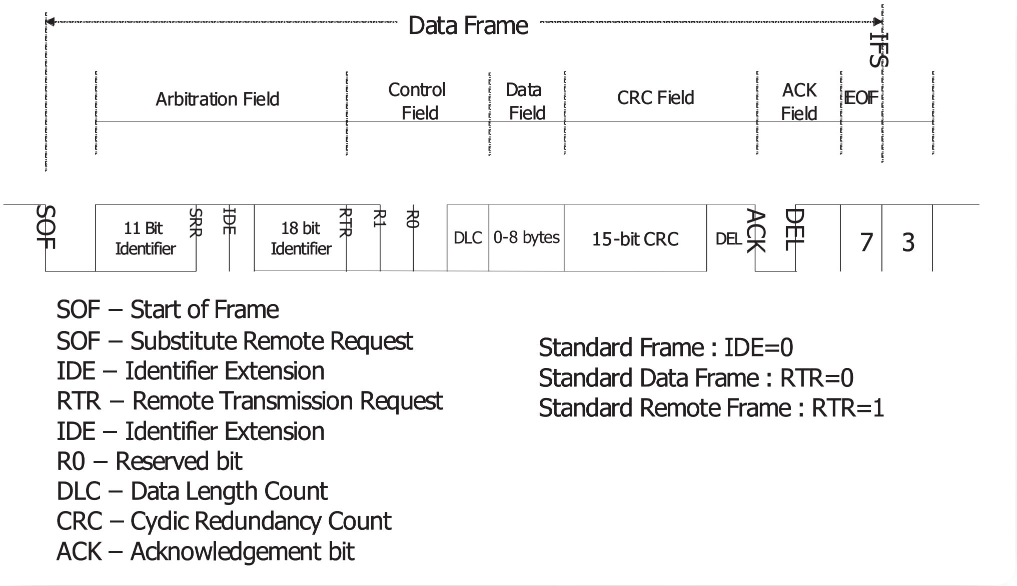
Simplify CAN and LIN In-vehicle Network Testing

An in-vehicle network integrates many modules that interact with the environment, and process high and low speed information. As a result, testing this network has become very challenging. This is mainly because parts of the network (like the nodes) have to process high-speed data to satisfy safety and emission requirements. Other parts of the network are low-speed that they are locally connected to lights and switches.

The automotive industry uses two main network technologies: Controller Area Network (CAN) and Local Interconnect Network (LIN). These technologies coexist in a heterogeneous network making debugging and testing more challenging and time consuming.

Tektronix
Enabling Innovation



▶ **Figure 1.** *Oscillator tolerance*

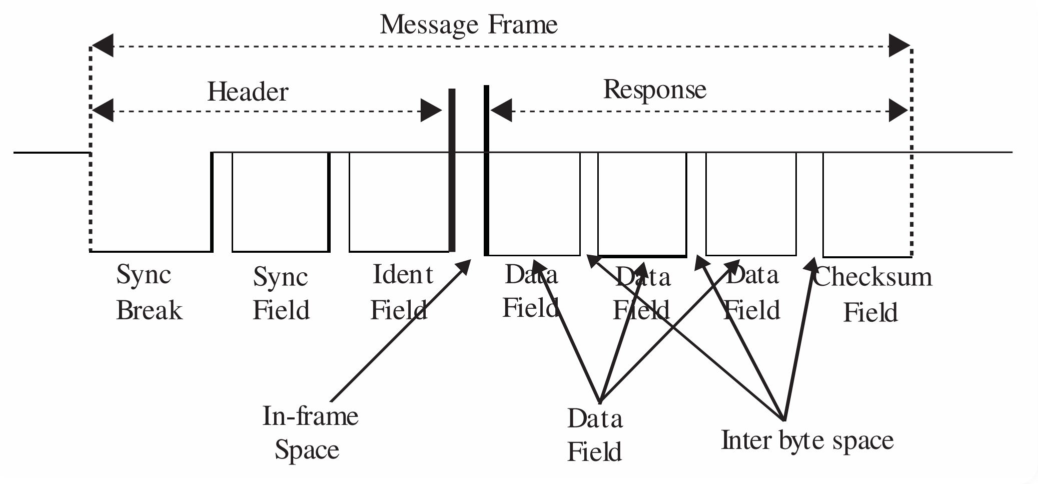
This application note briefly explains the basics of CAN and LIN protocols. It explains how the DPO7000 series and the LSA option, that combines integrated CAN specific trigger capabilities and the TDSVNM CAN and LIN timing and protocol decode software, address the unmet needs in debugging and testing an in-vehicle network. Some of the major features are:

- Measuring Oscillator tolerance and propagation delay saves time
- Measuring Bus Utilization and data rate enables the efficient use of the network
- Analyzing the CAN Eye diagram helps in locating noise problems in CAN message
- Capturing CAN and LIN messages resulting based on the content of the CAN frame and viewing it in protocol decoded format
- Capturing simultaneously the CAN and LIN data and viewing it in protocol decoded format helps in locating latency time at the gateways.
- Advanced triggering both for physical layer debugging and protocol decoding

Basics of CAN

CAN is a serial, asynchronous, multi-master communication protocol for connecting control modules. CAN supports bit rates in the range of 1Kbps to 1Mbps. The data rate less than 125Kbps normally known as low speed CAN. Data rate 125Kbps to 1Mbps is known as high speed CAN. CAN node has its own clock generator for sampling the incoming data. The timing parameter sync segment, propagation segment, phase segment 1 and phase segment 2 of the bit time can be configured individually for each CAN node, creating a common bit rate even though the CAN nodes oscillator clock rate is different.

CAN uses single wire, dual wire or fault tolerant techniques for signaling. In single wire CAN data rates are 33.3Kbps and 83.33Kbps and signaling is single ended. Dual wire CAN data rates are at high speed CAN and signaling is differential. Fault tolerant CAN intended for low speed CAN. If any wire is shorted to battery or ground, fault tolerant CAN still be operational.



▶ **Figure 2.**

Each CAN nodes in the network communicate by sending information in packet form. Each packet will contain overhead and payload. Depending on the type of communication, each packet is either called Data Frame, Remote frame, Overload frame and Error frame.

In in-vehicle network CAN node is requesting for data, then it will be send out a remote frame. If node is busy in processing the data and carrying out priority work, then node will place an overload frame on the bus, indicating currently it is busy. If a CAN node finds there is error in checksum, or data is corrupted, it will send out an error frame. Data frame will have data in the CAN message.

