Sensor Networks

David Andersen
Low-Power Computing
Carnegie Mellon University

Sensor Evolution

Integration of sensing, computation, and communication

- Low-power, wireless "motes" with tiny amount of CPU/memory
- Large federated networks for high-resolution sensing of environment

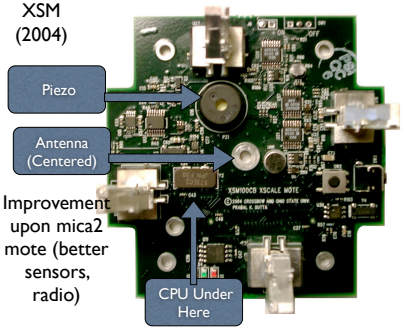
MICA (2002)

Speck (2003)

Telos (2004)

slide credit: Matt Welsh

XSM (2004)

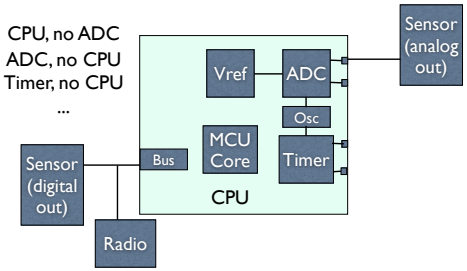


Improvement upon mica2 mote (better sensors, radio)

The Hardware

- Some common µ-processors
- Atmel ATMega128L
  - 128KB flash, 4K EEPROM, 4K SRAM
  - 0-8Mhz adjustable clock
  - 2.7 - 5.5V
  - Unlike desktop procs, *very* detailed power info available. :)

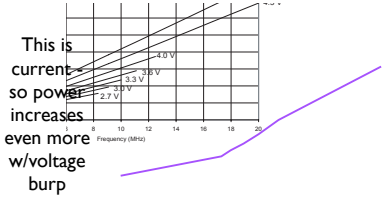
Typical Sensor Stuff



Sleepy Time

- Most modules on sensor board can be shut down (power supply gated)
- The CPU itself - can shut down ADC, internal voltages (10 µA, to give an idea of the power range), watchdog, i/o pins
  - Idle, Power-down (ext interrupts wake up)

Look familiar?



This is current so power increases even more w/voltage burp

8Mhz low-voltage: ~20mW
(paper measures 24)

ATMega 128L Sleeps

- Active 8Mhz: 20-24mW
- Idle 8Mhz: ~10mW
- Power-down, no watchdog: ~0.5µW
- Power-down, with w/d: ~30µW (paper)
  - 1Mhz watchdog - non-negligible.
  - Safety beats power in unattended sensor

Other option: TI

- MPS430 - 16bit - ex:
  - 16KB RAM, 128KB Flash
    - <= 18Mhz
    - 1.8V-3.6V, 5-8.5mW active @ 8Mhz; less if program fits in DRAM.
    - $5 or so. Cool toy.

Telos Mote

- CC2420 radio (2.4 Ghz, 802.15.4)
- 250 kbps, 100 m range

Several thousand produced, used by 100s of research groups

Great platform for experimentation (though not particularly small)

- Easy to integrate new sensors & actuators
- 15-20 mA active (5-6 days on 2 AAs)
- 5 µA sleeping (40+ years, but limited by shelf life of battery!)

Power

	XSM	Telos
Standby	30µW	15µW
Idle	~10mW	~150µW*
Active No radio	24mW	5mW
Rx	55mW	55mW
Tx	72mW	50mW

* Higher in reality?

#s change, game is the same - but favors more compute/less radio.

Irony

- Moteiv (maker of Telos Motes)
- Acquired by Sentilla
- Sentilla now sells...
  - Datacenter energy consumption monitors and analysis software. :)
  - but no motes. :(TI, however, sells very cute little dev boards.

And *almost*

- 32 bit platforms almost becoming usable for sensor nets
  - iMote2 (Crossbow - mica folks)
  - Intel PXA271 - 256K SRAM, 32MB SDRAM (woah), 32MB flash
  - 31mA active @ 13Mhz no radio, 44mA tx/rx. (at sub 1V?) Niiice. High sleep current, but give it a few years.

But regardless...

- Reasonable-but-ambitious goal: 1 year, 1 AAA battery.
- 1500 mWh
- There are 8760 hours in a year.
- Avg draw: 171μW
- oof.

Application-Specific Sleeping

- *Must* sleep a lot.
- Being useful while sleeping a lot: application-specific schedules. *Wake only when needed.*
  - When is it needed?

Two apps

- "Classical" sensor nets:
  - Sample temp & humidity every 5 minutes
  - Send to base station via neighbors
- "Event" sensor nets:
  - Watch frequently, report seldom
  - Events possibly irregular, outside control

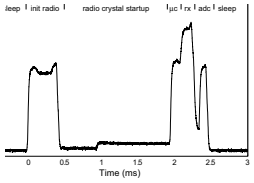
What draws power?

- The CPU
- The sensors themselves (they're physical devices...)
- The radio
  - Including relaying/collecting/broadcasting

B-MAC

- The LPL mode from the paper
- Wake on timer interrupt
- Startup: Wait for XO to stabilize
- Receive; sample signal energy
- Turn radio off, start analyzing signal strength

BMAC modes



MAC Design goals

- Low power
- Tiny implementation (4616 bytes in ROM, 277 bytes of RAM)
- But nasty to sender: If check channel every 100ms, then Tx preamble must be 100ms long.
- Assumption: *Very* infrequent Tx.
- No time sync as in BSD/802.11

"Classical" alternative

- Option 1: Batch the heck out of it;
- use LPL

- Option 2: Schedule a wake-up time for reporting
- Much more BSD-like.
- Which? Depends - how up-to-date do your measurements need to be?

Sensors...

- Are a bit of a PITA to program.
  - Mica2: 8-bit RISC-like system
  - Telos: 16-bit (HUGE improvement, but still...)
  - No memory protection, no conventional OS, processes, scheduler, etc. Not even what a rt-OS like VxWorks gives you

TinyOS

- popular OS platform for motes
- Fairly standard OS challenge:
  - Writing individual modules is OK
  - Making the system coherent is hard.
  - Provides a basic scheduler, interrupt support, etc., plus glue to link components

Paranoid Energy Mgmt

- Go back to that picture about timing in BMAC
  - Sample channel;
    tell ADC to take reading;
    immediately put radio to sleep again;
    while it's in that process, figure out if channel was busy
- And hey - maybe I also use the ADC for reading my light sensor... and... and...