ReliaBle EmBedded Systems

Embedded Linux - Von Systemarchitektur zu Echtzeit

Robert Berger - Reliable Embedded Systems e.U.
Consulting Training Engineering
https://www.ReliableEmbeddedSystems.com
robert.berger@ReliableEmbeddedSystems.com

Kurs Ziele

Der Zweck dieses Kurses ist es, einen grundlegenden Überblick über Embedded Linux, Interprozesskommunikation sowie Echtzeit mit Linux zu geben.

- /> Wie passen die (Puzzle) Teile zusammen?
- /> Welche Komponenten werden zum Aufbau eines Embedded Linux-Systems benötigt?
- /> Woher bekommt man sie und wie werden sie konfiguriert/gebaut/installiert?
- /> Woher kann man Hilfe bekommen?
- /> Wie sieht es mit Lizenzen aus?
- /> Echtzeit

Praktische Übungen vermitteln Ihnen die notwendige praktische Erfahrung um, nach einem erfolgreichen Abschluss dieses Kurses, Ihr eigenes Embedded Linux System entwickeln zu können.

Beschreibung

Dieses fünftägige Training kombiniert Theorie mit Hands-on Übungen um einige grundlegende Konzepte von Embedded Linux vorzustellen.
Hands-On-Sessions werden auf dem Host mit einem Yocto Project SDK und auf einer bestimmten Zielhardware (z.B. i.mx6 [1]) ausgeführt. Der gesamte Workshop soll Sie schnell auf den neuesten Stand bringen. Die Konzepte und Befehle, die notwendig sind, um Linux effektiv zu nutzen, werden durch eine Kombination von Theorie und Praxis beschrieben. Erfinden Sie das Rad nicht neu, sondern lernen Sie von einem erfahrenen Trainer und nehmen Sie das neu erworbene Wissen über Embedded Linux und die Fähigkeit, es effektiv in Ihr eigenes Embedded-Entwicklungsprojekt zu integrieren, mit nach Hause.

Erforderliche Vorkenntnisse

- /> Grundlegende Vertrautheit mit der Verwendung eines Linux-Systems (z.B. Ubuntu) als Endbenutzer im user space
- /> Grundlegende Vertrautheit mit einer Befehlszeilen-Shell
- /> Vertrautheit mit Konzepten und Programmierung in C für eingebettete Systeme
- /> Fähigkeit, Software mit C-Syntax zu entwickeln
- /> Grundlegende Vertrautheit mit der Verwendung von Embedded-Prozessor-Entwicklungsboards
- /> …und keine Angst, noch mehr von dem oben genannten zu lernen

Wer sollte teilnehmen?

Manager, Projektmanager, Software-, Hardware-, Entwicklungs-, Systemingenieure, Tester, Administratoren, Techniker und andere an der Technologie interessierte Personen, die möglichst schnell verstehen wollen, wie Embedded Linux funktioniert. Sie müssen Linux verwenden oder Sie haben den Luxus zu entscheiden, ob es sinnvoll ist, es zu benutzen oder nicht. Vielleicht haben Sie bereits versucht, Embedded Linux zu verwenden, sind sich aber nicht sicher, ob Sie alles richtig gemacht haben. Sie verwenden derzeit ein anderes Betriebssystem und möchten herausfinden, ob Linux vielleicht besser und/oder billiger ist. …

Welche Vortragsmethoden dürfen wir Ihnen anbieten?

Alle Schulungsunterlagen sind in englischer Sprache, der Vortrag kann entweder auf Englisch oder Deutsch erfolgen, wie Sie es wünschen, weltweit.

Während des Trainings bieten wir folgendes an:

	öffentlich (für 1 Teilnehmer) **vor Ort** (ab 2 Teilnehmer) mit Schulungsleiter	**online** mit Schulungsleiter	**vor Ort/online** Kombination mit Schulungsleiter
Arbeitsbuch	✓ auf Englisch/pdf gedruckt auf Anfrage	✓ auf Englisch/pdf gedruckt auf Anfrage	✓
Host Rechner	✓ Laptop	✓ Vm/Laptop gehosted	✓
Zielhardware (Target Board)	✓	✓ gehosted	✓
Fern-Zugriff auf Host Rechner und Board		✓	✓
(web) Screen-sharing Bildschirm Freigabe		✓	✓
(web) Audio/Video		✓	✓
(web) Whiteboard		✓	✓
(web) Chat		✓	✓
(web) Fragen&Antworten		✓	✓
(phone) audio-bridge Telefonkonferenz		✓	✓

Wenn Sie genaueres zum Thema "Vortragsmethoden für Schulungen" wissen möchten können Sie hier [2] nachlesen und/oder fragen Sie hier [3]. Schulungsteilnehmer behalten üblicherweise die Arbeitsbücher. Normalerweise setzen wir zwei Schulungsteilnehmer an einen Arbeitsplatz, damit sie sich gegenseitig helfen können.

