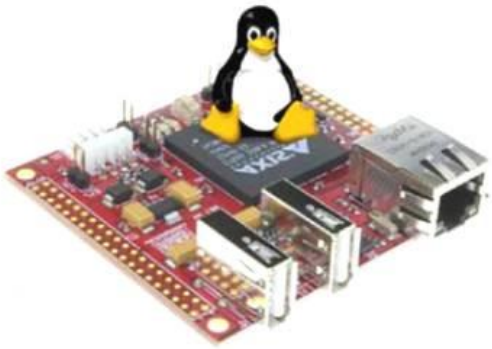

Embedded Linux Platform Developer

Course description

Advanced training program on Embedded Linux platform development with comprehensive coverage on target board bring up, Embedded Linux porting, Linux kernel BSP and Embedded Linux Device Driver Development

Course Highlights

- Detailed analysis of modern embedded hardware architectures and board bring up strategies
- In depth understanding of open source boot loader “uboot”
- Hands on sessions on uboot customization for new targets
- Hands on sessions on Embedded Linux porting strategies
- Analysis and detailed walkthrough of Embedded Linux board support code
- In depth coverage on embedded Linux driver stacks and driver implementations (I2C, SPI, GPIO)
- Analysis and walkthrough of Embedded Linux video and audio driver stacks
- Hands on sessions on multi-media application porting
- Hands on sessions on various hardware targets including Mini 2440, Omap, Zed etc.

VEDA Solutions
inspiring linux learning

301, Prashanthiram Towers, Saradhi Studio Lane, Ameerpet, Hyderabad
Ph:040-66100265 Email: info@techveda.org www.techveda.org

Who should attend

Developers looking to master or expecting assignments/projects in the following core areas:

- Embedded Linux Porting
- Linux BSP development
- Uboot Customization
- Embedded Linux Driver Development
- Android System Development

OR

- GNU C programmers looking to master Embedded Linux, BSP and Embedded Device Driver Development
- Experienced developers looking for a refresher course on Linux BSP and Embedded Device Driver Development.

Course Prerequisites

- Good programming skills using GNU C
- Application programming experience on Linux platform
- Linux kernel programming basics

OR

- Participants who have undergone "Linux Kernel and Driver Developer" course from Veda Solutions

Detailed Course Contents

Part 1: Embedded Hardware

1. Understanding Embedded Systems ● Embedded systems landscape ● Attributes of Embedded Systems ● Trends in Embedded technology ● Embedded Platform Architecture ● Embedded Processor Architecture ● Microcontroller, microprocessors and SOC **2. Understanding Embedded Boot sequence** ● Boot technology considerations ● Role of Bootstrap Processor ● Boot components ● Hardware Power sequences ● RESET vectors ● CPU initialization ● Device Initialization ● Boot device selection	**3. Understanding ARM Architecture** ● Introduction to ARM architecture ● Processor modes ● ARM Registers ● ARM Endian Support ● ARM Instruction Set ● Addressing modes ● The System Control Processor **4. Embedded Target boards (ARM Soc based)** ● Understanding Target board schematic ● Interpreting Target board manuals/data sheets ● Identifying key components on Target board ● Identifying various Device interfaces ● Understanding Memory maps

Part 2: Platform BSP

1. Embedded Board Support Packages

- Introduction to BSP
- BSP design considerations
- Various stages in Board Support Code

2. Working with components of BSP

- Introduction to BootROM
- BootROM design
- BootROM design considerations
- BootROM code
- BootROM executions trace
- Memory organization
- Introduction to FSBL
- FSBL functionalities
- FSBL design standards
- Getting hands on with FSBL (case study)
- Build & Running FSBL
- FSBL Call Graph
- Introduction to SSBL
- Types of SSBL
- SSBL as primary bootloader

3. Uboot

- Introduction to Uboot
- Key Features
- Supported Platforms & Processors
- Supported Boot modes

4. Uboot source walk-through

- Uboot source tree layout
- Memory organization
- Memory relocation
- Stack setup
- Important Misc.

5. Uboot build and setup

- Build structure
- Building U-Boot for emulator
- Building U-Boot for physical target
- Creating flash partitions
- Environment variables
- Commands
- Uboot call graph

6. Customizing Uboot for new targets

- Uboot source tree in detail
- Porting/upgrading u-boot
- Coding guidelines
- Customizing configuration files
- Target specific modifications
- Customizing/adding drivers
- Customizing/adding new commands
- Uboot scripts
- Hosting Apps on Uboot

Part 3: Embedded Linux & Application Porting

1. Embedded Linux overview

- Linux as an Embedded Operating system
- Linux and open source Ecosystem for Embedded Devices
- Linux kernel facilities for Embedded Devices
- Understanding Linux system Boot Process
- Introduction to Linux system components

2. Cross-compiler tool-chains

- Need for cross tool-chain
- Different tool-chains' build procedures
- Using pre-build cross tool-chain
- Building cross tool-chain using cross tool-NG
- Using Scratch box

3. Building kernel Image

- Kernel Source tree organization
- Introduction to kernel Build system
- Understanding Kconfig and Make scripts
- Kernel configuration for target
- Cross compiling kernel source
- Linker scripts
- Build analysis
- Kernel parameters

4. Upgrading/Porting kernel to specific target

- Prerequisites
- Generic procedure
- Kconfig language and Makefile
- Module by module porting

