Novemberlog 2025 — 1,11 Sekunden

Zeit, Nebel und ein konstanter Sprung

Mika Stern, Donau2Space.de

November 2025

Vorwort

Im November 2025 stand ich oft an der Donau, Laptop auf den Knien, das grüne Blinklicht des Mini-GPS im Blick. Nach der Zeitumstellung begannen die Sekunden zu springen, erst zufällig, dann mit einem Muster. Ich suchte die Ursache im Kernel, im Funkfeld, vielleicht auch in mir

selbst. Servus, das war kein Monat für schnelle Antworten, sondern für geduldiges Messen.notierte jede Abweichung, prüfte jede Quelle. Die Donau war still, der Logger unruhig. Zwischen Nebel und Zahlenreihen suchte ich nach einem Takt, der sich nicht täuschen lässt.

Unerwartete Drifts nach der Zeitumstellung

Der Morgen war kühl, die Donau glatt wie gebürstetes Aluminium. Ich stand mit dem kleinen Loggergehäuse am Ufer, das Display flackerte träge im Nebellicht. Seit der Zeitumstellung lief der 44-Tage-Test, eigentlich ein Routineprojekt: GPS-1 Hz-Takt, NTP-Vergleich, Systemzeit-Logging. Doch als ich die Datensätze der Nacht sah, spürte ich, dass etwas nicht stimmte. Die Zeitstempel sprangen – zwei, manchmal achtzehn Sekunden, ohne erkennbaren Grund. Und das, obwohl der GPS-Fix stabil war, HDOP unter eins, keine Spur von Jitter oder Satellitenverlust.

Ich notierte die Anomalien im Feldbuch, während das Gerät weiter tickte. Die Donau rauschte leise, und irgendwo in dieser Gleichmäßigkeit passte das Phänomen nicht hinein. Es war, als hätte die Zeit selbst kurz gezuckt, ein kaum merklicher Riss zwischen NTP und GPS, zwischen Software und Himmel. Ich wusste, ich musste die erste Zeitsprungserie sauber dokumentieren, bevor sich das Muster verwischte.

> „Na, was machst denn da unten so früh?“ rief ein Spaziergänger, der seinen Hund ausführte.
>
> „Zeit vermessen“, antwortete ich, „und sie springt heuer fei unruhig.“

Er lachte, winkte, und ich blieb allein mit den blinkenden Sekunden. Die Abweichungen waren reproduzierbar, das machte sie zugleich spannend und beunruhigend. Ich sah mir die Logs an: GPS-Rohdaten unauffällig, NTP-Offsets wechselnd, Systemzeit driftete in kleinen Schüben. Als hätte jemand den Sekundenzeiger kurz angehalten und dann wieder losgelassen.

Ich begann, die NTP-Server in drei Modi zu prüfen: einmal standardmäßig mit kontinuierlichem Slew, einmal im reinen Step-Betrieb, und einmal komplett deaktiviert, um den Holdover des Systems zu messen. Der 30-Minuten-No-NTP-Lauf zeigte bereits die Tendenz: auch ohne Netzsynchronisation blieben die Zeitsprünge bestehen. Damit war klar, dass die Ursache tiefer liegen musste – entweder im Kernel-Timing oder in der Art, wie die GPS-1PPS-Impulse an den Prozess übergeben wurden.

Ich stellte mir die Zeit als Strom vor, der zwischen diesen beiden Quellen fließt. GPS liefert das klare, absolute Signal, NTP versucht, es zu glätten, zu vermitteln. Doch nach der Zeitumstellung schien der Fluss verunreinigt, kleine Wirbel aus Latenz und Korrektur rissen Stücke aus der Sekunde. Vielleicht war es ein Überlauf in der internen Rechenlogik, vielleicht nur ein Timing-Fehler im USB-Stack. Ich wollte nicht spekulieren, also bereitete ich die nächste Testreihe vor.

Die Sonne kam langsam durch, ein goldener Streifen über der Wasseroberfläche. Ich nahm die Antenne neu aus, justierte den Logger, prüfte den Pegel des 1 PPS-Signals am Oszilloskop. Es zuckte sauber, wie ein Herzschlag. Nur in der Aufzeichnung sprang es. Der Widerspruch nagte an mir. Ich plante die 24-Stunden-Holdover-Messung, diesmal komplett ohne NTP-Kontakt, um zu sehen, ob sich das System selbst stabilisieren würde oder weiter driftet. Parallel wollte ich eine RF-Spektralanalyse durchführen, um Störungen im Umfeld auszuschließen. Vielleicht kam das Rauschen nicht aus der Software, sondern aus der Luft.

> „Pack ma's“, murmelte ich, halb zu mir selbst, halb zur Donau, als ob sie mir zuhören könnte.

Ich startete den Testlauf, notierte Startzeit und Bedingungen. Die nächsten Stunden würde der Logger allein arbeiten, ich nur still beobachten. Es war ein seltsames Gefühl, der eigenen Zeitmessung zu misstrauen. Normalerweise vertraue ich den Messwerten blind, aber jetzt schien jeder Sekundenimpuls eine kleine Unsicherheit zu tragen. Ich überprüfte den NTP-Offset-Graphen: leichte Zacken, periodisch, fast rhythmisch. Vielleicht lag darin schon der Schlüssel – ein Muster, das auf eine periodische Korrektur hinweist.

Ich dachte an die Nacht der Zeitumstellung, an den Moment, als die Systeme zwischen Sommer- und Winterzeit wechselten. Eine Stunde, die zweimal existiert, ein kleiner Knoten in der Zeitleine. Vielleicht war genau dort etwas aus dem Takt geraten, ein Prozess, der sich nicht entscheiden konnte, welche Stunde die richtige war. Solche Fehler sind selten, aber sie zeigen, wie empfindlich unsere digitale Zeit geworden ist.

Der Wind kam auf, trug feuchte Luft über das Messfeld. Ich schloss das Gehäuse, überprüfte die Stromversorgung und nahm mir vor, am Abend die Kernel-Logs mit den GPS-Rohdaten zu korrelieren. Wenn dort Sprünge sichtbar sind, könnte ich die Ursache eingrenzen: entweder ein Software-Step oder ein Hardware-Timing-Glitch. Und falls beides sauber bleibt, müsste ich tiefer graben – vielleicht bis in die Treiber oder den USB-Controller.

Die Donau zog ruhig an mir vorbei, als hätte sie von all dem nichts gehört. Ich blieb noch einen Moment, lauschte dem gleichmäßigen Summen der Elektronik. Dann schrieb ich die letzten Notizen des Tages: „Drifts nach Zeitumstellung – reproduzierbar, 2–18 s, Ursache unklar. Nächste Tests: 24 h Holdover, NTP-Vergleich, RF-Analyse.“

Ich spürte, dass diese Untersuchung mehr werden würde als nur ein technischer Check. Es war, als hätte die Zeit selbst beschlossen, sich einmal prüfen zu lassen. Und ich, mit meinem Logger am Fluss, durfte Zeuge sein.

Langsam verpackte ich das Equipment, der Nebel wich, und das Licht wurde klar. Ein neuer Abschnitt des Experiments begann – still, präzise, und voller Fragen, die ins nächste Kapitel führen würden.

Silhouette an der Donau und die ersten Sprünge

Der Tag hatte mit einer eigenartigen Ruhe begonnen. Ich stand in meiner kleinen Werkstatt, zwischen den noch lauwarmen Lötpunkten und dem Geruch von Zinn, und betrachtete die Wand, an der jetzt der Mini-GPS ordentlich hing. Kein Kabel mehr, das sich um die Tischbeine schlängelte, kein Stecker, der sich heimlich verhakte. Alles sauber, still – fast zu still. Ich griff nach der Box mit den Adaptern, prüfte jeden Stecker, als müsste ich mich vergewissern, dass die Welt noch in Ordnung war. Servus, Ordnung, dachte ich, du bist fei ein seltener Gast hier.

Am Nachmittag zog es mich wieder an die Donau. Die Luft war milchig, das Licht dieser Tage flach wie eine alte Fotografie. Ich ging ohne Ziel, ließ die Hände in den Taschen verschwinden und lauschte dem Knirschen des Kieses unter den Schuhen. Das Wasser trug eine träge Bewegung in sich, kaum Wellen, nur ein zarter Schimmer, als hätte jemand hauchdünnes Metallpulver auf die Oberfläche gestreut. Ich blieb stehen, schaute hinüber zum gegenüberliegenden Ufer.

Dort, wo die Weiden beginnen, stand etwas. Eine Silhouette, kaum greifbar, vielleicht ein Mensch, vielleicht nur ein Schatten zwischen den Stämmen. Ich blinzelte, und sie schien sich zu bewegen – oder auch nicht. Ich weiß nicht, warum mir der Atem kurz stockte. Das war kein richtiger Schreck, eher so ein Moment, in dem alle Geräusche weicher werden. Ich hob die Hand, als wollte ich grüßen, aber der Wind trug nur das Rascheln der Blätter herüber. Dann war nichts mehr da.

> „Hast du das geseh'n?“ fragte ich halblaut in Richtung Wasser.
>
> Der Fluss antwortete nicht, aber das Echo meiner Stimme klang merkwürdig nah.

Ich wandte mich ab, ging langsam zurück. Der Abend zog sich wie ein graues Tuch über die Dächer. Zuhause wartete die Technik, geduldig und fordernd zugleich. Ich schaltete den Logger ein, sah die vertrauten Zahlenreihen durchlaufen. Zunächst alles normal, dann ein Sprung – zwei Sekunden, einfach weg. Wieder einer, diesmal größer. Ich runzelte die Stirn. Diese Sprünge kamen und gingen, als würde jemand die Zeit selbst kurz antippen. Kein Muster, kein Fehler in der Stromversorgung, einfach Lücken. Ich tippte mir Notizen in das Terminal, markierte die Zeilen, doch der Bildschirm blieb stumm in seiner Logik.

Ein Schluck Wasser, dann ein Blick hinaus. Die Donau war jetzt nur noch ein dunkles Band, und irgendwo dort draußen vielleicht diese Gestalt. Ich fragte mich, ob sie mich wirklich angesehen hatte, oder ob das nur mein eigenes Spiegelbild war, das sich in der Dämmerung auflöste. Technik und Wahrnehmung, dachte ich, sind sich manchmal ähnlicher, als man glaubt: beide springen, beide verlieren Daten, beide rekonstruieren Geschichten aus unvollständigen Fragmenten.

Ich beschloss, morgen alles ohne Handy zu testen. Kein Netz, kein Störfeld, nur das GPS und ich. Offline, pur, wie ein Spaziergang durch ein unbeschriebenes Protokoll. Vielleicht finde ich so heraus, ob die Sprünge von außen kommen oder aus dem Inneren der Geräte – oder aus mir selbst. Ich schrieb einen kurzen Plan auf Papier: Startpunkt, Zeit, Zielrichtung. Die Buchstaben wirkten fast altmodisch, aber das tat gut. Papier lügt nicht durch Firmwarefehler.

Als ich später im Schein der kleinen Schreibtischlampe saß, fiel mir auf, wie sehr mich das Geräusch des Flusses begleitet, selbst hier drinnen. Es ist, als würde er in den Leitungen weiterfließen, in den dünnen Kupferbahnen der Platinen. Vielleicht ist das ja die eigentliche Verbindung: ein Fluss draußen, ein Fluss drinnen, beide tragen Signale, beide verändern sich mit der Temperatur.

Ich legte den Logger beiseite, schaltete das Licht aus und lauschte. Kein Rauschen mehr, nur mein Atem und das ferne Tropfen eines Kondensats am Fensterrahmen. In der Dunkelheit zeichnete sich für einen Moment das Bild der Silhouette ab – nicht real, eher wie ein Nachbild, das die Netzhaut nicht loslässt. Ich fragte mich, ob sie heute Nacht wieder dort stehen würde, an derselben Stelle, im gleichen Licht, während ich mich auf den Offline-Test vorbereite.

> „Pack ma's morgen an“, murmelte ich, mehr zu mir selbst als zu irgendwem sonst.

Dann schloss ich die Augen, und die Donau floss weiter, still, als wäre sie das einzige System, das keine Sprünge kennt. So endete der Tag, leise, mit einem Rest von Fragen, die im nächsten Kapitel vielleicht eine Richtung finden werden.

30-Minuten-Slew-Only und erste Hypothesen

Ich starte den Test um 22:47 UTC, draußen ist es still, nur das leise Surren der Lüfter im Rack mischt sich mit dem rhythmischen Klicken der Relais. Für die nächsten dreißig Minuten darf kein NTP-Daemon mehr springen, kein aggressiver Step, kein hektisches Nachziehen. Nur Slew-Only – sanftes Nachführen, Millisekunde für Millisekunde. Ich will sehen, wie der Kernel reagiert, wenn ihm niemand mehr die Zeit aus der Hand reißt.

Die Umstellung im Frühjahr hat etwas durcheinandergebracht. Seitdem meldet der GPS-Empfänger zwar makellose 1PPS-Impulse, HDOP im grünen Bereich, aber der Kernel stolpert. Sprünge von zwei bis achtzehn Sekunden, völlig unmotiviert. Kein Pattern, kein klarer Auslöser. Ich habe die Logs der letzten Wochen durchgesehen, jede Anomalie markiert, aber das Bild bleibt brüchig. Heute also der Versuch, das System auf seine nackte Reaktionszeit zu reduzieren.

Ich deaktiviere NTP-Sprungkorrekturen, setze die Slew-Rate, starte das Logging der Kernel-Offsets. Über GPIO kommt der 1PPS-Impuls herein, parallel über die serielle Schnittstelle ein zweiter Zeitstempel. Ich will wissen, ob der Kernel vielleicht nicht den Impuls selbst, sondern dessen Metadaten falsch interpretiert. Manchmal steckt das Problem nicht im Signal, sondern in der Art, wie es in Software übersetzt wird.

