# Tiny-CAN

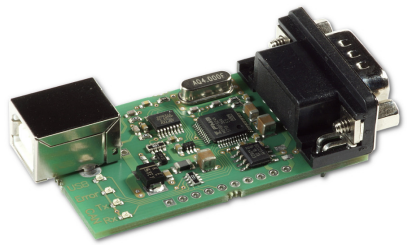
(Übersicht Tiny-CAN Hard & Software, Installationsanweisung)

Version: 4.5 vom 23.07.2020

# Inhaltsverzeichnis

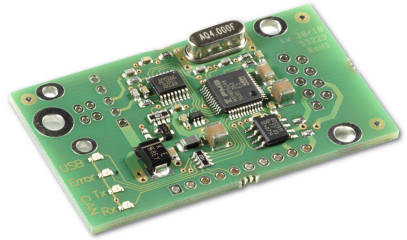
# 1. Übersicht Tiny-CAN Hardware

## Tiny-CAN I-XL (Ver. 2)



Low Cost CAN-USB-Adapter, ohne Gehäuse und galvanische Trennung. Aufgrund des geringen Stromverbrauchs und der geringen Abmessungen (54 x 30 x 15 mm) besonders gut in Verbindung mit dem „Raspberry PI“ geeignet.
Das Modul unterstützt den „Silent-Modus“ sowie „Transmit Message Request“ (Bestätigung versendeter CAN-Nachrichten).

## Tiny-CAN I-XL Embedded (Ver. 2)



Low Cost Embedded CAN-USB-Adapter, zur Integration in kundenspezifische Hardware ohne galvanische Trennung.
Die Hardware entspricht exakt dem Tiny-CAN I-XL nur ohne USB und Sub-D Stecker. In der „B“ Variante sind auch die LEDs nicht bestückt. Für die Kontaktierung stehen 2 Stiftleisten im Raster 2,54mm zur verfügung.

## Tiny-CAN II-XL (Ver. 2)


Standard Industrie CAN-USB-Adapter mit galvanischer Trennung.
Der Adapter für normale Anforderungen mit hervorragendem Preis Leistungsverhältnis.
Das Modul unterstützt den „Silent-Modus“ sowie den „Transmit Message Request“ (Bestätigung versendeter CAN-Nachrichten).

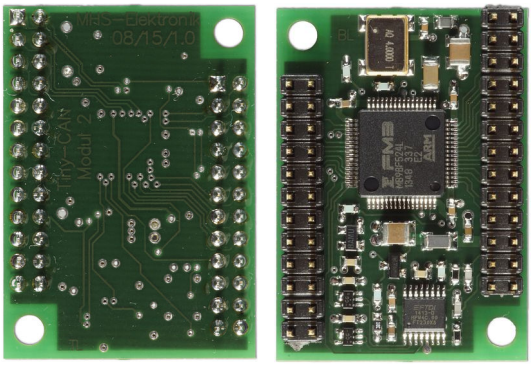
## Tiny-CAN IV-XL (Ver. 2)


CAN-USB-Adapter der speziell für höchste Anforderungen entwickelt wurde. Kein Datenverlust bei 100% Buslast (1 MBit/s), keine Fehler auf dem CAN Bus beim abziehen des USB-Kabels, Überwachung des Mikrocontrollers durch einen Hardware-Watchdog, galvanische Trennung, weiter Versorgungsspannungsbereich von 4 bis 6 V. Versorgung des CAN Line-Treibers von 5 V bei nur 4V am USB-Bus.
Das Modul unterstützt „Silent-Modus“, „Transmit Message Request“ (Bestätigung versendeter CAN-Nachrichten), „Automatic Retransmission disable“ und Hardware Time-Stamps mit 0,1ms Auflösung.

## Tiny-CAN LS

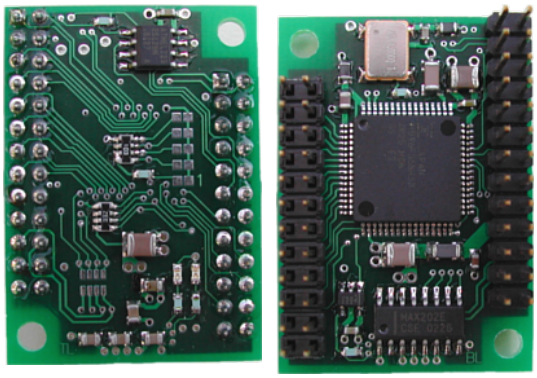

CAN-USB-Adapter für Lowspeed-CAN mit galvanischer Trennung, CAN-Line Treiber TJA1055T (ISO 11898-3), maximale Übertragungsgeschwindigkeit 125 kBit/s.
Das Modul unterstützt „Silent-Modus“, „Transmit Message Request“ (Bestätigung versendeter CAN-Nachrichten), „Automatic Retransmission disable“ und Hardware Time-Stamps mit 0,1ms Auflösung.

## Tiny-CAN M2



CAN-USB-Modul, speziell für den Embedded Bereich entwickelt, als Aufsteckmodul für die Integration in kundenspezifischer Hardware. Kundenspezifische Firmware-Anpassungen sind möglich, zahlreiche nicht genutzte I/Os (Digital, SPI, I2C) stehen zur Verfügung. Als besonderes Leistungsmerkmale unterstützt das Modul den „Silent-Modus“ sowie den „Transmit Message Request“ (Bestätigung versendeter CAN-Nachrichten).

## Tiny-CAN M232



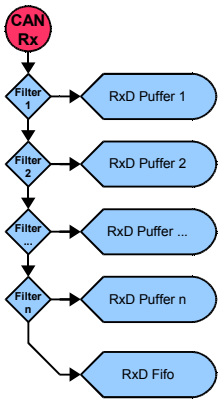
CAN-RS232-Modul, speziell für den Embedded Bereich entwickelt, als Aufsteckmodul für die Integration in kundenspezifischer Hardware. Es stehen 2 Schnittstellen zur Verfügung, eine mit TTL Pegel (5V) und eine mit RS232 Pegel (12V). Die Baudrate der RS232 Schnittstelle lässt sich über Löt-Jumper konfigurieren. 2 LEDs zeigen den Status des Moduls an. Kundenspezifische Firmware-Anpassungen sind möglich, zahlreiche nicht genutzte I/Os (Digital, SPI, I2C) stehen zur Verfügung. Als besonderes Leistungsmerkmal unterstützt das Modul den „Silent-Modus“ sowie den „Transmit Message Request“ (Bestätigung versendeter CAN-Nachrichten).

