Part II: Sensor Node Platforms & Energy Issues

Mani Srivastava

Sensor Node H/W-S/W Platforms

Event detection

In-node processing

Wireless communication with neighboring nodes

Acoustic, seismic, image, magnetic, etc. interface

Energy efficiency is the crucial h/w and s/w design criterion
Overview of this Section

- Survey of sensor node platforms
- Sources of energy consumption
- Energy management techniques

Variety of Real-life Sensor Node Platforms

- RSC WINS & Hidra
- Sensoria WINS
- UCLA’s iBadge
- UCLA’s Medusa MK-II
- Berkeley’s Motes
- Berkeley Piconodes
- MIT’s µAMPs
- And many more...

- Different points in (cost, power, functionality, form factor) space
Rockwell WINS & Hidra Nodes

- Consists of 2”x2” boards in a 3.5”x3.5”x3” enclosure
  - StrongARM 1100 processor @ 133 MHz
    - 4MB Flash, 1MB SRAM
- Various sensors
  - Seismic (geophone)
  - Acoustic
  - Magnetometer,
  - Accelerometer, temperature, pressure
- RF communications
  - Connexant’s RDSS9M Radio @ 100 kbps, 1-100 mW, 40 channels
- eCos RTOS
- Commercial version: Hidra
  - μC/OS-II
  - TDMA MAC with multihop routing
- http://wins.rsc.rockwell.com/

Sensoria WINS NG 2.0, sGate, and WINS Tactical Sensor

- WINS NG 2.0
  - Development platform used in DARPA SensIT
  - SH-4 processor @ 167 MHz
  - DSP with 4-channel 16-bit ADC
  - GPS
  - Imaging
  - Dual 2.4 GHz FH radios
  - Linux 2.4 + Sensoria APIs
  - Commercial version: sGate
- WINS Tactical Sensor Node
  - geo-location by acoustic ranging and angle
  - Time synchronization to 5 µs
  - Cooperative distributed event processing

Ref: based on material from Sensoria slides
Sensoria Node Hardware Architecture

Sensoria Node Software Architecture

Ref: based on material from Sensoria slides
Berkeley Motes

- Devices that incorporate communications, processing, sensors, and batteries into a small package
- Atmel microcontroller with sensors and a communication unit
  - RF transceiver, laser module, or a corner cube reflector
  - temperature, light, humidity, pressure, 3 axis magnetometers, 3 axis accelerometers
- TinyOS

The Mote Family

<table>
<thead>
<tr>
<th>Mote Type</th>
<th>Tec</th>
<th>renx2</th>
<th>renx2</th>
<th>dot</th>
<th>sica</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>9/99</td>
<td>10/00</td>
<td>6/01</td>
<td>8/01</td>
<td>2/02</td>
</tr>
<tr>
<td>Microcontroller</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>AT50LS8635</td>
<td>ATMega163</td>
<td>ATMega163</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prog. mem. (KB)</td>
<td>8</td>
<td>16</td>
<td>128</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAM (KB)</td>
<td>0.5</td>
<td>4</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nonvolatile storage</td>
<td>24LC256</td>
<td>AT45DB041B</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Chip</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Connection type</td>
<td>I2C</td>
<td>I2C</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Size (KB)</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default Power source</td>
<td>Li</td>
<td>Alk</td>
<td>Li</td>
<td>Alk</td>
<td>Alk</td>
</tr>
<tr>
<td>Size</td>
<td>CR2450</td>
<td>2xAA</td>
<td>CR2032</td>
<td>2xAA</td>
<td></td>
</tr>
<tr>
<td>Capacity (mAh)</td>
<td>575</td>
<td>2560</td>
<td>225</td>
<td>2560</td>
<td></td>
</tr>
<tr>
<td>Communication</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Radio</td>
<td>RFM TR10B</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rate (Kbps)</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10/40</td>
</tr>
<tr>
<td>Modulation type</td>
<td>OOK</td>
<td>OOK</td>
<td></td>
<td></td>
<td>OOK/ASK</td>
</tr>
</tbody>
</table>