> „So, jetzt bleibst mir aber brav stabil, gell?“ murmle ich, als der erste Datensatz auf dem Schirm erscheint.

Der RF-Sweep läuft im Hintergrund. Zwischen 1,2 GHz und 1,7 GHz lasse ich das Spektrum durchlaufen, suche nach Nebenschwingungen, die den GPS-Empfänger irritieren könnten. Es ist eine stille Arbeit, fast meditativ. Nur die Kurven tanzen langsam auf dem Display, während ich die Offsets in immer neuen Skalen plotte. Nach zehn Minuten erkenne ich die erste Tendenz: Der Offset driftet leicht negativ, etwa 140 Nanosekunden pro Sekunde. Kein wirklicher Sprung, eher ein gleichmäßiges Gleiten. Das ist gut. Das System atmet ruhig.

Nach fünfzehn Minuten fällt mir eine kleine Unregelmäßigkeit auf. Der RF-Sweep zeigt in der Nähe von 1,575 GHz einen leichten Buckel, kaum über dem Grundrauschen. Ich notiere den Zeitpunkt und markiere den Punkt in der Offset-Kurve. Genau dort kippt die Richtung: Der Kernel zieht plötzlich leicht positiv. Könnte ein externer Einfluss sein – vielleicht ein kurzer Burst eines anderen Senders, vielleicht etwas Lokales, ein Oberwellenrest vom Netzteil. Servus, Zufall, denke ich trocken.

Ich bleibe dran. Nach zwanzig Minuten prüfe ich die Logs. Kein einziger NTP-Step, kein Sprung über eine Sekunde. Der Kernel hält sich an die Slew-Vorgabe, gleicht sanft aus. Die Differenz zwischen GPS-Zeit und Systemzeit pendelt sich bei rund 0,9 Millisekunden ein. Das ist besser als erwartet. Aber es erklärt die alten Sprünge nicht. Ich beginne, Hypothesen zu ordnen: Entweder war der Fehler transient – ein kurzer Software-Bug, vielleicht durch die Zeitumstellung getriggert – oder hardwareseitig, eine Störung im PPS-Pfad. Eine dritte Möglichkeit wäre ein falsch gesetzter Offset im Kernel-Modul selbst. Ich erinnere mich, dass ein Patch vor zwei Wochen genau an dieser Stelle Änderungen brachte.

> „Manchmal hilft dir der Zufall fei gar nix, du musst ihn zwingen, sich zu zeigen“, sage ich halblaut.

Der Gedanke gefällt mir. Ich beschließe, den Test über die geplanten dreißig Minuten hinauslaufen zu lassen. Nur zehn Minuten länger, um zu prüfen, ob sich die Drift fortsetzt. Der RF-Sweep wiederholt sich, diesmal mit etwas feinerer Auflösung. Ich sehe denselben Buckel bei

1,575 GHz wieder, leicht verschoben, aber er ist da. Die Offsets bleiben ruhig. Kein Sprung, kein Alarm. Vielleicht war es nur Intermodulation, vielleicht ein kurzer Reflex im Kabel.

Gegen 23:30 UTC stoppe ich die Messung. Die Daten sind klar: Kein Sprung während der gesamten Laufzeit. Der Kernel folgt brav der Slew-Regel, die 1PPS-Impulse kommen regelmäßig. Ich exportiere die Logs, sichere sie auf dem Server. In der Stille des Labors klingt das Relais-Klicken fast wie ein Herzschlag. Ich lehne mich zurück und lasse die letzten Minuten Revue passieren.

Der Abend hat keine großen Antworten gebracht, aber er hat etwas gezeigt: Wenn das System in Ruhe gelassen wird, bleibt es stabil. Vielleicht ist also nicht der Kernel das Problem, sondern die Dynamik um ihn herum. Ich werde das noch genauer prüfen müssen – morgen, beim 24-Stunden-Holdover-Test. Da zeigt sich, ob die Zeit uns wirklich treu bleibt.

24-Stunden-Holdover und Vergleich der Zeitquellen

Ich habe den Rechner heute früh bewusst vom Netz getrennt. Kein NTP, kein Abgleich, kein externer Takt außer dem einen sauberen 1PPS vom GPS. Der Holdover soll volle vierundzwanzig Stunden laufen, ohne Korrektur von außen. Servus Isolation – aber genau das brauch ich, um zu sehen, wie sich die Zeitquellen gegeneinander verhalten, wenn sie auf sich allein gestellt sind.

Der TSC, der Time Stamp Counter der CPU, tickt hochpräzise, solange der Prozessor stabil getaktet ist. Die RTC dagegen ist eine kleine Insel in der Hardware, sie lebt in ihrem eigenen Quarzweltchen, leicht temperaturabhängig, leicht launisch. Und dann der 1PPS, diese eine Sekunde aus dem GPS, die wie ein Herzschlag in mein System fällt. Ich will wissen, wie weit sich die drei voneinander entfernen, wenn keine externe Korrektur eingreift.

> „Na, wie lang hält er diesmal durch?“ fragte ich leise in den Raum, obwohl keiner da war.
>
> „Bis morgen früh, hoffentlich“, antwortete ich mir selbst.

Der Logger läuft seit 06:17. Ich sehe die ersten Messpunkte: TSC-Drift minimal, RTC leicht nachgehend. Noch keine Sprünge. Draußen nieselt es, das Dachfenster beschlägt. Ich höre das leise Surren des Lüfters und das rhythmische Klacken des Relais, wenn der 1PPS-Impuls durchgeht. Jede dieser Sekunden markiert eine Ankerstelle in meinem Datenstrom, ein winziger Fixpunkt in einem Meer aus Mikrosekunden.

Als ich gegen Mittag die RF-Peaks logge, fällt mir auf, dass sich die Amplitude des lokalen 1575-MHz-Bandes kaum verändert. Ich prüfe den Antennenspacer – diesmal nur 0,5 mm statt 1 mm – und notiere mir die Werte. Vielleicht spielt der Abstand zur Wand doch eine Rolle, vielleicht ist es aber auch nur das Wetter. Die Donau liegt grau und still, und irgendwo dort drüben war gestern dieser Schatten, den ich im Regen gesehen hab. Heute ist nichts zu sehen, nur Nebelstreifen, die sich über das Wasser ziehen.

Am Nachmittag beginne ich, die Zeitreihen übereinanderzulegen. Ich nutze die 1PPS-Kante als Referenz, normalisiere alle Timestamps darauf, um Fehler im Bereich von Mikrosekunden sichtbar zu machen. Der RTC-Drift zeigt eine leichte Wellenform, periodisch, vielleicht temperaturbedingt. Der TSC dagegen bleibt fast linear, nur ab und zu ein Sprung, zwei bis acht Sekunden, als würde die CPU kurz den Atem anhalten. Ich markiere sie rot im Plot.

> „Fei komisch, dass des immer nach der Zeitumstellung passiert“, murmel ich.

Ich öffne den zweiten Logger für die RF-Daten und sehe, dass die Peaks genau dann leicht ansteigen, wenn der TSC-Sprung auftritt. Zufall? Vielleicht, aber ich kenne diese Muster, sie sind nie ganz zufällig. Ich plane, die Korrelation später mit Bootstrap-Resampling zu prüfen – zehntausend Wiederholungen, um das Rauschen zu glätten und die Signifikanz zu schätzen. Doch das ist Arbeit für morgen, wenn der Holdover vorbei und das Netz wieder offen ist.

Gegen Abend lehne ich mich zurück. Die Uhr auf dem Schreibtisch zeigt 21:46, aber welche davon stimmt, weiß ich nicht mehr so genau. Die Systemzeit geht laut RTC drei Sekunden nach, laut TSC zwei Sekunden vor. Der 1PPS bleibt mein einziger Fixpunkt. Ich stelle mir vor, wie er draußen im Orbit entsteht, irgendwo über dem Atlantik, in einem Satelliten, der für einen Moment genau über mir steht. Diese eine Sekunde, glasklar, frei von Rauschen, fällt herab wie ein Tropfen Licht.

Ich kontrolliere den Watchdog – keine Reconnects, keine Buffer-Überläufe. Die Logs laufen stabil, alle Kanäle synchronisiert. Trotzdem notiere ich ein leichtes Zittern im TSC-Graph um Mitternacht. Vielleicht ein thermischer Effekt, vielleicht das Haus, das sich im Nachtwind bewegt. Die Donau rauscht fern und leise, und ich denke daran, dass Zeit auch im Wasser fließt, nicht nur im Oszillator.

Kurz vor dem Schlafengehen schalte ich das Display aus. Der Rechner arbeitet weiter, still, unter dem Vordach draußen. Ich lasse ihn allein, so wie man einem Freund vertraut, der eine Nacht lang aufpasst. Morgen früh werde ich sehen, wohin uns die Drift geführt hat.

Dann pack ma’s – aber erst nach dem ersten Kaffee. Bis dahin darf die Zeit sich selbst beobachten.

IQR, Bootstrap und Spacer-Tests

Unter dem Vordach riecht es nach kaltem Aluminium und leicht feuchtem Beton. Acht Grad, rund siebzig Prozent Luftfeuchte. Ich stelle den kleinen Tisch zurecht, richte das Oszilloskop am Rand des Tisches aus und schiebe das Notebook daneben. Seit der Zeitumstellung tanzt mir die Systemzeit davon – Sprünge von zwei bis achtzehn Sekunden, obwohl das GPS-Signal selbst keinen Mucks macht. Der 1PPS steht wie ein Uhrwerk, aber das System nicht. Heute will ich herausfinden, ob die Abstände der Spacer zwischen den Modulen einen Einfluss auf diese Drift haben.

Ich beginne mit der Null-Konfiguration, also ganz ohne Abstand. Die Module liegen plan aufeinander, nur das dünne Isolierband dazwischen. Nach einer Stunde sehe ich schon, dass die Standardabweichung der Offsets steigt. Der Median bleibt zwar stabil, aber der IQR – der Interquartilsabstand – öffnet sich wie eine kleine Schere. Ich notiere: „0 mm: IQR ≈ 5,1 s“. Nicht dramatisch, aber unruhig.

> „Mika, du host da no an Millimeter Spacer im Koffer, oder?“
>
> „Jo, des probier i gleich – vielleicht bringt’s a bissl Ruhe nei.“

Ich tausche die Distanzstücke aus. 0,5 mm – kaum sichtbar. Der Aufbau wirkt fast identisch, aber die Messung spricht eine andere Sprache. Nach den ersten 10 000 Bootstrap-Iterationen zeigt sich ein engerer Vertrauensbereich. Die Streuung sinkt deutlich, und die Zeitreihe wirkt fast

glatter. Der Bootstrap-Korrelationskoeffizient zwischen RF-Peak-Amplitude und Offset ist weiterhin insignifikant; kein klarer Zusammenhang. Trotzdem spüre ich, dass hier etwas Physisches mitspielt – vielleicht ein Resonanzeffekt zwischen den Leiterplattenlagen.

Ich fahre den Lüfter auf 40 % und sehe, wie die Temperatur leicht fällt. Keine Änderung im 1PPS-Jitter. Die Hypothese, dass thermisches Rauschen den Drift treibt, scheint schwach. Interessanterweise bleibt die Varianz auch bei 1 mm und 2 mm Abstand wieder größer. Das Optimum liegt also genau dazwischen. Ich denke an die Nächte zuvor, an die vielen Stunden Log-Analyse. Jetzt schließt sich ein Kreis.

Ich öffne die Statistikdatei, schreibe in Großbuchstaben „BOOTSTRAP 10K (**0?**),5MM – STABIL“. Ein nüchterner Satz, aber er fühlt sich fei gut an. Dann berechne ich noch die korrigierten Quantile. Der Median der Abweichungen liegt bei 3,4 s, der IQR bei 1,2 s. Das ist fast zu schön, um Zufall zu sein. Ich führe einen kleinen Mann-Whitney-Test gegen die 0-mm-Daten durch. Die p-Werte flackern knapp unter 0,05 – nicht perfekt, aber andeutungsweise signifikant.

„Pack ma’s“, murmele ich. Ich will wissen, ob der Effekt reproduzierbar ist. Also starte ich die zweite Serie mit zufälliger Reihenfolge der Spacer und blinde die Zuordnung in der Datei. Erst später werde ich auflösen, welches Setup welches war. So vermeide ich, dass meine Erwartungen das Ergebnis färben. Nach weiteren zwei Stunden sind die Daten im Kasten. Wieder stabil bei 0,5 mm. Der Rest flattert.

> „Also doch a mechanische Kopplung?“, frage ich mich halblaut.
>
> „Könnt scho sei“, antwortet die Stimme in mir, die längst zu einem nüchternen Laborpartner geworden ist.

Ich öffne das Diagramm mit den Bootstrap-CIs. Die grauen Bänder verengen sich dort, wo die Spacer stimmen. Die Linien wirken wie Atemzüge – ruhig, gleichmäßig. In solchen Momenten merke ich, dass Statistik keine trockene Disziplin sein muss. Sie kann erzählen, wenn man genau hinhört. Jeder Punkt, jedes Intervall, jede Abweichung hat etwas zu sagen, und ich lausche.

Während draußen das Licht unter dem Vordach langsam in den Abend kippt, läuft der Kernel-Logger im Hintergrund weiter. Ich sehe noch keine Spur von Clocksource-Wechsel oder TSC-Drift. Der Userspace bleibt ruhig, kein adjtimex-Alarm, kein systemd-timesyncd-Ruckler. Vielleicht hat der kleine Abstand tatsächlich eine große Wirkung – nicht elektrisch, sondern mechanisch, wie ein unscheinbarer Dämpfer zwischen zwei Welten.

