SmallCAN

Ein Low-Power, Low-Cost, openSource Feldbussystem mit integraler Energieversorgung und hoher Teilnehmeranzahl

Der Wunsch nach Vernetzung und Automatisierung ist weiterhin ungebrochen, sowohl im industriellen als auch im privaten Umfeld. Während die Industrie mit Ethernet und Feldbussen (z.B. CAN und PROFIBUS) auf Industrie spezifizierte und zertifizierte Produkte im oberen Preissegment arbeitet, zeichnet sich bei Büro- und Heimanwendungen eine immer stärkere Trennung zwischen leistungsfähigen Computernetzwerken /Multimediabussen auf der einen, sowie spezialisierten Systemen wie EIB, LON, DALI und Funklösungen auf der anderen Seite ab. Diese Trennung basiert auf Kosten- und Aufwandsüberlegungen. Würde eine dem heutigen Trend zur Ausrüstung aller Geräte mit Ethernetschnittstellen und der damit verbundenen Notwendigkeit verteilter „Webserver" inklusive der dahinterliegenden sternförmigen Netzwerktechnik gelegt, wäre eine Kostenexplosion die Folge und Knoten mit relativ einfachen Funktionen (z.B. Lichtschalter) würde das 20-fache ihrer Grundkosten übersteigen und aufgrund ihres ständigen Stromverbrauchs dem Trend der globalen Energieeinsparung entgegen wirken. Daher sind spezialisierte Systeme mit geringen Kosten und geringem Energiebedarf notwendig, die aufgrund ihrer derzeitigen Kosten und begrenzenden Eigenschaften bisher jedoch keine Marktdurchdringung erreichen konnten. Das hier vorgestellte System stellt nun eine Möglichkeit dar, viele Teilnehmer-Knoten sehr kostengünstig miteinander zu vernetzen und gleichzeitig durch den offenen Zugang eine schnelle Anpassung an verschiedenste Aufgabenfelder zu möglichen (Architekturschablone). Das integrale Konzept und die Standardvorgaben vereinheitlichen dabei Stromversorgung, Datenkommunikation und Gehäuse, so dass der Fokus auf der primären Aufgabe liegen kann und die bisher üblichen, disjunkten elektronischen Inseln in einen kooperativen Verband treten.

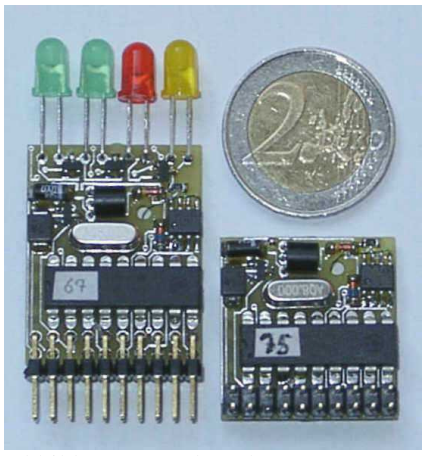

Abbildung 1: Buskoppler

Einordnung

Das vorgestellte System kann sowohl die Anforderungen an ein industrielles Feldbussystem erfüllen und ist gleichzeitig für die Heimautomatisierung geeignet und vereint die Vorteile einer Reihe anderer Systeme:

- Einfache Relaisansteuerung (wie beim ASI)
- Messwertübermittlung in kurzen Datenblöcken mit geringer Latenzzeit (wie beim CAN)
- Flexibel durch volle Programmierbarkeit von Funktionen (wie LON)
- Integrierte Energieversorgung (wie beim EIB) bei stark erhöhter Teilnehmeranzahl
- Wesentlich geringerer Energiebedarf als z.B. ELV RS485-System (Faktor 10 bis 20)
- Multimasterfähig (wie CAN)
- Preiswerter als LCN
- Wesentlich Leistungsfähiger als X10
- Geringerer Energiebedarf und breiteres Einsatzspektrum als digitalSTROM
- Arbeitet über einfache Telefonleitungen (o.ä.) wie HomePlug
- Integriert mühelos die Funktion diverser Spezialsysteme in ein gemeinsames System bei gleichzeitig einheitlichem Zugriff und ermöglicht eine echte und automatische energiesparende Netztrennung im Auszustand:
  DALI (Beleuchtung),
  MBUS (Verbrauchserfassung),
  ebus (Heizung), ...
- Ersetzt serielle Geräteanbindungen per RS485 oder RS232, beispielsweise für programmierbare Netzteile, Labormessgeräte, PV-Wechselrichter
- Kann Datenlogger, Überwachungs- und Diagnosegeräte ersetzen

Mit dem vorgestellten System sind daher Systeme „aus einem Guss" möglich. Die langfristige Verfügbarkeit ist durch den „OpenSource" Charakter gewährleistet, wodurch ein inhärenter Vorteil gegenüber der Vielzahl derzeit verfügbarer Lösungen gegeben ist. Preiswerte Standard-Elektroartikel lassen sich durch An- oder Einbau von Anwendungsadaptern zur Bustauglichkeit umrüsten.

Ziele

Ziel des SmallCAN ist ein offenes Feldbussystem, das für viele Anwendungszwecke gleichermaßen geeignet ist und gleichzeitig eine zuverlässige und wartungsfreie Systemlösung darstellt. Aus diesem Grunde werden ausschließlich kostengünstige und gut erhältliche Standardkomponenten verwendet. Die Wahl des Übertragungsmediums ist ebenfalls aus Kosten- und Verfügbarkeitsgründen getroffen worden und resultiert in der Verwendung handelsüblicher Telefonleitungen (Sternvierer),wobei gegebenenfalls vorhandene Leitungen genutzt werden können. Die notwendige Software ist entsprechend frei verfügbar, ebenso Dokumentation und Layouts, so dass eine langfristige Verfügbarkeit und der notwendige Spielraum für Änderungen gewährleistet werden können.

