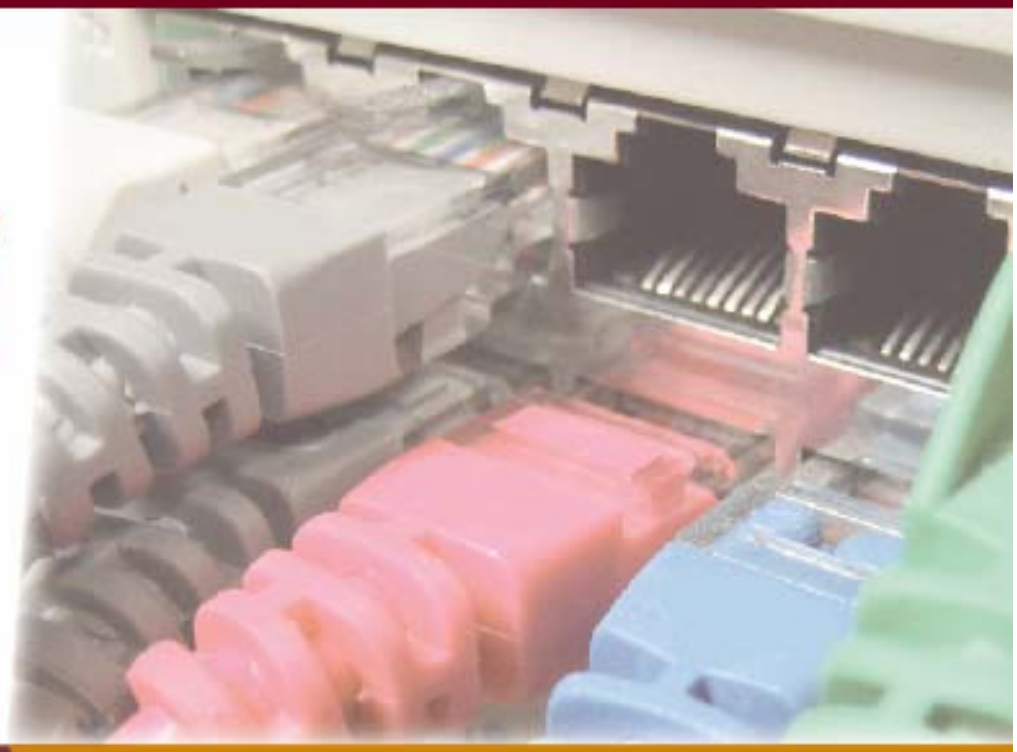
LAN eXtensions for Instrumentation



LXI Triggering, Revision 1.0
December 1st, 2005

White Paper No. 2

Table of Contents

Dan Pleasant
Agilent Technologies

Introduction

This paper describes LXI triggering: What it is, how to use it, and how it differs from prior triggering systems.

Test and measurement systems have always had a need for tight synchronization to control the timing of the various events that occur during a test. Traditionally, external trigger input/output connectors are placed on each instrument, with firmware that allows the instrument to be configured to react to received trigger signals or to output triggers when an internal event occurs. In addition, traditional instruments usually synchronize their own internal functions in a similar manner – a capability that is lost in newer synthetic instrument-based system designs and must instead be accomplished at the system level.

The LXI standard supports traditional triggering models of legacy instrumentation while adding new triggering capabilities that allow for more convenient and precise triggering. While traditional triggering will not be entirely replaced by LXI triggering, these new capabilities supply extra degrees of freedom to the system integrator/developer.

Intended audience

The paper is useful for those who wish to understand LXI triggering in a general sense and it serves as an introduction to the programming concepts that underlie the LXI triggering design.

The LXI standard includes a programming interface that allows test implementers to access the triggering capabilities of every LXI instrument in a consistent way. Although this paper does not cover the details of the LXI triggering API, users may find that this paper is an excellent prerequisite for understanding the details of the API.

General description of LXI triggering

The LXI specification defines three classes of instruments:

- Class C: Includes standard LAN connectivity and configuration features. This class of instruments supports traditional triggering methods but does not include any LXI triggering extensions.

- Class B: Includes Class C features with IEEE 1588 clock synchronization and LAN-based triggering. This class of instrument can use time-based triggers and LAN-based triggers in addition to the capabilities of Class C instruments.
- Class A: Includes Class B features with LXI trigger bus standard hardware triggering. This class of instrument may use LXI trigger bus-based hardware triggering in addition to all of the capabilities of Class B and C instruments.

With IEEE 1588 clock synchronization, Class A and B instruments carry the ability to use time-based triggering functions and to respond to triggers on the LAN. Class A adds a special programmable hardware bus, similar to the triggering available in VXI instruments, that can be used when time-based or LAN-based triggering is not appropriate.

Any class of LXI device may include traditional (or non-traditional) hardware trigger inputs and outputs, at the vendor's option. Class A and B instruments bring standard triggering features with a uniform controlling software API that will allow software to control hardware from different vendors with few changes.

IEEE 1588

The IEEE 1588 specification creates a method by which the various devices on a network can synchronize their clocks to a high degree of precision.[1] The achievable clock synchronization accuracy depends on the quality of the clocks in each device and the network configuration, but accuracies of less than 100 nsec are achievable with standard available hardware and accuracies of less than 10 nsec have already been achieved in laboratory conditions.

One of the convenient aspects of the IEEE 1588 specification is that it creates a clock synchronization system that is nearly completely transparent to all other network functionality. When an IEEE 1588-enables device is added to a network, it automatically senses the presence of other IEEE 1588-enabled devices on the network and announces its presence. The system automatically chooses the most stable clock and uses it as a reference, and will automatically reconfigure itself as devices are added or removed from the network. These functions all operate with no other required interaction.

While the IEEE 1588 standard supplies a standard means for synchronizing clocks, it has nothing to say about what a system might be able to accomplish once its clocks have been synchronized. The LXI specification fills in this gap by adding time-based triggers and a standard way to add time information to LAN-based triggers. Described in more detail in the following sections of this paper, this part of the LXI specification makes IEEE 1588 a useful capability in the test and measurement world.

LXI Triggering

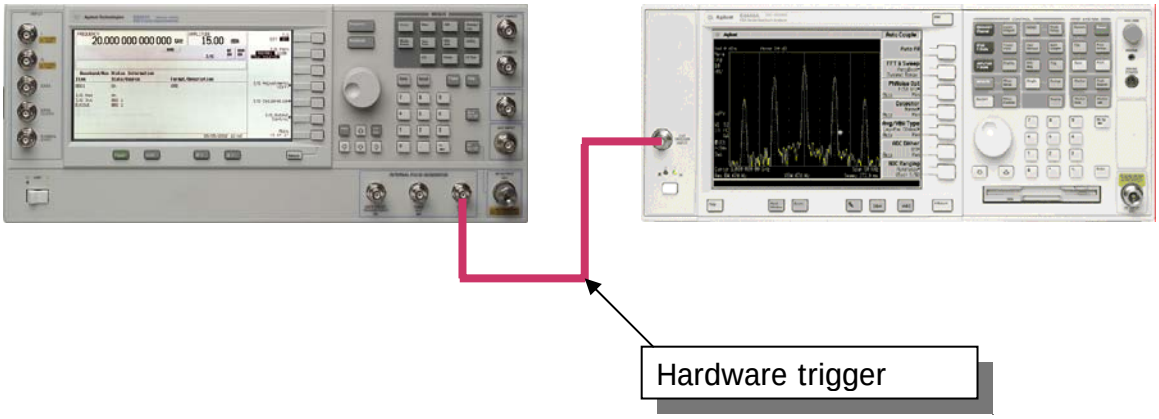
[1] More information on the IEEE 1588 specification may be obtained at http://ieee1588.nist.gov.

