Agilent Laboratories
3500 Deer Creek Rd.
Palo Alto, CA 94303

Date: 18 January 2005

The Application of IEEE 1588 to Test and Measurement Systems

John C. Eidson

Introduction

IEEE 1588 is a protocol used to synchronize real-time clocks in modules of a networked distributed system. The protocol was designed specifically to aid in the coordination of activities and the correlation of data in distributed systems typically found in test and measurement, industrial automation and similar environments.

This note discusses some basic ways of using IEEE 1588 in the context of test and measurement (T&M) and the additional functions that must be included in an instrument or instrument module to capitalize on IEEE 1588. System and component design issues that influence the performance of a T&M system are also discussed. It is assumed that readers are generally familiar with how IEEE 1588 works to synchronize clocks and except for details important to T&M applications the IEEE 1588 protocol itself will not be discussed. Readers wishing more information on IEEE 1588 or its application in areas other than T&M are encouraged to visit the IEEE 1588 web site at http://ieee1588.nist.gov.

Although IEEE 1588 is not limited to Ethernet networks all of the discussion in this note will assume that Ethernet is the network of choice. In particular the discussion will focus on the context of instrumentation being developed by members of the recently announced LXI consortium, see http://www.lxistandard.org/.

This note is organized into three main sections. The first section discusses the driving forces that make IEEE 1588 attractive and how this has manifested in other industries. The second section discusses general system level considerations surrounding the implementation of systems using IEEE 1588. The third section presents several examples of how IEEE 1588 would actually be used in a T&M system and what sort of changes need to be made in the internal architecture of instruments to capitalize on IEEE 1588. In the third section specific examples of actual or proposed applications from T&M and other industries are provided. All of these examples are based on papers and discussion presented at one or both of the two conferences held on IEEE 1588 (see the IEEE 1588 web site for copies of the conference proceedings).

These discussions draw on the experience of many people that have been working with this technology including:

- Engineers at Agilent Technologies
- Discussions in the IEEE 1588 multi-industry task forces and conferences

- Engineers in the industrial automation and power generation industries.

Background

In the general field of measurement and control several trends have been apparent over the last ten years:

1. Increasing complexity of systems and more exacting system coordination requirements,
2. Increased use of distributed architectures,
3. The use of network communication technology, and
4. Cost constraints.

In almost all fields requirements for system level performance have become more difficult to achieve. Cost of testing, tighter control for increased quality, and more complex and higher performance devices to be tested or built all combine to raise requirements. Some of this has resulted in the need for more capable, more precise, or faster devices such as instruments, controllers, sensors and actuators. However the most profound effect is in the system level control of these activities and the management of the associated data.

The move to distributed architectures has been seen as a way to increase performance and simplify the task of application controllers by partitioning applications into smaller, more manageable functions for which most of the control, data acquisition and data processing can be done locally. However in doing so the requirements on communication and precise coordination among these distributed components and with system level controllers have increased. This in turn has led to the use of networked technologies over bus or dedicated communication links due to the increased flexibility of networks. The final trend is pressure to reduce cost, both component and life cycle.

These trends, which had major consequences in industrial automation particularly in high-speed motion control, are just now beginning to manifest themselves in the T&M community with a time lag of perhaps 5 years. The most obvious example is the increasing role of the PC and the use of Ethernet networks, primarily driven by cost. The major obstacle industrial automation had to overcome to successfully adopt Ethernet was the perceived loss of timing control in comparison with proprietary bus or direct IO solutions. IEEE 1588 is now seen as the way to achieve precise timing in an increasingly Ethernet dominated control and data acquisition strategy in industrial automation.

Although industrial automation initially appears quite different than T&M a close examination of the system level control and timing issues shows that they are quite similar. They are definitely similar in architecture and increasingly not that different in the actual timing requirements. For example the motion control segment of industrial automation is seeking clock synchronization to roughly the 100 ns level! There are now several major field installations in the power industry that make extensive use of IEEE 1588 and the major automation vendors are showing preliminary IEEE 1588 based products to customers and in trade shows.

The LXI consortium has made a bet that LAN, specifically Ethernet, along with the increasingly modular and distributed architecture that Ethernet enables will be a major win in T&M. The consortium, like industrial automation, is looking to IEEE 1588 as a way to recover the precise timing that would otherwise be lost in the move to Ethernet.

General system considerations

IEEE 1588 synchronizes real-time clocks in modules of a distributed T&M system. IEEE 1588 creates a common sense of time throughout the system, in effect offering a time-service that may be used in a variety of ways discussed later. From a system view the following characteristics of this common sense of time are important:

1. Precision,
2. Accuracy,
3. Epoch,
4. Dependency on network topology,
5. Resources needed, and
6. Dynamic behavior.

Precision:

There are two facets to the precision of the common sense of time: the temporal precision within an individual module, and the temporal precision of a collection of modules.

The precision within an individual module is a measure of the scatter of time measurements made by the module relative to the time base of the module. Typically the real-time clock will be implemented in hardware as a counter driven by a local oscillator. The precision of the local time base is a function of the resolution of the counter and the noise characteristics of the oscillator. Since both applications and the IEEE 1588 protocol itself involve software, data types and computational considerations may also degrade precision.

The temporal precision of a group of modules depends on the temporal precision of each of the individual modules as well as on the precision that IEEE 1588 is able to achieve in synchronizing the clocks within the system. This is illustrated in Figure 1.

