BOOL;
END_VAR
VAR_OUTPUT
    ZentralEIN_LStkr1: BOOL;
    ZentralEIN_LStkr2: BOOL;
END_VAR
VAR
    Wand_offen: R_TRIG;
END_VAR

Wand_offen(CLK:= Trennwand_IN, Q=>);

(*wird die Trennwand geöffnet, dann Abgleich der Lichtstromkreise 1*)
IF ((Buero_re_LStkr1 OR Buero_li_LStkr1) AND Wand_offen.Q) THEN
    ZentralEIN_LStkr1 := TRUE;
ELSE
    ZentralEIN_LStkr1 := FALSE;
END_IF

(*wird die Trennwand geöffnet, dann Abgleich der Lichtstromkreise 2*)
IF ((Buero_re_LStkr2 OR Buero_li_LStkr2) AND Wand_offen.Q) THEN
    ZentralEIN_LStkr2 := TRUE;
ELSE
    ZentralEIN_LStkr2 := FALSE;
END_IF
```

Programme

PROGRAM Heisswasser_Bueros_001_002

VAR

(*Deklarationen Heisswasser Büro 001*)

Buero_001: FB_WeeklyTimeSwitch;

HW_Enable_B_001: BOOL;

HW_OnTime_B_001: TOD;

HW_OffTime_B_001: TOD;

HW_So_B_001: BOOL;

HW_Mo_B_001: BOOL;

HW_Di_B_001: BOOL;

HW_Mi_B_001: BOOL;

HW_Do_B_001: BOOL;

HW_Fr_B_001: BOOL;

HW_Sa_B_001: BOOL;

B_001_HW AT %Q*: BOOL;

(*Deklarationen Heisswasser Büro 002*)

Buero_002: FB_WeeklyTimeSwitch;

HW_Enable_B_002: BOOL;

HW_OnTime_B_002: TOD;

HW_OffTime_B_002: TOD;

HW_So_B_002: BOOL;

HW_Mo_B_002: BOOL;

HW_Di_B_002: BOOL;

HW_Mi_B_002: BOOL;

HW_Do_B_002: BOOL;

HW_Fr_B_002: BOOL;

HW_Sa_B_002: BOOL;

B_002_HW AT %Q*: BOOL;

END_VAR

(*Zeitschaltbaustein Büro 001*)

(*Aktivierung Zeitsteuerung Heißwasserbereiter Büro 001*)

(*Einschaltzeit für Heißwasserbereiter Büro 001*)

(*Ausschaltzeit für Heißwasserbereiter Büro 001*)

(*-----*)

(*-----*)

(* Auswahl der *)

(* Einschalttage *)

(* Büro 001 *)

(*-----*)

(*Schaltausgang Warmwasserbereiter Büro 001*)

(*Zeitschaltbaustein Büro 002*)

(*Aktivierung Zeitsteuerung Heißwasserbereiter Büro 002*)

(*Einschaltzeit für Heißwasserbereiter Büro 002*)

(*Ausschaltzeit für Heißwasserbereiter Büro 002*)

(*-----*)

(*-----*)

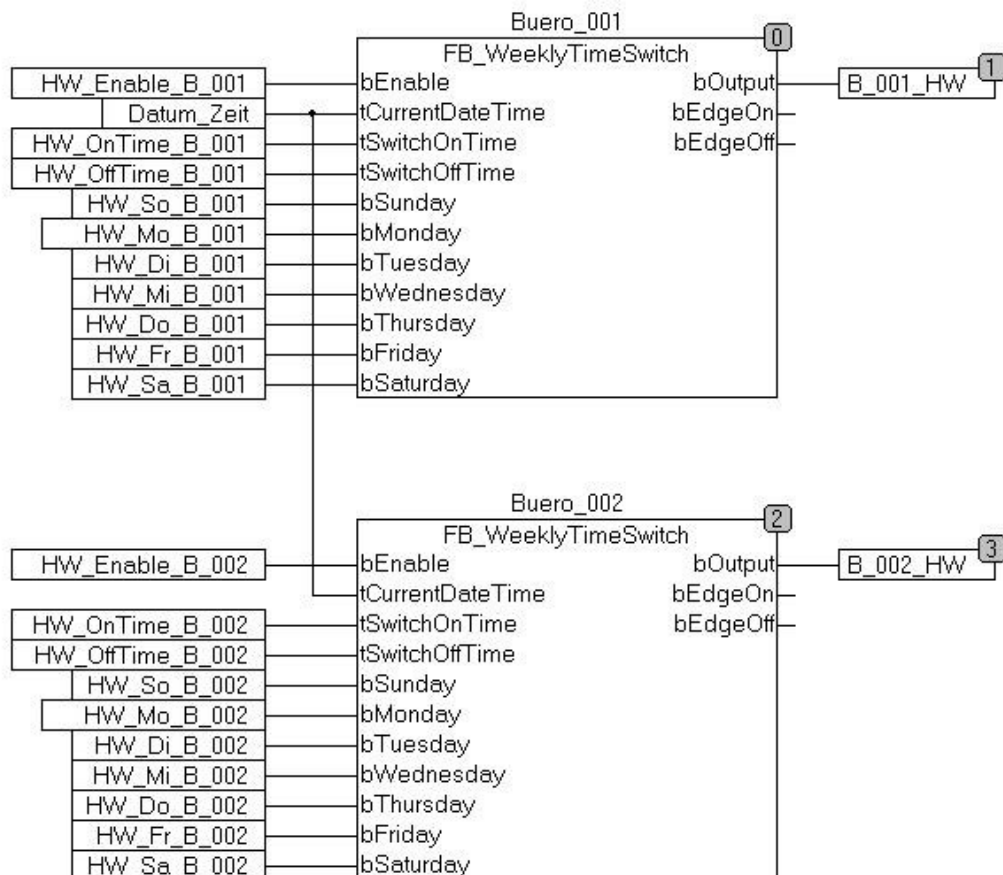
(* Auswahl der *)

(* Einschalttage *)

(* Büro 002 *)

(*-----*)

(*Schaltausgang Warmwasserbereiter Büro 002*)



Vergleich der Implementierung der Sensorik und Aktorik des KNX/EIB-Gebäudebussystems in