CAN has two protocol formats CAN2.0A and CAN2.0B. The difference between these two versions is number of bits for addressing. CAN2.0A supports 11 bit addressing and tolerant to CAN2.0B. CAN 2.0B support 29 bit addressing also known as extended addressing. CAN2.0B works with CAN2.0A.

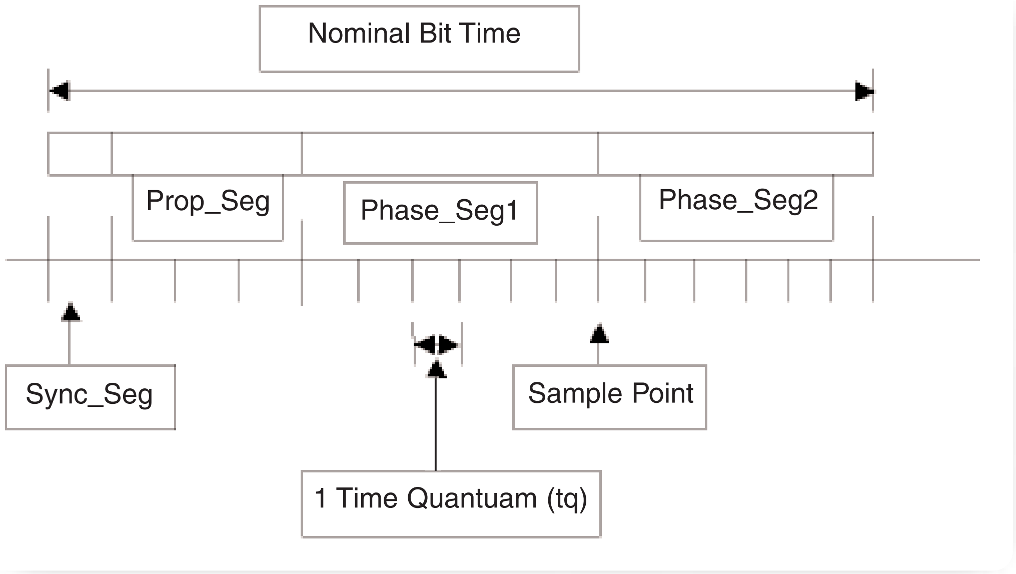
Typical data frame content for CAN2.0B is shown in Figure 1.

Basics of LIN

LIN is a single wire serial communications protocol based on the common UART interface. LIN uses single master, multiple slave networking architecture. Developed for automotive sensor and actuator networking applications for door control, lights and window operation. The LIN master node connects the LIN network with higher level networks, like controller area network (CAN).

LIN supports maximum of 20Kbps data rate resulting from the requirements by electro magnetic interference and clock synchronization.

Slave task is activated upon reception and filtering of the identifier and starts the transmission of the message response. The response comprises of two, four or eight data bytes and one checksum byte. The header and response part form one message format. A LIN message format is as shown in Figure 2.



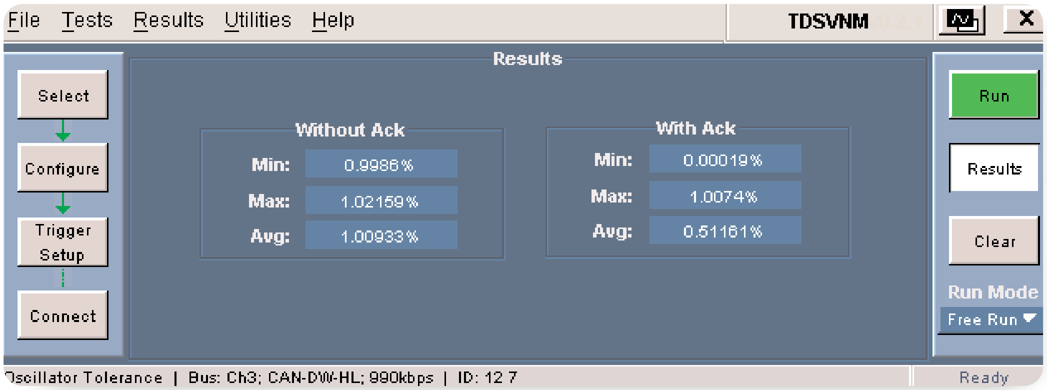
▶ **Figure 3.**

Measuring Oscillator Tolerance and Propagation Delay Helps in Solving Bit Timing Issues

Feature of the CAN protocol is bit rate, bit sample point and number of samples in a bit period are user programmable. During this process the user has to be aware of the relationship between bit timing parameters, the reference oscillator tolerance and the various signal propagation delays in the system.

According to CAN specification, 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 (tq), which is basic time unit of the bit time, is defined by programmers using Bit rate pre scalar. Figure 3 shows how you can select the time quanta in a bit time, then assign time quanta's for the four segments.

The Synchronization Segment (Sync_Seg) is that part of the bit time where edges of the CAN bus level is expected to occur. The distance between edges that occurs outside of sync_seg is called the phase error of that edge. The Propagation Time Segment (Prop_seg) is intended to compensate for the physical delay times within the CAN network. The phase buffer segments Phase_seg1 and Phase_seg2 surround the sample point. The re-synchronization jump width (SJW) defines how far a resynchronization moves the sample point inside the limits defined by the phase buffer segments to compensate for edge to edge errors.



Results

Without Ack		With Ack	
Min:	0.9986%	Min:	0.00019%
Max:	1.02159%	Max:	1.0074%
Avg:	1.00933%	Avg:	0.51161%

