White Paper

Precision Clock Synchronization

The Standard IEEE 1588

Table of contents

Precision Clock Synchronization
The Standard IEEE 1588

Precision Clock Synchronization
The Standard IEEE 1588

1 Introduction

Precise time information is especially important for distributed systems in automation technology. With the Precision Time Protocol (PTP) described in IEEE 1588, it is possible to synchronize distributed clocks with an accuracy of less than 1 microsecond via Ethernet networks for the very first time. The demands on the local clocks and the network and computing capacity are relatively low.
This White Paper provides you with an overview of the application possibilities and function of the Precision Time Protocol. Further technical information can be found additionally in the annex which explains details of the functional principle and hints on implementing PTP in devices.

2 Why time synchronization

We use clocks to synchronize ourselves with persons or processes. The necessary accuracy of the clock depends on the application. Anyone wanting to catch a train has to have his eye on the clock to within a minute. In competitive sport, a hundredth of a second can be decisive and drives in a packing machine need a synchronization in the microsecond range.

Many technical systems have a sense of time. An implicit system time exists when there is no actual clock and the timing behavior is determined by processes in the hardware and software. This is often sufficient in a lot of systems. An implicit time system is implemented, for example, by regular trigger events to every user which indicate the beginning of a unit of time and then trigger the appropriate actions.

The system time is explicitly available when it is represented by a clock. This is often necessary in complex systems especially. This decouples the communication from the execution. But not every clock is exact. Now and again it has to be checked whether the deviation is tolerable and whether the clock needs to be corrected. Communication between the individual clocks is necessary for this.

Two effects are in evidence when setting or synchronizing clocks: independent clocks initially run at an offset for one thing. To synchronize them, the more inaccurate clock is set to the more accurate one (offset correction). Another thing is that real clocks do not run at exactly the same speed. Therefore, the speed of the more inaccurate cock has to be regulated constantly (drift correction).

3 Previous solutions

There have previously been different ways to synchronize distributed clocks through a network. The most common of these are the Network Time Protocol (NTP) and the simpler Simple Network Time Protocol (SNTP) derived from it. These methods are widely distributed in LANs (Local Area Networks) or in the Internet and allow accuracies into the millisecond range. Another possibility is the use of radio signals from GPS satellites. However, this necessitates relatively expensive GPS receivers in every clock as well as the appropriate antennae. This theoretically gives you high-precision clocks but the high costs and effort often prevent it.

Another solution is to send a high-precision time pulse (e.g. pulse per second signal) to every user on separate lines. However, this entails an enormous additional wiring effort.

This is where the Precision Time Protocol (PTP) described in IEEE 1588 comes in. It has been developed with the following aims:

- Synchronization accuracy in the sub-microsecond range
- Minimum requirements of the processor performance and network bandwidth which enables it to be implemented on simple and low-cost devices
- Low administration effort
- Use via Ethernet networks but also via other multicast-capable networks
- Specification as an international standard

4 PTP applications

The idea for PTP was born at the end of the 90s in the USA at Agilent Technologies in the field of measuring technology. The process principle developed there was submitted to the IEEE as a suggestion and created the basis for the IEEE 1588 standard. At the end of 2002 PTP was passed as a standard under the name of "1588™ - IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems". In addition, PTP was also adopted as an IEC standard in May 2004 and was published under the name of IEC 61588.

PTP is arousing interest in many different applications. In automation technology, PTP is in demand wherever processes need to be synchronized exactly. Here Motion Control is an important field of application in the broadest sense. Because PTP helps to synchronize drives in a robot or a printing, packing or paper processing machine for example. Interactive robots are also connected by high-precision clocks or whole machine or plant parts are linked closely by PTP so that the processes that run can be synchronized exactly. Clocks running synchronously in every component enable distributed structures to be set up and the processes to be decoupled from the communication and processing of the control commands.

For this reason, the time synchronization according to IEEE 1588 has since become a part of almost all future real-time automation protocols. CIPsync, part of the Ethernet/IP frameworks of the ODVA, relies totally on PTP for Motion Control applications. PROFInet (PNO) has PTP as a synchronization protocol on its roadmap and ETHERNET Powerlink (EPSG) will also use PTP for synchronizing real-time segments.

Also, many companies are working on the evaluation and implementation of PTP outside of automation technology. Wherever measured values are detected and need to be put in relation to each other, PTP is a popular solution. In energy distribution systems, parameters such as currents and voltages are measured in distributed sensors, linked centrally and evaluated. Turbine controls use the PTP protocol to set up even more efficient plants. And, for monitoring processes, decentrally detected events are marked with precise time stamps and transferred to the control station for logging and analysis.

PTP is used in measuring technology for correlating decentrally detected physical variables, for example in high-frequency measuring technology. Geo-scientists use PTP to synchronize seismic measuring instruments over great distances and to be able to localize earthquake epicenters more exactly. In telecommunications, PTP is being considered for synchronizing networks or supplying mobile radio base stations with precise time pulses. There is also interest in time synchronization in accordance with IEEE 1588 in the fields of safety technology, automotive technology or military applications.

All these applications are currently in the development or prototype stage. However, in the next few years, many of these applications will come onto the market in regular products and PTP will find wide distribution.

5 The functional principle of PTP

PTP knows two types of clocks, masters and slaves. A clock in a terminating device is known as an ordinary clock, a clock in a transmission component like an Ethernet Switch as a boundary clock. A master which is controlled ideally by a radio clock or a GPS receiver, synchronizes the respective slaves connected to it.

The synchronization process is divided into two phases. First the time difference between the master and the slave is corrected, this is the offset correction. To do this, the master sends a synchronization message – SYNC message – with an estimated value of the time cyclically to the connected slaves. Parallel to this, the time at which the message leaves the sender is measured as precisely as possible, if possible by hardware support directly on the medium. The master then sends this actual exact transmission time of the corresponding sync message to the slaves in a second message - follow-up message. These also measure the reception time of these messages as exactly as possible and can correct the correction value (offset) to the master from it. The slave clock is then corrected by this offset. If the transmission line were to have no delay, both clocks would be synchronized.

