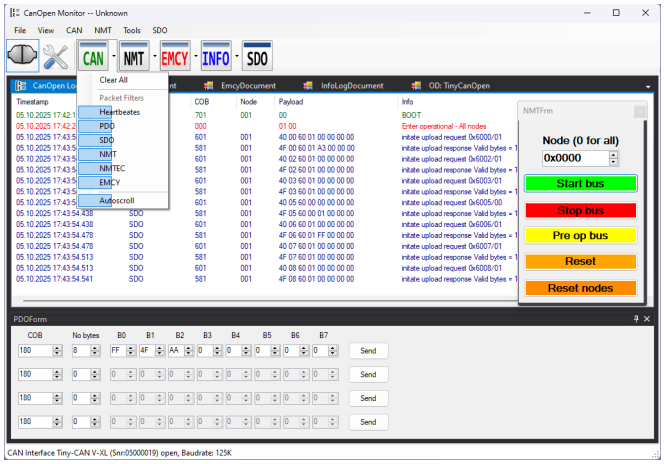
CANopen Analyser, Protokoll Stack, EDS-Editor alles Open-Source

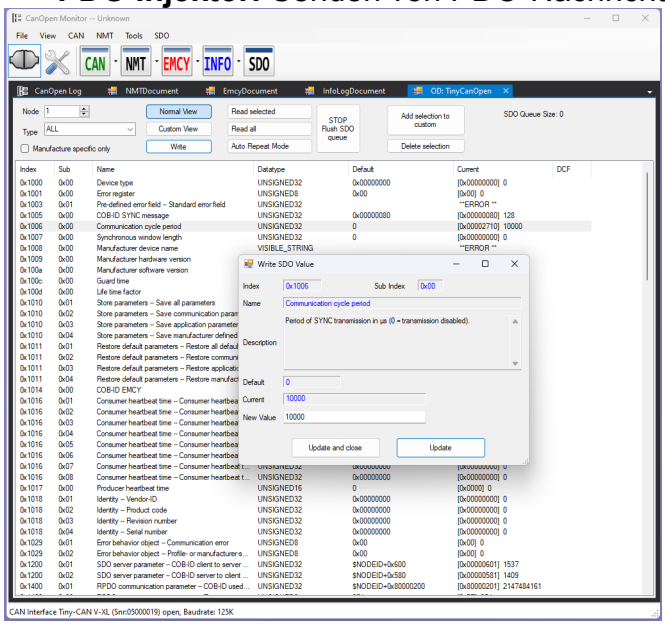
CanOpenMonitor – Erstes Open-Source CANopen Analyse Tool für Windows. Die Software dient der Inspektion von CANopen-Nachrichten, ermöglicht das lesen und schrieben in das Objektverzeichnis, überträgt Firmware Updates, konfiguriert CANopen Devices, setzt NMT-Kommandos ab und simuliert Fehler Nachrichten.



Im „CanOpen Log“ Fenster werden CANopen-Nachrichten aufgezeichnet und dekodiert. Der Datenverkehr lässt sich mittels Filterfunktion auf das wesentliche beschränken. Die aufgezeichneten Daten können als XML-Datei gespeichert und in das Programm zurück geladen werden. NMT und Emergency-Nachrichten werden zusätzlich in eigene Fenster angezeigt. Ein nettes gimmick ist noch das Plugins die Möglichkeit haben empfangene Nachrichten zu dekodieren und den Text in der Spalte „Info“ ändern.

Der CanOpenMonitor ist Plugin-fähig, die Liste der bereits existierenden Plugins hat es in sich:

- **SDO Editor Plugin:** Ermöglicht ein individuelles lesen und schrieben von Parametern im Objektverzeichnis einzelner CAN-Devices. Die Objekt Datenbank kann als EDS oder XDD Datei geladen werden. Es ist möglich einzelne, selektierte oder alle Einträge des Objektverzeichnis zyklisch zu lesen. Laden und schreiben einer DCF (Device Configuration File) Datei. Speichern der Parameter im EEPROM.
- **NMT Plugin:** Senden von NMT-Kommandos, „START/STOP/...“ an einzelne CAN-Knoten oder für den gesamten Bus.
- **EEPROM Plugin:** EEPROM mit „Default“-Werten initialisieren, Daten im EEPROM speichern.
- **Emergency Simulator Plugin:** Simulieren von Emergency-Nachrichten
- **FLASH Loader Plugin:** Wird für Firmware-Updates einzelner CANopen Devices benutzt.
- **PDO Injektor:** Senden von PDO-Nachrichten.

