LINUX TRAINING

Developing Linux Device Drivers

Master Linux device driver development. Learn to design, implement, and debug kernel modules for custom hardware.

DURATION
2 hours

LEVEL
Intermediate

FORMAT
Instructor-Led

- Globally Recognized Certification
- Expert-Led Hands-On Training
- Flexible Scheduling

Computer Pride | 1st Floor, ICEA Building, Kenyatta Avenue, Nairobi | www.computer-pride.co.ke

COURSE OVERVIEW

This intermediate-level course, "Developing Linux Device Drivers," is meticulously designed to equip aspiring and experienced system programmers with the essential skills to interact directly with hardware components through the Linux kernel. Participants will delve into the intricacies of kernel programming, mastering the C language within the unique environment of the Linux kernel. The curriculum covers fundamental concepts such as kernel modules, character device drivers, interrupt handling, and memory management, providing a robust foundation for low-level system development.

By the end of this course, you will be proficient in writing, debugging, and deploying your own Linux device drivers. This expertise is invaluable for those looking to specialize in embedded systems, Internet of Things (IoT) development, or any domain requiring deep integration between software and hardware. The practical, hands-on approach ensures you gain the confidence and capability to develop custom solutions, optimize system performance, and troubleshoot complex hardware-related issues, significantly advancing your professional toolkit.

Duration	2 hours
Level	Intermediate
Vendor	Linux

COURSE CURRICULUM

01 Module 1: Introduction to Linux Kernel & Modules

- Linux Kernel Architecture Overview
- Role of Device Drivers in the Kernel
- Kernel Modules: Loading, Unloading, Parameters
- Kernel Programming Fundamentals (printk, memory allocation in kernel space)
- Building and Installing Kernel Modules

02 Module 2: Character Device Drivers

- Character Device Driver Architecture
- Major and Minor Numbers
- File Operations Structure (open, read, write, ioctl, release)
- User-space Interaction with Device Files
- Developing a Simple Character Device Driver

03 Module 3: Interrupt Handling and Timers

- Introduction to Hardware Interrupts and IRQs
- Requesting and Releasing IRQs
- Interrupt Service Routines (ISRs) and Deferred Work (tasklets, workqueues)
- Kernel Timers and Delay Functions
- Handling Concurrency with Interrupts

04 Module 4: Memory Management and Concurrency Control

- Kernel Memory Management (kmalloc, vmalloc, DMA)
- User-Kernel Memory Copy Operations (copy_from_user, copy_to_user)
- Concurrency Control Mechanisms: Spinlocks, Mutexes, Semaphores
- Atomic Operations
- Wait Queues and Blocking Operations

05 Module 5: The Linux Device Model & Platform Drivers

- Understanding the Linux Device Model (kobjects, ksets)
- Platform Devices and Platform Drivers
- Driver Probing and Resource Management
- Introduction to Device Tree Overlays
- Bus-agnostic Driver Development Principles

06 Module 6: Debugging and Best Practices

- Kernel Debugging Techniques (printk, debugfs, dynamic debug)
- Using KDB/KGDB for In-Kernel Debugging (Introduction)
- Error Handling and Robustness in Device Drivers
- Kernel Coding Standards and Guidelines
- Performance Considerations for Drivers

CAREER PATHWAYS

Graduates of this program are prepared for the following roles:

> **Embedded Linux Software Engineer**

> **Linux Kernel Developer**

> **Systems Programmer (Hardware Interaction Specialist)**

FREQUENTLY ASKED QUESTIONS

Q: What foundational programming skills are essential before enrolling in this 'Developing Linux Device Drivers' course?

This intermediate-level course assumes a strong proficiency in C programming, including pointers, data structures, memory management, and build systems. Familiarity with basic Linux command-line operations, file system navigation, and general system administration concepts is also highly recommended to maximize your learning experience and effectively participate in the practical exercises.

Q: Will this course cover how to interact with specific hardware interfaces like I2C, SPI, or USB?

While the course primarily focuses on the core concepts and methodologies of writing character device drivers and managing fundamental kernel resources, it provides a robust foundation directly applicable to bus-specific drivers. Module 5 introduces the Linux Device Model and platform drivers, giving you the theoretical framework to understand how such interfaces integrate. Although we won't deep-dive into writing full drivers for I2C, SPI, or USB, the principles learned will equip you to confidently approach these with further independent study or specialized training.

Q: How does mastering Linux device driver development impact career opportunities in embedded systems or IoT?

Developing Linux device drivers is a pivotal skill for embedded systems and IoT, as it directly enables engineers to integrate custom or new hardware components with the Linux operating system. This expertise is highly sought after for roles involving board bring-up, optimizing hardware performance, writing drivers for sensors, actuators, and various peripherals, and ensuring stable and efficient interaction between the kernel and proprietary hardware. It significantly enhances your value in industries building custom hardware solutions, smart devices, and specialized embedded

platforms.

ABOUT COMPUTER PRIDE

East Africa's Premier IT Training Center

With over 15 years of excellence, Computer Pride delivers globally recognized certification programs from leading technology vendors. Located on Kenyatta Avenue in Nairobi's CBD, our state-of-the-art facilities and expert instructors have empowered thousands of professionals across East Africa.

15+	10,000+	50+
Years of Excellence	Professionals Trained	Vendor Certifications

READY TO ENROL?

Take the next step in your technology career.

Location
1st Flr, ICEA Building
Kenyatta Ave, Nairobi

Email
info@computer-pride.co.ke

Phone
+254 705 900 900

www.computer-pride.co.ke