Mobile and Sensor Systems

Lecture 6:
Sensor Systems and Mac Layer Routing

Dr. Cecilia Mascolo

UNIVERSITY OF CAMBRIDGE

What's in this lecture

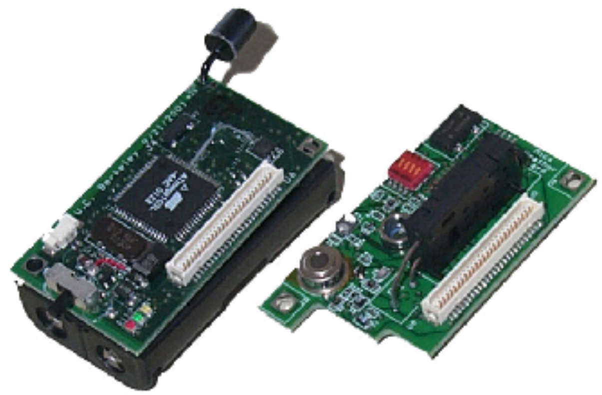
- We will describe sensor networks in general and the properties of sensor nodes
- We will introduce sensor network MAC Layer issues and some solutions.

Sensors and Sensor Networks

A sensor is a device which allows “sensing” of the environment. It is usually small and resource constrained.

A **sensor network** is composed of a large number of sensor nodes, which are deployed either inside the phenomenon or very close to it.

- Sometimes Random deployment
- Cooperative capabilities



UNIVERSITY OF CAMBRIDGE

Sensor Systems vs Standard or Mobile Systems

- Sensor nodes are limited in power, computational capacities and memory.
- Sensor nodes are prone to failures (especially because they are often deployed in challenging conditions)
- The topology of a sensor network might not change frequently:
  - Many deployment involved sensors with fixed location
  - Some deployments may have mobile sensors

Example applications

Seismic Structure response

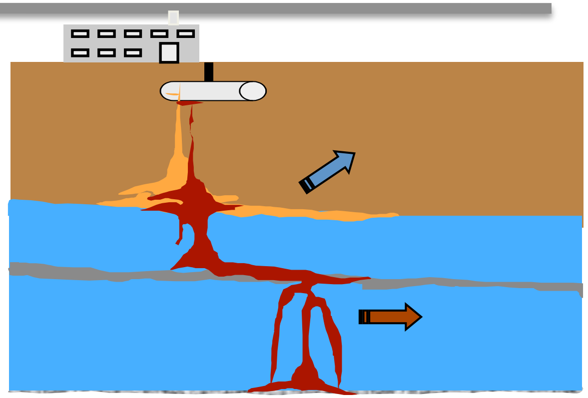
Marine Microorganisms

UNIVERSITY OF CAMBRIDGE

- Micro-sensors, on-board processing, wireless interfaces feasible at very small scale--can monitor phenomena “up close”

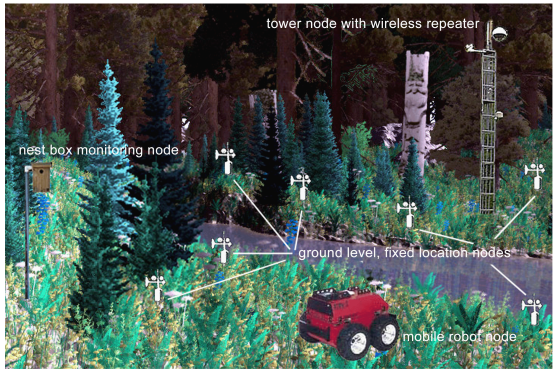
- Enables spatially and temporally dense environmental monitoring

