

Entwurf und Implementierung von CANopen Geräten

Rüdiger Härtel, Torsten Gedenk

port GmbH

Übersicht

Das CANopen-Protokoll findet große Anwendung in industriellen Maschinen. Mehrere Firmen bieten eine Bibliothek in Form von Quelltext oder vorkompilierten Objektcode bzw. DLL an, die das Protokoll konform zu den Standards des CiA und ISO EN 50325-4 implementiert. Die ersten Schritte mit einer neuen Technologie sind sehr zeitaufwendig besonders, wenn direkt auf einer neuen Zielhardware entwickelt werden soll.

Graphische Werkzeuge/Tools können die Entwicklungsingenieure besonders in dieser ersten Phase bei der Erstellung des Objektverzeichnisses unterstützen als auch den Quelltext in der Programmiersprache C für die Initialisierung der Bibliothek und die Dokumentation als EDS und HTML erzeugen.

Die Implementierung wird vereinfacht und ist besonders effizient, wenn auf einem Desktop-PC mit einem Betriebssystem wie Windows oder Linux begonnen wird, anstatt direkt auf der Zielhardware. Dann entfällt das Programmieren der Firmware und man kann alle Vorzüge eines modernen Debuggers nutzen. Die Übertragung der Firmware vom Betriebssystem auf die Zielhardware erfolgt dann durch Austauschen des CAN-Treibers und einer erneuten Kompilierung.

Dieser Artikel zeigt, dass ein Werkzeug, welches das Kommunikationsverhalten und das Objektverzeichnis auf einer hohen Beschreibungsebene realisiert Zeit und Kosten für die Realisierung eines CANopen Projektes stark reduziert werden können.

Einleitung

Standardisierte Anwendungsprotokolle bieten mehrere Vorteile, wenn ein Kommunikationsinterface implementiert werden soll, wie z.B. ein durchdachtes Konzept für den Zugriff auf Variablen von aussen, Trennung der Kommunikation in unterschiedliche Dienste und fertige, getestete Protokollbibliotheken. Auf der anderen Seite bedeutet die Verwendung einer fertigen Protokollbibliothek eine erhöhte Einarbeitungsphase und erhöhte Kosten bei der Realisierung der ersten Anwendung.

Dieser Artikel beschreibt das DesignTool, das den Geräteentwickler bei Entwurf, Implementierung und Dokumentation eines CANopen Gerätes unterstützt. Es zeigt wie ein Entwickler durch das DesignTool bei der Realisierung unterstützt werden kann. Außerdem wird auch kurz auf die notwendigen Daten für das DesignTool eingegangen.

Die Anforderungen, die in der Entwicklung des DesignTools gestellt wurden, sind hier kurz aufgezählt.

- Interaktive Bedienung
- Projekte erstellen, ändern, speichern
- Import/Export von EDS-Dateien

- Unterstützung durch Datenbanken für verschiedene Geräteprofile
- Unterstützung für die Konfiguration des CAN-Treibers durch eine Hardware Datenbank
- Erzeugung von Initialisierungsfunktion, EDS-Datei und Dokumentation

Für die Spezifikation und die spätere Implementierung des DesignTools waren unterschiedliche Aspekte zu beachten:

- Erstellung einer *intuitiv* zu bedienenden graphischen Oberfläche
- Datenmodell
- Speicherformat der Datenbanken
- Erzeugung des Quelltextes

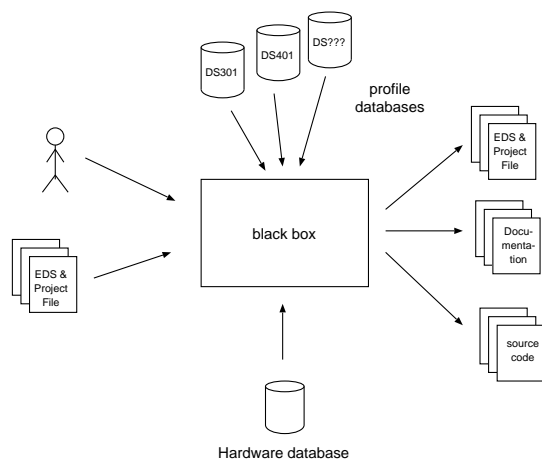


Abbildung 1: Prinzip

Graphische Oberfläche

Die Bedienschnittstelle wurde übersichtlich gestaltet. Alle notwendigen Daten wurden funktionell gegliedert und sind leicht erreichbar. Die dargestellten Daten bestehen aus den verschiedenen Teilen:

