CAN Goes America

Chris M. LeBlanc
National Instruments
11500 N. Mopac Expwy, Austin, Texas 78759
Tel: 512-683-6843
Fax: 512-683-5569
chris.leblanc@ni.com

Abstract
This paper touches on the current success of CAN, including its success in expanding into the American CAN market. Additionally, this paper takes a look at CAN as a technology component and illustrates the need for a well-designed test strategy that brings products to market in an efficient and timely manner. We talk about the issues facing test engineers and what to look for while solving applications that involve CAN as a component.

Expanding CAN Market

When you look at products and/or technologies that are successful, you can almost always see at least one common characteristic or trait. The popularity of the product or technology reaches worldwide acceptance. A few examples of successful products are the automobile, telephone, electronic games, and the personal digital assistant (PDA). These products are a far cry from CAN bus, but the principles are the same -you must leverage your core strengths in as many industries, within as many different markets as possible. However the penetration of a product/technology must begin somewhere, so that concentrated efforts are made to ensure success. In other words, a beachhead must be established. For CAN bus the beachhead was the automotive industry within the European market. So the question is, "has this beachhead expanded to other markets, such as the American and Japanese markets?"

The adoption of CAN within the European automotive industry has helped spread CAN around the world. Likewise, there is certainly a visible trend in the U.S. automotive market toward the adoption of CAN. According to the CAN in Automation group, America has captured 10 percent of the CAN market in 2001 with Asia and Europe coming in with 15 and 75 percent respectfully. This compares to 1999 where Asia was 7 percent; America was 8 percent, and Europe was 85 percent of the CAN market. According to this same source, the compounded annual growth rate, from 1999 through 2003, for CAN is 44.2 percent. Industry experts also believe that during the next few years, U.S. and Japanese automotive makers should release significantly more vehicles that use CAN.

We all know of CAN's success within the automobile industry, but certainly there are many other industries that have benefited from CAN. As a technology, CAN has two main industries that it serves, the automotive and automation industries. Industrial and building automation, machine control, medical equipment, maritime electronics, mass transportation (bus and rail), farm machinery, off-road machinery, and robot controls are examples of industries that are looking at CAN to solve distributed, real-time, deterministic communication needs. For anyone who has read the Eighth International CAN Conference literature, it also looks like slot machine gamblers and Monorail customers in Las Vegas benefit from CAN. As you can see, CAN is used in a wide variety of applications, and industries and as its popularity grows, we will see additional industries adopt CAN for similar needs.

Next to automotive, the industrial automation industry is the second most successful industry for CAN. Specifically, higher layer protocols (HLP), such as DeviceNet and CANopen, are built on top of the CAN bus. There are other HLPs that use CAN, but DeviceNet and CANopen are the two most popular. An HLP is a software layer on top of CAN that adds additional features and functionality for interoperability, connectivity, and diagnostics. These features are more useful in industrial automation, where engineers are developing machine control applications using products from multiple vendors. As far as market share, DeviceNet is dominating the American market. DeviceNet is also making inroads into Asia and is showing up more and more in Europe. According to the Venture Development Corporations (VDC) 2001 study, titled "Global Markets and User Needs for Industrial Distributed/Remote I/O, Second Edition," the DeviceNet distributed I/O market is $170 M in 2000 and growing to $250 M in 2005. This only includes distributed I/O products, which is only a small portion of the total DeviceNet market. Much of the DeviceNet and CANopen market consists of specific intelligent devices such as photoeyes, mass flow controllers, and motion controllers. CANopen, on the other hand, is extremely popular in Europe but seems to be localized to that market. The same VDC study estimates the CANopen distributed I/O market to be about $4.5 M; however, this number seems low based on personal customer interaction. Again, distributed I/O products will only be a small fraction of the total CANopen market. Another way of looking at this is to say CANopen is as popular in Europe as DeviceNet is in America.

It is interesting to see CAN grow in popularity and use. It is clear that CAN has spread to additional markets and industries outside its original beachhead. In the case of automotive, this is a good thing for consumers, as they will benefit from vendors offering more efficient vehicles with less weight and more functionality and comfort. This is, of course, dependant on the vendor's ability to bring new product to market in a timely manner. The remainder of this paper will examine the time-to-market issues that face the CAN market and specifically address a couple of technical issues facing engineers.