Although it will not be practical in all cases, the LXI specification recommends that LXI devices should be capable of triggering any available action by any available means. For example, if a measurement can be triggered by a traditional hardware trigger line, then it should also be triggerable by a LAN trigger, or by the LXI trigger bus, or by any other trigger input that is supported by the device. The test developer should be able to configure the device to accept the appropriate trigger source and route it to the desired action.

Since the LXI specification includes several different types of trigger inputs/outputs, LXI devices that implement this level of triggering flexibility will have several choices to make. The following paragraphs give a more detailed description of each LXI-supported triggering method. Later, this paper includes a discussion of the tradeoffs that must be considered when choosing between different triggering methods.

Traditional hardware triggering

All classes of LXI instruments are allowed to support traditional hardwired triggering inputs and outputs such as the connection shown in the figure below.

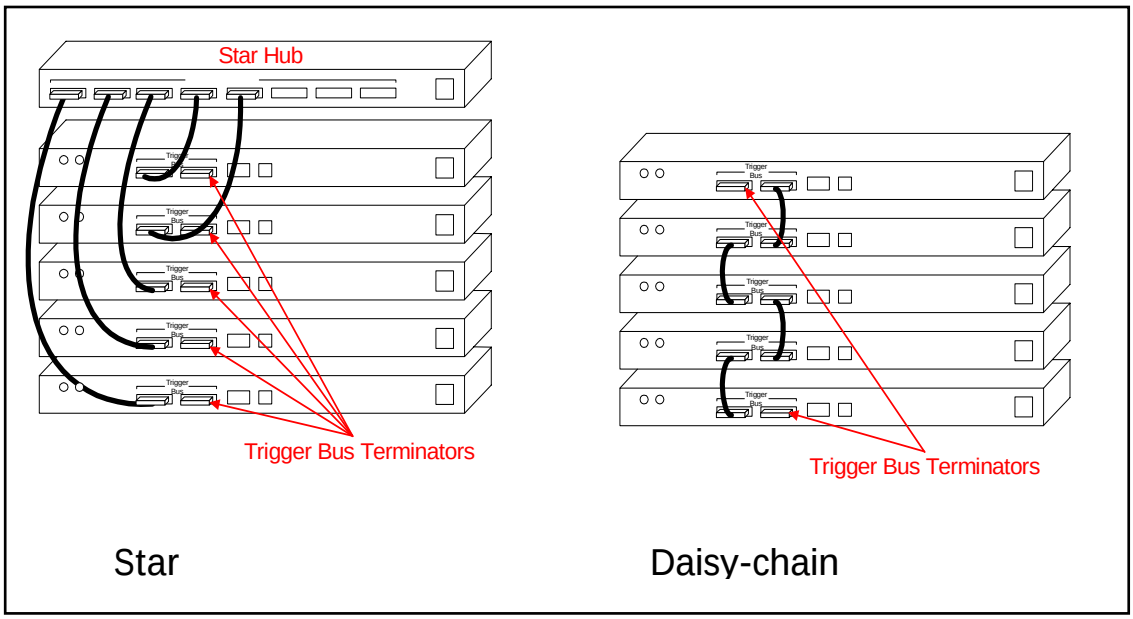


Each instrument shown here has a triggering connection on the front panel. The instruments must be manually connected, usually in some simple fashion such as a coaxial cable with BNC connectors on either end. In operation, one of the instruments transmits a trigger signal and the other responds to it. The receiving unit's response is generally pre-determined – if an instrument has more than one action that can be triggered, it will have a like number of trigger input connectors.

The LXI specification permits all traditional triggering methods to be implemented on any class of LXI instrumentation, without restriction. In addition, the programming API that controls these triggering functions is not required to be consistent with the LXI triggering API.

LXI Trigger Bus

The LXI specification supports the use of traditional hardware trigger lines and adds a new standard set of eight programmable hardware triggers that are arranged in a bus configuration. The LXI trigger bus, which is required only on LXI Class A devices, is built using differential M-LVDS pairs with controlled impedances. Each of the eight trigger lines may carry an independent trigger signal. The Trigger Bus may be connected in either a daisy-chain configuration or a star configuration, as illustrated in the figure below.



It may be useful to think of the LXI trigger bus as a collection of hardware triggers that are connected as a bus – either daisy-chained or in a star configuration or in a hybrid configuration that uses both – to each instrument in a test system. In the usual mode of operation, one LXI device is designated to drive a single line on the bus at any one time, and all other instruments act as receivers. A different instrument may simultaneously drive a different line. As with LAN triggers, each instrument's response to an LXI trigger bus signal is programmable. Likewise, an instrument may use a line on the trigger bus to transmit an internal signal, since the function of any one trigger signal is not fixed.

The LXI trigger bus also supports a wired-OR mode of operation, in which many devices drive a line on the bus at the same time. The voltage on the line will remain constant until all devices stop driving the bus. This mode of operation will be useful whenever the system must wait until several instruments are ready to act.

Time-based Triggering

LXI Class A and B devices, which include IEEE 1588 clock synchronization capabilities, may include time-based triggers.[2] A time-based trigger is analogous to an alarm clock – it can be set

[2] To allow instrument manufacturers some leeway with hardware designs, the LXI specification does not absolutely require the implementation of time-based triggers. Instead, the specification makes a strong recommendation that

to "go off" at a predetermined time. At that time, the instrument can be programmed to perform some specific function. This allows the different instruments in a test system to execute complex sequences of events autonomously, without intervention by the system controller.

For instance, a digitizer may be programmed to begin a measurement at a specified time, end it after 1 second, wait 100 msec, and begin another measurement. A receiver in the same test system could be programmed to change its frequency during the 100-msec pause in the digitizer's operation. With IEEE 1588-based clock synchronization, these operations would be executed, with near-perfect timing, with no connections between the instruments other than LAN.

Obviously, the use of time-based triggers requires the ability to schedule instrument actions in advance. Time-based triggers cannot be used to respond to asynchronous events.

Implemented in hardware, time-based triggers have essentially no latency[3] and their performance is not degraded by increasing distance between devices. Their timing accuracy depends only on the accuracy of the IEEE 1588 clock synchronization. This creates a new and different source of triggering problems: jitter. If all clocks in a system are perfectly synchronized, instruments can be programmed to perform actions at precisely the right time. Unfortunately this is not the case – there will be residual timing jitter and offset (see the section on test system configuration for more details). From time to time, the IEEE 1588 clock synchronization algorithm will make small adjustments to the system clock, creating some uncertainty as to the precise timing of any time-based trigger. It is necessary for engineers to understand the difference between latency and jitter, and to use the type of triggering that is most appropriate for their application.

