An Application of IEEE 1588 to Industrial Automation

Ken Harris, *Member, IEEE*

Abstract **– This paper describes an application of the IEEE 1588 standard to Industrial Automation. Key application use cases are identified that can benefit from time-based control techniques to improve performance results over traditional control methods. This paper will also briefly discuss how the 1588 standard may be adopted to suit these applications. Application problems specific to industrial automation are enumerated and candidate solutions described.**

Index Terms **– IEEE 1588, Precision Time Protocol, Time-Based Control, Industrial Automation**

I. INTRODUCTION

One trend in discrete part manufacturing is toward faster and higher precision part production – more parts per minute and better quality. Traditional control solutions can be stretched to their limits. By replacing traditional control solutions with time-based control, faster and higher precision goals can be realized. The IEEE 1588 standard provides a solution that can be easily adopted by the industrial control industry to distribute precision time for time-based control on the factory floor.

II. THE CASE FOR TIME-BASED CONTROL

In traditional sequential control systems where input sensors, output actuators, and industrial controllers are distributed over a local area network, the control algorithms are typically scan-based and asynchronous, and consequently suffer from significant processing jitter. Some systems employ change of state or event-triggered techniques to improve performance. However, time-based control provides the best performance alternative.

A. *Scan-Based Control*

For scan-based control, the process is as follows for a simple input, control, output sequence. Input data from sensor devices are sent to the controller at a periodic rate. The controller runs its control algorithm at a periodic rate and output results are sent to the output actuators at a periodic rate. The inputs and outputs change state asynchronously to the periodic input and output scan.

This input-process-output sequence creates a very elastic or jittery input to output delay. The delay jitter will be a function of when the input changes in relation to the asynchronous periodic scans of the input, controller and output transfer, network transport delays, and internal device delays. These delays are illustrated in **Figure 1**.

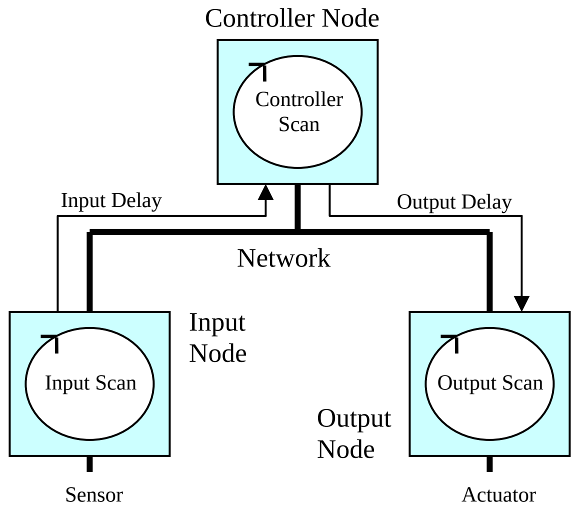


Figure 1- Jitter delays for Input-Process-Output Sequence

B. *Event-Triggered Control*

Event-triggered or change-of-state control can significantly reduce jitter. With change-of-state operation the input, control, and output, scan delays are eliminated. When an input transition is detected by the input device, it immediately sends it to the controller. The controller is interrupted when the input arrives and immediately executes its processing algorithm and sends the result to the output device. When the output message arrives at the output device it immediately actuates the output.

This approach will still incur jitter delays due to network transport. If a large number of input transitions occur at once, network congestion and packet loss may occur resulting in additional jitter delays and possibly machine failure. Also, since many I/O devices do not support event trigger mechanisms, this approach is often less viable. In practice, a traditional control system will use a combination of scan-based and event-based control mechanisms.

C. *Time-Based Control*

For many applications, the jitter won’t matter as long as the application response times are satisfied. However, some applications require more precision and have a low tolerance to jitter. For these applications, a time-based control system can solve these problems more effectively.

In a time-based system, an association is made between input and output events and time. Time becomes an integral function of the control system and control algorithms. All devices in the system have the same notion of time. In such a system, the input events are time-stamped and output events are scheduled. The control system precisely knows when the input was sampled and can precisely determine when the output should be actuated. The output device can schedule the output to actuate at a predetermined time.

The only jitter sources for this system are those associated with accurately time-stamping inputs and outputs.

