TECHNISCHE UNIVERSITÄT DRESDEN

Faculty of Computer Science, Operating Systems Group

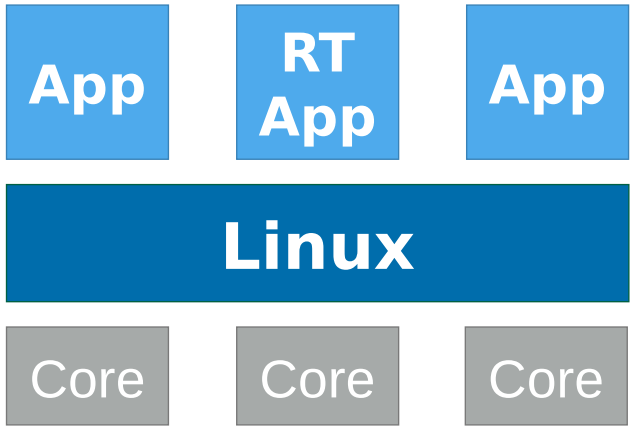
Combining Predictable Execution with Full-Featured Commodity Systems

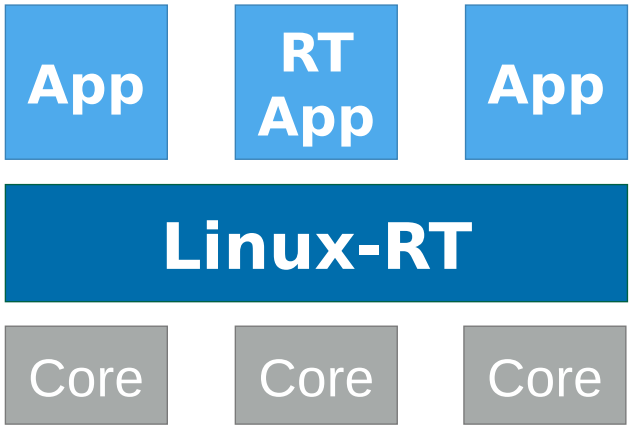
Adam Lackorzynski, Carsten Weinhold, Hermann Härtig

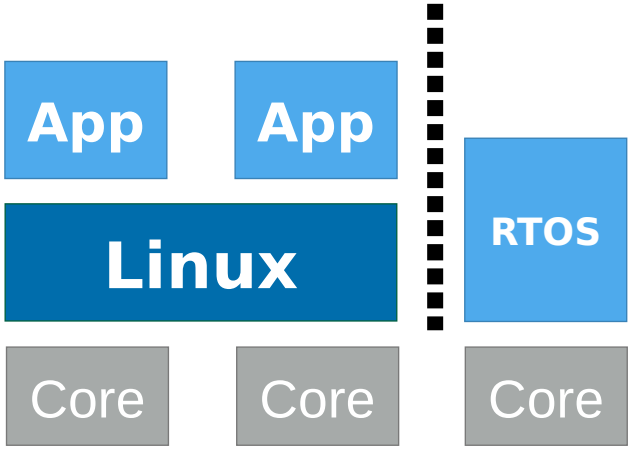
Jul 5th 2016, OSPERT Workshop

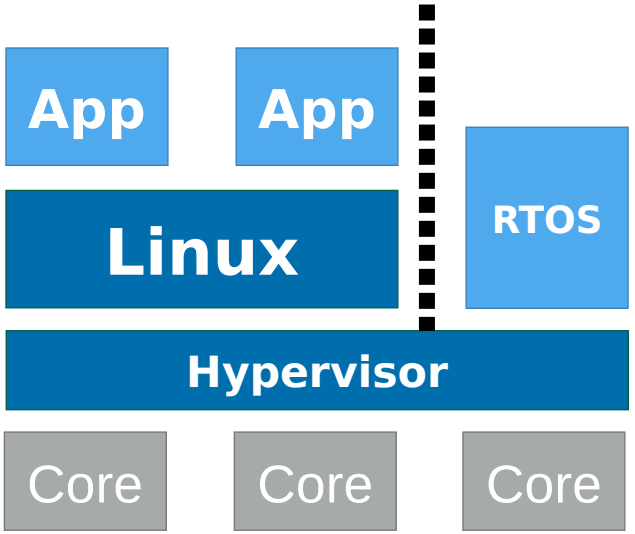
TECHNISCHE UNIVERSITÄT DRESDEN

Combining a GPOS and Real-Time



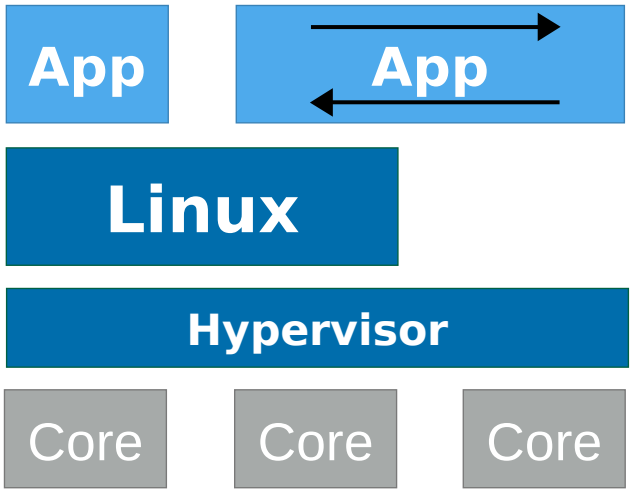




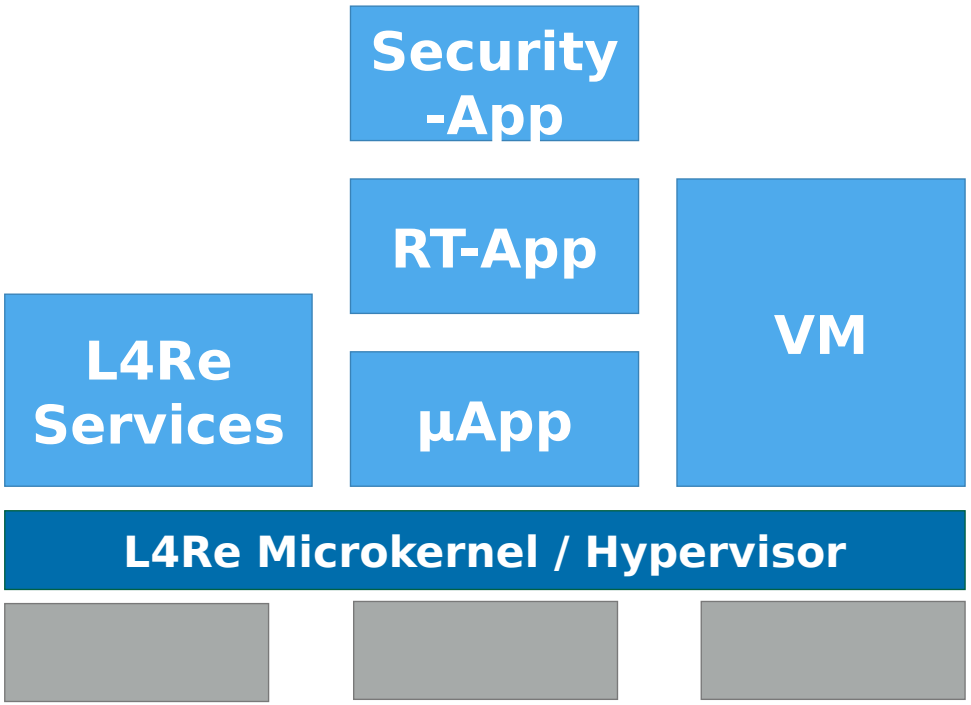


Our Goal

- Use GPOS (Linux) for non-RT parts of an RT application
- Run RT part outside of Linux
  - Separate „noisy“ Linux from sensitive parts
- Best of both worlds:
  - Use Linux for features
  - Run RT outside of Linux
- No gap to overcome
  - No application splitting
  - No proxying



