

CANopen Slave Source Code

Embedded Software for CANopen

The CANopen Slave Source Code contains all CANopen services required for implementing a slave. It thus offers the user the communication functionality required for internal device development.

Features and Advantages

Significant time savings can be achieved during product development by using the CANopen source code. The user can concentrate on the integration of the application; the implementation of the CANopen protocol is simplified significantly. The code corresponds to the most recent version of the CANopen DS-301 communication profile enabling a secure and CANopen-conforming implementation of the project.

To simplify connection of internal application data to the object directory, there are exemplary entries already in the object directory. Using these specifications, it is easy to create your own entries.

Functions

The CANopen source code offers the following functions/properties:

- > SDO Server
- > PDO/SYNC Handling
- > NMT Slave
- > Simple adjustment to various requirements
- > Broad spectrum of supported hardware platforms
- > Source code configuration and automatic generation of the object directory via CANerator CANopen

For special application areas there are expansion modules available for the Slave Source Code, which can be ordered separately:

- > Mini-Master
- > LSS

Application Areas

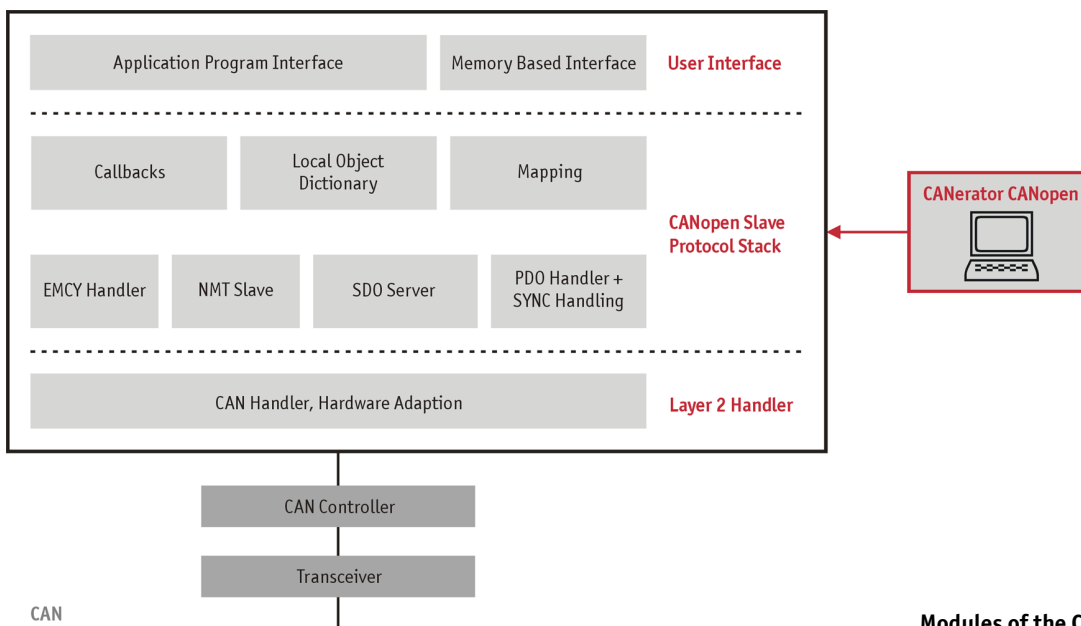
The code can be used everywhere customers would like to equip their devices with CANopen functionality, especially in systems where resources are at a premium.

SDO Server

The SDO Server offers an object directory and supports all defined SDO transfer mechanisms such as expedited transfer, segmented transfer, and block transfer for up to two parallel SDO servers.

PDO Handling

For the PDOs, the initial configuration is specified via static structures that the user can adjust to his circumstances. The code supports variable mapping up to a granularity of 1 byte. All PDOs are also configurable via the object directory. On receipt of a PDO a callback is initiated in which the user can place additional handling code. The user can create up to 200 PDOs and then transmit these event-based through changed data contents, synchronously through the reception of a SYNC message or time-triggered.



V1.4 2005-7

Modules of the CANopen Slave Source Code

The CANopen Slave Source Code by Vector is available for many popular microcontrollers. For current information, please visit our homepage on the Internet at:

www.vector-informatik.com/sourcecode

NMT Slave

The code supports both guarding and heartbeat. It is possible to react to heartbeat messages of other nodes (heartbeat consumer). This makes it possible to set up highly efficient monitoring mechanisms in a network. User-specific emergency messages can also be generated.

The code supports an interface for non-volatile storage of parameters. Here it is assumed that the memory is available via a file system or as flash.

Mini-Master

The full functionality of a CANopen manager is not required for many embedded networks. Often, by limiting functionality, emphasis can be placed on requiring very few resources and the simplest possible use.

This is the purpose of the add-on Mini-Master. It contains:

- > NMT Master – control of the status machine of the NMT slaves to start the network, for example. Monitoring of nodes by means of node guarding. Heartbeat monitoring is always a component of the Slave Source Code.
- > SDO Client – the slave can thus also initiate an SDO transfer. This makes it possible to adjust settings in the object directories of other nodes and to call up data that is not available through PDO (e.g. for diagnostic purposes).

LSS (Layer Setting Services)

Among other things, CANopen devices require a setting of baud rate and node number. Devices without access possibilities via switches (e.g. watertight sensors) require a setting of these values via the CAN bus itself. For this, CANopen specifies the LSS service. With the LSS add-on module, the LSS services necessary for the slave are made available as a source code module.

Adaptation of the Code

Before the code is translated, the user can specify the qualities to be supported allowing the code to be used even on systems with very limited resources. The adaptation occurs via the #define instructions that are set in a configuration file. By using the CANerator tools (sold separately), this adaptation is simplified significantly.

Local Object Directory

The object directory of the Slave Source Code can easily be expanded by the user. For this, appropriate entries must be added to the C files. Such an object directory contains the object's attributes and a pointer to the actual memory location. For each object it is possible to initiate callback functions with user-defined functionality when reading or writing. Thus, for example, an area test can be implemented when writing an object. With the aid of the CANopen CANerator, this object directory can be created automatically. For additional functional details, please refer to the CANopen CANerator product information.

Support during Integration

We can also support you during the integration of the CANopen Slave Source Code into your environment. Our support services range from custom-tailored training solutions to workshops to project work.