Embedded Networked Sensing will reveal previously unobservable phenomena

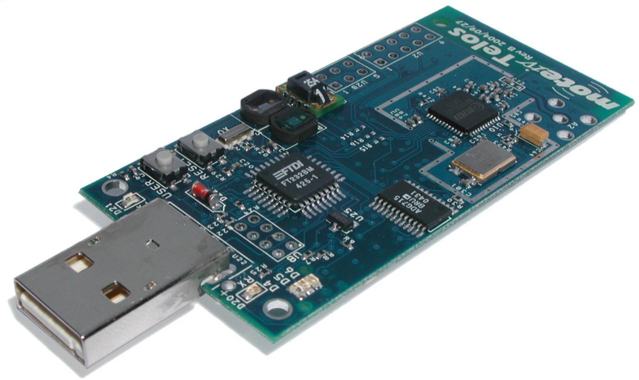


Contaminant Transport

Ecosystems, Biocomplexity

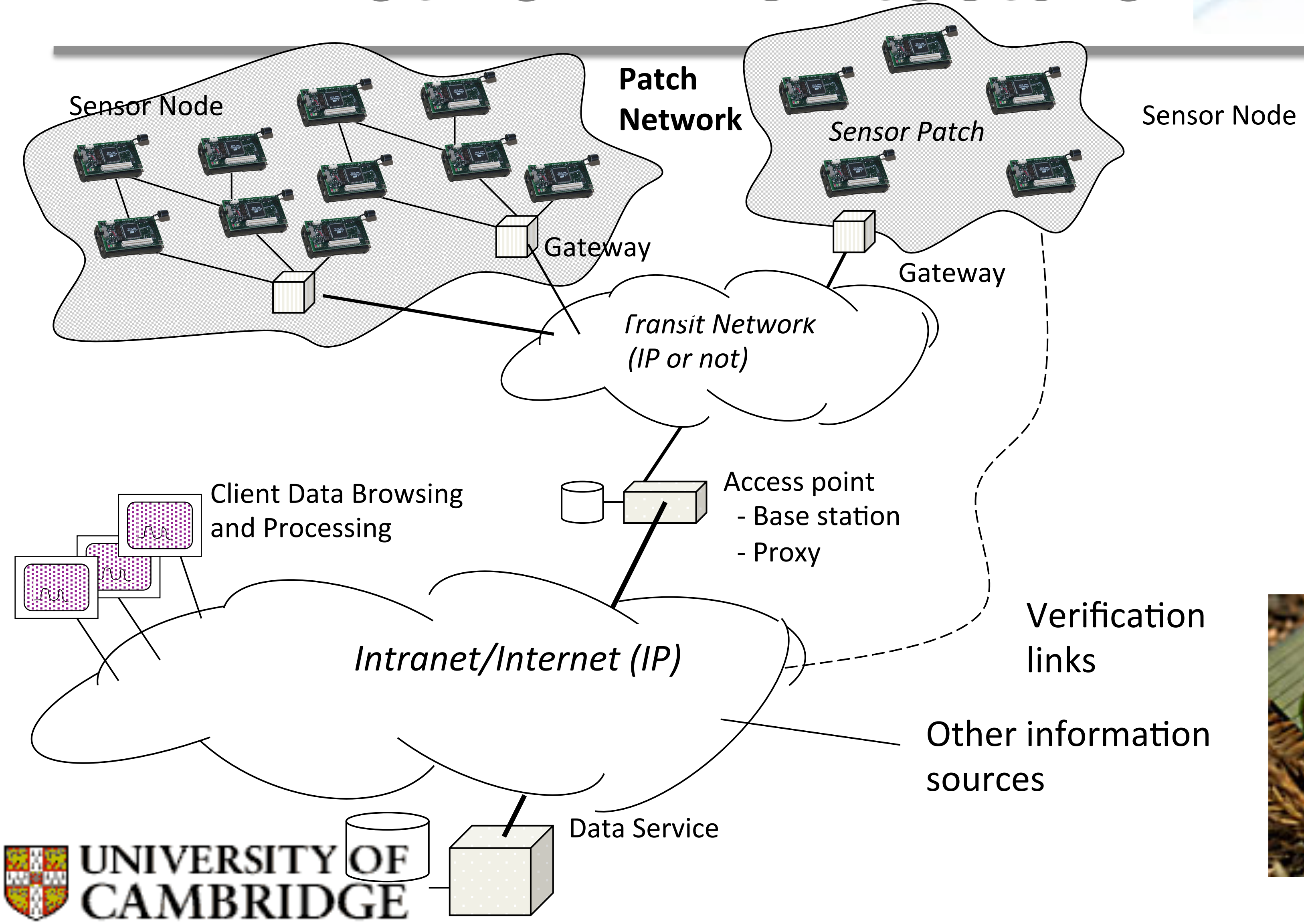


Experimental Platform

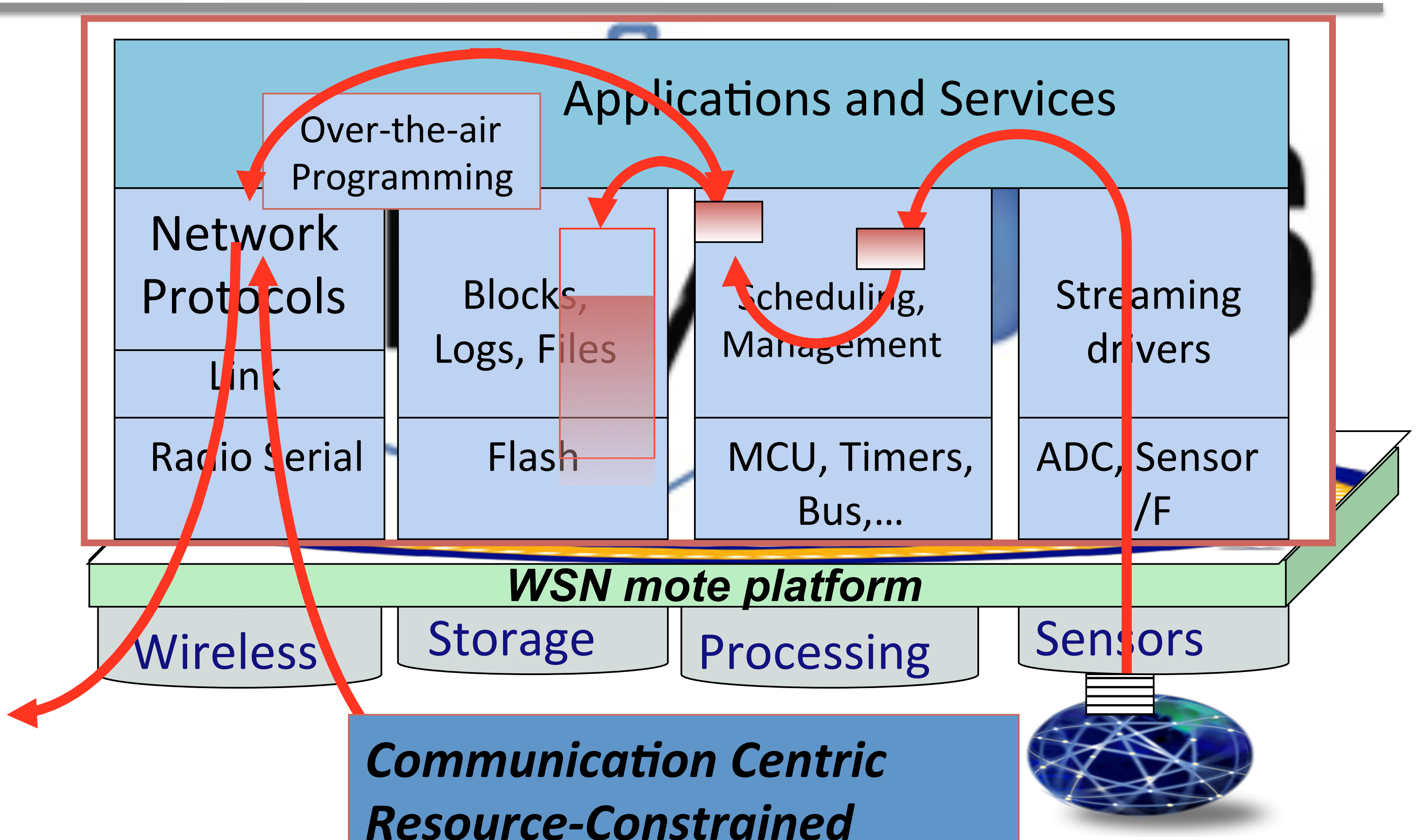


- Standards Based
  - USB
  - Radio:
    - IEEE 802.15.4 (CC2420 radio)
    - Zigbee: Ultralow power
- 8-bit microprocessor, 4MHz CPU
  - ATMEGA 128, ATMEL 8535, or Motorola HCS08
- ~4Kb RAM
  - holds run-time state (values of the variables) of the program
- ~128Kb programmable Flash memory
  - holds the application program
  - Downloaded via a programmer-board or wirelessly
- Additional Flash memory storage space

Canonical SensorNet Network Architecture



What happens in the node



Communication Centric
Resource-Constrained
Event-driven Execution

What Operating System runs on a sensor?

- Operating system useful to simplify programming tasks and to allow more control over operations of the system
- But what can we do with such a constrained device?
- Given the kind of applications needed it is important to support concurrency…[frequent and parallel collection from different sensors]