Some applications will not be sensitive to triggering jitter. For others, it will be found that triggering jitter is easier to handle than latency. Consider the case of a radar test range, where the distance between instruments may be considerable. The IEEE 1588 clock synchronization algorithm will align the system clocks, no matter how far apart they may be. Hardware trigger latencies are always roughly 1 nsec per foot. If instruments are far apart, then time-based triggering jitter will be less than hardware signal latency. For some applications, this may become a critical enabling technology.

For events that can be scheduled to occur at a known time and can tolerate some timing jitter, these triggers offer a very high-performance, low-latency triggering mechanism.

LAN Triggering

LAN-based triggers operate in essentially the same fashion as traditional hardware triggers: An electrical signal is transmitted over a wire. Upon receipt of the signal, the receiving instrument executes some pre-determined action. For LAN-based triggers, this analogy is correct but not

---

every Class A and B device should include one or more time-based triggers implemented in hardware. It is anticipated that most Class A and B devices will do so.

[3] It is possible to implement time-based triggers in software/firmware. The LXI spec allows this but recommends against it. If a time-based trigger is implemented in this fashion, CPU response times will add unpredictable latency to the triggering function.

complete. LAN-based triggers can carry information that hardware triggers cannot. In LXI instruments, this information includes a time stamp that is based on the synchronized system clocks.

In an LXI LAN-based trigger, the transmitted data packet must include the time that the originating event happened. This enables some important triggering models, particularly this provides the ability to produce triggers with zero effective delay. To achieve this, a measuring instrument continually makes measurements, when a time stamped trigger arrives, the instrument returns the measurement that corresponds to that time. This can be extended to provide negative or positive trigger delays and is only limited by the depth of the circular buffer that the instrument uses to store the measurements.

This is similar to the mechanism that is used inside scopes to provide pre-trigger. Using the LXI mechanisms, these tools are available to the system programmer and can be applied across instrument boundaries.

The LXI specification supplies two different techniques for communication of LAN triggers: broadcast triggers, which use the UDP protocol, and point-to-point triggers, which use TCP.

UDP-based Triggering

UDP-based triggers utilize the ability of the UDP protocol to implement multicast one-to-many communications. Any LXI module can send a trigger message on the LAN, and that message will be received by all modules on the same subnet (or outside the subnet, if routers are configured appropriately). The LXI programming interface defines a common publish-and-subscribe architecture to control the use of these triggers, which simply means that each module can be told when it should transmit a message and when it should respond to a received message.

The UDP protocol is faster (i.e. has lower latency) than the more common TCP protocol because it does not guarantee message delivery. In most test systems the use of modern LAN switches guarantees that UDP data will never be lost. For more critical use, the LXI specification supports a few techniques that can greatly enhance UDP reliability. However, for extremely long-range applications that use corporate extremely congested LAN, WANs or the Internet, the LXI specification also supports the use of TCP to transmit triggers.

TCP-based Triggering

TCP-based triggers are point-to-point communication vehicles – they cannot be used to trigger multiple instruments at the same time. Because TCP implements a complex combination of handshaking and data flow control, TCP-based triggers have a higher latency and lower maximum data throughput than UDP-based triggers.

A few applications will require the use of TCP-based LAN triggers. Most Internet routers are configured to block UDP multicast traffic; and without TCP's advanced protocols, UDP data is

more likely to be lost as it travels through longer distances. Other applications may require the virtual certainty of data delivery that TCP offers without the necessity of implementing any other protocols on top of UDP.

In summary TCP should be used where a point-point connection is needed, or where UDP reliability or routability could be a concern.

LAN packet format

To facilitate interoperability between instruments from different vendors, the LXI specification defines a specific data packet format that must be used for LAN triggering. This packet contains a bare minimum of information that is needed to allow instruments to communicate properly.

It is important to note some key aspects of the LXI packet format:

1. The data that can be sent is required to be contained in a single LAN packet. This is important both to facilitate low LAN trigger latencies and to eliminate the need for receiving devices to reassemble packets that have been split apart by the TCP/IP stack.
2. The LXI-specified portion of the packet is small. The remainder of the space is available for essentially unrestricted use by vendors and system integrators.
3. The LXI packets do not contain commands. This represents a departure from older protocols (such as VXI-11). The function of each packet is determined by the Event ID field and the configuration of the receiving device, as discussed later in this paper.

The LXI LAN packet contents are described as follows. (Refer to the LXI specification for complete details.)

Field	***Description***
HW Detect	Three bytes that contain the ASCII characters 'LXI'. This field may be used by packet sniffers to quickly detect LXI packets upon receipt and route them to a triggering function, reducing latency caused by the TCP/IP stack.
Domain	One byte that contains a user-settable value. Each LXI device is assigned to a domain; it's domain number is user-settable via a web page or API . When transmitting a LAN packet, each LXI device must put its own domain number into this field. Upon receipt, LXI devices must ignore packets if the 'Domain' field in the packet does not match the local domain setting. This allows multiple independent test systems to co-exist on the same LAN subnet without interfering with one another. Each system should be assigned a different domain number, and each instrument in a system should be given the same domain number.
Event ID	A 16-byte ASCII field that contains the name of the packet. Upon receipt of a packet, the device should examine this value. If the device has been pre-programmed to respond to a packet

	with the given ID, then it should execute the intended function. Otherwise it should ignore the packet.
Sequence	A 32-bit integer that is filled in by the transmitting device and incremented each time the device sends a new packet. This field may be used to detect packets that have been received multiple times or to detect lost packets.
Timestamp and Epoch	Two fields that, when combined, contain a IEEE 1588-defined format for time. The meaning of this time stamp is not specified by the LXI specification. Sometimes it will indicate the time at which an event occurred in the transmitting instrument. Other times it will indicate a time at which the receiving unit should execute some function. It is up to the test developer to properly configure all instruments so that they handle this value correctly for the task at hand. The preferred use model is for the timestamp to reflect the time of occurrence of some application defined event. Recipients are then programmed to take action with respect to this event time. This model maximizes the benefits of loose coupling particularly in modifying and extending existing systems.
Flags	A 16-bit field that contains binary flags that are used for miscellaneous purposes. Some of these flags are discussed elsewhere in this paper. Refer to the LXI specification document for full details.
Data Fields	A structured data area that is available for vendor-specific data transmission. This part of the packet is divided into different fields according to some simple rules (refer to the LXI specification for details). Each field is identified by a vendor-supplied ID number. IDs less than zero are reserved by the LXI Consortium for future standard useage.

The most important fields in a LAN trigger packet are the Event ID and Timestamp fields. Upon receiving a packet, an LXI device must examine the Event ID to determine what (if anything) should be done in response, and it must use the Timestamp field to extract the critical timing information needed to complete the operation successfully. A full discussion of this functionality is provided later in this paper.

Reserved Event IDs

The Event ID field is a 16-byte ASCII value that can contain any string. However, the LXI specification reserves some of these IDs for specific purposes.

