**O**pen
**S**ource
**A**utomation
**D**evelopment
**L**ab eG

Open Source Automation Development Lab (OSADL) eG · Im Neuenheimer Feld 583 · 69120 Heidelberg · www.osadl.org · info@osadl.org

# *Die OSADL QA-Farm*

## *Ausgangspunkt*

Für den Einsatz von Embedded-Systemen in Industrieprodukten müssen diese eine Reihe von unverzichtbaren Voraussetzungen erfüllen. Neben Zuverlässigkeit und Langlebigkeit ist es erforderlich, dass bestimmte für einen individuellen Anwendungsfall festgelegte Funktionsdaten ausnahmslos eingehalten werden. Aus diesem Grunde wurde die OSADL QA-Farm entwickelt, auf der über 150 Computer-Systeme kontinuierlich überwacht werden. Die dabei gewonnenen Daten stehen den jeweiligen Unternehmen, von denen die Systeme stammen, zur Verfügung und dienen der gezielten Optimierung von Hardware und Software. Ein Teil der Daten wird darüber hinaus der Allgemeinheit zur Verfügung gestellt. Nicht zuletzt stellen diese kontrolliert gewonnenen und dokumentierten Daten aber auch eine Basis für die wissenschaftliche Bearbeitung von Fragestellungen dar, die im Zusammenhang mit der Eignung und dem Einsatz von Embedded-Systemen in der Industrie stehen.

## *Realisierung*

Die OSADL QA-Farm selbst besteht aus einer Anzahl an offenen Testracks (siehe Abbildung 1), die jeweils über acht Einschubetagen für Testtabletts verfügen. Die Racks können an beliebiger Stelle lokalisiert sein, einzige Voraussetzungen sind Stromversorgung und Internetzugang. Zur Zeit stehen OSADL Testracks an zwei verschiedenen Orten in Deutschland. Jedes Testrack ist mit mehreren zentralen Steuer- und Kommunikationseinheiten ausgerüstet. Dabei handelt es sich um

- 10/100/1000 Mb/s Netzwerk-Switch mit Port-Mirroring
- Fernsteuerung der Stromversorgungen mit individuellem Power-Metering
- Seriell-zu-Netzwerk-Adapter
- Seriell-zu-USB-Adapter
- USB-Hub
- KVM-Switch mit Netzwerkanbindung
- Server für Kernel-Crossentwicklung und als Peer für Netzwerklasterzeugung

Die Prüflinge werden auf speziellen Lochraster-Tabletts (siehe Abbildung 2) aufgebaut, die wahlweise in eines der acht Einschubetagen eingesetzt werden. Auf den Tabletts befinden sich jeweils eine Netz-Steckdose zur Stromversorgung sowie zwei RJ45-Buchsen für Ethernet und RS232, die nach dem Einsetzen in das Rack mit den jeweiligen rackseitigen Gegenstücken verbunden werden. Optional können ein VGA-Graphikanschluss sowie USB- bzw. PS/2-Tastaturanschlüsse genutzt werden. Danach ist der Prüfling auf dem Rack betriebsbereit.

*Abbildung 1: OSADL Testrack*

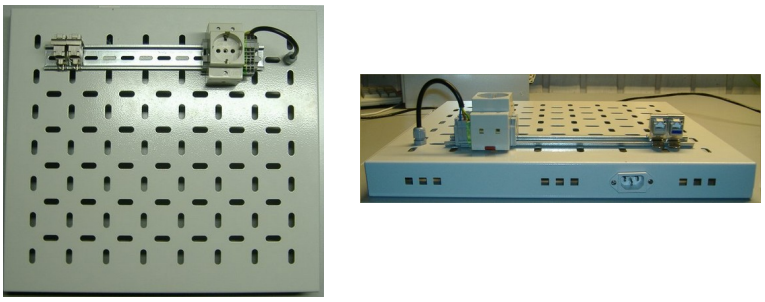

*Abbildung 2: Tablett zur Aufnahme der Prüflinge*

Zur Verwaltung, Speicherung und Visualisierung der Messdaten sind alle Testracks mit einem zentralen Auswerteserver über VPN-Kanäle verbunden (siehe Abbildung 3). An diesem Server können die Daten von überall auf der Welt über Standard-Webprotokoll abgefragt werden. Für die meisten Daten sind Alarmgrenzen definiert, deren Wichtigkeit in den beiden Kategorien „Warnung" und „Kritisch" unterteilt sind. Wird eine dieser Alarmgrenzen erreicht, erfolgt die Benachrichtigung von vorher festgelegten Personen mit Hilfe eines Eskalationssystems. Als Kommunikationswege stehen E-Mail, SMS, Fax und Voice-Nachricht zur Verfügung. Außerdem sind die betroffenen Variablen im Web-Interface gelb für Warnlevel bzw. rot bei kritischen Werten unterlegt.

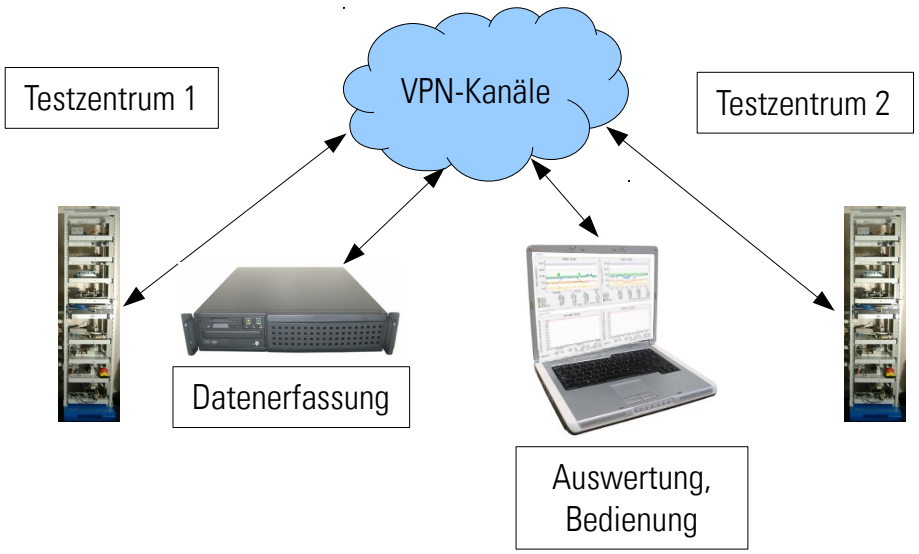


*Abbildung 3: Kommunikation der verschiedenen Komponenten der OSADL QA-Farm untereinander*

*Hardware im Test*

Die in der OSADL QA-Farm eingestellten Systeme stammen entweder von OSADL-Mitgliedsunternehmen, die an Stabilitäts- und Zuverlässigkeitsdaten der von ihnen verwendeten Hardware interessiert sind, oder werden von OSADL selbst für die Linuxkernel-Entwicklung verwendet. Je nach Absprache werden aber auch bei vielen der von Mitgliedsunternehmen eingestellten Systeme regelmäßig neue Kernelversionen aufgebracht und ge-

testet. Dies gibt den Unternehmen die Sicherheit, dass jederzeit ein Update auf eine neuere Kernelversion möglich ist, ohne dass vorher noch umfangreiche Entwicklungs- und Testarbeiten erforderlich sind. Entsprechend der Verwendung in der Industrie kommen Systeme der Prozessorfamilien ARM, MIPS, PowerPC und x86 zum Einsatz. Dabei handelt es sich sowohl um aktuelle Prozessoren, die gerade erst auf den Markt gekommen sind, als auch um bis zu 20 Jahre alte Designs. Letzteres ist deshalb wichtig, weil Industriesysteme durchaus eine derart lange Lebensdauer aufweisen können und es auch noch nach dieser Zeit problemlos möglich sein muss, einen aktuellen Linuxkernel zu installieren. Eine Auswahl der in der OSADL QA-Farm untersuchten Prozessoren findet sich in Tabelle 1. Bei der Auswahl an von OSADL eingestellten Systemen wurde darauf geachtet, dass eine möglichst große Bandbreite an Taktfrequenzen, Speichergröße und Prozessor-Topologien verfügbar ist. So reichen zum Beispiel die Taktfrequenzen von 133 bis 4200 MHz und der Speicherausbau von 26 MByte bis 128 GByte. Neben Singlecore-Prozessoren früherer Generationen sind Multicore-Prozessoren mit bis zu 32 Kernen sowie gemischte Multicore-Systeme mit mehreren Sockeln im Einsatz. Auch bei der Auswahl der Chipsätze und Peripheriegeräte wurde darauf geachtet, dass möglichst unterschiedliche Kontroller verschiedener Hersteller Verwendung finden.