Insbesondere wird ein Großteil des Aufwandes beim Entwurf kleiner elektronischer Schaltungen elegant eingespart: Neben den eigentlichen Schaltungskomponenten sind grundsätzlich „Nebenkomponenten“ wie beispielsweise ein (Stecker-) Netzteil, ein Gehäuse und eine Ausgangsschaltung notwendig. Das heißt, obwohl es sich meist um Sensorschaltungen handelt, ist gleichzeitig eine Festlegung der Ausgangstreiberschaltung notwendig, die als „universelle“ Auslegung meist ein billiges Kleinrelais mit hohem Stromverbrauch erhält und in der Regel unangepasst ist. Eine Anpassung an die (zum Designzeitpunkt noch nicht bekannten) Aktoren ist nicht möglich, somit ist weder eine Nutzung von verschleißfreien elektronischen AC oder DC-Relais möglich, noch der Einsatz von teuren, Steckdosentauglichen Leistungsrelais (16A). Auch besonders aufwändige Ansteuerungsarten wie Phasenanschnitt oder PulsPaketModulation sind als „Standardfall“ keinesfalls vorgesehen. Durch die Nutzung des SmallCAN entfällt dieses „Vereinigungsproblem“, da Sensor und Aktor getrennt voneinander entwickelt und getestet werden können. Eine Kombination beliebiger Sensoren, einer zwischengeschalteten (anpassbaren) Verarbeitung und eines auf den jeweiligen Aktor exakt abgestimmten (Relais-) Ausgangstreibers kann fallspezifisch, unter Berücksichtigung der Eignung, Verschleißfreiheit und der Kosten, bei der Planung beziehungsweise Installation erfolgen. Gleichzeitig wird die örtliche Abhängigkeit zwischen Sensor und Aktor aufgehoben (lange Analogsensorkabel entfallen ebenso wie Anschlusskabel zu Lampen, Motoren, etc.), da der SmallCAN als universelle Kommunikationsinfrastruktur dient und die Einzelkomponenten direkt am Einsatzort montiert werden. Auch die sonst schnell ansteigende Zahl an (störanfälligen und stromfressenden) Steckernetzteilen entfällt.

Ein weiterer wichtiger Punkt ist die Skalierbarkeit des Systems. Während konventionelle Regelgeräte grundsätzlich auf eine feste Anzahl von Ein- und Ausgängen festgelegt sind (die sich unter Berücksichtigung der Kosten meist in engen Grenzen bewegen), besteht hier die Möglichkeit (auch noch später) jederzeit zusätzliche Ein- und Ausgangkanäle hinzuzufügen. Aufgrund der hohen maximalen Teilnehmeranzahl ist somit eine fast beliebige Skalierbarkeit zu jedem Zeitpunkt gegeben.

Weiterhin besteht innerhalb des Projektes die Möglichkeit, Schaltungsmodule zur allgemeinen Verwendung systematisch aufzubereiten und zu archivieren sowie, mit einer einheitlichen Busschnittstelle versehen, zur direkten Nutzung bereit zu stellen (inklusive Energieversorgung). Somit kann ein umfangreiches Archiv für nützliche Kleinschaltungen entstehen. Durch den offenen Zugang zu Hard- und Software ist eine Anpassungsmöglichkeit auf spezielle eigene Anforderungen jederzeit gegeben.

Einer der am kontroversesten diskutierten Punkte betrifft die Gegenüberstellung von kabelgebundenen und funkbasierten Systemen. Funksysteme lassen sich zwar schnell montieren, aber ein regelmäßiger und rechtzeitiger Batteriewechsel kann einen erheblichen Aufwand und Einschränkungen in der Funktionsverfügbarkeit darstellen (nur wenige Sensoren können selbst Energie erzeugen). Auch Hilfsenergie, gar für Anzeigen und Beleuchtungen, steht bei funkbasierten Systemen in der Regel nicht zur Verfügung und erfordert somit ohnehin das Vorhandensein einer kabelgebundenen Energie, um die gesteuerten Verbraucher zu versorgen. Auch wenn keine Kabelverlegung notwendig ist, so verursacht das Funkmodul doch gewisse Kosten, freie Rechenleistung für Applikationen steht nicht bereit, das Netzwerkmanagement ist stark eingeschränkt und die Störunempfindlichkeit des Funkweges ist in jedem Einzelfall zu klären. Somit kann Funk die drahtgebundenen Lösungen bestenfalls ergänzen, aber nicht ersetzen. Auch wenn kurzfristig schnelle Lösungen mit Funkanbindungen erzielbar sind, so ist langfristig doch eine sichere und zuverlässige Kabellösung vorzuziehen, siehe auch [3].

Auch den so genannten Power-Net Lösungen (Informationsaustausch über das Leistungsnetz, z.B. 230 V Leitung) haften meist ähnliche Probleme in den Bereichen freie Rechenleistung, Netzwerkmanagement, Kosten und vor allem der Störempfindlichkeit an, da die Stromverteilungssysteme nicht für „Kabel-Funk“ ausgelegt sind. Hinzu kommen thermische Probleme der Busknoten durch die notwendigen vor-Ort Netzteile, abgesehen von dem daraus resultierenden Standby-Stromverbrauch des Gesamtsystems beziehungsweise Gebäudes.

Dezentralisierung

Das Idealziel ist eine vollständige Dezentralisierung des Systems. Die Verfolgung dieses Ziels würde die Ausstattung jedes Schalters, jedes Relais, jedes Sensors und jeder Lampe mit einem eigenen (integrierten) Buskoppler erfordern. Nur dadurch wäre eine absolut regelmäßige Struktur der Busverdrahtung möglich. Dieses Ziel ist mit Hilfe des SmallCAN erreichbar. Neben einer universellen Erweiterbarkeit und Übersichtlichkeit bietet diese Struktur den Vorteil, dass keine Querverbindungen notwendig sind und sich jeder Busknoten auf seine spezialisierte Aufgabe beschränken kann. Multifunktionale Koppler mit installationsbedingt wechselnder Funktionszusammenstellung sind also nicht notwendig. Solch eine „Mehrfachnut-

zung“ ist zwar innerhalb eines Gerätes sinnvoll, da sich die Zusammenstellung der Teilfunktionen aus der (im Allgemeinen feststehenden) Gesamtfunktion ergibt. Im Gegensatz dazu wäre bei einer Zusammenfassung von funktional unabhängigen oder räumlich getrennten Funktionen während der Installation jedes Mal eine individuelle Anpassung erforderlich (Funktion, Kabelbelegung, elektrische Kompatibilität). Anders ausgedrückt: Jedes Gerät kann auf die Einschaltung oder Ansteuerung seines Verbrauchers optimiert werden (mechanisches Relais oder elektronisches Relais, Kleinrelais oder Leistungsrelais, PWM oder begrenzte Schalthäufigkeit, ...), daher wird die Ansteuerung direkt in das Gerät integriert, jedoch keine zusätzliche Hardware außerhalb des Gerätes mit einbezogen (keine Sternverdrahtung).

