Embedded Linux Entwickler / Architekt

1 Skills

1.1 Spezialgebiete

- GNU/Linux-Experte *(from scratch und alle gängigen Distros)*
- Linux-Embedded *(Kernel, BSP, HIL, IOT, Realtime, ...)*
- Kernel-/Treiber-Entwicklung / Board support packages
- Build- und Deployment-Automatisierung und Entwicklungsumgebungen
- Cloud und IOT

1.2 Branchenerfahrung

- Automotive *(zB. Bahntechnik, Tankfahrzeuge, ...)*
- Medizinprodukte / Medizintechnik *(zB. IEC-62304 / Klasse C, Spirometrie, EKG, Reinigung/Sterilisation)*
- Finanzwesen - internationale Banken, Börsenhandel, Payment/Clearing
- IT Geräteentwicklung *(Embedded systems, IOT)*
- Internetprovider + Telco *(selbst ISP/Operator)*
- Medizin *(Ärzteverbände)*
- Maschinenbau / Anlagenbau / Gebäudetechnik
- Öffentliche Verwaltung und Rechtspflege
- Politik und NGOs *(selbst Abgeordneter)*
- Meßtechnik und -Kalibrierung
- Autohäuser und KFZ-Werkstätten
- Sport- und Kultur-Verbände
- Gastronomie
- Einzelhandel
- Textil-Manufaktur
- Neue Medien

1.3 Betriebssysteme, Programmierwerkzeuge, Toolkits, Libraries

- **Architekturen/Plattformen:**
  intel/x86, x86_64, m68k, ppc, arm, CBM, Embedded *(zB. Router, Bahntechnik, Maschinenbau)*
- **Interfaces:**
  RS282, RS485, CAN, ISA, USB, ATA/IDE, SCSI, WLAN, Ethernet, DSL, Cable, ISDN, SDH, SPI, SSI, MVB, uvm.
- **Betriebssysteme:**
  GNU/Linux, Debian, Devuan, *BSD, Plan9, Oberon, DOS, Novell, Win16, Win32, L4, GECOS, Android
- *nix utils *(awk,sed,perl,...)*
- **Programmiersprachen:**
  C/C++ *(über 20 Jahre)*, Oberon/Modula *(9 Jahre)*, Pascal *(über 10 Jahre)*, BASIC *(über 10 Jahre)* Unix-Shellprogrammierung *(über 20 Jahre)*, postgreSQL PL/SQL *(15 Jahre)*, Perl *(über 10 Jahre)*, PHP *(über 10 Jahre)*, Java *(über 10 Jahre)*, HTML + CSS *(über 10 Jahre)*, LaTeX *(über 10 Jahre)*, Javascript/ECMA-Script *(über 10 Jahre)*, Lua, Prolog, Assembler, XUL - XML Userinterface Language, Python *(5 Jahre)*, Ruby / Rails, Haskell, m4, tcl/tk
- **Builders:**
  GNU make, Imake, Autoconf/Autotools, Unitool, TreeBuild
- **Embedded-Toolkits:**
  PTXDist, ELBE, OpenWRT, embedian, yocto
- **SCM:**
  Git, CVS, Perforce P4, Subversion, Redmine, Bugzilla, Trac
- GNU Toolchains
- **GUI-Toolkits:**
  Gtk, Qt, Motif, Xt, Xaw
- **Code-Generators:**
  Yacc, Lex, Bison
- **Database tools and libraries:**
  unixODBC, JDBC, libpq *(PostgreSQL)*, mySQL, sqLite, BerkeleyDB
- **Security:**
  OpenSSL, GNU TLS, OpenSSH
- **XML:**
  SAX, Expat
- **Graphics:**
  X11, SDL, DirectFB, Linux-FB, Wayland
- **Networking/IPC:**
  SysV-IPC, 9P2000, Plan9Port *(Plan9 on Unix)*, NetBEUI, TCPIP

- **Typesetting:**
  LaTeX, Docbook
- **IDE:**
  Emacs, Borland, Eclipse, MSVS, KDevelop

2 Referenzprojekte

Im Folgenden Abschnitt wird eine Auswahl realisierter Projekte im Bereich Linux-Embedded *(unsortiert)* vorgestellt.

