# Linux Kernel and Device Driver Development Training

Linux drivers ready for a product and for upstream review, with the kernel frameworks used the way their maintainers intend.

| | | | |
|---|---|---|---|
| **5 days**<br>Duration | **Intermediate to advanced**<br>Level | **On site or remote**<br>Format | **English or French**<br>Language |

## Overview

This course teaches how to write Linux device drivers that are ready for production and for upstream submission. It starts from the kernel sources and modules, goes through the device model, memory management, interrupts and locking, then writes platform, I2C and PCI drivers that use DMA, clocks, resets and runtime power management as a driver needs them.

## What You Will Learn

- Build, configure and boot the kernel, and write kernel modules
- Write platform, I2C and PCI drivers bound through the device model and the device tree
- Allocate memory, map registers and use DMA safely
- Handle interrupts, deferred work and locking correctly
- Use clocks, resets and runtime power management in a driver
- Debug drivers with printk, ftrace, kprobes and oops analysis

### Who should attend

Engineers who write or maintain Linux drivers for custom hardware, and BSP developers who need to understand the drivers they integrate.

### Prerequisites

Solid C programming and good knowledge of the Linux command line. Notions of embedded Linux help.

## Program

Power management is part of every driver, so it is covered from the driver point of view rather than as a separate topic.

### Day 1

- The Linux kernel, its history and development process
- Getting, configuring and building the kernel, booting it
- Linux kernel modules

### Day 2

- Kernel frameworks and the Linux device model, character drivers
- Describing and accessing hardware: platform drivers, device identification and data
- Memory management: allocation, I/O memory, remapping, endianness

### Day 3

- Sleeping, waiting and scheduling
- Interrupt management and interrupt handlers
- Resource sharing and locking
- I2C device drivers: the bus, the Linux I2C model, the device tree, talking to the device

### Day 4

- DMA: the generic DMA API, DMAEngine, IOMMU and peripheral DMA
- PCI device drivers: enumeration, registration, probing, configuration space, BARs and DMA

### Day 5

- Clocks and resets from the consumer side
- Runtime power management and system sleep callbacks in a driver
- Kernel debugging: printk, kprobes, ftrace, oops and panic analysis, perf

## 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.

## Optional Modules

Each module can be added to the course, or replace some of its parts, for teams that work on a given subsystem.

- Linux audio drivers with ALSA SoC: PCM, controls, DAPM, codec, platform and machine drivers (3 days)
- Linux camera drivers with V4L2: sensors, sub-devices, MIPI CSI-2, controls, the media controller, memory-to-memory devices (3 days)
- PCIe host and endpoint controller drivers (2 days)
- Clock providers and the common clock framework (1 day)

## 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/linux-device-driver-development/