Fundamentals of Precision Time Protocol

Rudy Klecka
Principal Engineer
Cisco Systems

Presented at the ODVA
2015 Industry Conference & 17th Annual Meeting
October 13-15, 2015
Frisco, Texas, USA

Abstract

This session will provide a general background on IEEE 1588 Precision Time Protocol (PTP), how it works, some basic terminology, and its main uses in the market. There will be a discussion on PTP implementations (with a primary emphasis on Industrial products). The session will also touch on other related timing protocols and future enhancements to PTP.

Keywords

Precision Time Protocol, PTP, NTP, GPS, Two-Way Time Transfer.

“Timing is Everything”

“Timing is everything.” That quote can be applied to many things in life, comedy, business, debate – but also to engineering – and especially, as discussed in this paper, to Controls Engineering.

Time is a fundamental service to all control loops – whether it be with time constants of seconds in Process Control loops where the precision may not be critical to sub-millisecond Motion Control loops where nanoseconds matter – time is essential. Getting all the components of a system having the same concept of time can be achieved in many ways. Originally, assembly lines used cams to mechanically coordinate machine operation – much of which is still in use today by translating cam position to phase and frequency (i.e., time) digitally using communications protocols such as the ongoing peer-to-peer work in ODVA.

In the broader world, time is also distributed in various ways: GPS (or more generically GNSS), NTP, IRIG, and PTP are the most popular and well known. Each has their own raison d'être, because each was developed and adopted for different purposes. Without getting too much into the details of how the various protocols work and why they came to be, this paper aims to give a brief introduction to these various protocols, describe how they differ from PTP, then focus on the fundamentals of PTP and how it is currently evolving for the various industrial markets.

Various Time Distribution Protocols

GPS is probably the most widely recognized and used protocol for time distribution – at least in the US – with its worldly counterparts of Galileo (Europe), GLONASS (Russia; glo-NAS), and BeiDou (China; bay-DOH), with several other regional systems currently under development, making up a suite of GNSS protocols. As the “P” in “GPS” implies, these are not only used for time, but also for position – a key difference from PTP. GNSS technologies are also satellite based as opposed to Ethernet (or other
LAN/WAN technologies) like PTP and NTP, making GNSS suitable for use over wide areas – anywhere in view of the constellation of satellites. Which also points to one of its limitations: unlike PTP and NTP, which can be used deep inside buildings – anywhere you can access wired or wireless LANs, GNSS technologies require that you be “line-of-sight” to the GNSS constellation. You may have noticed that sometimes, deep inside a parking garage, for example, you can no longer access GPS on your iPhone – this is probably why. GPS is similar to PTP in its accuracy, as well – achieving accuracy on the order of 10s of nanoseconds – although with PTP there are applications achieving picosecond accuracy. Another notable difference is that GNSS is a “One Way Time Transfer” protocol, as opposed to PTP and NTP, which are “Two Way Time Transfer (TWTT)” protocols – more on that later...

Another time distribution protocol, barely known to the general populace but widely used in many industrial applications, is the Inter-range instrumentation group (IRIG) time codes. Originally developed for US Military use in 1960, the standard has seen adoption in a broad range of industrial applications – probably mostly due to a combination of a couple of factors: uses a wired (coaxial) medium and has reasonably good accuracy (microsecond), allowing it to be used indoors for many stringent applications.

Of all of these popular time technologies, the most similar to PTP is NTP: it is a Two-Way Time Transfer protocol distributed over LAN/WAN technologies. Where it differs significantly, and the reason PTP was developed as an improvement upon NTP, is with accuracy. NTP generally achieves around 100 millisecond accuracy in typical Internet based applications and, when used in controlled environments, 10 millisecond accuracy is easily achievable. PTP, on the other hand, can achieve 100 microsecond accuracy in most applications and, when hardware timestamping is utilized, sub-microsecond accuracy is easily achievable. The reason PTP can achieve orders of magnitudes better accuracy than NTP is because PTP adds mechanisms to the networking infrastructure (i.e., switches and routers) to minimize Packet Delay Variation (PDV) and Asymmetry (more on that later, too). In other words, NTP transfer time in an “over-the-top” application (meaning it is implemented at the Application layer) whereas PTP requires infrastructure changes lower in the stack.