2.1 Embedded: Spriometrie- und EKG-Meßcomputer für klinische Studien

- Weiterentwicklung und Moderinisierung medizinischen Meßgeräten für klinische Studien
- Spirometrie- und EKG-Messungen nach Studien-spezifischen Workflows
- Integration einer zugekauften EKG-Analyse-Kompente *(Corscience HES)*
- Portierung auf neue Hardware *(neues BSP)*
- Treiber-Entwicklung für den integrierten Drucker *(angepaßter HP OJ-470)*
- Source control management und vollautomatische Buildprozesse *(DevOps)*
- Einführung eines Configurations-Systems zur Ablösung bisheriger studienspezifischer Code-Anpassungen
- Entwicklung eines Standardgeräts zur regulären Nutzung in der Praxis *(außerhalb von Studien)*

Tools+Spachen: C/C++, Perl, GNU/Linux, Kernel-Entwicklung, Git, GCC, Crosscompiler, ARM/iMX.21, GNU make

2.2 Beratung/Entwicklung Embedded-Linux Medizintechnik

- Beratung und Entwicklung für ein Medizinprodukt *(Reinigung und Sterilisation)*
- IEC-62304 / Klasse C
- ptxdist-basiertes BSP *(Board Support Package)* nebst Treiberentwicklung/-Anpassung *(uA. auch Portierung von Qt5 auf KMS/DRI, GPU-Treiber, etc)*
- Build- und Deployment-Prozesse mit Hinblick auf Fertigung und Wartung im Feld *(DevOps)*
- Automatisierung von Dokumentation und Code-Generierung

- Sicherheitskonzepte *(zB. Manipulationssicherheit und Knowhow-Schutz)*
- Konzepte für Wartung und Langzeitpflege der Software
- cryptographisch gesicherte Wartungsschnittstelle
- Evaluation neuer Prozessor-/SOC-Generationen *(uA. IMX6)*
- Review und Abnahme der Zulieferungen externer Dienstleister

Tools+Spachen: LaTeX, C/C++, Linux-Kernel, ptxdist, Git, shellscript, python, cmake, GNU make, OpenGL/Mesa, Qt5, imx53, imx6, CAN

2.3 (IOT) Notrufsystem für Aufzugssteuerung

- Entwicklung von EN-81 konformer Notruf-Funktionalität in einer existierenden Aufzugs-Telemetrie
- Anbindung von Callcenter-Backends
- automatische Provisionierung und ERP-Anbindung
- Treiber-Entwicklung / Kernel-Anpassung *(HMI, Codecs, GSM-TA, ...)*
- Weiterentwicklung des Board Support Package
- Konsolidierung der SW-Architektur zur Unterstützung verschiedener Produktvarianten und landesspezifischer normativer Vorgaben
- Vereinfachung und Optimierung der gewachsenen Codebase, Ablösung von Legacy-Code durch Linux-Bordmittel
- Schulung zur Entwicklung unter GNU/Linux / Linux-Embedded
- Automatisierung und Optimierung von Build- und Deployment- Prozessen und Source-Control-Management
- Beratung bzgl. Software-Architektur und Entwicklungsstrategie
- Schulung der Kollegen bzgl. Entwicklung unter GNU/Linux

Tools+Spachen: C/C++, Linux-Kernel, PtxDist, Azure, Git, Perforce, Redmine, LaTeX, RS485, CAN, RS232, cmake, javascript

2.4 (IOT) Beratung/Entwicklung Embedded-Linux Aufzugstechnik

- Beratung bei einer Linux-Embedded-basierten Aufzugssteuerung
- Automatisierung und Optimierung von Build- und Deployment- Prozessen und Source-Control-Management *(DevOps)*
- Beratung bzgl. Software-Architektur und Entwicklungsstrategie

