UKZ: Die fehlende fundamentale Zeitbasis der Informatik

Eine algorithmische Prämisse unterhalb der Relativitätstheorie

Dirk Leopold Feiler

Unabhängiger Forscher · Kaiserslautern, Deutschland ukz-tech.online · contact@ukz-tech.online

Eingereicht: 22. Juni 2026 | Fachgebiete: Computational Sciences, Distributed Systems, Real-Time Computing

Abstract. Aktuelle Zeitstandards — UTC, TAI und GPS-Zeit — weisen fundamentale architektonische Schwächen auf, die in sicherheitskritischen, echtzeitkritischen und deterministischen Rechensystemen nicht länger akzeptabel sind. UTC opfert die strikte Monotonie durch die Einfügung von Schaltsekunden. TAI und GPS-Zeit führen Floating-Point-Akkumulationsfehler ein und erfordern komplexe Korrekturpipelines. Am kritischsten jedoch: Alle relativistischen Korrekturformeln (Zeitdilatation, Gravitationsrotverschiebung) setzen voraus, was sie selbst nicht liefern können: eine saubere, koordinierte und strikt monotone Zeitreferenz. Dieses Papier schlägt UKZ (Universelle Koordinierte Zeit) vor — eine rein algorithmische und architektonische Layer-0-Zeitprimitive, definiert durch den geschlossenen ganzzahligen Ausdruck $T_{UKZ}(t) = \lfloor (t - t_{real}(0)) / \Delta t \rfloor$. UKZ ist keine physikalische Theorie; sie ist eine strikt monotone, rein ganzzahlige Zeitreferenz, die vollständig unabhängig von Lorentz-Transformationen, Lichtgeschwindigkeit und metrischer Raumzeit operiert. Relativistische Korrekturen werden auf UKZ in Layer 2 aufgesetzt, nicht umgekehrt. Die vorgeschlagene Architektur steht daher logisch unterhalb der Relativitätstheorie und stellt die Bedingung der Möglichkeit für präzise physikalische Zeitmessung dar. Praktische Implikationen umfassen Sensorfusion in autonomen Fahrzeugen, deterministische verteilte Systeme und Compliance im Hochfrequenzhandel (MiFID II RTS 6).

Keywords: Systemzeit, monotone Uhr, ganzzahlige Zeitbasis, UTC, TAI, deterministische Systeme, verteilte Echtzeit, autonome Systeme, Schaltsekunden, UKZ, Zeitarchitektur, MiFID II

1. Einleitung

Zeit ist eines der fundamentalsten Primitive in der Informatik, dennoch bleibt ihre grundlegende Behandlung in der Systemarchitektur erstaunlich unvollständig. Moderne Systeme erben Zeitrepräsentationen, die für menschliche Kalenderzwecke oder für die pragmatische Abstimmung heterogener physikalischer Oszillatoren entworfen wurden. Sie wurden nicht mit den semantischen Anforderungen deterministischer, sicherheitskritischer oder hochfrequenter Rechensysteme im Sinn entwickelt.

Drei dominante Standards beherrschen die globale Zeitmessung in der Informatik:

Coordinated Universal Time (UTC), International Atomic Time (TAI) und GPS-Zeit. Jeder trägt strukturelle Schwächen, die sich direkt ins Systemverhalten fortpflanzen. Die periodische Einfügung von Schaltsekunden in UTC — mittlerweile 27 seit 1972 — verletzt die strikte Monotonie und macht UTC grundsätzlich ungeeignet als Systemuhr-Primitive. TAI und GPS-Zeit, obwohl prinzipiell monoton, werden in Software oft als Floating-Point-Werte dargestellt, die Akkumulationsfehler einführen und nicht-triviale

Korrekturmechanismen erfordern. Relativistische Anpassungen in Satellitenhöhe (~38 µs/Tag für GPS) setzen eine stabile Basisuhr voraus, die keiner der drei Standards auf Layer 0 sauber bereitstellen kann.

Dieses Papier führt UKZ (Universelle Koordinierte Zeit) ein: eine minimale architektonische Prämisse — ein strikt monotoner, rein ganzzahliger Zeitzähler, definiert durch einen einzigen geschlossenen Ausdruck. UKZ erhebt keine physikalischen Ansprüche. Sie ersetzt nicht die relativistische Physik. Sie etabliert die Schicht unterhalb der Physik — das computationale Substrat, auf dem alle höherwertige zeitliche Schlussfolgerung, einschließlich relativistischer Korrektur, sicher konstruiert werden kann.