One-Way Time Transfer (OWTT) Basics

Before delving into Two-Way Time Transfer, a description of One-Way Time Transfer (OWTT) is probably in order. As the name implies, OWTT protocols, the most popular being GPS, only send messages in one direction. That is, the masters only send time out to the slaves – there are no reverse mechanisms – which means there is no direct message for determining the amount of delay in time from the master to the slave. Conceptually the way GPS works is, in order to get time (and position) from a GPS network, the slave must see multiple (typically four) satellites and, since the delay through the atmosphere is (fairly) constant, comparing Time of Transmission (TOT; i.e., t1 timestamp) to the Time of Arrival (TOA; i.e., t2 timestamp) from multiple, synchronous, geostationary satellites, the slave, using its own concept of time, can calculate the Time of Flight (TOF; i.e., delay) from each. Basically: four equations and four unknowns. In practice, GPS/GNSS solves this problem is via “multilateration” – conceptually similar to triangulation and trilateration – and can be thought of as finding the intersection of four hyperboloids. Probably the main reason why GPS/GNSS (or other OWTT protocol) is not used for Industrial Automation applications is the fact that the medium most readily available in factories is constructed of LAN/WAN technologies and is therefore not even close to having constant delay (as with the atmosphere with GPS). Therefore, another method is required.

Two-Way Time Transfer (TWTT) Basics

All TWTT protocols follow the same basic mechanisms, and no discussion of the fundamentals of PTP would be complete without the basic message flow (see Figure 1:PTP TWTT Message Flow):
Starting with the terminology, there are two basic message types: Event Messages and General Messages – the difference between the two being that the former are timestamped and the latter are not.

Note that, since the message flow shown in Figure 1 (and mostly what is discussed subsequently) is showing a “Two Step, End-to-End” message flow, it does not include all the messages shown in Table 1. Therefore, the “pDelay” messages are not shown since those are unique to Peer-to-Peer implementations.

<table>
<thead>
<tr>
<th>Event Messages</th>
<th>General Messages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sync</td>
<td>Follow_Up</td>
</tr>
<tr>
<td>Delay_Request</td>
<td>Delay_Resp</td>
</tr>
<tr>
<td>pDelay_Request</td>
<td>Pdelay_Resp_Follow_Up</td>
</tr>
<tr>
<td>pDelay_Response</td>
<td>Announce</td>
</tr>
<tr>
<td></td>
<td>Signaling</td>
</tr>
<tr>
<td></td>
<td>Management</td>
</tr>
</tbody>
</table>

The “Master” is attempting to transfer time to the “Slave” via these messages, timestamping the Event Messages and passing other necessary information in the General Messages. The Master timestamps when the Sync Message leaves it with the “t1” timestamp, the arrival of which at the Slave is timestamped with “t2”. A "Delay Request" is then sent from the Slave back to the Master to determine the amount of delay between the Master and Slave, timestamping its departure at “t3” and arrival at the Master with “t4”. The t1 timestamp is transferred to the Slave either in the Sync message itself (“One-Step” method) or in a “Follow Up” message (“Two Step” method). Once the t4 timestamp is sent to the Slave in the “Delay Response” message, the Slave now has all four necessary timestamps.
These four timestamps form the basis for how time is determined. First by calculating the “Fractional Frequency Offset (FFO)” (using t1 and t2) and then the “Delay”. First FFO is determined from just a set of t1 and t2 timestamps using only Sync messages:

Figure 2: First Step: Syntonize

\[
\text{FFO} = \frac{(t2' - t2) - (t1' - t1)}{(t1' - t1)}
\]

In this FFO equation an assumption is made that the transmission delay between the Master and Slave is constant. In other words, there is no “Packet Delay Variation (PDV)” from transmission to transmission. We will examine how this assumption causes time error later.

Next, delay is calculated:
Using the following equation:

\[
\text{Delay} = \frac{(t_2 - t_1) + (t_4 - t_3)}{2},
\]

restated:

\[
\text{Delay} = \frac{\text{Delay}_1 + \text{Delay}_2}{2}
\]

Here, another assumption is made: that the delay from the Master to the Slave is equal to the delay from the Slave to the Master. In other words, the assumption is that there is no asymmetry – also a topic for later.

