Optimizing Test System Throughput, Cost and Integration Time Using LAN-Based Instruments

Application Note

As instrument control busses have evolved over time, from GPIB in the early 1970's to FireWire and VXI in the 80's and 90's, to PXI today, data transfer speeds have increased from about 1 MBits/s to over 400 MBits/s and have always required special interface cards to be added to PCs to enable their use. During the same time period, Ethernet has increased its speed from 3 MBits/s to 10 Gigabits/s and has become commonplace on most modern PCs. See **Figure 1**.

LAN was not originally designed for instrument control but for high speed data transfer of large *packets*. There is an overhead associated with every data transmission, called *latency*, which can make the short bursts of data that are common with instrument control take longer to transfer. So, although LAN control is inexpensive and easy to get working, throughput-critical applications need to exercise care to make the latency issue less of a factor. This application brief explains how to minimize latency when using LAN to control your instrumentation and explains why LAN is the better choice for cost-savings and ease of integration.

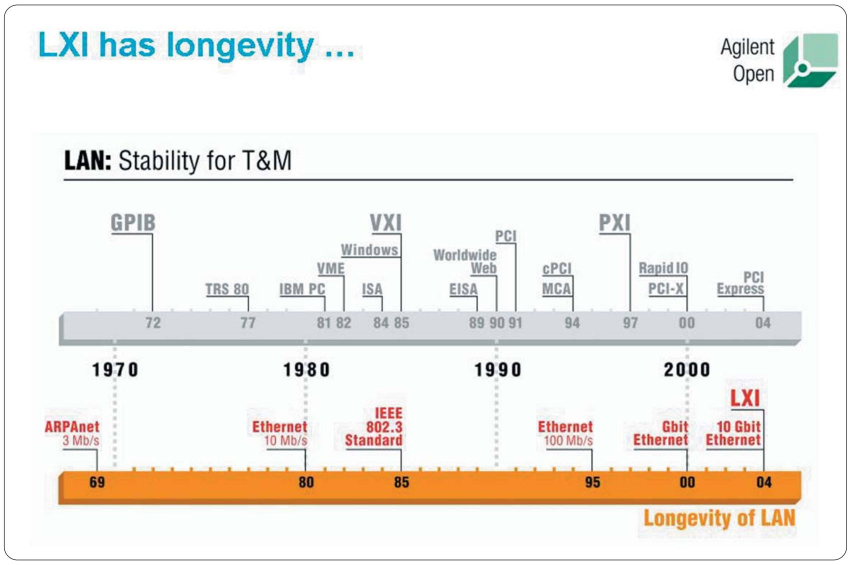


Figure 1. LAN has been with us as other busses have come and gone

Agilent Technologies

Latency

You may have seen the chart similar to the one shown in **Figure 2** at some National Instruments (NI) seminars that shows the effect of bus latency. By using standard, un-optimized LabVIEW drivers, NI showed that the execution time of a switch toggle sequence (closing a relay, testing for closed, opening the relay, testing for open) using an NI PXI reed multiplexer card appears to be 127% faster (3.74 ms vs. 8.3 ms) than that of a LAN-controlled Agilent 34980A Switch/Measure Unit using a similar switch card. But this is not the whole story. It is possible for the 34980A to match the speed of PXI in this example. When one controls the 34980A using native SCPI commands, the toggle time goes down to 3.9 ms on LAN, virtually the same as the PXI relay card.

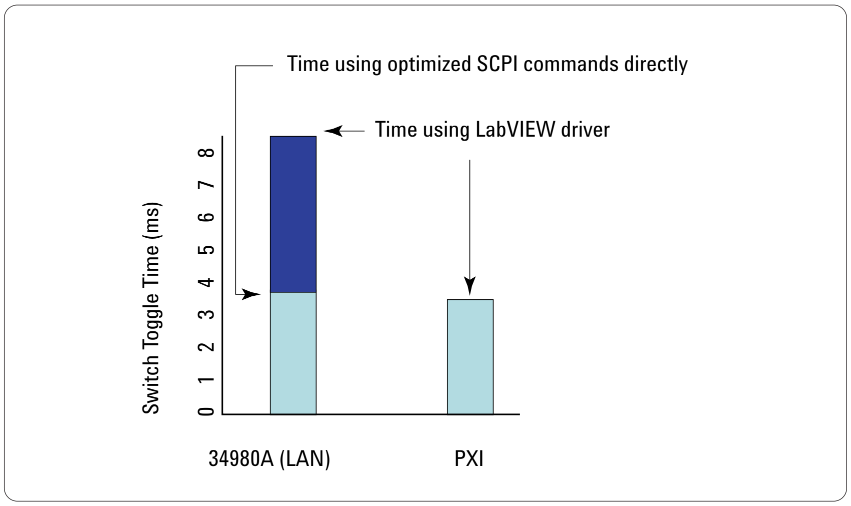


Figure 2. Time to close and open a single channel on a reed relay mux card using an NI PXI reed relay card and an Agilent 34980A with a similar reed relay card.

