Embedded Linux System Development Training

An embedded Linux system built from the ground up, with every piece understood, from the first bootloader instruction to the application.

4 days Duration	**Beginner to intermediate** Level	**On site or remote** Format	**English or French** Language

Overview

This course builds an embedded Linux system piece by piece, from the cross-compilation toolchain to a product ready for production. Each component is configured, built and booted on a real board: the bootloader, the Linux kernel and its device tree, the root filesystem and the services that run on it.

What You Will Learn

- Build and use a cross-compilation toolchain
- Configure, build and boot a bootloader and a Linux kernel on a real board
- Describe and customize the hardware of a board with the device tree
- Assemble a root filesystem and choose the right storage and filesystems
- Choose between Buildroot and the Yocto Project for a product
- Prepare a system for production: real time, boot time, security, updates and licensing

Who should attend

Engineers starting with embedded Linux, or moving from microcontrollers to Linux-based systems, and developers who use an embedded Linux board every day and want to understand how it is put together.

Prerequisites

Comfort with the Linux command line and basic C programming. No prior embedded Linux experience is needed.

Program

The program follows the order in which a system is built and started.

Day 1

- Embedded Linux systems, the kernel and the BSP
- Cross-compilation toolchains, building one with crosstool-NG
- Firmware and bootloaders: how x86 and embedded platforms boot, the bootloader landscape, Trusted Firmware-A
- Using U-Boot: the prompt, the environment, starting a kernel, automatic boot

Day 2

- Building and booting the Linux kernel: sources, configuration, modules
- Describing hardware with the device tree: syntax, bindings, pin multiplexing, customizing a board
- The root filesystem: device files, virtual filesystems, a minimal system with BusyBox

Day 3

- Block storage and filesystems, read-only roots and robustness against power cuts
- Raw flash storage with MTD, UBI and UBIFS
- Build systems: Buildroot and the Yocto Project
- System startup with systemd, service supervision and device management

Day 4

- Networking, wireless networks and network services
- Accessing hardware from user space: GPIO, LEDs, I2C, SPI and sensors
- Application development and debugging on the target
- Real-time Linux with PREEMPT_RT and latency measurement
- Boot time and power management
- Security and software updates
- From prototype to product: licensing, maintenance and production

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/embedded-linux-development/