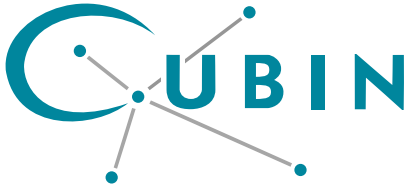
Counter Availability and Characteristics for Feed-forward Based Synchronization

Timothy Broomhead t.broomhead@ugrad.unimelb.edu.au
Julien Ridoux jridoux@unimelb.edu.au
Darryl Veitch dveitch@unimelb.edu.au

This work has been partly supported by grants from:
Cisco University Research Program Fund at Silicon Valley Community Foundation
Google Faculty Research Awards program



ARC Special Research Centre for
Ultra-Broadband Information Networks
THE UNIVERSITY OF MELBOURNE

Context

- **Robust Absolute and Difference clock**
  - Software clock synchronised over the network
  - RADclock is a set of principles and algorithms based on
    - a feed-forward approach
    - non linear filtering
  - Strong focus on robustness
  - Protocol “independent”

- **RADclock is a potential servo controller for IEEE 1588**
  - RADclock achieves ~10 µs accuracy on LAN
    - ISPCS 2007: “Ten Microseconds Over LAN, for Free”
  - RADclock is robust to network noise and disruption
    - ISPCS 2008: “The Cost of Variability”

Focus on counters

- **Feed-forward algo based on RAW counter timestamps**
- **RADclock implementation with Time Stamp Counter (TSC)**
  - Hardware counter of CPU cycles
  - 64 bit wide, “never” rolls over
  - Good stability
  - Quick access (low latency)
    - rdtsc() is 2-3 assembly instructions
  - Access from kernel and userland
- **Problem: the TSC is not reliable anymore**
  - TSC not synchronised across Multiple CPU cores / processors
  - CPU Frequency scaling to save power
  - Power management and sleeping modes

We need another counter

- **There are a few counters available on modern computers**
  - TSC - Time Stamp Counter of CPU cycle
  - ACPI - Advanced Configuration and Power Interface
  - HPET - High Precision Event Timer

- **But they ...**
  - roll over in a few minutes
  - No read_counter() function exported
    - not accessible directly from within the kernel
    - not exposed to userland and not accessible by applications

	TSC	HPET	ACPI
Freq.	2-3 GHz	14.3 MHz	3.57 MHz
Size (bits)	64	32/64	24/32

Current kernel support for Feedback algo

- **UNIX* kernel supports feedback sync' algorithms**
  - Essentially for the NTP daemon
    - IETF RFC 1589 "A Kernel Model for Precision Timekeeping"
  - Kernel feeds timestamps to NTP via the system clock
  - NTP feeds rate correction to the kernel system clock

- **"Indirect" access to any counter**
  - Timestamps accessible via system clock
  - gettimeofday() is a deforming lens