Kursinhalt

Tag 1

Einführung

/> Geschichte von Unix/Linux | Lizenzen | Standards | Mit freier Software arbeiten | "Spelunking" | Die Unix Philosophie

Host-Spielplatz

/> Architektur | Shell | Benutzer | `su` | FHS (file hierarchy standard) | Daemon | statische/shared Bibliotheken | Berechtigungen | Zugriffskontrolle | `umask` | harte/weiche Links | Mehr Berechtigungs-Bits | Framework zum Kompilieren | echte/effektive uid/gid

Tag 2

Prozesse und Interprozesskommunikation

/> Linux Architektur | Was ist ein Betriebssystem? | Was ist ein Scheduler? | Linux: Scheduler(s) - Prioritäten - Scheduling Klassen | Prozess - Task - Thread | `errno` | `fork` | Prozessbeendigung | Prozesszustände | Zombies | Sehen wir uns Prozesse an

/> Einfache Interprozesskommunikation

/» Shell-Umleitung | Shelling out | temporäre Dateien

/> Fortgeschrittene Interprozesskommunikation

/» generisch | Pipe | Signal | Unterbrochener Systemaufruf | Nachrichtenwarteschlange (Message Queue) | Semaphor | Mutex | gemeinsam genutzten Speicher (Shared Memory) | Socket | `select` | `poll`

/> Andere Interprozesskommunikations-Mechanismen | Interprozesskommunikations-Techniken die man vermeiden sollte

Besonderheiten von "Embedded"

/> Linux auf dem Desktop im Vergleich zu Linux auf einem Eingebetteten System | Software Release | Portierbarkeit | Bauen (Cross-/Native Toolchains) | Toolchains/Build-Systeme (crosstool-NG | buildroot | BitBake) | C-Bibliotheken | …

Tag 3

Eval Board

/> Einführung | Wie booted Linux auf einem PC und auf dem Eval Board? | Bootsequenz | Partitionen auf der SD Karte | Sehen wir wie es booted

Was man so braucht

/> Host

/» Werkzeugkasten

../ Das Yocto-Projekt | Der YP Autobuilder | Das Yocto SDK installieren

/» Vorbereitung der SD-Karte | Terminal-Emulation

../ `screen` | `byobu` | `minicom`

/» Server

../ tftp | NFS

/» Root file Systeme

/> Zielsystem

/» `git` | u-boot

- ../ Scripting | Schickes Zeug | Runterladen | Konfigurieren | Bauen | Installieren | Variablen und Befehle
- /» Flat device tree (fdt)
  - ../ Was ist ein fdt? | Device Tree Compiler | Beispiele
- /» Linux kernel
  - ../ Runterladen | Konfigurieren - `kconfig` | Bauen - `kbuild` |
- /» fdt
  - ../ Bauen | Installieren
- /» Kernel Module
  - ../ Bauen | Installieren

Tag 4

Root File Systeme

- /> Ein minimales Root-File-System vorbereiten | Erinnern Sie sich and den tftp Server?
- /> ramdisk | mtd | jffs2 | mtdparts | cramfs | roEXT2 | Vergleich von Root-File-Systemen

Anpassungen

- /> Anpassungen: Das System hängt | pseudo `tty` | `getty` | `runlevel` | Statische IP | `mount` nfs | `ssh` Server

User Space Fehlersuche/-behebung

- /> Was ist ein Fehler/Debugging? | Arten von Fehlern | Debugging-Strategie
- /> Einfache Werkzeuge
  - /» `lsof` | `ltrace` | `strace` | `procfs` | `top` | `netstat` | `syslog` | ...
- /> Fortgeschrittene Werkzeuge
  - /» Was ist ein Debugger?
  - /» `gdb`: `gdb` auf Zielhardware | `gdb` Remote-Debugging

JTAG Debugging (optional)

- /> JTAG | JTAG Setup | u-boot vor der Relokation | andere Möglichkeiten

Profiling/Tracing

- /> Profiling im Vergleich zu Tracing
- /> Profiling/Tracing Werkzeuge: `strace` | System-Timer | `gprof` | `gcov` | `perf`
- /> `top` und ähnliche Werkzeuge: `top` | `powertop` | `powerdebug`
- /> `crash` - `gdb` on steroids

