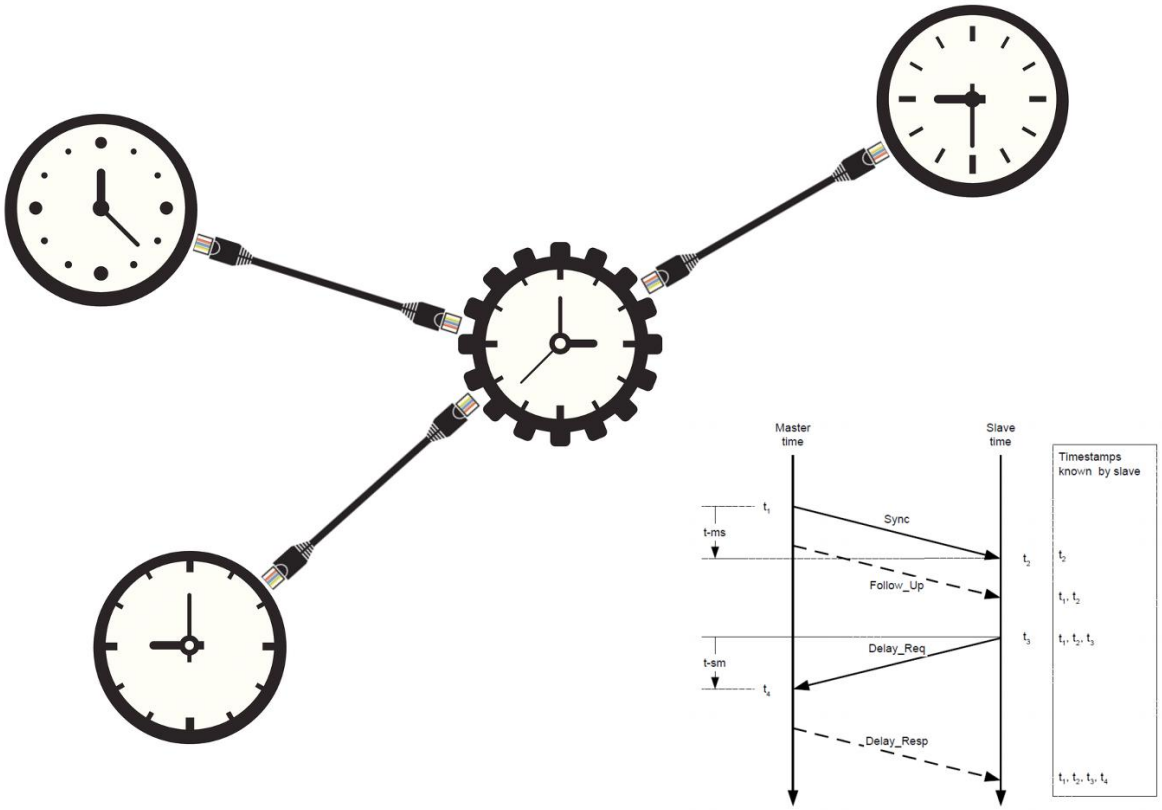
TSEP

Technical
Software
Engineering
Plazotta

IEEE 1588 PTP



Wie spät ist es?

Probleme und Lösungsansätze für IEEE 1588 Implementierungen.

Der IEEE 1588 Standard

Die Zeitsynchronisation über 1588 ist bereits seit 2008 als IEEE Norm festgeschrieben und wird bereits in verschiedensten Bereichen verwendet. Bisher war die Nutzung dieses Standards immer mit exotischer Hardware verbunden, also mit Implementierungen von Netzwerkadaptern in verschiedenen FPGAs oder Embedded Controller. Mit der Einführung der Intel Netzwerkchip Familien Intel I21x und Intel I35x ist dieser Standard nun für den Consumer Markt verfügbar. Somit ist die Grundlage für neue Projekte auf Basis von Consumer Hardware gelegt.

Für die softwaretechnische Umsetzung stehen einige käufliche IEEE 1588 Implementierungen zur Verfügung. Über das amerikanische LXI Konsortium steht sogar eine quasi kostenlose Implementierung des Standards zur Verfügung (es fallen nur Mitgliedsbeiträge an). Die deutsche Firma TSEP hat diese IEEE 1588 Implementierung in Zusammenarbeit mit dem LXI Konsortium entwickelt. TSEP vertreibt auch eine kostenpflichtige Version speziell für Firmen, die kein Interesse an einer LXI Mitgliedschaft haben.

Die grundsätzliche Frage, die sich bei jedem IEEE 1588 Projekt stellt, ist, mit welcher Genauigkeit die Zeitsynchronisation erfolgen muss. Die erreichbare Genauigkeit hängt in der Regel von der verwendeten Hardware, der Topologie und dem verwendeten Regelalgorithmus ab. Moderne IEEE 1588 Implementierungen haben die Möglichkeit, verschiedene Regelalgorithmen zu definieren und diese einfach auszutauschen. Auch die Firma TSEP hat den Regelalgorithmus als eigenständiges Modul mit definierten Schnittstellen festgelegt. Somit kann der Nutzer einfach einen eigenen Algorithmus definieren und diesen ins System einbringen und testen.

Wird IEEE 1588 zum Beispiel bei der Synchronisation von WLAN-Lautsprechern verwendet, ist das menschliche Gehör die Maßeinheit für die Genauigkeit. Das menschliche Gehör kann Laufzeitdifferenzen ab 10 µs erkennen. Somit muss hier die erreichte Genauigkeit für die Synchronisation der WLAN-Lautsprecher unter 10 µs liegen.

