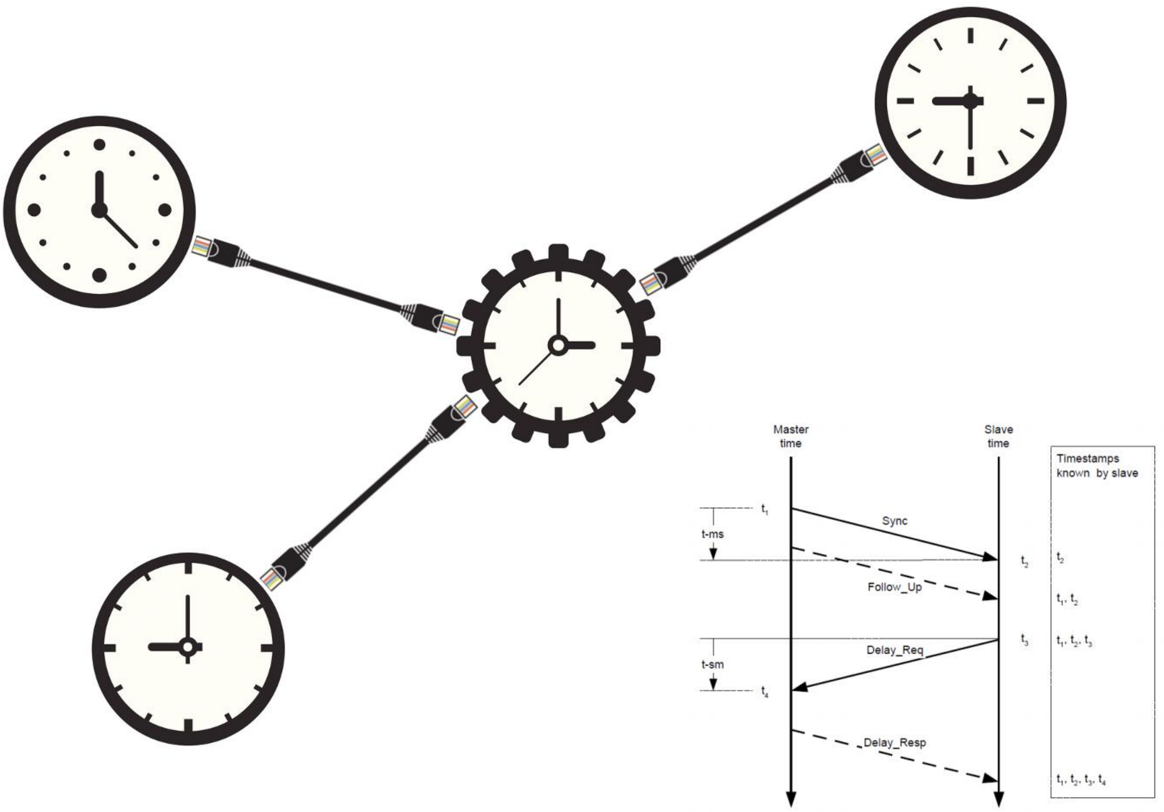
TSEP
Technical
Software
Engineering
Plazotta

Product Description IEEE 1588 Stack (PTP Stack)

IEEE 1588 PTP



The TSEP IEEE 1588 Stack is a software solution for the IEEE 1588 standard. With the help of the TSEP IEEE 1588 stack computers with the operating systems Windows, Linux and RTX64 (Interval Zero) can be equipped with a PTP stack.

General:

The time synchronization with IEEE 1588 has been codified since 2008 as an IEEE standard and is already used in various areas.

Previously, the use of this standard was always associated with exotic hardware, that is, implementations of network adapters in various FPGAs or embedded controllers. With the introduction of the Intel network chip families Intel I21x and Intel I35x, this standard is now available for the consumer market. Thus, laying the foundations for new projects based on consumer hardware.

