Open Source in Industry:
Trouble shooting of real-time Linux

Technical Heidelberg OSADL Talks, April 29, 2020, Online Session 3

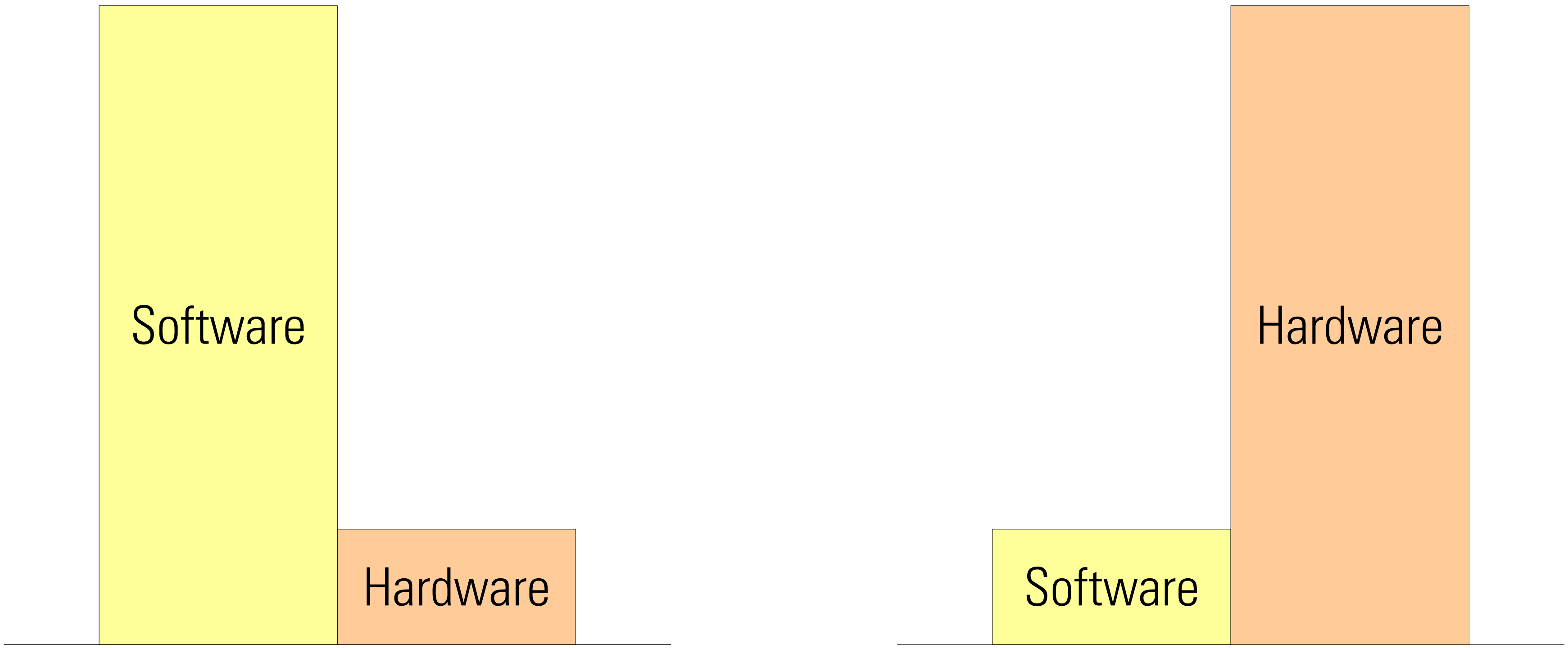
Determination of the real-time properties of a Linux system
Presentation of the OSADL QA Farm

Some information on today's sessions

- Please provide feedback on Legal HOT using the online form
  - Use the quick link **osadl.org/FB** (FeedBack), same as osadl.org/?id=3325
- You may ask questions during the session to be answered online, if possible
  - The quick link URL is **osadl.org/AQ** (AskQuestion), same as osadl.org/?id=3321
- You may join an online discussion on all topics of today at 4 pm
  - The quick link URL is **osadl.org/OD** (OnlineDiscussion), same as jitsi.osadl.org
  - Meeting name **OSADLTechnicalHOT**
  - Username and password will be displayed here after the last presentation

(We will show this slide again at the end of this session)

Issues leading to system latency



1979, e.g. Motorola
MC68000 @ 8 MHz
600 Dhrystones

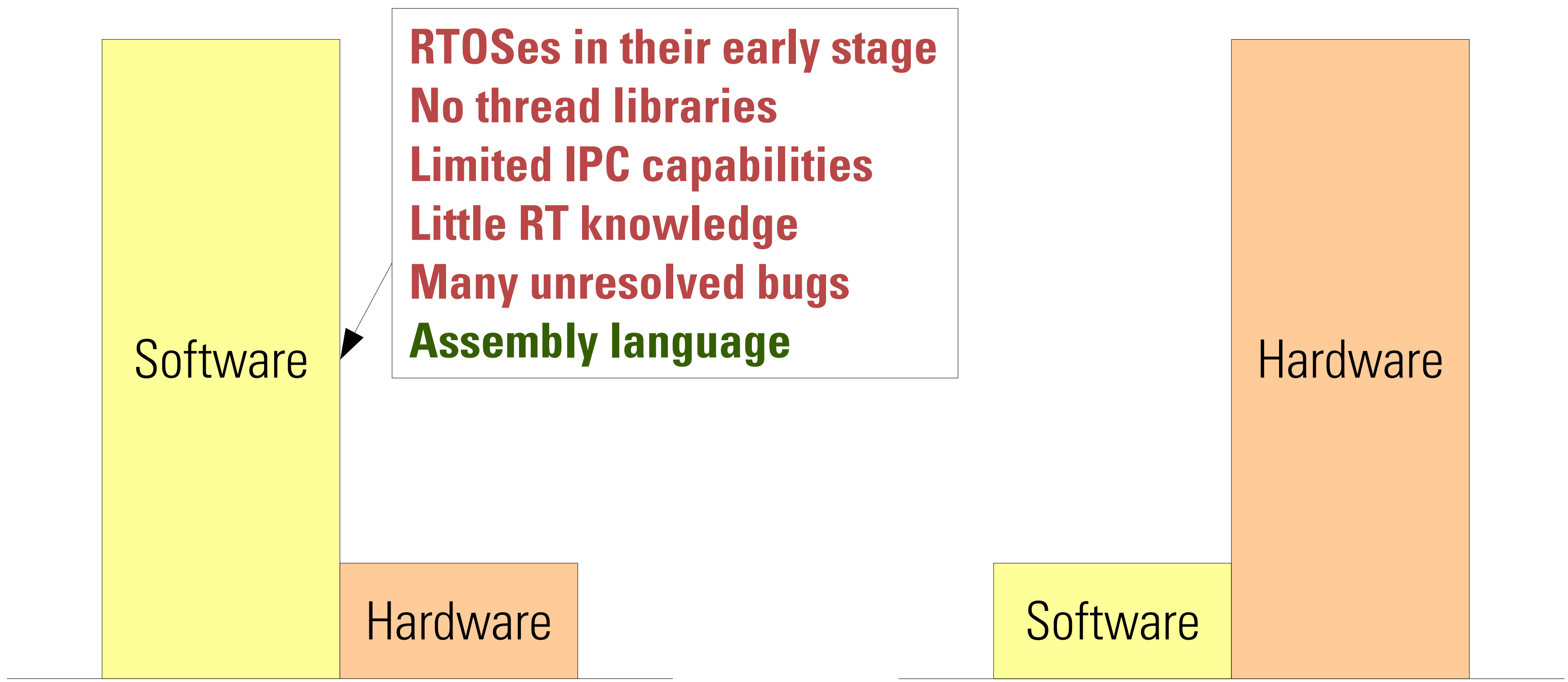
x 20,000

2009, e.g. Intel
Core 2 Duo @ 3 GHz
12,000,000 Dhrystones

Peak *vs.* worst-case performance

	1979	2009
Peak performance (e.g. Dhrystones)	600	12,000,000
Factor	1	20,000
Moore's Law [$2^{((2009-1979)/1.5)}$]	1	~1.048.576
Worst-case performance (e.g. signal latency)	~400 µs	20 µs
1/Factor	1	20

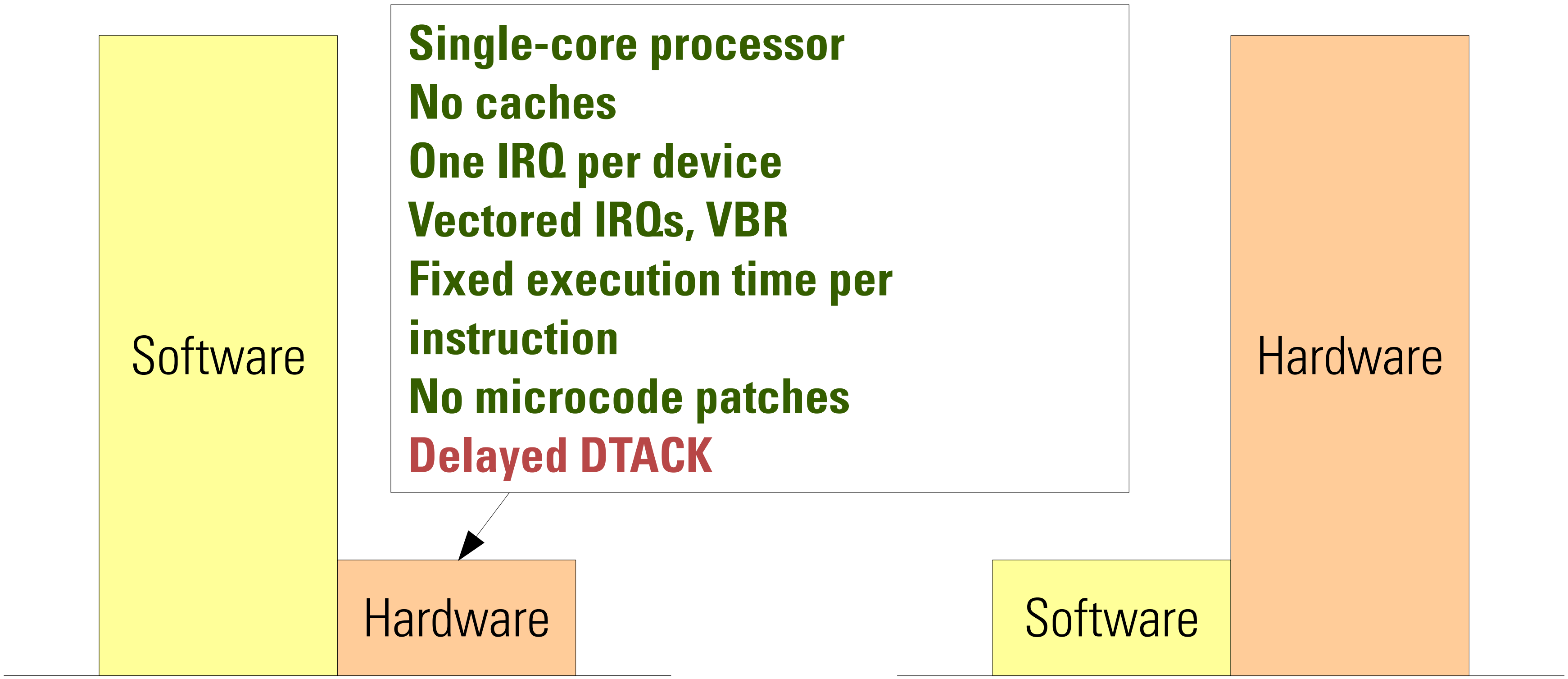
1979: Software issues related to system latency



1979, e.g. Motorola
MC68000 @ 8 MHz
600 Dhrystones

2009, e.g. Intel
Core 2 Duo @ 3 GHz
12,000,000 Dhrystones

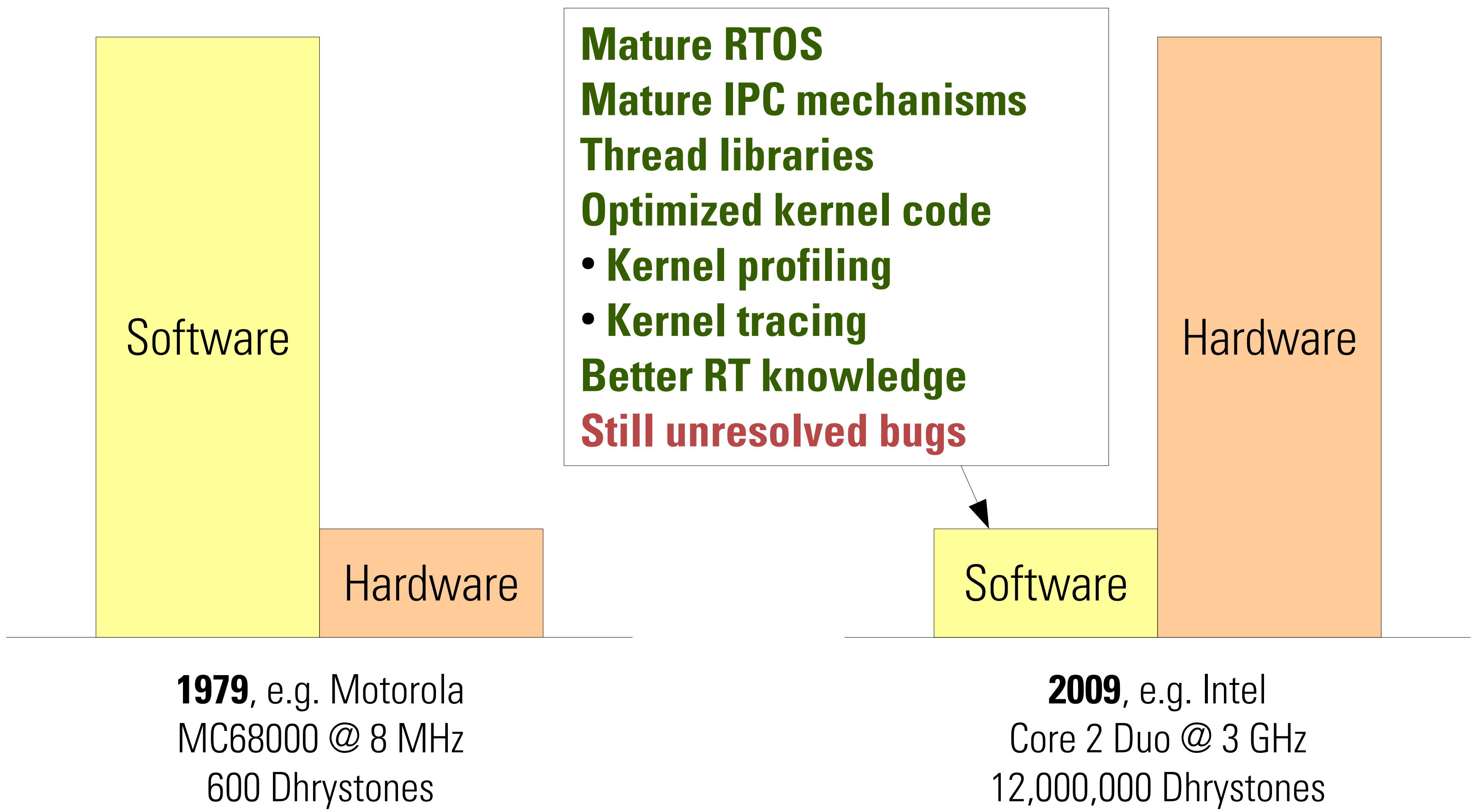
1979: Hardware issues related to system latency



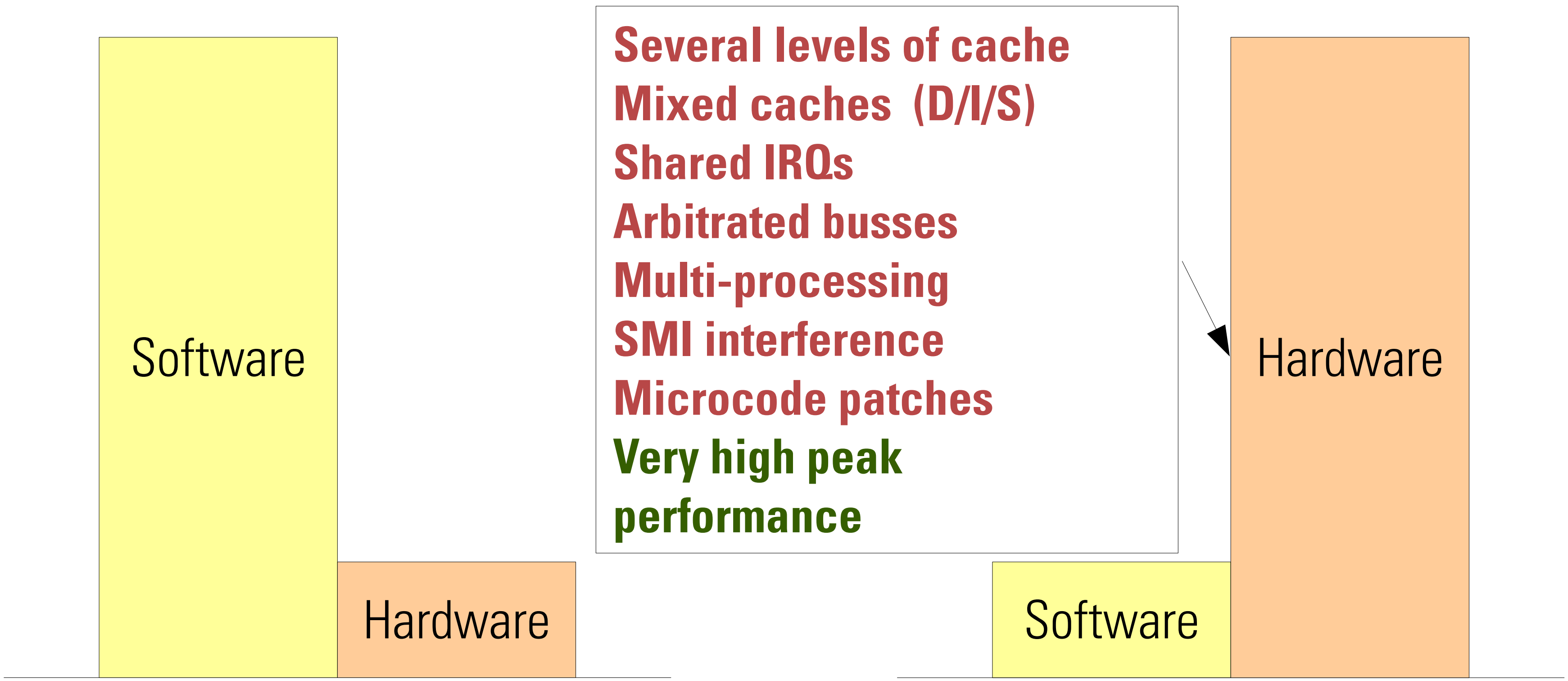
1979, e.g. Motorola
MC68000 @ 8 MHz
600 Dhrystones

2009, e.g. Intel
Core 2 Duo @ 3 GHz
12,000,000 Dhrystones

2009: Software issues related to system latency



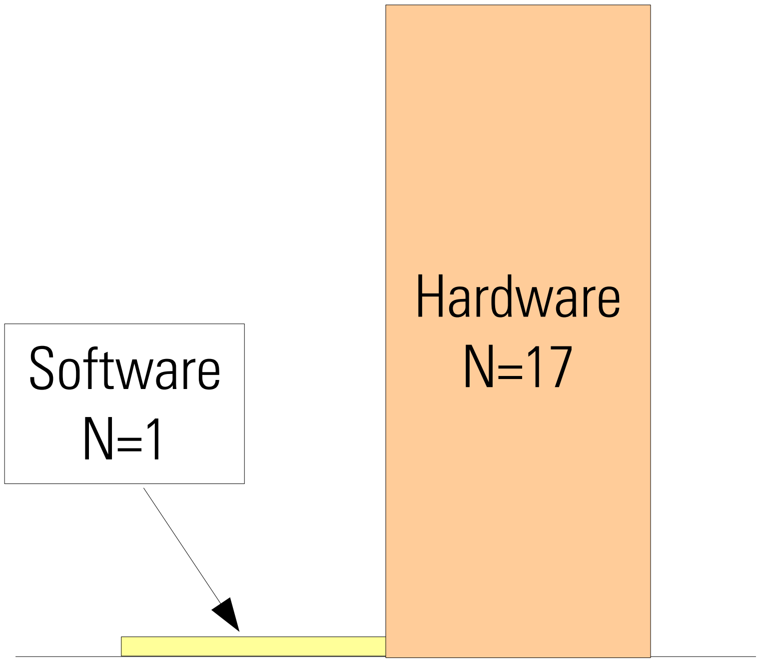
2009: Hardware issues related to system latency



1979, e.g. Motorola
MC68000 @ 8 MHz
600 Dhrystones

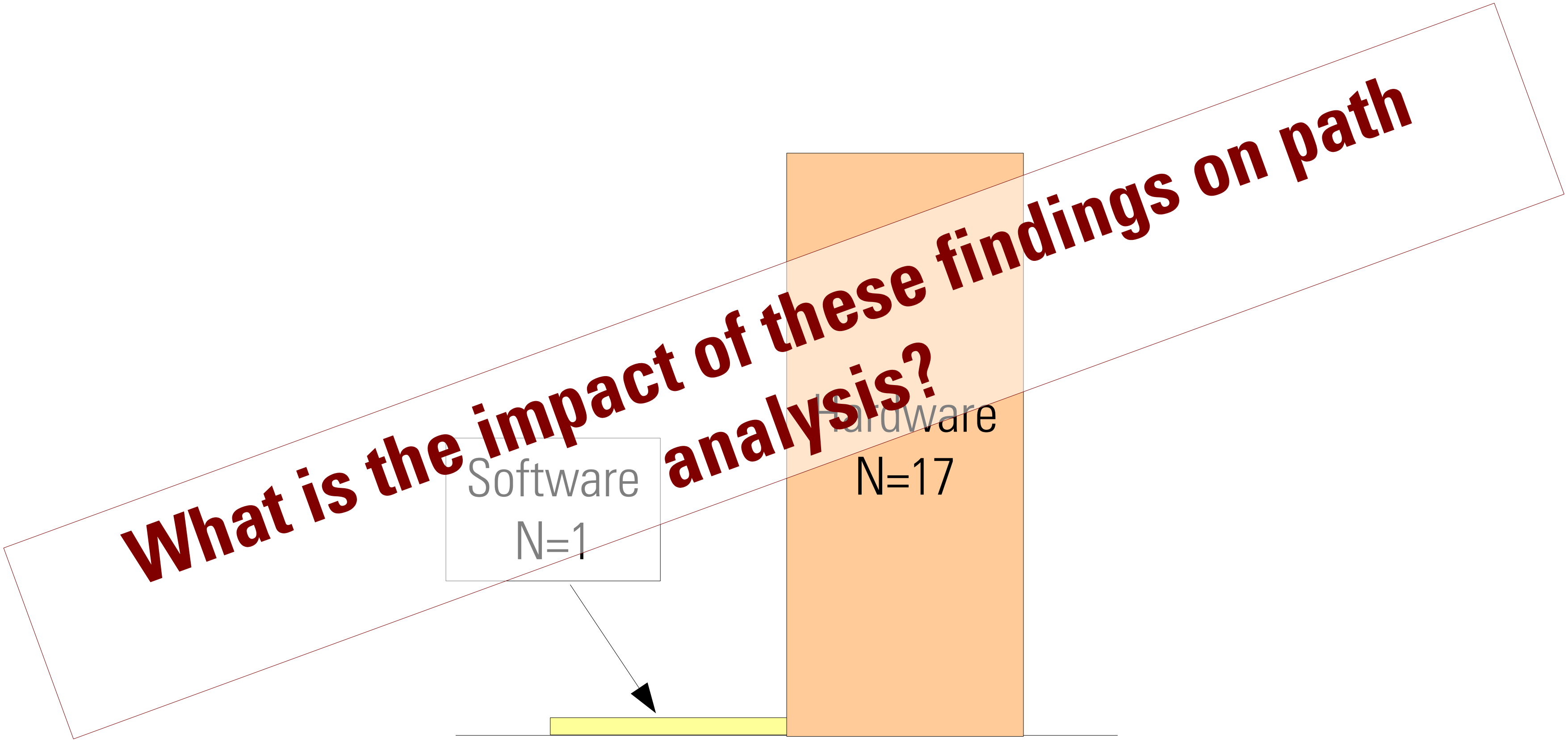
2009, e.g. Intel
Core 2 Duo @ 3 GHz
12,000,000 Dhrystones

latency-fighters@osadl.org



A total of 18 requests

latency-fighters@osadl.org



A total of 18 requests

Path analysis: 1979 *vs.* 2009

```
i = dram[0];
i++;
dram[0] = i;
```

```
movea.l   #dram,a0
move.l    (a0),d0
add.l     #1,d0
move.l    d0,(a0)
```

1979, e.g. Motorola
MC68000 @ 8 MHz
600 Dhrystones

```
mov    dram,eax
mov    eax,-4(ebp)
addl   $1,-4(ebp)
mov    -4(ebp),eax
mov    eax,dram
```

2009, e.g. Intel
Core 2 Duo @ 3 GHz
12,000,000 Dhrystones

Path analysis: 1979 *vs.* 2009

1979

```
movea.l   #dram,a0
move.l    (a0),d0
add.l     #1,d0
move.l    d0,(a0)
```

Load instruction from memory and execute it. Duration = **56** clock cycles

```
mov     dram,eax
mov     eax,-4(ebp)
addl    $1,-4(ebp)
mov     -4(ebp),eax
mov     eax,dram
```

1979, e.g. Motorola MC68000 @ 8 MHz
600 Dhrystones

2009, e.g. Intel Core 2 Duo @ 3 GHz
12,000,000 Dhrystones

Path analysis: 1979 *vs.* 2009

2009

```
movea.l   #dram,a0
move.l    (a0),d0
add.l     #1,d0
move.l    d0,(a0)
```

1979, e.g. Motorola
MC68000 @ 8 MHz
600 Dhrystones

Load instruction from cache and execute it.
Duration = **?**

```
mov     dram,eax
mov     eax,-4(ebp)
addl    $1,-4(ebp)
mov     -4(ebp),eax
mov     eax,dram
```

Instruction not in cache/no free cache lines

Data not in cache/no free cache lines

System Management Interrupt

Instruction may be emulated (microcode patch)

2009, e.g. Intel
Core 2 Duo @ 3 GHz
12,000,000 Dhrystones

Path analysis

Path analysis

- Generally accepted verification procedure
- Source code normally required
- Difficult to do in modern high-performance processors
- Required processor data often not disclosed
- Expensive procedure
- Normally not done by users
- Result of path analysis often not publicly available
- May need to be checked against empirical latency testing

Path analysis *vs.* latency testing

Path analysis

- Generally accepted verification procedure
- Source code normally required
- Difficult to do in modern high-performance processors
- Required processor data often not disclosed
- Expensive procedure
- Normally not done by users
- Result of path analysis often not publicly available
- May need to be checked against empirical latency testing

Latency testing

- Not considered a valid “verification”
- Source code not required
- System complexity irrelevant
- Easy procedure
- Can be done by everybody

Path analysis *vs.* latency testing

Path analysis

- Generally accepted verification procedure
- Source code normally required
- Difficult to do in modern high-performance processors
- Required processor data often not disclosed
- Expensive procedure
- Normally not done by users
- Result of path analysis often not publicly available
- May need to be checked against empirical latency testing

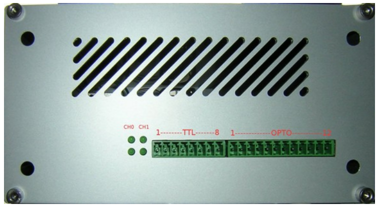
Latency testing

- Not considered a valid “verification”
- Source code not required
- System complexity irrelevant
- Easy procedure
- Can be done by everybody

Let's do it!