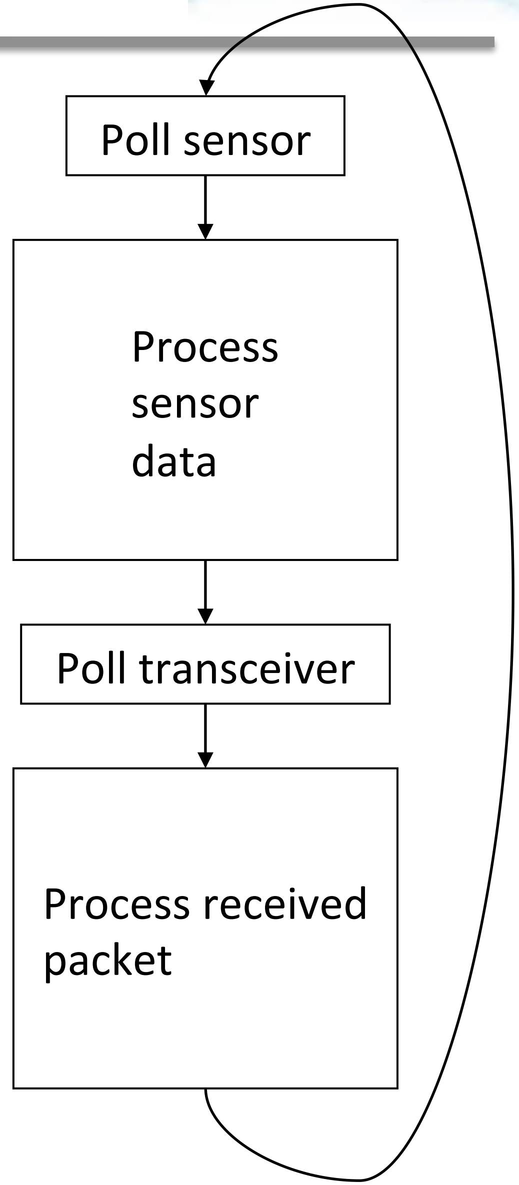
UNIVERSITY OF CAMBRIDGE

Main issue: How to support concurrency

- Simplest option: No concurrency, sequential processing of tasks
  - Not satisfactory: Risk of missing data (e.g., from transceiver) when processing data, etc.

  ! Interrupts/asynchronous operation has to be supported

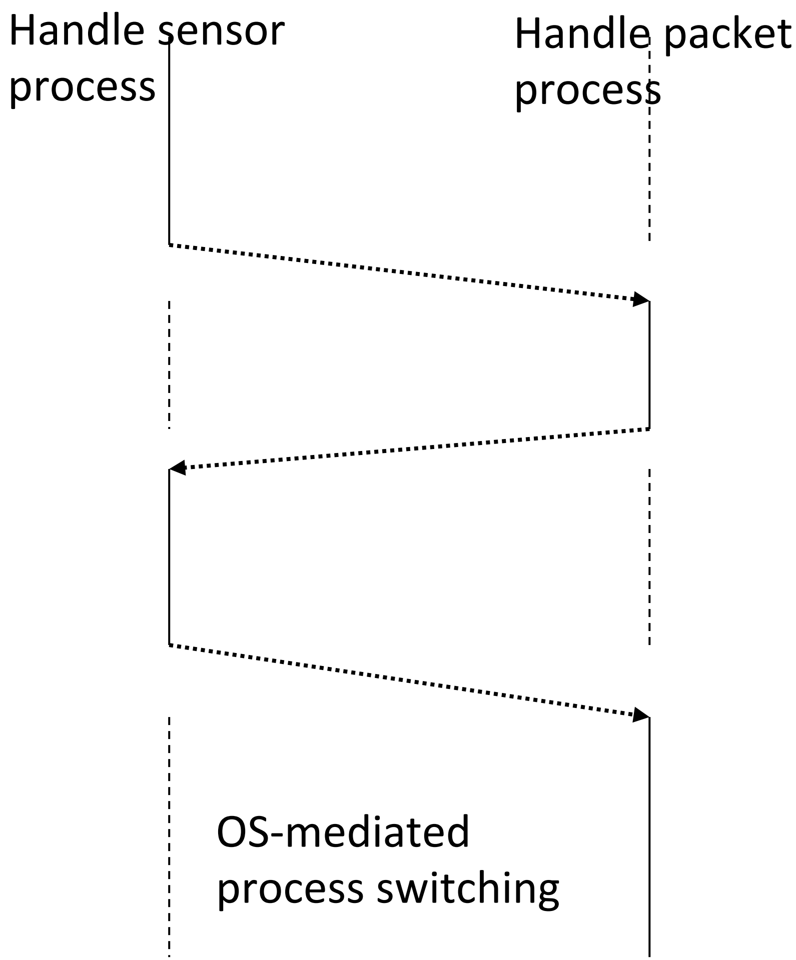
- Why concurrency is needed
  - Sensor node's CPU has to service the radio modem, the actual sensors, perform computation for application, execute communication protocol software, etc.



UNIVERSITY OF CAMBRIDGE

Traditional concurrency: Processes

- Traditional OS: processes/threads
  - Based on interrupts, context switching
  - But: memory overhead, execution overhead
- concurrency mismatch
  - One process per protocol entails too many context switches
  - Many tasks in WSN small with respect to context switching overhead