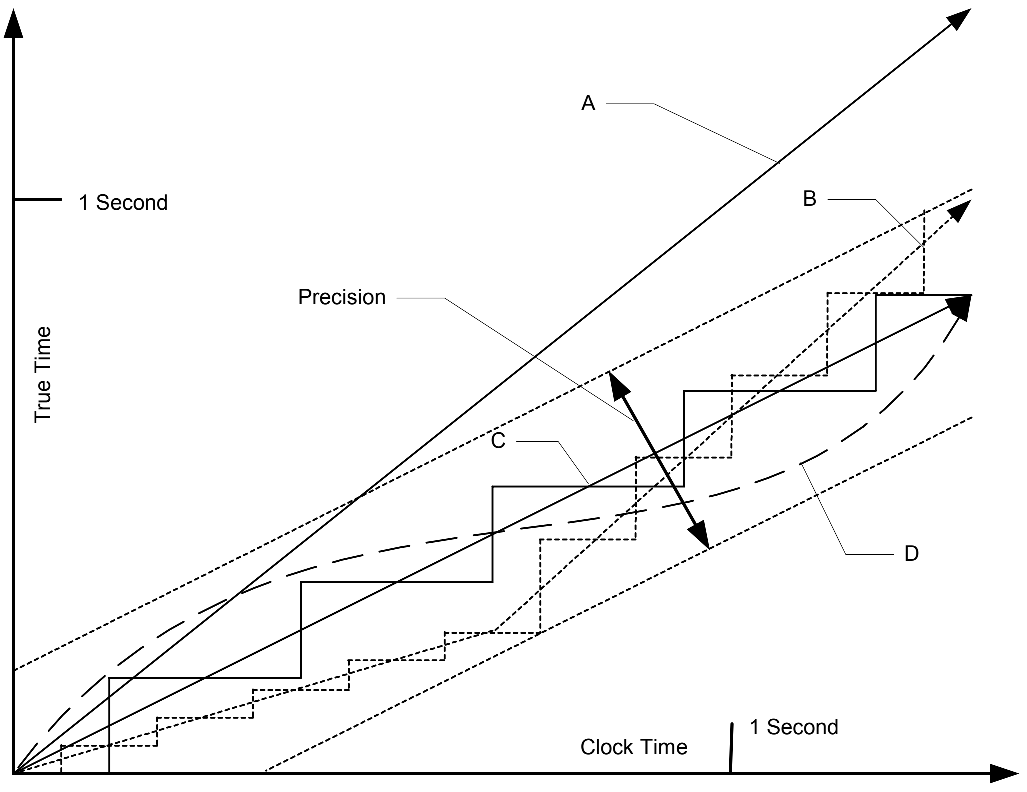

Figure 1: Time base precision in a T&M system

Shown is the temporal scatter for each of three clocks B, C, and D relative to true time A. Each of the clocks exhibits different resolution and the jitter. The jitter arises from random variations in the frequency of the oscillator that causes it to run fast or slow by a small amount on a random basis. For example, clock D has very fine resolution (not visible on this drawing) but exhibits jitter causing the wavy curve above and below its mean value. Clock C has very coarse resolution but low jitter about the mean value. Clock B has better resolution but a significant jitter. The relative positions of the scatter of the individual clocks will shift relative to each other due to the normal operation of IEEE 1588, which operates to keep the mean times of all slaves identical to that of the master. The extent of this scatter defines the system wide scatter or precision of the time base. Thus clocks B, C, and D have a system wide precision indicated by the two dotted lines of Figure 1. The net result is that when comparing the times measured on two different clocks the system wide precision limits both arithmetic and relational computations on time just as measurement precision limits computations for any other variable such as voltage.

Accuracy:

The accuracy of the common sense of time is a measure of the agreement between the second as realized in the system time base and the international definition of the second. For example in Figure 1 clocks B, C, and D all have the same time base since they are synchronized but a second in this time base is clearly shorter than a second in the true time base A. Since IEEE 1588 is a master slave synchronization protocol, the accuracy is determined by the accuracy of the time base of the grandmaster clock that is at the root of the clock hierarchy established by the protocol. All clocks tracing their time base to the grandmaster will have the same accuracy.

Epoch:
The epoch of the time base is the origin of the time as measured in the system, that is, time zero. Like accuracy, the epoch of the time base is determined by the grandmaster. In many T&M applications only relative time within the system is important and no correspondence with civil time is needed. Other systems require the time base to be traceable to UTC. This is actually quite easy to do and provides many advantages in keeping useful engineering data that can be correlated with data from other systems or with historical data.

The most straightforward way to tie the IEEE 1588 time base to UTC is to synchronize the grandmaster clock to a recognized UTC traceable source such as the GPS system or an NTP timeserver. This must be done with care to retain the desired precision of the resulting time base. NTP precision is on the order of milliseconds so only very long-term averages of NTP should be used in synchronizing the grandmaster clock. GPS precision is much better, on the order of 50 ns, but still requires averaging for the highest accuracy. Both NTP and GPS provide the necessary leap second information required for the translation between UTC and the IEEE 1588 time base.

It is interesting to note that the major vendors in both the power and industrial automation industries that are implementing IEEE 1588 all have reported success in linking the IEEE 1588 time base to UTC by means of GPS. Many of their customers have regulatory requirements to have UTC traceable timestamps for significant events in their systems.

Dependency on network topology
IEEE 1588 is a master slave synchronization protocol. Each slave adjusts its time base to agree with its master. The relative accuracy between the master and slave will vary with time due to the nature of the servo loop in the slave, the fluctuations in the oscillators of the two clocks, and most critically fluctuations introduced by network switches. In modern Ethernet systems all communication between devices goes through a switch or router. So-called repeaters or hubs are now rarely used. Switches introduce fluctuations that uncorrected will limit both precision and accuracy to between 100 and 400 ns depending on network traffic patterns. The IEEE 1588 solution is to modify these switches to include an IEEE 1588 clock serving as a transfer standard. This effectively eliminates the switch and network traffic patterns as a concern. These special switches are called boundary clocks in the IEEE 1588 standard. Fortunately these devices for Ethernet are beginning to appear on the market.

A complex IEEE 1588 system will have a hierarchy of IEEE 1588 enabled switches resulting in a cascading of servo loops with resulting degradation in the time base.

For example in Figure 2 the inaccuracy between clocks A and G in the linear topology, A, will be greater than in the branching topology, B since in the linear case there are seven cascaded control loops involved while in the tree topology only three such loops.

It is also clear that the placement of clocks with different precision is important. For example if in the linear topology of Figure 2 any of the clocks between clocks A and G had significantly poorer precision than the others then the precision of G relative to A could at best be that of the poor precision clock. In the tree topology it is possible to isolate poor precision clocks to their own branch of the tree thereby preserving the precision of the rest of the tree. It is clearly important that the branch points, IEEE 1588 enabled switches, have a precision significantly higher than any of the ordinary clocks slaved to them. It is also clear that the grandmaster clock, clock A in both linear and tree topologies, should have the best precision and accuracy of any of the clocks since it determines the quality of the time base for all. Since IEEE 1588 automatically configures the master slave hierarchy it is important for system designers to understand how the protocol operates so that they can correctly position clocks of varying precision in the physical network layout.

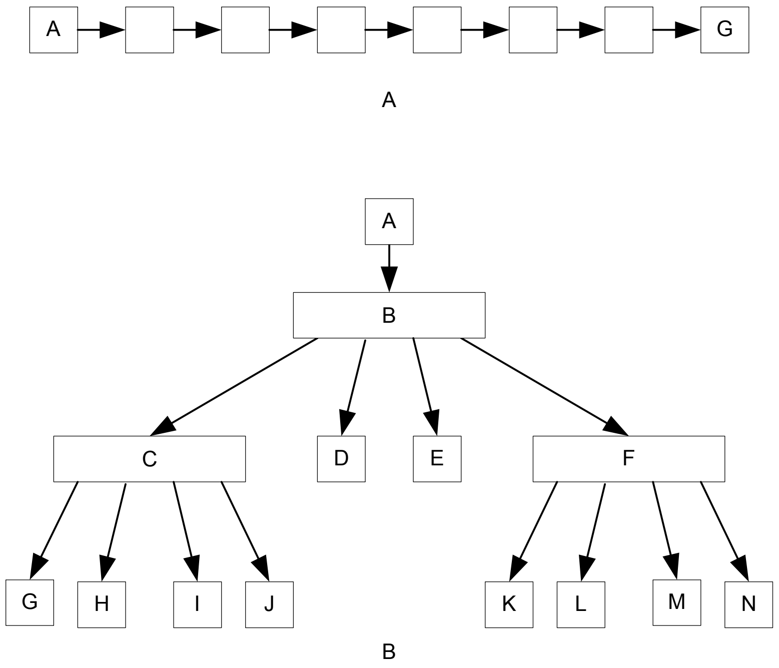

Figure 2: Clock topologies

In the previous discussion of the tree topology of Figure 2 it was assumed that all of the branch points, B, C, and F are IEEE 1588 enabled switches. These special switches serve as a time transfer point between two segments of a network. For properly designed clocks and IEEE 1588 switches the degradation in the time base over a single link, for example C to H, is independent of the network traffic provided the capacity of the link and of each node is adequate to ensure that IEEE 1588 synchronization messages to not experience what amounts to a denial of service attack. However if one of the IEEE 1588 switches, say C, is replaced by an ordinary Ethernet switch, then significant traffic dependent queuing in the switch can occur which will introduce additional jitter into the link between a master and slave, for example B and G. This will result in a degradation of the slave's time base compared to the case where clock C is an IEEE 1588 enabled switch.

It is imperative that the precision of each clock and IEEE 1588 switch and the overall system precision requirements be carefully considered when designing the network topology for a T&M system. For the most complex T&M systems the topology will look like Figure 2B. It is much more likely that T&M topology will appear as a single IEEE 1588 enabled switch with a few connected devices as illustrated in Figure 3.

