Network Time: Using Ethernet for Synchronization

Read About

IEEE 1588 Precision Time Protocol

Time Sensitive Networking

Real-time Ethernet

Introduction

Many embedded systems need to keep track of time, consistently and precisely, across multiple devices. Synchronized clocks can be used to coordinate actions, such as activating motors in an industrial process, or to time-stamp sensor readings that must be combined and analyzed. Synchronization of clocks may also be used to partition shared resources (such as network links) in distributed systems with critical real-time requirements.

Various schemes are available for setting and synchronizing real-time clocks. Some methods make use of common peripherals on commercial hardware, but cannot achieve high precision. Other methods achieve high precision but require specialized hardware interfaces. The emergence of IEEE 1588 PTP as a standard feature of network interfaces on commercial embedded hardware provides an attractive option for high-precision synchronization over general purpose Ethernet networks.

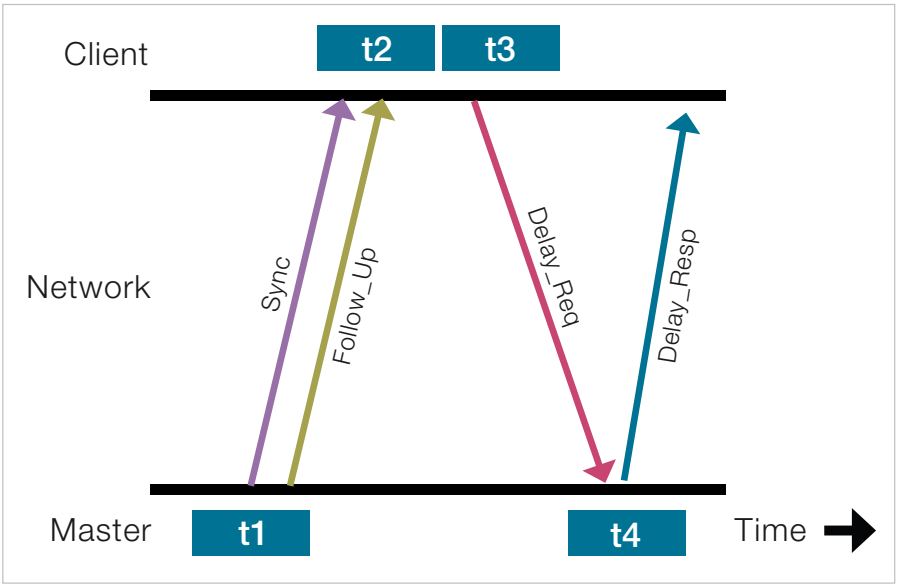

Figure 1: IEEE 1588 Sync Process

Info

curtisswrightds.com

Email

ds@curtisswright.com

Time in Embedded Systems

Today's embedded systems use a number of hardware and software mechanisms to keep time. These serve a variety of purposes that range from maintaining the file system within the OS, to calibrating high-precision sensor data. Since the requirements of these use cases vary, so do the implementations. For example:

- The "Real Time Clock" (RTC) uses dedicated hardware to maintain the time of day. Its time can be read and written by software. Most can also generate interrupts used for scheduling purposes. Typical implementations drift a few seconds per day. Often, this device uses battery backup to maintain time when the system is powered down.
- The "system clock" uses the CPU to maintain time of day, counting clock cycles. This is the "time" most commonly used by software applications through the gettimeofday() call. The initial time is typically obtained from the RTC at boot-up. Accuracy varies widely. Applications can be used to sync this time to match an external reference, or adjust the frequency.
- Hardware counters (for example, in an FPGA) use a shared clock reference to enable synchronization of multiple hardware modules that are connected together. This approach enables several modules to work in lock-step, but requires dedicated hardware and connectivity.

To be used to "tell time" each of these mechanisms must be set to an initial value - that is the "time of day". Time of day could be set manually by a user checking a "wall clock", or it can be set automatically to match some external reference – an authoritative clock outside the system. Common sources of "authoritative time" include atomic clocks maintained by national laboratories, and the clocks in GPS satellites.

Once they are set, clocks should also "keep time". Every clock will drift, running fast or slow. If clocks are not periodically synchronized, then they will diverge. On a simple computer with an RTC and system clock, there are two values for time of day – unless one is regularly copied to the other to keep them in line. In systems with multiple computers, each will have a different time of day unless they are somehow synchronized with each other.

In embedded systems that are connected using Ethernet and IP networks, protocols are available to support both getting time from an authoritative source and synchronizing multiple clocks within a system. By using the network to obtain and maintain time of day, designers can eliminate other single-purpose timing hardware and cables, reducing the size and complexity of their systems.

Terminology

In this paper, we will focus on clocks that maintain a value for time of day or measure time intervals. These are related to, but distinct from, the clock signals that oscillate at a particular frequency to drive digital logic.

For clocks in embedded systems, **accuracy** refers to how closely the time of a clock matches that actual time of day, compared to the authoritative clocks. Accuracy can be stated in terms of the error – the difference between the time of day on a clock, and the time of day on the reference clock.