- EDS Daten und Dokumentation
- Objektverzeichnis
- Aktivierung von Protokolldiensten
- Optimierungseinstellungen für die CANopen-Bibliothek und

- Hardwareeinstellungen

CANopen benutzt einen objektorientierten für die Anordnung und den Zugriff auf Kommunikations- und Prozessvariablen. Auf die Objekte kann mittels Index und Subindex zugegriffen werden. Alle Variablen, die von außen verändert werden sollen, werden in dem Objektverzeichnis abgelegt. Die Struktur des Objektverzeichnisses wird anhand eines Baumes dargestellt, wobei jedem Knoten ein Index und jedes Blatt ein Subindex entspricht. Die Indizes wurden entsprechend ihrer Funktion unterteilt, um die Übersicht zu verbessern.

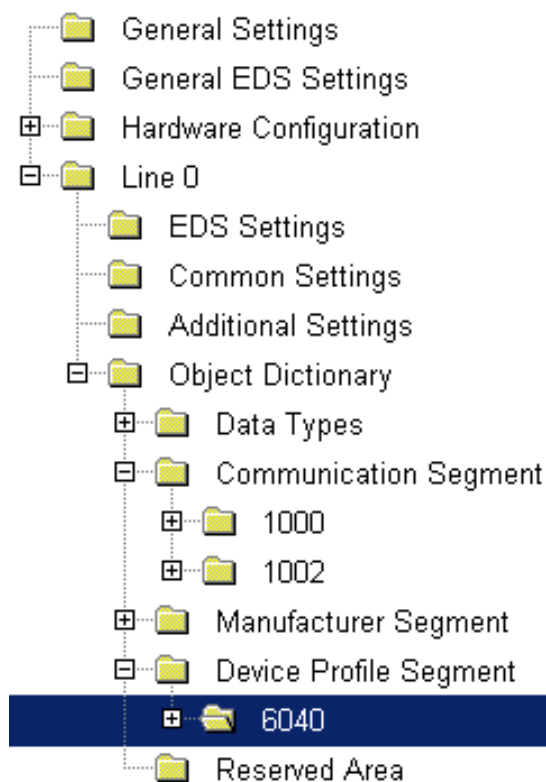


Abbildung 2: Objektverzeichnis als Baumansicht

Weitere Knoten für die Hardwareeinstellungen und EDS-Daten wurden hinzugefügt.

Mit dem Konzept von Objekten können alle möglichen Arten von Daten, wie einfache Datentypen, Strukturen und Felder erzeugt werden. Zusätzlich zu dem

eigentlichen Wert des Objektes, besitzt es weitere Attribute, wie Zugriffsrechte und oberer und unterer Grenzwert.

Diese Attribute werden in einer Art Editor angezeigt werden. Auf der einen Seite müssen durch diesen Editor alle Werte eines Objektes verändert werden können, das heißt für jeden vom CANopen-Standard vorgeschriebenen Wert muss ein Eingabefeld vorhanden sein. Auf der anderen Seite muß der Editor es ermöglichen, die Werte entsprechend der Funktion des Objektes einzugeben, um so von dem Protokoll zu abstrahieren und eine einfache Eingabemöglichkeit zu bieten. Der Editor wurde deshalb in eine Strukturansicht und eine Maskenansicht unterteilt. Während die Strukturansicht für jeden Index und Subindex gleich ist, ist Maskenansicht abhängig von dem ausgewählten Index. Für das Kommunikationsprofil DS301 sind Maskenansichten für die folgenden Indizes vorbereitet:

Index	Beschreibung
1000h	Device type
1006h	Communication cycle period
1007h	Synchronous window length
100Ch	Guard time
1016h	Consumer heartbeat list
1017h	Heartbeat producer time
1028h	Emergency consumer list
1029h	Error behaviour object
1200-12FFh	SDO comm. param.
1300-1340h	SRDO comm. param.
1340-13CFh	SRDO mapping
1400-15FFh 1800-19FFh	PDO comm. param.
1600-17FFh 1A00-1CFFh	PDO mapping
1FA0-1FCFh	Object scanner list
1FD9-1FFFh	Object dispatcher list

Table 1: Indizes mit Maskenansicht

Maskenansichten können auch für Indizes aus Geräteprofilen erstellt werden. Um eine spätere Erweiterbarkeit zu ermöglichen, wurden die Masken nicht fest in das Programm kodiert, sondern sind in den Datenbanken definiert.

Als eine dritte Editoransicht werden Optimierungen für das Objekt angeboten. Die Optimierungen beziehen sich auf die CANopen-Bibliothek von port. Optimierungseinstellungen beziehen sich auf:

- die Indexvariable und
- den damit verbundenen CANopen Dienst ¹
- Speicherklasse
- bestehende Variablen verwenden

Um Speicherreduzierungen durchführen zu können, kann die Speicherklasse jeder Variablen geändert werden. Eine Platzierung in ROM oder RAM

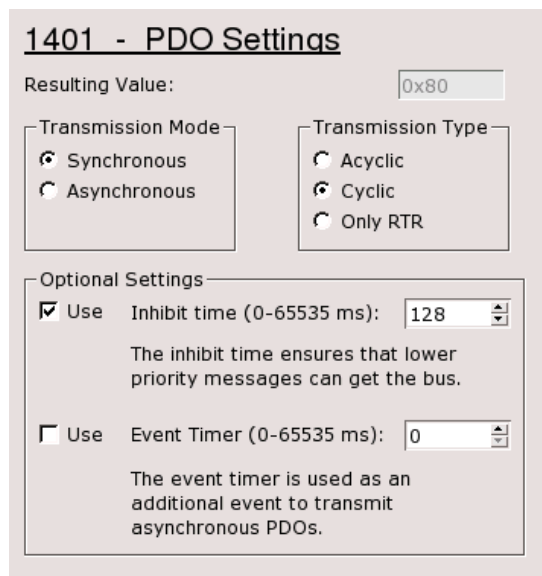


Abbildung 3: Maskenansicht für Index 1401

Segmenten ist möglich. Die Verwendung einer Variablen aus einer bestehenden Anwendung erlaubt es, die bisherige Kommunikationsschicht auszutauschen und durch das CANopen-Protokoll zu ersetzen.

Die Optimierung von CANopen-Diensten wirkt sich immer auf den Speicherverbrauch (RAM, ROM) der CANopen-Bibliothek aus. Der SDO-Dienst bietet unterschiedliche Übertragungsmodi (expedited, segmented, block). Für den PDO-Dienst kann dynamisches Mapping oder RTR aktiviert werden. Das sind zwei Beispiele für Optimierungen. Jeder Kommunikationsdienst hat seine eigenen Optimierungseinstellungen.

Um diese drei Ansichten gut anzuordnen und einfach zwischen diesen zu wechseln wurde das graphische Element Reiter/Register oder Tabs benutzt. Mittels Reiter können die Ansichten gut gruppiert werden, und die Übersicht wird nicht durch unterschiedliche, offene Dialoge gestört. Die Reiter wurden neben der Baumansicht angeordnet, so dass folgendes Bild entsteht.