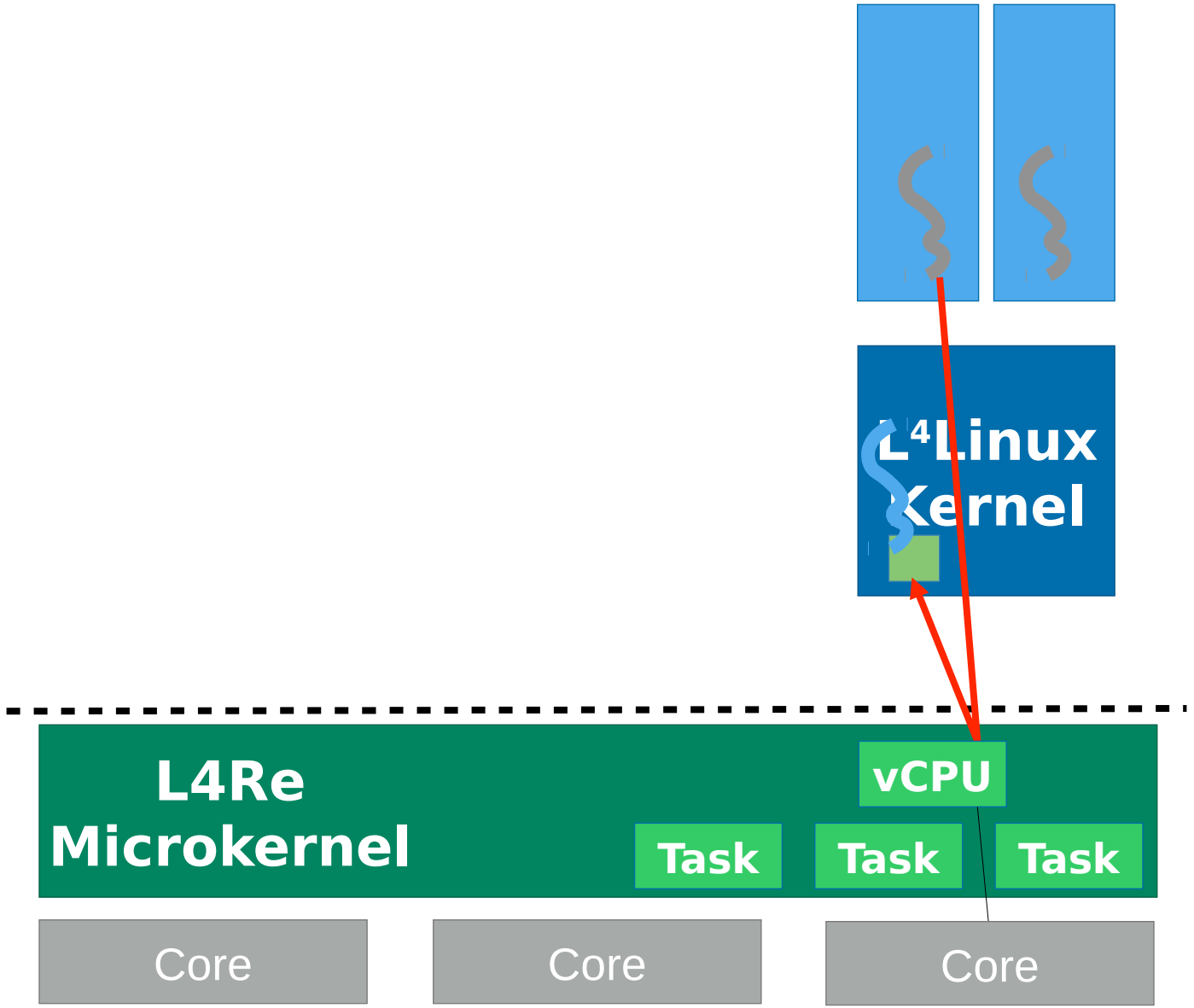
L4Re Microkernel System



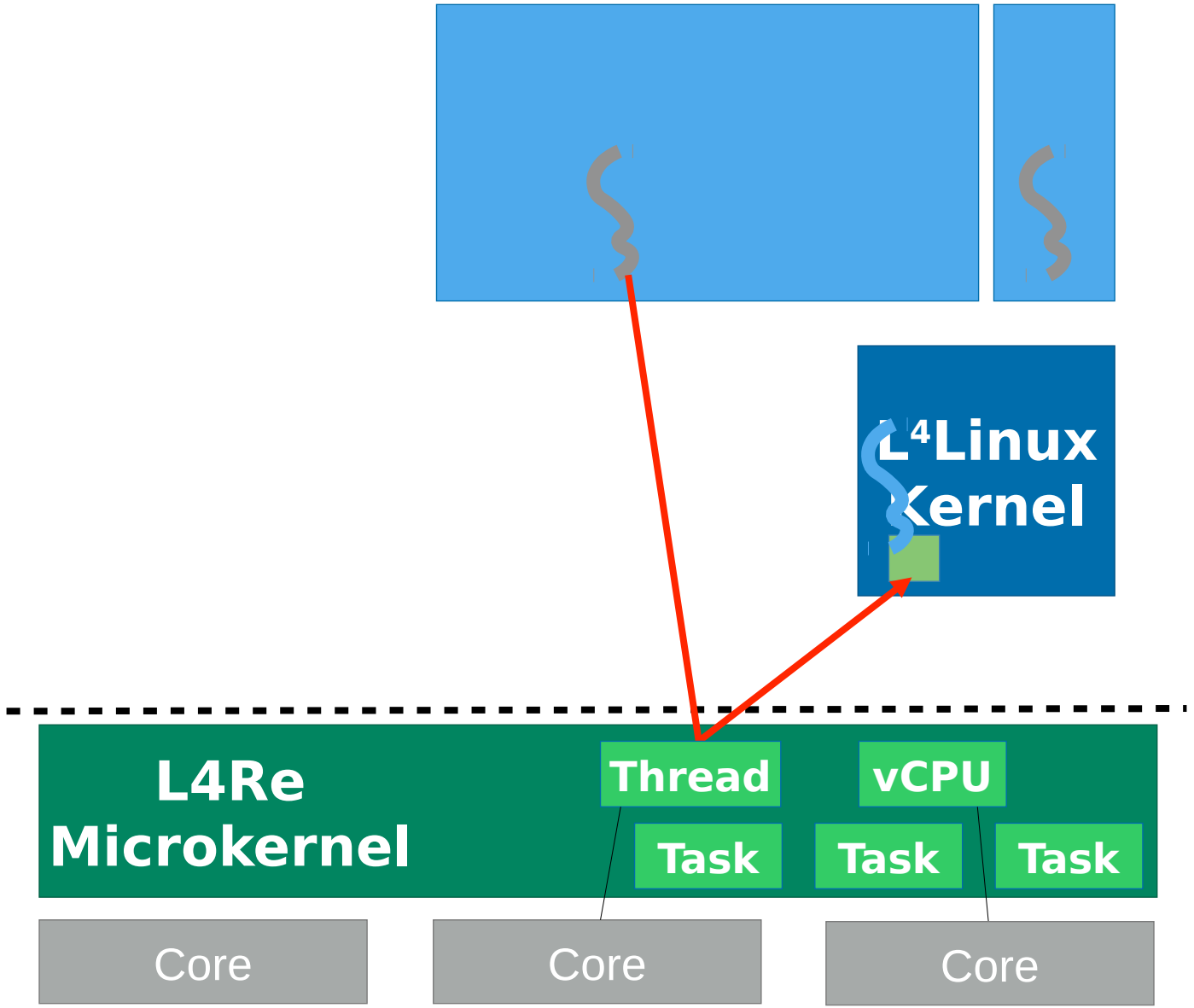
L4Re Virtualization Variants

- Hardware-assisted virtualization
  - Intel VT-x, AMDs SVM, ARM VE, MIPS VZ
- Para-Virtualization
  - No hardware support for Virtualization required
  - L^{4}Linux, FreeRTOS, L4OpenBSD, …
  - Tight integration...

L⁴Linux


L⁴Linux Kernel
L4Re Microkernel
vCPU
Task
Task
Task
Core
Core
Core

Detaching Linux Threads



Summary

- Thread of an application can run undisturbed from Linux activities
- Program can execute L4 system calls directly, without involving the Linux kernel
- Multi-threaded applications can run detached and normal in parallel

Evaluation / Benchmarks

FWQ (Fixed Work Quantum)

- Measure OS Noise
- Measure execution time jitter

Control loop hybrid application

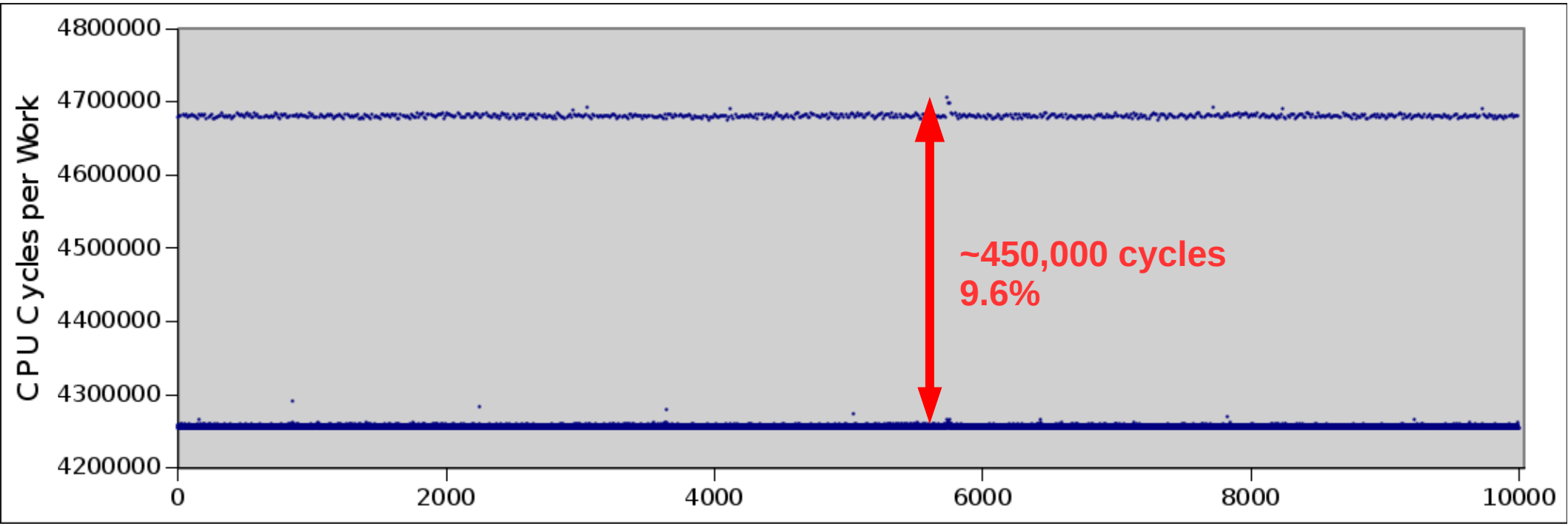
3 Workloads:

- Idle
- Build: Building a Linux kernel
- IO: Stressing disk and network

Intel Core™ i7-4770, Quad-Core at **2993**MHz

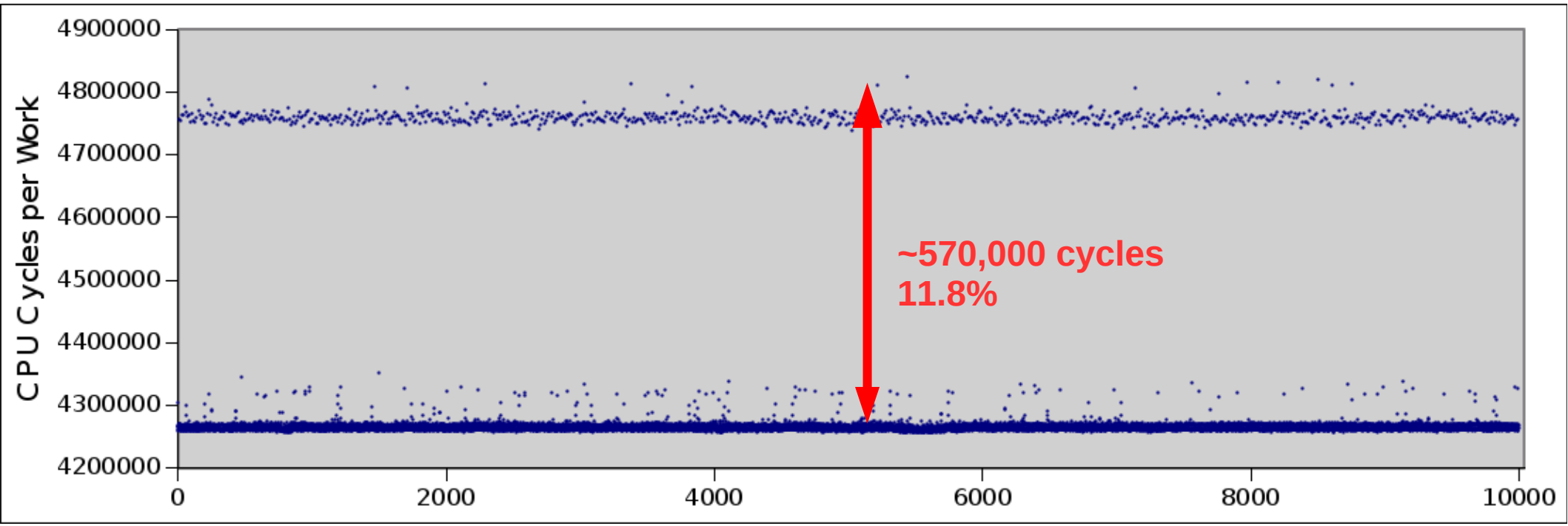
FWQ Linux Examples

- Linux idle



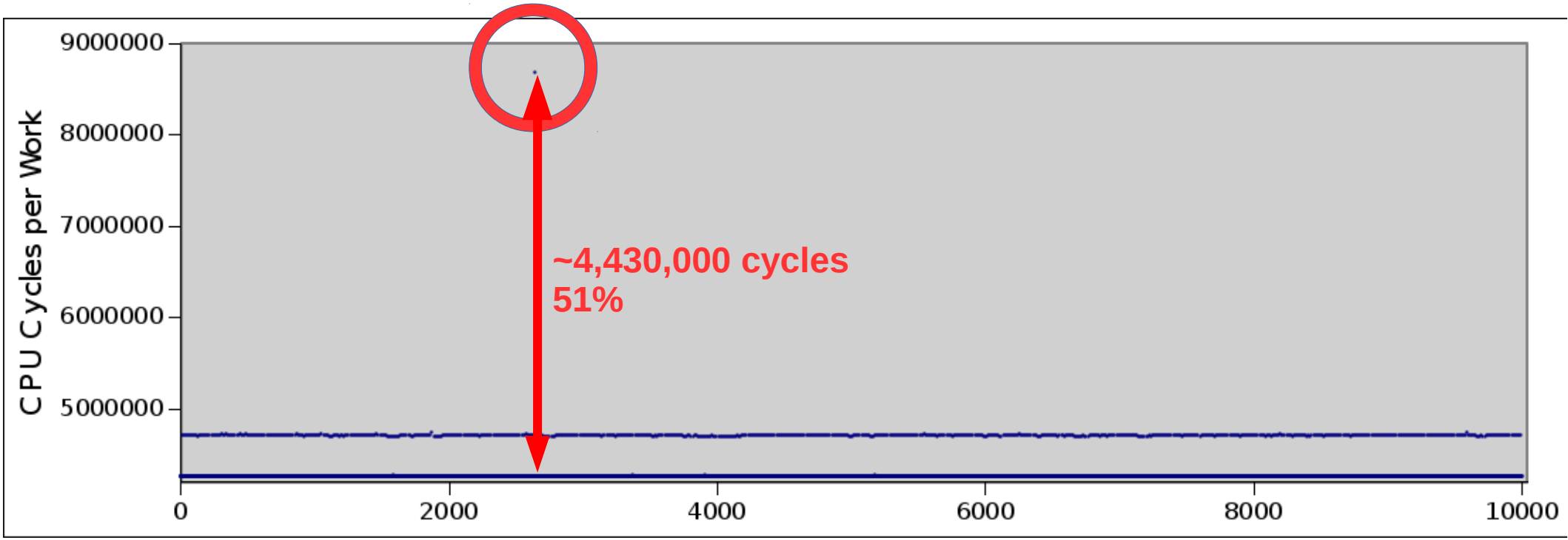
FWQ Linux Examples

- Linux build load



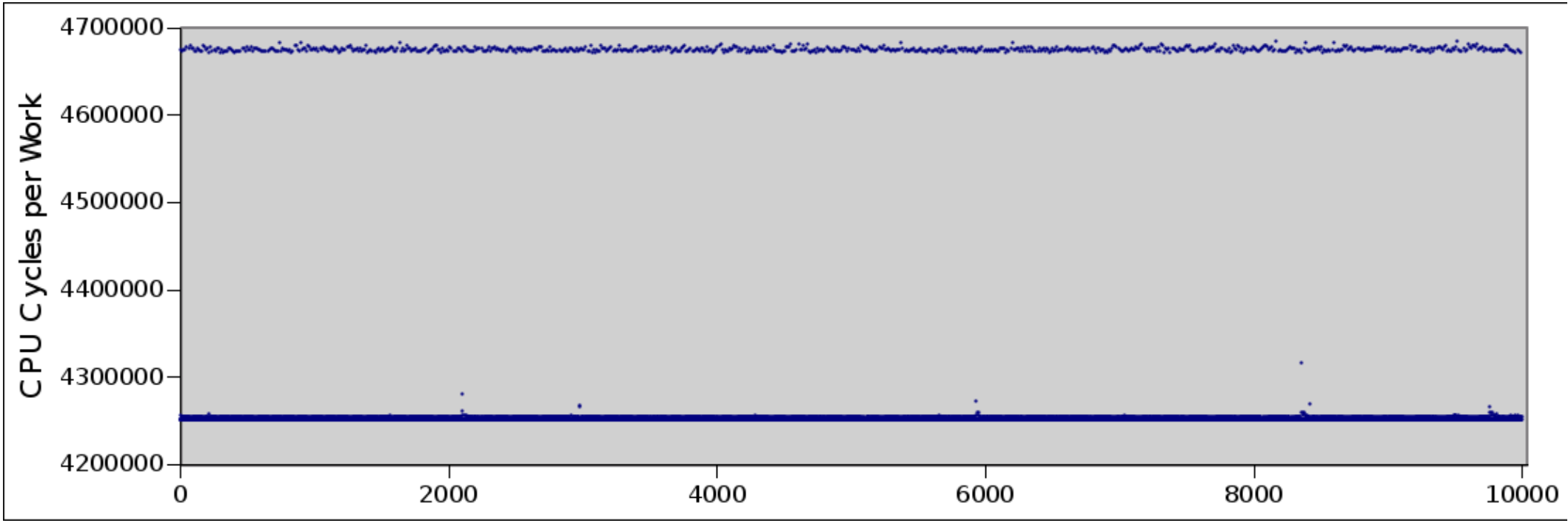
FWQ Linux Examples

- Linux I/O load