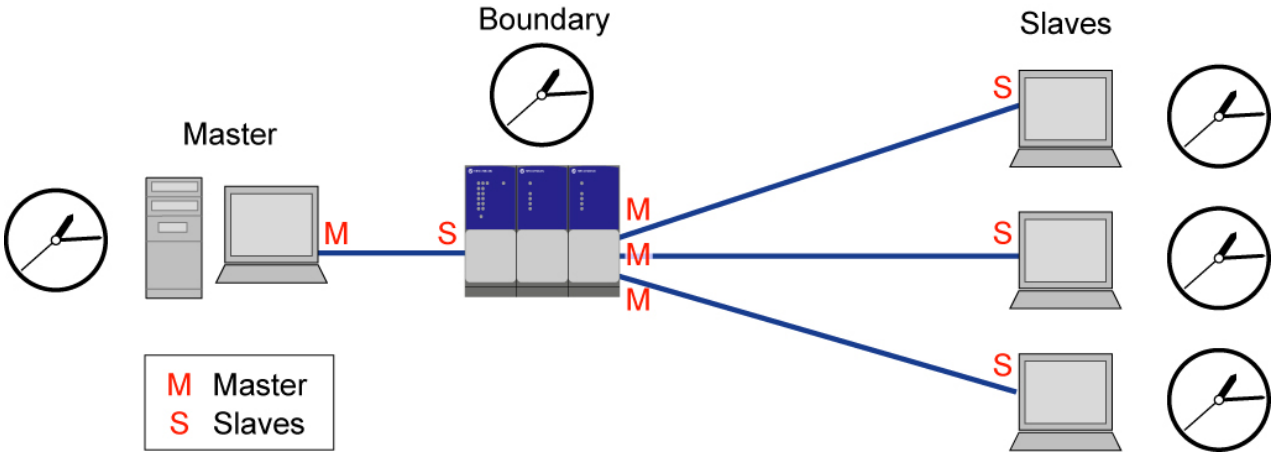


Figure 1: Boundary clock switches works as slaves in relation to the master clock and supply the other connected slaves as a master.

The second phase of the synchronization, the delay measurement, determines the run time between slave and master. This is determined by so-called Delay Request and Delay Response messages in a similar way and the clocks adjusted accordingly.

However, it is a symmetrical delay between the master and slave, i.e. same values for there and back, that is decisive for a delay measurement and its accuracy. This is satisfied practically by a direct connection with a length of cable. However, if there are network components such as switches or routers between the users, this is no longer the case. The so-called boundary clock was defined for such devices for this reason. Such a switch contains a clock which is synchronized to a connected master and then behaves as

a master at every other port and synchronizes the connected slaves. In this way, the synchronization always takes place in a point to point connection which exhibits very symmetrical run times and practically no jitter. Therefore, wait times or jitter when passing through the switches have no influence on the accuracy of the synchronization.
A detailed description of the functional principle can be fund in annex 1.

6 Implementation of PTP

If the Precision Time Protocol is to be used in a system, the PTP protocol stack must be implemented. This can be implemented very compactly and only makes minimum demands on the processor performance and the network bandwidth. This is very important for the implementation in simple and low-cost devices. PTP can even be implemented without any trouble in embedded systems with simple 16 or 32 bit microcontrollers. The only requirement for achieving a high precision is as exact a measurement of the PTP message transmission and reception times as possible. This must take place very close to the hardware (e.g. directly in the driver software) and with as great an accuracy as possible. In implementations as a purely software solution, therefore, the architecture and performance of the system restrict the attainable accuracy directly.

When using additional hardware support for time stamping, the precision can be increased considerably and effected virtually independently of the software. A little logic is necessary for this which can be integrated, for example, as FPGA or in ASICs directly at the network input.

If PTP is used through Ethernet networks, special attention must be paid to the network infrastructure. Since hubs have almost no influence on the accuracy of the protocol due to their practically constant throughput time, the run times must be taken into account when using switches, as already described in chapter 5. PTP measures the run times in the network and measures the clocks accordingly. However, run time variations, as they always occur in switches, lead to inaccuracy.

Since switches save the received data packets completely and queue effects can considerably delay the transmission under certain circumstances, great fluctuations may occur here. At low network load, this effect hardly has an influence, but at greater network load or in temporary load situations, this can considerably worsen the synchronization accuracy.

This is counteracted by using so-called boundary clock switches. These contain their own PTP instance which operates as a PTP slave in relation to the connected master clock and is therefore exactly synchronized by it. In relation to the connected terminating devices, the PTP slaves, every single switch port then operates in turn as a PTP master and synchronizes these slaves with its internal time. This compensates all run time fluctuations and wait times in the switches and enables maximum accuracy to be achieved even through larger Ethernet networks.

7 Results

Hirschmann was one of the first manufacturers to implement and optimize the Precision Time Protocol. A software stack was developed which implements the protocol very efficiently and a chip which supports the process for maximum precision.

The implementation achieves accuracies of about 5 to 50 us as a pure software solution. The precision is even higher with the hardware support for the time stamp detection.

