# Real-Time Linux Training: PREEMPT_RT, Latency and Tuning

A Linux system that meets its deadlines: PREEMPT_RT, real-time scheduling, and a worst case measured and tuned on the real board.

| 2 days | Intermediate to advanced | On site or remote | English or French |
|---|---|---|---|
| Duration | Level | Format | Language |

## Overview

This course shows how to build a Linux system that meets its deadlines. It explains where latencies come from in a standard kernel, what PREEMPT_RT changes now that it is part of mainline Linux, how to write an application that keeps its real-time guarantees, and how to measure, analyse and tune the worst case of a board under load.

The method comes from our work on industrial PCs and controllers running PREEMPT_RT, where the requirement is a number measured over days, not an average.

## What You Will Learn

- Explain where latencies come from and what PREEMPT_RT changes
- Build and configure a real-time kernel for a board
- Choose scheduling policies and priorities for the tasks of a system
- Write an application that keeps its real-time guarantees
- Measure the worst-case latency under load and find its cause with ftrace and rtla
- Tune the system: CPU isolation, interrupts, power management

### Who should attend

Engineers who build products with timing constraints on Linux: industrial control, motion, audio, measurement and communication equipment.

### Prerequisites

Good knowledge of embedded Linux and of C programming with POSIX threads.

## Program

The first day covers the kernel and the scheduler, the second the application, the measurement and the tuning.

### Day 1

- Real-time requirements: deadline, latency, determinism and the worst case
- The sources of latency in a standard kernel
- The preemption models of the kernel
- PREEMPT_RT: threaded interrupt handlers, sleeping locks and priority inheritance, its path to mainline in Linux 6.12
- Building and configuring a real-time kernel
- Scheduling classes and policies: SCHED_FIFO, SCHED_RR, SCHED_DEADLINE, real-time throttling

**Day 2**

- Writing a real-time application: locked memory, prefaulting, absolute-time periodic loops, priority-inheritance mutexes
- Measuring latency with cyclictest under load, over long runs
- Finding the cause of a latency with ftrace and rtla
- Tuning: CPU isolation, interrupt affinity, idle states, CPU frequency, console and firmware
- Beyond PREEMPT_RT: a Cortex-M coprocessor next to Linux
- A method for a real-time project

## Labs and Hardware

Labs run on QEMU machines and on the boards of our lab: Arm 32-bit: STM32MP157C-DK2, BeagleBone Black; Arm 64-bit: Raspberry Pi 4 and 5, NXP i.MX8M Mini, Khadas VIM3; x86: QEMU x86, Generic x86_64 machines, Intel development kits. Other boards, or the hardware of the customer, can be used on request.

Sessions take place on site, in France, Belgium, Switzerland, Germany and beyond, with the hardware provided by LABCSMART, or remotely. Remote attendees work on our build server, each with their own SSH account and workspace, and run the labs on QEMU machines and on real boards. The exercises are drawn from our own client projects. Courses are given in English or French.

## Who Teaches This Course

The course is designed and given by the founder of LABCSMART, an embedded Linux and Linux kernel engineer with more than eleven years of industrial experience. He contributes upstream to the Linux kernel for Renesas RZ SoCs, in the PCIe, audio, Ethernet, thermal, DMA and clock subsystems, and wrote two books published by Packt, *Linux Device Driver Development, Second Edition* (2022) and *Mastering Linux Device Driver Development* (2021), presented on our publications page.

**Plan a session for your team.** Every course is made of independent modules. A team can drop the modules it does not need and add others in their place, and we also build courses on demand around a given SoC, board or project. Contact: contact@labcsmart.com, https://labcsmart.com/company/contact/. Course page: https://labcsmart.com/training/real-time-linux-preempt-rt/