# 1.1 Funktionen und Leistungsmerkmale im Überblick

| | Tiny-CAN I-XL (Ver. 2) | Tiny-CAN I-XL Embedded (Ver. 2) | Tiny-CAN II-XL (Ver. 2) | Tiny-CAN IV-XL (Ver. 2) | Tiny-CAN LS | Tiny-CAN M2 | Tiny-CAN M232 |
|---|---|---|---|---|---|---|---|
| ***PC Interface*** | USB | | | | | | RS232 |
| ***Spannungsversorgung*** | USB, max. 90 mA | USB, max. 90 mA | USB, max. 150 mA | USB, max. 200 mA<br>4 - 6V | USB, max. 150 mA | 5V, ca. 26mA | 5V, ca. 36mA |
| ***4 Status LEDs: Power/USB, Error, CAN-Rx, CAN-Tx*** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| ***CAN Interface*** | High Speed (ISO 11898-2) Treiber: PCA82C251T | | | High Speed (ISO 11898-2)<br>Treiber: SN65HVD255D | Lowspeed (ISO 11898-3)<br>Treiber: TJA1055T | CAN „Line-Treiber“ nicht auf der Leiterplatte | |
| ***CAN-Anschluss*** | 9-pol. Sub-D-Stecker entsprechend CiA-DS102-1 | - | 9-pol. Sub-D-Stecker entsprechend CiA-DS102-1 | | | - | - |
| ***Übertragungsraten*** | 10 kBit/s – 1 MBit/s | | | | 10 kBit/s – 125 kBit/s | 10 kBit/s – 1 MBit/s | |
| ***Benutzerdefinierte CAN Übertragungsraten möglich*** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| ***CAN-Spezifikation*** | 2.0A (11-Bit ID) und 2.0B (29-Bit ID) | | | | | | |
| ***Größe Empfangs-FIFO*** | 900 | | | | | | 384 |
| ***Größe Sende-FIFO*** | 72 | | | | | | |
| ***Hardware Filter*** | 4 | | | | | | 8 |
| ***Intervall Puffer*** | 4 | | | 8 | 4 | | |
| ***Firmware Update über PC möglich*** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| ***Galvanische Trennung*** | ✗ | ✗ | ✓ | ✓ | ✓ | - | - |
| ***Silent Mode*** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| ***Transmit Message Request*** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| ***Automatic Retransmission disable*** | ✗ | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ |
| ***Gehäuse*** | ✗ | ✗ | ✓ | ✓ | ✓ | ✗ | ✗ |
| ***Hardware Timestamp*** | ✓<br>Auflösung: 0,1ms | ✓<br>Auflösung: 0,1ms | ✓<br>Auflösung: 0,1ms | ✓<br>Auflösung: 0,1ms | ✓<br>Auflösung: 0,1ms | ✓<br>Auflösung: 0,1ms | ✗ |
| ***CAN Safe*** | ✗ | ✗ | ✗ | ✓ | ✗ | - | - |
| ***Maximale Buslast*** | ca. 50% | | | 100% | | ca. 50% | ca. 10% |
| ***USB-Suspend-Mode*** | ✗ | ✗ | ✓ | ✓ | ✓ | - | - |
| ***Watchdog*** | ✓<br>Software-Watchdog | ✓<br>Software-Watchdog | ✓<br>Software-Watchdog | ✓<br>Hardware-Watchdog | ✓<br>Software-Watchdog | ✓<br>Software-Watchdog | ✓<br>Software-Watchdog |
| ***Prozessor*** | Cypress MB9BF524KPMC<br>(32-Bit/48MHz, 256 kB Flash, 32 kB RAM) | | | | | Cypress MB9BF524LPMC<br>(32-Bit/48MHz, 256 kB Flash, 32 kB RAM) | Cypress MB96F356RS<br>(16-Bit/24MHz, 288 kB Flash, 12 kB RAM) |
| ***USB-Controller*** | FTDI FT230XS | | | FTDI FT2232HL | FTDI FT230XS | | - |
| ***Zertifizierung*** | CE | | | | | | |
| ***Software*** | Tiny-CAN API & SLCAN API | | | Tiny-CAN API | | | |
| ***Betriebssysteme*** | Windows (ab XP), Apple (OS X), Linux | | | | | | |

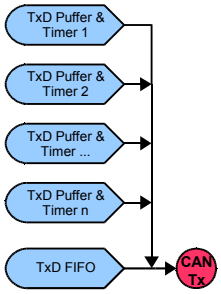
## *1.2 Erläuterungen Leistungsmerkmale der Tiny-CAN Hardware*

**Hardware Filter**




Je nach Hardware ist eine bestimmte Anzahl von Filtern vorhanden. Mit diesen Filtern ist es möglich, ein hohes Datenaufkommen auf dem USB-Bus zu reduzieren. Eine gefilterte Nachricht wird in einem dazugehörigen Puffer gespeichert. Dieser Puffer wird immer wieder überschrieben. Alle Daten, die nicht gefiltert wurden, gelangen in das „RxD-FIFO“ des Moduls.

**Intervall Puffer**




Je nach Hardware ist eine bestimmte Anzahl von TxD-Puffern mit Intervalltimern vorhanden. Diese ermöglichen das Versenden zyklischer Nachrichten in Echtzeit. Der normale Versand von CAN Nachrichten erfolgt über das „TxD-FIFO“ des Moduls.

**Silent Mode**
Zusätzlich zu den Funktionen „CAN Start“ und „CAN Stop“ steht der Silent Mode zur Verfügung.
Der CAN-Controller ist nur passiv am Bus, empfangene CAN Nachrichten werden nicht mit einem ACK quittiert. Bei Fehlern werden aktiv keine „Error Frames“ gesendet. Das Senden von CAN-Nachrichten ist nicht möglich. Der „Silent Mode“ wird verwendet, um einen CAN-Bus abzuhorchen, ohne diesen zu beeinflussen.

**Transmit Message Request**
Erfolgreich gesendete CAN Nachrichten werden in das Empfangs-FIFO zurückgeschrieben, um den Versand der Nachrichten zu bestätigen. Die Nachrichten werden durch das TxD Flag gekennzeichnet. Die Funktion kann via Software aktiviert/deaktiviert werden.

**Automatic Retransmission disable**
Das automatische wiederholte Versenden von CAN Nachrichten bei Fehlern wird unterbunden. Die Funktion kann via Software aktiviert/deaktiviert werden.

**Hardware Timestamp**
Ein Hardware Timestamp wird in der Hardware erstellt und beim Empfang bzw. dem erfolgreichen Versenden einer CAN Nachricht an der entsprechenden Stelle eingefügt.

**CAN Safe**
Vermeidet Fehler auf dem CAN Bus beim Abstecken des USB-Kabels oder Ausschalten des Computers

**Maximale Buslast**
Bei 1 MBit/s, Standart Frames mit 8 Byte Datenlänge, alle Hardware Filter aus

**USB-Suspend-Mode**
Reduzierung des Stromverbrauchs im Standby-Modus. Die Elektronik des Moduls wird zum größten Teil spannungslos geschaltet, ein Empfang oder Senden von CAN Nachrichten ist dann nicht möglich.

**Watchdog**
Überwachung des Prozessors

# 2. Installation

Die Datei „TinyCan_xxx.exe“ für Windows oder „tiny_can_xxx.tar.gz“ für Linux von http://www.mhs-elektronik.de downloaden und entpacken, bzw. installieren.

### Verzeichnisse

***.../tiny_can/...***

| | |
|---|---|
| *FtdiDriver* | [2]Systemtreiber für den USB Chip |
| *TCanCheck.exe* | [2]TCanCheck Utiliti – Überprüft die korrekte Installation, die Kommunikation zur Hardware und deren Firmware Version |
| *TCanCheck.ini* | [2]Konfigurationsdatei für TCanCheck |
| *uinst.exe* | [2]Paket Uninstaller |
| *CANcool* | [2]CAN Bus Monitor - CANcool |
| *TcanProg* | [2]Tiny-CAN Firmware Update Utility |
| *doku* | |
| *TinyCan.pdf* | Dieses Dokument |
| *tiny_can1xl_manual.pdf* | Hardware und Service Manual zum Tiny-CAN I-XL Adapter |
| *tiny_can1xl_v2_manual.pdf* | Hardware und Service Manual zum Tiny-CAN I-XL Adapter (Hardware Version 2) |
| *tiny_can1xl_embedded_manual.pdf* | Hardware und Service Manual zum Tiny-CAN I-XL Embedded Adapter |
| *tiny_can1xl_embedded_v2_manual.pdf* | Hardware und Service Manual zum Tiny-CAN I-XL Embedded Adapter (Hardware Version 2) |
| *tiny_can2xl_manual.pdf* | Hardware und Service Manual zum Tiny-CAN II-XL Adapter |
| *tiny_can2xl_v2_manual.pdf* | Hardware und Service Manual zum Tiny-CAN II-XL Adapter (Hardware Version 2) |
| *tiny_can4xl_manual.pdf* | Hardware und Service Manual zum Tiny-CAN IV-XL Adapter |
| *tiny_can4xl_v2_manual.pdf* | Hardware und Service Manual zum Tiny-CAN IV-XL Adapter (Hardware Version 2) |
| *tiny_can_ls_manual.pdf* | Hardware und Service Manual zum Tiny-CAN LS Adapter (Lowspeed-CAN) |
| *tiny_can_m1_manual.pdf* | Hardware und Service Manual zum Tiny-CAN M1 Modul |
| *tiny_can_m2_manual.pdf* | Hardware und Service Manual zum Tiny-CAN M2 Modul |
| *tiny_can_m232_manual.pdf* | Hardware und Service Manual zum Tiny-CAN M232 Modul |
| *can_monitor.pdf* | Tiny-CAN-Monitor Handbuch, Installation und Benutzung |
| *canlcd1_manual.pdf* | CAN-LCD 1 |
| *canlcd1_v2_manual.pdf* | CAN-LCD 1 (Ver. 2.1) |
| *canlcd2_manual.pdf* | CAN-LCD 2 |
| *canlcd3_manual.pdf* | CAN-LCD 3 |
| *CanLcdQuickStart.pdf* | CAN-LCD Quick Start Manual (Programmierung / Verdrahtung) |
| *can_api* | Tiny-CAN API – Treiber DLL/Lib |
| *dev* | |
| *TinyCanAPI.pdf* | Dokumentation der Tiny-CAN API |
| *lib* | Interface der Tiny-CAN API in C/C++ |
| *static* | Interface der Tiny-CAN API DLL zum statischen linken |
| *C* | Beispielprogramme für C |
| *can_monitor* | Die Quellen von Tiny-CAN-Monitor |
| *python* | Interface der Tiny-CAN API für Python |
| *c-sharp* | [2]Interface der Tiny-CAN API für C-Sharp, inkl. Beispiele |
| *vb6* | [2]Interface der Tiny-CAN API für Visual-Basic Version 6.0 |
| *vb2013* | [2]Interface der Tiny-CAN API für Visual-Basic 2013 |
| *Delphi* | [2]Interface der Tiny-CAN API für Delphi als Komponente, inkl. Beispiele |
| *labview* | Interface der Tiny-CAN API für LabView, inkl. Beispiele |
| *ProfiLab* | [2]ProfiLab CAN Plugin und Beispiele |
| *bin* | Tiny-CAN-Monitor Executeble / GTK+ |
| *etc* | [2]GTK+ |
| *share* | [2]GTK+ |
| *Lib* | [2]GTK+ |
| *BitCalc* | [2]CAN-Bus Bit-Timing Calculator |

