Zephyr Project Overview

Anas Nashif
Agenda

- Zephyr Project Overview
- Architecture Overview
- The Zephyr Kernel
- Getting Involved
What is Zephyr?

- Microcontroller operating system
- Very small memory footprint (will run in 8k)
- Open Source under Apache* 2.0 license, hosted by Linux* Foundation
- Supports multiple architectures
Small OS & RTOS Market Analysis

Opportunity to build a leading IoT OS
Why Zephyr Project?

- Strategic Investment
- Best-of-Breed RTOS
- True Open Source
- Permissively Licensed
- Established Code Base
- Secure
Zephyr Project Features

- Modularity
- Security
- Connectivity
- Architecture
- Community Developed
Agenda

- Zephyr Project Overview
- Architecture Overview
- The Zephyr Kernel
- Getting Involved
Top 10 IoT Technologies for 2017/2018 (Gartner)

- IoT Security
- IoT Analytics
- IoT Device Management
- Low-Power, Short-Range Networks
- Low-Power, Wide-Area Networks
- IoT Processors
- IoT Operating Systems
- Event Stream Processing
- IoT Platform
- IoT Standards and Ecosystem

Source: [http://www.gartner.com/newsroom/id/3221818](http://www.gartner.com/newsroom/id/3221818)
Top 10 IoT Technologies for 2017/2018 (Gartner)

- IoT Security
- IoT Analytics
- IoT Device Management
- Low-Power, Short-Range Networks
- Low-Power, Wide-Area Networks
- IoT Processors
- IoT Operating Systems
- Event Stream Processing
- IoT Platform
- IoT Standards and Ecosystem

Source: [http://www.gartner.com/newsroom/id/3221818](http://www.gartner.com/newsroom/id/3221818)
Zephyr Overview

- Provide an OS that runs best on MCUs for wearable and IoT devices, where the cost of the silicon is minimal
- Highly Configurable, Highly Modular
- Kernel mode only

**Two Modes:**
- **Nanokernel**: Limited functionality targeting small footprint (below 10k)
- **Microkernel** (superset of nanokernel): with additional functionality and features

- No user-space and no dynamic runtimes
- Memory and Resources are typically statically allocated
- Cross architecture (IA32, ARM*, ARC, others under discussion)

### System Diagram

<table>
<thead>
<tr>
<th>Applications</th>
<th>3rd Party Libraries</th>
</tr>
</thead>
<tbody>
<tr>
<td>C APIs</td>
<td></td>
</tr>
<tr>
<td>CoAP, MQTT*, HTTP*, LWM2M*</td>
<td></td>
</tr>
<tr>
<td>DTLS, TLS, ..</td>
<td></td>
</tr>
<tr>
<td>IPv4*, IPv6</td>
<td>6LowPAN</td>
</tr>
<tr>
<td>Device Drivers</td>
<td>Device Management</td>
</tr>
<tr>
<td>Device Management</td>
<td>Power Management</td>
</tr>
<tr>
<td>Microkernel</td>
<td>Nanokernel</td>
</tr>
<tr>
<td>Nanokernel</td>
<td>Platform</td>
</tr>
<tr>
<td>UART</td>
<td>SPI</td>
</tr>
<tr>
<td>GPIO</td>
<td>PC</td>
</tr>
<tr>
<td>Radio</td>
<td></td>
</tr>
</tbody>
</table>

*planned
Supporting Driver Frameworks & HALs

<table>
<thead>
<tr>
<th>Hardware Platform</th>
<th>Device Drivers</th>
<th>Applications</th>
<th>3rd Party Libraries</th>
</tr>
</thead>
<tbody>
<tr>
<td>UART</td>
<td>Microkernel</td>
<td>Applications</td>
<td>C APIs</td>
</tr>
<tr>
<td>SPI</td>
<td>Nanokernel</td>
<td>CoAP, MQTT*, HTTP*, LWM2M*</td>
<td>CoAP, MQTT*, HTTP*, LWM2M*</td>
</tr>
<tr>
<td>GPIO</td>
<td></td>
<td>DTLS, TLS, ..</td>
<td>3rd Party Libraries</td>
</tr>
<tr>
<td>I2C</td>
<td></td>
<td>IPv4*, IPv6</td>
<td>IPv4*, IPv6</td>
</tr>
<tr>
<td>Radios</td>
<td></td>
<td>6LowPAN</td>
<td>6LowPAN</td>
</tr>
</tbody>
</table>

**Zephyr Native Drivers**
- Intel® Quark™ Microcontroller Software Interface (QMSI)
- ARM® Cortex® Microcontroller Software Interface Standard (CMSIS)
- Synopsys Open Software Platform (embARC)

* planned
# Zephyr Connectivity: Current Status

<table>
<thead>
<tr>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoAP</td>
</tr>
<tr>
<td>DTLS</td>
</tr>
<tr>
<td>UDP</td>
</tr>
<tr>
<td>IPv6</td>
</tr>
<tr>
<td>6LoWPAN</td>
</tr>
</tbody>
</table>

## Bluetooth® Smart API

- Intel® Curie™ Module Bluetooth Driver
- HCI host stack
  - 3-wire UART HCI driver
  - 5-wire UART HCI driver
- 802.15.4 MAC (cc2520)
- 802.3 (Intel® Galileo 2 Ethernet transceiver)
Integrated Security

- Standardized building block and robust communication stacks
- Cryptographic library based on TinyCrypt2
- Static and single binary applications, Single address space, No loadable modules
- Planned security features:
  - Device Management and Updates
  - APIs to support vendor specific Crypto implementations (software/hardware)
  - Secure Key Storage
Agenda

- Zephyr Project Overview
- Architecture Overview
- The Zephyr Kernel
- Getting Involved
Zephyr Technical Overview (Video)
Zephyr Kernel – Key Features