To calculate the offset of the Slave from the Master:

\[
\text{Offset} = \frac{(t_2 - t_1) - (t_4 - t_3)}{2}
\]

Or:

\[
\text{Offset} = \text{Master Time} - \text{Slave Time} - \text{Delay}.
\]

In other words, if you are frequency locked (FFO = 0) and you assume \(\text{Delay}_1 = \text{Delay}_2\) (no asymmetry), then any difference is due to error in time.

**PTP Real World Errors: Asymmetry and PDV**

As mentioned above, there are a couple of assumptions made: (1) there is no PDV and (2) no Asymmetry, and there is also a third assumption: (3) the timestamps are perfect. Of course, nothing is perfect and these three assumptions are the three main causes of time error in most PTP implementations. First things first, looking at the sources of it is generally found that the PDV is...
overwhelming due to queuing. For example, an egress queue at a switch output port might have no packets in it when the first Sync message traverses the switch, but it might have several the next Sync message. The difference in time due to waiting on those other packets the second time is the PDV due to that queue.

Asymmetry is caused by anything that has different delay from the Master to the Slave than from the Slave to the Master. These could be fixed (e.g., pipelines) or variable (e.g., queues) delays either internal to a network element (Figure 5) or external transmission delays (e.g., different lengths of upstream versus downstream fiber – see Figure 6).
Although it may seem at first that such delay differences do not matter, when you consider normal speed of light or electrical propagation delays, even a ten meter difference in length can mean about 50
nanoseconds of error — and since queuing can easily add microseconds of difference, you can see real problems emerging even in geographically small applications.

**PTP Basic Terminology**

Before discussing how PTP helps solve these problems, we need to introduce some terminology defined in the IEEE 1588 standards. First are the different types of systems: Grandmaster Clocks (GMCs), Ordinary Clocks (OCs), Boundary Clocks (BCs) and Transparent Clocks (TCs). There is also a definition for a “Management Node” but that’s not commonly referenced.

Grandmaster Clocks are technically Ordinary Clocks that are the source of truth for time in a PTP network. Ordinary Clocks are the endpoints in a PTP network that are receiving time. Boundary Clocks and Transparent Clocks are the switches in a PTP network whose job it is to pass time from the GMCs to the OCs with as little distortion as possible. Boundary Clocks are used to “break up” the overall PTP time domains by terminating the messages from a Master and regenerate time to a sub-domain. This might be to allow spanning across VLANs, creating sub-domains that require tight intra-domain timing, for example during clock switchovers, or any of several other reasons (e.g., profile conversion, message rate conversion). Transparent Clocks are used to pass messages transparently from a Master to a Slave – generally to limit time distortion and to generally simplify both message and time topologies.

A basic PTP topology can be seen in Figure 7, here:

![Figure 7: Basic PTP Topology](image)

There are also two types of delay calculation mechanisms defined: End-to-End (E2E) and Peer-to-Peer (P2P). The E2E delay mechanism calculates the delay all the way from a slave to the master, through any TCs (or even non-PTP bridges). The P2P on the other hand, calculates delay to all “peers”, meaning between TCs, BCs and OCs, which, by design, means that all peers must be “time aware” and support the P2P delay mechanism. Therefore, the fundamental difference between the two is that, E2E can accommodate non-“Time Aware” network elements whereas in P2P PTP networks all elements must support PTP (and P2P mechanisms). Therefore, in many “brownfield” applications, E2E must be used.

**Transparent Clocks and Boundary Clocks**

With both TCs and BCs there are two functions they perform: Time Transfer and Message Transfer. Time Transfer refers to how the device passes time: Does it do so transparently, trying to act like a wire? Or, does it terminate time from the Master and regenerate time from its own clock towards the OCs, acting like a Phase Locked Loop (PLL) filtering time? Message Transfer, on the other hand, is more of a networking function – referring to how messages are treated in the device: Does the device terminate the messages from the Master and regenerate new ones towards the OCs, perhaps at different rates with
spanning across multiple VLANs? Or, does the device simply pass the messages from the Master along, perhaps only replicating them to many OCs? Technically, according to the IEEE 1588 standard, TCs are transparent in both respects, but BCs are only required to be a boundary for messaging – Time Transfer can be either transparent or boundary. However, most BC implementations choose to be a boundary device in both respects.

