The friendly operating system for the IoT!

Oliver «Oleg» Hahm
www.riot-os.org
oleg@riot-os.org
Our vision of the IoT
Wishlist for an IoT operating system
RIOT specs
Zoom on connectivity
Zoom on portability
RIOT as a platform for experiments
Join the RIOT
The Big Picture: a Giant Collision

Internet

Wireless

INTERNET OF THINGS

Cheap, tiny Hardware
Our Vision of the IoT

• A new world of interconnected hardware
• A new world at the application layer
• A new world in terms of user experience

➡️ Physical Computing
i.e. our interface to the Internet will no longer be predominantly a screen, a keyboard and/or a mouse
The Internet

- Internet
- > 4GB
- > 4GB
- ~ 2 GB
- 1-2 GB
- 512 MB
The Internet of Things

IoT = programmable world
IoT: From the Hardware Perspective

• The IoT is already here
  - Tiny, cheap & exciting new devices pop up daily
  - Mostly equipped with Atmel AVR, TI MSP430, or increasing numbers of ARM Cortex-M MCUs
  - Typically running with a CPU frequency < 100MHz and less than 100 kB RAM

Arduino Uno board
8 bit Atmel AVR

TI eZ430 Chronos watch
16 bit MSP 430 sub-GHz radio

SAM R21 Xplained Pro
32 bit ARM Cortex-M0+
2.4 GHz radio

Smart Dust
But: No IoT Until...

- ... a software big-bang happens
  - Similar to mobile phone industry since 2007 with iOS and Android dominance
  - Must have: de facto standard OS, providing consistent API & SDK across-hardware platforms
IoT: The Operating System

Question

IoT = programmable world
RIOT : The Friendly OS for the IoT

IoT = programmable world
• Our vision of the IoT
• Wishlist for an IoT operating system
• RIOT specs
• Zoom on connectivity
• Zoom on portability
• RIOT as a platform for experiments
• Join the RIOT
Wishlist for an IoT Operating System

An operating system for the IoT should:

- Support **heterogeneous hardware**
- Have a **low memory footprint**
- Provide **interoperability** with the Internet
- Make applications **portable**
Developing for the IoT

It should be easy to program, with support for:

- standard programming languages & techniques
- well known APIs (e.g. POSIX sockets)
- familiar debugging tools
- on-chip debugging capabilities
- comprehensive documentation
Developing for the IoT

It should be secure & independent:

- open source
- vendor-independent
- cloud-independent
- architecture-independent (8-bit, 16-bit, 32-bit)
AGENDA

• Our vision of the IoT
• Wishlist for an IoT operating system
• RIOT specs
• Zoom on connectivity
• Zoom on portability
• RIOT as a platform for experiments
• Join the RIOT
Meet RIOT

• Free, open source (LGPLv2.1) operating system for IoT
  – Write your code in ANSI-C or C++
  – Compliant with the most widely used POSIX features like pthreads and sockets
  – No IoT hardware needed for development
    • Run & debug RIOT as native process in Linux
RIOT Specs

• Microkernel architecture (for robustness)
  - The kernel itself uses ~1.5K RAM @ 32-bit
• Tickless scheduler (for energy efficiency)
• Deterministic O(1) scheduling (for real-time)
• Low latency interrupt handling (for reactivity)
• Modular structure (for adaptivity)
• Preemptive multi-threading & powerful IPC (for developer convenience)


AGENDA

• Our vision of the IoT
• Wishlist for an IoT operating system
• RIOT specs
• Zoom on connectivity
• Zoom on portability
• RIOT as a platform for experiments
• Join the RIOT
RIOT Supports Several Network Stacks

- BSD-like ports for: OpenWSN, LibCoAP, microcoap, relic, micro-ecc
- What's already there:
  - Application layer (CoAP, CBOR, UBJSON), Transport layer (UDP, TCP), Network layer (IPv6, 6LoWPAN, RPL, NHDP, AODVv2, CCN-lite), Link layer (IEEE 802.15.4 and 802.15.4e support)
  - Nativenet: network emulation & debugging
- On-going:
  - Bluetooth LE link layer support, Cooja and ns-3 simulator support, OLSRv2, & more...
Towards a Flexible Embedded Stack Design

Traditional Network Stack Architecture

RIOT Network Stack Redesign
AGENDA

• Our vision of the IoT
• Wishlist for an IoT operating system
• RIOT specs
• Zoom on connectivity
• **Zoom on portability**
• RIOT as a platform for experiments
• Join the RIOT
Code for RIOT is Portable

• Code your **application** once & run it everywhere
  - Mostly 32-bit platforms, but 8-bit and 16-bit platforms are supported, too
  - Independent from vendor-specific solutions

• Easy porting of RIOT to **new hardware**
  - Porting is a matter of hours, or days
  - e.g. support for new ARM Cortex-M boards is ‘trivial’
Portable Architecture

Hardware independent
- Application
  - Drivers
  - Sys
  - Core (Micro-Kernel)

Hardware dependent
- Board
  - Low-level Driver
- CPU
  - Low-level Driver Layer
  - RTC
  - ADC
  - Timer
  - GPIO
  - UART
  - SPI
  - I²C

Zoom on Board & CPU

Red: must have
Green: must have but shared by all ports with same architecture
Grey: optional for initial porting
AGENDA

• Our vision of the IoT
• Wishlist for an IoT operating system
• RIOT specs
• Zoom on connectivity
• Zoom on portability
• RIOT as a platform for experiments
• Join the RIOT
RIOT as a Platform for Experiments

- How to build the Internet of Plants with RIOT
  - Testing a distributed IoT application
  - Sensor monitoring & CoAP & 6LoWPAN

- Other use cases:
  - Run RIOT on an open testbed like IoT-LAB (tutorial available at https://www.iot-lab.info/tutorials/)
  - Emulation of virtual networks without changes to RIOT code
  - Connect real nodes to virtual topologies of RIOT instances
  - Experiments with new protocols & concepts for the IoT (e.g. content-centric networking)

- Low learning curve => RIOT as a teaching platform

Attend our tutorial tomorrow!
AGENDA

• Our vision of the IoT
• Wishlist for an IoT operating system
• RIOT specs
• Zoom on connectivity
• Zoom on portability
• RIOT as a platform for experiments
• Join the RIOT
In a Nutshell: RIOT is Accessible

• The goal is to be the fastest coding platform:
  - Code your IoT app or your IoT protocol in one afternoon

• Designed to be interoperable:
  - Standard APIs & standard network protocols
  - Contiki could run as a RIOT thread (but not the reverse ;)
  - RIOT can run as a Linux process

• Designed to be a modular solution:
  - From kernel-only to full stack including hardware support, network stacks, schedulers & your favorite API (POSIX, Arduino coming soon?)
RIOT Origins

History

• **2008 – Project roots:** The kernel was started as part of a research project

• **2010 – Towards the IoT:** Implementation of 6LoWPAN and RPL was initiated

• **2013 – RIOT goes public:** Branding of RIOT started, source code moved to Github
RIOT: Code evolution

»RIOT is one of the largest open-source teams in the world«
www.openhub.net/p/RIOT-OS, Jan. 2015
Join the RIOT

• Open source community
• ~ 250 forks on GitHub
  https://github.com/RIOT-OS/RIOT
• ~ 260 people on the developer mailing list: devel@riot-os.org
• Developers from all around the world
• Mentoring organisation for Google Summer of Code 2015
• Support & discussions on IRC:
  irc.freenode.org #riot-os
• ~ 750 followers on Twitter
Join our tutorial tomorrow!
Please take a look at
http://watr.li/workshop.html