Why is SCPI faster than the Agilent LabVIEW driver in this case?

When you send a command to an instrument over any interface, you have two choices of how that command actually makes its way from your program to the hardware:

- **Drivers** – software that provides function calls, available in a variety of programming environments, such as LabVIEW, Basic, C++, etc. Drivers for register-based instruments like PXI create the actual register data needed to set up an instrument and read data from it
- **Direct I/O using SCPI (“Standard Commands for Programmable Instruments”)** – industry standard ASCII command language used nearly universally on GPIB instruments, also usable via the VXI-11 specification with LAN and USB. Can be sent to many interfaces from many languages using simple low-level read and write commands

Drivers can be written to control SCPI-based instruments too. Such drivers actually emit SCPI as their end result, so they can be inherently slower than programs that send SCPI directly because they must spend time constructing the SCPI command before sending it. However, they can also be faster than manually generated SCPI if they keep track of state information (i.e., sending only changed information to the instrument), and if they concatenate commands efficiently. Unfortunately, there is very little way to tell ahead of time if a driver is coded efficiently. As we will see shortly, bus latency can add dramatically to execution times if commands that can be sent as a single string are instead sent as separate strings.

Because all PXI devices and most VXI devices are register-based, not ASCII-based, they require the use of a driver provided by the card vendor for your chosen development environment. This driver in turn communicates with an interface driver that handles the mundane tasks of sending and receiving data on the chosen interface (LAN, USB, GPIB, RS-232, FireWire, MXI-4, etc.). Interface drivers are low-level software routines that are part of the I/O libraries that you install. The two leading vendors of these I/O libraries are Agilent and NI. In the case of PXI via a remote PC, the interface driver communicates over the NI proprietary MXI interface using a proprietary interface card that you must first install in your PC. GPIB interfaces also typically require a proprietary card, available from several vendors, that must be plugged into your PC. LAN and USB interfaces, in contrast, are standard on most PCs and require only the use of inexpensive interconnect cables.

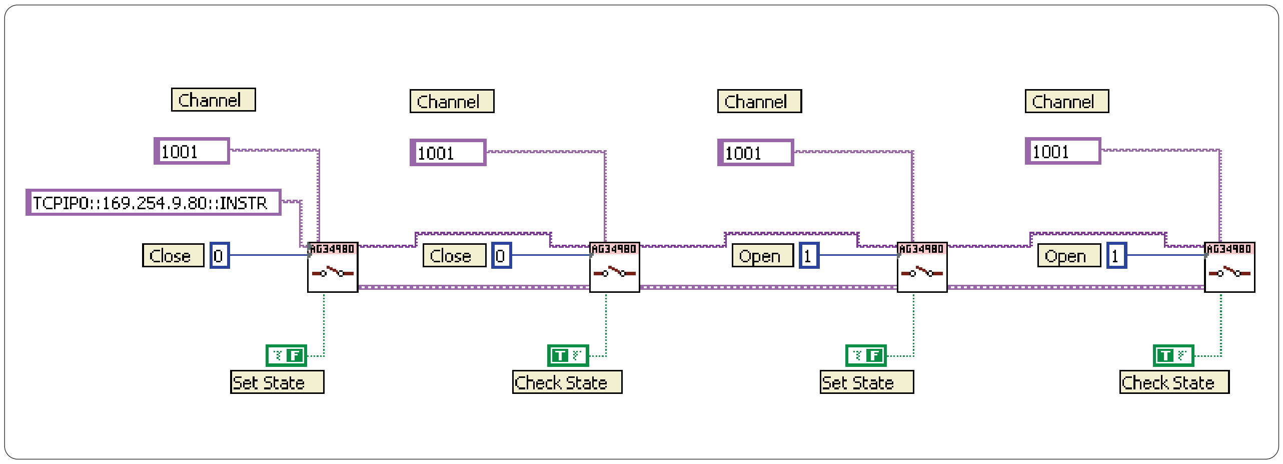
When you use an instrument driver, you are relying on the person who wrote the driver to have implemented the necessary functionality correctly and efficiently. In the case of register-based devices, it is required that the driver writer implement 100% of the instrument functionality, since that is the only way to control the instrument. With SCPI-based devices, drivers frequently do not have all of the possible instrument functions implemented. There is sometimes a “SCPI Pass-Through” mode though, allowing you to send any additional commands directly. You can also use VISA or VISA-COM to send SCPI directly to the instrument, saving you the extra effort to download, install and learn the instrument-specific driver. However, you then need to learn the relevant SCPI commands for the instrument. This is not necessarily a bad thing, however! It is not uncommon to find defects in vendor-supplied software drivers. If you take the time to understand the SCPI commands that control the instrument, debugging is much easier because you can turn on an I/O Monitor such as the one provided in Agilent’s T&M Toolkit or NI Spy and observe the SCPI messages that the driver is sending. If you find a bug in a driver that uses SCPI, you can code around it yourself while waiting for an update from the vendor. If you find a bug in a register-based driver, you have no choice but to wait for a fix or to not use that particular function.