Run Mode: Free Run

Oscillator Tolerance | Bus: Ch3; CAN-DW-HL; 990kbps | ID: 12 7 Ready

▶ **Figure 4.** *Oscillator Tolerance*

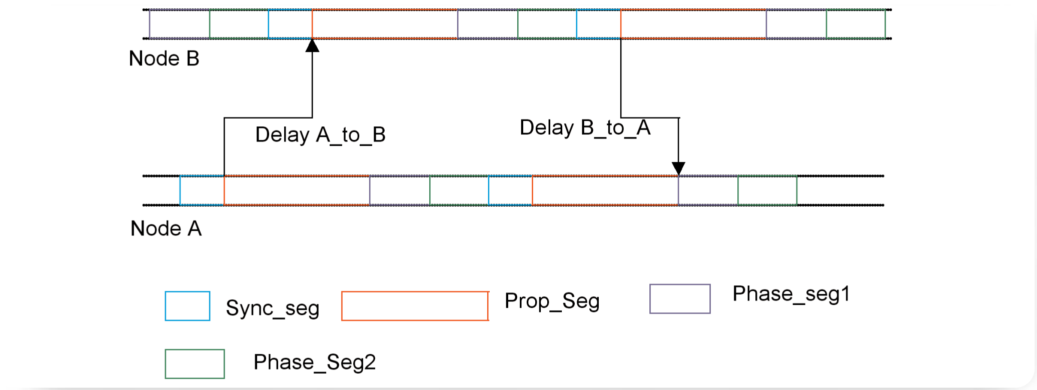
For proper functioning of the CAN network the physical delay times and oscillator tolerance range have to be considered though the bit rate can be achieved by different bit time configurations. Each node in a CAN network derives its bit timing from its own oscillator. This bit time will vary as per the oscillator tolerance. In real systems, the oscillator reference frequency, f_{clk}, will deviate from its nominal value due to initial tolerance offset, aging and ambient temperature variations. The sum of these deviations results in a total oscillator tolerance. Programmers needs factor in this variation while programming the timing registers.

The CAN and LIN timing and protocol decode software (nomenclature TDSVNM) that runs on the DPO7000 series automatically makes oscillator tolerance measurement. User specifies the ID of the CAN node. Result will include the total oscillator tolerance with and without ACK bit. In real world CAN node should accurately sample the ACK bit so that, ACK bit is recognized by the CAN controller. TDSVNM will also provide the data without the ACK bit, which would show the affect of the receiving CAN node on the transmitting node. By combining the oscillator tolerance of transmitting and receiving nodes, CAN controller timing registers can be programmed for error free network performance.

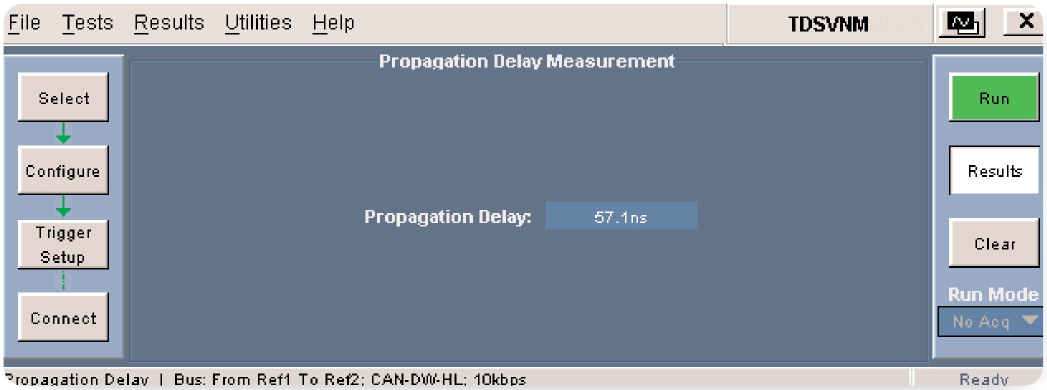
Along with oscillator tolerance information, propagation delay time information is required for accurately setting the sample point. The significance of propagation delay in a CAN system arises from the fact that CAN allows for non-destructive arbitration between nodes contending for access of the network as well as in-frame acknowledgement.

If we assume, there are two CAN nodes A and B. The one way propagation delay between two nodes A and B, defined as t_{prop} (A, B). Any CAN node synchronized to the bit stream on the CAN bus will be out of phase with the transmitter bit stream, caused by the signal propagation time between the two nodes.

In the example in Figure 5, both modes A and B are transmitters performing an arbitration for the CAN bus. The node A has sent its start of frame bit less than one bit time earlier than node, therefore node B has synchronized itself to the received edge from recessive to dominant. Since node B has synchronized this edge delay (A_to_B) after it has been transmitted, B's bit timing segments are shifted with regard to A. Node B



▶ **Figure 5.**



Propagation Delay Measurement

Propagation Delay: 57.1ns

Run Mode: No Acq

Propagation Delay | Bus: From Ref1 To Ref2: CAN-DW-HL: 10kbps

Ready

▶ **Figure 6.** *Propagation Delay*

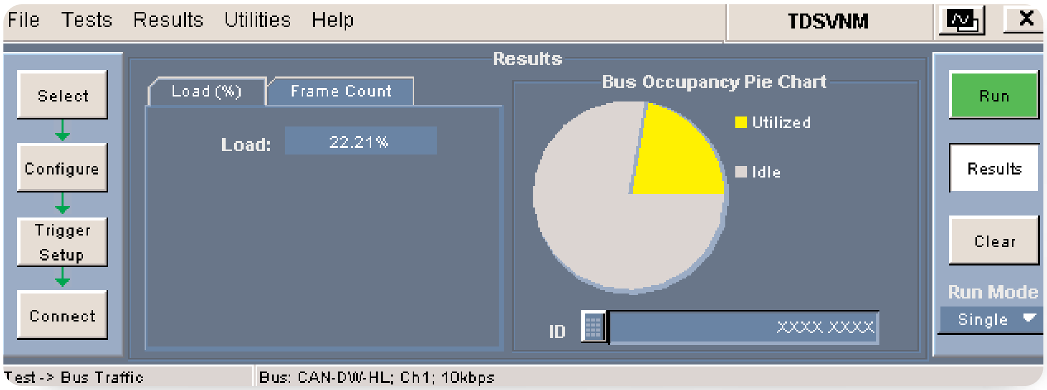
sends an identifier with higher priority and so it will win the arbitration at a specific identifier bit when it transmits a dominant bit while node A transmits a recessive bit. The dominant bit transmitted by node B will arrive at node A after the delay (B_to_A)

Due to oscillator tolerances, the actual position of node A's sample point can be anywhere inside the nominal range of node A's phase bugger segments, so the bit transmitted by node B must arrive at node A before the start of phase_seg1. This condition defines the length of Prop_seg.

If the edge from recessive to dominant transmitted by node B would arrive at node A after the start of Phase_Seg2, it could happen that node A samples a recessive bit instead of a dominant bit, resulting in a bit error and the destruction of the current frame by an error flag.

This problem demands the engineers to consider accurate data while programming the timing registers of the CAN controller. Now, the TDSVNM software provides the automated propagation delay measurement.

