Einführung in Embedded Linux und Yocto

Kursziele

Nach einer kurzen Einführung in Embedded Linux (2 Tage), möchten wir Ihnen das Wesentliche zur Nutzung des Yocto Projekts (3 Tage) vermitteln. Nach der Einführung werden wir sehen, wie ein BSP/Framework-Betreuer das Yocto-Projekt verwenden würde, sowie Entwickler, die vielleicht nicht einmal wissen wollen, dass sie es verwenden.

Beschreibung

Dieses fünftägige Training kombiniert Theorie mit praktischen Übungen, um Embedded Linux und das Yocto-Projekt vorzustellen.
Es beantwortet häufig gestellte Fragen wie:

- Was ist GNU/Linux?
- Warum Upstream verwenden?
- Woher bekommt man U-Boot/den Kernel? Wie konfiguriert/baut/installiert man das?
- Wie funktioniert Interprozesskommunikation und was sollte man verwenden/vermeiden?
- Ist es wirklich notwendig, für jedes GNU/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 zu machen?
- ... und vieles mehr

Auf der Zielhardware (z.B. phyBOARD-Mira i.MX 6 Quad - Full Featured) werden praktische Übungen durchgeführt. Nach dem Training können die Studenten für den Yoto spezifischen Teil ein Docker-Image mit Ubuntu 16.x und allen vorinstallierten Abhängigkeiten sowie die Beispiele herunterladen, um mit dem Kursmaterial in ihren eigenen Labors zu arbeiten. Bitte beachten Sie, dass die ersten zwei Tage Einführung in Embedded Linux möglicherweise nicht ausreichen, um dem Yocto-Training zu folgen. Vielleicht möchten Sie hier einen Blick drauf werfen. In den ersten beiden Tagen schauen wir uns kurz an, wie Embedded GNU/Linux funktioniert und konfigurieren/bauen unter anderem den Linux-Kernel.

Wer sollte teilnehmen?

Sie verwenden bereits GNU/Linux für Ihre Projekte und haben wahrscheinlich schon von dem Yocto Projekt gehört, haben es aber nicht gewagt, es näher zu betrachten oder hatten Schwierigkeiten damit. Sie wissen nicht, ob und wie Ihr täglicher Workflow in dem YP untergebracht werden kann und finden das YP im Allgemeinen ziemlich kompliziert. Warum brauchen wir all das, obwohl (angeblich) alles bisher viel einfacher war? Nach dem Training sollten Sie entscheiden können, ob Sie das YP brauchen oder nicht. Der Workshop richtet sich an Software-, Entwicklungs-, Systemingenieure, Tester, Administratoren, Ingenieure und andere an dem YP Interessierte, die minimale Kenntnisse von (Embedded) GNU/Linux haben.

Voraussetzungen

- Grundlegende Vertrautheit mit der Verwendung eines GNU/Linux-Systems (z.B. Ubuntu) als Endbenutzer im user space
- Grundlegende Vertrautheit mit einer Befehlszeilen-Shell
- Grundkenntnisse der Programmierung von User/Kernel-Space mit GNU/Linux
- Mittelmäßige Programmierkenntnisse in der Sprache C
- Es hilft, mit ”Embedded GNU/Linux Systems Architecture (5 days)" oder ”Introduction to Embedded Linux in Theory and Practice - a Crash Course (3 days)” vertraut zu sein, aber selbst wenn Sie es nicht sind, werden wir versuchen, Sie in den ersten zwei Tagen auf den notwendingen Wissensstand zu bringen.
- Es kann hilfreich sein, wenn Sie ”Embedded GNU/Linux Device Drivers and Kernel Internals (5 days)” besucht haben, aber das ist nicht wirklich eine Voraussetzung. Es genügt zu wissen, wie man den GNU/Linux-Kernel, Kernel-Treiber in/out of tree und den fdt baut, um dem BSP/Kernel Teil des Yocto Trainings folgen zu können und diese Themen werden wir in den ersten beiden Tagen behandeln.