New kernel support for Feed-Forward algo

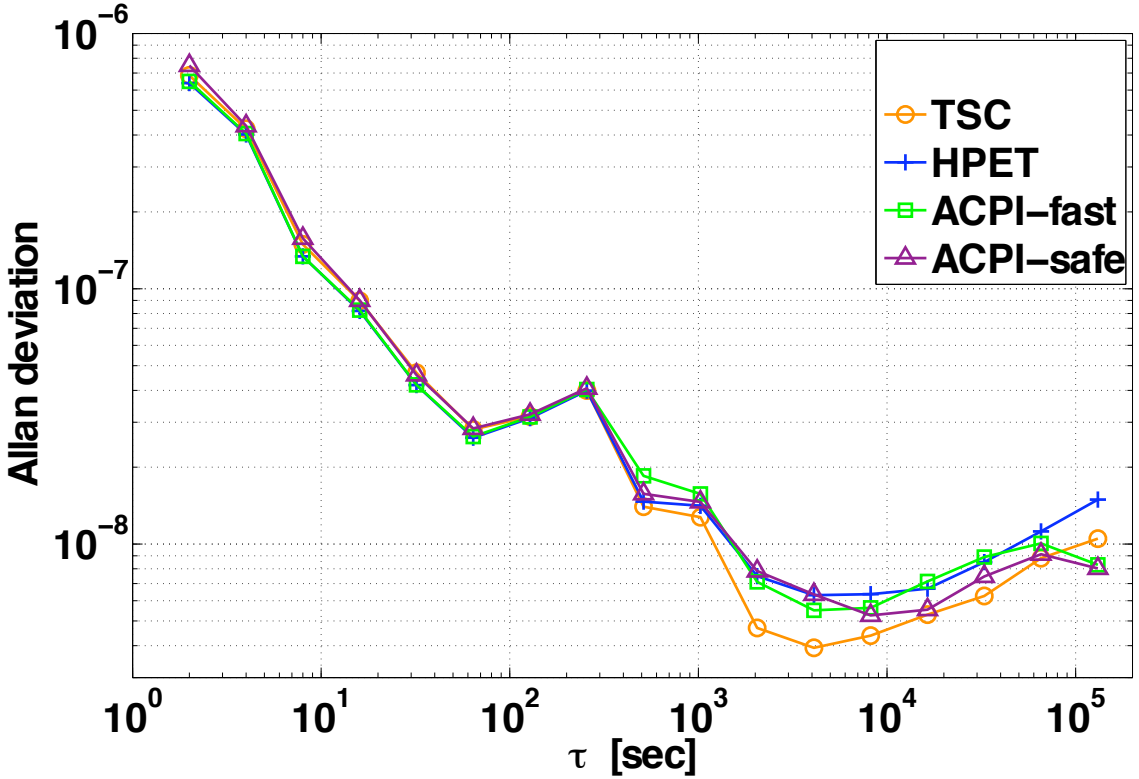
- **New virtual counter**
  - 64 bit wide
  - Monotonically accumulates at real counter rate
  - Cumulative view of real counter (no roll over)
- **Common interface that supports any real counter**
- **Direct access to RAW virtual timestamps (cycles)**
  - read_counter() function returns current virtual counter value
  - Accessible from kernel and userland (syscall)

Counters characteristics

- **Any counter present is now available to feed-forward algo**
- **Which one to choose?**
  - Do all counters meet basic stability requirements?
  - Which one is best?
- **Compare counters characteristics**
  - Use virtual counter interface for direct access to the counter
  - Stability, Latency ... and under stress
  - Operating system point of view

Counters stability - Allan deviation

- **Allan deviation using the virtual counter interface**
  - All counters < 1 PPM
  - All counters look similar at most time scales
    - Weekly variations due to consecutive captures and weather
  - All affected by air conditioning periodic cycles