[1] Nur in der Linux Version; [2] Nur in der Windows Version

## *2.1. Treiber Installation*

Alle Tiny-CAN Adapter und Module verwenden als USB-Controller einen USB zu RS232 Konverter von FTDI (http://www.ftdichip.com). Auf allen Betriebssystemen wird der Treiber des Herstellers, bzw. der im System enthaltene Treiber verwendet. Die „Tiny-CAN API“ wird über eine „DLL“ bzw. „Shared-Library“ abgebildet.

## 2.1.1. Windows

Unter Windows wird der „D2XX“ Treiber von FTDI verwendet. Dieser Treiber ist im Software-Paket enthalten und wird über das Tool „Tiny-Can Check“ automatisch installiert.

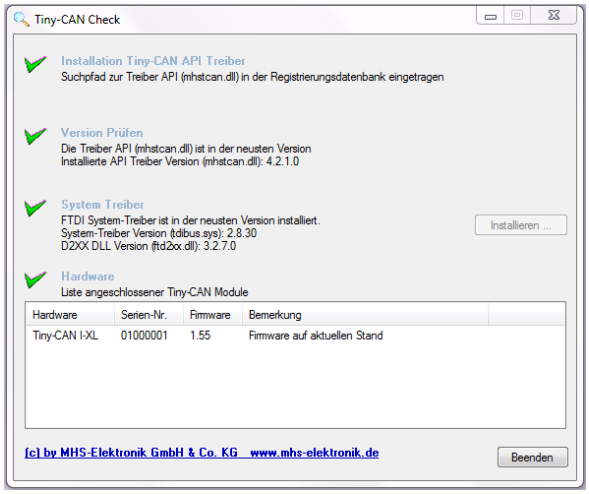
„Tiny-CAN Check“ überprüft die Installation und Versionen des FTDI und Tiny-CAN API Treibers.

Angeschlossene „Tiny-CANs“ werden ebenfalls überprüft und wenn erforderlich wird ein Firmware-Update angeboten.

| Hardware | Serien-Nr. | Firmware | Bemerkung | Aktion |
|---|---|---|---|---|
| TINY-CAN I-XL | 01000001 | 1.63 | Firmware veraltet, aktuelle Version: 1.65 | Update |

Durch einen Klick auf „Update“ wird die Firmware ersetzt.

Für den Fall dass auf dem Modul eine Alternative Firmware installiert ist, bietet „Tiny-CAN Check“ an diese zu ersetzen, auch ein misslungenes Update wird von „Tiny-CAN Check“ repariert.




Das Installationsprogramm legt folgende Einträge in der Windows-Registrierungsdatenbank an, welche für den Betrieb der Software erforderlich sind.

| Schlüssel | Name | Wert <Beschreibung> |
|---|---|---|
| HKEY_LOCAL_MACHINE\Software\Tiny-CAN\API | PATH | <Pfad in dem sich die Tiny-CAN API Treiber DLLs (mhstcan.dll, ...) befinden> |
| HKEY_LOCAL_MACHINE\Software\Tiny-CAN\PROFILAB\LIZENZ | PATH | <Pfad in dem sich das Lizenz File für den ProfiLab Plugin befindet> |
| HKEY_LOCAL_MACHINE\Software\Tiny-CAN\FIRMWARE-UPDATE | PATH | <Pfad in dem sich das Firmware-Update-Tool befindet> |

## 2.1.2. Linux

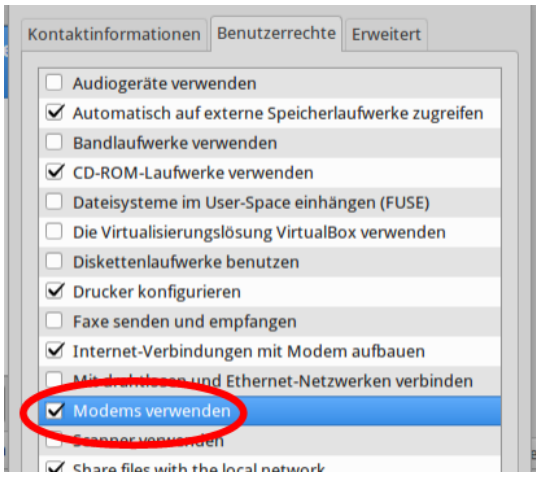
Unter Linux ist keine Installation eines Kernel Treibers notwendig. Das benötigte Kernel-Modul „ftdi_sio“ gehört bereits zum Linux Kernel.

Voraussetzungen für den Betrieb der Hardware:

- Sie benötigen die entsprechenden Zugriffsrechte. Unter Ubuntu müssen Sie dazu der Gruppe „dialout“ angehören.

  Über den Desktop einzustellen:
  „Einstellungen“ → „Benutzer und Gruppen“ → „Erweiterte Einstellungen“ → „Benutzerrechte“ → „Modems verwenden“ markieren.

- Das Paket „modemmanager“ stört den Zugriff auf die Tiny-CAN Hardware beim anstecken des Devices. Wenn möglich sollte dieses Paket deinstalliert werden.



Nützliche Befehle:

*Erkannte FTDI Chips auflisten:*

```
> lsusb | grep 0403:
Bus 001 Device 004: ID 0403:6010 Future Technology Devices International, Ltd FT2232C Dual USB-
UART/FIFO IC
```

*Serielle USB Devices anzeigen lassen:*

```
> dmesg | grep ttyUSB
[ 3875.092687] usb 1-2: FTDI USB Serial Device converter now attached to ttyUSB0
[ 3875.093117] usb 1-2: FTDI USB Serial Device converter now attached to ttyUSB1
```

*Es wurden die beiden Devices „ttyUSB0“ und „ttyUSB1“ erzeugt. Das Tiny-CAN IV-XL Modul erzeugt 2 Devices.*

