Time from Microseconds to Leap Seconds

or

Dr. StrangeTime

How I learned to stop worrying and love the leap-second

Poul-Henning Kamp

phk@FreeBSD.org

Dramatis Personæ:

Poul-Henning Kamp aka: phk@FreeBSD.org

Kernel hacker, Time-nut, Curmudgeon-in-training

Education:	High-School
Real Life experience:	27 years

Code to prove it:

- FreeBSD kernel
- MD5crypt (All your passwords are belong...)
- phkmalloc(3)
- Timecounters
- GEOM
- Jails (Semi-transparent virtual machines)
- Varnish (HTTP cache, all your WWW are belong...)
- Etc.

1969(?)

1971

Ti, enogtyve og fyrre .... <duut>
Ti, enogtyve og halvtreds...<duut>

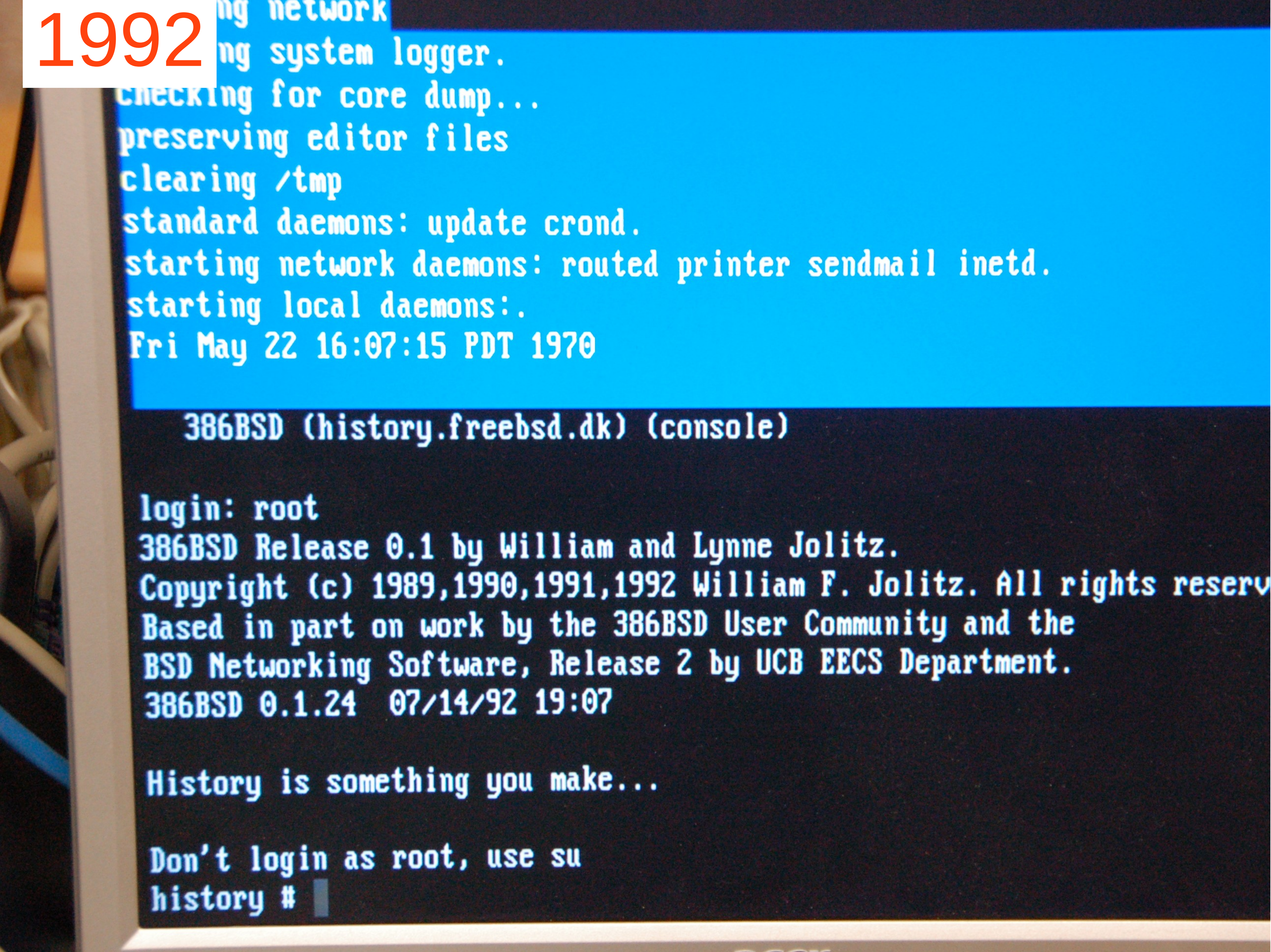
1975

1981

1987

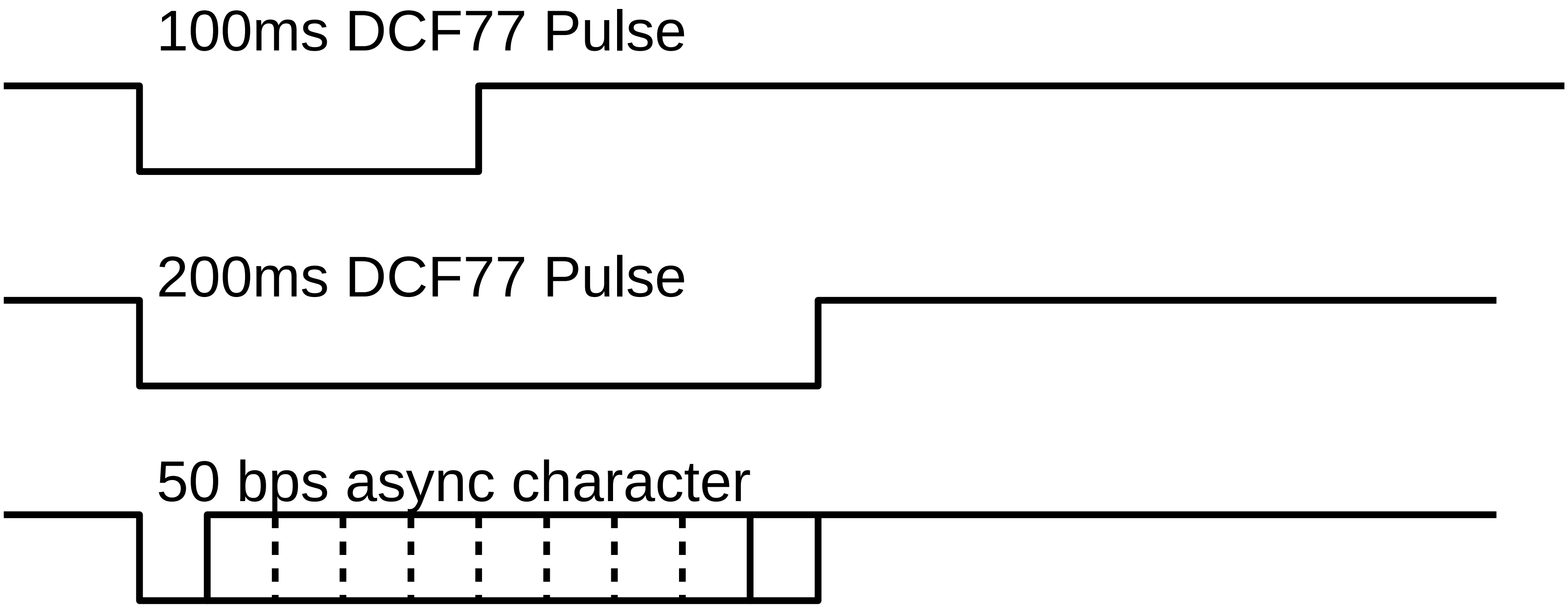

Zilog's present
to the future
system
8000
TM
Designed for tomorrow's user
as well as today's

1992


ng system logger.
for core dump...
preserving editor files
clearing /tmp
standard daemons: update crond.
starting network daemons: routed printer sendmail inetd.
starting local daemons:.
Fri May 22 16:07:15 PDT 1970
386BSD (history.freebsd.dk) (console)
login: root
386BSD Release 0.1 by William and Lynne Jolitz.
Copyright (c) 1989,1990,1991,1992 William F. Jolitz. All rights reserv
Based in part on work by the 386BSD User Community and the
BSD Networking Software, Release 2 by UCB EECS Department.
386BSD 0.1.24 07/14/92 19:07
History is something you make...
Don't login as root, use su
history #

1993


14:10 14
06.10.05
DCF-Zeit
DCF-Takt
MESZ
Mode
Set
Relais
Mo
Di
Mi
Do
Fr
Sa
So
Digitales Funkuhrensystem
DCM - 77


100ms DCF77 Pulse
200ms DCF77 Pulse
50 bps async character

1994

Get a job with TRW Financial Systems
Move to SF East-bay for training
Boy meets girl etc.
Release FreeBSD 2.0 etc
md5crypt(3)
phkmalloc(3)
lots and lots of FreeBSD code

1996

Back home in Denmark:

Microbenchmarks

”Boot two identical machines diskless,
run the test, for days if need be,
use oscilloscope on parallel port pins
to measure difference in speed.”

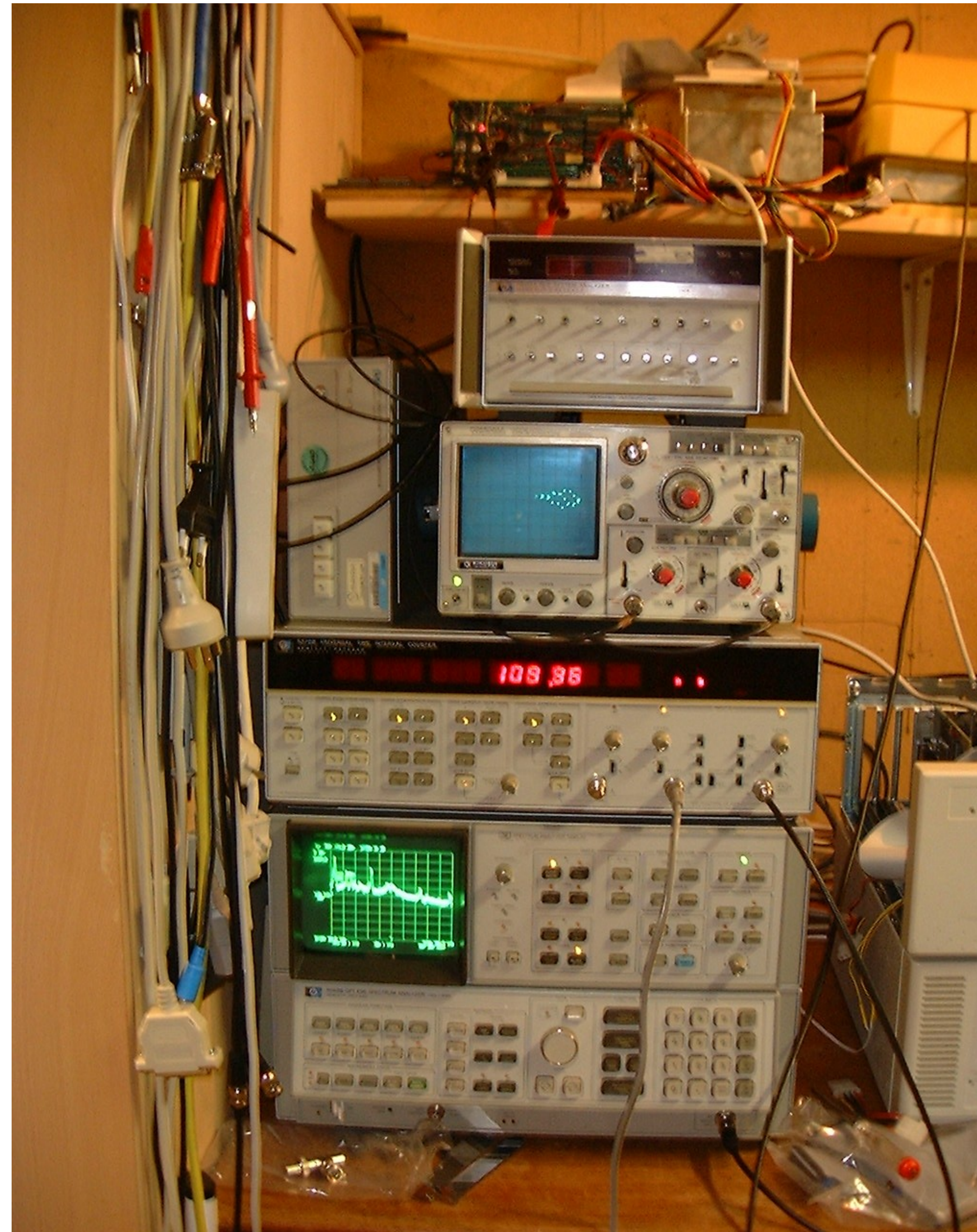
”Stddev not impressive”