IEC 61131-basierte Gebäudeautomatisierungssysteme von WAGO und Beckhoff

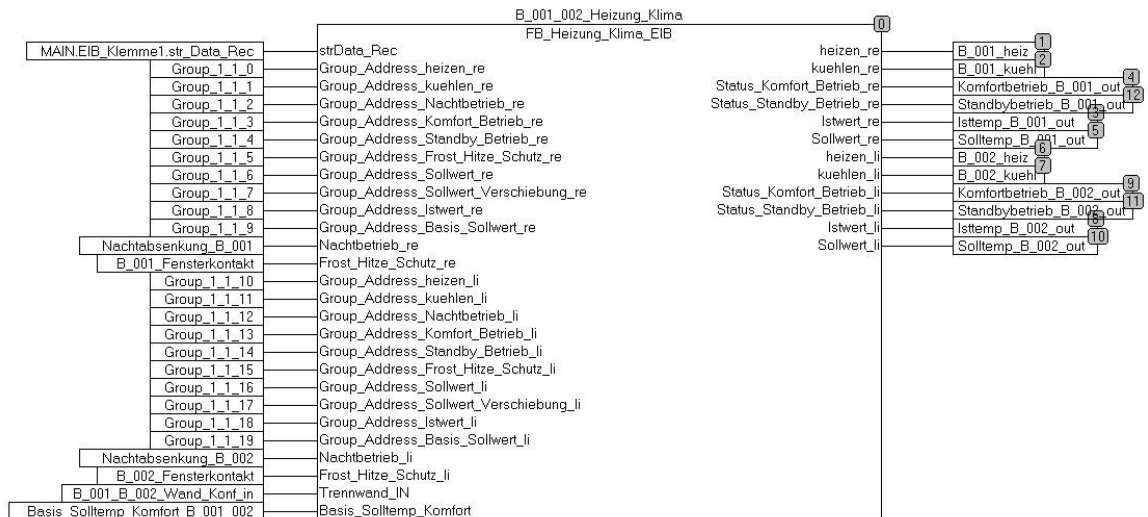
PROGRAM HeizungKlima_Buero_001_002

VAR

B_001_002_Heizung_Klima: FB_Heizung_Klima_EIB; Group_1_1_0: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=0); Group_1_1_1: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=1); Group_1_1_2: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=2); Group_1_1_3: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=3); Group_1_1_4: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=4); Group_1_1_5: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=5); Group_1_1_6: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=6); Group_1_1_7: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=7); Group_1_1_8: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=8); Group_1_1_9: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=9); Group_1_1_10: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=10); Group_1_1_11: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=11); Group_1_1_12: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=12); Group_1_1_13: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=13); Group_1_1_14: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=14); Group_1_1_15: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=15); Group_1_1_16: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=16); Group_1_1_17: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=17); Group_1_1_18: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=18); Group_1_1_19: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=1,NUMBER:=19); B_001_heiz AT %Q*: BOOL; B_002_heiz AT %Q*: BOOL; B_001_kuehl AT %Q*: BOOL; B_002_kuehl AT %Q*: BOOL; B_001_Fensterkontakt AT %I*: BOOL; B_002_Fensterkontakt AT %I*: BOOL; Nachtabsenkung_B_001: BOOL; Nachtabsenkung_B_002: BOOL; Komfortbetrieb_B_001_out: BOOL; Komfortbetrieb_B_002_out: BOOL; Standbybetrieb_B_001_out: BOOL; Standbybetrieb_B_002_out: BOOL; Isttemp_B_001_out: