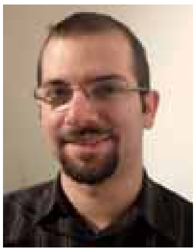
CAN driver for Windows with analyzer capability

Thread-safe virtualization, the access to .NET, and the built-in analyzing tool of a CAN driver for C/C++ enable an object-oriented software development of CAN applications.

Authors



Martin Andermann

Hartmut Keller

F&S Elektronik Systeme GmbH
Untere Waldplätze 23
DE-70569 Stuttgart
Tel.: +49-711-123-722-0
Fax: +49-711-123-722-99
info@fs-net.de

Link
www.fs-net.de

Despite the fact that CAN has its origins in the automotive sector, it has found its way to the embedded market. This is especially the case when sensors have to be queried or devices have to be controlled and if focus is on high transmission security. Today CAN is also used in medical devices or automation technology. F&S Elektronik Systeme is a manufacturer of embedded systems in the form of single board computers (SBC) and system-on-modules (SOM). These provide at least one CAN interface and are available with Linux, as well as Windows Embedded Compact operating systems.

System-on-chip manufacturers offer CAN drivers for Windows, but usually only with support of their own controllers, resulting in drivers that differ from each other. F&S has developed a CAN driver for C/C++, which offers the same interface on all manufacturer's boards, regardless of the used CAN controller. They also offer a suitable class library for .NET, simplifying the use of the driver with e.g. C#. The CAN driver provides an object-oriented interface to the driver, including CAN interface construction and destruction, support for exceptions in case of errors and taking advantage of overloaded functions.

The driver uses the concept of virtual send and receive channels. These channels give all applications and threads their own access to the CAN network. All channels are independent from each other. Each channel has its own transmit and receive buffer, as well as a separate acceptance filter. For example, on multi-core CPUs (central processing unit) the efficiency of the message processing can be increased by using multiple parallel threads. Each thread opens its own file handle, provides its own acceptance filter, and then receives its own messages.

Several distinct programs can access the CAN controller, but since a CAN controller usually does not receive its self-sent messages, a message sent on the network by one of these programs cannot be seen by other ▷

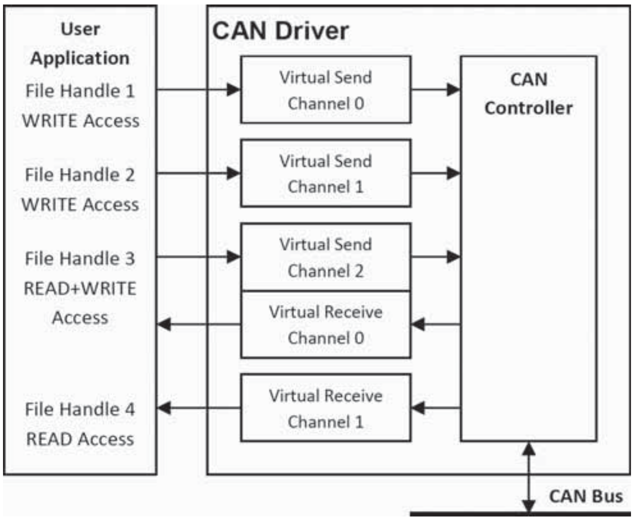

User Application
File Handle 1 WRITE Access
File Handle 2 WRITE Access
File Handle 3 READ+WRITE Access
File Handle 4 READ Access
CAN Driver
Virtual Send Channel 0
Virtual Send Channel 1
Virtual Send Channel 2
Virtual Receive Channel 0
Virtual Receive Channel 1
CAN Controller
CAN Bus