Four levels of latency tests

External measurement with simulation
OSADL's „Latency-Box“



Internal latency recording
Built-in kernel latency histograms

```
CONFIG_WAKEUP_LATENCY_HIST=y
CONFIG_INTERRUPT_OFF_HIST=y
CONFIG_SWITCHTIME_HIST=y
```

Internal measurement with simulation
Cyclictest

```
# cyclictest -a -t -n -p99
```

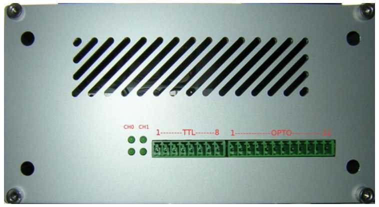
Real-world internal measurement
Application

```
# <application>
```

Four levels of latency tests

External measurement with simulation

OSADL's „Latency-Box“



Internal latency recording

Built-in kernel latency histograms

```
CONFIG_WAKEUP_LATENCY_HIST=y
CONFIG_INTERRUPT_OFF_HIST=y
CONFIG_SWITCHTIME_HIST=y
```

Internal measurement with simulation

Cyclictest

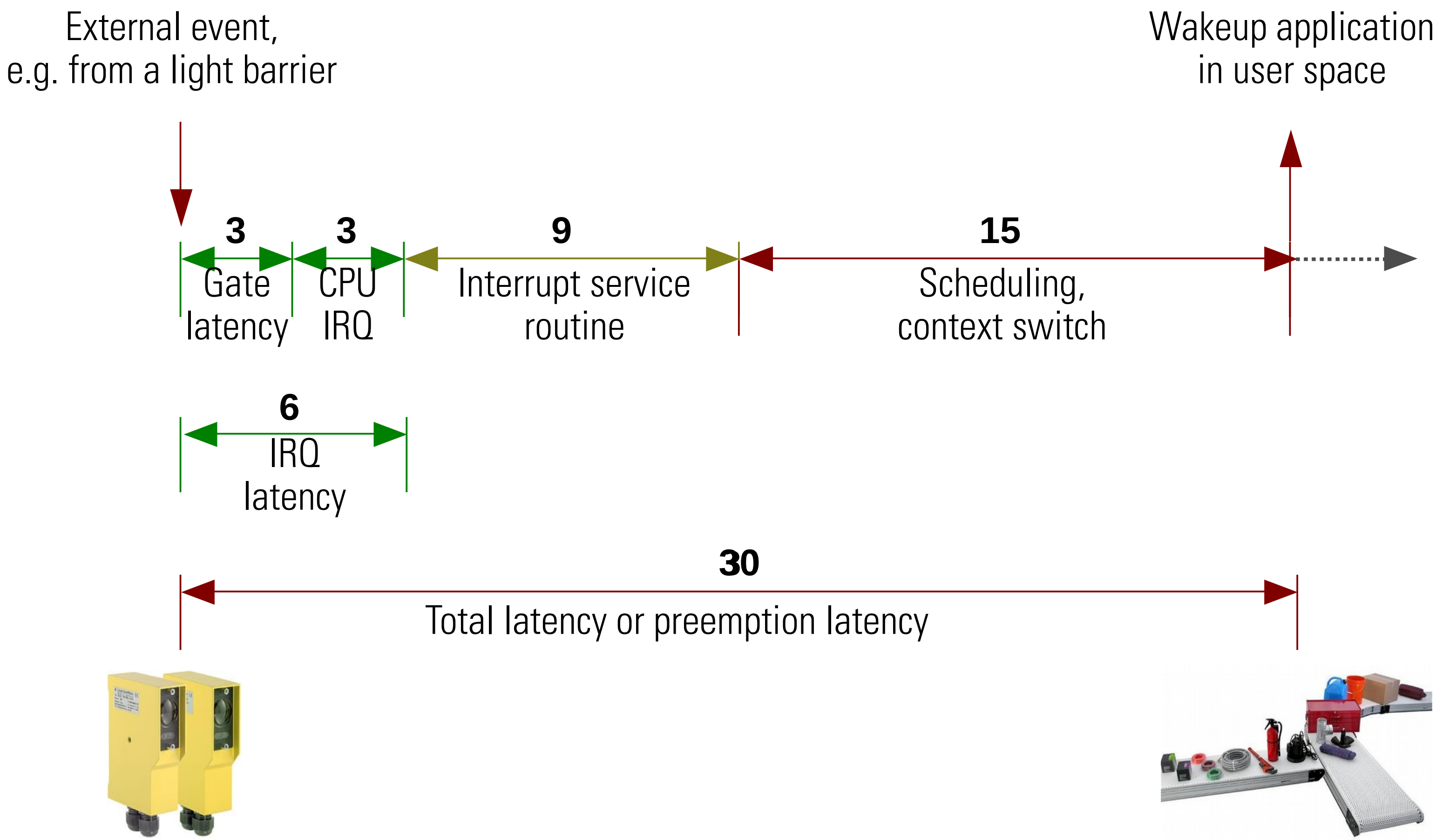
```
# cyclictest -a -t -n -p99
```

Real-world internal measurement

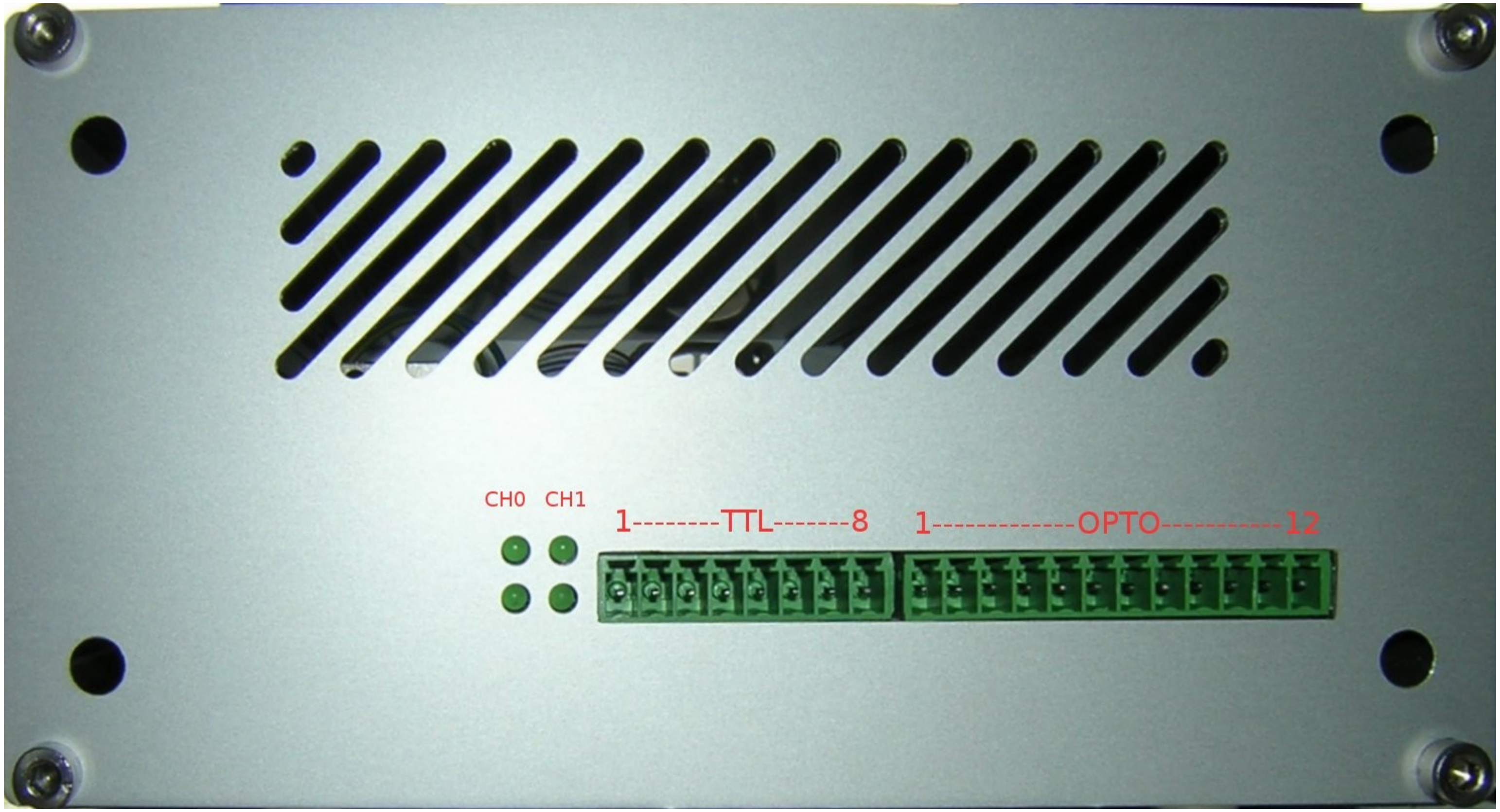
Application

```
# <application>
```

Signal path to be monitored

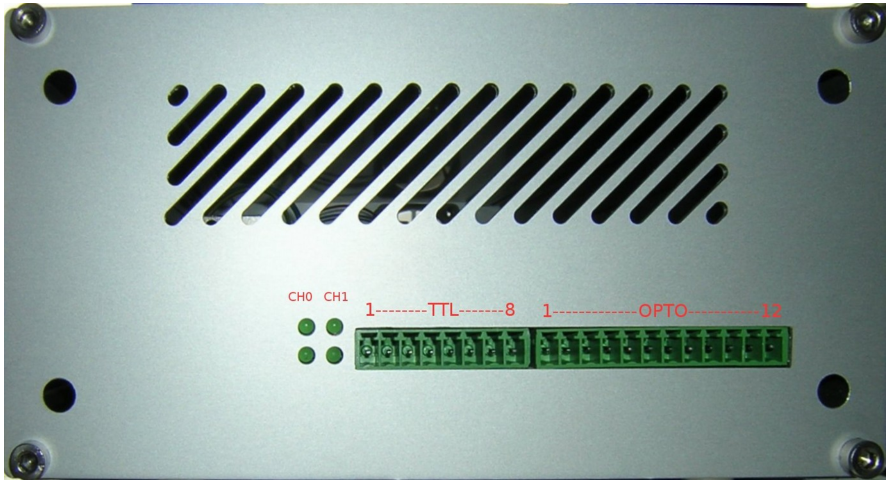


OSADL's „Latency Box“



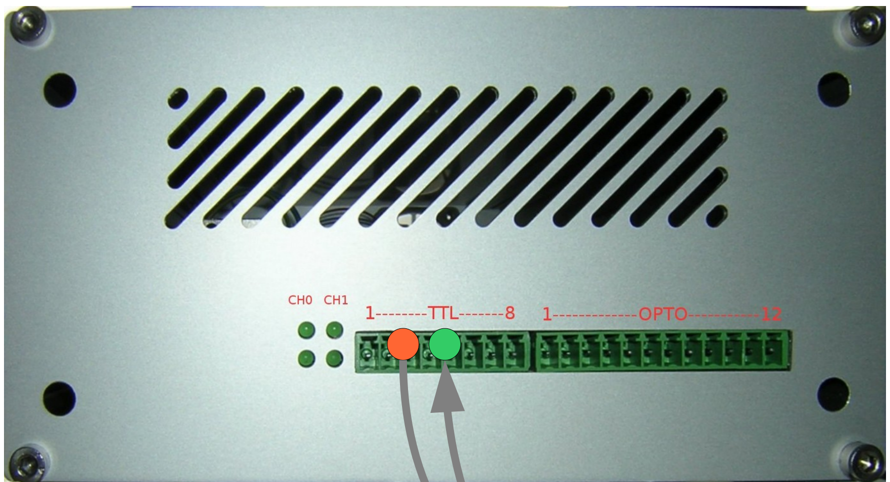
ELTEC systems

OSADL's „Latency Box“ - Specification

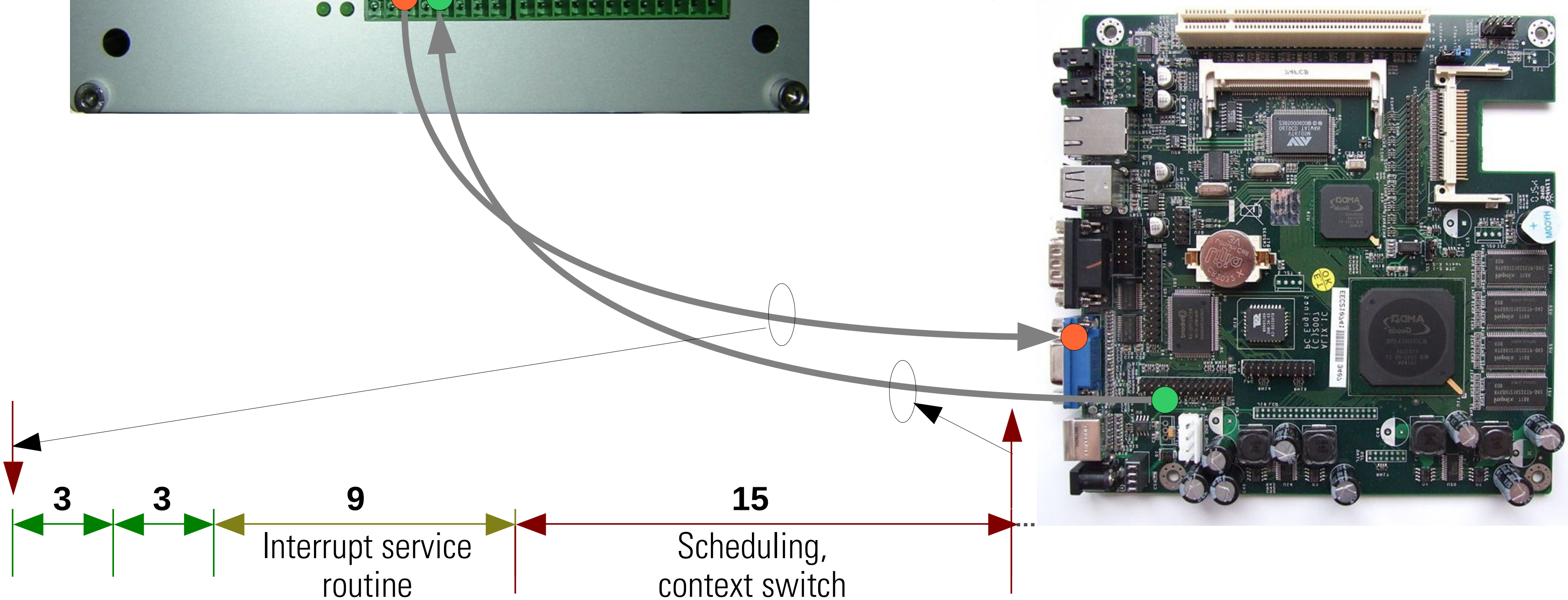


PowerPC 750FX@600MHz
64 MB SDRAM on SODIMM, 16 MB Flash-EPROM
10/100 Mb/s Network
2 serial channels RS232 and RS485
2 TTL Outputs, 4 TTL Inputs
4 Status LEDs
On-board FPGA

OSADL's „Latency Box“ connected to a CPU board



PowerPC 750FX@600MHz
64 MB SDRAM on SODIMM, 16 MB Flash-EPROM
10/100 Mb/s Network
2 serial channels RS232 and RS485
2 TTL Outputs, 4 TTL Inputs
4 Status LEDs
On-board FPGA



OSADL's „Latency Box“ data transfer

Histogram data

Line #1 `0` *(No latency recording below 1 µs duration)*

```
0
0
0
0
0
0
0
0
0
```

Line #11 `76` *(A total of 76 latency values between 10 and 11 µs duration)*

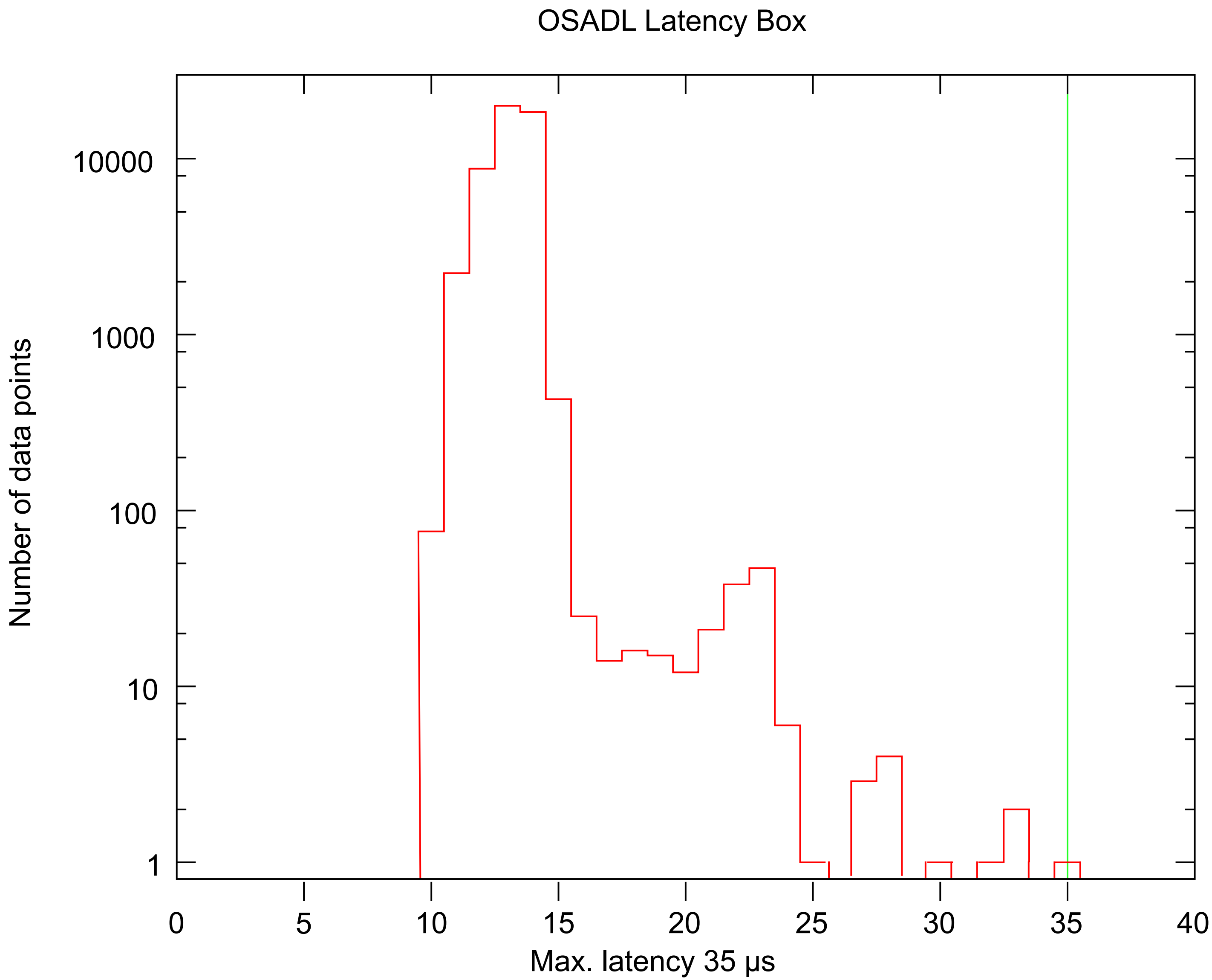
```
2238
8800
```

`20027` *(Most frequently observer latency values between 13 and 14 µs duration)*