REAL; Isttemp_B_002_out: REAL; Solltemp_B_001_out: REAL; Solltemp_B_002_out: REAL; Basis_Solltemp_Komfort_B_001_002: REAL := 21;	(*Baustein für Heizung und Klima Büro 001 und Büro 002*) (*Gruppenadresse heizen Büro 001*) (*Gruppenadresse kühlen Büro 001*) (*Gruppenadresse Nachtbetrieb Büro 001*) (*Gruppenadresse Komfortbetrieb Büro 001*) (*Gruppenadresse Standbybetrieb Büro 001*) (*Gruppenadresse Frost-/Hitzeschutz Büro 001*) (*Gruppenadresse akt. Sollwert Büro 001*) (*Gruppenadresse Sollwert-Verschiebung Büro 001*) (*Gruppenadresse Istwert Büro 001*) (*Gruppenadresse Basis Sollwert Büro 001*) (*Gruppenadresse heizen Büro 002*) (*Gruppenadresse kühlen Büro 002*) (*Gruppenadresse Nachtbetrieb Büro 002*) (*Gruppenadresse Komfortbetrieb Büro 002*) (*Gruppenadresse Standbybetrieb Büro 002*) (*Gruppenadresse Frost-/Hitzeschutz Büro 002*) (*Gruppenadresse akt. Sollwert Büro 002*) (*Gruppenadresse Sollwert-Verschiebung Büro 002*) (*Gruppenadresse Istwert Büro 002*) (*Gruppenadresse Basis Sollwert Büro 002*) (*Schaltausgang Büro 001 heizen*) (*Schaltausgang Büro 002 heizen*) (*Schaltausgang Büro 001 kühlen*) (*Schaltausgang Büro 002 kühlen*) (*Eingang Fensterkontakt Büro 001*) (*Eingang Fensterkontakt Büro 002*) (*Nachtabsenkung Büro 001*) (*Nachtabsenkung Büro 002*) (*Status Komfort Betrieb Büro 001*) (*Status Komfort Betrieb Büro 002*) (*Status Standby Betrieb Büro 001*) (*Status Standby Betrieb Büro 002*) (*aktuelle Ist-Temperatur in Büro 001*) (*aktuelle Ist-Temperatur in Büro 002*) (*aktuelle Soll-Temperatur in Büro 001*) (*aktuelle Soll-Temperatur in Büro 002*) (*Basis Solltemperatur*)
--	---

END_VAR

Vergleich der Implementierung der Sensorik und Aktorik des KNX/EIB-Gebäudebussystems in IEC 61131-basierte Gebäudeautomatisierungssysteme von WAGO und Beckhoff



PROGRAM Jalousie_Bueros_001_002

VAR

```
(*Deklarationen Jalousie aussen Büros 001 & 002*)
Büro_001_Jalousie_aussen: FB_Jalousiesignale_EIB; (*Baustein EIB-Eingangssignale für die Aussenjalousie Büro
001*)
