Algorithmic Foundations of Sensor Networks
Sotiris Nikoletseas - Professor

ContikiOS Introduction
6/4/2020
Sensor nodes

- Low power
- Low computational resources
- Collection of sensors
- Connectivity
Examples of sensor nodes

• Sensortag CC2650

• TelosB mote

• List of sensing nodes: https://en.wikipedia.org/wiki/List_of_wireless_sensor_nodes
TelosB Skymote

- Texas Instruments MSP430 Microcontroller 8MHz
- Low power
- 10 kB RAM
- Integrated Temperature, Light, Humidity sensor
- Integrated Onboard Antenna, support of IEEE 802.15.4 protocol of wireless communication for low power devices, operating at 2.4GHz
- USB programmable
- Energy-efficient management of node components (radio, sensors, actuators)
Contiki OS

- Contiki OS is an operating system for resource-constrained devices in the Internet of Things.
- Started at 2003 by Adam Dunkels who works for the Swedish Institute of Computer Science.
- Implemented stack supports various protocols for connection and application layers (IPv6 support, 6LoWPAN, RPL, CoAP etc.)
- Support for many constrained devices.
- Large community of developers and users (Atmel, Cisco, SAP)
- Open source
- Written in C
Contiki-NG

• Support for modern IOT platforms and tools.
• Updated guides and tutorials.
• Rebuilted configuration system
• New data logging and shell systems
• MCU-based ARM support
Programming in Contiki

1. Create a project folder
2. Write Contiki program
3. Configure and Compile
4. Flash to WSN Mote
5. Test
Task scheduling

- Process
- Timers
- Threading
- Task scheduling

- [Documentation: Multitasking and scheduling](#)
Hello-world.c

in ContikiOS
Architecture - timer libraries

• Clock
• Timer - functions for setting, resetting, and restarting timers.
• Stimer - similar to the timer library, but uses time values in seconds.
• Etimer - event timer library
• Ctimer - Provides a function callback that will be called when timer expiration occurs.
• Rtimer - scheduling and execution for real-time tasks.
Energy monitoring

• Useful to estimate what affects a node energy consumption
• Estimation through *powertrace* and *energest* modules.
• Software based estimation by calculating which modes are on/off.
• By knowing the estimated power consumption of each component of the running device, it is possible to estimate the energy consumption.

• *Zolertia Z1 energy usage simulation with Cooja simulator*
Networking in Contiki

- IEEE 802.15.4
- IPv6 addressing
- MAC drivers - CSMA
- Radio Duty Cycling protocols
- RPL - Routing Protocol for Low Power and Lossy Networks
Radio Duty Cycling - I

- Radio module is the most energy hungry module of a sensing node.
  - Solution - Turn it on / off to save energy.
- RDC protocols provide mechanisms for time rendez-vous of communicating nodes.
  - Low Power Listening
  - Low Power Probing
Radio Duty Cycling - II

- Contiki MAC protocols
  - Low-Power-Listening
    - ContikiMAC
    - X-MAC (first low-power listening protocol)
  - Low-Power-Probing
    - Contiki LPP
Evolution of RDC - TSCH

- TSCH - Time Slotted Channel Hopping
  - Connection rendez-vous not only in time but also in frequency.
  - Applied into the IEEE 802.15.4 in 2015
  - Offers better reliability
6TiSCH - TSCH in Contiki-ng

• Synchronization in large (340 nodes) networks is possible at high accuracy (97% of the time under 160 us) for a low cost (duty cycle of 0.3%)

• TSCH, when running dedicated slots, outperforms LPL in all key metrics: reliability, latency, duty cycle.

• At a micro-level, a TSCH and LPL spend about the same amount of energy for receptions, but TSCH has an edge (factor 3) on transmissions.

Routing in Sensor Network - RPL

- RPL - Routing Protocol for Low Power and Lossy Networks (RFC 6550)
- Principle: All nodes form a Destination Oriented Directed Acyclic Graph (DODAG), where the sink node is the root of the DAG.
- Graph is implemented with the use of 3 types of messages:
  - DIO (broadcast/ multicast), DIS, DAO (unicast)
- Supports 3 directions of traffic:
  - Upward: from any node toward a root
  - Downward: from the root to any node
  - Any-to-any: flows among arbitrary pairs of nodes in the DODAG graph
DODAG Formation Example

1. A multicasts DIOs that it’s member of DODAG ID itself with Rank 0.
2. B, C, D, E hear and determine that their rank (distance) is 1, 1, 3, 4, respectively from A
3. B, C, D, E send DAOs to A.
4. A accepts all
5. B and C multicast DIOs
6. D hears those and determines that its distance from B and C is 1, 2
7. E hears both B, C and determines that its distance from B and C is 2, 1
8. D sends a DAO to B
   E sends a DAO to C
9. B sends a DAO-Ack to D
   C sends a DAO-Ack to E

Washington University in St. Louis

http://www.cse.wustl.edu/~jain/cse570-13/

©2013 Raj Jain
Cooja - Simulator for Contiki OS

• Useful for rapid prototyping
• Allows quick deployment of large scale experiments.
• Implemented in JAVA
• Core features
  • Code compilation for Contiki-enabled platforms
  • Grid for motes deployment.
  • Radio Medium simulation
  • Simulation timeline
Nodes grid

Printf of nodes will appear here.

Event tracking of simulation.
Simulation script editor

Buffer view

Trace of Radio usage
Cooja simulation

• Getting started:
  • Cooja example from contiki-ng wiki
  • Autonomous Networks Research Group guide on Contiki
Questions?

Thank you!