The possibilities of a bus system can only be fully utilized in combination with powerful software tools for development, diagnostics, and support. In parallel with the rise and advancement of the CAN network, therefore a significant infrastructure comprising tools for bus analysis, management of communication data, development, and simulation of bus nodes and much more has become established. This article leads us from the modest beginnings of the CAN tools and the past challenges to the currently achieved state and the emerging trends for the future.

When the first vehicles with CAN in the drive train went into series production in 1990, even specialists were surprised by the robust and trouble-free operation demonstrated by the new bus system in actual use. Troubleshooting in the modern vehicles, however, proved to cause considerable problems for developers and repair shops. This was due, on the one hand, to the lack of know-how in the area of bus technology and networked electronics and, on the other hand, to the lack of suitable tools for bus diagnostics. In some cases, this helplessness resulted in vehicles needing to be sent to the factory for error analysis. There, the development engineers used provisional tools to try and localize errors and document the procedures.

Work on bit-level and with hex representations was, at this point in time, both inefficient and without alternative.

First technically adequate CAN tool

The engineers at Vector soon recognized this problem and began developing a user-friendly CAN tool for the motor vehicle sector in 1991. The application area of this software with the catchy name “CANalyzer” covered both development as well as troubleshooting. With completion of the first prototype in spring of 1992, Vector learned from the press about the founding of CAN in Automation (CiA), whose original roots lie in general automation. After immediately contacting Holger Zeltwanger, CiA co-founder, Vector became a very early member of the CAN user organization in April 1992. On the periphery of iNet in Sindelfingen, Germany, in June of 1992, the first CiA meeting with Vector in attendance was held. At the same time, the company presented CANalyzer to the public for the first time at this trade fair, a tool that finally allowed developers to work in a comfortable, technically adequate manner. The author of this article regularly gave lectures at CiA events and held training courses on the topic of CAN in the following years.
The computer and software world of that time was significantly different from that of today. Typical PCs were IBM PC/AT compatible computers equipped with 80386-CPU, clock speeds of 25 MHz and 1 MiB of RAM, while the operating system of choice was called MS-DOS. Portable computers also existed at that time, but they were truly heavy “boxes” which looked more like a sewing machine than a computer system (Figure 2). At nearly the same time as the CANalyzer presentation, Microsoft Windows 3.1 was launched on the market in early 1992. The predecessor versions of this operating system were found to be much too slow and unreliable for technical software with real-time requirements.

Vector’s CAN tool was still based on MS-DOS, but had a self-developed and fast graphical operating system with window management that had already proven itself in calibration tools developed for Bosch. This was a decisive advantage over the text-based MS-DOS tools from other producers, as it offered much better user guidance. Installation was very simple: The user simply copied the approximately 1 MiB of content located on a diskette to a PC directory, selected the correct graphics driver (Hercules, EGA, or VGA) and started an EXE file. Today, a single digital photo is about four times larger than the content of the installation diskette of the first CANalyzer version. For bus coupling, only ISA cards from Bosch and Daimler-Benz were initially available. For this purpose, the specialist secured the sales rights, in order to supply other users of CAN tools with the right hardware as well.

Depending on one's perspective, the first version of the tool can be characterized, on the one hand, as spartan and, on the other hand, as rather progressive. For example, the functionality of the first interface boards with the so-called FullCAN controllers (Intel AN82526) was still noticeably restricted. It only allowed the reception of messages whose identifiers the developer had previously entered in the object list of the controller. For the automotive ECUs of the time, the automatic reception of CAN messages on this basis was a great advantage, as the low-performance micro-controllers were not unnecessarily burdened. Universally usable CAN tools were, however, severely impaired as a result. Progressive, on the other hand, was the ability to freely program the tool. The self-developed programming language, CAPL, enabled developers and testers to both generate as well as manipulate CAN traffic. With the help of simple scripts, it was – even at that time – possible to generate correct as well as faulty data traffic for testing and stimulating test samples.
A great improvement in convenience came just a year later in 1993 with interface cards that were specially developed for CAN tools. They worked with simpler CAN controllers and could receive all messages. This, in turn, then required “high-performance” micro-controllers on the cards. Even if it might be a matter of course today, at this time it was a massive challenge to transfer the data of two high-speed CAN networks to the PC with 1 Mbit/s and 100 % bus loading without loss of messages.