Development Cycle of a New Technology

As with any new technology, CAN has gone through an extensive and thorough development cycle. In looking back, we hope to gain an insight into what was important during the different stages of development and foreshadow the important issues of the future.

When CAN development began in 1983 by the Robert Bosch Company, there were no suitable serial communication methods available that could handle the high-stressed electromagnetic noise environment of automobiles. Real-time capabilities were also needed to ensure that the engine control, automatic transmission control, and anti-lock brake systems could communicate with each other in a safe and reliable manner. However, at the time, there were no off-the-shelf tools or components readily available. Developers had to build custom hardware and embedded software (firmware). CAN applications at this point in time consisted of lab experiments and simulation to test development efforts. Semiconductor companies' development and offering of CAN controllers and transceivers did not begin until the specification was released in 1985.

With CAN controllers and transceivers readily available, the process of developing CAN based devices began. At this point, automotive makers began to plan the introduction of CAN-based controllers integrated into high-end models. The first automobile with CAN did not become available until 1991. The Mercedes S-class linked five electronic control units together with CAN. At this point in the development cycle, one-off prototypes and beta units were under development, and testing tools for these units were also in the development stage. There were tools on the market to help solve such applications, but many of these tools were carried over from the early development days. Sometimes these tools were adequate, while at other times, the application simply required a more sophisticated approach. In these cases, valuable resources were consumed for developing the one-off test tools necessary to bring the final end product to market.

By 1994, CAN development was in full gear and many different European automotive companies either already had vehicles on the road with CAN networks or were designing CAN networks into next generation vehicles. These plans involved more than just the high-end models but also the higher-volume middle and low-end models. With such a wide movement toward vehicles that use CAN, bringing these models to market and making sure the consumer was happy with the end product became very important. Manufactures began to demand testing tools that could ensure efficient production and quality products.

Time-to-market and quality are issues the automotive industry faces annually. Consumers expect a new model, not necessarily redesigned but with at least incremental improvements, each year, and the quality must always improve. This puts tremendous pressure on the automotive industry to deliver on these expectations. Therefore, time-to-market is key for any successful automotive company, and the introduction of new technology must encompass test strategies that are effective and easy to implement.

These new time-to-market test strategies must answer the question, "How does the new CAN system interact and respond to the real world as part of the end product?" The challenge with CAN is how do you test CAN devices, or networks of CAN devices, within the real world environment of a vehicle? The follow on question is how do you do this testing, which ensures quality, while still meeting the time-to-market needs of the market? This is also necessary in order to drive the cost of test down, so that all makes and models can be outfitted with CAN and not just the high-end models.

Issues With Bringing New CAN-based Products to Market

From a marketing perspective, time-to-market is everything, and any unknown variable slows the development cycle. In the case of end products that use CAN as a component, it must be verified that the new components operate as good, if not better, than the legacy solution.

As engineers develop new CAN devices to replace traditional sensor-based control systems, it becomes necessary to make sure that new devices function properly. One method for validating CAN devices is to measure the same physical signal in an alternate way. For example, a typical method is to use a traditional sensor, signal conditioning, and data acquisition (DAQ) interface. A typical PC-based validation system consists of a CAN interface connected to the CAN network and a DAQ interface connected to the signal conditioning system. Figure 1 illustrates such a system. For validating the CAN node, data from the CAN and DAQ systems must be synchronized and correlated by matching the timestamps of the incoming data. You can perform this process manually or as a post-processing step, which may require considerable software integration effort. This type of software synchronization may not be successful for tests that run over a relatively longer period of time due to clock or oscillator drift. Each interface acquires and timestamps data using its internal clock, which initializes at

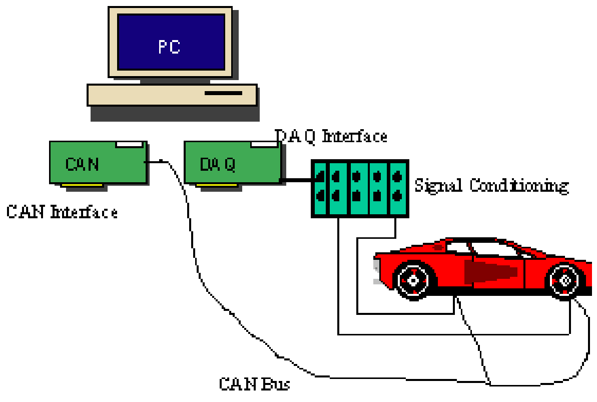

Figure 1 – PC-based test station used to measure the same real world signal using two different simultaneous methods. The first method is to se a CAN interface, and the second method uses a DAQ interface to verify that the CAN data

