

CANopen Slave Source Code

Embedded Software für CANopen

Der CANopen Slave Source Code beinhaltet alle für eine Slave-Implementierung erforderlichen CANopen-Dienste. Er bietet damit dem Anwender die notwendige Kommunikationsfunktionalität für eine eigene Geräteentwicklung.

Eigenschaften und Vorteile

Durch die Anwendung des CANopen Source Codes kann bei der Produktentwicklung erheblich Zeit eingespart werden. Der Anwender kann sich auf die Integration seiner eigenen Applikation konzentrieren, die Implementierung des CANopen-Protokolls wird hierbei wesentlich vereinfacht. Der Code entspricht der jeweils aktuellsten Version des Kommunikationsprofils CANopen DS-301 und ermöglicht somit eine sichere und CANopen-konforme Umsetzung des Projektes. Um die Anbindung der eigenen Applikationsdaten an das Objektverzeichnis zu erleichtern, sind bereits beispielhafte Einträge im Objektverzeichnis enthalten. Anhand dieser Vorgaben sind eigene Einträge leicht zu erstellen.

Funktionen

Der CANopen Source Code bietet folgende Funktionen/Eigenschaften:

- > SDO Server
- > PDO/SYNC Handling
- > NMT Slave
- > Einfache Anpassung an unterschiedliche Anforderungen
- > Breites Spektrum an unterstützten Hardwareplattformen
- > Source Code-Konfiguration und automatische Generierung des Objektverzeichnisses über CANerator CANopen

Darüber hinaus werden für spezielle Anwendungsgebiete Erweiterungsmodule für den Slave Source Code zur Verfügung gestellt, die separat bestellt werden können:

- > MiniMaster
- > LSS

Anwendungsgebiete

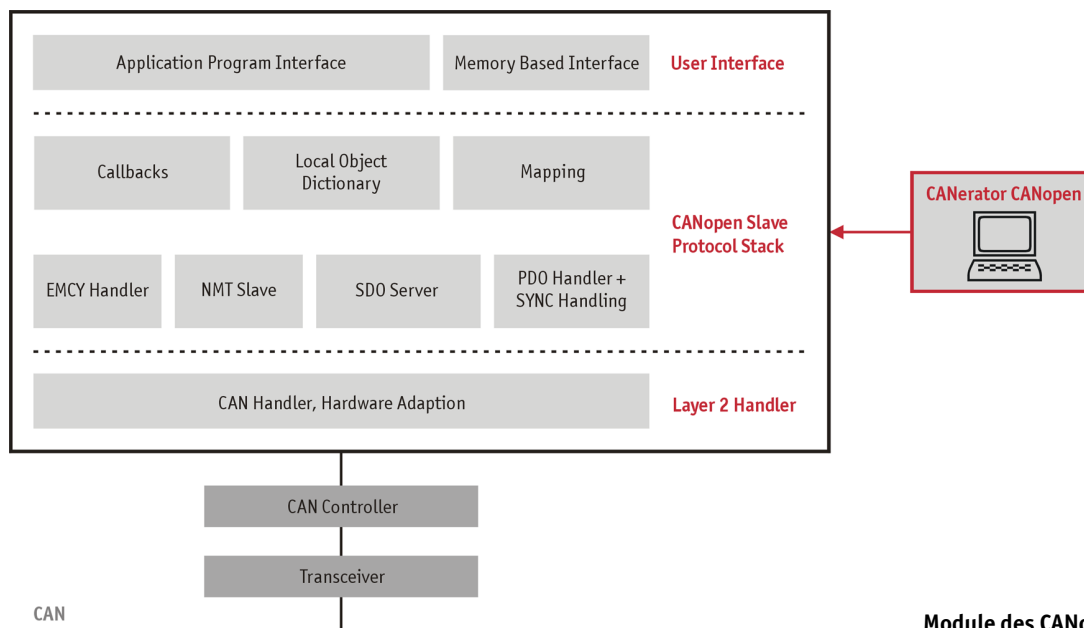
Der Code kann überall dort eingesetzt werden, wo der Kunde seine Geräte mit CANopen-Funktionalität ausstatten möchte. Sinnvoll ist insbesondere der Einsatz bei Systemen, die über eingeschränkte Ressourcen verfügen.

SDO Server

Der SDO Server bietet ein Objektverzeichnis und unterstützt alle definierten SDO-Transfermechanismen, wie Expedited Transfer, Segmented Transfer und Block Transfer, für bis zu zwei parallele SDO Server.