Je nach Grad des Ausbaus kann dies schnell zu hohen Teilnehmerzahlen führen (mehr als die bei Feldbussystemen sonst üblichen 32 bis 64 pro Segment), mit denen der SmallCAN spielend zurecht kommt (bis zu 1000 Knoten pro Bus). Ein besonderes Augenmerk sollte jedoch auf die spezifischen Kosten und den spezifischen Energiebedarf gelenkt werden.

Energieversorgung

Ähnlich wie bei batterieversorgten Systemen ist die Stromaufnahme von großer Bedeutung, da das Gesamtsystem mit bis zu 1000 Busknoten über das Datenkabel mit Strom versorgt werden soll (Maximalausdehnung 1000 m). Im Gegensatz zum Batteriebetrieb wird jedoch Hilfsenergie bereit gestellt, so dass auch aktive Sensoren und Leuchtanzeigen betrieben werden können (mit neusten Leuchtdiodentypen ist sogar eine Beleuchtung möglich).

Das Feldbussystem ist in sich geschlossen und potentialfrei ausgelegt, die Steuerung von Aktoren erfolgt daher galvanisch getrennt. Bei Aktoren ist zwar im Allgemeinen eine Energieversorgung in Form von 230 V~ oder 12 V= vorhanden, jedoch sind Transformatoren oder isolierende Wandler, die gleichzeitig effizient, klein, preiswert und zuverlässig sind, kaum möglich (von den Problemen einer möglichen direkten Netzspeisung ganz zu Schweigen). Systeme mit solchen „Power“ Konzepten lassen den Standby-Energiebedarf von Gebäuden schnell explodieren.

Somit sind die pro Busknoten zur Verfügung stehende Leistung sowie die Stromaufnahme begrenzt. Da die Datenübertragung im Basisband erfolgt, ist eine zentrale Bereitstellung der Energie zur Datenübertragung möglich (über eine Stromquelle), wodurch die Versorgung der Busknoten entlastet und eine einfache und preiswerte Busankopplung ermöglicht wird.

Buskoppler-Hardware

Die an die Hardware gestellten Forderungen lauten: Preiswert und klein, universell und geringe Stromaufnahme. Eingesetzt wird ein Mikrocontroller des Typs PIC16F88, der eine Weiterentwicklung des bekannten PIC16F84 darstellt. Neben der Stromversorgung sind die Busankopplung sowie optional einige LED-Anzeigen auf der Buskopplerplatine zusammengefasst. Es existiert lediglich eine Version des Buskopplers, wobei der LED-Teil bei Bedarf abgetrennt werden kann.

Der Buskoppler stellt einen vollständigen Busteilnehmer dar, lediglich Kabelklemmen und Gehäuse fehlen, sofern diese benötigt werden. Über die Steckerleiste ist ein problemloses Einsetzen in Bus-Applikationsadapter möglich, genauso wie in eigene Schaltungen oder Geräte. Durch diesen modularen Aufbau kann der eigentliche Kern, der Buskoppler, einheitlich gestaltet werden, die Applikationsadapter beschränken sich dann auf die wesentlichen, applikationsspezifischen Bauelemente.

Passend zu einer praktischen Auswahl an Gehäusen werden auch die zugehörigen Platinenlayouts bereit gestellt, so dass sich der Entwickler auf das Hinzufügen der Hauptkomponenten konzentrieren kann. Neben dem Standard- Sicherungsautomatengehäuse (Aufrastung wie ein Sicherungsautomat in Verteilerkästen auf TS35 möglich, Abbildung 3), stehen Versionen für Verteilerdosen (i12-Dose in Schutzart IP55, Abbildung 4),

Abbildung 3: BK auf TS35

Unterputzdosen (tiefe Dose für Lichtschalter und Steckdosen mit d=60mm) und EIB-Klemmdosen (Abbildung 5) bereit. Falls die Montage hinter einer Frontplatte oder Ähnlichem erfolgen soll, kann auch eine Lochraster-Version des Buskopplers direkt angeschraubt werden. Für spezielle Funktionen wie LCD Anzeigen oder wetterfeste Helligkeitssensoren sind besondere Gehäuse allerdings unabdingbar, auch dafür stehen Muster bereit.

Abbildung 4: BK in AP-Dose

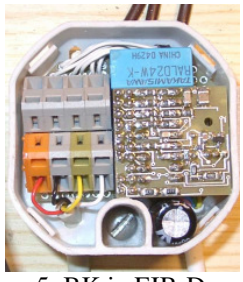

Abbildung 5: BK in EIB-Dose

Buskoppler-Softwarestruktur

Die Software des Buskopplers besteht aus modular zusammengesetzten Komponenten:
- Betriebssystem
- Standardfunktionen
- Interface [1]
- Erweiterten Standardfunktionen [1]
- Sonderfunktionen [1]
- Freie Sonderfunktionen [1]

Das Betriebssystem (BS) des Buskopplers enthält alle zur Buskommunikation notwendigen Routinen. Weiterhin wird die Schnittstelle zu den Sonderfunktionen bedient.

Standardfunktionen sind im Betriebssystem integriert und stellen grundlegende und häufig benötigte Fähigkeiten bereit wie den binären Zugriff auf den 8 Bit Hardwareport (Binär I/O), beispielsweise zur Relaisansteuerung oder Abfrage von Schalterkontakten mit Pegel- oder Flankendetektion.

Das Interface verbindet Betriebssystem und die zuladbaren Funktionen [1].

Erweiterte Standardfunktionen (eSTD) können bei Bedarf geladen werden und ermöglichen durch einfache Parametrierung die Nutzung der integrierten oder optionalen [2] Hardwareeinheiten wie PWM, ADU, UART, LEDs [2], bistabile Relais [2] und DC/DC-Konverter [2].

Sonderfunktionen (SF) steuern und kontrollieren die auf dem Applikationsadapter befindliche Hardware. An dieser Stelle kann die eigentliche Funktion des Busknotens beziehungsweise des Applikationsadapters programmiert werden, wobei wahlweise in Assembler (sehr effizient) oder in C (sehr komfortabel) gearbeitet werden kann. Hier steht ausreichend Rechenleistung für den Großteil der denkbaren Applikationen bereit, auch wenn es sich lediglich um einen 8/14 Bit Rechner handelt, der den Anforderungen eines Lichtschalters sicherlich gerecht wird!