¹ nur zutreffend für Kommunikationsparameter

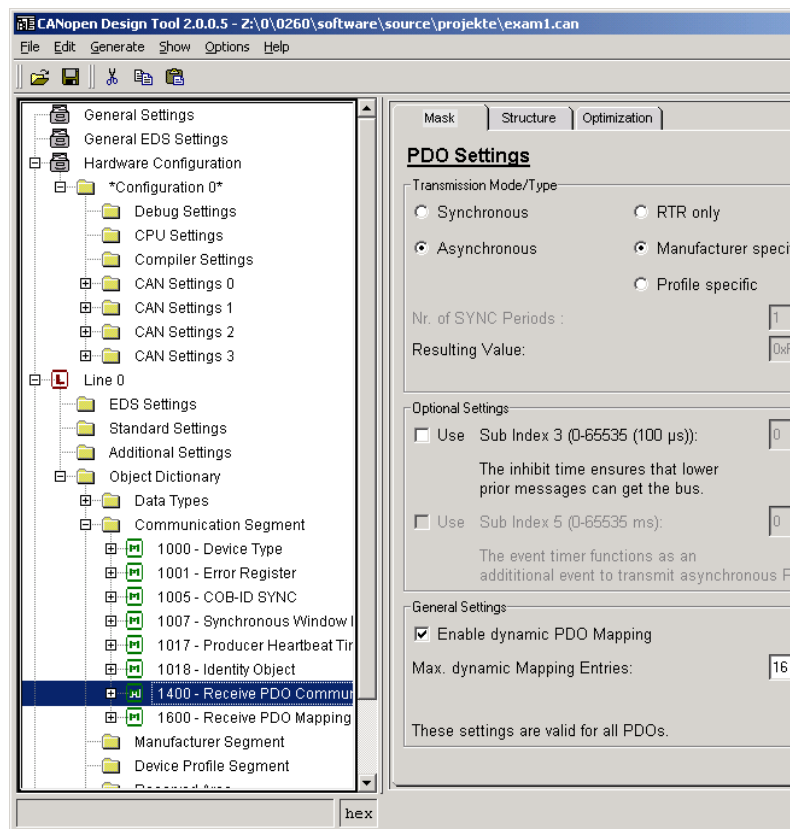


Abbildung 4: komplette Oberfläche.

dem Typ VARCHAR². Das Datenfeld für die Maskenansicht ist vom Datentyp BLOB³. Die Maskenansichten sind im Quelltext in der Datenbank hinterlegt und wird mittels eines Interpreters angezeigt. Datenbanken stellen die von dem Design Tool benötigten Informationen bereit. Sie enthalten die Profildefinition der verschiedenen Profile des CiA, wie DS301 und DS401, und die Hardware-Definitionen, die die Eigenschaften und Fähigkeiten des CAN-Treibers und der CANopen-Bibliothek aufweisen. Weitere Profile, wie z.B. DSP304, DSP402, DS406 und andere sind vorhanden. Ein Anwendungsprotokoll kennt die Definitionen der Hardware nicht und sollte nicht auf eine spezielle Hardware beschränkt sein. Dies wird durch ein gut durchdachtes Treiber-Konzept und eine Treiber-API erreicht. Eine feste Treiber-API bietet die Möglichkeit die Hardware-Profile in Datenbanken zu integrieren. Die Profildatenbanken enthalten die Objektbeschreibung für jeden Index durchzuführen. Die Hardwarekonfiguration. Zusätzlich muß die Datenbank Informationen für Optimierungseinstellungen und etwaige Maskenansichten zur Verfügung stellen. Außerdem Maskenansichten sind in der Datenbank vom

Name
C variable name
Object description
DefinelfExists
Opt_CreateVariable
Opt_CreateTypeDef
Opt_Memory_Specifier
Opt_SameStructure
Opt_UseDefinedVar
Opt_DefinedVarName
Mask_View

Tabella 2: zusätzliche Daten zu den EDS-Daten für einen Index

und wird durch Hardwaredatenbanken unterstützt.