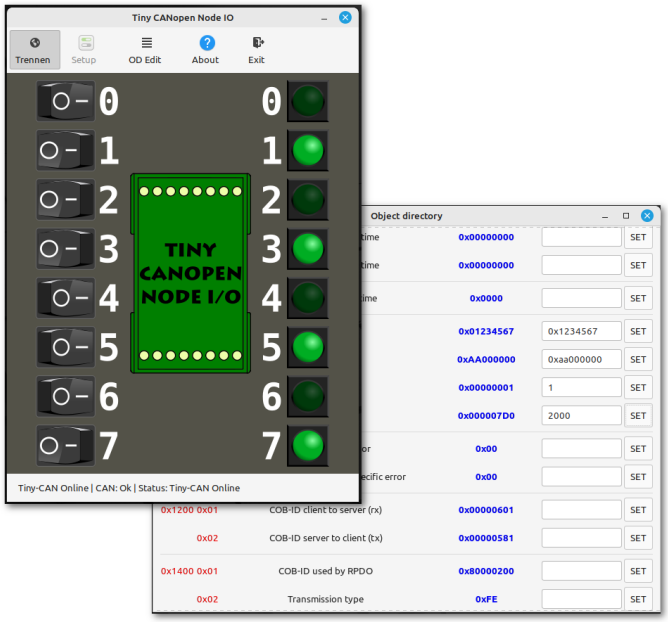


Wie bereits in der Einleitung erwähnt ist der CanOpenMonitor ein Open-Source Projekt. Die Quellen und eine Executable Version stehen auf GitHub zum Download bereit:
https://github.com/MHS-Elektronik/CanOpenMonitor

Die Applikation und die Plugins sind ursprünglich von Robin Cornelius entwickelt worden. Ich habe die Benutzer Oberfläche ansprechender gestaltet, viele Bugs gefixt, die Tiny-CAN Hardware implementiert und den Weg für CAN-FD eröffnet. Das Programm und die Plugins sind in C-Sharp basierend auf Microsoft

Forms entwickelt. Die CAN Treiber wurden in C entwickelt. Compilieren lässt sich das ganze mit der Visual Studio 2022 Community Edition:
https://visualstudio.microsoft.com/de/vs/community

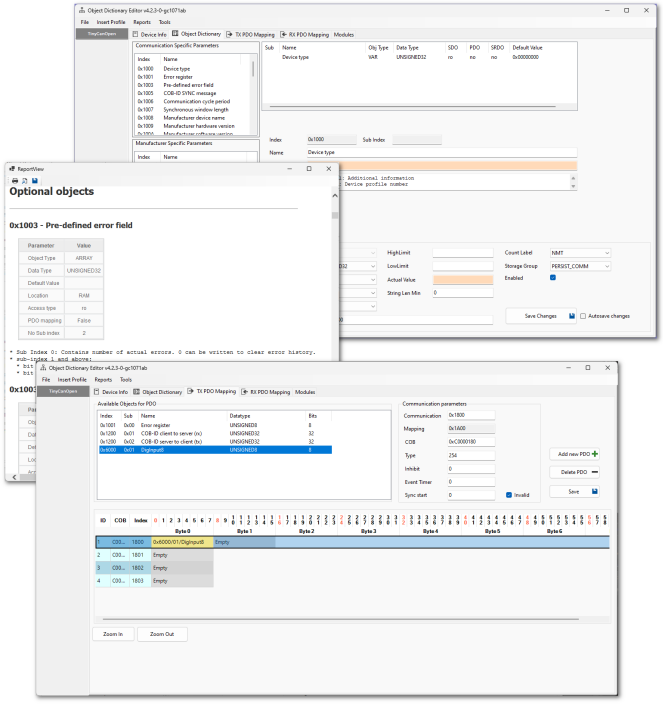
Tiny CANopen Node IO – Eine virtuelle CANopen Digital-IO Baugruppe



