

# Linux Device Drivers

OS Concepts • Driver Development

---



## What is Device Driver?

A device driver simplifies programming by acting as translator between a hardware device and the applications or operating systems that use it.

Drivers are hardware-dependent and operating-system-specific.

Device drivers are a layer between the kernel and the hardware that it controls. As such it is a very useful abstraction, since it greatly simplifies the kernel: instead of having the kernel talk to each device itself it exports a well defined interface and leaves this task to the individual device drivers..

## Why Linux Device Drivers?

Writing Linux drivers is easy. .... And fun!

Source is freely available as it's in Linux OS. Since Linux follows the UNIX model, and in UNIX everything is a file, users talk with device drivers through device files

## Difference between Windows & Linux Device Drivers

Given that the language of implementation is C/C++ the only difference would be the way in which the drivers interact with the kernel code.

This is where you would notice the biggest differences because the Windows API is GUI aware whereas the Linux API (POSIX) is not GUI aware.

One other difference however is that Linux drivers can be loaded as modules onto a running kernel without needing a restart

## Examples where Device Drivers are required

- printers
- video adapters
- network cards
- sound cards
- local buses of various sorts — in particular, for bus mastering on modern systems
- low-bandwidth I/O buses of various sorts (for pointing devices such as mice, keyboards, USB, etc.)
- computer storage devices such as hard disk, CD-ROM and floppy disk buses (ATA, SATA, SCSI)
- implementing support for different file systems
- image scanners
- digital cameras

## Who can join the course?

One can join with prerequisite of having knowledge of C programming.

## Course Contents:

---

- **An Introduction to Device Drivers** - Major and minor number, Allocation/deallocation of device number
- **Linux Kernel Extensibility Features** - Compiling and loading the kernel module, Difference between
- **kernel module and application**, Kernel symbol table, Kernel module parameter and Shared libraries
- **Device Drivers** - Character drivers, Allocating memory and I/O ports, Fundamental driver operations,
- **Character device registration**, Advanced character driver operations
- **Concurrent Programming** - Race condition, Solutions to race condition, Mutex semaphore and Nanosleep usage
- **Exception and Interrupt Handling** - IDT and GDT in Intel PC, Proc entry for IDT information,
- **Advanced PIC**, MP floating pointer and configuration table, Task let mechanism and System-calls
- **Kernel Timing Issues** -Jiffies and Container of macro
- **Case study:** UART/USB/ TTY/ PCI device driver
- **Debugging Techniques** - Printk, Proc and Trace
- **Block Drivers**, Network Drivers, TTY Drivers.

## Laboratory:

- Building and compiling Linux kernel module
- Creating proc entry in Linux for CMOS device data for normal user using dynamic module concept
- Race condition and concurrent programming
- Interrupt handling in Linux
- Kernel timers
- Writing device driver in Linux for CMOS device on PC
- UART/USB/TTY/PCI device driver development

---

### Indus Institute of Embedded Technologies (IET), Pune.

2nd Floor, Above Bank of Maharashtra,  
Near Modern Cafe  
Shivaji Nagar, Pune

Website- [www.embeddedsystemspune.com](http://www.embeddedsystemspune.com)

Email: [info@bicard.org](mailto:info@bicard.org)

Office No: **020-30209696**

Mob No: **9372202549**

---