Drivers for register-based instruments need only send several bytes of data over a very fast backplane, and no parsing (decoding) is required by the instrument. Since SCPI is an ASCII-based language, it takes longer to send the strings down an interface, and it requires parsing inside the instrument. The total time to handle this transfer can be well over 1 ms with modern instrument processors and high speed LAN. In fact, the time to send a single byte or a hundred bytes over LAN is about 1 ms with 100BaseT Ethernet interfaces.

How do I get around the latency issue?

In the switch toggle example mentioned previously, driver calls in LabVIEW resulted in four commands to the instrument and two responses from the instrument. Using PXI, this was fast, although it still took 3.74 ms to close and open a reed relay that really can close and open in less than 1 ms total. It is an unsolved mystery as to why the PXI relay took so long to toggle, even despite the higher backplane speeds. This implies that the lower latency times of PXI do not necessarily translate into higher throughput.

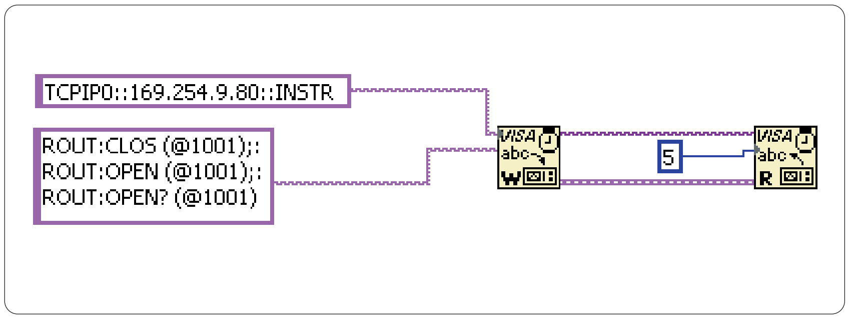
It is possible to construct SCPI commands that will optimize the LAN traffic into only 2 packets, thus saving about 4.4 ms and allowing the 34980A to toggle a relay in essentially the same time as its PXI competition – 3.9 ms. Let's look at the relay toggle example as implemented in LabVIEW:



This LabVIEW program emits the following SCPI code:

ROUT:CLOS (@1001)	1.4 ms	(Set state)
ROUT:CLOS? (@1001)	1.3 ms	(Check state)
Read“1”	1.5 ms	
ROUT:OPEN (@1001)	1.6 ms	(Set state)
ROUT:OPEN? (@1001)	1.0 ms	(Check state)
Read “1”	1.5 ms	
Total:	8.3 ms (6 LAN packets)	

By careful use of concatenated SCPI commands, this can be greatly simplified. However, instead of using the LabVIEW driver for the 34980A, you would use VISA calls:



Note: There was no need to execute a ROUT:CLOS? (@1001) between the CLOS and OPEN because the instrument does not execute the OPEN until the CLOS is done. If you added it anyway, the Read would return "1,1" and there would be no impact on execution time. This points out the benefit of understanding all the nuances of your instrument in order to extract the best possible throughput.