Freie Sonderfunktionen (FSF) sind unabhängig von der

[1] Zuladbare Funktionen
[2] Optionale Hardwareeinheiten

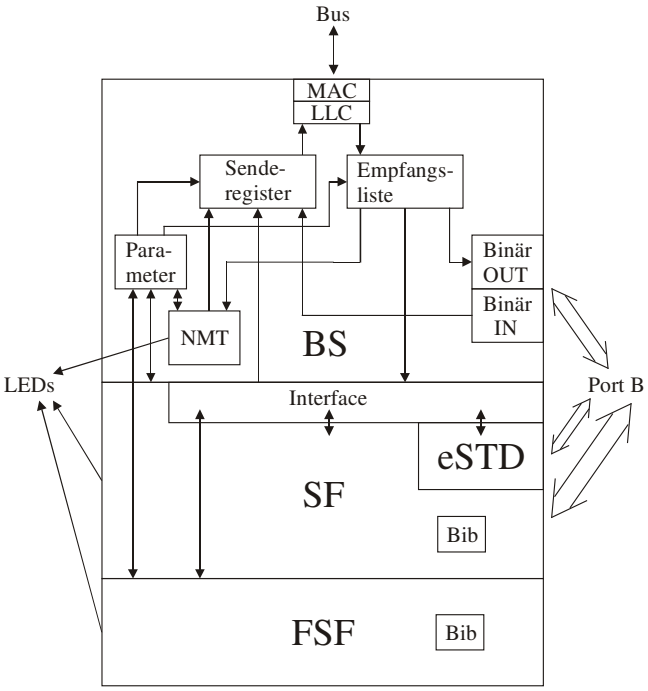

Abbildung 6: Interne Struktur eines Buskopplers

Hardware und führen bei Bedarf eine weitergehende und übergreifende Verarbeitung der per Bus bereitgestellten (Mess-)Daten in allgemeiner Form durch, wobei die „Echtzeitfähigkeit“ gewahrt bleibt. Beispiele sind Verknüpfung und Vergleich von Messwerten oder Schalterstellungen (gegebenenfalls unterschiedlicher Busknoten) mit Speicherung, Zählung oder Nutzung von Zeitgebern sowie Regelalgorithmen.

SmallCANsuite

Für die Konfiguration, Parametrierung, Visualisierung, Programmierung, Netzwerkmanagement, Internetanbindung und Applikationsverwaltung steht ein Software-Tool für zur Verfügung.

Bei der Entwicklung wurde Wert darauf gelegt, dass sowohl Windows- als auch Linux-Umgebungen genutzt werden können, daher wird das unter GPL veröffentlichte Framework Qt4.3 eingesetzt.

Die SmallCANSuite besteht aus zwei Teilen: Dem im Hintergrund laufenden SmallCANserver, der die Busanbindung über eine serielle RS232 Schnittstelle bereitstellt, und dem eigentlichen SmallCANtool, welches das Benutzerinterface (GUI) enthält und per TCP/IP auch (multiple) Zugriffe aus dem LAN/WAN erlaubt.

Zur Konfiguration der Busknoten stellt das SmallCANtool sowohl übersichtliche Dialoge als auch Detailansichten des Parameterspeichers bereit. Für die in die Busknoten geladenen (freien) Sonderfunktionen wird auf funktionsspezifische Plug-Ins (DLL/so) zurück gegriffen, so dass eine optimale und interaktive Einstellung sowie Test der jeweiligen Funktion möglich ist.

Prozessdaten können mit interaktiven Piktogrammen innerhalb einer Visualisierung grafisch dargestellt werden, wobei eine Anordnung der Elemente entsprechend der realen räumlichen Verhältnisse über einer (Grundriss-)Zeichnung oder einem Foto möglich ist. Per Mausbedienung können die realen Elemente gesteuert werden.

Ein Busmonitor ermöglicht die Verfolgung des Datenverkehrs, wobei dank der impliziten Semantik die Darstellung in einer lesbaren Form entsprechend der Datentypen erfolgt. Zusatztools ergänzen die Möglichkeiten zur Telegrammbearbeitung.

Ein umfangreiches Netzwerkmanagement zeigt Übersichts- und Detaildarstellungen aller Buskoppler inklusive Status und eröffnet Eingriffsmöglichkeiten über Steuerbefehle, Systemadressänderungen und Kopplerdeaktivierung.

Neben der Möglichkeit neue Software per Bus zu laden, kann auch ein kleiner Programmieradapter direkt am

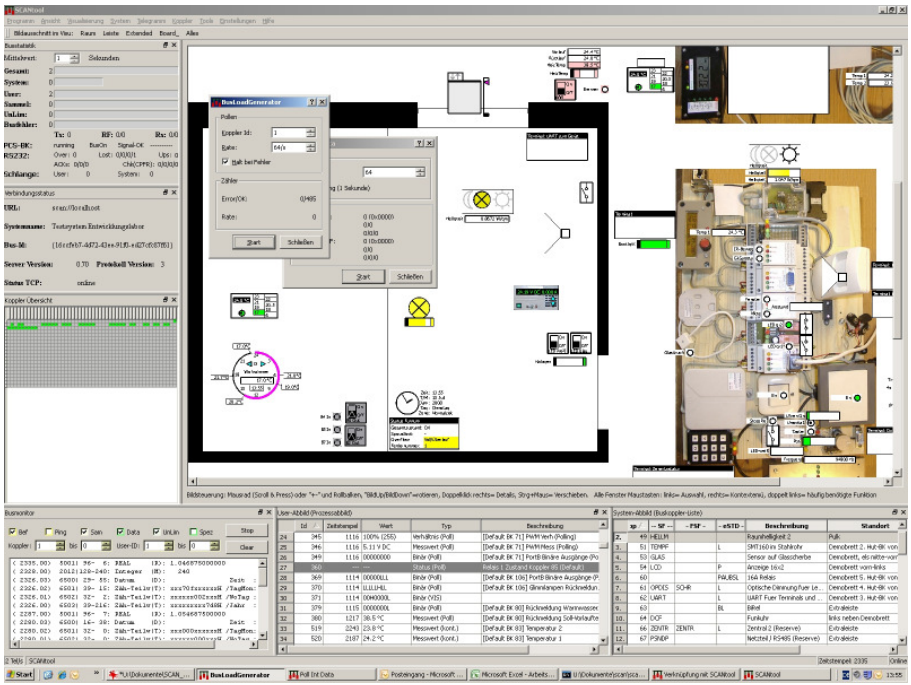

Abbildung 7: Screenshot des SmallCANtools

Rechner eingesetzt werden, wodurch insbesondere bei der Erstprogrammierung ein Aufspielen des Buskoppler-Betriebssystems möglich ist.