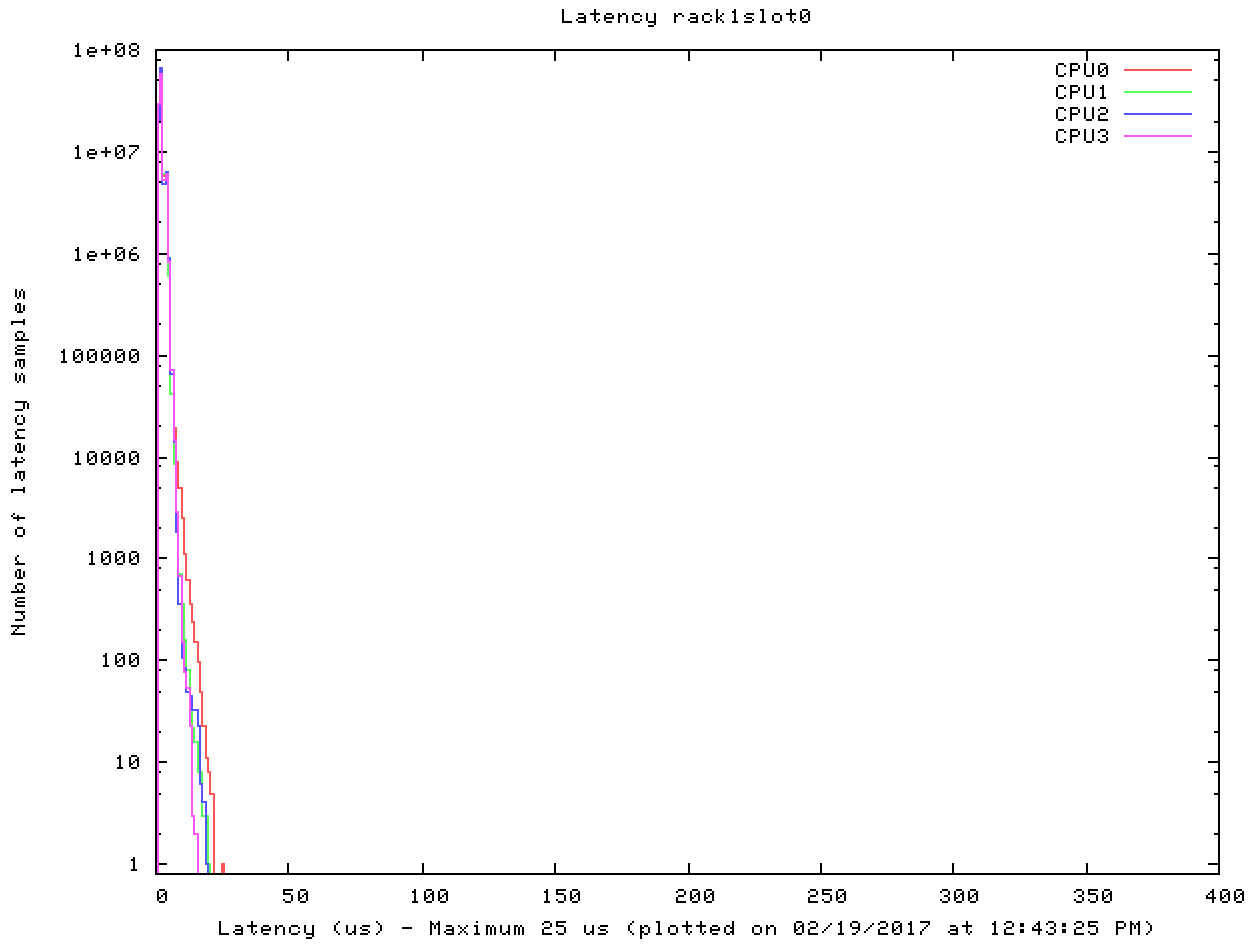
```
18433
430
25
14
[..]
```

Line #1000 `0` *(No overflow)*

OSADL's „Latency Box“ - data plot



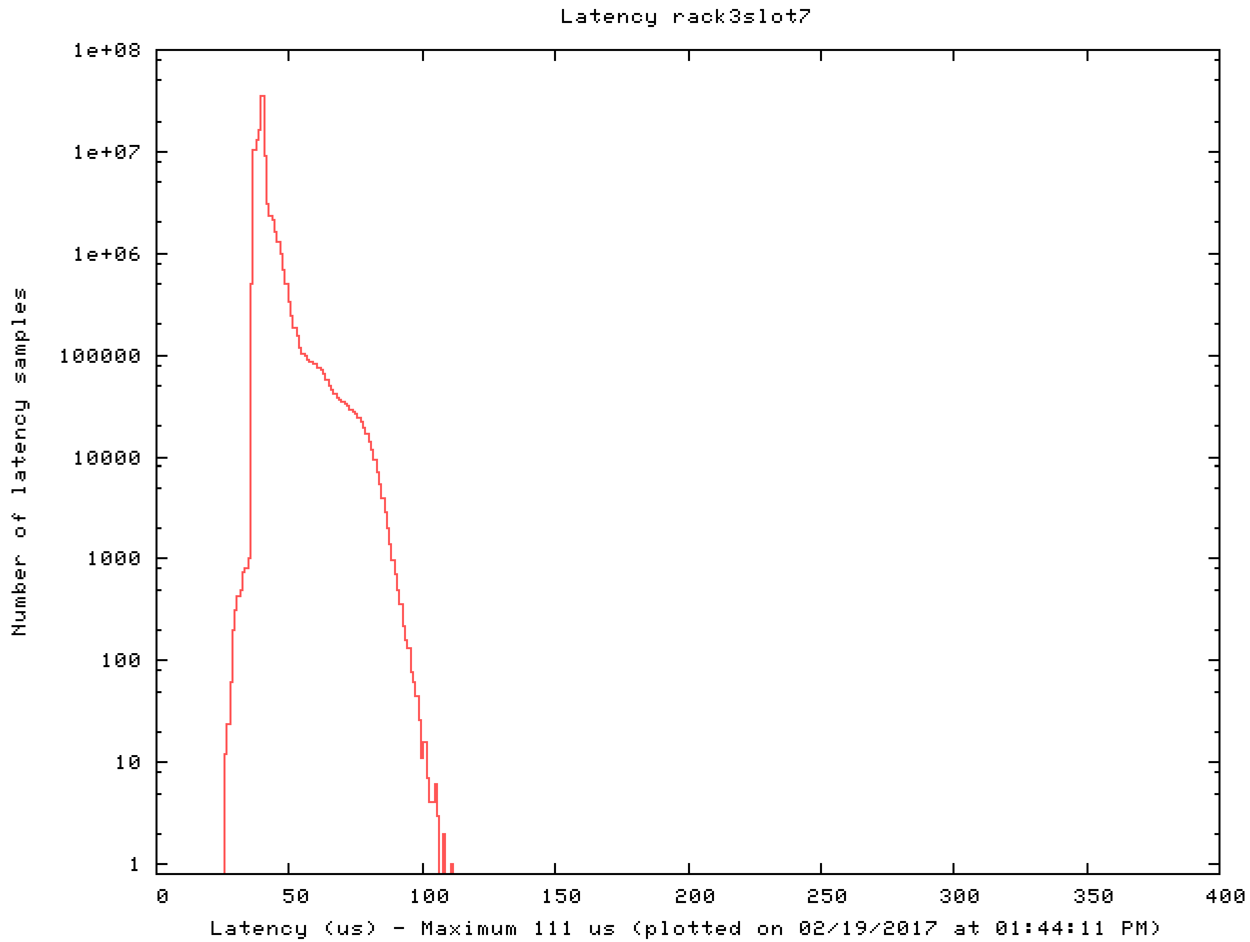
OSADL standard "latency plot" (RT system)



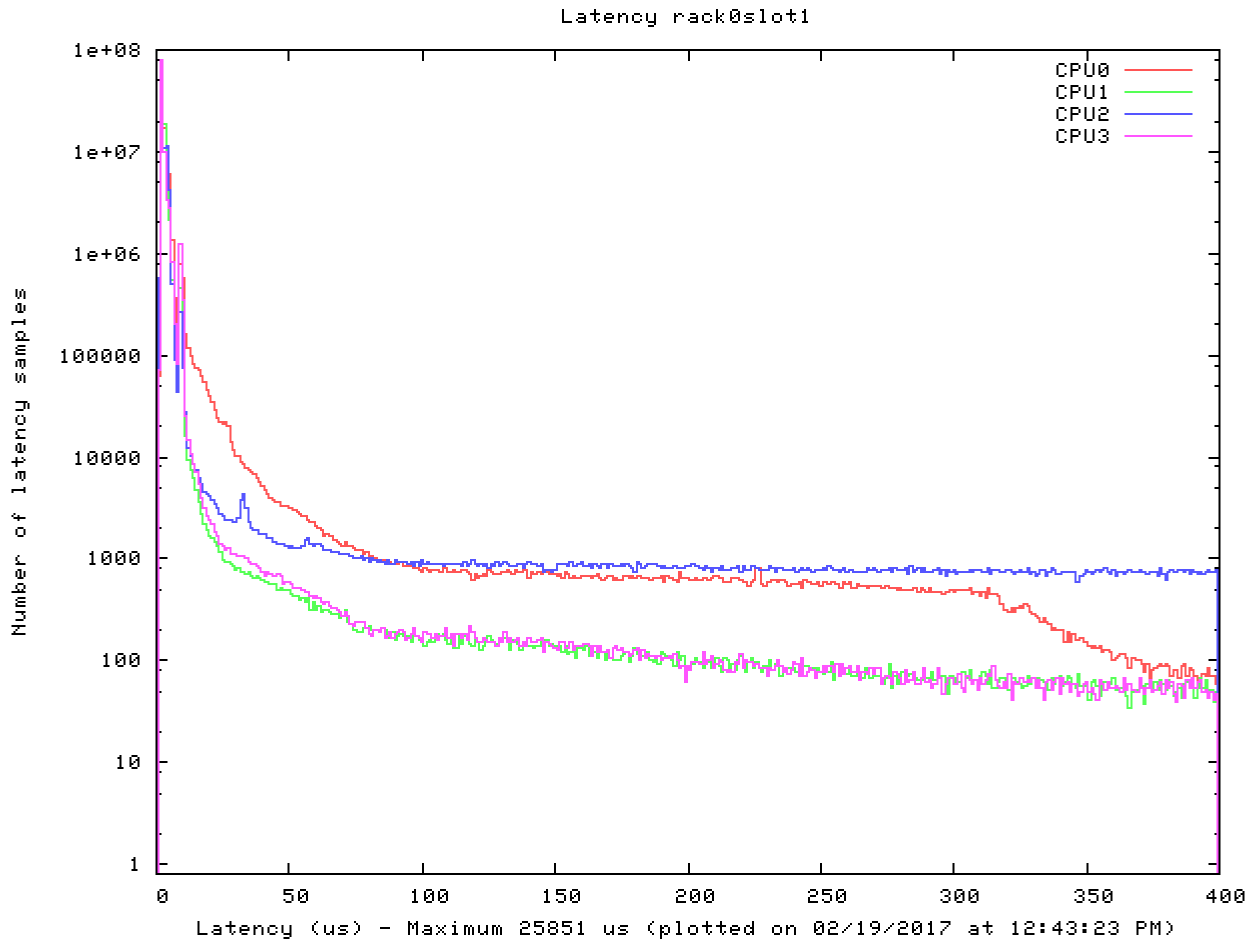
HOT HEIDELBERG OSADL TALKS 2020

OSADL
Open Source Automation Development Lab eG

OSADL standard “latency plot” (slow RT system)



OSADL standard “latency plot” (non-RT system)



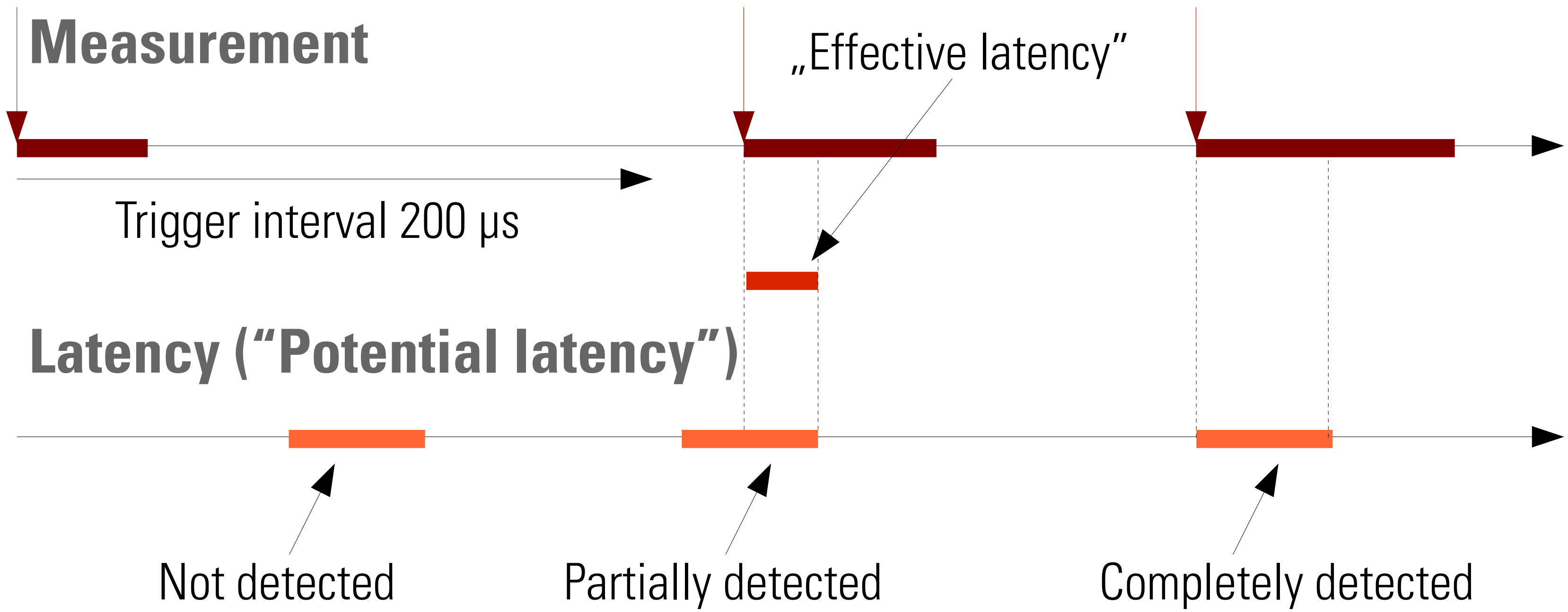
Test: „Potential latency“ *vs.* „Effective latency“

Find appropriate measurement parameters

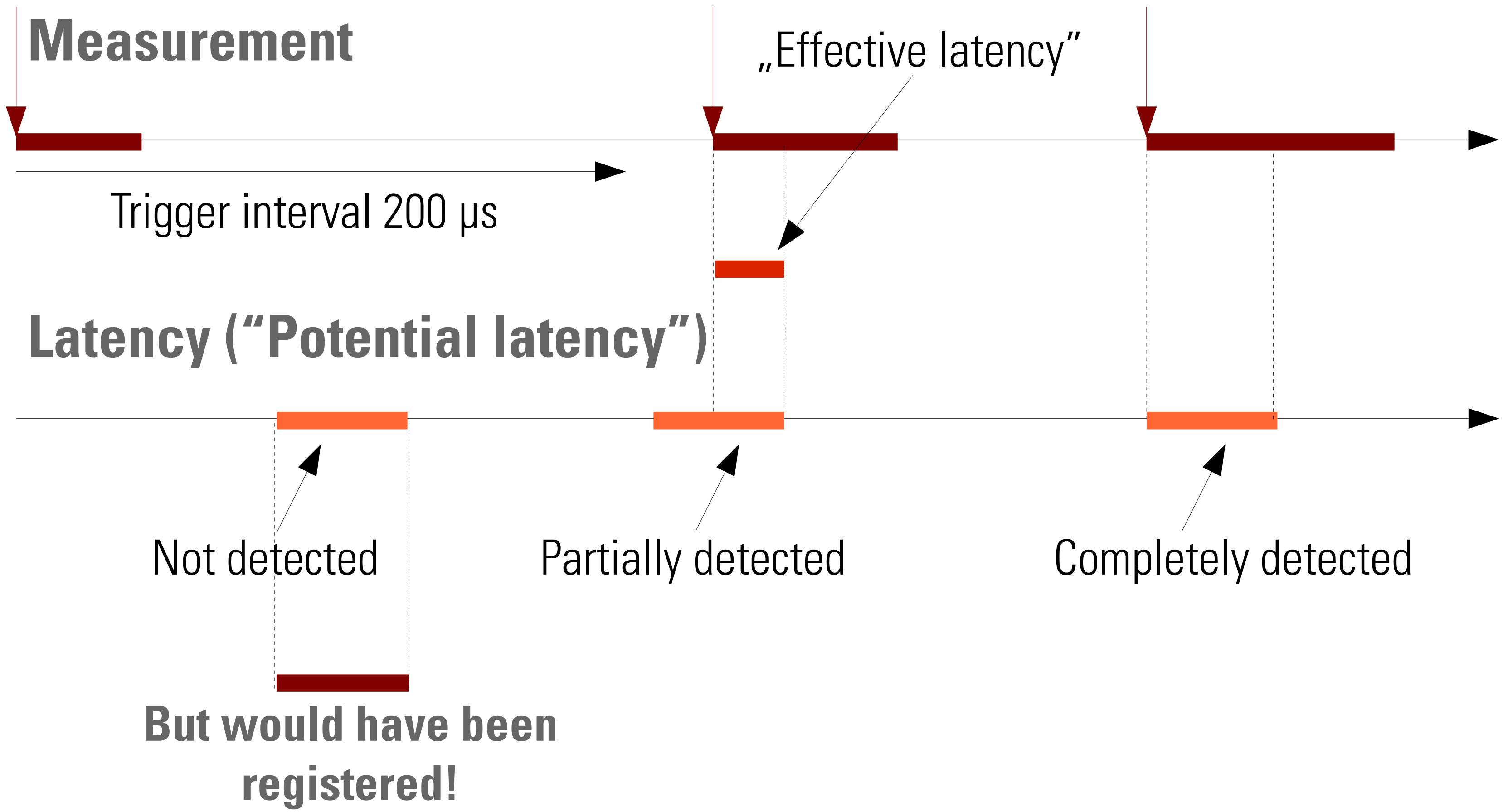
```
# cyclictest -m -n -Sp90 -i100 -d0
# /dev/cpu_dma_latency set to 0us
policy: fifo: loadavg: 10.43 6.56 3.38 2/1454 4126098

T: 0 (4122431) P:99 I:100 C:5154828 Min:      3 Act:    4 Avg:    6 Max:      42
T: 1 (4122432) P:99 I:100 C:5154687 Min:      3 Act:    4 Avg:    5 Max:      88
T: 2 (4122433) P:99 I:100 C:5154561 Min:      3 Act:    4 Avg:    5 Max:      40
T: 3 (4122434) P:99 I:100 C:5154439 Min:      3 Act:    7 Avg:    6 Max:      40
T: 4 (4122435) P:99 I:100 C:5154318 Min:      3 Act:    4 Avg:    6 Max:      31
T: 5 (4122436) P:99 I:100 C:5154196 Min:      3 Act:    5 Avg:    5 Max:      47
T: 6 (4122437) P:99 I:100 C:5153993 Min:      3 Act:    4 Avg:    6 Max:      41
T: 7 (4122438) P:99 I:100 C:5153936 Min:      3 Act:    4 Avg:    5 Max:      94
T: 8 (4122439) P:99 I:100 C:5153807 Min:      3 Act:    4 Avg:    5 Max:      39
T: 9 (4122440) P:99 I:100 C:5153662 Min:      3 Act:    5 Avg:    5 Max:      51
T:10 (4122441) P:99 I:100 C:5153517 Min:      3 Act:    5 Avg:    5 Max:      42
T:11 (4122442) P:99 I:100 C:5153371 Min:      3 Act:    4 Avg:    5 Max:      30
```

„Potential latency“ *vs.* „Effective latency“

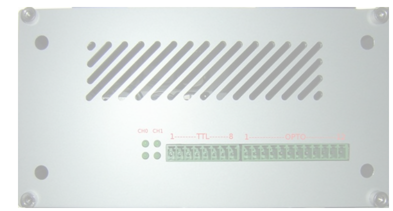


„Potential latency“ *vs.* „Effective latency“



Four levels of latency tests

External measurement with simulation
OSADL's „Latency-Box“



Internal latency recording
Built-in kernel latency histograms

```
CONFIG_WAKEUP_LATENCY_HIST=y
CONFIG_INTERRUPT_OFF_HIST=y
CONFIG_SWITCHTIME_HIST=y
```

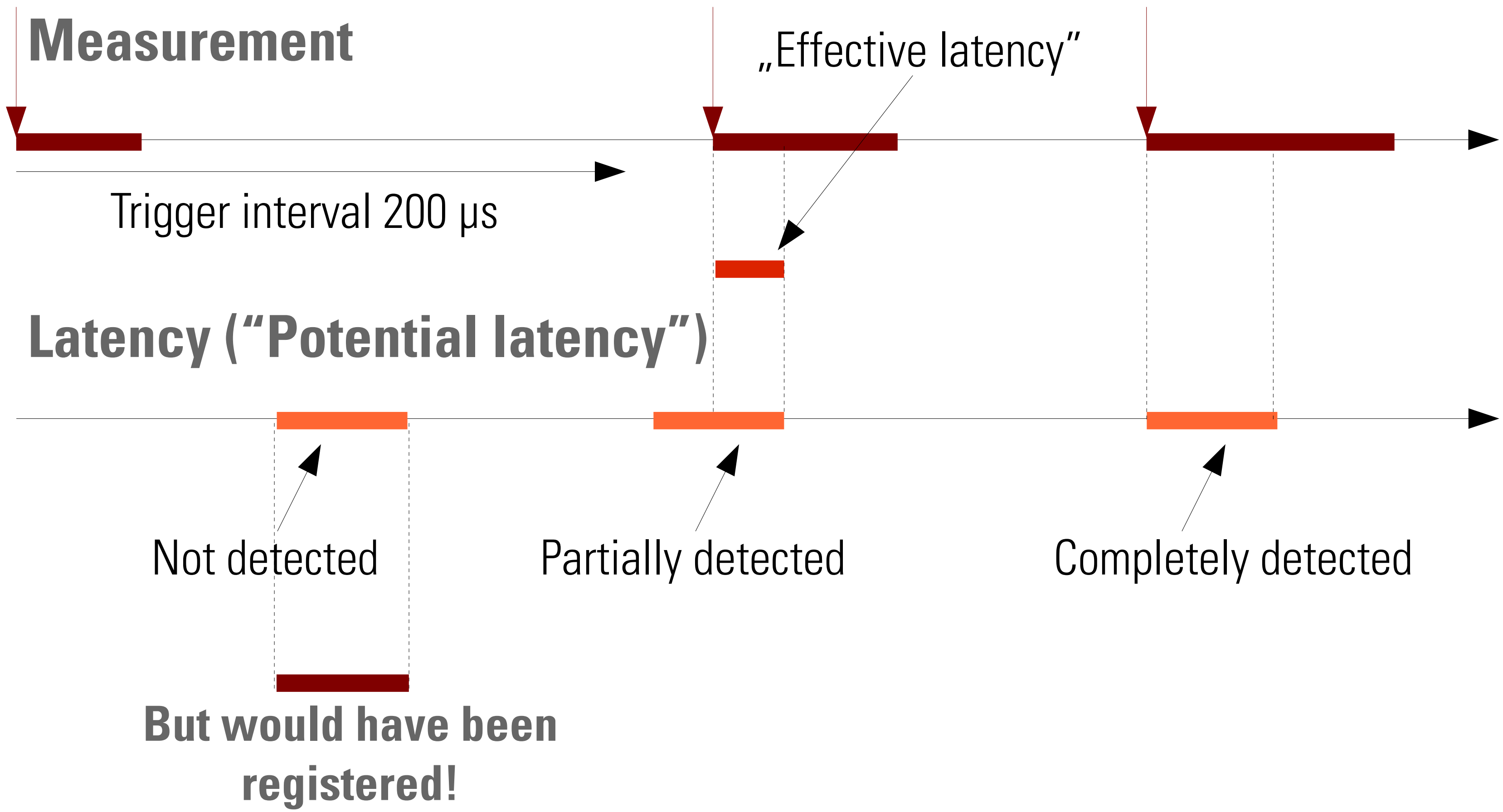
Internal measurement with simulation
Cyclictest

```
# cyclictest -a -t -n -p99
```

Real-world internal measurement
Application

```
# <application>
```

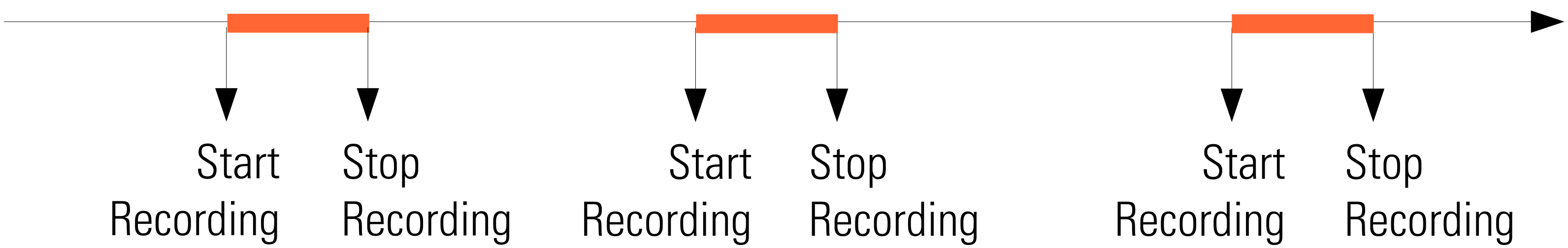
„Potential latency“ *vs.* „Effective latency“