Timetable

- Trainingszeiten Montag bis Freitag von 09:00 bis 17:00 Uhr mit Kaffee- und Mittagspausen

Trainer

Robert Berger berät und trainiert Menschen rund um die Welt auf einer Mission, ihnen dabei zu helfen, bessere Embedded-Software zu erstellen. Seine Spezialtäten sind Schulung und Beratung auf dem weiten Feld der Embedded-Software und zwar von kleinen Echtzeitsystemen bis hin zu Multi-Core-Embedded-Linux.

Die Bücher und Beispiele nehmen die Schulungsteilnehmer normalerweise nach der Schulung mit nach Hause.

01.- 02. April 2019 I Programm I Einführung in Embedded Linux

Einführung

- Einführung
- Geschichte

Eval Board

- Eval Board
- GNU/Linux auf einem PC booten
- GNU / Linux auf einem eingebetteten System booten
- Bootsequenz
- SD-Kartenpartitionen

Was man so braucht

- U-Boot
- U-Boot: Schickes Zeug
- U-Boot: Runterladen/Konfigurieren/Bauen/Installieren
- U-Boot: Befehle
- Fdt
- Kernel
- Kernel: Runterladen/kbuild
- Kbuild
- Kernel: Konfigurieren/Bauen/Installieren
- Kernel: fdt
- Kernel: Module

Kernel Module

- ...können sein
- init/exit
- Lizenzierung
- tainted Modul/Kernel
- EXPORT_SYMBOL()
- out of tree .ko Makefile
- Modul-Init-Werkzeuge
- Modul im Kernel Tree
- Bauen und Installieren
- Laden
- Parameterübergabe
- Zugriff auf TCB

Character Treiber

- Gerätedateien: Einführung, Gerätetypen, major/minor, Architektur
- Treiber-Kernel-Schnittstelle
- Gerätetreiber: Einführung, Anmeldung, Initialisierung, Öffnen/Schließen
- Misc. Char Treiber

User Space Fehlersuche/-behebung

- Fehlersuche/-behebung: Einfache Tools, lsof, ltrace,strace,…procfs, top, netstat, syslog,...
- Fehlersuche/-behebung: Fortgeschrittene Tools: Was ist ein Debugger?, gdb auf der Zielhardware, gdb Remote-Debugging

Kernel Fehlersuche/-behebung

- Debugging-Intro
- KGDB/KDB
- JTAG

Prozess/Interprozess-kommunikation	▪ IPC-Einführung: Unix/Linux-Architektur, Was ist Betriebssystem?, Was ist ein Scheduler?, Linux-Scheduler, Linux-Prioritäten, ◦Linux-Scheduler(s), Linux Scheduling Klassen, Prozesse/Tasks/Threads, errno, fork(), Prozessbeendigung, Prozesszustände, Zombies, Mehr über Prozesse, Sehen wir uns einen Prozess an ▪ Einfache Interprozesskommunikation: Shell-Umleitung, Shelling out, temporäre Dateien ▪ Interprozesskommunikation generisch ▪ Interprozesskommunikation: Nachrichtenübergabe(message passing) im Vergleich zu gemeinsam genutzter Speicher(shared memory) ▪ Fortgeschrittene Interprozesskommunikation: Pipes, Signale, Unterbrochene System Calls,POSIX.4 Nachrichtenwarteschlangen (Message Queues), Semaphoren Einführung, Mutex, Semaphoren, gemeinsam genutzten Speicher(Shared Memory), Sockets, select (self-pipe Trick), Andere Interprozesskommunikations-Mechanismen ▪ Interprozesskommunikations-Techniken die man vermeiden sollte
Echtzeit	▪ Voraussetzungen: Kernel im Vergleich zu User Space, Toolchain, Program Sections, Interrupts, Reentrant Code ▪ Echtzeit Einführung: Time/Utility Funktionen, Was ist Echtzeit?, Determinismus, Was ist harte Echtzeit? ▪ Echtzeit-Linux ▪ "Grade/Stufen" des Echtzeitverhaltens ▪ Dual Kernel: Xenomai, Messergebnisse ▪ Echtzeit-Mythen