Ref: from Levis & Culler, ASPLOS 2002
TinyOS

- System composed of concurrent FSM modules
  - Single execution context
- Component model
  - Frame (storage)
  - Commands & event handlers
  - Tasks (computation)
  - Command & Event interface
  - Easy migration across h/w - s/w boundary
- Two level scheduling structure
  - Preemptive scheduling of event handlers
  - Non-preemptive FIFO scheduling of tasks
- Compile time memory allocation
- NestC
- http://webs.cs.berkeley.edu

Bit_Arrival_Event_Handler
State: {bit_cnt}

Start

bit_cnt++

bit_cnt==8

Yes

Send Byte Event

bit_cnt = 0

Done

No

Ref: from Hill, Szewczyk et. al., ASPLOS 2000
UCLA iBadge

- Wearable Sensor Badge
  - acoustic in/out + DSP
  - temperature, pressure, humidity, magnetometer, accelerometer
  - ultrasound localization
  - orientation via magnetometer and accelerometer
  - bluetooth radio
- Sylph Middleware

Sylph Middleware

- Sensor Apps
- Speech Recognition Service
- Bayesian Fusion Service
- Storage Service
- Browsers
- Sensor Configuration Manager

Client application

Jini registry

Sylph middleware

Jini SOM

Proxy core

Microphone SM

iBadgeSM

Camera SM

Base station

Piconet

Smart toy

iBadge

Service discovery modules

Sensor modules

Network infrastructure

Sensing infrastructure

READ <data table>

[WHERE predicate clause]

[EVERY time interval]

[FOR duration]
UCLA Medusa MK-II Localizer Nodes

- 40MHz ARM THUMB
  - 1MB FLASH, 136KB RAM
  - 0.9MIPS/MHz 480MIPS/W (ATMega 242MIPS/W)
- RS-485 bus
  - Out of band data collection, formation of arrays
- 3 current monitors (Radio, Thumb, rest of the system)
- 540mAh Rechargeable Li-Ion battery

BWRC’s *PicoNode* TripWire Sensor Node

**Version 1: Light Powered**

**Version 2: Vibration Powered**

Ref: from Jan Rabaey, PAC/C Slides
**BWRC PicoNode (contd.)**

- Reactive inter- and intra-chip signaling
- Modules in power-down (low-leakage) mode by default
- Events at interface cause wake-up
- HW Modules selected to meet flexibility needs while optimizing energy efficiency (e.g. 8051 microcontroller)

Ref: from Jan Rabaey, PAC/C Slides

**Quick-and-dirty iPaq-based Sensor Node!**

- **WaveLan Card**
  - IEEE 802.11b Compliant
  - 11 Mbit/s Data Rate

- **HM2300 Magnetic Sensor**
  - uC Based with RS232
  - Range of +/- 2 Gauss
  - Adjustable Sampling Rate
  - X, Y, Z output
  - Device ID Management

- **iPAQ 3670**
  - Intel StrongARM
  - Power Management (normal, idle & sleep mode)
  - Programmable System Clock
  - IR, USB, Serial (RS232) transmission

- **Acoustic Sensor & Actuator**
  - Built-in microphone
  - Built-in speaker
Sensor Node Energy Roadmap
(DARPA PAC/C)

- Deployed (5W)
- PAC/C Baseline (.5W)
- (50 mW)
- (1mW)

Where does the energy go?

- Processing
  - excluding low-level processing for radio, sensors, actuators
- Radio
- Sensors
- Actuators
- Power supply
Processing