PC basierte Prozess-Applikationen können bei Bedarf geladen werden (PSF) und ermöglichen eine weitergehende und übergreifende Bearbeitung der Daten, ähnlich der freien Sonderfunktionen. Eine „Echtzeitfähigkeit“ im herkömmlichen Sinne kann hierbei jedoch nicht gewährleistet werden, im Gegensatz zu den freien Sonderfunktionen der Buskoppler (ein Mini-PC kann für diesen Fall zumindest eine durchgehende Verfügbarkeit bieten). Im Gegenzug sind jedoch viel umfangreichere Möglichkeiten in Bezug auf Komplexität, Speicherplatz (Nutzung der Festplatte), grafisches Benutzerinterface und Informationsvernetzung (Nutzung des Internet) vorhanden. Hier besteht die Möglichkeit maßgeschneiderte Lösungen laut Anforderungsprofil sowie Benutzerinterfaces nach Wunsch des Anwenders zu Implementieren.

Zusammenfassung

Ein vollständiges Feldbussystem vom physikalischen Medium bis zur spezifischen, grafischen Applikation wird offen zur Verfügung gestellt:

-Anzahl: Bis 1000 Teilnehmer
-Leitungslänge: Bis 1000m
-Kabel: Standard-Telefon (2x2x0,6)
-Telegramme/Sekunde: 138
-MAC: Multimaster, CSMA/CA
-Aufbau: Modular mit vereinheitlichtem Buskoppler
-Energieversorgung: Integriert, geringer Gesamtverbrauch, Hilfsenergie wird angeboten
-Netzwerkmanagement: Integriert
-Bauform: Standardgehäuse/PlugIn
-Betriebssystem: Unter Dual Lizenz
-Hardware: Wenige Standardelemente
-Software SmallCANsuite: GPL
-Programmierung: MiniProg oder Bus
-Fernzugriff: TCP/IP

Aufbau eines Systems

Es steht bereits eine Reihe von fertigen Applikationsadaptern zur Verfügung, die ein schnelles Einarbeiten ermöglichen. Alle bieten umfangreiche Hardware-Überwachung:
- Funkuhr (DCF77)
- Helligkeitssensor (30 Bit Auflösung)
- Dimmbares EVG (DALI und DSI)
- Raumthermostat
- Rolladensteuerung
- Temperaturmessung mit 4x SMT160
- VDS-Glasbruchsensor
- Netzteilfernsteuerung (RS485)
- Elektronisches Relais (2x8A/230V~) mit Phasen-/Paketdimmer , überwacht
- Relaisausgänge (2x 16A/230V~) (4x 2,3A/40V=) (8x Kurzschlussfest 25W/230V~) (und weitere)
- Eingangssignale (3x 230V~) (4x 4-40V=)
- Optischer Distanzsensor (GP2D02!)
- Frequenzmessung
- RS232 für Terminals (Handshake)
- PC Koppl. und Einspeisung (Zentral)
- LCD-Anzeige (bis 4x20 Zeichen)
- 10er Tastatur
- Heizungskessel-Ansteuerung

Software, Dokumentation und Layouts stehen zum Download bereit (Beta-Version) [2]. Zur Beschaffung der Hardware ist es derzeit noch notwendig, die Komponenten über die bekannten Elektronikversender zu beziehen (siehe Stückliste). Die Layoutdateien können per Mail zu einem PCB-Pool geschickt werden. Eine alternative Bereitstellung fertig bestückter Platinen wird angestrebt.

Die Materialkosten für einen Buskoppler liegen in kleinen Stückzahlen bei etwa 15 €. Die Kosten für Gehäuse, Klemmen und applikationsspezifische Leitplatten sind von gewünschten Einsatz und dem „Gesamtgerät“ abhängig.

Ausblick

Die Entwicklungsphase des Systems ist, obgleich einsatzbereit, noch nicht abgeschlossen: Software und Dokumentation unterliegen einer ständigen Weiterentwicklung. Insbesondere wird angestrebt, die Vielfalt der zur Verfügung stehenden Applikationsadapter weiter zu erhöhen.

Hier sind insbesondere die Leser zum Mitmachen aufgefordert, um dieses „Open Hardware“ Projekt zum Erfolg zu führen. Entsprechende Einsendungen sind jederzeit willkommen und werden dem OpenSource Pool hinzugefügt.

Um eine dauerhafte Internet-Anbindung ohne PC zu ermöglichen, ist die Entwicklung eines preiswerten Minirechners zur Aufnahme des SmallCANservers und der Systembeschreibungsdatei geplant. Dadurch würde jederzeit ein bequemer TCP-Zugriff möglich werden.

Literatur und Links

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

[2] www.openbus.org

[3] Scherer, T.: *Warten auf ZigBee.* Elektor 9/2007, Seite 60ff. Elektor-Verlag GmbH, Aachen.

SmallCAN

Teil 2: Theorie und technische Details

Nachdem im ersten Teil die praktische Nutzung des integralen Feldbussystems beschrieben wurde, wird hier im zweiten Teil auf die zugrunde liegenden theoretischen Hintergründe und technischen Details eingegangen.

Beim Entwurf des Systems wurde ein Übertragungsprotokoll entwickelt, dem die im Folgenden beschriebenen Betrachtungen bezüglich Datenrate, Paketgröße, Adressraum und Medienzugriff zugrunde liegen. Zusätzliche Überlegungen beziehen sich auf die Aufteilung der Buskapazität, das Netzwerkmanagement und Details der Hilfsenergienutzung.

Weiterhin wird eine Einordnung in das bekannte ISO/OSI-Schichtenmodell vorgenommen sowie eine kurze Beschreibung der internen Schnittstellen und Bibliotheken gegeben.

Datenrate und Paketgröße

Für den vorgesehenen Anwendungszweck ist eine relativ geringe Datenrate ausreichend, da im Wesentlichen kurze Informationsblöcke mit geringer Häufigkeit zu übertragen sind (ein Lichtschalter nutzt beispielsweise lediglich ein Bit und es ist nicht damit zu rechnen, dass dutzende von Lichtschalter innerhalb eines Gebäudes gleichzeitig betätigt werden), auch Messwerte wie (Raum-) Temperaturen oder Helligkeitswerte sind gelegentlich zu übertragen. Entscheidend für das System ist somit weniger die Datenrate als vielmehr die Anzahl der pro Sekunde übertragbaren Telegramme beziehungsweise Ereignisse.