Betrachtet man aber messtechnische Aufgaben, sind andere Genauigkeiten gefordert. Innerhalb der Messtechnik werden Messungen in der Regel mit Hilfe von Triggern ausgelöst. In der Regel sind diese Trigger Signaländerungen (steigende oder fallende Flanken, Überschreitung von Pegelwerten etc.). Diese Signale werden über Kabel von der Quelle zum Messgerät übertragen. Somit ist die Laufzeiten innerhalb des Triggerkabels die maßgebende Zielgröße für die Genauigkeit. Geht man von Kabellängen von ca. 5 Metern aus, was eher großzügig bemessen ist, kann man von einer Laufzeit von 25 ns ausgehen (Laufzeit von 5 ns per Meter). Somit wäre die Genauigkeit bei messtechnischen Problemen in dieser Größenordnung einzuordnen. Jedoch hat sich im Bereich Messtechnik mit der Vorstellung von 5G Technologien im

Mobilfunk diese Größenordnung deutlich nach unten verschoben. Für diese Technologien wären Genauigkeiten im Subnano-Bereich wünschenswert.

Der IEEE 1588 Regelalgorithmus

Bei IEEE 1588 wird versucht mehrere freilaufende Uhren zu synchronisieren. Jeder dieser Uhren ist in der Regel als Zähler implementiert, der mit einer vorgegeben Frequenz seinen Zähler inkrementiert. Aufgrund der Frequenz und des Zählerstandes kann nun jederzeit die aktuelle Zeit abgeleitet werden. Da es technisch nicht möglich ist, von mehreren Oszillatoren identische Frequenzen erzeugen zu lassen, muss die Frequenz nachgeregelt werden. Da es technisch deutlich einfacher ist, den Zählerzyklus zu manipulieren, wird dieser verändert. Diese Anpassung muss über einen Regelalgorithmus erfolgen, da die Anpassungen verschiedenen Störungen unterliegen. Zusätzlich müssen aber auch Störungen, die im Transportweg auftreten können, berücksichtigt werden. Da jede IEEE 1588 Implementierung auf ihrer eigenen Hardware und Hardware-Topologie basiert, kann es „den allgemeinen Regelalgorithmus“ nicht geben kann.

Prinzipiell kann man die Algorithmen in zwei Gruppen aufteilen. Die erste Gruppe basiert eher auf einfache Algorithmen, die sich in der Regel nur darauf konzentrieren, aus der ermittelten Zeitdifferenz zwischen Master und Slave (auch MeanPathDelay genannt) den Fehler in der Frequenz der eigenen Uhr zu ermitteln. Diese Art von Algorithmus ist unabhängig von der verwendeten Hardware-Topologie und liefert durchaus brauchbare Ergebnisse. Die freie LinuxPTP Implementierung und die TSEP Implementierung enthalten jeweils einen solchen Algorithmus standardmäßig.

Die zweite Gruppe sind Algorithmen, die versuchen, die Fehler, die im System stecken, zu ermitteln und diese bei der Berechnung des Fehlers der eigenen Frequenz miteinzubeziehen. Diese Algorithmen sind nur dann sinnvoll, wenn die verwendete Hardware und die zu erwartende Topologie bekannt ist. Aufgrund der verwendeten Hardware lassen sich dann die Fehlermodelle erstellen und verwenden. Für diese Art von Regelalgorithmen sind insbesondere Kalman-Filter geeignet, die speziell auf das entsprechende Problem modelliert werden können.

Jeder Regelalgorithmus enthält mindestens zwei Zustände. Im ersten Zustand ist der Offset zwischen Master und Slave (MeanPathDelay) so groß, dass der Algorithmus diese Lücke in akzeptabler Regelzeit nicht schließen kann. In diesem Zustand wird die vom Master erhaltene Zeit ohne Korrektur direkt als eigene Slave-Zeit übernommen in der Hoffnung, dass der im nächsten Synchronisationsintervall ermittelte MeanPathDelay Wert deutlich kleiner ist. Dieser Zustand wird solange beibehalten, bis ein akzeptabler MeanPathDelay erreicht ist. Dieser Zustand ist der Default-Zustand nach dem Starten der Uhr oder wenn aufgrund von Problemen die Synchronisation verloren geht. Im zweiten Zustand greift dann der eigentliche Regelalgorithmus, der

versucht die Korrekturwerte der eigenen Uhr zu ermitteln und die eigene Uhrzeit möglichst genau der Master-Uhrzeit anzunähern.

Einfache IEEE 1588 Regelalgorithmen

Wie bereits gesagt, sind diese Art von Algorithmen darauf ausgerichtet, nur den Korrekturwert der eigenen Uhr aufgrund des ermittelten MeanPathDelay zu bestimmen. Ein eigenes Fehlermodell kommt hier nicht oder nur spärlich zu tragen. Diese Art von Algorithmen ist relativ einfach und lässt sich unabhängig von der Hardware verwenden.

Auf Basis des Standard-Regelalgorithmus der TSEP IEEE 1588 Implementierung soll die Arbeitsweise eines solchen Algorithmus veranschaulicht werden. Dieser einfache Regelalgorithmus versucht im ersten Schritt, ungültige oder falsche MeanPathDelays nicht mit in die Regelung zu übernehmen. Die hier besprochene IEEE 1588 Implementierung basiert auf Intel Netzwerk Chips der Familie I21x. Diese verwenden das Gigabit Ethernet nach IEEE-Norm 802.3. Diese Art von Netzwerksysteme ist nicht deterministisch, jeder Teilnehmer kann jederzeit auf das Netzwerk zugreifen. Die Zugriffe werden über Paketkollisionen organisiert. Hierbei kann es dazu kommen, dass Pakete erst deutlich später übertragen werden, als eigentlich angenommen wird. Dieser Delay taucht nicht in den übertragenen Datenpaketen auf. Um die Regelalgorithmen vor diesen falschen und somit störenden Daten zu schützen, versucht der Regelalgorithmus diese falschen Datenpakete zu detektieren und nicht mit in den Regelalgorithmus einzubinden.

