VECTOR INDIA

DRIVES YOU TO INDUSTRY

DEVICE DRIVERS

729
MNCS HIRED IN 2024

All India Educational Excellence Award
Winner For 3 years in a row

1611
STUDENTS RECRUITED IN 2024

THE INSTITUTE

- **Directors with over a decade of rich industry experience in Design Development, Training & Recruitment.**
- **A state-of-the-art Programming Lab with 1:1 student to System ratio.**

PRE REQUISITE

- **Good in C Programming and Linux User space**

INSTRUCTIONS

- **Participants can attend training with windows/Ubuntu OS machine. (Our lab team will support to install VM Linux)**
- **All lab activities will be conducted on Raspberry PI platform. Participants can access our VectorRaspberry PI boards remotely for practicals.**

CLASS TIMINGS

7.00 PM TO 8.30 PM

MODE OF TRAINING	DURATION
ONLINE	4 WEEKS

DEVICE DRIVERS

Learn Device Drivers programming at Vector India to gain industry-relevant expertise in developing low-level software for hardware interaction, boosting your embedded systems career.

Why Vector India

19 yrs

Experience in embedded systems training and producing industry-ready talent

1,00,000+

Alumni, and 650+ corporate collaborations

100%

Genuine placement assistance with quality experiential training

TOPICS

LINUX KERNEL COMPILATION ON X86 DESKTOP MACHINE (OSL FLOW MODEL)

- Linux OS (user space) vs Linux Kernel.
- Types of Devices in Linux OS.
- Kernel Source Tree with git and get commands.
- Explore Kernel source tree structure.
- Configuring, Building and Installing customized Kernel.
- Boot process on X86 machine.

EMBEDDED LINUX

Overview of Embedded Linux System Architecture, Boot loader, root file system, Boot process on ARM Cross compilation, Tool Chain installation.

MODULES & DEVICE DRIVERS

- Mechanism vs Policy
- How Applications Use Device Drivers
- Walking Through a System Call Accessing a Device
- Error Numbers
- printk()
- The module driver() Macros
- Module parameters, Exporting Modules

CHARACTER DEVICES

- Device Nodes
- Major and Minor Numbers
- Reserving Major/Minor Numbers
- Accessing the Device Node
- Registering the Device
- udev
- dev printk() and Associates
- file operations Structure
- Driver Entry Points
- The file and inode Structures
- Miscellaneous Character Drivers

EMBEDDED LINUX
BUILD SYSTEM WITH YOCTO

- Yocto poky reference build system.
- Building a system image.
- Writing a minimal recipe, Adding dependencies.
- Development workflow with bitbake.
- Adding the custom application.
- Adding the custom library dependent application.
- Adding custom kernel module.
- Change the kernel version and apply kernel patches

MEMORY MANAGEMENT AND ALLOCATION

- Virtual and Physical Memory, Memory Zones
- Page Tables, kmalloc(), get free pages()
- vmalloc(), VM Split, VMA basics
- Slabs and Cache Allocations

MEMORY MAPPED I/O AND I/O MAPPED I/O

- Transferring Between Spaces
- put(get) user() and copy to(from) user()
- Direct Transfer: Kernel I/O and Memory Mapping
- Mapping User Pages, Memory Mapping
- User-Space Functions for mmap(), Driver Entry Point for mmap()
- Accessing Files from the Kernel, Memory Barriers
- Allocating and Mapping I/O Memory, Accessing I/O Memory

INTERRUPT HANDLING

- What are Interrupts and Exceptions?
- Exceptions, Asynchronous Interrupts, MSI
- Enabling/Disabling Interrupts
- What You Cannot Do at Interrupt Time
- IRQ Data Structures, Installing an Interrupt Handler
- Top and Bottom Halves, Softirqs, Tasklets,Work Queues
- New Work Queue API, Creating Kernel Threads
- Threaded Interrupt Handlers
- 1.h Interrupt Handling in User-Space

UNIFIED DEVICE MODEL AND SYSFS

- Unified Device Model, Basic Structures, Real Devices
- Sysfs, kset and kobject examples

DEVICE TREES

- What are Device Trees?
- What Device Trees Do and What They Do Not Do
- Device Tree Syntax
- Device Tree Walk Through
- Device Tree Bindings
- Device Tree support in Boot Loaders
- Using Device Tree Data in Drivers
- Coexistence and Conversion of Old Drivers

PLATFORM DRIVERS

- What are Platform Drivers?
- Main Data Structures, Registering Platform Devices
- An Example, Hardcoded Platform Data
- The New Way: Device Trees

KERNEL SYNCHRONIZATION

- Critical section, Mutex lock
- Semaphore, Spin lock, Kernel threads
- Synchronization in kernel threads, wait events

I2C AND SPI CLIENT DRIVER

- I2C subsystem
- I2C Send/Receive data
- SPI Subsystem

PCI

- What is PCI?
- PCI Device Drivers, Locating PCI Devices
- Accessing Configuration Space
- Accessing I/O and Memory Spaces PCI Express
- PCI DMA, Allocate consistent DMA
- Scatter and gather allocation, PCI interrupt handlers
- PCI utilities

USB DRIVERS

- What is USB?
- USB Topology
- Terminology
- Endpoints
- Descriptors
- USB Device Classes
- USB Support in Linux
- Registering USB Device Drivers
- Moving Data

BLOCK DRIVERS

- What are Block Drivers?
- Buffering
- Registering a Block Driver
- gendisk Structure
- Request Handling

MONITORING AND DEBUGGING

- **Debuginfo Packages**
- **Tracing and Profiling**
- **Sysctl**
- **SysRq Key**
- **Oops Messages debugging**
- **Kernel Debuggers**
- **Debugfs**
- **Use perf, eBPF, addr2line, kprobe,**
- **Debug with performance utilities,**
- **Explore phoronix utility**
- **Kernel Core Dumps**

HYDERABAD

#502, 5th floor, Nagasuri Plaza(Bank Of India Building) Behind HUDA Maithrivanam, Ameerpet, Hyderabad-500038

Ph: 040 2373 6669, Cell: +91 98666 66699

Email: info@vectorindia.org

BENGALURU

33/49, 27th Cross, 12th Main Jayanagar 4th Block
Bengaluru - 560011

Ph: 080 2654 6474, Cell: +91 87624 56789

Email: info.blr@vectorindia.org

CHENNAI

179, 1st Main Road , Nehru Nagar, Lane Opp to Turyaa Hotel, Perungudi, Chennai - 600096

Ph: 044 2454 3969, Cell: +91 94442 22459

Email: info.chen@vectorindia.org

Vector India Pvt.Ltd. **vectorindiainstitute**

@VectorInstitute **@Vector_India**

@VectorIndia9

VECTOR INDIA

www.vectorindia.org