When to use Transparent Clocks and Boundary Clocks

Given the above understanding of Time Transfer and Message Transfer and assuming that a BC is a boundary device in both respects (as is likely the case), the main reasons to use a BC are:

1. **Scalability** – breaking up E2E TC message hierarchy – which can cause issues at scale,
2. Tight local *correlation* – a BC creates a “local master” source of time for a sub-domain (e.g., a Cell/Area Zone or Machine) during “holdover” (loss of master),

As for the reasons to choose a TC:

1. Faster PTP convergence – when there is a network rearrangement, TCs allow for fast convergence since they do not run the Best Master Clock Algorithm (BMCA),
2. **Ease of implementation** – some devices may only support TC since a TC is a simple implementation – for example it does not require a PTP protocol stack,
3. Resistant to local oscillator drift - the 1588 standard mandates TCs use a feed-forward algorithm that reduces the effect of local oscillator drift – allowing the use of less expensive oscillators for otherwise equivalent performance.

PTP Profiles

IEEE 1588 really defines only the PTP protocol and the mechanisms to transport time – there are many aspects the standard leaves up to “profiles” to define as per their respective application. For example, PTP can run over multiple transport mechanisms – or “mappings”. For example, the “Default Profile” defined in IEEE 1588 uses UDP, but many other “profiles” have been developed that define transport over Layer 2, IPv4, IPv6 or others. Whether or not the messages are multi-casted or unicasted is also left up to the profile, along with message rates, various application specific extensions. The standard also does not define performance goals – that is, it does not specify how precise the recovered time must be. All of this is left up to application specific profiles.

Over the years, there have been many such profiles developed for different applications and market segments. In fact, there have been so many that an industry term “Profile Proliferation” has been adopted – generally used with negative connotations – with a movement to try to limit the number of new profiles and try to get broader adoption of the existing profiles. This is mainly due to the fact that supporting multiple profiles can be costly and complicated and there is often no reason for creating new profiles when existing ones would suffice – it just limits the broader adoption of compatible PTP implementations.

The ODVA standards organization was an early adopter of PTP, developing its own PTP Profile, namely the “CIP Sync” profile – which is fairly similar to the IEEE “Default Profile” with only minor additional requirements. The adoption (and subsequently only minor changes to) the Default Profile has proven to be beneficial in the broad adoption of the CIP Sync PTP profile. Many off-the-shelf PTP implementations can be purchased by vendors seeking to develop an ODVA certified CIP Sync device and, with only minor changes (e.g., System Time Step Compensation), this can be achieved with relatively little effort. Also, since the CIP Sync profile utilizes an E2E delay mechanism, this allows non-PTP devices to be used in a CIP Sync network.

Just to mention a few of the other PTP Profiles available, the ITU standards were also early adopters of IEEE 1588, developing PTP profiles for the telecom space. Namely, ITU G.8265.1 (frequency only),
G.8275.1, and most recently G.8275.2. Probably of most interest to the industrial space (other than CIP Sync) is the IEEE 802.1AS PTP profile (aka, “Generalized PTP (gPTP)”), developed originally in the IEEE Audio-Video Bridging (AVB) Working Group for home and professional audio and video applications, it is currently being revised (802.1ASREV) for automotive and industrial automation applications. There are also industrial PTP profiles developed for Substation Automation (e.g., C37.238) that may be of some interest here.

Regardless of the profile chosen, the mechanisms by which the problems of Asymmetry and PDV are solved are basically in the same manner.

**PTP Solutions: Timestamping, et al.**

The problems of Asymmetry and PDV are the main reasons that NTP has so much error and it is the main reason PTP was developed as an alternative to NTP: to add mechanisms to help alleviate these sources of error. The primary mechanism for doing so is timestamping. With timestamping, packets are stamped as close to the ingress of a system (switch/bridge or endpoint) as possible and as close to the egress of a system as possible. Initial implementations of PTP back when the standard was first developed used software timestamping, which mitigated much of error source and addressed the need of applications that just needed some improvement over PTP. Today, most implementations are hardware timestamped, which allowed improvements of several orders of magnitude again over the software implementations.