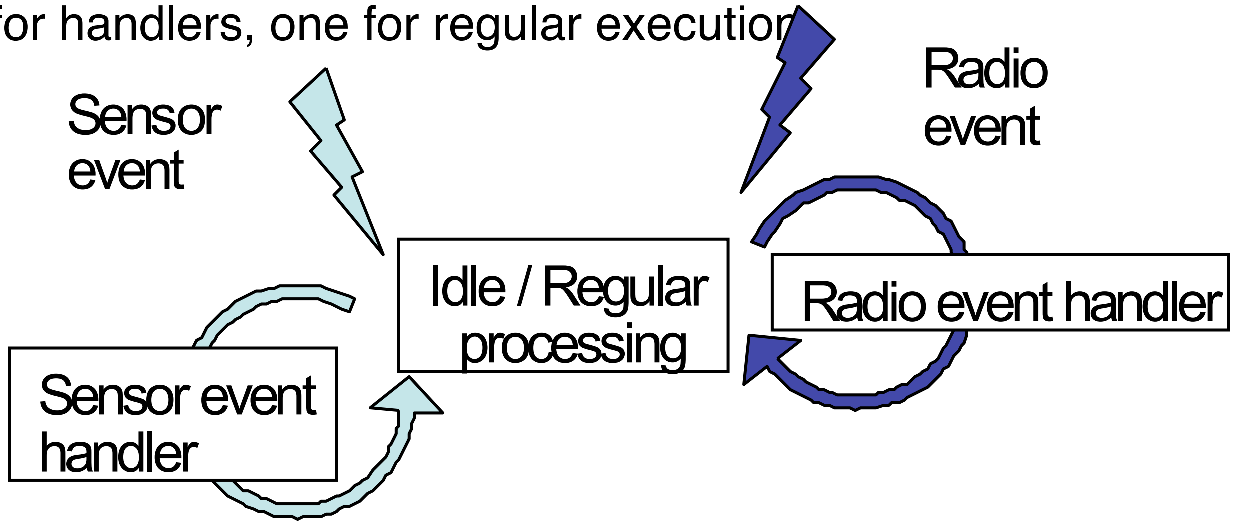


UNIVERSITY OF CAMBRIDGE

Event-based concurrency

- Alternative: Switch to ***event-based programming model***
  - Perform regular processing or be idle
  - React to events when they happen immediately
  - Basically: interrupt handler
- Problem: must not remain in interrupt handler too long
  - Danger of losing events
  - Only save data, post information that event has happened, then return

  ! ***Run-to-completion*** principle
  - Two contexts: one for handlers, one for regular execution



UNIVERSITY OF CAMBRIDGE

TinyOS: Tasks and Command/ Event Handlers

- TinyOS: an OS for sensor networks
- Event handlers must run to completion
  - Must not wait an indeterminate amount of time
  - Only a ***request*** to perform some action
- Tasks, on the other hand, can perform arbitrary, long computation
  - Also have to be run to completion
  - But can be interrupted by handlers

  ! No need for stack management, tasks are atomic with respect to each other

UNIVERSITY OF CAMBRIDGE

Energy Management

- Processing is not the greatest source of energy consumption
- **The main source of energy consumption is the radio**

- Strategy*: limit communication* and
- But also*: idle listening by the radio is expensive*. **Put the radio to sleep when idle**

Radio Duty Cycling

- Basic strategy: switch off the radio of all sensors at specific intervals
  – Very precise synchronization
  – Still probable idle time for sensors which do not communicate

- More refined strategy
  – Wave of switch off time depending on topology
  – Still probably overestimate of the communication needs of some sensors (traffic might be varying across the network)

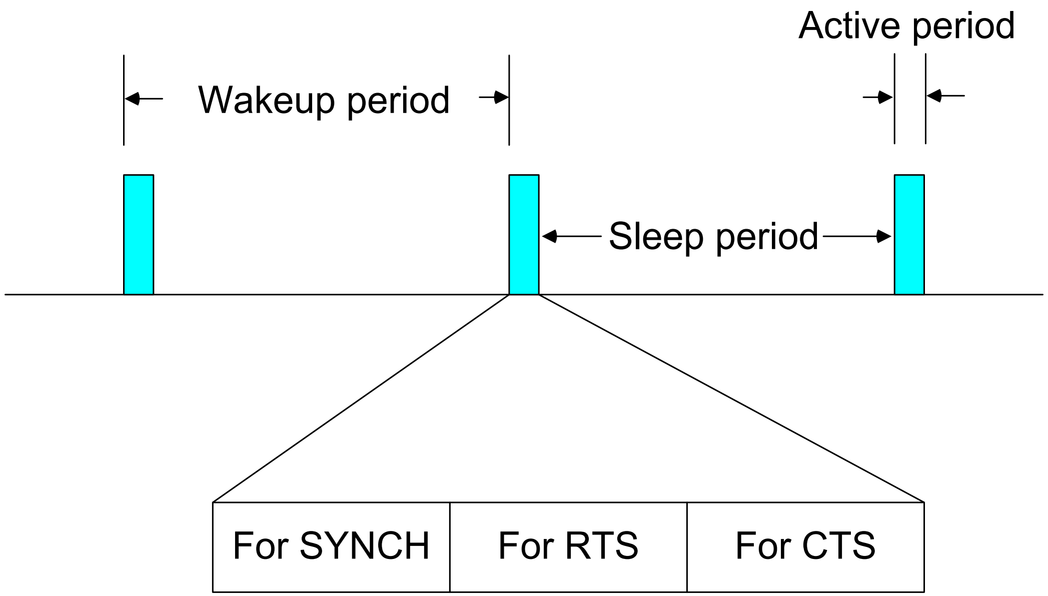
UNIVERSITY OF CAMBRIDGE

Dynamic duty cycling

- More refined strategy have been proposed which aim to allow sensors which transmit more to stay more awake and others to sleep more.
- The basic idea of most of these approaches is to keep nodes awake only when the need to transmit or receive
  - But how is this known?