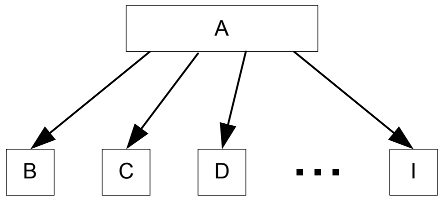

Figure 3: Typical T&M topology

By contrast systems in industrial automation applications are quite complex often involving 1000 or more distributed sensors or actuators and multiple levels of control. Both topologies of Figure 2 are used. The use in the linear topology has lead to the development of additional switching technology incorporating IEEE 1588. These linear topology switch devices, termed transparent clocks, were demonstrated by one vendor at the plug-fest held in conjunction with the 2004 IEEE 1588 conference. The specifications for these devices are likely to be added to the standard in the next revision.

Resources needed
IEEE 1588 is a very lightweight protocol. In the default case there will be only one or two packets per second on a link between an IEEE 1588 enabled switch and an attached slave clock. The memory and computational resources required are also low. However it is important for both component and system designers to ensure that the resources provided are adequate to meet the demands of the overall application without starving the IEEE 1588 protocol. The main concern is large data transfers between devices. While with properly designed clocks an occasional missed IEEE 1588 message is not significant, it is important that the system design does not cause multiple successive misses. If the network or device capabilities are not adequate and if IEEE 1588 traffic does not have sufficient priority then degradation of the time base may occur.

Dynamic behavior
In an IEEE 1588 system each slave clock synchronizes to its master typically with a simple control algorithm such as a proportional integral (PI) loop. The purpose of the control algorithm is to bring the clocks into synchrony and to average out or otherwise eliminate any noise that would otherwise disturb the time base. In general the reduction of noise requires longer averaging as the noise level increases. Experience has shown that with inexpensive oscillators and typical network conditions that an overall precision on the order of 50 to 100 nanoseconds standard deviation is quite easy to achieve. To achieve the higher precisions needed for some T&M applications components will typically have more stable oscillators and slaves will use longer averaging times. This will also produce longer lasting startup or other transients unless components are selected

that implement more sophisticated servo designs that adjust their time constants dynamically.

Application considerations

For this discussion it is assumed that a common sense of time with adequate precision and accuracy has been established in a T&M system. This section discusses how this common sense of time can be used to improve T&M applications and the needed application support features required in the instrumentation.

T&M applications seek to measure or control the physical world. In all but single instrument systems this requires that the overall application correctly manage events and that actions such as sampling are synchronized. It is in this area of event management and synchronization that the commons sense of time is most useful.

Event management

Events are occurrences in the system that requires some action to be taken. Typically these events are generated in a control program or in an instrument as a result of some measurement. Conventional components handle events as shown in Figure 4.

If the event information is to be conveyed over the network the message originates and terminates in application code in the sending and receiving devices. In the process it traverses the OS, MAC and PHY of both devices accruing considerable latency and jitter in the process, path A. The receiving device processes this event information to cause some action such as sampling an A/D. This sampling introduces more jitter both from the application code as well as from path C. The message-to-message latency and jitter on these paths makes the precise temporal control of actions in multiple devices difficult.

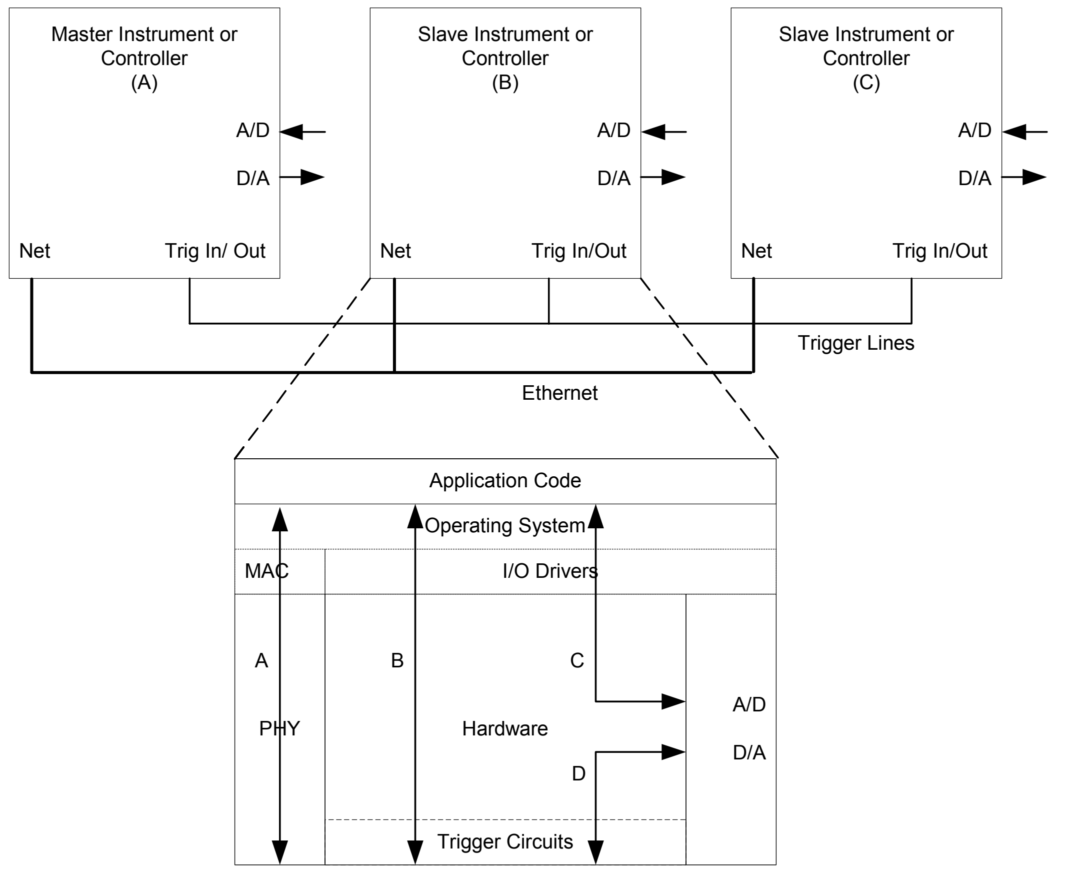

Figure 4: Conventional triggering

To circumvent this problem many instruments provide special hardware trigger circuits as shown. In this case there is a more direct coupling between an A/D and the generation of a hardwired, as opposed to a message, trigger. A similar situation occurs on the receiving end where the hardwired trigger is interpreted in hardware before being applied to the A/D or D/A. Careful calibration can produce very accurate triggering although it is still difficult to control the triggering of multiple devices to high precision. The precise timing of this triggering is also invisible to the application code, which makes the coordination between code and triggers more difficult.

In a time based system the internal architecture of a device appears as shown in Figure 5. In this case signals with precise timing requirements are generated in the blocks labeled time triggers. A time trigger accepts a timestamp generally from the application code via path B. The time trigger continuously compares this timestamp with the time in the local IEEE 1588 clock. When the clock time is greater than or equal to the timestamp the time trigger generates the trigger signal that may be applied directly to an A/D or D/A via path D or used to generate an external signal, for example a hardwired trigger, via path E. In like manner the timestamp register accepts a signal, for example from the A/D via path C or an external signal via path F, and captures a snapshot of the IEEE 1588 clock thus providing a precise timestamp of the event to the application code via path B.

The time triggers and timestamp registers allow management of event signals to the precision of the clock synchronization. In addition these times are completely visible to the application code. The presence of these time-based registers provides an additional degree of freedom to the application. They may be used independently or in conjunction with conventional hardwired triggers.

