iQST GmbH

Institute for Quality, Safety and Transportation

Geräteentwicklung für ein integrales und energie-sparendes Feldbussystem

Abschlussbericht über ein Entwicklungsprojekt,

gefördert unter dem AZ: 26964-21/0 von der

Deutschen Bundesstiftung Umwelt

Von

Dr.-Ing. Uwe Becker, Dr.-Ing. Tobias Ständer,

Steffen Bussenius, Sebastian Knüfermann

Braunschweig, April 2012

iQST GmbH

Institute for Quality, Safety and Transportation

Geräteentwicklung für ein integrales und energie-sparendes Feldbussystem

Abschlussbericht über ein Entwicklungsprojekt,

gefördert unter dem AZ: 26964-21/0 von der

Deutschen Bundesstiftung Umwelt

Von

Dr.-Ing. Uwe Becker, Dr.-Ing. Tobias Ständer,

Steffen Bussenius, Sebastian Knüfermann

Braunschweig, April 2012

06/02

Projektkennblatt
der
Deutschen Bundesstiftung Umwelt

Az	26964-21/0	Referat	21	Fördersumme	124.200,00 €

Antragstitel Geräteentwicklung für ein integrales und energiesparendes Feldbussystem

Stichworte

Laufzeit	Projektbeginn	Projektende	Projektphase(n)
2 Jahre	**3.3.2009**	**30.09.2011**	

Zwischenberichte

Bewilligungsempfänger iQST GmbH
Institute for Quality, Safety and Transportation
Hermann-Blenk-Str. 22
38108 Braunschweig

Tel 01635768550
Fax 0531 35444-16

Projektleitung
Dr.-Ing. Uwe Becker

Bearbeiter

Kooperationspartner

Zielsetzung und Anlaß des Vorhabens

Energie ist eine der wichtigsten Grundlagen moderner Gesellschaften, deren verantwortliche Nutzung aufgrund begrenzter Ressourcen immer dringlicher wird. Auch so genannte Kleinverbraucher benötigen infolge ihrer hohen Anzahl nennenswerte Energie. Insbesondere bei Geräten und Installationen mit sehr langer Betriebsdauer über mehrere Jahrzehnte sind (Fehl-)Entscheidungen und Investitionen unter energetischen Gesichtspunkten zu treffen. Zu diesen gehören elektrische Gebäudeinstallationen, die in hoher Stückzahl, fest montiert, langfristig hohe Energieverluste akkumulieren. Anreize zur Installation verbrauchsarmer Einrichtungen motivieren jedoch nur bei attraktiven Einstandspreisen bzw. Vertriebsstrategien. Mit dem neuen integrierten Gebäudeautomatisierungssystem SmallCAN soll der technologisch bedingte minimale Energieverbrauch adressiert werden, der bei einer attraktiven Preisbildung und mittels eines neuartigen Vertriebsgeschäftsmodells einen leichten Marktzugang ermöglicht. Arbeitsziel des Projektes ist, ausgehend von einer vorhandenen stabilen Grundlagenentwicklung eines energieminimalen Kommunikationssystems für die Gebäudetechnik und -automatisierung, das System durch Weiterentwicklung zur Praxistauglichkeit zu führen.

Darstellung der Arbeitsschritte und der angewandten Methoden

Neben der Vervollständigung des grundlegenden Systems ist eine Erprobung unter realistischen Bedingungen anhand einer Demonstrationsinstallation vorgesehen, welche die Praxistauglichkeit sowohl für den professionellen als auch den privaten Domotik-Markt nachweist. Zusätzlich soll die für den Praxiseinsatz notwendige Vielfalt und Bandbreite an Applikationsmodulen geschaffen werden.

Ergebnisse und Diskussion

Ausgehend von der Zielsetzung der Energieersparung und Integration der Gebäudeautomatisierung mit Bereitstellung neuer Funktionalitäten bei gleichzeitiger Kostenminimierung war es das Ziel, ein zuverlässiges und universelles low-Cost Feldbussystem für die Gebäudetechnik bereit zu stellen, dass sich durch geringen Energiebedarf und fast beliebige Erweiterbarkeit auszeichnet. Das vorhandene prototypische System ist in ersten Schritten zur Praxistauglichkeit weiter entwickelt worden. Die Server-Client-Architektur ist aufgrund der Anforderungen aus dem FutureWorkspace überarbeitet worden. Ebenfalls werden noch die Visualisierung überarbeitet.
Der Buskoppler selber ist inzwischen in einer ersten Auflage mit 1000 Einheiten automatisiert gefertigt und bestückt worden. Von einigen Applikationen sind Kleinserien von 5- 20 Einheiten gefertigt.
Die Softwareentwicklung ist durch die Neuorientierung und -strukturierung noch nicht endgültig abgeschlossen und wird weiter entwickelt auch für neue Applikationen.
Die ersten Messergebnisse aus dem „future-workspace“ (Implementierung in einem Büro der Zukunft im Wettbewerb zu anderen „Gebäudeautomatisierern“, Entwicklung von ca. 40 Anwendungsadaptern) sind sehr vielversprechend gerade in Hinsicht auf den Standby-Verbrauch und im direkten Vergleich zu den Mitbewerbern des Ausbaus.
Durch diese grundlegenden Vorarbeiten ist inzwischen eine weitere Förderung durch das BMWi erfolgt.
Weiter kann das System auch zur Komfortsteigerung, insbesondere durch neue Dienste zur Unterstützung von behinderten oder älteren Menschen dienlich sein. Hierzu wird eine Kooperation mit dem Braunschweiger Informatik- und Technologie-Zentrum, die BITZ GmbH angestrebt. Hier ist insbesondere das Projekt ehealth.Braunnschweig (http://www.ehealthbraunschweig.de/) zu nennen.
Erste Schritte in Richtung Ausbildung sind durchgeführt.
Prof. Dr.-Ing. Yongjian DING von der Fachhochschule Magdeburg Stendal wir im Frühjahr 2011 ein kleine Demonstrationswand erhalten um diese in der Lehre in einem Labor einzusetzen

Öffentlichkeitsarbeit und Präsentation

Diekhake, P.; Fähndrich, E.; Schnieder, E.; Becker, U.: SmallCAN: Integrierte Gebäudeautomatisierung durch einheitliches low-Power, low-Cost, OpenSource Feldbussystem.
Automation 2011, Baden-Baden, Deutschland, Juni 2011.

Einbau des SmallCAN im FutureWorkspace der TU Braunschweig in einem Büroraum.

Fazit

Durch die Förderung der DBU ist die Möglichkeit geschaffen worden, aus der Idee des SmallCAN mit einigen Prototypen erste Ergebnisse in Richtung eines Produktes zu erzielen. Durch diese Vorarbeiten ist es gelungen weiter Förderprojekte (Seit dem 1.10.2011 wird das BMWi-Projekt „Digaflex - Demonstrationsanlage einer integrierten Gebäudeautomatisierung mit low-Power, low-Cost Ansatz und flexiblem Gerätespektrum und flexibler Konfiguration“, Kennzeichen 03ET1016A gefördert. Dieses Projekt ist gemeinsam mit dem Institut für Verkehrssicherheit und Automatisierungstechnik beantragt worden. Arbeitsziel des Projektes ist es in zwei beispielhaften Demonstrationsinstallationen in ausgedehnten Liegenschaften die Vorteile über einen längeren Zeitraum zu demonstrieren.
Die Laufzeit beträgt 5 Jahre, wobei die letzten beiden Jahre ausschließlich dem Monitoring des installierten Systems dienen) zu erhalten sowie erste Schritte in Richtung Ausbildung durchzuführen

Inhalt

Verzeichnis Bilder

Verzeichnis Begriffe, Abkürzungen und Definitionen

GUI - Graphical User Interface

MBus - (Meter-Bus) ist ein Feldbus für die Verbrauchsdatenerfassung.

PIC - Peripheral Interface Controller

XML - xtensible Markup Language

Zusammenfassung

Energie ist eine der wichtigsten Grundlagen moderner Gesellschaften, deren verantwortliche Nutzung aufgrund begrenzter Ressourcen immer dringlicher wird. Insbesondere in Gebäuden lässt sich der Energiebedarf durch intelligente Lösungen wie z.B. Gebäudeautomatisierungssysteme (GA-Systeme) verbessern deren intelligente Vernetzung eine hohe Primärenergieeinsparung ermöglicht. GA-Systeme erfordern jedoch signifikante Kosten für Investition, Planung (Engineering), Installation und Betrieb (Energieverbrauch der Busteilnehmer, Wartungskosten) sowie bei Änderungen.

In dem Projekt „Geräteentwicklung für ein integrales und energiesparendes Feldbussystem“ konnte der praktische Einsatz des sich in der Grundlagenentwicklung befindliche energieoptimierte und kostenminimale Kommunikationssystem SmallCAN praktisch vollzogen werden. Die Vorteile dieses Kommunikationssystems gegenüber konventionellen Systemen konnte in dem Installationssobjekt zusammen mit dem Institut für Verkehrssicherheit und Automatisierungstechnik der TU Braunschweig im „future-workspace“ demonstriert werden. Es ist ein Büroraum im Wettbewerb mit etablierten GA-Ausrüstern aufgebaut worden. Die Verbrauchsmessungen werden z.Z. durchgeführt und erste Auswertungen sind für den Sommer 2012 zu erwarten. Mit der Weiterentwicklung des Systems für praxisnahe Gebäudeautomatisierungsanwendungen konnte ein umfangreiches Portfolio verschiedenster Anwendungsmodule geschaffen werden um Einzelkomponenten wie Pumpen, Ventile oder auch Lüfteranlagen individuell anzusteuern. Neue regelungstechnische Aufgaben der Gebäudeautomatisierung (z.B. Beispiel eine Helligkeitsregelung) können durch Zuhilfenahme vieler Akteure und deren kooperatives Zusammenspiel in modular entwickelten Softwaremodulen geschaffen werden. Durch einen externen Zugriff auf das System ist die vollständige Überwachung und Erprobung dieser Demonstrations-Installationen möglich. Durch den durch erste Messungen gezeigten geringen Energieverbrauch selbst sowie die intelligente Kooperation der Systemkomponenten durch flexibel konfigurierbaren und umfangreich auslegbaren Verarbeitungsfunktionen ist hinsichtlich der Energieeinsparungen von Gebäuden ein hohes Potenzial zu erwarten. Das Kommunikationssystem SmallCAN konnte hier erfolgreich in der Praxis umgesetzt werden. Damit lässt sich das System für viele weitere Anwendungsmöglichkeiten nutzen.

1 Einleitung

In vielen Bereichen der Gebäudetechnik werden Automatisierungssysteme angeboten, deren Nutzen auf die jeweilige Anwendung eingeschränkt ist. Die Stromversorgung dieser Systeme ist häufig dezentral und damit teuer (lokale Netzteile, Batterien), die Verkabelung sternförmig oder hierarchisch und damit aufwändig, eine Skalierung oder Erweiterung nicht oder lediglich mit hohem Aufwand möglich. Zu diesen Nachteilen der teils proprietären Pro- dukte werden häufig noch Probleme mit dem Datenübertragungsmedium beschrieben, ins- besondere bei Mitnutzung der Stromleitungen.

Auch so genannte Kleinverbraucher benötigen infolge ihrer hohen Anzahl nennenswerte Energie, wie es mittlerweile bei Glühlampen und Stand-by-Betrieb von Computern und elektronischen Unterhaltungsgeräten bekannt ist. Daher ist auch die Vorgabe von Energieeffizienz-Labeln mittlerweile öffentlicher Ausdruck eines sensiblen Energiebewusstseins bei elektrischen Alltagsgeräten geworden.

Mit einem neuen integrierten Gebäudeautomatisierungssystem soll neben der Energieeinsparung durch optimierte Betriebsführung auch der Eigen-Energieverbrauch des Systems selbst minimiert werden (geringer „Standby-Bedarf“ des Gebäudes). Wenn man sich vergegenwärtigt, dass in einem Gebäude mit mehreren hundert oder gar tausend Busgeräten die Busknoten 24 Stunden mit Energie versorgt werden müssen um die Funktion des Systems zu ermöglichen, so wird schnell deutlich, dass der Energiebedarf pro Busknoten von entscheidender Bedeutung von den „Standby-Verbrauch“ eines Gebäudes ist. Abgesehen von den batteriebetriebenen Geräten einiger Funk-Systeme, die in regelmäßigen Intervallen den Austausch von hunderten Batterien mit entsprechend negativen Umwelteffekten und Austauschkosten nach sich ziehen, weisen die bisherigen Systeme einen erheblichen Energiebedarf aus.

Derzeit existiert eine Reihe von spezialisierten Bussystemen der entsprechenden Gebäude- und Automatisierungsbranchen, die jeweils lediglich ihr Teilsegment der Gesamtheit aller Anwendungen abdecken. Diese Systeme sind zueinander inkompatibel und technisch so stark auf ihr Teilsegment ausgerichtet, so dass keine gemeinsame Basis gegeben ist.

Der praktische Einsatz dieser Systeme beschränkt sich derzeit auf einen geringen Anteil der potentiellen Gebäude, da aufgrund der nicht unerheblichen Kosten lediglich wenige Industrie- und Nutzbauten damit ausgerüstet werden. Dabei kommt es aufgrund der Teilabdeckungen häufig zu einer Parallelinstallation mehrerer Systeme, gegebenenfalls ergänzt um industrielle Prozessleittechnik (SPS-Schränke).

Der Eigenenergiebedarf der bisher installierten Systeme ist erheblich. Zur Energieversorgung der Komponenten werden meist lokale Kleinstnetzteile genutzt, deren Effizienz und Zuverlässigkeit begrenzt ist.

Gerade bei Geräten und Installationen mit sehr langer Betriebsdauer über mehrere Jahrzehnte sind Investitionen unter energetischen Gesichtspunkten als auch Wartungs und Installationskosten zu treffen. Zu diesen gehören elektrische Gebäudeinstallationen, die in hoher Stückzahl, fest montiert, langfristig hohe Energieverluste akkumulieren. Anreize zur Installation verbrauchsarmer Einrichtungen motivieren jedoch nur bei attraktiven Einstandspreisen bzw. Vertriebsstrategien.

Eine Integration aller Bereiche der Gebäudetechnik, sowohl gerätetechnisch als auch durch eine konsequent einfache und jederzeit erweiterungsfähige Verkabelungsstruktur, ist dabei Voraussetzung für eine intelligente Steuerung und Vernetzung des gesamten Gebäudes insbesondere unter energetischen Gesichtspunkten.

Mit dem neuen integrierten Gebäudeautomatisierungssystem SmallCAN soll der technologisch bedingte minimale Energieverbrauch adressiert werden, der bei einer attraktiven Preisbildung und mittels eines neuartigen Vertriebsgeschäftsmodells einen leichten Marktzugang ermöglicht.

Ausgehend von der Zielsetzung der Energieersparung und Integration der Gebäudeautomatisierung mit Bereitstellung neuer Funktionalitäten bei gleichzeitiger Kostenminimierung ist es das Ziel, ein zuverlässiges und universelles low-Cost Feldbussystem für die Gebäudetechnik bereit zu stellen, dass sich durch geringen Energiebedarf und fast beliebige Erweiterbarkeit auszeichnet.

Das vorhandene prototypische System soll zur Praxistauglichkeit entwickelt werden. Der

universelle Ansatz des Systems vereint die Vorteile der bisher verfügbaren Lösungen in einem einzigen System und ermöglicht somit eine Zusammenführung. Insbesondere der low-Cost Grundsatz und eine fast beliebige, auch nachträgliche, Erweiterbarkeit prädestinieren das System für einen breiten Einsatz in der Gebäudetechnik (bis zu 1000 Busteilnehmer an bis zu 1000 m Telefonkabel). Das System soll zuverlässig und wartungsfrei (kabelgebunden), vollständig dezentralisiert (ohne zentrale Steuereinheit, multimasterfähig) und ausreichend leistungsfähig sein.

Eine Erprobung des Systems unter realistischen Bedingungen soll im Labor durchgeführt sowie in der beispielhaften Installation validiert werden. Die daraus gewonnen Erkenntnisse sollen systematisch aufbereitet werden und in die Entwicklung zurück fließen.

Am Ende soll ein vollständiges und alle üblichen Applikationen abdeckendes praxistaugliches System in industrieller Qualität bereit stehen. Dies umfasst sowohl das eigentliche Bussystem, die diversen Applikationsadapter sowie die Software zur Konfiguration und Inbetriebnahme (im ersten Schritt tauglich für Fachinstallateure, später auch für Endkunden) beziehungsweise zur Visualisierung und Bedienung durch den Endkunden.

2 Geräteentwicklung Hard und Software

Ausgehend von einer vorhandenen stabilen Grundlagenentwicklung eines energieminimalen Kommunikationssystems, sollte durch Weiterentwicklung des Systems ein praxistaugliches Automatisierungssystem zur Verfügung gestellt werden.

Neben der Vervollständigung des grundlegenden Systems und Sicherung der Qualität ist eine Erprobung unter realistischen Bedingungen anhand einer Demonstrationsinstallation vorgesehen, welche die Praxistauglichkeit sowohl für den professionellen als auch den privaten Domotik-Markt nachweist.

Zusätzlich soll die für den Praxiseinsatz notwendige Vielfalt und Bandbreite an Applikationsmodulen geschaffen werden. Die Vorteile des SmallCAN Systems werden in einer exemplarischen Installation im Rahmen des Projektes „future-workspace“ in Zusammenarbeit mit dem Institut für Verkehrssicherheit und Automatisierungstechnik demonstriert (siehe Abbildung 1).

Abbildung 1: SmallCAN in der Demonstartionsanlage „future-workspace“. [Quelle: Hans Georg Esch fotografiert für Saint-Gobain Ecophon GmbH]

2.1 Hardwareentwicklung & Firmware

Für ein Laborsystem wurden insgesamt ca 40. Anwendungsadapter erstellt und auf eine Laborwand aufgebaut. Die Laborwand ist modular aufgebaut, womit sich das dort verbaute System zu jedem Zeitpunkt skalieren und umbauen lässt. An der Laborwand konnten neu entwickelte Anwendungsmodule hinsichtlich ihrer Funktion, sowie im Verbund mit anderen Busteilnehmern getestet werden und ggf. Anpassungen an Hardware und Software vorgenommen werden.

Für den Einbau von SmallCAN-Komponenten in reale Gebäudestrukturen wurde im Demonstrationssystem „future-workspace“ ein Büro mit dem Feldbussystem SmallCAN ausgestattet. Hier konnte in der Praxis die Bearbeitung von mess- und regelungstechnischen Aufgaben der Gebäudeautomatisierung durch das SmallCAN System getestet werden. Dazu mussten zunächst die Entwicklung und der Aufbau aller notwendigen Anwendungsadapter sowie deren vollständige Installation in das Demonstrationssystem erfolgen. Insgesamt sind im „future-workspace“ 86 Anwendungsmodule verbaut, die Sensoren und Aktoren verschiedenster Art über Telefonleitungen vernetzen.

2.1.2 Beispielhafte Beschreibung des Anwendungsmoduls für Wärmemengenzähler

Die Hardwareentwicklungsarbeiten sollen exemplarisch an der Ansteuerung eines Wärmemengenzählers der Firma Allmess durch ein SmallCAN-Anwendungsmodul erläutert werden. In Abbildung 2 ist der Wärmemengenzähler (WMZ) CF-Echo dargestellt, der zur Erfassung thermischer Größen wie angegebener Energie und Leistung von Klimageräten u.ä. genutzt werden kann. Der WMZ besteht aus einem Durchfluss-Sensor, Temperatursensoren zur Erfassung der Vorlauf- und Rücklauf-Temperaturen sowie ein Rechenwerk zur Bestimmung der thermischen Größen.

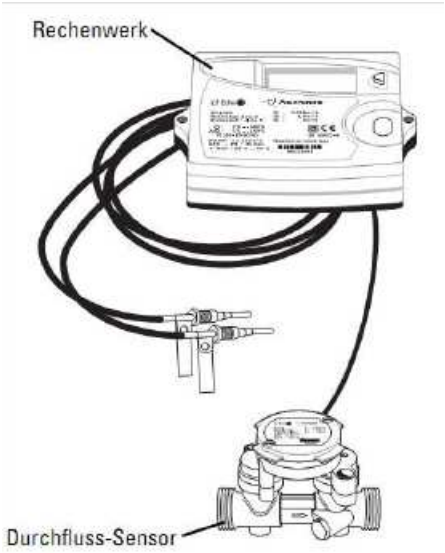


Abbildung 2: Wärmemengenzäher CF Echo der Fa. Allmess. [Quelle: Allmess]

Optional lässt sich ein Erweiterungsmodul auf das Rechenwerk aufstecken, um eine Schnittstelle an externe Geräte zu ermöglichen. Das MBUS-Schnittstellenmodul bietet die Möglichkeit über den etablierten Verbrauchsdatenerfassungsbus MBUS Daten extern bereitzustellen. Zur Erfassung wichtiger thermischer Größen in Gebäuden ist daher ein Anwendungsmodul für SmallCAN entwickelt worden, welches als Gateway zu MBUS fähigen Geräten der Gebäudeautomatisierung fungiert. Das Anwendungsmodul wird im Falle des Wärmemen-

genzählers CF-Echo an das MBUS-Schnittstellenmodul angeschlossen um mit dem Gerät zu kommunizieren.

Bei der Entwicklung der Hardwarekomponente ist die Einhaltung der elektrischen Übertragungsregeln des MBUS zu berücksichtigen. Nach den in der Bitübertragungsschicht des ISO- OSI Schichtenmodells einzuordnenden Spezifikationen ist der Schaltplan nach Abbildung 3 erstellt worden. Das Hardwaremodul ermöglicht weiterhin den Wärmemengenzähler mit Energie zu versorgen und macht ihn durch den Verzicht des internen Energiespeichers über einen sehr langen Zeitraum wartungsfrei. Um dem Ziel der Energieeinsparung gerecht zu werden, sind Bauteile geringer Leistungsaufnahme verwendet sowie schaltungstechnische Optimierungen durchgeführt worden.

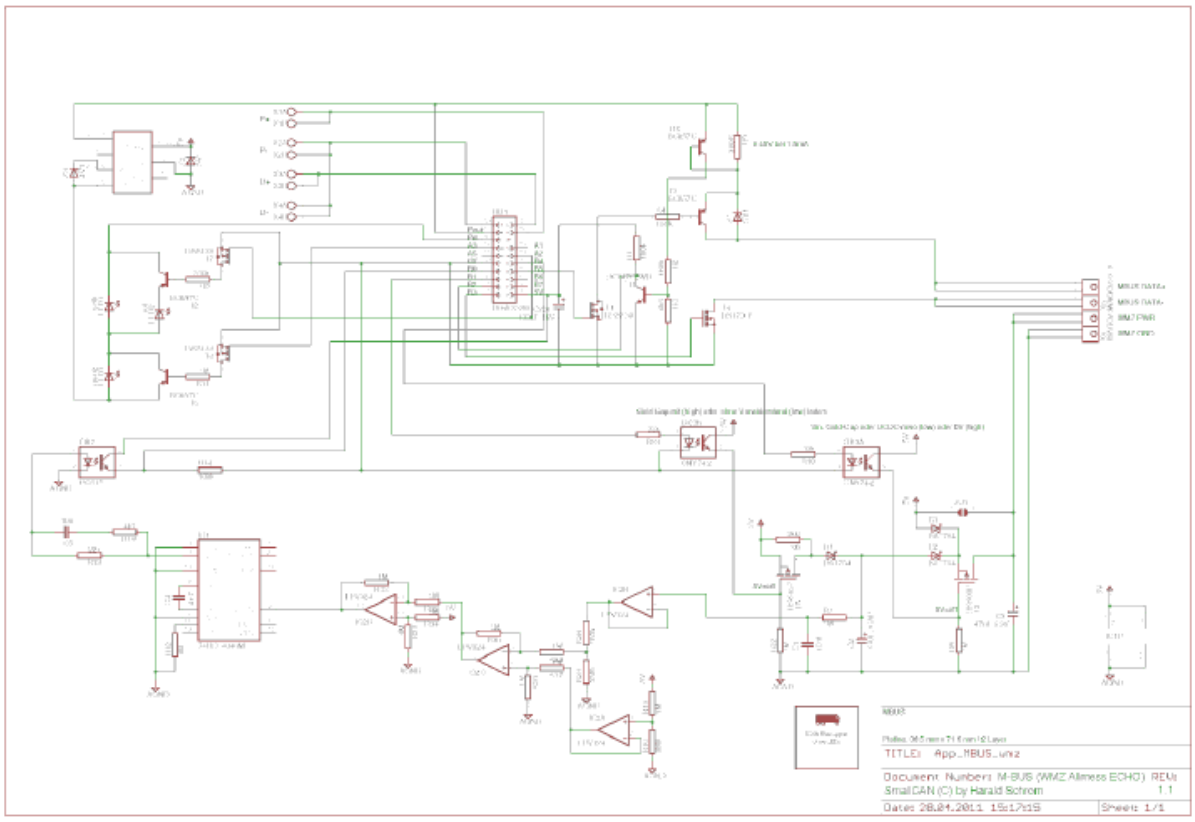


Abbildung 3: Schaltplan für das SmallCAN Anwendungsmodul „App_MBUS_wmz“.

Die Anordnung der einzelnen Bauteilkomponenten auf der Platine (Routing) erfolgte unter Einhaltung der allgemeinen Gesetzmäßigkeiten hinsichtlich Mindestdicken von Leiterbahnen und deren Mindestabstände untereinander zu jeweiligen Spannungsbereichen sowie allgemeingültigen Richtlinien der Platinen-Entwicklung. Da der WMZ selbst nicht potenzialfrei ist, insbesondere für den praktischen Einsatz des Anwendungsmoduls eine galvanische Trennung der Elektroniken seitens SmallCAN und seitens WMZ notwendig, damit keine Überspannungen seitens des WMZ zu Fehlfunktionen oder Zerstörungen von SmallCAN-

Komponenten führt. Ein Kleinserienauftrag eines exemplarischen Anwendungsadapters wurde durch eine externe Bestückungsfirma erfolgreich bearbeitet. Damit ist die Möglichkeit der automatisierten Bestückung in der Serienproduktion prinzipiell gegeben. Abbildung 4 zeigt die Anordnung der Bauteile auf dem Anwendungsmodul App_MBUS_wmz sowie die fertig be-stückte Platine eingebaut in dem Wärmemengenzähler CF-Echo.

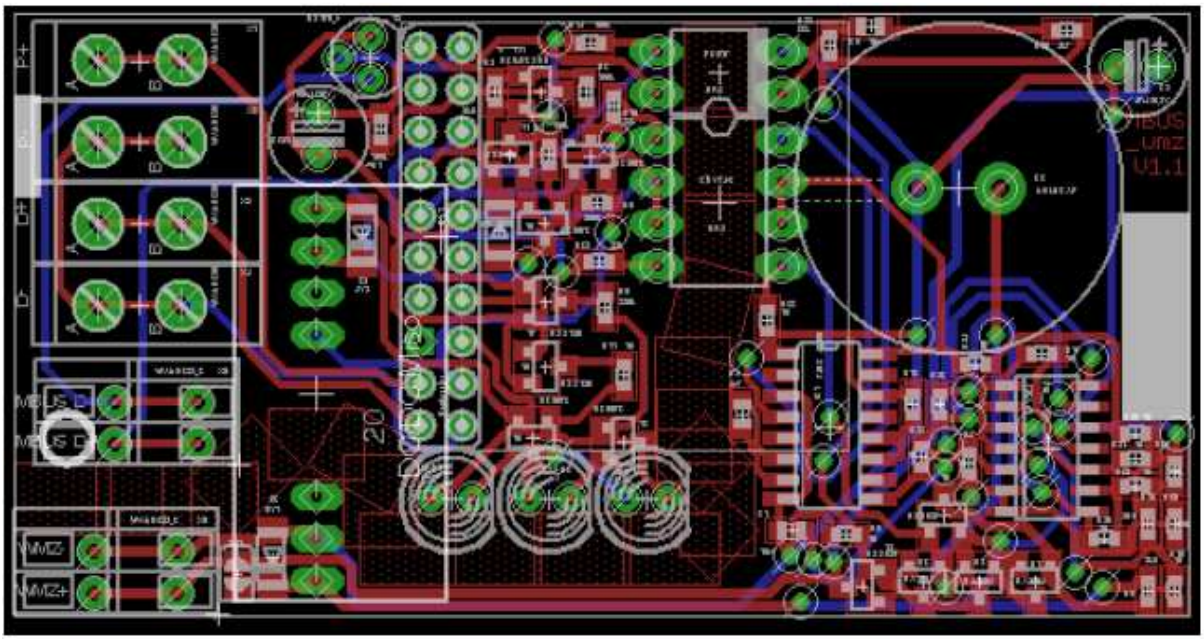


Abbildung 4: PCB „App_MBUS_wmz“.

Insgesamt sind auf der Platine „App_MBUS_wmz“ 68 Bauteile verbaut. Der vollständige Anwendungsadapter mit Buskoppler und aufgesteckten isolierten DC/DC-Wandler besteht aus 128 Bauteilen. Die bestücke und in den Wärmemengenzähler integrierte Platine ist in Abbildung 5 gezeigt. Zur Ansteuerung des Wärmemengenzählers wurde eine Firmware entwickelt, die sich auf den auf dem Buskoppler befindlichen Mikrocontroller-Chip programmieren lässt. Die Firmware fragt in regelmäßigen Abständen die in dem Rechenwerk ermittelten thermischen Größen ab und wandelt diese Daten in SmallCAN konforme Daten um. Für die Entwicklung der Firmware sind weitere Kenntnisse aus verschiedenen Schichten des ISO-OSI-Modells zu ermitteln und umzusetzen. Der zur Ansteuerung dieses M-Bus-fähigen Gerätes entwickelte Quellcode wurde in PIC-Assembler programmiert und beläuft sich auf 1685 Zeilen Code.

Wesentliche Bestandteile der Firmware ist die korrekte Anfrage an die MBUS-Schnittstelle, die Auswertung dessen Antwort, die Dekodierung der Messsignale, die Anpassung an die Datentypen die das SmallCAN-Bussystem bereitstellt sowie eine umfangreiche und notwendige Fehlererkennung.

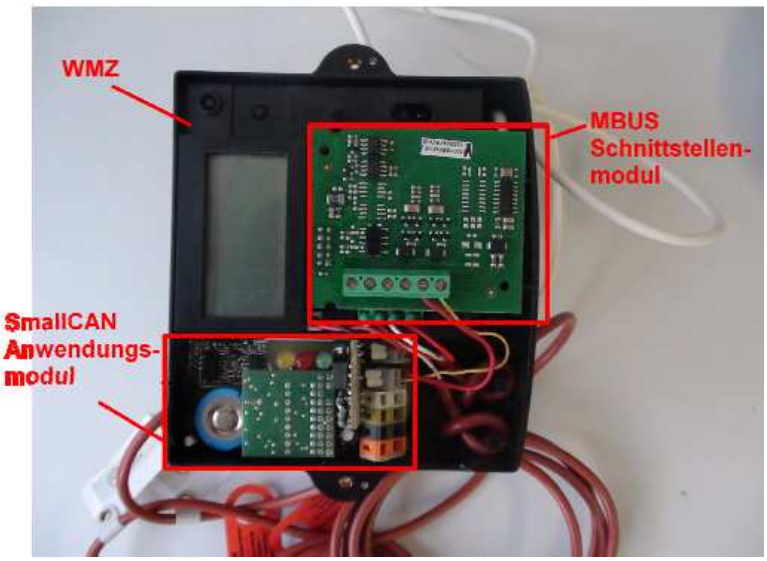


Abbildung 5: Bestückte Platine in WMZ.

Parallel zur Entwicklung eines Anwendungsadapters ist eine Dokumentation durchgeführt worden. Der Inhalt der Dokumentation eines jeden Applikationsadapters gliedert sich wie folgt:

1 Beschreibung

1.1 Grundfunktion

1.2 Hardware

1.3 Leistungsumfang

1.4 Übertragungs- und Geräteparameter

2 Parameter

3 Anzeigen

4 Startverhalten

5 Stopverhalten

6 Busverhalten

6.1 Versand

6.2 Empfang

6.3 Lastbegrenzung

6.4 Interne Werte

7 Rechenzeit

8 Protokolldefinition

9 Ansteuerung der Hardware

10 Ressourcen und Softwareschnittstellen

11 Aufbau der Software

12 Schaltplan

13 Gehäuse

13.1 Frontplattenbeschriftung

14 Bedienung der PC-Software

Zunächst wird die generelle Funktion des Applikationsmoduls beschrieben:

Über MBUS lassen sich Daten des Wärmemengenzählers (WMZ) CF-ECHO II der Firma Allmess (ACTARIS) auslesen und zusätzlich wird der WMZ mit Spannung versorgt.

In einer Kurzbeschreibung wird eine Tabelle ausgefüllt, in der u.a. hinterlegt wird, welche Bibliotheken für die Software genutzt wurden, welche EEPROM -Adressen reserviert wurden, welche Flags verwendet werden und wie die Pins des Buskopplers belegt werden. In der Tabelle sind weiterhin die Sende- und Empfangsnachrichten des Anwendungsmoduls hinterlegt.

Es folgt eine Beschreibung der Hardware:

Ein zum Buskoppler in Reihe geschalteter DCDC_miso versorgt die Sekundärseite mit ca. 5V. Die Sekundärseite wiederum versorgt den WMZ mit Spannung und ladet zugleich einen GoldCap-Kondensator (GC). Sollte es zu einem Spannungsausfall kommen, versorgt der GC den WMZ. Während der GC geladen wird, kann dessen Spannungswert mittels Polling von INT_SF1_0 ausgelesen werden. Des Weiteren kann vom Buskoppler mithilfe von Optokopplern das Laden des GC mit und ohne Vorwiderstandes eingestellt, sowie die WMZ Spannungsversorgung ausgeschaltet werden. Parallel zum Buskoppler und DCDC_miso wird eine MBUS- Daten Sende- und Empfangsperipherie versorgt.

Die Beschreibung des Leistungsumfanges für das Anwendungsmodul ist wie folgt beschrieben:

Der GC bietet eine 2,5 Stündige Spannungsversorgung des WMZ beim Stromausfall. Aktuali- sierung der WMZ- Messwerte in ca. 10 Sekunden. Diese 10s setzen sich zusammen aus 1s Checksumme ermitteln, 5s Messwerte umrechnen, 1s Werte zusammenfassen, 2s senden aus SOND1 und SOND2 und 1s um neue Messwerte anfordern. Bei Übermittlungs- fehlern der MBUS- Daten rotes Blinken bzw. auslösen des HW- Status.

Zur Beschreibung der Übertragungs- und Geräteparameter ist die Abbildung 6 in die Do- kumentation aufgenommen worden. Sie stellt die gültigen Buspegel für den MBUS dar.

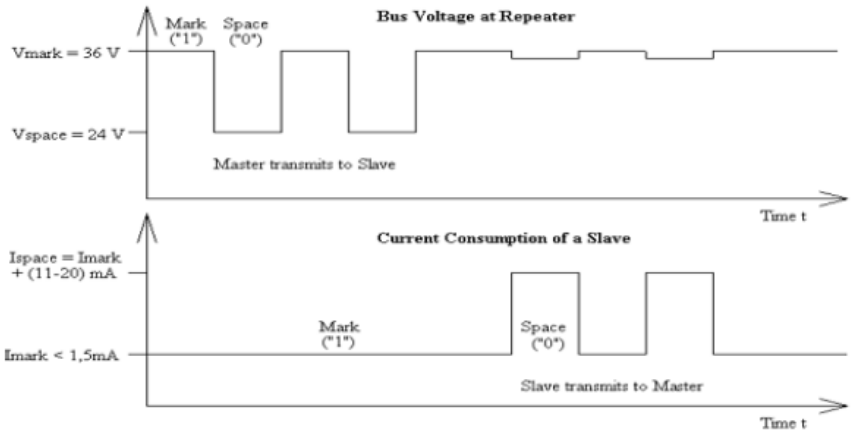


Abbildung 6: Strom und Spannungsverhalten bei der M-BUS-Kommunikation. [Quelle: www.m-bus.de]

Hinsichtlich der Parameter sind folgende Anmerkungen dokumentiert worden:

In der EEPROM- Speicherstelle 128 wird der Spannungsoffset, der bei der ersten Inbetrieb- nahme gemessen wurde, eingetragen.

In der EEPROM- Speicherstelle 129 wird der Spannungsanstiegsfaktor, der bei der ersten Inbetriebnahme angepasst wurde, eingetragen.

Es besteht ein Compiler-Schalter, mithilfe dessen die Überprüfung der Checksumme ein- bzw. ausgeschaltet werden kann.

Bei SF1 wird eine durch 4 teilbare Adresse bzw. eine Adresse, die als Binärzahl mit 11 en- det, (z.B. 91 = 1011011) gewählt. Bei SF2 muss die Adresse nach unten 3 Adressen als Platzhalter frei haben, da Telegramme über 3 Subadressen versendet werden.

Folgende Anzeigen sind für den Anwendungsadapter realisiert worden:

Die LED grün2 blinkt beim Anfragen neuer Daten vom WMZ. Die rote LED blinkt bei Emp- fangsfehlern im MBUS- Empfang.

Das Verhalten beim Starten und Stoppen des Buskopplers ist in der Dokumentation ebenfalls beschrieben:
Beim Start werden sofort die Messwerte des WMZ ausgelesen. Bestehen einige Messwerte nicht, d.h. gibt der WMZ einen Fehlerfall bei diesen aus, so werden diese Messwerte als 0 initialisiert.

Beim Buskopplerstop wird der GC weiter geladen und der WMZ weiter mit Spannung versorgt. Wird der DCDC_miso entfernt, so wird der WMZ noch ca. 2,5 Stunden vom GC Spannungsversorgt. Tabelle 1 zeigt die Messdaten, die das Anwendungsmodul für den Wärmemengenzähler versendet.

Versand ID	Sub ID	Größe	Auflösung	Bus-Datentyp
SF1	0	Energiewert [kWh]	0,001 kWh	ZAHL48
SF2	0	Leistung [W th]	10 W	MESS
SF2	1	Durchfluss [L/s]	1 L/s	MESS
SF2	2	Temp Rücklauf [°C]	0,1 °C	MESS
SF2	3	Temp Vorlauf [°C]	0,1 °C	MESS

Tabelle 1: Nachrichtenversand des Anwendungsmoduls für den WMZ.

Tabelle 2 zeigt interne Messdaten des Wärmemengenzählers, die ggf. abgefragt werden können.

Versand ID	Wertwert	Auflösung	Bitbreite
INT_SF1_0	Spannung am Gold-Cap	0,01 V	12 Bit
INT-SF2_0	Unbenutzt		

Tabelle 2: Interne Messwerte des Anwendungsmoduls für den WMZ.

Das Kapitel 9 der Dokumentation befasst sich mit der genutzten Protokolldefinition. Im Fall des M-BUS Protokolls ist der Datenrahmen für die Anfrage an den WMZ durch das Anwendungsmodul wie folgt definiert:

Start	Steuerfeld	Adressfeld	Prüfsumme	Ende
10h	4Bh	FEh	49h	16h

Der Antwort-Datenrahmen vom WMZ weist folgende Struktur auf:

1	2	3	4	5	6	7-27	28-31			
Start	Länge	Länge	Start	Steuer-feld	Adress-feld	...	**Energiewert**			
68h	4D /5D /54 /64h		68h	08h	NNh	...	I1	I2	I3	I4

32-39	40-42			43-44	45-47			48-49	50-51		52-53
...	**Leistung**			...	**Durchfluss**			...	**Vorlauf T**		...
...	V1	V2	V3	...	V1	V2	V3	...	V1	V2	...

54-55		56-81	82	83
Rücklauf T		...	Prüfsumme	Ende
V1	V2	...	XXh	16h

Dabei steht V1 für die niederwertigsten 2 BCD-Zahlen und die nachfolgenden V2, V3, V4 sind die höherwertigen BCD-Zahlen. Der einzige Wert, der als Integer kodiert ist, ist der Energie- wert. Dabei steht I1 für die niederwertigste Integer-Zahl und die nachfolgenden I2, I3, I4 sind die höherwertigen Integer-Zahlen.

Zur Ansteuerung der Hardware ist folgendes dokumentiert:

Das Senden über MBUS erfolgt mit Spannungspegeln im Bereich von 19-28V. Diese werden mithilfe der Verschaltung von R4, T1, T4, T9 und Z1 erzeugt.

Zum Senden werden am Pin B5 die Transistoren T1 und T9 zum angesteuert. Bei einem High-Signal werden T1 und T9 leitend und T9 überbrückt die Z-Diode Z1. Nun liegt am Ausgang MBUS_DATA+ eine Spannung von 28 V (MBUS High-Signal) an.

Bei einem Low-Signal am Pin B5 sind die Transistoren gesperrt und über die Diode Z1 fallen 9V ab, damit liegt am MBUS_DATA+ eine Spannung von 19 V (MBUS Low-Signal) an.

Das Empfangen erfolgt mit Strompegeln im Bereich von 0-1,5mA. Diese werden mithilfe der Verschaltung von R1, R2, R3, R5, T10 und T11 erzeugt.

Zum Empfangen wird die Spannung am Transistor T11 mit Pin B2 gemessen. Wenn ein Strom > 1,5mA zwischen MBUS_DATA+ und MBUS_DATA- fließt, fallen über R5 0,45V ab

und T10 wird leitend. Daraufhin wird T11 leitend und am Pin B2 liegt ein Low-Signal (MBUS Low-Signal) an. Fließt kein Strom zwischen MBUS_DATA+ und MBUS_DATA-, sperren T10 und T11 und somit liegt am Pin B2 ein High-Signal (MBUS High-Signal) an.

