



Zephyr-Projekt

Echtzeit-OS für das Internet der Dinge

Projektarbeit

Die Linux Foundation hat mit dem Projekt Zephyr mit der Entwicklung eines Echtzeit-Betriebssystems für das Internet der Dinge (IoT) begonnen. Zephyr ist ein Open-Source-Betriebssystem mit dem Ziel ein solides OS für IoT Geräte mit geringen Ressourcen bereitzustellen. Es nutzt eine echtzeitfähige Kombination aus Nano- und Microkernel. Im Gegensatz zu einem Linux Kernel benötigt Zephyr nur zwischen 8 und 512 KByte an Arbeitsspeicher. Aktuell werden folgenden Plattformen unterstützt: x86, ARM und ARC EM4

Studiengang: Elektro- und Kommunikationstechnik

Autoren: Aaron Schmocker, David Wyss

Betreuer: Martin Aebersold

Auftraggeber: Martin Aebersold

Experten: Martin Aebersold

Datum: 29.09.2016

Versionen

Version	Datum	Status	Bemerkungen
0.1	29.09.2016	Entwurf	Titelblatt erstellt und Template für Linux angepasst

Management Summary

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus scelerisque, leo sed iaculis ornare, mi leo semper urna, ac elementum libero est at risus. Donec eget aliquam urna. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc fermentum nunc sollicitudin leo porttitor volutpat. Duis ac enim lectus, quis malesuada lectus. Aenean vestibulum suscipit justo, in suscipit augue venenatis a. Donec interdum nibh ligula. Aliquam vitae dui a odio cursus interdum quis vitae mi. Phasellus ornare tortor fringilla velit accumsan quis tincidunt magna eleifend. Praesent nisl nibh, cursus in mattis ac, ultrices ac nulla. Nulla ante urna, aliquet eu tempus ut, feugiat id nisl. Nunc sit amet mauris vitae turpis scelerisque mattis et sed metus. Aliquam interdum congue odio, sed semper elit ullamcorper vitae. Morbi orci elit, feugiat vel hendrerit nec, sollicitudin non massa. Quisque lacus metus, vulputate id ullamcorper id, consequat eget orci

Cite test: [1] [3] [?]

Inhaltsverzeichnis

Management Summary	i
1. Einleitung	1
1.1. TEST	1
1.2. TEST2	1
2. ZephyrOverview	3
2.1. Übersicht über Zephyr	3
2.2. Ziele	3
2.3. Aufbau	4
2.4. Aufsetzen der SDK unter Ubuntu	4
2.5. Vergleich mit anderen RTOS	4
2.6. Sicherheitsaspekte	4
2.7. Portierungsaufwand	4
3. nRF52DK	5
3.1. Aufsetzen der nordic SDK unter Ubuntu	5
3.2. Vorteile	5
3.3. Technische Beschreibung des Boards	5
3.4. Bluetooth Low Energy	5
3.5. Analog to Digital Converter	5
3.6. I2C Bus	5
4. DemoApp	7
4.1. Technische Übersicht	7
4.2. Hardware Dokumentation	7
4.3. Software Dokumentation	7
4.4. Softwaretests	7
5. Fazit	9
5.1. Vergleich Ist/Soll	9
5.2. Wünschenswerte Erweiterungen	9
Selbständigkeitserklärung	11
Glossar	12
Literaturverzeichnis	15
Abbildungsverzeichnis	17
Tabellenverzeichnis	19
A. Beliebiger Anhang	21
Stichwortverzeichnis	21
B. Weiterer Anhang	23
B.1. Test 1	23

1. Einleitung

1.1. TEST

1.2. TEST2

2. ZephyrOverview

2.1. Übersicht über Zephyr

Zephyr ist laut Beschreibung der Linux-Foundation [2] ein Open-Source-Echtzeitbetriebssystem, speziell optimiert für Anwendungen im Internet der Dinge. Die Architektur basiert auf einer echtzeitfähigen Kombination von Nano- und Mikrokern. Es wird aktuell von der Linux-Foundation in Zusammenarbeit mit den Firmen Intel, NXP und Synopsys in der Form eines Collaborative-Projects entwickelt. Dadurch soll versucht werden die bei der Linux- und Open-Source-Entwicklung verwendeten Arbeitsweisen und Ideen auch im Bereich der Industrie einzubringen. Ziel ist es ein robustes und sicheres Betriebssystem für das Internet der Dinge zu schaffen. Zephyr ist vollständig Open-Source steht laut Information des Newsportals Heise.de [3] unter der Apache Lizenz Version 2.0. Dieses Lizenzierungsmodell kommt Firmen und Unternehmen entgegen, welche den Einsatz von Open-Source-Software generell scheuen da diese oft unter der GNU General Public License (GPL) stehen. [2] Wird in Produkten Software verwendet welche unter der GPL lizenziert ist, zwingt dieses Lizenzmodell die Firmen dazu ihre Produkte ebenfalls unter GPL zu veröffentlichen. Dies beinhaltet auch sämtliche Änderungen welche vorgenommen wurden. Bei der Apache-Lizenz ist dies gemäss Definition [1] nicht zwingend.

