CS 425 / ECE 428 Distributed Systems Fall 2020

Indranil Gupta (Indy)

Lecture 26 B: Sensors and Their Networks

Jokes for this Topic

- I went to a restaurant and told the waiter I was watching my weight, but still wanted a little dessert. He pointed to the menu and recommended the Raspberry Pi.

Everything's Getting Smaller

- Smallest state-of-the-art transistor today is made of a single Gold atom
  - Still in research, not yet in industry.
- Pentium P4 contains 42 M transistors
- Gold atomic weight is 196 ~ 200.
- 1 g of Au contains 3 X 10^{21} atoms => 7.5 X 10^{18} P4 processors from a gram of Au => 1 billion P4's per person
- CPU speedup ~ $\sqrt{}$(# transistors on die)

Sensors Have Been Around for Centuries

- Coal mines have always had CO/CO2 sensors: “canary in a coal mine”
- Industry has used sensors for a long time, e.g., in assembly line

Today…

- Excessive Information
  - Environmentalists collecting data on an island
  - Army needs to know about enemy troop deployments
  - Humans in society face information overload
- Sensor Networking technology can help filter and process this information

Trends

Growth of any technology requires

I. Hardware

II. Operating Systems and Protocols

III. Killer applications

   – Military and Civilian

Sensor Nodes

- Motivating factors for emergence: applications, Moore’s Law (or variants), wireless comm., MEMS (micro electro mechanical sensors)
- Canonical *Sensor Node* contains

1. Sensor(s) to convert a different energy form to an electrical impulse e.g., to measure temperature
2. Microprocessor
3. Communications link e.g., wireless
4. Power source e.g., battery

Sensor Motes

- Size: small
  - MICA motes: Few inches
  - MicaDot: Few centimeters
  - Intel Motes: Few centimeters
  - Even smaller: Golem Dust=11.7 cu. mm
- Everything on one chip: micro-everything
  - processor, transceiver, battery, sensors, memory, bus
  - MICA: 4 MHz, 40 Kbps, 4 KB SRAM / 512 KB Serial Flash, lasts 7 days at full blast on 2 x AA batteries

Types of Sensors

- Micro-sensors (MEMS, Materials, Circuits)
  - acceleration, vibration, sound, gyroscope, tilt, magnetic, motion, pressure, temp, light, moisture, humidity, barometric
- Chemical
  - CO, CO2, radon
- Biological
  - pathogen detectors
- [In some cases, actuators too (mirrors, motors, smart surfaces, micro-robots)]

I2C Bus

- Developed By Philips
- Inter-IC connect
  - e.g., connect sensor to microprocessor
- Simple features
  - Has only 2 wires
  - Bi-directional
  - serial data (SDA) and serial clock (SCL) bus
- Up to 3.4 Mbps

Transmission Medium

- Spec, MICA: Radio Frequency (RF)
  - Broadcast medium, routing is “store and forward”, links are bidirectional
- Smart Dust : smaller size but RF needs high frequency => higher power consumption

Optical transmission: simpler hardware, lower power
  - Directional antennas only, broadcast costly
  - Line of sight required
  - Switching links costly : mechanical antenna movements
  - Passive transmission (reflectors) => “wormhole” routing
  - Unidirectional links

Summary: Sensor Node

- Small Size : few mm to a few inches
- Limited processing and communication
  - MhZ clock, MB flash, KB RAM, 100's Kbps (wireless) bandwidth
- Limited power (MICA: 7-10 days at full blast)
- Failure prone nodes and links (due to deployment, fab, wireless medium, etc.)

- But easy to manufacture and deploy in large numbers
- *Need to offset this with scalable and fault-tolerant OS's and protocols*

Sensor Node Operating System

Issues

- Size of code and run-time memory footprint
  - Embedded System OS’s inapplicable: need hundreds of KB ROM
- Workload characteristics
  - Continuous ? Bursty ?
- Application diversity
  - Want to reuse sensor nodes
- Tasks and processes
  - Scheduling
  - Hard and soft real-time
- Power consumption
- Communication

TinyOS for Sensor Nodes

Developed at Berkeley (2000’s), then @Crossbow Inc.

–Bursty dataflow-driven computations

–Multiple data streams => concurrency-intensive

–Real-time computations (hard and soft)

–Power conservation

–Size

–Accommodate diverse set of applications

□ TinyOS:

- Event-driven execution (*reactive* mote)
- Modular structure (components) and clean interfaces

Programming TinyOS Motes

- Use a variant of C called NesC
- NesC defines *components*
- A component is either
  - A *module* specifying a set of methods and internal storage (~like a Java static class)

    A module corresponds to either a hardware element on the chip (e.g., the clock or the LED), or to a user-defined software module

    Modules implement and use *interfaces*
  - Or a *configuration*, a set of other components *wired* together by specifying the unimplemented methods
- A complete NesC application then consists of one top level configuration

TinyOS Components

- Component invocation is event driven, arising from hardware events
- Static allocation only avoids run-time overhead
- Scheduling: dynamic, hard (or soft) real-time
- Explicit interfaces accommodate different applications

Deploying Your Application

(applies to MICA Mote)

- On your PC
  - Write NesC program
  - Compile to an executable for the mote
  - (Simulate and Debug)
  - Plug the mote into the port through a connector board
  - Install the program
- On the mote
  - Turn the mote on, and it’s already running your application

Energy Savings

- Power saving modes:
  - MICA: active, idle, sleep
- Tremendous variance in energy supply and demand
  - Sources: batteries, solar, vibration, AC
  - Requirements: long term deployment v. short term deployment, bandwidth intensiveness
  - 1 year on 2xAA batteries => 200 uA average current

Fallout

- TinyOS is small: Software Footprint = 3.4 KB
  - Can’t load a lot of data
- Power saving modes:
  - MICA: active, idle, sleep
- Radio Transmit is the most expensive (12 mA)
  - CPU Active: 4.6 mA
  - => Better compute that transmit
- => Lead to in-network aggregation approaches
  - Build trees among sensor nodes, base station at root of tree
  - Internal nodes receive values from children, calculate summaries (e.g., averages) and transmit these
  - More power-efficient than transmitting raw values or communicating directly with base station

Fallout (2)

- Correct direction for future technology
  - Today’s Growth rates: data > storage > CPU > communication > batteries
- Due to hostile environments (battlefields, environmental observation) and cheap fabrication
  - High failure rates in sensor nodes
  - Need sensor networks to be
    - Self-organizing
    - Self-managing
    - Self-healing
    - Scalable: Number of messages as function of number of nodes
- Broader (but related direction)
  - ASICs: Application-Specific Integrated Chips
  - FPGAs: Field Programmable Gate Arrays
  - Faster because move more action into hardware!

Summary

- Sensor nodes are cheap and battery-limited
- Deploy them in inhospitable terrains =>
  - Need to conserve power
  - Be smart about design of OS and distributed protocols
- TinyOS design
- Distributed Protocol Challenges

Some Topics for you To Look up

- Raspberry PI
  - Cheap computer, programmable
- Arduino
- Home automation systems: Nest, AMX, Homelogic, Honeywell, etc.
  - Power concerns smaller (since connected to power), but key security and accuracy concerns
- Network such devices together
  - Often called "Internet of Things"
  - Also called "cyberphysical systems"
- Cars today are networks of sensors
- Combination of humans and machines often called "Cyberphysical systems"
  - Operation theater (in hospitals) are becoming networks of sensors

Announcements

- HW4, MP4 due soon!
- Have a good Thanksgiving Break!