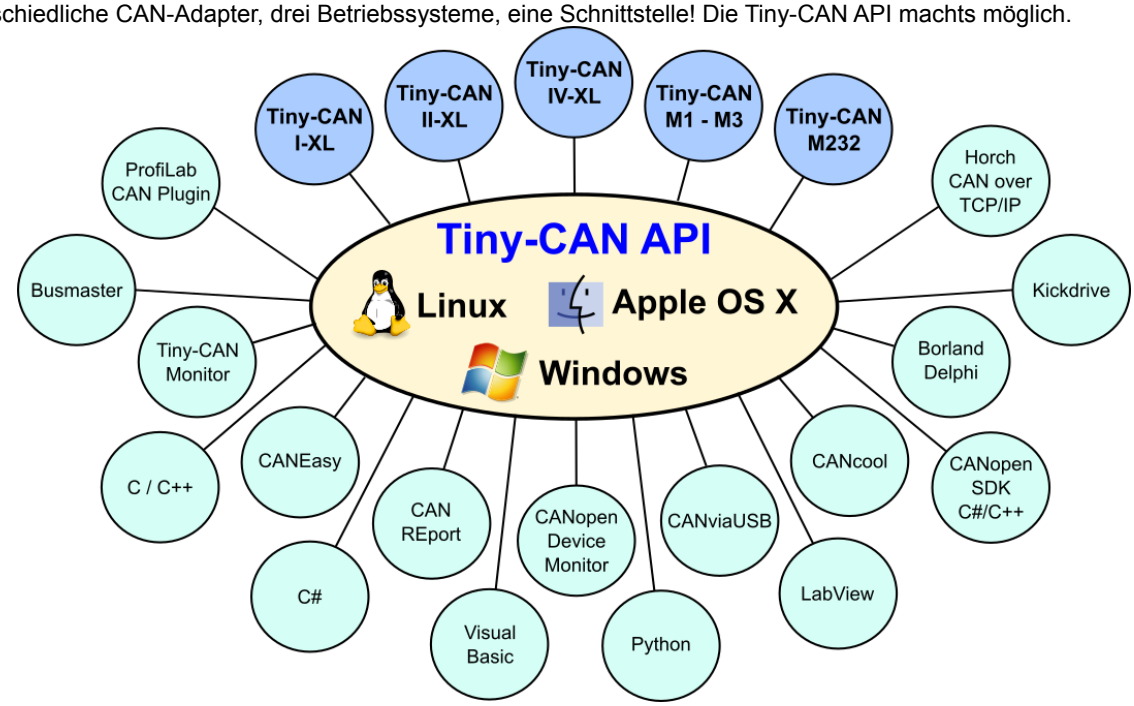
*Geöffnete Serielle USB Devices anzeigen:*

```
> lsof | grep ttyUSB0
can_monitor  2263      klaus   12u      CHR      188,0      0t0     143612 /dev/ttyUSB0
```

*Die erste Spalte gibt das Programm, die letzte Spalte das Device aus.*
*In unserem Beispiel wird das Device „/dev/ttyUSB0“ von „can_monitor“ benutzt. Wenn „can_monitor“ nicht gestartet ist darf „lsof“ nichts ausgeben.*

# 3. Tiny-CAN API

Unterschiedliche CAN-Adapter, drei Betriebssysteme, eine Schnittstelle! Die Tiny-CAN API machts möglich.

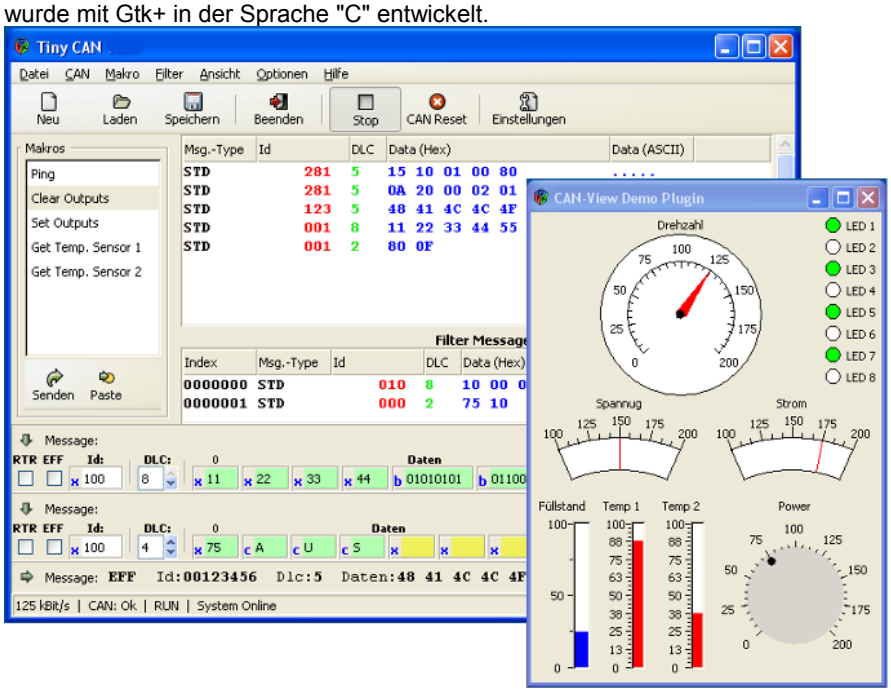



Weitere Informationen zur Tiny-CAN API:
**„Tiny-CAN API Referenzhandbuch“ (.../tiny_can/dev/TinyCanAPI.pdf)**.

# 4. Eigene Tools

## *4.1 Tiny-CAN Monitor*

Tiny-CAN-Monitor, ein CAN-Monitor für Windows und Linux. Das Programm ist ein GNU - Open Source Projekt und wurde mit Gtk+ in der Sprache "C" entwickelt.

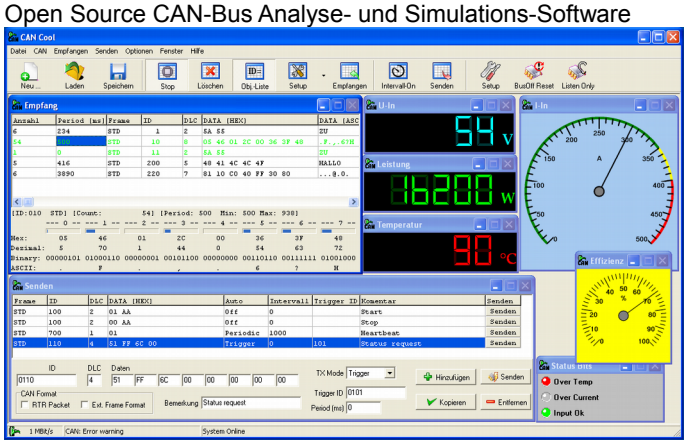



### Installation

Die Installation und Benutzung des Programms sind im Dokument **„Tiny-CAN Handbuch“** (**.../tiny_can/doku/can_monitor.pdf)** beschrieben.

## *4.2 CANcool*

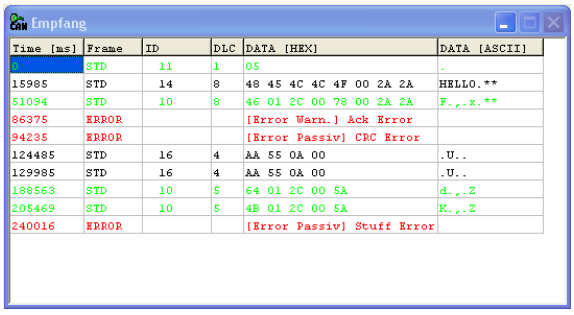
Open Source CAN-Bus Analyse- und Simulations-Software




Funktionen:

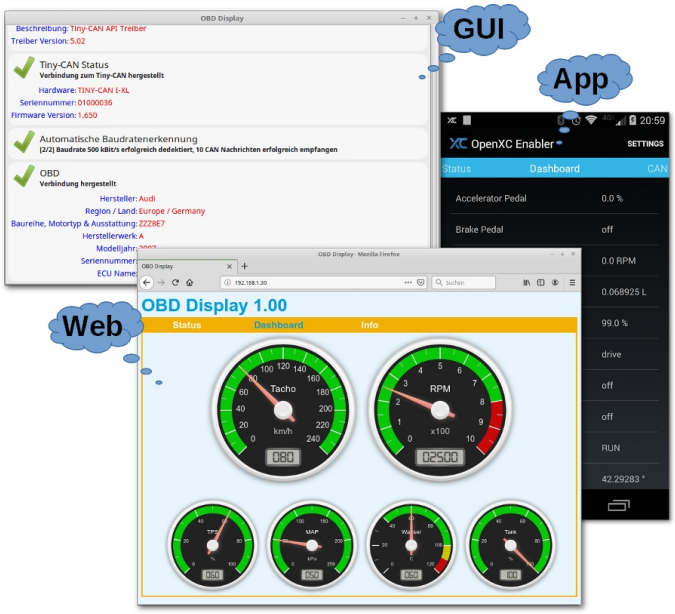
- Dank Multithreading Can-Trace Aufzeichnung ohne Datenverlust
- Abspeichern eines CAN Trace in einer Datei
- Auflistung von CAN Objekten mit statistischer Auswertung
- Aufzeichnung einzelner CAN-Bus Fehler, nur mit dem Tiny-CAN IV-XL möglich
- Anzeige von CAN-Signalen als Zeigerinstrument, Digitalanzeige und LEDs
- Filterfunktion: bekannte/unbekannte/alle Nachrichten anzeigen
- Anlegen einer Transmit-Liste, jede Nachricht mit eigenem Sende-Button
- Speichern und Laden der Transmit-Liste aus einer Datei
- Auto Transmit via Timer, Trigger Message und RTR Anforderung
- Kopieren empfangener Nachrichten direkt in die Transmit-Liste

Das Programm wurde objektorientiert in der Sprache Pascal (Delphi 7) entwickelt.
Der Zugriff auf den CAN Adapter basiert auf der Delphi Tiny-CAN Komponente über die Tiny-CAN API. Alle verwendeten Komponenten liegen dem Paket als Quellen bei.



# 5. Applikationsbeispiele: J1939, OBD-II, IoT

## *5.1 OBD Display – Fahrzeuginformationen mit dem Rasperry PI/Linux anzeigen*




Über die OBD-II Schnittstelle werden per CAN-Bus Messdaten(SID 0x01), Fahrzeuginformationen wie Fahrgestellnummer (SID 0x09) und der Fehlerspeicher (Diagnostic Trouble Codes, SID 0x03) abgefragt. Ohne IoT läuft heute gar nichts mehr, die wichtigsten Fahrzeugdaten werden als HTML5 Seite über einen Apache Webserver bereitgestellt. Für Apps steht eine „JSON over TCP/IP“ Schnittstelle zur Verfügung.

**https://github.com/MHS-Elektronik/OBD-Display**

## *5.2 J1939 Display – Betriebsdaten von Blockheizkraftwerken online und lokal visualisieren*


Als Schnittstelle zum Web greift die Software auf HTML5, SQLite, RRDtool und Modbus TCP zurück.

**https://github.com/MHS-Elektronik/J1939Display**

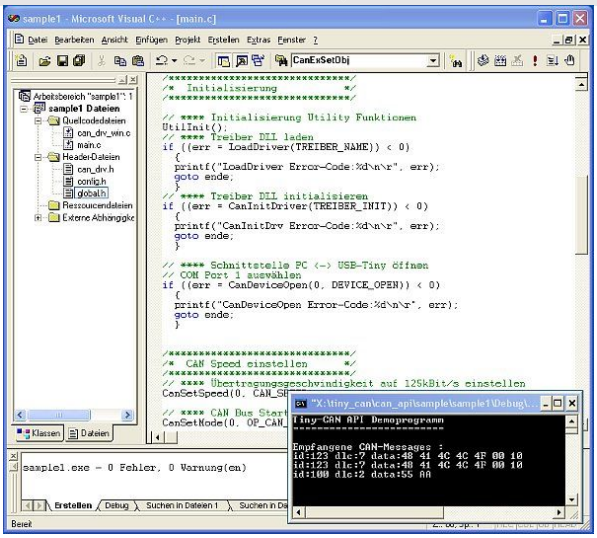
# 6. Applikationsentwicklung mit der Tiny-CAN API

Die Basis für alle unterstützten Programmiersprachen ist die API in Form einer DLL/Shared Library, die in „C“ entwickelt wurde.

Weitere Informationen zur Tiny-CAN API:
**„Tiny-CAN API Referenzhandbuch“ (.../tiny_can/dev/TinyCanAPI.pdf)**.

# 6.1 Unterstützte Sprachen, Umgebungen

## C / C++



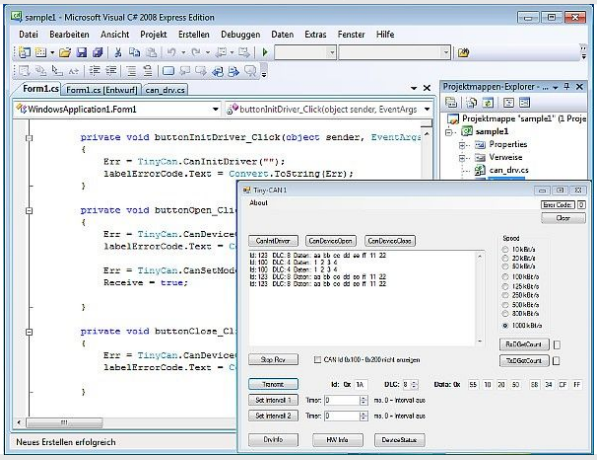
In einer Vielzahl von Beispielen wird die Anwendung der API Funktionen veranschaulicht:

- Empfang und Senden von CAN Nachrichten
- Ereignissteuerung über Callbackfunktionen
- Setzen von Filtern
- Versand Zyklischer CAN Nachrichten
- Abfrage der verwendeten Hardware und deren Features
- Setzen benutzerdefinierter Übertragungsgeschwindigkeiten
- Benutzung der API von mehreren Threads
- Handling mehrerer Tiny-CANs
- Verwendung virtueller FIFOs

Compiler:

- Microsoft VC6
- Microsoft VC2013
- „gcc“ mit CodeBlocks
- „gcc“ mit Makefiles

## C#



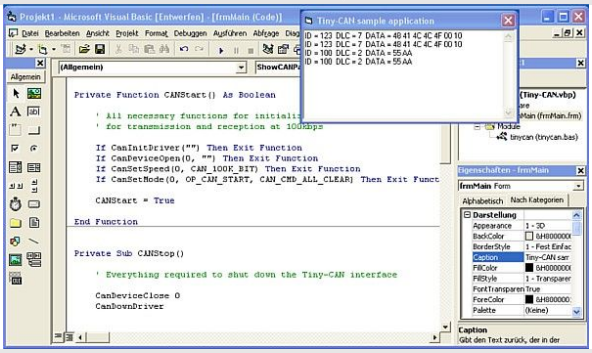
Die Klasse „TinyCan“ (can_drv.cs) bildet die gesamte Tiny-CAN API im Stil von C# ab.

Die Beispielprogramme veranschaulichen:

- Empfang und Senden von CAN Nachrichten
- Ereignissteuerung über Callbackfunktionen
- Setzen von Filtern
- Versand Zyklischer CAN Nachrichten
- Abfrage der verwendeten Hardware und deren Features
- Event Funktionen

**Microsoft Visual C# Homepage: msdn.microsoft.com/de-de/vcsharp**

## Visual Basic Ver. 6 und VB2013

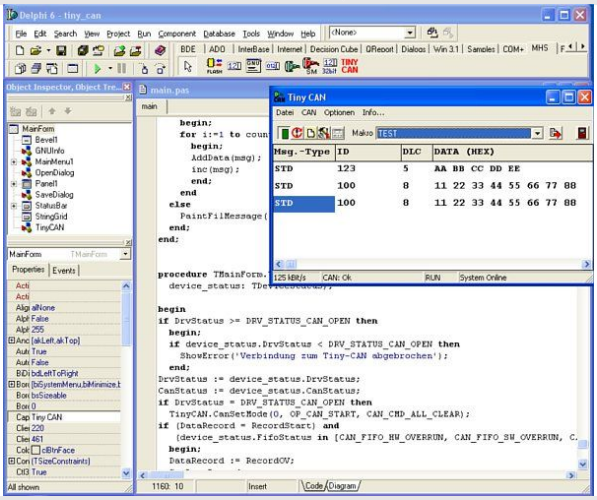


Die Einbindung der Tiny-CAN API erfolgt in Visual Basic mit der Datei „tinycan.bas“, bzw „tinycan.vb“, der volle Fuktionsumfang der Tiny-CAN API wurde in VB nicht abgebildet.