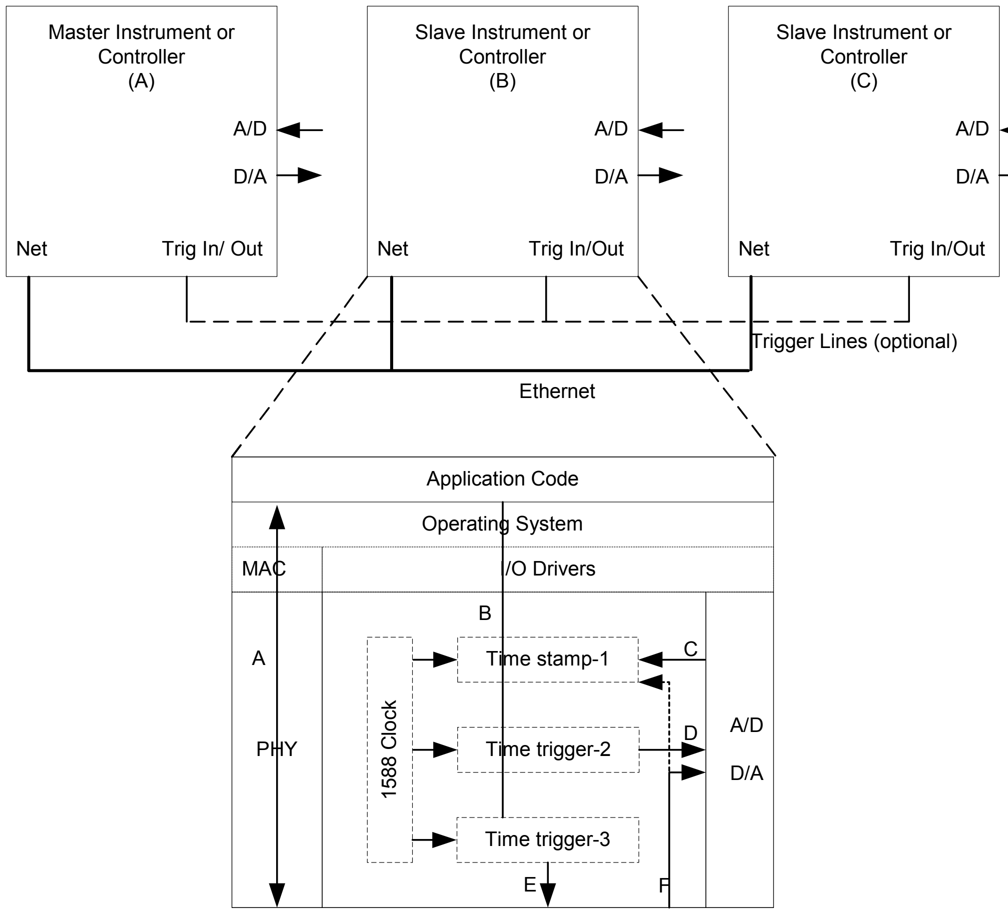

Figure 5: Time based triggering

As noted earlier precisions of between 50 to 100 nanoseconds standard deviation are quite easy to achieve even with inexpensive components. With a bit more care the precision can be improved to 10 nanoseconds. Results in our laboratories to date have reduced this value to about 4 nanoseconds between two directly connected clocks.

Synchronization

It is quite common in T&M systems to synchronize sampling rates in multiple devices to allow more efficient management of the application and analysis of the data. This is implemented in conventional instruments as illustrated in Figure 6. In this case the sampling signals controlling the A/D and D/A converters or other circuits are generated in the sample clock generation block in the hardware. This block may be configurable, for example rate adjustment, by the application code via path B. Typically this block is driven by the local oscillator of the device. For critical synchronization applications provision is often made to drive the sample clock generator from an external 10 MHz signal using phase locked loops as shown. This is necessary if the sampling frequencies are to agree to better than perhaps 0.01% since oscillators of greater accuracy can be quite expensive. In the absence of an external signal and the PLL the sampling rates can differ due to the inaccuracy of the different oscillators.

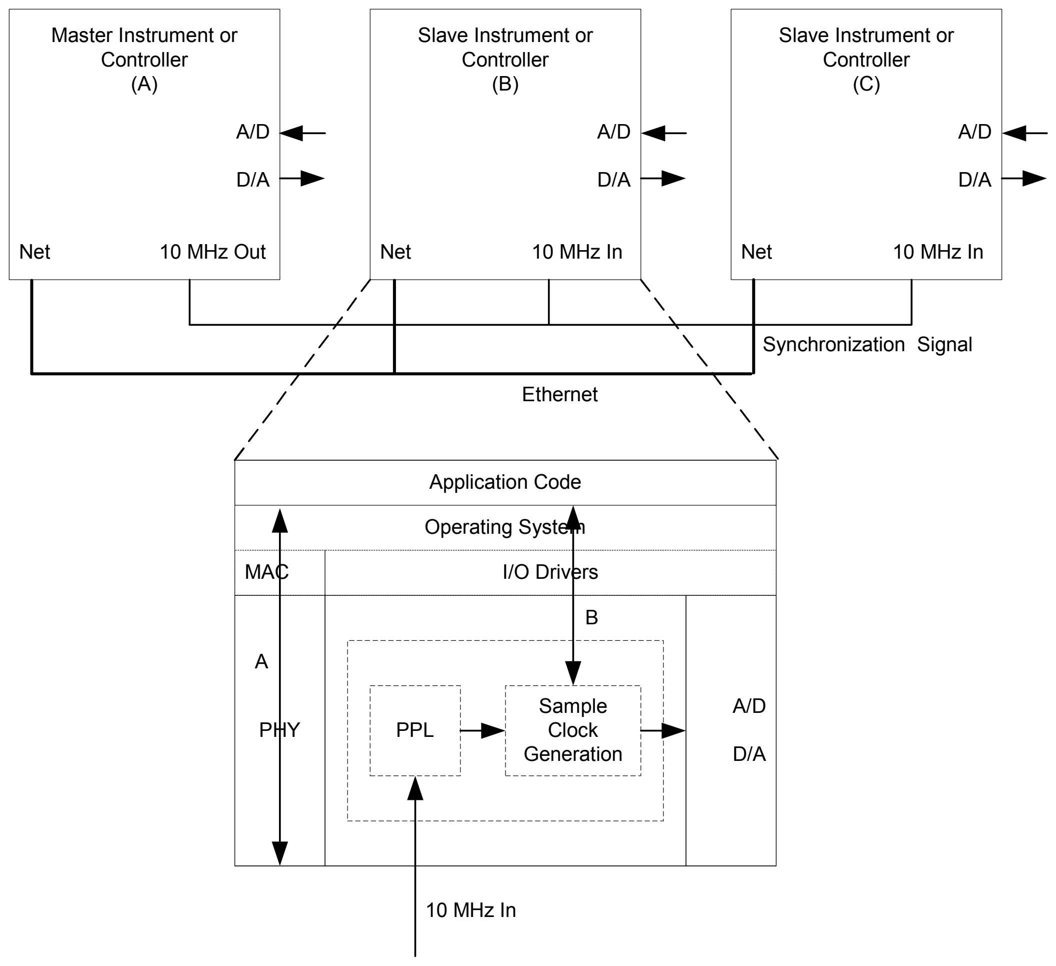

Figure 6: Conventional sampling synchronization

In a time based device the sample clock generation block is driven from the local IEEE 1588 clock as shown in Figure 7. Thus the synchronization of sampling between multiple devices will depend on the precision of the synchronization of the IEEE 1588 clocks in these devices. As noted earlier it is clear that precision on the order of a few nanoseconds will be achievable in practical devices. This means that the epochs of the sampling signals will agree to this precision. The frequency agreement under these conditions will be at least a part per billion or better.