Precision refers to the ability of a clock to measure or keep time over an interval. All free-running clocks will drift over time, running either fast or slow. Precision can be stated as a ratio, for example in the parts per million for typical clocks based on crystal oscillators.

Synchronization is the processing of setting a clock to a time that agrees with the time on another clock. Depending on the process used, there may be a synchronization error representing the difference between the time on the two clocks immediately after they are synchronized. When two clocks have been synchronized, they are said to be **isochronous**.

For some systems, it is also important to adjust clock signals so that they oscillate at the same frequency. When two clock signals are tuned to the same frequency, they are said to be **plesiochronous**.

Time of Day – Network Time Protocol

Network Time Protocol (NTP) allows computers to synchronize their clocks over an IP network. First deployed in the early 1980s, NTP is one of the oldest IP services still in use. Today most computers connected to the Internet use NTP to retrieve the correct time of day from a time server.

The many public NTP servers on the Internet typically get their time from one of the authoritative servers connected to atomic clocks. In turn they can relay time to other NTP servers or NTP clients running on computers.

Clients who set their time using NTP are synchronizing with a highly accurate source; however, the quality of the synchronization may be relatively low. To synchronize with an NTP server, a client will send a request, and wait for

the reply. Both the request and reply may need to traverse several hops in the network, being delayed by switches and routers in the path. These delays can add up and when a client receives the time of day response from an NTP server, it may already be hundreds of milliseconds old.

NTP attempts to adjust for the delay in the network by noting how long it takes to receive a response from the server. Since the round-trip time in the network time is known, the client can estimate that the network delay of the reply is half of that time. However this approximation is far from precise. Congestion in the network can cause a variable delay and the latency of the send and receive paths may be different. As a result, NTP performed across the public Internet can typically achieve synchronization accuracy of only tens of milliseconds.

NTP can also be used on a local network or within a modular embedded system. Here the transmission delays are significantly less, and also more predictable – particularly if the packets are only transiting a single switch without congestion. In this case NTP may be able to achieve synchronization error of less than 1 millisecond between client and server within the system. The accuracy of the time of day, however, will be limited by the synchronization of the server with an authoritative source.

Global Time – Using GPS

Accurate time is a fundamental element of the Global Positioning System (GPS). Each satellite in the constellation uses an atomic clock to maintain an extremely precise time of day, which is then broadcast to receivers. Receivers use the minute differences in the time received from different satellites to triangulate their position on or above the earth.

Since GPS receivers use a highly accurate time of day in the process of calculating their position, they can also serve as a useful reference clock. Many receivers are capable of outputting the time of day over a serial port for use by a computer client. Since serial ports are rather slow, this is typically only useful for synchronization to within a few milliseconds. To allow clients to benefit from the full precision of the GPS receiver's calculated time, another interface is required.

To enable high precision synchronization, many receivers will also provide a so-called 1PPS (one pulse per second) output. As its name suggests, this interface produces a pulse once per second, at the top the second. Using the edge of this pulse to reset the fractional seconds part of the time of day, a clock can be synchronized with the highly accurate GPS reference with an error of 1ns or less.

In embedded systems, a GPS receiver can provide an extremely accurate and precise clock solution. However for individual processors in a system to take advantage of this reference, they must have dedicated hardware that can make use of the 1PPS signal. In addition, the serial port and 1PPS must be routed to each client device that will synchronize to the GPS. In larger systems, this can involve significant complexity and cabling.

Synchronization over Ethernet - IEEE 1588 Precision Time Protocol

To address the limitations of NTP and GPS for synchronization of systems on a local network, a new approach was developed: Precision Time Protocol (PTP). Like NTP, PTP achieves synchronization using the same network connections used for data connectivity. However with PTP, typical devices can achieve sub-microsecond synchronization. And when all devices in the network provide full hardware support for the latest PTP standard, synchronization within a few nanoseconds is possible.

PTP is defined by the IEEE 1588 standard. Originally standardized in 2002, PTP received a major update in 2008. These two versions are often referred to as PTPv1 (1588-2002) and PTPv2 (1588-2008). Since the 2008 standard re-defined the format of the PTP messages, the two versions are incompatible; many implementations will support only one or the other.

Like NTP, PTP sends its timing messages over standard Ethernet/IP network links. Synchronization messages carry a time value that can be used to set the receiver's clock. However PTP differs from NTP in several respects.

NTP is a client/server protocol where the roles must be explicitly established. Client devices must be configured with the name or address of the server they will use, and the server must be running the NTP service. In PTP, a different model is used. Most devices will operate as "ordinary" clocks that are not explicitly configured as a server or client. Instead, ordinary clocks will negotiate with peers on the network to select one of the clocks to serve as the "master". The "best master clock" algorithm uses information about the quality of each clock based on information such as the quality of its oscillator and whether it is synchronized to an authoritative external reference.