Internal recording of potential latencies

- Preemption off
- Interrupts off
- Preemption and interrupts off

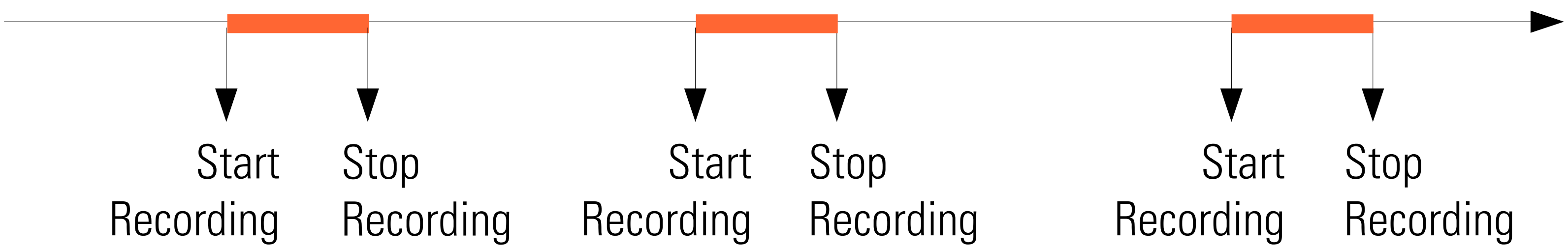
Duration of critical section



Internal recording of effective latencies

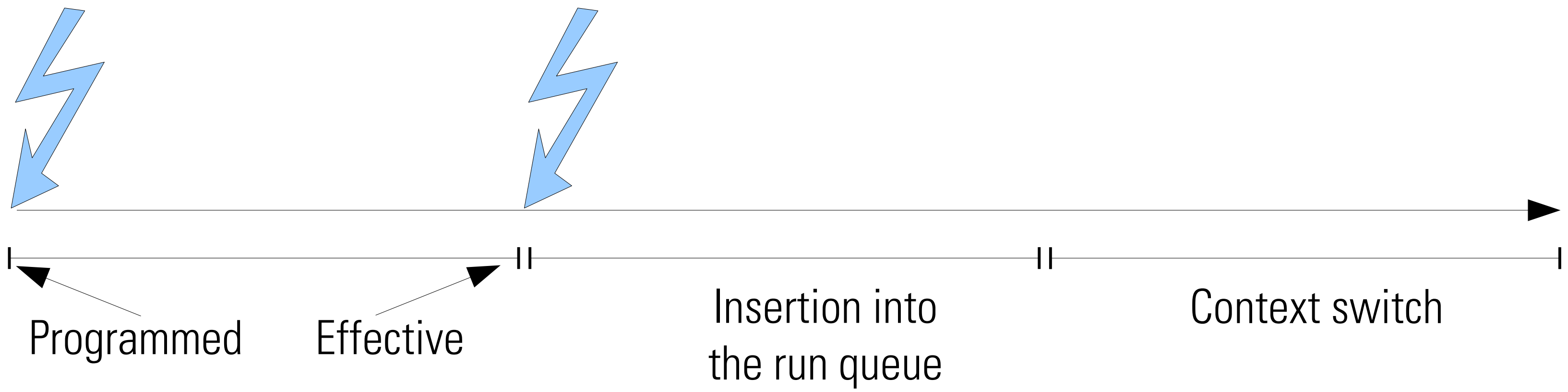
- Wakeup time
- Context switch

Recording of execution time



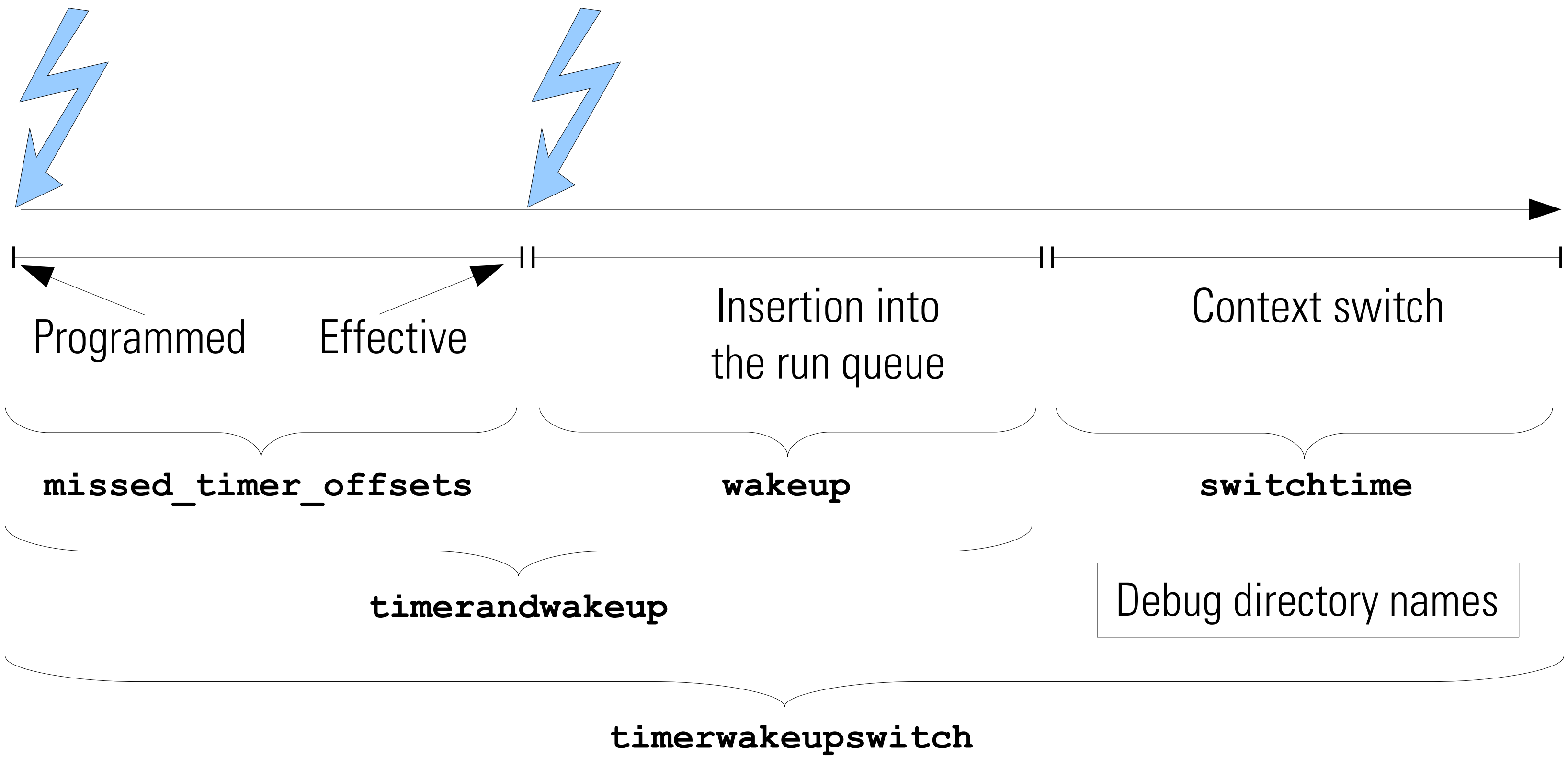
Internal recording of effective latencies, sections

Restarting a waiting application by timer expiration



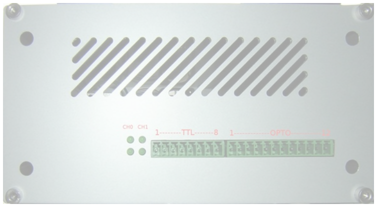
Internal recording of effective latencies, variables

Restarting a waiting application by timer expiration



Four levels of latency tests

External measurement with simulation
OSADL's „Latency-Box“



Internal continuous recording
Built-in kernel latency histograms

```
CONFIG_WAKEUP_LATENCY_HIST=y
CONFIG_INTERRUPT_OFF_HIST=y
CONFIG_SWITCHTIME_HIST=y
```

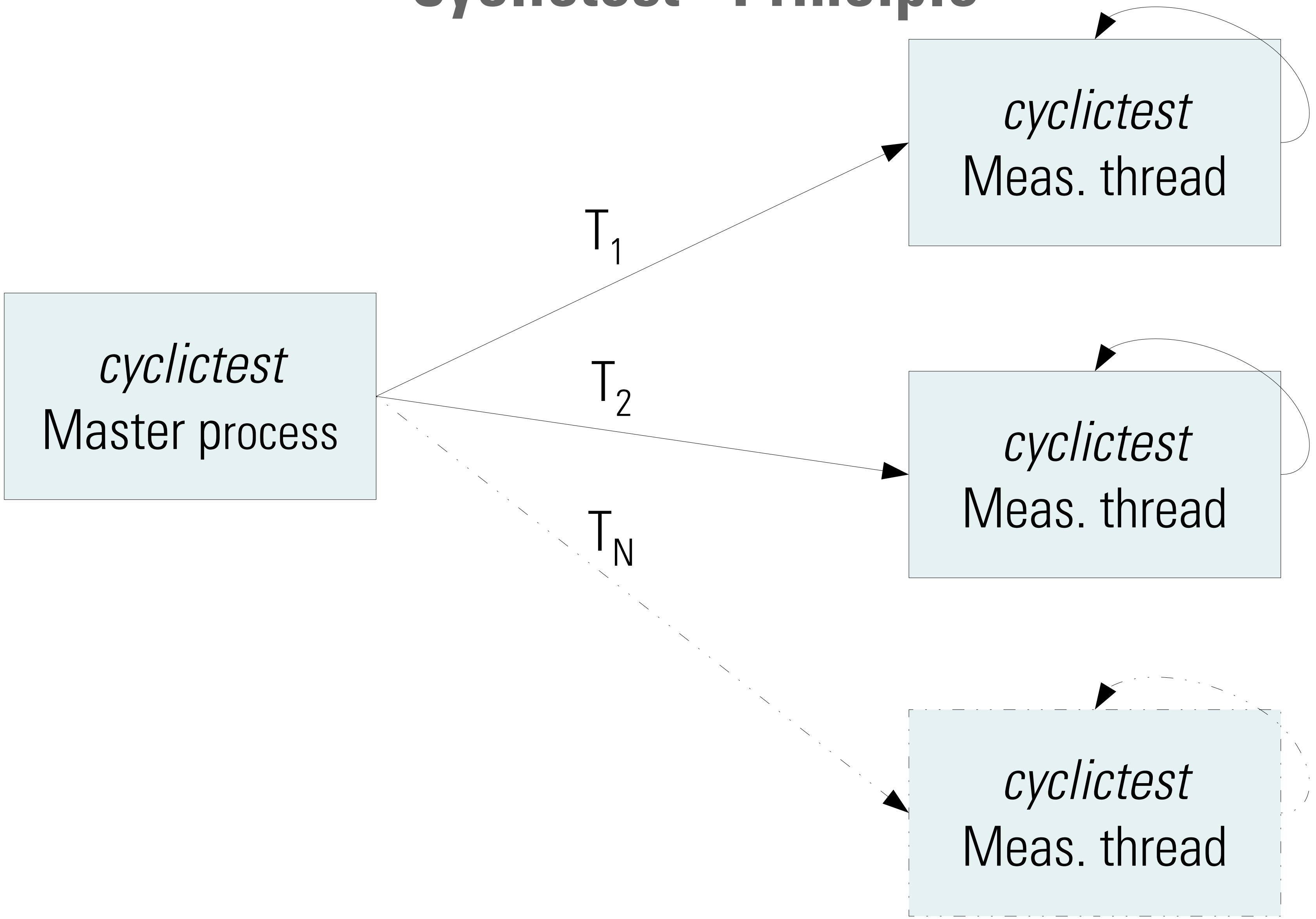
Internal measurement with simulation
Cyclictest

```
# cyclictest -a -t -n -p99
```

Real-world internal measurement
Application

```
# <application>
```

Cyclictest - Principle



Cyclictest: Command line parameters

```
# cyclictest -a -t -n -p99 -i100 -d50
560.44 586.11 606.12 211/1160 3727
T: 0 (18617) P:99 I:100 C:1,011,846,111 Min:  2 Act:  4 Avg:  5 Max: 39
T: 1 (18618) P:98 I:150 C:  708,641,019 Min:  2 Act:  5 Avg: 11 Max: 57
```

-a PROC *Affinity*. Run all threads on processor number **PROC**. If **PROC** is not specified, run thread #N on processor #N.

-t NUM *Threads*.Create **NUM** test threads (default is 1). If **NUM** is not specifed, **NUM** is set to the number of available CPUs.

-n *Nanosleep*. Run the tests with **clock_nanosleep()**.This is the standard and should always be used.

-p99 *Priority*. Set the priority of the first thread. The given priority is assigned to the first test thread. Each further thread receives the priority reduced by the number of the thread.

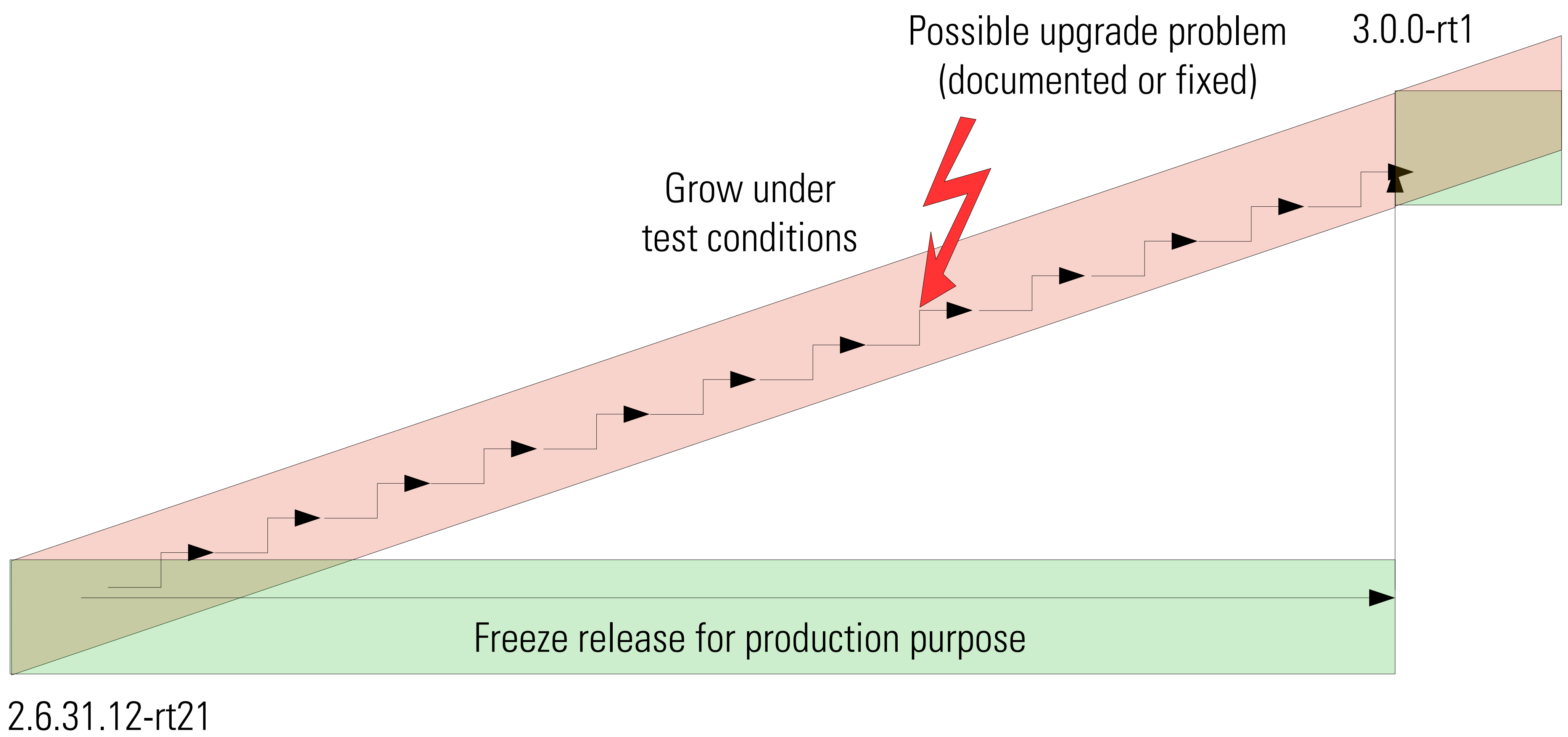
-i100 *Interval*. Repetition interval of the first thread in µs (default is 1000 µs).

-d50 *Delay of additional threads*. Set the distance of thread intervals in µs (default is 500 µs). When cyclictest is called with the -t option and more than a single thread is created, then this distance value is added to the interval of the threads.

Why are we testing computer boards and systems?

- Use as release testing for OSADL's „Latest Stable“ Linux real-time kernel
- Provide selection criteria for automation hardware
- Generate availability and stability data of individual systems
- **„Freeze and grow”**
- **Generate reliable data for certification purpose (e.g. real-time)**

„Freeze and grow“



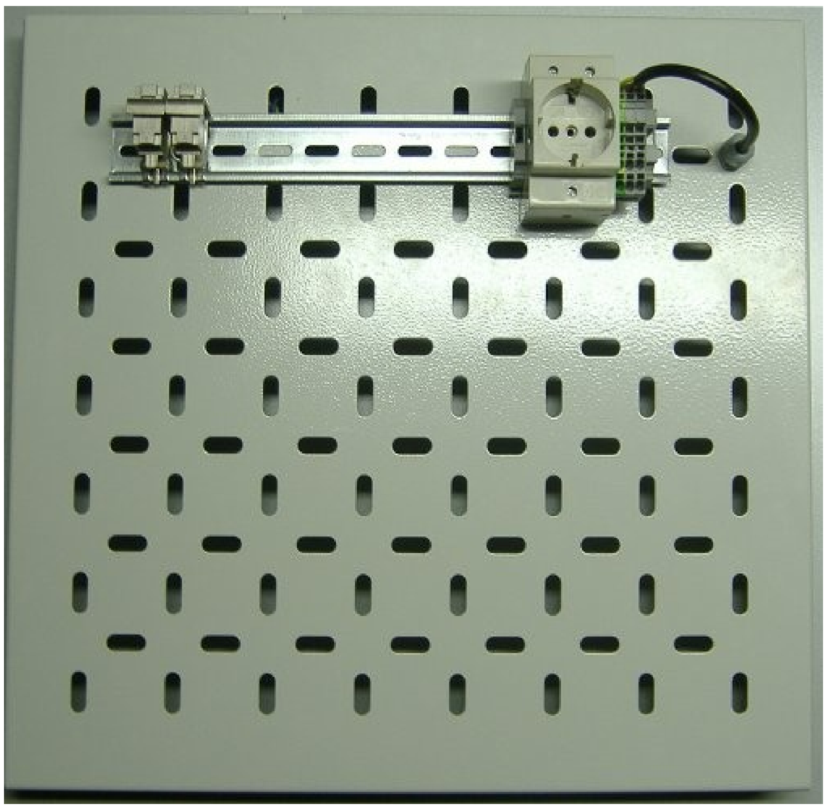
OSADL QA Farm osadl.org/QA (1)

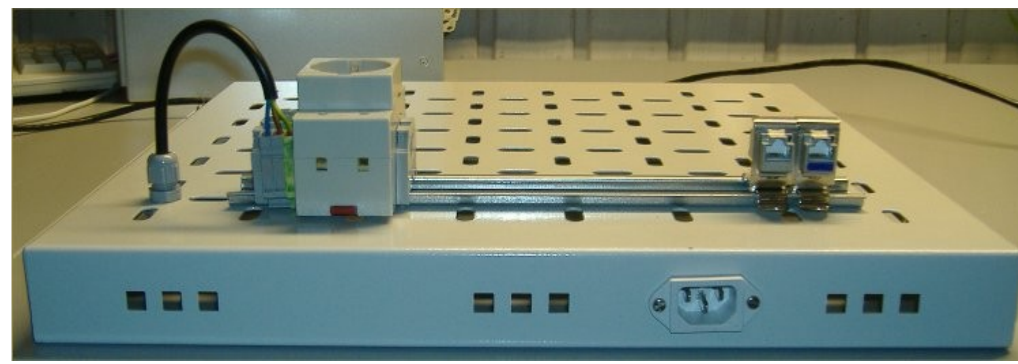
OSADL Test Rack

- Eight individual tablets
- Power supply 220 V, Ethernet, RS232
- 10/100/1000 Mb/s Switch with port mirroring
- Power distribution unit with power monitoring for every tablet
- Remote power switch for every tablet
- Serial network adapter for every tablet
- KVM switch (optional) for every tablet
- One central server per rack

OSADL QA Farm osadl.org/QA (2)

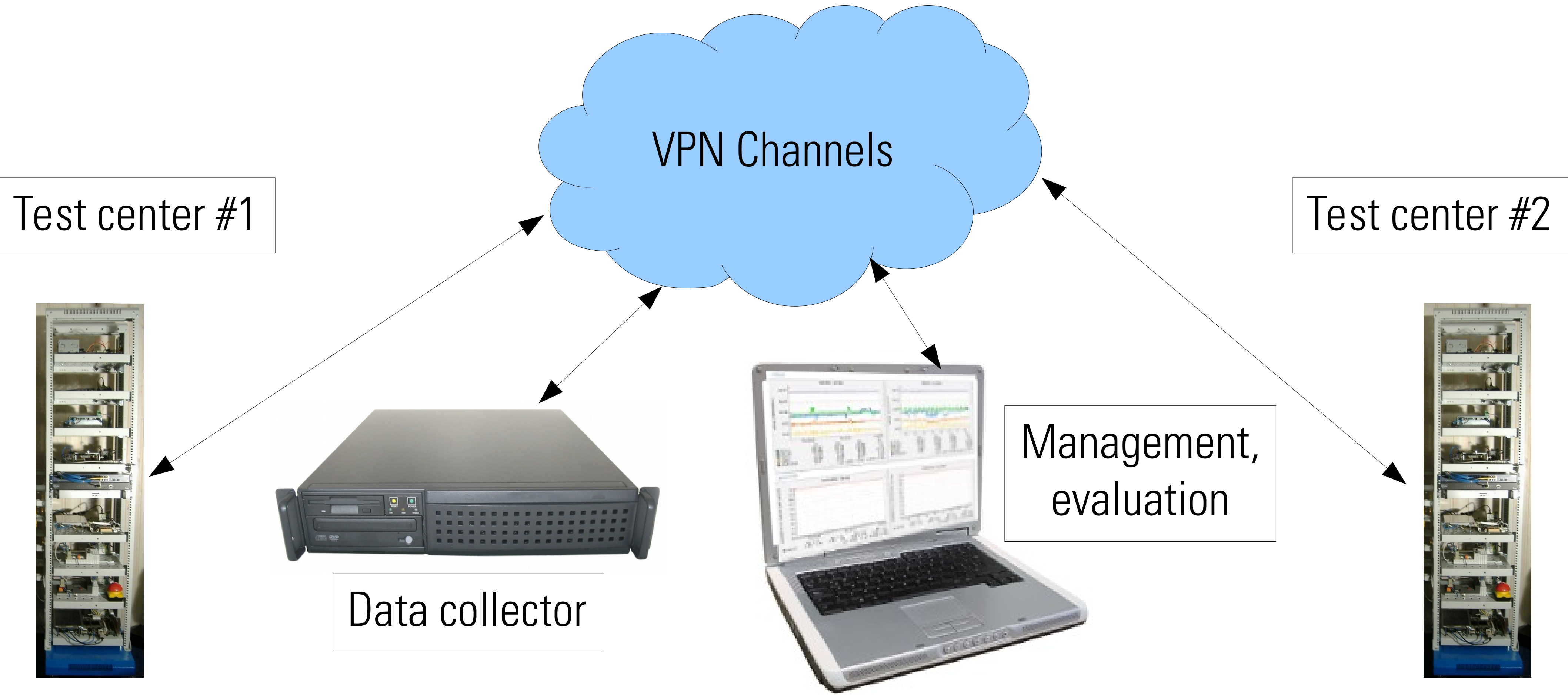
Mounting the individual systems on specially designed removable tablets





OSADL QA Farm osadl.org/QA (3)

Cloud-based communication between test systems, data collectors and admin systems