Das Sample öffnet ein CAN-Device mit 100kBit/s und zeigt die Empfangenen CAN Nachrichten an.

**Microsoft Visual Basic Homepage: msdn.microsoft.com/de-de/vbasic**

# Delphi



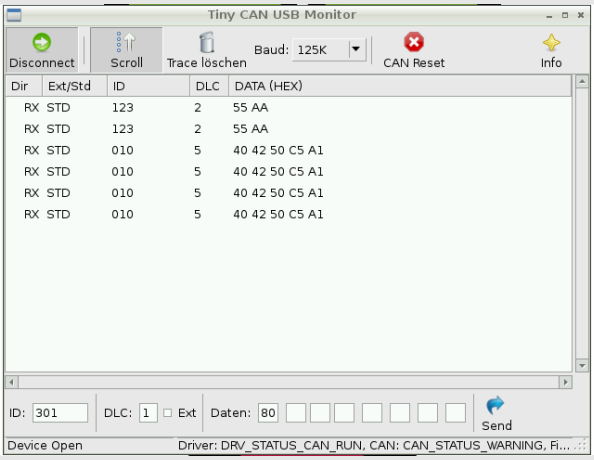
Die Einbindung der Tiny-CAN API erfolgt in Delphi mit einer Komponente, die den vollen Funktionsumfang der API zur Verfügung stellt.

Die Funktionalität der Komponente wird anhand mehrerer Beispiele veranschaulicht.

Versionen:
- Delpi 6
- Delphi EX

**Delphi Homepage: www.embarcadero.com**

# Python



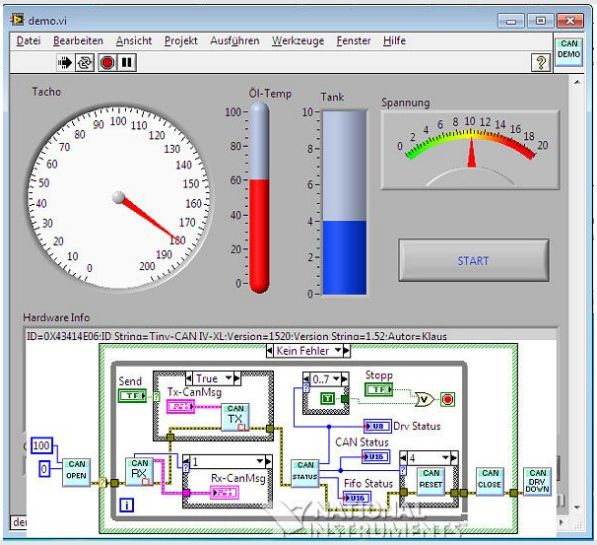

Die Einbindung der Tiny-CAN API erfolgt in Python mit der Datei „mhsTinyCanDriver.py“. Alle Funktionen der API werden in Python abgebildet.

Die Beispielprogramme demonstrieren:
- Kleiner grafischer CAN-Monitor
- Empfang und Senden von CAN Nachrichten
- Ereignissteuerung über Callbackfunktionen
- Setzen von Filtern
- Versand Zyklischer CAN Nachrichten
- Öffnen des CAN-Device im „Listen Only Mode“

Versionen:
- 2.7.x
- 3.x.x

**Python Homepage: www.python.org**

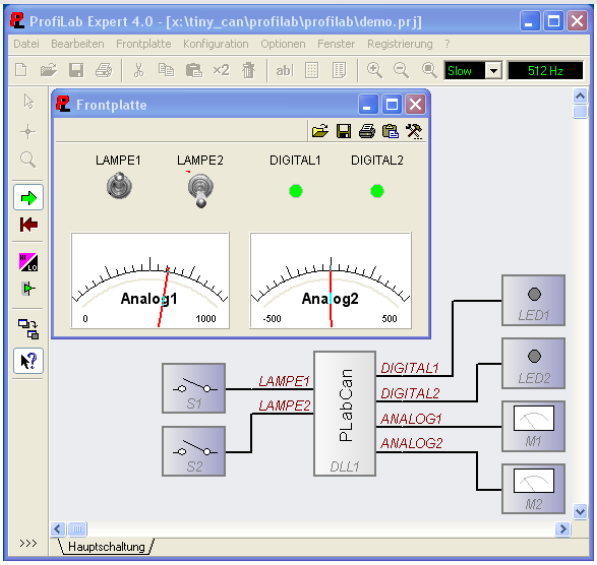
# LabView




Die auf einer Wrapper-DLL aufbauende Library enthält die „VIs“ welche die Funktionen der API widerspiegeln.
4 Applikationsbeispiele veranschaulichen die Funktionalität und Verwendung der Library.

**National Instruments Homepage: www.ni.com/labview**

## Profilab CAN Plugin

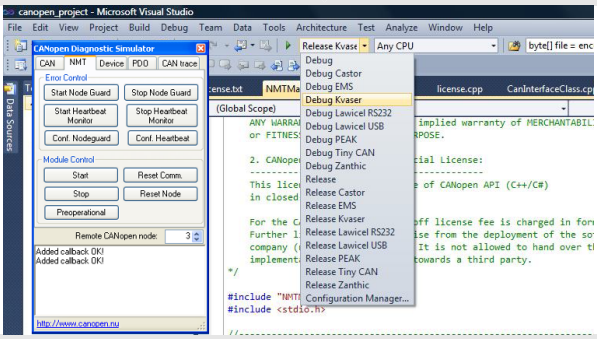



Mit wenigen Mausklicks werden CAN-Botschaften in Signale aufgeteilt, die dann als analoger oder digitaler Ein/Ausgang im Schaltungssymbol erscheinen. Auch das verarbeiten von "Muxer"-Nachrichten ist ohne Probleme möglich. Die einzelnen Signale können dann mit allen im ProfiLab enthaltenen Elementen verbunden werden, z.B. Schalter, Taster, Displays, Analoge Zeigerinstrumente usw. Die Programmierung erfolgt auf rein grafischer Ebene.

ProfiLab lässt sich nicht nur zur Visualisierung, sondern auch zur Steuerung einsetzen. Zahlreiche Bauteile zur Verknüpfung und Verarbeitung von Signalen stehen zur Verfügung. Mit dem integrierten Compiler ist es möglich, fertige Projekte sozusagen in Stand-Alone-Anwendungen zu verwandeln. Diese, von der ProfiLab-Software erstellten Anwendungen, können Sie auf jedem Windows-PC starten, ohne dass dort die Originalsoftware installiert sein muss.

**ABACOM Ingenieurgesellschaft Homepage: www.abacom-online.de**

## CANopen Interface SDK (C# / C++)




CANopen Interface SDK ist eine C++/C# Library zum Entwickeln eigener CANopen Applikationen auf dem PC. Das Paket beinhaltet SDO client/server, NMT master/slave und eine „raw“ CAN API, es wird eine Vielzahl von CAN Interfaces unterstützt.
Die Library ist OpenSource und steht unter der GPL v.3.0 Lizenz.