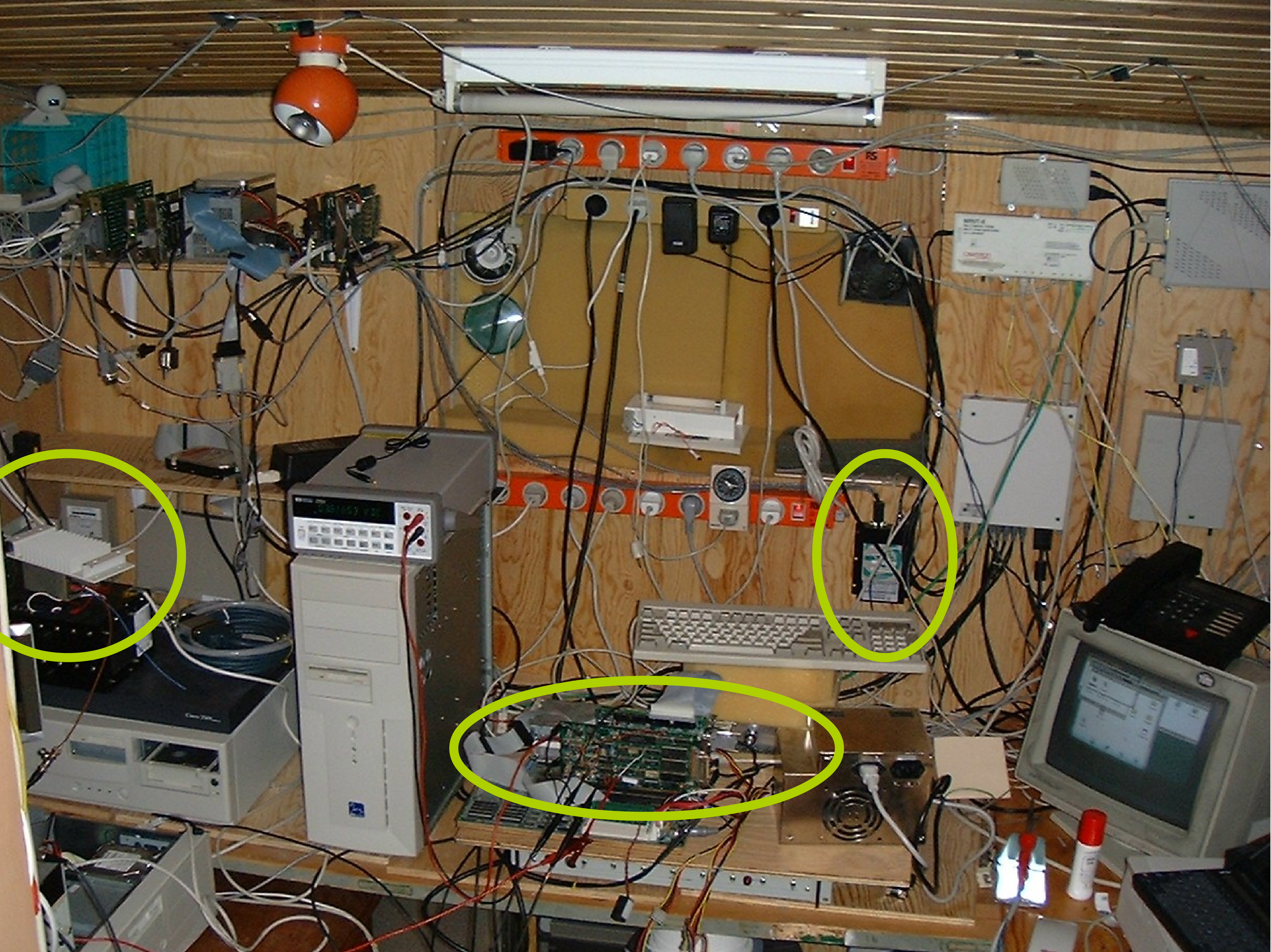
Start of serious time-nuttery

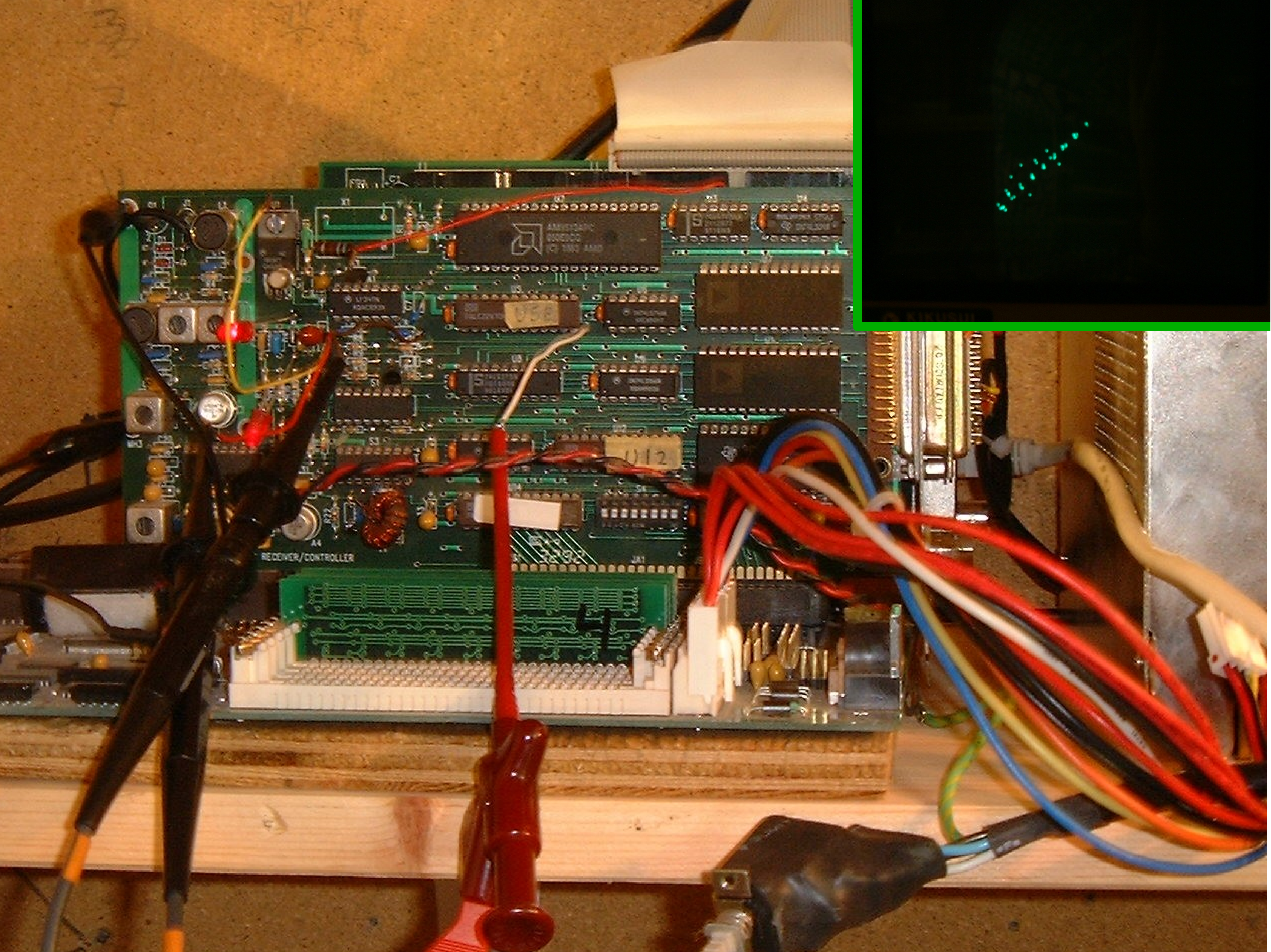
A Big Thank You! to:

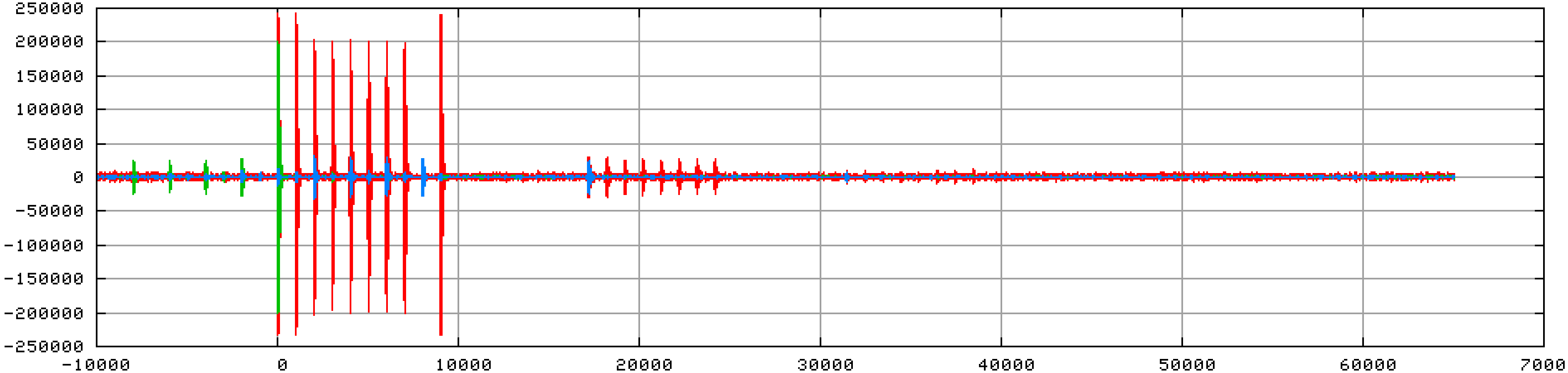
John R. Vig's tutorial
Dave Mills
Corby Dawson
Tom V. Baak

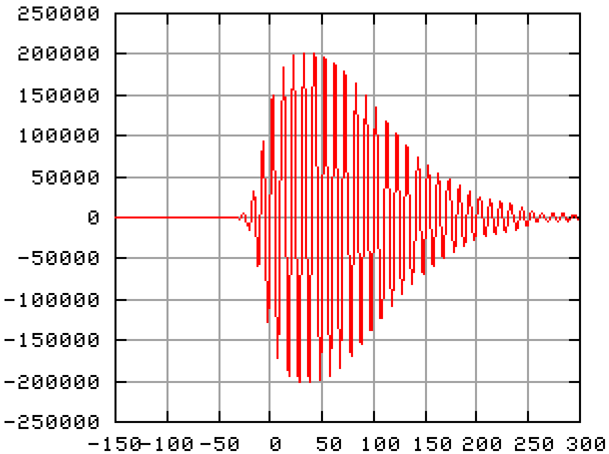
& eBay



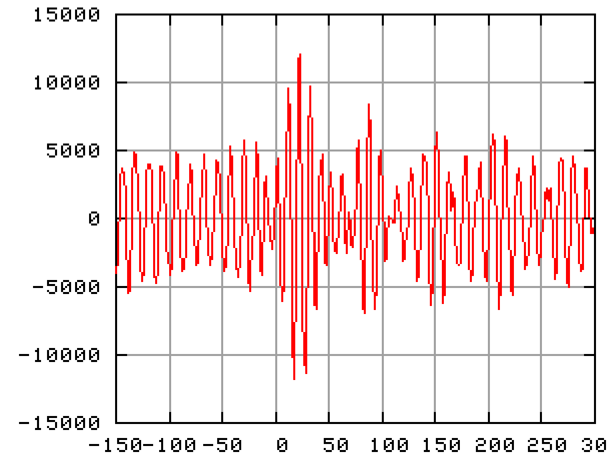



RECEIVER/CONTROLLER

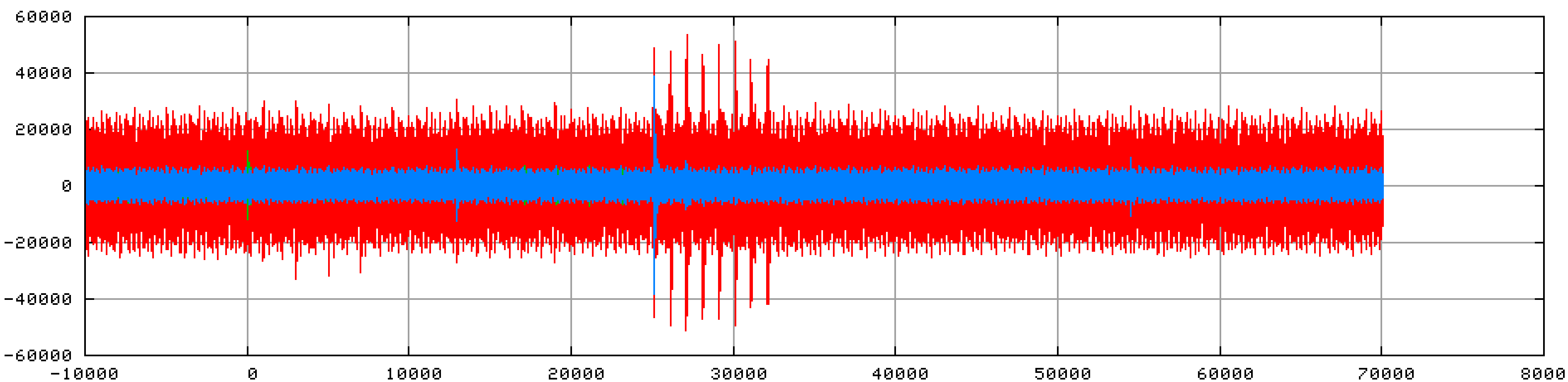


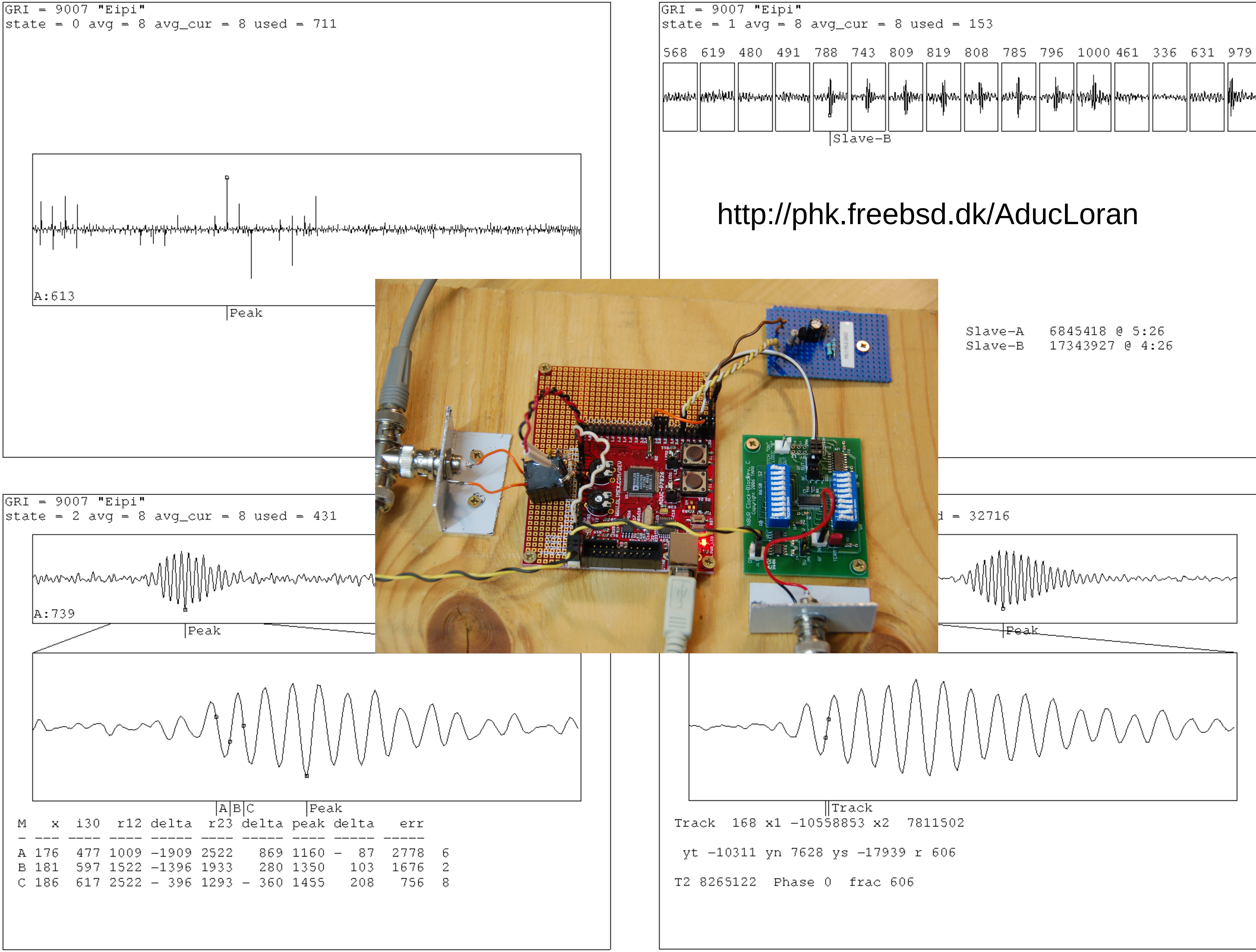


7499M Loran-C



8000-X Chayka




GRI = 9007 "Eipi"
state = 0 avg = 8 avg_cur = 8 used = 711
A:613
Peak
GRI = 9007 "Eipi"
state = 1 avg = 8 avg_cur = 8 used = 153
568 619 480 491 788 743 809 819 808 785 796 1000 461 336 631 979
Slave-B
http://phk.freebsd.dk/AducLoran
Slave-A 6845418 @ 5:26
Slave-B 17343927 @ 4:26
GRI = 9007 "Eipi"
state = 2 avg = 8 avg_cur = 8 used = 431
A:739
Peak
A B C Peak
M x i30 r12 delta r23 delta peak delta err
A 176 477 1009 -1909 2522 869 1160 - 87 2778 6
B 181 597 1522 -1396 1933 280 1350 103 1676 2
C 186 617 2522 - 396 1293 - 360 1455 208 756 8
Peak
Track
Track 168 x1 -10558853 x2 7811502
yt -10311 yn 7628 ys -17939 r 606
T2 8265122 Phase 0 frac 606