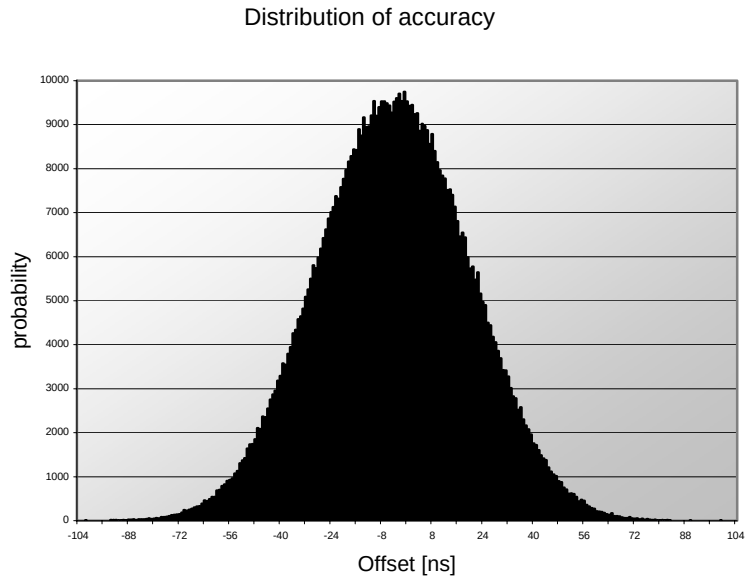


Figure 2: The Precision Time Protocol achieves accuracies of below 100ns with hardware support

In a pilot system in which several ordinary clocks are connected by an Ethernet switch with boundary clock function, a typical accuracy of +/-60ns was achieved, practically independently of the network load or workload of the CPU. Variants of the PTP boundary clock are available for the Industrial-Ethernet switches of the MICE family from Hirschmann.

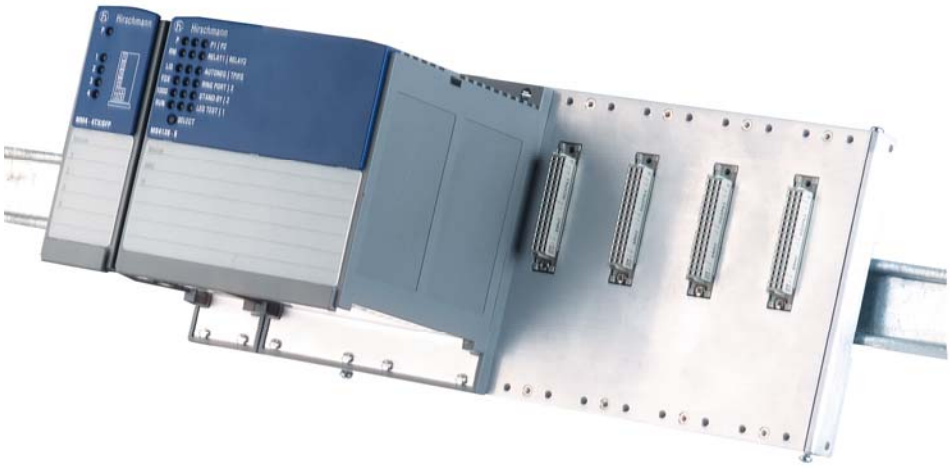


Figure 3: The Industrial Ethernet switches of the MICE series support the Precision Time Protocol according to IEEE 1588

8 Cooperation between Hirschmann and the Zurich University of Winterthur

In cooperation with the Zurich University of Winterthur (ZHW), Hirschmann has been working for some time of the further development and optimization of IEEE 1588. In the scope of this cooperation, the ZHW offers interested users support in the implementation of PTP ordinary clocks. This includes software stacks and VHDL designs which are based on the technology developed by Hirschmann and additional services such as evaluation kits, training or design support. For further information, see the web site "http://ines.zhwin.ch/ieee1588/index.html".

9 Further developments

IEEE 1588 has a wide acceptance on the automation and measuring technology market. But the method's potential is also of interest to other branches. To meet additional requirements, the P1588 project was started in February 2005 in the IEEE Committee with the aim of extending the standard in the following directions:

• Greater accuracy: The aim is to achieve accuracies in the nanosecond range and below. The prerequisite for this are higher quality oscillators, shorter sync intervals, higher resolution of the time stamp and better symmetry.

• Fail safety: If the master clock fails, the reconfiguration must take place in such a way that the synchronization is not impaired.

• SNMP Management: SNMP is the most widely used network management protocol in Ethernet networks. A PTP MIB should therefore be specified.

• Different Ethernet headers: The inclusion of Tagged Ethernet Frames (VLAN), IPv6 packets and fields for protocol extension should be enabled.

• Transparent Clock: Topologies with linear and ring structure are often used in automation technology. The rowing of boundary clocks which each represent their own control circuit is a problem because of the accumulation of errors. The Transparent Clock is suggested as an alternative. This implements an end-to-end control (unlike the cascade of single regulators). The switches measure the throughput time of the PTP messages and manipulate the time stamps contained in the messages accordingly. With the aid of the Transparent Clock, the fast reconfiguration of the network in the case of an error is possible.
• Application in other network types: A few details such as message format, addressing and reference point of the time stamp must be specified for this. Candidates are WLAN, DeviceNet and Telekom networks.
• Security: The synchronization mechanism must be protected against undesirable access in open networks.