Zusätzlich kann mithilfe von T4 am Pin B3 der MBUS_DATA- hochohmig geschaltet werden. Damit fließt zwischen MBUS_DATA+ und MBUS_DATA- kein Strom, was das Stromsparen ermöglicht.

Auslesen der GC-Spannung:

Der Spannungswert des GC wird mithilfe einer OP Verschaltung aus Werten 0-4,7V in Spannungswerte 1,2 – 4V umwandelt. Diese Spannungswerte werden mithilfe eines VCO IC1 (Volt zu Frequenzwandler) über einen Optokoppler OK2 übertragen und am Pin B0 gemessen.

Das Auslesen erfolgt über Polling von INT_SF1_0. Spannungen im Bereich von 0-4,7V werden in Zahlenwerten von 0 bis 470 dargestellt. Die Abbildung 7 der Spannung auf den Zahlenwert erfolgt nicht genau, es entstehen leichte Abweichungen.

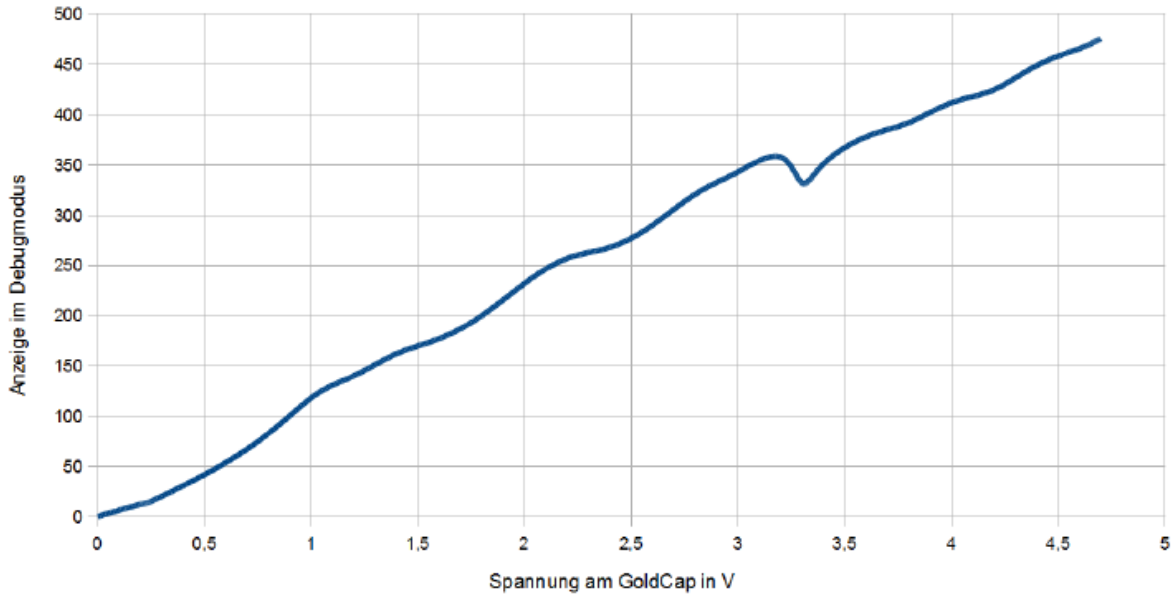


Abbildung 7: Spannungsmessung am Goldcap

Der kleine Einknick entsteht durch die Umschaltung des GoldCaps zwischen laden mit Vorwiderstand und laden ohne Vorwiderstand. Ab einer Spannung von 3,35V wird ohne Vorwiderstand geladen.

Der Sekundärstromkreis wird vom DCDC_miso versorgt. Durch die Z-Diode Z2 stellt sich eine Spannung von ca. 4,8V ein.

Am Pin B1 wird der Optokoppler OK3B angesteuert, der den Transistor T5 leitend bzw. sperrend schaltet. Ist T5 leitend, so wird der GC ohne Vorwiderstand und wenn T5 sperrt, wird der GC mit Vorwiderstand geladen. Am Pin B4 wird der Optokoppler OK3A angesteuert, der den Transistor T3 leitend bzw. sperrend schaltet. Ist T3 leitend, so wird der WMZ mit Spannung versorgt und wenn T3 sperrt, liegt am WMZ keine Spannung an. Die Dioden D2 und D3 sorgen dafür, dass der WMZ entweder über den DCDC_miso oder bei einem Spannungsaus- fall vom GC versorgt wird.

Zur Beschreibung der Firmware soll hier nur auf wesentliche Elemente eingegangen werden. Tabelle 3 gibt zunächst alle für die Firmware definierten Flags an:

FL_RXCOMPL_0	Zeigt ob ein Empfang auf dem MP Bus noch nicht abgeschlossen bzw. evaluiert wurde.
FL_TXCOMPL_SEND1_0	Zeigt ob die Ausgewertenten Daten noch nicht abgesendet wurden von SOND1.
FL_TXCOMPL_SEND2_0	Zeigt ob die Ausgewertenten Daten noch nicht abgesendet wurden von SOND2.
FL_CHECK_PNT_0	Flag zum prüfen des Empfangs-Pointers.
FL_NEW_VALUE_0	Neuer Frequenz-Wert vom VCO eingegangen.
FL_GC_WMZ_0	Auswahl ob GC-Spannung oder WMZ-Werte berechnet und gesendet werden sollten.
FL_CALCCOMPL_0	Zeigt ob Berechnung der empfangenen Werte abgeschlossen ist.
FL_SENDCOMPL_0	Zeigt ob neu Ermittelten Werte raus gesendet wurden.
FL_NEW_POWER_1	Flag, dass neue Leistung gemessen wurde.
FL_NEW_FLOW_1	Flag, dass neuer Durchfluss gemessen wurde.
FL_NEW_TEMPRUECK_1	Flag, dass neue Temperatur im Rücklauf gemessen wurde.
FL_NEW_TEMPVOR_1	Flag, dass neue Temperatur im Vorlauf gemessen wurde.
FL_MEAS_1	Flag zum starten der Messung der Pulse.
FL_MEAS_OLD_1	Alter Wert von PIN B0
PIN_IN_OLD_1	Alter Wert von PIN B0.
FL_NEW_ENERGY_1	Flag, dass neuer Energiewert gemessen wurde.
FL_SEND2_SF20	Zeigt an, ob Sub-ID 0 gesendet wurde.
FL_SEND2_SF21	Zeigt an, ob Sub-ID 1 gesendet wurde.
FL_SEND2_SF22	Zeigt an, ob Sub-ID 2 gesendet wurde.
FL_SEND2_SF23	Zeigt an, ob Sub-ID 3 gesendet wurde.
FL_CHECK0_2	Zeigt an, ob Checksumme berechnet wurde.
FL_CALC1_2	Zeigt an, ob der Wert für Temperatur-Rücklauf berechnet wurde.
FL_CALC2_2	Zeigt an, ob der Wert für Temperatur-Vorlauf berechnet wurde.
FL_CALC3_2	Zeigt an, ob der Wert für Energiewert berechnet wurde.
FL_CALC4_2	Zeigt an, ob der Wert für Durchfluss berechnet wurde.
FL_CALC5_2	Zeigt an, ob der Wert für Leistung berechnet wurde.

Tabelle 3: verwendete Flags für die WMZ-Firmware. 19

Exemplarisch zeigt Abbildung 8 das Flussdiagramm zur Berechnung der Daten aus dem Antwort-Datenrahmen des WMZ nachdem die Checksumme erfolgreich geprüft wurde.

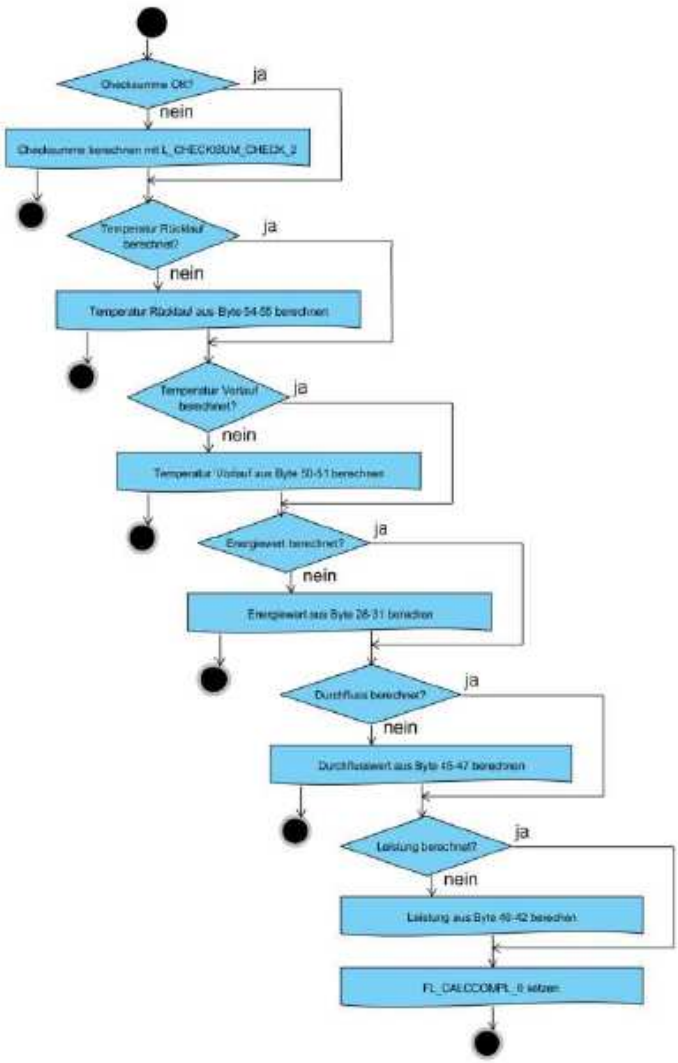


Abbildung 8: Flussdiagramm zur Berechnung der WMZ-Werte.

Entsprechende Dokumentationen wurden für alle Anwendungsmodule angefertigt.

2.1.2 Firmwareentwicklung

Bezugnehmend auf die Zielsetzung des Projektes der Energieersparung sowie der Bereitstellung neuer Funktionalitäten in der Gebäudeautomation sind intelligente Regelungsfunktionen (verarbeitende Funktionen) erstellt worden, um die verschiedenen Anwendungsmodule sinnvoll interagieren zu lassen. Die verarbeitenden Funktionen werden in der Programmiersprache C geschrieben und als zusätzliches Modul zu der entwickelten Firmware auf den auf den Mikrocontroller geschrieben. Die Anwendung der verarbeitenden Funktionen ist auch im Demonstrationsmodell „future workspace“ im vollen Umfang erfolgt und konnte die funktionellen Vorgaben auch in der Praxis erfüllen. Hierzu siehe Anhang A1.

2.2 Ausgangslage Busserver/GUI

Die Anbindung des SmallCAN-Busses an die PC-Welt geschieht über die serielle Schnittstelle wie in Abbildung 9 dargestellt.

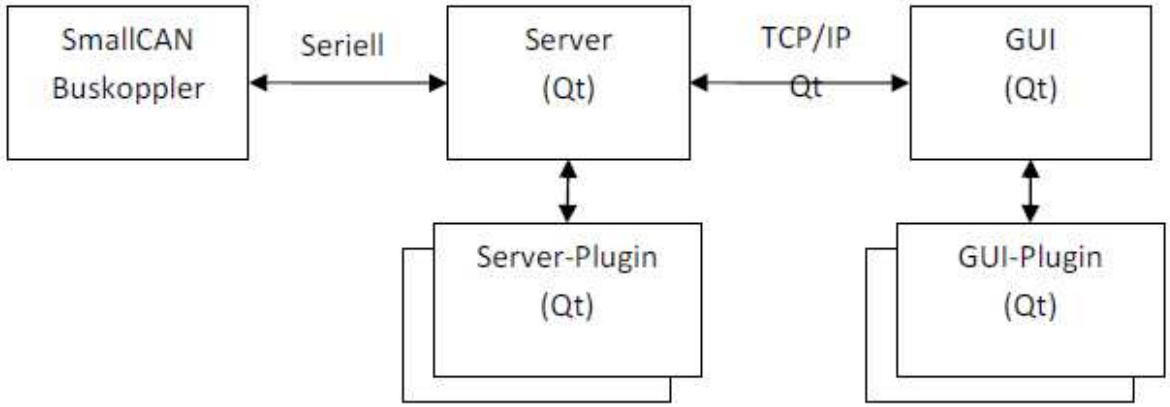


Abbildung 9: Ausgangsarchitektur der SmallCAN-Anbindung.

In der Übertragungskette erfüllen die einzelnen Blöcke folgende Rollen:

- SmallCAN-Buskoppler sendet Bustelegramme per RS232 zum PC und vom PC auf dem Bus. Um den Sicherheits-Integritätslevel drei (SIL3) zu erreichen, werden alle Telegramme sowohl vom Buskoppler zum PC als auch in der Gegenrichtung einzeln mit einer Checksumme quittiert. Der Buskoppler hält eine Warteschlange von 16 Telegrammen bereit, weil der Serverprozess von einem nicht echtzeitfähigen Betriebssystem (Windows, Linux, Mac iOS) zu variablen aufgerufen wird.
- Der Server verwaltet Systemabbild (den Zustand aller Buskoppler auf dem SmallCAN) und den Benutzerabbild (aktueller Wert aller Busvariablen). Außerdem leitet der Server alle Bustelegramme von der seriellen Schnittstelle per TCP zur GUI und zurück.
- GUI stellt mehrere Tabellen dar: aktuelle Telegramme auf dem Bus, Systemabbild, Benutzerabbild, Busstatistik, graphische Darstellung des Benutzerabbildes mit der Möglichkeit, die Busobjekte zu verändern. Weiterhin bietet GUI eine Plugin-Schnittstelle, um mit Hilfe von Plugins die einzelnen Funktionen auf den Buskopplern zu parametrieren.

Ein Überblick über die Komplexität des Codes gibt sein Umfang in Zeilen:

- 1000 Zeilen PIC-Assembler im anbindenden SmallCAN-Buskoppler

- 7300 Zeilen C++ gemeinsame Bibliothek von Server und GUI
- 5300 Zeilen C++ im Server
- 25000 Zeilen C++ im GUI
- 35000 Zeilen C++ in den Plugins zur Konfiguration der Funktionen

Im Laufe des vorliegenden Projektes wurde eine neue Implementierung entwickelt, die folgende Kennzahlen aufweist:

- 1200 Zeilen PIC-Assembler im anbindenden SmallCAN-Buskoppler
- 23000 Zeilen C++ Server
- 3600 Zeilen XML-Beschreibung der Funktionen
- 7500 Zeilen C++ GUI

Um die Notwendigkeit einer Neustrukturierung der Softwarearchitektur abzuleiten werden im Weiteren Vor- und Nachteile der aktuellen Architektur analysiert.

2.2.1 Vor- und Nachteile der aktuellen Architektur

Zu den Vorteilen zählen:

- Einfache Architektur
- Hohe Erweiterbarkeit der GUI durch Plugins

Die Nachteile:

- Das Protokoll Buskoppler <-> PC mit Quittierung jedes Telegramms führt unter Windows bereits bei geringer Last zum Abriss des Datenstroms, weil der FIFO im Buskoppler überläuft.
- Das Protokoll Server <-> GUI basiert auf Datenstrukturen von C++-Bibliothek Qt. Es lassen sich nur Qt-Clients anbinden.
- Der GUI-Entwickler muss jede Feinheit des SmallCAN-Systems verstehen, weil der Server nur zur Datenweiterleitung und „-protokollierung“ verwendet wird.
- Der SmallCAN-Installateur muss sich mit allen Bus-Datentypen, Speicheraufbau der Buskoppler und den einzelnen Funktionen auskennen, um diese zu installieren und miteinander zu verknüpfen.
- Die Bus-Adressen agieren als IDs der einzelnen Busobjekte (Busvariablen). Es gibt ei- ne feste Zuordnung dieser Adressen zu den GUI-Elementen. Die Busadressen können aber verändert werden, um das Laufzeitverhalten der einzelnen Komponen-

ten zu verbessern. In diesem Fall müssen die Veränderungen in mehreren GUI-Arten nachgezogen werden.

- Bei einem industriellen Einsatz gibt es z. B. den BACnet-Standard, nachdem die einzelnen Komponenten in ein Gesamtnetz integriert werden. Z. B. die TU Braunschweig verwaltet über 80 Gebäude mit verschiedenen Installationen und Herstellern mit Hilfe nur einer graphischen Benutzeroberfläche, die alle Komponenten per BACnet einbindet. Der BACnet-Standard verlangt feste, unveränderbare IDs für die einzelnen Objekte. Diese können nicht auf veränderbaren SmallCAN-Telegramm-IDs basieren.
- Der kommerzielle Erfolg des SmallCAN-Busses bei Privatanwendern hängt vor allem von der graphischen Benutzeroberfläche ab. Hier sind mehrere Möglichkeiten denkbar sowohl bei der Hardware (Touchscreen, Smartphone, PC, Pad), Darstellungsarten (2D, Animationen, 3D, detailliert/einfach), Plattform (Internetbasiert im Browser, fest installierter Client). Der Aufwand für die Entwicklung einer neuen graphischen Benutzeroberfläche ist wegen des Verhältnisses zehnmal aufwendiger als der gemeinsame Teil (Server).
- Der Hardware/Software-Entwickler, der einzelne Busapplikationen entwickelt, muss auch für einfache Parameter ein Plugin mit Qt in C++ programmieren. Sollten mehrere verschiedene graphischen Benutzeroberflächen benutzt werden, muss auch der Entwickler für diese neue Plugins programmieren.
- Änderung der Plugin-Schnittstelle erfordert eine Anpassung im Großteil des Codes in den Plugins, die von verschiedenen Entwicklern stammen.
- Begrenzte Testbarkeit des Codes wegen der engen GUI-Kopplung.
- Es gibt keine Kontrolle der Konfiguration, der Installateur/Benutzer ist verantwortlich für

    - kollisionsfreie Adressenvergabe
    - Einhaltung der Datentypen auf dem Bus
    - Zuordnung der Telegramme an die richtige Funktionen und richtige Eingänge

  Um diese Aufgaben wahrzunehmen, muss der Installateur den zukünftigen Busverkehr abschätzen und jede eingesetzte Funktion kennen (jeweils 30 Seiten Dokumentation).

Ziel der vorhandenen Software war die Überprüfung der Machbarkeit mit minimalem Aufwand. Dies spiegelt sich auch darin wider, dass es keine Testfälle existieren: sobald eine Teilfunktion einmal läuft, wird die nächste implementiert. Eine Codeänderung ist unter diesen Umständen mit hohen Risiken verbunden.

2.2.2 Anforderungen an die neue Server-Client-Architektur

Die Anforderungen haben sich im Laufe des Projektes herauskristallisiert (z.B. FutureWorkspace). Der Hauptunterschied zur vorhandenen Architektur liegt in der Konzentration der Funktionalität im Server und Verwendung einer schlanken graphischen Benutzeroberfläche. Damit werden die Vor- und Nachteile der bisherigen Software vertauscht: die Serverarchitektur wird komplex, dafür ist die Entwicklung neuer Busfunktionen und Oberflächen einfach. Im Weiteren werden die Rahmenbedingungen für den Busserver analysiert.

2.2.2.1 Server-Hardware

Eines der wichtigsten Merkmale des SmallCAN-Bussystems ist sein geringer Stromverbrauch. Um dieses Ziel auch mit dem Busserver zu erfüllen, wurde ein sparsamer MiniPC basierend auf ARM9-Board von Hectronic (www.hectronic.se) entwickelt, mit einem Stromverbrauch von 1.5 Watt. Dieser läuft mit 180 MHz ARM9-Prozessor unter Linux und beinhaltet 32 MB RAM. Eine andere Zielhardware stellt ein Windows-PC dar, wobei diese meist um mehrere Größenordnungen leistungsfähiger ist, als das ARM9-Board. Eine mögliche Zielhardware könnte auch z. B. ein Fritzbox-Modem sein, der bei einer DSL-Anbindung sowieso immer aktiv ist.

Hauptbegrenzungsfaktor ist in diesem Zusammenhang der Speicher (RAM). Auf dem MiniPC ließe sich auch eine Webserver-basierte graphische Benutzeroberfläche starten (z. B. http://www.webtoolkit.eu/wt), so dass der SmallCAN von einem beliebigen Ort ohne Instal- lationsaufwand überwacht und gesteuert werden kann. Der Webserver würde ca. 20MB RAM beanspruchen. Für den Busserver würde 2-4MB für die Verwaltung von 1000 Buskopp- lern und bis zu 30.000 Busobjekte übrig bleiben.

2.2.2.2 Betriebssystem

Der Server soll sowohl unter Linux, als auch unter Windows lauffähig sein. Diese Betriebssysteme unterscheiden sich vor allem in der Behandlung der seriellen Schnittstelle. In der bisherigen Implementierung musste der Server unter Windows mit Administratorrechten mit höchster Priorität gestartet werden, um die serielle Schnittstelle größtenteils rechtzeitig zu bedienen.

Am Projektbeginn wurde die Möglichkeit untersucht, einen speziellen Treiber für Windows zu programmieren, der das Anbindungsprotokoll zum SmallCAN-Buskoppler umsetzt. Nach außen (zum Busserver) würden dann nur Telegramme und keine Bytes kommuniziert.

Parallel dazu wurde der Anbindungsprotokoll auf eine Sammelbestätigung umgestellt: statt jedes einzelne Telegramm zu bestätigen, werden bis zu 15 Telegramme auf einmal bestätigt. Damit bekommt das Betriebssystem des Servers bis zu 90 ms Zeit, um die Kontrolle an den Busserver zu übergeben. Ab Windows XP wird jeder Task alle 15 ms aufgerufen. Somit hat man auch unter hoher Belastung (z. B. Abspielen eines Videos) genug Reserven.

2.2.2.3 Anbindung der Clients (z. B. graphische Oberfläche)

Es gibt mehrere Möglichkeiten, die Kommunikation mit den Clients durchzuführen. Man muss folgende Entscheidungen treffen:

- Netzprotokoll: UDP vs. TCP
- Telegrammen: binär vs. textbasiert (z.B. XML)

Als optimal scheint in diesem Zusammenhang ein binäres Protokoll über TCP zu sein. Ein

textbasiertes Protokoll erfordert ca. 10mal größere Bandbreite, die würde

- die Anzahl der parallel laufenden Clients am MiniPC-Server einschränken,
- knappen Ressourcen des Busservers (RAM und CPU) verbrauchen,
- Verbindungskosten und Übertragungszeiten zu mobilen Clients (Smartphones) erhöhen

Es gibt mehrere Alternativen für die Kommunikation:

- Proprietäre Telegramme. Der Nachteil besteht in einem hohen Aufwand und der großen Anzahl an nötigen Implementierungen
- CORBA: generisches Standard für ein TCP basiertes binäres Protokoll, das Program- miersprachen und Betriebssystem unabhängig ist. Der Nachteil: der Hipe von CORBA liegt 10 Jahre zurück. Es liegen nur für C/C++ und Java gute Implementierungen vor. Anbindung an Smartphones wäre aufwendig (Android, iOS).
- ZeroC-Ice (http://www.zeroc.com/), ein Opensource-Nachfolger von CORBA. Bietet Implementierungen für die meisten wichtigen Programmiersprachen und Betriebssysteme.
- Apache Thrift (keine Callbacks, nur Polling möglich)

- Google-Buffers (Nur Datentransport, keine RPC, nur java, c++, python)
- BACnet speziell für Gebäudeautomatisierung entwickelt, ein UDP-basiertes, binäres Protokoll. Nachteile:

    o großer Aufwand auf der Client-Seite

    o nurrudimentärevorhandeneOpensource-Implementierungen

    o deckt nur den BACnet-Teil der Kommunikation ab.

  Vorteil: existierende (teure) industrielle Clients.

Es wurde entschieden, dass der Busserver eine ZeroC-Ice Schnittstelle bereitstellt. Daran können graphische Clients, remote-Prozesssteuerung und auch ein ZeroC-Ice <-> BACnet-Umsetzer angebunden werden. Dies vereinfacht die Implementierung des Busservers und verlagert das Management von BACnet-Verbindungen in eine spezielle Softwareeinheit.

Daraus ergeben sich folgende Aufgabenstellungen:

2.2.2.4 Serielle Busanbindung

Der erste Baustein der neuen Architektur war die serielle Kommunikation mit dem Buskoppler. Dies erforderte die Einarbeitung und Neuimplementierung der Anbindungsfunktion in Assembler auf dem Buskoppler. Hauptschwierigkeit dabei ist fehlende Möglichkeit zu debuggen. Die Assemblerfunktion muss schrittweise aufgebaut und getestet werden. Sobald eine Änderung des Algorithmus nötig ist, wird die Vorgehensweise wiederholt. Auf der Serverseite wurden 1710 Zeilen Code für die Funktionalität und 600 Zeilen für die Tests verwendet. Das relativ umfangreiche Kommunikationsprotokoll wurde in drei Klassen aufgeteilt (vgl. Abbildung 10).

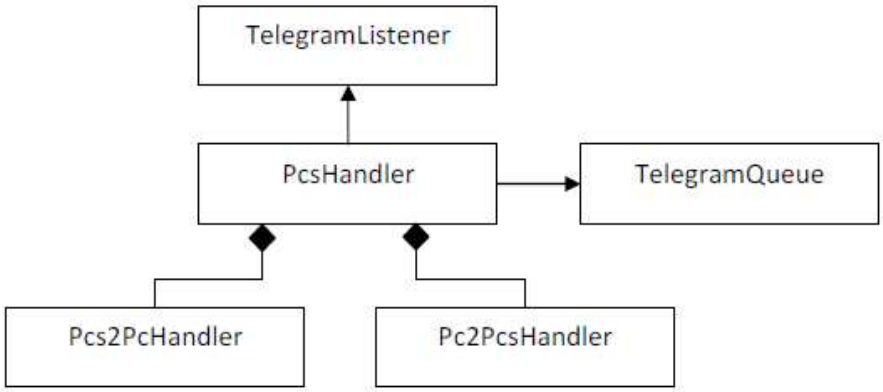


Abbildung 10: Anbindung der seriellen Schnittstelle in die Serverinfrastruktur

Sowohl die einzelnen Klassen als auch die Funktion wurden ausgiebig getestet (die Testabdeckung von ca. 90%). Dabei wurden sowohl manuelle offline Tests durchgeführt, als auch online Test. Bei den Online-Tests wurde zwischen dem Port-Treiber eine „Störungsroutine“ eingeschaltet, die jedes zwanzigstes Bit zufällig umkippte. Die Störungen wurden sowohl von Buskoppler als auch von PC simuliert. Dadurch konnte eine große Vielfalt an möglichen Fehlern offenbart werden, die weniger aus dem Protokoll, sondern viel mehr durch Timing-Problem der Implementierung entstanden sind. Z. B. muss die Implementierung „Bestätigungstelegramme“ von den „Bustelegrammen“ trennen. Falls ein „falsches“ Bit gekippt wird, muss die Implementierung raten, ob jetzt zwei kaputte „Bestätigungstelegramme“ angekommen sind und ein richtiges „Bustelegramm“, oder ein kaputtes „Bustelegramm“ und zwei normale „Bestätigungstelegramme“. Je nach Entscheidung läuft das Protokoll verschiedene Ausführungspfade. Nach fast zwei Monaten Tests wurde auch diese sehr hohe Fehlerrate korrekt abgearbeitet. In nur sehr seltenen Fällen, bei denen innerhalb von sechs Bytes zwei spezielle Bitfehler auftreten, wurde die Übertragung trotz der Checksumme unbemerkt verfälscht.

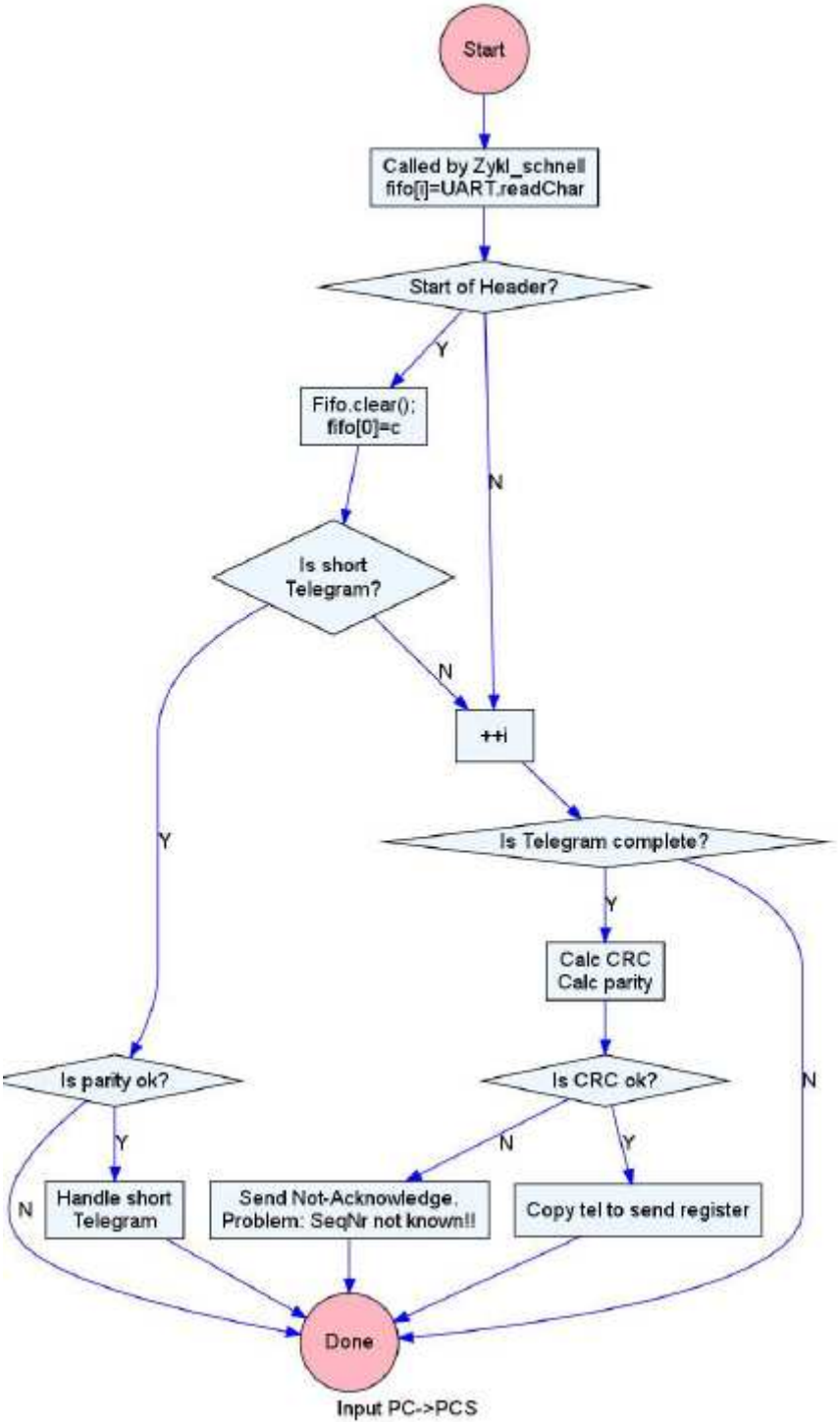


Abbildung 11: Assembler-Routine, die Telegramme von PC verarbeitet und per SmallCAN versendet

2.2.2.5 Systemdialoge

Bei der Abbildung des Zustandes von SmallCAN muss zwischen dem Systemabbild (Zustand der Hardware) und dem Benutzerabbild (Zustand der Variablen, z.B. Ventilstellungen, Messwerte etc.) unterschieden werden. Während die meisten Variablen durch ein Bustele-

gramm übertragen werden, wird beim Aufbau des Systemabbilds über zustandsbehaftete Protokolle kommuniziert.

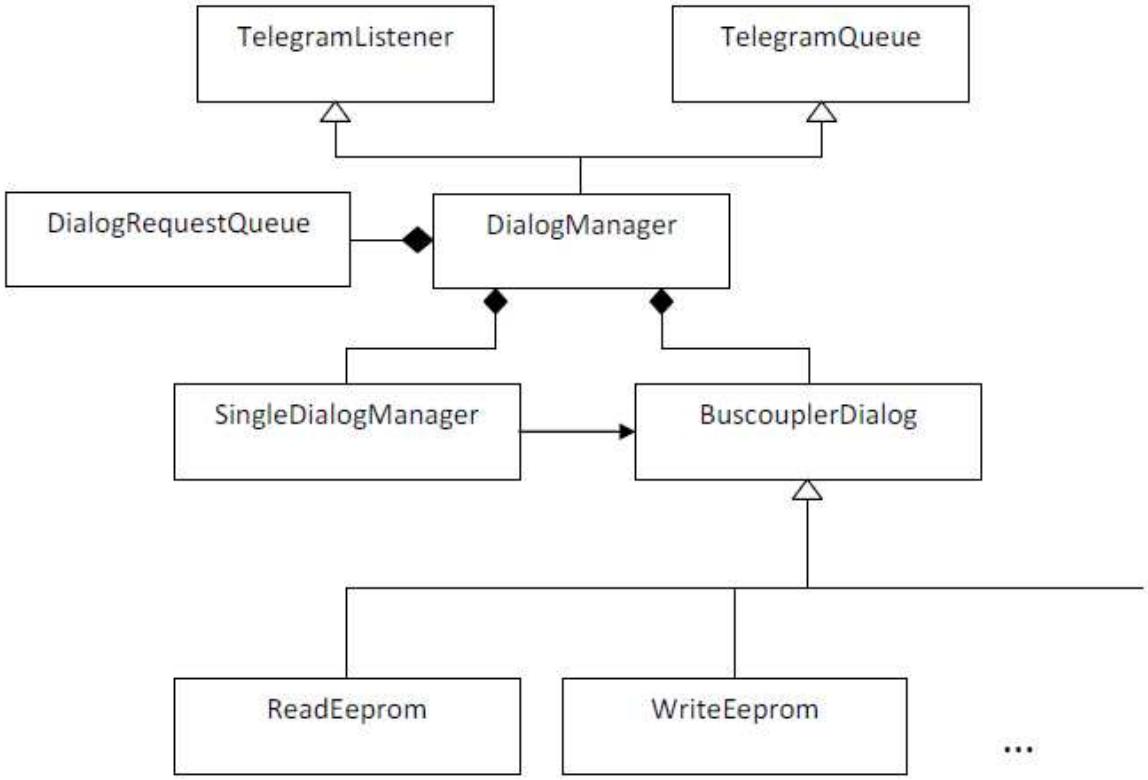


Abbildung 12: SystemDialoge-Bibliothek

Die einzelnen Klassen erfüllen bei der Kommunikation folgende Rollen:

- Die Telegrammzustellung zu den einzelnen Dialogen, Starten der neuen Dialoge verwaltet der DialogManager. Dieser beinhaltet auch einen Timer, um plötzliche Hardware-Fehler während der Dialoge aufzudecken.
- Verwaltung der Zustände, die in allen Dialogklassen identisch sind, geschieht in SingleDialogManager.
- Die meisten Buscouplerdialoge sind zustandslose Klassen, die ihren Zustand in SingleDialogManager verwalten. Ausnahmen bilden Dialoge zum Lesen des Eeproms, die nur einmal auf dem Bus stattfinden dürfen.
- Alle DialogKlassen haben noch einen „passiven“ Modus, bei dem sie den Dialogen anderer zuhören und den Zustand der Buskoppler daraus ableiten.

Der vorliegende Aufbau hat zwei wesentliche Vorteile, gegenüber vorherigen „einfacheren“ Struktur:

1. Sie verbrauchen ca. 10 mal weniger Speicher (RAM)

2. Der Speicher wird nur einmal allokiert und dann statisch verwaltet. Dies verringert die Wahrscheinlichkeit, dass nach 10-20 Jahren ununterbrochenem Betrieb der Hauptspeicher zu stark fragmentiert ist.

Sobald ein Dialog abgeschlossen ist, verändert er den Zustand einer Buskoppler-Klasse, worüber alle anderen „Buskoppler-Zuhörer“ informiert werden.

2.2.2.6 System**modell**

Bisher wurden die Funktionen auf den Buskopplern als Blackboxes betrachtet, deren Eigenschaften (Eeprom-Parameter, Sende und Empfangstelegramme) per Plugins erreichbar sind. Dies ist ein sehr flexibler Ansatz, erfordert aber den hohen Programmieraufwand für die Plugin-Entwicklung. Es stellt sich auch heraus, dass in den meisten Fällen, die Parameter und Objekte der Funktionen nach ähnlichen Verfahren auf Eeprom der Buskoppler bzw. Bustelegramme umrechenbar sind. In den seltenen Fällen, wo es komplexe, nicht allgemein gültige Algorithmen nötig sind, um z.B. aus mehreren Bustelegrammen eine komplexe Textmeldung zu berechnen, ist der Einsatz von Plugins weiterhin unumgänglich.

Bisher wurden über 50 verschiedene Funktionen für die Buskoppler entwickelt. Ein Objektmodell war nötig, um

- Ein- und Ausgänge der Funktionen in einer graphischen Benutzeroberfläche zu visualisieren,
- feste globale IDs für alle Systemobjekte konsequent zu vergeben,
- automatisch Ausgänge mit Eingängen unter Einhaltung der Datentypen und Bitposition zu verknüpfen.

Eine gut lesbare Möglichkeit, das Interface einer Funktion zu beschreiben stellt die XML-Sprache dar. Nach mehreren Anläufen wurde folgende Grundstruktur in XML-Form festgesetzt (vgl. Abbildung 13).

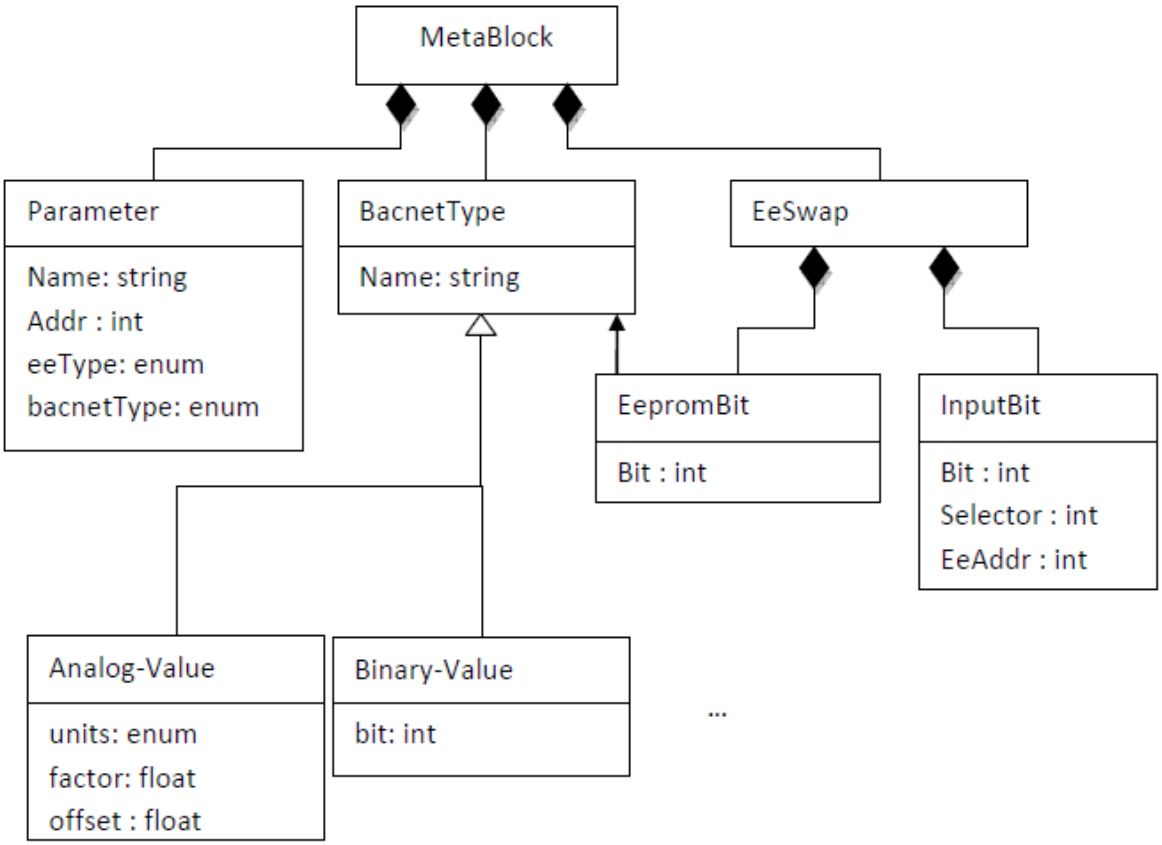


Abbildung 13: Klassendiagram der XML-Beschreibung der Buskoppler-Funktionen

Ein Beispiel für die XML-Spezifikation einer freien Sonderfunktion:

<MetaBlock ID="AUSLG" type="FSF" version="0">
<parameter name="HeizVentilModus" addr="255" eeType="uint8" bacnetType="multi- state-value"/>
<parameter name="KuehlVentilModus" addr="254" eeType="uint8" bacnet-Type="multi-state-value"/>
<parameter name="LowScaling" addr="253" eeType="uint8" bacnetType="analog-value" />
<parameter name="UpperScaling" addr="252" eeType="uint8" bacnetType="analog-value"/>
<parameter name="Skalierungsfaktor" addr="251" eeType="uint8" bacnetType="analog-value"/>
<analog-value name="HeizAnforderung" units="percent">
<input selector="6"><ScanRatio/></input>
</analog-value>
<analog-value name="Kuehlanforderung" units="percent">
<input selector="14"><ScanRatio/></input> </analog-value>
<analog-value name="Zuluft_Temperatur" units="degrees-Celsius">
<input selector="22"><ScanInt14 factor="0.1"/></input>
</analog-value>

```
        <binary-value name="Pumpe" bit="0">
        <output basisAddr="FSF1" offset="0"><ScanBinState/></output>
    </binary-value>
    <binary-value name="Lufter" bit="0">
        <output basisAddr="FSF1" offset="1"><ScanBinState/></output>
    </binary-value>
    <analog-value name="Lufter_Abluft" units="percent">
    <output basisAddr="FSF1" offset="2"><ScanRatio/></output>
    </analog-value>
    <analog-value name="Lufter_Zuluft" units="percent">
        <output basisAddr="FSF1" offset="3"><ScanRatio/></output>
    </analog-value>
    <multi-state-value name="HeizVentilModus" bitSize="2">
        <StateText>modus0</StateText>
        <StateText>modus1</StateText>
        <StateText>modus2</StateText>
        <StateText>modus3</StateText>
        <output basisAddr="FSF1" offset="4"><ScanSelection/></output>
    </multi-state-value>
    <multi-state-value name="KuehlVentilModus" bitSize="2">
        <StateText>modus0</StateText>
        <StateText>modus1</StateText>
        <StateText>modus2</StateText>
        <StateText>modus3</StateText>
        <output basisAddr="FSF1" offset="5"><ScanSelection/></output>
    </multi-state-value>
    <analog-value name="HeizVentilStellung" units="percent">
        <output basisAddr="FSF1" offset="6"><ScanRatio/></output>
    </analog-value>
    <analog-value name="KuehlVentilStellung" units="percent">
        <output basisAddr="FSF1" offset="7"><ScanRatio/></output>
    </analog-value>
</MetaBlock>
```

Die relativ aufwendige Struktur resultiert aus dem Wunsch nach möglichst großer Freiheit für den Assembler-Programmierer und noch beherrschbarer Komplexität für den Busserver. Die Rahmenbedingungen ergeben sich aus bereits existierenden Buskopplerfunktionen:

- Dem Client wird die Businfrastruktur in Form von BACnet-Objekten präsentiert. Dieser soll nicht den Busverkehr verstehen.
- Ein BACnet-Object kann auf unterschiedliche Arten auf SmallCAN-Datentypen abgebildet werden, wobei die Abbildung oft über Parameter im Eeprom im Betrieb veränderbar ist. Deswegen ist eine Zusatzschicht in Form von BacnetType nötig, die die BACnet-Objekte auf die einzelnen SmallCAN-Inputs und –Outputs abbilden.