- Schulung der Kollegen bzgl. Entwicklung unter GNU/Linux
- Ablösung von Legacy-Code durch Linux-Bordmittel
- Code-Reviews und Cleanup

Tools+Spachen: C/C++, Linux-Kernel, PtxDist, Git, Perforce, LaTeX, RS485, CAN, RS232, cmake, javascript, DevOps

2.5 (IOT) Beratung/Entwicklung Embedded-Linux Aufzugstechnik

- Weiterentwicklung eines Telemetrie-/Diagnose-Gerät für Aufzugsanlagen *(predictive maintenance)*
- BSP- und Kernel-Anpassungen
- Ablösung von Legacy-Code durch Linux-Bordmittel und Optimierung
- Refactoring der HW-Ansteuerung / Treiberschicht *(HMI, Modem/GSM-TA, etc)*
- Refactoring der Techniker-Funktionen
- Refactoring der Cloud-Kommunikation
- Implementierung von Selbstdiagnose *(zB. Verbindungstest, Aufzugswärter, etc)*
- Anbindung verschiedener Aufzugssteuerungen
- Harmonisierung auf internationale Gegebenheiten *(div. Gehäuse/LED-Panel, Bedienfunktionen, etc.)*
- Architektur- und Code-Review

Tools+Spachen: C/C++, Linux-Kernel, PtxDist, Azure, Git, Perforce, Redmine, LaTeX, RS485, CAN, RS232, cmake, javascript

2.6 (IOT) Beratung zu Entwicklungsmethoden im Embedded-Linux-Bereich

- Kunde ist Hersteller von Meß- und Erfassungstechnik für Tanklastfahrzeuge *(zB. Treibstoff, Milch, etc)*
- Konzept zur Modernisierung der Basis-Software und Entwicklungsprozesse, Build-/Deployment, DevOps, Langzeitpflege und Portierung auf neue Hardware
- Konzeption für hochverfügbare Datenbank-Infrastruktur

Tools+Spachen: Git, SVN, C/C++, Linux-Kernel, GNU make, USB, DevOps

2.7 CAN/RS485 datalogger (Maschinenbau / IOT)

- Konzeption / Entwicklung eines CAN- und RS485 Datalogger *(mobile und im Feld schwer zugängliche Geräte)*
- Für mobile und im Feld schwer zugängliche Geräte mit oft nur sporadischer Datenverbindung
- Minimales Linux-System *(PtxDist)* mit kundenspezifischem BSP *(incl. Kernel-/Treiber-Entwicklung)*
- Logging über verschiedene Schnittstellen *(zB. CAN, RS485, RS232, MODBUS, IP)* und kundenspezifischen Protokollen
- ggf. Sonderlösungen zur Datenanbindung *(zB. AX25 / packet radio)*
- Logging auf interne microSD und Upload via UUCP und RSYNC.
- Vollautomatisches Build, Deployment, Field-Updates
- Konfigurations-Management
- Factory-Tools

Tools+Spachen: C/C++, Linux-Kernel, Barebox, PtxDist, Git, CAN, RS232, RS485, UUCP, DevOps

2.8 (DevOps) Docker-Buildpackage: Build-Automatisierung für Debian-/Ubuntu-Pakete

- Tool zum einfachen und vollautomatischen Build von deb-Paketen und Paket-Repositories
- strikt isolierte Build-Umgebung via Docker
- deutlich leichter aufzusetzen als bisherige Tools *(zB. pbuilder oder buildd)*
- dient uA. der Pflege kundenspezifischer oder auch eigener LTS-Repositories
- einfache Integration in CI *(zB. Jenkins)*

Tools+Spachen: Debian, APT, dpkg, Docker, docker-buildpackage, CI, Jenkins, DevOps

2.9 (IOT) Kernel / BSP für Duagon IONIA (Automotive, Bahntechnik)

- Generisches Board Support Package für Duagon IONIA *(Bahntechnik)*
- Basis-System und reproduzierbares Build via PtxDist
- aktueller Mainline-Kernel und Barebox
- Kernel-Treiber für Backplane *(serial/tty)* und verschiedene Module *(IIO)*
- Beratung/Schulung zu IIO