Sensor-MAC (S-MAC)

- idle listening is particularly unsuitable if average data rate is low
  - Most of the time, nothing happens
- Idea: Switch nodes off, ensure that neighboring nodes turn on simultaneously to allow packet exchange (rendez-vous)
  - Only in these ***active periods***, packet exchanges happen
  - Need to also exchange wakeup schedule between neighbors
  - When awake, essentially perform RTS/CTS
- Use SYNCH, RTS, CTS phases



UNIVERSITY OF CAMBRIDGE

S-MAC

- SYNC phase divided into time slots using CSMA and backoff to send schedule to neighbours
- Y chooses a slot and if no signal was received before it will start to transmit its schedule to X otherwise wait for next wake up of X
- RTS phase: X listens for RTS packets (CSMA contention)
- CTS phase: X sends one and extends its wake up time

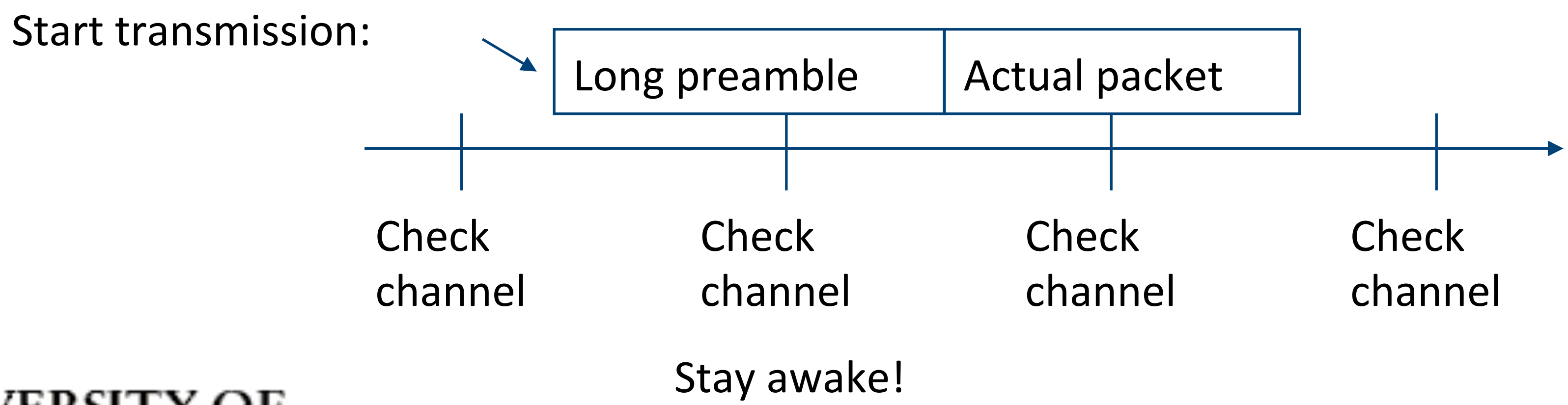
UNIVERSITY OF CAMBRIDGE

S-MAC synchronized islands

- Nodes try to pick up schedule synchronization from neighboring nodes
- If no neighbor found, nodes pick some schedule to start with
- If additional nodes join, some node might learn about two different schedules from different nodes
  - “Synchronized islands”
- To bridge this gap, it has to follow both schemes and use more energy

Preamble Sampling

- So far: Periodic sleeping supported by some means to synchronize wake up of nodes to ensure rendez-vous between sender and receiver
- Alternative option: Don't try to explicitly synchronize nodes
  - Have receiver sleep and only periodically sample the channel
- Use ***long preambles*** to ensure that receiver stays awake to catch actual packet. Example: BMAC and WiseMAC