- Es werden weitere SmallCAN-spezifische Abbildungsarten über eigene XML- Elemente spezifiziert. Kompliziert wird es noch wegen der Möglichkeit, mehrere BACnet-Objekte in verschiedenen Bustelegrammen zu kombinieren. Auf der SmallCAN-Ebene ermöglicht dies ein flexibles Prioritätenmanagement, für den Busserver wird die Komplexität der Abbildung erhöht.
- Es gibt recht generische Abbildungen, wie z. B. die Umrechnung eines 14Bit-Int-Messwertes in ein Fließkomma-Wert mit Hilfe von zwei Parametern: y = a*x +b. Andererseits gibt es sehr spezifische Abbildungen, wie z.B. die Abbildung eines SmallCAN-Telegramms auf ein Textstring, der in Eeprom recht komplex gespeichert wird. Für solche Abbildungen werden weiterhin Plugins benötigt.

Neben frei programmierbaren Funktionen, die in XML spezifiziert werden, wird ein Teil der Funktionalität von den Standardfunktionen des Betriebssystems bereitgestellt. Deren Parameter sind zum Teil in Eeproms platziert und können zur Laufzeit geändert werden. Zusammen mit den XML-basierten Beschreibungen ergibt sich folgende Architektur für das Systemmodel (Abbildung 14).

Die Unterklassen von FunctionModel erfüllen drei wichtige Funktionen:

- Sie grenzen die ID-Bereiche bei der ersten Initialisierung des Buskopplers ein.
- Sie führen Mapping von SmallCAN auf Ice-Datenstruktur und zurück.
- Sie liefern Informationen, die für das Verbindungsmanagement benötigt werden.

Insgesamt beinhaltet die Bibliothek SystemModell 28 Klassen mit 7500 Zeilen Code. Hinzu kommen 750 Zeilen für die Tests, die fast 60% des Codes abdecken.

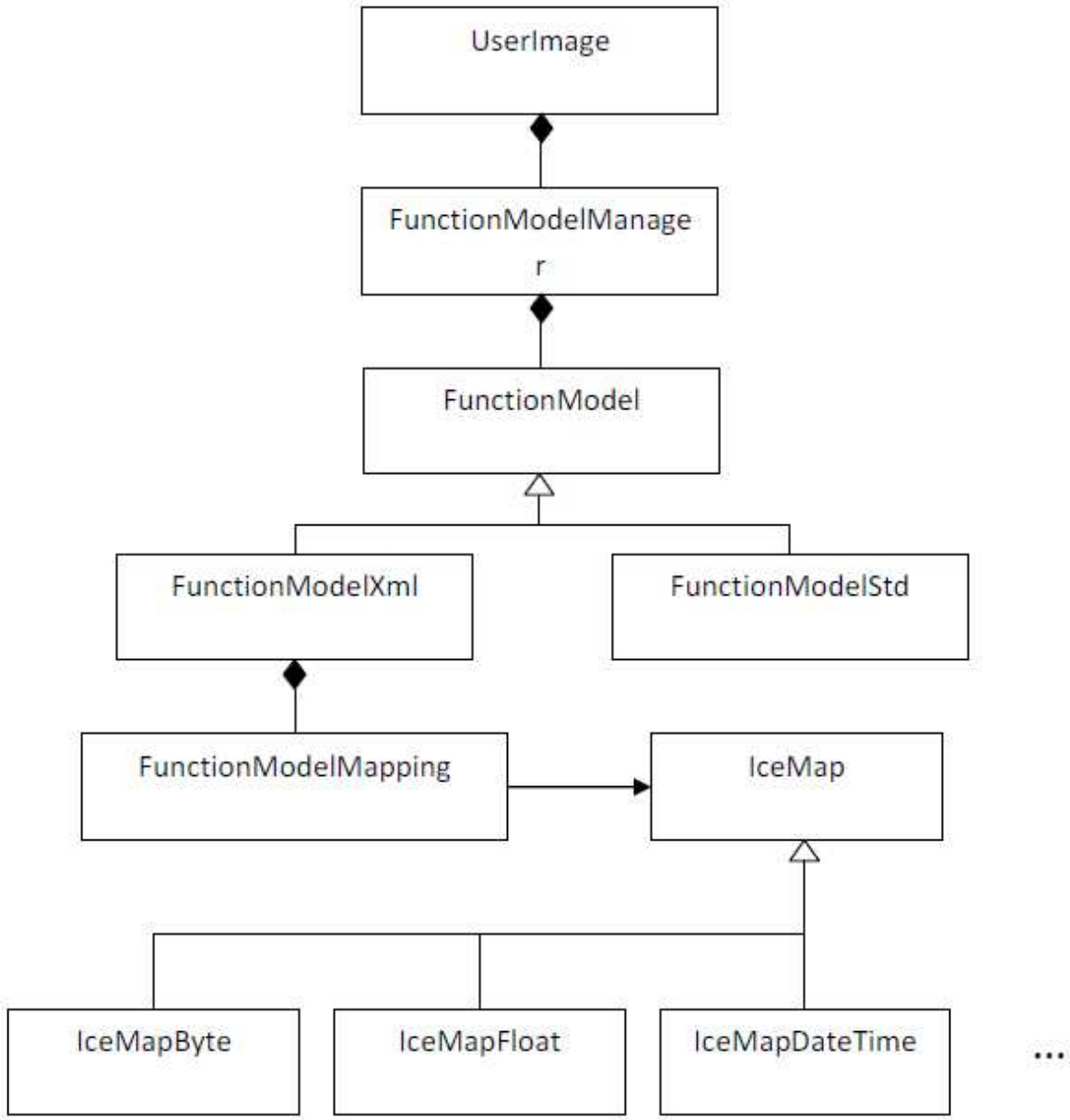


Abbildung 14: Grundstruktur des Systemmodells.

2.2.2.7 ZeroC-Ice Schnittstelle

Die Anbindung an den Busserver geschieht über die Corba-ähnliche Ice-Schnittstelle. Im Ge- gensatz zu UDP-basiertem Protokoll von BACnet, wo die Datentypen in den Telegram- men mitkodiert sind, erfordert Ice eine vorherige Festlegung auf die zu übertragenen Datenty- pen. Dies einerseits vereinfacht eine typsichere Übertragung und Umwandlung in Zielspra- chen (C++, Java, PHP, Python, .Net, etc) verkompliziert aber das Interface. Aus Performance- gründen sollte die Übertragung in möglichst großen Paketen möglichst selten erfolgen, d.h. es werden in einem Aufruf Variablen verschiedener Datentypen übertragen. Um dies zu ermöglichen wurde eine Grundstruktur definiert, die alle möglichen Datentypen in einzelne Sequenzen ablegt.

```
struct PropertyStruct {
        IntPropertyList intValues;
        FloatPropertyList floatValues;
        DoublePropertyList doubleValues;
        StringPropertyList stringValues;
        BoolPropertyList boolValues;
        ObjectRefPropertyList objValues;
        StringListPropertyList stringListValues;
        DateTimePropertyList dateTimeValues;
         BlobPropertyList blobValues;
};
```

Das Übertragungsprotokoll von Ice ist so konzipiert, dass für eine leere Sequenz nur ein Byte verwendet wird, so dass auch bei einer leeren Datenstruktur maximal neun Bytes übertragen werden. Auf der Client-Seite müssen dann die einzelnen Sequenzen iteriert werden, um die neuen Werte typsicher zu visualisieren.

Nach BACnet-Standard verfügen die einzelnen Objekte über mehrere Parameter, wobei der aktuelle Wert (z.B. die gemessene Temperatur) nur einer davon ist. Der BACnet-Standard spezifiziert alle möglichen Parameter für alle BACnet-Objekte und ordnet diesen eine Ganzzahl-ID zu.

Ein Beispiel für einen Int-Parameter:

```
        struct IntProperty {
                PropertyIdentifier id;
                int value;
        };
```

wobei der PropertyIdentifier sowohl das BACnet-Objekt als auch den Parameter spezifiziert:

```
struct PropertyIdentifier {
        int objectId;
        short propertyId;
};
```

Diese Arte der Definition wird benutzt, um mit der einzigen „PropertyStruct“ alle möglichen Parameter aller Busobjekte einschließlich der Buskoppler zu lesen und zu schreiben. Dies ermöglicht ein sehr schlankes Interface für den Busserver:

```
interface ServerObject {
void readProperties(PropertyList propList, out PropertyStruct propValues);
void writeProperties(PropertyStruct propValues);
void registerPropertyListener(PropertyList propList, Ice::Identity ident, out PropertyStruct propValues);
};
```

Der Client bekommt die Möglichkeit, die Werteänderungen sowohl per Polling als auch per Callback zu bekommen. Auf diese Weise können sowohl die Abrufintervalle als auch der Umfang der übertragenen Informationen ja nach Verbindungstyp (UMTS, DSL, Ethernet) und je nach dargestellter Information vom Client bestimmt werden.

Für die Verwaltung der aktuellen Werte der Busvariablen wir die gleiche Struktur verwendet. Dies vereinfacht den Versand von Ice-Nachrichten, weil keine dynamische Speicherallokation benötigt wird.

2.2.2.8 ID-Vergabe

Basis für die Kommunikation mit den Clients stellen unveränderliche IDs der einzelnen Busobjekte dar. Da das Bussystem einer Änderung unterworfen ist: neue Buskoppler kommen hinzu, Funktionen auf den Buskopplern werden durch neue Versionen ausgetauscht, bzw. durch völlig neue Funktionen ersetzt. Deswegen ist die ID-Vergabe eine nicht triviale Aufgabe.

Im BACnet-Standard werden 32-bittige IDs aus dem ObjectTyp und ObjectNummer gebildet. Bei dem Busserver werden die IDs anders vergeben. Es werden ID-Bereiche definiert, die für bestimmte Aufgaben genutzt werden:

1 – Hauptkontainer für alle anderen Buselemente, DEVICE nach BACnet-Definition.

2-50 reserviert

50 – 1050 – Buskoppler

ab 1100 werden die IDs linear vergeben nach dem Muster:

Funktion

Parameter 1 .. N

Sicherheitsabstand für die spätere Versionierung

Busobjekte

Sicherheitsabstand für die spätere Versionierung

Danach folgt die nächste Funktion. Auf diese Weise lassen sich die Parameter und die Busobjekte schnell den Funktionen zuordnen (die Suche läuft logarithmisch), die ihrerseits die Umwandlung von Ice-Werten in Bustelegramme mit Hilfe der IceMap-Klassen durchführen.

2.2.2.9 User-Abbild

Aufgabe des User-Abbilds ist die Bereitstellung der aktuellen Werte der Busobjekte und die Weiterleitung deren Änderungen an die Clients. Dabei dient die Klasse Telegram2Value der Umwandlung der Bustelegramme in Union-basierte Werte (Abbildung 15). Dies ist nötig, weil manche Werte über mehrere Telegramme verteilt sind, z.B. DateTime oder 48bit- Integer über 4 Telegramme.

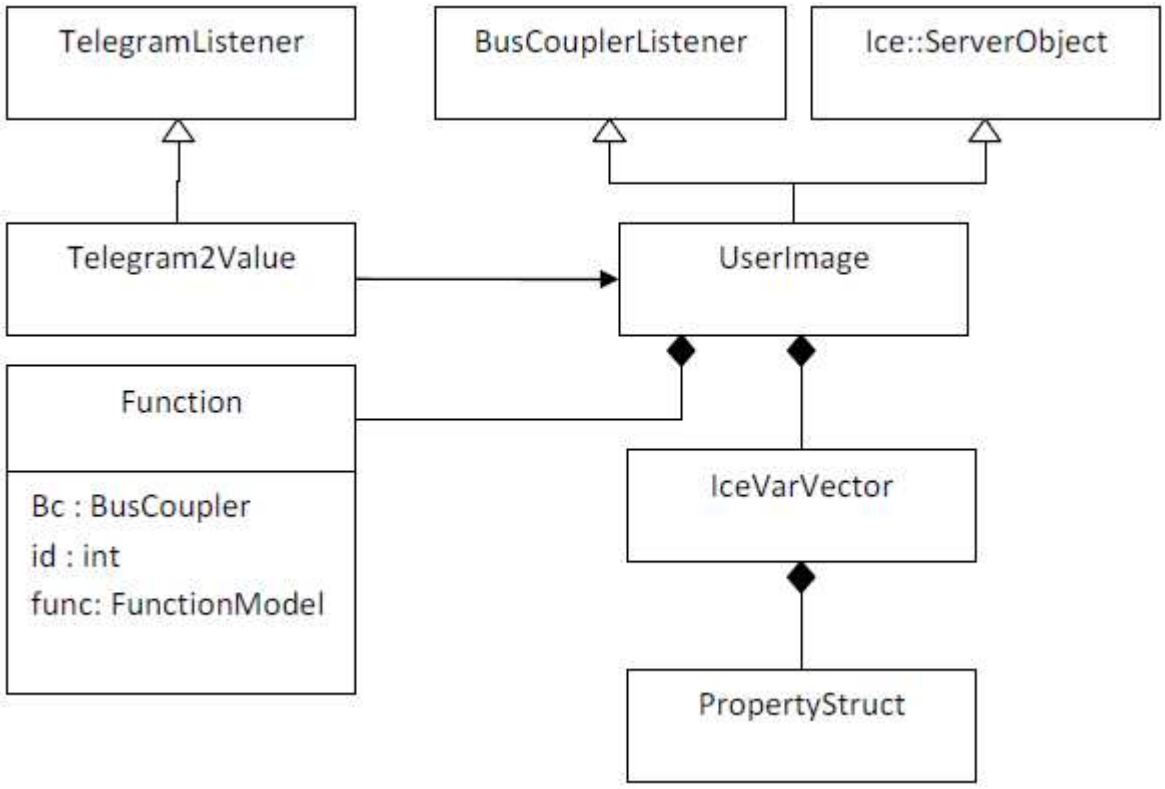


Abbildung 15: Einbindung von UserImage in die Serverarchitektur

IceVarVector bietet ein Interface zu PropertyStruct an, über den sowohl die aktuellen Werte gesetzt als auch die geänderten Werte bis zum Versand aufbewahrt werden. Die Klasse Funktion speichert die BACnet-ID der Funktion, basierend auf der alle Unterobjekte der Funktion über die FunctionModel erreichbar sind.

2.2.2.10 Verbindungsmanagement

Das Verbindungsmanagement erfüllt mehrere Aufgaben:

- Für eine Quellverbindung findet es mögliche Zielports im Sinne der BACnet-IDs. Für eine gewünschte Verbindung
    - o Findet es passende Versandtelegramme
    - o FindetespassendePortsderZielfunktion
    - o Führt es falls möglich eine Adaptation der Bitpositionen durch, z.B. ein Schalter auf dem Bit2 wird mit einem Relais auf dem Bit 4 verbunden.
    - o beauftragt es ein WriteEeprom-Dialog mit der Änderung der Empfangsliste des Zielbuskopplers.

- Für das Löschen einer vorhandenen Verbindung
    - findet espassenden Eintrag in der Empfangsliste des Zielbuskopplers
    - führt es einen „Dialog“ mit dem Client, ob alle anderen Verbindungen des Eintrags auch gelöscht werden sollen.

Hauptschwierigkeit beim Verbindungsmanagement ist, dass bis zu acht binäre Werte in einem Telegramm übertragen werden. Nicht in allen Fällen lassen sich also einzelne binäre Werte voneinander unabhängig verbinden.

2.2.2.11 Visualisierung

Ziel der Neustrukturierung des Busservers war die Vereinfachung einer neuen Visualisierung. Dies ist unter anderem dadurch erreicht, dass die Clients von der Busbehandlung abstrahiert werden und nur mit Standard-Datentypen agieren. Weitere Vereinfachungen betreffen unveränderbare IDs der Buselemente, die mit den graphischen Visualisierungselementen somit fest verknüpft werden können.

Aktuell werden zwei Visualisierungsarten weiter entwickelt:

- Ein editierbarer Baum, indem alle Buskoppler, alle Funktionen, alle Parameter und alle Busvariablen vorhanden sind. Die aktuellen Werte werden dargestellt und veränderbare Werte lassen sich editieren, per Bus übertragen und falls erfolgreich auch in dem Baum darstellen (vgl. Abbildung 16).
- Eine QML-basierte rein graphische Darstellung, die sowohl für mausbedienbare PCs als auch auf mobilen Plattformen mit Touchscreen eingesetzt wird (vgl. Abbildung 17).

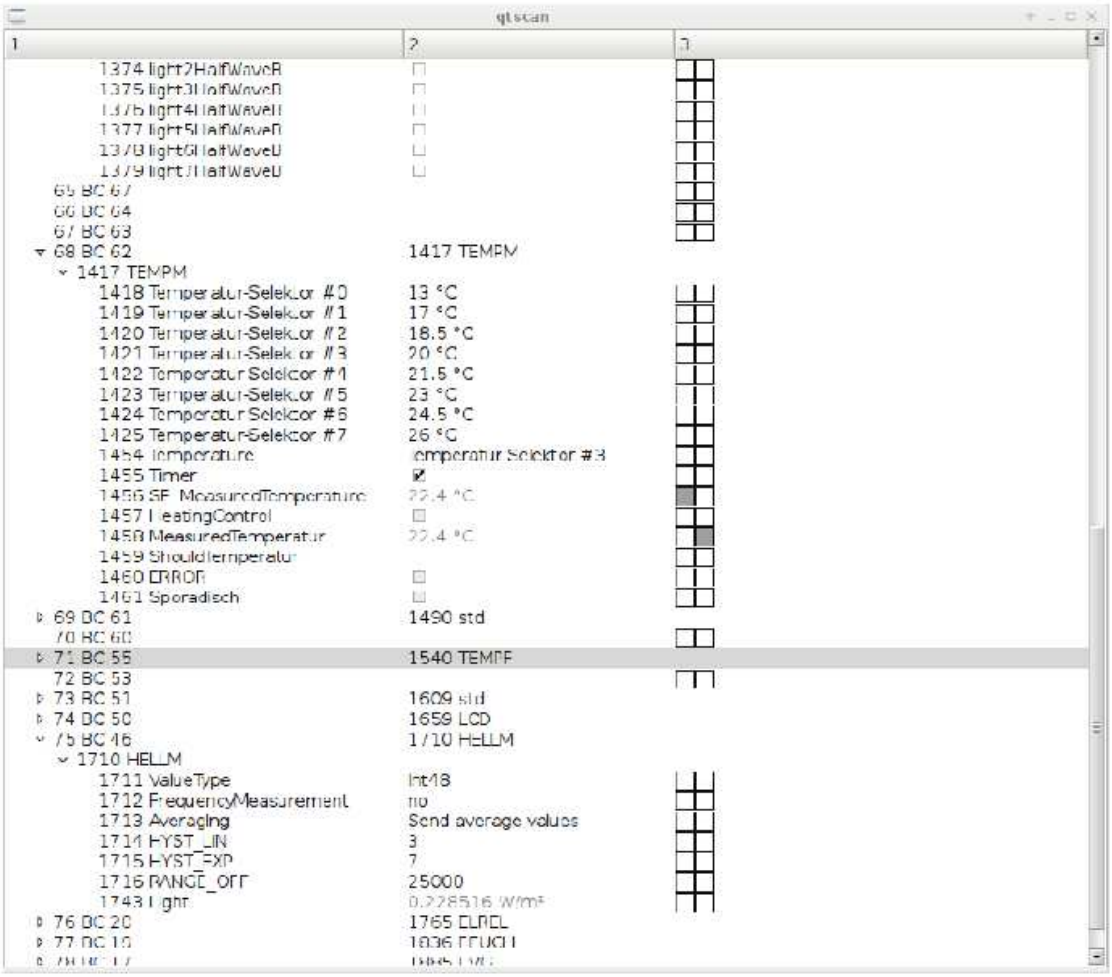


Abbildung 16: Baumansicht auf den SmallCAN-Bus

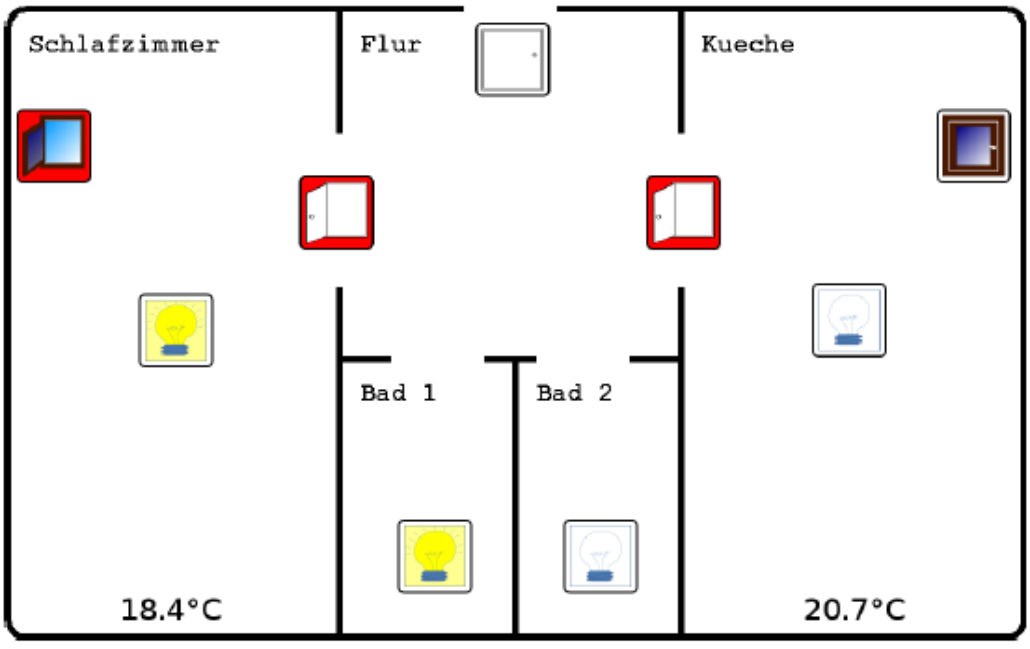


Abbildung 17: Vereinfachte Benutzeroberfläche 40

2.3 CE-Kennzeichnung

Um den SmallCAN in den Verkehr bringen zu können, ist es erforderlich die Notwendigkeit einer CE-Kennzeichnung des Produktes zu überprüfen. Dazu wird in den nachfolgenden Abschnitten das Prozedere zur Analyse der CE-Kennzeichnungspflicht und das Prozedere zur Erfüllung der Anforderungen zur CE-Kennzeichnung beschrieben.

2.3.1 Wie bekommt man eine CE-Kennzeichnung?

Für das Erlangen einer CE-Kennzeichnung sind nachfolgende Schritte notwendig:

1. Relevanzprüfung der CE-Richtlinien
2. Relevanzprüfung der harmonisierten Normen
3. Durchführung einer Gefährdungsanalyse und Risikobeurteilung
4. Erstellung von technischen Unterlagen
5. Aufsetzen einer Konformitätserklärung
6. Anbringen der CE-Kennzeichnung am Produkt

2.3.2 Ausgangspunkt Richtlinien

Die Grundlage einer jeden potenziellen CE-Kennzeichnung ist eine strukturierte Analyse der bestehenden CE-Richtlinien und der in diese Richtlinien referenzierten harmonisierten Normen. Aktuell existieren in Europa 28 verschiedene CE-Richtlinien [1], welche sich jeweils auf verschiedene Produktarten beziehen. Um das Produkt SmallCAN-Buskoppler konform dem geltenden europäischen Recht zu kennzeichnen, müssen zunächst die für das Produkt relevanten Richtlinien identifiziert werden.

2.3.3 Analyse der harmonisierten Normen

Harmonisierte Normen sind Normen, die eine oder mehrere Richtlinien zur Einhaltung der Anwendungen zur CE-Kennzeichnung konkretisieren. Harmonisierte Normen sind europäische Normen, die durch Comité Européen de Normalisation Electrotechnique (CENELEC), Europäische Komitee für Normung (CEN) und European Telecommunications Standards Institute (ETSI) im Auftrag der Europäischen Kommission und der EFTA erarbeitet werden.

Durch eine Einteilung der harmonisierten Normen in die folgenden Kategorien (A, B, C), soll eine Wiederholung identischer Norminhalte in Normen verhindert werden.

- A-Normen sind Sicherheitsgrundnormen, die Begriffe, Gestaltungsleitsätze und allgemeine Aspekte der CE-Richtlinien klären.
- B-Normen sind Sicherheitsgruppennormen, die bestimmte Produktgruppen gleichermaßen betreffen.
- C-Normen sind Sicherheitsproduktnormen, die sich auf einzelne Produkte und ihre Sicherheitsanforderungen beziehen.

Für die CE-Konformitätserklärung müssen alle relevanten Normen, die auf das jeweilige Produkt zutreffen und für die CE-Kennzeichnung relevant sind, dokumentiert werden, um deren Einhaltung bei einer späteren Überprüfung nachweisen zu können. Des Weiteren muss die in der Norm geforderte Risikoanalyse durchgeführt und ebenfalls, für einen späteren Nachweis, dokumentiert werden. Im Folgenden wird der Prozess einer Risikoanalyse beschrieben.

2.3.4 Grundlagen zur Risikoanalyse

Für eine CE-Kennzeichnung ist die Durchführung einer Risikoanalyse erforderlich die im Folgenden beschrieben. Des Weiteren wird aufgezeigt, wie eine Risikoanalyse durchgeführt wird.

Eine Risikoanalyse ist eine Analyse zur Beurteilung von Risiken. Sie dient zur frühzeitigen Erkennung von Gefährdungen (Gefährdungsidentifikation), und zur Bewertung der mit ihr verbundenen Risikopotenziale. Das Risikopotenzial wird durch die Kombination der Eintrittswahrscheinlichkeit des Gefährdungsereignisses und des potenziell resultierenden Schadensausmaßes charakterisiert.

Zur Durchführung einer Risikoanalyse, werden potenzielle Fehlermöglichkeiten des Produktes ermittelt und eingestuft. Durch das Ermitteln der Fehler werden Gefährdungen sichtbar. Zur Berechnung des Risikos werden folgende Parameter multipliziert:

1. Häufigkeit / Wahrscheinlichkeit, dass der Fehler auftritt
2. Das Ausmaß des Fehlers

Zur Abschätzung dieser Parameter schlägt die Sicherheitsgrundnorm IEC 61508-5 folgende Kategorisierung vor.

Risikoparameter		Klassifizierung	Erläuterungen
Auswirkung (*C*)	C_1	Geringe Verletzung	1 Das Klassifizierungssystem ist entwickelt worden, um Verletzungen und Tod von Personen zu berücksichtigen. Für Umwelt- und Materialschäden müssten andere Klassifizierungsverfahren entwickelt werden. 2 Bei der Interpretation von C_1, C_2, C_3 und C_4 müssen die Auswirkungen des Unfalls und normale Heilungsprozesse betrachtet werden.
	C_2	Schwere irreversible Verletzung einer oder mehrerer Personen; Tod einer Person	
	C_3	Tod mehrerer Personen	
	C_4	Tod sehr vieler Personen	
Häufigkeit und Aufenthaltsdauer im gefährlichen Bereich (*F*)	F_1	Seltener bis öfterer Aufenthalt im gefährlichen Bereich	3 Siehe Anmerkung 1 oben.
	F_2	Häufiger bis dauernder Aufenthalt im gefährlichen Bereich	
Möglichkeit, den gefährlichen Vorfall zu vermeiden (*P*)	P_1	Möglich unter bestimmten Bedingungen	4 Dieser Parameter zieht in Betracht: – Betrieb eines Prozesses (überwacht (d. h. betrieben durch ausgebildete oder nicht ausgebildete Personen) oder nicht überwacht); – Geschwindigkeit der Entwicklung des gefährlichen Vorfalls (z. B. plötzlich, schnell, langsam); – Leichtigkeit der Erkennung der Gefahr (z. B. unmittelbar erkennbar, durch technische Maßnahmen aufgedeckt, ohne technische Maßnahmen aufgedeckt); – Vermeidung des gefährlichen Vorfalls (z. B. Fluchtwege möglich, nicht möglich oder unter bestimmten Bedingungen möglich); – aktuelle Sicherheitserfahrung (diese Erfahrung kann von identischen oder ähnlichen EUC oder ähnlichen EUC herrühren, oder kann nicht vorhanden sein).
	P_2	Beinahe unmöglich	
Wahrscheinlichkeit des unerwünschten Ereignisses (*W*)	W_1	Eine sehr geringe Wahrscheinlichkeit, dass die unerwünschten Ereignisse auftreten, und nur wenige unerwünschte Ereignisse sind wahrscheinlich.	5 Der Faktor "*W*" dient zur Bestimmung der Häufigkeit des unerwünschten Ereignisses, ohne die Berücksichtigung jeglicher sicherheitsbezogener Systeme (E/E/PE oder andere Technologie), aber unter Berücksichtigung der externen Einrichtungen zur Risikominderung. 6 Wenn wenig oder gar keine Erfahrungen mit der EUC oder einem ähnlichen EUC oder EUC-Leit- oder Steuerungssystem bestehen, kann die Bestimmung des Faktors "*W*" durch Berechnung erfolgen. In solchen Fällen muss eine "worst-case"-Vorhersage gemacht werden.
	W_2	Eine geringe Wahrscheinlichkeit, dass die unerwünschten Ereignisse auftreten, und wenige unerwünschte Ereignisse sind wahrscheinlich.	
	W_3	Eine relativ hohe Wahrscheinlichkeit, dass die unerwünschten Ereignisse auftreten, und häufige unerwünschte Ereignisse sind wahrscheinlich.	

Abbildung 18: Kategorisierung der Risikoparameter [nach: IEC 61508-5]

Wurden die Parameter mittels der dargestellten Tabelle abgeschätzt, wird an Hand des Risikographen (siehe Abbildung 19) eine Risikobewertung durchgeführt.

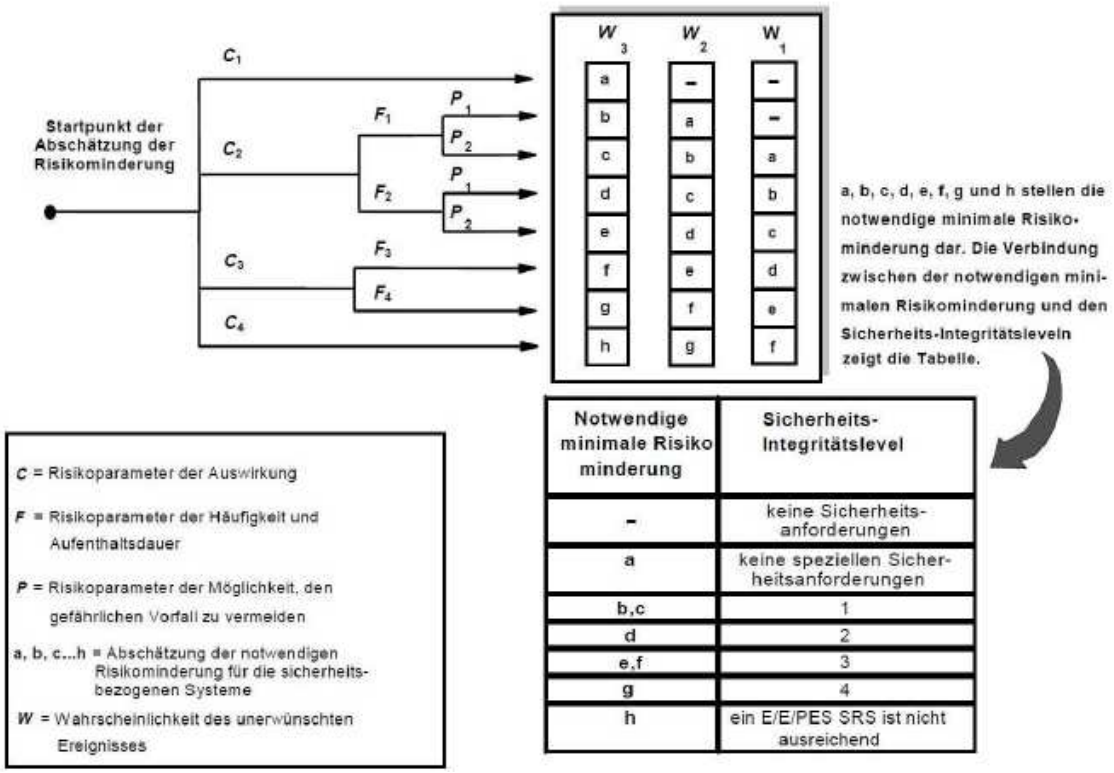


Abbildung 19: Risikograph [nach: IEC 61508-5]

2.3.5 Konformitätsnachweis für des SmallCAN-Buskopplers

Im Folgenden Abschnitt wird die Durchführung der einzelnen Aktivitäten zur CE- Kennzeichnung des SmallCAN-Buskopplers beschrieben. Des Weiteren wird eine Risikoanalyse durchgeführt. Darüber hinaus wird aufgeführt, welche Dokumente zum Erlangen eines Konformitätsnachweises relevant sind.

Das nachfolgende Flussdiagramm (Abbildung 20) visualisiert den Weg zum Erreichen einer Konformitätserklärung für den SmallCAN-Buskoppler.

Um für den SmallCAN-Buskoppler eine Konformitätserklärung zu erlangen, werden die 28 CE-Richtlinien augenscheinlich auf Relevanz den SmallCAN-Buskoppler überprüft, so dass im Folgenden nur die nachstehend gelisteten produktbezogene Richtlinien betrachtet werden.

1. 89/106/EWG Bauprodukte
2. 2004/108/EG Elektromagnetische Verträglichkeit (EMV)
3. 2006/95/EG Niederspannungsrichtlinie

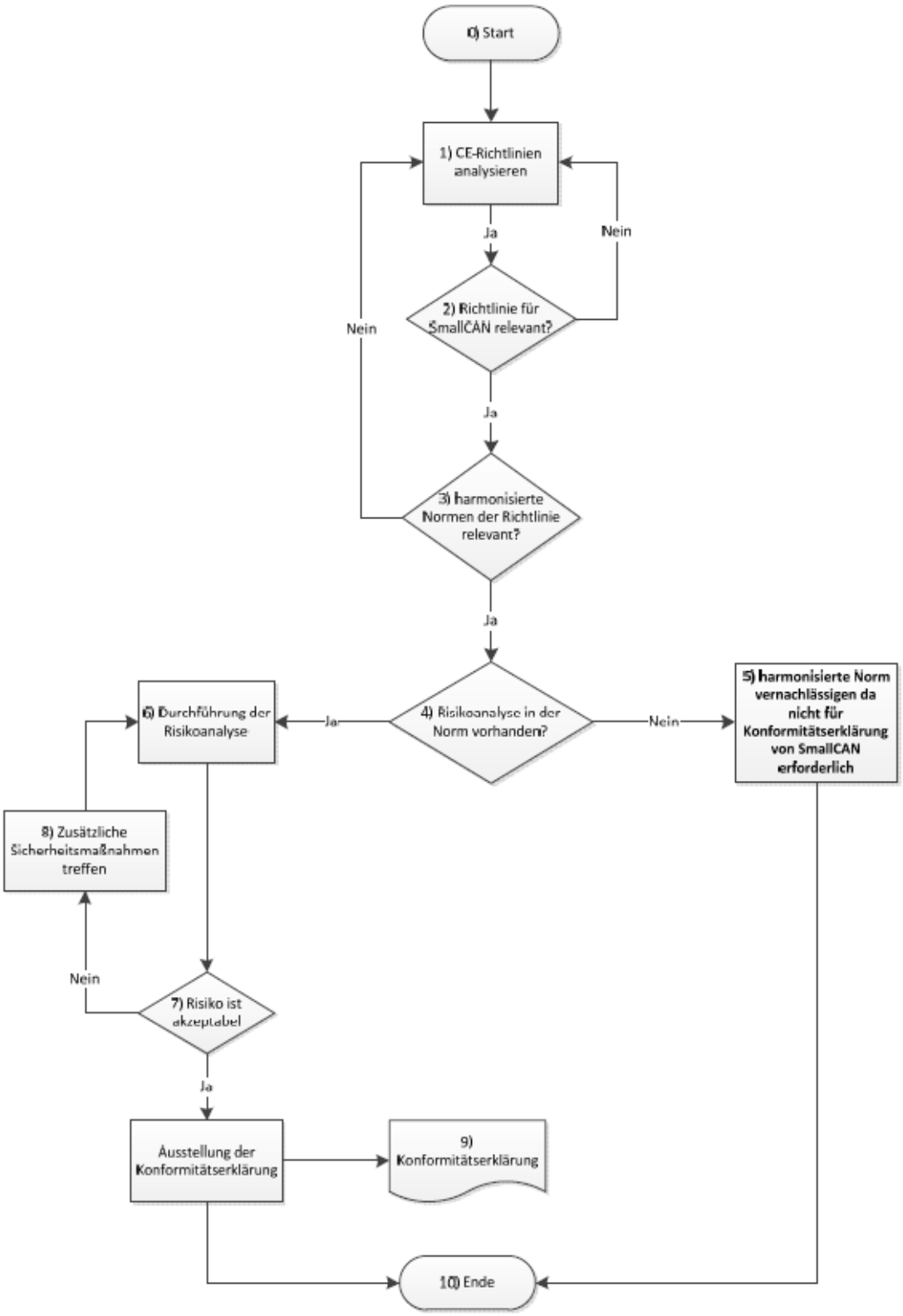


Abbildung 20: Flussdiagramm

Im Folgenden wurden die 3 identifizierten Richtlinien im Detail hinsichtlich ihrer Relevanz für SmallCAN überprüft. Hierbei wurde festgestellt, dass sich die Niederspannungsrichtlinie lediglich auf Spannungsbereiche zwischen 50V und 1000V Wechselspannung beziehungs-

weise zwischen 75V und 1500V Gleichspannung bezieht. Da SmallCAN nur mit einer Spannung zwischen 20V und 30V betrieben wird, konnte die Niederspannungsrichtlinie im Folgenden vernachlässigt werden.

Des Weiteren kann auch die Bauproduktrichtlinie ausgeschlossen werden, da diese nach detaillierterer Analyse keine für das Produkt SmallCAN-Buskoppler relevanten Normen enthält. Nach einer ersten Untersuchung der harmonisierten Normen, sind im Folgenden 6 harmonisierte Normen weiter zu verfolgen (siehe Tabelle 4).

	Richtlinien Bezeichnung	Richtlinie zur CE-Kennzeichnung	Harmonisierte Normen	Normtitel
1	2004/108/EG	EMV-Richtlinie	EN 50090	Elektrische Systemtechnik für Heim und Gebäude (EHSG)
2	2004/108/EG	EMV-Richtlinie	EN 60730	Automatische elektrische Regel- und Steuergeräte für den Hausgebrauch und ähnlichen Anwendungen
3	2004/108/EG	EMV-Richtlinie	EN 60947	Niederspannungsschalter
4	2004/108/EG	EMV-Richtlinie	EN 61000	Elektromagnetische Verträglichkeit
5	2004/108/EG	EMV-Richtlinie	EN 61131	Speicherprogrammierbare Steuerungen
6	2004/108/EG	EMV-Richtlinie	EN 61326	Elektrische Mess-, Steuer-,Regel- und Laborgeräte EMV-Anforderung

Tabelle 4: Liste von relevanten Richtlinien und harmonisierter Normen

Um eine Konformitätserklärung zu erhalten, sind nach [3] die harmonisierten Normen relevant, die die Anforderung der Durchführung einer Risikoanalyse enthalten. Diese Anforderung zur Durchführung einer Risikoanalyse ist nur in der Norm EN 50090 enthalten. Sie verweist auf hinsichtlich der Durchführung einer Risikoanalyse auf das in der Sicherheitsgrundnorm DIN EN 61508 beschriebene Prozedere zur Risikoanalyse

2.3.6 Durchführung der Risikoanalyse für SmallCAN-Buskoppler

Der Startpunkt einer jeden Risikoanalyse ist eine detaillierte Systembeschreibung. Im Rahmen der Durchführung der Risikoanalyse für den SmallCAN-Buskoppler wird hierbei auf ein Schema-Darstellung der Systemarchitektur (siehe: Abbildung 21) der Funktionsweise des Systems SmallCAN aufgesetzt.