Tools+Spachen: C/C++, Linux-Kernel, PtxDist, ADC, Git, CAN, RS232, Duagon IONIA, MVB, cmake

2.10 (IOT) Lizenzmanagement für Embedded Devices (Automotive, Bahntechnik, Schiffsbau)

- Konzeption/Infrastruktur für Lizenzmanagement bei Telemetrie-/Diagnosegeräten
- Verwaltung der Anlagen und Freischaltung gebuchter Features im Embedded-Device
- Verwaltung der Lizenznutzung zugekaufter SW-Komponenten
- Autogenerierung kundenspezifischer Geräteconfigurationen
- Cryptographische Absicherung und Secure Boot
- Reporting-Werkzeuge, ERP-Anbindung, etc

Tools+Spachen: PtxDist, OpenSSL, GnuPG, Packaging, Jenkins, imx6, trusted boot, PostgreSQL, ERP, DevOps

2.11 (IOT) Maßgeschneiderte GNU/Linux-Firmware für WLAN-VoIP-Router

- in einem Forschungsprojekt zu mobiler Infrastruktur für Telefonie- und Videodienste auf Basis gängiger WLAN-Technik
- angepaßte/optimierte Firmware für gängige WLAN-Router und Integration in Provider-Backbones *(zB. via MPLS)*
- Board Support Packages auf Basis eines selbst entwickelten Embedded Build-System *metux Briegel Builder*
- Build- und Deploymentprozesse für hochautomatisierte Pflege und Wartung *(DevOps)*
- Integration von MPLS in die Linux-basierte Firmware
- Container-Lösung zur einfachen Integration von Drittsoftware auf die Router *(zB. Nutzung der Leerlaufkapazitäten für Cloud-Computing)*
- Anpassung zahlreicher Opensource-Pakete für Embedded / Crosscompiling
- Weiterentwicklung von Linux-Treibern *(WLAN, MPLS/RSVP, etc)*

Tools+Spachen: C/C++, Perl, Java, MPLS, WLAN, Intel Geode, Crosstool, Briegel, GNU make, GNU autotools, diffutils, CVS, Subversion, Shellscript, GNU/Linux, metux Briegel

2.12 (IOT) HW-Anforderungen und Embedded-Linux BSP (Bahntechnik)

- Entwicklung eines Board Support Package für ein Bahntechnik-Gerät *(predictive maintenance)*
- HW-Systemanforderungen und Beratung zur Chip-Auswahl
- Spezifikation der Schnittstelle zwischen CPU und ADC-Baugruppe/FPGA
- Anpassung von Kernel und Bootloader, Treiber-Entwicklung
- automatisierte Build-Umgebung via PtxDist und Jenkins
- Factory-Tools

Tools+Spachen: C/C++, Linux-Kernel, Barebox, PtxDist, ADC, Git, CAN, RS232, Bitbucket, Jenkins, DevOps

2.13 (IOT) Beratung/Entwicklung Embedded-Linux/Diagnose - Bahntechnik, Hochsee-Schiffahrt, Kraftwerke

- Einführung nachhaltiger Entwicklungsprozesse für ein Telemetrie-/Diagnose-Gerät in der Bahntechnik, Hochsee-Schiffahrt und Kraftwerken.
- Source control management und Release management
- Hohe Ausfallsicherheit und einfache Langzeit-Wartung
- Konsolidierung der Embedded-Linux Plattform *(uA. generische Treiber-APIs, zB. IIO)*
- robuste, vollautomatische Build-/Deployment-Prozesse *(DevOps)*
- Kernel-Portierung / Treiberentwicklung für div. Controller *(uA. Duagon IONIA)*
- Configurations- und Lizenzmanagement
- Portierung von nodejs auf embedded systems
- Beratung zur SW-Architektur und Entwicklungs-Prozessen
- Beratung zur Chipset-Auswahl / HW-Architektur für Custom Boards
- Evaluation div. HW *(zB. NI cRIO)* auf SW-technische Nutzbarkeit für GNU/Linux-Systeme