## ARM

Broadcom
- BCM2708 @700 MHz, 32 bit
- BCM2709 @900 MHz, 32 bit
- BCM2835 @1400 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
- i.MX6UL @528 MHz, 32 bit

Marvell
- SheevaPlug @1200 MHz, 32 bit

Texas Instruments
- AM3517 @600 MHz, 32 bit
- AM437x @1000 MHz, 32 bit
- AM5728 @1000 MHz, 32 bit
- OMAP3525 @720 MHz, 32 bit
- OMAP4430 X2 @1008 MHz, 32 bit
- OMAP4460 X2 @1200 MHz, 32 bit

Xilinx
- Zync @666 MHz, 32 bit

## MIPS

ICT
- Loongson 2F @800 MHz, 64 bit

## PowerPC

Freescale
- MPC 5200 @396 MHz, 32 bit
- MPC8349 @533 MHz, 32 bit
- QorIQ-2020 @1200 MHz, 32 bit
- e5500 T1040RDB @1200 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
- FX-8150 X8 @3600 MHz, 64 bit
- Embedded Kaveri @2700 MHz, 64 bit
- Ryzen Threadripper 1950X @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
- Core i7-3770 @3400 MHz, 64 bit
- Xeon E3-1220L V2 @2300 MHz, 64 bit
- Xeon X5650 @2670 MHz, 64 bit
- Core i7-4960X 3600 MHz, 64 bit
- Core i3-6100U @2300 MHz, 64 bit
- Core i5-6442EQ @1900 MHz, 64 bit
- Core i7-7700K @4200 MHz, 64 bit
- Core i7-7828EQ @3000 MHz, 64 bit

VIA
- C3 Samuel 2 @533 MHz, 32 bit
- C7 @1000 MHz, 32 bit
- Nano X2 L4050 @1400 MHz, 64 bit
- QuadCore L4700 @1200 MHz, 64 bit

*Tabelle 1: Hersteller und Architekturen von Systemen in der OSADL QA-Farm (Auswahl)*

*Gemessene Variablen*

Die in der OSADL QA-Farm gemessenen Variablen lassen sich in die Bereiche Benchmark, Disk, Netzwerk, NFS, Prozesse, Real-time System, E-Mail, Sensoren, Zeitsynchronisation, System und Virtualisierung einteilen. Eine Übersicht ausgewählter Variablen findet sich in Tabelle 2.

**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

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

**NFS**
- NFS Client
- NFSv4 Client

**Prozesse**
- 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

**Sensoren**
- Fans
- HDD temperature
- Power consumption
- Temperatures

**Zeitsynchronisation**
- 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

**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

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

*Tabelle 2: Gemessene Variablen von Systemen in der OSADL QA-Farm (Auswahl)*

*Variablen mit besonderer Bedeutung für den Einsatz von Embedded-Systemen in der Industrie*

Besondere Bedeutung für den Einsatz von Embedded-Systemen in der Industrie haben Variablen, die im Zusammenhang stehen mit der Reaktionszeit des Systems auf asynchrone externe und interne Ereignisse (Echtzeitfähigkeit) sowie mit Temperaturverlauf und Stromaufnahme bei verschiedenen Lastszenarien. Ferner sollten zu Vergleichszwecken die jeweiligen Leistungsparameter von CPU, FPU und GPU bestimmt und registriert werden.

Nicht zuletzt ist es erforderlich, die Versions- und Releasenummern des Linuxkernels aufzuzeichnen, damit im Falle plötzlicher Veränderungen überprüft werden kann, ob es sich um eine Regression bei Kernel-Upgrade handelt. Als Beispiele für die Aufzeichnung dieser Variablen werden im Folgenden die Registrierungen von Reaktionsverzögerung als Maß der Echtzeitfähigkeit sowie von Temperaturverlauf, Lüfterdrehzahl, Taktfrequenz, Schlafstadien, Stromaufnahme und Uptime dargestellt. Außerdem wird ein Beispiel für die Registrierung der Version des Linuxkernels gezeigt. Schließlich findet sich im Folgenden auch ein Beispiel für eine spezielle Teststrecke, die nur bei genau für diesen Zweck eingebrachten Systemen installiert ist.

*Reaktionsverzögerung (Echtzeitfähigkeit)*

Kontinuierlich aufgezeichnete Präemptionslatenz:
Immer wenn ein Echtzeitprozess infolge eines abgelaufenen Timers gestartet werden soll, wird die programmierte Weckzeit mit der tatsächlichen Weckzeit verglichen und die Differenz als Timerverzögerung in einem Histogramm gespeichert. Bei Mehrkern-Prozessoren erfolgt die Aufzeichnung der Werte jedes einzelnen Kerns. Bei Prozessoren mit Energiespar-Mechanismen kann die Aufwachverzögerung vom jeweiligen Energiesparmodus abhängig sein wie in Abbildung 4 zu sehen ist. Dargestellt ist das aus dem Histogramm ermittelte Maximum der Timerverzögerung über konsekutive Messperioden von jeweils 5 Minuten Dauer.

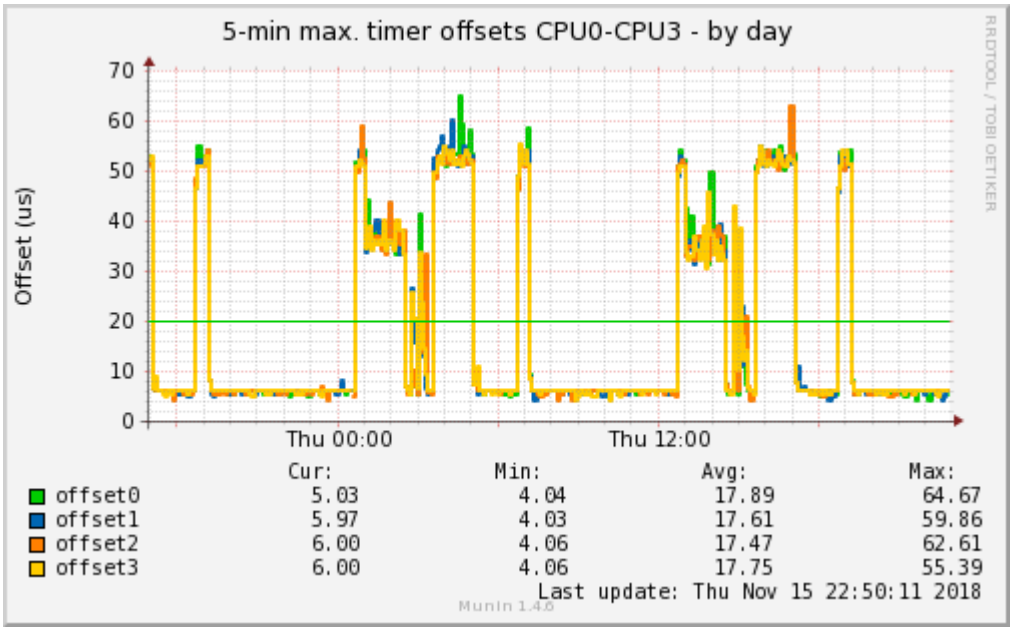


*Abbildung 4: Kontinuierlich aufgezeichnete Timerverzögerung über 30 Stunden*

Nach der Timerverarbeitung wird der Echtzeitprozess vom Scheduler zur Ausführung gebracht, und es erfolgt der sogenannte Kontext-Switch, mit dem in den Userspace verzweigt und die Ausführung des Echtzeitprozesses fortgesetzt wird. Die Zeitdifferenzen zwischen der tatsächlichen Weckzeit und dem Ende des Kontext-Switch werden ebenfalls ermittelt, in einem Histogramm gespeichert und deren Maximum in Intervallen von 5 Minuten graphisch dargestellt. In einem dritten Histogramm wird die jeweilige Summe von Timerverzögerung und Scheduler-Verzögerung abgespeichert und in gleicher Weise verarbeitet und angezeigt (Abbildung 5). Diese Summe entspricht sehr weitgehend der sogenannten Präemptionslatenz und stellt ein wichtiges Maß für die Echtzeitfähigkeit dar. Der Vorteil dieser Methode liegt darin, dass die Aufzeichnung kontinuierlich und unabhängig von der Systemintegration erfolgen kann. Bei einer Anomalie des Systems kann zudem retrospektiv analysiert werden, ob zum Zeitpunkt der Anomalie eine ungewöhnliche Präemptionslatenz aufgetreten ist.