- Common sensor node processors:
  - Atmel AVR, Intel 8051, StrongARM, XScale, ARM Thumb, SH Risc
- Power consumption all over the map, e.g.
  - 16.5 mW for ATmega128L @ 4MHz
  - 75 mW for ARM Thumb @ 40 MHz
- But, don’t confuse low-power and energy-efficiency!
  - Example
    - 242 MIPS/W for ATmega128L @ 4MHz (4nJ/Instruction)
    - 480 MIPS/W for ARM Thumb @ 40 MHz (2.1 nJ/Instruction)
  - Other examples:
    - 0.2 nJ/Instruction for Cygnal C8051F300 @ 32KHz, 3.3V
    - 0.35 nJ/Instruction for IBM 405LP @ 152 MHz, 1.0V
    - 0.5 nJ/Instruction for Cygnal C8051F300 @ 25MHz, 3.3V
    - 0.8 nJ/Instruction for TMS320VC5510 @ 200 MHz, 1.5V
    - 1.1 nJ/Instruction for Xscale PXA250 @ 400 MHz, 1.3V
    - 1.3 nJ/Instruction for IBM 405LP @ 380 MHz, 1.8V
    - 1.9 nJ/Instruction for Xscale PXA250 @ 130 MHz, .85V (leakage!)
  - And, the above don’t even factor in operand size differences!
- However, need power management to actually exploit energy efficiency
  - Idle and sleep modes, variable voltage and frequency

Radio

- Energy per bit in radios is a strong function of desired communication performance and choice of modulation
  - Range and BER for given channel condition (noise, multipath and Doppler fading)
- Watch out: different people count energy differently
  - E.g.
    - Mote’s RFM radio is only a transceiver, and a lot of low-level processing takes place in the main CPU
    - While, typical 802.11b radios do everything up to MAC and link level encryption in the “radio”
- Transmit, receive, idle, and sleep modes
- Variable modulation, coding
- Currently around 150 nJ/bit for short ranges
- More later…
**Computation & Communication**

- Radios benefit less from technology improvements than processors.
- The relative impact of the communication subsystem on the system energy consumption will grow.

**Sensing**

- Several energy consumption sources
  - transducer
  - front-end processing and signal conditioning
    - analog, digital
  - ADC conversion
- Diversity of sensors: no general conclusions can be drawn
  - Low-power modalities
    - Temperature, light, accelerometer
  - Medium-power modalities
    - Acoustic, magnetic
  - High-power modalities
    - Image, video, beamforming
Actuation

- Emerging sensor platforms
  - Mounted on mobile robots
  - Antennas or sensors that can be actuated
- Energy trade-offs not yet studied
- Some thoughts:
  - Actuation often done with fuel, which has much higher energy density than batteries
    - E.g. anecdotal evidence that in some UAVs the flight time is longer than the up time of the wireless camera mounted on it
  - Actuation done during boot-up or once in a while may have significant payoffs
    - E.g. mechanically repositioning the antenna once may be better than paying higher communication energy cost for all subsequent packets
    - E.g. moving a few nodes may result in a more uniform distribution of node, and thus longer system lifetime

Power Analysis of RSC’s WINS Nodes

<table>
<thead>
<tr>
<th>MCU Mode</th>
<th>Sensor Mode</th>
<th>Radio Mode</th>
<th>Power (mW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active</td>
<td>On</td>
<td>Tx (Power: 36.3 mW)</td>
<td>1000.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tx (Power: 19.1 mW)</td>
<td>985.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tx (Power: 13.8 mW)</td>
<td>942.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tx (Power: 3.47 mW)</td>
<td>815.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tx (Power: 2.81 mW)</td>
<td>807.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tx (Power: 0.09 mW)</td>
<td>787.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tx (Power: 0.03 mW)</td>
<td>777.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tx (Power: 0.02 mW)</td>
<td>777.1</td>
</tr>
<tr>
<td>Active</td>
<td>On</td>
<td>Rx</td>
<td>754.6</td>
</tr>
<tr>
<td></td>
<td>On</td>
<td>Idle</td>
<td>727.5</td>
</tr>
<tr>
<td>Active</td>
<td>On</td>
<td>Sleep</td>
<td>415.3</td>
</tr>
<tr>
<td>Active</td>
<td>Removed</td>
<td>Removed</td>
<td>385.3</td>
</tr>
<tr>
<td>Sleep</td>
<td>On</td>
<td>Removed</td>
<td>64.0</td>
</tr>
<tr>
<td>Active</td>
<td>Removed</td>
<td>Removed</td>
<td>580.0</td>
</tr>
</tbody>
</table>