Tag 5

Echtzeit

- /> Voraussetzungen
  - /» Kernel im Vergleich zu User-Space | "Program Sections" | Interrupts | Reentrant Code
- /> Echtzeit Einführung
  - /» Was ist Echtzeit? | Was ist harte Echtzeit?
- /> Echtzeit Linux
  - /» Hypervisor | Dual Kernel | Voll präemptiver Kernel
- /> "Grade/Stufen" des Echtzeitverhaltens
  - /» explizite/implizite Präemptions-Punkte | Echtzeit Präemptions Patch | Voll präemptiver Kernel | harte Echtzeiterweiterungen
  - /» (optional) Kernel patchen/konfigurieren/bauen | auf dem Board ausführen
- /> Dual Kernel
  - /» RTLinux | RTAI | Xenomai
    - ../ Xenomai im Vergleich zu RTAI | Adeos | Xenomai Architektur/SW Stack/Skins
    - ../ (optional) Kernel patchen/konfigurieren/bauen | auf dem Board ausführen
- /> Messergebnisse: Nicht Echtzeit Kernel | Voll präemptiver Kernel | Dual Kernel (Xenomai)
- /> Echtzeit Mythen

Ähnliche Kurse

- /> *"Embedded Linux - Von Systemarchitektur zu Echtzeit (5 Tage)"*
  `https://rlbl.me/elisa`
  `https://rlbl.me/elisa-de-pdf`
- /> *"Auffrischungskurs Embedded Linux & Einführung in das Yocto Project (5 Tage)"*
  `https://rlbl.me/intely`
  `https://rlbl.me/intely-r-de-pdf`
- /> *"Embedded Linux Kernel Interna und Gerätetreiber (5 Tage)"*
  `https://rlbl.me/ldd`
  `https://rlbl.me/ldd-de-pdf`
- /> *"Das Yocto Project - Eine ausführliche Übersicht (4 Tage)"*
  `https://rlbl.me/yocto`
  `https://rlbl.me/yocto-r-de-pdf`
- /> *"Einführung in Embedded Linux & Echtzeit, das Yocto Project aus der Vogelperspektive (4 Tage)"*
  `https://rlbl.me/entirety`
  `https://rlbl.me/entirety-de-pdf`
- /> *"Embedded Linux und Anschließen von Hardware (4 Tage) - kommt bald"*
  `https://rlbl.me/elisha`
- /> *"Linux Gerätereiber kurz und bündig (4 Tage) - kommt bald"*
  `https://rlbl.me/coldd`

/> *"Einführung in Embedded Linux in Theorie und Praxis - ein Intensivkurs (3 Tage)"*
https://rlbl.me/elin
https://rlbl.me/elin-de-pdf

/> *"(Embedded) Linux Fehlersuche-/behebung (3 Tage)"*
https://rlbl.me/lindeb
https://rlbl.me/lindeb-r-de-pdf

/> *"FreeRTOS in Theorie und Praxis (3 Tage)"*
https://rlbl.me/freertos
https://rlbl.me/freertos-r-de-pdf

/> *"Das Zephyr Project - ein Überblick (3 Tage) - kommt bald"*
https://rlbl.me/zephyr

/> *"Das Yocto-Projekt für Fortgeschrittene - Lizenzierung (1 Tag)"*
https://rlbl.me/yopalic
https://rlbl.me/yopalic-r-de-pdf

Verweise

[1] *"Zielhardware"*
http://rlbl.me/hw

[2] *"Vortragsmethoden für Schulungen"*
http://rlbl.me/delivery

[3] *"Reliable Embedded Systems: Kontakt"*
http://rlbl.me/contact

Trainer

Robert Berger hat seit 1993 Praxis- und Führungserfahrung in der Industrie bei Design und Entwicklung von Software für "Eingebettete Systeme" mit und ohne harten Echtzeit Anforderungen. Er arbeitet seit dem Beginn des 21. Jahrhunderts mit Linux auch im Desktop/Server Bereich, aber hauptsächlich mit "Eingebetteten Systemen" (Automotive, Industrial Control, Robotics, Telecom, Consumer Electronics,...). Man trifft ihn regelmäßig bei internationalen Events wie z.B. "Embedded World", "Embedded Software Engineering Kongress", "Embedded Systems Conference", "Embedded Linux Conference" und "Yocto Project Summit" als Experten und Vortragenden. Seine Spezialitäten sind vorwiegend Schulungen aber auch Beratungen (auf Deutsch oder Englisch) weltweit und zwar von kleinsten Echtzeit Systemen (FreeRTOS) bis hin zu Systemen mit mehreren Prozessoren/Kernen und Embedded Linux (User-, Kernel-Space, Gerätetreiber, Hardware Interfacing, Debugging, Multi-core, Yocto Projekt,...) mit Schwerpunkt Free- und Open Source Software. Robert ist ein Weltenbummler. Er ist CEO & Embedded Software Evangelist bei Reliable Embedded Systems e.U. (St. Barbara/Österreich) und wenn er keine Kundenbesuche macht ist er bei seiner Familie in Athen/Griechenland.

Vielen Dank für Ihr Interesse!

Für Anfragen schicken Sie bitte eine E-mail an:
training@ReliableEmbeddedSystems.com