Linux Device Drivers Training

3-day session

Overview	Understanding the responsibilities and structure of a Linux device. Understanding the development and operating environment for a LDD. Understanding Character drivers, Block drivers and Network drivers. Understanding Debug and Deployment of device drivers. Practical labs.
Duration	Three days - 24 hours (8hours a day) 50% of lecture, 50% of practical labs
Trainer	http://www.linkedin.com/in/pravinkumarsinha
Audience	Professional device driver developers for Linux
Prerequisite	**Knowledge of C programming** In particular, participants must be familiar with creating and dealing with complex data types and structures, with pointers to such symbols, as well as with function pointers. Advance C training agenda is available at http://www.minhinc.com/training/c/advance-c-slides.php can be dowloaded from http://www.minhinc.com/training/advance-c-slides.pdf **Knowledge of system programing Concepts** In particular, participants must be familiar with various aspects of system programming, i.e. File System, Process management, Virtual memory management, IPC, Signals etc. Linux internals training agenda is available at http://www.minhinc.com/training/li/advance-li-slides.php
Setup	Raspberry pi 3, Ubuntu 16/17 LTS

Description

This course gives developers the knowledge of design, write and debug linux device drivers. All examples are written and executed on Raspberry Pi3 hardware with Raspibian (flavour of Debian) OS installed. All cross compilation will be done on Ubuntu 16/17 LTS.

Lecture

Lecture session will be course content presentation through the trainer.
Any source code example related to the topic will be demonstrated, it would include executing the binaries. Complete lecture material can be downloaded from http://www.minhinc.com/training/advance-ldd-slides.pdf

Labs

Labs session would be completely hands on session where each example (with example data and execution instruction) would be provided to the students. Students can verify their results with the results provided in the material.

Day 1 Morning

Lecture - Introduction to Linux Kernel

- Kernel Mode vs. User Mode
- Kernel features
- Kernel user interface (/proc and /sys)
- User space drivers
- Kernel versions and
- CONFIG_MODVERSIONS
- Kernel data types

Lecture - Building and Running Modules

- Include files
- Retrieving kernel sources
- Build and Install (or recompile) a kernel)
- Getting and Installing the kernel sources.
- Configuring and compiling the Replacement kernel.
- Safely Booting the freshly compiled
- Kernel with Grub.

Day 1 Afternoon

Lab

- Bootloader Architecture
  - Bootloader compilation and downloading on Target board.
  - U-Boot Commands
  - Bootloader commands and usage,
  - Bootloader code customization, adding new Ethernet drivers to U-Boot
- Linux Kernel Architecture
  - Linux Source code browsing
  - Configuring and compiling Linux Kernel
  - Build Root file system with busybox

Day 2 Morning

Lecture - Linux Kernel modules

- Various kernel modules
- The format of device driver module
- Loading and unloading device driver module
- The Hello world module
- Compilation and Loading
- The kernel symbol table
- Module parameters
- Module dependencies

Lecture - Character Drivers

- Character driver module API
- Hardware Resources needed by the Driver.
- Major and Minor numbers.
- Char device registration.
- Scull introduction.
- Executing character device driver.

Day 2 Afternoon

Lab

- Module
  - Write a simple module to toggle GPIO with timer
  - Link it statically and dynamically
- Character special device
  - Register a character device type
  - Write a driver to toggle GPIO with every write operation
  - Read the status of GPIO with read operation

Day 3 Morning

Lecture - Concurrency and Race condition

- Concurrency and its management
- Semaphores and Mutexes.
- Completions
- Spinlocks
- Locking traps
- Atomic operations
- Typical locking issues
- Using the lock validator to identify the resource of locking problem.

Lecture - Advanced char driver operation

- Probing for and allocating the I/O Address.
- Interrupts.
- ioctl
- Blocking I/O, poll and select
- Access control on a Device File
- PCI device drivers
- The PCI bus
- Read/Write Functionality.
- Asynchronous operator.

Day 3 Afternoon

Lab

- Interrupt
  - Setup ISR for interrupt and trigger it through GPIO

Day 4 Morning

Lecture - Memory Management

- Supporting mmap()
- Zero copy user space access for I/O
- Atomic kmaps
- Performing Direct I/O, DMA
- Quick access to high - memory via kmap atomic
- Allocating by pages, vmalloc

Lecture - Block Drivers

- RAM disk drivers
- Registration, Entry points
- The Block Device operation.
- The request_for
- Remaining functions

Day 4 Afternoon

Lab

- Write a simple block driver

Day 5 Morning

Lecture - Network Drivers	Lecture - Debugging Kernel Code
• Comparison to Block and Character Drivers. • Exploring the skelton Driver - stage 1 • Snull design • Connecting to the kernel • Opening and Closing • Packet Transmission • Packet Reception • Exploring the skelton Driver - stage 2 • The Interrupt Handler • Receive Interrupt Mitigation • Changes in Link state • The virtual Network Interface	• Using printk • Oops and Ksymoops • The Magic Sys Rq Key • Using strace • Remote debugging via the Serial Port • Debugging by query & watching. • Debugging system faults.

Day 5 Afternoon

Lab
• Write a dummy network driver

Linux Device Driver Essentials

Linux Device Driver Essenstials- Training Course

Minh, Inc.

DISCLAIMER

Text of this document is written in Bembo Std Otf(13 pt) font.

Code parts are written in Consolas (10 pts) font.

This training material is provided through **Minh, Inc.**, B'lore, India
Pdf version of this document is available at http://www.minhinc.com/training/advance-ldd-slides.pdf
For suggestion(s) or complaint(s) write to us at sales@minhinc.com

Document modified on Sep-30-2019

Document contains 15 pages.