A complete list of reserved Event IDs may be found in the LXI specification. Of particular importance are the names 'LAN0' through 'LAN7'. These eight IDs are reserved for a specific use, and all LXI Class A and B devices are required to support them.[4]

These names are reserved to help ensure LXI device interoperability and to facilitate hardware assistance for LAN trigger signals. As described in a preceding section of this paper, the LXI Trigger Bus carries eight trigger signals in hardware. The LXI programming API refers to these trigger lines by the names 'LXI0' through 'LXI7'. The reserved names for LAN triggers provide a logical equivalent to the LXI Trigger Bus signal names.

Proprietary Vendor Protocols in Data Fields

While the purpose and use of the Event ID field is defined by the LXI specification, the use of the data fields is not. Vendors are allowed to insert any data that they choose in this section of the LAN packet, subject to minimal formatting constraints.

Since it is not a part of the LXI specification, it cannot be assumed that data in this part of the LXI LAN packet will be interpreted and understood by LXI devices from different manufacturers. For this reason, the data fields should usually not be used in applications that require interoperability among a wide range of vendors' instruments.

As LXI device design matures, it is anticipated that vendor-specific data fields may be used in several different ways:

1. Proprietary data may be inserted into the data fields that allows a single vendor's instruments to work better with one another, giving them a market advantage over other vendors.
2. Vendors may publish the format and semantics of data and allow other instruments from other vendors to interpret it, creating a vendor-specific extension to the LXI standard.
3. The LXI standard may be extended in the future to use some of the space in the data fields.

Note that the use of the data fields makes it possible for vendors to use LXI LAN data packets for purposes other than triggering. Arbitrary data may be transmitted between LXI instruments using this format. The LXI specification does not prohibit this type of use, although vendors may find that the use of other communication channels would supply a more satisfactory solution.

Design Philosophy, Tradeoffs and Best Practices

[4] While the LAN packet format allows any 16-byte string to be placed in the Event ID field, the LXI specification does not require devices to support arbitrary values. Vendors may choose to support only specific values. However, vendors are required to support the values 'LAN0' through 'LAN7'.

To make the best use of LXI triggering, it's important to understand the design intent of the system and the tradeoffs that are inherent in it. This section includes discussion of a variety of relevant topics that may help users understand how to optimally configure and use LXI instruments.

System Configuration

In current T&M systems the individual instruments are typically configured via a test program executing in a control computer. These are usually command driven via some sort of IVI driver over a GPIB bus, or by a slot-0 controller in a VXI environment. The individual settings for each instrument or module are configured in a sequence of such commands. The configuration is usually updated by the same means at appropriate points in the test sequence or at the beginning of a new test. In effect there is a master controller that maintains and distributes configuration information to each device as the test sequence progresses. In this context configuration typically includes instrument state settings such as range, parameters, switch settings, etc. Any timing configuration is usually confined to things like sweep rates, hold-off times, sampling intervals and the like.

The LXI system permits exactly the same style of configuration. The obvious significant difference is that the LAN is used rather than a GPIB bus or a backplane. Gateways to these media clearly can permit configuration of existing devices from the LAN based LXI control computer. However LXI systems as they evolve may present some alternatives to the current method of configuration. These are summarized as follows:

- It is expected and permitted by the LXI standard that individual modules will have provision for accepting configuration information well beyond what is used in current practice. In particular it is expected that:
    - Modules will accept multiple configuration sets. Each of these will be named.
    - Configuration information may include:
        - All of the currently used configuration, e.g. state, intervals…
        - Time specifications based on the global timebase,
        - Bindings between event messages and named configurations
        - Executable specifications either hardware or firmware based
        - FPGA configurations
- The configuration can be changed by switching from the current to a new named active set based on any of the mechanisms allowed by the LXI specification:
    - A command from a controller via the LAN
    - An event message received via the LAN from a peer
    - Based on a time specification relative to the system global timebase established by IEEE 1588
    - Based on a hardware signal
- The distribution of configurations may be managed by multiple computers some of which may be remote from the location of the T&M installation.

In principle these new alternatives could be implemented to a greater or lesser extent in existing T&M architectures. However LAN communications and the style of programming and the attendant supporting services makes it more likely in an LXI system.

The likely advantages of these new configuration alternatives over present practice include:

- Possible increase in throughput. To the extent that a sequence of tests involved repeated use of configurations a savings in communication time normally used to reconfigure will be realized. The cost of course is that the modules will require memory to store the multiple communications.
- Increased flexibility is system design. This is a general benefit of distributed computing, of which configuration is just a part. It allows logical and physical clustering of related functions, which generally leads to simpler logic in each of the individual sections of the distribution.

Controlling the Actions of Instruments

In current T&M systems the data acquired by the instruments is transferred to the controller. In most cases this transfer is initiated by the controller via an IVI or similar command requesting a particular piece of data. If multiple sources of data are involved in a test the controller must ensure that the necessary ordering and association information between data from the several sources in established and maintained. This function is intimately intertwined with the control and configuration of instruments in current systems.

In an LXI system there are other options for collecting the data. For Class A and B LXI instruments all data is to be timestamped at the source, e.g. the module, based on the global timebase. Note that this does not mean that every element in a time series has an explicit timestamp – it is clearly possible and allowed that the timestamp be associated with the start of a series with a specified time interval between elements. Eventually even Class C instruments are expected to implement source level timestamping to some degree of accuracy. With source timestamped data the requirement on the receiving device to establish and maintain the temporal relationship is eliminated. Data can be delivered at a time optimized with respect to actions within the module itself or to reduce congestion on the LAN. More importantly this permits the data collection function to be separated from the control function. It can even be done in a separate computer.

While polling can be used it is much more desirable to use a publish-subscribe model of communication for data transmission (see below for a discussion of the publish-subscribe model). This allows multiple recipients of the data with less demand on the sources and is better suited to scheduled transfers. It is expected that much of the data transfer will be scheduled based on time or at major points in the execution of complex test sequences signaled by direct LAN messages.

Analyzing the Acquired Data

Data analysis is either done in a non-time critical mode after receipt or when some real-time decision must be made the analysis is done in real-time. This will not change in the LXI system. What will likely change is where it is done. It will certainly be separated from the logic remaining in the control computer.

If the analysis is non-time critical then it should be done in a program devoted specifically to this task. This will often be in a separate computer or at least as a distinct function in the computer that manages configuration and high-level supervision.

Real-time analysis should be done where it makes the most sense. Most often this will be in one of the LXI modules involved in the real-time decision and will be implemented as part of the code or FPGA configuration distributed when the test was setup. If the decisions are needed by multiple modules there are two alternatives: the decision can be communicated via the means discussed earlier, or the computation may be done in parallel in each of the modules and each arrives at the decision independently. The latter of course requires that each module subscribe to the required data.

LXI Triggering API

Although instrument control languages such as SCPI are not prohibited, the LXI specification requires vendors to implement an IVI driver for every instrument. Part of the standard IVI driver interface for LXI devices is used to control triggering functions.