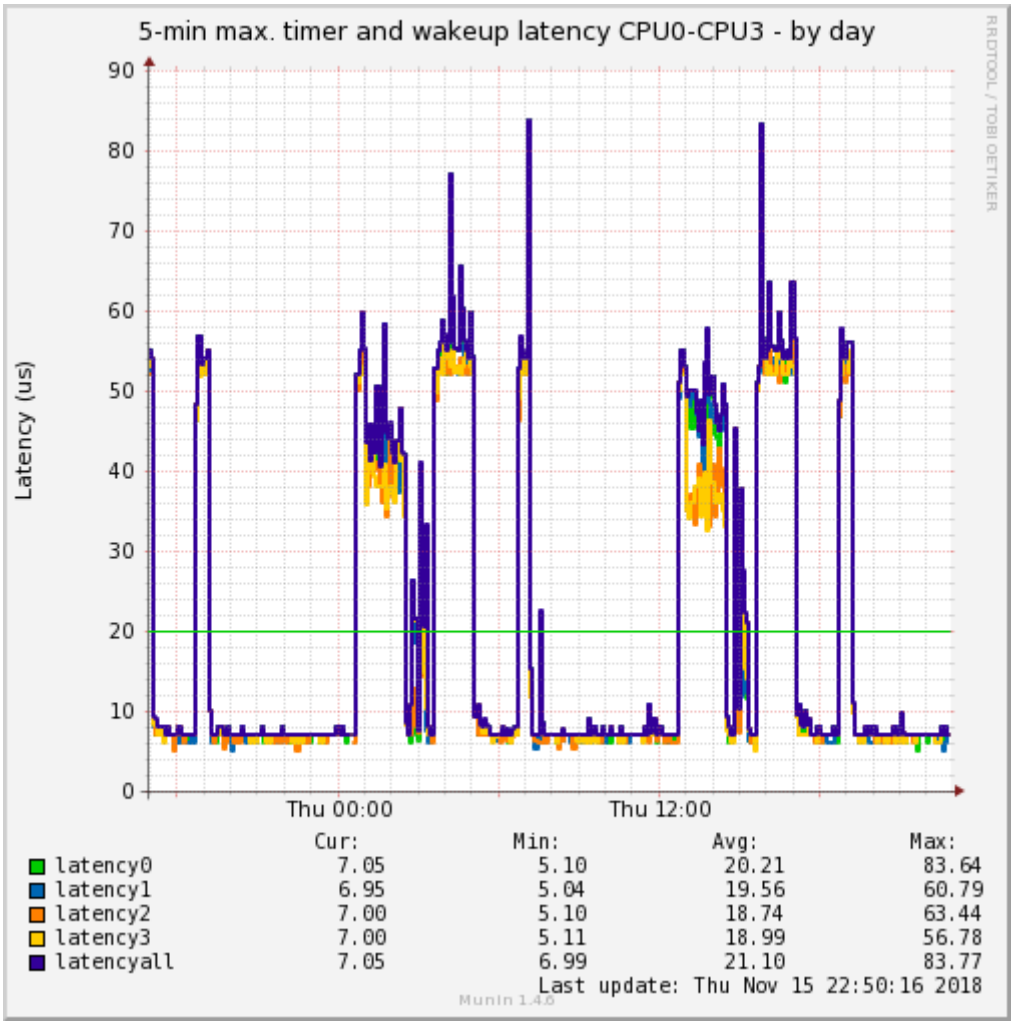


*Abbildung 5: Kontinuierlich aufgezeichnete Summe von Timer- und Aufwachverzögerung als Maß für die Präemptionslatenz über 30 Stunden*

Stimulierte Präemptionslatenz:
Weiterhin werden für jeweils zweimal fünfeinhalb Stunden pro Tag Timer-Interrupts mit einer Frequenz von 5 kHz erzeugt, und es wird ebenfalls die Verzögerung zwischen programmiertem und tatsächlichem Aufwachzeitpunkt – diesmal allerdings direkt im Userspace – gemessen und registriert. Dabei erfolgen 100 Millionen Aufwachzyklen pro Prozessorkern; die längste dabei jemals gemessene Präemptionslatenz gilt wiederum als Maß für die Echtzeitfähigkeit des Systems. Für die Darstellung des Ergebnisses einer Messung wird das Histogramm mit allen 100 Millionen Werten verwendet, wobei eine lineare x-Achse und eine logarithmische y-Achse gewählt werden, damit auch sehr selten gemessene Werte abgelesen werden können. Dies ist wichtig, da die längste jemals gemessene Aufwachverzögerung in der Regel nur wenige Male oder auch sogar nur ein einziges Mal auftritt. Beim in Abbildung 6 wiedergegebenen Beispiel einer solchen Messung mit 100 Millionen Zyklen ist dies der Fall; die maximale Gesamtverzögerung des Systems in Höhe von 55 µs wurde nur ein einziges Mal registriert. Dieser Wert gilt als wichtigstes Ergebnis der Messung, mit der die Echtzeitfähigkeit des Systems klassifiziert wird. Um eine hohe statistische Sicherheit dieses Messergebnisses zu erreichen, werden die einzelnen Histogramme einer gemeinsamen Auswertung zugeführt. Dabei wird wiederum die längste jemals gemessene Aufwachverzögerung bestimmt – jetzt aber nicht von einer einzelnen Messung sondern von vielen tausend konsekutiven Messperioden. Bei zum Beispiel zwei Messungen pro Tag und insgesamt 2000 Messperioden erstreckt sich der Messzeitraum über fast drei Jahre. Um eine Übersicht über alle Messergeb-

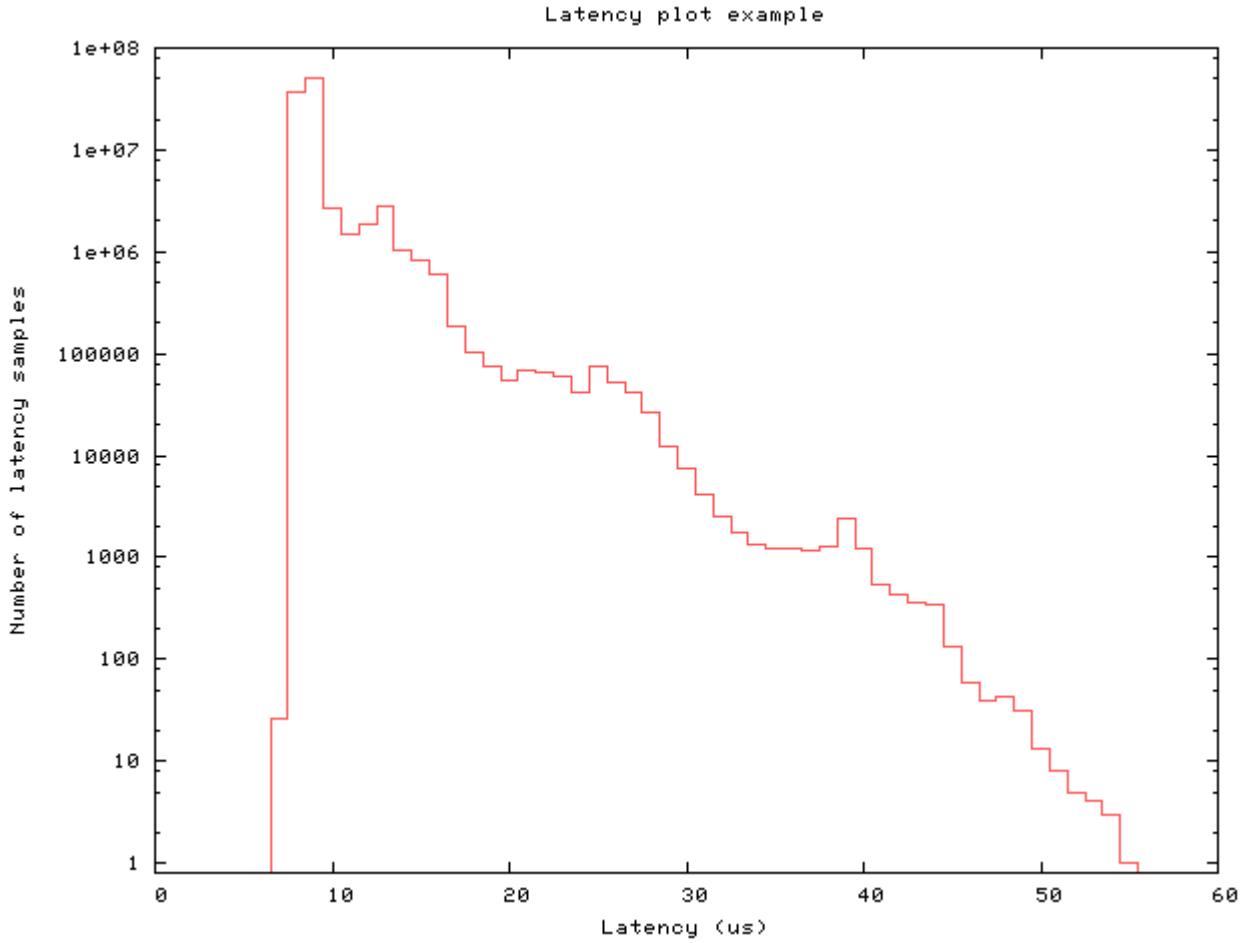


*Abbildung 6: Beispiel für Messergebnis einer stimulierten Latenzmessung (5 kHz, 100 Millionen Messzyklen)*