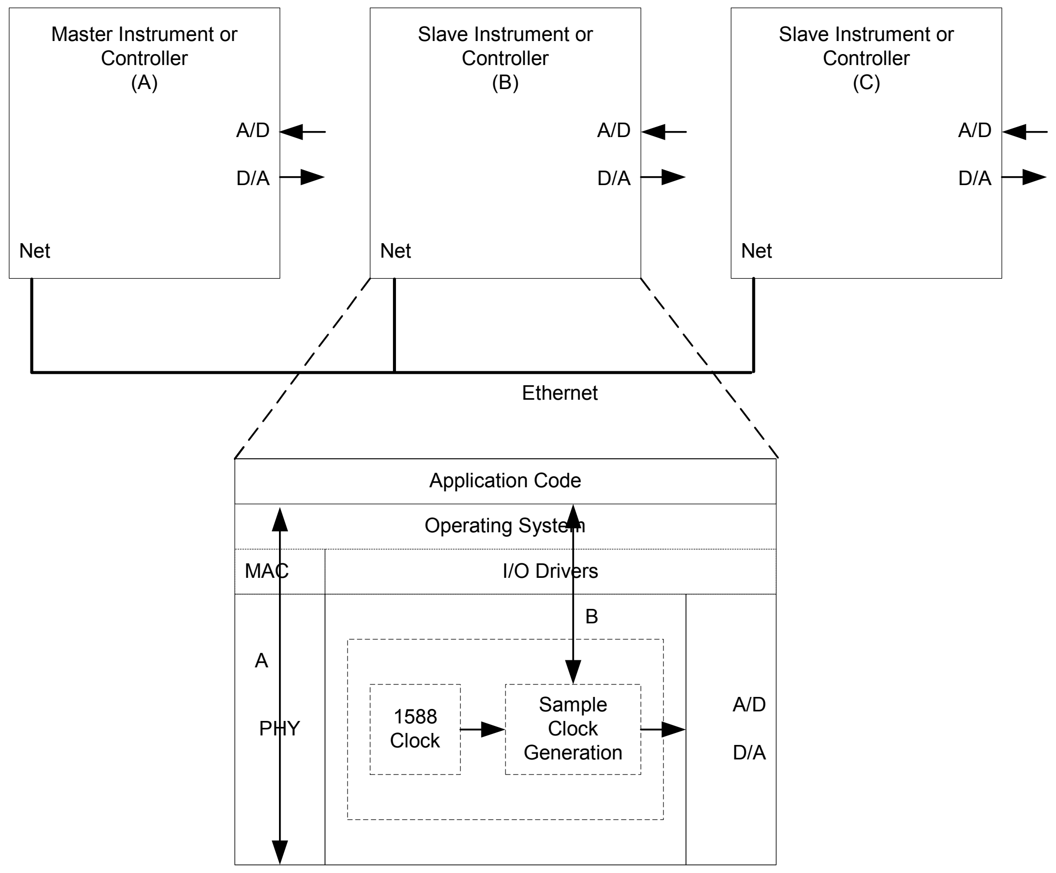

Figure 7: Time based sampling synchronization

System application coordination

There are three fundamental types of application coordination enabled by a common sense of time:

1. Source timestamping of data
2. Random event triggering
3. Scheduled event triggering

Each of these is discussed in greater detail in the following sections.

Source timestamping of data

The sample clock generator of Figure 7 and the timestamp registers of Figure 5 both enable the generation of a timestamp for each sampled or generated datum. These capabilities are summarized in Figure 8. The timestamp block captures a timestamp whenever a sample is made for both measuring and actuating (source) devices. The actual triggering of this sample can be by any of the means discussed. It is also possible to capture a timestamp for the trigger signal or any other significant event associated with the sampling process or the application.

These event timestamps allow post acquisition correlation of data and events based on the timestamps. This relieves the application, typically in the controller, from having to maintain the order of events and data. In non time-based systems ordering is usually based on the handling of interrupts, order of polling or system calls, and processor execution times. Such code is relatively difficult to write and maintain if precise timing information is required.

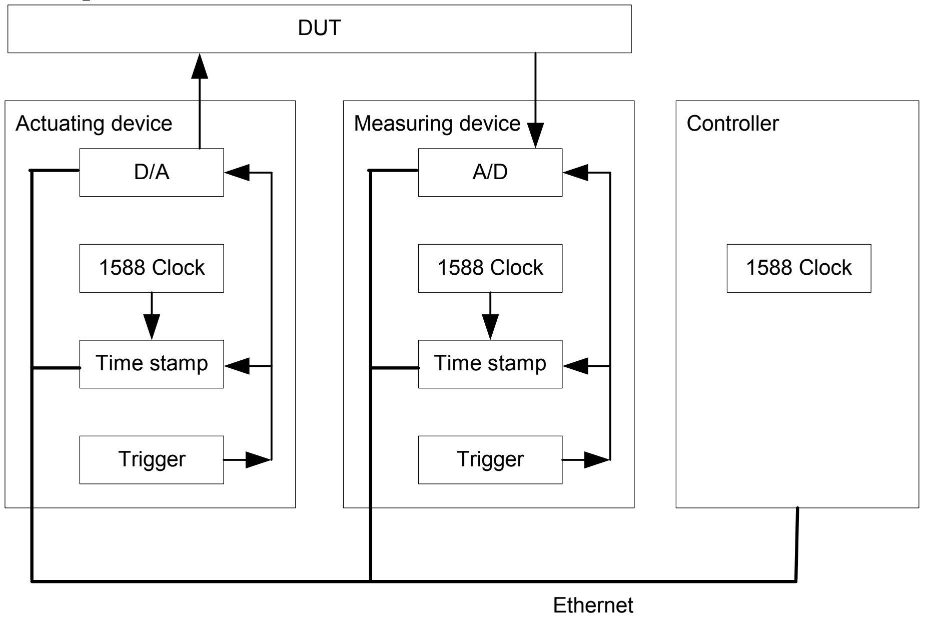

Figure 8: Timestamping of data

The ability to timestamp these events where they occur and then deliver the data as value-timestamp pairs ensures that the needed correlations can be made. The temporal precision of these correlations is limited by the precision of the synchronization of the clocks in the devices producing the data. Timestamp value pairs can be communicated as pairs. In the case where the data is a time-series it is possible to communicate only the starting timestamp and the interval along with the ordered list of values. The obvious disadvantage of time-series encoding is that missing values are more difficult to identify.

The timestamping of data at the point of acquisition has proven to be the low hanging fruit of IEEE 1588 technology. As noted this enables very precise ordering and correlation of events and data in a distributed system with essentially no real-time code needed in the controller. This has been used to great advantage by General Electric in developing a new, Ethernet-based control architecture for the monitoring and control of large gas turbine generators. This system has roughly 1000 sensors and actuators monitoring various signals and sensors on the turbine. The data is timestamped at the sensor based on a local IEEE 1588 clock and sent to a central data repository where it is used for monitoring and for fault trace analysis in the event of turbine malfunction. There are several projects under way at a number of companies to use this same technique in a variety of military test and monitoring applications.

Random event triggering

It is quite common in T&M systems to detect an event, for example a threshold crossing, in one device and to use this event to trigger actions in other devices. This random event triggering is illustrated in Figure 9.