**Weitere Informationen: http://www.datalink.se**

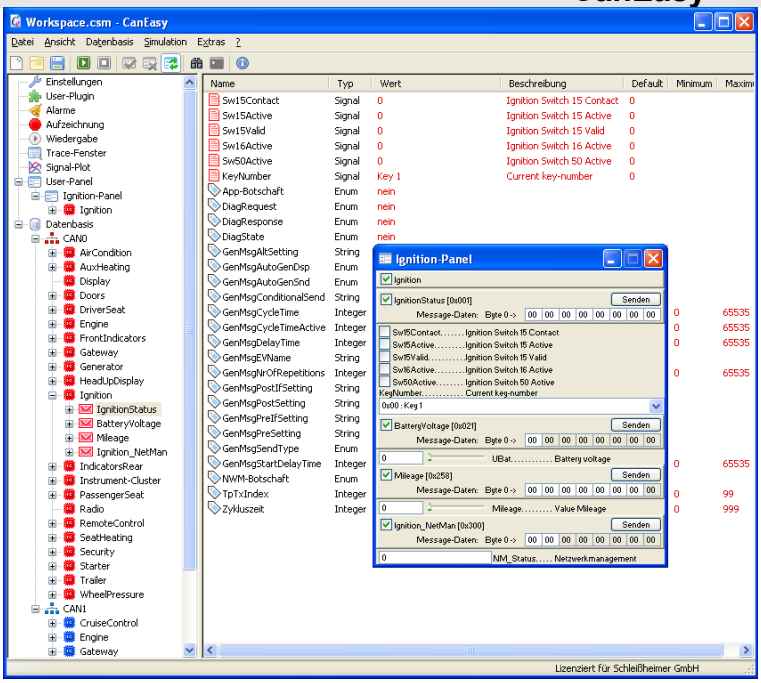
## Unterstützte Betriebssysteme:

| | |
|---|---|
| Ab WindowsXP | www.microsoft.com |
| Linux ab Kernel 3.xx | www.linux-foundation.org |
| Apple OS X ab Version 10.4 (Tiger) | www.apple.com |

Alle Betriebssysteme werden in der 32-Bit und 64-Bit Version unterstützt.

# 7. Third Party Tools

## CanEasy



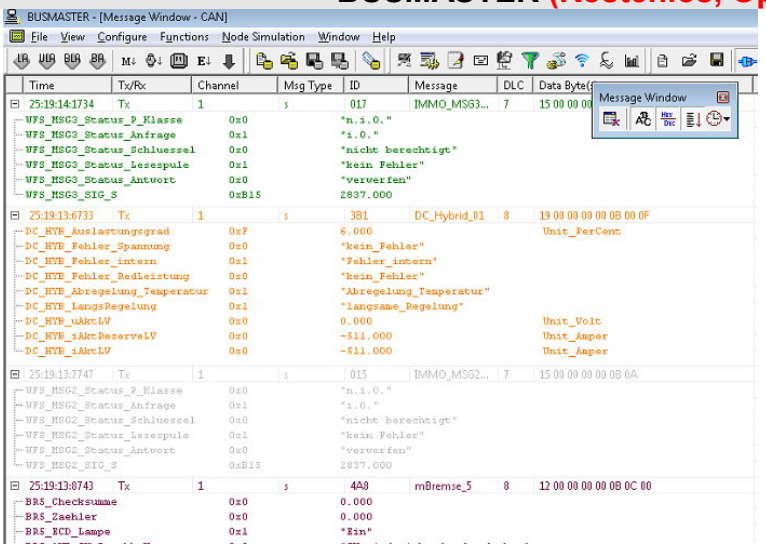

CanEasy 3.0 ist eine windowsbasierte Simulations-, Analyse- und Testumgebung für Entwicklung und Test von CAN-Bus-Systemen. Einfachheit ist das oberste Prinzip beim Einrichten, Bedienen und Erweitern des Systems.

Die Vorteile von CanEasy:

- Restbussimulation: Simuliert Botschaften und Signale fehlender Steuergeräte
- wird automatisch generiert und weitgehend konfiguriert – keine langwierige Vorbereitung und Programmierung
- lässt sich komfortabel bedienen über Baumstruktur, drag & drop und Kontextmenüs
- besitzt intuitive Kernanwendungen für die Analyse: User-Panels, Signal-Plots, Trace-Fenster
- gibt mehrtägige CAN-Aufzeichnungen mehrerer CAN-Busse mit allen Botschaften vollständig oder gefiltert wieder
- ermöglicht softwarebasiertes Testen realer Steuergeräte
- integriert neue Funktionen, Regeln und Ansichten über Plug-Ins

**Weitere Informationen: http://www.caneasy.de**

## BUSMASTER (Kostenlos, Open Source)



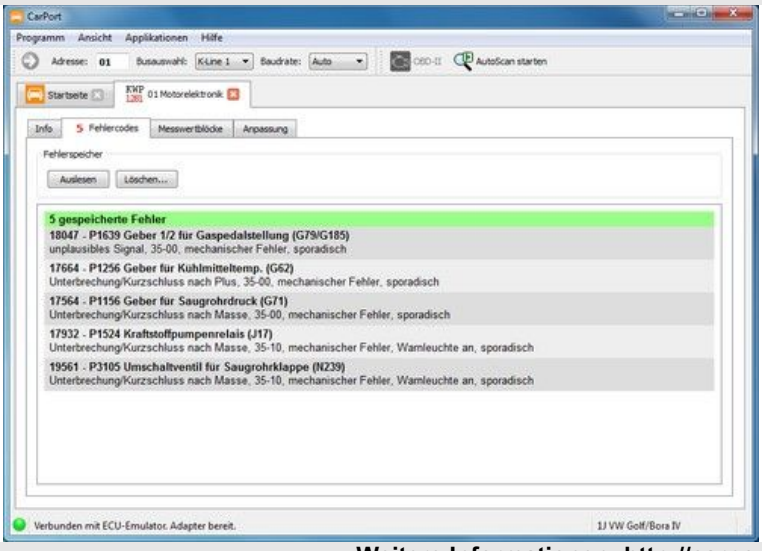

BUSMASTER ist eine Open-Source-PC-Software für den Entwurf, die Überwachung, die Analyse und die Simulation von CAN-Netzwerken. BUSMASTER basiert auf dem Softwarewerkzeug CANvas und wurde von RBEI konzipiert, entworfen und entwickelt.

Funktionen auf einen Blick

- Erstellung und Bearbeitung von CAN-Datenbasen
- Filtern von Messages durch Hardware oder Software
- Aufzeichnung und Wiedergabe von CAN-Messages
- Programmieren von Knoten im ANSI C Funktionseditor
- Importfilter für DBC-Datenbasen und CAPL-Programme

**Weitere Informationen: http://rbei-etas.github.com/busmaster**

## CarPort




Software für die KFZ-Diagnose
**KOSTENFREIE EDITION erhältlich**

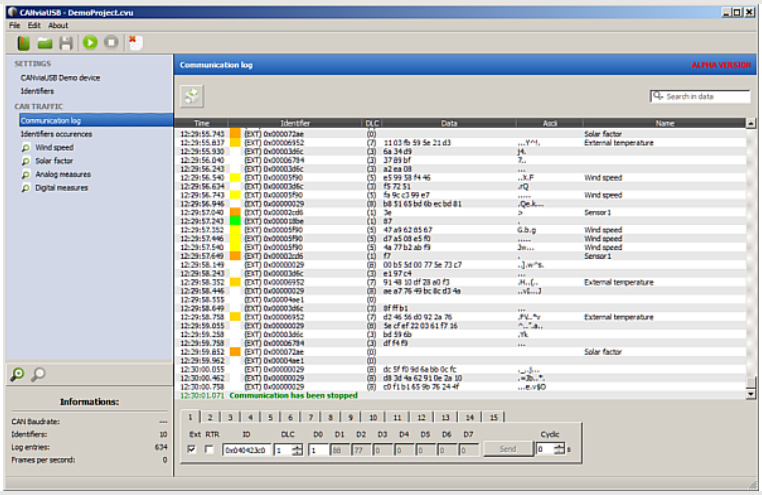
Fehlerbeschreibungen
CarPort zeigt tausende volkswagenspezifische Fehlercodes inklusive Beschreibung an.

Steuergerätescan
CarPort bietet den ersten vollintegrierten Steuergerätescan - so erhalten Sie schnell einen kompletten Überblick über den Zustand aller verbauten Steuergeräte.

Grafische Ausgabe
CarPort kann mehrere Messwerte gleichzeitig grafisch darstellen - als Kurve oder in einer Instrumentenansicht.