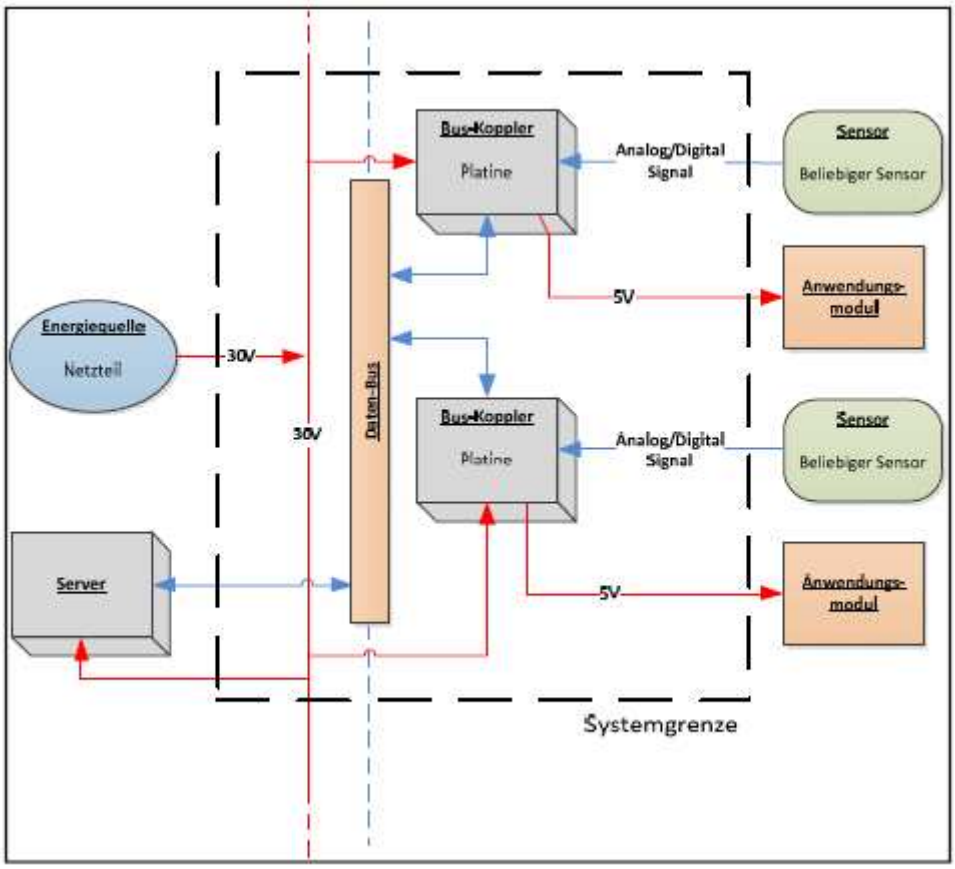


Abbildung 21: Systemarchitektur

Durch die Systemarchitektur werden alle externen und internen Schnittstellen des Systems visualisiert und dokumentiert (siehe Tabelle 5). Diese Schnittstellen und die darauf realisierten Funktionen (siehe Tabelle 5) werden in der Risikoanalyse betrachtet und auf mögliche Risiken geprüft (siehe Tabelle 6).

Anhand der in Abbildung 2 dargestellten Tabelle aus EN 61508 werden die verschiedenen möglichen Fehlfunktionen hinsichtlich:

- C: Schadensausmaß
- F: Häufigkeit und Aufenthaltsdauer
- P: Möglichkeit der Gefahrenabwehr
- W: Wahrscheinlichkeit des unerwünschten Ereignisses

abgeschätzt und kategorisiert. Im Anschluss daran werden die möglichen Fehlfunktionen unter Verwendung des in Abbildung 19 dargestellten Risikographen analysiert. Anhang A2 gibt einen Überblick über die Ergebnisse der durchgeführten Risikoanalyse für das System SmallCAN (ohne Applikation). Tabelle 7 stellt noch einmal die Fehlfunktionen mit dem größten Risikopotenzial dar.

Externe Schnittstellen

Schnittstellen zwischen Systemkomponenten übrigen Systemen

Interne Systemkomponente	Signalfluss	Signaltyp	Externe Systemkomponente	Realisierte Funktion
Buskoppler	<--	30V	Energiequelle	Spannungsversorgung Buskoppler
Buskoppler	-->	5V	Anwendungsmodul	Ansteuerung Anwendungsmodul
Buskoppler	<--	Analog/Digital	Sensor	Sensordaten übertragen
Daten-Bus	<-->	Digital / 24 V	Server	Datenaustausch

Interne Schnittstellen

Schnittstellen zwischen Systemkomponenten übrigen Systemen

Interne Systemkomponente	Signalfluss	Signaltyp	Externe Systemkomponente	Realisierte Funktion
Buskoppler	<-->	Digital / 24V	Daten-Bus	Datenaustausch

Tabelle 5: Schnittstellen des Sytstems

Nr.	Funktion	Fehlfunktionsbezeichnung	Nr.	Mögl. Fehlfunktionen	Kommentar
1	Spannungsversorgung des Bk	Fehlerhafte Spannungsversorgung des Bk	1	Zu hohe Spannung	Mögliches Abrauchen des Bk/ Brandgefahr
			2	Zu niedrige Spannung	Ausfall Bk / --> Fehlfunktion Anwendungsmodul
			3	Verpolung	Mögliches Abrauchen des Bk/ Brandgefahr
2	Ansteuerung Anwendungsmodul	Fehlerhafte Ansteuerung Anwendungsmodul	1	Ansteuerung ohne Anforderung	Gefährdung durch Anwendungsmodul
			2	Keine Ansteuerung trotz Anforderung	Gefährdung durch Anwendungsmodul
			3	Fehlerhafte Ansteuerspannung	Gefährdung durch Anwendungsmodul
3	Sensordaten übertragen	Fehlerhaftes Übertragen der Sensordaten	1	Fehlerhafte Einlesen von Sensordaten	-->Falsche Berechnung von Steuergrößen-->Gefährdung durch Anwendungsmodul
			2	Keine Übertragung der Sensordaten	-->Falsche Berechnung von Steuergrößen-->Gefährdung durch Anwendungsmodul
4	Datenaustausch Daten-Bus-->Server	Fehlerhafter Datenaustausch Server->Daten-Bus	1	Verfälschen von Telegrammen	-->Falsche Berechnung von Steuergrößen-->Gefährdung durch Anwendungsmodul
			2	Ausfall des Daten-Bus	Gefährdung durch Anwendungsmodul
5	Datenaustausch Buskoppler-->Daten-Bus	Fehlerhafter Datenaustausch Buskoppler-->Daten-Bus	1	Senden fehlerhafter Telegramme	Gefährdung durch Anwendungsmodul
			2	Zu häufiges Senden von Telegrammen	Gefährdung durch Anwendungsmodul
			3	Kein Senden von Telegrammen	Gefährdung durch Anwendungsmodul
			4	Fehlerhaftes Empfangen von Telegrammen	Gefährdung durch Anwendungsmodul
			5	Kein Empfang von Telegrammen	Gefährdung durch Anwendungsmodul
			6	Schwankende Signalspannung	Gefährdung durch Anwendungsmodul
6	Verarbeitung von Daten	Fehlerhafte Verarbeitung von Daten	1	Fehlerhafte Umsetzung der empfangenen Daten	Gefährdung durch Anwendungsmodul
			2	Keine Umsetzung der empfangenen Daten	z.B. uC hängt in Warteschleife /Gefährdung durch Anwendungsmodul

Tabelle 6: Liste der Fehlfunktionen

Nr.	Fehlfunktions-bezeichnung	Nr.	Mögl. Fehlfunktionen	Mögliches unerwünschtes Ereignis	Abschätzung der notwendigen Risikominderung (EN 61508)				Notwendige minimale Risiko-minderung (a..g)	SIL (1..4)	Bereits getroffene Maßnahmen	Zusätzlich zu treffende Maßnahmen gemäß EN 61508-2
					Auswirkung [C] (1..4)	Häufigkeit und Aufenthalts dauer [F] (1..2)	Möglichkeit den gefährlichen Vorfall zu vermeiden [P] (1..2)	Wahrscheinlich keit des unerwünschten Ereignisses [W] (1..3)				
1	Fehlerhafte Spannungsversorgung des Buskoppler (BK)	1	Zu hohe Spannung	Rauchentwicklung / Brandgefahr	2	2	1	1	b	1	HW: Strombegrenzer; Spannungsregler (V_{out}=50V); Supressor-Diode gegen Spitzenspannung Organisatorisch: Bedienungsanleitung, Anschluss durch Fachpersonal	keine
		3	Verpolung	Rauchentwicklung / Brandgefahr	2	2	1	1	b	1	HW: Verpolungsschutzdiode; mechanische Codierung Organisatorisch: Bedienungsanleitung, Anschluss durch Fachpersonal	keine

Tabelle 7: Zusammenfassung der Ergebnisse der Risikoanalyse (Auszug)

Bei genauerer Betrachtung stellt man fest, dass selbst die im Rahmen der Risikoanalyse mit einem SIL 1 eingestuften Funktionsumfänge, keiner Implementierung zusätzlicher sicherheitstechnischer Maßnahmen bedürfen. Diese Entscheidung liegt darin begründet, dass die SIL1-spezifischen Anforderungen aus der IEC 61508 bereits im Rahmen des Entwicklungsprozesses beachtet und umgesetzt wurden.

3 Status des Systems

Durch diese erste Grundentwicklung des SmallCAN ist es gelungen einen wettbewerbsfähigen Buskoppler mit einigen Anwendungsapplikationen zur Steuerung bzw. Regelung der Energieströme in einem Gebäude aufzubauen. SmallCAN steht dabei z.B. mit dem durch viele Firmen unterstützten Bussystem zur Gebäudeautomatisierung KNX/EIB im Wettbewerb.

Wir sehen hier doch einige Vorteile für den SmallCAN. Z.B. bezüglich der Verschaltungstechnik können bei KNX/EIB maximal 64 Teilnehmern in einer Linie verschaltet werden, bei mehreren Teilnehmern ist dann ein neues Netzteil notwendig. Bei SmallCAN können ca. 1000 Teilnehmern in einer Linie verschaltet werden mit einem Netzteil. Bei dem SmallCAN system ist ein autarker Betrieb möglich, da die Regelungs- und Steuerungssoftware auf die einzelnen Buskoppler verteilt ist und frei programmierbar (dezentrale Organisation). Bei KNX ist eine Änderung der Steuerungssoftware kostenpflichtig und nicht vollständig frei programmierbar.

Der Energieverbrauch pro Busankoppler beträgt bei KNX 150 mW und bei SmallCAN 68mW. Die Kosten bei KNX betragen pro Busankoppler zwischen 42,23 EUR - 85,46 EUR und der SmallCAN liegt bei einem Verkaufspreis von z.Z. ca. 14 EUR bei einer Fertigung von 1000 Stück.

Als Beispiel für ein Sensormodul (CO2-Sensor) betragen die Kosten KNX = 277 EUR/ SmallCAN ca. 100 EUR und der Stromverbrauch KNX = 8mA / SmallCAN= 2mA.

Die Gesamtenergieersparnis bei SmallCAN gegenüber dem KNX-System hängt sehr stark von der Ausstattungstiefe ab. Gegen Ende des 2. Quartals 2012 wird ein Einfamilienhaus mit SmallCAN ausgerüstet. Anschließend ist ein direkter Vergleich möglich.

Einige Kostenbeispiele im Vergleich zeigt die folgende Tabelle 8.

SmallCAN	Preis/ Stück	KNX	Preis/ Stück
Rollo Schaltungen	ca. 80 €	Rollo Schaltungen	143,12 €
Thermostat-Regler	ca. 50 €	Thermostat-Regler	250,18 €
Bus Zentraleinheit	ca. 65 €	Bus Zentraleinheit	242,10 €
Schaltereingänge	ca. 30 €	2 fach Tastermodul	82,36 €
vier-Kanal-Halogen	ca. 55 €	vier-Kanal-Halogen	326,45 €
230 V dim.	cd. 90 €	230 V dim.	391,00 €
Steckdose	ca. 55 €	Steckdose	241,55 €
RS232	ca. 70€	RS232	143,12 €
Stellventile/Heizungsaktor	cda. 75 €	Stellventile/Heizungsaktor	255 €
230 V Schaltbar	ca. 60 €	230 V Schaltbar	241,55 €
M Bus Gateway (z.B. Wärmemengenzähler)	ca. 60 €	M Bus Gateway (z.B. Wärme	903,76 €
DALI Gateway	ca. 45 €	DALI Gateway	581,40 €

Tabelle 8: Vergleich der Kosten verschiedener Applikatione SmallCAN / KNX

Bei den Applikationen (Aktoren) und Sensoren sind es z.Z. Kleinserien bis max. 20 Einheiten. Hier ist durchaus durch eine entsprechende Fertigung von größeren Einheiten noch eine Kostenreduzierung zu erwarten.

Durch selbst zu erstellende, individuelle Softwarekomponenten, die jederzeit erweitert werden können, ist es möglich sämtliche Informationen als Nachrichten auf den Bus abzulegen, die weitergenutzt werden können. Die Vielfalt dieser Daten ermöglicht es, sehr detaillierte und effektive Regelungsstrategien durchzuführen. Weiter werden die Hardwarekomponenten mit umfangreicher Schaltungstechnik bestückt, beispielsweise zur Prüfung von Relaisausgängen. Diese können dann geprüft, und die Diagnosedaten auf den Bus abgelegt werden und stehen damit wieder anderen Anwendungen zur Verfügung.

Alle Daten werden über eine einheitliche Schnittstelle an die Benutzeroberfläche gegeben. Dadurch können auch alle Daten zur Anzeige gebracht werden ohne Investitionsaufwand zur Interoperabilität leisten zu müssen. Die Software für die Benutzeroberfläche bei SmallCAN ist Betriebssystemunabhängig und kann deshalb individuell gestaltet werden.

Ein Server für den Opensource Ansatz wird z.Z. aufgesetzt.

Bei der Ausstattung der Büroräume ist die funktionalen Spezifikation gleich und konstant. Dadurch ist der Gesamtenergieverbrauch ein Maß für die Energieoptimierung bei gleichbleibenden Komfort.

Beispielhafte Maßnahmen sind:

1. verschiedenen Temperaturregelungsmodis (Nachtabsenkung, Vorhaltebetrieb, Stützbetrieb, Komfortbetrieb),

2. Netztrennung von Verbrauchern zur Vermeidung von Standby-Verbräuchen etc.

3. Vermeiden von energetische nicht geeigneten Systemzuständen, hervorgerufen durch Benutzer, beispielsweise die Einführung von Temperaturobergrenzen und Temperaturuntergrenzen abhängig von Sommer und Winterzeit.

Bisher hat sich im Vergleich zu den anderen Wettbewerbern ein Energieverbrauch von nur ca. 25 % gegenüber den nicht mit SmallCAN ausgestatteten Räumen ergeben. Für eine verlässliche Aussage ist hierzu jedoch eine Langzeitmessung über mindestens 2 Jahr notwendig.

Die ersten bisherigen Ergebnisse zeigen, das durchaus nennenswerte Energieeinsparungen möglich sind. Die Kostenansätze des Systems sind wie gezeigt doch erheblich unter den Kosten von z.B. KNX/EIB.

Durch die aufgesetzten Folgeprojekte wird eine weiter Verbreitung der Vorhabensergebnisse erreicht. Dies geschieht auch z.B. durch die Einbindung der Handwerkskammer und damit der Verbreitung bei Elektroinstallateuren und Heizungsbauern. Die Herausforderung besteht im Wettbewerb zu KNX/EIB (quasi Standard). Hier ist durch aus auch ein miteinander denkbar.

Desweiteren sind weiter Veröffentlichungen auf entsprechenden Tagungen geplant.

4 Fazit

Durch die Förderung der DBU ist die Möglichkeit geschaffen worden, aus der Idee des SmallCAN mit einigen Prototypen erste Ergebnisse in Richtung eines Produktes zu erzielen. Der Buskoppler selber ist inzwischen in einer ersten Auflage mit 1000 Einheiten automatisiert gefertigt und bestückt worden. Von einigen Applikationen sind Kleinserien von 5- 20 Einheiten gefertigt.

Die Softwareentwicklung ist durch die Neuorientierung und -strukturierung noch nicht endgültig abgeschlossen und wird weiter entwickelt auch für neue Applikationen.

Die ersten Messergebnisse aus dem „future-workspace“ sind sehr vielversprechend gerade in Hinsicht auf den Standby-Verbrauch gerade im Vergleich zu den Mitbewerbern des Ausbaus (ca. nur 25% des Energieverbrauchs zu den Mitwebbewerbern).

Durch diese grundlegenden Vorarbeiten ist inzwischen eine weitere Förderung durch das BMWi erfolgt.

Seit dem 1.10.2011 wird das BMWi-Projekt „Digaflex - Demonstrationsanlage einer integrierten Gebäudeautomatisierung mit low-Power, low-Cost Ansatz und flexiblem Gerätespektrum und flexibler Konfiguration“, Kennzeichen 03ET1016A gefördert. Dieses Projekt ist gemeinsam mit dem Institut für Verkehrssicherheit und Automatisierungstechnik beantragt worden. Arbeitsziel des Projektes ist es in zwei beispielhaften Demonstrationsinstallationen in ausgedehnten Liegenschaften die Vorteile über einen längeren Zeitraum zu demonstrieren. Die Laufzeit beträgt 5 Jahre, wobei die letzten beiden Jahre ausschließlich dem Monitoring des installierten Systems dienen.

Weiter kann das System auch zur Komfortsteigerung, insbesondere durch neue Dienste zur Unterstützung von behinderten oder älteren Menschen dienlich sein. Hierzu wird eine Kooperation mit dem Braunschweiger Informatik- und Technologie-Zentrum, die BITZ GmbH angestrebt. Hier ist insbesondere das Projekt ehealth.Braunnschweig (http://www.ehealth-braunschweig.de/) zu nennen.

Erste Schritte in Richtung Ausbildung sind durchgeführt.

Prof. Dr.-Ing. Yongjian DING von der Fachhochschule Magdeburg Stendal wir im Frühjahr 2011 ein kleine Demonstrationswand erhalten um diese in der Lehre in einem Labor einzusetzen.

Mit der Handwerkskammer Braunschweig finden z.Z. sehr vielversprechende Gespräche statt das System im Berufsbildungszentrum einzusetzen.

Z.Z. werden außerdem mit weiteren potentiellen Kandidaten erfolgversprechende Gespräche über den Demonstrationseinsatz von SmallCAN geführt (u.a. Solvis Braunschweig).

Das Gesamtfazit ist durch die erreichten Ziele positiv, es haben sich aber auch noch einige Herausforderungen während der Entwicklung gezeigt (wie z.B. bei der Softwareentwicklung beschrieben).

4 Literaturverzeichnis

Quellenverzeichnis

[1] Deutschen Patent- und Markenamt, Patent-Nr. 10034087: Feldbussystem mit minimierter Hardwarearchitektur.

[2] Schrom, H.: Realisierung eines optimierten Feldbussystems und Modellierung mit Petrinetzen. Dissertation, TU-Braunschweig, 2003.

[3] Schrom, H.: Optimiertes Feldbussystem. VDM-Verlag, Saarbrücken, 2007

[4] Schrom, H., Schnieder, E.: SCAN, A hardware minimised low cost / low power bus. MICRO.tec 2000, Hannover, Germany, 25.-27. September 2000.

[5] Schrom, H., Pfeiffer, A., Schnieder, E.: SmallCAN, ein Low-Power, Low-Cost, openSource Feldbussystem mit integraler Energieversorgung und hoher Teilnehmeranzahl. Elektor, eingereicht.

[6] http://www.eg-richtlinien-online.de/cn/bGV2ZWw9dHBsLXJpY2h0bGluaWVu.html , Letzter Zugriff 02.12.2012

[7] Losano, Mario G.: Turbulenzen im Rechtssystem der modernen Gesellschaft: Pyramide, Stufenbau und Netzwerkcharakter der Rechtordnung als ordnungsstiftende Modelle. Duncker+Humblot, 2007

[8] Jo Horstkotte: CE-Zeichen für Chefs: Was Sie zur CE-Kennzeichnungspflicht wissen sollten. 2010

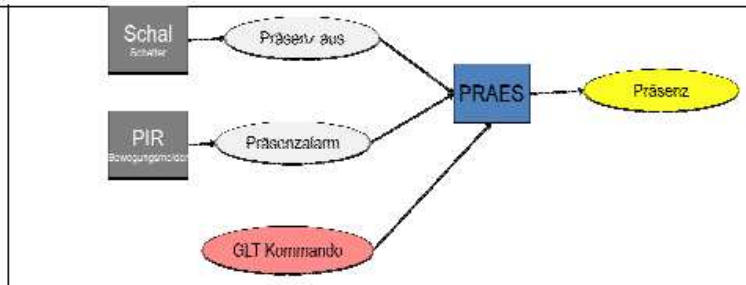
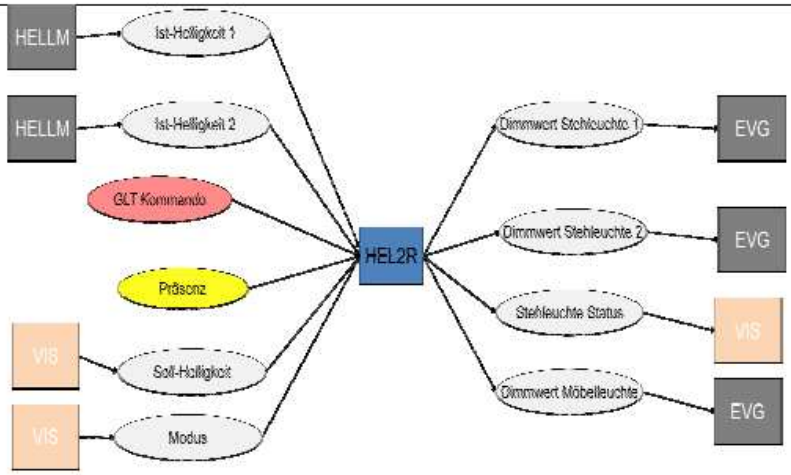
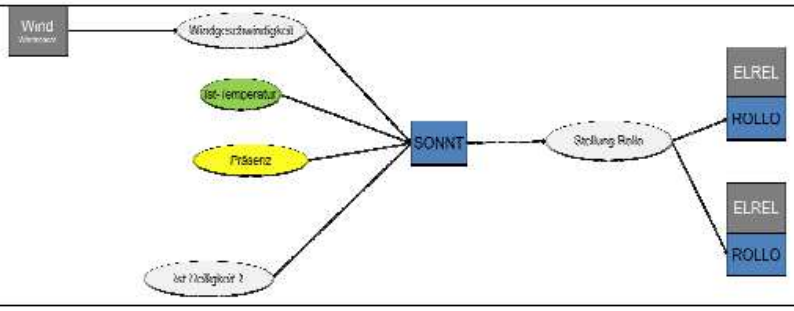
[9] Diekhake, P.; Fähndrich, E.; Schnieder, E.; Becker, U.: SmallCAN: Integrierte Gebäudeautomatisierung durch einheitliches low-Power, low-Cost, OpenSource Feldbussystem. Automation 2011, Baden-Baden, Deutschland, Juni 2011.

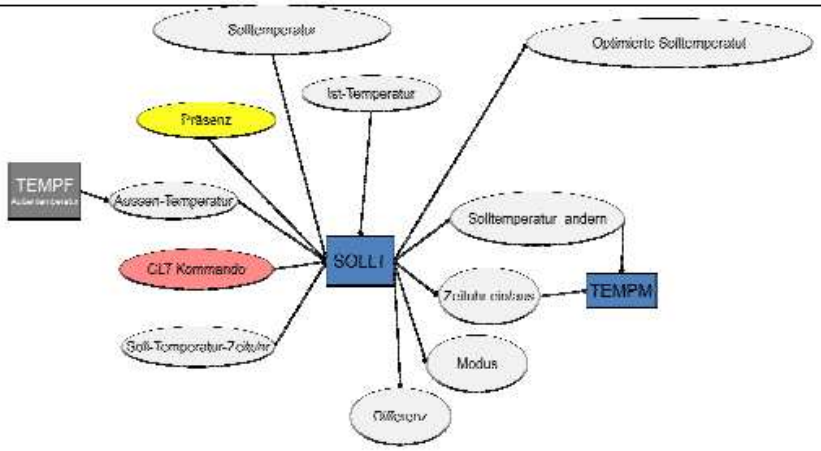
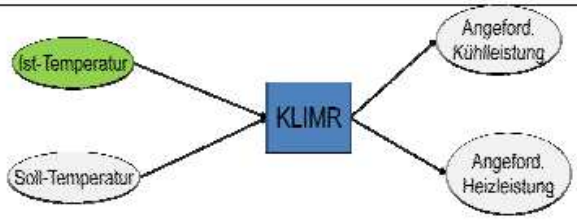
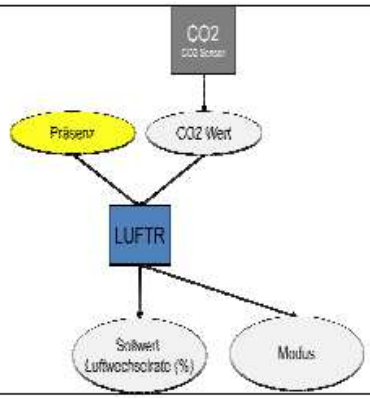
5 Anhänge

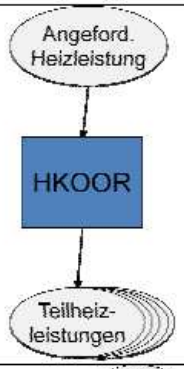
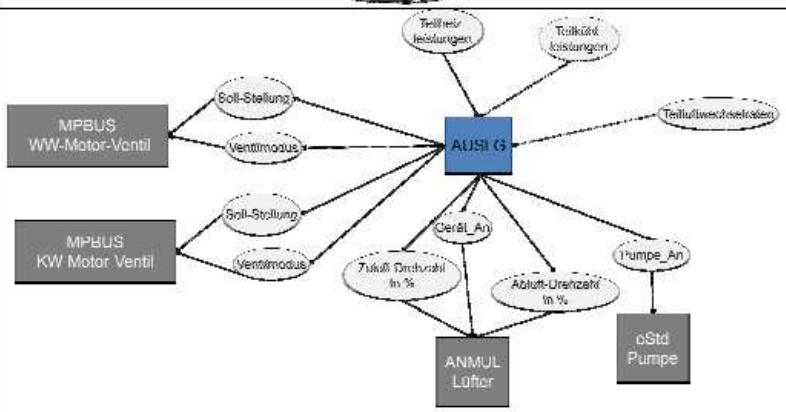
5.1 Anhang A1

Folgende wesentlichen verarbeitenden Funktionen seien in Tabelle A1 beschrieben und graphisch dargestellt.

Tabelle A1: wesentliche entwickelte verarbeitenden Funktionen.

PRAES_FSF (Präsenzdetektor)	Diese Funktion liest die Präsenzmeldungen des PIR Sensors eines Raums, Gebäudeleittechnikkommandos und ein Schalterstatus ein und ermittelt daraus eine kurze oder lange Präsenz	
HEL2R_FSF (Helligkeitsregler)	Diese Funktion liest die kurze oder lange Präsenz vom Präsenzmaster, die aktuellen Gebäudeleittechnik-Kommandos, die aktuelle Helligkeit, eine Soll-Helligkeit und einen Modus-Schalter für den Regler-Betrieb und steuert entsprechend Möbelleuchten und Stehleuchten an.	
SONNS_FSF (Sonnenschutz)	Diese Funktion liest die Windgeschwindigkeit sowie die Sonneneinstahlung ein und steuert bei Präsenz entsprechend eine	

	Beschattungs-anlage	
SOLLT (Solltempera-turbestimmun g)	Diese Funktion liest die Außen-lufttemperatur, Präsenz, Soll-temperaturen des Nutzers und der Zeituhr sowie die Raumtempera-tur ein und ermittelt mit-tels Behaglich-keitsdiagramm eine optimierte Solltemperatur	
KLIMR_FSF (Klimaregler)	Diese Funktion liest die Soll-Temperatur und die Ist-Temp eines Raums und berechnet da-raus eine Leistungszu- bzw. -abfuhr, um die ge-wünschte Raumtempera-tur zu errei-chen.	
LUFTR_FSF (Lüftungsmas-ter)	Diese Funktion liest den Ist-CO2-Wert ein und den Soll-wert aus dem EEPROM und berechnet da-raus eine Luftwechselrate in % und gibt den Modus aus.	

HKOOR_FSF Heizungskoordinator	Diese Funktion liest die Soll-Leistungsanforderung für den jeweiligen Raum und verteilt diese Anforderung auf bis zu 8 zur Verfügung stehenden Geräte	
AUSLG_FSF (Außenluftgerät)	Diese Funktion liest zahlreiche Eingangswerte (Heizteilleistung, Kühlteilleistung, Teilluftwechselrate) und steuert auf dieser Basis ein Außenluftgerät, das aus mehreren Buskopplern besteht (Ventile, Lüfter, Pumpe)	

Exemplarische verarbeitende Funktion für das Außenluftgerät

Am Beispiel einer eines Außenluftgerätes soll die verarbeitende Funktion näher erläutert werden. Ein Lüftergerät der Firma Trox besteht aus Radiallüfter für Zu- und Abluft und einer einfachen Lüfterelektronik die Stufenlos über eine 0-10V Schnittstelle angesteuert werden kann, und somit die Lüftergeschwindigkeiten bestimmt. Wird die Lüfterelektronik nicht mit Energie gespeist, schließt sich automatisch die Außenklappe. Das Lüftergerät verfügt weiter über Kalt- und Warmwasseranschlüsse sowie einen Konvektor zur thermischen Energieübertragung. Zum Betrieb des Klimagerätes muss neben der Ansteuerung des Lüftergerätes auch Warm- und Kaltwasserventile sowie eine Umwälzpumpe betrieben werden. Die Ansteuerung des Lüftergerätes erfolgt über ein Analogmodul. Die Ansteuerung der digitalen Stellventile wird durch ein Anwendungsmodul zur Ansteuerung des firmeninternen Busses der Fa. Belimo (siehe Tabelle xy: Nr 36) durchgeführt. Die Umwälzpumpe wird über eine Relaisplatine geschaltet. Diese vier vernetzten SmallCAN Komponenten (Lüftergeräte, Kalt- und Warmwasser-Ventile, Umwälzpumpe) müssen sinnvoll miteinander kommunizieren um ein Klimagerät zu bilden. Dazu liest die verarbeitende Funktion AUSLG die Sollwerte für Heizleis- tung, Kühlleistung oder der erforderlichen Luftwechselrate ein und steuert entsprechend die Einzelkomponenten an. Da diese Interaktionen ausschließlich nachrichtenbasiert

erfolgen, ist die Lokalisierung dieses Softwaremoduls frei wählbar. Für den Modus einer geforderten Luftwechselrate werden die Stellventile geschlossen und Frischluft durch das Öffnen der Außenluftklappe und Ansteuern der Lüfter in den zu versorgenden Raum geführt.

Folgender Quellcode ist dafür erforderlich.

```
//Lüften
if (Luftwechselrate_Anforderung>0)//wenn Anforderung vom Lüftungsregler
 {
if (!Luefter_An) // Wenn die Lüfter noch aus sind, dann einschalten
 {
Luefter_An=1;
FL_SEND_POWER = 1;
}
Luefter_Abluft = Luftwechselrate_Anforderung;
Luefter_Zuluft = Luftwechselrate_Anforderung;
FL_SEND_ABLUFT = 1;
FL_SEND_ZULUFT = 1;
 FLS_FREI1 = 1;
}
```

Damit bei niedrigen Temperaturen (Außentemperatur <=5℃) der Konvektor nicht einfriert ist ein Frostschutz zu realisieren, der folgende Spezifikationen einhalten muss:

- Anfahrschaltung bei jedem Gerätestart: Vor der Öffnung der Außenluftklappe muss zunächst das Ventil geöffnet werden bis die Differenz von Rücklauftemperatur und Vorlauftemperatur 5 Kelvin entspricht (TRL=TVL-5K). Erst dann wird die Stromversorgung für die Lüfter eingeschaltet, da sich hierdurch die Außenluftklappe öffnet.
- Die Rücklauftemperatur darf nicht kleiner als 25℃ sein
- Wenn die Pumpe stoppt (keine Umwälzung), schaltet die Stromversorgung für dieLüfter ab und die Außenluftklappe schließt.
- Sollte das Anfahren 3x hintereinander fehlschlagen, so wird ein Fehler gemeldet und das Gerät verriegelt. Die Entriegelung erfolgt durch den Administrator

Abbildung A1 stellt die Spezifikationen des Frostschutzes in einem Petrinetz dar. Unter der Bedingung, dass die Außentemperatur kleiner gleich 5℃ ist der Frostschutz aktiv und es muss die Rücklauftemperatur größer 25℃ und das Ger ät entriegelt sein, damit das Warm-

wasserventil geöffnet werden darf. Kann nicht innerhalb einer vorgegebenen Zeit die Differenz zwischen Vorlauftemperatur und Rücklauftemperatur erreicht werden, erfolgt der ers- te Fehlversuch. Nach drei Fehlversuchen wird das Gerät verriegelt und kann nur über einen Reset über den Bus wieder entriegelt werden. Ist die Anfahrschaltung erfolgreich darf das Gerät eingeschaltet und damit die Außenklappe geöffnet werden. Falls bei eingeschalteten Gerät die Rücklauftemperatur unter 25℃ sinkt oder die Pumpe stoppt, wird das Lüftergerät automatisch ausgeschaltet.

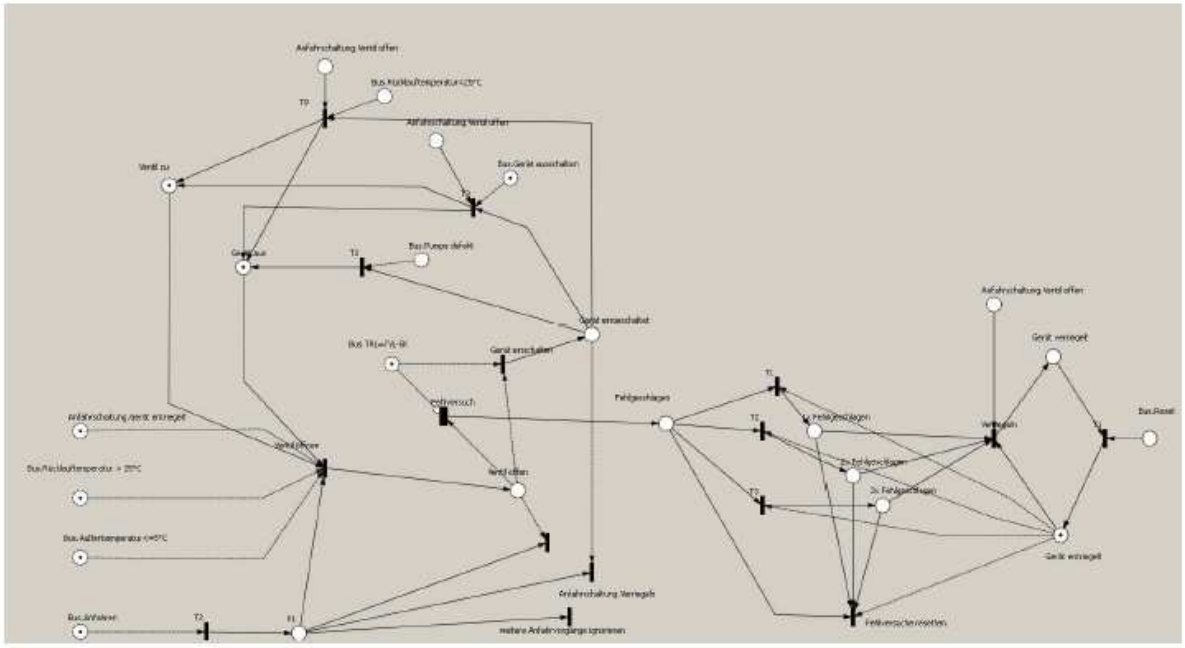


Abbildung A1: Petrinetz-Modell für den Frostschutz des Außenluftgerätes

Um den Frostschutz zu ermöglichen muss die verarbeitende Funktion AUSLG auch die Messwerte der Sensoren für Außenluft Vorlauf- und Rücklauf-Temperaturen sowie den Pumpenzustand erfassen. Der Quellcode der verarbeitenden Funktion AUSLG inklusiv der Frostschutzfunktion beläuft sich auf 840 Zeilen C-Code.

5.2 Anhang A2 Risikoeinstufung

Nr.	Fehlfunktions-bezeichnung	Nr.	Mögl. Fehlfunktionen	Mögliches unerwünschtes Ereignis	Abschätzung der notwendigen Risikominderung (EN 61508)				Notwendige minimale Risiko-minderung	SIL	Bereitsgetroffene Maßnahmen	Zusätzlich zu treffende Maßnahmen gemäß EN 61508
					Auswirkung [C]	Häufigkeit und Aufenthaltsdauer [F]	Möglichkeit den gefährlichen Vorfall zu vermeiden [P]	Wahrscheinlichkeit des unerwünschten Ereignisses [W]				
1	Fehlerhafte Spannungsversorgung des Buskoppler (Bk)	1	Zu hohe Spannung	Rauchentwicklung / Brandgefahr	2	2	1	1	b	1	**HW:** Strombegrenzer; Spannungsregler (V_{max}=50V); Supressor-Diode gegen Spitzenspannung **Organisatorisch:** Bedienungsanleitung, Anschluss durch Fachpersonal	keine
		2	Zu niedrige Spannung	Ausfall Bk (ab 17V) / --> Fehlfunktion Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**HW:** Stützkondensator, Pull-Down, Pull-Up **SW:** Programmroutine (Buskoppler in FailSafe)	keine
		3	Verpolung	Rauchentwicklung / Brandgefahr	2	2	1	1	b	1	**HW:** Verpolungsschutzdiode; mechanische Codierung **Organisatorisch:** Bedienungsanleitung, Anschluss durch Fachpersonal	keine
2	Fehlerhafte Ansteuerung Anwendungsmodul	1	Ansteuerung ohne Anforderung	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**Organisatorisch:** Software-/Hardwaretests	keine
		2	Keine Ansteuerung trotz Anforderung	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**Organisatorisch:** Software-/Hardwaretests	keine
		3	Fehlerhafte Ansteuerspannung	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**HW:** Stützkondensator, Pull-Down, Pull-Up **Organisatorisch:** Software-/Hardwaretests	keine
3	Fehlerhaftes Übertragen der Sensordaten	1	Fehlerhafte Einlesen von Sensordaten	-->Falsche Berechnung von Steuergrößen--> Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**HW:** Bei sicherheitsrelevanten Anwendungsmodulen ist Redundanz und Plausibilisierung über weiteren Bk möglich.	keine
		2	Keine Übertragung der Sensordaten	-->Falsche Berechnung von Steuergrößen--> Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Fehlerdiagnose, Fehler-Telegramm **Organisatorisch:** Visualisierung (z.B. über LEDs)	keine
4	Fehlerhafter Datenaustausch Server->Daten-Bus	1	Vorfälschen von Telegrammen	-->Falsche Berechnung von Steuergrößen--> Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Checksumme	keine
		2	Ausfall des Daten-Bus	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Fehlerdiagnose (durch Server) **Organisatorisch:** Visualisierung (z.B. über LEDs)	keine
5	Fehlerhafter Datenaustausch Buskoppler-->Daten-Bus	1	Senden fehlerhafter Telegramme	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Checksumme	keine
		2	Zu häufiges Senden von Telegrammen ("Babbling-Idiot-Problem")	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Lastbegrenzung (138 Bits/s), Überwachung durch Server, Abschalten (Bus-Off) und Reset des fehlerhaften Bk, Telegramm-Priorisierung	keine
		3	Kein Senden von Telegrammen	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Datenstau-Telegramm (Bk), Fehlerdiagnose (durch Server) **Organisatorisch:** Visualisierung (z.B. über LEDs)	keine
		4	Fehlerhaftes Empfangen von Telegrammen	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Checksumme	keine
		5	Kein Empfang von Telegrammen	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Fehlerdiagnose, Fehler-Telegramm (LOST-Telegramm) **Organisatorisch:** Visualisierung (z.B. über LEDs)	keine
		6	Schwankende Signalspannung	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Überwachung der Signalqualität, Fehler-Telegramm **Organisatorisch:** Visualisierung (z.B. über LEDs)	keine
6	Fehlerhafte Verarbeitung von Daten	1	Fehlerhafte Umsetzung der empfangenen Daten	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**HW:** Bei sicherheitsrelevanten Anwendungsmodulen ist Redundanz und Plausibilisierung über weiteren Bk möglich. **Organisatorisch:** Software-/Hardwaretests	keine
		2	Keine Umsetzung der empfangenen Daten	z.B. uC hängt in Warteschleife /Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Betriebssystem arbeitet "Quasi-Parallel" (keine Warteschleife)	keine

iQST GmbH

Institute for Quality, Safety and Transportation

Geräteentwicklung für ein integrales und energie-sparendes Feldbussystem

Abschlussbericht über ein Entwicklungsprojekt,

gefördert unter dem AZ: 26964-21/0 von der

Deutschen Bundesstiftung Umwelt

Von

Dr.-Ing. Uwe Becker, Dr.-Ing. Tobias Ständer,

Steffen Bussenius, Sebastian Knüfermann

Braunschweig, April 2012

iQST GmbH

Institute for Quality, Safety and Transportation

Geräteentwicklung für ein integrales und energie-sparendes Feldbussystem

Abschlussbericht über ein Entwicklungsprojekt,

gefördert unter dem AZ: 26964-21/0 von der

Deutschen Bundesstiftung Umwelt

Von

Dr.-Ing. Uwe Becker, Dr.-Ing. Tobias Ständer,

Steffen Bussenius, Sebastian Knüfermann

Braunschweig, April 2012

06/02

Projektkennblatt
der
Deutschen Bundesstiftung Umwelt

Az	26964-21/0	Referat	21	Fördersumme	124.200,00 €

Antragstitel Geräteentwicklung für ein integrales und energiesparendes Feldbussystem

Stichworte

Laufzeit	Projektbeginn	Projektende	Projektphase(n)
2 Jahre	**3.3.2009**	**30.09.2011**	

Zwischenberichte

Bewilligungsempfänger iQST GmbH
Institute for Quality, Safety and Transportation
Hermann-Blenk-Str. 22
38108 Braunschweig

Tel 01635768550
Fax 0531 35444-16

Projektleitung
Dr.-Ing. Uwe Becker

Bearbeiter

Kooperationspartner

Zielsetzung und Anlaß des Vorhabens

Energie ist eine der wichtigsten Grundlagen moderner Gesellschaften, deren verantwortliche Nutzung aufgrund begrenzter Ressourcen immer dringlicher wird. Auch so genannte Kleinverbraucher benötigen infolge ihrer hohen Anzahl nennenswerte Energie. Insbesondere bei Geräten und Installationen mit sehr langer Betriebsdauer über mehrere Jahrzehnte sind (Fehl-)Entscheidungen und Investitionen unter energetischen Gesichtspunkten zu treffen. Zu diesen gehören elektrische Gebäudeinstallationen, die in hoher Stückzahl, fest montiert, langfristig hohe Energieverluste akkumulieren. Anreize zur Installation verbrauchsarmer Einrichtungen motivieren jedoch nur bei attraktiven Einstandspreisen bzw. Vertriebsstrategien. Mit dem neuen integrierten Gebäudeautomatisierungssystem SmallCAN soll der technologisch bedingte minimale Energieverbrauch adressiert werden, der bei einer attraktiven Preisbildung und mittels eines neuartigen Vertriebsgeschäftsmodells einen leichten Marktzugang ermöglicht. Arbeitsziel des Projektes ist, ausgehend von einer vorhandenen stabilen Grundlagenentwicklung eines energieminimalen Kommunikationssystems für die Gebäudetechnik und -automatisierung, das System durch Weiterentwicklung zur Praxistauglichkeit zu führen.