Im Gegensatz dazu treten bei Geräten der Computer- und Unterhaltungsindustrie große Datenmengen auf. Da solche Geräte im Preisbereich zwischen 100 € bis 1000 € angesiedelt sind, spielen deren Netzwerkkosten keine entscheidende Rolle, so dass man sich dabei auch eine kostenintensive sternförmige Verdrahtung leisten kann.

Aus diesen Überlegungen folgt, dass für den hier anvisierten Anwendungszweck eine Datenbreite von 16 Bit völlig ausreichend ist, die aus Gründen einer einfacheren Implementierung konstant gehalten wird. Sollte der seltene Fall eintreten, dass größere Datenbreiten benötigt werden, beispielsweise zur Übertragung des Zählerstandes eines Stromzählers, so können diese Information auf mehrere Telegramme verteilt werden (die Übertragungshäufigkeit kann dabei begrenzt werden, da die Ablesung von Zählern im Allgemeinen nur in größeren Zeitabständen erfolgt).

Adressraum

Der erforderliche Adressraum wird durch die maximale Anzahl der Busknoten und der Datenobjekte bestimmt: Es bietet sich ein 16 Bit Adressraum an, analog zur Datenbreite. Damit ist eine Systemadressierung der Buskoppler mehrerer Segmente möglich, gleichzeitig steht ein großzügiger Nutzerbereich zur Verfügung. Konkret sind bis zu 1000 Teilnehmer pro Segment vorgesehen, so dass sich ein Segment auf ein gesamtes Einfamilienhaus oder eine Etage eines Hochhauses erstrecken kann. Der Nutzerbereich ist mit 32767 möglichen Adressen großzügig ausgelegt, da pro Busknoten durchaus mehrere Mess- und Statuswerte bereitgestellt werden können.

ISO/OSI-Schichtenmodell

Kommunikationssysteme werden üblicherweise in das so genannte ISO/OSI-Schichtenmodell eingeordnet, darauf wird hier im Weiteren Bezug genommen. Das Modell besteht aus 7 aufeinander aufbauenden Schichten (siehe auch [3]):

7: Application-Layer
6: Presentation-Layer
5: Session-Layer
4: Transport-Layer
3: Network-Layer
2: Datalink-Layer (MAC & LLC)
1: Physical-Layer

Medienzugriffsverfahren

Als Übertragungsmedium wird ein echter physikalischer Bus eingesetzt (Physical-Layer), bei der Zugriffskontrolle (MAC) wird das CSMA/CA Verfahren genutzt, welches durch bitweise Arbitrierung Kollisionen vermeidet und so in allen Fällen die vollständige Nutzung der gesamten Übertragungskapazität des Übertragungsmediums ermöglicht. Weiterhin ist dieses Verfahren multimasterfähig und durch die Wahl der Adressen wird gleichzeitig eine Priorisierung der Telegramme untereinander ermöglicht, so dass zeitkritische Signale bevorzugt behandelt werden können.

Eine geringe Busbelastung wird dadurch erzielt, dass lediglich Ereignisse und veränderte Werte übertragen werden. Dabei wird von einem nicht periodischen, zufälligen Auftreten ausgegangen, so dass ein stochastisches Zugriffsverfahren wie CSMA/CD angebracht ist. Funktionsrückmeldungen werden lediglich auf Anfrage geliefert, da im Normalfall von einer korrekten Verarbeitung der Information ausgegangen wird, sofern eine Übertragungsbestätigung vorliegt, der Empfänger funktionsfähig ist und keinen Fehlerstatus meldet.

Um die Ausfallsicherheit zu erhöhen, schaltet ein Buskoppler automatisch seinen Daten-Ausgangstreiber ab, wenn die Fehlerrate der selbst verursachten Fehler eine voreingestellte Grenze überschreitet, insbesondere wenn keinerlei Antwort registriert wird (Babbling Idiot Problem). Zusätzlich ist eine Hardwareabsicherung vorgesehen, die statische Pegel vom Ausgangstreiber fernhält (bei Absturz oder defekt des Mikrokontrollers).

Netzwerkmanagement

Ein integriertes Netzwerkmanagement (NMT) ermöglicht den Fernzugriff auf EEPROM Konfigurationsspeicher und Flash-Programmspeicher. Es werden automatisch Statusmeldungen generiert, sobald sich der Betriebszustand eines Busknotens verändert. Gemeldet werden Speicherfehler, Probleme mit der Signalqualität der Datenübertragung, busbedingter Datenstau, zeitweiser Kurzschluss, interne Timingfehler und Ähnliches. Auch das Auftreten von Telegrammfehlern wie Rahmen- und CRC-Fehler kann abgefragt werden.

Da der Versand der Statusmeldungen automatisch erfolgt, ist ein Mechanismus implementiert der verhindert, dass bei potentiell globalen Fehlern wie Datenstau oder Signalqualität eine Überflutung mit Statusmeldungen erfolgt. Als Gegenmaßnahme wird bei massenhaftem Auftreten von solchen Statustelegrammen ein busweiter Versandabbruch ausgelöst.

Weiterhin erfolgt eine Meldung, wenn es zu Telegrammkollisionen aufgrund doppelt vergebener Systemadressen kommt. Der zentrale Teil des NMT kann diese Konflikte unter Zuhilfenahme der Buskopplerseriennummern automatisch auflösen und die Adressen der betroffenen Buskoppler ändern.

Transport und Präsentation

Die Funktionalität von Network- und Session-Layer wird innerhalb eines Bussegmentes nicht benötigt. Transport- noch Presentation-Layer werden von Feldbussystemen traditionell nicht genutzt, die entsprechende Funktionalität wird dabei in den Applikation-Layer verbannt. Trotzdem ist dies Funktionalität vorhanden und wird hier daher explizit unterstützt: Im Buskoppler stehen Bibliotheken und Betriebssystemunterstützung bereit, im SmallCANtool sind sie als eigene Schicht implementiert.

Zum Transportbereich zählen im Wesentlichen die Paketierung von Zählern (Datenbreite größer 16 Bit) und Texten (2 Zeichen pro Telegramm) sowie im Systemkanal die Behandlung von Datenblöcken (EEPROM, Systeminformation, Programmcode).