- Summary
- Processor
  - Active = 360 mW
    - doing repeated transmit/receive
  - Sleep = 41 mW
  - Off = 0.9 mW
- Sensor = 23 mW
- Processor : Tx = 1 : 2
- Processor : Rx = 1 : 1
- Total Tx : Rx = 4 : 3 at maximum range
  - comparable at lower Tx
Power Analysis of Mote-Like Node

<table>
<thead>
<tr>
<th>MCU Mode</th>
<th>Network Mode</th>
<th>Radio Mode</th>
<th>Mod. Adater</th>
<th>Data Rate</th>
<th>Power (mW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active</td>
<td>On</td>
<td>TzPower: 0.766 mW</td>
<td>OOK</td>
<td>2.4 kbps</td>
<td>24.88</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TzPower: 0.097 mW</td>
<td>OOK</td>
<td>2.4 kbps</td>
<td>39.24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TzPower: 0.766 mW</td>
<td>ASK</td>
<td>39.2 kbps</td>
<td>25.17</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TzPower: 0.097 mW</td>
<td>ASK</td>
<td>39.2 kbps</td>
<td>20.08</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TzPower: 0.766 mW</td>
<td>ASK</td>
<td>2.4 kbps</td>
<td>20.51</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TzPower: 0.097 mW</td>
<td>ASK</td>
<td>2.4 kbps</td>
<td>21.36</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TzPower: 0.766 mW</td>
<td>ASK</td>
<td>39.2 kbps</td>
<td>27.46</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TzPower: 0.097 mW</td>
<td>ASK</td>
<td>39.2 kbps</td>
<td>22.06</td>
</tr>
<tr>
<td>Active</td>
<td>On</td>
<td>Rx</td>
<td>Any</td>
<td>Any</td>
<td>22.30</td>
</tr>
<tr>
<td>Active</td>
<td>On</td>
<td>Idle</td>
<td>Any</td>
<td>Any</td>
<td>22.06</td>
</tr>
<tr>
<td>Active</td>
<td>Off</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>9.72</td>
</tr>
<tr>
<td>Idle</td>
<td>Off</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>5.04</td>
</tr>
<tr>
<td>Sleep</td>
<td>Off</td>
<td>Off</td>
<td>Any</td>
<td>Any</td>
<td>4.02</td>
</tr>
</tbody>
</table>

Some Observations

- Using low-power components and trading-off unnecessary performance for power savings can have orders of magnitude impact
- Node power consumption is strongly dependent on the operating mode
  - E.g. WINS consumes only 1/6-th the power when MCU is asleep as opposed to active
- At short ranges, the Rx power consumption > T power consumption
  - multihop relaying not necessarily desirable
- Idle radio consumes almost as much power as radio in Rx mode
  - Radio needs to be completely shut off to save power as in sensor networks idle time dominates
    - MAC protocols that do not “listen” a lot
- Processor power fairly significant (30-50%) share of overall power
- In WINS node, radio consumes 33 mW in “sleep” vs. “removed”
  - Argues for module level power shutdown
- Sensor transducer power negligible
  - Use sensors to provide wakeup signal for processor and radio
  - Not true for active sensors though...
### Energy Management Problem

- Actuation energy is the highest
  - Strategy: ultra-low-power “sentinel” nodes
    - Wake-up or command movement of mobile nodes
- Communication energy is the next important issue
  - Strategy: energy-aware data communication
    - Adapt the instantaneous performance to meet the timing and error rate constraints, while minimizing energy/bit
- Processor and sensor energy usually less important