Momentan unterstützt der Zephyr-Kernel gemäss Angaben auf der Projektseite [4] Prozessoren der Architekturen ARC, ARM-v7 aber auch x86. Dadurch ist das System auf populären Plattformen wie dem Arduino 101 Board, dem Arduino Due Board, dem NXP Freedom DK lauffähig und Intel Galileo Gen 2. Zur Kommunikation stehen unter anderem Protokolle Ipv4, Ipv6, Bluetooth 4.0, LoWPAN zur Verfügung.

2.2. Ziele

Zephyr ist für den Einsatz auf Geräten mit geringem Speicherplatz und feststehender Hardwarekonfiguration gedacht. Darunter fallen unter anderem Steuerungen für Heizungs- und Beleuchtungssysteme aber auch Geräte aus allen Bereichen des täglichen Lebens mit Internet-Anbindung.

Das ZephyrProjekt verfolgt laut Beschreibung der Linux-Foundation [2] folgende Ziele:

- Kleiner footprint - lauffähig mit minimal 10kB
- CPU unabhängige Architektur
- Modular und Skalierbar
- Hoche Sicherheitsstandards
- Unterstützt von Grund auf viele unterschiedliche Boards und Kommunikationsprotokolle
- Mächtige Entwicklungswerkzeuge
- OpenSource Kernel mit Apache v2.0 Lizenz

2.3. Aufbau

Das Zephyr OS setzt auf eine Kombination von Nano- und Mikrokernel. Dadurch soll Zephyr bereits mit nur 10 Kbyte an Speicherplatz lauffähig sein. Das macht Zephyr besonders für Anwendungen auf kleinen Mikrokontrollern attraktiv. Bei einem herkömmlichen Linux-Kernel wäre dies nicht denkbar. Gängige Adaptionen für Smartphone SoCs benötigen laut [1] in der kleinsten Konfiguration noch bis zu 200KB RAM und rund 1MB Flash.

Der Nanokernel bietet Echtzeit-Fähigkeiten. Die Zeit die der Nanokernel für die Abarbeitung einer Aufgabe benötigt ist also deterministisch. Dies unabhängig davon wie stark das System gerade ausgelastet ist. Für alle Aufgaben welche keine Anforderungen an Echtzeit-Verarbeitung stellen verfügt das System über einen Microkernel. [1] <http://linuxdevices.linuxgizmos.com/my-linux-is-smaller-than-your-linux-a/>

2.4. Aufsetzen der SDK unter Ubuntu

2.5. Vergleich mit anderen RTOS

2.6. Sicherheitsaspekte

2.7. Portierungsaufwand

3. nRF52DK

3.1. Aufsetzen der nordic SDK unter Ubuntu

3.2. Vorteile

3.3. Technische Beschreibung des Boards

3.4. Bluetooth Low Energy

3.5. Analog to Digital Converter

3.6. I2C Bus

4. DemoApp

4.1. Technische Übersicht

4.2. Hardware Dokumentation

4.3. Software Dokumentation

4.4. Softwaretests

5. Fazit

5.1. Vergleich Ist/Soll

5.2. Wünschenswerte Erweiterungen

Selbständigkeitserklärung

Ich/wir bestätige/n, dass ich/wir die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der im Literaturverzeichnis angegebenen Quellen und Hilfsmittel angefertigt habe/n. Sämtliche Textstellen, die nicht von mir/uns stammen, sind als Zitate gekennzeichnet und mit dem genauen Hinweis auf ihre Herkunft versehen.

Ort, Datum: [Biel/Burgdorf], 29.09.2016

Namen Vornamen: [Test Peter] [Müster Rösä]

Unterschriften:

Glossar

API	Application Programming Interface
CPU	Central Processing Unit
BLE	Bluetooth Low-Energy
SoM	System On Module
SoC	System On Chip
GPL	GNU General Public License

Literaturverzeichnis

- [1] A. S. Foundation, "Apache license 2.0." [Online]. Available: <http://www.apache.org/licenses/LICENSE-2.0>
- [2] L. Foundation, "Linux foundation announces rtos for iot devices."
- [3] T. Leemhuis, "Linux-foundation startet projekt für iot-betriebssystem."
- [4] ZephyrProject, "Zephyrproject documentation." [Online]. Available: <https://www.zephyrproject.org/doc/board/board.html>

Abbildungsverzeichnis

Tabellenverzeichnis

A. Beliebiger Anhang

B. Weiterer Anhang

B.1. Test 1

Phasellus eget velit massa, sed faucibus nisi. Etiam tincidunt libero viverra lorem bibendum ut rutrum nisi volutpat. Donec non quam vitae lacus egestas suscipit at eu nisi. Maecenas non orci risus, at egestas tellus. Vivamus quis est pretium mauris fermentum consectetur. Cras non dolor vitae nulla molestie facilisis. Aliquam euismod nisl eget risus pretium non suscipit nulla feugiat. Nam in tortor sapien.

B.1.1. Umfeld

Nam lectus nibh, laoreet eu ultrices nec, consequat nec sem. Nulla leo turpis, suscipit in vulputate a, dapibus molestie quam. Vestibulum pretium, purus sed suscipit tempus, turpis purus fermentum diam, id cursus enim mi a tortor. Proin imperdiet varius pellentesque. Nam congue, enim sit amet iaculis venenatis, dui neque ornare purus, laoreet porttitor nunc justo vel velit. Suspendisse potenti. Nulla facilisi.