- Multi-threading services, including both priority-based, non-preemptive fibers and priority-based, preemptive tasks (with optional round robin time-slicing).
- Interrupt services, including both compile-time and run-time registration of interrupt handlers, which can be written in C or assembly language.
- Inter-thread synchronization services, including binary semaphores, counting semaphores, and mutex semaphores.
- Inter-thread data passing services, including basic message queues, enhanced message queues, and byte streams.
- Memory allocation services, including dynamic allocation and freeing of fixed-size or variable-size memory blocks.
Zephyr Kernel – Key Features

- Power management services, including tick-less idle and an advanced idling infrastructure.
- Highly configurable, allowing an application to incorporate only the capabilities it needs, and to specify their quantity and size.
- Zephyr requires all system resources to be defined at compile-time to reduce code size and increase performance.
- Provides minimal run-time error checking to reduce code size and increase performance. An optional error checking infrastructure is provided that can assist in debugging during application development.
- Library based RTOS (“kernel-less”)
Library-Based RTOS ("kernel-less")

- One single executable which is executed in one single address space
- No loader is required to dynamically load applications at run-time
  - Minimizing the operating system code
  - System calls are implemented as function calls
  - No context switches are required when calling an operating system call
- Lack of security through hardware memory separation
  - Application and operating system calls are implemented as thread in the same address space
  - Bugs in one part of the system can affect the whole system

Often more efficient and less time consuming as a full context switch with address space changes.

On small microcontrollers on which only one application is executed this disadvantage is acceptable.
Zephyr Nanokernel Overview

- A high-performance, multi-threaded execution environment with a basic set of kernel features
- Ideal for systems with sparse memory (the kernel itself requires as little as 2 KB!) or only simple multi-threading requirements (such as a set of interrupt handlers and a single background task)
- Examples of such systems include:
  - embedded sensor hubs
  - environmental sensors
  - simple LED wearables
  - store inventory tags
Nanokernel Scheduling and Objects

Kernel Services:
- FIFO
- LIFO
- Stack
- Mutex
- Timer
- Semaphore

Background Task
Server
Fiber
ISR
ISR
ISR

Kernel Services
Nanokernel Scheduling and Objects

- Cooperatively scheduled
- Run until they yield or call a blocking API
  - Marked as not runnable
  - Next highest priority fiber is then run
- Typically used for device drivers and performance-critical work
Zephyr Microkernel Overview

- Supplements the capabilities of the nanokernel to provide a richer set of kernel features
- Suitable for systems with
  - heftier memory (50 to 900 KB)
  - multiple communication devices (like Wi-Fi and Bluetooth® Low Energy)
  - and multiple data processing tasks
- Examples of such systems include:
  - Fitness wearables
  - Smart watches
  - IoT wireless gateways
Microkernel Scheduling and Objects

Kernel Services

- Memory Maps
- Memory Pools
- Event
- Pipe
- Mailbox
- FIFO
- LIFO
- Stack
- Mutex
- Timer
- Semaphore
Microkernel Scheduling and Objects

- A task is scheduled when no fibers are runnable
- Tasks are preemptible
- Highest priority task runs first
- Round-robin time-slicing between tasks of equal priority
- Used for data processing
Agenda

- Zephyr Project Overview
- Architecture Overview
- The Zephyr Kernel
- Getting Involved
Best-of-Breed Tools

- Kbuild - The build system of the Linux kernel
- Kconfig - The configuration system of the Linux kernel
- Rich and Powerful SDK developed specifically for Zephyr and powered by the Yocto project:
  - 5 different cross-compilers for all supported architectures and platforms
  - Support for baremetal c library, based on newlib
  - Host tools needed for debugging and downloading images (flashing) into target platforms
- Builds natively on Linux*, MacOS* and Microsoft* Windows
- Building on all various operating systems using Docker containers
Supported Platforms

<table>
<thead>
<tr>
<th>Platform</th>
<th>Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arduino* 101</td>
<td><img src="image1" alt="Arduino 101" /></td>
</tr>
<tr>
<td>Intel® Quark™ D2000 CRB</td>
<td><img src="image2" alt="Intel Quark D2000 CRB" /></td>
</tr>
<tr>
<td>2nd Generation Intel® Galileo</td>
<td><img src="image3" alt="2nd Generation Intel Galileo" /></td>
</tr>
<tr>
<td>FRDM-K64F</td>
<td><img src="image4" alt="FRDM-K64F" /></td>
</tr>
<tr>
<td>Arduino Due</td>
<td><img src="image5" alt="Arduino Due" /></td>
</tr>
<tr>
<td>ARC EM Starter Kit</td>
<td><img src="image6" alt="ARC EM Starter Kit" /></td>
</tr>
<tr>
<td>ST Nucleo F103RB</td>
<td><img src="image7" alt="ST Nucleo F103RB" /></td>
</tr>
</tbody>
</table>

► More platforms and boards to follow ◀
Summary and Next Steps

- Established code-base with strong community and industry support
- Focused and committed to support industry standards
- Innovative and forward looking by design
- Open to all
- A true open-source project, ready for your ideas, feedback and contributions
Participate!

- Impact architecture
- Direction
- Marketing / Advocacy
- Decision making

Examine the code and join!
Visit Us!
Booth #108
Zephyr™ Project Demos from our member companies

www.zephyrproject.org

<table>
<thead>
<tr>
<th>Other Zephyr™ Project Sessions @ ELC/OpenIoT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zephyr™ Project Security</td>
</tr>
<tr>
<td>Zephyr™ Project Overview</td>
</tr>
<tr>
<td>Power Management in Zephyr™ RTOS</td>
</tr>
</tbody>
</table>