This LabVIEW program emits the following code:

```
ROUT:CLOS(@1001);:ROUT:OPEN (@1001);:ROUT:OPEN? (@1001)  1.4 ms
Read "1"                                                 2.5 ms
-----------------------------------------------------------------
Total:                                    3.9 ms (2 LAN Packets)
```

From a textual language such as Visual Basic, such commands can be output directly using VISA or VISA-COM statements. For example, using Agilent's T&M Toolkit in the .NET environment, a typical statement is:

```
myAgilent34980A.WriteLine ("ROUT:CLOS (@1001);:ROUT:OPEN (@1001);:ROUT:OPEN? (@1001)")
s = myAgilent34980A.Read()
```

Other considerations

Speed is not the only factor you must consider when choosing a control interface. Cost and ease of implementation are two other big factors. **Figures 3** and **4** point out these issues.

3 Rack Units

Agilent 34980A with 8 slots *and* a 6.5-digit DMM: $2350 USD

LAN Cable: ~$5 USD

No driver installation or PC reboot required

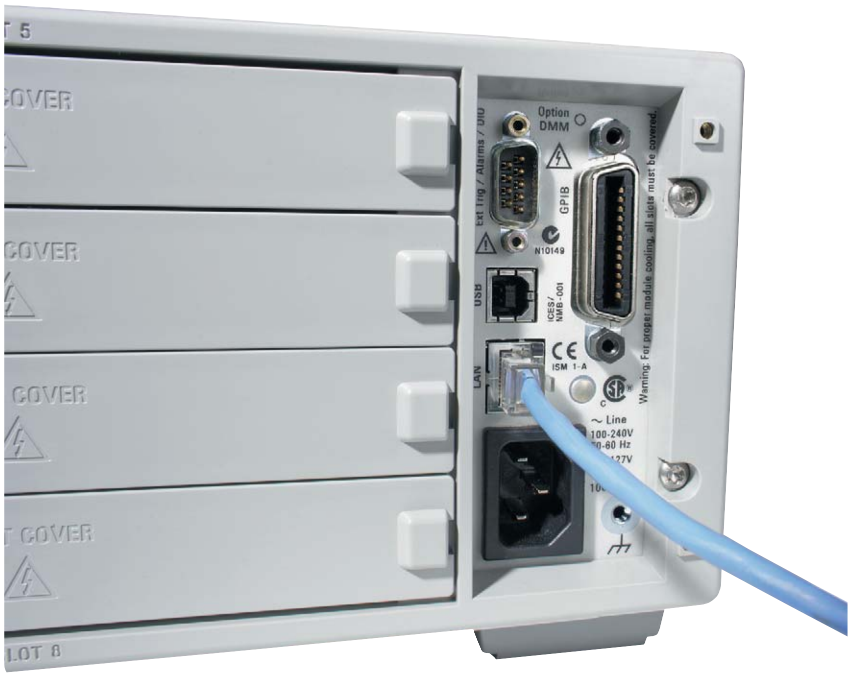


Figure 3. Rear panel of Agilent 34980A showing LAN cable. This inexpensive cable connects directly to many PCs without the need for an installed I/O card

4 Rack Units

NI PXI 8-slot cage with 6.5/7.5-digit DMM (7 slots left): ~$4000 USD (picture shows older model with 4 SCXI slots also)

MXI-4 cable and I/O card for PC and for PXI cage: ~$1500 USD

Requires driver installation and PC reboot.

