Real-Time Linux with *PREEMPT_RT* training

Course duration

2 days – 16 hours

Language

Materials	English
Oral Lecture	English French

Trainer

One of the following engineers

- Maxime Chevallier

Contact

training@bootlin.com

+33 484 258 097

bootlin

bootlin.com

Onsite training

Audience

Companies and engineers interested in writing and benchmarking real-time applications and drivers on an embedded Linux system.

Training objectives

- Be able to understand the characteristics of a real-time operating system
- Be able to download, build and use the *PREEMPT_RT* patch
- Be able to identify and benchmark the hardware platform in terms of real-time characteristics
- Be able to configure the Linux kernel for deterministic behavior.
- Be able to develop, trace and debug real-time user-space Linux applications.

Prerequisites

- **Knowledge and practice of UNIX or GNU/Linux commands**: participants must be familiar with the Linux command line. Participants lacking experience on this topic should get trained by themselves, for example with our freely available on-line slides.
- **Minimal experience in embedded Linux development**: participants should have a minimal understanding of the architecture of embedded Linux systems: role of the Linux kernel vs. user-space, development of Linux user-space applications in C. Following Bootlin's Embedded Linux course allows to fulfill this pre-requisite.
- **Minimal English language level: B1**, according to the *Common European Framework of References for Languages*, for our sessions in English. See the CEFR grid for self-evaluation.

Pedagogics

- Lectures delivered by the trainer: 50% of the duration
- Practical labs done by participants: 50% of the duration
- Electronic copies of presentations, lab instructions and data files. They are freely available here.

Certificate

Only the participants who have attended all training sessions, and who have scored over 50% of correct answers at the final evaluation will receive a training certificate from Bootlin.

Disabilities

Participants with disabilities who have special needs are invited to contact us at *training@bootlin.com* to discuss adaptations to the training course.

Required equipement

For on-site session delivered at our customer location, our customer must provide:

- Video projector
- One PC computer on each desk (for one or two persons) with at least 16 GB of RAM, and Ubuntu Linux 24.04 installed in a free partition of at least 30 GB
- Distributions other than Ubuntu Linux 24.04 are not supported, and using Linux in a virtual machine is not supported.
- Unfiltered and fast connection to Internet: at least 50 Mbit/s of download bandwidth, and no filtering of web sites or protocols.
- PC computers with valuable data must be backed up before being used in our sessions.

For on-site sessions organized at Bootlin premises, Bootlin provides all the necessary equipment.

Hardware platform for practical labs

STM32MP1 Discovery Kit

One of these Discovery Kits from STMicroelectronics: **STM32MP157A-DK1**, **STM32MP157D-DK1**, **STM32MP157C-DK2** or **STM32MP157F-DK2**

- STM32MP157, dual Cortex-A7 processor from STMicroelectronics
- USB powered
- 512 MB DDR3L RAM
- Gigabit Ethernet port
- 4 USB 2.0 host ports
- 1 USB-C OTG port
- 1 Micro SD slot
- On-board ST-LINK/V2-1 debugger
- Arduino compatible headers
- Audio codec, buttons, LEDs
- LCD touchscreen (DK2 kits only)

Training Schedule

Day 1 - Morning

Lecture	Introduction to Real-Time behaviour and determinism	▪ Definition of a Real-Time Operating System ▪ Specificities of multi-task systems ▪ Common locking and prioritizing patterns ▪ Overview of existing Real-Time Operating Systems ▪ Approaches to bring Real-Time capabilities to Linux
Lecture	The *PREEMPT_RT* patch	▪ History and future of the *PREEMPT_RT* patch ▪ Real-Time improvements from *PREEMPT_RT* in mainline Linux ▪ The internals of *PREEMPT_RT* ▪ Interrupt handling: threaded interrupts, softirqs ▪ Locking primitives: mutexes and spinlocks, sleeping spinlocks ▪ Preemption models
Lab	Building a mainline Linux Kernel with the *PREEMPT_RT* patch	▪ Downloading the Linux Kernel, and applying the patch ▪ Configuring the Kernel ▪ Booting the Kernel on the target hardware

Day 1 - Afternoon

Lecture	Hardware configuration and limitations for Real-Time	▪ Interrupts and deep firmware ▪ Interaction with power management features: CPU frequency scaling and sleep states ▪ DMA
Lecture	Tools: Benchmarking, Stressing and Analyzing	▪ Benchmarking with *cyclictest* ▪ System stressing with *stress-ng* and *hackbench* ▪ The Linux Kernel tracing infrastructure ▪ Latency and scheduling analysis with *ftrace*, *kernelshark* or *LTTng*
Lab	Tools: Benchmarking, Stressing and Analyzing	▪ Usage of benchmarking and stress tools ▪ Common benchmarking techniques ▪ Benchmarking and configuring the hardware platform

Day 2 - Morning

Lecture	Kernel infrastructures and configuration	▪ Good practices when writing Linux kernel drivers ▪ Scheduling policies and priorities: *SCHED_FIFO*, *SCHED_RR*, *SCHED_DEADLINE* ▪ CPU and IRQ Affinity ▪ Memory management ▪ CPU isolation with *isolcpus*
Lecture	Real-Time Applications programming patterns	▪ POSIX real-time API ▪ Thread management and configuration ▪ Memory management: memory allocation and memory locking, stack ▪ Locking patterns: mutexes, priority inheritance ▪ Inter-Process Communication ▪ Signaling

Day 2 - Afternoon

Lab	Debugging a demo application	▪ Make a demo userspace application deterministic ▪ Use the tracing infrastructure to identify the cause of a latency ▪ Learn how to use the POSIX API to manage threads, locking and memory ▪ Learn how to use the CPU affinities and configure the scheduling policy