OSADL QA Farm osadl.org/QA (4)

Exhaustive and transparent documentation of every system

- Vendor, board
- BIOS version
- Distribution
- Kernel
- Kernel command line
- Command to generate latency plot histogram data
- CPU, interrupts, scaling governor, timer, RT features
- RAM, DIMMs
- PCI components
- BIOS analysis
- Kernel configuration, off-tree patches, script to reproduce kernel source tree

Processor families/processors under test (selection)

ARM

Broadcom
- BCM2708 @700 MHz, 32 bit

Freescale
- i.MX27 @400 MHz, 32 bit
- i.MX35 @532 MHz, 32 bit
- i.MX53 @886 MHz, 32 bit
- **i.MX6 X4 @996 MHz, 32 bit**

Marvell
- **SheevaPlug @1200 MHz, 32 bit**

Texas Instruments
- AM3517 @600 MHz, 32 bit
- OMAP3525 @720 MHz, 32 bit
- OMAP4430 X2 @1008 MHz, 32 bit
- OMAP4460 X2 @1200 MHz, 32 bit

MIPS

ICT
- Loongson 2F @800 MHz, 64 bit

PowerPC

Freescale
- MPC 5200 @396 MHz, 32 bit

x86/x86_64

AMD
- K6 3D, @333 MHz, 32 bit
- LX-800 @500 MHz, 32 bit
- Athlon XP 2000+, 32 bit
- Athlon 64 2800+, 64 bit
- **G-Series T56N @1400 MHz, 64 bit**
- Phenom II X6 @3200 MHz, 64 bit
- **Opteron X32 @2100 MHz, 64 bit**
- **Kaveri A10 7850k @3700 MHz, 64 bit**

Intel
- Pentium @133 MHz, 32 bit
- Atom D510 @1667 MHz, 64 bit
- Atom N270 @1600 MHz, 32 bit
- Atom D2700 @2133 MHz, 64 bit
- Celeron M @1500 MHz, 32 bit
- Pentium M @2300 MHz, 32 bit
- Xeon @2000 MHz, 32 bit
- Core 2 Duo @2400 MHz, 64 bit
- Core 2 Quad @2400 MHz, 32 bit
- Nehalem 975 @3333 MHz, 32 bit
- **Gulftown X990 @3467 MHz, 64 bit**
- **Sandybridge 3770 @3400 MHz, 64 bit**
- **Haswell 4960X @3600 MHz, 64 bit**

VIA
- C3 Samuel 2 @533 MHz, 32 bit
- C7 @1000 MHz, 32 bit
- **Nano X2 L4050 @1400 MHz, 64 bit**

Continuously determined variables (1)

Benchmarks
- GL benchmark gltestperf
- UnixBench (multi-core)
- UnixBench (single-core)
- UnixBench 2D graphics performance

Disk
- Disk IOs per device
- Disk latency per device
- Disk throughput per device
- Disk usage in percent
- Disk utilization per device
- File system mount-scheduled checks
- File system time-scheduled checks
- Filesystem usage (in bytes)
- Inode usage in percent
- IO Service time
- IOstat
- S.M.A.R.T values of every drive

Network
- eth0 errors
- eth0 traffic
- Firewall Throughput
- HTTP loadtime of a page
- Netstat

NFS
- NFS Client
- NFSv4 Client

Processes
- Fork rate
- Number of threads
- Processes
- Processes priority
- VMstat

Real-time system
- 5-min max. timer and wakeup latency
- 5-min max. timer offsets
- 5-min max. wakeup latency
- RT Features

Email
- Sendmail email traffic
- Sendmail email volumes
- Sendmail queued mails

Sensors
- Fans
- HDD temperature
- Power consumption
- Temperatures

Continuously determined variables (2)

System

- Available entropy
- C states
- CPU frequency
- CPU usage
- File table usage
- Individual interrupts
- Inode table usage
- Interrupts and context switches
- Kernel version
- Load average
- Logged in users
- Memory usage
- Split memory usage
- Application memory usage
- Swap in/out
- Uptime

Virtual systems

- Virtual domain block device I/O
- Virtual domain CPU time
- Virtual domain memory usage
- Virtual domain network I/O

Time synchronization

- NTP kernel PLL estimated error (secs)
- NTP kernel PLL frequency (ppm + 0)
- NTP kernel PLL offset (secs)
- NTP states
- NTP timing statistics for system peer

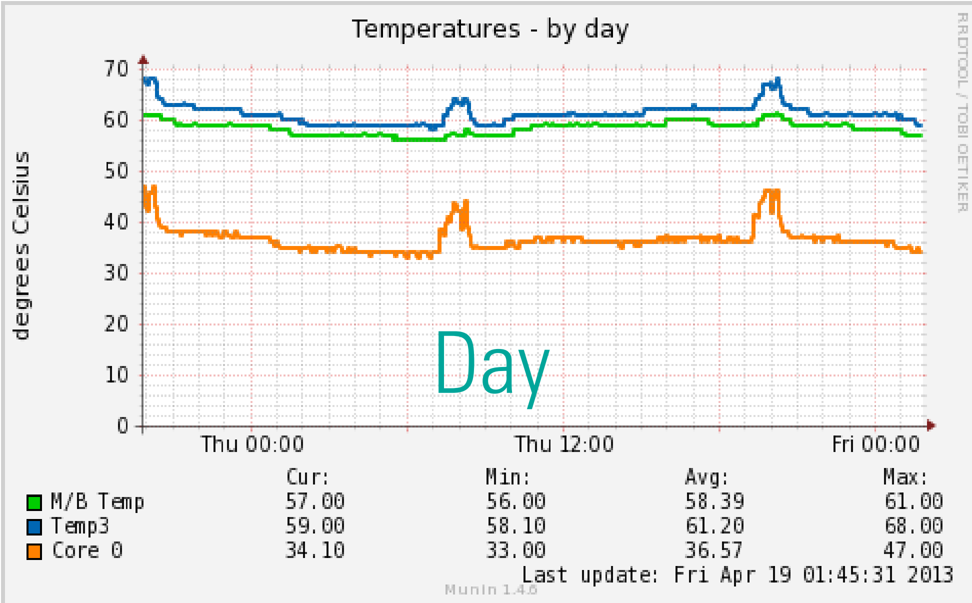
CPU and graphics benchmarks

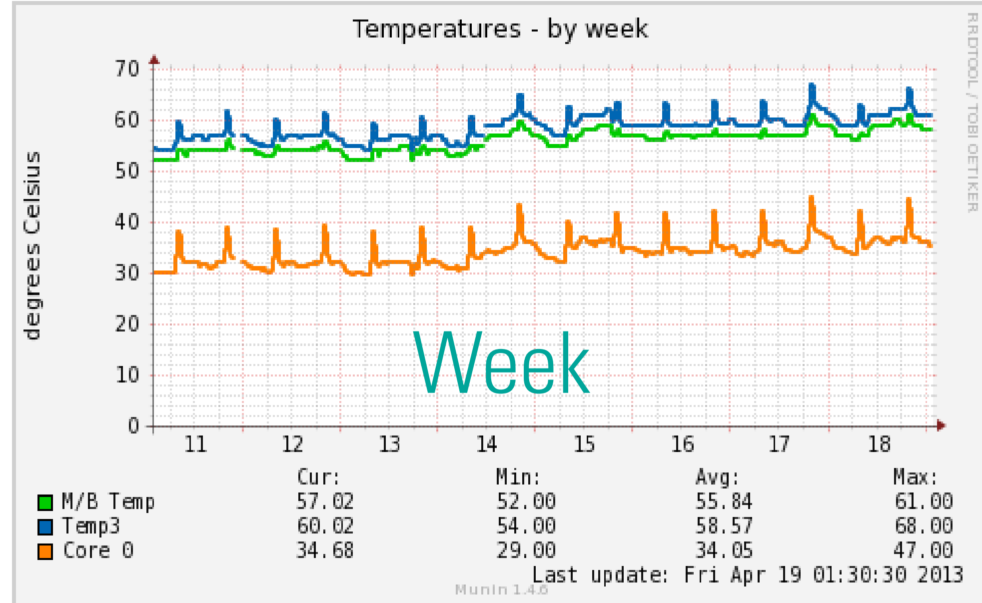
Slowest (reddest)

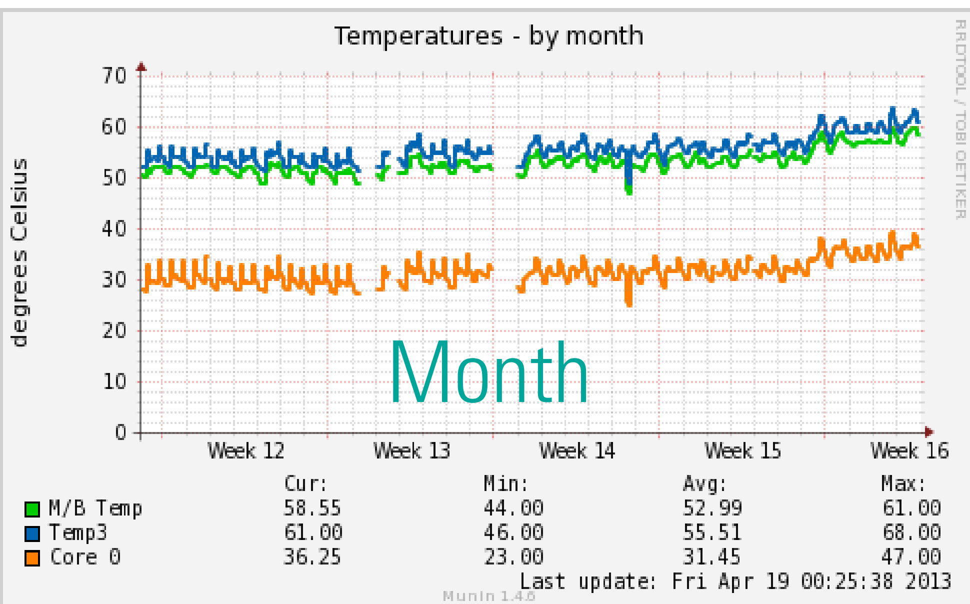
r5s0	51053	5215.2	7310.4	209.98	1800.7	1240.1	24.04. 18:12
r9s1	53541	4884.6	6549.8	85.08	1364.0	839.7	24.04. 18:11
r0s0	62655	10528.0	9547.5	253.67	3142.4	1716.1	24.04. 18:12
r8s2	62708	3780.9	7659.2	301.56	1432.7	1296.2	24.04. 18:10
r3s8	69034	14246.8	16194.3	171.38	4926.6	1984.7	24.04. 18:13
r7s2	89680	22966.9	6662.2	189.33	2779.4	1752.0	21.04. 06:14
r5s1	102987	24195.3	19470.7	150.85	4038.8	2330.2	24.04. 18:12
r0s2	105523	8066.6	12745.5	129.70	3714.9	1764.1	24.04. 18:11
r8s8	124787	14457.6	12704.9	178.59	2922.2	1961.1	24.04. 06:11
r0s3	149833	28304.6	15877.8	102.33	5621.2	2359.5	24.04. 18:11
r8s3	171306	23773.2	12331.1	214.99	5476.0	2757.5	24.04. 18:11
r4s6	180687	31339.9	17958.5	284.93	4211.0	2948.6	24.04. 18:11
r0s8	194089	22765.4	9104.9	149.12	7525.3	2557.9	24.04. 18:16

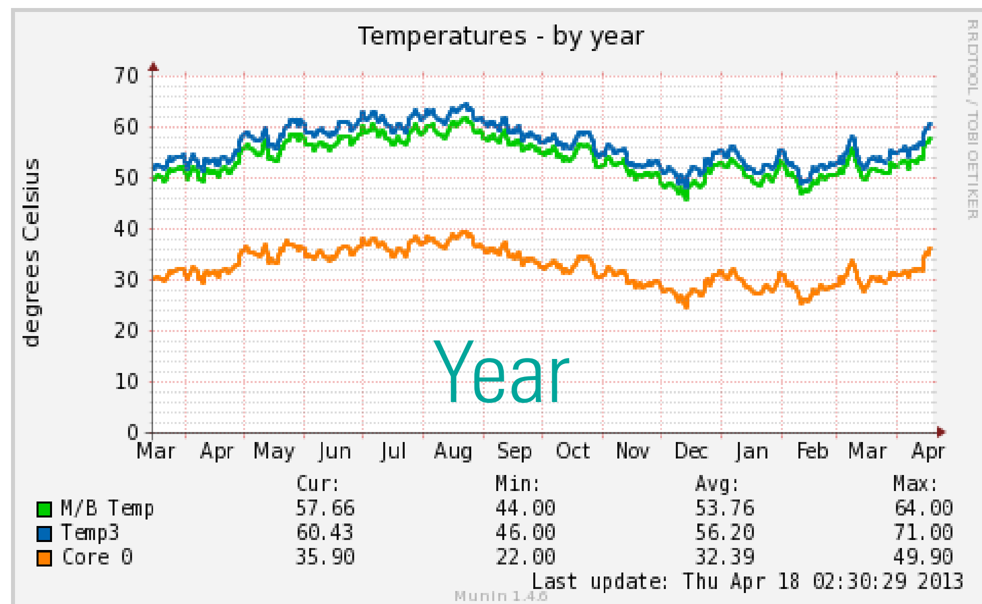
Fastest (greenest)

Four different time resolutions (e.g. temperatures)









Alert colors of warnings and alarms (Munin)

Warning

Alarm

- rack1slot2.osadl.org [benchmarks disk network nfs processes sendmail sensors system time]
- rack1slot3.osadl.org [benchmarks disk network nfs processes sendmail sensors system time]
- rack1slot4.osadl.org [benchmarks disk network nfs processes sendmail sensors system time]
- rack1slot6.osadl.org [benchmarks disk network processes sendmail system time]
- rack1slot8.osadl.org [benchmarks disk network nfs processes sendmail sensors system time]
- rack2slot0.osadl.org [benchmarks disk network nfs postfix processes sendmail sensors system time]
- rack2slot2.osadl.org [benchmarks disk network nfs processes system time]
- rack2slot3.osadl.org [benchmarks disk network nfs processes system time]
- rack2slot5.osadl.org [benchmarks disk network nfs processes system time]
- rack2slot6.osadl.org [benchmarks disk network nfs processes sendmail sensors system time]
- rack2slot8.osadl.org [benchmarks disk network nfs processes system time]
- rack3slot0.osadl.org [benchmarks disk network nfs processes sendmail sensors system time]
- rack3slot1.osadl.org [benchmarks disk network nfs processes sendmail sensors system time]
- rack3slot2.osadl.org [benchmarks disk network nfs processes sendmail sensors system time]
- rack3slot3.osadl.org [benchmarks disk memory network nfs processes sendmail sensors system time]

Event recording with Nagios (1)

Host Status Totals

Up	Down	Unreachable	Pending
75	10	0	0

All Problems	All Types
10	85

Service Status Totals

Ok	Warning	Unknown	Critical	Pending
634	17	30	17	18

All Problems	All Types
64	716

Service Overview For All Host Groups

OSADL Test Racks (osadl-test-racks)

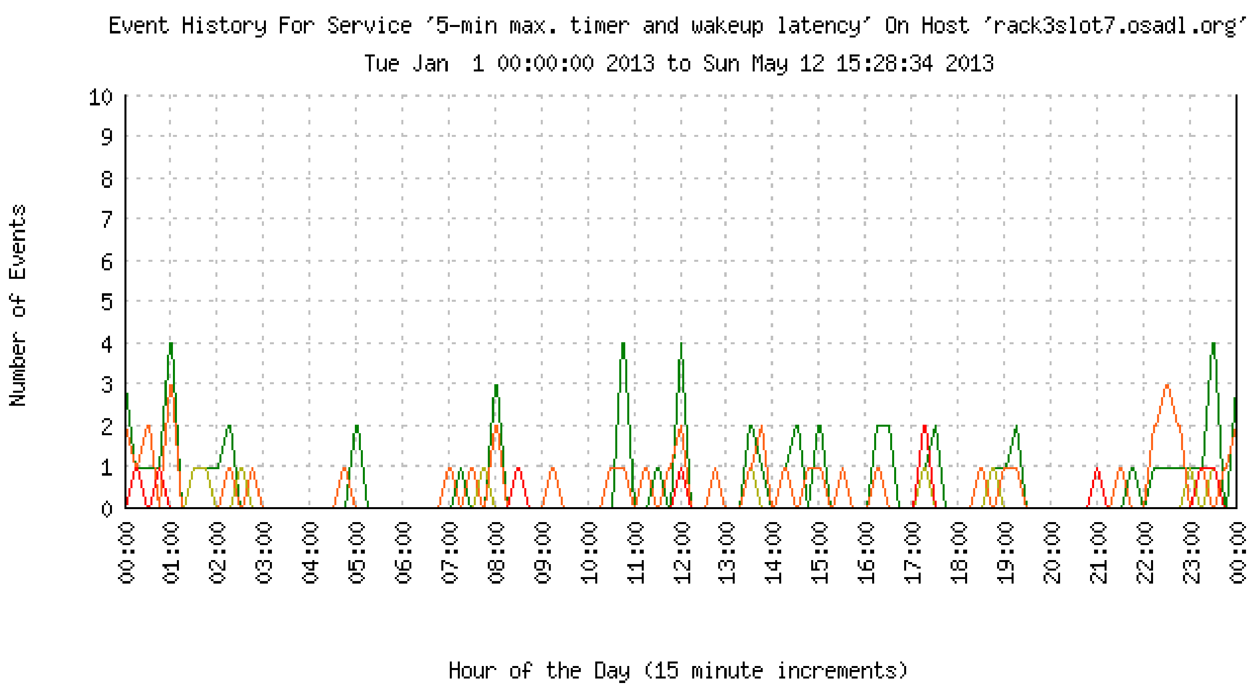
Host	Status	Services	Actions
ou-int.osadl.org	UP	1 OK	
ou.osadl.org	UP	1 OK	
rack0slot0.osadl.org	UP	10 OK	
rack0slot1.osadl.org	UP	6 OK 3 WARNING 1 UNKNOWN	
rack0slot2.osadl.org	UP	10 OK	
rack0slot3.osadl.org	UP	9 OK 1 PENDING	
rack0slot4.osadl.org	UP	9 OK	
rack0slot5.osadl.org	UP	4 OK 1 CRITICAL	

Web-Alm Servers (web-alm-servers)

Host	Status	Services	Actions
dns.web-alm.net	UP	1 OK	
mail.web-alm.net	UP	1 OK	
swiss.web-alm.net	UP	6 OK	
toro.web-alm.net	UP	4 OK	
www.osadl.org	UP	5 OK 1 PENDING	

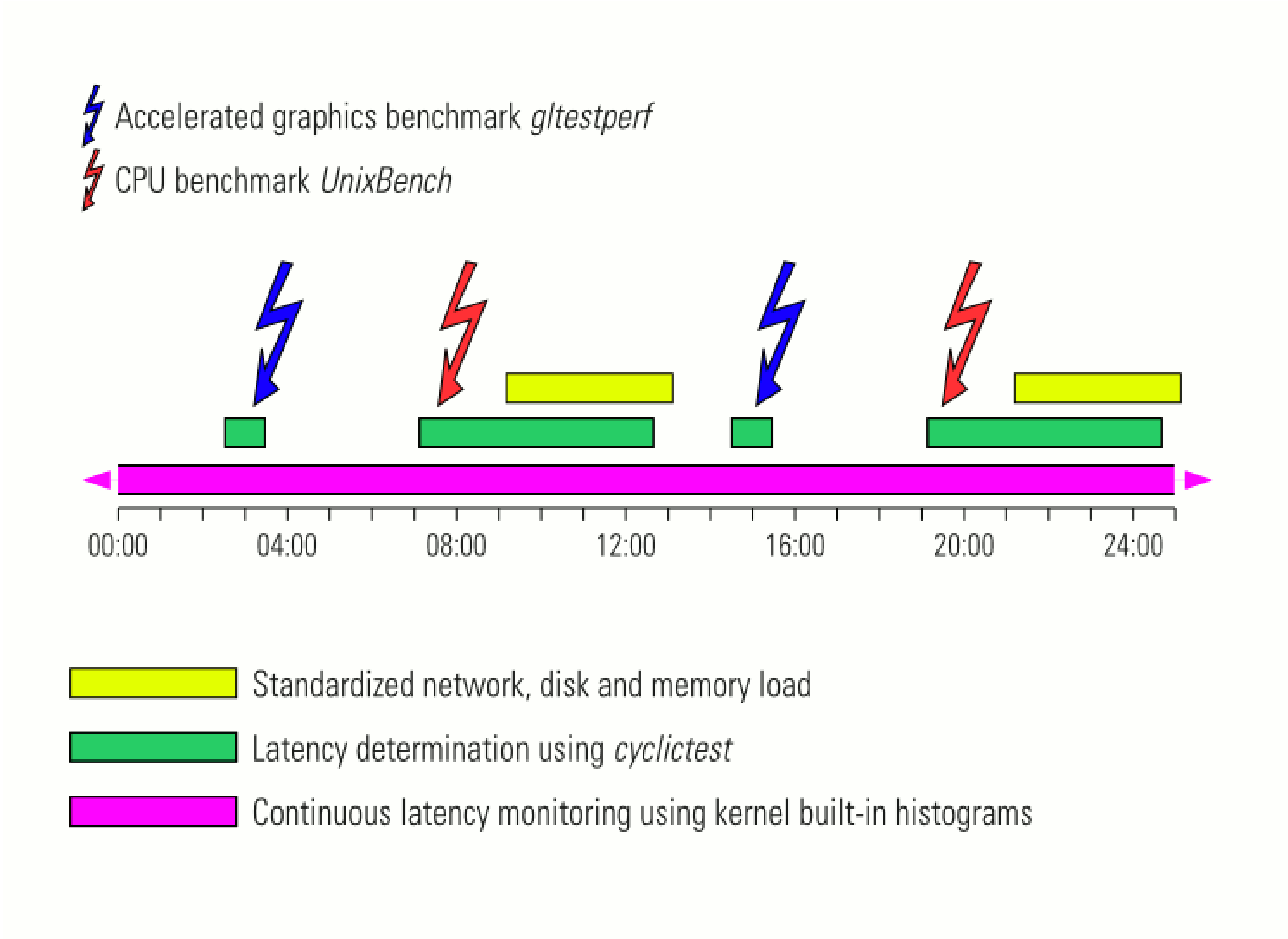
Event recording with Nagios (2)

Service alert histogram,
e.g. hour-of-the-day analysis of latency peaks in current year

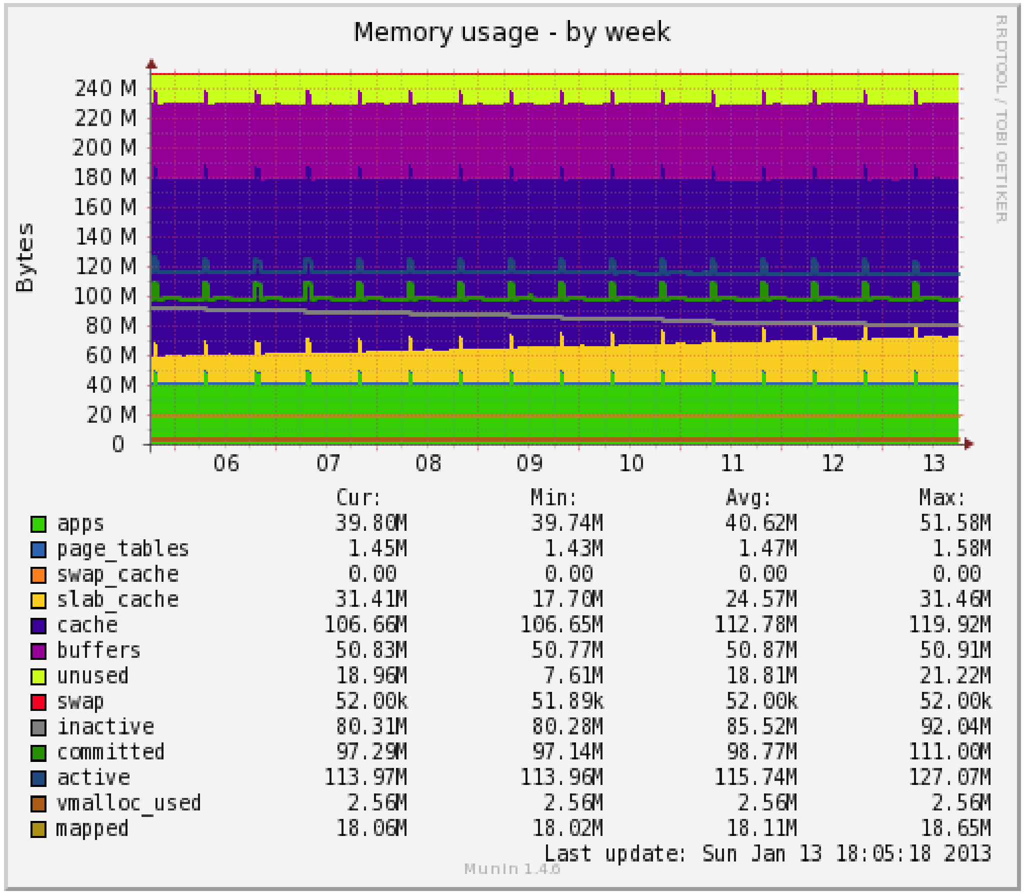


EVENT TYPE	MIN	MAX	SUM	AVG
Recovery (Ok):	0	4	62	0.65
Warning:	0	1	12	0.12
Unknown:	0	3	44	0.46
Critical:	0	2	9	0.09