One of the strengths of the LXI programming interface is its uniform approach to triggering. The API for LXI triggering treats all sources of triggering equally. There is no difference, other than the name, between a procedural call that uses a LAN trigger and a hardware trigger. Programmers will find that this uniform approach to trigger control means that only one set of API calls will need to be learned, and changing from one type of trigger to another will be easy.

LXI triggering is based on a set of names that are given to any triggering source. The names ‘LXI0’ through ‘LXI7’ are assigned to the eight lines on the LXI trigger bus; the names ‘LAN0’ through ‘LAN7’ are reserved for specific LAN-based triggers.

The details of the LXI triggering API will not be documented in this paper. Refer to the LXI API documentation for complete details. However, a simple example will illustrate the approach that has been used.

Imagine that an IVI driver named ‘Arb’ is being used to program an arbitrary waveform generator. If a programmer desires to use the LXI trigger bus line number 2 as the source of a trigger signal, this line of code would be written:

```
Arb.Trigger.Source = “LXI2”;
```

If the programmer instead wishes to use a LAN trigger with the same ID, this line of code would simply be changed to:

```
Arb.Trigger.Source = “LAN2”;
```

The use of names for each trigger source makes it possible to define an API that is both general and extensible, and programmers will find that learning only one type of triggering API is an advantage.

Publish/subscribe interface to all triggering functions

The LXI triggering interface is designed around the common "publish and subscribe" design pattern. This means that each instrument must be told when to transmit any trigger signal, and each instrument must be told whether or not it should respond to any received trigger signal.

This type of architecture requires each device in a system to be configured before any actions are taken. Instruments do not "know" how to respond to a LAN trigger or an LXI Trigger Bus signal unless programmed in advance. Likewise, instruments do not transmit trigger signals unless programmed in advance to do so.

A simple example may illustrate this concept. Imagine that a system of instruments includes a receiver and a digitizer. The receiver is capable of transmitting a trigger signal when its frequency has been changed and its local oscillator has settled. The digitizer is capable of capturing data upon receipt of a trigger.

Using traditional triggering, the receiver would normally have a "trigger out" port and the digitizer would have a "trigger in" port. These ports would be connected by a cable. The receiver would always drive this trigger signal whenever its local oscillator settles. The digitizer would always capture data whenever the trigger signal from the receiver arrives (unless it has been configured to ignore the trigger input).

Using LXI triggering, the receiver would *not* automatically transmit a trigger signal every time its local oscillator settles. Instead, it would have to be configured to transmit a trigger signal, and told which trigger signal to transmit. The trigger might be a LAN packet with an ID of 'LAN1', or it might use the 'LXI1' line on the LXI Trigger Bus, but in any event there is no fixed default trigger – the instrument must be programmed.

Likewise, the digitizer would not automatically respond to a received trigger. Although it would receive a trigger from the receiver,[5] it would not know how to respond to it unless programmed in advance. To enable correct operation, the digitizer would have to be programmed to respond to the trigger by starting a measurement. If the receiver is programmed to transmit a trigger on Trigger Bus line 'LXI1', then the digitizer must be programmer to respond to Trigger Bus line 'LXI1'.

This architecture makes it possible to easily switch between hardware and LAN triggers, and it eliminates the need for a complicated set of standard commands in LAN trigger packets.

[5] Remember that LXI Trigger Bus and UDP-based LAN triggers are received by all instruments on the local subnet. TCP-based LAN triggers utilize a point-to-point communication channel, so the digitizer in this example would not receive a TCP-based LAN trigger unless the TCP link was set up in advance.

Scheduled vs. Event Driven Trigger Operation

The LXI specification supports two distinct models of triggering for instrument control and synchronization: scheduled and event-driven.

Scheduled synchronization makes use of time-based triggers that utilize the IEEE 1588 synchronized clocks. This type of operation is applicable to situations where actions can be scheduled in advance. Many common test activities fall into this category. For instance, a frequency sweep can be executed by programming a transmitter and receiver to change frequencies simultaneously every 10 msec, while a digitizer is programmed to follow each frequency change with a measurement 2 msec later. These actions can all be coordinated through the use of the synchronized clocks and time-based triggers, with no further interactions between the instruments.

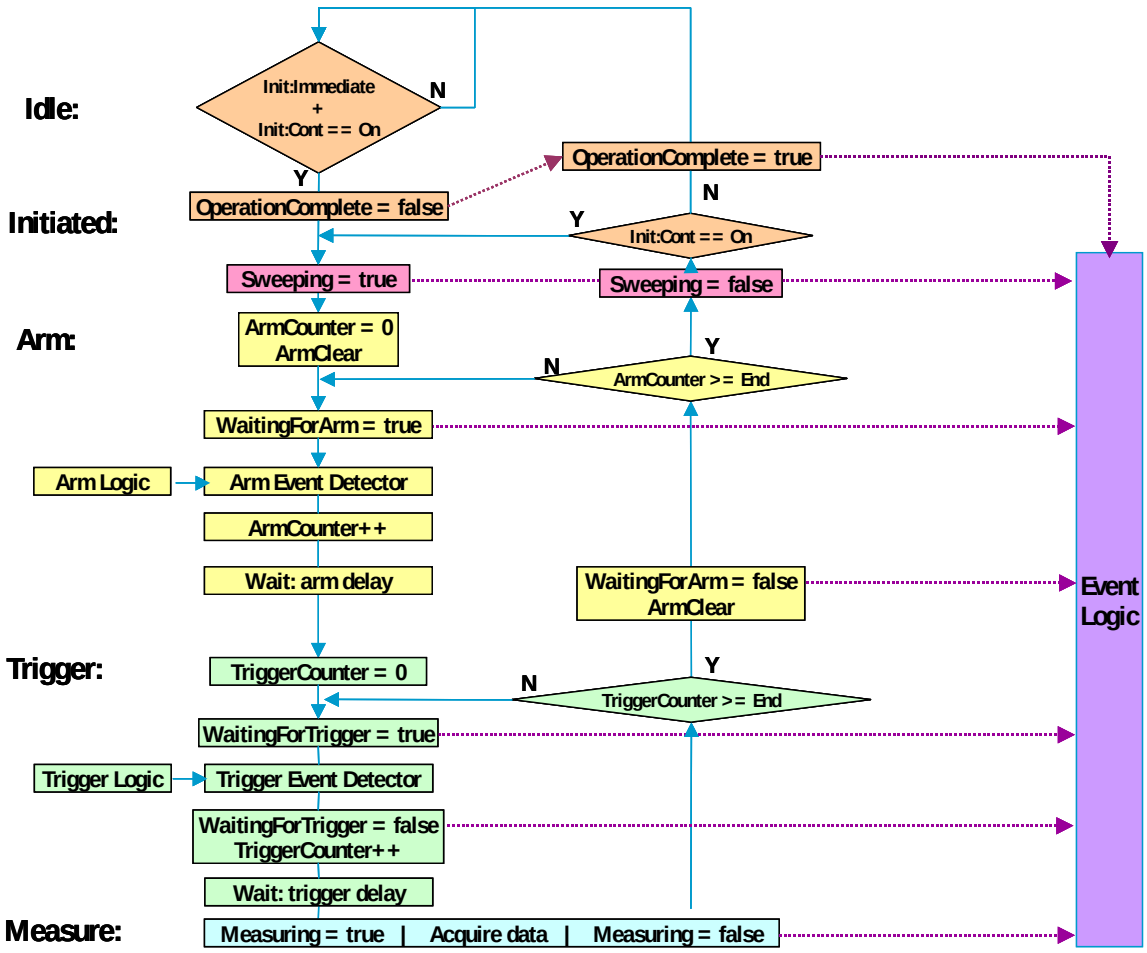
Event-driver triggers use the traditional approach to triggering, wherein some event causes one instrument to transmit a trigger to another. In LXI, these triggers can be transmitted over the LAN, over the LXI Trigger Bus, or over a traditional hardware trigger line. This type of trigger is the only solution for responding to asynchronous events whose timing cannot be predicted or controlled.