Hardwaredatenbanken, im Gegensatz zu den Profildatenbanken, unterliegen der ständigen Änderung aufgrund von erweiterter Treiberfunktionalität oder Unterstützung neuer Standalone CAN-Controller oder neuer Mikrocontroller mit integriertem CAN-Controller. Diese Änderungen, können durchaus nicht nur den Inhalt der Datenbank, sondern auch die Struktur und das Datenmodell betreffen. Das wurde bei der Konzeption der Hardwaredatenbank berücksichtigt. Die Änderungen an der Datenbank spiegeln sich somit auch in den Eingabemasken für die Hardwarekonfiguration wider. Deshalb war es notwendig, daß die Hardwaredatenbank neben den eigentlichen Daten auch die Eingabemasken speichert. Durch diesen Ansatz ist es möglich graphische Oberflächen für die Konfiguration aktueller Hardware als auch neuer zukünftiger Hardware in der Datenbank zu verwalten.

Das interne Datenmodell des Design-Tools muß ein CANopen-Gerät abbilden und alle Informationen für die Erzeugung von Quelltext und Dokumentation enthalten.

Im Gegensatz zu dem Datenmodell der Profildatenbanken muß die interne

Struktur in der Lage sein auch mehrere Linien zu handhaben, um Anwendungen mit Mikrocontrollern zu unterstützen, die zwei oder mehrere CAN-Controller zu Verfügung stellen.

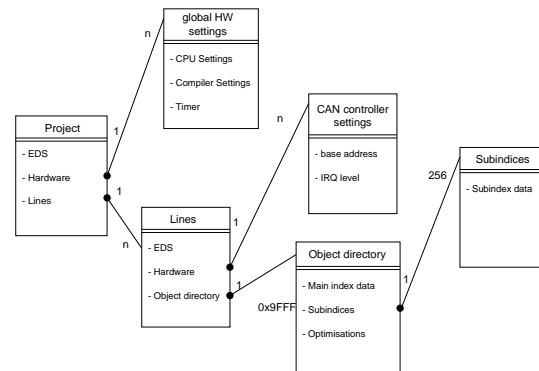


Abbildung 5: Internes Datamodell

Speicherformat der Datenbanken

Für die Auswahl eines geeigneten Datenbankformates wurden CODB-Datenbanken, EDS-Dateien, eine kommerzielle Datenbankerweiterung und die eXtensible Markup Language (XML) auf ihre Verwendbarkeit geprüft. Das Speicherformat sollte folgenden Anforderungen genügen:

- erweiterbar
- lesbar und änderbar für Menschen
- Betriebssystem unabhängig

Die CODB-Dateien (CANopen Datenbank) sind ein durch den CiA definiertes Format, welches von dem Programm EDS-Checker benutzt wird. Das Format benutzt die CSV-Notierung (CSV - Komma getrennte Werte). Dabei müssen die Daten zu einem Index in eine Zeile geschrieben und es darf kein Zeilenumbruch stattfinden. Dieses Format enthält nur die Daten, die zur Prüfung eines EDS notwendig sind. Eine Erweiterung ist nicht vorgesehen.

EDS-Dateien haben eine flache Struktur und sind in Abschnitte unterteilt. Neben der Beschreibung eines Index enthalten

sie Informationen zu Muss- und Kann-Objekten. EDS-Dateien können wie CODB-Dateien Kommentare enthalten und es darf kein Zeilenumbruch innerhalb einer Datenbeschreibung stattfinden.

Obwohl eine kommerzielle Datenbankerweiterung die Daten nicht menschenlesbar ablegt, erfüllt sie alle Forderungen, da durch ein SQL-Interface Daten auf einfache Art und Weise gelesen und manipuliert werden können. Durch die Erfahrung mit anderen Projekten erschwert eine externe Erweiterung die Entwicklung und den Support. Dies wird besonders dann spürbar, wenn das verwendete Betriebssystem sehr starke Änderungen erfährt, wie letztlich mit Windows geschehen.

Auch XML erfüllt alle Anforderungen. Die Daten werden als Text abgelegt und können bei Bedarf verschlüsselt werden. Einzig ein Parser ist notwendig, um XML verwenden zu können. Parser sind für alle gängigen Programmiersprachen verfügbar. Für die Bearbeitung von XML-Dateien sind viele verschiedene kommerzielle und freie Programme verfügbar.