Monitoring and benchmark schedule

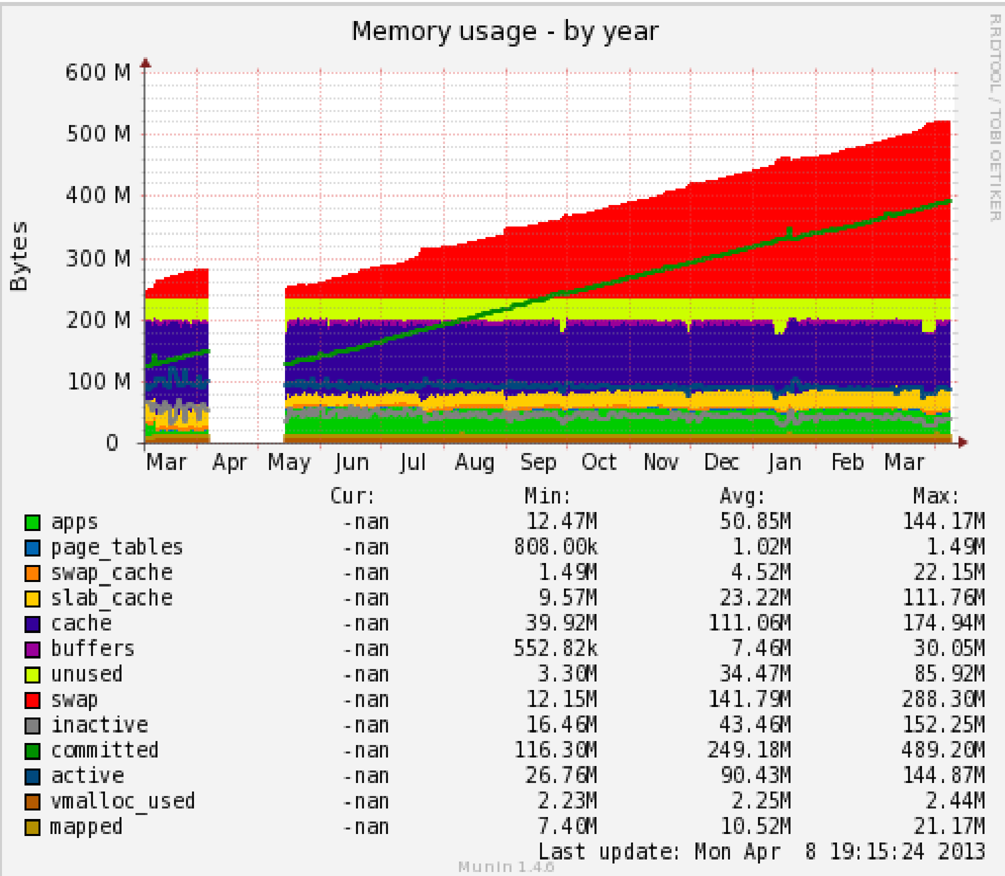


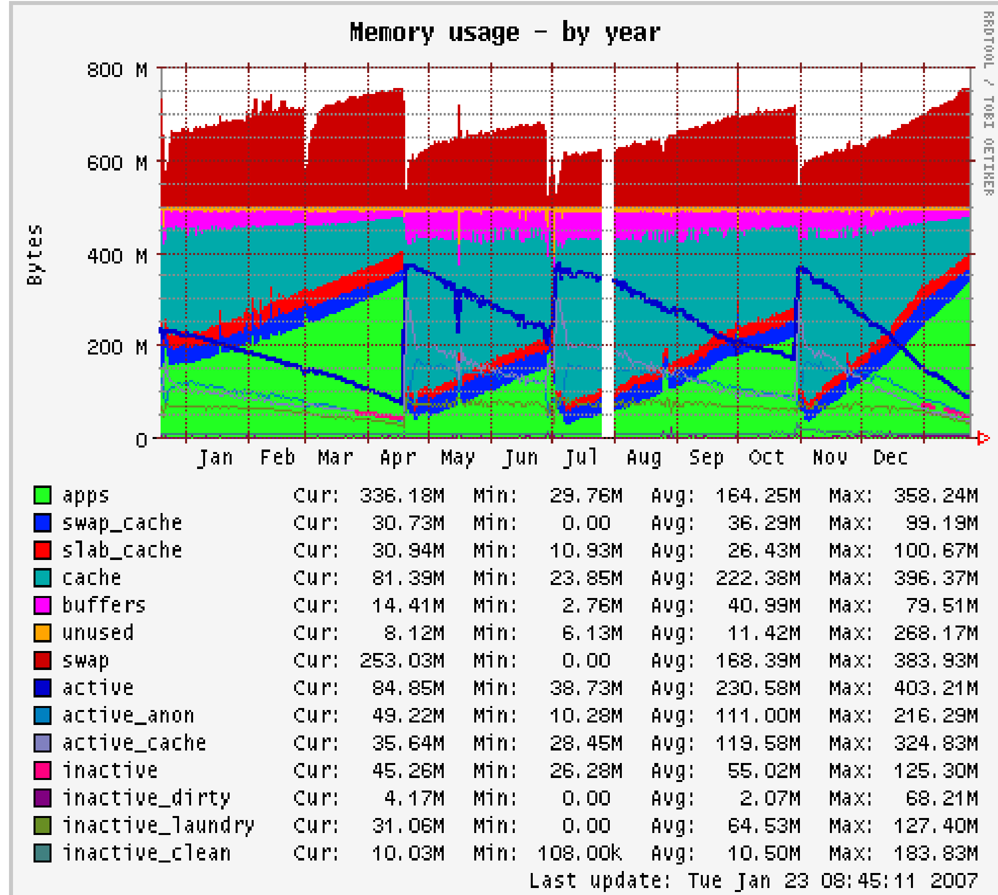
Example 1a: Memory leak diagnosis



Normal (no leak)