10 Summary

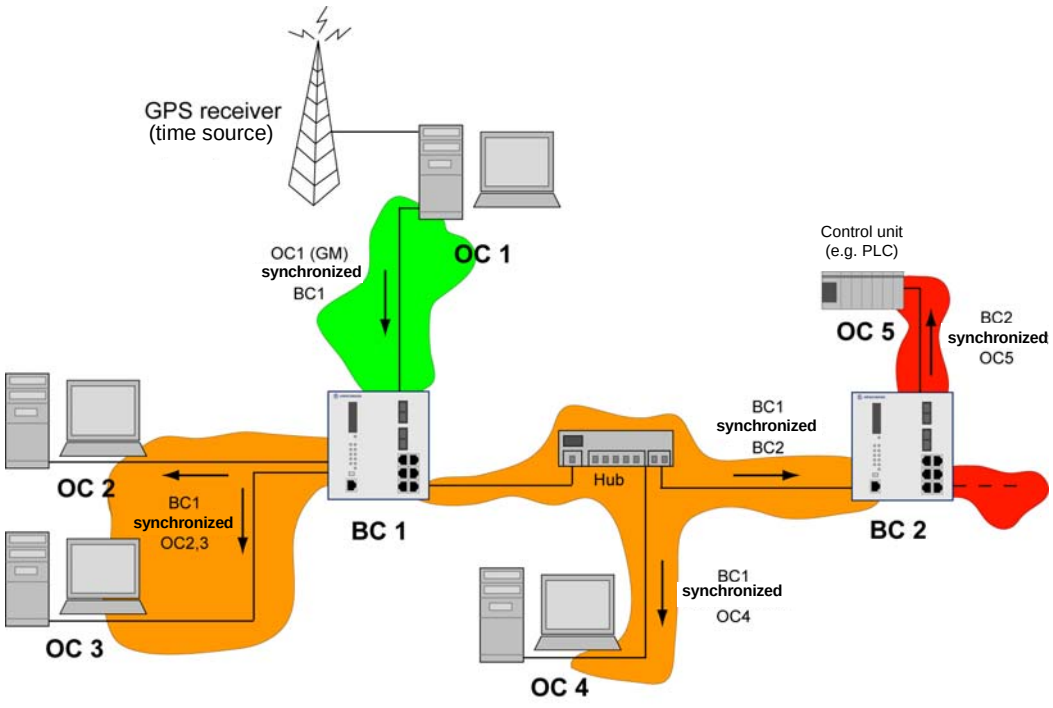
PTP has already proven its capabilities in the lab and is now ready to make the leap to practically applications. You can be assured that PTP will find further distribution over the next few years and that many applications can be implemented much easier and more efficiently than with existing technology.

Further information:	
http://www.hirschmann.de	Hirschmann Automation and Control GmbH
http://ieee1588.nist.gov/	IEEE 1588 Standardization Group
http://ines.zhwin.ch/ieee1588/	Zurich University of Winterthur, PTP Stacks and Services

Annex 1: The PTP protocol in detail

The IEEE 1588 standard describes the Precision Time Protocol (PTP) including the basic synchronization algorithms. An initialization phase is described additionally which selected the most accurate clock in the network as the dominant system clock by means of the so-called Best Master Clock algorithm.
The behavior of the clocks is described in the standard on status machines, data types and their representation are defined in addition. For management and configuration of the clocks in the network, the standard defines its own simple management protocol via which the user can request the states of all the participating clocks on the one hand but can also configure them on the other. For example, he can determine which clock he prefers to assume the function of grand master.
The procedures for regulating the clocks and the user interface (API) through which the clock can be addressed by the application are not specified.

PTP is based on IP Multicast communication and is not restricted to Ethernet but can be used in principle on any multicast-capable bus system. The multicast communication offers the advantage of simplicity because no IP address management needs to be implemented on the PTP nodes. PTP can be scaled for a high number of PTP nodes specially by using IP Multicast.



OC: Ordinary Clock (source or sink, 1 port)
BC: Boundary Clock (switch or router, 1-n ports)
GM: Grand Master (synchronization source)

Figure 4: PTP synchronization domain

Time synchronization, the algorithm

For time synchronization, the time of the master is reported to the slave as accurately as possible in PTP. The trick is to compensate all the processing times and run times by this method so that the most optimum possible synchronization can be established. The synchronization is divided into two processes for this.

First the masters and slaves correct their time difference with the offset measurement. In this offset correction, the master sends a clear synchronization message - the SYNC message - to the connected slave cyclically at defined intervals (every 2 seconds in the basic setting). This sync message contains the current time of the master clock. However, since the reading out of the clock, the processing of the log, the handling of the communication stack and the transmission of the data via the Ethernet controller require an undefined time, the time information in the sync message is already out of data when it leaves the master. Therefore the actual transmission time of the message is measured as close as possible to the physical interface (ideally directly at the Ethernet port by hardware) and this information is sent to the slave by means of a second message, the follow-up message.

Upon reception of the follow-up message, the slave calculates the correction value (offset Θ) to the master under consideration of the reception time stamp of the sync message. The slave clock must then be corrected by this offset. These precise time stamps now render the time fluctuations of the protocol stacks - protocol jitter - above insignificant.

If there were no delay on the transmission line, both clocks would now be in sync.

The second phase of the synchronization process determines the delay time – (the run time on the network) – between slave and master, the delay measurement. For this, the slave sends a so-called "Delay Request" packet to the master and determines the exact transmission time of the message. The master generates a time stamp on receiving the packet and sends the reception time back to the slave in a "Delay Response" packet. From the local transmission and reception time stamp of the master, the slave determines the delay time between the slave and the master. The delay measurement is made irregularly and at greater intervals (4 - 60 seconds) than an offset measurement. This relives the load on the network and especially on the terminating devices. However, it is a symmetrical delay ($T_{Delay\_Snd} = T_{Delay\_Ret} = T_{Delay}$) between master and slave that is decisive for a delay measurement and its accuracy..

Figure 5 illustrates the synchronization algorithm.