nisse zu ermöglichen, werden die Histogramme graphisch hintereinander gestellt und in Pseudo-3D-Technik visualisiert. Auch bei dieser Darstellung wird eine logarithmische y-Achse gewählt, damit bereits ein einzelner Ausreißer in Form einer dünnen Nadel sichtbar gemacht werden kann. Das in Abbildung 7 gezeigte Beispiel stammt von einem Embedded-System mit einem fehlerhaften Netzwerk-Kontroller. Dieser Fehler führt dazu, dass sehr seltene Aufwachverzögerungen auftreten – und zwar in einer Häufigkeit von etwa 65 Ausreißer auf 650 Milliarden Messzyklen. Selbst diese geringe Fehlerrate von etwa 0,0001 ppm ist absolut inakzeptabel und bedeutet, dass ein derartiges System niemals für eine industrielle Steuerung verwendet werden kann und darf. Dabei muss besonders darauf hingewiesen werden, dass während der über siebenjährigen Beobachtungszeit mehrfach Perioden von bis zu fast 20 Tagen Dauer aufgetreten sind, in denen es zu überhaupt keiner Verlängerung der Aufwachverzögerung gekommen ist. Diese Beispielmessung stellt somit ein weiteres wichtiges Argument für die Relevanz der mit der OSADL QA-Farm vorgenommenen Messungen über extrem lange Zeiträume dar; denn bei einer üblichen höchstens einige Stunden dauernden Labormessung wäre dieser Fehler mit hoher Wahrscheinlichkeit nicht aufgetreten und daher übersehen worden. Allerdings sind derartig fehlerhafte Embedded-Systeme eine seltene Ausnahme. Die meisten gemessenen Systeme weisen keinerlei Ausreißer auf. Wenn derartige Messergebnisse ohne Ausreißer über einen sehr langen Zeitraum bei einer Systembelastung wie unter Produktivbedingungen erhoben werden können, dann kann dies als Hinweis darauf betrachtet werden, dass solche Systeme mit Recht als deterministische Systeme bezeichnet werden können. (z.B. Abbildung 8).

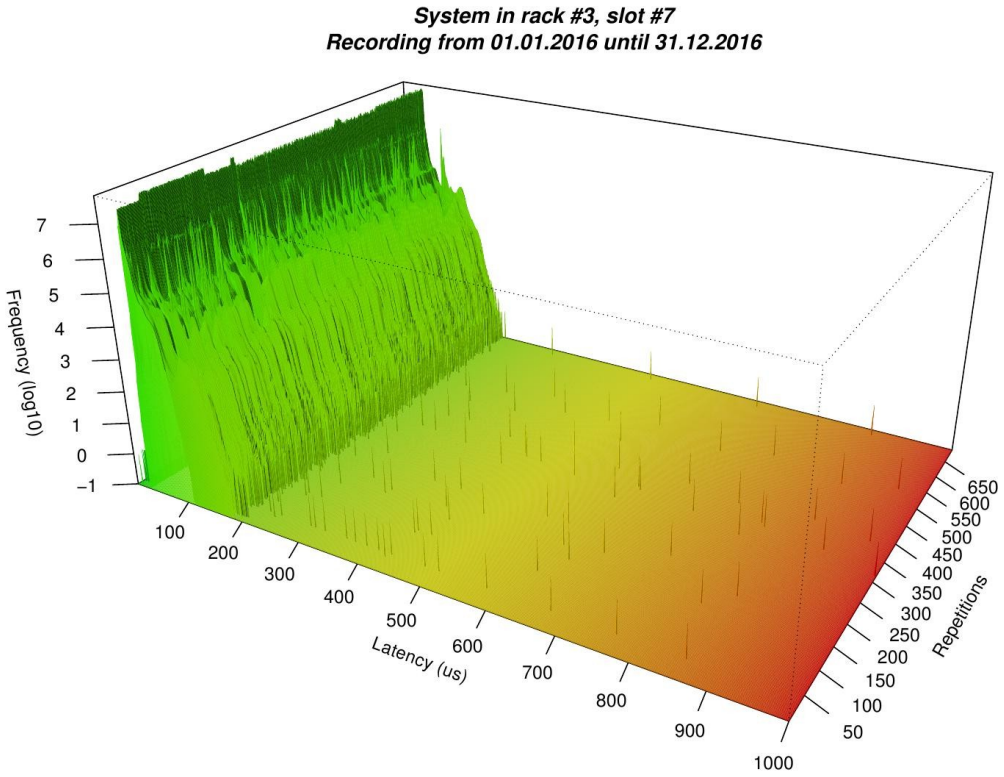


*Abbildung 7: Konsekutive Latenzhistogramme mit vielen Ausreißern bei fehlerhaftem Embedded-System*

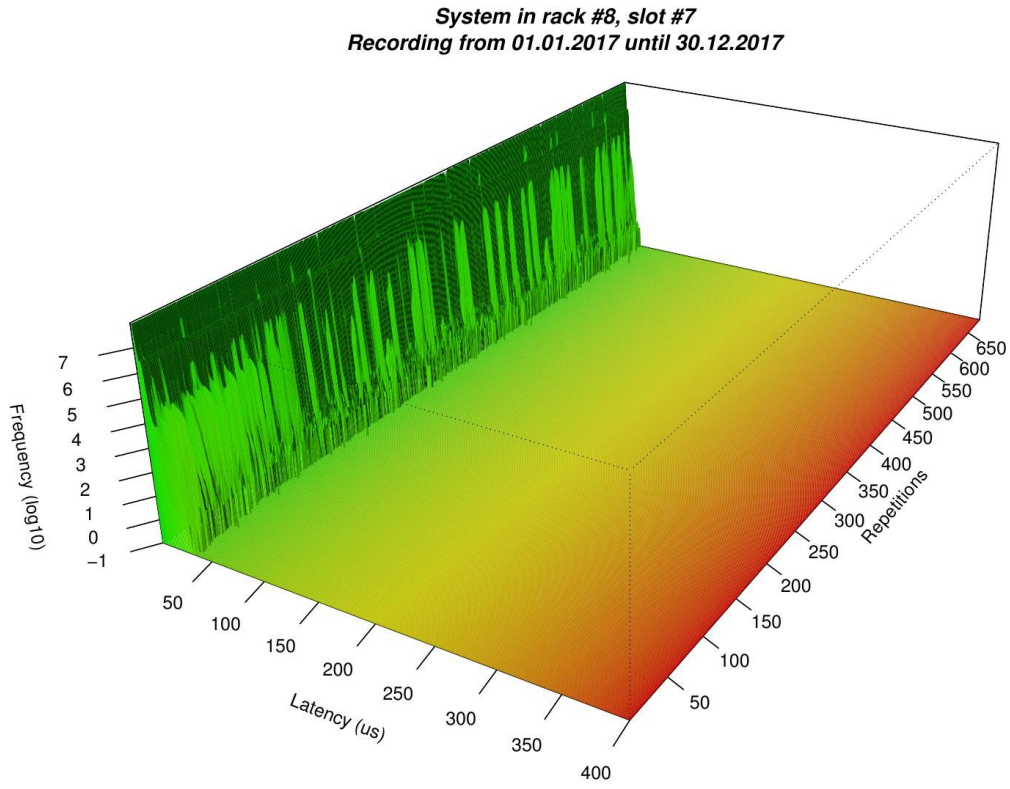


*Abbildung 8: Deterministisches Antwortverhalten eines für industrielle Steuerungsaufgaben ideal geeigneten Embedded-Systems*

*Temperaturverlauf der verschiedenen Systemkomponenten*

Moderne Embedded-Systeme sind mit einer Vielzahl an Sensoren ausgerüstet, mit denen die Temperatur vieler Systemkomponenten sowie Versorgungsspannungen, Lüfterdrehzahl und neuerdings auch Energieaufnahme gemessen wird. Für die Langzeitstabilität und die Lebensdauer von Embedded-Systemen ist es wichtig, die jeweilige Temperatur zu kennen und zu minimieren. Diese Messung ist aber auch wichtig, um im Fehlerfall eine – direkte oder indirekte – thermische Ursache von anderen Ursachen unterscheiden zu können. Bei gleichzeitiger Messung von Temperatur und Lüfterdrehzahl kann man darüber hinaus überprüfen, ob eine eventuell vor-