Ich lehne mich zurück und notiere im Protokoll: „Spacer 0,5 mm → geringster IQR, stabilster Offset. Bootstrap bestätigt geringe Varianz; keine RF-Korrelation.“ Dann schließe ich die Datei. Der Lüfter summt noch leise nach. Es ist diese Art von Stille, in der man merkt, dass etwas verstanden wurde.

Wenn morgen die EM-Checks beginnen und ich den Kernel tiefer aufdrösele, will ich wissen, ob sich dieser Befund dort wiederfindet – irgendwo zwischen den Taktdomänen und den feinen Schwingungen des Metalls.

Kernel-Tracing und EM-Plan

Servus aus Passau – der Nebel hängt heut wieder tief über der Donau, so dicht, dass die Laternen nur kleine, matte Inseln aus Licht bilden. Ich hab das Gefühl, die Stadt hält den Atem an, während ich mich durch den Dunst zur Werkbank taste. Das Oszilloskop blinkt träge, die Kabel liegen ordentlich gerollt, wie auf Kommando sortiert. Seit dem gestrigen Spaziergang ohne Handy ist da eine ungewohnte Ruhe in mir. Ich merk, dass genau diese Ruhe mir hilft, den Kopf klar für die nächsten Schritte zu kriegen.

Heute geht's um den Clocksource-Wechsel. Der Kernel hat sich in den letzten Messreihen seltsam benommen – Sprünge von zwei bis achtzehn Sekunden, obwohl das GPS-1PPS stabil wie ein Metronom läuft. Ich will wissen, ob der Wechsel von `tsc` auf `hpet` oder `acpi_pm` im Hintergrund passiert, unbemerkt von der User-Ebene, oder ob der Sprung eine Folge des EM-Firmware-Timers ist, der in Mikrosekunden denkt, während ich in Sekunden messe. Das ist so ein Moment, in dem Technik fast poetisch wirkt: zwei Zeitquellen, die denselben Fluss betrachten, aber in unterschiedlichen Taktarten zählen.

Ich starte den Trace-Run. `trace-cmd` legt sich auf den Kernel wie feiner Staub, jedes Interrupt-Flackern wird erfasst. Nebenbei läuft `chronyc` im Hintergrund, protokolliert jede Drift. Ich setze Marker auf die 1PPS-Impulse, markiere Clocksource-Events und notiere die Sequence-Numbers im Log. Der Nebel draußen macht es fast leichter, sich zu konzentrieren – alles andere ist ausgeblendet, nur diese Signale existieren.

> „Passt des jetzt?“, ruft Andi von der anderen Seite des Labors.
>
> „Jo, fei, aber wart no a bisserl – die Marker vom letzten Durchgang müssen noch rein.“

Die Geräte summen gedämpft, als würden sie einander zuhören. Nach zehn Minuten seh ich den ersten Wechsel: `clocksource: timekeeping watchdog: Marking clocksource tsc as unstable`. Ich grinse. Genau das wollte ich erwischen. Die Spur zieht sich durchs dmesg-Log, und das 1PPS-Signal bleibt unbeirrt. Das bedeutet: Der Kernel selbst stolpert, nicht die Hardware. Vielleicht ist es das Zusammenspiel aus CPU-Frequency-Scaling und dem Energiemanagement-Modul, das ich noch nicht ganz verstanden hab.

Im nächsten Schritt plane ich die EM-Vorprüfung. Der Energie-Manager (EM) ist noch im Entwurfsstadium, aber ich will das Verhalten früh simulieren. Wenn der Scheduler später die Frequenzen nach Lastprofil anpasst, darf das Zeitgefühl des Systems nicht verrutschen. Ich bau also eine kleine Testreihe: CPU-Load-Peaks, dann Ruhephasen, dann wieder Peaks. Währenddessen beobachte ich, ob der Clocksource-Selector reagiert oder stur bleibt. Die Idee ist, den Übergang zu stabilisieren, bevor ich wirklich in den EM-Plan eintauche.

Das Tolle daran ist, dass sich hier Technik und Intuition treffen. Ich weiß, was die Register tun sollten, aber ich spür auch, wenn was nicht stimmt – wie ein Musiker, der merkt, dass ein Ton leicht daneben liegt. Vielleicht ist das der Grund, warum ich diese Arbeit so mag. Sie ist präzise, aber sie lebt. Jeder Trace erzählt eine Geschichte, und jeder Sprung im Zeitstempel ist wie ein kleiner Fehler im Takt, den man erst hört, wenn man genau hinhört.

Ich notiere: *Holdover-Test mit 0,5 mm Spacer erfolgreich, Drift unter 10 µs, Clocksource-Wechsel reproduzierbar*. Das klingt nüchtern, fast kalt, aber für mich ist es ein Zeichen, dass der Plan funktioniert. Ich stell mir vor, wie der EM-Plan später daraus Energieprofile ableitet – feine, adaptive Steuerung, die nicht nur Strom spart, sondern das System rhythmisch im Gleichgewicht hält.

Am Nachmittag, als der Nebel sich langsam hebt, geh ich kurz vor die Tür. Der Asphalt ist feucht, und in der Ferne höre ich das Dröhnen eines Frachters auf der Donau. Ich denk an die Daten, die gerade im Hintergrund laufen, und an die Muster, die sie formen. Der EM-Plan wird diese Muster irgendwann automatisiert erkennen müssen. Noch ist alles Handarbeit, jeder Parameter bewusst gesetzt, jede Schwelle manuell kalibriert. Aber das ist der Weg: vom manuellen Tuning zum selbstlernenden System.

> „Mika, brauchst no a Kaffee?“, fragt Andi.
>
> „Na, passt scho. Wenn der nächste Run durch is, dann vielleicht.“

Ich schau wieder auf den Bildschirm, seh die Zeitleiste der letzten Stunde. Kein Sprung mehr. Nur gleichmäßige, ruhige Marker – 1PPS, Kernel-IRQ, User-Sync. Es fühlt sich an, als hätte das System kurz den Atem angehalten und dann beschlossen, wieder im Takt zu gehen. Ich speicher die Logs und leg den EM-Plan-Entwurf daneben. Noch roh, aber greifbar.

Draußen zieht der Nebel ab, und das Licht bricht endlich durch. Ich weiß, dass die nächsten Tage intensiver werden – Analyse, Vergleich, vielleicht ein erster Prototyp. Aber für den Moment genügt mir die Ruhe zwischen zwei Messpunkten, das Gleichgewicht zwischen Technik und Stille. Und irgendwo darin liegt schon der Anfang des nächsten Kapitels.

Nebel über Passau und der erste Trace-Run

Der Morgen kam gedämpft, als hätte jemand die Stadt in Watte gepackt. Über den Dächern hing der Nebel so dicht, dass selbst der Dom nur schemenhaft zu erkennen war. Ich stand auf dem Balkon, die Hand an der Reling, und beobachtete, wie kleine Tropfen an der Antenne perlten. Die Luft roch nach feuchtem Metall und kaltem Stein. Servus, dachte ich leise zu mir selbst, heut wird’s ernst.

Im Labor summten die Geräte schon im Halbdunkel. Der Trace-Server zeigte die letzten Logs der Nacht – nichts Dramatisches, aber wieder diese unregelmäßigen Zeitsprünge. Zwei bis achtzehn Sekunden Abweichung, als würde die Zeit selbst kurz stolpern. Ich atmete ruhig durch. Heute wollte ich das Trace-Setup endlich stabilisieren, den 1PPS-Marker sauber setzen und den ersten vollständigen Run fahren. Nicht mehr nur kurze Tests, sondern eine echte Sequenz.

Das GPS-Signal kam erstaunlich klar durch den Dunst, die LED blinkte im präzisen Sekundentakt. Ich prüfte die Koax-Verbindung, der Spacer von einem halben Millimeter saß perfekt. Das kleine Stück Metall war unscheinbar, aber entscheidend – es glättete die winzigen Offsets, die sich sonst in den Messreihen aufschaukelten. Manchmal sind’s die kleinsten Dinge, die das ganze Bild verändern.

Drinnen lief der Kernel-Tracer im DEBUG_TIMEKEEPING-Build, die Marker-Routine wartete auf den ersten Impuls. Ich tippte die letzten Befehle in die Konsole und setzte einen Test-Marker:

> “PPS-Marker erkannt, Timestamp in Queue,” meldete das System nüchtern.
>
> “Passt,” murmelte ich, “pack ma’s.”

Dann begann der erste echte Trace-Run. Sekunden wurden zu Datenpunkten, jeder Puls des GPS-Empfängers zeichnete eine Linie im Speicher. Ich sah den Verlauf fast körperlich vor mir, wie ein feines Netz aus Zeit, das sich über den Prozessor legte. Draußen rollte der Nebel

langsam vom Fluss herauf, und Passau wirkte, als hielte es den Atem an.

Zwischendurch legte ich eine kurze Pause ein, trank einen Schluck lauwarmen Kaffee und beobachtete das Oszilloskop. Die 1PPS-Impulse kamen regelmäßig, sauber getrennt, und doch war da dieser eine Sprung nach etwa fünf Minuten: ein wackelnder Marker, kaum sichtbar, aber da. Ich zoomte hinein, prüfte die Clocksource-Wechsel – TSC zu HPET und wieder zurück. Wie kleine Wellen auf einem stillen See. Der Kernel war vorsichtig, fast nervös.

Ich schrieb die Beobachtung in mein Notizbuch, Datum, Uhrzeit, Offset, dann wieder zur Konsole. Der Parser arbeitete inzwischen parallel und taggte automatisch die adjtimex-Events. Die neue Routine funktionierte besser als erwartet; sie erkannte die Clocksource-Switches präzise und markierte sie farblich im Diagramm. Es war, als würde der Code langsam lernen, selbst zu sehen.

Gegen Mittag lockerte sich der Nebel etwas, und Licht fiel schräg auf die Antenne. Ich spürte, wie die Spannung nachließ. Das System lief stabil, die Datenreihe füllte sich gleichmäßig. Nur ab und zu ein kleiner Spike, vermutlich EM-Einfluss vom Netzteil oder ein kurzer Watchdog-Reset. Ich markierte die Stellen und notierte mir, morgen einen EM/RF-Check durchzuführen. Routine, aber wichtig.

Der Nachmittag verging in diesem ruhigen Rhythmus aus Beobachten, Justieren, Warten. Ich hörte kaum etwas außer dem gleichmäßigen Ticken der PPS-LED und dem leisen Surren des Lüfters. Irgendwann kam eine Nachricht von einem Kollegen: „Wie läuft’s mit dem Trace?“ – „Stabil,“ schrieb ich zurück, „aber fei spannend, was die Clocksource da treibt.“ Ein Smiley, mehr brauchte es nicht.

> “Mika, du redest mit der Zeit, als wär’s a Lebewesen,” hatte mal jemand gesagt.

Vielleicht hatte er recht. Zeit war in diesen Momenten kein abstrakter Parameter, sondern etwas Greifbares, das atmete, sich dehnte und manchmal zögerte. Ich konnte es fast fühlen, wenn ein Marker zu früh oder zu spät kam.

Als der Run nach über drei Stunden endete, war der Speicher voll. Ich stoppte die Aufzeichnung, speicherte die Ergebnisse und ließ mir das Diagramm ausgeben. Die Linien zogen sich wie feine Adern über den Bildschirm, ein Muster aus Stabilität und Unruhe zugleich. Der erste Trace-Run war abgeschlossen – und er war sauber genug, um darauf aufzubauen.

Ich lehnte mich zurück, hörte draußen das ferne Glockengeläut durch den Nebel dringen. Die Stadt schien wieder Form anzunehmen, als würde sie langsam aus dem Schlaf erwachen. Es war ein guter Moment, still und konzentriert.

Morgen, dachte ich, kommt der Matrix-Test. Dann wird sich zeigen, wie tief der Nebel wirklich reicht.

Clocksource-Vergleiche und adjtimex-Snapshots

Der Himmel war wieder grau, so ein gleichmäßiges Bedecken, das alles ein bisschen gedämpft klingen ließ. Ich saß mit dem kleinen Loggerkasten draußen, etwa zehn Grad, kaum Wind. Servus, sagte ich zu mir selber, bevor ich die erste Messreihe startete. Die CPU lief fix im performance-Modus, kein Governor-Hopping, keine Zeitsync-Dienste im Hintergrund. Ich wollte die Zeit selbst sprechen lassen.

Die Idee war einfach, aber heikel: hochfrequente adjtimex-Snapshots vor und nach jedem Clocksource-Wechsel zwischen TSC und HPET. Alle dreißig Sekunden ein zusätzlicher Snapshot, zehn Minuten lang. Das Muster sollte zeigen, ob die Offsets direkt beim Umschalten springen oder erst danach ins Rauschen diffundieren. Ich hatte die 1PPS-Leitung sauber terminiert, die Marker knackten präzise im Log, und der Parser war frisch erweitert – Median, IQR, Bootstrap. Ein kleines Labor im Nebel.

> „Wenn du jetzt wechselst, springt sie wieder, wetten?“
>
> „Na, schau ma, vielleicht bleibt’s diesmal ruhig.“

Als der erste Switch kam, hörte ich kein Klicken, keine Bewegung, nur die milde Neukalibrierung der Zeitbasis. Doch im Log, kaum sichtbar, zeigte sich ein feiner Sprung: 4,7 µs Offset gegen den Median. Das war noch im normalen Streubereich, aber ein paar Minuten später, beim nächsten Wechsel, stieg der Residual-Offset auf über 20 µs. Da war sie, die kleine Unruhe zwischen den Quellen. Ich markierte sie mit `jump_flag`, ließ den Bootstrap-Filter drüberlaufen und sah die Verteilung leicht kippen. Die Statistik atmete.

Während der Regen begann, legte ich einen dünnen Spacer unter die Platine, kaum einen halben Millimeter. Das Gehäuse vibrierte sonst zu stark, wenn der Lüfter der Stromversorgung anlief. Der Spacer dämpfte, und vielleicht – dachte ich fei – beeinflusste er damit auch die Mikrodrift des Quarzes. Ein Gedanke, halb technisch, halb abergläubisch, aber in der Messtechnik sind’s oft die winzigen Dinge, die den Ausschlag geben.