**Weitere Informationen: http://carport-diagnose.de**

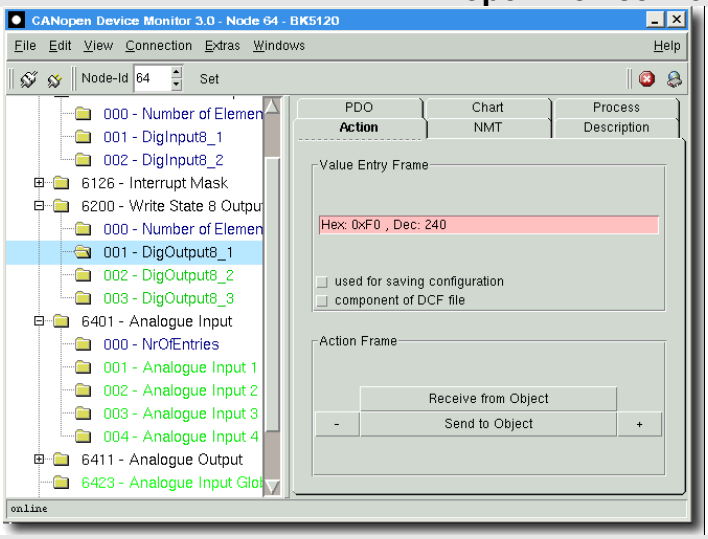
## CANviaUSB




Ein einfacher CAN Monitor. CAN ID´s können Namen und Farben zugeordnet werden. Tolle Filter Funktion.

**Weitere Informationen: http://www.canviausb.com**

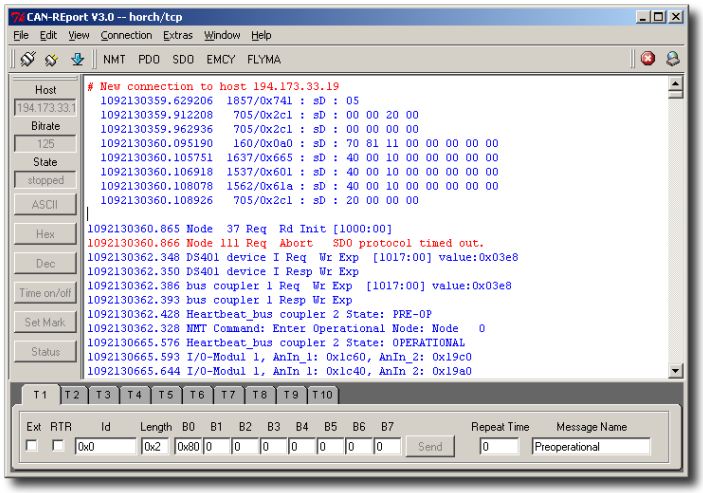
## CANopen Device Monitor




Der CANopen Device-Monitor von port dient der grafischen Inspektion und Konfiguration von CANopen Geräten im CANopen-Netzwerk. Die integrierte Skriptfähigkeit erlaubt neben dem Zugriff auf die implementierten CANopen-Dienste, die Ausführung von Steueraufgaben im Netzwerk und die Realisierung von Test- und Steuerapplikationen mit geringem Aufwand.
Die Informationen zur Darstellung des Geräte-Objektverzeichnisses werden der EDS-Datei des Gerätes entnommen (EDS - Electronic Data Sheet) oder können direkt vom Gerät gescannt werden. Spezielle Gerätekonfigurationen können als DCF-Dateien geladen, gespeichert und gescannt werden. Zusätzlich können Konfigurationen ganzer Netzwerke in Projektdateien gespeichert werden.

**Weitere Informationen: http://www.port.de**

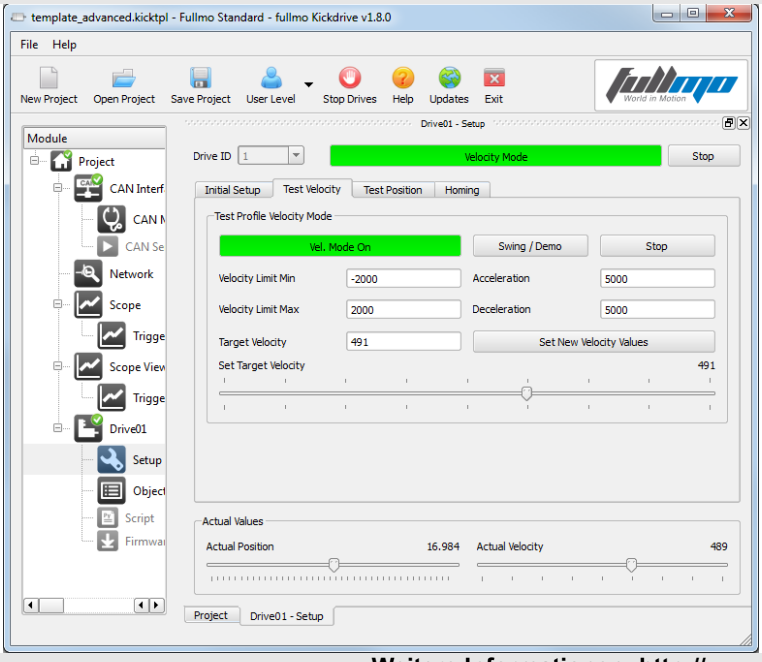
## CAN-REport




Der CAN-Analyzer CAN-REport von port ist ein leistungsfähiges und vielseitig einsetzbares Werkzeug zur Analyse und Inbetriebnahme von CAN-basierenden Netzwerken nach CAN 2.0A und 2.0B, wie CANopen und DeviceNet.
Die integrierte Skriptfähigkeit erlaubt neben dem einfachen Aufzeichnen von CAN-Telegrammen eine universelle Verwendung bei Entwicklung, Test und Wartung, vor allem im Bereich der industriellen Automatisierungstechnik.
Durch die Trennung von Hardware-Interface (CAN-Anbindung) und Visualisierungssoftware wird ausserdem die Anwendung über TCP/IP-Netzwerke ermöglicht.

**Weitere Informationen: http://www.port.de**

## fullmo Kickdrive




fullmo Kickdrive ist eine Windows-Anwendung für Konfiguration und Test von CAN/CANopen-Netzwerken, CAN/CANopen Positionierantrieben und anderen Feldbuskomponenten. Es kann für die unterschiedlichsten Konfigurations-, Inbetriebsetzungs- und Serviceaufgaben eingesetzt werden. fullmo Kickdrive ist einfach zu benutzen, portabel, nahezu beliebig skalierbar, sowie in hohem Maße programmierbar über Python Skript und die Qt Quick GUI-Sprache.

**Weitere Informationen: http://www.kickdrive.de**

# 8. Links

| | | |
|---|---|---|
| MHS-Elektronik | www.tiny-can.de | Die Tiny-CAN Projekt-Seite |
| CAN in Automation e.V. | www.can-cia.org | Alle erdenklichen Informationen über CAN, CANopen, CAN-Bus Normen, Hardware, usw. |
| GTK | www.gtk.org | Die Homepage der GTK Organistation |
| Glade | glade.gnome.org | Glade ist eine freie visuelle Programmierumgebung (Rapid Application Development, RAD) zum intuitiven Erstellen von GTK+-Benutzeroberflächen (GUI). |
| Code::Blocks | www.codeblocks.org/ | Code::Blocks ist eine OpenSource, Cross Platform C++ IDE<br>Tiny-CAN-Monitor wure damit erstellt |
| GIW | sourceforge.net/projects/giw | GTK Instrumentation Widgets<br>Wird in den Demo Plugin verwendet |
| MegaTunix | megatunix.sourceforge.net | Ein OpenSource Tuning Projekt, die Anzeigeinstumente sind sehr schön. |
| Roland Bock | www.eudoxos.de | GtkDatabox ist mal einen Blick wert |
| FTDI | www.ftdichip.com | Treiber und Informationen zum USB Chip |

# 9. Danksagungen

| | |
|---|---|
| Für die Implementierung der Tiny-CAN API in C# | Herr Thomas Forck <forck@kloecker-gmbh.com><br>Firma: Gebr. Klöcker GmbH (http://www.kloecker-gmbh.com) |
| Für die Implementierung der Tiny-CAN API in Python | Herr Rene Maurer <rene@cumparsita.ch><br>Firma: Omnitron Engineering & Software (http://www.omnitron.ch)<br>Herr Patrick Menschel <menschel.p@posteo.de> |
| Für die Implementierung der Tiny-CAN API in Visual-Basic Version 6.0 | Tobias Schleicher <t.schleicher@medos-ag.com><br>Firma: MEDOS Medizintechnik AG (http://www.medos-ag.com) |