Topic 12
TinyOS

Bhaskaran Raman,
Department of CSE, IIT Kanpur

http://www.cse.iitk.ac.in/users/braman/courses/wless-spring2007/
TinyOS

  - Section-2

- More related to **embedded computing** than wireless

- Necessary for **projects**
TinyOS Goals

• An "operating-system" for embedded sensor nodes
• Different requirements for such platforms
  – Should be designed for current & future hardware
  – Cater to a wide variety of applications
  – Limited resources: memory, power
  – Concurrency-intensive operation: data driven
TinyOS Design Overview (1 of 2)

• Modular framework:
  – A set of software components and interfaces
  – No strict definition of system/user boundary

• Issues addressed by this approach:
  – Adaptation to heterogeneous hardware
    • Reuse of software
  – Adaptation to different application requirements
    • Put together required software components
  – Memory resource constraints
    • Use only the required components
TinyOS Design Overview (2 of 2)

- Event-driven concurrency model:
  - Hardware *events* and software *tasks*

- Issues addressed by this approach:
  - Requirement for concurrency
    - Event-driven model is natural: no blocking or polling
  - Limited memory
    - Many concurrent tasks using just one stack
  - Power savings
    - No tasks ==> sleep
TinyOS Design

- **Interface**: a set of *commands* and *events*
  - Command: sub-routine to perform some action
  - Event: completion of request, or external trigger
    - Can be bound to a hardware interrupt

- **Component**:
  - *Provides* a set of interfaces (used by others)
  - *Uses* a set of interfaces (provided by others)

- An application "wire s”t ogether" the interfaces of a set of components
**Blink: An Example TinyOS Appln.**

- Split into Blink.nc & BlinkM.nc
- **BlinkM.nc:**
  - The *module*: the actual implementation
- **Blink.nc:**
  - The *configuration*: the “wiring-up” of interfaces
- Other examples: CountLeds, CountSend, CountReceive