Tools+Spachen: C/C++, Linux-Kernel, PtxDist, Nodejs, Barebox, ADC, Git, RS485, CAN, RS232, Duagon IONIA, MVB, cmake, javascript, DevOps

2.14 HIL-Test/Werksprüfung: Gebäudetechnik / Embedded-Linux

- Test-System zur Werksprüfung von Premium-Haustechnik

- Netzwerk-Schnittstellen *(Last- und Langzeittests, Stabilität, ...)*
- Display- und Touch: Kalibierung und Qualitätskontrolle
- Thermische Prüfung
- Prüfung von IOs.
- Test von Audio/Video-Streaming, Codec, GPU

Tools+Spachen: C/C++, Linux-Kernel, PtxDist, Git, RS485, CAN, RS232, cmake, Qt, GStreamer, Labview

2.15 (DevOps) Vollautomatisches Deployment virtueller Entwicklungsumgebungen

- Bereitstellung von kundenspezifischen Embedded-Entwicklungsumgebungen als VM- und Container-Images
- Vollautomatisches Build und Provisionierung *(via Ansbile und Vagrant)*
- automatisches Rollout auf den Endgeräten
- CI-Integration *(Jenkins)* und automatische Provisierung der CI selbst
- Paketierung diverser Zusatz-SW für Debian/Ubuntu *(VTK8, Eclipse, Sigasi Studio, Vivado, ...)*
- Automatisierung des Deployment kommerzieller Software *(Vivado, Sigasi, Sysgo PikeOS/Codeo, Modelsim, Questa)*
- Lizenz-Prüfung *(zB. Einhaltung div. FOSS-Lizenzen)*

Tools+Spachen: Debian, APT, Ansible, Vagrant, Jenkins, ESXI, Docker, Vivado, Modelsim, Questa, PikeOS, Codeo, Eclipse, Sigasi

2.16 Embedded-Linux Entwicklungsplattform und Paket-Portierung

- Entwicklung kundenspezifischer Firmware für DSL-Router
- ermöglicht die Bereitstellung mehrere virtueller Anschlüsse und öffentlicher WLAN-Zugänge über bereits im Feld vorhandene DSL-Router
- Kernel-/Treiber-Anpassung, Build-/Deployment-Prozesse
- MPLS-Integration
- Anpassung zahlreicher Opensource-Pakete für Embedded/Crosscompiling

Tools+Spachen: C/C++, GNU make, Linux-Kernel, MPLS, Crosstool, Briegel, Diffutils, CVS, Perl, Shellscript, GNU/Linux, metux Briegel

3 R+D: Eigene Projekte und Produkte

3.1 Linux-Kernel: Modbus protocol stack

- Generische MODBUS Socket API *(analog zum CANSocket)*
- Treiber-Framework für MODBUS
- Integration in andere Kernel-Subsysteme, um via MODBUS angeschlossene Geräte über Standard-APIs *(zB. LED, hwmon, IIO, etc)* anzusprechen
- Security via Linux-Firewall *(netfilter/BPF)*

Tools+Spachen: Linux-Kernel, Modbus, Socket-API, Kernel-Treiber, netfilter, Linux-Embedded

3.2 Modellierung und Code-Generator für CAN-Umgebungen

- Modellierung von CAN-basierten Systemen *(Messages, Payloads, Bus-Topologien, Kommunikationsbeziehungen, etc)* via YAML
- Konsistenz- und Konfliktprüfungen
- leicht anpaßbare Code-Generatoren für die einzelnen Bus-Teilnehmer *(MCUs, Switches, Prüfgeräte, etc)* mit verschieden Zielsprachen und -Plattformen
- autogenerierte Firewall- und Routing-Regeln *(zB. netfilter/BPF)*
- für DevOps und agile Entwicklung gerüstet: leicht integrierbares und portierbares Baukastensystem statt monolithische SW-Suite