Zur automatischen, korrekten Präsentation der Daten werden „ungenutzte“ Bit im Datenbereich der Bustelegramme verwendet. Somit können die Daten, ohne Kenntnis der aktuellen Buskonfiguration und ohne zusätzliche Buslast durch die Übertragung von statischen Informationen zu erzeugen, jederzeit in einer korrekten Art und Weise strukturiert dargestellt werden. Es kann also nicht vorkommen, dass im Busmonitor zusammenhangloser Zeichensalat angezeigt wird. Die Funktionen innerhalb der Buskoppler können anhand dieser Zusatzinformation die Kompatibilität der eingehenden Daten überprüfen. Dadurch wird beispielsweise verhindert, dass Relais durch (zufällige) Bit eines Messwertes oder einer Textausgabe angesteuert werden oder dass unsinnige Vergleiche zwischen verschiedenen Datentypen durchgeführt werden. Allerdings bezieht sich diese implizite Semantik lediglich auf die Datentypen, das heißt Ursprung, Ziel, Zweck und gegebenenfalls physikalische Einheit können hier nicht enthalten sein, da lediglich überzählige Bit genutzt werden.

Aufteilung der Buskapazität

Unter gleichzeitiger Berücksichtigung aller vorgenannten Randbedingungen lassen sich 138 Telegramme pro Sekunde übertragen [1].

Für Managementaufgaben sind Systemtelegramme hoher Priorität vorgesehen, wodurch Konfigurations-, Überwachungs- und Korrektureingriffe jederzeit möglich sind. Sogar ein Programm-Upload von neuer Software ist im laufenden Busbetrieb möglich! Um eine Störung des regulären Datenverkehrs zu vermeiden, erfolgt eine Begrenzung der maximalen Systemtelegrammrate auf 25 Telegramme pro Sekunde (beziehungsweise Mindestabstand von 40ms), die Übertragungskapazität wird somit in zwei virtuelle Kanäle für System und User aufgespalten (siehe Abbildung 1).

Im Userbereich wird das so genannte Objektdatenmodell gentutz, das heißt jeder Versandadresse ist genau ein (Mess-)Wert eindeutig zugeordnet, der sich im Laufe der Zeit verändern kann (abweichend kann es sich auch um einen Datenstrom handeln).

Als Sicherung gegen allzu kommunikative Busknoten ist im Userbereich eine Begrenzung der Telegrammrate auf 35 Tel/Sek pro Busknoten vorgesehen. Um jedoch die ungenutzte Restkapazität des Busmediums durch einzelne Busknoten voll ausschöpfen zu können, beispielsweise zur Übertragung von seriellen Datenströmen, wird diese Begrenzung im untersten Adressbereich aufgehoben (unlim-Bereich). Möglich sind dort dann bis zu 126 bidirektionale Verbindungen, optional können dabei ein serieller Handshake über den Bus geführt sowie ein Load-Balancing aktiviert werden.

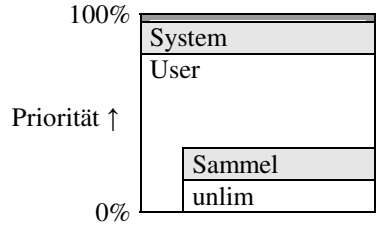

Abbildung 1: Zuordnung Prioritäten und Kapazitätsverteilung

Das obere Ende des unlim-Bereiches kann auch durch Systemtelegramme genutzt werden, so dass bei Managementzugriffen ebenfalls eine Nutzung von freien Restkapazitäten erfolgen kann, insbesondere beim Auslesen eines Parameterspeichers (EEPROM-Daten) oder einer allgemeinen Überprüfung der Funktionsbereitschaft (Sammel-Ping).

Bereitstellung von Hilfsenergie

Durch die gemeinsame Führung von Energie und Daten in einem Kabel ist es notwendig, schnelle Änderungen der Stromaufnahme zu vermeiden. Hilfsenergie wird daher hauptsächlich durch Serienschaltung mit dem Buskoppler bereit gestellt (11 V bei 2 mA), was für (geschaltete) Anzeigen und Beleuchtungen ebenso ausreichend ist wie für andere Zwecke und keinen zusätzlichen Strombedarf nach sich zieht. Auch die stabilisierte 5 V Versorgungsspannung kann direkt genutzt werden, sofern die mittlere Stromaufnahme etwa 0,6 mA nicht überschreitet. Für stromintensive Anwendungen können über einen Mini DC/DC-Konverter bis zu 60 mA bei 1,5 V oder 20 mA bei 5 V entnommen werden, was jedoch lediglich für Sonderfälle zulässig ist.

Zur Senkung der mittleren Stromaufnahme ist es meistens ausreichend, Sensoren nur kurzzeitig für die Dauer des Messvorganges zu aktivieren, beziehungsweise existieren häufig sparsame Sensortypen (wie beispielsweise der Temperatursensor SMT160 mit 0,2 mA Stromaufnahme). Elektrische Ausgänge werden im Allgemeinen über Optokoppler oder mechanische Relais geführt, wobei sich der Strombedarf von Optokopplern mit geringer Schaltfrequenz durchaus auf 0,1 mA reduzieren lässt, bei mechanischen Relais sorgen bistabile Ausführungen (bis 16A/230V erhältlich) für eine fast leistungslose Ansteuerung.

Parametrierung

Alle logischen Verbindungen werden über die Vergabe der IDs hergestellt. Dabei gilt der Grundsatz, dass jeder Sensor beziehungsweise jeder Messwert eine exklusive Versand-ID erhält (dies wird auf verarbeitende Funktionen entsprechend ausgedehnt) und lediglich bei Veränderungen gegenüber dem vorhergehenden Wert beziehungsweise beim Eintreten von Ereignissen ein Versand ausgelöst wird. Die ID bestimmt gleichzeitig die Priorität beim Busversand, hohe IDs werden bevorzugt. Alle Empfänger, meist Aktoren wie Relais (dies wird ebenfalls auf verarbeitende Funktionen ausgedehnt), hören alle von den Sendern kommenden Telegramme mit und entscheiden anhand einer internen Empfangsliste, welche dieser Datenobjekte zur Ausgabe oder Verarbeitung genutzt werden sollen. Beim Empfang wird auch das busknoteninterne Ziel bestimmt (die meisten Busknoten enthalten mehrere Empfangs-Funktionen) sowie zusätzliche interne Parameter festgelegt.