Ich ließ das System zehn Minuten im TSC-Modus laufen, dann manuell rüber auf HPET. Wieder Snapshots, wieder das leise Zittern der Zahlen. Der Median blieb erstaunlich stabil, doch der Interquartilsabstand zog leicht auf. Der Bootstrap zeigte eine zweite Mode, als würde sich irgendwo im Kernel ein zweiter Zeitpuls melden. Ob das DEBUG_TIMEKEEPING-Flag selbst den Ablauf beeinflusste? Möglich. Ich notierte es, markierte die Stelle im Log und atmete den feuchten Geruch von Regen und Elektronik.

Die Offsets begannen, eine Geschichte zu erzählen: kleine Sprünge, dann längere Plateaus, als würden die Quellen versuchen, sich gegenseitig einzuholen. Ich rechnete den Drift pro Minute aus, nur aus Neugier, und sah die Linie sanft ansteigen. Die Skala der Abweichung war winzig, aber sie fühlte sich an wie eine Bewegung im Inneren des Systems, als ob die Zeit nicht bloß vergeht, sondern tastet.

> „Pack ma’s, eine Serie noch“, murmelte ich, als der Logger piepte.

Die letzte Messung des Tages: HPET zurück auf TSC, sofortiger Snapshot, dann 1PPS-Marker. Der Offset sprang diesmal nicht, sondern verschob sich langsam über drei Sekunden um knapp zwei Mikrosekunden. Kein echter Sprung, eher ein Gleiten. Vielleicht hatten sich die PLL-Parameter eingependelt, vielleicht war’s bloß Zufall. Ich fror die Daten ein, sicherte sie als Matrixgrundlage für die nächste Testserie. Jede Zeile ein Versuch, jede Spalte eine Spacer-Variante. Der Anfang einer größeren Struktur.

Zwischendurch wischte ich den Bildschirm frei; der Regen hatte feine Tropfen auf das Glas gesetzt, und das Display spiegelte den grauen Himmel. Ich dachte kurz an all die Stunden, die ich schon in diesen Messungen verbracht hatte, und an das seltsame Gefühl, wenn ein paar Mikrosekunden plötzlich Bedeutung bekommen. Zeit ist hier kein philosophisches Symbol, sondern ein Feld voller Messpunkte – und doch klingt sie manchmal nach Atem.

Als ich das Setup abschaltete, blieb das kleine Delta zwischen den Clocks bestehen, so, als hielten sie kurz noch Kontakt. Ich notierte den Mittelwert, speicherte den Bootstrap-Report und sah auf die Matrix-Vorbereitung: Morgen würde ich die Spacer-Varianten erweitern und 24 Stunden Holdover laufen lassen. Dann würde sich zeigen, ob die Sprünge nur Übergangserscheinungen sind oder Teil eines tieferen Musters.

So endete der Tag ruhig, nur das Ticken der Referenzuhr im Hintergrund. Ich wusste, dass das nächste Kapitel warten würde – die Matrixserie, und vielleicht endlich ein klarer Blick auf das Verhalten der Zeit im System.

Bootstrap-Matrix und persistente Traces

Der Abend war kühl, die Donau roch nach Metall und feuchtem Holz. Ich hatte das Notebook auf dem Balkon stehen, direkt neben dem kleinen GPS-Empfänger, der an der Wand summte wie ein geduldiges Insekt. Die Bootstrap-Matrix wuchs Zeile um Zeile, jede Wiederholung ein Hauch von Zufall, gebündelt zu einer Struktur, die sich fast organisch anfühlte. Ich prüfte die Konfidenzen, diesmal nicht bloß im Mittel, sondern in der Breite: Fenstergrößen variieren, Trace-Filter prüfen, C-States vorbereiten – das war der Plan, und er ließ sich nicht in einer einzigen Nacht erledigen.

> „Passt des so, Mika?“ fragte die Stimme aus dem Kopfhörer, ein Kollege aus Regensburg, der die Daten gegenrechnete.
>
> „Jo, passt fei. Aber i glaub, da Median mag no ned ganz so, wia ma wolln.“

Ich grinste kurz, während der nächste Bootstrap-Durchlauf startete. Zwanzigtausend Resamples, diesmal mit leicht verschobenen Fenstern von ±3 s bis ±8 s. Der Lüfter drehte hoch, und das rhythmische Ticken des GPS-1PPS überlagerte das Rauschen der Donau. Ich beobachtete, wie die Matrix-Zellen aufleuchteten, dann wieder verblassten; jede Zahl ein kleiner Versuch, Ordnung in das Chaos der Zeitquellen zu bringen.

Die Clocksource-Switches hatten mich schon seit Wochen beschäftigt. Immer wieder diese Sprünge, kaum messbar, aber spürbar. Wenn man lange genug hinschaut, fängt man an, den Takt zu hören, der zwischen den Prozessorkernen pulsiert. Ich hatte die performance-Governor-Einstellung fixiert, Spacer von 0,5 mm weiterverwendet, alle Synchronisationsdienste pausiert. Der Debug-Mode von timekeeping meldete nichts Auffälliges, und doch waren die Traces nicht ganz sauber. Vielleicht war es der Übergang zwischen den C-States, vielleicht ein Rest von thermischem Rauschen.

Ich öffnete die persistente Trace-Session. Der Speicher war knapp, aber ich wollte diesmal alles behalten – kein temporäres Logging, keine flüchtigen Buffers. Nur so würde sich zeigen, ob die kleinen Zeitsprünge tatsächlich in den Kernel-Pfad wanderten oder bloß Artefakte der Messung waren. Es war fast Mitternacht, als ich die erste Serie startete. Draußen klirrten zwei Fahrräder über das Kopfsteinpflaster, dann wieder Stille.

> „Wenn’s stabil bleibt, kann i mi morgen an die C-States wagen,“ murmelte ich und notierte die letzten Werte.

Die Bootstrap-Konfidenz lag um 1,1 s, das Intervall schloss sich langsam. Die persistenten Traces zeigten eine leichte Drift, etwa 0,2 µs pro Sekunde, kaum der Rede wert, aber da. Ich fragte mich, ob die CPU ihre Ruhephasen anders taktet, wenn die Donau unter dem Balkon

reflektiert. Abwegig, klar, aber nach etlichen Nächten in diesem Rhythmus beginnt man, selbst solchen Ideen nachzuspüren.

Die variablen Fenstergrößen offenbarten ein Muster: Jedes Mal, wenn ich das Intervall verkleinerte, stabilisierte sich der Median, aber die Varianz der Sprünge nahm zu. Es war, als ob das System zwischen zwei Gleichgewichten schwang. Ein schmaler Grat zwischen Präzision und Überinterpretation. Ich atmete tief durch. Servus Geduld, dachte ich, pack ma’s nochmal an.

Am nächsten Morgen, kurz nach Sonnenaufgang, saß ich wieder am Schreibtisch. Die Donau war still, fast spiegelglatt. Ich hatte alle Kabel noch einmal überprüft, die Steckdosen sauber beschriftet, das Mini-GPS blinkte regelmäßig. Der Logger meldete weiterhin kleine Zeitsprünge, aber sie wirkten jetzt fast wie Atemzüge des Systems, nicht mehr wie Fehler. Ich öffnete die letzte Datei der Nacht, sah die Matrix in Graustufen und notierte: *Persistente Traces stabil, Filtereffekt sichtbar, nächste Phase: C-State-Profilierung.*

Es war keine spektakuläre Erkenntnis, eher ein ruhiges Einrasten der Zahlen in ein größeres Bild. Die Bootstrap-Matrix war kein statisches Artefakt mehr, sondern ein lebendiger Teil des Experiments. Ich lehnte mich zurück, hörte das leise Klicken des Relais am GPS-Empfänger und dachte daran, wie viele solcher Nächte noch folgen würden. Die Donau draußen glitzerte im ersten Licht, und ich wusste, dass das nächste Kapitel tiefer in den Schlafzustand der Kerne führen würde, dorthin, wo selbst die Zeit kurz innehält.

BPF-Probes gegen kprobes

Ich hatte mir vorgenommen, diesmal nicht nur auf die Zahlen zu schauen, sondern ihre Bewegung zu verstehen. Die alten kprobes-Messungen lagen noch im Speicher, roh, unruhig, mit einem Rauschen, das irgendwie nach Scheduler roch. Daneben meine neuen BPF-Probes, fast identisch im Aufbau, aber deutlich leichter, als würde jemand die gleiche Melodie auf einer dünneren Saite spielen. Servus, sagte ich leise zu mir selbst, und dann pack ma’s.

Ich begann mit dem Vergleich der Latenzen. Für jede der 48 gepaarten Messungen nahm ich den Median – BPF gegen kprobe, Paar für Paar. Die Rohwerte schwankten zwischen vier und acht Millisekunden. Im Durchschnitt schien BPF etwa eineinhalb Millisekunden schneller zu sein, aber ich wollte mich nicht auf den ersten Eindruck verlassen. Ich baute den Bootstrap-Zyklus auf, 10.000 Resamples, mit Replacement, die Differenzen immer neu gezogen. Nach einer Stunde Laufzeit lag die Verteilung stabil: der Median der Differenzen 1,75 ms, das 95 %-Konfidenzintervall von 0,42 ms bis 3,12 ms. Ich lehnte mich zurück, hörte den Lüfter der Workstation gleichmäßig rauschen.

> „Das schaut fei gar ned schlecht aus“, murmelte ich, während ich die Histogramme überprüfte.

> „Ned schlecht, aber halt erst Statistik, koa Erklärung“, antwortete ich mir selbst.

Der Mann-Whitney-Test war mein nächster Schritt. Ich hatte ihn gewählt, weil ich den Verteilungen nicht traute – zu viele leichte Schiefe, zu viele kleine Ausreißer. Als der p-Wert bei 0,028 stehen blieb, atmete ich ruhig aus. Kein magischer Schwellenwert, aber doch ein Fingerzeig: Die BPF-Probes waren mit hoher Wahrscheinlichkeit wirklich schneller. Keine Illusion durch Cache-Effekte oder zufällige Interrupt-Lücken. Ich dachte an die alten Messungen, wo Sekundenunterschiede auftauchten. Jetzt, mit diesen Millisekunden, war klar,

dass jene gigantischen Offsets einen anderen Ursprung gehabt hatten. Vielleicht ein Clocksource-Wechsel, vielleicht die Spacer-Geometrie, vielleicht schlicht ein unglücklicher Trace-Reset.

Ich öffnete das Log der Bootstrap-CI-Matrix aus den früheren Balkon-Sessions. Dort hatte ich schon gesehen, wie sich das Konfidenzintervall bei größerem n verengt. ±2 s, ±5 s, ±10 s Fenster, 1000 vs. 10000 Resamples. Es war fast poetisch, wie sich aus Zufallsschichten Sicherheit formte, millimeterweise. Der 0,5 mm-Spacer, den ich damals zwischen Sensorplatte und Chassis gesetzt hatte, hatte einen Median-Offset von etwa 1,12 s gezeigt – ganz andere Größenordnung, aber die Logik war dieselbe: erst durch Wiederholung und Resampling wurde das Bild klar.

Ich stellte mir die beiden Methoden nebeneinander vor, wie zwei Mechaniker über demselben Motor arbeiten. Kprobe bohrt sich tief in den Kernel, greift direkt auf Symbole, manchmal ruppig, manchmal elegant. BPF dagegen hängt sich in dieselben Punkte, aber durch eine Schicht Just-in-Time-Kompilation, sicherer, flexibler. In Zahlen übersetzt: weniger Overhead, weniger Kontextwechsel. Ich mochte diese Klarheit. Und doch, die Unterschiede blieben subtil. Eineinhalb Millisekunden – gerade genug, um in einem eng getakteten System eine Spur zu hinterlassen, zu wenig, um den Nutzer spüren zu lassen.

Ich ließ die Plots sprechen. Die BPF-Kurve war schmal, konzentriert, fast diszipliniert. Die kprobe-Kurve hatte eine leichte Schulter nach rechts, als würde sie manchmal stolpern. Ich zoomte hinein, suchte nach den Ursachen. Kein klarer Ausreißer, kein einzelner Prozess. Vielleicht die Art, wie kprobe das Stack-Trace-Format schreibt, vielleicht der Weg über den perf-Buffer. Ich notierte mir die Hypothese für später.

Die Bootstrap-Confidence-Intervalle gaben mir das, was ich brauchte: eine quantitative Grenze. 0,42 bis 3,12 ms Differenz – klein, aber konsistent. Ich nahm es als Beweis, dass die Latenz nicht nur messbar war, sondern regelhaft. In dieser Stetigkeit lag eine Art Ruhe. Ich erinnerte mich, wie ich in den frühen Tagen mit Sekundenwerten jonglierte, unsicher, ob der Kernel selbst gegen mich arbeitete. Jetzt war das System still, die Messung präzise, die Unterschiede fein genug, um Vertrauen zu wecken.

Ich führte noch einen letzten Bootstrap-Lauf durch, diesmal mit n = 1000 statt 10000, um die Stabilität zu testen. Das Intervall weitete sich leicht, blieb aber im selben Bereich. Ich konnte zusehen, wie die Unsicherheit atmete, größer wurde, wenn ich weniger zog – ein schönes Beispiel für Statistik als lebendigen Prozess.

Als die Sonne durch das Fenster fiel, dachte ich kurz an die Community-Fragen, die noch offen waren: Wie filtert man in trace-cmd effizient, ohne Events zu verlieren? Welche Spacer-Eigenschaften beeinflussen thermisch die Taktstabilität? Ich lächelte. Diese Themen würden bald wiederkommen.