Example 1b: Memory leak diagnosis

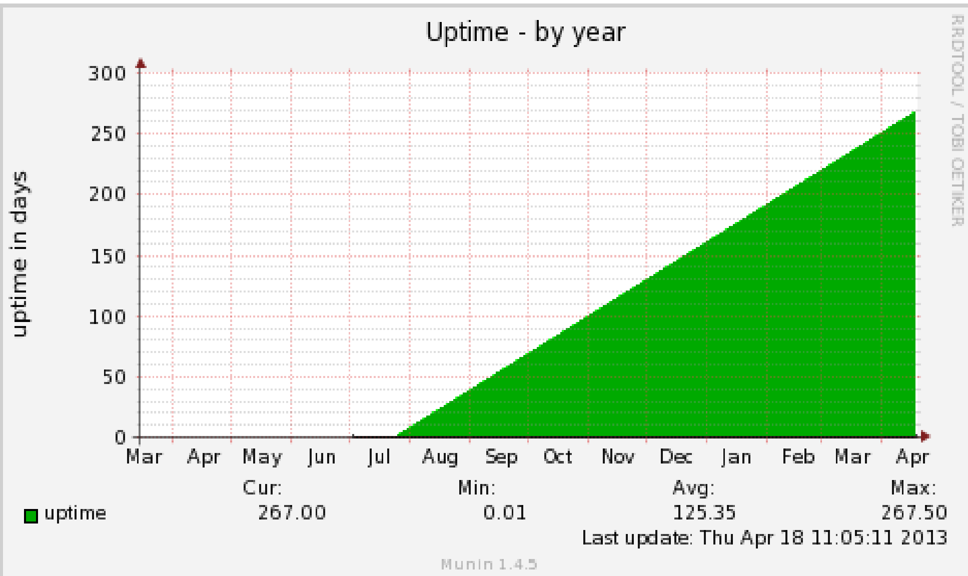


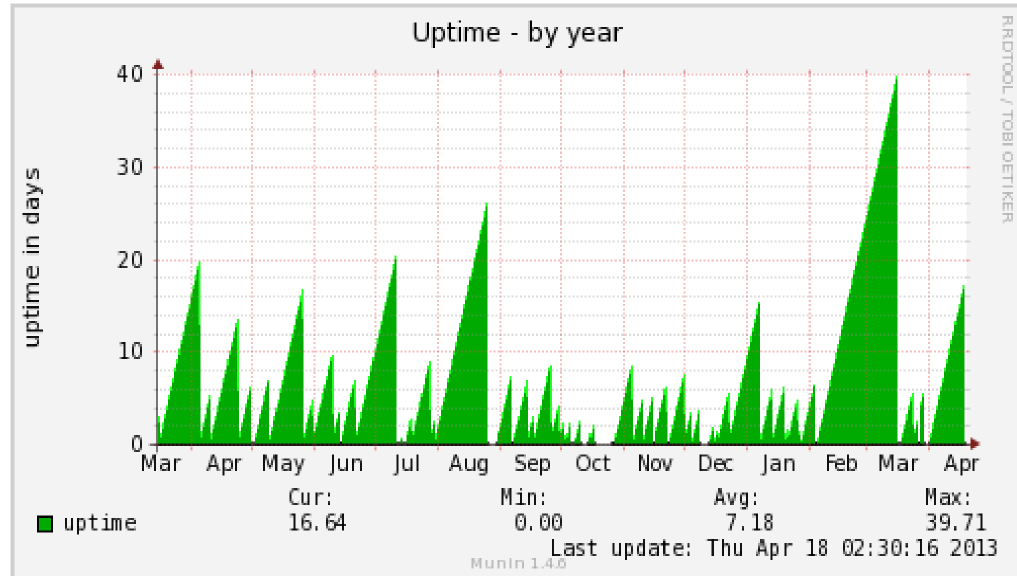


System leak

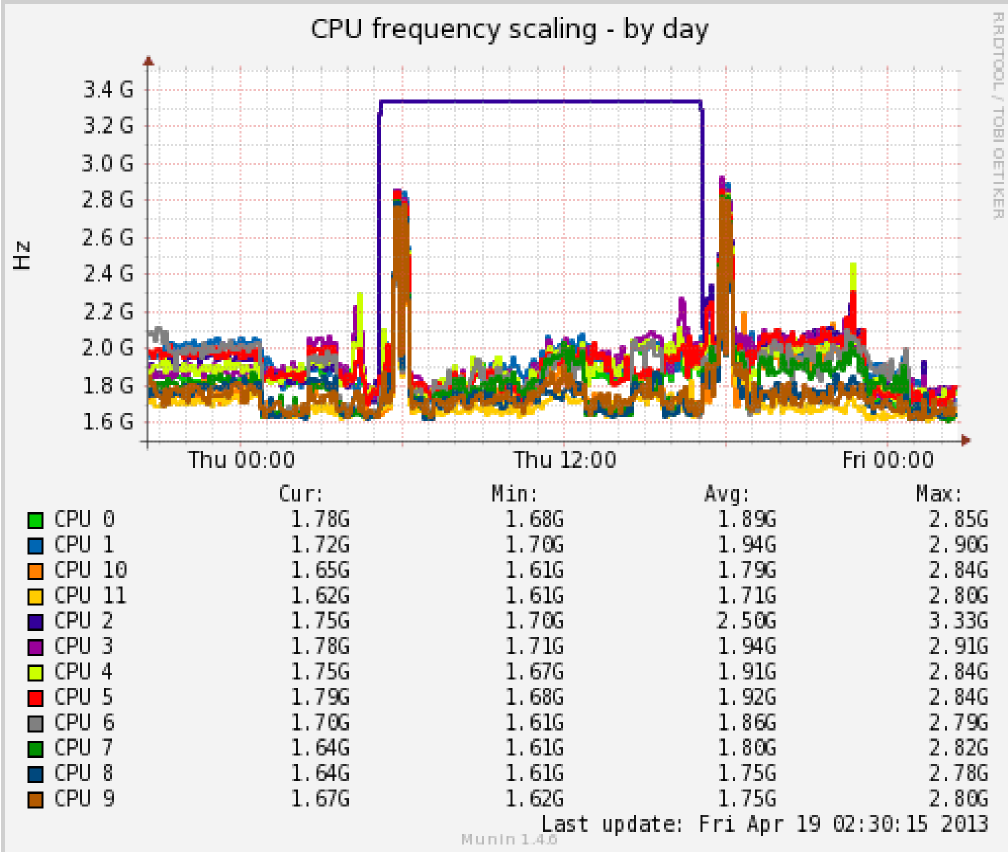
Application leak

Example 2: Stable vs. instable system

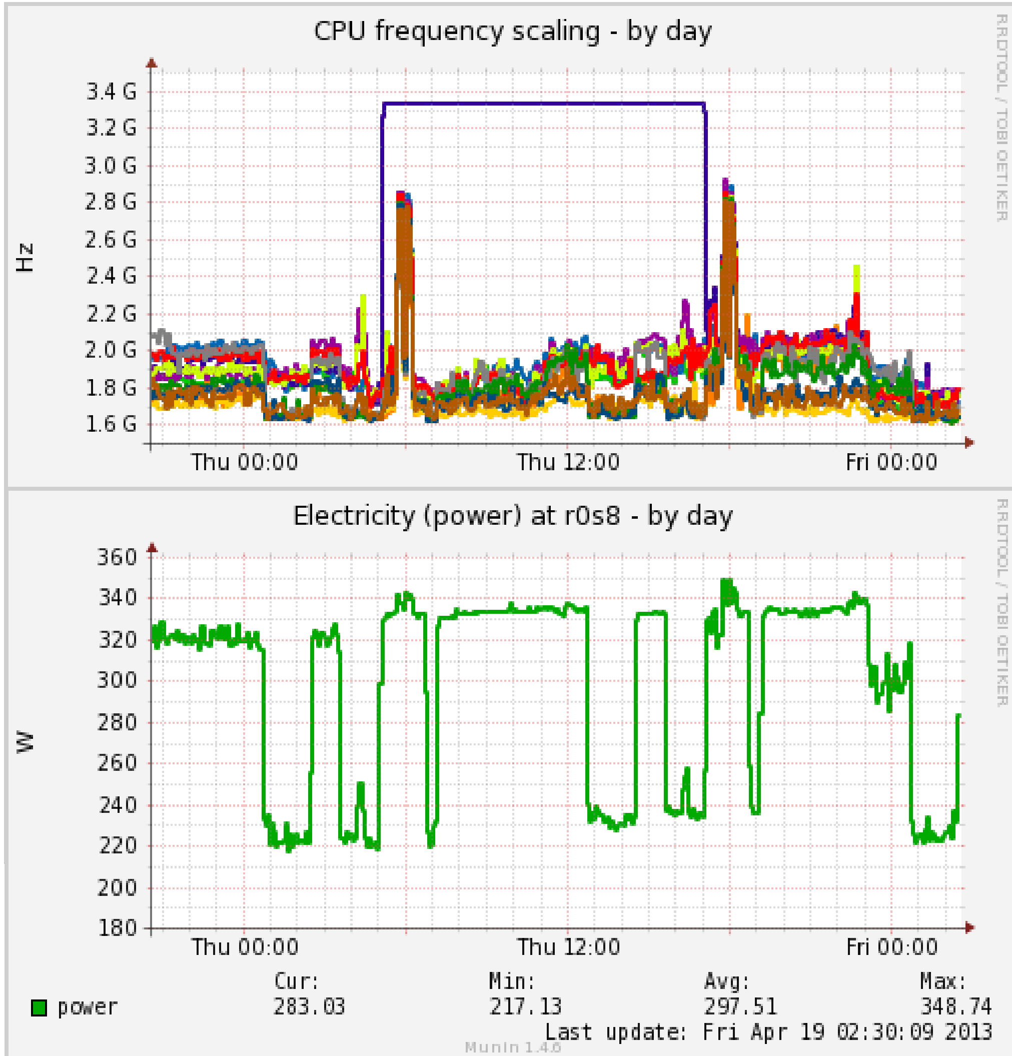




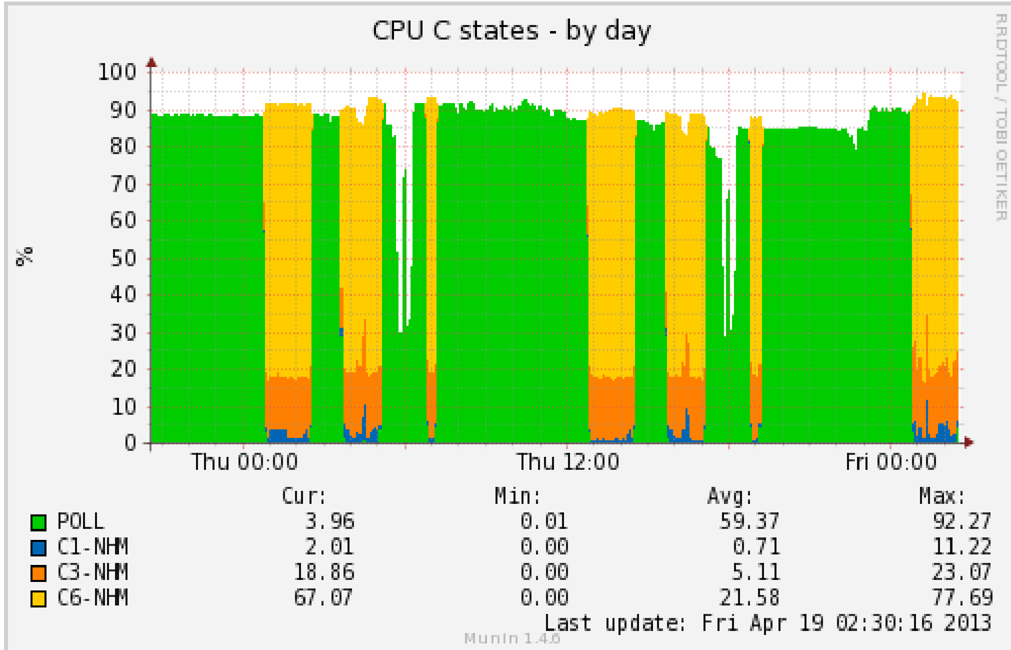
Example 3a: Power management



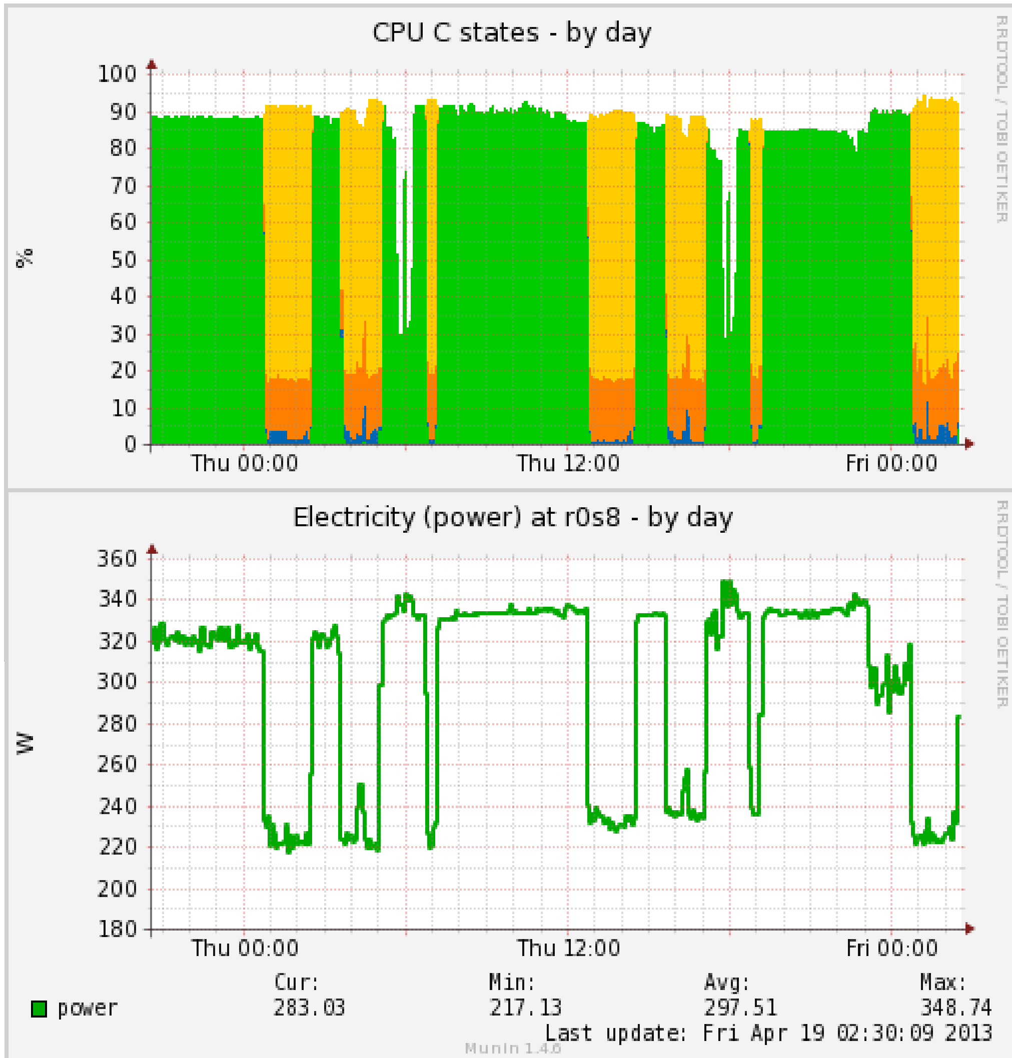
Example 3b: Power management



Example 3c: Power management

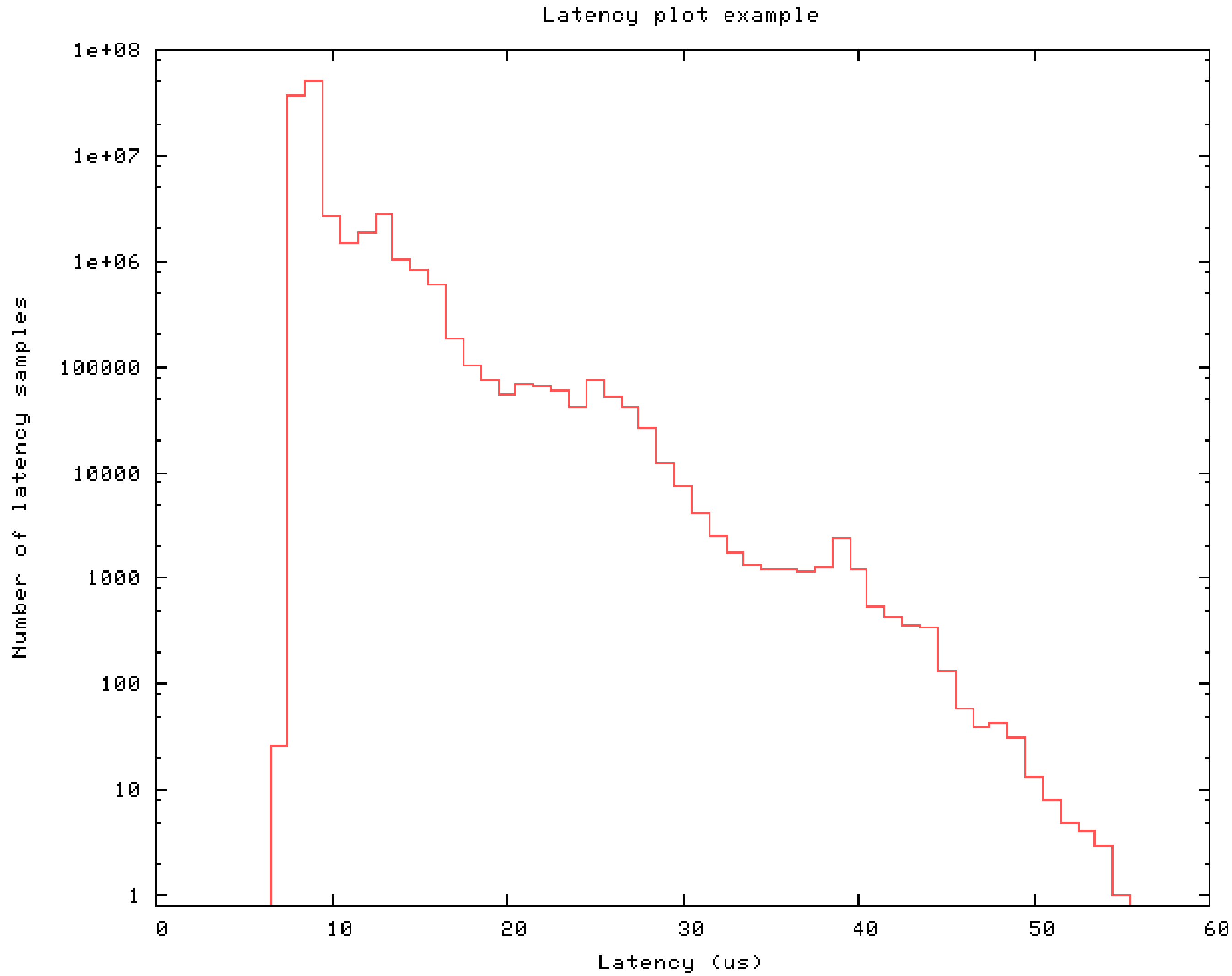


Example 3d: Power management



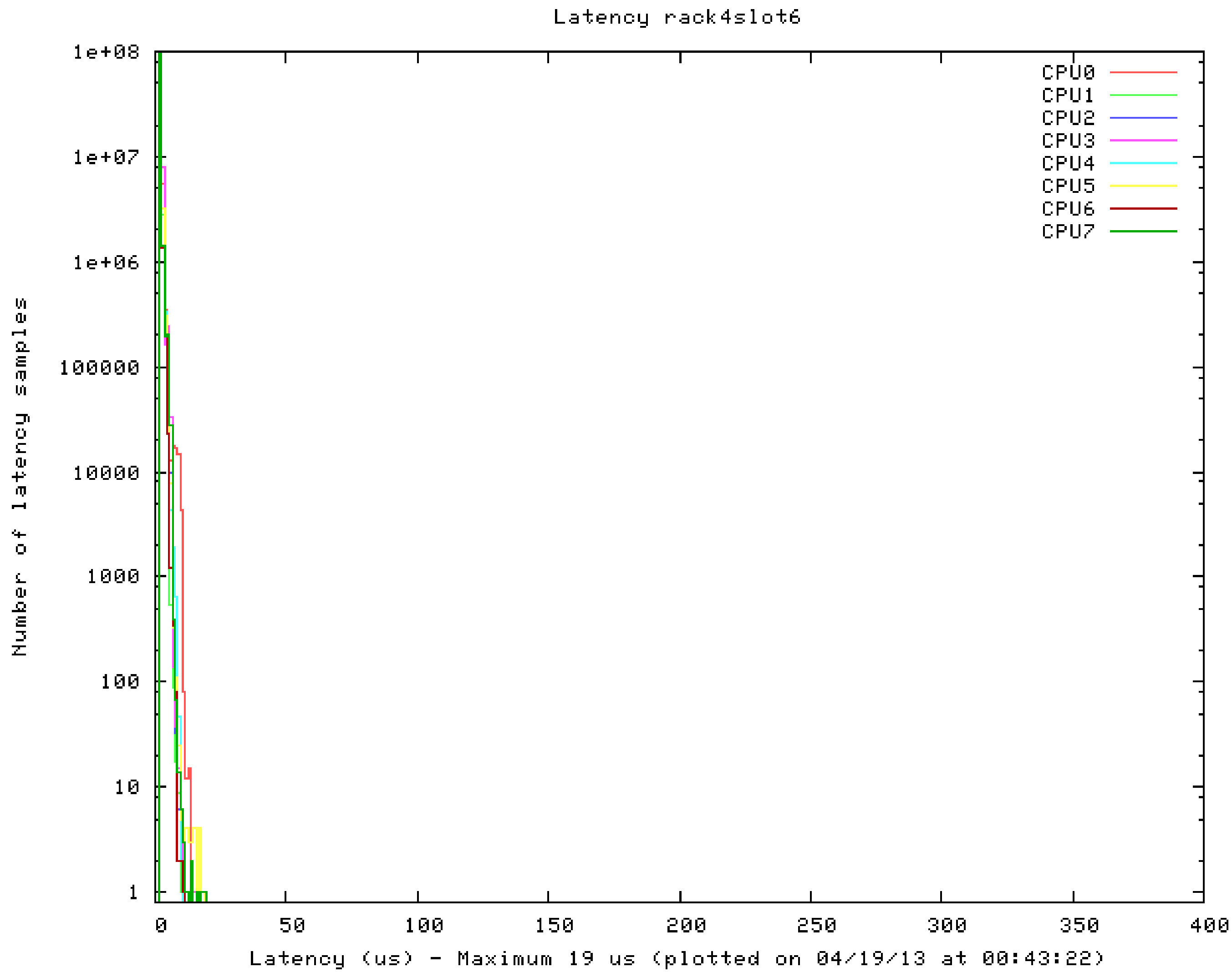
Example 4a: Determinism

Latency plot with linear x scale and logarithmic y scale



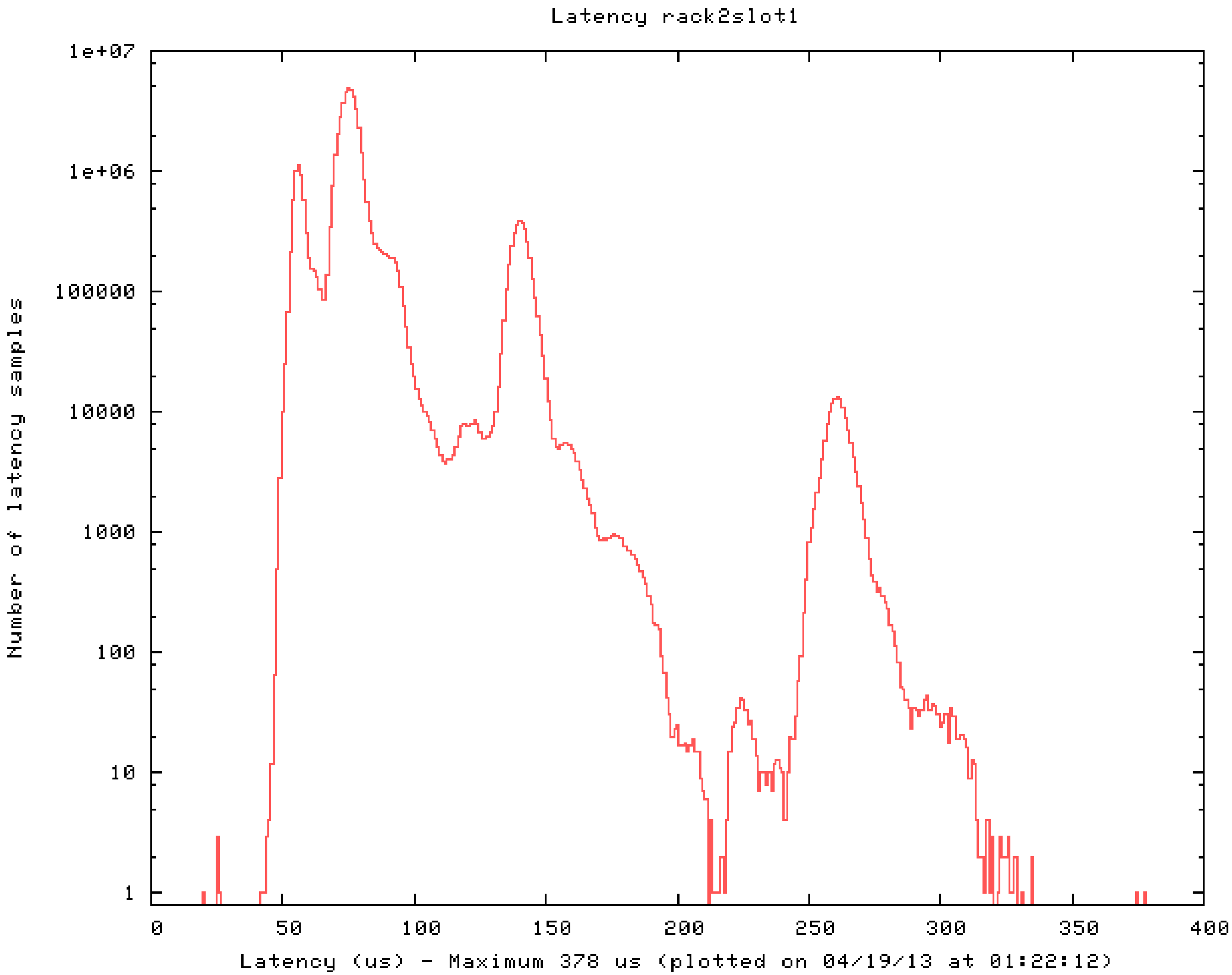
Example 4b: Determinism

Standard OSADL plot (very low maximum latency)



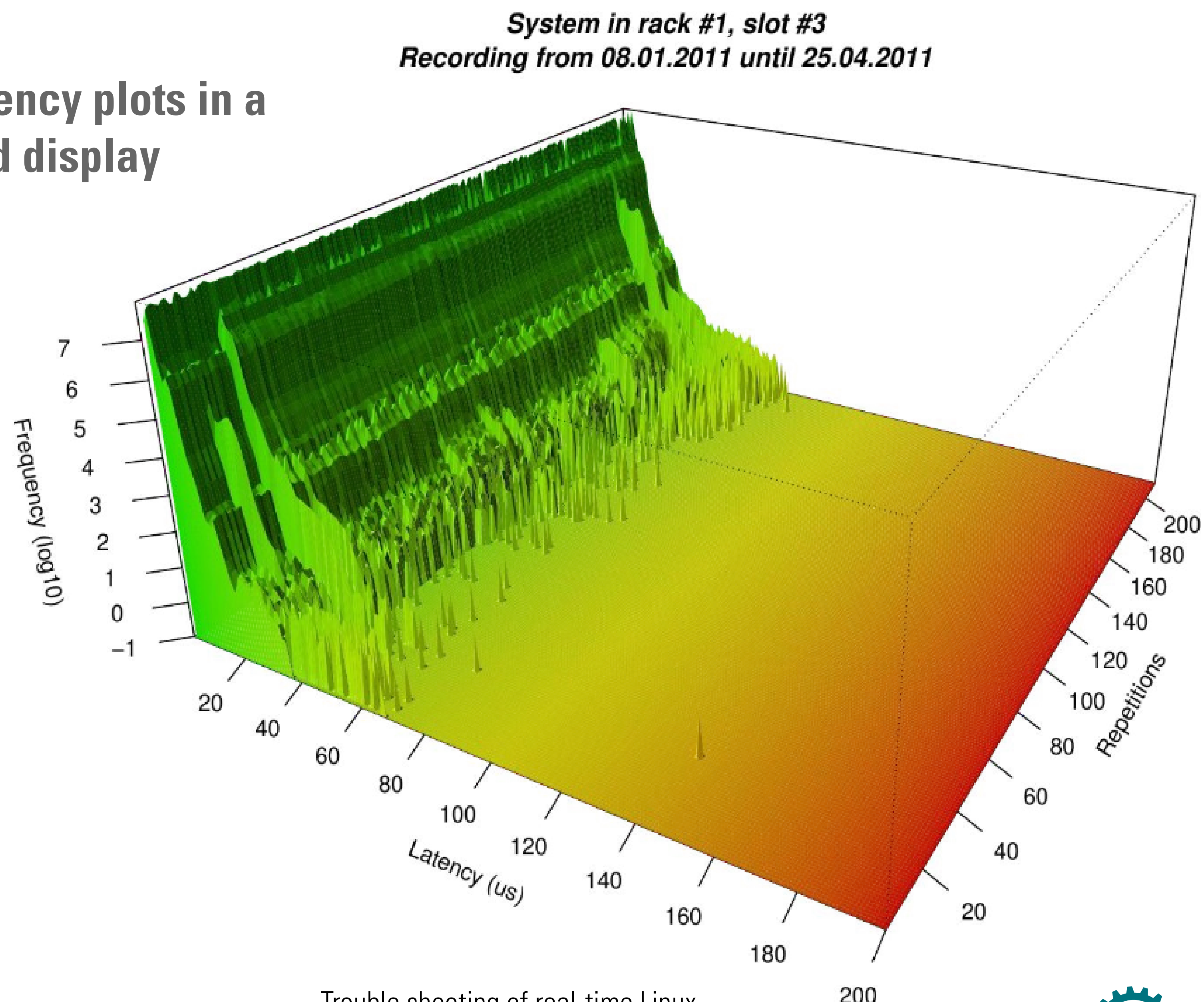
Example 4c: Determinism

Standard OSADL plot (relatively high maximum latency)



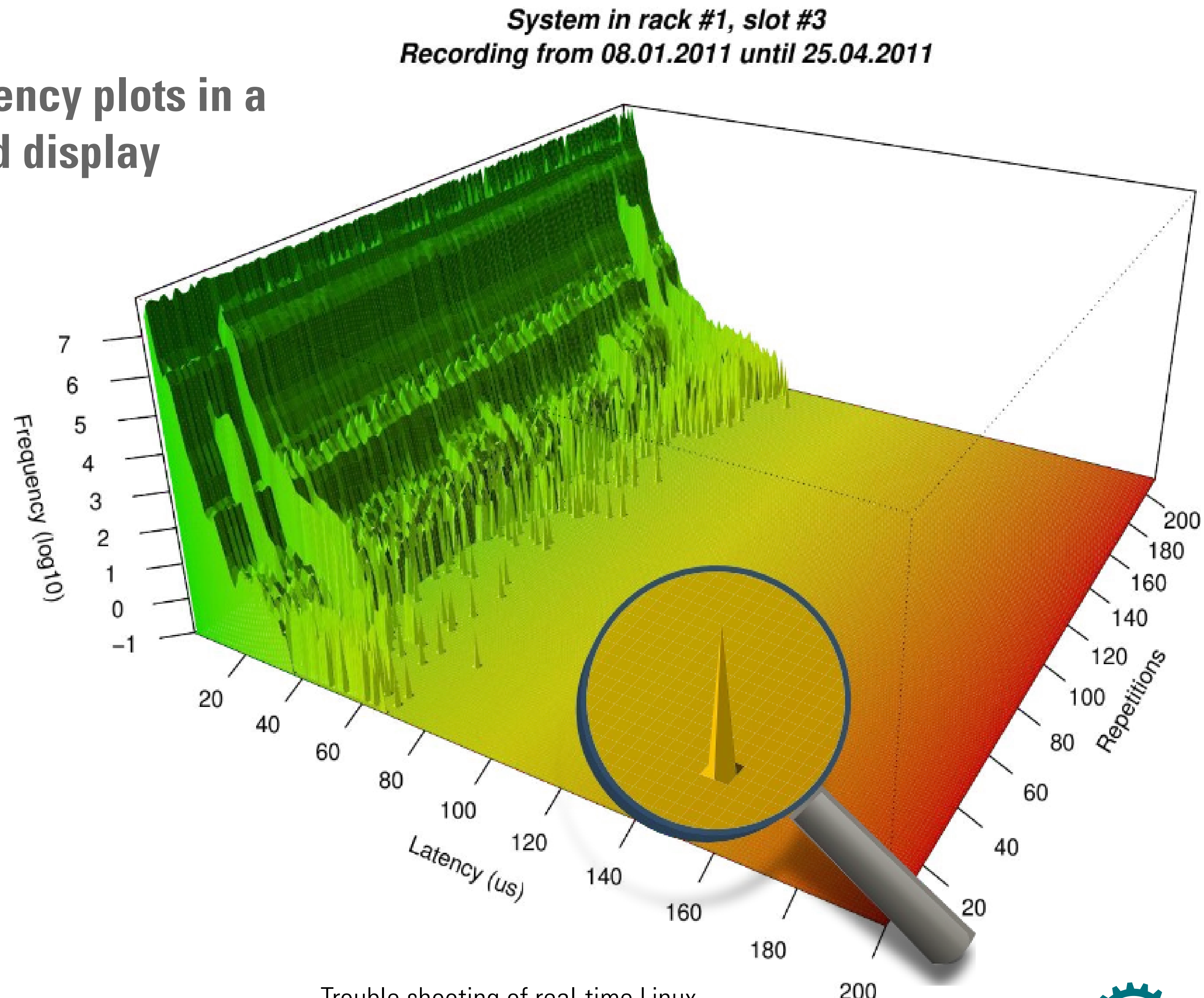
Repetitive latency plots each of 100 million cycles (1)

Consecutive latency plots in a single combined display



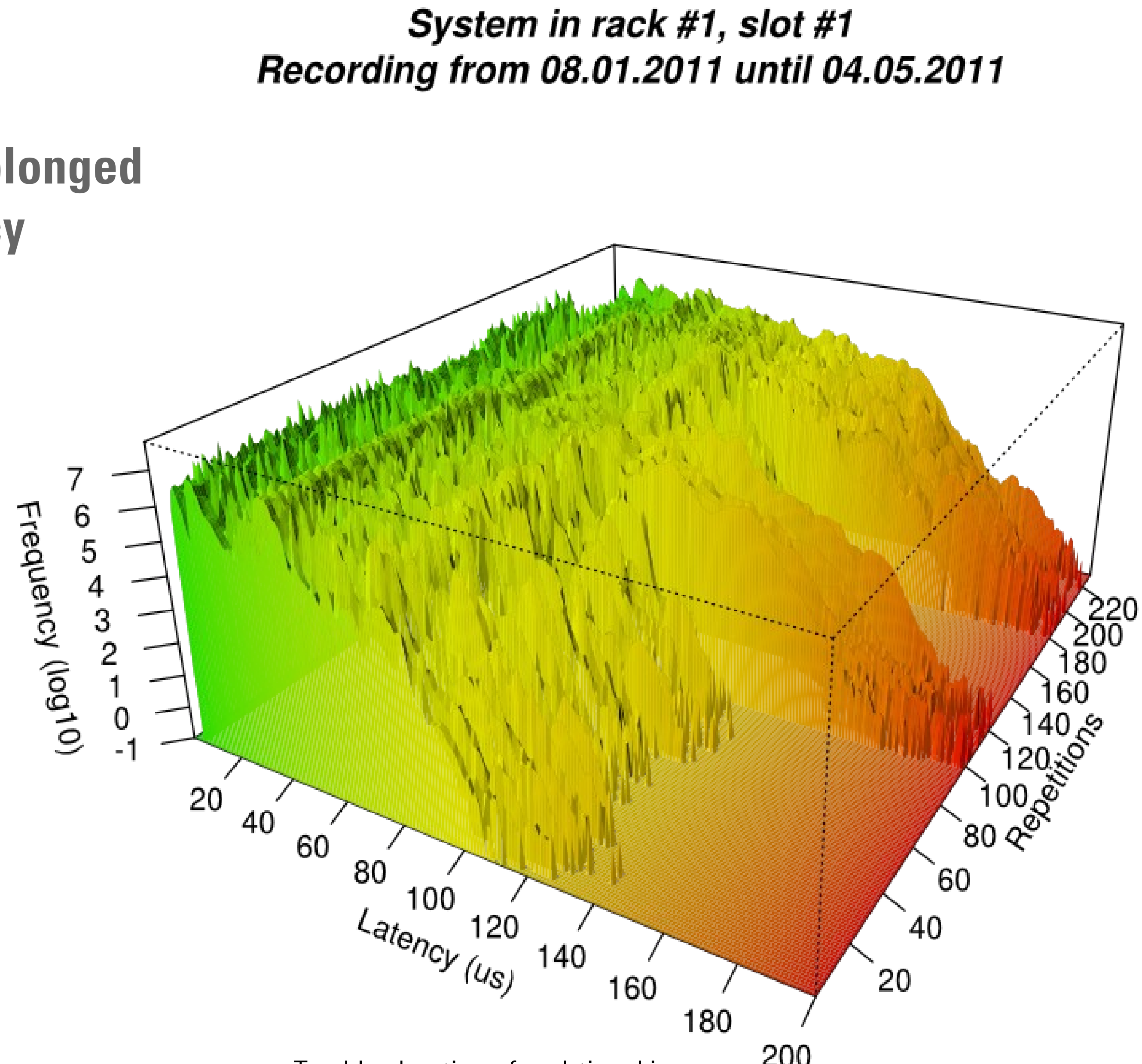
Repetitive latency plots each of 100 million cycles (2)