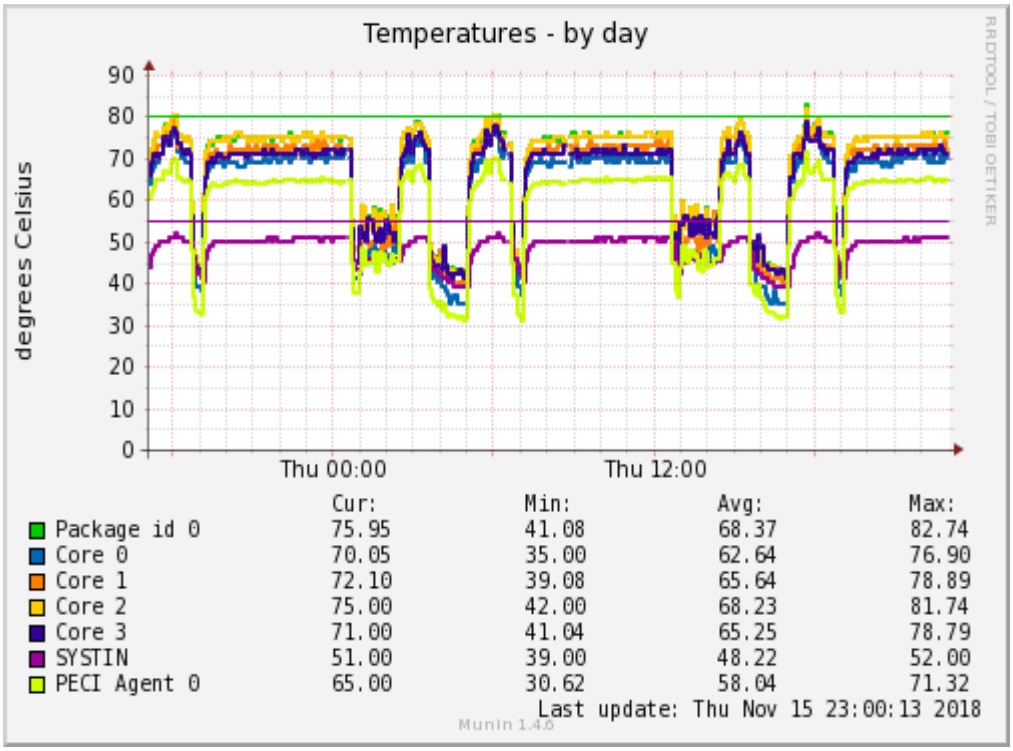


*Abbildung 9: Temperaturverlauf eines Mehrkernsystems über 30 Stunden unter Last- und Ruhebedingungen*

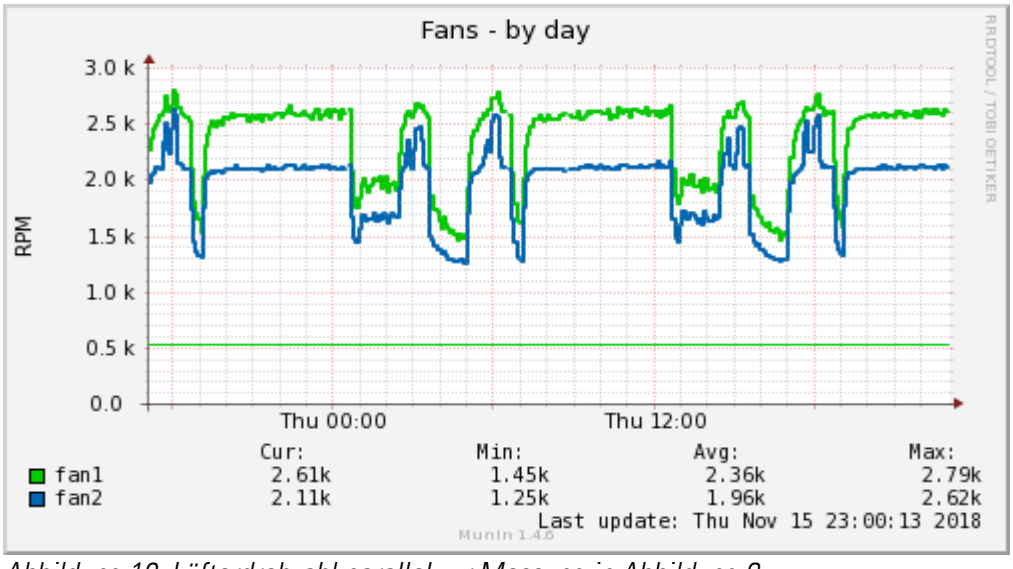


*Abbildung 10: Lüfterdrehzahl parallel zur Messung in Abbildung 9*

handene Lüfterregelung korrekt funktioniert oder nicht. Fehlerfreie Funktion wird dadurch angezeigt, dass die Lüfterdrehzahl proportional zur Temperatur ansteigt. Steigt die Temperatur trotz steigender Lüfterdrehzahl stärker als erwartet, ist vermutlich das Staubfilter verstopft; bleibt die Lüfterdrehzahl konstant, ist die Regelung defekt. In jedem Fall muss eine entsprechende Alarmierung ausgelöst werden, da anderenfalls das System wegen Überhitzung zu Schaden kommen kann. Beispielhafte Verläufe von Temperatur (Prozessor, Motherboard usw.) und Lüfterdrehzahl sind in den Abbildungen 9 und 10 wiedergegeben. Deutlich ist die funktionsfähige Lüfterregelung daran zu erkennen, dass die erhöhte Temperatur der CPU-Kerne zu einer Erhöhung der Lüfterdrehzahl und damit zu einer Reduktion der Temperatur des Motherboards führt. Gleichwohl ist aber die Ventilatorleistung des Lüfters nicht ausreichend, um jeglichen Temperaturanstieg der CPU-Kerne zu verhindern, und es kommt sporadisch zu einem durchaus nicht unerheblichen Temperaturanstieg bis kurz über die Warngrenze von 80 °C. Die Phasen erhöhter Temperatur sind auf spezielle Stress-Szenarien zurückzuführen, während denen das System eine Reihe von definierten Lastzyklen absolvieren muss.

*Versionsnummern des Linuxkernels*

Neben der jeweils installierten Version des Linuxkernels (Major-Nummer) wird der Patchlevel (Minor-Nummer) und der Sublevel des Stable-Trees aufgezeichnet. Bei Echtzeitkerneln erfolgt zusätzlich die Registrierung des RT-Release. Als Beispiel findet sich in Abbildung 11 eine 13-monatige Registrierung der Version des Linuxker-