Timecounters: Efficient and precise timekeeping in SMP kernels.

Poul-Henning Kamp
The FreeBSD Project

ABSTRACT

The FreeBSD timecounters are an architecture-independent implementation of a binary timescale using whatever hardware support is at hand for tracking time. The binary timescale converts using simple multiplication to canonical timescales based on micro- or nano-seconds and can interface seamlessly to the NTP PLL/FLL facilities for clock synchronisation. Timecounters are implemented using lock-less stable-storage based primitives which scale efficiently in SMP systems. The math and implementation behind timecounters will be detailed as well as the mechanisms used for synchronisation.

Introduction

Despite digging around for it, I have not been able to positively identify the first computer which knew the time of day. The feature probably

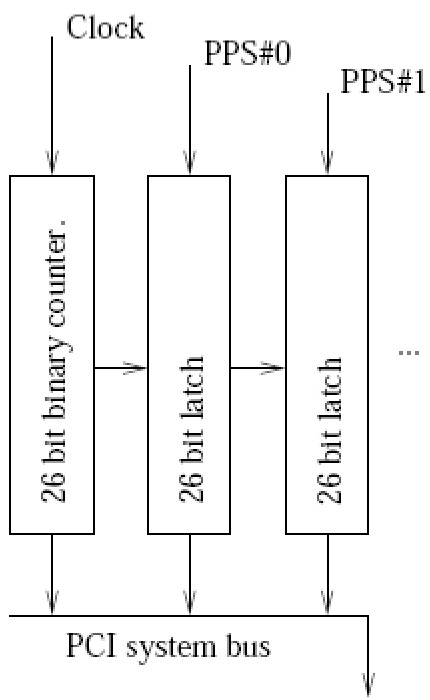
million years, provided we stick to the preventative maintenance schedules. This is a feat roughly in line with to knowing the circumference of the Earth with one micrometer precision, in real time.

Ideal timecounter hardware

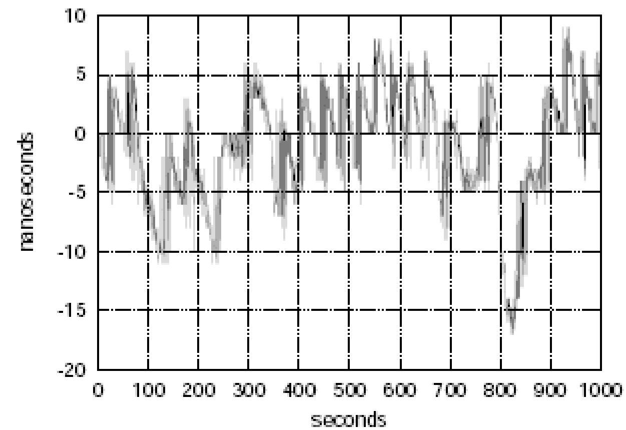
As proof of concept, a sort of an existentialist protest against the sorry state describe above, the author undertook a project to prove that it is possible to do better than that, since none of the standard hardware offered a way to fully validate the timecounter design.

Using a COTS product, ''HOT1'', from Virtual Computers Corporation [VCC2002] containing a FPGA chip on a PCI form factor card, a 26 bit timecounter running at 100MHz was successfully implemented.

In order to show that timestamping does not necessarily have to be done using unpredictable and uncalibratable interrupts, an array of latches were implemented as well, which allow up to 10 external signals to latch the reading of the counter when an external PPS signal transitions from logic high to logic low or vice versa.

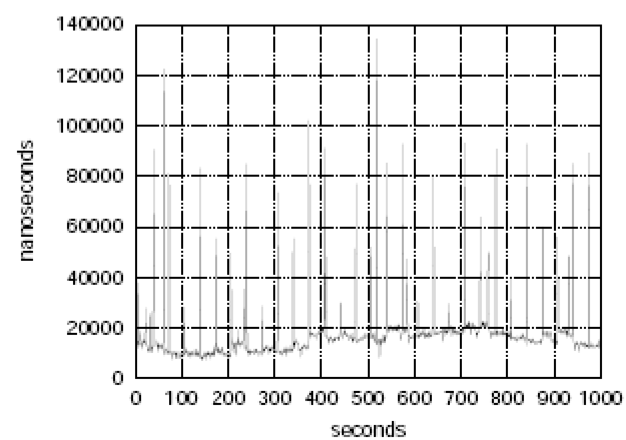


Using this setup, an standard 133 MHz Pentium based PC is able to timestamp the PPS output of the Motorola UT+ GPS receiver with a precision of ± 10 nanoseconds ± one count which in practice averages out to roughly ± 15 nanoseconds[9]:

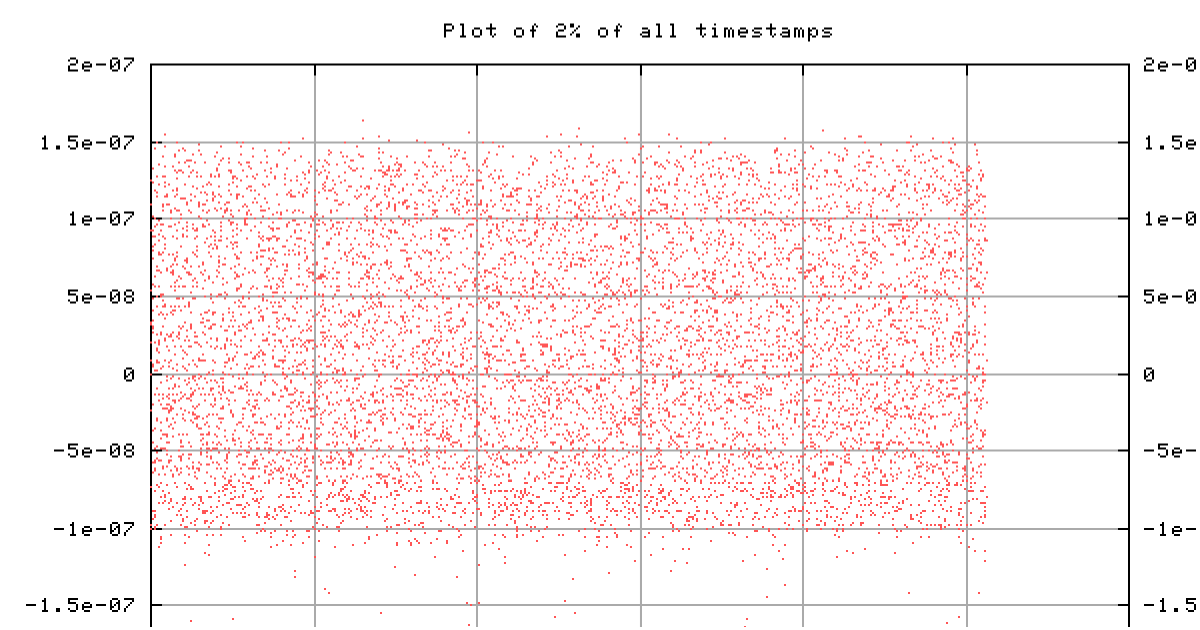


It shold be noted that the author is no hardware wizard and a number of issues in the implementation results in less than ideal noise performance.

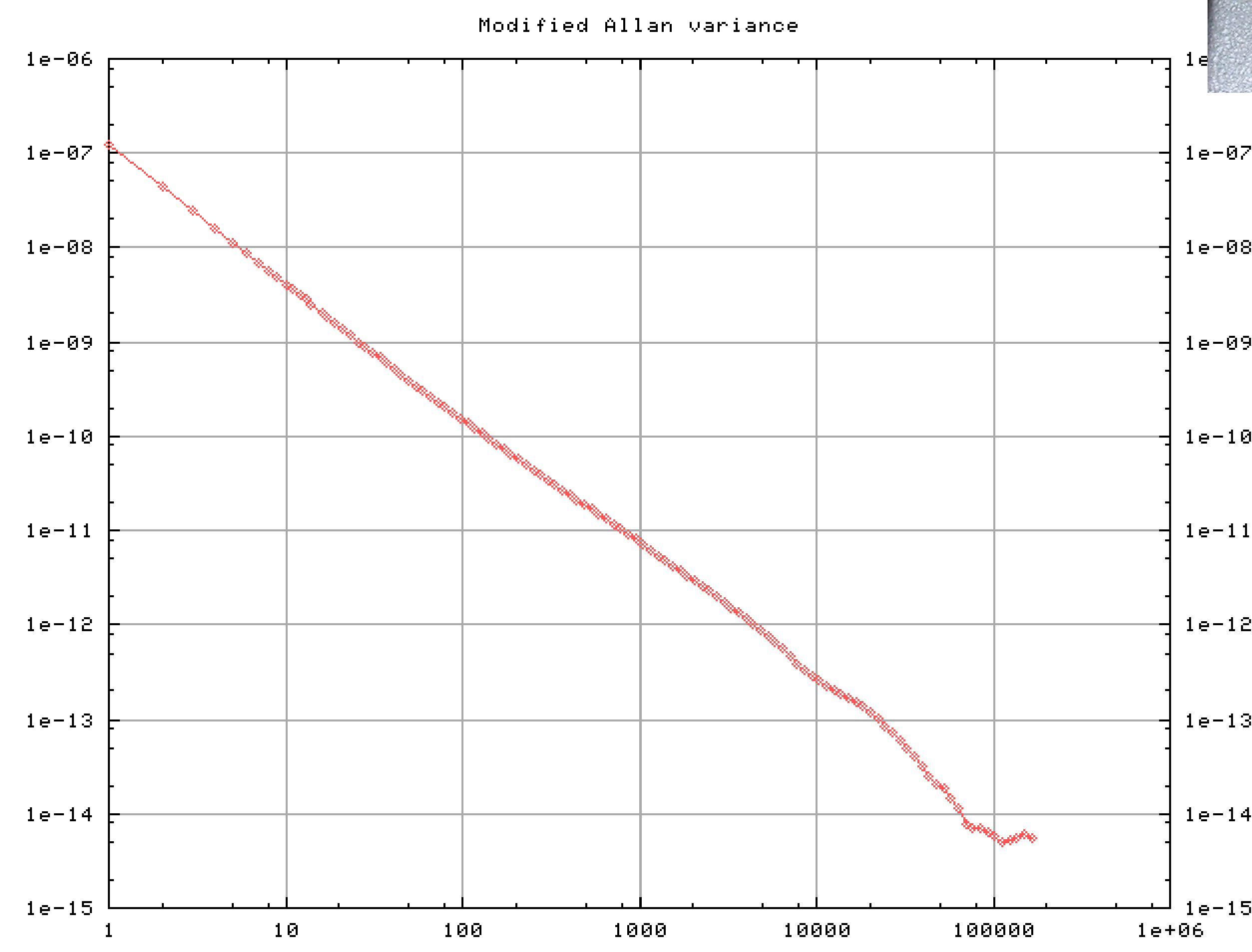
Now compare this to ''ideal'' timecounter to the normal setup where the PPS signal is used to trigger an interrupt via the DCD pin on a serial port, and the interrupt handler calls `nanotime()` to timestamp the external event [10]:

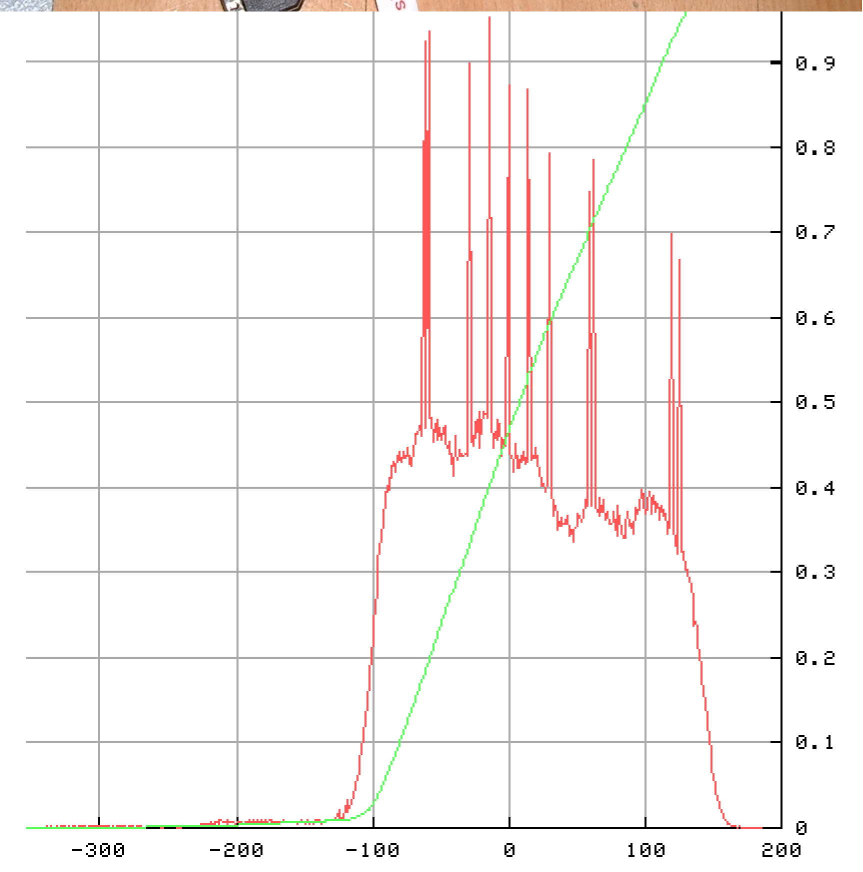


It is painfully obvious that the interrupt latency is the dominant noise factor in PPS timestamping in the second case. The asymetric distribution of the


Plot of 2% of all timestamps
2e-07
1.5e-07
1e-07
5e-08
0
-5e-08
-1e-07
-1.5e-07


PCIbus
PCMCIA
Serial Card
ARGOSY
sio4
sio5


Modified Allan variance
1e-06
1e-07
1e-08
1e-09
1e-10
1e-11
1e-12
1e-13
1e-14
1e-15
1
10
100
1000
10000
100000
1e+06


0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-300
-200
-100
0
100
200

The Nanokernel[1]

David L. Mills[2]
University of Delaware
and
Poul-Henning Kamp
FreeBSD Project

Abstract

Internet timekeeping has come a long way since first demonstrated almost two decades ago. In that era most computer clocks were driven by the power grid and wandered several seconds per day relative to UTC. As computers and the Internet became ever faster, hardware and software synchronization technology became much more sophisticated. The Network Time Protocol (NTP) evolved over four versions with ever better accuracy now limited only by the underlying computer hardware clock and adjustment mechanism.