the beginning of the acquisition. Over a period of time, depending on the oscillator tolerance, the time on each interface may drift with respect to the other and the host PC time. This results in different timestamps for data acquired on different boards at the same instant, making software synchronization and data correlation a difficult task. A simple way to correct for clock drift is to provide a common, hardware-based clock source to the multiple hardware (CAN and DAQ) interfaces.

As stated, the straightforward way to resolve correlation issues is by providing a common timing signal between multiple interfaces. The obvious choice is to use a hardware synchronization signal line for this purpose. All interfaces in the system should have a common capability to connect this timing signal both physically (by external cabling or connections on a back-plane) and logically (in software). This signal can serve the purpose of a start/stop trigger, sampling clock, or an event that can be "mastered" by one interface and "slaved" by another in the system.

As illustrated in Figure 1, engineers can use the DAQ interface to acquire the value of traditional sensors mounted directly on the CAN device. This is done to give a real world equivalent of what the CAN device should report. The DAQ interface is responsible for performing the analog to digital conversion of traditional sensors and queues the data to either the on-board memory of the interface or PC memory. On the other hand, CAN devices publish their data on the CAN network asynchronously. This data message (or frame) is received by a CAN interface and is queued to the on-board memory of the interface or PC memory.

Specifically, if the DAQ interface masters the synchronization signal, then a provision should be made in the CAN interface to accept this signal, timestamp it, and transfer it to memory. It should be noted that while the synchronization signals are being received, the CAN controller is also time stamping the received CAN data messages and queuing them to memory. At the same time, the DAQ interface should do the same. Thus, a CAN interface memory snapshot is a time-sorted list of CAN messages and synchronization signals. Likewise, the DAQ interface memory snapshot contains the data from the sampled analog channels. The correlation process would involve a simple procedure that steps through the CAN list and picks the CAN data closest (before or after) the synchronization signal. This correlation step is necessary to validate the data from the CAN device and is illustrated in figure 2.

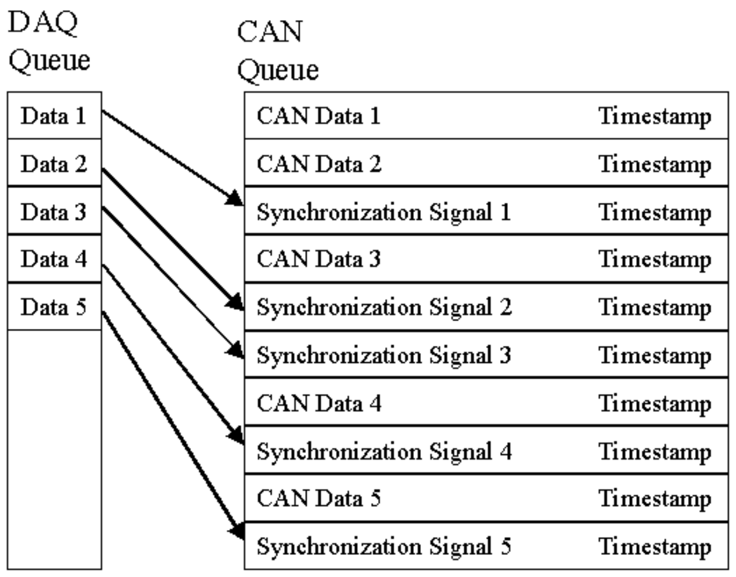

Figure 2: Memory snapshot of CAN and DAQ queue used for correlation of data.

On the other hand, if the CAN interface masters the synchronization signal, it should assert the synchronization signal on receiving a CAN message. The DAQ interface accepts this signal and uses it as a trigger to perform an acquisition (See Figure 3). In this case, no extra step is needed, as the DAQ acquisition is done at the same time the CAN interface receives the CAN message. A definite delay (latency) would occur in the propagation of the synchronization signal to the DAQ interface plus the time it takes to perform the analog to digital conversion. However, this time would be negligible and relatively easy to account for, because it would be measurable and deterministic.

