Stand 10.06.2026

Embedded-Linux-Architektur: Kernel-Treiberentwicklung - Live-Online-Training

**Ziele -
Ihr Nutzen**

Wie entwickle ich einen Kernel-Treiber? Auf was muss ich bei Embedded- und Echtzeit-Systemen achten?

Essentiell für die Entwicklung eines performanten Treibers ist ein grundlegendes Verständnis der Kernel-Architektur. Genau hier setzt das Training an.

Zuerst wird ein Überblick über den Kernel-Aufbau gegeben und dann die für Embedded-Systeme relevanten Teile aufeinander aufbauend detailliert beleuchtet.

Aus diesen Puzzleteilen ergibt sich eine Gesamtsicht auf das Betriebssystem, wie sie für eine professionelle Treiberentwicklung vonnöten ist.

In der Übungsaufgabe ist ein Grundgerüst für einen Kernel-Treiber gegeben; dieses wird sukzessive um die besprochenen Mechanismen erweitert.

Am Ende des Trainings haben Sie einen kompletten Treiber erstellt und sind in der Lage, in Ihrem Projekt Treiber zu entwickeln.

Teilnehmer

Software-Entwickler, Software-Architekten

Voraussetzungen

Das Niveau dieses Trainings setzt die Kenntnisse voraus, wie sie im Training "Embedded Echtzeit-Linux" vermittelt werden.

Live Online Training

* Preis je Teilnehmer, in Euro zzgl. USt.

Anmeldecode: L-LIN-AR

Präsenz-Training - Deutsch

Termin	Dauer
26.10. – 29.10.2026	4 Tage

Live-Online - Englisch

Dauer
4 Tage

Präsenz-Training - Englisch

Dauer
4 Tage

Stand 10.06.2026

Embedded-Linux-Architektur: Kernel-Treiberentwicklung - Live-Online-Training

Inhalt

Linux-Kernel Grundlagen

- System-Schnittstelle, Privilegstufen
- Virtuelles Filesystem, Adressräume
- Gerätetreiber-Klassen (Character, Block, Net)
- Kernel-Module

Character-Device-Treiber

- Implementierung der Datei-Schnittstelle
- Device Nodes
- Udev-Dämon
- Hardware-Zugriff; Register, IO-Memory, DMA
- /proc- und /sys-Filesystem; Verwendung im Kernel-Treiber

Scheduling

- Scheduling-Klassen
- Prozesse und Threads, Kernel Threads
- Wait Queue; unterbrechbares Warten

Interrupts

- Interrupt Service Routine
- Sekundärreaktionen (SoftIRQ, Tasklet, Kernel Timer)
- High-Resolution-Timer (hrtimer)

Synchronisierungsmechanismen

- Atomare Variablen
- Preemption Sperre, Interrupt-Sperre
- Ringspeicher, Kernel-FIFO
- Semaphore, Mutex, RW-Semaphore
- Completion
- Spin Lock, RW-Lock, Sequence Lock
- Diagnose von Lockingproblemen

Speicherverwaltung

- Speicherschutz, Memory Management Unit (MMU)
- Speichertypen, DMA, High Memory
- Verwaltung physikalischen Speichers
- SLAB-Allocator, Kernel-Malloc
- Datenaustausch mit Userspace, Memory Mapping

Hardware

- Alle Übungsaufgaben werden auf dem phyBOARD mit Arm Cortex®-A8 (AM-335x) unter Verwendung von frei zugänglichen Open-Source-Tools durchgeführt (Remote-Zugang).