The Vector tools brought a significant improvement in operation and convenience with the implementation of the CANdb data format. It is used to store and manage all parameters, messages, and signals of pre-configured CAN networks. In particular, the system permits the use of symbolic designations for messages and signals and it supports the conversion of raw bus data into physical quantities. Temperatures, for example, can be presented directly in degrees Celsius instead of in cryptic binary values, the enumeration values of codes can be displayed and much more. CANdb now gave users the ability to work with expressive and familiar designators, making coding details and ambiguous data fields in hexadecimal numbers a thing of the past. Other tool producers quickly realized this as well. As a result, CANdb soon became the de facto industry standard for pre-configured CAN networks.

The simulation of not yet implemented ECUs’ bus traffic and even complete network topologies with CANoe (CANopen environment) in 1995 marked another milestone in the history of CAN tools. For ECU developments, these remaining bus simulations are the prerequisite for realistic tests which provide information on the behavior and maturity level of the test sample, even if the final environments do not yet exist. In the meantime, Windows had established itself as the dominant PC operating system and the significantly faster Pentium-based computers finally allowed it to be used with tools.

From specialized CAN profile to multibus tool

Through the rapid proliferation of the CAN network in both the automotive sector and in areas such as factory automation, medical technology, railway technology, and commercial vehicle technology or aviation, numerous CAN-based higher-level protocols and profiles became established. Vector responded to this development with a series of protocol-specific variants of its CAN tools. Among the most important were the options SDS (Honeywell), Devicenet (Allen Bradley), and CANopen (CiA) for automation technology, J1939 for the commercial vehicle sector as well as CANaero for the aviation industry.

In parallel, the automotive industry developed the specialized network systems LIN, MOST, and Flexray, which were no longer based on CAN. It was now a matter of expanding the existing CAN tools accordingly and giving the users the ability to continue to perform their tests and measurements in the multibus environments of the modern vehicles using the tools with which they were familiar. The highly accurate capturing, synchronization and processing of the numerous messages and signals from the various parallel bus systems continues today to be one of the greatest challenges for a tool producer.

With CAN FD (CAN Flexible Data Rate), Bosch presented an interesting successor to Classical CAN in 2012. The main new features are a user data length that was expanded from 8 bytes to 64 bytes as well as the ability to switch to significantly higher data transmission rates. While, in the opinion of specialists, CAN will continue to be a factor for at least ten more years, the series introduction of CAN FD is just around the corner and the next generations of vehicles from various manufacturers will be equipped with it. One big advantage of CAN FD is that the existing CAN concepts such as bus arbitration, message detection, event control, etc., are retained and the developers do not need to deal with a completely different behavior. The same holds for the integration of CAN FD in the tools.

A switch to automotive Ethernet, on the other hand, involves significant changes in the familiar ways of working and thinking. The high bandwidth of Ethernet is essential for intelligent ADAS applications (Advanced Driver Assistance Systems) as well as future autonomous driving. The systems use so-called environmental models, which continuously are to be fed with large quantities of data by onboard cameras, radar sensors, etc. Instead of using a physically connected common bus system, with Ethernet one uses a distributed logical network made up of numerous switches and electrical point-to-point connections. This necessitates entirely new approaches both by network and ECU developers as well as by vehicle manufacturers. On top of this comes the handling of the significantly greater volume of data. While the previous automotive networks are defined largely statically, in future applications – especially in combination with Ethernet – service-oriented communication will play an important role. This results in

Related articles

- Marc Lobmeyer, Roman Marktl (Vector): Tips and tricks for the use of CAPL (part 1)
- Marc Lobmeyer, Roman Marktl (Vector): Tips and tricks for the use of CAPL (part 2)
- Marc Lobmeyer, Roman Marktl (Vector): Tips and tricks for the use of CAPL (part 3)
- Mirko Donatzer (Vector): Eye diagram analysis for CAN FD
greater flexibility and reusability of function units; new platforms can thereby be quickly assembled using the modular concept. On startup, the ECUs then communicate which network nodes are interested in which information and who can supply the information.

