Building an Internet Radio on the Freescale i.MX31PDK with LinuxLink 3.0

Webinar Series

Session 1

Assembling a small footprint Linux platform for the i.MX31PDK

We will start our webinar in few minutes.
Thank you for your patience

Series Overview

- **Session 1 – today**

  Assembling a small footprint Linux platform for the i.MX31PDK

- **Session 2 – March 16, 2009 11.30am EST**

  Building a custom internet radio GUI on GTK+ with Glade for the i.MX31PDK

- **Session 3 – March 30, 2009 11.30am EST**

  Adding audio codecs and a wireless technologies i.e. Bluetooth subsystem on the i.MX31PDK

- **Session 4 – April 13, 2009 11.30am EST**

  System debugging and testing with the i.MX31PDK

Agenda – Session 1

- Introduction
- Assemble a small footprint Linux platform with the Factory Web Wizard
- Install the image and Desktop Factory on a Linux host
- Footprint optimization techniques
- Using Desktop Factory we will do the following:
  - Verify if any updates are available
  - Modify the kernel for fast booting
  - Root filesystem adjustments
- Deploy the images on the i.MX31PDK board

An Introduction to Timesys

The Company

- Founded in 1996, Carnegie Mellon
- Headquarters in Pittsburgh, PA

Embedded Linux from a Trusted Source
LinuxLink by timesys®

The History

- First real-time Linux distribution
- Open-sourced in 2005
- Launched LinuxLink

LinuxLink — A framework for Linux development

- Great initial Linux platform for a given development kit
- Suite of scriptable tools to configure, customize, build, and test
- Access to man-years of embedded Linux expertise & best practices
- Moderately priced

Open source friendly — no monkey business

Why Pay for Linux — Isn't it Free?

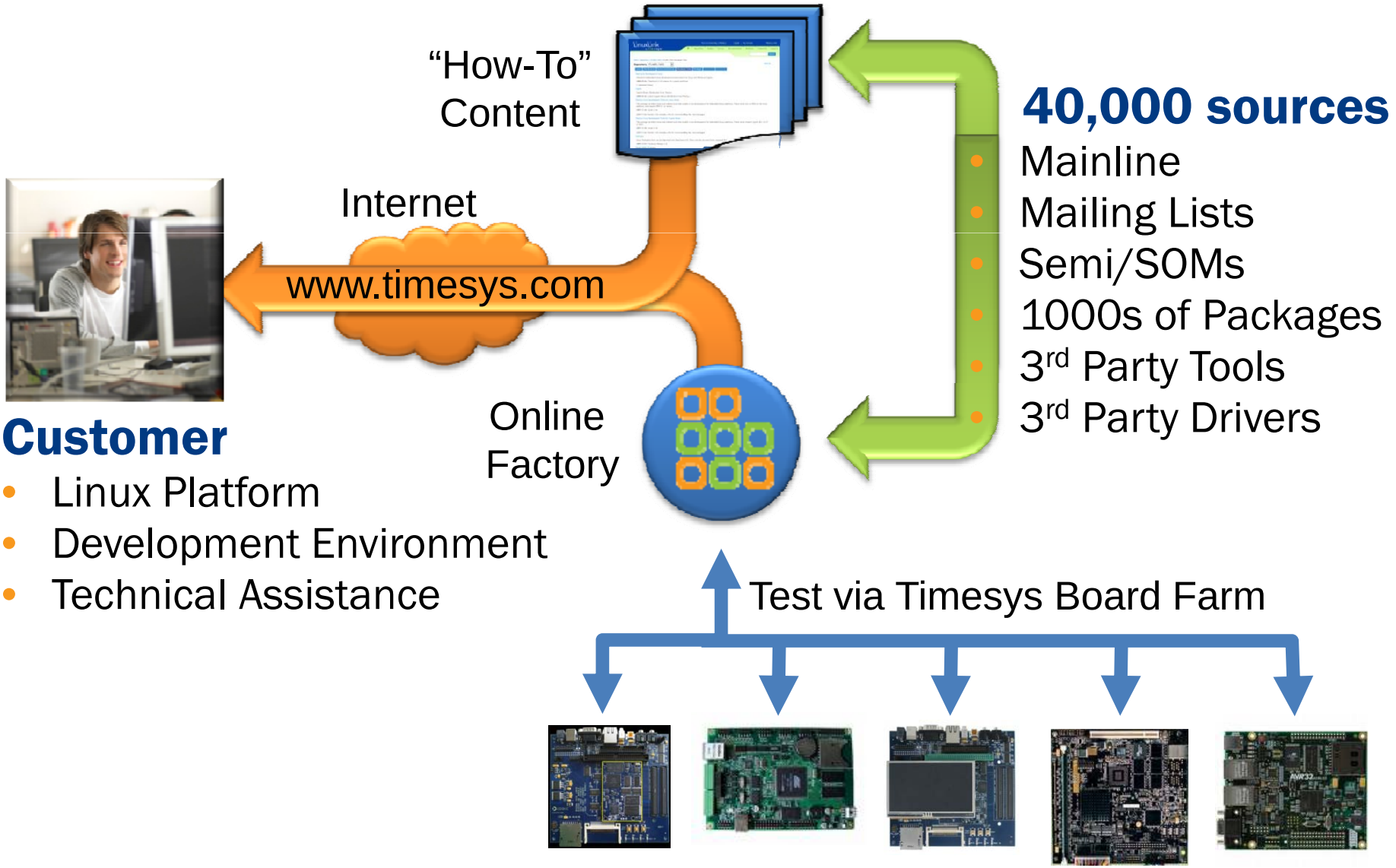
- **Assembling a Linux platform can be very complex**
  - The code is "free", but…
  - Achieving a consistent and repeatable build can be challenging
- **Difficult to keep pace**
  - Over 40,000 independent sources on the Web
  - Maintained by thousands of developers
- **Difficult to pick the right combinations**
  - Hidden dependencies, abandoned projects
  - Numerous revision conflicts