Zusätzlich zu den grundlegenden Datenobjekten sind Statusmeldungen und Rückmeldungen vorgesehen. Hardware-Statusmeldungen geben Auskunft über den Zustand der angekoppelten Hardware, Rückmeldungen ermöglichen eine Abfrage des Ist-Zustandes der Aktoren. Der Istzustand kann vom Sollzustand abweichen, beispielsweise bei Motoren um nennenswerten Umlaufzeiten. Im allgemeinen Fall werden Rückmeldungen jedoch lediglich auf Anforderung versandt, um der optionalen Visualisierung eine korrekte Darstellung zu ermöglichen.

Da pro Funktion im Normalfall lediglich zwei Sendeobjekte vorgesehen sind, können für besondere Fälle Versandblöcke mit bis zu 8 Sub-IDs pro Kanal eingesetzt werden. Dies ermöglicht den Versand von (gleichartigen) Objekten wie beispielsweise mehreren Temperatursensoren an einem Warmwasser-Solarspeicher oder Objekten mit hoher Auflösung wie Stromzähler, Wasserzählern und ähnlichen.

Sollen verschiedenartige Objekte oder mehrere kontinuierliche Messwerte auf den gleichen Aktor beziehungsweise Eingang wirken, so sind Mixer-Funktionen (freie Sonderfunktion) als verarbeitende Instanzen zwischen zu schalten. Diese enthalten, neben der Empfangsliste, ein auf den jeweiligen Anwendungszweck abgestimmtes Verfahren zur Bestimmung eines „Mischwertes“.

Interne Software-Schnittstelle zwischen Betriebssystem und Funktionen

Die Neuerstellung von (freien) Sonderfunktionen soll eine schnelle Anpassung von Hardwarekomponenten nebst der entsprechenden Datenaufbereitung und Verarbeitungsfunktionalität ermöglichen. Daher ist, neben der Vorgabe eines Quelltext-Grundgerüstes, die Form der Softwareschnittstelle zum Betriebssystem von Bedeutung.

Nach Anmeldung werden diverse interne Funktionsaufrufe bereitgestellt:
- Initialisierung und Systemstart
- Verschiedene Timer im Sekunden und Millisekundenbereich
- Interruptroutine (periodisch, µs)
- Zyklische Verarbeitung (Schleifen)
- Telegrammempfang
- Ausführung des Telegrammversandes und erfolgreicher Versand
- System-Stopp (Reset/ Power-Down)

Ein Versandwunsch wird durch einfaches Setzen eines Bit angemeldet, beispielsweise innerhalb der zyklischen- oder Timersegmenten, das Betriebssystem initiiert dann entsprechend der Adressprioritäten der Sendeliste den Versand mit der parametrierten Adresse und ruft die Routine „Telegrammversand“ auf. Ein Telegrammempfang wird anhand der Empfangsliste vom Betriebssystem erkannte und per Funktionsaufruf „Telegrammeingang“ gemeldet. Altnativ besteht die

Möglichkeit einer eigenen Implementierung von Versand- und Empfangsroutinen (Mega-Sonderfunktion).

Weiterhin kann zur Überwachung der Hardwarekomponenten ein Hardwarestatus ermittelt und per Statustelegramm automatisch gemeldet werden, die Auswahl des gewünschten Verhalten der 8 Bit des Hardwarestatus-Registers erfolgt per „#define“. Durch die Nutzung des Includes „HWSTAT“ kann somit ein integriertes Diagnosesystem schnell aufgebaut werden.

Bibliotheken

Es steht eine Reihe von Bibliotheken zur Verfügung, die eine zügige Programmierung erleichtern sollen und Systemspezifika kapseln.

Insbesondere das interne Rechnen mit großen Datenbreiten erfordert auf einem 8 Bit Rechner einen gewissen Aufwand. Da generische Rechenfunktionen große Laufzeiten nach sich ziehen, wurde ein Registermodell mit 4 Rechenregistern zu je 64 Bit implementiert, auf dem spezialisierte Operationen wählbarer Bitbreite operieren (Multiplikation, Division, Addition, Subtraktion, Vergleich, ...).

Zur unkomplizierten Behandlung von Messwerten sind umfangreiche Funktionen wie „Messwertaufbereitung“ vorhanden. Auch systemspezifische Funktionen wie Speicherzugriff, Datentypkonvertierung und Versandpaketierung stehen zur Verfügung, ebenso wie eine wachsende Liste von Funktionen zur Hardwareansteuerung wie beispielsweise LCD-Controller (inklusive ASCII-Steuercodes) oder DALI beziehungsweise DSI-Interfaces.

Schnittstellen im Small-CANtool

PC-basierte Applikationen im SmallCANtool (PSF) sowie Visualisierungselemente greifen über eine definierte Schnittstelle auf die Daten zu. Diese Applikations-Schnittstelle wird durch einen „Skalierungsverteiler“ angeboten, der das Prozessabbild bereit hält und die Applikationen über „Signals &Slots“, ein Konzept des Qt-Frameworks, über neu eingehende Telegramme informiert. Diese Instanz ist gleichzeitig für Transport (Paketierung) und Präsentation zuständig. Anhand einer XML-Systembeschreibungsdatei werden, soweit möglich, zusätzliche Angaben wie Bezeichnung, physikalische Einheit, Skalierung, Ortsangabe bereitgestellt und verarbeitet.

Minimalsystem

Zum Aufbau eines Minimalsystems werden mindestens zwei Busknoten benötigt, wobei eine zentrale Energieeinspeisung mit integrierter PC Kopplung obligatorisch ist (für erste Versuche kann alternativ ein 1 kOhm Widerstand verwendet werden) [2].

Eigenerstellte Applikationsadapter

OpenSource Projekte leben von der Mitarbeit engagierter Nutzer. Neu erstellte Applikationsadapter und Funktionen sollten daher dem OpenSource Pool zugeführt werden. Vollständige Unterlagen sollten im Sinne der Wiederverwendbarkeit und Zuverlässigkeit angestrebt werden, das heißt neben dem Quelltext der Funktionen und einem Platinenlayout sollten auch die Dokumentation (laut Vorlage) sowie eine DLL/so zur Konfiguration erstellt werden.

Literatur und Links

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

[2] www.openbus.org

[3] de.wikipedia.org/wiki/OSI-Modell