The clock frequency in modern workstations is stabilized by an uncompensated quartz or surface acoustic wave (SAW) resonator, which are sensitive to temperature, power supply and component variations. Using NTP and traditional Unix kernels, incidental timing errors with an uncompensated clock oscillator is in the order of a few hundred microseconds relative to a precision source. Using new kernel software described in this paper, much better performance can be achieved. Experiments described in this paper demonstrate that errors with a modern workstation and uncompensated clock oscillator are in the order of a microsecond relative to a GPS receiver or other precision timing source.

Something funny happened on the way to the airport...

CASIMO:

Total replacement of Danish ATC system

$300M + 13 year project

First component specified: ”MasterClock”

NTP
Better than 1/256th second
SNMP management

”We called Dave Mills, and he told us to call you...”

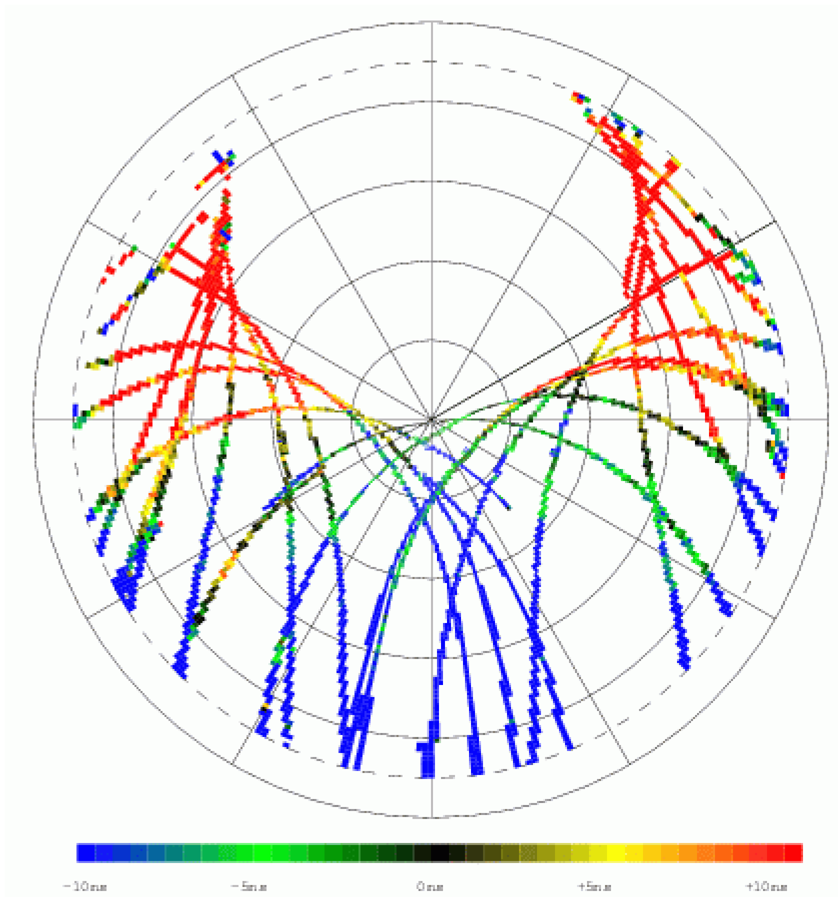

OCXO
goes
here
TRACO
POWER

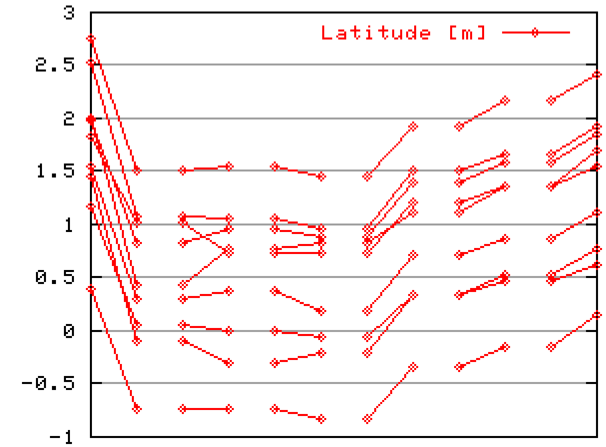
NTPns

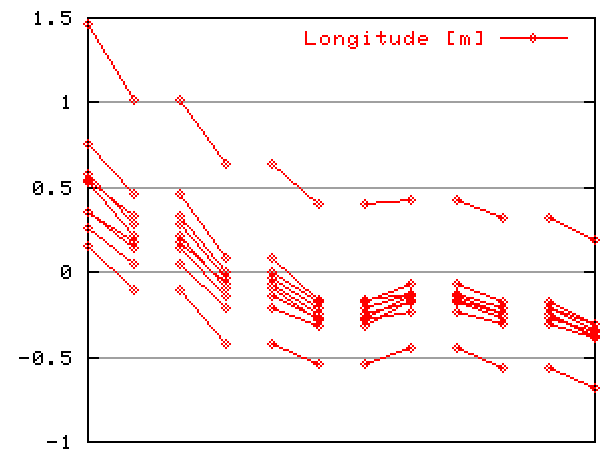
- NTP for Nanoseconds
- Focused on Primary NTP service
- Multiple refclocks
- Clock combination instead of selection
- Modular
- Manageable
- Stats collection
- &c.

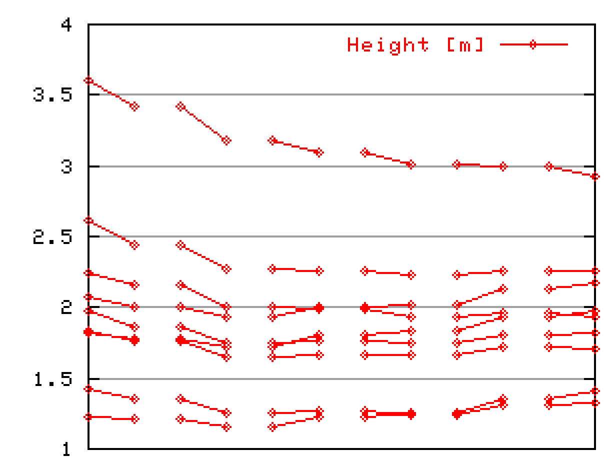
```
gps# uptime
 9:39PM  up 130 days, 11:16, 1 user, load averages: 0.33, 0.15, 0.10
gps# telnet localhost 123
Trying ::1...
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
NTPns > show oncore 0
serial port = /dev/cuad4        serial number = P05YWT
state = 12      visible/track/lock = 8/8/8      dop = 0.0 [m]
raim_limit 0.000001000/0.000001000 mask angle 10/10
2011-08-23 21:39:33.000728474
Leap second info: 2013-11-28 00:00:00 NONE
            71461227 seconds (827 days) from now
lat = 200823845 (55.784401), lon = 45071571 (12.519881), ht 8894 (88.94)
http://maps.google.com/maps?ll=55.784401,12.519881&spn=0.03,0.08&t=k
flat = 200823845, flon = 45071571, fht 8894
rcv_status = 0x8400 = PosHold NarrowTrack AntOK
raim_solution = OK, raim_status = detection+isolation
raim_removed = 00000000 raim_1sigma = 0.000000038 [s]   raim_sawtooth = 2 [ns]
clock_bias = 13 [ns]    osc_offset = 91307 [Hz] osc_temp = 32.0 [C]
utc_status = 0xcf = enabled decoded     utc_offset = 15
site_survey = 0 (~0 sec left)
Sat Dopler Elev Azi Health Mode SigStr IODE Status Offset
 11  -1570   65 170   00      8    53     71   8a0  0.000670518
 14  -1979   35  53   00      8    48     64   8a0  0.000670534
 17   2690   24 314   00      8    45     12   8a0  0.000670527
 19  -3793   13 173   00      8    43     87   8a0  0.000670505
 20   2586   45 243   00      8    52     77   8a0  0.000670519
 24   -145   83 210   00      8    52     86   8a1  0.000670514
 28  -2245   12 273   00      8    42     45   8a0  0.000670493
 32    692   74 219   00      8    52     87   8a0  0.000670517
NTPns > show ntpv4 0 partner
IP number         port leap v m  s  p   P         offset refid
Max partners:            10000
                     total   ours others
partners              2279   1053   1226
partners good         1923    973    950
partners bad           356     80    276
partners > 1s          188     18    170
partners < 1s          178     65    113
partners < 100ms       349     65    284
partners < 10ms        649    232    417
partners < 1ms         915    673    242
NTPns >
```


http://fs.freebsd.dk/cgi-bin/ntp_pll.cgi?w=360&h=160&l=8y&s=ntp_dix
casimo naviair
Achenblog BAB RC MAKE Froomkin DIY Ing:PV ACM Queue Music
DCF77 GPS DIX -- 300s 900s 1h 3h 12h 1d 2d 3d 4d 1w 1m 1y -- PLLA RES SRC
st
20% sdy 10% sdy a my
b frequency offset
c min = -2.397620e-16 avg = 1.010454e-15
max = 2.922909e-13
sdy 20% sdy 10% sdy y my
z
wa wamin min(wa) = 9.550050e-05
dl zc
j
RRDTOOL / TOBI OETIKER


-10ns
-5ns
0ns
+5ns
+10ns


Latitude [m]
3
2.5
2
1.5
1
0.5
0
-0.5
-1


Longitude [m]
1.5
1
0.5
0
-0.5
-1


Height [m]
4
3.5
3
2.5
2
1.5
1

NTP Vandalism

GPS.dix.dk – Only for stratum 2 use, by prior agreement.

Free bandwidth & hosting at Danish Internet eXchange (DIX)

D-Link sold millions of routers hardcoded for public NTPs1 list.

Significant packet load for a poor i486 133MHz machine.

Took Open Letter + Slashdot & ComputerWorld to wake D-Link up

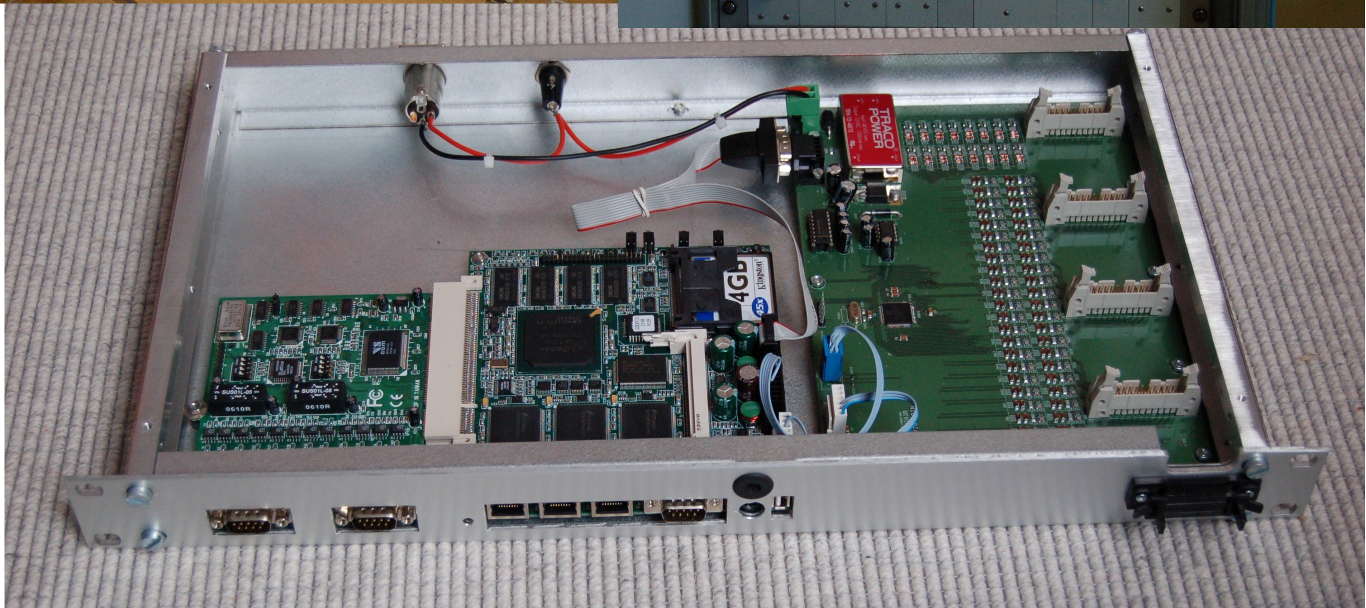
D-Link paid for bandwidth used

NTP Vandalism

- Certain PC malware used NTP to coordinate dDoS attacks
- Hardcoded NTPs1 list (again!)
- Easily Recognizable Packets
- gps.dix.dk provided almost complete list of infected machines

