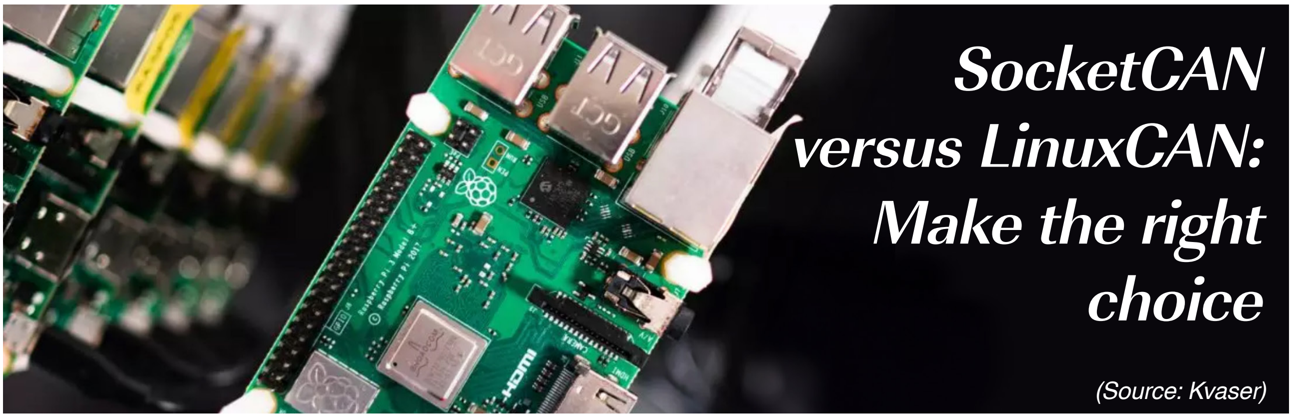

SocketCAN versus LinuxCAN: Make the right choice

(Source: Kvaser)

Developing a CAN system, you need something to interface between CAN nodes and a computer that will send commands to the nodes and receive data from them. You are using a device from Kvaser. The project dictates you are using a Linux computer (e.g. Raspberry Pi or Ubuntu). How do you get the Kvaser device to connect to the PC?

In order to connect a PC to the CAN network, you need to install drivers. Since you are using Linux, there are two options: LinuxCAN drivers or SocketCAN drivers. Is there a difference? Yes, and which you pick depends on what it is you want to do.

LinuxCAN is Kvaser's own set of packaged drivers for Linux. They are packaged with a copy of the Kvaser CANlib SDK (software development kit) for custom programming with our API (application programming interface). One installation contains drivers for all of our devices (excluding Ethernet devices).

SocketCAN is a set of open-source CAN drivers and a networking stack, which is included in many Linux distributions. In other words, SocketCAN comes pre-installed and you just need to set SocketCAN up. SocketCAN contains the drivers of more than just Kvaser devices. SocketCAN uses the Berkeley socket API, the Linux network stack, and implements the CAN device drivers as network interfaces. A complete list of Kvaser devices available for use with SocketCAN is given on the Kvaser website.

Kvaser contributes driver updates for SocketCAN, but Kvaser does not maintain SocketCAN. On the Kvaser website, you can find beta drivers for SocketCAN. These are drivers that will be used to make updates to SocketCAN in the Linux kernel, but may not have been approved or included in the latest kernel. Kvaser hardware support is dependent on the Linux Kernel version since we are dependent on the SocketCAN group for when updates are included that recognize newer hardware.

What's the difference?

Both SocketCAN and LinuxCAN allow you to use a Kvaser device on a Linux machine. Here are some key differences that can help you decide what to use:

- SocketCAN is a collection of network socket based drivers. It is like a generic driver. You can think of SocketCAN like this: When you plug in a new mouse and it is registered as "Generic Mouse". SocketCAN provides the functionality to access a CAN network. SocketCAN allows for a different device to be swapped as needed, even if the device is from another company. SocketCAN commands can also be run from the bash command prompt, meaning you can set up basic send and receive functions from the command terminal.
- LinuxCAN matches the Kvaser Windows API. The API is specific to Kvaser and the drivers are specific to Kvaser devices. With that, you can access more features that can be programmed with the LinuxCAN SDK. These features can range from programming the LEDs on a unit, to adding Kvaser Memorator extraction commands. LinuxCAN also supports LIN functionality, which SocketCAN does not. However, LinuxCAN does not allow you to run commands from the bash prompt in a terminal window.

Can both driver sets be used together?

Yes and no. First, when LinuxCAN is installed, the installer blacklists the SocketCAN drivers for Kvaser products. This is done to prevent driver conflicts. This means that the Kvaser devices could only use the LinuxCAN drivers. That being said, you can still use another company's device with the SocketCAN drivers. Kvaser API commands would still only work with the Kvaser device, and SocketCAN commands would only work with the devices using the SocketCAN drivers.

When should I use one over the other?

This question is purely project dependent. If you have a project that needs to make use of functionality that is specific to a Kvaser device, then LinuxCAN would be the better choice. If you need something where the CAN interface could be swapped out, or needed similar functionality, then SocketCAN may be the better choice. ▷

Brief news: Tools

- ***CAN troubleshooting:*** The CANBUSview XL III by Indu-Sol (Germany) enables physical analysis of CAN CC networks including bar diagrams and oscilloscope features. Logical monitoring of the data link layer protocols and data recording is possible. CANopen, DeviceNet, J1939, and Safetybus p are supported.
- ***MCU debug engine:*** The Universal Debug Engine (UDE) from PLS (Germany) offers interactive debug functions and visualization options while software development and testing of MCUs. Now, it supports the CAN-FD-capable S32M2 motor control units from NXP used in automotive body and comfort applications. Demos of the improved UDE 2024 also supporting the Infineon's Traveo T2G (and further) MCUs are presented at Embedded World 2024 in Nuremberg.
- ***Sensor monitoring:*** Parker Sensocontrol (Germany) introduced the Parker Service Master Compact measuring device that provides real-time monitoring and diagnostics in mobile equipment and industrial hydraulics applications. The IP65-rated device provides connections for up to six Parker CAN (and other) sensors.
- ***Software-in-the-loop tests:*** Vector Informatik (Germany) offers software-in-the-loop tests on real target hardware, which does not have to be the final device. This allows software components with strong hardware dependency to be tested at their functional interfaces. This feature is integrated in CANoe4SW and CANoe.

of

Summary

Here is a short overview on the functionality of the both:

SocketCAN:

- Provides drivers for Kvaser as well as other companies.
- Available on the Linux kernel – no installation required.
- Commands can be run from the bash prompt.
- No LIN capability.

LinuxCAN:

- Specific to Kvaser devices to allow advanced functionality.
- Blacklists Kvaser SocketCAN drivers to avoid conflicts.
- Commands cannot be run from the bash prompt.
- Has LIN capability.

Author

Adam Raymer
Kvaser
adra@kvaser.com
www.kvaser.com