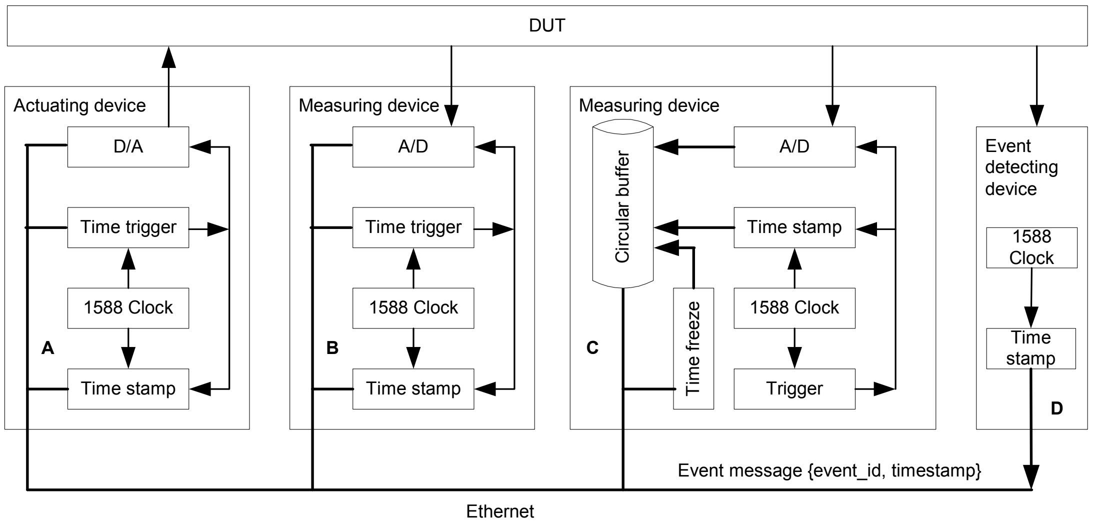

Figure 9: Random event triggering

In Figure 9 device D detects some application significant event and captures a timestamp marking the occurrence time. These events typically but not necessarily relate to the device under test. Device D then broadcasts an event message containing this timestamp along with an identifier for the event. The identifier can be:

1. Instrument specific. For example the event could be an internal change of state in device D resulting from the expiration of a timeout indicating no event was detected from the DUT. Instrument specific identifiers are more difficult to use since all receiving devices must know the details of the sending instrument in order to interpret the message.
2. T&M specific. This class of identifiers indicates events common to all T&M instrumentation and procedures. Examples include the start or completion of a measurement, or 'calibration complete'.
3. Application specific. These identifiers have meaning only in the context of a specific application.

Devices receiving the event message will have been preprogrammed to know what action if any to take based on the identifier and timestamp. In conventional message based systems the timing of such actions can only be related to the receipt time of the message, which as we have noted can have significant temporal jitter and delay. In the time based system these actions can be based on the actual event time. The message delivery latency and jitter is only significant if the action must occur before the message is received, i.e. a causality violation. Thus in Figure 9 devices A and B each have a time trigger block that can be set:

- To a time in the future with respect to the message receipt time (thus preserving causality) and
- With a definite time relationship to the event timestamp to preserve precise timing requirements of the application.

The trigger block and the local IEEE 1588 clock control the timing of actions, for example triggering a D/A in device A. Clearly there is still a premium on short message delivery latency for many applications. However there is no need to precisely control the delivery time but only the requirement to ensure that causality is not violated. Even if the message arrives too late to take the required action, a causality violation, this fact and the magnitude of the violation are immediately evident.

Device C illustrates a technique for overcoming the problem of message delivery latency in the case of measurement devices. In device C the measurement is triggered by any means, for example rapid periodic sampling, and the resulting value timestamp pairs are stored in a circular buffer. When the event message is received the buffer is frozen and the data of interest is retrieved based on comparison to the event timestamp and the timestamps in the buffer. This is how logic analyzers typically work. Thus the application designer can make an engineering tradeoff between the desired temporal resolution of the measurement, maximum message delivery latency and buffer size.

In all of these examples the precision to the timing relationships between events and the resulting actions is determined by the precision of the clock synchronization and the devices themselves but is independent of message delivery latency provided causality is not violated.

This model of control is being considered for use in fault recovery in complex control systems in industrial control. These techniques have been used for years in safety critical systems. In these systems each component is preprogrammed with a set of instructions to execute in the event of certain kinds of failure, such as a break in the communication link. The presence of a synchronized clock allows these ‘reaction scripts’ to assume some level of coordination with other devices based on time even in the absence of a communication link. In the T&M world this model of control is more likely to occur in implementing measurements surrounding some change of state in the DUT. The circular buffer technique in device C of Figure 9 is particularly useful in this regard in that it allows precise measurement of data before and after the causal event.

Scheduled event triggering
Not all actions taken in a T&M system are the result of random events. Many actions can be scheduled. These schedules may be based on absolute time as discussed in this section or periodic or relative to some predefined event as discussed later. A T&M system using absolute time based schedules is illustrated in Figure 10.

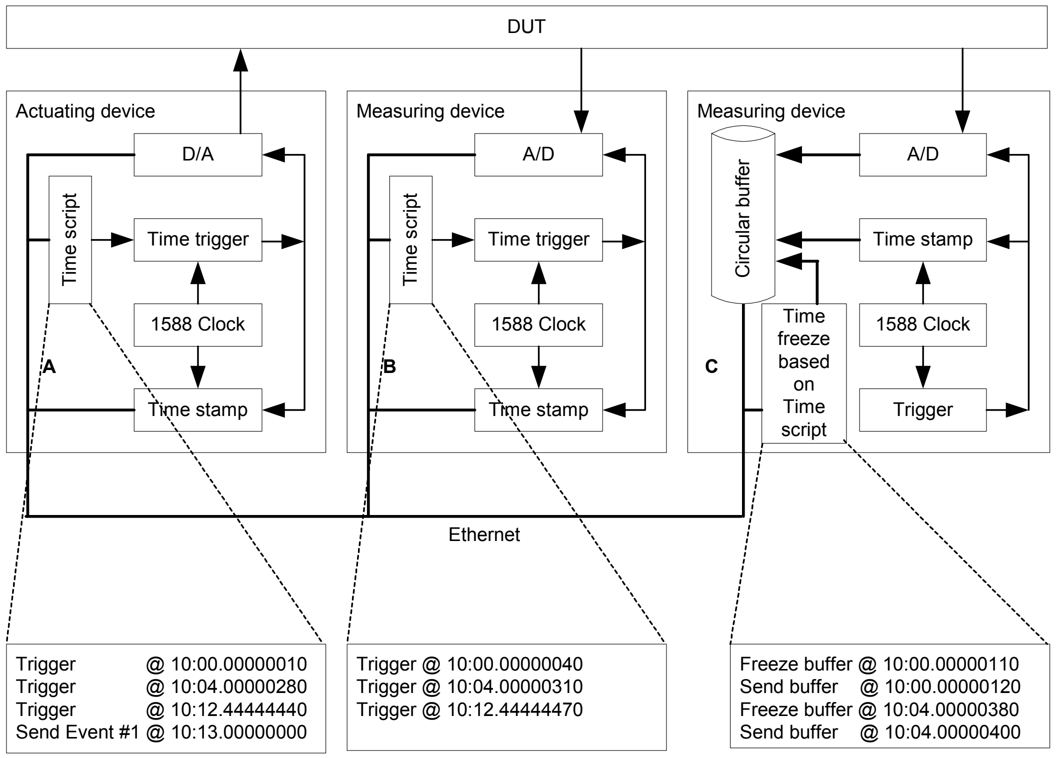

Figure 10: Scheduled event (episodic)

In this case the application controller typically preconfigures each device with a time script indicating what actions and their timing the device is to implement. As illustrated the actions can include not only simple actions such as triggering a D/A or A/D, but also more complex actions such as freezing a buffer, sending an event message, or delivering a block of data.

This methodology also makes it quite easy to have different but precise, temporally coordinated behavior in multiple devices. For example the time offset between triggering the D/A in device A and the triggering of the A/D in device B is specified to be 300 nanoseconds in Figure 10.