PDO Handling

Für die PDOs wird die Anfangskonfiguration über statische Strukturen vorgegeben, die der Anwender an seine Gegebenheiten anpassen kann. Der Code unterstützt variables Mapping bis zu einer Granularität von 1 Byte. Alle PDOs sind natürlich auch über das Objektverzeichnis konfigurierbar. Weiterhin kann beim Empfang eines PDO ein Callback ausgelöst werden, in dem der Anwender weiteren Behandlungscode platzieren kann. Der Anwender kann bis zu 200 PDOs anlegen und diese dann ereignisbasiert durch



V1.3 2005-7

Module des CANopen Slave Source Codes

Der CANopen Slave Source Code von Vector ist für eine Vielzahl der gängigen Mikrocontroller verfügbar. Aktuelle Informationen dazu finden Sie auf unserer Homepage unter www.vector-informatik.com/sourcecode

geänderte Dateninhalte synchron durch den Empfang einer SYNC-Nachricht oder auch zeitgetriggert übertragen.

NMT Slave

Der Code unterstützt sowohl Guarding als auch Heartbeat. Dabei kann auch auf Heartbeat-Nachrichten anderer Knoten reagiert werden (Heartbeat Consumer). Damit lassen sich in einem Netzwerk sehr effiziente Überwachungsmechanismen etablieren. Weiterhin können anwenderspezifische Emergency-Nachrichten generiert werden.

Der Code unterstützt auch eine Schnittstelle zum nichtflüchtigen Abspeichern von Parametern. Hierbei wird davon ausgegangen, dass der Speicher über ein Dateisystem oder als Flash zur Verfügung steht.

Mini-Master

Bei vielen Embedded-Netzwerken wird nicht die komplette Funktionalität eines CANopen-Managers benötigt. Oft werden unter Einschränkung der Funktionalität Schwerpunkte auf geringen Ressourcenbedarf und einfachste Nutzung gelegt.

Für diesen Zweck gibt es das Add-On Mini-Master. Es beinhaltet:

- > NMT Master – Steuern der Statusmaschine der NMT Slaves, um zum Beispiel das Netzwerk zu starten. Überwachung der Knoten über Node Guarding. Die Heartbeat-Überwachung ist immer Bestandteil des Slave Source Codes.
- > SDO Client – Der Slave kann damit auch einen SDO Transfer initiieren. Hierüber lassen sich Einstellungen in den Objektverzeichnissen der anderen Knoten durchführen und Daten abrufen, die nicht über PDO verfügbar sind – zum Beispiel zu Diagnosezwecken.

LSS (Layer Setting Services)

CANopen-Geräte erfordern unter anderem eine Einstellung von Baudrate und Knotennummer. Geräte ohne Zugriffsmöglichkeit über Schalter, z.B. wasserdichte Sensoren, erfordern eine Einstellung dieser Werte über den CAN-Bus selbst. CANopen spezifiziert hierfür den LSS-Dienst. Mit dem Add-On-Modul LSS werden die für den Slave notwendigen LSS-Dienste als Source Code-Modul zur Verfügung gestellt.

Anpassung des Codes

Bevor der Code übersetzt wird, kann der Anwender die zu unterstützenden Eigenschaften festlegen. Damit kann der Code auch auf Systemen mit sehr eingeschränkten Ressourcen zum Einsatz kommen. Die Anpassung erfolgt dabei über #define-Anweisungen, die in einer Konfigurationsdatei einzustellen sind. Durch Einsatz des separat erhältlichen Tools CANerator CANopen wird diese Anpassung wesentlich erleichtert.

Lokales Objektverzeichnis

Das Objektverzeichnis des Slave Source Codes kann durch den Anwender leicht erweitert werden. Dazu sind die entsprechenden Einträge in C-Dateien hinzuzufügen. Ein solcher Objektverzeichniseintrag enthält die Attribute des Objektes und einen Zeiger auf den eigentlichen Speicherplatz. Für jedes Objekt ist es möglich, beim Lesen oder Schreiben Callback-Funktionen mit benutzerdefinierter Funktionalität auszulösen. Damit kann z.B. eine Bereichsüberprüfung beim Schreiben eines Objektes implementiert werden. Mit Hilfe des CANerator CANopen kann dieses Objektverzeichnis auch automatisch erstellt werden. Weitere funktionale Details finden Sie in der Produktinformation des CANerator CANopen.

Unterstützung bei der Integration

Bei der Integration des CANopen Slave Source Codes in Ihre Umgebung können wir Sie natürlich ebenfalls unterstützen. Dabei reichen die Unterstützungsleistungen von zugeschnittenen Schulungsangeboten, über Workshops bis hin zu Projektarbeit.