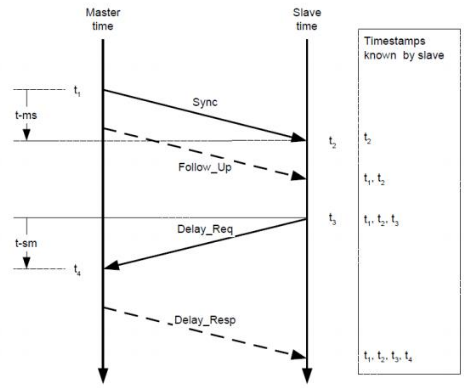
In 2016 TSEP decided to implement its very own IEEE 1588 stack, and in 2017 the first executable prototype was shown. By now the TSEP IEEE 1588 stack is available for multiple platforms and is already used by our customers.

The fundamental question that arises with every IEEE 1588 project is the accuracy with which the time synchronization must take place. The achievable accuracy usually depends on the hardware used, the topology and the control algorithm used. Modern IEEE 1588 implementations have the ability to define various control algorithms and easily exchange them.

The company TSEP has also defined the control algorithm as an independent module with defined interfaces. Thus, the user can simply define their own algorithm and bring this into the system and test it.

For example, when using IEEE 1588 to synchronize wireless LAN speakers, human hearing is the measure of accuracy. The human ear can recognize runtime differences from 10 µs. Thus, the achieved accuracy for the synchronization of the WLAN loudspeakers must be below 10 µs. But if metrological tasks are considered, other accuracies are required. Within metrology, measurements are usually triggered by triggers. As a rule, these triggers are signal changes (rising or falling edges, exceeding level values, etc.). These signals are transmitted via cable from the source to the meter. Thus, the transit times within the trigger cable is the authoritative target for accuracy. Assuming cable lengths of about 5 meters, which is rather generous, one can expect a running time of 25 ns (running time of 5 ns per meter). Thus, the accuracy of metrological problems would be on this scale. However, in the field of measurement technology, with the introduction of 5G technologies in mobile communications, this magnitude has shifted significantly downwards. For these technologies, accuracies in the subnano range would be desirable.

Technical Details:



The TSEP IEEE 1588 stack has been written completely in C++ and is based on modern coding guidelines. To be independent of a specific development environment, CMake was used as the preferred code management tool. Additionally the source code is strictly object oriented.

The TSEP IEEE 1588 stack can both act as ordinary clock ("client") and as master clock.

A master clock distributes its own timestamp with so-called sync messages over the used Ethernet connection, which will be received by connected ordinary clocks. An ordinary clock will respond and then start to synchronize with the master clock.

The TSEP IEEE 1588 stack supports one step as well as two step sync messages. The former embeds the timestamp directly into the sync message, which requires hardware support. The latter sends the timestamp in a separate follow-up message, which will work on any hardware lacking direct timestamping.
To facilitate precise time synchronization even in networks with non IEEE 1588 conformant devices (i.e. switches and routers), the TSEP IEEE 1588 stack uses so-called end-to-end (E2E) communication.

Once synchronization took place, the TSEP IEEE 1588 stack can also activate hardware triggers, given the used hardware does support it. With these synchronized triggers it is possible to initiate measurements on multiple devices simultaneously or trigger in specific intervals, i.e. to get a PPS signal.

The TSEP IEEE 1588 stack supports every management message defined by the IEEE 1588 standard.
These messages are used to retrieve clock information or manipulate clock settings.

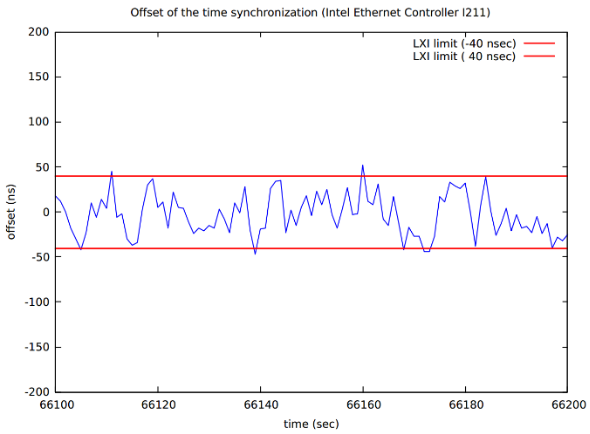
The TSEP IEEE 1588 stack can be executed as a standalone application and is thusly capable of running directly on the requested device. However a supported network interface is required (i.e. Intel I21x).