This episodic scheduled event model will find extensive use in T&M for creating test sequences that execute on occasion within some large suite of tests. In many cases these scripts can be prepositioned in the distributed devices. This allows very precisely coordinated time optimized measurements and sequences of measurements with little intervention from the initialing controller.

An obvious extension to the simple scheduling model just discussed is to allow the time scripts to specify periodic behavior. This case is illustrated in Figure 11. Here the time scripts specify the action and timing of an initial action and in addition specify a repeat interval. Again these time scripts permit very complex timing relationships to be implemented while maintaining precise temporal coordination. For example it would be possible to program repeating chirp signals from a variable frequency generator in which the chirp temporal behavior itself could be specified or modified by means of a time script.

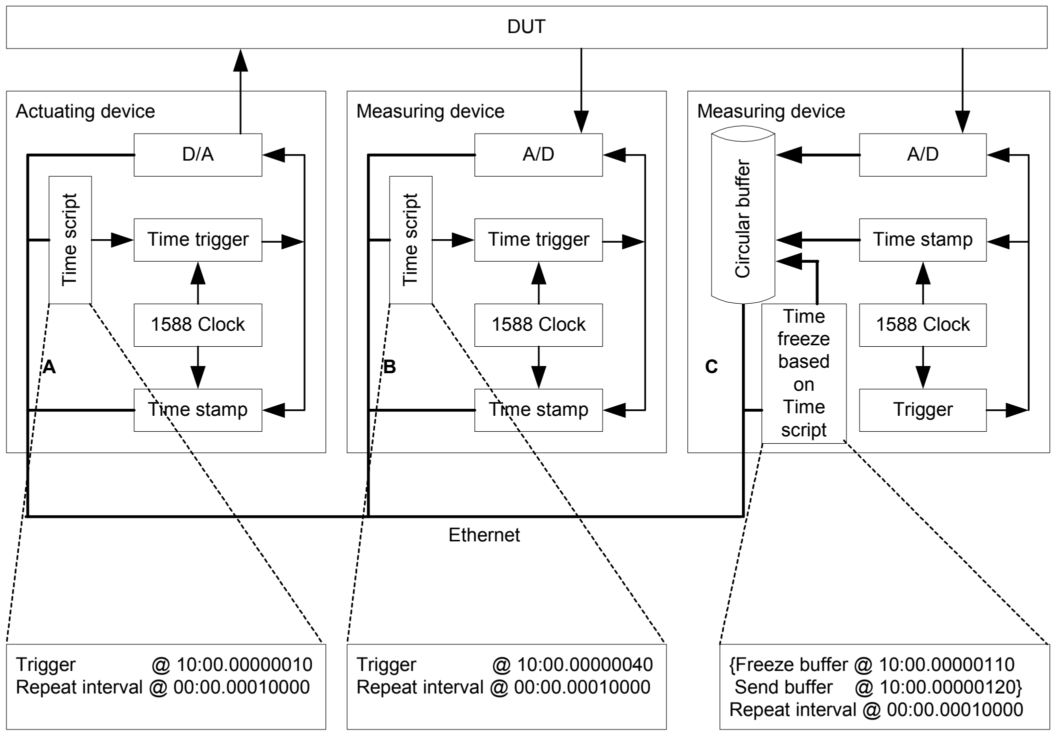

Figure 11: Scheduled event (periodic)

The periodic model will find extensive use in T&M system that do continuous or long term monitoring. The distributed devices will run independently but remain synchronized. Any resulting data will be timestamped for later correlation and interpretation. The large-scale industrial systems discussed earlier also make use of this type of scheduling.

The final scheduling model is illustrated in Figure 12. This model is exactly like the episodic and periodic except that the time scripts are written with respect to the time of a future event, in this case t_{start}, rather than an absolute time. This clearly provides much more flexibility to application designers and can be used to create very complex timing relationships including cascading relationships in which the time scripts themselves issue event messages containing event timestamps and identifiers derived from the original message.

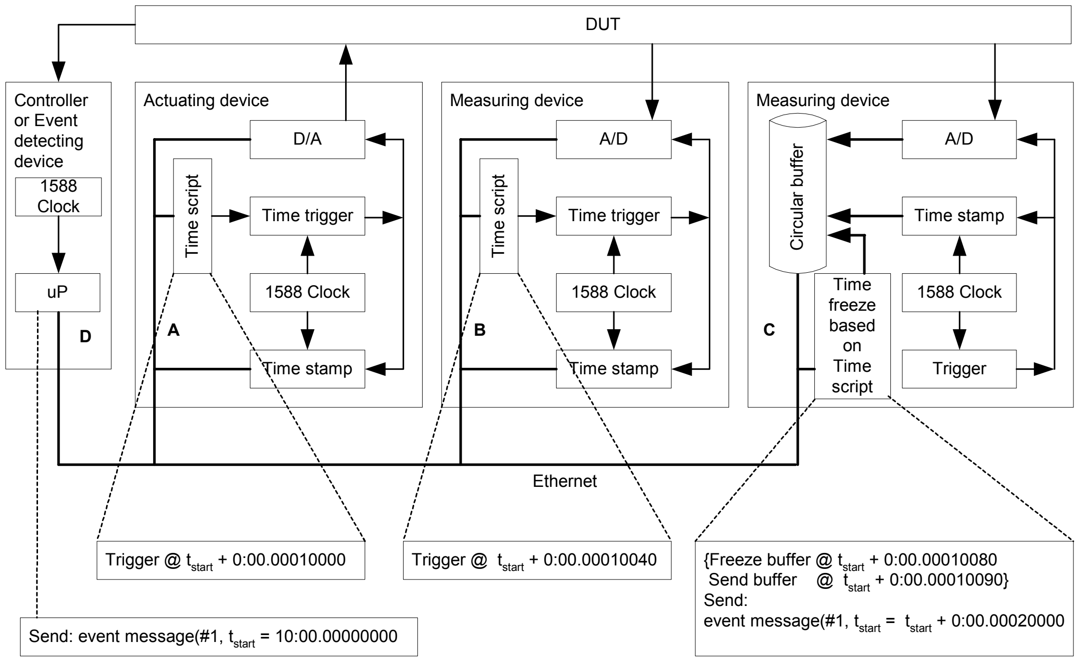
DUT
Controller or Event detecting device
1588 Clock
uP
D
Actuating device
D/A
Time script
Time trigger
1588 Clock
Time stamp
A
Measuring device
A/D
Time script
Time trigger
1588 Clock
Time stamp
B
Measuring device
Circular buffer
A/D
Time stamp
1588 Clock
Trigger
Time freeze based on Time script
C
Ethernet
Trigger @ t_{start} + 0:00.00010000
Trigger @ t_{start} + 0:00.00010040
{Freeze buffer @ t_{start} + 0:00.00010080
Send buffer @ t_{start} + 0:00.00010090}
Send:
event message(#1, t_{start} = t_{start} + 0:00.00020000
Send: event message(#1, t_{start} = 10:00.00000000

Figure 12: Scheduled event (cascaded)

The examples shown in this section are all relatively simple. These techniques can be combined with each other as suggested in the discussion on cascaded schedules.

The cascaded model is actually the most general of these models. It clearly can be used in the generation of complex T&M measurement sequences including those that have branch points for which the next sequence is not know until run-time. This model is being designed into motion control applications in industrial applications as a way of capturing complicated motion profiles that must be executed either periodically or based on some event. These occur in packaging machines, printing and other web handling machines, robotics and the like. A proprietary version of a clock synchronized control system using this model has been used in large industrial plants. The technique was well accepted but the users now require all control to be based on standards, yet another driving force for the use of IEEE 1588 within the industrial automation community.