It is also possible to combine the two approaches. For instance, a receiver may send an event-driven trigger to a digitizer. If programmed properly, the digitizer may then initialize its own internal time-based trigger. This time-based trigger could be set to "go off" some time later, at which time a measurement could be taken.

Triggering state machine

The LXI specification supports a flexible arm/trigger state machine based on the SCPI triggering model.[6] The diagram below illustrates an implementation of the full arm/trigger state machine. (Refer to the LXI specification for a full description of arm/trigger capabilities.)

[6] The LXI specification does not mandate the implementation of the entire arm/trigger state machine. It is anticipated that early versions of LXI instruments will implement only a subset of the state machine. Eventually more instruments will implement the full arm/trigger model, where appropriate.



This state machine accepts inputs from multiple trigger sources – e.g., LAN triggers 'LAN0' through 'LAN7' and LXI Trigger Bus signals 'LXI0' through 'LXI7'. Depending on the current state, each trigger source is treated as an 'arm' or a 'trigger' signal. It is possible to specify multiple events for both the 'arm' and 'trigger' states. For instance, a device may be programmed to trigger a measurement upon receipt of a signal on 'LXI2', but not until a LAN packet 'LAN5' has arrived *and* an internal time-based trigger has gone off.

Since vendors are not required to implement this full state machine model, every instrument must document the capabilities that are implemented. Refer to vendor documentation for detailed information on any specific LXI device.

Trigger Latency versus Time-based Trigger Jitter

Traditional hardware triggers have always suffered from a certain amount of latency – the time it takes for the trigger signal to travel from the transmitting instrument to the receiving instrument. This latency is obviously a function of the distance that the trigger signal must travel. Trigger latency is not a problem for most applications but some time-critical applications require very low latency, and sometimes test implementers are forced to characterize trigger latency and try to account for it in their measurement results.

LXI Trigger Bus signals and traditional hardware triggers, being simple physical voltages, share the same latency characteristics. LAN-based triggers and time-based triggers behave differently.

LXI time-based triggers have essentially no latency. Common implementations of time-based triggers will involve a countdown timer in hardware and a short and fast internal hardware signal between the timer and the instrument's measurement functions. Although the timer signal may be routed through many gates in an FPGA, the latency should be very small and very repeatable. On the other hand, the IEEE 1588 clock synchronization algorithm will introduce a certain amount of jitter into the timing of the trigger. To a measurement, this will have the same effect as a random amount of latency that changes with each measurement – except that the latency could (in extreme cases where timing is tight and the local instrument's clock is running slightly fast) appear to be negative.

LAN triggers suffer from both latency and jitter. The physical LAN signal must travel between instruments, and the received data must pass through the various layers of the instruments TCP/IP stack before it is received by the application. LAN traffic patterns can delay packet arrival due either to queuing delays in switches or collisions in repeaters. The presence of a common sense of time based on IEEE 1588 allows the system integrator to control the timing of LAN traffic to optimize system performance.

Because of LAN and CPU performance issues, LAN triggers can be expected to have low performance with regard to latency and jitter. However, LAN triggers carry timestamp information with them that can be used to completely eliminate the effects of both jitter and latency in some important cases. The following section offers an explanation.

How to use timestamps

Generally, timestamps are used to indicate one of two things:

1. The time in the past that something happened
2. The time in the future that something is supposed to happen

Properly-designed LXI instruments should be able to capture time stamps for nearly every internal event. A captured time stamp may then be inserted into LAN packets and broadcast to other instruments. The receiving instrument may then use the time stamp to determine when the event actually happened, regardless of LAN latency, jitter, and data throughput issues.

Suppose a digitizer is implemented with a circular data capture buffer (which is common among modern digitizers). When an event occurs elsewhere in the test system, the originating instrument can use its IEEE 1588-synchronized clock to capture a very precise time stamp for the event.[7] This time stamp can be sent in a LAN trigger packet to the

[7] Typically, a hardware latching mechanism will be used to capture the time stamp in memory, from whence it may be later retrieved by software.

digitizer. The digitizer will receive the LAN trigger at some later time, but it can use the time stamp to "look back" in its data capture buffer to save the data that exactly matches, in time, the point at which the event happened.

Time stamps of this nature can also be useful for detecting timing problems in a test system. Upon receipt of a LAN packet, an LXI device can examine the time stamp to tell how long it took for the message to arrive. If the time stamp indicates a time that is too far in the past, it can raise an error.

Finally, time stamps can be used to control the timing of events that should happen in the future. For instance, a digitizer may receive a LAN trigger with a time stamp that is in the future, in which case it simply waits until the specified time arrives and then initiates a measurement (possibly using an LXI time-based trigger).

In general it is a good practice for event generators to distribute the time of the event and for the recipients to be programmed to take actions based on this time. This loose coupling separates the detection and generation of events from their interpretation in the receiving devices. This separation makes it much easier to modify and extend code. Experience has shown that designs that make the event generators specify the action of recipients result in complex, hard-to-maintain software – the time-driven equivalent of spaghetti code.

Instrument Configuration

Although it is not a requirement, the LXI specification recommends that devices should be fully programmable with regards to triggering.

At a minimum, LXI devices must be able to accept configuration commands for 'LAN0' through 'LAN7' trigger packets and for the LXI Trigger Bus lines 'LXI0' through 'LXI7'. This means that every triggerable function in the device should accept any of those trigger sources.

In the best case, it should be possible to download arbitrary executable code into a device and specify that that code should be executed upon receipt of a trigger. The LXI specification currently provides no guidance as to how such code should be downloaded, but the programming API is designed to enable the use of arbitrary code modules once they have been downloaded into the instrument.

LAN Packet Losses

Network-savvy engineers will know that UDP data transmissions, which are the basis of LXI LAN trigger packet transmissions, do not guarantee packet delivery. LAN traffic problems can cause packets to be lost in transmission, and LAN configuration problems can cause packets to arrive more than once. The TCP protocol handles these problems with a very high probability of success, but the UDP protocol makes no attempt. How can this be handled in an LXI test system?

First, it is important to realize that modern network configurations minimize these problems. A test system that uses a small unrouteable subnet (probably the dominant configuration) with a modern LAN switch will virtually never experience packet loss or multiple deliveries. This type of test system will probably never experience problems of this type and need not go to any lengths to account for them.