03.-05. April 2019 I Programm I Einführung in Yocto

Einführung in Yocto

- Was ist Yocto?
- Warum das YP benutzen?
- Was ist das YP?
- Einige Werkzeuge unter dem YP Schirm: Poky, BitBake, OE-Core, Metadaten

Der Yocto Autobuilder

- Einführung: Was ist der Yocto Autobuilder?, Docker container (pull, container starten)
- Yocto Build-Umgebung ohne Docker/Yocto Autobuilder

Der YP Workflow

- Einführung
- Arbeitsablauf (Workflow): OE-Architektur
- Konfiguration
  - Benutzer Konfiguration
  - Eigenschaften (Features):
    Maschinen Eigenschaften (Machine Features), Distro Eigenschaften (Distro Features), Kombinierte Eigenschaften (Combined Features), Image Eigenschaften (Image Features)
  - Recipe Versionierung:
    Einführung, Bindestriche, Basierend auf Versions Kontrollsystemen (Source Code Management), Fallstricke, Entwicklungs-/Stabile Versionen, Overrides
  - Metadaten (Recipes)
  - Maschinen (BSP) Konfiguration
  - Distributions Policy
- Sourcen
- Bauen: Sources fetchen: do_fetch, do_unpack, Patchen, Konfigurieren/Kompilieren/Installieren, Pseudo, recipetool
  - Beispiele für Recipes:
    Paket von einer einzelnen .c Datei, Paket basierend auf Autotools, App. in mehrere Pakete aufgeteilt
  - Analyse der Ausgabe/Pakete (Packaging)
  - Generierung von Images
  - Generierung von SDKs
  - Images anpassen: Einführung, local.conf, IMAGE_FEATURES, Benutzerdefinierte .bb-Dateien - von core-image vererbt, Benutzerdefinierte .bb-Dateien - basierend auf core-image-minimal, Benutzerdefinierte packagegroups

BitBake

- Geschichte
- Syntax: Variable Erweiterung (Variable Expansion), Variable Zuweisung (Variable Assignment), Vor-/Anhängen (Pre-/Append), Entfernen (Override Stil Syntax), Variable Flag-Syntax, Bedingte (Conditional) Syntax (Overrides)
- Fehlersuche/-behebung: BitBake Fehlersuche/-behebung, Recipes finden, Images finden, Packagegroups finden, BitBake Umgebung, BitBake logs

	▪ Noch mal BitBaken: das Bauen eines spezifischen Tasks erzwingen, cleansstate, Stamp ungültig machen ▪ Devshell ▪ Abhängigkeiten ▪ Pakete ▪ Killall Bitbake ▪ BitBake mit Ncurses-Wrapper ▪ Werkzeuge/Tweaks ▪ Aufräumen: Aufräumen um Speicherplatz zu gewinnen, Aufräumen damit neu gebaut werden muss
Layer	▪ Einführung ▪ bitbake-layers Werkzeug ▪ yocto-layer Werkzeug
BSP	▪ Intro ▪ Systementwicklungs-Workflow ▪ BSP Entwicklungsanleitung: bsp-tool
Kernel	▪ Intro ▪ Systementwicklungs-Workflow ▪ Kernel-Entwicklungsanleitung: defconfig + Konfigurationsfragmente, in tree kmod, out of tree kmod, fdt
Anwendungs/Software Entwicklungskit	▪ Software-Entwicklungskit: Einführung, Cross-Entwicklungs Toolchain, Sysroot, Der QEMU-Emulator, Eclipse Yocto Plug-in, Werkzeuge zur Leistungsverbesserung ▪ Installieren von SDKs und Toolchains ▪ Cross-Toolchains/SDKs: Einführung, Erstellen eines Cross-Toolchain-Installers, Verwenden des Standard-SDKs, Cross-Toolchain+Makefile, Cross-Toolchain+Autotools, Autotooled lib + App., recipes, Erweiterbares (Extensible) SDK
Paketverwaltung	▪ Softwareaktualisierungen ▪ Mit Paketen arbeiten: IPK, Einen package feed erstellen, Installieren eines Pakets mit opkg auf dem Zielsystem
Lizensierung	▪ Einführung ▪ Fügen Sie dem YP eine benutzerdefinierte Lizenz hinzu ▪ Erfüllung der Open-Source-Lizenz Bestimmungen mit dem YP
Devtool	▪ Einführung ▪ Ein Recipe hinzufügen/bauen/installieren ▪ Einen Layer erstellen/hinzufügen ▪ Finish ▪ Ein Recipe ändern/aktualisieren ▪ Bauen/Ausführen ▪ Ein Image bauen