Büro_002_Jalousie_aussen: FB_Jalousiesignale_EIB; (*Baustein EIB-Eingangssignale für die Aussenjalousie Büro
002*)
Büro_001_002_Konferenz_aussen: FB_Konferenz_Jalousie; (*Baustein zur Steuerung der
Aussenjalousie im Konferenzraum-Modus*)
Wand_001_002_offen_Jal_aussen: FB_Position_Jalousie; (*Baustein zum Abgleich der Position der
Aussenjalousie im Konferenzraum-Modus*)
Büro_001_Aussenjalousie: FB_VenetianBlindEx; (*Jalousiebaustein für die Aussenjalousie Büro 001*)
Büro_002_Aussenjalousie: FB_VenetianBlindEx; (*Jalousiebaustein für die Aussenjalousie Büro 002*)
Group_1_0_2: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=0,NUMBER:=2); (*EIB
Gruppenadresse Aussenjalousie Büro 001 Schritt-Betrieb*)
Group_1_0_3: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=0,NUMBER:=3); (*EIB
Gruppenadresse Aussenjalousie Büro 001 Fahr-Betrieb*)
Group_1_0_6: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=0,NUMBER:=6); (*EIB
Gruppenadresse Aussenjalousie Büro 002 Schritt-Betrieb*)
Group_1_0_7: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=0,NUMBER:=7); (*EIB
Gruppenadresse Aussenjalousie Büro 002 Fahr-Betrieb*)
B_001_Jal_au_up AT %Q*: BOOL; (*Schaltausgang Aussenjalousie Büro 001
hochfahren*)
B_001_Jal_au_down AT %Q*: BOOL; (*Schaltausgang Aussenjalousie Büro 001
runterfahren*)
Position_Jal_au_B_001: USINT; (*Position Aussenjalousie Büro 001*)
B_002_Jal_au_up AT %Q*: BOOL; (*Schaltausgang Aussenjalousie Büro 002
hochfahren*)
B_002_Jal_au_down AT %Q*: BOOL; (*Schaltausgang Aussenjalousie Büro 002
runterfahren*)
Position_Jal_au_B_002: USINT; (*Position Aussenjalousie Büro 002*)
B_001_Beschattung: F_TRIG; (*Aussenjalousie Büro 001 hochfahren nach Beschattungsende*)
Beschattung_aussen_B_001: BOOL; (*Ansteuerung der Beschattung Aussenjalousie Büro
001*)
Beschattungs_Pos_aussen_B_001: USINT := 70; (*Angabe der Beschattungsposition Aussenjalousie Büro 001*)
B_002_Beschattung: F_TRIG; (*Aussenjalousie Büro 002 hochfahren nach Beschattungsende*)
Beschattung_aussen_B_002: BOOL; (*Ansteuerung der Beschattung Aussenjalousie Büro
002*)
