Programming Embedded Sensor Networks

Feng Zhao
PARC (formerly known as Xerox PARC)
www.parc.com/zhao
Acknowledgement

- PARC CoSense team:
  - Patrick Cheung, Maurice Chu, Leo Guibas, Qingfeng Huang, Jie Liu, Julia Liu, Jim Reich
  - Student Interns: Jaewon Shin, Judy Liebman, Soham Mazumdar, Dragan Petrovic

- External collaborations:
  - Diffusion: Deborah Estrin, John Heidemann, Fabio Silva (UCLA/USC/ISI)
  - DOA Estimation: Kung Yao (UCLA)
  - TinyGALS: Elaine Cheong (Berkeley)

- Funded in part by the DARPA SensIT Program under contract F30602-00-C-0175
PARC Smart Matter Research

- Smart Matter Research since 1994
- Collaborative Sensing since 1997

Sensor networks
- Collaborative processing
- Embedded software design
- Scalable Information architecture

Actuator networks
- Modular reconfiguration
- Constraint based control
- Adaptive optimization

MEMS devices
- MEMS signal processing
- Energy harvesting
- Large-area sensor/actuator arrays
Smart, Networked Sensors

- Of 9.6 billion uP to be shipped in 2005, 98% will be embedded processors!
- Intel plans to put a radio on every uP

- Riding on Moore’s law, smart sensors get
  
<table>
<thead>
<tr>
<th>More powerful</th>
<th>Easy to use</th>
<th>Inexpensive &amp; simple</th>
<th>Super-cheap &amp; tiny</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensoria WINSNG 2.0</td>
<td>HP iPAQ w/802.11</td>
<td>Crossbow MICA mote</td>
<td>Smart Dust (in progress)</td>
</tr>
<tr>
<td>CPU: 300 MIPS</td>
<td>CPU: 240 MIPS</td>
<td>4 MIPS CPU (integer only)</td>
<td>CPU, Memory: TBD</td>
</tr>
<tr>
<td>1.1 GFLOP FPU</td>
<td>32MB Flash</td>
<td>8KB Flash</td>
<td>(LESS!)</td>
</tr>
<tr>
<td>32MB RAM</td>
<td>64MB RAM</td>
<td>512B RAM</td>
<td>Sensors: integrated</td>
</tr>
<tr>
<td>Sensors: external</td>
<td>Both integrated and off-board sensors</td>
<td>on board stack</td>
<td></td>
</tr>
</tbody>
</table>

Source: Gartner Dataquest, 2002
Embedded software are difficult to write

Hardware challenges
• Limited processing capabilities
• Limited power resources
• Limited memory
• Limited bandwidth

Small programs on small devices
Sample Sensor Hardware: Berkeley motes

- **CPU:**
  - 8-bit, 4 MHz Atmel processor
  - No floating-point arithmetic support

- **Radio:**
  - 916 MHz RFM @10Kbps
  - Distance 30-100ft
  - Adjustable strength for RF transmission & reception

- **Storage:**
  - 8 KB instruction flash
  - 512 bytes data RAM
  - 512 bytes EEPROM (on processor)

- **OS:**
  - TinyOS, event driven (3.5KB code space)

- **Sensors:**
  - 10-bit ADC mux’d over 7 analog input channels
  - Sensing: light, sound, temperature, acceleration, magnetic field, pressure, humidity, RF signal strength
Power Breakdown...

<table>
<thead>
<tr>
<th></th>
<th>Active</th>
<th>Idle</th>
<th>Sleep</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>5 mA</td>
<td>2 mA</td>
<td>5 µA</td>
</tr>
<tr>
<td>Radio</td>
<td>7 mA (TX)</td>
<td>4.5 mA (RX)</td>
<td>5 µA</td>
</tr>
<tr>
<td>EE-Prom</td>
<td>3 mA</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>LED’s</td>
<td>4 mA</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Photo Diode</td>
<td>200 µA</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Temperature</td>
<td>200 µA</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Rene motes data, Jason Hill

Communication/computation ratio:

• Rene motes:
  • Comm: (7mA*3V/10e3)*8=16.8µJ per 8bit
  • Comp: 5mA*3V/4e6=3.8 nJ per instruction
  • Ratio: 4,400 instruction/hop

• Sensoria nodes:
  • Comm: (100mW/56e3)*32=58µJ per 32bit
  • Comp: 750mW/1.1e9=0.7nJ per instruction
  • Ratio: 82,000 instruction/hop

This means

— Lithium Battery runs for 35 hours at peak load and years at minimum load, a three orders of magnitude difference!