Programmänderungen vorbehalten

Training for Professionals
Einführung in Embedded Linux und Yocto
01.-05. April 2019 I Workshop I WEKA FACHMEDIEN

DESIGN& ELEKTRONIK
KNOW-HOW FÜR ENTWICKLER

Anmeldecoupon

Ausfüllen, abschicken, teilnehmen.

*Alle mit * gekennzeichneten Felder sind Pflichtfelder.*
Sie erhalten eine Anmeldebestätigung per Mail.

Kontakt:

Laura Lermer
Tel.: + 49 (0) 89 / 255 56 – 1725
Fax: + 49 (0) 89 / 255 56 – 0725
Email: llermer@weka-fachmedien.de

Nachname * | **Vorname *** | **Anrede ***

Firma | **Abteilung** | **Jobtitel**

Straße/Hausnr.*

PLZ * | **Ort ***

Tel./Fax | **Email ***

Hiermit melde ich mich verbindlich an: *

Einführung in Embedded Linux

O Teilnahme vom 01.-02. April 2019

Einführung in Yocto

O Teilnahme vom am 03.-05. April 2019

Einführung in Embedded Linux und Yocto

O Teilnahme vom 01.-05. April 2019

Datum / Unterschrift *

Einführung in Embedded Linux und Yocto	
Teilnahmegebühren	
Nur Einführung in Embedded Linux I 01.-02. April 2019	€ 1.290,00
Nur Einführung in Yocto I 03.-05. April 2019	€ 1.690,00
Kombi Embedded Linux und Yocto I 01.-05. April 2019	€ 2.490,00

Teilnahmebedingungen:
Es gelten die Allgemeinen Geschäftsbedingungen unter www.training-for-professionals.de.
Die Preise verstehen sich zzgl. der gesetzl. MwSt. (19%). In den Teilnahmegebühren enthalten sind die Teilnahme an den gebuchten Tagen, Unterlagen und Teilnahmezertifikat, sowie Erfrischungen und Mittagsbüffet. Bei Stornierung der Anmeldung bis 22 Tage vor Trainingsbeginn erheben wir eine Bearbeitungsgebühr in Höhe von € 100,00 (zzgl. gesetzl. MwSt.), bei Absage ab 21 Tage vor Trainingsbeginn oder Nichterscheinen wird die gesamte Teilnahmegebühr fällig. Eine Vertretung des angemeldeten Teilnehmers ist jederzeit möglich. Der Veranstalter behält sich vor, bei Nichterreichen einer Mindesteilnehmerzahl, den Workshop abzusagen. Hierdurch entsteht kein Anspruch des Teilnehmers auf Schadensersatz. Bei Anmeldung von mind. 2 Personen einer Firma, erhält die zweite Person und jeder folgende Teilnehmer derselben Firma 10% Rabatt auf die Teilnahmegebühr.

Veranstaltungsort: WEKA FACHMEDIEN GmbH, Richard-Reitzner-Allee 2, 85540 Haar bei München

Faxen Sie den ausgefüllten Coupon an +49 (0) 89 / 255 56 – 0725 oder buchen Sie direkt im Internet unter www.training-for-professionals.de