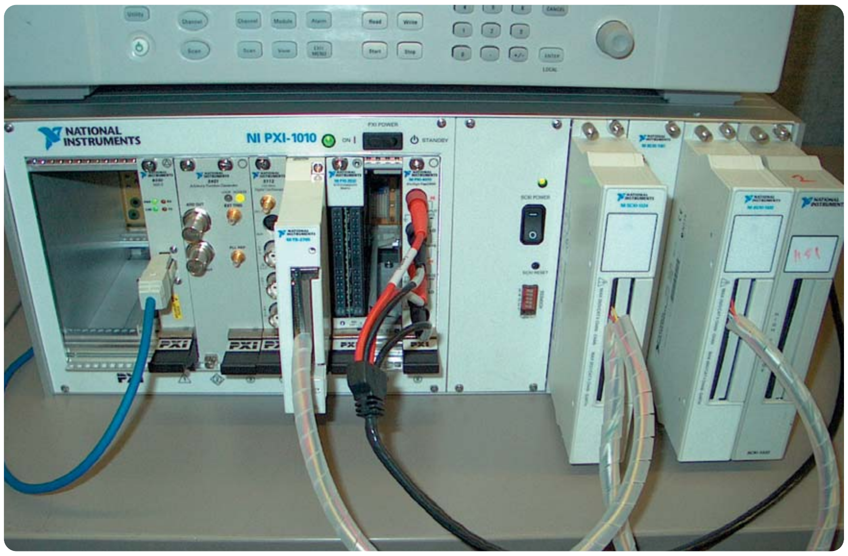


Figure 4. An NI PXI/SCXI cage with an NI MXI-3 interface. PXI cardcages typically require the use of either embedded PCs or external PCs with controller cards installed in them.

The difference between using a 34980A and using an equivalent 8-slot PXI chassis with a MXI interface and a DMM:

- Savings of over $3000 USD using the 34980A
- One extra slot using the 34980A
- Easier install process with no PC reboot required using the 34980A
- One fewer EIA units of rack space (3 U vs. 4 U) using the 34980A. Think what you could put in that extra 1 U. How about an Agilent N6700 modular power supply or an Agilent N5700 high power supply?

Summary

If program execution time is important to you, think about the code that is being emitted by your program and try to concatenate as many SCPI strings as possible to save LAN overhead. Drivers do not typically do this as efficiently as a human, so when ultimate throughput is your goal, consider using SCPI, using VISA or VISA-COM to control the instrument yourself rather than relying on a driver.

If cost is more important to you, the 34980A is often a much lower cost solution.

If rack space is paramount, the 34980A uses one less EIA unit than a PXI cage, giving you space for a high power or modular power supply from Agilent.

Agilent Technologies' Test and Measurement Support, Services, and Assistance
Agilent Technologies aims to maximize the value you receive, while minimizing your risk and problems. We strive to ensure that you get the test and measurement capabilities you paid for and obtain the support you need. Our extensive support resources and services can help you choose the right Agilent products for your applications and apply them successfully. Every instrument and system we sell has a global warranty. Two concepts underlie Agilent's overall support policy: "Our Promise" and "Your Advantage."

Our Promise
Our Promise means your Agilent test and measurement equipment will meet its advertised performance and functionality. When you are choosing new equipment, we will help you with product information, including realistic performance specifications and practical recommendations from experienced test engineers. When you receive your new Agilent equipment, we can help verify that it works properly and help with initial product operation.

Your Advantage
Your Advantage means that Agilent offers a wide range of additional expert test and measurement services, which you can purchase according to your unique technical and business needs. Solve problems efficiently and gain a competitive edge by contracting with us for calibration, extra-cost upgrades, out-of-warranty repairs, and on-site education and training, as well as design, system integration, project management, and other professional engineering services. Experienced Agilent engineers and technicians worldwide can help you maximize your productivity, optimize the return on investment of your Agilent instruments and systems, and obtain dependable measurement accuracy for the life of those products.

Agilent Email Updates

www.agilent.com/find/emailupdates
Get the latest information on the products and applications you select.

Agilent Direct

www.agilent.com/find/agilentdirect
Quickly choose and use your test equipment solutions with confidence.

Agilent Open

www.agilent.com/find/open
Agilent Open simplifies the process of connecting and programming test systems to help engineers design, validate and manufacture electronic products. Agilent offers open connectivity for a broad range of system-ready instruments, open industry software, PC-standard I/O and global support, which are combined to more easily integrate test system development.

www.agilent.com

For more information on Agilent Technologies' products, applications or services, please contact your local Agilent office. The complete list is available at:

www.agilent.com/find/contactus

Phone or Fax

United States:
(tel) 800 829 4444
(fax) 800 829 4433

Canada:
(tel) 877 894 4414
(fax) 800 746 4866

China:
(tel) 800 810 0189
(fax) 800 820 2816

Europe:
(tel) 31 20 547 2111

Japan:
(tel) (81) 426 56 7832
(fax) (81) 426 56 7840

Korea:
(tel) (080) 769 0800
(fax) (080) 769 0900

Latin America:
(tel) (305) 269 7500

Taiwan:
(tel) 0800 047 866
(fax) 0800 286 331

Other Asia Pacific Countries:
(tel) (65) 6375 8100
(fax) (65) 6755 0042
Email: tm_ap@agilent.com
Contacts revised: 09/26/05

Product specifications and descriptions in this document subject to change without notice.

Agilent Technologies