FERN07

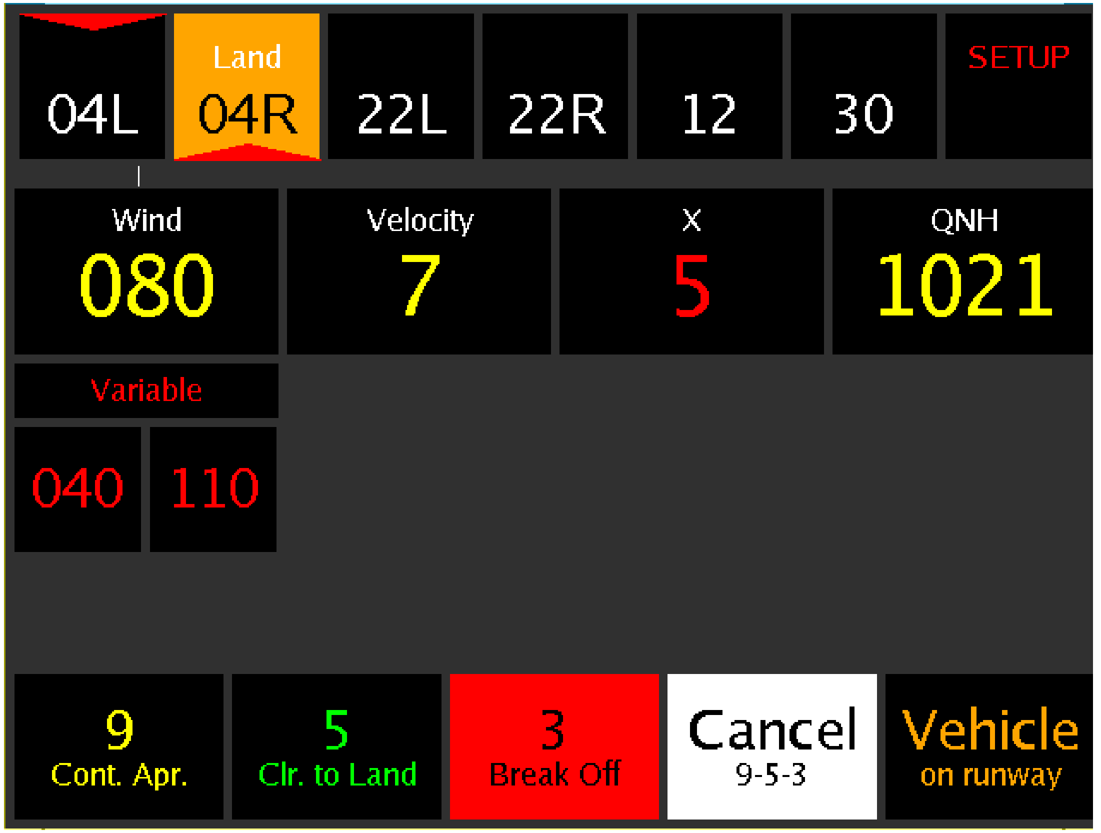
VOR + DME proxy
- SNMP interface
- Logging

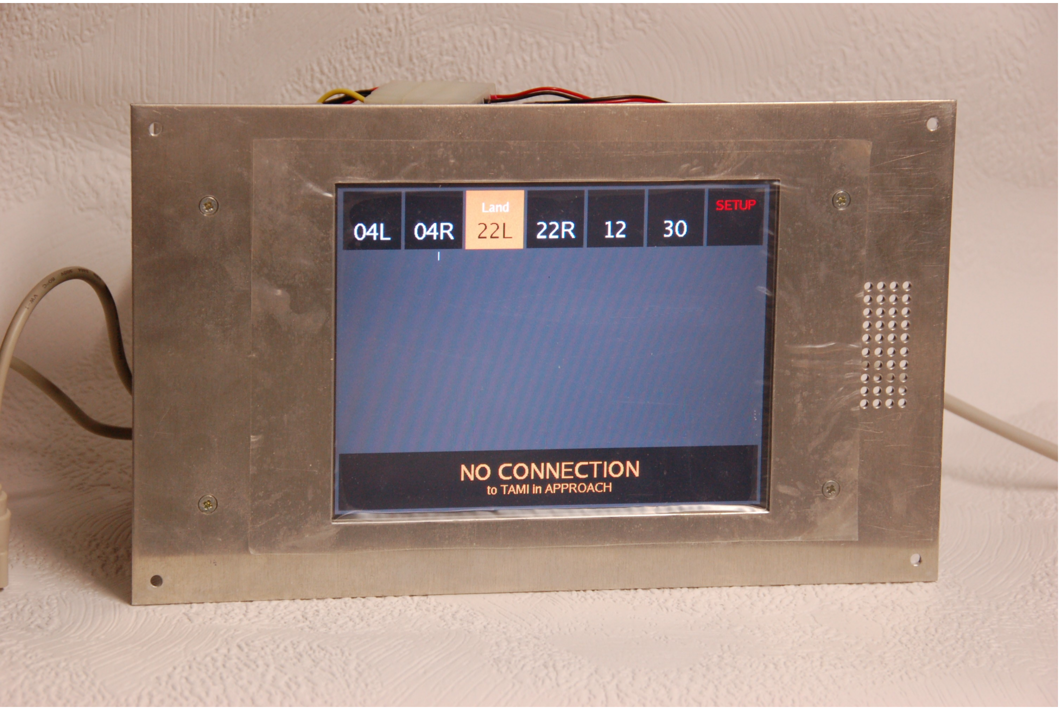


TAMI

Flight Controller Panel

- Runways In Use
- Runway restrictions
- METOPS
- 9-5-3 Alerts





http://pubs.opengroup.org/onlinepubs/009695399/toc.htm

A.4.14 Seconds Since the Epoch

Coordinated Universal Time (UTC) includes leap seconds. However, in POSIX time (seconds since the Epoch), **leap seconds are ignored** (not applied) to provide an easy and compatible method of computing time differences. Broken-down POSIX time is therefore not necessarily UTC, despite its appearance.

[...]

Most systems' notion of "time" is that of a continuously increasing value, so this value should increase even during leap seconds. **However, not only do most systems not keep track of leap seconds, but most systems are probably not synchronized to any standard time reference.** Therefore, it is inappropriate to require that a time represented as seconds since the Epoch precisely represent the number of seconds between the referenced time and the Epoch.

Legal POSIX leap-second behaviour:

23:59:58	23:59:58	23:59:58	23:59:58.00
23:59:59	23:59:59	23:59:59	23:59:58.75
Hang	23:59:59	00:00:00	23:59:59.50
00:00:00	00:00:00	00:00:00	00:00:00.25
00:00:01	00:00:01	00:00:01	00:00:01.00

Interval time == absolute time

```
T += 3600;          /* Same time next hour */
```

Or

```
T += 3600;          /* Again in an hour */


T -= T % 86400;     /* Start of today */
```

http://support.microsoft.com/kb/909614

When the Windows Time service is working as a Network Time Protocol (NTP) client

The Windows Time service does not indicate the value of the Leap Indicator when the Windows Time service receives a packet that includes a leap second. (The Leap Indicator indicates whether an impending leap second is to be inserted or deleted in the last minute of the current day.) **Therefore, after the leap second occurs, the NTP client that is running Windows Time service is one second faster than the actual time. This difference is resolved at the next time synchronization**
.

When the Windows Time service is working as an NTP server

No method exists to include a leap second for the Windows Time service.

How a leap second is included depends on NTP server settings.

Who's afraid of leapseconds ?

skudsekund

About 17 results (0.08 seconds)

Jan 1, 2005–Jun 1, 2006

Skudsekund - Wikipedia, den frie encyklopædi - [Translate this page]
da.wikipedia.org/wiki/**Skudsekund** - Cached
24 Nov 2005 - Et **skudsekund** er et ekstra sekund, der af og til indsættes i den almindelige tidsregning UTC for at sikre, at det almindelige klokkeslæt bliver ved med at ...
Hvorfor skudsekunder? - Hvordan indsættes skudsekundet?

Network Time Protocol - Wikipedia, den frie encyklopædi - [Translate thi
da.wikipedia.org/wiki/Network_Time_Protocol - Cached
7 Feb 2006 - Et **skudsekund** skal registreres i referencetidskilderne samme dag som det ...

Show more results from wikipedia.org

Tiden går i stå et sekund nytårsnat - dr.dk/Nyheder/Indland - [Translate
www.dr.dk › Nyheder › Indland - Cached
27 Dec 2005 - 00:59:59 bliver til 00:59:60, inden klokken slår 01:00:00. Der bliver simpelthen sat et **skudsekund** ind, så tiden passer til Jordens rotation.

What actually happens during leap-seconds ?

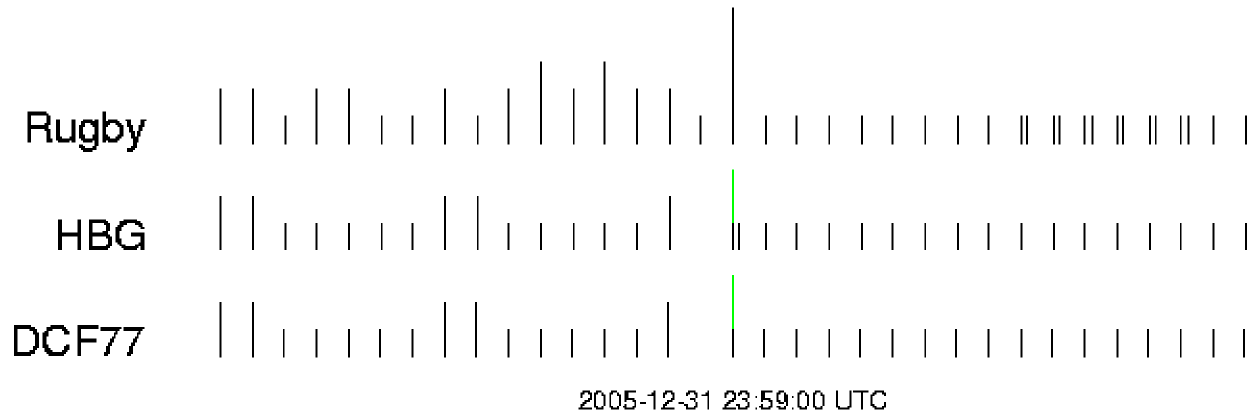
25% of all NTP Stratum 1 servers gets it wrong

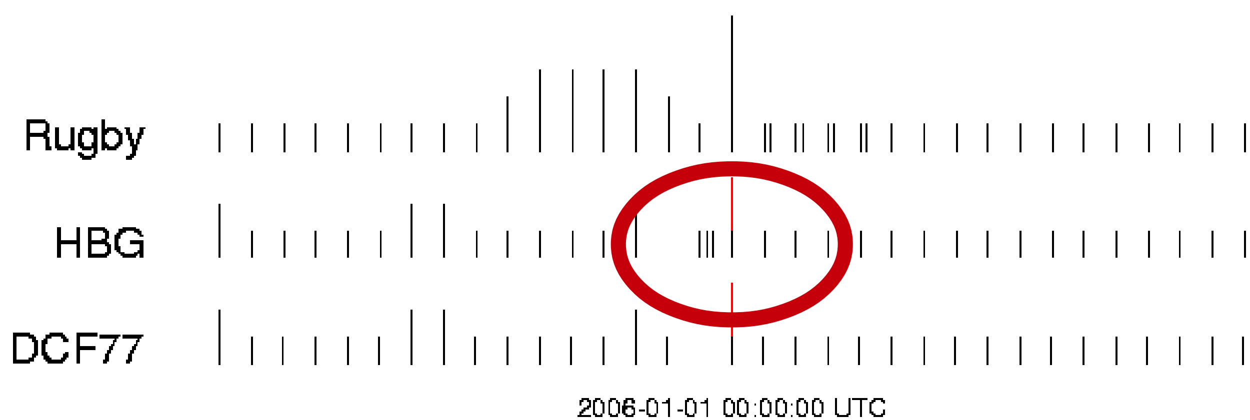
33% of all time broadcasts gets it wrong

Airplanes told ”You're on your own until further notice”
- ”Really interesting light show at the ATC console...”
- Airplane moves ~300m/sec

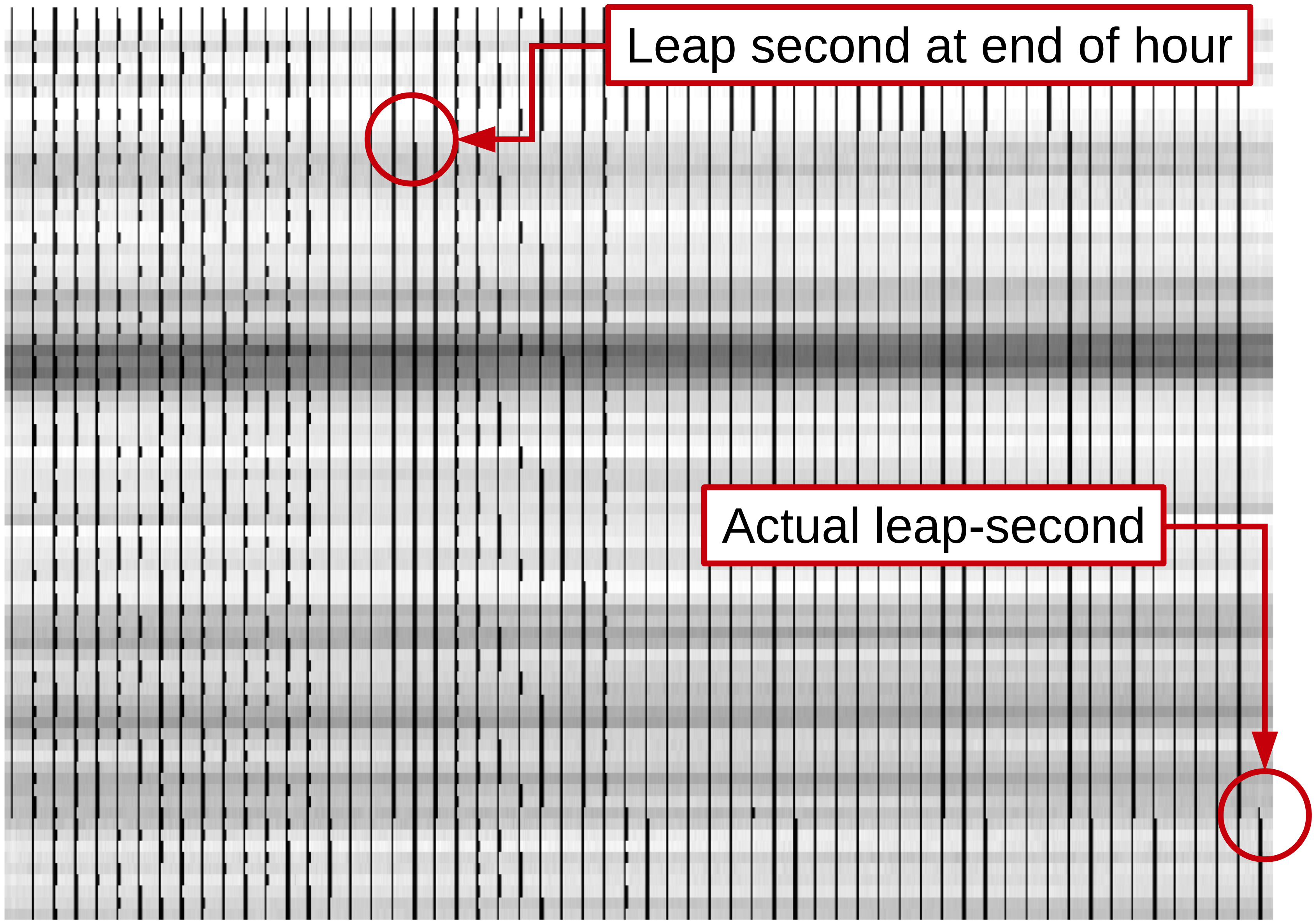
Insulin production facilities shut down
- Scheduled maintenance window moved.
- FDA tracking requirement: ”Precision of 1 second, traceable”