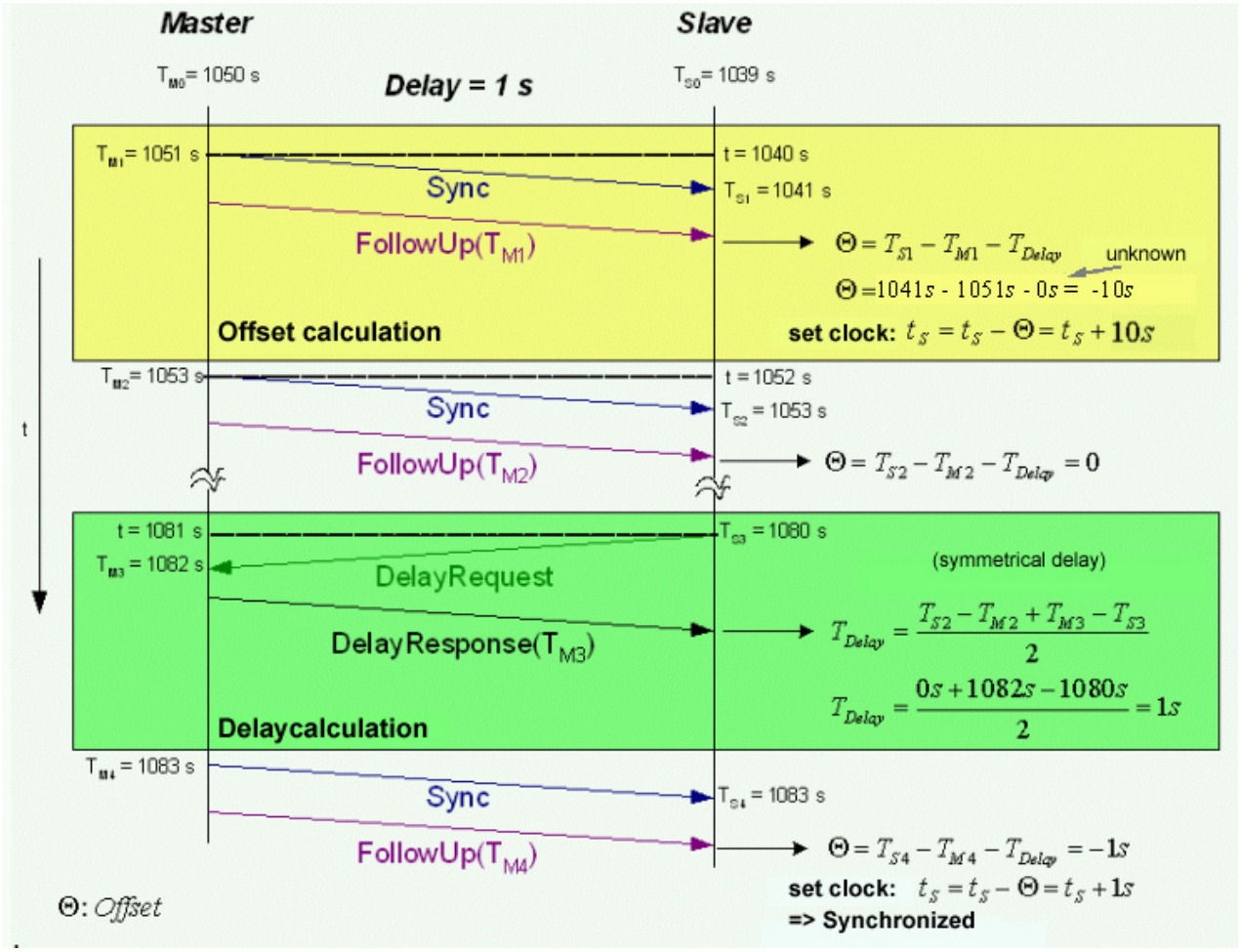


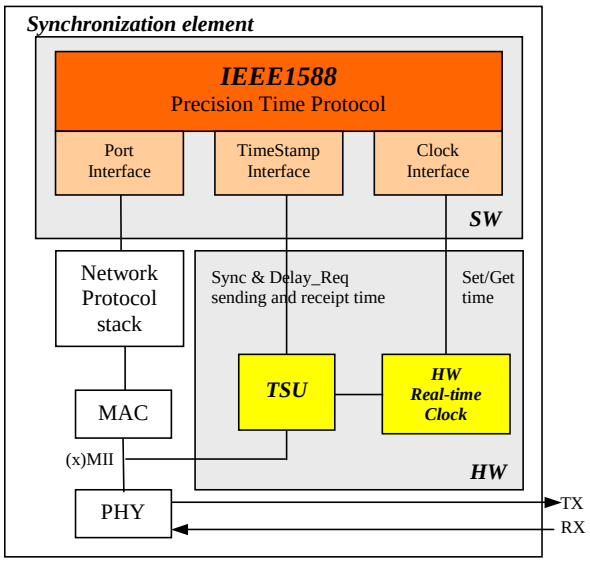
Figure 5: Synchronization algorithm

Automatic master selection – Best Master Clock algorithm

The Precision Time Protocol determines master-slave relations between communicating PTP synchronization elements. Determination of the states (master/slave)is the job of the Best Master Clock (BMC) algorithm which compares the properties (accuracy, stratum, drift, variance, ...) of the communicating clocks and derives the states for all local ports of the PTP element. Every PTP element runs the BMC algorithm locally and therefore determines the status of all local ports. This transfer the current properties of the master clock to the slaves cyclically in synchronization messages. The advantage of this algorithm is that the nodes doe not have to negotiate these statuses but can calculate them individually. This guarantees that the PTP network can be configured automatically in a tree structure starting with the best available clock, the grand master. A configuration with several or no master and unstable statuses are prevented by this algorithm.

Annex 2 – The implementation of PTP in detail

The special thing about the architecture of PTP is the separation of the time-critical part implemented in the hardware and the protocol decoupled from hard time conditions, the software part. The protocol therefore runs in a low priority process for example on a processor with low performance requirements.
The hardware unit consists of a high-precision real-time clock and a time stamp unit (TSU) for generating the time stamp. The software component implements the actual IEEE1588 protocol with the link to the real-time clock and the HW time stamp unit. Figure 6 illustrates the interaction of the hardware and software components of an IEEE1588 synchronization element.