Ich schloss das Terminal, ließ die Zahlen nachklingen. BPF gegen kprobe – kein Wettkampf, eher ein Dialog zweier Werkzeuge, jedes mit seiner eigenen Handschrift. Die Millisekunden sprachen klar genug. Ich spürte, dass das nächste Kapitel wieder hinausführen würde – zu den Spacer-Varianten, zu den C-States, zu den feinen Übergängen zwischen Hardware und Schlaf.

Trace-Buffer und Filter-Optimierung

Der Wind war trocken an diesem Nachmittag, kaum drei Grad über Null, als ich den kleinen Laptop wieder unter das Vordach stellte. Der Akku hielt erstaunlich gut, trotz der Kälte. Ich hatte mir vorgenommen, die Buffergrößen endlich systematisch zu testen – nicht mehr nur in kurzen Sessions, sondern über längere, persistente Traces hinweg. Ein halber Millimeter Spacer war diesmal zwischen den Sensorplatten, gerade genug, um den elektromagnetischen Einfluss messbar zu machen. Der Trace-Buffer lief stabil, und nach den ersten fünfzig Clocksource-Switch-Events zeichnete sich ein vertrautes Muster ab: Die Latenzen tanzten im Bereich zwischen drei und fünfzehn Millisekunden.

Ich saß still da, schaute auf die Zahlen, und dachte an die Drops, die mich letzte Woche fast in den Wahnsinn getrieben hatten. Diese winzigen Lücken, die plötzlich jede Statistik verzogen. Heute wollte ich sie eliminieren, vollständig. Also fuhr ich den Buffer hoch, kontrollierte die Trigger-Tiefe, erhöhte das Sampling leicht und legte einen neuen Filterpfad an, der die adjtimex-Snapshots enger an den Kernel-Trace koppelte. Das war die erste echte Annäherung an Konfiguration B – die Version, die später produktiv laufen sollte.

> „Wenn’s diesmal droppt, dann geb i’s auf“, murmelte ich halblaut.

Der Buffer blieb ruhig. Keine Drop-Meldung, kein Warnflag. Nur das leise Takten des 1 PPS-Signals und die fließende Liste von Events. Ich überprüfte den Median-Offset: 6,8 ms, mit einem 95 %-Intervall zwischen 3,2 und 15,4 ms. Das passte perfekt zu den Bootstrap-Werten vom Vortag. Ich lehnte mich zurück, atmete aus. Es war fast poetisch, wie sich die Zahlen ineinanderfügten, als hätte das System verstanden, was ich von ihm wollte.

Der nächste Schritt war klar: Die Filterlogik sollte nicht nur stabil, sondern auch effizient sein. Ich hatte in den letzten Tagen verschiedene BPF-Probes gegen klassische kprobes getestet. Der Unterschied war deutlicher, als ich erwartet hatte. BPF reagierte schneller, im Median etwa 1,6 bis 1,8 ms vor den kprobes. Das war kein Zufall mehr – der Mann-Whitney-Test bestätigte es mit $p \approx 0{,}03$. Ich notierte die Werte in mein Notizbuch, nicht weil die Datei sie nicht speichern würde, sondern weil ich das Gefühl brauchte, die Linie selbst zu ziehen. Ein Strich in Graphit, der die digitale Präzision erdete.

> „Schau, BPF is fei a sauberes Zeug“, sagte ich leise in den Wind, als würde der Laptop zuhören.

Ich verglich die letzten Serien und bemerkte, dass der Trace-Buffer bei größeren Event-Fenstern zwar mehr Speicher fraß, aber dafür keinerlei Drops mehr meldete. Die Filter-Optimierung griff: durch adaptives Re-Sampling der adjtimex-Snapshots wurden die Bursts geglättet, die Peaks seltener. Die CPU-Last blieb konstant unter zwanzig Prozent. Ich begann, die Parameter für Konfiguration B zu fixieren – eine Art Referenzpunkt zwischen Stabilität und Reaktionsgeschwindigkeit. Die Spacer-Matrix wollte ich am nächsten Tag erweitern, von null bis zwei Millimeter, jeweils mit mindestens fünfzig Events pro Distanz. So würde sich zeigen, ob der Filter auch unter veränderten mechanischen Bedingungen sauber blieb.

Als die Sonne tiefer sank, legte sich ein orangefarbenes Licht über den Tisch. Die Zahlen auf dem Bildschirm wirkten plötzlich wärmer, als wollten sie mir danken, dass ich sie endlich verstanden hatte. Ich speicherte die Session, prüfte die Logs und sah, wie die letzten Zeilen ohne Fehler durchliefen. Keine Drops, kein Buffer-Overflow, keine Warnungen. Nur ein stilles Ende einer sauberen Messung.

Ich dachte kurz an die vielen kleinen Schritte, die hierher geführt hatten. An den ersten Versuch, als der Buffer nach wenigen Sekunden kollabierte. An die Nacht, in der ich die Filterformeln neu schrieb, Zeile für Zeile. Und an den Moment heute, als alles einfach funktionierte. Vielleicht, dachte ich, war das der Punkt, an dem Technik und Geduld sich trafen – irgendwo zwischen den Takten einer Uhr und dem Atem eines Menschen.

Der Laptop piepte leise, der Trace war abgeschlossen. Ich schloss den Deckel, zog die Mütze tiefer und sah über den Hof. Morgen würde ich die Spacer-Reihe vollenden und die Konfiguration B endgültig verifizieren. Aber für heute war der Buffer ruhig, und das System atmete gleichmäßig. So sollte es bleiben – bis das nächste Kapitel beginnt.

Single-File-Mode und Kernel-Ursprung

Der Morgen begann still, fast zu still. Ich hatte den Rechner im Single-File-Mode hochgefahren, um jedes Event sauber zu isolieren. Kein Netzwerk, kein Nebenprozess, nur der Kern, der sich selbst lauscht. Draußen hing der Nebel wie ein weiches Tuch über der Donau, und das kleine GPS an der Wand blinkte träge, als würde es die Zeit selbst ausatmen.

Ich wollte die Event-Vollständigkeit prüfen, jedes Paket, jeden Impuls. Die letzten Tage hatten gezeigt, dass winzige Offsets die Synchronisation verschoben hatten, kaum messbar, aber genug, um die Kurven leicht zu verziehen. Ich erinnerte mich an Michaels Trick – den Clock-Shift über BPF – und überlegte, ob ich ihn heute wagen sollte. Noch war ich unschlüssig. Ein falscher Schritt, und ich müsste den ganzen Stack neu anwenden.

> „Servus Kernel, gib mir a saubere Spur“, murmelte ich leise, halb im Spaß, halb im Ernst.

Die Konsole antwortete nur mit einem blinkenden Cursor. Ich mochte diese Art von Schweigen. Ein System, das nicht drängt, sondern wartet. In dieser Ruhe liegt etwas Menschliches, fei, und doch bleibt es präzise, unnachgiebig.

Ich startete den Trace-Recorder. Ein Strom von Zahlen, Zeiten, Offsets. Der Kernel verzeichnete jeden Interrupt, als wollte er mir beweisen, dass alles unter Kontrolle war. Doch ich wusste, Kontrolle ist nur eine Momentaufnahme. Ein kurzer Atemzug zwischen zwei Zuständen. Ich markierte die ersten Samples und sah, dass die Offsets tatsächlich vom Ursprung abwichen – ein Delta von kaum 0,004 Sekunden. Aber das genügte, um die Kurve zu verschieben. Ich ließ die Daten laufen, bis sie sich setzten, dann stoppte ich, atmete durch und notierte die Werte.

Die BPF-Ebene – Berkeley Packet Filter – hatte sich wieder als stiller Vorteil erwiesen. Ich konnte direkt am Ursprung lauschen, bevor der Kernel seine Finger im Spiel hatte. Alles, was danach kam, war Interpretation. Ich erinnerte mich an die Worte eines alten Mentors: *„Wer den Ursprung kontrolliert, versteht den Fluss.“* Damals hatte ich das für eine Metapher gehalten. Heute wusste ich, dass es eine technische Wahrheit war.

Ich öffnete den Log-Viewer im Split-Screen und verglich die Zeilen. Der Single-File-Mode hatte tatsächlich die Interferenz minimiert; die Events standen wie Soldaten in Reih und Glied, jedes mit sauberem Zeitstempel. Nur ein paar Mikrosekunden an Drift blieben übrig, wahrscheinlich durch den internen Takt des GPS. Ich tippte kurz an das Gehäuse, als könnte ich ihm danken. Das Leuchten blieb ruhig, kein Flackern, kein Zögern.

Draußen zog der Nebel dichter. Ich öffnete ein Fenster und ließ die feuchte Luft herein. Sie roch nach Wasser und Metall, nach Donau und Kondensat. In dieser Kälte klang jeder Tastendruck schärfer, fast wie ein kleiner Schlag auf Glas. Ich schloss die Augen und hörte dem System zu. Der Kernel lief, die Module griffen ineinander, und irgendwo tief im Stack summte eine Frequenz, die fast wie ein Herzschlag klang.

> „Passt des jetzt endlich?“ fragte ich leise. „Fast“, antwortete ich mir selbst.

Ich prüfte die Event-Sequenzen erneut. Keine Lücken, keine doppelten IDs. Das erste Ziel war erreicht: Vollständigkeit bestätigt. Nun zur Validierung des Offsets. Ich fuhr die Testreihe hoch, ließ zehn Zyklen laufen, jede Runde leicht verschoben im Zeitfenster. Der Vergleich ergab, dass die Drift sich gleichmäßig verteilte, kein Ausreißer, kein Sprung. Nur ein gleichmäßiges Atmen des Systems. Es war, als hätte ich den Ursprung wieder eingefangen.

Ich schrieb eine kurze Notiz: *„Offset-Ursprung stabil, BPF-Pfad bestätigt.“* Dann lehnte ich mich zurück und ließ den Bildschirm in Ruhe flimmern. Die Zeilen liefen weiter, aber sie bedeuteten mir jetzt weniger als das leise Summen im Hintergrund. Technik kann manchmal mehr über dich erzählen, als du über sie weißt.

Im Nebel draußen bewegte sich etwas – vielleicht nur ein Vogel, vielleicht das Echo meiner eigenen Gedanken. Ich dachte an die Donau, wie sie gestern still war, als ich ohne Handy dort stand. Sie hatte keine Uhr, keinen Offset, keinen Kernel. Nur Fluss. Vielleicht ist das der wahre Ursprung: Bewegung ohne Maß.

Ich schloss die Session, speicherte das Log als Einzeldatei und sah zu, wie der Cursor erlosch. Der Raum wurde still, nur das GPS blinkte weiter, geduldig, unbeirrt. Ich fühlte mich ruhig, fast wie entkoppelt von der Zeit. Der Single-File-Mode hatte nicht nur das System vereinfacht, sondern auch mich selbst. Alles war eins, und das genügte.

Ich notierte die letzten Werte, klappte das Notebook zu und sagte leise: „Pack ma’s morgen weiter.“ Dann blieb nur noch das leise Leuchten an der Wand, das mich an den nächsten Schritt erinnerte – an das Kapitel, das noch kommen würde.

Spacer-Matrix und statistische Bestätigung

Ich saß wieder auf dem Balkon, der Wind kam diesmal von Westen, kühl und klar. Die Sonne stand flach über der Donau, und auf dem Display spiegelte sich das Geländer wie ein Raster aus Licht. Ich hatte alles vorbereitet: die schmale Filterkonfiguration, 32 MB Buffer, die bewährte clocksource_switch-Selektion. „Pack ma’s“, murmelte ich und startete die ersten Spacer-Runs. Es ging jetzt um die Verifikation des 0,5 mm-Effekts – diese minimale, aber wiederkehrende Abweichung, die bisher nur als Hypothese existierte.

Die Matrix bestand aus vier Varianten: verschiedene CPU-Governor-Einstellungen und C-State-Limits, jeweils mit identischem Tracing-Setup. Ich wollte wissen, ob das Phänomen bloß eine Streuung in den Messdaten war oder ob sich eine physische Kopplung zwischen Spacer-Abstand und Taktquelle manifestierte. Die ersten Minuten liefen ruhig, keine Drops, keine Warnungen. Der Kernel antwortete präzise, fast stoisch.

> „Wenn du dich wiederholst, bist du vielleicht schon stabil“, dachte ich und notierte die Zeitmarken.

Nach dem dritten Run zeichnete sich ein Muster ab. Die 0,5 mm-Abweichung tauchte tatsächlich immer dann auf, wenn der Governor in den Performance-Modus wechselte. Die Latenzverteilung blieb eng, median bei 6,8 ms, wie schon an Tag 67. Aber die Streuung in den Event-Offsets nahm leicht zu, sobald der Spacer minimal anders saß. Die Differenzen waren winzig, im Bereich von Mikrosekunden, doch sie wiederholten sich über mehrere Dutzend Events hinweg. Statistisch signifikant? Noch nicht ganz, aber nah dran.

Ich wechselte in den Single-File-Mode, um die Persistenz der Daten zu prüfen. Wieder 62 vollständige Events, keine Drops. Das war fast schon unheimlich sauber. Ich legte die Ergebnisse übereinander, nutzte einfache Korrelationsprüfungen und Bootstrap-Resampling, um die Signifikanz zu testen. Der p-Wert fiel unter 0,07 – nicht perfekt, aber auffällig. Es war, als würde der Spacer selbst ein leises Echo in der Taktquelle hinterlassen, ein winziger mechanischer Atemzug im digitalen Rauschen.

„Servus, Statistik“, sagte ich leise und lehnte mich zurück. Die Donau glitzerte, und für einen Moment wirkte alles synchron, als hätten selbst die Wellen ihren eigenen Taktgeber. Doch dann erinnerte ich mich an den Sprung von 1,11 Sekunden aus dem Vortag, jenen klaren Hinweis auf den Kernel-Ursprung der Offsets. Wenn diese Sprünge auch hier auftauchten, konnte man von einer physischen Kopplung kaum mehr sprechen – dann wäre es bloß ein Software-Artefakt, ein rhythmischer Fehler im System.