3.3 Plan9 subsystem für Linux

- Integration von Plan9 Basisfunktionen im Linux-Kernel
- Textuelle Usernames statt nur numerische IDs
- Deaktivierung von SUID innerhalb User-/Mount Namespace
- Mouting und Namespace-Forking für unprivilegierte User
- Authentifizierung/User-Wechsel via /dev/capuse

- Portierung des Factotum Authentifizierungs-Agent
- sukzessive Implementation diverser Plan9-Services

Tools+Spachen: Linux-Kernel, Plan9, C, LXC, coreutils, gcc, make, bash, rc, factotum

3.4 Linux-Kernel: Text-basierte Kernel-Schnittstellen via Dateisystem

- Textuelle Kernel-Schnittstellen und Devices als virtuelle Filesysteme statt klassische IOCTLs
- Vereinfacht Scripting und Debugging *(zB. keine nativen Bindings bei Scriptsprachen mehr nötig)*
- Keine Abhängigkeiten von Binärformaten *(zB. struct alignments, byteorder, etc)* mehr
- Ermöglicht direkte Verwendung von Geräten via Netzwerk-Filesysteme *(zB. NFS, 9P, etc)*

Tools+Spachen: C/C++, GNU make, Linux-Kernel, GCC, 9P, Shellscript

3.5 GNU Droid – Android-Umgebung auf GNU/Linux

- Android-Subsystem für GNU/Linux-Systeme
- Ermöglicht den Betrieb von Android-Apps innerhalb klassischer GNU/Linux Desktops
- Verwendung von Android-Komponenten als GUI-Toolkit
- Portierung des extreme powersaving *(wakelock, etc)* auf mobile Desktops
- Dalvik als Alternative zur klassischen JVM
- Integration klassischer GNU/Linux-Komponenten in die Android-Welt

Tools+Spachen: Android, Linux-Kernel, Git, C/C++, Java, Dalvik, cmake, GNU make, auto-tools, ptxdist, Shellscript

3.6 Briegel Builder - vollautomatisiertes Buildsystem für Firmware und Distros

- Buildsystem für sehr speziell angepaßte GNU/Linux-Systeme / -Images
- ähnlich wie zB. Ptxdist, jedoch außerhalb der Embedded-Welt, zB. Container, einsetzbar
- Vollautomatisiertes, reproduzierbares Build

Tools+Spachen: Java, PHP, GNU Make, Ant, PostgreSQL , Unitool, GNU Autotools, Kaffe, GCJ, Crosstool

3.7 Comprehensive Source Database

- Normalisierte Datenbank mit vielen Opensource-Quellpaketen und entsprechenden Versionen
- Crawler zur automatischen Erfassung neuer Versionen / Releases
- Automatisierte Benachrichtigungen und Anbindung an Build-/QA-Plattformen

Tools+Spachen: C/C++, PHP, PostgreSQL , plsql, Perl, CVS, Git, SVN, Mercurial, Shellscript, DevOps

3.8 OpenSource Software QM Taskforce

- Dachprojekt zur Pflege / QM von zahlreichen Opensource-Paketen
- Normalisierte Repository-Layouts and Versions-Schemata
- Automatisiertes Sammeln von Patches/Bugfixes aus den verschiedensten Distributionen
- Langzeitpflege *(LTS)* von ausgewählten Versionen
- Sicherstellung automatisierter Buildprozesse *(insbesondere zB. für Embedded: Crosscompiling)*

Tools+Spachen: DevOps, Git, C, Java, Perl, GNU diffutils, Subversion, CVS, Git, PHP, GNU make, GCC, Crosstool, Shellscript

3.9 Linux-Kernel: Toolkit zur Meta-Configuration

- Highlevel-Configuration des Kernel auf Basis von Aufgaben und Hardware-Profilen, statt individuelle Optionen
- Autogenerierung der Kernel-Config für verschiedenste Versionen aus der Meta-Configuration heraus
- Ableitung der Treiber-Configuration aus dem Devicetree

Tools+Spachen: C/C++, GNU make, Linux-Kernel, GCC, Shellscript, Git, DevOps