2. Defizite aktueller Zeitstandards

2.1 UTC und das Schaltsekundenproblem

UTC ist ein Hybridstandard: Sie versucht gleichzeitig Atomzeit (via TAI) und die Rotationsperiode der Erde zu verfolgen. Dies wird durch die unregelmäßige
Einfügung von Schaltsekunden erreicht, entschieden vom International Earth
Rotation and Reference Systems Service (IERS). Die Konsequenzen für die Informatik sind schwerwiegend: UTC ist keine monotone Funktion.
Softwaresysteme, die ein Schaltsekundenereignis beobachten, erleben denselben Zeitstempel zweimal oder — je nach Implementierung — einen Rücksprung. Verteilte Systeme, die über diese Grenze hinweg synchronisieren, zeigen undefiniertes Verhalten in jedem Protokoll, das Wanduhr-Monotonie voraussetzt. Hochprofilierte Vorfälle bei Cloudflare, Reddit und dem Linux-Kernel-Scheduler sind direkt auf Fehler bei der Schaltsekundenbehandlung zurückzuführen.

Bemerkenswert ist, dass die ITU-R seit 2012 über die Abschaffung der Schaltsekunden debattiert. Die BIPM/ITU-Radiocommunication Assembly stimmte im November 2022 zu, Schaltsekundeneinfügungen bis 2035 einzustellen — ein Eingeständnis, dass der aktuelle Mechanismus architektonisch untragbar ist. UKZ antizipiert diesen Schluss und verallgemeinert ihn.

2.2 TAI und Floating-Point-Akkumulation

TAI vermeidet das Schaltsekundenproblem, indem sie SI-Sekunden kontinuierlich seit dem 1. Januar 1958 zählt. TAI selbst ist grundsätzlich ein ganzzahliger Zähler von SI-Sekunden; jedoch werden TAI-abgeleitete Zeitstempel in vielen Software-Implementierungen als IEEE-754-Double-Precision-Floating-Point-Werte dargestellt, um Sub-Sekunden-Präzision zu ermöglichen. Bei aktuellen TAI-Epochenwerten (~$2{,}0 \times 10^9$ Sekunden) beträgt die Repräsentationsauflösung eines 64-Bit-Doubles bei dieser Größenordnung etwa 240 Nanosekunden — unzureichend für Nanosekunden-Präzisionsanforderungen in moderner Sensorfusion und deterministischer Ablaufplanung.

2.3 Die zirkuläre Abhängigkeit in der relativistischenZeitmessung

Ein tieferes strukturelles Problem unterliegt allen drei Standards: Relativistische Korrekturformeln für Zeitdilatation und Gravitationsrotverschiebung sind bezüglich einer Referenzrahmen-Zeit definiert. Diese Referenzzeit muss selbst wohldefiniert, koordiniert und monoton sein, bevor die Korrekturen sinnvoll angewendet werden können. Mit anderen Worten: Die relativistische Physik setzt genau das voraus, was aktuelle Standards auf Layer 0 nicht sauber bereitstellen. UKZ löst diese zirkuläre Abhängigkeit, indem sie den ganzzahligen Zähler (Layer 0) sauber von allen darüberliegenden physikalischen Interpretationsschichten trennt.

3. UKZ: Formale Definition

Sei t_{real} die physikalisch verstrichene Zeit, gemessen durch einen vereinbarten Atomoszillator, t_0 der Epochenanker (ein fester Referenzzeitpunkt) und Δt die gewählte Tick-Auflösung (z.B. 1 ns, 1 µs oder 1 ms, je nach Einsatzkontext). Der UKZ-Zähler ist dann definiert als:

$$T_{UKZ}(t) = \lfloor (t_{real} - t_0) / \Delta t \rfloor$$

Dieser Ausdruck weist die folgenden formalen Eigenschaften auf:

Eigenschaft	**Formale Aussage**	**Konsequenz**
Strikte Monotonie	$t_1 < t_2 \Rightarrow T_{UKZ}(t_1) \leq T_{UKZ}(t_2)$	Delta-Berechnungen sind immer nicht-negativ
Ganzzahlig	$T_{UKZ}(t) \in \mathbb{Z}$ für alle $t \geq t_0$	Kein Floating-Point-Fehler; exakte Subtraktion
Begrenztheit	Bestimmt durch Bit-Breite b: $T_{max} = 2^b - 1$	64-Bit: Überlauf nach ~585 Jahren bei 1 ns Auflösung
Unabhängigkeit	Keine Abhängigkeit von geografischem Rahmen, Gravitationspotential oder Geschwindigkeit	Physisch invariantes Zählersubstrat
Komponierbarkeit	Physikalische Korrekturen angewendet als $f(T_{UKZ})$	Relativistische Schicht sauber getrennt

Tabelle 1: Formale Eigenschaften des UKZ-Zählers.