Table 1 shows relative delta jitter delays for the three control mechanisms discussed. The delay numbers indicate the processing delays for the component. The delta jitter is the maximum minus the minimum jitter for the component. Notice how the time-based approach eliminates the jitter sources in the control system.

Table 1 - Delta jitter for various control mechanisms

Jitter or Delay Source	Delay	Delta Jitter (max – min)		
		Scan-Based	Event-Triggered	Time-Based
Input	0.2 msec	10 msec	0	0
Input Network	1 msec	1 msec	1 msec	0
Controller	10 msec	100 msec	10 msec	0
Output Network	1 msec	1 msec	1 msec	0
Output	0.2 msec	10 msec	0	0
Total	12.4 msec	122 msec	12 msec	0

III. A Real World Example

The advantages of time-based control can best be illustrated with a real-world example. In a high-speed conveyor diverter application, individually manufactured parts travel along a conveyor at a constant rate of speed. A "part" might be a candy bar, a diaper or any discretely manufactured product. In this system, the intent is to detect the presence of individual parts as they move down the conveyor, perform on-the-fly analysis of the part to determine if it is a defective part, and then trigger actuation downstream to reject the defective part.

If the resolution of the control system does not match the speed of the conveyor system, then the wrong part or more than one part will be rejected.

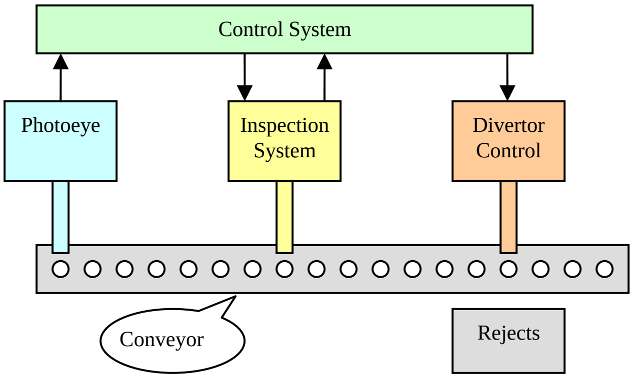


Figure 2 - High-Speed Conveyor Diverter Block Diagram

A. *Traditional Scan-Based Control Operation*

In this example, an input sensor such as a photoelectric eye is mounted along the conveyor to detect the presence of a part. The "part detected" status input is sent to the controller as part of the input scan and provides a registration mechanism to track the part as it moves down the conveyor. Knowing the speed of the conveyor, the controller can calculate the location of the part at any given time. An optical inspection system will also be located along the conveyor. The inspection system examines the parts as they move down the system and determines whether the part is good or defective. The controller matches a defective part detected by the inspection system with the part moving down the conveyor and at the appropriate time signals the diverter system to reject the part.

The maximum speed of the conveyor, and consequently the maximum number of parts that can be manufactured per minute, will be determined by the total input-to-output jitter. Using the numbers from the previous section for scan-based control, the maximum speed and parts per minute are calculated as follows:

Part resolution = 122 msec jitter
Maximum speed = 1/122 = ~8 parts per second
Maximum ppm = 8 * 60 = 480 parts per minute

B. *Time-Based Control Operation*

Now consider the same system using time-based control. When the photoelectric eye detects a part, a time stamp is recorded indicating the time that the part was detected. The controller sends the inspection system a time-stamped schedule to signal when the part should be inspected. The controller sends the diverter system a time-stamped schedule to signal when a defective part should be diverted. For this case, the maximum speed of the system is limited by the delays through the system from input to output. The maximum speed and parts per minute are calculated as follows:

Part resolution = 12.4 msec jitter
Maximum speed = 1/12.4 = ~80 parts per second
Maximum ppm = 80 * 60 = 4,800 parts per minute

As previously discussed, the jitter delays are confined to the electrical and mechanical delays for the input detection and output actuation device. Only the transport delays become a factor. The precision of the system is limited to the transport delays, and the time-stamping and scheduling accuracies of the respective input and output devices.

Using a time-base control mechanism shows a ten-fold improvement in performance. Jitter sources in the system are virtually eliminated. System performance is only limited by the processing speed of input, control and output devices

IV. APPLICATION OF TIME TO INDUSTRIAL CONTROL

Table 2 lists several applications of time to industrial control. For example, time-stamping and scheduled outputs were described in the real-world example of the previous section. Each application of time is briefly discussed in this section, along with examples of how it may be applied to industrial control solutions.