- **Difficult to find tools that work**
  - Many open source tools are available
  - Difficult to assemble the associated patches and libraries
- **Limited-to-no support**

LinuxLink Reduces the Complexity of Open Source Linux Development via a Unique Framework

- **Quickly build an initial platform via the Web**
  - Select a Timesys developed starting point or build your own with the Online Factory
- **Customize via a properly installed, configured and tested desktop environment**
  - Patch/configure/build with Desktop Factory
  - Debug/profile/trace/tune with Toolbox
  - Scriptable
- **Obtain help with common development tasks**
  - Technical Assistance, “How-to” documentation
- **Maintain alignment with the community**
  - Obtain updates/alerts/advice that are relevant to your embedded Linux platform

How Does LinuxLink Work?

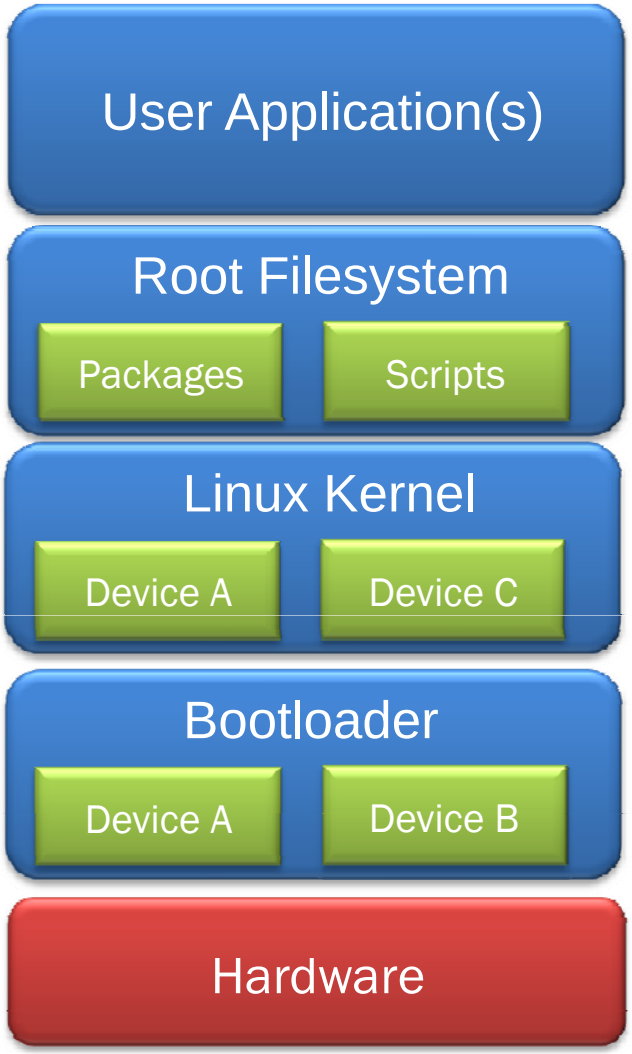


Customer

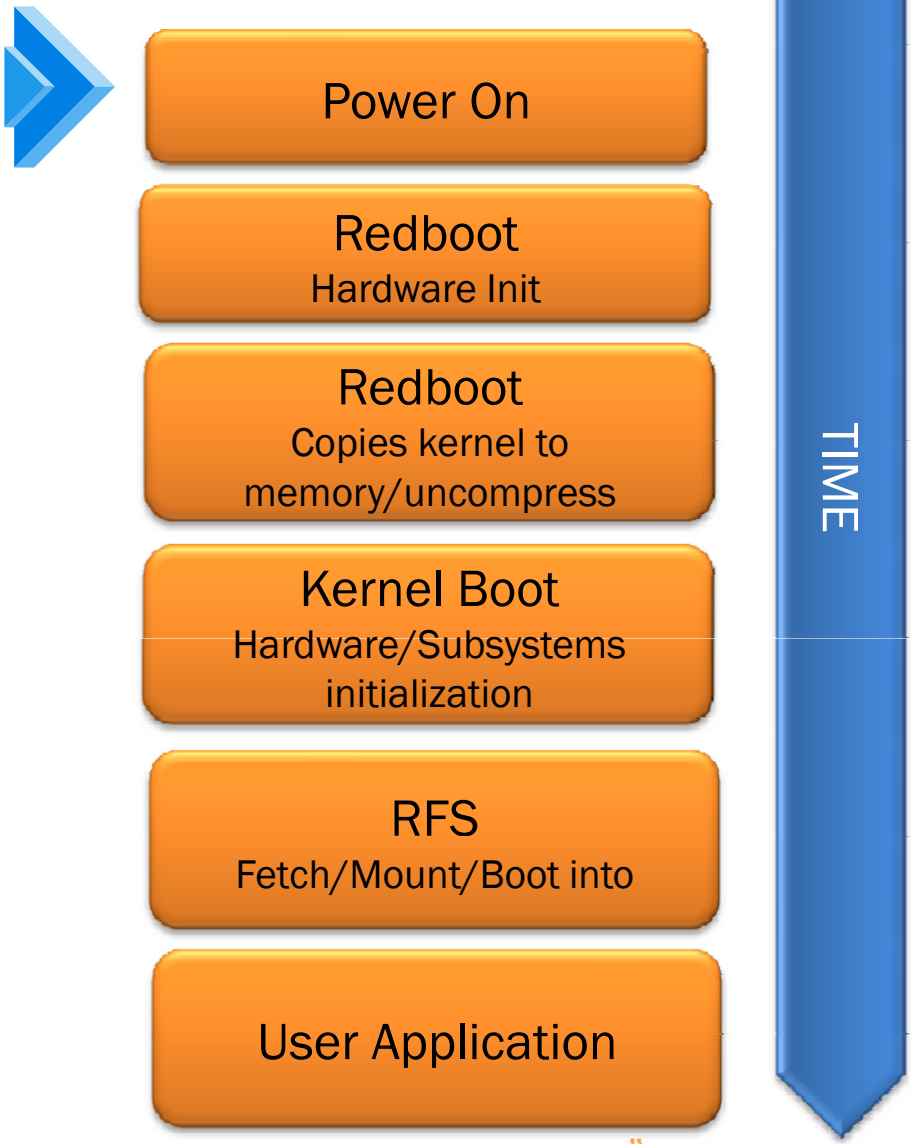
- Linux Platform
- Development Environment
- Technical Assistance

Boot Process in short

Typical Linux System Structure



Boot Process Sequence



Footprint optimization techniques

- **Hardware**
  - Component selection matters
    - NOR flash for XIP
    - Clock Speed
- **Bootloader**
  - Initialize only the needed hardware and pass the control quickly to the Linux kernel
  - Enable only features that are needed (load, boot)
  - Remove hardware probing
  - Typically only few functions used in production system
- **Linux Kernel**
  - Focal point of our today's session
  - Many options available based on requirements
  - Number of options are easy to implement
- **Root filesystem and applications**
  - Minimize the initialization process

Making Linux Kernel small and fast

- **Use uncompressed kernel**
  - Uncompressing takes time
- **Remove unused kernel options**
  - Not used networking i.e. IPV6, multiple file systems
  - Debug features and symbols (for final deployment)
- **Build device drivers as Loadable Kernel Modules**
  - Keep the features needed at boot time built into the kernel
  - Remaining drivers built as LKMs will make kernel smaller
- **Consider various approaches for your RFS deployment**
  - JFFS2 with appended journal summary (skip flash scan)
  - CRAMFS, UBIFS
- **Suppress the console output**
  - Use “quiet” with your kernel command line
  - Remove any possible verbosity from the kernel (i.e. printk)