US Nuclear deterrent in ”special mode”
- ”Cost of multi-digit millions”
- (unconfirmed)

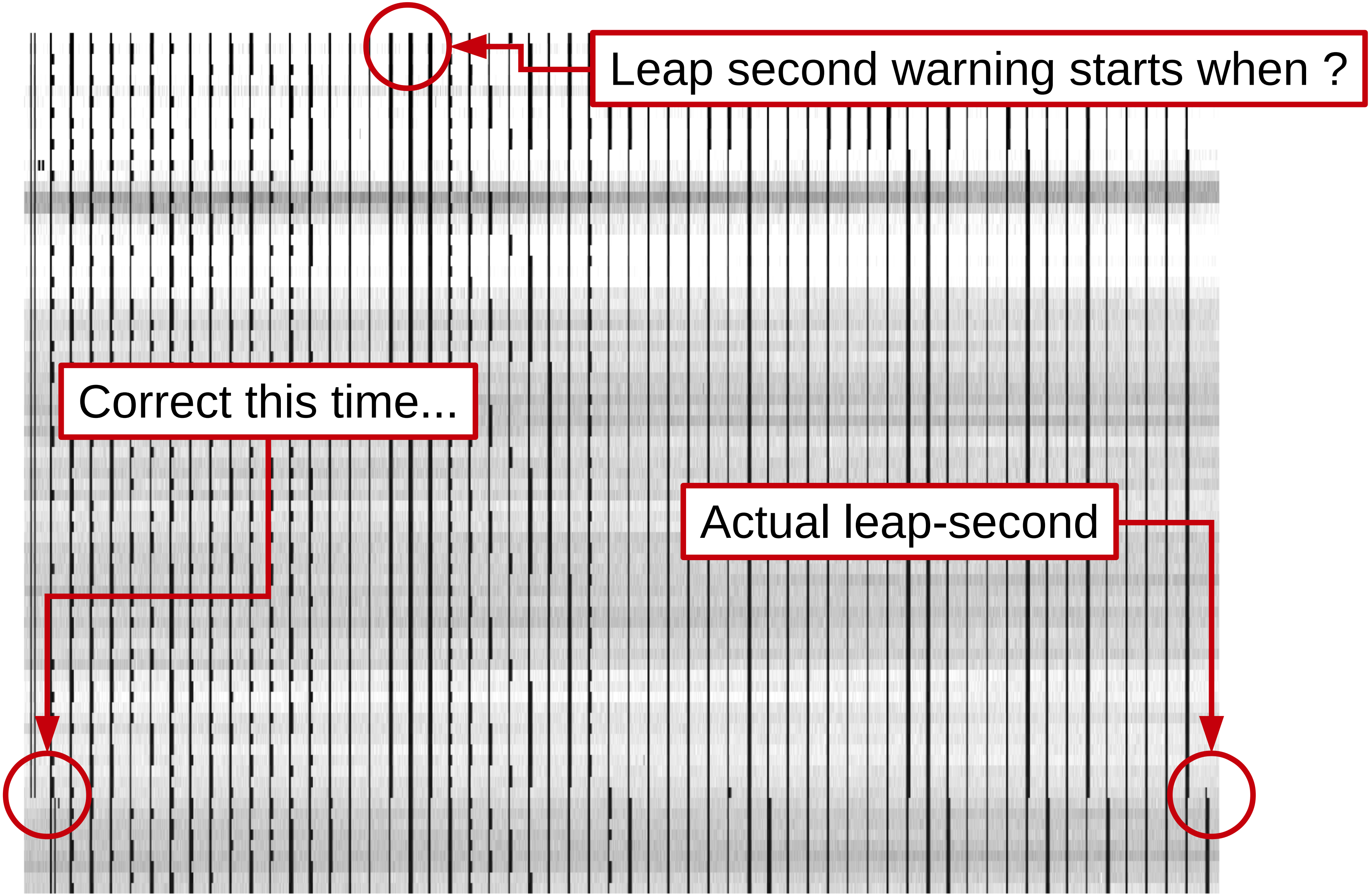

Rugby
HBG
DCF77
2005-12-31 23:59:00 UTC


Rugby
HBG
DCF77
2006-01-01 00:00:00 UTC

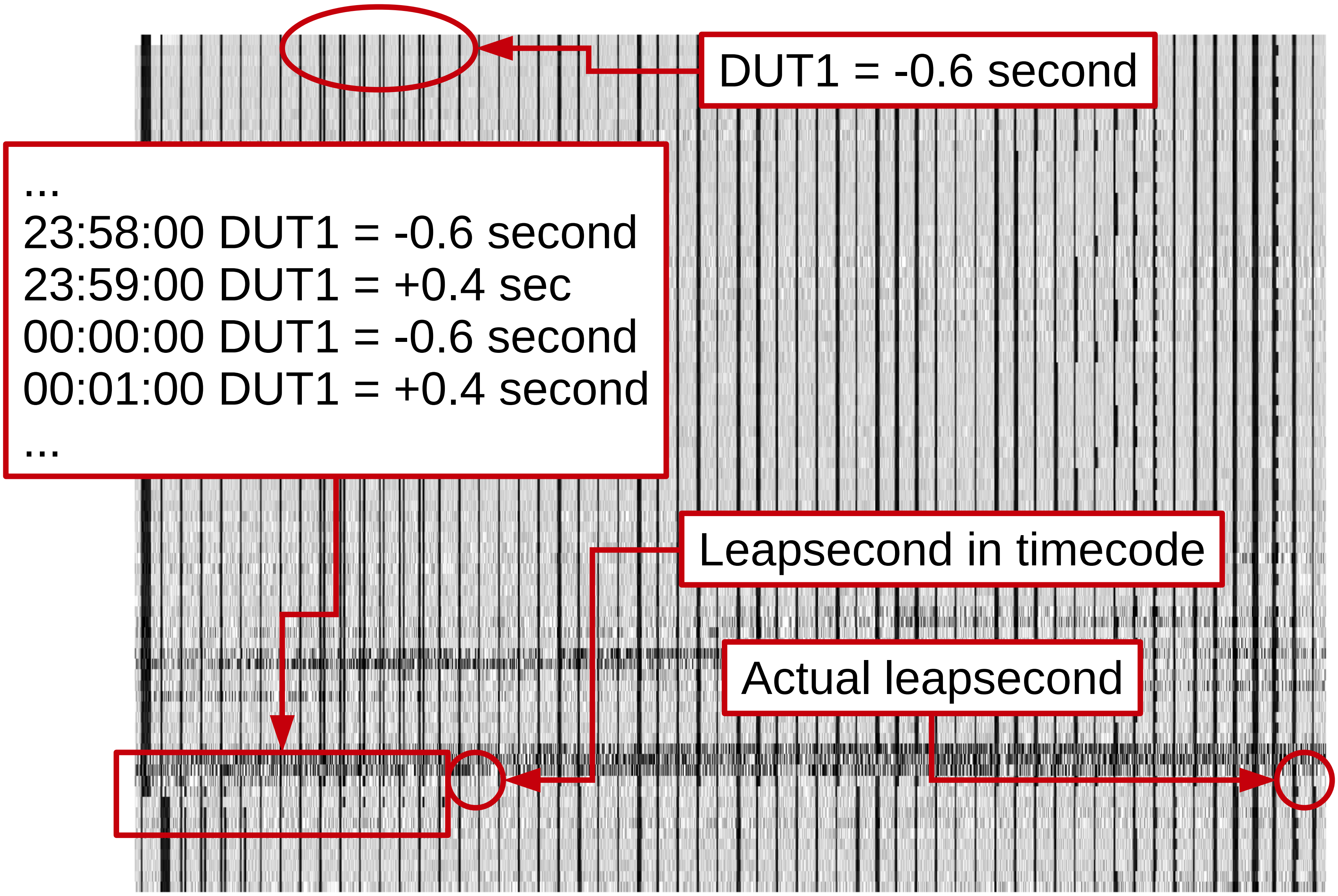
DCF77 77.5kHz -- 2007-12-31 leap second

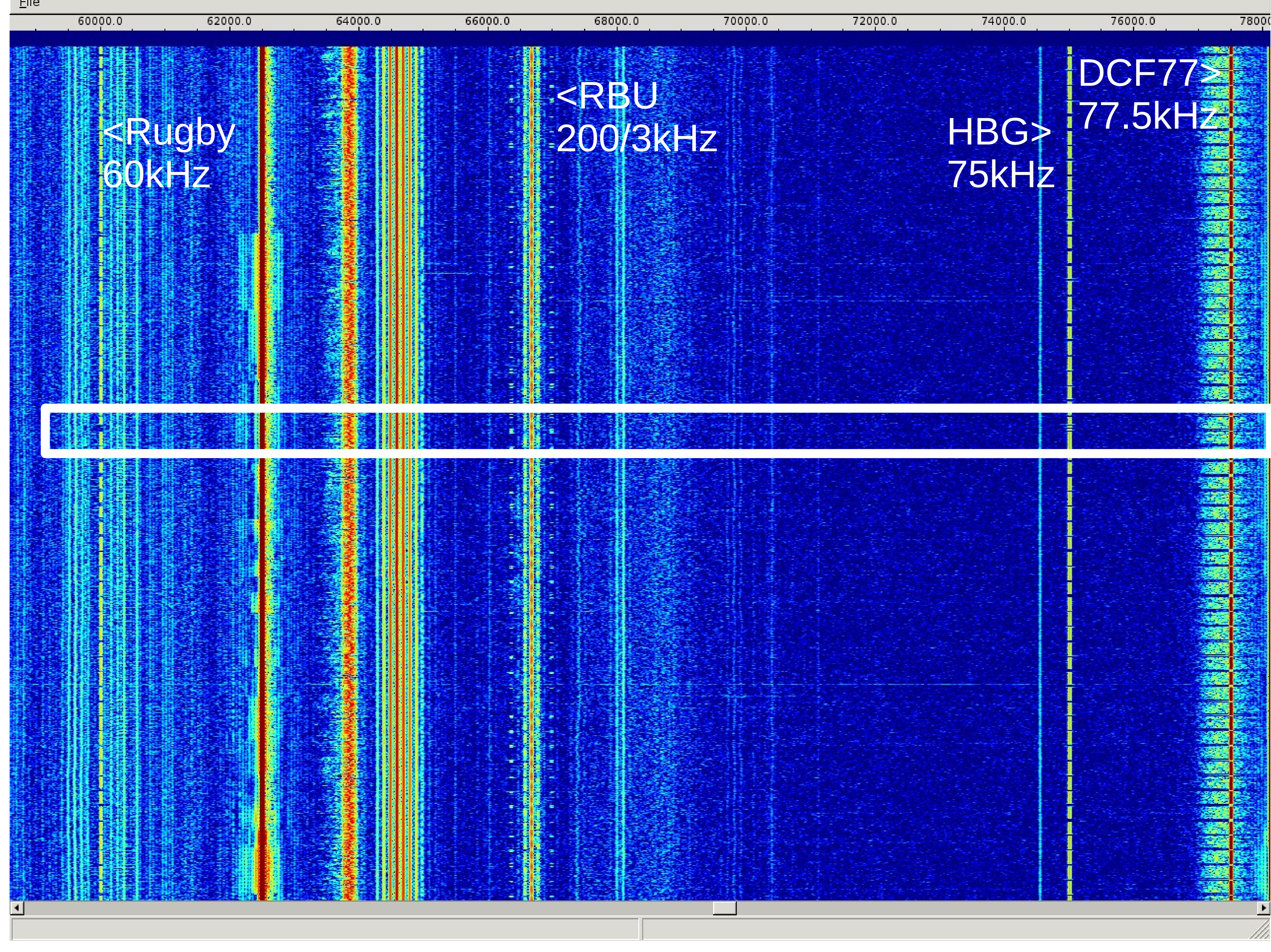


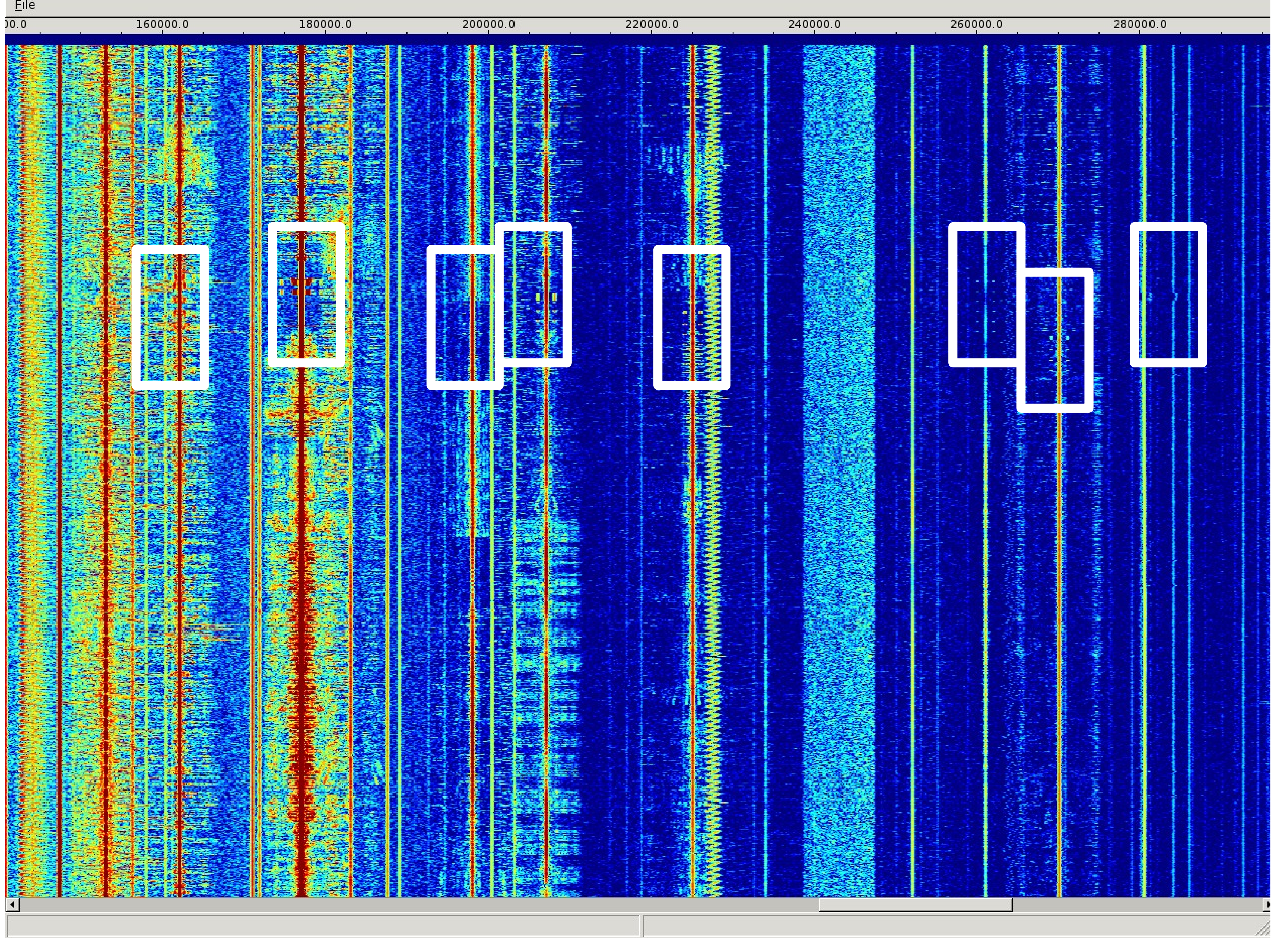
HBG 75kHz -- 2007-12-31 leap second



MSF 60kHz -- 2007-12-31 leap second




File
60000.0
62000.0
64000.0
66000.0
68000.0
70000.0
72000.0
74000.0
76000.0
<Rugby
60kHz
<RBU
200/3kHz
HBG>
75kHz
DCF77>
77.5kHz


File
00.0
160000.0
180000.0
200000.0
220000.0
240000.0
260000.0
280000.0

What will happen during the next leap-seconds ?

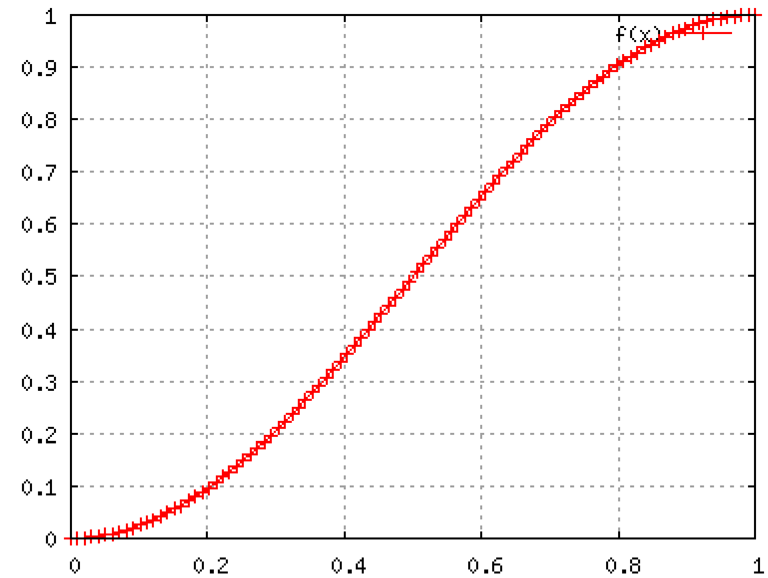
- Modern robotic productionlines will have to shut down
  - Semiconductors / Photovoltaic / Cars / Electronics
  - IEEE-1588: microsecond synchronization for robots.

- Systems will go haywire ”unexpectedly”
  - SCADA, Building automation, Alarm systems etc.
  - ”UNIX did one thing, Windows another etc.”

- Consultants will make fortunes
  - See Y2K.

Googleblog (2011-09-15)

lie(t) = (1.0 - cos(pi * t / w)) / 2.0



What we learned

The leap smear is talked about internally in the Site Reliability Engineering group as one of our coolest workarounds, that took a lot of experimentation and verification, but paid off by ultimately saving us massive amounts of time and energy in inspecting and refactoring code. It meant that we didn't have to sweep our entire (large) codebase, and Google engineers developing code don't have to worry about leap seconds.

http://googleblog.blogspot.com/2011/09/time-technology-and-leaping-seconds.html

Leap-seconds 20 years from now

- Autonomous cars:
  - Bumper-bumper-trains on highways
  - ”driver” sleeps in the back-seat
    - Reality in Heathrow Airport today--->

- Power-grid regulation:
  - Cell-based micro-grids
  - Distributed Frequency/Voltage regulation
  - DC/AC converters without angular momentum
  - (”We don't care about timescales, we just use GPS receivers”)

- Other utilities:
  - Water, waste, gas, internet, cell phones...
  - No detected leap-second awareness as of yet.

INTERNATIONAL CONFERENCE

HELD AT WASHINGTON

FOR THE PURPOSE OF FIXING

A PRIME MERIDIAN

AND

A UNIVERSAL DAY.

OCTOBER, 1884.

Absent:

Chili: Mr. F. V. Gormas and Mr. A. B. Tupper.
Denmark: Mr. C. S. A. de Bille.
Liberia: Mr. Wm. Coppinger.
Netherlands: Mr. G. de Weckherlin.
Turkey: Rustem Effendi.

United States: Rear-Admiral C. R. P. Rodge
M. Rutherfurd, Mr. W. F. Allen, Com
Sampson, Professor Cleveland Abbe.
Venezuela: Dr. A. M. Soteldo.

Absent:

Denmark: Mr. C. S. A. de Bille.
Salvador: Mr. Antonio Batres.

Lov om Tidens Bestemmelse (* 1)

VI Christian den Niende, af Guds Naade

Stormarn, Ditmarsken, Lauenborg og Old

stadfæstet følgende Lov:

For all parts of the country with the exception of the Faroe Islands time shall hereafter be reckoned as the mean solar time at the 15th Longitude East of Greenwich.

§ 1

For alle Dele af Landet med Undtagelse af Færøerne skal Tiden herefter bestemmes lige med Middelsoltiden for den 15de Længdegrad Øst for Greenwich.

§ 2

Denne Lov træder i Kraft den 1ste Januar 1894.

Hvorefter alle vedkommende sig have at rette.

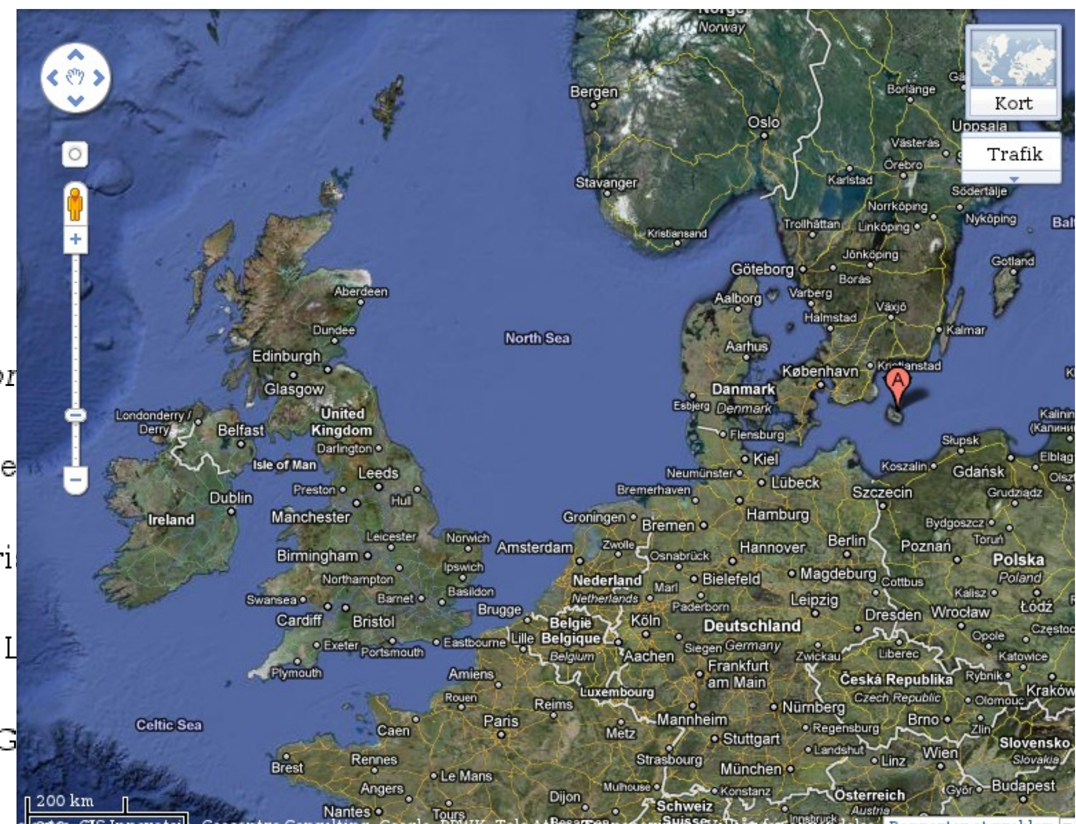
Givet paa Amalienbor

Under Vor Konge

Chri

(L

G



EU Directive 2000/84/EC (Daylight Savings Time)

Intention: DST synchronization throughout EU

Term used:	**Language:**
Greenwich time	EL, ET, HU, LV
Greenwich time (GMT)	SV
Greenwich mean time	EN, FI, LT, MT, SK
Universal time	ES, FR, IT, PT, RO
Universal time (GMT)	PL
World time	DE, NL
World time (GMT)	CS
World time (UTC)	DA
Universal coordinated time	SL

Who cares about Earth Orientation:

People who point things away from the planet
(Telescopes, Antennæ, rockets)
Rule of thumb: They have ”phd” after their names.

Who cares about Time ≡ Earth Orientation

(A minority of ?) Astronomers
~~People with sundials~~ (china = 1 timezone)
~~Navigators~~ (uses tables or software)

Who doesn't have a clue or care:

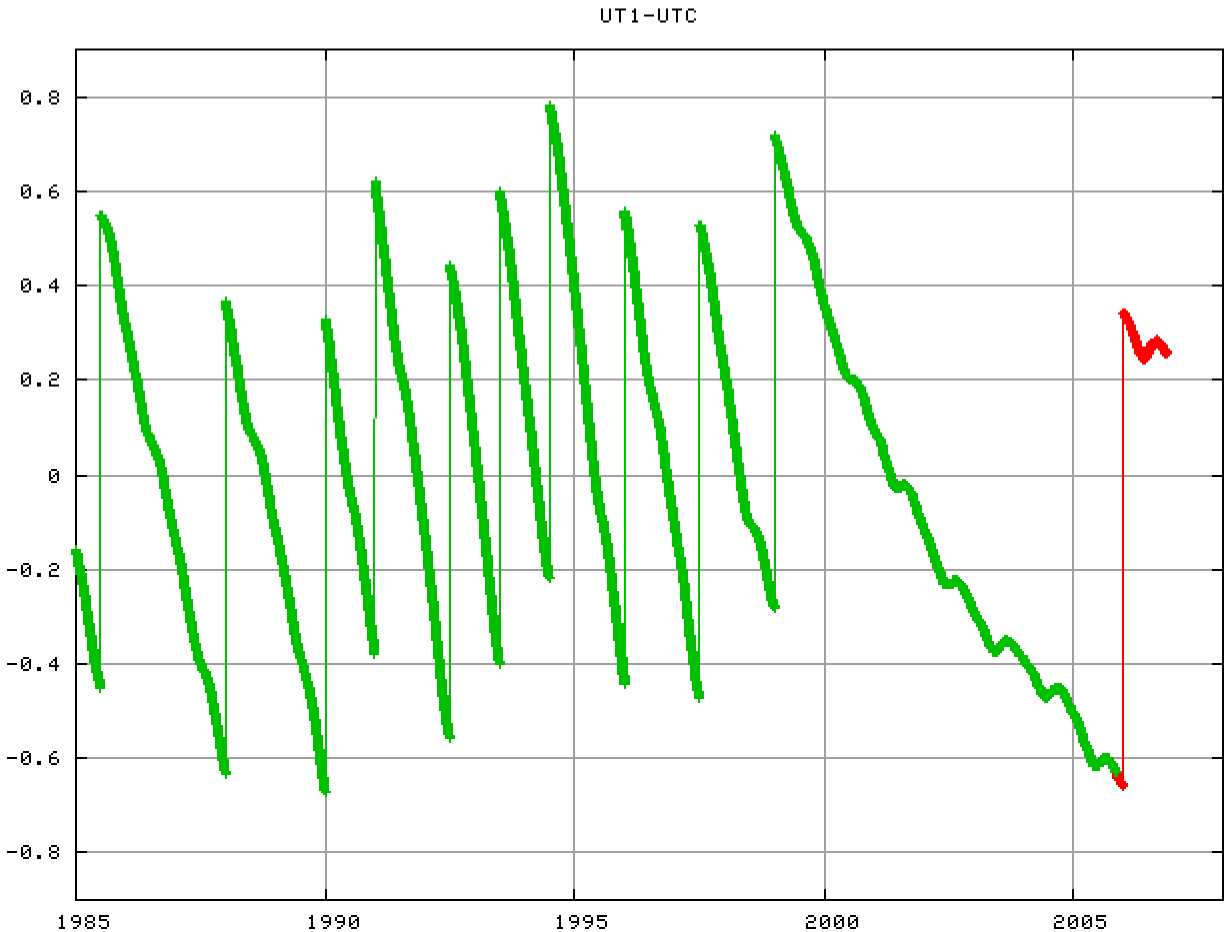
Everybody else (incl. 99.99% of programmers).