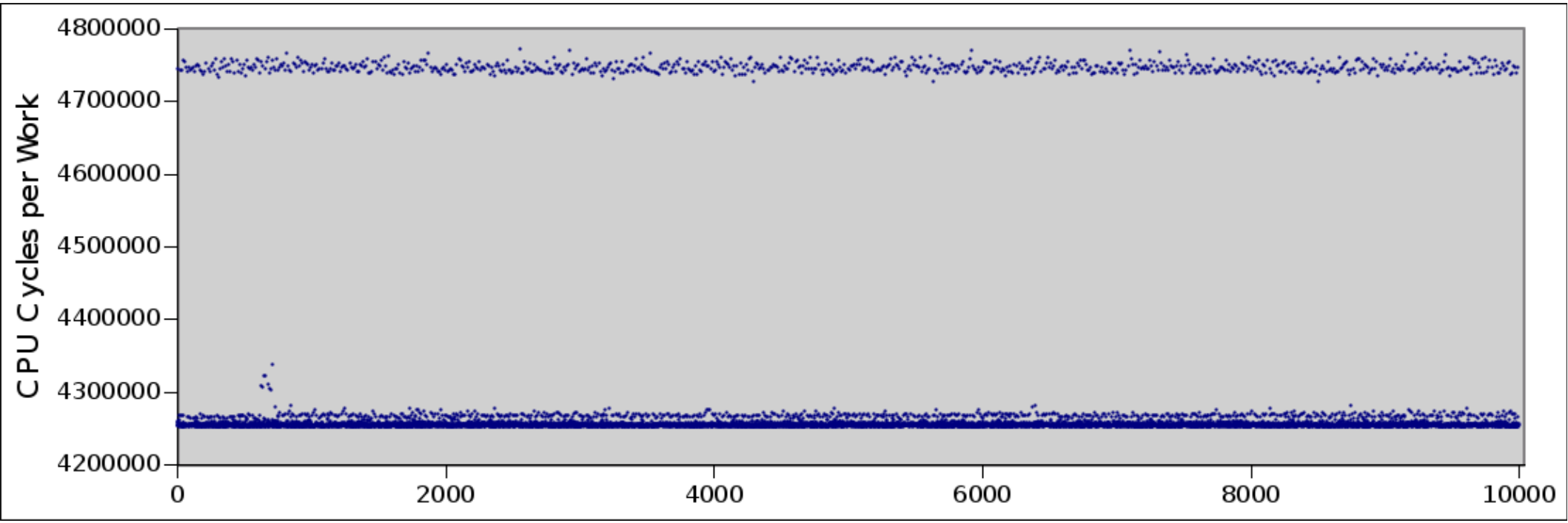
FWQ Linux PREEMPT Examples

- Linux PREEMPT Idle



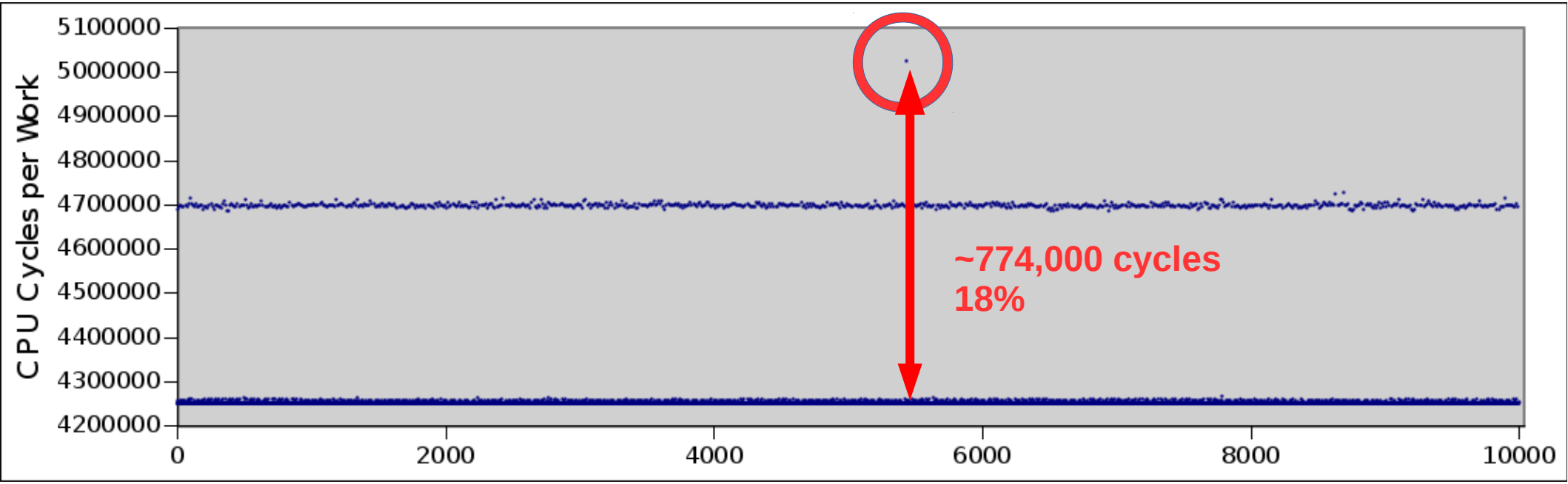
FWQ Linux PREEMPT Examples

- Linux PREEMPT build load



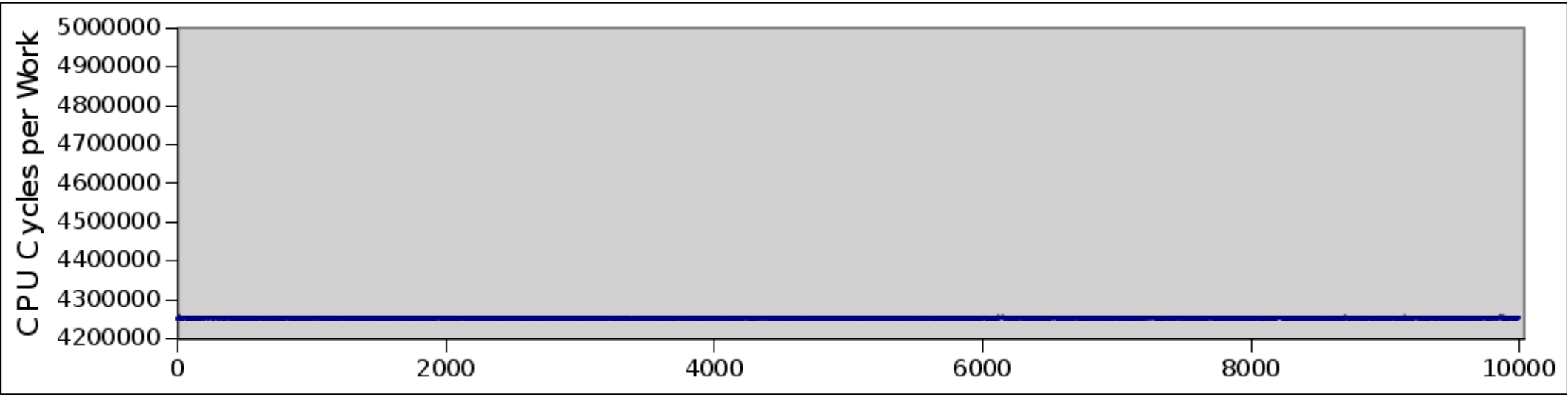
FWQ Linux PREEMPT Examples

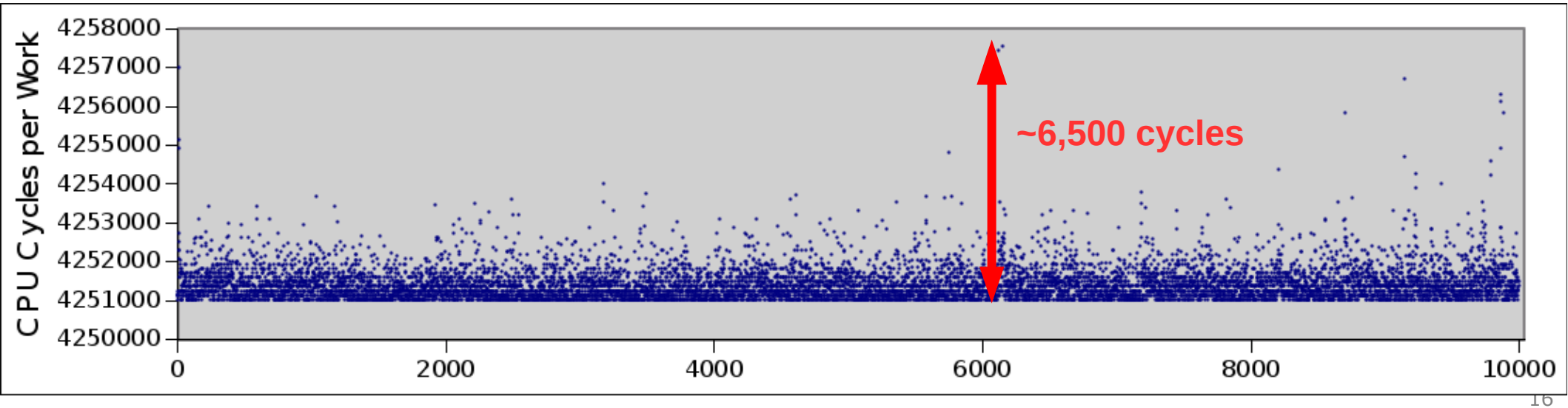
- Linux PREEMPT I/O load



FWQ L⁴Linux with Detaching

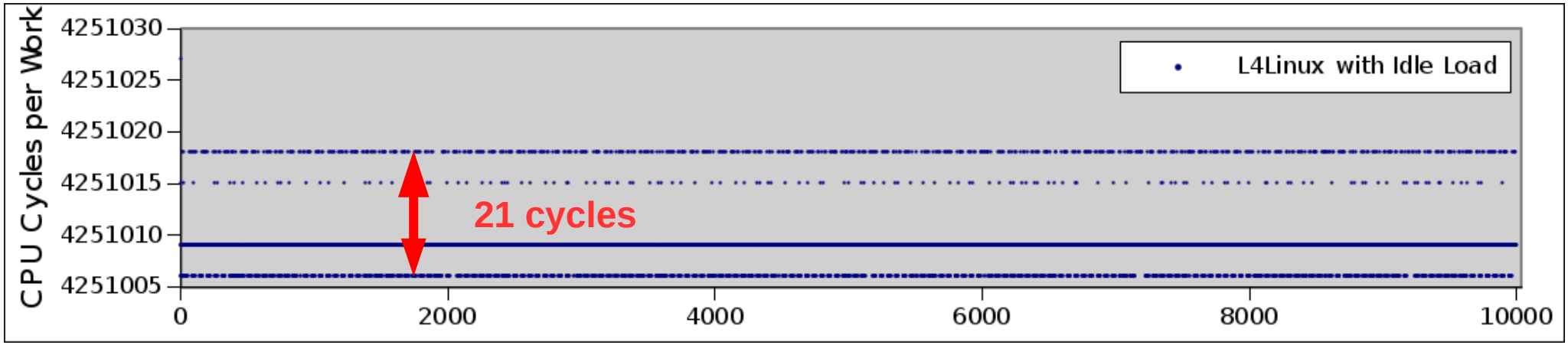
