**November 18, 2020**

**Computer Timekeeping (and Synchronization)**

**Kevin Stanton, Ph.D.**

**Sr. PrincipalEngineer, Intel Corporation**

Time-Sync Singularity 

▪ What if my local notion of “now” could be used to

unambiguously order events

➔The proof of causality

▪ ***Temporal Inflection Point***

▪ Note: Much of this presentation was developed for a Keynote at ISPCS

2019, in Portland Oregon (International Symposium on Precision Clock

Synchronization)

▪ This was in some ways a continuation of the theme of “The Last Inch

Problem” (Eidson and Stanton at ISPCS, 2015)

Time 

2

From Keynote @ ISPCS 2019, Portland Oregon

A Simple Question: “Is it Now, Now?”

clock\_gettime(CLOCK\_REALTIME, **&now**); *// What does* ***now*** *mean?*

// With respect to whom?

// With what accuracy?

// With what level of trust?

// When actually was **now**?

**The above function returns a Number with units of ns.** 

**Future Applications need to know more!**

Time 

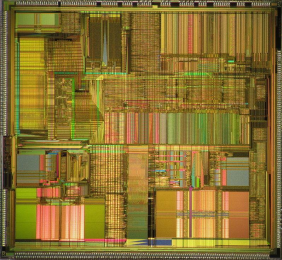
3

Https://archive.org/details/byte Intel Confidential -magazine-1977-11/page/n49?q=timekeeping

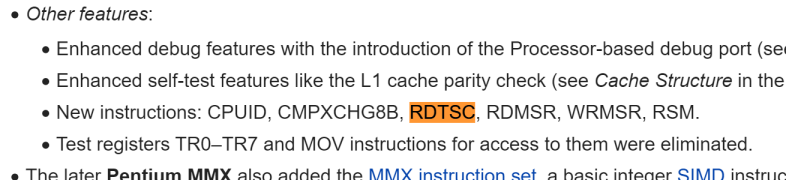
Time

5

Introducing, a Timekeeping



INSTRUCTION!

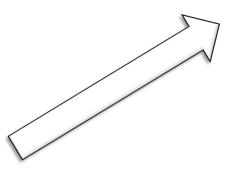


https://en.wikipedia.org/wiki/P5\_(microarchitecture)

Time

6

Games Tuned Timing to the CPU Performance

The “Turbo Button” 

**Time Dilation to un-“Slow Down the World”**

Time 

8

Battery Powered: Low-Power CPU Features

+Turbo

**CPU Frequency Modulated with Demand**

9

https://software.intel.com/en-us/articles/intel-power-gadget

Time

Multi-Core CPUs



**Time Synchronization Required WITHIN the CPU Complex**

Time

10

Oscillator → Local CPU Time (e.g. TSC)

**CPU/SoC Component CPU**

**CPU CPU**

**Always Running Timer (ART)** 

**+offset +offset** 

**\* ÷ +offset** M N

**TSC**

****

**TSC as a Function of Oscillator Frequency**

Time 

11

Some Representative Definitions



https://software.intel.com/content/www/us/en/develop/download/intel-64-and-ia-32-architectures-sdm-combined-volumes-1-2a-2b-2c-2d-3a-3b-3c-3d-and-4.html

Time 

12

Synchronized Time comes From The Outside 

CPU



NIC







LAN

Time Source

E.g. (one of many profiles of) PTP/1588

Time 

13

“NIC” Time is FAR from Software





CPU

Snapshot



NIC







LAN

Time Source

Time 

14

**Time “now”**

CPU Time → Synchronized Time 

**TSC**

(1) clock\_gettime(CLOCK\_MONOTONIC\_RAW, &now);

▪ Returns current TSC value scaled to nominal nanoseconds

(2) clock\_gettime(CLOCK\_MONOTONIC, &now);

▪ Returns current TSC value scaled to track TAI, in nanoseconds





**\*ns/tsc** 

**tick**

(1)

**m** 

(3) clock\_gettime(CLOCK\_REALTIME, &now);

▪ Returns CLOCK\_MONOTONIC + (now-1/1/1970) [incl. leap seconds] 

**c**

**Cross-Timestamp**

**\* +** 

(2)

(3)



(4) ioctl(phc\_fd,PTP\_SYS\_OFFSET[\_PRECISE], &offset ) ▪ returns the triple:

- eth\_ptp\_time; realtime; monotonic\_raw

(4) Snapshot 



**PTP**

**NTP**

**GPS**

**…**

**POSIX: Piecewise-Linear Clock Model: y[n]=mx[n]+c** 

Time 

15

Sample Scenario:

Using PCIe PTM to Cross-Timestamp (PTM=Precision Time Measurement)

1. Device Driver Triggers Cross-Timestamp

2. Device initiates *PTM Request* TLP to Root Complex

3. System Time is Returned (delays are compensated ) 



4. (PTM Time, PTP Time) returned to NIC Device Driver 

Computer System

System

Time

5. Software “disciplines” Coefficients per clock: m (and c) PCIe Root Complex Cross Timestamps,

Delays over PCIe links and through

Captured Simultaneously

From PCIe Specification

System Time\_1 PTP Network Time

Switch





Switches can be compensated



Other I/O



System Time\_2

Other I/O DeviceTime

16 

Device NIC

LAN

**Cross Timestamps** ➔ **‘m’ and ‘c’ Coefficients** 

Time

**The effect of path asymmetry (using .11 TM as the example)**

Station A

Tx/Rx Asymmetry

uncertainty

T

i

m

t

e

o

Software Application

Timestamp

measurement points

uncertainty

T

i

m

R

e

x

s

o

**Hardware Timestamping require a *reference* for Rx and Tx timestamp measurements Each system knows the delay between the PHY and where the actual Rx and Tx timestamps are captured**

**Each timestamp includes some uncertainty The channel also introduces path asymmetry**

x

T

s

t

a

p

m

T

i

m

e

s

t

a

p

m

t

t

a

p

m

Propagation

Delay

R

T

x

i

m

t

o

t

e

o

x

s

T

t

a

p

m

**(and additional uncertainty)**

**In practice, correction of the difference between the Rx offset and the Tx offset in each system is sufficient information**

The rest appears as fixed channel delay and channel uncertainty

uncertaintyuncertainty

Timestamp

measurement points

**Instantaneous time error = Instantaneous delay asymmetry /2**

**Long-term average time error = average delay asymmetry**

Station B

Software Application

Derived from ieee802.org/1/files/public/docs2014/as-kbstanton-8021AS-tutorial-0714-v01.pdf Kevin B. Stanton

Measuring time-synchronization

CPU NIC NIC CPU







oscope

Time 

18

Time 

19

Virtual Time?



Should virtual machine provide virtual time or Physical Time?

Credit: Vikram D. https://software.intel.com/en-us/articles/workload-consolidation-in-industrial-iot

Intel Confidential

The Time-Accuracy Singularity

Trusted Time that’s 

KNOWN to be BETTER

than the BEST-CASE

communication latency

▪ The limit of causality

Intel Confidential

Time

20

Time 

21

Temporal Coordination Demands Dependable Accurate Time Intel Confidential https://www.youtube.com/watch?v=ufK2XRGUjuc

Time 

22

Summary / Call to Action

Software Timekeeping is Ready for the Next Big Challenge:

Help address the big challenges:

1. A plurality of “Nows” that I can TRUST

2. “Now” with QUANTIFIED WORST-CASE ACCURACY wrt Source

3. Reach THE SINGULARITY: Worst-case time accuracy that’s better than the best-case network latency So that Distributed Software knows what it can do with ***“Now”***

Intel Confidential



23