

Virtualization of a Real-Time Operating System for Robot Control with a Focus on Real-Time Compliance

Halil Pamuk, BSc.

Sebastian Rauh, MSc. Beng.



Introduction

- Robots perform time-critical tasks
- Delays → catastrophic consequences
- General-Purpose Operating System vs. Real-Time Operating System

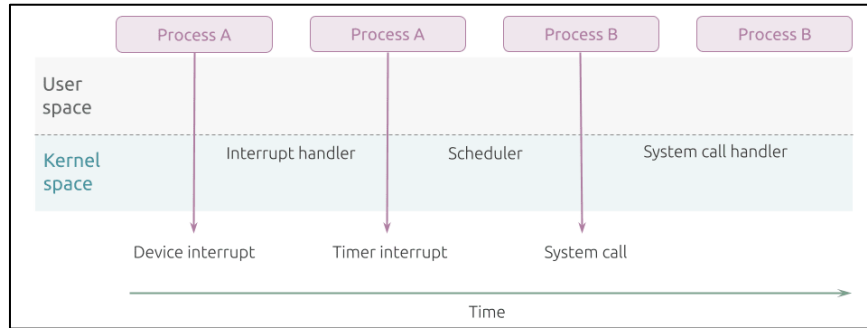


Figure 1: Non-preemptible Kernel [1]

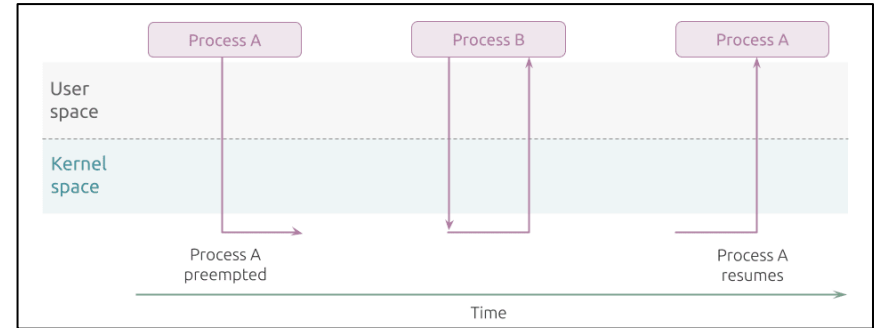


Figure 2: Preemptible Kernel [1]

Problem and Task Definition

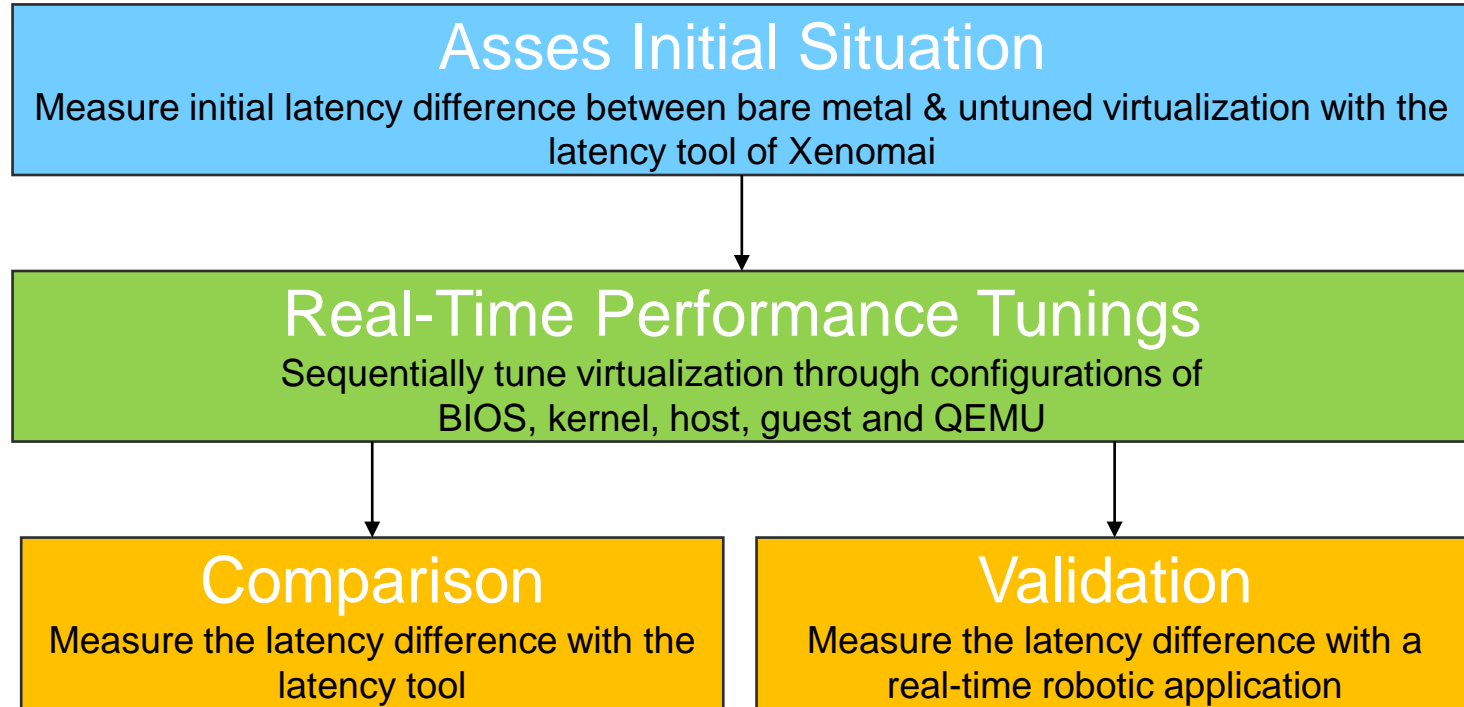
Virtualization of Real-Time Operating Systems	
Advantages	Disadvantages
Scalability & flexibility	Increased overhead and latency
Cheaper	Performance variability
Remote management	Complexity