Automated oscillator tolerance and propagation delay measurement provided by the TDSVNM software will enable the users to program the register by taking real world data from the in-vehicle network. See Figure 6.



Load: 22.21%

Run Mode: Single

Test -> Bus Traffic

Bus: CAN-DW-HL; Ch1; 10kbps

▶ **Figure 7.** *Bus Utilization Result*

Bus Utilization and Data Rate Measurement Enable the Efficient Use of the In-vehicle Network

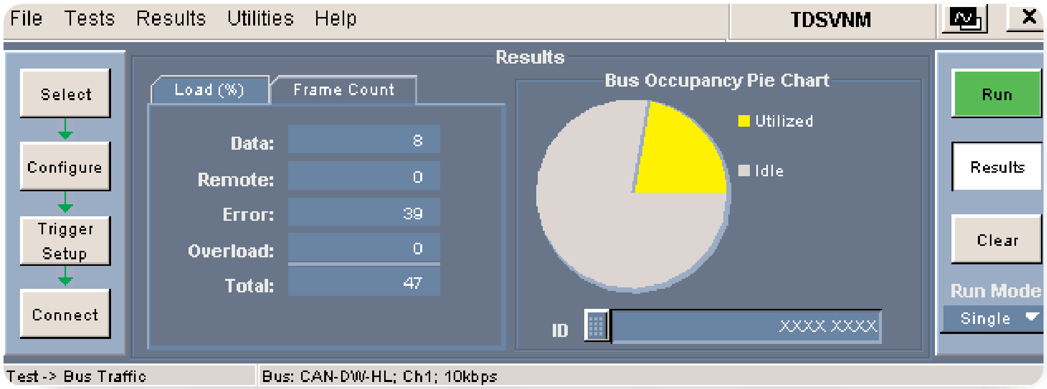
Automotive industry recommends that the in-vehicle network should operate at 30% of the bus load for efficient use. By operating more than 30%, would create nodes in the vehicle to generate overload frame or may not allow low priority activity to take place. Resulting in improper functioning of the in-vehicle network.

The CAN nodes transmit messages when the bus is idle. Because of wired AND logic, when two nodes start transmitting the message simultaneously, the higher priority message overrides the lower priority message. The node with the lowest priority once again tries to access on the bus when the bus is idle. It may be possible that lower priority node will never get an access to the bus for transmission due to the overloading of the bus. Lower priority nodes will have longer latency time than desirable. To analyze the bus utilization and debug problems, it is important to know the condition at which bus is overloaded. This could be any of the following:

1. A message with particular ID,
2. Error frames
3. Overload frames

The TDSVNM software now provides capability to measure bus utilization when particular ID, Data content, error frame, overload frame. Using these capability users can debug the network loading at different operating condition of the vehicle. Figure 7 shows bus utilization result.

To further analyze the bus utilization, user can chose to view the different frame counts in pie chart. Frame count view provides the date frame, error frame and remote and overload frame count. This distribution can point to cause for overloading the CAN bus. Within the data frame, user can also view the data frame counts for a particular ID. This should help to test whether higher priority node is occupying the bus. Using this result users can program the CAN schedule for more acceptable performance and if required reassign the priority of the CAN nodes resulting efficient use of the CAN bus. Figure 8 shows the distribution of frames in CAN traffic.



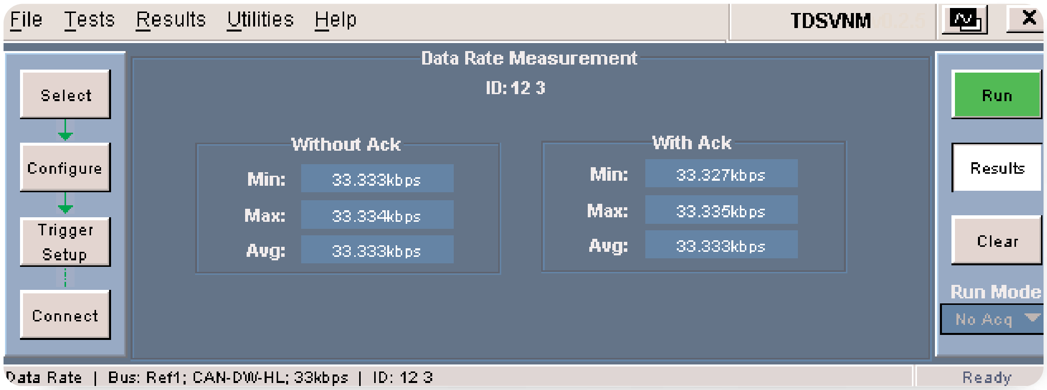
Data: 8
Remote: 0
Error: 39
Overload: 0
Total: 47

ID XXXX XXXX

Run Mode: Single

Test -> Bus Traffic | Bus: CAN-DW-HL; Ch1; 10kbps

▶ **Figure 8.** *Bus Utilization Distribution by Frame Type*



Data Rate Measurement
ID: 12 3

	Without Ack	With Ack
Min:	33.333kbps	33.327kbps
Max:	33.334kbps	33.335kbps
Avg:	33.333kbps	33.333kbps