Table 2 - Application of Time to Industrial Control

Time Usage	Control Application	Industry Example
Time of Day	General Logging Scheduling	Shift Management
Time-Stamped Inputs	Alarm & Events Sequence of Events First Fault Detection	Power Generation Pipeline
Scheduled Outputs	High-Speed Sort and Diverter (1,000 now, 10,000 future parts per minute)	Discrete part manufacturing such as candy bars, cigarettes, juice boxes, diapers, razor blades, etc.
Synchronized Outputs	Motion Position and Velocity Control	Coordinated Drives Robotics

A. *Time of Day*

Time is distributed to all devices in the system to perform scheduled activities based on time of day. These activities might include shift startup, shutdown, and reporting operations. For these systems, it is also desirable to automatically distribute time zone and daylight saving time to all factory-floor devices, and to manage the annual transitions between standard and daylight saving time.

B. *Time-Stamped Inputs*

An input is time stamped to record the time of the input transition on the rising edge, falling edge or both. The input may be a physical device such as a sensor or logical input such as a detected alarm condition. The time stamp is carried with the input data for detection of alarms and events. Applications use time-stamped inputs for time-based control, general logging, and trending or statistical analysis. One primary application is to determine from a set of time stamps the sequence in which a set of events has occurred or when the first fault occurred among a sequence of faults.

C. *Scheduled Outputs*

A time-stamped or scheduled output is used to schedule the time when the output should be applied. Applications use scheduled outputs for time-based control to precisely actuate or assert an output. The output may be physical as in the case of sort or diverter application or logical such as triggering some control action.

D. *Synchronized or Coordinated Control*

Synchronized or coordinated control is used to simultaneously coordinate the actions in one or more devices in a time coordinated manner. This is very prevalent in distributed motion control where actions for individual motor drives are synchronized to each other. It is also required in robotics to coordinate individual axis of motion.

V. ADOPTING 1588 FOR INDUSTRIAL CONTROL

This section describes how the IEEE 1588 standard may be adopted or applied to industrial automation systems to satisfy applications such as those previously discussed. The adoption described is based on the ODVA standard and defines a PTP Profile for devices that are compliant for all of the ODVA supported networks. The ODVA adoption of 1588 is commonly referred to as CIP Sync™. CIP Sync supports version 2 of the 1588 standard.

A. *CIP Sync Profile*

The CIP Sync profile specifies the same set of defaults as the Delay Request / Response Default PTP Profile in the 1588 standard. These defaults satisfy most of the industrial application requirements for distributing time throughout the control system. The defaults, such as multicast messaging, sync updates in the one-second range, and the delay request-response path measurement mechanism are all suitable for implementing clocks over industrial networks to meet synchronization requirements.

B. *PTP Clock Types*

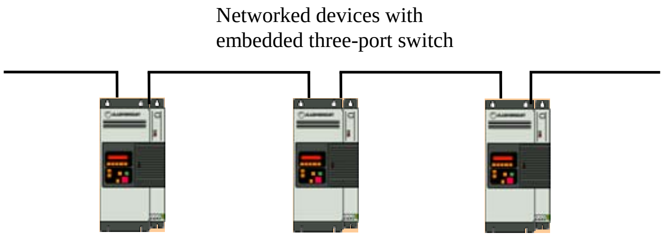
Table 3 shows a list of 1588 clocks and how they best fit automation devices.

Table 3 - PTP Clocks and Automation Devices

Clock	**Type**	**Device**	**Application**
Grandmaster Ordinary Clock	GPS Handset NTP Radio – IREG	Clock Source Handset Controller Handset Switch	All
Slave Only Ordinary Clock	Software Clock or Hardware Assisted	Sensor Actuator Motor Drive	Software Clocks < 100 µsec Hardware Assist < 100 nsec
Hybrid Clock	Ordinary Clock	Sensor	Linear

	With a Transparent Clock	Actuator Motor Drive	Distributed Applications
Boundary Clock		Switch or Router	All
Transparent Clock	End-To-End (E2E)	Switch or Router	All

Notice that many I/O devices will implement the hybrid clock. The 1588 concept of a hybrid clock is very attractive for many devices to satisfy long linear distributed applications where daisy-chained connections are preferred over star connections, and to reduce wiring and switch requirements. These applications may be as simple as distributing devices along a robot arm or as extensive as distributing devices along a quarter-mile, paper-mill application.