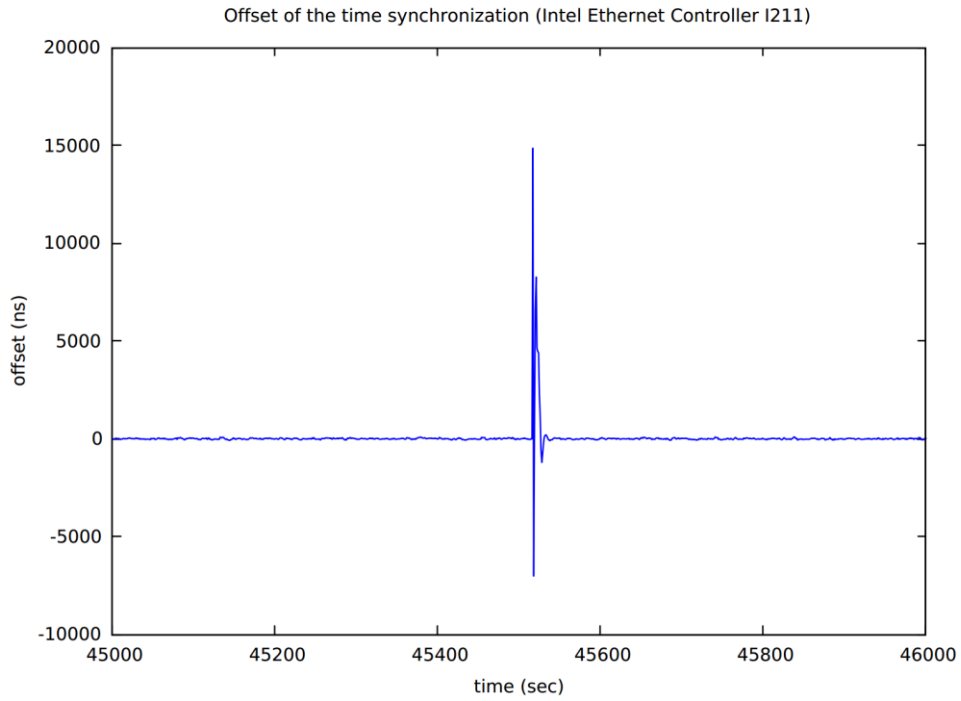


Abbildung 1: Falscher MeanPathDelay durch Netzwerkverzögerungen

Das obenstehende Diagramm zeigt ein solches falsches Paket, das nachfolgend in den Regelalgorithmus einbezogen wurde und dann ausgeglichen werden musste. Diese falschen Daten sind an den deutlich erhöhten MeanPathDelay zu erkennen. Um diese falschen MeanPathDelays zu detektieren, wird die Standardabweichung des MeanPathDelay berechnet:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(xi - \mu)^2}{n}}$$

σ Standardabweichung
μ Erwartungswert
n Anzahl der Meßwerte

Abbildung 2: Berechnung der Standardweichung MeanPathDelay

Überschreitet nun ein neuer MeanPathDelay die berechnete Standardabweichung deutlich, wird dieser nicht für die weitere Verarbeitung verwendet.

Im nächsten Schritt wird versucht, aus den berechneten MeanPathDelay, die Korrektur der eigenen Uhr zu berechnen. Hierzu wird die Abweichung des Slaves vom Master, die über den MeanpathDelay Algorithmus bestimmt wurde, umgelegt auf die Frequenz des eigenen Zählers (Uhr). Hierzu wird im ersten Schritt der Fehler der eigenen Uhr pro Zählerschritt berechnet:

$$\Delta F\ [sec] = \frac{\Delta tm\ [sec]}{\Delta ts\ [sec]\ *\ (1\ /\ tps)}$$

ΔF: Fehler der Uhr pro Tick
Δtm: Offset Master/Slave = MeanPathDelay
Δts: Zeitdauer seit letzter Berechnung
tps: Anzahl Tick in einer Sekunde

Abbildung 3: Berechnung des Fehler der internen Uhr

Der Intel I21x kann die Zeit, nach dem der eigene Zähler inkrementiert wird (alle 8 ns), in gewissen Grenzen variieren. Der Fehler pro Zählerinkrement wird durch den Algorithmus (nach obiger Formel) in die Hardware programmiert.

Bei solchen einfachen Regelalgorithmen kommt es immer wieder zum Aufschwingen des Systems, da der Regelalgorithmus in Abhängigkeit vom ermittelten Fehler den Korrekturwert entsprechend anwendet. Um ein solches Aufschwingen zu vermeiden, betrachtet der TSEP IEEE 1588 Regelalgorithmus die 1. Ableitung des MeanPathDelays.

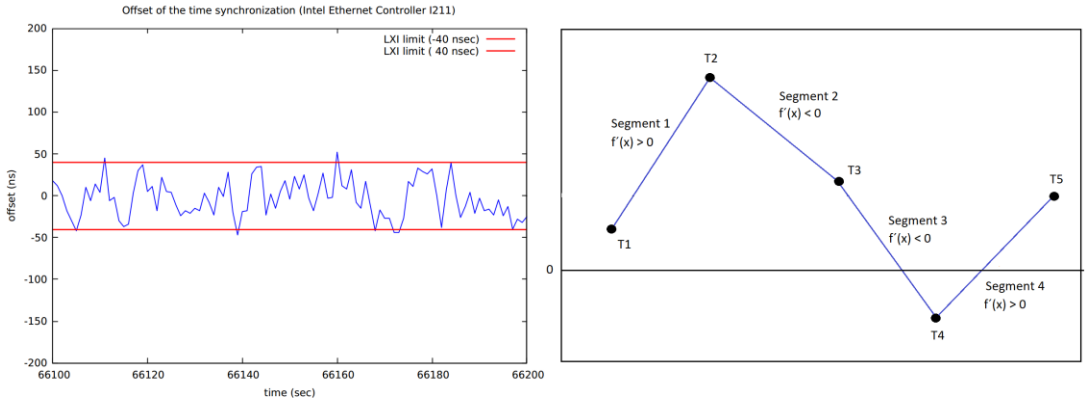


Abbildung 4: Erweiterter Regelalgorithmus

Wie in dem obigen Diagramm zu sehen ist kann der Regelalgorithmus bereits am Ende von Segment 2, also bei T3, erkennen, dass die entsprechende Regelung greift, und die Anpassung entsprechend korrigieren.