5. Linux File Systems for Embedded Storage

- Need for flash file systems
- Linux File system support for Flash memory
- Understanding jffs2 file system
- Enabling kernel support for jffs2
- Understanding UBI file system
- Enabling kernel support for UBIFS
- Understanding Cramfs
- Enabling kernel support for Cramfs
- Understanding SquashFS
- Kernel support for SquashFS
- Understanding YAFFS2
- Kernel support for YAFFS2

6. Root File system

- Understanding Linux Init process
- Need of root file system
- Understanding Unix File system hierarchy
- Choosing root file system layout
- Startup scripts
- Choosing system binaries and utilities
- Cross-compiling and hosting Apps
- Populating device nodes
- Need for log daemons
- Setting up log daemons
- Building fs image

7. Deploying & Testing Linux system on target

- Flashing U-boot to target
- Booting Linux kernel from DDR
- Flashing kernel image to target
- Deploying RFS through initrd(ramdisk)
- Deploying RFS through initramfs
- Deploying RFS through NFS
- Flashing root file system to target
- Kernel boot from NOR flash
- Kernel boot from NAND flashes
- Kernel boot from SD/MMC
- Kernel boot from Serial flashes

8. Network services and utilities

- nfs
- Telnet
- Ssh
- Dhcp
- Snmp
- http

9. Graphical interface frameworks

- X.org
- Fltk
- Nano-x
- Gtk
- WxEmbedded
- Qt for Embedded

10. Porting multimedia Apps

- Audio Apps
- Video Apps
- Direct media layer
- DirectFB
- Video for Linux
- Digital video broadcasting

Part 4: Embedded Linux BSP

1. Linux BSP basics

- Linux BSP components
- Platform Devices
- Understanding timers
- Understanding interrupts
- Understanding clocks
- Understanding power management
- Understanding memory map
- Understanding interrupts
- Understanding GPIO

2. Hardware Clocks

- Introduction to clocks
- Clock types and significance
- Linux clock management framework
- Using clock interfaces in drivers

3. Power Management

- Introduction to power management
- Device power management and significance
- Linux power management framework
- Using power API's in drivers

4. Device Tree (FDT)

- Introduction to Device tree
- Device Tree Script (dts)
- Device Tree compiler(dtc)
- Device Tree Blob (dtb)
- Booting Linux with device tree
- Device tree and drivers

Part 5: Embedded Linux Device Drivers

1. Linux device driver model

- Introduction to device driver model
- Design objectives
- Hotplug drivers
- Sysfs filesystem
- Procfs filesystem
- Debugfs filesystem

2. Linux device drivers

- Overview of Linux device drivers
- Categories of Linux device drivers

3. Handling Platform devices

- Linux platform driver stack
- Platform devices Enumeration
- Hands on with platform drivers

4. RTC

- Basic RTC operation
- Linux RTC subsystem
- Implementing RTC driver

5. Watchdog

- Basic watchdog operation
- Linux watchdog subsystem
- Implementing watchdog driver

6. UART controller

- UART introduction
- Linux UART stack
- Implementing UART driver

7. Console Devices

- Introduction to console
- Need for console dev
- Writing Console Driver

8. Handling i2c interface

- Introduction to i2c
- i2c protocol
- Linux i2c driver stack
- i2c adapter driver
- i2c bus manager
- i2c client drivers

9. Handling SPI interface ● Introduction to SPI Bus ● SPI protocols ● Linux SPI subsystem ● SPI drivers **10. Handling GPIO** ● Introduction to GPIO ● Linux GPIO management ● Accessing GPIO interfaces **11. Handling Input devices** ● Introduction to input devices ● Linux Input subsystem ● Implementing input drivers (Keypad, mouse, Touch screen) **12. USB** ● USB protocol introduction ● Linux USB subsystem ● Linux USB OTG subsystem ● Implementing USB driver for Mass storage protocol **13. SD/MMC** ● SD/MMC protocol introduction ● Linux SD/MMC subsystem ● Implementing SD/MMC driver	**14. NAND** ● Introduction to MTD devices ● Linux MTD Subsystem ● NAND device introduction ● Implementing NAND driver **15. Ethernet** ● Network protocols introduction ● Introduction to Ethernet ● Implementing Ethernet driver **16. Audio Devices** ● Basic audio operation ● Audio codec's introduction ● Linux Audio Subsystem ● Implementing audio drivers **17. Display/LCD** ● Basic display operation ● Linux Display Subsystem ● Implementing display drivers **18. Frame buffer and Video Devices** ● Introduction to Linux Video Subsystem ● Analysis of Frame Buffer Driver

Part 6: Debugging Tools

1. User space tools ● GDB, gdb server ● Valgrind **2. Kernel space tools** ● Printk ● Kernel OOPS ● KDB ● KGDB	**3. Boot time measurement tools** ● Grabserial ● Bootchart

Part 6: Open Source Development

1. Open Source Contribution ● Importance/Benefits of open source contributions ● Open Source project management Practices ● Development tools required **2. Source control tools (GIT)** ● Overview of GIT ● Useful git commands ● Open source projects hosted under Git **3. Using GIT** ● How to clone an existing git tree ● How to create your own working branch in existing git tree ● How to create patches against the reference tree ● Review of few useful commands	**4. Merging GIT trees** ● How to upgrade your source tree in sync with mainline tree version ● How GIT tree merging works ● How to merge the mainline tree changes into your internal tree ● How to merge BSP into mainline **5. Contributing patches** ● How to send your own patches to mainline GIT tree ● Precautions, while sending patches to mainline ● Other important open source contribution tips

YEDA Solutions
inspiring linux learning

301, Prashanthiram Towers, Saradhi Studio Lane, Ameerpet, Hyderabad
Ph:040-66100265 Email: info@techveda.org www.techveda.org