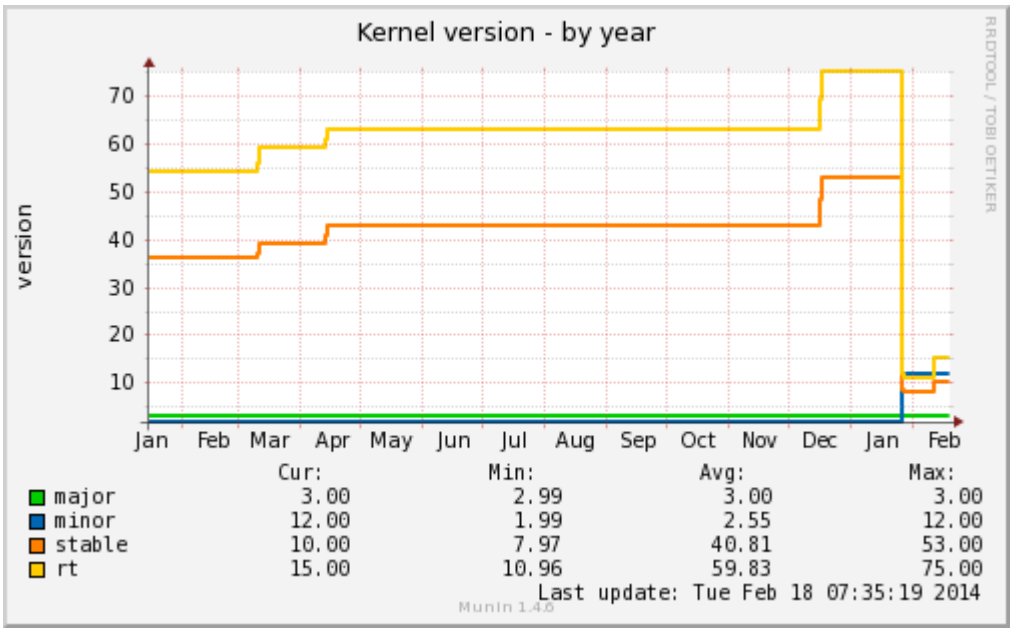


*Abbildung 11: Verlauf der Versionsnummern des Linuxkernels während 12 Monaten*

nels von linux-3.0.37-rt54 bis linux-3.12.10-rt15. Deutlich sind die Zeitpunkte und Versionssprünge der jeweiligen Kernel-Upgrades zu erkennen. Diese Aufzeichnung dient in erster Linie dazu, um sogenannte „Regressions", d.h. ungewollte Verschlechterungen irgendwelcher Leistungsparameter, zu erkennen. Sollte dies einmal der Fall sein, kann im Prinzip mit Hilfe der aufgezeichneten Versionsnummern vor und nach der Verschlechterung eine Ursachenanalyse vorgenommen werden, und es kann die fehlerhafte Komponente des Upgrade korrigiert oder rückgängig gemacht werden. Dass durchaus erhebliche Veränderungen in der Leistungsfähigkeit einzelner Baugruppen vorkommen können, zeigt Abbildung 12, in der die Ergebnisse der zweimal am Tag vorgenommenen 2D-Leistungsmessung der akzelerierten Graphik wiedergegeben sind. Deutlich erkennt man eine erhebliche Leistungssteigerung ab etwa 12. Mai, die offensichtlich durch Userspace-Komponenten bewirkt wurde. Diese Leistungssteigerung wird durch einen Kernel-Upgrade bzw. durch dabei vorgenommene Veränderun-

gen praktisch komplett zunichte gemacht. Der weitere Kernel-Upgrade Ende Januar führt nur zu einer Wiederherstellung der ehemaligen Performance der Textausgabe, während alle anderen Tests weiterhin die niedrige Performance aufweisen. Vermutlich sind in diesem speziellen Fall die Versionssprünge des Kernels zu groß und die hier aufgezeichneten Daten nicht ausreichend, um die Ursache dieser erheblichen Regression zu finden. Aber es ist erkennbar, dass die parallele Aufzeichnung von Kernel-Versionsnummern und Kenndaten der Systemleistung eine komplette Regressionsüberwachung der Kernelentwicklung ermöglicht.

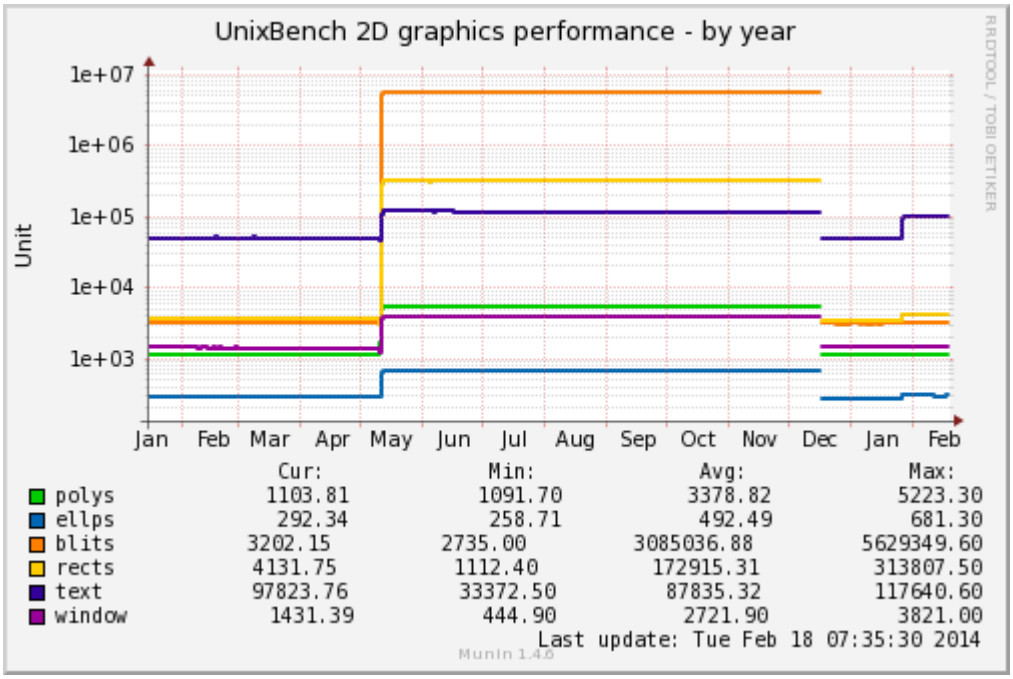


*Abbildung 12: Leistung der akzelerierten 2D-Graphik im Jahresverlauf*

*Taktfrequenz, Schlafstadien und Energieaufnahme*

Seit einigen Jahren ist das Interesse an Maßnahmen zur Energieeinsparung erheblich gestiegen und hat auch zu weitgehenden Bemühungen geführt, die Leistungsaufnahme von Embedded-Systemen zu reduzieren. Dies betraf zunächst CPU und Graphik-Prozessor (GPU), aber auch andere Komponenten mit relativ hohem Stromverbrauch wie Speicher und Busse bieten sich dafür an. Das wesentliche Prinzip besteht in allen Fällen darin, den Stromverbrauch in Abhängigkeit von der geforderten Leistung zu reduzieren bzw. nicht benötigte Systemkomponenten komplett abzuschalten. Für CPUs wurden sogenannte P-States und C-States eingeführt; erstere betreffen die stufenweise Reduzierung der Taktfrequenz des Prozessors, letztere beziehen sich auf sogenannte Schlafstadien. Daher werden in der OSADL QA-Farm sowohl Taktfrequenz und Schlafstadien der Prozessoren als auch die Energieaufnahme der Systeme kontinuierlich aufgezeichnet. In den Abbildungen 13 bis 15 sind die entsprechenden Verläufe dieser Variablen in einer 30-Stundenaufzeichnung miteinander in Beziehung gesetzt. Deutlich ist die erhöhte Energieaufnahme bei höherer Taktfrequenz und ausgeschalteten Schlafstadien (Polling) zu erkennen.