About programmers:

95% of all programmers think they are in the top 5%.

The rest are sure they are at least above average.

-- Linus Torvalds



”Dot-Com”

What can we do about leap seconds ?

Cheapest:

Discontinue Leap Seconds

Pro: UTC becomes POSIX time
Software ”just works” without changes.

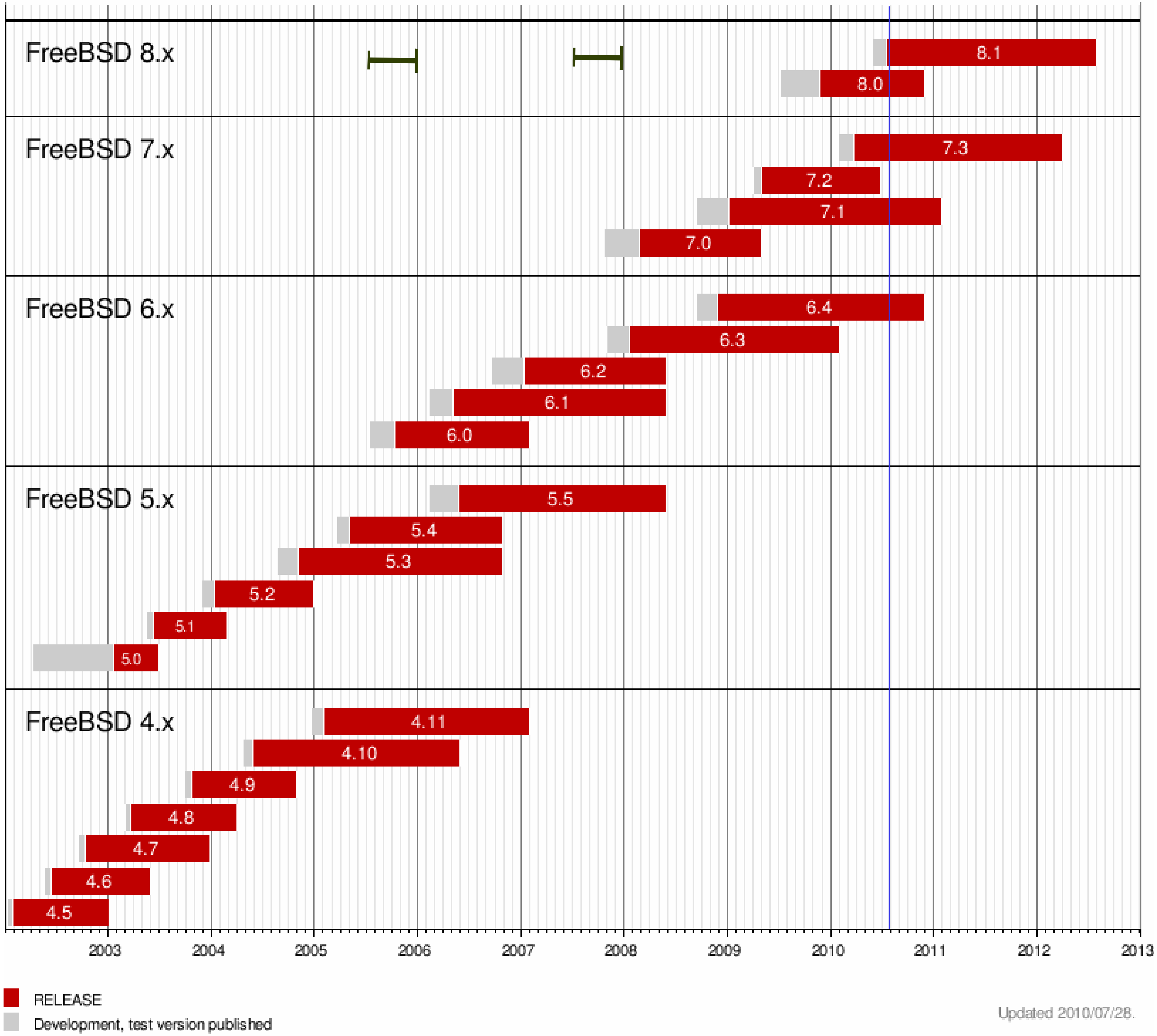
Con: ~~Civil timekeeping decoupled from sun~~
UTC not Earth orientation estimator (= DUT1 unbounded)
Time signals cannot represent DUT1

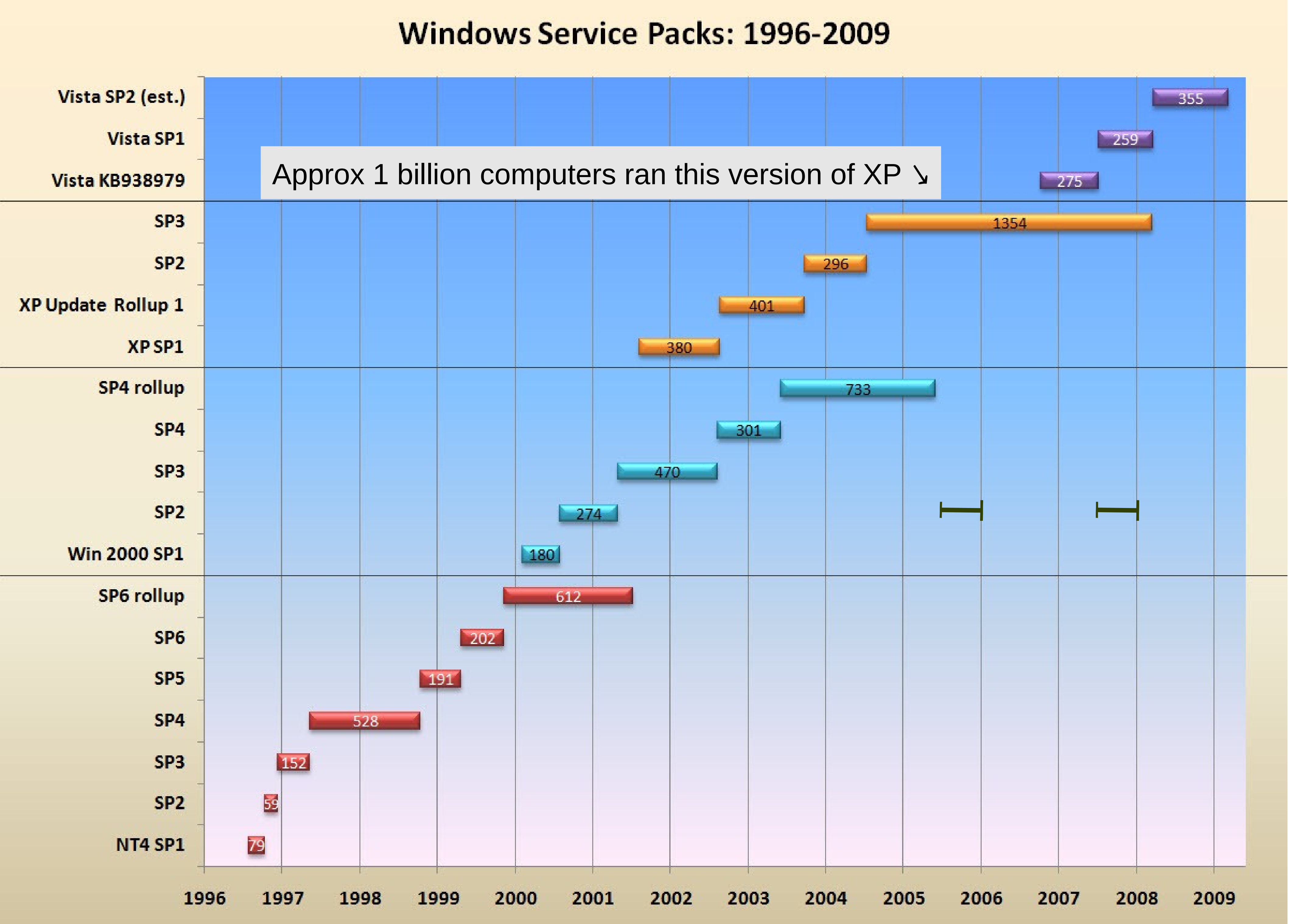
Most expensive:

Keep leap seconds unchanged

Pro: No change to UTC, DUT1 or time signals

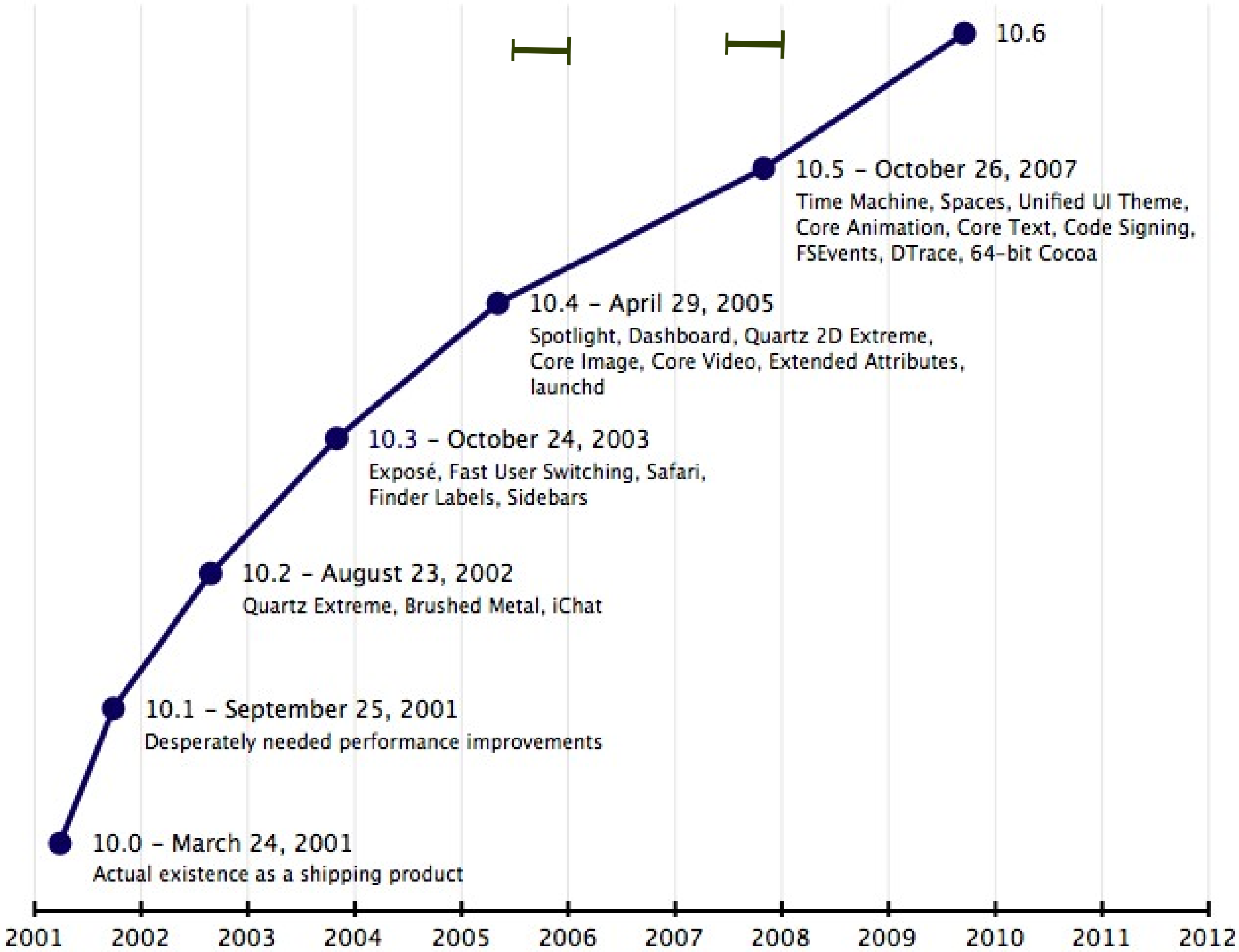
Con: Revision of ISO-C and POSIX required
High cost in software audit/development/test.
Cost avoided -> Software malfunction -> people hurt.


FreeBSD 8.x
8.1
8.0
FreeBSD 7.x
7.3
7.2
7.1
7.0
FreeBSD 6.x
6.4
6.3
6.2
6.1
6.0
FreeBSD 5.x
5.5
5.4
5.3
5.2
5.1
5.0
FreeBSD 4.x
4.11
4.10
4.9
4.8
4.7
4.6
4.5
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
RELEASE
Development, test version published
Updated 2010/07/28.



Windows Service Packs: 1996-2009
Approx 1 billion computers ran this version of XP ↘
Vista SP2 (est.)
Vista SP1
Vista KB938979
SP3
SP2
XP Update Rollup 1
XP SP1
SP4 rollup
SP4
SP3
SP2
Win 2000 SP1
SP6 rollup
SP6
SP5
SP4
SP3
SP2
NT4 SP1
355
259
275
1354
296
401
380
733
301
470
274
180
612
202
191
528
152
59
79
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009

Apple OSX



10.6
10.5 – October 26, 2007
Time Machine, Spaces, Unified UI Theme,
Core Animation, Core Text, Code Signing,
FSEvents, DTrace, 64-bit Cocoa
10.4 – April 29, 2005
Spotlight, Dashboard, Quartz 2D Extreme,
Core Image, Core Video, Extended Attributes,
launchd
10.3 – October 24, 2003
Exposé, Fast User Switching, Safari,
Finder Labels, Sidebars
10.2 – August 23, 2002
Quartz Extreme, Brushed Metal, iChat
10.1 – September 25, 2001
Desperately needed performance improvements
10.0 – March 24, 2001
Actual existence as a shipping product
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012

What can we do about leap seconds ?

Cheapest:

Discontinue Leap Seconds

Workable:

Leapseconds announced 10-20 years ahead of time

Pro: Makes leap-second handling an OS issue
Computers ”born” with leapsecond knowledge.
99.9+% of programmers taken out of the loop

Con: DUT1 < 1s not guaranteed (Past performance: DUT1 < 3s)
Time signal formats cannot represent DUT1
Some programs/systems/protocols will still fail

Most expensive:

Keep leap seconds unchanged