Most improvements in PTP accuracy today are working to improve either timestamping accuracy, generally by moving the “Timestamp Reference Plane” closer to the wire (or fiber), or by improving the PTP servo (the PLL that recovers time), by either improving the hardware (e.g., better, more stable oscillator) or improving the servo algorithms (or both). However, there are more extreme applications where even more is being done to quantify, compensate for, or eliminate asymmetry and even introducing new mechanisms in hardware to those ends. One such effort in particular is the White Rabbit Project (Figure 8), an initiative from CERN (European Organization for Nuclear Research) originally used for timing of the Large Hadron Collider (LHC). This project aims to achieve:

- sub-nanosecond synchronization
- connecting thousands of nodes
- typical distances of 10 km between nodes
- Ethernet-based gigabit rate reliable data transfer
- fully open hardware, firmware and software
- multi-vendor commercially produced hardware

![White Rabbit Project](image_url)

**Figure 8: White Rabbit Project**
Probably not of much practical interest to most, but what is interesting are the levels of accuracy achievable with off-the-shelf hardware and by paying close attention to the sources of asymmetry, PDV and timestamp error – all of which can be applied to varying degrees to the Industrial Automation applications using PTP – potentially improving CIP Sync implementations.

What does it mean to me?

To understand what these sources of error mean in Industrial Automation applications, it might be good to look at some real data.

Isolated 16-Axis Star Topology

![Diagram of 16-Axis Star Topology]

Figure 9: Real World Industrial Automation Comparison

Given the machine shown in Figure 9, monitoring the clock recovered off an axes on Switch 3 and therefore seeing the time error due to PDV and asymmetry between the Controller (the GMC) and the axes (the OC). In order to show what the PTP mechanisms accomplish, clock jitter is measured both first at various traffic levels with PTP disabled and then with PTP enabled.
As can be seen in Figure 10, with PTP disabled the clock sees PDV due to queuing in the switches that is not compensated for by the PTP mechanisms. As can be seen, with PTP disabled, with no traffic loading there is no significant clock jitter. However, with an increase to 40% traffic loading, the clock jitter increases to an average of 3 microseconds with some packets seeing greater than 10 microseconds. When traffic is further increased to 80% loading, the average clock jitter moves up to 5 microseconds with peaks up to 12 microseconds. Then, remaining at 80% traffic loading and enabling PTP, the clock jitter is eliminated.

What this means at the application layer is that clock jitter translates to position error. To determine this position error, multiply your application speed by the average clock jitter to determine position error due to network jitter.

\[
\text{Position Error} = \text{Clock Error} \times (\text{RPM} / 60 \text{ secs/min})
\]

Given a machine with servo drives operating at 6000 RPMs, with clock errors from the previous example, with 80% loading and PTP disabled, the average position error would be:

\[
\text{Position Error} = 5 \text{ usecs} \times (6000 \text{ RPM} / 60 \text{ secs/min}) = 500 \text{ uRevs} = 3.141593 \text{ mRads}
\]

**Summary**

Time is fundamental to many Industrial Automation applications, especially in closed loop control applications, increasingly as the speed of the loops increase. There are many methods for distributing time – each with their pros and cons – with PTP having emerged as the method of time distribution best for Industrial Automation. PTP uses several mechanisms and methods to alleviate the affects of common TWTT protocol issues (such as PDV and Asymmetry). The network designers have to choose between BCs and TCs, E2E and P2P, and other such configurations depending upon the needs of the application. All of these choices will optimize the precision of time at the endpoints – ultimately allowing for the minimization of position error.
References:


The ideas, opinions, and recommendations expressed herein are intended to describe concepts of the author(s) for the possible use of ODVA technologies and do not reflect the ideas, opinions, and recommendation of ODVA per se. Because ODVA technologies may be applied in many diverse situations and in conjunction with products and systems from multiple vendors, the reader and those responsible for specifying ODVA networks must determine for themselves the suitability and the suitability of ideas, opinions, and recommendations expressed herein for intended use. Copyright ©2015 ODVA, Inc. All rights reserved. For permission to reproduce excerpts of this material, with appropriate attribution to the author(s), please contact ODVA on: TEL +1 734-975-8840 FAX +1 734-922-0027 EMAIL odva@odva.org WEB www.odva.org. CIP, Common Industrial Protocol, CIP Energy, CIP Motion, CIP Safety, CIP Sync, CompoNet, ControlNet, DeviceNet, and EtherNet/IP are trademarks of ODVA, Inc. All other trademarks are property of their respective owners.