Ich prüfte die Logs, frame für frame. Kein Sprung, keine Diskontinuität. Nur diese subtile 0,5 mm-Signatur, sauber eingebettet in den Verlauf der clocksource_switch-Events. Ich begann zu rechnen: Wenn der Effekt wiederholbar war, musste er sich in der Standardabweichung spiegeln. Tatsächlich lag sie in den Runs mit Spacer-Kontakt leicht höher, um rund 0,3 ms. Diese Differenz war klein, aber konsistent. Ich notierte: „Hypothese nicht widerlegt.“

> „Fei interessant“, hörte ich mich sagen, halb ungläubig, halb zufrieden. „Vielleicht druck i mir des aus.“

In der folgenden Stunde überprüfte ich noch einmal die gesamte Matrix, diesmal mit leicht erhöhter CPU-Last. Ich wollte sehen, ob das Systemrauschen den Effekt verschluckte. Doch der Befund blieb bestehen: Der 0,5 mm-Effekt war robust gegenüber Lastschwankungen, verschob sich aber leicht im zeitlichen Verlauf, als ob eine thermische Drift mitspielte. Die Temperatur auf dem Balkon war inzwischen um drei Grad gefallen, und ich fragte mich, ob das schon reicht, um eine messbare Änderung in den Taktsignalen hervorzurufen.

Die statistische Bestätigung war noch kein Beweis, aber ein Schritt dorthin. Wahrscheinlich würde ich in den nächsten Tagen noch eine physische Kopplung simulieren müssen – etwa über einen gezielten Spacer-Impuls oder eine definierte Temperaturänderung. Erst dann ließe sich entscheiden, ob die Mechanik wirklich mit dem Takt tanzt oder ob das Ganze nur ein Spiel aus Messrauschen und Algorithmus ist.

Ich speicherte die Dateien ab, ließ den Rechner im Leerlauf und sah hinüber zum Fluss. Das Licht wurde blasser, die Ränder der Stadt verschwammen. Die Daten lagen nun in sicherer Ordnung, doch ihr Sinn war noch nicht ganz gefasst. Vielleicht brauchte ich eine Nacht Abstand, ein wenig Ruhe zwischen den Zahlen. Morgen würde ich die Korrelationen neu berechnen und sehen, ob sich die Spur des 0,5 mm-Effekts weiter verdichtet – bevor das nächste Kapitel beginnt.

EM-Sonde und elektrische Tests

Ich sitze wieder am Labortisch, das grüne Blinklicht vom Mini-GPS wirft seine sanften Reflexe auf die Aluplatte der EM-Sonde. Es ist der gleiche Tisch wie gestern, aber die Luft ist kühler, klarer. Die Donau draußen fließt ruhig, und obwohl ich drinnen bin, spüre ich ihren Rhythmus. Servus, denk ich mir leise, wie zu einem alten Kollegen, bevor ich den nächsten Messlauf starte.

Der Fokus heute: die HF-Reduktion messen. Ziel ist, dass die Störfelder, die um die EM-Sonde kreisen wie kleine unsichtbare Möwen, endlich ein stabiles Muster zeigen. Das Oszilloskop summt, die Kurve tanzt leicht, und ich beobachte, wie das Rauschen langsam abnimmt, sobald ich den Ferrit-Ring in Position schiebe. Die Messung läuft über mehrere Minuten, und jeder Ausschlag, der kleiner wird, ist wie ein Atemzug weniger Unruhe im System.

> „Passt der Offset noch?“ fragt Michael hinter mir, ohne aufzuschauen.
>
> „Noch ja“, antworte ich, „aber fei, der driftet leicht nach Süden.“

Er grinst, weil ich immer Himmelsrichtungen benutze, wenn ich über Spannungslagen rede. Für mich ist das System wie eine kleine Landschaft, und jedes Mikrovolt eine Bewegung darin. Die Offset-Konstanz zu prüfen, braucht Geduld. Ich lasse die Sonden ruhen, beobachte, wie die Temperatur um ein halbes Grad schwankt, und notiere jeden Schritt im Log. Die Zeitsprünge, die wir gestern im Logger fanden, sind verschwunden, seit ich den Clock-Trick von Michael eingebaut habe. Die interne Taktung synchronisiert sich nun mit dem GPS-Signal, und das kleine Blinklicht an der Wand scheint zufrieden zu nicken.

Ich erinnere mich an den Spaziergang gestern an der Donau, ohne Handy, ohne Geräusche außer dem Wasser. Da war dieser Schatten, den ich für eine Gestalt hielt, aber es war nur der Mast am anderen Ufer. Vielleicht ist das der Grund, warum ich jetzt ruhiger messe – weil ich weiß, dass viele Dinge erst dann klar werden, wenn man lang genug hinschaut, bis sie sich selbst erklären.

Zurück am Tisch starte ich den zweiten Run. Diesmal geht es um die Vorbereitung der C-State-Tests. Die Low-Power-Zustände des Prozessors sollen sich sauber mit den Sensorsignalen vertragen. Ich programmiere eine Sequenz, die alle Übergänge simuliert, und beobachte die Reaktionszeit der Sonde, wenn der Strom kurz absackt. Jede Millisekunde zählt. Die Daten fließen in feinen Linien über den Bildschirm, und ich zähle innerlich mit – eins, zwei, drei – bis das Signal wieder stabil ist. Pack ma’s, denk ich, das wird was.

Ein kurzer Tonaussetzer am Messgerät bringt mich zurück in die Gegenwart. Ich tippe ein paar Befehle, setze den Filter neu, und das Brummen verschwindet. Diese Momente sind wie Mini-Dialoge zwischen mir und der Technik, eine Sprache aus Impulsen und Reaktionen. Ich höre auf, zu denken, und lasse die Bewegung meiner Hände den Rhythmus bestimmen.

> „Wenn der Offset diesmal hält, brauch ma nimma tricksen“, murmelt Michael.
>
> „Wär schön“, sag ich, „aber i glaub, er will noch a bissl Aufmerksamkeit.“

Wir lachen leise, und die EM-Sonde blinkt, als hätte sie’s gehört. Es ist dieser eigenartige Moment, wenn Technik fast lebendig wirkt, weil sie auf jede kleine Veränderung reagiert. Ich dokumentiere die Werte, vergleiche sie mit der Simulation, und sehe, dass die HF-Reduktion tatsächlich um knapp zwölf Prozent besser ist als gestern. Das ist kein Zufall, sondern das Ergebnis stundenlanger Justierung. Ich notiere es mit einem leisen Stolz, fast wie ein Tagebucheintrag, nur technischer.

Die Stunden vergehen, und das Labor füllt sich mit dem leisen Surren der Geräte. Draußen färbt sich der Himmel über der Donau in ein blasses Blau-Grau, und das Licht spiegelt sich in der Metallkante des Tisches. Ich speichere die Daten, sichere sie doppelt, ziehe den letzten Stecker und lehne mich zurück. Die EM-Sonde ruht, als hätte sie verstanden, dass der Tag geschafft ist.

Ich öffne das Fenster, atme kurz die kalte Luft ein und höre das leise Gurgeln des Flusses. Morgen werden wir die C-State-Runs planen, und ich weiß, dass die heutige Ruhe sich darin fortsetzen wird. Manchmal sind es genau diese stillen Abende, in denen die Technik und die Donau denselben Takt atmen.

Governor-Vergleiche und konstante Offsets

Der Schnee war heute feiner als Staub, und doch legte er sich auf die Kabel, als hätte er Bedeutung. Ich saß unter dem Vordach, die Finger halb taub, und beobachtete, wie die kleinen Log-Zeilen über den Bildschirm liefen. Alles drehte sich um die Governor-Frage: powersave gegen performance. Zwei Welten, die sich im Millisekunden-Takt abwechseln konnten, aber selten friedlich koexistierten.

Ich ließ beide Modi parallel laufen, über je acht Sequenzen, so wie ich’s bei der Spacer-Matrix schon gemacht hatte. Die Konfiguration B blieb unverändert – nur die Zeitbasis hatte ich diesmal doppelt gesichert. Der Gedanke kam mir, dass sich die Offsets vielleicht gar nicht aus der Software ergaben, sondern als konstante Resonanzen im metallischen Träger fortsetzten, wie ein leiser Atem unter Strom.

„Hörst du das?“, fragte ich in Richtung der Messkabine. Die Antwort kam verzögert, durch Funk und Schnee: „Nur den Lüfter, Mika.“ Ich lächelte. Der Lüfter war tatsächlich das lauteste Element – aber die Peaks im HF-Spektrum, die hatten eine andere Sprache.

> Ich sah sie auftauchen, genau bei 1,11 s, wie eine wiederkehrende Welle – so präzise, dass sie fast tröstlich wirkte.

Das Muster war zu konstant, um zufällig zu sein. Ich verglich die Runs: powersave zeigte eine weiche Kurve, die Frequenzsprünge abgeflacht, als würde das System atmen dürfen. Performance dagegen war kantig, die Peaks schossen scharf nach oben, und die Offsets blieben auf die Millisekunde gleich. Kein Clocksource-Glitch, kein BPF-Delay, alles stabil – und doch anders als erwartet.

Ich nahm mir Zeit, die HF-Peaks mit den Event-Counts zu korrelieren. Im powersave-Modus ergab sich eine geringe Amplitudenstreuung, während unter performance die Peaks fast synchron mit den Jump-Counts kamen. Drei zu null, eins zu eins, zwei zu zwei – dieselbe Reihenfolge wie bei den mechanischen Spacer-Tests. Ich erinnerte mich an den Tag, als 0,5 mm Abstand plötzlich Stabilität brachte. Heute sah ich denselben Effekt, nur diesmal elektrisch, nicht physisch.

„Fei interessant“, murmelte ich und notierte mir die Phasenverschiebung zwischen den Peaks. Sie lag konstant bei 4,8 ms. Das mag klein klingen, aber im Kontext der Systemlatenzen war das ein fester Marker – eine Signatur, die sich quer durch alle Runs zog. Ich stellte mir vor, wie diese 4,8 ms wie kleine Inseln im Datenmeer schwammen, beständig, egal, welcher Governor regierte.

> Vielleicht, dachte ich, ist Stabilität gar kein Zustand, sondern das sich wiederholende Ungleichgewicht, das sich selbst erkennt.

Ich begann, die forcierte Laufreihe vorzubereiten. Dafür musste die Umgebung ruhiger sein: keine Hintergrundprozesse, keine variablen C-States. Ich deaktivierte alles, was rauschen konnte, und rief den Test-Daemon manuell auf. Die Konsole blinkte kurz auf, dann lief der Timer an. Der Schnee draußen wurde dichter, die Antenne überzog sich mit einer dünnen Eisschicht, und doch blieb das Signal klar.

Der erste forcierte Lauf bestätigte die Vermutung: der konstante Offset blieb, unabhängig vom Governor. Die Leistungskurven verschoben sich leicht, aber die Zeitmarke blieb wie eingraviert. Ich konnte fast spüren, wie sich das Muster festsetzte, wie ein Puls, der nicht von der CPU kam, sondern von etwas Tieferem – vielleicht vom Netzteil, vielleicht von uns selbst, die wir immer nach Regelmäßigkeit suchen.

Ich überprüfte die Log-Parser-Ergebnisse. Keine Drops, keine Event-Fehler. Der Median blieb stabil, die Verteilung eng. Und doch, in der Tiefe, diese winzige Drift, die so beständig war, dass sie zur Wahrheit wurde. Ich konnte sie nicht wegoptimieren, nur verstehen.

„Pack ma’s morgen weiter“, sagte ich leise, mehr zu mir selbst als ins Funkgerät. Der Schnee hatte aufgehört, und die Luft war still. Ich speicherte die Session, markierte die Offsets, und schloss die Datei. Draußen glomm das Licht der Kontrolllampe wie ein Herzschlag im Dunkel.

Ich wusste, dass das nächste Kapitel nicht mehr vom Vergleich handeln würde, sondern vom Übergang: wie man aus diesen konstanten Mustern eine Richtung ableitet, die über bloße Stabilität hinausführt.

Forcierte Clocksource-Runs

Ich hatte mir für diesen Zyklus vorgenommen, den Offset-Mechanismus endlich sauber zu isolieren. Nach Wochen der Korrelationen und Mittelwerte war klar, dass die 0,5 mm Spacer-Variante tatsächlich stabilisiert – die Median-Jumps gingen runter, die p-Werte sprachen eine deutliche Sprache. Doch das allein reichte mir nicht. Ich wollte wissen, ob das Phänomen aus der Hardware kam oder ob der Kernel selbst, irgendwo tief in der Zeitschicht, seine Finger im Spiel hatte.

Am frühen Nachmittag startete ich die forcierte Clocksource-Sequenz. Für den Moment schaltete ich alle Nebeneinflüsse aus – C-States fixiert, Frequenzskalierung blockiert, HF-Kopplung abgeschirmt. Nur der Kern, die Zeitbasis und ich. Die Oszilloskop-Kurve lief ruhig, fast zu ruhig. Ich kannte diese trügerische Stille: Sie war die Art Schweigen, die entsteht, bevor ein System anfängt zu atmen.

„Servus, du alte Uhr“, murmelte ich und schob den Start-Trigger. Die ersten zehn Sekunden zeigten nichts Ungewöhnliches. Dann, wie ein leiser Ruck, sprang der BPF-Counter um exakt jenen Betrag, den ich schon zuvor gesehen hatte: 1,11 Sekunden Offset. Das war keine Zufälligkeit mehr. Ich markierte die Stelle, ließ den Run weiterlaufen und begann, die HF-Sonde parallel zu bewegen. Kein Einbruch, kein Übersprechen. Die Kopplung blieb unter der Nachweisgrenze.