TSU - TimeStamp Unit

Figure 6: Component layout of a synchronization element

Important for a generic portable implementation of the PTP protocol is a software component modeling which is as independent as possible from the operating system. To achieve this, Hirschmann has introduced three layers with different levels of abstraction in its solution. The ***protocol layer implements the*** operating system-independent Precision Time Protocol. The ***OS Abstraction Layer*** forms the interface between PTP and the used operation system. The functions - tasks / processes, semaphores, timers, sockets, etc. - offered by the operating system are integrated through the ***OS Layer***. The following figure shows the interaction of the individual steps.

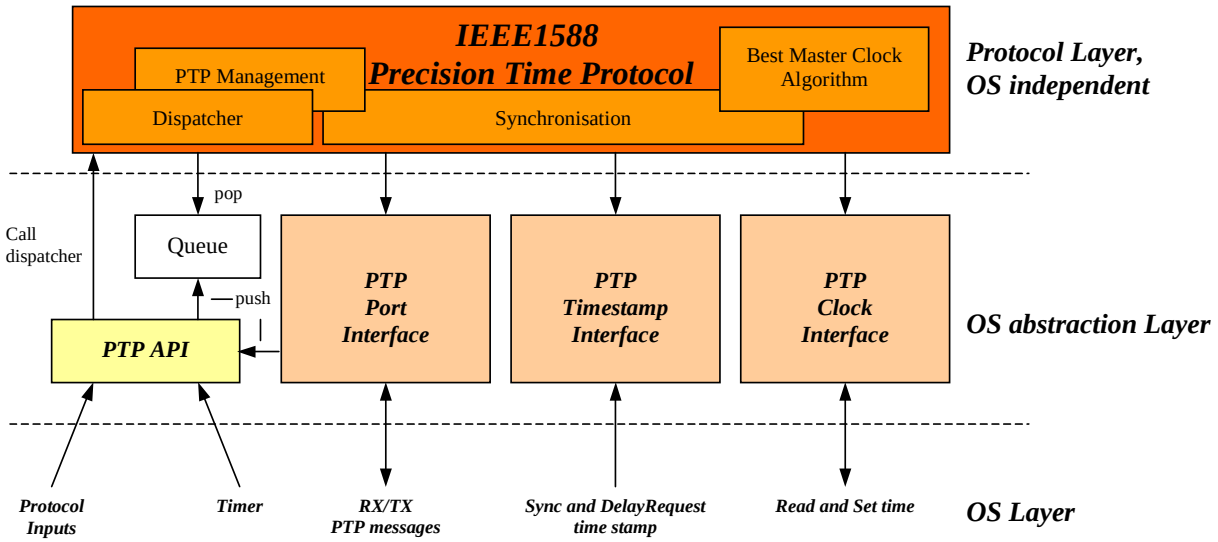


Figure 7: Interaction diagram

The top layer implements PTP for synchronization of clocks in a network and can be used on different communication elements (PC, Switch, Router, etc.). This is where the actual intelligence for synchronizing the individual communication elements is to be found. Within the ***protocol layer*** only ANSI/ISO C-conform functions have been used. This ensures that the log can be ported on different platforms without great intervention in the functional principle. A protocol dispatcher guarantees the atomic execution of functions in a single process. The communication between the protocol and the OS Abstraction Layer is made by a queue and three defined interfaces.

The middle layer encapsulates operating system-dependent functions which have to be adapted if necessary.

The ***Timestamp Interface*** provides the Precision Time Protocol with the detected time stamps for Sync and DelayRequest messages. Depending on the expansion stage (precision requirement), the time stamp is generated either by a HW unit (TSU) or by the software itself. "Software time stamps" are best generated in the operating system-dependent NIC driver (RX-ISR, send process) as close to the transport medium as possible.

The local clock is read and modified via the ***Clock Interface***. These functions may also have to be adapted depending on the platform. Applications which do not provide a real-time clock use the operating system clock or optimized solutions such as the nanokernel under UNIX derivates. In addition to control of the local clock, this interface also contains the control algorithms which are responsible for the quality [accuracy, stability, transient behavior, etc.] of the time synchronization.

The ***Port Interface*** is used to send ***or*** receive PTP messages. The IEEE1588 telegrams exclusively use UDP/IP Multicast packets and therefore enable them to be sent and received via the socket interface of the IP protocol stack. Time requirements can be neglected because the time stamp is generated directly at the transport medium.

The inputs to the protocol (configuration, diagnosis, PTP telegrams) run through the ***PTP API***.

Low porting effort when changing over to a different platform

This process has meant that only the bottom two layers need to be adapted when porting the protocol to a different platform which leaves the actual protocol (~ 90 - 95 % of the source texts) unaffected.

The software section of the described PTP reference architecture has been changed to object-based C. This has meant that the advantages of object-oriented analysis/design/programming could be applied despite using the C programming language: methodology, encapsulation, reusability, etc.

To check the easy portability of the software, Hirschmann has effected reference implementations of ordinary and boundary clocks under the platforms Linux, VxWorks and Windows.

The precision of a software implementation

After completing the work, it was first examined what precision can be achieved with a *purely software implementation*. Measurements both on Linux platforms and on Windows and VxWorks platforms gave a typical *synchronization accuracy* of ±10µs to ±200 µs.

The achieved accuracy is basically dependent on the used platform, the network load, system workload/internal bus load and the place where the time stamp is generated. The nearer to the hardware the time stamp is generated, the higher the precision.

Hardware for maximum precision

A software implementation is inadequate for synchronization accuracies in the sub-microsecond range, therefore the necessary time stamps must be generated in a so-called hardware time stamp unit directly on the transport medium.