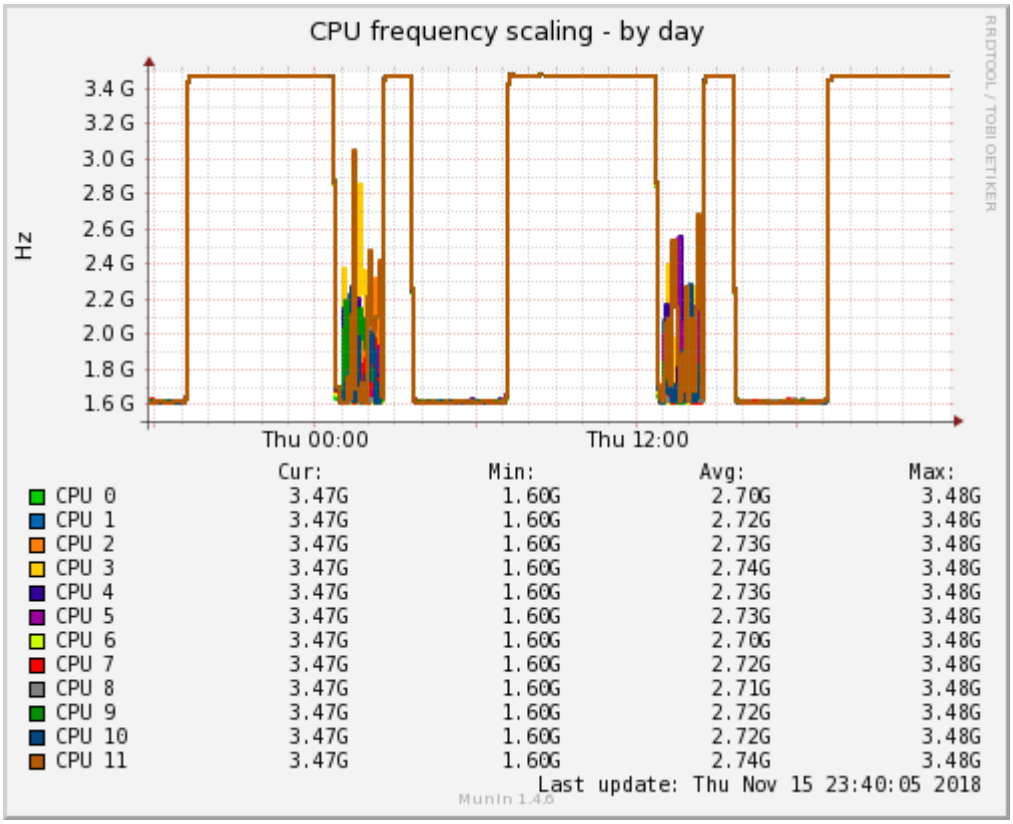


*Abbildung 13: Taktfrequenzen der zwölf Kerne eines Mehrkernsystems*

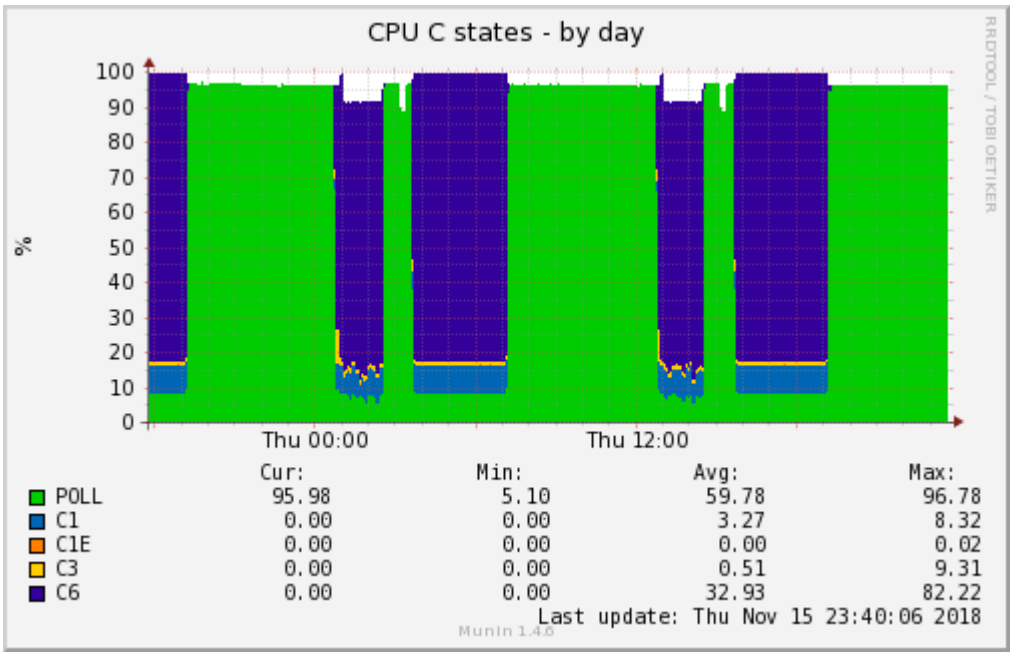


*Abbildung 14: Schlafstadien (alle Kerne gemittelt) des Systems in Abbildung 13*

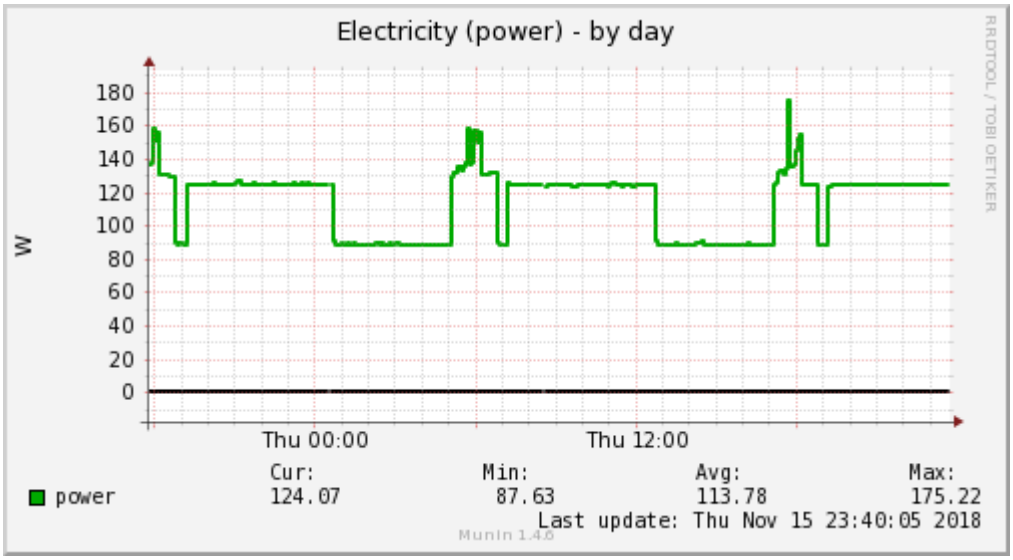


*Abbildung 15: Energieaufnahme des Systems in Abbildung 13*

*Systemstabilität*

Zur Erfassung der Systemstabilität wird die sogenannte Uptime, d.h. die Zeit in Tagen, die seit dem letzten Systemstart vergangen ist, aufgezeichnet. Bewusst herbeigeführten Neustarts werden dokumentiert, da diese nicht fälschlicherweise als Instabilität eines Systems gewertet werden dürfen. Abbildung 16 zeigt ein offensichtlich sehr stabiles System, da es über 700 Tage kontinuierlich in Betrieb ist und weder ein spontaner Neustart aufgetreten ist, noch aus irgendwelchen Gründen ein manueller Neustart erforderlich war.

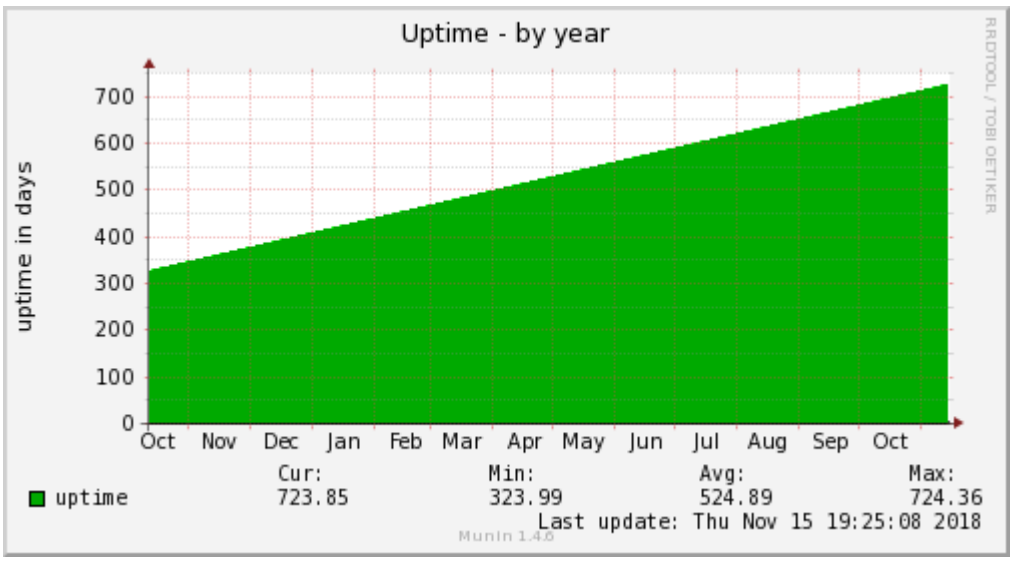


*Abbildung 16: Offensichtlich sehr stabiles System, das ohne Neustart mehr als 700 Tage kontinuierlich in Betrieb ist*

Im Gegensatz zum System in Abbildung 16 leidet das System in Abbildung 17 offensichtlich an häufigen Systemabstürzen, weswegen das System jeweils wieder neu gestartet werden muss. Mit Hilfe von Logdaten sowie speziellen Tools muss ermittelt werden, welche Ursache zu den häufigen Systemabstürzen führt.