Wie gesagt lassen sich mit diesen einfachen Regelalgorithmen unabhängig von Fehlermodellen und Hardware-Topologien einfache und funktionierende Systeme aufbauen. Natürlich sind hierbei Abstriche in der Genauigkeit zu machen. Deshalb sind diese Regelalgorithmen in der Regel bei allen IEEE 1588 Implementierungen standardmäßig enthalten und erlauben den Nutzer einen einfachen Einstieg in die IEEE 1588 Problematik.

Komplexe IEEE 1588 Regelalgorithmen

Soll jedoch auf Fehlermodelle und Hardware-Topologien eingegangen werden, muss mit anderen Ansätzen gearbeitet werden. Für derartige Probleme können Kalman Filter (oder auch Kalman-Bucy-Stratonovich-Filter) verwendet werden. Benannt ist der Kalman-Filter seinen Entdeckern Rudolf E. Kálmán, Richard S. Bucy und Ruslan L. Stratonovich, die das Verfahren unabhängig voneinander entdeckt bzw. wesentliche Beiträge dazu geliefert haben. Der Kalman-Filter dient dazu, Fehler in realen Messwerten zu reduzieren und Schätzungen für nicht messbare Systemgrößen zu liefern. Voraussetzung dabei ist, dass die notwendigen Werte durch ein mathematisches Modell beschrieben werden können. Die Besonderheit des 1960 von Kálmán vorgestellten Filters [3] bildet dabei seine spezielle mathematische Struktur, die den Einsatz in Echtzeitsystemen verschiedener technischer Bereiche ermöglicht. Dazu zählt u. a. der Einsatz in elektronischen Regelkreisen in Kommunikationssystemen. Im Rahmen der mathematischen Schätztheorie spricht man auch von einem bayesschen Minimum-Varianz-Schätzer für lineare stochastische Systeme in Zustandsraumdarstellung.

Der Kalman Filter versucht, Fehlermodelle mit in die Schätzung für den eigentlichen Korrekturwert einzubringen. Hierzu ist es im ersten Ansatz notwendig, die Fehlerquellen innerhalb einer IEEE 1588 Realisierung abschätzen zu können.

Stabilität des Oszillators innerhalb der Intel I21x Netzwerkchips

Eine Fehlergröße bei einer IEEE 1588 Implementierung ist die Stabilität des verwendeten internen Zählers. Die aktuelle Uhrzeit wird aufgrund eines internen Zählers abgeleitet. Der Takt, mit dem der Zähler inkrementiert wird, wird von einem Oszillator oder einer anderen Quelle abgeleitet. Bei den Intel Netzwerkchips I21x und I35x ist diese von dem verwendeten Ethernet Takt, also 125 MHz, abgeleitet.

Abbildung 5: Messaufbau mit Omicron Grandmaster Clock

Um einen Überblick über die Stabilität dieses Taktes zu bekommen, wurden im TSEP Labor Messungen vorgenommen (siehe Bild oben). Es wurden mehrere Rechnerboards und PC-Einsteckkarten mit den entsprechenden Netzwerkchips vermessen. Hierzu wurde die interne Uhr auf den Netzwerkchips mit einer konstanten Anpassung über mehrere Stunden betrieben. Als Messgröße wurde ein pps-Signal (Pulse per Second)

auf dem GPIO des Intel Netzwerkchips programmiert und mit einem entsprechenden Oszilloskop nachgemessen. Als Grandmaster Clock wurde eine Omicron Grandmaster Clock verwendet.

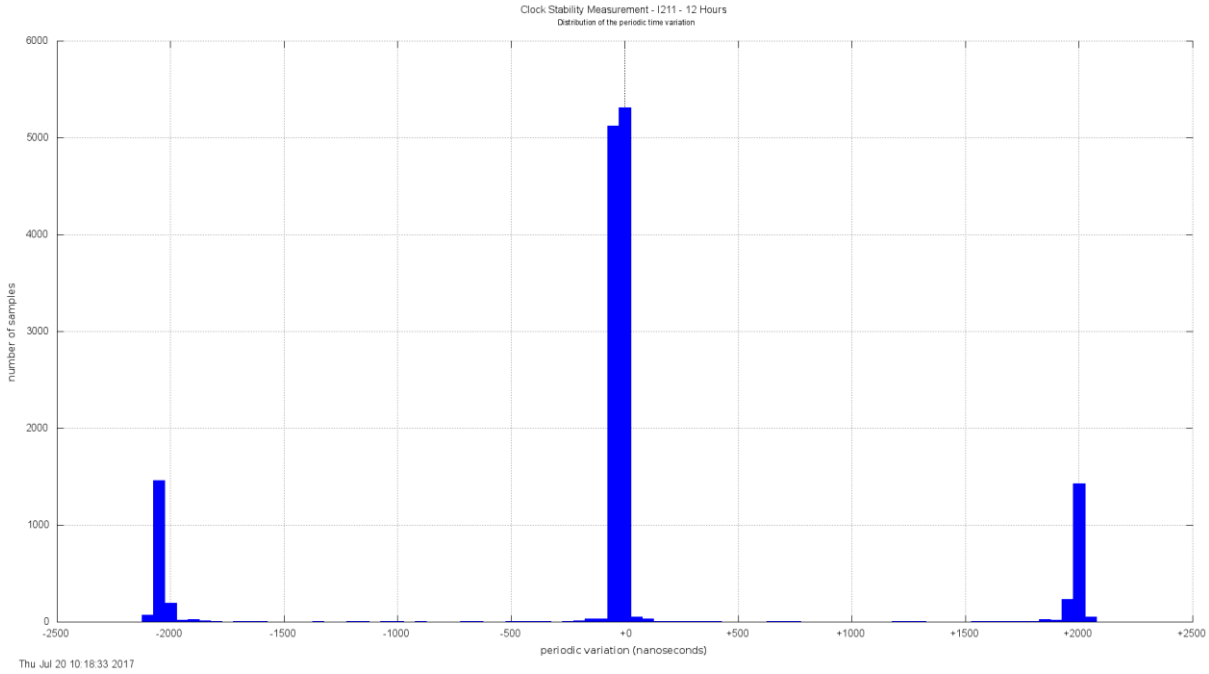


Abbildung 6: Fehlerverteilung Raumtemperatur

Das obige Diagramm zeigt die Differenz der Periodendauer des pps-Signals. In diesem Diagramm kann man entnehmen, dass sich der Fehler des Taktes um die Mittellage gruppiert. Die Messung zeigt, dass sich der Fehler bei ca. +/- 2000 ns bewegt. Basierend auf der 125-MHz-Taktung der Uhr ergibt sich ein Fehler pro Taktung/Inkrement der Uhr von:

2000 ns / 125 MHz = 16 x 10-15 sec

Diese Fehlergröße konnte auch bei anderen Intel 21x Netzwerkkarten oder Embedded Computer mit Intel I21x nachgemessen werden.

Ein weiterer Punkt ist die Temperaturstabilität der internen Uhr der Intel I21x Chips. Hierzu wurde die Umgebungstemperatur des Netzwerkchips variiert. Nachfolgend sind zwei Diagramme aufgeführt, die den Fehler bei den Ecktemperaturen für den Chip darstellen.

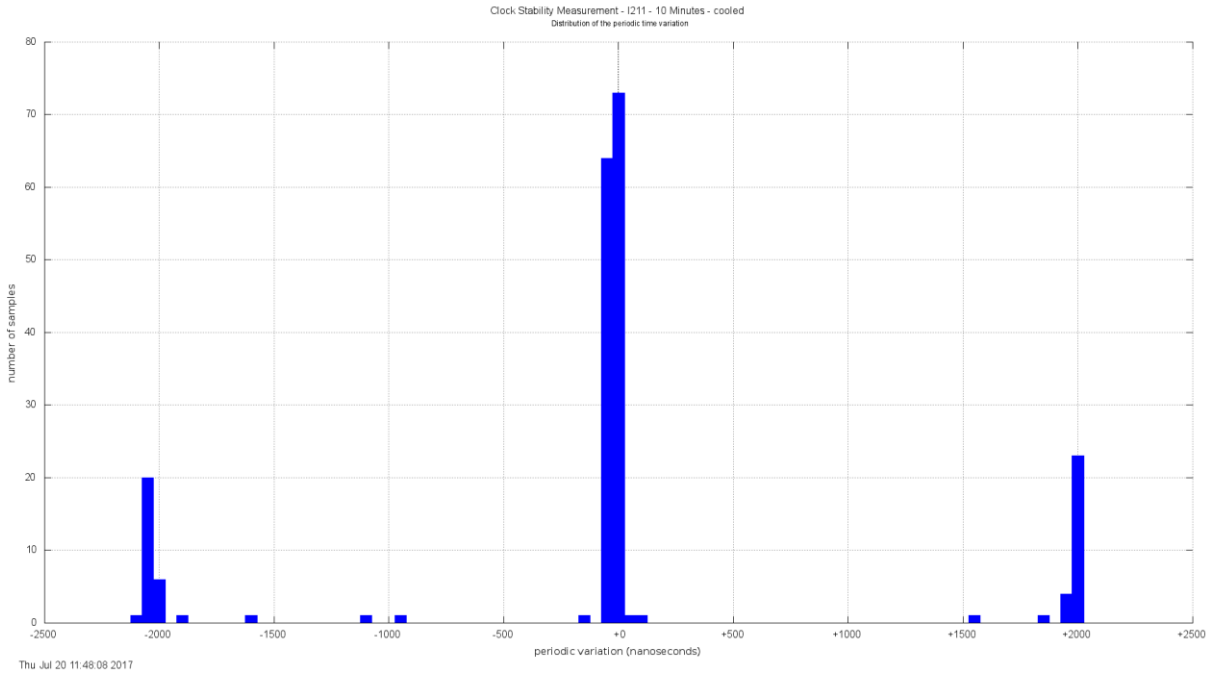


Abbildung 7: Fehlerverteilung gekühlter Netzwerkchip

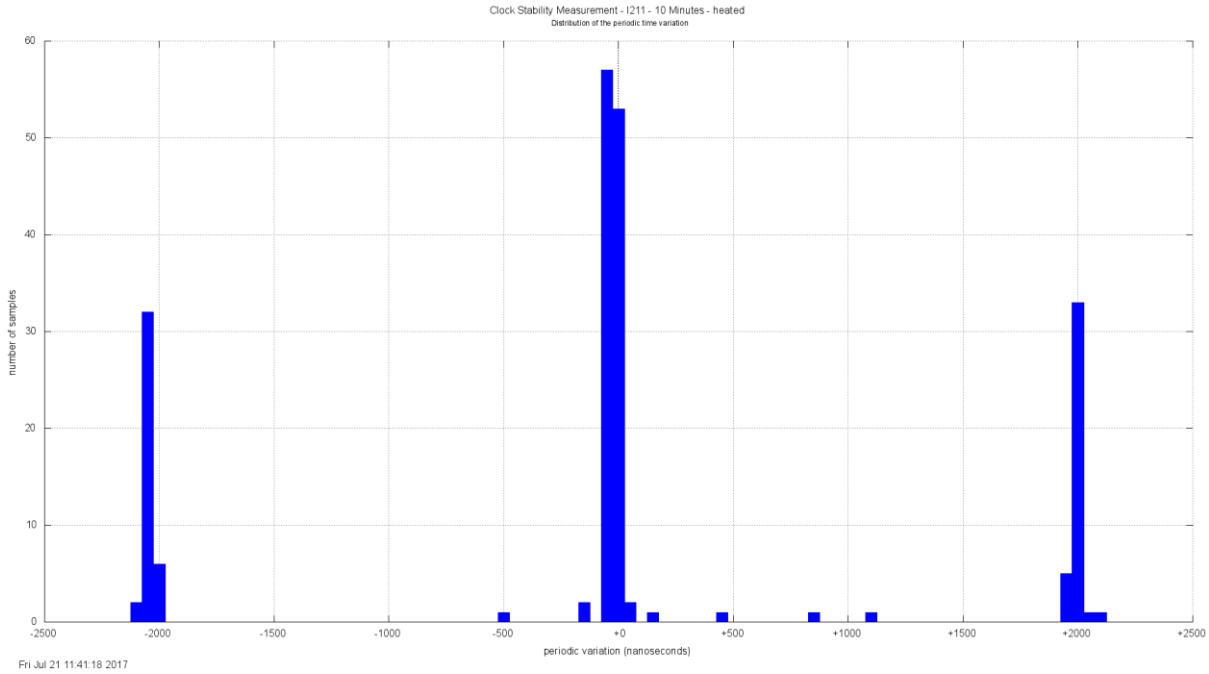


Abbildung 8: Fehlerverteilung erwärmter Netzwerkchip

Aus den Diagrammen kann man sehen, dass die Fehlerverteilung deutlich anders ist. Bei erhöhter Temperatur tritt der Fehler auch erkennbar häufiger auf.

Multiplexen bei der Gigabit Ethernet Datenübertragung

Um die nachfolgenden Problematik besser verstehen zu können, müssen noch einige Grundlagen zu Gigabit Ethernet nach IEEE-Norm 802.3 und den Intel Netzwerkchips erörtert werden. Entstanden ist das 1 GB Ethernet aus dem zuvor etablierten 100 MB Ethernet Standard. Die verwendeten Ethernet CAT 5 Kabel waren für die Übertragung von Signalen mit 125 MHz vorgesehen. Diese Kabel hatten hierzu vier Paare (jeweils zwei Drähte). Jedoch wurden nur über zwei Paare Daten übertragen. Bei Gigabit Ethernet wird nun über alle vier Paare jeweils zwei Bits übertagen, also:

125 MHz x 2 Bits x 4 Kommunikationskanäle = 1000 Mbps =1Gbps

Es werden also immer zwei Bits auf den vier verdrehten Paaren übertragen. Der Sender muss jedes Byte in vier Teile aufteilen und der Empfänger diese wieder zusammen führen. Der Verarbeitungstakt ist hierbei wie gesagt 125 MHz.

Die vier Drahtpaare werden hierbei gleichzeitig für beide Richtungen verwendet. Die innerhalb des Gigabit Ethernet verwendete Frequenz ist 125 MHz laut GMII (Gigabit Media Independent Interface). Die Frequenzen am Sender und Empfänger sind nicht zwangsweise gekoppelt. Abhängig von der Datenübertagung wird die Frequenz für die Daten aus dem eigenen Takt oder aus dem des Netzwerkes abgeleitet. Aufgrund dieses Vorgehens und der wahrscheinlichen Annahme, dass die vier Leitungen nicht identisch lang sind, muss mit einer unterschiedlichen Laufzeit für die einzelnen Teildaten (4 x 2 Bits) gerechnet werden. Erst nachdem alle vier Datenteile eines Bytes zur Verfügung stehen, können