GPS clocks are best applied where precise time of day is required or subsystems are distributed over a large geographical area. However, a controller or a switch with a “hand set” clock is sufficient for many applications.

For switches and routers that implement transparent clocks, the end-to-end (E2E) mechanism is preferred over the peer-to-peer (P2P) mechanism for maximum interoperability in nonhomogenous systems (Note: This is due to the restriction that P2P devices only work with P2P devices). Most systems will contain both 1588 and non-1588 compliant switches.

C. *Domain Support*

In the CIP Sync profile, only one PTP clock domain is supported, domain 0, to simplify device and system implementation and deployment.

D. *PTP Options Supported*

Grandmaster devices such as controllers and GPS clocks should implement the PTP Alternate Time Scale option. This option provides the mechanism to distribute local time zone and daylight saving time throughout the factory floor.

For clock backup and redundancy, the normal operation of the Best Master Control Algorithm (BMCA) will be sufficient for most applications. However, backup grandmasters may implement the PTP grandmaster clusters option for faster switch-over time.

E. *Handling Step Changes to Time*

The IEEE 1588 standard defines the mechanism to distribute and synchronize time across the system, but does not specify how to handle perturbations in time that may occur during normal operation of the control system. These changes may occur due to one or more of the following conditions:

- The user adjusts the master clock whose type is “hand set.”
- A master with a more accurate clock becomes available (new grandmaster). This may occur during system startup or after the system has been running for some time.
- The time master is temporarily disconnected from the slave clock and then reconnected. In this situation, given any discrepancy in time between the master and the slave, a step change will occur.

Some applications need a very stable clock for scheduling periodic control algorithms. This is especially true for motion control applications that run precision control loops. A significant jump in the system time will impact periodic control algorithms.

What is needed is a clock that can tolerate steps in time and still minimize impact to the control system.

F. *CIP Sync Clock Model*

CIP Sync defines a clock model to address step changes in time. The clock model is illustrated in **Figure 3**.