Darstellung der Arbeitsschritte und der angewandten Methoden

Neben der Vervollständigung des grundlegenden Systems ist eine Erprobung unter realistischen Bedingungen anhand einer Demonstrationsinstallation vorgesehen, welche die Praxistauglichkeit sowohl für den professionellen als auch den privaten Domotik-Markt nachweist. Zusätzlich soll die für den Praxiseinsatz notwendige Vielfalt und Bandbreite an Applikationsmodulen geschaffen werden.

Ergebnisse und Diskussion

Ausgehend von der Zielsetzung der Energieersparung und Integration der Gebäudeautomatisierung mit Bereitstellung neuer Funktionalitäten bei gleichzeitiger Kostenminimierung war es das Ziel, ein zuverlässiges und universelles low-Cost Feldbussystem für die Gebäudetechnik bereit zu stellen, dass sich durch geringen Energiebedarf und fast beliebige Erweiterbarkeit auszeichnet. Das vorhandene prototypische System ist in ersten Schritten zur Praxistauglichkeit weiter entwickelt worden. Die Server-Client-Architektur ist aufgrund der Anforderungen aus dem FutureWorkspace überarbeitet worden. Ebenfalls werden noch die Visualisierung überarbeitet.
Der Buskoppler selber ist inzwischen in einer ersten Auflage mit 1000 Einheiten automatisiert gefertigt und bestückt worden. Von einigen Applikationen sind Kleinserien von 5- 20 Einheiten gefertigt.
Die Softwareentwicklung ist durch die Neuorientierung und -strukturierung noch nicht endgültig abgeschlossen und wird weiter entwickelt auch für neue Applikationen.
Die ersten Messergebnisse aus dem „future-workspace“ (Implementierung in einem Büro der Zukunft im Wettbewerb zu anderen „Gebäudeautomatisierern“, Entwicklung von ca. 40 Anwendungsadaptern) sind sehr vielversprechend gerade in Hinsicht auf den Standby-Verbrauch und im direkten Vergleich zu den Mitbewerbern des Ausbaus.
Durch diese grundlegenden Vorarbeiten ist inzwischen eine weitere Förderung durch das BMWi erfolgt.
Weiter kann das System auch zur Komfortsteigerung, insbesondere durch neue Dienste zur Unterstützung von behinderten oder älteren Menschen dienlich sein. Hierzu wird eine Kooperation mit dem Braunschweiger Informatik- und Technologie-Zentrum, die BITZ GmbH angestrebt. Hier ist insbesondere das Projekt ehealth.Braunnschweig (http://www.ehealthbraunschweig.de/) zu nennen.
Erste Schritte in Richtung Ausbildung sind durchgeführt.
Prof. Dr.-Ing. Yongjian DING von der Fachhochschule Magdeburg Stendal wir im Frühjahr 2011 ein kleine Demonstrationswand erhalten um diese in der Lehre in einem Labor einzusetzen

Öffentlichkeitsarbeit und Präsentation

Diekhake, P.; Fähndrich, E.; Schnieder, E.; Becker, U.: SmallCAN: Integrierte Gebäudeautomatisierung durch einheitliches low-Power, low-Cost, OpenSource Feldbussystem.
Automation 2011, Baden-Baden, Deutschland, Juni 2011.

Einbau des SmallCAN im FutureWorkspace der TU Braunschweig in einem Büroraum.

Fazit

Durch die Förderung der DBU ist die Möglichkeit geschaffen worden, aus der Idee des SmallCAN mit einigen Prototypen erste Ergebnisse in Richtung eines Produktes zu erzielen. Durch diese Vorarbeiten ist es gelungen weiter Förderprojekte (Seit dem 1.10.2011 wird das BMWi-Projekt „Digaflex - Demonstrationsanlage einer integrierten Gebäudeautomatisierung mit low-Power, low-Cost Ansatz und flexiblem Gerätespektrum und flexibler Konfiguration“, Kennzeichen 03ET1016A gefördert. Dieses Projekt ist gemeinsam mit dem Institut für Verkehrssicherheit und Automatisierungstechnik beantragt worden. Arbeitsziel des Projektes ist es in zwei beispielhaften Demonstrationsinstallationen in ausgedehnten Liegenschaften die Vorteile über einen längeren Zeitraum zu demonstrieren.
Die Laufzeit beträgt 5 Jahre, wobei die letzten beiden Jahre ausschließlich dem Monitoring des installierten Systems dienen) zu erhalten sowie erste Schritte in Richtung Ausbildung durchzuführen

Inhalt

Verzeichnis Bilder

Verzeichnis Begriffe, Abkürzungen und Definitionen

GUI - Graphical User Interface

MBus - (Meter-Bus) ist ein Feldbus für die Verbrauchsdatenerfassung.

PIC - Peripheral Interface Controller

XML - xtensible Markup Language

Zusammenfassung

Energie ist eine der wichtigsten Grundlagen moderner Gesellschaften, deren verantwortliche Nutzung aufgrund begrenzter Ressourcen immer dringlicher wird. Insbesondere in Gebäuden lässt sich der Energiebedarf durch intelligente Lösungen wie z.B. Gebäudeautomatisierungssysteme (GA-Systeme) verbessern deren intelligente Vernetzung eine hohe Primärenergieeinsparung ermöglicht. GA-Systeme erfordern jedoch signifikante Kosten für Investition, Planung (Engineering), Installation und Betrieb (Energieverbrauch der Busteilnehmer, Wartungskosten) sowie bei Änderungen.

In dem Projekt „Geräteentwicklung für ein integrales und energiesparendes Feldbussystem“ konnte der praktische Einsatz des sich in der Grundlagenentwicklung befindliche energieoptimierte und kostenminimale Kommunikationssystem SmallCAN praktisch vollzogen werden. Die Vorteile dieses Kommunikationssystems gegenüber konventionellen Systemen konnte in dem Installationssobjekt zusammen mit dem Institut für Verkehrssicherheit und Automatisierungstechnik der TU Braunschweig im „future-workspace“ demonstriert werden. Es ist ein Büroraum im Wettbewerb mit etablierten GA-Ausrüstern aufgebaut worden. Die Verbrauchsmessungen werden z.Z. durchgeführt und erste Auswertungen sind für den Sommer 2012 zu erwarten. Mit der Weiterentwicklung des Systems für praxisnahe Gebäudeautomatisierungsanwendungen konnte ein umfangreiches Portfolio verschiedenster Anwendungsmodule geschaffen werden um Einzelkomponenten wie Pumpen, Ventile oder auch Lüfteranlagen individuell anzusteuern. Neue regelungstechnische Aufgaben der Gebäudeautomatisierung (z.B. Beispiel eine Helligkeitsregelung) können durch Zuhilfenahme vieler Akteure und deren kooperatives Zusammenspiel in modular entwickelten Softwaremodulen geschaffen werden. Durch einen externen Zugriff auf das System ist die vollständige Überwachung und Erprobung dieser Demonstrations-Installationen möglich. Durch den durch erste Messungen gezeigten geringen Energieverbrauch selbst sowie die intelligente Kooperation der Systemkomponenten durch flexibel konfigurierbaren und umfangreich auslegbaren Verarbeitungsfunktionen ist hinsichtlich der Energieeinsparungen von Gebäuden ein hohes Potenzial zu erwarten. Das Kommunikationssystem SmallCAN konnte hier erfolgreich in der Praxis umgesetzt werden. Damit lässt sich das System für viele weitere Anwendungsmöglichkeiten nutzen.

1 Einleitung

In vielen Bereichen der Gebäudetechnik werden Automatisierungssysteme angeboten, deren Nutzen auf die jeweilige Anwendung eingeschränkt ist. Die Stromversorgung dieser Systeme ist häufig dezentral und damit teuer (lokale Netzteile, Batterien), die Verkabelung sternförmig oder hierarchisch und damit aufwändig, eine Skalierung oder Erweiterung nicht oder lediglich mit hohem Aufwand möglich. Zu diesen Nachteilen der teils proprietären Pro- dukte werden häufig noch Probleme mit dem Datenübertragungsmedium beschrieben, ins- besondere bei Mitnutzung der Stromleitungen.

Auch so genannte Kleinverbraucher benötigen infolge ihrer hohen Anzahl nennenswerte Energie, wie es mittlerweile bei Glühlampen und Stand-by-Betrieb von Computern und elektronischen Unterhaltungsgeräten bekannt ist. Daher ist auch die Vorgabe von Energieeffizienz-Labeln mittlerweile öffentlicher Ausdruck eines sensiblen Energiebewusstseins bei elektrischen Alltagsgeräten geworden.

Mit einem neuen integrierten Gebäudeautomatisierungssystem soll neben der Energieeinsparung durch optimierte Betriebsführung auch der Eigen-Energieverbrauch des Systems selbst minimiert werden (geringer „Standby-Bedarf“ des Gebäudes). Wenn man sich vergegenwärtigt, dass in einem Gebäude mit mehreren hundert oder gar tausend Busgeräten die Busknoten 24 Stunden mit Energie versorgt werden müssen um die Funktion des Systems zu ermöglichen, so wird schnell deutlich, dass der Energiebedarf pro Busknoten von entscheidender Bedeutung von den „Standby-Verbrauch“ eines Gebäudes ist. Abgesehen von den batteriebetriebenen Geräten einiger Funk-Systeme, die in regelmäßigen Intervallen den Austausch von hunderten Batterien mit entsprechend negativen Umwelteffekten und Austauschkosten nach sich ziehen, weisen die bisherigen Systeme einen erheblichen Energiebedarf aus.

Derzeit existiert eine Reihe von spezialisierten Bussystemen der entsprechenden Gebäude- und Automatisierungsbranchen, die jeweils lediglich ihr Teilsegment der Gesamtheit aller Anwendungen abdecken. Diese Systeme sind zueinander inkompatibel und technisch so stark auf ihr Teilsegment ausgerichtet, so dass keine gemeinsame Basis gegeben ist.

Der praktische Einsatz dieser Systeme beschränkt sich derzeit auf einen geringen Anteil der potentiellen Gebäude, da aufgrund der nicht unerheblichen Kosten lediglich wenige Industrie- und Nutzbauten damit ausgerüstet werden. Dabei kommt es aufgrund der Teilabdeckungen häufig zu einer Parallelinstallation mehrerer Systeme, gegebenenfalls ergänzt um industrielle Prozessleittechnik (SPS-Schränke).

Der Eigenenergiebedarf der bisher installierten Systeme ist erheblich. Zur Energieversorgung der Komponenten werden meist lokale Kleinstnetzteile genutzt, deren Effizienz und Zuverlässigkeit begrenzt ist.

Gerade bei Geräten und Installationen mit sehr langer Betriebsdauer über mehrere Jahrzehnte sind Investitionen unter energetischen Gesichtspunkten als auch Wartungs und Installationskosten zu treffen. Zu diesen gehören elektrische Gebäudeinstallationen, die in hoher Stückzahl, fest montiert, langfristig hohe Energieverluste akkumulieren. Anreize zur Installation verbrauchsarmer Einrichtungen motivieren jedoch nur bei attraktiven Einstandspreisen bzw. Vertriebsstrategien.

Eine Integration aller Bereiche der Gebäudetechnik, sowohl gerätetechnisch als auch durch eine konsequent einfache und jederzeit erweiterungsfähige Verkabelungsstruktur, ist dabei Voraussetzung für eine intelligente Steuerung und Vernetzung des gesamten Gebäudes insbesondere unter energetischen Gesichtspunkten.

Mit dem neuen integrierten Gebäudeautomatisierungssystem SmallCAN soll der technologisch bedingte minimale Energieverbrauch adressiert werden, der bei einer attraktiven Preisbildung und mittels eines neuartigen Vertriebsgeschäftsmodells einen leichten Marktzugang ermöglicht.

Ausgehend von der Zielsetzung der Energieersparung und Integration der Gebäudeautomatisierung mit Bereitstellung neuer Funktionalitäten bei gleichzeitiger Kostenminimierung ist es das Ziel, ein zuverlässiges und universelles low-Cost Feldbussystem für die Gebäudetechnik bereit zu stellen, dass sich durch geringen Energiebedarf und fast beliebige Erweiterbarkeit auszeichnet.

Das vorhandene prototypische System soll zur Praxistauglichkeit entwickelt werden. Der

universelle Ansatz des Systems vereint die Vorteile der bisher verfügbaren Lösungen in einem einzigen System und ermöglicht somit eine Zusammenführung. Insbesondere der low-Cost Grundsatz und eine fast beliebige, auch nachträgliche, Erweiterbarkeit prädestinieren das System für einen breiten Einsatz in der Gebäudetechnik (bis zu 1000 Busteilnehmer an bis zu 1000 m Telefonkabel). Das System soll zuverlässig und wartungsfrei (kabelgebunden), vollständig dezentralisiert (ohne zentrale Steuereinheit, multimasterfähig) und ausreichend leistungsfähig sein.

Eine Erprobung des Systems unter realistischen Bedingungen soll im Labor durchgeführt sowie in der beispielhaften Installation validiert werden. Die daraus gewonnen Erkenntnisse sollen systematisch aufbereitet werden und in die Entwicklung zurück fließen.

Am Ende soll ein vollständiges und alle üblichen Applikationen abdeckendes praxistaugliches System in industrieller Qualität bereit stehen. Dies umfasst sowohl das eigentliche Bussystem, die diversen Applikationsadapter sowie die Software zur Konfiguration und Inbetriebnahme (im ersten Schritt tauglich für Fachinstallateure, später auch für Endkunden) beziehungsweise zur Visualisierung und Bedienung durch den Endkunden.

2 Geräteentwicklung Hard und Software

Ausgehend von einer vorhandenen stabilen Grundlagenentwicklung eines energieminimalen Kommunikationssystems, sollte durch Weiterentwicklung des Systems ein praxistaugliches Automatisierungssystem zur Verfügung gestellt werden.

Neben der Vervollständigung des grundlegenden Systems und Sicherung der Qualität ist eine Erprobung unter realistischen Bedingungen anhand einer Demonstrationsinstallation vorgesehen, welche die Praxistauglichkeit sowohl für den professionellen als auch den privaten Domotik-Markt nachweist.

Zusätzlich soll die für den Praxiseinsatz notwendige Vielfalt und Bandbreite an Applikationsmodulen geschaffen werden. Die Vorteile des SmallCAN Systems werden in einer exemplarischen Installation im Rahmen des Projektes „future-workspace“ in Zusammenarbeit mit dem Institut für Verkehrssicherheit und Automatisierungstechnik demonstriert (siehe Abbildung 1).

Abbildung 1: SmallCAN in der Demonstartionsanlage „future-workspace“. [Quelle: Hans Georg Esch fotografiert für Saint-Gobain Ecophon GmbH]

2.1 Hardwareentwicklung & Firmware

Für ein Laborsystem wurden insgesamt ca 40. Anwendungsadapter erstellt und auf eine Laborwand aufgebaut. Die Laborwand ist modular aufgebaut, womit sich das dort verbaute System zu jedem Zeitpunkt skalieren und umbauen lässt. An der Laborwand konnten neu entwickelte Anwendungsmodule hinsichtlich ihrer Funktion, sowie im Verbund mit anderen Busteilnehmern getestet werden und ggf. Anpassungen an Hardware und Software vorgenommen werden.

Für den Einbau von SmallCAN-Komponenten in reale Gebäudestrukturen wurde im Demonstrationssystem „future-workspace“ ein Büro mit dem Feldbussystem SmallCAN ausgestattet. Hier konnte in der Praxis die Bearbeitung von mess- und regelungstechnischen Aufgaben der Gebäudeautomatisierung durch das SmallCAN System getestet werden. Dazu mussten zunächst die Entwicklung und der Aufbau aller notwendigen Anwendungsadapter sowie deren vollständige Installation in das Demonstrationssystem erfolgen. Insgesamt sind im „future-workspace“ 86 Anwendungsmodule verbaut, die Sensoren und Aktoren verschiedenster Art über Telefonleitungen vernetzen.

2.1.2 Beispielhafte Beschreibung des Anwendungsmoduls für Wärmemengenzähler

Die Hardwareentwicklungsarbeiten sollen exemplarisch an der Ansteuerung eines Wärmemengenzählers der Firma Allmess durch ein SmallCAN-Anwendungsmodul erläutert werden. In Abbildung 2 ist der Wärmemengenzähler (WMZ) CF-Echo dargestellt, der zur Erfassung thermischer Größen wie angegebener Energie und Leistung von Klimageräten u.ä. genutzt werden kann. Der WMZ besteht aus einem Durchfluss-Sensor, Temperatursensoren zur Erfassung der Vorlauf- und Rücklauf-Temperaturen sowie ein Rechenwerk zur Bestimmung der thermischen Größen.

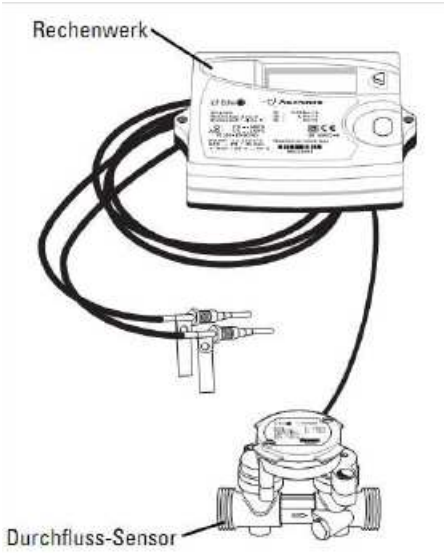


Abbildung 2: Wärmemengenzäher CF Echo der Fa. Allmess. [Quelle: Allmess]

Optional lässt sich ein Erweiterungsmodul auf das Rechenwerk aufstecken, um eine Schnittstelle an externe Geräte zu ermöglichen. Das MBUS-Schnittstellenmodul bietet die Möglichkeit über den etablierten Verbrauchsdatenerfassungsbus MBUS Daten extern bereitzustellen. Zur Erfassung wichtiger thermischer Größen in Gebäuden ist daher ein Anwendungsmodul für SmallCAN entwickelt worden, welches als Gateway zu MBUS fähigen Geräten der Gebäudeautomatisierung fungiert. Das Anwendungsmodul wird im Falle des Wärmemen-

genzählers CF-Echo an das MBUS-Schnittstellenmodul angeschlossen um mit dem Gerät zu kommunizieren.

Bei der Entwicklung der Hardwarekomponente ist die Einhaltung der elektrischen Übertragungsregeln des MBUS zu berücksichtigen. Nach den in der Bitübertragungsschicht des ISO- OSI Schichtenmodells einzuordnenden Spezifikationen ist der Schaltplan nach Abbildung 3 erstellt worden. Das Hardwaremodul ermöglicht weiterhin den Wärmemengenzähler mit Energie zu versorgen und macht ihn durch den Verzicht des internen Energiespeichers über einen sehr langen Zeitraum wartungsfrei. Um dem Ziel der Energieeinsparung gerecht zu werden, sind Bauteile geringer Leistungsaufnahme verwendet sowie schaltungstechnische Optimierungen durchgeführt worden.

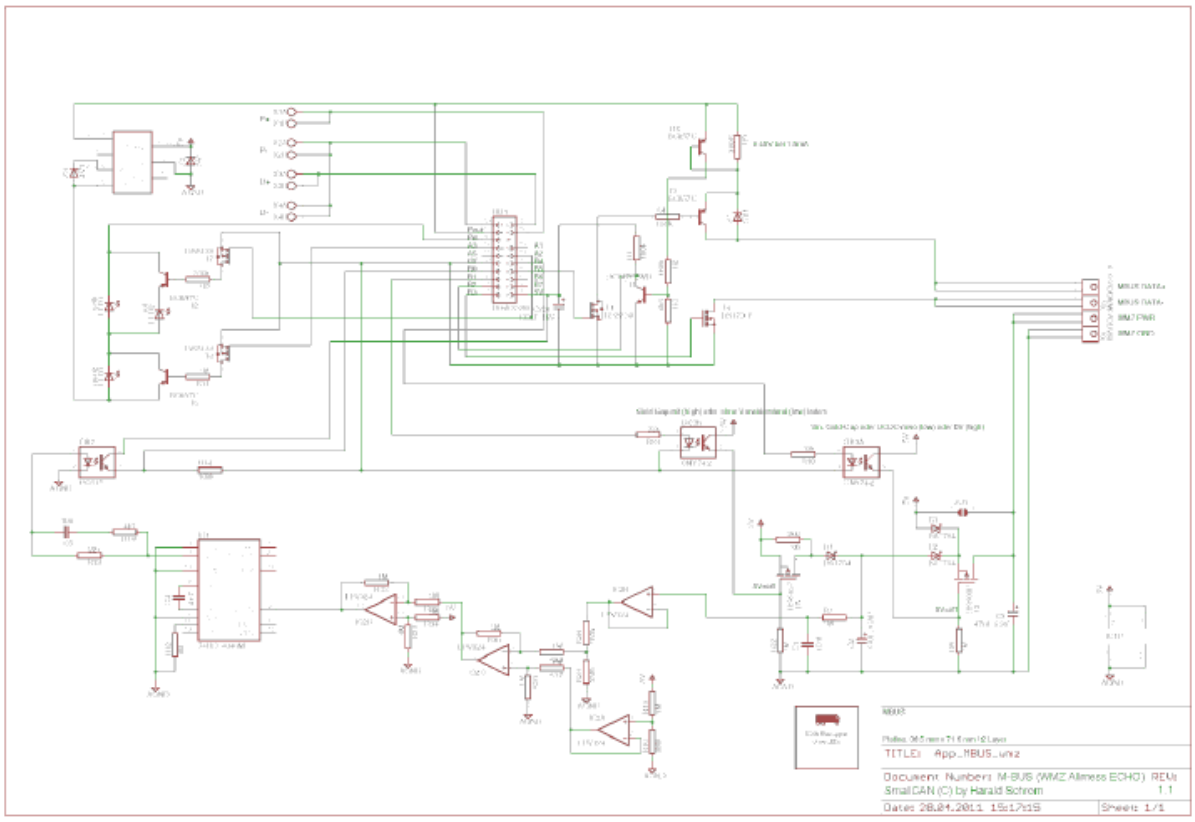


Abbildung 3: Schaltplan für das SmallCAN Anwendungsmodul „App_MBUS_wmz“.

Die Anordnung der einzelnen Bauteilkomponenten auf der Platine (Routing) erfolgte unter Einhaltung der allgemeinen Gesetzmäßigkeiten hinsichtlich Mindestdicken von Leiterbahnen und deren Mindestabstände untereinander zu jeweiligen Spannungsbereichen sowie allgemeingültigen Richtlinien der Platinen-Entwicklung. Da der WMZ selbst nicht potenzialfrei ist, insbesondere für den praktischen Einsatz des Anwendungsmoduls eine galvanische Trennung der Elektroniken seitens SmallCAN und seitens WMZ notwendig, damit keine Überspannungen seitens des WMZ zu Fehlfunktionen oder Zerstörungen von SmallCAN-

Komponenten führt. Ein Kleinserienauftrag eines exemplarischen Anwendungsadapters wurde durch eine externe Bestückungsfirma erfolgreich bearbeitet. Damit ist die Möglichkeit der automatisierten Bestückung in der Serienproduktion prinzipiell gegeben. Abbildung 4 zeigt die Anordnung der Bauteile auf dem Anwendungsmodul App_MBUS_wmz sowie die fertig be-stückte Platine eingebaut in dem Wärmemengenzähler CF-Echo.

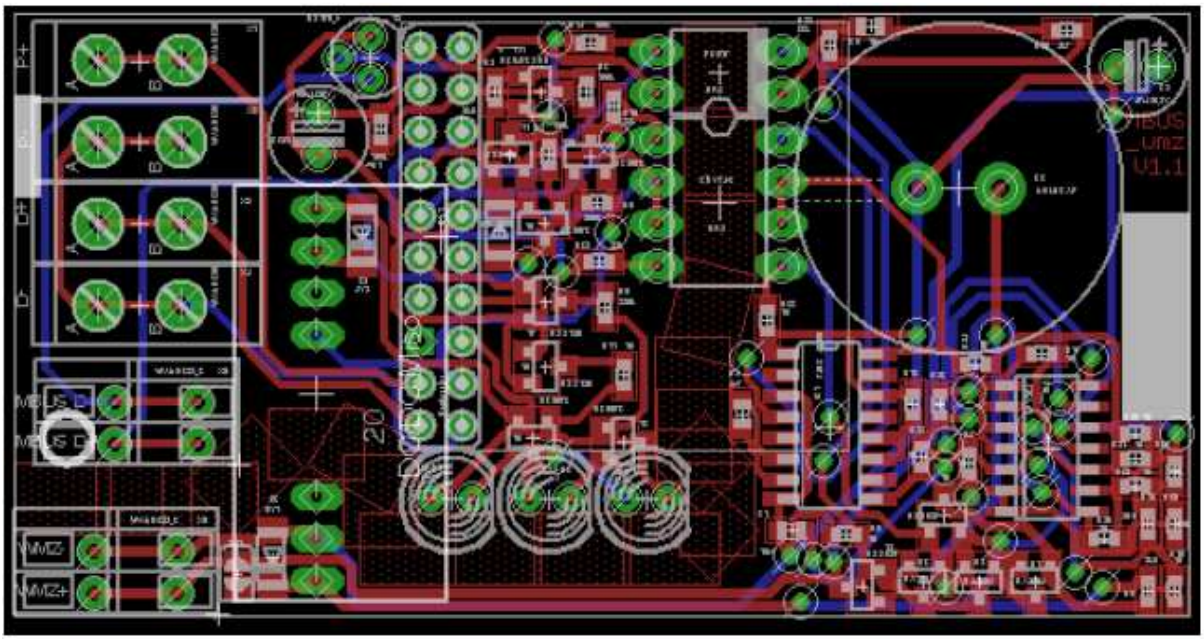


Abbildung 4: PCB „App_MBUS_wmz“.

Insgesamt sind auf der Platine „App_MBUS_wmz“ 68 Bauteile verbaut. Der vollständige Anwendungsadapter mit Buskoppler und aufgesteckten isolierten DC/DC-Wandler besteht aus 128 Bauteilen. Die bestücke und in den Wärmemengenzähler integrierte Platine ist in Abbildung 5 gezeigt. Zur Ansteuerung des Wärmemengenzählers wurde eine Firmware entwickelt, die sich auf den auf dem Buskoppler befindlichen Mikrocontroller-Chip programmieren lässt. Die Firmware fragt in regelmäßigen Abständen die in dem Rechenwerk ermittelten thermischen Größen ab und wandelt diese Daten in SmallCAN konforme Daten um. Für die Entwicklung der Firmware sind weitere Kenntnisse aus verschiedenen Schichten des ISO-OSI-Modells zu ermitteln und umzusetzen. Der zur Ansteuerung dieses M-Bus-fähigen Gerätes entwickelte Quellcode wurde in PIC-Assembler programmiert und beläuft sich auf 1685 Zeilen Code.

Wesentliche Bestandteile der Firmware ist die korrekte Anfrage an die MBUS-Schnittstelle, die Auswertung dessen Antwort, die Dekodierung der Messsignale, die Anpassung an die Datentypen die das SmallCAN-Bussystem bereitstellt sowie eine umfangreiche und notwendige Fehlererkennung.

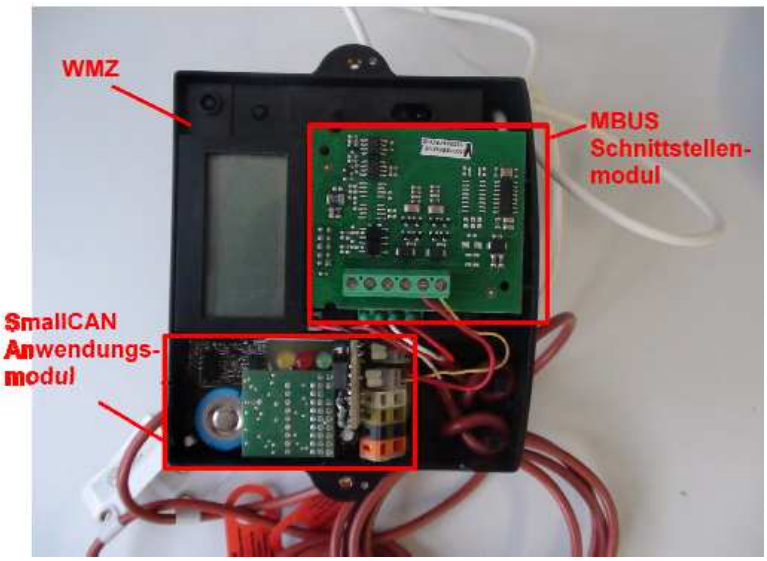


Abbildung 5: Bestückte Platine in WMZ.

Parallel zur Entwicklung eines Anwendungsadapters ist eine Dokumentation durchgeführt worden. Der Inhalt der Dokumentation eines jeden Applikationsadapters gliedert sich wie folgt:

1 Beschreibung

1.1 Grundfunktion

1.2 Hardware

1.3 Leistungsumfang

1.4 Übertragungs- und Geräteparameter

2 Parameter

3 Anzeigen

4 Startverhalten

5 Stopverhalten

6 Busverhalten

6.1 Versand

6.2 Empfang

6.3 Lastbegrenzung

6.4 Interne Werte

7 Rechenzeit

8 Protokolldefinition

9 Ansteuerung der Hardware

10 Ressourcen und Softwareschnittstellen

11 Aufbau der Software

12 Schaltplan

13 Gehäuse

13.1 Frontplattenbeschriftung

14 Bedienung der PC-Software

Zunächst wird die generelle Funktion des Applikationsmoduls beschrieben:

Über MBUS lassen sich Daten des Wärmemengenzählers (WMZ) CF-ECHO II der Firma Allmess (ACTARIS) auslesen und zusätzlich wird der WMZ mit Spannung versorgt.

In einer Kurzbeschreibung wird eine Tabelle ausgefüllt, in der u.a. hinterlegt wird, welche Bibliotheken für die Software genutzt wurden, welche EEPROM -Adressen reserviert wurden, welche Flags verwendet werden und wie die Pins des Buskopplers belegt werden. In der Tabelle sind weiterhin die Sende- und Empfangsnachrichten des Anwendungsmoduls hinterlegt.

Es folgt eine Beschreibung der Hardware:

Ein zum Buskoppler in Reihe geschalteter DCDC_miso versorgt die Sekundärseite mit ca. 5V. Die Sekundärseite wiederum versorgt den WMZ mit Spannung und ladet zugleich einen GoldCap-Kondensator (GC). Sollte es zu einem Spannungsausfall kommen, versorgt der GC den WMZ. Während der GC geladen wird, kann dessen Spannungswert mittels Polling von INT_SF1_0 ausgelesen werden. Des Weiteren kann vom Buskoppler mithilfe von Optokopplern das Laden des GC mit und ohne Vorwiderstandes eingestellt, sowie die WMZ Spannungsversorgung ausgeschaltet werden. Parallel zum Buskoppler und DCDC_miso wird eine MBUS- Daten Sende- und Empfangsperipherie versorgt.

Die Beschreibung des Leistungsumfanges für das Anwendungsmodul ist wie folgt beschrieben:

Der GC bietet eine 2,5 Stündige Spannungsversorgung des WMZ beim Stromausfall. Aktuali- sierung der WMZ- Messwerte in ca. 10 Sekunden. Diese 10s setzen sich zusammen aus 1s Checksumme ermitteln, 5s Messwerte umrechnen, 1s Werte zusammenfassen, 2s senden aus SOND1 und SOND2 und 1s um neue Messwerte anfordern. Bei Übermittlungs- fehlern der MBUS- Daten rotes Blinken bzw. auslösen des HW- Status.

Zur Beschreibung der Übertragungs- und Geräteparameter ist die Abbildung 6 in die Do- kumentation aufgenommen worden. Sie stellt die gültigen Buspegel für den MBUS dar.

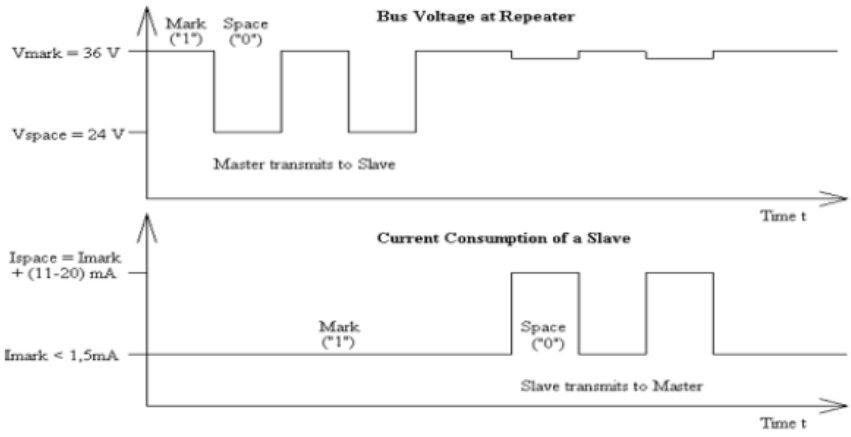


Abbildung 6: Strom und Spannungsverhalten bei der M-BUS-Kommunikation. [Quelle: www.m-bus.de]

Hinsichtlich der Parameter sind folgende Anmerkungen dokumentiert worden:

In der EEPROM- Speicherstelle 128 wird der Spannungsoffset, der bei der ersten Inbetrieb- nahme gemessen wurde, eingetragen.

In der EEPROM- Speicherstelle 129 wird der Spannungsanstiegsfaktor, der bei der ersten Inbetriebnahme angepasst wurde, eingetragen.

Es besteht ein Compiler-Schalter, mithilfe dessen die Überprüfung der Checksumme ein- bzw. ausgeschaltet werden kann.

Bei SF1 wird eine durch 4 teilbare Adresse bzw. eine Adresse, die als Binärzahl mit 11 en- det, (z.B. 91 = 1011011) gewählt. Bei SF2 muss die Adresse nach unten 3 Adressen als Platzhalter frei haben, da Telegramme über 3 Subadressen versendet werden.

Folgende Anzeigen sind für den Anwendungsadapter realisiert worden:

Die LED grün2 blinkt beim Anfragen neuer Daten vom WMZ. Die rote LED blinkt bei Emp- fangsfehlern im MBUS- Empfang.

Das Verhalten beim Starten und Stoppen des Buskopplers ist in der Dokumentation ebenfalls beschrieben:
Beim Start werden sofort die Messwerte des WMZ ausgelesen. Bestehen einige Messwerte nicht, d.h. gibt der WMZ einen Fehlerfall bei diesen aus, so werden diese Messwerte als 0 initialisiert.

Beim Buskopplerstop wird der GC weiter geladen und der WMZ weiter mit Spannung versorgt. Wird der DCDC_miso entfernt, so wird der WMZ noch ca. 2,5 Stunden vom GC Spannungsversorgt. Tabelle 1 zeigt die Messdaten, die das Anwendungsmodul für den Wärmemengenzähler versendet.

Versand ID	Sub ID	Größe	Auflösung	Bus-Datentyp
SF1	0	Energiewert [kWh]	0,001 kWh	ZAHL48
SF2	0	Leistung [W th]	10 W	MESS
SF2	1	Durchfluss [L/s]	1 L/s	MESS
SF2	2	Temp Rücklauf [°C]	0,1 °C	MESS
SF2	3	Temp Vorlauf [°C]	0,1 °C	MESS

Tabelle 1: Nachrichtenversand des Anwendungsmoduls für den WMZ.

Tabelle 2 zeigt interne Messdaten des Wärmemengenzählers, die ggf. abgefragt werden können.

Versand ID	Wertwert	Auflösung	Bitbreite
INT_SF1_0	Spannung am Gold-Cap	0,01 V	12 Bit
INT-SF2_0	Unbenutzt		

Tabelle 2: Interne Messwerte des Anwendungsmoduls für den WMZ.

Das Kapitel 9 der Dokumentation befasst sich mit der genutzten Protokolldefinition. Im Fall des M-BUS Protokolls ist der Datenrahmen für die Anfrage an den WMZ durch das Anwendungsmodul wie folgt definiert:

Start	Steuerfeld	Adressfeld	Prüfsumme	Ende
10h	4Bh	FEh	49h	16h

Der Antwort-Datenrahmen vom WMZ weist folgende Struktur auf:

1	2	3	4	5	6	7-27	28-31			
Start	Länge	Länge	Start	Steuer-feld	Adress-feld	...	**Energiewert**			
68h	4D /5D /54 /64h		68h	08h	NNh	...	I1	I2	I3	I4

32-39	40-42			43-44	45-47			48-49	50-51		52-53
...	**Leistung**			...	**Durchfluss**			...	**Vorlauf T**		...
...	V1	V2	V3	...	V1	V2	V3	...	V1	V2	...

54-55		56-81	82	83
Rücklauf T		...	Prüfsumme	Ende
V1	V2	...	XXh	16h

Dabei steht V1 für die niederwertigsten 2 BCD-Zahlen und die nachfolgenden V2, V3, V4 sind die höherwertigen BCD-Zahlen. Der einzige Wert, der als Integer kodiert ist, ist der Energie- wert. Dabei steht I1 für die niederwertigste Integer-Zahl und die nachfolgenden I2, I3, I4 sind die höherwertigen Integer-Zahlen.

Zur Ansteuerung der Hardware ist folgendes dokumentiert:

Das Senden über MBUS erfolgt mit Spannungspegeln im Bereich von 19-28V. Diese werden mithilfe der Verschaltung von R4, T1, T4, T9 und Z1 erzeugt.

Zum Senden werden am Pin B5 die Transistoren T1 und T9 zum angesteuert. Bei einem High-Signal werden T1 und T9 leitend und T9 überbrückt die Z-Diode Z1. Nun liegt am Ausgang MBUS_DATA+ eine Spannung von 28 V (MBUS High-Signal) an.

Bei einem Low-Signal am Pin B5 sind die Transistoren gesperrt und über die Diode Z1 fallen 9V ab, damit liegt am MBUS_DATA+ eine Spannung von 19 V (MBUS Low-Signal) an.

Das Empfangen erfolgt mit Strompegeln im Bereich von 0-1,5mA. Diese werden mithilfe der Verschaltung von R1, R2, R3, R5, T10 und T11 erzeugt.

Zum Empfangen wird die Spannung am Transistor T11 mit Pin B2 gemessen. Wenn ein Strom > 1,5mA zwischen MBUS_DATA+ und MBUS_DATA- fließt, fallen über R5 0,45V ab

und T10 wird leitend. Daraufhin wird T11 leitend und am Pin B2 liegt ein Low-Signal (MBUS Low-Signal) an. Fließt kein Strom zwischen MBUS_DATA+ und MBUS_DATA-, sperren T10 und T11 und somit liegt am Pin B2 ein High-Signal (MBUS High-Signal) an.