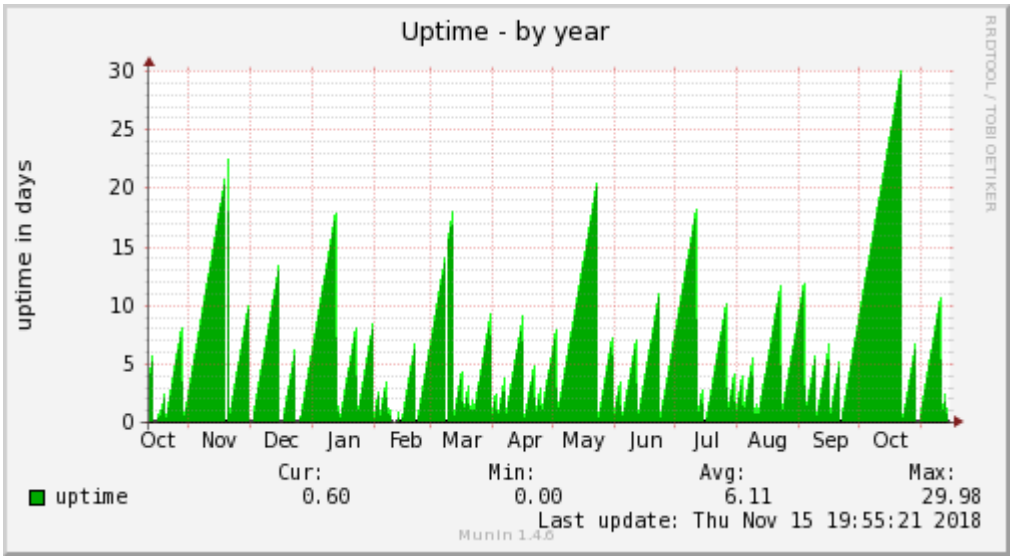


*Abbildung 17: Beispiel für ein instabiles System, das nur ein einziges Mal erst nach 30 Tagen und einige Male nach 20 Tagen wieder neu gestartet werden musste, ansonsten aber jeweils nach nur wenigen Tagen. Es handelt sich also um ein extrem instabiles System, das ohne Behebung der Ursache niemals produktiv eingesetzt werden kann und darf.*

*Spezielle Testverfahren und -aufbauten*

Alle bisher genannten Messungen werden standardmäßig bei allen Systemen durchgeführt. Darüber hinaus wurden in der OSADL QA-Farm spezielle Testverfahren und -aufbauten auf ausgewählten, teilweise nur für diesen Zweck in die OSADL QA-Farm eingebrachten Systemen installiert. Im einzelnen handelt es sich dabei um Latenzmessungen verschiedener Art zum Beispiel im Zusammenhang mit

- Echtzeit-Optimierung
- Peer-to-peer UDP Duplex-Link
- PTP/TSN
- Powerlink
- Ethercat
- Netzwerklast
- Virtualisierung mit kvm

*Beispiel für speziellen Testaufbau: PTP/TSN*

Für TSN-basierte Echtzeitkommunikation über Standard-Ethernet ist es erforderlich, eine hochgenaue zeitliche Synchronisation der beteiligten Computer-Systeme zu erreichen. Um die Qualität dieser Synchronisation im Langzeitverlauf zu erfassen, wurde ein PTP-Grandmaster mit Slave-Systemen verbunden, und es werden die Jitterwerte zwischen System und Netzwerkkontroller auf dem Grandmaster sowie zwischen Netzwerk und Netzwerkkontroller und zwischen Netzwerkkontroller und System auf den Slave-Systemen aufgezeichnet. In den Abbildungen 18 bis 20 sind diese Daten jeweils in der Wochenansicht wiedergegeben. Die Messergebnisse belegen, dass es gelungen ist, die Systeme mit Hilfe von PTP außerordentlich genau zu synchronisieren, wobei der Jitter zwischen den verschiedenen Zeitdomänen immer im einstelligen Mikrosekundenbereich liegt.

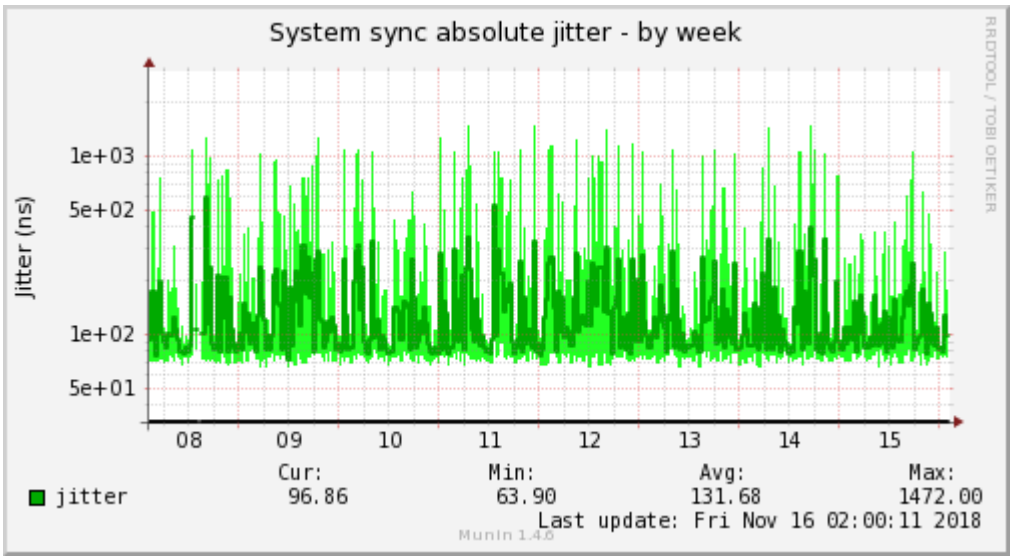


*Abbildung 18: Jitter des PTP Grandmaster*

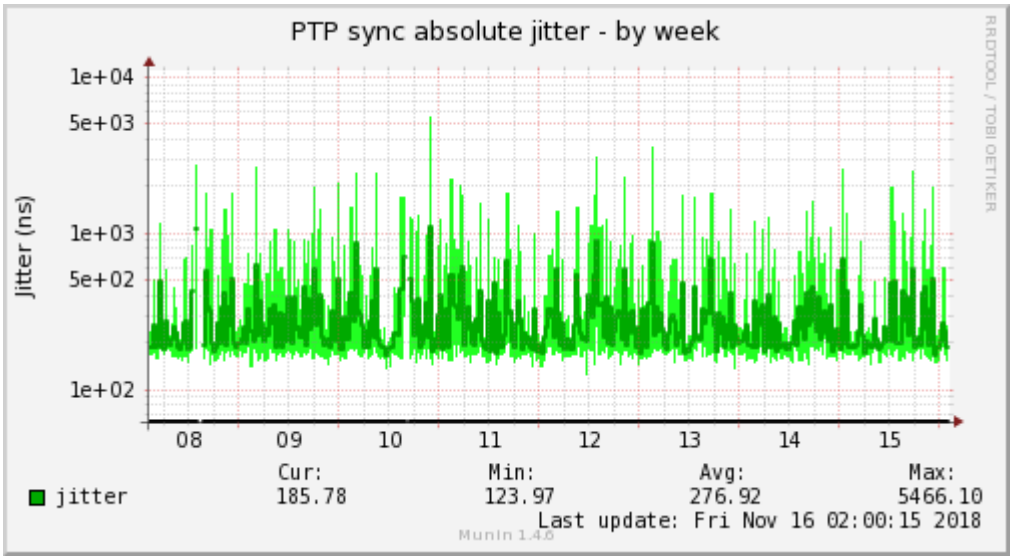


*Abbildung 19: Jitter zwischen Netzwerk und Netzwerkkontroller des PTP-Slave*

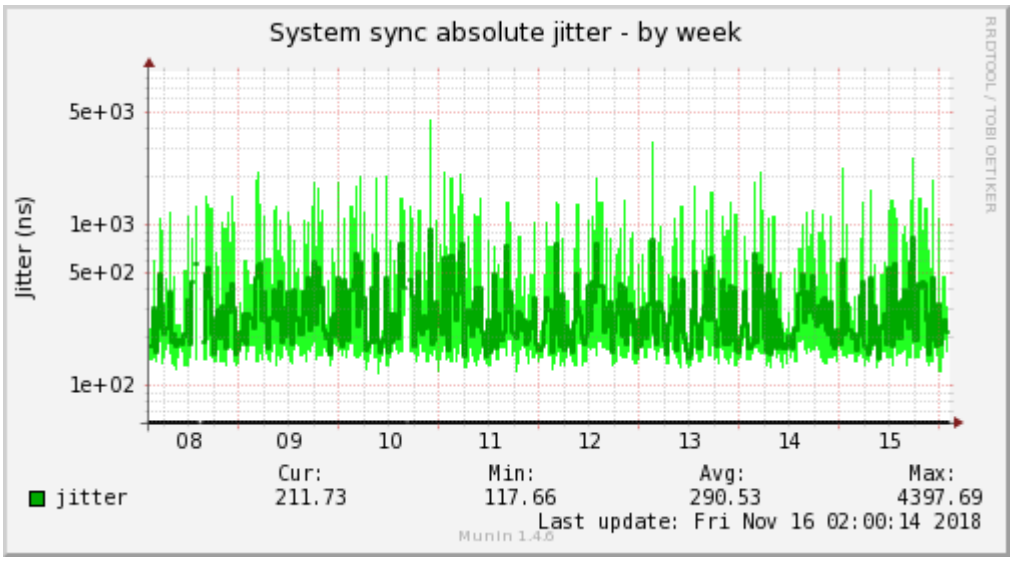


*Abbildung 20: Jitter zwischen Netzwerkkontroller und System des PTP-Slave*