Eye diagrams for CAN

(Source: Adobe Stock)

Eye diagrams enable quick evaluating of the signal integrity of serial data systems. This article explains how to generate eye diagrams for CAN, including separate arbitration and data phase diagrams for CAN FD and CAN XL. It also describes how to use information gained from eye diagrams to evaluate CAN networks.

The eye diagram is a simple method for evaluating the signal quality of serial data. It is a standard analysis tool for high-speed data signals such as PCI Express, SAS (serial attached SCSI: small computer system interface), SATA (serial ATA: AT attachment), etc. It is much less commonly used for serial networks such as CAN, but it provides a simple way to characterize the signal quality.

Unlike high-speed network systems, on the CAN network data is transmitted in short bursts with blocks of bits. Acquiring multiple waveforms and "stacking" them on a persistence display is how the traditional eye diagram is formed, showing the history of all the acquisitions. Although this method is possible, it is not useful for CAN, because, in this mode, there is no way to distinguish whether the data comes from a single frame or even from the same device on the network.

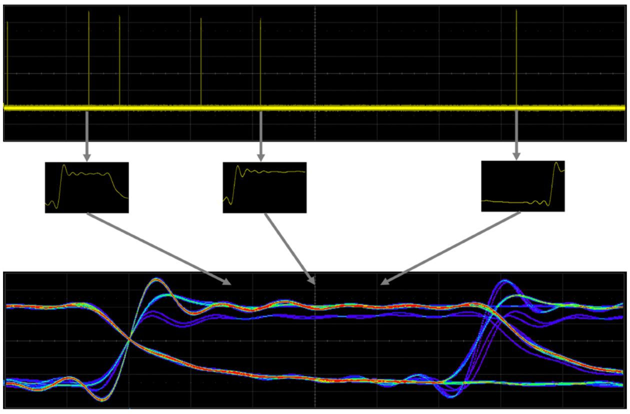


Figure 1: For the classic eye diagram, the oscilloscope is triggered on any edge while running in persistence mode. For CAN, this kind of eye diagram will contain bits from different data frames and different nodes. (Source: Teledyne Lecroy)

A better way to form the eye diagram is to acquire a record containing a complete data frame and to overlay all bits from this one data frame to form the eye diagram. This can be done with the use of a special "slicer" software function in the oscilloscope, which breaks down the waveforms into slices. These "slices" are overlaid on top of each other to generate the eye diagram of the data frame. To cut the slices into the correct length and adjust them on top of each other, a "virtual" clock must be used. In the case of CAN, this clock needs to be extracted from the waveform itself because it is not available as a separate signal.

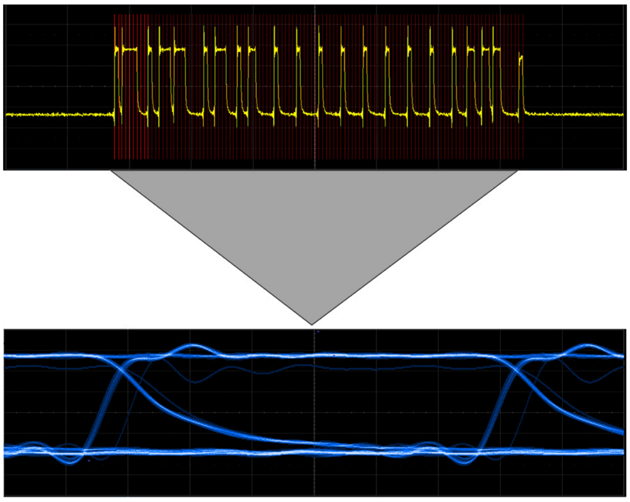


Figure 2: For a "sliced" eye diagram, all bits from a data frame are overlaid using a virtual clock. In the top graph, the red bars show the separation of the bits extracted by the virtual clock. (Source: Teledyne Lecroy)

This method works quite well for CAN CC (classic) with a constant bit rate in a frame, but what happens if this method is used for CAN FD (flexible data rate) or CAN XL (extended data-field length) where the bit rate changes?