- Detached build load

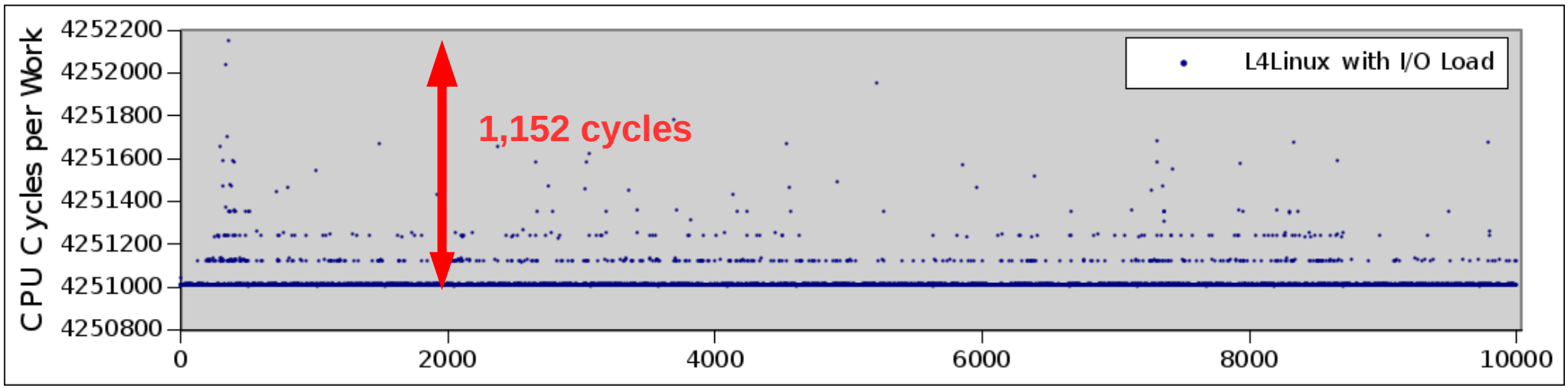




FWQ L⁴Linux with Detaching

- With idle, build load & I/O load





TECHNISCHE UNIVERSITÄT DRESDEN

FWQ Numbers

Deltas to minimum in CPU cycles	Linux PREEMPT	Detached Thread in L⁴Linux
Idle	434000 (10.2%)	21 (0.0005%)
Build Load	520000 (12.2%)	6528 (0.15%)
I/O Load	773000 (18.2%)	1152 (0.027%)