These techniques can also be used in conjunction with more familiar triggering mechanisms like trigger busses and hardwired triggers. For example timestamping of data is clearly applicable to traditional techniques as a way of making the timing visible to the applications. Traditional triggering typically requires the use of arming to limit the time periods when triggers are accepted. Figure 13 illustrates a simple arming state machine.

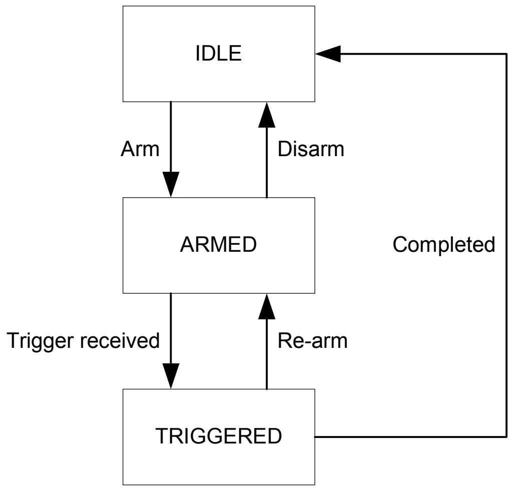

Figure 13: Simple arming state machine

Normally the system or a component is in the IDLE state. Upon receiving the Arm event the state changes to ARMED state, the only state in which Trigger received events have any resulting action. When armed the receipt of a trigger event causes the state to change to TRIGGERED state. In this state whatever resulting actions are allowed may take place. Upon completion there are two possible paths. One is the result of a Re-arm event, which allows repetitive triggering; the other is the Completed event, which returns to the idle condition. Note that any or all of the events shown in Figure 13 may be time-based events arising from scripts in a device or generated by a controller.

Time triggers based in hardware can also be used to generate interrupts to the local microprocessor. This can be used in a variety of ways from generating timeouts, to starting new trains of logical program execution. This use of triggers has an advantage over the normal technique of operating system generated timeouts in that they can be generated synchronously in multiple nodes without requiring an interchange of messages.

Finally mention must be made of the use of synchronized clocks for establishing a sort of global time grid in a system. This technique is used quite commonly in control environments to regulate the progress of a process. In the case of T&M it could be used to regulate the progression of a test sequence. The time grid establishes a set of time ticks at regular intervals that are used as synchronization points by the applications. By selecting the tick spacing to suit the application considerable simplification in the control structures of the distributed programs can be achieved since there is no need for message exchange. Each element of the system is designed to start and reach certain completion points by a designated tick. Since the time base is system wide the ticks are identified by their time and the precision will be that of the clock synchronization which will be much better than that of a single message exchange that would otherwise have to be used. As in other cases failures to meet the completion points is immediately evident.

A few words of caution

Throughout this note various situations are noted where two timestamps or a timestamp and the clock are tested for equality. The operation of a trigger block and the operation of a time script both require equality tests. Consider the following:

1. The time base has finite, non-zero precision. This means that it is not possible to state, in other than the computer science sense of identical bits, that two times are in fact the same. This is no different than the question of whether two measured voltages are the same. Recall that precision includes resolution of hardware and data types, noise and other impairments arising from the oscillators, and the action of the slave servos.
2. Depending on how the servo adjusts the clock, the time base may not include all of the values allowed by the number of bits present. For example if the adjustment is made by bit augmentation or deletion then it is almost certain that the clocks will occasionally skip a numerical value. Clearly any comparison must take this into account for example by always using a greater than or equals comparison rather than just equals. Failure to do this will result in events in time scripts, or time triggers failing to execute event though the stated time has passed.
3. If greater than or equals is used then the times in a trigger block or time script can become stale. That is the time has passed, presumably resulting in an execution, but repeated comparison will always indicate that the event has occurred again. The obvious solution is to delete or otherwise inactivate a time trigger or statement in a time script once it has executed.
4. A policy needs to be in place to handle time triggers or scripts statements that when configured are already in the past. This is clearly application dependent. In some cases this condition should generate an error exception, in others the immediate execution of the resulting action.
5. A policy needs to be in place for handling messages that contain no timestamp. A useful policy is to treat the implicit time as ‘now’.

It is tempting to use time triggers to implement some hardware action by having the trigger interrupt the local microprocessor that in turn manages the hardware action. While this can certainly be done in most cases the timing precision of the resulting action will be drastically reduced from the clock synchronization precision due to the jitter of the operating system, i.e. from a few nanoseconds to tens of microseconds at best. Critical timing always should involve direct hardware support coupled to time triggers or timestamp registers.

It is very easy to create time specifications that are impossible to realize. To guarantee that a timestamp can be captured at any instant or that multiple time triggers can always execute requires that dedicated hardware be available for each temporal instance. This is clearly not possible. A given device will always have a finite number of timestamp registers and time triggers. This means that careful attention needs to be given to the execution of software within these devices to harvest timestamps and to reload time triggers in such a way as to meet the temporal requirements. Component designers should carefully document the resulting timing limitations so that application designers can create viable systems.

IEEE 1588 time bases are continuous from their epoch. If a UTC time base is required then the issue of leap seconds must be managed. As noted the IEEE 1588 time base does not reflect the insertion of a leap second, unlike NTP, but does provide for making the leap second information available to applications. There needs to be agreement in the LXI community on where the transformation between the continuous time base of IEEE 1588 and the discontinuous time base of UTC should be made. Note that the leap second corrections can result in missing time values in UTC but not in IEEE 1588. Thus computing the interval between two timestamps from an IEEE 1588 time base will give the correct answer whereas the difference between two UTC timestamps must involve correction for any leap second insertion in the interval. On the other hand specifying a time in calendar time, for example every day at noon, is easy in UTC but may require a leap second correction to obtain the correct time in an IEEE 1588 time base.

Summary

The presence of a common sense of time allows individual devices in a distributed T&M system to take action based on time. This note has sketched a number of techniques to implement this capability including timestamp and time trigger registers, time scripts, and circular buffers. The discussion has also shown a number of ways in which this capability may be used to create both simple and complex interactions among instruments and controllers.

Several obvious benefits have been discussed:

- The utilization of source generated timestamps for post acquisition data and event correlation.
- The ability to precisely coordinate actions in several devices in a distributed T&M system. Simultaneous actions or actions with defined time relationships are straightforward.
- Timing precision is a function of the precision of the time base and individual devices rather than being a function of the receipt time of a message. This greatly relaxes the constraints on both the messaging system and application software.

There are some less obvious consequences of the time-based capability.

- The specification of interactions via time scripts allows a tradeoff between configuration, the loading of the time scripts, and run time messaging. In effect the common sense of time replaces the need for most run time control messages in a system making optimum use of time scripts.
- The use of time scripts also permits better partitioning of time critical and non-time critical tasks in controllers and instruments. For example a controller can partition complex tests into configuration and operational. The operation phase will generally be started with a control message but will then proceed independently allowing the controller to focus on processing received data efficiently.
- Once configured, many complex instrument sequences can take place either strictly on the basis of time or augmented by instrument generated event messages

to continue cascades of events. This again allows for simpler code in the controllers.

- Timestamping of critical events and data in the components of a distributed T&M system provides greater visibility for the system integration and debug phase of a project. Our experience indicates that in properly partitioned implementations the presence of these timestamps makes it clear whether a fault is due to the implementation or application code of a given device or in the logic of the interactions between devices.

The use of a common sense of time as described in this note should be regarded as providing another degree of freedom or design flexibility to system designers and integrators. Time based techniques may be used independently or in conjunction with message based techniques allowing better optimization of system performance. Each technique should be used where it provides the greatest advantage.