Auffrischungskurs Embedded Linux & Einführung in das Yocto Project

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

Kurs Ziele

Nach einem kurzen *"Auffrischungskurs Embedded Linux (2 Tage)"*, möchten wir Ihnen das Wesentliche zur Nutzung des *"Yocto Projekts (3 Tage)"* vermitteln. Nach der *"Auffrischung"* werden wir sehen, wie BSP/Framework-Betreuer das *"YoctoYocto-Projekt"* verwenden würden. Wir werden auch sehen wie Entwickler es verwenden können, die vielleicht nicht einmal wissen wollen, dass sie es verwenden.

Beschreibung

Diese fünftägige Schulung kombiniert Theorie mit praktischen Übungen um Konzepte von *"Embedded Linux"* und des *"Yocto-Projekts"* vorzustellen. Sie wurde entwickelt, um Sie schnell auf den neuesten Stand zu bringen.
Sie beantwortet häufig gestellte Fragen wie:

- /> Was ist Linux?
- /> Woher bekommt man U-Boot/den Kernel? Wie wird so was konfiguriert/gebaut/installiert?
- /> Warum sollte man "Upstream" verwenden?
- /> Wie funktioniert IPC und was sollte man verwenden/vermeiden?
- /> Ist es wirklich notwendig, für jedes Linux Projekt eine andere Version der Toolchain / Bibliotheken / Pakete zu verwenden, und dann auch noch bei jedem Projekt einem anderen Workflow zu folgen?
- /> Können Sie sicherstellen, dass die Entwicklungsumgebung für alle Entwickler/Lieferanten identisch ist und dass man in 10+ Jahren immer noch die gleichen Builds wie heute reproduzieren kann?
- /> Kann das YP dabei helfen herausfinden, unter welchen Softwarelizenzen die von Ihnen verwendeten Pakete lizenziert sind oder bevorzugen Sie stattdessen Bekanntschaft mit einem Copyright-Troll [1] zu machen?
- /> …und vieles mehr

Hands-On-Sessions werden auf dem Host, in einem Docker-Container und auf einer bestimmten Zielhardware (z.B. i.mx6 [2]) ausgeführt. Sie erhalten ein brandneues Hardware-Kit [2] damit Sie in der Lage sind auch nach der Schulung das Gelernte zu wiederholen bzw. mit neuen Dingen zu experimentieren. Nach dem Training können Sie einen auf Ubuntu basierenden Docker-Container sowie sämtliche Beispiele herunterladen um mit dem Kursmaterial in Ihren eigenen Labors zu arbeiten. Bitte beachten Sie, dass die ersten zwei Tage *"Auffrischungskurs Embedded Linux"* davon ausgehen, dass Sie schon (Embedded) Linux Erfahrung mitbringen um dem *"Yocto-Training"* folgen zu können. In den ersten beiden Tagen betrachten wir kurz, wie Embedded Linux funktioniert und konfigurieren/bauen unter anderem den Linux-Kernel. Falls Sie einen Anfängerkurs für Embedded Linux suchen wären *"Embedded Linux - Von Systemarchitektur zu Echtzeit (5 Tage)"* [3] bzw. *"Einführung in Embedded Linux in Theorie und Praxis - ein Intensivkurs (3 Tage)"* [4] empfehlenswert.

Erforderliche Vorkenntnisse

- Vertrautheit mit der Verwendung eines Linux-Systems (z.B. Ubuntu) als Endbenutzer im user space
- Grundlegende Vertrautheit mit einer Befehlszeilen-Shell
- Grundkenntnisse der Programmierung von User/Kernel-Space mit Linux
- Es hilft folgendes zu kennen: *"Embedded Linux - Von Systemarchitektur zu Echtzeit (5 Tage)"* [3] oder *"Einführung in Embedded Linux in Theorie und Praxis - ein Intensivkurs (3 Tage)"* [4] aber selbst wenn nicht, versuchen wir, Sie in den ersten 2 Tagen auf den notwendingen Wissensstand zu bringen.
- Es kann helfen, falls Sie *"Embedded Linux Kernel Interna und Gerätetreiber (5 Tage)"* [5], besucht haben, aber das ist keine Voraussetzung. Es genügt zu wissen, wie man den Linux-Kernel, Kernel-Treiber in/out of tree und den fdt baut, um dem BSP/Kernel Teil des Yocto Trainings folgen zu können. Diese Themen werden wir in den ersten beiden Tagen behandeln.