Consecutive latency plots in a single combined display



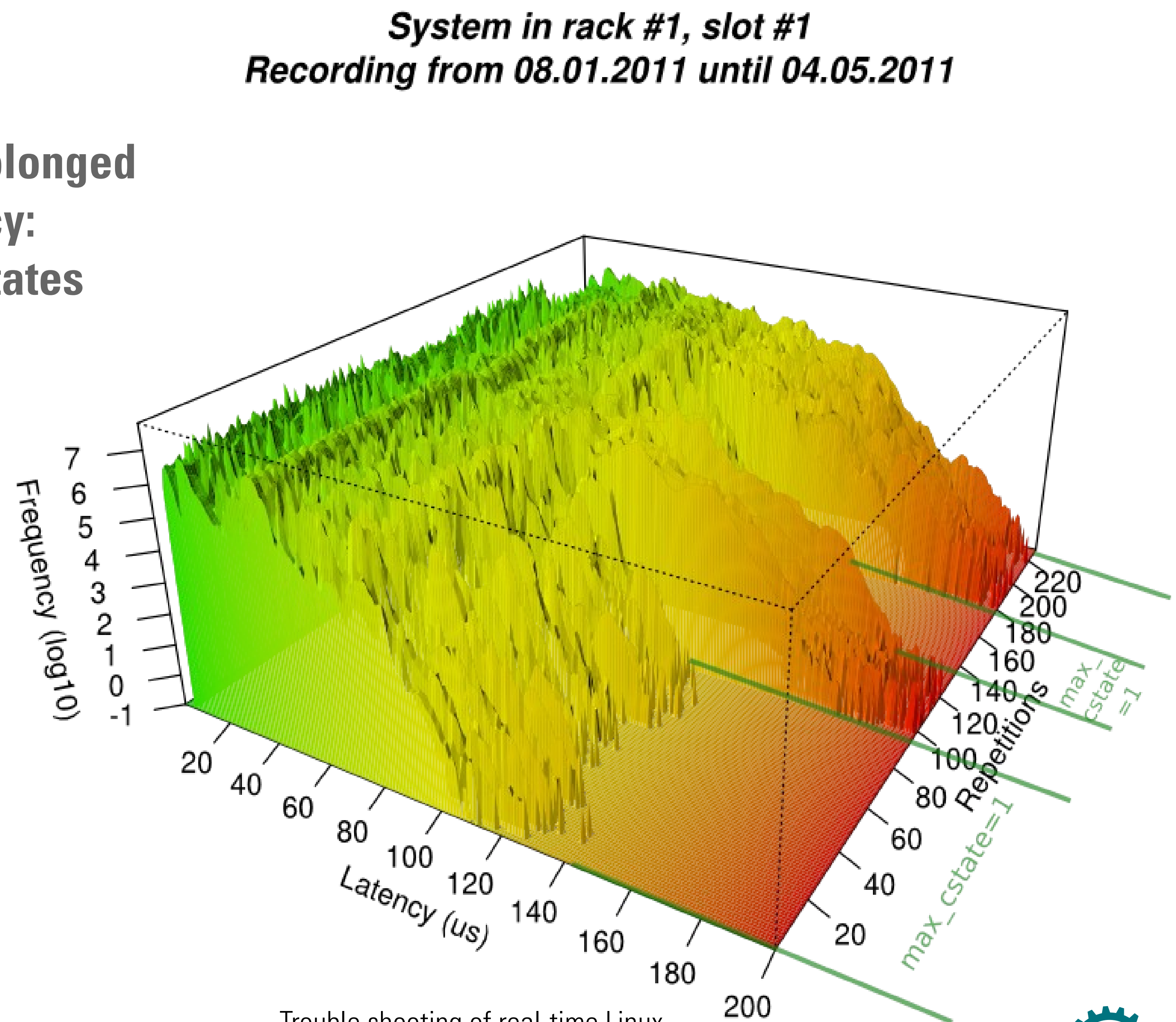
Example 5a: Real-time optimization

Periods with prolonged maximum latency



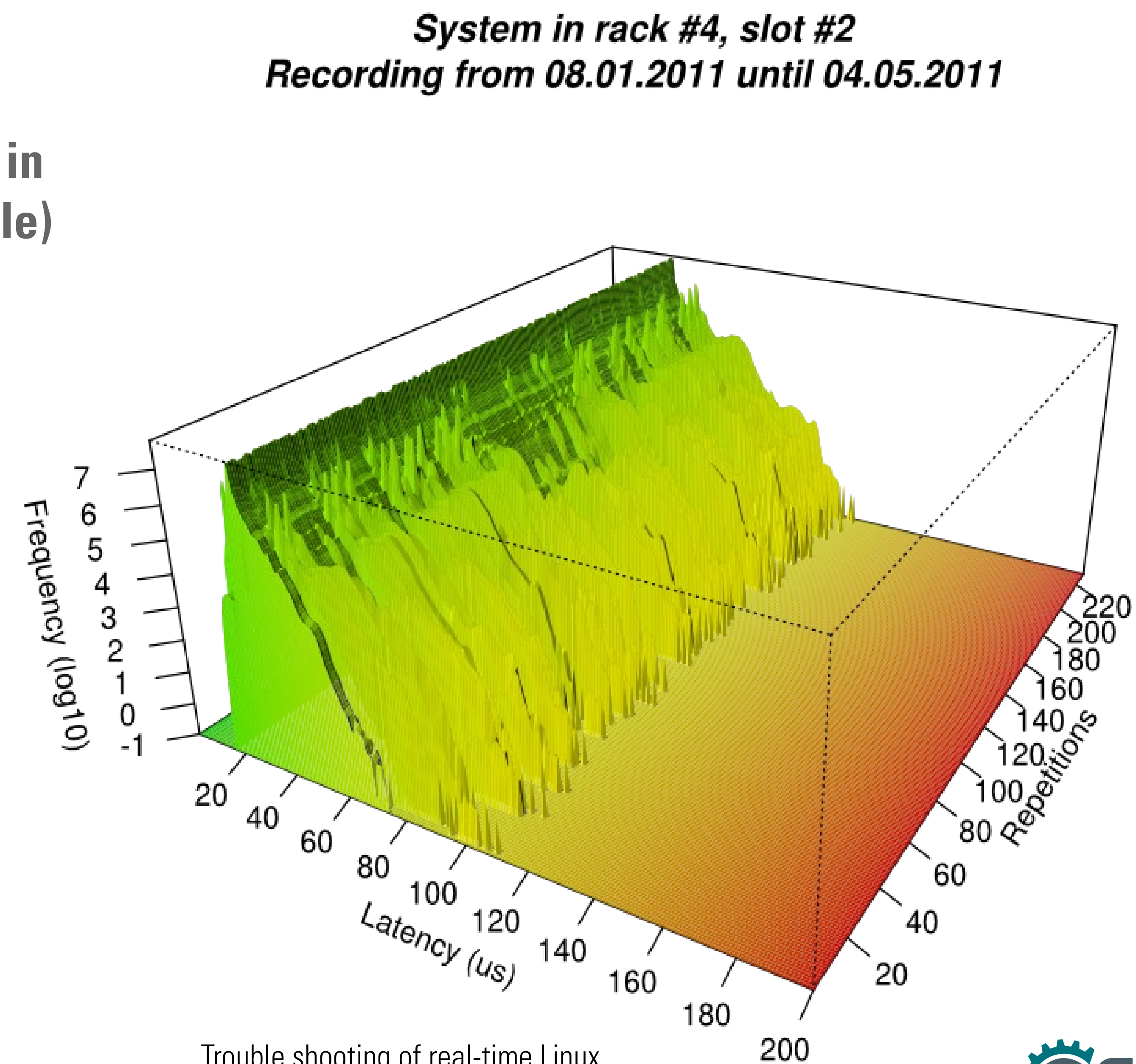
Example 5b: Real-time optimization

Periods with prolonged maximum latency: Enabled sleep states



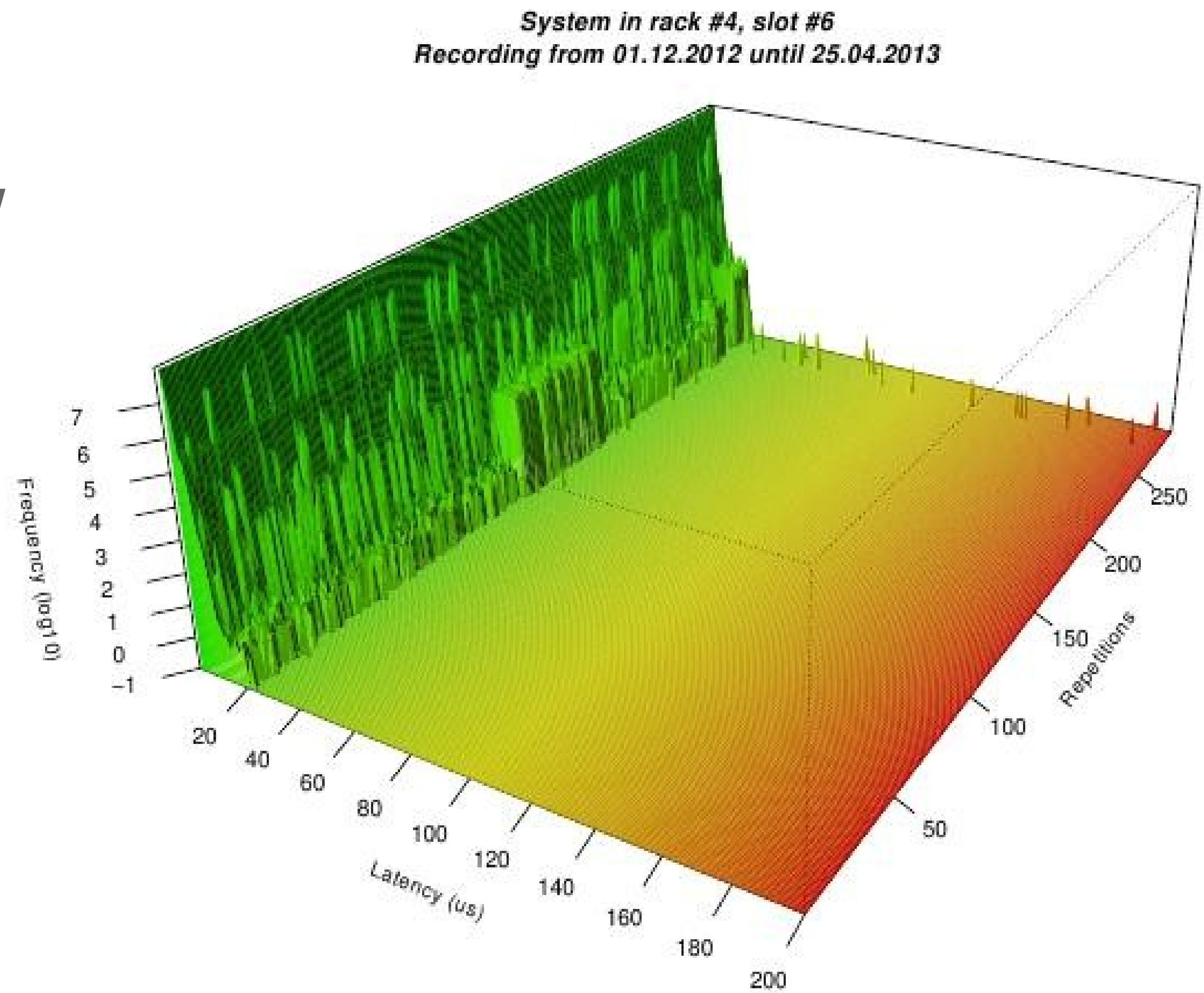
Example 6: Real-time optimization

Determinism (no outlier in more than 22 billion cycle)

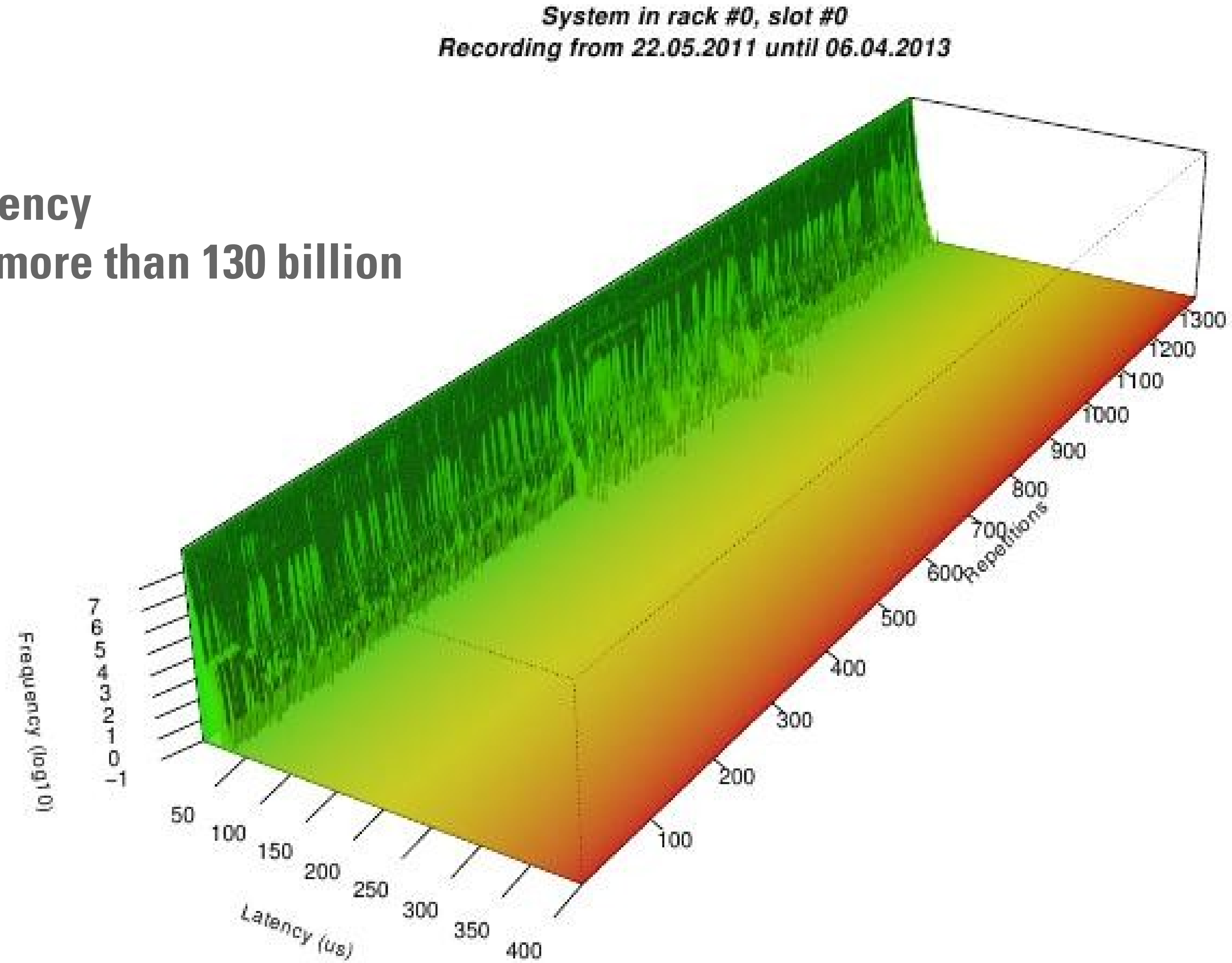


Example 7: Long-term latency plot

Very short latency



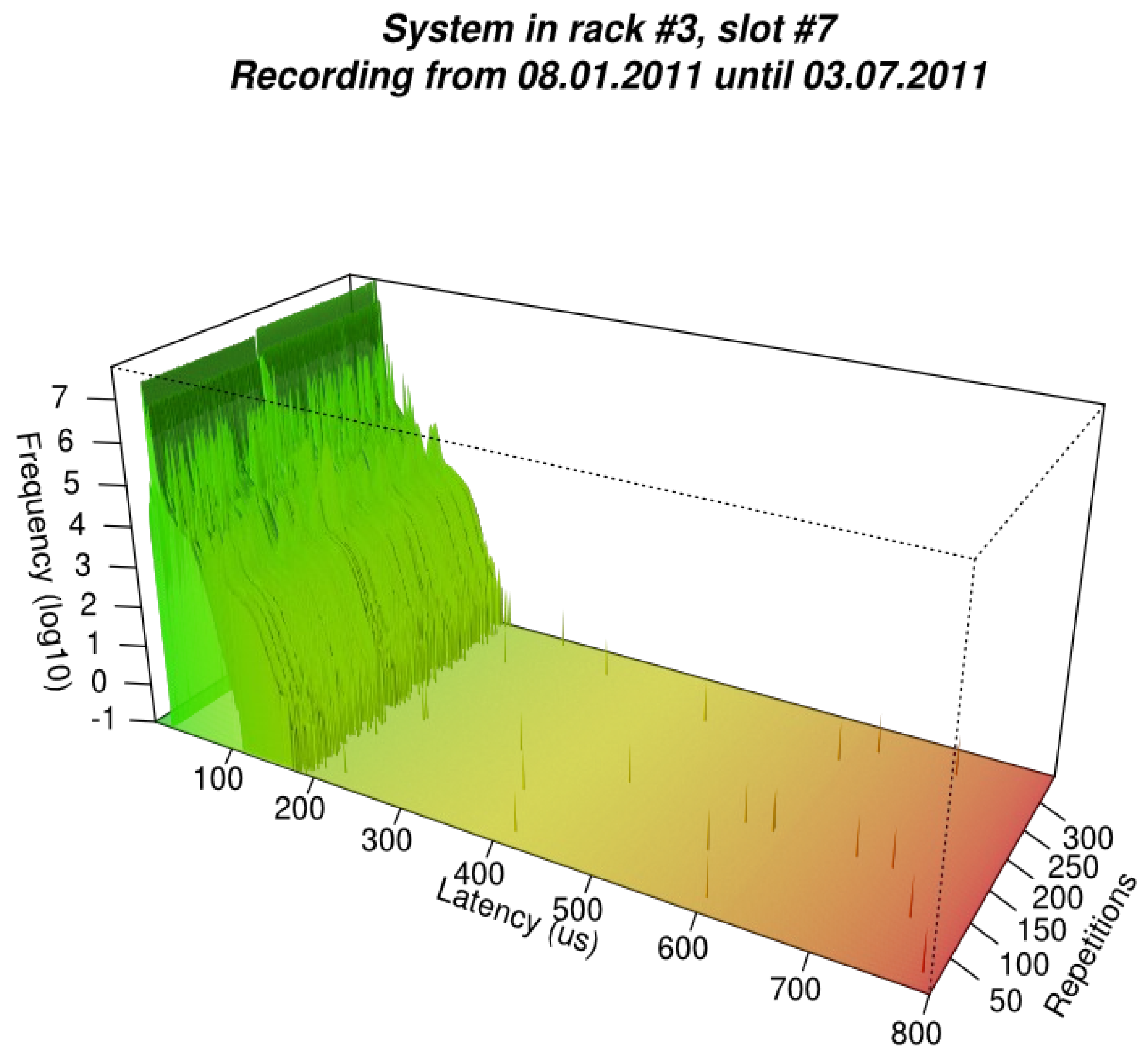
Example 8: Long-term latency plot



Very short latency
No outlier in more than 130 billion cycles

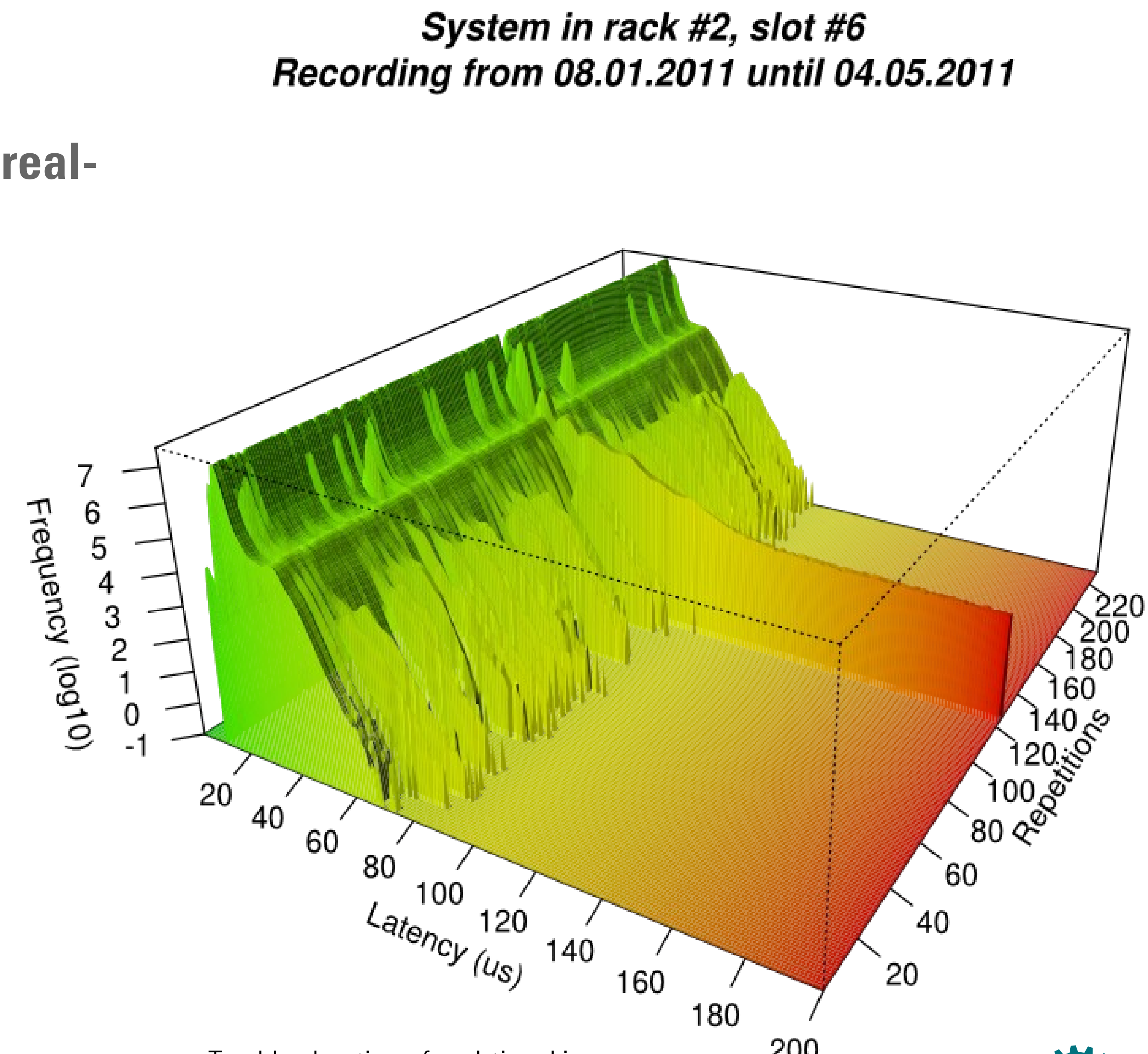
Example 9: Long-term latency plot

Sporadic outliers due to a DMA problem of the Ethernet controller



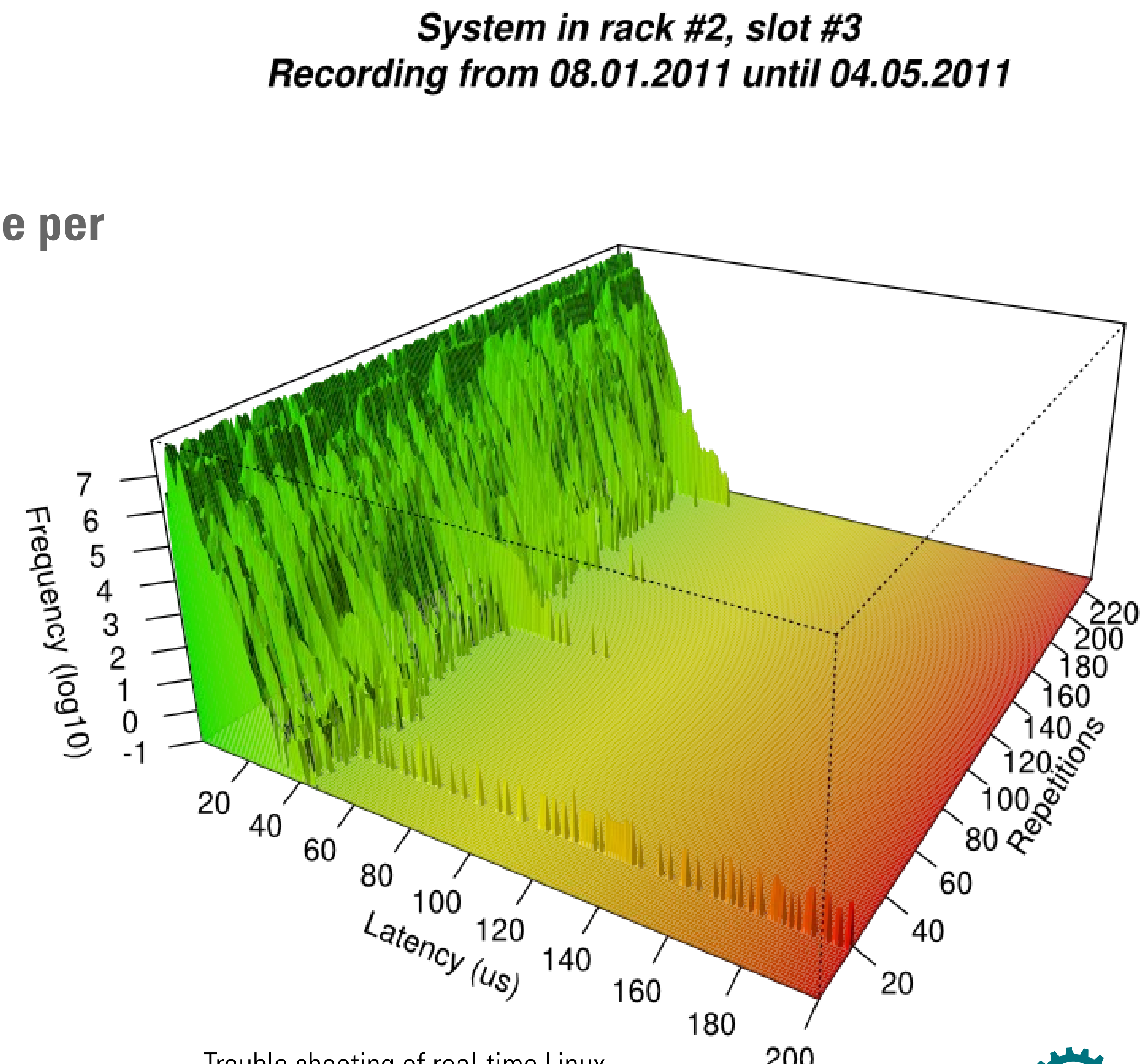
Example 10: Long-term latency plot

Erroneous use of a non-real-time kernel



Example 11: Long-term latency plot

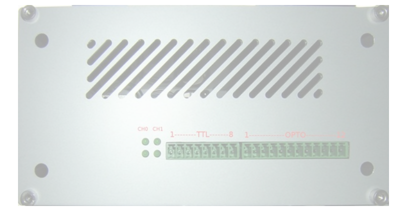
Highest system priority assigned more than once per core



Four levels of latency tests

External measurement with simulation

OSADL's „Latency-Box“



Internal continuous recording

Built-in kernel latency histograms

```
CONFIG_WAKEUP_LATENCY_HIST=y
CONFIG_INTERRUPT_OFF_HIST=y
CONFIG_PREEMPT_OFF_HIST=y
```

Internal measurement with simulation

Cyclictest

```
# cyclictest -a -t -n -p99
```

Real-world internal measurement

Application

```
# <application>
```