Supported operating systems are Windows (7 and 10) and Linux (Ubuntu 16.04). Additionally the TSEP IEEE 1588 stack supports the real-time operating system RTX by IntervalZero. The TSEP IEEE 1588 stack combined with RTX64 by IntervalZero is ideally suited for every time-critical use case.

With this, every operation will be executed at the accurate time and also in real-time! Since Intel does not provide windows drivers with IEEE 1588 support for I21x and I35x network adapters, TSEP does provide its own drivers for this.

The source code is identical for all available platforms (common source) and can be compiled easily on each of them.

Servo Algorithm:



IEEE 1588 tries to synchronize several freewheeling clocks. Each of these clocks is typically implemented as a counter that increments its counter at a given frequency. Due to the frequency and the meter reading, the current time can now be derived at any time. Since it is not technically possible to have identical frequencies generated by several oscillators, the frequency must be readjusted. Since it is technically much easier to manipulate the counter cycle, this is changed. This adjustment must be made via a control algorithm, as the adjustments are subject to various disturbances. In addition, however, faults that may occur in the transport route must also be taken into account. Since every IEEE 1588 implementation is based on its own hardware and hardware topology, there cannot be "the general control algorithm".

For this reason the servo algorithm of the TSEP IEEE 1588 stack is located in a separate library (.dll or .so). Therefore every customer can implement its own servo algorithm independently of the TSEP IEEE 1588 stack. TSEP provides a standard servo algorithm, which focuses only on the determined time difference between master and slave (also called MeanPathDelay) to determine the error in the frequency of their own clock. This type of algorithm is independent of the hardware topology used and provides quite useful results. To determine errors within the system and include them into the calculation of their own frequency, our customers can implement their own servo algorithms. These custom algorithms only make sense if the hardware used and the expected topology is known. Due to the hardware used, the error models can then be created and used. For this type of control algorithms Kalman filters are particularly suitable, which can be modeled specifically for the corresponding problem.

Each control algorithm contains at least two states. In the first state, the offset between master and slave (MeanPathDelay) is so large that the algorithm cannot close this gap within an acceptable control time. In this state, the time received from the master is taken over directly without correction as a separate slave time in the hope that the MeanPathDelay value determined in the next synchronization interval is significantly smaller. This state is maintained until an acceptable MeanPathDelay is reached. This state is the default state after starting the clock or if synchronization is lost due to problems. In the second state, the actual control algorithm is used, which tries to determine the correction values of the own clock and to approximate its own time as exactly as possible to the master time.

The TSEP standard servo algorithm attempts not to incorporate invalid or incorrect MeanPathDelay into the controlling. With Gigabit Ethernet according to IEEE 802.3 the data transmission is non deterministic, any participant can access the network at any time. The access is organized via packet collisions. This can lead to packets being transmitted much later than it is actually assumed. This delay does not appear in the transmitted data packets. In order to protect the servo algorithms against these false and thusly disturbing data, the servo algorithm tries to detect these false data packets and they are excluded in the control algorithm.