Figure 1: CAN driver with virtual channels for read and write access

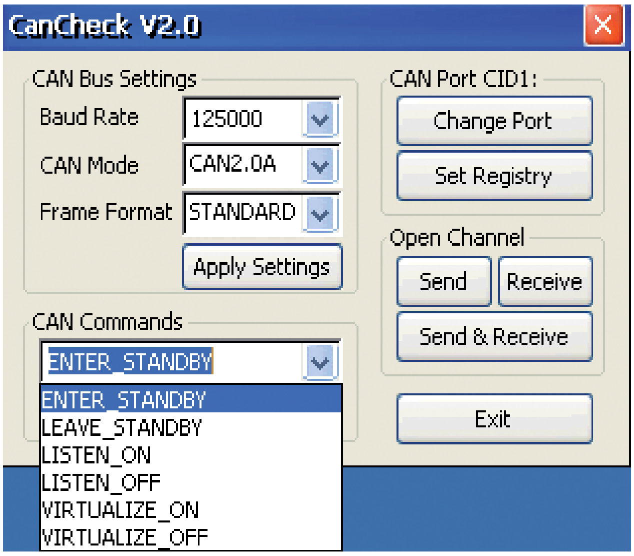
local programs. To overcome this disadvantage, the driver can be set into a so-called Virtualize mode. Now the messages are not only sent out to the physical CAN network, but are also forwarded by the driver to the receive channels of all the other local programs. From the programs' point of view, each of them now has its own (virtual) CAN controller. The Virtualize mode can be (de-)activated during the runtime. The CAN driver also offers the possibility to switch to a Listen Only mode. In this case, the transmitter of the controller gets deactivated. This prevents an accidental sending of messages.

The Enter Standby command puts the CAN controller into a sleep mode to reduce electrical consumption. This mode is left automatically, when there is communication on the network, or manually, with the Leave Standby command. All these mode changes are reported as events to all virtual receive channels.

To find out whether there is a communication malfunction on the CAN network or an error in the own software, a CAN analyzer, in form of external hardware, is often required. The built-in CanCheck tool in the driver makes such an analyzer unnecessary in many cases. It can either be set as a communication partner on a second board, or directly on the development board. The tool can be used like a sniffer program, showing the live traffic on the CAN network. The received data can be saved in a log file for later analysis. The tool can also generate arbitrary CAN messages for test purposes. It is a graphic tool, which is operated by mouse or touch. During development, the focus was put on compact dialogs, so the program is also applicable on small displays. If there is no display available, company's boards use a virtual display, which can be made visible on a PC with a remote desktop connection (e.g. CerDisp). Used in this way, the tool can be controlled remotely.

In the "CAN Bus Settings" field of the tool's main window, the user can adjust the network settings, for example the CAN port (if there is more than one controller available), the used bit-rate, or whether the frames with 11-bit or 29-bit CAN-ID (identifier) are used. The settings can be permanently saved directly from the dialog into the Windows registry. The "CAN Commands" such as Standby, Listen Only and