<table>
<thead>
<tr>
<th></th>
<th>MICA mote Berkeley</th>
<th>WINS node RSC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Transmit</td>
<td>Transmit</td>
</tr>
<tr>
<td></td>
<td>720 nJ/bit</td>
<td>6600 nJ/bit</td>
</tr>
<tr>
<td></td>
<td>Receive</td>
<td>Receive</td>
</tr>
<tr>
<td></td>
<td>110 nJ/bit</td>
<td>3300 nJ/bit</td>
</tr>
<tr>
<td></td>
<td>Processor</td>
<td>Processor</td>
</tr>
<tr>
<td></td>
<td>4 nJ/op</td>
<td>1.6 nJ/op</td>
</tr>
<tr>
<td></td>
<td>~ 200 ops/bit</td>
<td>~ 6000 ops/bit</td>
</tr>
</tbody>
</table>

### Processor Energy Management

- Knobs
  - Shutdown
  - Dynamic scaling of frequency and supply voltage
  - More recent: dynamic scaling of frequency, supply voltage, and threshold voltage
- All of the above knobs incorporated into sensor node OS schedulers
  - e.g. PA-eCos by UCLA & UCI has Rate-monotonic Scheduler with shutdown and DVS
- Gains of 2x-4x typically, in CPU power with typical workloads
- Predictive approaches
  - Predict computation load and set voltage/frequency accordingly
  - Exploit the resiliency of sensor nets to packet and event losses
  - Now, losses due to computation noise
Radio Energy Management

- During operation, the required performance is often less than the peak performance the radio is designed for.
- How do we take advantage of this observation, in both the sender and the receiver?

Energy in Radio: the Deeper Story....

- Wireless communication subsystem consists of three components with substantially different characteristics.
- Their relative importance depends on the transmission range of the radio.
Examples

- The RF energy increases with transmission range
- The electronics energy for transmit and receive are typically comparable

Energy Consumption of the Sender

- Parameter of interest:
  - energy consumption per bit
  \[
  E_{bit} = \frac{P}{T_{bit}}
  \]
**Effect of Transmission Range**

- Long-range
- Short-range
- Medium-range

**Transmission time**

**Power Breakdowns and Trends**

- **Radiated power**: 63 mW (18 dBm)
- **Intersil PRISM II (Nokia C021 wireless LAN)**
- **Power amplifier**: 600 mW (~11% efficiency)
- **Analog electronics**: 240 mW
- **Digital electronics**: 170 mW

**Trends:**
- Move functionality from the analog to the digital electronics
- Digital electronics benefit most from technology improvements
- Borderline between ‘long’ and ‘short’-range moves towards shorter transmit distances
Radio Energy Management #1: Shutdown

- **Principle**
  - Operate at a fixed speed and power level
  - Shut down the radio after the transmission
  - No superfluous energy consumption
- **Gotcha**
  - When and how to wake up?
  - More later …

Radio Energy Management #2: Scaling along the Performance-Energy Curve

**Principle**
- Vary radio ‘control knobs’ such as modulation and error coding
- Trade off energy versus transmission time

Modulation scaling
- fewer bits per symbol

Code scaling
- more heavily coded
**When to Scale?**

- Scaling results in a convex curve with an energy minimum $E_{\text{min}}$.
- It only makes sense to slow down to transmission time $t^*$ corresponding to this energy minimum.

**Scaling vs. Shutdown**

- Use scaling while it reduces the energy.
- If more time is allowed, scale down to the minimum energy point and subsequently use shutdown.
Long-range System

- The shape of the curve depends on the relative importance of RF and electronics
- This is a function of the transmission range
- Long-range systems have an operational region where they benefit from scaling

Short-range Systems

- Short-range systems have an operational region where scaling is not beneficial
- Best strategy is to transmit as fast as possible and shut down
Sensor Node Radio Power Management Summary

Short-range links

- **Shutdown** based
- Turn off sender and receiver
- Topology management schemes exploit this
  e.g. Schurgers et. al. @ ACM MobiHoc ‘02

Long-range links

- **Scaling** based
- Slow down transmissions
- Energy-aware packet schedulers exploit this
  e.g. Raghunathan et. al. @ ACM ISLPED ‘02