Beschattungs_Pos_aussen_B_002: USINT := 70; (*Angabe der Beschattungsposition Aussenjalousie Büro 002*)

(*Deklarationen Jalousie innen Büros 001 & 002*)
Büro_001_Jalousie_innen: FB_Jalousiesignale_EIB; (*Baustein EIB-Eingangssignale für die Innenjalousie Büro
001*)
Büro_002_Jalousie_innen: FB_Jalousiesignale_EIB; (*Baustein EIB-Eingangssignale für die Innenjalousie Büro
002*)
Büro_001_002_Konferenz_innen: FB_Konferenz_Jalousie; (*Baustein zur Steuerung der Innenjalousie
im Konferenzraum-Modus*)
Wand_001_002_offen_Jal_innen: FB_Position_Jalousie; (*Baustein zum Abgleich der Position der
Innenjalousie im Konferenzraum-Modus*)
Büro_001_Innenjalousie: FB_VenetianBlindEx; (*Jalousiebaustein für die Innenjalousie Büro 001*)
```

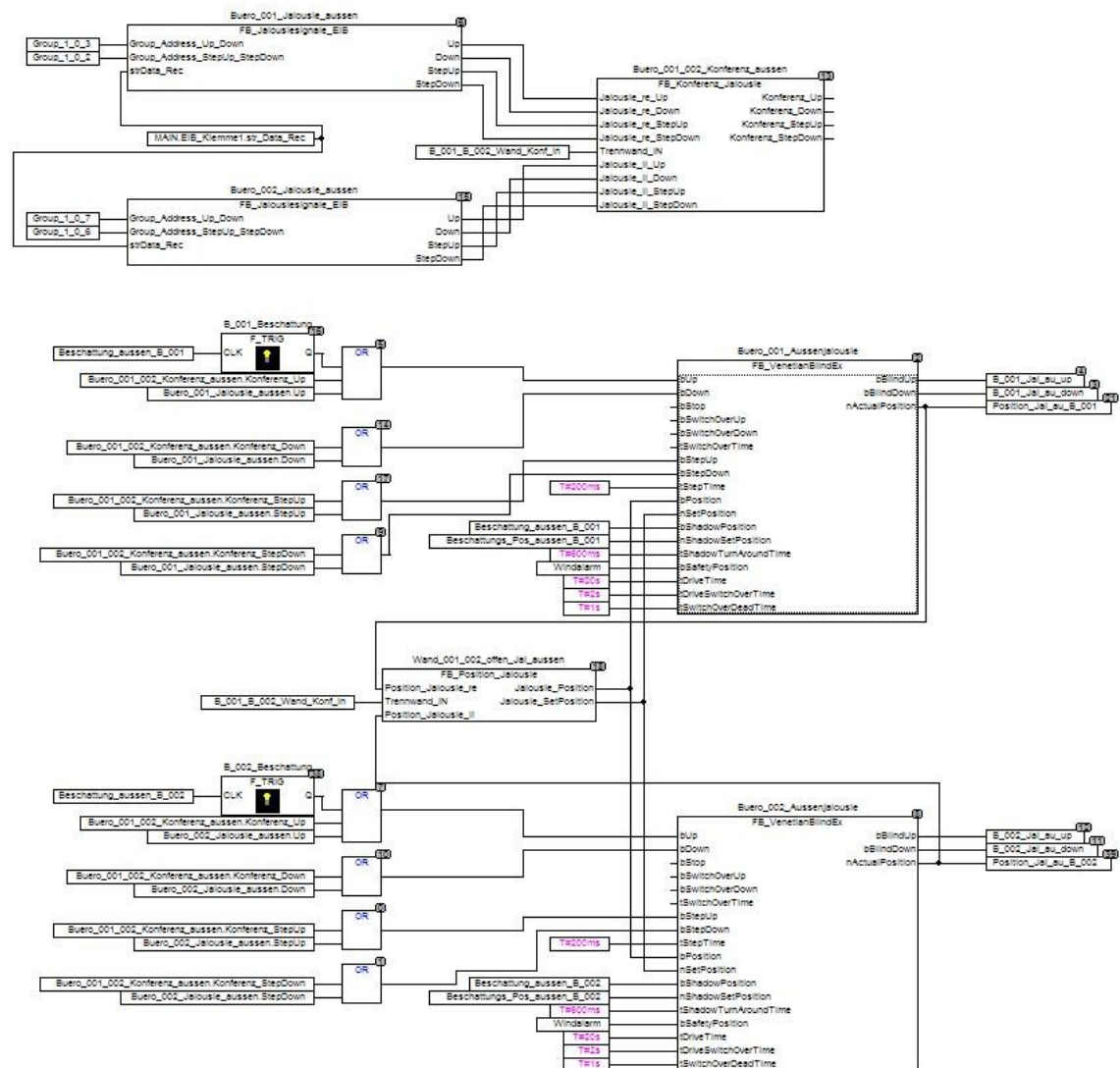
Vergleich der Implementierung der Sensorik und Aktorik des KNX/EIB-Gebäudebussystems in IEC 61131-basierte Gebäudeautomatisierungssysteme von WAGO und Beckhoff

```

Buero_002_Innenjalousie: FB_VenetianBlindEx;          (*Jalousiebaustein für die Innenjalousie Büro 002*)
Group_1_0_0: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=0,NUMBER:=0);          (*EIB
Gruppenadresse Innenjalousie Büro 001 Schritt-Betrieb*)
Group_1_0_1: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=0,NUMBER:=1);          (*EIB
Gruppenadresse Innenjalousie Büro 001 Fahr-Betrieb*)
Group_1_0_4: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=0,NUMBER:=4);          (*EIB
Gruppenadresse Innenjalousie Büro 002 Schritt-Betrieb*)
Group_1_0_5: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=0,NUMBER:=5);          (*EIB
Gruppenadresse Innenjalousie Büro 002 Fahr-Betrieb*)
B_001_Jal_in_up AT %Q*: BOOL;          (*Schaltausgang Innenjalousie Büro 001
hochfahren*)
B_001_Jal_in_down AT %Q*: BOOL;          (*Schaltausgang Innenjalousie Büro 001
runterfahren*)
Position_Jal_in_B_001: USINT;          (*Position Innenjalousie Büro 002*)
B_002_Jal_in_up AT %Q*: BOOL;          (*Schaltausgang Innenjalousie Büro 002
hochfahren*)
