MERCURY

Advanced Embedded Linux Programming and System

Dubai - UAE: +971 4 450 5697
Istanbul - Türkiye: +90 539 599 1206
Amman - Jordan: +962 79 712 3347

MERCURY

Advanced Embedded Linux Programming and System

Introduction:

Embedded Linux systems are at the heart of modern digital devices, from industrial controllers and automotive electronics to IoT platforms and mobile communications infrastructure. This Advanced Embedded Linux Programming and System course empowers professionals with practical, in-depth knowledge of developing, configuring, and optimizing embedded Linux environments.

Participants will dive into low-level programming, kernel modules, bootloader configuration, and real-time performance tuning. The Advanced Embedded Linux Programming and System program delivers hands-on insights into Linux-based system architecture, peripheral integration, and advanced debugging methods. It is ideal for engineers and developers aiming to elevate their embedded system skills.

This Advanced Embedded Linux Programming and System training equips attendees with robust capabilities to handle real-world embedded Linux development challenges. Participants will be able to build, deploy, and maintain secure and efficient embedded Linux systems in production environments.

Targeted Groups:

This Advanced Embedded Linux Programming and System training targets professionals seeking specialized knowledge and skills:

- Embedded systems engineers are working on Linux-based devices.
- Firmware developers are transitioning to Linux platforms.
- System programmers require real-time system optimization skills.
- Software developers are building hardware-software integrated applications.
- DevOps or QA engineers are involved in testing embedded products.
- Electrical and electronics engineers seeking experience with the Linux kernel.
- Technical project leads oversee the implementation of Linux systems.
- IoT developers are working on Linux-enabled smart devices.
- Mechatronics and robotics engineers use Linux-based boards.

Targeted Competencies:

Participants will gain the following competencies during the Advanced Embedded Linux Programming and System program:

- Proficiency in embedded Linux system architecture.
- Expertise in Linux kernel customization and driver development.
- Ability to cross-compile and deploy Linux for ARM/SoC boards.
- Confidence in debugging using GDB, strace, and perf.
- Skills in configuring bootloaders U-Boot and root filesystems.
- Capacity to manage hardware-software integration.
- Aptitude in real-time and low-latency Linux tuning.
- Competence in embedded Linux system security strategies.

Dubai - UAE: +971 4 450 5697
Istanbul - Türkiye: +90 539 599 1206
Amman - Jordan: +962 79 712 3347

MERCURY

Course Objectives:

Participants will achieve the following objectives by completing the Advanced Embedded Linux Programming and System course:

- Understand Linux architecture and the constraints of embedded systems.
- Develop custom Linux kernel modules and drivers.
- Configure and compile custom embedded Linux distributions.
- Manage bootloaders and hardware initialization sequences.
- Interface with GPIO, I2C, SPI, and other peripherals.
- Optimize system performance and memory usage.
- Implement security features for embedded Linux systems.
- Use cross-compilation techniques and toolchains effectively.
- Apply real-time Linux extensions and scheduling techniques.
- Conduct advanced debugging and performance profiling.
- Develop startup scripts and embedded shell automation.
- Handle device tree and kernel configuration challenges.
- Integrate Linux with embedded networking protocols.
- Develop secure over-the-air OTA update mechanisms for Linux-based devices.

Course Content:

Unit 1: Linux System Architecture and Embedded Concepts:

- Introduction to Linux for embedded systems.
- Overview of embedded Linux distributions Yocto, Buildroot.
- Comparison between desktop and embedded Linux.
- Understanding memory models and resource constraints.
- Exploring the Linux kernel structure and process model.
- Linux device model and user-space interactions.
- Filesystem Types and Partitioning for Embedded Devices.
- Role of bootloaders in system initialization.
- Handling system calls, daemons, and init systems.

Unit 2: Kernel Development and Device Driver Programming:

- Building and customizing the Linux kernel.
- Writing and compiling loadable kernel modules LKM.
- Developing character and block device drivers.
- Using sysfs, procfs, and udev for device management.
- Managing kernel symbols and exporting interfaces.
- Kernel debugging with printk, dynamic debug, and tracepoints.
- Working with interrupts, ISRs, and shared IRQs.
- Handling DMA and memory-mapped I/O.
- Device tree structure and overlay implementation.

Dubai - UAE: +971 4 450 5697
Istanbul - Türkiye: +90 539 599 1206
Amman - Jordan: +962 79 712 3347

Unit 3: Embedded Linux Build Systems and Bootloaders:

- Introduction to cross-compilation toolchains GCC, Clang.
- Creating minimal Linux root filesystems.
- Using Yocto Project for embedded Linux builds.
- Buildroot vs. Yocto: Use Cases and Configurations.
- Bootloader Concepts: U-Boot Overview and Environment Variables.
- Customizing U-Boot for different hardware platforms.
- Kernel boot parameters and command line debugging.
- Secure boot and chain-of-trust principles.
- Handling multi-stage booting in embedded systems.

Unit 4: Peripheral Interfacing and Real-Time Linux:

- Accessing GPIO, UART, I2C, and SPI interfaces.
- Working with PWM, ADCs, and timers in Linux.
- Using the Linux IIO Industrial I/O subsystem.
- Writing user-space drivers with sysfs or mmap.
- Implementing real-time scheduling in Linux PREEMPT_RT.
- Comparing soft real-time vs hard real-time requirements.
- Tuning latency and response time in embedded kernels.
- Measuring system performance with ftrace and perf tools.
- Synchronization primitives: mutexes, semaphores, spinlocks.

Unit 5: Security, Debugging, and Deployment Strategies:

- Applying secure coding practices in kernel space.
- Managing user permissions and capability models.
- Securing file systems and restricting execution.
- Kernel hardening techniques and access control frameworks.
- Debugging with gdbserver, Valgrind, and strace.
- Using crash dumps and kernel oops analysis.
- Strategies for OTA over-the-air updates in embedded systems.
- Packaging and deploying embedded Linux firmware.
- Case studies: industrial, automotive, and medical applications.

Final Insights & Key Takeaways:

The Advanced Embedded Linux Programming and System course prepares professionals to tackle real-world development challenges across industries. From deep kernel work to hardware integration and real-time optimizations, learners will master end-to-end skills. Participants will build secure, high-performance, production-ready Linux embedded systems. It bridges theory with practice through hands-on labs and real-device simulations.