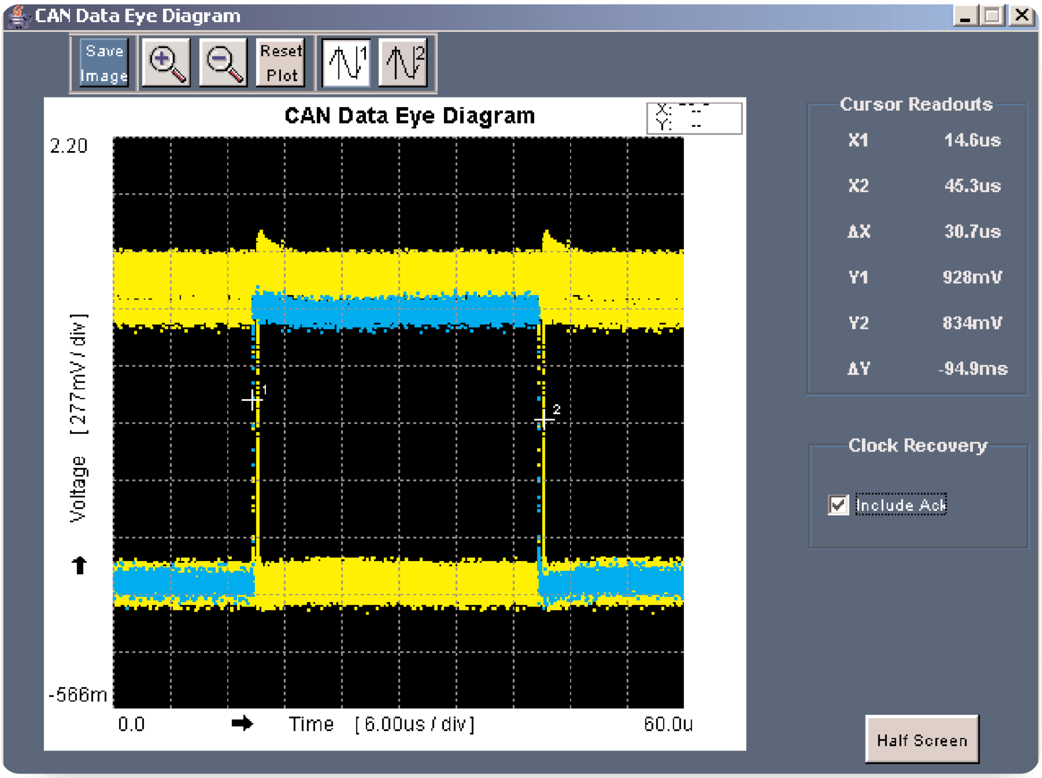
Run Mode: No Acq

Data Rate | Bus: Ref1; CAN-DW-HL; 33kbps | ID: 12 3 | Ready

▶ **Figure 9.** *Data Rate Result*

In-vehicle network has a large number of nodes. As we are aware, each of the CAN nodes can operate at different CAN system clock. However using timing register baud rate pre-scalar all the CAN nodes can be made to operate at one fixed data rate. Due to human errors and change in CAN system clock, the data rate may not match with CAN bus data rate when integrated in the in-vehicle network. When this node transmits or receiving any CAN message, nodes will generate error messages causing inefficiency in the network.

Using the DPO7000 series and the option LSA that combines the TDSVNM CAN and LIN Timing and Protocol Decode software with integrated CAN specific trigger capabilities, users can capture a CAN message for a specific ID and measure the data rate. Using this measurement, users can quickly locate the node whose data rate is not matching with data rate of other CAN nodes. See Figure 9.



Cursor Readouts

X1	14.6us
X2	45.3us
ΔX	30.7us
Y1	928mV
Y2	834mV
ΔY	-94.9ms

Clock Recovery

☑ Include Ack

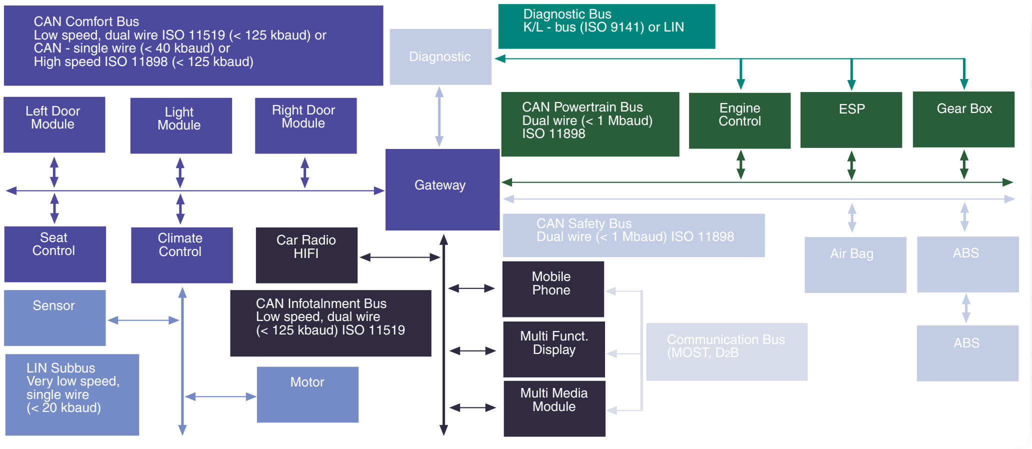
▶ **Figure 10.** *CAN Eye with ACK Bit*

CAN Eye Diagram Analysis Helps in Locating Noise Problems in CAN Message

In-vehicle network operates in electrically noisy environment. This would cause the CAN signals to be vulnerable to electrical noise resulting amplitude aberration, addition of jitter, spikes and glitches in the signal. This distorted signal results in mal-functioning of the in-vehicle network.

Using the TDSVNM software, user can capture desired CAN message by specifying ID as trigger condition. TDSVNM automatically sets up the oscilloscope to trigger and capture the desired CAN messages with very high time and amplitude resolution, so that all the signal information is captured. The Eye diagram plot will quickly show the change in amplitude variation and jitter in the signal.

Figure 10 shows the CAN Eye diagram. Cursors would enable the amplitude and jitter measurement. Eye diagram with and without allows isolating the problem in transmitting or receiving node. Blue trace represents the ack bit in CAN eye diagram.

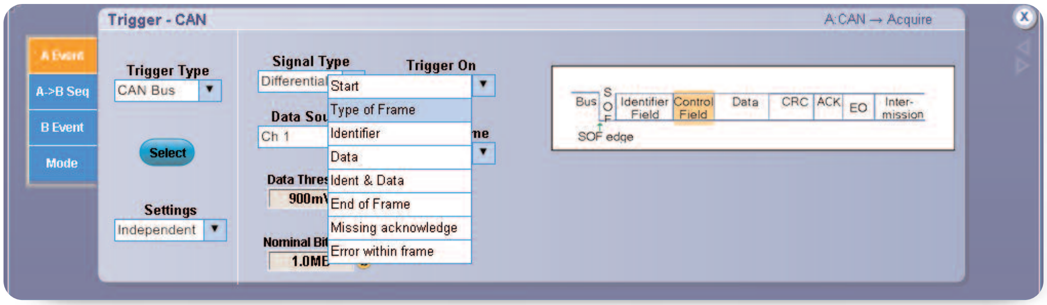


▶ **Figure 11.** *Typical In-vehicle Network*

Protocol Decoding of CAN and LIN Messages

Once the physical layer is fully operational in an in-vehicle network, the firmware will be integrated with the ECU's. During this activity users like to view the physical layer activity in data link layer format of the OSI layer. Figure 11 shows a typical in-vehicle network.

