

# Develop Linux Based Embedded Systems

*During the last years, the Embedded Systems used in many applications have gained in performance and complexity. More and more functionality have been added as requirements increase and this has been possible as a result of the increase in hardware performance.*

## Course: Develop Linux Based Embedded Systems Day(s): 3

The evolution has moved Embedded Systems from simple fixed function devices, like control and measurement of a simple application, to full feature, multi application devices with rich user interfaces (GUIs and web servers) and standard "PC" communication protocols like TCP/IP.

Linux, which is basically a UNIX, initially designed for PCs using Intel's 386 CPU, got early on a design that made it suitable to port to other CPU architectures as well. Many of today's powerful CPUs for Embedded Systems have passed the 386 in performance and are well beyond it. This makes it possible and attractive to use Linux in Embedded Systems.

### Goal

You will learn how to use a complete set of "Best of Breed" development tools to develop an Embedded Linux system. Emphasis will be on using and configure the breadth of functionality and applications available for Linux.

### Participants

If you quickly and efficiently want to get going with the development of an Embedded Linux system, then this training is for you.

### Previous knowledge

You know programming and have basic knowledge in the C-programming language, equal to the "C programming for embedded systems, part I" training.

You should also have some prior experience using Linux/UNIX as a user, and have some experience on how to use development environments and debuggers for Embedded Systems.

### Practical exercises / Tools

Approximately half of the time will be on hands-on exercises. They have been designed to highlight the development process for Embedded Linux projects.

We use a PC as host for the development environment and connect to an ARM-based target system (OMAP5912 OSK). On the target we run Embedded Linux.

We will develop and debug the applications using both hardware debugger (Lauterbach) and software debugger (for example GDB).

## Content

### Introduction

- What's Linux?

### Overview

- Training contents and design. Terminology
- Kernel, distribution, host/target system, tool chain, device drivers, file system, memory management and protection.

### System design

- What kind of system could be an Embedded Linux candidate?
- Linux and Real-Time.
- Eclipse based tools.

### Efficient development methods

- Difference and similarities in development methods.

### Tools

- Suitable development environment.

### Configuration

- Kernel configuration and target file system creation.

### Dependencies

- Standards and conventions.
- FHS (File system Hierarchy Standard).
- Runlevels. Startup scripts.

### Application Development

- How to develop your own Embedded Linux application. Static and dynamic libraries.

### System services and device drivers

- Network services, TCP/IP.

- Dynamic loading of modules.

### Deployment

- Different deployment models.
- ROM/flash, network, root disk.

### Debugging

- The difference between debugging the kernel and applications.
- File formats and debug information.
- How to download/debug applications on a target system?
- Profiling.
- Finding memory leaks.

### Resources

- Where can I find more information?