More exotic test systems that utilize larger networks – company intranets, or even the Internet – will have to take some special care to ensure that data is not lost. Users can minimize these problems by using some care in configuring their networks, and the LXI specification itself provides some support for more complex solutions.

Using UDP on the Internet

Both the IEEE 1588 specification and the LXI specification rely on UDP multicasting. On large networks like the Internet, multicast UDP packets are commonly blocked by network routers. This prevents both clock synchronization and trigger packet reception.

If the network path between all devices is under the control of the system integrator, then routers can be configured so that they do not block UDP multicast packets. Some care should be taken with this solution, as UDP multicasts by other applications can create a significant network load.

If routers cannot be configured, then two steps can be taken to enable the use of LXI instruments on the network:

1. Use an alternative means of clock synchronization. For instance, a distributed test system may be able to use GPS receivers to synchronize clocks. Compared to IEEE 1588, this may degrade clock synchronization accuracy but the result may still be acceptable, depending on the application. In addition, any instruments that exist in a local cluster can use IEEE 1588 to synchronize their clocks, and GPS can be used to synchronize each remote cluster.[8]
2. Use TCP, rather than UDP, for LAN triggers. This entails some extra system configuration because TCP uses a point-to-point communications channel so TCP sessions must be individually set up. But TCP communications will be transmitted across the network.

Also, some firewalls may be configured to block data transmissions to unknown TCP ports. These firewalls must be configured to allow traffic on port 5044.

Handling Multiple Packet Receptions

[8] The use of GPS typically requires averaging the received time for several minutes to achieve synchronization accuracy of better than 100 ns. For precise synchronization of 'islands of time', GPS filtered using high quality clocks can achieve nanosecond levels of synchronization – far better than achievable via other means normally employed in test and measurement systems for long distance synchronization.

If a network's topology is sufficiently complicated so that there is more than one possible path from one device to another, UDP packets may arrive at the receiver more than once. The LXI specification acknowledges this situation by requiring that all LXI devices must be able to detect multiple packet receptions and ignore them.

Support for this functionality is built in to the LXI LAN packet format. The packet includes a field that contains a sequence number. Each instrument maintains its own sequence, and increments the value every time a packet is transmitted. Receiving devices can use this value, along with the timestamp, to detect duplicate packets.

Minimizing Packet Losses

In cases where LAN packet losses are expected, or where even unlikely losses would be catastrophic, some extra care must be taken to ensure that losses do not occur. The LXI specification supports several techniques that can minimize packet losses, although it does not require the use of any of them.

The easiest solution to packet losses is to use point-to-point TCP transmissions rather than UDP multicasts. This results in longer trigger latencies and somewhat more complicated software configurations but drives the probability of data loss to an insignificant value.[9]

If the lower latency of UDP is required, the LXI specification supports three techniques that can help eliminate packet losses:

1. Transmit packets more than once

One of the flags in the LXI LAN packet is used to indicate whether or not the packet is a retransmission of another packet. The LXI specification requires all devices to ignore packets that have this bit set if they have already received the original packet. This allows test implementers to routinely transmit UDP packets more than once. Upon receipt, devices can examine the packet's time stamp to determine whether a delayed packet will have an effect on the measurement.

2. Implement a handshaking protocol

Another of the flags in the LXI LAN packet indicates whether or not the packet is part of a handshaking protocol. LXI devices that do not implement any sort of handshaking are required to ignore these packets. This allows the implementation of a TCP-like handshaking protocol without the added latency of TCP.

[9] There is no LAN protocol that guarantees data delivery with 100% confidence. The TCP protocol is widely used and highly reliable, but even TCP cannot absolutely guarantee data delivery.

3. Utilize a "time-out" technique

If sufficiently programmable, instruments may be able to execute software that expects to receive a LAN trigger within a certain time frame. If the trigger is not receiver, the device can send an error message or request retransmission.

Because these techniques are not required by the LXI specification, their use is not guaranteed to be consistent across various vendors' instruments, nor are they guaranteed to be implemented at all. The LXI specification supplies only minimal support to allow vendors to enhance reliability for some applications without creating interoperability problems.

Be Sure of Clock Synchronization

When LXI Class A and B instruments are powered on, it can take up to two minutes for the IEEE 1588 clock synchronization algorithms to complete their work. Obviously, it would be unwise to start a test program running before that time.

The LXI programming API includes a method for determining whether or not each device's clock is properly synchronized. It is important to check the clock's state before beginning any test programs that rely on it.

Tradeoffs

The LXI specification supplies test system programmers with several new triggering technologies. Any of these triggering methods may be preferred for a given application. On the other hand, many applications will not be sensitive to the triggering technique in use and the designer may choose the one that best suits his or her preferences.

The tradeoffs between the various triggering techniques are not difficult to understand but may not be immediately obvious. The following table attempts to address some of the more common situations, and gives some insight into cases where one triggering technique may be more or less appropriate.

Triggering method	Use if…	Don't use if…
LAN triggers	• Instruments are far apart • Trigger cables are not desired • The time stamp that is contained with the trigger is useful (e.g., circular data capture buffer)	• Very low trigger latency is needed • Very fast trigger rates are needed
LXI trigger bus	• Low latency is needed and instruments are not too far apart • Low jitter is needed	• More than eight triggers are needed • Instruments are more than about 25m apart • Propagation delays cannot be tolerated
Time-based triggers	• Trigger latency cannot be tolerated • Instruments are far apart, and low-latency triggering is needed	• Trigger jitter cannot be tolerated
Vendor-specific triggers	• Special-purpose electrical signals or connectors are used • External triggers are used that cannot be	• LXI triggering capabilities are sufficient

	routed onto the LXI trigger bus • Interfacing with non-LXI instruments	

Comparisons with other systems

Will the performance of Ethernet-based instruments match that of GPIB, MMS, or VXI-based systems? There is no single answer to this question. The complexity of bus operations, combined with the plethora of available custom solutions that are designed to make each bus faster for one application or another, makes simple comparisons impossible. Further, different applications have different requirements – a bus that works well for one application may fail miserably with another. Still, it is possible to distill some general characteristics of the available instrumentation busses. Refer to the table below for a comparison.

	Latency	**Maximum Throughput**
100 Mbit/sec Ethernet	~1 msec and up	10 Mbytes/sec
1000 Mbit/sec Ethernet	~1 msec and up	100 Mbytes/sec
GPIB	~100 μsec	~1.5 Mbytes/sec
MMS	10 μsec max, 2 μsec typical	3.1 Mbytes/sec
VXI	Microseconds to milliseconds	80 Mbytes/sec

This table lists the approximate theoretical maximum data transfer rates. *The throughput obtained in actual use is always lower.* The latencies reported in this table are likewise approximate, and (particularly in the case of Ethernet) can increase greatly with distance between instruments and traffic on the network. Nevertheless, it can be seen that the use of Ethernet for instrument control should have useful performance overall, especially when combined with the advanced capabilities of LXI triggering, which can often be used to eliminate the effects of LAN latency.

Test system configuration topics

When configuring networks, LXI system integrators can take a few steps that will help to optimize clock synchronization and LAN trigger latencies.