B_002_Jal_in_down AT %Q*: BOOL;          (*Schaltausgang Innenjalousie Büro 002
runterfahren*)
Position_Jal_in_B_002: USINT;          (*Position Innenjalousie Büro 002*)
END_VAR

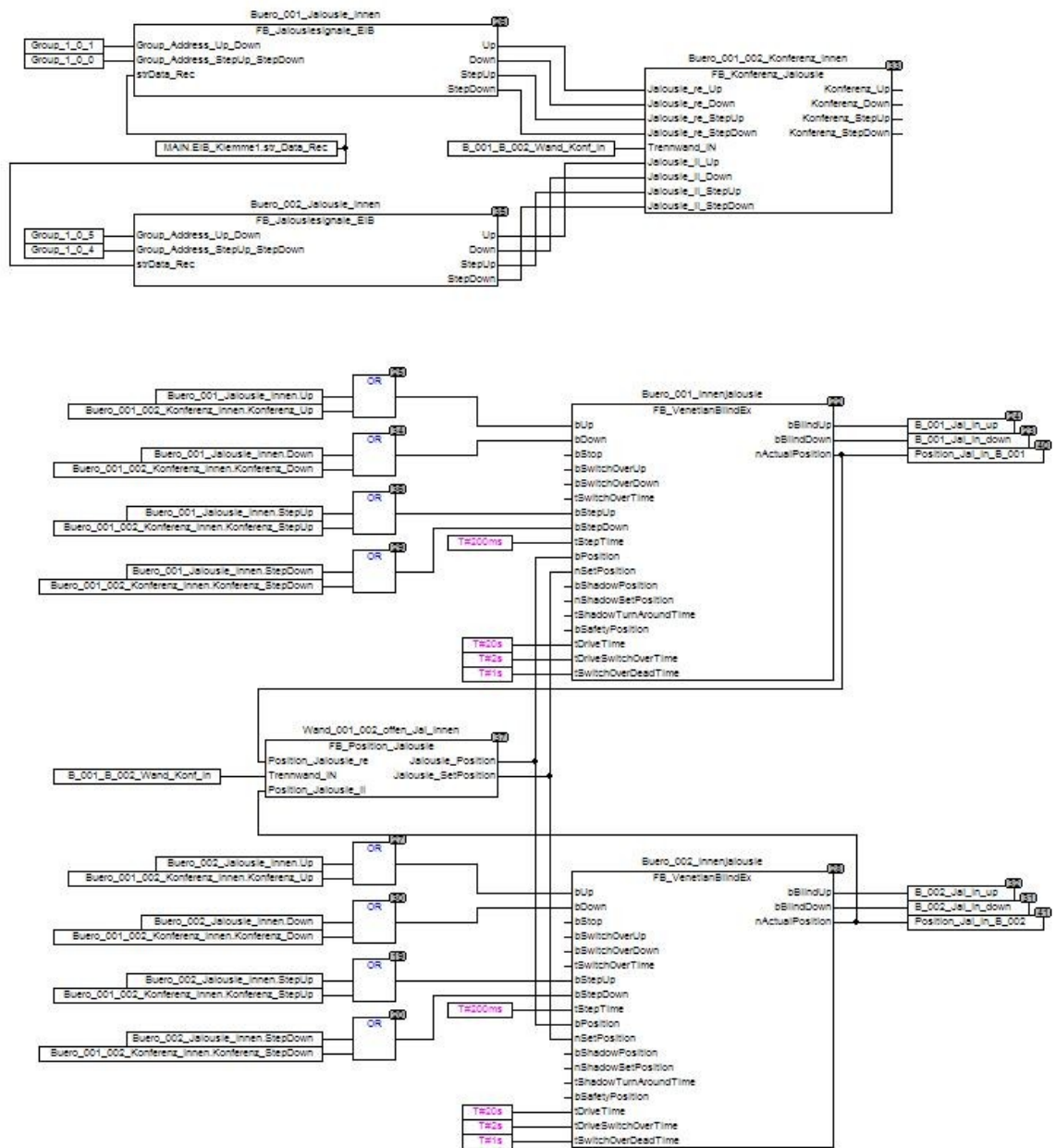
```

Büros 001 und 002 Jalousie aussen einzeln und Konferenz



Vergleich der Implementierung der Sensorik und Aktorik des KNX/EIB-Gebäudebussystems in IEC 61131-basierte Gebäudeautomatisierungssysteme von WAGO und Beckhoff

Büros 001 und 002 Jalousie innen einzeln und Konferenz



PROGRAM Leinwand_Bueros_001_002

VAR

(*Deklarationen Leinwand im Konferenzraum*)

Leinwand_001_002: FB_VenetianBlindEx;

(*Funktionsbaustein hoch und runter fahren der Leinwand*)

Leinwand_001_002_zu: F_TRIG;

(*Hochfahren wenn Trennwand

geschlossen wird*)

Leinwand_up_001_002: EIB_BIT_SEND;

(*EIB-Sendebaustein Leinwand hochfahren*)

Leinwand_down_001_002: EIB_BIT_SEND;

(*EIB-Sendebaustein Leinwand runterfahren*)

Group_Address_1_2_0: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=2,NUMBER:=0);

(*Gruppenadresse Leinwand hochfahren Büro 001 002*)

Group_Address_1_2_1: EIB_GROUP_ADDR := (MAIN:=1,SUB_MAIN:=2,NUMBER:=1);

(*Gruppenadresse Leinwand runterfahren Büro 001 002*)

Position_leinwand_001_002: USINT;

(*Position der Leinwand Büro 001 002*)

B_001_002_Leinwand_hoch_in AT %I*: BOOL;

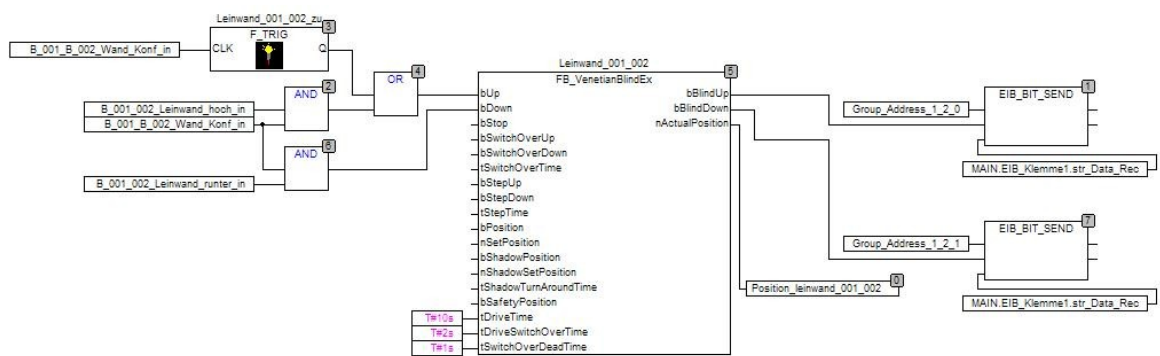
(*Tasteingang für Leinwand hoch im Büro 001 002*)

B_001_002_Leinwand_runter_in AT %I*: BOOL;

(*Tasteingang für Leinwand runter im Büro 001

002*)

END_VAR

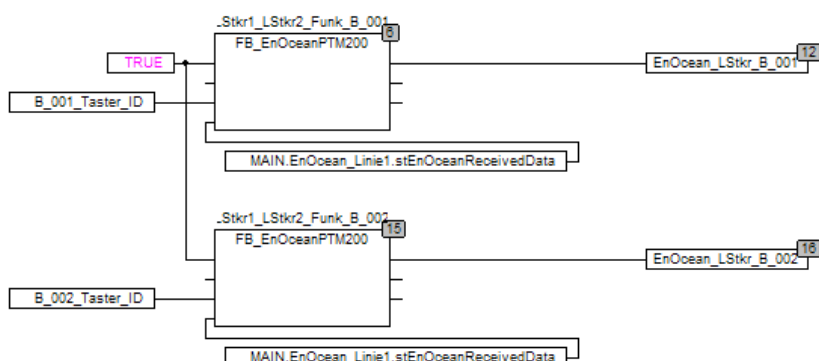
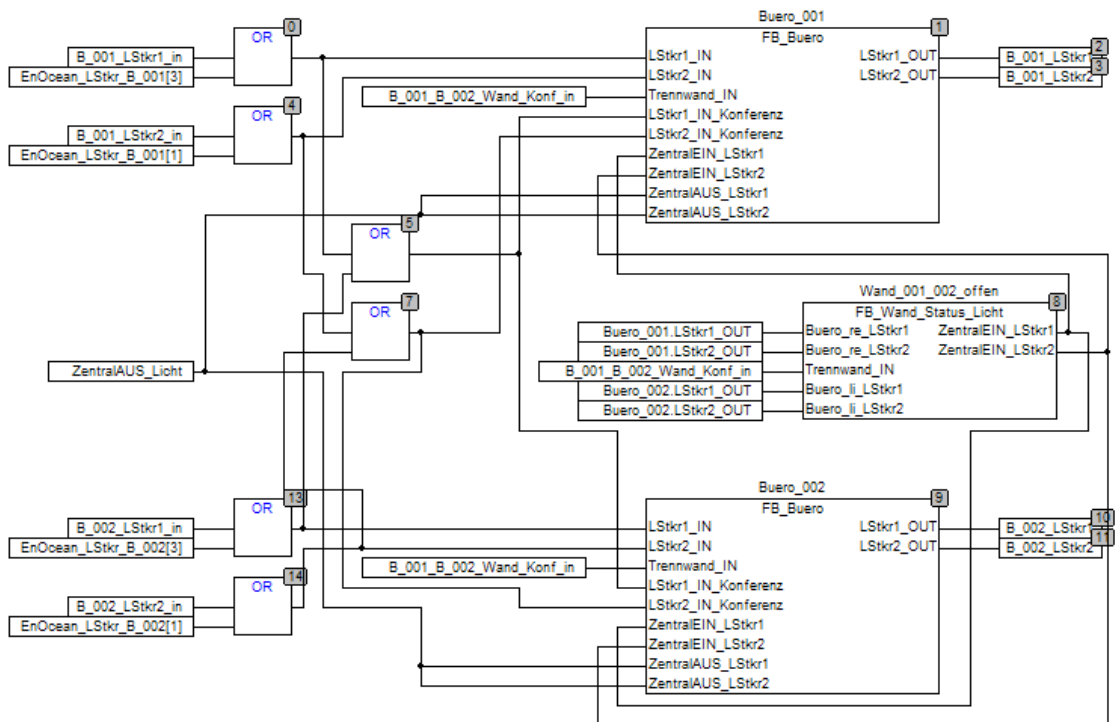


PROGRAM Licht_Bueros_001_002
VAR

(*Deklarationen Lichtstromkreise Büros 001 & 002*)
Buero_001: FB_Buero; (*Baustein zur Steuerung Lichtfunktionen im Büro 001*)
Buero_002: FB_Buero; (*Baustein zur Steuerung Lichtfunktionen im Büro 002*)
Wand_001_002_offen: FB_Wand_Status_Licht; (*Abgleich der Lichtstromkreise im Konferenzraum-Modus*)
B_001_LStkr1_in AT %I*: BOOL; (*Tasteingang für Lichtstromkreis 1 im Büro 001*)
B_001_LStkr2_in AT %I*: BOOL; (*Tasteingang für Lichtstromkreis 2 im Büro 001*)