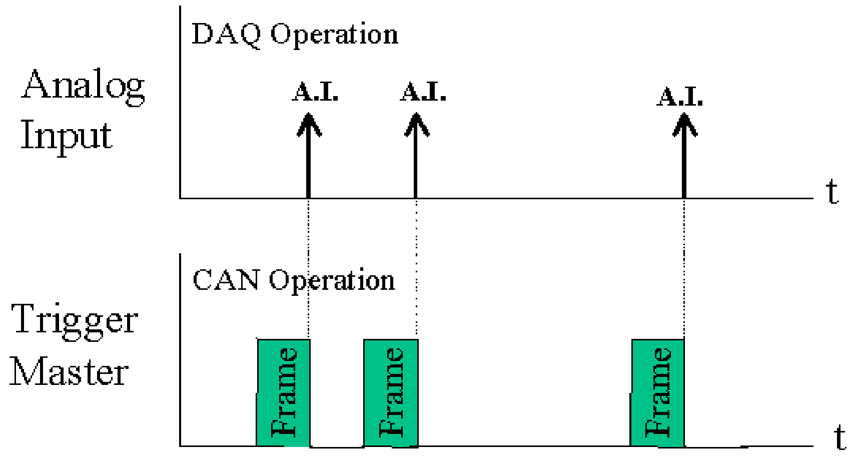

Figure 3: Timing diagram for a CAN mastering trigger and timing signal. Because the CAN data is asynchronous, DAQ analog input operations must wait until new CAN data is received.

Device and Network Simulation and Control

In addition to the already mentioned triggering and timing challenges, the CAN device(s) may not be physically available for actual system (network) testing. The reason is devices may not have been developed yet or the manufacturing of the device has a long lead-time. In this case, the device (or a network of devices) may need to be simulated, either using theoretical data or by replaying time-stamped data recorded during a test. A major challenge, on a PC-based simulation system, is to make sure that the data is transmitted in the sequence it was recorded or the device responds appropriately, with minimum jitter (typically less than 100 micro seconds).

For example, when an electronic control unit (ECU) is simulated in software, it may require some computation on incoming CAN data messages and respond with a control message, such as a PID control loop, to another CAN device on the network. For an effective control algorithm, it is crucial to compute and transmit the control signal with as little jitter as possible. On a PC-based system running a non real-time operating system, the biggest challenge is to overcome and control the jitter, caused by the operating system performing other non-critical task. To overcome this problem, the simulation or control application must run on a real-time operating system, which should give predictable jitter, highly deterministic closed-loop performance, and increased reliability.

Often, engineers use custom hardware along with a real-time operating system to port and run the simulation code on an intelligent interface that uses a micro controller or an embedded processor. This type of a platform poses obvious limitations on program size, cost, flexibility, and portability. The ideal solution is a compact hardware platform that is based on an industry standard, for example PXI (PCI eXtensions for Instrumentation) that can host a real-time operating system and is flexible enough to accept multiple interfaces. In addition, the programs developed for this target platform should also run on other platforms (for example, a desktop PC or notebook computer) or operating systems (for example, Windows) without modification for flexibility in both development and testing.

It is worth noting that in addition to the real-time requirements of simulation and control, some validation applications may also require real-time performance. In this situation, the system should support the timing and triggering synchronization capabilities discussed earlier with real-time simulation and control capabilities. Meaning the real-time platform of choice should leverage the capability to physically and logically connect timing and triggering signals between multiple interfaces.

Conclusion

Has CAN reached America? Of course it has, and it will continue to grow and solve the communication needs of not just the automotive industry but also the automation and machine control industries. The challenge for the future of CAN is not the development of CAN solutions and tools but rather bringing these end products to market. The best-designed product in the world will not be successful until the market has a chance to purchase and make use of that product. Therefore, the challenge is to develop a test strategy that helps companies get their products to market in the least amount of time with the highest quality possible. This means your test strategy must be flexible enough to handle the mixed-signal needs of a complete test system, including CAN support.

References

Z. Chasmawala and C. LeBlanc: “A Platform for Real-Time, Synchronized In-Vehicle CAN Measurements”, COTS Journal, October 2001.
U. Kiencke and T. Kytola: “CAN, a Ten Years’ Anniversarial Review”, 3rd International CAN Conference, October 1996.
CAN Newsletter (CiA), “CAN sales figures”, p6, Q4 2001