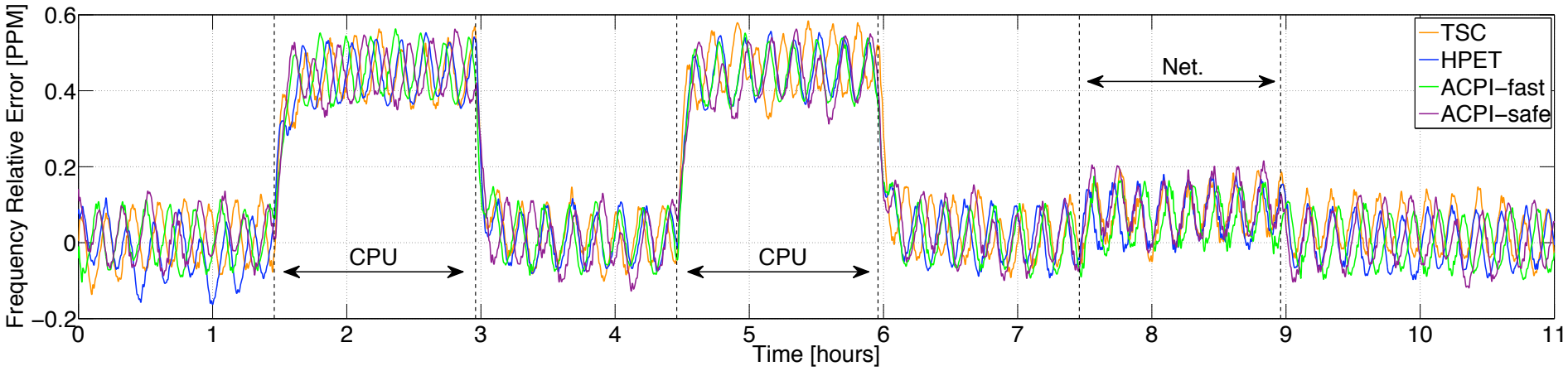


Counters under stress - Stability

- **Starts with stable conditions and alternate period of stress**
- **CPU Stress scenario**
  - Userland program that continuously hogs the CPU
  - Assumption: this should affect the TSC
- **PCI bus congestion**
  - Generate very heavy network traffic load
  - Use tcpdump to capture all packets in promiscuous mode
  - Assumption: this should affect ACPI and HPET
    - ACPI and HPET accessed via PCI bus

Counters under stress - Stability

- **Measure relative counter frequency error (in PPM)**
  - All counters show similar reaction
  - Worst case < 1PPM
- **Variation corresponds to temperature impact**
  - Not a direct consequence of the stress tests



Measuring the Counter Latency

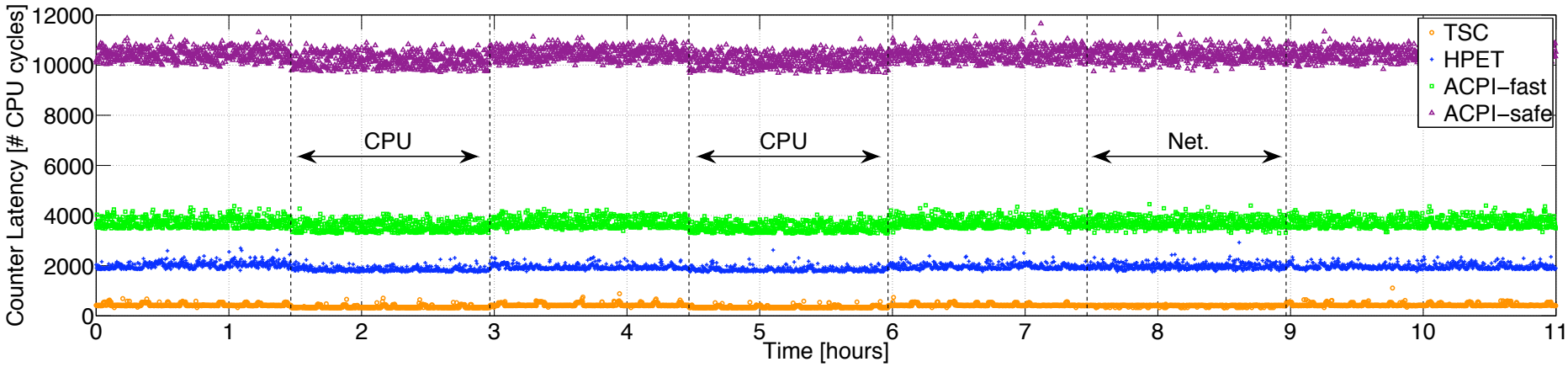
- **Counters have different access methods**
  - TSC: read register on CPU with few assembly instructions
  - HPET & ACPI: bus architecture (e.g. PCI)

- **Metric: number of CPU cycles to read a given counter**
  - before = rdtsc()
  - access counter via “virtual counter” interface
  - after = rdtsc()