B_002_LStkr1_in AT %I*: BOOL; (*Tasteingang für Lichtstromkreis 1 im Büro 002*)
B_002_LStkr2_in AT %I*: BOOL; (*Tasteingang für Lichtstromkreis 2 im Büro 002*)
B_001_LStkr1 AT %Q*: BOOL; (*Schaltausgang für Lichtstromkreis 1 im Büro 001*)
B_001_LStkr2 AT %Q*: BOOL; (*Schaltausgang für Lichtstromkreis 2 im Büro 001*)
B_002_LStkr1 AT %Q*: BOOL; (*Schaltausgang für Lichtstromkreis 1 im Büro 002*)
B_002_LStkr2 AT %Q*: BOOL; (*Schaltausgang für Lichtstromkreis 2 im Büro 002*)

(*Deklarationen EnOcean*)

LStkr1_LStkr2_Funk_B_001: FB_EnOceanPTM200; (*Baustein für den Empfang der EnOcean
TasterSignale für Büro 001*)
LStkr1_LStkr2_Funk_B_002: FB_EnOceanPTM200; (*Baustein für den Empfang der EnOcean
TasterSignale für Büro 002*)
B_001_Taster_ID: UDINT := 1140678; (*ID des EnOcean Tasters für Büro 001*)
B_002_Taster_ID: UDINT := 1141737; (*ID des EnOcean Tasters für Büro 002*)
EnOcean_LStkr_B_001: ARRAY [0..3] OF BOOL; (*Tasten des EnOcean Tasters für Büro 001*)
EnOcean_LStkr_B_002: ARRAY [0..3] OF BOOL; (*Tasten des EnOcean Tasters für Büro 002*)
END_VAR



PROGRAM Licht_Flur_001_002

VAR

(*Deklarationen Lichtstromkreis 1 Flur 001 002*)

LStkr1: FB_FlurLicht;

(*Baustein zur Steuerung des

Lichtstromkreises 1 im Flur 001 002*)

FI_001_002_LStkr1_in AT %I*: BOOL;

(*Taster zum einschalten des Lichtstromkreises 1 im Flur 001

002*)

FI_001_002_LStkr1 AT %Q*: BOOL;

(*Schaltausgang des Lichtstromkreises 1 im Flur 001 002*)

(*Deklarationen Lichtstromkreis 2 Flur 001 002*)

LStkr2: FB_FlurLicht;

(*Baustein zur Steuerung des

Lichtstromkreises 2 im Flur 001 002*)

FI_001_002_LStkr2_in AT %I*: BOOL;

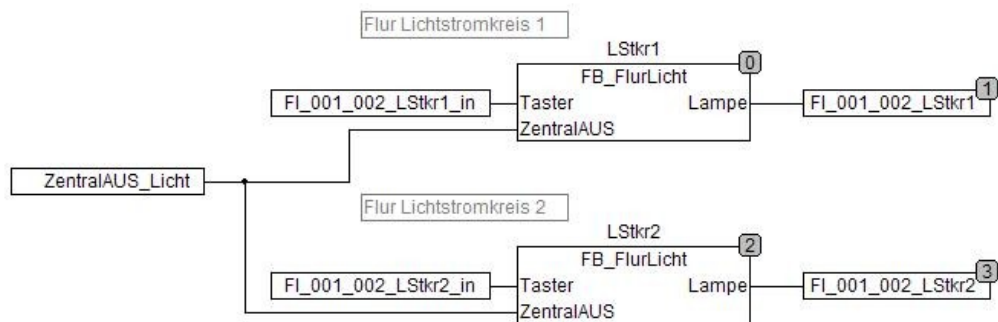
(*Taster zum einschalten des Lichtstromkreises 2 im Flur 001

002*)

FI_001_002_LStkr2 AT %Q*: BOOL;

(*Schaltausgang des Lichtstromkreises 2 im Flur 001 002*)

END_VAR



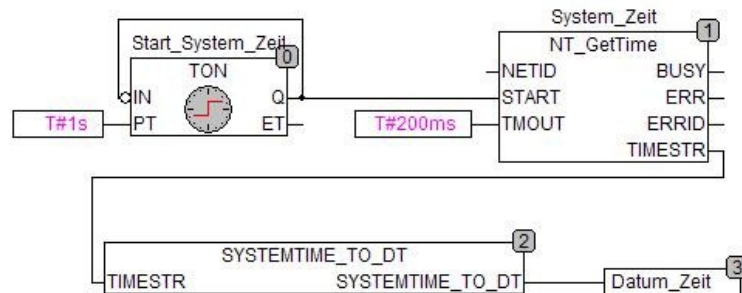
PROGRAM Uhr

VAR

System_Zeit: NT_GetTime;

Start_System_Zeit: TON;

END_VAR



PROGRAM MAIN

VAR

(*Deklarationen EnOcean Initialisierung*)