To get a usable eye diagram for CAN FD and CAN XL, where different parts in the frame use different clock speeds, it is needed to refine the generation of the eye diagram by splitting the CAN frame into the two parts, the arbitration phase and the data phase. The easiest way, and this is also the way most oscilloscope vendors have implemented the eye diagram for CAN, is to create the eye diagram only from the data phase of the CAN frame.

A much better way is to generate separate eye diagrams for both the arbitration phase and data phase ▷

of each frame. To be able to separate the phases and overlay the correct bits for the different eye diagrams, the oscilloscope must decode and interpret the CAN frame first. For example, in the Figure 3, only the bits highlighted in blue are relevant to the arbitration-phase eye diagram. To get a correct eye diagram containing all parts of the arbitration phase, bits that occur both before and after the data phase have to be overlayed. Similarly, the second eye diagram for the data phase uses only the part of the frame that is highlighted in green in Figure 3.

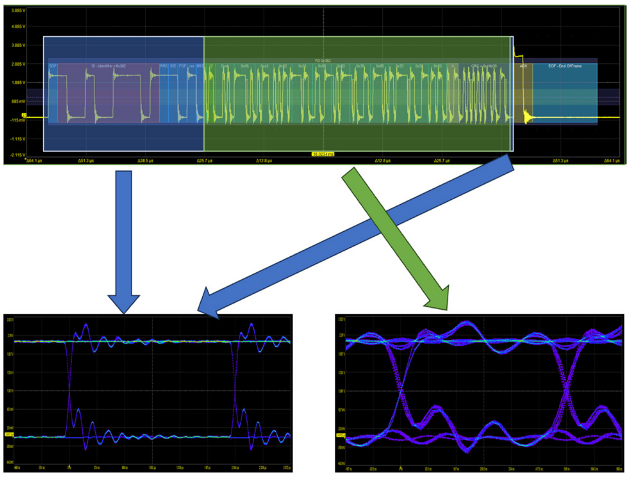


Figure 3: To get a useful eye diagram for a CAN FD or CAN XL network, two kinds of eye diagrams have to be generated, one for the arbitration phase (blue) and a second one for the data phase (green) (Source: Teledyne Lecroy)

Both eye diagrams also need a different virtual clock because of different data rates in both phases. The eye diagrams do not include the acknowledge bit, since it does not belong to the data frame and other nodes send it. For generating a CAN XL eye diagram, this method has to be further extended. In CAN XL there is a third class of bits, the ADS (arbitration to data sequence) and DAS (data to arbitration sequence) bit sequences, which are needed by the protocol to give the receiver additional time to synchronize to the higher bit rate (ADS) and before returning to the arbitration phase (DAS). Because a real receiver does not care about the signal shape during this “transition” time, these bits have to be ignored when generating the eye diagrams, because including them would most likely give a false fail indicator when doing the mask testing.

Calculation of the virtual clock

To get a better understanding of what is needed for the virtual clock, one needs to look at how synchronization works in CAN. The bit time is divided into four segments: the synchronization segment, the propagation time segment, the phase buffer segment 1, and the phase buffer segment 2. Each segment consists of a specific, programmable number of time quanta. The length of the time quantum (t_q), which is the basic time unit of the bit time, is defined by the CAN controller’s system clock and the bit-rate prescaler (BRP). More information can be found in the CAN standard (ISO 11898-1).

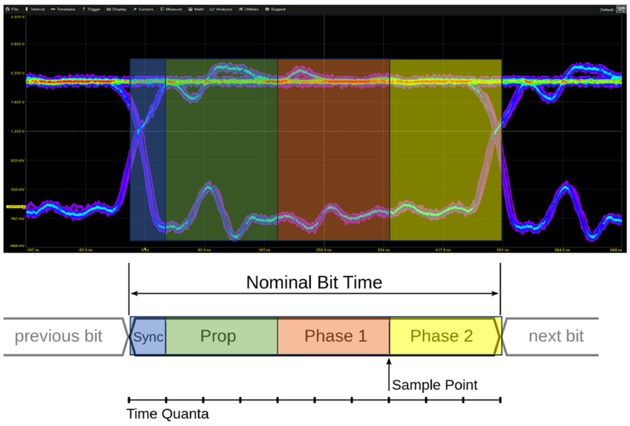


Figure 4: The bit time is divided into four segments: the synchronization segment, the propagation time segment, the phase buffer segment 1, and the phase buffer segment 2. Each segment consists of a specific, programmable number of time quanta. (Source: Teledyne Lecroy)