Network Switch versus Hub

The IEEE 1588 clock synchronization algorithm will exhibit best performance if network switching fluctuations are kept as low as possible. Absent some sort of traffic control the lowest timing fluctuations are accomplished using a hub rather than a switch. Unfortunately, the advantages of using a switch are considerable. A hub is a half-duplex device while a switch is full-duplex. Switches, unlike hubs, typically include store-and-forward data buffers that prevent packet losses and data collisions.

In general, test systems should use high-quality network switches rather than hubs. Very simple test systems that do not experience high LAN traffic may find that hubs work well enough, but this is not expected to be the dominant case. For general use and for any application that requires high synchronization accuracy independent of traffic loads, a boundary clock should be used.

Boundary Clock

IEEE 1588 clock synchronization requires one network device to be the master clock, to which all other clocks are slaved. Because the lowest-latency path in any network is likely to be the path between the switch and each network device, the best place to put the IEEE 1588 master clock is in the switch itself. In the IEEE 1588 specification such a switch is referred to as a boundary clock.

Use of a boundary clock can greatly improve the timing accuracy of IEEE 1588 clock synchronization. The use of a boundary clock, while not mandatory, is highly recommended.

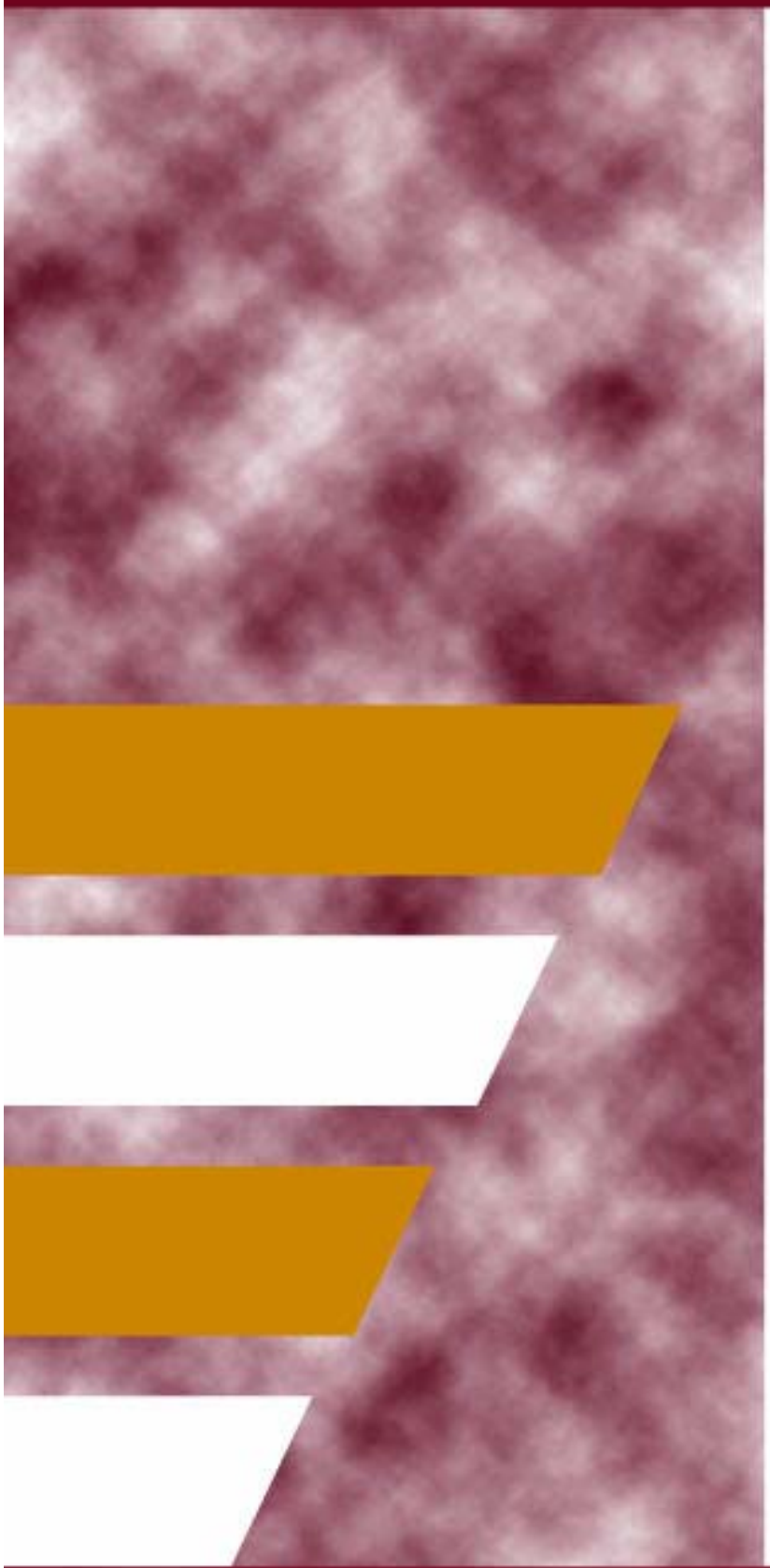
Achievable synchronization accuracy

The timing accuracy achieved using IEEE 1588 depends on the design of the individual instrument clocks and the design of the network. In lightly loaded networks using ordinary switches, experience has shown that accuracies on the order of 100-250 ns (standard deviation) are achievable in compact topologies typically found in T&M systems. If the ordinary switches are replaced with boundary clocks the jitter standard deviation (with currently available boundary clocks) is about 20 ns. The time offset (as opposed to the jitter) in such systems will be a few nanoseconds in properly designed devices. Laboratory results indicate that timing jitter extending to the sub-nanosecond range is achievable with appropriate implementations of boundary and instrument clocks. At this level the removal of residual offset will require very careful design and calibration of all devices and possibly of the network cabling itself.

Setting Up the Network.

A summary of “best practices” for setting up an LXI-compatible network is below.

1. Use a switch rather than a hub
2. Use a boundary clock
3. Configure all firewalls so that port 5044 is not blocked
4. Configure all routers so that they allow UDP multicasts



Aeroflex, Inc.
Agilent Technologies
Keithley Instruments, Inc.
Measurement Computing Corp.
Pickering Interfaces Ltd
Racal Instruments
Rohde & Schwarz GmbH & Co KG
VXI Technology, Inc.

Anritsu
Bruel & Kjaer S & V

ADLINK Technology, Inc.
Analogic Sky
BAE Systems
California Instruments
Complete Networks, Inc.
Elgar Electronics
Fluke Corporation
GOEPEL electronic GmbH
IOTech, Inc.
Lambda Americas
Pacific Power Source, Inc.
Phase Matrix, Inc.
Teradyne
The Math Works
Universal Switching Corporation
Venture Design Services, Inc.
ZTEC Instruments

Acqiris
AMREL
Data Translation
Department of Defense (DoD)
Electronic Systems Innovation
Kepco, Inc.
Pacific MindWorks
Shaanxi Hitech Electronic Co., Ltd
Symtx, Inc.
Yokogawa Electric Corporation