Wer sollte teilnehmen?

Sie verwenden Linux für Ihre Projekte und haben von dem YP gehört, aber es nicht genauer beleuchtet oder hatten Probleme damit. Außerdem möchten Sie Kenntnisse zu Embedded Linux, IPC und Echtzeit auffrischen. Sie wissen nicht, ob und wie Ihr täglicher Workflow mit dem YP zu realisieren ist und finden das YP im Allgemeinen kompliziert. Der Workshop richtet sich an Software-, Entwicklungs-, Systemingenieure, Tester, Administratoren, Ingenieure und andere an dem YP Interessierte, die bereits grundlegende Kenntnisse von (Embedded) Linux haben.

Kursinhalt

Tag 1

Einführung

- Einführung | Geschichte

Eval Board

- Wie booted Linux auf einem PC und auf dem Eval Board? | Boot Sequenz | Partitionen auf der SD Karte | Sehen wir wie es booted

Was man so braucht

- `git` | u-boot
  - Scripting | Schickes Zeug | Runterladen | Konfigurieren | Bauen | Installieren
- Linux kernel
  - Runterladen | Konfigurieren - `kconfig` | Bauen - `kbuild` | Kernel Module | Installieren
- Flat device tree (fdt)
  - Den fdt untersuchen | Bauen | Installieren

Kernel Module

- /> `init`/`exit` | Lizenzierung | verschmutzes(r) Modul/Kernel | `EXPORT_SYMBOL` | out of tree `.ko` makefile
- /> Modul-Init-Werkzeuge
- /> Modul im Kernel Tree
- /> Parameterübergabe

Character Treiber

- /> Gerätedateien | Gerätetypen | Geräteknoten major/minor | Kernel-Treiber-Schnittstelle
- /> Gerätetreiber: Einführung | Anmeldung | Initialisierung | Öffnen/Schließen
- /> Miscellaneous Character Treiber

User Space Fehlersuche/-behebung

- /> Einfache Tools
  - /» `lsof` | `ltrace` | `strace` | `procfs` | `top` | `netstat` | `syslog` | ...
- /> Fortgeschrittene Tools
  - /» Was ist ein Debugger?
  - /» `gdb`: `gdb` auf Zielhardware | `gdb` Remote-Debugging

Kernel Fehlersuche/-behebung (optional)

- /> Einführung | `kgdb`/`kdb` + `agent-proxy` | JTAG

Tag 2

Prozess und Interprozesskommunikation

- /> Linux-Architektur | Betriebssystem | Scheduler - Prioritäten - Scheduling Klassen | Prozesse - Tasks - Threads | `errno` | `fork` | Prozessbeendigung | Prozesszustände | Zombies | Sehen wir uns einen Prozess an | Einfache Interprozesskommunikation
  - /» Shell-Umleitung | Shelling out | temporäre Dateien
- /> Fortgeschrittene Interprozesskommunikation
  - /» Pipes | Signale | Unterbrochene System Calls | POSIX.4 Nachrichtenwarteschlangen (Message Queues) | Semaphor | Mutex | gemeinsam genutzter Speicher (Shared Memory) | Sockets | `select` | `poll`
- /> Andere Interprozesskommunikations-Mechanismen | Interprozesskommunikations-Techniken die man vermeiden sollte

Echtzeit

- /> Voraussetzungen
  - /» 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
- /> Dual Kernel: Adeos/Xenomai
  - /» Adeos Patch | Xenomai | Kernel patchen/konfigurieren/bauen | auf dem Board ausführen
- /> Echtzeit Mythen

Tag 3

Yocto Einführung

- /> Was ist Yocto? | Was ist das YP? (Features/Herausforderungen) | Einige Werkzeuge unter dem projektübergreifenden YP (Poky | BitBake | OE-Core | Meta Daten) | Warum sollte man das YP verwenden?