In Figure 11, multiple buses are interconnected using gateway. Comfort bus and infotainment bus normally operated less than 125Kbps and power train bus and safety bus operating at 500Kbps to 1Mbps data rate. The LIN bus is interfaced to comfort bus.



Trigger - CAN

A:CAN → Acquire

Trigger Type: CAN Bus

Settings: Independent

Signal Type: Differential

Data Sou: Ch 1

Data Thres: 900m\

Nominal Bit: 1.0MB

Trigger On: Start

▶ **Figure 12.** *Trigger Setup in TDSVNM*

There will be exchange of data between different buses using the gateway. In this heterogeneous network, there is need for following analysis.

– Viewing information in data link layer format by capturing bus traffic by triggering on specific CAN frame contents.

– Time stamp between the CAN messages

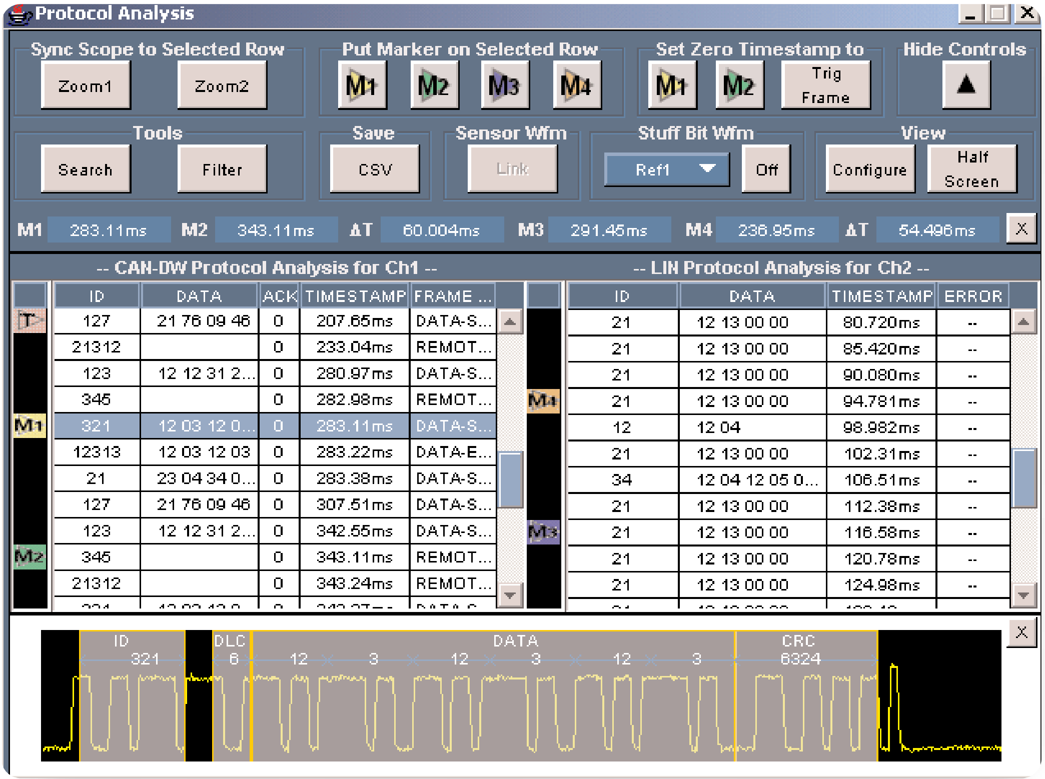
– Characterize the communication between different segments of the network

– Ability to simultaneously view the protocol activities in two segments of the network. Two segments could be CAN-CAN or CAN-LIN

– Characterize the gateway performance for latency time for exchanging the message between the segments and accuracy of exchanging the message

The protocol decoding capabilities offered by the DPO7000 series and the LSA option, that brings together its integrated CAN specific trigger capabilities and the TDSVNM software provides industry leading solution to address the above needs.

Using the TDSVNM software that interfaces with the integrated CAN trigger features of the DPO7000 series, the users are able to capture CAN traffic at specific content of the CAN frame. TDSVNM solution can monitor either of the CAN 2.0A or CAN 2.0B frame. Users have the flexibility to select frame type such as data frame, overload frame, remote frame or error frame. Based on the content in the frame, user can specify the content to be monitored and initiate the capture. The DPO7000 trigger setup user interface is shown in Figure 12.



M1 283.11ms M2 343.11ms ΔT 60.004ms M3 291.45ms M4 236.95ms ΔT 54.496ms

-- CAN-DW Protocol Analysis for Ch1 --

	ID	DATA	ACK	TIMESTAMP	FRAME ...
T	127	21 76 09 46	0	207.65ms	DATA-S...
	21312		0	233.04ms	REMOT...
	123	12 12 31 2...	0	280.97ms	DATA-S...
	345		0	282.98ms	REMOT...
M1	321	12 03 12 0...	0	283.11ms	DATA-S...
	12313	12 03 12 03	0	283.22ms	DATA-E...
	21	23 04 34 0...	0	283.38ms	DATA-S...
	127	21 76 09 46	0	307.51ms	DATA-S...
	123	12 12 31 2...	0	342.55ms	DATA-S...
M2	345		0	343.11ms	REMOT...
	21312		0	343.24ms	REMOT...

-- LIN Protocol Analysis for Ch2 --

	ID	DATA	TIMESTAMP	ERROR
	21	12 13 00 00	80.720ms	--
	21	12 13 00 00	85.420ms	--
	21	12 13 00 00	90.080ms	--
M4	21	12 13 00 00	94.781ms	--
	12	12 04	98.982ms	--
	21	12 13 00 00	102.31ms	--
	34	12 04 12 05 0...	106.51ms	--
	21	12 13 00 00	112.38ms	--
M3	21	12 13 00 00	116.58ms	--
	21	12 13 00 00	120.78ms	--
	21	12 13 00 00	124.98ms	--

▶ **Figure 13.** *CAN and LIN Protocol Decoded Information*

Captured data is in YT format. It is very difficult to interpret the NRZ data and know the content in CAN message. TDSVNM decodes the YT waveform and displays the data in user selectable Hex or binary format, saving time and not prone any human errors. The protocol decoded result is shown in Figure 13.