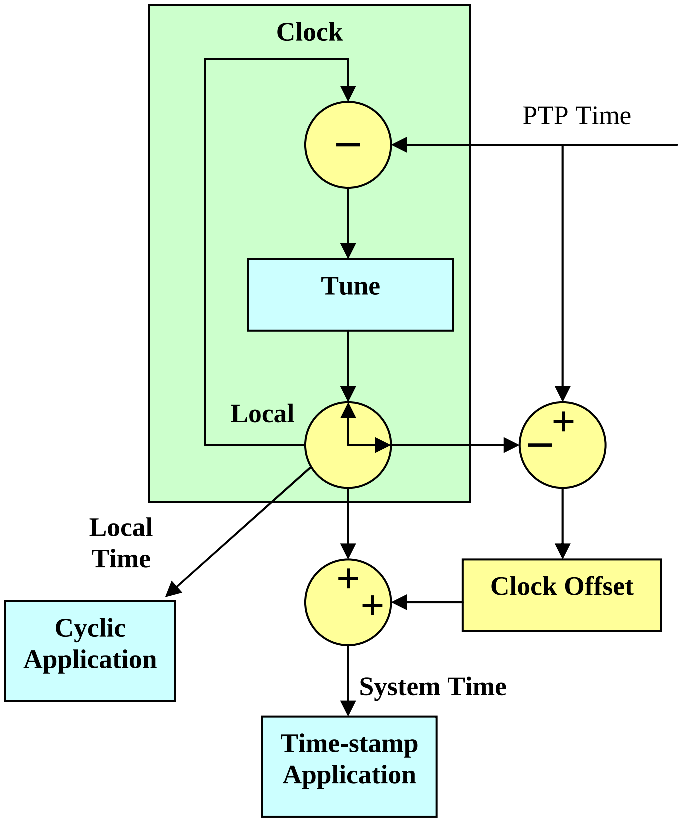


Figure 3 - Clock Model With Local Clock and Offset

The local clock is a frequency-disciplined clock that is tuned (i.e. syntonized) by the 1588 Precision Time Protocol each time a PTP Sync and/or follow-up message is received. The clock will tick at some nominal rate and be adjusted by the PTP protocol and tuning algorithm to match the rate of the masters' clock. This tuning process may take several seconds during startup and initialization of the system for a default one-second sync rate. The clock may be set to zero at startup, but the absolute value is never set thereafter. The accuracy of the clock will depend on whether the implementation is software or hardware assist, the resolution of the clock, the crystal, and all the other design parameters that go into making an accurate frequency-disciplined clock.

The local clock is used to schedule all periodic or cyclic operations on the device.

An offset value (ClockOffset) is maintained between PTP time transmitted in sync messages and the local time (LocalTime). When a device needs the current PTP time (PTPTime) it will read the local time and add the offset value to get the current system time. ClockOffset and PTPTime equations are as follows:

ClockOffset = PTPTime – LocalTime
PTPTime = LocalTime + ClockOffset

Note: If UCT time is required, the device will adjust the time value for leap seconds.

G. Tuning the Clock

Tuning is adjusted based on two conditions: 1. A small frequency offset from the master clock or 2. A large frequency offset from the master clock such as a step change. The magnitude of this offset, small or large, is determined by the SyncThreshold value. The SyncThreshold is set based on the synchronization requirements of the application.

Offset From Master < SyncThreshold: Tune clock
Offset From Master > SyncThreshold: Tune clock but clamp clock frequency adjustment

When the offset from the master exceeds the threshold, the clock frequency adjustment is clamped. The clamp rate depends on the requirements of the application. For a step change in time, this normally results in a small one-time perturbation to the clock. If the step is a result of a grandmaster change, then the clamping period may continue for some time as the clock frequency is tuned to the frequency of the new master. During this time, the application may operate in a slightly degraded mode.

Periodic or cyclic events may then be scheduled based on the local clock and will not be affected by large step changes in time – provided that the local clock remains synchronized to the PTP master-clock frequency.

H. Tuning Results for a Time Step at Master Clock

Figure 4 shows the result of clamping the slave clock when a time-step change at the master is detected by the slave. The slave clock limits the response to the assumed change in frequency of the master clock. Note that a small step response is incurred for two tuning cycles.

This paper presents a fairly simplistic solution for handling step changes, which will suffice for many applications. But a more elegant solution might skip the clock tuning for one cycle or apply other filtering techniques to minimize or null the response.

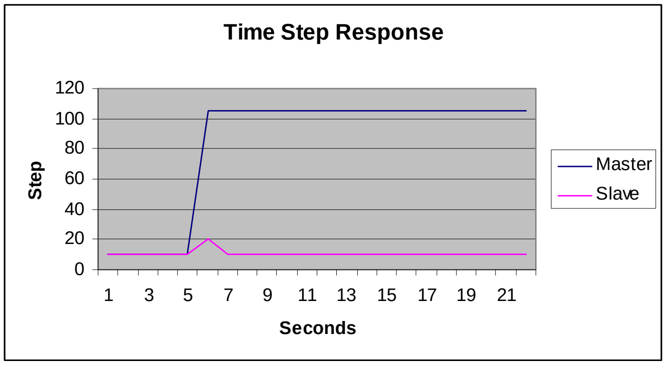


Figure 4 - Slave Time-Step Response to Master Step

I. Tuning Results for Time Step Due to Master Change

Figure 5 shows the result of clamping the slave clock when a time step due to master change is detected by the slave. The slave clock limits the response to the change in frequency of the master clock. Note that a small step response is incurred for several tuning cycles.

