Embedded Linux Driver Development

Online Live

Workshop

Applicable Technologies	Requirements	Contact
Processor based embedded Linux systems like AMD Zynq ™ and others	Knowledge of working with the Linux-Shell, use Makefiles, create and change C-programs. Basic knowledge of processor and FPGA hardware.	Michael Schwarz P. +49 7664 91313-15 E. info@plc2.de

Fee (net per person)	Inclusive	Duration
OL € 1,900	Training material	3 days
WO € 2,300	Plus beverages during breaks Lunch	3 days

Workshop

Today, semiconductors are used in almost every device. The multitude of applications in connectivity and multimedia applications in more and more complex devices makes the development of custom device drivers with the Linux kernel a severe task.

This course sets the stage for shortened development time by equipping participants with the knowledge to design their own device driver using all major kernel interfaces and structural elements. Through hands-on learning with more than 50 % practical exercises, participants gain confidence in using the kernel components needed for a particular project, as well as a solid understanding of the overall framework.

The main focus of this course is the character drivers, platform drivers, and the sysfs interface. Additionally, the fundamentals of the Linux DMA API are discussed.

The workshop is designed for engineers to gain the knowledge and skills about the components of the Linux kernel using such for successful device driver development.

Due to accompanying exercises, the course offers in-depth and practice-oriented training. Attendees of the online live course will do the practical exercises in the afternoon on their own.

Agenda

01. **Linux basics for driver development**
    - Layers of a Linux system
    - Virtual, logical and physical addressing
    - Minimal kernel module

02. **Character drivers**
    - Device numbers, files and numbers structure
    - Dynamic memory allocation

03. **Platform drivers**
    - Device tree
    - I/O hardware access and managing clocks
    - Interrupts

04. **Timing and working**
    - Linux timing mechanisms and delay
    - Kernel threads, softirqs, tasklets and work queues
    - Jiffies, high resolution timers

05. **Synchronisation**
    - Mutex and semaphores, Mutexes
    - Spinlocks, Atomics, Completions and wait queues

06. **Controlling a driver**
    - Ioctl, sysfs
    - Sysfs with binary attributes

EMDALO TECHNOLOGIES

AMD Authorized Training Provider