Das zweite Open-Source Tool, ein Spaßprojekt für Lehrzwecke, wird von mir zum Testen des CANOpenMonitor verwendet. Als Physikalisches CAN Interface dient ein Tiny-CAN Adpter, die Kommunikation zwischen Tiny CANopen Node IO und dem CANOpenMonitor findet also auf einem Physikalischen CAN-Bus statt, nur die Schalter und LEDs sind nicht echt. „Tiny CANopen Node IO“ basiert auf die CiA Standarts 301 und 401-1 (https://can-cia.org). Als CANopen Backend läuft „CANopenNode“. Auf das Objektverzeichnis wird über die API-Calls von „CANopenNode“ zugegriffen, zusätzlich Informationen wie Daten-Type, Zugriffs-Type, Parameter Name werden aus der EDS Datei ausgelesen. Die Software ist in „C“ programmiert, basiert auf „GTK3“ (https://gtk.org) und „GooCanvas“, läuft somit unter Linux, Windows und auf den kleinem Rasperry PI. Die Quellen und compilierte Versionen für alle oben genanten Systeme finden Sie hier: https://github.com/MHS-Elektronik/TinyCanOpen. Als Entwicklungsumgebung für das Programm wurde Code::Blocks verwendet: https://www.codeblocks.org

CANopenNode – Freier ANSI C Embedded CANopen Stack
Mit über 1600 Stars auf GitHub wohl das größte Highlight in dieser Liste der Open Source Programme, https://github.com/CANopenNode/CANopenNode
Zu CANopenNode gibt es bereits einen sehr guten Artikel den Sie hier https://can-newsletter.org/software/protocol-stacks/220628_open-source-canopen-protocol-stack-extended_cnlm finden. Ich will hier nur auf ein paar Programmtechnische Details eingehen. Der CANopenNode Stack ist so entwickelt worden das er ohne ein Echtzeitbetriebssystem auskommt. Im wesentlichen ist zur Funktion des Protokoll Stacks der Aufruf von 3 Funktion notwendig, all diese 3 Funktionen sind nicht blockierend. Die Funktion „CO_CANinterrupt“ ist die Interrupt Routine die beim Empfang und versenden von CAN Paketen aufgerufen wird. Die Funktion „tmrTask_thread“ wird von einem Timer Interrupt zyklisch, normalerweise alle 1ms aufgerufen. Dann ist noch die „CO_process“ Funktion die zyklisch von der Main-Schleife aus aufgerufen werden muss. Die Zeitdifferenz des letzten Aufrufs muss der Funktion als Parameter übergeben werden.
In ein Projekt sollten alle Dateien des Stacks eingebunden werden. Die einzelnen Module werden dann über die „define“-Makros in „301/CO_config.h“ eingebunden und konfiguriert.

CANopenEditor – EDS-Datei Edtitor und noch vieles mehr



Neben dem laden und speichern von EDS-Dateien werden weitere Dateiformate wie z.B. XDD unterstützt. Das Programm lässt sich zum Konvertieren der Dateien verwenden. Beim Erzeugen eines neuen Projekts ist die Import Profile Funktion sehr hilfreich, Teile einer vorhandenen EDS/XDD-Datei lassen sich importieren. Die CANopen Profile DS301, DS401 und DS302 können aus Templet Dateien importiert werden. Zusätzlich ist das erstellen der Dokumentation im „HTML“ oder „Markdown“-Format möglich. Der CANopenEditor ist der größte Liebling von CANopenNode. Das Programm erzeugt die beiden für den Betrieb von CANopenNode essentiellen Dateien „OD.c“ und „OD.h“. Die beiden Dateien beinhalten das Objektverzeichnis. Der Zugriff auf das Objektverzeichnis kann direkt oder über API-Funktionen von CANopenNode erfolgen.

Fazit

Die Open-Source Welt bietet alle für die Entwicklung von CANopen Devices Notwendigen Werkzeuge an. CANopenNode und der CANopenEditor müssen sich nicht mehr hinter kommerziellen Tools verstecken. CANopenNode ist sauber strukturiert in C programmiert, eine Dokumentation und viele Beispiele für diverse Mikrocontroller, Linux und Windows sind vorhanden. Der CANopen Stack wird permanent weiterentwickelt und gepflegt. Beim CANopenEditor verhält es sich ähnlich, auch dieses Programm wird gepflegt und weiterentwickelt. Der CANopenMonitor hingt seinen beiden Brüdern etwas hinter her aber auch hier tut sich gerade einiges. Für alle Tools gilt, die Portierung nach CAN-FD ist meiner Meinung nach nur noch eine Frage der Zeit.