So werden Sie Teil unserer Yocto Project-Community (optional)

- /> Erste Schritte | So erreichen Sie uns | Sich beteiligen (Participate) | Beitragen (Contribute) | Soziale Medien | Veranstaltungen

Entwicklungsumgebung

- /> Was wird für einen YP-Build benötigt? | Layer Versionen/Abhängigkeiten | Hostabhängigkeiten | Konfigurationsdateien | Host-/Kernel-Abhängigkeiten

Der YP Workflow

- /> Einleitung
- /> Konfiguration (Benutzer | Meta Daten | Maschine | Regeln für die Distribution)
- /> Eigenschaften (Features) | Versionierung von Rezepten | Layers
- /> Quellen
- /> Quellen fetchen | Patchen | Konfigurieren/Kompilieren/Installieren | `SSTATE` | Pseudo | `recipetool` | Beispiele für Rezepte | `PACKAGECONFIG` | Pakete und deren Inhalt | Output-Analyse/Packaging | Erzeugen von Images | Erzeugen von SDKs | Tasks
- /> Anpassen von Images (Intro | `local.conf` | `IMAGE_FEATURES` | benutzerdefinierte `.bb` files | benutzerdefinierte Paketgruppen)

Tag 4

BitBake

- /> Geschichte
- /> Syntax (Ersetzen des Variablennamens durch den Inhalt [Variable Expansion] | Variablenzuweisung [Variable Assignment] | Vor-/Anhängen [Pre-/Append] | Entfernen [Removal] | Bedingte [Conditional] Syntax - `OVERRIDES`)
- /> BitBake Debugging (Debug Level | Rezepte/Images/Packagegroups finden | BitBake Umgebung/-Tasks/Logging | bauen/spezifischen Task erzwingen | `cleansstate` | `stamp` für ungültig erklären | `devshell` | Abhängigkeits-Explorer | alle BitBake Instanzen töten | grafische Wrapper für BitBake)
- /> Aufräumen (um Speicherplatz zu gewinnen | damit neu gebaut werden muss)

Layer

- /> Intro | `bitbake-layers` tool | dynamische Layer

BSP

- /> Intro | Systementwicklungs-Workflow | BSP Entwicklungsanleitung (`bsp-tool` - auf die aktuelle Poky Version portiert | Erstellung eines BSPs | Patches, die noch nicht Mainline sind)

Kernel

- /> Intro | Systementwicklungs-Workflow | Kernel-Entwicklungsanleitung (`defconfig` | `defconfig` + Konfigurationsfragment | in tree kmod | out of tree kmod | "klassischer" fdt | fdt mit `devicetree.bbclass` | ...)

Tag 5

Software Entwicklungskit

- /> Einführung | Cross-Entwicklungs Toolchain | Sysroot | `BBCLASSEXTEND` | Multilib | Der QEMU-Emulator | Eclipse Yocto Plug-in (veraltet) | User Space Werkzeuge | Installieren von SDKs und Toolchains
  - /» Cross-Toolchains/SDKs
    - ../ Erstellen eines Cross-Toolchain-Installers
    - ../ Verwenden des Standard-SDK (Cross-Toolchain+Makefile/Autotools/Autotools lib + App | Rezepte)
    - ../ Erstellen/Verwenden eines Erweiterbaren (Extensible) SDK

Paketverwaltung

- /> Softwareaktualisierungen (SW Update) im Vergleich zu "Paket Management" | Mit Paketen arbeiten | IPK | Einen package feed erstellen | Installieren eines Pakets mit opkg auf dem Zielsystem

Lizenzierung

- /> Einführung | eine benutzerdefinierte Lizenz hinzufügen | eine "kommerzielle" Lizenz hinzufügen | Firmware Lizenzen | Open-Source-Lizenz Bestimmungen

