FEABHAS

Developing Linux Device Drivers

Course category	Embedded Linux
Training area	Operating Systems
Course code	EL-504
Duration	5 days
Additional information	For price: Contact Feabhas (available for on-site delivery only)

Implementing Linux on custom hardware will, in most cases, require you to write device drivers.

This course will show you how to create Linux Device Drivers and that work with a recent version of the Linux kernel that are able to handle hardware events and present a standard interface to applications.

This course presents a detailed view of Linux device drivers with an emphasis on topics specific to embedded environments: cross compilation; remote debugging and direct hardware manipulation. It uses a combination of theory and practice, using a development board with an ARM core.

No prior knowledge of Linux device drivers is assumed, making it ideal for engineers porting from code from an RTOS to Linux.

Course objectives:

- Demonstrate how to write drivers for custom hardware
- Provide insight into porting drivers from an RTOS to Linux, e.g. the separation between application and kernel code
- Describe the development tools needed, including debug strategies
- Examine the way drivers can affect real-time behaviour and best practice to avoid scheduling latencies

Delegates will learn:

- How to write kernel modules
- How to create robust drivers using mutexes and spinlocks to serialise access to shared data
- How to debug kernel code running on a remote embedded target
- Working with GPIO
- How to handle interrupts, including deferred processing using tasklets and work queues
- How to access hardware resources
- The details of memory management and memory mapping techniques
- An introduction to writing a USB driver

Pre-requisites:

- Good ‘C’ programming skills
- General knowledge of an RTOS or embedded operating systems
- Knowledge of Linux or Unix is essential
- Some knowledge of Linux user space is an advantage
- Able to use a command line interface

Who should attend:

Software engineers who are developing applications for embedded or real-time Linux.

Engineers wishing to assess the suitability of Linux for their next application.

Duration:

Five days.

Course material:

- Student workbook

Course workshop

During the lab sessions, students will write several fully-function device drivers, including a FIFO, a RAM disk and a loop-back network interface. All exercises are developed and cross-compiled on a PC running Linux.

The target platform will be the BeagleBone Black, which uses an ARM Cortex-A8. This will help delegates understand the issues encountered when writing for embedded platforms.

Writing Kernel Modules

- Structure of a kernel module
- Compiling and loading modules

Introduction to character device drivers

- Major and minor numbers
- Basic operations – open, read, write and release
- Example driver based on a fifo

Debugging Kernel code and device drivers

- Kernel oops messages
- Debugging with gdb and kgdb

The Linux driver model

- sysfs and the /sys directory
- Adding device classes and class attributes

Task synchronisation

- Putting tasks to sleep using wait queues
- Re-entrancy issues
- Mutexes, semaphores and spinlocks

Device Tree

- An introduction to device tree and it's usage
- Creating an example platform driver to bridge the Device Tree – Kernel divide

Input and output

- Interfacing with the real world
- Accessing memory and i/o mapped resources

Time

- Delays and sleeps
- Using kernel timers.

Interrupts

- Installing an interrupt handler; interrupt context and process context
- Deferred processing using a bottom half or tasklet.

Memory management

- Allocating memory by pages and bytes
- Slab caches
- Techniques to map device memory directly into user space using mmap
- Getting direct access to user buffers

Block Device Drivers

- Anatomy of a block device: example RAM disk driver

USB Device Drivers

- How USB devices work with the kernel and an introduction to Linux USB device model.

Network Device Drivers

- Anatomy of a network device: example loop-back interface

Board Support Packages

- Customising the Linux configuration menus

Feabhas Ltd - UK Office 15-17 Lotmead Business Park, Wanborough, Swindon, SN4 0UY, UK •

+44 (0) 1793 792909 • info@feabhas.com • www.feabhas.com