XML wurde aus den dargestellten Möglichkeiten als Format für die Datenbanken von Geräteprofilen und der Beschreibung der Hardware ausgewählt. Mit XML wird auch die Realisierung von Erweiterungen von Drittherstellern vereinfacht.

Erzeugung des Quelltextes

Das DesignTool erzeugt alle Dateien, die für ein CANopen-Projekt notwendig sind. Die Hauptanforderung besteht darin, Quelltext zu erzeugen, der nach einem Kompilervorgang ohne weitere Änderungen lauffähig ist. Durch die einfache Erzeugung des Objektverzeichnisses ist es möglich, den Speicherbedarf für das Objektverzeichnis bereits vor der konkreten Implementierung zu ermitteln.

Der erzeugte Quelltext für ein CANopen-Projekt ist in drei Dateien enthalten:

Konfigurationsdatei (header)

enthält C-Defines mit denen die CANopen-Bibliothek angepaßt wird, z.B. Aktivierung/Deaktivierung von CANopen-Diensten und Konfiguration der Hardware, wie Setzen des IRQ-Levels (CAN/Timer) und Basisadresse des CAN-Controllers.

Objektverzeichnisdefinition

enthält die Strukturdefinition des Objektverzeichnisses passend zur CANopen-Bibliothek als C-Code.

CANopen-Initialisierungsfunktion

eine Funktion, die die ausgewählten Dienste initialisiert, wie z.B. PDO, SDO, EMCY, TIME.

Zusätzlich zum Quelltext wird auch die EDS-Datei, Dokumentation und eine weitere Datei für das Gerätekonfigurationstool CDM erstellt. Die Dateien werden in einem Vorgang erzeugt. Dadurch wird sichergestellt, daß Gerät, EDS-Datei und Dokumentation immer auf dem neuesten Stand und konsistent zueinander sind.

Zusammenfassung

Das DesignTool abstrahiert von der Detailimplementierung eines CANopen-Protokolls und erzeugt als Ergebnis Quelltext und Dokumentation. Dadurch wird der Entwickler von fehleranfälligen, einfachen und monotonen Arbeiten entlastet. Dies führt zur Vereinfachung und Zeitersparnis im Entwicklungsprozess.

Die Quelltexterzeugung vereinfacht die Entwicklung zu einem großen Teil. Wenn Objekte zugefügt oder Attribute geändert werden sollen, muss der Entwickler nicht Elemente einer Struktur oder eines Feldes ändern, sondern benutzt hierfür die graphische Oberfläche. Hierdurch können sich die Ingenieure auf ihre eigentliche Anwendung konzentrieren.

Das gleiche trifft auf die Erweiterung der Anwendung mit neuen CANopen-Diensten zu. Wenn das Objekt für ein SDO-Client zugefügt wird, wird automatisch der Dienst aktiviert und die Initialisierung des Dienstes im Quelltext zugefügt.

Mit der Integration der Hardwareeinstellungen in das DesignTool sind alle Konfigurationsmöglichkeiten in einer Software untergebracht und direkt zugreifbar und änderbar. Durch die Verwaltung mehrerer Hardware-Konfigurationen ist es möglich, ein und dasselbe Projekt auf verschiedener Hardware oder sogar auf einem Betriebssystem zu entwickeln und zu einem späteren Zeitpunkt auf die Zielhardware zu wechseln. Das ist besonders dann sehr von Vorteil, wenn die Zielhardware noch nicht fertig ist, die Software-Entwicklung aber schon beginnen muss.

References

1. CAN in Automation, *Application Layer and Communication Profile DS301* (1 June 2000).
2. CAN in Automation, *Electronic Data Sheet Specification for CANopen DS306* (4 December 2002).
3. W3C, *Extensible Markup Language (XML) 1.0 (Second Edition)* (6 October 2000).