▷



CanCheck V2.0

CAN Bus Settings

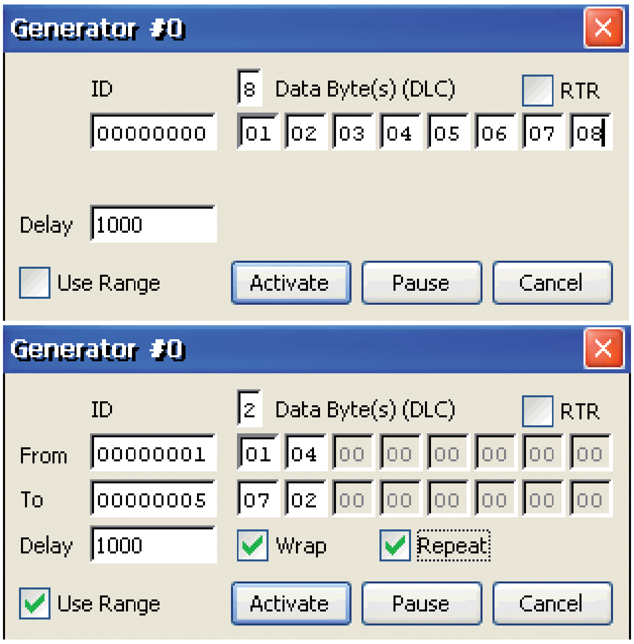
Baud Rate: 125000

CAN Mode: CAN2.0A

Frame Format: STANDARD

CAN Commands: ENTER_STANDBY

Figure 2: The main window



Generator #0

ID: 00000000

8 Data Byte(s) (DLC): 01 02 03 04 05 06 07 08

□ RTR

Delay: 1000

□ Use Range

Generator #0

ID From: 00000001

ID To: 00000005

2 Data Byte(s) (DLC)

From: 01 04 00 00 00 00 00 00

To: 07 02 00 00 00 00 00 00

□ RTR

Delay: 1000

☑ Wrap ☑ Repeat

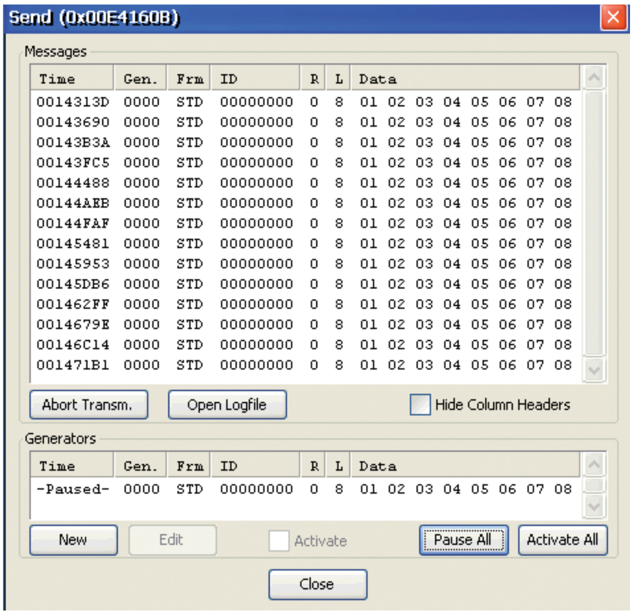
☑ Use Range

Figure 3: The send generator window

Virtualize can be selected and sent from a drop-down list. Every click on one of the buttons in the "Open Channel" field creates a new send channel, a receive channel, or a combined send/receive channel each in its own window.

The send functionality of the tool offers the possibility to create multiple so-called send-generators. The generators run separately and can also be paused or activated separately. The easiest case when creating a generator is to transfer a message with a constant CAN-ID and up to eight constant data bytes. By defining a start and end value, it is also possible to automatically increment and decrement CAN-ID and data bytes individually. The wrap-over function and repetitions can be activated. The "Delay" value defines the pause between the separate CAN messages of each generator. By setting the RTR (remote transmission request) flag, it is possible to send CAN request frames.

In the receive window, one can either see the complete communication on the CAN network, or messages with a certain CAN-ID (or CAN-ID range) only. This can be done by adjusting the acceptance filter. The tool decodes the acceptance filter to show which bits are actually checked (0/1) or generally accepted (+). Different CAN events like "Message received", mode adjustments, as well as errors like "Overrun" or "Arbitration lost" are visualized with different symbols. Toggling the buttons with the event symbols enables or disables the appropriate event, i.e. the event is shown in the events list or it is ignored. This functionality can be used to reduce the list to error messages only. ◀



Send (0x00E4160B)

Messages

Time	Gen.	Frm	ID	R	L	Data
0014313D	0000	STD	00000000	0	8	01 02 03 04 05 06 07 08
00143690	0000	STD	00000000	0	8	01 02 03 04 05 06 07 08
00143B3A	0000	STD	00000000	0	8	01 02 03 04 05 06 07 08
00143FC5	0000	STD	00000000	0	8	01 02 03 04 05 06 07 08
00144488	0000	STD	00000000	0	8	01 02 03 04 05 06 07 08
00144AEB	0000	STD	00000000	0	8	01 02 03 04 05 06 07 08
00144FAF	0000	STD	00000000	0	8	01 02 03 04 05 06 07 08
00145481	0000	STD	00000000	0	8	01 02 03 04 05 06 07 08
00145953	0000	STD	00000000	0	8	01 02 03 04 05 06 07 08
00145DB6	0000	STD	00000000	0	8	01 02 03 04 05 06 07 08
001462FF	0000	STD	00000000	0	8	01 02 03 04 05 06 07 08
0014679E	0000	STD	00000000	0	8	01 02 03 04 05 06 07 08
00146C14	0000	STD	00000000	0	8	01 02 03 04 05 06 07 08
001471B1	0000	STD	00000000	0	8	01 02 03 04 05 06 07 08