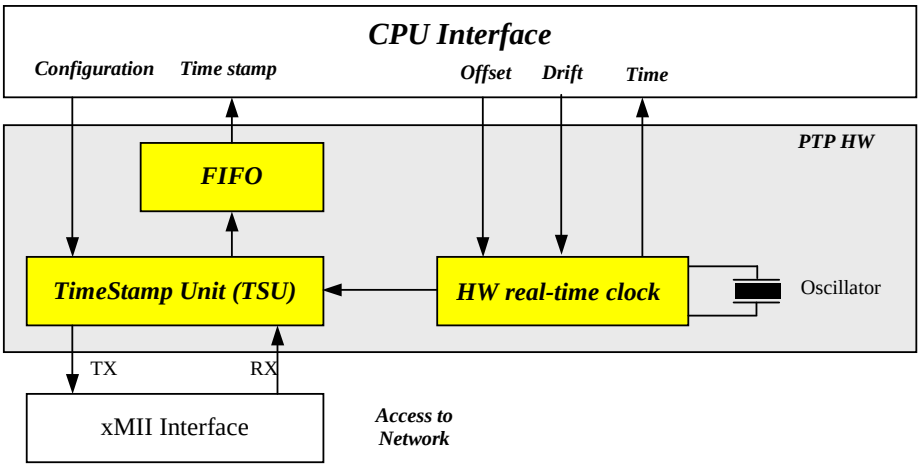


Figure 8: Hardware block diagram

The ***Timestamp Unit*** (TSU) has also been converted into hardware. It is based on the Media Independent Interface (MII). If the TSU detects a *Sync* or *DelayRequest* message, it notes the time of occurrence. In

addition, it extracts information for clear identification of the message. The TSU buffers this information in a FIFO and therefore enables the detected time stamp to be assigned to the corresponding PTP message later.

The absolute time of the synchronization element is managed by the ***HW real-time clock*** and serves as a high precision metronome. The HW clock counts oscillator ticks which the clock driver converts into seconds and nanoseconds. An integral part of the digital real-time clock is the offset (phase) and drift (frequency) correction unit. It enables the adjustment of a real-time clock to a reference clock in the sub-microsecond range. The separation of the phase and frequency unit enables the specific setting intervention of the control which achieves a fast convergence of the system. The system can therefore control phase jumps – caused by a change in the topology for example – independently of the drift correction unit. This has the advantages that the local clock does not have to change the frequency and a phase jump is soon compensated.

The ***offset compensation*** adjusts the absolute value of the local clock to the reference clock.

The ***drift correction*** causes frequency synchronization of the local real-time clock with the reference clock. The result is that the local clock "ticks" with the frequency of the master. The challenge is now to determine the relative drift δ of the local clock to the reference clock. The following equation can be assumed for the drift correction by way of simplification.

$$f(t)_{Slave} = f(t)_{Master} + \delta + \varepsilon_{Quatization}$$

Whereby $f(t)$ represents the present frequency of the appropriate clock and $\varepsilon_{Quantization}$ the quantization error.

The whole hardware can be implemented in a FPGA at the first attempt whereby the application requires about 20,000 to 30,000 Logic Array Gates.

The basic conditions for high-precision time synchronization

With regard to the synchronization accuracy, PTP defines the minimum requirements of a local clock and gives recommendations for improving the precision. The following aspects must be considered.

- Operating system and network ***Stack Jitter***

Jitter within a synchronization element can be avoided by time stamps in the hardware.

- ***Latency*** between master and slave

Symmetrical latency (send and return direction identical) between masters and slaves compensates the delay measurement. But asymmetry causes an unncompensatable time difference between master and slave.

- ***Jitter*** in ***network coupling elements*** (repeaters, switches, routers)

Jitter tests with network coupling elements have shown that the jitter of hubs and repeaters is negligible. If, however, you observe switches in high load or overload, you will see that these have a considerable delay jitter. Values up to one millisecond are no rarity. The jitter in a switch depends on the message length and the number of packets which are already buffered. Even if only one long packet has preceded the synchronization packet in the switch, this can cause a jitter of up to 125µs. Therefore it is recommended to

equip switches with the boundary clock function for a high-precision network. In this way, the user ensures that the synchronization is always made only by a point to point connection, i.e. by a physical connection, the jitter of which is negligible.

- ***Resolution*** and ***stability*** of the ***local clock***

If you consider the above aspects, the local clock of the synchronization element largely determines the accuracy of synchronization. The first limiting factor is the resolution of the local clock which is determined by the crystal frequency of the oscillator. However, the decisive aspect is down to the stability, especially the short-term stability of the oscillator. The quality of the synchronization accuracy in the transient state (drift compensated) is therefore dependent on the relative drift between the master and slave between two consecutive synchronization telegrams.

Regulation, the distinguishing feature

PTP describes the process by which master and slave determine their time difference (offset). But the protocol cannot set this variable directly. Different algorithms must be used to determine the offset and the drift or to verify the value. The higher the precision of the local clock in relation to the reference clock, the more these algorithms have to be improved. The verification of the offset could be achieved simply for example by comparing the new value with the previous one. When this then exceeds a certain difference, it is assumed to be invalid. However, it must be considered that something may really have changed and the new values should be accepted if they appears several times in succession.

The drift compensation (frequency synchronization) is rather more complex and takes place over a longer period of time.

The quality of the implementation of the control will differentiate the providers in future. The users can base their evaluation of a control on the following criteria.

- The transient behavior of a control characterizes its quality. The decisive variable here is the transient time. That is the time a local clock requires to synchronize to the reference clock (frequency and phase).
- Interference immunity and stability of a transient clock. If the clock is synchronized, the control should have a certain inertia in order not to counteract minimum outliers with a high setting intervention of the controller. Nevertheless, the control must respond quickly to permanent changes (topology change or master change) to restore the synchronization. This is achieved for example by an adaptive control.

Results

Two ordinary clocks implemented in Hirschmann Ethernet switches of the MICE family were connected in a first attempt in a laboratory set-up. A high network load was applied additionally with a generator for rigorous testing of the protocol.

In order to be able to test the difference between the local clock and the reference clock as application realistically as possible, both units were given a Pulse Per Second (PPS) signal output and connected to an oscilloscope (figure 9). The deviation of the two signals was measured and the frequency adjustment of the deviation represented very elegantly in this way.