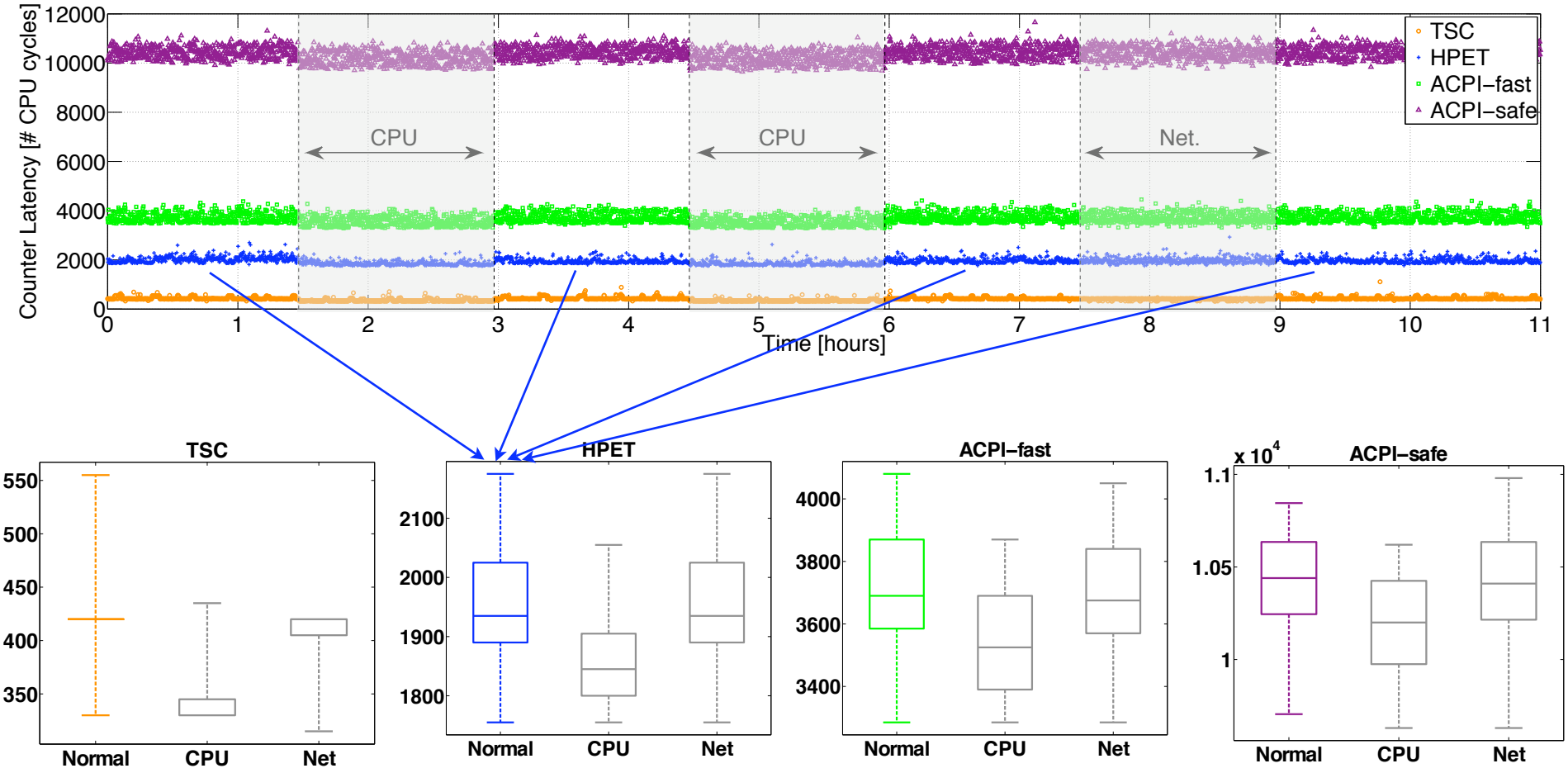
- **measured latency = after - before**
  - Measured in CPU cycles
  - Converted to time assuming 3GHz CPU

Counter Latency

- **Measure latency of each counter**
- **Apply same stress scenario**
- **Observe first difference between counters**



