Introduction to Realtime Linux

Jan Altenberg

Linutronix GmbH
Overview

1. What is Realtime?

2. Linux and Realtime

3. Results: Which latencies can be achieved with the different approaches?

4. Conclusion
What is Realtime?

Fast execution time?
What is Realtime?

Performance?
What is Realtime?

It's all about DETERMINISM!
What is Realtime?

- Correctness means execution at the correct time
- Missing the timeslot will lead to an error condition
Realtime

Remember!
Missing the timeslot will lead to an error condition
Realtime

Missing the timeline will cause a damage to your machine or even a person might get hurt:
What about "Softrealtime"?

...PLEASE PLEEAASSEE forget about this word!!! :)

Jan Altenberg

Linutronix GmbH
Who is using it?

- industry / automation
- multimedia systems
- aerospace
- financial services
- ...

Jan Altenberg

Linutronix GmbH
Requirements

- Deterministic timing behaviour
- Preemption
- Priority Inheritance / Priority Ceiling
Priority Inversion

- task 1
- task 2
- task 3

prio
blocks

task 2
gets interrupted

task 3
Approaches

- Dual-Kernel
- In-Kernel / Single Kernel
Dual-Kernel

Micro Kernel

Hardware
Single-Kernel

Hardware
RTAI

- Prof. Paolo Mantegazza, University of Milano
- Dual-Kernel approach
- Realtime in kernelspace
- Realtime in userspace very limited
- Design goal: Lowest latencies
- Supported platforms: x86, x86_64, and a couple of ARM platforms
RTAI
Xenomai

- Founded 2001
- Realtime in userspace
- Skins can emulate the API of different RTOSes
- Dual-Kernel approach
- Supported platforms: x86, x86_64, PowerPC, ARM, ia64
Xenomai

![Diagram of Xenomai architecture]

- Real-time task
- Linux application
- Real-time task
- Linux (Domain B)
- Xenomai Nucleus (Domain A)
- ADEOS/I-pipe
- Hardware interrupts
- Hardware
Known issues of dual-kernel approaches

- Special API
- Special tools and libraries
- Microkernel needs to be ported for new HW and new Linux versions
- Bad scaling on big platforms
Preempt RT

- In-Kernel approach
- Founded by: Thomas Gleixner, Ingo Molnar
- Huge community
- Most of the features already made it into "Mainline"
- POSIX realtime
- Highly accepted in the community
"Controlling a laser with Linux is crazy, but everyone in this room is crazy in his own way. So if you want to use Linux to control an industrial welding laser, I have no problem with your using Preempt RT" - Linus Torvalds auf dem Kernel Summit 2006
How Preempt RT brings Realtime to Linux?

Remember once again...

Preemption is the most important requirement for a Realtime System
How Preempt RT brings Realtime to Linux?

- Locking Primitives: Spinlocks are replaced by RT Mutexes. Raw Spinlocks behave like the original Spinlocks.
- Interrupt Handlers run in a kernel thread
Preempt RT

Real-time process  User space  Non real-time process

Real-time task  Linux Kernel preemptable  Non real-time task

Hardware

© linutronix GmbH
Latency Measurements on a Cortex A9 platform

- ARM Cortex A9 SOC (Altera Cyclone V)
- System load: 100% CPU load with hackbench
- IRQ tests at 10 kHz with the OSADL Latency Box
- Test duration 12h
Load scenario: hackbench

- Starts \( n \) groups of 20 clients and 20 servers
- Each client sends 100 messages to each server via a socket connection
What has been measured?

Latency and Jitter
Latency measurement

- Channel 1: IRQ source (trigger)
- Channel 2: IRQ response
Userspace Latency: The most important usecase

The most important usecase is the latency of a userspace task. Usually a userspace task needs to be synced with an external event.
Xenomai: latency userspace task
PREEMPT_RT: latency userspace task

![Graph showing occurrences over time](image-url)
PREEMPT_RT: latency userspace task (isolated CPU)
Latency userspace task - comparison

User Space Comparision at 10 kHz

TIME MICROSECONDS

OCCURRENCES
Latency within the Kernel

...or how to compare apples with pears!! ;-)
Latency: Kernel - Xenomai
Latency: Kernel - PREEMPT_RT

![Graph showing latency measurements](image-url)
Latency: Kernel - PREEMPT_RT (isolated CPU)
Latency: Kernel - PREEMPT_RT with FIQ (fast interrupt)
Latency: Kernel - Comparison

Kernel Space Comparison at 10 kHz

Legend:
- rt-irq-noisol
- xn-irq-noisol
- rt-irq-isol
- FIQ

Axes:
- Y-axis: Occurrences
- X-axis: Time Microseconds (0 to 150)

Y-axis labels: 1e+09, 1e+08, 1e+07, 1e+06, 100000, 10000, 1000, 100, 10, 1
Conclusion

- PREEMPT_RT became the de-facto standard for Realtime Linux
- Integration of PREEMPT_RT in Mainline Linux
- Simple usage of PREEMPT_RT
- Microkernels are hard to handle
- For the most common use-cases the Microkernels do NOT have better latencies
- FIQs offer fast latencies, but are hard to handle
Questions?