Zusätzlich kann mithilfe von T4 am Pin B3 der MBUS_DATA- hochohmig geschaltet werden. Damit fließt zwischen MBUS_DATA+ und MBUS_DATA- kein Strom, was das Stromsparen ermöglicht.

Auslesen der GC-Spannung:

Der Spannungswert des GC wird mithilfe einer OP Verschaltung aus Werten 0-4,7V in Spannungswerte 1,2 – 4V umwandelt. Diese Spannungswerte werden mithilfe eines VCO IC1 (Volt zu Frequenzwandler) über einen Optokoppler OK2 übertragen und am Pin B0 gemessen.

Das Auslesen erfolgt über Polling von INT_SF1_0. Spannungen im Bereich von 0-4,7V werden in Zahlenwerten von 0 bis 470 dargestellt. Die Abbildung 7 der Spannung auf den Zahlenwert erfolgt nicht genau, es entstehen leichte Abweichungen.

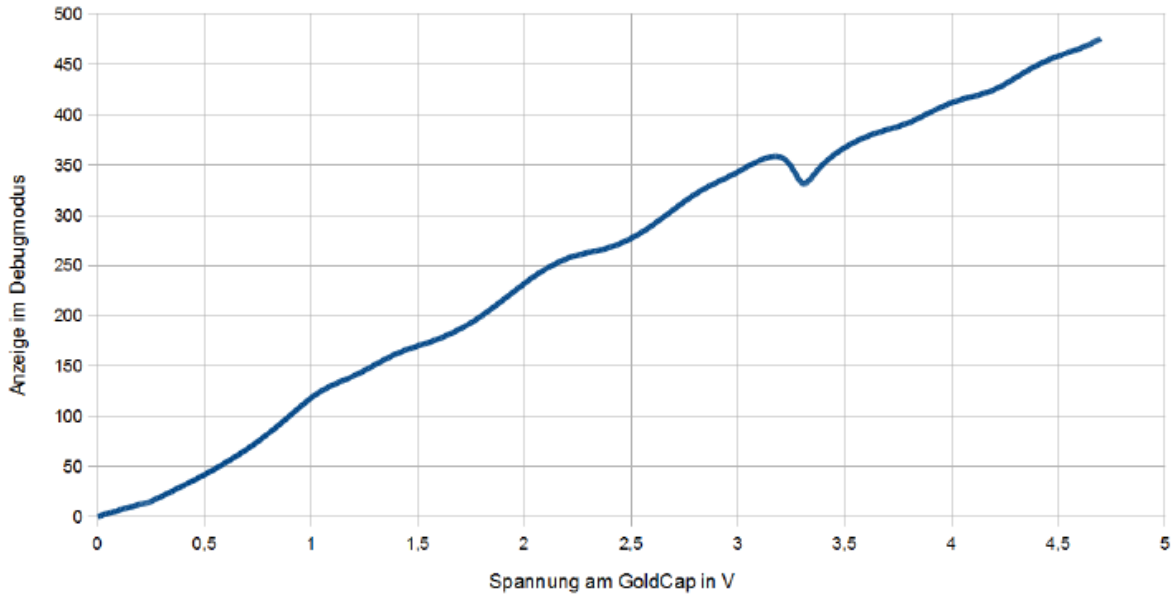


Abbildung 7: Spannungsmessung am Goldcap

Der kleine Einknick entsteht durch die Umschaltung des GoldCaps zwischen laden mit Vorwiderstand und laden ohne Vorwiderstand. Ab einer Spannung von 3,35V wird ohne Vorwiderstand geladen.

Der Sekundärstromkreis wird vom DCDC_miso versorgt. Durch die Z-Diode Z2 stellt sich eine Spannung von ca. 4,8V ein.

Am Pin B1 wird der Optokoppler OK3B angesteuert, der den Transistor T5 leitend bzw. sperrend schaltet. Ist T5 leitend, so wird der GC ohne Vorwiderstand und wenn T5 sperrt, wird der GC mit Vorwiderstand geladen. Am Pin B4 wird der Optokoppler OK3A angesteuert, der den Transistor T3 leitend bzw. sperrend schaltet. Ist T3 leitend, so wird der WMZ mit Spannung versorgt und wenn T3 sperrt, liegt am WMZ keine Spannung an. Die Dioden D2 und D3 sorgen dafür, dass der WMZ entweder über den DCDC_miso oder bei einem Spannungsaus- fall vom GC versorgt wird.

Zur Beschreibung der Firmware soll hier nur auf wesentliche Elemente eingegangen werden. Tabelle 3 gibt zunächst alle für die Firmware definierten Flags an:

FL_RXCOMPL_0	Zeigt ob ein Empfang auf dem MP Bus noch nicht abgeschlossen bzw. evaluiert wurde.
FL_TXCOMPL_SEND1_0	Zeigt ob die Ausgewertenten Daten noch nicht abgesendet wurden von SOND1.
FL_TXCOMPL_SEND2_0	Zeigt ob die Ausgewertenten Daten noch nicht abgesendet wurden von SOND2.
FL_CHECK_PNT_0	Flag zum prüfen des Empfangs-Pointers.
FL_NEW_VALUE_0	Neuer Frequenz-Wert vom VCO eingegangen.
FL_GC_WMZ_0	Auswahl ob GC-Spannung oder WMZ-Werte berechnet und gesendet werden sollten.
FL_CALCCOMPL_0	Zeigt ob Berechnung der empfangenen Werte abgeschlossen ist.
FL_SENDCOMPL_0	Zeigt ob neu Ermittelten Werte raus gesendet wurden.
FL_NEW_POWER_1	Flag, dass neue Leistung gemessen wurde.
FL_NEW_FLOW_1	Flag, dass neuer Durchfluss gemessen wurde.
FL_NEW_TEMPRUECK_1	Flag, dass neue Temperatur im Rücklauf gemessen wurde.
FL_NEW_TEMPVOR_1	Flag, dass neue Temperatur im Vorlauf gemessen wurde.
FL_MEAS_1	Flag zum starten der Messung der Pulse.
FL_MEAS_OLD_1	Alter Wert von PIN B0
PIN_IN_OLD_1	Alter Wert von PIN B0.
FL_NEW_ENERGY_1	Flag, dass neuer Energiewert gemessen wurde.
FL_SEND2_SF20	Zeigt an, ob Sub-ID 0 gesendet wurde.
FL_SEND2_SF21	Zeigt an, ob Sub-ID 1 gesendet wurde.
FL_SEND2_SF22	Zeigt an, ob Sub-ID 2 gesendet wurde.
FL_SEND2_SF23	Zeigt an, ob Sub-ID 3 gesendet wurde.
FL_CHECK0_2	Zeigt an, ob Checksumme berechnet wurde.
FL_CALC1_2	Zeigt an, ob der Wert für Temperatur-Rücklauf berechnet wurde.
FL_CALC2_2	Zeigt an, ob der Wert für Temperatur-Vorlauf berechnet wurde.
FL_CALC3_2	Zeigt an, ob der Wert für Energiewert berechnet wurde.
FL_CALC4_2	Zeigt an, ob der Wert für Durchfluss berechnet wurde.
FL_CALC5_2	Zeigt an, ob der Wert für Leistung berechnet wurde.

Tabelle 3: verwendete Flags für die WMZ-Firmware. 19

Exemplarisch zeigt Abbildung 8 das Flussdiagramm zur Berechnung der Daten aus dem Antwort-Datenrahmen des WMZ nachdem die Checksumme erfolgreich geprüft wurde.

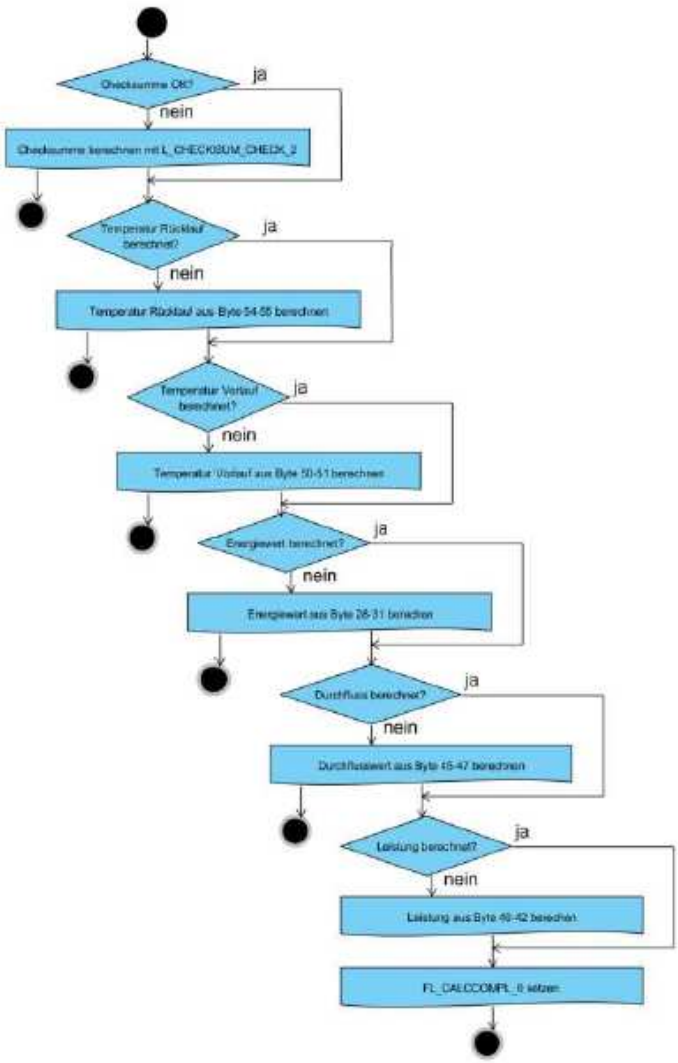


Abbildung 8: Flussdiagramm zur Berechnung der WMZ-Werte.

Entsprechende Dokumentationen wurden für alle Anwendungsmodule angefertigt.

2.1.2 Firmwareentwicklung

Bezugnehmend auf die Zielsetzung des Projektes der Energieersparung sowie der Bereitstellung neuer Funktionalitäten in der Gebäudeautomation sind intelligente Regelungsfunktionen (verarbeitende Funktionen) erstellt worden, um die verschiedenen Anwendungsmodule sinnvoll interagieren zu lassen. Die verarbeitenden Funktionen werden in der Programmiersprache C geschrieben und als zusätzliches Modul zu der entwickelten Firmware auf den auf den Mikrocontroller geschrieben. Die Anwendung der verarbeitenden Funktionen ist auch im Demonstrationsmodell „future workspace“ im vollen Umfang erfolgt und konnte die funktionellen Vorgaben auch in der Praxis erfüllen. Hierzu siehe Anhang A1.

2.2 Ausgangslage Busserver/GUI

Die Anbindung des SmallCAN-Busses an die PC-Welt geschieht über die serielle Schnittstelle wie in Abbildung 9 dargestellt.

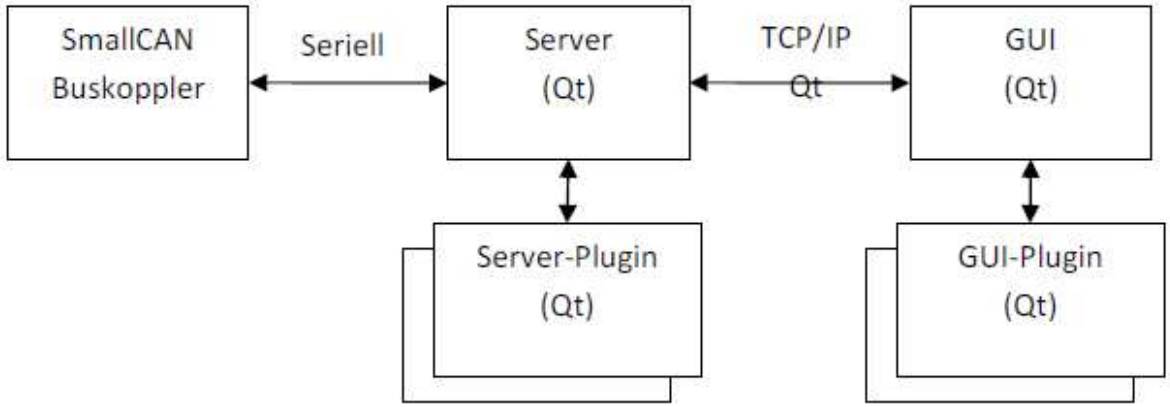


Abbildung 9: Ausgangsarchitektur der SmallCAN-Anbindung.

In der Übertragungskette erfüllen die einzelnen Blöcke folgende Rollen:

- SmallCAN-Buskoppler sendet Bustelegramme per RS232 zum PC und vom PC auf dem Bus. Um den Sicherheits-Integritätslevel drei (SIL3) zu erreichen, werden alle Telegramme sowohl vom Buskoppler zum PC als auch in der Gegenrichtung einzeln mit einer Checksumme quittiert. Der Buskoppler hält eine Warteschlange von 16 Telegrammen bereit, weil der Serverprozess von einem nicht echtzeitfähigen Betriebssystem (Windows, Linux, Mac iOS) zu variablen aufgerufen wird.
- Der Server verwaltet Systemabbild (den Zustand aller Buskoppler auf dem SmallCAN) und den Benutzerabbild (aktueller Wert aller Busvariablen). Außerdem leitet der Server alle Bustelegramme von der seriellen Schnittstelle per TCP zur GUI und zurück.
- GUI stellt mehrere Tabellen dar: aktuelle Telegramme auf dem Bus, Systemabbild, Benutzerabbild, Busstatistik, graphische Darstellung des Benutzerabbildes mit der Möglichkeit, die Busobjekte zu verändern. Weiterhin bietet GUI eine Plugin-Schnittstelle, um mit Hilfe von Plugins die einzelnen Funktionen auf den Buskopplern zu parametrieren.

Ein Überblick über die Komplexität des Codes gibt sein Umfang in Zeilen:

- 1000 Zeilen PIC-Assembler im anbindenden SmallCAN-Buskoppler

- 7300 Zeilen C++ gemeinsame Bibliothek von Server und GUI
- 5300 Zeilen C++ im Server
- 25000 Zeilen C++ im GUI
- 35000 Zeilen C++ in den Plugins zur Konfiguration der Funktionen

Im Laufe des vorliegenden Projektes wurde eine neue Implementierung entwickelt, die folgende Kennzahlen aufweist:

- 1200 Zeilen PIC-Assembler im anbindenden SmallCAN-Buskoppler
- 23000 Zeilen C++ Server
- 3600 Zeilen XML-Beschreibung der Funktionen
- 7500 Zeilen C++ GUI

Um die Notwendigkeit einer Neustrukturierung der Softwarearchitektur abzuleiten werden im Weiteren Vor- und Nachteile der aktuellen Architektur analysiert.

2.2.1 Vor- und Nachteile der aktuellen Architektur

Zu den Vorteilen zählen:

- Einfache Architektur
- Hohe Erweiterbarkeit der GUI durch Plugins

Die Nachteile:

- Das Protokoll Buskoppler <-> PC mit Quittierung jedes Telegramms führt unter Windows bereits bei geringer Last zum Abriss des Datenstroms, weil der FIFO im Buskoppler überläuft.
- Das Protokoll Server <-> GUI basiert auf Datenstrukturen von C++-Bibliothek Qt. Es lassen sich nur Qt-Clients anbinden.
- Der GUI-Entwickler muss jede Feinheit des SmallCAN-Systems verstehen, weil der Server nur zur Datenweiterleitung und „-protokollierung“ verwendet wird.
- Der SmallCAN-Installateur muss sich mit allen Bus-Datentypen, Speicheraufbau der Buskoppler und den einzelnen Funktionen auskennen, um diese zu installieren und miteinander zu verknüpfen.
- Die Bus-Adressen agieren als IDs der einzelnen Busobjekte (Busvariablen). Es gibt ei- ne feste Zuordnung dieser Adressen zu den GUI-Elementen. Die Busadressen können aber verändert werden, um das Laufzeitverhalten der einzelnen Komponen-

ten zu verbessern. In diesem Fall müssen die Veränderungen in mehreren GUI-Arten nachgezogen werden.

- Bei einem industriellen Einsatz gibt es z. B. den BACnet-Standard, nachdem die einzelnen Komponenten in ein Gesamtnetz integriert werden. Z. B. die TU Braunschweig verwaltet über 80 Gebäude mit verschiedenen Installationen und Herstellern mit Hilfe nur einer graphischen Benutzeroberfläche, die alle Komponenten per BACnet einbindet. Der BACnet-Standard verlangt feste, unveränderbare IDs für die einzelnen Objekte. Diese können nicht auf veränderbaren SmallCAN-Telegramm-IDs basieren.
- Der kommerzielle Erfolg des SmallCAN-Busses bei Privatanwendern hängt vor allem von der graphischen Benutzeroberfläche ab. Hier sind mehrere Möglichkeiten denkbar sowohl bei der Hardware (Touchscreen, Smartphone, PC, Pad), Darstellungsarten (2D, Animationen, 3D, detailliert/einfach), Plattform (Internetbasiert im Browser, fest installierter Client). Der Aufwand für die Entwicklung einer neuen graphischen Benutzeroberfläche ist wegen des Verhältnisses zehnmal aufwendiger als der gemeinsame Teil (Server).
- Der Hardware/Software-Entwickler, der einzelne Busapplikationen entwickelt, muss auch für einfache Parameter ein Plugin mit Qt in C++ programmieren. Sollten mehrere verschiedene graphischen Benutzeroberflächen benutzt werden, muss auch der Entwickler für diese neue Plugins programmieren.
- Änderung der Plugin-Schnittstelle erfordert eine Anpassung im Großteil des Codes in den Plugins, die von verschiedenen Entwicklern stammen.
- Begrenzte Testbarkeit des Codes wegen der engen GUI-Kopplung.
- Es gibt keine Kontrolle der Konfiguration, der Installateur/Benutzer ist verantwortlich für

    - kollisionsfreie Adressenvergabe
    - Einhaltung der Datentypen auf dem Bus
    - Zuordnung der Telegramme an die richtige Funktionen und richtige Eingänge

  Um diese Aufgaben wahrzunehmen, muss der Installateur den zukünftigen Busverkehr abschätzen und jede eingesetzte Funktion kennen (jeweils 30 Seiten Dokumentation).

Ziel der vorhandenen Software war die Überprüfung der Machbarkeit mit minimalem Aufwand. Dies spiegelt sich auch darin wider, dass es keine Testfälle existieren: sobald eine Teilfunktion einmal läuft, wird die nächste implementiert. Eine Codeänderung ist unter diesen Umständen mit hohen Risiken verbunden.

2.2.2 Anforderungen an die neue Server-Client-Architektur

Die Anforderungen haben sich im Laufe des Projektes herauskristallisiert (z.B. FutureWorkspace). Der Hauptunterschied zur vorhandenen Architektur liegt in der Konzentration der Funktionalität im Server und Verwendung einer schlanken graphischen Benutzeroberfläche. Damit werden die Vor- und Nachteile der bisherigen Software vertauscht: die Serverarchitektur wird komplex, dafür ist die Entwicklung neuer Busfunktionen und Oberflächen einfach. Im Weiteren werden die Rahmenbedingungen für den Busserver analysiert.

2.2.2.1 Server-Hardware

Eines der wichtigsten Merkmale des SmallCAN-Bussystems ist sein geringer Stromverbrauch. Um dieses Ziel auch mit dem Busserver zu erfüllen, wurde ein sparsamer MiniPC basierend auf ARM9-Board von Hectronic (www.hectronic.se) entwickelt, mit einem Stromverbrauch von 1.5 Watt. Dieser läuft mit 180 MHz ARM9-Prozessor unter Linux und beinhaltet 32 MB RAM. Eine andere Zielhardware stellt ein Windows-PC dar, wobei diese meist um mehrere Größenordnungen leistungsfähiger ist, als das ARM9-Board. Eine mögliche Zielhardware könnte auch z. B. ein Fritzbox-Modem sein, der bei einer DSL-Anbindung sowieso immer aktiv ist.

Hauptbegrenzungsfaktor ist in diesem Zusammenhang der Speicher (RAM). Auf dem MiniPC ließe sich auch eine Webserver-basierte graphische Benutzeroberfläche starten (z. B. http://www.webtoolkit.eu/wt), so dass der SmallCAN von einem beliebigen Ort ohne Instal- lationsaufwand überwacht und gesteuert werden kann. Der Webserver würde ca. 20MB RAM beanspruchen. Für den Busserver würde 2-4MB für die Verwaltung von 1000 Buskopp- lern und bis zu 30.000 Busobjekte übrig bleiben.

2.2.2.2 Betriebssystem

Der Server soll sowohl unter Linux, als auch unter Windows lauffähig sein. Diese Betriebssysteme unterscheiden sich vor allem in der Behandlung der seriellen Schnittstelle. In der bisherigen Implementierung musste der Server unter Windows mit Administratorrechten mit höchster Priorität gestartet werden, um die serielle Schnittstelle größtenteils rechtzeitig zu bedienen.

Am Projektbeginn wurde die Möglichkeit untersucht, einen speziellen Treiber für Windows zu programmieren, der das Anbindungsprotokoll zum SmallCAN-Buskoppler umsetzt. Nach außen (zum Busserver) würden dann nur Telegramme und keine Bytes kommuniziert.

Parallel dazu wurde der Anbindungsprotokoll auf eine Sammelbestätigung umgestellt: statt jedes einzelne Telegramm zu bestätigen, werden bis zu 15 Telegramme auf einmal bestätigt. Damit bekommt das Betriebssystem des Servers bis zu 90 ms Zeit, um die Kontrolle an den Busserver zu übergeben. Ab Windows XP wird jeder Task alle 15 ms aufgerufen. Somit hat man auch unter hoher Belastung (z. B. Abspielen eines Videos) genug Reserven.

2.2.2.3 Anbindung der Clients (z. B. graphische Oberfläche)

Es gibt mehrere Möglichkeiten, die Kommunikation mit den Clients durchzuführen. Man muss folgende Entscheidungen treffen:

- Netzprotokoll: UDP vs. TCP
- Telegrammen: binär vs. textbasiert (z.B. XML)

Als optimal scheint in diesem Zusammenhang ein binäres Protokoll über TCP zu sein. Ein

textbasiertes Protokoll erfordert ca. 10mal größere Bandbreite, die würde

- die Anzahl der parallel laufenden Clients am MiniPC-Server einschränken,
- knappen Ressourcen des Busservers (RAM und CPU) verbrauchen,
- Verbindungskosten und Übertragungszeiten zu mobilen Clients (Smartphones) erhöhen

Es gibt mehrere Alternativen für die Kommunikation:

- Proprietäre Telegramme. Der Nachteil besteht in einem hohen Aufwand und der großen Anzahl an nötigen Implementierungen
- CORBA: generisches Standard für ein TCP basiertes binäres Protokoll, das Program- miersprachen und Betriebssystem unabhängig ist. Der Nachteil: der Hipe von CORBA liegt 10 Jahre zurück. Es liegen nur für C/C++ und Java gute Implementierungen vor. Anbindung an Smartphones wäre aufwendig (Android, iOS).
- ZeroC-Ice (http://www.zeroc.com/), ein Opensource-Nachfolger von CORBA. Bietet Implementierungen für die meisten wichtigen Programmiersprachen und Betriebssysteme.
- Apache Thrift (keine Callbacks, nur Polling möglich)

- Google-Buffers (Nur Datentransport, keine RPC, nur java, c++, python)
- BACnet speziell für Gebäudeautomatisierung entwickelt, ein UDP-basiertes, binäres Protokoll. Nachteile:

    o großer Aufwand auf der Client-Seite

    o nurrudimentärevorhandeneOpensource-Implementierungen

    o deckt nur den BACnet-Teil der Kommunikation ab.

  Vorteil: existierende (teure) industrielle Clients.

Es wurde entschieden, dass der Busserver eine ZeroC-Ice Schnittstelle bereitstellt. Daran können graphische Clients, remote-Prozesssteuerung und auch ein ZeroC-Ice <-> BACnet-Umsetzer angebunden werden. Dies vereinfacht die Implementierung des Busservers und verlagert das Management von BACnet-Verbindungen in eine spezielle Softwareeinheit.

Daraus ergeben sich folgende Aufgabenstellungen:

2.2.2.4 Serielle Busanbindung

Der erste Baustein der neuen Architektur war die serielle Kommunikation mit dem Buskoppler. Dies erforderte die Einarbeitung und Neuimplementierung der Anbindungsfunktion in Assembler auf dem Buskoppler. Hauptschwierigkeit dabei ist fehlende Möglichkeit zu debuggen. Die Assemblerfunktion muss schrittweise aufgebaut und getestet werden. Sobald eine Änderung des Algorithmus nötig ist, wird die Vorgehensweise wiederholt. Auf der Serverseite wurden 1710 Zeilen Code für die Funktionalität und 600 Zeilen für die Tests verwendet. Das relativ umfangreiche Kommunikationsprotokoll wurde in drei Klassen aufgeteilt (vgl. Abbildung 10).

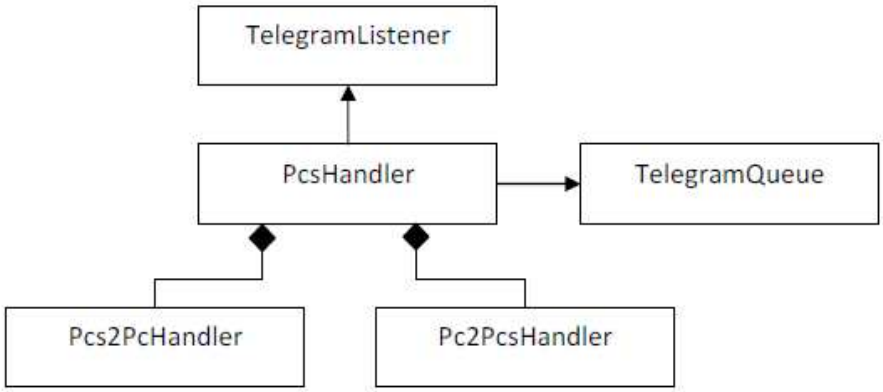


Abbildung 10: Anbindung der seriellen Schnittstelle in die Serverinfrastruktur

Sowohl die einzelnen Klassen als auch die Funktion wurden ausgiebig getestet (die Testabdeckung von ca. 90%). Dabei wurden sowohl manuelle offline Tests durchgeführt, als auch online Test. Bei den Online-Tests wurde zwischen dem Port-Treiber eine „Störungsroutine“ eingeschaltet, die jedes zwanzigstes Bit zufällig umkippte. Die Störungen wurden sowohl von Buskoppler als auch von PC simuliert. Dadurch konnte eine große Vielfalt an möglichen Fehlern offenbart werden, die weniger aus dem Protokoll, sondern viel mehr durch Timing-Problem der Implementierung entstanden sind. Z. B. muss die Implementierung „Bestätigungstelegramme“ von den „Bustelegrammen“ trennen. Falls ein „falsches“ Bit gekippt wird, muss die Implementierung raten, ob jetzt zwei kaputte „Bestätigungstelegramme“ angekommen sind und ein richtiges „Bustelegramm“, oder ein kaputtes „Bustelegramm“ und zwei normale „Bestätigungstelegramme“. Je nach Entscheidung läuft das Protokoll verschiedene Ausführungspfade. Nach fast zwei Monaten Tests wurde auch diese sehr hohe Fehlerrate korrekt abgearbeitet. In nur sehr seltenen Fällen, bei denen innerhalb von sechs Bytes zwei spezielle Bitfehler auftreten, wurde die Übertragung trotz der Checksumme unbemerkt verfälscht.

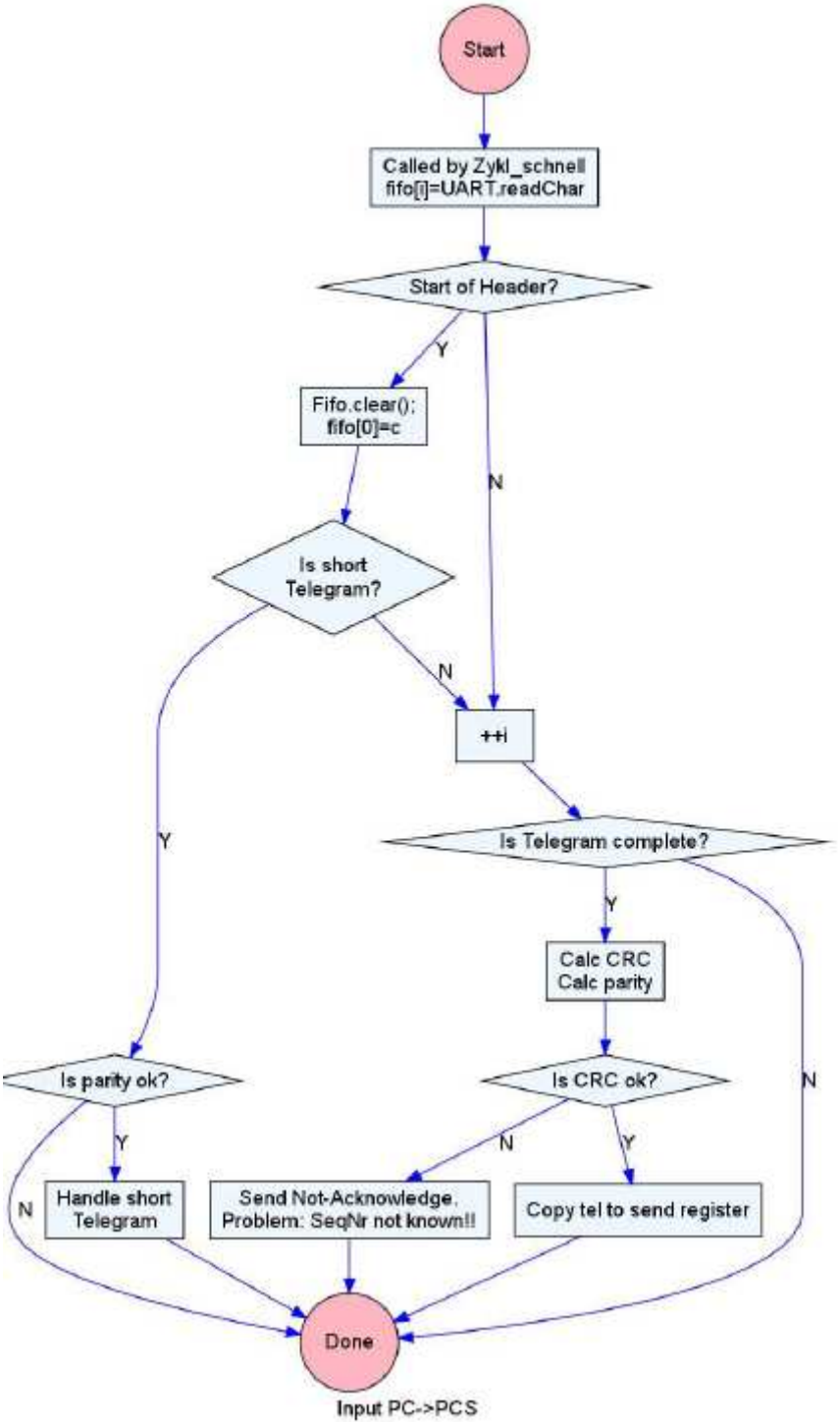


Abbildung 11: Assembler-Routine, die Telegramme von PC verarbeitet und per SmallCAN versendet

2.2.2.5 Systemdialoge

Bei der Abbildung des Zustandes von SmallCAN muss zwischen dem Systemabbild (Zustand der Hardware) und dem Benutzerabbild (Zustand der Variablen, z.B. Ventilstellungen, Messwerte etc.) unterschieden werden. Während die meisten Variablen durch ein Bustele-

gramm übertragen werden, wird beim Aufbau des Systemabbilds über zustandsbehaftete Protokolle kommuniziert.

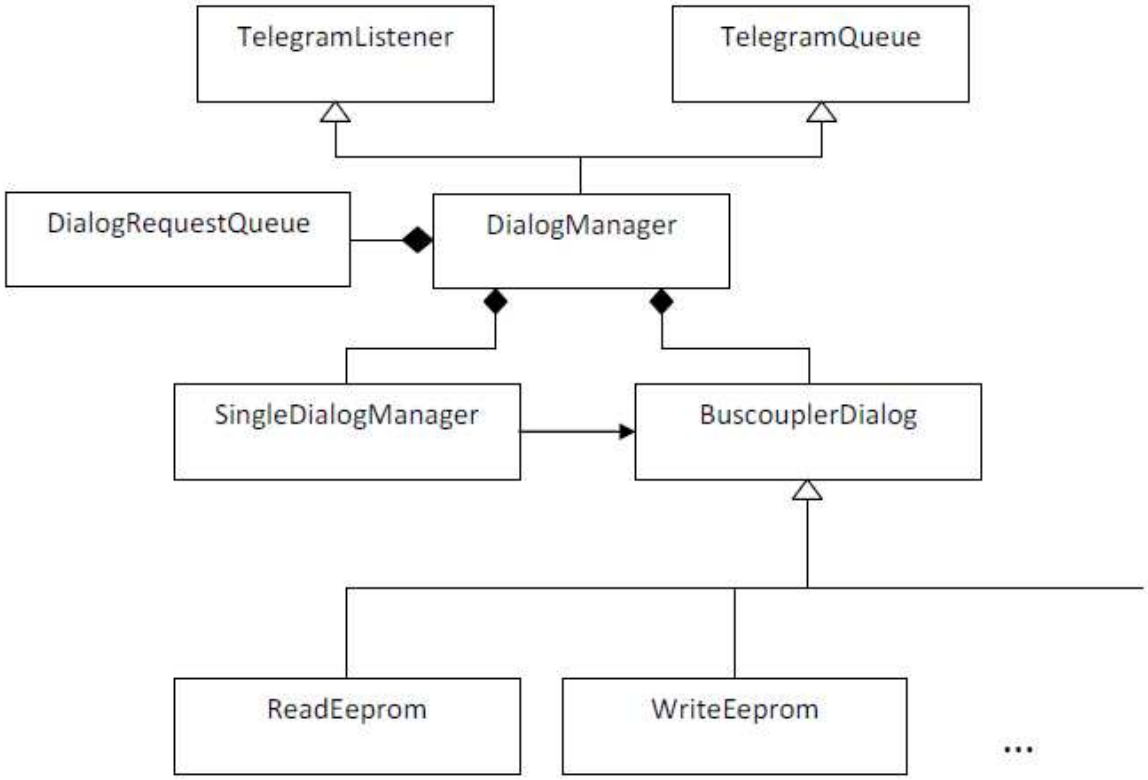


Abbildung 12: SystemDialoge-Bibliothek

Die einzelnen Klassen erfüllen bei der Kommunikation folgende Rollen:

- Die Telegrammzustellung zu den einzelnen Dialogen, Starten der neuen Dialoge verwaltet der DialogManager. Dieser beinhaltet auch einen Timer, um plötzliche Hardware-Fehler während der Dialoge aufzudecken.
- Verwaltung der Zustände, die in allen Dialogklassen identisch sind, geschieht in SingleDialogManager.
- Die meisten Buscouplerdialoge sind zustandslose Klassen, die ihren Zustand in SingleDialogManager verwalten. Ausnahmen bilden Dialoge zum Lesen des Eeproms, die nur einmal auf dem Bus stattfinden dürfen.
- Alle DialogKlassen haben noch einen „passiven“ Modus, bei dem sie den Dialogen anderer zuhören und den Zustand der Buskoppler daraus ableiten.

Der vorliegende Aufbau hat zwei wesentliche Vorteile, gegenüber vorherigen „einfacheren“ Struktur:

1. Sie verbrauchen ca. 10 mal weniger Speicher (RAM)

2. Der Speicher wird nur einmal allokiert und dann statisch verwaltet. Dies verringert die Wahrscheinlichkeit, dass nach 10-20 Jahren ununterbrochenem Betrieb der Hauptspeicher zu stark fragmentiert ist.

Sobald ein Dialog abgeschlossen ist, verändert er den Zustand einer Buskoppler-Klasse, worüber alle anderen „Buskoppler-Zuhörer“ informiert werden.

2.2.2.6 System**modell**

Bisher wurden die Funktionen auf den Buskopplern als Blackboxes betrachtet, deren Eigenschaften (Eeprom-Parameter, Sende und Empfangstelegramme) per Plugins erreichbar sind. Dies ist ein sehr flexibler Ansatz, erfordert aber den hohen Programmieraufwand für die Plugin-Entwicklung. Es stellt sich auch heraus, dass in den meisten Fällen, die Parameter und Objekte der Funktionen nach ähnlichen Verfahren auf Eeprom der Buskoppler bzw. Bustelegramme umrechenbar sind. In den seltenen Fällen, wo es komplexe, nicht allgemein gültige Algorithmen nötig sind, um z.B. aus mehreren Bustelegrammen eine komplexe Textmeldung zu berechnen, ist der Einsatz von Plugins weiterhin unumgänglich.

Bisher wurden über 50 verschiedene Funktionen für die Buskoppler entwickelt. Ein Objektmodell war nötig, um

- Ein- und Ausgänge der Funktionen in einer graphischen Benutzeroberfläche zu visualisieren,
- feste globale IDs für alle Systemobjekte konsequent zu vergeben,
- automatisch Ausgänge mit Eingängen unter Einhaltung der Datentypen und Bitposition zu verknüpfen.

Eine gut lesbare Möglichkeit, das Interface einer Funktion zu beschreiben stellt die XML-Sprache dar. Nach mehreren Anläufen wurde folgende Grundstruktur in XML-Form festgesetzt (vgl. Abbildung 13).

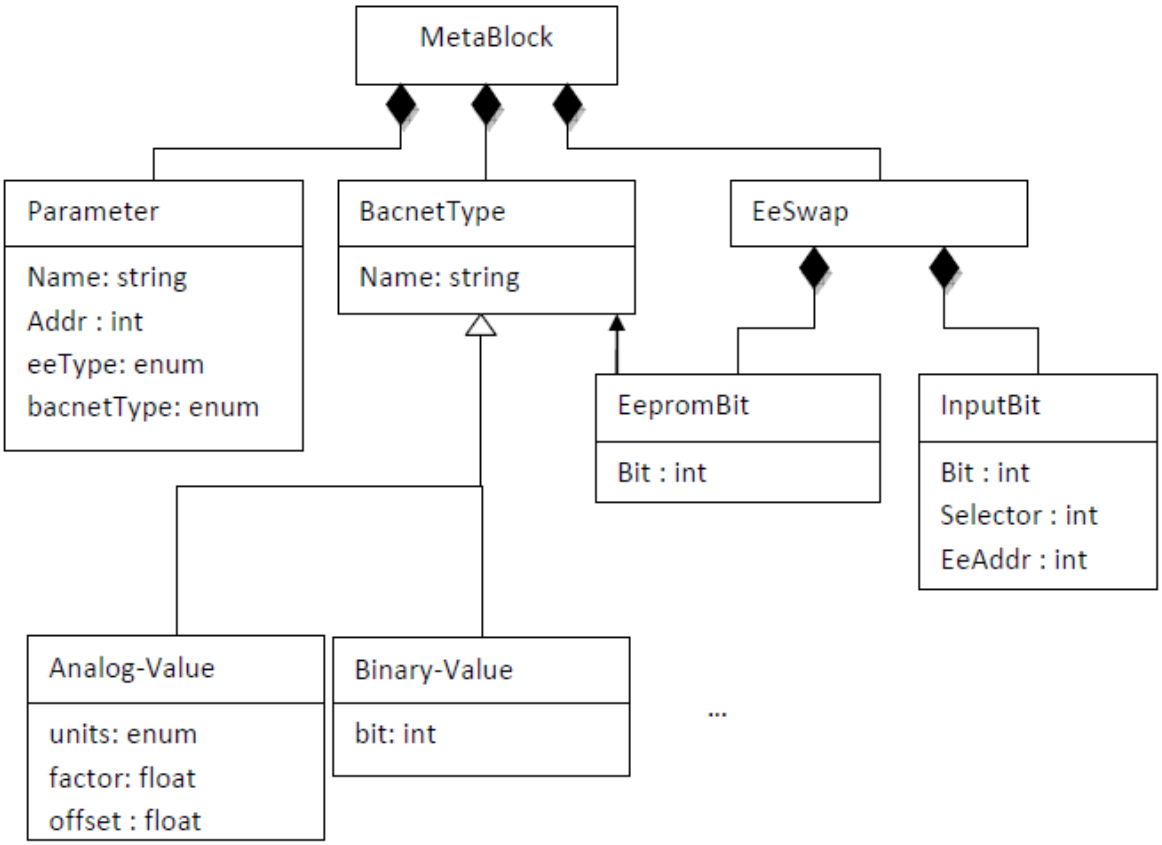


Abbildung 13: Klassendiagram der XML-Beschreibung der Buskoppler-Funktionen

Ein Beispiel für die XML-Spezifikation einer freien Sonderfunktion:

<MetaBlock ID="AUSLG" type="FSF" version="0">
<parameter name="HeizVentilModus" addr="255" eeType="uint8" bacnetType="multi- state-value"/>
<parameter name="KuehlVentilModus" addr="254" eeType="uint8" bacnet-Type="multi-state-value"/>
<parameter name="LowScaling" addr="253" eeType="uint8" bacnetType="analog-value" />
<parameter name="UpperScaling" addr="252" eeType="uint8" bacnetType="analog-value"/>
<parameter name="Skalierungsfaktor" addr="251" eeType="uint8" bacnetType="analog-value"/>
<analog-value name="HeizAnforderung" units="percent">
<input selector="6"><ScanRatio/></input>
</analog-value>
<analog-value name="Kuehlanforderung" units="percent">
<input selector="14"><ScanRatio/></input> </analog-value>
<analog-value name="Zuluft_Temperatur" units="degrees-Celsius">
<input selector="22"><ScanInt14 factor="0.1"/></input>
</analog-value>

```
        <binary-value name="Pumpe" bit="0">
        <output basisAddr="FSF1" offset="0"><ScanBinState/></output>
    </binary-value>
    <binary-value name="Lufter" bit="0">
        <output basisAddr="FSF1" offset="1"><ScanBinState/></output>
    </binary-value>
    <analog-value name="Lufter_Abluft" units="percent">
    <output basisAddr="FSF1" offset="2"><ScanRatio/></output>
    </analog-value>
    <analog-value name="Lufter_Zuluft" units="percent">
        <output basisAddr="FSF1" offset="3"><ScanRatio/></output>
    </analog-value>
    <multi-state-value name="HeizVentilModus" bitSize="2">
        <StateText>modus0</StateText>
        <StateText>modus1</StateText>
        <StateText>modus2</StateText>
        <StateText>modus3</StateText>
        <output basisAddr="FSF1" offset="4"><ScanSelection/></output>
    </multi-state-value>
    <multi-state-value name="KuehlVentilModus" bitSize="2">
        <StateText>modus0</StateText>
        <StateText>modus1</StateText>
        <StateText>modus2</StateText>
        <StateText>modus3</StateText>
        <output basisAddr="FSF1" offset="5"><ScanSelection/></output>
    </multi-state-value>
    <analog-value name="HeizVentilStellung" units="percent">
        <output basisAddr="FSF1" offset="6"><ScanRatio/></output>
    </analog-value>
    <analog-value name="KuehlVentilStellung" units="percent">
        <output basisAddr="FSF1" offset="7"><ScanRatio/></output>
    </analog-value>
</MetaBlock>
```

Die relativ aufwendige Struktur resultiert aus dem Wunsch nach möglichst großer Freiheit für den Assembler-Programmierer und noch beherrschbarer Komplexität für den Busserver. Die Rahmenbedingungen ergeben sich aus bereits existierenden Buskopplerfunktionen:

- Dem Client wird die Businfrastruktur in Form von BACnet-Objekten präsentiert. Dieser soll nicht den Busverkehr verstehen.
- Ein BACnet-Object kann auf unterschiedliche Arten auf SmallCAN-Datentypen abgebildet werden, wobei die Abbildung oft über Parameter im Eeprom im Betrieb veränderbar ist. Deswegen ist eine Zusatzschicht in Form von BacnetType nötig, die die BACnet-Objekte auf die einzelnen SmallCAN-Inputs und –Outputs abbilden.