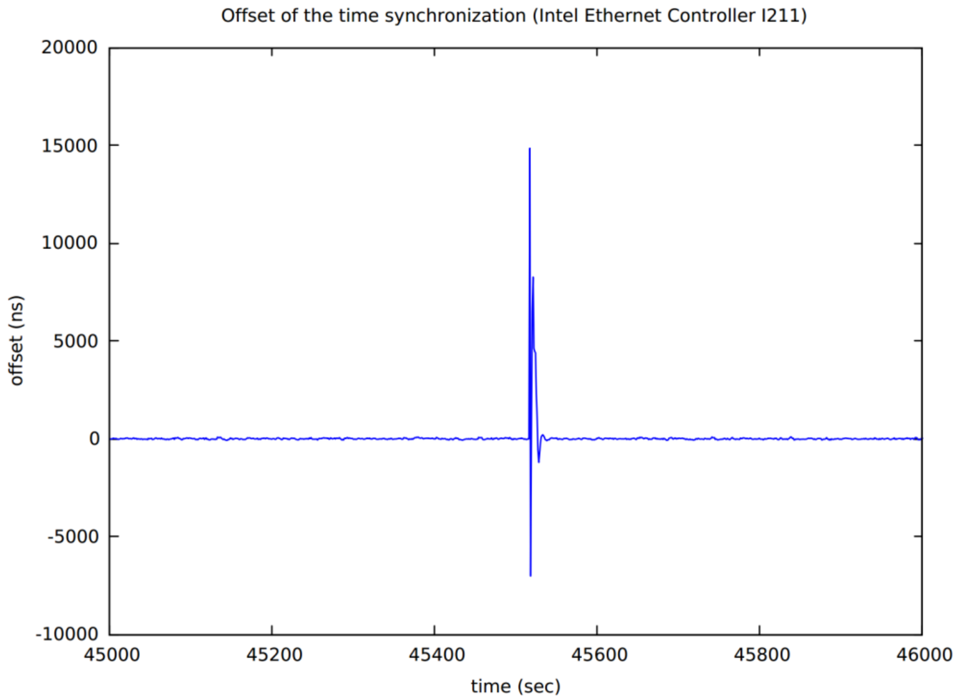


Figure 1: Wrong MeanPathDelay through network delay

The diagram above shows such a wrong package that was subsequently included in the control algorithm and then had to be balanced.

These false data can be recognized by the significantly increased MeanPathDelay. To detect these false MeanPathDelay, the standard deviation of MeanPathDelay is calculated:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(xi - \mu)^2}{n}} \qquad \begin{array}{l} \sigma \text{ Standardabweichung} \\ \mu \text{ Erwartungswert} \\ n \text{ Anzahl der Meßwerte} \end{array}$$

Figure 2: Calculation of the standard deviation MeanPathDelay

If a new MeanPathDelay clearly exceeds the calculated standard deviation, it will not be used for further processing.

In the next step an attempt is made to calculate the correction of the own clock from the calculated MeanPathDelay. For this purpose, the deviation of the slave from the master, which was determined by the MeanPathDelay algorithm, is transferred to the frequency of the own counter (clock). For this purpose, the error of the own clock per counter step is calculated in the first step:

$$\Delta F\ [sec] = \frac{\Delta tm\ [sec]}{\Delta ts\ [sec] * (1\ /\ tps)} \qquad \begin{array}{ll} \Delta F: & \text{Fehler der Uhr pro Tick} \\ \Delta tm: & \text{Offset Master/Slave = MeanPathDelay} \\ \Delta ts: & \text{Zeitdauer seit letzter Berechnung} \\ tps: & \text{Anzahl Tick in einer Sekunde} \end{array}$$

Figure 3: Calculation of internal clock error

In the case of such simple control algorithms, the system starts to oscillate again and again because the control algorithm uses the correction value correspondingly, depending on the detected error. To avoid such upsurge, the TSEP IEEE 1588 control algorithm considers the 1st derivative of the MeanPathDelay.

