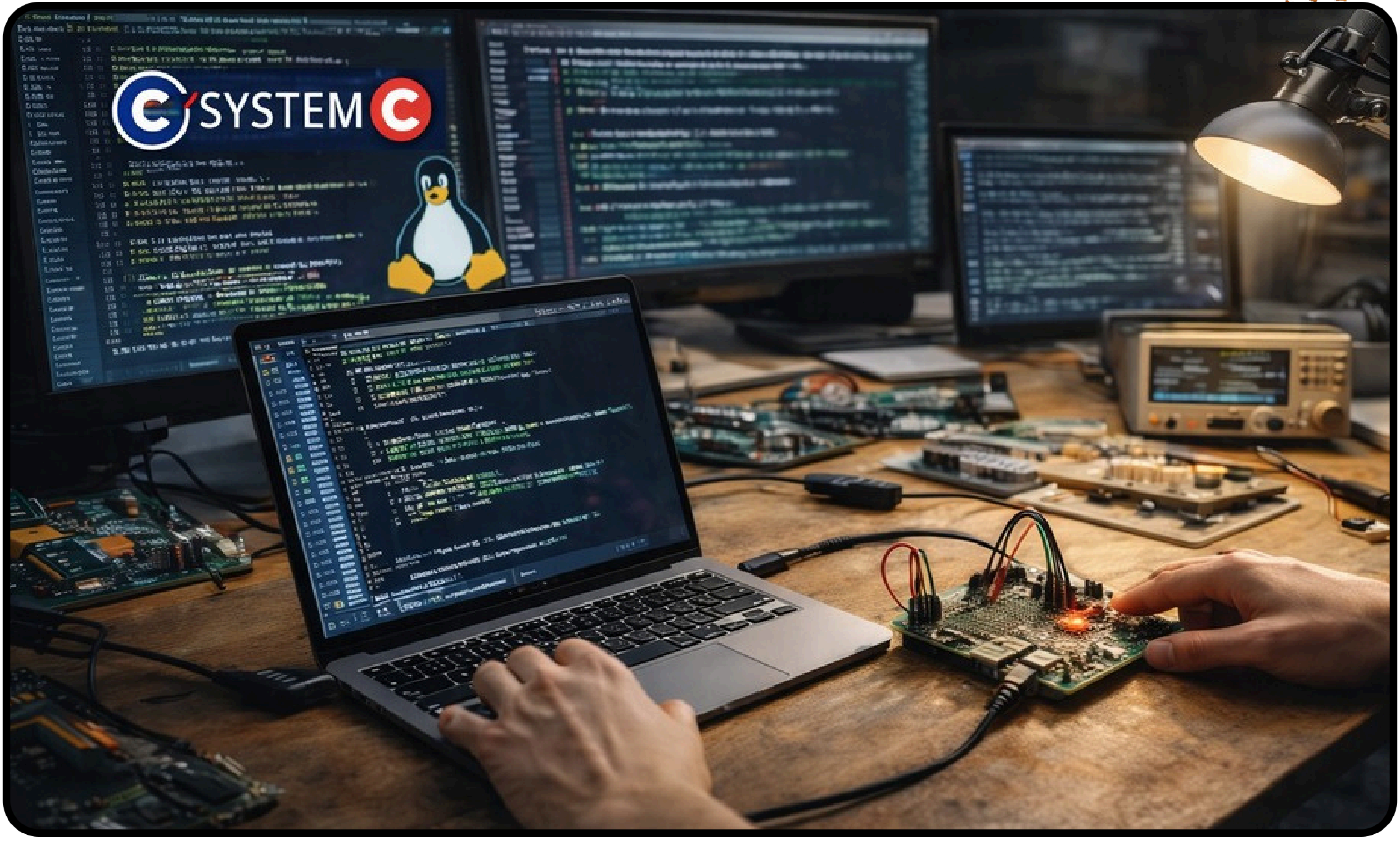
CHENNAI
INSTITUTE OF TECHNOLOGY
(Autonomous)

EMBEDDED LINUX

INTRODUCTION

Embedded Linux is a widely used operatingsystem inmodern embedded systems, supporting applications such as smartphones, routers, industrial controllers, automotive systems, and IoT devices. Designed to run efficiently on resource-constrained hardware, Embedded Linux offers flexibility, reliability, and direct access to hardware resources through the Linux kernel.



ABSTRACT

This workshop focuses on introducingparticipantsto the fundamentals of Embedded Linux with a strong emphasis on character device driver development. Participants will learn how to design and implement a simple character device driver that enables communication between user-space applications and hardware through kernel-level access. The workshop also covers the process of cross-compiling device drivers for target platforms such as the Raspberry Pi and other embedded processors, ensuring proper alignment with the target system architecture. Through guided hands-on sessions, participants will gain practical experience in compiling kernel modules, loading and unloading drivers, and testing driver functionality on embedded hardware. These activities help participants understand how Linux device drivers act as a critical bridge between software applications and underlying hardware components. By the end of the workshop, participants will have a foundational understanding of embedded Linux workflows, cross-compilation techniques, and driver development practices, preparing them for further exploration in embedded system design and real-world industry applications.

SESSION OUTLINE

- **WORKSHOP RUNTIME:** 120 MINUTES
- **HARDWARE REQUIRED :** USER LAPTOP (any processor), RASPBERRY PI 4/5
- **SOFTWARE REQUIRED :** Ubuntu OS, GCC Compiler, Cross Compilation Toolchain (ARM / AArch64) , GNU Make, RASPBERRY PI 4/5

(All Requirements will be provided by the Embedded Linux Workshop Committee)

SESSION	CONTENTS
INTRODUCTION TO LINUX	Basics of Linux, evolution of the Linux kernel, history of Linux operating systems, and why Linux dominates embedded systems
LINUX OS ARCHITECTURE & DRIVER EVOLUTION	Overview of Linux OS blocks including user space, kernel space, system calls, and the evolution of device drivers in Linux
CHARACTER DEVICE DRIVER FUNDAMENTALS	Introduction to character drivers, their use cases, file operations (open, read, write, close), and building a basic character device driver
CROSS COMPILATION & RASPBERRY PI ARCHITECTURE	Basics of cross compilation, toolchain setup, Raspberry Pi processor architecture, and alignment with target system architecture
DRIVER DEPLOYMENT ON RASPBERRY PI	Deployment of the character device driver for Raspberry Pi 5 architecture, loading and unloading kernel modules, and verifying driver functionality

PREREQUIREMENTS

- This workshop is designed to be beginner-friendly and suitable for students from various technical backgrounds. No prior experience in Embedded Linux or device driver development is required. A basic understanding of C programming and computer fundamentals will be helpful but is not mandatory. Participants are expected to have an interest in operating systems, embedded systems, and low-level programming.
- Participants are encouraged to bring their own laptops with a Linux-based operating system (Ubuntu preferred) installed to ensure smooth participation in hands-on sessions.
- Due to limited availability of Linux-configured systems, only a minimum number of laptops can be provided by the organizers. Participants are therefore strongly encouraged to bring their own laptops with a Linux operating system installed to avoid any

  inconvenience and ensure smooth participation inthehands-on sessions.

CONCLUSION

This Embedded Linux workshop provided participants with a strong foundation in Linux architecture, character device driver development, cross-compilation, and Raspberry Pi deployment. Through hands-on practice, participants gained practical insight into kernel-level programming and real-world embedded system workflows.