

## Embedded-Linux-Architektur: Kernel-Treiberentwicklung

### 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.

## Embedded-Linux-Architektur: Kernel-Treiberentwicklung

### 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.
- Dieses Übungsboard können die Teilnehmer zur weiteren Vertiefung des Gelernten behalten.

**Offenes Training**

<b>Termin</b>	<b>Preis *</b>	<b>Dauer</b>
20.04.2020 – 23.04.2020	2.200,00 €	4 Tage
06.07.2020 – 09.07.2020	2.200,00 €	4 Tage
26.10.2020 – 29.10.2020	2.200,00 €	4 Tage
15.03.2021 – 18.03.2021	2.200,00 €	4 Tage

\* Preis je Teilnehmer, in Euro zzgl. USt.

Anmeldecode: LINUXE-AR

**Coaching**

Unsere Coaching-Angebote bieten den großen Vorteil, dass unsere Experten ihr Wissen und ihre Erfahrungen direkt in Ihren Lösungsprozess einbringen und damit unmittelbar zu Ihrem Projekterfolg beitragen.

Für Ihre Anfrage oder weiterführende Informationen stehen wir Ihnen gern zur Verfügung.