- Es werden weitere SmallCAN-spezifische Abbildungsarten über eigene XML- Elemente spezifiziert. Kompliziert wird es noch wegen der Möglichkeit, mehrere BACnet-Objekte in verschiedenen Bustelegrammen zu kombinieren. Auf der SmallCAN-Ebene ermöglicht dies ein flexibles Prioritätenmanagement, für den Busserver wird die Komplexität der Abbildung erhöht.
- Es gibt recht generische Abbildungen, wie z. B. die Umrechnung eines 14Bit-Int-Messwertes in ein Fließkomma-Wert mit Hilfe von zwei Parametern: y = a*x +b. Andererseits gibt es sehr spezifische Abbildungen, wie z.B. die Abbildung eines SmallCAN-Telegramms auf ein Textstring, der in Eeprom recht komplex gespeichert wird. Für solche Abbildungen werden weiterhin Plugins benötigt.

Neben frei programmierbaren Funktionen, die in XML spezifiziert werden, wird ein Teil der Funktionalität von den Standardfunktionen des Betriebssystems bereitgestellt. Deren Parameter sind zum Teil in Eeproms platziert und können zur Laufzeit geändert werden. Zusammen mit den XML-basierten Beschreibungen ergibt sich folgende Architektur für das Systemmodel (Abbildung 14).

Die Unterklassen von FunctionModel erfüllen drei wichtige Funktionen:

- Sie grenzen die ID-Bereiche bei der ersten Initialisierung des Buskopplers ein.
- Sie führen Mapping von SmallCAN auf Ice-Datenstruktur und zurück.
- Sie liefern Informationen, die für das Verbindungsmanagement benötigt werden.

Insgesamt beinhaltet die Bibliothek SystemModell 28 Klassen mit 7500 Zeilen Code. Hinzu kommen 750 Zeilen für die Tests, die fast 60% des Codes abdecken.

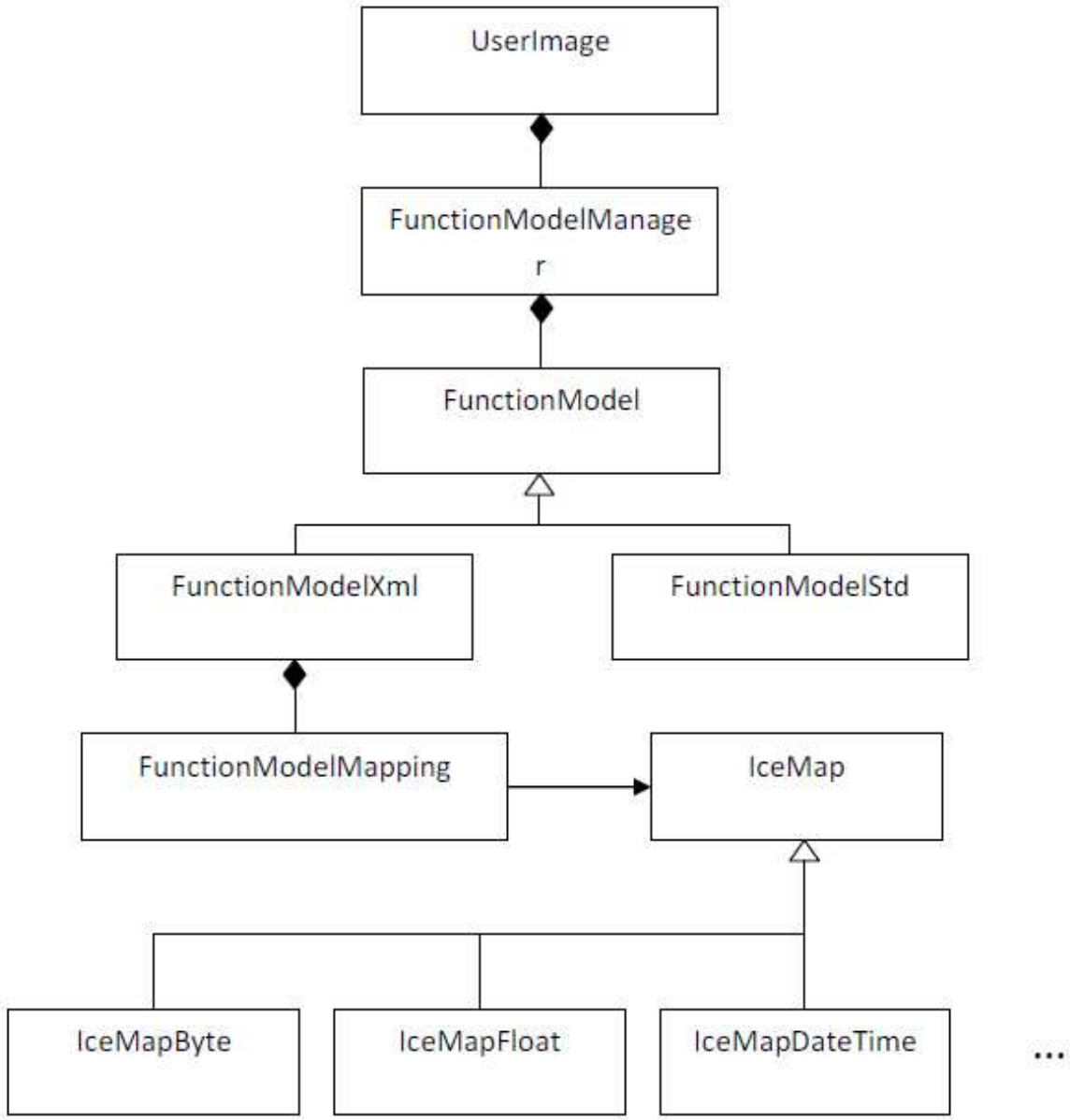


Abbildung 14: Grundstruktur des Systemmodells.

2.2.2.7 ZeroC-Ice Schnittstelle

Die Anbindung an den Busserver geschieht über die Corba-ähnliche Ice-Schnittstelle. Im Ge- gensatz zu UDP-basiertem Protokoll von BACnet, wo die Datentypen in den Telegram- men mitkodiert sind, erfordert Ice eine vorherige Festlegung auf die zu übertragenen Datenty- pen. Dies einerseits vereinfacht eine typsichere Übertragung und Umwandlung in Zielspra- chen (C++, Java, PHP, Python, .Net, etc) verkompliziert aber das Interface. Aus Performance- gründen sollte die Übertragung in möglichst großen Paketen möglichst selten erfolgen, d.h. es werden in einem Aufruf Variablen verschiedener Datentypen übertragen. Um dies zu ermöglichen wurde eine Grundstruktur definiert, die alle möglichen Datentypen in einzelne Sequenzen ablegt.

```
struct PropertyStruct {
        IntPropertyList intValues;
        FloatPropertyList floatValues;
        DoublePropertyList doubleValues;
        StringPropertyList stringValues;
        BoolPropertyList boolValues;
        ObjectRefPropertyList objValues;
        StringListPropertyList stringListValues;
        DateTimePropertyList dateTimeValues;
         BlobPropertyList blobValues;
};
```

Das Übertragungsprotokoll von Ice ist so konzipiert, dass für eine leere Sequenz nur ein Byte verwendet wird, so dass auch bei einer leeren Datenstruktur maximal neun Bytes übertragen werden. Auf der Client-Seite müssen dann die einzelnen Sequenzen iteriert werden, um die neuen Werte typsicher zu visualisieren.

Nach BACnet-Standard verfügen die einzelnen Objekte über mehrere Parameter, wobei der aktuelle Wert (z.B. die gemessene Temperatur) nur einer davon ist. Der BACnet-Standard spezifiziert alle möglichen Parameter für alle BACnet-Objekte und ordnet diesen eine Ganzzahl-ID zu.

Ein Beispiel für einen Int-Parameter:

```
        struct IntProperty {
                PropertyIdentifier id;
                int value;
        };
```

wobei der PropertyIdentifier sowohl das BACnet-Objekt als auch den Parameter spezifiziert:

```
struct PropertyIdentifier {
        int objectId;
        short propertyId;
};
```

Diese Arte der Definition wird benutzt, um mit der einzigen „PropertyStruct“ alle möglichen Parameter aller Busobjekte einschließlich der Buskoppler zu lesen und zu schreiben. Dies ermöglicht ein sehr schlankes Interface für den Busserver:

```
interface ServerObject {
void readProperties(PropertyList propList, out PropertyStruct propValues);
void writeProperties(PropertyStruct propValues);
void registerPropertyListener(PropertyList propList, Ice::Identity ident, out PropertyStruct propValues);
};
```

Der Client bekommt die Möglichkeit, die Werteänderungen sowohl per Polling als auch per Callback zu bekommen. Auf diese Weise können sowohl die Abrufintervalle als auch der Umfang der übertragenen Informationen ja nach Verbindungstyp (UMTS, DSL, Ethernet) und je nach dargestellter Information vom Client bestimmt werden.

Für die Verwaltung der aktuellen Werte der Busvariablen wir die gleiche Struktur verwendet. Dies vereinfacht den Versand von Ice-Nachrichten, weil keine dynamische Speicherallokation benötigt wird.

2.2.2.8 ID-Vergabe

Basis für die Kommunikation mit den Clients stellen unveränderliche IDs der einzelnen Busobjekte dar. Da das Bussystem einer Änderung unterworfen ist: neue Buskoppler kommen hinzu, Funktionen auf den Buskopplern werden durch neue Versionen ausgetauscht, bzw. durch völlig neue Funktionen ersetzt. Deswegen ist die ID-Vergabe eine nicht triviale Aufgabe.

Im BACnet-Standard werden 32-bittige IDs aus dem ObjectTyp und ObjectNummer gebildet. Bei dem Busserver werden die IDs anders vergeben. Es werden ID-Bereiche definiert, die für bestimmte Aufgaben genutzt werden:

1 – Hauptkontainer für alle anderen Buselemente, DEVICE nach BACnet-Definition.

2-50 reserviert

50 – 1050 – Buskoppler

ab 1100 werden die IDs linear vergeben nach dem Muster:

Funktion

Parameter 1 .. N

Sicherheitsabstand für die spätere Versionierung

Busobjekte

Sicherheitsabstand für die spätere Versionierung

Danach folgt die nächste Funktion. Auf diese Weise lassen sich die Parameter und die Busobjekte schnell den Funktionen zuordnen (die Suche läuft logarithmisch), die ihrerseits die Umwandlung von Ice-Werten in Bustelegramme mit Hilfe der IceMap-Klassen durchführen.

2.2.2.9 User-Abbild

Aufgabe des User-Abbilds ist die Bereitstellung der aktuellen Werte der Busobjekte und die Weiterleitung deren Änderungen an die Clients. Dabei dient die Klasse Telegram2Value der Umwandlung der Bustelegramme in Union-basierte Werte (Abbildung 15). Dies ist nötig, weil manche Werte über mehrere Telegramme verteilt sind, z.B. DateTime oder 48bit- Integer über 4 Telegramme.

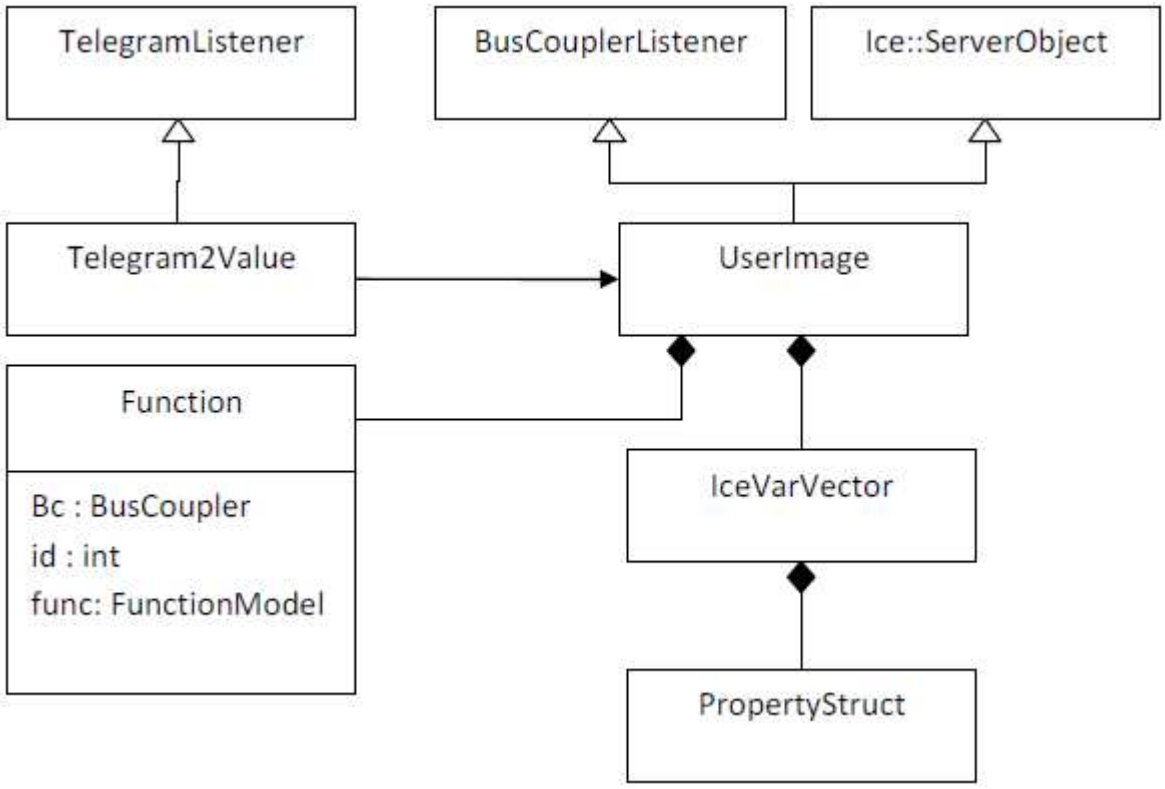


Abbildung 15: Einbindung von UserImage in die Serverarchitektur

IceVarVector bietet ein Interface zu PropertyStruct an, über den sowohl die aktuellen Werte gesetzt als auch die geänderten Werte bis zum Versand aufbewahrt werden. Die Klasse Funktion speichert die BACnet-ID der Funktion, basierend auf der alle Unterobjekte der Funktion über die FunctionModel erreichbar sind.

2.2.2.10 Verbindungsmanagement

Das Verbindungsmanagement erfüllt mehrere Aufgaben:

- Für eine Quellverbindung findet es mögliche Zielports im Sinne der BACnet-IDs. Für eine gewünschte Verbindung
    - o Findet es passende Versandtelegramme
    - o FindetespassendePortsderZielfunktion
    - o Führt es falls möglich eine Adaptation der Bitpositionen durch, z.B. ein Schalter auf dem Bit2 wird mit einem Relais auf dem Bit 4 verbunden.
    - o beauftragt es ein WriteEeprom-Dialog mit der Änderung der Empfangsliste des Zielbuskopplers.

- Für das Löschen einer vorhandenen Verbindung
    - findet espassenden Eintrag in der Empfangsliste des Zielbuskopplers
    - führt es einen „Dialog“ mit dem Client, ob alle anderen Verbindungen des Eintrags auch gelöscht werden sollen.

Hauptschwierigkeit beim Verbindungsmanagement ist, dass bis zu acht binäre Werte in einem Telegramm übertragen werden. Nicht in allen Fällen lassen sich also einzelne binäre Werte voneinander unabhängig verbinden.

2.2.2.11 Visualisierung

Ziel der Neustrukturierung des Busservers war die Vereinfachung einer neuen Visualisierung. Dies ist unter anderem dadurch erreicht, dass die Clients von der Busbehandlung abstrahiert werden und nur mit Standard-Datentypen agieren. Weitere Vereinfachungen betreffen unveränderbare IDs der Buselemente, die mit den graphischen Visualisierungselementen somit fest verknüpft werden können.

Aktuell werden zwei Visualisierungsarten weiter entwickelt:

- Ein editierbarer Baum, indem alle Buskoppler, alle Funktionen, alle Parameter und alle Busvariablen vorhanden sind. Die aktuellen Werte werden dargestellt und veränderbare Werte lassen sich editieren, per Bus übertragen und falls erfolgreich auch in dem Baum darstellen (vgl. Abbildung 16).
- Eine QML-basierte rein graphische Darstellung, die sowohl für mausbedienbare PCs als auch auf mobilen Plattformen mit Touchscreen eingesetzt wird (vgl. Abbildung 17).

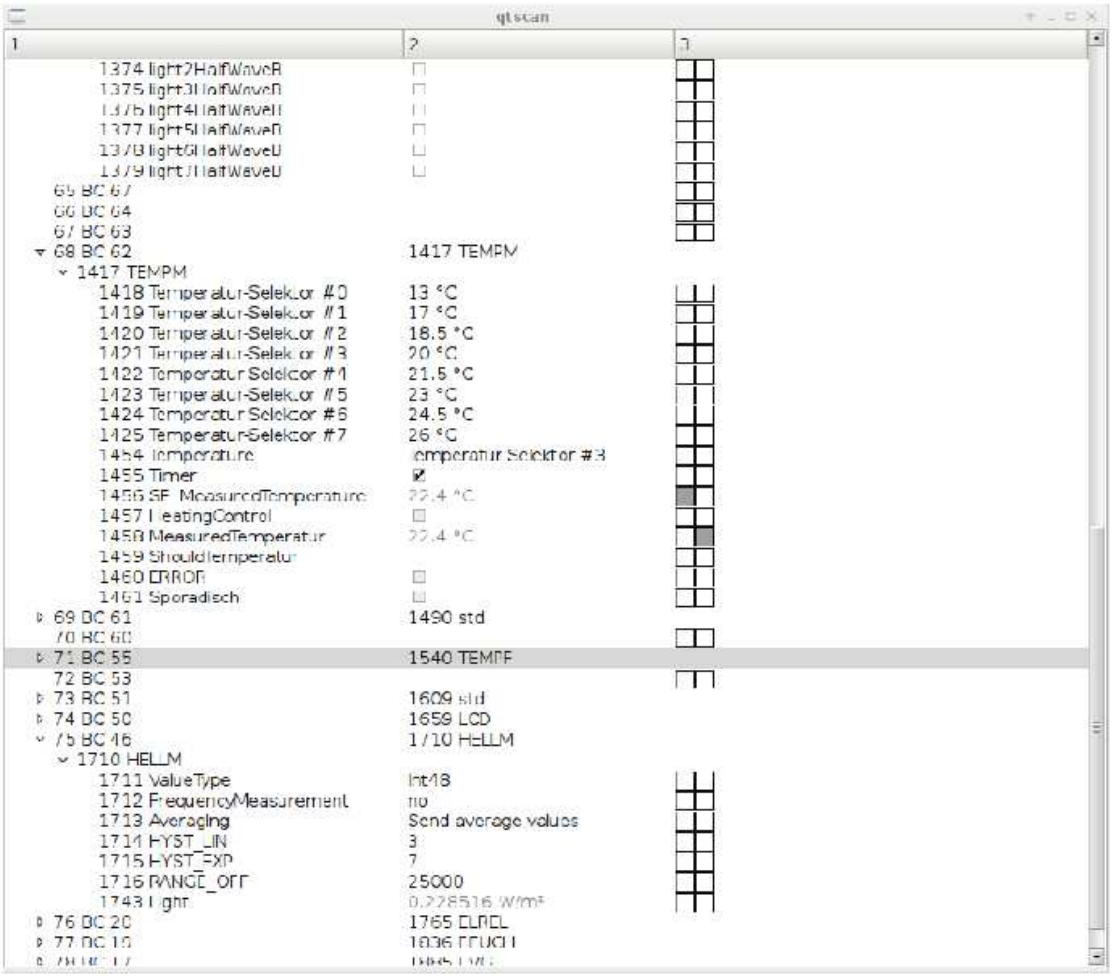


Abbildung 16: Baumansicht auf den SmallCAN-Bus

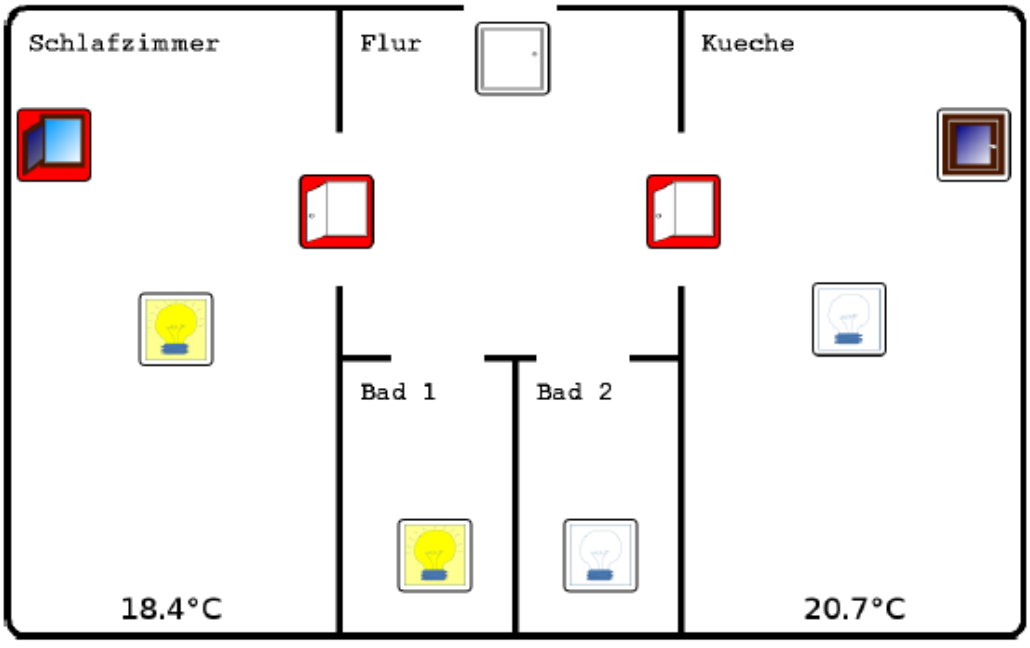


Abbildung 17: Vereinfachte Benutzeroberfläche 40

2.3 CE-Kennzeichnung

Um den SmallCAN in den Verkehr bringen zu können, ist es erforderlich die Notwendigkeit einer CE-Kennzeichnung des Produktes zu überprüfen. Dazu wird in den nachfolgenden Abschnitten das Prozedere zur Analyse der CE-Kennzeichnungspflicht und das Prozedere zur Erfüllung der Anforderungen zur CE-Kennzeichnung beschrieben.

2.3.1 Wie bekommt man eine CE-Kennzeichnung?

Für das Erlangen einer CE-Kennzeichnung sind nachfolgende Schritte notwendig:

1. Relevanzprüfung der CE-Richtlinien
2. Relevanzprüfung der harmonisierten Normen
3. Durchführung einer Gefährdungsanalyse und Risikobeurteilung
4. Erstellung von technischen Unterlagen
5. Aufsetzen einer Konformitätserklärung
6. Anbringen der CE-Kennzeichnung am Produkt

2.3.2 Ausgangspunkt Richtlinien

Die Grundlage einer jeden potenziellen CE-Kennzeichnung ist eine strukturierte Analyse der bestehenden CE-Richtlinien und der in diese Richtlinien referenzierten harmonisierten Normen. Aktuell existieren in Europa 28 verschiedene CE-Richtlinien [1], welche sich jeweils auf verschiedene Produktarten beziehen. Um das Produkt SmallCAN-Buskoppler konform dem geltenden europäischen Recht zu kennzeichnen, müssen zunächst die für das Produkt relevanten Richtlinien identifiziert werden.

2.3.3 Analyse der harmonisierten Normen

Harmonisierte Normen sind Normen, die eine oder mehrere Richtlinien zur Einhaltung der Anwendungen zur CE-Kennzeichnung konkretisieren. Harmonisierte Normen sind europäische Normen, die durch Comité Européen de Normalisation Electrotechnique (CENELEC), Europäische Komitee für Normung (CEN) und European Telecommunications Standards Institute (ETSI) im Auftrag der Europäischen Kommission und der EFTA erarbeitet werden.

Durch eine Einteilung der harmonisierten Normen in die folgenden Kategorien (A, B, C), soll eine Wiederholung identischer Norminhalte in Normen verhindert werden.

- A-Normen sind Sicherheitsgrundnormen, die Begriffe, Gestaltungsleitsätze und allgemeine Aspekte der CE-Richtlinien klären.
- B-Normen sind Sicherheitsgruppennormen, die bestimmte Produktgruppen gleichermaßen betreffen.
- C-Normen sind Sicherheitsproduktnormen, die sich auf einzelne Produkte und ihre Sicherheitsanforderungen beziehen.

Für die CE-Konformitätserklärung müssen alle relevanten Normen, die auf das jeweilige Produkt zutreffen und für die CE-Kennzeichnung relevant sind, dokumentiert werden, um deren Einhaltung bei einer späteren Überprüfung nachweisen zu können. Des Weiteren muss die in der Norm geforderte Risikoanalyse durchgeführt und ebenfalls, für einen späteren Nachweis, dokumentiert werden. Im Folgenden wird der Prozess einer Risikoanalyse beschrieben.

2.3.4 Grundlagen zur Risikoanalyse

Für eine CE-Kennzeichnung ist die Durchführung einer Risikoanalyse erforderlich die im Folgenden beschrieben. Des Weiteren wird aufgezeigt, wie eine Risikoanalyse durchgeführt wird.

Eine Risikoanalyse ist eine Analyse zur Beurteilung von Risiken. Sie dient zur frühzeitigen Erkennung von Gefährdungen (Gefährdungsidentifikation), und zur Bewertung der mit ihr verbundenen Risikopotenziale. Das Risikopotenzial wird durch die Kombination der Eintrittswahrscheinlichkeit des Gefährdungsereignisses und des potenziell resultierenden Schadensausmaßes charakterisiert.

Zur Durchführung einer Risikoanalyse, werden potenzielle Fehlermöglichkeiten des Produktes ermittelt und eingestuft. Durch das Ermitteln der Fehler werden Gefährdungen sichtbar. Zur Berechnung des Risikos werden folgende Parameter multipliziert:

1. Häufigkeit / Wahrscheinlichkeit, dass der Fehler auftritt
2. Das Ausmaß des Fehlers

Zur Abschätzung dieser Parameter schlägt die Sicherheitsgrundnorm IEC 61508-5 folgende Kategorisierung vor.

Risikoparameter		Klassifizierung	Erläuterungen
Auswirkung (*C*)	C_1	Geringe Verletzung	1 Das Klassifizierungssystem ist entwickelt worden, um Verletzungen und Tod von Personen zu berücksichtigen. Für Umwelt- und Materialschäden müssten andere Klassifizierungsverfahren entwickelt werden. 2 Bei der Interpretation von C_1, C_2, C_3 und C_4 müssen die Auswirkungen des Unfalls und normale Heilungsprozesse betrachtet werden.
	C_2	Schwere irreversible Verletzung einer oder mehrerer Personen; Tod einer Person	
	C_3	Tod mehrerer Personen	
	C_4	Tod sehr vieler Personen	
Häufigkeit und Aufenthaltsdauer im gefährlichen Bereich (*F*)	F_1	Seltener bis öfterer Aufenthalt im gefährlichen Bereich	3 Siehe Anmerkung 1 oben.
	F_2	Häufiger bis dauernder Aufenthalt im gefährlichen Bereich	
Möglichkeit, den gefährlichen Vorfall zu vermeiden (*P*)	P_1	Möglich unter bestimmten Bedingungen	4 Dieser Parameter zieht in Betracht: – Betrieb eines Prozesses (überwacht (d. h. betrieben durch ausgebildete oder nicht ausgebildete Personen) oder nicht überwacht); – Geschwindigkeit der Entwicklung des gefährlichen Vorfalls (z. B. plötzlich, schnell, langsam); – Leichtigkeit der Erkennung der Gefahr (z. B. unmittelbar erkennbar, durch technische Maßnahmen aufgedeckt, ohne technische Maßnahmen aufgedeckt); – Vermeidung des gefährlichen Vorfalls (z. B. Fluchtwege möglich, nicht möglich oder unter bestimmten Bedingungen möglich); – aktuelle Sicherheitserfahrung (diese Erfahrung kann von identischen oder ähnlichen EUC oder ähnlichen EUC herrühren, oder kann nicht vorhanden sein).
	P_2	Beinahe unmöglich	
Wahrscheinlichkeit des unerwünschten Ereignisses (*W*)	W_1	Eine sehr geringe Wahrscheinlichkeit, dass die unerwünschten Ereignisse auftreten, und nur wenige unerwünschte Ereignisse sind wahrscheinlich.	5 Der Faktor "*W*" dient zur Bestimmung der Häufigkeit des unerwünschten Ereignisses, ohne die Berücksichtigung jeglicher sicherheitsbezogener Systeme (E/E/PE oder andere Technologie), aber unter Berücksichtigung der externen Einrichtungen zur Risikominderung. 6 Wenn wenig oder gar keine Erfahrungen mit der EUC oder einem ähnlichen EUC oder EUC-Leit- oder Steuerungssystem bestehen, kann die Bestimmung des Faktors "*W*" durch Berechnung erfolgen. In solchen Fällen muss eine "worst-case"-Vorhersage gemacht werden.
	W_2	Eine geringe Wahrscheinlichkeit, dass die unerwünschten Ereignisse auftreten, und wenige unerwünschte Ereignisse sind wahrscheinlich.	
	W_3	Eine relativ hohe Wahrscheinlichkeit, dass die unerwünschten Ereignisse auftreten, und häufige unerwünschte Ereignisse sind wahrscheinlich.	

Abbildung 18: Kategorisierung der Risikoparameter [nach: IEC 61508-5]

Wurden die Parameter mittels der dargestellten Tabelle abgeschätzt, wird an Hand des Risikographen (siehe Abbildung 19) eine Risikobewertung durchgeführt.

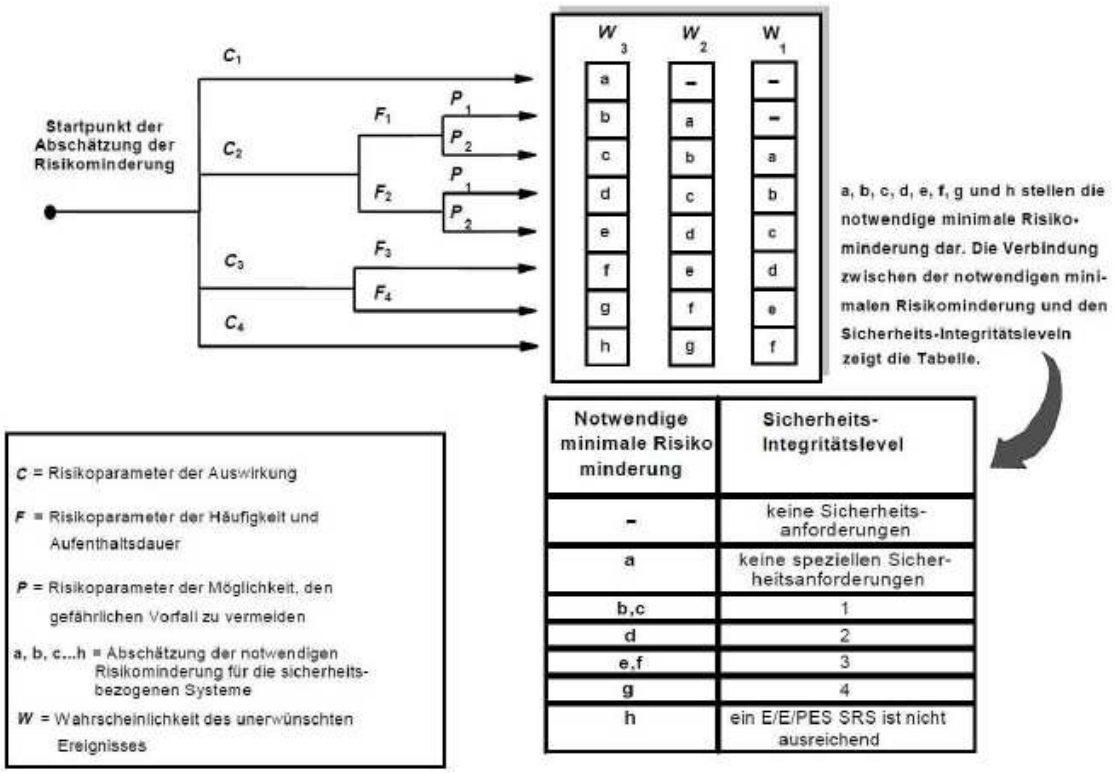


Abbildung 19: Risikograph [nach: IEC 61508-5]

2.3.5 Konformitätsnachweis für des SmallCAN-Buskopplers

Im Folgenden Abschnitt wird die Durchführung der einzelnen Aktivitäten zur CE- Kennzeichnung des SmallCAN-Buskopplers beschrieben. Des Weiteren wird eine Risikoanalyse durchgeführt. Darüber hinaus wird aufgeführt, welche Dokumente zum Erlangen eines Konformitätsnachweises relevant sind.

Das nachfolgende Flussdiagramm (Abbildung 20) visualisiert den Weg zum Erreichen einer Konformitätserklärung für den SmallCAN-Buskoppler.

Um für den SmallCAN-Buskoppler eine Konformitätserklärung zu erlangen, werden die 28 CE-Richtlinien augenscheinlich auf Relevanz den SmallCAN-Buskoppler überprüft, so dass im Folgenden nur die nachstehend gelisteten produktbezogene Richtlinien betrachtet werden.

1. 89/106/EWG Bauprodukte
2. 2004/108/EG Elektromagnetische Verträglichkeit (EMV)
3. 2006/95/EG Niederspannungsrichtlinie

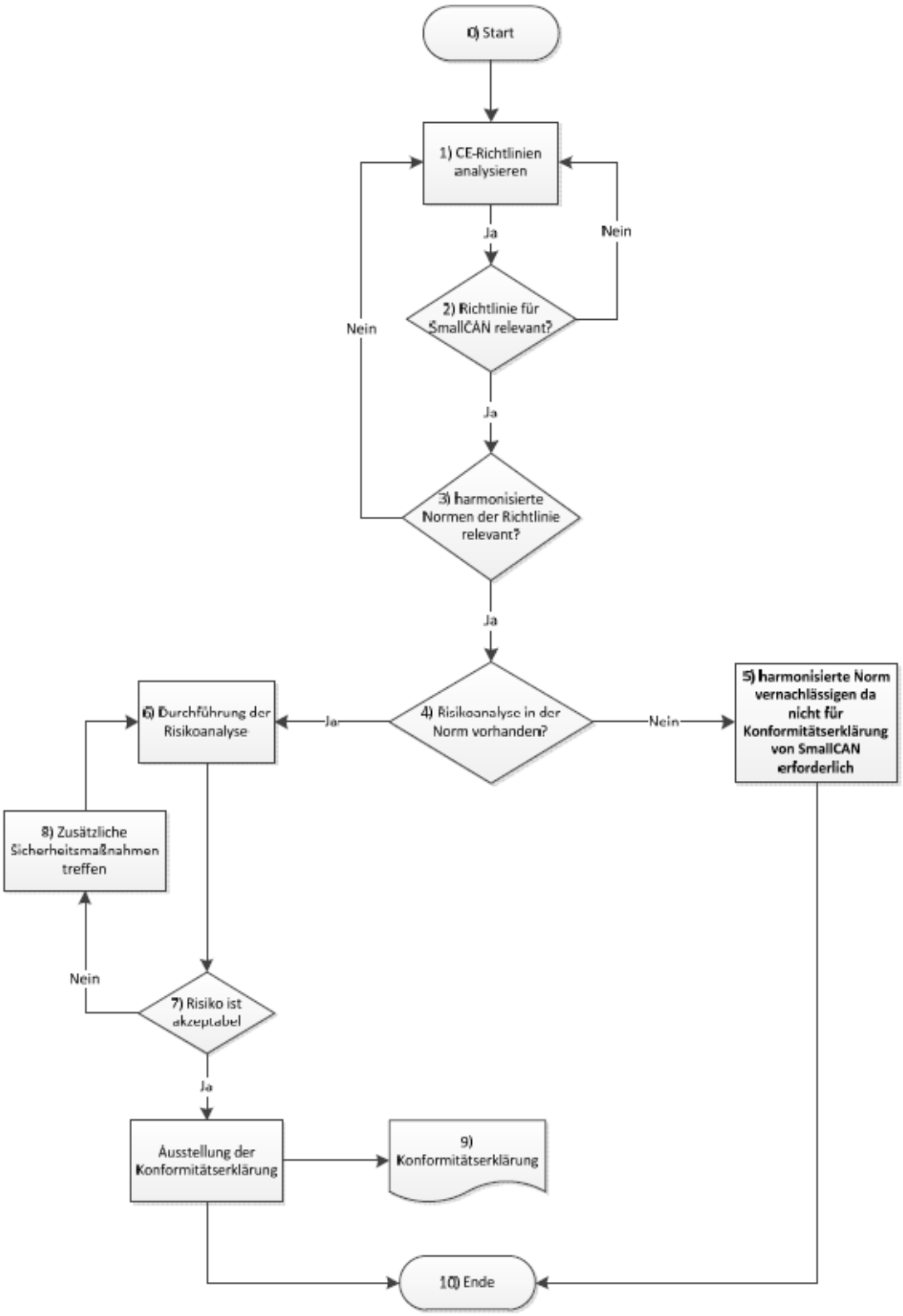


Abbildung 20: Flussdiagramm

Im Folgenden wurden die 3 identifizierten Richtlinien im Detail hinsichtlich ihrer Relevanz für SmallCAN überprüft. Hierbei wurde festgestellt, dass sich die Niederspannungsrichtlinie lediglich auf Spannungsbereiche zwischen 50V und 1000V Wechselspannung beziehungs-

weise zwischen 75V und 1500V Gleichspannung bezieht. Da SmallCAN nur mit einer Spannung zwischen 20V und 30V betrieben wird, konnte die Niederspannungsrichtlinie im Folgenden vernachlässigt werden.

Des Weiteren kann auch die Bauproduktrichtlinie ausgeschlossen werden, da diese nach detaillierterer Analyse keine für das Produkt SmallCAN-Buskoppler relevanten Normen enthält. Nach einer ersten Untersuchung der harmonisierten Normen, sind im Folgenden 6 harmonisierte Normen weiter zu verfolgen (siehe Tabelle 4).

	Richtlinien Bezeichnung	Richtlinie zur CE-Kennzeichnung	Harmonisierte Normen	Normtitel
1	2004/108/EG	EMV-Richtlinie	EN 50090	Elektrische Systemtechnik für Heim und Gebäude (EHSG)
2	2004/108/EG	EMV-Richtlinie	EN 60730	Automatische elektrische Regel- und Steuergeräte für den Hausgebrauch und ähnlichen Anwendungen
3	2004/108/EG	EMV-Richtlinie	EN 60947	Niederspannungsschalter
4	2004/108/EG	EMV-Richtlinie	EN 61000	Elektromagnetische Verträglichkeit
5	2004/108/EG	EMV-Richtlinie	EN 61131	Speicherprogrammierbare Steuerungen
6	2004/108/EG	EMV-Richtlinie	EN 61326	Elektrische Mess-, Steuer-,Regel- und Laborgeräte EMV-Anforderung

Tabelle 4: Liste von relevanten Richtlinien und harmonisierter Normen

Um eine Konformitätserklärung zu erhalten, sind nach [3] die harmonisierten Normen relevant, die die Anforderung der Durchführung einer Risikoanalyse enthalten. Diese Anforderung zur Durchführung einer Risikoanalyse ist nur in der Norm EN 50090 enthalten. Sie verweist auf hinsichtlich der Durchführung einer Risikoanalyse auf das in der Sicherheitsgrundnorm DIN EN 61508 beschriebene Prozedere zur Risikoanalyse

2.3.6 Durchführung der Risikoanalyse für SmallCAN-Buskoppler

Der Startpunkt einer jeden Risikoanalyse ist eine detaillierte Systembeschreibung. Im Rahmen der Durchführung der Risikoanalyse für den SmallCAN-Buskoppler wird hierbei auf ein Schema-Darstellung der Systemarchitektur (siehe: Abbildung 21) der Funktionsweise des Systems SmallCAN aufgesetzt.