EnOcean_Linie1: FB_EnOceanReceive;

EnOcean_Enable: BOOL := TRUE;

EnOcean_Error: BOOL;

EnOcean_DataIn AT %I*: ST_EnOceanInData;

EnOcean_DataOut AT %Q*: ST_EnOceanOutData;

(*Deklarationen EIB Initialisierung*)

EIB_Klemme1: KL6301;

EIB_Activate: BOOL := TRUE;

EIB_Phys_Addr: EIB_PHYS_ADDR := (Area:=1,Line:=1,Device:=1);

EIB_Group_Filter: ARRAY [1..4] OF EIB_GROUP_FILTER :=

(GROUP_ADDR:=(MAIN:=1,SUB_MAIN:=0,NUMBER:=0),GROUP_LEN:=63),

(GROUP_ADDR:=(MAIN:=1,SUB_MAIN:=1,NUMBER:=0),GROUP_LEN:=63),

(GROUP_ADDR:=(MAIN:=1,SUB_MAIN:=2,NUMBER:=0),GROUP_LEN:=63),

(GROUP_ADDR:=(MAIN:=1,SUB_MAIN:=1,NUMBER:=64),GROUP_LEN:=63);

EIB_Active: BOOL;

EIB_Ready: BOOL;

KL6301_IN AT %I*: ARRAY [1..24] OF BYTE;

KL6301_OUT AT %Q*: ARRAY [1..24] OF BYTE;

END_VAR

(*Initialisierung EnOcean Klemme 1*)

EnOcean_Linie1(

bEnable:= EnOcean_Enable,

stEnOceanInData:= EnOcean_DataIn,

stEnOceanOutData:= EnOcean_DataOut,

bError=> EnOcean_Error,

nErrorId=> ,

stEnOceanReceivedData=>);

(*Initialisierung EIB Klemme 1*)

EIB_Klemme1(

bActivate:= EIB_Activate,

idx:= 1,

EIB_PHYS_ADDR:= EIB_Phys_Addr,

EIB_GROUP_FILTER:= EIB_Group_Filter,

KL6301_IN:= KL6301_IN,

KL6301_OUT:= KL6301_OUT,

bActive=> EIB_Active,

bReady=> EIB_Ready,

bError=> ,

iErrorId=> ,

str_Data_Rec=>);

(*Aufrufe allgemeine Programme*)

Uhr();

(*Aufrufe Büroprogramme*)

Heisswasser_Bueros_001_002();

HeizungKlima_Buero_001_002();

Jalousie_Bueros_001_002();

Licht_Bueros_001_002();

Leinwand_Bueros_001_002();

Heisswasser_Bueros_003_004();

HeizungKlima_Buero_003_004();

Jalousie_Bueros_003_004();

Licht_Bueros_003_004();

Leinwand_Bueros_003_004();

Heisswasser_Bueros_005_006();

HeizungKlima_Buero_005_006();

Jalousie_Bueros_005_006();

Licht_Bueros_005_006();

Leinwand_Bueros_005_006();

Heisswasser_Bueros_007_008();

HeizungKlima_Buero_007_008();

Jalousie_Bueros_007_008();

Licht_Bueros_007_008();

Leinwand_Bueros_007_008();

(*Aufrufe Flurprogramme*)

Licht_Flur_001_002();

Licht_Flur_003_004();

Licht_Flur_005_006();

Licht_Flur_007_008();