4. Geschichtete Zeitarchitektur

Ein zentraler architektonischer Beitrag von UKZ ist die explizite Stratifizierung zeitlicher Semantik über Schichten. Dies spiegelt etablierte Praxis in Netzwerkprotokollstacks und Betriebssystemdesign wider, wurde jedoch bisher nicht systematisch auf die Zeitmessung angewendet:

Layer	**Name**	**Inhalt**	**Abhängig von**
0	UKZ-Zähler	$T_{UKZ}(t)$ — ganzzahliger Tick-Zähler	Nur physikalischer Oszillator
1	Epochen-Abbildung	Konvertierung zu Kalender-/SI-Einheiten	Layer 0
2	Relativistische Korrektur	Lorentz-Faktor, Gravitationsblauverschiebung, angewendet als Delta auf Layer 0	Layer 0 + Layer 1

3	Anwendungszeit	Zeitstempel, Dauern, Ereignisordnung in Anwendungsprotokollen	Layer 0–2

Tabelle 2: Geschichtete UKZ-Zeitarchitektur.

Der entscheidende Einblick ist, dass Layer 2 (relativistische Korrekturen) auf der Ausgabe von Layer 0 als Transformation operiert, nicht als Ko-Definition. Dies löst die in Abschnitt 2.3 identifizierte zirkuläre Abhängigkeit. Einsteins allgemeine und spezielle relativistische Zeitkorrekturen bleiben vollständig gültig — sie werden einfach auf der korrekten architektonischen Schicht angewendet.

5. Einordnung bestehender Ansätze

5.1 Logische Uhren in verteilten Systemen

Lamports logische Uhren [1] und Vektoruhren [2] etablieren Ereignisordnung in verteilten Systemen, ohne auf physikalische Zeit zurückzugreifen. UKZ ist davon unterschieden: Sie ist nicht bloß ein logischer Ordnungsmechanismus, sondern ein physikalisch verankerter ganzzahliger Zähler mit einer definierten SI-Zeit-Epoche. Sie ist stärker eingeschränkt als logische Uhren (sie trägt physikalische Zeitinformation), aber architektonisch sauberer als UTC oder TAI (sie garantiert Monotonie und Ganzzahligkeit).

5.2 POSIX-Monotonuhr

POSIX definiert CLOCK_MONOTONIC als eine nicht-abnehmende Uhr, die nicht diskontinuierlichen Anpassungen unterliegt. Dies ist der nächste bestehende Gegenpart zu UKZ. Jedoch ist CLOCK_MONOTONIC implementierungsdefiniert, nicht über verteilte Knoten standardisiert, trägt keine formale Epoche und wird typischerweise als struct timespec (Sekunden + Nanosekunden) statt als reine Ganzzahl dargestellt. UKZ schlägt vor, diese Intuition als erstklassige architektonische Primitive mit geteilter Epoche und definierter Bit-Breite zu standardisieren und zu formalisieren.

5.3 Der Einwand der „Newtonschen absoluten Zeit"

Ein potenzieller Einwand gegen UKZ ist, dass sie eine Rückkehr zur Newtonschen absoluten Zeit darstellt — eine Vorstellung, die die Relativitätstheorie definitiv widerlegt hat. Dieser Einwand begeht einen Kategorienfehler. Newtons absolute Zeit war ein physikalischer Anspruch über die metaphysische Natur der Dauer. UKZ ist ein architektonischer Anspruch über die angemessene computationale
Repräsentation von Zeit auf Systemebene. UKZ behauptet nicht, dass Zeit in allen Referenzrahmen identisch fließt; sie behauptet, dass ein verteiltes Rechensystem einen geteilten ganzzahligen Zähler als Koordinationssubstrat benötigt, auf dem jede gewünschte rahmenabhängige Korrektur angewendet werden kann. Der Unterschied liegt zwischen Ontologie und Architektur.

6. Anwendungsdomänen

6.1 Sensorfusion in autonomen Systemen

Multisensorfusion in autonomen Fahrzeugen erfordert Zeitstempelung von Ereignissen aus LiDAR-, Kamera-, Radar-, IMU- und GNSS-Quellen mit Sub-Millisekunden-Präzision und garantierter Ordnung. Jede Schaltsekunde oder

Floating-Point-Ambiguität in der gemeinsamen Zeitbasis pflanzt sich direkt in Positions-Schätzfehler fort. UKZ stellt das erforderliche monotone ganzzahlige Substrat bereit; GNSS-abgeleitete absolute Position kann auf UKZ-Layer 1 abgebildet werden, ohne die Integrität der Ordnungsschicht zu beeinträchtigen.