Counter Latency - Normal

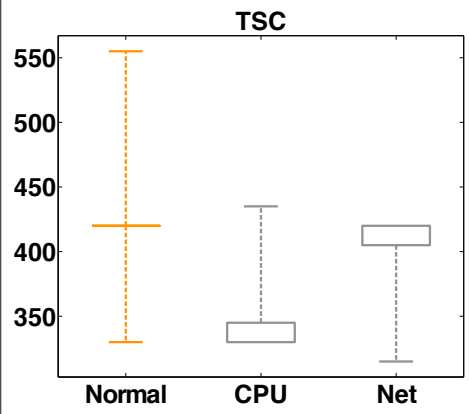


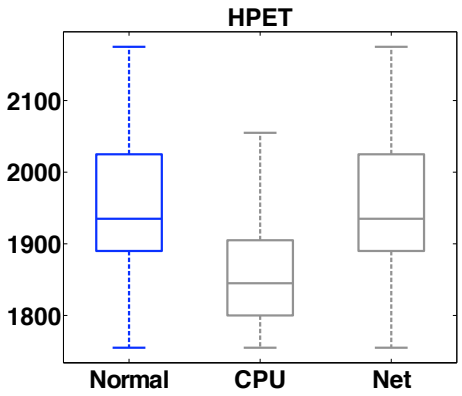
Counter Latency - Normal

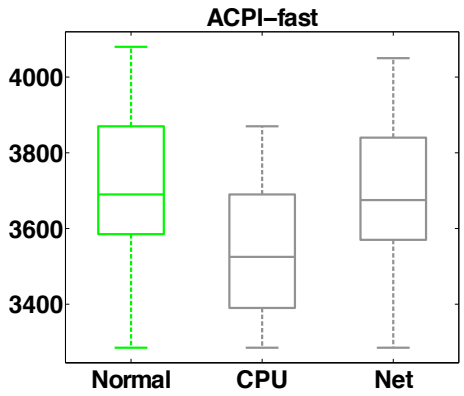
- **Median values very different**
  - TSC the fastest
  - ACPI-safe about 25 times slower
  - ACPI-* latency > 1 µs

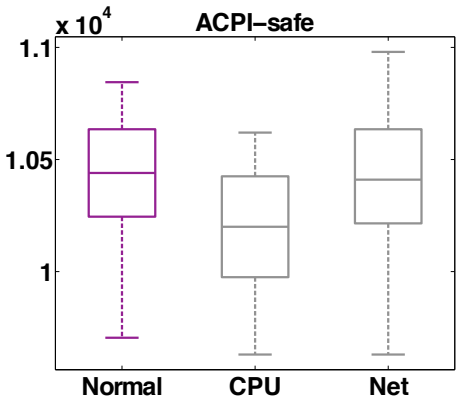
	Median µs (cycles)	IQR ns (cycles)
TSC	0.14 (420)	~0
HPET	0.65 (1935)	45 (135)
ACPI fast	1.23 (3690)	93 (280)
ACPI safe	3.48 (10440)	130 (390)

- **Variability of counters is relatively small**
  - Can be modelled as a almost constant offset error