diese auch wieder zusammengesetzt werden, weshalb es zu Verzögerungen kommen kann. Je nach Szenario können hier bis zu 20 ns Verzögerung entstehen. Die gesamte Problematik wird sehr gut im Dokument „Improving IEEE 1588 synchronization accuracy in 1000BASE-T systems“ [1] beschrieben. Da die aufgezeigten Probleme leider im ns-Bereich liegen, beeinflussen sie die Genauigkeit der Zeitsynchronisation über IEEE 1588 erheblich. Mit kupferbasierten Gigabit Ethernet Implementierungen wird es damit sehr aufwendig bis nahezu unmöglich, den Subnano-Bereich zu erreichen. Dieser Fakt war sicherlich auch einer der Gründe, wieso das White Rapid System [2] auf Netzwerke mit Glasfasertechnik zurückgreift.

Berücksichtigung von Laufzeitfehler bei „Long Linear Paths“

Eigentlich kann man nur bei kleinen Systemen oder in der Entwicklung von Systemen mit einer Grandmaster Clock, einer Transparent Clock (Switch) und einer Slave Clock rechnen. In der Realität muss man jedoch von Verbindungen ausgehen, in der die Grandmaster Clock über mehrere Transparent Clocks oder sogar non-PTP konforme Switches laufen muss. Alle diese Faktoren summieren sich über die Transportkette vom Master zum Slave. Um den eigenen Regelalgorithmus zu verbessern, kann man versuchen, die Fehler, die in den einzelnen Vermittlungsknoten auftreten, zu erfassen und in den eigenen Regelalgorithmus einzubinden. Hierzu kann mit Hilfe des Kalman-Filters ein Modell aufgebaut werden, das den einzelnen Fehlern Rechnung trägt. Das Dokument [4] „Accurate Time Synchronization in PTP-based Industrial Networks with Long Linear Path“ von Daniele Fontanelli und David Macii zeigt eine Modellierung mit Hilfe eines Kalman-Filters für dieses Problem auf. Das Dokument zeigt auch, dass mit Hilfe dieses Ansatzes eine Verbesserung der Genauigkeit von IEEE 1588 Systemen in diesem Szenario erreicht werden können.

Der Kalman Filter

Bereits 1960 entwickelte Rudolf E. Kalman für zeitdiskrete lineare Systeme ein spezielles Verfahren, um aus verrauschten und teilweise redundanten Messungen die Zustände eines Systems (inklusive deren Parameter) zu schätzen. Dieses Verfahren wurde als Kalman-Filter bekannt und erstmals in [3] veröffentlicht. Seither sind viele verschiedene Varianten des Kalman-Filters veröffentlicht worden. Die nachfolgende Beschreibung kann auch detailliert in [5] nachgelesen werden.