6.2 Deterministische verteilte Echtzeitsysteme

Deterministische Ausführung in verteilten Systemen (z.B. zeitgetriggerte Architekturen nach AS6802 oder TT-Ethernet) erfordert, dass alle Knoten eine gemeinsame Vorstellung von „jetzt" teilen, die frei von Ambiguität ist. UKZ, synchronisiert über Knoten via PTP (IEEE 1588) oder Äquivalent, stellt einen geteilten ganzzahligen Zähler bereit, der schaltsekundeninduzierte Unsicherheit eliminiert.

6.3 Hochfrequenzhandel und MiFID II RTS 6

Die europäische Finanzregulierung unter MiFID II RTS 6 schreibt Mikrosekunden-genaue Zeitstempelgenauigkeit für Handelssysteme vor. Die Regulierung impliziert eine monotone, auditierbare Zeitreferenz. Die Nicht-Monotonie von UTC stellt ein Compliance-Risiko dar. UKZ stellt die architektonische Primitive bereit, die RTS-6-Anforderungen erfüllt und gleichzeitig eindeutige Post-Trade-Ereignisrekonstruktion ermöglicht.

7. Bit-Breiten-Varianten und Überlaufanalyse

Bit-Breite	Maximalwert	Auflösu ng	Überlauf (1 ns Tick)	Anwendungsfall
32-Bit	~$4{,}3 \times 10^{9}$	1 µs	~4295 Sekunden (~71 min)	Embedded / Kurzlebig
48-Bit	~$2{,}8 \times 10^{14}$	1 µs	~89 Jahre	IoT / Industrie
64-Bit	~$1{,}8 \times 10^{19}$	1 ns	~585 Jahre	Rechenzentrum / HFT / Autonom
128-Bit	> 10^{38}	1 ps	Kosmologisch	Physik / Forschung

Tabelle 3: UKZ-Bit-Breiten-Varianten mit Überlaufanalyse.

8. Synchronisationsstrategie

UKZ schreibt kein spezifisches Synchronisationsprotokoll vor. Sie definiert das semantische Ziel: Alle Knoten in einem verteilten System sollten Zähler halten, die um höchstens ε Ticks differieren, wobei ε anwendungsdefiniert ist. Bestehende Synchronisationsinfrastruktur kann direkt auf UKZ abgebildet werden:

PTP / IEEE 1588: Sub-Mikrosekunden-Synchronisation über Ethernet; bildet sauber auf UKZ-Tick-Ausrichtung ab, ohne Modifikation des
Synchronisationsprotokolls selbst.

NTP: Millisekunden-Bereich-Synchronisation; ausreichend für viele verteilte Anwendungen bei µs-Auflösung UKZ-Ticks.

GPS-disziplinierte Oszillatoren: Bieten Nanosekunden-ausgerichtete absolute Epochenverankerung für t
.

0

Entscheidend ist, dass keine dieser Synchronisationsmechanismen Schaltsekunden oder Nicht-Monotonie in den UKZ-Zähler selbst einführt. Diskontinuierliche Anpassungen, die für Kalenderausrichtung erforderlich sind, werden auf Layer 1 verbannt, wodurch der Layer-0-Zähler jederzeit strikt monoton bleibt.

9. Vergleichende Zusammenfassung

Eigenschaft	**UTC**	**TAI**	**GPS-Zeit**	**POSIX Monoton**	**UKZ**
Strikt monoton	Nein	Ja	Ja	Ja*	Ja
Ganzzahlig	Nein	Nein	Nein	Teilweise	Ja
Schaltsekundenfrei	Nein	Ja	Ja	Ja	Ja
Geteilte Epoche (verteilt)	Ja	Ja	Ja	Nein	Ja
Floating-Point-frei	Nein	Nein	Nein	Nein	Ja
Relativistische Schichttrennung	Nein	Nein	Nein	n.v.	Ja
Standardisierte Bit-Breite	n.v.	n.v.	n.v.	Nein	Ja
Δ = einfache Subtraktion	Nein	Nein	Nein	Teilweise	Ja

*Tabelle 4: Vergleichende Eigenschaften von Zeitstandards. * POSIX CLOCK_MONOTONIC ist pro Knoten monoton, aber nicht über verteilte Knoten standardisiert.*

10. Diskussion

UKZ ist bewusst ein minimaler Vorschlag. Sie versucht nicht, Kalenderzeit, relativistische Korrektur oder bestehende Synchronisationsinfrastruktur zu ersetzen. Sie argumentiert für eine saubere architektonische Trennung: Das computationale Substrat der Zeit (Layer 0) sollte eine Sache ganzzahligen Zählens sein, Punkt. Alle menschenlesbaren, physikalisch korrigierten oder rahmenabhängigen Repräsentationen sind Transformationen, die oberhalb dieser Schicht angewendet werden.