Another Issue: Start-up Time

\[ E_{\text{radio}} = E_{\text{tx}} + E_{\text{td}} \]
\[ E_{\text{radio}} = \left[ P_{\text{tx}}(T_{\text{on}} + T_{\text{start}}) \right] + \left[ P_{\text{td}}(T_{\text{on}} + T_{\text{start}}) + P_{\text{out}}(T_{\text{on}}) \right] \]

Fixed Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P_{\text{tx}} )</td>
<td>81 mW</td>
</tr>
<tr>
<td>( P_{\text{td}} )</td>
<td>180 mW</td>
</tr>
<tr>
<td>( T_{\text{start}} )</td>
<td>466 ( \mu )s</td>
</tr>
</tbody>
</table>

Tunable Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P_{\text{out}} )</td>
<td></td>
</tr>
<tr>
<td>( T_{\text{on}} )</td>
<td></td>
</tr>
<tr>
<td>( E_{\text{FEC}} )</td>
<td></td>
</tr>
</tbody>
</table>

Ref: Shih et. al., Mobicom 2001
Wasted Energy

- Fixed cost of communication: startup time
  - High energy per bit for small packets

Sensor Node with Energy-efficient Packet Relaying [Tsiatsis01]

- Problem: sensor nodes often simply relays packets
  - e.g. > 2/3-rd pkts. in some sample tracking simulations
- Traditional: main CPU woken up, packets sent across bus
  - power and latency penalty
- One fix: radio with a packet processor handles the common case of relaying
  - packets redirected as low in the protocol stack as possible
- Challenge: how to do it so that every new routing protocol will not require a new radio firmware or chip redesign?
  - packet processor classifies and modifies packets according to application-defined rules
  - can also do ops such as combining of packets with redundant information

![Diagram showing Traditional and Energy-efficient Approaches](image_url)
Putting it All Together: Power-aware Sensor Node

PASTA Sensor Node Hardware Stack

Energy-aware RTOS, Protocols, & Middleware

Dynamic Voltage & Freq. Scaling

Scalable Sensor Processing

Freq., Power, Modulation, & Code Scaling

Coordinated Power Management

PA-APIs for Communication, Computation, & Sensing

Future Directions: Sensor-field Level Power Management

<table>
<thead>
<tr>
<th>TYPE</th>
<th>STATE</th>
<th>SENS</th>
<th>CPU</th>
<th>COMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tripwire</td>
<td>ON</td>
<td>OFF</td>
<td>STEM</td>
<td></td>
</tr>
<tr>
<td>Tracker</td>
<td>OFF</td>
<td>OFF</td>
<td>STEM</td>
<td></td>
</tr>
</tbody>
</table>

Two types of nodes
- Tripwire nodes that are always sense
  - Low-power presence sensing modalities such as seismic or magnetic
  - Tracker nodes that sense on-demand
    - Higher power modalities such as LOB

Approach
- Network self-configures so that gradients are established from Tripwire nodes to nearby Tracker nodes
- Radios are all managed via STEM
- Event causes nearby Tripwire nodes to trip
- Tripped Tripwire nodes collaboratively contact suitable Tracker nodes
  - Path established via STEM
  - Tracker nodes activate their sensors
- Range or AoA information from Tracker Nodes is fused (e.g. Kalman Filter) to get location
  - In-network processing
    - Centralized: where should the fusion center be?
    - Distributed: fusion tree
- Result of fusion sent to interested user nodes
- Set of active Tracker Nodes changes as target moves
  - Process similar to hand-off

Data
Wakeup
Line of Bearing (LOB)
Fusion center
Tools

- Sensor Network-level Simulation Tools
  - Ns-2 enhancements by ISI
  - Ns-2 based SensorSim/SensorViz by UCLA
  - C++-based LECSim by UCLA
  - PARSEC-based NESLsim by UCLA

- Node-level Simulation Tools
  - MILAN by USC for WINS and µAMPS
  - ToS-Sim for Motes

- Processor-level Simulation Tools
  - JoulesTrack by MIT
SensorViz

Trace Data from Experiments

Node Locations
Target Trajectories
Sensor Readings
User Trajectories
Query Traffic

Power Measurements

Power Models