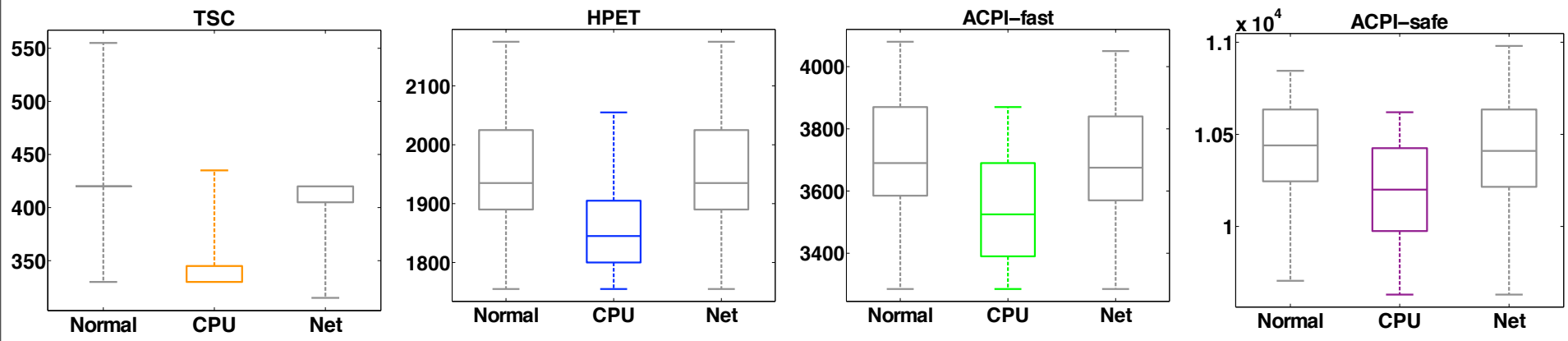






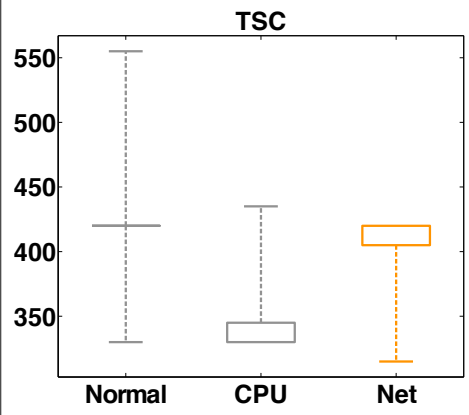
Counter Latency - CPU Stress

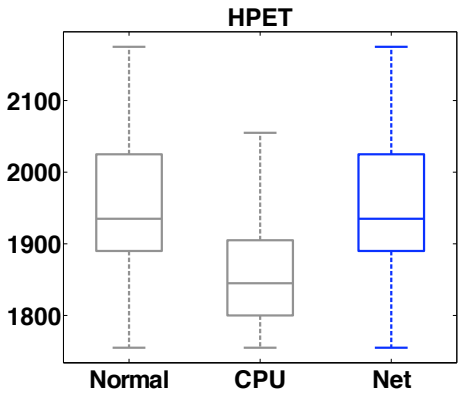
- **Minimum remains the same**
- **Median latency of each counter seems to improve!**
  - CPU caches are “hot” + pipelining and cache optimisations
  - Actually, methodology itself is improved and suffers less noise
- **Variability barely affected**
  - Worse case: smaller probability to be interrupted, less outliers

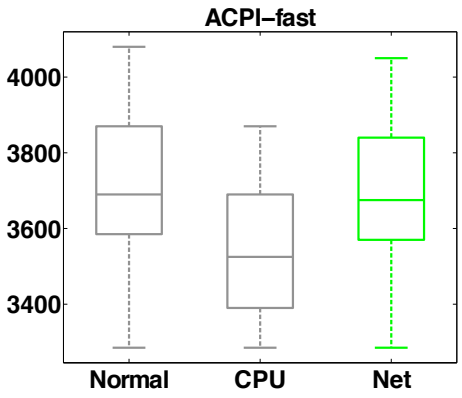


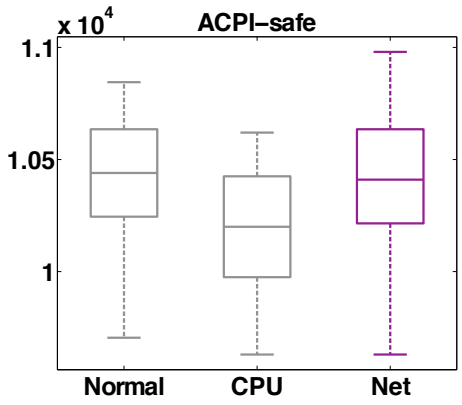
Counter Latency - Network stress

- **Results extremely similar to "Normal"**
- **Except for extreme outliers (not shown here)**
  - They correspond to process being interrupted
  - Same proportion of measurements are outliers
  - ... but process interrupted for a longer time



