

Advanced Embedded Linux and BSP Development

Course 40940 – 40 Hours

Overview

Linux is a great OS for embedded devices. Good knowledge of linux layers from the kernel to the user application is a key to the success of the final product. Porting linux to a custom board can also be a complex task depends on the differences between the evaluation board and the custom board.

This course goes deep inside the development process of an Embedded linux system covers many practical tasks in user space and kernel space. It covers how to port linux to a custom board, how to use linux effectively and some patterns and good practices

On Completion, Delegates will be able to

- Building BSP (board support package)
- Advanced topics in user space programming
- Linux network subsystem
- Hard real-time in linux
- Practices and patterns

Who Should Attend

Embedded Linux developers and software engineers who wants to write more effective code in user and kernel space

Prerequisites

Delegates should have a working knowledge with embedded Linux (course 91100)

Course Contents

Embedded Linux Refresher

- Processes and threads
- Synchronization and IPC
- Writing a character device driver

User Space Programming – Advanced Topics

- Writing a user space device drivers
- Using select/poll/epoll
- Patterns and practices
 - Processes and threads
 - Synchronization objects and IPC
- Handling signals
- Advanced networking programming
- Debugging user space applications
- Simulating custom hardware

Linux Kernel

- Kernel configuration
- Special configuration options
- Applying external patches
- Building the kernel
- Debugging configuration options
- Kernel boot process in details
- Adding a kernel module statically
- Debugging kernel components

File System

- Building file system - good practices
- Choosing the required components for development and production
- Building FS using buildroot
- Choosing the best FS type
- Flash file systems
- Initramfs

Building Linux Board Support Package (BSP)

- Identifying the differences between the evaluation and the custom boards
- Adapting the boot loader to the new board
- Initialize the hardware properly
- CPU, clock, timers and interrupts
- Memory devices - flash and ram
- Peripherals
- Platform devices and drivers
- Adding support for completely new hardware
- I2C and SPI (user/kernel)

Moving Data between Kernel and User Space

- Mapping pages
- Net-links
- UDP sockets
- Ioctl
- Virtual file systems
- Signals and events

The Block layer and Linux VFS

- Block layer
- The page cache
- Linux VFS
- Writing linux file system

Linux Networking Internals

- Network subsystem
- Inside the network stack
- Frame transmission and reception
- Writing a network device driver
- Writing network filters and firewall
- Handling packets on user space
- Adding custom protocols

Debugging and Trace Tools

- Debugging tools
- Profiling tools - kernel and user
- Tracing tools – kernel and user
- Misc tools in the development process

Linux and Hard Real-time

- Real time requirements
- Kernel preemption modes
- RT-Preempt patch
- Xenomai

Graphics, GUI and Multimedia

- QT
- DirectFB
- SDL
- V4L
- Other useful libraries