Control Loop

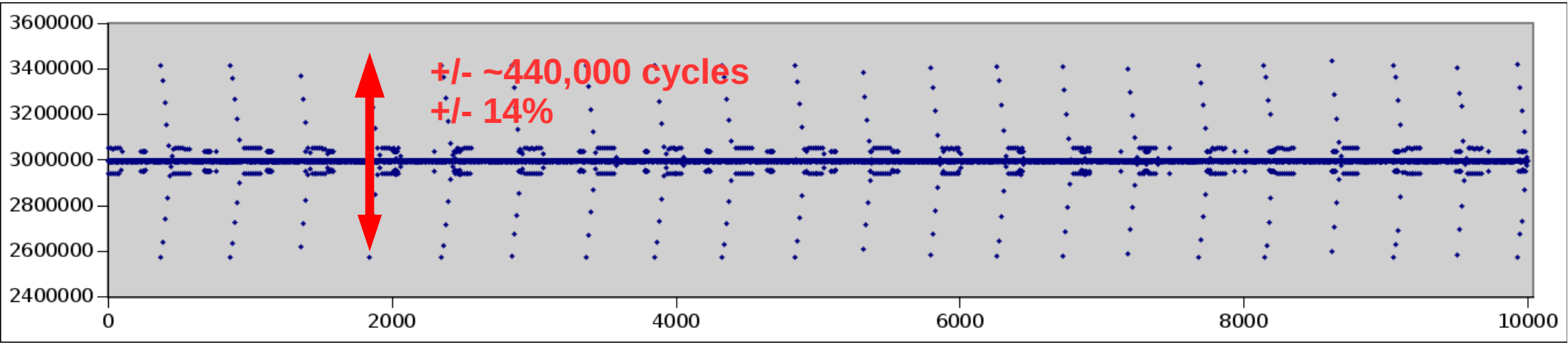
- Looking at interrupt latency
- Use Linux to launch the following code:

```
next = now() + 1000µs;
while (1) {
   wait_for_time(next);
   tsc = read_tsc();
   histogramm[i++] = tsc - prev_tsc;
   prev_tsc = tsc;
   /* do work */
   next += 1000µs;
}
```

- Linux and L4 implementations of now() and wait_for_time()