The synchronization accuracy achieved was about ± 60 ns (max. jitter). The measurement was made over a period of 84 hours.

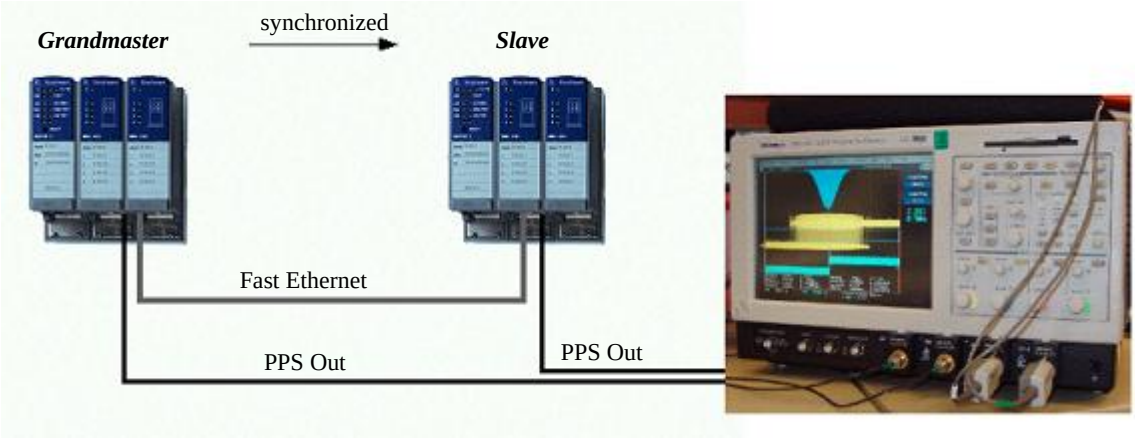


Figure 9: Measuring set-up

Figure 10 shows the frequency distribution of the offset values between the master and slave in nanoseconds. The standard deviations is 14.9 ns and the average value 4.2 ns.

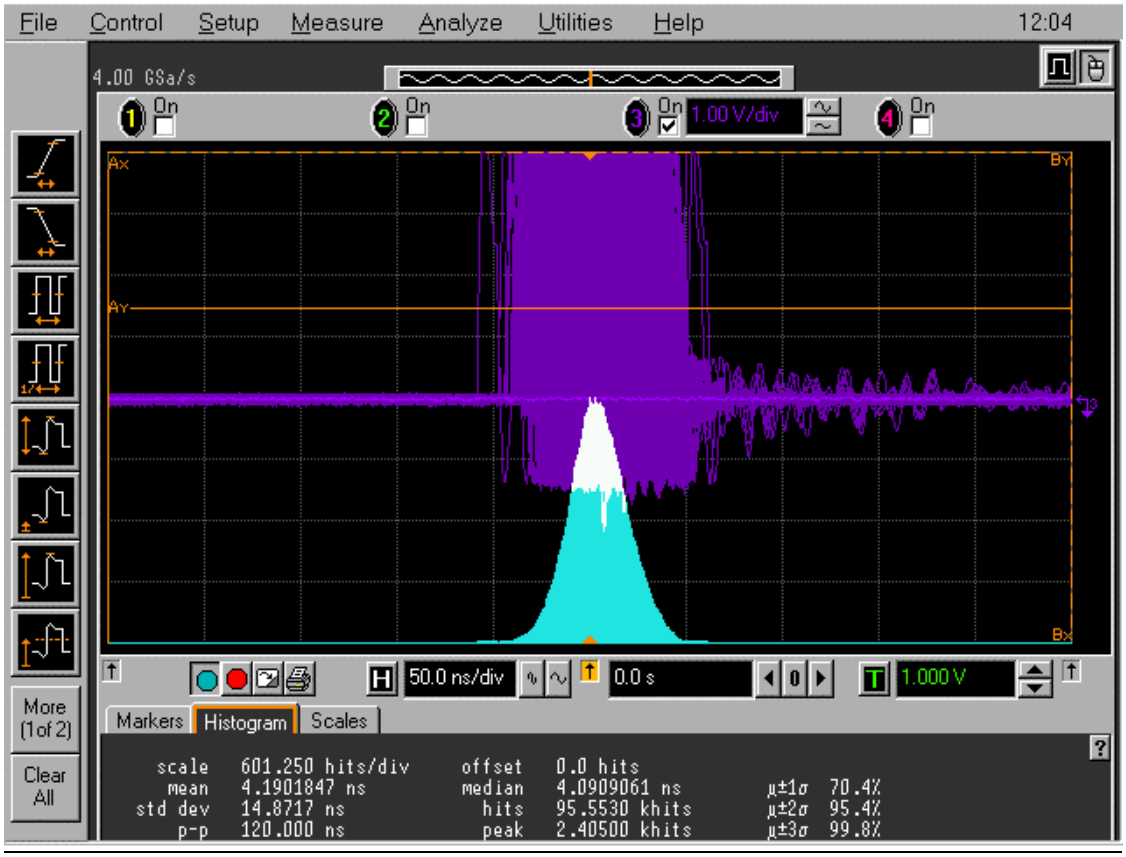


Figure 10: Achieved accuracy

The drift setting of the oscillators restricts the synchronization accuracy in the existing prototypes. A quartz frequency of 50 MHz (± 50 ppm) gives a resolution of 20 ns. The system can therefore only set the drift in the range of ±20 ns per second. If you now look at the relative drift between the local clock and the master clock during two consecutive synchronization telegrams, it soon becomes clear that the short-term stability of the oscillators has a decisive influence on the synchronization accuracy in the transient state.

Andreas Dreher, Dirk Mohl
Hirschmann Automation and Control GmbH
Neckartenzlingen, Germany