Damit der Kalman-Filter richtig eingesetzt werden kann - ist es notwendig, dass die Grundbedingungen des Messsystems bekannt sind. Jeder klassische Kalman-Filter besteht in der Regel aus einer Zustandsraumbeschreibung und dem realen Messsystems mit dem ihm eigenen System- und Messrauschen. Aus diesem System können mit Hilfe des Kalman-Filters die Prädiktion und die Korrektur berechnet werden. Grundsätzlich schätzt ein Kalman-Filter die Ausgangsgröße ŷ(k) und vergleicht diese mit der gemessenen Ausgangsgröße y(k) des realen Messsystems. Die Differenz Δy(k) der beiden Werte wird mit der Kalman-Verstärkung K(k) gewichtet und zur Korrektur des geschätzten Zustandsvektors ẍ(k) verwendet. Die Struktur lässt sich wie folgt beschreiben:

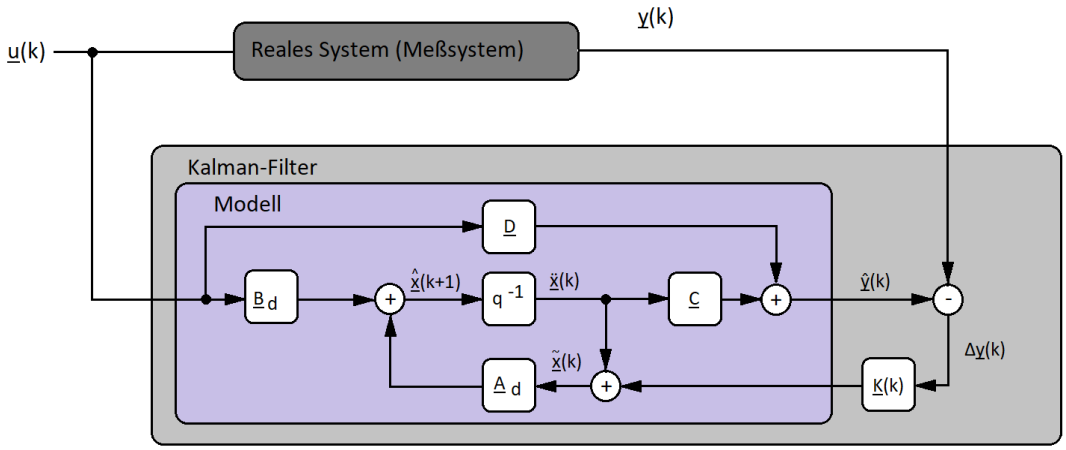


Abbildung 9: Struktur eines Kalman-Filters

Aus dieser Struktur heraus lassen sich die fünf Kalman-Filter Grundgleichungen ableiten:

Prädiktion:

$$\ddot{\underline{x}}(k+1) = \underline{A_d} * \dot{\underline{x}}(k) + \underline{B_d} * u(k)$$
$$\acute{\underline{P}}(k+1) = \underline{A_d} * \dot{P}(k) * \underline{A_d}^T + \underline{G_d} * \underline{Q}(k) * \underline{G_d}^T \quad \text{mit } Q(k) = \text{Varianz}(\underline{z}(k))$$

Korrektur:

$$\underline{K}(k) = \acute{\underline{P}}(k) * \underline{C}^T * (\underline{C} * \acute{\underline{P}}(k) * \underline{C}^T + R(k))^{-1} \text{ mit } \underline{R}(k) = \text{Varianz}(\underline{v}(k))$$
$$\dot{\underline{x}}(k) = \ddot{\underline{x}}(k) + \underline{K}(k) * (\underline{y}(k) - \underline{C} * \ddot{\underline{x}}(k) - \underline{D} * \underline{u}(k))$$
$$\dot{P}(k) = (\underline{I} - \underline{K}(k) * \underline{C}) * \acute{\underline{P}}(k)$$

Auf die Herleitung wird aufgrund der Komplexität verzichtet. Sie kann aber in [3] oder [5] nachgelesen werden.

Bei dem Entwurf eines Kalman-Filters ist es notwendig, das physikalische System mit Hilfe von Differentialgleichungen zeitkontinuierlich zu beschreiben. Es legt den Ausgangsvektor $\underline{y}(t)$ fest. Es ist darauf zu achten, dass dieser Ausgangsvektor alle rauschbehafteten Größen enthält. Die nichtrauschbehafteten Größen werden separat im Ausgangsvektor $\underline{u}(t)$ beschrieben. Bei der Bestimmung der Zustandsvariablen $\underline{x}(t)$ kann es mehrere Ansätze geben. In der Regel sollten diese Möglichkeiten evaluiert werden und der Ansatz, der das Problem an besten beschreibt, sollte dann verwendet werden.

Sind alle Gleichungen und Parameter gewählt, liegt das System in folgender Form vor:

$$\dot{\underline{x}}(t) = \underline{A} * \underline{x}(t) + \underline{B} * \underline{u}(t) + \underline{G} * \underline{z}(t)$$
$$\underline{y}(t) = C * \underline{x}(t) + \underline{D} * \underline{u}(t)$$

Diese zeitkontinuierliche Beschreibung muss dann in eine zeitdiskrete Beschreibung überführt werden. Hierzu können folgende Formeln verwendet werden:

$$\underline{A_d} = e^{A * T_S}$$
$$B_d = \int_0^{Ts} e^{A*v} B dv \quad \text{oder } B_d = \underline{A_d} * \underline{B}$$

T_S ist die Abtastrate des Systems. Die Matrizen $\underline{C}$ und $\underline{D}$ bleiben im zeitdiskreten System identisch.

Zur Bestimmung der Matrix $\underline{G_d}$ können mehrere Ansätze verwendet werden, wie die Abtastung durch den Dirac Impuls. Die verwendete Methode muss individuell bestimmt werden.

Nachdem die Beschreibung des Systems im Zustandsraum vorliegt, kann eine Überprüfung der Beobachtbarkeit erfolgen. Als Kriterium hierfür gilt nach [3] oder [5], dass ein lineares zeitinvariantes System der Ordnung n dann beobachtbar ist, wenn die Bobachtbarkeitsmatrix $S_{B,}$ bzw. S^*_B den Rang n besitzt. Es können aber auch Kriterien von Gilbert oder von Hautus verwendet werden. Sollte das System nicht beobachtbar sein, kann es aufgeteilt werden in ein beobachtbares und in ein nicht beobachtbares.