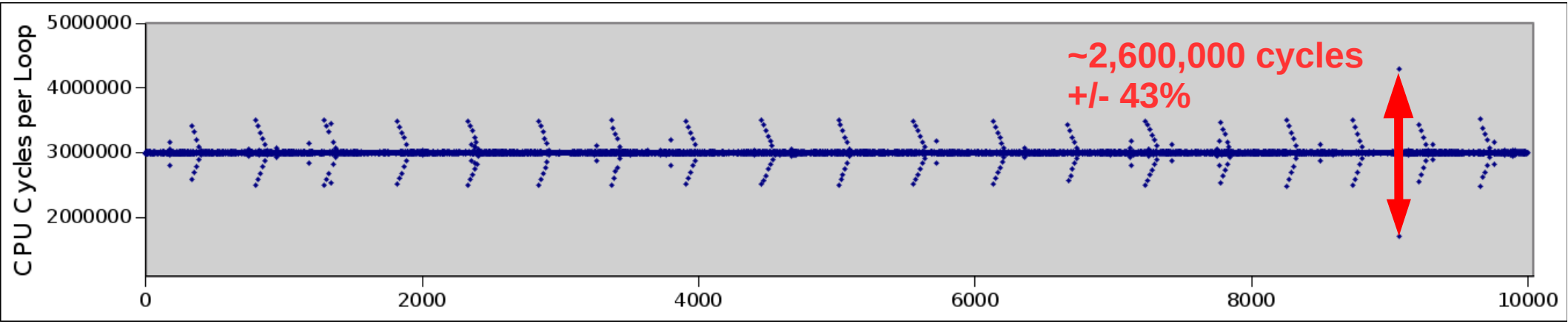
Control Loop with Linux

- Linux Idle



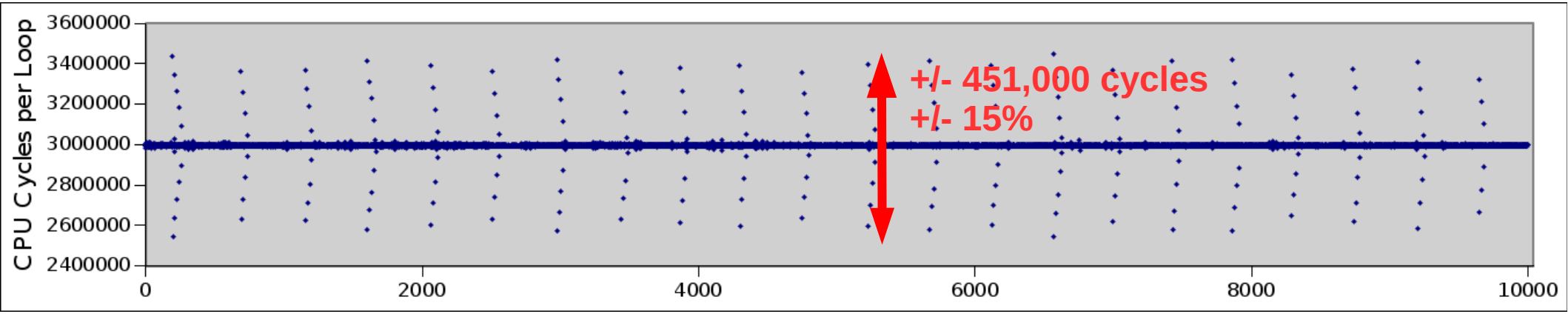
Control Loop with Linux

- Linux build load



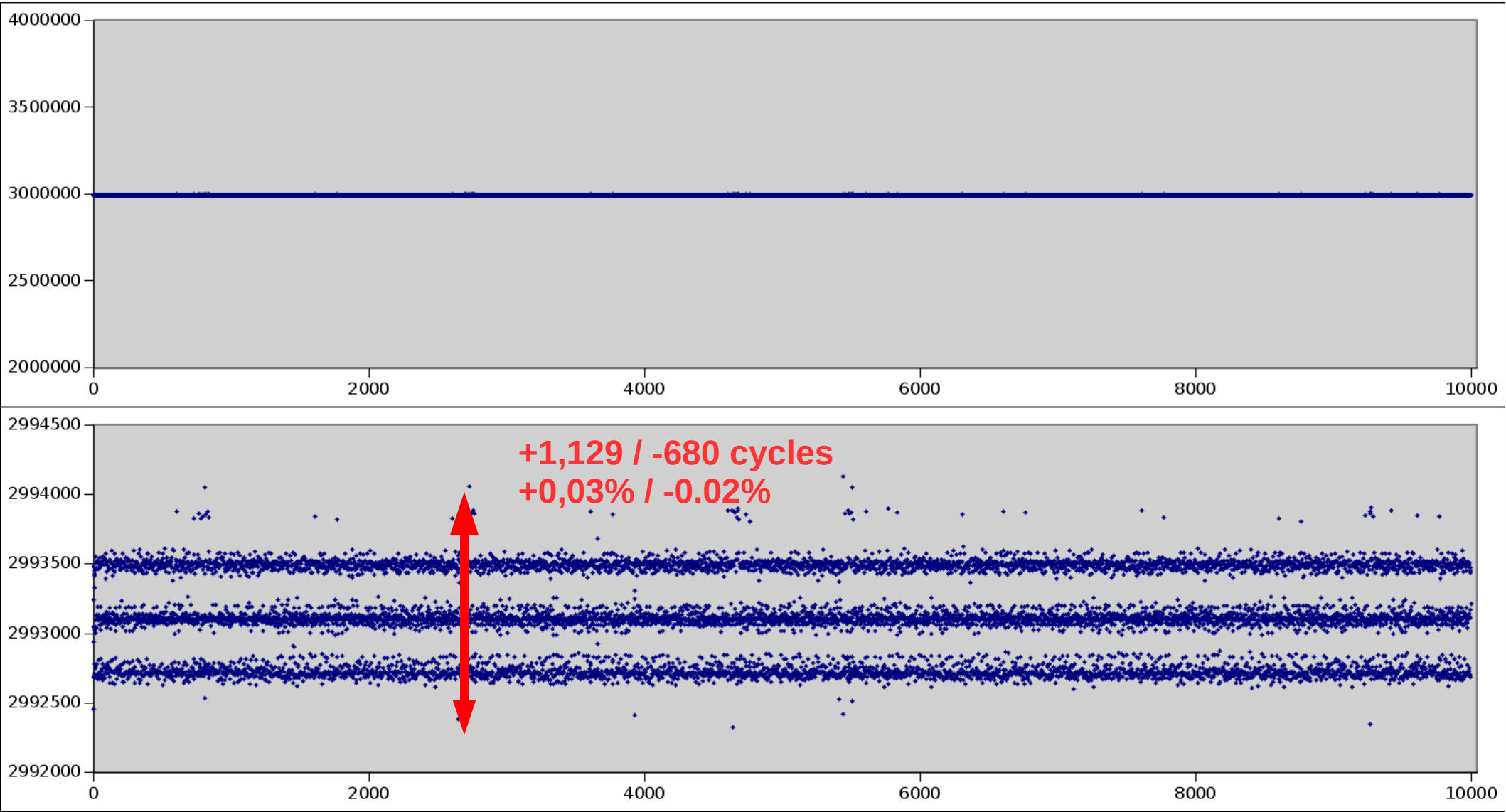
Control Loop with Linux

- Linux I/O load



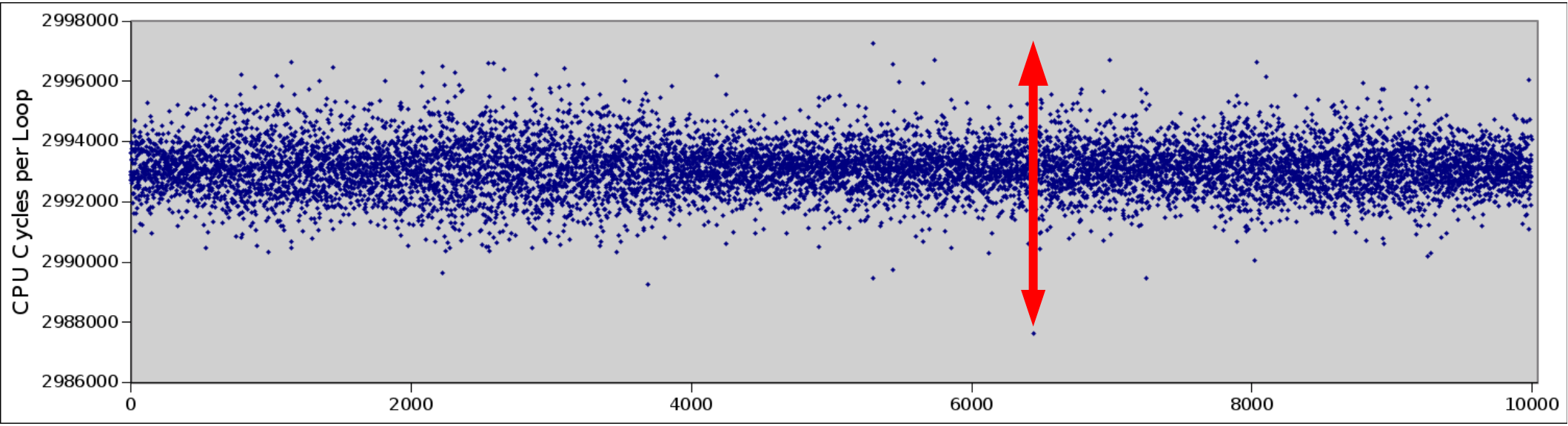
Control Loop in Detached Thread

- L⁴Linux detached Idle



Control Loop in Detached Thread

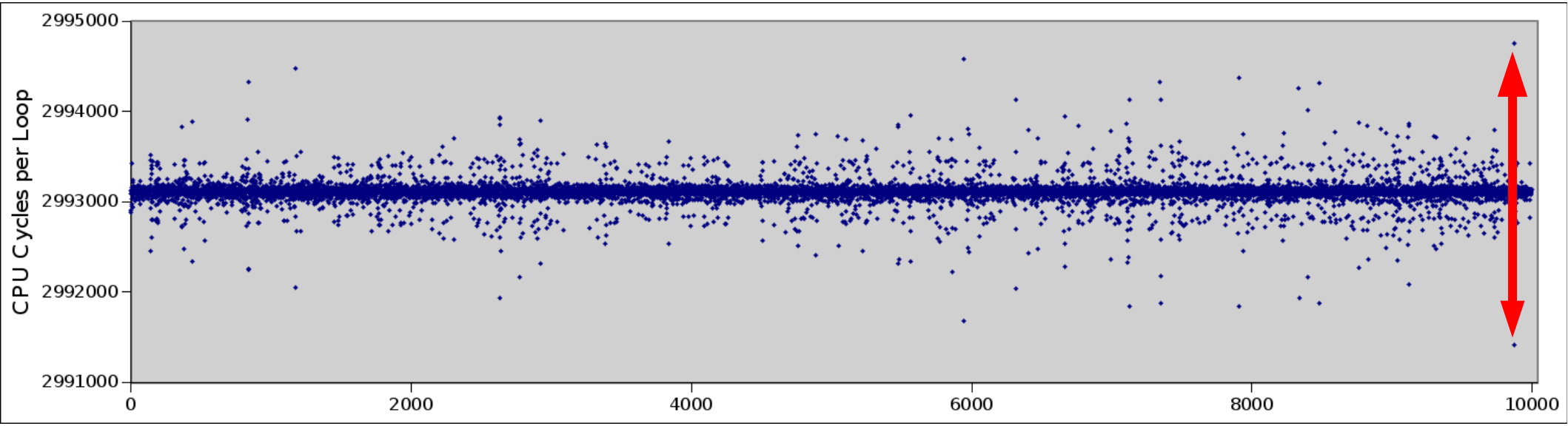
- L[4]Linux detached build load



+4,200 / -5,400 cycles
+0,14% / -0.18%

Control Loop in Detached Thread

- L⁴Linux detached I/O load



+1,753 / -1,586 cycles
+0,06% / -0.053%

Control Loop Execution

Maximum Deviation in CPU cycles	Linux	L⁴Linux Detached Thread
Idle	862000 (+14,7% / -14.1%)	1809 (+0,03% / -0.02%)
Build Load	2599000 (+/- 43,4)	9633 (+0,14% / -0.18%)
I/O Load	903000 (+/- 15,1%)	3339 (+0,06% / -0.053%)

Conclusion

- Separating a part of a program from a GPOS is possible
- Avoids separation gap

- Real-time and HPC have similar goals

l4re.org