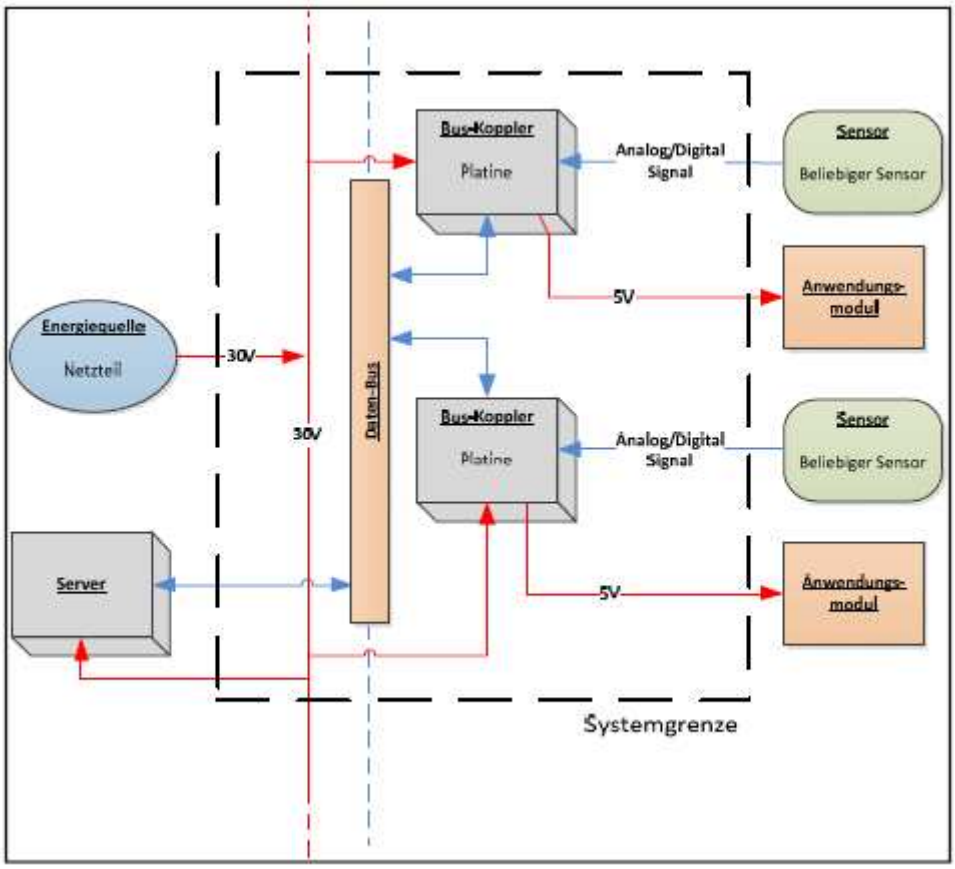


Abbildung 21: Systemarchitektur

Durch die Systemarchitektur werden alle externen und internen Schnittstellen des Systems visualisiert und dokumentiert (siehe Tabelle 5). Diese Schnittstellen und die darauf realisierten Funktionen (siehe Tabelle 5) werden in der Risikoanalyse betrachtet und auf mögliche Risiken geprüft (siehe Tabelle 6).

Anhand der in Abbildung 2 dargestellten Tabelle aus EN 61508 werden die verschiedenen möglichen Fehlfunktionen hinsichtlich:

- C: Schadensausmaß
- F: Häufigkeit und Aufenthaltsdauer
- P: Möglichkeit der Gefahrenabwehr
- W: Wahrscheinlichkeit des unerwünschten Ereignisses

abgeschätzt und kategorisiert. Im Anschluss daran werden die möglichen Fehlfunktionen unter Verwendung des in Abbildung 19 dargestellten Risikographen analysiert. Anhang A2 gibt einen Überblick über die Ergebnisse der durchgeführten Risikoanalyse für das System SmallCAN (ohne Applikation). Tabelle 7 stellt noch einmal die Fehlfunktionen mit dem größten Risikopotenzial dar.

Externe Schnittstellen

Schnittstellen zwischen Systemkomponenten übrigen Systemen

Interne Systemkomponente	Signalfluss	Signaltyp	Externe Systemkomponente	Realisierte Funktion
Buskoppler	<--	30V	Energiequelle	Spannungsversorgung Buskoppler
Buskoppler	-->	5V	Anwendungsmodul	Ansteuerung Anwendungsmodul
Buskoppler	<--	Analog/Digital	Sensor	Sensordaten übertragen
Daten-Bus	<-->	Digital / 24 V	Server	Datenaustausch

Interne Schnittstellen

Schnittstellen zwischen Systemkomponenten übrigen Systemen

Interne Systemkomponente	Signalfluss	Signaltyp	Externe Systemkomponente	Realisierte Funktion
Buskoppler	<-->	Digital / 24V	Daten-Bus	Datenaustausch

Tabelle 5: Schnittstellen des Sytstems

Nr.	Funktion	Fehlfunktionsbezeichnung	Nr.	Mögl. Fehlfunktionen	Kommentar
1	Spannungsversorgung des Bk	Fehlerhafte Spannungsversorgung des Bk	1	Zu hohe Spannung	Mögliches Abrauchen des Bk/ Brandgefahr
			2	Zu niedrige Spannung	Ausfall Bk / --> Fehlfunktion Anwendungsmodul
			3	Verpolung	Mögliches Abrauchen des Bk/ Brandgefahr
2	Ansteuerung Anwendungsmodul	Fehlerhafte Ansteuerung Anwendungsmodul	1	Ansteuerung ohne Anforderung	Gefährdung durch Anwendungsmodul
			2	Keine Ansteuerung trotz Anforderung	Gefährdung durch Anwendungsmodul
			3	Fehlerhafte Ansteuerspannung	Gefährdung durch Anwendungsmodul
3	Sensordaten übertragen	Fehlerhaftes Übertragen der Sensordaten	1	Fehlerhafte Einlesen von Sensordaten	-->Falsche Berechnung von Steuergrößen-->Gefährdung durch Anwendungsmodul
			2	Keine Übertragung der Sensordaten	-->Falsche Berechnung von Steuergrößen-->Gefährdung durch Anwendungsmodul
4	Datenaustausch Daten-Bus-->Server	Fehlerhafter Datenaustausch Server->Daten-Bus	1	Verfälschen von Telegrammen	-->Falsche Berechnung von Steuergrößen-->Gefährdung durch Anwendungsmodul
			2	Ausfall des Daten-Bus	Gefährdung durch Anwendungsmodul
5	Datenaustausch Buskoppler-->Daten-Bus	Fehlerhafter Datenaustausch Buskoppler-->Daten-Bus	1	Senden fehlerhafter Telegramme	Gefährdung durch Anwendungsmodul
			2	Zu häufiges Senden von Telegrammen	Gefährdung durch Anwendungsmodul
			3	Kein Senden von Telegrammen	Gefährdung durch Anwendungsmodul
			4	Fehlerhaftes Empfangen von Telegrammen	Gefährdung durch Anwendungsmodul
			5	Kein Empfang von Telegrammen	Gefährdung durch Anwendungsmodul
			6	Schwankende Signalspannung	Gefährdung durch Anwendungsmodul
6	Verarbeitung von Daten	Fehlerhafte Verarbeitung von Daten	1	Fehlerhafte Umsetzung der empfangenen Daten	Gefährdung durch Anwendungsmodul
			2	Keine Umsetzung der empfangenen Daten	z.B. uC hängt in Warteschleife /Gefährdung durch Anwendungsmodul

Tabelle 6: Liste der Fehlfunktionen

Nr.	Fehlfunktions-bezeichnung	Nr.	Mögl. Fehlfunktionen	Mögliches unerwünschtes Ereignis	Abschätzung der notwendigen Risikominderung (EN 61508)				Notwendige minimale Risiko-minderung (a..g)	SIL (1..4)	Bereits getroffene Maßnahmen	Zusätzlich zu treffende Maßnahmen gemäß EN 61508-2
					Auswirkung [C] (1..4)	Häufigkeit und Aufenthalts dauer [F] (1..2)	Möglichkeit den gefährlichen Vorfall zu vermeiden [P] (1..2)	Wahrscheinlich keit des unerwünschten Ereignisses [W] (1..3)				
1	Fehlerhafte Spannungsversorgung des Buskoppler (BK)	1	Zu hohe Spannung	Rauchentwicklung / Brandgefahr	2	2	1	1	b	1	HW: Strombegrenzer; Spannungsregler (V_{out}=50V); Supressor-Diode gegen Spitzenspannung Organisatorisch: Bedienungsanleitung, Anschluss durch Fachpersonal	keine
		3	Verpolung	Rauchentwicklung / Brandgefahr	2	2	1	1	b	1	HW: Verpolungsschutzdiode; mechanische Codierung Organisatorisch: Bedienungsanleitung, Anschluss durch Fachpersonal	keine

Tabelle 7: Zusammenfassung der Ergebnisse der Risikoanalyse (Auszug)

Bei genauerer Betrachtung stellt man fest, dass selbst die im Rahmen der Risikoanalyse mit einem SIL 1 eingestuften Funktionsumfänge, keiner Implementierung zusätzlicher sicherheitstechnischer Maßnahmen bedürfen. Diese Entscheidung liegt darin begründet, dass die SIL1-spezifischen Anforderungen aus der IEC 61508 bereits im Rahmen des Entwicklungsprozesses beachtet und umgesetzt wurden.

3 Status des Systems

Durch diese erste Grundentwicklung des SmallCAN ist es gelungen einen wettbewerbsfähigen Buskoppler mit einigen Anwendungsapplikationen zur Steuerung bzw. Regelung der Energieströme in einem Gebäude aufzubauen. SmallCAN steht dabei z.B. mit dem durch viele Firmen unterstützten Bussystem zur Gebäudeautomatisierung KNX/EIB im Wettbewerb.

Wir sehen hier doch einige Vorteile für den SmallCAN. Z.B. bezüglich der Verschaltungstechnik können bei KNX/EIB maximal 64 Teilnehmern in einer Linie verschaltet werden, bei mehreren Teilnehmern ist dann ein neues Netzteil notwendig. Bei SmallCAN können ca. 1000 Teilnehmern in einer Linie verschaltet werden mit einem Netzteil. Bei dem SmallCAN system ist ein autarker Betrieb möglich, da die Regelungs- und Steuerungssoftware auf die einzelnen Buskoppler verteilt ist und frei programmierbar (dezentrale Organisation). Bei KNX ist eine Änderung der Steuerungssoftware kostenpflichtig und nicht vollständig frei programmierbar.

Der Energieverbrauch pro Busankoppler beträgt bei KNX 150 mW und bei SmallCAN 68mW. Die Kosten bei KNX betragen pro Busankoppler zwischen 42,23 EUR - 85,46 EUR und der SmallCAN liegt bei einem Verkaufspreis von z.Z. ca. 14 EUR bei einer Fertigung von 1000 Stück.

Als Beispiel für ein Sensormodul (CO2-Sensor) betragen die Kosten KNX = 277 EUR/ SmallCAN ca. 100 EUR und der Stromverbrauch KNX = 8mA / SmallCAN= 2mA.

Die Gesamtenergieersparnis bei SmallCAN gegenüber dem KNX-System hängt sehr stark von der Ausstattungstiefe ab. Gegen Ende des 2. Quartals 2012 wird ein Einfamilienhaus mit SmallCAN ausgerüstet. Anschließend ist ein direkter Vergleich möglich.

Einige Kostenbeispiele im Vergleich zeigt die folgende Tabelle 8.

SmallCAN	Preis/ Stück	KNX	Preis/ Stück
Rollo Schaltungen	ca. 80 €	Rollo Schaltungen	143,12 €
Thermostat-Regler	ca. 50 €	Thermostat-Regler	250,18 €
Bus Zentraleinheit	ca. 65 €	Bus Zentraleinheit	242,10 €
Schaltereingänge	ca. 30 €	2 fach Tastermodul	82,36 €
vier-Kanal-Halogen	ca. 55 €	vier-Kanal-Halogen	326,45 €
230 V dim.	cd. 90 €	230 V dim.	391,00 €
Steckdose	ca. 55 €	Steckdose	241,55 €
RS232	ca. 70€	RS232	143,12 €
Stellventile/Heizungsaktor	cda. 75 €	Stellventile/Heizungsaktor	255 €
230 V Schaltbar	ca. 60 €	230 V Schaltbar	241,55 €
M Bus Gateway (z.B. Wärmemengenzähler)	ca. 60 €	M Bus Gateway (z.B. Wärme	903,76 €
DALI Gateway	ca. 45 €	DALI Gateway	581,40 €

Tabelle 8: Vergleich der Kosten verschiedener Applikatione SmallCAN / KNX

Bei den Applikationen (Aktoren) und Sensoren sind es z.Z. Kleinserien bis max. 20 Einheiten. Hier ist durchaus durch eine entsprechende Fertigung von größeren Einheiten noch eine Kostenreduzierung zu erwarten.

Durch selbst zu erstellende, individuelle Softwarekomponenten, die jederzeit erweitert werden können, ist es möglich sämtliche Informationen als Nachrichten auf den Bus abzulegen, die weitergenutzt werden können. Die Vielfalt dieser Daten ermöglicht es, sehr detaillierte und effektive Regelungsstrategien durchzuführen. Weiter werden die Hardwarekomponenten mit umfangreicher Schaltungstechnik bestückt, beispielsweise zur Prüfung von Relaisausgängen. Diese können dann geprüft, und die Diagnosedaten auf den Bus abgelegt werden und stehen damit wieder anderen Anwendungen zur Verfügung.

Alle Daten werden über eine einheitliche Schnittstelle an die Benutzeroberfläche gegeben. Dadurch können auch alle Daten zur Anzeige gebracht werden ohne Investitionsaufwand zur Interoperabilität leisten zu müssen. Die Software für die Benutzeroberfläche bei SmallCAN ist Betriebssystemunabhängig und kann deshalb individuell gestaltet werden.

Ein Server für den Opensource Ansatz wird z.Z. aufgesetzt.

Bei der Ausstattung der Büroräume ist die funktionalen Spezifikation gleich und konstant. Dadurch ist der Gesamtenergieverbrauch ein Maß für die Energieoptimierung bei gleichbleibenden Komfort.

Beispielhafte Maßnahmen sind:

1. verschiedenen Temperaturregelungsmodis (Nachtabsenkung, Vorhaltebetrieb, Stützbetrieb, Komfortbetrieb),

2. Netztrennung von Verbrauchern zur Vermeidung von Standby-Verbräuchen etc.

3. Vermeiden von energetische nicht geeigneten Systemzuständen, hervorgerufen durch Benutzer, beispielsweise die Einführung von Temperaturobergrenzen und Temperaturuntergrenzen abhängig von Sommer und Winterzeit.

Bisher hat sich im Vergleich zu den anderen Wettbewerbern ein Energieverbrauch von nur ca. 25 % gegenüber den nicht mit SmallCAN ausgestatteten Räumen ergeben. Für eine verlässliche Aussage ist hierzu jedoch eine Langzeitmessung über mindestens 2 Jahr notwendig.

Die ersten bisherigen Ergebnisse zeigen, das durchaus nennenswerte Energieeinsparungen möglich sind. Die Kostenansätze des Systems sind wie gezeigt doch erheblich unter den Kosten von z.B. KNX/EIB.

Durch die aufgesetzten Folgeprojekte wird eine weiter Verbreitung der Vorhabensergebnisse erreicht. Dies geschieht auch z.B. durch die Einbindung der Handwerkskammer und damit der Verbreitung bei Elektroinstallateuren und Heizungsbauern. Die Herausforderung besteht im Wettbewerb zu KNX/EIB (quasi Standard). Hier ist durch aus auch ein miteinander denkbar.

Desweiteren sind weiter Veröffentlichungen auf entsprechenden Tagungen geplant.

4 Fazit

Durch die Förderung der DBU ist die Möglichkeit geschaffen worden, aus der Idee des SmallCAN mit einigen Prototypen erste Ergebnisse in Richtung eines Produktes zu erzielen. Der Buskoppler selber ist inzwischen in einer ersten Auflage mit 1000 Einheiten automatisiert gefertigt und bestückt worden. Von einigen Applikationen sind Kleinserien von 5- 20 Einheiten gefertigt.

Die Softwareentwicklung ist durch die Neuorientierung und -strukturierung noch nicht endgültig abgeschlossen und wird weiter entwickelt auch für neue Applikationen.

Die ersten Messergebnisse aus dem „future-workspace“ sind sehr vielversprechend gerade in Hinsicht auf den Standby-Verbrauch gerade im Vergleich zu den Mitbewerbern des Ausbaus (ca. nur 25% des Energieverbrauchs zu den Mitwebbewerbern).

Durch diese grundlegenden Vorarbeiten ist inzwischen eine weitere Förderung durch das BMWi erfolgt.

Seit dem 1.10.2011 wird das BMWi-Projekt „Digaflex - Demonstrationsanlage einer integrierten Gebäudeautomatisierung mit low-Power, low-Cost Ansatz und flexiblem Gerätespektrum und flexibler Konfiguration“, Kennzeichen 03ET1016A gefördert. Dieses Projekt ist gemeinsam mit dem Institut für Verkehrssicherheit und Automatisierungstechnik beantragt worden. Arbeitsziel des Projektes ist es in zwei beispielhaften Demonstrationsinstallationen in ausgedehnten Liegenschaften die Vorteile über einen längeren Zeitraum zu demonstrieren. Die Laufzeit beträgt 5 Jahre, wobei die letzten beiden Jahre ausschließlich dem Monitoring des installierten Systems dienen.

Weiter kann das System auch zur Komfortsteigerung, insbesondere durch neue Dienste zur Unterstützung von behinderten oder älteren Menschen dienlich sein. Hierzu wird eine Kooperation mit dem Braunschweiger Informatik- und Technologie-Zentrum, die BITZ GmbH angestrebt. Hier ist insbesondere das Projekt ehealth.Braunnschweig (http://www.ehealth-braunschweig.de/) zu nennen.

Erste Schritte in Richtung Ausbildung sind durchgeführt.

Prof. Dr.-Ing. Yongjian DING von der Fachhochschule Magdeburg Stendal wir im Frühjahr 2011 ein kleine Demonstrationswand erhalten um diese in der Lehre in einem Labor einzusetzen.

Mit der Handwerkskammer Braunschweig finden z.Z. sehr vielversprechende Gespräche statt das System im Berufsbildungszentrum einzusetzen.

Z.Z. werden außerdem mit weiteren potentiellen Kandidaten erfolgversprechende Gespräche über den Demonstrationseinsatz von SmallCAN geführt (u.a. Solvis Braunschweig).

Das Gesamtfazit ist durch die erreichten Ziele positiv, es haben sich aber auch noch einige Herausforderungen während der Entwicklung gezeigt (wie z.B. bei der Softwareentwicklung beschrieben).

4 Literaturverzeichnis

Quellenverzeichnis

[1] Deutschen Patent- und Markenamt, Patent-Nr. 10034087: Feldbussystem mit minimierter Hardwarearchitektur.

[2] Schrom, H.: Realisierung eines optimierten Feldbussystems und Modellierung mit Petrinetzen. Dissertation, TU-Braunschweig, 2003.

[3] Schrom, H.: Optimiertes Feldbussystem. VDM-Verlag, Saarbrücken, 2007

[4] Schrom, H., Schnieder, E.: SCAN, A hardware minimised low cost / low power bus. MICRO.tec 2000, Hannover, Germany, 25.-27. September 2000.

[5] Schrom, H., Pfeiffer, A., Schnieder, E.: SmallCAN, ein Low-Power, Low-Cost, openSource Feldbussystem mit integraler Energieversorgung und hoher Teilnehmeranzahl. Elektor, eingereicht.

[6] http://www.eg-richtlinien-online.de/cn/bGV2ZWw9dHBsLXJpY2h0bGluaWVu.html , Letzter Zugriff 02.12.2012

[7] Losano, Mario G.: Turbulenzen im Rechtssystem der modernen Gesellschaft: Pyramide, Stufenbau und Netzwerkcharakter der Rechtordnung als ordnungsstiftende Modelle. Duncker+Humblot, 2007

[8] Jo Horstkotte: CE-Zeichen für Chefs: Was Sie zur CE-Kennzeichnungspflicht wissen sollten. 2010

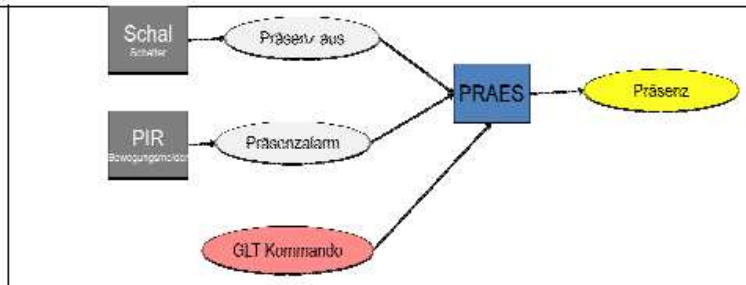
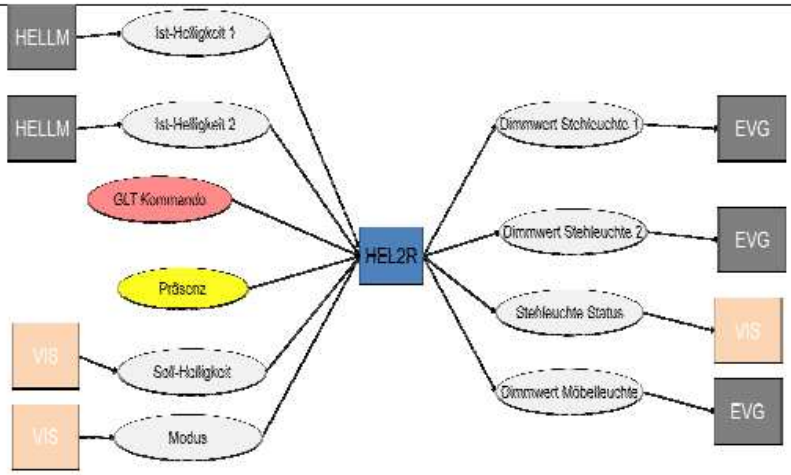
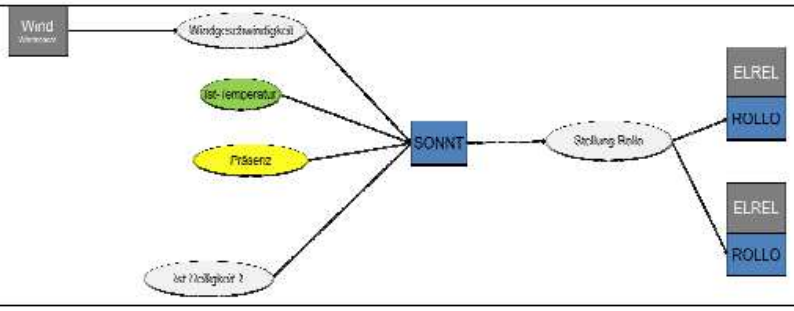
[9] Diekhake, P.; Fähndrich, E.; Schnieder, E.; Becker, U.: SmallCAN: Integrierte Gebäudeautomatisierung durch einheitliches low-Power, low-Cost, OpenSource Feldbussystem. Automation 2011, Baden-Baden, Deutschland, Juni 2011.

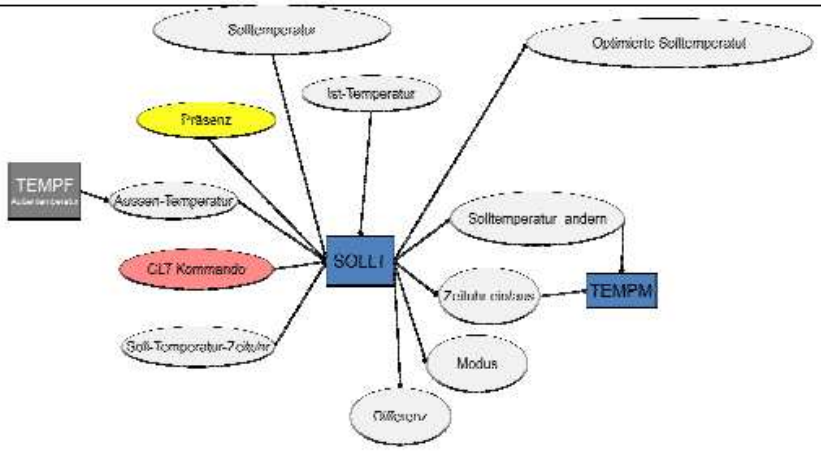
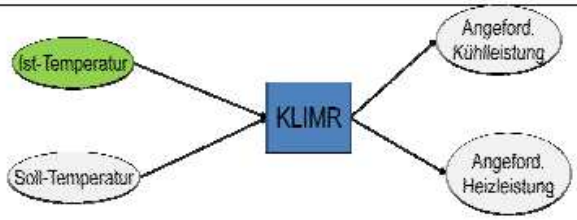
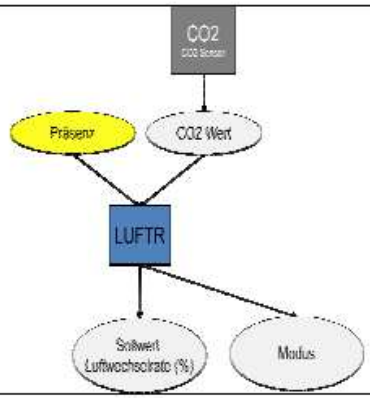
5 Anhänge

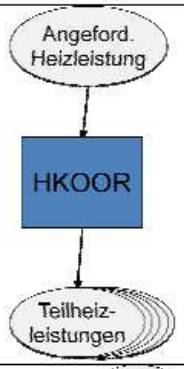
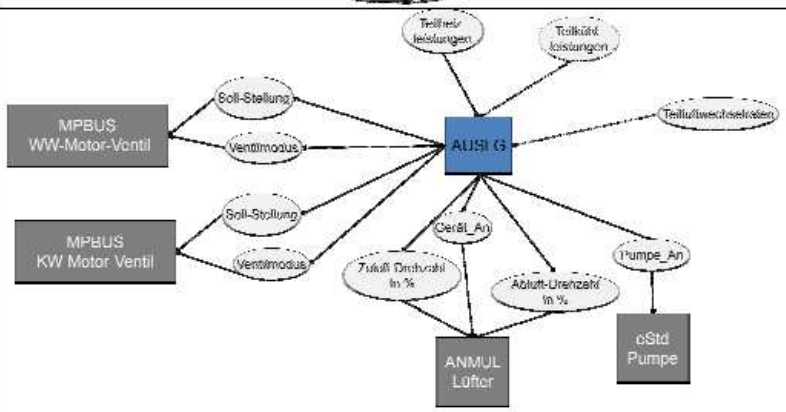
5.1 Anhang A1

Folgende wesentlichen verarbeitenden Funktionen seien in Tabelle A1 beschrieben und graphisch dargestellt.

Tabelle A1: wesentliche entwickelte verarbeitenden Funktionen.

PRAES_FSF (Präsenzdetektor)	Diese Funktion liest die Präsenzmeldungen des PIR Sensors eines Raums, Gebäudeleittechnikkommandos und ein Schalterstatus ein und ermittelt daraus eine kurze oder lange Präsenz	
HEL2R_FSF (Helligkeitsregler)	Diese Funktion liest die kurze oder lange Präsenz vom Präsenzmaster, die aktuellen Gebäudeleittechnik-Kommandos, die aktuelle Helligkeit, eine Soll-Helligkeit und einen Modus-Schalter für den Regler-Betrieb und steuert entsprechend Möbelleuchten und Stehleuchten an.	
SONNS_FSF (Sonnenschutz)	Diese Funktion liest die Windgeschwindigkeit sowie die Sonneneinstahlung ein und steuert bei Präsenz entsprechend eine	

	Beschattungs-anlage	
SOLLT (Solltempera-turbestimmun g)	Diese Funktion liest die Außen-lufttemperatur, Präsenz, Soll-temperaturen des Nutzers und der Zeituhr sowie die Raumtempera-tur ein und ermittelt mit-tels Behaglich-keitsdiagramm eine optimierte Solltemperatur	
KLIMR_FSF (Klimaregler)	Diese Funktion liest die Soll-Temperatur und die Ist-Temp eines Raums und berechnet da-raus eine Leistungszu- bzw. -abfuhr, um die ge-wünschte Raumtempera-tur zu errei-chen.	
LUFTR_FSF (Lüftungsmas-ter)	Diese Funktion liest den Ist-CO2-Wert ein und den Soll-wert aus dem EEPROM und berechnet da-raus eine Luftwechselrate in % und gibt den Modus aus.	

HKOOR_FSF Heizungskoordinator	Diese Funktion liest die Soll-Leistungsanforderung für den jeweiligen Raum und verteilt diese Anforderung auf bis zu 8 zur Verfügung stehenden Geräte	
AUSLG_FSF (Außenluftgerät)	Diese Funktion liest zahlreiche Eingangswerte (Heizteilleistung, Kühlteilleistung, Teilluftwechselrate) und steuert auf dieser Basis ein Außenluftgerät, das aus mehreren Buskopplern besteht (Ventile, Lüfter, Pumpe)	

Exemplarische verarbeitende Funktion für das Außenluftgerät

Am Beispiel einer eines Außenluftgerätes soll die verarbeitende Funktion näher erläutert werden. Ein Lüftergerät der Firma Trox besteht aus Radiallüfter für Zu- und Abluft und einer einfachen Lüfterelektronik die Stufenlos über eine 0-10V Schnittstelle angesteuert werden kann, und somit die Lüftergeschwindigkeiten bestimmt. Wird die Lüfterelektronik nicht mit Energie gespeist, schließt sich automatisch die Außenklappe. Das Lüftergerät verfügt weiter über Kalt- und Warmwasseranschlüsse sowie einen Konvektor zur thermischen Energieübertragung. Zum Betrieb des Klimagerätes muss neben der Ansteuerung des Lüftergerätes auch Warm- und Kaltwasserventile sowie eine Umwälzpumpe betrieben werden. Die Ansteuerung des Lüftergerätes erfolgt über ein Analogmodul. Die Ansteuerung der digitalen Stellventile wird durch ein Anwendungsmodul zur Ansteuerung des firmeninternen Busses der Fa. Belimo (siehe Tabelle xy: Nr 36) durchgeführt. Die Umwälzpumpe wird über eine Relaisplatine geschaltet. Diese vier vernetzten SmallCAN Komponenten (Lüftergeräte, Kalt- und Warmwasser-Ventile, Umwälzpumpe) müssen sinnvoll miteinander kommunizieren um ein Klimagerät zu bilden. Dazu liest die verarbeitende Funktion AUSLG die Sollwerte für Heizleis- tung, Kühlleistung oder der erforderlichen Luftwechselrate ein und steuert entsprechend die Einzelkomponenten an. Da diese Interaktionen ausschließlich nachrichtenbasiert

erfolgen, ist die Lokalisierung dieses Softwaremoduls frei wählbar. Für den Modus einer geforderten Luftwechselrate werden die Stellventile geschlossen und Frischluft durch das Öffnen der Außenluftklappe und Ansteuern der Lüfter in den zu versorgenden Raum geführt.

Folgender Quellcode ist dafür erforderlich.

```
//Lüften
if (Luftwechselrate_Anforderung>0)//wenn Anforderung vom Lüftungsregler
 {
if (!Luefter_An) // Wenn die Lüfter noch aus sind, dann einschalten
 {
Luefter_An=1;
FL_SEND_POWER = 1;
}
Luefter_Abluft = Luftwechselrate_Anforderung;
Luefter_Zuluft = Luftwechselrate_Anforderung;
FL_SEND_ABLUFT = 1;
FL_SEND_ZULUFT = 1;
 FLS_FREI1 = 1;
}
```

Damit bei niedrigen Temperaturen (Außentemperatur <=5℃) der Konvektor nicht einfriert ist ein Frostschutz zu realisieren, der folgende Spezifikationen einhalten muss:

- Anfahrschaltung bei jedem Gerätestart: Vor der Öffnung der Außenluftklappe muss zunächst das Ventil geöffnet werden bis die Differenz von Rücklauftemperatur und Vorlauftemperatur 5 Kelvin entspricht (TRL=TVL-5K). Erst dann wird die Stromversorgung für die Lüfter eingeschaltet, da sich hierdurch die Außenluftklappe öffnet.
- Die Rücklauftemperatur darf nicht kleiner als 25℃ sein
- Wenn die Pumpe stoppt (keine Umwälzung), schaltet die Stromversorgung für dieLüfter ab und die Außenluftklappe schließt.
- Sollte das Anfahren 3x hintereinander fehlschlagen, so wird ein Fehler gemeldet und das Gerät verriegelt. Die Entriegelung erfolgt durch den Administrator

Abbildung A1 stellt die Spezifikationen des Frostschutzes in einem Petrinetz dar. Unter der Bedingung, dass die Außentemperatur kleiner gleich 5℃ ist der Frostschutz aktiv und es muss die Rücklauftemperatur größer 25℃ und das Ger ät entriegelt sein, damit das Warm-

wasserventil geöffnet werden darf. Kann nicht innerhalb einer vorgegebenen Zeit die Differenz zwischen Vorlauftemperatur und Rücklauftemperatur erreicht werden, erfolgt der ers- te Fehlversuch. Nach drei Fehlversuchen wird das Gerät verriegelt und kann nur über einen Reset über den Bus wieder entriegelt werden. Ist die Anfahrschaltung erfolgreich darf das Gerät eingeschaltet und damit die Außenklappe geöffnet werden. Falls bei eingeschalteten Gerät die Rücklauftemperatur unter 25℃ sinkt oder die Pumpe stoppt, wird das Lüftergerät automatisch ausgeschaltet.

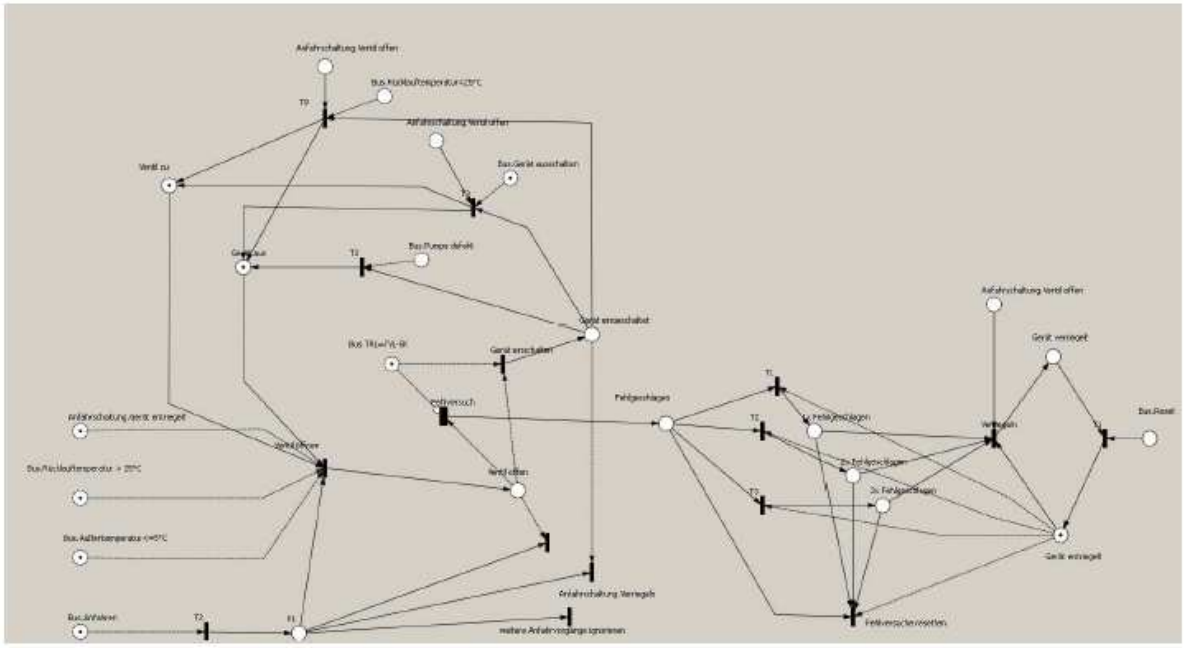


Abbildung A1: Petrinetz-Modell für den Frostschutz des Außenluftgerätes

Um den Frostschutz zu ermöglichen muss die verarbeitende Funktion AUSLG auch die Messwerte der Sensoren für Außenluft Vorlauf- und Rücklauf-Temperaturen sowie den Pumpenzustand erfassen. Der Quellcode der verarbeitenden Funktion AUSLG inklusiv der Frostschutzfunktion beläuft sich auf 840 Zeilen C-Code.

5.2 Anhang A2 Risikoeinstufung

Nr.	Fehlfunktions-bezeichnung	Nr.	Mögl. Fehlfunktionen	Mögliches unerwünschtes Ereignis	Abschätzung der notwendigen Risikominderung (EN 61508)				Notwendige minimale Risiko-minderung	SIL	Bereitsgetroffene Maßnahmen	Zusätzlich zu treffende Maßnahmen gemäß EN 61508
					Auswirkung [C]	Häufigkeit und Aufenthaltsdauer [F]	Möglichkeit den gefährlichen Vorfall zu vermeiden [P]	Wahrscheinlichkeit des unerwünschten Ereignisses [W]				
1	Fehlerhafte Spannungsversorgung des Buskoppler (Bk)	1	Zu hohe Spannung	Rauchentwicklung / Brandgefahr	2	2	1	1	b	1	**HW:** Strombegrenzer; Spannungsregler (V_{max}=50V); Supressor-Diode gegen Spitzenspannung **Organisatorisch:** Bedienungsanleitung, Anschluss durch Fachpersonal	keine
		2	Zu niedrige Spannung	Ausfall Bk (ab 17V) / --> Fehlfunktion Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**HW:** Stützkondensator, Pull-Down, Pull-Up **SW:** Programmroutine (Buskoppler in FailSafe)	keine
		3	Verpolung	Rauchentwicklung / Brandgefahr	2	2	1	1	b	1	**HW:** Verpolungsschutzdiode; mechanische Codierung **Organisatorisch:** Bedienungsanleitung, Anschluss durch Fachpersonal	keine
2	Fehlerhafte Ansteuerung Anwendungsmodul	1	Ansteuerung ohne Anforderung	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**Organisatorisch:** Software-/Hardwaretests	keine
		2	Keine Ansteuerung trotz Anforderung	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**Organisatorisch:** Software-/Hardwaretests	keine
		3	Fehlerhafte Ansteuerspannung	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**HW:** Stützkondensator, Pull-Down, Pull-Up **Organisatorisch:** Software-/Hardwaretests	keine
3	Fehlerhaftes Übertragen der Sensordaten	1	Fehlerhafte Einlesen von Sensordaten	-->Falsche Berechnung von Steuergrößen--> Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**HW:** Bei sicherheitsrelevanten Anwendungsmodulen ist Redundanz und Plausibilisierung über weiteren Bk möglich.	keine
		2	Keine Übertragung der Sensordaten	-->Falsche Berechnung von Steuergrößen--> Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Fehlerdiagnose, Fehler-Telegramm **Organisatorisch:** Visualisierung (z.B. über LEDs)	keine
4	Fehlerhafter Datenaustausch Server->Daten-Bus	1	Vorfälschen von Telegrammen	-->Falsche Berechnung von Steuergrößen--> Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Checksumme	keine
		2	Ausfall des Daten-Bus	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Fehlerdiagnose (durch Server) **Organisatorisch:** Visualisierung (z.B. über LEDs)	keine
5	Fehlerhafter Datenaustausch Buskoppler-->Daten-Bus	1	Senden fehlerhafter Telegramme	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Checksumme	keine
		2	Zu häufiges Senden von Telegrammen ("Babbling-Idiot-Problem")	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Lastbegrenzung (138 Bits/s), Überwachung durch Server, Abschalten (Bus-Off) und Reset des fehlerhaften Bk, Telegramm-Priorisierung	keine
		3	Kein Senden von Telegrammen	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Datenstau-Telegramm (Bk), Fehlerdiagnose (durch Server) **Organisatorisch:** Visualisierung (z.B. über LEDs)	keine
		4	Fehlerhaftes Empfangen von Telegrammen	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Checksumme	keine
		5	Kein Empfang von Telegrammen	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Fehlerdiagnose, Fehler-Telegramm (LOST-Telegramm) **Organisatorisch:** Visualisierung (z.B. über LEDs)	keine
		6	Schwankende Signalspannung	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Überwachung der Signalqualität, Fehler-Telegramm **Organisatorisch:** Visualisierung (z.B. über LEDs)	keine
6	Fehlerhafte Verarbeitung von Daten	1	Fehlerhafte Umsetzung der empfangenen Daten	Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**HW:** Bei sicherheitsrelevanten Anwendungsmodulen ist Redundanz und Plausibilisierung über weiteren Bk möglich. **Organisatorisch:** Software-/Hardwaretests	keine
		2	Keine Umsetzung der empfangenen Daten	z.B. uC hängt in Warteschleife /Gefährdung durch Anwendungsmodul	1	-	-	2	-	Keine Sicherheits-anforderungen	**SW:** Betriebssystem arbeitet "Quasi-Parallel" (keine Warteschleife)	keine