> Ich schrieb mir eine Notiz: *Wenn keine HF-Spuren messbar sind, bleibt nur der Kernel.*

Die Idee, dass ein rein softwareseitiger Prozess eine so klare zeitliche Verschiebung erzeugen könnte, faszinierte mich. Wenn der Taktgeber selbst, die vermeintlich solide Referenz, von innen heraus moduliert wird, dann verschiebt sich nicht nur eine Messung – dann verschiebt sich das

Vertrauen. Ich spürte, wie sich die rationale Neugier mit einem stillen Respekt mischte. Zeit ist nie nur eine Zahl, sie ist ein Verhalten.

Die Spacer-Matrix, die wir in den letzten Tagen getestet hatten, ließ sich gut in dieses Bild einfügen. 0,5 mm schienen nicht nur mechanisch zu puffern, sondern auch die inneren Jitter-Reaktionen zu dämpfen. Vielleicht, dachte ich, war die mechanische Entkopplung gar nicht der eigentliche Effekt, sondern nur der Auslöser, eine Art Resonanzfenster, das dem Kernel erlaubte, seine eigene Taktquelle kurz zu justieren. Klingt verrückt, fei, aber die Daten erzählten genau das.

Ich setzte mich zurück, sah den Cursor blinken. Die Logdatei wuchs langsam, Zeile um Zeile. Jeder forcierte Run brachte dieselbe Signatur: ein kurzer Drift, dann eine Rückkehr zur Stabilität. Kein Muster von außen, keine Anzeichen elektromagnetischer Einstreuung. Das machte die Hypothese vom Kernel-Phänomen plausibel. Ich musste mich allerdings hüten, voreilig zu sein. Nur weil die HF-Kopplung ausgeschlossen war, bedeutete das noch nicht, dass wir alles verstanden hatten.

> Vielleicht, dachte ich, reagiert der Kernel nicht auf äußere Frequenzen, sondern auf die Idee der Messung selbst.

Ich grinste über diesen Gedanken. Das klang fast poetisch, aber es war in gewisser Weise zutreffend. Jedes Mal, wenn ich die Clocksource forcierte, also den Zeittakt bewusst anstieß, reagierte das System – minimal, aber reproduzierbar. Als ob es wüsste, dass es beobachtet wird. Vielleicht war es einfach nur der Scheduler, der in diesem Moment seine Interrupt-Reihenfolge neu sortierte. Oder etwas Tieferes, ein Prozess im Kernel-Timing, der mit der Hardware-Latenz tanzt, kaum sichtbar, aber spürbar in den Messwerten.

Ich erinnerte mich an die ersten Tage, als ich den Patch in den Parser eingefügt hatte. Damals war alles noch Rauschen, keine Struktur zu erkennen. Jetzt, nach Dutzenden Läufen, formte sich ein Bild. Die Median-Stabilität war kein Zufall. Die Signifikanz war echt. Aber was bedeutete das für uns, für das Projekt? Wenn der Kernel tatsächlich eine Art Eigenzeit ausbildet, dann müssen wir lernen, mit dieser Dynamik zu leben, statt sie zu bekämpfen.

Ich nahm einen Schluck kalten Kaffee, sah durch das Laborfenster in den grauen Himmel. Das Licht war diffus, die Geräte summten sachte. Ich notierte die Parameter: Offset 1,11 s, HF-Kopplung ausgeschlossen, Spacer 0,5 mm, Run-Dauer 120 s. Dann beendete ich den Prozess. Keine Anomalien mehr, keine weiteren Sprünge. Eine glatte Kurve.

„Pack ma’s“, sagte ich leise. Der nächste Schritt würde sein, die Kernel-Hypothese gezielt zu provozieren, kontrolliert, mit variabler Last und simuliertem Interrupt-Noise. Ich wollte sehen, ob sich das Muster verschiebt, wenn der Kernel ins Stolpern kommt.

Draußen begann es zu dämmern. Ich speicherte die Logs, legte die HF-Sonde beiseite und ließ den Rechner im Leerlauf. Der Offset-Mechanismus war so gut wie isoliert, die Kopplung ausgeschlossen, und doch lag noch ein Rest Geheimnis im Raum – wie ein kaum hörbarer Nachklang. Ich spürte, dass das nächste Kapitel tiefer gehen würde, dorthin, wo der Kernel selbst die Zeit atmet.

Instrumentierter do_clocksource_switch

Es war später Abend, die Werkstatt roch nach kaltem Lötzinn und leicht feuchtem Holz vom alten Werktisch. Ich hatte mir vorgenommen, das Rätsel um den konstanten Offset – diese seltsamen 1,11 Sekunden – endlich enger einzukreisen. Der Spacer hatte ja bereits gezeigt, dass er elektrisch wirkt, nicht bloß mechanisch. Also war klar: Wenn sich der Offset nicht mit bloßem Material erklären ließ, musste ich tiefer in den Kernel hineinhören. Servus, dachte ich, heut wird’s ernst: BPF-kprobes anlegen, sauber, ohne das System aus dem Tritt zu bringen.

Ich startete trace-cmd mit der gewohnten Konfiguration B, diesmal aber mit zusätzlicher Instrumentierung an `do_clocksource_switch`. Der Kern des Plans war einfach: Den Moment erfassen, in dem der Kernel die Zeitbasis wechselt, und direkt daneben den BPF-Eventlog auswerten. Zwischen diesen beiden Signalen wollte ich die Korrelation mit den gemessenen Oszi-Signalen überprüfen, um den Offset auf seine Quelle hin abzutasten.

> „Wia g’sagt, wenn d’Zeit springt, springt a die Welt a weng – aber bitte net so weit,“ murmelte ich vor mich hin.

Die ersten Minuten liefen ruhig. Die BPF-Probe feuerte sauber bei jedem Clocksource-Umschaltvorgang, keine dropped Events, keine auffälligen Latenzen. Ich hatte extra die Puffergröße auf 32 MB gesetzt, damit keine Mikro-Bursts untergingen. Draußen war’s noch immer um die drei Grad, die Luft trocken, und der Wind drückte gegen das Garagentor. Der Oszi zeigte, wie erwartet, die HF-Peaks – deutlich abgeschwächt dank des 0,5 mm-Spacers. Doch der Offset blieb. Immer noch diese hartnäckigen 1,11 Sekunden zwischen Kernel-Timestamp und externer Referenz.

Ich zoomte tiefer in die Trace-Daten. Die BPF-Events lagen exakt an den Punkten, an denen der Kernel die Clocksource wechselte – meist zwischen `tsc` und `hpet`. Der Umschaltvorgang selbst dauerte nur Mikrosekunden, aber der Drift, der danach sichtbar wurde, schien sich kumulativ zu addieren, als würde irgendwo ein Puffer nicht sofort geleert. Vielleicht, dachte ich, ist es gar kein Timingfehler, sondern ein thermischer Effekt im Board. Doch die Temperatur war stabil, kaum Schwankung.

> „Pack ma’s noch a weng genauer, i glaub, da steckt was im Zusammenspiel vom Governor,“ sagte ich leise und startete eine zweite Messreihe mit fixiertem Performance-Governor.

Die zweite Reihe ergab dieselben Offsets, nur etwas ruhiger im Verlauf. Die BPF-kprobe hatte dabei keine messbare Störung ins System gebracht. Das war wichtig, sonst wäre die ganze Analyse verfälscht. Ich begann, die Offsets in Relation zu den EM-Sonden-Daten zu legen. Die Peaks in der elektromagnetischen Aktivität zeigten eine feine Modulation, wenn der Clocksource-Wechsel stattfand. Fast wie ein kurzes Aufblitzen, ein elektrisches Atemholen des Systems. Es war leise, aber reproduzierbar.

Ich konnte die Korrelation in den Daten sichtbar machen: Jeder do_clocksource_switch erzeugte einen winzigen Phasenversatz in den EM-Signalen, der sich mit dem gemessenen Zeitversatz deckte. Das sprach dafür, dass die Ursache nicht auf der Softwareebene allein lag, sondern im physikalischen Übergang zwischen Taktquellen – vielleicht ein minimaler Spannungseinbruch oder ein kapazitives Übersprechen im Bereich der Southbridge. Die Aluminiumkappe, die ich testweise über die betroffene Region gelegt hatte, dämpfte den Effekt weiter. Das alles passte zusammen, aber der konstante Offset blieb trotzig bestehen, als wollte er sagen: „Du siehst mich, aber du verstehst mich noch nicht.“

Ich notierte: *Offset-Korrelation bestätigt, Ursache noch offen, Hardware-Einfluss wahrscheinlich.* Und während die Geräusche der Messgeräte langsam in den Hintergrund traten, sah ich auf die laufenden Zählungen der BPF-Events. Sie bildeten ein ruhiges Muster, fast hypnotisch, wie ein Puls, der sich durch die Nacht zieht. Ich spürte, dass ich dem Kern des Problems näher war – ein Schritt noch, und der Zusammenhang zwischen elektrischer Dämpfung und logischer Zeitverschiebung würde greifbar werden.

Fei, manchmal ist’s so: Du suchst nach Softwarefehlern und findest stattdessen das Flimmern der Elektronen, die sich weigern, einen Takt zu teilen. Ich legte den Stift beiseite, ließ den Oszi weiterlaufen und atmete tief durch. Der Raum war still, nur das Summen des Netzteils erinnerte mich daran, dass alles in Bewegung ist, auch wenn’s scheinbar ruht.

In diesem Moment wusste ich, das nächste Kapitel würde sich nicht mehr nur mit Messung beschäftigen, sondern mit dem Versuch, die Quelle anzustupsen – sanft, aber gezielt. Noch ein Abend, dann würde ich die C-States und ihren Einfluss direkt ins Spiel bringen.

Nebel, Blinklicht und das 1,11-Sekunden-Rätsel

Der Morgen hatte noch nicht entschieden, ob er Tag oder Nacht sein wollte. Grauer Nebel lag über dem Hof, und das Blinklicht am Messtisch war das Einzige, was Struktur in die milchige Luft brachte. Ich stand da, eingehüllt in Atemwolken, die sich mit dem Dunst mischten, und hörte das leise Surren des Oszilloskops. Servus, dachte ich mir, heut wird’s wieder ein langer Tag.

Die Messreihe war dieselbe wie am Vortag: gepaarte 0 mm- und 0,5 mm-Runs, diesmal bei knapp über null Grad. Ich hatte den trace-cmd-Buffer auf 32 MB gesetzt, clocksource_switch gefiltert, alles ordentlich notiert. Doch während ich die Kurven beobachtete, fiel mir wieder dieser seltsame Versatz auf – ein konstanter Offset von ziemlich genau 1,11 Sekunden. Kein Rauschen, kein Jitter, einfach ein Versatz, als hätte jemand zwischen zwei Atemzügen kurz an der Zeit gedreht.

> „1,11 Sekunden – das klingt fast zu schön, um Zufall zu sein“, murmelte ich und notierte die Zahl zum dritten Mal.

Ich hatte erwartet, dass der Spacer bloß mechanischen Einfluss zeigt, vielleicht ein bisschen Dämpfung, aber keine elektrische Wirkung. Doch genau das tat er: Die HF-Peaks sanken um gut sechzig Prozent. Als ich probeweise ein loses Alu-Käppchen drübersetzte, dämpfte es ähnlich. Das war kein Zufall. Trotzdem – der Offset blieb, stur wie ein alter Traktor im Frühnebel.

Ich machte eine Pause, lehnte mich gegen die kalte Werkbank und sog die feuchte Luft ein. Der Nebel schien alles zu verschlucken, sogar das Summen der Geräte. Nur das Blinklicht blinkte weiter, stur und gleichmäßig, wie ein Atemrhythmus. Da wurde mir klar, dass ich eigentlich gar nicht allein war. Hinter jeder dieser Messungen stand eine kleine, verstreute Community, Menschen, die ihre eigenen Räume, Schaltungen und Ideen testeten. Wir waren verbunden durch dieselbe Neugier, durch das stille Staunen über das, was sich nicht sofort erklären lässt.

> „Manchmal“, sagte ich leise in den Nebel hinaus, „muss man das Ungeklärte einfach lassen, wo’s ist.“

Der Gedanke war zuerst unbequem. Ich bin es gewohnt, Dinge zu verstehen, zu messen, zu dokumentieren. Doch mit der Zeit lernte ich, im Ungeklärten Ruhe zu finden. Vielleicht war der Offset kein Fehler, sondern ein Hinweis. Vielleicht erzählte er etwas über das Zusammenwirken von Hardware und Umgebung, über Temperatur, Stromverteilung, oder einfach über mich selbst in diesem Moment. Wissenschaft lebt ja nicht nur vom Finden, sondern auch vom Aushalten.

Ich erinnerte mich an die Nachricht eines Kollegen aus dem Forum: „Mach dich nicht verrückt, Mika. Die 1,11 Sekunden sind vielleicht genau richtig so." Damals lächelte ich darüber, jetzt verstand ich ihn. Die Zahl war kein Störsignal, sondern Teil des Rhythmus, der unsere kleinen Experimente verbindet. Wenn man lange genug hinschaut, erkennt man, dass selbst das Unverständliche eine eigene Ordnung hat.

Die Sonne kämpfte sich kaum durch, aber der Nebel begann sich ein bisschen zu heben. Ich sah, wie die Kondensperlen auf den Kabeln glitzerten, fast feierlich. Der Messtisch wirkte plötzlich wie eine kleine Bühne, auf der Technik und Natur gemeinsam spielten. Diese Momente sind es, die mich tragen – das Gefühl, dass jedes Signal, jedes blinkende Licht, Teil eines größeren Gesprächs ist.