There are two types of synchronization in CAN: the hard synchronization and the resynchronization. Hard synchronization occurs at every start-of-frame (SOF) when a recessive to dominant bit edge is detected after the intermission or idle period. Resynchronization occurs throughout the frame whenever a recessive to dominant bit edge is detected outside of the synchronization segment. If an edge is too late, the phase buffer segment 1 is lengthened to compensate and to adjust the time between the edge and the sampling point. If it is too early, the phase buffer segment 2 is shortened to adjust the time between the edge and the sampling point. To simulate this behavior in the eye diagram on the oscilloscope, the resynchronization of the virtual clock is done also on a recessive-to-dominant edge. In a real receiver, this setting can only be made with the resolution of a time quantum, which differs from how the setting is made with an oscilloscope, which can set the edge with a much higher resolution. This difference in resolution must be taken into account to overlay a mask on the oscilloscope eye diagram.

Generation of eye diagrams

For the data phase, the timing for the eye diagram has to be adjusted in a way similar to how it is done in a real receiver during resynchronization. This means that the software clock recovery has to be resynchronized on every recessive-to-dominant edge.

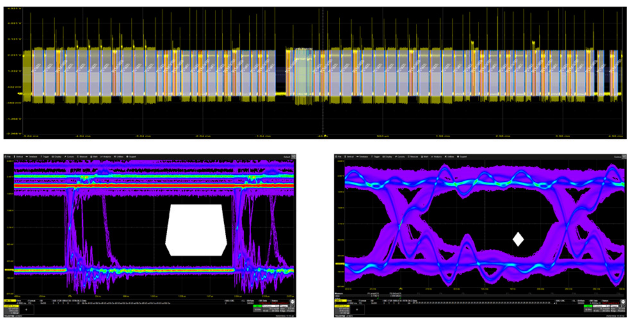


Figure 5: Two eye diagrams should be generated for CAN frames: one for the arbitration phase (lower left) and one for the data phase (lower right) (Source: Teledyne Lecroy)

The lower right of Figure 5 shows an eye diagram of the data phase. In this phase, all bits are aligned like a real receiver that uses resynchronization on each recessive-to-dominant data edge. To overlay all data bits, the nominal bit rate for all edges between the resynchronizations is used.

To align the eye diagram of the arbitration phase (lower left of Figure 5), it is necessary to overlay acquisitions in such a way that the transitions do not hit the mask. The solution to the problem of applying a mask is to use the first recessive-to-dominant edge (the leading edge of the start-of-frame) as the reference. According to the definition in ISO 11898, all other arbitration transitions must fall (occur) in front of the mask area, so that a violation of this is a real reception problem.

In many oscilloscopes (including the Teledyne Lecroy oscilloscopes), masks for CAN are available where, as is common in other standards, the mask is centered on the middle of the bit. In real CAN systems, this type of mask is not useful, as due to the expected ringing, the sampling point is not set in the middle of the bit length. For testing a real system, the mask needs to reflect this behavior. Figure 5 shows examples of masks that are better suited to the requirements of a real CAN network. The possibility of limiting an eye diagram to a participant or a specific frame via a filter function enables a more in-depth analysis and simplifies troubleshooting.

Analyzing the CAN network

Knowing the sampling point allows to check if there is a stable condition at the time of the sampling point. The timing of the ringing can also be quantified by simply measuring the ringing in the eye diagram. Figure 6 shows a very typical reflection, but it is unclear where in the network this refection occurs.