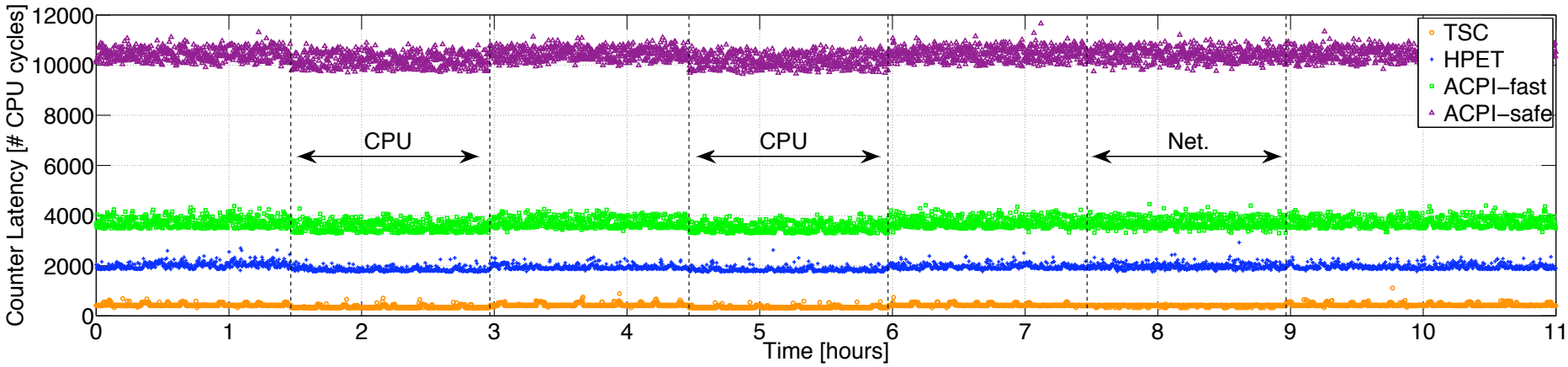






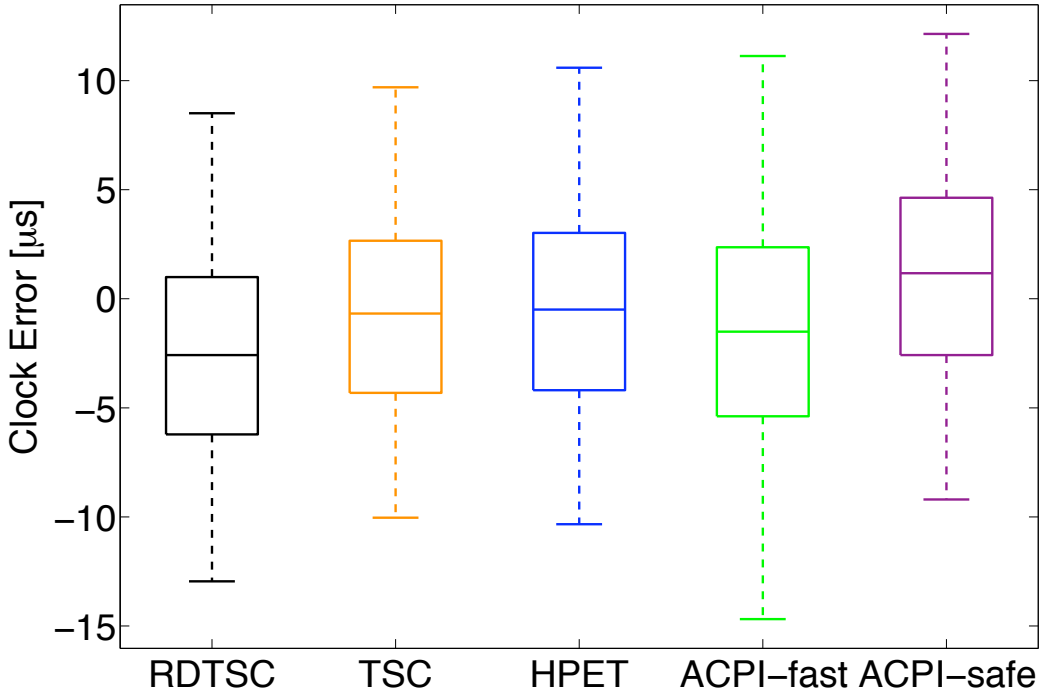
Counter Latency

- **Counters have different latencies**
  - But variability of their latency is small
- **Counters latency not affected by stress scenarios**



RADclock performance comparison

- **RADclock performance**
  - Sync' over LAN to a stratum-1 server
  - Median error < 10 µs
  - IQR error < 10 µs



- **Any counter can be used**
  - Similar performance
  - HPET quite good

Conclusion - Virtual counter

- **A Virtual Counter interface for feed-forward sync' algos**
  - Architecture independent (multi-core, power management etc)
  - Can access any raw counter from kernel and userland

- **Studied characteristics of TSC, HPET and ACPI**
  - Stability:
    - All counters are very similar in "Normal" conditions
    - All affected by temperature in a similar manner
  - Latency:
    - TSC is the fastest, but if not possible, HPET is a very good alternative
    - Small variability even under stress

- **RADclock: µs accuracy over LAN with any counter**
  - Code source available under GPL licence
  - http://www.cubinlab.ee.unimelb.edu.au/radclock/