Panasonic CR2354
560 mAh
Embedded software are difficult to write

Hardware challenges
- Limited processing capabilities
- Limited power resources
- Limited memory
- Limited bandwidth

Networking challenges
- Ad hoc/dynamic system formation
- Unreliable links, competing media access
- No universal routing protocols
- No central name and registry services
- Simultaneously application host and routers

Limited infrastructure support
An application of wireless sensor network: fire monitoring
Embedded software are difficult to write

Hardware challenges
- Limited processing capabilities
- Limited power resources
- Limited memory
- Limited bandwidth

Networking challenges
- Ad hoc/dynamic system formation
- Unreliable links, competing media access
- No universal routing protocols
- No central name and registry services
- Simultaneously application host and routers

Application challenges
- Dynamic collaboration among nodes
- Global property from local execution
- Competing events/tasks
- Real-time missions

Massively distributed multitasking
A central problem: define and manage collaborative sensor groups dynamically based on their relevance to the current task and available network resources.
Tracking Events

- 17 nodes (6 DOA, 11 amplitude)
- Scale: 1 square=5 ft.
- 0.5 sec update interval
- Packet loss significant
Rethinking the network infrastructure

In wireless ad hoc networks, networking is intimately coupled with sensing, interaction, and control needs and hence application semantics.

- Break down traditional barriers of OSI model
  - IDSQ considers both communication cost and information utility using algorithms and dynamic data evaluated at the application level
  - Implemented the node selection in network, via routing decision

- Data-centric and ad-hoc
  - Address nodes based on geography and capability, not by name
  - Peer-to-peer, mesh topology; dynamic and highly mobile connectivity

- Group management vital to scalability
  - Limit data propagation to sensors relevant to measurement at hand
Node-Level Programming Architecture

- Modularity
- Component reusability
- Correctness by construction
- Maintainability

TinyGALS: Globally Asynchronous Locally Synchronous
Synthesizing an application-specific OS
Before and After…

TinyOS

TinyGALS

8 method call interfaces that handles 3 threads of execution using pending locks extensively.

Every module is single threaded. Components are completely sequential.
Programming embedded sensor networks: a comparison

OS/Network-centric view
Spend more time designing component-level abstractions

Information-centric view
Spend more time designing application-level abstractions
The PIECES Approach

- Raise the abstraction level beyond individual nodes
  - Provide high-level primitives that act on application “states”
  - Shield programmers from handling low-level events

- Models of collaboration
  - Abstract out common patterns of collaborative processing
  - Mix and match communication protocols from library
  - Modularize software through well-structured interfaces
Examples of Collaboration Groups

- N-hop neighbor group
- Geographically constrained group
  - Defined by geographic extent
- Publish-subscribe group
  - Defined by producers and consumers of shared interests
- Acquaintance group
  - Roaming members keep persistent connectivity
State-Centric Programming

- **Principals:**
  - Create and maintain states of physical phenomena
  - are triggered by events or time to update these states
  - can be mobile to achieve optimal use of resources

- **Agents**
  - recruited by one principal to obtain information from other principals through collaboration groups.
A Demo: tracking multiple, interacting events
Co-design information and software architectures

Software Technologies
- scalable software architectures
- formal methods
- software reuse
- software synthesis
- networking
- compilers
- operating systems
- ...

Information Technologies
- scalable information architecture
- semantics/ontology
- abstraction
- uncertainty management
- attention
- adaptation
- learning
- ...

parc
Palo Alto Research Center
As untethered sensors, actuators, embedded processors become ubiquitous, we need new ways to program and organize them.

<table>
<thead>
<tr>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query interface/data management</td>
</tr>
<tr>
<td>Information fusion</td>
</tr>
<tr>
<td>Collaboration group management</td>
</tr>
<tr>
<td>Storage, time, location services</td>
</tr>
<tr>
<td>OS</td>
</tr>
<tr>
<td>Networking</td>
</tr>
<tr>
<td>processor</td>
</tr>
<tr>
<td>sensors</td>
</tr>
<tr>
<td>processor</td>
</tr>
<tr>
<td>sensors</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>processor</td>
</tr>
<tr>
<td>sensors</td>
</tr>
</tbody>
</table>
To Probe Further

- PARC Cosense Project: [www.parc.com/ecca](http://www.parc.com/ecca)
- Other centers: UCLA CENS, Intel Berkeley Lablet, CAL(IT)2, …
- Conferences:
  - IEEE/ACM IPSN04, Berkeley, April 2004
  - ACM Sensys03, Los Angeles, November 2003
  - ACM WSNA03, Sept 2003 (already taken place)
  - EWSN04, Berlin, Jan. 2004
- Journals:
  - New ACM Trans. Sensor Networks, to be launched in 2004
  - A dozen special issues
- Books:
  - Morgan Kaufmann textbook
  - CRC Press book