Devtool

/> Einführung | Ein Rezept hinzufügen/bauen/installieren | Einen Layer erstellen/hinzufügen | Finish | Ein Rezept Ändern/Aktualisieren | Bauen/Ausführen | Ein Image bauen

Remote-/Online-Buchungsoptionen

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	**remote/online öffentlich** 1 Teilnehmer von einer Firma	**remote/online privat** ab 3 Teilnehmer[2] in einer Buchung z. B. von einer Firma
	GTR[1] ab 3 Teilnehmer mit Schulungsleiter	
(ship-it) Arbeitsbuch	auf Englisch/pdf - Download Link, gedruckt auf Anfrage✓ behält der Schulungsteilnehmer✓	
(ship-it) Beispiele	Download Link + am Host installiert✓ behält der Schulungsteilnehmer✓	
(target) Kit [2]	wird dem Teilnehmer geliefert✓ um nach dem Training mit Trainingsmaterial experimentieren zu können✓ behält der Schulungsteilnehmer✓	
(ship-it) Build Umgebung	Download-/Installationsanweisungen + auf dem Host installiert✓ behält der Schulungsteilnehmer✓	
(Host/Target) Fern-Zugriff	✓	
(host) Host Maschine	Laptop/VM/Container - gehostet✓	
(target) Board	gehosted✓	
(target) Board Power over IP	gehosted✓	
(target) Board Zugriff auf Konsole	gehosted✓	
(target) Board booted über tftp/nfs	gehosted✓	
(web) Screen Sharing	gehosted✓	
(web) Audio/Video	gehosted✓	
(web) Whiteboard	gehosted✓	
(web) Chat	gehosted✓	
(web) F&A[3]	gehosted✓	
backup: (web) Audio	Telefonkonferenz - weltweite Einwahlnummern✓	
(host/target) Shell	gehosted - ssh (port 22) - ssh client✓	
Backup: (host/target) Shell über https	gehosted - https (port 443) - Browser✓	
benutzerdefinierter Inhalt		Der Inhalt kann an Ihre Bedürfnisse angepasst werden[4]✓
benutzerdefinierte Probleme		unternehmens-/projektspezifische Probleme können diskutiert & vielfach gelöst werden ✓
große Flexibilität		Beratung inkludiert[5]✓

Technische Voraussetzungen für die Teilnahme an einer Remote-/Online-Schulung

/> (ship-it/Web/Host/Zielhardware/Telefon) E-Mail-Adresse, um Login Informationen zu erhalten.

/> (Web) Screen Sharing/Audio/Video/Whiteboard/Chat/Fragen&Antworten:
`https://www.bigmarker.com` erfordert dies: `https://rlbl.me/bm-req`.

/> Backup: (Web)/Audio (Telefon) Telefonkonferenz:
`https://www.turbobridge.com/international.html`

/> (Host/Zielhardware) Shell: Port 22 nicht blockiert:
so etwas wie: `ssh <user>@vlabx.dyndns.org`

/> Backup: (Host/Zielhardware) Shell über Browser: Port 443 nicht blockiert:
so etwas wie: `https://vlabx.dyndns.org`

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`

[1] guaranteed to run - findet statt
[2] wenn Sie 3 Plätze und mehr buchen gibt es Rabatt
[3] Fragen und Antworten
[4] Kapitel meiner vorhandenen Schulungen können hinzugefügt/entfernt werden
[5] während des Trainings

/> *"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] *"The rise of copyright trolls"*
https://lwn.net/Articles/721458/

[2] *"Zielhardware"*
https://rlbl.me/hw

[3] *"Embedded Linux - Von Systemarchitektur zu Echtzeit (5 Tage)"*
https://rlbl.me/elisa

[4] *"Einführung in Embedded Linux in Theorie und Praxis - ein Intensivkurs (3 Tage)"*
https://rlbl.me/elin

[5] *"Embedded Linux Kernel Interna und Gerätetreiber (5 Tage)"*
https://rlbl.me/ldd

[6] *"Vortragsmethoden für Schulungen"*
https://rlbl.me/delivery

[7] *"Reliable Embedded Systems: Kontakt"*
https://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