- **Research Question:** Is it possible, and if so, how can the latency of a real-time operating system virtualization be reduced using Yocto, Xenomai, and QEMU to a level that is closer to that of bare metal (below 50 μ s)?

Application Context and Conditions

- This work was written at SIGMATEK GmbH & Co KG
- **Host OS:** Ubuntu 22.04.4 LTS, PREEMPT-RT
- **Guest OS:** Salamander 4
 - Built with Yocto [2]
 - Virtualized through Quick Emulator (QEMU) [3]
 - Hard real-time with Xenomai 3 [4]
- Trace-cmd [5] and Kernelshark [6] for kernel tracing and visualization

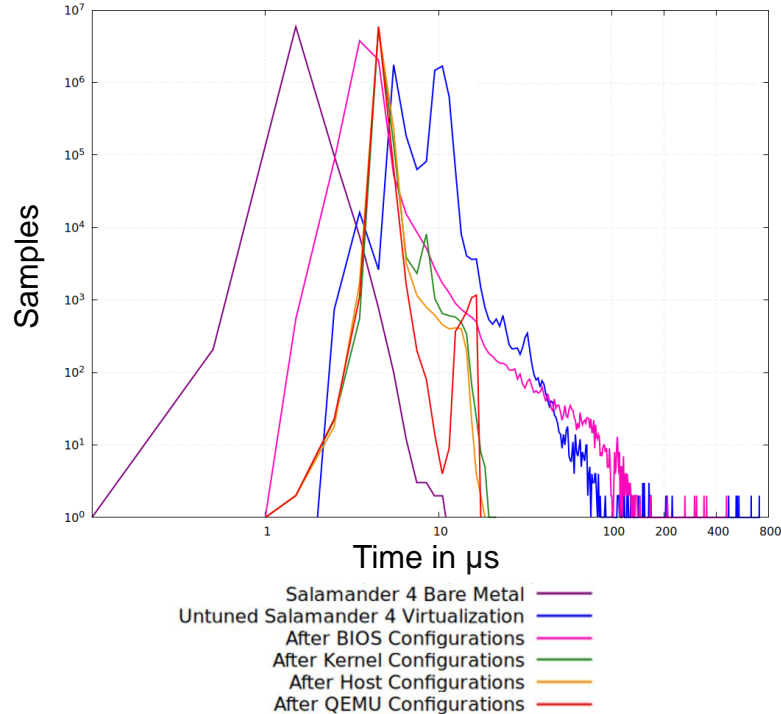
Methodology



Results – Latency Tool

- 10 minutes with a sampling period of 100 μs , priority of 99

Figure 1: Comparison of Latency Distributions



Version	Latency (μs)			Overruns
	Min	Avg	Max	
Bare Metal	0.613	1.380	10.709	0
Untuned Virtualization	2.536	8.940	707.622	43
After BIOS Configurations	0.969	3.948	457.545	22
After Kernel Configurations	2.545	4.811	21.694	0
After Host Configurations	2.591	4.834	18.441	0
After QEMU Configurations	2.614	4.779	17.134	0

Table 1: Comparison of Latency Results

Results – Robotic Application

- Difference between command issuance time and signal arrival at PWM
- 1,000 samples

Version	Latency (ms)			Std Dev (ms)
	Min	Avg	Max	
Bare Metal	1.211	1.347	1.49	0.082
Untuned Virtualization	3.1	24.603	129.46	13.876
Tuned Virtualization	1.219	2.62	3.988	0.812

Table 2: Comparison of Robotic Application Latency Results

Discussion

✓ Latency Tool

- Worst latency decreased from 707.622 μs to 17.134 μs
- Close to bare metal's 10.709 μs
- No overruns
- Goal achieved

✓ Robotic Application

- Worst latency dropped from 129 ms to 3.988 ms
- Close to bare metal's 1.49 ms
- Tunings validated

Outlook

- Additional configurations
- Other hypervisors and virtualization technologies
- More testing under workloads

References

- [1] <https://ubuntu.com/blog/what-is-real-time-linux-ii>
- [2] <https://docs.yoctoproject.org/>
- [3] <https://www.qemu.org/>
- [4] <https://xenomai.org/>
- [5] <https://trace-cmd.org/>
- [6] <https://kernelshark.org/>