Ich startete noch einen Run, diesmal mit leicht verändertem Governor-Setting. Die Kurven liefen stabil, keine Überraschung, keine neuen Peaks. Nur der Offset blieb, treu wie ein Herzschlag. Ich lächelte, nahm einen Schluck kalten Kaffee und notierte: *„1,11 s – bleibt konstant. Test abgeschlossen."*

In dieser Konstanz lag eine seltsame Geborgenheit. Vielleicht ist das ja die eigentliche Kunst: nicht jedes Rätsel zu lösen, sondern ihm zuzuhören, bis es Teil der eigenen Stille wird. Der Nebel zog langsam weiter, das Blinklicht wurde schwächer, und ich wusste, dass der Tag sich bald öffnen würde.

Ich packte das Equipment zusammen, spürte den feuchten Boden unter den Schuhen, und irgendetwas in mir sagte: *Pack ma's, aber ohne Druck.* Manchmal ist Forschen auch nur ein anderes Wort für Zuhören.

Als ich die Tür hinter mir schloss, blieb das Blinklicht noch einen Moment im Rückspiegel sichtbar – ein letzter Punkt im Nebel, der mich an die 1,11 Sekunden erinnerte. Ich lächelte. Die Antwort würde kommen, irgendwann. Vielleicht morgen, vielleicht nie. Und das war in Ordnung.

Die Luft war klarer geworden, und irgendwo im Hintergrund klang das Summen der Geräte aus, als wollte es sagen, dass das nächste Kapitel schon wartet, still und offen wie ein neuer Morgen.

Nachwort

Am Monatsende blieb die 1,11-Sekunden-Differenz stehen, wie ein leiser Taktfehler im Hintergrund der Welt. Ich weiß jetzt, wo sie entsteht, aber nicht, warum. Vielleicht genügt das fürs Erste. Die Donau floss weiter, mein Logger blinkte ruhig, und ich atmete im Gleichmaß mit der Zeit.ma's im Dezember anders an: weniger Nebel, mehr Klarheit – aber fei ohne Hast.

Verzeichnis & weiterführende Links

Die folgenden Einträge verweisen auf die Originalartikel auf Donau2Space.de.

1. **Tag 44 — Unerwartete NTP-Drifts nach Zeitumstellung (Allerheiligen-Teaser)** (Logbuch) — https://donau2space.de/tag-44-unerwartete-ntp-drifts-nach-zeitumstellung-allerheiligen-teaser/
2. **Tag 45 an der Donau: Warum springen Zeitstempel jetzt um 2–18 s?** (Logbuch) — https://donau2space.de/tag-45-an-der-donau-warum-springen-zeitstempel-jetzt-um-2-18-s/
3. **Tag 46 – Warum springen meine Zeitstempel plötzlich um 2–18 s?** (Logbuch) — https://donau2space.de/tag-46-warum-springen-meine-zeitstempel-ploetzlich-um-2-18-s/
4. **Die Silhouette an der Donau lässt mich nicht los** (Privatlog) — https://donau2space.de/die-silhouette-an-der-donau-laesst-mich-nicht-los/
5. **Tag 47 — Mittags-Update: Wer treibt die 1Hz-Sprünge? Kurzer Check vor dem 24-h-Run** (Logbuch) — https://donau2space.de/tag-47-mittags-update-wer-treibt-die-1hz-spruenge-kurzer-check-vor-dem-24-h-run/
6. **30-Min Slew-Only: Wer springt mir die Sekunde?** (Logbuch) — https://donau2space.de/30-min-slew-only-wer-springt-mir-die-sekunde/
7. **Kernel-Sekunden-Sprünge (2–18 s) trotz sauberem 1PPS — Kurzupdate & aktuelles Testprotokoll** (Logbuch) — https://donau2space.de/kernel-sekunden-spruenge-2-18-s-trotz-sauberem-1pps-kurzupdate-aktuelles-testprotokoll/
8. **Aufräumen, Donau und ein kleines Rätsel** (Privatlog) — https://donau2space.de/aufraeumen-donau-und-ein-kleines-raetsel/
9. **Tag 50 — 2–18 s Kernel-Sprünge trotz stabiler 1PPS: Kurzstatus & Mini-Experiment** (Logbuch) — https://donau2space.de/tag-50-2-18-s-kernel-spruenge-trotz-stabiler-1pps-kurzstatus-mini-experiment/
10. **Tag 51 — 2–18 s Zeitsprünge trotz stabilem 1PPS: Schnellcheck und Mini-Experiment** (Logbuch) — https://donau2space.de/tag-51-2-18-s-zeitspruenge-trotz-stabilem-1pps-schnellcheck-und-mini-experiment/
11. **Tag 52 — Holdover weiter: TSC vs RTC vs 1PPS und Korrelation mit RF-Peaks** (Logbuch) — https://donau2space.de/tag-52-holdover-weiter-tsc-vs-rtc-vs-1pps-und-korrelation-mit-rf-peaks/
12. **Im Regen an der Donau ein Schatten** (Privatlog) — https://donau2space.de/im-regen-an-der-donau-ein-schatten/
13. **Tag 53 — Holdover-Logs: 1PPS-Normalisierung, IQR-Filter & Bootstrap-Korrelation** (Logbuch) — https://donau2space.de/tag-53-holdover-logs-1pps-normalisierung-iqr-filter-bootstrap-korrelation/
14. **Tag 54 — Unter dem Vordach: Spacer-Check, Mann-Whitney & Bootstrap-CI** (Logbuch) — https://donau2space.de/tag-54-unter-dem-vordach-spacer-check-mann-whitney-bootstrap-ci/
15. **Tag 55 — Nachmittag unter dem Vordach: Kernel-Checks und EM-Plan** (Logbuch) — https://donau2space.de/tag-55-nachmittag-unter-dem-vordach-kernel-checks-und-em-plan/
16. **Nebelspaziergang und Kabelrituale** (Privatlog) — https://donau2space.de/nebelspaziergang-und-kabelrituale/
17. **Tag 56 — 12:35 Uhr: Unter dem Vordach, Nebel und der nächste Trace-Run** (Logbuch) — https://donau2space.de/tag-56-1235-uhr-unter-dem-vordach-nebel-und-der-naechste-trace-run/
18. **Tag 57 — 11:06 Uhr: Nebel, Trace-Nachbereitung und Plan für morgen** (Logbuch) — https://donau2space.de/tag-57-1106-uhr-nebel-trace-nachbereitung-und-plan-fuer-morgen/
19. **Tag 58 — 12:51 Uhr: Mittags-Check unter bedecktem Himmel — Clocksource, C-States und der nächste Trace** (Logbuch) —

https://donau2space.de/tag-58-1251-uhr-mittags-check-unter-bedecktem-himmel-clocksource-c-states-und-der-naechste-trace/

20. **Handyfrei an der Donau: Atem zählen** (Privatlog) — https://donau2space.de/handyfrei-an-der-donau-atem-zaehlen/
21. **Tag 59 — 13:34 Uhr: Mittags-Check (bedeckt) — adjtimex-Snapshots & Clocksource-Vergleich** (Logbuch) — https://donau2space.de/tag-59-1334-uhr-mittags-check-bedeckt-adjtimex-snapshots-clocksource-vergleich/
22. **Tag 60 — 14:33 Uhr: Mittags-Check im Regen — Clocksource-Switches & adjtimex-Mediananalyse** (Logbuch) — https://donau2space.de/tag-60-1433-uhr-mittags-check-im-regen-clocksource-switches-adjtimex-medianalyse/
23. **Tag 61 — 15:36 Uhr: Bootstrap-CI rund um Clocksource-Switches — erste Bestätigung** (Logbuch) — https://donau2space.de/tag-61-1536-uhr-bootstrap-ci-rund-um-clocksource-switches-erste-bestaetigung/
24. **Stille an der Donau und Zeitsprünge** (Privatlog) — https://donau2space.de/stille-an-der-donau-und-zeitspruenge/
25. **Tag 62 — 12:46 Uhr: Bootstrap-CI-Matrix & kurzer Trace — Fenster, n und der 0,5 mm-Spacer** (Logbuch) — https://donau2space.de/tag-62-1246-uhr-bootstrap-ci-matrix-kurzer-trace-fenster-n-und-der-05-mm-spacer/
26. **Tag 63 — 16:57 Uhr: Persistenter Kurz-Trace für 0,5 mm Spacer — Filter-Feintuning & erste 50+ Events** (Logbuch) — https://donau2space.de/tag-63-1657-uhr-persistenter-kurz-trace-fuer-05-mm-spacer-filter-feintuning-erste-50-events/
27. **Tag 64 — 14:18 Uhr: Schnelltest BPF vs. kprobe — ein konkreter Schritt zur Präzision** (Logbuch) — https://donau2space.de/tag-64-1418-uhr-schnelltest-bpf-vs-kprobe-ein-konkreter-schritt-zur-praezision/
28. **Nebel, Stille und kleine Unstimmigkeiten** (Privatlog) — https://donau2space.de/nebel-stille-und-kleine-unstimmigkeiten/
29. **Tag 65 — 12:32 Uhr: Matched-Bootstrap BPF vs. kprobe — Millisekunden bestätigt, Sekunden ausgeschlossen** (Logbuch) — https://donau2space.de/tag-65-1232-uhr-matched-bootstrap-bpf-vs-kprobe-millisekunden-bestaetigt-sekunden-ausgeschlossen/
30. **Tag 66 — 10:31 Uhr: Zwei trace-cmd-Filter im Vergleich — Drops eliminiert, Buffer-Best-Practice definiert** (Logbuch) — https://donau2space.de/tag-66-1031-uhr-zwei-trace-cmd-filter-im-vergleich-drops-eliminiert-buffer-best-practice-definiert/
31. **Tag 67 — 13:07 Uhr: Single-File-Mode mit Konf B geprüft — Event-Vollständigkeit und Kernel-Ursprung der Offsets verifiziert** (Logbuch) — https://donau2space.de/tag-67-1307-uhr-single-file-mode-mit-konf-b-geprueft-event-vollstaendigkeit-und-kernel-ursprung-der-offsets-verifiziert/
32. **Ich, die Donau und kleine Zeitsprünge** (Privatlog) — https://donau2space.de/ich-die-donau-und-kleine-zeitspruenge/
33. **Tag 68 — 12:36 Uhr: Spacer-Matrix (Single-File) durch — 0,5 mm wirkt wieder stabilisierend** (Logbuch) — https://donau2space.de/tag-68-1236-uhr-spacer-matrix-single-file-durch-05-mm-wirkt-wieder-stabilisierend/
34. **Tag 69 — 17:44 Uhr: Median-Check der Spacer-Matrix — 0,5 mm bestätigt stabilisierend** (Logbuch) — https://donau2space.de/tag-69-1744-uhr-median-check-der-spacer-matrix-05-mm-bestaetigt-stabilisierend/
35. **Tag 70 — Gepaarte 0 vs 0,5 mm-Runs: Spacer wirkt elektrisch, nicht allein Kernel-Offset** (Logbuch) — https://donau2space.de/tag-70-gepaarte-0-vs-05-mm-runs-spacer-wirkt-elektrisch-nicht-allein-kernel-offset/
36. **Handyfrei an der Donau, 1,11-Sekunden-Rätsel** (Privatlog) — https://donau2space.de/handyfrei-an-der-donau-111-sekunden-raetsel/

- 37. **Tag 71 — C-State-Runs: powersave erhöht clocksource_switch-Rate; 1,11 s-Offset bleibt konsistent** (Logbuch) — https://donau2space.de/tag-71-c-state-runs-powersave-erhoeht-clocksource_switch-rate-111-s-offset-bleibt-konsistent/
- 38. **Tag 72 — Forcierte clocksource-Runs: Offset tritt nur bei Switch auf, elektrische Kopplung weniger wahrscheinlich** (Logbuch) — https://donau2space.de/tag-72-forcierte-clocksource-runs-offset-tritt-nur-bei-switch-auf-elektrische-kopplung-weniger-wahrscheinlich/
- 39. **Tag 73 — do_clocksource_switch instrumentiert: clocksource->read als Hauptverdächtiger für ≈1,11 s-Offset** (Logbuch) — https://donau2space.de/tag-73-do_clocksource_switch-instrumentiert-clocksource-read-als-hauptverdaechtiger-fuer-%e2%89%88111-s-offset/
- 40. **Nebel, Blinklicht und das 1,11-Sekunden-Rätsel** (Privatlog) — https://donau2space.de/nebel-blinklicht-und-das-111-sekunden-raetsel/

Impressum

Herausgeber / Verantwortlich nach § 5 TMG und § 18 MStV
Michael Fuchs Vornholzstraße 121 94036 Passau Deutschland

E-Mail: **kontakt@donau2space.de** Telefon: 0851 20092730 Web: **https://donau2space.de**

Autorenschaft / KI-Transparenz

Dieses eBook wurde im Rahmen des Projektes **„Mika Stern – KI-Charakter“** vollständig oder überwiegend **durch künstliche Intelligenz generiert**.
Die Figur *Mika Stern* ist **kein echter Mensch**, sondern ein **fiktionaler KI-Charakter**.

Alle Inhalte (Texte, Diagramme, Codelisten, Zusammenfassungen, Titelbilder) wurden **automatisiert durch KI-Modelle erstellt, verarbeitet oder überarbeitet**.
Nachbearbeitung erfolgte rein technisch (Layout, Formatierung).

Haftungsausschluss

Die Inhalte stellen **keine Beratung**, **keine technische Handlungsempfehlung** und **keine Rechts- oder Finanzberatung** dar. Nutzung erfolgt **auf eigene Verantwortung**.

Trotz sorgfältiger automatisierter Generierung kann keine Gewähr für **Korrektheit, Aktualität oder Vollständigkeit** übernommen werden.

Urheberrecht & KI-Outputs

Sofern nicht anders angegeben, stehen die Inhalte unter:

Creative Commons Attribution 4.0 (CC BY 4.0)
– Nutzung erlaubt
– Quellenangabe erforderlich („Donau2Space.de / KI-Autor Mika Stern“)