Making Linux Kernel small and fast

- **Linux kernel options we will look at today**

Kernel Option	Comment
CONFIG_EMBEDDED	Disables or tweaks a number of kernel options and settings.
CONFIG_IKCONFIG	Saves complete kernel configuration in the kernel
CONFIG_KALLSYMS	Prints symbolic crash information and backtraces
CONFIG_BUG	Disables BUG and WARN functions
CONFIG_HOTPLUG	Can be disabled if no external devices will be attached and if you use static device files
CONFIG_DNOTIFY	File change notification to user space
CONFIG_EXT2	Disable if using jffs2 file system
CONFIG_PRINTK	Makes kernel silent when disabled
CONFIG_PRINTK_TIME	A way to track where time is spent at boot time
CONFIG_CC_OPTIMIZE_FOR_SIZE	Will select –Os instead of –O2 resulting in a smaller kernel

Deployment

- **We'll use TFTP service to transfer images to the target**
  - **Runs typically as xinetd process**
  - **Disabled when newly installed. Modify /etc/xinetd.d/tftpd**

- **Flashing procedure**

```
RedBoot> fis erase –f 0x100000 –l 0xff00000
RedBoot> fis init
RedBoot> fconfig
>>fis load kernel; exec –c “noinitrd console=ttymxc0,115200
root=/dev/mtdblock2 rw rootfstype=jffs2”

RedBoot> load –r –b 0x100000 zImage-mx31
RedBoot> fis create –f 0x100000 kernel
RedBoot> load –r –b 0x100000 rootfs.jffs2
RedBoot> fis create –f 0x600000 root

RedBoot> reset
```

What we have accomplished

- **Built a starting point with Online Factory**
  - Experiment on day one with a pre-built starting point
- **Modified Linux kernel using desktop tools**
  - Optimized for footprint
  - Optimized for fast boot
- **Altered root filesystem**
  - Removed unneeded startup scripts
- **Deployed the system on target's NAND flash**
  - Transferred and flashed images via TFTP
  - Configured bootloader for autoboot

Next Time

- **Session #2 is scheduled for Monday, March 16th at 11.30am EST**

Subject: Building a custom internet radio GUI on GTK+ with Glade for the i.MX31PDK

- Install Glade tools on a Linux host
- Design a simple Internet Radio application w/ GUI
- Development and initial testing using host environment
- Add needed code to launch a navigation application i.e. web browser
- Patch/modify the navigation package to launch a playback application
- more fun to come…

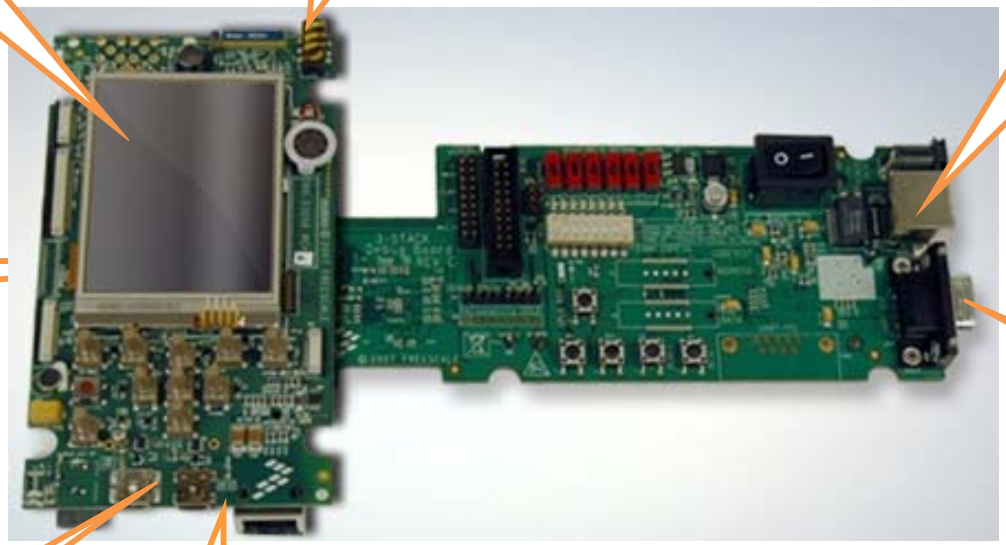
What we will use on i.MX31PDK

- **Graphics**
- **Custom Application**
- **Touchscreen**
- **Applications**
  - Audio player
  - Calibration
  - Web browser
  - More ...

- **Audio**

-Alsa Mixer
- Audio Streaming
- Support for codecs

- **Ethernet**
  - Secure Connection
  - Transfer (FTP/SCP)
  - Console (Telnet/SSH)
  - Network Root FileSystem



NAND Flash
- Boot from
- Additional storage

- **Serial port communication**

- **USB**
  - FileSystem (CF, USB stick)
  - Extentions

SD/MMC Card
- Additional storage

How Do I Learn More?

- **Go to www.timesys.com to learn more**
  - Webinars
  - LinuxLink Radio Podcasts
  - Request a Free Factory Test Drive
- **Schedule a more detailed introduction/training session**
  - WebEx
  - Face-to-face