LINUX TRAINING

Developing Embedded Linux Device Drivers

Master Embedded Linux device driver development. Learn to build, debug, and optimize drivers 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 Embedded Linux Device Drivers," offers a deep dive into the essential skills required to design, implement, and debug device drivers for embedded Linux systems. Participants will gain a comprehensive understanding of the Linux kernel's architecture relevant to device drivers, focusing on how hardware interacts with the operating system. The curriculum covers core concepts such as kernel modules, character device drivers, platform drivers, and advanced topics like interrupt handling, memory management, and working with device trees, empowering you to integrate custom hardware seamlessly into embedded Linux environments.

By mastering embedded Linux device driver development, you will unlock critical capabilities for a wide array of high-demand technology sectors. This course is meticulously designed to equip you with the practical expertise needed to build robust and efficient drivers, enhancing system performance and reliability for specialized hardware. You will learn to navigate the intricacies of kernel programming, debug complex issues, and adhere to industry best practices, making you an invaluable asset in fields ranging from IoT and automotive to industrial automation and consumer electronics development.

Duration	2 hours
Level	Intermediate
Vendor	Linux

COURSE CURRICULUM

01 Module 1: Introduction to Embedded Linux and Device Drivers

- Overview of Embedded Linux Architecture
- Role of Device Drivers in Embedded Systems
- Linux Kernel Modules: Loading, Unloading, Parameters
- Kernel vs. User Space Interaction

02 Module 2: Basic Driver Concepts and Character Devices

- Kernel Programming Fundamentals (printk, jiffies, kernel data structures)
- Character Device Driver Structure
- Registering and Unregistering Character Devices
- File Operations (open, release, read, write, ioctl)

03 Module 3: Platform Drivers and Device Tree Integration

- Understanding Platform Bus and Platform Devices
- Implementing Platform Drivers
- Introduction to Device Tree (DT) Syntax and Structure
- Parsing Device Tree Properties in Drivers

04 Module 4: Interrupt Handling and Timers

- Interrupt Basics and IRQ Management
- Implementing Interrupt Service Routines (ISRs)
- Bottom Halves: Tasklets and Workqueues
- Kernel Timers and Delayed Work

05 Module 5: Memory Management and I/O Access

- Kernel Memory Allocation (kmalloc, vmalloc)
- Direct Memory Access (DMA) Concepts and API
- I/O Port and Memory-Mapped I/O Access
- Concurrency and Synchronization Mechanisms (Spinlocks, Mutexes, Semaphores)

06 Module 6: Debugging and Advanced Driver Topics

- Common Debugging Techniques (printk, debugfs, ftrace)
- Error Handling in Device Drivers
- Introduction to Other Bus Drivers (I2C, SPI, USB basics)
- Driver Development Best Practices and Coding Standards

CAREER PATHWAYS

Graduates of this program are prepared for the following roles:

> **Embedded Linux Software Engineer**

> **Linux Kernel Driver Developer**

> **IoT Device Firmware Engineer**

FREQUENTLY ASKED QUESTIONS

Q: What foundational knowledge should I have before enrolling in this "Developing Embedded Linux Device Drivers" course?

Participants should have a solid understanding of the C programming language, experience with Linux command-line operations, and basic familiarity with embedded systems concepts. Knowledge of Linux system programming is beneficial but not strictly required as kernel-specific concepts will be introduced.

Q: How will this course specifically enhance my ability to troubleshoot and debug embedded Linux hardware issues in real-world projects?

This course dedicates significant attention to debugging techniques within the Linux kernel environment. You'll learn to use tools like printk, debugfs, ftrace, and various kernel-level debugging methodologies, enabling you to effectively diagnose and resolve complex hardware-software interaction problems specific to embedded Linux devices, significantly reducing development cycles.

Q: What types of embedded hardware or architectures are primarily focused on during the driver development exercises in this course?

While the principles taught are universally applicable to various architectures (ARM, MIPS, etc.), the course will primarily use examples and practical exercises targeted towards common ARM-based embedded platforms, as they are prevalent in modern IoT and industrial embedded systems. The focus is on teaching driver frameworks that are architecture-agnostic where possible, but practical application will lean towards ARM examples for hands-on learning.

ABOUT COMPUTER PRIDE

computerpride

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