Abschließend müssen noch das Systemrauschen Q(k) und das Messrauschen R(k) beschrieben werden.

$$\mathbf{Q(k) = Varianz(\underline{z}(k))}$$
$$\mathbf{\underline{R}(k) = Varianz(\underline{v}(k))}$$

Um Kalman-Filter optimal zu nutzen, müssen diese beiden Größen möglichst genau ermittelt werden. In einfachen Systemen kann man von annähernd konstanten Größen sprechen, was aber im Umfeld von IEEE 1588 nicht zutrifft. Es ist davon auszugehen, dass sich diese Größen während der Laufzeit ändern und somit adaptiv angepasst werden.

Es gibt eine Variante des Kalman-Filters, die mit einer adaptiven Bestimmung der beiden Kovarianz-Matrizen arbeitet: der ROSE-Filter (Rapid Ongoing Stochastic Covariance Estimation Filter). Das Prinzip basiert darauf, die Kovarianz des Messrauschens $\underline{R}(k)$ über die Beobachtung der messtechnisch erfassbaren Größe y(k) mittels zwei eingebetteter einfachen Kalman-Filter (mit konstanter Kalman-Verstärkung) zyklisch neu zu bestimmen. Analog hierzu wird die Kovarianz des Systemrauschens $\underline{Q}(k)$ mit Hilfe des Wertes Δy(k) und der gemessenen Größe y(k), der Kovarianz des Schätzfehlers $\underline{\acute{P}}(k+1)$, der Kovarianz Messrauschens $\underline{R}(k)$ und eines einfachen Kalman-Filter bestimmt.

Wenn man die einzelnen Schritte betrachtet, die zu einer vollständigen Beschreibung des Zustandsraumes führt und der Ermittlung der Kovarianzen des System- und Messrauschens, kann man sagen, dass die Erstellung eines Kalman-Filters alles andere als trivial ist. Jedoch können die Randbedingungen des Messsystems optimal in den Kalman-Filter eingebettet werden, was zu einer bestmöglichen Korrektur der internen IEEE 1588 Uhr führt.

Fazit

Der erfolgreiche Einsatz einer IEEE 1588 Implementierung basiert nicht nur auf der Verwendung eines bestehenden IEEE 1588 Stacks oder einer speziellen Hardware. Der problemorientierte Lösungsansatz ist hier der Schlüssel zum Erfolg. Die Möglichkeiten von IEEE 1588 sind breit gefächert. Ohne jedoch die Anforderungen an die notwendige Genauigkeit für eine IEEE 1588 Implementierung und die zur Verfügung stehende Hardware Topologie zu kennen, ist es nur schwer möglich, ein in den Rahmen der Anforderungen funktionierendes System bereitzustellen. Die Analyse im Vorfeld ist zwingend notwendig und erfordert ein entsprechendes Knowhow.

Die Wahl der verwendeten Hardware ist von essenzieller Bedeutung, da die einzelnen Komponenten die Genauigkeit unterschiedlich beeinflussen. Besonders für hohe Genauigkeiten im Subnano-Bereich ist die Verwendung von glasfaserbasierten Systemen zwingend notwendig. Das White Rabit Projekt hat hier gute Vorarbeit geleistet und die entsprechende Hardware und die notwendigen Randbedingungen geschaffen. Auch die Intel Netzwerk Chips können mit glasfaserbasierten PHYs arbeiten, entsprechende Hardware ist auf dem Markt als Consumer Ware verfügbar.

Für hochgenaue Systeme ist ein perfekt abgestimmter Regelalgorithmus unabdingbar. Die Wahl des entsprechenden Algorithmus ist nicht einfach, erfordert einiges an Knowhow und muss an die Gegebenheiten der Hardware und deren Topologie angepasst werden. Die Umsetzung und Simulation des Algorithmus, gerade bei hochgenauen Systemen ist ein nicht unerheblicher Teil einer IEEE 1588 Implementierung. Sie ist aber auch Schlüssel zum Erfolg einer solchen Implementierung. Für die Implementierung solcher effizienter Regelalgorithmen eigenen sich Kalman-Filter sehr gut. Die Beschreibung des Zustandsraums und der Kovarianzen des System- und Messrauschens erfordern aber einiges an Aufwand und Zeit.

Der Artikel erhebt nicht den Anspruch, alle Facetten an dieser überaus komplexen Materie darzustellen, sondern soll einen kleinen Überblick geben, welche Probleme und Lösungsansätze vorhanden sind. Jeder verfügbare IEEE 1588 Stack stellt nur das Handwerkszeug für eine Umsetzung dar. Die richtige Verwendung und die Umsetzung des Regelalgorithmus ist die eigentliche Aufgabe.

Und ja, wie spät ist es eigentlich jetzt?

Literaturverzeichnis:

[1] Improving IEEE 1588 synchronization accuracy in 1000BASE-T systems, Rodney Greenstreet and Alejandro Zepeda

[2] White Rabbit Project. White rabbit wiki, 2016.

[3] Kalman, R. E.: A New Approach to Linear Filtering and Prediction Problems (Transaction of the ASME, Journal of Basic Engineering). 1960, S. 35–45

[4] Accurate Time Synchronization in PTP-based Industrial Networks with Long Linear Path, Daniele Fontanelli and David Macii

[5] R. Marchthaler, S. Dingler: Kalman-Filter, Springer Vieweg

Autor:

Peter Plazotta, Dipl. Ing (FH)
CEO von TSEP und seit 30 Jahren mit dem Design und der Entwicklung von Systemsoftware vor allem in den Bereichen T&M, Telekommunikation und Automotive beschäftigt. Er beschäftigt sich seit über 10 Jahren mit dem Thema IEEE 1588 und hat in diesem Bereich mehrere Implementierungen mit seiner Firma begleitet.

Peter.Plazotta@tsep.com