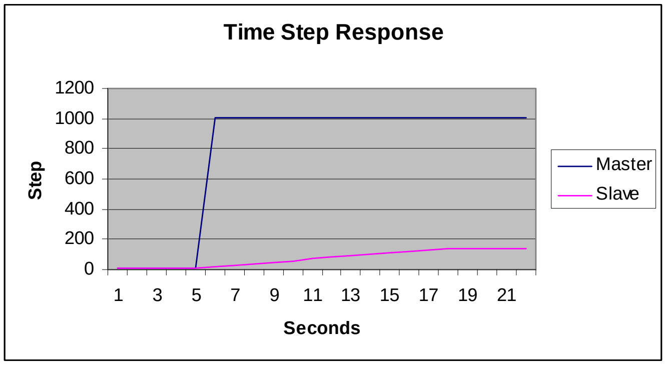


Figure 5 - Slave Time-Step Response to Master Change

J. Time-Stamp Compensation

When a jump to the master time occurs, not all clocks in the system will see the jump at the same time. For instance, note the master clock and two slave clocks in **Figure 6**. If there is a step change to the master's clock, the amount of time it takes to propagate to the Node 1 slave clock and the Node 2 slave clock will differ by the delay through the PTP boundary clock. For some implementations, the delay through the boundary clock may be as long as one sync interval (when the sync interval is one second). Time stamps generated by Node 1 will differ by the amount of the time step until all clocks see the same step and adjust their clocks accordingly.

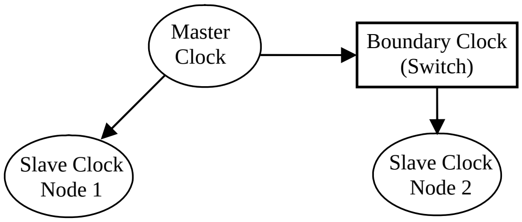


Figure 6 - Time-Step Change With PTP Clocks

When comparing time stamps between two nodes or two time stamps within the same node, some consideration should be given to whether a step has occurred during the time interval between the two time stamps. Adjusting one or both of the time stamps to account for this time step is referred to as time-stamp compensation.

An example is illustrated in **Figure 7**. A time step occurs between time 200 and what should be time 300 on Node 1 making the time actually 100300. Node 2's time is still at 300 because it has not yet seen the time step. Any comparison of time stamps at time 100300 (at Node 1) and 300 (at Node 2) should adjust one of the time stamps so that the comparison is at the same time base.

A similar process should be followed when comparing two time stamps from the same node. Again refer to Figure 7. A time step occurs at Node 2 between times 310 and 410. Notice that time 310 is adjusted to time 100310 to account for the step change.

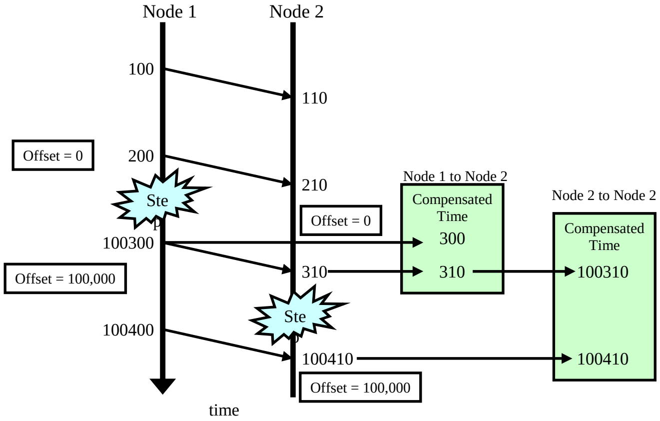


Figure 7 – Time-Stamp Compensation Examples

One algorithm for compensating time stamps may be implemented by capturing the ClockOffset (see section V.F CIP Sync Clock Model) along with the time stamp. This offset value is carried with the time-stamp value to be applied to the compensating algorithm. The offset provides an indication of when the step has occurred.

Two algorithms are defined in the CIP Sync profile: One for comparing time stamps between two nodes and one for comparing time stamps within a single node.

K. Conclusion

Time-base control will satisfy the requirements for high-performance applications in industrial automation. The IEEE 1588 standard is ideal and easily adaptable for distributing a precision time-base to all devices of the control system. The challenge will be to migrate control-system devices particularly I/O to the IEEE 1588 standard and time-based control solutions. This task will be aided as 1588 becomes more prevalent in the industrial automation arena.

REFERENCES

[1] IEEE 1588-2008 Standard for Precision Clock Synchronization Protocol for Networked Measurement and Control Systems.
[2] *Common Industrial Protocol (CIP) Volume I Edition 3.4 July 2008*, ODVA. Available http://www.odva.org