UNIVERSITY OF CAMBRIDGE

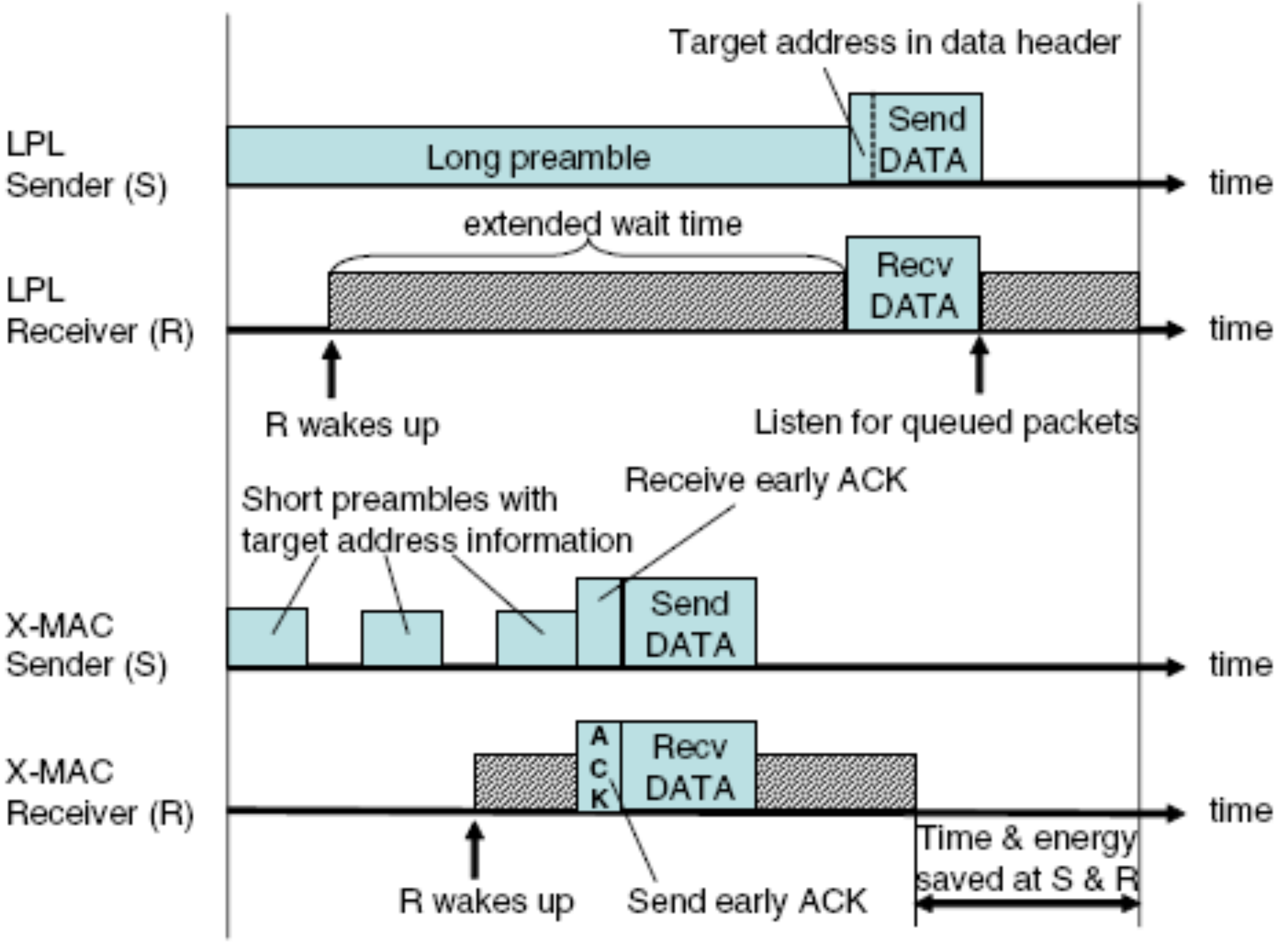
Problems with this technique

- A node waking up while preamble is transmitted needs to stay awake until the end to know if he is the target receiver!
- Also it might be that the receiver is awake much earlier than the end of the preamble
- This means that energy consumption is very dependent on network density

- Can you think of alternative solutions?