Once a master has been established using the master election algorithm, all other ordinary clocks on the network

segment will synchronize to that master. The master will send regular “sync” messages to all the clients on the network, and they will use the time in the message to adjust their local clock.

In more complex networks, there may be multiple network segments, each with a local master. In this case, special handling is required to bridge between the different segments. This role is performed by a “boundary clock”. Since these more complex networks may also have multiple master clocks, the clock selected to be the root clock is termed the “grandmaster”. In most embedded systems, all the devices exist on a single network segment and a single master is sufficient. However the term “grandmaster” is occasionally used to refer to a computer specifically designed to act as the timing reference, for example because it is connected to a GPS receiver or includes a high-precision clock device.

As in NTP, PTP devices will attempt to measure the delay in the network so they can account for it when a sync message is received from the master.

Since PTP is typically deployed on local networks, this delay is generally smaller and more predictable than when synchronizing using NTP over a WAN. However variable delays due to queueing and processing are present even on high-performance Ethernet switches. To address this, the PTPv2 (IEEE 1588-2008) standard introduced a major new concept – transparent clocks.

In PTPv2, the switches that forward PTP messages between ordinary clocks can measure and report the delay that they introduce to the path. Switches note the time when they receive a PTP packet on an ingress interface, then note the time again when the packet is transmitted on an egress interface. The switch can then inform the PTP message recipient of the delay introduced by the switch. In this way, the network delay can be fully eliminated as a source of synchronization error. Switches that are PTP-aware and measure their transit time are said to be performing the “transparent clock” role.

Depending on the hardware a switch uses to process PTP packets, it can use one of two methods to act as a transparent clock. With the “1-step” method, the sync message from the PTP master is updated by the switch as it is sent out of the switch. When the ordinary clock receives the sync message, the network delay has already been calculated. Since this method requires specialized hardware in the switch that modifies packets in the egress path, some switches will instead use a “2-step” method. Instead of updating the sync message directly, the switch will send a second, “follow-up” packet which communicates the delay measured between receipt and transmission. The recipient can use this second packet to adjust for the network delay. In either case, the switch must have hardware timestamping to measure the transit time. For this reason, switches with silicon that supports the 1588-2008 transparent clock feature have only become available in recent years.

Implementing PTP in Embedded Systems

Implementing PTP in embedded systems can be as simple as installing a software client that implements the protocol. A reference implementation – ptpd – is available for many platforms. However to achieve the highest levels of synchronization, designers will need more than just software.

Processing of PTP packets in hardware is a common feature in most of the latest PHY devices and Ethernet adapters. These devices maintain a local clock and will use PTP sync messages to update the hardware clock directly. The time-of-day value stored in the network device can then be used by software directly, or used to update other clocks (such as the system clock or RTC).

When combined with an appropriate driver, support for PTP in networking devices provides application software with access to a high precision clock that is synchronized to other modules in a system. Software access may not be sufficient, however, for applications that require data to be time-stamped with high precision. Software that needs to read and write registers or respond to interrupts may be too slow.

In such cases, specialized hardware may be required (for example in an FPGA) that can sync with the PTP clock directly. For this reason, many PTP-capable adapters and PHYs will output a hardware signal (similar to the GPS 1PPS) that can be used to drive timing-aware hardware.

Synchronization for Networking - TSN

One of the techniques used to avoid congestion in time-sensitive Ethernet systems is time-division multiplexing, or transmission scheduling. With this approach, time slots are assigned to each of the endpoints in the system to ensure that only one station will be transmitting at a time. By taking turns, in a round-robin schedule, the system can ensure that there is never a collision or congestion. Several specifications for time-sensitive networking (TSN) are being

Author

Andrew McCoubrey
Product Marketing Manager
Curtiss-Wright Defense Solutions

developed by the IEEE under the 802.1 standard. These extend the AVB (audio-video bridging) standards previously developed to support real-time multi-media production and distribution.

For endpoints to stick to a schedule and avoid transmitting at the same time, all endpoints in the system must be synchronized to a common time-of-day. PTP provides a synchronization solution that can enable a high-precision shared time base, and is a critical element underlying many of the TSN standards.

In a system where all the endpoints have synchronized time bases, and agree on a TDM schedule for when they are allowed to transmit, congestion can be completely avoided. This technique is used in several of the deterministic Ethernet technologies, but can also be implemented by any system designer that has control over all the Ethernet devices on a network.

For additional discussion of Deterministic Ethernet and Ethernet for Real Time, refer to our white paper: Ethernet for Real Time Embedded Systems.

Conclusion

Maintaining accurate time is critical for many embedded computing applications. The IEEE 1588 “Precision Time Protocol” provides a mechanism for synchronizing connected systems using the over the Ethernet network.

Curtiss-Wright has a long history of designing embedded computing products for the most demanding environments, and over a decade of experience developing Ethernet switches, providing critical connectivity for sensor processing, avionics and mission systems. Contact Curtiss-Wright to learn more about our networking products.

Learn More

Read more: Networking Technology

Product Guide: Switching and Routing

Product: Networking Cards

W110.0617