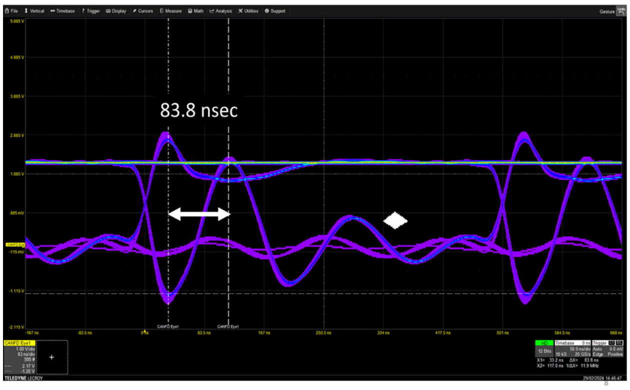
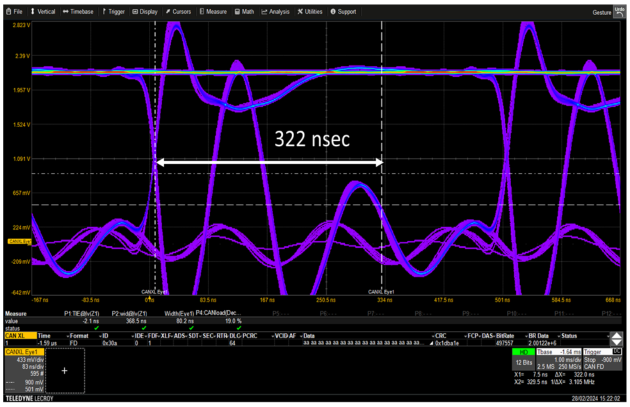


Figure 6: The timing of a reflection can be measured by using oscilloscope cursors to measure the time on the eye diagram between the first maxima and the first reflection maxima (Source: Teledyne Lecroy)

To get information about the reflection point, the time between the first maxima and the first reflection maxima on the eye diagram can be measured. In Figure 6, the cursor measurement shows a delta of ca. 84 ns. It can be assumed that a typical runtime in a CAN network is 5 ns/m, which corresponds to approximately 16,8 m. This is not the correct distance, because it has to be considered that the signal travels to the reflection point and that it will also travel back the same distance, so that this value has to be divided by 2. Based on the cursor measurement, this reflection happens in a distance of 8,4 m.



Bit rate	Sample Point (in nsec)					
	60%	66%	70%	75%	80%	85%
1.5 Mbit/s	400.0	440.0	466.7	500.0	533.3	566.7
2 Mbit/s	300.0	330.0	350.0	375.0	400.0	425.0
2.5 Mbits/s	240.0	264.0	280.0	300.0	320.0	340.0

Figure 7: The eye diagram can be used to measure the time until the reflection has leveled off below the CAN signal thresholds (Source: Teledyne Lecroy)

However, this example can be used to estimate up to which data rate this reflection can be accepted, or up to which data rate the system will work without errors. Since the time for the reflections is fixed, it can be measured on the eye diagram when the earliest sampling point may be in order to reliably detect the signal. The measurement for the given example shows that the signal reflection is settled after 320 ns. From this data, the maximum data rate can be determined at which the sampling point occurs after the last reflection crosses the detection level. If a sampling point is assumed at 70 %, it can be seen that with 320 ns (for reflection settlement) the chosen data rate is already very close to the maximum possible data rate, because the sampling point will be at 350 ns. For a system with a sampling point at 85 %, the 320 ns settling time would allow a theoretical bit rate of 2,66 Mbit/s.

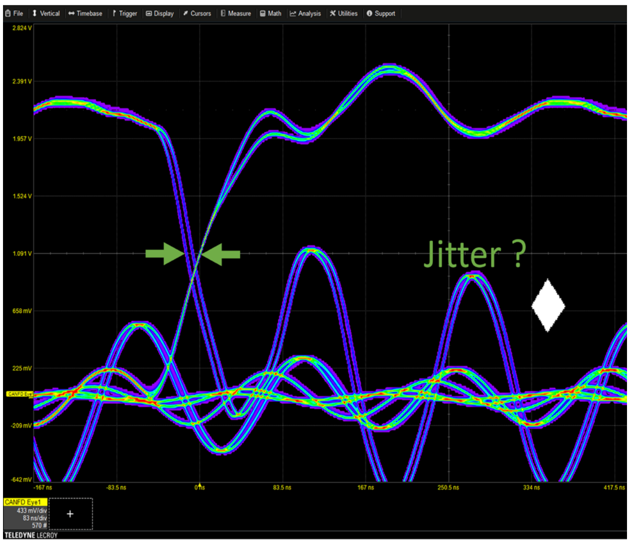


Figure 8: In the eye diagram, it looks as if the signal has different bit lengths, which could be considered as jitter (Source: Teledyne Lecroy)