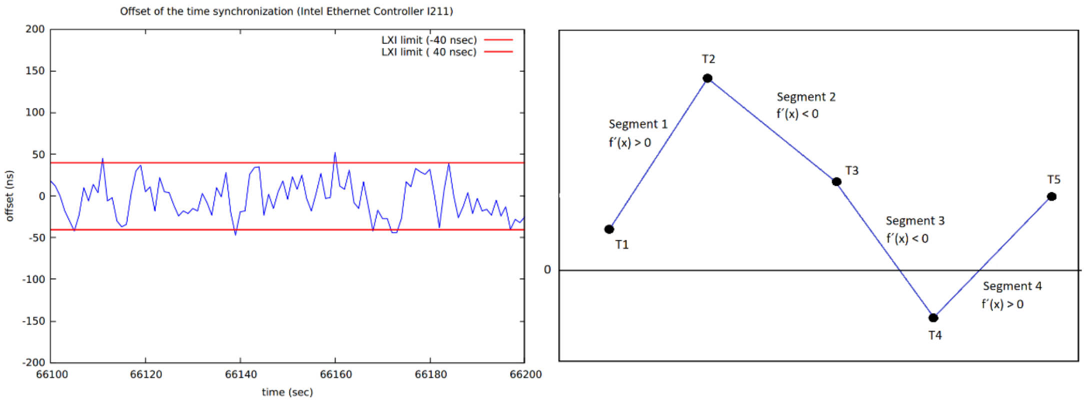


Figure 4: Extended control algorithm

As can be seen in the diagram above, the control algorithm already at the end of segment 2, that is to say at T3, can recognize that the corresponding regulation is taking effect and corrects the adaptation accordingly.

The TSEP standard algorithm allows you to build simple and well-functioning systems independent of fault models and hardware topologies. Of course there are compromises in accuracy, but you can build systems with an average accuracy of +/- 80ns.

With an adequate topology even a range of +/- 40ns is possible.

Examples of use / advantages:

With the growing complexity of nowadays measurement tasks and number of measurement devices, traditional and centralized measurement concepts are hardly future-proof. In addition requirements for measurement systems have changed drastically. Trigger lines with 10 meters length and a cable delay of 40ns may have been reasonable in the past, today they are no longer tolerable.

With growing complexity of measurement tasks, the cable length between the individual components gets even bigger. Ultimately a vicious cycle, but with IEEE 1588 this cycle can be broken. Hardware lines from the IEEE 1588 component can trigger measurement tasks on their own, provided that the component is synchronized with a highly accurate master clock.

Not only metrology will benefit from this, but also automation and production engineering can take advantage of this timed approach.

Now that required hardware is available for the consumer market, this technology is ready to use for a larger audience. On the software side the TSEP IEEE 1588 is a low priced, platform independent and high performance solution for your systems.

Licensing and Pricing:

Basically, the TSEP IEEE 1588 Stack requires a license on each device. For precompiled versions, the customer license number is written to the binaries individually for each customer, so it can be reconstructed which customer has equipped which device with the TSEP IEEE 588 Stack.

TSEP has two license models for the TSEP IEEE 1588 stack.

The first model is intended for customers who would rather sell a small number of devices with IEEE 1588. For these customers, TSEP provides a precompiled version of the TSEP IEEEE 1588 stack, which must be sold through a quantity control system. TSEP does not use any dongle or similar for the license verification of this license model. The graduations for this license model are 100/500/1000 licenses. For this license model, a one-off **provisioning fee of € 999** must be paid, which will be charged for the provision of the individual customer license.

The second license model is aimed at customers who do not want to have a piece-count commitment to the TSEP IEEE 1588 stack. With this model, the customer receives the complete source code of the stack and the drivers. The customer can then indefinitely equip any number of devices with the TSEP IEEE 1588 Stack. In addition, this model includes a multi-year support contract that provides customer support and enhancements to the TSEP IEE 1588 stack.

▪ **Binary license up to 100 licenses**	**Price per license 99,-- €**
▪ **Binary license from 100 up to 500 licenses**	**Price per license 49,-- €**
▪ **Binary license from 500 up to 1000 licenses**	**Price per license 29,-- €**
▪ **Source-Code license**	**On request**
▪ **Support**	**Per Year 600,-- €**