Diese Trennung ist prinzipiell nicht neu — sie spiegelt die Trennung zwischen Maschinencode und Quellsprache, zwischen physikalischer Schicht und Netzwerkprotokoll, zwischen Speicherrepräsentation und

Anwendungsdatenmodell wider. Was neu ist, ist die explizite Formalisierung dieser Trennung für die Zeitmessung auf Systemebene und der Vorschlag, sie als erstklassiges architektonisches Konzept zu standardisieren.

Die nächste bestehende Verkörperung dieser Idee ist die CLOCK_MONOTONIC_RAW des Linux-Kernels, die einen Hardware-Oszillator-Zähler bereitstellt, der von NTP-Anpassungen unberührt bleibt. UKZ schlägt vor, dieses Konzept zu einem formalen, verteilbaren, epochenverankerten Standard mit definierter Bit-Breite-Semantik zu erheben.

UKZ nimmt eine distinkte Position relativ zu logischen Uhren in verteilten
Systemen ein: Im Gegensatz zu Lamport-Uhren oder Vektoruhren, die Ereignisordnung ohne physikalische Verankerung etablieren, ist UKZ an einen physikalischen Oszillator gekoppelt und trägt eine definierte SI-Zeit-Epoche. Sie ist präziser spezifiziert und direkter als Ingenieurstandard umsetzbar.

Offene Fragen für zukünftige Arbeit umfassen: Formale Spezifikation des Epochenanker-Protokolls; Definition der Abbildungsfunktion von UKZ-Ticks auf
UTC/TAI für Interoperabilität; und Analyse des UKZ-Verhaltens unter Hardware-Oszillator-Drift und Resynchronisationsereignissen.

11. Schlussfolgerung

Aktuelle globale Zeitstandards sind architektonisch ungeeignet als Layer-0-Primitive in deterministischen, echtzeitkritischen und verteilten Rechensystemen. UTC verletzt Monotonie. TAI und GPS-Zeit werden in Software oft als Floating-Point-Werte dargestellt und führen Akkumulationsfehler ein. Alle relativistischen Korrekturrahmen setzen eine saubere Basisuhr voraus, die sie selbst nicht bereitstellen.

UKZ löst diese Probleme durch eine einzige architektonische Entscheidung: Definiere die Systemzeitbasis als strikt monotonen, rein ganzzahligen Zähler
$T_{UKZ}(t) = \lfloor (t_{real} - t_0) / \Delta t \rfloor$, und verlange, dass alle höherwertigen zeitlichen Semantiken — einschließlich relativistischer Korrekturen — als Transformationen auf diesem Zähler ausgedrückt werden. Dies ist keine Ablehnung der relativistischen Physik; es ist die korrekte Platzierung der Physik im Software-Stack.

Der Vorschlag ist bereit zur Evaluation in Standardisierungsgremien wie IEEE POSIX, IETF NTP Working Group und ISO TC 154.

Referenzen

[1] Lamport, L. (1978). Time, clocks, and the ordering of events in a distributed system.Communications of the ACM, 21(7), 558–565.

[2] Fidge, C. J. (1988). Timestamps in message-passing systems that preserve the partialordering. Proceedings of the 11th Australian Computer Science Conference, 56–66.

[3] BIPM / ITU-R (2022). Resolution on the future of UTC and the abolition of leap seconds.ITU-R Radiocommunication Assembly, Genf.

[4] IEEE Std 1588-2019 (2020). IEEE Standard for a Precision Clock SynchronizationProtocol for Networked Measurement and Control Systems. IEEE.

[5] POSIX.1-2017, IEEE Std 1003.1-2017. The Open Group Base Specifications Issue 7,2018 Edition.

[6] European Securities and Markets Authority (2017). MiFID II RTS 6 — RegulatoryTechnical Standard on the level of accuracy of business clocks. ESMA.

[7] Nelson, R. A., et al. (2001). The leap second: its history and possible future. Metrologia,38(6), 509–529.

—

Korrespondenz: Dirk Leopold Feiler · Kaiserslautern, Deutschland · ukz-tech.online · contact@ukz-tech.online
Preprint eingereicht: 22. Juni 2026 · Fachklassifikation: cs.DC (Distributed, Parallel, and Cluster Computing), cs.OS (Operating Systems)