Other derived signal integrity parameters

Usually, the eye diagram is also used to evaluate noise and jitter in the system. To see how this can be applied to CAN, first let's have a look at the noise. Due to the reflections occurring in the system, an overlay of reflection and noise is usually seen in the eye diagram, but this does not preclude analysis at the sampling point. In the eye diagram of a CAN signal, it is very possible to determine whether the levels at the sampling point are below or above the defined thresholds of 0,4 V for a recessive bit and 0,9 V for a dominant bit.

Automotive cybersecurity: Etas and Keysight cooperate

The two CiA members are partnering to provide automakers and their suppliers with an automotive cybersecurity solution to ensure vehicles are protected when they hit the road. Through this arrangement, the Escrypt Cycurfuzz smart tool from Etas will be integrated into Keysight's automotive cybersecurity test platform. The growing connectivity of road vehicles exposes them to cybersecurity risks that must be mitigated to ensure the development of secure and safe vehicles. One component of an effective, comprehensive security testing is fuzzing, a test automation method that injects invalid, malformed, or unexpected inputs into a device to reveal defects and vulnerabilities.

By embedding the Etas tool into the Keysight test platform, carmakers can fuzz test targets via CAN interfaces. Integrated into the test platform, the cybersecurity tool provides software to do CAN fuzzing at the ECU (electronic control unit) or system level. This offers a way to automatically scan the CAN-connected device under test for unknown vulnerabilities and to uncover software weaknesses. This ensures that the cybersecurity risks associated with the CAN connectivity are effectively mitigated.

The Keysight automotive cybersecurity solution enables automated testing of all OSI (open systems interconnection) layers for in-vehicle interfaces including CAN, automotive Ethernet, Wi-Fi, cellular, and Bluetooth. This can validate the robustness of ECUs and TCUs (telematic control unit) or the entire vehicle. In addition, the solution enables automakers and suppliers to comply with international cybersecurity regulations such as UN R-155, which refers the ISO/SAE 21434 standard.

Eric Cesa, Vice President Sales North America and Brazil, Etas, said: "Our agreement with Keysight is an example of how collaboration enables automakers and their suppliers to constantly improve the software quality and cyber resilience of their products." Thomas Goetzl, Vice President and General Manager of Keysight's Automotive and Energy Solutions, added: "Future-proof testing is vital to mitigating the risks of evolving automotive cyberattacks. Working with Etas to incorporate this fast-fuzzing test into our automotive cyber-security solution will help end-users meet industry standards while growing their threats database knowledge." *hz*

But what about analyzing the jitter in an eye diagram for a CAN system? In Figure 8, a CAN signal can be seen that has different bit lengths, which are normally considered as jitter. But is that really the case? A closer look at the process of resynchronization is needed here. Whenever a node performs a resynchronization and inserts or removes an additional time tick, the length of the bit is changing. In Figure 8, while this looks like jitter in the eye diagram, it is not. So, one has to be careful about trying to identify jitter using the eye diagram of CAN signals. There is another important point that needs to be considered: since there is a resynchronization after five bits at the latest due to the insertion of the stuff bits, the analysis of the jitter in the eye diagram is not very meaningful.

Conclusion

Eye diagrams are very useful for analyzing the signal integrity of a CAN network and provide very helpful information about reflections or violations of the levels at the sampling point. However, it is important to create the eye diagram based on the CAN definitions and to carry out a separate analysis for the arbitration and data phase. ◀

(This article is based on the 18th international CAN Conference (iCC) presentation by Thomas Stueber. The complete paper is published in the 18th iCC proceedings 2024; CiA, Nuremberg.)

Author

Thomas Stueber
Teledyne Lecroy
lecroy.contact.germany@teledyne.com
www.teledyne.com

Engineering