The TDSVNM software offers various analysis tools that will allow user to analyze the data in many forms: For example it uses time stamp with reference to trigger condition with the flexibility to select the desired frame for reference using a marker. By placing two markers between any two desired CAN message, user can know the time between the two messages. You can find the latency time of the communication and the accuracy of that data exchange between different buses, simultaneously. No other oscilloscope software in the market can do this.

CAN triggering features	DPO7000 Series built-in CAN triggers	ATM1 trigger module
Elementary	Data frame Error frame Remote frame Overload frame ID, Data, ID & Data ACK Start/End of Frame	Data frame Error frame Passive and Active Remote frame Overload frame ID, Data, DLC ACK, Form, CRC
Advanced		ID and Data values can be set to <, >, ?, within and out of range Logical OR among any three CAN 2.0A/B messages If-then-else state triggering Number of states: 3 Maximum number of events per if/then: 2 Maximum number of trigger actions per if/then: 3 Number of counters and timers: two each

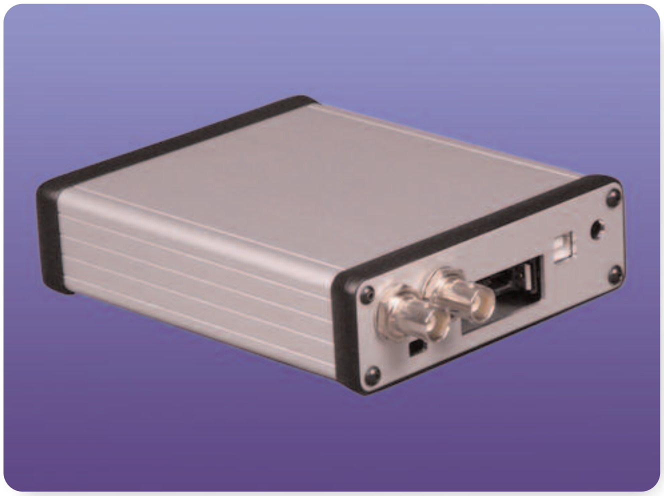
▶ **Table 1.** *Trigger features with ATM1.*

Doing Even More Protocol Debugging With More Trigger Features

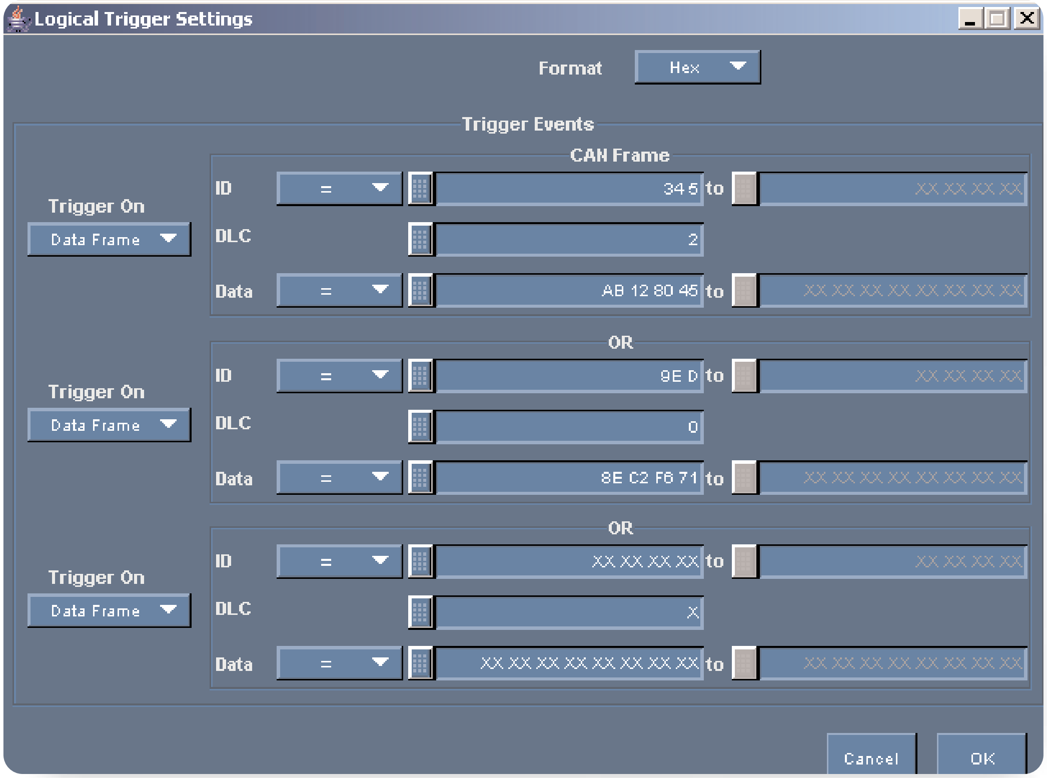
Using the build-in trigger features of the DPO7000 series, users can also synchronize to the start or the end of a frame, to a specific identifier, data or even a missing acknowledge or even an error within a frame.

However, if users need even more debugging capabilities like complex triggering and data filtering, the ATM1 automotive trigger module from Crescent Heart Software provides an industry leading solution to address these needs. Using ATM1, users can trigger on the DLC within the frame, passive error frame or active error frame and can also perform cascaded triggers or if-then-else triggers.

Table 1 details the trigger features available with ATM1 automotive trigger module.



▶ **Figure 14.** *ATM1, External Trigger Module.*



Logical Trigger Settings

Format: Hex

Trigger Events

CAN Frame

Trigger On: Data Frame
ID = 34 5 to XX XX XX XX
DLC 2
Data = AB 12 80 45 to XX XX XX XX XX XX XX XX

OR

Trigger On: Data Frame
ID = 9E D to XX XX XX XX
DLC 0
Data = 8E C2 F6 71 to XX XX XX XX XX XX XX XX

OR

Trigger On: Data Frame
ID = XX XX XX XX to XX XX XX XX
DLC X
Data = XX XX XX XX XX XX XX XX to XX XX XX XX XX XX XX XX