UNIVERSITY OF CAMBRIDGE

XMAC



UNIVERSITY OF CAMBRIDGE

XMAC

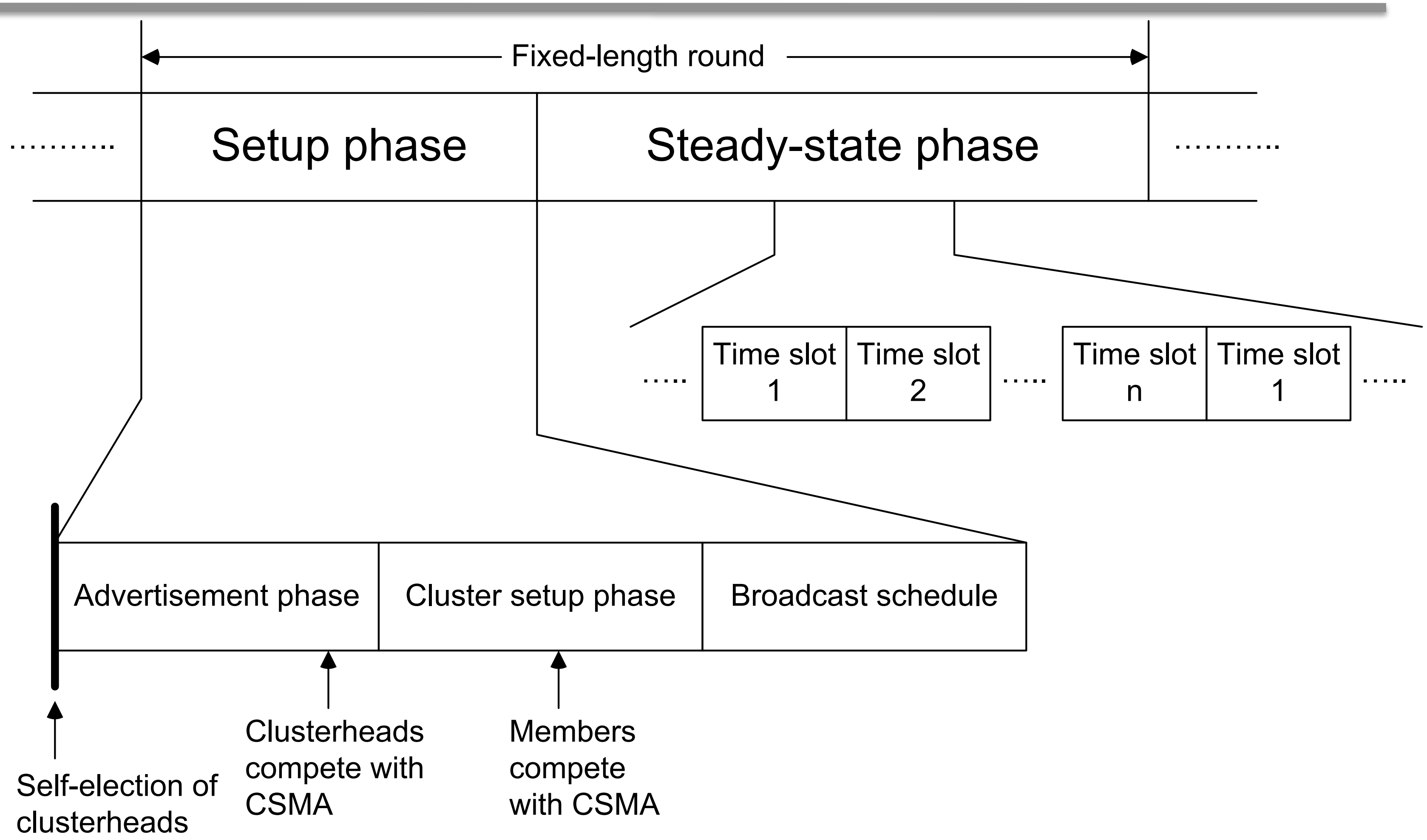
- Short preamble
  - Reduce latency and reduce energy consumption
- Target in preamble
  - Minimize overhearing problem.
- Adding wait time between preambles
  - Reduces latency for the case where destination is awake before preamble completes.

UNIVERSITY OF CAMBRIDGE

Low-Energy Adaptive Clustering Hierarchy (LEACH)

- Given: dense network of nodes, reporting to a central sink, each node can reach sink directly
- Idea: Group nodes into “***clusters***”, controlled by ***clusterhead***
  - Setup phase; details: later
  - About 5% of nodes become clusterhead (depends on scenario)
  - Role of clusterhead is rotated to share the burden
  - Clusterheads advertise themselves, ordinary nodes join CH with strongest signal
  - Clusterheads organize
    - CDMA code for all member transmissions
    - TDMA schedule to be used within a cluster
- In steady state operation
  - CHs collect & aggregate data from all cluster members
  - Report aggregated data to sink using CSMA

LEACH rounds



UNIVERSITY OF CAMBRIDGE

Summary

- We have described sensor nodes properties and sensor nodes operation
- We have introduced various MAC layer sensor network protocols

References

- TinyOS tutorial: http://www.tinyos.net/tinyos-1.x/doc/tutorial/
- SMAC: Ye, W., Heidemann, J., and Estrin, D. 2004. Medium access control with coordinated adaptive sleeping for wireless sensor networks. IEEE/ACM Trans. Netw. 12, 3 (Jun. 2004), 493-506.
- WISEMAC: El-Hoiydi, A. and Decotignie, J. 2004. WiseMAC: an ultra low power MAC protocol for the downlink of infrastructure wireless sensor networks. In Proceedings of the Ninth international Symposium on Computers and Communications 2004 Volume 2 (Iscc"04) - Volume 02 (June 28 - July 01, 2004). ISCC. IEEE Computer Society, Washington, DC, 244-251.
- X-MAC: M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks," in *Proceedings of the 4th international conference on Embedded networked sensor systems* Boulder, Colorado, USA: ACM, 2006.
- LEACH: Wendi Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan, Energy-Efficient Communication Protocols for Wireless Microsensor Networks, Proc. Hawaaian Int'l Conf. on Systems Science, January 2000.