What history teaches us

In spite of modern computers with multi-core CPUs and computing power and memory sizes that are about 1 000 times greater, today’s challenges differ very little from those in the early stages of CAN. The fight for performance remains and the available resources are always pushed to the limits. Today’s tools (Figure 3) must be able to handle extreme real-time requirements with reaction times in the sub-microsecond range. The required data transmission rates are about 100 Mbyte/s, i.e., approx. 1 000 Mbit/s. Here as well, we again encounter the factor of 1 000, if one thinks back to the 1 Mbit/s of the first CAN tools. Compared to the single diskette that held the first CANalyzer, a current full-installation of CANoe/CANalyzer with all options has a data volume in excess of 3 GiB, which would be equivalent to approximately 2 000 of the above mentioned installation diskettes.

Vector congratulates CiA on its 25th anniversary and is thankful for the excellent collaboration over the many years. We can only speculate on what communication systems will bring in the next 25 years and what the data networks will look like. CAN and, above all, CAN FD have the chance to continue to play an important role in the coming decades.

About the author

With respect to the CAN network, Martin Litschel can certainly be considered a pioneer. In the 1980s, he played a key role at Robert Bosch in defining the CAN protocol and was also involved in the first semiconductor implementation; he then led the development group beginning in 1987. Just one year later, he left Bosch and, together with two friends, founded Vector Informatik. Until 2014 he was responsible there for hardware and network tool development as executive director. Today he is involved with Vector as authorized officer and is active on the Vector foundation as foundation trustee.
Hardware & Software for CAN Applications

**PCAN-Explorer 6**
Professional CAN Communication Software

The PCAN-Explorer is a versatile, professional Windows® software for the observation, control, and simulation of CAN busses. Besides CAN 2.0 A/B, the PCAN-Explorer 6 now also supports the CAN FD standard.

- Simultaneous connections with multiple CAN interfaces independent of their hardware type
- Support for the CAN specifications 2.0 A/B and FD
- Transmission of CAN FD data with bit rates up to 12 Mbit/s
- Clear display of the CAN traffic with various information
- Configurable symbolic message representation
- Easy manual and periodic message transmission
- Playback of trace files with optional loop function
- Multiple flexible filters, also usable for recorded trace files
- Management of settings, information, and files in projects
- Export of entire projects including linked files to handy packets for archiving or sharing
- Automation of small tasks or complex processes with macros or VBScript
- Functionality upgrades with add-ins

**Now:** Basic training for groups or individuals. If you are interested, please contact training@peak-system.com.

**PCAN-PCI Express FD**
CAN FD Interface for PCI Express

The CAN FD interface PCAN-PCI Express FD is a plug-in card (PCIe-x1) for the PCIe slot with data transfer rates up to 12 Mbit/s. Delivery includes a monitor software, APIs, and drivers for Windows® and Linux.

- Available with 4, 2, or 1 High-speed CAN channels (4 channel version expected to be available in Q1 2017)
- Complies with CAN specifications 2.0 A/B and FD
- CAN FD support for ISO and Non-ISO standards switchable
- CAN FD bit rates for the data field up to 12 Mbit/s
- CAN bit rates from 25 kbit/s up to 1 Mbit/s
- CAN bus connection via D-Sub, 9-pin (in accordance with CIA-303-I)
- Galvanic isolation on the CAN connection up to 500 V, separate for each CAN channel
- CAN termination and 5-Volt supply at the CAN connection can be activated, separately for each CAN channel
- PCIe data transfer via bus master DMA. DMA memory access operations with 32- and 64-bit addresses
- Extended operating temperature range from -40 to 85 °C
- Measurement of bus load including error frames and overload frames on the physical bus
- Induced error generation for incoming and outgoing CAN messages

PEAK-System Technik GmbH
Otto-Roethl-Str. 69, 64293 Darmstadt, Germany
Phone: +49 6151 8173-20 - Fax: +49 6151 8173-29
E-mail: info@peak-system.com
www.peak-system.com

Meet us in hall 1, booth 483