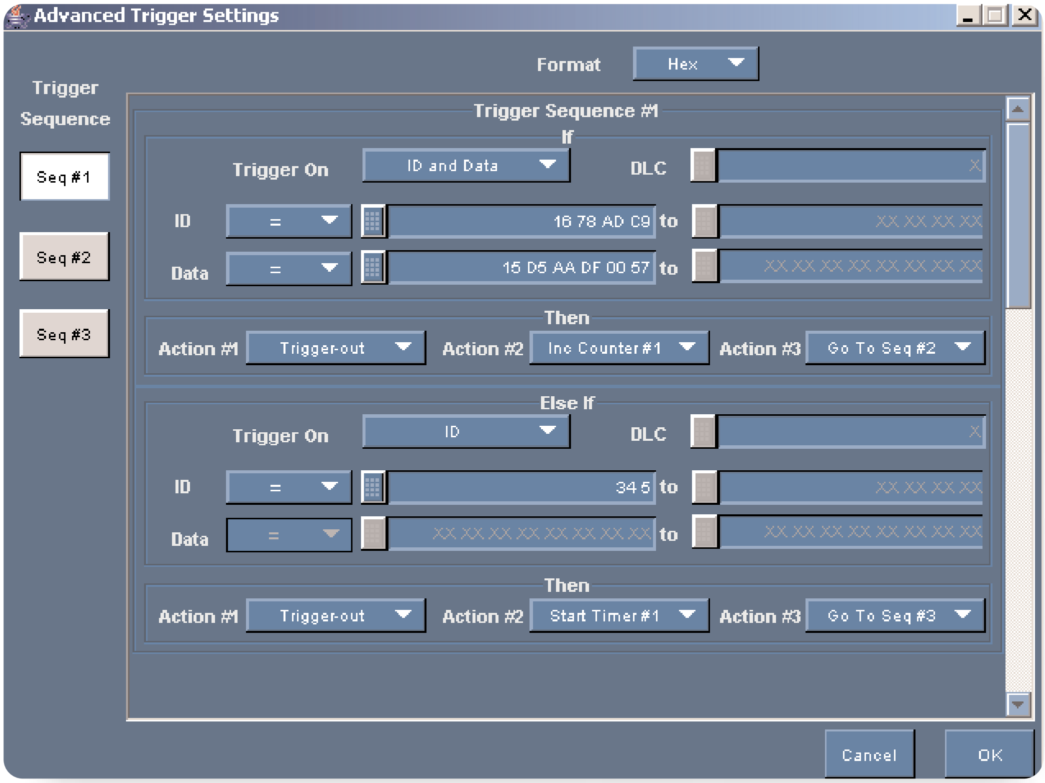
▶ **Figure 15.** *Logical trigger setup between any three frame content.*

Figures 15 and 16 show how the ATM1 trigger features can be set up in the TDSVNM software. The LSA option, that combines integrated CAN specific trigger capabilities and the TDSVNM CAN and LIN timing and protocol decode software transforms your general purpose Tektronix DPO7000 series oscilloscope into a sophisticated in-vehicle network management tool. It provides industry-leading capabilities:

– Measuring Oscillator tolerance and propagation delay to save time

– Measuring Bus Utilization and data rate to enable the efficient use of the network

– Analyzing the CAN Eye diagram to locate noise problems in CAN message



Advanced Trigger Settings

Format: Hex

Trigger Sequence: Seq #1, Seq #2, Seq #3

Trigger Sequence #1

If

Trigger On: ID and Data — DLC: X

ID: = 16 78 AD C9 to XX XX XX XX

Data: = 15 D5 AA DF 00 57 to XX XX XX XX XX XX XX XX

Then

Action #1: Trigger-out — Action #2: Inc Counter #1 — Action #3: Go To Seq #2

Else If

Trigger On: ID — DLC: X

ID: = 34 5 to XX XX XX XX

Data: = XX XX XX XX XX XX XX XX to XX XX XX XX XX XX XX XX

Then

Action #1: Trigger-out — Action #2: Start Timer #1 — Action #3: Go To Seq #3

▶ **Figure 16.** *If-then-else triggers to monitor multiple trigger conditions.*

- Capturing CAN and LIN messages resulting based on the content of the CAN frame and viewing it in protocol decoded format
- Capturing simultaneously the CAN and LIN data and viewing it in protocol decoded format to locate latency time at the gateways.
- Advanced triggering both for physical layer debugging and protocol decoding
- Complex triggering and data filtering with the additional ATM1 Trigger module.

Let the DPO7000 Series get you greater insight into your in-vehicle network and get your work done faster to ensure a seamless and reliable operation.

Reference

- Application Note from Philips AN97046
- Technical Paper *The configuration of the CAN Bit Timing* by Bosch

Contact Tektronix:

ASEAN / Australasia / Pakistan (65) 6356 3900
Austria +41 52 675 3777
Balkan, Israel, South Africa and other ISE Countries +41 52 675 3777
Belgium 07 81 60166
Brazil & South America 55 (11) 3741-8360
Canada 1 (800) 661-5625
Central East Europe, Ukraine and Baltics +41 52 675 3777
Central Europe & Greece +41 52 675 3777
Denmark +45 80 88 1401
Finland +41 52 675 3777
France & North Africa +33 (0) 1 69 86 81 81
Germany +49 (221) 94 77 400
Hong Kong (852) 2585-6688
India (91) 80-22275577
Italy +39 (02) 25086 1
Japan 81 (3) 6714-3010
Luxembourg +44 (0) 1344 392400
Mexico, Central America & Caribbean 52 (55) 56666-333
Middle East, Asia and North Africa +41 52 675 3777
The Netherlands 090 02 021797
Norway 800 16098
People's Republic of China 86 (10) 6235 1230
Poland +41 52 675 3777
Portugal 80 08 12370
Republic of Korea 82 (2) 528-5299
Russia & CIS 7 095 775 1064
South Africa +27 11 254 8360
Spain (+34) 901 988 054
Sweden 020 08 80371
Switzerland +41 52 675 3777
Taiwan 886 (2) 2722-9622
United Kingdom & Eire +44 (0) 1344 392400
USA 1 (800) 426-2200

For other areas contact Tektronix, Inc. at: 1 (503) 627-7111

Last Updated June 15 2005

For Further Information
Tektronix maintains a comprehensive, constantly expanding collection of application notes, technical briefs and other resources to help engineers working on the cutting edge of technology. Please visit **www.tektronix.com**

Copyright © 2005, Tektronix, Inc. All rights reserved. Tektronix products are covered by U.S. and foreign patents, issued and pending. Information in this publication supersedes that in all previously published material. Specification and price change privileges reserved. TEKTRONIX and TEK are registered trademarks of Tektronix, Inc. All other trade names referenced are the service marks, trademarks or registered trademarks of their respective companies.

10/05 FLG/BT 61W-19063-1

Tektronix
Enabling Innovation