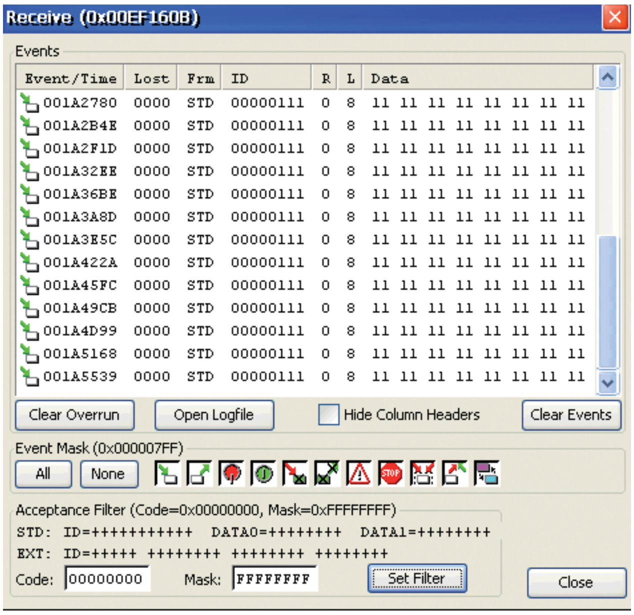
□ Hide Column Headers

Generators

Time	Gen.	Frm	ID	R	L	Data
-Paused-	0000	STD	00000000	0	8	01 02 03 04 05 06 07 08

□ Activate

Figure 4: The send window



Receive (0x00EF160B)

Events

Event/Time	Lost	Frm	ID	R	L	Data
001A2780	0000	STD	00000111	0	8	11 11 11 11 11 11 11 11
001A2B4E	0000	STD	00000111	0	8	11 11 11 11 11 11 11 11
001A2F1D	0000	STD	00000111	0	8	11 11 11 11 11 11 11 11
001A32EE	0000	STD	00000111	0	8	11 11 11 11 11 11 11 11
001A36BE	0000	STD	00000111	0	8	11 11 11 11 11 11 11 11
001A3A8D	0000	STD	00000111	0	8	11 11 11 11 11 11 11 11
001A3E5C	0000	STD	00000111	0	8	11 11 11 11 11 11 11 11
001A422A	0000	STD	00000111	0	8	11 11 11 11 11 11 11 11
001A45FC	0000	STD	00000111	0	8	11 11 11 11 11 11 11 11
001A49CB	0000	STD	00000111	0	8	11 11 11 11 11 11 11 11
001A4D99	0000	STD	00000111	0	8	11 11 11 11 11 11 11 11
001A5168	0000	STD	00000111	0	8	11 11 11 11 11 11 11 11
001A5539	0000	STD	00000111	0	8	11 11 11 11 11 11 11 11

□ Hide Column Headers

Event Mask (0x000007FF)

Acceptance Filter (Code=0x00000000, Mask=0xFFFFFFFF)

```
STD: ID=+++++++++++  DATA0=++++++++  DATA1=++++++++
EXT: ID=+++++ ++++++++ ++++++++ ++++++++
```

Code: 00000000 Mask: FFFFFFFF

Figure 5: The receive window

ifm electronic

Your engine has been talking to you for years...

Visit us at
SPS/IPC/Drives 2014
hall 7A · stand 7A-302

...now it's time to listen.

ifm mobile controllers and displays – connection via SAE J 1939 allows both reading data and controlling the engine.
All devices are freely programmable to IEC61131-3 with CODESYS.
Pre-written SAE J 1939 libraries to aid with software development.
Compliant to Tier 3, Tier 4 and Tier 4 final as well as up to Euro 6.

IP 65 IP 67 · Resistant EMC · Resistant · Temperature range −40...+85°C · Vibration and shock resistant · E1 · SIL 2

ifm – close to you!

www.ifm.com/gb/j1939