Real Time and Embedded Linux Software Development

Course Description

The GNU / Linux operating system is the operating system of choice for many embedded and real time developers: the main reasons being that the source code is free, there are no runtime royalties, and it is a robust reliable operating system with excellent networking support. This hands-on course focuses on real time and embedded Linux with real time and embedded aspects of kernel programming. Lab work using an embedded device is an integral part of the course.

Training Duration: 5 days

Who Should Attend? – Embedded and RT programmers developing devices using the Linux kernel and driver developers for internal or external peripherals

Prerequisites:

Usage of Linux's command line, text editing (any text editor), C programming, make, use of Linux C compiler, Linux system calls. Basic knowledge of device drivers and kernel modules is a pro.

Course Outline:

1. **Introduction (day 1)**
   - Linux overview
   - Real time and embedded
   - The kernel and its role
   - Linux supported architectures

2. **Build Root (Day 1)**
   - Project overview
   - Getting buildroot
   - Quick start:
   - Configuration interfaces
   - using a predefined configuration

3. **Cross tool chains (Day 1)**
   - The need for cross tool chains.
   - Tools naming convention.
   - Getting and installing a tools chain.
   - Cross building software
   - Cross debugging
   - uClibc

4. **Configuring and Building the Linux Kernel (Day 2)**
   - Getting the sources
   - The structure of source tree
   - Configuring and building the kernel
   - Compiling the kernel
   - Kernel modules:
     - Cross compiling modules
     - Integrating modules into the kernel source tree
   - Configuring buildroot:
     - Configuring the kernel in buildroot

5. **Customizing Buildroot (Day 2)**
   - Integrating Additional packages into buildroot:
     - dl
     - packages
     - • config.in
   - Rootfs overlays

6. **Device Trees (Day 2 + 3)**
   - Working without device trees
   - What is a device tree
   - DTS and DTB
   - Device tree integration into driver code.
   - The syntax of DTS files

7. **Linux Boot Sequence (Day 3)**
   - Embedded Linux boot process
   - Kernel boot parameters
   - Bootloaders, U-Boot
   - Buildroot configuring system components:
     - init, busy box, U-Boot
   - root-fs:
     - initrd & initramfs
     - overlayfs

8. **Net-Booting and The Network File System (NFS) (Day 3)**
   - How does NFS aid the embedded development process
   - Preparing NFS
   - Mounting an NFS volume
   - NFS daemons
   - Exports file
   - root-fs over NFS
   - tftp
   - DHCP

9. **User-mode Programming (Day 4)**
   - librt overview.
   - Scheduling policies and priorities.
   - CPU affinity.
   - Memory
   - RT signals:
     - explanation
     - comparison with standard signals
   - Asynchronous I/O

10. **Linux and Real Time (Day 5)**
    - RTOS memory issues and Linux.
    - Linux hardware interaction
    - Latency (kernel, interrupt, scheduler)
    - Kernel preemption

- Linux hard real time extensions
- Applying the RT patch
- Threaded IRQ's
- Voluntarily giving up CPU - cond_resched
- Controlling kernel preemption:
    - preempt_disable
    - preempt_enable
    - preempt_count
- spinlocks and raw spinlocks.
- Priority inheritance
- Priority inversion
- Don't do's

11. Introduction to The Yocto Project (Day 5)

- Project overview
- The Yocto project development environment
- Setting up a Yocto project:
- Supported build hosts
- Build host packages
- Getting Yocto
- Example - Building an image and testing it on an emulator
- Development models:
    - System development
    - Application development
- Image development
- Receipes